Red Hat Enterprise Linux 8

Identity Management の設定および管理

IdM にログインし、サービス、ユーザー、ホスト、グループ、アクセス制御ルール、および証明書を管理します。

Last Updated: 2024-05-11

Red Hat Enterprise Linux 8 Identity Management の設定および管理

IdM にログインし、サービス、ユーザー、ホスト、グループ、アクセス制御ルール、および証明書を管理します。

法律上の通知

概要

Red Hat Identity Management (IdM) の主な機能は、ユーザー、グループ、ホスト、アクセス制御ルール、および証明書の管理です。ただし、IdM で管理タスクを実行するには、その前にサービスにログインする必要があります。コマンドラインまたは IdM Web UI を使用してログインする場合は、IdM の認証方法として Kerberos およびワンタイムパスワードを使用できます。 統合または外部の認証局 (CA) を使用して、IdM で証明書を管理できます。Ansible Playbook などの多くのツールを使用して、証明書を要求、更新、および置き換えることができます。IdM サーバーの Web サーバー証明書と LDAP サーバー証明書を置き換えるには、手動アクションを実行する必要があります。

目次

多様性を受け入れるオープンソースの強化

Red Hat では、コード、ドキュメント、Web プロパティーにおける配慮に欠ける用語の置き換えに取り組んでいます。まずは、マスター (master)、スレーブ (slave)、ブラックリスト (blacklist)、ホワイトリスト (whitelist) の 4 つの用語の置き換えから始めます。この取り組みは膨大な作業を要するため、今後の複数のリリースで段階的に用語の置き換えを実施して参ります。詳細は、Red Hat CTO である Chris Wright のメッセージ を参照してください。

Identity Management では、次のような用語の置き換えが計画されています。

- **ブラックリスト** から **ブロックリスト**
- **ホワイトリスト** から **許可リスト**
- **スレーブ** から **セカンダリー**
- **マスター** という言葉は、文脈に応じて、より正確な言葉に置き換えられています。
  - **IdM マスター** から **IdM サーバー**
  - **CA 更新マスター** から **CA 更新サーバー**
  - **CRL マスター** から **CRL パブリッシャーサーバー**
  - **マルチマスター** から **マルチサプライヤー**

RED HAT ドキュメントへのフィードバック (英語のみ)

Red Hat ドキュメントに関するご意見やご感想をお寄せください。また、改善点があればお知らせください。

Jira からのフィードバック送信 (アカウントが必要)

1. Jira の Web サイトにログインします。
2. 上部のナビゲーションバーで **Create** をクリックします。
3. **Summary** フィールドにわかりやすいタイトルを入力します。
4. **Description** フィールドに、ドキュメントの改善に関するご意見を記入してください。ドキュメントの該当部分へのリンクも追加してください。
5. ダイアログの下部にある **Create** をクリックします。

第1章 コマンドラインから IDENTITY MANAGEMENT へのログイン

Identity Management (IdM) では、Kerberos プロトコルを使用してシングルサインオンに対応します。シングルサインオンとは、ユーザーが正しいユーザー名およびパスワードを一度だけ入力すれば、システムが認証情報を再度求めることなく、IdM サービスにアクセスできるという機能です。

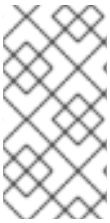
重要

IdM では、ユーザーが、対応する Kerberos プリンシパル名を使用して IdM クライアントマシンのデスクトップ環境にログインすると、SSSD (System Security Services Daemon) が、そのユーザーの TGT (Ticket-Granting Ticket) を自動的に取得します。これは、ログインしてから、**kinit** ユーティリティーを使用して IdM リソースにアクセスする必要がなくなることを意味します。

Kerberos 認証情報キャッシュを削除している場合、または Kerberos TGT の有効期限が切れている場合に IdM リソースにアクセスするには、手動で Kerberos チケットを要求する必要があります。以下のセクションでは、IdM で Kerberos を使用している場合の基本的なユーザー操作を説明します。

1.1. KINIT による IDM への手動ログイン

kinit ユーティリティーを使用して Identity Management (IdM) 環境に対して手動で認証するには、次の手順に従います。**kinit** ユーティリティーは、IdM ユーザーの代わりに Kerberos の TGT (Ticket-Granting Ticket) を取得して、キャッシュに格納します。



注記

この手順は、最初の Kerberos TGT を破棄したか、有効期限が切れている場合にのみ使用します。ローカルマシンに、IdM ユーザーとしてログインすると、IdM に自動的にログインします。これは、ログイン後に IdM リソースにアクセスするのに **kinit** ユーティリティーを使用する必要がないことを示しています。

手順

1. IdM にログインします。

   - ローカルシステムに現在ログインしているユーザーのユーザー名で、(ユーザー名を指定せずに) **kinit** を使用します。たとえば、ローカルシステムにログインしているユーザーが **example_user** の場合は、次のコマンドを実行します。

     ```
     [example_user@server ~]$ kinit
     Password for example_user@EXAMPLE.COM:
     [example_user@server ~]$
     ```

     ローカルユーザーのユーザー名と、IdM のユーザーエントリーが一致しないと、認証に失敗します。

     ```
     [example_user@server ~]$ kinit
     kinit: Client 'example_user@EXAMPLE.COM' not found in Kerberos database while
     getting initial credentials
     ```

- ローカルユーザー名に対応しない Kerberos プリンシパルを使用して、**kinit** ユーティリティーに必要なユーザー名を渡します。たとえば、**admin** ユーザーとしてログインするには、次のコマンドを実行します。

```
[example_user@server ~]$ kinit admin
Password for admin@EXAMPLE.COM:
[example_user@server ~]$
```

2. 必要に応じて、ログインが成功したことを確認するには、**klist** ユーティリティーを使用して、キャッシュした TGT を表示します。以下の例では、キャッシュに **example_user** プリンシパルのチケットが含まれています。これは、このホストでは IdM サービスにアクセスするのは、**example_user** にのみ許可されていることを示しています。

```
$ klist
Ticket cache: KEYRING:persistent:0:0
Default principal: example_user@EXAMPLE.COM

Valid starting       Expires              Service principal
11/10/2019 08:35:45  11/10/2019 18:35:45  krbtgt/EXAMPLE.COM@EXAMPLE.COM
```

1.2. アクティブなユーザーの KERBEROS チケットの破棄

ユーザーのアクティブな Kerberos チケットを含む認証情報キャッシュをクリアするには、次の手順に従います。

手順

1. Kerberos チケットを破棄するには、次のコマンドを実行します。

```
[example_user@server ~]$ kdestroy
```

2. 必要に応じて、Kerberos チケットが破棄されたことを確認するには、次のコマンドを実行します。

```
[example_user@server ~]$ klist
klist: Credentials cache keyring 'persistent:0:0' not found
```

1.3. KERBEROS 認証用の外部システムの設定

Identity Management (IdM) ユーザーが Kerberos 認証情報を使用して外部システムから IdM にログインできるように外部システムを設定するには、この手順に従います。

外部システムの Kerberos 認証を有効にすることは、インフラストラクチャーに、複数のレルムまたは重複ドメインが含まれている場合に特に便利です。また、**ipa-client-install** を実行してシステムを IdM ドメインに登録していない場合にも便利です。

IdM ドメインのメンバーではないシステムから IdM への Kerberos 認証を有効にするには、IdM 固有の Kerberos 設定ファイルを外部システムに定義します。

前提条件

- 外部システムに **krb5-workstation** パッケージがインストールされている。

パッケージがインストールされているかどうかを確認するには、次の CLI コマンドを使用します。

```
# yum list installed krb5-workstation
Installed Packages
krb5-workstation.x86_64    1.16.1-19.el8    @BaseOS
```

手順

1. IdM サーバーから外部システムに **/etc/krb5.conf** ファイルをコピーします。以下に例を示します。

```
# scp /etc/krb5.conf root@externalsystem.example.com:/etc/krb5_ipa.conf
```

警告

外部マシンにある既存の **krb5.conf** ファイルは上書きしないでください。

2. 外部システムで、コピーした IdM の Kerberos 設定ファイルを使用するように、端末セッションを設定します。

```
$ export KRB5_CONFIG=/etc/krb5_ipa.conf
```

KRB5_CONFIG 変数は、ログアウトまで一時的に存在します。ログアウト時に削除されないように、この変数のファイル名を変えてエクスポートします。

3. **/etc/krb5.conf.d/** ディレクトリーの Kerberos 設定部分を、外部システムにコピーします。

外部システムのユーザーが、**kinit** ユーティリティーを使用して IdM サーバーで認証できるようになりました。

1.4. 関連情報

- **krb5.conf(5)** の man ページ
- **kinit(1)** の man ページ
- **klist(1)** の man ページ
- **kdestroy(1)** の man ページ

第2章 IDENTITY MANAGEMENT サービスの表示、開始、および停止

Identity Management (IdM) サーバーは、ドメインコントローラー (DC) として機能する Red Hat Enterprise Linux システムです。IdM サーバーでさまざまなサービスが実行していますが、中でも注目すべきは Directory Server、Certificate Authority (CA)、DNS、および Kerberos です。

2.1. IDM サービス

IdM サーバーおよびクライアントにインストールして実行できるサービスには、さまざまなものがあります。

IdM サーバーがホストするサービスのリスト

以下のサービスの多くは、IdM サーバーへのインストールが必須というわけではありません。たとえば、認証局 (CA) や DNS サーバーなどのサービスは、IdM ドメイン内にない外部サーバーにインストールできます。

Kerberos

krb5kdc サービスおよび **kadmin** サービス

IdM は、シングルサインオンに対応する **Kerberos** プロトコルを使用します。Kerberos では、正しいユーザー名とパスワードを一度提示するだけで済み、システムから認証情報を再度求められることなく IdM サービスにアクセスできます。

Kerberos は 2 つの部分に分類されます。

- **krb5kdc** サービス。Kerberos 認証サービスおよびキー配布センター (KDC) デーモンです。
- **kadmin** サービス。Kerberos V5 データベース管理プログラムです。

IdM で Kerberos を使用して認証する方法は、コマンドラインからの Identity Management へのログイン および Web UI で IdM にログイン: Kerberos チケットの使用 を参照してください。

LDAP ディレクトリーサーバー

dirsrv サービス

IdM の **LDAP ディレクトリーサーバー** インスタンスは、Kerberos、ユーザーアカウント、ホストエントリー、サービス、ポリシー、DNS などの情報をはじめとした、IdM 情報をすべて保存します。LDAP ディレクトリーサーバーインスタンスは、Red Hat Directory Server と同じテクノロジーをベースにしています。ただし、IdM 固有のタスクに合わせて調整されます。

認証局

pki-tomcatd サービス

統合 **認証局 (CA)** は、Red Hat Certificate System と同じテクノロジーをベースにしています。**pki** は、Certificate System サービスにアクセスするコマンドラインインターフェイスです。

必要な証明書をすべて単独で作成して提供する場合は、統合 CA なしでサーバーをインストールすることもできます。

詳細は、CA サービスの計画 を参照してください。

DNS (Domain Name System)

named サービス

IdM は、動的サービス検出に **DNS** を使用します。IdM クライアントのインストールユーティリティーは、DNS からの情報を使用して、クライアントマシンを自動的に設定できます。クライアントを IdM ドメインに登録したら、クライアントは DNS を使用してドメイン内の IdM サーバーおよびサービスを検索します。Red Hat Enterprise Linux の DNS (Domain Name System) プロトコルの **BIND** (Berkeley Internet Name Domain) 実装には、**名前付き** の DNS サーバーが含まれています。 **named-pkcs11** は、PKCS#11 暗号化標準に対するネイティブサポートありで構築された BIND DNS サーバーのバージョンです。

詳細は、Planning your DNS services and host names を参照してください。

Apache HTTP サーバー

httpd サービス

Apache HTTP Web サーバー には、IdM Web UI があり、認証局とその他の IdM サービスの間の通信も管理します。

Samba / Winbind

SMB サービスおよび **winbind** サービス

Samba は、Red Hat Enterprise Linux に、Common Internet File System (CIFS) プロトコルとも呼ばれる Server Message Block (SMB) プロトコルを実装します。smb サービス経由で SMB プロトコルを使用すると、ファイル共有や共有プリンターなどのサーバーのリソースにアクセスできます。Active Directory (AD) 環境で信頼を設定している場合には、'Winbind' サービスが IdM サーバーと AD サーバー間の通信を管理します。

ワンタイムパスワード (OTP) 認証

ipa-otpd サービス

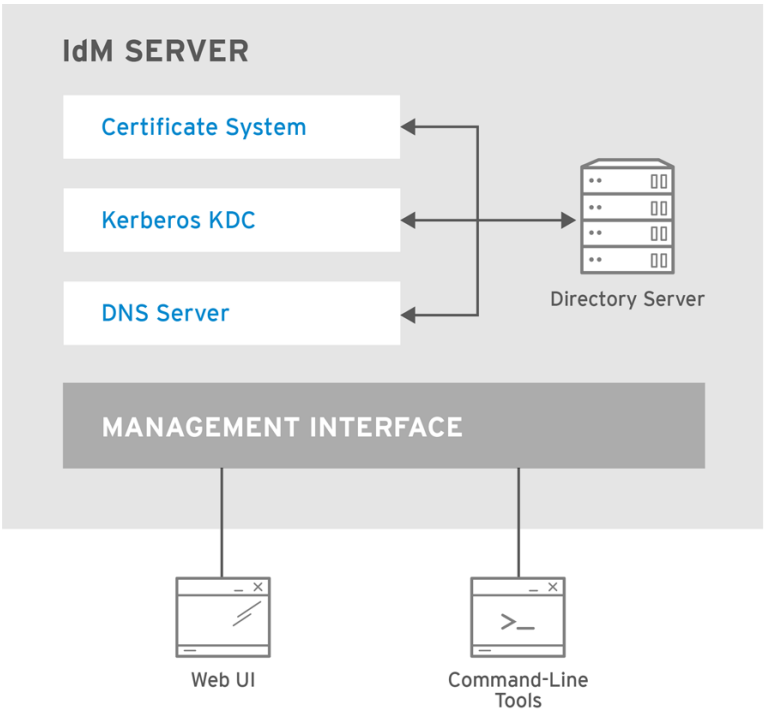
ワンタイムパスワード (OTP) は、2 要素認証の一部として、認証トークンがセッション 1 回だけ使用できるように生成するパスワードです。OTP 認証は、**ipa-otpd** サービスを介して Red Hat Enterprise Linux に実装されています。

詳細は ワンタイムパスワードを使用して Identity Management Web UI へのログイン を参照してください。

OpenDNSSEC

ipa-dnskeysyncd サービス

OpenDNSSEC は、DNSSEC (DNS Security Extensions) キーおよびゾーンの署名の記録プロセスを自動化する DNS マネージャーです。**ipa-dnskeysyncd** サービスは、IdM Directory Server と OpenDNSSEC との間の同期を管理します。



IdM クライアントがホストするサービスのリスト

- System Security Services Daemon: **sssd** サービス

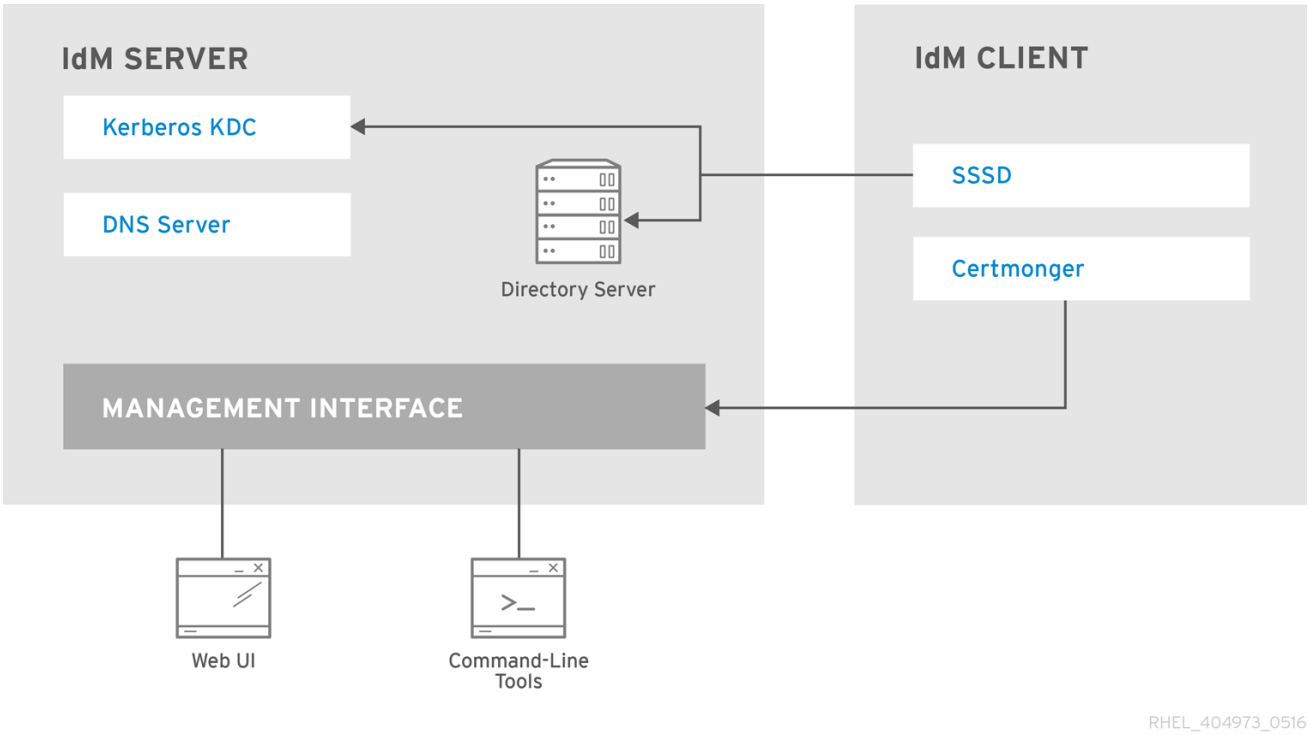
SSSD (**System Security Services Daemon**) は、ユーザー認証およびキャッシュ認証情報を管理するクライアント側のアプリケーションです。キャッシュを使用すると、IdM サーバーが利用できなくなったり、クライアントがオフラインになったりした場合に、ローカルシステムが通常の認証操作を継続できるようになります。

詳細は SSSD とその利点について を参照してください。

- certmonger: **certmonger** サービス

certmonger サービスは、クライアント上の証明書を監視、更新します。このサービスは、システム上のサービスに対して新しい証明書を要求できます。

詳細は、certmonger を使用したサービスの IdM 証明書の取得 を参照してください。



2.2. IDM サービスの状態の表示

IdM サーバーに設定されている IdM サービスの状態を表示するには、**ipactl status** コマンドを実行します。

```
[root@server ~]# ipactl status
Directory Service: RUNNING
krb5kdc Service: RUNNING
kadmin Service: RUNNING
named Service: RUNNING
httpd Service: RUNNING
pki-tomcatd Service: RUNNING
smb Service: RUNNING
winbind Service: RUNNING
ipa-otpd Service: RUNNING
ipa-dnskeysyncd Service: RUNNING
ipa: INFO: The ipactl command was successful
```

サーバーの **ipactl status** コマンドの出力は、IdM 設定により異なります。たとえば、IdM デプロイメントに DNS サーバーが含まれていない場合は、**named** サービスがリストに表示されません。

注記

IdM の Web UI を使用して、特定の IdM サーバーで実行しているすべての IdM サービスの状態を表示することはできません。Kerberos に対応し、複数のサーバーで実行しているサービスは、IdM の Web UI の **Identity → Services** タブで表示できます。

サーバー全体、または個々のサービスのみを起動または停止できます。

IdM サーバー全体を起動、停止、または再起動する場合は、以下を参照してください。

- Identity Management サーバー全体の起動と停止

個々の IdM サービスを起動、停止、または再起動する場合は、以下を参照してください。

- 個々の Identity Management サービスの開始および停止

IdM ソフトウェアのバージョンを表示するには、次を参照してください。

- IdM ソフトウェアのバージョンを表示する方法

2.3. IDENTITY MANAGEMENT サーバー全体の起動と停止

ipa systemd サービスを使用して、IdM サーバー全体を、インストールしたすべてのサービスを停止、起動、または再起動します。**systemctl** ユーティリティーを使用して **ipa** systemd サービスを制御すると、すべてのサービスが適切な順序で停止、開始、または再起動されます。**ipa** systemd サービスは、IdM サービスを起動する前に RHEL IdM 設定もアップグレードし、IdM サービスの管理時に適切な SELinux コンテキストを使用します。**systemctl ipa** コマンドを実行するには、有効な Kerberos チケットは必要ありません。

ipa systemd service コマンド

IdM サーバー全体を起動するには、次のコマンドを実行します。

```
# systemctl start ipa
```

IdM サーバー全体を停止するには、次のコマンドを実行します。

```
# systemctl stop ipa
```

IdM サーバー全体を再起動するには、次のコマンドを実行します。

```
# systemctl restart ipa
```

IdM を設定するすべてのサービスのステータスを表示するには、**ipactl** ユーティリティーを使用します。

```
# ipactl status
```

重要

- IdM サービスは、**ipactl** ユーティリティーで、起動、停止、または再起動しないでください。代わりに **systemctl ipa** コマンドを使用して、予測可能な環境で **ipactl** ユーティリティーを呼び出します。
- **ipactl** コマンドは、IdM の Web UI では使用できません。

2.4. 個々の IDENTITY MANAGEMENT サービスの開始および停止

IdM 設定ファイルを手動で変更することは推奨されていません。ただし、特定の状況では、管理者が特定のサービスを手動で設定する必要があります。このような場合は、**systemctl** ユーティリティーを使用して、個々の IdM サービスを停止、開始、または再開します。

たとえば、その他の IdM サービスを変更せずに、Directory Server の挙動をカスタマイズした場合は、**systemctl** を使用します。

```
# systemctl restart dirsrv@REALM-NAME.service
```

また、Active Directory と IdM の信頼を最初にデプロイする場合は、**/etc/sssd/sssd.conf** ファイルを変更して、以下を追加します。

- リモートサーバーのレイテンシーが長い環境で、タイムアウト設定オプションを調整するための特定のパラメーター
- Active Directory サイトのアフィニティーを調整するための特定のパラメーター
- グローバルの IdM 設定では提供されない特定の設定オプションのオーバーライド

/etc/sssd/sssd.conf ファイルに加えた変更を適用する場合は、次のコマンドを実行します。

```
# systemctl restart sssd.service
```

System Security Services Daemon (SSSD) は、設定を自動的に再読み込みまたは再適用しないため、**systemctl restart sssd.service** を実行する必要があります。

変更が、IdM の ID 範囲に影響を及ぼす場合は、サーバーを完全に再起動することが推奨されます。

重要

複数の IdM ドメインサービスを再起動するには、常に **systemctl restart ipa** を使用します。IdM サーバーにインストールされているサービス間での依存関係により、サービスを開始および停止する順番は極めて重要です。**ipa** systemd サービスは、サービスが適切な順序で開始および停止されるようにします。

便利な systemctl コマンド

特定の IdM サービスを開始するには、次のコマンドを実行します。

```
# systemctl start name.service
```

特定の IdM サービスを停止するには、次のコマンドを実行します。

```
# systemctl stop name.service
```

特定の IdM サービスを再開するには、次のコマンドを実行します。

```
# systemctl restart name.service
```

特定の IdM サービスの状態を表示するには、次のコマンドを実行します。

```
# systemctl status name.service
```

重要

IdM の Web UI を使用して、IdM サーバーで実行している個々のサービスを開始または停止することはできません。Web UI で可能なのは、**Identity → Services** に移動してサービスを選択し、Kerberos に対応する設定を修正することです。

関連情報

- Identity Management サーバー全体の起動と停止

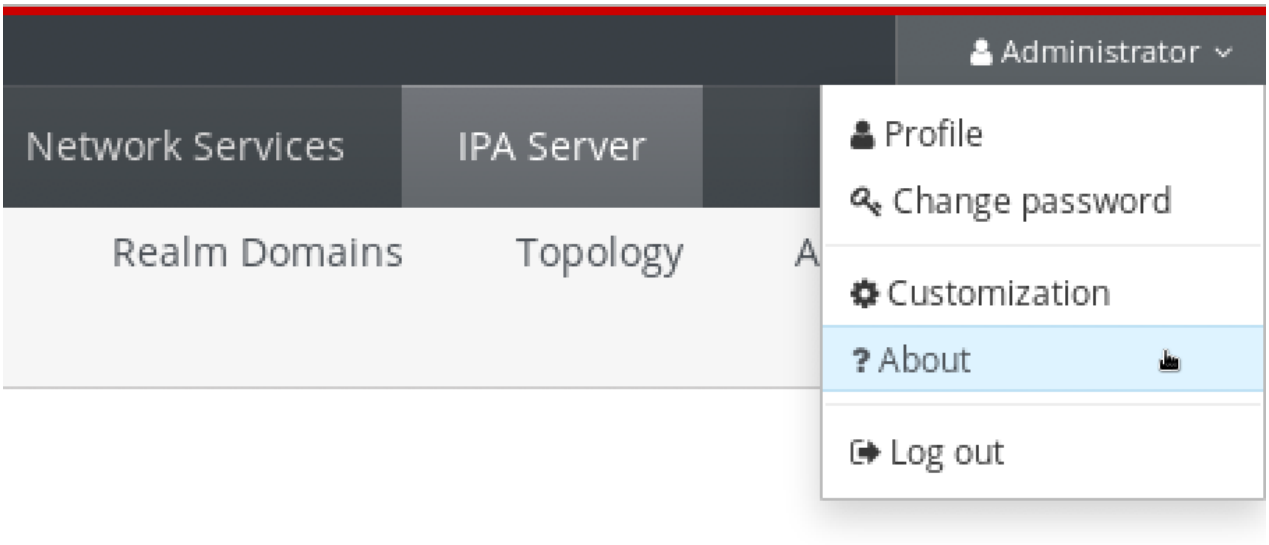
2.5. IDM ソフトウェアのバージョンを表示する方法

IdM バージョン番号は次の方法で表示できます。

- IdM WebUI
- **ipa** コマンド
- **rpm** コマンド

WebUI を介したバージョンの表示

IdM WebUI では、右上のユーザー名メニューから **About** を選択して、ソフトウェアバージョンを表示できます。



ipa コマンドによるバージョンの表示

コマンドラインから、**ipa --version** コマンドを使用します。

```
[root@server ~]# ipa --version
VERSION: 4.8.0, API_VERSION: 2.233
```

rpm コマンドによるバージョンの表示

IdM サービスが適切に動作していない場合は、**rpm** ユーティリティーを使用して、現在インストールされている **ipa-server** パッケージのバージョン番号を確認できます。

```
[root@server ~]# rpm -q ipa-server
ipa-server-4.8.0-11.module+el8.1.0+4247+9f3fd721.x86_64
```

第3章 IDM コマンドラインユーティリティーの概要

Identity Management (IdM) コマンドラインユーティリティーの基本的な使用方法を説明します。

前提条件

- IdM サーバーをインストールしていて、アクセス可能である。
  詳細は、Identity Management のインストール を参照してください。
- IPA コマンドラインインターフェイスを使用する場合は、有効な Kerberos チケットを使用して IdM に対してを認証している。
  有効な Kerberos チケットを取得する方法は、コマンドラインから Identity Management へのログイン を参照してください。

3.1. IPA コマンドラインインターフェイスとは

IPA コマンドラインインターフェイス (CLI) は、Identity Management (IdM) の管理向けの基本的なコマンドラインインターフェイスです。

新しいユーザーを追加するための **ipa user-add** コマンドなど、IdM を管理するための多くのサブコマンドがサポートされています。

IPA CLI では以下を行うことができます。

- ネットワーク内のユーザー、グループ、ホスト、その他のオブジェクトを追加、管理、または削除する。
- 証明書を管理する。
- エントリーを検索する。
- オブジェクトを表示し、オブジェクトリストを表示する。
- アクセス権を設定する。
- 正しいコマンド構文でヘルプを取得する。

3.2. IPA のヘルプとは

IPA ヘルプは、IdM サーバー用の組み込みドキュメントシステムです。

IPA コマンドラインインターフェイス (CLI) は、読み込んだ IdM プラグインモジュールから、利用可能なヘルプトピックを生成します。IPA ヘルプユーティリティーを使用するには、以下が必要です。

- IdM サーバーがインストールされ、実行している。
- 有効な Kerberos チケットで認証されている。

オプションを指定せずに **ipa help** コマンドを実行すると、基本的なヘルプの使用方法と、最も一般的なコマンドの例が表示されます。

さまざまな **ipa help** のユースケースに対して、次のオプションを使用できます。

```
$ ipa help [TOPIC | COMMAND | topics | commands]
```

- **[]** - 括弧は、すべてのパラメーターが任意であることを示しており、**ipa help** のみを入力すれば、コマンドが実行できます。
- | - パイプ文字は **または** の意味になります。したがって、基本的な **ipa help** コマンドを使用して、**TOPIC**、**COMMAND**、**topics** または **commands** を指定できます。
  - **topics** – コマンド **ipa help topics** を実行して、IPA ヘルプでカバーされている **user**、**cert**、**server** などのトピックのリストを表示できます。
  - **TOPIC** – 大文字の TOPIC は変数になります。したがって、特定のトピック (**ipa help user** など) を指定できます。
  - **commands** – コマンド **ipa help commands** を入力して、**user-add**、**ca-enable**、**server-show** などの IPA ヘルプでカバーされているコマンドのリストを表示できます。
  - **COMMAND** – 大文字の COMMAND は変数になります。したがって、**ipa help user-add** などの特定のコマンドを指定できます。

3.3. IPA ヘルプトピックの使用

次の手順では、コマンドラインインターフェイスで IPA ヘルプを使用する方法について説明します。

手順

1. 端末を開き、IdM サーバーに接続します。
2. ヘルプに記載されているトピックのリストを表示するには、**ipa help topics** を実行します。

```
$ ipa help topics
```

3. トピックの 1 つを選択し、**ipa help [topic_name]** のパターンに従ってコマンドを作成します。**topic_name** 文字列の代わりに、前の手順でリストしたトピックの 1 つを追加します。この例では、**user** トピックを使用します。

```
$ ipa help user
```

4. IPA ヘルプの出力が長すぎるため、テキスト全体を表示できない場合は、以下の構文を使用します。

```
$ ipa help user | less
```

スクロールダウンすれば、ヘルプ全体を表示できます

IPA CLI は、**ユーザー** トピックのヘルプページを表示します。概要を読むと、トピックのコマンドを使用するパターンに関して、多くの例を確認できます。

3.4. IPA HELP コマンドの使用

次の手順では、コマンドラインインターフェイスで IPA help コマンドを作成する方法について説明します。

手順

1. 端末を開き、IdM サーバーに接続します。

2. ヘルプで使用できるコマンドのリストを表示するには、**ipa help commands** コマンドを実行します。

```
$ ipa help commands
```

3. コマンドの 1 つを選択し、**ipa help <COMMAND>** のパターンに従ってヘルプコマンドを作成します。**<COMMAND>** 文字列の代わりに、前の手順でリストしたコマンドの 1 つを追加します。

```
$ ipa help user-add
```

関連情報

- **ipa** の man ページ

3.5. IPA コマンドの構造

IPA CLI は、以下のタイプのコマンドを区別します。

- **組み込みコマンド** – 組み込みコマンドはすべて、IdM サーバーで利用できます。
- **プラグインにより提供されたコマンド**

IPA コマンドの構造を使用すると、さまざまなタイプのオブジェクトを管理できます。以下に例を示します。

- ユーザー
- ホスト
- DNS レコード
- 証明書

その他にも多数あります。

このようなほとんどのオブジェクトでは、IPA CLI に、以下を行うためのコマンドが含まれます。

- 追加 (**add**)
- 修正 (**mod**)
- 削除 (**del**)
- 検索 (**find**)
- 表示 (**show**)

コマンドの構造は次のとおりです。

ipa user-add、**ipa user-mod**、**ipa user-del**、**ipa user-find**、**ipa user-show**

ipa host-add、**ipa host-mod**、**ipa host-del**、**ipa host-find**、**ipa host-show**

ipa dnsrecord-add、**ipa dnsrecord-mod**、**ipa dnsrecord-del**、**ipa dnsrecord-find**、**ipa dnrecord-show**

ipa user-add [options] でユーザーを作成できます。**[options]** は任意です。**ipa user-add** コマンドのみを使用する場合、スクリプトは、詳細を 1 つずつ要求します。

既存のオブジェクトを変更するには、オブジェクトを定義する必要があります。そのため、コマンドには、オブジェクト **ipa user-mod USER_NAME [options]** も含まれます。

3.6. IPA コマンドを使用した IDM へのユーザーアカウントの追加

以下の手順では、コマンドラインを使用して Identity Management (IdM) データベースに新しいユーザーを追加する方法について説明します。

前提条件

- IdM サーバーにユーザーアカウントを追加するには、管理者権限が必要です。

手順

1. 端末を開き、IdM サーバーに接続します。
2. 新しいユーザーを追加するコマンドを入力します。

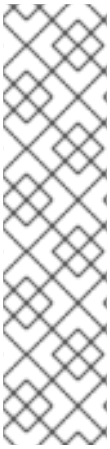
   ```
   $ ipa user-add
   ```

   このコマンドは、ユーザーアカウントの作成に必要な基本データの提供を求めるスクリプトを実行します。
3. **First name:** フィールドに、新規ユーザーの名前を入力して、**Enter** キーを押します。
4. **Last name:** フィールドに、新規ユーザーの苗字を入力し、**Enter** キーを押します。
5. **User login [suggested user name]:**にユーザー名を入力します。または、提案されたユーザー名を使用する場合は、**Enter** キーを押します。
   ユーザー名は、IdM データベース全体で一意にする必要があります。そのユーザー名がすでに存在するためにエラーが発生した場合は、**ipa user-add** コマンドでそのプロセスを再度実行し、別の一意のユーザー名を使用します。

ユーザー名を追加すると、ユーザーアカウントが IdM データベースに追加され、IPA コマンドラインインターフェイス (CLI) は以下の出力を出力します。

```
----------------------
Added user "euser"
----------------------
User login: euser
First name: Example
Last name: User
Full name: Example User
Display name: Example User
Initials: EU
Home directory: /home/euser
GECOS: Example User
Login shell: /bin/sh
Principal name: euser@IDM.EXAMPLE.COM
Principal alias: euser@IDM.EXAMPLE.COM
Email address: euser@idm.example.com
UID: 427200006
```

```
GID: 427200006
Password: False
Member of groups: ipausers
Kerberos keys available: False
```



注記

デフォルトでは、ユーザーアカウントにユーザーパスワードは設定されていません。ユーザーアカウントの作成中にパスワードを追加するには、次の構文で **ipa user-add** コマンドを使用します。

```
$ ipa user-add --first=Example --last=User --password
```

次に、IPA CLI は、ユーザー名とパスワードを追加または確認するように要求します。

ユーザーがすでに作成されている場合は、**ipa user-mod** コマンドでパスワードを追加できます。

関連情報

- パラメーターの詳細は、**ipa help user-add** コマンドを実行してください。

3.7. IPA コマンドで IDM のユーザーアカウントの変更

各ユーザーアカウントの多くのパラメーターを変更できます。たとえば、新しいパスワードをユーザーに追加できます。

基本的なコマンド構文は **user-add** 構文とは異なります。たとえば、パスワードを追加するなど、変更を実行する既存のユーザーアカウントを定義する必要があるためです。

前提条件

- ユーザーアカウントを変更するには、管理者権限が必要です。

手順

1. 端末を開き、IdM サーバーに接続します。
2. **ipa user-mod** コマンドを入力し、変更するユーザーと、パスワードを追加するための **--password** などのオプションを指定します。

   ```
   $ ipa user-mod euser --password
   ```

   このコマンドは、新しいパスワードを追加できるスクリプトを実行します。
3. 新しいパスワードを入力し、**Enter** キーを押します。

IPA CLI は次の出力を出力します。

```
----------------------
Modified user "euser"
----------------------
User login: euser
First name: Example
```

```
Last name: User
Home directory: /home/euser
Principal name: euser@IDM.EXAMPLE.COM
Principal alias: euser@IDM.EXAMPLE.COM
Email address: euser@idm.example.com
UID: 427200006
GID: 427200006
Password: True
Member of groups: ipausers
Kerberos keys available: True
```

これでユーザーパスワードがアカウントに対して設定され、ユーザーが IdM にログインできます。

関連情報

- パラメーターの詳細は、**ipa help user-mod** コマンドを実行してください。

3.8. IDM ユーティリティーに値をリスト形式で提供する方法

Identity Management (IdM) は、多値属性の値をリスト形式で保存します。

IdM は、多値リストを提供する次の方法に対応します。

- 同じコマンド呼び出しで、同じコマンドライン引数を複数回指定します。

```
$ ipa permission-add --right=read --permissions=write --permissions=delete ...
```

- または、リストを中括弧で囲むこともできます。この場合、シェルはデプロイメントを実行します。

```
$ ipa permission-add --right={read,write,delete} ...
```

上記の例では、パーミッションをオブジェクトに追加する **permission-add** コマンドを表示します。この例では、このオブジェクトについては触れていません。**...** の代わりに、権限を追加するオブジェクトーを追加する必要があります。

このような多値属性をコマンド行から更新すると、IdM は、前の値リストを新しいリストで完全に上書きします。したがって、多値属性を更新するときは、追加する 1 つの値だけでなく、新しいリスト全体を指定する必要があります。

たとえば、上記のコマンドでは、パーミッションのリストには、読み取り、書き込み、および削除が含まれます。**permission-mod** コマンドでリストを更新する場合は、すべての値を追加する必要があります。すべての値を追加しないと、追加されていない値は削除されます。

例 1: - **ipa permission-mod** コマンドは、以前に追加した権限をすべて更新します。

```
$ ipa permission-mod --right=read --right=write --right=delete ...
```

または

```
$ ipa permission-mod --right={read,write,delete} ...
```

例 2 - **ipa permission-mod** コマンドは、コマンドに含まれないため、**--right=delete** 引数を削除します。

```
$ ipa permission-mod --right=read --right=write ...
```

または

```
$ ipa permission-mod --right={read,write} ...
```

3.9. IDM ユーティリティーで特殊文字を使用する方法

特殊文字を含むコマンドライン引数を **ipa** コマンドに渡す場合は、この文字をバックスラッシュ (\) でエスケープします。たとえば、一般的な特殊文字には、山かっこ (< および >)、アンパサンド (&)、アスタリスク (*)、またはバーティカルバー (|) があります。

たとえば、アスタリスク (*) をエスケープするには、次のコマンドを実行します。

```
$ ipa certprofile-show certificate_profile --out=exported\*profile.cfg
```

シェルが特殊文字を正しく解析できないため、エスケープしていない特殊文字をコマンドに含めると、予想通りに機能しなくなります。

第4章 コマンドラインから IDENTITY MANAGEMENT エントリーの検索

次のセクションでは、オブジェクトの検索または表示に役立つ IPA コマンドの使用方法を説明します。

4.1. IDM エントリーのリスト表示の概要

ipa *-find コマンドを使用すると、特定のタイプの IdM エントリーを検索できます。

すべての **find** コマンドを表示するには、次の ipa help コマンドを使用します。

```
$ ipa help commands | grep find
```

特定のユーザーが IdM データベースに含まれているかどうかの確認が必要になる場合があります。次のコマンドを使用すると、ユーザーをリスト表示できます。

```
$ ipa user-find
```

指定の属性にキーワードが含まれるユーザーグループのリストを表示するには、次のコマンドを実行します。

```
$ ipa group-find keyword
```

たとえば、**ipa group-find admin** コマンドは、名前または説明に文字列 **admin** が含まれるグループのリストを表示します。

```
----------------
3 groups matched
----------------
  Group name: admins
  Description: Account administrators group
  GID: 427200002

  Group name: editors
  Description: Limited admins who can edit other users
  GID: 427200002

  Group name: trust admins
  Description: Trusts administrators group
```

ユーザーグループの検索の際には、特定のユーザーを含むグループに検索結果を絞り込むことも可能です。

```
$ ipa group-find --user=user_name
```

また、特定のユーザーを含まないグループを検索するには、次のコマンドを実行します。

```
$ ipa group-find --no-user=user_name
```

4.2. 特定のエントリーの詳細の表示

ipa *-show コマンドを使用して、特定の IdM エントリーの詳細を表示します。

手順

- ホスト **server.example.com** に関する詳細を表示します。

```
$ ipa host-show server.example.com

Host name: server.example.com
Principal name: host/server.example.com@EXAMPLE.COM
...
```

4.3. 検索サイズおよび時間制限の調整

IdM ユーザーのリストを要求するなど、一部のクエリーでは、エントリー数が大量に返される場合があります。この検索操作を調整して、**ipa user-find** などの **ipa *-find** コマンドの実行時や、Web UI で対応するリストを表示する際に、全体的なサーバーのパフォーマンスを向上できます。

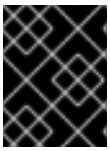
検索サイズ制限

クライアントの CLI または IdM Web UI にアクセスするブラウザーからサーバーに送信されるリクエストで返される最大エントリー数を定義します。
デフォルト - 100 エントリー

検索時間の制限

検索の実行までにサーバーが待機する最大時間 (秒) を定義します。検索がこの制限に到達したら、サーバーは検索を停止し、停止するまでの期間に検出されたエントリーを返します。
デフォルト - 2 秒

この値が **-1** に設定されていると、IdM は、検索時に制限を適用しません。



重要

検索のサイズや時間制限を高く設定しすぎると、サーバーのパフォーマンスに影響を及ぼすことがあります。

4.3.1. コマンドラインで検索サイズおよび時間制限の調整

以下の手順では、コマンドラインで検索サイズと時間制限を調整する方法について説明します。

- グローバル
- 特定のエントリーの場合

手順

1. 現在の検索時間およびサイズ制限を CLI で表示するには、**ipa config-show** コマンドを使用します。

```
$ ipa config-show

Search time limit: 2
Search size limit: 100
```

2. すべてのクエリーに対して **グローバルに** 制限を調整するには、**ipa config-mod** コマンドを使用して、**--searchrecordslimit** および **--searchtimelimit** のオプションを追加します。以下に例を示します。

```
$ ipa config-mod --searchrecordslimit=500 --searchtimelimit=5
```

3. 特定のクエリーに対してのみ **一時的に** 制限を調整するには、コマンドに **--sizelimit** または **--timelimit** オプションを追加してください。以下に例を示します。

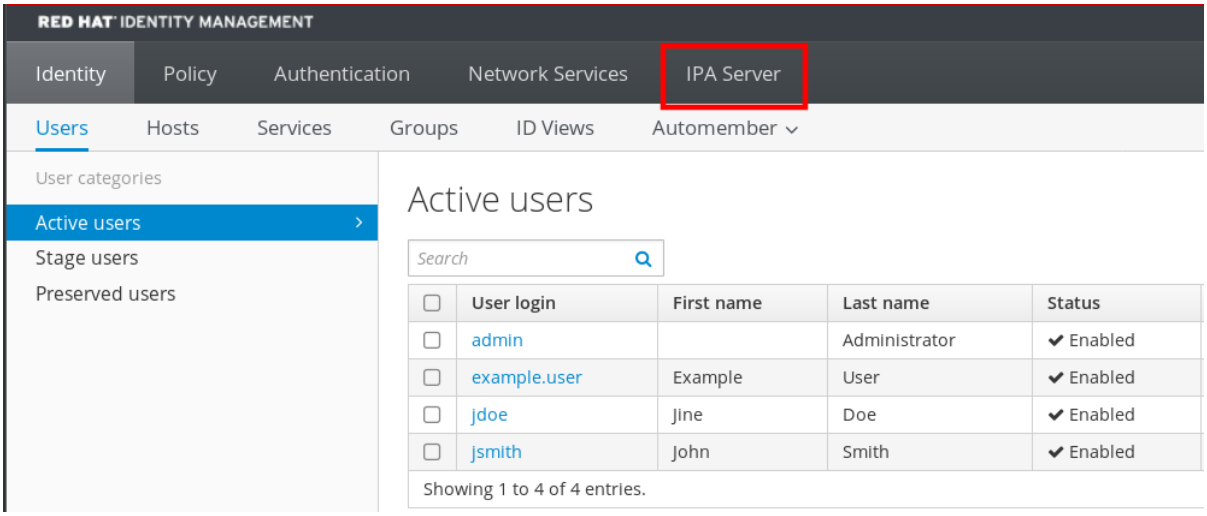
```
$ ipa user-find --sizelimit=200 --timelimit=120
```

4.3.2. Web UI で検索サイズおよび時間制限の調整

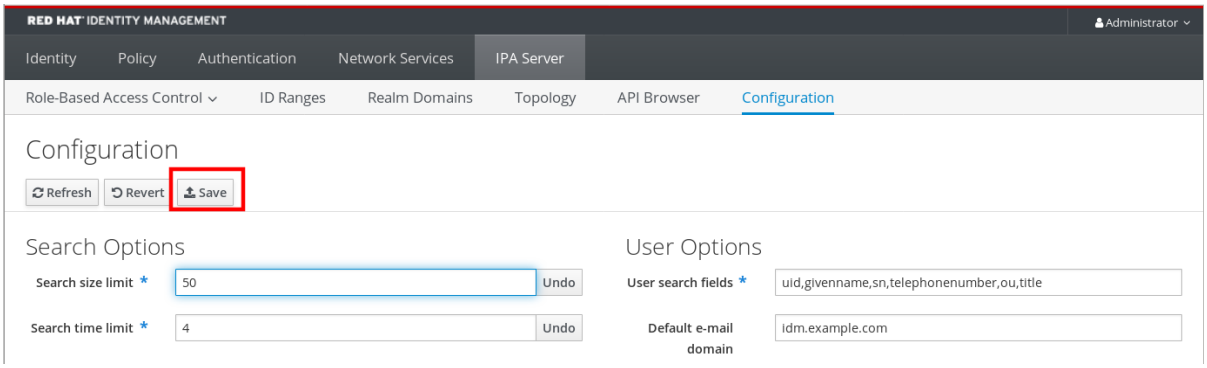
以下の手順では、IdM Web UI でグローバル検索のサイズと時間制限を調整する方法について説明します。

手順

1. IdM Web UI にログインします。
2. **IPA Server** をクリックします。



3. **IPA Server** タブで、**Configuration** をクリックします。
4. **Search Options** エリアに必要な値を設定します。
デフォルト値は以下の通りです。
   - 検索サイズの制限 - 100 エントリー
   - 検索時間の制限 - 2 秒
5. ページ上部にある **Save** をクリックします。


RED HAT IDENTITY MANAGEMENT
Administrator
Identity
Policy
Authentication
Network Services
IPA Server
Role-Based Access Control
ID Ranges
Realm Domains
Topology
API Browser
Configuration
Configuration
Refresh
Revert
Save
Search Options
Search size limit *
50
Undo
Search time limit *
4
Undo
User Options
User search fields *
uid,givenname,sn,telephonenumber,ou,title
Default e-mail domain
idm.example.com

第5章 WEB ブラウザーで IDM WEB UI へのアクセス

IdM (Identity Management) Web UI は、IdM 管理用の Web アプリケーションであり、IdM コマンドラインインターフェイス (CLI) のグラフィカルな代替手段です。

5.1. IDM WEB UI とは

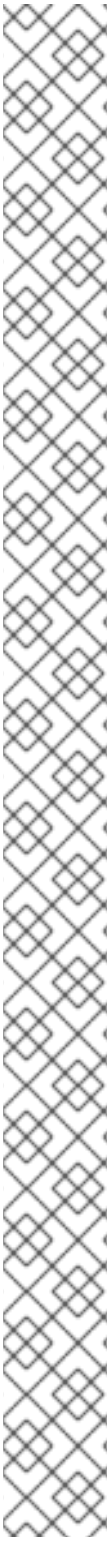
IdM (Identity Management) Web UI は、IdM 管理用の Web アプリケーションです。IdM Web UI には、以下のユーザーとしてアクセスできます。

- **IdM ユーザー** - IdM サーバーでユーザーに付与されている権限に応じて制限されている一連の操作。基本的に、アクティブな IdM ユーザーは IdM サーバーにログインして自身のアカウントを設定できます。その他のユーザーまたは IdM サーバーの設定は変更できません。
- **管理者** - IdM サーバーへのフルアクセス権
- **Active Directory ユーザー** - ユーザーに付与されている権限に応じた一連の操作Active Directory ユーザーが Identity Management の管理可能に詳細は IdM を管理する AD ユーザーの有効化 を参照してください。

5.2. WEB UI へのアクセスに対応している WEB ブラウザー

Identity Management (IdM) は、Web UI への接続に、以下のブラウザーをサポートします。

- Mozilla Firefox 38 以降
- Google Chrome 46 以降



注記

ブラウザーで TLS v1.3 を使用しようとすると、スマートカードで IdM Web UI にアクセスできなくなる場合があります。

```
[ssl:error] [pid 125757:tid 140436077168384] [client 999.999.999.999:99999] AH:
verify client post handshake
[ssl:error] [pid 125757:tid 140436077168384] [client 999.999.999.999:99999]
AH10158: cannot perform post-handshake authentication
[ssl:error] [pid 125757:tid 140436077168384] SSL Library Error: error:14268117:SSL
routines:SSL_verify_client_post_handshake:extension not received
```

これは、最新バージョンのブラウザーで TLS Post-Handshake Authentication (PHA) がデフォルトで有効になっていないか、PHA をサポートしていないためです。スマートカード認証で IdM Web UI にアクセスする場合など、Web サイトの一部にのみ TLS クライアント証明書を必要とする場合は、PHA が必要です。

Mozilla Firefox 68 以降でこの問題を解決するには、TLS PHA を有効にします。

1. アドレスバーに **about:config** と入力して、Mozilla Firefox 設定メニューにアクセスします。
2. 検索バーに **security.tls.enable_post_handshake_auth** と入力します。
3. トグルボタンをクリックして、パラメーターを true に設定します。

現在 PHA をサポートしていない Chrome でこの問題を解決するには、TLS v1.3 を無効にします。

1. **/etc/httpd/conf.d/ssl.conf** 設定ファイルを開きます。
2. **-TLSv1.3** を **SSLProtocol** オプションに追加します。

```
SSLProtocol all -TLSv1 -TLSv1.1 -TLSv1.3
```

3. **httpd** サービスを再起動します。

```
service httpd restart
```

IdM は **ssl.conf** ファイルを管理するため、パッケージの更新時にそのコンテンツが上書きされる可能性があることに注意してください。IdM パッケージの更新後に、カスタム設定を確認します。

5.3. WEB UI へのアクセス

次の手順では、パスワードを使用して、IdM (Identity Management) Web UI に最初にログインする方法を説明します。

最初のログイン後に、IdM サーバーを認証するように設定できます。

- Kerberos チケット
  詳細は、Kerberos authentication in Identity Management を参照してください。
- スマートカード
  詳細は、スマートカード認証用の IdM サーバーの設定 を参照してください。

- ワンタイムパスワード (OTP) - パスワードと組み合わせることができ、Kerberos 認証に利用されます。
  詳細は、One time password (OTP) authentication in Identity Management を参照してください。

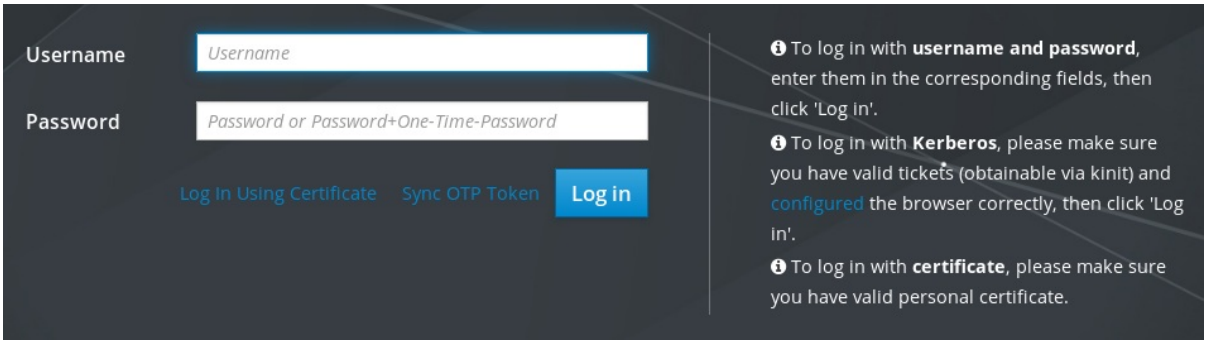
手順

1. ブラウザーのアドレスバーに、IdM サーバーの URL を入力します。名前は次の例のようになります。

   ```
   https://server.example.com
   ```

   server.example.com を、IdM サーバーの DNS 名に変更するだけです。

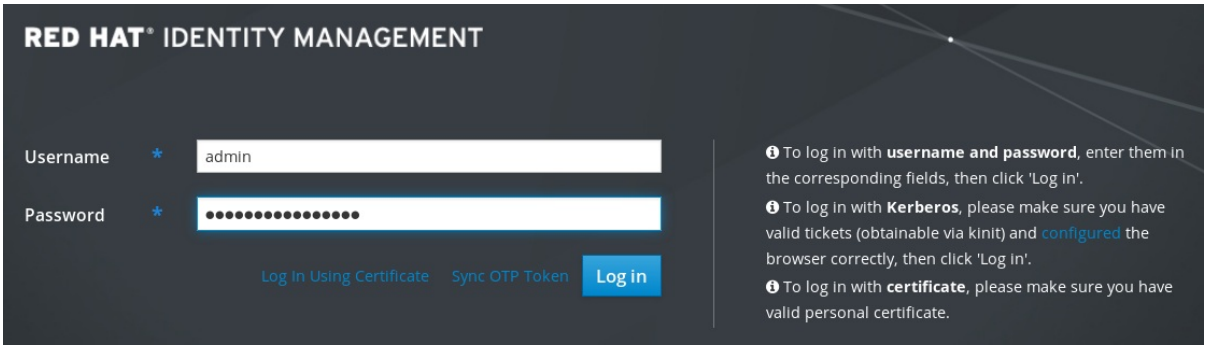
   これで、ブラウザーに IdM Web UI ログイン画面が開きます。



   - サーバーが応答しない、またはログイン画面が開かない場合は、接続している IdM サーバーの DNS 設定を確認してください。
   - 自己署名証明書を使用する場合は、ブラウザーに警告が表示されます。証明書を確認し、セキュリティー例外を許可して、ログインを続行します。
     セキュリティーの例外を回避するために、認証局が署名した証明書をインストールします。

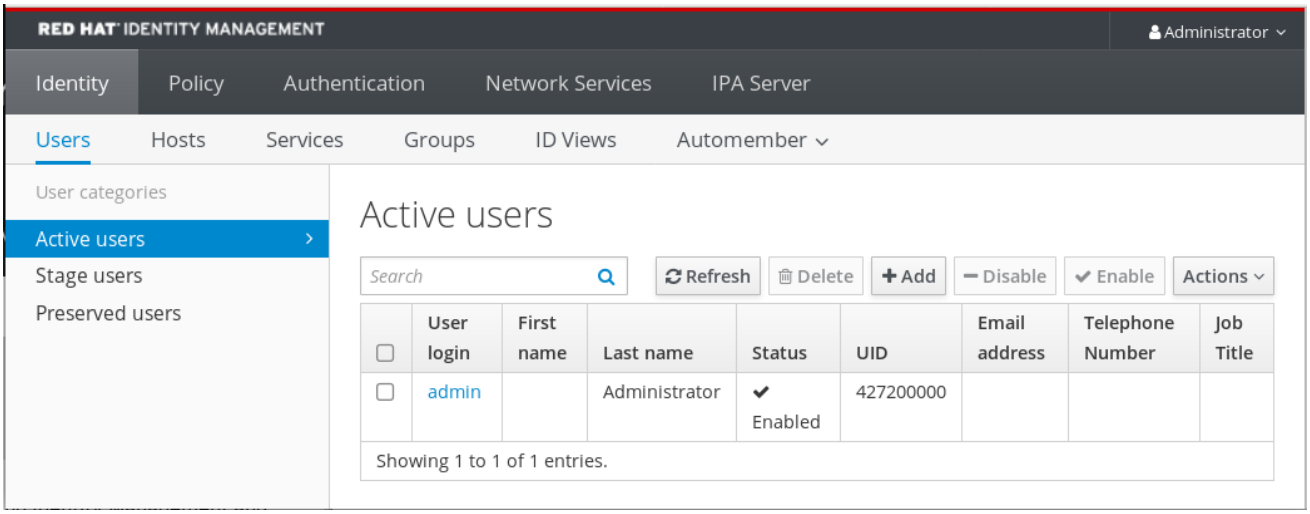
2. Web UI ログイン画面で、IdM サーバーのインストール時に追加した管理者アカウントの認証情報を入力します。
   詳細は Identity Management サーバーのインストール: 統合 DNS と統合 CA の場合 を参照してください。

   IdM サーバーに、個人アカウントの認証情報がすでに入力されている場合は、それを入力できます。



3. **Log in** をクリックします。

ログインに成功したら、IdM サーバーの設定を開始できます。


RED HAT IDENTITY MANAGEMENT
Administrator
Identity
Policy
Authentication
Network Services
IPA Server
Users
Hosts
Services
Groups
ID Views
Automember
User categories
Active users
Stage users
Preserved users
Active users
Search
Refresh
Delete
Add
Disable
Enable
Actions
User login
First name
Last name
Status
UID
Email address
Telephone Number
Job Title
admin
Administrator
Enabled
427200000
Showing 1 to 1 of 1 entries.

第6章 WEB UI で IDM にログイン: KERBEROS チケットの使用

IdM Web UI への Kerberos ログインと、Kerberos 認証を使用した IdM へのアクセスを有効にするように環境を設定する方法について詳しく説明します。

前提条件

- ネットワーク環境にインストールしている IdM サーバー
  詳細は Red Hat Enterprise Linux 8 の Identity Management のインストール を参照してください。

6.1. IDENTITY MANAGEMENT における KERBEROS 認証

Identity Management (IdM) は、シングルサインオンに対応する Kerberos プロトコルを使用します。シングルサインオン認証により、ユーザーが正しいユーザー名およびパスワードを一度入力すれば、再度システムに認証情報を求められることなく、Identity Management サービスにアクセスできます。

DNS および証明書の設定が適切に設定されている場合は、インストール直後に、IdM サーバーが Kerberos 認証を提供します。詳細は、Identity Management のインストール を参照してください。

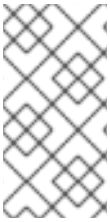
ホストで Kerberos 認証を使用するには、以下をインストールします。

- IdM クライアント
  詳細は Identity Management クライアントをインストールするためのシステムの準備 を参照してください。

- krb5conf パッケージ

6.2. KINIT による IDM への手動ログイン

kinit ユーティリティーを使用して Identity Management (IdM) 環境に対して手動で認証するには、次の手順に従います。**kinit** ユーティリティーは、IdM ユーザーの代わりに Kerberos の TGT (Ticket-Granting Ticket) を取得して、キャッシュに格納します。



注記

この手順は、最初の Kerberos TGT を破棄したか、有効期限が切れている場合にのみ使用します。ローカルマシンに、IdM ユーザーとしてログインすると、IdM に自動的にログインします。これは、ログイン後に IdM リソースにアクセスするのに **kinit** ユーティリティーを使用する必要がないことを示しています。

手順

1. IdM にログインします。

   - ローカルシステムに現在ログインしているユーザーのユーザー名で、(ユーザー名を指定せずに) **kinit** を使用します。たとえば、ローカルシステムにログインしているユーザーが **example_user** の場合は、次のコマンドを実行します。

     ```
     [example_user@server ~]$ kinit
     Password for example_user@EXAMPLE.COM:
     [example_user@server ~]$
     ```

ローカルユーザーのユーザー名と、IdM のユーザーエントリーが一致しないと、認証に失敗します。

```
[example_user@server ~]$ kinit
kinit: Client 'example_user@EXAMPLE.COM' not found in Kerberos database while
getting initial credentials
```

- ローカルユーザー名に対応しない Kerberos プリンシパルを使用して、**kinit** ユーティリティーに必要なユーザー名を渡します。たとえば、**admin** ユーザーとしてログインするには、次のコマンドを実行します。

```
[example_user@server ~]$ kinit admin
Password for admin@EXAMPLE.COM:
[example_user@server ~]$
```

2. 必要に応じて、ログインが成功したことを確認するには、**klist** ユーティリティーを使用して、キャッシュした TGT を表示します。以下の例では、キャッシュに **example_user** プリンシパルのチケットが含まれています。これは、このホストでは IdM サービスにアクセスするのは、**example_user** にのみ許可されていることを示しています。

```
$ klist
Ticket cache: KEYRING:persistent:0:0
Default principal: example_user@EXAMPLE.COM

Valid starting       Expires              Service principal
11/10/2019 08:35:45  11/10/2019 18:35:45  krbtgt/EXAMPLE.COM@EXAMPLE.COM
```

6.3. KERBEROS 認証用のブラウザーの設定

Kerberos チケットによる認証を有効にするには、ブラウザーの設定が必要になることもあります。

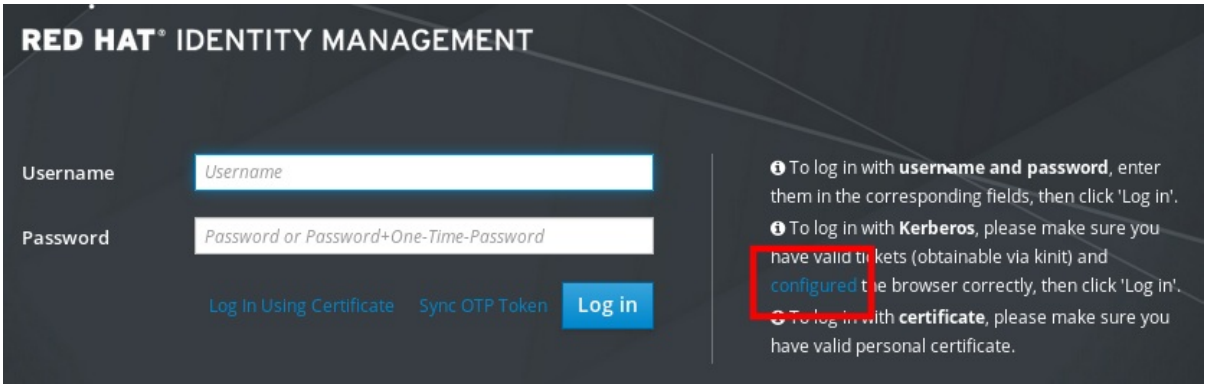
以下の手順は、IdM ドメインにアクセスする Kerberos ネゴシエーションに対応するのに役に立ちます。

Kerberos に対応する方法はブラウザーによって異なるため、異なる設定が必要です。IdM Web UI には、次のブラウザーに関するガイドラインが含まれています。

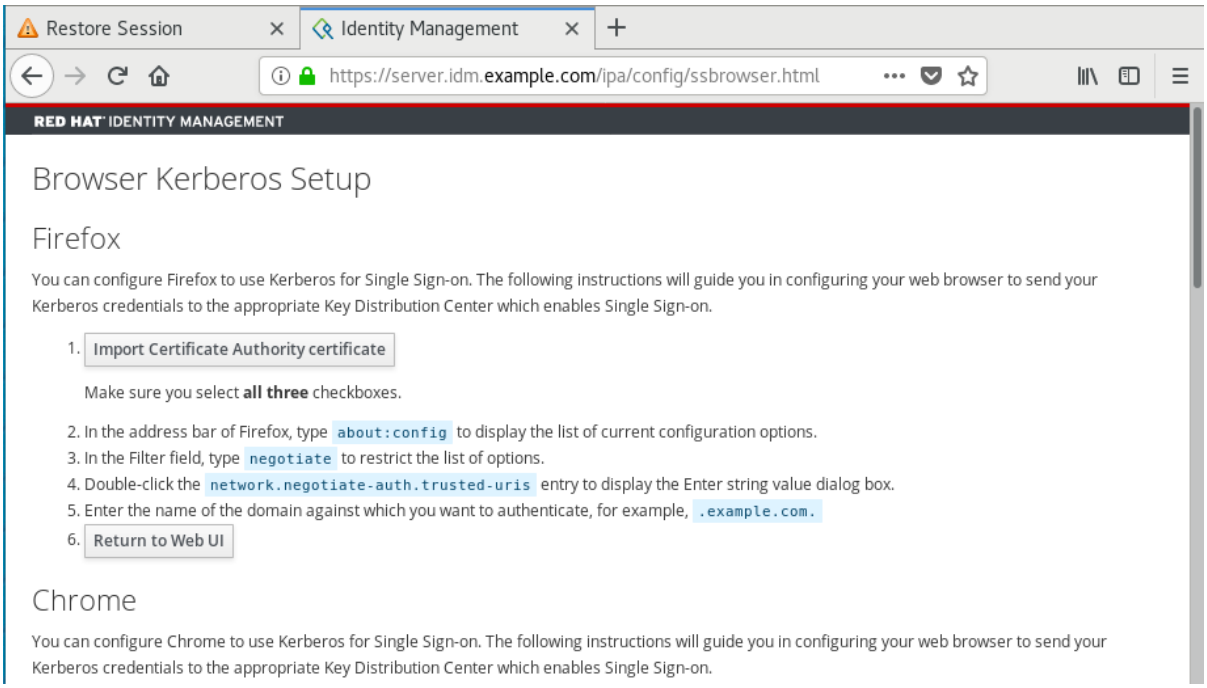
- Firefox
- Chrome

手順

1. Web ブラウザーで、WebUI ログインダイアログを開きます。
2. Web UI のログイン画面で、ブラウザー設定のリンクをクリックします。



3. 設定ページの手順に従います。



設定が完了したら、IdM Web UI に戻り、**ログイン** をクリックします。

6.4. KERBEROS チケットで WEB UI へのログイン

Kerberos Ticket-Granting Ticket (TGT) を使用して IdM Web UI にログインするには、次の手順に従います。

TGT は、事前定義された時間で有効期限が切れます。デフォルトの時間間隔は 24 時間で、IdM Web UI でそれを変更できます。

期限が切れたら、チケットを更新する必要があります。

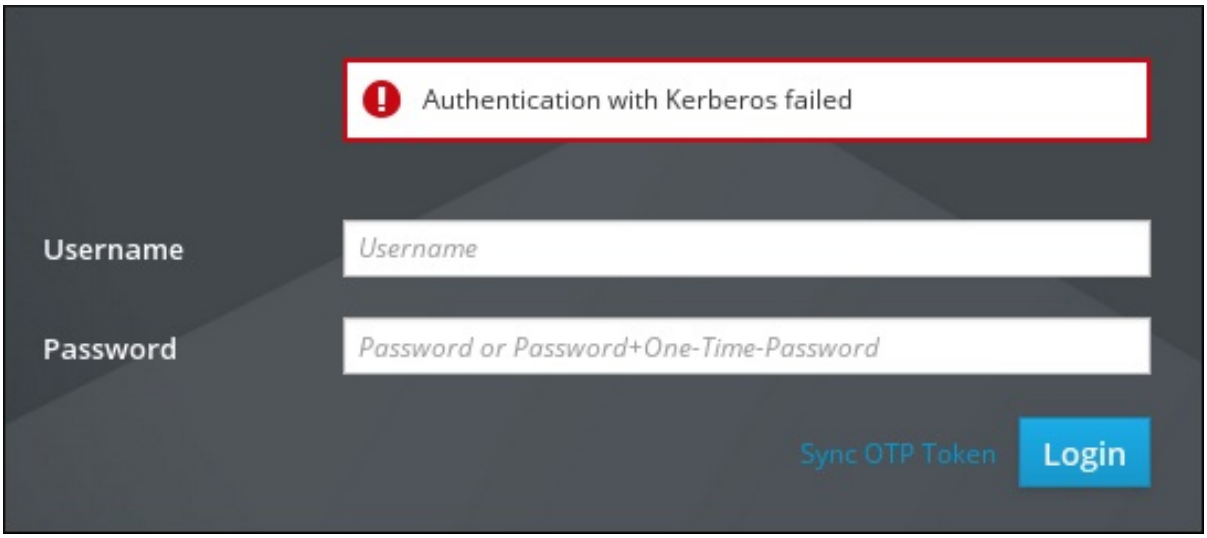
- kinit コマンドの使用
- Web UI ログインダイアログで、IdM ログイン認証情報を使用

手順

- IdM Web UI を開きます。
  Kerberos 認証が正しく機能し、有効なチケットがある場合は、自動的に認証されて Web UI が開きます。

チケットの有効期限が切れている場合は、最初に認証情報を使用して認証する必要があります。ただし、次からはログインダイアログを開かずに IdM Web UI が自動的に開きます。

エラーメッセージ **Authentication with Kerberos failed** が表示された場合は、ブラウザーが Kerberos 認証用に設定されていることを確認してください。Kerberos 認証用のブラウザーの設定 を参照してください。



6.5. KERBEROS 認証用の外部システムの設定

Identity Management (IdM) ユーザーが Kerberos 認証情報を使用して外部システムから IdM にログインできるように外部システムを設定するには、この手順に従います。

外部システムの Kerberos 認証を有効にすることは、インフラストラクチャーに、複数のレルムまたは重複ドメインが含まれている場合に特に便利です。また、**ipa-client-install** を実行してシステムを IdM ドメインに登録していない場合にも便利です。

IdM ドメインのメンバーではないシステムから IdM への Kerberos 認証を有効にするには、IdM 固有の Kerberos 設定ファイルを外部システムに定義します。

前提条件

- 外部システムに **krb5-workstation** パッケージがインストールされている。
  パッケージがインストールされているかどうかを確認するには、次の CLI コマンドを使用します。

```
# yum list installed krb5-workstation
Installed Packages
krb5-workstation.x86_64    1.16.1-19.el8    @BaseOS
```

手順

1. IdM サーバーから外部システムに **/etc/krb5.conf** ファイルをコピーします。以下に例を示します。

```
# scp /etc/krb5.conf root@externalsystem.example.com:/etc/krb5_ipa.conf
```

警告

外部マシンにある既存の **krb5.conf** ファイルは上書きしないでください。

2. 外部システムで、コピーした IdM の Kerberos 設定ファイルを使用するように、端末セッションを設定します。

```
$ export KRB5_CONFIG=/etc/krb5_ipa.conf
```

KRB5_CONFIG 変数は、ログアウトまで一時的に存在します。ログアウト時に削除されないように、この変数のファイル名を変えてエクスポートします。

3. **/etc/krb5.conf.d/** ディレクトリーの Kerberos 設定部分を、外部システムにコピーします。

4. Kerberos 認証用のブラウザーの設定 の説明に従って、外部システムでブラウザーを設定します。

外部システムのユーザーが、**kinit** ユーティリティーを使用して IdM サーバーで認証できるようになりました。

6.6. ACTIVE DIRECTORY ユーザーの WEB UI ログイン

Active Directory ユーザーに対して Web UI ログインを有効にするには、**デフォルトの信頼ビュー** で、Active Directory の各ユーザーに対して ID のオーバーライドを定義します。以下に例を示します。

```
[admin@server ~]$ ipa idoverrideuser-add 'Default Trust View' ad_user@ad.example.com
```

関連情報

- Active Directory ユーザーの ID ビューの使用

第7章 ワンタイムパスワードを使用した IDENTITY MANAGEMENT WEB UI へのログイン

IdM Web UI へのアクセスは、いくつかの方法を使用して保護できます。基本的なものはパスワード認証です。

パスワード認証のセキュリティーを向上させるために、2 つ目の手順を追加して、自動生成ワンタイムパスワード (OTP) を要求できます。最も一般的な使用方法は、ユーザーアカウントに関連付けられたパスワードと、ハードウェアまたはソフトウェアのトークンにより生成された期限付きワンタイムパスワードを組み合わせることです。

以下のセクションでは、次のことができます。

- IdM で OTP 認証がどう機能するかを理解する。
- IdM サーバーで OTP 認証を設定する。
- IdM で OTP バリデーション用に RADIUS サーバーを設定する。
- OTP トークンを作成し、そのトークンを、電話の FreeOTP アプリと同期する。
- ユーザーパスワードとワンタイムパスワードの組み合わせて、IdM Web UI に対して認証する。
- Web UI でトークンを再同期する。

7.1. 前提条件

- Web ブラウザーで IdM Web UI へのアクセス

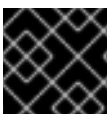
7.2. IDENTITY MANAGEMENT におけるワンタイムパスワード (OTP) 認証

ワンタイムパスワードにより、認証セキュリティーに関する手順が追加されます。認証では、自身のパスワードと、自動生成されたワンタイムパスワードが使用されます。

ワンタイムパスワードを生成するには、ハードウェアまたはソフトウェアのトークンを使用できます。IdM は、ソフトウェアトークンとハードウェアトークンの両方をサポートします。

Identity Management は、以下にある、2 つの標準 OTP メカニズムに対応しています。

- HMAC ベースのワンタイムパスワード (HOTP) アルゴリズムは、カウンターに基づいています。HMAC は、Hashed Message Authentication Code (ハッシュメッセージ認証コード) を表しています。
- 時間ベースのワンタイムパスワード (TOTP) アルゴリズムは、時間ベースの移動要素に対応する HOTP の拡張機能です。



重要

IdM は、Active Directory 信頼ユーザーの OTP ログインに対応していません。

7.3. WEB UI でのワンタイムパスワードの有効化

Identity Management (IdM) 管理者は、IdM ユーザーに対して 2 要素認証 (2FA) をシステム全体でまたは個別に有効にすることができます。ユーザーは、コマンドラインまたは Web UI ログインダイアログ

の専用フィールドで、通常のパスワードの後にワンタイムパスワード (OTP) を入力します。これらのパスワードの間にはスペースを入れません。

2FA を有効にしても、2FA が強制されるわけではありません。LDAP バインドに基づくログインを使用する場合、IdM ユーザーはパスワードを入力するだけで認証できます。ただし、**krb5** ベースのログインを使用する場合は、2FA が強制されます。Red Hat は今後のリリースで設定オプションを提供して、管理者が次のいずれかを選択できるようにする予定です。

- ユーザーが独自のトークンを設定できるようにします。この場合、**krb5** ベースのログインでは 2FA が強制されますが、LDAP バインドでは依然として強制されません。
- ユーザーが独自のトークンを設定できないようにします。この場合、2FA は LDAP バインドと **krb5** ベースのログインの両方で強制されます。

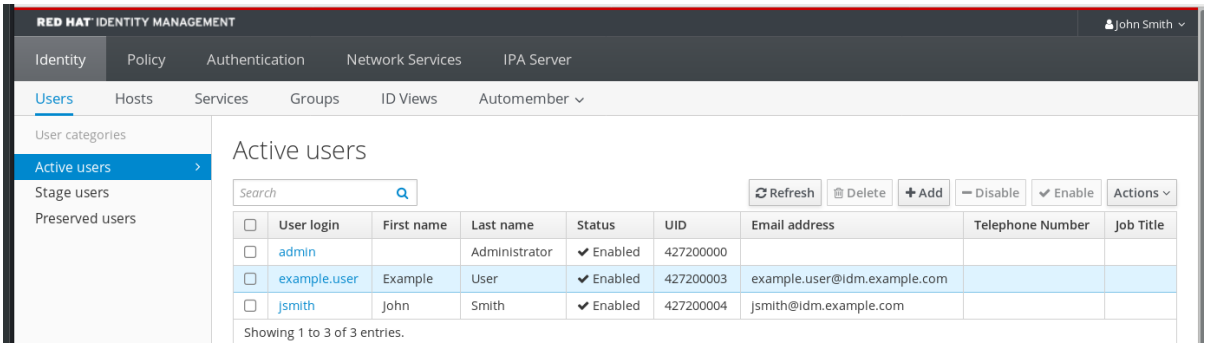
IdM Web UI を使用して、個々の **example.user** IdM ユーザーに対して 2FA を有効にするには、以下の手順を完了します。

前提条件

- 管理者権限

手順

1. IdM **admin** 権限を使用して IdM Web UI にログインします。
2. **Identity → Users → Active users**タブを開きます。



3. **example.user** を選択してユーザー設定を開きます。
4. **User authentication types** で、**Two factor authentication (password + OTP)** を選択します。
5. **Save** をクリックします。

この時点で、IdM ユーザーに対して OTP 認証が有効になります。

次に、管理者権限を持つユーザーまたは **example.user** が、**example.user** アカウントに新しいトークン ID を割り当てる必要があります。

7.4. IDM での OTP バリデーション用の RADIUS サーバー設定

プロプライエタリーのワンタイムパスワード (OTP) ソリューションから Identity Management (IdM) ネイティブの OTP ソリューションへの大規模なデプロイメントの移行を可能にするために、IdM では、ユーザーのサブセットに対して OTP バリデーションをサードパーティーの RADIUS サーバーにオフロードすることができます。管理者は、各プロキシーが単一の RADIUS サーバーのみを参照できる RADIUS プロキシーのセットを作成します。複数のサーバーに対応する必要がある場合は、複数の RADIUS サーバーを参照する仮想 IP ソリューションを作成することが推奨されます。

このようなソリューションは、**keepalived** デーモンなどを使用して、RHEL IdM の外部でビルドする必要があります。次に、管理者はこれらのプロキシーセットのいずれかをユーザーに割り当てます。ユーザーが RADIUS プロキシーが設定されている限り、IdM は他のすべての認証メカニズムをバイパスします。

注記

IdM は、サードパーティーシステムのトークンに対するトークン管理または同期のサポートを提供しません。

OTP バリデーション用に RADIUS サーバーを設定し、プロキシーサーバーにユーザーを追加する手順を実行します。

前提条件

- radius ユーザー認証方法が有効になっている。詳細は、Web UI でのワンタイムパスワードの有効化 を参照してください。

手順

1. RADIUS プロキシーを追加します。

```
$ ipa radiusproxy-add proxy_name --secret secret
```

このコマンドは、必要な情報を挿入するように求められます。

RADIUS プロキシーの設定には、クライアントとサーバーとの間の共通のシークレットを使用して認証情報をラップする必要があります。**--secret** パラメーターにこのシークレットを指定します。

2. 追加したプロキシーにユーザーを割り当てます。

```
ipa user-mod radiususer --radius=proxy_name
```

3. 必要に応じて、RADIUS に送信するユーザー名を設定します。

```
ipa user-mod radiususer --radius-username=radius_user
```

これにより、RADIUS プロキシーサーバーがユーザーの OTP 認証の処理を開始します。

ユーザーが IdM ネイティブ OTP システムに移行する準備ができたら、ユーザーの RADIUS プロキシー割り当てを削除するだけです。

7.4.1. 低速ネットワークで RADIUS サーバーを実行する場合の KDC タイムアウト値の変更

低速ネットワークで RADIUS プロキシーを実行している場合などの特定の状況では、ユーザーがトークンを入力するのを待機している間に接続がタイムアウトして、RADIUS サーバーが応答する前に Identity Management (IdM) Kerberos Distribution Center (KDC) が接続を閉じます。

KDC のタイムアウト設定を変更するには、以下を実行します。

1. **/var/kerberos/krb5kdc/kdc.conf** ファイルの **[otp]** セクションで **timeout** パラメーターの値を変更します。たとえば、タイムアウトを **120** 秒に設定するには、以下のようにします。

```
[otp]
DEFAULT = {
  timeout = 120
  ...
}
```

2. **krb5kdc** サービスを再起動します。

```
# systemctl restart krb5kdc
```

関連情報

- ナレッジベース記事 How to configure FreeRADIUS authentication in FIPS mode

7.5. WEB UI での OTP トークンの追加

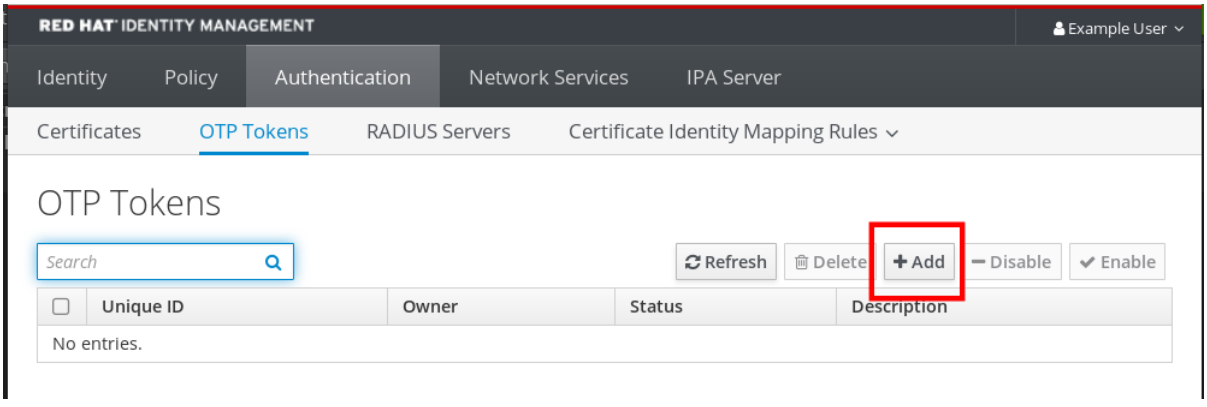
次のセクションは、IdM Web UI およびソフトウェアトークンジェネレーターに、トークンを追加するのに役立ちます。

前提条件

- IdM サーバーでアクティブなユーザーアカウント。
- 管理者が、IdM Web UI の特定のユーザーアカウントに対して OTP を有効にしている。
- FreeOTP などの OTP トークンを生成するソフトウェアデバイス。

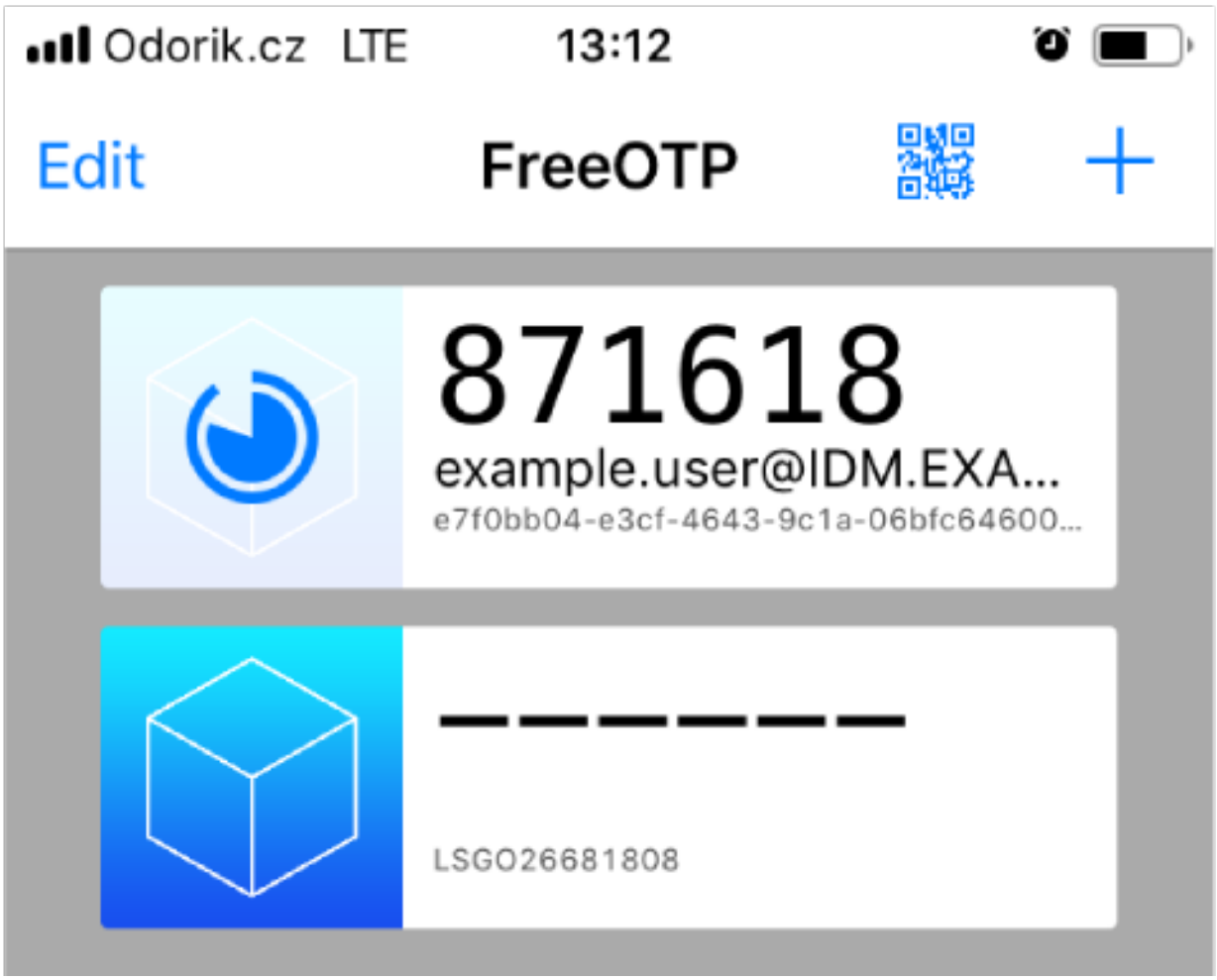
手順

1. ユーザー名とパスワードを使用して IdM Web UI にログインします。
2. **Authentication → OTP Tokens**タブを開いて、携帯電話でトークンを作成します。
3. **Add** をクリックします。



4. **Add OTP token** ダイアログボックスに何も入力せず、**Add** をクリックします。
この段階で、IdM サーバーはデフォルトパラメーターを使用してトークンを作成し、QR コード付きページを開きます。
5. QR コードを携帯電話にコピーします。
6. **OK** をクリックして QR コードを閉じます。

これで、ワンタイムパスワードを生成して、IdM Web UI にログインできるようになりました。



7.6. ワンタイムパスワードで WEB UI にログイン

ワンタイムパスワード (OTP) を使用して IdM Web UI に初めてログインする際には、この手順に従います。

前提条件

- OTP 認証を使用しているユーザーアカウントに対して、Identity Management サーバーで OTP 設定が有効になっている。管理者およびユーザー自身が、OTP を有効にできる。
  OTP 設定を有効にする場合は、Web UI でワンタイムパスワードの有効化 を参照してください。
- 設定された OTP トークンを生成するハードウェアまたはソフトウェアのデバイス

手順

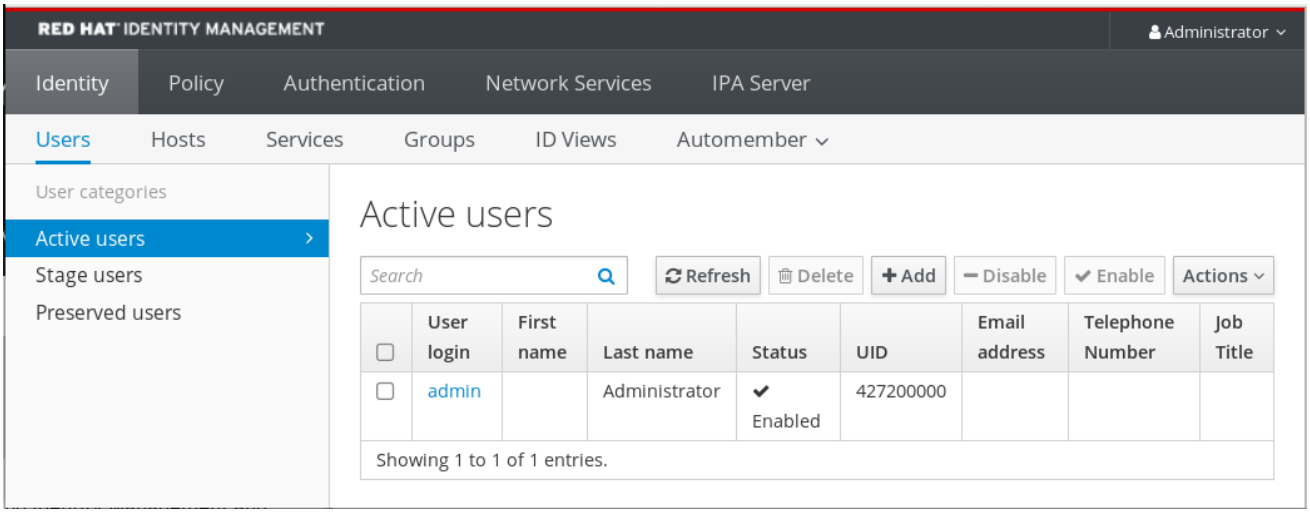
1. Identity Management ログイン画面で、自身のユーザー名、または IdM サーバー管理者アカウントのユーザー名を入力します。
2. 上記で入力したユーザーのパスワードを追加します。
3. デバイスでワンタイムパスワードを生成します。
4. パスワードの直後にワンタイムパスワードを入力します (空白文字は追加しない)。

5. **Log in** をクリックします。
   認証に失敗した場合は、OTP トークンを同期します。

   CA が自己署名証明書を使用する場合は、ブラウザーに警告が表示されます。証明書を確認し、セキュリティー例外を許可して、ログインを続行します。

   IdM Web UI が開かない場合は、Identity Management サーバーの DNS 設定を確認してください。

ログインが成功すると、IdM Web UI が表示されます。



7.7. WEB UI で OTP トークンの同期

OTP (ワンタイムパスワード) でのログインに失敗した場合、OTP トークンは正しく同期しません。

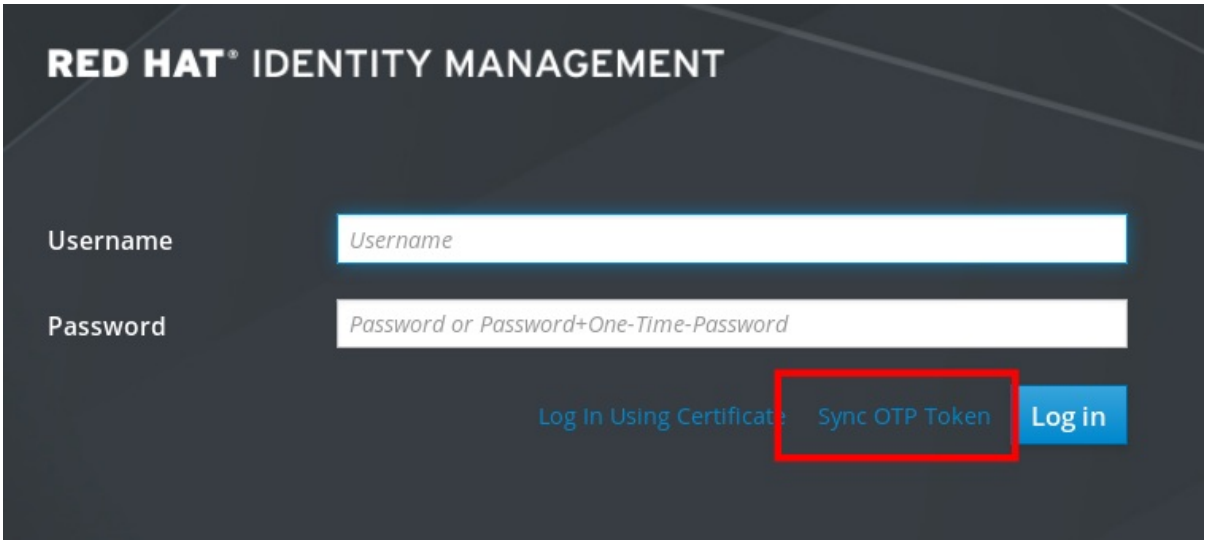
以下のテキストは、トークンの再同期を説明します。

前提条件

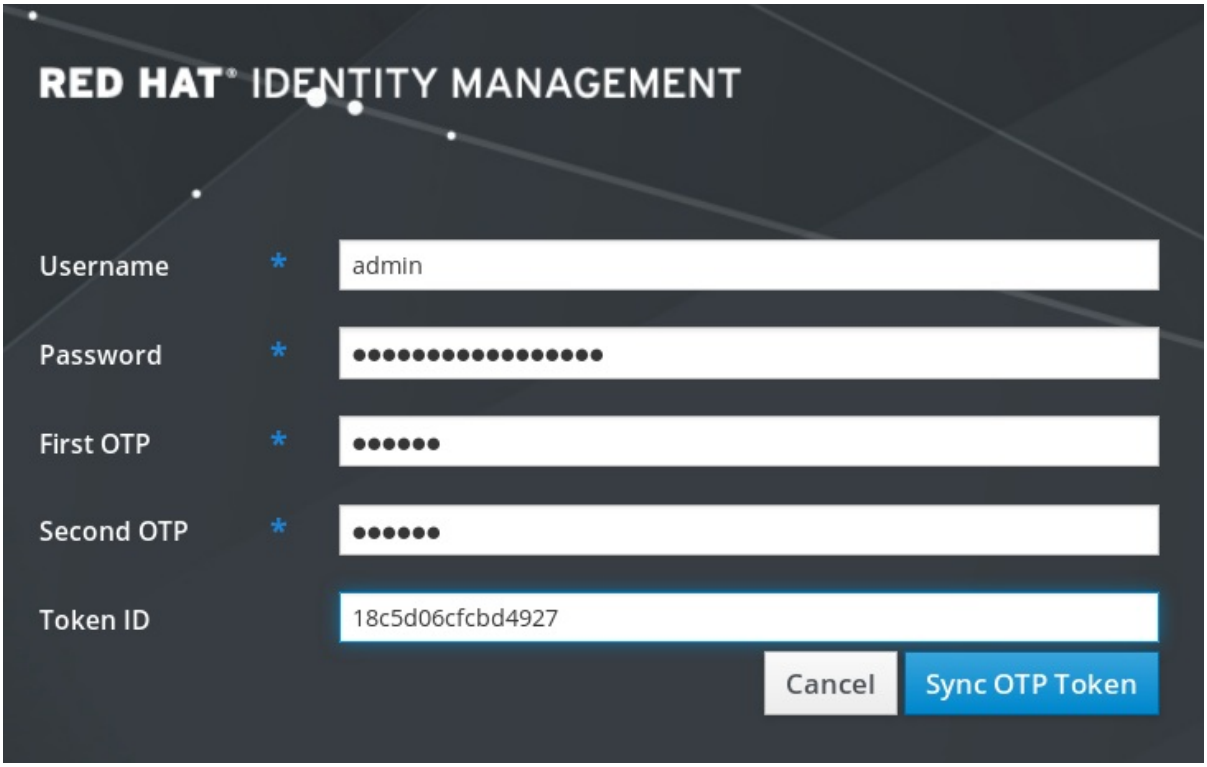
- ログイン画面を開いている。
- 設定した OTP トークンを生成するデバイス。

手順

1. IdM Web UI ログイン画面で、**Sync OTP Token** をクリックします。



2. ログイン画面で、ユーザー名と、Identity Management パスワードを入力します。
3. ワンタイムパスワードを生成し、**First OTP** フィールドに入力します。
4. ワンタイムパスワードをもう一度生成し、**Second OTP** フィールドに入力します。
5. 必要に応じて、トークン ID を入力してします。



6. **Sync OTP Token**をクリックします。

同期に成功したら、IdM サーバーにログインできます。

7.8. 期限切れパスワードの変更

Identity Management の管理者は、ユーザーが次回ログインする時にパスワードを変更するように強制できます。これを設定すると、パスワードを変更しないと IdM Web UI にログインできなくなります。

パスワードの有効期限は、Web UI に初めてログインしたときに発生する可能性があります。

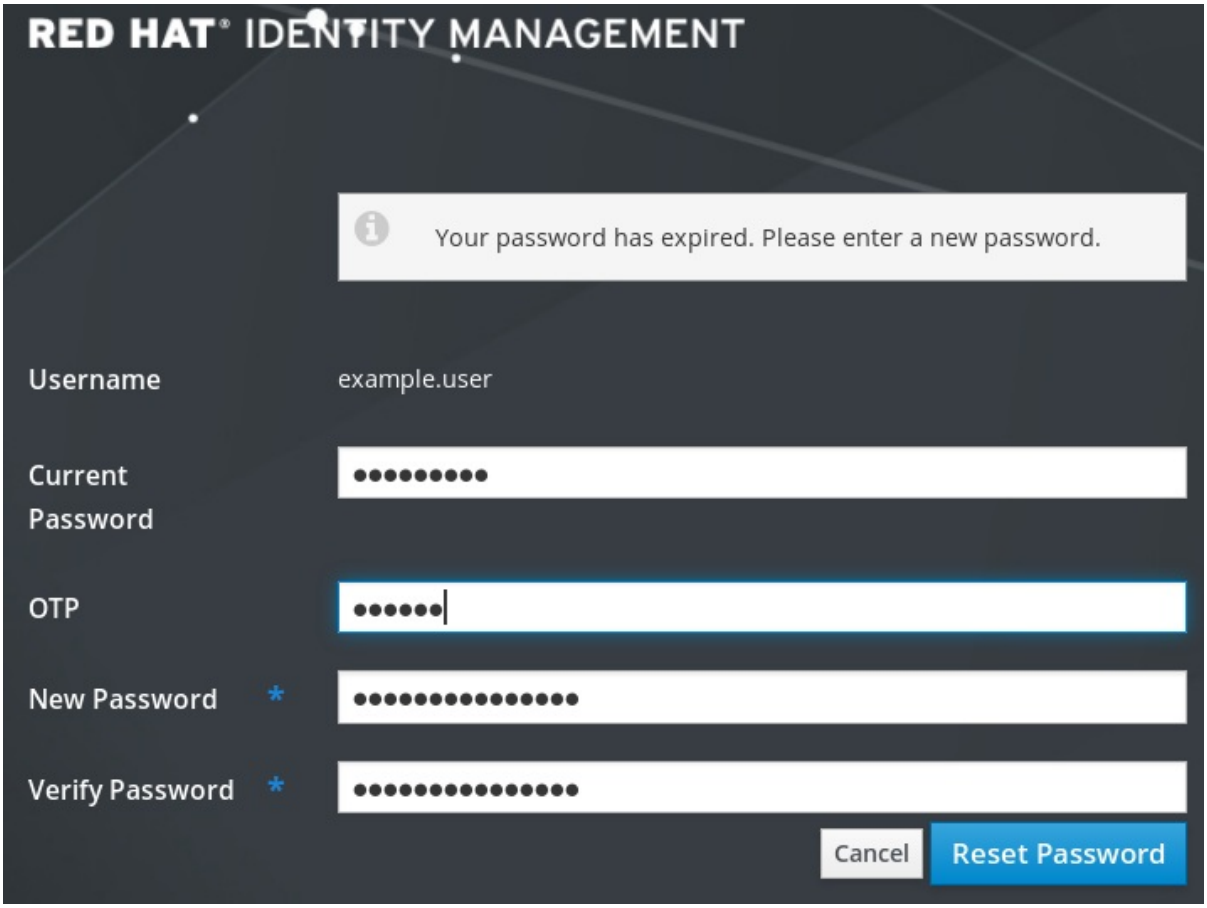
有効期限のパスワードのダイアログが表示されたら、手順の指示に従ってください。

前提条件

- ログイン画面を開いている。
- IdM サーバーへのアクティブなアカウント。

手順

1. パスワード有効期限のログイン画面に、ユーザー名を入力します。
2. 上記で入力したユーザーのパスワードを追加します。
3. ワンタイムパスワード認証を使用する場合は、OTP フィールドにワンタイムパスワードを生成します。
   OTP 認証を有効にしていない場合は、このフィールドを空白のままにします。
4. 確認のために新しいパスワードを 2 回入力します。
5. **Reset Password** をクリックします。



パスワード変更が成功すると、通常のログインダイアログが表示されます。新しいパスワードでログインします。

第8章 IDM で SSSD を使用した認証のトラブルシューティング

Identity Management (IdM) 環境の認証には、さまざまなコンポーネントが含まれます。

IdM クライアントで、以下を行います。

- SSSD サービス
- Name Services Switch (NSS)
- PAM (プラグ可能な認証モジュール)

IdM サーバーで、以下を行います。

- SSSD サービス
- IdM Directory Server。
- IdM Kerberos Key Distribution Center (KDC)

Active Directory (AD) ユーザーとして認証している場合は、以下を行います。

- AD ドメインコントローラー上の Directory Server。
- AD ドメインコントローラー上の Kerberos サーバー

ユーザーを認証するには、SSSD サービスで以下の機能を実行できる必要があります。

- 認証サーバーからユーザー情報を取得します。
- ユーザーに認証情報を求められ、それらの認証情報を認証サーバーに渡し、結果を処理します。

ユーザー情報を保存する SSSD サービスとサーバー間の情報フロー方法を説明し、環境で失敗した認証試行のトラブルシューティングを行うには、次を参照してください。

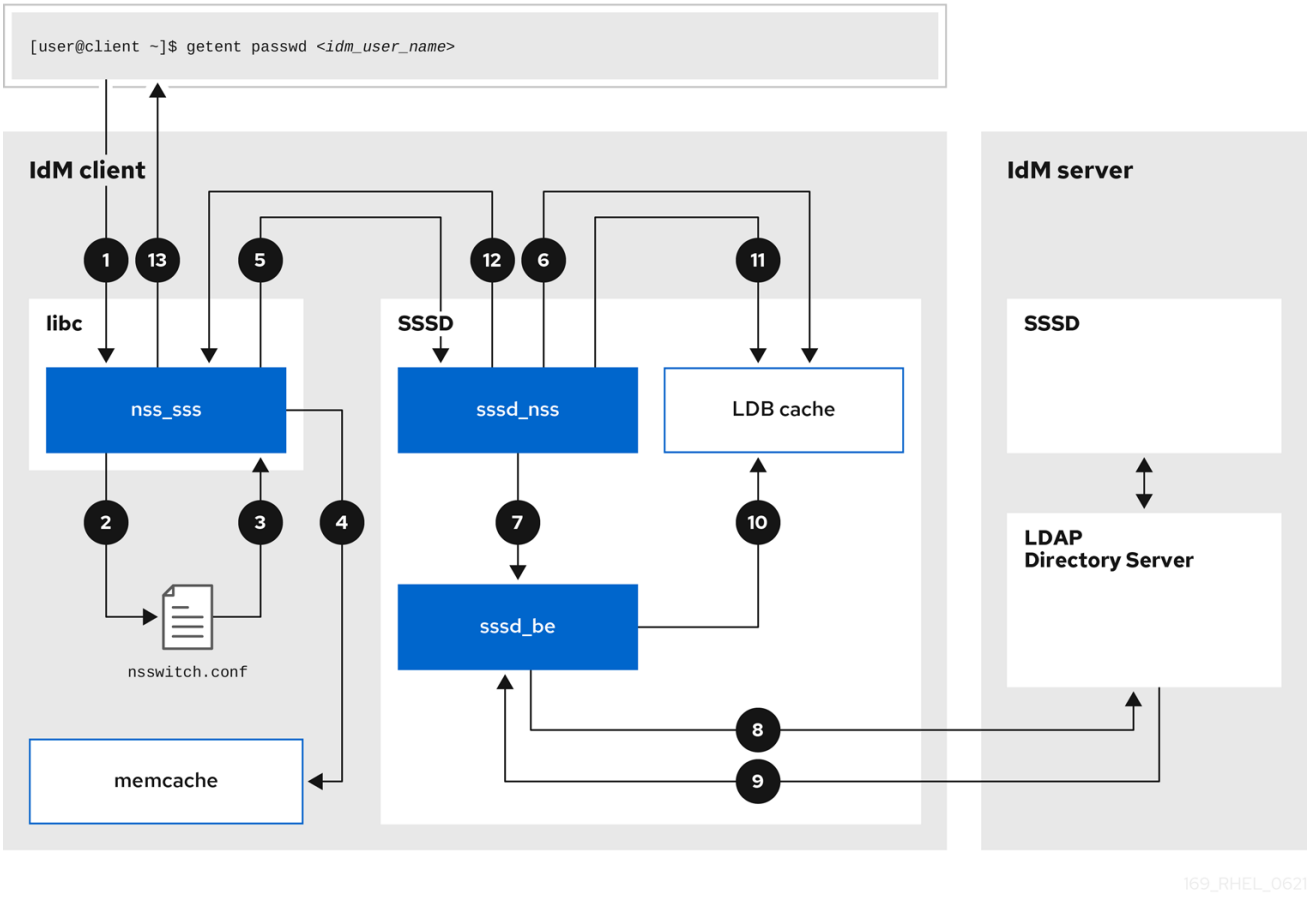
1. SSSD で IdM ユーザー情報を取得するデータフロー
2. SSSD で AD ユーザー情報を取得する際のデータフロー
3. IdM で SSSD を使用してユーザーとして認証する場合にデータフロー
4. 認証問題の範囲の制限
5. SSSD ログファイルおよびログレベル
6. sssd.conf ファイルで SSSD の詳細なロギングの有効化
7. sssctl コマンドを使用した SSSD の詳細なロギングの有効化
8. SSSD サービスからデバッグログを収集し、IdM サーバーによる認証問題のトラブルシューティング
9. SSSD サービスからデバッグログを収集し、IdM クライアントによる認証問題のトラブルシューティング

10. SSSD バックエンドでのクライアント要求の追跡

11. ログアナライザーツールを使用したクライアント要求の追跡

8.1. SSSD で IDM ユーザー情報を取得するデータフロー

以下の図は、**getent passwd <idm_user_name>** コマンドを使用して IdM ユーザー情報の要求時に IdM クライアントと IdM サーバーとの間の情報フローを簡単に説明します。



1. **getent** コマンドは、**libc** ライブラリーから **getpwnam** 呼び出しをトリガーします。

2. **libc** ライブラリーは、**/etc/nsswitch.conf** 設定ファイルを参照して、どのサービスがユーザー情報を提供するかを確認し、SSSD サービスのエントリー **sss** を検出します。

3. **libc** ライブラリーは、**nss_sss** モジュールを開きます。

4. nss_sss モジュールは、ユーザー情報のメモリーマップキャッシュを確認します。データがキャッシュに存在する場合は、**nss_sss** モジュールがそれを返します。

5. ユーザー情報が memory-mapped キャッシュにない場合、リクエストは SSSD **sssd_nss** レスポンダープロセスに渡されます。

6. SSSD サービスはキャッシュをチェックします。データがキャッシュに存在し、有効な場合は、**sssd_nss** レスポンダーがキャッシュからデータを読み取って、アプリケーションに返します。

7. データがキャッシュにない場合や期限切れである場合、**sssd_nss** レスポンダーは適切なバックエンドプロセスに対してクエリーを実行し、応答を待機します。SSSD サービス

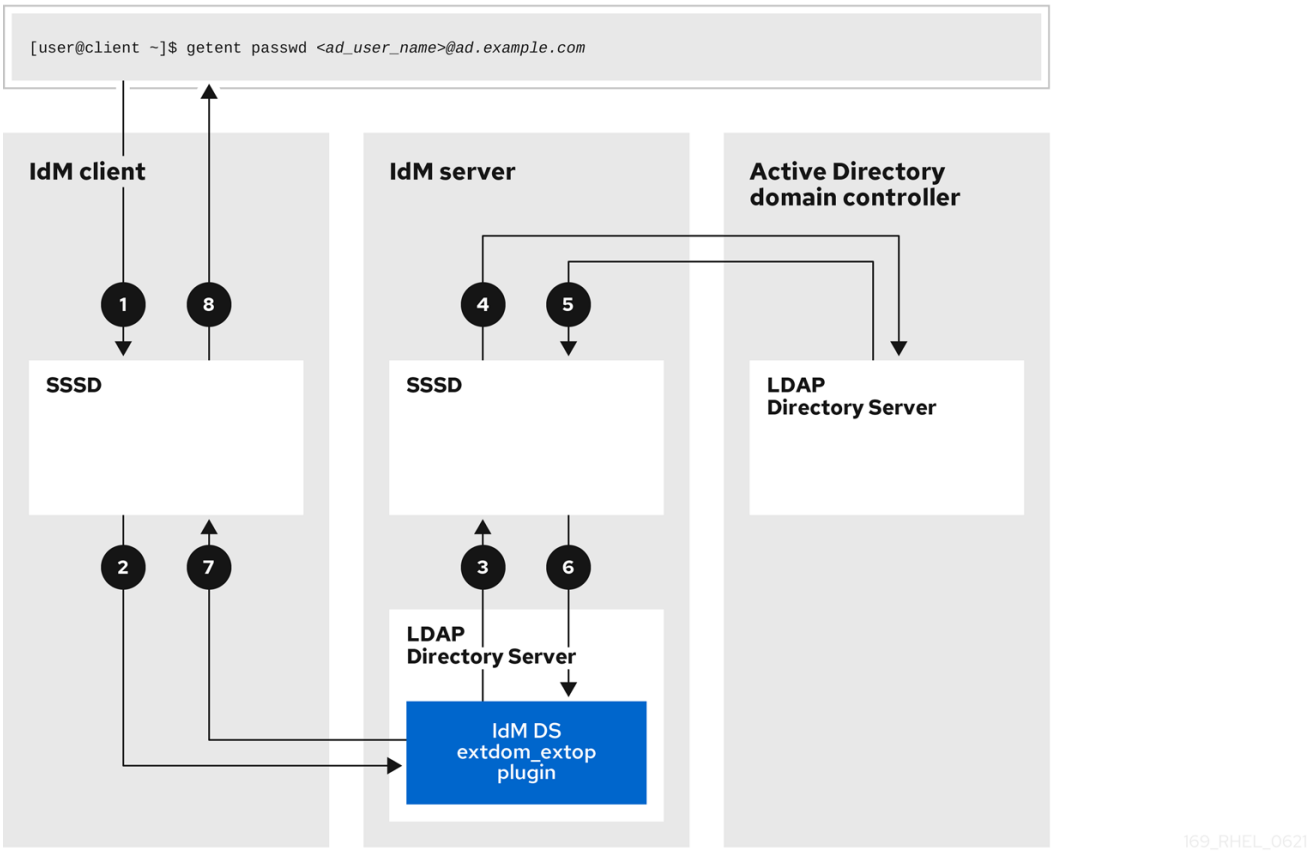
は、**sssd.conf** 設定ファイルで **id_provider=ipa** を設定して有効にした、IdM 環境の IPA バックエンドを使用します。

8. **sssd_be** バックエンドプロセスは IdM サーバーに接続して、IdM LDAP Directory Server から情報を要求します。
9. IdM サーバーの SSSD バックエンドは、IdM クライアントの SSSD バックエンドプロセスに対応します。
10. クライアントの SSSD バックエンドは、生成されるデータを SSSD キャッシュに保存し、キャッシュが更新されたレスポンダープロセスを警告します。
11. **sssd_nss** フロントエンドレスプロセスが SSSD キャッシュから情報を取得します。
12. **sssd_nss** レスポンダーは、**nss_sss** レスポンダーにユーザー情報を送信し、リクエストを完了します。
13. **libc** ライブラリーは、要求したアプリケーションにユーザー情報を返します。

8.2. SSSD で AD ユーザー情報を取得する際のデータフロー

IdM 環境と Active Directory(AD) ドメインとの間でフォレスト間の信頼を確立した場合は、IdM クライアントの AD ユーザー情報を取得する際に情報フローが、AD ユーザーデータベースへの追加の手順とともに、IdM クライアントの AD ユーザー情報の取得時に非常に似ています。

以下の図は、**getent passwd <ad_user_name@ad.example.com>** コマンドを使用してユーザーが AD ユーザーに関する情報を要求する際に、情報フローを簡素化します。この図には、SSSD で IdM ユーザー情報を取得するデータフロー が含まれません。IdM クライアントの SSSD サービス、IdM サーバーの SSSD サービス、および AD ドメインコントローラー上の LDAP データベースとの間の通信にフォーカスします。



1. IdM クライアントは、AD ユーザー情報に関するローカルの SSSD キャッシュを検索します。
2. IdM クライアントにユーザー情報がない場合や、情報が古い場合に、クライアントの SSSD サービスが IdM サーバーの **extdom_extop** プラグインに問い合わせて、LDAP 拡張操作を実行し、情報を要求します。
3. IdM サーバーの SSSD サービスは、ローカルキャッシュで AD ユーザー情報を検索します。
4. IdM サーバーに SSSD キャッシュにユーザー情報がない場合や、その情報が古い場合は、LDAP 検索を実行して、AD ドメインコントローラーからユーザー情報を要求します。
5. IdM サーバーの SSSD サービスは、AD ドメインコントローラーから AD ユーザー情報を受け取り、キャッシュに保存します。
6. **extdom_extop** プラグインは、LDAP 拡張操作を完了する IdM サーバーの SSSD サービスから情報を受信します。
7. IdM クライアントの SSSD サービスは、LDAP 拡張操作から AD ユーザー情報を受信します。
8. IdM クライアントは、AD ユーザー情報を SSSD キャッシュに保存し、要求したアプリケーションに情報を返します。

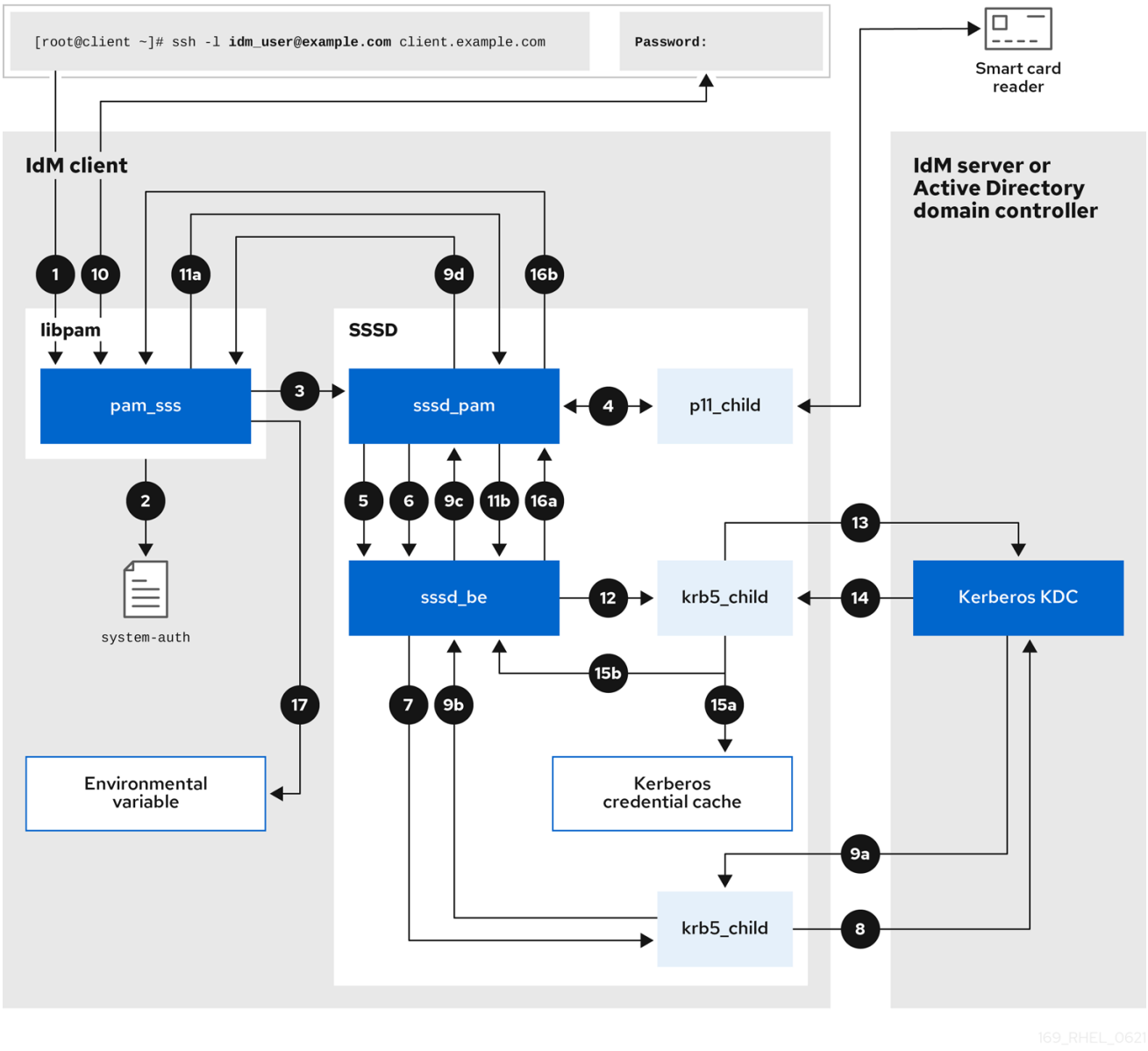
8.3. IDM で SSSD を使用してユーザーとして認証する場合にデータフロー

IdM サーバーまたはクライアントでユーザーとして認証するには、以下のコンポーネントが必要です。

- sshd サービスなどの認証要求を開始するサービス
- PAM (プラグ可能な認証モジュール) ライブラリーとそのモジュール。

- SSSD サービス、そのレスポンダー、およびバックエンド。
- スマートカード認証が設定されている場合は、スマートカードリーダー。
- 認証サーバー:
  - IdM ユーザーは、IdM Kerberos Key Distribution Center (KDC) に対して認証されます。
  - Active Directory (AD) ユーザーは、AD ドメインコントローラー (DC) に対して認証されます。

以下の図は、コマンドラインの SSH サービスを介してホストにローカルでログインしようとすると、情報フローを簡単に認証する必要がある場合を示しています。



1. **ssh** コマンドで認証を試みると、**libpam** ライブラリーがトリガーされます。
2. **libpam** ライブラリー は、認証の試行を要求するサービスに対応する **/etc/pam.d/** ディレクトリーの PAM ファイルを参照します。この例では、ローカルホストの SSH サービス経由で認証されている例では、**pam_sss.so** ライブラリーは **/etc/pam.d/system-auth** 設定ファイルを確認し、SSSD PAM の **pam_sss.so** エントリーを検出します。

```
auth    sufficient    pam_sss.so
```

3. 利用可能な認証方法を判断するため、**libpam** ライブラリーは **pam_sss** モジュールを開き、SSSD サービスの **sssd _pam** PAM レスポンダーに **SSS_PAM_PREAUTH** リクエスト を送信します。

4. スマートカード認証が設定されていると、SSSD サービスは一時的な **p11_child** プロセスを生成し、スマートカードを確認し、そこから証明書を取得します。

5. ユーザーにスマートカード認証が設定されていると、**sssd_pam** レスポンダーは、スマートカードの証明書とユーザーを照合します。**sssd_pam** レスポンダーは、グループメンバーシップがアクセス制御に影響する可能性があるため、ユーザーが属するグループの検索も実行します。

6. **sssd_pam** レスポンダーは、**SSS_PAM_PREAUTH** 要求を **sssd_be** バックエンドレスに送信し、パスワードや 2 要素認証などのサーバーがサポートする認証方法を表示します。SSSD サービスが IPA レスポンダーを使用する IdM 環境では、デフォルトの認証方法は Kerberos です。この例では、ユーザーは簡単な Kerberos パスワードで認証されます。

7. **sssd_be** レスポンダーは一時的な **krb5_child** プロセスを起動します。

8. **krb5_child** プロセスは、IdM サーバーの KDC に連絡して、利用可能な認証方法を確認します。

9. KDC はリクエストに応答します。

    a. **krb5_child** プロセスは応答を評価し、結果を **sssd_be** バックエンドプロセスに送信します。

    b. **sssd_be** バックエンドプロセスが結果を受け取ります。

    c. **sssd_pam** レスポンダーは結果を受け取ります。

    d. **pam_sss** モジュールは結果を受け取ります。

10. ユーザーにパスワード認証が設定されていると、**pam_sss** モジュールにより、パスワードの入力が求められます。スマートカード認証が設定されていると、**pam_sss** モジュールにより、スマートカードの PIN の入力が求められます。

11. モジュールは、ユーザー名とパスワードを使用して **SSS_PAM_** AUTHENTICATE 要求を送信します。これは以下が実行されます。

    a. **sssd_pam** レスポンダー。

    b. **sssd_be** バックエンドプロセス。

12. **sssd_be** プロセスは、KDC に問い合わせる一時的な **krb5_child** プロセスを起動します。

13. **krb5_child** process は、ユーザー名とパスワードを使用して KDC から Kerberos チケット保証チケット (TGT) の取得を試みます。

14. **krb5_child** プロセスは、認証の試行の結果を受け取ります。

15. **krb5_child** プロセス:

    a. TGT を認証情報キャッシュに保存します。

b. **sssd_be** バックエンドプロセスに認証結果を返します。

16. 認証結果は **sssd_be** プロセスから以下を行います。

a. **sssd_pam** レスポンダー。

b. **pam_sss** モジュール。

17. **pam_sss** モジュールは、その他のアプリケーションが参照できるように、環境変数をユーザーの TGT の場所で設定します。

8.4. 認証問題の範囲の制限

ユーザーを正常に認証するには、ユーザー情報を保存するデータベースから SSSD サービスでユーザー情報を取得できる必要があります。以下の手順では、認証プロセスの異なるコンポーネントをテストする手順を説明します。これにより、ユーザーがログインできない場合に認証の問題のスコープを制限する方法を説明します。

手順

1. SSSD サービスおよびそのプロセスが実行していることを確認します。

```
[root@client ~]# pstree -a | grep sssd
  |-sssd -i --logger=files
  |   |-sssd_be --domain implicit_files --uid 0 --gid 0 --logger=files
  |   |-sssd_be --domain example.com --uid 0 --gid 0 --logger=files
  |   |-sssd_ifp --uid 0 --gid 0 --logger=files
  |   |-sssd_nss --uid 0 --gid 0 --logger=files
  |   |-sssd_pac --uid 0 --gid 0 --logger=files
  |   |-sssd_pam --uid 0 --gid 0 --logger=files
  |   |-sssd_ssh --uid 0 --gid 0 --logger=files
  |   `-sssd_sudo --uid 0 --gid 0 --logger=files
  |-sssd_kcm --uid 0 --gid 0 --logger=files
```

2. クライアントが IP アドレスを使用してユーザーデータベースサーバーに接続できることを確認します。

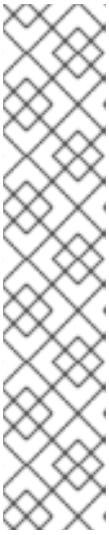
```
[user@client ~]$ ping <IP_address_of_the_database_server>
```

この手順が失敗した場合は、ネットワークとファイアウォール設定が、IdM クライアントとサーバー間の直接通信が許可されていることを確認してください。firewalld の使用および設定 を参照してください。

3. クライアントが、完全修飾ホスト名を使用して IdM LDAP サーバー (IdM ユーザー用) または AD ドメインコントローラー (AD ユーザーの場合) を検出して連絡できることを確認します。

```
[user@client ~]$ dig -t SRV _ldap._tcp.example.com @<name_server>
[user@client ~]$ ping <fully_qualified_host_name_of_the_server>
```

この手順が失敗した場合は、**/etc/resolv.conf** ファイルを含む Dynamic Name Service (DNS) の設定を確認してください。DNS サーバーの順序の設定 を参照してください。



注記

デフォルトでは、SSSD サービスは DNS サービス (SRV) レコードを介して LDAP サーバーと AD DC を自動的に検出しようとします。**sssd.conf** 設定ファイルで以下のオプションを設定すると、SSSD サービスが特定のサーバーを使用するように制限できます。

- **ipa_server = <fully_qualified_host_name_of_the_server>**
- **ad_server = <fully_qualified_host_name_of_the_server>**
- **ldap_uri = <fully_qualified_host_name_of_the_server>**

このオプションを使用する場合は、そのオプションに記載されているサーバーと通信できることを確認します。

4. クライアントが LDAP サーバーに対して認証でき、**ldapsearch** コマンドでユーザー情報を取得できることを確認します。

   a. LDAP サーバーが **server.example.com** などの IdM サーバーである場合は、ホストの Kerberos チケットを取得し、ホスト Kerberos プリンシパルで認証されるデータベース検索を実行します。

   ```
   [user@client ~]$ kinit -k 'host/client.example.com@EXAMPLE.COM'
   [user@client ~]$ ldapsearch -LLL -Y GSSAPI -h server.example.com -b
   “dc=example,dc=com” uid=<user_name>
   ```

   b. LDAP サーバーが **server.example.com** などの Active Directory (AD) Domain Controller (DC) サーバーである場合は、ホストの Kerberos チケットを取得し、ホスト Kerberos プリンシパルで認証されるデータベース検索を実行します。

   ```
   [user@client ~]$ kinit -k 'CLIENT$@AD.EXAMPLE.COM'
   [user@client ~]$ ldapsearch -LLL -Y GSSAPI -h server.ad.example.com -b
   “dc=example,dc=com” sAMAccountname=<user_name>
   ```

   c. LDAP サーバーがプレーン LDAP サーバーであり、**sssd.conf** ファイルに **ldap_default_bind_dn** および **ldap_default_authtok** オプションを設定した場合は、同じ **ldap_default_bind_dn** アカウントとして認証されます。

   ```
   [user@client ~]$ ldapsearch -xLLL -D "cn=ldap_default_bind_dn_value" -W -h
   ldapserver.example.com -b “dc=example,dc=com” uid=<user_name>
   ```

   この手順が失敗した場合は、データベース設定で、ホストが LDAP サーバーを検索できることを確認します。

5. SSSD サービスは Kerberos 暗号化を使用するため、ログインできないユーザーとして Kerberos チケットを取得できます。

   a. LDAP サーバーが IdM サーバーの場合:

   ```
   [user@client ~]$ kinit <user_name>
   ```

   b. LDAP サーバーデータベースが AD サーバーの場合:

   ```
   [user@client ~]$ kinit <user_name@AD.EXAMPLE.COM>
   ```

-

この手順が失敗した場合は、Kerberos サーバーが適切に動作し、すべてのサーバーが同期され、ユーザーアカウントがロックされていないことを確認します。

6. コマンドラインに関するユーザー情報を取得できることを確認します。

```
[user@client ~]$ getent passwd <user_name>
[user@client ~]$ id <user_name>
```

この手順が失敗した場合は、クライアントの SSSD サービスがユーザーデータベースから情報を受信できることを確認します。

a. **/var/log/messages** ログファイルのエラーを確認します。

b. SSSD サービスで詳細なロギングを有効にし、デバッグログを収集して、問題のソースに関するログを確認します。

c. **(オプション)** Red Hat テクニカルサポートケースを作成し、収集したトラブルシューティング情報を提供します。

7. ホストで **sudo** を実行することが許可されている場合は、**sssctl** ユーティリティーを使用して、ユーザーがログインを許可されていることを確認します。

```
[user@client ~]$ sudo sssctl user-checks -a auth -s ssh <user_name>
```

この手順が失敗した場合は、PAM 設定、IdM HBAC ルール、IdM RBAC ルールなどの承認設定を確認します。

a. ユーザーの UID が、**/etc/login.defs** ファイルで定義されている **UID_MIN** 以上であることを確認してください。

b. **/var/log/secure** ログファイルおよび **/var/log/messages** ログファイルで認証エラーを確認します。

c. SSSD サービスで詳細なロギングを有効にし、デバッグログを収集して、問題のソースに関するログを確認します。

d. **(オプション)** Red Hat テクニカルサポートケースを作成し、収集したトラブルシューティング情報を提供します。

関連情報

- sssd.conf ファイルで SSSD の詳細なロギングの有効化
- sssctl コマンドを使用した SSSD の詳細なロギングの有効化
- SSSD サービスからデバッグログを収集し、IdM サーバーによる認証問題のトラブルシューティング
- SSSD サービスからデバッグログを収集し、IdM クライアントによる認証問題のトラブルシューティング

8.5. SSSD ログファイルおよびログレベル

それぞれの SSSD サービスは、**/var/log/sssd/** ディレクトリーに独自のログファイルを記録します。**example.com** IdM ドメインの IdM サーバーのログファイルは、以下のようになります。

```
[root@server ~]# ls -l /var/log/sssd/
total 620
-rw-------. 1 root root      0 Mar 29 09:21 krb5_child.log
-rw-------. 1 root root  14324 Mar 29 09:50 ldap_child.log
-rw-------. 1 root root 212870 Mar 29 09:50 sssd_example.com.log
-rw-------. 1 root root      0 Mar 29 09:21 sssd_ifp.log
-rw-------. 1 root root      0 Mar 29 09:21 sssd_implicit_files.log
-rw-------. 1 root root      0 Mar 29 09:21 sssd.log
-rw-------. 1 root root 219873 Mar 29 10:03 sssd_nss.log
-rw-------. 1 root root      0 Mar 29 09:21 sssd_pac.log
-rw-------. 1 root root  13105 Mar 29 09:21 sssd_pam.log
-rw-------. 1 root root   9390 Mar 29 09:21 sssd_ssh.log
-rw-------. 1 root root      0 Mar 29 09:21 sssd_sudo.log
```

8.5.1. SSSD ログファイルの目的

krb5_child.log

Kerberos 認証に関連する有効期限の短いヘルパープロセスのログファイル。

ldap_child.log

LDAP サーバーとの通信用の Kerberos チケットの取得に関連する短期ヘルパープロセスのログファイル。

sssd_<example.com>.log

sssd.conf ファイルのドメインセクションごとに、SSSD サービスは LDAP サーバーとの通信に関する情報を別のログファイルに記録します。たとえば、**example.com** という名前の IdM ドメインがある環境では、SSSD サービスは **sssd_example.com.log** という名前のファイルのログにその情報を記録します。ホストが **ad.example.com** という名前の AD ドメインと直接統合されている場合は、**sssd_ad.example.com.log** という名前のファイルのログに情報が記録されます。

注記

IdM 環境と、AD ドメインを持つフォレスト間の信頼があると、AD ドメインに関する情報は引き続き IdM ドメインのログファイルに記録されます。

同様に、ホストが AD ドメインに直接統合されている場合は、プライマリードメインのログファイルに、子ドメインに関する情報が書き込まれます。

selinux_child.log

SELinux 情報を取得および設定する短期間ヘルパープロセスのログファイル。

sssd.log

SSSD を監視して、レスポンダーおよびバックエンドプロセスと通信するためのログファイル。

sssd_ifp.log

InfoPipe レスポンダーのログファイル。システムバスからアクセス可能なパブリック D-Bus インターフェイスを提供します。

sssd_nss.log

ユーザーおよびグループ情報を取得する Name Services Switch (NSS) レスポンダーのログファイル。

sssd_pac.log

AD Kerberos チケットから PAC を収集する Microsoft Privilege Attribute Certificate (PAC) レスポンダー用のログファイルは、PAC から PAC に関する情報を取得します。これにより、AD ユーザーを直接要求しないようにします。

sssd_pam.log

PAM (Pluggable Authentication Module) レスポンダー用のログファイルです。

sssd_ssh.log

SSH レスポンダープロセスのログファイル。

8.5.2. SSSD ロギングレベル

デバッグレベルを設定すると、それ下のすべてのデバッグレベルが有効になります。たとえば、debug レベルを 6 に設定すると、デバッグレベル 0 から 5 も有効になります。

表8.1 SSSD ロギングレベル

レベル	説明
0	**致命的な障害が発生しました。**SSSD サービスが起動しなかったり、終了しないようにするエラー。これは、RHEL 8.3 以前のデフォルトのデバッグログレベルです。
1	**重大なエラー**SSSD サービスを終了しないものの、主要な機能は 1 つ以上正しく機能しません。
2	**深刻なエラー。**特定の要求または操作が失敗したことを示すエラー。これは、RHEL 8.4 以降のデフォルトのデバッグログレベルです。
3	**マイナーな障害が発生しました。**レベル 2 で操作の失敗がキャプチャーされたエラー。
4	**設定。**
5	**関数** データ。
6	**操作**関数のメッセージを追跡します。
7	**内部制御** 関数のメッセージトレース。
8	**関数内部** 変数の内容。
9	非常に **低いレベルのトレース** 情報。

8.6. SSSD.CONF ファイルで SSSD の詳細なロギングの有効化

デフォルトでは、RHEL 8.4 以降の SSSD サービスは、重大な失敗 (デバッグレベル 2) のみをログに記録しますが、認証問題のトラブルシューティングに必要な詳細レベルではログに記録されません。

SSSD サービスの再起動時に詳細なロギングを有効にするには、**/etc/sssd/sssd.conf** 設定ファイルの

各セクションに **debug_level=<integer>** オプションを追加します。ここで、**<integer>** の値は 0 から 9 の数字になります。デバッグレベルは最大 3 つのログで、最大 3 つのログで、レベル 8 以上では、多くの詳細なログメッセージを提供します。**レベル 6** は、認証の問題のデバッグに役立ちます。

前提条件

- **sssd.conf** 設定ファイルを編集し、SSSD サービスを再起動するには、root パスワードが必要です。

手順

1. テキストエディターで **/etc/sssd/sssd.conf** ファイルを開きます。
2. **debug_level** オプションをファイルのすべてのセクションに追加し、デバッグレベルを、選択した詳細に設定します。

```
[domain/example.com]
debug_level = 6
id_provider = ipa
...

[sssd]
debug_level = 6
services = nss, pam, ifp, ssh, sudo
domains = example.com

[nss]
debug_level = 6

[pam]
debug_level = 6

[sudo]
debug_level = 6

[ssh]
debug_level = 6

[pac]
debug_level = 6

[ifp]
debug_level = 6
```

3. **sssd.conf** ファイルを保存して閉じます。
4. SSSD サービスを再起動して、新しい設定を読み込みます。

```
[root@server ~]# systemctl restart sssd
```

関連情報

- SSSD ログファイルおよびログレベル

8.7. SSSCTL コマンドを使用した SSSD の詳細なロギングの有効化

デフォルトでは、RHEL 8.4 以降の SSSD サービスは、重大な失敗 (デバッグレベル 2) のみをログに記録しますが、認証問題のトラブルシューティングに必要な詳細レベルではログに記録されません。

sssctl debug-level <integer> コマンドを使用して、コマンドラインで SSSD サービスのデバッグレベルを変更できます。ここで、**<integer>** の値は 0 から 9 の数字になります。デバッグレベルは最大 3 つのログで、最大 3 つのログで、レベル 8 以上では、多くの詳細なログメッセージを提供します。レベル 6 は、認証の問題のデバッグに役立ちます。

前提条件

- **sssctl** コマンドを実行するには、root パスワードが必要です。

手順

- sssctl debug-level コマンドを使用して、希望の詳細度に対して選択したデバッグレベルを設定します。

```
[root@server ~]# sssctl debug-level 6
```

関連情報

- SSSD ログファイルおよびログレベル

8.8. SSSD サービスからデバッグログを収集し、IDM サーバーによる認証問題のトラブルシューティング

IdM ユーザーが IdM サーバーへの認証を試行する際に問題が発生した場合は、サーバー上の SSSD サービスで詳細なデバッグロギングを有効にし、ユーザーに関する情報の取得を試行するログを収集します。

前提条件

- **sssctl** コマンドを実行して SSSD サービスを再起動するには、root パスワードが必要です。

手順

1. IdM サーバーで詳細な SSSD デバッグロギングを有効にします。

```
[root@server ~]# sssctl debug-level 6
```

2. 認証問題が発生しているユーザーの SSSD キャッシュでオブジェクトを無効にするため、LDAP サーバーを省略し、SSSD がすでにキャッシュされている情報を取得しません。

```
[root@server ~]# sssctl cache-expire -u idmuser
```

3. 古い SSSD ログを削除して、トラブルシューティングのデータセットを最小限に抑える。

```
[root@server ~]# sssctl logs-remove
```

4. 認証問題が発生し、試行前後にタイムスタンプを収集する際に、ユーザーが認証問題が発生しようと試みます。これらのタイムスタンプは、データセットのスコープをさらに絞り込むことができます。

```
[root@server sssd]# date; su idmuser; date
Mon Mar 29 15:33:48 EDT 2021
su: user idmuser does not exist
Mon Mar 29 15:33:49 EDT 2021
```

5. **(オプション)** 詳細な SSSD ログの収集を続行しない場合は、デバッグレベルを下げます。

```
[root@server ~]# sssctl debug-level 2
```

6. 障害のある要求に関する情報を SSSD ログで確認します。たとえば、**/var/log/sssd/sssd_example.com.log** ファイルを確認すると、SSSD サービスが **cn=accounts,dc=example,dc=com** LDAP サブツリーでユーザーを見つけられなかったことを示しています。これは、ユーザーが存在しないか、別の場所に存在することを示しています。

```
(Mon Mar 29 15:33:48 2021) [sssd[be[example.com]]] [dp_get_account_info_send] (0x0200):
Got request for [0x1][BE_REQ_USER][name=idmuser@example.com]
...
(Mon Mar 29 15:33:48 2021) [sssd[be[example.com]]] [sdap_get_generic_ext_step] (0x0400):
calling ldap_search_ext with [(&(uid=idmuser)(objectclass=posixAccount)(uid=)(&
(uidNumber=)(!(uidNumber=0))))][cn=accounts,dc=example,dc=com].
(Mon Mar 29 15:33:48 2021) [sssd[be[example.com]]] [sdap_get_generic_op_finished]
(0x0400): Search result: Success(0), no errmsg set
(Mon Mar 29 15:33:48 2021) [sssd[be[example.com]]] [sdap_search_user_process] (0x0400):
Search for users, returned 0 results.
(Mon Mar 29 15:33:48 2021) [sssd[be[example.com]]] [sysdb_search_by_name] (0x0400):
No such entry
(Mon Mar 29 15:33:48 2021) [sssd[be[example.com]]] [sysdb_delete_user] (0x0400): Error: 2
(No such file or directory)
(Mon Mar 29 15:33:48 2021) [sssd[be[example.com]]] [sysdb_search_by_name] (0x0400):
No such entry
(Mon Mar 29 15:33:49 2021) [sssd[be[example.com]]]
[ipa_id_get_account_info_orig_done] (0x0080): Object not found, ending request
```

7. 認証問題の原因を判断できない場合は、以下を行います。

   a. 最近生成した SSSD ログを収集します。

```
[root@server ~]# sssctl logs-fetch sssd-logs-Mar29.tar
```

   b. Red Hat テクニカルサポートケースを作成し、以下を提供します。

      i. SSSD ログ: **sssd-logs-Mar29.tar**

      ii. ログに対応する要求のタイムスタンプおよびユーザー名を含むコンソールの出力。

```
[root@server sssd]# date; id idmuser; date
Mon Mar 29 15:33:48 EDT 2021
id: ‘idmuser’: no such user
Mon Mar 29 15:33:49 EDT 2021
```

8.9. SSSD サービスからデバッグログを収集し、IDM クライアントによる認証問題のトラブルシューティング

IdM クライアントに IdM ユーザーとして認証を試行する際に問題が発生した場合は、IdM サーバーでユーザー情報を取得できることを確認します。IdM サーバーでユーザー情報を取得できない場合は、(IdM サーバーから情報を取得する) IdM クライアントでそれを取得できなくなります。

認証の問題が IdM サーバーから生成されていないことを確認したら、IdM サーバーと IdM クライアントの両方から SSSD デバッグログを収集していました。

前提条件

- IdM サーバーではなく、IdM クライアントで認証の問題のみがあります。
- **sssctl** コマンドを実行して SSSD サービスを再起動するには、root パスワードが必要です。

手順

1. **クライアントで**、テキストエディターで **/etc/sssd/sssd.conf** ファイルを開きます。
2. **クライアントで**、**ipa_server** オプションをファイルの **[domain]** セクションに追加し、IdM サーバーに設定します。これにより、IdM クライアントは他の IdM サーバーの自動検出を避け、このテストを 1 つのクライアントおよびサーバー 1 台だけに制限します。

```
[domain/example.com]
ipa_server = server.example.com
...
```

3. **クライアントで** **sssd.conf** ファイルを保存して閉じます。
4. **クライアントで** SSSD サービスを再起動して、設定の変更を読み込みます。

```
[root@client ~]# systemctl restart sssd
```

5. **サーバーおよびクライアントで**、詳細な SSSD デバッグロギングを有効にします。

```
[root@server ~]# sssctl debug-level 6
```

```
[root@client ~]# sssctl debug-level 6
```

6. **サーバーおよびクライアントで**、認証問題が発生しているユーザーの SSSD キャッシュの検証オブジェクトでは、LDAP データベースを迂回せず、SSSD がすでにキャッシュされています。

```
[root@server ~]# sssctl cache-expire -u idmuser
```

```
[root@client ~]# sssctl cache-expire -u idmuser
```

7. **サーバーおよびクライアントで**、古い SSSD ログを削除して、トラブルシューティングのデータセットを最小限に抑える。

```
[root@server ~]# sssctl logs-remove
```

```
[root@server ~]# sssctl logs-remove
```

8. **クライアントで**、認証問題が発生し、試行前後にタイムスタンプを収集する際に、ユーザーが認証問題が発生しようと試みます。これらのタイムスタンプは、データセットのスコープをさらに絞り込むことができます。

```
[root@client sssd]# date; su idmuser; date
Mon Mar 29 16:20:13 EDT 2021
su: user idmuser does not exist
Mon Mar 29 16:20:14 EDT 2021
```

9. **(オプション) サーバーおよびクライアント** 詳細な SSSD ログの収集したくない場合はデバッグレベルを下げます。

```
[root@server ~]# sssctl debug-level 0
```

```
[root@client ~]# sssctl debug-level 0
```

10. **サーバーおよびクライアントで**、失敗した要求に関する情報を SSSD ログを確認します。
    a. クライアントログのクライアントからの要求を確認します。
    b. サーバーログのクライアントからの要求を確認します。
    c. サーバーログでリクエストの結果を確認します。
    d. サーバーからリクエストの結果を受信するクライアントの結果を確認します。
11. 認証問題の原因を判断できない場合は、以下を行います。
    a. IdM サーバーおよび IdM クライアントで最近生成した SSSD ログを収集します。ホスト名またはロールに応じてラベルを付けます。

```
[root@server ~]# sssctl logs-fetch sssd-logs-server-Mar29.tar
```

```
[root@client ~]# sssctl logs-fetch sssd-logs-client-Mar29.tar
```

    b. Red Hat テクニカルサポートケースを作成し、以下を提供します。
        i. SSSD デバッグログ:
            A. サーバーから **sssd-logs-server-Mar29.tar**
            B. クライアントからの **sssd-logs-client-Mar29.tar**
        ii. ログに対応する要求のタイムスタンプおよびユーザー名を含むコンソールの出力。

```
[root@client sssd]# date; su idmuser; date
Mon Mar 29 16:20:13 EDT 2021
su: user idmuser does not exist
Mon Mar 29 16:20:14 EDT 2021
```

8.10. SSSD バックエンドでのクライアント要求の追跡

SSSD は要求を非同期に処理します。別の要求のメッセージが同じログファイルに追加されるため、一意の要求識別子とクライアント ID を使用して、バックエンドログ内のクライアント要求を追跡できます。一意のリクエスト識別子は、**RID#<integer>** の形式でデバッグログに追加され、クライアント ID はフォーム **[CID #<integer]** に追加されます。これにより、個々の要求に関連するログを分離でき、複数の SSSD コンポーネントからのログファイル全体でリクエストを最初から最後まで追跡できます。

前提条件

- デバッグロギングを有効にし、IdM クライアントから要求が送信されている。
- SSSD ログファイルの内容を表示するための root 権限を持っている。

手順

1. SSSD ログファイルを確認するには、**less** ユーティリティーを使用してログファイルを開きます。たとえば、**/var/log/sssd/sssd_example.com.log** を表示するには、次のコマンドを実行します。

```
[root@server ~]# less /var/log/sssd/sssd_example.com.log
```

2. クライアント要求に関する情報は、SSSD ログを確認します。

```
(2021-07-26 18:26:37): [be[testidm.com]] [dp_req_destructor] (0x0400): [RID#3] Number of active DP request: 0
(2021-07-26 18:26:37): [be[testidm.com]] [dp_req_reply_std] (0x1000): [RID#3] DP Request AccountDomain #3: Returning [Internal Error]: 3,1432158301,GetAccountDomain() not supported
(2021-07-26 18:26:37): [be[testidm.com]] [dp_attach_req] (0x0400): [RID#4] DP Request Account #4: REQ_TRACE: New request. [sssd.nss CID #1] Flags [0x0001].
(2021-07-26 18:26:37): [be[testidm.com]] [dp_attach_req] (0x0400): [RID#4] Number of active DP request: 1
```

SSSD ログファイルからのこの出力例は、2 つの異なる要求について一意の識別子の **RID#3** および **RID#4** を示しています。

ただし、SSSD クライアントインターフェイスへの 1 つのクライアント要求が、バックエンドで複数の要求をトリガーすることが多いため、クライアント要求とバックエンドの要求との間に 1 対 1 の相関関係がなくなります。バックエンド内の複数のリクエストには異なる RID 番号がありますが、最初の各バックエンドリクエストには一意のクライアント ID が含まれているため、管理者は単一のクライアントリクエストに対して複数の RID 番号を追跡できます。

以下の例は、1 つのクライアントリクエスト **[sssd.nss CID #1]** と、バックエンドで生成された複数のリクエスト (**[RID#5]** から **[RID#13]**) を示しています。

```
(2021-10-29 13:24:16): [be[ad.vm]] [dp_attach_req] (0x0400): [RID#5] DP Request [Account #5]: REQ_TRACE: New request. [sssd.nss CID #1] Flags [0x0001].
(2021-10-29 13:24:16): [be[ad.vm]] [dp_attach_req] (0x0400): [RID#6] DP Request [AccountDomain #6]: REQ_TRACE: New request. [sssd.nss CID #1] Flags [0x0001].
(2021-10-29 13:24:16): [be[ad.vm]] [dp_attach_req] (0x0400): [RID#7] DP Request [Account #7]: REQ_TRACE: New request. [sssd.nss CID #1] Flags [0x0001].
(2021-10-29 13:24:17): [be[ad.vm]] [dp_attach_req] (0x0400): [RID#8] DP Request [Initgroups #8]: REQ_TRACE: New request. [sssd.nss CID #1] Flags [0x0001].
(2021-10-29 13:24:17): [be[ad.vm]] [dp_attach_req] (0x0400): [RID#9] DP Request [Account #9]: REQ_TRACE: New request. [sssd.nss CID #1] Flags [0x0001].
(2021-10-29 13:24:17): [be[ad.vm]] [dp_attach_req] (0x0400): [RID#10] DP Request [Account #10]:
```

```
REQ_TRACE: New request. [sssd.nss CID #1] Flags [0x0001].
(2021-10-29 13:24:17): [be[ad.vm]] [dp_attach_req] (0x0400): [RID#11] DP Request [Account #11]:
REQ_TRACE: New request. [sssd.nss CID #1] Flags [0x0001].
(2021-10-29 13:24:17): [be[ad.vm]] [dp_attach_req] (0x0400): [RID#12] DP Request [Account #12]:
REQ_TRACE: New request. [sssd.nss CID #1] Flags [0x0001].
(2021-10-29 13:24:17): [be[ad.vm]] [dp_attach_req] (0x0400): [RID#13] DP Request [Account #13]:
REQ_TRACE: New request. [sssd.nss CID #1] Flags [0x0001].
```

8.11. ログアナライザーツールを使用したクライアント要求の追跡

System Security Services Daemon (SSSD) には、複数の SSSD コンポーネントからのログファイル全体でリクエストを最初から最後まで追跡するために使用できるログ解析ツールが含まれています。

8.11.1. ログアナライザツールのしくみ

ログ解析ツールを使用すると、複数の SSSD コンポーネントからのログファイル全体で SSSD リクエストを最初から最後まで追跡できます。**sssctl analyze** コマンドを使用してアナライザーツールを実行します。

ログアナライザツールは、SSSD の NSS および PAM の問題をトラブルシューティングし、SSSD デバッグログをより簡単に確認するのに役立ちます。SSSD プロセス全体の特定のクライアントリクエストにのみ関連する SSSD ログを抽出して出力できます。

SSSD は、ユーザー認証 (**su**、**ssh**) 情報とは別に、ユーザーおよびグループの ID 情報 (**id**、**getent**) を追跡します。NSS レスポンダのクライアント ID(CID) は PAM レスポンダーの CID とは独立しており、NSS と PAM のリクエストを解析すると重複した数値が表示されます。**--pam** オプションを **sssctl analyze** コマンドとともに使用して、PAM リクエストを確認します。

注記

SSSD メモリーキャッシュから返されたリクエストはログに記録されず、ログアナライザツールで追跡できません。

関連情報

- **sudo sssctl analyze request --help**
- **sudo sssctl analyze --help**
- **sssd.conf** man ページ
- **sssctl** の man ページ

8.11.2. ログアナライザツールの実行

ログアナライザーツールを使用して SSSD でクライアントリクエストを追跡するには、次の手順に従います。

前提条件

- ログ解析機能を有効にするには、**/etc/sssd/sssd.conf** ファイルの [$responder] セクション、および [domain/$domain] セクションで **debug_level** を 7 以上に設定する必要がある。

- 分析するログは、**libtevent** チェーン ID をサポートする互換性のあるバージョンの SSSD、つまり RHEL 8.5 以降の SSSD からのものである必要がある。

手順

1. ログアナライザツールを **list** モードで実行して、追跡しているリクエストのクライアント ID を特定し、**-v** オプションを追加して詳細な出力を表示します。

```
# sssctl analyze request list -v
```

SSSD に対して行われた最近のクライアントリクエストの詳細なリストが表示されます。

注記

PAM リクエストを分析する場合は、**sssctl analyze request list** コマンドを **-pam** オプション付きで実行します。

2. **show [unique client ID]** オプションを指定してログアナライザツールを実行し、指定したクライアント ID 番号に関連するログを表示します。

```
# sssctl analyze request show 20
```

3. 必要に応じて、ログファイルに対してログアナライザツールを実行できます。次に例を示します。

```
# sssctl analyze request --logdir=/tmp/var/log/sssd
```

関連情報

- **sssctl analyze request list --help**
- **sssctl analyze request show --help**
- **sssctl** の man ページ。

8.12. 関連情報

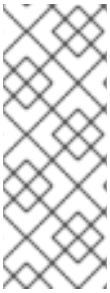
- General SSSD Debugging Procedures

第9章 ANSIBLE PLAYBOOK を使用して IDM を管理する環境の準備

Identity Management (IdM) を管理するシステム管理者は、Red Hat Ansible Engine を使用する際に以下を行うことが推奨されます。

- ホームディレクトリーに Ansible Playbook 専用のサブディレクトリー (例: **~/MyPlaybooks**) を作成します。
- **/usr/share/doc/ansible-freeipa/*** と **/usr/share/doc/rhel-system-roles/*** ディレクトリーおよびサブディレクトリーから **~/MyPlaybooks** ディレクトリーにサンプル Ansible Playbook をコピーして調整します。
- **~/MyPlaybooks** ディレクトリーにインベントリーファイルを追加します。

このプラクティスを使用すると、すべての Playbook を 1 か所で見つけることができます。



注記

マネージドノードで **root** 権限を呼び出さずに **ansible-freeipa** Playbook を実行できます。例外には、**ipaserver**、**ipareplica**、**ipaclient**、**ipasmartcard_server**、**ipasmartcard_client**、および **ipabackup ansible-freeipa** ロールを使用する Playbook が含まれます。これらのロールには、ディレクトリーおよび **dnf** ソフトウェアパッケージマネージャーへの特権アクセスが必要です。

Red Hat Enterprise Linux IdM ドキュメントの Playbook は、次の セキュリティー設定 を前提としています。

- IdM **admin** は、管理ノードのリモート Ansible ユーザーです。
- Ansible vault に暗号化された IdM **admin** パスワードを保存します。
- Ansible vault を保護するパスワードをパスワードファイルに配置しました。
- ローカルの ansible ユーザーを除く全員に対して、vault パスワードファイルへのアクセスをブロックします。
- vault パスワードファイルを定期的に削除して再作成します。

別のセキュリティー設定 も検討してください。

9.1. ANSIBLE PLAYBOOK を使用して IDM を管理するためのコントロールノードと管理ノードの準備

~/MyPlaybooks ディレクトリーを作成し、それを使用して Ansible Playbook を保存および実行できるように設定するには、次の手順に従います。

前提条件

- 管理対象ノードに IdM サーバー (**server.idm.example.com** および **replica.idm.example.com**) をインストールしている。

- DNS およびネットワークを設定し、コントロールノードから直接管理対象ノード (**server.idm.example.com** および **replica.idm.example.com**) にログインすることができる。
- IdM **admin** のパスワードを把握している。

手順

1. **~/MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks
```

2. **~/MyPlaybooks/ansible.cfg** ファイルを以下の内容で作成します。

```
[defaults]
inventory = /home/your_username/MyPlaybooks/inventory
remote_user = admin
```

3. **~/MyPlaybooks/inventory** ファイルを以下の内容で作成します。

```
[eu]
server.idm.example.com

[us]
replica.idm.example.com

[ipaserver:children]
eu
us
```

この設定は、これらの場所にあるホストの 2 つのホストグループ (**eu** と **us**) を定義します。さらに、この設定は、**eu** および **us** グループのすべてのホストを含む **ipaserver** ホストグループを定義します。

4. [オプション] SSH 公開鍵および秘密鍵を作成します。テスト環境でのアクセスを簡素化するには、秘密鍵にパスワードを設定しないでください。

```
$ ssh-keygen
```

5. 各マネージドノードの IdM **admin** アカウントに SSH 公開鍵をコピーします。

```
$ ssh-copy-id admin@server.idm.example.com
$ ssh-copy-id admin@replica.idm.example.com
```

これらのコマンドでは、IdM **管理者** パスワードを入力します。

6. Vault パスワードを含む **password_file** ファイルを作成します。

```
redhat
```

7. ファイルを変更する権限を変更します。

```
$ chmod 0600 password_file
```

8. IdM の **admin** パスワードを保存する **secret.yml** Ansible Vault を作成します。

a. Vault パスワードを保存するように **password_file** を設定します。

```
$ ansible-vault create --vault-password-file=password_file secret.yml
```

b. プロンプトが表示されたら、**secret.yml** ファイルの内容を入力します。

```
ipaadmin_password: Secret123
```

注記

Playbook で暗号化された **ipaadmin_password** を使用するには、**vars_file** ディレクティブを使用する必要があります。たとえば、IdM ユーザーを削除する単純な Playbook は次のようになります。

```
---
- name: Playbook to handle users
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml

  tasks:
  - name: Delete user robot
    ipauser:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: robot
      state: absent
```

Playbook を実行するときに、**--vault-password-file=password_file** オプションを追加して、Ansible に Vault パスワードを使用して **ipaadmin_password** を復号するように指示します。以下に例を示します。

```
ansible-playbook -i inventory --vault-password-file=password_file del-user.yml
```

警告

セキュリティー上の理由から、各セッションの終了時に Vault パスワードファイルを削除し、新しいセッションの開始時に手順 7 ~ 9 を繰り返します。

関連情報

- ansible-freeipa Playbook に必要な認証情報を提供するさまざまな方法
- Ansible Playbook で Identity Management サーバーのインストール
- インベントリーの構築方法

9.2. ANSIBLE-FREEIPA PLAYBOOK に必要な認証情報を提供するさまざまな方法

ansible-freeipa ロールおよびモジュールを使用する Playbook の実行に必要な認証情報を提供するための様々な方法には長所と短所があります。

Playbook にパスワードを平文で保存する

利点:

- Playbook を実行するたびにプロンプトが表示されません。
- 実装が簡単。

短所:

- ファイルにアクセスできるすべてのユーザーがパスワードを読み取ることができます。内部または外部のリポジトリーなどで間違った権限を設定してファイルを共有すると、セキュリティーが損なわれる可能性があります。
- 高いメンテナンス作業: パスワードが変更された場合は、すべての Playbook で変更する必要があります。

Playbook の実行時に対話的にパスワードを入力する

利点:

- パスワードはどこにも保存されないため、誰もパスワードを盗むことはできません。
- パスワードを簡単に更新できます。
- 実装が簡単。

短所:

- スクリプトで Ansible Playbook を使用している場合は、パスワードを対話的に入力する必要があると不便な場合があります。

パスワードを Ansible Vault に保存し、Vault パスワードをファイルに保存します。

利点:

- ユーザーパスワードは暗号化されて保存されます。
- 新しい Ansible Vault を作成することで、ユーザーパスワードを簡単に更新できます。
- **ansible-vault rekey --new-vault-password-file=NEW_VAULT_PASSWORD_FILE secret.yml** コマンドを使用して、ansible Vault を保護するパスワードファイルを簡単に更新できます。
- スクリプトで Ansible Playbook を使用している場合は、Ansible Vault を対話的に保護するパスワードを入力する必要がないのは便利です。

短所:

- 機密性の高いプレーンテキストパスワードを含むファイルは、ファイルのアクセス許可やその他のセキュリティー対策によって保護することが重要です。

パスワードを Ansible Vault に保存し、Vault のパスワードを対話的に入力する

利点:

- ユーザーパスワードは暗号化されて保存されます。
- Vault のパスワードはどこにも保存されていないため、誰も盗むことはできません。
- 新しい Ansible Vault を作成することで、ユーザーパスワードを簡単に更新できます。
- **ansible-vault rekey file_name** コマンドを使用して、Vault パスワードも簡単に更新できます。

短所:

- スクリプトで Ansible Playbook を使用している場合は、Vault のパスワードを対話的に入力する必要があると不便な場合があります。

関連情報

- Ansible Playbook を使用して IdM を管理するためのコントロールノードと管理ノードの準備
- What is Zero trust?
- Ansible Vault による機密データの保護

第10章 ANSIBLE PLAYBOOK でのグローバル IDM 設定

Ansible **設定** モジュールを使用すると、Identity Management (IdM) のグローバル設定パラメーターを取得および設定できます。

- Ansible Playbook での IdM 設定の取得
- Ansible Playbook での IdM CA 更新サーバーの設定
- Ansible Playbook での IdM ユーザーのデフォルトシェルの設定
- Ansible を使用した IdM ドメインの NETBIOS 名の設定
- Ansible を使用して IdM ユーザーとグループに SID があることを確認する

10.1. ANSIBLE PLAYBOOK での IDM 設定の取得

以下の手順では、Ansible Playbook を使用して、現在のグローバル IdM 設定に関する情報を取得する方法を説明します。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. Ansible Playbook ファイル **/usr/share/doc/ansible-freeipa/playbooks/config/retrieve-config.yml** を開いて編集します。

```
---
- name: Playbook to handle global IdM configuration
  hosts: ipaserver
  become: no
  gather_facts: no

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Query IPA global configuration
    ipaconfig:
      ipaadmin_password: "{{ ipaadmin_password }}"
```

```
    register: serverconfig

  - debug:
      msg: "{{ serverconfig }}"
```

2. 以下を変更してファイルを調整します。
   - IdM 管理者のパスワード。
   - その他の値 (必要な場合)。
3. ファイルを保存します。
4. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file /usr/share/doc/ansible-
freeipa/playbooks/config/retrieve-config.yml
[...]
TASK [debug]
ok: [server.idm.example.com] => {
    "msg": {
        "ansible_facts": {
            "discovered_interpreter_
        },
        "changed": false,
        "config": {
            "ca_renewal_master_server": "server.idm.example.com",
            "configstring": [
                "AllowNThash",
                "KDC:Disable Last Success"
            ],
            "defaultgroup": "ipausers",
            "defaultshell": "/bin/bash",
            "emaildomain": "idm.example.com",
            "enable_migration": false,
            "groupsearch": [
                "cn",
                "description"
            ],
            "homedirectory": "/home",
            "maxhostname": "64",
            "maxusername": "64",
            "pac_type": [
                "MS-PAC",
                "nfs:NONE"
            ],
            "pwdexpnotify": "4",
            "searchrecordslimit": "100",
            "searchtimelimit": "2",
            "selinuxusermapdefault": "unconfined_u:s0-s0:c0.c1023",
            "selinuxusermaporder": [
                "guest_u:s0$xguest_u:s0$user_
            ],
            "usersearch": [
```

```
                "uid",
                "givenname",
                "sn",
                "telephonenumber",
                "ou",
                "title"
            ]
        },
        "failed": false
    }
}
```

10.2. ANSIBLE PLAYBOOK での IDM CA 更新サーバーの設定

組み込みの認証局 (CA) を使用する Identity Management (IdM) デプロイメントでは、CA 更新サーバーが IdM システム証明書を維持および更新します。IdM デプロイメントを確実に堅牢化します。

IdM CA 更新サーバーロールの詳細は、IdM CA 更新サーバーの使用 を参照してください。

以下の手順では、Ansible Playbook を使用して IdM CA 更新サーバーを設定する方法を説明します。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. 必要に応じて、現在の IdM CA 更新サーバーを特定します。

```
$ ipa config-show | grep 'CA renewal'
  IPA CA renewal master: server.idm.example.com
```

2. **inventory.file** などのインベントリーファイルを作成して、そのファイルに **ipaserver** を定義します。

```
[ipaserver]
server.idm.example.com
```

3. Ansible Playbook ファイル **/usr/share/doc/ansible-freeipa/playbooks/config/set-ca-renewal-master-server.yml** を開いて編集します。

```
---
- name: Playbook to handle global DNS configuration
  hosts: ipaserver
  become: no
  gather_facts: no
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml

  tasks:
  - name: set ca_renewal_master_server
    ipaconfig:
      ipaadmin_password: "{{ ipaadmin_password }}"
      ca_renewal_master_server: carenewal.idm.example.com
```

4. 以下を変更してファイルを調整します。
   - **ipaadmin_password** 変数で設定した IdM 管理者のパスワード
   - **ca_renewal_master_server** 変数で設定した CA 更新サーバーの名前。
5. ファイルを保存します。
6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file /usr/share/doc/ansible-
freeipa/playbooks/config/set-ca-renewal-master-server.yml
```

検証手順

CA 更新サーバーが変更されたことを確認します。

1. IdM 管理者として **ipaserver** にログインします。

```
$ ssh admin@server.idm.example.com
Password:
[admin@server /]$
```

2. IdM CA 更新サーバーの ID を要求します。

```
$ ipa config-show | grep 'CA renewal'
IPA CA renewal master:  carenewal.idm.example.com
```

この出力には、**carenewal.idm.example.com** サーバーが新しい CA 更新サーバーであることが分かります。

10.3. ANSIBLE PLAYBOOK での IDM ユーザーのデフォルトシェルの設定

シェルとは、コマンドを受け入れて変換するプログラムです。Red Hat Enterprise Linux (RHEL) では、**bash**、**sh**、**ksh**、**zsh**、**fish** など、複数のシェルが利用できます。**Bash** (または **/bin/bash**) は、ほとんどの Linux システムで一般的なシェルです。通常は、RHEL のユーザーアカウントのデフォルトシェルです。

以下の手順では、Ansible Playbook を使用して、IdM ユーザーのデフォルトシェルとして別のシェルである **sh** を設定する方法を説明します。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. 必要に応じて、Ansible Playbook **retrieve-config.yml** を使用して、IdM ユーザーの現在のシェルを特定します。詳細は、Ansible Playbook での IdM 設定の取得 を参照してください。
2. **inventory.file** などのインベントリーファイルを作成して、そのファイルに **ipaserver** を定義します。

```
[ipaserver]
server.idm.example.com
```

3. Ansible Playbook **/usr/share/doc/ansible-freeipa/playbooks/config/ensure-config-options-are-set.yml** ファイルを開いて編集します。

```
---
- name: Playbook to ensure some config options are set
  hosts: ipaserver
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml

  tasks:
  # Set defaultlogin and maxusername
  - ipaconfig:
      ipaadmin_password: "{{ ipaadmin_password }}"
      defaultshell: /bin/bash
      maxusername: 64
```

4. 以下を変更してファイルを調整します。
  - **ipaadmin_password** 変数で設定した IdM 管理者のパスワード
  - **defaultshell** 変数で設定されている IdM ユーザーのデフォルトのシェルが **/bin/sh** に設定されます。

5. ファイルを保存します。

6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file /usr/share/doc/ansible-
freeipa/playbooks/config/ensure-config-options-are-set.yml
```

検証手順

IdM で新しいセッションを開始すると、デフォルトのユーザーシェルが変更されていることを確認できます。

1. IdM 管理者として **ipaserver** にログインします。

```
$ ssh admin@server.idm.example.com
Password:
[admin@server /]$
```

2. 現在のシェルを表示します。

```
[admin@server /]$ echo "$SHELL"
/bin/sh
```

ログインしているユーザーが **sh** シェルを使用している。

10.4. ANSIBLE を使用した IDM ドメインの NETBIOS 名の設定

NetBIOS 名は、Microsoft Windows (SMB) タイプの共有およびメッセージングに使用されます。NetBIOS 名を使用して、ドライブをマップしたり、プリンターに接続したりできます。

以下の手順に従って、Ansible Playbook を使用して Identity Management (IdM) ドメインの NetBIOS 名を設定します。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - **ansible-freeipa** パッケージをインストールしている。

想定条件

- この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
- この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されており、ボールトファイルのパスワードを知っていることを前提としています。

手順

1. **~/MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **netbios-domain-name-present.yml** Ansible Playbook ファイルを作成します。
3. 以下の内容をファイルに追加します。

```
---
- name: Playbook to change IdM domain netbios name
  hosts: ipaserver
  become: no
  gather_facts: no


  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml

  tasks:
    - name: Set IdM domain netbios name
      ipaconfig:
        ipaadmin_password: "{{ ipaadmin_password }}"
        netbios_name: IPADOM
```

4. ファイルを保存します。
5. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory netbios-domain-name-present.yml
```

プロンプトが表示されたら、ボールトファイルのパスワードを入力します。

関連情報

- NetBIOS 名を設定するためのガイドライン

10.5. ANSIBLE を使用して IDM ユーザーとグループに SID があることを確認する

Identity Management (IdM) サーバーは、ローカルドメインの ID 範囲のデータに基づいて、一意のセキュリティー識別子 (SID) を IdM ユーザーおよびグループに内部的に割り当てることができます。SID は、ユーザーオブジェクトとグループオブジェクトに格納されます。

IdM ユーザーとグループに SID があることを確認する目的は、特権属性証明書 (PAC) の生成を許可することです。これは、IdM-IdM 信頼への最初のステップです。IdM ユーザーおよびグループが SID を持っている場合、IdM は PAC データを使用して Kerberos チケットを発行できます。

次の目標を達成するには、次の手順に従ってください。

- 既存の IdM ユーザーおよびユーザーグループの SID を生成します。
- IdM の新しいユーザーおよびグループの SID の生成を有効にします。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - **ansible-freeipa** パッケージをインストールしている。

想定条件

- この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
- この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されており、ボールトファイルのパスワードを知っていることを前提としています。

手順

1. **~/MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **sids-for-users-and-groups-present.yml** Ansible Playbook ファイルを作成します。
3. 以下の内容をファイルに追加します。

```
---
- name: Playbook to ensure SIDs are enabled and users and groups have SIDs
  hosts: ipaserver
  become: no
  gather_facts: no

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml

  tasks:
    - name: Enable SID and generate users and groups SIDS
      ipaconfig:
        ipaadmin_password: "{{ ipaadmin_password }}"
        enable_sid: true
        add_sids: true
```

enable_sid 変数は、将来の IdM ユーザーおよびグループの SID 生成を有効にします。**add_sids** 変数は、既存の IdM ユーザーおよびグループの SID を生成します。

注記

add_sids: true を使用する場合は、**enable_sid** 変数を **true** に設定する必要もあります。

4. ファイルを保存します。
5. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory sids-for-users-and-groups-present.yml
```

- プロンプトが表示されたら、ボールトファイルのパスワードを入力します。

関連情報

- IdM ID 範囲におけるセキュリティーおよび相対識別子のロール.

10.6. 関連情報

- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-config.md** を参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/config** ディレクトリーのサンプルの Playbook を参照してください。

第11章 コマンドラインでユーザーアカウントの管理

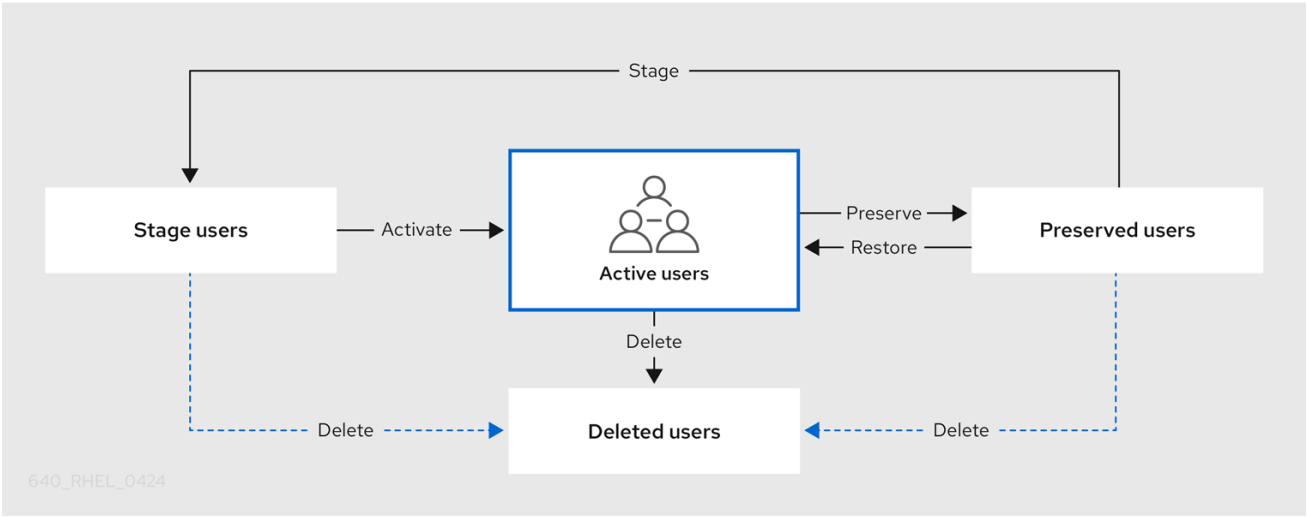
IdM (Identity Management) のユーザーライフサイクルには、次のようないくつかの段階があります。

- ユーザーアカウントを作成する
- ステージユーザーアカウントをアクティベートする
- ユーザーアカウントを保存する
- アクティブユーザー、ステージユーザー、または保存済みユーザーのアカウントを削除する
- 保存済みユーザーアカウントを復元する

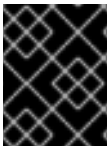
11.1. ユーザーのライフサイクル

Identity Management (IdM) は、次の 3 つのユーザーアカウント状態に対応します

- **ステージ** ユーザーは認証できません。これは初期状態です。アクティブユーザーに必要なユーザーアカウントプロパティーをすべて設定できるわけではありません (例: グループメンバーシップ)。
- **アクティブ** ユーザーは認証が可能です。必要なユーザーアカウントプロパティーはすべて、この状態で設定する必要があります。
- **保存済み** ユーザーは、以前にアクティブであったユーザーで、現在は非アクティブであるとみなされており、IdM への認証ができません。保存済みユーザーには、アクティブユーザーのときに有効になっていたアカウントプロパティーの大部分が保持されていますが、ユーザーグループからは除外されています。



IdM データベースからユーザーエントリーを完全に削除できます。



重要

削除したユーザーアカウントを復元することはできません。ユーザーアカウントを削除すると、そのアカウントに関連する情報がすべて完全に失われます。

新規管理者は、デフォルトの管理ユーザーなど、管理者権限を持つユーザーのみが作成できます。すべての管理者アカウントを誤って削除した場合は、Directory Manager が、Directory Server に新しい管理者を手動で作成する必要があります。

警告

admin ユーザーを削除しないでください。**admin** は IdM で必要な事前定義ユーザーであるため、この操作では特定のコマンドで問題が生じます。別の **admin** ユーザーを定義して使用する場合は、管理者権限を少なくとも 1 つのユーザーに付与してから、**ipa user-disable admin** を使用して、事前定義された admin ユーザーを無効にします。

警告

ローカルユーザーを IdM に追加しないでください。Name Service Switch (NSS) は、ローカルユーザーとグループを解決する前に、IdM ユーザーとグループを常に解決します。つまり、たとえば IdM グループのメンバーシップは、ローカルユーザーでは機能しません。

11.2. コマンドラインを使用したユーザーの追加

以下のようにユーザーを追加できます。

- **アクティブ** - ユーザーがアクティブに使用できるユーザーアカウント
- **ステージ** - ユーザーは、このアカウントを使用できません。新規ユーザーアカウントを準備する場合は、このタイプを使用します。ユーザーがアカウントを使用する準備ができると、アクティベートできます。

以下の手順では、**ipa user-add** コマンドを使用して、アクティブなユーザーを IdM サーバーに追加する方法を説明します。

同様に、**ipa stageuser-add** コマンドでステージユーザーアカウントを作成できます。

注記

IdM は、一意のユーザー ID (UID) を新しいユーザーアカウントに自動的に割り当てます。手動で行うこともできますが、サーバーは UID 番号が一意かどうかを検証しません。このため、複数のユーザーエントリーに同じ ID 番号が割り当てられる可能性があります。Red Hat は、複数のエントリーに同じ UID を割り当てることがないようにすることを推奨します。

前提条件

- IdM、またはユーザー管理者ロールを管理する管理者権限

- Kerberos チケットを取得している。詳細は、Using kinit to log in to IdM manually を参照してください。

手順

1. 端末を開き、IdM サーバーに接続します。
2. ユーザーのログイン、ユーザーの名前、および名字を追加します。メールアドレスを追加することもできます。

```
$ ipa user-add user_login --first=first_name --last=last_name --email=email_address
```

IdM は、以下の正規表現で説明できるユーザー名をサポートします。

```
[a-zA-Z0-9_.][a-zA-Z0-9_.-]{0,252}[a-zA-Z0-9_.$-]?
```

注記

ユーザー名の末尾がドル記号 ($) で終わる場合は、Samba 3.x マシンでのサポートが有効になります。

大文字を含むユーザー名を追加すると、IdM が名前を保存する際に自動的に小文字に変換されます。したがって、IdM にログインする場合は、常にユーザー名を小文字で入力する必要があります。また、**user** と **User** など、大文字と小文字のみが異なるユーザー名を追加することはできません。

ユーザー名のデフォルトの長さは、最大 32 文字です。これを変更するには、**ipa config-mod --maxusername** コマンドを使用します。たとえば、ユーザー名の最大長を 64 文字にするには、次のコマンドを実行します。

```
$ ipa config-mod --maxusername=64
 Maximum username length: 64
...
```

ipa user-add コマンドには、多くのパラメーターが含まれます。リストを表示するには、ipa help コマンドを使用します。

```
$ ipa help user-add
```

ipa help コマンドの詳細は、IPA のヘルプとは を参照してください。

IdM ユーザーアカウントをリスト表示して、新規ユーザーアカウントが正常に作成されたかどうかを確認できます。

```
$ ipa user-find
```

このコマンドは、すべてのユーザーアカウントと、その詳細をリストで表示します。

11.3. コマンドラインでユーザーのアクティベート

ステージからアクティブに移行してユーザーアカウントをアクティベートするには、**ipa stageuser-activate** コマンドを使用します。

前提条件

- IdM、またはユーザー管理者ロールを管理する管理者権限
- Kerberos チケットを取得している。詳細は、Using kinit to log in to IdM manually を参照してください。

手順

1. 端末を開き、IdM サーバーに接続します。
2. 次のコマンドで、ユーザーアカウントをアクティベートします。

```
$ ipa stageuser-activate user_login
-------------------------
Stage user user_login activated
-------------------------
...
```

IdM ユーザーアカウントをリスト表示して、新規ユーザーアカウントが正常に作成されたかどうかを確認できます。

```
$ ipa user-find
```

このコマンドは、すべてのユーザーアカウントと、その詳細をリストで表示します。

11.4. コマンドラインでユーザーの保存

ユーザーアカウントを削除しても、保存しておくことはできますが、後で復元するオプションはそのままにしておきます。ユーザーアカウントを保持するには、**ipa user-del** コマンドまたは **ipa stageuser-del** コマンドで、**--preserve** オプションを使用します。

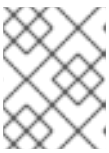
前提条件

- IdM、またはユーザー管理者ロールを管理する管理者権限
- Kerberos チケットを取得している。詳細は、Using kinit to log in to IdM manually を参照してください。

手順

1. 端末を開き、IdM サーバーに接続します。
2. 次のコマンドで、ユーザーアカウントを保存します。

```
$ ipa user-del --preserve user_login
--------------------
Deleted user "user_login"
--------------------
```



注記

ユーザーアカウントが削除されたという出力が表示されたにもかかわらず、アカウントは保持されています。

11.5. コマンドラインを使用したユーザーの削除

IdM (Identity Management) を使用すると、ユーザーを完全に削除できます。以下を削除できます。

- アクティブユーザーの場合 - **ipa user-del**
- ステージユーザーの場合 - **ipa stageuser-del**
- 保存済みユーザーの場合 - **ipa user-del**

複数のユーザーを削除するときは、**--continue** オプションを使用して、エラーに関係なくコマンドを続行します。成功および失敗した操作の概要は、コマンドが完了したときに標準出力ストリーム (**stdout**) に出力されます。

```
$ ipa user-del --continue user1 user2 user3
```

--continue を使用しないと、コマンドはエラーが発生するまでユーザーの削除を続行し、停止と終了を行います。

前提条件

- IdM、またはユーザー管理者ロールを管理する管理者権限
- Kerberos チケットを取得している。詳細は、Using kinit to log in to IdM manually を参照してください。

手順

1. 端末を開き、IdM サーバーに接続します。
2. 次のコマンドで、ユーザーアカウントを削除します。

```
$ ipa user-del user_login
--------------------
Deleted user "user_login"
--------------------
```

ユーザーアカウントは、IdM から完全に削除されました。

11.6. コマンドラインでユーザーの復元

保存済みユーザーは、以下のステータスに復元できます。

- アクティブユーザー - **ipa user-undel**
- ステージユーザー - **ipa user-stage**

ユーザーアカウントを復元しても、そのアカウントの以前の属性がすべて復元されるわけではありません。たとえば、ユーザーのパスワードが復元されず、再設定する必要があります。

前提条件

- IdM、またはユーザー管理者ロールを管理する管理者権限

- Kerberos チケットを取得している。詳細は、Using kinit to log in to IdM manually を参照してください。

手順

1. 端末を開き、IdM サーバーに接続します。
2. 次のコマンドで、ユーザーアカウントをアクティベートします。

```
$ ipa user-undel user_login
----------------------------
Undeleted user account "user_login"
----------------------------
```

または、ユーザーアカウントをステージユーザーとして復元することもできます。

```
$ ipa user-stage user_login
----------------------------
Staged user account "user_login"
----------------------------
```

検証手順

- IdM ユーザーアカウントをリスト表示して、新規ユーザーアカウントが正常に作成されたかどうかを確認できます。

```
$ ipa user-find
```

このコマンドは、すべてのユーザーアカウントと、その詳細をリストで表示します。

第12章 IDM WEB UI でユーザーアカウントの管理

Identity Management (IdM) は、さまざまなユーザーのライフサイクル状況の管理に役立つ 複数のステージ を提供します。

ユーザーアカウントの作成

従業員が新しい会社で働き始める前に ステージユーザーアカウントを作成 し、従業員の初出勤日に合わせてアカウントをアクティベートできるように準備します。
この手順を省略し、アクティブなユーザーアカウントを直接作成できるようにします。この手順は、ステージユーザーアカウントの作成に類似しています。

ユーザーアカウントをアクティベートする

従業員の最初の就業日に アカウントをアクティベート します。

ユーザーアカウントを無効にする

ユーザーが数か月間育児休暇を取得する場合は、一時的にアカウントを無効にする 必要があります。

ユーザーアカウントを有効にする

ユーザーが戻ってきたら、アカウントを再度有効にする 必要があります。

ユーザーアカウントを保存する

ユーザーが会社を辞める場合は、しばらくしてから会社に戻ってくる可能性を考慮して、アカウントを復元することができる状態で削除する 必要があります。

ユーザーアカウントを復元する

2 年後にユーザーが復職する場合は、保存済みアカウントを復元 する必要があります。

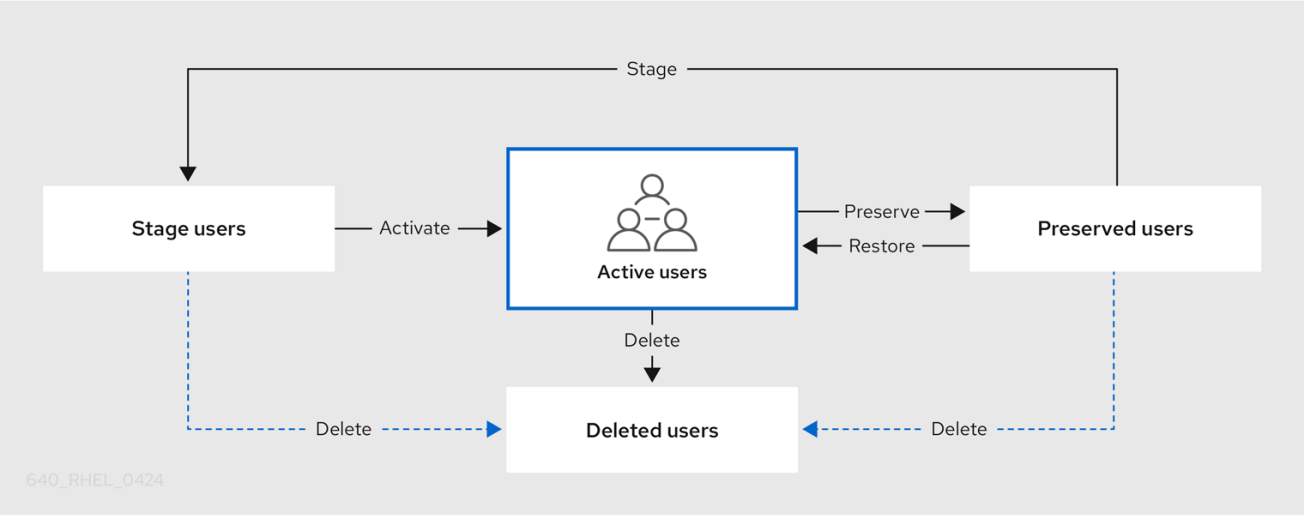
ユーザーアカウントを削除する

従業員が解雇された場合は、バックアップなしで アカウントを削除します。

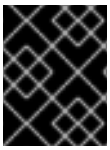
12.1. ユーザーのライフサイクル

Identity Management (IdM) は、次の 3 つのユーザーアカウント状態に対応します

- **ステージ** ユーザーは認証できません。これは初期状態です。アクティブユーザーに必要なユーザーアカウントプロパティーをすべて設定できるわけではありません (例: グループメンバーシップ)。
- **アクティブ** ユーザーは認証が可能です。必要なユーザーアカウントプロパティーはすべて、この状態で設定する必要があります。
- **保存済み** ユーザーは、以前にアクティブであったユーザーで、現在は非アクティブであるとみなされており、IdM への認証ができません。保存済みユーザーには、アクティブユーザーのときに有効になっていたアカウントプロパティーの大部分が保持されていますが、ユーザーグループからは除外されています。



IdM データベースからユーザーエントリーを完全に削除できます。



重要

削除したユーザーアカウントを復元することはできません。ユーザーアカウントを削除すると、そのアカウントに関連する情報がすべて完全に失われます。

新規管理者は、デフォルトの管理ユーザーなど、管理者権限を持つユーザーのみが作成できます。すべての管理者アカウントを誤って削除した場合は、Directory Manager が、Directory Server に新しい管理者を手動で作成する必要があります。

警告

admin ユーザーを削除しないでください。**admin** は IdM で必要な事前定義ユーザーであるため、この操作では特定のコマンドで問題が生じます。別の **admin** ユーザーを定義して使用する場合は、管理者権限を少なくとも 1 つのユーザーに付与してから、**ipa user-disable admin** を使用して、事前定義された admin ユーザーを無効にします。

警告

ローカルユーザーを IdM に追加しないでください。Name Service Switch (NSS) は、ローカルユーザーとグループを解決する前に、IdM ユーザーとグループを常に解決します。つまり、たとえば IdM グループのメンバーシップは、ローカルユーザーでは機能しません。

12.2. WEB UI でユーザーの追加

通常は、新入社員が働き始める前に、新しいユーザーアカウントを作成する必要があります。このようなステージアカウントにはアクセスできず、後でアクティベートする必要があります。

注記

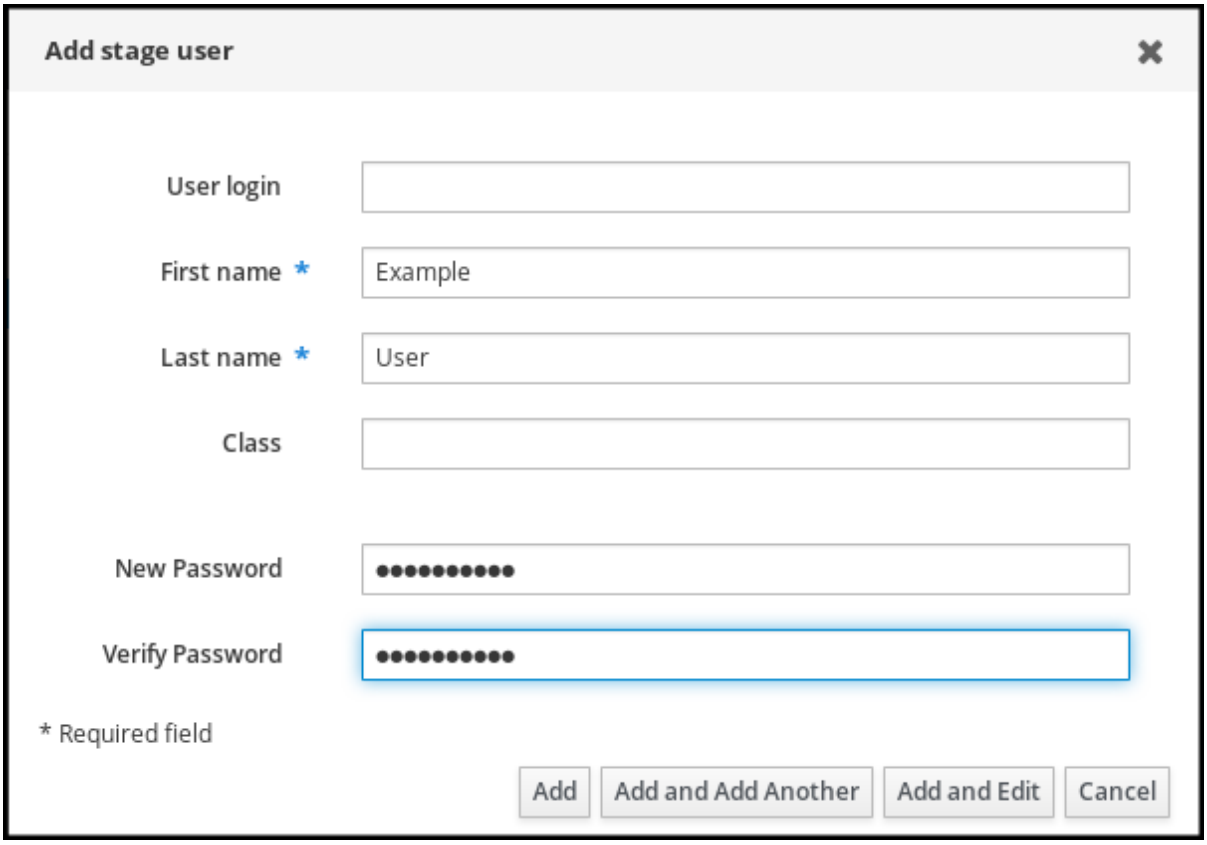
または、直接、アクティブなユーザーアカウントを作成できます。アクティブユーザーを追加する場合は、以下の手順に従って、**アクティブユーザー** タブでユーザーアカウントを追加します。

前提条件

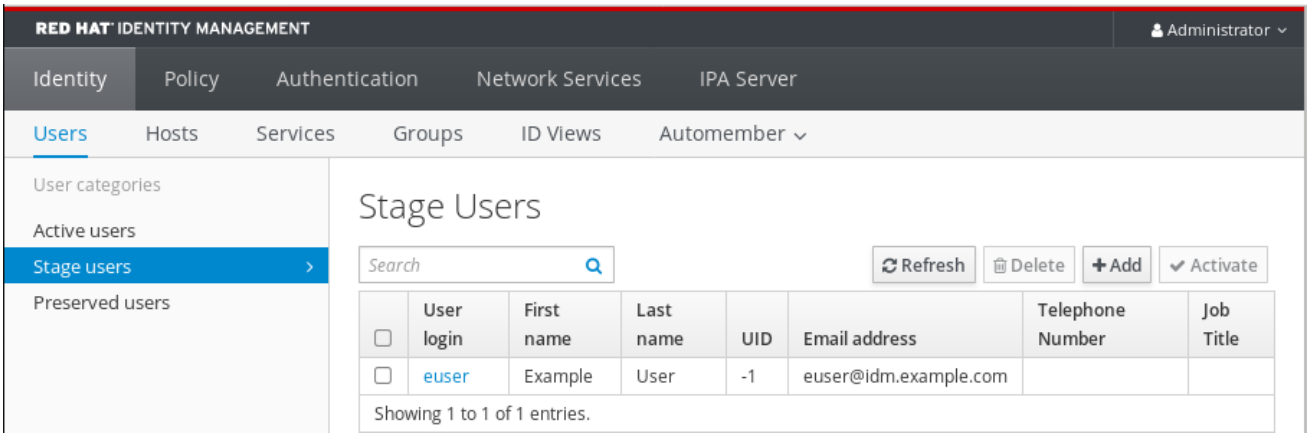
- IdM、またはユーザー管理者ロールを管理する管理者権限

手順

1. IdM Web UI にログインします。
   詳細は Web ブラウザーで IdM Web UI へのアクセス を参照してください。
2. **ユーザー → ステージユーザー** タブに移動します。
   または、**ユーザー → アクティブユーザー** にユーザーアカウントを追加できますが、アカウントにユーザーグループを追加することはできません。
3. **+ 追加** アイコンをクリックします。
4. **ステージユーザーの追加** ダイアログボックスで、新規ユーザーの **名前** と **名字** を入力します。
5. (必要に応じて) **ユーザーログイン** フィールドにログイン名を追加します。
   空のままにすると、IdM サーバーは、名字の前に、名前の最初の 1 文字が追加された形式で、ログイン名を作成します。ログイン名には、32 文字まで使用できます。
6. (必要に応じて) GID ドロップダウンメニューで、ユーザーに含まれるグループを選択します。
7. [オプション] **パスワード** および **パスワードの確認** フィールドに、パスワードを入力して確定し、両方が一致していることを確認します。
8. **Add** ボタンをクリックします。



この時点では、**ステージユーザー** テーブルでユーザーアカウントを確認できます。



注記

ユーザー名をクリックすると、電話番号、住所、職業の追加などの詳細設定を編集できます。

12.3. IDM WEB UI でステージユーザーのアクティベート

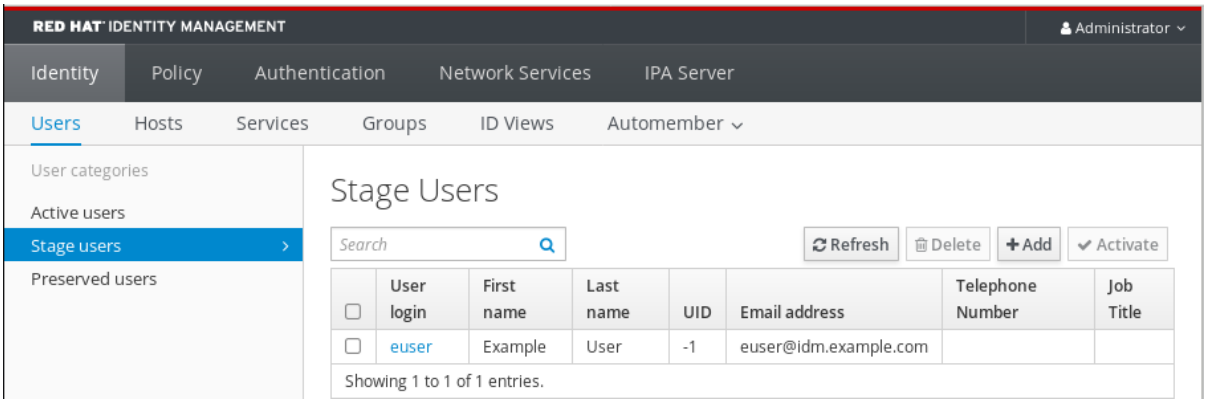
ユーザーが IdM にログインする前、およびユーザーを IdM グループに追加する前に、この手順に従ってステージユーザーアカウントをアクティベートする必要があります。

前提条件

- IdM Web UI、またはユーザー管理者ロールを管理する管理者権限
- IdM に、1 つ以上のステージユーザーアカウント

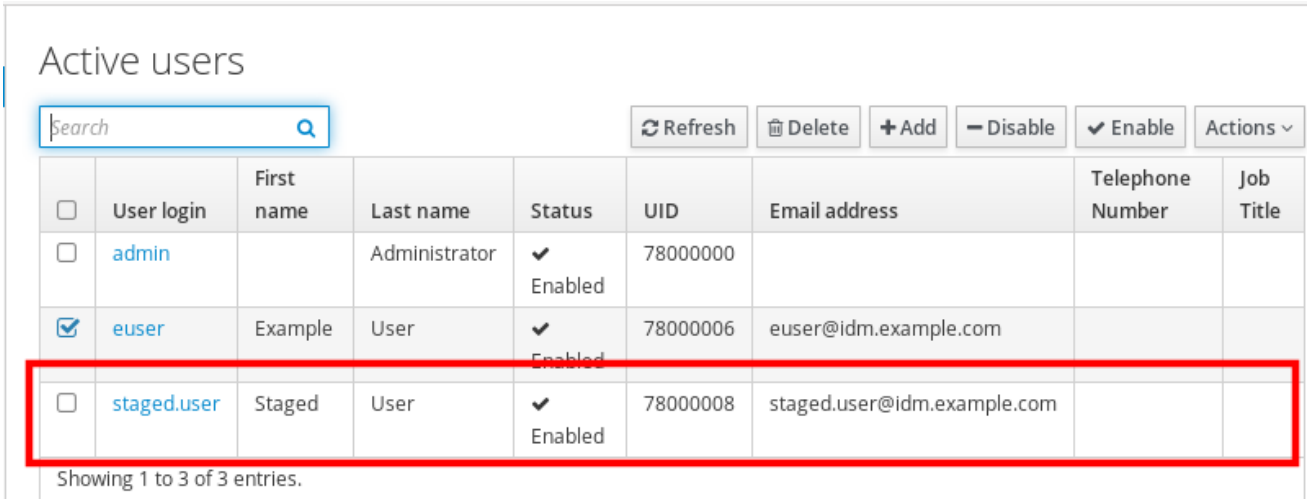
手順

1. IdM Web UI にログインします。
   詳細は Web ブラウザーで IdM Web UI へのアクセス を参照してください。
2. **ユーザー → ステージユーザー** タブに移動します。
3. 有効にするユーザーアカウントのチェックボックスをクリックします。
4. **アクティベート** ボタンをクリックします。



5. **Confirmation** ダイアログボックスで **OK** をクリックします。

アクティベーションに成功したら、IdM Web UI により、ユーザーがアクティベートされ、ユーザーアカウントが **アクティブユーザー** に移動したことを示す緑色の確認が表示されます。このアカウントはアクティブで、ユーザーは IdM ドメインと IdM Web UI に対して認証できます。ユーザーは、初回ログイン時にパスワードを変更するように求められます。



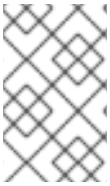
注記

このステージで、アクティブなユーザーアカウントをユーザーグループに追加できます。

12.4. WEB UI でのユーザーアカウントの無効化

アクティブなユーザーアカウントを無効にできます。ユーザーアカウントを無効にすると、ユーザーアカウントはアカウントを非アクティブにできるため、そのユーザーアカウントを使用して Kerberos などの IdM サービスを認証および使用したり、タスクを実行することができません。

無効にしたユーザーアカウントはそのまま IdM に残り、関連する情報は何も変更しません。保存済みユーザーアカウントとは異なり、無効にしたユーザーアカウントはアクティブな状態のままとなり、ユーザーグループのメンバーになります。



注記

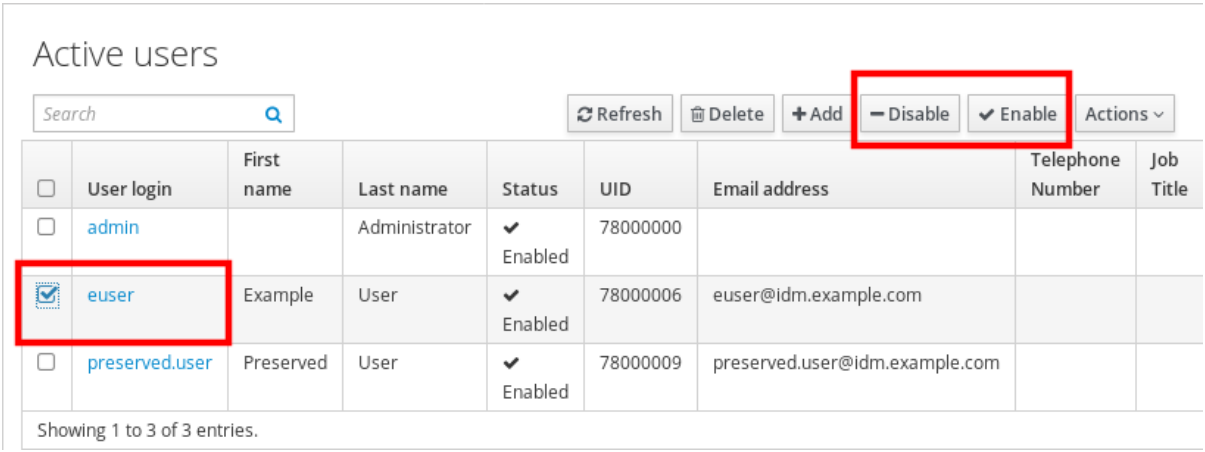
ユーザーアカウントを無効にした後、既存の接続はユーザーの Kerberos TGT や他のチケットの有効期限が切れるまで有効です。チケットの期限が切れると、ユーザーが更新できなくなります。

前提条件

- IdM Web UI、またはユーザー管理者ロールを管理する管理者権限

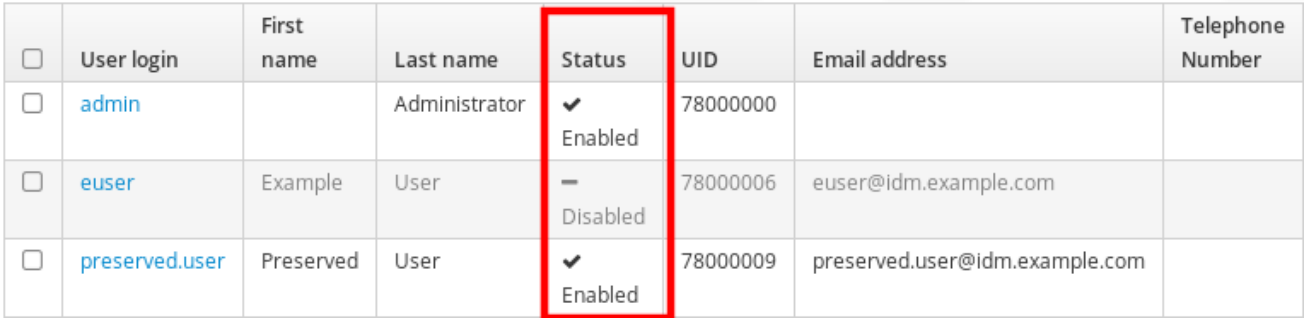
手順

1. IdM Web UI にログインします。
   詳細は Web ブラウザーで IdM Web UI へのアクセス を参照してください。
2. **ユーザー → アクティブユーザー** タブに移動します。
3. 無効にするユーザーアカウントのチェックボックスをクリックします。
4. **無効** ボタンをクリックします。



5. **確認** ダイアログボックスで、**OK** ボタンをクリックします。

無効化の手順に成功した場合は、**アクティブユーザー** テーブルの状態の列で確認できます。



12.5. WEB UI でユーザーアカウントの有効化

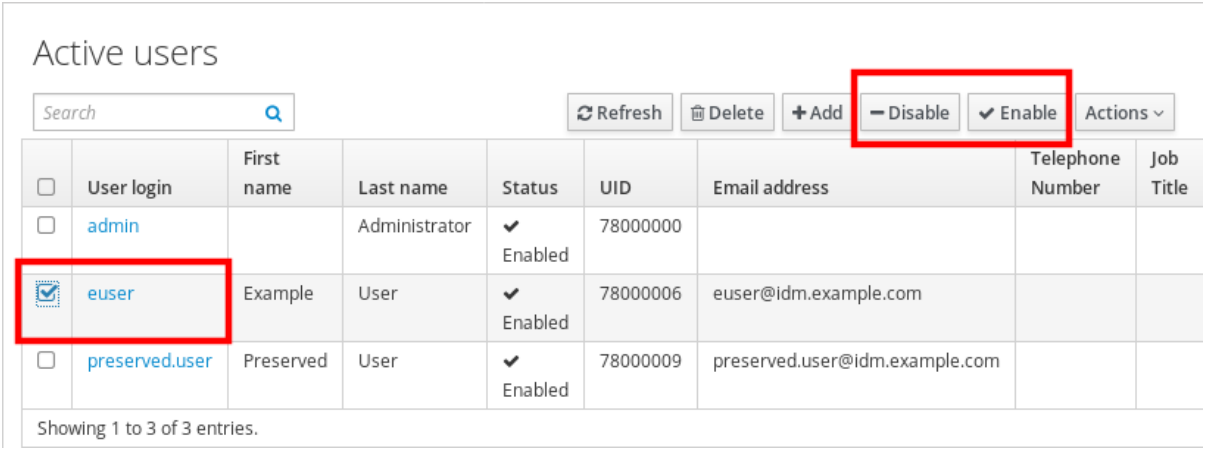
IdM を使用して、無効にしたアクティブなユーザーアカウントを再度有効にできます。ユーザーアカウントを有効にすると、無効になったアカウントが有効になります。

前提条件

- IdM Web UI、またはユーザー管理者ロールを管理する管理者権限

手順

1. IdM Web UI にログインします。
2. **ユーザー → アクティブユーザー** タブに移動します。
3. 有効にするユーザーアカウントのチェックボックスをクリックします。
4. **有効** ボタンをクリックします。



5. **確認** ダイアログボックスで、**OK** ボタンをクリックします。

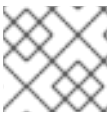
変更に成功すると、**アクティブユーザー** テーブルの状態の列で確認できます。

12.6. IDM WEB UI でアクティブなユーザーの保存

ユーザーアカウントを保存すると、**アクティブユーザー** タブからアカウントを削除した状態で、IdM でアカウントを維持できます。

従業員が退職する場合は、ユーザーアカウントを保存します。ユーザーアカウントを数週間または数か月間 (たとえば育児休暇) 無効にする場合は、ユーザーアカウントを無効にします。詳細は、Web UI でのユーザーアカウントの無効化 を参照してください。保存済みアカウントはアクティブではないため、そのユーザーが内部ネットワークにはアクセスできないものの、すべてのデータが含まれる状態でデータベース内に残ります。

復元したアカウントをアクティブモードに戻すことができます。



注記

保存済みユーザーのリストは、以前のユーザーアカウントの履歴を提供します。

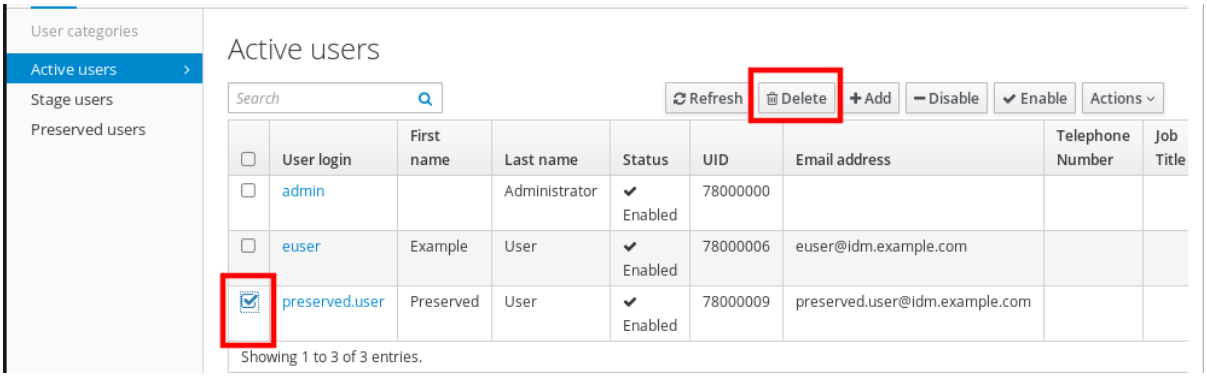
前提条件

- IdM (Identity Management) Web UI、またはユーザー管理者ロールを管理する管理者権限

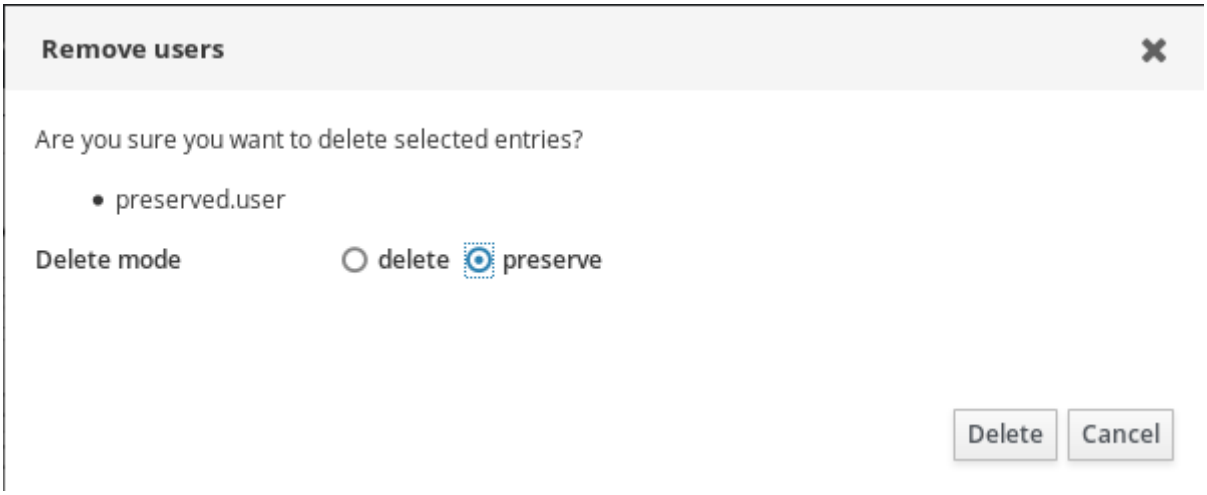
手順

手順

1. IdM Web UI にログインします。
   詳細は Web ブラウザーで IdM Web UI へのアクセス を参照してください。
2. **ユーザー → アクティブユーザー** タブに移動します。
3. 保存するユーザーアカウントのチェックボックスをクリックします。
4. **削除** ボタンをクリックします。



5. **ユーザーの削除** ダイアログボックスで、**削除モード** ラジオボタンを、**保存** に切り替えます。
6. **削除** ボタンをクリックします。



これにより、そのユーザーアカウントは、**保存済みユーザー** に移動します。

保存済みユーザーを復元する必要がある場合は、IdM Web UI でユーザーの復元 を参照してください。

12.7. IDM WEB UI でユーザーの復元

IdM (Identity Management) を使用すると、保存済みユーザーアカウントをアクティブな状態で復元できます。保存済みユーザーをアクティブなユーザーまたはステージユーザーに復元できます。

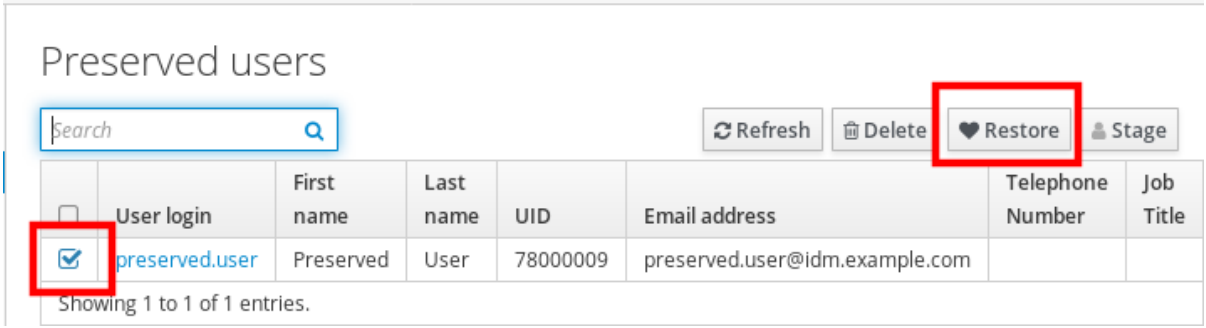
前提条件

- IdM Web UI、またはユーザー管理者ロールを管理する管理者権限

手順

1. IdM Web UI にログインします。
   詳細は Web ブラウザーで IdM Web UI へのアクセス を参照してください。

2. **ユーザー → 保存済みユーザー** タブに移動します。
3. 復元するユーザーアカウントのチェックボックスをクリックします。
4. **復元** ボタンをクリックします。



5. **確認** ダイアログボックスで、**OK** ボタンをクリックします。

IdM Web UI は、緑色の確認を表示し、ユーザーアカウントを **アクティブユーザー** タブに移動します。

12.8. IDM WEB UI でユーザーの削除

ユーザーの削除は元に戻せない操作であり、グループメンバーシップやパスワードなど、ユーザーアカウントが IdM データベースから完全に削除されます。ユーザーの外部設定 (システムアカウントやホームディレクトリーなど) は削除されませんが、IdM からはアクセスできなくなります。

以下を削除できます。

- アクティブなユーザー - IdM Web UI では、以下のオプションを利用できます。
  - ユーザーを一時的に保存する
    詳細は IdM Web UI でアクティブなユーザーの保存 を参照してください。
  - 完全に削除する
- ステージユーザー - ステージユーザーを完全に削除できます。
- 保存済みユーザー - 保存済みユーザーを完全に削除できます。

以下の手順では、アクティブなユーザーの削除を説明します。以下のタブでも同じようにユーザーアカウントを削除できます。

- **ステージユーザー** タブ
- **保存済みユーザー** タブ

前提条件

- IdM Web UI、またはユーザー管理者ロールを管理する管理者権限

手順

1. IdM Web UI にログインします。
   詳細は Web ブラウザーで IdM Web UI へのアクセス を参照してください。
2. **ユーザー → アクティブユーザー** タブに移動します。

ユーザー → ステージユーザー または **ユーザー → 保存済みユーザー** でも、ユーザーアカウントを削除できます。

3. **削除** アイコンをクリックします。
4. **ユーザーの削除** ダイアログボックスで、**モードの削除** ラジオボタンを、**削除** に切り替えます。
5. **削除** ボタンをクリックします。

ユーザーアカウントが、IdM から完全に削除されました。

第13章 ANSIBLE PLAYBOOK を使用したユーザーアカウントの管理

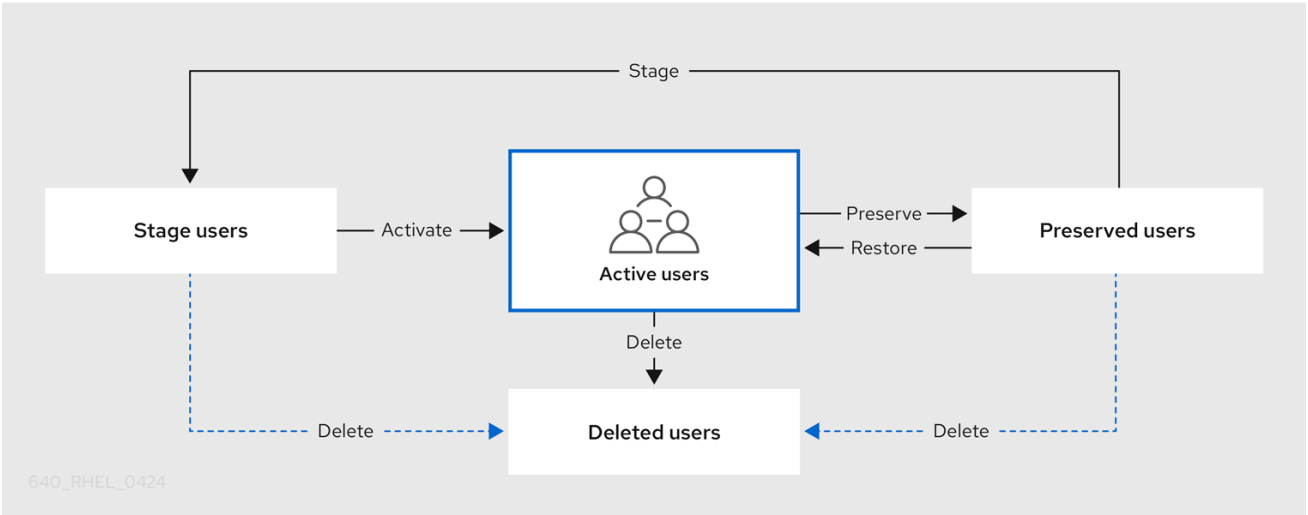
Ansible Playbook を使用して IdM のユーザーを管理できます。ユーザーのライフサイクル を示した後、本章では以下の操作に Ansible Playbook を使用する方法を説明します。

- **YML** ファイルに直接リストされている ユーザーを 1 つ存在させる 手順
- **YML** ファイルに直接リストされている ユーザーを複数存在させる 手順
- **YML** ファイルから参照される **JSON** ファイルにリストされている ユーザーを複数存在させる 手順
- **YML** ファイルに直接リストされている ユーザーがないことを確認 します。

13.1. ユーザーのライフサイクル

Identity Management (IdM) は、次の 3 つのユーザーアカウント状態に対応します

- **ステージ** ユーザーは認証できません。これは初期状態です。アクティブユーザーに必要なユーザーアカウントプロパティーをすべて設定できるわけではありません (例: グループメンバーシップ)。
- **アクティブ** ユーザーは認証が可能です。必要なユーザーアカウントプロパティーはすべて、この状態で設定する必要があります。
- **保存済み** ユーザーは、以前にアクティブであったユーザーで、現在は非アクティブであるとみなされており、IdM への認証ができません。保存済みユーザーには、アクティブユーザーのときに有効になっていたアカウントプロパティーの大部分が保持されていますが、ユーザーグループからは除外されています。



IdM データベースからユーザーエントリーを完全に削除できます。

重要

削除したユーザーアカウントを復元することはできません。ユーザーアカウントを削除すると、そのアカウントに関連する情報がすべて完全に失われます。

新規管理者は、デフォルトの管理ユーザーなど、管理者権限を持つユーザーのみが作成できます。すべての管理者アカウントを誤って削除した場合は、Directory Manager が、Directory Server に新しい管理者を手動で作成する必要があります。

警告

admin ユーザーを削除しないでください。**admin** は IdM で必要な事前定義ユーザーであるため、この操作では特定のコマンドで問題が生じます。別の **admin** ユーザーを定義して使用する場合は、管理者権限を少なくとも 1 つのユーザーに付与してから、**ipa user-disable admin** を使用して、事前定義された admin ユーザーを無効にします。

警告

ローカルユーザーを IdM に追加しないでください。Name Service Switch (NSS) は、ローカルユーザーとグループを解決する前に、IdM ユーザーとグループを常に解決します。つまり、たとえば IdM グループのメンバーシップは、ローカルユーザーでは機能しません。

13.2. ANSIBLE PLAYBOOK を使用して IDM ユーザーを存在させる手順

以下の手順では、Ansible Playbook を使用して IdM にユーザーを 1 つ存在させる方法を説明します。

前提条件

- IdM **admin** のパスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **inventory.file** などのインベントリーファイルを作成して、そのファイルに **ipaserver** を定義します。

```
[ipaserver]
server.idm.example.com
```

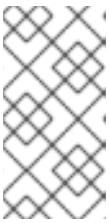
2. IdM に存在させるユーザーのデータを指定して Ansible Playbook ファイルを作成します。この手順は、**/usr/share/doc/ansible-freeipa/playbooks/user/add-user.yml** ファイルのサンプルをコピーして変更し、簡素化できます。たとえば、**idm_user** という名前のユーザーを作成し、**Password123** をユーザーパスワードとして追加するには、次のコマンドを実行します。

```
---
- name: Playbook to handle users
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Create user idm_user
    ipauser:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: idm_user
      first: Alice
      last: Acme
      uid: 1000111
      gid: 10011
      phone: "+555123457"
      email: idm_user@acme.com
      passwordexpiration: "2023-01-19 23:59:59"
      password: "Password123"
      update_password: on_create
```

ユーザーを追加するには、以下のオプションを使用する必要があります。

- **名前**: ログイン名
- **first**: 名前 (名) の文字列
- **last**: 名前 (姓) の文字列

利用可能なユーザーオプションの完全なリストは、**/usr/share/doc/ansible-freeipa/README-user.md** Markdown ファイルを参照してください。



注記

update_password: on_create オプションを使用する場合には、Ansible はユーザー作成時にのみユーザーパスワードを作成します。パスワードを指定してユーザーが作成されている場合には、Ansible では新しいパスワードは生成されません。

3. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file path_to_playbooks_directory/add-IdM-
user.yml
```

検証手順

- **ipa user-show** コマンドを使用して、新しいユーザーアカウントが IdM に存在するかどうかを確認できます。

  1. admin として **ipaserver** にログインします。

     ```
     $ ssh admin@server.idm.example.com
     Password:
     [admin@server /]$
     ```

  2. admin の Kerberos チケットを要求します。

     ```
     $ kinit admin
     Password for admin@IDM.EXAMPLE.COM:
     ```

  3. **idm_user** に関する情報を要求します。

     ```
     $ ipa user-show idm_user
       User login: idm_user
       First name: Alice
       Last name: Acme
       ....
     ```

  idm_userという名前のユーザー が IdM に存在しています。

13.3. ANSIBLE PLAYBOOK を使用して IDM ユーザーを複数存在させる手順

以下の手順では、Ansible Playbook を使用して IdM にユーザーを複数存在させる方法を説明します。

前提条件

- IdM **admin** のパスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

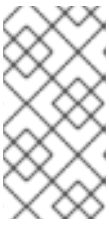
手順

1. **inventory.file** などのインベントリーファイルを作成して、そのファイルに **ipaserver** を定義します。

```
[ipaserver]
server.idm.example.com
```

2. IdM に存在させるユーザーのデータを指定して Ansible Playbook ファイルを作成します。この手順は、**/usr/share/doc/ansible-freeipa/playbooks/user/ensure-users-present.yml** ファイルのサンプルをコピーして変更し、簡素化できます。たとえば、ユーザー **idm_user_1**、**idm_user_2**、**idm_user_3** を作成し、**idm_user_1** のパスワードを **Password123** として追加します。

```
---
- name: Playbook to handle users
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Create user idm_users
    ipauser:
      ipaadmin_password: "{{ ipaadmin_password }}"
      users:
      - name: idm_user_1
        first: Alice
        last: Acme
        uid: 10001
        gid: 10011
        phone: "+555123457"
        email: idm_user@acme.com
        passwordexpiration: "2023-01-19 23:59:59"
        password: "Password123"
      - name: idm_user_2
        first: Bob
        last: Acme
        uid: 100011
        gid: 10011
      - name: idm_user_3
        first: Eve
        last: Acme
        uid: 1000111
        gid: 10011
```



注記

update_password: on_create オプションを指定しないと、Ansible は Playbook が実行されるたびにユーザーパスワードを再設定します。最後に Playbook が実行されてからユーザーがパスワードを変更した場合には、Ansible はパスワードを再設定します。

3. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file path_to_playbooks_directory/add-
users.yml
```

検証手順

- **ipa user-show** コマンドを使用して、ユーザーアカウントが IdM に存在するかどうかを確認できます。

  1. 管理者として **ipaserver** にログインします。

     ```
     $ ssh administrator@server.idm.example.com
     Password:
     [admin@server /]$
     ```

  2. **idm_user_1** に関する情報を表示します。

     ```
     $ ipa user-show idm_user_1
       User login: idm_user_1
       First name: Alice
       Last name: Acme
       Password: True
       ....
     ```

  idm_user_1 という名前のユーザーが IdM に存在しています。

13.4. ANSIBLE PLAYBOOK を使用して JSON ファイルに指定してある複数の IDM ユーザーを存在させる手順

以下の手順では、Ansible Playbook を使用して IdM に複数のユーザーを存在させる方法を説明します。ユーザーは **JSON** ファイルに保存されます。

前提条件

- IdM **admin** のパスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **inventory.file** などのインベントリーファイルを作成して、そのファイルに **ipaserver** を定義します。

   ```
   [ipaserver]
   server.idm.example.com
   ```

2. 必要なタスクが含まれる Ansible Playbook ファイルを作成します。存在させるユーザーのデータが指定された **JSON** ファイルを参照します。この手順は、**/usr/share/doc/ansible-freeipa/ensure-users-present-ymlfile.yml** ファイルのサンプルをコピーして変更し、簡素化できます。

```
---
- name: Ensure users' presence
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Include users.json
    include_vars:
      file: users.json

  - name: Users present
    ipauser:
      ipaadmin_password: "{{ ipaadmin_password }}"
      users: "{{ users }}"
```

3. **users.json** ファイルを作成し、IdM ユーザーを追加します。この手順を簡素化するには、**/usr/share/doc/ansible-freeipa/playbooks/user/users.json** ファイルのサンプルをコピーして変更できます。たとえば、ユーザー **idm_user_1**、**idm_user_2**、**idm_user_3** を作成し、**idm_user_1** のパスワードを **Password123** として追加します。

```
{
  "users": [
    {
      "name": "idm_user_1",
      "first": "Alice",
      "last": "Acme",
      "password": "Password123"
    },
    {
      "name": "idm_user_2",
      "first": "Bob",
      "last": "Acme"
    },
    {
      "name": "idm_user_3",
      "first": "Eve",
      "last": "Acme"
    }
  ]
}
```

4. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file path_to_playbooks_directory/ensure-users-
present-jsonfile.yml
```

検証手順

- **ipa user-show** コマンドを使用して、ユーザーアカウントが IdM に存在するかどうかを確認できます。

  1. 管理者として **ipaserver** にログインします。

     ```
     $ ssh administrator@server.idm.example.com
     Password:
     [admin@server /]$
     ```

  2. **idm_user_1** に関する情報を表示します。

     ```
     $ ipa user-show idm_user_1
       User login: idm_user_1
       First name: Alice
       Last name: Acme
       Password: True
       ....
     ```

  idm_user_1 という名前のユーザーが IdM に存在しています。

13.5. ANSIBLE PLAYBOOK を使用してユーザーが存在しないことを確認する手順

以下の手順では、Ansible Playbook を使用して、特定のユーザーが IdM に存在しないことを確認する方法を説明します。

前提条件

- IdM **admin** のパスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **inventory.file** などのインベントリーファイルを作成して、そのファイルに **ipaserver** を定義します。

   ```
   [ipaserver]
   server.idm.example.com
   ```

2. IdM に存在させないユーザーを指定して Ansible Playbook ファイルを作成します。この手順は、**/usr/share/doc/ansible-freeipa/playbooks/user/ensure-users-present.yml** ファイルのサンプルをコピーして変更し、簡素化できます。たとえば、ユーザー **idm_user_1**、**idm_user_2**、**idm_user_3** を削除するには、次のコマンドを実行します。

```
---
- name: Playbook to handle users
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Delete users idm_user_1, idm_user_2, idm_user_3
    ipauser:
      ipaadmin_password: "{{ ipaadmin_password }}"
      users:
      - name: idm_user_1
      - name: idm_user_2
      - name: idm_user_3
      state: absent
```

3. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file path_to_playbooks_directory/delete-
users.yml
```

検証手順

ipa user-show コマンドを使用して、ユーザーアカウントが IdM に存在しないことを確認できます。

1. 管理者として **ipaserver** にログインします。

```
$ ssh administrator@server.idm.example.com
Password:
[admin@server /]$
```

2. **idm_user_1** に関する要求情報:

```
$ ipa user-show idm_user_1
ipa: ERROR: idm_user_1: user not found
```

idm_user_1 という名前のユーザーは IdM に存在しません。

13.6. 関連情報

- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-user.md** Markdown ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/user** ディレクトリーのサンプルの Ansible Playbook を参照してください。

第14章 IDM CLI でのユーザーグループの管理

本章では、IdM CLI を使用したユーザーグループ管理について説明します。

ユーザーグループは、共通の特権、パスワードポリシーなどの特性が指定された一連のユーザーです。

Identity Management (IdM) のユーザーグループには以下が含まれます。

- IdM ユーザー
- 他の IdM ユーザーグループ
- 外部ユーザー (IdM の外部に存在するユーザー)

14.1. IDM のさまざまなグループタイプ

IdM は、以下のグループタイプをサポートします。

POSIX グループ (デフォルト)

POSIX グループは、メンバーの Linux POSIX 属性に対応します。Active Directory と対話するグループは POSIX 属性を使用できないことに注意してください。
POSIX 属性は、ユーザーを個別のエンティティーとして識別します。ユーザーに関連する POSIX 属性の例には、**uidNumber** (ユーザー番号 (UID))、および **gidNumber** (グループ番号 (GID)) が含まれます。

非 POSIX グループ

非 POSIX グループは、POSIX 属性に対応していません。たとえば、これらのグループには GID が定義されていません。
このタイプのグループのすべてのメンバーは、IdM ドメインに属している必要があります。

外部グループ

外部グループを使用して、以下のような IdM ドメイン外の ID ストアに存在するグループメンバーを追加できます。

- ローカルシステム
- Active Directory ドメイン
- ディレクトリーサービス

外部グループは、POSIX 属性に対応していません。たとえば、これらのグループには GID が定義されていません。

表14.1 デフォルトで作成されたユーザーグループ

グループ名	デフォルトのグループメンバー
ipausers	すべての IdM ユーザー
admins	管理権限を持つユーザー (デフォルトの **admin** ユーザーを含む)
editors	特権のないレガシーグループ

グループ名	デフォルトのグループメンバー
trust admins	Active Directory 信頼を管理する権限を持つユーザー

ユーザーをユーザーグループに追加すると、ユーザーはグループに関連付けられた権限とポリシーを取得します。たとえば、ユーザーに管理権限を付与するには、ユーザーを **admins** グループに追加します。

警告

admins グループを削除しないでください。**admins** は IdM で必要な事前定義グループであるため、この操作では特定のコマンドで問題が生じます。

さらに、IdM で新しいユーザーが作成されるたびに、IdM は、デフォルトで **ユーザーのプライベートグループ** を作成します。プライベートグループの詳細は、プライベートグループのないユーザーの追加 を参照してください。

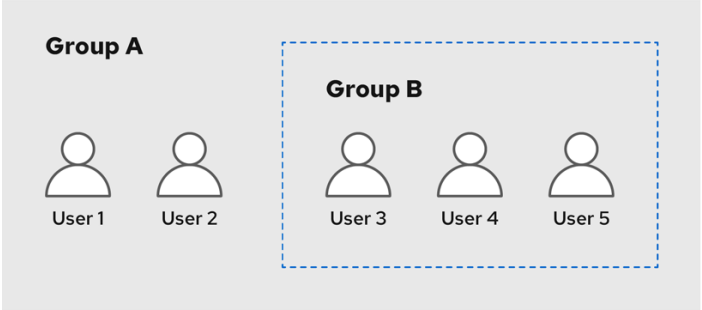
14.2. 直接および間接のグループメンバー

IdM のユーザーグループ属性は、直接メンバーと間接メンバーの両方に適用されます。グループ B がグループ A のメンバーである場合、グループ B のすべてのユーザーはグループ A の間接メンバーと見なされます。

たとえば、以下の図の場合:

- ユーザー 1 とユーザー 2 は、グループ A の **直接メンバー** です。
- ユーザー 3、ユーザー 4、およびユーザー 5 は、グループ A の **間接メンバー** です。

図14.1 直接および間接グループメンバーシップ



640_RHEL_0424

ユーザーグループ A にパスワードポリシーを設定すると、そのポリシーはユーザーグループ B のすべてのユーザーにも適用されます。

14.3. IDM CLI を使用したユーザーグループの追加

IdM CLI を使用してユーザーグループを追加するには、次の手順に従います。

前提条件

- 管理者としてログインしている。詳細は、Using kinit to log in to IdM manually を参照してください。

手順

- **ipa group-add group_name** コマンドを使用してユーザーグループを追加します。たとえば、group_a を作成するには、次のコマンドを実行します。

```
$ ipa group-add group_a
---------------------
Added group "group_a"
---------------------
  Group name: group_a
  GID: 1133400009
```

  デフォルトでは、**ipa group-add** は、POSIX ユーザーグループを追加します。別のグループタイプを指定するには、**ipa group-add** にオプションを追加します。

  - **--nonposix** は、非 POSIX グループを作成します。
  - **--external** は、外部グループを作成します。
    グループタイプの詳細は、IdM のさまざまなグループタイプ を参照してください。

  ユーザーグループの追加時にカスタムの GID を指定するには、**--gid=custom_GID** オプションを使用します。これを行う場合は、ID の競合を避けるように注意してください。カスタム GID を指定しない場合、IdM は使用可能な ID 範囲から GID を自動的に割り当てます。

14.4. IDM CLI を使用したユーザーグループの検索

IdM CLI を使用して既存のユーザーグループを検索するには、次の手順に従います。

手順

- **ipa group-find** コマンドを使用して、すべてのユーザーグループを表示します。グループタイプを指定するには、**ipa group-find** にオプションを追加します。
  - **ipa group-find --posix** コマンドを使用して、すべての POSIX グループを表示します。
  - **ipa group-find --nonposix** コマンドを使用して、すべての非 POSIX グループを表示します。
  - **ipa group-find --external** コマンドを使用して、すべての外部グループを表示します。
    異なるグループタイプの詳細は、IdM のさまざまなグループタイプ を参照してください。

14.5. IDM CLI を使用したユーザーグループの削除

IdM CLI を使用してユーザーグループを削除するには、には、次の手順に従います。グループを削除しても、IdM からグループメンバーは削除されないことに注意してください。

前提条件

- 管理者としてログインしている。詳細は、Using kinit to log in to IdM manually を参照してください。

手順

- **ipa group-del group_name** コマンドを使用してユーザーグループを削除します。たとえば、group_a を削除するには、次のコマンドを実行します。

```
$ ipa group-del group_a
--------------------------
Deleted group "group_a"
--------------------------
```

14.6. IDM CLI でユーザーグループにメンバーの追加

ユーザーおよびユーザーグループの両方をユーザーグループのメンバーとして追加できます。詳細は、IdM のさまざまなグループタイプ および 直接および間接のグループメンバー を参照してください。IdM CLI を使用してユーザーグループにメンバーを追加するには、次の手順に従います。

前提条件

- 管理者としてログインしている。詳細は、Using kinit to log in to IdM manually を参照してください。

手順

- **ipa group-add-member** コマンドを使用して、ユーザーグループにメンバーを追加します。次のオプションを使用して、メンバーのタイプを指定します。
  - **--users** は、IdM ユーザーを追加します
  - **--external** は、**DOMAIN\user_name** 形式または **user_name@domain** 形式で、IdM ドメイン外に存在するユーザーを追加します
  - **--groups** は、IdM ユーザーグループを追加します。

  たとえば、group_b を group_a のメンバーとして追加するには、次のコマンドを実行します。

```
$ ipa group-add-member group_a --groups=group_b
Group name: group_a
GID: 1133400009
Member users: user_a
Member groups: group_b
Indirect Member users: user_b
-------------------------
Number of members added 1
-------------------------
```

  group_b のメンバーは、group_a の間接メンバーになりました。

重要

グループを別のグループのメンバーとして追加する場合は、再帰グループを作成しないでください。たとえば、グループ A がグループ B のメンバーである場合は、グループ B をグループ A のメンバーとして追加しないでください。再帰的なグループにより予期しない動作が発生する可能性があります。

注記

ユーザーグループにメンバーを追加した後、更新が Identity Management 環境のすべてのクライアントに広がるまでに時間がかかる場合があります。これは、特定のホストがユーザー、グループ、およびネットグループを解決するときに、**System Security Services Daemon** (SSSD) が最初にキャッシュを調べて、サーバーで不足または期限切れのレコードのみを検索するためです。

14.7. ユーザープライベートグループなしでユーザーの追加

デフォルトでは、IdM で新しいユーザーが作成されるたびに、IdM がユーザーのプライベートグループ (UPG) を作成します。UPG は特定のグループタイプです。

- UPG の名前は、新しく作成されたユーザーと同じです。
- ユーザーは UPG の唯一のメンバーです。UPG には他のメンバーを含めることができません。
- プライベートグループの GID は、ユーザーの UID と一致します。

ただし、UPG を作成せずにユーザーを追加することは可能です。

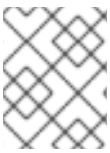
14.7.1. ユーザープライベートグループのないユーザー

NIS グループまたは別のシステムグループが、ユーザープライベートグループに割り当てられる GID をすでに使用している場合は、UPG を作成しないようにする必要があります。

これは、以下の 2 つの方法で実行できます。

- プライベートグループをグローバルに無効にすることなく、UPG なしで新しいユーザーを追加する。プライベートグループがグローバルに有効になっている場合にユーザープライベートグループのないユーザーを追加 を参照してください。
- すべてのユーザーに対して UPG をグローバルに無効にしてから、新しいユーザーを追加する。すべてのユーザーに対してユーザープライベートグループをグローバルで無効にする および ユーザープライベートグループをグローバルで無効にしている時のユーザーの追加 を参照してください。

どちらの場合も、IdM では、新しいユーザーを追加するときに GID を指定する必要があります。指定しないと、操作は失敗します。これは、IdM には新しいユーザーの GID が必要ですが、デフォルトのユーザーグループ **ipausers** は非 POSIX グループであるため、関連付けられた GID がないためです。指定する GID は、既存のグループに対応する必要がありません。



注記

GID を指定しても、新しいグループは作成されません。この属性は IdM に必要であるため、新しいユーザーに GID 属性のみを設定します。

14.7.2. プライベートグループがグローバルに有効になっている場合にユーザープライベートグループのないユーザーを追加

システムで UPG が有効になっている場合でも、ユーザープライベートグループ (UPG) を作成せずにユーザーを追加できます。これには、新しいユーザーの GID を手動で設定する必要があります。これが必要な理由の詳細については、ユーザープライベートグループのないユーザー を参照してください。

手順

- IdM が UPG を作成しないようにするには、**ipa user-add** コマンドに **--noprivate** オプションを追加します。
  コマンドを成功させるには、カスタム GID を指定する必要があることに注意してください。たとえば、GID 10000 の新しいユーザーを追加するには、次のコマンドを実行します。

```
$ ipa user-add jsmith --first=John --last=Smith --noprivate --gid 10000
```

14.7.3. すべてのユーザーに対してユーザープライベートグループをグローバルに無効にする

ユーザープライベートグループ (UPG) をグローバルに無効にできます。これにより、すべての新しいユーザーの UPG が作成されなくなります。既存のユーザーはこの変更の影響を受けません。

手順

1. 管理者権限を取得します。

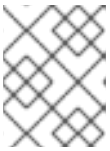
```
$ kinit admin
```

2. IdM は、Directory Server Managed Entries プラグインを使用して UPG を管理します。プラグインのインスタンスをリスト表示します。

```
$ ipa-managed-entries --list
```

3. IdM が UPG を作成しないようにするには、ユーザープライベートグループを管理するプラグインインスタンスを無効にします。

```
$ ipa-managed-entries -e "UPG Definition" disable
Disabling Plugin
```



注記

後で **UPG 定義** インスタンスを再有効にするには、**ipa-managed-entries -e "UPG Definition" enable** コマンドを使用します。

4. Directory Server を再起動して、新しい設定を読み込みます。

```
$ sudo systemctl restart dirsrv.target
```

UPG が無効になった後にユーザーを追加するには、GID を指定する必要があります。詳細は、ユーザープライベートグループがグローバルで無効になっている場合のユーザーの追加 を参照してください。

検証手順

- UPG がグローバルで無効になっているかどうかを確認するには、再度 disable コマンドを使用します。

```
$ ipa-managed-entries -e "UPG Definition" disable
Plugin already disabled
```

14.7.4. ユーザープライベートグループがグローバルで無効になっている場合のユーザーの追加

ユーザープライベートグループ (UPG) がグローバルで無効になっている場合、IdM は GID を新しいユーザーに自動的に割り当てません。ユーザーを正常に追加するには、GID を手動で割り当てるか、automember を使用して割り当てる必要があります。これが必要な理由の詳細については、Users without a user private group を参照してください。

前提条件

- UPG は、すべてのユーザーに対してグローバルに無効にする必要があります。詳細は、すべてのユーザーに対してユーザープライベートグループをグローバルに無効にする をご覧ください。

手順

- UPG の作成が無効になっているときに新しいユーザーの追加が成功することを確認するには、次のいずれかを選択します。
  - 新しいユーザーを追加するときにカスタムの GID を指定します。GID は、既存のユーザーグループに対応する必要はありません。
    たとえば、コマンドラインからユーザーを追加する場合は、**--gid** オプションを **ipa user-add** コマンドに追加します。
  - automember ルールを使用して、GID のある既存のグループにユーザーを追加します。

14.8. IDM CLI を使用して IDM ユーザーグループにメンバーマネージャーとしてユーザーまたはグループを追加する手順

IdM CLI を使用してユーザーまたはグループをメンバーマネージャーとして IdM ユーザーグループに追加するには、次の手順に従います。メンバーマネージャーは、ユーザーまたはグループを IdM ユーザーグループに追加できますが、グループの属性を変更できません。

前提条件

- 管理者としてログインしている。詳細は、Using kinit to log in to IdM manually を参照してください。
- メンバーマネージャーとして追加するユーザーまたはグループの名前と、メンバーマネージャーが管理するグループ名を用意する。

手順

- **ipa group-add-member-manager** コマンドを使用して、ユーザーをメンバーマネージャーとして IdM ユーザーグループに追加します。
  たとえば、ユーザー **test** を **group_a** のメンバーマネージャーとして追加します。

```
$ ipa group-add-member-manager group_a --users=test
Group name: group_a
GID: 1133400009
Membership managed by users: test
-------------------------
Number of members added 1
-------------------------
```

ユーザー **test** は **group_a** のメンバーを管理できるようになりました。

- **ipa group-add-member-manager** コマンドを使用して、グループをメンバーマネージャーとして IdM ユーザーグループに追加します。
たとえば、 グループ **group_admins** を **group_a** のメンバーマネージャーとして追加します。

```
$ ipa group-add-member-manager group_a --groups=group_admins
Group name: group_a
GID: 1133400009
Membership managed by groups: group_admins
Membership managed by users: test
-------------------------
Number of members added 1
-------------------------
```

グループ **group_admins** は **group_a** のメンバーを管理できるようになりました。

注記

ユーザーグループにメンバーマネージャーを追加してから、更新が Identity Management 環境のすべてのクライアントに広がるまでに時間がかかる場合があります。

検証手順

- **ipa group-show** コマンドを使用して、ユーザーおよびグループがメンバーマネージャーとして追加されたことを確認します。

```
$ ipa group-show group_a
Group name: group_a
GID: 1133400009
Membership managed by groups: group_admins
Membership managed by users: test
```

関連情報

- 詳細は、**ipa group-add-member-manager --help** を参照してください。

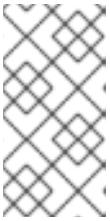
14.9. IDM CLI を使用したグループメンバーの表示

IdM CLI を使用してグループのメンバーを表示するには、次の手順に従います。直接グループメンバーと間接グループメンバーの両方を表示できます。詳細は、直接および間接のグループメンバー を参照してください。

手順

- グループのメンバーのリストを表示するには、**ipa group-show group_name** コマンドを使用します。以下に例を示します。

```
$ ipa group-show group_a
  ...
  Member users: user_a
  Member groups: group_b
  Indirect Member users: user_b
```



注記

間接メンバーのリストには、信頼された Active Directory ドメインの外部ユーザーが含まれません。Active Directory 信頼ユーザーオブジェクトは、Identity Management 内に LDAP オブジェクトとして存在しないため、Identity Management インターフェイスには表示されません。

14.10. IDM CLI を使用してユーザーグループからメンバーを削除

IdM CLI を使用してユーザーグループからメンバーを削除するには、次の手順に従います。

前提条件

- 管理者としてログインしている。詳細は、Using kinit to log in to IdM manually を参照してください。

手順

1. **オプション。ipa group-show** コマンドを使用して、削除するメンバーがグループに含まれていることを確認します。
2. **ipa group-remove-member** コマンドを使用して、ユーザーグループからメンバーを削除します。
   以下のオプションを使用して、削除するメンバーを指定します。
   - **--users** は、IdM ユーザーを削除します
   - **--external** は、**DOMAIN\user_name** または **user_name@domain** の形式で、IdM ドメイン外に存在するユーザーを削除します
   - **--groups** は、IdM ユーザーグループを削除します

   たとえば、**group_name** という名前のグループから、**user1**、**user2**、および **group1** を削除するには、次のコマンドを実行します。

```
$ ipa group-remove-member group_name --users=user1 --users=user2 --groups=group1
```

14.11. IDM CLI を使用してユーザーまたはグループを IDM ユーザーグループのメンバーマネージャーから取り消す手順

IdM CLI を使用して IdM ユーザーグループのメンバーマネージャーからユーザーまたはグループを削除するには、次の手順に従います。メンバーマネージャーは、IdM ユーザーグループからユーザーまたはグループを削除できますが、グループの属性を変更できません。

前提条件

- 管理者としてログインしている。詳細は、Using kinit to log in to IdM manually を参照してください。
- 削除する既存のメンバーマネージャーのユーザーまたはグループの名前と、そのメンバーマネージャーが管理するグループ名が必要です。

手順

- **ipa group-remove-member-manager** コマンドを使用してユーザーを IdM ユーザーグループのメンバーマネージャーから削除します。
  たとえば、ユーザー **test** を **group_a** のメンバーマネージャーから削除します。

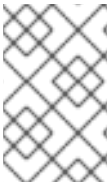
```
$ ipa group-remove-member-manager group_a --users=test
Group name: group_a
GID: 1133400009
Membership managed by groups: group_admins
-------------------------
Number of members removed 1
-------------------------
```

  ユーザー **test** は **group_a** のメンバーを管理できなくなりました。

- **ipa group-remove-member-manager** コマンドを使用してグループを IdM ユーザーグループのメンバーマネージャーから削除します。
  たとえば、 グループ **group_admins** を **group_a** のメンバーマネージャーから削除します。

```
$ ipa group-remove-member-manager group_a --groups=group_admins
Group name: group_a
GID: 1133400009
-------------------------
Number of members removed 1
-------------------------
```

  グループ **group_admins** は **group_a** のメンバーを管理できなくなりました。



注記

ユーザーグループからメンバーマネージャーを削除してから、更新が Identity Management 環境のすべてのクライアントに広がるまでに時間がかかる場合があります。

検証手順

- **ipa group-show** コマンドを使用して、ユーザーおよびグループがメンバーマネージャーから削除されたことを確認します。

```
$ ipa group-show group_a
Group name: group_a
GID: 1133400009
```

関連情報

- 詳細は、**ipa group-remove-member-manager --help** を参照してください。

14.12. IDM でのローカルグループとリモートグループのグループマージの有効化

グループは、Identity Management (IdM) や Active Directory (AD) などのドメインによって提供されて一元管理されるか、ローカルシステムの **etc/group** ファイルで管理されます。ほとんどの場合、ユーザーは一元管理されたストアに依存しています。しかし、ソフトウェアによっては、現在も既知のグループのメンバーシップに基づいてアクセス制御を管理している場合があります。

ドメインコントローラーおよびローカルの **etc/group** ファイルのグループを管理する場合は、グループのマージを有効にすることができます。ローカルファイルとリモートサービスの両方を確認するように **nsswitch.conf** ファイルを設定できます。グループが両方に存在する場合、メンバーユーザーのリストが結合され、単一の応答で返されます。

以下の手順では、ユーザー **idmuser** のグループのマージを有効にする方法について説明します。

手順

1. **/etc/nsswitch.conf** ファイルに **[SUCCESS=merge]** を追加します。

```
# Allow initgroups to default to the setting for group.
initgroups: sss [SUCCESS=merge] files
```

2. **idmuser** を IdM に追加します。

```
# ipa user-add idmuser
First name: idm
Last name: user
----------------------
Added user "idmuser"
----------------------
User login: idmuser
First name: idm
Last name: user
Full name: idm user
Display name: idm user
Initials: tu
Home directory: /home/idmuser
GECOS: idm user
Login shell: /bin/sh
Principal name: idmuser@IPA.TEST
Principal alias: idmuser@IPA.TEST
Email address: idmuser@ipa.test
UID: 19000024
GID: 19000024
Password: False
Member of groups: ipausers
Kerberos keys available: False
```

3. ローカルの **audio** グループの GID を確認します。

```
$ getent group audio
--------------------
audio:x:63
```

4. **audio** グループを IdM に追加します。

```
$ ipa group-add audio --gid 63
-------------------
Added group "audio"
-------------------
Group name: audio
GID: 63
```

注記

audio グループを IdM に追加するときに定義する GID は、ローカルの **audio** グループの GID と同じである必要があります。

5. **idmuser** ユーザーを IdM の **audio** グループに追加します。

```
$ ipa group-add-member audio --users=idmuser
Group name: audio
GID: 63
Member users: idmuser
-------------------------
Number of members added 1
-------------------------
```

検証

1. **idmuser** としてログインします。
2. **idmuser** のセッションにローカルグループがあることを確認します。

```
$ id idmuser
uid=1867800003(idmuser) gid=1867800003(idmuser)
groups=1867800003(idmuser),63(audio),10(wheel)
```

14.13. ANSIBLE を使用して、IDM クライアント上のローカルサウンドカードへのユーザー ID オーバーライドアクセスを付与する

ansible-freeipa グループ および **idoverrideuser** モジュールを使用して、アイデンティティー Management (IdM) または Active Directory (AD) ユーザーを IdM クライアント上のローカル **オーディオ** グループのメンバーにすることができます。これにより、IdM または AD ユーザーにホスト上のサウンドカードへの特権アクセスが付与されます。この手順では、最初の Playbook タスクで **aduser@addomain.com** ID オーバーライドが追加された **デフォルトの信頼ビュー** ID ビューの例を使用します。次の Playbook タスクでは、RHEL ホスト上のローカル **オーディオ** グループの GID に対応する GID 63 の **オーディオ** グループが IdM に作成されます。同時に、 aduser@addomain.com ID オーバーライドが IdM オーディオグループにメンバーとして追加されます。

前提条件

- 手順の最初の部分を実行する IdM クライアントへの **ルート** アクセス権を持っていること。この例では、**client.idm.example.com** です。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - RHEL 8.10 以降を使用しています。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- AD フォレストは IdM と信頼関係にあります。この例では、AD ドメインの名前は **addomain.com** であり、ローカル **オーディオ** グループでの存在が保証されている AD ユーザーの完全修飾ドメイン名 (FQDN) は **aduser@addomain.com** です。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **client.idm.example.com** で、**/etc/nsswitch.conf** ファイルに **[SUCCESS=merge]** を追加します。

```
[...]
# Allow initgroups to default to the setting for group.
initgroups: sss [SUCCESS=merge] files
```

2. ローカルの **audio** グループの GID を確認します。

```
$ getent group audio
---------------------
audio:x:63
```

3. Ansible コントロールノードで、**aduser@addomain.com** ユーザーオーバーライドをデフォルトの信頼ビューに追加するタスクを含む **add-aduser-to-audio-group.yml** Playbook を作成します。

```
---
- name: Playbook to manage idoverrideuser
  hosts: ipaserver
  become: false

  tasks:
  - name: Add aduser@addomain.com user to the Default Trust View
    ipaidoverrideuser:
      ipaadmin_password: "{{ ipaadmin_password }}"
      idview: "Default Trust View"
      anchor: aduser@addomain.com
```

4. 同じ Playbook 内の別の Playbook タスクを使用して、グループ **オーディオをGID** 63 で IdM に追加します。aduser idoverrideuser をグループに追加します。

```
- name: Add the audio group with the aduser member and GID of 63
  ipagroup:
    ipaadmin_password: "{{ ipaadmin_password }}"
    name: audio
    idoverrideuser:
    - aduser@addomain.com
    gidnumber: 63
```

5. ファイルを保存します。

6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory add-aduser-to-audio-group.yml
```

検証

1. AD ユーザーとして IdM クライアントにログインします。

```
$ ssh aduser@addomain.com@client.idm.example.com
```

2. AD ユーザーのグループメンバーシップを確認します。

```
$ id aduser@addomain.com
uid=702801456(aduser@addomain.com) gid=63(audio) groups=63(audio)
```

関連情報

- idoverrideuser および ipagroup **ansible-freeipa** アップストリームドキュメント
- IdM でのローカルグループとリモートグループのグループマージの有効化

第15章 IDM WEG UI でのユーザーグループの管理

本章では、IdM Web UI を使用したユーザーグループ管理を説明します。

ユーザーグループは、共通の特権、パスワードポリシーなどの特性が指定された一連のユーザーです。

Identity Management (IdM) のユーザーグループには以下が含まれます。

- IdM ユーザー
- 他の IdM ユーザーグループ
- 外部ユーザー (IdM の外部に存在するユーザー)

15.1. IDM のさまざまなグループタイプ

IdM は、以下のグループタイプをサポートします。

POSIX グループ (デフォルト)

POSIX グループは、メンバーの Linux POSIX 属性に対応します。Active Directory と対話するグループは POSIX 属性を使用できないことに注意してください。
POSIX 属性は、ユーザーを個別のエンティティーとして識別します。ユーザーに関連する POSIX 属性の例には、**uidNumber** (ユーザー番号 (UID))、および **gidNumber** (グループ番号 (GID)) が含まれます。

非 POSIX グループ

非 POSIX グループは、POSIX 属性に対応していません。たとえば、これらのグループには GID が定義されていません。
このタイプのグループのすべてのメンバーは、IdM ドメインに属している必要があります。

外部グループ

外部グループを使用して、以下のような IdM ドメイン外の ID ストアに存在するグループメンバーを追加できます。

- ローカルシステム
- Active Directory ドメイン
- ディレクトリーサービス

外部グループは、POSIX 属性に対応していません。たとえば、これらのグループには GID が定義されていません。

表15.1 デフォルトで作成されたユーザーグループ

グループ名	デフォルトのグループメンバー
ipausers	すべての IdM ユーザー
admins	管理権限を持つユーザー (デフォルトの **admin** ユーザーを含む)
editors	特権のないレガシーグループ

グループ名	デフォルトのグループメンバー
trust admins	Active Directory 信頼を管理する権限を持つユーザー

ユーザーをユーザーグループに追加すると、ユーザーはグループに関連付けられた権限とポリシーを取得します。たとえば、ユーザーに管理権限を付与するには、ユーザーを **admins** グループに追加します。

警告

admins グループを削除しないでください。**admins** は IdM で必要な事前定義グループであるため、この操作では特定のコマンドで問題が生じます。

さらに、IdM で新しいユーザーが作成されるたびに、IdM は、デフォルトで **ユーザーのプライベートグループ** を作成します。プライベートグループの詳細は、プライベートグループのないユーザーの追加 を参照してください。

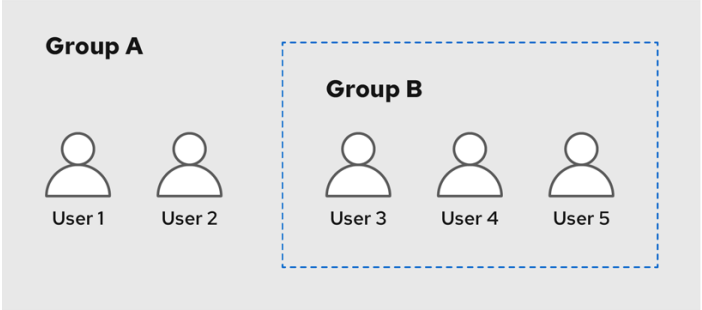
15.2. 直接および間接のグループメンバー

IdM のユーザーグループ属性は、直接メンバーと間接メンバーの両方に適用されます。グループ B がグループ A のメンバーである場合、グループ B のすべてのユーザーはグループ A の間接メンバーと見なされます。

たとえば、以下の図の場合:

- ユーザー 1 とユーザー 2 は、グループ A の **直接メンバー** です。
- ユーザー 3、ユーザー 4、およびユーザー 5 は、グループ A の **間接メンバー** です。

図15.1 直接および間接グループメンバーシップ



640_RHEL_0424

ユーザーグループ A にパスワードポリシーを設定すると、そのポリシーはユーザーグループ B のすべてのユーザーにも適用されます。

15.3. IDM WEB UI を使用したユーザーグループの追加

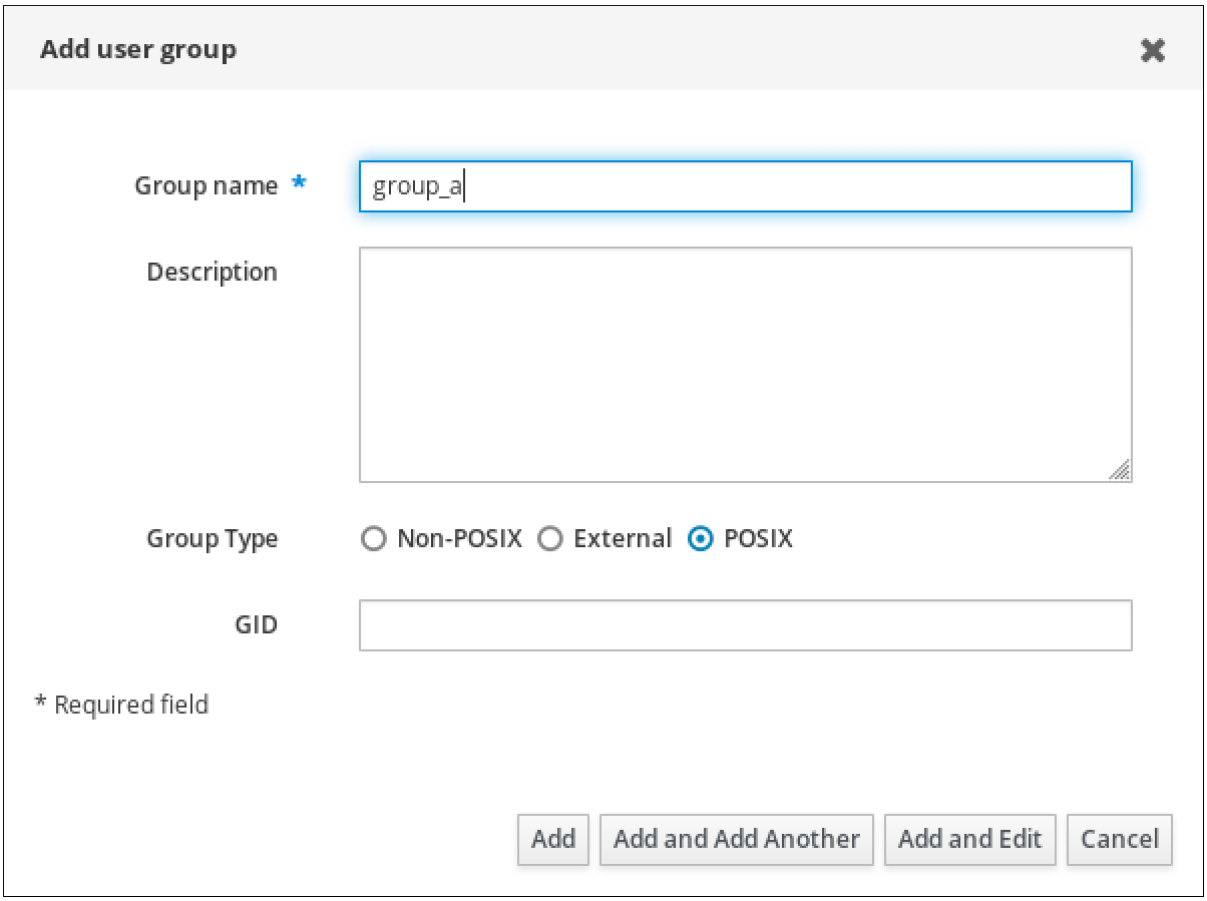
IdM Web UI を使用してユーザーグループを追加するには、次の手順に従います。

前提条件

- IdM Web UI にログインしている。

手順

1. **Identity → Groups** の順にクリックし、左のサイドバーで **User Groups** を選択します。
2. **Add** をクリックして、グループの追加を開始します。
3. グループの情報を入力します。ユーザーグループタイプの詳細は、IdM のさまざまなグループタイプ を参照してください。
   グループのカスタム GID を指定できます。これを行う場合は、ID の競合を避けるように注意してください。カスタム GID を指定しない場合、IdM は使用可能な ID 範囲から GID を自動的に割り当てます。



4. **Add** をクリックして確定します。

15.4. IDM WEB UI を使用したユーザーグループの削除

IdM Web UI を使用してユーザーグループを削除するには、次の手順に従います。グループを削除しても、IdM からグループメンバーは削除されないことに注意してください。

前提条件

- IdM Web UI にログインしている。

手順

1. **Identity → Groups** の順にクリックし、**User Groups** を選択します。
2. 削除するグループを選択します。
3. **Delete** をクリックします。
4. **Delete** をクリックして確定します。

15.5. IDM WEB UI でユーザーグループにメンバーの追加

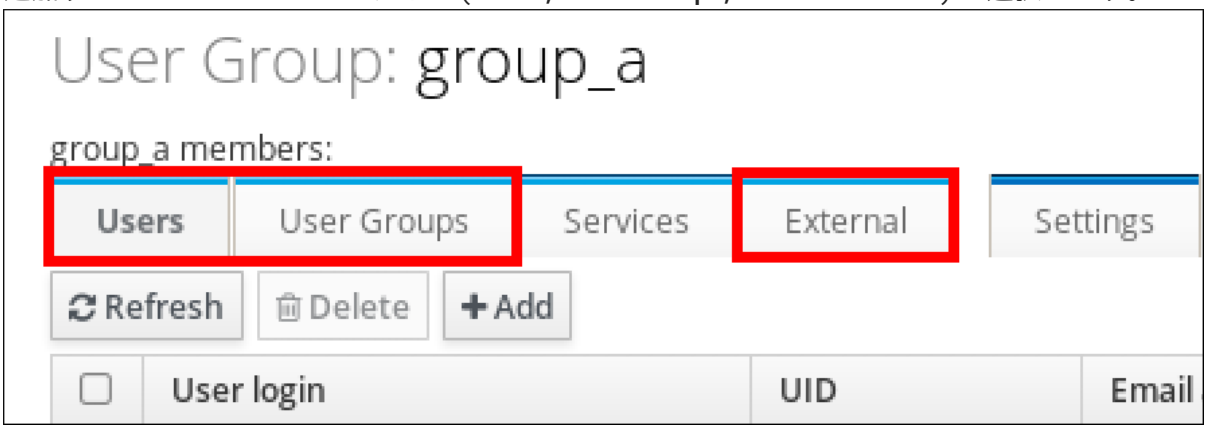
ユーザーおよびユーザーグループの両方をユーザーグループのメンバーとして追加できます。詳細は、IdM のさまざまなグループタイプ および 直接および間接のグループメンバー を参照してください。

前提条件

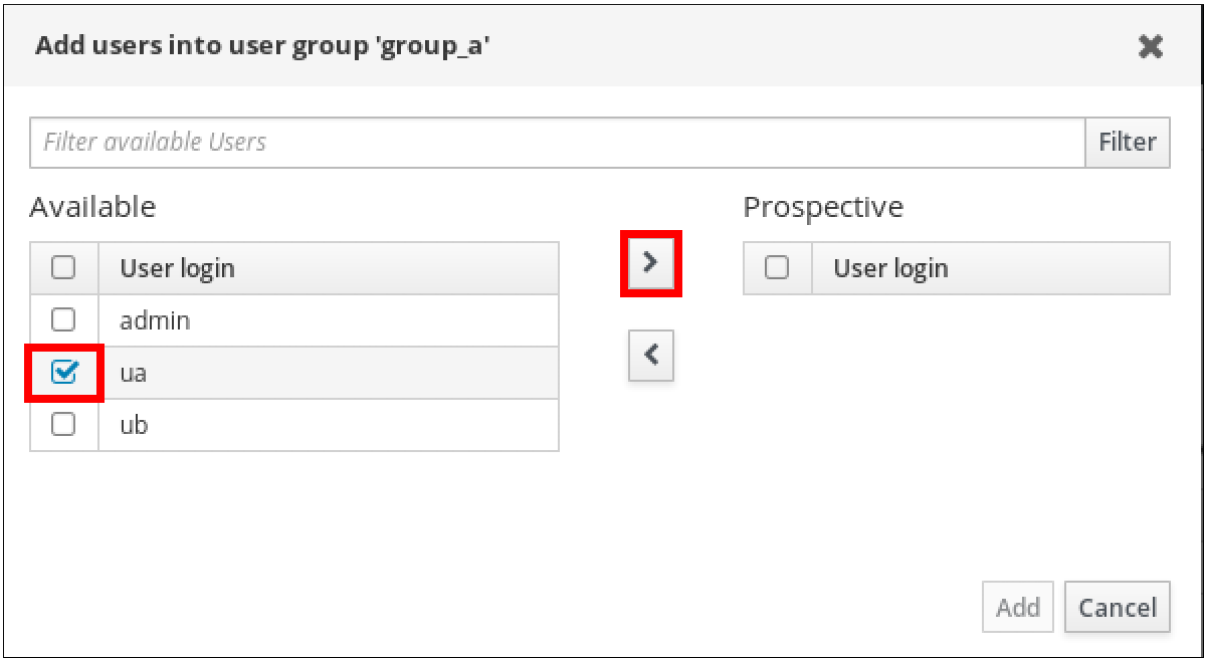
- IdM Web UI にログインしている。

手順

1. **Identity → Groups** の順にクリックし、左のサイドバーで **User Groups** を選択します。
2. グループ名をクリックします。
3. 追加するグループメンバーのタイプ (**Users**, **User Groups**, または **External**) を選択します。



4. **Add** をクリックします。
5. 追加するメンバーの横にあるチェックボックスを 1 つ以上選択します。
6. 右矢印をクリックして、選択したメンバーをグループに移動します。



7. **Add** をクリックして確定します。

15.6. WEB UI を使用して IDM ユーザーグループにメンバーマネージャーとしてユーザーまたはグループを追加する手順

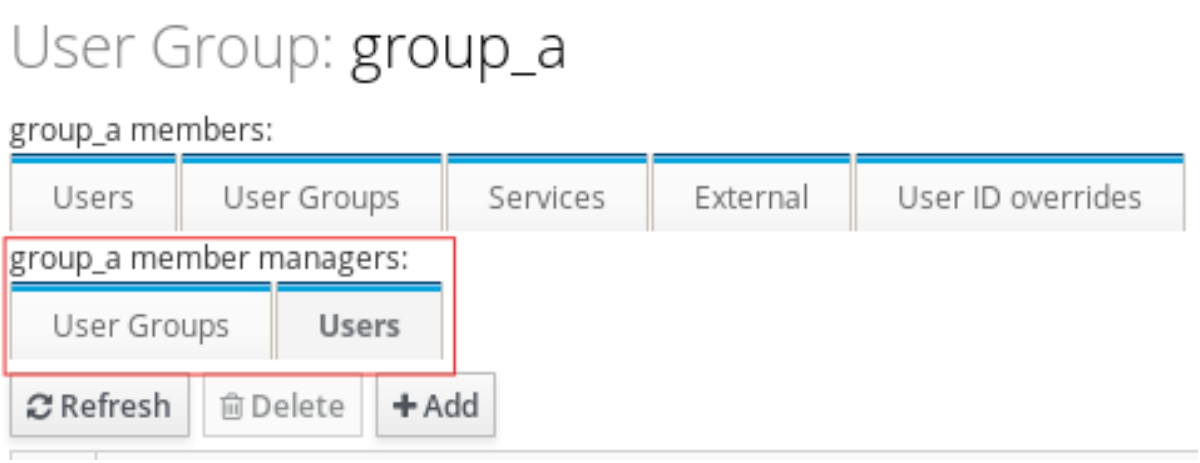
Web UI を使用してユーザーまたはグループをメンバーマネージャーとして IdM ユーザーグループに追加するには、次の手順に従います。メンバーマネージャーは、ユーザーまたはグループを IdM ユーザーグループに追加できますが、グループの属性を変更できません。

前提条件

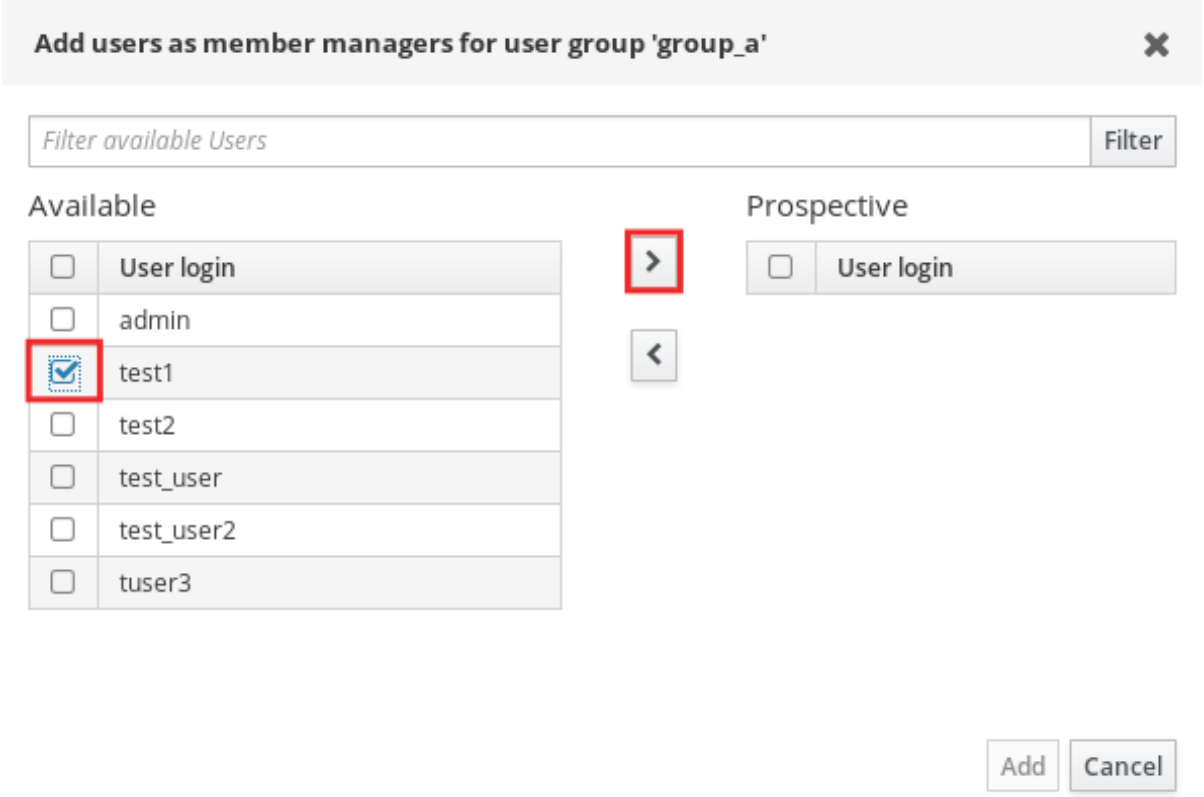
- IdM Web UI にログインしている。
- メンバーマネージャーとして追加するユーザーまたはグループの名前と、メンバーマネージャーが管理するグループ名を用意する。

手順

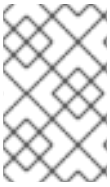
1. **Identity → Groups** の順にクリックし、左のサイドバーで **User Groups** を選択します。
2. グループ名をクリックします。
3. 追加するグループメンバーマネージャーのタイプ (**Users** または **User Groups**) を選択します。



4. **Add** をクリックします。
5. 追加するメンバーの横にあるチェックボックスを 1 つ以上選択します。
6. 右矢印をクリックして、選択したメンバーをグループに移動します。



7. **Add** をクリックして確定します。

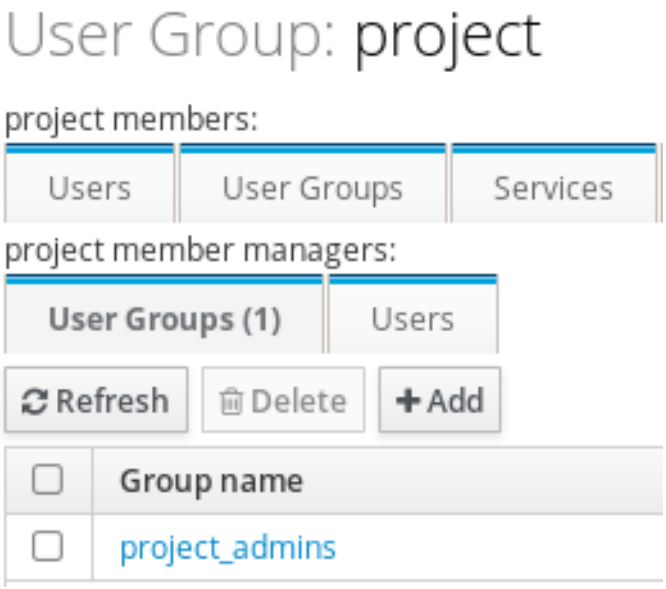


注記

ユーザーグループにメンバーマネージャーを追加してから、更新が Identity Management 環境のすべてのクライアントに広がるまでに時間がかかる場合があります。

検証手順

- 新たに追加したユーザーまたはユーザーグループが、ユーザーまたはユーザーグループのメンバーマネージャーのリストに追加されていることを確認します。



関連情報

- 詳細は、**ipa group-add-member-manager --help** を参照してください。

15.7. IDM WEB UI を使用したグループメンバーの表示

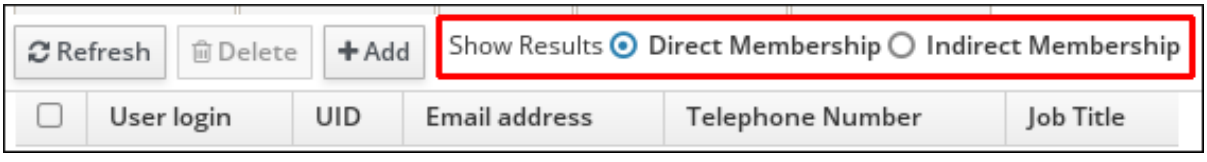
IdM Web UI を使用してグループのメンバーを表示するには、次の手順に従います。直接グループメンバーと間接グループメンバーの両方を表示できます。詳細は、直接および間接のグループメンバー を参照してください。

前提条件

- IdM Web UI にログインしている。

手順

1. **Identity → Groups** を選択します。
2. 左のサイドバーで **User Groups** を選択します。
3. 表示するグループの名前をクリックします。
4. **直接メンバーシップ** と **間接メンバーシップ** を切り替えます。



15.8. IDM WEB UI を使用してユーザーグループからメンバーの削除

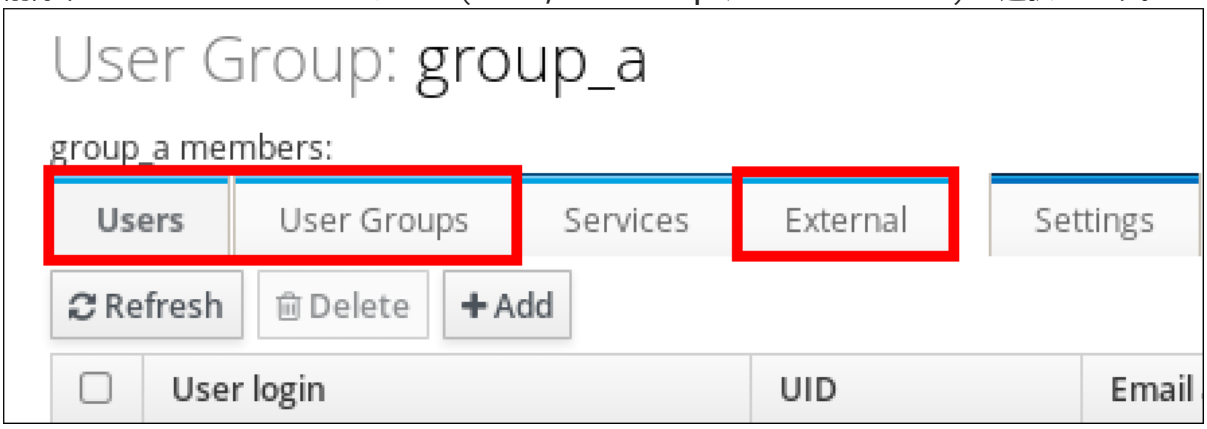
IdM Web UI を使用してユーザーグループからメンバーを削除するには、次の手順に従います。

前提条件

- IdM Web UI にログインしている。

手順

1. **Identity → Groups** の順にクリックし、左のサイドバーで **User Groups** を選択します。
2. グループ名をクリックします。
3. 削除するグループメンバーのタイプ (**Users, User Groups**、または **External**) を選択します。



4. 削除するメンバーの横にあるチェックボックスを選択します。
5. **Delete** をクリックします。
6. **Delete** をクリックして確定します。

15.9. WEB UI を使用してユーザーまたはグループを IDM ユーザーグループのメンバーマネージャーから取り消す手順

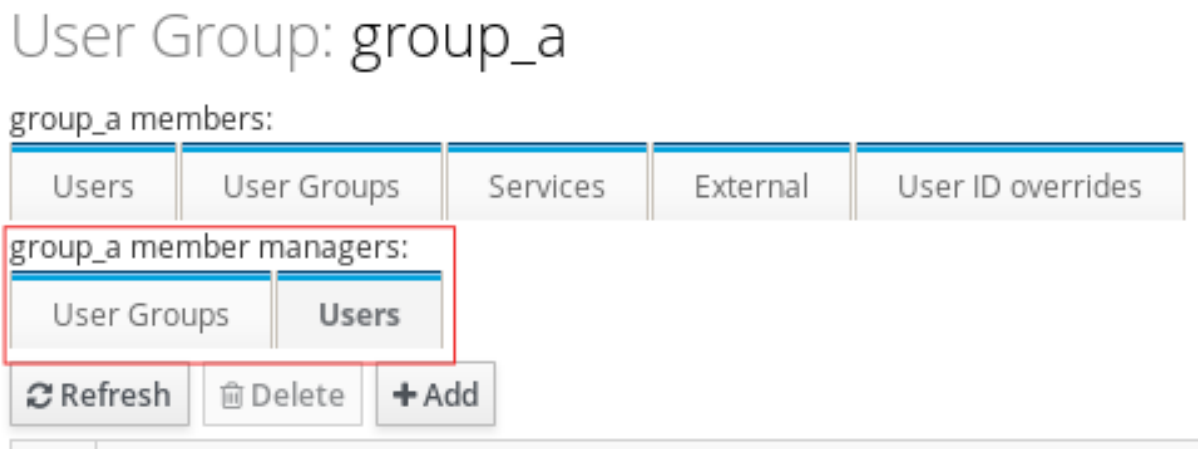
Web UI を使用して IdM ユーザーグループのメンバーマネージャーからユーザーまたはグループを削除するには、次の手順に従います。メンバーマネージャーは、IdM ユーザーグループからユーザーまたはグループを削除できますが、グループの属性を変更できません。

前提条件

- IdM Web UI にログインしている。
- 削除する既存のメンバーマネージャーのユーザーまたはグループの名前と、そのメンバーマネージャーが管理するグループ名が必要です。

手順

1. **Identity → Groups** の順にクリックし、左のサイドバーで **User Groups** を選択します。
2. グループ名をクリックします。
3. 削除するメンバーマネージャーのタイプ (**Users** または **User Groups**) を選択します。



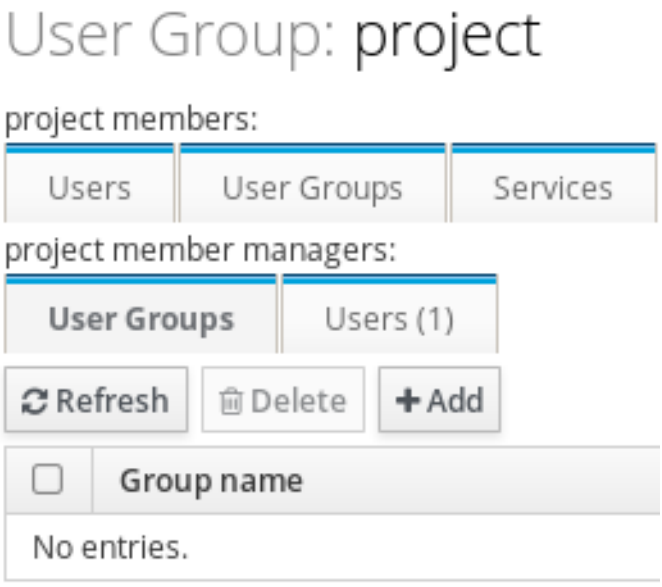
4. 削除するメンバーマネージャーの横にあるチェックボックスを選択します。
5. **Delete** をクリックします。
6. **Delete** をクリックして確定します。

注記

ユーザーグループからメンバーマネージャーを削除してから、更新が Identity Management 環境のすべてのクライアントに広がるまでに時間がかかる場合があります。

検証手順

- ユーザーまたはユーザーグループが、ユーザーまたはユーザーグループのメンバーマネージャーリストから削除されていることを確認します。



関連情報

- 詳細は、**ipa group-add-member-manager --help** を参照してください。

第16章 ANSIBLE PLAYBOOK を使用したユーザーグループの管理

本セクションでは、Ansible Playbook を使用したユーザーグループの管理について紹介します。

ユーザーグループは、共通の特権、パスワードポリシーなどの特性が指定された一連のユーザーです。

Identity Management (IdM) のユーザーグループには以下が含まれます。

- IdM ユーザー
- 他の IdM ユーザーグループ
- 外部ユーザー (IdM の外部に存在するユーザー)

このセクションには、以下のトピックが含まれます。

- IdM のさまざまなグループタイプ
- 直接および間接のグループメンバー
- Ansible Playbook を使用して IdM グループおよびグループメンバーの存在を確認する
- Ansible を使用して AD ユーザーが IdM を管理できるようにする手順
- Ansible Playbook を使用して IDM ユーザーグループにメンバーマネージャーを存在させる手順
- Ansible Playbook を使用して IDM ユーザーグループにメンバーマネージャーを存在させないようにする手順

16.1. IDM のさまざまなグループタイプ

IdM は、以下のグループタイプをサポートします。

POSIX グループ (デフォルト)

POSIX グループは、メンバーの Linux POSIX 属性に対応します。Active Directory と対話するグループは POSIX 属性を使用できないことに注意してください。
POSIX 属性は、ユーザーを個別のエンティティーとして識別します。ユーザーに関連する POSIX 属性の例には、**uidNumber** (ユーザー番号 (UID))、および **gidNumber** (グループ番号 (GID)) が含まれます。

非 POSIX グループ

非 POSIX グループは、POSIX 属性に対応していません。たとえば、これらのグループには GID が定義されていません。
このタイプのグループのすべてのメンバーは、IdM ドメインに属している必要があります。

外部グループ

外部グループを使用して、以下のような IdM ドメイン外の ID ストアに存在するグループメンバーを追加できます。

- ローカルシステム
- Active Directory ドメイン
- ディレクトリーサービス

外部グループは、POSIX 属性に対応していません。たとえば、これらのグループには GID が定義されていません。

表16.1 デフォルトで作成されたユーザーグループ

グループ名	デフォルトのグループメンバー
ipausers	すべての IdM ユーザー
admins	管理権限を持つユーザー (デフォルトの **admin** ユーザーを含む)
editors	特権のないレガシーグループ
trust admins	Active Directory 信頼を管理する権限を持つユーザー

ユーザーをユーザーグループに追加すると、ユーザーはグループに関連付けられた権限とポリシーを取得します。たとえば、ユーザーに管理権限を付与するには、ユーザーを **admins** グループに追加します。

警告

admins グループを削除しないでください。**admins** は IdM で必要な事前定義グループであるため、この操作では特定のコマンドで問題が生じます。

さらに、IdM で新しいユーザーが作成されるたびに、IdM は、デフォルトで **ユーザーのプライベートグループ** を作成します。プライベートグループの詳細は、プライベートグループのないユーザーの追加 を参照してください。

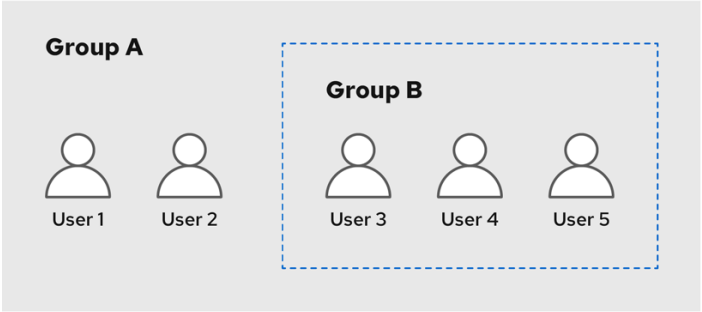
16.2. 直接および間接のグループメンバー

IdM のユーザーグループ属性は、直接メンバーと間接メンバーの両方に適用されます。グループ B がグループ A のメンバーである場合、グループ B のすべてのユーザーはグループ A の間接メンバーと見なされます。

たとえば、以下の図の場合:

- ユーザー 1 とユーザー 2 は、グループ A の **直接メンバー** です。
- ユーザー 3、ユーザー 4、およびユーザー 5 は、グループ A の **間接メンバー** です。

図16.1 直接および間接グループメンバーシップ



ユーザーグループ A にパスワードポリシーを設定すると、そのポリシーはユーザーグループ B のすべてのユーザーにも適用されます。

16.3. ANSIBLE PLAYBOOK を使用して IDM グループおよびグループメンバーの存在を確認する

以下の手順では、Ansible Playbook を使用して、IdM グループとグループメンバー (ユーザーとユーザーグループの両方) を存在させる方法を説明します。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- Ansible Playbook で参照するユーザーが IdM に存在する。Ansible を使用してユーザーの存在を確認する方法は、Ansible Playbook を使用したユーザーアカウントの管理 を参照してください。

手順

1. **inventory.file** などのインベントリーファイルを作成して、そのファイルに **ipaserver** を定義します。

```
[ipaserver]
server.idm.example.com
```

2. 必要なユーザーおよびグループ情報を使用して Ansible Playbook ファイルを作成します。

```
---
- name: Playbook to handle groups
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Create group ops with gid 1234
    ipagroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: ops
      gidnumber: 1234

  - name: Create group sysops
    ipagroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: sysops
      user:
      - idm_user

  - name: Create group appops
    ipagroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: appops

  - name: Add group members sysops and appops to group ops
    ipagroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: ops
      group:
      - sysops
      - appops
```

3. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file path_to_playbooks_directory/add-group-
members.yml
```

検証手順

ipa group-show コマンドを使用すると、ops グループに sysops および appops がダイレクトメンバーとして追加され、**idm_user** が間接メンバーとして追加されているかどうかを確認できます。

1. 管理者として **ipaserver** にログインします。

```
$ ssh admin@server.idm.example.com
Password:
[admin@server /]$
```

2. ops についての情報を表示します。

```
ipaserver]$ ipa group-show ops
  Group name: ops
  GID: 1234
```

```
Member groups: sysops, appops
Indirect Member users: idm_user
```

appops および **sysops** グループ (後者には **idm_user** ユーザーを含む) が IdM に存在します。

関連情報

- **/usr/share/doc/ansible-freeipa/README-group.md** Markdown ファイルを参照してください。

16.4. ANSIBLE を使用して単一のタスクで複数の IDM グループを追加する

ansible-freeipa ipagroup モジュールを使用すると、単一の Ansible タスクで複数の Identity Management (IdM) ユーザーグループを追加、変更、削除できます。そのためには、**ipagroup** モジュールの **groups** オプションを使用します。

groups オプションを使用すると、特定のグループにのみ適用される複数のグループ変数を指定することもできます。このグループは **name** 変数で定義します。これは、**groups** オプションの唯一の必須変数です。

この手順を単一のタスクで完了して、IdM に **sysops** グループと **appops** グループが存在することを確認します。sysops グループは非 posix グループとして定義し、appops グループは外部グループとして定義します。

前提条件

- コントロールノードでは、
  - Ansible バージョン 2.14 以降を使用している。
  - **ansible-freeipa** パッケージをインストールしている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成した。
  - RHEL 8.9 以降を使用している。
  - **ipaadmin_password** を **Secret.yml** Ansible Vault に保存している。

手順

1. 次の内容を含む Ansible Playbook ファイル **add-nonposix-and-external-groups.yml** を作成します。

```
---
- name: Playbook to add nonposix and external groups
  hosts: ipaserver
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml

  tasks:
  - name: Add nonposix group sysops and external group appops
    ipagroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      groups:
```

```
    - name: sysops
      nonposix: true
    - name: appops
      external: true
```

2. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
<path_to_inventory_directory>/hosts <path_to_playbooks_directory>/add-nonposix-
and-external-groups.yml
```

関連情報

- **ansible-freeipa** アップストリームドキュメントのグループモジュール

16.5. ANSIBLE を使用して AD ユーザーが IDM を管理できるようにする手順

Ansible Playbook を使用してユーザー ID オーバーライドが Identity Management (IdM) グループに存在することを確認するには、次の手順に従います。ユーザー ID オーバーライドは、Active Directory (AD) とのトラストを確立した後に、デフォルトの Trust View で作成した AD ユーザーのオーバーライドです。Playbook を実行すると、AD 管理者などの AD ユーザーは、2 つの異なるアカウントとパスワードを持たなくても IdM を完全に管理できるようになります。

前提条件

- IdM **admin** のパスワードを把握している。
- AD とのトラストをインストール している。
- IdM に AD ユーザーのユーザー ID オーバーライドがすでに存在する。存在しない場合は、**ipa idoverrideuser-add 'default trust view' ad_user@ad.example.com** コマンドで作成します。
- ユーザー ID オーバーライドを追加しようとしているグループが、IdM にすでに存在する 。
- IdM 4.8.7 バージョン以降の IdM を使用している。サーバーにインストールされている IdM のバージョンを表示するには、**ipa --version** を実行します。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **~/MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. 次の内容で **add-useridoverride-to-group.yml** playbook を作成します。

```
---
- name: Playbook to ensure presence of users in a group
  hosts: ipaserver


  - name: Ensure the ad_user@ad.example.com user ID override is a member of the admins
group:
    ipagroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: admins
      idoverrideuser:
      - ad_user@ad.example.com
```

上記の例では、以下のようになります。

- Secret123 は IdM **admin** パスワードです。
- **admins** は、**ad_user@ad.example.com** ID オーバーライドを追加する IdM POSIX グループの名前です。このグループのメンバーには、完全な管理者権限があります。
- **ad_user@ad.example.com** は、AD 管理者のユーザー ID オーバーライドです。ユーザーは、信頼が確立された AD ドメインに保存されます。

3. ファイルを保存します。

4. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory add-
useridoverride-to-group.yml
```

関連情報

- AD ユーザーの ID のオーバーライド
- /usr/share/doc/ansible-freeipa/README-group.md
- /usr/share/doc/ansible-freeipa/playbooks/user
- Using ID views in Active Directory environments
- IdM を管理する AD ユーザーの有効化

16.6. ANSIBLE PLAYBOOK を使用して IDM ユーザーグループにメンバーマネージャーを存在させる手順

以下の手順では、Ansible Playbook を使用して、ユーザーとユーザーグループ両方の IdM メンバーマネージャーが存在させる方法を説明します。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- メンバーマネージャーとして追加するユーザーまたはグループの名前と、メンバーマネージャーが管理するグループ名を用意する。

手順

1. **inventory.file** などのインベントリーファイルを作成して、そのファイルに **ipaserver** を定義します。

```
[ipaserver]
server.idm.example.com
```

2. 必要なユーザーおよびグループメンバー管理情報を使用して、Ansible Playbook ファイルを作成します。

```
---
- name: Playbook to handle membership management
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure user test is present for group_a
    ipagroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: group_a
      membermanager_user: test

  - name: Ensure group_admins is present for group_a
    ipagroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: group_a
      membermanager_group: group_admins
```

3. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file path_to_playbooks_directory/add-member-
managers-user-groups.yml
```

検証手順

ipa group-show コマンドを使用すると、**group_a** グループにメンバーマネージャーの **test** が含まれており、**group_admins** が **group_a** のメンバーであることが確認できます。

1. 管理者として **ipaserver** にログインします。

```
$ ssh admin@server.idm.example.com
Password:
[admin@server /]$
```

2. **managergroup1** についての情報を表示します。

```
ipaserver]$ ipa group-show group_a
  Group name: group_a
  GID: 1133400009
  Membership managed by groups: group_admins
  Membership managed by users: test
```

関連情報

- **ipa host-add-member-manager --help** を参照してください。
- **ipa** の man ページを参照してください。

16.7. ANSIBLE PLAYBOOK を使用して IDM ユーザーグループにメンバーマネージャーを存在させないようにする手順

以下の手順では、Ansible Playbook を使用して、ユーザーとユーザーグループ両方の IdM メンバーマネージャーが存在させないようにする方法を説明します。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

- 削除する既存のメンバーマネージャーのユーザーまたはグループの名前と、そのメンバーマネージャーが管理するグループ名が必要です。

手順

1. **inventory.file** などのインベントリーファイルを作成して、そのファイルに **ipaserver** を定義します。

```
[ipaserver]
server.idm.example.com
```

2. 必要なユーザーおよびグループメンバー管理情報を使用して、Ansible Playbook ファイルを作成します。

```
---
- name: Playbook to handle membership management
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure member manager user and group members are absent for group_a
    ipagroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: group_a
      membermanager_user: test
      membermanager_group: group_admins
      action: member
      state: absent
```

3. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file path_to_playbooks_directory/ensure-
member-managers-are-absent.yml
```

検証手順

ipa group-show コマンドを使用すると、**group_a** グループにメンバーマネージャーの **test** が含まれておらず、**group_admins** が **group_a** のメンバーであることが確認できます。

1. 管理者として **ipaserver** にログインします。

```
$ ssh admin@server.idm.example.com
Password:
[admin@server /]$
```

2. group_a についての情報を表示します。

```
ipaserver]$ ipa group-show group_a
  Group name: group_a
  GID: 1133400009
```

関連情報

- **ipa host-remove-member-manager --help** を参照してください。
- **ipa** の man ページを参照してください。

第17章 IDM CLI を使用したグループメンバーシップの自動化

自動グループメンバーシップを使用すると、属性に基づいてユーザーとホストをグループに自動割り当てできます。たとえば、以下を行うことができます。

- 従業員のマネージャー、ロケーション、またはその他の属性に基づいて従業員のユーザーエントリーをグループに分類する。
- クラス、ロケーション、またはその他の属性に基づいてホストを分類する。
- 全ユーザーまたは全ホストを 1 つのグローバルグループに追加する。

本章では、以下のトピックについて説明します。

17.1. 自動グループメンバーシップの利点

ユーザーの自動メンバーシップを使用すると、以下が可能になります。

- **グループメンバーシップの手動管理してオーバーヘッドを削減する**
  すべてのユーザーおよびグループを手作業で割り当てる必要がなくなります。
- **ユーザーおよびホスト管理における一貫性を向上する**
  ユーザーとホストは、厳密に定義された基準かつ自動評価された基準をもとにグループに割り当てられます。
- **グループベースの設定管理を簡素化する**
  さまざまな設定がグループに定義され、**sudo** ルール、自動マウント、またはアクセス制御などの個別のグループメンバーに適用されます。ユーザーとホストをグループに自動的に追加すると、この設定の管理が容易になります。

17.2. AUTOMEMBER ルール

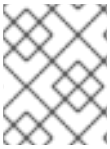
自動グループメンバーシップを設定するには、管理者は automember ルールを定義します。automember ルールは、特定のユーザーまたはホストターゲットグループに適用されます。これは、一度に複数のグループに適用できません。

管理者はルールの作成後に条件を追加します。条件では、ユーザーまたはホストをターゲットグループに含めるか、除外するかを指定します。

- **包含条件**
  ユーザーまたはホストのエントリーが包含条件を満たす場合には、ターゲットグループに含まれます。

- **除外条件**
  ユーザーまたはホストのエントリーが除外条件を満たす場合には、ターゲットグループには含まれません。

この条件は、Perl 互換正規表現 (PCRE) 形式の正規表現として指定します。PCRE の詳細は、**pcresyntax(3)** の man ページを参照してください。



注記

IdM は、包含条件よりも除外条件を先に評価します。競合が発生した場合は、包含条件よりも除外条件が優先されます。

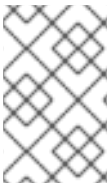
automember ルールは、今後作成されるすべてのエントリーに適用されます。このエントリーは指定されたターゲットグループに自動的に追加されます。エントリーが複数の automember ルールで指定された条件を満たす場合には、該当するグループすべてに追加されます。

既存のエントリーは新規ルールの影響を **受けません**。既存のエントリーを変更する場合は、IdM CLI を使用した既存のエントリーへの automember ルールの適用 を参照してください。

17.3. IDM CLI を使用した AUTOMEMBER ルールの追加

IdM CLI を使用して automember ルールを追加するには、次の手順に従います。automember ルールの詳細は、automember ルール を参照してください。

automember ルールを追加した後に、automember ルールへの条件の追加 で説明されている手順に従って条件を追加できます。



注記

既存のエントリーは新規ルールの影響を **受けません**。既存のエントリーを変更する場合は、IdM CLI を使用した既存のエントリーへの automember ルールの適用 を参照してください。

前提条件

- 管理者としてログインしている。詳細は、Using kinit to log in to IdM manually を参照してください。
- 新しいルールの対象グループは IdM に存在している必要があります。

手順

1. **ipa automember-add** コマンドを入力して、automember ルールを追加します。
2. プロンプトが表示されたら、以下を指定します。
   - **Automember rule**。これはターゲットグループ名です。
   - **Grouping Type**。これは、ルールがユーザーグループまたはホストグループを対象にするかどうかを指定します。ユーザーグループを対象に設定するには、**group** と入力します。ホストグループを対象に設定するには、**hostgroup** と入力します。

たとえば、**user_group** という名前のユーザーグループの automember ルールを追加するには、以下を実行します。

```
$ ipa automember-add
Automember Rule: user_group
Grouping Type: group
--------------------------------
Added automember rule "user_group"
--------------------------------
  Automember Rule: user_group
```

検証手順

- IdM CLI を使用して既存の automember ルールを表示 して、IdM に既存の automember ルールと条件を表示できます。

17.4. IDM CLI を使用した AUTOMEMBER ルールへの条件追加

automember ルールを設定した後、IdM CLI を使用してその automember ルールに条件を追加できます。automember ルールの詳細は、automember ルール を参照してください。

前提条件

- 管理者としてログインしている。詳細は、Using kinit to log in to IdM manually を参照してください。
- ターゲットルールは IdM に存在している必要があります。詳細は IdM CLI を使用した automember ルールの追加 を参照してください。

手順

1. **ipa automember-add-condition** コマンドを使用して、包含または除外の条件を定義します。
2. プロンプトが表示されたら、以下を指定します。
   - **Automember rule**。これはターゲットルール名です。詳細は、Automember ルール を参照してください。
   - **Attribute Key**.これは、フィルターの適用先となるエントリー属性を指定します。たとえば、ユーザーの **uid** です。
   - **Grouping Type**。これは、ルールがユーザーグループまたはホストグループを対象にするかどうかを指定します。ユーザーグループを対象に設定するには、**group** と入力します。ホストグループを対象に設定するには、**hostgroup** と入力します。
   - **Inclusive regex** および **Exclusive regex**。これらは、正規表現として条件を指定します。条件を 1 つだけ指定する場合は、他の条件の入力を求められたら **Enter** を押します。

   たとえば、以下の条件は、ユーザーログイン属性 (**uid**) に値 (.*) が指定されている全ユーザーを対象にしています。

```
$ ipa automember-add-condition
Automember Rule: user_group
Attribute Key: uid
Grouping Type: group
```

```
[Inclusive Regex]: .*
[Exclusive Regex]:
--------------------------------
Added condition(s) to "user_group"
--------------------------------
  Automember Rule: user_group
  Inclusive Regex: uid=.*
----------------------------
Number of conditions added 1
----------------------------
```

別の例として、automembership ルールを使用して、Active Directory (AD) から同期した全 Windows ユーザーを対象にできます。これには、**objectClass** 属性で **ntUser** が指定された全ユーザーを対象にし、全 AD ユーザーに共有される条件を作成します。

```
$ ipa automember-add-condition
Automember Rule: ad_users
Attribute Key: objectclass
Grouping Type: group
[Inclusive Regex]: ntUser
[Exclusive Regex]:
-----------------------------------
Added condition(s) to "ad_users"
-----------------------------------
  Automember Rule: ad_users
  Inclusive Regex: objectclass=ntUser
---------------------------
Number of conditions added 1
---------------------------
```

検証手順

- IdM CLI を使用して既存の automember ルールを表示 して、IdM に既存の automember ルールと条件を表示できます。

17.5. IDM CLI を使用した既存の AUTOMEMBER ルールの表示

IdM CLI を使用して既存の automember ルールを表示するには、次の手順に従います。

前提条件

- 管理者としてログインしている。詳細は、Using kinit to log in to IdM manually を参照してください。

手順

1. **ipa automember-find** コマンドを入力します。
2. プロンプトが表示されたら、**Grouping type** を指定します。
   - ユーザーグループを対象に設定するには、**group** と入力します。
   - ホストグループを対象に設定するには、**hostgroup** と入力します。
     以下に例を示します。

```
$ ipa automember-find
Grouping Type: group
---------------
1 rules matched
---------------
  Automember Rule: user_group
  Inclusive Regex: uid=.*
----------------------------
Number of entries returned 1
----------------------------
```

17.6. IDM CLI を使用した AUTOMEMBER ルールの削除

IdM CLI を使用して automember ルールを削除するには、次の手順に従います。

automember ルールを削除すると、そのルールに関連付けられた条件もすべて削除されます。ルールから特定の条件のみを削除するには、IdM CLI を使用した automember ルールからの条件削除 を参照してください。

前提条件

- 管理者としてログインしている。詳細は、Using kinit to log in to IdM manually を参照してください。

手順

1. **ipa automember-del** コマンドを実行します。
2. プロンプトが表示されたら、以下を指定します。
   - **Automember rule**。これは、削除するルールです。
   - **Grouping rule**。これは、削除するルールがユーザーグループのルールであるか、ホストグループのルールであるかどうかを指定します。**group** または **hostgroup** を入力します。

17.7. IDM CLI を使用した AUTOMEMBER ルールからの条件削除

automember ルールから特定の条件を削除するには、次の手順に従います。

前提条件

- 管理者としてログインしている。詳細は、Using kinit to log in to IdM manually を参照してください。

手順

1. **ipa automember-remove-condition** コマンドを実行します。
2. プロンプトが表示されたら、以下を指定します。
   - **Automember rule**。これは、条件を削除するルールの名前です。
   - **Attribute Key**.これはターゲットエントリー属性です。たとえば、ユーザーの **uid** です。

- **Grouping Type**。これは、ユーザーグループまたはホストグループのどちらの条件を削除するかを指定します。**group** または **hostgroup** を入力します。
- **Inclusive regex** および **Exclusive regex**。この正規表現で、削除する条件を指定します。条件を 1 つだけ指定する場合は、他の条件の入力を求められたら **Enter** を押します。
  以下に例を示します。

```
$ ipa automember-remove-condition
Automember Rule: user_group
Attribute Key: uid
Grouping Type: group
[Inclusive Regex]: .*
[Exclusive Regex]:
--------------------------------
Removed condition(s) from "user_group"
--------------------------------
  Automember Rule: user_group
------------------------------
Number of conditions removed 1
------------------------------
```

17.8. IDM CLI を使用した AUTOMEMBER ルールの既存のエントリーへの適用

Automember ルールは、ルールの追加後に作成されたユーザーおよびホストエントリーに自動的に適用されます。Automember ルールは、ルールの追加前に既存のエントリーに対して遡って適用されることはありません。

以前に追加したエントリーに automember ルールを適用するには、自動メンバーシップを手動で再構築する必要があります。自動メンバーシップを再構築すると、既存の automember ルールがすべて再評価され、すべてのユーザーまたはホストエントリーまたは特定のエントリーに適用されます。

注記

エントリーがグループの包含条件に一致しない場合でも、自動メンバーシップを再構築しても、グループからユーザーまたはホストエントリーは削除 **されません**。手動で削除するには、IdM CLI を使用してユーザーグループからメンバーを削除 または CLI で IdM ホストグループメンバーの削除 を参照してください。

前提条件

- 管理者としてログインしている。詳細は、Using kinit to log in to IdM manually を参照してください。

手順

- 自動メンバーシップを再構築するには、**ipa automember-rebuild** コマンドを実行します。以下のオプションを指定して、ターゲットにエントリーを指定します。
  - 全ユーザーの自動メンバーシップを再構築するには、**--type=group** オプションを使用します。

```
$ ipa automember-rebuild --type=group
--------------------------------------------------------
```

```
Automember rebuild task finished. Processed (9) entries.
-----------------------------------------------------
```

- 全ホストの自動メンバーシップを再構築するには、**--type=hostgroup** オプションを使用します。
- 指定したユーザーの自動メンバーシップを再構築するには、**--users=target_user** オプションを使用します。

```
$ ipa automember-rebuild --users=target_user1 --users=target_user2
-----------------------------------------------------
Automember rebuild task finished. Processed (2) entries.
-----------------------------------------------------
```

- 指定したホストの自動メンバーシップを再構築するには、**--hosts=client.idm.example.com** を使用します。

17.9. デフォルトの AUTOMEMBER グループの設定

デフォルトの automember グループを設定すると、automember ルールに一致しない新規ユーザーまたはホストエントリーは自動的にこのデフォルトグループに追加されます。

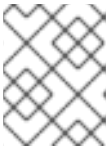
前提条件

- 管理者としてログインしている。詳細は、Using kinit to log in to IdM manually を参照してください。
- デフォルトとして設定されるターゲットグループが IdM にある。

手順

1. **ipa automember-default-group-set** コマンドを入力して、デフォルトの automember グループを設定します。
2. プロンプトが表示されたら、以下を指定します。
   - **Default (fallback) Group**。ターゲットグループ名を指定します。
   - **Grouping Type**。ターゲットがユーザーグループか、ホストグループであるかを指定します。ユーザーグループを対象に設定するには、**group** と入力します。ホストグループを対象に設定するには、**hostgroup** と入力します。
     以下に例を示します。

```
$ ipa automember-default-group-set
Default (fallback) Group: default_user_group
Grouping Type: group
-------------------------------------------------
Set default (fallback) group for automember "default_user_group"
-------------------------------------------------
  Default (fallback) Group:
cn=default_user_group,cn=groups,cn=accounts,dc=example,dc=com
```



注記

現在のデフォルトの automember グループを削除するには、**ipa automember-default-group-remove** コマンドを実行します。

検証手順

- グループが正しく設定されていることを確認するには、**ipa automember-default-group-show** コマンドを実行します。コマンドは、現在のデフォルトの automember グループを表示します。以下に例を示します。

```
$ ipa automember-default-group-show
Grouping Type: group
  Default (fallback) Group:
cn=default_user_group,cn=groups,cn=accounts,dc=example,dc=com
```

第18章 IDM WEB UI を使用したグループメンバーシップの自動化

自動グループメンバーシップを使用すると、属性に基づいてユーザーとホストをグループに自動的に割り当てることができます。たとえば、以下を行うことができます。

- 従業員のマネージャー、ロケーション、またはその他の属性に基づいて従業員のユーザーエントリーをグループに分類する。
- クラス、ロケーション、またはその他の属性に基づいてホストを分類する。
- 全ユーザーまたは全ホストを 1 つのグローバルグループに追加する。

本章では、以下のトピックについて説明します。

18.1. 自動グループメンバーシップの利点

ユーザーの自動メンバーシップを使用すると、以下が可能になります。

- **グループメンバーシップの手動管理してオーバーヘッドを削減する**
  すべてのユーザーおよびグループを手作業で割り当てる必要がなくなります。
- **ユーザーおよびホスト管理における一貫性を向上する**
  ユーザーとホストは、厳密に定義された基準かつ自動評価された基準をもとにグループに割り当てられます。
- **グループベースの設定管理を簡素化する**
  さまざまな設定がグループに定義され、**sudo** ルール、自動マウント、またはアクセス制御などの個別のグループメンバーに適用されます。ユーザーとホストをグループに自動的に追加すると、この設定の管理が容易になります。

18.2. AUTOMEMBER ルール

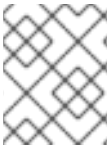
自動グループメンバーシップを設定するには、管理者は automember ルールを定義します。automember ルールは、特定のユーザーまたはホストターゲットグループに適用されます。これは、一度に複数のグループに適用できません。

管理者はルールの作成後に条件を追加します。条件では、ユーザーまたはホストをターゲットグループに含めるか、除外するかを指定します。

- **包含条件**
  ユーザーまたはホストのエントリーが包含条件を満たす場合には、ターゲットグループに含まれます。

- **除外条件**
  ユーザーまたはホストのエントリーが除外条件を満たす場合には、ターゲットグループには含まれません。

この条件は、Perl 互換正規表現 (PCRE) 形式の正規表現として指定します。PCRE の詳細は、**pcresyntax(3)** の man ページを参照してください。



注記

IdM は、包含条件よりも除外条件を先に評価します。競合が発生した場合は、包含条件よりも除外条件が優先されます。

automember ルールは、今後作成されるすべてのエントリーに適用されます。このエントリーは指定されたターゲットグループに自動的に追加されます。エントリーが複数の automember ルールで指定された条件を満たす場合には、該当するグループすべてに追加されます。

既存のエントリーは新規ルールの影響を **受けません**。既存のエントリーを変更する場合は、IdM Web UI を使用した automember ルールの既存エントリーへの適用 を参照してください。

18.3. IDM WEB UI を使用した AUTOMEMBER ルールの追加

IdM Web UI を使用して automember ルールを追加するには、次の手順に従います。automember ルールの詳細は、automember ルール を参照してください。

注記

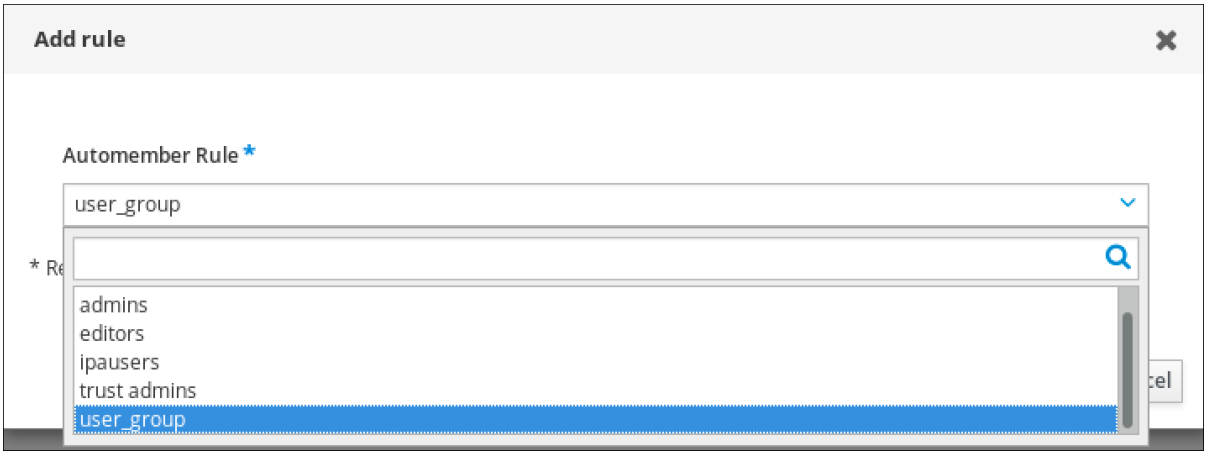
既存のエントリーは新規ルールの影響を **受けません**。既存のエントリーを変更する場合は、IdM Web UI を使用した automember ルールの既存エントリーへの適用 を参照してください。

前提条件

- IdM Web UI にログインしている。
- **admins** グループのメンバーである。
- 新しいルールのターゲットグループが IdM に存在する。

手順

1. **Identity → Automember** をクリックして、**User group rule** または **Host group rules** を選択します。
2. **Add** をクリックします。
3. **Automember rule** フィールドで、ルールを適用するグループを選択します。これはターゲットグループ名です。



4. **Add** をクリックして確定します。

5. 必要に応じて、IdM Web UI を使用した automember ルールへの条件の追加 で説明されている手順に従って、新しいルールに条件を追加できます。

18.4. IDM WEB UI を使用した AUTOMEMBER ルールへの条件の追加

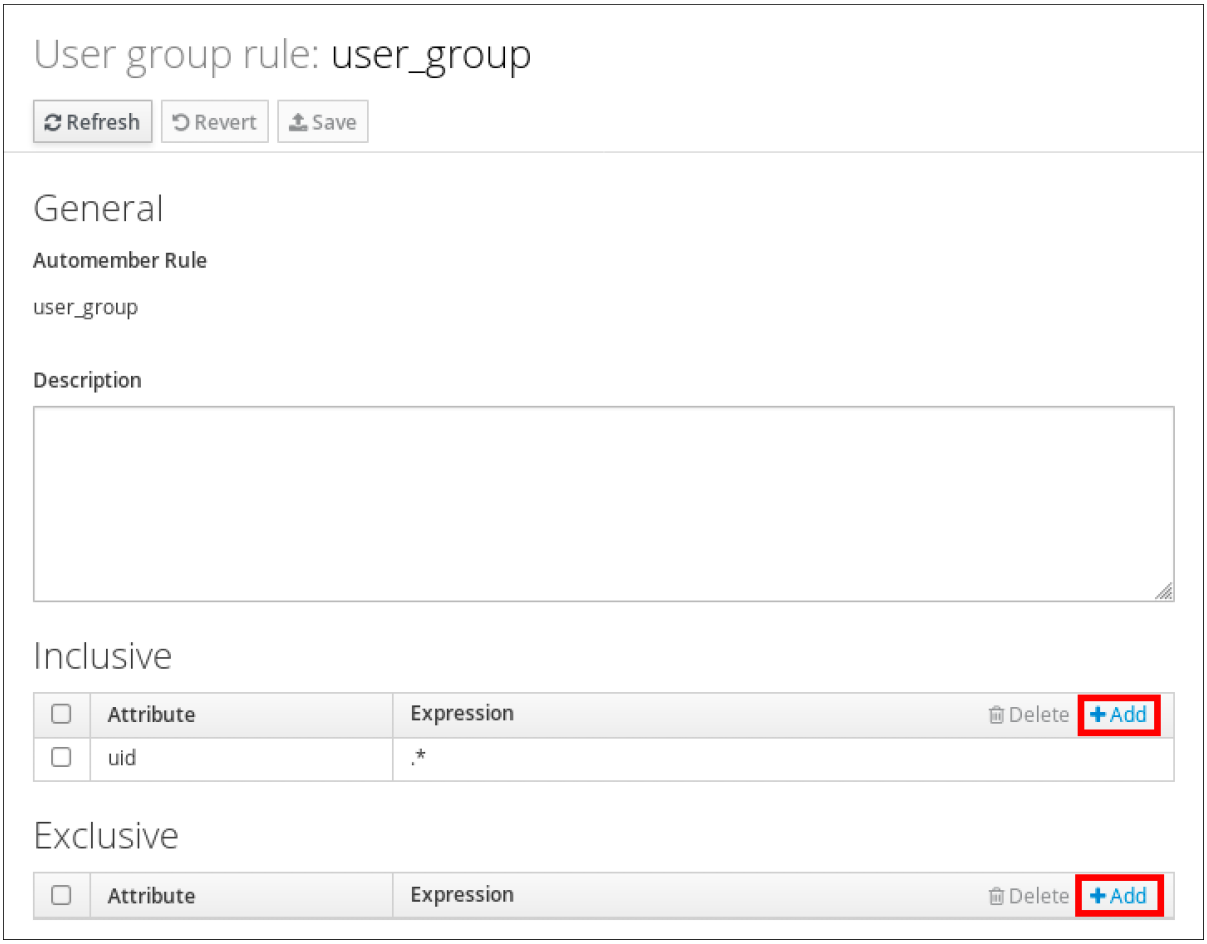
automember ルールを設定した後、IdM Web UI を使用してその automember ルールに条件を追加できます。automember ルールの詳細は、automember ルール を参照してください。

前提条件

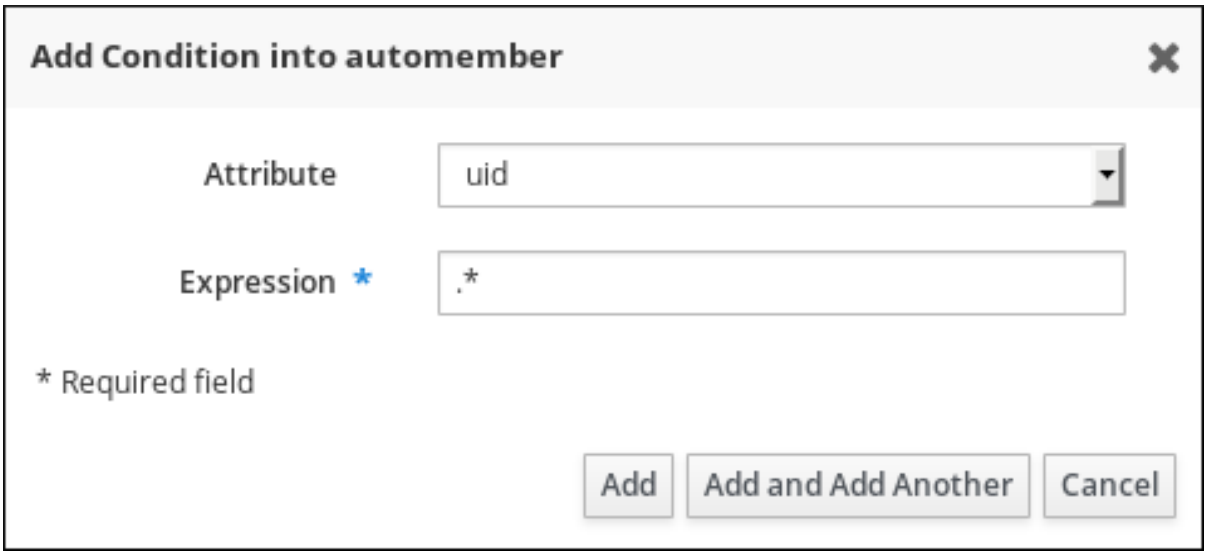
- IdM Web UI にログインしている。
- **admins** グループのメンバーである。
- ターゲットルールが IdM に存在する。

手順

1. **Identity → Automember** をクリックして、**User group rule** または **Host group rules** を選択します。
2. 条件を追加するルールをクリックします。
3. **Inclusive** セクションまたは **Exclusive** セクションで、Add をクリックします。



4. **Attribute** フィールドで、必要な属性 (**uid** など) を選択します。
5. **Expression** フィールドに正規表現を定義します。
6. **Add** をクリックします。
   たとえば、以下の条件は、ユーザー ID (uid) 属性に値 (.*) が指定されているすべてのユーザーを対象とします。



18.5. IDM WEB UI を使用した既存の AUTOMEMBER ルールおよび条件の表示

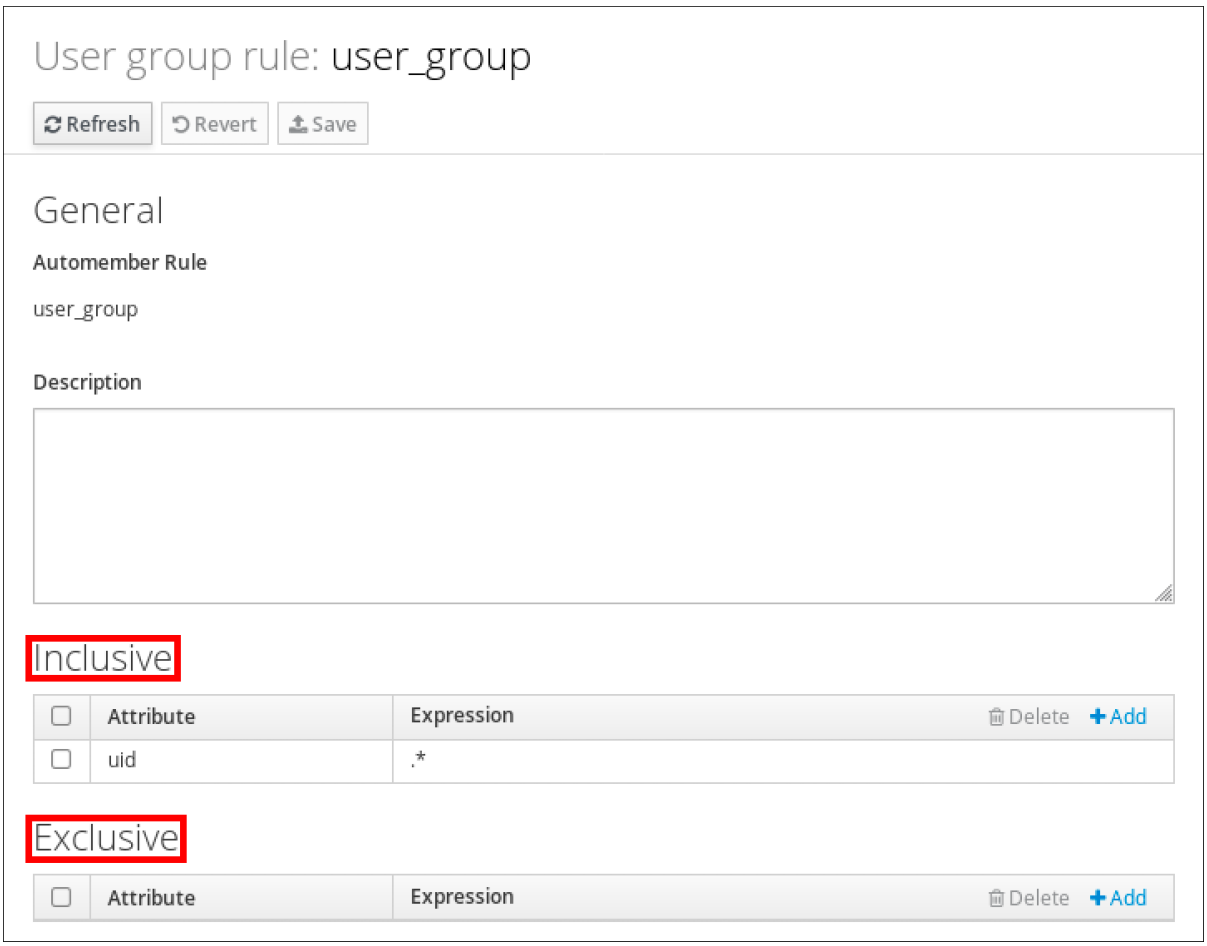
IdM Web UI を使用して既存の automember ルールと条件を表示するには、次の手順に従います。

前提条件

- IdM Web UI にログインしている。
- **admins** グループのメンバーである。

手順

1. **Identity → Automember** をクリックして、**User group rule** または **Host group rules** を選択して、それぞれの automember ルールを表示します。
2. 必要に応じて、ルールをクリックして、**Inclusive** セクションまたは **Exclusive** セクションにそのルールの条件を表示します。



18.6. IDM WEB UI を使用した AUTOMEMBER ルールの削除

IdM Web UI を使用して automember ルールを削除するには、次の手順に従います。

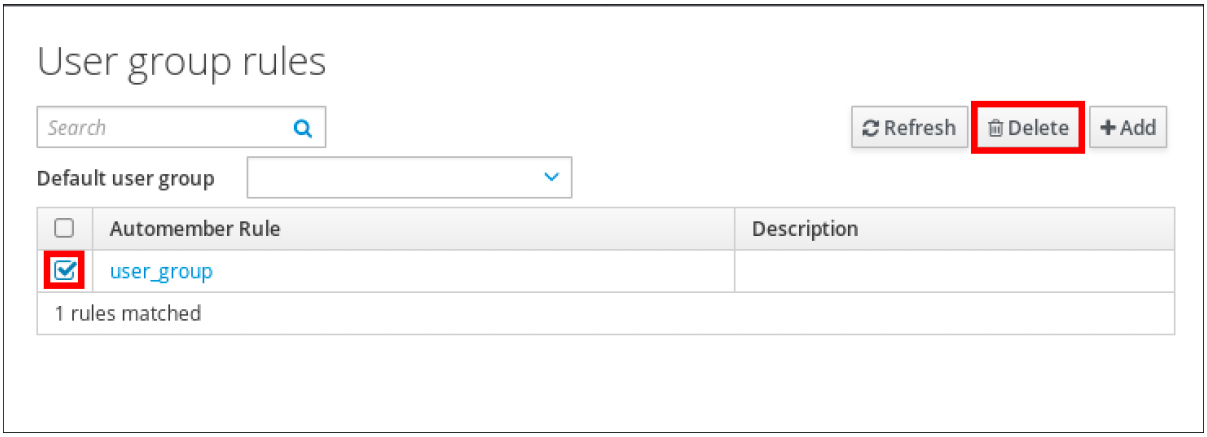
automember ルールを削除すると、そのルールに関連付けられた条件もすべて削除されます。ルールから特定の条件のみを削除するには、IdM Web UI を使用した automember ルールからの条件削除 を参照してください。

前提条件

- IdM Web UI にログインしている。
- **admins** グループのメンバーである。

手順

1. **Identity → Automember** をクリックして、**User group rule** または **Host group rules** を選択して、それぞれの automember ルールを表示します。
2. 削除するルールの横にあるチェックボックスを選択します。
3. **Delete** をクリックします。



4. **Delete** をクリックして確定します。

18.7. IDM WEB UI を使用した AUTOMEMBER ルールからの条件削除

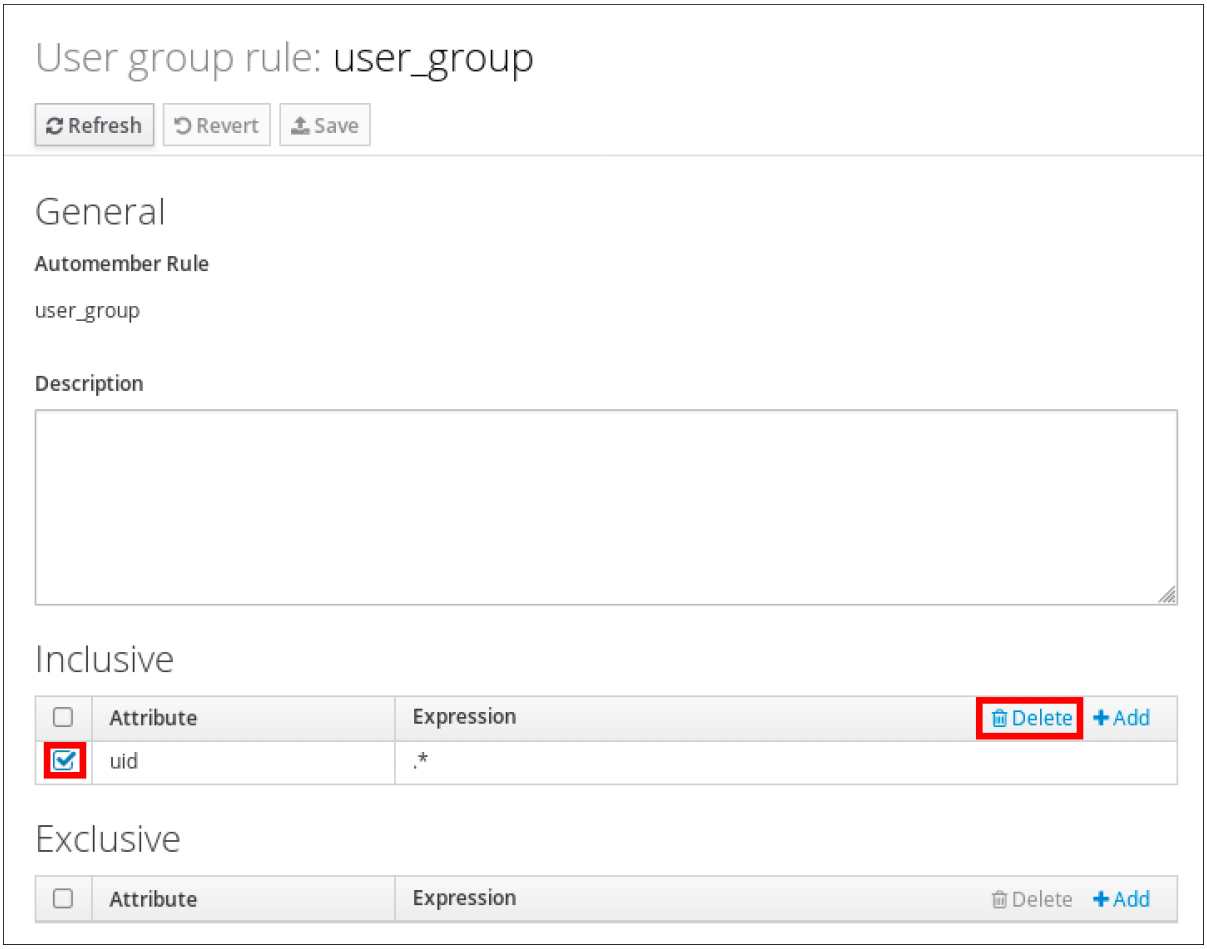
IdM Web UI を使用して automember ルールから特定の条件を削除するには、次の手順に従います。

前提条件

- IdM Web UI にログインしている。
- **admins** グループのメンバーである。

手順

1. **Identity → Automember** をクリックして、**User group rule** または **Host group rules** を選択して、それぞれの automember ルールを表示します。
2. ルールをクリックして、**Inclusive** セクションまたは **Exclusive** セクションでそのルールの条件を表示します。
3. 削除する条件の横にあるチェックボックスを選択します。
4. **Delete** をクリックします。



5. **Delete** をクリックして確定します。

18.8. IDM WEB UI を使用した AUTOMEMBER ルールの既存エントリーへの適用

Automember ルールは、ルールの追加後に作成されたユーザーおよびホストエントリーに自動的に適用されます。Automember ルールは、ルールの追加前に既存のエントリーに対して遡って適用されることはありません。

以前に追加したエントリーに automember ルールを適用するには、自動メンバーシップを手動で再構築する必要があります。自動メンバーシップを再構築すると、既存の automember ルールがすべて再評価され、すべてのユーザーまたはホストエントリーまたは特定のエントリーに適用されます。

注記

エントリーがグループの包含条件に一致しない場合でも、自動メンバーシップを再構築しても、グループからユーザーまたはホストエントリーは削除 **されません**。手動で削除するには、IdM Web UI を使用してユーザーグループからメンバーの削除 または IdM Web UI でホストグループメンバーの削除 を参照してください。

18.8.1. 全ユーザーまたは全ホストの自動メンバーシップの再構築

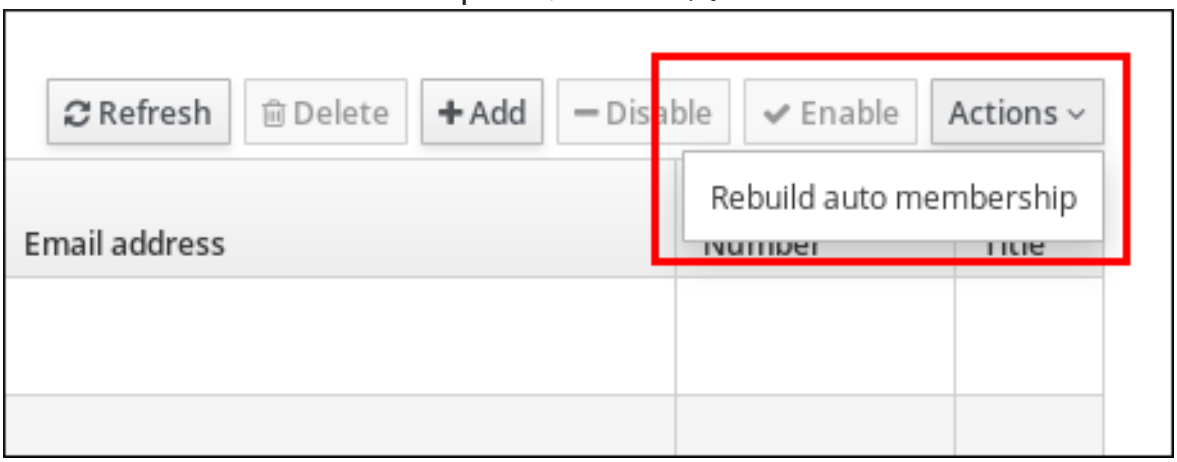
すべてのユーザーまたはホストエントリーの自動メンバーシップを再構築するには、次の手順に従います。

前提条件

- IdM Web UI にログインしている。
- **admins** グループのメンバーである。

手順

1. **Identity → Users** または **Hosts** を選択します。
2. **Actions → Rebuild auto membership** をクリックします。



18.8.2. ユーザーまたはホスト 1 つに対する自動メンバーシップの再構築

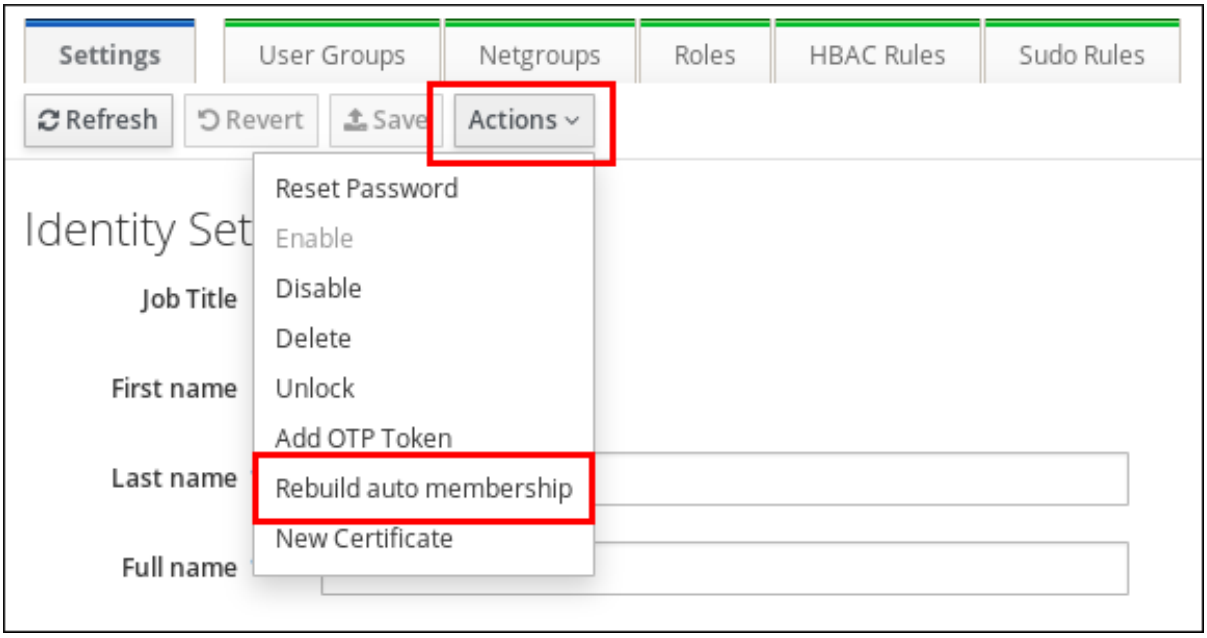
特定のユーザーまたはホストエントリーの自動メンバーシップを再構築するには、次の手順に従います。

前提条件

- IdM Web UI にログインしている。
- **admins** グループのメンバーである。

手順

1. **Identity → Users** または **Hosts** を選択します。
2. 必要なユーザー名またはホスト名をクリックします。
3. **Actions → Rebuild auto membership** をクリックします。



18.9. IDM WEB UI を使用したデフォルトのユーザーグループの設定

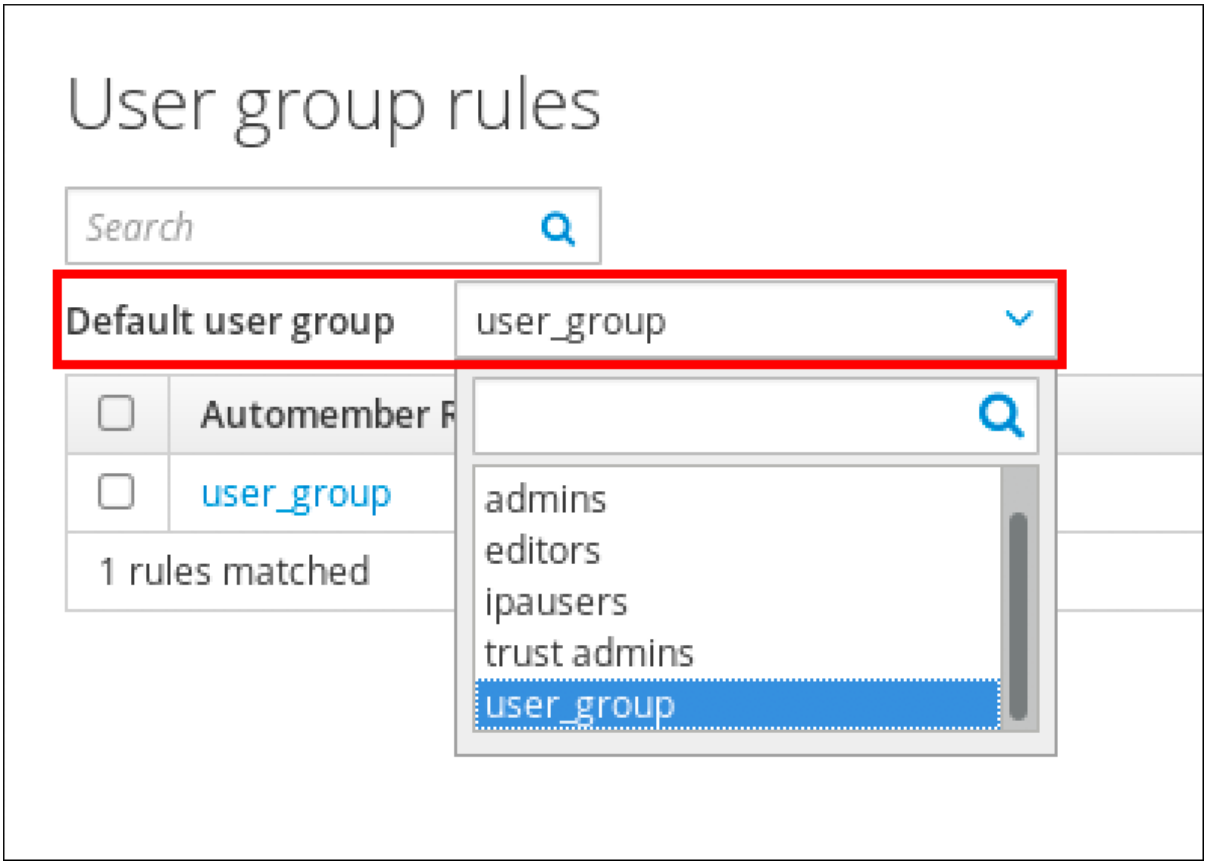
デフォルトのユーザーグループの設定時に、automember ルールに一致しない新規ユーザーエントリーは自動的にこのデフォルトグループに追加されます。

前提条件

- IdM Web UI にログインしている。
- **admins** グループのメンバーである。
- デフォルトとして設定するターゲットユーザーグループが IdM にある。

手順

1. **Identity → Automember** をクリックして、**User group rules** を選択します。
2. **Default user group** フィールドで、デフォルトのユーザーグループとして設定するグループを選択します。



18.10. IDM WEB UI を使用したデフォルトのホストグループの設定

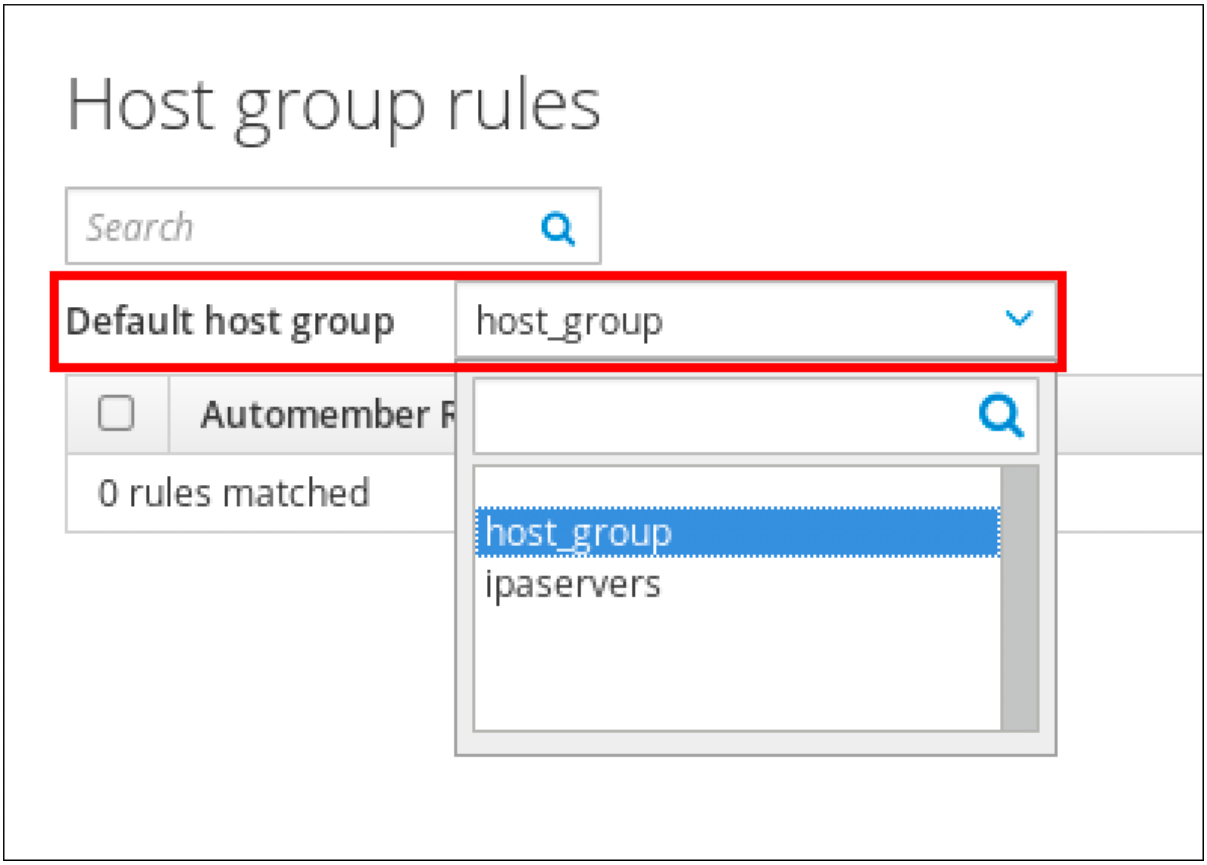
デフォルトのホストグループの設定時に、automember ルールに一致しない新規ホストエントリーが自動的にこのデフォルトグループに追加されます。

前提条件

- IdM Web UI にログインしている。
- **admins** グループのメンバーである。
- デフォルトとして設定されるターゲットホストグループが IdM にある。

手順

1. **Identity → Automember** をクリックして、**Host group rules** を選択します。
2. **Default host group** フィールドで、デフォルトのホストグループとして設定するグループを選択します。


Host group rules
Search
Default host group
host_group
Automember R
0 rules matched
host_group
ipaservers

第19章 ANSIBLE を使用した IDM のグループメンバーシップの自動化

自動グループメンバーシップを使用すると、ユーザーとホストのユーザーグループとホストグループを、その属性に基づいて自動的に割り当てることができます。たとえば、以下を行うことができます。

- 従業員のユーザーエントリーを、従業員のマネージャー、場所、役職などの属性に基づいてグループに分割します。コマンドラインに **ipa user-add --help** と入力すると、すべての属性をリスト表示できます。
- ホストを、クラス、場所、またはその他の属性に基づいてグループに分割します。コマンドラインに **ipa host-add --help** と入力すると、すべての属性をリスト表示できます。
- 全ユーザーまたは全ホストを 1 つのグローバルグループに追加する。

Red Hat Ansible Engine を使用すると、Identity Management (IdM) で自動グループメンバーシップの管理を自動化できます。

このセクションでは、以下のトピックについて説明します。

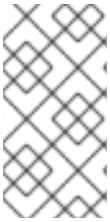
- IdM 管理用の Ansible コントロールノードの準備
- Ansible を使用した IdM ユーザーグループの automember ルールが存在することの確認
- Ansible を使用した IdM ユーザーグループの automember ルールに条件が存在することの確認
- Ansible を使用した IdM ユーザーグループの automember ルールに条件がないことの確認
- Ansible を使用した IdM グループの automember ルールがないことの確認
- Ansible を使用した IdM ホストグループの automember ルールに条件が存在することの確認

19.1. IDM 管理用の ANSIBLE コントロールノードの準備

Identity Management (IdM) を管理するシステム管理者は、Red Hat Ansible Engine を使用する際に以下を行うことが推奨されます。

- ホームディレクトリーに Ansible Playbook 専用のサブディレクトリー (例: **~/MyPlaybooks**) を作成します。
- **/usr/share/doc/ansible-freeipa/*** と **/usr/share/doc/rhel-system-roles/*** ディレクトリーおよびサブディレクトリーから **~/MyPlaybooks** ディレクトリーにサンプル Ansible Playbook をコピーして調整します。
- **~/MyPlaybooks** ディレクトリーにインベントリーファイルを追加します。

この方法に従うことで、すべての Playbook を 1 カ所で見つけることができます。また、root 権限を呼び出さなくても Playbook を実行できます。



注記

ipaserver、**ipareplica**、**ipaclient**、**ipabackup**、**ipasmartcard_server**、および **ipasmartcard_client ansible-freeipa** のロールを実行するために必要なのは、管理対象ノードでの **root** 権限のみです。これらのロールには、ディレクトリーおよび **dnf** ソフトウェアパッケージマネージャーへの特権アクセスが必要です。

~/MyPlaybooks ディレクトリーを作成し、それを使用して Ansible Playbook を保存および実行できるように設定するには、次の手順に従います。

前提条件

- 管理対象ノードに IdM サーバー (**server.idm.example.com** および **replica.idm.example.com**) をインストールしている。
- DNS およびネットワークを設定し、コントロールノードから直接管理対象ノード (**server.idm.example.com** および **replica.idm.example.com**) にログインすることができる。
- IdM **admin** のパスワードを把握している。

手順

1. Ansible 設定および Playbook のディレクトリーをホームディレクトリーに作成します。

```
$ mkdir ~/MyPlaybooks/
```

2. **~/MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks
```

3. **~/MyPlaybooks/ansible.cfg** ファイルを以下の内容で作成します。

```
[defaults]
inventory = /home/your_username/MyPlaybooks/inventory

[privilege_escalation]
become=True
```

4. **~/MyPlaybooks/inventory** ファイルを以下の内容で作成します。

```
[ipaserver]
server.idm.example.com

[ipareplicas]
replica1.idm.example.com
replica2.idm.example.com

[ipacluster:children]
ipaserver
ipareplicas

[ipacluster:vars]
ipaadmin_password=SomeADMINpassword

[ipaclients]
ipaclient1.example.com
ipaclient2.example.com

[ipaclients:vars]
ipaadmin_password=SomeADMINpassword
```

この設定は、これらの場所にあるホストの 2 つのホストグループ (**eu** と **us**) を定義します。さらに、この設定は、**eu** および **us** グループのすべてのホストを含む **ipaserver** ホストグループを定義します。

5. [オプション] SSH 公開鍵および秘密鍵を作成します。テスト環境でのアクセスを簡素化するには、秘密鍵にパスワードを設定しないでください。

```
$ ssh-keygen
```

6. 各マネージドノードの IdM **admin** アカウントに SSH 公開鍵をコピーします。

```
$ ssh-copy-id admin@server.idm.example.com
$ ssh-copy-id admin@replica.idm.example.com
```

これらのコマンドを入力する場合は、IdM **admin** パスワードを入力する必要があります。

関連情報

- Ansible Playbook で Identity Management サーバーのインストール
- インベントリーの構築方法.

19.2. ANSIBLE を使用した IDM ユーザーグループの AUTOMEMBER ルールが存在することの確認

以下の手順では、Ansible Playbook を使用して、Identity Management (IdM) グループの **automember** ルールが存在することを確認する方法について説明しますこの例では、**testing_group** ユーザーグループに対して **automember** ルールの存在が保証されます。

前提条件

- IdM **admin** のパスワードを把握している。
- **testing_group** ユーザーグループが IdM に存在します。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **~/MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/automember/** ディレクトリーにある **automember-group-present.yml** Ansible Playbook ファイルをコピーします。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/automember/automember-group-present.yml automember-group-present-copy.yml
```

3. **automember-group-present-copy.yml** ファイルを開いて編集します。

4. **ipaautomember** タスクセクションで次の変数を設定して、ファイルを調整します。

   - **ipaadmin_password** 変数は IdM **admin** のパスワードに設定します。
   - **name** 変数を testing_group に設定します。
   - **automember_type** 変数を group に設定します。
   - **state** 変数は **present** に設定されていることを確認します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Automember group present example
  hosts: ipaserver
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure group automember rule admins is present
    ipaautomember:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: testing_group
      automember_type: group
      state: present
```

5. ファイルを保存します。

6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory automember-group-present-copy.yml
```

関連情報

- 自動グループメンバーシップの利点 および automember ルール を参照してください。
- Ansible を使用した IdM ユーザーグループの automember ルールが存在することの確認 を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-automember.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/automember** ディレクトリーを参照してください。

19.3. ANSIBLE を使用した、IDM ユーザーグループの AUTOMEMBER ルールに指定した条件が存在することの確認

以下の手順では、Ansible Playbook を使用して、Identity Management (IdM) グループの **automember** ルールに、指定した条件が存在することを確認する方法について説明しますこの例では、**testing_group** グループに対して、**automember** ルールに UID 関連の条件が存在することが保証されます。.* 条件を指定することで、今後使用する IdM ユーザーがすべて自動的に **testing_group** のメンバーになるようにします。

前提条件

- IdM **admin** のパスワードを把握している。
- **testing_group** ユーザーグループおよび automember ユーザーグループルールが IdM に存在します。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **~/MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/automember/** ディレクトリーにある **automember-hostgroup-rule-present.yml** Ansible Playbook ファイルをコピーして、名前を付けます (**automember-usergroup-rule-present.yml** など)。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/automember/automember-hostgroup-rule-present.yml automember-usergroup-rule-present.yml
```

3. **automember-usergroup-rule-present.yml** を開いて編集します。
4. 次のパラメーターを変更して、ファイルを調整します。
   - ユースケースに対応するように Playbook の名前を変更します (例: **Automember user group rule member present**)。
   - ユースケースに合わせて、タスクの名前を変更します (例: **Ensure an automember condition for a user group is present**)。
   - **ipaautomember** タスクセクションで、以下の変数を設定します。

- **ipaadmin_password** 変数は IdM **admin** のパスワードに設定します。
- **name** 変数を **testing_group** に設定します。
- **automember_type** 変数を **group** に設定します。
- **state** 変数は **present** に設定されていることを確認します。
- **action** 変数が **member** に設定されていることを確認します。
- **inclusive key** 変数を **UID** に設定します。
- **inclusive expression** 変数を .* に設定します。

以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Automember user group rule member present
  hosts: ipaserver
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure an automember condition for a user group is present
    ipaautomember:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: testing_group
      automember_type: group
      state: present
      action: member
      inclusive:
        - key: UID
          expression: .*
```

5. ファイルを保存します。
6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory automember-usergroup-rule-present.yml
```

検証手順

1. IdM 管理者としてログインします。

```
$ kinit admin
```

2. ユーザーを追加します。以下に例を示します。

```
$ ipa user-add user101 --first user --last 101
----------------------
Added user "user101"
----------------------
  User login: user101
  First name: user
```

```
Last name: 101
...
Member of groups: ipausers, testing_group
...
```

関連情報

- IdM CLI を使用した既存エントリーへの automember ルールの適用 を参照してください。
- 自動グループメンバーシップの利点 および automember ルール を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-automember.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/automember** ディレクトリーを参照してください。

19.4. ANSIBLE を使用した IDM ユーザーグループの AUTOMEMBER ルールに条件がないことの確認

以下の手順では、Ansible Playbook を使用して、Identity Management (IdM) グループの **automember** ルールに条件がないことを確認する方法を説明します。この例では、**automember** ルールに条件がないことが保証されており、**initials** が**dp** であるユーザーを含める必要があることを指定しています。automember ルールが **testing_group** グループに適用されます。条件を適用することにより、今後は、イニシャルが**dp**である IdM ユーザーが **testing_group**のメンバーにならないようにします。

前提条件

- IdM **admin** のパスワードを把握している。
- **testing_group** ユーザーグループおよび automember ユーザーグループルールが IdM に存在します。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **~/MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/automember/** ディレクトリーにある **automember-hostgroup-rule-absent.yml** Ansible Playbook ファイルをコピーして、名前を付けます (**automember-usergroup-rule-absent.yml** など)。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/automember/automember-hostgroup-rule-absent.yml automember-usergroup-rule-absent.yml
```

3. **automember-usergroup-rule-absent.yml** を開いて編集します。

4. 次のパラメーターを変更して、ファイルを調整します。

   - ユースケースに対応するように Playbook の名前を変更します (例: **Automember user group rule member absent**)。
   - ユースケースに合わせて、タスクの名前を変更します (例: **Ensure an automember condition for a user group is absent**)。
   - **ipaautomember** タスクセクションで、以下の変数を設定します。
     - **ipaadmin_password** 変数は IdM **admin** のパスワードに設定します。
     - **name** 変数を **testing_group** に設定します。
     - **automember_type** 変数を **group** に設定します。
     - **state** 変数は、**absent** に設定されていることを確認します。
     - **action** 変数が **member** に設定されていることを確認します。
     - **inclusive key** 変数を **initials** に設定します。
     - **inclusive expression** 変数を **dp** に設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Automember user group rule member absent
  hosts: ipaserver
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure an automember condition for a user group is absent
    ipaautomember:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: testing_group
      automember_type: group
      state: absent
      action: member
      inclusive:
        - key: initials
          expression: dp
```

5. ファイルを保存します。

6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory automember-usergroup-rule-absent.yml
```

検証手順

1. IdM 管理者としてログインします。

```
$ kinit admin
```

2. automember グループを表示します。

```
$ ipa automember-show --type=group testing_group
 Automember Rule: testing_group
```

出力に **Inclusive Regex: initials=dp** エントリーがない場合は、**testing_group** automember ルールに指定した条件が含まれていないことを確認します。

関連情報

- IdM CLI を使用した既存エントリーへの automember ルールの適用 を参照してください。
- 自動グループメンバーシップの利点 および automember ルール を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-automember.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/automember** ディレクトリーを参照してください。

19.5. ANSIBLE を使用した IDM ユーザーグループの AUTOMEMBER ルールがないことの確認

以下の手順では、Ansible Playbook を使用して、Identity Management (IdM) グループに **automember** ルールがないことを確認する方法を説明します。この例では、**testing_group** グループに **automember** ルールがないことが保証されます。

注記

automember ルールを削除すると、そのルールに関連付けられた条件もすべて削除されます。ルールから特定の条件のみを削除するには、Ansible を使用した IdM ユーザーグループの automember ルールに条件がないことの確認 を参照してください。

前提条件

- IdM **admin** のパスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。

- この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。

- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **~/MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/automember/** ディレクトリーにある **automember-group-absent.yml** Ansible Playbook ファイルをコピーします。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/automember/automember-group-absent.yml automember-group-absent-copy.yml
```

3. **automember-group-absent-copy.yml** を開いて編集します。

4. **ipaautomember** タスクセクションで次の変数を設定して、ファイルを調整します。

   - **ipaadmin_password** 変数は IdM **admin** のパスワードに設定します。
   - **name** 変数を testing_group に設定します。
   - **automember_type** 変数を group に設定します。
   - **state** 変数は、**absent** に設定されていることを確認します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Automember group absent example
  hosts: ipaserver
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure group automember rule admins is absent
    ipaautomember:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: testing_group
      automember_type: group
      state: absent
```

5. ファイルを保存します。

6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory automember-group-absent.yml
```

関連情報

- 自動グループメンバーシップの利点 および automember ルール を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-automember.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/automember** ディレクトリーを参照してください。

19.6. ANSIBLE を使用した IDM ホストグループの AUTOMEMBER ルールに条件が存在することの確認

以下の手順に従って、Ansible を使用して、IdM ホストグループの automember ルールに条件が存在することを確認します。この例では、**FQDN** が **.*.idm.example.com** のホストが、**primary_dns_domain_hosts** ホストグループのメンバーであることと、**FQDN** が **.*.example.org** であるホストが **primary_dns_domain_hosts** ホストグループのメンバーではないことを確認する方法を説明します。

前提条件

- IdM **admin** のパスワードを把握している。
- **primary_dns_domain_hosts** ホストグループおよび automember ホストグループルールが IdM に存在します。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **~/MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/automember/** ディレクトリーにある **automember-hostgroup-rule-present.yml** Ansible Playbook ファイルをコピーします。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/automember/automember-hostgroup-rule-present.yml automember-hostgroup-rule-present-copy.yml
```

3. **automember-hostgroup-rule-present-copy.yml** を開いて編集します。
4. **ipaautomember** タスクセクションで次の変数を設定して、ファイルを調整します。

- **ipaadmin_password** 変数は IdM **admin** のパスワードに設定します。
- **name** 変数を **primary_dns_domain_hosts** に設定します。
- **automember_type** を **hostgroup** に設定します。
- **state** 変数は **present** に設定されていることを確認します。
- **action** 変数が **member** に設定されていることを確認します。
- **inclusive key** 変数が **fqdn** に設定されていることを確認します。
- 対応する **inclusive expression** 変数を**.*.idm.example.com**に設定します。
- **exclusive key** 変数を **UID** に設定します。
- 対応する **exclusive expression** 変数を **.*.example.org** に設定します。

以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Automember user group rule member present
  hosts: ipaserver
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure an automember condition for a user group is present
    ipaautomember:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: primary_dns_domain_hosts
      automember_type: hostgroup
      state: present
      action: member
      inclusive:
        - key: fqdn
          expression: .*.idm.example.com
      exclusive:
        - key: fqdn
          expression: .*.example.org
```

5. ファイルを保存します。
6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory automember-hostgroup-rule-present-copy.yml
```

関連情報

- IdM CLI を使用した既存エントリーへの automember ルールの適用 を参照してください。
- 自動グループメンバーシップの利点 および automember ルール を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-automember.md** ファイルを参照してください。

- **/usr/share/doc/ansible-freeipa/playbooks/automember** ディレクトリーを参照してください。

19.7. 関連情報

- Ansible Playbook を使用したユーザーアカウントの管理
- Ansible Playbook を使用したホストの管理
- Ansible Playbook を使用したユーザーグループの管理
- IdM CLI を使用したホストグループの管理

第20章 IDM のアクセス制御

アクセス制御は、ホストやサービスなどの他のユーザーまたはオブジェクトに対して操作を実行するためにユーザーに付与された権限またはアクセス許可を定義します。Identity Management (IdM) では、いくつかのアクセス制御領域を提供して、どのようなアクセスが付与され、誰に付与されているかを明確にします。その一環として、IdM は、ドメイン内のリソースへのアクセス制御と、IdM 設定そのものへのアクセス制御とを区別します。

本章では、IdM ユーザーが、ドメイン内のリソースおよび IdM 設定そのものの両方で利用できるさまざまな内部アクセス制御メカニズムの概要を説明します。

20.1. IDM のアクセス制御命令

Identity Management (IdM) のアクセス制御構造は、389 Directory Server のアクセス制御に基づいています。アクセス制御命令 (ACI) を使用すると、他のエントリー経由で特定の IdM ユーザーのアクセスを許可または拒否できます。IdM ユーザーを含むすべてのエントリーは LDAP に保存されます。

ACI には、以下の 3 つの部分があります

アクター

何かを実行するためのパーミッションが付与されているエンティティーです。LDAP アクセス制御モデルでは、たとえば、ユーザーが識別名 (DN) を使用してディレクトリーにバインドする場合にのみ ACI ルールを適用するように指定できます。このような指定は、**バインドルール** と呼ばれます。これは、ユーザーが誰であるかを定義し、オプションで、特定の時刻または特定のマシンへの試行の制限など、バインドの試行に他の制限を要求できます。

Target

Actor が操作を実行できるエントリー。

操作タイプ

Actor が実行できるアクションのタイプを決定します。最も一般的な操作は、追加、削除、書き込み、読み取り、および検索です。IdM では、管理者以外のユーザーの読み取りおよび検索の権限は制限されており、IdM の CLI よりも IdM の Web UI の方でさらに制限されています。

LDAP 操作を試行すると、以下が発生します。

1. IdM クライアントは、バインド操作の一環として、ユーザー認証情報を IdM サーバーに送信します。
2. IdM サーバーの DS は、ユーザー認証情報を確認します。
3. IdM サーバーの DS は、ユーザーアカウントを確認して、要求された操作を実行するパーミッションをユーザーが持っているかどうかを確認します。

20.2. IDM のアクセス制御方法

Identity Management (IdM) では、アクセス制御方法が以下のカテゴリーに分類されます。

セルフサービスルール

ユーザーがユーザー自身の個人エントリーに対して実行できる操作を定義します。このアクセス制御タイプでは、ユーザーエントリー内の特定の属性に対してのみ書き込みパーミッションを許可します。ユーザーは、特定の属性の値を更新できますが、そのような属性を追加または削除することはできません。

委譲ルール

委譲ルールを使用すると、特定のユーザーグループが、別のユーザーグループ内のユーザーの特定の属性に対して書き込み (編集) 操作を実行できます。セルフサービスルールと同様、この形式のアクセス制御ルールは、特定の属性の値の編集に限られます。エントリー全体を追加または削除したり、指定されていない属性を制御したりする機能は付与されません。

ロールベースのアクセス制御

特別なアクセス制御グループを作成し、IdM ドメイン内のすべてのタイプのエンティティーに対して、より広範囲な権限を付与します。ロールには編集、追加、および削除の権限が付与されるので、選択された属性だけでなくエントリー全体に対する完全な制御が付与されます。
特定のロール (**Enrollment Administrator**、**IT Security Specialist**、および **IT Specialist** など) は、デフォルトで IdM ですでに使用できます。追加のロールを作成して、ホスト、自動マウント設定、netgroup、DNS 設定、IdM 設定などの任意のタイプのエントリーを管理できます。

関連情報

- Ansible Playbook を使用した IdM でのセルフサービスルールの管理
- Ansible Playbook を使用してユーザーグループにパーミッションを委譲してユーザーを管理する手順
- Ansible Playbook を使用した IdM でのロールベースアクセス制御の管理

第21章 CLI を使用した IDM でのセルフサービスルールの管理

Identity Management (IdM) のセルフサービスルールと、コマンドラインインターフェイス (CLI) でセルフサービスアクセスルールを作成および編集する方法について説明します。

21.1. IDM でのセルフサービスアクセス制御

セルフサービスのアクセス制御ルールは、Identity Management (IdM) エンティティーが IdM Directory Server エントリーで実行できる操作を定義します (例: IdM ユーザーが独自のパスワードを更新できるなど)。

この制御方法では、認証された IdM エンティティーがその LDAP エントリー内の特定の属性を編集できますが、エントリー全体での **追加** や **削除** の操作は許可されません。

警告

セルフサービスのアクセス制御規則を使用する場合は注意が必要です。アクセス制御ルールを誤って設定すると、エンティティーの特権が誤って昇格する可能性があります。

21.2. CLI を使用したセルフサービスルールの作成

コマンドラインインターフェイス (CLI) を使用して IdM でセルフサービスアクセスルールを作成するには、次の手順に従います。

前提条件

- IdM、または **ユーザー管理者** ロールを管理する管理者権限
- 有効な Kerberos チケット。詳細は、Using kinit to log in to IdM manually を参照してください。

手順

- セルフサービスルールを追加するには、**ipa selfservice-add** コマンドを使用して、以下の 2 つのオプションを指定します。

  --permissions

  アクセス制御命令 (ACI) が付与する **読み取り** パーミッションおよび **書き込み** パーミッションを設定します。

  --attrs

  ACI がパーミッションを付与する属性の完全なリストを設定します。

たとえば、セルフサービスルールを作成して、ユーザーが独自の名前の詳細を変更できるようにするには、次のコマンドを実行します。

```
$ ipa selfservice-add "Users can manage their own name details" --permissions=write --
attrs=givenname --attrs=displayname --attrs=title --attrs=initials
--------------------------------------------------------------
```

```
Added selfservice "Users can manage their own name details"
----------------------------------------------------------
  Self-service name: Users can manage their own name details
  Permissions: write
  Attributes: givenname, displayname, title, initials
```

21.3. CLI を使用したセルフサービスルールの編集

コマンドラインインターフェイス (CLI) を使用して IdM でセルフサービスアクセスルールを編集するには、次の手順に従います。

前提条件

- IdM、または **ユーザー管理者** ロールを管理する管理者権限
- 有効な Kerberos チケット。詳細は、Using kinit to log in to IdM manually を参照してください。

手順

1. **オプション**: **ipa selfservice-find** コマンドを使用して、既存のセルフサービスルールを表示します。
2. **オプション**: **ipa selfservice-show** コマンドを使用して、変更するセルフサービスルールの詳細を表示します。
3. **ipa selfservice-mod** コマンドを使用してセルフサービスルールを編集します。

以下に例を示します。

```
$ ipa selfservice-mod "Users can manage their own name details" --attrs=givenname --
attrs=displayname --attrs=title --attrs=initials --attrs=surname
----------------------------------------------------------
Modified selfservice "Users can manage their own name details"
----------------------------------------------------------
Self-service name: Users can manage their own name details
Permissions: write
Attributes: givenname, displayname, title, initials
```

重要

ipa selfservice-mod コマンドを使用すると、以前に定義されたパーミッションと属性が上書きされるため、定義する必要のある新しいパーミッションおよび属性と共に、既存のパーミッションおよび属性の完全なリストも常に含めるようにしてください。

検証手順

- **ipa selfservice-show** コマンドを使用して、編集したセルフサービスルールを表示します。

```
$ ipa selfservice-show "Users can manage their own name details"
----------------------------------------------------------
Self-service name: Users can manage their own name details
Permissions: write
Attributes: givenname, displayname, title, initials
```

21.4. CLI を使用したセルフサービスルールの削除

コマンドラインインターフェイス (CLI) を使用して IdM でセルフサービスアクセスルールを削除するには、次の手順に従います。

前提条件

- IdM、または **ユーザー管理者** ロールを管理する管理者権限
- 有効な Kerberos チケット。詳細は、Using kinit to log in to IdM manually を参照してください。

手順

- **ipa selfservice-del** コマンドを使用してセルフサービスルールを削除します。

以下に例を示します。

```
$ ipa selfservice-del "Users can manage their own name details"
--------------------------------------------------------
Deleted selfservice "Users can manage their own name details"
--------------------------------------------------------
```

検証手順

- **ipa selfservice-find** コマンドを使用して、すべてのセルフサービスルールを表示します。削除したばかりのルールがなくなっているはずです。

第22章 IDM WEB UI を使用したセルフサービスルールの管理

Identity Management (IdM) のセルフサービスルールと、Web インターフェイス (IdM Web UI) でセルフサービスアクセスルールを作成および編集する方法について説明します。

22.1. IDM でのセルフサービスアクセス制御

セルフサービスのアクセス制御ルールは、Identity Management (IdM) エンティティーが IdM Directory Server エントリーで実行できる操作を定義します (例: IdM ユーザーが独自のパスワードを更新できるなど)。

この制御方法では、認証された IdM エンティティーがその LDAP エントリー内の特定の属性を編集できますが、エントリー全体での **追加** や **削除** の操作は許可されません。

警告

セルフサービスのアクセス制御規則を使用する場合は注意が必要です。アクセス制御ルールを誤って設定すると、エンティティーの特権が誤って昇格する可能性があります。

22.2. IDM WEB UI を使用したセルフサービスルールの作成

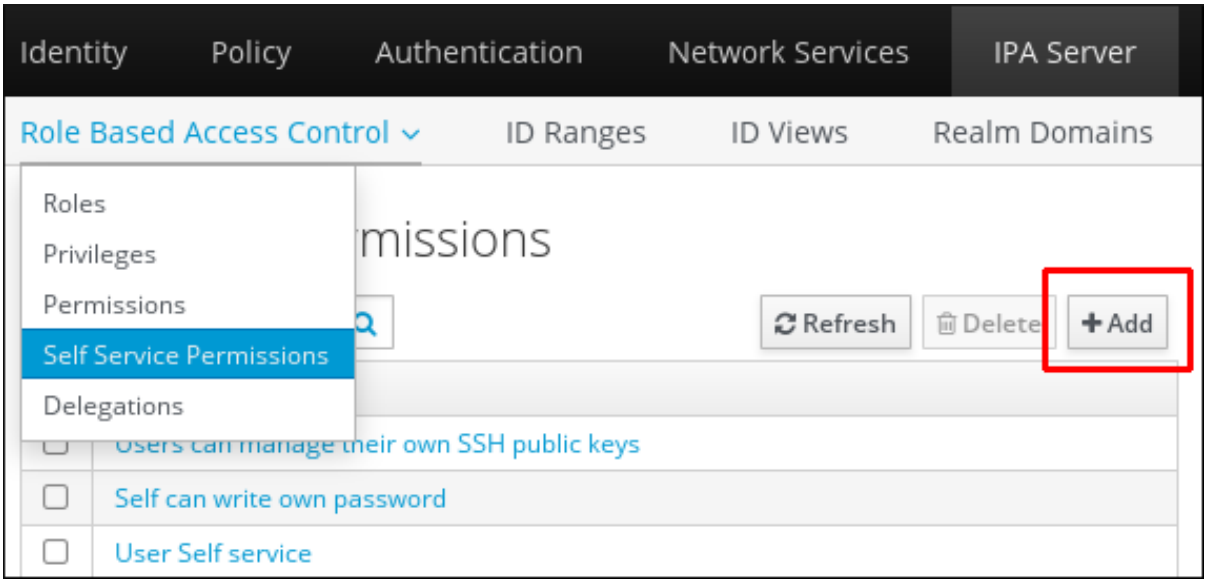
Web インターフェイス (IdM Web UI) を使用して IdM でセルフサービスアクセスルールを作成するには、次の手順に従います。

前提条件

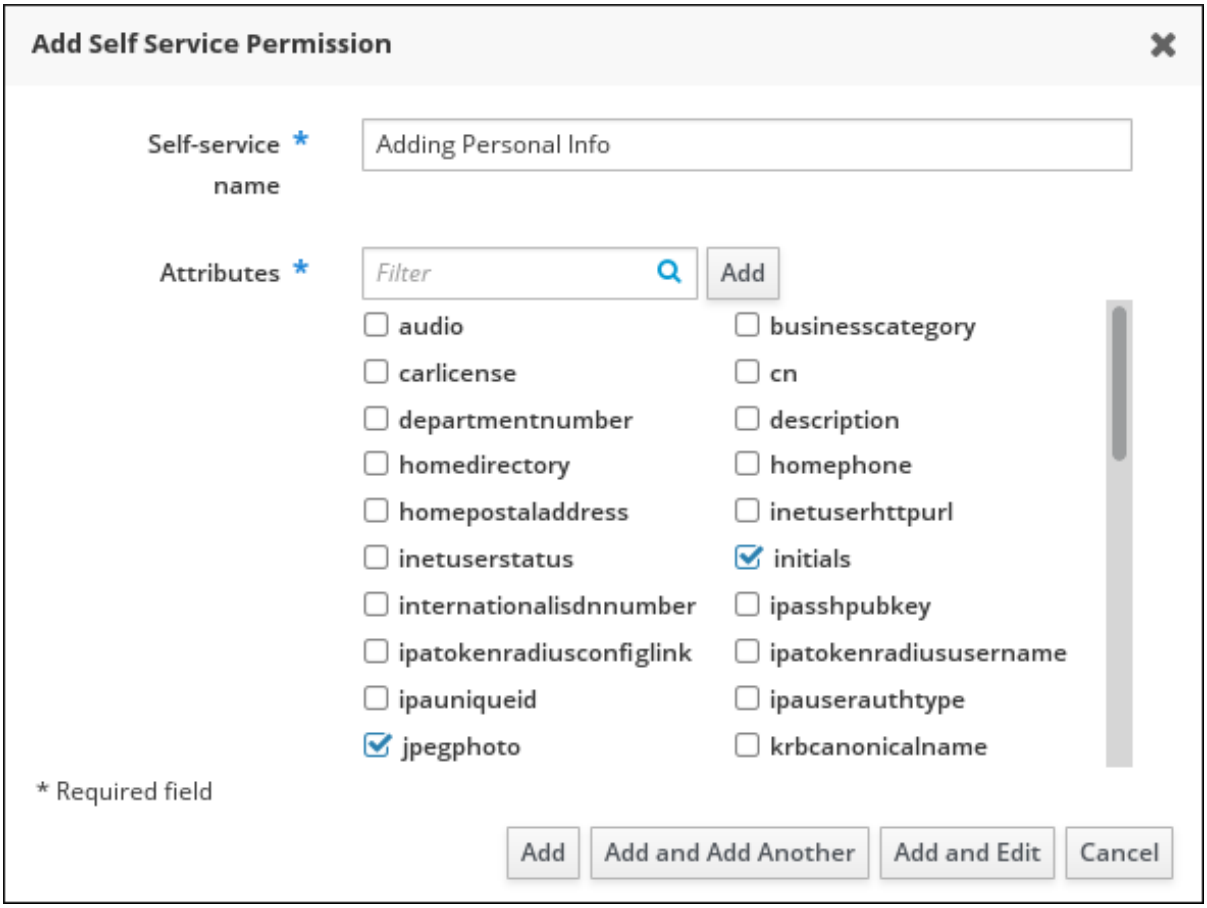
- IdM、または **ユーザー管理者** ロールを管理する管理者権限
- IdM Web UI にログインしている。詳細は Web ブラウザーで IdM Web UI へのアクセス を参照してください。

手順

1. **IPA Server** タブで **Role-Based Access Control** サブメニューを開き、**Self Service Permissions** を選択します。
2. セルフサービスアクセスルールの一覧の右上にある **Add** をクリックします。



3. **Add Self Service Permission** ウィンドウが開きます。**Self-service name** フィールドに新しいセルフサービスルールの名前を入力します。空白を使用できます。



4. ユーザーによる編集を可能にする属性の横にあるチェックボックスを選択します。

5. **必要に応じて** アクセス可能にする属性が一覧にない場合には、一覧に追加できます。

   a. **Add** ボタンをクリックします。

   b. 以下の **Add Custom Attribute** ウィンドウの **Attribute** テキストフィールドに属性名を入力します。

   c. **OK** ボタンをクリックして属性を追加します。

d. 新しい属性が選択されていることを確認します。

6. フォームの下部にある **Add** ボタンをクリックして、新しいセルフサービスルールを保存します。
または、**Add and Edit** ボタンをクリックしてセルフサービスルールを編集するか、**Add and Add another** ボタンをクリックしてさらにルールを保存および追加できます。

22.3. IDM WEB UI を使用したセルフサービスルールの編集

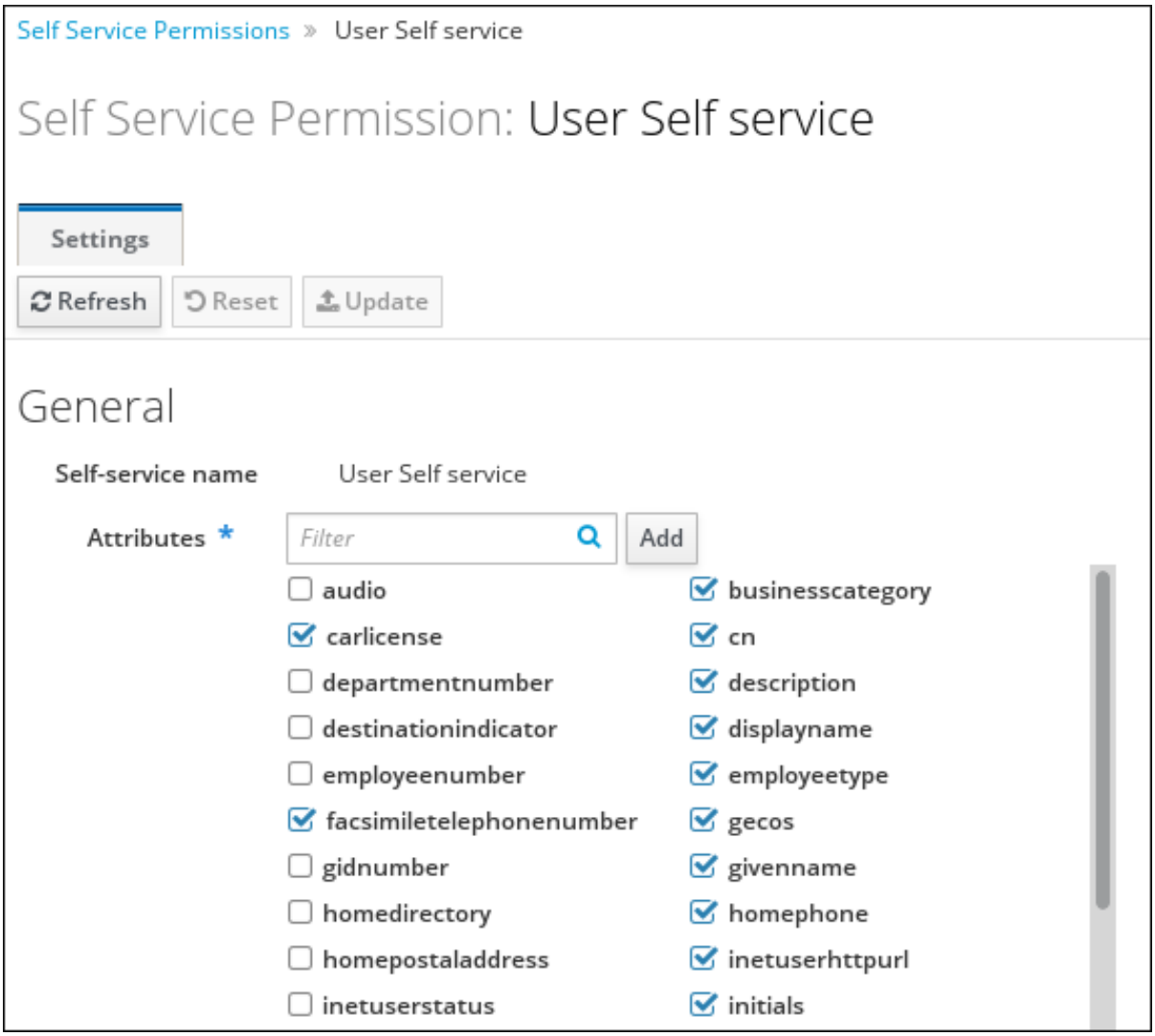
Web インターフェイス (IdM Web UI) を使用して IdM でセルフサービスアクセスルールを編集するには、次の手順に従います。

前提条件

- IdM、または **ユーザー管理者** ロールを管理する管理者権限
- IdM Web UI にログインしている。詳細は Web ブラウザーで IdM Web UI へのアクセス を参照してください。

手順

1. **IPA Server** タブで **Role-Based Access Control** サブメニューを開き、**Self Service Permissions** を選択します。
2. 変更するセルフサービスルールの名前をクリックします。



3. 編集ページでは、セルフサービスルールに追加または削除する属性のリストの編集だけが可能です。適切なチェックボックスを選択または選択解除します。

4. **Save** ボタンをクリックして、セルフサービスルールへの変更を保存します。

22.4. IDM WEB UI を使用したセルフサービスルールの削除

Web インターフェイス (IdM Web UI) を使用して IdM のセルフサービスアクセスルールを削除するには、次の手順に従います。

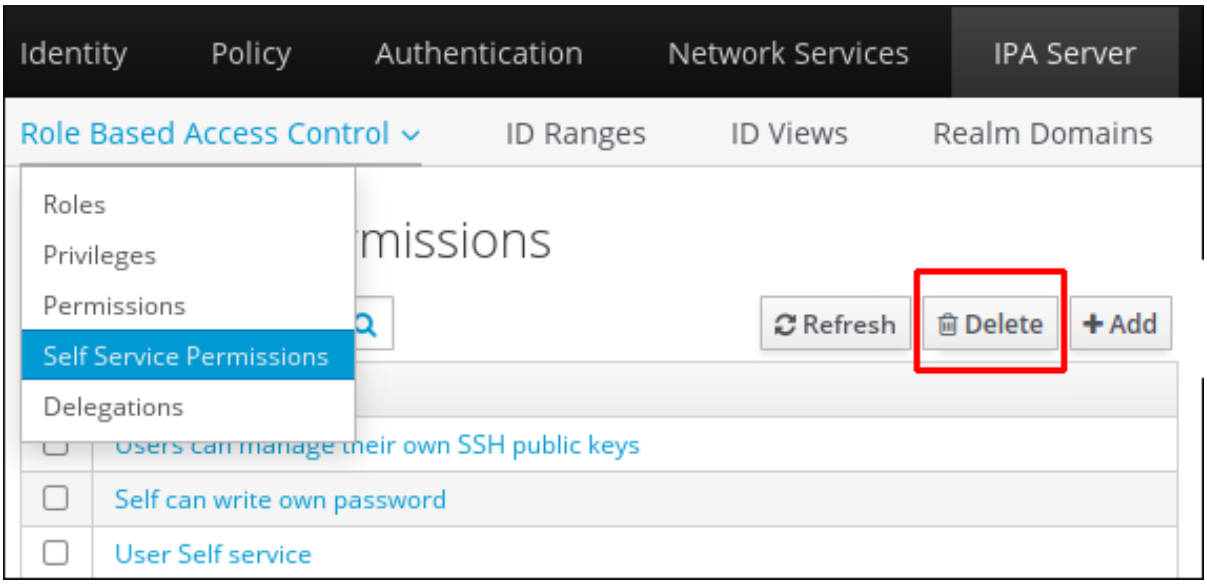
前提条件

- IdM、または **ユーザー管理者** ロールを管理する管理者権限
- IdM Web UI にログインしている。詳細は Web ブラウザーで IdM Web UI へのアクセス を参照してください。

手順

1. **IPA Server** タブで **Role-Based Access Control** サブメニューを開き、**Self Service Permissions** を選択します。

2. 削除するルールの横にあるチェックボックスを選択し、リストの右側にある **Delete** ボタンをクリックします。



3. ダイアログが開き、**Delete** をクリックして確認します。

第23章 ANSIBLE PLAYBOOK を使用した IDM でのセルフサービスルールの管理

本セクションでは、Identity Management (IdM) のセルフサービスルールを紹介し、Ansible Playbook を使用してセルフサービスアクセスルールを作成および編集する方法を説明します。セルフサービスのアクセス制御ルールにより、IdM エンティティーは IdM Directory Server エントリーで指定の操作を実行できます。

- IdM でのセルフサービスアクセス制御
- Ansible を使用してセルフサービスルールを存在させる手順
- Ansible を使用してセルフサービスルールがないことを確認する手順
- Ansible を使用してセルフサービスルールに固有の属性を含める手順
- Ansible を使用してセルフサービスルールに固有の属性を含めいないようにする手順

23.1. IDM でのセルフサービスアクセス制御

セルフサービスのアクセス制御ルールは、Identity Management (IdM) エンティティーが IdM Directory Server エントリーで実行できる操作を定義します (例: IdM ユーザーが独自のパスワードを更新できるなど)。

この制御方法では、認証された IdM エンティティーがその LDAP エントリー内の特定の属性を編集できますが、エントリー全体での **追加** や **削除** の操作は許可されません。

警告

セルフサービスのアクセス制御規則を使用する場合は注意が必要です。アクセス制御ルールを誤って設定すると、エンティティーの特権が誤って昇格する可能性があります。

23.2. ANSIBLE を使用してセルフサービスルールを存在させる手順

以下の手順では、Ansible Playbook を使用してセルフサービスルールを定義し、Identity Management (IdM) サーバーに存在させる方法を説明します。この例では、**ユーザーが自分の名前の情報を管理できる** 新しいルールでは、ユーザーに、自分の **givenname**、**displayname**、**title**、および **initials** 属性を変更できるよ権限を付与します。たとえば、表示名や初期などを変更することができます。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。

- この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
- この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。

- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/selfservice/** ディレクトリーにある **selfservice-present.yml** のコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/selfservice/selfservice-present.yml
selfservice-present-copy.yml
```

3. Ansible Playbook ファイル (**selfservice-present-copy.yml**) を開きます。
4. **ipaselfservice** タスクセクションに以下の変数を設定してファイルを調整します。
   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は、新しいセルフサービスルールの名前に設定します。
   - **permission** 変数は、付与するパーミッションをコンマ区切りのリスト (**read** および **write**) で設定します。
   - **attribute** 変数は、ユーザーが管理できる属性 (**givenname**、**displayname**、**title**、および **initials**) をリストとして設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Self-service present
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure self-service rule "Users can manage their own name details" is present
    ipaselfservice:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: "Users can manage their own name details"
      permission: read, write
      attribute:
      - givenname
      - displayname
      - title
      - initials
```

5. ファイルを保存します。

6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory selfservice-present-copy.yml
```

関連情報

- IdM でのセルフサービスアクセス制御 を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-selfservice.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/selfservice** ディレクトリーを参照してください。

23.3. ANSIBLE を使用してセルフサービスルールがないことを確認する手順

以下の手順では、Ansible Playbook を使用して、指定したセルフサービスルールが IdM 設定に存在しないことを確認する方法を説明します。以下の例では、**ユーザーが自分の名前の詳細** のセルフサービスルールが IdM に存在しないことを確認する方法を説明します。これにより、ユーザーは自分の表示名や初期などを変更できないようにします。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/selfservice/** ディレクトリーにある **selfservice-absent.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/selfservice/selfservice-absent.yml
selfservice-absent-copy.yml
```

3. Ansible Playbook ファイル (**selfservice-absent-copy.yml**) を開きます。
4. **ipaselfservice** タスクセクションに以下の変数を設定してファイルを調整します。
   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は、セルフサービスルールの名前に設定します。
   - **state** 変数は **absent** に設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Self-service absent
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure self-service rule "Users can manage their own name details" is absent
    ipaselfservice:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: "Users can manage their own name details"
      state: absent
```

5. ファイルを保存します。
6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory selfservice-
absent-copy.yml
```

関連情報

- IdM でのセルフサービスアクセス制御 を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-selfservice.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/selfservice** ディレクトリーのサンプルの Playbook を参照してください。

23.4. ANSIBLE を使用してセルフサービスルールに固有の属性を含める手順

以下の手順では、Ansible Playbook を使用して、既存のセルフサービスルールに特定の設定を追加する方法を説明します。この例では、**ユーザーは自分の名前詳細を管理できる** のセルフサービスルールに、**surname** のメンバー属性が含まれるようにします。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- **ユーザーが独自の名前の詳細** セルフサービスルールが IdM に存在する。

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/selfservice/** ディレクトリーにある **selfservice-member-present.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/selfservice/selfservice-member-present.yml selfservice-member-present-copy.yml
```

3. Ansible Playbook ファイル (**selfservice-member-present-copy.yml**) を開きます。
4. **ipaselfservice** タスクセクションに以下の変数を設定してファイルを調整します。
   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は、変更するセルフサービスルールの名前に設定します。
   - **attribte** 変数は **surname** に設定します。
   - **action** 変数は **member** に設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Self-service member present
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure selfservice "Users can manage their own name details" member attribute surname is present
    ipaselfservice:
      ipaadmin_password: "{{ ipaadmin_password }}"
```

```
name: "Users can manage their own name details"
attribute:
- surname
action: member
```

5. ファイルを保存します。

6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory selfservice-member-present-copy.yml
```

関連情報

- IdM でのセルフサービスアクセス制御 を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーで利用可能な **README-selfservice.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/selfservice** ディレクトリーのサンプルの Playbook を参照してください。

23.5. ANSIBLE を使用してセルフサービスルールに固有の属性を含めないようにする手順

以下の手順では、Ansible Playbook を使用して、セルフサービスルールに特定の設定が割り当てられないようにする方法を説明します。この Playbook を使用して、セルフサービスルールで不必要なアクセス権限を付与しないようにします。この例では、**ユーザーが独自の名前の詳細を管理できる** セルフサービスルールに **givenname** と **surname** のメンバー属性が含まれないようにします。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- **ユーザーが独自の名前の詳細** セルフサービスルールが IdM に存在する。

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/selfservice/** ディレクトリーにある **selfservice-member-absent.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/selfservice/selfservice-member-absent.yml selfservice-member-absent-copy.yml
```

3. Ansible Playbook ファイル (**selfservice-member-absent-copy.yml**) を開きます。

4. **ipaselfservice** タスクセクションに以下の変数を設定してファイルを調整します。

   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は、変更するセルフサービスルールの名前に設定します。
   - **属性** 変数は **givenname** および **surname** に設定します。
   - **action** 変数は **member** に設定します。
   - **state** 変数は **absent** に設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Self-service member absent
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure selfservice "Users can manage their own name details" member attributes givenname and surname are absent
    ipaselfservice:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: "Users can manage their own name details"
      attribute:
      - givenname
      - surname
      action: member
      state: absent
```

5. ファイルを保存します。

6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory selfservice-member-absent-copy.yml
```

関連情報

- IdM でのセルフサービスアクセス制御 を参照してください。

- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-selfservice.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/selfservice** ディレクトリーのサンプルの Playbook を参照してください。

第24章 IDM CLI を使用してユーザーグループにパーミッションを委譲してユーザーを管理する手順

委譲は、セルフサービスルールおよびロールベースのアクセス制御 (RBAC) などの IdM のアクセス制御メソッドの 1 つです。委譲を使用して、あるユーザーのグループにパーミッションを割り当てて別のユーザーのグループのエントリーを管理できます。

このセクションでは、以下のトピックについて説明します。

24.1. 委譲ルール

委譲ルール を作成して、ユーザーグループにパーミッションを委譲してユーザーを管理できます。

委譲ルールを使用すると、特定のユーザーグループが、別のユーザーグループ内のユーザーの特定の属性に対して書き込み (編集) 操作を実行できます。このようなアクセス制御ルールは、委譲ルールで指定された属性のサブセットの編集に限定されており、エントリー全体の追加や削除、未指定の属性の制御はできません。

委譲ルールにより、IdM の既存のユーザーグループにパーミッションが付与されます。委任を使用すると、**managers** ユーザーグループで **employees** ユーザーグループでユーザーの選択された属性を管理できます。

24.2. IDM CLI を使用した委譲ルールの作成

IdM CLI を使用して委譲ルールを作成するには、次の手順に従います。

前提条件

- **admins** グループのメンバーとしてログインしている。

手順

- **ipa delegation-add** コマンドを入力します。以下のオプションを指定します。
  - **--group**: ユーザーグループ内のユーザーのエントリーに対する **パーミッションを付与されている** グループです。
  - **--membergroup**: 委譲グループのメンバーが **エントリーを編集できる** グループです。
  - **--permissions**: 指定の属性を表示 (**read**) して、追加または変更 (**write**) する権限をユーザーに指定するかどうか。パーミッションを指定しないと、**書き込み** パーミッションのみが追加されます。
  - **--attrs**: メンバーグループのユーザーが表示または編集できる属性です。

以下に例を示します。

```
$ ipa delegation-add "basic manager attributes" --permissions=read --permissions=write --
attrs=businesscategory --attrs=departmentnumber --attrs=employeetype --
attrs=employeenumber --group=managers --membergroup=employees
----------------------------------------
Added delegation "basic manager attributes"
----------------------------------------
  Delegation name: basic manager attributes
  Permissions: read, write
  Attributes: businesscategory, departmentnumber, employeetype, employeenumber
  Member user group: employees
  User group: managers
```

24.3. IDM CLI を使用した既存の委譲ルールの表示

IdM CLI を使用して既存の委譲ルールを表示するには、次の手順に従います。

前提条件

- **admins** グループのメンバーとしてログインしている。

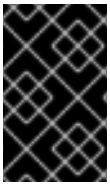
手順

- **ipa delegation-find** コマンドを入力します。

```
$ ipa delegation-find
--------------------
1 delegation matched
--------------------
  Delegation name: basic manager attributes
  Permissions: read, write
  Attributes: businesscategory, departmentnumber, employeenumber, employeetype
  Member user group: employees
  User group: managers
----------------------------
Number of entries returned 1
----------------------------
```

24.4. IDM CLI を使用した委譲ルールの変更

IdM CLI を使用して既存の委譲ルールを変更するには、次の手順に従います。



重要

--attrs オプションはそれまでにサポートされていた属性をすべて上書きするので、新規属性に加えて属性の完全リストを常に含めるようにしてください。これは、**--permissions** オプションにも適用されます。

前提条件

- **admins** グループのメンバーとしてログインしている。

手順

- 必要に応じて、任意の変更を加えて **ipa delegation-mod** コマンドを実行します。たとえば、**displayname** 属性を **basic manager attributes** のルールの例に追加するには、次のコマンドを実行します。

```
$ ipa delegation-mod "basic manager attributes" --attrs=businesscategory --attrs=departmentnumber --attrs=employeetype --attrs=employeenumber --attrs=displayname
---------------------------------------------
Modified delegation "basic manager attributes"
---------------------------------------------
  Delegation name: basic manager attributes
  Permissions: read, write
  Attributes: businesscategory, departmentnumber, employeetype, employeenumber, displayname
  Member user group: employees
  User group: managers
```

24.5. IDM CLI を使用した委譲ルールの削除

IdM CLI を使用して既存の委譲ルールを削除するには、この手順に従います。

前提条件

- **admins** グループのメンバーとしてログインしている。

手順

- **ipa delegation-del** コマンドを入力します。
- プロンプトが表示されたら、削除する委譲ルールの名前を入力します。

```
$ ipa delegation-del
Delegation name: basic manager attributes
---------------------------------------------
Deleted delegation "basic manager attributes"
---------------------------------------------
```

第25章 WEB UI を使用してユーザーグループにパーミッションを委譲してユーザーを管理する手順

委譲は、セルフサービスルールおよびロールベースのアクセス制御 (RBAC) などの IdM のアクセス制御メソッドの 1 つです。委譲を使用して、あるユーザーのグループにパーミッションを割り当てて別のユーザーのグループのエントリーを管理できます。

このセクションでは、以下のトピックについて説明します。

- 委譲ルール
- IdM WebUI を使用した委譲ルールの作成
- IdM WebUI を使用した既存の委譲ルールの表示
- IdM WebUI を使用した委譲ルールの変更
- IdM WebUI を使用した委譲ルールの削除

25.1. 委譲ルール

委譲ルール を作成して、ユーザーグループにパーミッションを委譲してユーザーを管理できます。

委譲ルールを使用すると、特定のユーザーグループが、別のユーザーグループ内のユーザーの特定の属性に対して書き込み (編集) 操作を実行できます。このようなアクセス制御ルールは、委譲ルールで指定された属性のサブセットの編集に限定されており、エントリー全体の追加や削除、未指定の属性の制御はできません。

委譲ルールにより、IdM の既存のユーザーグループにパーミッションが付与されます。委任を使用すると、**managers** ユーザーグループで **employees** ユーザーグループでユーザーの選択された属性を管理できます。

25.2. IDM WEBUI を使用した委譲ルールの作成

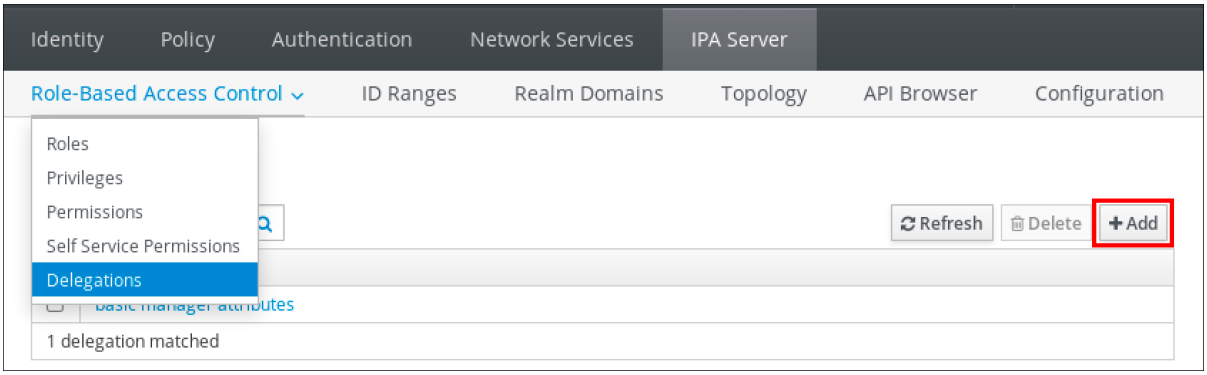
IdM WebUI を使用して委譲ルールを作成するには、次の手順に従います。

前提条件

- **admins** グループのメンバーとして IdM Web UI にログインしている。

手順

1. **IPA Server** メニューから、**Role-Based Access Control** → **Delegations** をクリックします。
2. **Add** をクリックします。



3. **Add delegation** ウィンドウで、以下を実行します。

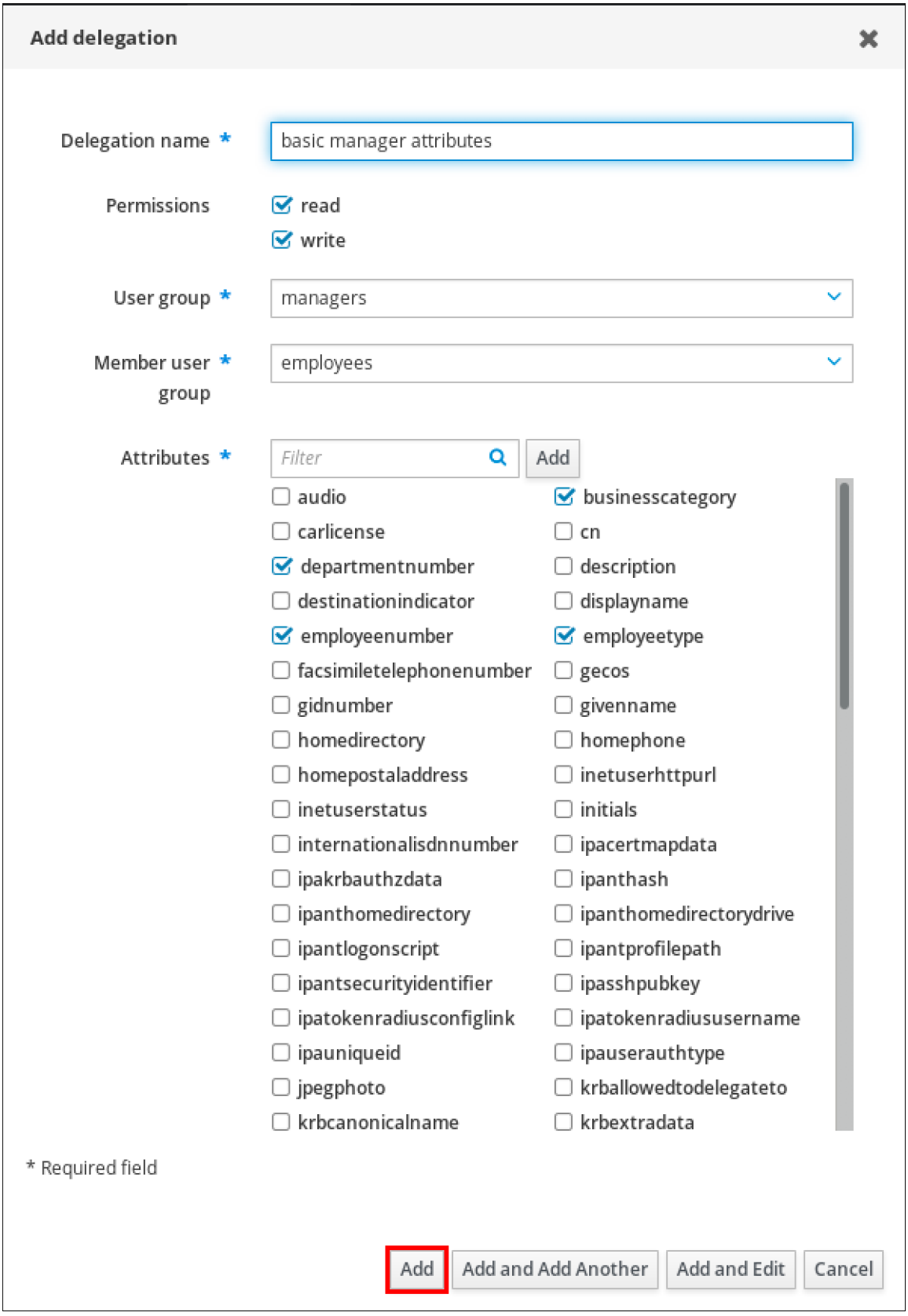
   a. 新しい委譲ルールに名前を付けます。

   b. 指定の属性を表示する権限 (**read**)、および指定の属性を追加または変更する権限 (**write**) をユーザーに指定しているかどうかを示すチェックボックスを選択して、パーミッションを設定します。

   c. ユーザーグループのドロップダウンメニューで、**パーミッションを付与している** グループを選択し、メンバーグループのユーザーエントリーを表示または編集します。

   d. **Member user group** ドロップダウンメニューで、委譲グループのメンバーが **エントリーを編集できる** グループを選択します。

   e. 属性ボックスで、パーミッションを付与する属性のチェックボックスを選択します。



f. **Add** ボタンをクリックして、新規委譲ルールを保存します。

25.3. IDM WEBUI を使用した既存の委譲ルールの表示

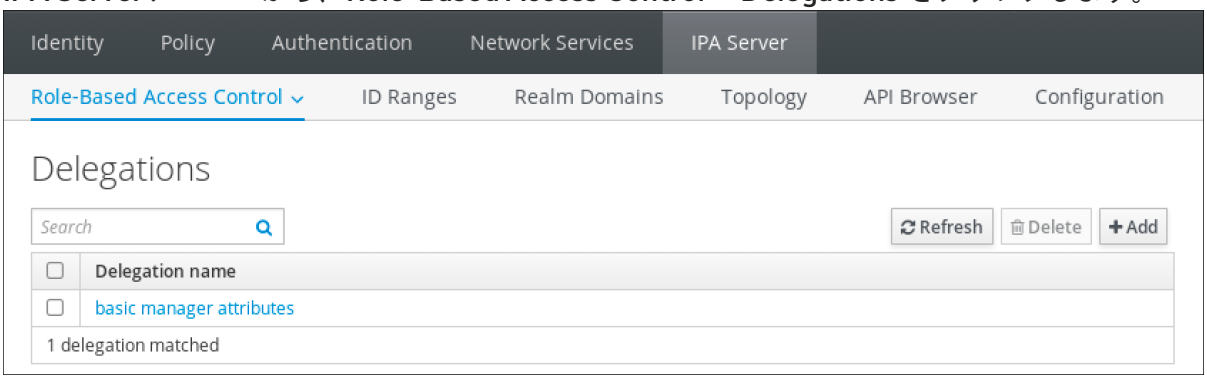
IdM WebUI を使用して既存の委譲ルールを表示するには、次の手順に従います。

前提条件

- **admins** グループのメンバーとして IdM Web UI にログインしている。

手順

- **IPA Server** メニューから、**Role-Based Access Control → Delegations** をクリックします。



25.4. IDM WEBUI を使用した委譲ルールの変更

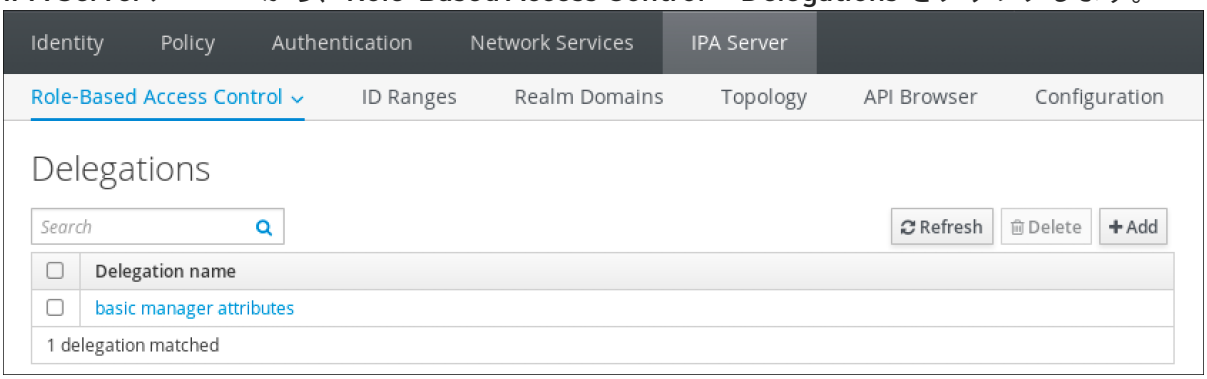
IdM WebUI を使用して既存の委譲ルールを変更するには、次の手順に従います。

前提条件

- **admins** グループのメンバーとして IdM Web UI にログインしている。

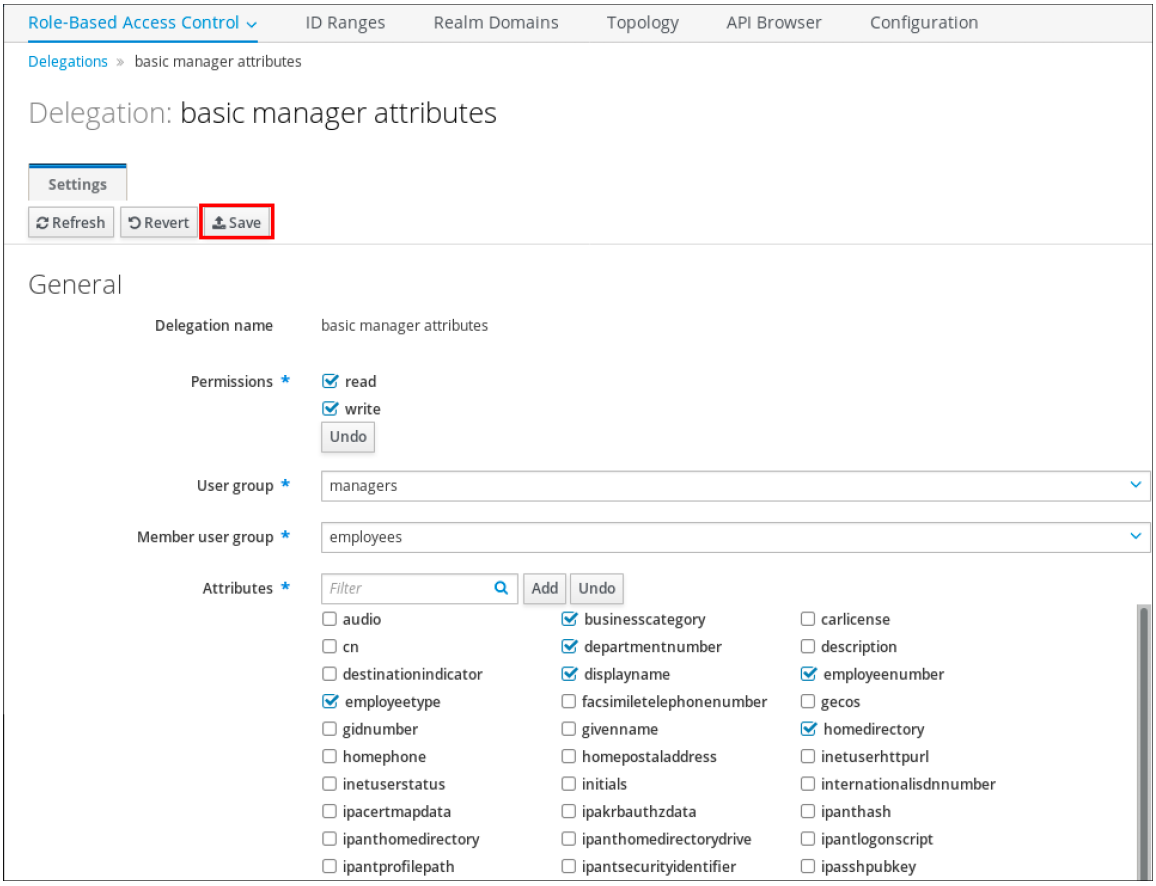
手順

1. **IPA Server** メニューから、**Role-Based Access Control → Delegations** をクリックします。



2. 変更するルールをクリックします。
3. 必要な変更を行います。
   - ルールの名前を変更します。
   - 指定の属性を表示する権限 (**read**)、および指定の属性を追加または変更する権限 (**write**) をユーザーに指定しているかどうかを示すチェックボックスを選択して、付与されたパーミッションを変更します。
   - ユーザーグループのドロップダウンメニューで、**パーミッションを付与している** グループを選択し、メンバーグループのユーザーエントリーを表示または編集します。
   - **Member user group** ドロップダウンメニューで、委譲グループのメンバーが **エントリーを編集できる** グループを選択します。

- 属性ボックスで、パーミッションを付与する属性のチェックボックスを選択します。属性のパーミッションを削除するには、関連するチェックボックスの選択を解除します。



- Save ボタンをクリックして変更を保存します。

25.5. IDM WEBUI を使用した委譲ルールの削除

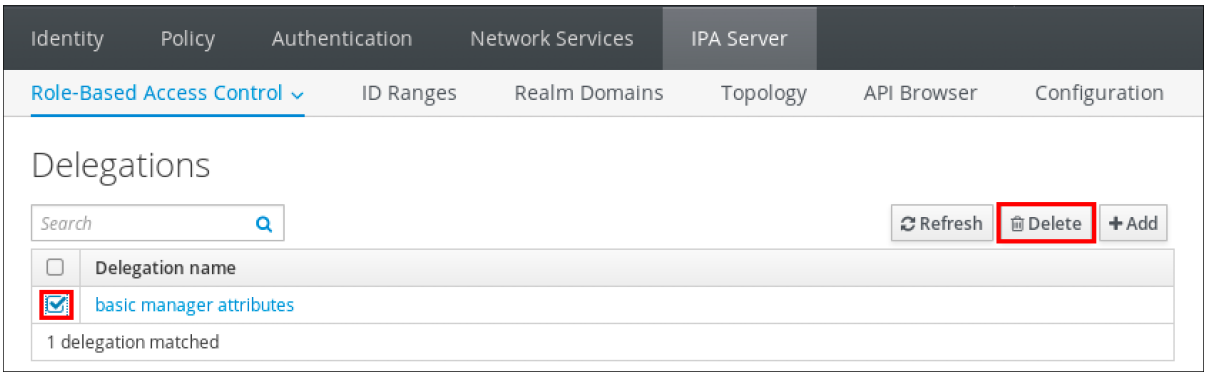
IdM WebUI を使用して既存の委譲ルールを削除するには、次の手順に従います。

前提条件

- **admins** グループのメンバーとして IdM Web UI にログインしている。

手順

1. **IPA Server** メニューから、**Role-Based Access Control → Delegations** をクリックします。
2. 削除するルールの横にあるチェックボックスを選択します。
3. **Delete** をクリックします。



4. **Delete** をクリックして確定します。

第26章 ANSIBLE PLAYBOOK を使用してユーザーグループにパーミッションを委譲してユーザーを管理する手順

委譲は、セルフサービスルールおよびロールベースのアクセス制御 (RBAC) などの IdM のアクセス制御メソッドの 1 つです。委譲を使用して、あるユーザーのグループにパーミッションを割り当てて別のユーザーのグループのエントリーを管理できます。

このセクションでは、以下のトピックについて説明します。

- 委譲ルール
- IdM の Ansible インベントリーファイルの作成
- Ansible を使用して委譲ルールを存在させる手順
- Ansible を使用して委譲ルールがないことを確認する手順
- Ansible を使用して委譲ルールに特定の属性を含める手順
- Ansible を使用して委譲ルールに特定の属性を含めないようにする手順

26.1. 委譲ルール

委譲ルール を作成して、ユーザーグループにパーミッションを委譲してユーザーを管理できます。

委譲ルールを使用すると、特定のユーザーグループが、別のユーザーグループ内のユーザーの特定の属性に対して書き込み (編集) 操作を実行できます。このようなアクセス制御ルールは、委譲ルールで指定された属性のサブセットの編集に限定されており、エントリー全体の追加や削除、未指定の属性の制御はできません。

委譲ルールにより、IdM の既存のユーザーグループにパーミッションが付与されます。委任を使用すると、**managers** ユーザーグループで **employees** ユーザーグループでユーザーの選択された属性を管理できます。

26.2. IDM の ANSIBLE インベントリーファイルの作成

Ansible を使用する場合は、ホームディレクトリーに Ansible Playbook 専用のサブディレクトリーを作成して、**/usr/share/doc/ansible-freeipa/*** と **/usr/share/doc/rhel-system-roles/*** サブディレクトリーからコピーして調整できるようにします。この方法には、以下の利点があります。

- すべての Playbook を 1 カ所で見つけることがでる。
- **root** 権限を呼び出さずに Playbook を実行できる。

手順

1. Ansible 設定および Playbook のディレクトリーをホームディレクトリーに作成します。

```
$ mkdir ~/MyPlaybooks/
```

2. **~/MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks
```

3. **~/MyPlaybooks/ansible.cfg** ファイルを以下の内容で作成します。

```
[defaults]
inventory = /home/<username>/MyPlaybooks/inventory

[privilege_escalation]
become=True
```

4. **~/MyPlaybooks/inventory** ファイルを以下の内容で作成します。

```
[eu]
server.idm.example.com

[us]
replica.idm.example.com

[ipaserver:children]
eu
us
```

この設定は、これらの場所にあるホストの 2 つのホストグループ (**eu** と **us**) を定義します。さらに、この設定は、**eu** および **us** グループのすべてのホストを含む **ipaserver** ホストグループを定義します。

26.3. ANSIBLE を使用して委譲ルールを存在させる手順

以下の手順では、Ansible Playbook を使用して、新しい IdM 委譲ルールの特権を定義して、その存在を確認する方法を説明します。この例では、新しい **basic manager attributes** 委譲ルールにより、**managers** グループが **employees** グループのメンバーに対して以下の属性の読み取りと書き込みを行うことができます。

- **businesscategory**
- **departmentnumber**
- **employeenumber**
- **employeetype**

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。

- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/delegation/** にある **delegation-present.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/delegation/delegation-present.yml
delegation-present-copy.yml
```

3. Ansible Playbook ファイル **delegation-present-copy.yml** を開きます。

4. **ipadelegation** タスクセクションに以下の変数を設定して、ファイルを調整します。

   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は新しい委譲ルールの名前に設定します。
   - **permission** 変数は、付与するパーミッションをコンマ区切りのリスト (**read** および **write**) で設定します。
   - **attribute** 変数は、委譲されたユーザーグループが管理できる属性のリスト (**businesscategory**、**departmentnumber**、**employeenumber** および **employeetype**) に変数を設定します。
   - **group** 変数は、属性の表示や変更権限を付与したグループの名前に設定します。
   - **membergroup** 変数は、属性の表示または変更が可能なグループ名に設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Playbook to manage a delegation rule
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure delegation "basic manager attributes" is present
    ipadelegation:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: "basic manager attributes"
      permission: read, write
      attribute:
      - businesscategory
      - departmentnumber
      - employeenumber
      - employeetype
      group: managers
      membergroup: employees
```

5. ファイルを保存します。

6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i ~/MyPlaybooks/inventory
delegation-present-copy.yml
```

関連情報

- 委譲ルール を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-delegation.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/ipadelegation** ディレクトリーのサンプルの Playbook を参照してください。

26.4. ANSIBLE を使用して委譲ルールがないことを確認する手順

以下の手順では、Ansible Playbook を使用して、指定した委譲ルールが IdM 設定に存在しないことを確認する方法を説明します。以下の例では、カスタムの **basic manager attributes** 委譲ルールが IdM に存在しないことを確認する方法を説明します。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks>/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/delegation/** ディレクトリーにある **delegation-absent.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/delegation/delegation-present.yml
delegation-absent-copy.yml
```

3. Ansible Playbook ファイル **delegation-absent-copy.yml** を開きます。
4. **ipadelegation** タスクセクションに以下の変数を設定して、ファイルを調整します。
   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は委譲ルールの名前に設定します。
   - **state** 変数は **absent** に設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Delegation absent
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure delegation "basic manager attributes" is absent
    ipadelegation:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: "basic manager attributes"
      state: absent
```

5. ファイルを保存します。
6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i ~/MyPlaybooks/inventory
delegation-absent-copy.yml
```

関連情報

- 委譲ルール を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-delegation.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/ipadelegation** ディレクトリーのサンプルの Playbook を参照してください。

26.5. ANSIBLE を使用して委譲ルールに特定の属性を含める手順

以下の手順では、Ansible Playbook を使用して、委譲ルールに特定の設定を指定する方法を説明します。この Playbook を使用して、以前に作成した委譲ロールを変更できます。この例では、**basic manager attributes** 委譲ルールに **departmentnumber** メンバー属性のみが含まれるようにします。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。

- Ansible バージョン 2.14 以降を使用している。
- Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
- この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
- この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。

- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- **basic manager attributes** 委譲ルールが IdM に存在する。

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/delegation/** にある **delegation-member-present.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/delegation/delegation-member-present.yml delegation-member-present-copy.yml
```

3. Ansible Playbook ファイル **delegation-member-present-copy.yml** を開きます。
4. **ipadelegation** タスクセクションに以下の変数を設定して、ファイルを調整します。
   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は、変更する委譲ルールの名前に設定します。
   - **attribute** 変数は **departmentnumber** に設定します。
   - **action** 変数は **member** に設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Delegation member present
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure delegation "basic manager attributes" member attribute departmentnumber
is present
    ipadelegation:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: "basic manager attributes"
      attribute:
      - departmentnumber
      action: member
```

5. ファイルを保存します。
6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i ~/MyPlaybooks/inventory
delegation-member-present-copy.yml
```

関連情報

- 委譲ルール を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-delegation.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/ipadelegation** ディレクトリーのサンプルの Playbook を参照してください。

26.6. ANSIBLE を使用して委譲ルールに特定の属性を含めないようにする手順

以下の手順では、Ansible Playbook を使用して、委譲ルールに特定の設定が割り当てられないようにする方法を説明します。この Playbook を使用して、委譲ロールが不必要なアクセス権限を付与しないようします。この例では、**basic manager attributes** 委譲ルールに **employeenumber** および **employeetype** メンバー属性が含まれないようにします。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- **basic manager attributes** 委譲ルールが IdM に存在する。

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/delegation/** にある **delegation-member-absent.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/delegation/delegation-member-absent.yml delegation-member-absent-copy.yml
```

3. Ansible Playbook ファイル **delegation-member-absent-copy.yml** を開きます。

4. **ipadelegation** タスクセクションに以下の変数を設定して、ファイルを調整します。

   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は、変更する委譲ルールの名前に設定します。
   - **attribute** 変数は **employeenumber** および **employeetype** に設定します。
   - **action** 変数は **member** に設定します。
   - **state** 変数は **absent** に設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Delegation member absent
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure delegation "basic manager attributes" member attributes employeenumber and employeetype are absent
    ipadelegation:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: "basic manager attributes"
      attribute:
      - employeenumber
      - employeetype
      action: member
      state: absent
```

5. ファイルを保存します。

6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i ~/MyPlaybooks/inventory delegation-member-absent-copy.yml
```

関連情報

- 委譲ルール を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-delegation.md** ファイルを参照してください。

- **/usr/share/doc/ansible-freeipa/playbooks/ipadelegation** ディレクトリーのサンプルの Playbook を参照してください。

第27章 CLI で IDM でのロールベースのアクセス制御の管理

Identity Management (IdM) のロールベースのアクセス制御と、コマンドラインインターフェイス (CLI) で実行する次の操作について詳しく説明します。

- パーミッションの管理
- 特権管理
- ロールの管理

27.1. IDM のロールベースのアクセス制御

IdM のロールベースアクセス制御 (RBAC) は、セルフサービス制御および委譲アクセス制御とは非常に異なる種類の権限をユーザーに付与します。

ロールベースのアクセス制御は、以下の 3 つの部分で設定されます。

- **パーミッション** は、ユーザーの追加または削除、グループの変更、読み取りアクセスの有効化など、特定のタスクを実行する権限を付与します。
- **特権** は、新規ユーザーの追加に必要な全権限など、権限を組み合わせます。
- **ロール** は、ユーザー、ユーザーグループ、ホスト、またはホストグループに特権のセットを付与します。

27.1.1. IdM のパーミッション

パーミッションは、ロールベースのアクセス制御の中で最も低いレベルの単位で、操作を適用する LDAP エントリーと合わせて操作を定義します。ブロックの構築と同様に、パーミッションは必要に応じて多くの権限に割り当てることができます。
1 つ以上の **権限** を使用して、許容される操作を定義します。

- **write**
- **read**
- **search**
- **compare**
- **add**
- **delete**
- **all**

上記の操作は、3 つの基本的な **ターゲット** に適用されます。

- **subtree**: ドメイン名 (DN) (この DN のサブツリー)
- **target filter**: LDAP フィルター
- **target**: DN。ワイルドカードでエントリーを指定可能。

また、以下の便利なオプションは、対応する属性を設定します。

- **type**: オブジェクトのタイプ (ユーザー、グループなど) (**subtree** および **target filter** を設定します)。
- **memberof**: グループのメンバー。**target filter** を設定します。
- **targetgroup**: 特定のグループを変更する権限 (グループメンバーシップの管理権限の付与など) を付与します (**target** を設定します)。

IdM パーミッションを使用すると、どのユーザーがどのオブジェクトにアクセスできるか、さらにこのようなオブジェクトの属性にアクセスできるかどうかを制御できます。IdM を使用すると、個別の属性を許可または拒否したり、ユーザー、グループ、sudo などの特定の IdM 機能を、全匿名ユーザー、全認証済みユーザー、または特定の特権ユーザーグループ限定などと、全体的な表示設定を変更したりできます。
たとえば、このアプローチではパーミッション指定に柔軟性があるので、アクセスが必要な特定のセクションのみにユーザーまたはグループのアクセスを制限し、他のセクションをこれらのユーザーまたはグループには完全に表示されないように設定する場合に、管理者にとって便利です。

注記

パーミッションには他のパーミッションを含めることはできません。

27.1.2. デフォルトの管理パーミッション

管理パーミッションは、IdM にデフォルトで含まれているパーミッションです。このパーミッションはユーザーが作成した他のパーミッションと同様に機能しますが、以下の相違点があります。

- この管理パーミッションは削除できず、名前、場所、ターゲットの属性を変更できません。
- このパーミッションには 3 つの属性セットがあります。
  - **デフォルト** の属性。IdM で管理されているため、ユーザーは変更できません。
  - **包含** 属性。ユーザーが別途追加する属性。
  - **除外** 属性。ユーザーが削除する属性。

管理パーミッションは、デフォルトおよび包含属性セットに表示されている属性すべてに適用されますが、除外セットに表示されている属性には適用されません。

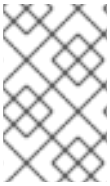
注記

管理パーミッションを削除できませんが、パーミッションにバインドタイプを設定し、すべての特権から管理パーミッションを削除して管理パーミッションを効果的に無効にできます。

管理パーミッションの名前はすべて **System:** から始まります (例: **System: Add Sudo rule** または **System: Modify Services**)。以前のバージョンの IdM では、デフォルトのパーミッションに異なるスキームを使用していました。たとえば、ユーザーはパーミッションの削除はできず、特権に割り当てるしかできませんでした。これらのデフォルトパーミッションのほとんどは、管理パーミッションに切り替わっていますが、以下のパーミッションは引き続き以前のスキームを使用します。

- Automember Rebuild メンバーシップタスクの追加
- 設定サブエントリーの追加
- レプリカ合意の追加

- 証明書削除保留
- CA から証明書のステータス取得
- DNA 範囲の読み取り
- DNA 範囲の変更
- PassSync Manager の設定の読み取り
- PassSync Manager 設定の変更
- レプリカ合意の読み込み
- レプリカ合意の修正
- レプリカ合意の削除
- LDBM データベース設定の読み取り
- 証明書の要求
- CA ACL を無視する証明書の要求
- 別のホストからの証明書の要求
- CA からの証明書の取得
- 証明書の取り消し
- IPA 設定の書き込み



注記

コマンドラインから管理パーミッションを変更しようとし、変更不可な属性の変更をシステム側が許可しない場合には、コマンドに失敗します。Web UI から管理パーミッションを変更しようとした場合には、変更できない属性が無効になります。

27.1.3. IdM の特権

特権は、ロールに適用されるパーミッションのグループです。
パーミッションは単一の操作を実行する権限を提供しますが、IdM タスクを成功させるには、複数のパーミッションが必要なものがあります。したがって、特権は、特定のタスクを実行するために必要な異なるパーミッションを組み合わせたものです。
たとえば、新しい IdM ユーザーにアカウントを設定するには、以下の権限が必要です。

- 新規ユーザーエントリーの作成
- ユーザーパスワードのリセット
- 新規ユーザーのデフォルト IPA ユーザーグループへの追加

これらの 3 つの低レベルのタスクを、**ユーザーの追加** という名前のカスタム特権の形式で、権限がより高いレベルのタスクに組み合わせることで、システム管理者はロールを管理しやすくなります。IdM には、すでにいくつかのデフォルト特権が含まれています。ユーザーとユーザーグループとは別に、権

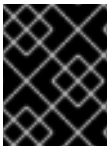
限はホストおよびホストグループ、およびネットワークサービスにも割り当てられます。これにより、特定のネットワークサービスを使用するホストセットのユーザーセットによって、操作をきめ細かく制御できます。

注記

特権には、他の特権を含めることはできません。

27.1.4. IdM のロール

ロールは、ロールに指定したユーザーが所有する権限のリストです。
実際には、パーミッションでは、指定の低階層のタスク (ユーザーエントリーの作成、グループへのエントリーの追加など) を実行する権限を付与し、特権では、高階層のタスク (指定のグループへの新規ユーザーの作成など) に必要なこれらのパーミッションの 1 つ以上を組み合わせます。ロールは必要に応じて、管理者ロールでユーザーの追加、変更、削除ができるなど、特権をまとめます。



重要

ロールは、許可されたアクションを分類するために使用されます。ロールは、特権昇格されないようにしたり、特権の分離を実装するツールとしては使用しません。

注記

ロールに他のロールを含めることはできません。

27.1.5. Identity Management で事前定義されたロール

Red Hat Identity Management には、以下の事前定義済みのロールが含まれています。

表27.1 Identity Management の定義済みロール

ロール	特権	説明
登録管理者	ホストの登録	クライアントまたはホストの登録を行います。
helpdesk	Modify Users、Reset passwords、Modify Group membership	簡単なユーザー管理タスクを実行します。
IT Security Specialist	Netgroups Administrators、HBAC Administrator、Sudo Administrator	ホストベースのアクセス制御、sudo ルールなどのセキュリティーポリシーを管理します。
IT Specialist	Host Administrators、Host Group Administrators、Service Administrators、Automount Administrators	ホストの管理を行います

ロール	特権	説明
Security Architect	Delegation Administrator、Replication Administrators、Write IPA Configuration、Password Policy Administrator	Identity Management 環境の管理、信頼の作成、レプリカ合意を作成します。
User Administrator	User Administrators、Group Administrators、Stage User Administrators	ユーザーおよびグループの作成を行います

27.2. CLI での IDM パーミッションの管理

コマンドラインインターフェイス (CLI) を使用して Identity Management (IdM) のパーミッションを管理するには、次の手順に従います。

前提条件

- IdM、または **ユーザー管理者** ロールを管理する管理者権限
- 有効な Kerberos チケット。詳細は、Using kinit to log in to IdM manually を参照してください。

手順

1. **ipa permission-add** コマンドを使用して、新しいパーミッションエントリーを作成します。たとえば、**dns admin** という名前のパーミッションを追加するには、次のコマンドを実行します。

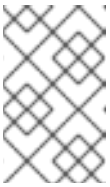
```
$ ipa permission-add "dns admin"
```

2. 以下のオプションでパーミッションのプロパティーを指定します。

- **--bindtype** は、バインドルールの種別を指定します。このオプションでは、**all**、**anonymous** および **permission** 引数を使用できます。**permission** バインドタイプは、ロールを使用してこのパーミッションを付与されたユーザーのみが実行できます。以下に例を示します。

```
$ ipa permission-add "dns admin" --bindtype=all
```

--bindtype を指定しないと、**permission** がデフォルト値になります。



注記

特権には、デフォルト以外のバインドルールタイプが指定されたパーミッションを追加できません。特権に既存のパーミッションは、デフォルト以外のバインドルールタイプには設定できません。

- **--right** は、パーミッションが付与する権限をリスト表示します。これは、非推奨の **--permissions** オプションに代わるものです。使用できる値は **add**、**delete**、**read**、**search**、**compare**、**write**、**all** になります。

複数の属性を設定するには、複数の **--right** オプションを使用するか、中括弧内にコンマ区切りリストを使用します。以下に例を示します。

```
$ ipa permission-add "dns admin" --right=read --right=write
```

```
$ ipa permission-add "dns admin" --right={read,write}
```

注記

add および **delete** はエントリーレベルの操作 (ユーザーの削除、グループの追加など) ですが、**read**、**search**、**compare**、**write** は、属性レベル (**userCertificate** への書き込み可、**userPassword** の読み込み不可) に近いです。

- **--attrs** は、パーミッションが付与される属性のリストを提供します。
複数の属性を設定するには、複数の **--attrs** オプションを使用するか、オプションを中括弧内にコンマ区切りリストでリスト表示します。以下に例を示します。

```
$ ipa permission-add "dns admin" --attrs=description --attrs=automountKey
```

```
$ ipa permission-add "dns admin" --attrs={description,automountKey}
```

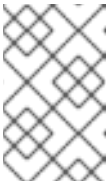
--attrs で指定の属性が存在し、指定のオブジェクトタイプで使用可能な属性である必要があります。この条件を満たさない場合には、コマンドがスキーマ構文エラーで失敗します。

- **--type** は、ユーザー、ホスト、サービスなどのパーミッションが適用されるエントリーオブジェクトタイプを定義します。各タイプには、独自の許可属性セットがあります。
以下に例を示します。

```
$ ipa permission-add "manage service" --right=all --type=service --attrs=krbprincipalkey --attrs=krbprincipalname --attrs=managedby
```

- **--subtree** ではサブツリーエントリーを指定します。フィルターはこのサブツリーエントリーの配下にあるエントリーを対象とします。既存のサブツリーエントリーを指定します。**--subtree** ではワイルドカードや存在しないドメイン名 (DN) は使用できません。ディレクトリーに DN を追加します。
IdM は簡素化されたフラットディレクトリーツリー構造を使用しているため、**--subtree** を使用して自動マウントの場所 (他の設定のコンテナーまたは親エントリー) など、一部のエントリーを対象にできます。以下に例を示します。

```
$ ipa permission-add "manage automount locations" --subtree="ldap://ldap.example.com:389/cn=automount,dc=example,dc=com" --right=write --attrs=automountmapname --attrs=automountkey --attrs=automountInformation
```



注記

--type オプションおよび **--subtree** オプションを同時に使用できません。**--subtree** を簡素化したものとして、**--type** のフィルターに含まれている内容を確認できます (これにより、管理者の作業が簡単になります)。

- **--filter** は LDAP フィルターを使用して、パーミッションが適用されるエントリーを特定し

ます。
IdM は、指定のフィルターの有効性を自動的に確認します。このフィルターには、有効な LDAP フィルターを使用できます。以下に例を示します。

```
$ ipa permission-add "manage Windows groups" --filter="(!(objectclass=posixgroup))" --right=write --attrs=description
```

- **--memberof** は、グループが存在することを確認した後に、指定したグループのメンバーにターゲットフィルターを設定します。たとえば、このパーミッショを持つユーザーがエンジニアリンググループのメンバーのログインシェルを変更できるようにするには、以下を実行します。

```
$ ipa permission-add ManageShell --right="write" --type=user --attr=loginshell --memberof=engineers
```

- **--targetgroup** は、グループが存在することを確認した後に、ターゲットを指定のユーザーグループに設定します。たとえば、このパーミッションを持つグループが、エンジニアリンググループのメンバー属性に (メンバーの追加や削除ができるように) 書き込みできるようにするには、以下を実行します。

```
$ ipa permission-add ManageMembers --right="write" --subtree=cn=groups,cn=accounts,dc=example,dc=test --attr=member --targetgroup=engineers
```

- 必要に応じて、ターゲットドメイン名 (DN) を指定できます。
  - **--target** は、パーミッションを適用する DN を指定します。ワイルドカードは使用できます。
  - **--targetto** は、エントリーの移動先に設定できる DN サブツリーを指定します。
  - **--targetfrom** は、エントリーの移動元に設定できる DN サブツリーを指定します。

27.3. 既存のパーミッションのコマンドオプション

必要に応じて、既存のパーミッションを変更するには、以下のバリアントを使用します。

- 既存のパーミッションを編集するには、**ipa permission-mod** コマンドを使用します。パーミッションの追加と同じコマンドオプションを使用できます。
- 既存のパーミッションを見つけるには、**ipa permission-find** コマンドを使用します。パーミッションの追加と同じコマンドオプションを使用できます。
- 特定のパーミッションを表示するには、**ipa permission-show** コマンドを使用します。**--raw** 引数は、生成される未編集の 389-ds ACI を表示します。以下に例を示します。

```
$ ipa permission-show <permission> --raw
```

- **ipa permission-del** コマンドは、パーミッションを完全に削除します。

関連情報

- **ipa** の man ページを参照してください。

- **ipa help** コマンドを参照してください。

27.4. CLI での IDM 権限の管理

コマンドラインインターフェイス (CLI) を使用して Identity Management (IdM) の特権を管理するには、次の手順に従います。

前提条件

- IdM、または **ユーザー管理者** ロールを管理する管理者権限
- 有効な Kerberos チケット。詳細は、Using kinit to log in to IdM manually を参照してください。
- 既存のパーミッション。パーミッションの詳細は、CLI での IdM パーミッションの管理 を参照してください。

手順

1. **ipa privilege-add** コマンドを使用して、特権エントリーを追加します。
たとえば、名前が **managing filesystems** の特権を説明を付けて追加するには、次のコマンドを実行します。

```
$ ipa privilege-add "managing filesystems" --desc="for filesystems"
```

2. **privilege-add-permission** コマンドを使用して必要なパーミッションを特権グループに割り当てます。
たとえば、パーミッション **managing automount** および **managing ftp services** を、**managing filesystems** 権限に追加するには、次のコマンドを実行します。

```
$ ipa privilege-add-permission "managing filesystems" --permissions="managing automount"
--permissions="managing ftp services"
```

27.5. 既存の特権のコマンドオプション

必要に応じて、既存の特権を変更するには、以下のバリアントを使用します。

- 既存の特権を変更するには、**ipa privilege-mod** コマンドを使用します。
- 既存の特権を見つけるには、**ipa privilege-find** コマンドを使用します。
- 特定の特権を表示するには、**ipa privilege-show** コマンドを使用します。
- **ipa privilege-remove-permission** コマンドは、特権から 1 つ以上のパーミッションを削除します。
- **ipa privilege-del** コマンドは、特権を完全に削除します。

関連情報

- **ipa** の man ページを参照してください。
- **ipa help** コマンドを参照してください。

27.6. CLI での IDM ロールの管理

コマンドラインインターフェイス (CLI) を使用して Identity Management (IdM) のロールを管理するには、次の手順に従います。

前提条件

- IdM、または **ユーザー管理者** ロールを管理する管理者権限
- 有効な Kerberos チケット。詳細は、Using kinit to log in to IdM manually を参照してください。
- 既存の特権。特権の詳細は、CLI での IdM 特権の管理 を参照してください。

手順

1. **ipa role-add** コマンドを使用して、新規ロールエントリーを追加します。

```
$ ipa role-add --desc="User Administrator" useradmin
-----------------------
Added role "useradmin"
-----------------------
Role name: useradmin
Description: User Administrator
```

2. **ipa role-add-privilege** コマンドを使用して、必要な特権をロールに追加します。

```
$ ipa role-add-privilege --privileges="user administrators" useradmin
Role name: useradmin
Description: User Administrator
Privileges: user administrators
----------------------------
Number of privileges added 1
----------------------------
```

3. **ipa role-add-member** コマンドを使用して、必要なメンバーをロールに追加します。使用可能なメンバータイプ: ユーザー、グループ、ホスト、およびホストグループ。
たとえば、**useradmins** という名前のグループを、以前に作成した **useradmin** ロールに追加するには、次のコマンドを実行します。

```
$ ipa role-add-member --groups=useradmins useradmin
Role name: useradmin
Description: User Administrator
Member groups: useradmins
Privileges: user administrators
------------------------
Number of members added 1
------------------------
```

27.7. 既存ロールのコマンドオプション

必要に応じて、既存のロールを変更するには、以下のバリアントを使用します。

- 既存のロールを変更するには、**ipa role-mod** コマンドを使用します。

- 既存のロールを見つけるには、**ipa role-find** コマンドを使用します。
- 特定のロールを表示するには、**ipa role-show** コマンドを使用します。
- ロールからメンバーを削除するには、**ipa role-remove-member** コマンドを使用します。
- **ipa role-remove-privilege** コマンドは、ロールから 1 つ以上の権限を削除します。
- **ipa role-del** コマンドは、ロールを完全に削除します。

関連情報

- **ipa** man ページを参照してください。
- **ipa help** コマンドを参照してください。

第28章 IDM WEB UI を使用したロールベースのアクセス制御の管理

Identity Management (IdM) のロールベースのアクセス制御と、Web インターフェイス (Web UI) で実行する次の操作について詳しく説明します。

- パーミッションの管理
- 特権管理
- ロールの管理

28.1. IDM のロールベースのアクセス制御

IdM のロールベースアクセス制御 (RBAC) は、セルフサービス制御および委譲アクセス制御とは非常に異なる種類の権限をユーザーに付与します。

ロールベースのアクセス制御は、以下の 3 つの部分で設定されます。

- **パーミッション** は、ユーザーの追加または削除、グループの変更、読み取りアクセスの有効化など、特定のタスクを実行する権限を付与します。
- **特権** は、新規ユーザーの追加に必要な全権限など、権限を組み合わせます。
- **ロール** は、ユーザー、ユーザーグループ、ホスト、またはホストグループに特権のセットを付与します。

28.1.1. IdM のパーミッション

パーミッションは、ロールベースのアクセス制御の中で最も低いレベルの単位で、操作を適用する LDAP エントリーと合わせて操作を定義します。ブロックの構築と同様に、パーミッションは必要に応じて多くの権限に割り当てることができます。
1 つ以上の **権限** を使用して、許容される操作を定義します。

- **write**
- **read**
- **search**
- **compare**
- **add**
- **delete**
- **all**

上記の操作は、3 つの基本的な **ターゲット** に適用されます。

- **subtree**: ドメイン名 (DN) (この DN のサブツリー)
- **target filter**: LDAP フィルター
- **target**: DN。ワイルドカードでエントリーを指定可能。

また、以下の便利なオプションは、対応する属性を設定します。

- **type**: オブジェクトのタイプ (ユーザー、グループなど) (**subtree** および **target filter** を設定します)。
- **memberof**: グループのメンバー。**target filter** を設定します。
- **targetgroup**: 特定のグループを変更する権限 (グループメンバーシップの管理権限の付与など) を付与します (**target** を設定します)。

IdM パーミッションを使用すると、どのユーザーがどのオブジェクトにアクセスできるか、さらにこのようなオブジェクトの属性にアクセスできるかどうかを制御できます。IdM を使用すると、個別の属性を許可または拒否したり、ユーザー、グループ、sudo などの特定の IdM 機能を、全匿名ユーザー、全認証済みユーザー、または特定の特権ユーザーグループ限定などと、全体的な表示設定を変更したりできます。
たとえば、このアプローチではパーミッション指定に柔軟性があるので、アクセスが必要な特定のセクションのみにユーザーまたはグループのアクセスを制限し、他のセクションをこれらのユーザーまたはグループには完全に表示されないように設定する場合に、管理者にとって便利です。

注記

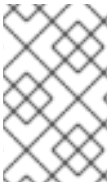
パーミッションには他のパーミッションを含めることはできません。

28.1.2. デフォルトの管理パーミッション

管理パーミッションは、IdM にデフォルトで含まれているパーミッションです。このパーミッションはユーザーが作成した他のパーミッションと同様に機能しますが、以下の相違点があります。

- この管理パーミッションは削除できず、名前、場所、ターゲットの属性を変更できません。
- このパーミッションには 3 つの属性セットがあります。
  - **デフォルト** の属性。IdM で管理されているため、ユーザーは変更できません。
  - **包含** 属性。ユーザーが別途追加する属性。
  - **除外** 属性。ユーザーが削除する属性。

管理パーミッションは、デフォルトおよび包含属性セットに表示されている属性すべてに適用されますが、除外セットに表示されている属性には適用されません。



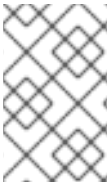
注記

管理パーミッションを削除できませんが、パーミッションにバインドタイプを設定し、すべての特権から管理パーミッションを削除して管理パーミッションを効果的に無効にできます。

管理パーミッションの名前はすべて **System:** から始まります (例: **System: Add Sudo rule** または **System: Modify Services**)。以前のバージョンの IdM では、デフォルトのパーミッションに異なるスキームを使用していました。たとえば、ユーザーはパーミッションの削除はできず、特権に割り当てるしかできませんでした。これらのデフォルトパーミッションのほとんどは、管理パーミッションに切り替わっていますが、以下のパーミッションは引き続き以前のスキームを使用します。

- Automember Rebuild メンバーシップタスクの追加
- 設定サブエントリーの追加

- レプリカ合意の追加
- 証明書削除保留
- CA から証明書のステータス取得
- DNA 範囲の読み取り
- DNA 範囲の変更
- PassSync Manager の設定の読み取り
- PassSync Manager 設定の変更
- レプリカ合意の読み込み
- レプリカ合意の修正
- レプリカ合意の削除
- LDBM データベース設定の読み取り
- 証明書の要求
- CA ACL を無視する証明書の要求
- 別のホストからの証明書の要求
- CA からの証明書の取得
- 証明書の取り消し
- IPA 設定の書き込み



注記

コマンドラインから管理パーミッションを変更しようとし、変更不可な属性の変更をシステム側が許可しない場合には、コマンドに失敗します。Web UI から管理パーミッションを変更しようとした場合には、変更できない属性が無効になります。

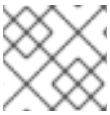
28.1.3. IdM の特権

特権は、ロールに適用されるパーミッションのグループです。
パーミッションは単一の操作を実行する権限を提供しますが、IdM タスクを成功させるには、複数のパーミッションが必要なものがあります。したがって、特権は、特定のタスクを実行するために必要な異なるパーミッションを組み合わせたものです。
たとえば、新しい IdM ユーザーにアカウントを設定するには、以下の権限が必要です。

- 新規ユーザーエントリーの作成
- ユーザーパスワードのリセット
- 新規ユーザーのデフォルト IPA ユーザーグループへの追加

これらの 3 つの低レベルのタスクを、**ユーザーの追加** という名前のカスタム特権の形式で、権限がより高いレベルのタスクに組み合わせることで、システム管理者はロールを管理しやすくなります。IdM には、すでにいくつかのデフォルト特権が含まれています。ユーザーとユーザーグループとは別に、権

限はホストおよびホストグループ、およびネットワークサービスにも割り当てられます。これにより、特定のネットワークサービスを使用するホストセットのユーザーセットによって、操作をきめ細かく制御できます。



注記

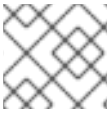
特権には、他の特権を含めることはできません。

28.1.4. IdM のロール

ロールは、ロールに指定したユーザーが所有する権限のリストです。
実際には、パーミッションでは、指定の低階層のタスク (ユーザーエントリーの作成、グループへのエントリーの追加など) を実行する権限を付与し、特権では、高階層のタスク (指定のグループへの新規ユーザーの作成など) に必要なこれらのパーミッションの 1 つ以上を組み合わせます。ロールは必要に応じて、管理者ロールでユーザーの追加、変更、削除ができるなど、特権をまとめます。

重要

ロールは、許可されたアクションを分類するために使用されます。ロールは、特権昇格されないようにしたり、特権の分離を実装するツールとしては使用しません。



注記

ロールに他のロールを含めることはできません。

28.1.5. Identity Management で事前定義されたロール

Red Hat Identity Management には、以下の事前定義済みのロールが含まれています。

表28.1 Identity Management の定義済みロール

ロール	特権	説明
登録管理者	ホストの登録	クライアントまたはホストの登録を行います。
helpdesk	Modify Users、Reset passwords、Modify Group membership	簡単なユーザー管理タスクを実行します。
IT Security Specialist	Netgroups Administrators、HBAC Administrator、Sudo Administrator	ホストベースのアクセス制御、sudo ルールなどのセキュリティーポリシーを管理します。
IT Specialist	Host Administrators、Host Group Administrators、Service Administrators、Automount Administrators	ホストの管理を行います

ロール	特権	説明
Security Architect	Delegation Administrator、Replication Administrators、Write IPA Configuration、Password Policy Administrator	Identity Management 環境の管理、信頼の作成、レプリカ合意を作成します。
User Administrator	User Administrators、Group Administrators、Stage User Administrators	ユーザーおよびグループの作成を行います

28.2. IDM WEB UI でのパーミッションの管理

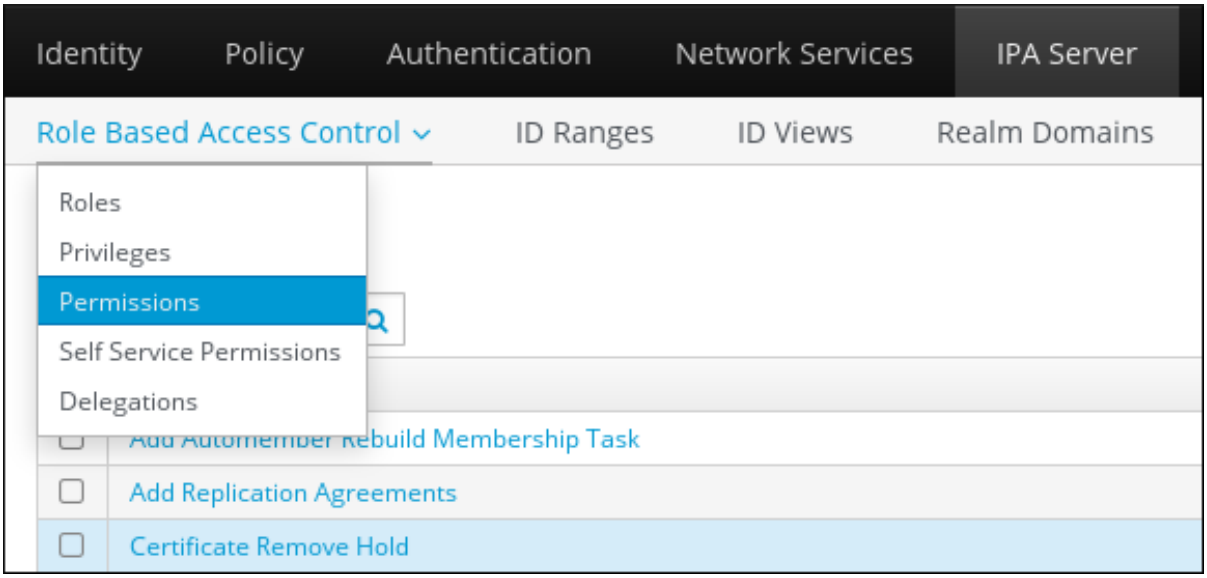
Web インターフェイス (IdM Web UI) を使用して Identity Management (IdM) のパーミッションを管理するには、次の手順に従います。

前提条件

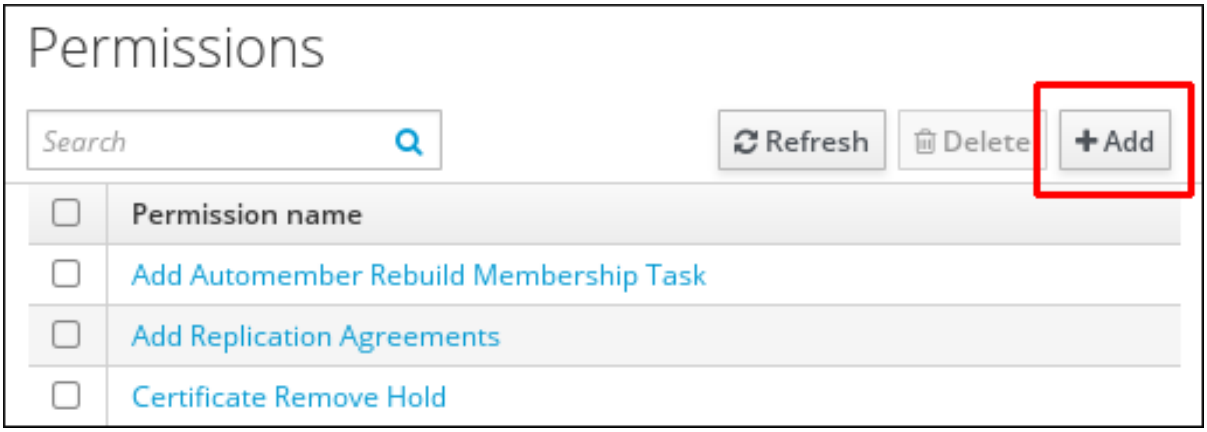
- IdM、または **ユーザー管理者** ロールを管理する管理者権限
- IdM Web UI にログインしている。詳細は Web ブラウザーで IdM Web UI へのアクセス を参照してください。

手順

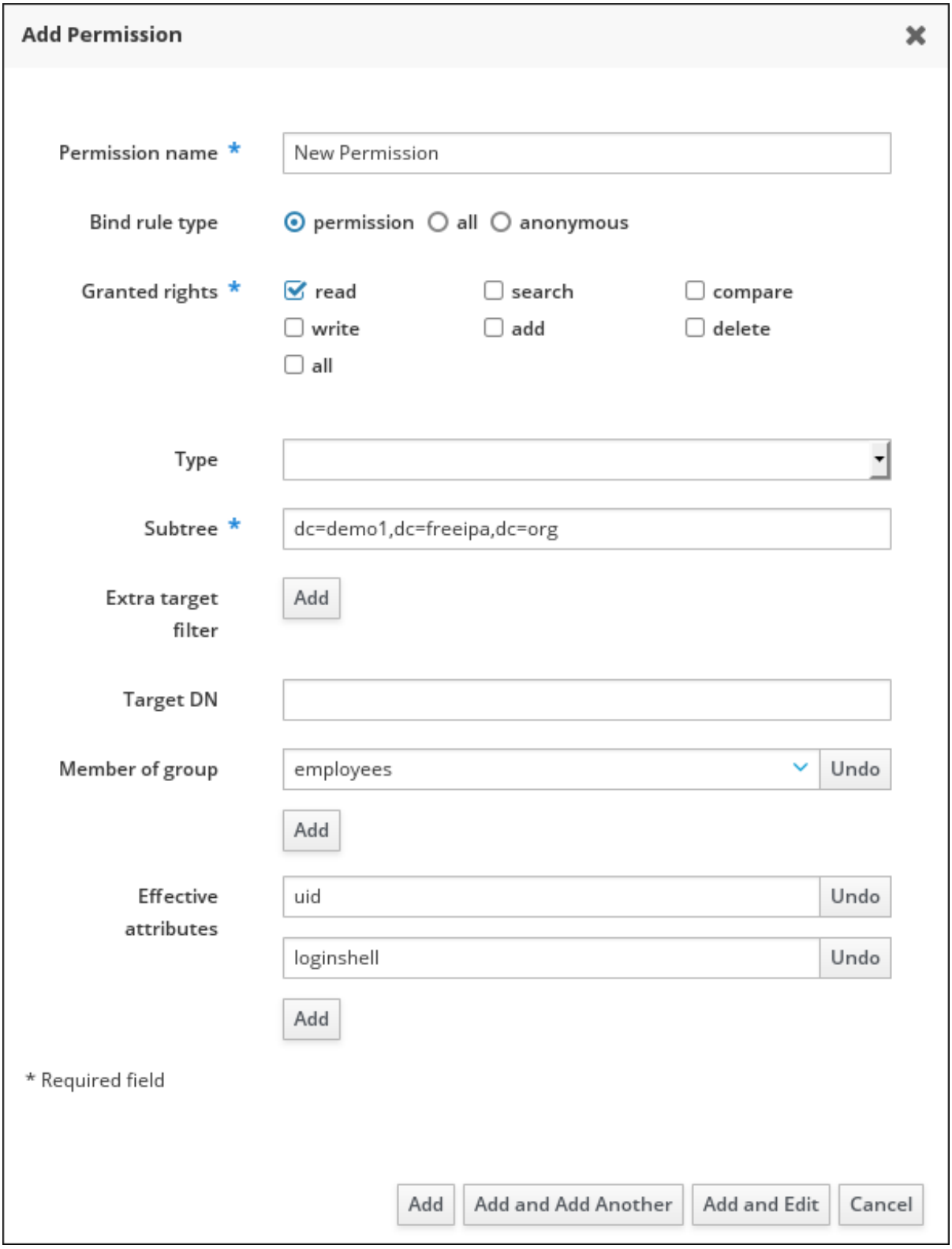
1. 新しいパーミッションを追加するには、**IPA Server** タブで **Role-Based Access Control** サブメニューを開き、**Permissions** を選択します。



2. パーミッションのリストが開きます。パーミッションのリストの上部にある **Add** ボタンをクリックします。

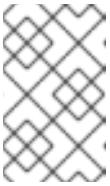


3. **Add Permission** フォームが開きます。新しいパーミッションの名前を指定し、そのプロパティーを適宜定義します。



4. 適切なバインドルールタイプを選択します。

   - **permission** はデフォルトのパーミッションタイプで、権限およびロール経由でアクセスを付与します。
   - **all**: パーミッションを全認証ユーザーに適用することを指定します。
   - **anonymous**: 認証されていないユーザーを含め、すべてのユーザーにパーミッションを適用することを指定します。



注記

特権には、デフォルト以外のバインドルールタイプが指定されたパーミッションを追加できません。特権に既存のパーミッションは、デフォルト以外のバインドルールタイプには設定できません。

5. **Granted rights** でこのパーミッションを付与する権限を選択します。

6. パーミッションのターゲットエントリーを識別する方法を定義します。

   - **Type**: ユーザー、ホスト、またはサービスなどのエントリータイプを指定します。**Type** 設定の値を選択すると、このエントリータイプの ACI でアクセス可能な対応の属性をすべて **Effective Attributes** に表示します。**Type** を定義すると、 **Subtree** および **Target DN** が事前定義された値のいずれかに設定されます。

   - **Subtree** (必須): サブツリーエントリーを指定します。このサブツリーエントリーの下にあるすべてのエントリーが対象になります。**Subtree** ではワイルドカードや存在しないドメイン名 (DN) を使用できないので、既存のサブツリーエントリーを指定します。例: **cn=automount,dc=example,dc=com**

   - **Extra target filter**: DAP フィルターを使用して、パーミッションが適用されるエントリーを特定します。フィルターには、任意の有効な LDAP フィルターを使用できます (たとえば、**(!(objectclass=posixgroup))**
) IdM は指定のフィルターの有効性を自動的にチェックします。無効なフィルターを入力して、パーミッションを保存しようとすると、IdM からこの件について警告が表示されます。

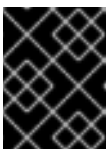
   - **Target DN**: ドメイン名 (DN) を指定し、ワイルドカードを受け入れます。例: **uid=*,cn=users,cn=accounts,dc=com**

   - **Member of group**: 指定したグループのメンバーにターゲットフィルターを設定します。フィルター設定を指定してから **Add** をクリックすると、IdM がフィルターを検証します。すべてのパーミッション設定が正しい場合は、IdM により検索が実行されます。パーミッション設定の一部が正しくない場合には、IdM により、どの設定が正しく設定されているかを示すメッセージが表示されます。

7. パーミッションに属性を追加します。

   - **Type** を設定する場合は、利用可能な ACI 属性のリストから **Effective attributes** を選択します。

   - **Type** を使用しない場合は、**Effective attributes** フィールドに属性を手動で書き込みます。一度に 1 つの属性を追加します。複数の属性を追加するには、**Add** をクリックして別の入力フィールドを追加します。



重要

パーミッションの属性を設定しない場合には、パーミッションはデフォルトですべての属性が含まれます。

8. フォーム下部の **Add** ボタンでパーミッションの追加を完了します。

   - **Add** ボタンをクリックしてパーミッションを保存し、パーミッションのリストに戻ります。

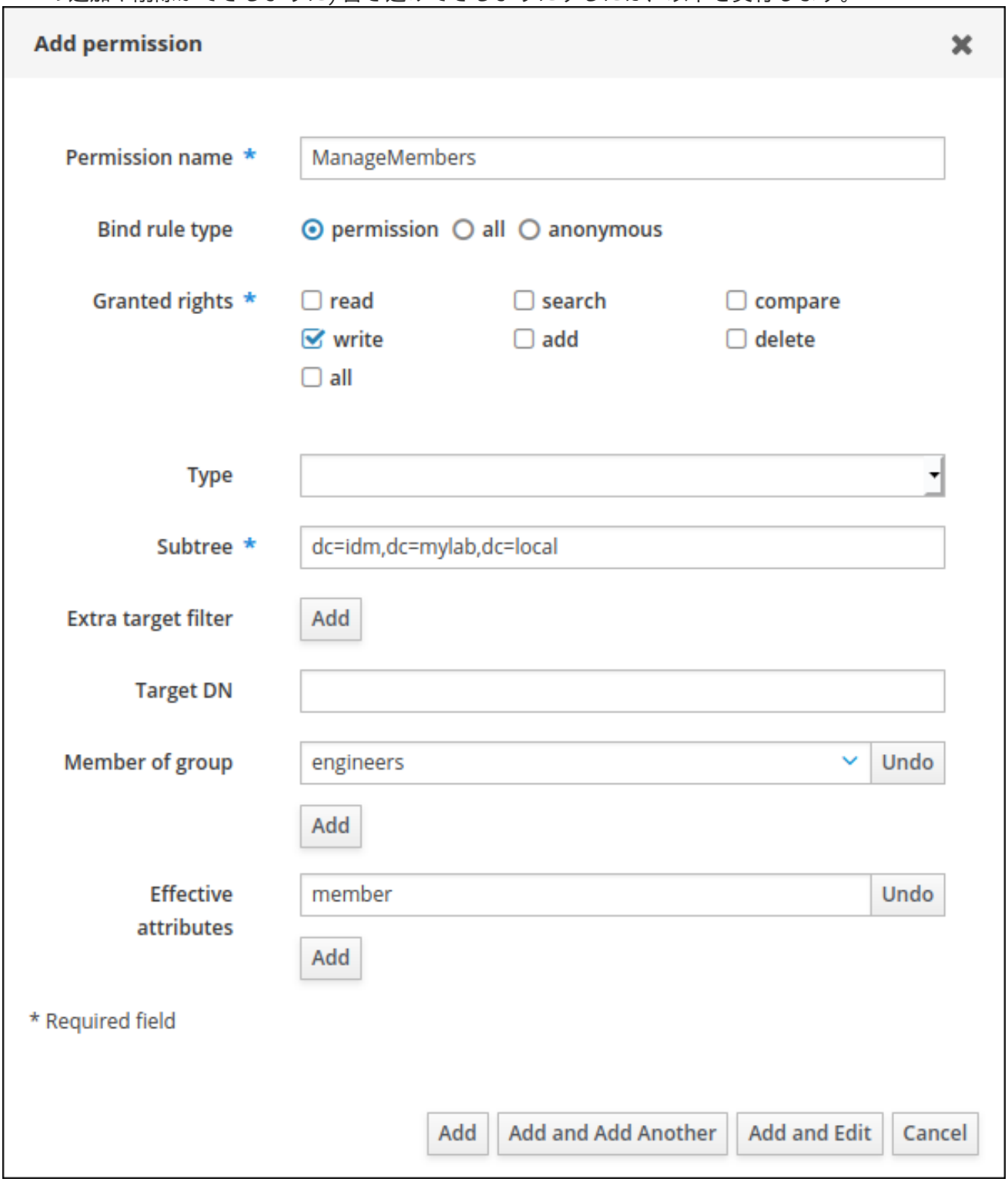
- パーミッションを保存し、**Add and Add another** ボタンをクリックして、継続して同じフォームに別のパーミッションを追加できます。
- **Add and Edit** ボタンを使用すると、新規作成したパーミッションを保存して編集を継続できます。

9. **オプション。**また、パーミッションのリストから名前をクリックして **パーミッション権限** ページを表示し、既存のパーミッションのプロパティーを編集することもできます。
10. **オプション。**既存のパーミッションを削除する必要がある場合は、リストで名前の横にあるチェックボックスにチェックマークを入れて、**Delete** ボタンをクリックして、**Remove permissions** ダイアログを表示します。

注記

デフォルトの管理パーミッションに対する操作は、変更できない属性は IdM Web UI で無効になっており、管理パーミッションを完全に削除することができないんど、制限されています。
ただし、すべての特権から管理されているパーミッションを削除して、パーミッションにバインドタイプが設定されている管理されているパーミッションを効果的に無効にすることができます。

たとえば、このパーミッションを持つグループが、エンジニアリンググループのメンバー属性に (メンバーの追加や削除ができるように) 書き込みできるようにするには、以下を実行します。



28.3. IDM WEBUI での特権の管理

Web インターフェイス (IdM Web UI) を使用して IdM の特権を管理するには、次の手順に従います。

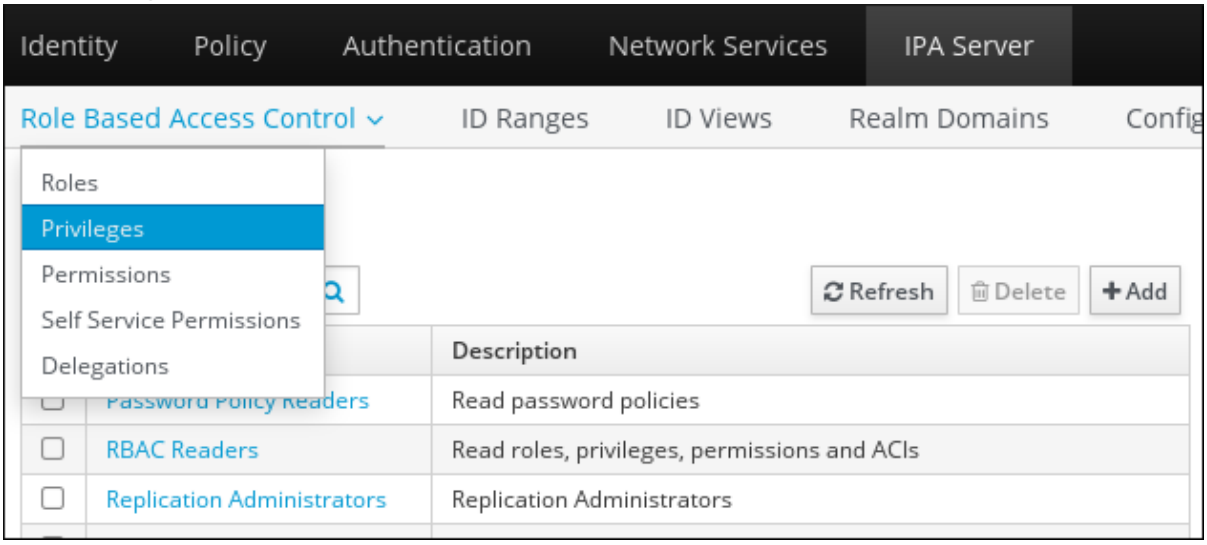
前提条件

- IdM、または **ユーザー管理者** ロールを管理する管理者権限
- IdM Web UI にログインしている。詳細は Web ブラウザーで IdM Web UI へのアクセス を参照してください。

- 既存のパーミッション。パーミッションの詳細は、IdM Web UI でのパーミッションの管理 を参照してください。

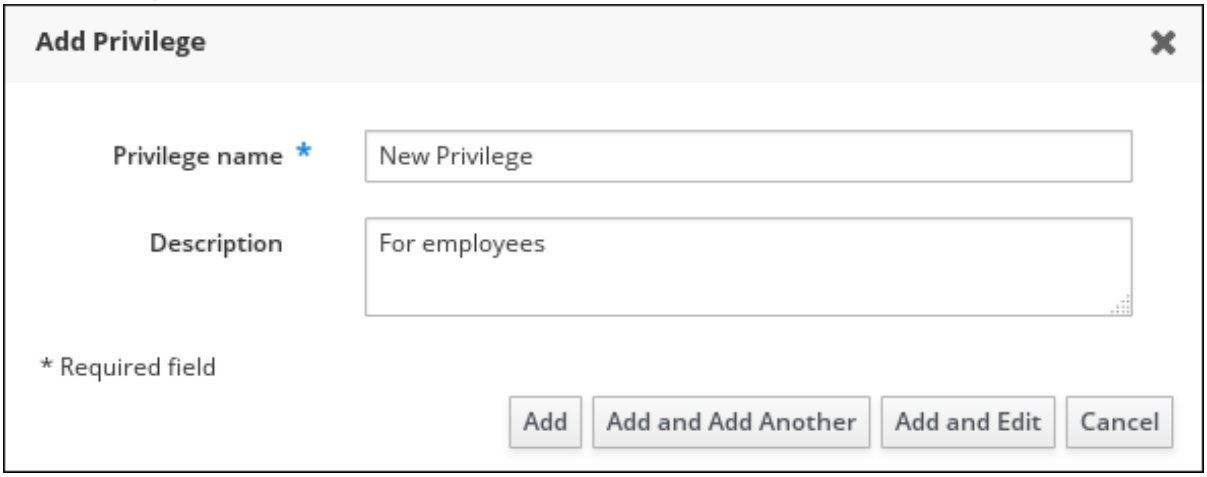
手順

1. 新しい特権を追加するには、**IPA Server** タブで **Role-Based Access Control** サブメニューを開き、**Privileges** を選択します。



2. 特権のリストが開きます。特権リストの上部にある **Add** ボタンをクリックします。

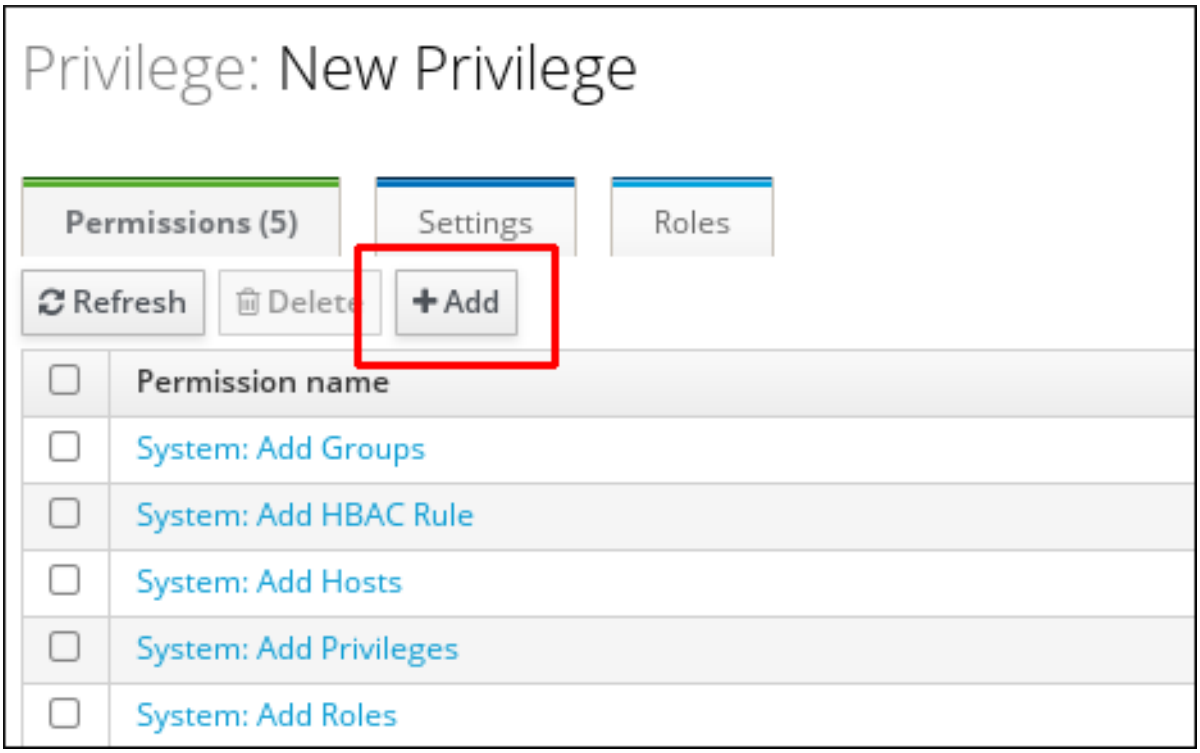
3. **Add Privilege** フォームが開きます。特権の名前と説明を入力します。



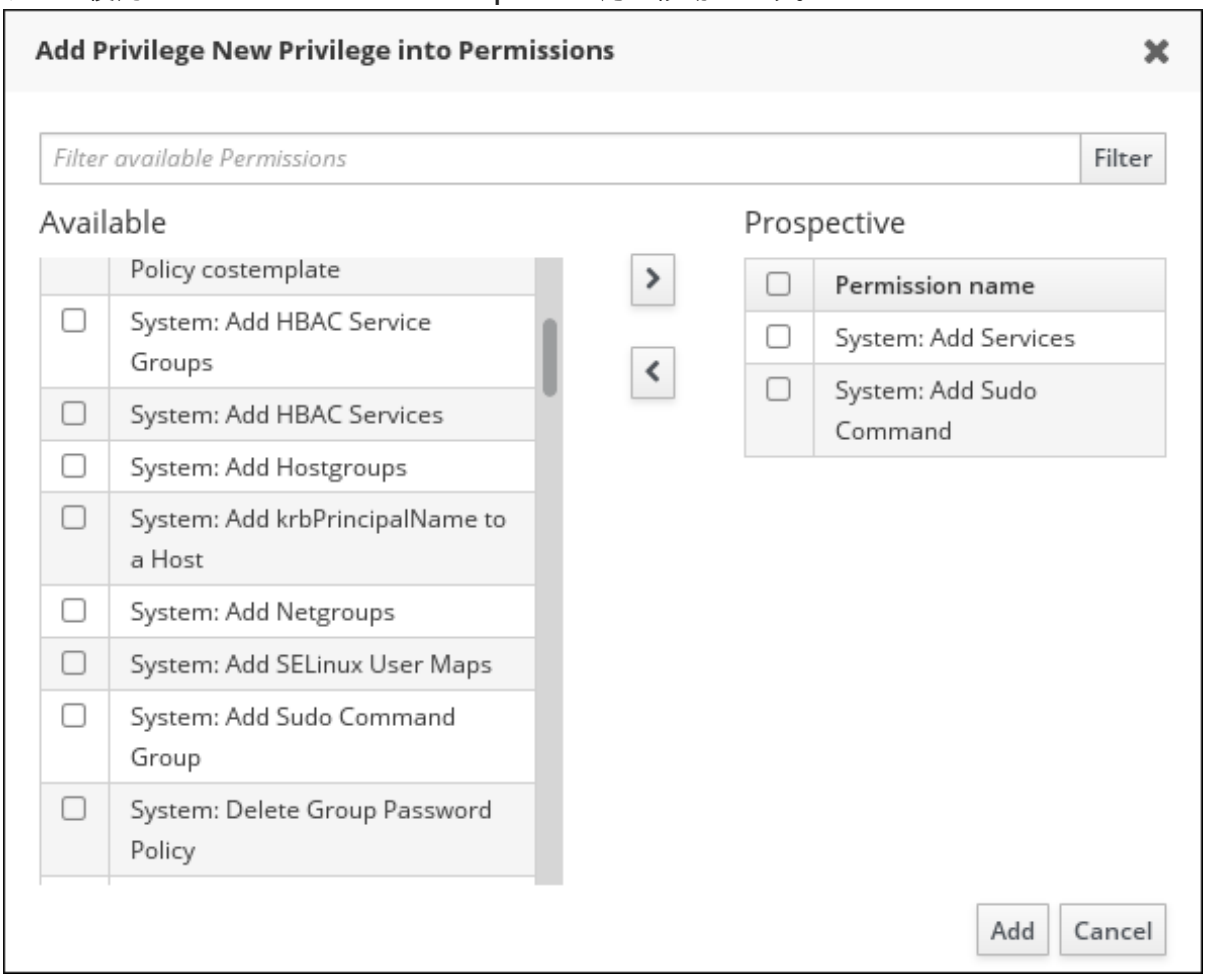
4. 新しい特権を保存し、特権設定ページに移動し、パーミッションを追加するには、**Add and Edit** ボタンをクリックします。

5. 特権リストから特権名をクリックして、特権のプロパティーを編集します。特権設定ページが開きます。

6. **Permissions** タブには、選択した特権に含まれるパーミッションのリストが表示されます。リスト上部の **Add** ボタンをクリックして、パーミッションを特権に追加します。



7. 追加する各パーミッションの名前の横にあるチェックボックスにチェックマークを入れ、> ボタンを使用してパーミッションを **Prospective** 列に移動します。



8. **Add** ボタンをクリックして確定します。

9. **オプション。**パーミッションを削除する必要がある場合は、関連するパーミッションの横にあるチェックボックス (**Remove privileges from permissions**) を表示してから、**Delete** ボタンをクリックします。

10. **オプション。**既存の特権を削除する必要がある場合は、リストで名前の横にあるチェックボックスにチェックを入れて、**Delete** ボタンをクリックすると、**Remove privileges** ダイアログが開きます。

28.4. IDM WEB UI でのロールの管理

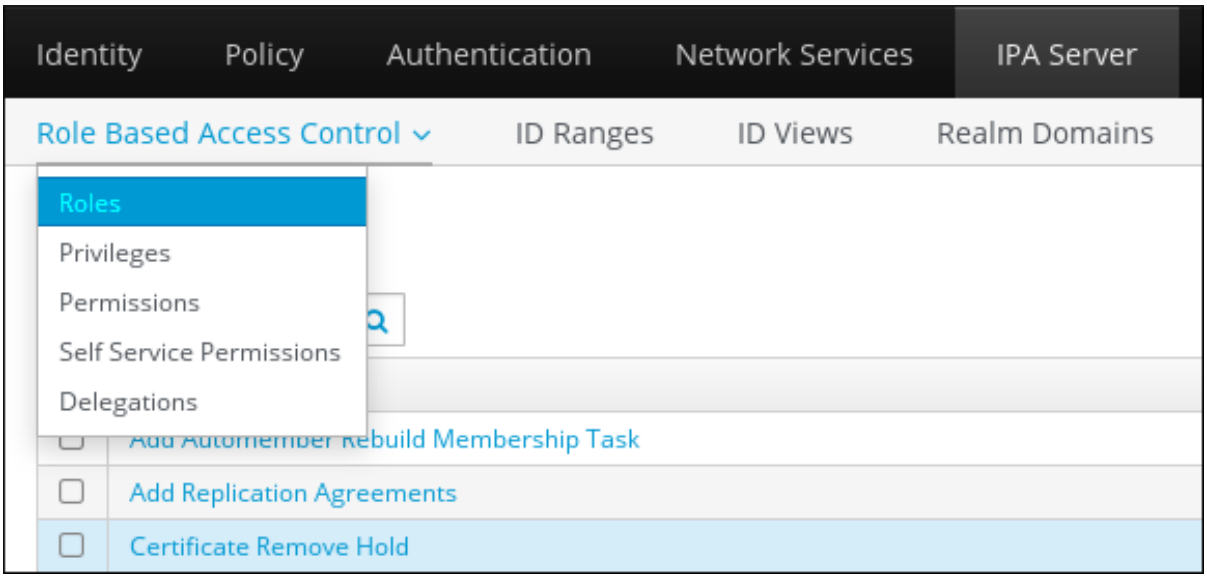
Web インターフェイス (IdM Web UI) を使用して Identity Management (IdM) のロールを管理するには、次の手順に従います。

前提条件

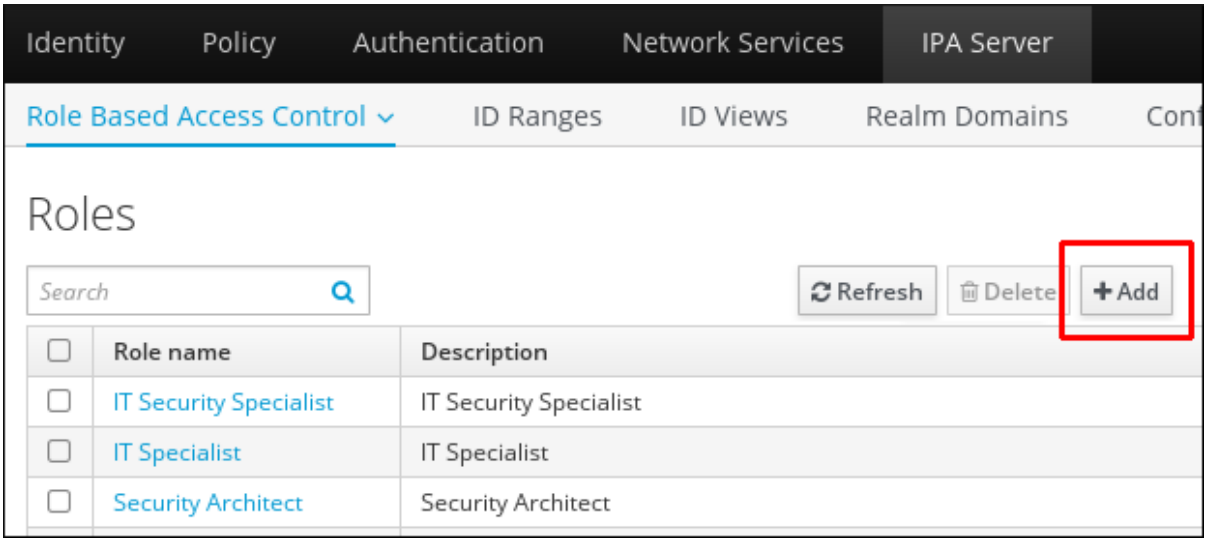
- IdM、または **ユーザー管理者** ロールを管理する管理者権限
- IdM Web UI にログインしている。詳細は Web ブラウザーで IdM Web UI へのアクセス を参照してください。
- 既存の特権。権限の詳細は、IdM Web UI で権限の管理 を参照してください。

手順

1. 新規ロールを追加するには、**IPA Server** タブの **Role-Based Access Control** サブメニューを開き、**Roles** を選択します。



2. ロールのリストが開きます。ロールベースのアクセス制御手順のリストの上部にある **Add** ボタンをクリックします。

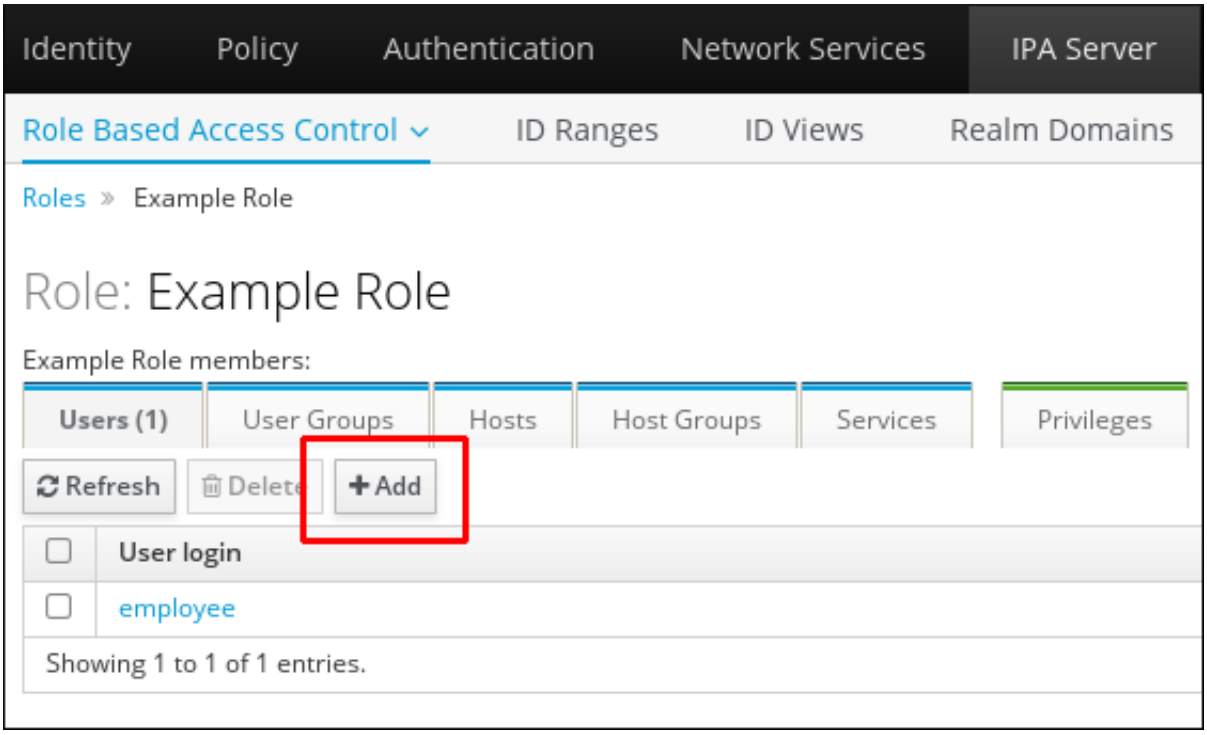


3. **Add Role** フォームが開きます。ロール名と説明を入力します。

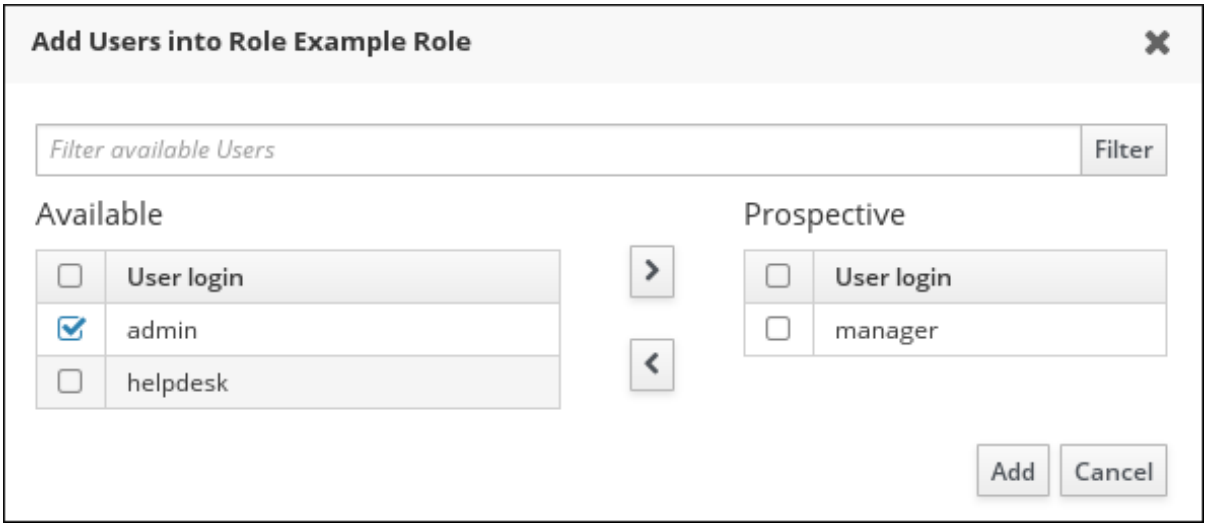
4. **Add and Edit** ボタンをクリックし、新規ロールを保存してロール設定ページに移動し、権限およびユーザーを追加します。

5. ロールリストのロール名をクリックして、ロールのプロパティーを編集します。ロール設定ページが開きます。

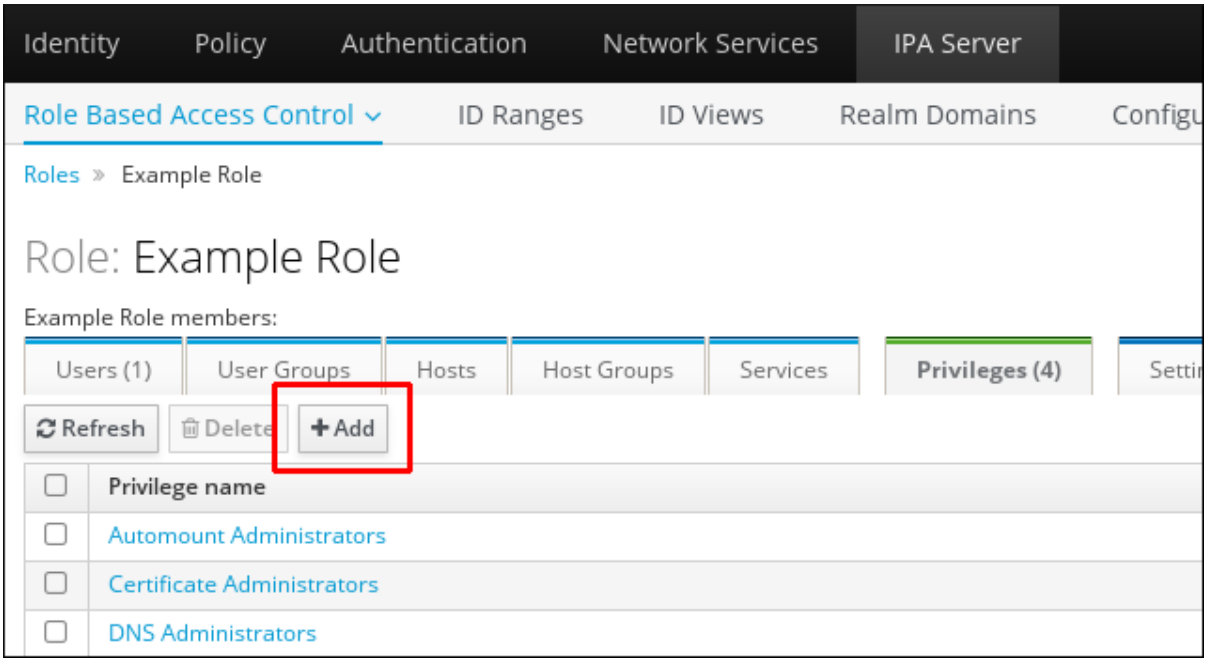
6. 関連するリストの上部にある **Add** ボタンをクリックして、**Users**、**Users Groups**、**Hosts**、**Host Groups**、または **Services** タブを使用してメンバーを追加します。



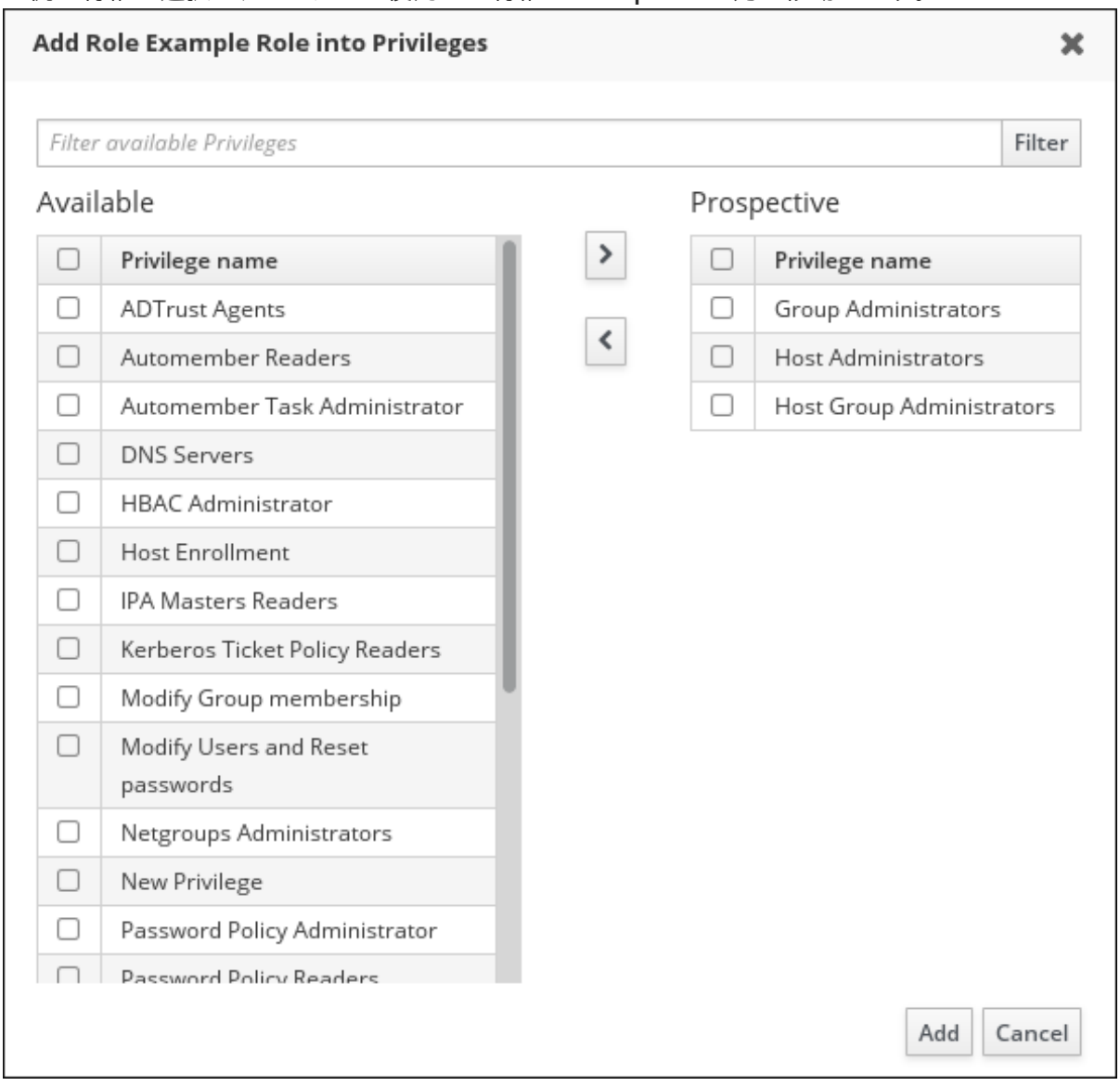
7. 開いているウィンドウで、左側のメンバーを選択し、> ボタンを使用して、メンバーを **Prospective** 列に移動します。



8. **Privileges** タブの上部にある **Add** をクリックします。



9. 左側の特権を選択し、> ボタンを使用して特権を **Prospective** 列に移動します。



10. **Add** ボタンをクリックして保存します。

11. **オプション。**ロールから特権またはメンバーを削除する必要がある場合は、削除するエンティティーの名前の横にあるチェックボックスにチェックマークを入れてから **Delete** ボタンをクリックします。ダイアログが開きます。

12. **オプション。**既存のロールを削除する必要がある場合は、リストで名前の横にあるチェックボックスを編集した後に **Delete** ボタンをクリックすると、**Remove roles** ダイアログが表示します。

第29章 ANSIBLE PLAYBOOK を使用した IDM でのロールベースアクセス制御の管理

ロールベースアクセス制御 (RBAC) は、ロールおよび権限関連を定義する、ポリシーに依存しないアクセス制御メカニズムです。Identity Management (IdM) の RBAC のコンポーネントは、ロール、権限、パーミッションです。

- **パーミッション** は、ユーザーの追加または削除、グループの変更、読み取りアクセスの有効化など、特定のタスクを実行する権限を付与します。
- **特権** は、新規ユーザーの追加に必要な全権限など、権限を組み合わせます。
- **ロール** は、ユーザー、ユーザーグループ、ホスト、またはホストグループに特権のセットを付与します。

特に大企業では、RBAC を使用すると、責任の領域を個別に設定する階層管理システムを作成できます。

本章では、Ansible Playbook を使用した RBAC の管理時に行う以下の操作について説明します。

- IdM のパーミッション
- デフォルトの管理パーミッション
- IdM の特権
- IdM のロール
- IdM の事前定義されたロール
- Ansible を使用して特権のある IdM RBAC ロールを存在させる手順
- Ansible を使用して IdM RBAC ロールを設定しないようにする手順
- Ansible を使用して、ユーザーグループに IdM RBAC ロールを割り当てる手順
- Ansible を使用して特定のユーザーに IdM RBAC ロールが割り当てられないようにする手順
- Ansible を使用してサービスを IdM RBAC ロールに所属させるように設定する手順
- Ansible を使用してホストを IdM RBAC ロールに所属させるように設定する手順
- Ansible を使用してホストグループを IdM RBAC ロールに所属させるように設定する手順

29.1. IDM のパーミッション

パーミッションは、ロールベースのアクセス制御の中で最も低いレベルの単位で、操作を適用する LDAP エントリーと合わせて操作を定義します。ブロックの構築と同様に、パーミッションは必要に応じて多くの権限に割り当てることができます。
1 つ以上の **権限** を使用して、許容される操作を定義します。

- **write**
- **read**
- **search**

- **compare**
- **add**
- **delete**
- **all**

上記の操作は、3 つの基本的な **ターゲット** に適用されます。

- **subtree**: ドメイン名 (DN) (この DN のサブツリー)
- **target filter**: LDAP フィルター
- **target**: DN。ワイルドカードでエントリーを指定可能。

また、以下の便利なオプションは、対応する属性を設定します。

- **type**: オブジェクトのタイプ (ユーザー、グループなど) (**subtree** および **target filter** を設定します)。
- **memberof**: グループのメンバー。**target filter** を設定します。
- **targetgroup**: 特定のグループを変更する権限 (グループメンバーシップの管理権限の付与など) を付与します (**target** を設定します)。

IdM パーミッションを使用すると、どのユーザーがどのオブジェクトにアクセスできるか、さらにこのようなオブジェクトの属性にアクセスできるかどうかを制御できます。IdM を使用すると、個別の属性を許可または拒否したり、ユーザー、グループ、sudo などの特定の IdM 機能を、全匿名ユーザー、全認証済みユーザー、または特定の特権ユーザーグループ限定などと、全体的な表示設定を変更したりできます。
たとえば、このアプローチではパーミッション指定に柔軟性があるので、アクセスが必要な特定のセクションのみにユーザーまたはグループのアクセスを制限し、他のセクションをこれらのユーザーまたはグループには完全に表示されないように設定する場合に、管理者にとって便利です。

注記

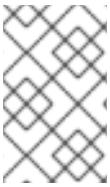
パーミッションには他のパーミッションを含めることはできません。

29.2. デフォルトの管理パーミッション

管理パーミッションは、IdM にデフォルトで含まれているパーミッションです。このパーミッションはユーザーが作成した他のパーミッションと同様に機能しますが、以下の相違点があります。

- この管理パーミッションは削除できず、名前、場所、ターゲットの属性を変更できません。
- このパーミッションには 3 つの属性セットがあります。
  - **デフォルト** の属性。IdM で管理されているため、ユーザーは変更できません。
  - **包含** 属性。ユーザーが別途追加する属性。
  - **除外** 属性。ユーザーが削除する属性。

管理パーミッションは、デフォルトおよび包含属性セットに表示されている属性すべてに適用されますが、除外セットに表示されている属性には適用されません。



注記

管理パーミッションを削除できませんが、パーミッションにバインドタイプを設定し、すべての特権から管理パーミッションを削除して管理パーミッションを効果的に無効にできます。

管理パーミッションの名前はすべて **System:** から始まります (例: **System: Add Sudo rule** または **System: Modify Services**)。以前のバージョンの IdM では、デフォルトのパーミッションに異なるスキームを使用していました。たとえば、ユーザーはパーミッションの削除はできず、特権に割り当てるしかできませんでした。これらのデフォルトパーミッションのほとんどは、管理パーミッションに切り替わっていますが、以下のパーミッションは引き続き以前のスキームを使用します。

- Automember Rebuild メンバーシップタスクの追加
- 設定サブエントリーの追加
- レプリカ合意の追加
- 証明書削除保留
- CA から証明書のステータス取得
- DNA 範囲の読み取り
- DNA 範囲の変更
- PassSync Manager の設定の読み取り
- PassSync Manager 設定の変更
- レプリカ合意の読み込み
- レプリカ合意の修正
- レプリカ合意の削除
- LDBM データベース設定の読み取り
- 証明書の要求
- CA ACL を無視する証明書の要求
- 別のホストからの証明書の要求
- CA からの証明書の取得
- 証明書の取り消し
- IPA 設定の書き込み

注記

コマンドラインから管理パーミッションを変更しようとし、変更不可な属性の変更をシステム側が許可しない場合には、コマンドに失敗します。Web UI から管理パーミッションを変更しようとした場合には、変更できない属性が無効になります。

29.3. IDM の特権

特権は、ロールに適用されるパーミッションのグループです。
パーミッションは単一の操作を実行する権限を提供しますが、IdM タスクを成功させるには、複数のパーミッションが必要なものがあります。したがって、特権は、特定のタスクを実行するために必要な異なるパーミッションを組み合わせたものです。
たとえば、新しい IdM ユーザーにアカウントを設定するには、以下の権限が必要です。

- 新規ユーザーエントリーの作成
- ユーザーパスワードのリセット
- 新規ユーザーのデフォルト IPA ユーザーグループへの追加

これらの 3 つの低レベルのタスクを、**ユーザーの追加** という名前のカスタム特権の形式で、権限がより高いレベルのタスクに組み合わせることで、システム管理者はロールを管理しやすくなります。IdM には、すでにいくつかのデフォルト特権が含まれています。ユーザーとユーザーグループとは別に、権限はホストおよびホストグループ、およびネットワークサービスにも割り当てられます。これにより、特定のネットワークサービスを使用するホストセットのユーザーセットによって、操作をきめ細かく制御できます。

注記

特権には、他の特権を含めることはできません。

29.4. IDM のロール

ロールは、ロールに指定したユーザーが所有する権限のリストです。
実際には、パーミッションでは、指定の低階層のタスク (ユーザーエントリーの作成、グループへのエントリーの追加など) を実行する権限を付与し、特権では、高階層のタスク (指定のグループへの新規ユーザーの作成など) に必要なこれらのパーミッションの 1 つ以上を組み合わせます。ロールは必要に応じて、管理者ロールでユーザーの追加、変更、削除ができるなど、特権をまとめます。

重要

ロールは、許可されたアクションを分類するために使用されます。ロールは、特権昇格されないようにしたり、特権の分離を実装するツールとしては使用しません。

注記

ロールに他のロールを含めることはできません。

29.5. IDENTITY MANAGEMENT で事前定義されたロール

Red Hat Identity Management には、以下の事前定義済みのロールが含まれています。

表29.1 Identity Management の定義済みロール

ロール	特権	説明
登録管理者	ホストの登録	クライアントまたはホストの登録を行います。

ロール	特権	説明
helpdesk	Modify Users、Reset passwords、Modify Group membership	簡単なユーザー管理タスクを実行します。
IT Security Specialist	Netgroups Administrators、HBAC Administrator、Sudo Administrator	ホストベースのアクセス制御、sudo ルールなどのセキュリティーポリシーを管理します。
IT Specialist	Host Administrators、Host Group Administrators、Service Administrators、Automount Administrators	ホストの管理を行います
Security Architect	Delegation Administrator、Replication Administrators、Write IPA Configuration、Password Policy Administrator	Identity Management 環境の管理、信頼の作成、レプリカ合意を作成します。
User Administrator	User Administrators、Group Administrators、Stage User Administrators	ユーザーおよびグループの作成を行います

29.6. ANSIBLE を使用して特権のある IDM RBAC ロールを存在させる手順

デフォルトのロール以外で、ロールベースのアクセス制御 (RBAC) を使用して Identity Management (IdM) のリソースを詳細にわたり制御するには、カスタムロールを作成します。

以下の手順では、Ansible Playbook を使用して、新しい IdM カスタムロールの特権を定義し、その存在を確認する方法を説明します。この例では、新しい **user_and_host_administrator** ロールには、デフォルトで IdM で存在する以下の特権を一意に組み合わせたものが含まれます。

- **Group Administrators**
- **User Administrators**
- **Stage User Administrators**
- **Group Administrators**

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。

- この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
- この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。

- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **~/<MyPlaybooks>/** ディレクトリーに移動します。

```
$ cd ~/<MyPlaybooks>/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/role/** にある **role-member-user-present.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/role/role-member-user-present.yml role-member-user-present-copy.yml
```

3. Ansible Playbook ファイル (**role-member-user-present-copy.yml**) を開きます。
4. **iparole** タスクセクションに以下の変数を設定して、ファイルを調整します。
   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は新規ロールの名前に設定します。
   - **特権** リストは、新しいロールに追加する IdM 権限の名前に設定します。
   - 必要に応じて、**user** 変数は、新規ロールを付与するユーザーの名前に設定します。
   - 必要に応じて、**group** 変数は、新規ロールを付与するグループの名前に設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Playbook to manage IPA role with members.
  hosts: ipaserver
  become: true
  gather_facts: no

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - iparole:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: user_and_host_administrator
      user: idm_user01
      group: idm_group01
      privilege:
      - Group Administrators
      - User Administrators
      - Stage User Administrators
      - Group Administrators
```

–

5. ファイルを保存します。

6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
~/<MyPlaybooks>/inventory role-member-user-present-copy.yml
```

関連情報

- Ansible Vault を使用したコンテンツの暗号化 を参照してください。
- IdM のロール を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-role** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/iparole** ディレクトリーのサンプルの Playbook を参照してください。

29.7. ANSIBLE を使用して IDM RBAC ロールを設定しないようにする手順

Identity Management (IdM) のロールベースアクセス制御 (RBAC) を管理するシステム管理者は、誤って管理者がユーザーに割り当てることがないように、使用しなくなったロールが削除されていることを確認する必要があります。

以下の手順では、Ansible Playbook を使用してロールが削除されていることを確認する方法を説明します。以下の例では、カスタムの **user_and_host_administrator** ロールが IdM に存在しないことを確認する方法を説明します。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **~/<MyPlaybooks>/** ディレクトリーに移動します。

```
$ cd ~/<MyPlaybooks>/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/role/** にある **role-is-absent.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/role/role-is-absent.yml role-is-absent-copy.yml
```

3. Ansible Playbook ファイル (**role-is-absent-copy.yml**) を開きます。

4. **iparole** タスクセクションに以下の変数を設定して、ファイルを調整します。

   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は、ロールの名前に設定します。
   - **state** 変数は、**absent** に設定されていることを確認します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Playbook to manage IPA role with members.
  hosts: ipaserver
  become: true
  gather_facts: no

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - iparole:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: user_and_host_administrator
      state: absent
```

5. ファイルを保存します。

6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i ~/<MyPlaybooks>/inventory role-is-absent-copy.yml
```

関連情報

- Ansible Vault を使用したコンテンツの暗号化 を参照してください。
- IdM のロール を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-role** Markdown ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/iparole** ディレクトリーのサンプルの Playbook を参照してください。

29.8. ANSIBLE を使用して、ユーザーグループに IDM RBAC ロールを割り当てる手順

Identity Management (IdM) のロールベースアクセス制御 (RBAC) を管理するシステム管理者は、junior administrators など、特定のユーザーグループにロールを割り当てます。

以下の例では、Ansible Playbook を使用して、同梱の IdM RBAC **helpdesk** ロールを **junior_sysadmins** に割り当てる方法を説明します。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **~/<MyPlaybooks>/** ディレクトリーに移動します。

```
$ cd ~/<MyPlaybooks>/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/role/** にある **role-member-group-present.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/role/role-member-group-present.yml
role-member-group-present-copy.yml
```

3. Ansible Playbook ファイル (**role-member-group-present-copy.yml**) を開きます。
4. **iparole** タスクセクションに以下の変数を設定して、ファイルを調整します。
   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は、割り当てるロールの名前に設定します。
   - **group** 変数はグループ名に設定します。
   - **action** 変数は **member** に設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Playbook to manage IPA role with members.
  hosts: ipaserver
  become: true
  gather_facts: no
```

```
vars_files:
- /home/user_name/MyPlaybooks/secret.yml
tasks:
- iparole:
    ipaadmin_password: "{{ ipaadmin_password }}"
    name: helpdesk
    group: junior_sysadmins
    action: member
```

5. ファイルを保存します。

6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
~/<MyPlaybooks>/inventory role-member-group-present-copy.yml
```

関連情報

- Ansible Vault を使用したコンテンツの暗号化 を参照してください。
- IdM のロール を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-role** Markdown ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/iparole** ディレクトリーのサンプルの Playbook を参照してください。

29.9. ANSIBLE を使用して特定のユーザーに IDM RBAC ロールが割り当てられないようにする手順

Identity Management (IdM) のロールベースアクセス制御 (RBAC) を管理するシステム管理者は、会社内で別の役職に異動した後など、特定のユーザーに RBAC ロールが割り当てられないようにすることもできます。

以下の手順では、Ansible Playbook を使用して、 **user_01** および **user_02** という名前のユーザーが **helpdesk** ロールに割り当てられないようにする方法を説明します。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。

- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **~/<MyPlaybooks>/** ディレクトリーに移動します。

```
$ cd ~/<MyPlaybooks>/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/role/** にある **role-member-user-absent.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/role/role-member-user-absent.yml role-member-user-absent-copy.yml
```

3. Ansible Playbook ファイル (**role-member-user-absent-copy.yml**) を開きます。

4. **iparole** タスクセクションに以下の変数を設定して、ファイルを調整します。

   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は、割り当てるロールの名前に設定します。
   - **user** リストをユーザーの名前に設定します。
   - **action** 変数は **member** に設定します。
   - **state** 変数は **absent** に設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Playbook to manage IPA role with members.
  hosts: ipaserver
  become: true
  gather_facts: no

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - iparole:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: helpdesk
      user
      - user_01
      - user_02
      action: member
      state: absent
```

5. ファイルを保存します。

6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
~/<MyPlaybooks>/inventory role-member-user-absent-copy.yml
```

関連情報

- Ansible Vault を使用したコンテンツの暗号化 を参照してください。
- IdM のロール を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-role** Markdown ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/iparole** ディレクトリーのサンプルの Playbook を参照してください。

29.10. ANSIBLE を使用してサービスを IDM RBAC ロールに所属させるように設定する手順

Identity Management (IdM) のロールベースアクセス制御 (RBAC) を管理するシステム管理者は、IdM に登録されている特定のサービスが、特定のロールのメンバーになっていることを確認する必要がある場合があります。以下の例では、カスタムの **web_administrator** ロールを使用して **client01.idm.example.com** サーバー上で実行中の **HTTP** サービスを管理できるようにする方法を説明します。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- **web_administrator** ロールが IdM に存在する。
- **HTTP/client01.idm.example.com@IDM.EXAMPLE.COM** サービスが IdM に存在する。

手順

1. **~/<MyPlaybooks>/** ディレクトリーに移動します。

```
$ cd ~/<MyPlaybooks>/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/role/** にある **role-member-service-present.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/role/role-member-service-present-absent.yml role-member-service-present-copy.yml
```

3. Ansible Playbook ファイル (**role-member-service-present-copy.yml**) を開きます。

4. **iparole** タスクセクションに以下の変数を設定して、ファイルを調整します。

   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は、割り当てるロールの名前に設定します。
   - **service** リストはサービス名に設定します。
   - **action** 変数は **member** に設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Playbook to manage IPA role with members.
  hosts: ipaserver
  become: true
  gather_facts: no

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - iparole:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: web_administrator
      service:
      - HTTP/client01.idm.example.com
      action: member
```

5. ファイルを保存します。

6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i ~/<MyPlaybooks>/inventory role-member-service-present-copy.yml
```

関連情報

- Ansible Vault を使用したコンテンツの暗号化 を参照してください。
- IdM のロール を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-role** Markdown ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/iparole** ディレクトリーのサンプルの Playbook を参照してください。

29.11. ANSIBLE を使用してホストを IDM RBAC ロールに所属させるように設定する手順

Identity Management (IdM) でロールベースアクセス制御を管理するシステム管理者は、特定のホストまたはホストグループが特定のロールに関連付けられていることを確認する必要がある場合があります。以下の例では、カスタムの **web_administrator** ロールが **HTTP** サービスを実行している **client01.idm.example.com** の IdM ホストを管理できるようにする方法を説明します。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- **web_administrator** ロールが IdM に存在する。
- **client01.idm.example.com** ホストが IdM に存在する。

手順

1. **~/<MyPlaybooks>/** ディレクトリーに移動します。

```
$ cd ~/<MyPlaybooks>/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/role/** にある **role-member-host-present.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/role/role-member-host-present.yml role-member-host-present-copy.yml
```

3. Ansible Playbook ファイル (**role-member-host-present-copy.yml**) を開きます。
4. **iparole** タスクセクションに以下の変数を設定して、ファイルを調整します。
   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は、割り当てるロールの名前に設定します。
   - **host** リストをホストの名前に設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Playbook to manage IPA role with members.
  hosts: ipaserver
  become: true
  gather_facts: no

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - iparole:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: web_administrator
      host:
      - client01.idm.example.com
      action: member
```

5. ファイルを保存します。

6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
~/<MyPlaybooks>/inventory role-member-host-present-copy.yml
```

関連情報

- Ansible Vault を使用したコンテンツの暗号化 を参照してください。
- IdM のロール を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-role** Markdown ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/iparole** ディレクトリーのサンプルの Playbook を参照してください。

29.12. ANSIBLE を使用してホストグループを IDM RBAC ロールに所属させるように設定する手順

Identity Management (IdM) でロールベースアクセス制御を管理するシステム管理者は、特定のホストまたはホストグループが特定のロールに関連付けられていることを確認する必要がある場合があります。以下の例では、カスタムの **web_administrator** ロールが **HTTP** サービスを実行している IdM ホストの **web_servers** グループを管理できるようにする方法を説明します。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。

- この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
- この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。

- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- **web_administrator** ロールが IdM に存在する。
- **web_servers** ホストグループが IdM に存在する。

手順

1. **~/<MyPlaybooks>/** ディレクトリーに移動します。

```
$ cd ~/<MyPlaybooks>/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/role/** にある **role-member-hostgroup-present.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/role/role-member-hostgroup-present.yml role-member-hostgroup-present-copy.yml
```

3. Ansible Playbook ファイル (**role-member-hostgroup-present-copy.yml**) を開きます。
4. **iparole** タスクセクションに以下の変数を設定して、ファイルを調整します。
   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は、割り当てるロールの名前に設定します。
   - **hostgroup** リストはホストグループ名に設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Playbook to manage IPA role with members.
  hosts: ipaserver
  become: true
  gather_facts: no

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - iparole:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: web_administrator
      hostgroup:
      - web_servers
      action: member
```

5. ファイルを保存します。

6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
~/<MyPlaybooks>/inventory role-member-hostgroup-present-copy.yml
```

関連情報

- Ansible Vault を使用したコンテンツの暗号化 を参照してください。
- IdM のロール を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-role** Markdown ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/iparole** ディレクトリーのサンプルの Playbook を参照してください。

第30章 ANSIBLE PLAYBOOK を使用した RBAC 権限の管理

ロールベースアクセス制御 (RBAC) は、ロール、権限およびパーミッションで定義する、ポリシーに依存しないアクセス制御メカニズムです。特に大企業では、RBAC を使用すると、責任の領域を個別に設定する階層管理システムを作成できます。

本章では、Ansible Playbook を使用して Identity Management (IdM) で RBAC 権限を管理する以下の操作について説明します。

- Ansible を使用してカスタムの RBAC 特権を存在させる手順
- Ansible を使用してカスタムの IdM RBAC 特権にメンバーパーミッションを存在させる手順
- Ansible を使用して IdM RBAC 特権にパーミッションが含まれないようにする手順
- Ansible を使用してカスタムの IdM RBAC 権限の名前を変更する手順
- Ansible を使用して IdM RBAC 特権が含まれないようにする手順

前提条件

- RBAC の概念と原則 を理解している。

30.1. ANSIBLE を使用してカスタムの IDM RBAC 特権を存在させる手順

Identity Management (IdM) のロールベースアクセス制御 (RBAC) でカスタム権限を完全に機能させるには、ステージごとに進めていく必要があります。

1. パーミッションが割り当てられていない特権を作成します。
2. 選択したパーミッションを特権に追加します。

以下の手順では、後でパーミッションを追加できるように、Ansible Playbook を使用して空の特権を作成する方法を説明します。この例では、ホスト管理に関連するすべての IdM パーミッションを組み合わせられるように **full_host_administration** という名前の特権を作成する方法を説明します。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/privilege/** にある **privilege-present.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/privilege/privilege-present.yml privilege-present-copy.yml
```

3. Ansible Playbook ファイル (**privilege-present-copy.yml**) を開きます。

4. **ipaprivilege** タスクセクションに以下の変数を設定してファイルを調整します。

   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は、新しい特権 **full_host_administration** の名前に設定します。
   - 必要に応じて、**description** 変数を使用して特権を記述します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Privilege present example
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure privilege full_host_administration is present
    ipaprivilege:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: full_host_administration
      description: This privilege combines all IdM permissions related to host administration
```

5. ファイルを保存します。

6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory privilege-present-copy.yml
```

30.2. ANSIBLE を使用してカスタムの IDM RBAC 特権にメンバーパーミッションを存在させる手順

Identity Management (IdM) のロールベースアクセス制御 (RBAC) でカスタム権限を完全に機能させるには、ステージごとに進めていく必要があります。

1. パーミッションが割り当てられていない特権を作成します。

2. 選択したパーミッションを特権に追加します。

以下の手順では、Ansible Playbook を使用して、前の手順で作成した特権にパーミッションを追加する方法を説明します。この例では、ホスト管理に関連する IdM パーミッションをすべて、**full_host_administration** という名前の特権に追加する方法を説明します。デフォルトでは、パーミッションは **Host Enrollment**、**Host Administrators** および **Host Group Administrator** 特権の間で分散されます。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- **full_host_administration** 権限が存在する。Ansible を使用して特権を作成する方法の詳細は、Ansible を使用してカスタムの IdM RBAC 特権を存在させる手順 を参照してください。

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/privilege/** にある **privilege-member-present.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/privilege/privilege-member-present.yml
privilege-member-present-copy.yml
```

3. Ansible Playbook ファイル (**privilege-member-present-copy.yml**) を開きます。
4. **ipaprivilege** タスクセクションに以下の変数を設定してファイルを調整します。
   - 使用しているユースケースに合わせて、タスクの **名前** を調節します。
   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は、特権の名前に設定します。
   - **permission** は、特権に追加するパーミッションの名前を設定します。
   - **action** 変数が **member** に設定されていることを確認します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Privilege member present example
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure that permissions are present for the "full_host_administration" privilege
    ipaprivilege:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: full_host_administration
      permission:
      - "System: Add krbPrincipalName to a Host"
      - "System: Enroll a Host"
      - "System: Manage Host Certificates"
      - "System: Manage Host Enrollment Password"
      - "System: Manage Host Keytab"
      - "System: Manage Host Principals"
      - "Retrieve Certificates from the CA"
      - "Revoke Certificate"
      - "System: Add Hosts"
      - "System: Add krbPrincipalName to a Host"
      - "System: Enroll a Host"
      - "System: Manage Host Certificates"
      - "System: Manage Host Enrollment Password"
      - "System: Manage Host Keytab"
      - "System: Manage Host Keytab Permissions"
      - "System: Manage Host Principals"
      - "System: Manage Host SSH Public Keys"
      - "System: Manage Service Keytab"
      - "System: Manage Service Keytab Permissions"
      - "System: Modify Hosts"
      - "System: Remove Hosts"
      - "System: Add Hostgroups"
      - "System: Modify Hostgroup Membership"
      - "System: Modify Hostgroups"
      - "System: Remove Hostgroups"
```

5. ファイルを保存します。

6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory privilege-member-present-copy.yml
```

30.3. ANSIBLE を使用して IDM RBAC 特権にパーミッションが含まれないようにする手順

Identity Management (IdM) のシステム管理者は、IdM のロールベースアクセス制御をカスタマイズできます。

以下の手順では、Ansible Playbook を使用して、特権からパーミッションを削除する方法を説明します。この例では、管理者がセキュリティー上のリスクを考慮するため、デフォルトの **Certificate Administrators** 特権から **Request Certificates ignoring CA ACLs** を削除する方法を説明します。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/privilege/** にある **privilege-member-present.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/privilege/privilege-member-absent.yml
privilege-member-absent-copy.yml
```

3. Ansible Playbook ファイル (**privilege-member-absent-copy.yml**) を開きます。
4. **ipaprivilege** タスクセクションに以下の変数を設定してファイルを調整します。
   - 使用しているユースケースに合わせて、タスクの **名前** を調節します。
   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は、特権の名前に設定します。
   - **permission** のリストは、特権から削除するパーミッションの名前に設定します。
   - **action** 変数が **member** に設定されていることを確認します。
   - **state** 変数は **absent** に設定されていることを確認します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Privilege absent example
  hosts: ipaserver
```

```
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure that the "Request Certificate ignoring CA ACLs" permission is absent from
the "Certificate Administrators" privilege
    ipaprivilege:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: Certificate Administrators
      permission:
      - "Request Certificate ignoring CA ACLs"
      action: member
      state: absent
```

5. ファイルを保存します。

6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory privilege-member-absent-copy.yml
```

30.4. ANSIBLE を使用してカスタムの IDM RBAC 権限の名前を変更する手順

Identity Management (IdM) のシステム管理者は、IdM のロールベースアクセス制御をカスタマイズできます。

以下の手順では、たとえば、パーミッションの一部を削除したなどの理由から、特権の名前を変更する方法を説明します。パーミッションを削除した結果、特権の名前は正確ではなくなりました。この例では、管理者は **full_host_administration** 特権の名前を **limited_host_administration** に変更します。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- **full_host_administration** 権限が存在する。特権の追加方法は、Ansible を使用してカスタムの IdM RBAC 特権を存在させる手順 を参照してください。

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/privilege/** にある **privilege-present.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/privilege/privilege-present.yml rename-privilege.yml
```

3. Ansible Playbook ファイル (**rename-privilege.yml**) を開いて編集します。

4. **ipaprivilege** タスクセクションに以下の変数を設定してファイルを調整します。

- **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
- **name** 変数は、現在の特権名に設定します。
- **rename** 変数を追加して、特権の新しい名前に設定します。
- **state** 変数を追加し、 **renamed** に設定します。

5. 以下のように、Playbook 自体の名前を変更します。

```
---
- name: Rename a privilege
  hosts: ipaserver
```

6. 以下のように、Playbook のタスクの名前を変更します。

```
[...]
tasks:
- name: Ensure the full_host_administration privilege is renamed to limited_host_administration
  ipaprivilege:
  [...]
```

以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Rename a privilege
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure the full_host_administration privilege is renamed to limited_host_administration
    ipaprivilege:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: full_host_administration
      rename: limited_host_administration
      state: renamed
```

7. ファイルを保存します。

8. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory rename-privilege.yml
```

30.5. ANSIBLE を使用して IDM RBAC 特権が含まれないようにする手順

Identity Management (IdM) のシステム管理者は、IdM のロールベースアクセス制御をカスタマイズできます。以下の手順では、Ansible Playbook を使用して RBAC 特権が削除されていることを確認する方法を説明します。この例では、**CA administrator** 特権が存在しないことを確認する方法を説明します。この手順が終わると、IdM で認証局を管理できるユーザーは **admin** だけになります。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/privilege/** ディレクトリーにある **privilege-absent.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/privilege/privilege-absent.yml privilege-absent-copy.yml
```

3. Ansible Playbook ファイル (**privilege-absent-copy.yml**) を開きます。
4. **ipaprivilege** タスクセクションに以下の変数を設定してファイルを調整します。
   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は、削除する権限の名前に設定します。
   - **state** 変数が **absent** に設定されていることを確認します。
5. 以下のように、Playbook のタスクの名前を変更します。

```
[...]
tasks:
- name: Ensure privilege "CA administrator" is absent
  ipaprivilege:
  [...]
```

以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Privilege absent example
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure privilege "CA administrator" is absent
    ipaprivilege:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: CA administrator
      state: absent
```

6. ファイルを保存します。

7. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory privilege-absent-copy.yml
```

30.6. 関連情報

- IdM の特権 を参照してください。
- IdM のパーミッション を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーで利用可能な **README-privilege** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/ipaprivilege** ディレクトリーのサンプルの Playbook を参照してください。

第31章 ANSIBLE PLAYBOOK を使用した IDM での RBAC パーミッションの管理

ロールベースアクセス制御 (RBAC) は、ロール、権限およびパーミッションで定義する、ポリシーに依存しないアクセス制御メカニズムです。特に大企業では、RBAC を使用すると、責任の領域を個別に設定する階層管理システムを作成できます。

本章では、Ansible Playbook を使用して Identity Management (IdM) で RBAC パーミッションの管理時に行う、以下の操作について説明します。

- Ansible を使用して RBAC パーミッションを存在させる手順
- Ansible を使用して属性を含めて RBAC パーミッションを追加する手順
- Ansible を使用して RBAC パーミッションをバインドする手順
- Ansible を使用して属性を IdM RBAC パーミッションをメンバーにする手順
- Ansible を使用して属性が IdM RBAC パーミッションのメンバーに含まれないようにする手順
- Ansible を使用して IdM RBAC パーミッションの名前を変更する手順

前提条件

- RBAC の概念と原則 を理解している。

31.1. ANSIBLE を使用して RBAC パーミッションを存在させる手順

Identity Management (IdM) のシステム管理者は、IdM のロールベースアクセス制御 (RBAC) をカスタマイズできます。

以下の手順では、Ansible Playbook を使用して、パーミッションを特権に追加できるように IdM にパーミッションを追加する方法を説明します。この例では、目的とする以下の状態を達成する方法を説明します。

- **MyPermission** パーミッションが存在する。
- **MyPermission** パーミッションだけがホストに適用できる。
- パーミッションを含む特権を付与されたユーザーは、エントリーに対して以下の操作すべてを実行できます。
  - Write
  - Read
  - Search
  - Compare
  - Add
  - Delete

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/permission/** ディレクトリーにある **permission-present.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/permission/permission-present.yml
permission-present-copy.yml
```

3. Ansible Playbook ファイル (**permission-present-copy.yml**) を開きます。
4. **ipapermission** タスクセクションに以下の変数を設定して、ファイルを調整します。
   - 使用しているユースケースに合わせて、タスクの **名前** を調節します。
   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数はパーミッションの名前に設定します。
   - **object_type** 変数は **host** に設定します。
   - **right** 変数は **all** に設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Permission present example
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure that the "MyPermission" permission is present
    ipapermission:
      ipaadmin_password: "{{ ipaadmin_password }}"
```

```
name: MyPermission
object_type: host
right: all
```

5. ファイルを保存します。

6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory permission-present-copy.yml
```

31.2. ANSIBLE を使用して属性を含めて RBAC パーミッションを追加する手順

Identity Management (IdM) のシステム管理者は、IdM のロールベースアクセス制御 (RBAC) をカスタマイズできます。

以下の手順では、Ansible Playbook を使用して、パーミッションを特権に追加できるように IdM にパーミッションを追加する方法を説明します。この例では、目的とする以下の状態を達成する方法を説明します。

- **MyPermission** パーミッションが存在する。
- **MyPermission** パーミッションだけがホストの追加に使用できる。
- パーミッションを含む特権を付与されたユーザーは、ホストのエントリーに対して以下の操作すべてを実行できる。
  - Write
  - Read
  - Search
  - Compare
  - Add
  - Delete
- **MyPermission** パーミッションを含む特権のあるユーザーが作成したホストエントリーに **description** の値を追加できる。

注記

IdM LDAP スキーマでは、パーミッションの作成または変更時に指定できる属性のタイプは制約されません。ただし、**object_type** が **host** の場合に **attrs: car_licence** を指定すると、後でパーミッションを実行して、車のライセンス値をホストに追加使用とすると **ipa: ERROR: attribute "car-license" not allowed** というエラーメッセージが表示されます。

前提条件

- IdM 管理者パスワードを把握している。

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/permission/** ディレクトリーにある **permission-present.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/permission/permission-present.yml
permission-present-with-attribute.yml
```

3. Ansible Playbook ファイル (**permission-present-with-attribute.yml**) を開きます。
4. **ipapermission** タスクセクションに以下の変数を設定して、ファイルを調整します。
   - 使用しているユースケースに合わせて、タスクの **名前** を調節します。
   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数はパーミッションの名前に設定します。
   - **object_type** 変数は **host** に設定します。
   - **right** 変数は **all** に設定します。
   - **attrs** 変数は **description** に設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Permission present example
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure that the "MyPermission" permission is present with an attribute
    ipapermission:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: MyPermission
```

```
    object_type: host
    right: all
    attrs: description
```

5. ファイルを保存します。

6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory permission-present-with-attribute.yml
```

関連情報

- RHEL 7 の **Linux Domain Identity, Authentication and Policy Guide**の User and group schema を参照してください。

31.3. ANSIBLE を使用して RBAC パーミッションをバインドする手順

Identity Management (IdM) のシステム管理者は、IdM のロールベースアクセス制御 (RBAC) をカスタマイズできます。

以下の手順では、Ansible Playbook を使用して、パーミッションを特権に追加できないように IdM からパーミッションを削除する方法を説明します。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/permission/** ディレクトリーにある **permission-absent.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/permission/permission-absent.yml
permission-absent-copy.yml
```

3. Ansible Playbook ファイル (**permission-absent-copy.yml**) を開きます。
4. **ipapermission** タスクセクションに以下の変数を設定して、ファイルを調整します。
   - 使用しているユースケースに合わせて、タスクの **名前** を調節します。
   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数はパーミッションの名前に設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Permission absent example
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure that the "MyPermission" permission is absent
    ipapermission:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: MyPermission
      state: absent
```

5. ファイルを保存します。
6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

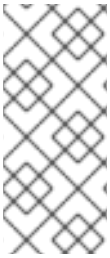
```
$ ansible-playbook --vault-password-file=password_file -v -i inventory permission-
absent-copy.yml
```

31.4. ANSIBLE を使用して属性を IDM RBAC パーミッションをメンバーにする手順

Identity Management (IdM) のシステム管理者は、IdM のロールベースアクセス制御 (RBAC) をカスタマイズできます。

以下の手順では、Ansible Playbook を使用して、属性が IdM の RBAC パーミッションのメンバーであることを確認する方法を説明します。この手順を行うと、パーミッションのあるユーザーは、属性のあるエントリーを作成できます。

この例では、**MyPermission** パーミッションを含む特権を持つユーザーにより作成されたホストエントリーに、**gecos** および **description** の値を設定する方法を説明します。



注記

IdM LDAP スキーマでは、パーミッションの作成または変更時に指定できる属性のタイプは制約されません。ただし、**object_type** が **host** の場合に **attrs: car_licence** を指定すると、後でパーミッションを実行して、車のライセンス値をホストに追加使用とすると **ipa: ERROR: attribute "car-license" not allowed** というエラーメッセージが表示されます。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- MyPermission パーミッションが存在する。

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/permission/** ディレクトリーにある **permission-member-present.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/permission/permission-member-present.yml permission-member-present-copy.yml
```

3. Ansible Playbook ファイル (**permission-member-present-copy.yml**) を開きます。
4. **ipapermission** タスクセクションに以下の変数を設定して、ファイルを調整します。
   - 使用しているユースケースに合わせて、タスクの **名前** を調節します。
   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数はパーミッションの名前に設定します。
   - **attrs** リストを **description** および **gecos** 変数に設定します。
   - **action** 変数が **member** に設定されていることを確認します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Permission member present example
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure that the "gecos" and "description" attributes are present in
"MyPermission"
    ipapermission:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: MyPermission
      attrs:
      - description
      - gecos
      action: member
```

5. ファイルを保存します。

6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory permission-
member-present-copy.yml
```

31.5. ANSIBLE を使用して属性が IDM RBAC パーミッションのメンバーに含まれないようにする手順

Identity Management (IdM) のシステム管理者は、IdM のロールベースアクセス制御 (RBAC) をカスタマイズできます。

以下の手順では、Ansible Playbook を使用して、属性が IdM の RBAC パーミッションに含まれないようにする方法を説明します。この手順を実行すると、パーミッションのあるユーザーが IdM LDAP にエントリーを作成すると、そのエントリーで、値に属性を関連付けて設定できません。

この例では、目的とする以下の状態を達成する方法を説明します。

- **MyPermission** パーミッションが存在する。
- **MyPermission** パーミッションを含む特権のあるユーザーが作成したホストエントリーに **description** 属性を使用できない。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。

    - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。

- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- MyPermission パーミッションが存在する。

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/permission/** ディレクトリーにある **permission-member-absent.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/permission/permission-member-absent.yml permission-member-absent-copy.yml
```

3. Ansible Playbook ファイル (**permission-member-absent-copy.yml**) を開きます。
4. **ipapermission** タスクセクションに以下の変数を設定して、ファイルを調整します。
    - 使用しているユースケースに合わせて、タスクの **名前** を調節します。
    - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
    - **name** 変数はパーミッションの名前に設定します。
    - **attrs** 変数は **description** に設定します。
    - **action** 変数は **member** に設定します。
    - **state** 変数が **absent** に設定されていることを確認します。

    以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Permission absent example
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure that an attribute is not a member of "MyPermission"
    ipapermission:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: MyPermission
      attrs: description
      action: member
      state: absent
```

5. ファイルを保存します。

6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory permission-member-absent-copy.yml
```

31.6. ANSIBLE を使用して IDM RBAC パーミッションの名前を変更する手順

Identity Management (IdM) のシステム管理者は、IdM のロールベースアクセス制御をカスタマイズできます。

以下の手順では、Ansible Playbook を使用してパーミッションの名前を変更する方法を説明します。この例では、**MyPermission** の名前を **MyNewPermission** に変更する方法を説明します。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- **MyPermission** が IdM に存在する。
- **MyNewPermission** が IdM に存在しない。

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/permission/** ディレクトリーにある **permission-renamed.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/permission/permission-renamed.yml permission-renamed-copy.yml
```

3. Ansible Playbook ファイル (**permission-renamed-copy.yml**) を開きます。
4. **ipapermission** タスクセクションに以下の変数を設定して、ファイルを調整します。
   - 使用しているユースケースに合わせて、タスクの **名前** を調節します。

- **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
- **name** 変数はパーミッションの名前に設定します。

以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Permission present example
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Rename the "MyPermission" permission
    ipapermission:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: MyPermission
      rename: MyNewPermission
      state: renamed
```

5. ファイルを保存します。
6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory permission-renamed-copy.yml
```

31.7. 関連情報

- IdM のパーミッション を参照してください。
- IdM の特権 を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーで利用可能な **README-permission** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/ipapermission** ディレクトリーのサンプルの Playbook を参照してください。

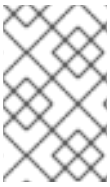
第32章 IDM でのユーザーパスワードの管理

32.1. IDM ユーザーパスワードは誰がどのように変更するのか

他のユーザーのパスワードを変更するパーミッションのない通常ユーザーは、独自の個人パスワードのみを変更できます。新しいパスワードは、そのユーザーがメンバーとなっているグループに適用される IdM パスワードポリシーに合致する必要があります。パスワードポリシーの設定方法は、IdM パスワードポリシーの定義 を参照してください。

管理者およびパスワード変更権限を持つユーザーは、新しいユーザーに初期パスワードを設定し、既存のユーザーのパスワードをリセットできます。これらのパスワードには、以下が該当します。

- IdM パスワードポリシーを満たす必要はありません。
- 最初のログインに成功したら失効します。このような場合、IdM はユーザーが期限切れのパスワードを直ちに変更するよう要求します。この動作を無効にするには、次回のログイン時にパスワード変更を求められることなく、IdM でパスワードリセットを有効にする を参照してください。



注記

LDAP Directory Manager(DM) ユーザーは、LDAP ツールを使用してユーザーパスワードを変更できます。新しいパスワードは、任意の IdM パスワードポリシーを上書きできます。DM によって設定されたパスワードは最初のログイン後に有効期限が切れません。

32.2. IDM WEB UI でのユーザーパスワードの変更

Identity Management (IdM) ユーザーは、IdM Web UI でユーザーパスワードを変更できます。

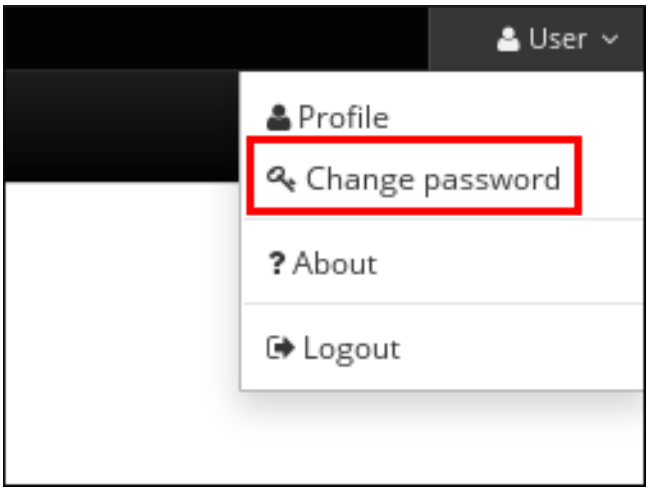
前提条件

- IdM Web UI にログインしている。

手順

1. 右上隅の **User name → Change password**をクリックします。

図32.1 パスワードのリセット



2. 現在のパスワードおよび新しいパスワードを入力します。

32.3. IDM WEB UI での別のユーザーのパスワードのリセット

Identity Management (IdM) の管理ユーザーは、IdM Web UI で他のユーザーのパスワードを変更できます。

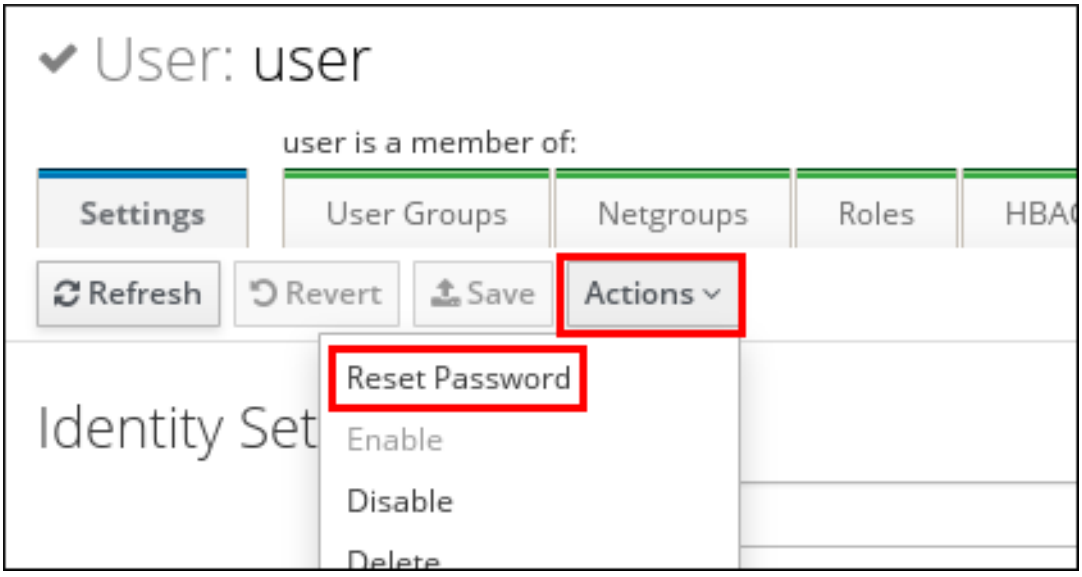
前提条件

- 管理ユーザーとして IdM Web UI にログインしている。

手順

1. **Identity → Users** を選択します。
2. 編集するユーザー名をクリックします。
3. **Actions → Reset password** をクリックします。

図32.2 パスワードのリセット



4. 新しいパスワードを入力し、**Reset Password** をクリックします。

図32.3 新しいパスワードの確認

32.4. DIRECTORY MANAGER ユーザーパスワードのリセット

Identity Management (IdM) Directory Manager のパスワードを紛失した場合は、リセットできます。

前提条件

- IdM サーバーに **root** にアクセスできる。

手順

1. **pwdhash** コマンドを使用して、新しいパスワードハッシュを生成します。以下に例を示します。

```
# pwdhash -D /etc/dirsrv/slapd-IDM-EXAMPLE-COM password
{PBKDF2_SHA256}AAAgABU0bKhyjY53NcxY33ueoPjOUWtl4iyYN5uW...
```

   Directory Server 設定へのパスを指定すると、**nsslapd-rootpwstoragescheme** 属性に設定されたパスワードストレージスキームが自動的に使用され、新しいパスワードを暗号化します。

2. トポロジー内のすべての IdM サーバーで、以下の手順を実行します。

   a. サーバーにインストールされている IdM サービスをすべて停止します。

```
# ipactl stop
```

   b. **/etc/dirsrv/IDM-EXAMPLE-COM/dse.ldif** ファイルーを編集し、**nsslapd-rootpw** 属性を、**pwdhash** コマンドで生成された値に設定します。

```
nsslapd-rootpw:
{PBKDF2_SHA256}AAAgABU0bKhyjY53NcxY33ueoPjOUWtl4iyYN5uW...
```

   c. サーバーにインストールされている IdM サービスをすべて起動します。

```
# ipactl start
```

32.5. IDM CLI でのユーザーパスワードの変更または別のユーザーのパスワードのリセット

Identity Management (IdM) コマンドラインインターフェイス (CLI) を使用して、ユーザーパスワードを変更できます。管理ユーザーの場合は、CLI を使用して別のユーザーのパスワードをリセットできます。

前提条件

- IdM ユーザーの TGT (Ticket-Granting Ticket) を取得している。
- 別のユーザーのパスワードをリセットする場合は、IdM の管理ユーザーの TGT を取得している必要がある。

手順

- ユーザーの名前と **--password** オプションを指定して、**ipa user-mod** コマンドを入力します。このコマンドにより、新しいパスワードの入力が求められます。

```
$ ipa user-mod idm_user --password
Password:
Enter Password again to verify:
```

```
--------------------
Modified user "idm_user"
--------------------
...
```

注記

ipa user-mod の代わりに **ipa passwd idm_user** を使用することもできます。

32.6. 次回のログイン時にパスワード変更を求められることなく、IDM でパスワードリセットを有効にする

デフォルトでは、管理者が別のユーザーのパスワードをリセットすると、初回のログインに成功したらパスワードが期限切れになります。IdM Directory Manager では、各 IdM 管理者に次の特権を指定できます。

- 初回ログイン後にパスワードの変更をユーザーに要求することなく、パスワードの変更操作を行うことができます。
- 強度や履歴の強制が適用されないようにパスワードポリシーをバイパスします。

警告

パスワードポリシーをバイパスすると、セキュリティー上の脅威になる可能性があります。これらの追加の特権を付与するユーザーを選択するときは注意してください。

前提条件

- Directory Manager のパスワードを把握している。

手順

1. ドメイン内のすべての Identity Management (IdM) サーバーで、次の変更を行います。
   a. **ldapmodify** コマンドを実行して、LDAP エントリーを変更します。IdM サーバーの名前と 389 ポートを指定し、Enter キーを押します。

   ```
   $ ldapmodify -x -D "cn=Directory Manager" -W -h server.idm.example.com -p 389
   Enter LDAP Password:
   ```

   b. Directory Manager パスワードを入力します。
   c. **ipa_pwd_extop** パスワード同期エントリーの識別名を入力し、Enter キーを押します。

   ```
   dn: cn=ipa_pwd_extop,cn=plugins,cn=config
   ```

   d. 変更の **modify** 型を指定し、Enter キーを押します。

```
changetype: modify
```

e. LDAP が実行する修正のタイプと、その属性を指定します。Enter キーを押します。

```
add: passSyncManagersDNs
```

f. **passSyncManagersDNs** 属性に管理ユーザーアカウントを指定します。属性は多値です。たとえば、**admin** ユーザーに、Directory Manager の電源をリセットするパスワードを付与するには、次のコマンドを実行します。

```
passSyncManagersDNs: \
uid=admin,cn=users,cn=accounts,dc=example,dc=com
```

g. Enter キーを 2 回押して、エントリーの編集を停止します。

手順全体を以下に示します。

```
$ ldapmodify -x -D "cn=Directory Manager" -W -h server.idm.example.com -p 389
Enter LDAP Password:
dn: cn=ipa_pwd_extop,cn=plugins,cn=config
changetype: modify
add: passSyncManagersDNs
passSyncManagersDNs: uid=admin,cn=users,cn=accounts,dc=example,dc=com
```

passSyncManagerDNs にリスト表示されている **admin** ユーザーに、追加特権が追加されました。

32.7. IDM ユーザーのアカウントがロックされているかどうかの確認

Identity Management (IdM) 管理者は、IdM ユーザーのアカウントがロックされているかどうかを確認できます。そのためには、ユーザーの最大許容ログイン試行回数と、ユーザーの実際の失敗ログイン回数を比較する必要があります。

前提条件

- IdM の管理ユーザーの TGT (Ticket-Granting Ticket) を取得している。

手順

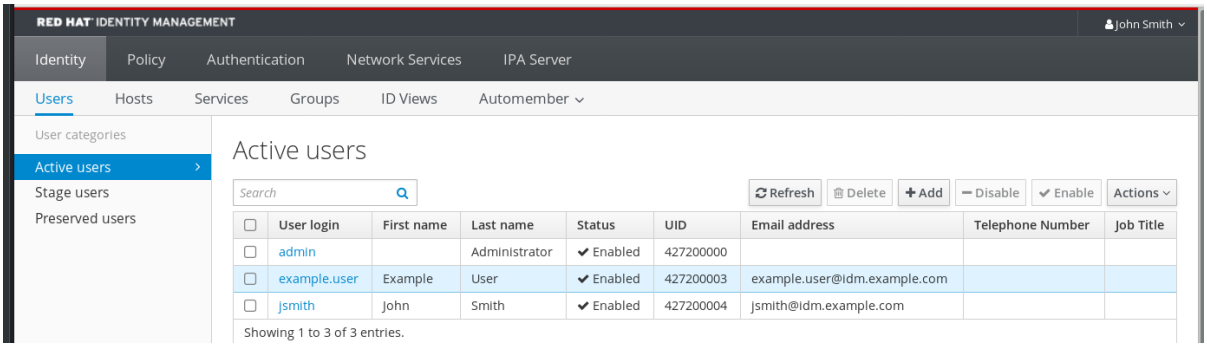
1. ユーザーアカウントのステータスを表示して、失敗したログインの数を確認します。

```
$ ipa user-status example_user
-----------------------
Account disabled: False
-----------------------
  Server: idm.example.com
  Failed logins: 8
  Last successful authentication: N/A
  Last failed authentication: 20220229080317Z
  Time now: 2022-02-29T08:04:46Z
----------------------------
Number of entries returned 1
----------------------------
```

2. 特定のユーザーに許可されたログイン試行回数を表示します。

   a. IdM 管理者として IdM Web UI にログインします。

   b. **Identity → Users → Active users**タブを開きます。



   a. ユーザー名をクリックして、ユーザー設定を開きます。

   b. **パスワードポリシー** セクションで、**Max failures** アイテムを探します。

3. **ipa user-status** コマンドの出力に表示されているログイン失敗回数と、IdM Web UI に表示されている**Max failures** 数を比較します。ログインに失敗した回数が、許可されている最大ログイン試行回数と等しい場合、ユーザーアカウントはロックされます。

関連情報

- IIdM でのパスワード失敗後のユーザーアカウントのロック解除

32.8. IIDM でのパスワード失敗後のユーザーアカウントのロック解除

ユーザーが間違ったパスワードを一定回数使用してログインしようとすると、Identity Management (IdM) はユーザーアカウントをロックするため、ユーザーはログインできなくなります。セキュリティー上の理由から、IdM では、ユーザーアカウントがロックされていることを示す警告メッセージは表示されません。代わりに、CLI プロンプトがユーザーにパスワードを何度も要求し続ける場合があります。

IdM は、指定した時間が経過した後にユーザーアカウントを自動的にアンロックします。または、以下の手順でユーザーアカウントのロックを手動で解除することもできます。

前提条件

- IdM 管理ユーザーの Ticket-Granting Ticket を取得している。

手順

- ユーザーアカウントのロックを解除するには、**ipa user-unlock** コマンドを実行します。

```
$ ipa user-unlock idm_user
-----------------------
Unlocked account "idm_user"
-----------------------
```

この後、ユーザーは再度ログインできるようになります。

関連情報

- IdM ユーザーのアカウントがロックされているかどうかの確認

32.9. IDM のユーザーに対する、最後に成功した KERBEROS 認証の追跡の有効化

パフォーマンス上の理由から、Red Hat Enterprise Linux 8 で実行している Identity Management (IdM) には、ユーザーが最後に成功した Kerberos 認証のタイムスタンプが保存されません。そのため、**ipa user-status** などの特定のコマンドではタイムスタンプが表示されません。

前提条件

- IdM の管理ユーザーの TGT (Ticket-Granting Ticket) を取得している。
- 手順を実行している IdM サーバーへの **root** アクセス権限がある。

手順

1. 現在有効なパスワードプラグイン機能を表示します。

```
# ipa config-show | grep "Password plugin features"
  Password plugin features: AllowNThash, KDC:Disable Last Success
```

この出力は、**KDC:Disable Last Success** プラグインが有効になっていることを示しています。このプラグインにより、最後に成功した Kerberos 認証試行が ipa user-status 出力に表示されなくなります。

2. 現在有効な **ipa config-mod** コマンドに、すべての機能の **--ipaconfigstring=feature** パラメーターを追加します (**KDC:Disable Last Success** を除く)。

```
# ipa config-mod --ipaconfigstring='AllowNThash'
```

このコマンドは、**AllowNThash** プラグインのみを有効にします。複数の機能を有効にするには、機能ごとに **--ipaconfigstring=feature** パラメーターを個別に指定します。

3. IdM を再起動します。

```
# ipactl restart
```

第33章 IDM パスワードポリシーの定義

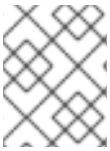
本章では、Identity Management (IdM) パスワードポリシーについて、また、Ansible Playbook を使用して IdM に新規パスワードポリシーを追加する方法を説明します。

33.1. パスワードポリシーとは

パスワードポリシーは、パスワードが満たす必要のある一連のルールです。たとえば、パスワードポリシーでは、パスワードの最小長と最大有効期間を定義できます。このポリシーの対象となる全ユーザーには、十分に長いパスワードを設定して、指定の条件を満たす頻度でパスワードを変更する必要があります。このようにパスワードポリシーを使用することで、ユーザーのパスワードが検出されて悪用されるリスクが軽減されます。

33.2. IDM のパスワードポリシー

パスワードは、Identity Management (IdM) ユーザーが IdM Kerberos ドメインに対して認証する最も一般的な方法です。パスワードポリシーでは、このような IdM ユーザーのパスワードが満たす必要条件を定義します。



注記

IdM パスワードポリシーは基礎となる LDAP ディレクトリーで設定されますが、Kerberos Key Distribution Center (KDC) はパスワードポリシーを強制的に使用します。

パスワードポリシー属性 は、IdM でのパスワードポリシーの定義に使用できる属性をリスト表示します。

表33.1 パスワードポリシーの属性

属性	説明	例
Max lifetime	パスワードのリセットが必要になるまでの、パスワードの有効期間 (日) の上限です。デフォルト値は 90 日です。 この属性が 0 に設定されている場合、パスワードの有効期限は切れないことに注意してください。	Max lifetime = 180 ユーザーパスワードは 180 日間のみ有効です。有効期限が経過すると、IdM は変更を求めるプロンプトを表示します。
Min lifetime	パスワードを変更してから次に変更操作を行うまでに最小限開ける必要のある時間。	Min lifetime = 1 ユーザーがパスワードの変更後に、次に変更するまでに最低でも 1 時間待機する必要があります。
History size	保存される以前のパスワード数。パスワードの履歴にあるパスワードを再利用できませんが、保存されていない以前のものは使用できます。	History size = 0 この場合、パスワード履歴は空になり、ユーザーは以前のパスワードをどれでも再利用できます。

属性	説明	例
Character classes	パスワードで使用する必要のある文字クラスの数。文字クラスは次のとおりです。 * 大文字 * 小文字 * 数字 * コンマ (,)、ピリオド (.)、アスタリスク (*) などの特殊文字 * 他の UTF-8 文字 1 つの文字を複数回連続で使用すると、文字クラスが 1 つ減少します。以下に例を示します。 * **Secret1** には、大文字、小文字、数字の 3 つの文字クラスがあります。 * **Secret111** には、大文字、小文字、数字が含まれていますが、**1** を 繰り返し使用使用したため、ペナルティが -1 で文字クラスが 2 つになりります。	Character classes = 0 必要なクラスのデフォルト数は 0 です。番号を設定するには、**--minclasses** オプションを指定して **ipa pwpolicy-mod** コマンドを実行します。 この表の下に記載されている 重要 の注意事項も併せて参照してください。
Min length	パスワードの最小長。 追加のパスワードポリシーオプション のいずれかが設定されていると、パスワードの最小長は 6 文字です。	Min length = 8 8 文字未満のパスワードは使用できません。
Max failures	IdM がユーザーアカウントをロックするまでのログイン試行の最大失敗数。	Max failures = 6 ユーザーがパスワードを誤って 7 回入力すると、IdM はユーザーアカウントをロックします。
Failure reset interval	失敗したログイン試行回数を IdM がリセットするまでの時間 (秒単位)。	Failure reset interval = 60 **Max failures** で定義されたログイン試行回数が 1 分以上経過すると、ユーザーはユーザーアカウントがロックされる心配なく再ログインを試みることができます。

属性	説明	例
Lockout duration	**Max failures** で定義された回数のログイン試行に失敗した後にユーザーアカウントがロックされる時間 (秒単位)。	Lockout duration = 600 アカウントがロックされると、10 分間ログインできません。

重要

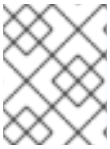
国際文字や記号を使用できないハードウェアセットが各種ある場合には、文字クラス要件に英語と共通記号を使用してください。パスワードの文字クラスポリシーの詳細は、Red Hat ナレッジベースの What characters are valid in a password? を参照してください。

33.3. ANSIBLE PLAYBOOK を使用して IDM にパスワードポリシーを存在させる手順

Ansible Playbook を使用して Identity Management (IdM) にパスワードポリシーを存在させるには、次の手順に従います。

IdM におけるデフォルトの **global_policy** パスワードポリシーでは、パスワード内の異なる文字クラスの数は 0 に設定されています。履歴サイズも 0 に設定されています。

以下の手順に従って、Ansible Playbook を使用して、IdM グループにより強力なパスワードポリシーを適用します。



注記

IdM グループのパスワードポリシーのみを定義できます。個別ユーザーにパスワードポリシーを定義できません。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- IdM 管理者パスワードを把握している。
- IdM にパスワードポリシーが存在することを確認するグループ。

手順

1. **inventory.file** などのインベントリーファイルを作成し、**[ipaserver]** セクションに IdM サーバーの **FQDN** を定義します。

```
[ipaserver]
server.idm.example.com
```

2. Ansible Playbook を作成して、存在させるパスワードポリシーを定義します。この手順は、**/usr/share/doc/ansible-freeipa/playbooks/pwpolicy/pwpolicy_present.yml** ファイルの例をコピーして変更し、簡素化できます。

```
---
- name: Tests
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure presence of pwpolicy for group ops
    ipapwpolicy:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: ops
      minlife: 7
      maxlife: 49
      history: 5
      priority: 1
      lockouttime: 300
      minlength: 8
      minclasses: 4
      maxfail: 3
      failinterval: 5
```

   各変数の意味については、パスワードポリシーの属性 を参照してください。

3. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file
path_to_playbooks_directory_/new_pwpolicy_present.yml
```

Ansible Playbook を使用して、**ops** グループのパスワードポリシーを IdM に存在させることができました。

重要

ops パスワードポリシーの優先度は 1に設定されますが、**global_policy** パスワードポリシーには優先度が設定されません。上記の理由から、**ops** ポリシーは **ops** グループの**global_policy** より自動的に優先され、すぐに有効になります。

global_policy は、ユーザーにポリシーが設定されていない場合のフォールバックポリシーとして機能し、グループポリシーよりも優先されることはありません。

関連情報

- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-pwpolicy.md** ファイルを参照してください。

- Password policy priorities を参照してください。

33.4. IDM での追加のパスワードポリシーオプション

Identity Management (IdM) 管理者は、**libpwquality** 機能セットに基づく追加のパスワードポリシーオプションを有効にすることで、デフォルトのパスワード要件を強化できます。追加のパスワードポリシーオプションには、以下が含まれます。

--maxrepeat

新しいパスワードに使用できる、連続する同一文字数の上限を指定します。

--maxsequence

新しいパスワードにおける単調な文字シーケンスの最大長を指定します。このような配列の例は、12345 または **fedcb** です。このようなパスワードのほとんどは、簡素化チェックに合格しません。

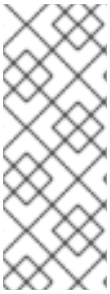
--dictcheck

ゼロ以外の場合は、パスワード (修正可能) が辞書の単語と一致するかどうかを確認します。現在、**libpwquality** は、**cracklib** ライブラリーを使用してディクショナリーの確認を実行しています。

--usercheck

ゼロ以外の場合は、パスワード (修正可能) に、何らかの形式でユーザー名が含まれているかどうかを確認します。ユーザー名が 3 文字より短い場合は実行されません。

既存のパスワードには、追加のパスワードポリシーオプションを適用できません。追加オプションのいずれかを適用すると、IdM は、パスワードの最小文字数である **--minlength** オプションを自動的に 6 文字に設定します。



注記

RHEL 7 サーバーと RHEL 8 サーバーが混在する環境では、RHEL 8.4 以降で実行されているサーバーにのみ追加のパスワードポリシー設定を適用できます。ユーザーが IdM クライアントにログインし、IdM クライアントが RHEL 8.3 以前で実行している IdM サーバーと通信している場合は、システム管理者が設定した新しいパスワードポリシーの要件は適用されません。一貫した動作を確認するには、すべてのサーバーを RHEL 8.4 以降にアップグレードまたは更新します。

関連情報:

- IdM グループへの追加のパスワードポリシーの適用
- **pwquality(3)** man ページ

33.5. IDM グループへの追加のパスワードポリシーオプションの適用

Identity Management (IdM) で追加のパスワードポリシーオプションを適用するには、次の手順に従います。ここでは、新しいパスワードにユーザー名が含まれていないことと、パスワードに同じ文字が連続して 2 文字以内になるようにすることで、**マネージャー** グループのパスワードポリシーを強化する方法を説明します。

前提条件

- IdM 管理者としてログインしている。

- **マネージャー** グループが IdM に存在している。
- **マネージャー** パスワードポリシーが IdM に存在している。

手順

1. **マネージャー** グループのユーザーが提案するすべての新しいパスワードに、ユーザー名の確認を適用します。

```
$ ipa pwpolicy-mod --usercheck=True managers
```

注記

パスワードポリシーの名前を指定しないと、デフォルトの **global_policy** が変更されます。

2. **マネージャー** パスワードポリシーで、同一の連続した文字の上限を 2 に設定します。

```
$ ipa pwpolicy-mod --maxrepeat=2 managers
```

パスワードに、同一の連続した文字が 2 文字を超える場合は、パスワードが使用できなくなります。たとえば、**eR873mUi111YJQ** の組み合わせは、連続して 3 つの **1** を含むため、使用できません。

検証

1. **test_user** という名前のテストユーザーを追加します。

```
$ ipa user-add test_user
First name: test
Last name: user
----------------------------
Added user "test_user"
----------------------------
```

2. テストユーザーを **マネージャー** グループに追加します。
   a. IdM Web UI で、**Identity → Groups → User Groups** をクリックします。
   b. **managers** をクリックします。
   c. **Add** をクリックします。
   d. **Add users into user group 'managers'**ページで、**test_user** をチェックします。
   e. **>** 矢印をクリックして、ユーザーを **Prospective** 列に移動します。
   f. **Add** をクリックします。
3. テストユーザーのパスワードをリセットします。
   a. **Identity → Users** に移動します。
   b. **test_user** をクリックします。

c. **Actions** メニューで、**Reset Password** をクリックします。

d. ユーザーの一時パスワードを入力します。

4. コマンドラインで、**test_user** の Kerberos Ticket-Granting Ticket (TGT) を取得してみてください。

```
$ kinit test_user
```

a. 一時パスワードを入力します。

b. パスワードを変更する必要があることがシステムから通知されます。**test_user** のユーザー名を含むパスワードを入力します。

```
Password expired. You must change it now.
Enter new password:
Enter it again:
Password change rejected: Password not changed.
Unspecified password quality failure while trying to change password.
Please try again.
```

注記

Kerberos には、詳細なエラーパスワードポリシーの報告はなく、特定のケースでは、パスワードが拒否された理由を明確に示していません。

c. 入力したパスワードが拒否されたことがシステムから通知されます。連続して 3 文字以上の同一文字を含むパスワードを入力します。

```
Password change rejected: Password not changed.
Unspecified password quality failure while trying to change password.
Please try again.

Enter new password:
Enter it again:
```

d. 入力したパスワードが拒否されたことがシステムから通知されます。**マネージャー** パスワードポリシーの基準を満たすパスワードを入力します。

```
Password change rejected: Password not changed.
Unspecified password quality failure while trying to change password.
Please try again.

Enter new password:
Enter it again:
```

5. 取得した TGT を表示します。

```
$ klist
Ticket cache: KCM:0:33945
Default principal: test_user@IDM.EXAMPLE.COM
```

```
Valid starting       Expires              Service principal
07/07/2021 12:44:44  07/08/2021 12:44:44
krbtgt@IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
```

マネージャー のパスワードポリシーが、**マネージャー** グループのユーザーに対して正しく機能するようになりました。

関連情報

- IdM での追加のパスワードポリシー

33.6. ANSIBLE PLAYBOOK を使用して追加のパスワードポリシーオプションを IDM グループに適用する

Ansible Playbook を使用して追加のパスワードポリシーオプションを適用し、特定の IdM グループのパスワードポリシー要件を強化できます。この目的には、**maxrepeat**、**maxsequence**、**dictcheck**、および **usercheck** パスワードポリシーオプションを使用できます。この例では、**managers** グループに次の要件を設定する方法を説明します。

- ユーザーの新しいパスワードに、ユーザーのそれぞれのユーザー名が含まれていない。
- パスワードに含まれる連続する同一の文字が 2 文字以下である。
- パスワードに含まれる単調な文字列が 3 文字以内である。これは、システムが **1234** や **abcd** などの文字列を含むパスワードを受け入れないことを意味します。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成した。
  - **ipaadmin_password** を **Secret.yml** Ansible Vault に保存している。
- IdM にパスワードポリシーが存在することを確認するグループ。

手順

1. Ansible Playbook ファイル **manager_pwpolicy_present.yml** を作成して、存在させるパスワードポリシーを定義します。この手順を簡素化するには、次の例をコピーして変更します。

```
---
- name: Tests
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure presence of usercheck and maxrepeat pwpolicy for group managers
    ipapwpolicy:
```

```
    ipaadmin_password: "{{ ipaadmin_password }}"
    name: managers
    usercheck: True
    maxrepeat: 2
    maxsequence: 3
```

2. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file
path_to_playbooks_directory_/manager_pwpolicy_present.yml
```

検証

1. **test_user** という名前のテストユーザーを追加します。

```
$ ipa user-add test_user
First name: test
Last name: user
----------------------------
Added user "test_user"
----------------------------
```

2. テストユーザーを **マネージャー** グループに追加します。
   a. IdM Web UI で、**Identity → Groups → User Groups** をクリックします。
   b. **managers** をクリックします。
   c. **Add** をクリックします。
   d. **Add users into user group 'managers'**ページで、**test_user** をチェックします。
   e. **>** 矢印をクリックして、ユーザーを **Prospective** 列に移動します。
   f. **Add** をクリックします。
3. テストユーザーのパスワードをリセットします。
   a. **Identity → Users** に移動します。
   b. **test_user** をクリックします。
   c. **Actions** メニューで、**Reset Password** をクリックします。
   d. ユーザーの一時パスワードを入力します。
4. コマンドラインで、**test_user** の Kerberos Ticket-Granting Ticket (TGT) を取得してみてください。

```
$ kinit test_user
```

   a. 一時パスワードを入力します。
   b. パスワードを変更する必要があることがシステムから通知されます。**test_user** のユーザー名を含むパスワードを入力します。

```
Password expired. You must change it now.
Enter new password:
Enter it again:
Password change rejected: Password not changed.
Unspecified password quality failure while trying to change password.
Please try again.
```

注記

Kerberos には、詳細なエラーパスワードポリシーの報告はなく、特定のケースでは、パスワードが拒否された理由を明確に示していません。

c. 入力したパスワードが拒否されたことがシステムから通知されます。連続して 3 文字以上の同一文字を含むパスワードを入力します。

```
Password change rejected: Password not changed.
Unspecified password quality failure while trying to change password.
Please try again.

Enter new password:
Enter it again:
```

d. 入力したパスワードが拒否されたことがシステムから通知されます。3 文字を超える単調な文字列を含むパスワードを入力します。たとえば、**1234** や **fedc** などの文字列です。

```
Password change rejected: Password not changed.
Unspecified password quality failure while trying to change password.
Please try again.

Enter new password:
Enter it again:
```

e. 入力したパスワードが拒否されたことがシステムから通知されます。**マネージャー** パスワードポリシーの基準を満たすパスワードを入力します。

```
Password change rejected: Password not changed.
Unspecified password quality failure while trying to change password.
Please try again.

Enter new password:
Enter it again:
```

5. TGT を取得したことを確認します。これは、有効なパスワードを入力した後にのみ可能です。

```
$ klist
Ticket cache: KCM:0:33945
Default principal: test_user@IDM.EXAMPLE.COM

Valid starting       Expires              Service principal
07/07/2021 12:44:44  07/08/2021 12:44:44
krbtgt@IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
```

関連情報

- IdM での追加のパスワードポリシー
- **/usr/share/doc/ansible-freeipa/README-pwpolicy.md**
- **/usr/share/doc/ansible-freeipa/playbooks/pwpolicy**

第34章 パスワード失効に関する通知の管理

ipa-client-epn パッケージに含まれる Expiring Password Notification (EPN) ツールを使用して、設定期間内にパスワードが失効する Identity Management (IdM) ユーザーのリストを構築できます。EPN ツールをインストール、設定、および使用するには、該当のセクションを参照してください。

- Expiring Password Notification (EPN) ツールの概要
- Expiring Password Notification ツールのインストール
- EPN ツールを実行してパスワードが失効するユーザーへのメール送信
- ipa-epn.timer を有効にして、パスワードが失効する全ユーザーへのメールの送信
- Expiring Password Notification (EPN) のメールテンプレートの変更

34.1. EXPIRING PASSWORD NOTIFICATION (EPN) ツールの概要

Expiring Password Notification (EPN) ツールは、設定期間内にパスワードが失効する Identity Management (IdM) ユーザーのリストの作成に使用可能なスタンドアロンツールです。

IdM 管理者は、EPN を使用して以下を行うことができます。

- 対象ユーザーのリストを JSON 形式で表示する。これは、ドライランモードを実行時に作成されます。
- 特定の日または日付の範囲に送信される電子メール数を計算する。
- パスワード期限切れのメール通知をユーザーに送信する。
- **lpa-epn.timer** が EPN ツールを毎日実行し、定義済みの未来の日付範囲内にパスワードが執行するユーザーに対してメールを送信するように設定する。
- メール通知をカスタマイズして、ユーザーに送信する。

注記

ユーザーアカウントが無効な場合には、パスワードが期限切れなってもメール通知は送信されません。

34.2. EXPIRING PASSWORD NOTIFICATION ツールのインストール

Expiring Password Notification (EPN) ツールをインストールするには、次の手順に従います。

前提条件

- スマートホストで設定したローカルの Postfix SMTP サーバーを使用して、Identity Management (IdM) レプリカまたは IdM クライアントに EPN ツールをインストールします。

手順

- EPN ツールをインストールします。

```
# yum install ipa-client-epn
```

34.3. EPN ツールを実行してパスワードが失効するユーザーへのメール送信

Expiring Password Notification (EPN) ツールを実行してパスワードの期限が切れるユーザーにメールを送信するには、次の手順に従います。

注記

EPN ツールはステートレスです。特定の日付にパスワードが失効するユーザーに対してメールの送信に失敗した場合には、EPN ツールには失敗したユーザーのリストは保存されません。

前提条件

- **ipa-client-epn** パッケージがインストールされている。Expiring Password Notification ツールのインストール を参照してください。
- 必要に応じて、**ipa-epn** メールテンプレートをカスタマイズする。期限切れのパスワード通知テンプレートの変更 を参照してください。

手順

1. **epn.conf** 設定ファイルを更新して、今後パスワードが失効するユーザーに通知されるように、EPN ツールのオプションを設定します。

```
# vi /etc/ipa/epn.conf
```

2. 必要に応じて **notify_ttls** を更新します。デフォルトでは、28、14、7、3 および 1 日以内にパスワードが期限切れになるユーザーに通知します。

```
notify_ttls = 28, 14, 7, 3, 1
```

3. SMTP サーバーおよびポートを設定します。

```
smtp_server = localhost
smtp_port = 25
```

4. メールで失効通知を送信するメールアドレスを指定します。配信に失敗したメールは以下のアドレスに返されます。

```
mail_from =admin-email@example.com
```

5. **/etc/ipa/epn.conf** ファイルを保存します。

6. **--dry-run** オプションなしでツールを実行した場合には、EPN ツールをドライランモードで実行し、パスワード失効メールの通知を送信するユーザーのリストを生成します。

```
ipa-epn --dry-run
[
    {
     "uid": "user5",
     "cn": "user 5",
     "krbpasswordexpiration": "2020-04-17 15:51:53",
     "mail": "['user5@ipa.test']"
```

```
    }
]
[
    {
     "uid": "user6",
     "cn": "user 6",
     "krbpasswordexpiration": "2020-12-17 15:51:53",
     "mail": "['user5@ipa.test']"
     }
]
The IPA-EPN command was successful
```

注記

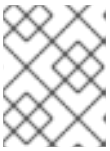
返されたユーザーのリストが非常に大きく、かつ **--dry-run** オプションなしでツールを実行すると、メールサーバーで問題が発生する可能性があります。

7. ドライランモードで EPN ツールを実行時に返された全ユーザーのリストに失効メールを送信するには、**--dry-run** オプションをなしで EPN ツールを実行します。

```
ipa-epn
[
  {
    "uid": "user5",
    "cn": "user 5",
    "krbpasswordexpiration": "2020-10-01 15:51:53",
    "mail": "['user5@ipa.test']"
  }
]
[
  {
    "uid": "user6",
    "cn": "user 6",
    "krbpasswordexpiration": "2020-12-17 15:51:53",
    "mail": "['user5@ipa.test']"
  }
]
The IPA-EPN command was successful
```

8. EPN を監視システムに追加して、**--from-nbdays** および **--to-nbdays** オプションで EPN を呼び出し、特定の時間内に期限切れになるユーザーパスワード数を確認できます。

```
# ipa-epn --from-nbdays 8 --to-nbdays 12
```



注記

--from-nbdays および **--to-nbdays** で EPN ツールを呼び出すと、自動的にドライランモードで実行されます。

検証手順

- EPN ツールを実行し、メール通知が送信されていることを確認します。

関連情報

関連情報

- **ipa-epn** の man ページを参照してください。
- **epn.conf** の man ページを参照してください。

34.4. IPA-EPN.TIMER を有効にして、パスワードが失効する全ユーザーへのメールの送信

ipa-epn.timer を使用して Expiring Password Notification (EPN) ツールを実行し、パスワードの期限が切れるユーザーにメールを送信するには、次の手順に従います。**ipa-epn.timer** は **epn.conf** ファイルを解析し、そのファイルに設定された未来の日付範囲内にパスワードの期限が切れるユーザーにメールを送信します。

前提条件

- **ipa-client-epn** パッケージがインストールされている。 Expiring Password Notification (EPN) ツールのインストール を参照してください。
- 必要に応じて、**ipa-epn** メールテンプレートをカスタマイズする。Expiring Password Notification (EPN) のメールテンプレートの変更 を参照してください。

手順

- **ipa-epn.timer** を起動します。

```
systemctl start ipa-epn.timer
```

タイマーを起動すると、デフォルトでは、EPN ツールは毎日午前 1 時に実行されます。

関連情報

- **ipa-epn** の man ページを参照してください。

34.5. EXPIRING PASSWORD NOTIFICATION (EPN) のメールテンプレートの変更

Expiring Password Notification (EPN) のメールメッセージのテンプレートをカスタマイズするには、次の手順に従います。

前提条件

- **ipa-client-epn** パッケージがインストールされている。

手順

1. EPN メッセージテンプレートを開きます。

```
# vi /etc/ipa/epn/expire_msg.template
```

2. 必要に応じてテンプレートテキストを更新します。

```
Hi {{ fullname }},
```

```
Your password will expire on {{ expiration }}.

Please change it as soon as possible.
```

テンプレートでは以下の変数を使用できます。

- User ID: uid
- Full name: fullname
- First name: first
- Last name: last
- Password expiration date: expiration

3. メッセージテンプレートファイルを保存します。

検証手順

- EPN ツールを実行し、メール通知に更新したテキストが含まれていることを確認します。

関連情報

- **ipa-epn** の man ページを参照してください。

第35章 ID ビューを使用した IDM クライアントのユーザー属性値をオーバーライドする

Identity Management (IdM) ユーザーが、IdM LDAP サーバーに保存されているユーザー属性またはグループ属性 (ログイン名、ホームディレクトリー、認証に使用する証明書、**SSH** 鍵など) をオーバーライドする場合、IdM 管理者は IdM ID ビューを使用して、特定の IdM クライアントの値を再定義できます。たとえば、IdM へのログインに最も一般的に使用される IdM クライアントのユーザーに、別のホームディレクトリーを指定できます。

本章では、クライアントとして IdM に登録されたホストで IdM ユーザーに関連付けられた POSIX 属性値を再定義する方法を説明します。

35.1. ID ビュー

Identity Management (IdM) の ID ビューは、以下の情報を指定する IdM クライアント側のビューです。

- 一元定義された POSIX ユーザーまたはグループ属性の新しい値
- 新しい値が適用されるクライアントホスト。

ID ビューには、1 つ以上の上書きが含まれます。オーバーライドは、一元管理された POSIX 属性値の特定の代替です。

IdM サーバーでは、IdM クライアントに ID ビューのみを定義できます。IdM クライアントにクライアント側の上書きをローカルで設定することはできません。

たとえば、ID ビューを使用して、以下の目的を達成できます。

- 環境ごとに異なる属性値を定義する。たとえば、IdM 管理者または別の IdM ユーザーに、IdM クライアントごとに異なるホームディレクトリーを指定できます。ある IdM クライアントのユーザーのホームディレクトリーとして **/home/encrypted/username** を、別のクライアントでは **/dropbox/username** を設定できます。代わりに、この状況で ID ビューを使用すると便利です。たとえば、クライアントの **/etc/sssd/sssd.conf** ファイルにある **fallback_homedir**、**override_homedir** または他のホームディレクトリー変数を変更すると、すべてのユーザーに影響します。手順例は、IdM クライアントで IdM ユーザーのホームディレクトリーをオーバーライドする ID ビューの追加を参照してください。
- 以前に生成された属性値は、ユーザーの UID の上書きなど、別の値に置き換えます。この機能は、たとえば、IdM ユーザーの UID を 1009 にするなど、通常は LDAP で行うのが困難なシステム全体の変更を実現する場合に便利です。IdM ユーザー UID の生成に使用される IdM ID 範囲は、1000 や 10000 程度の小さい値からは開始されません。IdM ユーザーがすべての IdM クライアントで UID 1009 のローカルユーザーの権限を借用する理由が存在する場合は、ID ビューを使用して、IdM でユーザーが作成したときに生成された IdM ユーザーの UID をオーバーライドできます。

重要

ID ビューは、IdM サーバーではなく、IdM クライアントにのみ適用できます。

関連情報

- Active Directory ユーザーの ID ビューの使用
- SSSD クライアント側のビュー

35.2. ID ビューが SSSD パフォーマンスに対して与える可能性のある悪影響

ID ビューを定義すると、IdM は指定の上書き値を、IdM サーバーの System Security Services Daemon (SSSD) キャッシュに配置します。その後、IdM クライアントで実行している SSSD は、サーバーキャッシュから上書き値を取得します。

特定の最適化と ID ビューを同時に実行できないため、ID ビューを適用すると、SSSD (System Security Services Daemon) のパフォーマンスに悪影響を与える可能性があります。たとえば、ID ビューは、SSSD がサーバー上でグループを検索するプロセスの最適化を防ぎます。

- ID ビューを使用すると、グループ名が上書きされた場合、SSSD は返されたグループメンバー名リストの各メンバーをチェックする必要があります。
- ID ビューを使用しないと、SSSD はグループオブジェクトのメンバー属性からユーザー名だけを収集できます。

SSSD のキャッシュが空の場合や、キャッシュを消去した場合に、この負の影響が現れ、全エントリーが無効になります。

35.3. ID ビューがオーバーライドできる属性

ID ビューは、ユーザーおよびグループ ID のオーバーライドで構成されます。上書きは、新しい POSIX 属性値を定義します。

ユーザー ID およびグループ ID のオーバーライドは、以下の POSIX 属性の新しい値を定義できます。

ユーザー属性

- ログイン名 (**uid**)
- GECOS エントリー (**gecos**)
- UID 番号 (**uidNumber**)
- GID 番号 (**gidNumber**)
- ログインシェル (**loginShell**)
- ホームディレクトリー (**homeDirectory**)
- SSH 公開鍵 (**ipaSshPubkey**)
- 証明書 (**userCertificate**)

グループ属性

- グループ名 (**cn**)
- グループ GID 番号 (**gidNumber**)

35.4. ID ビューのコマンドのヘルプの取得

IdM コマンドラインインターフェイス (CLI) で、Identity Management (IdM) ID ビューに関連するコマンドのヘルプを表示できます。

前提条件

- IdM ユーザーの Kerberos チケットを取得している。

手順

- ID ビューおよびオーバーライドの管理に使用するコマンドをすべて表示するには、次のコマンドを実行します。

```
$ ipa help idviews
ID Views

Manage ID Views

IPA allows to override certain properties of users and groups[...]
[...]
Topic commands:
  idoverridegroup-add          Add a new Group ID override
  idoverridegroup-del          Delete a Group ID override
[...]
```

- 特定のコマンドの詳細なヘルプを表示するには、コマンドに **--help** オプションを追加します。

```
$ ipa idview-add --help
Usage: ipa [global-options] idview-add NAME [options]

Add a new ID View.
Options:
  -h, --help      show this help message and exit
  --desc=STR      Description
[...]
```

35.5. ID ビューを使用して、特定のホストにある IDM ユーザーのログイン名をオーバーライドする

特定の IdM ユーザーに関連付けられた POSIX 属性値をオーバーライドする特定の IdM クライアントの ID ビューを作成するには、次の手順に従います。この手順では、**idm_user** という名前の IdM ユーザーが、**user_1234** ログイン名を使用して **host1** という名前の IdM クライアントにログインできるようにする ID ビューの例を使用します。

前提条件

- IdM 管理者としてログインしている。

手順

1. 新しい ID ビューを作成します。たとえば、**example_for_host1** という名前の ID ビューを作成するには、次のコマンドを実行します。

```
$ ipa idview-add example_for_host1
----------------------------
Added ID View "example_for_host1"
----------------------------
  ID View Name: example_for_host1
```

2. ユーザーの上書きを **example_for_host1** ID ビューに追加します。ユーザーログインをオーバーライドするには、以下を実行します。

   - **ipa idoverrideuser-add** コマンドを入力します。
   - ID ビューの名前を追加します。
   - ユーザー名 (アンカーとも呼ばれます) を追加します。
   - **--login** オプションを追加します。

   ```
   $ ipa idoverrideuser-add example_for_host1 idm_user --login=user_1234
   ----------------------------
   Added User ID override "idm_user"
   ----------------------------
     Anchor to override: idm_user
     User login: user_1234
   ```

   利用可能なオプションのリストは、ipa idoverrideuser-add --help を実行します。

注記

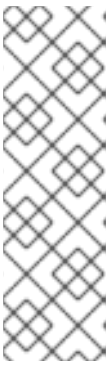
ipa idoverrideuser-add --certificate コマンドは、指定された ID ビューのアカウントの既存証明書をすべて置き換えます。別の証明書を追加するには、代わりに **ipa idoverrideuser-add-cert** コマンドを使用します。

```
$ ipa idoverrideuser-add-cert example_for_host1 user --
certificate="MIIEATCC..."
```

3. 必要に応じて、**ipa idoverrideuser-mod** コマンドを使用すると、既存のユーザーの上書きに新しい属性値を指定できます。

4. **example_for_host1** を **host1.idm.example.com** ホストに適用します。

```
$ ipa idview-apply example_for_host1 --hosts=host1.idm.example.com
----------------------------
Applied ID View "example_for_host1"
----------------------------
hosts: host1.idm.example.com
------------------------------------------
Number of hosts the ID View was applied to: 1
------------------------------------------
```



注記

ipa idview-apply コマンドでは、**--hostgroups** オプションも使用できます。このオプションは、ID ビューを、指定のホストグループに所属するホストに適用しますが、ホストグループ自体との関連付けは行いません。代わりに、**--hostgroups** オプションは指定されたホストグループのメンバーを拡張して、**--hosts** オプションを個別にすべて適用します。

つまり、今後ホストがホストグループに追加されても、ID ビューは新しいホストには適用されません。

5. 新しい設定を **host1.idm.example.com** システムに適用するには、次のコマンドを実行します。

   a. root でシステムに対して SSH 接続します。

   ```
   $ ssh root@host1
   Password:
   ```

   b. SSSD キャッシュを削除します。

   ```
   root@host1 ~]# sss_cache -E
   ```

   c. SSSD デーモンを再起動します。

   ```
   root@host1 ~]# systemctl restart sssd
   ```

検証手順

- **user_1234** の認証情報がある場合は、その認証情報を使用して **host1** で IdM にログインできます。

  1. ログイン名 **user_1234** を使用して **host1** に SSH 接続します。

  ```
  [root@r8server ~]# ssh user_1234@host1.idm.example.com
  Password:

  Last login: Sun Jun 21 22:34:25 2020 from 192.168.122.229
  [user_1234@host1 ~]$
  ```

  2. 作業ディレクトリーを表示します。

  ```
  [user_1234@host1 ~]$ pwd
  /home/idm_user/
  ```

- または、**host1** に root 認証情報がある場合は、その認証情報を使用して **idm_user** および **user_1234** の **id** コマンドの出力を確認できます。

```
[root@host1 ~]# id idm_user
uid=779800003(user_1234) gid=779800003(idm_user) groups=779800003(idm_user)
[root@host1 ~]# user_1234
uid=779800003(user_1234) gid=779800003(idm_user) groups=779800003(idm_user)
```

35.6. IDM ID ビューの変更

Identity Management (IdM) の ID ビューは、特定の IdM ユーザーに関連する POSIX 属性値をオーバーライドします。既存の ID ビューを変更するには、次の手順に従ってください。具体的には、**idm_user** という名前のユーザーが IdM クライアントの **host1.idm.example.com** の **/home/idm_user/** ではなく、ユーザーのホームディレクトリーとして **/home/user_1234/** ディレクトリーを使用できるように ID ビューを変更する方法を説明します。

前提条件

- **host1.idm.example.com** への root アクセス権限がある。

- **admin** などの必要な権限を持つユーザーとしてログインしている。
- IdM クライアント **host1** に適用される **idm_user** の ID ビューが設定されている。

手順

1. root で、**idm_user** がユーザーのホームディレクトリーとして **host1.idm.example.com** で使用するディレクトリーを作成します。

```
[root@host1 /]# mkdir /home/user_1234/
```

2. ディレクトリーの所有権を変更します。

```
[root@host1 /]# chown idm_user:idm_user /home/user_1234/
```

3. ID ビューが現在適用されているホストを含め、ID ビューを表示します。**example_for_host1** という名前の ID ビューを表示するには、次のコマンドを実行します。

```
$ ipa idview-show example_for_host1 --all
  dn: cn=example_for_host1,cn=views,cn=accounts,dc=idm,dc=example,dc=com
  ID View Name: example_for_host1
  User object override: idm_user
  Hosts the view applies to: host1.idm.example.com
  objectclass: ipaIDView, top, nsContainer
```

この出力は、現在 ID ビューが **host1.idm.example.com** に適用されることを示しています。

4. **example_for_host1** ID ビューのユーザーのオーバーライドを変更します。ユーザーのホームディレクトリーをオーバーライドするには、次のコマンドを実行します。
   - **ipa idoverrideuser-add** コマンドを入力します。
   - ID ビューの名前を追加します。
   - ユーザー名 (アンカーとも呼ばれます) を追加します。
   - **--homedir** オプションを追加します。

```
$ ipa idoverrideuser-mod example_for_host1 idm_user --
homedir=/home/user_1234
-----------------------------
Modified a User ID override "idm_user"
-----------------------------
  Anchor to override: idm_user
  User login: user_1234
  Home directory: /home/user_1234/
```

利用可能なオプションのリストは、**ipa idoverrideuser-mod --help** を実行します。

5. 新しい設定を **host1.idm.example.com** システムに適用するには、次のコマンドを実行します。
   a. root でシステムに対して SSH 接続します。

```
$ ssh root@host1
Password:
```

b. SSSD キャッシュを削除します。

```
root@host1 ~]# sss_cache -E
```

c. SSSD デーモンを再起動します。

```
root@host1 ~]# systemctl restart sssd
```

検証手順

1. **idm_user** として **host1** に **SSH** 接続します。

```
[root@r8server ~]# ssh idm_user@host1.idm.example.com
Password:

Last login: Sun Jun 21 22:34:25 2020 from 192.168.122.229
[user_1234@host1 ~]$
```

2. 作業ディレクトリーを出力します。

```
[user_1234@host1 ~]$ pwd
/home/user_1234/
```

関連情報

- デフォルト信頼ビューの変更による AD ユーザーのグローバル属性の定義

35.7. IDM クライアントで IDM ユーザーのホームディレクトリーをオーバーライドする ID ビューの追加

Identity Management (IdM) の ID ビューは、特定の IdM ユーザーに関連する POSIX 属性値をオーバーライドします。この手順では、**host1** という名前の IdM クライアントの **idm_user** に適用される ID ビューを作成し、ユーザーが **/home/idm_user/** ではなく、ユーザーホームディレクトリーとして **/home/user_1234/** ディレクトリーを使用できるようにします。

前提条件

- **host1.idm.example.com** への root アクセス権限がある。
- **admin** などの必要な権限を持つユーザーとしてログインしている。

手順

1. root で、**idm_user** がユーザーのホームディレクトリーとして **host1.idm.example.com** で使用するディレクトリーを作成します。

```
[root@host1 /]# mkdir /home/user_1234/
```

2. ディレクトリーの所有権を変更します。

```
[root@host1 /]# chown idm_user:idm_user /home/user_1234/
```

3. ID ビューを作成します。たとえば、**example_for_host1** という名前の ID ビューを作成するには、次のコマンドを実行します。

```
$ ipa idview-add example_for_host1
---------------------------
Added ID View "example_for_host1"
---------------------------
  ID View Name: example_for_host1
```

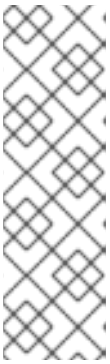
4. ユーザーの上書きを **example_for_host1** ID ビューに追加します。ユーザーのホームディレクトリーをオーバーライドするには、次のコマンドを実行します。

   - **ipa idoverrideuser-add** コマンドを入力します。
   - ID ビューの名前を追加します。
   - ユーザー名 (アンカーとも呼ばれます) を追加します。
   - **--homedir** オプションを追加します。

```
$ ipa idoverrideuser-add example_for_host1 idm_user --homedir=/home/user_1234
----------------------------
Added User ID override "idm_user"
----------------------------
  Anchor to override: idm_user
  Home directory: /home/user_1234/
```

5. **example_for_host1** を **host1.idm.example.com** ホストに適用します。

```
$ ipa idview-apply example_for_host1 --hosts=host1.idm.example.com
----------------------------
Applied ID View "example_for_host1"
----------------------------
hosts: host1.idm.example.com
-----------------------------------------
Number of hosts the ID View was applied to: 1
-----------------------------------------
```



注記

ipa idview-apply コマンドでは、**--hostgroups** オプションも使用できます。このオプションは、ID ビューを、指定のホストグループに所属するホストに適用しますが、ホストグループ自体との関連付けは行いません。代わりに、**--hostgroups** オプションは指定されたホストグループのメンバーを拡張して、**--hosts** オプションを個別にすべて適用します。

つまり、今後ホストがホストグループに追加されても、ID ビューは新しいホストには適用されません。

6. 新しい設定を **host1.idm.example.com** システムに適用するには、次のコマンドを実行します。

   a. root でシステムに対して SSH 接続します。

```
$ ssh root@host1
Password:
```

b. SSSD キャッシュを削除します。

```
root@host1 ~]# sss_cache -E
```

c. SSSD デーモンを再起動します。

```
root@host1 ~]# systemctl restart sssd
```

検証手順

1. **idm_user** として **host1** に **SSH** 接続します。

```
[root@r8server ~]# ssh idm_user@host1.idm.example.com
Password:
Activate the web console with: systemctl enable --now cockpit.socket

Last login: Sun Jun 21 22:34:25 2020 from 192.168.122.229
[idm_user@host1 /]$
```

2. 作業ディレクトリーを出力します。

```
[idm_user@host1 /]$ pwd
/home/user_1234/
```

関連情報

- IdM クライアントの AD ユーザーのデフォルト信頼ビューの属性を ID ビューでオーバーライドする。

35.8. IDM ホストグループへの ID ビューの適用

ipa idview-apply コマンドでは、**--hostgroups** オプションを使用できます。ただし、このオプションは、指定のホストに現在所属するホストに ID ビューを適用する 1 回限りの操作として機能しますが、ホストグループ自体と ID ビューを動的に関連付けることはありません。**--hostgroups** オプションは、指定したホストグループのメンバーを拡張して、**--hosts** オプションを個別にすべて適用します。

新しいホストを後でホストグループに追加する場合は、**--hosts** オプションで **ipa idview-apply** コマンドを使用して、新しいホストに ID ビューを手動で適用する必要があります。

同様に、ホストグループからホストを削除すると、ID ビューは削除後でも、ホストに割り当てられます。削除されたホストから ID ビューの適用を解除するには、**ipa idview-unapply id_view_name --hosts=name_of_the_removed_host** コマンドを実行する必要があります。

次の目標を達成するには、次の手順に従ってください。

1. ホストグループを作成し、そのグループにホストを追加する方法。
2. ID ビューをホストグループに適用する方法。
3. ホストグループに新規ホストを追加し、ID ビューを新しいホストに適用する方法。

前提条件

- ホストグループに適用する ID ビューが IdM に存在することを確認します。たとえば、ID ビューを作成して、特定の IdM クライアントで IdM ユーザーログイン名をオーバーライドするには、ID ビューを使用して、特定のホストにある IdM ユーザーのログイン名をオーバーライドする を参照してください。

手順

1. ホストグループを作成し、そのグループにホストを追加します。

   a. ホストグループを作成します。たとえば、**baltimore** という名前のホストグループを作成するには、次のコマンドを実行します。

   ```
   [root@server ~]# ipa hostgroup-add --desc="Baltimore hosts" baltimore
   ---------------------------
   Added hostgroup "baltimore"
   ---------------------------
   Host-group: baltimore
   Description: Baltimore hosts
   ```

   b. ホストグループにホストを追加します。たとえば、**host102** および **host103** を **baltimore** ホストグループに追加するには、次のコマンドを実行します。

   ```
   [root@server ~]# ipa hostgroup-add-member --hosts={host102,host103} baltimore
   Host-group: baltimore
   Description: Baltimore hosts
   Member hosts: host102.idm.example.com, host103.idm.example.com
   -------------------------
   Number of members added 2
   -------------------------
   ```

2. ホストグループのホストに ID ビューを適用します。たとえば、**example_for_host1** ID ビューを **baltimore** ホストグループに適用するには、次のコマンドを実行します。

   ```
   [root@server ~]# ipa idview-apply --hostgroups=baltimore
   ID View Name: example_for_host1
   ----------------------------------------
   Applied ID View "example_for_host1"
   ----------------------------------------
     hosts: host102.idm.example.com, host103.idm.example.com
   ------------------------------------------------
   Number of hosts the ID View was applied to: 2
   ------------------------------------------------
   ```

3. ホストグループに新規ホストを追加し、ID ビューを新しいホストに適用します。

   a. ホストグループに新規ホストを追加します。たとえば、**somehost.idm.example.com** ホストを **baltimore** ホストグループに追加するには、次のコマンドを実行します。

   ```
   [root@server ~]# ipa hostgroup-add-member --hosts=somehost.idm.example.com
   baltimore
     Host-group: baltimore
     Description: Baltimore hosts
     Member hosts:  host102.idm.example.com,
   ```

```
  host103.idm.example.com,somehost.idm.example.com
  -------------------------
  Number of members added 1
  -------------------------
```

b. 必要に応じて、ID ビュー情報を表示します。たとえば、**example_for_host1** ID ビューの詳細を表示するには、次のコマンドを実行します。

```
[root@server ~]# ipa idview-show example_for_host1 --all
  dn: cn=example_for_host1,cn=views,cn=accounts,dc=idm,dc=example,dc=com
  ID View Name: example_for_host1
[...]
  Hosts the view applies to: host102.idm.example.com, host103.idm.example.com
  objectclass: ipaIDView, top, nsContainer
```

この出力では、ID ビューが **baltimore** ホストグループに新規追加されたホスト **somehost.idm.example.com** に適用されていないことが分かります。

c. ID ビューを新規ホストに適用します。たとえば、ID ビュー **example_for_host1** を **somehost.idm.example.com** に適用するには、次のコマンドを実行します。

```
[root@server ~]# ipa idview-apply --host=somehost.idm.example.com
ID View Name: example_for_host1
-----------------------------------------
Applied ID View "example_for_host1"
-----------------------------------------
  hosts: somehost.idm.example.com
-------------------------------------------
Number of hosts the ID View was applied to: 1
-------------------------------------------
```

検証手順

- ID ビュー情報を再度表示します。

```
[root@server ~]# ipa idview-show example_for_host1 --all
  dn: cn=example_for_host1,cn=views,cn=accounts,dc=idm,dc=example,dc=com
  ID View Name: example_for_host1
[...]
  Hosts the view applies to: host102.idm.example.com, host103.idm.example.com,
somehost.idm.example.com
  objectclass: ipaIDView, top, nsContainer
```

この出力は、ID ビューが **baltimore** ホストグループに新規追加されたホスト **somehost.idm.example.com** に適用されていることを示しています。

35.9. ANSIBLE を使用して、特定のホスト上の IDM ユーザーのログイン名とホームディレクトリーをオーバーライドする

idoverrideuser ansible-freeipa モジュールを使用して、特定のアイデンティティー Management (IdM) ユーザーに関連付けられた POSIX 属性値をオーバーライドする特定の IdM クライアントの ID ビューを作成するには、この手順を実行します。この手順では、idm_user という名前の IdM ユーザー

が、**user_1234** ログイン名を使用して host1 という名前の IdM クライアントにログインできるようにする ID ビューの例を使用します。さらに、ID ビューは idm_user のホームディレクトリーを変更するので、host1 にログインした後、ユーザーのホームディレクトリーは **/home/user_1234/** になります。

前提条件

- コントロールノードでは、
  - Ansible バージョン 2.14 以降を使用している。
  - **ansible-freeipa** パッケージをインストールしている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成した。
  - RHEL 8.10 以降を使用しています。
  - **ipaadmin_password** を **Secret.yml** Ansible Vault に保存している。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. 次の内容を含む Ansible Playbook ファイル **add-idoverrideuser-with-name-and-homedir.yml** を作成します。

```
---
- name: Playbook to manage idoverrideuser
  hosts: ipaserver
  become: false
  gather_facts: false
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml

  tasks:
  - name: Ensure idview_for_host1 is present
    idview:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: idview_for_host1
  - name: Ensure idview_for_host1 is applied to host1.idm.example.com
    idview:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: idview_for_host1
      host: host1.idm.example.com
      action: member
  - name: Ensure idm_user is present in idview_for_host1 with homedir /home/user_1234
and name user_1234
    ipaidoverrideuser:
      ipaadmin_password: "{{ ipaadmin_password }}"
      idview: idview_for_host1
      anchor: idm_user
      name: user_1234
      homedir: /home/user_1234
```

2. Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
<path_to_inventory_directory>/inventory <path_to_playbooks_directory>/add-
idoverrideuser-with-name-and-homedir.yml
```

3. [オプション] **ルート** 認証情報を持っている場合は、新しい設定を **host1.idm.example.com** システムにすぐに適用できます。

   a. root でシステムに対して SSH 接続します。

```
$ ssh root@host1
Password:
```

   b. SSSD キャッシュを削除します。

```
root@host1 ~]# sss_cache -E
```

   c. SSSD デーモンを再起動します。

```
root@host1 ~]# systemctl restart sssd
```

検証

1. **idm_user** として **host1** に **SSH** 接続します。

```
[root@r8server ~]# ssh idm_user@host1.idm.example.com
Password:

Last login: Sun Jun 21 22:34:25 2020 from 192.168.122.229
[user_1234@host1 ~]$
```

2. 作業ディレクトリーを出力します。

```
[user_1234@host1 ~]$ pwd
/home/user_1234/
```

関連情報

- **ansible-freeipa** アップストリームドキュメントの idoverrideuser モジュール

35.10. ANSIBLE を使用して、IDM クライアントで SSH キーログインを有効にする ID ビューを設定する

idoverrideuser ansible-freeipa モジュールを使用して、IdM ユーザーが特定の SSH キーを使用して特定の IdM クライアントにログインできるようにするには、この手順を実行します。この手順では、**idm_user** という名前の IdM ユーザーが SSH キーを使用して **host1.idm.example.com** という名前の IdM クライアントにログインできるようにする ID ビューの例を使用します。

注記

この ID ビューは、特定の HBAC ルールを強化するために使用できます。

前提条件

- コントロールノードでは、
  - Ansible バージョン 2.14 以降を使用している。
  - **ansible-freeipa** パッケージをインストールしている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成した。
  - RHEL 8.10 以降を使用しています。
  - **ipaadmin_password** を **Secret.yml** Ansible Vault に保存している。
- **idm_user** の SSH 公開鍵にアクセスできます。
- **idview_for_host1** ID ビューが存在します。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. 次の内容を含む Ansible Playbook ファイル **Ensure-idoverrideuser-can-login-with-sshkey.yml** を作成します。

```
---
- name: Playbook to manage idoverrideuser
  hosts: ipaserver
  become: false
  gather_facts: false
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml

  tasks:
  - name: Ensure test user idm_user is present in idview idview_for_host1 with sshpubkey
    ipaidoverrideuser:
      ipaadmin_password: "{{ ipaadmin_password }}"
      idview: idview_for_host1
      anchor: idm_user
      sshpubkey:
      - ssh-rsa AAAAB3NzaC1yc2EAAADAQABAAABgQCqmVDpEX5gnSjKuv97Ay ...
  - name: Ensure idview_for_host1 is applied to host1.idm.example.com
    ipaidview:
      ipaadmin_password:  "{{ ipaadmin_password }}"
      name: idview_for_host1
      host: host1.idm.example.com
      action: member
```

2. Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
<path_to_inventory_directory>/inventory <path_to_playbooks_directory>/ensure-
idoverrideuser-can-login-with-sshkey.yml
```

3. [オプション] **ルート** 認証情報を持っている場合は、新しい設定を **host1.idm.example.com** システムにすぐに適用できます。

   a. root でシステムに対して SSH 接続します。

   ```
   $ ssh root@host1
   Password:
   ```

   b. SSSD キャッシュを削除します。

   ```
   root@host1 ~]# sss_cache -E
   ```

   c. SSSD デーモンを再起動します。

   ```
   root@host1 ~]# systemctl restart sssd
   ```

検証

- 公開鍵を使用して **host1** に **SSH 接続** します。

```
[root@r8server ~]# ssh -i ~/.ssh/id_rsa.pub idm_user@host1.idm.example.com

Last login: Sun Jun 21 22:34:25 2023 from 192.168.122.229
[idm_user@host1 ~]$
```

出力により、正常にログインしたことが確認されます。

関連情報

- **ansible-freeipa** アップストリームドキュメントの idoverrideuser モジュール

35.11. ANSIBLE を使用して、IDM クライアント上のローカルサウンドカードへのユーザー ID オーバーライドアクセスを付与する

ansible-freeipa グループ および **idoverrideuser** モジュールを使用して、アイデンティティー Management (IdM) または Active Directory (AD) ユーザーを IdM クライアント上のローカル **オーディオ** グループのメンバーにすることができます。これにより、IdM または AD ユーザーにホスト上のサウンドカードへの特権アクセスが付与されます。この手順では、最初の Playbook タスクで **aduser@addomain.com** ID オーバーライドが追加された **デフォルトの信頼ビュー** ID ビューの例を使用します。次の Playbook タスクでは、RHEL ホスト上のローカル **オーディオ** グループの GID に対応する GID 63 の **オーディオ** グループが IdM に作成されます。同時に、 aduser@addomain.com ID オーバーライドが IdM オーディオグループにメンバーとして追加されます。

前提条件

- 手順の最初の部分を実行する IdM クライアントへの **ルート** アクセス権を持っていること。この例では、**client.idm.example.com** です。

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - RHEL 8.10 以降を使用しています。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- AD フォレストは IdM と信頼関係にあります。この例では、AD ドメインの名前は **addomain.com** であり、ローカル **オーディオ** グループでの存在が保証されている AD ユーザーの完全修飾ドメイン名 (FQDN) は **aduser@addomain.com** です。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **client.idm.example.com** で、**/etc/nsswitch.conf** ファイルに **[SUCCESS=merge]** を追加します。

```
[...]
# Allow initgroups to default to the setting for group.
initgroups: sss [SUCCESS=merge] files
```

2. ローカルの **audio** グループの GID を確認します。

```
$ getent group audio
---------------------
audio:x:63
```

3. Ansible コントロールノードで、**aduser@addomain.com** ユーザーオーバーライドをデフォルトの信頼ビューに追加するタスクを含む **add-aduser-to-audio-group.yml** Playbook を作成します。

```
---
- name: Playbook to manage idoverrideuser
  hosts: ipaserver
  become: false

  tasks:
  - name: Add aduser@addomain.com user to the Default Trust View
    ipaidoverrideuser:
      ipaadmin_password: "{{ ipaadmin_password }}"
      idview: "Default Trust View"
      anchor: aduser@addomain.com
```

4. 同じ Playbook 内の別の Playbook タスクを使用して、グループ **オーディオをGID** 63 で IdM に追加します。aduser idoverrideuser をグループに追加します。

```
- name: Add the audio group with the aduser member and GID of 63
  ipagroup:
    ipaadmin_password: "{{ ipaadmin_password }}"
    name: audio
    idoverrideuser:
    - aduser@addomain.com
    gidnumber: 63
```

5. ファイルを保存します。

6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory add-aduser-to-audio-group.yml
```

検証

1. AD ユーザーとして IdM クライアントにログインします。

```
$ ssh aduser@addomain.com@client.idm.example.com
```

2. AD ユーザーのグループメンバーシップを確認します。

```
$ id aduser@addomain.com
uid=702801456(aduser@addomain.com) gid=63(audio) groups=63(audio)
```

関連情報

- idoverrideuser および ipagroup **ansible-freeipa** アップストリームドキュメント
- IdM でのローカルグループとリモートグループのグループマージの有効化

35.12. ANSIBLE を使用して、IDM ユーザーが特定の UID を持つ ID ビューに存在することを確認する

独自のコンピューターがあるラボで作業しているが、**/home/** ディレクトリーがサーバーによってエクスポートされた共有ドライブ内にある場合は、次の 2 人のユーザーを持つことができます。

- システム全体のユーザーであり、アイデンティティー Management (IdM) に集中的に保存されます。
- アカウントがローカルであり、問題のシステムに保存されているもの。

IdM ユーザーとしてログインしている場合でも、ローカルユーザーとしてログインしている場合でも、ファイルに完全にアクセスする必要がある場合は、両方のユーザーに同じ **UID を** 付与することでそれが可能になります。

ansible-freeipa idoverrideuser モジュールを使用して次の操作を行うには、この手順を完了します。

- **idview_for_host01** という名前の ID ビューを host01 に適用します。

- idview_for_host01 で、**UID が20001** の idm_user のユーザー ID オーバーライドが存在することを確認します。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - RHEL 8.10 以降を使用しています。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **idview_for_host1** ID ビューが存在します。

手順

1. Ansible コントロールノードで、次の内容の **Ensure-idmuser-and-local-user-have-access-to-same-files.yml** Playbook を作成します。

```
---
- name: Ensure both local user and IdM user have access to same files
  hosts: ipaserver
  become: false
  gather_facts: false

  tasks:
  - name: Ensure idview_for_host1 is applied to host1.idm.example.com
    ipaidview:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: idview_for_host01
      host: host1.idm.example.com
  - name: Ensure idmuser is present in idview_for_host01 with the UID of 20001
    ipaidoverrideuser:
      ipaadmin_password: "{{ ipaadmin_password }}"
      idview: idview_for_host01
      anchor: idm_user
      UID: 20001
```

2. ファイルを保存します。
3. Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory ensure-idmuser-and-local-user-have-access-to-same-files.yml
```

関連情報

- **ansible-freeipa** アップストリームドキュメントの idoverrideuser モジュール

35.13. ANSIBLE を使用して、IDM ユーザーが 2 つの証明書を使用して IDM クライアントにログインできるようにする

通常、パスワードを使用して IdM にログインするアイデンティティー Management (IdM) ユーザーが、スマートカードのみを使用して特定の IdM クライアントに認証されるようにする場合は、そのクライアント上のユーザーの認証を要求する ID ビューを作成できます。

ansible-freeipa idoverrideuser モジュールを使用して次の操作を行うには、この手順を完了します。

- **idview_for_host01** という名前の ID ビューを host01 に適用します。
- idview_for_host01 で、2 つの証明書を持つ **idm_user** のユーザー ID オーバーライドが存在することを確認します。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - RHEL 8.10 以降を使用しています。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
  - この例では、**cert1.b64** および **cert2.b64** 証明書が、Playbook を実行しているディレクトリーと同じディレクトリーにあることを前提としています。
- **idview_for_host01** ID ビューが存在します。

手順

1. Ansible コントロールノードで、次の内容の **Ensure-idmuser-present-in-idview-with-certificates.yml** Playbook を作成します。

```
---
- name: Ensure both local user and IdM user have access to same files
  hosts: ipaserver
  become: false
  gather_facts: false

  tasks:
  - name: Ensure idview_for_host1 is applied to host01.idm.example.com
    ipaidview:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: idview_for_host01
      host: host01.idm.example.com
```

```
- name: Ensure an IdM user is present in ID view with two certificates
  ipaidoverrideuser:
    ipaadmin_password: "{{ ipaadmin_password }}"
    idview: idview_for_host01
    anchor: idm_user
    certificate:
    - "{{ lookup('file', 'cert1.b64', rstrip=False) }}"
    - "{{ lookup('file', 'cert2.b64', rstrip=False) }}"
```

rstrip=False ディレクティブにより、検索されたファイルの末尾から空白が削除されません。

2. ファイルを保存します。

3. Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory ensure-idmuser-present-in-idview-with-certificates.yml
```

関連情報

- **ansible-freeipa** アップストリームドキュメントの idoverrideuser モジュール

35.14. ANSIBLE を使用して IDM グループに IDM クライアントのサウンドカードへのアクセス権を付与する

ansible-freeipa idview および **idoverridegroup** モジュールを使用して、アイデンティティー Management (IdM) または Active Directory (AD) ユーザーを IdM クライアント上のローカル **オーディオ** グループのメンバーにすることができます。これにより、IdM または AD ユーザーにホスト上のサウンドカードへの特権アクセスが付与されます。

この手順では、RHEL ホスト上のローカル **オーディオ** グループの GID に対応する **GID 63** で **オーディオ** グループ ID オーバーライドが追加された **idview_for_host01** ID ビューの例を使用します。**idview_for_host01** ID ビューは、**host01.idm.example.com** という名前の IdM クライアントに適用されます。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - RHEL 8.10 以降を使用しています。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。

手順

1. [オプション] RHEL ホスト上のローカル **オーディオ** グループの GID を識別します。

```
$ getent group audio
--------------------
audio:x:63
```

2. Ansible コントロールノードで、次のタスクを含む **give-idm-group-access-to-sound-card-on-idm-client.yml** Playbook を作成します。

```
---
- name: Playbook to give IdM group access to sound card on IdM client
  hosts: ipaserver
  become: false

  tasks:
  - name: Ensure the audio group exists in IdM
    ipagroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: audio

  - name: Ensure idview_for_host01 exists and is applied to host01.idm.example.com
    ipaidview:
      ipaadmin_password:  ”{{ ipaadmin_password }}"
      name: idview_for_host01
      host: host01.idm.example.com

  - name: Add an override for the IdM audio group with GID 63 to idview_for_host01
    ipaidoverridegroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      idview: idview_for_host01
      anchor: audio
      GID: 63
```

3. ファイルを保存します。

4. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory give-idm-group-access-to-sound-card-on-idm-client.yml
```

検証

1. IdM クライアントで、IdM 管理者の認証情報を取得します。

```
$ kinit admin
Password:
```

2. テスト IdM ユーザーを作成します。

```
$ ipa user-add testuser --first test --last user --password
User login [tuser]:
Password:
Enter Password again to verify:
```

```
------------------
Added user "tuser"
------------------
```

3. ユーザーを IdM オーディオグループに追加します。

```
$ ipa group-add-member --tuser audio
```

4. tuser として host01.idm.example.com にログインします。

```
$ ssh tuser@host01.idm.example.com
```

5. ユーザーのグループメンバーシップを確認します。

```
$ id tuser
uid=702801456(tuser) gid=63(audio) groups=63(audio)
```

関連情報

- idoverridegroup、idview、ipagroup **ansible-freeipa** アップストリームドキュメント
- IdM でのローカルグループとリモートグループのグループマージの有効化

35.15. NIS ドメインの IDENTITY MANAGEMENT への移行

NIS ドメインを IdM に移行する際に、ファイルやディレクトリーのパーミッションを変更しないように、ID ビューを使用して既存のホストにホスト固有の UID と GID を設定することができます。

前提条件

- **kinit admin** コマンドを使用して、管理者として自分自身を認証済みである。

手順

1. IdM ドメインにユーザーとグループを追加します。
   a. **ipa user-add** コマンドを使用して、ユーザーを作成します。詳細は、IdM へのユーザーの追加 を参照してください。
   b. **ipa group-add** コマンドを使用して、グループを作成します。詳細は、IdM へのグループの追加 を参照してください。
2. ユーザーの作成中に生成された IDs IdM をオーバーライドします。
   a. **ipa idview-add** コマンドを使用して、新しい ID ビューを作成します。詳細は、ID ビューコマンドのヘルプの取得 を参照してください。
   b. **ipa idoverrideuser-add** および **idoverridegroup-add** をそれぞれ使用して、ユーザーとグループの ID オーバーライドを ID ビューに追加します。
3. **ipa idview-apply** コマンドを使用して、特定のホストに ID ビューを割り当てます。
4. NIS ドメインの使用を停止します。

検証

検証

1. すべてのユーザーとグループが ID ビューに正しく追加されたかどうかを確認するには、**ipa idview-show** コマンドを使用します。

```
$ ipa idview-show example-view
  ID View Name: example-view
  User object overrides: example-user1
  Group object overrides: example-group
```

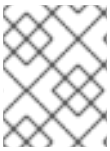
第36章 ACTIVE DIRECTORY ユーザーの ID ビューの使用

IdM-AD 信頼環境において、Active Directory (AD) ユーザーの POSIX 属性に新しい値を指定するために、ID ビューを使用することができます。

デフォルトでは、IdM はすべての AD ユーザーに**デフォルト信頼ビュー** を適用します。個々の IdM クライアントで追加の ID ビューを設定することで、特定のユーザーが受け取る POSIX 属性をさらに調整することができます。

36.1. デフォルト信頼ビューの仕組み

デフォルト信頼ビュー、信頼ベースのセットアップで AD ユーザーとグループに常に適用されるデフォルトの ID ビューです。これは、**ipa-adtrust-install** コマンドを使用して信頼を確立する際に自動的に作成され、削除することはできません。



注記

デフォルト信頼ビューは AD ユーザーおよびグループのオーバーライドのみを受け入れ、IdM ユーザーおよびグループのオーバーライドは受け入れません。

Default Trust View を使用すると、AD ユーザーおよびグループのカスタム POSIX 属性を定義できます。これにより、AD で定義された値を上書きできます。

表36.1 デフォルト信頼ビューの適用

	AD の値	デフォルトの信頼ビュー	結果
Login	ad_user	ad_user	ad_user
UID	111	222	222
GID	111	(値なし)	111

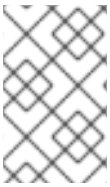
追加の ID ビューを設定して、IdM クライアントのデフォルト信頼ビューをオーバーライドすることもできます。IdM は、デフォルト信頼ビューの上に、ホスト固有の ID ビューからの値を適用します。

- ホスト固有の ID ビューに属性が定義されている場合、IdM はこの ID ビューの値を適用します。
- ホスト固有の ID ビューで属性が定義されていない場合、IdM はデフォルト信頼ビューからの値を適用します。

表36.2 デフォルト信頼ビューの上にホスト固有の ID ビューを適用する

	AD の値	デフォルトの信頼ビュー	ホスト固有の ID ビュー	結果
Login	ad_user	ad_user	(値なし)	ad_user
UID	111	222	333	333

	AD の値	デフォルトの信頼ビュー	ホスト固有の ID ビュー	結果
GID	111	(値なし)	333	333



注記

ホスト固有の ID ビューのみを適用して、IdM クライアントのデフォルト信頼ビューをオーバーライドできます。IdM サーバーとレプリカは、常にデフォルト信頼ビューの値を適用します。

関連情報

- ID ビューを使用した IdM クライアントのユーザー属性値をオーバーライドする

36.2. デフォルト信頼ビューの変更による AD ユーザーのグローバル属性の定義

Active Directory (AD) ユーザーの POSIX 属性を IdM デプロイメント全体を通じてオーバーライドしたい場合は、デフォルト信頼ビューでそのユーザーのエントリーを変更します。この手順では、AD ユーザー **ad_user@ad.example.com** の GID を 732000006 に設定します。

前提条件

- IdM 管理者として認証されている。
- グループが GID とともに存在するか、グループの ID オーバーライドで GID を設定する必要があります。

手順

1. IdM 管理者として、GID 番号を 732000006 に変更する AD ユーザーの ID オーバーライドをデフォルト信頼ビューに作成してください。

```
# ipa idoverrideuser-add 'Default Trust View' ad_user@ad.example.com --
gidnumber=732000006
```

2. すべての IdM サーバーとクライアントの SSSD キャッシュから **ad_user@ad.example.com** ユーザーのエントリーをクリアします。これにより、古いデータが削除され、新しいオーバーライド値が適用されるようになります。

```
# sssctl cache-expire -u ad_user@ad.example.com
```

検証

- **ad_user@ad.example.com** ユーザーの情報を取得して、GID が更新された値を反映することを確認します。

```
# id ad_user@ad.example.com
uid=702801456(ad_user@ad.example.com) gid=732000006(ad_admins)
groups=732000006(ad_admins),702800513(domain users@ad.example.com)
```

36.3. IDM クライアントの AD ユーザーのデフォルト信頼ビューの属性を ID ビューでオーバーライドする。

Active Directory (AD) ユーザーのデフォルト信頼ビューから、いくつかの POSIX 属性をオーバーライドすることもできます。たとえば、ある特定の IdM クライアントで AD ユーザーに異なる GID を与える必要がある場合があります。ID ビューを使用して、AD ユーザーのデフォルト信頼ビューの値をオーバーライドし、単一のホストにそれを適用することができます。この手順では、**host1.idm.example.com** IdM クライアントの **ad_user@ad.example.com** AD ユーザーの GID を 732001337 に設定する方法を説明します。

前提条件

- **host1.idm.example.com** IdM クライアントへの root アクセス権限がある。
- 必要な権限を持つユーザー (例:**admin** ユーザー) としてログインしている。

手順

1. ID ビューを作成します。たとえば、**example_for_host1** という名前の ID ビューを作成するには、次のコマンドを実行します。

```
$ ipa idview-add example_for_host1
----------------------------
Added ID View "example_for_host1"
----------------------------
  ID View Name: example_for_host1
```

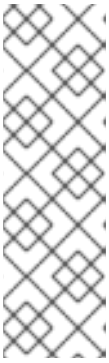
2. ユーザーの上書きを **example_for_host1** ID ビューに追加します。ユーザーの GID をオーバーライドするには、以下を実行します。
   - **ipa idoverrideuser-add** コマンドを入力します。
   - ID ビューの名前を追加します。
   - ユーザー名 (アンカーとも呼ばれます) を追加します。
   - **--gidnumber=** オプションを追加します。

```
$ ipa idoverrideuser-add example_for_host1 ad_user@ad.example.com --
gidnumber=732001337
----------------------------
Added User ID override "ad_user@ad.example.com"
----------------------------
  Anchor to override: ad_user@ad.example.com
  GID: 732001337
```

3. **example_for_host1** を **host1.idm.example.com** IdM クライアントに適用します。

```
$ ipa idview-apply example_for_host1 --hosts=host1.idm.example.com
----------------------------
Applied ID View "example_for_host1"
----------------------------
hosts: host1.idm.example.com
```

```
----------------------------------------------
Number of hosts the ID View was applied to: 1
----------------------------------------------
```



注記

ipa idview-apply コマンドでは、**--hostgroups** オプションも使用できます。このオプションは、ID ビューを、指定のホストグループに所属するホストに適用しますが、ホストグループ自体との関連付けは行いません。代わりに、**--hostgroups** オプションは指定されたホストグループのメンバーを拡張して、**--hosts** オプションを個別にすべて適用します。

つまり、今後ホストがホストグループに追加されても、ID ビューは新しいホストには適用されません。

4. **host1.idm.example.com** IdM クライアントの SSSD キャッシュから、**ad_user@ad.example.com** ユーザーのエントリーをクリアします。これにより、古いデータが削除され、新しいオーバーライド値が適用されるようになります。

```
[root@host1 ~]# sssctl cache-expire -u ad_user@ad.example.com
```

検証手順

1. ad_user@ad.example.com として、host1 に **SSH** で接続します。

```
[root@r8server ~]# ssh ad_user@ad.example.com@host1.idm.example.com
```

2. **ad_user@ad.example.com** ユーザーの情報を取得して、GID が更新された値を反映することを確認します。

```
[ad_user@ad.example.com@host1 ~]$ id ad_user@ad.example.com
uid=702801456(ad_user@ad.example.com) gid=732001337(admins2)
groups=732001337(admins2),702800513(domain users@ad.example.com)
```

36.4. IDM ホストグループへの ID ビューの適用

ipa idview-apply コマンドでは、**--hostgroups** オプションを使用できます。ただし、このオプションは、指定のホストに現在所属するホストに ID ビューを適用する 1 回限りの操作として機能しますが、ホストグループ自体と ID ビューを動的に関連付けることはありません。**--hostgroups** オプションは、指定したホストグループのメンバーを拡張して、**--hosts** オプションを個別にすべて適用します。

新しいホストを後でホストグループに追加する場合は、**--hosts** オプションで **ipa idview-apply** コマンドを使用して、新しいホストに ID ビューを手動で適用する必要があります。

同様に、ホストグループからホストを削除すると、ID ビューは削除後でも、ホストに割り当てられます。削除されたホストから ID ビューの適用を解除するには、**ipa idview-unapply id_view_name --hosts=name_of_the_removed_host** コマンドを実行する必要があります。

次の目標を達成するには、次の手順に従ってください。

1. ホストグループを作成し、そのグループにホストを追加する方法。
2. ID ビューをホストグループに適用する方法。

3. ホストグループに新規ホストを追加し、ID ビューを新しいホストに適用する方法。

前提条件

- ホストグループに適用する ID ビューが IdM に存在することを確認します。たとえば、ID ビューを作成して、特定の IdM クライアントで IdM ユーザーログイン名をオーバーライドするには、ID ビューを使用して、特定のホストにある IdM ユーザーのログイン名をオーバーライドする を参照してください。

手順

1. ホストグループを作成し、そのグループにホストを追加します。

   a. ホストグループを作成します。たとえば、**baltimore** という名前のホストグループを作成するには、次のコマンドを実行します。

   ```
   [root@server ~]# ipa hostgroup-add --desc="Baltimore hosts" baltimore
   ----------------------------
   Added hostgroup "baltimore"
   ----------------------------
   Host-group: baltimore
   Description: Baltimore hosts
   ```

   b. ホストグループにホストを追加します。たとえば、**host102** および **host103** を **baltimore** ホストグループに追加するには、次のコマンドを実行します。

   ```
   [root@server ~]# ipa hostgroup-add-member --hosts={host102,host103} baltimore
   Host-group: baltimore
   Description: Baltimore hosts
   Member hosts: host102.idm.example.com, host103.idm.example.com
   -------------------------
   Number of members added 2
   -------------------------
   ```

2. ホストグループのホストに ID ビューを適用します。たとえば、**example_for_host1** ID ビューを **baltimore** ホストグループに適用するには、次のコマンドを実行します。

   ```
   [root@server ~]# ipa idview-apply --hostgroups=baltimore
   ID View Name: example_for_host1
   ---------------------------------------
   Applied ID View "example_for_host1"
   ---------------------------------------
     hosts: host102.idm.example.com, host103.idm.example.com
   -----------------------------------------
   Number of hosts the ID View was applied to: 2
   -----------------------------------------
   ```

3. ホストグループに新規ホストを追加し、ID ビューを新しいホストに適用します。

   a. ホストグループに新規ホストを追加します。たとえば、**somehost.idm.example.com** ホストを **baltimore** ホストグループに追加するには、次のコマンドを実行します。

   ```
   [root@server ~]# ipa hostgroup-add-member --hosts=somehost.idm.example.com
   baltimore
     Host-group: baltimore
   ```

```
  Description: Baltimore hosts
  Member hosts:  host102.idm.example.com,
host103.idm.example.com,somehost.idm.example.com
-------------------------
Number of members added 1
-------------------------
```

b. 必要に応じて、ID ビュー情報を表示します。たとえば、**example_for_host1** ID ビューの詳細を表示するには、次のコマンドを実行します。

```
[root@server ~]# ipa idview-show example_for_host1 --all
  dn: cn=example_for_host1,cn=views,cn=accounts,dc=idm,dc=example,dc=com
  ID View Name: example_for_host1
[...]
  Hosts the view applies to: host102.idm.example.com, host103.idm.example.com
  objectclass: ipaIDView, top, nsContainer
```

この出力では、ID ビューが **baltimore** ホストグループに新規追加されたホスト **somehost.idm.example.com** に適用されていないことが分かります。

c. ID ビューを新規ホストに適用します。たとえば、ID ビュー **example_for_host1** を **somehost.idm.example.com** に適用するには、次のコマンドを実行します。

```
[root@server ~]# ipa idview-apply --host=somehost.idm.example.com
ID View Name: example_for_host1
-----------------------------------------
Applied ID View "example_for_host1"
-----------------------------------------
  hosts: somehost.idm.example.com
-------------------------------------------
Number of hosts the ID View was applied to: 1
-------------------------------------------
```

検証手順

- ID ビュー情報を再度表示します。

```
[root@server ~]# ipa idview-show example_for_host1 --all
  dn: cn=example_for_host1,cn=views,cn=accounts,dc=idm,dc=example,dc=com
  ID View Name: example_for_host1
[...]
  Hosts the view applies to: host102.idm.example.com, host103.idm.example.com,
somehost.idm.example.com
  objectclass: ipaIDView, top, nsContainer
```

この出力は、ID ビューが **baltimore** ホストグループに新規追加されたホスト **somehost.idm.example.com** に適用されていることを示しています。

第37章 ID 範囲を手動で調整

IdM サーバーは、一意のユーザー ID (UID) とグループ ID (GID) の番号を生成します。異なる ID 範囲を作成し、レプリカに割り当てることで、同じ ID 番号が生成されないようにします。デフォルトでは、このプロセスは自動的に実行されます。ただし、IdM サーバーのインストール時に IdM ID 範囲を手動で調整したり、レプリカの DNA ID 範囲を手動で定義したりできます。

37.1. ID 範囲

ID 番号は **ID 範囲** に分類されます。個別のサーバーとレプリカに別々の数値範囲を指定することで、エントリーに対して発行された ID 番号が別のサーバーまたはレプリカの別のエントリーですでに使用されている可能性がなくなります。

ID 範囲には、以下の 2 つのタイプがあります。

- 最初のサーバーのインストール時に割り当てられる IdM **ID 範囲**。この範囲は作成後に変更できません。ただし、元の IdM ID 範囲に加えて、新しい IdM ID 範囲を作成できます。詳細は、自動 ID 範囲の割り当て および 新しい IdM ID 範囲の追加 を参照してください。

- ユーザーが変更できる**DNA (Distributed Numeric Assignment)** の ID 範囲。既存の IdM ID 範囲内に収まるようにする必要があります。詳細は、DNA ID 範囲の手動割り当て を参照してください。
  レプリカには、**次の** DNA ID 範囲を割り当てることもできます。レプリカは、現在の範囲で ID が不足すると、次の範囲を使用します。レプリカが削除された場合、次の範囲は自動的に割り当てられないため、 手動で割り当てる 必要があります。

範囲は、ドメインのバックエンド 389 Directory Server インスタンスの一部として、DNA プラグインによってサーバーとレプリカとの間で更新され、共有されます。

DNA 範囲の定義は、次の 2 つの属性によって設定されます。

- サーバーの次に使用可能な番号: DNA 範囲の下限値

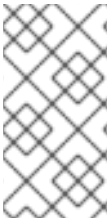
- 範囲サイズ: DNA 範囲内の ID の数

初期の下限範囲は、プラグインインスタンスの設定時に設定されます。その後、プラグインは下限値を更新します。利用可能な数を範囲に分割すると、サーバーは、互いに重複せずに、継続的に数字を割り当てることができます。

37.2. 自動 ID 範囲の割り当て

IdM ID 範囲

デフォルトでは、IdM サーバーのインストール時に IdM ID 範囲が自動的に割り当てられます。**Ipa-server-install** コマンドは、使用可能な合計 1 万の範囲から、20 万個の ID を無作為に選択して割り当てます。このようにランダムな範囲を選択すると、今後別の 2 つの IdM ドメインを統合する場合に、ID の競合が発生する可能性を大幅に削減できます。

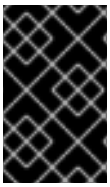


注記

この IdM の ID 範囲は、作成後は修正できません。DNA (Distributed Numeric Assignment) の ID 範囲を手動で調整できるのは、DNA ID 範囲の手動割り当て で説明されているコマンドを使用する場合だけです。インストール時に、IdM ID 範囲に一致する DNA 範囲が自動的に作成されます。

DNA ID 範囲

IdM サーバー 1 台をインストールしている場合は、このサーバーが DNA ID 範囲全体を制御します。新規レプリカをインストールし、レプリカが独自の DNA ID 範囲を要求すると、サーバーの初期 ID 範囲が分割され、サーバーとレプリカの間で分散されます。レプリカは、初期サーバーで使用可能な DNA ID 範囲の残りの半分を受け取ります。次に、サーバーとレプリカは、新規ユーザーまたはグループのエントリーに元の ID 範囲に対応する部分を使用します。また、レプリカが割り当てられた ID 範囲を使い果たしそうになり、ID が 100 未満しか残っていない場合には、レプリカは他の使用可能なサーバーに接続して、新しい DNA ID 範囲を要求します。



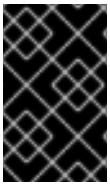
重要

レプリカをインストールしても、即座に **ID** 範囲を受け取ることはありません。レプリカは、DNA プラグインの初回使用時 (ユーザーの初回追加時など) に ID 範囲を受け取ります。

レプリカが DNA ID 範囲を要求する前に最初のサーバーが機能しなくなると、レプリカはサーバーに問い合わせて ID 範囲を要求することができません。レプリカに新しいユーザーを追加しようとすると失敗します。このような場合は、無効になったサーバーに割り当てられている ID 範囲を確認 し、ID 範囲を手動でレプリカに割り当てる ことができます。

37.3. サーバーインストール時の IDM ID 範囲の手動割り当て

デフォルトの動作をオーバーライドし、無作為に割り当てる代わりに、IdM ID 範囲を手動で設定できます。



重要

UID の値が 1000 以下の ID 範囲は設定しないでください。1000 以下の値はシステム使用向けに予約されています。また、SSSD サービスは ID の値 0 を処理しないので、0 値が含まれる ID 範囲は設定しないでください。

手順

- **ipa-server-install** では以下の 2 つのオプションを使用することで、サーバーのインストール時に IdM ID 範囲を手動で定義できます。
  - **--idstart**: UID および GID 番号の開始値を指定します。
  - **--idmax**: UID および GID 番号の最大値を指定します。デフォルトでは、この値は **--idstart** の開始値に 199,999 を加えたものになります。

検証手順

- ID 範囲が正しく割り当てられているかを確認するには、**ipa idrange-find** コマンドを使用して、割り当てられた IdM ID 範囲を表示します。

```
# ipa idrange-find
----------------
1 range matched
----------------
  Range name: IDM.EXAMPLE.COM_id_range
  First Posix ID of the range: 882200000
  Number of IDs in the range: 200000
```

```
  Range type: local domain range
----------------------------
Number of entries returned 1
----------------------------
```

37.4. 新しい IDM ID 範囲の追加

レプリカの ID が不足し、元の IdM ID 範囲を使い果たした場合など、場合によっては、元の IdM 範囲に加え、新しい IdM ID 範囲の作成が必要になる場合があります。

重要

新しい IdM ID 範囲を追加しても、新しい DNA ID 範囲は自動的に作成されません。必要に応じて、レプリカに新しい DNA ID 範囲を手動で割り当てる必要があります。割り当て方法の詳細は、DNA ID 範囲の手動割り当て を参照してください。

手順

1. 新しい IdM ID 範囲を作成するには、**ipa idrange-add** コマンドを使用します。新しい範囲名、範囲の最初の ID 番号、および範囲サイズを指定する必要があります。

```
# ipa idrange-add IDM.EXAMPLE.COM_new_range --base-id=1000000 --range-size=200000
----------------------------------------
Added ID range "IDM.EXAMPLE.COM_new_range"
----------------------------------------
  Range name: IDM.EXAMPLE.COM_new_range
  First Posix ID of the range: 1000000
  Number of IDs in the range: 200000
  Range type: local domain range
```

2. Directory Server を再起動します。

```
# systemctl restart dirsrv@IDM.EXAMPLE.COM.service
```

これにより、新しい範囲の UID を使用してユーザーを作成するときに、セキュリティー識別子 (SID) が割り当てられるようになります。

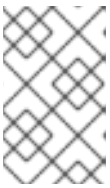
3. オプション: ID 範囲をすぐに更新します。

   a. System Security Services Daemon (SSSD) キャッシュをクリアします。

```
# sss_cache -E
```

   b. SSSD デーモンを再起動します。

```
# systemctl restart sssd
```



注記

SSSD キャッシュをクリアせずにサービスを再起動すると、SSSD は、IdM サーバーに保存されているドメインリストとその他の設定データを更新するときに、新しい ID 範囲のみを検出します。

検証手順

- **ipa idrange-find** コマンドを使用すると、新しい範囲が正しく設定されているかどうかを確認できます。

```
# ipa idrange-find
----------------
2 ranges matched
----------------
  Range name: IDM.EXAMPLE.COM_id_range
  First Posix ID of the range: 882200000
  Number of IDs in the range: 200000
  Range type: local domain range

  Range name: IDM.EXAMPLE.COM_new_range
  First Posix ID of the range: 1000000
  Number of IDs in the range: 200000
  Range type: local domain range
----------------------------
Number of entries returned 2
----------------------------
```

37.5. IDM ID 範囲におけるセキュリティーおよび相対識別子のロール

Identity Management (IdM) ID 範囲は、いくつかのパラメーターによって定義されます。

- 範囲名
- 範囲の最初の POSIX ID
- 範囲サイズ: 範囲内の ID の数
- 対応する **RID 範囲** の最初の **相対 ID** (RID)
- **セカンダリー RID 範囲** の最初の RID

これらの値は、**ipa idrange-show** コマンドを使用して表示できます。

```
$ ipa idrange-show IDM.EXAMPLE.COM_id_range
  Range name: IDM.EXAMPLE.COM_id_range
  First Posix ID of the range: 196600000
  Number of IDs in the range: 200000
  First RID of the corresponding RID range: 1000
  First RID of the secondary RID range: 1000000
  Range type: local domain range
```

セキュリティー識別子

ローカルドメインの ID 範囲からのデータは、IdM サーバーによって内部的に使用され、一意の **セキュリティー識別子** (SID) が IdM ユーザーおよびグループに割り当てられます。SID は、ユーザーオブジェクトとグループオブジェクトに格納されます。ユーザーの SID は、以下で構成されます。

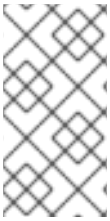
- ドメイン SID
- ドメイン SID に追加された 4 桁の 32 ビット値である、ユーザーの **相対識別子** (RID)

たとえば、ドメイン SID が S-1-5-21-123-456-789 で、このドメインのユーザーの RID が 1008 の場合、ユーザーの SID は SID of S-1-5-21-123-456-789-1008 になります。

相対識別子

RID 自体は、次の方法で計算されます。

ユーザーの POSIX UID から範囲の最初の POSIX ID を引き、対応する RID 範囲の最初の RID を結果に追加します。たとえば、**idmuser** の UID が 196600008、最初の POSIX ID が 196600000、そして最初の RID が 1000 の場合、**idmuser** の RID は 1008 になります。



注記

ユーザーの RID を計算するアルゴリズムは、対応する RID を計算する前に、特定の POSIX ID が割り当てられた ID 範囲内にあるかどうかをチェックします。たとえば、最初の ID が 196600000 で範囲サイズが 200000 の場合、1600000 の POSIX ID は ID 範囲外となるり、アルゴリズムはその RID を計算しません。

セカンダリー相対識別子

IdM では、POSIX UID は POSIX GID と同一にすることができます。これは、196600008 の UID を持つ **idmuser** がすでに存在する場合でも、196600008 の GID を持つ新しい **idmgroup** グループを作成できることを意味します。

ただし、SID で定義できるオブジェクトは、ユーザー **または** グループの 1 つだけです。**idmuser** 用にすでに作成されている S-1-5-21-123-456-789-1008 の SID は、**idmgroup** と共有することはできません。**idmgroup** の代替 SID を生成する必要があります。

IdM は、SID との競合を避けるために、**セカンダリー相対識別子** (セカンダリー RID) を使用します。このセカンダリー RID は、以下で構成されます。

- セカンダリー RID ベース
- 範囲サイズ (デフォルトではベース範囲サイズと同じ)。

上記の例では、セカンダリー RID ベースは 1000000 に設定されています。新しく作成された **idmgroup** の RID を計算するには、ユーザーの POSIX UID から範囲の最初の POSIX ID を引き、結果にセカンダリー RID 範囲の最初の RID を追加します。したがって、**idmgroup** には 1000008 の RID が割り当てられます。その結果、**idmgroup** の SID は S-1-5-21-123-456-789-1000008 になります。

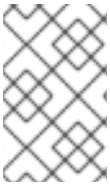
ユーザーまたはグループオブジェクトが以前に手動で設定された POSIX ID で作成されている場合にのみ、IdM はセカンダリー RID を使用して SID を計算します。そうでない場合、自動割り当てにより、同じ ID が 2 回割り当てられることを防ぎます。

関連情報

- Ansible を使用して新規ローカル IdM ID 範囲を追加する方法

37.6. ANSIBLE を使用して新規ローカル IDM ID 範囲を追加する方法

レプリカの ID が不足し、元の IdM ID 範囲を使い果たした場合など、場合によっては、元の IdM 範囲に加え、新しい Identity Management (IdM) ID 範囲の作成が必要になる場合があります。以下の例は、Ansible Playbook を使用して新しい IdM ID 範囲を作成する方法を説明しています。



注記

新しい IdM ID 範囲を追加しても、新しい DNA ID 範囲は自動的に作成されません。必要に応じて、新しい DNA ID 範囲を手動で割り当てる必要があります。割り当て方法の詳細は、DNA ID 範囲の手動割り当て を参照してください。

前提条件

- IdM **admin** のパスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **~/MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. 次の内容で **idrange-present.yml** playbook を作成します。

```
---
- name: Playbook to manage idrange
  hosts: ipaserver
  become: no

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure local idrange is present
    ipaidrange:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: new_id_range
      base_id: 12000000
      range_size: 200000
      rid_base: 1000000
      secondary_rid_base: 200000000
```

3. ファイルを保存します。
4. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory idrange-present.yml
```

5. **ipaserver** に **SSH** 接続し、Directory Server を再起動します。

```
# systemctl restart dirsrv@IDM.EXAMPLE.COM.service
```

これにより、新しい範囲の UID を使用してユーザーを作成するときに、セキュリティー識別子 (SID) が割り当てられるようになります。

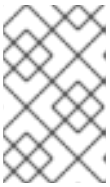
6. オプション: ID 範囲をすぐに更新します。

a. **ipaserver** で、System Security Services Daemon (SSSD) キャッシュをクリアします。

```
# sss_cache -E
```

b. **ipaserver** で SSSD デーモンを再起動します。

```
# systemctl restart sssd
```



注記

SSSD キャッシュをクリアせずにサービスを再起動すると、SSSD は、IdM サーバーに保存されているドメインリストとその他の設定データを更新するときに、新しい ID 範囲のみを検出します。

検証手順

- **ipa idrange-find** コマンドを使用すると、新しい範囲が正しく設定されているかどうかを確認できます。

```
# ipa idrange-find
----------------
2 ranges matched
----------------
  Range name: IDM.EXAMPLE.COM_id_range
  First Posix ID of the range: 882200000
  Number of IDs in the range: 200000
  Range type: local domain range

  Range name: IDM.EXAMPLE.COM_new_id_range
  First Posix ID of the range: 12000000
  Number of IDs in the range: 200000
  Range type: local domain range
----------------------------
Number of entries returned 2
----------------------------
```

関連情報

- IdM ID 範囲におけるセキュリティーおよび相対識別子のロール

37.7. AD への信頼を削除した後の ID 範囲の削除

IdM 環境と Active Directory (AD) 環境間の信頼を削除している場合は、それに関連付けられている ID 範囲を削除することを推奨します。

警告

信頼できるドメインに関連付けられた ID 範囲に割り当てられた ID は、IdM に登録されているシステムのファイルおよびディレクトリーの所有権に引き続き使用される可能性があります。

削除した AD 信頼に対応する ID 範囲を削除すると、AD ユーザーが所有するファイルおよびディレクトリーの所有権を解決できなくなります。

前提条件

- AD 環境への信頼を削除している。

手順

1. 現在使用されている ID 範囲をすべて表示します。

```
[root@server ~]# ipa idrange-find
```

2. 削除した信頼に関連付けられた ID 範囲の名前を識別します。ID 範囲の名前の最初の部分は、信頼の名前 (**AD.EXAMPLE.COM_id_range** など) になります。

3. 範囲を削除します。

```
[root@server ~]# ipa idrange-del AD.EXAMPLE.COM_id_range
```

4. SSSD サービスを再起動して、削除した ID 範囲への参照を削除します。

```
[root@server ~]# systemctl restart sssd
```

関連情報

- コマンドラインを使用した信頼の削除 を参照してください。
- Removing the trust using the IdM Web UI を参照してください。

37.8. 現在割り当てられている DNA ID 範囲の表示

サーバーで現在アクティブな DNA (Distributed Numeric Assignment) の ID 範囲と、次の DNA 範囲が割り当てられている場合にはその範囲の両方を表示できます。

手順

- トポロジー内のサーバーに設定されている DNA ID 範囲を表示するには、以下のコマンドを使用します。
  - **ipa-replica-manage dnarange-show** は、全サーバー (サーバーを指定した場合は指定されたサーバーでのみ) に設定されている現在の DNA ID 範囲を表示します。以下に例を示します。

```
# ipa-replica-manage dnarange-show
serverA.example.com: 1001-1500
serverB.example.com: 1501-2000
serverC.example.com: No range set

# ipa-replica-manage dnarange-show serverA.example.com
serverA.example.com: 1001-1500
```

  - **ipa-replica-manage dnanextrange-show** は、全サーバーに現在設定されている次の DNA ID 範囲を表示します。サーバーを指定した場合は、指定したサーバー上でのみ表示されます。以下に例を示します。

```
# ipa-replica-manage dnanextrange-show
serverA.example.com: 2001-2500
serverB.example.com: No on-deck range set
serverC.example.com: No on-deck range set

# ipa-replica-manage dnanextrange-show serverA.example.com
serverA.example.com: 2001-2500
```

37.9. 手動による ID 範囲の割り当て

特定の状況では、DNA (Distributed Numeric Assignment) の ID 範囲を手動で割り当てする必要があります。たとえば、以下のような場合です。

- レプリカの ID がなくなり、IdM ID 範囲がすべて使われている。
  レプリカに割り当てられた DNA ID 範囲を使い果たし、IdM 範囲で使用可能な空き ID がなくなったため、ID の追加要求に失敗した場合。

  この状況を解決するには、レプリカに割り当てられた DNA ID 範囲を拡張します。これは、以下の 2 つの方法で実行できます。
  - 別のレプリカに割り当てられる DNA ID 範囲を短くし、新たに利用可能な値を、ID 範囲を使い果たしたレプリカに割り当てます。
  - 新しい IdM ID 範囲を作成し、この作成した IdM 範囲内でレプリカに新しい DNA ID 範囲を設定します。
    新しい IdM ID 範囲を作成する方法は 新しい IdM ID 範囲の追加 を参照してください。
- レプリカが機能しなくなる
  レプリカが停止して削除する必要がある場合、レプリカの DNA ID 範囲は自動的に取得されません。つまり、以前にレプリカに割り当てられていた DNA ID 範囲は使用できなくなります。DNA ID 範囲を復元し、他のレプリカで使用できるようにします。

  これを行うには、その範囲を別のサーバーに手動で割り当てる前に、ID 範囲の値を調べます。また、UID や GID が重複しないように、回復した範囲からの ID の値がユーザーまたはグループに割り当てられていないことを確認します。これは、既存のユーザーおよびグループの UID と GID を調べて実行できます。

DNA ID 範囲の手動割り当て のコマンドを使用して、レプリカに DNA ID 範囲を手動で割り当てることができます。

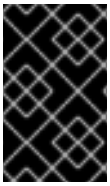
注記

新しい DNA ID 範囲を割り当てると、サーバーまたはレプリカ上の既存のエントリーの UID は同じになります。現在の DNA ID 範囲を変更しても、IdM は過去に割り当てられた範囲の記録を保持するため、これにより問題が発生することはありません。

37.10. DNA ID 範囲の手動割り当て

場合によっては、機能していないレプリカに割り当てられた DNA ID 範囲を再割り当てするなど、既存レプリカに DNA (Distributed Numeric Assignment) の ID 範囲を手動で割り当てないといけない場合があります。詳細は、手動による ID 範囲の割り当て を参照してください。

DNA ID 範囲を手動で調整する場合は、新たに調整した範囲が IdM ID 範囲に含まれていることを確認してください。これは、**ipa idrange-find** コマンドを使用して確認できます。そうでない場合、コマンドは失敗します。



重要

ID 範囲を重複しないように注意してください。サーバーまたはレプリカに割り当てた ID 範囲のいずれかが重複すると、この 2 つのサーバーにより、異なるエントリーに同じ ID 値を割り当てる可能性があります。

前提条件

- **オプション。**機能していないレプリカから DNA ID 範囲を復元する場合は、最初に 現在割り当てられている DNA ID 範囲の表示 に記載のコマンドを使用して ID 範囲を見つけます。

手順

- 指定のサーバーの現在の DNA ID 範囲を定義するには、**ipa-replica-manage dnarange-set** を使用します。

```
# ipa-replica-manage dnarange-set serverA.example.com 1250-1499
```

- 指定のサーバーの次の DNA ID 範囲を定義するには、**ipa-replica-manage dnanextrange-set** を使用します。

```
# ipa-replica-manage dnanextrange-set serverB.example.com 1500-5000
```

検証手順

- 現在割り当てられている DNA ID 範囲の表示 で説明されているコマンドを使用して、新しい DNA 範囲が正しく設定されているかどうかを確認できます。

第38章 SUBID 範囲の手動管理

コンテナー化環境では、IdM ユーザーが subID 範囲を手動で割り当てる必要がある場合があります。次の手順では、サブ ID 範囲を管理する方法について説明します。

38.1. IDM CLI を使用した SUBID 範囲の生成

Identity Management (IdM) の管理者は、subID 範囲を生成し、それを IdM ユーザーに割り当てることができます。

前提条件

- IdM ユーザーが存在する。
- IdM **admin** Ticket-Granting Ticket (TGT) を取得済みである。詳細は、kinit による IdM への手動ログイン を参照してください。
- この手順を実行する IdM ホストへの **root** アクセス権がある。

手順

1. [オプション] 既存のサブ ID 範囲を確認します。

```
# ipa subid-find
```

2. subID 範囲が存在しない場合は、次のいずれかの方法を選択します。
   - 1 つの subID 範囲を生成し、1 つの IdM ユーザーに割り当てます。

```
# ipa subid-generate --owner=idmuser

Added subordinate id "359dfcef-6b76-4911-bd37-bb5b66b8c418"

  Unique ID: 359dfcef-6b76-4911-bd37-bb5b66b8c418
  Description: auto-assigned subid
  Owner: idmuser
  SubUID range start: 2147483648
  SubUID range size: 65536
  SubGID range start: 2147483648
  SubGID range size: 65536
```

   - 複数の subID 範囲を生成し、すべての IdM ユーザーに割り当てます。

```
# /usr/libexec/ipa/ipa-subids --all-users

Found 2 user(s) without subordinate ids
  Processing user 'user4' (1/2)
  Processing user 'user5' (2/2)
Updated 2 user(s)
The ipa-subids command was successful
```

3. [オプション] デフォルトで新しい IdM ユーザーに subID 範囲を割り当てます。

```
# ipa config-mod --user-default-subid=True
```

-

検証

- ユーザーに subID 範囲が割り当てられていることを確認します。

```
# ipa subid-find --owner=idmuser

1 subordinate id matched

  Unique ID: 359dfcef-6b76-4911-bd37-bb5b66b8c418
  Owner: idmuser
  SubUID range start: 2147483648
  SubUID range size: 65536
  SubGID range start: 2147483648
  SubGID range size: 65536

Number of entries returned 1
```

38.2. IDM WEBUI インターフェイスを使用した SUBID 範囲の生成

アイデンティティー管理 (IdM) 管理者は、サブ ID 範囲を生成し、IdM WebUI インターフェイスでユーザーに割り当てることができます。

前提条件

- IdM ユーザーが存在します。
- IdM **管理者** Kerberos チケット (TGT) を取得しました。詳細は、Web UI で IdM にログイン: Kerberos チケットの使用 を参照してください。
- この手順を実行する IdM ホストへの **root** アクセス権がある。

手順

1. IdM WebUI インターフェイスで、**Subordinate IDs** タブをデプロイメントし、**Subordinate IDs** を選択します。
2. **Subordinate IDs** インターフェイスが表示されたら、インターフェイスの右上にある **Add** をクリックします。**サブ ID の追加** ウィンドウが表示されます。
3. **サブ ID の追加** ウィンドウで、サブ ID 範囲を割り当てるユーザーである所有者を選択します。
4. **Add** ボタンをクリックします。

検証

- **Subordinate IDs** タブの下にあるテーブルを確認します。新しいレコードがテーブルに表示されます。所有者は、サブ ID 範囲を割り当てたユーザーです。

38.3. IDM CLI を使用して IDM ユーザーのサブ ID 情報を表示する

アイデンティティー Management (IdM) ユーザーは、IdM ユーザーのサブ ID 範囲を検索し、関連情報を表示できます。

前提条件

- IdM クライアントでサブ ID 範囲を設定しました。
- IdM ユーザーのチケット許可チケット (TGT) を取得しました。詳細は、kinit による IdM への手動ログイン を参照してください。

手順

- サブ ID 範囲の詳細を表示するには:
  - 範囲の所有者であるアイデンティティー Management (IdM) ユーザーの一意の ID ハッシュがわかっている場合:

```
$ ipa subid-show 359dfcef-6b76-4911-bd37-bb5b66b8c418

  Unique ID: 359dfcef-6b76-4911-bd37-bb5b66b8c418
  Owner: idmuser
  SubUID range start: 2147483648
  SubUID range size: 65536
  SubGID range start: 2147483648
  SubGID range size: 65536
```

  - その範囲内の特定のサブ ID がわかっている場合:

```
$ ipa subid-match --subuid=2147483670

1 subordinate id matched

  Unique ID: 359dfcef-6b76-4911-bd37-bb5b66b8c418
  Owner: uid=idmuser
  SubUID range start: 2147483648
  SubUID range size: 65536
  SubGID range start: 2147483648
  SubGID range size: 65536

Number of entries returned 1
```

38.4. GETSUBID コマンドを使用して SUBID 範囲をリスト表示する

システム管理者は、コマンドラインインターフェイスを使用して、アイデンティティー Management (IdM) またはローカルユーザーのサブ ID 範囲をリスト表示できます。

前提条件

- **idmuser** ユーザーは IdM に存在します。
- **shadow-utils-subid** パッケージがインストールされている。
- **/etc/nsswitch.conf** ファイルを編集できます。

手順

1. **/etc/nsswitch.conf** ファイルを開き、**subid** 変数を **sss** 値に設定して、**shadow-utils** ユーティリティーが IdM サブ ID 範囲を使用するように設定します。

```
[...]
subid: sss
```

注記

subid フィールドには 1 つの値のみを指定できます。**subid** フィールドを **sss** ではなく **ファイル** 値に設定するか、値なしに設定すると、**shadow-utils** ユーティリティーは **/etc/subuid** および **/etc/subgid** ファイルの subID 範囲を使用するように設定されます。

2. IdM ユーザーのサブ ID 範囲をリスト表示します。

```
$ getsubids idmuser
0: idmuser 2147483648 65536
```

最初の値 2147483648 は、サブ ID 範囲の開始を示します。2 番目の値 65536 は、範囲のサイズを示します。

第39章 ANSIBLE を使用した IDM でのレプリケーショントポロジーの管理

複数の Identity Management (IdM) サーバーを維持し、冗長性の目的で相互に複製して、サーバーの損失を軽減または防止することができます。たとえば、1 台のサーバーに障害が発生しても、その他のサーバーがドメインにサービスを提供し続けます。障害が発生していないサーバーの 1 台から新しいレプリカを作成し、失われたサーバーを回復することもできます。

IdM サーバーに保存されているデータは、レプリカ合意に基づいて複製されます。2 台のサーバーでレプリカ合意が設定されている場合は、データを共有します。レプリケートされるデータはトポロジーの **suffix** に保存されます。2 つのレプリカに接尾辞間でレプリカ合意があると、接尾辞はトポロジー **segment** を形成します。

本章では、**Red Hat Ansible Engine** を使用して IdM レプリカ合意、トポロジーセグメント、およびトポロジー接尾辞を管理する方法を説明します。本章は以下のセクションで設定されます。

- Ansible を使用して、レプリカ合意が IdM に存在することを確認
- Ansible を使用して複数の IdM レプリカ間でレプリカ合意を存在させる手順
- Ansible を使用して 2 つのレプリカ間でレプリカ合意が存在するかどうかの確認
- Ansible を使用してトポロジーの接尾辞が IdM に存在することを確認
- Ansible を使用した IdM レプリカの再初期化
- Ansible を使用して IdM にレプリカ合意がないことを確認する手順

39.1. ANSIBLE を使用して、レプリカ合意が IDM に存在することを確認

Identity Management (IdM) サーバーに保存されているデータは、レプリカ合意に基づいて複製されます。2 台のサーバーでレプリカ合意が設定されている場合は、データを共有します。レプリカ合意は常に双方向のものです。最初のレプリカからサーバーから別のレプリカにデータが複製されるだけでなく、別ののレプリカから最初のレプリカにもデータが複製されます。

この手順に従い、Ansible Playbook を使用して、**server.idm.example.com** と **replica.idm.example.com** との間で **domain** タイプのレプリカ合意が存在することを確認説明します。

前提条件

- トポロジーで IdM レプリカを接続するためのガイドライン に記載されている IdM トポロジーを設計するための推奨事項を確実に理解している。
- IdM **admin** のパスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。

- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **~/MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible -freeipa/playbooks/topology/** ディレクトリーにある **add-topologysegment.yml** Ansible Playbook ファイルをコピーします。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/topology/add-topologysegment.yml
add-topologysegment-copy.yml
```

3. **add-topologysegment-copy.yml** ファイルを開いて編集します。

4. **ipatopologysegment** タスクセクションに以下の変数を設定して、ファイルを調整します。

   - **ipaadmin_password** 変数は IdM **admin** のパスワードに設定します。
   - 追加するセグメントのタイプに応じて、**suffix** 変数を **domain** または **ca** のいずれかに設定します。
   - **left** の変数をレプリカ合意の左ノードに設定する IdM サーバーの名前に設定します。
   - レプリカ合意の適切なノードとなる IdM サーバーの名前に **right** 変数を設定します。
   - **state** 変数は **present** に設定されていることを確認します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Playbook to handle topologysegment
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
- name: Add topology segment
    ipatopologysegment:
      ipaadmin_password: "{{ ipaadmin_password }}"
      suffix: domain
      left: server.idm.example.com
      right: replica.idm.example.com
      state: present
```

5. ファイルを保存します。

6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory add-
topologysegment-copy.yml
```

関連情報

- レプリカ合意、トポロジー接尾辞、およびトポロジーセグメントの説明 を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-topology.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/topology** ディレクトリーのサンプルの Playbook を参照してください。

39.2. ANSIBLE を使用して複数の IDM レプリカ間でレプリカ合意を存在させる手順

Identity Management (IdM) サーバーに保存されているデータは、レプリカ合意に基づいて複製されます。2 台のサーバーでレプリカ合意が設定されている場合は、データを共有します。レプリカ合意は常に双方向のものです。最初のレプリカからサーバーから別のレプリカにデータが複製されるだけでなく、別ののレプリカから最初のレプリカにもデータが複製されます。

以下の手順に従って、IdM の複数のレプリカのペア間でレプリカ合意が存在することを確認します。

前提条件

- トポロジー内のレプリカの接続 に記載されている IdM トポロジーを設計するための推奨事項を理解しておく。
- IdM **admin** のパスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **~/MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible -freeipa/playbooks/topology/** ディレクトリーにある **add-topologysegments.yml** Ansible Playbook ファイルをコピーします。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/topology/add-topologysegments.yml
add-topologysegments-copy.yml
```

3. **add-topologysegments-copy.yml** ファイルを開いて編集します。

4. **vars** セクションに以下の変数を設定して、ファイルを調整します。

   - **ipaadmin_password** 変数は IdM **admin** のパスワードに設定します。
   - すべてのトポロジーセグメントについて、**ipatopology_segments** セクションに行を追加し、以下の変数を設定します。
     - 追加するセグメントのタイプに応じて、**suffix** 変数を **domain** または **ca** のいずれかに設定します。
     - **left** の変数をレプリカ合意の左ノードに設定する IdM サーバーの名前に設定します。
     - レプリカ合意の適切なノードとなる IdM サーバーの名前に **right** 変数を設定します。

5. **add-topologysegments-copy.yml** ファイルの **tasks** セクションで、**state** 変数が **present** に設定されていることを確認します。
   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Add topology segments
  hosts: ipaserver
  gather_facts: false

  vars:
    ipaadmin_password: "{{ ipaadmin_password }}"
    ipatopology_segments:
    - {suffix: domain, left: replica1.idm.example.com , right: replica2.idm.example.com }
    - {suffix: domain, left: replica2.idm.example.com , right: replica3.idm.example.com }
    - {suffix: domain, left: replica3.idm.example.com , right: replica4.idm.example.com }
    - {suffix: domain+ca, left: replica4.idm.example.com , right: replica1.idm.example.com }

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Add topology segment
    ipatopologysegment:
      ipaadmin_password: "{{ ipaadmin_password }}"
      suffix: "{{ item.suffix }}"
      name: "{{ item.name | default(omit) }}"
      left: "{{ item.left }}"
      right: "{{ item.right }}"
      state: present
      #state: absent
      #state: checked
      #state: reinitialized
    loop: "{{ ipatopology_segments | default([]) }}"
```

6. ファイルを保存します。

7. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory add-topologysegments-copy.yml
```

関連情報

- レプリカ合意、トポロジー接尾辞、およびトポロジーセグメントの説明 を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-topology.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/topology** ディレクトリーのサンプルの Playbook を参照してください。

39.3. ANSIBLE を使用して 2 つのレプリカ間でレプリカ合意が存在するかどうかの確認

Identity Management (IdM) サーバーに保存されているデータは、レプリカ合意に基づいて複製されます。2 台のサーバーでレプリカ合意が設定されている場合は、データを共有します。レプリカ合意は常に双方向のものです。最初のレプリカからサーバーから別のレプリカにデータが複製されるだけでなく、別ののレプリカから最初のレプリカにもデータが複製されます。

以下の手順に従って、IdM のレプリカのペア間でレプリカ合意が存在することを確認します。

前提条件

- トポロジー内のレプリカの接続 に記載されている Identity Management (IdM) トポロジーを設計するための推奨事項を理解しておく。
- IdM **admin** のパスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **~/MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible -freeipa/playbooks/topology/** ディレクトリーにある **check-topologysegments.yml** Ansible Playbook ファイルをコピーします。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/topology/check-topologysegments.yml
check-topologysegments-copy.yml
```

3. **check-topologysegments-copy.yml** ファイルを開いて編集します。

4. **vars** セクションに以下の変数を設定して、ファイルを調整します。

   - **ipaadmin_password** 変数は IdM **admin** のパスワードに設定します。
   - すべてのトポロジーセグメントについて、**ipatopology_segments** セクションに行を追加し、以下の変数を設定します。
     - 追加するセグメントのタイプに応じて、**suffix** 変数を **domain** または **ca** のいずれかに設定します。
     - **left** の変数をレプリカ合意の左ノードに設定する IdM サーバーの名前に設定します。
     - レプリカ合意の適切なノードとなる IdM サーバーの名前に **right** 変数を設定します。

5. **check-topologysegments-copy.yml** ファイルの **tasks** セクションで、**state** 変数が **present** に設定されていることを確認します。
   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Add topology segments
  hosts: ipaserver
  gather_facts: false

  vars:
    ipaadmin_password: "{{ ipaadmin_password }}"
    ipatopology_segments:
    - {suffix: domain, left: replica1.idm.example.com, right: replica2.idm.example.com }
    - {suffix: domain, left: replica2.idm.example.com , right: replica3.idm.example.com }
    - {suffix: domain, left: replica3.idm.example.com , right: replica4.idm.example.com }
    - {suffix: domain+ca, left: replica4.idm.example.com , right:
replica1.idm.example.com }

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Check topology segment
    ipatopologysegment:
      ipaadmin_password: "{{ ipaadmin_password }}"
      suffix: "{{ item.suffix }}"
      name: "{{ item.name | default(omit) }}"
      left: "{{ item.left }}"
      right: "{{ item.right }}"
      state: checked
    loop: "{{ ipatopology_segments | default([]) }}"
```

6. ファイルを保存します。

7. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory check-
topologysegments-copy.yml
```

関連情報

- トポロジー合意、接尾辞、およびセグメントの概念の詳細は、レプリカ合意、トポロジー接尾辞、およびトポロジーセグメント を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-topology.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/topology** ディレクトリーのサンプルの Playbook を参照してください。

39.4. ANSIBLE を使用してトポロジーの接尾辞が IDM に存在することを確認

Identity Management (IdM) のレプリカ合意のコンテキストでは、トポロジー接尾辞はレプリケートされるデータを保存します。IdM は、**domain** と **ca** の 2 種類のトポロジー接尾辞に対応します。それぞれの接尾辞は、個別のバックエンドである個別のレプリケーショントポロジーを表します。レプリカ合意が設定されると、同じタイプのトポロジー接尾辞を 2 つの異なるサーバーに結合します。

domain 接尾辞には、ユーザー、グループ、ポリシーなどのドメイン関連のデータがすべて含まれます。**ca** 接尾辞には、Certificate System コンポーネントのデータが含まれます。これは認証局 (CA) がインストールされているサーバーにのみ存在します。

以下の手順に従って、Ansible Playbook を使用して、トポロジー接尾辞が IdM に存在することを確認します。この例では、**domain** 接尾辞が IdM に存在することを確認する方法を説明します。

前提条件

- IdM **admin** のパスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **~/MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/topology/** ディレクトリーにある **verify-topologysuffix.yml** Ansible Playbook ファイルをコピーします。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/topology/ verify-topologysuffix.yml
verify-topologysuffix-copy.yml
```

3. Ansible Playbook ファイル **verify-topologysuffix-copy.yml** を開きます。
4. **ipatopologysuffix** セクションに以下の変数を設定して、ファイルを調整します。
   - **ipaadmin_password** 変数は IdM **admin** のパスワードに設定します。
   - **suffix** 変数は **domain** に設定し ます。**ca** 接尾辞が存在することを確認する場合は、変数を **ca** に設定します。
   - **state** 変数が **verified** に設定されていることを確認します。他のオプションは使用できません。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Playbook to handle topologysuffix
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Verify topology suffix
    ipatopologysuffix:
      ipaadmin_password: "{{ ipaadmin_password }}"
      suffix: domain
      state: verified
```

5. ファイルを保存します。
6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

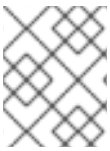
```
$ ansible-playbook --vault-password-file=password_file -v -i inventory verify-topologysuffix-copy.yml
```

関連情報

- レプリカ合意、トポロジー接尾辞、およびトポロジーセグメントの説明 を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-topology.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/topology** ディレクトリーのサンプルの Playbook を参照してください。

39.5. ANSIBLE を使用した IDM レプリカの再初期化

レプリカが長期間オフラインである場合や、そのデータベースが破損している場合は、初期化できます。初期化により、更新リストのデータでレプリカが更新されます。たとえば、バックアップからの権威復元が必要な場合に使用できます。



注記

レプリケーションの更新とは対照的に、レプリカが変更エントリーのみを送信する間、データベース全体を再初期化します。

コマンドを実行するローカルホストは、再初期化されたレプリカです。データの取得元となるレプリカを指定するには、**direction** オプションを使用します。

以下の手順に従って、Ansible Playbook を使用して **server.idm.example.com** から **replica.idm.example.com** の **domain** データを再初期化します。

前提条件

- IdM **admin** のパスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **~/MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible -freeipa/playbooks/topology/** ディレクトリーにある **reinitialize-topologysegment.yml** Ansible Playbook ファイルをコピーします。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/topology/reinitialize-topologysegment.yml reinitialize-topologysegment-copy.yml
```

3. **reinitialize-topologysegment-copy.yml** ファイルを開いて編集します。
4. **ipatopologysegment** セクションに以下の変数を設定して、ファイルを調整します。
   - **ipaadmin_password** 変数は IdM **admin** のパスワードに設定します。
   - **suffix** 変数は **domain** に設定し ます。**ca** データを再初期化する場合は、変数を **ca** に設定します。
   - **left** の変数をレプリカ合意の左ノードに設定します。
   - レプリカ合意の **right** なノードに正しい変数を設定します。
   - **direction** 変数は再初期化されるデータの方向に設定します。**left-to-right** は、左のノードから適切なノードにデータフローがあることを意味します。
   - **state** 変数が **reinitialized** に設定されていることを確認します。
     以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Playbook to handle topologysegment
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Reinitialize topology segment
    ipatopologysegment:
      ipaadmin_password: "{{ ipaadmin_password }}"
      suffix: domain
      left: server.idm.example.com
      right: replica.idm.example.com
      direction: left-to-right
      state: reinitialized
```

5. ファイルを保存します。

6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory reinitialize-topologysegment-copy.yml
```

関連情報

- レプリカ合意、トポロジー接尾辞、およびトポロジーセグメントの説明 を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-topology.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/topology** ディレクトリーのサンプルの Playbook を参照してください。

39.6. ANSIBLE を使用して IDM にレプリカ合意がないことを確認する手順

Identity Management (IdM) サーバーに保存されているデータは、レプリカ合意に基づいて複製されます。2 台のサーバーでレプリカ合意が設定されている場合は、データを共有します。レプリカ合意は常に双方向のものです。最初のレプリカからサーバーから別のレプリカにデータが複製されるだけでなく、別ののレプリカから最初のレプリカにもデータが複製されます。

以下の手順に従って、2 つのレプリカ間のレプリカ合意が IdM に存在しないことを確認します。この例では、**domain** タイプのレプリカ合意が、**replica01.idm.example.com** と **replica02.idm.example.com** 間で存在させないようにする方法を説明します。

前提条件

- トポロジー内のレプリカの接続 に記載されている IdM トポロジーを設計するための推奨事項を理解しておく。
- IdM **admin** のパスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。

- Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
- この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
- この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。

- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **~/MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible -freeipa/playbooks/topology/** ディレクトリーにある **delete-topologysegment.yml** Ansible Playbook ファイルをコピーします。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/topology/delete-topologysegment.yml
delete-topologysegment-copy.yml
```

3. **delete-topologysegment-copy.yml** ファイルを開いて編集します。
4. **ipatopologysegment** タスクセクションに以下の変数を設定して、ファイルを調整します。
   - **ipaadmin_password** 変数は IdM **admin** のパスワードに設定します。
   - **suffix** 変数は **domain** に設定し ます。また、**ca** データが左右ノードと右のノード間で複製されないようにするには、変数を **ca** に設定します。
   - **left** の変数を、レプリカ合意の左ノードである IdM サーバーの名前に設定します。
   - **右側** の変数を、レプリカ合意の右のノードである IdM サーバーの名前に設定します。
   - **state** 変数は、**absent** に設定されていることを確認します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Playbook to handle topologysegment
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
- name: Delete topology segment
    ipatopologysegment:
      ipaadmin_password: "{{ ipaadmin_password }}"
      suffix: domain
      left: replica01.idm.example.com
      right: replica02.idm.example.com:
      state: absent
```

5. ファイルを保存します。

6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory delete-topologysegment-copy.yml
```

関連情報

- レプリカ合意、トポロジー接尾辞、およびトポロジーセグメントの説明 を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-topology.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/topology** ディレクトリーのサンプルの Playbook を参照してください。

39.7. 関連情報

- Planning the replica topology を参照してください。
- Installing an IdM replica を参照してください。

第40章 ユーザーの外部プロビジョニングのための IDM 設定

システム管理者は、Identity Management (IdM) が、ID 管理用の外部ソリューションでユーザーのプロビジョニングをサポートするように設定できます。

ipa ユーティリティーを使用する代わりに、外部プロビジョニングシステムの管理者は **ldapmodify** ユーティリティーを使用して IdM LDAP にアクセスできます。管理者は、ldapmodify を使用して CLI から、または LDIF ファイル を使用して、個々のステージユーザーを追加できます。

IdM 管理者が外部プロビジョニングシステムを完全に信頼して、検証済みのユーザーだけを追加することを前提とします。ただし、新たにアクティブユーザーを直接追加できるように、外部プロビジョニングシステムの管理者に **User Administrator** の IdM ロールを割り当てなくても構いません。

外部プロビジョニングシステムで作成されたステージユーザーを自動的にアクティブユーザーに移動するように スクリプトを設定 できます。

本章には、以下の項が含まれます。

1. Identity Management (IdM) の準備 して外部プロビジョニングシステムを使用してステージユーザーを IdM に追加する。
2. 外部プロビジョニングシステムが追加したユーザーをステージユーザーからアクティブユーザーに移動する スクリプトを作成 する。
3. 外部プロビジョニングシステムを使用して IdM ステージユーザーを追加する。これには 2 つの方法があります。
   - LDIF ファイルを使用した IdM ステージユーザーの追加
   - ldapmodify を使用した CLI からの IdM ステージユーザーの直接追加

40.1. ステージユーザーアカウントの自動アクティベーションに向けた IDM アカウントの準備

以下の手順では、外部プロビジョニングシステムが使用する 2 つの IdM ユーザーアカウントを設定する方法を説明します。適切なパスワードポリシーが指定されたグループにアカウントを追加すると、外部プロビジョニングシステムが IdM でユーザーのプロビジョニングを管理できるようになります。以下では、ステージユーザーの追加用に外部システムが使用するユーザーアカウントには **provisionator** という名前が付けられます。ステージユーザーを自動アクティベートする時に使用されるユーザーアカウントは **activator** という名前です。

前提条件

- 以下の手順を実行するホストが IdM に登録されている。

手順

1. IdM 管理者としてログインします。

```
$ kinit admin
```

2. ステージユーザーを追加する特権指定して **provisionator** という名前のユーザーを作成します。
   a. provisionator ユーザーアカウントを追加します。

```
$ ipa user-add provisionator --first=provisioning --last=account --password
```

a. provisionator ユーザーに必要な特権を割り当てます。

i. ステージユーザーの追加を管理する **System Provisioning** というカスタムロールを作成します。

```
$ ipa role-add --desc "Responsible for provisioning stage users" "System
Provisioning"
```

ii. **Stage User Provisioning** の特権をロールに追加します。この特権により、ステージユーザーを追加できます。

```
$ ipa role-add-privilege "System Provisioning" --privileges="Stage User Provisioning"
```

iii. provisionator ユーザーをロールに追加します。

```
$ ipa role-add-member --users=provisionator "System Provisioning"
```

iv. provisionator が IdM に存在することを確認します。

```
$ ipa user-find provisionator --all --raw
--------------
1 user matched
--------------
  dn: uid=provisionator,cn=users,cn=accounts,dc=idm,dc=example,dc=com
  uid: provisionator
  [...]
```

3. ユーザーアカウントを管理する特権を指定して **activator** ユーザーを作成します。

a. activator ユーザーアカウントを追加します。

```
$ ipa user-add activator --first=activation --last=account --password
```

b. デフォルトの **User Administrator** ロールにユーザーを追加して、activator ユーザーに必要な特権を付与します。

```
$ ipa role-add-member --users=activator "User Administrator"
```

4. アプリケーションアカウントのユーザーグループを作成します。

```
$ ipa group-add application-accounts
```

5. グループのパスワードポリシーを更新します。以下のポリシーは、アカウントのパスワードの有効期限やロックアウトを防ぎますが、複雑なパスワードを必要とすることでリスクの可能性を低減します。

```
$ ipa pwpolicy-add application-accounts --maxlife=10000 --minlife=0 --history=0 --
minclasses=4 --minlength=8 --priority=1 --maxfail=0 --failinterval=1 --lockouttime=0
```

6. (必要に応じて) パスワードポリシーが IdM に存在することを確認します。

```
$ ipa pwpolicy-show application-accounts
  Group: application-accounts
  Max lifetime (days): 10000
  Min lifetime (hours): 0
  History size: 0
[...]
```

7. アプリケーションアカウントのグループにプロビジョニングアカウントおよびアクティベーションアカウントを追加します。

```
$ ipa group-add-member application-accounts --users={provisionator,activator}
```

8. ユーザーアカウントのパスワードを変更します。

```
$ kpasswd provisionator
$ kpasswd activator
```

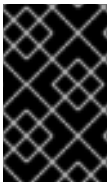
新しい IdM ユーザーのパスワードはすぐに失効するため、パスワードの変更が必要になります。

関連情報:

- コマンドラインを使用したユーザーアカウントの管理 を参照してください。
- Delegating Permissions over Users を参照してください。
- IdM パスワードポリシーの定義 を参照してください。

40.2. IDM ステージユーザーアカウントの自動アクティベーションの設定

この手順では、ステージユーザーをアクティベートするスクリプトを作成する方法を説明します。システムは、指定した間隔でスクリプトを自動的に実行します。これにより、新規ユーザーアカウントが自動的にアクティベートされ、作成直後に使用できます。



重要

以下の手順では、外部プロビジョニングシステムの所有者がユーザーをすでに検証しており、スクリプトが IdM への追加前に IdM 側で追加の検証を必要としていないことを前提としています。

IdM サーバーの 1 つでのみアクティベーションプロセスを有効にするだけで十分です。

前提条件

- **provisionator** アカウントおよび **activator** アカウントが IdM に存在している。詳細は ステージユーザーアカウントの自動アクティベーション用の IdM アカウントの準備 を参照してください。
- この手順を実行する IdM サーバーで root 権限がある。
- IdM 管理者としてログインしている。
- 外部プロビジョニングシステムを信頼している。

手順

1. アカウントのアクティベーション用に keytab ファイルを生成します。

```
# ipa-getkeytab -s server.idm.example.com -p "activator" -k /etc/krb5.ipa-activation.keytab
```

複数の IdM サーバーでアクティベーションプロセスを有効にする場合は、1 つのサーバーだけで keytab ファイルを生成します。次に、keytab ファイルを他のサーバーにコピーします。

2. 以下の内容を含む **/usr/local/sbin/ipa-activate-all** スクリプトを作成して全ユーザーをアクティベートします。

```
#!/bin/bash

kinit -k -i activator

ipa stageuser-find --all --raw | grep "  uid:" | cut -d ":" -f 2 | while read uid; do ipa stageuser-activate ${uid}; done
```

3. **ipa-activate-all** スクリプトのパーミッションおよび所有権を編集して、実行可能なファイルに変更します。

```
# chmod 755 /usr/local/sbin/ipa-activate-all
# chown root:root /usr/local/sbin/ipa-activate-all
```

4. systemd ユニットファイル **/etc/systemd/system/ipa-activate-all.service** を作成して、以下の内容を追加します。

```
[Unit]
Description=Scan IdM every minute for any stage users that must be activated

[Service]
Environment=KRB5_CLIENT_KTNAME=/etc/krb5.ipa-activation.keytab
Environment=KRB5CCNAME=FILE:/tmp/krb5cc_ipa-activate-all
ExecStart=/usr/local/sbin/ipa-activate-all
```

5. systemd タイマー **/etc/systemd/system/ipa-activate-all.timer** を作成して、以下の内容を追加します。

```
[Unit]
Description=Scan IdM every minute for any stage users that must be activated

[Timer]
OnBootSec=15min
OnUnitActiveSec=1min

[Install]
WantedBy=multi-user.target
```

6. 新しい設定を再読み込みします。

```
# systemctl daemon-reload
```

7. **ipa-activate-all.timer** を有効にします。

```
# systemctl enable ipa-activate-all.timer
```

8. **ipa-activate-all.timer** を起動します。

```
# systemctl start ipa-activate-all.timer
```

9. (必要に応じて) **ipa-activate-all.timer** デーモンが実行していることを確認します。

```
# systemctl status ipa-activate-all.timer
● ipa-activate-all.timer - Scan IdM every minute for any stage users that must be activated
   Loaded: loaded (/etc/systemd/system/ipa-activate-all.timer; enabled; vendor preset:
disabled)
   Active: active (waiting) since Wed 2020-06-10 16:34:55 CEST; 15s ago
  Trigger: Wed 2020-06-10 16:35:55 CEST; 44s left

Jun 10 16:34:55 server.idm.example.com systemd[1]: Started Scan IdM every minute for any
stage users that must be activated.
```

40.3. LDIF ファイルで定義されている IDM ステージユーザーの追加

IdM LDAP にアクセスし、LDIF ファイルを使用してステージユーザーを追加するには、次の手順に従います。以下の例では、ユーザーを 1 つ追加していますが、一括モードで 1 つのファイルに複数のユーザーを追加できます。

前提条件

- IdM 管理者が、**provisionator** アカウントとパスワードを作成している。詳細は ステージユーザーアカウントの自動アクティベーション用の IdM アカウントの準備 を参照してください。
- 外部管理者が **provisionator** アカウントのパスワードを知っている。
- LDAP サーバーから IdM サーバーに SSH 接続できる。
- 以下のような、ユーザーのライフサイクルを正しく処理できるように IdM ステージユーザーに割り当てる必要のある最小限の属性セットを提供できる。
  - **識別名** (dn)
  - **共通名** (cn)
  - **名前 (姓)** (sn)
  - **uid**

手順

1. 外部サーバーで、新規ユーザーに関する情報が含まれる LDIF ファイルを作成します。

```
dn: uid=stageidmuser,cn=staged
users,cn=accounts,cn=provisioning,dc=idm,dc=example,dc=com
changetype: add
objectClass: top
objectClass: inetorgperson
uid: stageidmuser
```

```
sn: surname
givenName: first_name
cn: full_name
```

2. 外部サーバーから IdM サーバーへの LDIF ファイルを転送します。

```
$ scp add-stageidmuser.ldif provisionator@server.idm.example.com:/provisionator/
Password:
add-stageidmuser.ldif                                         100%  364
217.6KB/s   00:00
```

3. **SSH** プロトコルを使用して、**provisionator** として IdM サーバーに接続します。

```
$ ssh provisionator@server.idm.example.com
Password:
[provisionator@server ~]$
```

4. IdM サーバーで、provisionator アカウントの Kerberos ticket-granting ticket (TGT) を取得します。

```
[provisionator@server ~]$ kinit provisionator
```

5. -f オプションと LDIF ファイルの名前を指定して **ldapadd** コマンドを入力します。IdM サーバー名とポート番号を指定します。

```
~]$ ldapadd -h server.idm.example.com -p 389 -f  add-stageidmuser.ldif
SASL/GSSAPI authentication started
SASL username: provisionator@IDM.EXAMPLE.COM
SASL SSF: 256
SASL data security layer installed.
adding the entry "uid=stageidmuser,cn=staged
users,cn=accounts,cn=provisioning,dc=idm,dc=example,dc=com"
```

40.4. LDAPMODIFY を使用した CLI からの IDM ステージユーザーの直接追加

Identity Management (IdM) LDAP にアクセスし、**ldapmodify** ユーティリティーを使用してステージユーザーを追加するには、次の手順に従います。

前提条件

- IdM 管理者が、**provisionator** アカウントおよびパスワードを作成している。詳細は ステージユーザーアカウントの自動アクティベーション用の IdM アカウントの準備 を参照してください。
- 外部管理者が **provisionator** アカウントのパスワードを知っている。
- LDAP サーバーから IdM サーバーに SSH 接続できる。
- 以下のような、ユーザーのライフサイクルを正しく処理できるように IdM ステージユーザーに割り当てる必要のある最小限の属性セットを提供できる。
  - **識別名** (dn)

- **共通名** (cn)
- **名前 (姓)** (sn)
- **uid**

手順

1. IdM の ID および認証情報を使用して、**SSH** プロトコルを使用して IdM サーバーに接続します。

```
$ ssh provisionator@server.idm.example.com
Password:
[provisionator@server ~]$
```

2. 新規ステージユーザーを追加するロールを持つ IdM ユーザーである **provisionator** アカウントの TGT を取得します。

```
$ kinit provisionator
```

3. **ldapmodify** コマンドを入力し、Generic Security Services API (GSSAPI) を認証に使用する Simple Authentication and Security Layer (SASL) メカニズムとして指定します。IdM サーバーとポート名を指定します。

```
# ldapmodify -h server.idm.example.com -p 389 -Y GSSAPI
SASL/GSSAPI authentication started
SASL username: provisionator@IDM.EXAMPLE.COM
SASL SSF: 56
SASL data security layer installed.
```

4. 追加するユーザーの **dn** を入力します。

```
dn: uid=stageuser,cn=staged
users,cn=accounts,cn=provisioning,dc=idm,dc=example,dc=com
```

5. 実行する変更の種類として **add** を入力します。

```
changetype: add
```

6. ユーザーのライフサイクルを正しく処理できるようにするために必要な LDAP オブジェクトクラスのカテゴリーを指定します。

```
objectClass: top
objectClass: inetorgperson
```

追加のオブジェクトクラスを指定できます。

7. ユーザーの **uid** を入力します。

```
uid: stageuser
```

8. ユーザーの **cn** を入力します。

```
cn: Babs Jensen
```

9. ユーザーの名前 (姓) を入力します。

```
sn: Jensen
```

10. **Enter** を再度押して、これがエントリーの最後であることを確認します。

```
[Enter]

adding new entry "uid=stageuser,cn=staged
users,cn=accounts,cn=provisioning,dc=idm,dc=example,dc=com"
```

11. **Ctrl** + **C** を使用して接続を終了します。

検証手順

ステージエントリーの内容を検証し、プロビジョニングシステムにより、必要な全 POSIX 属性が追加され、ステージエントリーのアクティベートの準備ができていることを確認します。

- 新規ステージユーザーの LDAP 属性を表示するには、**ipa stageuser-show --all --raw** コマンドを実行します。

```
$ ipa stageuser-show stageuser --all --raw
  dn: uid=stageuser,cn=staged
users,cn=accounts,cn=provisioning,dc=idm,dc=example,dc=com
  uid: stageuser
  sn: Jensen
  cn: Babs Jensen
  has_password: FALSE
  has_keytab: FALSE
  nsaccountlock: TRUE
  objectClass: top
  objectClass: inetorgperson
  objectClass: organizationalPerson
  objectClass: person
```

1. ユーザーは、**nsaccountlock** 属性を使用して明示的に無効化されている点に注意してください。

40.5. 関連情報

- Using ldapmodify to manage IdM users externally を参照してください。

第41章 LDAPMODIFY を使用した IDM ユーザーの外部管理

IdM 管理者は、**ipa** コマンドを使用してディレクトリーの内容を管理できます。または、**ldapmodify** コマンドを使用して同様に管理することもできます。このコマンドを対話的に使用して、すべてのデータをコマンドラインで直接指定できます。ファイル内のデータを LDAP データ交換形式 (LDIF) で **ldapmodify** コマンドに提供することもできます。

41.1. IDM ユーザーアカウントを外部で管理するためのテンプレート

次のテンプレートは、IdM でのさまざまなユーザー管理操作に使用できます。これらのテンプレートでは、以下の目的を達成するために **ldapmodify** を使用して変更する必要のある属性がどれであるかがわかります。

- 新規ステージユーザーの追加
- ユーザーの属性変更
- ユーザーの有効化
- ユーザーの無効化
- ユーザーの保存

テンプレートは LDAP データ交換形式 (LDIF) でフォーマットされます。LDIF は、LDAP ディレクトリーのコンテンツおよび更新リクエストを表す標準的なプレーンテキストデータ交換形式です。

テンプレートを使用し、プロビジョニングシステムの LDAP プロバイダーを設定して、IdM ユーザーアカウントを管理できます。

詳細な手順例は、以下のセクションを参照してください。

- LDIF ファイルで定義されている IdM ステージユーザーの追加
- ldapmodify を使用した CLI からの IdM ステージユーザーの直接追加
- ldapmodify での IdM ユーザーの保存

新規ステージユーザーを追加するためのテンプレート

- **UID および GID が自動的に割り当てられた** ユーザーを追加するためのテンプレート。作成したエントリーの識別名 (DN) は **uid=user_login** で開始する必要があります。

```
dn: uid=user_login,cn=staged
users,cn=accounts,cn=provisioning,dc=idm,dc=example,dc=com
changetype: add
objectClass: top
objectClass: inetorgperson
uid: user_login
sn: surname
givenName: first_name
cn: full_name
```

- **UID と GID が静的に割り当てられている** ユーザーを追加するためのテンプレート

```
dn: uid=user_login,cn=staged
```

```
users,cn=accounts,cn=provisioning,dc=idm,dc=example,dc=com
changetype: add
objectClass: top
objectClass: person
objectClass: inetorgperson
objectClass: organizationalperson
objectClass: posixaccount
uid: user_login
uidNumber: UID_number
gidNumber: GID_number
sn: surname
givenName: first_name
cn: full_name
homeDirectory: /home/user_login
```

ステージユーザーの追加時に IdM オブジェクトクラスを指定する必要はありません。IdM は、ユーザーのアクティベート後にこれらのクラスを自動的に追加します。

既存ユーザーを変更するためのテンプレート

- **ユーザーの属性の変更**:

```
dn: distinguished_name
changetype: modify
replace: attribute_to_modify
attribute_to_modify: new_value
```

- **ユーザーの無効化**:

```
dn: distinguished_name
changetype: modify
replace: nsAccountLock
nsAccountLock: TRUE
```

- **ユーザーの有効化**

```
dn: distinguished_name
changetype: modify
replace: nsAccountLock
nsAccountLock: FALSE
```

nssAccountLock 属性を更新してもステージユーザーおよび保存済みユーザーには影響はありません。更新操作が正常に完了しても、属性値は **nssAccountLock: TRUE** のままになります。

- **ユーザーの保持**

```
dn: distinguished_name
changetype: modrdn
newrdn: uid=user_login
deleteoldrdn: 0
newsuperior: cn=deleted users,cn=accounts,cn=provisioning,dc=idm,dc=example,dc=com
```

注記

ユーザーの変更前に、ユーザーのログインを検索してユーザーの識別名 (DN) を取得します。以下の例では、**user_allowed_to_modify_user_entries** ユーザーは、**activator** または IdM 管理者など、ユーザーおよびグループの情報の変更を許可されたユーザーです。以下の例のパスワードは、このユーザーのパスワードです。

```
[...]
# ldapsearch -LLL -x -D
"uid=user_allowed_to_modify_user_entries,cn=users,cn=accounts,dc=idm,dc=example,dc=com" -w "Secret123" -H ldap://r8server.idm.example.com -b
"cn=users,cn=accounts,dc=idm,dc=example,dc=com" uid=test_user
dn: uid=test_user,cn=users,cn=accounts,dc=idm,dc=example,dc=com
memberOf: cn=ipausers,cn=groups,cn=accounts,dc=idm,dc=example,dc=com
```

41.2. IDM グループアカウントを外部で管理するためのテンプレート

次のテンプレートは、IdM でのさまざまなユーザーグループ管理操作に使用できます。これらのテンプレートでは、以下の目的を達成するために **ldapmodify** を使用して変更する必要のある属性がどれであるかがわかります。

- 新規グループの作成
- 既存グループの削除
- グループへのメンバーの追加
- グループからメンバーの削除

テンプレートは LDAP データ交換形式 (LDIF) でフォーマットされます。LDIF は、LDAP ディレクトリーのコンテンツおよび更新リクエストを表す標準的なプレーンテキストデータ交換形式です。

テンプレートを使用して、プロビジョニングシステムの LDAP プロバイダーを設定して、IdM グループアカウントを管理できます。

新規グループの作成

```
dn: cn=group_name,cn=groups,cn=accounts,dc=idm,dc=example,dc=com
changetype: add
objectClass: top
objectClass: ipaobject
objectClass: ipausergroup
objectClass: groupofnames
objectClass: nestedgroup
objectClass: posixgroup
uid: group_name
cn: group_name
gidNumber: GID_number
```

グループの変更

- **既存グループの削除**

```
dn: group_distinguished_name
changetype: delete
```

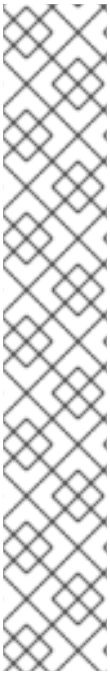
- **グループへのメンバーの追加**

```
dn: group_distinguished_name
changetype: modify
add: member
member: uid=user_login,cn=users,cn=accounts,dc=idm,dc=example,dc=com
```

ステージまたは保存済みユーザーをグループに追加しないでください。更新操作が正常に完了しても、ユーザーはグループのメンバーとしては更新されません。アクティブなユーザーのみがグループに所属できます。

- **グループからのメンバーの削除**

```
dn: distinguished_name
changetype: modify
delete: member
member: uid=user_login,cn=users,cn=accounts,dc=idm,dc=example,dc=com
```



注記

グループの変更前に、グループ名で検索してグループの識別名 (DN) を取得します。

```
# ldapsearch -YGSSAPI -H ldap://server.idm.example.com -b
"cn=groups,cn=accounts,dc=idm,dc=example,dc=com" "cn=group_name"
dn: cn=group_name,cn=groups,cn=accounts,dc=idm,dc=example,dc=com
ipaNTSecurityIdentifier: S-1-5-21-1650388524-2605035987-2578146103-11017
cn: testgroup
objectClass: top
objectClass: groupofnames
objectClass: nestedgroup
objectClass: ipausergroup
objectClass: ipaobject
objectClass: posixgroup
objectClass: ipantgroupattrs
ipaUniqueID: 569bf864-9d45-11ea-bea3-525400f6f085
gidNumber: 1997010017
```

41.3. LDAPMODIFY コマンドの対話的な使用

対話モードで LDAP (Lightweight Directory Access Protocol) エントリーを変更できます。

手順

1. コマンド行で、**ldapmodify** コマンドの後に LDAP Data Interchange Format (LDIF) ステートメントを入力します。

例41.1testuser の電話番号の変更

```
# ldapmodify -Y GSSAPI -H ldap://server.example.com
dn: uid=testuser,cn=users,cn=accounts,dc=example,dc=com
```

```
changetype: modify
replace: telephoneNumber
telephonenumber: 88888888
```

-Y オプションを使用するには、Kerberos チケットを取得する必要があることに注意してください。

2. **Ctlr+D** を押してインタラクティブモードを終了します。

3. または、**ldapmodify** コマンドの後に LDIF ファイルを指定します。

例41.2 ldapmodify コマンドは、LDIF ファイルから変更データを読み取ります。

```
# ldapmodify -Y GSSAPI -H ldap://server.example.com -f ~/example.ldif
```

関連情報

- **ldapmodify** コマンドの使用方法に関する詳細は、**ldapmodify(1)** の man ページを参照してください。
- **LDIF** 構造の詳細は、**ldif(5)** man ページを参照してください。

41.4. LDAPMODIFY での IDM ユーザーの保存

ldapmodify を使用して IdM ユーザーを保存する (従業員が退職した後にユーザーアカウントを非アクティブ化する) には、次の手順に従います。

前提条件

- ユーザーを保存するロールが割り当てられた IdM ユーザーとして認証できる。

手順

1. ユーザーを保存するロールを持つ IdM ユーザーとしてログインします。

```
$ kinit admin
```

2. **ldapmodify** コマンドを入力し、Generic Security Services API (GSSAPI) を認証に使用する Simple Authentication and Security Layer (SASL) メカニズムとして指定します。

```
# ldapmodify -Y GSSAPI
SASL/GSSAPI authentication started
SASL username: admin@IDM.EXAMPLE.COM
SASL SSF: 256
SASL data security layer installed.
```

3. 保存するユーザーの **dn** を入力します。

```
dn: uid=user1,cn=users,cn=accounts,dc=idm,dc=example,dc=com
```

4. 実行する変更のタイプとして **modrdn** を入力します。

```
changetype: modrdn
```

5. ユーザーの **newrdn** を指定します。

```
newrdn: uid=user1
```

6. 以下のようにユーザーの保存を指定します。

```
deleteoldrdn: 0
```

7. **新しい上位 DN** を指定します。

```
newsuperior: cn=deleted users,cn=accounts,cn=provisioning,dc=idm,dc=example,dc=com
```

ユーザーを保存すると、そのエントリーをディレクトリー情報ツリー (DIT) 内の新しい場所に移動します。上記の理由から、新しい親エントリーの DN を新しい上位 DN として指定する必要があります。

8. **Enter** を再度押して、これがエントリーの最後であることを確認します。

```
[Enter]

modifying rdn of entry "uid=user1,cn=users,cn=accounts,dc=idm,dc=example,dc=com"
```

9. **Ctrl** + **C** を使用して接続を終了します。

検証手順

- 保存済みユーザーをリスト表示して、ユーザーが保存されていることを確認します。

```
$ ipa user-find --preserved=true
--------------
1 user matched
--------------
  User login: user1
  First name: First 1
  Last name: Last 1
  Home directory: /home/user1
  Login shell: /bin/sh
  Principal name: user1@IDM.EXAMPLE.COM
  Principal alias: user1@IDM.EXAMPLE.COM
  Email address: user1@idm.example.com
  UID: 1997010003
  GID: 1997010003
  Account disabled: True
  Preserved user: True
----------------------------
Number of entries returned 1
----------------------------
```

第42章 IDM CLI でのホストの管理

本章では、Identity Management (IdM) の ホスト および ホストエントリー と、IdM CLI でホストとホストエントリーを管理する際に以下の操作が実行されます。

- ホストの登録
- IdM ホストエントリーの追加
- IdM ホストエントリーの削除
- ホストの再登録
- ホストの名前変更
- ホストの無効化
- ホストの再有効化

本章には、前提条件、コンテキスト、および操作の結果の 概要表 も含まれています。

42.1. IDM のホスト

Identity Management (IdM) は、以下の ID を管理します。

- ユーザー
- サービス
- ホスト

ホストはマシンを表します。ホストには、IdM LDAP に IdM ID となるエントリーがあります。これは IdM サーバーの 389 Directory Server のインスタンスです。

IdM LDAP のホストエントリーは、その他のホストとドメイン内のサービスとの関係を確立するために使用されます。この関係では、ドメイン内ホストの認可および制御の **委譲** が不可欠な要素です。ホストは、**ホストベースのアクセス制御** (HBAC) ルールで使用できます。

IdM ドメインは、共通の ID 情報、共通ポリシー、および共有サービスを使用して、マシン間で共通性を確立します。ドメインのクライアントとしてのドメイン機能に属するマシンです。これは、ドメインが提供するサービスを使用することを意味します。IdM ドメインは、マシン専用の 3 つの主なサービスを提供します。

- DNS
- Kerberos
- 証明書の管理

IdM のホストは、そのホストで実行しているサービスと密接に接続されています。

- サービスエントリーは、ホストに関連付けられています。
- ホストには、ホストとサービスの両方の Kerberos プリンシパルが格納されます。

42.2. ホスト登録

本セクションでは、ホストを IdM クライアントとして登録し、登録中および登録後に何が起こるかを説明します。本セクションでは、IdM ホストの登録と、IdM ユーザーの登録を比較します。また、本セクションでは、ホストで利用可能な代替タイプの認証も概説します。

ホストの登録は、以下の要素で構成されます。

- IdM LDAP でのホストエントリーの作成: 場合によっては、IdM CLI で **ipa host-add** コマンドを使用するか、同等の IdM Web UI 操作 を使用します。
- ホストで IdM サービス (System Security Services Daemon (SSSD)、Kerberos、certmonger など) を設定し、ホストを IdM ドメインに参加させる。

2 つのアクションは、個別に実行することも一緒に実行することもできます。

個別に実行すると、異なるレベルの特権を持つ 2 人のユーザー間で 2 つのタスクを分割できます。これは、一括デプロイメントに役立ちます。

ipa-client-install コマンドは、2 つのアクションを一緒に実行できます。コマンドは、そのエントリーがまだ存在していない場合は IdM LDAP にホストエントリーを作成し、そのホストに Kerberos サービスと SSSD サービスの両方を設定します。このコマンドにより、ホストが IdM ドメイン内に移動し、接続先の IdM サーバーを識別できるようになります。IdM が管理する DNS ゾーンにホストが属する場合は、**ipa-client-install** でホストに DNS レコードも追加します。コマンドはクライアントで実行する必要があります。

42.3. ホストの登録に必要なユーザー権限

ホスト登録操作では、権限のないユーザーが不要なマシンを IdM ドメインに追加しないように、認証が必要になります。必要な特権は、次のようないくつかの要因によって異なります。

- ホストエントリーが **ipa-client-install** の実行とは別に作成される場合
- ワンタイムパスワード (OTP) が登録に使用される場合

必要に応じて IdM LDAP にホストエントリーを手動で作成するためのユーザー特権

CLI コマンド **ipa host-add** または IdM Web UI を使用して、IdM LDAP にホストエントリーを作成するのに必要なユーザー特権は **ホストの管理者** です。**ホスト管理者** の特権は、**IT スペシャリスト** ロールから取得できます。

クライアントを IdM ドメインに参加させるためのユーザー特権

ホストは、**ipa-client-install** コマンドの実行時に IdM クライアントとして設定されます。**ipa-client-install** コマンドの実行に必要な認証情報のレベルは、以下のような登録シナリオのどれに該当するかによって異なります。

- IdM LDAP のホストエントリーが存在しません。このシナリオでは、管理者の完全な認証情報または **ホスト管理者** ロールが必要です。完全な管理者とは **admins** グループのメンバーです。**ホスト管理者** ロールは、ホストの追加およびホストの登録の特権を提供します。このシナリオの詳細は ユーザー認証情報を使用したクライアントのインストール: 対話的なインストール を参照してください。
- IdM LDAP のホストエントリーが存在します。このシナリオでは、**ipa-client-install** を正常に実行するには、制限された管理者の認証情報が必要です。この場合、制限されている管理者には、**ホストの登録** 特権を提供する **登録管理者** ロールがあります。詳細は ユーザー認証情報を使用したクライアントのインストール: 対話的なインストール を参照してください。
- IdM LDAP にホストエントリーが存在し、完全または限定された管理者により、ホストの OTP

が生成されました。このシナリオでは、正しい OTP を指定して **--password** オプションを指定して **ipa-client-install** コマンドを実行すると、通常のユーザーとして IdM クライアントをインストールできます。詳細は ワンタイムパスワードでクライアントのインストール: 対話的なインストール を参照してください。

登録後、IdM ホストは、IdM リソースにアクセスできるように、新しいセッションをすべて認証します。IdM サーバーがマシンを信頼し、そのマシンにインストールされているクライアントソフトウェアからの IdM 接続を受け入れるには、マシン認証が必要です。クライアントを認証すると、IdM サーバーはそのリクエストに応答できます。

42.4. IDM ホストとユーザーの登録と認証: 比較

IdM のユーザーとホストの間には多くの類似点があります。この類似点には、登録ステージで見られるものと、デプロイメントステージでの認証に関係するものがあります。

- 登録段階 (ユーザーおよびホストの登録):
  - 管理者は、ユーザーまたはホストが実際に IdM に参加する前に、ユーザーとホストの LDAP エントリーを作成できます。コマンドは、ステージユーザーの場合は **ipa stageuser-add** で、ホストの場合は **ipa host-add** です。
  - ホストで **ipa-client-install** コマンドを実行すると、**キーテーブル** (または略してキータブ)、(ある程度ユーザーパスワードに類似する) 対称キーを含むファイルが作成され、ホストが IdM レルムに参加します。同様に、ユーザーはアカウントをアクティブ化するときにパスワードを作成するように求められ、IdM レルムに参加します。
  - ユーザーパスワードは、ユーザーのデフォルトの認証方法ですが、キータブはホストのデフォルトの認証方法です。キータブは、ホストのファイルに保存されます。

表42.1 ユーザーおよびホストの登録

アクション	ユーザー	ホスト
登録前	$ ipa stageuser-add user_name [--password]	$ ipa host-add host_name [--random]
アカウントのアクティブ化	$ ipa stageuser-activate user_name	$ ipa-client install [--password] (ホスト自体で実行する必要があります)

- デプロイメント段階 (ユーザーおよびホストセッションの認証):
  - ユーザーが新しいセッションを開始すると、ユーザーはパスワードを使用して認証を行います。同様に、切り替え時に、ホストがそのキータブファイルを提示して認証を行います。SSSD (System Security Services Daemon) は、このプロセスをバックグラウンドで管理します。
  - 認証が成功すると、ユーザーまたはホストは、Kerberos チケット発行許諾チケット (TGT) を取得します。
  - 次に、TGT を使用して、特定のサービスの特定のチケットを取得します。

表42.2 ユーザーおよびホストセッションの認証

	ユーザー	ホスト
認証のデフォルト手段	**パスワード**	**キータブ**
セッションの開始 (通常のユーザー)	$ **kinit user_name**	**[ホストへの切り替え]**
認証成功の結果	**TGT** は、特定サービスへのアクセスの取得に使用されます。	**TGT** は、特定サービスへのアクセスの取得に使用されます。

TGT およびその他の Kerberos チケットは、サーバーにより定義された Kerberos サービスおよびポリシーの一部として生成されます。Kerberos チケットの最初の付与、Kerberos 認証情報の更新、および Kerberos セッションの破棄もすべて IdM サービスにより自動的に処理されます。

IdM ホストの代替認証オプション

キータブとは別に、IdM は、その他の 2 つのタイプのマシン認証にも対応しています。

- SSH 鍵。ホストの SSH 公開キーが作成され、ホストエントリーにアップロードされます。そこから、SSSD (System Security Services Daemon) は IdM を ID プロバイダーとして使用し、OpenSSH およびその他のサービスと一緒に機能して、IdM の中央にある公開鍵を参照できます。
- 機械の証明書。この場合、マシンは IdM サーバーの認証局により発行され、IdM の Directory Server に保存されている SSL 証明書を使用します。次に、証明書はマシンに送信され、サーバーに対する認証時に提示されます。クライアントでは、証明書は certmonger というサービスにより管理されます。

42.5. ホスト操作

次のセクションでは、ホストの登録と有効化に関連する最も一般的な操作、前提条件、コンテキスト、およびこれらの操作を実行した結果について説明します。

表42.3 ホスト操作パート 1

アクション	アクションの前提条件	コマンド実行が妥当な時期	システム管理者によりアクションの実行方法と実行するコマンド
クライアントの登録	詳細は、**Identity Management のインストール** の Identity Management クライアントをインストールするためのシステムの準備 を参照してください。	ホストが IdM レルムに参加する時	IdM ドメインでマシンをクライアントとして登録する場合は、2 つのプロセスがあります。**ipa host-add** コマンドを実行すると、クライアントにホストエントリーが作成されて (389 Directory Server インスタンスに格納されます) から、クライアントをプロビジョニングするキータブが作成されます。プロセスは、いずれも **ipa-client-install** コマンドで自動的に実行されます。この手順は個別に実行することもできます。これにより、管理者は、クライアントを実際に設定する前にマシンと IdM を準備できます。これにより、一括デプロイメントなど、より柔軟な設定シナリオが可能になります。

アクション	アクションの前提条件	コマンド実行が妥当な時期	システム管理者によりアクションの実行方法と実行するコマンド
クライアントの無効化	ホストに、IdM のエントリーが必要です。ホストにはアクティブなキータブが必要です。	(メンテナンス目的などで) IdM レルムからホストを一時的に削除する時	**ipa host-disable host_name**
クライアントの有効化	ホストに、IdM のエントリーが必要です。	一時的に無効にしたホストが再びアクティブになるようにする時	**ipa-getkeytab**
クライアントの再登録	ホストに IdM へのエントリーが必要です。	元のホストが失われていて、同じホスト名でホストがインストールされている時	**ipa-client-install --keytab** または **ipa-client-install --force-join**
クライアントの登録解除	ホストに、IdM のエントリーが必要です。	IdM レルムからホストを完全に削除する時	**ipa-client-install --uninstall**

表42.4 ホスト操作パート 2

アクション	管理者はコマンドを実行するマシン	アクションの実行時に発生する動作、ホストが IdM で機能する場合の結果、および導入または削除される制限
クライアントの登録	2 ステップでの登録の場合 - **ipa host-add** は、任意の IdM クライアントで実行できます。**ipa-client-install** の次のステップは、クライアント自体で実行する必要があります。	デフォルトでは、SSSD が認証と認可のために IdM サーバーに接続するように設定します。必要に応じて、PAM (Pluggable Authentication Module) と NSS (Name Switching Service) を、Kerberos および LDAP を介して IdM サーバーと連携するように設定することができます。
クライアントの無効化	IdM の任意のマシン (ホスト自体も含む)。	ホストの Kerberos 鍵および SSL 証明書が無効にされており、ホストで実行しているすべてのサービスが無効になります。

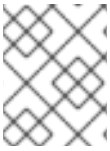
アクション	管理者はコマンドを実行するマシン	アクションの実行時に発生する動作、ホストが IdM で機能する場合の結果、および導入または削除される制限
クライアントの有効化	IdM のマシン。無効なホストで実行する場合は、LDAP 認証情報を提供する必要があります。	ホストの Kerberos 鍵と SSL 証明書が再び有効になり、ホストで実行しているすべての IdM サービスが再度有効になります。
クライアントの再登録	再登録するホスト。LDAP 認証情報を提供する必要があります。	ホスト用に新しい Kerberos キーが生成され、以前のキーが置き換えられます。
クライアントの登録解除	登録解除するホスト。	このコマンドは IdM の設定を解除し、マシンを以前の状態に戻そうとします。このプロセスには、IdM サーバーからホストの登録を解除するステップが含まれます。登録解除には、IdM サーバーでプリンシパルキーを無効にすることで設定されます。**/etc/krb5.keytab** (**host/<fqdn>@REALM**) のマシンプリンシパルは、登録解除する IdM サーバーに認証するのに使用されます。このプリンシパルが存在しない場合は登録解除に失敗します。管理者はホストプリンシパル (**ipa host-disable <fqdn>**) を無効にする必要があります。

42.6. IDM LDAP のホストエントリー

Identity Management (IdM) ホストエントリーには、ホストに関する情報とホストに含めることができる属性が含まれています。

LDAP ホストエントリーは、IdM 内のクライアントに関するすべての関連情報が含まれます。

- ホストに関連付けられたサービスエントリー
- ホストおよびサービスプリンシパル
- アクセス制御ルール
- 物理的な場所やオペレーティングシステムなどのマシン情報



注記

IdM Web UI の **Identity → Hosts** タブには、IdM LDAP に保存されている特定のホストに関する情報がすべて表示されないことに注意してください。

ホストエントリー設定プロパティー

ホストエントリーには、ホストに関する情報 (物理的な場所、MAC アドレス、鍵、証明書など、システム設定を除く) を含めることができます。

この情報は、ホストエントリーが手動で作成された場合に、作成時に設定できます。また、この情報のほとんどは、ホストがそのドメインに登録してからホストエントリーに追加できます。

表42.5 ホスト設定プロパティー

UI フィールド	コマンドラインオプション	説明
説明	**--desc**=description	ホストの説明。
局所性	**--locality**=locality	ホストの地理的な場所。
場所	**--location**=location	データセンターのラックなど、ホストの物理的な場所。
プラットフォーム	**--platform**=string	ホストのハードウェアまたはアーキテクチャー。
オペレーティングシステム	**--os**=string	ホストのオペレーティングシステムとバージョン。
MAC アドレス	**--macaddress**=address	ホストの MAC アドレス。これは多値属性です。MAC アドレスは、NIS プラグインにより、ホスト用の NIS の **ethers** マップを作成するために使用されます。
SSH 公開鍵	**--sshpubkey**=string	ホストの完全 SSH 公開鍵。これは複数値の属性であるため、複数の鍵を設定できます。
プリンシパル名 (編集不可)	**--principalname**=principal	ホストの Kerberos プリンシパル名。**-p** に別のプリンシパルを明示的に設定しない限り、クライアントのインストール時にホスト名がデフォルトになります。これはコマンドラインツールを使用して変更できますが、UI で変更することはできません。
ワンタイムパスワードの設定	**--password**=string	このオプションは、一括登録で使用できるホストのパスワードを設定します。
-	**--random**	このオプションは、一括登録で使用されるランダムパスワードを生成します。
-	**--certificate**=string	ホストの証明書ブロブ。
-	**--updatedns**	これにより、IP アドレスが変更した場合に、ホストが DNS エントリーを動的に更新できるかどうかが設定されます。

42.7. IDM CLI で IDM ホストエントリーの追加

コマンドラインインターフェイス (CLI) を使用して Identity Management (IdM) にホストエントリーを追加するには、次の手順に従います。

ホストエントリーは、**host-add** コマンドを使用して作成されます。このコマンドは、ホストエントリーを IdM Directory Server に追加します。CLI で **ipa help host** を入力し、**ipa host** の man ページで、**host-add** で利用可能なオプションの完全リストを取得します。

ホストを IdM に追加する場合は、いくつかのシナリオがあります。

- 最も基本的な方法として、クライアントホスト名を指定してクライアントを Kerberos レルムに追加し、IdM LDAP サーバーにエントリーを作成します。

```
$ ipa host-add client1.example.com
```

- DNS を管理するために IdM サーバーを設定している場合は、**--ip-address** オプションを使用して DNS リソースレコードにホストを追加します。

例42.1 静的 IP アドレスを持つホストエントリーの作成

```
$ ipa host-add --ip-address=192.168.166.31 client1.example.com
```

- 追加するホストに静的 IP アドレスがない場合や、クライアントの設定時に IP アドレスが不明な場合は、**ipa host-add** コマンドで **--force** オプションを使用します。

例42.2 DHCP でホストエントリーの作成

```
$ ipa host-add --force client1.example.com
```

たとえば、ラップトップは IdM クライアントとして事前設定されている場合がありますが、設定時には IP アドレスがありません。**--force** を使用すると、基本的に IdM DNS サービスにプレースホルダーエントリーが作成されます。DNS サービスがレコードを動的に更新すると、ホストの現在の IP アドレスが検出され、DNS レコードが更新されます。

42.8. IDM CLI でホストエントリーの削除

- **host-del** コマンドを使用して、ホストレコードを削除します。IdM ドメインに DNS が統合されている場合は、**--updatedns** オプションを使用して、あらゆる種類のホストに関連するレコードを DNS から削除します。

```
$ ipa host-del --updatedns client1.example.com
```

42.9. IDENTITY MANAGEMENT クライアントの再登録

本セクションでは、Identity Management クライアントを再登録するさまざまな方法を説明します。

42.9.1. IdM におけるクライアントの再登録

再登録の間、クライアントは新しい鍵 (Kerberos および SSH) を生成しますが、LDAP データベースのクライアントのアイデンティティーは変更されません。再登録後、ホストは、IdM サーバーとの接続を失う前と同じ **FQDN** を持つ同じ LDAP オブジェクトに、キーとその他の情報を保持します。

重要

ドメインエントリーがアクティブなクライアントのみを再登録できます。クライアントをアンインストール (**ipa-client-install --uninstall** を使用) した場合や、ホストエントリーを無効 (**ipa host-disable** を使用) にした場合は再登録できません。

クライアントの名前を変更すると、再登録することができません。これは、Identity Management では LDAP にあるクライアントのエントリーのキー属性はクライアントのホスト名 **FQDN** であるためです。クライアントの再登録中はクライアントの LDAP オブジェクトは変更されませんが、クライアントの名前を変更すると、クライアントの鍵とその他の情報は新しい **FQDN** を持つ異なる LDAP オブジェクトに格納されます。そのため、IdM からホストをアンインストールし、ホストのホスト名を変更して、新しい名前で IdM クライアントとしてインストールするのが、クライアントの名前を変更する唯一の方法です。クライアント名を変更する方法は、Identity Management クライアントシステムの名前の変更 を参照してください。

クライアント再登録中に行われること

Identity Management は再登録中に以下を行います。

- 元のホスト証明書を破棄する。
- 新規の SSH 鍵を作成する。
- 新規のキータブを生成する。

42.9.2. ユーザー認証情報でクライアントの再登録: 対話的な再登録

許可されたユーザーの認証情報を使用して、Identity Management クライアントを対話的に再登録するには、次の手順に従います。

1. 同じホスト名のクライアントマシンを再作成します。
2. クライアントマシンで **ipa-client-install --force-join** コマンドを実行します。

```
# ipa-client-install --force-join
```

3. スクリプトにより、アイデンティティーがクライアントの再登録に使用されるユーザーの入力が求められます。たとえば、登録管理者 (Enrollment Administrator) ロールを持つ **hostadmin** ユーザーなどが該当します。

```
User authorized to enroll computers: hostadmin
Password for hostadmin@EXAMPLE.COM:
```

関連情報

- **Identity Management のインストール** の ユーザー認証情報でクライアントのインストール: 対話的なインストール を参照してください。

42.9.3. クライアントのキータブでクライアントの再登録: 非対話的な再登録

前提条件

- **/tmp** や **/root** などのディレクトリーに元のクライアントキータブファイルをバックアップします。

手順

以下の手順に従って、クライアントシステムのキータブを使用して、Identity Management (IdM) クライアントを非対話的に再登録します。たとえば、クライアントのキータブを使用した再登録は自動インストールに適しています。

1. 同じホスト名のクライアントマシンを再作成します。
2. バックアップした場所から、再作成したクライアントマシンの **/etc/** ディレクトリーにキータブファイルをコピーします。
3. **ipa-client-install** ユーティリティーを使用してクライアントを再登録し、**--keytab** オプションでキータブの場所を指定します。

```
# ipa-client-install --keytab /etc/krb5.keytab
```

注記

登録を開始するために認証する場合は、**--keytab** オプションで指定するキータブのみが使用されます。再登録中、IdM はクライアントに対して新しいキータブを生成します。

42.9.4. インストール後の Identity Management クライアントのテスト

コマンドラインインターフェイスにより、**ipa-client-install** が正常に実行されたことが通知されますが、独自のテストを行うこともできます。

Identity Management クライアントが、サーバーに定義したユーザーに関する情報を取得できることをテストするには、サーバーに定義したユーザーを解決できることを確認します。たとえば、デフォルトの **admin** ユーザーを確認するには、次のコマンドを実行します。

```
[user@client1 ~]$ id admin
uid=1254400000(admin) gid=1254400000(admins) groups=1254400000(admins)
```

認証が適切に機能することをテストするには、別の IdM ユーザーで **su -** を実行します。

```
[user@client1 ~]$ su - idm_user
Last login: Thu Oct 18 18:39:11 CEST 2018 from 192.168.122.1 on pts/0
[idm_user@client1 ~]$
```

42.10. IDENTITY MANAGEMENT クライアントシステムの名前の変更

ここでは、Identity Management クライアントシステムのホスト名を変更する方法を説明します。

警告

クライアントの名前は手動で変更します。ホスト名の変更が絶対に必要である場合のみ実行してください。

Identity Management クライアントの名前を変更するには、以下を行う必要があります。

1. ホストを準備します。詳細は、名前を変更するための IdM クライアントの準備 を参照してください。
2. ホストから IdM クライアントをアンインストールします。詳細は、Identity Management クライアントのアンインストール を参照してください。
3. ホストの名前を変更します。詳細は、ホストシステムの名前変更 を参照してください。
4. 新しい名前でホストに IdM クライアントをインストールします。詳細は、**Identity Management のインストール** の Identity Management クライアントのインストール を参照してください。
5. IdM クライアントのインストール後にホストを設定します。詳しくはサービスの再追加、証明書の再生成、およびホストグループの再追加 を参照してください。

42.10.1. 名前を変更するための IdM クライアントの準備

現在のクライアントをアンインストールする前に、クライアントの設定を書き留めます。新しいホスト名のマシンを再登録した後にこの設定を適用します

- マシンで実行しているサービスを特定します。
  - **ipa service-find** コマンドを使用して、証明書のあるサービスを特定して出力します。

    ```
    $ ipa service-find old-client-name.example.com
    ```

  - さらに、各ホストには ipa service-find の出力に表示されないデフォルトの **host service** があります。ホストサービスのサービスプリンシパルは **ホストプリンシパル** とも呼ばれ、**host/old-client-name.example.com** になります。
- **ipa service-find old-client-name.example.com** により表示されるすべてのサービスプリンシパルは、**old-client-name.example.com** 上の対応するキータブの場所を決定します。

  ```
  # find / -name "*.keytab"
  ```

  クライアントシステムの各サービスには、**ldap/old-client-name.example.com@EXAMPLE.COM** のように service_name/host_name@REALM の形式を取る Kerberos プリンシパルがあります。
- マシンが所属するすべてのホストグループを特定します。

  ```
  # ipa hostgroup-find old-client-name.example.com
  ```

42.10.2. Identity Management クライアントのアンインストール

クライアントをアンインストールすると、クライアントは Identity Management ドメインから削除され、SSSD (System Security Services Daemon) などのシステムサービスの Identity Management 設定もすべて削除されます。これにより、クライアントシステムの以前の設定が復元します。

手順

1. **ipa-client-install --uninstall** コマンドを実行します。

```
[root@client]# ipa-client-install --uninstall
```

2. クライアントホストの DNS エントリーを、手動でサーバーから削除します。

```
[root@server]# ipa dnsrecord-del
Record name: old-client-client
Zone name: idm.example.com
No option to delete specific record provided.
Delete all? Yes/No (default No): yes
----------------------
Deleted record "old-client-name"
```

3. **/etc/krb5.keytab** 以外のキータブについては、古いプリンシパルを削除します。

```
[root@client ~]# ipa-rmkeytab -k /path/to/keytab -r EXAMPLE.COM
```

4. IdM サーバーで、ホストエントリーを削除します。これにより、すべてのサービスが削除され、そのホストに発行されたすべての証明書が無効になります。

```
[root@server ~]# ipa host-del client.example.com
```

42.10.3. ホストシステムの名前変更

必要に応じてマシンの名前を変更します。以下に例を示します。

```
[root@client]# hostnamectl set-hostname new-client-name.example.com
```

これで、新しいホスト名で、Identity Management クライアントを Identity Management ドメインに再インストールできるようになります。

42.10.4. サービスの再追加、証明書の再生成、およびホストグループの再追加

手順

1. Identity Management (IdM) サーバーで、名前を変更するための IdM クライアントの準備 に定義された各サービスに新しいキータブを追加します。

```
[root@server ~]# ipa service-add service_name/new-client-name
```

2. 名前を変更するための IdM クライアントの準備 で割り当てた証明書のあるサービスに対して証明書を生成します。これには、以下を行います。

- IdM 管理ツールの使用
- **certmonger** ユーティリティーの使用

3. 名前を変更するための IdM クライアントの準備 で特定されたホストグループにクライアントを再追加します。

42.11. ホストエントリーの無効化と再有効化

このセクションでは、Identity Management (IdM) でホストを無効にして再度有効にする方法を説明します。

42.11.1. ホストの無効化

IdM でホストエントリーを無効にするには、この手順を完了します。

ドメインサービス、ホスト、およびユーザーはアクティブなホストにアクセスできます。メンテナンス上の理由などで、アクティブなホストを一時的に削除することが必要になる場合があります。このような状況でホストを削除すると、ホストエントリーと、関連するすべての設定が完全に削除されるため、望ましくありません。代わりに、ホストを無効にするオプションを選択してください。

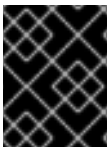
ホストを無効にすると、ドメインからドメインユーザーを完全に削除せずに、そのドメインにアクセスできなくなります。

手順

- **host-disable** コマンドを使用してホストを無効にします。ホストを無効にすると、ホストで現在アクティブなキータブが強制終了します。以下に例を示します。

```
$ kinit admin
$ ipa host-disable client.example.com
```

ホストを無効にすると、ホストはすべての IdM ユーザー、ホスト、およびサービスが利用できなくなりました。



重要

ホストエントリーを無効にすると、そのホストが無効になるだけではありません。そのホストで設定されているすべてのサービスも無効にします。

42.11.2. ホストの再有効化

無効な IdM ホストを再度有効にするには、次の手順に従います。

ホストを無効にすると、アクティブなキータブが強制的に終了し、設定エントリーを変更せずにホストが IdM ドメインから削除されます。

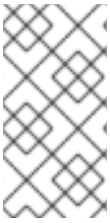
手順

- ホストを再度有効にするには、以下を追加して、**ipa-getkeytab** コマンドを使用します。
  - キータブを要求する IdM サーバーを指定する **-s** オプション
  - プリンシパル名を指定する **-p** オプション

- キータブを保存するファイルを指定する **-k** オプション

たとえば、**client.example.com** の **server.example.com** から新規ホストキータブを要求し、キータブを **/etc/krb5.keytab** ファイルに保存するには、次のコマンドを実行します。

```
$ ipa-getkeytab -s server.example.com -p host/client.example.com -k /etc/krb5.keytab -D
"cn=directory manager" -w password
```



注記

管理者の認証情報を使用して、**-D "uid=admin,cn=users,cn=accounts,dc=example,dc=com"** を指定することもできます。認証情報は、ホストのキータブの作成を許可されたユーザーに対応することが重要です。

ipa-getkeytab コマンドがアクティブな IdM クライアントまたはサーバーで実行する場合は、ユーザーが、**kinit admin** などを使用して TGT を取得した場合に、LDAP 認証情報 (**-D** および **-w**) を使用せずに実行できます。無効化されたホストでコマンドを直接実行するには、LDAP 認証情報を提供して IdM サーバーに認証します。

第43章 IDM WEB UI でホストエントリーの追加

この章では、Identity Management (IdM) のホストと、IdM Web UI でホストエントリーを追加する操作を説明します。

43.1. IDM のホスト

Identity Management (IdM) は、以下の ID を管理します。

- ユーザー
- サービス
- ホスト

ホストはマシンを表します。ホストには、IdM LDAP に IdM ID となるエントリーがあります。これは IdM サーバーの 389 Directory Server のインスタンスです。

IdM LDAP のホストエントリーは、その他のホストとドメイン内のサービスとの関係を確立するために使用されます。この関係では、ドメイン内ホストの認可および制御の **委譲** が不可欠な要素です。ホストは、**ホストベースのアクセス制御** (HBAC) ルールで使用できます。

IdM ドメインは、共通の ID 情報、共通ポリシー、および共有サービスを使用して、マシン間で共通性を確立します。ドメインのクライアントとしてのドメイン機能に属するマシンです。これは、ドメインが提供するサービスを使用することを意味します。IdM ドメインは、マシン専用の 3 つの主なサービスを提供します。

- DNS
- Kerberos
- 証明書の管理

IdM のホストは、そのホストで実行しているサービスと密接に接続されています。

- サービスエントリーは、ホストに関連付けられています。
- ホストには、ホストとサービスの両方の Kerberos プリンシパルが格納されます。

43.2. ホスト登録

本セクションでは、ホストを IdM クライアントとして登録し、登録中および登録後に何が起こるかを説明します。本セクションでは、IdM ホストの登録と、IdM ユーザーの登録を比較します。また、本セクションでは、ホストで利用可能な代替タイプの認証も概説します。

ホストの登録は、以下の要素で構成されます。

- IdM LDAP でのホストエントリーの作成: 場合によっては、IdM CLI で **ipa host-add** コマンドを使用するか、同等の IdM Web UI 操作 を使用します。
- ホストで IdM サービス (System Security Services Daemon (SSSD)、Kerberos、certmonger など) を設定し、ホストを IdM ドメインに参加させる。

2 つのアクションは、個別に実行することも一緒に実行することもできます。

個別に実行すると、異なるレベルの特権を持つ 2 人のユーザー間で 2 つのタスクを分割できます。これは、一括デプロイメントに役立ちます。

ipa-client-install コマンドは、2 つのアクションを一緒に実行できます。コマンドは、そのエントリーがまだ存在していない場合は IdM LDAP にホストエントリーを作成し、そのホストに Kerberos サービスと SSSD サービスの両方を設定します。このコマンドにより、ホストが IdM ドメイン内に移動し、接続先の IdM サーバーを識別できるようになります。IdM が管理する DNS ゾーンにホストが属する場合は、**ipa-client-install** でホストに DNS レコードも追加します。コマンドはクライアントで実行する必要があります。

43.3. ホストの登録に必要なユーザー権限

ホスト登録操作では、権限のないユーザーが不要なマシンを IdM ドメインに追加しないように、認証が必要になります。必要な特権は、次のようないくつかの要因によって異なります。

- ホストエントリーが **ipa-client-install** の実行とは別に作成される場合
- ワンタイムパスワード (OTP) が登録に使用される場合

必要に応じて IdM LDAP にホストエントリーを手動で作成するためのユーザー特権

CLI コマンド **ipa host-add** または IdM Web UI を使用して、IdM LDAP にホストエントリーを作成するのに必要なユーザー特権は **ホストの管理者** です。**ホスト管理者** の特権は、**IT スペシャリスト** ロールから取得できます。

クライアントを IdM ドメインに参加させるためのユーザー特権

ホストは、**ipa-client-install** コマンドの実行時に IdM クライアントとして設定されます。**ipa-client-install** コマンドの実行に必要な認証情報のレベルは、以下のような登録シナリオのどれに該当するかによって異なります。

- IdM LDAP のホストエントリーが存在しません。このシナリオでは、管理者の完全な認証情報または **ホスト管理者** ロールが必要です。完全な管理者とは **admins** グループのメンバーです。**ホスト管理者** ロールは、ホストの追加およびホストの登録の特権を提供します。このシナリオの詳細は ユーザー認証情報を使用したクライアントのインストール: 対話的なインストール を参照してください。
- IdM LDAP のホストエントリーが存在します。このシナリオでは、**ipa-client-install** を正常に実行するには、制限された管理者の認証情報が必要です。この場合、制限されている管理者には、**ホストの登録** 特権を提供する **登録管理者** ロールがあります。詳細は ユーザー認証情報を使用したクライアントのインストール: 対話的なインストール を参照してください。
- IdM LDAP にホストエントリーが存在し、完全または限定された管理者により、ホストの OTP が生成されました。このシナリオでは、正しい OTP を指定して **--password** オプションを指定して **ipa-client-install** コマンドを実行すると、通常のユーザーとして IdM クライアントをインストールできます。詳細は ワンタイムパスワードでクライアントのインストール: 対話的なインストール を参照してください。

登録後、IdM ホストは、IdM リソースにアクセスできるように、新しいセッションをすべて認証します。IdM サーバーがマシンを信頼し、そのマシンにインストールされているクライアントソフトウェアからの IdM 接続を受け入れるには、マシン認証が必要です。クライアントを認証すると、IdM サーバーはそのリクエストに応答できます。

43.4. IDM ホストとユーザーの登録と認証: 比較

IdM のユーザーとホストの間には多くの類似点があります。この類似点には、登録ステージで見られるものと、デプロイメントステージでの認証に関係するものがあります。

- 登録段階 (ユーザーおよびホストの登録):
  - 管理者は、ユーザーまたはホストが実際に IdM に参加する前に、ユーザーとホストの LDAP エントリーを作成できます。コマンドは、ステージユーザーの場合は **ipa stageuser-add** で、ホストの場合は **ipa host-add** です。
  - ホストで **ipa-client-install** コマンドを実行すると、**キーテーブル** (または略してキータブ)、(ある程度ユーザーパスワードに類似する) 対称キーを含むファイルが作成され、ホストが IdM レルムに参加します。同様に、ユーザーはアカウントをアクティブ化するときにパスワードを作成するように求められ、IdM レルムに参加します。
  - ユーザーパスワードは、ユーザーのデフォルトの認証方法ですが、キータブはホストのデフォルトの認証方法です。キータブは、ホストのファイルに保存されます。

表43.1 ユーザーおよびホストの登録

アクション	ユーザー	ホスト
登録前	$ ipa stageuser-add user_name [--password]	$ ipa host-add host_name [--random]
アカウントのアクティブ化	$ ipa stageuser-activate user_name	$ ipa-client install [--password] (ホスト自体で実行する必要があります)

- デプロイメント段階 (ユーザーおよびホストセッションの認証):
  - ユーザーが新しいセッションを開始すると、ユーザーはパスワードを使用して認証を行います。同様に、切り替え時に、ホストがそのキータブファイルを提示して認証を行います。SSSD (System Security Services Daemon) は、このプロセスをバックグラウンドで管理します。
  - 認証が成功すると、ユーザーまたはホストは、Kerberos チケット発行許諾チケット (TGT) を取得します。
  - 次に、TGT を使用して、特定のサービスの特定のチケットを取得します。

表43.2 ユーザーおよびホストセッションの認証

	ユーザー	ホスト
認証のデフォルト手段	**パスワード**	**キータブ**
セッションの開始 (通常のユーザー)	$ kinit user_name	**[ホストへの切り替え]**
認証成功の結果	**TGT** は、特定サービスへのアクセスの取得に使用されます。	**TGT** は、特定サービスへのアクセスの取得に使用されます。

TGT およびその他の Kerberos チケットは、サーバーにより定義された Kerberos サービスおよびポリシーの一部として生成されます。Kerberos チケットの最初の付与、Kerberos 認証情報の更新、および Kerberos セッションの破棄もすべて IdM サービスにより自動的に処理されます。

IdM ホストの代替認証オプション

キータブとは別に、IdM は、その他の 2 つのタイプのマシン認証にも対応しています。

- SSH 鍵。ホストの SSH 公開キーが作成され、ホストエントリーにアップロードされます。そこから、SSSD (System Security Services Daemon) は IdM を ID プロバイダーとして使用し、OpenSSH およびその他のサービスと一緒に機能して、IdM の中央にある公開鍵を参照できます。
- 機械の証明書。この場合、マシンは IdM サーバーの認証局により発行され、IdM の Directory Server に保存されている SSL 証明書を使用します。次に、証明書はマシンに送信され、サーバーに対する認証時に提示されます。クライアントでは、証明書は certmonger というサービスにより管理されます。

43.5. IDM LDAP のホストエントリー

Identity Management (IdM) ホストエントリーには、ホストに関する情報とホストに含めることができる属性が含まれています。

LDAP ホストエントリーは、IdM 内のクライアントに関するすべての関連情報が含まれます。

- ホストに関連付けられたサービスエントリー
- ホストおよびサービスプリンシパル
- アクセス制御ルール
- 物理的な場所やオペレーティングシステムなどのマシン情報

注記

IdM Web UI の **Identity → Hosts** タブには、IdM LDAP に保存されている特定のホストに関する情報がすべて表示されないことに注意してください。

ホストエントリー設定プロパティー

ホストエントリーには、ホストに関する情報 (物理的な場所、MAC アドレス、鍵、証明書など、システム設定を除く) を含めることができます。

この情報は、ホストエントリーが手動で作成された場合に、作成時に設定できます。また、この情報のほとんどは、ホストがそのドメインに登録してからホストエントリーに追加できます。

表43.3 ホスト設定プロパティー

UI フィールド	コマンドラインオプション	説明
説明	**--desc**=description	ホストの説明。
局所性	**--locality**=locality	ホストの地理的な場所。

UI フィールド	コマンドラインオプション	説明
場所	**--location**=location	データセンターのラックなど、ホストの物理的な場所。
プラットフォーム	**--platform**=string	ホストのハードウェアまたはアーキテクチャー。
オペレーティングシステム	**--os**=string	ホストのオペレーティングシステムとバージョン。
MAC アドレス	**--macaddress**=address	ホストの MAC アドレス。これは多値属性です。MAC アドレスは、NIS プラグインにより、ホスト用の NIS の **ethers** マップを作成するために使用されます。
SSH 公開鍵	**--sshpubkey**=string	ホストの完全 SSH 公開鍵。これは複数値の属性であるため、複数の鍵を設定できます。
プリンシパル名 (編集不可)	**--principalname**=principal	ホストの Kerberos プリンシパル名。**-p** に別のプリンシパルを明示的に設定しない限り、クライアントのインストール時にホスト名がデフォルトになります。これはコマンドラインツールを使用して変更できますが、UI で変更することはできません。
ワンタイムパスワードの設定	**--password**=string	このオプションは、一括登録で使用できるホストのパスワードを設定します。
-	**--random**	このオプションは、一括登録で使用されるランダムパスワードを生成します。
-	**--certificate**=string	ホストの証明書ブロブ。
-	**--updatedns**	これにより、IP アドレスが変更した場合に、ホストが DNS エントリーを動的に更新できるかどうかが設定されます。

43.6. WEB UI でのホストエントリーの追加

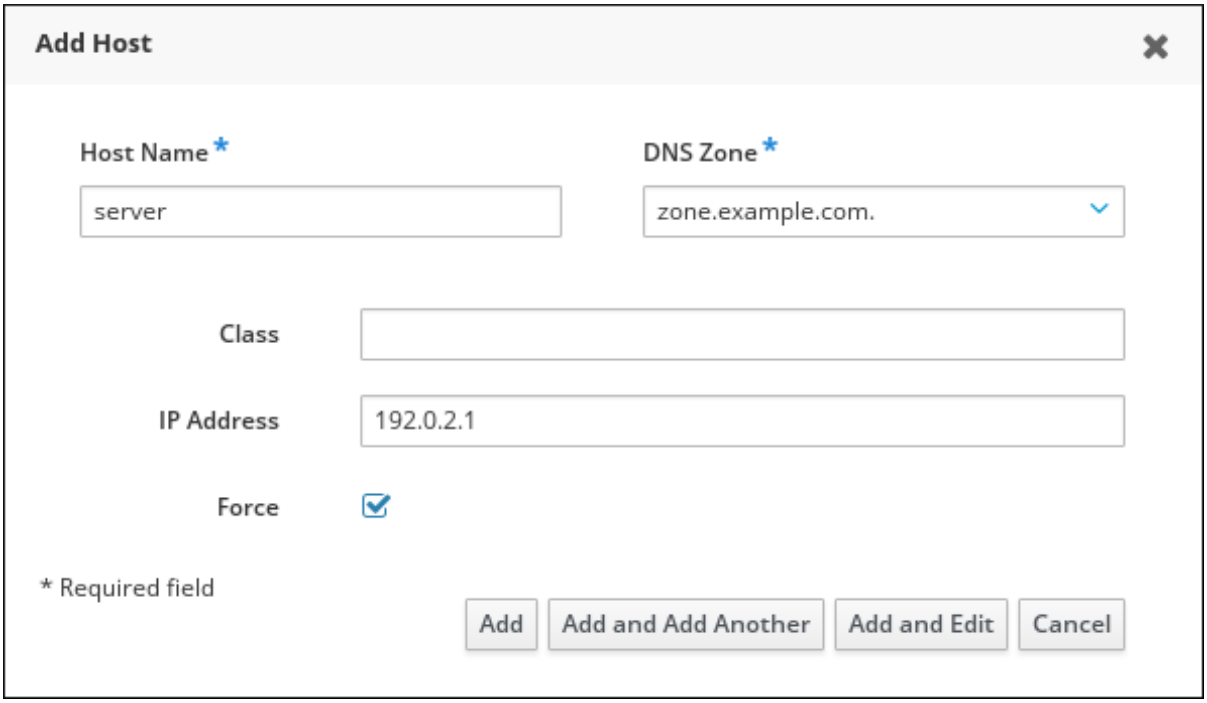
1. **Identity** タブを開き、サブタブの **ホスト** を選択します。

2. ホストリストの上部にある **追加** をクリックします。

図43.1 ホストエントリーの追加

3. マシンの名前を入力し、ドロップダウンリストで、設定済みのゾーンの中からドメインを選択します。ホストに静的 IP アドレスが割り当てられている場合は、ホストエントリーにそのアドレスを追加して、DNS エントリーが完全に作成されるようにします。
Class フィールドには、現時点では特定の目的はありません。

図43.2 ホストウィザードの追加



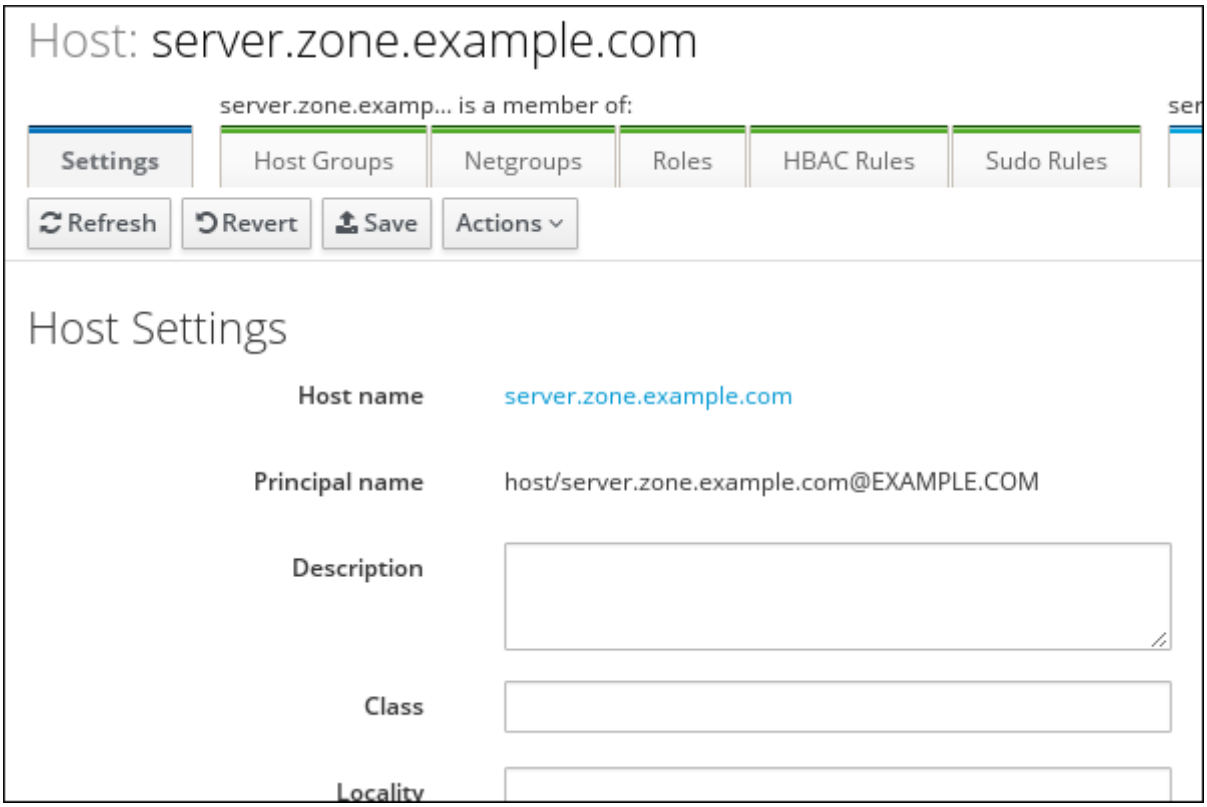
DNS ゾーンは IdM で作成できます。IdM サーバーが DNS サーバーを管理しない場合は、通常のテキストフィールドなど、メニューエリアでゾーンを手動で入力できます。

注記

ホストが DNS 経由で解決できるかどうかの確認を行わないようにするには、**強制** チェックボックスを選択します。

4. **追加および編集** ボタンをクリックして、拡張されたエントリーページに直接選択し、その他の属性情報を入力します。ホストのハードウェアと物理的な場所に関する情報は、ホストエントリーに追加できます。

図43.3 拡張されたエントリーページ



第44章 ANSIBLE PLAYBOOK を使用したホストの管理

Ansible は、システムの設定、ソフトウェアのデプロイ、ローリング更新の実行に使用する自動化ツールです。Ansible には Identity Management (IdM) のサポートが含まれ、Ansible モジュールを使用してホスト管理を自動化できます。

Ansible Playbook を使用してホストおよびホストエントリーを管理する際に、以下のコンセプトに基づき、操作が実行されます。

- **FQDN** でのみ定義されている IdM ホストエントリーを存在させる手順
- IP アドレスを使用して IdM ホストエントリーを存在させる手順
- 無作為のパスワードが指定された IdM ホストエントリーを複数存在させる手順
- 複数の IP アドレスが指定された IdM ホストエントリーを存在させる手順
- IdM ホストエントリーがないことを確認する手順

44.1. ANSIBLE PLAYBOOK を使用して FQDN が指定された IDM ホストエントリーを存在させる手順

以下の手順に従って、Ansible Playbook を使用して、Identity Management (IdM) にホストエントリーが存在することを確認します。ホストエントリーは、**完全修飾ドメイン名** (FQDN) によってのみ定義されます。

以下の条件のいずれかが当てはまる場合は、ホストの **FQDN** 名を指定するだけで十分です。

- IdM サーバーが DNS を管理するよう設定されていない。
- ホストに静的 IP アドレスがないか、ホストの設定時に IP アドレスが不明である。**FQDN** だけで定義されたホストを追加すると、基本的に IdM DNS サービスにプレースホルダーエントリーが作成されます。たとえば、ラップトップは IdM クライアントとして事前設定されている場合がありますが、設定時には IP アドレスがありません。DNS サービスがレコードを動的に更新すると、ホストの現在の IP アドレスが検出され、DNS レコードが更新されます。

注記

Ansible ない場合に、**ipa host-add** コマンドを使用すると、ホストエントリーが IdM に作成されます。ホストを IdM に追加すると、IdM でのホストの状態が present になります。Ansible は冪等性に依存しているので、Ansible を使用して IdM にホストを追加するには、ホストの状態を Present (**state: present**) として定義した Playbook を作成する必要があります。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。

- この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
- この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。

- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **inventory.file** などのインベントリーファイルを作成して、そのファイルに **ipaserver** を定義します。

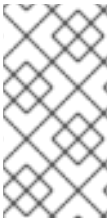
```
[ipaserver]
server.idm.example.com
```

2. IdM に存在させるホストの **FQDN** を使用して Ansible Playbook ファイルを作成します。この手順は、**/usr/share/doc/ansible-freeipa/playbooks/host/add-host.yml** ファイルのサンプルをコピーして変更し、簡素化できます。

```
---
- name: Host present
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Host host01.idm.example.com present
    ipahost:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: host01.idm.example.com
      state: present
      force: yes
```

3. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file path_to_playbooks_directory/ensure-host-
is-present.yml
```



注記

以下の手順では、IdM LDAP サーバーにホストエントリーが作成されますが、ホストは IdM Kerberos レルムには登録されません。登録されるようにするには、ホストを IdM クライアントとしてデプロイする必要があります。詳細は、Ansible Playbook を使用した Identity Management クライアントのインストール を参照してください。

検証手順

1. admin として IdM サーバーにログインします。

```
$ ssh admin@server.idm.example.com
Password:
```

2. **ipa host-show** コマンドを入力し、ホストの名前を指定します。

```
$ ipa host-show host01.idm.example.com
  Host name: host01.idm.example.com
  Principal name: host/host01.idm.example.com@IDM.EXAMPLE.COM
  Principal alias: host/host01.idm.example.com@IDM.EXAMPLE.COM
  Password: False
  Keytab: False
  Managed by: host01.idm.example.com
```

この出力で、**host01.idm.example.com** が IdM に存在することを確認します。

44.2. ANSIBLE PLAYBOOK を使用して DNS 情報など IDM ホストエントリーを存在させる手順

以下の手順に従って、Ansible Playbook を使用して、Identity Management (IdM) にホストエントリーが存在することを確認します。ホストエントリーは、ホストの **完全修飾ドメイン名** (FQDN) および IP アドレスで定義されます。

注記

Ansible ない場合に、**ipa host-add** コマンドを使用すると、ホストエントリーが IdM に作成されます。ホストを IdM に追加すると、IdM でのホストの状態が present になります。Ansible は冪等性に依存しているので、Ansible を使用して IdM にホストを追加するには、ホストの状態を Present (**state: present**) として定義した Playbook を作成する必要があります。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **inventory.file** などのインベントリーファイルを作成して、そのファイルに **ipaserver** を定義します。

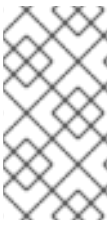
```
[ipaserver]
server.idm.example.com
```

2. IdM に存在させるホストの **完全修飾ドメイン名** (FQDN) で Ansible Playbook ファイルを作成します。また、IdM サーバーが DNS を管理するように設定され、ホストの IP アドレスが分かっている場合は、**ip_address** パラメーターの値を指定します。ホストを DNS リソースレコードに存在させるには、IP アドレスが必要です。この手順は、**/usr/share/doc/ansible-freeipa/playbooks/host/host-present.yml** ファイルのサンプルをコピーして変更し、簡素化できます。また、その他の追加情報を含めることもできます。

```
---
- name: Host present
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure host01.idm.example.com is present
    ipahost:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: host01.idm.example.com
      description: Example host
      ip_address: 192.168.0.123
      locality: Lab
      ns_host_location: Lab
      ns_os_version: CentOS 7
      ns_hardware_platform: Lenovo T61
      mac_address:
      - "08:00:27:E3:B1:2D"
      - "52:54:00:BD:97:1E"
      state: present
```

3. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file path_to_playbooks_directory/ensure-host-
is-present.yml
```



注記

以下の手順では、IdM LDAP サーバーにホストエントリーが作成されますが、ホストは IdM Kerberos レルムには登録されません。登録されるようにするには、ホストを IdM クライアントとしてデプロイする必要があります。詳細は、Ansible Playbook を使用した Identity Management クライアントのインストール を参照してください。

検証手順

1. admin として IdM サーバーにログインします。

```
$ ssh admin@server.idm.example.com
Password:
```

2. **ipa host-show** コマンドを入力し、ホストの名前を指定します。

```
$ ipa host-show host01.idm.example.com
  Host name: host01.idm.example.com
  Description: Example host
```

```
Locality: Lab
Location: Lab
Platform: Lenovo T61
Operating system: CentOS 7
Principal name: host/host01.idm.example.com@IDM.EXAMPLE.COM
Principal alias: host/host01.idm.example.com@IDM.EXAMPLE.COM
MAC address: 08:00:27:E3:B1:2D, 52:54:00:BD:97:1E
Password: False
Keytab: False
Managed by: host01.idm.example.com
```

この出力で、**host01.idm.example.com** が IdM に存在することを確認します。

44.3. ANSIBLE PLAYBOOK を使用して無作為のパスワードが指定された IDM ホストエントリーを複数存在させる手順

ipahost モジュールでは、システム管理者は、Ansible タスク 1 つだけを使用して、IdM に複数のホストエントリーが存在するか、存在しないかを確認できます。以下の手順に従って、**fully-qualified domain names** (FQDN) でのみ定義されるホストエントリーを複数存在することを確認します。Ansible Playbook を実行すると、ホストのパスワードが無作為に生成されます。

注記

Ansible ない場合に、**ipa host-add** コマンドを使用すると、ホストエントリーが IdM に作成されます。ホストを IdM に追加すると、IdM でのホストの状態が present になります。Ansible は冪等性に依存しているので、Ansible を使用して IdM にホストを追加するには、ホストの状態を Present (**state: present**) として定義した Playbook を作成する必要があります。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **inventory.file** などのインベントリーファイルを作成して、そのファイルに **ipaserver** を定義します。

```
[ipaserver]
server.idm.example.com
```

2. IdM に存在させるホストの **完全修飾ドメイン名** (FQDN) で Ansible Playbook ファイルを作成します。IdM にホストが既存し、**update_password** が **on_create** に制限されている場合にも、Ansible Playbook を使用して各ホストに対して無作為にパスワードを作成するには、**random: yes** と **force: yes** のオプションを追加します。この手順を簡素化するには、**/usr/share/doc/ansible-freeipa/README-host.md** Markdown ファイルからサンプルをコピーして変更できます。

```
---
- name: Ensure hosts with random password
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Hosts host01.idm.example.com and host02.idm.example.com present with random
passwords
    ipahost:
      ipaadmin_password: "{{ ipaadmin_password }}"
      hosts:
      - name: host01.idm.example.com
        random: yes
        force: yes
      - name: host02.idm.example.com
        random: yes
        force: yes
    register: ipahost
```

3. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file path_to_playbooks_directory/ensure-hosts-
are-present.yml
[...]
TASK [Hosts host01.idm.example.com and host02.idm.example.com present with random
passwords]
changed: [r8server.idm.example.com] => {"changed": true, "host":
{"host01.idm.example.com": {"randompassword": "0HoIRvjUdH0Ycbf6uYdWTxH"},
"host02.idm.example.com": {"randompassword": "5VdLgrf3wvojmACdHC3uA3s"}}}
```

注記

無作為に作成されたワンタイムパスワード (OTP) を使用して IdM クライアントとしてホストをデプロイするには、Ansible Playbook を使用した IdM クライアント登録の認証オプション または ワンタイムパスワードを使用したクライアントのインストール: 対話型インストール を参照してください。

検証手順

1. admin として IdM サーバーにログインします。

```
$ ssh admin@server.idm.example.com
Password:
```

2. **ipa host-show** コマンドを入力し、ホストのいずれかの名前を指定します。

```
$ ipa host-show host01.idm.example.com
  Host name: host01.idm.example.com
  Password: True
  Keytab: False
  Managed by: host01.idm.example.com
```

この出力で、**host01.idm.example.com** が無作為に作成されたパスワードが指定された IdM に存在することを確認します。

44.4. ANSIBLE PLAYBOOK を使用して複数の IP アドレスが指定された IDM ホストエントリーを存在させる手順

以下の手順に従って、Ansible Playbook を使用して Identity Management (IdM) にホストエントリーが存在することを確認します。ホストエントリーは、**完全修飾ドメイン名** (FQDN) と複数の IP アドレスで定義されます。

注記

Ansible **ipahost** モジュールでは、**ipa host** ユーティリティーとは対照的に、ホストの IPv4 および IPv6 アドレスが複数存在させたり、または存在させなかったりできます。**ipa host-mod** コマンドは IP アドレスを処理できません。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **inventory.file** などのインベントリーファイルを作成して、そのファイルに **ipaserver** を定義します。

```
[ipaserver]
server.idm.example.com
```

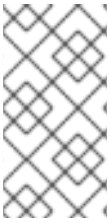
2. Ansible Playbook ファイルを作成します。**ipahost** 変数の **名前** として、IdM に存在させるホストの **完全修飾ドメイン名** (FQDN) を指定します。**ip_address** 構文を使用して、複数の IPv4 および IPv6 **ip_address** 値をそれぞれ別の行に指定します。この手順

は、**/usr/share/doc/ansible-freeipa/playbooks/host/host-member-ipaddresses-present.yml** ファイルのサンプルをコピーして変更し、簡素化できます。追加情報を含めることもできます。

```
---
- name: Host member IP addresses present
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure host101.example.com IP addresses present
    ipahost:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: host01.idm.example.com
      ip_address:
      - 192.168.0.123
      - fe80::20c:29ff:fe02:a1b3
      - 192.168.0.124
      - fe80::20c:29ff:fe02:a1b4
      force: yes
```

3. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file path_to_playbooks_directory/ensure-host-
with-multiple-IP-addreses-is-present.yml
```



注記

この手順では、IdM LDAP サーバーにホストエントリーは作成されますが、ホストは IdM Kerberos レルムに登録されません。登録されるようにするには、ホストを IdM クライアントとしてデプロイする必要があります。詳細は、Ansible Playbook を使用した Identity Management クライアントのインストール を参照してください。

検証手順

1. admin として IdM サーバーにログインします。

```
$ ssh admin@server.idm.example.com
Password:
```

2. **ipa host-show** コマンドを入力し、ホストの名前を指定します。

```
$ ipa host-show host01.idm.example.com
  Principal name: host/host01.idm.example.com@IDM.EXAMPLE.COM
  Principal alias: host/host01.idm.example.com@IDM.EXAMPLE.COM
  Password: False
  Keytab: False
  Managed by: host01.idm.example.com
```

この出力で、**host01.idm.example.com** が IdM に存在することを確認します。

3. IdM DNS レコードにホストの複数の IP アドレスが存在することを確認するには、**ipa dnsrecord-show** コマンドを入力し、以下の情報を指定します。

   - IdM ドメインの名前
   - ホストの名前

   ```
   $ ipa dnsrecord-show idm.example.com host01
   [...]
     Record name: host01
     A record: 192.168.0.123, 192.168.0.124
     AAAA record: fe80::20c:29ff:fe02:a1b3, fe80::20c:29ff:fe02:a1b4
   ```

   この出力では、Playbook で指定された IPv4 アドレスおよび IPv6 アドレスがすべて **host01.idm.example.com** ホストエントリーに正しく関連付けられていることを確認します。

44.5. ANSIBLE PLAYBOOK を使用して IDM ホストエントリーがないことを確認する手順

以下の手順に従って、Ansible Playbook を使用して Identity Management (IdM) にホストエントリーがないことを確認します。

前提条件

- IdM 管理者の認証情報

手順

1. **inventory.file** などのインベントリーファイルを作成して、そのファイルに **ipaserver** を定義します。

   ```
   [ipaserver]
   server.idm.example.com
   ```

2. IdM に存在させないホストの **完全修飾ドメイン名** (FQDN) を指定して Ansible Playbook ファイルを作成します。IdM ドメインに DNS が統合されている場合は、**updatedns: yes** オプションを使用して、あらゆる種類のホストに関連するレコードを DNS から削除します。
   この手順は、**/usr/share/doc/ansible-freeipa/playbooks/host/delete-host.yml** ファイルのサンプルをコピーして変更し、簡素化できます。

   ```
   ---
   - name: Host absent
     hosts: ipaserver

     vars_files:
     - /home/user_name/MyPlaybooks/secret.yml
     tasks:
     - name: Host host01.idm.example.com absent
       ipahost:
         ipaadmin_password: "{{ ipaadmin_password }}"
         name: host01.idm.example.com
         updatedns: yes
         state: absent
   ```

3. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file path_to_playbooks_directory/ensure-host-
absent.yml
```

注記

この手順の結果は以下のようになります。

- IdM Kerberos レルムにホストが存在していない。
- IdM LDAP サーバーにホストエントリーが存在しない。

SSSD (System Security Services Daemon) などのシステムサービスの特定の IdM 設定をクライアントホスト自体から削除するには、クライアントで **ipa-client-install --uninstall** コマンドを実行する必要があります。詳細は、IdM クライアントのアンインストール を参照してください。

検証手順

1. admin として **ipaserver** にログインします。

```
$ ssh admin@server.idm.example.com
Password:
[admin@server /]$
```

2. **host01.idm.example.com** に関する情報を表示します。

```
$ ipa host-show host01.idm.example.com
ipa: ERROR: host01.idm.example.com: host not found
```

この出力では、ホストが IdM に存在しないことを確認します。

44.6. 関連情報

- **/usr/share/doc/ansible-freeipa/README-host.md** Markdown ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/host** ディレクトリーにある追加の Playbook を表示します。

第45章 IDM CLI を使用したホストグループの管理

次の操作を使用して、コマンドラインインターフェイス (CLI) でホストグループとそのメンバーを管理する方法について詳しく説明します。

- ホストグループおよびそのメンバーの表示
- ホストグループの作成
- ホストグループの削除
- ホストグループメンバーの追加
- ホストグループメンバーの削除
- ホストグループメンバーマネージャーの追加
- ホストグループメンバーマネージャーの削除

45.1. IDM のホストグループ

IdM ホストグループを使用すると、重要な管理タスク (特にアクセス制御) を一元管理できます。

ホストグループの定義

ホストグループは、一般的なアクセス制御ルールやその他の特性を持つ IdM ホストセットが含まれるエンティティーです。たとえば、企業の部門、物理的な場所、またはアクセス制御要件に基づいてホストグループを定義できます。

IdM のホストグループには以下が含まれます。

- IdM サーバーおよびクライアント
- その他の IdM ホストグループ

デフォルトで作成されたホストグループ

デフォルトでは、IdM サーバーは、全 IdM サーバーホストのホストグループ **ipaservers** を作成します。

直接および間接のグループメンバー

IdM のグループ属性は、直接メンバーと間接メンバーの両方に適用されます。ホストグループ B がホストグループ A のメンバーである場合、ホストグループ B のすべてのユーザーはホストグループ A の間接メンバーと見なされます。

45.2. CLI での IDM ホストグループの表示

コマンドラインインターフェイス (CLI) を使用して IdM ホストグループを表示するには、次の手順に従います。

前提条件

- IdM、またはユーザー管理者ロールを管理する管理者権限
- 有効な Kerberos チケット。詳細は、Using kinit to log in to IdM manually を参照してください。

手順

1. **ipa hostgroup-find** コマンドを使用して、すべてのホストグループを検索します。

```
$ ipa hostgroup-find
------------------
1 hostgroup matched
------------------
  Host-group: ipaservers
  Description: IPA server hosts
----------------------------
Number of entries returned 1
----------------------------
```

ホストグループのすべての属性を表示するには、**--all** オプションを追加します。以下に例を示します。

```
$ ipa hostgroup-find --all
------------------
1 hostgroup matched
------------------
  dn: cn=ipaservers,cn=hostgroups,cn=accounts,dc=idm,dc=local
  Host-group: ipaservers
  Description: IPA server hosts
  Member hosts: xxx.xxx.xxx.xxx
  ipauniqueid: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
  objectclass: top, groupOfNames, nestedGroup, ipaobject, ipahostgroup
----------------------------
Number of entries returned 1
----------------------------
```

45.3. CLI を使用した IDM ホストグループの作成

コマンドラインインターフェイス (CLI) を使用して IdM ホストグループを作成するには、次の手順に従います。

前提条件

- IdM、またはユーザー管理者ロールを管理する管理者権限
- 有効な Kerberos チケット。詳細は、Using kinit to log in to IdM manually を参照してください。

手順

1. **ipa hostgroup-add** コマンドを使用してホストグループを追加します。
たとえば、**group_name** という名前の IdM ホストグループを作成して説明を追加するには、次のコマンドを実行します。

```
$ ipa hostgroup-add --desc 'My new host group' group_name
---------------------
Added hostgroup "group_name"
---------------------
```

```
  Host-group: group_name
  Description: My new host group
----------------------
```

45.4. CLI での IDM ホストグループの削除

コマンドラインインターフェイス (CLI) を使用して IdM ホストグループを削除するには、次の手順に従います。

前提条件

- IdM、またはユーザー管理者ロールを管理する管理者権限
- 有効な Kerberos チケット。詳細は、Using kinit to log in to IdM manually を参照してください。

手順

1. **ipa hostgroup-del** コマンドを使用してホストグループを削除します。
たとえば、**group_name** という名前の IdM ホストグループを削除するには、次のコマンドを実行します。

```
$ ipa hostgroup-del group_name
--------------------------
Deleted hostgroup "group_name"
--------------------------
```

注記

グループを削除しても、IdM からグループメンバーは削除されません。

45.5. CLI での IDM ホストグループメンバーの追加

コマンド 1 つで、ホストとホストグループを IdM ホストグループのメンバーとして追加できます。

前提条件

- IdM、またはユーザー管理者ロールを管理する管理者権限
- 有効な Kerberos チケット。詳細は、Using kinit to log in to IdM manually を参照してください。
- **オプション。ipa hostgroup-find** コマンドを使用して、ホストおよびホストグループを検索します。

手順

1. ホストグループにメンバーを追加するには、**ipa hostgroup-add-member** を使用して、関連する情報を指定します。以下のオプションを使用して、追加するメンバーのタイプを指定できます。
    - **--hosts** オプションを使用して、ホストを IdM ホストグループに追加します。
    たとえば、**example_member** という名前のホストを **group_name** という名前のグループに追加するには、次のコマンドを実行します。

```
$ ipa hostgroup-add-member group_name --hosts example_member
Host-group: group_name
Description: My host group
Member hosts: example_member
-------------------------
Number of members added 1
-------------------------
```

- **--hostgroups** オプションを使用して、IdM ホストグループにホストグループを追加します。
  たとえば、**nested_group** という名前のホストグループを **group_name** という名前のグループに追加するには、次のコマンドを実行します。

```
$ ipa hostgroup-add-member group_name --hostgroups nested_group
Host-group: group_name
Description: My host group
Member host-groups: nested_group
-------------------------
Number of members added 1
-------------------------
```

- 以下の構文を使用すると、単一のコマンドで複数のホストと複数のホストグループを IdM ホストグループに追加できます。

```
$ ipa hostgroup-add-member group_name --hosts={host1,host2} --hostgroups=
{group1,group2}
```

重要

ホストグループを別のホストグループのメンバーとして追加する場合は、再帰グループを作成しないでください。たとえば、グループ A がグループ B のメンバーである場合は、グループ B をグループ A のメンバーとして追加しないでください。再帰的なグループにより予期しない動作が発生する可能性があります。

45.6. CLI での IDM ホストグループメンバーの削除

コマンド 1 つで IdM ホストグループからホストとホストグループを削除できます。

前提条件

- IdM、またはユーザー管理者ロールを管理する管理者権限
- 有効な Kerberos チケット。詳細は、Using kinit to log in to IdM manually を参照してください。
- **オプション。ipa hostgroup-find** コマンドを使用して、削除するメンバーがグループに含まれていることを確認します。

手順

1. ホストグループメンバーを削除するには、**ipa hostgroup-remove-member** コマンドを使用して、関連する情報を指定します。以下のオプションを使用して、削除するメンバーのタイプを指定できます。

- **--hosts** オプションを使用して、IdM ホストグループからホストを削除します。
  たとえば、**example_member** という名前のホストを **group_name** という名前のグループから削除するには、次のコマンドを実行します。

```
$ ipa hostgroup-remove-member group_name --hosts example_member
Host-group: group_name
Description: My host group
-------------------------
Number of members removed 1
-------------------------
```

- **--hostgroups** オプションを使用して、IdM ホストグループからホストグループを削除します。
  たとえば、**nested_group** という名前のグループから **group_name** という名前のホストグループを削除するには、次のコマンドを実行します。

```
$ ipa hostgroup-remove-member group_name --hostgroups example_member
Host-group: group_name
Description: My host group
-------------------------
Number of members removed 1
-------------------------
```

注記

グループを削除しても、IdM からグループメンバーは削除されません。

- 以下の構文を使用すると、単一のコマンドで IdM ホストグループから複数のホストとホストグループを削除できます。

```
$ ipa hostgroup-remove-member group_name --hosts={host1,host2} --hostgroups={group1,group2}
```

45.7. CLI を使用した IDM ホストグループメンバーマネージャーの追加

コマンド 1 つで、ホストとホストグループを IdM ホストグループのメンバーとして追加できます。メンバーマネージャーは、ホストまたはホストグループを IdM ホストグループに追加できますが、ホストグループの属性は変更できません。

前提条件

- IdM、またはユーザー管理者ロールを管理する管理者権限
- 有効な Kerberos チケット。詳細は、Using kinit to log in to IdM manually を参照してください。
- メンバーマネージャーとして追加するホストまたはホストグループの名前と、管理するホストグループ名が必要です。

手順

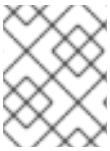
1. **オプション。ipa hostgroup-find** コマンドを使用して、ホストおよびホストグループを検索します。

2. ホストグループにメンバーマネージャーを追加するには、**ipa hostgroup-add-member-manager** を使用します。
たとえば、 **example_member** という名前のユーザーを、**group_name** という名前のグループにメンバーマネージャーとして追加するには、次のコマンドを実行します。

```
$ ipa hostgroup-add-member-manager group_name --user example_member
Host-group: group_name
Member hosts: server.idm.example.com
Member host-groups: project_admins
Member of netgroups: group_name
Membership managed by users: example_member
-------------------------
Number of members added 1
-------------------------
```

3. **--groups** オプションを使用して、1つ以上のホストグループをメンバーマネージャーとして IdM ホストグループに追加します。
たとえば、**admin_group** という名前のホストグループを、**group_name** という名前のグループにメンバーマネージャーとして追加するには、次のコマンドを実行します。

```
$ ipa hostgroup-add-member-manager group_name --groups admin_group
Host-group: group_name
Member hosts: server.idm.example.com
Member host-groups: project_admins
Member of netgroups: group_name
Membership managed by groups: admin_group
Membership managed by users: example_member
-------------------------
Number of members added 1
-------------------------
```



注記

ホストグループにメンバーマネージャーを追加してから、更新が Identity Management 環境のすべてのクライアントに広がるまでに時間がかかる場合があります。

検証手順

- **ipa group-show** コマンドを使用して、ホストユーザーおよびグループがメンバーマネージャーとして追加されたことを確認します。

```
$ ipa hostgroup-show group_name
Host-group: group_name
Member hosts: server.idm.example.com
Member host-groups: project_admins
Membership managed by groups: admin_group
Membership managed by users: example_member
```

関連情報

- 詳細は、**ipa hostgroup-add-member-manager --help** を参照してください。
- 詳細は、**ipa hostgroup-show --help** を参照してください。

45.8. CLI での IDM ホストグループメンバーマネージャーの削除

1 つのコマンドで IdM ホストグループのメンバーマネージャーからホストとホストグループを削除できます。メンバーマネージャーは、IdM ホストグループからホストグループのメンバーマネージャーを削除できますが、ホストグループの属性は変更できません。

前提条件

- IdM、またはユーザー管理者ロールを管理する管理者権限
- 有効な Kerberos チケット。詳細は、Using kinit to log in to IdM manually を参照してください。
- 削除する既存のメンバーマネージャーのホストグループ名と、管理するホストグループ名が必要です。

手順

1. **オプション。ipa hostgroup-find** コマンドを使用して、ホストおよびホストグループを検索します。
2. ホストグループからメンバーマネージャーを削除するには、**ipa hostgroup-remove-member-manager** コマンドを使用します。
   たとえば、 **example_member** という名前のユーザーを、**group_name** という名前のグループのメンバーマネージャーから削除するには、次のコマンドを実行します。

```
$ ipa hostgroup-remove-member-manager group_name --user example_member
Host-group: group_name
Member hosts: server.idm.example.com
Member host-groups: project_admins
Member of netgroups: group_name
Membership managed by groups: nested_group
---------------------------
Number of members removed 1
---------------------------
```

3. **--groups** オプションを使用して、IdM ホストグループのメンバーマネージャーからホストグループを 1 つまたは複数削除します。
   たとえば、**group_name** という名前のグループのメンバーマネージャーから **nested_group** という名前のホストグループを削除するには、次のコマンドを実行します。

```
$ ipa hostgroup-remove-member-manager group_name --groups nested_group
Host-group: group_name
Member hosts: server.idm.example.com
Member host-groups: project_admins
Member of netgroups: group_name
---------------------------
Number of members removed 1
---------------------------
```

注記

ホストグループからメンバーマネージャーを削除してから、更新が Identity Management 環境のすべてのクライアントに広がるまでに時間がかかる場合があります。

検証手順

- **ipa group-show** コマンドを使用して、メンバーマネージャーからホストユーザーとホストグループが削除されていることを確認します。

```
$ ipa hostgroup-show group_name
Host-group: group_name
Member hosts: server.idm.example.com
Member host-groups: project_admins
```

関連情報

- 詳細は、**ipa hostgroup-remove-member-manager --help** を参照してください。
- 詳細は、**ipa hostgroup-show --help** を参照してください。

第46章 IDM WEB UI を使用したホストグループの管理

次の操作を使用して、Web インターフェイス (Web UI) でホストグループとそのメンバーを管理する方法について詳しく説明します。

- ホストグループおよびそのメンバーの表示
- ホストグループの作成
- ホストグループの削除
- ホストグループメンバーの追加
- ホストグループメンバーの削除
- ホストグループメンバーマネージャーの追加
- ホストグループメンバーマネージャーの削除

46.1. IDM のホストグループ

IdM ホストグループを使用すると、重要な管理タスク (特にアクセス制御) を一元管理できます。

ホストグループの定義

ホストグループは、一般的なアクセス制御ルールやその他の特性を持つ IdM ホストセットが含まれるエンティティーです。たとえば、企業の部門、物理的な場所、またはアクセス制御要件に基づいてホストグループを定義できます。

IdM のホストグループには以下が含まれます。

- IdM サーバーおよびクライアント
- その他の IdM ホストグループ

デフォルトで作成されたホストグループ

デフォルトでは、IdM サーバーは、全 IdM サーバーホストのホストグループ **ipaservers** を作成します。

直接および間接のグループメンバー

IdM のグループ属性は、直接メンバーと間接メンバーの両方に適用されます。ホストグループ B がホストグループ A のメンバーである場合、ホストグループ B のすべてのユーザーはホストグループ A の間接メンバーと見なされます。

46.2. IDM WEB UI でのホストグループの表示

Web インターフェイス (Web UI) を使用して IdM ホストグループを表示するには、次の手順に従います。

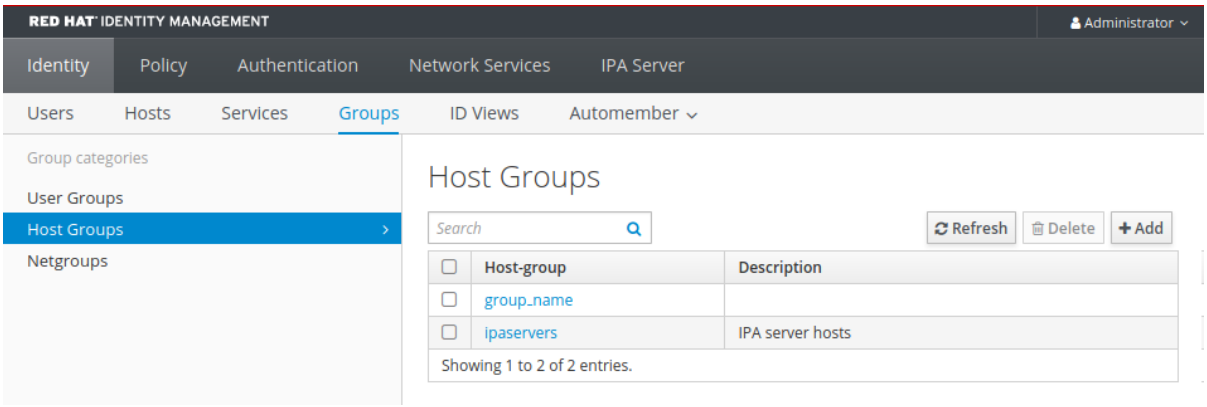
前提条件

- IdM、またはユーザー管理者ロールを管理する管理者権限

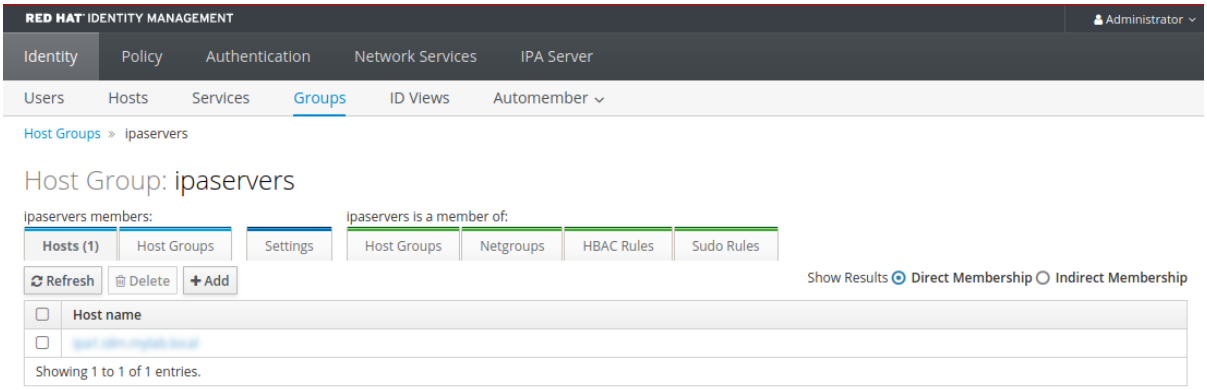
- IdM Web UI にログインしている。詳細は Web ブラウザーで IdM Web UI へのアクセス を参照してください。

手順

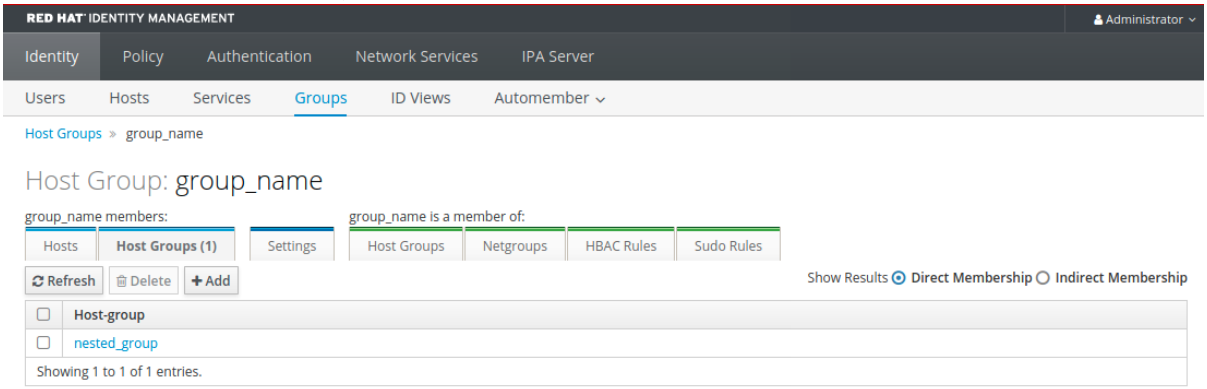
1. **Identity → Groups** をクリックし、**Host Groups** タブを選択します。
   - このページには、既存のホストグループとその説明が記載されています。
   - 特定のホストグループを検索できます。



2. リストのグループをクリックして、このグループに所属するホストを表示します。結果は、直接または間接メンバーに限定できます。



3. **ホストグループ** タブを選択して、このグループに所属するホストグループを表示します (ネスト化されたホストグループ)。結果は、直接または間接メンバーに限定できます。



46.3. IDM WEB UI でのホストグループの作成

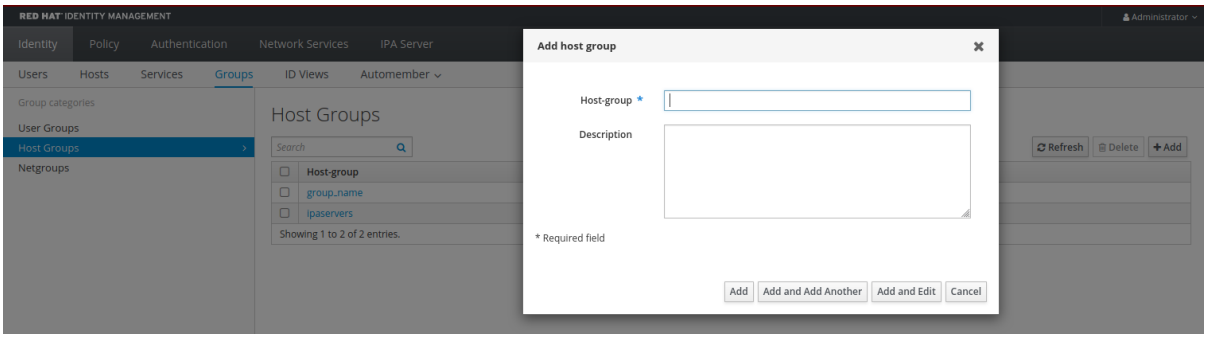
Web インターフェイス (Web UI) を使用して IdM ホストグループを作成するには、次の手順に従います。

前提条件

- IdM、またはユーザー管理者ロールを管理する管理者権限
- IdM Web UI にログインしている。詳細は Web ブラウザーで IdM Web UI へのアクセス を参照してください。

手順

1. **Identity → Groups** をクリックし、**Host Groups** タブを選択します。
2. **Add** をクリックします。**Add host group** ダイアログが表示されます。
3. Group: name (必須) および description (オプション) についての情報を指定します。
4. **Add** をクリックして確定します。



46.4. IDM WEB UI でのホストグループの削除

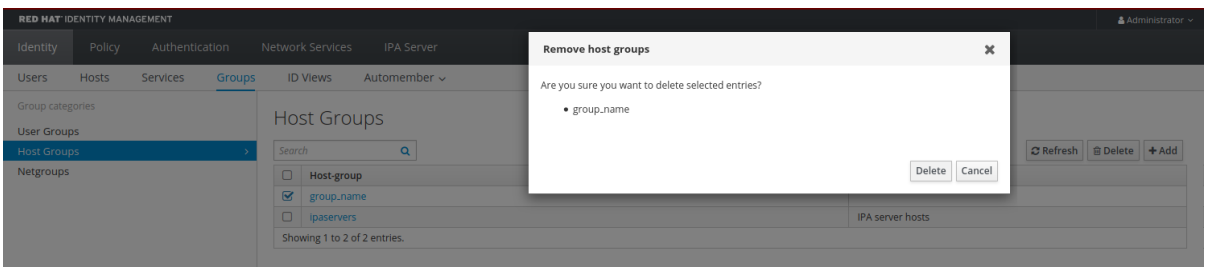
Web インターフェイス (Web UI) を使用して IdM ホストグループを削除するには、次の手順に従います。

前提条件

- IdM、またはユーザー管理者ロールを管理する管理者権限
- IdM Web UI にログインしている。詳細は Web ブラウザーで IdM Web UI へのアクセス を参照してください。

手順

1. **Identity → Groups** をクリックし、**Host Groups** タブを選択します。
2. 削除する IdM ホストグループを選択し、**削除** をクリックします。確認ダイアログが表示されます。
3. **Delete** をクリックして確定します。



注記

ホストグループを削除しても、IdM からグループメンバーは削除されません。

46.5. IDM WEB UI でのホストグループメンバーの追加

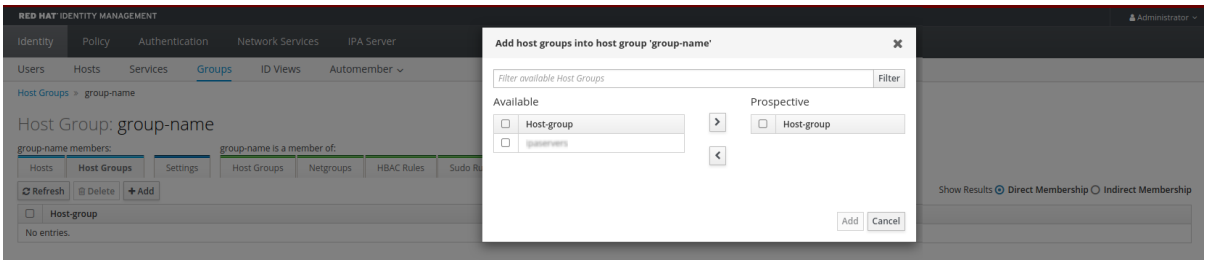
Web インターフェイス (Web UI) を使用して IdM にホストグループメンバーを追加するには、次の手順に従います。

前提条件

- IdM、またはユーザー管理者ロールを管理する管理者権限
- IdM Web UI にログインしている。詳細は Web ブラウザーで IdM Web UI へのアクセス を参照してください。

手順

1. **Identity → Groups** をクリックし、**Host Groups** タブを選択します。
2. メンバーを追加するグループの名前をクリックします。
3. 追加するメンバーのタイプに応じて、**ホスト** または **ホストグループ** をクリックします。対応するダイアログが表示されます。
4. 追加するホストまたはホストグループを選択し、› ボタンをクリックして **Prospective** コラムに移動します。
5. **Add** をクリックして確定します。



46.6. IDM WEB UI でのホストグループメンバーの削除

Web インターフェイス (Web UI) を使用して IdM のホストグループメンバーを削除するには、次の手順に従います。

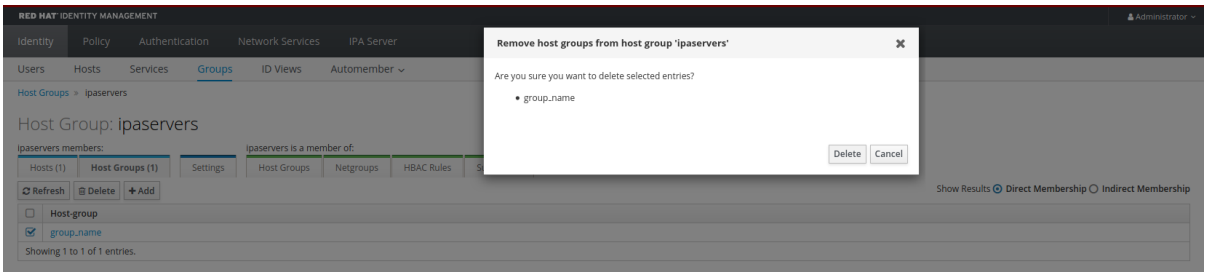
前提条件

- IdM、またはユーザー管理者ロールを管理する管理者権限
- IdM Web UI にログインしている。詳細は Web ブラウザーで IdM Web UI へのアクセス を参照してください。

手順

1. **Identity → Groups** をクリックし、**Host Groups** タブを選択します。
2. メンバーを削除するグループの名前をクリックします。

3. 削除するメンバーのタイプに応じて、**ホスト** または **ホストグループ** をクリックします。
4. 削除するメンバーの横にあるチェックボックスを選択します。
5. 削除をクリックします。確認ダイアログが表示されます。



6. Delete をクリックして確定します。選択したメンバーが削除されます。

46.7. WEB UI を使用した IDM ホストグループメンバーマネージャーの追加

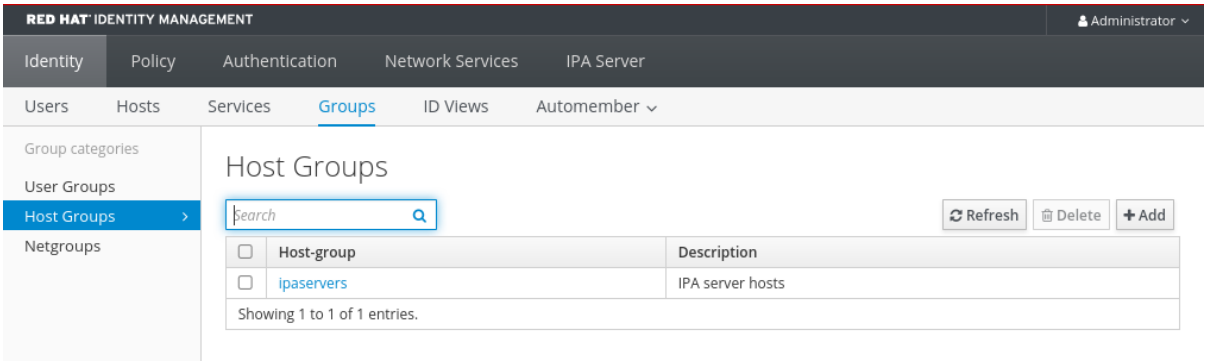
Web インターフェイス (Web UI) を使用して IdM でユーザーまたはユーザーグループをホストグループメンバーマネージャーとして追加するには、次の手順に従います。メンバーマネージャーは、ホストグループのメンバーマネージャーを IdM ホストグループに追加できますが、ホストグループの属性は変更できません。

前提条件

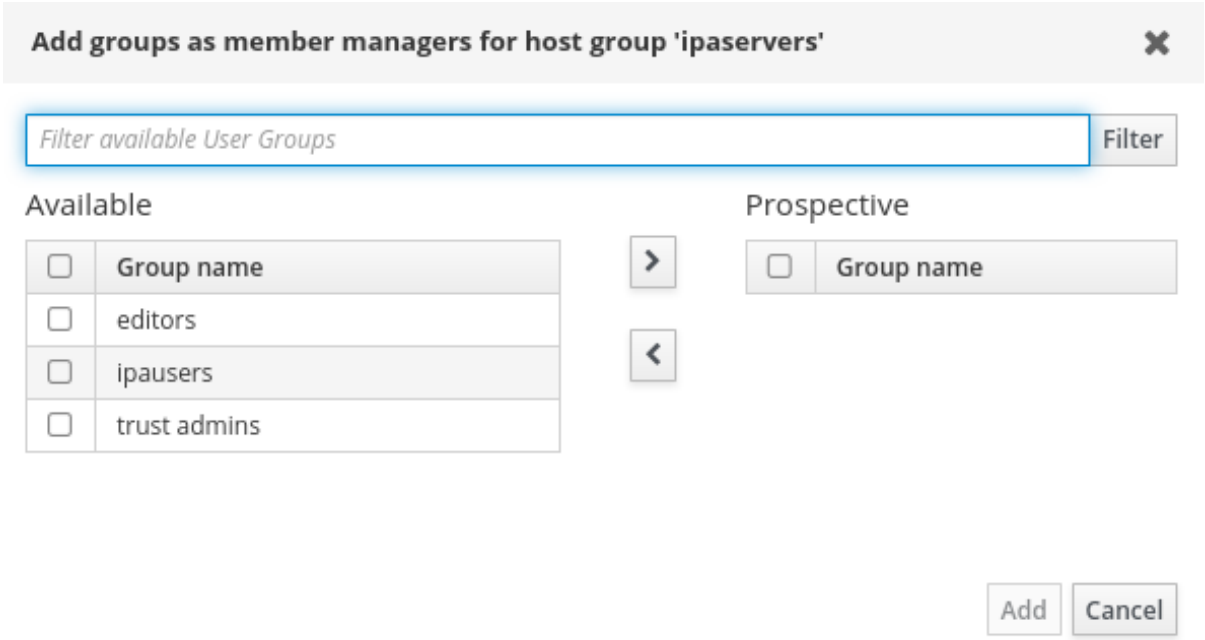
- IdM、またはユーザー管理者ロールを管理する管理者権限
- IdM Web UI にログインしている。詳細は Web ブラウザーで IdM Web UI へのアクセス を参照してください。
- メンバーマネージャーとして追加するホストグループ名と、管理するホストグループ名が必要です。

手順

1. **Identity → Groups** をクリックし、**Host Groups** タブを選択します。



2. メンバーマネージャーを追加するグループの名前をクリックします。
3. 追加するメンバーマネージャーのタイプに合わせて、メンバーマネージャータブの **User Groups** または **Users** をクリックします。対応するダイアログが表示されます。
4. **Add** をクリックします。



5. 追加するユーザーまたはユーザーグループを選択し、> ボタンをクリックして **Prospective** コラムに移動します。

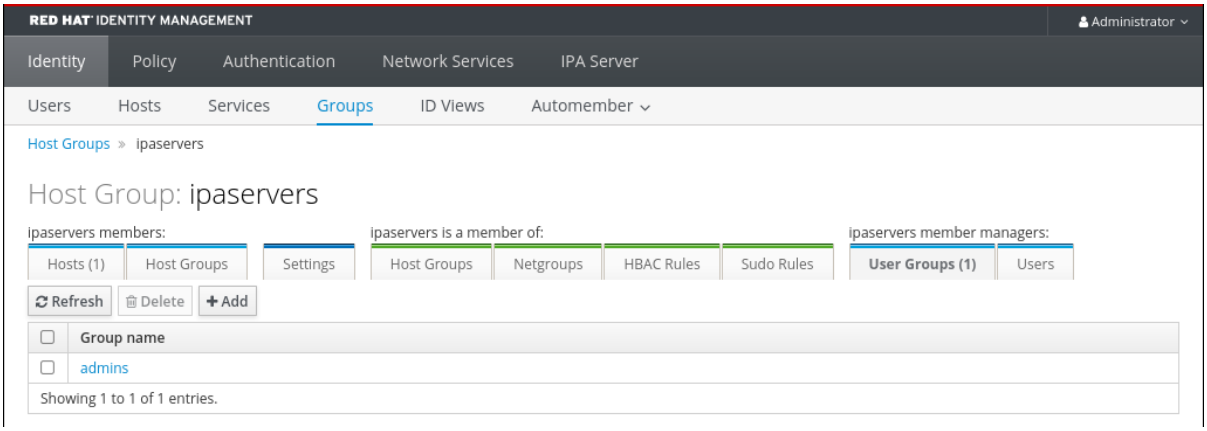
6. **Add** をクリックして確定します。

注記

ホストグループにメンバーマネージャーを追加してから、更新が Identity Management 環境のすべてのクライアントに広がるまでに時間がかかる場合があります。

検証手順

- ホストグループダイアログで、ユーザーグループまたはユーザーがグループまたはユーザーのメンバーマネージャーのリストに追加されていることを確認します。



46.8. WEB UI を使用した IDM ホストグループメンバーマネージャーの削除

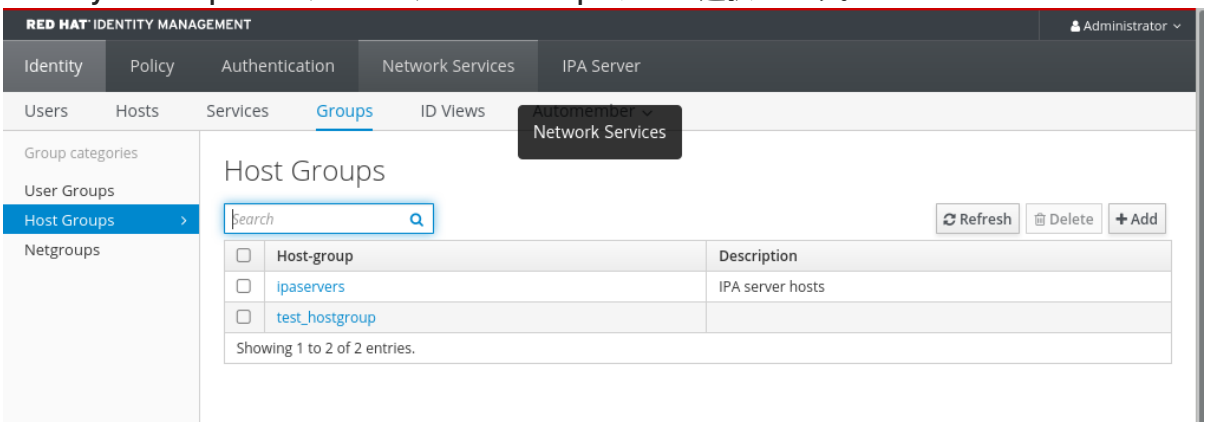
Web インターフェイス (Web UI) を使用して、IdM のホストグループメンバーマネージャーからユーザーまたはユーザーグループを削除するには、次の手順に従います。メンバーマネージャーは、IdM ホストグループからホストグループのメンバーマネージャーを削除できますが、ホストグループの属性は変更できません。

前提条件

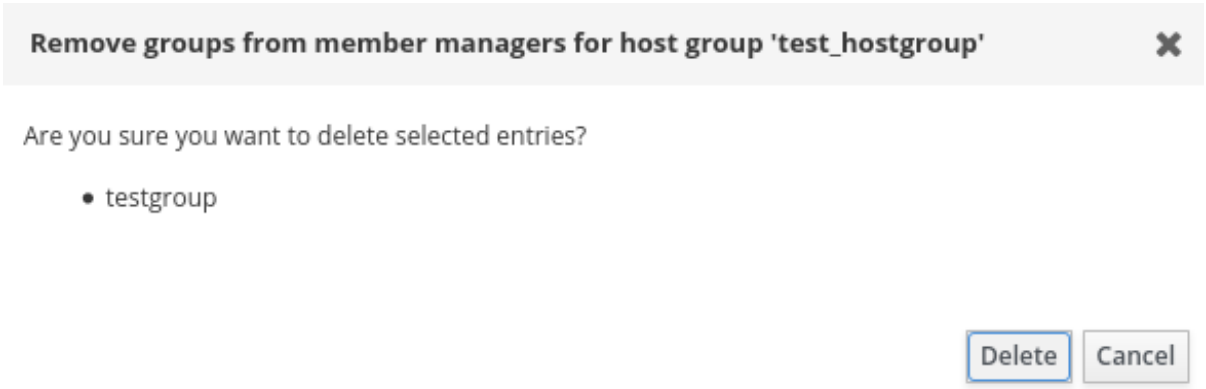
- IdM、またはユーザー管理者ロールを管理する管理者権限
- IdM Web UI にログインしている。詳細は Web ブラウザーで IdM Web UI へのアクセス を参照してください。
- 削除する既存のメンバーマネージャーのホストグループ名と、管理するホストグループ名が必要です。

手順

1. **Identity → Groups** をクリックし、**Host Groups** タブを選択します。



2. メンバーマネージャーを削除するグループの名前をクリックします。
3. 削除するメンバーマネージャーのタイプに合わせて、メンバーマネージャータブの **User Groups** または **Users** をクリックします。対応するダイアログが表示されます。
4. 削除するユーザーまたはユーザーグループを選択し、**Delete** をクリックします。
5. **Delete** をクリックして確定します。

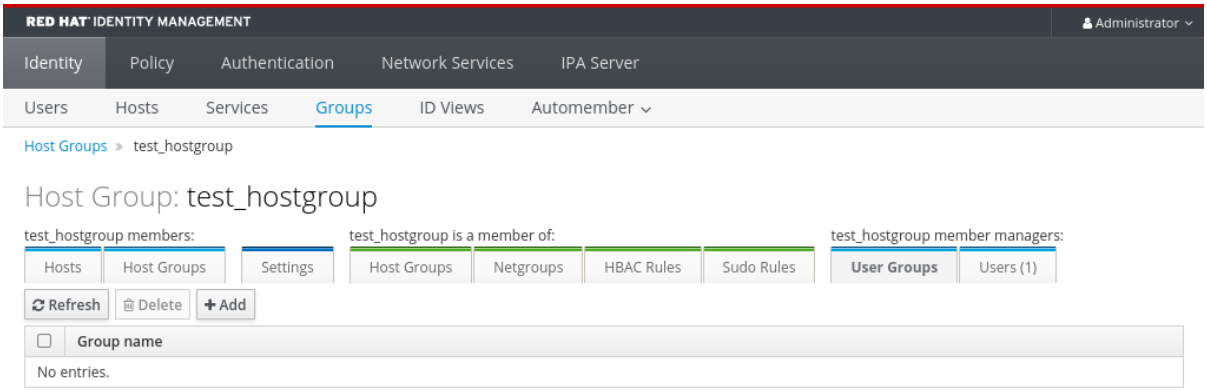


注記

ホストグループからメンバーマネージャーを削除してから、更新が Identity Management 環境のすべてのクライアントに広がるまでに時間がかかる場合があります。

検証手順

- ホストグループダイアログで、ユーザーグループまたはユーザーがグループまたはユーザーのメンバーマネージャーのリストから削除されていることを確認します。


RED HAT IDENTITY MANAGEMENT
Administrator
Identity
Policy
Authentication
Network Services
IPA Server
Users
Hosts
Services
Groups
ID Views
Automember
Host Groups » test_hostgroup
Host Group: test_hostgroup
test_hostgroup members:
Hosts
Host Groups
Settings
test_hostgroup is a member of:
Host Groups
Netgroups
HBAC Rules
Sudo Rules
test_hostgroup member managers:
User Groups
Users (1)
Refresh
Delete
Add
Group name
No entries.

第47章 ANSIBLE PLAYBOOK を使用したホストグループの管理

Identity Management (IdM) のホストグループ と、Ansible を使用して Identity Management (IdM) のホストグループに関係する操作を実行する方法について詳しく説明します。

- IdM のホストグループ
- IdM ホストグループを存在させる手順
- IdM ホストグループにホストを存在させる手順
- IdM ホストグループのネスト化
- IdM ホストグループにメンバーマネージャーを存在させる手順
- IdM ホストグループにホストを存在させないようにする方法
- ネスト化されたホストグループを IdM ホストグループに存在させないようにする方法
- IdM ホストグループにメンバーマネージャーを存在させないようにする方法

47.1. IDM のホストグループ

IdM ホストグループを使用すると、重要な管理タスク (特にアクセス制御) を一元管理できます。

ホストグループの定義

ホストグループは、一般的なアクセス制御ルールやその他の特性を持つ IdM ホストセットが含まれるエンティティーです。たとえば、企業の部門、物理的な場所、またはアクセス制御要件に基づいてホストグループを定義できます。

IdM のホストグループには以下が含まれます。

- IdM サーバーおよびクライアント
- その他の IdM ホストグループ

デフォルトで作成されたホストグループ

デフォルトでは、IdM サーバーは、全 IdM サーバーホストのホストグループ **ipaservers** を作成します。

直接および間接のグループメンバー

IdM のグループ属性は、直接メンバーと間接メンバーの両方に適用されます。ホストグループ B がホストグループ A のメンバーである場合、ホストグループ B のすべてのユーザーはホストグループ A の間接メンバーと見なされます。

47.2. ANSIBLE PLAYBOOK を使用して IDM ホストグループを存在させる手順

以下の手順に従って、Ansible Playbook を使用して、Identity Management (IdM) にホストグループが存在することを確認します。

注記

Ansible を使用しない場合には、**ipa hostgroup-add** コマンドでホストグループエントリーを IdM に作成します。ホストグループを IdM に追加すると、IdM でのホストグループの状態が present になります。Ansible は冪等性に依存しているので、Ansible を使用して IdM にホストグループを追加するには、ホストの状態を Present (**state: present**) として定義した Playbook を作成する必要があります。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **inventory.file** などのインベントリーファイル を作成し、そのファイルで、ターゲットに設定する IdM サーバーのリストと合わせて **ipaserver** を定義します。

```
[ipaserver]
server.idm.example.com
```

2. 必要なホストグループ情報を使用して Ansible Playbook ファイルを作成します。たとえば、**databases** という名前のホストグループを存在させるには、**- ipahostgroup** タスクで **name: databases** を指定します。この手順は、**/usr/share/doc/ansible-freeipa/playbooks/user/ensure-hostgroup-is-present.yml** ファイルのサンプルをコピーして変更し、簡素化できます。

```
---
- name: Playbook to handle hostgroups
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  # Ensure host-group databases is present
  - ipahostgroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: databases
      state: present
```

Playbook で **state: present** が指定されていると、IdM にホストグループがない場合のホストグループの追加要求という意味です。

3. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file path_to_playbooks_directory/ensure-
hostgroup-is-present.yml
```

検証手順

1. admin として **ipaserver** にログインします。

```
$ ssh admin@server.idm.example.com
Password:
[admin@server /]$
```

2. admin の Kerberos チケットを要求します。

```
$ kinit admin
Password for admin@IDM.EXAMPLE.COM:
```

3. IdM に存在させるホストグループに関する情報を表示します。

```
$ ipa hostgroup-show databases
  Host-group: databases
```

データベース ホストグループが IdM に存在します。

47.3. ANSIBLE PLAYBOOK を使用して IDM ホストグループにホストを存在させる手順

以下の手順に従って、Ansible Playbook を使用して、Identity Management (IdM) のホストグループにホストが存在することを確認します。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- Ansible Playbook で参照するホストが IdM に存在する。詳細は Ansible Playbook を使用した IdM ホストエントリーの存在の確認 を参照してください。

- Ansible Playbook ファイルから参照するホストグループが IdM に追加されている。詳細は、Ansible Playbook を使用して IdM ホストグループを存在させる手順 を参照してください。

手順

1. **inventory.file** などのインベントリーファイル を作成し、そのファイルで、ターゲットに設定する IdM サーバーのリストと合わせて **ipaserver** を定義します。

```
[ipaserver]
server.idm.example.com
```

2. 必要なホスト情報を使用して Ansible Playbook ファイルを作成します。**ipahostgroup** 変数の **name** パラメーターを使用してホストグループの名前を指定します。**ipahostgroup** 変数の **host** パラメーターを使用してホストの名前を指定します。この手順は、**/usr/share/doc/ansible-freeipa/playbooks/hostgroup/ensure-hosts-and-hostgroups-are-present-in-hostgroup.yml** ファイルのサンプルをコピーして変更し、簡素化できます。

```
---
- name: Playbook to handle hostgroups
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  # Ensure host-group databases is present
  - ipahostgroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: databases
      host:
      - db.idm.example.com
      action: member
```

この Playbook は、**db.idm.example.com** ホストを **データベース** ホストグループに追加します。**action: member** の行は、Playbook の実行時に **databases** グループ自体の追加を試行しないことを示します。代わりに、**db.idm.example.com** の **databases** への追加を試行するだけです。

3. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file path_to_playbooks_directory/ensure-hosts-
or-hostgroups-are-present-in-hostgroup.yml
```

検証手順

1. admin として **ipaserver** にログインします。

```
$ ssh admin@server.idm.example.com
Password:
[admin@server /]$
```

2. admin の Kerberos チケットを要求します。

```
$ kinit admin
Password for admin@IDM.EXAMPLE.COM:
```

3. ホストグループに関する情報を表示して、どのホストが存在するかを確認します。

```
$ ipa hostgroup-show databases
  Host-group: databases
  Member hosts: db.idm.example.com
```

db.idm.example.com ホストは、**database** ホストグループのメンバーとして存在します。

47.4. ANSIBLE PLAYBOOK を使用した IDM ホストグループのネスト化

以下の手順に従って、Ansible Playbook を使用して、Identity Management (IdM) ホストグループにネスト化されたホストグループが存在することを確認します。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- Ansible Playbook ファイルから参照するホストグループが IdM に存在する。詳細は、Ansible Playbook を使用して IdM ホストグループを存在させる手順 を参照してください。

手順

1. **inventory.file** などのインベントリーファイル を作成し、そのファイルで、ターゲットに設定する IdM サーバーのリストと合わせて **ipaserver** を定義します。

```
[ipaserver]
server.idm.example.com
```

2. 必要なホストグループ情報を使用して Ansible Playbook ファイルを作成します。ネストされたホストグループ **A** が、Ansible Playbook のホストグループ **B** に存在することを確認するには、**- ipahostgroup** 変数で **name** 変数を使用して、ホストグループ **B** の名前を指定します。**hostgroup** 変数でネスト化されたホストグループ **A** の名前を指定します。この手順は、**/usr/share/doc/ansible-freeipa/playbooks/hostgroup/ensure-hosts-and-hostgroups-are-present-in-hostgroup.yml** ファイルのサンプルをコピーして変更し、簡素化できます。

```
---
```

```
- name: Playbook to handle hostgroups
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  # Ensure hosts and hostgroups are present in existing databases hostgroup
  - ipahostgroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: databases
      hostgroup:
      - mysql-server
      - oracle-server
      action: member
```

この Ansible Playbook は、**database** ホストグループに **myqsl-server** および **oracle-server** ホストグループが存在することを確認します。**action: member** 行は、Playbook が実行されると、**databases** グループ自体を IdM に追加しようとはしません。

3. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file path_to_playbooks_directory/ensure-hosts-
or-hostgroups-are-present-in-hostgroup.yml
```

検証手順

1. admin として **ipaserver** にログインします。

```
$ ssh admin@server.idm.example.com
Password:
[admin@server /]$
```

2. admin の Kerberos チケットを要求します。

```
$ kinit admin
Password for admin@IDM.EXAMPLE.COM:
```

3. ネスト化されたホストグループが含まれるホストグループに関する情報を表示します。

```
$ ipa hostgroup-show databases
  Host-group: databases
  Member hosts: db.idm.example.com
  Member host-groups: mysql-server, oracle-server
```

mysql-server および **oracle-server** ホストグループは、**databases** ホストグループに存在します。

47.5. ANSIBLE PLAYBOOK を使用して IDM ホストグループにメンバーマネージャーを存在させる手順

以下の手順では、Ansible Playbook を使用して、IdM ホストおよびホストグループにメンバーマネージャーを存在させる方法を説明します。

前提条件

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- メンバーマネージャーとして追加するホストまたはホストグループの名前と、管理するホストグループ名が必要です。

手順

1. **inventory.file** などのインベントリーファイルを作成して、そのファイルに **ipaserver** を定義します。

```
[ipaserver]
server.idm.example.com
```

2. 必要なホストおよびホストグループメンバー管理情報を使用して Ansible Playbook ファイルを作成します。

```
---

- name: Playbook to handle host group membership management
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure member manager user example_member is present for group_name
      ipahostgroup:
        ipaadmin_password: "{{ ipaadmin_password }}"
        name: group_name
        membermanager_user: example_member

  - name: Ensure member manager group project_admins is present for group_name
      ipahostgroup:
        ipaadmin_password: "{{ ipaadmin_password }}"
        name: group_name
        membermanager_group: project_admins
```

3. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file path_to_playbooks_directory/add-member-
managers-host-groups.yml
```

検証手順

ipa group-show コマンドを使用して group_name グループのメンバーマネージャーとして example_member と project_admins が含まれていることを確認できます。

1. 管理者として **ipaserver** にログインします。

```
$ ssh admin@server.idm.example.com
Password:
[admin@server /]$
```

2. **testhostgroup** に関する情報を表示します。

```
ipaserver]$ ipa hostgroup-show group_name
  Host-group: group_name
  Member hosts: server.idm.example.com
  Member host-groups: testhostgroup2
  Membership managed by groups: project_admins
  Membership managed by users: example_member
```

関連情報

- **ipa hostgroup-add-member-manager --help** を参照してください。
- **ipa** の man ページを参照してください。

47.6. ANSIBLE PLAYBOOK を使用して IDM ホストグループにホストを存在させないようにする方法

以下の手順に従って、Ansible Playbook を使用して、Identity Management (IdM) のホストグループにホストがないことを確認します。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

- Ansible Playbook で参照するホストが IdM に存在する。詳細は Ansible Playbook を使用した IdM ホストエントリーの存在の確認 を参照してください。
- Ansible Playbook ファイルから参照するホストグループが IdM に存在する。詳細は、Ansible Playbook を使用して IdM ホストグループを存在させる手順 を参照してください。

手順

1. **inventory.file** などのインベントリーファイル を作成し、そのファイルで、ターゲットに設定する IdM サーバーのリストと合わせて **ipaserver** を定義します。

```
[ipaserver]
server.idm.example.com
```

2. 必要なホストおよびホストグループ情報を使用して Ansible Playbook ファイルを作成します。**ipahostgroup** 変数の **name** パラメーターを使用してホストグループの名前を指定します。**ipahostgroup** 変数の **host** パラメーターを使用して、ホストグループに、存在させないようにするホストの名前を指定します。この手順は、**/usr/share/doc/ansible-freeipa/playbooks/hostgroup/ensure-hosts-and-hostgroups-are-absent-in-hostgroup.yml** ファイルのサンプルをコピーして変更し、簡素化できます。

```
---
- name: Playbook to handle hostgroups
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  # Ensure host-group databases is absent
  - ipahostgroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: databases
      host:
      - db.idm.example.com
      action: member
      state: absent
```

この Playbook では、**databases** ホストグループに **db.idm.example.com** ホストを存在させないようにできます。**action: member** の行で、Playbook の実行時に **databases** グループ自体の削除を試行しないように指定します。

3. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file path_to_playbooks_directory/ensure-hosts-
or-hostgroups-are-absent-in-hostgroup.yml
```

検証手順

1. admin として **ipaserver** にログインします。

```
$ ssh admin@server.idm.example.com
Password:
[admin@server /]$
```

2. admin の Kerberos チケットを要求します。

```
$ kinit admin
Password for admin@IDM.EXAMPLE.COM:
```

3. ホストグループと、そのホストグループに含まれるホストに関する情報を表示します。

```
$ ipa hostgroup-show databases
  Host-group: databases
  Member host-groups: mysql-server, oracle-server
```

db.idm.example.com ホストは **データベース** ホストグループに存在していません。

47.7. ANSIBLE PLAYBOOK を使用して IDM ホストグループに、ネスト化されたホストグループを存在させないようにする方法

以下の手順に従って、Ansible Playbook を使用して、Identity Management (IdM) の外部ホストグループからネスト化されたホストグループがないことを確認します。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- Ansible Playbook ファイルから参照するホストグループが IdM に存在する。詳細は、Ansible Playbook を使用して IdM ホストグループを存在させる手順 を参照してください。

手順

1. **inventory.file** などのインベントリーファイル を作成し、そのファイルで、ターゲットに設定する IdM サーバーのリストと合わせて **ipaserver** を定義します。

```
[ipaserver]
server.idm.example.com
```

2. 必要なホストグループ情報を使用して Ansible Playbook ファイルを作成します。**-ipahostgroup** 変数の **name** 変数を使用して、外部ホストグループの名前を指定します。**hostgroup** 変数でネスト化されたホストグループの名前を指定します。この手順は、**/usr/share/doc/ansible-freeipa/playbooks/hostgroup/ensure-hosts-and-hostgroups-are-absent-in-hostgroup.yml** ファイルのサンプルをコピーして変更し、簡素化できます。

```
---
- name: Playbook to handle hostgroups
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  # Ensure hosts and hostgroups are absent in existing databases hostgroup
  - ipahostgroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: databases
      hostgroup:
      - mysql-server
      - oracle-server
      action: member
      state: absent
```

この Playbook は、**mysql-server** および **oracle-server** ホストグループが **databases** ホストグループにないことを確認します。**action: member** 行は、Playbook の実行時に、**databases** グループ自体の IdM からの削除は試行されないようにします。

3. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file path_to_playbooks_directory/ensure-hosts-
or-hostgroups-are-absent-in-hostgroup.yml
```

検証手順

1. admin として **ipaserver** にログインします。

```
$ ssh admin@server.idm.example.com
Password:
[admin@server /]$
```

2. admin の Kerberos チケットを要求します。

```
$ kinit admin
Password for admin@IDM.EXAMPLE.COM:
```

3. ネスト化されたホストグループを存在させないホストグループに関する情報を表示します。

```
$ ipa hostgroup-show databases
  Host-group: databases
```

この出力では、**mysql-server** および **oracle-server** のネスト化されたホストグループが、外部 **databases** のホストグループにないことを確認します。

47.8. ANSIBLE PLAYBOOK を使用して IDM ホストグループを存在させないようにする方法

以下の手順に従って、Ansible Playbook を使用して Identity Management (IdM) にホストグループがないことを確認します。

注記

Ansible を使用しない場合には、**ipa hostgroup-del** コマンドでホストグループエントリーを IdM から削除します。IdM からホストグループを削除すると、IdM にホストグループが存在しない状態になります。Ansible は冪等性に依存しているので、Ansible を使用して IdM からホストグループを削除するには、ホストの状態を Absent (**state: absent**) として定義した Playbook を作成する必要があります。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **inventory.file** などのインベントリーファイル を作成し、そのファイルで、ターゲットに設定する IdM サーバーのリストと合わせて **ipaserver** を定義します。

```
[ipaserver]
server.idm.example.com
```

2. 必要なホストグループ情報を使用して Ansible Playbook ファイルを作成します。この手順は、**/usr/share/doc/ansible-freeipa/playbooks/user/ensure-hostgroup-is-absent.yml** ファイルのサンプルをコピーして変更し、簡素化できます。

```
---
- name: Playbook to handle hostgroups
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - Ensure host-group databases is absent
    ipahostgroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: databases
      state: absent
```

この Playbook では、IdM から **databases** ホストグループを存在させないようにします。**state: absent** は、IdM からホストグループが削除されていない限り、ホストグループの削除要求を意味します。

3. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file path_to_playbooks_directory/ensure-
hostgroup-is-absent.yml
```

検証手順

1. admin として **ipaserver** にログインします。

```
$ ssh admin@server.idm.example.com
Password:
[admin@server /]$
```

2. admin の Kerberos チケットを要求します。

```
$ kinit admin
Password for admin@IDM.EXAMPLE.COM:
```

3. 存在させないようにするホストグループの情報を表示します。

```
$ ipa hostgroup-show databases
ipa: ERROR: databases: host group not found
```

databases ホストグループが IdM に存在しません。

47.9. ANSIBLE PLAYBOOK を使用して IDM ホストグループからホストを存在させないようにする方法

以下の手順では、Ansible Playbook を使用して、IdM ホストおよびホストグループにメンバーマネージャーを存在させないようにする方法を説明します。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- メンバーマネージャーから削除するユーザーまたはユーザーグループの名前と、管理するホストグループの名前が必要です。

手順

1. **inventory.file** などのインベントリーファイルを作成して、そのファイルに **ipaserver** を定義します。

```
[ipaserver]
server.idm.example.com
```

2. 必要なホストおよびホストグループメンバー管理情報を使用して Ansible Playbook ファイルを作成します。

```
---

- name: Playbook to handle host group membership management
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure member manager host and host group members are absent for
group_name
    ipahostgroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: group_name
      membermanager_user: example_member
      membermanager_group: project_admins
      action: member
      state: absent
```

3. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file path_to_playbooks_directory/ensure-
member-managers-host-groups-are-absent.yml
```

検証手順

ipa group-show コマンドを使用して、**group_name** グループに **example_member** または **project_admins** がメンバーマネージャーとして含まれているかどうかを確認できます。

1. 管理者として **ipaserver** にログインします。

```
$ ssh admin@server.idm.example.com
Password:
[admin@server /]$
```

2. **testhostgroup** に関する情報を表示します。

```
ipaserver]$ ipa hostgroup-show group_name
  Host-group: group_name
  Member hosts: server.idm.example.com
  Member host-groups: testhostgroup2
```

関連情報

- **ipa hostgroup-add-member-manager --help** を参照してください。
- **ipa** の man ページを参照してください。

第48章 ユーザー、ホスト、およびサービス用の KERBEROS プリンシパルエイリアスの管理

新しいユーザー、ホスト、またはサービスを作成すると、以下の形式で Kerberos プリンシパルが自動的に追加されます。

- **user_name@REALM**
- host/**host_name@REALM**
- **service_name/host_name@REALM**

管理者は、エイリアスを使用して、ユーザー、ホスト、またはサービスが Kerberos アプリケーションに対して認証できるようにすることができます。これは、次のような状況で役立ちます。

- ユーザー名の変更後に、ユーザーが以前のユーザー名と新しいユーザー名の両方でログインする必要がある。
- IdM Kerberos レルムがメールドメインと異なる場合でも、ユーザーはメールアドレスを使用してログインする必要がある。

ユーザーの名前を変更すると、オブジェクトはエイリアスと以前の正規プリンシパル名を保持することに注意してください。

48.1. KERBEROS プリンシパルエイリアスの追加

Identity Management (IdM) 環境では、エイリアス名を既存の Kerberos プリンシパルに関連付けることができます。これにより、セキュリティーが強化され、IdM ドメイン内の認証プロセスが簡素化されます。

手順

- エイリアス名 **useralias** をアカウント **user** に追加するには、次のように入力します。

```
# ipa user-add-principal <user> <useralias>
--------------------------------
Added new aliases to user "user"
--------------------------------
        User login: user
  Principal alias: user@IDM.EXAMPLE.COM, useralias@IDM.EXAMPLE.COM
```

ホストまたはサービスにエイリアスを追加するには、代わりにそれぞれ **ipa host-add-principal** または **ipa service-add-principal** コマンドを使用します。

エイリアス名を使用して認証する場合は、**kinit** コマンドで **-C** オプションを使用します。

```
# kinit -C <useralias>
Password for <user>@IDM.EXAMPLE.COM:
```

48.2. KERBEROS プリンシパルエイリアスの削除

Identity Management (IdM) 環境で Kerberos プリンシパルに関連付けられているエイリアス名を削除できます。

手順

- アカウント **user** からエイリアス **useralias** を削除するには、次のように入力します。

```
# ipa user-remove-principal <user> <useralias>
--------------------------------
Removed aliases from user "user"
--------------------------------
  User login: user
  Principal alias: user@IDM.EXAMPLE.COM
```

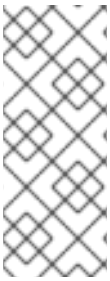
ホストまたはサービスからエイリアスを削除するには、代わりにそれぞれ **ipa host-remove-principal** または **ipa service-remove-principal** コマンドを使用します。

正規のプリンシパル名は削除できないことに注意してください。

```
# ipa user-show <user>
  User login: user
  ...
  Principal name: user@IDM.EXAMPLE.COM
  ...

# ipa user-remove-principal user user
ipa: ERROR: invalid 'krbprincipalname': at least one value equal to the canonical principal name must be present
```

48.3. KERBEROS エンタープライズプリンシパルエイリアスの追加

Identity Management (IdM) 環境では、エンタープライズプリンシパルエイリアス名を既存の Kerberos エンタープライズプリンシパルに関連付けることができます。エンタープライズプリンシパルエイリアスは、ユーザープリンシパル名 (UPN) 接尾辞、NetBIOS 名、または信頼された Active Directory フォレストドメインのドメイン名以外の任意のドメイン接尾辞を使用できます。



注記

エンタープライズプリンシパルエイリアスを追加または削除する場合は、2 つのバックスラッシュ (\\) を使用して @ 記号をエスケープします。エスケープしないと、シェルが @ 記号を Kerberos レルム名の一部として解釈し、次のエラーが発生します。

```
ipa: ERROR: The realm for the principal does not match the realm for this IPA server
```

手順

- エンタープライズプリンシパルエイリアス **user@example.com** を **user** アカウントに追加するには、以下を実行します。

```
# ipa user-add-principal <user> <user\\@example.com>
--------------------------------
Added new aliases to user "user"
--------------------------------
        User login: user
  Principal alias: user@IDM.EXAMPLE.COM, user\@example.com@IDM.EXAMPLE.COM
```

ホストまたはサービスにエンタープライズエイリアスを追加するには、代わりにそれぞれ **ipa host-add-principal** または **ipa service-add-principal** コマンドを使用します。

エンタープライズプリンシパル名を使用して認証する場合は、**kinit** コマンドで **-E** オプションを使用します。

```
# kinit -E <user@example.com>
Password for user\@example.com@IDM.EXAMPLE.COM:
```

48.4. KERBEROS エンタープライズプリンシパルエイリアスの削除

Identity Management (IdM) 環境で Kerberos エンタープライズプリンシパルに関連付けられているエンタープライズプリンシパルエイリアス名を削除できます。

注記

エンタープライズプリンシパルエイリアスを追加または削除する場合は、2 つのバックスラッシュ (\\) を使用して @ 記号をエスケープします。エスケープしないと、シェルが @ 記号を Kerberos レルム名の一部として解釈し、次のエラーが発生します。

```
ipa: ERROR: The realm for the principal does not match the realm for this IPA server
```

手順

- エンタープライズプリンシパルエイリアス **user@example.com** をアカウント **user** から削除するには、次のように入力します。

```
# ipa user-remove-principal <user> <user\\@example.com>
--------------------------------
Removed aliases from user "user"
--------------------------------
  User login: user
  Principal alias: user@IDM.EXAMPLE.COM
```

ホストまたはサービスからエイリアスを削除するには、代わりにそれぞれ **ipa host-remove-principal** または **ipa service-remove-principal** コマンドを使用します。

第49章 KERBEROS フラグの管理

Kerberos フラグは、Kerberos 対応ネットワーク環境内で認証メカニズム、承認レベル、およびセキュリティープロトコルを指定するために重要です。Kerberos フラグを使用すると、安全なアクセス制御を確保し、不正アクセスから保護し、異なる Kerberos 実装間の相互運用性を向上させることができます。

49.1. サービスおよびホスト向けの KERBEROS フラグ

さまざまな Kerberos フラグを使用して、Kerberos チケットの動作に関する特定の側面を定義できます。これらのフラグは、サービスとホストの Kerberos プリンシパルに追加できます。

Identity Management(IdM) のプリンシパルは、以下の Kerberos フラグを受け入れます。

- **OK_AS_DELEGATE**
  このフラグを使用して、委譲用に信頼される Kerberos チケットを指定します。

  Active Directory(AD) クライアントは、Kerberos チケットで **OK_AS_DELEGATE** フラグをチェックして、ユーザーの認証情報を特定サーバーに転送または委譲できるかどうかを判断します。AD は、TGT(Ticket-granting Ticket) を OK_AS_DELEGATE が設定されたサービスまたはホストにのみ転送します。このフラグを使用すると、SSSD(System Security Services デーモン) は、IdM クライアントマシンのデフォルトの Kerberos 認証情報キャッシュに AD ユーザー TGT を追加できます。

- **REQUIRES_PRE_AUTH**
  このフラグを使用して、事前認証チケットのみがプリンシパルに対して認証できることを指定します。

  REQUIRES_PRE_AUTH フラグを設定すると、キー配布センター (KDC) は追加の認証を要求します。KDC は、TGT が事前認証されている場合に限り、**REQUIRES_PRE_AUTH** が設定されたプリンシパルに TGT を発行します。

  REQUIRES_PRE_AUTH をクリアすると、選択したサービスまたはホストの事前認証を無効にすることができます。これにより、KDC の負荷は軽減されますが、長期キーに対するブルートフォース攻撃が成功する可能性がわずかに高まります。

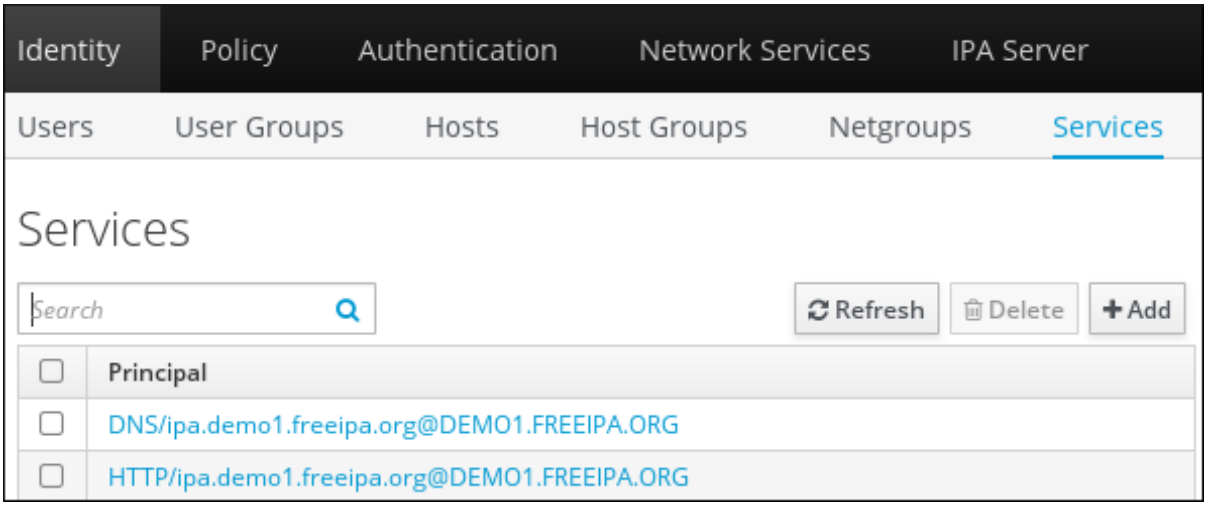
- **OK_TO_AUTH_AS_DELEGATE**
  OK_TO_AUTH_AS_DELEGATE フラグを使用して、サービスがユーザーの代わりに kerberos チケットを取得できることを指定します。ユーザーに代わって他のチケットを取得するには、サービスに **OK_AS_DELEGATE** フラグと、キー配布センター側で許可された対応するポリシー決定が必要であることに注意してください。

49.2. WEB UI からの KERBEROS フラグの設定

IdM Web UI を使用して Kerberos フラグを設定できます。次の手順では、プリンシパルに Kerberos フラグを設定します。

手順

1. メニューで **アイデンティティー → Services** を選択します。



2. フラグを追加するサービスをクリックします。

3. 設定するオプションのチェックを選択します。

   - **OK_AS_DELEGATE** フラグを設定するには、**委譲に対して信頼されているか** どうかをチェックします。
   - **REQUIRES_PRE_AUTH** フラグを設定するには、**事前認証が必要を** チェックします。
   - **OK_TO_AUTH_AS_DELEGATE** フラグを設定するには、**Trusted to authenticate as user を** チェックします。

49.3. コマンドラインからの KERBEROS フラグの設定および削除

コマンドラインを使用して Kerberos フラグを追加または削除できます。**ipa service-mod** コマンドは、フラグに次のコマンドオプションを使用します。

- **OK_AS_DELEGATE**:**--ok-as-delegate**
- **REQUIRES_PRE_AUTH**:**--requires-pre-auth**
- **OK_TO_AUTH_AS_DELEGATE**:**--ok-to-auth-as-delegate**

オプション値を **1** に設定すると、原則のフラグが有効になります。オプション値を **0** に設定すると、フラグが無効になります。

次の手順では、**service/ipa.example.com** @example.com プリンシパルの **OK_AS_DELEGATE** フラグを有効または無効にします。

手順

- **service/ipa.example.com@example.com** プリンシパルに **OK_AS_DELEGATE** フラグを追加するには、次を実行します。

```
$ ipa service-mod service/ipa.example.com@EXAMPLE.COM --ok-as-delegate=1
```

- **service/ipa.example.com@example.com** プリンシパルから **OK_AS_DELEGATE** フラグを削除するには、次を実行します。

```
$ ipa service-mod service/ipa.example.com@EXAMPLE.COM --ok-as-delegate=0
```

49.4. コマンドラインからの KERBEROS フラグの表示

コマンドラインを使用して Kerberos フラグ設定を表示できます。次の手順では、**demo/ipa.example.com** @EXAMPLE.COM プリンシパルの **OK_AS_DELEGATE** フラグを表示します。

手順

プリンシパルに **OK_AS_DELEGATE** が設定されているかどうかを確認するには:

1. **kvno** ユーティリティーを実行します。

```
$ kvno demo/ipa.example.com@EXAMPLE.COM
```

2. フラグ設定を表示するには、**klist -f** コマンドを実行します。**0** 文字は **OK_AS_DELEGATE** フラグが無効であることを意味します。

```
$ klist -f
Ticket cache: KEYRING:persistent:0:0
Default principal: admin@EXAMPLE.COM

Valid starting  Expires   Service principal
02/19/2024 09:59:02 02/20/2024 08:21:33 demo/ipa/example.com@EXAMPLE.COM
    Flags: FAT0
```

第50章 PAC 情報による KERBEROS セキュリティーの強化

RHEL 8.5 以降の Identity Management (IdM) では、特権属性証明書 (PAC) 情報をデフォルトで使用できます。また、RHEL 8.5 より前にインストールされた IdM デプロイメントで、セキュリティー識別子 (SID) を有効にすることもできます。

50.1. IDM での特権属性証明書 (PAC) の使用

セキュリティーを強化するために、RHEL Identity Management (IdM) は、新しいデプロイメントでデフォルトで特権属性証明書 (PAC) 情報を含む Kerberos チケットを発行するようになりました。PAC には、セキュリティー識別子 (SID)、グループメンバーシップ、ホームディレクトリー情報など、Kerberos プリンシパルに関する豊富な情報が含まれています。

Microsoft Active Directory (AD) がデフォルトで使用する SID は、再利用されることのないグローバルに一意の識別子です。SID は複数の名前空間を表します。各ドメインには、各オブジェクトの SID の接頭辞である SID があります。

RHEL 8.5 以降、IdM サーバーまたはレプリカをインストールすると、インストールスクリプトはデフォルトでユーザーおよびグループの SID を生成します。これにより、IdM が PAC データを操作できるようになります。RHEL 8.5 より前に IdM をインストールし、AD ドメインとの信頼を設定していない場合は、IdM オブジェクトの SID が生成されていない可能性があります。IdM オブジェクトの SID の生成に関する詳細は、IdM でのセキュリティー識別子 (SID) の有効化 を参照してください。

Kerberos チケットの PAC 情報を評価することで、リソースアクセスをより詳細に制御できます。たとえば、あるドメインの管理者アカウントは、他のドメインの管理者アカウントとは一意に異なる SID を持っています。AD ドメインへの信頼がある IdM 環境では、UID が 0 のすべての Linux **root** アカウントなど、さまざまな場所で繰り返される可能性のある単純なユーザー名や UID ではなく、グローバルに一意の SID に基づいてアクセス制御を設定できます。

50.2. IDM でのセキュリティー識別子 (SID) の有効化

RHEL 8.5 より前に IdM をインストールし、AD ドメインとの信頼を設定していない場合は、IdM オブジェクトのセキュリティー識別子 (SID) が生成されていない可能性があります。これは、以前は SID を生成する場合、**ipa-adtrust-install** コマンドを実行して **信頼コントローラー** ロールを IdM サーバーに追加することが唯一の方法だったためです。

RHEL 8.6 以降、IdM の Kerberos では、IdM オブジェクトに SID が必要です。これは、特権属性証明書 (PAC) 情報に基づくセキュリティーに必要です。

前提条件

- RHEL 8.5 より前に IdM をインストールしている。
- Active Directory ドメインとの信頼の設定の一部である **ipa-sidgen** タスクを実行していない。
- IdM 管理者アカウントとして認証可能である。

手順

- SID の使用を有効にし、**SIDgen** タスクをトリガーして、既存のユーザーとグループの SID を生成します。このタスクはリソースを大量に消費する可能性があります。

```
[root@server ~]# ipa config-mod --enable-sid --add-sids
```

検証

- IdM **admin** ユーザーアカウントエントリーに、**-500** で終わる SID (ドメイン管理者用に予約されている SID) のある **ipantsecurityidentifier** 属性があることを確認します。

```
[root@server ~]# ipa user-show admin --all | grep ipantsecurityidentifier
  ipantsecurityidentifier: S-1-5-21-2633809701-976279387-419745629-500
```

関連情報

- IdM での特権属性証明書 (PAC) の使用
- How to solve users unable to authenticate to IPA/IDM with PAC issues - S4U2PROXY_EVIDENCE_TKT_WITHOUT_PAC error KCS ソリューション
- 信頼コントローラーおよび信頼エージェント
- Integrate SID configuration into base IPA installers

第51章 KERBEROS チケットポリシーの管理

Identity Management (IdM) の Kerberos チケットポリシーは、Kerberos チケットアクセス、期間、および更新に対する制限を設定します。IdM サーバーで実行している Key Distribution Center (KDC) の Kerberos チケットポリシーを設定できます。

Kerberos チケットポリシーを管理する場合、次の概念や操作を実行します。

- IdM KDC のロール
- IdM Kerberos チケットポリシータイプ
- Kerberos 認証インジケーター
- IdM サービスの認証インジケーターの有効化
- グローバルチケットライフサイクルポリシーの設定
- 認証インジケーターごとのグローバルチケットポリシーの設定
- ユーザーのデフォルトチケットポリシーの設定
- ユーザーの個別認証インジケーターチケットポリシーの設定
- **krbtpolicy-mod** コマンドの認証インジケーターオプション

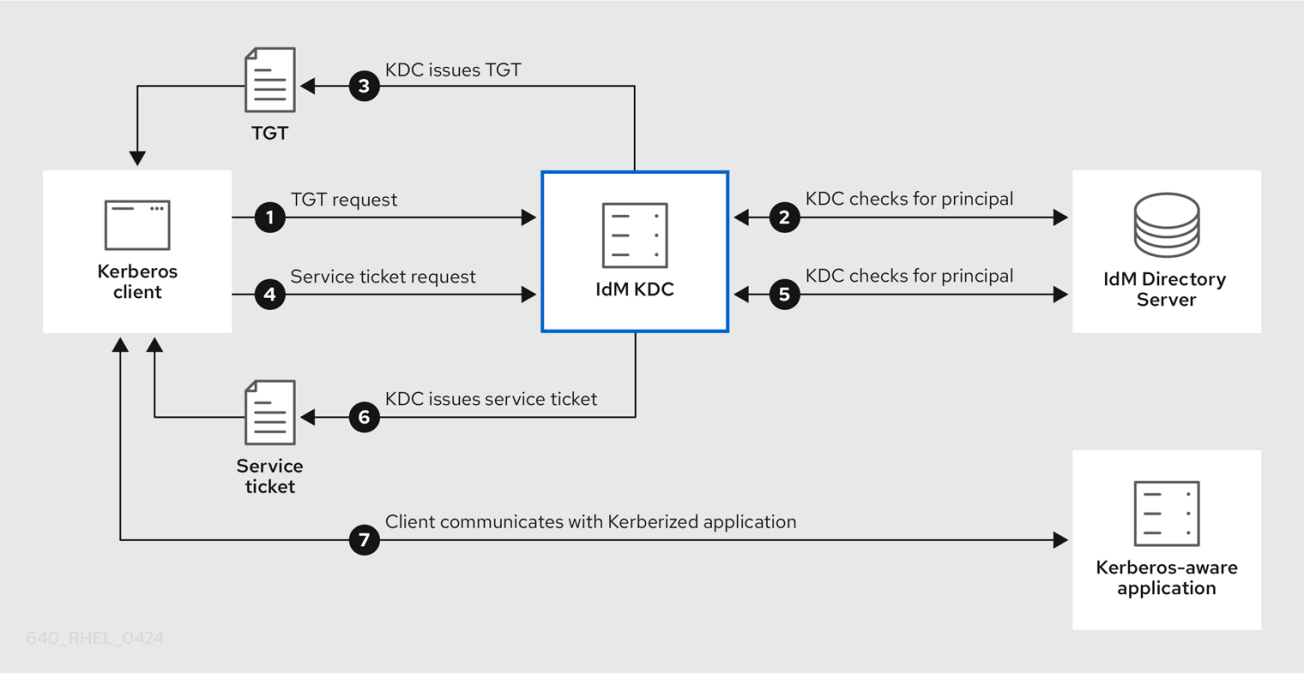
51.1. IDM KDC のロール

Identity Management の認証メカニズムは、Key Distribution Center (KDC) が設定する Kerberos インフラストラクチャーを使用します。KDC は、認証情報を保存し、IdM ネットワーク内のエンティティーから発信されるデータの信頼性を確保する信頼できる認証局です。

各 IdM ユーザー、サービス、およびホストは Kerberos クライアントとして機能し、一意の Kerberos **プリンシパル** で識別されます。

- ユーザーの場合: **identifier@REALM** (例: **admin@EXAMPLE.COM**)
- サービスの場合: **service/fully-qualified-hostname@REALM** (例: **http/server.example.com@EXAMPLE.COM**)
- ホストの場合: **host/fully-qualified-hostname@REALM** (例: **host/client.example.com@EXAMPLE.COM**)

以下の図では、Kerberos クライアント、KDC、およびクライアントの通信先となる Kerberos を使用するアプリケーション間の通信を簡単にまとめています。



1. Kerberos クライアントは、Kerberos プリンシパルとして認証することで KDC に対して身分を証明します。たとえば、IdM ユーザーは **kinit ユーザー名** を実行し、パスワードを指定します。

2. KDC はデータベースのプリンシパルを確認し、クライアントを認証し、Kerberos チケットポリシー を評価してリクエストを付与するかどうかを判断します。

3. KDC は、適切なチケットポリシーに従って、ライフサイクルおよび 認証インジケーター で Ticket-Granting Ticket (TGT) を発行します。

4. TGT により、クライアントは KDC から **サービスチケット** を要求し、ターゲットホストで Kerberos を使用するサービスと通信します。

5. KDC は、クライアントの TGT が有効であるかどうかを確認し、チケットポリシーに対してサービスチケット要求を評価します。

6. KDC はクライアントに **サービスチケット** を発行します。

7. サービスチケットを使用すると、クライアントはターゲットホストのサービスと暗号化された通信を開始できます。

51.2. IDM KERBEROS チケットポリシータイプ

IdM Kerberos チケットポリシーは、以下のチケットポリシータイプを実装します。

接続ポリシー

異なるレベルのセキュリティーで Kerberos を使用するサービスを保護するには、接続ポリシーを定義して、TGT (Ticket-granting ticket) の取得にクライアントが使用する事前認証メカニズムをもとにルールを有効にすることができます。
たとえば **client1.example.com** に接続するには、スマートカード認証が、**client2.example.com** の **testservice** アプリケーションにアクセスするには、2 要素認証が必要です。

接続ポリシーを有効するには、**認証インジケーター** をサービスに関連付けます。必要な認証インジケーターがサービスチケット要求に含まれるクライアントのみがこれらのサービスにアクセスできます。詳細は、Kerberos 認証インジケーター を参照してください。

チケットライフサイクルポリシー

各 Kerberos チケットには **有効期間** と潜在的な **更新期間** があります。チケットは最長期間に達する前に更新できますが、更新期間の超過後には更新できません。
デフォルトのグローバルチケットの有効期間は 1 日 (86400 秒) で、デフォルトのグローバル最大更新期間は 1 週間 (604800 秒) です。これらのグローバル値を調整するには、グローバルチケットライフサイクルポリシーの設定 を参照してください。

独自のチケットライフサイクルポリシーを定義することもできます。

- 各認証インジケーターに異なるグローバルチケットライフサイクル値を設定するには、認証インジケーターごとのグローバルチケットポリシーの設定 を参照してください。
- 使用する認証方法に関係なく適用する単一のユーザーのチケットライフサイクル値を定義するには、ユーザーのデフォルトチケットポリシーの設定 を参照してください。
- 単一ユーザーにのみ適用される認証インジケーター別のチケットライフサイクル値を定義するには、ユーザーの個別認証インジケーターチケットポリシーの設定 を参照してください。

51.3. KERBEROS 認証インジケーター

Kerberos Key Distribution Center (KDC) は、**認証インジケーター** を、ID の証明にクライアントが使用する事前認証メカニズムに基づいて TGT (Ticket-granting Ticket) に割り当てます。

otp

2 要素認証 (パスワード + ワンタイムパスワード)

radius

radius 認証 (通常は 802.1x 認証)

pkinit

PKINIT 、スマートカード、または証明書での認証

hardened

強化パスワード (SPAKE または FAST)[1]

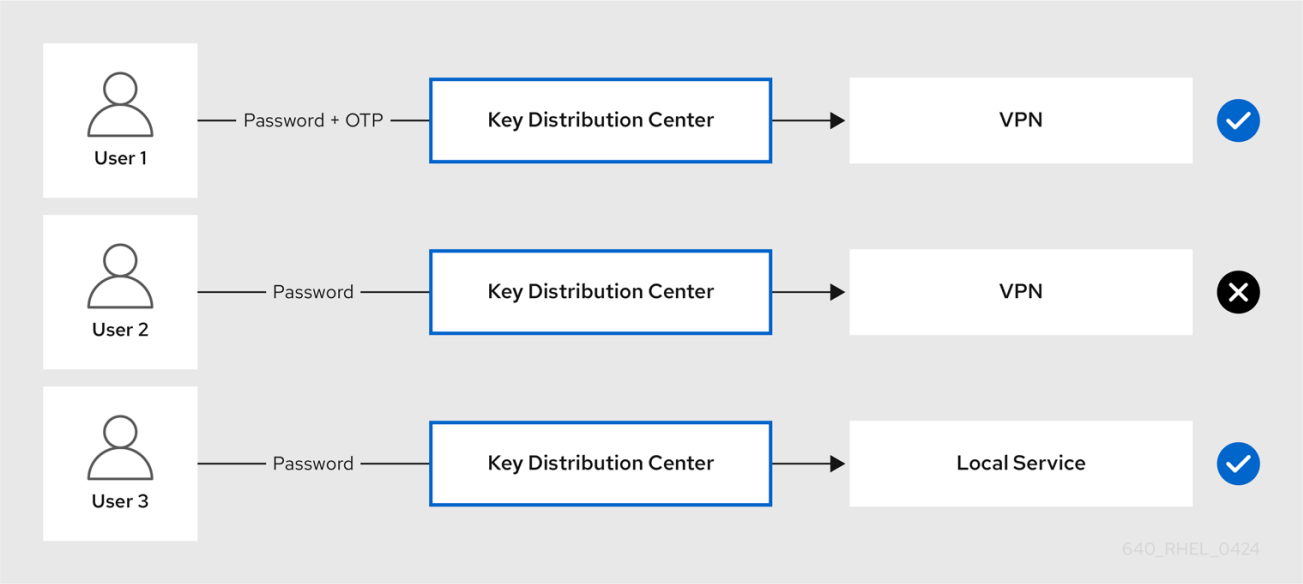
次に KDC は、TGT からの認証インジケーターを、TGT から取得するサービスチケット要求に割り当てます。KDC は、認証インジケーターに基づいて、サービスアクセス制御、チケットの最大有効期間、および更新可能な期間などのポリシーを有効にします。

認証インジケーターおよび IdM サービス

サービスまたはホストを認証インジケーターに関連付けると、対応する認証メカニズムを使用して TGT を取得したクライアントのみがアクセスできるようになります。KDC はアプリケーションやサービスではなく、サービスチケット要求の認証インジケーターをチェックし、Kerberos 接続ポリシーに基づいて要求を付与または拒否します。

たとえば、仮想プライベートネットワーク (VPN) に接続するために 2 要素認証を要求するには、**otp** 認証インジケーターをそのサービスに関連付けます。KDC から最初の TGT を取得するのにワンタイムパスワードを使用したユーザーのみが VPN にログインできます。

図51.1 otp 認証インジケータを必要とする VPN サービスの例



サービスまたはホストに認証インジケーターが割り当てられていない場合には、メカニズムに関係なく認証されたチケットを受け入れます。

関連情報

- IdM サービスの認証インジケーターの有効化
- IdM クライアントでの GSSAPI 認証の有効化および sudo の Kerberos 認証インジケーターの有効化

51.4. IDM サービスの認証インジケーターの有効化

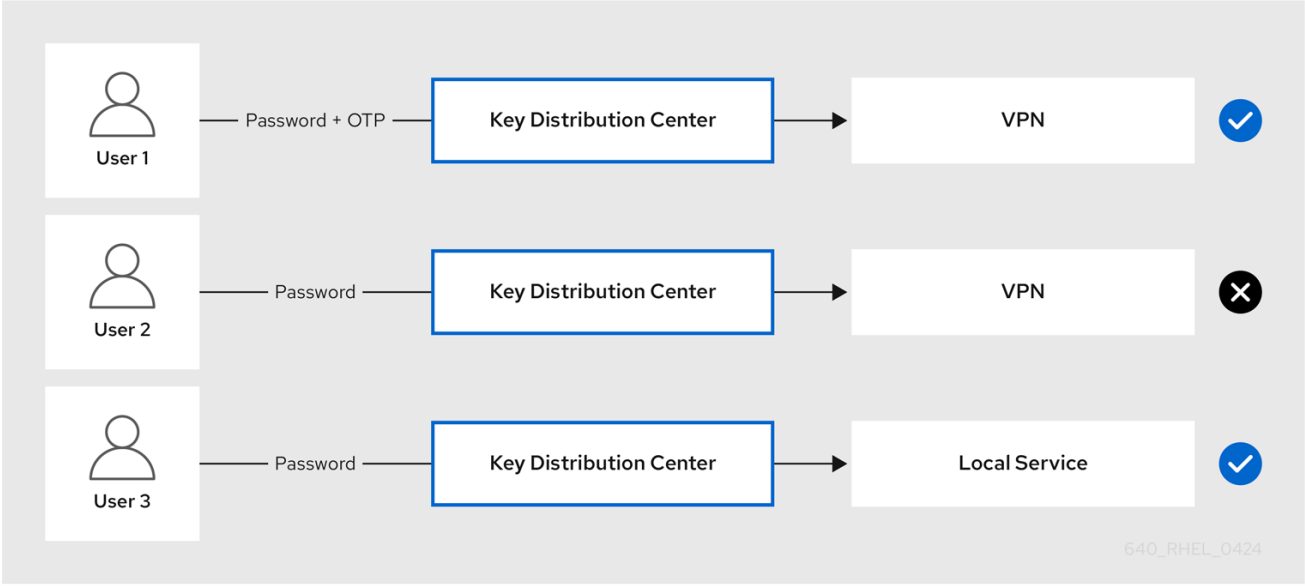
Identity Management (IdM) がサポートする認証メカニズムは、認証強度によって異なります。たとえば、標準パスワードと組み合わせて、ワンタイムパスワード (OTP) を使用して、最初の Kerberos Ticket-Granting Ticket (TGT) を取得することは、標準パスワードのみを使用する認証よりも安全と見なされます。

認証インジケーターを特定の IdM サービスに関連付けることで、IdM 管理者として、これらの特定の事前認証メカニズムを使用して最初の Ticket-Granting Ticket (TGT) を取得したユーザーのみがサービスにアクセスできるようにサービスを設定できます。

この方法では、以下のように異なる IdM サービスを設定できます。

- ワンタイムパスワード (OTP) などのより強力な認証方法を使用して最初の TGT を取得したユーザーのみが、VPN などのセキュリティーにとって重要なサービスにアクセスできます。
- パスワードなど、より簡単な認証方法を使用して初期の TGT を取得したユーザーは、ローカルログインなどの重要でないサービスにのみアクセスできます。

図51.2 異なる技術を使用した認証の例



以下の手順では、IdM サービスを作成し、着信サービスチケット要求から特定の Kerberos 認証インジケーターを必要とするように設定する方法を説明します。

51.4.1. IdM サービスエントリーおよびその Kerberos キータブの作成

IdM ホストで実行しているサービスの IdM に **IdM サービス** エントリーを追加すると、対応する Kerberos プリンシパルが作成され、サービスが SSL 証明書、Kerberos キータブ、またはその両方を要求できるようになります。

以下の手順では、IdM サービスエントリーを作成して、関連の Kerberos キータブを生成し、対象のサービスとの通信を暗号化する方法を説明します。

前提条件

- サービスが、Kerberos プリンシパル、SSL 証明書、またはその両方を保存できる。

手順

1. **ipa service-add** コマンドで IdM サービスを追加して、これに関連付けられた Kerberos プリンシパルを作成します。たとえば、ホスト **client.example.com** で実行する **testservice** アプリケーションの IdM サービスエントリーを作成するには、次のコマンドを実行します。

```
[root@client ~]# ipa service-add testservice/client.example.com
---------------------------------------------------------------
Modified service "testservice/client.example.com@EXAMPLE.COM"
---------------------------------------------------------------
  Principal name: testservice/client.example.com@EXAMPLE.COM
  Principal alias: testservice/client.example.com@EXAMPLE.COM
  Managed by: client.example.com
```

2. クライアント上でサービスの Kerberos キータブを生成し、保存します。

```
[root@client ~]# ipa-getkeytab -k /etc/testservice.keytab -p
testservice/client.example.com
Keytab successfully retrieved and stored in: /etc/testservice.keytab
```

検証手順

1. **ipa service-show** コマンドを使用して、IdM サービスに関する情報を表示します。

```
[root@server ~]# ipa service-show testservice/client.example.com
  Principal name: testservice/client.example.com@EXAMPLE.COM
  Principal alias: testservice/client.example.com@EXAMPLE.COM
  Keytab: True
  Managed by: client.example.com
```

2. **klist** コマンドを使用して、サービスの Kerberos キータブの内容を表示します。

```
[root@server etc]# klist -ekt /etc/testservice.keytab
Keytab name: FILE:/etc/testservice.keytab
KVNO Timestamp           Principal
---- ------------------- ------------------------------------------------------
   2 04/01/2020 17:52:55 testservice/client.example.com@EXAMPLE.COM (aes256-cts-
hmac-sha1-96)
   2 04/01/2020 17:52:55 testservice/client.example.com@EXAMPLE.COM (aes128-cts-
hmac-sha1-96)
   2 04/01/2020 17:52:55 testservice/client.example.com@EXAMPLE.COM (camellia128-cts-
cmac)
   2 04/01/2020 17:52:55 testservice/client.example.com@EXAMPLE.COM (camellia256-cts-
cmac)
```

51.4.2. IdM CLI を使用した IdM サービスへの認証インジケーターの関連付け

Identity Management (Id M) 管理者は、クライアントアプリケーションによって提示されるサービスチケットに特定の認証インジケーターが含まれていることを要求するようにホストまたはサービスを設定できます。たとえば、Kerberos Ticket-Granting Ticket (TGT) を取得する際に、パスワードで有効な IdM 2 要素認証トークンを使用したユーザーのみが、そのホストまたはサービスにアクセスできるようにできます。

受信サービスチケット要求からの特定の Kerberos 認証インジケーターを要求するようにサービスを設定するには、次の手順に従います。

前提条件

- IdM ホストで実行しているサービスの IdM サービスエントリーを作成している。IdM サービスエントリーおよびその Kerberos キータブの作成 を参照してください。
- IdM の管理ユーザーの Ticket-Granting Ticket (TGT) を取得している。

警告

内部 IdM サービスに認証インジケーターを割り当て **ない** でください。以下の IdM サービスでは、PKINIT およびマルチファクター認証方式で必要なインタラクティブな認証ステップを実行できません。

> **host**/server.example.com@EXAMPLE.COM
> **HTTP**/server.example.com@EXAMPLE.COM
> **ldap**/server.example.com@EXAMPLE.COM
> **DNS**/server.example.com@EXAMPLE.COM
> **cifs**/server.example.com@EXAMPLE.COM

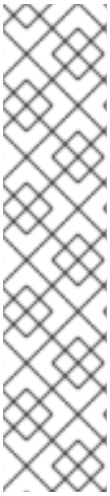
手順

- **ipa service-mod** コマンドを使用して、サービスに必要な認証インジケーターを **--auth-ind** 引数で識別して指定します。

認証方法	--auth-ind 値
2 要素認証	**otp**
radius 認証	**radius**
PKINIT 、スマートカード、または証明書での認証	**pkinit**
強化パスワード (SPAKE または FAST)	**hardened**

たとえば、スマートカードまたは OTP 認証で認証したユーザーには **client.example.com** ホストの **testservice** プリンシパルを取得させるようにするには、以下を実行します。

```
[root@server ~]# ipa service-mod testservice/client.example.com@EXAMPLE.COM --
auth-ind otp --auth-ind pkinit
----------------------------------------------------------
Modified service "testservice/client.example.com@EXAMPLE.COM"
----------------------------------------------------------
  Principal name: testservice/client.example.com@EXAMPLE.COM
  Principal alias: testservice/client.example.com@EXAMPLE.COM
  Authentication Indicators: otp, pkinit
  Managed by: client.example.com
```



注記

サービスからすべての認証インジケーターを削除するには、インジケーターの空のリストを指定します。

```
[root@server ~]# ipa service-mod
testservice/client.example.com@EXAMPLE.COM --auth-ind ''
-----------------------------------------------------
Modified service "testservice/client.example.com@EXAMPLE.COM"
-----------------------------------------------------
  Principal name: testservice/client.example.com@EXAMPLE.COM
  Principal alias: testservice/client.example.com@EXAMPLE.COM
  Managed by: client.example.com
```

検証手順

- **ipa service-show** コマンドを使用して、必要な認証インジケーターなど、IdM サービスに関する情報を表示します。

```
[root@server ~]# ipa service-show testservice/client.example.com
  Principal name: testservice/client.example.com@EXAMPLE.COM
  Principal alias: testservice/client.example.com@EXAMPLE.COM
  Authentication Indicators: otp, pkinit
  Keytab: True
  Managed by: client.example.com
```

関連情報

- IdM サービスの Kerberos サービスチケットの取得
- IdM クライアントでの GSSAPI 認証の有効化および sudo の Kerberos 認証インジケーターの有効化

51.4.3. IdM Web UI を使用した IdM サービスへの認証インジケーターの関連付け

Identity Management (IdM) 管理者は、ホストまたはサービスが、クライアントアプリケーションが提示するサービスチケットを特定の認証インジケーターを含むように設定できます。たとえば、Kerberos Ticket-Granting Ticket (TGT) を取得する際に、パスワードで有効な IdM 2 要素認証トークンを使用したユーザーのみが、そのホストまたはサービスにアクセスできるようにできます。

IdM Web UI を使用して、受信チケット要求からの特定の Kerberos 認証インジケーターを要求するようにホストまたはサービスを設定するには、次の手順に従います。

前提条件

- 管理ユーザーとして IdM Web UI にログインしている。

手順

1. **Identity → Hosts** または **Identity → Services** を選択します。
2. 必要なホストまたはサービスの名前をクリックします。
3. **Authentication indicators** で、必要な認証方法を選択します。

- たとえば、**OTP** を選択すると、Kerberos TGT を取得する際に、パスワードで有効な IdM 2 要素認証トークンを使用したユーザーのみが、ホストまたはサービスにアクセスできるようになります。
- **OTP** と **RADIUS** の両方を選択した場合は、Kerberos TGT を取得する際に有効な IdM 二要素認証トークンとパスワードを使用したユーザー **および** Kerberos TGT の取得に RADIUS サーバーを使用したユーザーの両方が、アクセスを許可されます。

4. ページ上部にある **Save** をクリックします。

関連情報

- IdM サービスの Kerberos サービスチケットの取得
- IdM クライアントでの GSSAPI 認証の有効化および sudo の Kerberos 認証インジケーターの有効化

51.4.4. IdM サービスの Kerberos サービスチケットの取得

以下の手順では、IdM サービスの Kerberos サービスチケットを取得する方法を説明します。この手順を使用して、特定の Kerberos 認証インジケーターが Ticket-Granting Ticket (TGT) に存在することを強制するなど、Kerberos チケットポリシーをテストできます。

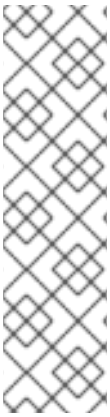
前提条件

- 対応する **IdM サービス** エントリーを作成している (使用しているサービスが内部 IdM サービスではない場合)。IdM サービスエントリーおよびその Kerberos キータブの作成 を参照してください。
- Kerberos Ticket-Granting Ticket (TGT) がある。

手順

- サービスチケットを取得するには、**kvno** コマンドに **-S** オプションを指定して、IdM サービスの名前と管理するホストの完全修飾ドメイン名を指定します。

```
[root@server ~]# kvno -S testservice client.example.com
testservice/client.example.com@EXAMPLE.COM: kvno = 1
```



注記

IdM サービスにアクセスする必要があり、現在の Ticket-Granting Ticket (TGT) に必要な Kerberos 認証インジケーターが関連付けられていない場合は、**kdestroy** コマンドで現在の Kerberos 認証情報キャッシュを消去し、新しい TGT を取得します。

```
[root@server ~]# kdestroy
```

たとえば、パスワードを使用して認証し、**pkinit** 認証インジケーターが関連付けられた IdM サービスにアクセスする必要がある場合は、現在の認証情報キャッシュを破棄し、スマートカードで再認証します。Kerberos 認証インジケーター を参照してください。

検証手順

- **klist** コマンドを使用して、サービスチケットがデフォルトの Kerberos 認証情報キャッシュにあることを確認します。

```
[root@server etc]# klist_
Ticket cache: KCM:1000
Default principal: admin@EXAMPLE.COM

Valid starting       Expires              Service principal
04/01/2020 12:52:42  04/02/2020 12:52:39  krbtgt/EXAMPLE.COM@EXAMPLE.COM
04/01/2020 12:54:07 04/02/2020 12:52:39
testservice/client.example.com@EXAMPLE.COM
```

51.4.5. 関連情報

- Kerberos 認証インジケーター を参照してください。

51.5. グローバルチケットライフサイクルポリシーの設定

グローバルチケットポリシーは、すべてのサービスチケットとユーザーごとのチケットポリシーが定義されていないユーザーに適用されます。

以下の手順では、**ipa krbtpolicy-mod** コマンドを使用して、グローバル Kerberos チケットポリシーのチケットの最大有効期間と最大更新期間を調整する方法を説明します。

ipa krbtpolicy-mod コマンドを使用する場合は、以下の引数のいずれかを指定します。

- **--maxlife** - チケットの最大有効期間 (秒単位)
- **--maxrenew** - 更新可能な最大期間 (秒単位)

手順

1. グローバルチケットポリシーを変更するには、以下を実行します。

```
[root@server ~]# ipa krbtpolicy-mod --maxlife=$((8*60*60)) --maxrenew=$((24*60*60))
  Max life: 28800
  Max renew: 86400
```

この例では、最大有効期間は 8 時間 (8 * 60 分 * 60 秒) に設定され、最大の更新可能期間は 1 日 (24 * 60 分 * 60 秒) に設定されています。

2. オプション: グローバルの Kerberos チケットポリシーをデフォルトのインストール値にリセットするには、以下を実行します。

```
[root@server ~]# ipa krbtpolicy-reset
  Max life: 86400
  Max renew: 604800
```

検証手順

- グローバルチケットポリシーを表示します。

```
[root@server ~]# ipa krbtpolicy-show
  Max life: 28800
  Max renew: 86640
```

関連情報

- ユーザーのデフォルトチケットポリシーの設定 を参照してください。
- ユーザーの個々の認証インジケーターチケットポリシーの設定 を参照してください。

51.6. 認証インジケーターごとのグローバルチケットポリシーの設定

各認証インジケーターのグローバルのチケット最大有効期間と最大更新可能期間を調整するには、次の手順に従います。この設定は、ユーザー別のチケットポリシーが定義されていないユーザーに適用されます。

ipa krbtpolicy-mod コマンドを使用して、それぞれに割り当てられた 認証インジケーター に合わせて、Kerberos チケットのグローバルの最大有効期間または更新可能な期間を指定します。

手順

- たとえば、グローバルな 2 要素のチケットの有効期間と更新期間の値を 1 週間に、グローバルスマートカードチケットの有効期間と更新期間の値を 2 週間に設定するには以下を実行します。

```
[root@server ~]# ipa krbtpolicy-mod --otp-maxlife=604800 --otp-maxrenew=604800 --
pkinit-maxlife=172800 --pkinit-maxrenew=172800
```

検証手順

- グローバルチケットポリシーを表示します。

```
[root@server ~]# ipa krbtpolicy-show
  Max life: 86400
  OTP max life: 604800
  PKINIT max life: 172800
  Max renew: 604800
  OTP max renew: 604800
  PKINIT max renew: 172800
```

OTP および PKINIT の値は、グローバルなデフォルトの **Max life** および **Max renew** 値とは異なることに注意してください。

関連情報

- **krbtpolicy-mod** コマンドの認証インジケーターオプション を参照してください。
- ユーザーのデフォルトチケットポリシーの設定 を参照してください。
- ユーザーの個々の認証インジケーターチケットポリシーの設定 を参照してください。

51.7. ユーザーのデフォルトチケットポリシーの設定

一意の Kerberos チケットポリシーを定義して、単一のユーザーだけに適用できます。これらのユーザーごとの設定は、すべての認証インジケーターに対してグローバルチケットポリシーをオーバーライドします。

ipa krbtpolicy-mod username コマンドを使用して、最低でも以下のいずれかの引数を指定します。

- **--maxlife** - チケットの最大有効期間 (秒単位)
- **--maxrenew** - 更新可能な最大期間 (秒単位)

手順

1. たとえば、IdM **管理者** ユーザーの最大チケット期間を 2 日に、最大更新期間を 2 週に設定するには、次のコマンドを実行します。

```
[root@server ~]# ipa krbtpolicy-mod admin --maxlife=172800 --maxrenew=1209600
  Max life: 172800
  Max renew: 1209600
```

2. オプション: ユーザーのチケットポリシーをリセットするには、以下を実行します。

```
[root@server ~]# ipa krbtpolicy-reset admin
```

検証手順

- ユーザーに適用される有効な Kerberos チケットポリシーを表示します。

```
[root@server ~]# ipa krbtpolicy-show admin
  Max life: 172800
  Max renew: 1209600
```

関連情報

- グローバルチケットライフサイクルポリシーの設定 を参照してください。
- 認証インジケーターごとのグローバルチケットポリシーの設定 を参照してください。

51.8. ユーザーの個別認証インジケーターチケットポリシーの設定

管理者は、ユーザーに、認証インジケーター別に異なる Kerberos チケットポリシーを定義できます。たとえば、IdM **管理者** ユーザーは、チケットを OTP 認証で取得した場合には 2 日間、スマートカード認証で取得した場合には 1 週間更新できるようにポリシーを設定できます。

これらの認証インジケーター設定は、**ユーザーの** デフォルトのチケットポリシー、**グローバル** デフォルトチケットポリシー、および **グローバル** 認証インジケーターチケットポリシーをオーバーライドします。

ipa krbtpolicy-mod username コマンドを使用して、それぞれに割り当てられた 認証インジケーター に合わせて、ユーザー Kerberos チケットのカスタム最大有効期間および最大の更新可能期間を設定します。

手順

1. たとえば、ワンタイムパスワード認証でチケットを取得した場合に IdM **admin** ユーザーが 2 日間 Kerberos チケットを更新できるようにするには、**--otp-maxrenew** オプションを設定します。

```
[root@server ~]# ipa krbtpolicy-mod admin --otp-maxrenew=$((2*24*60*60))
  OTP max renew: 172800
```

2. オプション: ユーザーのチケットポリシーをリセットするには、以下を実行します。

```
[root@server ~]# ipa krbtpolicy-reset username
```

検証手順

- ユーザーに適用される有効な Kerberos チケットポリシーを表示します。

```
[root@server ~]# ipa krbtpolicy-show admin
  Max life: 28800
  Max renew: 86640
```

関連情報

- **krbtpolicy-mod** コマンドの認証インジケーターオプション を参照してください。
- ユーザーのデフォルトチケットポリシーの設定 を参照してください。
- グローバルチケットライフサイクルポリシーの設定 を参照してください。
- 認証インジケーターごとのグローバルチケットポリシーの設定 を参照してください。

51.9. KRBTPOLICY-MOD コマンドの認証インジケーターオプション

以下の引数で認証インジケーターの値を指定します。

表51.1 krbtpolicy-mod コマンドの認証インジケーターオプション

認証インジケーター	最大有効期間の引数	最大更新期間の引数
otp	**--otp-maxlife**	**--otp-maxrenew**
radius	**--radius-maxlife**	**--radius-maxrenew**
pkinit	**--pkinit-maxlife**	**--pkinit-maxrenew**
hardened	**--hardened-maxlife**	**--hardened-maxrenew**

[1] 強化されたパスワードは、Single-Party Public-Key Authenticated Key Exchange (SPAKE) の事前認証または Flexible Authentication via Secure Tunneling (FAST) 防御を使用して、総当たりパスワード辞書攻撃を受けないようにセキュリティー保護されています。

第52章 IDM の KERBEROS PKINIT 認証

Kerberos(PKINIT) の初期認証の公開鍵暗号化は、Kerberos の事前認証メカニズムです。Identity Management (IdM) サーバーには、Kerberos PKINIT 認証のメカニズムが含まれています。

52.1. デフォルトの PKINIT 設定

IdM サーバーのデフォルトの PKINIT 設定は、認証局 (CA) 設定によって異なります。

表52.1 IdM のデフォルトの PKINIT 設定

CA 設定	PKINIT の設定
CA なし、外部 PKINIT 証明書の提供なし	ローカル PKINIT: IdM はサーバーの内部用途でのみ PKINIT を使用します。
CA なし、外部 PKINIT 証明書の IdM への提供あり	IdM は、外部の Kerberos 鍵配布センター (KDC) 証明書と CA 証明書を使用して PKINIT を設定します。
統合 CA あり	IdM は、IdM CA が署名した証明書を使用して PKINIT を設定します。

52.2. 現在の PKINIT 設定の表示

IdM には、ドメインの PKINIT 設定をクエリーするのに使用できるコマンドが複数用意されています。

手順

- ドメインの PKINIT のステータスを確認するには、**ipa pkinit-status** コマンドを使用します。

```
$ ipa pkinit-status
  Server name: server1.example.com
  PKINIT status: enabled
  [...output truncated...]
  Server name: server2.example.com
  PKINIT status: disabled
  [...output truncated...]
```

  このコマンドは、**enabled** または **disabled** として PKINIT 設定の状態を表示します。

  - **Enabled**: PKINIT は、統合 IdM CA または外部 PKINIT 証明書により署名された証明書を使用して設定されます。
  - **Disabled**: IdM は、IdM サーバーでの内部目的でのみ PKINIT を使用します。

- IdM クライアントの PKINIT に対応するアクティブな Kerberos 鍵配布センター (KDC) がある IdM サーバーをリスト表示するには、任意のサーバーで **ipa config-show** コマンドを使用します。

```
$ ipa config-show
  Maximum username length: 32
  Home directory base: /home
```

```
Default shell: /bin/sh
Default users group: ipausers
[...output truncated...]
IPA masters capable of PKINIT: server1.example.com
[...output truncated...]
```

52.3. IDM での PKINIT の設定

IdM サーバーが PKINIT を無効にした状態で動作している場合は、以下の手順に従って有効にします。たとえば、**--no-pkinit** オプションを **ipa-server-install** ユーティリティーまたは **ipa-replica-install** ユーティリティーで渡した場合には、サーバーは PKINIT が無効な状態で動作します。

前提条件

- 認証局 (CA) がインストールされているすべての IdM サーバーが、同じドメインレベルで稼働していることを確認します。

手順

1. PKINIT がサーバーで有効になっているかどうかを確認します。

```
# kinit admin

Password for admin@IDM.EXAMPLE.COM:
# ipa pkinit-status --server=server.idm.example.com
1 server matched
----------------
Server name: server.idm.example.com
PKINIT status:enabled
----------------------------
Number of entries returned 1
----------------------------
```

PKINIT が無効になっている場合は、次の出力が表示されます。

```
# ipa pkinit-status --server server.idm.example.com
-----------------
0 servers matched
-----------------
----------------------------
Number of entries returned 0
----------------------------
```

--server <server_fqdn> パラメーターを省略することで、このコマンドを使用して、PKINIT が有効になっているすべてのサーバーを検索することもできます。

2. CA なしで IdM を使用している場合は、以下の手順を実行します。

   a. IdM サーバーに、Kerberos Key Distribution Center (KDC) 証明書に署名した CA 証明書をインストールします。

```
# ipa-cacert-manage install -t CT,C,C ca.pem
```

b. すべての IPA ホストを更新するには、すべてのレプリカとクライアントで **ipa-certupdate** コマンドを繰り返し実行します。

```
# ipa-certupdate
```

c. **ipa-cacert-manage list** コマンドを使用して、CA 証明書がすでに追加されているかどうかを確認します。以下に例を示します。

```
# ipa-cacert-manage list
CN=CA,O=Example Organization
The ipa-cacert-manage command was successful
```

d. **ipa-server-certinstall** ユーティリティーを使用して、外部 KDC 証明書をインストールします。KDC 証明書は以下の条件を満たしている必要があります。

- コモンネーム **CN=fully_qualified_domain_name,certificate_subject_base** で発行されている。
- Kerberos プリンシパル **krbtgt/REALM_NAME@REALM_NAME** を含む。
- KDC 認証のオブジェクト識別子 (OID) **1.3.6.1.5.2.3.5** を含む。

```
# ipa-server-certinstall --kdc kdc.pem kdc.key

# systemctl restart krb5kdc.service
```

e. PKINIT のステータスを確認します。

```
# ipa pkinit-status
  Server name: server1.example.com
  PKINIT status: enabled
  [...output truncated...]
  Server name: server2.example.com
  PKINIT status: disabled
  [...output truncated...]
```

3. CA 証明書のある IdM を使用している場合は、次のように PKINIT を有効にします。

```
# ipa-pkinit-manage enable
  Configuring Kerberos KDC (krb5kdc)
  [1/1]: installing X509 Certificate for PKINIT
  Done configuring Kerberos KDC (krb5kdc).
  The ipa-pkinit-manage command was successful
```

IdM CA を使用している場合、コマンドは CA から PKINIT KDC 証明書を要求します。

関連情報

- **ipa-server-certinstall(1)** の man ページ

52.4. 関連情報

- Kerberos PKINIT の詳細は、MIT Kerberos ドキュメントの PKINIT 設定

第53章 IDM KERBEROS キータブファイルの維持

Kerberos キータブファイルとは何か、および Identity Management (IdM) がそれを使用してサービスが Kerberos で安全に認証できるようにする方法について詳しく説明します。

この情報を使用して、これらの機密ファイルを保護する必要がある理由を理解し、IdM サービス間の通信の問題をトラブルシューティングできます。

詳細は、以下のトピックを参照してください。

- Identity Management が Kerberos キータブファイルを使用する方法
- Kerberos キータブファイルが IdM データベースと同期していることの確認
- IdM Kerberos キータブファイルとその内容のリスト
- IdM マスターキーの暗号化タイプの表示

53.1. IDENTITY MANAGEMENT が KERBEROS キータブファイルを使用する方法

Kerberos キータブは、Kerberos プリンシパルとそれに対応する暗号化キーを含むファイルです。ホスト、サービス、ユーザー、およびスクリプトは、キータブを使用して、人間の介入を必要とせずに、Kerberos Key Distribution Center (KDC) に対して安全に認証することができます。

IdM サーバー上のすべての IdM サービスには、Kerberos データベースに格納された一意の Kerberos プリンシパルがあります。たとえば、IdM サーバー **east.idm.example.com** および **west.idm.example.com** が DNS サービスを提供する場合、IdM はこれらのサービスを識別するために 2 つの一意の DNS Kerberos プリンシパルを作成します。これらは、命名規則 **<service>/host.domain.com@REALM.COM** に従います:

- **DNS/east.idm.example.com@IDM.EXAMPLE.COM**
- **DNS/west.idm.example.com@IDM.EXAMPLE.COM**

IdM は、Kerberos キーのローカルコピーをキーバージョン番号 (KVNO) とともに保存するために、これらのサービスごとにサーバー上にキータブを作成します。たとえば、デフォルトのキータブファイル **/etc/krb5.keytab** には **host** プリンシパルが格納されます。このプリンシパルは、Kerberos レルム内のそのマシンを表し、ログイン認証に使用されます。KDC は、**aes256-cts-hmac-sha1-96** および **aes128-cts-hmac-sha1-96** など、サポートするさまざまな暗号化アルゴリズムの暗号化キーを生成します。

klist コマンドを使用して、キータブファイルの内容を表示できます。

```
[root@idmserver ~]# klist -ekt /etc/krb5.keytab
Keytab name: FILE:/etc/krb5.keytab
KVNO Timestamp           Principal
---- ------------------- ------------------------------------------------------
   2 02/24/2022 20:28:09 host/idmserver.idm.example.com@IDM.EXAMPLE.COM (aes256-cts-hmac-sha1-96)
   2 02/24/2022 20:28:09 host/idmserver.idm.example.com@IDM.EXAMPLE.COM (aes128-cts-hmac-sha1-96)
   2 02/24/2022 20:28:09 host/idmserver.idm.example.com@IDM.EXAMPLE.COM (camellia128-cts-
```

```
cmac)
   2 02/24/2022 20:28:09 host/idmserver.idm.example.com@IDM.EXAMPLE.COM (camellia256-cts-
cmac)
```

関連情報

- Kerberos キータブファイルが IdM データベースと同期していることの確認
- IdM Kerberos キータブファイルとその内容のリスト

53.2. KERBEROS キータブファイルが IDM データベースと同期していることの確認

Kerberos パスワードを変更すると、IdM は対応する新しい Kerberos キーを自動的に生成し、そのキーバージョン番号 (KVNO) を増やします。Kerberos キータブが新しいキーと KVNO で更新されていない場合、そのキータブに依存して有効なキーを取得するサービスは、Kerberos キー配布センター (KDC) に対して認証できない可能性があります。

IdM サービスの 1 つが別のサービスと通信できない場合は、次の手順を使用して、Kerberos キータブファイルが IdM データベースに保存されているキーと同期していることを確認します。それらが同期していない場合は、更新されたキーと KVNO を使用して Kerberos キータブを取得します。この例では、IdM サーバーの更新された **DNS** プリンシパルを比較して取得します。

前提条件

- キータブファイルを取得するには、IdM 管理者アカウントとして認証する必要がある。
- 他のユーザーが所有するキータブファイルを変更するには、**root** アカウントとして認証する必要がある。

手順

1. 検証しているキータブでプリンシパルの KVNO を表示します。次の例では、**/etc/named.keytab** ファイルに、KVNO が 2 の **DNS/server1.idm.example.com@EXAMPLE.COM** プリンシパルのキーがあります。

```
[root@server1 ~]# klist -ekt /etc/named.keytab
Keytab name: FILE:/etc/named.keytab
KVNO Timestamp           Principal
---- ------------------- ------------------------------------------------------
   2 11/26/2021 13:51:11 DNS/server1.idm.example.com@EXAMPLE.COM (aes256-cts-
hmac-sha1-96)
   2 11/26/2021 13:51:11 DNS/server1.idm.example.com@EXAMPLE.COM (aes128-cts-
hmac-sha1-96)
   2 11/26/2021 13:51:11 DNS/server1.idm.example.com@EXAMPLE.COM (camellia128-cts-
cmac)
   2 11/26/2021 13:51:11 DNS/server1.idm.example.com@EXAMPLE.COM (camellia256-cts-
cmac)
```

2. IdM データベースに保存されているプリンシパルの KVNO を表示します。この例では、IdM データベースのキーの KVNO がキータブの KVNO と一致しません。

```
[root@server1 ~]# kvno DNS/server1.idm.example.com@EXAMPLE.COM
DNS/server1.idm.example.com@EXAMPLE.COM: kvno = 3
```

3. IdM 管理者アカウントとして認証します。

```
[root@server1 ~]# kinit admin
Password for admin@IDM.EXAMPLE.COM:
```

4. プリンシパルの更新された Kerberos キーを取得し、キータブに保存します。**root** ユーザーとしてこのステップを実行して、**named** ユーザーが所有する **/etc/named.keytab** ファイルを変更できるようにします。

```
[root@server1 ~]# ipa-getkeytab -s server1.idm.example.com -p
DNS/server1.idm.example.com -k /etc/named.keytab
```

検証

1. プリンシパルの更新された KVNO をキータブに表示します。

```
[root@server1 ~]# klist -ekt /etc/named.keytab
Keytab name: FILE:/etc/named.keytab
KVNO Timestamp           Principal
---- ------------------- ------------------------------------------------------
   4 08/17/2022 14:42:11 DNS/server1.idm.example.com@EXAMPLE.COM (aes256-cts-
hmac-sha1-96)
   4 08/17/2022 14:42:11 DNS/server1.idm.example.com@EXAMPLE.COM (aes128-cts-
hmac-sha1-96)
   4 08/17/2022 14:42:11 DNS/server1.idm.example.com@EXAMPLE.COM (camellia128-cts-
cmac)
   4 08/17/2022 14:42:11 DNS/server1.idm.example.com@EXAMPLE.COM (camellia256-cts-
cmac)
```

2. IdM データベースに保存されているプリンシパルの KVNO を表示し、キータブの KVNO と一致することを確認します。

```
[root@server1 ~]# kvno DNS/server1.idm.example.com@EXAMPLE.COM
DNS/server1.idm.example.com@EXAMPLE.COM: kvno = 4
```

関連情報

- Identity Management が Kerberos キータブファイルを使用する方法
- IdM Kerberos キータブファイルとその内容のリスト

53.3. IDM KERBEROS キータブファイルとその内容のリスト

以下の表は、IdM Kerberos キータブファイルの場所、内容、および目的を示しています。

表53.1テーブル

キータブの場所	内容	目的

キータブの場所	内容	目的
/etc/krb5.keytab	**host** プリンシパル	ログイン時にユーザーの認証情報を確認します (**nfs** プリンシパルがない場合に NFS によって使用されます)。
/etc/dirsrv/ds.keytab	**ldap** プリンシパル	IdM データベースに対してユーザーを認証し、IdM レプリカ間でデータベースの内容を安全に複製します。
/var/lib/ipa/gssproxy/http.keytab	**HTTP** プリンシパル	Apache サーバーに対して認証します。
/etc/named.keytab	**DNS** プリンシパル	DNS レコードを安全に更新します。
/etc/ipa/dnssec/ipa-dnskeysyncd.keytab	**ipa-dnskeysyncd** プリンシパル	OpenDNSSEC と LDAP の同期を維持します。
/etc/pki/pki-tomcat/dogtag.keytab	**dogtag** プリンシパル	認証局 (CA) と通信します。
/etc/samba/samba.keytab	**cifs** プリンシパルと **host** プリンシパル	Samba サービスと通信します。
/var/lib/sss/keytabs/ad-domain.com.keytab	**HOSTNAME$@AD-DOMAIN.COM** 形式の Active Directory (AD) ドメインコントローラー (DC) プリンシパル	IdM-AD 信頼を介して AD DC と通信します。

関連情報

- Identity Management が Kerberos キータブファイルを使用する方法
- Kerberos キータブファイルが IdM データベースと同期していることの確認

53.4. IDM マスターキーの暗号化タイプの表示

Identity Management (IdM) 管理者は、IdM マスターキーの暗号化タイプを表示できます。これは、IdM Kerberos Distribution Center (KDC) が保存時に他のすべてのプリンシパルを暗号化するために使用するキーです。暗号化の種類を知ることで、デプロイメントの FIPS 標準との互換性を判断するのに役立ちます。

RHEL 8.7 の時点で、暗号化タイプは **aes256-cts-hmac-sha384-192** です。この暗号化タイプは、FIPS 140-3 への準拠を目的としたデフォルトの RHEL 9 FIPS 暗号化ポリシーと互換性があります。

以前の RHEL バージョンで使用されていた暗号化タイプは、FIPS 140-3 標準に準拠する RHEL 9 システムと互換性がありません。FIPS モードの RHEL 9 システムを RHEL 8 FIPS 140-2 デプロイメントと互換性を持たせるには、RHEL 9 システムで **FIPS:AD-SUPPORT** 暗号化ポリシーを有効にします。

注記

Microsoft の Active Directory 実装は、SHA-2 HMAC を使用する RFC8009 Kerberos 暗号化タイプをまだサポートしていません。したがって、IdM-AD 信頼が設定されている場合、IdM マスターキーの暗号化タイプが **aes256-cts-hmac-sha384-192** であっても、FIPS:AD-SUPPORT 暗号サブポリシーの使用が必要になります。

前提条件

- IdM デプロイメント内の RHEL 8 レプリカのいずれかに **root** アクセス権がある。

手順

- レプリカで、コマンドラインインターフェイスで暗号化の種類を表示します。

```
# kadmin.local getprinc K/M | grep -E '^Key:'
Key: vno 1, aes256-cts-hmac-sha1-96
```

出力の **aes256-cts-hmac-sha1-96** キーは、IdM デプロイメントが RHEL 8.6 以前を実行しているサーバーにインストールされたことを示しています。出力に **aes256-cts-hmac-sha384-192** キーが存在する場合、IdM デプロイメントが RHEL 8.7 以降を実行しているサーバーにインストールされたことを示します。

第54章 IDM での KDC プロキシーの使用

一部の管理者は、デプロイメントでデフォルトの Kerberos ポートにアクセスできないようにすることを選択する場合があります。ユーザー、ホスト、およびサービスが Kerberos 認証情報を取得できるようにするために、**HTTPS** ポート 443 を介して Kerberos と通信するプロキシーとして **HTTPS** サービスを使用できます。

Identity Management (IdM) では、**Kerberos Key Distribution Center Proxy**(KKDCP) がこの機能を提供します。

IdM サーバーでは、KKDCP はデフォルトで有効で、**https://server.idm.example.com/KdcProxy** で利用できます。IdM クライアントでは、KDCP にアクセスするために、その Kerberos 設定を変更する必要があります。

54.1. KKDCP を使用するための IDM クライアントの設定

Identity Management (IdM) システム管理者は、IdM サーバーで Kerberos Key Distribution Center Proxy (KKDCP) を使用するように IdM クライアントを設定できます。これは、デフォルトの Kerberos ポートが IdM サーバーでアクセスできず、**HTTPS** ポート 443 が Kerberos サービスにアクセスする唯一の方法である場合に役立ちます。

前提条件

- IdM クライアントへの **root** アクセス権限がある。

手順

1. **/etc/krb5.conf** ファイルを開いて編集します。
2. **[realms]** セクションで、**kdc**、**admin_server**、および **kpasswd_server** オプションの KKDCP の URL を入力します。

```
[realms]
EXAMPLE.COM = {
  kdc = https://kdc.example.com/KdcProxy
  admin_server = https://kdc.example.com/KdcProxy
  kpasswd_server = https://kdc.example.com/KdcProxy
  default_domain = example.com
}
```

   冗長性の場合は、パラメーター **kdc**、**admin_server**、および **kpasswd_server** を複数回追加して、別の KDCP サーバーを示すことができます。

3. **sssd** を再起動して、変更を有効にします。

```
~]# systemctl restart sssd
```

54.2. IDM サーバーで KKDCP が有効になっていることの確認

Identity Management (Id M) サーバーでは、属性と値のペア **ipaConfigString=kdcProxyEnabled** がディレクトリーに存在する場合、Apache Web サーバーが起動するたびに Kerberos Key Distribution Center Proxy (KKDCP) が自動的に有効になります。この場合は、シンボリックリンク **/etc/httpd/conf.d/ipa-kdc-proxy.conf** が作成されます。

非特権ユーザーであっても、IdM サーバーで KKDCP が有効になっているかどうかを確認できます。

手順

- シンボリックリンクが存在することを確認します。

```
$ ls -l /etc/httpd/conf.d/ipa-kdc-proxy.conf
lrwxrwxrwx. 1 root root 36 Jun 21  2020 /etc/httpd/conf.d/ipa-kdc-proxy.conf -> /etc/ipa/kdcproxy/ipa-kdc-proxy.conf
```

この出力は、KKDCP が有効になっていることを確認します。

54.3. IDM サーバーでの KDCP の無効化

Identity Management (IdM) システム管理者は、IdM サーバーで Kerberos Key Distribution Center Proxy (KKDCP) を無効にできます。

前提条件

- IdM サーバーへの **root** アクセス権限がある。

手順

1. ディレクトリーから **ipaConfigString=kdcProxyEnabled** 属性と値のペアを削除します。

```
# ipa-ldap-updater /usr/share/ipa/kdcproxy-disable.uldif
Update complete
The ipa-ldap-updater command was successful
```

2. **httpd** サービスを再起動します。

```
# systemctl restart httpd.service
```

現在の IdM サーバーで、KDCP が無効になりました。

検証手順

- シンボリックリンクが存在しないことを確認します。

```
$ ls -l /etc/httpd/conf.d/ipa-kdc-proxy.conf
ls: cannot access '/etc/httpd/conf.d/ipa-kdc-proxy.conf': No such file or directory
```

54.4. IDM サーバーでの KDCP の再有効化

IdM サーバーでは、Kerberos Key Distribution Center Proxy (KKDCP) はデフォルトで有効で、**https://server.idm.example.com/KdcProxy** で利用できます。

KDCP がサーバーで無効になっている場合は、再度有効にできます。

前提条件

- IdM サーバーへの **root** アクセス権限がある。

手順

1. **ipaConfigString=kdcProxyEnabled** 属性と値のペアをディレクトリーに追加します。

```
# ipa-ldap-updater /usr/share/ipa/kdcproxy-enable.uldif
Update complete
The ipa-ldap-updater command was successful
```

2. **httpd** サービスを再起動します。

```
# systemctl restart httpd.service
```

現在の IdM サーバーで KDCP が有効になりました。

検証手順

- シンボリックリンクが存在することを確認します。

```
$ ls -l /etc/httpd/conf.d/ipa-kdc-proxy.conf
lrwxrwxrwx. 1 root root 36 Jun 21  2020 /etc/httpd/conf.d/ipa-kdc-proxy.conf ->
/etc/ipa/kdcproxy/ipa-kdc-proxy.conf
```

54.5. KKDCP サーバー I の設定

以下の設定により、複数の Kerberos サーバーが使用される IdM KKDCP と Active Directory (AD) レルム間のトランスポートプロトコルとして TCP を使用できるようになります。

前提条件

- **root** アクセスがある。

手順

1. **/etc/ipa/kdcproxy/kdcproxy.conf** ファイルの **[global]** セクションにある **use_dns** パラメーターを **false** に設定します。

```
[global]
use_dns = false
```

2. プロキシーされたレルム情報を **/etc/ipa/kdcproxy/kdcproxy.conf** ファイルに入れます。たとえば、プロキシーを使用する [AD.**EXAMPLE.COM**] レルムの場合、次のようにレルム設定パラメーターをリスト表示します。

```
[AD.EXAMPLE.COM]
kerberos = kerberos+tcp://1.2.3.4:88 kerberos+tcp://5.6.7.8:88
kpasswd = kpasswd+tcp://1.2.3.4:464 kpasswd+tcp://5.6.7.8:464
```

重要

特定のオプションが複数回指定される可能性がある **/etc/krb5.conf** および **kdc.conf** とは対照的に、レルム設定パラメーターは、スペースで区切られた複数のサーバーをリストする必要があります。

3. Identity Management (IdM) サービスを再起動します。

```
# ipactl restart
```

関連情報

- Red Hat ナレッジベースの Configure IPA server as a KDC Proxy for AD Kerberos communication を参照してください。

54.6. KKDCP サーバー II の設定

次のサーバー設定は、DNS サービスレコードに依存して、通信する Active Directory (AD) サーバーを検索します。

前提条件

- **root** アクセスがある。

手順

1. **/etc/ipa/kdcproxy/kdcproxy.conf** ファイルの **global** セクションで、**use_dns** パラメーターを **true** に設定します。

```
[global]
configs = mit
use_dns = true
```

configs パラメーターを使用すると、他の設定モジュールをロードできます。この場合、設定は MIT **libkrb5** ライブラリーから読み取られます。

2. **オプション**: DNS サービスレコードを使用したくない場合は、明示的な AD サーバーを **/etc/krb5.conf** ファイルの **[realms]** セクションに追加します。たとえば、プロキシーを使用するレルムが AD.**EXAMPLE.COM** の場合は、以下を追加します。

```
[realms]
AD.EXAMPLE.COM = {
    kdc = ad-server.ad.example.com
    kpasswd_server = ad-server.ad.example.com
}
```

3. Identity Management (IdM) サービスを再起動します。

```
# ipactl restart
```

関連情報

- Red Hat ナレッジベースの Configure IPA server as a KDC Proxy for AD Kerberos communication を参照してください。

第55章 IDM クライアントの IDM ユーザーへの SUDO アクセスの許可

Identity Management でユーザーに **sudo** アクセス権を付与する方法を詳しく説明します。

55.1. IDM クライアントの SUDO アクセス

システム管理者は、root 以外のユーザーに、通常 **root** ユーザー用に予約されている管理コマンドを実行できるようにする **sudo** アクセスを付与できます。その結果、ユーザーが、通常、**root** ユーザー用に予約される管理コマンドを実行する場合は、コマンドの前に **sudo** を付けることができます。パスワードを入力すると、そのコマンドは **root** ユーザーとして実行されます。データベースサービスアカウントなどの別のユーザーまたはグループとして **sudo** コマンドを実行するには、**sudo** ルールの **RunAs エイリアス** を設定できます。

Red Hat Enterprise Linux (RHEL) 8 ホストが Identity Management (IdM) クライアントとして登録されている場合は、以下の方法で、どの IdM ユーザーがホストでどのコマンドを実行できるかを定義する **sudo** ルールを指定できます。

- ローカルの **/etc/sudoers** ファイル
- IdM での一元設定

コマンドラインインターフェイス (CLI) と IdM Web UI を使用して、IdM クライアントの **sudo 集約ルール** を作成できます。

RHEL 8.4 以降では、Generic Security Service Application Programming Interface (GSSAPI) を使用して **sudo** のパスワードレス認証を設定することもできます。これは、UNIX ベースのオペレーティングシステムがネイティブで Kerberos サービスにアクセスして認証する方法です。**pam_sss_gss.so** Pluggable Authentication Module (PAM) を使用して SSSD サービスを介して GSSAPI 認証を呼び出し、有効な Kerberos チケットを使用して **sudo** コマンドに対して認証を行うことができます。

関連情報

- sudo アクセスの管理 を参照してください。

55.2. CLI での IDM クライアントの IDM ユーザーへの SUDO アクセス許可

Identity Management (IdM) では、特定の IdM ホストで IdM ユーザーアカウントの特定コマンドに **sudo** アクセスを付与できます。最初に **sudo** コマンドを追加してから、1 つまたは複数のコマンドに対して **sudo** ルールを作成します。

たとえば、**idmclient** マシンで **/usr/sbin/reboot** コマンドを実行する権限を **idm_user** に付与する **idm_user_reboot** の **sudo** ルールを作成するには、以下の手順を実行します。

前提条件

- IdM 管理者としてログインしている。
- IdM で **idm_user** のユーザーアカウントを作成し、ユーザーのパスワードを作成してそのアカウントのロックを解除している。CLI を使用して新しい IdM ユーザーを追加する方法の詳細は、コマンドラインを使用したユーザーの追加 を参照してください。
- **idmclient** ホストにローカル **idm_user** アカウントが存在しない。**idm_user** ユーザーは、ローカルの **/etc/passwd** ファイルには表示されません。

手順

1. IdM の **管理者** として Kerberos チケットを取得します。

```
[root@idmclient ~]# kinit admin
```

2. **sudo** コマンドの IdM データベースに **/usr/sbin/reboot** コマンドを追加します。

```
[root@idmclient ~]# ipa sudocmd-add /usr/sbin/reboot
---------------------------------------
Added Sudo Command "/usr/sbin/reboot"
---------------------------------------
  Sudo Command: /usr/sbin/reboot
```

3. **idm_user_reboot** という名前の **sudo** ルールを作成します。

```
[root@idmclient ~]# ipa sudorule-add idm_user_reboot
--------------------------------
Added Sudo Rule "idm_user_reboot"
--------------------------------
  Rule name: idm_user_reboot
  Enabled: TRUE
```

4. **/usr/sbin/reboot** コマンドを **idm_user_reboot** ルールに追加します。

```
[root@idmclient ~]# ipa sudorule-add-allow-command idm_user_reboot --sudocmds
'/usr/sbin/reboot'
  Rule name: idm_user_reboot
  Enabled: TRUE
  Sudo Allow Commands: /usr/sbin/reboot
-------------------------
Number of members added 1
-------------------------
```

5. **idm_user_reboot** ルールを IdM **idmclient** ホストに適用します。

```
[root@idmclient ~]# ipa sudorule-add-host idm_user_reboot --hosts
idmclient.idm.example.com
Rule name: idm_user_reboot
Enabled: TRUE
Hosts: idmclient.idm.example.com
Sudo Allow Commands: /usr/sbin/reboot
-------------------------
Number of members added 1
-------------------------
```

6. **idm_user** アカウントを **idm_ user_reboot** ルールに追加します。

```
[root@idmclient ~]# ipa sudorule-add-user idm_user_reboot --users idm_user
Rule name: idm_user_reboot
Enabled: TRUE
Users: idm_user
Hosts: idmclient.idm.example.com
Sudo Allow Commands: /usr/sbin/reboot
```

```
-------------------------
Number of members added 1
-------------------------
```

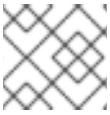
7. 必要に応じて、**idm_user_reboot** ルールの有効性を定義します。

   a. **sudo** ルールが有効である時間を定義するには、**--setattr sudonotbefore=DATE** オプションを指定して **ipa sudorule-mod sudo_rule_name** コマンドを使用します。**DATE** 値は、**yyyymmddHHMMSSZ** 形式に準拠し、明示的に指定される秒数である必要があります。たとえば、**idm_user_reboot** ルールの有効性の開始を 2025 12:34:00 年 12 月 31 に設定するには、次のコマンドを実行します。

   ```
   [root@idmclient ~]# ipa sudorule-mod idm_user_reboot --setattr
   sudonotbefore=20251231123400Z
   ```

   b. sudo ルールが有効な停止時間を定義するには、**--setattr sudonotafter=DATE** オプションを使用します。たとえば、**idm_user_reboot** ルールの有効期間の最後を 2026 12:34:00 年 12 月 31 に設定するには、次のコマンドを実行します。

   ```
   [root@idmclient ~]# ipa sudorule-mod idm_user_reboot --setattr
   sudonotafter=20261231123400Z
   ```



注記

サーバーからクライアントへの変更の伝播には数分かかる場合があります。

検証手順

1. **idmclient** ホストに **idm_user** アカウントとしてログインします。
2. **idm_user** アカウントが実行可能な **sudo** ルールを表示します。

```
[idm_user@idmclient ~]$ sudo -l
Matching Defaults entries for idm_user on idmclient:
    !visiblepw, always_set_home, match_group_by_gid, always_query_group_plugin,
    env_reset, env_keep="COLORS DISPLAY HOSTNAME HISTSIZE KDEDIR
LS_COLORS",
    env_keep+="MAIL PS1 PS2 QTDIR USERNAME LANG LC_ADDRESS LC_CTYPE",
    env_keep+="LC_COLLATE LC_IDENTIFICATION LC_MEASUREMENT
LC_MESSAGES",
    env_keep+="LC_MONETARY LC_NAME LC_NUMERIC LC_PAPER LC_TELEPHONE",
    env_keep+="LC_TIME LC_ALL LANGUAGE LINGUAS _XKB_CHARSET XAUTHORITY
KRB5CCNAME",
    secure_path=/sbin\:/bin\:/usr/sbin\:/usr/bin

User idm_user may run the following commands on idmclient:
    (root) /usr/sbin/reboot
```

3. **sudo** を使用してマシンを再起動します。プロンプトが表示されたら、**idm_user** のパスワードを入力します。

```
[idm_user@idmclient ~]$ sudo /usr/sbin/reboot
[sudo] password for idm_user:
```

55.3. AD での IDM クライアントの IDM ユーザーへの SUDO アクセス許可

Identity Management (IdM) システム管理者は、IdM ユーザーグループを使用して、アクセス許可、ホストベースのアクセス制御、**sudo** ルール、および IdM ユーザーに対するその他の制御を設定できます。IdM ユーザーグループは、IdM ドメインリソースへのアクセスを許可および制限します。

Active Directory (AD) **ユーザー** と AD **グループ** の両方を IdM ユーザーグループに追加できます。これを実行するには、以下を行います。

1. AD ユーザーまたはグループを **非 POSIX** 外部 IdM グループに追加します。
2. 非 POSIX 外部 IdM グループを IdM **POSIX** グループに追加します。

その後、POSIX グループの権限を管理することで、AD ユーザーの権限を管理できます。例えば、特定のコマンドの **sudo** アクセスを、特定の IdM ホストの IdM POSIX ユーザーグループに付与できます。

注記

AD ユーザーグループを、IdM 外部グループにメンバーとして追加することもできます。これにより、1 つの AD レルムにユーザーおよびグループの管理を維持することで、Windows ユーザーのポリシーの定義が容易になります。

重要

IdM の SUDO ルールに AD ユーザーの ID オーバーライドを使用 **しない** でください。AD ユーザーの ID オーバーライドは、AD ユーザー自体ではなく、AD ユーザーの POSIX 属性のみを表します。

ID オーバーライドをグループメンバーとして追加できます。ただし、この機能は IdM API で IdM リソースを管理するためにのみ使用できます。グループメンバーとして ID オーバーライドを追加する可能性は POSIX 環境に拡張されていないため、**sudo** またはホストベースのアクセス制御 (HBAC) ルールのメンバーシップには使用できません。

この手順では、**ad_users_reboot sudo** ルールを作成して、**administrator@ad-domain.com** AD ユーザーに、**idmclient** IdM ホストで **/usr/sbin/reboot** コマンドを実行するパーミッションを付与します。これは通常、**root** ユーザー用に予約されています。**administrator@ad-domain.com** は **ad_users_external** 非 POSIX グループのメンバーであり、これは **ad_users** POSIX グループのメンバーでもあります。

前提条件

- IdM **admin** Kerberos の チケット許可チケット (TGT) を取得しました。
- IdM ドメインと **ad-domain.com** AD ドメインの間にフォレスト間の信頼が存在します。
- **idmclient** ホストにローカル **管理者** アカウントが存在しません。**管理者** ユーザーがローカルの **/etc/passwd** ファイルにリストされていません。

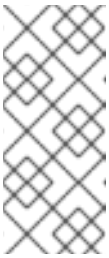
手順

1. **administrator@ad-domain** メンバーを持つ **ad_users_external** グループを含む **ad_users** グループを作成します。

a. **オプション**: AD ドメインで対応するグループを作成または選択して、IdM レルムで AD ユーザーを管理するために使用します。複数の AD グループを使用して、それらを IdM 側の異なるグループに追加できます。

b. **ad_users_external** グループを作成し、**--external** オプションを追加して、IdM ドメイン外のメンバーが含まれていることを示します。

```
[root@ipaserver ~]# ipa group-add --desc='AD users external map'
ad_users_external --external
-----------------------------
Added group "ad_users_external"
-----------------------------
  Group name: ad_users_external
  Description: AD users external map
```



注記

ここで指定する外部グループが、Active Directory セキュリティーグループドキュメントで定義されているように、**global** または **universal** グループスコープを持つ AD セキュリティーグループであることを確認してください。たとえば、グループスコープが **domain local** であるため、**Domain users** または **Domain admins** AD セキュリティーグループは使用できません。

c. **ad_users** グループを作成します。

```
[root@ipaserver ~]# ipa group-add --desc='AD users' ad_users
---------------------
Added group "ad_users"
---------------------
  Group name: ad_users
  Description: AD users
  GID: 129600004
```

d. **administrator@ad-domain.com** AD ユーザーを外部メンバーとして **ad_users_external** に追加します。

```
[root@ipaserver ~]# ipa group-add-member ad_users_external --external
"administrator@ad-domain.com"
 [member user]:
 [member group]:
  Group name: ad_users_external
  Description: AD users external map
  External member: S-1-5-21-3655990580-1375374850-1633065477-513
------------------------
Number of members added 1
------------------------
```

AD ユーザーは、**DOMAIN\user_name** または **user_name@DOMAIN** などの完全修飾名で識別される必要があります。次に、AD ID がユーザーの AD SID にマップされます。同じことが AD グループの追加にも当てはまります。

e. **ad_users_external** を **ad_users** にメンバーとして追加します。

```
[root@ipaserver ~]# ipa group-add-member ad_users --groups ad_users_external
```

```
  Group name: ad_users
  Description: AD users
  GID: 129600004
  Member groups: ad_users_external
-------------------------
Number of members added 1
-------------------------
```

2. **ad_users** のメンバーに、**idmclient** ホストで **/usr/sbin/reboot** を実行する権限を付与します。

   a. **sudo** コマンドの IdM データベースに **/usr/sbin/reboot** コマンドを追加します。

```
[root@idmclient ~]# ipa sudocmd-add /usr/sbin/reboot
-----------------------------------
Added Sudo Command "/usr/sbin/reboot"
-----------------------------------
  Sudo Command: /usr/sbin/reboot
```

   b. **ad_users_reboot** という名前の **sudo** ルールを作成します。

```
[root@idmclient ~]# ipa sudorule-add ad_users_reboot
--------------------------------
Added Sudo Rule "ad_users_reboot"
--------------------------------
  Rule name: ad_users_reboot
  Enabled: True
```

   c. **/usr/sbin/reboot** コマンドを **ad_users_reboot** ルールに追加します。

```
[root@idmclient ~]# ipa sudorule-add-allow-command ad_users_reboot --sudocmds
'/usr/sbin/reboot'
  Rule name: ad_users_reboot
  Enabled: True
  Sudo Allow Commands: /usr/sbin/reboot
-------------------------
Number of members added 1
-------------------------
```

   d. **ad_users_reboot** ルールを IdM **idmclient** ホストに適用します。

```
[root@idmclient ~]# ipa sudorule-add-host ad_users_reboot --hosts
idmclient.idm.example.com
Rule name: ad_users_reboot
Enabled: True
Hosts: idmclient.idm.example.com
Sudo Allow Commands: /usr/sbin/reboot
-------------------------
Number of members added 1
-------------------------
```

   e. **ad_users** グループを **ad_users_reboot** ルールに追加します。

```
[root@idmclient ~]# ipa sudorule-add-user ad_users_reboot --groups ad_users
Rule name: ad_users_reboot
```

```
Enabled: TRUE
User Groups: ad_users
Hosts: idmclient.idm.example.com
Sudo Allow Commands: /usr/sbin/reboot
-------------------------
Number of members added 1
-------------------------
```

注記

サーバーからクライアントへの変更の伝播には数分かかる場合があります。

検証手順

1. **ad_users** グループの間接メンバーである **administrator@ad-domain.com** で **idmclient** ホストにログインします。

```
$ ssh administrator@ad-domain.com@ipaclient
Password:
```

2. オプションで、**administrator@ad-domain.com** が実行できる **sudo** コマンドを表示します。

```
[administrator@ad-domain.com@idmclient ~]$ sudo -l
Matching Defaults entries for administrator@ad-domain.com on idmclient:
    !visiblepw, always_set_home, match_group_by_gid, always_query_group_plugin,
    env_reset, env_keep="COLORS DISPLAY HOSTNAME HISTSIZE KDEDIR
LS_COLORS",
    env_keep+="MAIL PS1 PS2 QTDIR USERNAME LANG LC_ADDRESS LC_CTYPE",
    env_keep+="LC_COLLATE LC_IDENTIFICATION LC_MEASUREMENT
LC_MESSAGES",
    env_keep+="LC_MONETARY LC_NAME LC_NUMERIC LC_PAPER LC_TELEPHONE",
    env_keep+="LC_TIME LC_ALL LANGUAGE LINGUAS _XKB_CHARSET XAUTHORITY
KRB5CCNAME",
    secure_path=/sbin\:/bin\:/usr/sbin\:/usr/bin

User administrator@ad-domain.com may run the following commands on idmclient:
    (root) /usr/sbin/reboot
```

3. **sudo** を使用してマシンを再起動します。プロンプトが表示されたら、**administrator@ad-domain.com** のパスワードを入力します。

```
[administrator@ad-domain.com@idmclient ~]$ sudo /usr/sbin/reboot
[sudo] password for administrator@ad-domain.com:
```

関連情報

- Active Directory ユーザーおよび Identity Management グループ
- Include users and groups from a trusted Active Directory domain into SUDO rules

55.4. IDM WEB UI を使用した IDM クライアントでの IDM ユーザーへの SUDO アクセス権の付与

Identity Management (IdM) では、特定の IdM ホストで IdM ユーザーアカウントの特定コマンドに **sudo** アクセスを付与できます。最初に **sudo** コマンドを追加してから、1 つまたは複数のコマンドに対して **sudo** ルールを作成します。

idmclient マシンで **/usr/sbin/reboot** コマンドを実行する権限を **idm_user** に付与する **idm_user_reboot** の sudo ルールを作成するには、以下の手順を実行します。

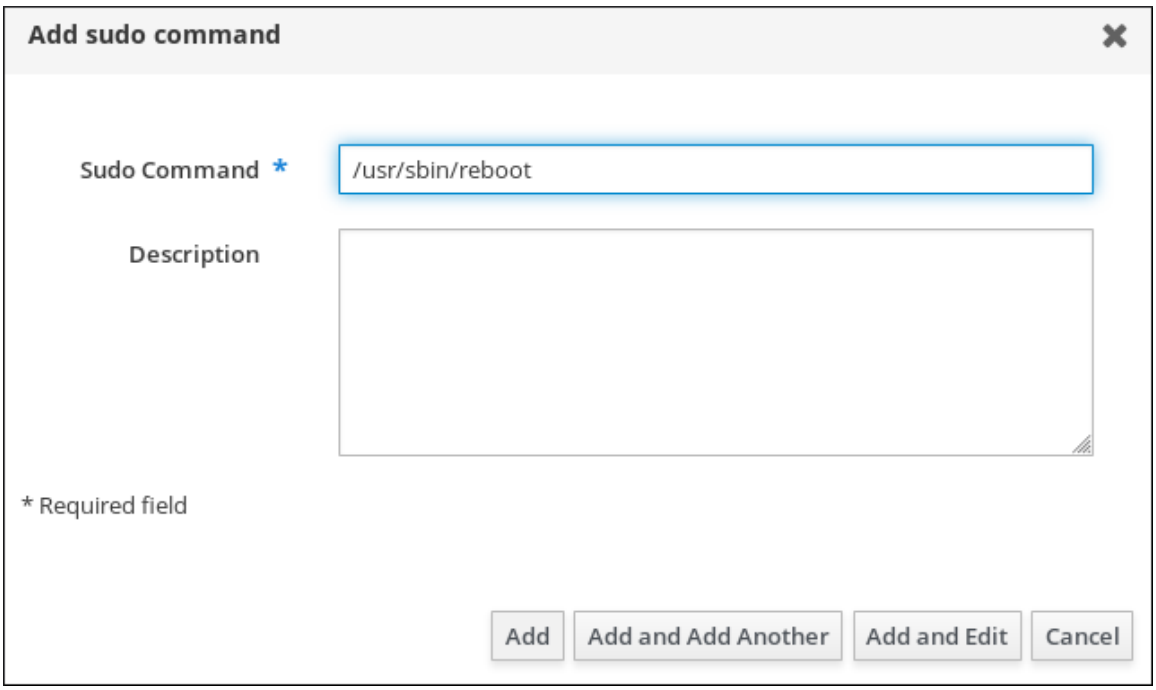
前提条件

- IdM 管理者としてログインしている。
- IdM で **idm_user** のユーザーアカウントを作成し、ユーザーのパスワードを作成してそのアカウントのロックを解除している。コマンドラインインターフェイスを使用して新しい IdM ユーザーを追加する方法の詳細は、コマンドラインを使用したユーザーの追加 を参照してください。
- **idmclient** ホストにローカル **idm_user** アカウントが存在しない。**idm_user** ユーザーは、ローカルの **/etc/passwd** ファイルには表示されません。

手順

1. **sudo** コマンドの IdM データベースに **/usr/sbin/reboot** コマンドを追加します。
   a. **Policy → Sudo → Sudo Commands** の順に移動します。
   b. 右上にある **Add** をクリックして、**Add sudo command** ダイアログボックスを開きます。
   c. **sudo**: **/usr/sbin/reboot** を使用してユーザーが実行できるコマンドを入力します。

   図55.1 IdM sudo コマンドの追加



   d. **Add** をクリックします。
2. 新しい **sudo** コマンドエントリーを使用して sudo ルールを作成し、**idm_user** が **idmclient** マシンを再起動できるようにします。
   a. **Policy → Sudo → Sudo** ルールに移動します。

b. 右上にある Add をクリックして、Add sudo rule ダイアログボックスを開きます。

c. **sudo** ルールの名前を入力します (idm_user_reboot)。

d. Add and Edit をクリックします。

e. ユーザーを指定します。

   i. Who セクションで、Specified Users and Groups のラジオボタンを選択します。

   ii. User category the rule applies to のサブセクションで Add をクリックして、Add users into sudo rule "idm_user_reboot" ダイアログボックスを開きます。

   iii. Add users into sudo rule "idm_user_reboot" ダイアログボックスにある Available 列で、idm_user チェックボックスを選択し、これを Prospective 列に移動します。

   iv. Add をクリックします。

f. ホストを指定します。

   i. Access this host セクションで、Specified Hosts and Groups ラジオボタンを確認します。

   ii. Host category this rule applies to サブセクションで Add をクリックして、Add hosts into sudo rule "idm_user_reboot" ダイアログボックスを開きます。

   iii. Add hosts into sudo rule "idm_user_reboot" ダイアログボックスにある Available 列で、idmclient.idm.example.com チェックボックスを選択し、これを Prospective 列に移動します。

   iv. Add をクリックします。

g. コマンドを指定します。

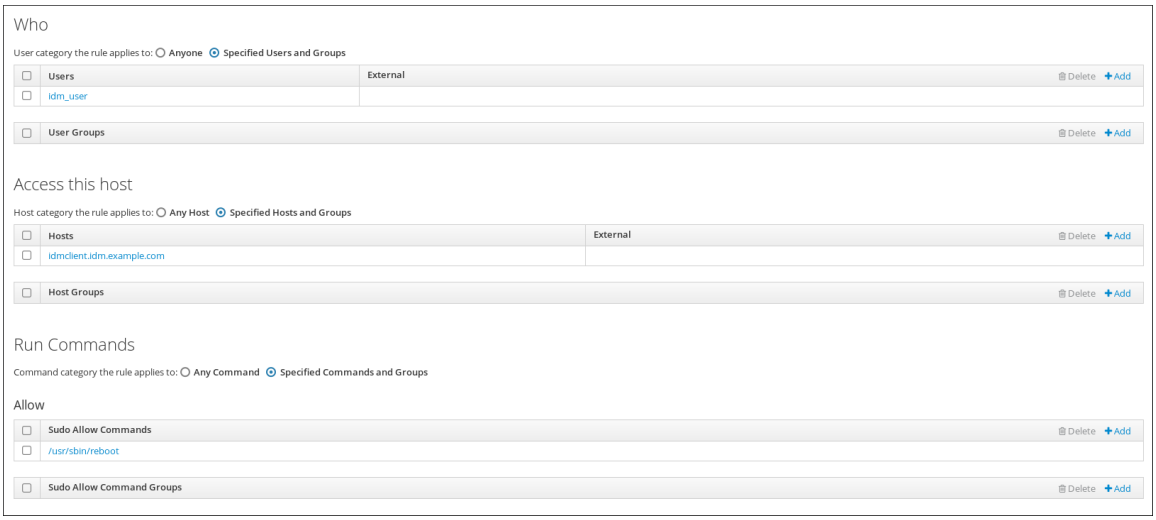
   i. Run Commands セクションの Command category the rule applies to サブセクションで、Specified Commands and Groups ラジオボタンにチェックを入れます。

   ii. Sudo Allow Commands サブセクションで Add をクリックして、Add allow sudo commands into sudo rule "idm_user_reboot" ダイアログボックスを開きます。

   iii. Add allow sudo commands into sudo rule "idm_user_reboot" ダイアログボックスにある Available 列で、**/usr/sbin/reboot** チェックボックスを選択し、これを Prospective 列に移動します。

   iv. Add をクリックして、idm_sudo_reboot ページに戻ります。

図55.2 IdM sudo ルールの追加



h. 左上隅にある **Save** をクリックします。

新しいルールはデフォルトで有効になります。

注記

サーバーからクライアントへの変更の伝播には数分かかる場合があります。

検証手順

1. **idmclient** に **idm_user** としてログインします。
2. **sudo** を使用してマシンを再起動します。プロンプトが表示されたら、**idm_user** のパスワードを入力します。

```
$ sudo /usr/sbin/reboot
[sudo] password for idm_user:
```

sudo ルールが正しく設定されている場合には、マシンが再起動します。

55.5. IDM クライアントでサービスアカウントとしてコマンドを実行する CLI での SUDO ルールの作成

IdM では、**RunAs エイリアス** を使用して、**sudo** ルールを設定し、別のユーザーまたはグループとして **sudo** コマンドを実行できます。たとえば、データベースアプリケーションをホストする IdM クライアントが存在し、そのアプリケーションに対応するローカルサービスアカウントとしてコマンドを実行する必要があるとします。

この例を使用して、**run_third-party-app_report** と呼ばれるコマンドラインに **sudo** ルールを作成し、**idm_user** アカウントが **idmclient** ホストの **thirdpartyapp** サービスアカウントとして **/opt/third-party-app/bin/report** コマンドを実行できるようにします。

前提条件

- IdM 管理者としてログインしている。

- IdM で **idm_user** のユーザーアカウントを作成し、ユーザーのパスワードを作成してそのアカウントのロックを解除している。CLI を使用して新しい IdM ユーザーを追加する方法の詳細は、コマンドラインを使用したユーザーの追加 を参照してください。
- **idmclient** ホストにローカル **idm_user** アカウントが存在しない。**idm_user** ユーザーは、ローカルの **/etc/passwd** ファイルには表示されません。
- **idmclient** ホストに、**third-party-app** という名前のカスタムアプリケーションがインストールされている。
- **third-party-app** アプリケーションの **report** コマンドが、**/opt/third-party-app/bin/report** ディレクトリーにインストールされている。
- **third-party-app** アプリケーションにコマンドを実行するために、**thirdpartyapp** という名前のローカルサービスアカウントを作成している。

手順

1. IdM の **管理者** として Kerberos チケットを取得します。

```
[root@idmclient ~]# kinit admin
```

2. **/opt/third-party-app/bin/report** コマンドを、**sudo** コマンドの IdM データベースに追加します。

```
[root@idmclient ~]# ipa sudocmd-add /opt/third-party-app/bin/report
----------------------------------------------------
Added Sudo Command "/opt/third-party-app/bin/report"
----------------------------------------------------
  Sudo Command: /opt/third-party-app/bin/report
```

3. **run_third-party-app_report** という名前の **sudo** ルールを作成します。

```
[root@idmclient ~]# ipa sudorule-add run_third-party-app_report
-------------------------------------------
Added Sudo Rule "run_third-party-app_report"
-------------------------------------------
  Rule name: run_third-party-app_report
  Enabled: TRUE
```

4. **--users=<user>** オプションを使用して、**sudorule-add-runasuser** コマンドに RunAs ユーザーを指定します。

```
[root@idmclient ~]# ipa sudorule-add-runasuser run_third-party-app_report --
users=thirdpartyapp
  Rule name: run_third-party-app_report
  Enabled: TRUE
  RunAs External User: thirdpartyapp
------------------------
Number of members added 1
------------------------
```

ユーザー (または **--groups=*** オプションで指定したグループ) は、ローカルサービスアカウントや Active Directory ユーザーなどの IdM の外部に配置できます。グループ名には **%** 接頭辞を追加しないでください。

5. **/opt/third-party-app/bin/report** コマンドを **run_third-party-app_report** ルールに追加します。

```
[root@idmclient ~]# ipa sudorule-add-allow-command run_third-party-app_report --
sudocmds '/opt/third-party-app/bin/report'
Rule name: run_third-party-app_report
Enabled: TRUE
Sudo Allow Commands: /opt/third-party-app/bin/report
RunAs External User: thirdpartyapp
-------------------------
Number of members added 1
-------------------------
```

6. **run_third-party-app_report** ルールを IdM **idmclient** ホストに適用します。

```
[root@idmclient ~]# ipa sudorule-add-host run_third-party-app_report --hosts
idmclient.idm.example.com
Rule name: run_third-party-app_report
Enabled: TRUE
Hosts: idmclient.idm.example.com
Sudo Allow Commands: /opt/third-party-app/bin/report
RunAs External User: thirdpartyapp
-------------------------
Number of members added 1
-------------------------
```

7. **idm_user** アカウントーを **run_third-party-app_report** ルールに追加します。

```
[root@idmclient ~]# ipa sudorule-add-user run_third-party-app_report --users idm_user
Rule name: run_third-party-app_report
Enabled: TRUE
Users: idm_user
Hosts: idmclient.idm.example.com
Sudo Allow Commands: /opt/third-party-app/bin/report
RunAs External User: thirdpartyapp
-------------------------
Number of members added 1
```

注記

サーバーからクライアントへの変更の伝播には数分かかる場合があります。

検証手順

1. **idmclient** ホストに **idm_user** アカウントとしてログインします。
2. 新しい sudo ルールをテストします。
   a. **idm_user** アカウントが実行可能な **sudo** ルールを表示します。

```
[idm_user@idmclient ~]$ sudo -l
Matching Defaults entries for idm_user@idm.example.com on idmclient:
    !visiblepw, always_set_home, match_group_by_gid, always_query_group_plugin,
    env_reset, env_keep="COLORS DISPLAY HOSTNAME HISTSIZE KDEDIR
```

```
LS_COLORS",
    env_keep+="MAIL PS1 PS2 QTDIR USERNAME LANG LC_ADDRESS LC_CTYPE",
    env_keep+="LC_COLLATE LC_IDENTIFICATION LC_MEASUREMENT
LC_MESSAGES",
    env_keep+="LC_MONETARY LC_NAME LC_NUMERIC LC_PAPER
LC_TELEPHONE",
    env_keep+="LC_TIME LC_ALL LANGUAGE LINGUAS _XKB_CHARSET
XAUTHORITY KRB5CCNAME",
    secure_path=/sbin\:/bin\:/usr/sbin\:/usr/bin

User idm_user@idm.example.com may run the following commands on idmclient:
    (thirdpartyapp) /opt/third-party-app/bin/report
```

b. **report** コマンドを **thirdpartyapp** サービスアカウントとして実行します。

```
[idm_user@idmclient ~]$ sudo -u thirdpartyapp /opt/third-party-app/bin/report
[sudo] password for idm_user@idm.example.com:
Executing report...
Report successful.
```

55.6. IDM クライアントでサービスアカウントとしてコマンドを実行する IDM WEBUI での SUDO ルールの作成

IdM では、**RunAs エイリアス** を使用して、**sudo** ルールを設定し、別のユーザーまたはグループとして **sudo** コマンドを実行できます。たとえば、データベースアプリケーションをホストする IdM クライアントが存在し、そのアプリケーションに対応するローカルサービスアカウントとしてコマンドを実行する必要があるとします。

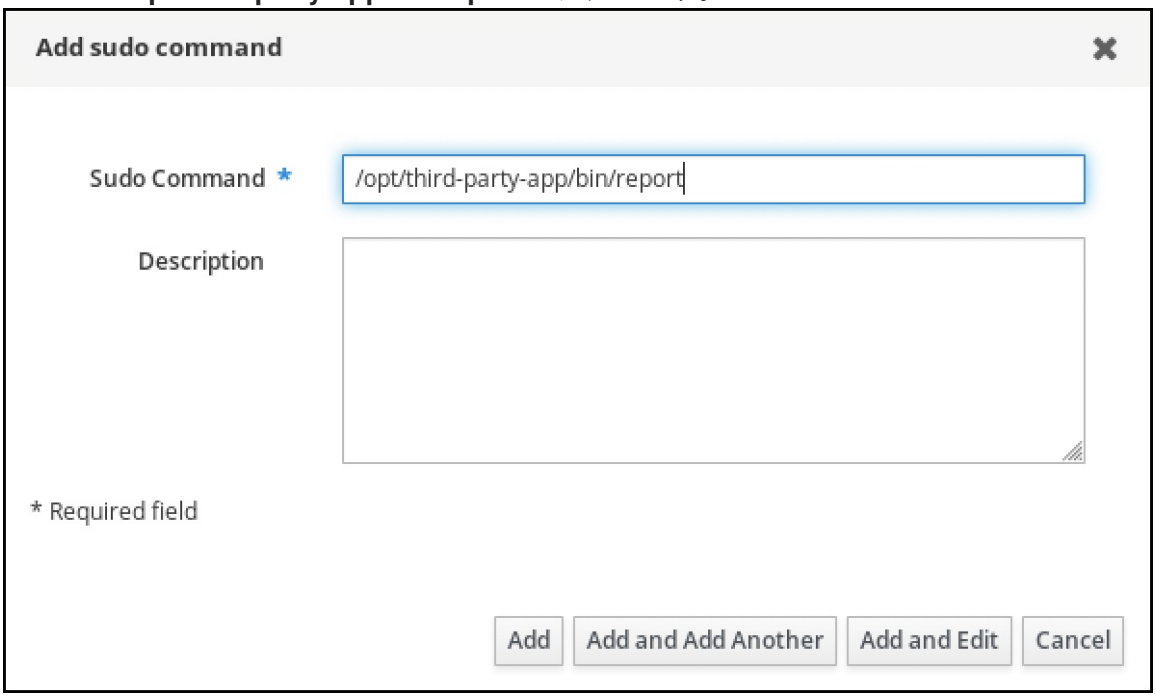
この例を使用して、**run_third-party-app_report** という IdM WebUI に **sudo** ルールを作成し、**idm_user** アカウントが **idmclient** ホストで **thirdpartyapp** サービスアカウントとして **/opt/third-party-app/bin/report** コマンドを実行できるようにします。

前提条件

- IdM 管理者としてログインしている。
- IdM で **idm_user** のユーザーアカウントを作成し、ユーザーのパスワードを作成してそのアカウントのロックを解除している。CLI を使用して新しい IdM ユーザーを追加する方法の詳細は、コマンドラインを使用したユーザーの追加 を参照してください。
- **idmclient** ホストにローカル **idm_user** アカウントが存在しない。**idm_user** ユーザーは、ローカルの **/etc/passwd** ファイルには表示されません。
- **idmclient** ホストに、**third-party-app** という名前のカスタムアプリケーションがインストールされている。
- **third-party-app** アプリケーションの **report** コマンドが、**/opt/third-party-app/bin/report** ディレクトリーにインストールされている。
- **third-party-app** アプリケーションにコマンドを実行するために、**thirdpartyapp** という名前のローカルサービスアカウントを作成している。

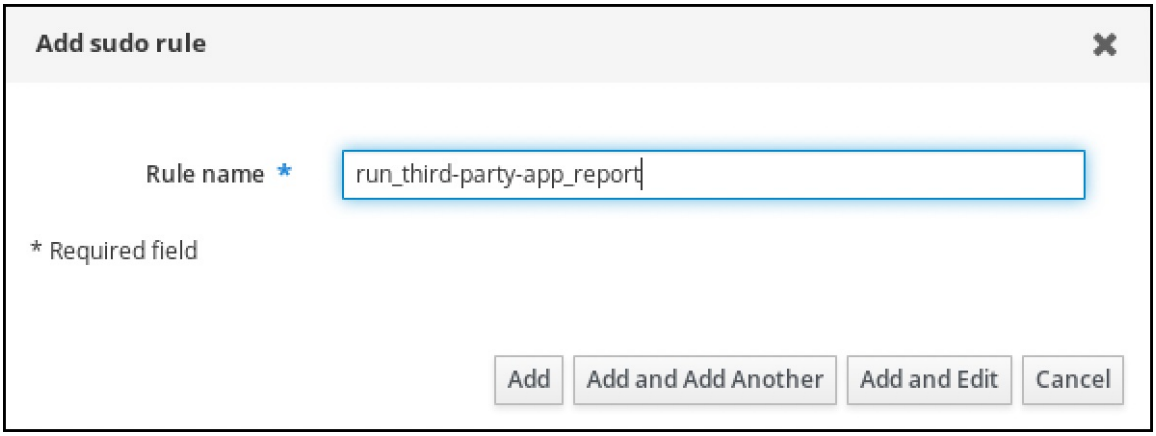
手順

1. **/opt/third-party-app/bin/report** コマンドを、**sudo** コマンドの IdM データベースに追加します。
   a. **Policy → Sudo → Sudo Commands** の順に移動します。
   b. 右上にある **Add** をクリックして、**Add sudo command** ダイアログボックスを開きます。
   c. コマンド **/opt/third-party-app/bin/report** を入力します。



   d. **Add** をクリックします。

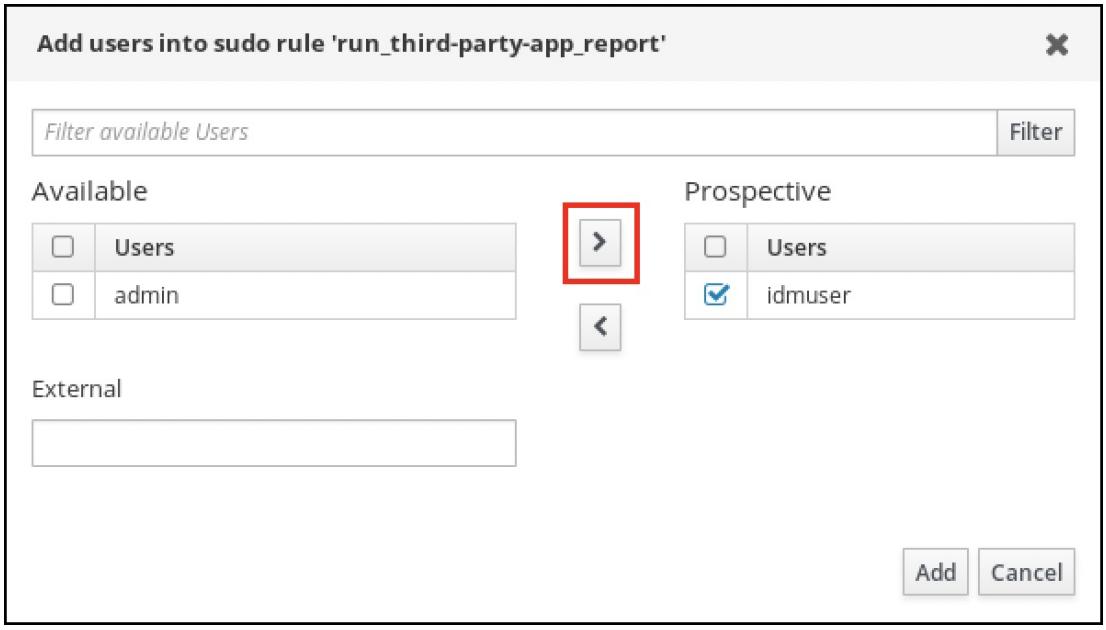
2. 新しい **sudo** コマンドエントリーを使用して、新しい **sudo** ルールを作成します。
   a. **Policy → Sudo → Sudo** ルールに移動します。
   b. 右上にある **Add** をクリックして、**Add sudo rule** ダイアログボックスを開きます。
   c. **sudo** ルールの名前 **run_third-party-app_report** を入力します。



   d. **Add and Edit** をクリックします。
   e. ユーザーを指定します。
      i. **Who** セクションで、**Specified Users and Groups** のラジオボタンを選択します。

ii. User category the rule applies toのサブセクションで Add をクリックして、Add users into sudo rule "run_third-party-app_report" ダイアログボックスを開きます。

iii. Available 列の Add users into sudo rule "run_third-party-app_report" ダイアログボックスで、idm_user チェックボックスをオンにして、これを Prospective 列に移動します。



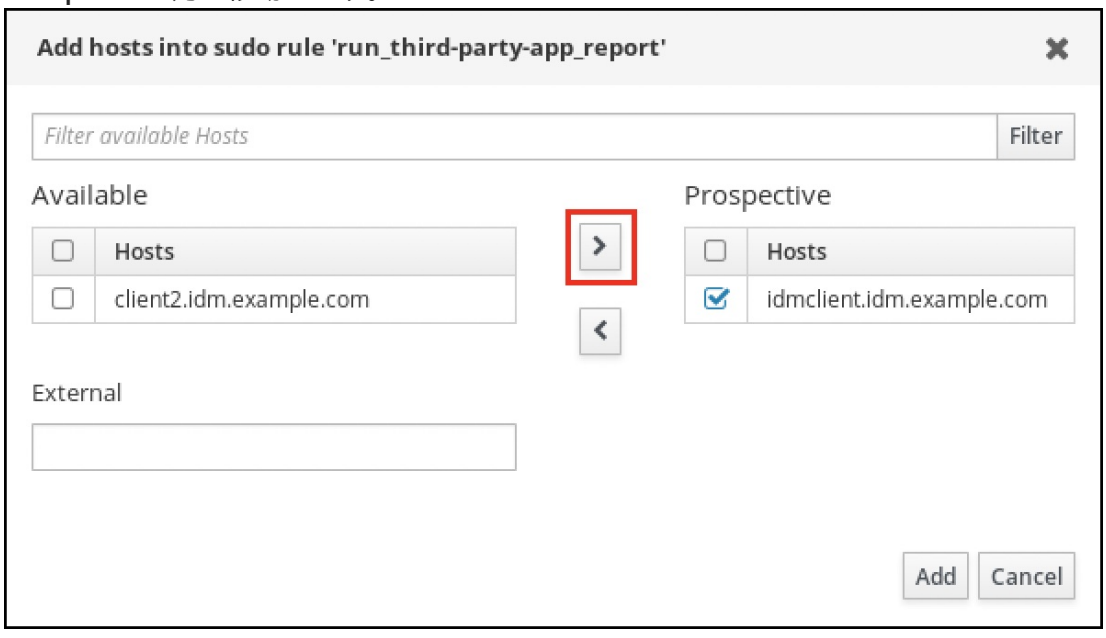
iv. Add をクリックします。

f. ホストを指定します。

i. Access this host セクションで、Specified Hosts and Groups ラジオボタンを確認します。

ii. Host category this rule applies to サブセクションで Add をクリックして、 Add hosts into sudo rule "run_third-party-app_report" ダイアログボックスを開きます。

iii. Available 列の Add hosts into sudo rule "run_third-party-app_report" ダイアログボックスで、idmclient.idm.example.com チェックボックスをオンにして、これを Prospective 列に移動します。



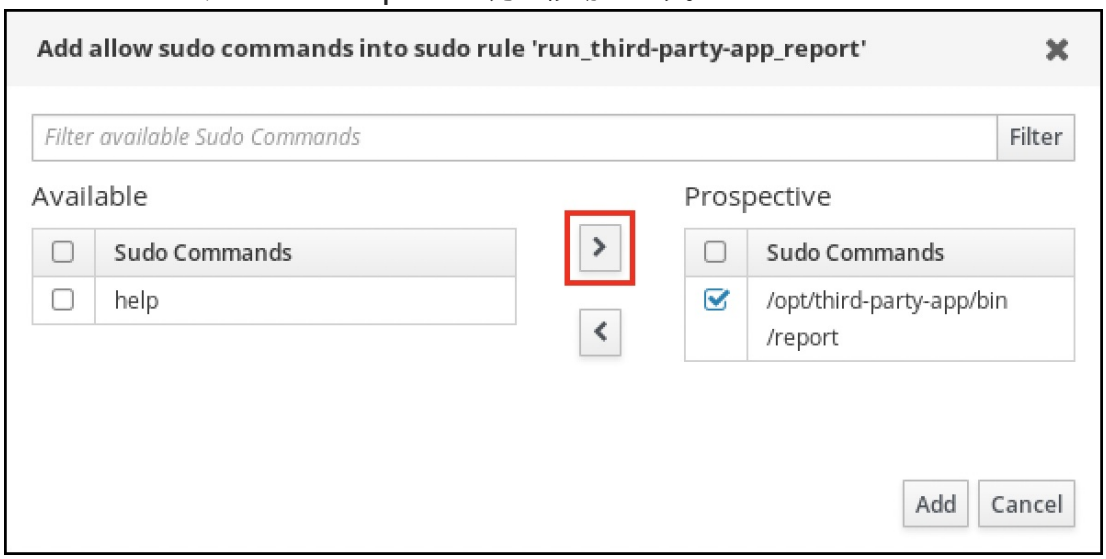
iv. Add をクリックします。

g. コマンドを指定します。

i. **Run Commands** セクションの **Command category the rule applies to**サブセクションで、**Specified Commands and Groups** ラジオボタンにチェックを入れます。

ii. **Sudo Allow Commands** サブセクションで **Add** をクリックして、**Add allow sudo commands into sudo rule "run_third-party-app_report"** ダイアログボックスを開きます。

iii. **Available** 列の **Add allow sudo commands into sudo rule "run_third-party-app_report"** ダイアログボックスで、**/opt/third-party-app/bin/report** チェックボックスをオンにして、これを **Prospective** 列に移動します。



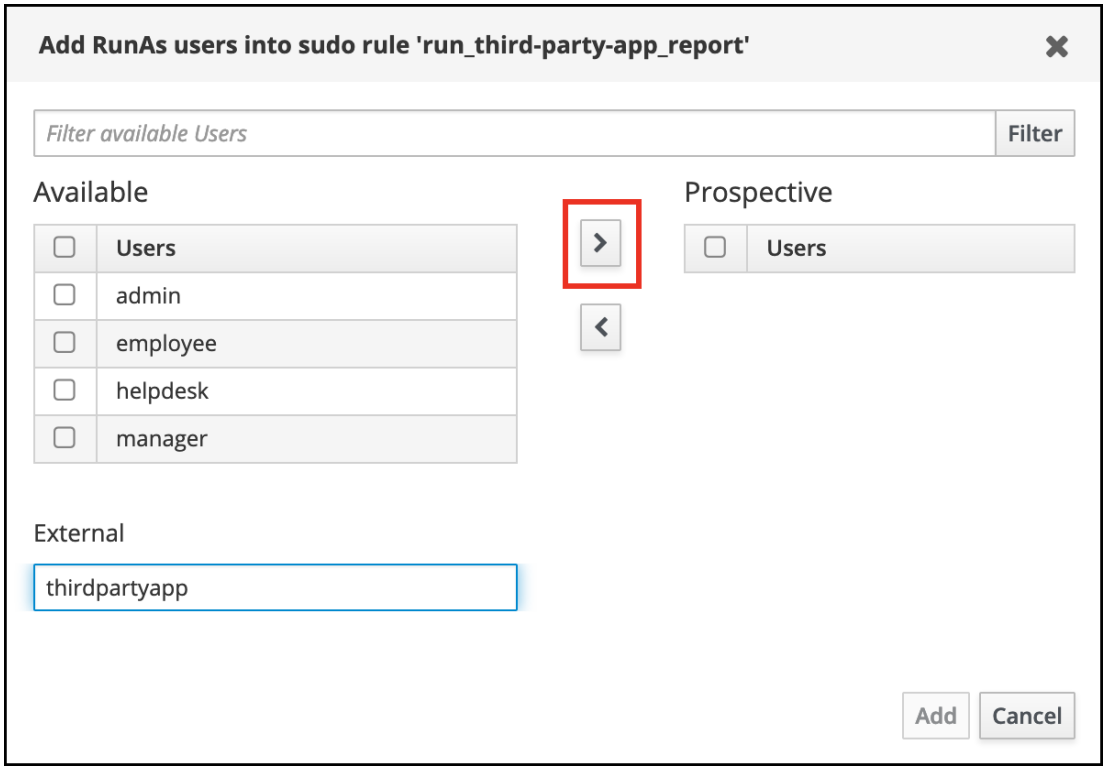
iv. **Add** をクリックして、**run_third-party-app_report** のページに戻ります。

h. RunAs ユーザーを指定します。

i. **As Whom** で、**指定したユーザーとグループ** のラジオボタンを確認します。

ii. **RunAs ユーザー** サブセクションで **Add** をクリックして、**Add RunAs users into sudo rule "run_third-party-app_report"** ダイアログボックスを開きます。

iii. **Add RunAs users into sudo rule "run_third-party-app_report"** ダイアログボックスで、**External** ボックスに **thirdpartyapp** サービスアカウントを入力し、これを **Prospective** 列に移動します。

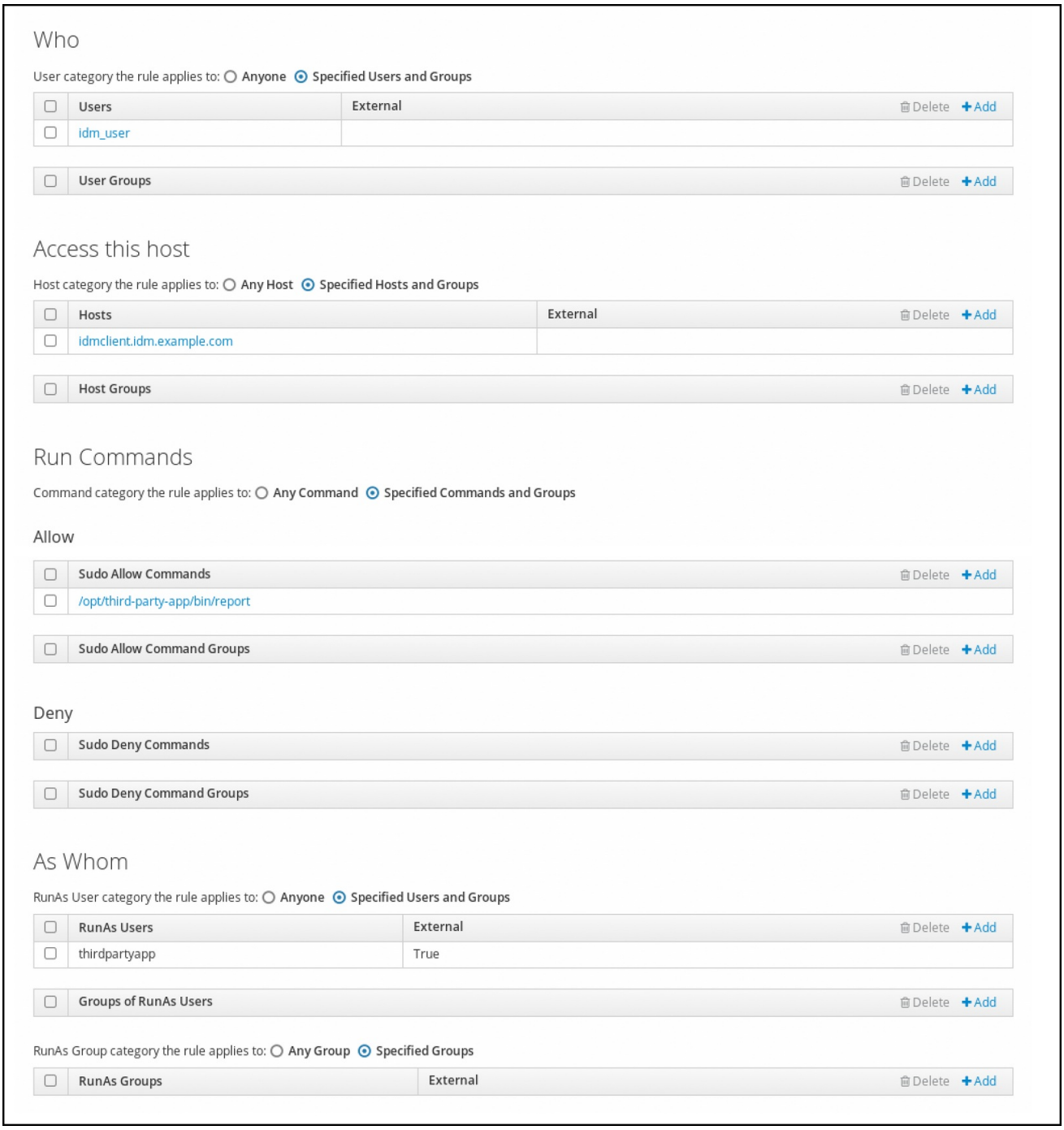


iv. **Add** をクリックして、**run_third-party-app_report** のページに戻ります。

i. 左上隅にある **Save** をクリックします。

新しいルールはデフォルトで有効になります。

図55.3 sudo ルールの詳細



注記

サーバーからクライアントへの変更の伝播には数分かかる場合があります。

検証手順

1. **idmclient** ホストに **idm_user** アカウントとしてログインします。
2. 新しい sudo ルールをテストします。
   a. **idm_user** アカウントが実行可能な **sudo** ルールを表示します。

```
[idm_user@idmclient ~]$ sudo -l
Matching Defaults entries for idm_user@idm.example.com on idmclient:
    !visiblepw, always_set_home, match_group_by_gid, always_query_group_plugin,
```

```
    env_reset, env_keep="COLORS DISPLAY HOSTNAME HISTSIZE KDEDIR
LS_COLORS",
    env_keep+="MAIL PS1 PS2 QTDIR USERNAME LANG LC_ADDRESS LC_CTYPE",
    env_keep+="LC_COLLATE LC_IDENTIFICATION LC_MEASUREMENT
LC_MESSAGES",
    env_keep+="LC_MONETARY LC_NAME LC_NUMERIC LC_PAPER
LC_TELEPHONE",
    env_keep+="LC_TIME LC_ALL LANGUAGE LINGUAS _XKB_CHARSET
XAUTHORITY KRB5CCNAME",
    secure_path=/sbin\:/bin\:/usr/sbin\:/usr/bin

User idm_user@idm.example.com may run the following commands on idmclient:
    (thirdpartyapp) /opt/third-party-app/bin/report
```

b. **report** コマンドを **thirdpartyapp** サービスアカウントとして実行します。

```
[idm_user@idmclient ~]$ sudo -u thirdpartyapp /opt/third-party-app/bin/report
[sudo] password for idm_user@idm.example.com:
Executing report...
Report successful.
```

55.7. IDM クライアントでの SUDO の GSSAPI 認証の有効化

以下の手順では、**pam_sss_gss.so** PAM モジュールを介して、**sudo** コマンドおよび **sudo -i** コマンドの IdM クライアントで、Generic Security Service Application Program Interface (GSSAPI) 認証を有効にする方法を説明します。この設定により、IdM ユーザーは Kerberos チケットを使用して **sudo** コマンドに対する認証が可能になります。

前提条件

- IdM ホストに適用する IdM ユーザーの **sudo** ルールを作成している。この例では、**idmclient** ホストで **/usr/sbin/reboot** コマンドを実行するパーミッションを **idm_user** アカウントに付与する **idm_user_reboot sudo** ルールが作成済みです。
- **idmclient** ホストが RHEL 8.4 以降を実行している。
- **/etc/sssd/sssd.conf** ファイルと、**/etc/pam.d/** ディレクトリーの PAM ファイルを変更するための **root** 特権がある。

手順

1. **/etc/sssd/sssd.conf** 設定ファイルを開きます。
2. **[domain/<domain_name>]** セクションに以下のエントリーを追加します。

```
[domain/<domain_name>]
pam_gssapi_services = sudo, sudo-i
```

3. **/etc/sssd/sssd.conf** ファイルを保存して閉じます。
4. SSSD サービスを再起動して、設定の変更を読み込みます。

```
[root@idmclient ~]# systemctl restart sssd
```

5. RHEL 8.8 以降を実行している場合:

   a. (オプション) **sssd authselect** プロファイルを選択したかどうかを確認します。

   ```
   # authselect current
   Profile ID: sssd
   ```

   出力に、**sssd authselect** プロファイルが選択されていることが示されます。

   b. **sssd authselect** プロファイルが選択されている場合は、GSSAPI 認証を有効にします。

   ```
   # authselect enable-feature with-gssapi
   ```

   c. **sssd authselect** プロファイルが選択されていない場合は、それを選択して GSSAPI 認証を有効にします。

   ```
   # authselect select sssd with-gssapi
   ```

6. RHEL 8.7 以前を実行している場合:

   a. **/etc/pam.d/sudo** の PAM 設定ファイルを開きます。

   b. 以下のエントリーを、**/etc/pam.d/sudo** ファイルの **auth** セクションの最初の行に追加します。

   ```
   #%PAM-1.0
   auth sufficient pam_sss_gss.so
   auth       include      system-auth
   account    include      system-auth
   password   include      system-auth
   session    include      system-auth
   ```

   c. **/etc/pam.d/sudo** ファイルを保存して閉じます。

検証手順

1. **idm_user** アカウントとしてホストにログインします。

```
[root@idm-client ~]# ssh -l idm_user@idm.example.com localhost
idm_user@idm.example.com's password:
```

2. **idm_user** アカウントで Ticket-Granting Ticket があることを確認します。

```
[idmuser@idmclient ~]$ klist
Ticket cache: KCM:1366201107
Default principal: idm_user@IDM.EXAMPLE.COM

Valid starting       Expires              Service principal
01/08/2021 09:11:48  01/08/2021 19:11:48
krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
 renew until 01/15/2021 09:11:44
```

3. **(オプション) idm_user** アカウントの Kerberos 認証情報がない場合は、現在の Kerberos 認証情報を削除し、正しい認証情報を要求します。

```
[idm_user@idmclient ~]$ kdestroy -A

[idm_user@idmclient ~]$ kinit idm_user@IDM.EXAMPLE.COM
Password for idm_user@idm.example.com:
```

4. パスワードを指定せずに **sudo** を使用してマシンを再起動します。

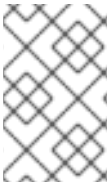
```
[idm_user@idmclient ~]$ sudo /usr/sbin/reboot
```

関連情報

- IdM の用語 一覧の GSSAPI エントリー
- IdM Web UI で IdM クライアントの IdM ユーザーへの sudo アクセスの許可
- CLI での IdM クライアントの IdM ユーザーへの sudo アクセス許可
- **pam_sss_gss(8)** の man ページ
- **sssd.conf (5)** の man ページ

55.8. IDM クライアントでの GSSAPI 認証の有効化および SUDO の KERBEROS 認証インジケーターの有効化

以下の手順では、**pam_sss_gss.so** PAM モジュールを介して、**sudo** コマンドおよび **sudo -i** コマンドの IdM クライアントで、Generic Security Service Application Program Interface (GSSAPI) 認証を有効にする方法を説明します。また、スマートカードを使用してログインしたユーザーのみが Kerberos チケットでこれらのコマンドに対して認証されます。



注記

この手順をテンプレートとして使用し、他の PAM 対応サービスに対して SSSD で GSSAPI 認証を設定して、さらに特定の認証インジケーターが Kerberos チケットにアタッチされているユーザーだけにアクセスを限定することができます。

前提条件

- IdM ホストに適用する IdM ユーザーの **sudo** ルールを作成している。この例では、**idmclient** ホストで **/usr/sbin/reboot** コマンドを実行するパーミッションを **idm_user** アカウントに付与する **idm_user_reboot sudo** ルールが作成済みです。
- **idmclient** ホストにスマートカード認証を設定している。
- **idmclient** ホストが RHEL 8.4 以降を実行している。
- **/etc/sssd/sssd.conf** ファイルと、**/etc/pam.d/** ディレクトリーの PAM ファイルを変更するための **root** 特権がある。

手順

1. **/etc/sssd/sssd.conf** 設定ファイルを開きます。
2. **[domain/<domain_name>]** セクションに以下のエントリーを追加します。

```
[domain/<domain_name>]
pam_gssapi_services = sudo, sudo-i
pam_gssapi_indicators_map = sudo:pkinit, sudo-i:pkinit
```

3. **/etc/sssd/sssd.conf** ファイルを保存して閉じます。
4. SSSD サービスを再起動して、設定の変更を読み込みます。

```
[root@idmclient ~]# systemctl restart sssd
```

5. **/etc/pam.d/sudo** の PAM 設定ファイルを開きます。
6. 以下のエントリーを、**/etc/pam.d/sudo** ファイルの **auth** セクションの最初の行に追加します。

```
#%PAM-1.0
auth sufficient pam_sss_gss.so
auth       include      system-auth
account    include      system-auth
password   include      system-auth
session    include      system-auth
```

7. **/etc/pam.d/sudo** ファイルを保存して閉じます。
8. **/etc/pam.d/sudo-i** の PAM 設定ファイルを開きます。
9. 以下のエントリーを、**/etc/pam.d/sudo-i** ファイルの **auth** セクションの最初の行に追加します。

```
#%PAM-1.0
auth sufficient pam_sss_gss.so
auth       include      sudo
account    include      sudo
password   include      sudo
session    optional     pam_keyinit.so force revoke
session    include      sudo
```

10. **/etc/pam.d/sudo-i** ファイルを保存して閉じます。

検証手順

1. **idm_user** アカウントとしてホストにログインし、スマートカードで認証します。

```
[root@idmclient ~]# ssh -l idm_user@idm.example.com localhost
PIN for smart_card
```

2. スマートカードユーザーを使用して Ticket-Granting Ticket があることを確認します。

```
[idm_user@idmclient ~]$ klist
Ticket cache: KEYRING:persistent:1358900015:krb_cache_TObtNMd
Default principal: idm_user@IDM.EXAMPLE.COM

Valid starting       Expires              Service principal
```

```
02/15/2021 16:29:48  02/16/2021 02:29:48
krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
 renew until 02/22/2021 16:29:44
```

3. **idm_user** アカウントが実行可能な **sudo** ルールを表示します。

```
[idm_user@idmclient ~]$ sudo -l
Matching Defaults entries for idmuser on idmclient:
    !visiblepw, always_set_home, match_group_by_gid, always_query_group_plugin,
    env_reset, env_keep="COLORS DISPLAY HOSTNAME HISTSIZE KDEDIR
LS_COLORS",
    env_keep+="MAIL PS1 PS2 QTDIR USERNAME LANG LC_ADDRESS LC_CTYPE",
    env_keep+="LC_COLLATE LC_IDENTIFICATION LC_MEASUREMENT
LC_MESSAGES",
    env_keep+="LC_MONETARY LC_NAME LC_NUMERIC LC_PAPER LC_TELEPHONE",
    env_keep+="LC_TIME LC_ALL LANGUAGE LINGUAS _XKB_CHARSET XAUTHORITY
KRB5CCNAME",
    secure_path=/sbin\:/bin\:/usr/sbin\:/usr/bin

User idm_user may run the following commands on idmclient:
    (root) /usr/sbin/reboot
```

4. パスワードを指定せずに **sudo** を使用してマシンを再起動します。

```
[idm_user@idmclient ~]$ sudo /usr/sbin/reboot
```

関連情報

- PAM サービスの GSSAPI 認証を制御する SSSD オプション
- IdM の用語 一覧の GSSAPI エントリー
- スマートカード認証用の Identity Management の設定
- Kerberos 認証インジケーター
- IdM Web UI で IdM クライアントの IdM ユーザーへの sudo アクセスの許可
- CLI での IdM クライアントの IdM ユーザーへの sudo アクセス許可
- **pam_sss_gss(8)** の man ページ
- **sssd.conf (5)** の man ページ

55.9. PAM サービスの GSSAPI 認証を制御する SSSD オプション

/etc/sssd/sssd.conf 設定ファイルに以下のオプションを使用すると、SSSD サービス内の GSSAPI 設定を調整できます。

pam_gssapi_services

SSSD を使用した GSSAPI 認証はデフォルトで無効になっています。このオプションを使用すると、PAM モジュール **pam_sss_gss.so** を使用して GSSAPI 認証を試すことができる PAM サービスをコンマ区切りのリストで指定できます。GSSAPI 認証を明示的に無効にするには、このオプションを **-** に設定します。

pam_gssapi_indicators_map

このオプションは、Identity Management (IdM) ドメインにのみ適用されます。このオプションを使用して、サービスへの PAM のアクセスを付与するのに必要な Kerberos 認証インジケーターをリスト表示します。ペアの形式は **<PAM_service>:_<required_authentication_indicator>_** でなければなりません。
有効な認証インジケーターは以下のとおりです。

- **OTP** - 2 要素認証
- **radius** - RADIUS 認証
- **pkinit** - PKINIT、スマートカード、または証明書での認証
- **hardened** - 強化パスワード

pam_gssapi_check_upn

このオプションはデフォルトで有効となっており、**true** に設定されています。このオプションを有効にすると、SSSD サービスでは Kerberos 認証情報と同じユーザー名が必要になります。**false** の場合には、**pam_sss_gss.so** の PAM モジュールは、必要なサービスチケットを取得できるすべてのユーザーを認証します。

例

次のオプションでは、**sudo** と **sudo-i** サービスの Kerberos 認証を有効にします。この認証では、**sudo** ユーザーはワンタイムパスワードで認証する必要があり、ユーザー名と Kerberos プリンシパルが同じでなければなりません。この設定は **[pam]** セクションにあるため、すべてのドメインに適用されます。

```
[pam]
pam_gssapi_services = sudo, sudo-i
pam_gssapi_indicators_map = sudo:otp
pam_gssapi_check_upn = true
```

これらのオプションを個別の **[domain]** セクションで設定して、**[pam]** セクションのグローバル値を上書きすることもできます。次のオプションは、異なる GSSAPI 設定を各ドメインに適用します。

idm.example.com ドメインの場合

- **sudo** と **sudo -i** サービスの GSSAPI 認証を有効にする。
- **sudo** コマンドには、証明書またはスマートカード認証オーセンティケーターが必要である。
- **sudo -i** コマンドにはワンタイムパスワード認証が必要である。
- ユーザー名と Kerberos プリンシパルを常に一致させる必要がある。

ad.example.com ドメインの場合

- **sudo** サービスに対してのみ GSSAPI 認証を有効にする。
- ユーザー名とプリンシパルを強制的に一致させない。

```
[domain/idm.example.com]
pam_gssapi_services = sudo, sudo-i
pam_gssapi_indicators_map = sudo:pkinit, sudo-i:otp
```

```
pam_gssapi_check_upn = true
...

[domain/ad.example.com]
pam_gssapi_services = sudo
pam_gssapi_check_upn = false
...
```

関連情報

- Kerberos 認証インジケーター

55.10. SUDO の GSSAPI 認証のトラブルシューティング

IdM から Kerberos チケットを使用して **sudo** サービスに対する認証できない場合は、以下のシナリオを使用して設定のトラブルシューティングを行います。

前提条件

- **sudo** サービスの GSSAPI 認証が有効化されている。IdM クライアントでの sudo の GSSAPI 認証の有効化 を参照してください。
- **/etc/sssd/sssd.conf** ファイルと、**/etc/pam.d/** ディレクトリーの PAM ファイルを変更するための **root** 特権がある。

手順

- 以下のエラーが表示された場合、Kerberos サービスはホスト名をもとに、サービスチケットに合わせて正しいレルムを解決できない可能性があります。

  ```
  Server not found in Kerberos database
  ```

  このような場合は、**/etc/krb5.conf** の Kerberos 設定ファイルの **[domain_realm]** セクションにホスト名を直接追加します。

  ```
  [idm-user@idm-client ~]$ cat /etc/krb5.conf
  ...

  [domain_realm]
   .example.com = EXAMPLE.COM
   example.com = EXAMPLE.COM
   server.example.com = EXAMPLE.COM
  ```

- 以下のエラーが表示される場合には、Kerberos 認証情報がありません。

  ```
  No Kerberos credentials available
  ```

  このような場合は、**kinit** ユーティリティーを使用して Kerberos 認証情報を取得するか、SSSD で認証します。

  ```
  [idm-user@idm-client ~]$ kinit idm-user@IDM.EXAMPLE.COM
  Password for idm-user@idm.example.com:
  ```

- **/var/log/sssd/sssd_pam.log** ログファイルに以下のエラーのいずれかが表示される場合には、Kerberos 認証情報と、現在ログインしたユーザーのユーザー名とが一致しません。

  ```
  User with UPN [<UPN>] was not found.

  UPN [<UPN>] does not match target user [<username>].
  ```

  このような場合は、SSSD で認証されたことを確認するか、**/etc/sssd/sssd.conf** ファイルで **pam_gssapi_check_upn** オプションを無効にすることを検討してください。

  ```
  [idm-user@idm-client ~]$ cat /etc/sssd/sssd.conf
  ...

  pam_gssapi_check_upn = false
  ```

- 他のトラブルシューティングを行う場合は、PAM モジュール **pam_sss_gss.so** のデバッグ出力を有効してください。

  - **/etc/pam.d/sudo** や **/etc/pam.d/sudo-i** など、PAM ファイルに **pam_sss_gss.so** の全エントリーの最後に **debug** オプションを追加します。

    ```
    [root@idm-client ~]# cat /etc/pam.d/sudo
    #%PAM-1.0
    auth       sufficient   pam_sss_gss.so   debug
    auth       include      system-auth
    account    include      system-auth
    password   include      system-auth
    session    include      system-auth
    ```

    ```
    [root@idm-client ~]# cat /etc/pam.d/sudo-i
    #%PAM-1.0
    auth       sufficient   pam_sss_gss.so   debug
    auth       include      sudo
    account    include      sudo
    password   include      sudo
    session    optional     pam_keyinit.so force revoke
    session    include      sudo
    ```

  - **pam_sss_gss.so** モジュールで認証を試み、コンソールの出力を確認します。この例では、ユーザーには Kerberos 認証情報がありません。

    ```
    [idm-user@idm-client ~]$ sudo ls -l /etc/sssd/sssd.conf
    pam_sss_gss: Initializing GSSAPI authentication with SSSD
    pam_sss_gss: Switching euid from 0 to 1366201107
    pam_sss_gss: Trying to establish security context
    pam_sss_gss: SSSD User name: idm-user@idm.example.com
    pam_sss_gss: User domain: idm.example.com
    pam_sss_gss: User principal:
    pam_sss_gss: Target name: host@idm.example.com
    pam_sss_gss: Using ccache: KCM:
    pam_sss_gss: Acquiring credentials, principal name will be derived
    pam_sss_gss: Unable to read credentials from [KCM:] [maj:0xd0000, min:0x96c73ac3]
    pam_sss_gss: GSSAPI: Unspecified GSS failure.  Minor code may provide more
    information
    ```

```
pam_sss_gss: GSSAPI: No credentials cache found
pam_sss_gss: Switching euid from 1366200907 to 0
pam_sss_gss: System error [5]: Input/output error
```

55.11. ANSIBLE PLAYBOOK を使用して IDM クライアントでの IDM ユーザーの SUDO アクセスを確認する

Identity Management (IdM) では、特定の IdM ホストの IdM ユーザーアカウントに **sudo** アクセスが付与されるようにできます。

この手順では、**idm_user_reboot** という名前の **sudo** ルールが存在するように設定します。このルールは、**idmclient** マシンで **/usr/sbin/reboot** コマンドを実行するパーミッションを **idm_user** に付与します。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- IdM に **idm_user** のユーザーアカウントが存在することを確認し、そのユーザーのパスワードを作成してアカウントのロックを解除している。コマンドラインインターフェイスを使用して新しい IdM ユーザーを追加する方法の詳細は、コマンドラインを使用したユーザーの追加 を参照してください。
- **idmclient** にローカルの **idm_user** アカウントがない。(**idm_user** ユーザーは **idmclient** の **/etc/passwd** ファイルに表示されていない)。

手順

1. **inventory.file** などのインベントリーファイルを作成し、そこに **ipaservers** を定義します。

```
[ipaservers]
server.idm.example.com
```

2. **sudo** コマンドを 1 つまたは複数追加します。
   a. **ensure-reboot-sudocmd-is-present.yml** Ansible Playbook を作成し、**sudo** コマンドの IdM データベースに **/usr/sbin/reboot** コマンドが存在するようにします。この手順は、**/usr/share/doc/ansible-freeipa/playbooks/sudocmd/ensure-sudocmd-is-present.yml** ファイルのサンプルをコピーして変更し、簡素化できます。

```
---
- name: Playbook to manage sudo command
```

```
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  # Ensure sudo command is present
  - ipasudocmd:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: /usr/sbin/reboot
      state: present
```

b. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file path_to_playbooks_directory/ensure-
reboot-sudocmd-is-present.yml
```

3. コマンドを参照する **sudo** ルールを作成します。

a. **sudo** コマンドエントリーを使用して sudo ルールが存在することを確認する **ensure-sudorule-for-idmuser-on-idmclient-is-present.yml** Ansible Playbook を作成します。sudo ルールは、 **idm_user** が **idmclient** マシンを再起動することを許可します。この手順は、**/usr/share/doc/ansible-freeipa/playbooks/sudorule/ensure-sudorule-is-present.yml** ファイルのサンプルをコピーして変更し、簡素化できます。

```
---
- name: Tests
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  # Ensure a sudorule is present granting idm_user the permission to run /usr/sbin/reboot
on idmclient
  - ipasudorule:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: idm_user_reboot
      description: A test sudo rule.
      allow_sudocmd: /usr/sbin/reboot
      host: idmclient.idm.example.com
      user: idm_user
      state: present
```

b. Playbook を実行します。

```
$ ansible-playbook -v -i path_to_inventory_directory/inventory.file
path_to_playbooks_directory/ensure-sudorule-for-idmuser-on-idmclient-is-
present.yml
```

検証手順

idm_user が **sudo** を使用して **idmclient** を再起動できることを確認し、IdM サーバーに存在するように設定した **sudo** ルールが **idmclient** で機能することをテストします。サーバーに加えられた変更がクライアントで反映されるまで数分かかる場合があります。

1. idmclient に idm_user としてログインします。
2. **sudo** を使用してマシンを再起動します。プロンプトが表示されたら、idm_user のパスワードを入力します。

```
$ sudo /usr/sbin/reboot
[sudo] password for idm_user:
```

sudo が正しく設定されている場合には、マシンが再起動します。

関連情報

- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-sudocmd.md** ファイル、**README-sudocmdgroup.md** ファイル、および **README-sudorule.md** ファイルを参照してください。

第56章 ホストベースのアクセス制御ルールの設定

ホストベースのアクセス制御 (HBAC) ルールを使用して、Identity Management (IdM) ドメインのアクセス制御を管理できます。HBAC ルールは、どのユーザーまたはユーザーグループが、どのサービスまたはサービスグループ内のサービスを使用して、指定されたホストまたはホストグループにアクセスできるかを定義します。たとえば、HBAC ルールを使用して次の目的を達成できます。

- 指定のユーザーグループのメンバーだけがドメイン内の特定のシステムにアクセスできるように制限する。
- ドメイン内のシステムにアクセスするために特定のサービスのみを使用できます。

デフォルトでは、IdM は **allow_all** という名前のデフォルトの HBAC ルールで設定されます。この設定では、IdM ドメイン全体の関連するサービスをどれでも使用して、すべてのユーザーがすべてのホストに普遍的にアクセスできます。

デフォルトの **allow_all** ルールを独自の HBAC ルールセットに置き換えることで、さまざまなホストへのアクセスを微調整できます。個別のユーザー、ホスト、またはサービスではなく、ユーザーグループ、ホストグループ、またはサービスグループに HBAC ルールを適用して、アクセス制御管理を集中化および簡素化できます。

56.1. WEBUI を使用した IDM ドメインでの HBAC ルールの設定

ホストベースのアクセス制御用にドメインを設定するには、次の手順を実行します。

1. IdM WebUI で HBAC ルールを作成 します。
2. 新しい HBAC ルールをテスト します。
3. デフォルトの **allow_all** HBAC ルールを無効化 します。

注記

カスタム HBAC ルールを作成する前に **allow_all** ルールを無効にしないでください。無効にすると、どのユーザーもホストにアクセスできなくなります。

56.1.1. IdM WebUI での HBAC ルールの作成

IdM WebUI を使用してホストベースのアクセス制御用にドメインを設定するには、以下の手順に従います。この例では、任意のサービスを使用してドメイン内のすべてのシステムにアクセスする権限を、1 人のユーザー **sysadmin** に付与する方法を説明します。

注記

IdM は、ユーザーのプライマリーグループを、IdM グループオブジェクトへのリンクの代わりに、**gidNumber** 属性の数値として保存します。このため、HBAC ルールで参照できるのはユーザーの補助グループだけで、プライマリーグループは参照できません。

前提条件

- ユーザー **sysadmin** が IdM に存在する。

手順

1. **Policy** > **Host-Based Access Control** > **HBAC Rules**を選択します。
2. **Add** をクリックして、新規ルールの追加を開始します。
3. ルールの名前を入力し、**Add and Edit** をクリックして HBAC ルール設定ページを開きます。
4. **Who** エリアで、**Specified Users and Groups** を選択します。次に、**Add** をクリックしてユーザーまたはグループを追加します。
5. **Available** ユーザーのリストから **sysadmin** ユーザーを選択し、**>** をクリックして **Prospective** ユーザーのリストに移動し、**Add** をクリックします。
6. **Accessing** エリアで **Any Host** を選択して、HBAC ルールをすべてのホストに適用します。
7. **Via Service** エリアで **Any Service** を選択して、HBAC ルールをすべてのサービスに適用します。

注記

主なサービスとサービスグループのみが、デフォルトで HBAC ルール用に設定されます。

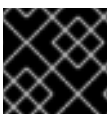
- 現在利用可能なサービスのリストを表示するには、**Policy** > **Host-Based Access Control** > **HBAC Services** を選択します。
- 現在利用可能なサービスグループのリストを表示するには、**Policy** > **Host-Based Access Control** > **HBAC Service Groups** を選択します。

さらにサービスとサービスグループを追加するには、カスタム HBAC サービス用の HBAC サービスエントリーの追加 および HBAC サービスグループの追加 を参照してください。

8. **HBAC ルール** 設定ページで行った変更を保存するには、ページの上部にある **Save** をクリックします。

56.1.2. IdM WebUI での HBAC ルールのテスト

IdM では、シミュレートシナリオを使用して、さまざまな状況で HBAC 設定をテストできます。シミュレートしたテストを実行することで、実稼働環境に HBAC ルールをデプロイする前に、設定ミスやセキュリティーリスクを見つけることができます。



重要

実稼働環境で使用する前に、カスタム HBAC ルールを常にテストしてください。

IdM では、信頼された Active Directory (AD) ユーザーに対する HBAC ルールの影響については検証されない点に注意してください。IdM LDAP ディレクトリーには AD データが保存されないため、IdM は、HBAC シナリオをシミュレートするときに AD ユーザーのグループメンバーシップを解決できません。

手順

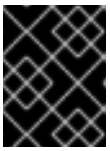
1. **Policy** > **Host-Based Access Control** > **HBAC Test**を選択します。

2. Who ウィンドウで、テスト実行時に使用するアイデンティティーを持つユーザーを指定し、**Next** をクリックします。

3. Accessing ウィンドウで、ユーザーがアクセスするホストを指定し、**Next** をクリックします。

4. Via Service ウィンドウで、ユーザーが使用するサービスを指定し、**Next** をクリックします。

5. Rules ウィンドウで、テストする HBAC ルールを選択し、**Next** をクリックします。ルールを選択しない場合は、すべてのルールがテストされます。
Include Enabled を選択すると、ステータスが **Enabled** であるすべてのルールでテストを実行します。Include Disabled を選択すると、ステータスが **Disabled** であるすべてのルールでテストを実行します。HBAC ルールのステータスを表示および変更するには、**Policy > Host-Based Access Control > HBAC Rules** を選択します。



重要

複数のルールでテストを実行した場合、選択したルールの少なくとも 1 つがアクセスを許可していれば成功となります。

6. Run Test ウィンドウで、**Run Test** をクリックします。

7. テスト結果を確認します。

   - **ACCESS DENIED** と表示された場合、テストでユーザーのアクセスが許可されていないことを示しています。

   - **ACCESS GRANTED** と表示された場合、ユーザーはホストに正常にアクセスできることを示しています。

   デフォルトでは、IdM はテスト結果を表示する際に、テストされている HBAC ルールをすべてリスト表示します。

   - **Matched** を選択すると、正常なアクセスを許可したルールが表示されます。

   - **Unmatched** を選択すると、アクセスを阻止したルールが表示されます。

56.1.3. IdM WebUI での HBAC ルールの無効化

HBAC ルールを無効にすることはできますが、ルールは非アクティブ化されるだけで、削除されません。HBAC ルールを無効にした場合は、後で再度有効にできます。

注記

カスタム HBAC ルールを初めて設定する場合は、HBAC ルールを無効にすると便利です。新しい設定がデフォルトの **low_all** HBAC ルールで上書きされないようにするには、**low_all** を無効にする必要があります。

手順

1. **Policy > Host-Based Access Control > HBAC Rules**を選択します。

2. 無効にする HBAC ルールを選択します。

3. **Disable** をクリックします。

4. **OK** をクリックして、選択した HBAC ルールを無効にすることを確認します。

56.2. CLI を使用した IDM ドメインでの HBAC ルールの設定

ホストベースのアクセス制御用にドメインを設定するには、次の手順を実行します。

1. IdM CLI で HBAC ルールを作成 します。
2. 新しい HBAC ルールをテスト します。
3. デフォルトの **allow_all** HBAC ルールを無効化 します。

注記

カスタム HBAC ルールを作成する前に、**allow_all** ルールを無効にしないでください。カスタムルールを作成する前にこれを無効にすると、すべてのユーザーのすべてのホストへのアクセスが拒否されます。

56.2.1. IdM CLI での HBAC ルールの作成

IdM CLI を使用してホストベースのアクセス制御用にドメインを設定するには、以下の手順に従います。この例では、任意のサービスを使用してドメイン内のすべてのシステムにアクセスする権限を、1 人のユーザー **sysadmin** に付与する方法を説明します。

注記

IdM は、ユーザーのプライマリーグループを、IdM グループオブジェクトへのリンクの代わりに、**gidNumber** 属性の数値として保存します。このため、HBAC ルールで参照できるのはユーザーの補助グループだけで、プライマリーグループは参照できません。

前提条件

- ユーザー **sysadmin** が IdM に存在する。

手順

1. **ipa hbacrule-add** コマンドを使用して、ルールを追加します。

```
$ ipa hbacrule-add
Rule name: rule_name
----------------------------
Added HBAC rule "rule_name"
----------------------------
  Rule name: rule_name
  Enabled: TRUE
```

2. **sysadmin** ユーザーのみに HBAC ルールを適用するには、 **ipa hbacrule-add-user** コマンドを使用します。

```
$ ipa hbacrule-add-user --users=sysadmin
Rule name: rule_name
  Rule name: rule_name
  Enabled: True
  Users: sysadmin
```

```
-------------------------
Number of members added 1
-------------------------
```

注記

すべてのユーザーに HBAC ルールを適用するには、**ipa hbacrule-mod** コマンドを使用して、all ユーザーカテゴリー **--usercat=all** を指定します。HBAC ルールが個々のユーザーまたはグループに関連付けられていると、**ipa hbacrule-mod --usercat=all** は失敗します。この場合は、**ipa hbacrule-remove-user** コマンドを使用して、ユーザーとグループを削除します。

3. ターゲットホストを指定します。すべてのホストに HBAC ルールを適用するには、**ipa hbacrule-mod** コマンドを使用して、all ホストカテゴリーを指定します。

```
$ ipa hbacrule-mod rule_name --hostcat=all
------------------------------
Modified HBAC rule "rule_name"
------------------------------
  Rule name: rule_name
  Host category: all
  Enabled: TRUE
  Users: sysadmin
```

注記

HBAC ルールが個々のホストまたはグループに関連付けられていると、**ipa hbacrule-mod --hostcat=all** は失敗します。この場合は、**ipa hbacrule-remove-host** コマンドを使用して、ホストとグループを削除します。

4. ターゲットの HBAC サービスを指定します。すべてのサービスに HBAC ルールを適用するには、**ipa hbacrule-mod** コマンドを使用して、all サービスカテゴリーを指定します。

```
$ ipa hbacrule-mod rule_name --servicecat=all
------------------------------
Modified HBAC rule "rule_name"
------------------------------
  Rule name: rule_name
  Host category: all
  Service category: all
  Enabled: True
  Users: sysadmin
```

注記

HBAC ルールが個々のサービスまたはグループに関連付けられていると、**ipa hbacrule-mod --servicecat=all** は失敗します。この場合は、**ipa hbacrule-remove-service** コマンドを使用して、サービスとグループを削除します。

検証

- HBAC ルールが正しく追加されたことを確認します。

a. **ipa hbacrule-find** コマンドを使用して、HBAC ルールが IdM に存在することを確認します。

b. **ipa hbacrule-show** コマンドを使用して、HBAC ルールのプロパティーを確認します。

関連情報

- 詳細は、ipa hbacrule-add --help を参照してください。
- カスタム HBAC サービス用の HBAC サービスエントリーの追加 を参照してください。
- HBAC サービスグループの追加 を参照してください。

56.2.2. IdM CLI での HBAC ルールのテスト

IdM では、シミュレートシナリオを使用して、さまざまな状況で HBAC 設定をテストできます。シミュレートしたテストを実行することで、実稼働環境に HBAC ルールをデプロイする前に、設定ミスやセキュリティーリスクを見つけることができます。

実稼働環境で使用する前に、カスタム HBAC ルールを常にテストしてください。

IdM では、信頼された Active Directory (AD) ユーザーに対する HBAC ルールの影響については検証されない点に注意してください。IdM LDAP ディレクトリーには AD データが保存されないため、IdM は、HBAC シナリオをシミュレートするときに AD ユーザーのグループメンバーシップを解決できません。

手順

1. **ipa hbactest** コマンドを使用して、HBAC ルールをテストします。1 つの HBAC ルールをテストする方法と、複数の HBAC ルールをテストする方法があります。

   - 1 つの HBAC ルールをテストするには、以下を実行します。

     ```
     $ ipa hbactest --user=sysadmin --host=server.idm.example.com --service=sudo --
     rules=rule_name
     ----------------------
     Access granted: True
     ----------------------
       Matched rules: rule_name
     ```

   - 複数の HBAC ルールをテストするには、以下を実行します。

     a. **sysadmin** にすべてのホストで **ssh** の使用のみを許可する 2 番目のルールを追加します。

     ```
     $ ipa hbacrule-add --hostcat=all rule2_name
     $ ipa hbacrule-add-user --users sysadmin rule2_name
     $ ipa hbacrule-add-service --hbacsvcs=sshd rule2_name
       Rule name: rule2_name
       Host category: all
       Enabled: True
       Users: admin
       HBAC Services: sshd
     -------------------------
     Number of members added 1
     -------------------------
     ```

b. 次のコマンドを実行して、複数の HBAC ルールをテストします。

```
$ ipa hbactest --user=sysadmin --host=server.idm.example.com --service=sudo --
rules=rule_name --rules=rule2_name
--------------------
Access granted: True
--------------------
  Matched rules: rule_name
  Not matched rules: rule2_name
```

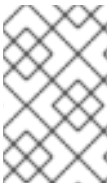
出力において、**Matched rules** には正常なアクセスを許可したルールがリスト表示され、**Not matched rules** にはアクセスを妨げたルールがリスト表示されます。**--rules** オプションを指定しない場合は、すべてのルールが適用されることに注意してください。**--rules** を使用すると、各ルールを個別にテストするのに役立ちます。

関連情報

- 詳細は、**ipa hbactest --help** を参照してください。

56.2.3. IdM CLI での HBAC ルールの無効化

HBAC ルールを無効にすることはできますが、ルールは非アクティブ化されるだけで、削除されません。HBAC ルールを無効にした場合は、後で再度有効にできます。



注記

カスタム HBAC ルールを初めて設定する場合は、HBAC ルールを無効にすると便利です。新しい設定がデフォルトの **low_all** HBAC ルールで上書きされないようにするには、**low_all** を無効にする必要があります。

手順

- **ipa hbacrule-disable** コマンドを使用します。たとえば、**allow_all** ルールを無効にするには、次のコマンドを実行します。

```
$ ipa hbacrule-disable allow_all
-----------------------------
Disabled HBAC rule "allow_all"
-----------------------------
```

関連情報

- 詳細は、**ipa hbacrule-disable --help** を参照してください。

56.3. カスタム HBAC サービス用の HBAC サービスエントリーの追加

主なサービスとサービスグループは、デフォルトで HBAC ルール用に設定されますが、その他のプラグ可能な認証モジュール (PAM) サービスを HBAC サービスとして設定することもできます。これにより、HBAC ルールでカスタム PAM サービスを定義できるようになります。これらの PAM サービスファイルは、RHEL システムの **etc/pam.d** ディレクトリーにあります。

注記

サービスの HBAC サービスとしての追加と、ドメインへのサービスの追加は同じではありません。サービスをドメインに追加すると、ドメイン内の他のリソースでそのサービスを使用できるようにはなりますが、HBAC ルールで使用できるようにはなりません。

56.3.1. IdM WebUI でのカスタム HBAC サービス用の HBAC サービスエントリーの追加

カスタム HBAC サービスエントリーを追加するには、以下で説明する手順に従います。

手順

1. **Policy > Host-Based Access Control > HBAC Services**を選択します。
2. **Add** をクリックして HBAC サービスエントリーを追加します。
3. サービスの名前を入力し、**Add** をクリックします。

56.3.2. IdM CLI でのカスタム HBAC サービスの HBAC サービスエントリーの追加

カスタム HBAC サービスエントリーを追加するには、以下で説明する手順に従います。

手順

- **ipa hbacsvc-add** コマンドを使用します。たとえば、**tftp** サービスのエントリーを追加するには、次のコマンドを実行します。

```
$ ipa hbacsvc-add tftp
-------------------------
Added HBAC service "tftp"
-------------------------
  Service name: tftp
```

関連情報

- 詳細は、**ipa hbacsvc-add --help** を参照してください。

56.4. HBAC サービスグループの追加

HBAC サービスグループにより、HBAC ルールの管理を簡素化できます。たとえば、個々のサービスを HBAC ルールに追加する代わりに、サービスグループ全体を追加できます。

56.4.1. IdM WebUI での HBAC サービスグループの追加

IdM WebUI で HBAC サービスグループを追加するには、以下に概説する手順に従います。

手順

1. **Policy > Host-Based Access Control > HBAC Service Groups**を選択します。
2. **Add** をクリックして HBAC サービスグループを追加します。
3. サービスグループの名前を入力し、**Edit** をクリックします。

4. サービスグループ設定ページで **追加** を選択し、HBAC サービスをグループのメンバーとして追加します。

56.4.2. IdM CLI での HBAC サービスグループの追加

IdM CLI で HBAC サービスグループを追加するには、以下に概説する手順に従います。

手順

1. ターミナルで **ipa hbacsvcgroup-add** コマンドを使用して、HBAC サービスグループを追加します。たとえば、**login** という名前のグループを追加するには、次のコマンドを実行します。

```
$ ipa hbacsvcgroup-add
Service group name: login
------------------------------
Added HBAC service group "login"
------------------------------
  Service group name: login
```

2. **ipa hbacsvcgroup-add-member** コマンドを使用して、HBAC サービスをグループのメンバーとして追加します。たとえば、**sshd** サービスを **login** グループに追加するには、次のコマンドを実行します。

```
$ ipa hbacsvcgroup-add-member
Service group name: login
[member HBAC service]: sshd
  Service group name: login
  Member HBAC service: sshd
------------------------
Number of members added 1
------------------------
```

関連情報

- 詳細は、**ipa hbacsvcgroup-add --help** を参照してください。
- 詳細は、**ipa hbacsvcgroup-add-member --help** を参照してください。

第57章 ANSIBLE PLAYBOOK を使用して IDM にホストベースのアクセス制御ルールを存在させる手順

Ansible は、システムの設定、ソフトウェアのデプロイ、ローリング更新の実行に使用する自動化ツールです。これには、Identity Management (IdM) のサポートが含まれます。

Identity Management (IdM) ホストベースのアクセスポリシーと、Ansible を使用してそれを定義する方法を説明します。

57.1. IDM のホストベースのアクセス制御ルール

ホストベースのアクセス制御 (HBAC) ルールは、サービスグループ内のサービスを使用して、どのユーザーまたはグループがどのホストまたはホストグループにアクセスできるかを定義します。システム管理者は、HBAC ルールを使用して以下の目的を達成できます。

- 指定のユーザーグループのメンバーだけがドメイン内の特定のシステムにアクセスできるように制限する。
- ドメイン内のシステムにアクセスする時に特定のサービスだけを使用できるようにする。

デフォルトでは、IdM は **allow_all** という名前のデフォルトの HBAC ルールで設定されます。この設定では、どのユーザーでも関連のあるサービスをどれでも使用して IdM ドメイン全体にあるすべてのホストに普遍的にアクセスできます。

デフォルトの **allow_all** ルールを独自の HBAC ルールセットに置き換えることで、さまざまなホストへのアクセスを微調整できます。個別のユーザー、ホスト、またはサービスではなく、ユーザーグループ、ホストグループ、またはサービスグループに HBAC ルールを適用して、アクセス制御管理を集中化および簡素化できます。

57.2. ANSIBLE PLAYBOOK を使用して IDM に HBAC ルールを存在させる手順

以下の手順に従って、Ansible Playbook を使用して Identity Management (IdM) にホストベースのアクセス制御 (HBAC) ルールが存在することを確認します。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

- HBAC ルールに使用するユーザーとユーザーグループが IdM に存在する。詳細は、Ansible Playbook を使用したユーザーアカウントの管理 および Ansible Playbook を使用した IdM グループおよびグループメンバーの存在の確保 を参照してください。
- HBAC ルールを適用するホストおよびホストグループが IdM に存在する。詳細は、Ansible Playbook を使用したホストの管理 および Ansible Playbook を使用したホストグループの管理 を参照してください。

手順

1. **inventory.file** などのインベントリーファイルを作成して、そのファイルに **ipaserver** を定義します。

```
[ipaserver]
server.idm.example.com
```

2. Ansible Playbook ファイルを作成して、存在させる HBAC ポリシーを定義します。この手順は、**/usr/share/doc/ansible-freeipa/playbooks/hbacrule/ensure-hbacrule-allhosts-present.yml** ファイルのサンプルをコピーして変更し、簡素化できます。

```
---
- name: Playbook to handle hbacrules
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  # Ensure idm_user can access client.idm.example.com via the sshd service
  - ipahbacrule:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: login
      user: idm_user
      host: client.idm.example.com
      hbacsvc:
      - sshd
      state: present
```

3. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file path_to_playbooks_directory/ensure-new-
hbacrule-present.yml
```

検証手順

1. 管理者として IdM Web UI にログインします。
2. **Policy → Host-Based-Access-Control → HBAC Test** の順に選択します。
3. **Who** タブで idm_user を選択します。
4. **Accessing** タブで **client.idm.example.com** を選択します。
5. **Via service** タブで **sshd** を選択します。

6. **Rules** タブで **login** を選択します。

7. **Run test** タブで **Run test** ボタンをクリックします。ACCESS GRANTED が表示されると、HBAC ルールが正常に実装されています。

関連情報

- **/usr/share/doc/ansible-freeipa** ディレクトリーの **README-hbacsvc.md** ファイル、**README-hbacsvcgroup.md** ファイル、および **README-hbacrule.md** ファイルを参照してください。

- **/usr/share/doc/ansible-freeipa/playbooks** ディレクトリーのサブディレクトリーにある Playbook を参照してください。

第58章 レプリケーショントポロジーの管理

本章では、Identity Management(IdM) ドメイン内のサーバー間のレプリケーションを管理する方法を説明します。

関連情報

- レプリカトポロジーの計画

58.1. レプリカ合意、トポロジー接尾辞、およびトポロジーセグメントの説明

レプリカを作成すると、Identity Management (IdM) が初期サーバーとレプリカ間にレプリカ合意を作成します。複製されるデータはトポロジーの接尾辞に保存され、2 つのレプリカの接尾辞間でレプリカ合意があると、接尾辞がトポロジーセグメントを形成します。これらの概念は、以下のセクションで詳細に説明されています。

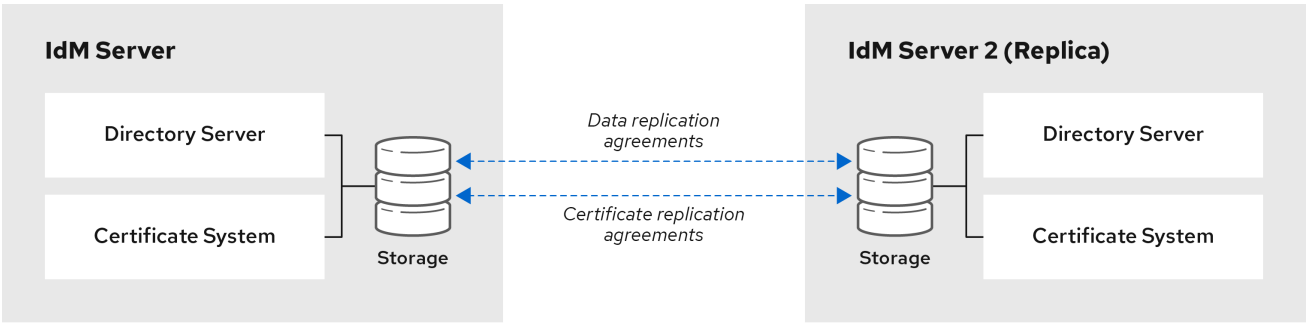
- レプリカ合意
- トポロジー接尾辞
- トポロジーセグメント

58.1.1. IdM レプリカ間のレプリカ合意

管理者が、既存のサーバーに基づいてレプリカを作成すると、Identity Management (IdM) は、初期サーバーとレプリカとの間に **レプリカ合意** を作成します。レプリカ合意は、データと設定が 2 台のサーバー間で継続的に複製されることを保証します。

IdM は、**複数の読み取り/書き込みレプリカ複製** を使用します。この設定では、レプリカ合意に参加しているすべてのレプリカが更新の受信と提供を行うので、サプライヤーとコンシューマーとみなされます。レプリカ合意は常に双方向です。

図58.1 サーバーとレプリカ合意



IdM は、2 種類のレプリカ合意を使用します。

ドメインのレプリカ合意

この合意は、識別情報を複製します。

証明書のレプリカ合意

この合意は、証明書情報を複製します。

両方の複製チャンネルは独立しています。2 台のサーバー間で、いずれかまたは両方の種類のレプリカ合意を設定できます。たとえば、サーバー A とサーバー B にドメインレプリカ合意のみが設定されている場合は、証明書情報ではなく ID 情報だけが複製されます。

58.1.2. トポロジー接尾辞

トポロジー接尾辞は、レプリケートされるデータを保存します。IdM は、**domain** と **ca** の 2 種類のトポロジー接尾辞に対応します。それぞれの接尾辞は、個別のサーバーである個別のレプリケーショントポロジーを表します。

レプリカ合意が設定されると、同じタイプのトポロジー接尾辞を 2 つの異なるサーバーに結合します。

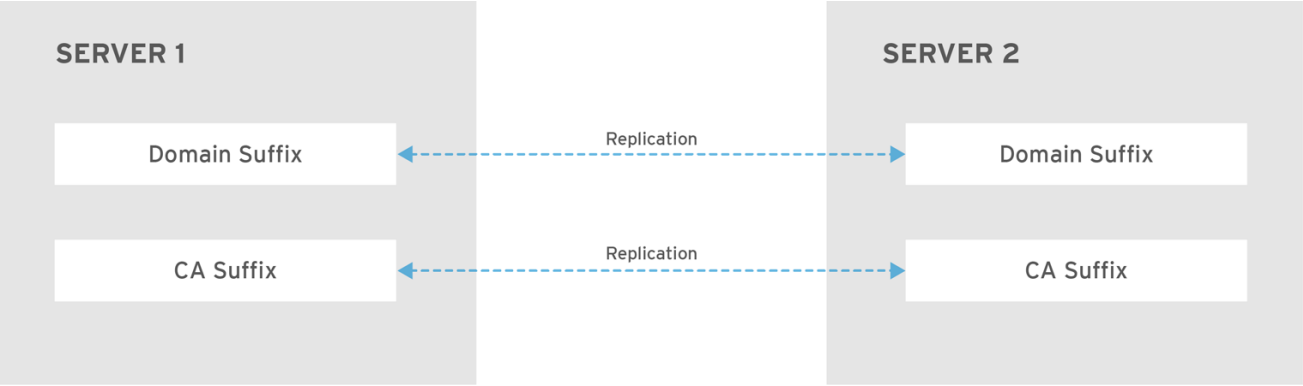
domain 接尾辞: dc=example,dc=com

domain 接尾辞には、ドメイン関連のデータがすべて含まれています。
2 つのレプリカの **domain** 接尾辞間でレプリカ合意が設定されると、ユーザー、グループ、およびポリシーなどのディレクトリーデータが共有されます。

ca 接尾辞: o=ipaca

ca 接尾辞には、Certificate System コンポーネントのデータが含まれます。これは認証局 (CA) がインストールされているサーバーにのみ存在します。
2 つのレプリカの **ca** 接尾辞間でレプリカ合意が設定されると、証明書データが共有されます。

図58.2 トポロジー接尾辞



新規レプリカのインストール時には、**ipa-replica-install** スクリプトが 2 つのサーバー間に初期トポロジーレプリカ合意をセットアップします。

例58.1 トポロジー接尾辞の表示

ipa topologysuffix-find コマンドでトポロジー接尾辞のリストが表示されます。

```
$ ipa topologysuffix-find
----------------------------
2 topology suffixes matched
----------------------------
  Suffix name: ca
  Managed LDAP suffix DN: o=ipaca

  Suffix name: domain
  Managed LDAP suffix DN: dc=example,dc=com
```

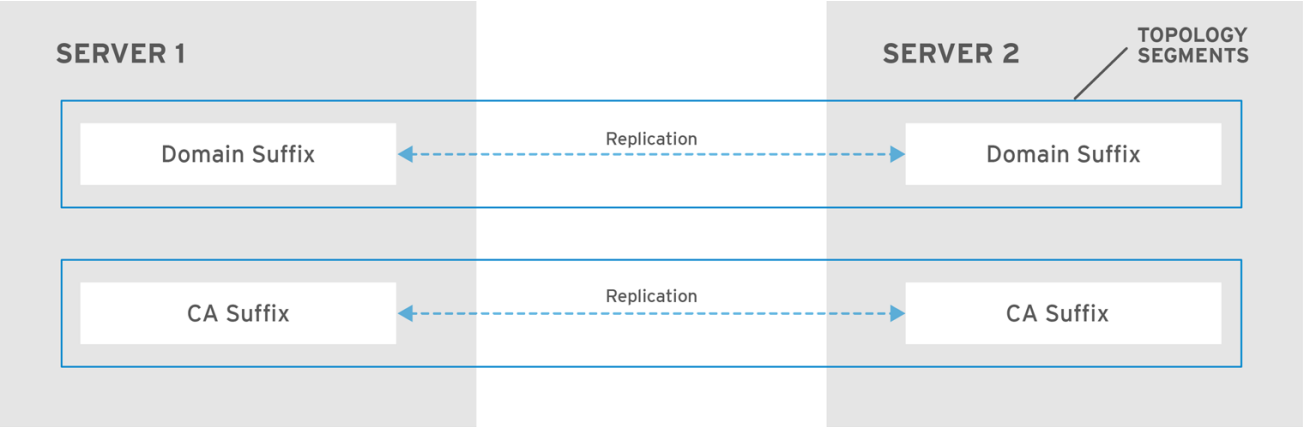
```
----------------------------
Number of entries returned 2
----------------------------
```

58.1.3. トポロジーセグメント

2 つのレプリカの接尾辞間でレプリカ合意があると、接尾辞は **トポロジーセグメント** を形成します。各トポロジーセグメントは、**左ノード**と**右ノード** で設定されます。ノードは、レプリカ合意に参加しているサーバーを表します。

IdM のトポロジーセグメントは常に双方向です。各セグメントは、サーバー A からサーバー B、およびサーバー B からサーバー A への 2 つのレプリカ合意を表します。そのため、データは両方の方向でレプリケートされます。

図58.3 トポロジーセグメント



例58.2 トポロジーセグメントの表示

ipa topologysegment-find コマンドで、ドメインまたは CA 接尾辞に設定されたトポロジーセグメントが表示されます。たとえば、ドメイン接尾辞の場合は、以下のようになります。

```
$ ipa topologysegment-find
Suffix name: domain
-----------------
1 segment matched
-----------------
  Segment name: server1.example.com-to-server2.example.com
  Left node: server1.example.com
  Right node: server2.example.com
  Connectivity: both
----------------------------
Number of entries returned 1
----------------------------
```

この例では、ドメイン関連のデータのみが **server1.example.com** と **server2.example.com** の 2 つのサーバー間で複製されます。

特定セグメントの詳細を表示するには、**ipa topologysegment-show** コマンドを使用します。

```
$ ipa topologysegment-show
```

```
Suffix name: domain
Segment name: server1.example.com-to-server2.example.com
  Segment name: server1.example.com-to-server2.example.com
  Left node: server1.example.com
  Right node: server2.example.com
  Connectivity: both
```

58.2. トポロジーグラフを使用したレプリケーショントポロジーの管理

Web UI のトポロジーグラフは、ドメイン内のサーバー間の関係を表示します。Web UI を使用すると、トポロジーの表現を操作および変換できます。

トポロジーグラフへのアクセス

トポロジーグラフにアクセスするには、以下を実行します。

1. **IPA Server → Topology → Topology Graph** を選択します。
2. トポロジーに加えた変更がグラフに反映されていない場合は、**Refresh** をクリックします。

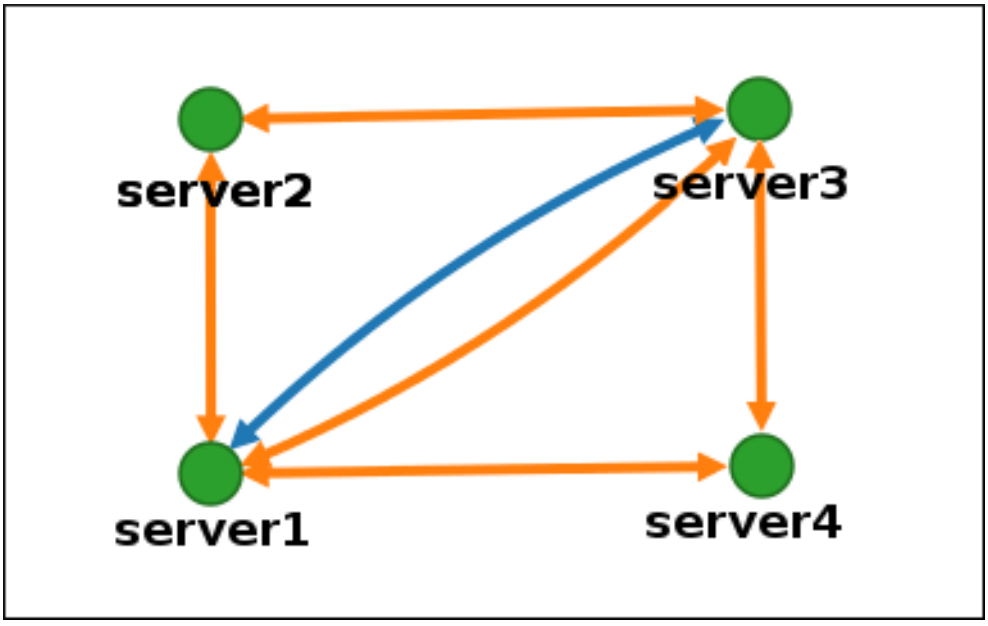
トポロジーグラフの解釈

ドメインのレプリカ合意に参加しているサーバーは、オレンジ色の矢印によって接続されます。CA のレプリカ合意に参加しているサーバーは、青色の矢印によって接続されます。

トポロジーグラフの例: 推奨されるトポロジー

以下の推奨トポロジーの例は、4 台のサーバーに対して考えられる推奨トポロジーの 1 つを示しています。各サーバーは少なくとも 2 つの他のサーバーに接続されており、複数のサーバーが CA サーバーです。

図58.4 推奨されるトポロジーの例

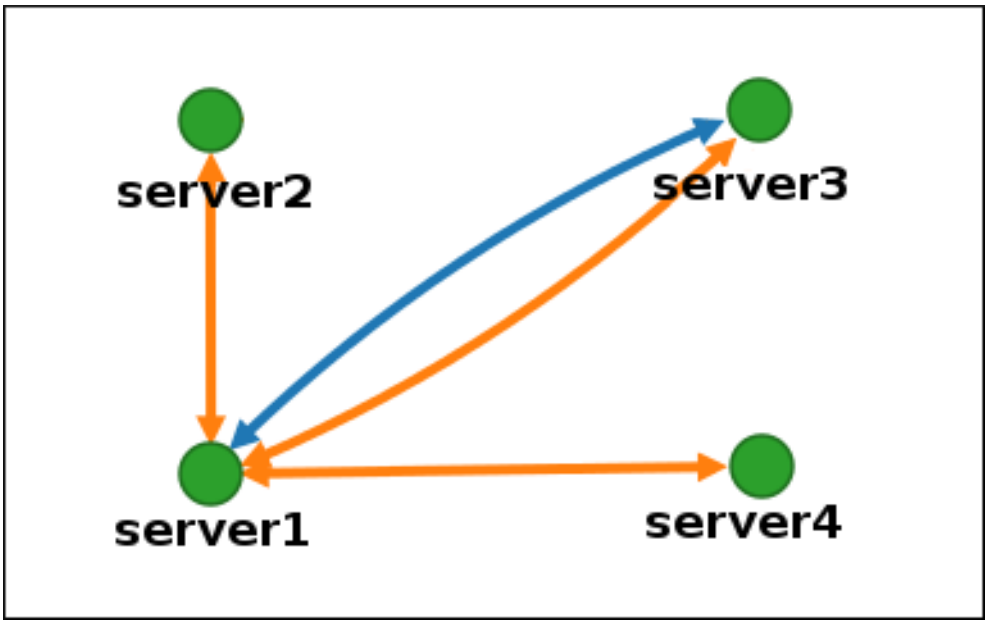


トポロジーグラフの例: 推奨されないトポロジー

推奨されないトポロジーの例では、**server1** が単一障害点になります。その他のすべてのサーバーは、このサーバーとのレプリカ合意がありますが、他のサーバーとは合意がありません。したがって、**server1** が失敗すると、他のすべてのサーバーは分離されます。

このようなトポロジーの作成は避けてください。

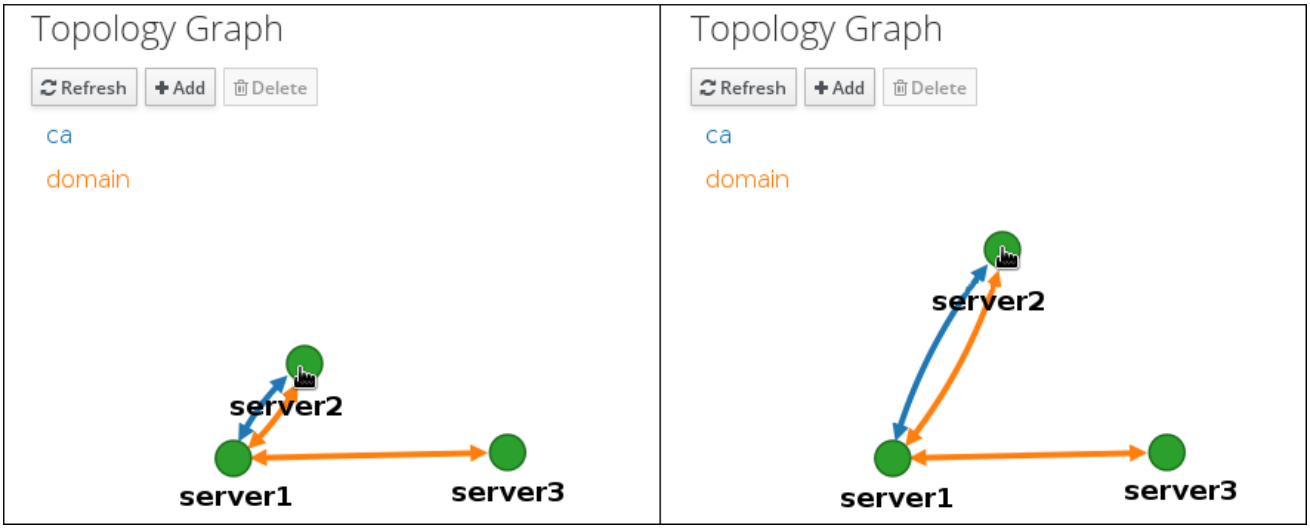
図58.5 推奨されないトポロジーの例: 単一障害点



トポロジービューのカスタマイズ

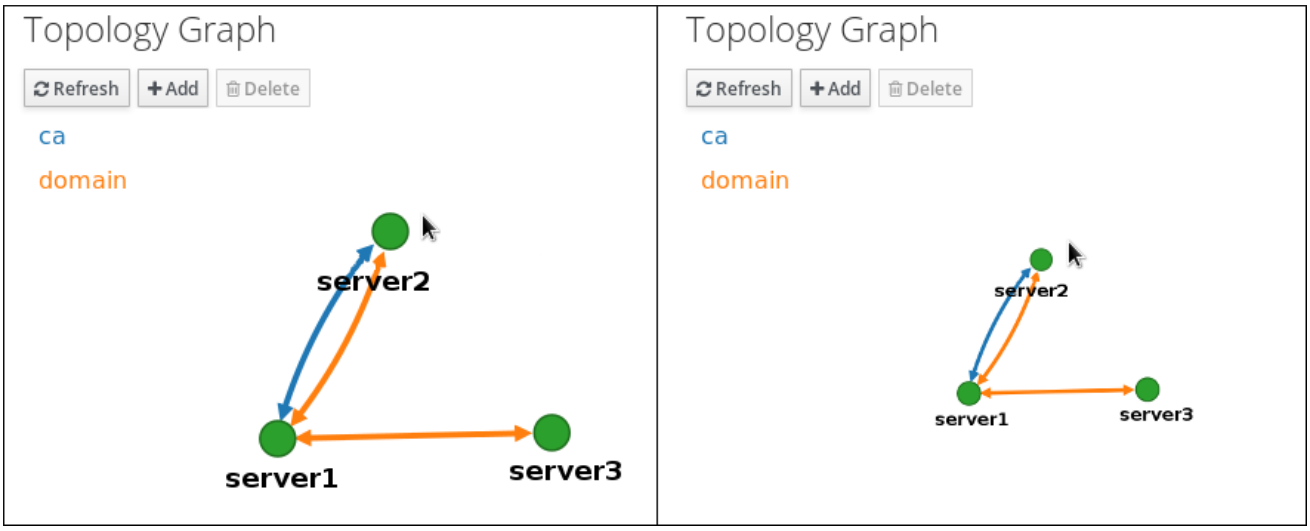
マウスをドラッグして、個別のトポロジーノードを移動できます。

図58.6 トポロジーグラフのノードの移動



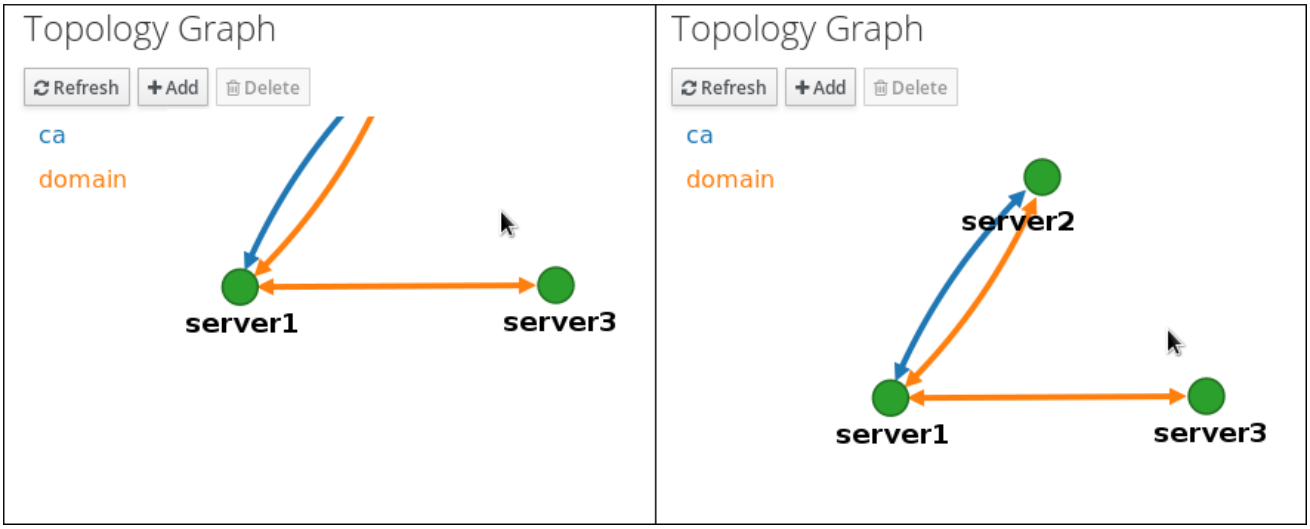
マウスのホイールを使用して、トポロジーグラフを拡大および縮小できます。

図58.7 トポロジーグラフのズーム



マウスの左ボタンを保持することで、トポロジーグラフのキャンバスを移動できます。

図58.8 トポロジーグラフのキャンバスの移動



58.3. WEB UI を使用した 2 台のサーバー間のレプリケーションの設定

Identity Management (IdM) の Web インターフェイスを使用すると、2 つのサーバーを選択し、そのサーバー間に新しいレプリカ合意を作成できます。

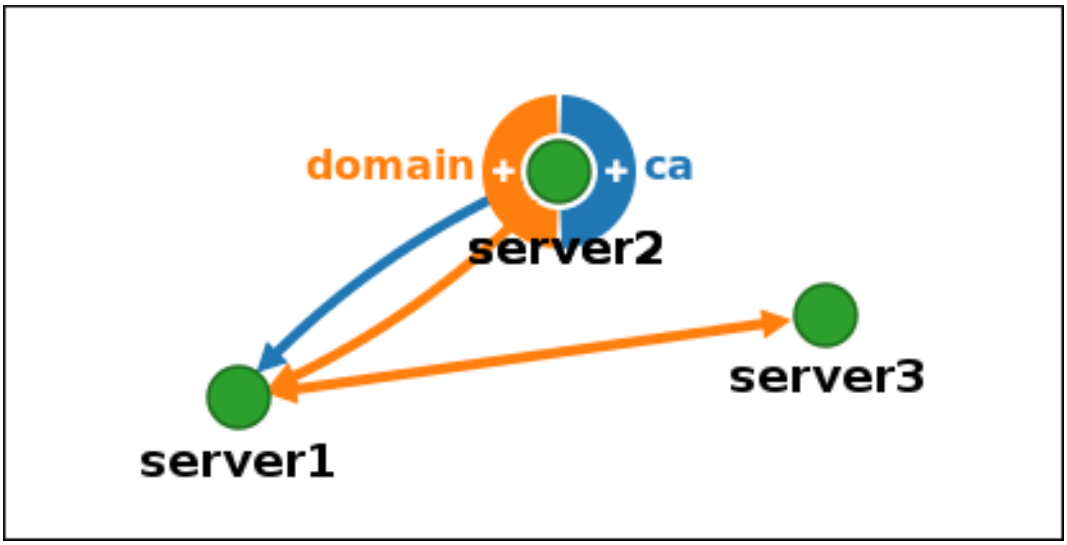
前提条件

- IdM 管理者認証情報がある。

手順

1. トポロジーグラフで、サーバーノードの 1 つにマウスを合わせます。

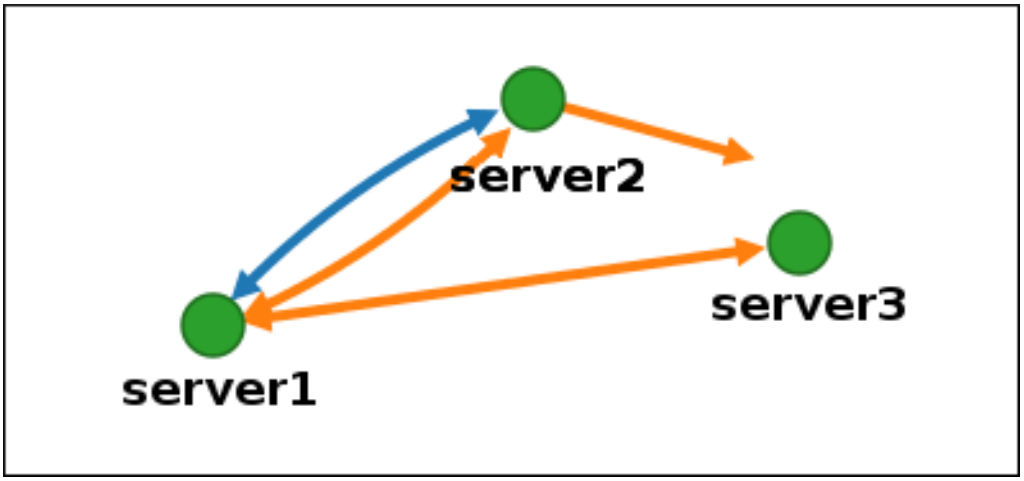
図58.9 ドメインまたは CA オプション



2. 作成するトポロジーセグメントのタイプに応じて、**domain** または円の **ca** 部分をクリックします。

3. 新しいレプリカ合意を表す新しい矢印が、マウスポインターの下に表示されます。マウスを他のサーバーノードに移動し、そこでクリックします。

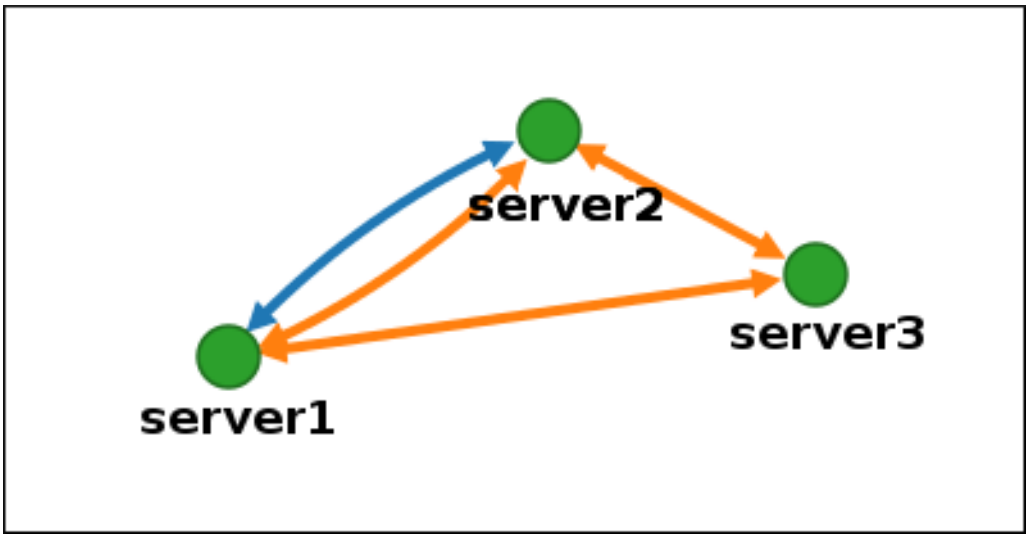
図58.10 新規セグメントの作成



4. **Add Topology Segment** ウィンドウで **Add** をクリックして、新規セグメントのプロパティーを確認します。

2 台のサーバー間の新しいトポロジーセグメントは、サーバーをレプリカ合意に参加させます。トポロジーグラフには、更新されたレプリケーショントポロジーが表示されるようになりました。

図58.11 新規に作成されたセグメント



58.4. WEB UI を使用した 2 台のサーバー間のレプリケーションの停止

Identity Management (IdM) の Web インターフェイスを使用して、サーバーからレプリカ合意を削除できます。

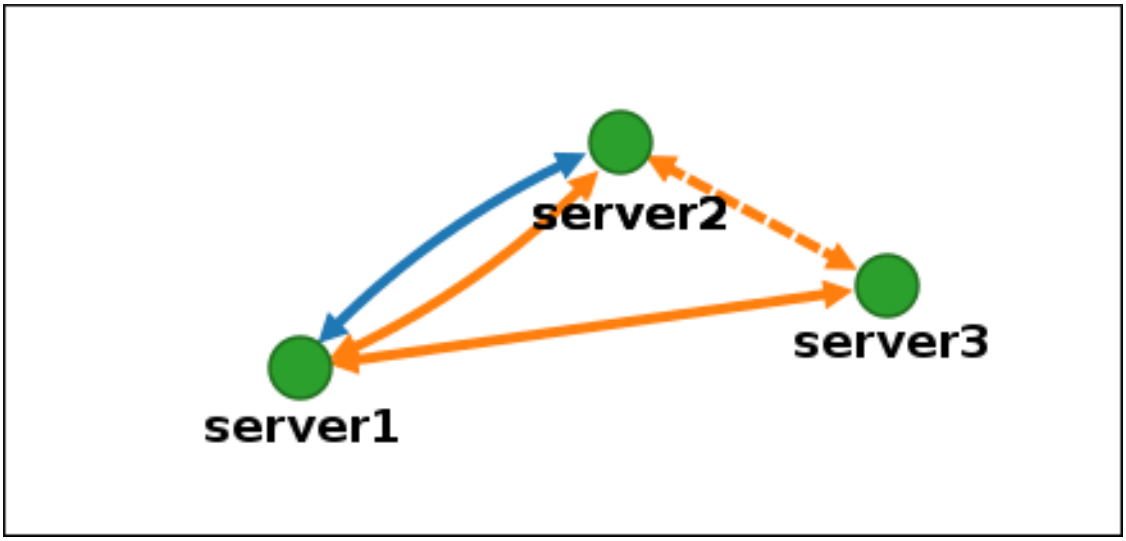
前提条件

- IdM 管理者認証情報がある。

手順

1. 削除するレプリカ合意を表す矢印をクリックします。これにより、矢印がハイライト表示されます。

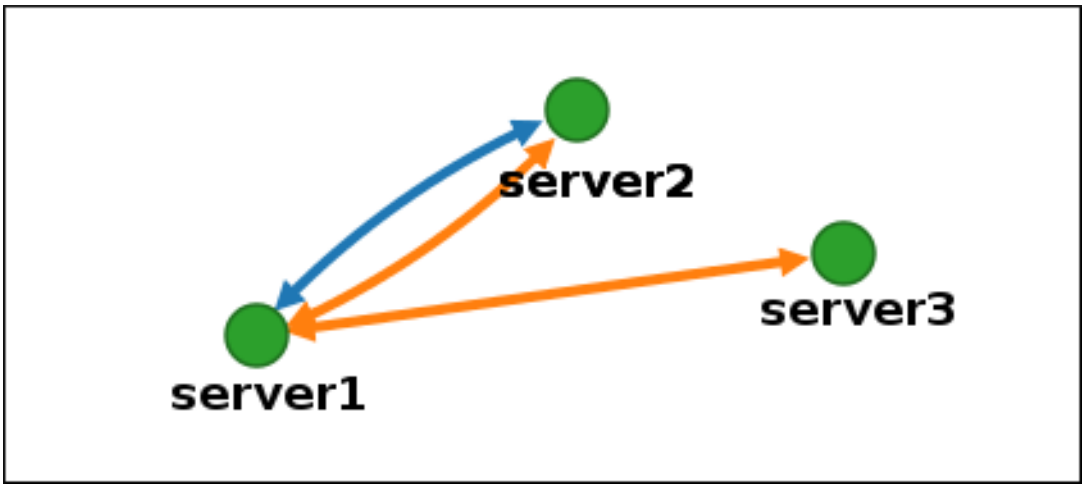
   図58.12 トポロジーセグメントのハイライト表示

2. **Delete** をクリックします。
3. **Confirmation** ウィンドウで **OK** をクリックします。

IdM は、2 台のサーバー間のトポロジーセグメントを削除します。これにより、そのレプリカ合意が削除されます。トポロジーグラフには、更新されたレプリケーショントポロジーが表示されるようになりました。

図58.13 トポロジーセグメントの削除



58.5. CLI を使用した 2 つのサーバー間のレプリケーションの設定

ipa topologysegment-add コマンドを使用して、2 台のサーバー間のレプリカ合意を設定できます。

前提条件

- IdM 管理者認証情報がある。

手順

1. **ipa topologysegment-add** コマンドを使用して、2 つのサーバーのトポロジーセグメントを作成します。プロンプトが表示されたら、以下を指定します。
   - 必要なトポロジー接尾辞: **domain** または **ca**
   - 2 つのサーバーを表す、左ノードと右のノード
   - オプションで、セグメントのカスタム名
     以下に例を示します。

     ```
     $ ipa topologysegment-add
     Suffix name: domain
     Left node: server1.example.com
     Right node: server2.example.com
     Segment name [server1.example.com-to-server2.example.com]: new_segment
     ----------------------------
     Added segment "new_segment"
     ----------------------------
       Segment name: new_segment
       Left node: server1.example.com
       Right node: server2.example.com
       Connectivity: both
     ```

     新しいセグメントを追加すると、サーバーをレプリカ合意に参加させます。

2. **オプション:ipa topologysegment-show** コマンドを使用して、新しいセグメントが設定されたことを確認します。

   ```
   $ ipa topologysegment-show
   ```

```
Suffix name: domain
Segment name: new_segment
  Segment name: new_segment
  Left node: server1.example.com
  Right node: server2.example.com
  Connectivity: both
```

58.6. CLI を使用した 2 つのサーバー間のレプリケーションの停止

ipa topology_segment-del コマンドを使用して、コマンドラインからレプリカ合意を終了できます。

前提条件

- IdM 管理者認証情報がある。

手順

1. レプリケーションを停止するには、サーバー間の対応するレプリケーションセグメントを削除する必要があります。これを実行するには、セグメント名を知っている必要があります。
名前が分からない場合は、**ipa topologysegment-find** コマンドを使用してすべてのセグメントを表示し、出力で必要なセグメントを見つけます。プロンプトが表示されたら、必要なトポロジー接尾辞 (**domain** または **ca**) を指定します。以下に例を示します。

```
$ ipa topologysegment-find
Suffix name: domain
------------------
8 segments matched
------------------
  Segment name: new_segment
  Left node: server1.example.com
  Right node: server2.example.com
  Connectivity: both

...

----------------------------
Number of entries returned 8
----------------------------
```

2. **ipa topologysegment-del** コマンドを使用して、2 台のサーバー間のトポロジーセグメントを削除します。

```
$ ipa topologysegment-del
Suffix name: domain
Segment name: new_segment
----------------------------
Deleted segment "new_segment"
----------------------------
```

セグメントを削除すると、レプリカ合意が削除されます。

3. **オプション:ipa topologysegment-find** コマンドを使用して、セグメントが表示されなくなったことを確認します。

```
$ ipa topologysegment-find
Suffix name: domain
------------------
7 segments matched
------------------
  Segment name: server2.example.com-to-server3.example.com
  Left node: server2.example.com
  Right node: server3.example.com
  Connectivity: both

...

----------------------------
Number of entries returned 7
----------------------------
```

58.7. WEB UI を使用したトポロジーからのサーバーの削除

Identity Management (IdM) の Web インターフェイスを使用して、トポロジーからサーバーを削除できます。

前提条件

- IdM 管理者認証情報がある。
- 削除するサーバーが、残りのトポロジーで他のサーバーに接続する **唯一のサーバーではない**。この場合、他のサーバーが分離されますが、これは許可されていません。
- 削除するサーバーが、最後の CA または DNS サーバー **ではない**。

警告

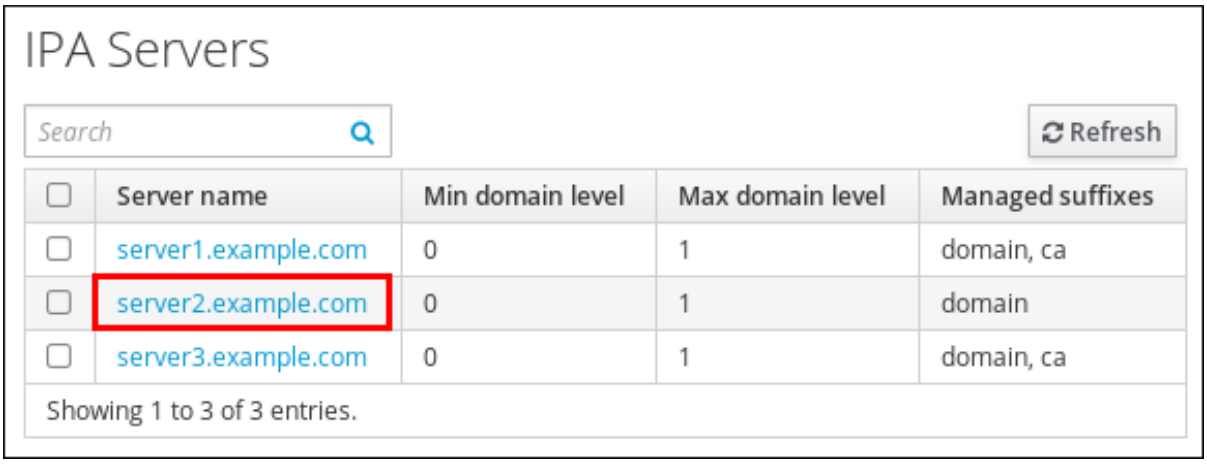
サーバーの削除は元に戻せないアクションです。サーバーを削除すると、トポロジーに戻す唯一の方法は、マシンに新しいレプリカをインストールすることです。

手順

サーバーコンポーネントをマシンからアンインストールせずにトポロジーからサーバーを削除するには、以下を実行します。

1. **IPA Server → Topology → IPA Servers** を選択します。
2. 削除するサーバーの名前をクリックします。

図58.14 サーバーの選択



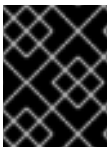
3. **Delete Server** をクリックします。

58.8. CLI を使用したトポロジーからのサーバーの削除

コマンドラインインターフェイスを使用して、トポロジーからサーバーを削除できます。

前提条件

- IdM 管理者認証情報がある。
- 削除するサーバーが、残りのトポロジーで他のサーバーに接続する **唯一のサーバーではない**。この場合、他のサーバーが分離されますが、これは許可されていません。
- 削除するサーバーが、最後の CA または DNS サーバー **ではない**。



重要

サーバーの削除は元に戻せないアクションです。サーバーを削除すると、トポロジーに戻す唯一の方法は、マシンに新しいレプリカをインストールすることです。

手順

server1.example.com を削除するには、次のコマンドを実行します。

1. 別のサーバーで **ipa server-del** コマンドを実行して、**server1.example.com** を削除します。このコマンドは、サーバーを参照するすべてのトポロジーセグメントを削除します。

```
[user@server2 ~]$ ipa server-del
Server name: server1.example.com
Removing server1.example.com from replication topology, please wait...
-----------------------------------------------------------
Deleted IPA server "server1.example.com"
-----------------------------------------------------------
```

2. **オプション**: **server1.example.com** で、**ipa server-install --uninstall** コマンドを実行して、マシンからサーバーコンポーネントをアンインストールします。

```
[root@server1 ~]# ipa server-install --uninstall
```

58.9. WEB UI を使用した IDM サーバーでのサーバーロールの表示

IdM サーバーにインストールされるサービスに基づいて、さまざまな **サーバーロール** を実行できます。以下に例を示します。

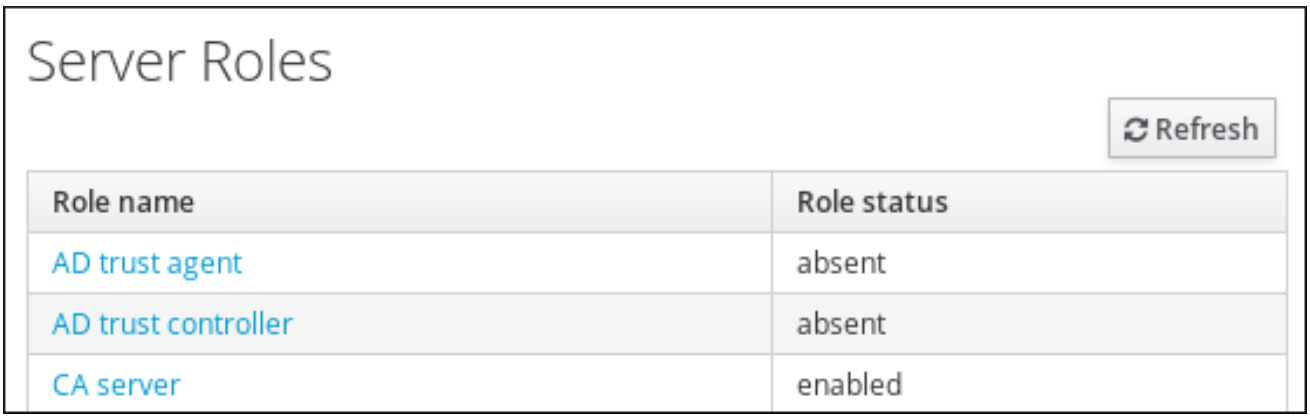
- CA サーバー
- DNS サーバー
- キーリカバリー認証局 (KRA) サーバー

サポートされるサーバーロールの完全なリストは、**IPA Server → Topology → Server Roles**を参照してください。

注記

- Role status が **absent** の場合は、トポロジー内でそのロールを実行しているサーバーがないことを示しています。
- Role status が **enabled** の場合は、トポロジー内でそのロールを実行しているサーバーが 1 台以上あることを示しています。

図58.15 Web UI でのサーバーロール



58.10. CLI を使用した IDM サーバーでのサーバーロールの表示

IdM サーバーにインストールされるサービスに基づいて、さまざまな **サーバーロール** を実行できます。以下に例を示します。

- CA サーバー
- DNS サーバー
- キーリカバリー認証局 (KRA) サーバー

以下のコマンドを使用して、トポロジー内でどのサーバーがどのロールを実行するかを表示できます。

- **ipa config-show** コマンドを実行すると、すべての CA サーバーおよび現行の CA 更新サーバーが表示されます。

```
$ ipa config-show
  ...
  IPA masters: server1.example.com, server2.example.com, server3.example.com
```

```
IPA CA servers: server1.example.com, server2.example.com
IPA CA renewal master: server1.example.com
```

- **ipa server-show** コマンドは、特定のサーバーで有効なロールのリストを表示します。たとえば、**server.example.com** で有効にしたロールのリストは、以下のようになります。

```
$ ipa server-show
Server name: server.example.com
  ...
Enabled server roles: CA server, DNS server, KRA server
```

- **ipa server-find --servrole** は、特定のサーバーロールが有効になっているすべてのサーバーを検索します。たとえば、すべての CA サーバーを検索するには、以下を実行します。

```
$ ipa server-find --servrole "CA server"
---------------------
2 IPA servers matched
---------------------
  Server name: server1.example.com
  ...

  Server name: server2.example.com
  ...
----------------------------
Number of entries returned 2
----------------------------
```

58.11. レプリカの CA 更新サーバーおよび CRL パブリッシャーサーバーへのプロモート

IdM デプロイメントで組み込み認証局 (CA) を使用する場合は、IdM CA サーバーの 1 つが CA サブシステム証明書の更新を管理する CA 更新サーバーとして機能します。IdM CA サーバーの 1 つは、証明書失効リストを生成する IdM CRL パブリッシャーサーバーとしても機能します。デフォルトでは、CA 更新サーバーおよび CRL パブリッシャーサーバーロールは、システム管理者が **ipa-server-install** または **ipa-ca-install** コマンドを使用して CA ロールをインストールした最初のサーバーにインストールされます。

前提条件

- IdM 管理者認証情報がある。

手順

- 現在の CA 更新サーバーを変更します。
- CRL を生成するようにレプリカを設定します。

58.12. 非表示レプリカの降格または昇格

レプリカのインストール後、レプリカの表示状態を設定できます。

非表示のレプリカの詳細は、非表示のレプリカモード を参照してください。

レプリカが CA 更新サーバーである場合は、このレプリカを非表示にする前に、サービスを別のレプリカに移動します。

詳細は IdM CA 更新サーバーの変更およびリセット を参照してください。

手順

- レプリカを非表示にするには、次のコマンドを実行します。

```
# ipa server-state replica.idm.example.com --state=hidden
```

次のコマンドを実行すれば、レプリカを表示できます

```
# ipa server-state replica.idm.example.com --state=enabled
```

トポロジー内のすべての非表示のレプリカのリストを表示するには、次のコマンドを実行します。

```
# ipa config-show
```

すべてのレプリカが有効になっている場合は、コマンドの出力に非表示のレプリカは記載されません。

第59章 IDENTITY MANAGEMENT の公開鍵証明書

X.509 公開鍵証明書は、Identity Management (IdM)のユーザー、ホスト、およびサービスを認証するために使用されます。X.509 証明書は、認証のほかに、デジタル署名と暗号化も可能にし、プライバシー、完全性、否認不可を実現します。

証明書には、以下の情報が含まれます。

- 証明書が認証する発行先
- 証明書を署名した CA の発行元
- 証明書の有効性の開始日と終了日
- 証明書の有効な用途
- 発行先の公開鍵

公開鍵により暗号化したメッセージは、対応した秘密鍵により複号できます。証明書および、勝目所に含まれる公開鍵は、公開できますが、ユーザー、ホスト、またはサービスは、対応の秘密鍵を秘密にしておく必要があります。

59.1. IDM の認証局

認証局は信頼の階層で機能します。内部認証局 (CA) がある IdM 環境では、CA が署名している正しい証明書を、すべての IdM ホスト、ユーザー、およびサービスが信頼します。このルート CA とは別に、IdM は、ルート CA から証明書に署名する権限を付与されたサブ CA にも対応します。多くの場合、このようなサブ CA が署名できる証明書は、VPN 証明書 など、特定の種類の証明書です。最後に、IdM は外部 CA の使用に対応します。以下 の表は、IdM の各種 CA の用途に関する詳細を紹介します。

表59.1 IdM での統合および外部 CA の用途の比較

CA の名前	説明	用途	関連リンク
ipa CA	Dogtag アップストリームプロジェクトをベースとする統合 CA	統合 CA は、ユーザー、ホスト、およびサービスの証明書の作成、取り消し、および発行が可能です。	ipa CA を使用した新規ユーザー証明書の要求およびクライアントへのエクスポート
IdM サブ CA	**ipa** CA の下位となる統合 CA	IdM サブ CA は、**ipa** CA が証明書の署名を許可した CA です。多くの場合に、これらの証明書は、VPN 証明書など、特定の種類の証明書です。	証明書のサブセットだけに信頼するアプリケーションを制限する手順
外部 CA	外部 CA は、統合 IdM CA またはそのサブ CA 以外の CA です。	IdM ツールを使用して、これらの CA が発行する証明書をユーザー、サービス、またはホストに対して追加し、削除します。	IdM ユーザー、ホスト、およびサービスの外部署名証明書の管理

証明書の観点からは、自己署名 IdM CA と、外部署名の間に違いがありません。

CA のロールの目的は以下のとおりです。

- デジタル証明書を発行する。
- 証明書を署名して、証明書に名前のある発行先が公開鍵を所有することを認定する。発行先は、ユーザー、ホスト、またはサービスのいずれかです。
- 証明書をキャンセルして、証明書失効リスト (CRL) および Online Certificate Status Protocol (OCSP) で失効ステータスを提供できる。

関連情報

- CA サービスの計画 を参照してください。

59.2. 証明書および KERBEROS の比較

証明書は、Kerberos チケットが実行したのと同様の機能を実行します。Kerberos は、セキュアでないネットワーク上で通信するノードが、安全な方法で互いに ID を証明できるように、チケットベースで機能するコンピューターネットワーク認証プロトコルです。以下の表では、Kerberos および X.509 の証明書を比較します。

表59.2 証明書および Kerberos の比較

特徴	Kerberos	X.509
認証	はい	はい
プライバシー	任意	はい
インテグリティー	任意	はい
関係する暗号の種類	対称	非対称
デフォルトの有効性	短い (1 日)	長い (2 年)

デフォルトでは、Identity Management の Kerberos は、通信相手の ID のみを保証します。

59.3. IDM でユーザーを認証する証明書を使用する利点と問題点

IdM でユーザーを認証する証明書を使用する利点は次のとおりです。

- 通常、スマートカードの秘密鍵を保護する PIN は、通常のパスワードよりも簡単で覚えやすいものになります。
- デバイスによっては、スマートカードに保存されている秘密鍵をエクスポートできません。これによりセキュリティーが強化されます。
- スマートカードはログアウトを自動化できます。IdM は、ユーザーがリーダーからスマートカードを取り外すと、ユーザーをログアウトするように設定できます。
- 秘密鍵を盗むには、スマートカードへの物理的なアクセスが必要で、スマートカードはハッキング攻撃に対して安全になります。

- スマートカード認証は 2 要素認証の一例です。つまり、所有しているもの (カード) と、知っているもの (PIN) の両方が必要です。
- スマートカードは、電子メールの暗号化など、他の目的に使用できる鍵を提供するため、パスワードよりも柔軟性があります。
- IdM クライアントである共有マシンでスマートカードを使用しても、通常、システム管理者に対して追加で発生する問題はありません。実際、スマートカード認証は共有マシンにとって理想的な選択肢です。

IdM のユーザーを認証する証明書を使用する問題点は次のとおりです。

- スマートカードまたは証明書を紛失したり忘れたりすることで、実質的にロックアウトされる可能性があります。
- PIN を複数回誤入力すると、カードがロックされる可能性があります。
- 一般に、セキュリティー担当者や承認者などによる要求と承認の間には中間ステップがあります。セキュリティー担当者または管理者が、IdM で **ipa cert-request** コマンドを実行する必要があります。
- スマートカードとリーダーは、ベンダーとドライバーに固有である傾向があります。多くのリーダーはさまざまなカードに使用できますが、一部のベンダーのスマートカードは、別のベンダーのリーダーや、その目的で設計されていないリーダーでは機能しない場合があります。
- 証明書とスマートカードの場合は、管理者が短期間で習得しなければならない内容が多くなります。

第60章 IDM と機能する証明書形式への変換

このユーザーストーリーでは、IdM システム管理者が、特定の IdM コマンドで証明書の正しい形式を使用するようにする方法を説明します。これは、たとえば以下の状況で役に立ちます。

- ユーザープロファイルに外部証明書を読み込んでいる。詳細は、IdM ユーザーアカウントに読み込む外部証明書の変換 を参照してください。
- IdM サーバーをスマートカード認証用に設定する場合、または IdM クライアントをスマートカード認証用に設定する場合 に、外部 CA 証明書を使用して、ユーザーが外部認証局から発行された証明書が含まれるスマートカードで IdM を認証できるようにしている。
- NSS データベースから、証明書と秘密鍵の両方を含む pkcs #12 形式で、証明書をエクスポートしている。詳細は、NSS データベースから PKCS #12 ファイルへの証明書と秘密鍵のエクスポート を参照してください。

60.1. IDM での証明書の形式およびエンコード

IdM におけるスマートカード認証を含む証明書認証は、ユーザーが提示する証明書と、ユーザーの IdM プロファイルに保存されている証明書または証明書データを比較することによって進められます。

システムの設定

IdM プロファイルに格納されるものは証明書のみで、対応する秘密鍵ではありません。また、認証中に、ユーザーが、対応する秘密鍵の所有していることを表示する必要があります。ユーザーは、証明書と秘密鍵の両方が含まれる PKCS #12 ファイル、またはこれら 2 つのファイル (証明書が含まれるファイルと、秘密鍵が含まれているファイル) のいずれかを提示して行います。

したがって、ユーザープロファイルに証明書を読み込むなどのプロセスでは、秘密鍵を含まない証明書ファイルのみが使用できます。

同様に、システム管理者が外部の CA 証明書を提供している場合は、パブリックデータ (秘密鍵がない証明書) のみを提供します。スマートカード認証用の IdM サーバーまたは IdM クライアントを設定する **ipa-advise** ユーティリティーでは、外部 CA の証明書が含まれる入力ファイルが必要ですが、秘密鍵は必要ありません。

証明書のエンコーディング

2 つの一般的な証明書エンコーディング **PEM** (Privacy-enhanced Electronic Mail) および **DER** (Distinguished Encoding Rules) があります。**base64** 形式は **PEM** 形式とほぼ同じですが、ヘッダーおよびフッター (**-----BEGIN CERTIFICATE----- および -----END CERTIFICATE-----**) は含まれません。

DER を使用してエンコードされた証明書は、バイナリーファイルの X509 デジタル証明書です。証明書はバイナリーファイルで、人間は判読できません。**DER** ファイルは、ファイル名の拡張子 **.der** を使用することもありますが、ファイル名の拡張子 **.crt** および **.cer** が **DER** 証明書が含まれることもあります。鍵を含む **DER** ファイルの名前は **.key** です。

PEM Base64 を使用してエンコードされた証明書は、人間が判読できるファイルです。このファイルには、-----BEGIN ...の行頭に付けられた ASCII (Base64) のデータが含まれます。**PEM** ファイルは、**.pem** ファイル拡張子を使用することもありますが、ファイル名の拡張子 **.crt** および **.cer** に **PEM** 証明書が含まれる場合もあります。鍵を含む **PEM** ファイルの名前は **.key** です。

ipa コマンドには、許可される証明書の種類に、さまざまな制限があります。たとえば、**ipa user-add-cert** コマンドでは、**base64** 形式でエンコードされた証明書のみが使用できますが、**ipa-server-certinstall** は、**PEM**、**DER**、**PKCS #7**、**PKCS #8** および **PKCS #12** の証明書が使用できます。

表60.1 証明書のエンコーディング

エンコーディング形式	人間が判別可能	一般的なファイル名の拡張子	エンコーディング形式を使用できる IdM コマンドの例
PEM/base64	はい	.pem、.crt、.cer	ipa user-add-cert、ipa-server-certinstall など
DER	いいえ	.der、.crt、.cer	ipa-server-certinstall など

IdM の証明書関連のコマンドおよび形式 は、コマンドが受け入れる証明書形式を含むその他の **ipa** コマンドをリスト表示します。

ユーザー認証

ブラウザーのデータベースに秘密鍵と証明書の両方を保存することで、Web UI を使用して IdM にアクセスすると、証明書に対応する秘密鍵をユーザーが所有していることを証明します。

CLI を使用して IdM にアクセスすると、以下のいずれかの方法で、証明書に対応する秘密鍵をユーザーが所有していることを証明します。

- ユーザーは、**kinit -X** コマンドの **X509_user_identity** パラメーターの値として、証明書と鍵の両方が含まれるスマートカードに接続するスマートカードモジュールへのパスを追加します。

```
$ kinit -X X509_user_identity='PKCS11:opensc-pkcs11.so' idm_user
```

- ユーザーは、**kinit -X** コマンドの **X509_user_identity** パラメーターの値として、証明書が含まれるファイルと、秘密鍵が含まれるファイルの 2 つを追加します。

```
$ kinit -X X509_user_identity='FILE:`/path/to/cert.pem,/path/to/cert.key`' idm_user
```

便利な証明書コマンド

証明書データ (発行先や発行者など) を表示するには、次のコマンドを実行します。

```
$ openssl x509 -noout -text -in ca.pem
```

2 つの証明書で、異なる行を比較するには、次のコマンドを実行します。

```
$ diff cert1.crt cert2.crt
```

2 つの証明書の出力を 2 列で表示して、2 つの証明書で異なる行を比較するには、次のコマンドを実行します。

```
$ diff cert1.crt cert2.crt -y
```

60.2. IDM ユーザーアカウントに読み込む外部証明書の変換

このセクションでは、外部証明書がユーザーエントリーに追加される前に、適切にエンコードされおよび形式が正しいことを確認する方法を説明します。

60.2.1. 前提条件

- 証明書が Active Directory 認証局によって発行され、**PEM** エンコーディングを使用する場合は、**PEM** ファイルが **UNIX** 形式に変換されていることを確認します。ファイルを変換するには、dos2unix パッケージが提供する **dos2unix** ユーティリティーを使用します。

60.2.2. IdM CLI での外部証明書の変換および IdM ユーザーアカウントへの読み込み

IdM CLI では、最初の行および最後の行 (-----BEGIN CERTIFICATE----- および -----END CERTIFICATE-----) が削除された **PEM** 証明書のみが使用できます。

以下の手順に従って、外部証明書を **PEM** 形式に変換し、IdM CLI を使用して IdM ユーザーアカウントに追加します。

手順

1. 証明書を **PEM** 形式に変換します。
   - 証明書が **DER** 形式の場合は、次のようになります。

     ```
     $ openssl x509 -in cert.crt -inform der -outform pem -out cert.pem
     ```

   - ファイルが **PKCS #12** 形式で、共通のファイル名の拡張子が **.pfx** および **.p12** で、証明書、秘密鍵、およびその他のデータが含まれている場合は、**openssl pkcs12** ユーティリティーを使用して証明書をデプロイメントします。プロンプトが表示されたら、そのファイルに保存されている秘密鍵をパスワードで保護します。

     ```
     $ openssl pkcs12 -in cert_and_key.p12 -clcerts -nokeys -out cert.pem
     Enter Import Password:
     ```

2. 管理者の認証情報を取得します。

   ```
   $ kinit admin
   ```

3. 以下のいずれかの方法で、**IdM CLI** を使用して証明書をユーザーアカウントに追加します。
   - 文字列を **ipa user-add-cert** コマンドに追加する前に、**sed** ユーティリティーを使用して **PEM** ファイルの最初の行および最後の行 (-----BEGIN CERTIFICATE----- および -----END CERTIFICATE-----) を削除します。

     ```
     $ ipa user-add-cert some_user --certificate="$(sed -e '/BEGIN
     CERTIFICATE/d;/END CERTIFICATE/d' cert.pem)"
     ```

   - 最初の行と最後の行 (-----BEGIN CERTIFICATE----- および -----END CERTIFICATE-----) がない証明書ファイルのコンテンツを、**ipa user-add-cert** コマンドにコピーして貼り付けます。

     ```
     $ ipa user-add-cert some_user --
     certificate=MIIDlzCCAn+gAwIBAgIBATANBgkqhki...
     ```

注記

最初の行および最後の行 (-----BEGIN CERTIFICATE----- および -----END CERTIFICATE-----) を削除せずに、直接証明書を含む **PEM** ファイルを、**ipa user-add-cert** コマンドへの入力として直接渡すことはできません。

```
$ ipa user-add-cert some_user --cert=some_user_cert.pem
```

このコマンドの結果、ipa: ERROR: Base64 decoding failed: Incorrect padding エラーメッセージが表示されます。

4. 必要に応じて、システムで証明書が許可されているかどうかを確認するには、次のコマンドを実行します。

```
[idm_user@r8server]$ ipa user-show some_user
```

60.2.3. IdM ユーザーアカウントに読み込むための IdM Web UI での外部証明書の変換

以下の手順に従って、外部証明書を **PEM** 形式に変換し、これを IdM Web UI の IdM ユーザーアカウントに追加します。

手順

1. **CLI** を使用して、証明書を **PEM** 形式に変換します。
   - 証明書が **DER** 形式の場合は、次のようになります。

```
$ openssl x509 -in cert.crt -inform der -outform pem -out cert.pem
```

   - ファイルが **PKCS #12** 形式で、共通のファイル名の拡張子が **.pfx** および **.p12** で、証明書、秘密鍵、およびその他のデータが含まれている場合は、**openssl pkcs12** ユーティリティーを使用して証明書をデプロイメントします。プロンプトが表示されたら、そのファイルに保存されている秘密鍵をパスワードで保護します。

```
$ openssl pkcs12 -in cert_and_key.p12 -clcerts -nokeys -out cert.pem
Enter Import Password:
```

2. エディターで証明書を開き、コンテンツをコピーします。IdM Web UI には **PEM** 形式および **base64** 形式が使用できるため、ヘッダー行-----BEGIN CERTIFICATE-----およびフッター行-----END CERTIFICATE-----を含めることができます。
3. IdM Web UI で、セキュリティー担当者としてログインします。
4. **Identity → Users → some_user** の順に選択します。
5. **証明書** の横にある **追加** をクリックします。
6. 表示されるウインドウに、PEM 形式のコンテンツを貼り付けます。
7. **Add** をクリックします。

証明書がシステムに受け入れられる場合は、ユーザープロファイルの **証明書** 間でリストに表示されるのを確認できます。

60.3. 証明書をブラウザーに読み込むための準備

ユーザー証明書をブラウザーにインポートする前に、証明書と対応する秘密鍵が **PKCS #12** 形式にあることを確認してください。その他の準備作業が必要な一般的な状況は、次の 2 つです。

- 証明書が NSS データベースにある。この状況での続行方法の詳細は、Exporting a certificate and private key from an NSS database into a PKCS #12 file を参照してください。
- 証明書と秘密鍵が別々の **PEM** ファイルにある。この状況での続行方法の詳細は、Combining certificate and private key PEM files into a PKCS #12 file を参照してください。

その後、**PEM** 形式の CA 証明書と、**PKCS #12** 形式のユーザー証明書をブラウザーにインポートするには、証明書認証を有効にするためのブラウザーの設定 および Identity Management ユーザーとして証明書を使用した Identity Management Web UI の認証 の手順に従います。

60.3.1. NSS データベースから PKCS #12 ファイルへの証明書と秘密鍵のエクスポート

手順

1. 証明書を、NSS データベースから **PKCS12** 形式にエクスポートするには、**pk12util** コマンドを使用します。たとえば、**~/certdb** ディレクトリーに保存されている NSS データベースから、**~/some_user.p12** ファイルに、**some_user** ニックネームを持つ証明書をエクスポートする場合は、次のコマンドを実行します。

```
$ pk12util -d ~/certdb -o ~/some_user.p12 -n some_user
Enter Password or Pin for "NSS Certificate DB":
Enter password for PKCS12 file:
Re-enter password:
pk12util: PKCS12 EXPORT SUCCESSFUL
```

2. **.p12** ファイルに適切なパーミッションを設定します。

```
# chmod 600 ~/some_user.p12
```

PKCS #12 ファイルには秘密鍵も含まれるため、その他のユーザーがファイルを使用できないように保護する必要があります。それ以外の場合は、ユーザー権限になりすますことができます。

60.3.2. 証明書と秘密鍵の PEM ファイルを PKCS #12 ファイルに統合

以下の手順に従って、証明書と、別の **PEM** ファイルに保存されている対応する鍵を **PKCS #12** ファイルに組み合わせます。

手順

- **certfile.cer** に保存されている証明書と、**certfile.key** に保存されている鍵を、証明書および鍵の両方が含まれる **certfile.p12** ファイルに追加します。

```
$ openssl pkcs12 -export -in certfile.cer -inkey certfile.key -out certfile.p12
```

60.4. IDM における証明書関連のコマンドおよび形式

次の表に、IdM における証明書関連のコマンドを、使用できる形式と共に示します。

表60.2 IdM 証明書コマンドおよび形式

コマンド	使用できる形式	備考
ipa user-add-cert some_user --certificate	base64 PEM 証明書	
ipa-server-certinstall	PEM および DER 証明書、PKCS#7 証明書チェーン、PKCS#8 および生の秘密鍵、PKCS#12 証明書および秘密鍵	
ipa-cacert-manage install	DER、PEM、PKCS#7	
ipa-cacert-manage renew --external-cert-file	PEM および DER 証明書、PKCS#7 証明書チェーン	
ipa-ca-install --external-cert-file	PEM および DER 証明書、PKCS#7 証明書チェーン	
ipa cert-show <cert serial> --certificate-out /path/to/file.pem	該当なし	**<cert_serial>** シリアル番号がある証明書で、PEM でエンコードされた **file.pem** ファイルを作成します。
ipa cert-show <cert serial> --certificate-out /path/to/file.pem	該当なし	**<cert_serial>** シリアル番号がある証明書で、PEM でエンコードされた **file.pem** ファイルを作成します。**--chain** オプションを使用すると、PEM ファイルに、証明書チェーンを含む証明書が含まれます。
ipa cert-request --certificate-out=FILE /path/to/req.csr	該当なし	新しい証明書で、PEM 形式の **req.csr** ファイルを作成します。
ipa cert-request --certificate-out=FILE /path/to/req.csr	該当なし	新しい証明書で、PEM 形式の **req.csr** ファイルを作成します。**--chain** オプションを使用する場合には、PEM ファイルに、証明書チェーンを含む証明書が含まれます。

第61章 統合 IDM CA を使用したユーザー、ホスト、およびサービスの証明書の管理

統合 CA、**ipa** CA、およびそのサブ CA を使用して Identity Management (IdM)で証明書を管理する方法の詳細は、以下のセクションを参照してください。

- IdM Web UI を使用したユーザー、ホスト、またはサービスの新しい証明書の要求
- IdM CLI を使用した IdM CA からのユーザー、ホスト、またはサービスの新規証明書の要求
  - certutil を使用した IdM CA からのユーザー、ホスト、またはサービスの新規証明書の要求
    - **certutil** ユーティリティーを使用して IdM CA から新規ユーザー証明書を要求して、その証明書を IdM クライアントにエクスポートする具体的な例については、新しいユーザー証明書を要求し、クライアントにエクスポート を参照してください。
  - openssl を使用した IdM CA からのユーザー、ホスト、またはサービスの新規証明書の要求

certmonger ユーティリティーを使用して、IdM CA からサービスの新しい証明書を要求することもできます。詳細は certmonger を使用した IdM CA からのサービスの新規証明書の要求 を参照してください。

前提条件

- IdM デプロイメントに統合 CA が含まれている。
  - IdM で CA サービスを計画する方法の詳細は、CA サービスの計画 を参照してください。
  - 統合 DNS と統合 CA をルート CA として使用する IdM サーバーをインストールする方法は、IdM サーバーのインストール: 統合 DNS と統合 CA をルート CA として使用する場合 を参照してください。
  - 統合 DNS と外部 CA を root CA として使用する IdM サーバーをインストールする方法は、IdM サーバーのインストール: 統合 DNS と外部 CA を root CA として使用する場合 を参照してください。
  - 統合 DNS なしで、統合 CA をルート CA として使用する IdM サーバーをインストールする方法は、IdM サーバーのインストール: 統合 DNS がなく統合 CA を root CA として使用する場合 を参照してください。
  - (オプション) IdM デプロイメントは、証明書で認証するユーザーに対応します。
    - IdM デプロイメントが IdM クライアントファイルシステムに保存されている証明書を使用したユーザー認証をサポートするように設定する方法については、IdM クライアントのデスクトップに保存されている証明書を使用した認証の設定 を参照してください。
    - IdM クライアントに挿入するスマートカードに保存されている証明書を使用してユーザー認証をサポートするように IdM デプロイメントを設定する方法は、スマートカード認証用の Identity Management の設定 を参照してください。
    - Active Directory 証明書システムが発行するスマートカードを使用してユーザー認証をサポートするように IdM デプロイメントを設定する方法は、IdM でスマートカード認証用に ADCS が発行した証明書の設定 を参照してください。

61.1. IDM WEB UI でのユーザー、ホスト、またはサービスの新規証明書の要求

Identity Management (IdM) の Web UI を使用して、統合 IdM 認証局 (CA) から IdM エンティティー (**ipa** CA またはそのサブ CA) の新しい証明書を要求するするには、次の手順に従います。

IdM エンティティーには、以下が含まれます。

- ユーザー
- ホスト
- サービス

重要

サービスは通常、秘密鍵の保存先となる専用のサービスノードで実行されます。サービスの秘密鍵を IdM サーバーにコピーすることは、安全ではないとみなされます。したがって、サービスの証明書を要求する場合には、サービスノードで証明書署名要求 (CSR) を作成します。

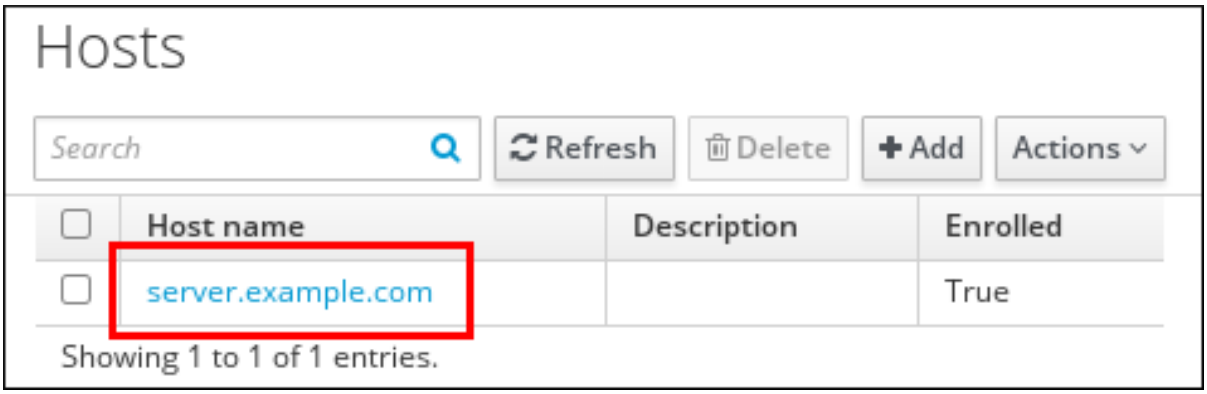
前提条件

- IdM デプロイメントに統合 CA が含まれている。
- IdM 管理者として IdM Web UI にログインしている。

手順

1. **Identity** タブで、 **Users**、**Hosts**、または **Services** のサブタブを選択します。
2. ユーザー、ホスト、またはサービス名をクリックして、設定ページを開きます。

   図61.1 ホストのリスト



3. **Actions → New Certificate** をクリックします。
4. 必要に応じて、発行元の CA およびプロファイル ID を選択します。
5. 画面の **certutil** コマンドライン (CLI) ユーティリティーの使用手順に従います。
6. **Issue** をクリックします。

61.2. CERTUTIL を使用した IDM CA からのユーザー、ホスト、またはサービスの新規証明書の要求

標準の IdM 状況で Identity Management (IdM) ユーザー、ホスト、またはサービスの証明書を要求するには、**certutil** ユーティリティーを使用できます。ホストまたはサービスの Kerberos エイリアスが証明書を使用できるようにするには、代わりに openssl ユーティリティーを使用して証明書を要求 します。

以下の手順に従って、**certutil** を使用して、**ipa** (IdM 認証局(CA)) から IdM ユーザー、ホスト、またはサービスの証明書を要求します。

重要

サービスは通常、秘密鍵の保存先となる専用のサービスノードで実行されます。サービスの秘密鍵を IdM サーバーにコピーすることは、安全ではないとみなされます。したがって、サービスの証明書を要求する場合には、サービスノードで証明書署名要求 (CSR) を作成します。

前提条件

- IdM デプロイメントに統合 CA が含まれている。
- IdM 管理者として IdM コマンドラインインターフェイス (CLI) にログインしている。

手順

1. 証明書データベースの一時ディレクトリーを作成します。

```
# mkdir ~/certdb/
```

2. 以下のように、新しい一時証明書データベースを作成します。

```
# certutil -N -d ~/certdb/
```

3. CSR を作成し、出力をファイルにリダイレクトします。たとえば、4096 ビット証明書の CSR を作成し、発行先を **CN=server.example.com,O=EXAMPLE.COM** に設定するには、以下を実行します。

```
# certutil -R -d ~/certdb/ -a -g 4096 -s "CN=server.example.com,O=EXAMPLE.COM" -8 server.example.com > certificate_request.csr
```

4. IdM サーバーで実行している CA に証明書要求ファイルを送信します。新しく発行した証明書に関連付ける Kerberos プリンシパルを指定します。

```
# ipa cert-request certificate_request.csr --principal=host/server.example.com
```

IdM の **ipa cert-request** コマンドは、次のデフォルトを使用します。

- **caIPAserviceCert** 証明書プロファイル
  カスタムプロファイルを選択するには、**--profile-id** オプションを使用します。
- 統合 IdM のルート CA (**ipa**)
  サブ CA を選択するには、**--ca** オプションを使用します。

関連情報

関連情報

- **ipa cert-request --help** コマンドの出力を参照してください。
- Identity Management での証明書プロファイルの作成および管理 を参照してください。

61.3. OPENSSL を使用した IDM CA からのユーザー、ホスト、またはサービスの新規証明書の要求

ホストまたはサービスの Kerberos エイリアスが証明書を使用できるようにするには、**openssl** ユーティリティーを使用して、Identity Management (IdM) ホストまたはサービスの証明書を要求してください。標準の状況では、代わりに certutil ユーティリティーを使用して新しい証明書を要求することを検討してください。

以下の手順に従って、**openssl** を使用して、IdM ホスト、または **ipa** (IdM 認証局) からサービスの証明書を要求します。

重要

サービスは通常、秘密鍵の保存先となる専用のサービスノードで実行されます。サービスの秘密鍵を IdM サーバーにコピーすることは、安全ではないとみなされます。したがって、サービスの証明書を要求する場合には、サービスノードで証明書署名要求 (CSR) を作成します。

前提条件

- IdM デプロイメントに統合 CA が含まれている。
- IdM 管理者として IdM コマンドラインインターフェイス (CLI) にログインしている。

手順

1. Kerberos プリンシパル **test/server.example.com** のエイリアスを 1 つ以上作成します。例: **test1/server.example.com** および **test2/server.example.com**
2. CSR で dnsName (**server.example.com**) と otherName (**test2/server.example.com**) の subjectAltName を追加します。これには、UPN otherName および subjectAltName を指定する以下の行を追加するように、**openssl.conf** ファイルを 設定します。

```
otherName=1.3.6.1.4.1.311.20.2.3;UTF8:test2/server.example.com@EXAMPLE.COM
DNS.1 = server.example.com
```

3. **openssl** を使用して証明書要求を作成します。

```
openssl req -new -newkey rsa:2048 -keyout test2service.key -sha256 -nodes -out
certificate_request.csr -config openssl.conf
```

4. IdM サーバーで実行している CA に証明書要求ファイルを送信します。新しく発行した証明書に関連付ける Kerberos プリンシパルを指定します。

```
# ipa cert-request certificate_request.csr --principal=host/server.example.com
```

IdM の **ipa cert-request** コマンドは、次のデフォルトを使用します。

- **caIPAserviceCert** 証明書プロファイル
  カスタムプロファイルを選択するには、**--profile-id** オプションを使用します。
- 統合 IdM のルート CA (**ipa**)
  サブ CA を選択するには、**--ca** オプションを使用します。

関連情報

- **ipa cert-request --help** コマンドの出力を参照してください。
- Identity Management での証明書プロファイルの作成および管理 を参照してください。

61.4. 関連情報

- 統合 IdM CA を使用した証明書の失効 を参照してください。
- 統合 IdM CA を使用した証明書の復元 を参照してください。
- 証明書のサブセットのみを信頼するアプリケーションの制限 を参照してください。

第62章 ANSIBLE を使用した IDM 証明書の管理

ansible-freeipa ipacert モジュールを使用して、Identity Management (IdM) ユーザー、ホスト、およびサービスの SSL 証明書を要求、取り消し、および取得できます。保留された証明書を復元することもできます。

62.1. ANSIBLE を使用した IDM ホスト、サービス、ユーザーの SSL 証明書の要求

ansible-freeipa ipacert モジュールを使用して、Identity Management (IdM) ユーザー、ホスト、およびサービスの SSL 証明書を要求できます。その後、これらの証明書を使用して IdM に対する認証を行うことができます。

Ansible Playbook を使用して IdM 認証局 (CA) から HTTP サーバーの証明書を要求するには、この手順を完了します。

前提条件

- コントロールノードでは、
  - Ansible バージョン 2.14 以降を使用している。
  - **ansible-freeipa** パッケージをインストールしている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成した。
  - **ipaadmin_password** を **Secret.yml** Ansible Vault に保存している。
- IdM デプロイメントに統合 CA がある。

手順

1. ユーザー、ホスト、またはサービスの証明書署名要求 (CSR) を生成します。たとえば、**openssl** ユーティリティーを使用して client.idm.example.com で実行されている **HTTP** サービスの CSR を生成するには、次のように入力します。

   ```
   # openssl req -new -newkey rsa:2048 -days 365 -nodes -keyout new.key -out new.csr -subj '/CN=client.idm.example.com,O=IDM.EXAMPLE.COM'
   ```

   その結果、CSR は **new.csr** に保存されます。

2. 次の内容を含む Ansible Playbook ファイル **request-certificate.yml** を作成します。

   ```
   ---
   - name: Playbook to request a certificate
     hosts: ipaserver

     vars_files:
     - /home/user_name/MyPlaybooks/secret.yml

     tasks:
     - name: Request a certificate for a web server
       ipacert:
         ipaadmin_password: "{{ ipaadmin_password }}"
   ```

```
    state: requested
    csr: |
      -----BEGIN CERTIFICATE REQUEST-----

MIGYMEwCAQAwGTEXMBUGA1UEAwwOZnJlZWlwYSBydWxlcyEwKjAFBgMrZXADIQBs
      HlqIr4b/XNK+K8QLJKIzfvuNK0buBhLz3LAzY7QDEqAAMAUGAytlcANBAF4oSCbA
      5aIPukCidnZJdr491G4LBE+URecYXsPknwYb+V+ONnf5ycZHyaFv+jkUBFGFeDgU
      SYaXm/gF8cDYjQI=
      -----END CERTIFICATE REQUEST-----
    principal: HTTP/client.idm.example.com
  register: cert
```

証明書要求を **new.csr** の CSR に置き換えます。

3. 証明書を要求します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
<path_to_inventory_directory>/hosts <path_to_playbooks_directory>/request-
certificate.yml
```

関連情報

- **ansible-freeipa** アップストリームドキュメントの cert モジュール

62.2. ANSIBLE を使用して IDM ホスト、サービス、ユーザーの SSL 証明書を取り消す

ansible-freeipa ipacert モジュールを使用して、Identity Management (IdM) ユーザー、ホスト、およびサービスが IdM への認証に使用する SSL 証明書を取り消すことができます。

Ansible Playbook を使用して HTTP サーバーの証明書を取り消すには、この手順を完了します。証明書を取り消す理由は “keyCompromise” です。

前提条件

- コントロールノードでは、
  - Ansible バージョン 2.14 以降を使用している。
  - **ansible-freeipa** パッケージをインストールしている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成した。
  - **ipaadmin_password** を **Secret.yml** Ansible Vault に保存している。
  - **openssl x509 -noout -text -in <path_to_certificate>** コマンドを入力するなどして、証明書のシリアル番号を取得している。この例では、証明書のシリアル番号は 123456789 です。
- IdM デプロイメントに統合 CA がある。

手順

1. 次の内容を含む Ansible Playbook ファイル **revoke-certificate.yml** を作成します。

```
---
- name: Playbook to revoke a certificate
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml

  tasks:
  - name: Revoke a certificate for a web server
    ipacert:
      ipaadmin_password: "{{ ipaadmin_password }}"
      serial_number: 123456789
      revocation_reason: "keyCompromise"
      state: revoked
```

2. 証明書を取り消します。

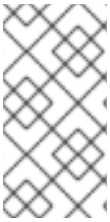
```
$ ansible-playbook --vault-password-file=password_file -v -i
<path_to_inventory_directory>/hosts <path_to_playbooks_directory>/revoke-
certificate.yml
```

関連情報

- **ansible-freeipa** アップストリームドキュメントの cert モジュール
- RFC 5280 の Reason Code

62.3. ANSIBLE を使用して IDM ユーザー、ホスト、およびサービスの SSL 証明書を復元する

ansible-freeipa ipacert モジュールを使用すると、Identity Management (IdM) ユーザー、ホスト、またはサービスが IdM への認証に以前に使用した、取り消された SSL 証明書を復元できます。



注記

復元できるのは、保留された証明書のみです。たとえば、秘密キーを紛失したかどうかがわからないなどの理由で、秘密キーを保留した可能性があります。ただし、キーを回復し、回復までの間に誰もそのキーにアクセスしていないと確信しているため、証明書を復元したいと考えています。

Ansible Playbook を使用して、IdM に登録されたサービスの証明書を保留から解放するには、この手順を完了します。この例では、HTTP サービスの証明書の保留を解除する方法について説明します。

前提条件

- コントロールノードでは、
  - Ansible バージョン 2.14 以降を使用している。
  - **ansible-freeipa** パッケージをインストールしている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成した。

- **ipaadmin_password** を **Secret.yml** Ansible Vault に保存している。

- IdM デプロイメントに統合 CA がある。
- たとえば、**openssl x509 -noout -text -in path/to/certificate** コマンドを入力するなどして、証明書のシリアル番号を取得している。この例では、証明書のシリアル番号は **123456789** です。

手順

1. 次の内容を含む Ansible Playbook ファイル **restore-certificate.yml** を作成します。

```
---
- name: Playbook to restore a certificate
  hosts: ipaserver
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml

  tasks:
  - name: Restore a certificate for a web service
    ipacert:
      ipaadmin_password: "{{ ipaadmin_password }}"
      serial_number: 123456789
      state: released
```

2. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
<path_to_inventory_directory>/hosts <path_to_playbooks_directory>/restore-
certificate.yml
```

関連情報

- **ansible-freeipa** アップストリームドキュメントの cert モジュール

62.4. ANSIBLE を使用して IDM ユーザー、ホスト、およびサービスの SSL 証明書を取得する

ansible-freeipa ipacert モジュールを使用すると、Identity Management (IdM) ユーザー、ホスト、またはサービスに対して発行された SSL 証明書を取得し、マネージドノード上のファイルに保存できます。

前提条件

- コントロールノードでは、
  - Ansible バージョン 2.14 以降を使用している。
  - **ansible-freeipa** パッケージをインストールしている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成した。
  - **ipaadmin_password** を **Secret.yml** Ansible Vault に保存している。

- **openssl x509 -noout -text -in <path_to_certificate>** コマンドを入力するなどして、証明書のシリアル番号を取得している。この例では、証明書のシリアル番号は 123456789 で、取得した証明書を保存するファイルは **cert.pem** です。

手順

1. 次の内容を含む Ansible Playbook ファイル **retrieve-certificate.yml** を作成します。

```
---
- name: Playbook to retrieve a certificate and store it locally on the managed node
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml

  tasks:
  - name: Retrieve a certificate and save it to file 'cert.pem'
    ipacert:
      ipaadmin_password: "{{ ipaadmin_password }}"
      serial_number: 123456789
      certificate_out: cert.pem
      state: retrieved
```

2. 証明書を取得します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
<path_to_inventory_directory>/hosts <path_to_playbooks_directory>/retrieve-
certificate.yml
```

関連情報

- **ansible-freeipa** アップストリームドキュメントの cert モジュール

第63章 IDM ユーザー、ホスト、およびサービスの外部署名証明書の管理

この章では、Identity Management (IdM) コマンドラインインターフェイス (CLI) および IdM Web UI を使用して、外部認証局 (CA) によって発行されたユーザー、ホスト、またはサービス証明書を追加または削除する方法について説明します。

63.1. IDM CLI を使用した外部 CA からの IDM ユーザー、ホスト、またはサービスへの証明書の追加

Identity Management (IdM) 管理者は、Identity Management (IdM) CLI を使用して、外部署名証明書を IdM ユーザー、ホスト、またはサービスのアカウントに追加できます。

前提条件

- 管理ユーザーの Ticket-Granting Ticket (TGT) を取得している。

手順

- IdM ユーザーに証明書を追加するには、次のコマンドを実行します。

```
$ ipa user-add-cert user --certificate=MIQTPrajQAwg...
```

このコマンドでは、以下の情報を指定する必要があります。

  - ユーザーの名前。
  - base64 でエンコードされた DER 証明書

注記

証明書の内容をコマンドラインにコピーして貼り付ける代わりに、証明書を DER 形式に変換し、これを base64 に再プロビジョニングできます。たとえば、**user_cert.pem**証明書を**ユーザー**に追加するには、次のコマンドを入力します。

```
$ ipa user-add-cert user --certificate="$(openssl x509 -outform der -in
user_cert.pem | base64 -w 0)"
```

ipa user-add-cert コマンドは、オプションを追加せずに対話的に実行できます。

IdM ホストに証明書を追加するには、次のコマンドを実行します。

- **ipa host-add-cert**

IdM サービスに証明書を追加するには、次のコマンドを実行します。

- **ipa service-add-cert**

関連情報

- 統合 IdM CA を使用したユーザー、ホスト、およびサービスの証明書の管理

63.2. IDM WEB UI を使用した外部 CA からの IDM ユーザー、ホスト、またはサービスへの証明書の追加

Identity Management (IdM) 管理者は、Identity Management (IdM) Web UI を使用して、外部署名証明書を IdM ユーザー、ホスト、またはサービスのアカウントに追加できます。

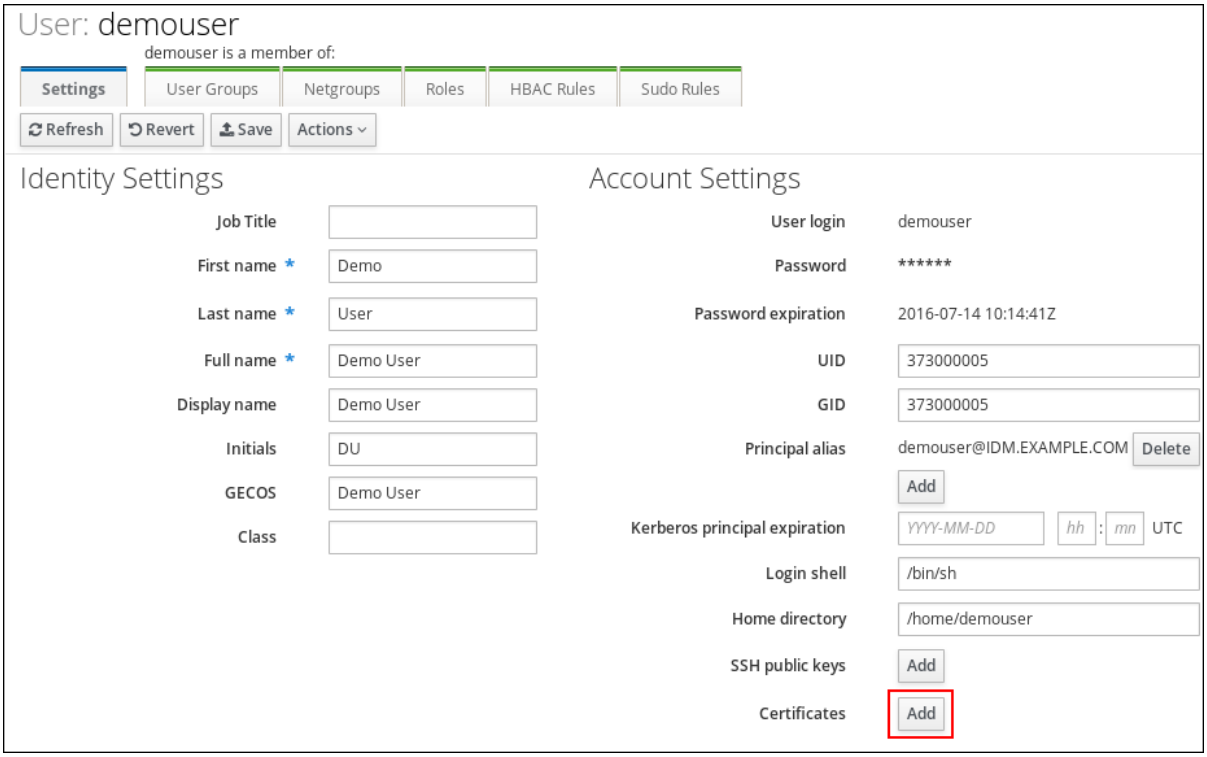
前提条件

- 管理者として Identity Management (IdM) Web UI にログインしている。

手順

1. **Identity** タブを開き、**Users**、**Hosts**、または **Services** サブタブを選択します。
2. ユーザー、ホスト、またはサービス名をクリックして、設定ページを開きます。
3. **Certificates** エントリーの横にある **Add** をクリックします。

図63.1 ユーザーアカウントへの証明書の追加



4. Base64 または PEM でエンコードされた形式で証明書をテキストフィールドに貼り付け、**Add** をクリックします。
5. **Save** をクリックして変更を保存します。

63.3. IDM CLI を使用した IDM ユーザー、ホスト、またはサービスアカウントからの外部 CA 発行の証明書削除

Identity Management (IdM) 管理者は、Identity Management (IdM) CLI を使用して、外部署名証明書を IdM ユーザー、ホスト、またはサービスのアカウントから削除できます。

前提条件

- 管理ユーザーの Ticket-Granting Ticket (TGT) を取得している。

手順

- IdM ユーザーから証明書を削除するには、次のコマンドを実行します。

```
$ ipa user-remove-cert user --certificate=MIQTPrajQAwg...
```

このコマンドでは、以下の情報を指定する必要があります。

  - ユーザーの名前。
  - base64 でエンコードされた DER 証明書

注記

証明書の内容をコマンドラインにコピーして貼り付ける代わりに、証明書を DER 形式に変換し、これを base64 に再プロビジョニングできます。たとえば、**user_cert.pem** 証明書を **user** から削除するには、次のように入力します。

```
$ ipa user-remove-cert user --certificate="$(openssl x509 -outform der -in
user_cert.pem | base64 -w 0)"
```

ipa user-remove-cert コマンドは、オプションを追加せずに対話的に実行できます。

IdM ホストから証明書を削除するには、次のコマンドを実行します。

- **ipa host-remove-cert**

IdM サービスから証明書を削除するには、次のコマンドを実行します。

- **ipa service-remove-cert**

関連情報

- 統合 IdM CA を使用したユーザー、ホスト、およびサービスの証明書の管理

63.4. IDM WEB UI を使用した IDM ユーザー、ホスト、またはサービスアカウントからの外部 CA 発行証明書の削除

Identity Management (IdM) 管理者は、Identity Management (IdM) Web UI を使用して、外部署名証明書を IdM ユーザー、ホスト、またはサービスのアカウントから削除できます。

前提条件

- 管理者として Identity Management (IdM) Web UI にログインしている。

手順

1. **Identity** タブを開き、**Users**、**Hosts**、または **Services** サブタブを選択します。
2. ユーザー、ホスト、またはサービス名をクリックして、設定ページを開きます。

3. 削除する証明書の横にある **Actions** をクリックし、**Delete** を選択します。
4. **Save** をクリックして変更を保存します。

63.5. 関連情報

- Ansible Playbook を使用して IdM サービスエントリーに外部署名証明書を存在させる手順

第64章 IDENTITY MANAGEMENT での証明書プロファイルの作成および管理

証明書プロファイルは、証明書署名要求 (CSR) が受け入れ可能であるかどうかを判断するために、証明書の署名時に認証局 (CA) により使用されます。受け入れ可能な場合には、証明書にどのような機能および拡張機能が含まれるかを判断します。特定のタイプの証明書を発行するのに、証明書プロファイルを関連付けます。証明書プロファイルと CA アクセス制御リスト (ACL) を組み合わせることで、カスタム証明書プロファイルへのアクセスを定義し、制御できます。

証明書プロファイルの作成方法の説明では、例として S/MIME 証明書を使用します。一部のメールプログラムは、Secure Multipurpose Internet Mail Extension (S/MIME) プロトコルを使用して、デジタル署名および暗号化されたメールをサポートします。S/MIME を使用して電子メールメッセージの署名または暗号化を行うには、メッセージの送信者に S/MIME 証明書が必要です。

- 証明書プロファイルの概要
- 証明書プロファイルの作成
- CA アクセス制御リストの概要
- 証明書プロファイルへのアクセスを制御する CA ACL の定義
- 証明書プロファイルおよび CA ACL を使用した証明書発行
- 証明書プロファイルの変更
- 証明書プロファイルの設定パラメーター

64.1. 証明書プロファイルの概要

証明書プロファイルを使用すると、証明書の内容や、以下のような証明書の発行時の制約を決定できます。

- 証明書署名要求をエンコードするために使用する署名アルゴリズム。
- 証明書のデフォルトの有効性。
- 証明書の取り消しに使用できる失効理由。
- プリンシパルのコモンネーム (cn) が subject alternative name フィールドにコピーされる場合。
- 証明書に存在する必要がある機能およびエクステンション。

特定のタイプの証明書を発行するのに、証明書プロファイルを 1 つ関連付けます。IdM のユーザー、サービス、およびホストには、さまざまな証明書プロファイルを定義できます。IdM には、デフォルトで次の証明書プロファイルが含まれています。

- **caIPAserviceCert**
- **IECUserRoles**
- **kdcs_PKINIT_Certs** (内部で使用)

さらに、特定の目的で証明書を発行できるカスタムプロファイルを作成およびインポートできます。たとえば、特定のプロファイルの使用を 1 つのユーザーまたはグループに制限し、他のユーザーやグルー

プがそのプロファイルを使用して証明書を認証用に発行できないようにさせます。カスタム証明書プロファイルを作成するには、**ipa certprofile** コマンドを使用します。

関連情報

- **ipa help certprofile** コマンドを参照してください。

64.2. 証明書プロファイルの作成

S/MIME 証明書を要求するプロファイル設定ファイルを作成して、コマンドラインから証明書プロファイルを作成するには、次の手順に従います。

手順

1. 既存のデフォルトプロファイルをコピーしてカスタムプロファイルを作成します。

```
$ ipa certprofile-show --out smime.cfg caIPAserviceCert
-----------------------------------------------
Profile configuration stored in file 'smime.cfg'
-----------------------------------------------
  Profile ID: caIPAserviceCert
  Profile description: Standard profile for network services
  Store issued certificates: TRUE
```

2. テキストエディターで新たに作成されたプロファイル設定ファイルを開きます。

```
$ vi  smime.cfg
```

3. **プロファイル ID** は、**smime** など、プロファイルの用途を反映する名前に変更します。

注記

新規作成されたプロファイルをインポートする場合は、**profileId** フィールドがある場合には、profiledId はコマンドラインで指定した ID と一致する必要があります。

4. Extended Key Usage 設定を更新します。Extended Key Usage のデフォルト拡張設定は、TLS サーバーとクライアント認証向けです。たとえば、S/MIME の場合に、電子メール保護のために Extended Key Usage を設定する必要があります。

```
policyset.serverCertSet.7.default.params.exKeyUsageOIDs=1.3.6.1.5.5.7.3.4
```

5. 新しいプロファイルをインポートします。

```
$ ipa certprofile-import smime --file smime.cfg \
  --desc "S/MIME certificates" --store TRUE

-----------------------
Imported profile "smime"
-----------------------
  Profile ID: smime
  Profile description: S/MIME certificates
  Store issued certificates: TRUE
```

検証手順

- 新しい証明書プロファイルがインポートされたことを確認します。

```
$ ipa certprofile-find

------------------
4 profiles matched
------------------
  Profile ID: caIPAserviceCert
  Profile description: Standard profile for network services
  Store issued certificates: TRUE

  Profile ID: IECUserRoles
  Profile description: User profile that includes IECUserRoles extension from request
  Store issued certificates: TRUE

  Profile ID: KDCs_PKINIT_Certs
  Profile description: Profile for PKINIT support by KDCs
  Store issued certificates: TRUE

  Profile ID: smime
  Profile description: S/MIME certificates
  Store issued certificates: TRUE
----------------------------
Number of entries returned 4
----------------------------
```

関連情報

- **ipa help certprofile** を参照してください。
- RFC 5280, section 4.2.1.12 を参照してください。

64.3. CA アクセス制御リストの概要

認証局のアクセス制御リスト (CA ACL) ルールは、どのプリンシパルにどのプロファイルを使用して証明書を発行するかを定義します。これには、CA ACL を使用できます。以下に例を示します。

- 特定のプロファイルで証明書を発行できるユーザー、ホスト、またはサービスを決定します。
- 証明書を発行できる IdM 認証局またはサブ CA の特定

たとえば、CA ACL を使用して、ロンドンオフィス関連の IdM グループに所属するユーザーだけが、ロンドンのオフィスから作業する社員向けのプロファイルを使用するように限定できます。

CA ACL ルールを管理する **ipa caacl** ユーティリティーを使用すると、特権ユーザーは、指定した CA ACL を追加、表示、変更、または削除できます。

関連情報

- **ipa help caacl** を参照してください。

64.4. 証明書プロファイルへのアクセスを制御する CA ACL の定義

caacl ユーティリティーを使用して CA アクセス制御リスト (ACL) ルールを定義し、グループ内のユーザーがカスタム証明書プロファイルにアクセスできるようにするには、次の手順に従います。この場合は、S/MIME ユーザーのグループと CA ACL を作成し、そのグループのユーザーが **smime** 証明書プロファイルにアクセスできるようにする方法を説明します。

前提条件

- IdM 管理者の認証情報を取得していることを確認している。

手順

1. 証明書プロファイルのユーザーに新しいグループを作成します。

```
$ ipa group-add smime_users_group
--------------------------------
Added group "smime users group"
--------------------------------
  Group name: smime_users_group
  GID: 75400001
```

2. **smime_user_group** グループに追加する新規ユーザーを作成します。

```
$ ipa user-add smime_user
First name: smime
Last name: user
----------------------
Added user "smime_user"
----------------------
  User login: smime_user
  First name: smime
  Last name: user
  Full name: smime user
  Display name: smime user
  Initials: TU
  Home directory: /home/smime_user
  GECOS: smime user
  Login shell: /bin/sh
  Principal name: smime_user@IDM.EXAMPLE.COM
  Principal alias: smime_user@IDM.EXAMPLE.COM
  Email address: smime_user@idm.example.com
  UID: 1505000004
  GID: 1505000004
  Password: False
  Member of groups: ipausers
  Kerberos keys available: False
```

3. **smime_user** は **smime_users_group** に追加します。

```
$ ipa group-add-member smime_users_group --users=smime_user
  Group name: smime_users_group
  GID: 1505000003
  Member users: smime_user
-------------------------
Number of members added 1
-------------------------
```

4. CA ACL を作成し、グループのユーザーが証明書プロファイルにアクセスできるようにします。

```
$ ipa caacl-add smime_acl
-------------------------
Added CA ACL "smime_acl"
-------------------------
  ACL name: smime_acl
  Enabled: TRUE
```

5. CA ACL にユーザーグループを追加します。

```
$ ipa caacl-add-user smime_acl --group smime_users_group
  ACL name: smime_acl
  Enabled: TRUE
  User Groups: smime_users_group
-------------------------
Number of members added 1
-------------------------
```

6. CA ACL に証明書プロファイルを追加します。

```
$ ipa caacl-add-profile smime_acl --certprofile smime
  ACL name: smime_acl
  Enabled: TRUE
  Profiles: smime
  User Groups: smime_users_group
-------------------------
Number of members added 1
-------------------------
```

検証手順

- 作成した CA ACL の詳細を表示します。

```
$ ipa caacl-show smime_acl
  ACL name: smime_acl
  Enabled: TRUE
  Profiles: smime
  User Groups: smime_users_group
...
```

関連情報

- **ipa** の man ページを参照してください。
- **ipa help caacl** を参照してください。

64.5. 証明書プロファイルおよび CA ACL を使用した証明書発行

認証局のアクセス制御リスト (CA ACL) が許可する場合は、証明書プロファイルを使用して証明書を要求できます。CA ACL 経由でアクセスが付与されたカスタム証明書プロファイルを使用して、ユーザーの S/MIME 証明書を要求するするには、次の手順に従います。

前提条件

- 証明書プロファイルが作成されている。
- ユーザーが必要な証明書プロファイルを使用して証明書を要求可能な CA ACL が作成されている。

注記

cert-request コマンドを実行するユーザーが以下の条件に該当する場合には、CA ACL チェックを回避できます。

- **admin** ユーザーである。
- **Request Certificate ignoring CA ACLs** パーミッションがある。

手順

1. ユーザーの証明書要求を生成します。例: OpenSSL の使用:

```
$ openssl req -new -newkey rsa:2048 -days 365 -nodes -keyout private.key -out cert.csr -subj '/CN=smime_user'
```

2. IdM CA からユーザーの新しい証明書を要求します。

```
$ ipa cert-request cert.csr --principal=smime_user --profile-id=smime
```

必要に応じて、--ca **sub-CA_name** オプションをコマンドに指定して、ルート CA ではなくサブ CA から証明書を要求します。

検証手順

- 新たに発行した証明書がユーザーに割り当てられていることを確認します。

```
$ ipa user-show user
  User login: user
  ...
  Certificate: MIICfzCCAWcCAQA...
  ...
```

関連情報

- **ipa(a)** の man ページを参照してください。
- **ipa help user-show** コマンドを参照してください。
- **ipa help cert-request** コマンドを参照してください。
- **openssl(lssl)** の man ページを参照してください。

64.6. 証明書プロファイルの変更

ipa certprofile-mod コマンドを使用して、コマンドラインで証明書プロファイルを直接変更するには、次の手順に従います。

手順

1. 変更する証明書プロファイルの証明書プロファイル ID を確認します。IdM に現在保存されている証明書プロファイルをすべて表示するには、次のコマンドを実行します。

```
# ipa certprofile-find

------------------
4 profiles matched
------------------
  Profile ID: caIPAserviceCert
  Profile description: Standard profile for network services
  Store issued certificates: TRUE

  Profile ID: IECUserRoles
  ...

  Profile ID: smime
  Profile description: S/MIME certificates
  Store issued certificates: TRUE
----------------------------
Number of entries returned
----------------------------
```

2. 証明書プロファイルの説明を変更します。たとえば、既存のプロファイルを使用して S/MIME 証明書のカスタム証明書プロファイルを作成した場合は、説明を新しい使用方法に合わせて変更します。

```
# ipa certprofile-mod smime --desc "New certificate profile description"
-----------------------------------
Modified Certificate Profile "smime"
-----------------------------------
   Profile ID: smime
   Profile description: New certificate profile description
   Store issued certificates: TRUE
```

3. テキストエディターでカスタマー証明書プロファイルファイルを開き、要件に合わせてを変更します。

```
# vi smime.cfg
```

証明書プロファイルの設定ファイルで指定できるオプションの詳細は、証明書プロファイル設定パラメーター を参照してください。

4. 既存の証明書プロファイルの設定ファイルを更新します。

```
# ipa certprofile-mod _profile_ID_ --file=smime.cfg
```

検証手順

- 証明書プロファイルが更新されたことを確認します。

```
$ ipa certprofile-show smime
  Profile ID: smime
  Profile description: New certificate profile description
  Store issued certificates: TRUE
```

関連情報

- **ipa(a)** の man ページを参照してください。
- **ipa help certprofile-mod** を参照してください。

64.7. 証明書プロファイルの設定パラメーター

証明書プロファイル設定パラメーターは、CA プロファイルディレクトリー **/var/lib/pki/pki-tomcat/ca/profiles/ca** の **profile_name** ファイルに保存されます。プロファイルのパラメーター (defaults、inputs、outputs および constraints) はすべて、ポリシーセット 1 つで設定されます。証明書プロファイルのポリシーセットには **policyset.policyName.policyNumber** という名前があります。たとえば、ポリシーセット **serverCertSet** の場合、以下を実行します。

```
policyset.list=serverCertSet
policyset.serverCertSet.list=1,2,3,4,5,6,7,8
policyset.serverCertSet.1.constraint.class_id=subjectNameConstraintImpl
policyset.serverCertSet.1.constraint.name=Subject Name Constraint
policyset.serverCertSet.1.constraint.params.pattern=CN=[^,]+,.+
policyset.serverCertSet.1.constraint.params.accept=true
policyset.serverCertSet.1.default.class_id=subjectNameDefaultImpl
policyset.serverCertSet.1.default.name=Subject Name Default
policyset.serverCertSet.1.default.params.name=CN=$request.req_subject_name.cn$, OU=pki-ipa,
O=IPA
policyset.serverCertSet.2.constraint.class_id=validityConstraintImpl
policyset.serverCertSet.2.constraint.name=Validity Constraint
policyset.serverCertSet.2.constraint.params.range=740
policyset.serverCertSet.2.constraint.params.notBeforeCheck=false
policyset.serverCertSet.2.constraint.params.notAfterCheck=false
policyset.serverCertSet.2.default.class_id=validityDefaultImpl
policyset.serverCertSet.2.default.name=Validity Default
policyset.serverCertSet.2.default.params.range=731
policyset.serverCertSet.2.default.params.startTime=0
```

各ポリシーセットには、評価順で証明書プロファイルに設定されたポリシーのリスト (ポリシー ID 番号) が含まれます。サーバーは、受信するリクエストごとに各ポリシーセットを評価します。証明書要求を 1 つ受信すると、セット 1 つが評価され、その他のプロファイルセットは無視されます。デュアルキーペアが発行されると、最初の証明書要求に対して最初のポリシーセットが評価され、2 番目の証明書要求に対して 2 番目のセットが評価されます。デュアルキーペアを発行するときに、単一の証明書を発行する場合や 2 つ以上のセットを発行する場合は、複数のポリシーセットは必要ありません。

表64.1 証明書プロファイル設定ファイルパラメーター

パラメーター	説明

パラメーター	説明
desc	end-entities ページに表示される証明書プロファイルの説明 (フリーテキスト)。例: **desc=This certificate profile is for enrolling server certificates with agent authentication**。
enable	プロファイルを有効にし、end-entities ページからアクセスできるようにします。例: **enable=true**
auth.instance_id	証明書要求の認証に使用する認証マネージャープラグインを設定します。自動登録では、認証に成功すると、CA は証明書をすぐに発行します。認証に失敗した場合や、認証プラグインが指定されていない場合には、エージェントにより手動で承認されるように、要求はキューに配置されます。例: **auth.instance_id=AgentCertAuth**
authz.acl	承認の制約を指定します。これは主に、グループ評価のアクセス制御リスト (ACL) を設定するために使用されます。たとえば、**caCMCUserCert** パラメーターでは、CMC リクエストの署名者が Certificate Manager Agents グループに所属する必要があります。 **authz.acl=group="Certificate Manager Agents** ディレクトリーベースのユーザー証明書の更新では、このオプションを使用して、元の要求元と現在認証されているユーザーが同じであることを確認できます。エンティティーは、承認の評価前に認証を行う (バインドまたは基本的にはシステムにログイン) 必要があります。
name	証明書プロファイルの名前。たとえば、**name=Agent-Authenticated Server Certificate Enrollment** など。この名前は、エンドユーザーの登録または更新ページに表示されます。
input.list	証明書プロファイルが対応する入力を名前別に表示します。たとえば、**input.list=i1,i2** のようになります。
input.input_id.class_id	入力 ID (input.list にリスト表示される入力の名前) 別に入力の java クラス名を示します。例: **input.i1.class_id=certReqInputImpl**。

パラメーター	説明
output.list	証明書プロファイルの出力形式を名前別に表示します。例: **output.list=o1**。
output.output_id.class_id	output.list に名前が付けられた出力形式の Java クラス名を指定します。例: **output.o1.class_id=certOutputImpl**。
policyset.list	設定した証明書プロファイルルールをリスト表示します。デュアル証明書の場合には、1 セットのルールが署名鍵に、もう 1 つは暗号鍵に適用されます。単一の証明書は、証明書プロファイルルールのセットを 1 つだけ使用します。例: **policyset.list=serverCertSet**
policyset.policyset_id.list	評価順で証明書プロファイルに設定されたポリシーのリスト (ポリシー ID 番号) を表示します。例: **policyset.serverCertSet.list=1,2,3,4,5,6,7,8**
policyset.policyset_id.policy_number.constraint.class_id	プロファイルルールにデフォルト設定された制約プラグインの java クラス名を指定します。例: policyset.serverCertSet.1.constraint.class_id=subjectNameConstraintImpl。
policyset.policyset_id.policy_number.constraint.name	制約のユーザー定義の名前を指定します。例: policyset.serverCertSet.1.constraint.name=Subject Name Constraint
policyset.policyset_id.policy_number.constraint.params.attribute	制約で使用可能な属性値を指定します。設定可能な属性は、制約のタイプによって異なります。例: policyset.serverCertSet.1.constraint.params.pattern=CN=.*
policyset.policyset_id.policy_number.default.class_id	プロファイルルールに、デフォルトセットの java クラス名を指定します。例: policyset.serverCertSet.1.default.class_id=userSubjectNameDefaultImpl
policyset.policyset_id.policy_number.default.name	デフォルトのユーザー定義名を指定します。例: policyset.serverCertSet.1.default.name=Subject Name Default
policyset.policyset_id.policy_number.default.params.attribute	デフォルトに使用可能な属性の値を指定します。設定可能な属性は、デフォルトのタイプによって異なります。例: policyset.serverCertSet.1.default.params.name=CN=(Name)$request.requestor_name$

第65章 IDM での証明書の有効性の管理

Identity Management (IdM) では、既存の証明書と将来発行する証明書の両方の有効性を管理できますが、その方法は異なります。

65.1. IDM CA が発行した既存の証明書の効力の管理

IdM では、証明書の有効期限を表示する次の方法を使用できます。

- IdM WebUI での有効期限の表示
- CLI での有効期限の表示

IdM CA により発行した既存の証明書の効力を次の方法で管理できます。

- 元の証明書署名要求 (CSR) または秘密鍵から生成された新しい CSR を使用して新しい証明書を要求することにより、証明書を更新します。次のユーティリティーを使用して、新しい証明書を要求できます。

  certmonger

  certmonger を使用して、サービス証明書を要求できます。証明書の有効期限が切れる前に、**certmonger** は証明書を自動的に更新するため、サービス証明書の継続的な効力が保証されます。詳細は、certmonger でサービスの IdM 証明書の取得 を参照してください。

  certutil

  certutil を使用して、ユーザー、ホスト、およびサービスの証明書を更新できます。ユーザー証明書を要求する方法は、新しいユーザー証明書を要求し、クライアントにエクスポート を参照してください。

  openssl

  openssl を使用して、ユーザー、ホスト、およびサービスの証明書を更新できます。

- 証明書を取り消します。詳細は、次を参照してください。
  - IdM WebUI で統合 IdM CA での証明書の失効
  - IdM CLI で統合 IdM CA での証明書の失効
- 証明書が一時的に取り消された場合は、証明書を復元します。詳細は、次を参照してください。
  - IdM WebUI で統合 IdM CA で証明書の復元
  - IdM CLI で統合 IdM CA で証明書の復元

65.2. IDM CA が発行する将来の証明書の効力の管理

IdM CA が発行する将来の証明書の効力を管理するには、証明書プロファイルを変更、インポート、または作成します。詳細は Identity Management での証明書プロファイルの作成および管理 を参照してください。

65.3. IDM WEBUI での証明書の有効期限の表示

IdM WebUI を使用して、IdM CA が発行したすべての証明書の有効期限を表示できます。

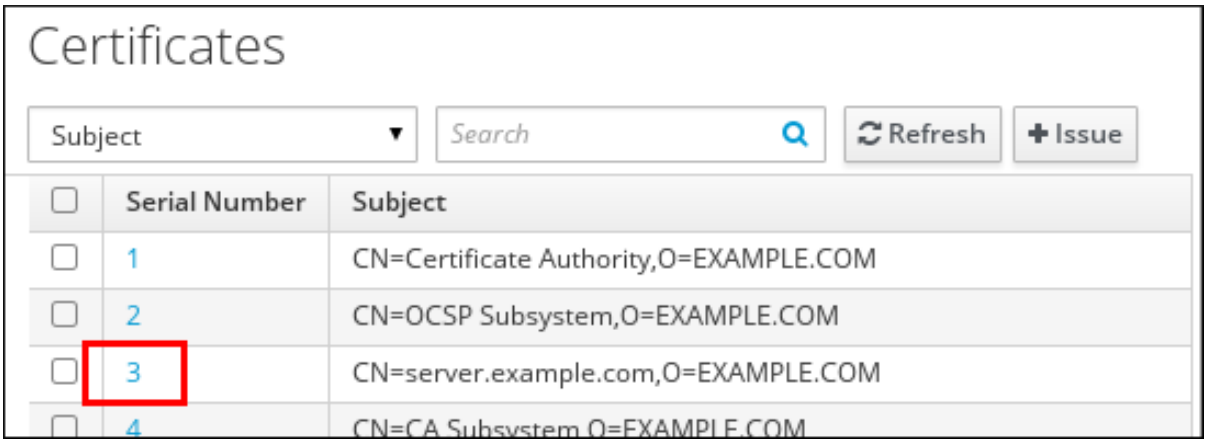
前提条件

- 管理者の認証情報を取得していることを確認している。

手順

1. **Authentication** メニューで、**Certificates** > **Certificates** をクリックします。
2. 証明書のシリアル番号をクリックして、証明書情報ページを開きます。

   図65.1 証明書のリスト



3. 証明書の情報ページで、**失効日** 情報を見つけます。

65.4. CLI での証明書の有効期限の表示

コマンドラインインターフェイス (CLI) を使用して、証明書の有効期限を表示できます。

手順

- **openssl** ユーティリティーを使用して、人間が読める形式でファイルを開きます。

```
$ openssl x509 -noout -text -in ca.pem
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 1 (0x1)
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: O = IDM.EXAMPLE.COM, CN = Certificate Authority
        Validity
            Not Before: Oct 30 19:39:14 2017 GMT
            Not After : Oct 30 19:39:14 2037 GMT
```

65.5. 統合 IDM CA を使用した証明書の失効

65.5.1. 証明書失効の理由

失効した証明書は無効であり、認証に使用できません。理由 6 の **証明書の保留** を除き、すべての失効は永続的です。

デフォルトの失効理由は 0 (**未指定**) です。

表65.1 証明書の失効理由

ID	理由	説明
0	指定なし	
1	鍵が侵害された	証明書を発行した鍵が信頼されなくなった。 考えられる原因 - トークンの消失、ファイルへの不適切なアクセス。
2	CA が侵害された	証明書を発行した CA は信頼されなくなった。
3	所属が変更した	考えられる原因: * 退職したか、別の部門に移動した。 * ホストまたはサービスが廃止された。
4	置き換え	現在の証明書から新しい証明書に置き換えられた。
5	運用停止	ホストまたはサービスの使用を停止している。
6	証明書が保留になっている	証明書は一時的に取り消されている。証明書は後で復元できます。
8	CRL から削除された	証明書は、証明書失効リスト (CRL) に含まれていない。
9	特権が撤回された	ユーザー、ホスト、またはサービスは、証明書の使用を許可されなくなった。
10	侵害された属性機関 (Attribute Authority)	属性機関証明書は信頼されなくなった。

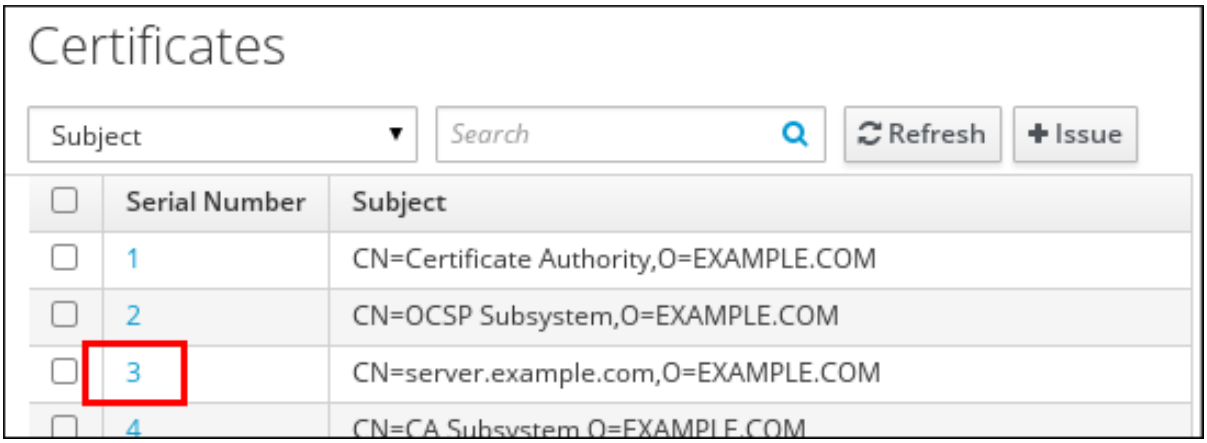
65.5.2. IdM WebUI を使用して統合 IdM CA で証明書の失効

証明書の秘密鍵を紛失した場合は、証明書を無効にして不正使用を防ぐ必要があります。IdM CA が発行した証明書を IdM WebUI を使用して取り消すには、この手順を完了します。

手順

1. **Authentication** > **Certificates** > **Certificates** をクリックします。
2. 証明書のシリアル番号をクリックして、証明書情報ページを開きます。

図65.2 証明書のリスト



3. 証明書情報ページで、**Actions → Revoke Certificate** をクリックします。
4. 取り消しの理由を選択し、**Revoke** をクリックします。詳細は 証明書失効の理由 を参照してください。

65.5.3. IdM CLI を使用した統合 IdM CA での証明書の失効

証明書の秘密鍵を紛失した場合は、証明書を無効にして不正使用を防ぐ必要があります。IdM CLI で、IdM CA が発行した証明書を取り消すには、以下の手順を行います。

手順

- **ipa cert-revoke** コマンドを使用して、次を指定します。
  - 証明書のシリアル番号
  - 失効理由の ID 番号。詳細は 証明書失効の理由 を参照してください。

たとえば、理由 1 (**侵害された鍵**) のためにシリアル番号 **1032** の証明書を失効させるには、次のコマンドを実行します。

```
$ ipa cert-revoke 1032 --revocation-reason=1
```

新しい証明書の要求の詳細は、次のドキュメントを参照してください。

- 新しいユーザー証明書を要求し、クライアントにエクスポート
- certmonger でサービスの IdM 証明書の取得

65.6. 統合 IDM CA を使用した証明書の復元

理由 6 (**証明書の保留**) のために証明書が失効し、証明書の秘密鍵が侵害されていない場合は、証明書を再度復元できます。証明書を復元するには、次のいずれかの手順を使用します。

- IdM WebUI で統合 IdM CA を使用した証明書の復元
- IdM CLI で統合 IdM CA を使用した証明書の復元

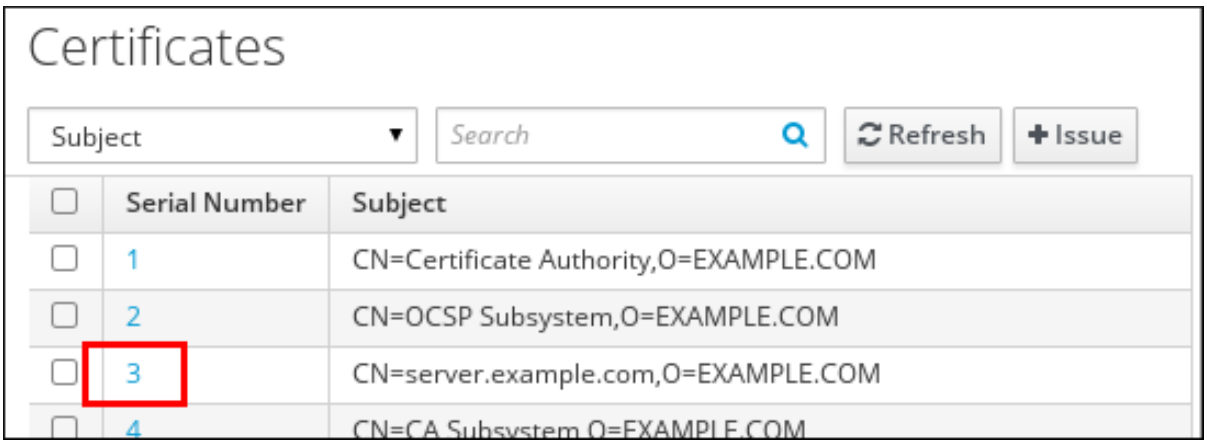
65.6.1. IdM WebUI を使用して統合 IdM CA で証明書の復元

IdM WebUI を使用して、理由 6 (**証明書の保留**) のために取り消された IdM 証明書を復元するには、この手順を完了します。

手順

1. **Authentication** メニューで、**Certificates** > **Certificates** をクリックします。
2. 証明書のシリアル番号をクリックして、証明書情報ページを開きます。

   図65.3 証明書のリスト



3. 証明書情報ページで、**Actions → Restore Certificate** をクリックします。

65.6.2. IdM CA を使用して、統合 IdM CA で証明書の失効

IdM CLI を使用して、理由 6 (**証明書の保留**) のために取り消された IdM 証明書を復元するには、この手順を完了します。

手順

- **ipa cert-remove-hold** コマンドを使用して、証明書のシリアル番号を指定します。以下に例を示します。

```
$ ipa cert-remove-hold 1032
```

第66章 スマートカード認証用の IDENTITY MANAGEMENT の設定

Identity Management (IdM) では、以下によるスマートカード認証に対応しています。

- IdM 認証局が発行するユーザー証明書
- 外部認証局が発行するユーザー証明書

両方のタイプの証明書に対して、IdM でスマートカード認証を設定できます。このシナリオでは、**rootca.pem** CA 証明書は、信頼された外部の認証局の証明書を含むファイルです。

IdM でのスマートカード認証については、スマートカード認証について を参照してください。

スマートカード認証の設定に関する詳細は、以下を参照してください。

- スマートカード認証用の IdM サーバーの設定
- スマートカード認証用の IdM クライアントの設定
- IdM Web UI のユーザーエントリーへの証明書の追加
- IdM CLI でユーザーエントリーへの証明書の追加
- スマートカードを管理および使用するツールのインストール
- スマートカードでの証明書の保存
- スマートカードを使用して IdM へのログイン
- スマートカード認証で GDM アクセスの設定
- スマートカード認証で su アクセスの設定

66.1. スマートカード認証用の IDM サーバーの設定

Identity Management (IdM) CA が信頼する <EXAMPLE.ORG> ドメインの認証局 (CA) によって証明書が発行されたユーザーに対してスマートカード認証を有効にする場合は、次の証明書を取得し、IdM サーバーを設定する **ipa-advise** スクリプトを実行するときにそれらを追加できるようにする必要があります。

- <EXAMPLE.ORG> CA の証明書を直接、または 1 つ以上のサブ CA を通じて発行した root CA の証明書。証明書チェーンは、その CA が証明書を発行した Web ページからダウンロードできます。詳細は、証明書認証を有効にするためのブラウザーの設定 の手順 1 - 4a を参照してください。
- IdM CA 証明書。IdM CA インスタンスが実行されている IdM サーバーの **/etc/ipa/ca.crt** ファイルから CA 証明書を取得できます。
- すべての中間 CA、つまり <EXAMPLE.ORG> CA と IdM CA の中間 CA の証明書。

スマートカード認証用に IdM サーバーを設定するには、以下を行います。

1. PEM 形式の CA 証明書を含むファイルを取得します。
2. ビルトイン **ipa-advise** スクリプトを実行します。

3. システム設定をリロードします。

前提条件

- IdM サーバーへの root アクセス権限がある。
- ルート CA 証明書とすべての中間 CA 証明書があります。

手順

1. 設定を行うディレクトリーを作成します。

```
[root@server]# mkdir ~/SmartCard/
```

2. そのディレクトリーに移動します。

```
[root@server]# cd ~/SmartCard/
```

3. PEM 形式のファイルに保存されている関連する CA 証明書を取得します。CA 証明書が DER などの異なる形式のファイルに保存されている場合は、これを PEM 形式に変換します。IdM 認証局の証明書は PEM 形式で、**/etc/ipa/ca.crt** ファイルにあります。
DER ファイルを PEM ファイルに変換します。

```
# openssl x509 -in <filename>.der -inform DER -out <filename>.pem -outform PEM
```

4. 便宜上、設定を行うディレクトリーに証明書をコピーします。

```
[root@server SmartCard]# cp /tmp/rootca.pem ~/SmartCard/
[root@server SmartCard]# cp /tmp/subca.pem ~/SmartCard/
[root@server SmartCard]# cp /tmp/issuingca.pem ~/SmartCard/
```

5. 必要に応じて、外部の認証局の証明書を使用する場合は、**openssl x509** ユーティリティーを使用して **PEM** 形式のファイルの内容を表示し、**Issuer** および **Subject** の値が正しいことを確認します。

```
[root@server SmartCard]# openssl x509 -noout -text -in rootca.pem | more
```

6. 管理者の権限を使用して、組み込みの **ipa-advise** ユーティリティーで設定スクリプトを生成します。

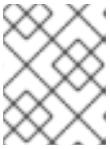
```
[root@server SmartCard]# kinit admin
[root@server SmartCard]# ipa-advise config-server-for-smart-card-auth > config-server-for-smart-card-auth.sh
```

config-server-for-smart-card-auth.sh スクリプトは、以下の操作を実行します。

- IdM Apache HTTP サーバーを設定します。
- キー配布センター (KDC) の Kerberos (PKINIT) で、初回認証用の公開鍵暗号化機能を有効にします。
- スマートカード認可要求を受け入れるように IdM Web UI を設定します。

7. スクリプトを実行し、root CA と サブ CA 証明書が含まれる PEM ファイルを引数として追加します。

```
[root@server SmartCard]# chmod +x config-server-for-smart-card-auth.sh
[root@server SmartCard]# ./config-server-for-smart-card-auth.sh rootca.pem subca.pem
issuingca.pem
Ticket cache:KEYRING:persistent:0:0
Default principal: admin@IDM.EXAMPLE.COM
[...]
Systemwide CA database updated.
The ipa-certupdate command was successful
```



注記

ルート CA 証明書を、サブ CA 証明書の前に引数として追加します。また、CA またはサブ CA 証明書の有効期限が切れていないことを確認します。

8. 必要に応じて、ユーザー証明書を発行した認証局が Online Certificate Status Protocol (OCSP) レスポンダーを提供しない場合は、IdM Web UI への認証に対する OCSP チェックを無効にすることが必要になる場合があります。

   a. **/etc/httpd/conf.d/ssl.conf** ファイルで **SSLOCSPEnable** パラメーターを **off** に設定します。

   ```
   SSLOCSPEnable off
   ```

   b. 変更をすぐに有効にするには、Apache デーモン (httpd) を再起動します。

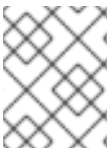
   ```
   [root@server SmartCard]# systemctl restart httpd
   ```

警告

IdM CA が発行したユーザー証明書のみを使用する場合は、OCSP チェックを無効にしないでください。OCSP レスポンダーは IdM に含まれます。

ユーザー証明書を発行した CA が、OCSP サービスリクエストをリッスンする場所に関する情報がユーザー証明書に含まれていない場合に、OCSP チェックを有効にしたまま、ユーザー証明書が IdM サーバーにより拒否されないようにする方法は、Apache mod_ssl 設定オプションの **SSLOCSPDefaultResponder** ディレクティブを参照してください。

これで、スマートカード認証にサーバーが設定されました。



注記

トポロジー全体でスマートカード認証を有効にするには、各 IdM サーバーで手順を実行します。

66.2. ANSIBLE を使用したスマートカード認証用の IDM サーバー設定

Ansible を使用して、Identity Management (IdM) CA が信頼する <EXAMPLE.ORG> ドメインの認証局 (CA) によって証明書が発行されたユーザーのスマートカード認証を有効にできます。そのためには、**ipasmartcard_server ansible-freeipa** ロールスクリプトを使用して Ansible Playbook を実行するときに使用できるように、次の証明書を取得する必要があります。

- <EXAMPLE.ORG> CA の証明書を直接、または 1 つ以上のサブ CA を通じて発行した root CA の証明書。証明書チェーンは、その CA が証明書を発行した Web ページからダウンロードできます。詳細は、証明書認証を有効にするためのブラウザーの設定 の手順 4 を参照してください。
- IdM CA 証明書。CA 証明書は、任意の IdM CA サーバーの **/etc/ipa/ca.crt** ファイルから取得できます。
- <EXAMPLE.ORG> CA と IdM CA の中間にあるすべての CA の証明書。

前提条件

- IdM サーバーへの **root** アクセス権限がある。
- IdM **admin** のパスワードを把握している。
- ルート CA 証明書、IdM CA 証明書、すべての中間 CA の証明書がある。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. CA 証明書が **DER** などをはじめとする別の形式のファイルに保存されている場合、それらを **PEM** 形式に変換します。

```
# openssl x509 -in <filename>.der -inform DER -out <filename>.pem -outform PEM
```

   IdM 認証局の証明書は **PEM** 形式で、**/etc/ipa/ca.crt** ファイルにあります。

2. 必要に応じて、**openssl x509** ユーティリティーを使用して **PEM** 形式のファイルの内容を表示し、**Issuer** および **Subject** の値が正しいことを確認します。

```
# openssl x509 -noout -text -in root-ca.pem | more
```

3. **~/MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

4. CA 証明書専用のサブディレクトリーを作成します。

```
$ mkdir SmartCard/
```

5. 便宜上、必要なすべての証明書を **~/MyPlaybooks/SmartCard/** ディレクトリーにコピーします。

```
# cp /tmp/root-ca.pem ~/MyPlaybooks/SmartCard/
# cp /tmp/intermediate-ca.pem ~/MyPlaybooks/SmartCard/
# cp /etc/ipa/ca.crt ~/MyPlaybooks/SmartCard/ipa-ca.crt
```

6. Ansible インベントリーファイルで、以下を指定します。

- スマートカード認証用に設定する IdM サーバー。
- IdM 管理者パスワード。
- CA の証明書へのパス (次の順序に従う)。
  - ルート CA 証明書ファイル
  - 中間 CA 証明書ファイル
  - IdM CA 証明書ファイル

ファイルは次のようになります。

```
[ipaserver]
ipaserver.idm.example.com

[ipareplicas]
ipareplica1.idm.example.com
ipareplica2.idm.example.com

[ipacluster:children]
ipaserver
ipareplicas

[ipacluster:vars]
ipaadmin_password= "{{ ipaadmin_password }}"
ipasmartcard_server_ca_certs=/home/<user_name>/MyPlaybooks/SmartCard/root-ca.pem,/home/<user_name>/MyPlaybooks/SmartCard/intermediate-ca.pem,/home/<user_name>/MyPlaybooks/SmartCard/ipa-ca.crt
```

7. 次の内容で **install-smartcard-server.yml** playbook を作成します。

```
---
- name: Playbook to set up smart card authentication for an IdM server
  hosts: ipaserver
  become: true

  roles:
  - role: ipasmartcard_server
    state: present
```

8. ファイルを保存します。

9. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory install-smartcard-server.yml
```

ipasmartcard_server Ansible ロールは、次のアクションを実行します。

- IdM Apache HTTP サーバーを設定します。
- キー配布センター (KDC) の Kerberos (PKINIT) で、初回認証用の公開鍵暗号化機能を有効にします。
- スマートカード認可要求を受け入れるように IdM Web UI を設定します。

10. 必要に応じて、ユーザー証明書を発行した認証局が Online Certificate Status Protocol (OCSP) レスポンダーを提供しない場合は、IdM Web UI への認証に対する OCSP チェックを無効にすることが必要になる場合があります。

a. **root** として IdM サーバーに接続します。

```
ssh root@ipaserver.idm.example.com
```

b. **/etc/httpd/conf.d/ssl.conf** ファイルで **SSLOCSPEnable** パラメーターを **off** に設定します。

```
SSLOCSPEnable off
```

c. 変更をすぐに有効にするには、Apache デーモン (httpd) を再起動します。

```
# systemctl restart httpd
```

警告

IdM CA が発行したユーザー証明書のみを使用する場合は、OCSP チェックを無効にしないでください。OCSP レスポンダーは IdM に含まれます。

ユーザー証明書を発行した CA が、OCSP サービスリクエストをリッスンする場所に関する情報がユーザー証明書に含まれていない場合に、OCSP チェックを有効にしたまま、ユーザー証明書が IdM サーバーにより拒否されないようにする方法は、Apache mod_ssl 設定オプションの **SSLOCSPDefaultResponder** ディレクティブを参照してください。

これで、インベントリーファイルにリストされているサーバーがスマートカード認証用に設定されました。

注記

トポロジー全体でスマートカード認証を有効にするには、Ansible Playbook の **hosts** 変数を **ipacluster** に設定します。

```
---
- name: Playbook to setup smartcard for IPA server and replicas
  hosts: ipacluster
[...]
```

関連情報

- **/usr/share/doc/ansible-freeipa/playbooks/** ディレクトリーで **ipasmartcard_server** ロールを使用するサンプル Playbook

66.3. スマートカード認証用の IDM クライアントの設定

以下の手順に従って、スマートカード認証用に IdM クライアントを設定します。この手順は、認証にスマートカードを使用しているときに接続する各 IdM システム、クライアント、またはサーバーで実行する必要があります。たとえば、ホスト A からホスト B への **ssh** 接続を有効にするには、スクリプトをホスト B で実行する必要があります。

管理者として、以下を使用して、この手順でスマートカード認証を有効にします。

- **ssh** プロトコル
  詳細は、スマートカード認証で SSH アクセスの設定 を参照してください。
- コンソールのログイン
- GNOME Display Manager (GDM)
- **su** コマンド

この手順は、IdM Web UI に対する認証には必要ありません。IdM Web UI の認証には 2 つのホストが関係しますが、どちらも IdM クライアントである必要はありません。

- ブラウザーが実行されているマシン。マシンは IdM ドメインの外にある場合があります。
- **httpd** が実行している IdM サーバー

以下の手順は、IdM サーバーではなく、IdM クライアントでスマートカード認証を設定していることを前提としています。このため、2 台のコンピューターが必要です。設定スクリプトを生成する IdM サーバーと、スクリプトを実行する IdM クライアントが必要になります。

前提条件

- Configuring the IdM server for smart card authentication に従って、IdM サーバーがスマートカード認証用に設定されている。
- IdM サーバーと IdM クライアントに root アクセス権限がある。
- ルート CA 証明書とすべての中間 CA 証明書があります。

- **--mkhomedir** オプションを使用して IdM クライアントをインストールし、リモートユーザーが正常にログインできるようにしている。ホームディレクトリーを作成しない場合、デフォルトのログイン場所はディレクトリー構造のルート / になります。

手順

1. IdM サーバーで、管理者権限を使用して、**ipa-advise** で設定スクリプトを生成します。

```
[root@server SmartCard]# kinit admin
[root@server SmartCard]# ipa-advise config-client-for-smart-card-auth > config-client-for-smart-card-auth.sh
```

config-client-for-smart-card-auth.sh スクリプトは、以下の操作を実行します。

- スマートカードデーモンを設定する。
- システム全体のトラストストアを設定します。
- System Security Services Daemon (SSSD) を設定して、ユーザーがユーザー名とパスワード、またはスマートカードで認証できるようにします。スマートカード認証用の SSSD プロファイルオプションの詳細は、RHEL のスマートカード認証オプション を参照してください。

2. IdM サーバーから、IdM クライアントマシンの任意のディレクトリーに、スクリプトをコピーします。

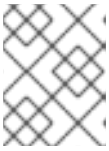
```
[root@server SmartCard]# scp config-client-for-smart-card-auth.sh root@client.idm.example.com:/root/SmartCard/
Password:
config-client-for-smart-card-auth.sh        100%   2419       3.5MB/s   00:00
```

3. 便宜上、IdM サーバーから、上の手順で使用した IdM クライアントマシンのディレクトリーに、PEM 形式の CA 証明書ファイルをコピーします。

```
[root@server SmartCard]# scp {rootca.pem,subca.pem,issuingca.pem} root@client.idm.example.com:/root/SmartCard/
Password:
rootca.pem                    100%   1237     9.6KB/s   00:00
subca.pem                     100%   2514    19.6KB/s   00:00
issuingca.pem                 100%   2514    19.6KB/s   00:00
```

4. クライアントマシンで、スクリプトを実行し、CA 証明書を含む PEM ファイルを引数として追加します。

```
[root@client SmartCard]# kinit admin
[root@client SmartCard]# chmod +x config-client-for-smart-card-auth.sh
[root@client SmartCard]# ./config-client-for-smart-card-auth.sh rootca.pem subca.pem issuingca.pem
Ticket cache:KEYRING:persistent:0:0
Default principal: admin@IDM.EXAMPLE.COM
[...]
Systemwide CA database updated.
The ipa-certupdate command was successful
```



注記

ルート CA 証明書を、サブ CA 証明書の前に引数として追加します。また、CA またはサブ CA 証明書の有効期限が切れていないことを確認します。

これで、クライアントがスマートカード認証に対して設定されました。

66.4. ANSIBLE を使用したスマートカード認証用の IDM クライアント設定

ansible-freeipa ipasmartcard_client モジュールを使用して特定の Identity Management (IdM) クライアントを設定し、IdM ユーザーがスマートカードで認証できるようにするには、次の手順に従います。この手順を実行し、以下のいずれかを使用して IdM にアクセスする IdM ユーザーのスマートカード認証を有効にします。

- **ssh** プロトコル
  詳細は、スマートカード認証で SSH アクセスの設定 を参照してください。
- コンソールのログイン
- GNOME Display Manager (GDM)
- **su** コマンド

注記

この手順は、IdM Web UI に対する認証には必要ありません。IdM Web UI の認証には 2 つのホストが関係しますが、どちらも IdM クライアントである必要はありません。

- ブラウザーが実行されているマシン。マシンは IdM ドメインの外にある場合があります。
- **httpd** が実行している IdM サーバー

前提条件

- Ansible を使用したスマートカード認証用の IdM サーバー設定 に説明されているとおり、IdM サーバーがスマートカード認証用に設定されている。
- IdM サーバーと IdM クライアントに root アクセス権限がある。
- ルート CA 証明書、IdM CA 証明書、すべての中間 CA の証明書がある。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. CA 証明書が **DER** などをはじめとする別の形式のファイルに保存されている場合、それらを **PEM** 形式に変換します。

```
# openssl x509 -in <filename>.der -inform DER -out <filename>.pem -outform PEM
```

IdM CA 証明書は **PEM** 形式で、**/etc/ipa/ca.crt** ファイルにあります。

2. 必要に応じて、**openssl x509** ユーティリティーを使用して **PEM** 形式のファイルの内容を表示し、**Issuer** および **Subject** の値が正しいことを確認します。

```
# openssl x509 -noout -text -in root-ca.pem | more
```

3. Ansible コントロールノードで、**~/MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

4. CA 証明書専用のサブディレクトリーを作成します。

```
$ mkdir SmartCard/
```

5. 便宜上、必要なすべての証明書を **~/MyPlaybooks/SmartCard/** ディレクトリーにコピーします。以下はその例です。

```
# cp /tmp/root-ca.pem ~/MyPlaybooks/SmartCard/
# cp /tmp/intermediate-ca.pem ~/MyPlaybooks/SmartCard/
# cp /etc/ipa/ca.crt ~/MyPlaybooks/SmartCard/ipa-ca.crt
```

6. Ansible インベントリーファイルで、以下を指定します。

- スマートカード認証用に設定する IdM クライアント。
- IdM 管理者パスワード。
- CA の証明書へのパス (次の順序に従う)。
  - ルート CA 証明書ファイル
  - 中間 CA 証明書ファイル
  - IdM CA 証明書ファイル

ファイルは次のようになります。

```
[ipaclients]
ipaclient1.example.com
ipaclient2.example.com

[ipaclients:vars]
ipaadmin_password=SomeADMINpassword
ipasmartcard_client_ca_certs=/home/<user_name>/MyPlaybooks/SmartCard/root-ca.pem,/home/<user_name>/MyPlaybooks/SmartCard/intermediate-ca.pem,/home/<user_name>/MyPlaybooks/SmartCard/ipa-ca.crt
```

7. 次の内容で **install-smartcard-clients.yml** playbook を作成します。

```
---
- name: Playbook to set up smart card authentication for an IdM client
  hosts: ipaclients
  become: true

  roles:
  - role: ipasmartcard_client
    state: present
```

8. ファイルを保存します。

9. Ansible Playbook を実行します。Playbook とインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory install-
smartcard-clients.yml
```

ipasmartcard_client Ansible ロールは、次のアクションを実行します。

- スマートカードデーモンを設定する。
- システム全体のトラストストアを設定します。
- System Security Services Daemon (SSSD) を設定して、ユーザーがユーザー名とパスワード、またはスマートカードで認証できるようにします。スマートカード認証用の SSSD プロファイルオプションの詳細は、RHEL のスマートカード認証オプション を参照してください。

これで、インベントリーファイルの **ipaclients** セクションにリストされているクライアントがスマートカード認証用に設定されました。

注記

--mkhomedir オプションを使用して IdM クライアントをインストールしている場合、リモートユーザーはホームディレクトリーにログインできます。それ以外の場合、デフォルトのログイン場所はディレクトリー構造のルート / です。

関連情報

- **/usr/share/doc/ansible-freeipa/playbooks/** ディレクトリーで **ipasmartcard_server** ロールを使用するサンプル Playbook

66.5. IDM WEB UI のユーザーエントリーへの証明書の追加

IdM Web UI のユーザーエントリーに外部証明書を追加するには、以下の手順に従います。

注記

証明書全体をアップロードする代わりに、IdM のユーザーエントリーに証明書マッピングデータをアップロードすることもできます。システム管理者向けのスマートカード認証の設定を容易にするために、完全な証明書または証明書マッピングデータのいずれかが含まれるユーザーエントリーを、対応する証明書マッピングルールと併用できます。詳細は以下を参照してください。

認証を設定するための証明書マッピングルール

注記

ユーザーの証明書が IdM 認証局によって発行された場合、証明書はユーザーエントリーにすでに保存されているため、この手順に従う必要はありません。

前提条件

- ユーザーエントリーに追加できる証明書がある。

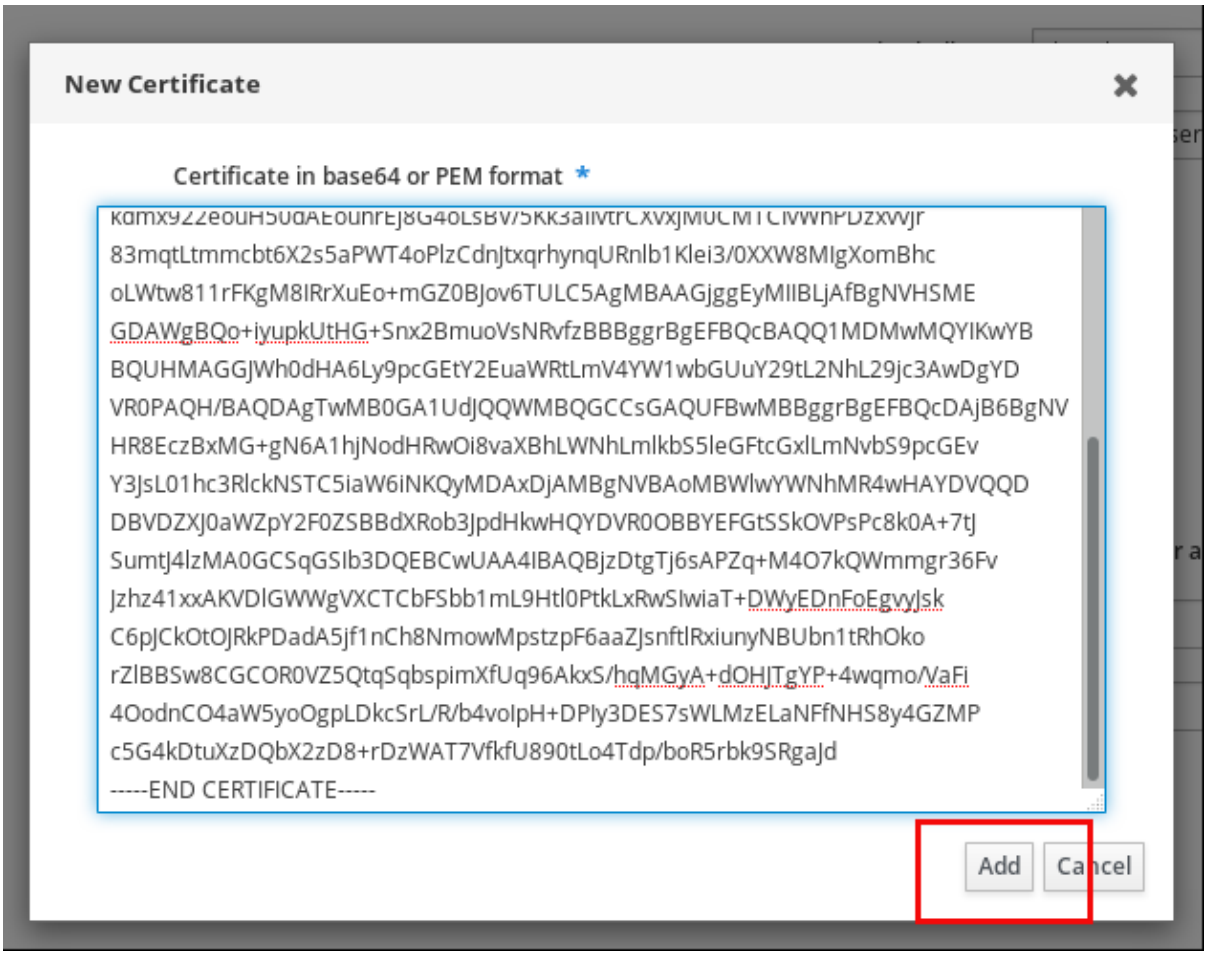
手順

1. 別のユーザーに証明書を追加する場合は、管理者として IdM Web UI にログインします。独自のプロファイルに証明書を追加する場合は、管理者の認証情報が必要ありません。
2. **Users → Active users → sc_user** の順に移動します。
3. **Certificate** オプションを探して、**Add** をクリックします。
4. コマンドラインインターフェイスで、**cat** ユーティリティーまたはテキストエディターを使用して、**PEM** 形式の証明書を表示します。

```
[user@client SmartCard]$ cat testuser.crt
```

5. CLI で、証明書をコピーし、Web UI で開いたウィンドウにこれを貼り付けます。
6. **Add** をクリックします。

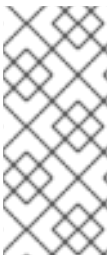
図66.1 IdM Web UI で新しい証明書の追加



sc_user エントリーに外部証明書が含まれるようになりました。

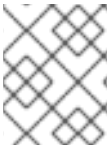
66.6. IDM CLI でユーザーエントリーへの証明書の追加

IdM CLI のユーザーエントリーに外部証明書を追加するには、以下の手順に従います。



注記

証明書全体をアップロードする代わりに、IdM のユーザーエントリーに証明書マッピングデータをアップロードすることもできます。システム管理者向けのスマートカード認証の設定を容易にするために、完全な証明書または証明書マッピングデータのいずれかが含まれるユーザーエントリーを、対応する証明書マッピングルールと併用できます。詳細は、認証を設定するための証明書マッピングルール を参照してください。



注記

ユーザーの証明書が IdM 認証局によって発行された場合、証明書はユーザーエントリーにすでに保存されているため、この手順に従う必要はありません。

前提条件

- ユーザーエントリーに追加できる証明書がある。

手順

1. 別のユーザーに証明書を追加する場合は、管理者として IdM Web CLI にログインします。

```
[user@client SmartCard]$ kinit admin
```

独自のプロファイルに証明書を追加する場合は、管理者の認証情報は必要ありません。

```
[user@client SmartCard]$ kinit sc_user
```

2. ヘッダーとフッターのある証明書を含む環境変数を作成し、1 行に連結します。これは、**ipa user-add-cert** コマンドで必要な形式です。

```
[user@client SmartCard]$ export CERT=`openssl x509 -outform der -in testuser.crt | base64 -w0 -`
```

testuser.crt ファイルの証明書は、**PEM** 形式である必要があることに注意してください。

3. **ipa user-add-cert** コマンドを使用して、sc_user のプロファイルに証明書を追加します。

```
[user@client SmartCard]$ ipa user-add-cert sc_user --certificate=$CERT
```

sc_user エントリーに外部証明書が含まれるようになりました。

66.7. スマートカードを管理および使用するツールのインストール

前提条件

- **gnutls-utils** パッケージがインストールされている。
- **opensc** パッケージがインストールされている。
- **pcscd** サービスを実行している。

スマートカードを設定する前に、対応するツール (証明書を生成して **pscd** サービスを起動できるもの) をインストールする必要があります。

手順

1. **opensc** パッケージおよび **gnutls-utils** パッケージをインストールします。

```
# {PackageManagerCommand} -y install opensc gnutls-utils
```

2. **pcscd** サービスを開始します。

```
# systemctl start pcscd
```

検証手順

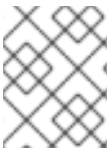
- **pcscd** サービスが稼働していることを確認します。

```
# systemctl status pcscd
```

66.8. スマートカードを準備し、証明書と鍵をスマートカードにアップロードする

pkcs15-init ツールを使用してスマートカードを設定するには、この手順に従います。このツールは、以下を設定するのに役立ちます。

- スマートカードの消去
- 新しい PIN および任意の PIN ブロック解除キー (PUK) の設定
- スマートカードでの新規スロットの作成
- スロットへの証明書、秘密鍵、および公開鍵の保存
- 必要に応じて、特定のスマートカードではこのタイプのファイナライズが必要なため、スマートカードの設定をロックします。



注記

pkcs15-init ツールは、すべてのスマートカードで機能するとは限りません。使用しているスマートカードで動作するツールを使用する必要があります。

前提条件

- **pkcs15-init** ツールを含む **opensc** パッケージがインストールされている。
  詳細は スマートカードを管理および使用するツールのインストール を参照してください。
- カードがリーダーに挿入され、コンピューターに接続されている。
- スマートカードに保存する秘密鍵、公開鍵、および証明書がある。この手順の **testuser.key**、**testuserpublic.key**、および **testuser.crt** は、秘密鍵、公開鍵、および証明書に使用される名前です。
- 現在のスマートカードユーザー PIN およびセキュリティーオフィス PIN (SO-PIN)

手順

1. スマートカードを消去して PIN で自身を認証します。

```
$ pkcs15-init --erase-card --use-default-transport-keys
Using reader with a card: Reader name
PIN [Security Officer PIN] required.
Please enter PIN [Security Officer PIN]:
```

   カードが削除されました。

2. スマートカードを初期化し、ユーザーの PIN と PUK を設定します。また、セキュリティー担当者の PIN と PUK を設定します。

```
$ pkcs15-init --create-pkcs15 --use-default-transport-keys \ --pin 963214 --puk 321478 --so-pin 65498714 --so-puk 784123
Using reader with a card: Reader name
```

   pcks15-init ツールは、スマートカードに新しいスロットを作成します。

3. スロットのラベルと認証 ID を設定します。

```
$ pkcs15-init --store-pin --label testuser \ --auth-id 01 --so-pin 65498714 --pin 963214 --puk
321478
Using reader with a card: Reader name
```

ラベルは人間が判読できる値に設定されます (この場合は **testuser**)。**auth-id** は 16 進数の値である必要があります。この場合、**01** に設定されます。

4. スマートカードの新しいスロットに秘密鍵を保存し、ラベルを付けます。

```
$ pkcs15-init --store-private-key testuser.key --label testuser_key \ --auth-id 01 --id 01 --pin
963214
Using reader with a card: Reader name
```

注記

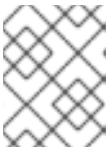
--id に指定する値は、秘密鍵を保存するときと、次の手順で証明書を保存するときに同じである必要があります。**--id** に独自の値を指定することを推奨します。そうしないと、より複雑な値がツールによって計算されます。

5. スマートカードの新しいスロットに証明書を保存し、ラベル付けします。

```
$ pkcs15-init --store-certificate testuser.crt --label testuser_crt \ --auth-id 01 --id 01 --format
pem --pin 963214
Using reader with a card: Reader name
```

6. (必要に応じて) スマートカードの新しいスロットに公開鍵を保存し、ラベルを付けます。

```
$ pkcs15-init --store-public-key testuserpublic.key --label testuserpublic_key --auth-id 01 --id
01 --pin 963214
Using reader with a card: Reader name
```



注記

公開鍵が秘密鍵または証明書に対応する場合は、秘密鍵または証明書の ID と同じ ID を指定します。

7. (オプション) スマートカードの中には、設定をロックしてカードを最終処理する必要があります。

```
$ pkcs15-init -F
```

この段階では、スマートカードには、新たに作成されたスロットに証明書、秘密鍵、および公開鍵が含まれます。ユーザーの PIN と PUK、およびセキュリティー担当者の PIN と PUK も作成しました。

66.9. スマートカードを使用して IDM へのログイン

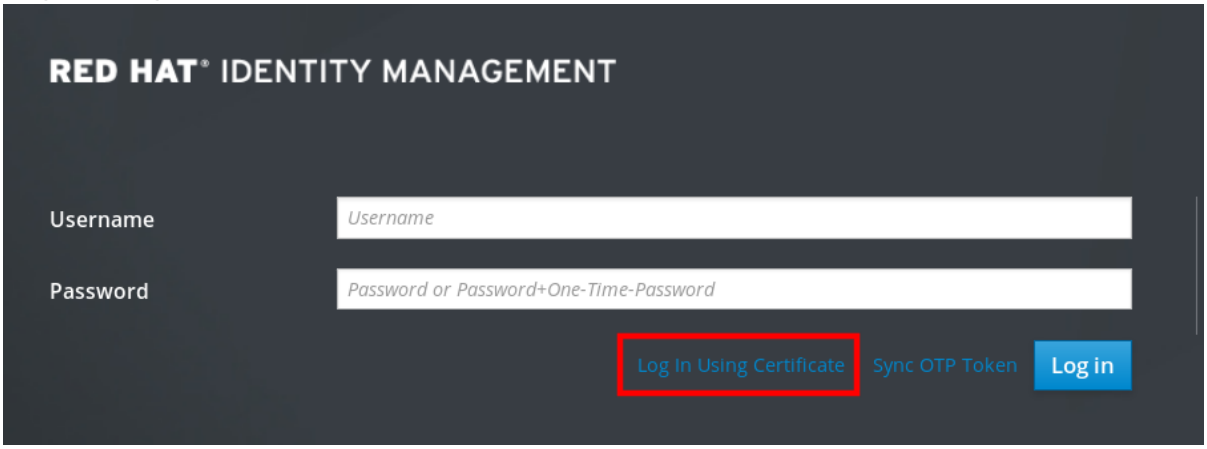
IdM Web UI へのログインにスマートカードを使用するには、以下の手順に従います。

前提条件

- Web ブラウザーが、スマートカード認証を使用できるように設定されている。
- IdM サーバーはスマートカード認証用に設定されています。
- スマートカードにインストールされた証明書は、IdM サーバーによって発行されるか、IdM のユーザーエントリーに追加されています。
- スマートカードのロックを解除するために必要な PIN を知っています。
- スマートカードがリーダーに挿入されました。

手順

1. ブラウザーで IdM Web UI を開きます。
2. **Log In Using Certificate** をクリックします。

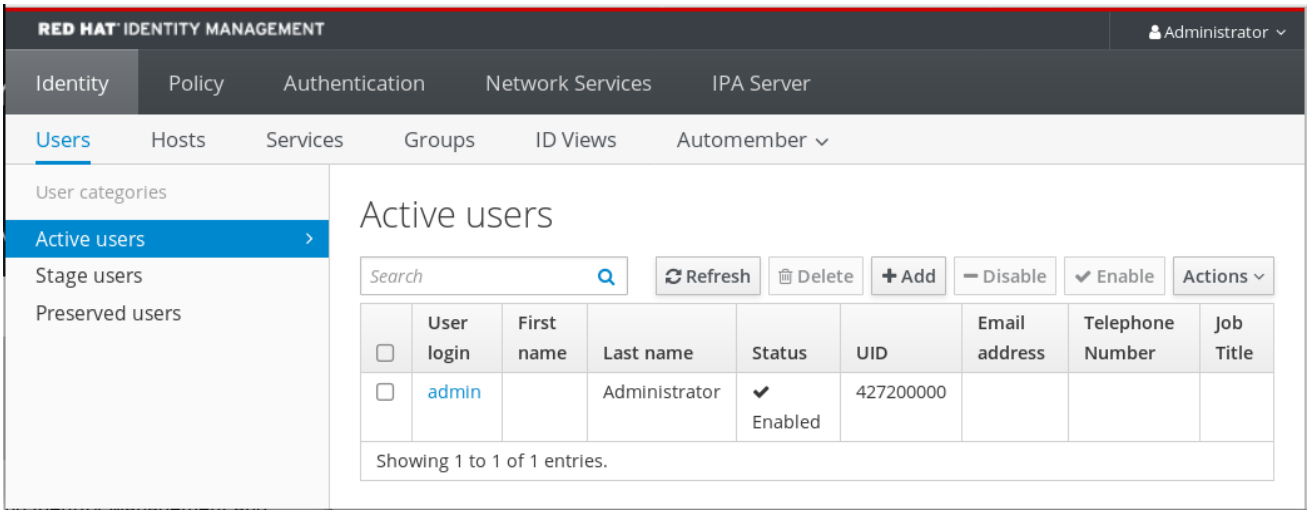


3. **Password Required** ダイアログボックスが開いたら、スマートカードのロックを解除する PIN を追加して、**OK** ボタンをクリックします。
User Identification Request ダイアログボックスが開きます。

   スマートカードに複数の証明書が含まれている場合は、**Choose a certificate to present as identification** の下にあるドロップダウンリストで、認証に使用する証明書を選択します。

4. **OK** ボタンをクリックします。

これで、IdM Web UI に正常にログインできるようになりました。



66.10. IDM クライアントでスマートカード認証を使用して GDM にログインする

GNOME Desktop Manager (GDM) には認証が必要です。パスワードは使用できますが、認証にスマートカードを使用することもできます。

以下の手順に従って、スマートカード認証を使用して GDM にアクセスします。

前提条件

- システムはスマートカード認証用に設定されています。詳細は、スマートカード認証用の IdM クライアントの設定 を参照してください。
- スマートカードに、証明書と秘密鍵が含まれている。
- ユーザーアカウントは、IdM ドメインのメンバーです。
- スマートカードの証明書は、以下を使用してユーザーエントリーにマッピングします。
  - 特定のユーザーエントリーへの証明書の割り当て。詳細は Adding a certificate to a user entry in the IdM Web UI または Adding a certificate to a user entry in the IdM CLI を参照してください。
  - アカウントに適用される証明書マッピングデータ。詳細は、スマートカードにおける認証を設定するための証明書マッピングルール を参照してください。

手順

1. スマートカードをリーダーに挿入します。
2. スマートカード PIN を入力します。
3. **Sign In** をクリックします。

RHEL システムにログインし、IdM サーバーが提供する TGT がある。

検証手順

- **端末** ウィンドウで **klist** を入力し、結果を確認します。

```
$ klist
Ticket cache: KEYRING:persistent:1358900015:krb_cache_TObtNMd
Default principal: example.user@REDHAT.COM

Valid starting       Expires              Service principal
04/20/2020 13:58:24  04/20/2020 23:58:24  krbtgt/EXAMPLE.COM@EXAMPLE.COM
 renew until 04/27/2020 08:58:15
```

66.11. SU コマンドでのスマートカード認証の使用

別のユーザーへの変更には認証が必要です。パスワードまたは証明書を使用できます。以下の手順に従って、**su** コマンドでスマートカードを使用します。これは、**su** コマンドを入力すると、スマートカード PIN の入力が求められます。

前提条件

- IdM サーバーとクライアントがスマートカード認証用に設定されました。
  - スマートカード認証用の IdM サーバーの設定 を参照してください。
  - スマートカード認証用の IdM クライアントの設定 を参照してください。
- スマートカードに、証明書と秘密鍵が含まれている。スマートカードでの証明書の保存 を参照してください。
- カードがリーダーに挿入され、コンピューターに接続されている。

手順

- 端末で、**su** コマンドで別のユーザーに移動します。

```
$ su - example.user
PIN for smart_card
```

  設定が正しい場合は、スマートカードの PIN を入力するよう求められます。

第67章 IDM でスマートカード認証用に ADCS が発行した証明書の設定

Active Directory (AD)証明書サービスが証明書を発行するユーザーに IdM でスマートカード認証を設定するには、次のコマンドを実行します。

- デプロイメントは、Identity Management (IdM) と Active Directory (AD) と間のフォレスト間の信頼に基づいている場合
- AD にアカウントが保存されているユーザーに対してスマートカード認証を許可したい場合
- 証明書が作成され、Active Directory Certificate Services (ADCS) に保存される場合

スマートカード認証の概要については、スマートカード認証について を参照してください。

設定は、次の手順で行われます。

- Active Directory から IdM サーバーおよびクライアントへの CA 証明書およびユーザー証明書のコピー
- ADCS 証明書を使用したスマートカード認証用の IdM サーバーおよびクライアントの設定
- 証明書および秘密鍵をスマートカードに保存できるように PFX (PKCS#12) ファイルの変換
- sssd.conf ファイルでのタイムアウトの設定
- スマートカード認証用の証明書マッピングルールの作成

前提条件

- Identity Management (IdM) および Active Directory (AD) 信頼がインストールされている。詳細は、IdM と AD との間の信頼のインストール を参照してください。
- Active Directory 証明書サービス (ADCS) がインストールされ、ユーザーの証明書が生成されている。

67.1. 信頼の設定と証明書の使用に必要な WINDOWS SERVER 設定

Windows Server で以下を設定する必要があります。

- Active Directory 証明書サービス (ADCS) がインストールされる
- 認証局が作成される
- 必要に応じて、認証機関の Web 登録を使用している場合は、IIS (Internet Information Services) を設定する必要がある。

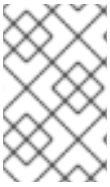
証明書をエクスポートします。

- 鍵には **2048** ビット以上が必要
- 秘密鍵を含める
- Personal Information Exchange (**PKCS #12(.PFX)**) の形式の証明書が必要
    - 証明書のプライバシーを有効にする

67.2. SFTP を使用して ACTIVE DIRECTORY から証明書のコピー

スマートカード認証を使用できるようにするには、次の証明書ファイルをコピーする必要があります。

- IdM サーバーにある **CER** 形式のルート CA 証明書 (**adcs-winserver-ca.cer**)
- IdM クライアントの **PFX** 形式の秘密鍵を持つユーザー証明書 (**aduser1.pfx**)



注記

この手順では、SSH アクセスが許可されていることを想定しています。SSH が使用できない場合、ユーザーは AD サーバーから IdM サーバーおよびクライアントにファイルをコピーする必要があります。

手順

1. **IdM サーバー** から接続し、**adcs-winserver-ca.cer** ルート証明書を IdM サーバーにコピーします。

```
root@idmserver ~]# sftp Administrator@winserver.ad.example.com
Administrator@winserver.ad.example.com's password:
Connected to Administrator@winserver.ad.example.com.
sftp> cd <Path to certificates>
sftp> ls
adcs-winserver-ca.cer    aduser1.pfx
sftp>
sftp> get adcs-winserver-ca.cer
Fetching <Path to certificates>/adcs-winserver-ca.cer to adcs-winserver-ca.cer
<Path to certificates>/adcs-winserver-ca.cer                100%  1254    15KB/s 00:00
sftp quit
```

2. **IdM クライアント** から接続し、**aduser1.pfx** ユーザー証明書をクライアントにコピーします。

```
[root@client1 ~]# sftp Administrator@winserver.ad.example.com
Administrator@winserver.ad.example.com's password:
Connected to Administrator@winserver.ad.example.com.
sftp> cd /<Path to certificates>
sftp> get aduser1.pfx
Fetching <Path to certificates>/aduser1.pfx to aduser1.pfx
<Path to certificates>/aduser1.pfx                100%  1254    15KB/s 00:00
sftp quit
```

これで、CA 証明書は IdM サーバーに保存され、ユーザー証明書はクライアントマシンに保存されます。

67.3. ADCS 証明書を使用したスマートカード認証用の IDM サーバーおよびクライアントの設定

IdM 環境でスマートカード認証を使用できるように、IdM (Identity Management) サーバーおよびクライアントを設定する必要があります。IdM には、必要なすべての変更を行う **ipa-advise** スクリプトが含まれています。

- 必要なパッケージをインストールする

- IdM サーバーおよびクライアントの設定
- CA 証明書を予想される場所にコピーします。

IdM サーバーで **ipa-advise** を実行できるようになります。

以下の手順に従って、スマートカード認証用にサーバーとクライアントを設定します。

- IdM サーバー - **ipa-advise** スクリプトを準備して、スマートカード認証用に IdM サーバーを設定します。
- IdM サーバー - **ipa-advise** スクリプトを準備して、スマートカード認証用に IdM クライアントを設定します。
- IdM サーバー - AD 証明書を使用して IdM サーバーに **ipa-advise** サーバースクリプトを適用します。
- クライアントスクリプトを IdM クライアントマシンに移動します。
- IdM サーバー - AD 証明書を使用して IdM クライアントに **ipa-advise** クライアントスクリプトを適用します。

前提条件

- 証明書が IdM サーバーにコピーされている。
- Kerberos チケットを取得している。
- 管理者権限を持つユーザーとしてログインしている。

手順

1. IdM サーバーで、クライアントを設定する **ipa-advise** スクリプトを使用します。

```
[root@idmserver ~]# ipa-advise config-client-for-smart-card-auth > sc_client.sh
```

2. IdM サーバーで、サーバーを設定する **ipa-advise** スクリプトを使用します。

```
[root@idmserver ~]# ipa-advise config-server-for-smart-card-auth > sc_server.sh
```

3. IdM サーバーで、スクリプトを実行します。

```
[root@idmserver ~]# sh -x sc_server.sh adcs-winserver-ca.cer
```

   - IdM Apache HTTP サーバーを設定します。
   - キー配布センター (KDC) の Kerberos (PKINIT) で、初回認証用の公開鍵暗号化機能を有効にします。
   - スマートカード認可要求を受け入れるように IdM Web UI を設定します。

4. **sc_client.sh** スクリプトをクライアントシステムにコピーします。

```
[root@idmserver ~]# scp sc_client.sh root@client1.idm.example.com:/root
Password:
sc_client.sh                  100%  2857   1.6MB/s   00:00
```

5. Windows 証明書をクライアントシステムにコピーします。

```
[root@idmserver ~]# scp adcs-winserver-ca.cer root@client1.idm.example.com:/root
Password:
adcs-winserver-ca.cer                 100%  1254   952.0KB/s   00:00
```

6. クライアントシステムで、クライアントスクリプトを実行します。

```
[root@idmclient1 ~]# sh -x sc_client.sh adcs-winserver-ca.cer
```

CA 証明書が IdM サーバーとクライアントシステムに正しい形式でインストールされました。次の手順は、ユーザー証明書をスマートカード自体にコピーすることです。

67.4. PFX ファイルの変換

PFX (PKCS#12) ファイルをスマートカードに保存する前に、以下を行う必要があります。

- ファイルを PEM 形式に変換する。
- 秘密鍵と証明書を 2 つの異なるファイルに抽出する。

前提条件

- PFX ファイルが IdM クライアントマシンにコピーされます。

手順

1. IdM クライアントで、PEM 形式に変換します。

```
[root@idmclient1 ~]# openssl pkcs12 -in aduser1.pfx -out aduser1_cert_only.pem -clcerts -
nodes
Enter Import Password:
```

2. 鍵を別のファイルにデプロイメントします。

```
[root@idmclient1 ~]# openssl pkcs12 -in adduser1.pfx -nocerts -out adduser1.pem >
aduser1.key
```

3. パブリック証明書を別のファイルにデプロイメントします。

```
[root@idmclient1 ~]# openssl pkcs12 -in adduser1.pfx -clcerts -nokeys -out
aduser1_cert_only.pem > aduser1.crt
```

この時点で、**aduser1.key** および **aduser1.crt** をスマートカードに保存できます。

67.5. スマートカードを管理および使用するツールのインストール

前提条件

- **gnutls-utils** パッケージがインストールされている。
- **opensc** パッケージがインストールされている。
- **pcscd** サービスを実行している。

スマートカードを設定する前に、対応するツール (証明書を生成して **pscd** サービスを起動できるもの) をインストールする必要があります。

手順

1. **opensc** パッケージおよび **gnutls-utils** パッケージをインストールします。

```
# {PackageManagerCommand} -y install opensc gnutls-utils
```

2. **pcscd** サービスを開始します。

```
# systemctl start pcscd
```

検証手順

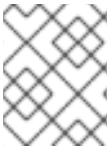
- **pcscd** サービスが稼働していることを確認します。

```
# systemctl status pcscd
```

67.6. スマートカードを準備し、証明書と鍵をスマートカードにアップロードする

pkcs15-init ツールを使用してスマートカードを設定するには、この手順に従います。このツールは、以下を設定するのに役立ちます。

- スマートカードの消去
- 新しい PIN および任意の PIN ブロック解除キー (PUK) の設定
- スマートカードでの新規スロットの作成
- スロットへの証明書、秘密鍵、および公開鍵の保存
- 必要に応じて、特定のスマートカードではこのタイプのファイナライズが必要なため、スマートカードの設定をロックします。



注記

pkcs15-init ツールは、すべてのスマートカードで機能するとは限りません。使用しているスマートカードで動作するツールを使用する必要があります。

前提条件

- **pkcs15-init** ツールを含む **opensc** パッケージがインストールされている。
  詳細は スマートカードを管理および使用するツールのインストール を参照してください。

- カードがリーダーに挿入され、コンピューターに接続されている。
- スマートカードに保存する秘密鍵、公開鍵、および証明書がある。この手順の **testuser.key**、**testuserpublic.key**、および **testuser.crt** は、秘密鍵、公開鍵、および証明書に使用される名前です。
- 現在のスマートカードユーザー PIN およびセキュリティーオフィス PIN (SO-PIN)

手順

1. スマートカードを消去して PIN で自身を認証します。

```
$ pkcs15-init --erase-card --use-default-transport-keys
Using reader with a card: Reader name
PIN [Security Officer PIN] required.
Please enter PIN [Security Officer PIN]:
```

   カードが削除されました。

2. スマートカードを初期化し、ユーザーの PIN と PUK を設定します。また、セキュリティー担当者の PIN と PUK を設定します。

```
$ pkcs15-init --create-pkcs15 --use-default-transport-keys \ --pin 963214 --puk 321478 --so-pin 65498714 --so-puk 784123
Using reader with a card: Reader name
```

   pcks15-init ツールは、スマートカードに新しいスロットを作成します。

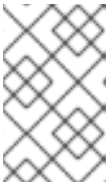
3. スロットのラベルと認証 ID を設定します。

```
$ pkcs15-init --store-pin --label testuser \ --auth-id 01 --so-pin 65498714 --pin 963214 --puk 321478
Using reader with a card: Reader name
```

   ラベルは人間が判読できる値に設定されます (この場合は **testuser**)。**auth-id** は 16 進数の値である必要があります。この場合、**01** に設定されます。

4. スマートカードの新しいスロットに秘密鍵を保存し、ラベルを付けます。

```
$ pkcs15-init --store-private-key testuser.key --label testuser_key \ --auth-id 01 --id 01 --pin 963214
Using reader with a card: Reader name
```



注記

--id に指定する値は、秘密鍵を保存するときと、次の手順で証明書を保存するときに同じである必要があります。**--id** に独自の値を指定することを推奨します。そうしないと、より複雑な値がツールによって計算されます。

5. スマートカードの新しいスロットに証明書を保存し、ラベル付けします。

```
$ pkcs15-init --store-certificate testuser.crt --label testuser_crt \ --auth-id 01 --id 01 --format pem --pin 963214
Using reader with a card: Reader name
```

6. (必要に応じて) スマートカードの新しいスロットに公開鍵を保存し、ラベルを付けます。

```
$ pkcs15-init --store-public-key testuserpublic.key --label testuserpublic_key --auth-id 01 --id
01 --pin 963214
Using reader with a card: Reader name
```

注記

公開鍵が秘密鍵または証明書に対応する場合は、秘密鍵または証明書の ID と同じ ID を指定します。

7. (オプション) スマートカードの中には、設定をロックしてカードを最終処理する必要があります。

```
$ pkcs15-init -F
```

この段階では、スマートカードには、新たに作成されたスロットに証明書、秘密鍵、および公開鍵が含まれます。ユーザーの PIN と PUK、およびセキュリティー担当者の PIN と PUK も作成しました。

67.7. SSSD.CONF でタイムアウトの設定

スマートカード証明書による認証は、SSSD で使用されるデフォルトのタイムアウトよりも時間がかかる場合があります。タイムアウトの期限切れは次の原因で発生する可能性があります。

- 読み込みが遅い
- 物理デバイスから仮想環境への転送
- スマートカードに保存されている証明書が多すぎる
- OCSP (Online Certificate Status Protocol) を使用して証明書を検証する場合は、OCSP レスポンダーからの応答が遅い

この場合は、**sssd.conf** ファイルにある次のタイムアウトを、たとえば 60 秒まで延長できます。

- **p11_child_timeout**
- **krb5_auth_timeout**

前提条件

- root としてログインしている。

手順

1. **sssd.conf** ファイルを開きます。

```
[root@idmclient1 ~]# vim /etc/sssd/sssd.conf
```

2. **p11_child_timeout** の値を変更します。

```
[pam]
p11_child_timeout = 60
```

3. **krb5_auth_timeout** の値を変更します。

```
[domain/IDM.EXAMPLE.COM]
krb5_auth_timeout = 60
```

4. 設定を保存します。

現在、スマートカードとの相互作用は 1 分間 (60 秒) 実行でき、その後、認証はタイムアウトで失敗します。

67.8. スマートカード認証用の証明書マッピングルールの作成

AD (Active Directory) および IdM (Identity Management) にアカウントを持つユーザーに対して証明書を 1 つ使用する場合は、IdM サーバーで証明書マッピングルールを作成できます。

このようなルールを作成すると、ユーザーは両方のドメインのスマートカードで認証できます。

証明書マッピングルールの詳細は、認証を設定するための証明書マッピングルール を参照してください。

第68章 IDENTITY MANAGEMENT での証明書マッピングルールの設定

証明書マッピングルールは、Identity Management (IdM) 管理者が特定のユーザーの証明書にアクセスしない場合に、シナリオで証明書を使用して認証できるため便利な方法です。これは通常、証明書が外部の認証局によって発行されたことが原因です。

68.1. 認証を設定するための証明書マッピングルール

次のシナリオでは、証明書マッピングルールの設定が必要になる場合があります。

- 証明書は、IdM ドメインが信頼関係にある Active Directory (AD) の証明書システムによって発行されています。
- 証明書は外部の認証局によって発行されています。
- IdM 環境は大規模で、多くのユーザーがスマートカードを使用しています。この場合、完全な証明書の追加は複雑になる可能性があります。件名と発行者はほとんどのシナリオで予測可能なため、完全な証明書よりも事前に追加する方が簡単です。

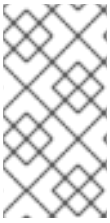
システム管理者は、証明書マッピングルールを作成し、特定のユーザーに証明書を発行する前に、ユーザーエントリーに証明書マッピングデータを追加できます。証明書を発行すると、完全な証明書がまだユーザーエントリーにアップロードされていなくても、ユーザーは証明書を使用してログインできます。

さらに、証明書は定期的に更新されるため、証明書マッピングルールにより管理オーバーヘッドが削減されます。ユーザーの証明書が更新される場合、管理者はユーザーエントリーを更新する必要はありません。たとえば、マッピングが **Subject** と **Issuer** の値に基づいている場合、および新しい証明書の Subject と Issuer が以前と同じ場合は、マッピングは引き続き適用されます。一方で、完全な証明書を使用した場合、管理者は古い証明書に置き換わる新しい証明書をユーザーエントリーにアップロードする必要があります。

証明書マッピングを設定するには、以下を実行します。

1. 管理者は、証明書マッピングデータまたは完全な証明書をユーザーアカウントにロードする必要があります。
2. 管理者は、証明書の情報と一致する証明書マッピングデータエントリーがアカウントに含まれているユーザーが IdM に正常にログインできるように、証明書マッピングルールを作成する必要があります。

証明書マッピングルールが作成されると、エンドユーザーが ファイルシステム または スマートカード に保存されている証明書を提示すると、認証は成功します。



注記

キー配布センター (KDC) には、証明書マッピングルールのキャッシュがあります。キャッシュは最初の **certauth** 要求の際に入力され、タイムアウトは 300 秒にハードコーディングされています。KDC は、再起動するかキャッシュが期限切れにならない限り、証明書マッピングルールへの変更を認識しません。

マッピングルールを設定する個々のコンポーネントの詳細と、そのコンポーネントの取得方法および使用方法は、IdM における ID マッピングルールのコンポーネント および マッチングルールで使用する証明書からの発行者の取得 を参照してください。

注記

証明書マッピングルールは、証明書を使用するユースケースに応じて異なります。たとえば、証明書を使用して SSH を使用している場合、証明書から公開鍵を抽出するには完全な証明書が必要です。

68.2. IDM における ID マッピングルールのコンポーネント

IdM で **ID マッピングルール** を作成する際に、異なるコンポーネントを設定します。各コンポーネントには、オーバーライドできるデフォルト値があります。コンポーネントは、Web UI または CLI のいずれかで定義できます。CLI では、**ipa certmaprule-add** コマンドを使用して、ID マッピングルールが作成されます。

マッピングルール

マッピングルールコンポーネントでは、証明書を 1 人または複数のユーザーアカウントに関連付けます (または **マップ** します)。ルールは、証明書を目的のユーザーアカウントに関連付ける LDAP 検索フィルターを定義します。
さまざまな認証局 (CA) が発行する証明書にはさまざまなプロパティーがあり、さまざまなドメインで使用される可能性があります。そのため、IdM はマッピングルールを無条件に適用せず、適切な証明書にのみ適用されます。適切な証明書は、**マッチングルール** を使用して定義されます。

マッピングルールのオプションを空のままにすると、証明書は、DER でエンコードされたバイナリーファイルとして、**userCertificate** 属性で検索されることに注意してください。

--maprule オプションを使用して、CLI でマッピングルールを定義します。

マッチングルール

マッチングルールコンポーネントは、マッピングルールを適用する証明書を選択します。デフォルトのマッチングルールは、**digitalSignature 鍵** の使用と、**clientAuth 拡張鍵** の使用で、証明書と一致します。
--matchrule オプションを使用して、CLI にマッチングルールを定義します。

ドメインリスト

ドメインリストは、ID マッピングルールの処理時に IdM がユーザーを検索する ID ドメインを指定します。このオプションを指定しないと、IdM は、IdM クライアントが所属しているローカルドメイン内でのみユーザーを検索します。
--domain オプションを使用して CLI にドメインを定義します。

優先度

複数のルールが証明書に適用される場合は、最も優先度が高いルールが優先されます。その他のルールはすべて無視されます。

- 数値が低いほど、ID マッピングルールの優先度が高くなります。たとえば、優先度 1 のルールは、優先度 2 のルールよりも高く設定されています。
- ルールに優先度の値が定義されていないと、優先度が最も低くなります。

--priority オプションを使用して、CLI にマッピングルールの優先度を定義します。

証明書マッピングルールの例 1

CLI を使用して、証明書の **Subject** が IdM のユーザーアカウントの **certmapdata** エントリーと一致する場合に、**EXAMPLE.ORG** 組織の **Smart Card CA** によって発行された証明書の認証を許可する **simple_rule** という証明書マッピングルールを定義するには、以下を実行します。

```
# ipa certmaprule-add simple_rule --matchrule '<ISSUER>CN=Smart Card
CA,O=EXAMPLE.ORG' --maprule '(ipacertmapdata=X509:<I>{issuer_dn!nss_x500}<S>
{subject_dn!nss_x500})'
```

68.3. マッチングルールで使用する証明書からのデータの取得

この手順では、証明書からデータを取得して、そのデータをコピーして証明書マッピングルールの一致ルールに貼り付ける方法について説明します。一致ルールに必要なデータを取得するには、**sssctl cert-show** または **sssctl cert-eval-rule** コマンドを使用します。

前提条件

- PEM 形式のユーザー証明書があります。

手順

1. 必要なデータを取得できるように、正しくエンコードされていることを確認する証明書を指す変数を作成します。

```
# CERT=$(openssl x509 -in /path/to/certificate -outform der|base64 -w0)
```

2. **sssctl cert-eval-rule** を使用して、一致するデータを確認します。次の例では、証明書のシリアル番号が使用されています。

```
# sssctl cert-eval-rule $CERT --match='<ISSUER>CN=adcs19-WIN1-
CA,DC=AD,DC=EXAMPLE,DC=COM' --map='LDAPU1:(altSecurityIdentities=X509:<I>
{issuer_dn!ad_x500}<SR>{serial_number!hex_ur})'
Certificate matches rule.
Mapping filter:

    (altSecurityIdentities=X509:<I>DC=com,DC=example,DC=ad,CN=adcs19-WIN1-
CA<SR>0F0000000000DB8852DD7B246C9C0F0000003B)
```

この場合、**altSecurityIdentities=** 以降のすべてをユーザーの AD の **altSecurityIdentities** 属性に追加します。SKI マッピングを使用する場合は、**--map='LDAPU1:(altSecurityIdentities=X509:<SKI>{subject_key_id!hex_u})'** を使用します。

3. オプションで、証明書の発行者が **ad.example.com** ドメインの **adcs19-WIN1-CA** と一致し、証明書のシリアル番号がユーザーアカウントの **altSecurityIdentities** エントリーと一致する必要があることを指定する一致ルールに基づいて、CLI で新しいマッピングルールを作成するには、以下を実行します。

```
# ipa certmaprule-add simple_rule --matchrule '<ISSUER>CN=adcs19-WIN1-
CA,DC=AD,DC=EXAMPLE,DC=COM' --maprule 'LDAPU1:(altSecurityIdentities=X509:<I>
{issuer_dn!ad_x500}<SR>{serial_number!hex_ur})'
```

68.4. IDM に保存されたユーザーの証明書マッピングの設定

ユーザーに設定されている証明書認証が IdM に保存されている場合に IdM で証明書マッピングを有効にするには、システム管理者は次のタスクを完了する必要があります。

- 証明書マッピングルールを設定して、マッピングルールおよび証明書マッピングデータエントリーで指定された条件に一致する証明書を持つ IdM ユーザーが IdM に対して認証できるようにします。
- IdM ユーザーエントリーに証明書マッピングデータを入力すると、証明書マッピングデータエントリーで指定された値がすべて含まれている場合に、ユーザーが複数の証明書を使用して認証できるようになります。

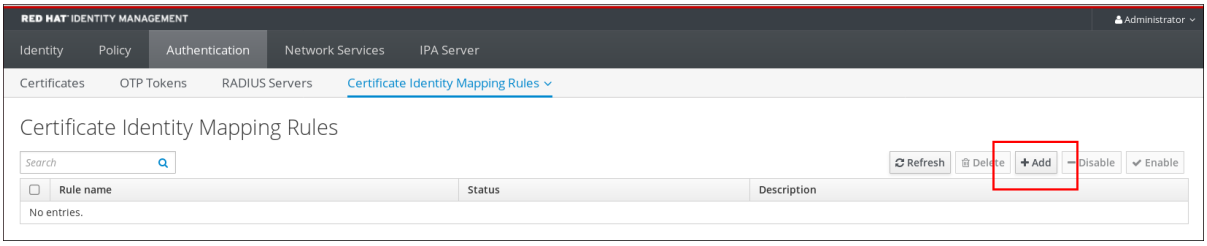
前提条件

- IdM にユーザーがアカウントがある。
- 管理者が、ユーザーエントリーに追加する証明書全体または証明書マッピングデータのいずれかを所有している。

68.4.1. IdM Web UI で証明書マッピングルールの追加

1. 管理者として IdM Web UI にログインします。
2. **Authentication → Certificate Identity Mapping Rules → Certificate Identity Mapping Rules** の順に移動します。
3. **Add** をクリックします。

図68.1 IdM Web UI で新しい証明書マッピングルールの追加



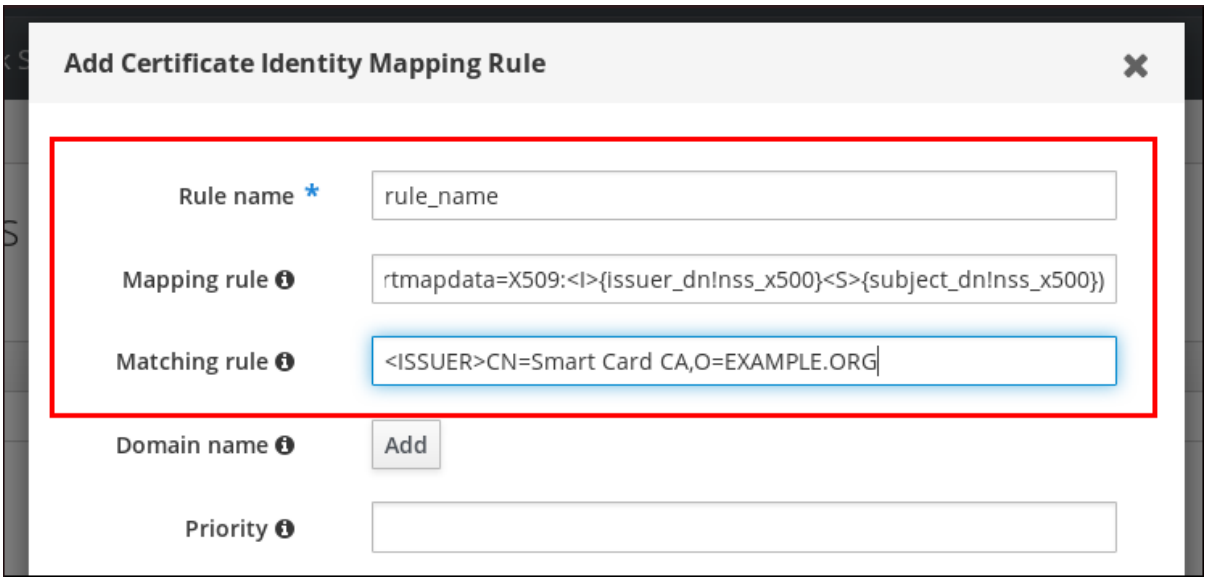
4. ルール名を入力します。
5. マッピングルールを入力します。たとえば、IdM に提示された証明書の **Issuer** および **Subject** エントリーを Idm で検索し、提示された証明書に含まれるこの 2 つのエントリーで見つかった情報に基づいて認証するかどうかを決定するには、次のコマンドを実行します。

```
(ipacertmapdata=X509:<I>{issuer_dn!nss_x500}<S>{subject_dn!nss_x500})
```

6. マッチングルールを入力します。たとえば、**EXAMPLE.ORG** 組織の **スマートカード CA** が発行する証明書のみが IdM に対して認証できるようにするには、次のコマンドを実行します。

```
<ISSUER>CN=Smart Card CA,O=EXAMPLE.ORG
```

図68.2 IdM Web UI への証明書マッピングルールの詳細の入力



7. ダイアログボックスの下部にある **Add** をクリックして、ルールを追加し、ダイアログボックスを閉じます。

8. System Security Services Daemon (SSSD) は、証明書マッピングルールを定期的に再読み込みします。新たに作成したルールがすぐに読み込まれるようにする場合は、次のコマンドを実行して SSSD を再起動します。

```
# systemctl restart sssd
```

これで、証明書マッピングルールセットが設定され、スマートカードの証明書で検出されたマッピングルールで指定されたデータの種類と、IdM ユーザーエントリーの証明書マッピングデータを比較します。一致するファイルが見つかると、一致するユーザーが認証されます。

68.4.2. IdM CLI での証明書マッピングルールの追加

1. 管理者の認証情報を取得します。

```
# kinit admin
```

2. マッピングルールを入力し、マッピングルールの基となっているマッチングルールを入力します。たとえば、提示する証明書内の **Issuer** および **Subject** のエントリーを IdM で検索し、提示された証明書に含まれるこの 2 つのエントリーで見つかった情報に基づいて認証するかどうかを決定し、**EXAMPLE.ORG** 組織の **Smart Card CA** が発行する証明書のみを認識するには、次のコマンドを実行します。

```
# ipa certmaprule-add rule_name --matchrule '<ISSUER>CN=Smart Card CA,O=EXAMPLE.ORG' --maprule '(ipacertmapdata=X509:<I>{issuer_dn!nss_x500}<S>{subject_dn!nss_x500})'
----------------------------------------------------
Added Certificate Identity Mapping Rule "rule_name"
----------------------------------------------------
  Rule name: rule_name
  Mapping rule: (ipacertmapdata=X509:<I>{issuer_dn!nss_x500}<S>{subject_dn!nss_x500})
  Matching rule: <ISSUER>CN=Smart Card CA,O=EXAMPLE.ORG
  Enabled: TRUE
```

3. System Security Services Daemon (SSSD) は、証明書マッピングルールを定期的に再読み込みします。新たに作成したルールがすぐに読み込まれるようにする場合は、次のコマンドを実行して SSSD を再起動します。

```
# systemctl restart sssd
```

これで、証明書マッピングルールセットが設定され、スマートカードの証明書で検出されたマッピングルールで指定されたデータの種類と、IdM ユーザーエントリーの証明書マッピングデータを比較します。一致するファイルが見つかると、一致するユーザーが認証されます。

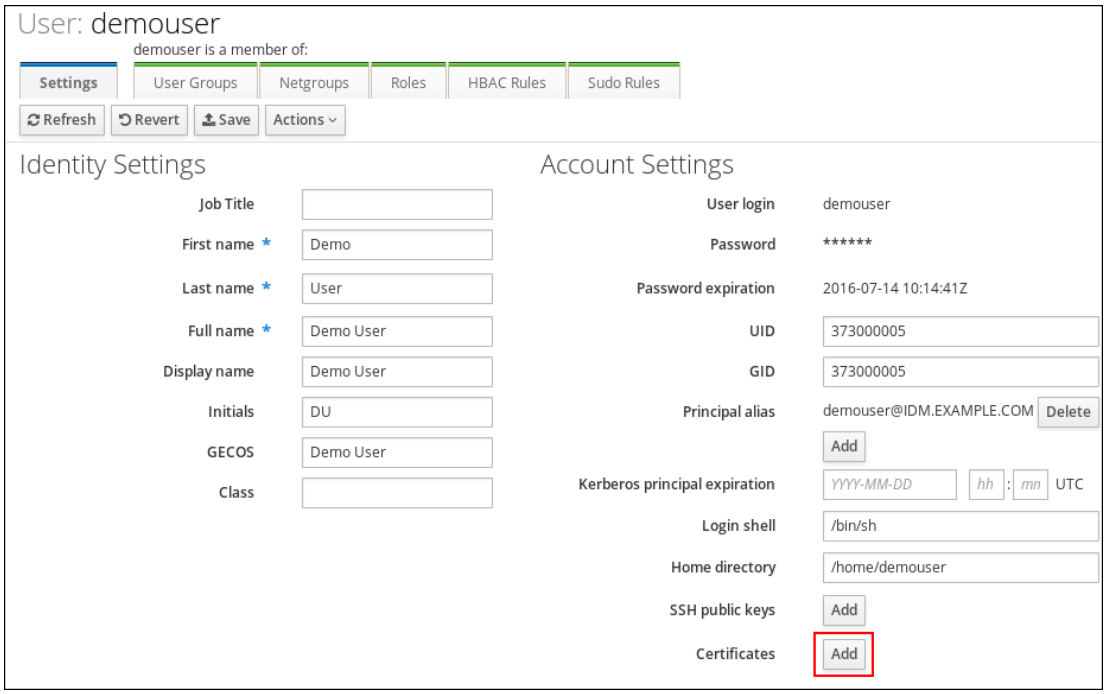
68.4.3. IdM Web UI のユーザーエントリーへの証明書マッピングデータの追加

1. 管理者として IdM Web UI にログインします。
2. **Users → Active users → idm_user** に移動します。
3. **Certificate mapping data** オプションを見つけ、**Add** をクリックします。
4. 以下のいずれかのオプションを選択します。
   - **idm_user** の証明書がある場合:
     a. コマンドラインインターフェイスで、**cat** ユーティリティーまたはテキストエディターを使用して証明書を表示します。

```
[root@server ~]# cat idm_user_certificate.pem
-----BEGIN CERTIFICATE-----
MIIFFTCCA/2gAwIBAgIBEjANBgkqhkiG9w0BAQsFADA6MRgwFgYDVQQKDA9JRE0u
RVhBTVBMRS5DT00xHjAcBgNVBAMMFUNlcnRpZmljYXRlIEF1dGhvcml0eTAeFw0x

ODA5MDIxODE1MzlaFw0yMDA5MDIxODE1MzlaMCwxGDAWBgNVBAoMD0lETS5FWEFN
[...output truncated...]
```

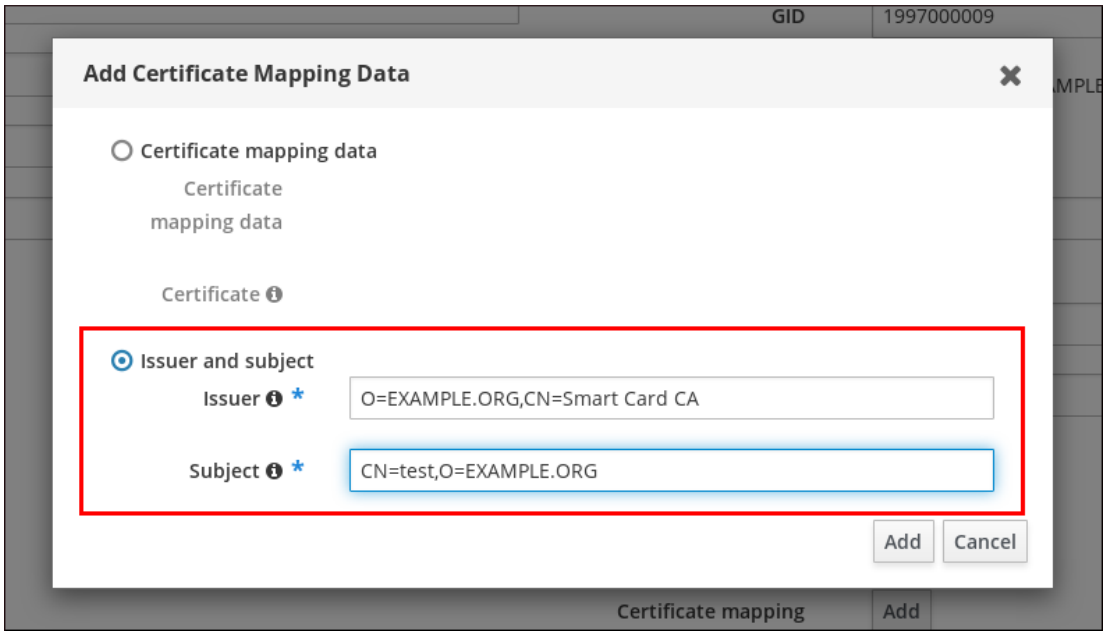
     b. 証明書をコピーします。
     c. IdM Web UI で、**Certificate** の横にある **Add** をクリックして、開いたウィンドウに証明書を貼り付けます。

図68.3 ユーザーの証明書マッピングデータの追加 - 証明書



- **idm_user** の証明書を自由に使用できないけれど、証明書の **Issuer** および **Subject** がわかっている場合は、**Issuer and subject** のラジオボタンをオンにして、2 つのそれぞれのボックスに値を入力します。

図68.4 ユーザーの証明書マッピングデータの追加 - 発行者および発行先



5. **Add** をクリックします。

検証手順

必要に応じて、**.pem** 形式の証明書全体へのアクセスがある場合は、ユーザーと証明書がリンクされていることを確認します。

1. **sss_cache** ユーティリティーを使用して、SSSD キャッシュで **idm_user** の記録を無効にし、**idm_user** 情報を再読み込みします。

```
# sss_cache -u idm_user
```

2. **ipa certmap-match** コマンドに、IdM ユーザーの証明書が含まれるファイルの名前を付けて実行します。

```
# ipa certmap-match idm_user_cert.pem
--------------
1 user matched
--------------
 Domain: IDM.EXAMPLE.COM
 User logins: idm_user
----------------------------
Number of entries returned 1
----------------------------
```

この出力では、証明書マッピングデータが **idm_user** に追加され、対応するマッピングルールが存在することを確認します。これは、定義した証明書マッピングデータに一致する証明書を使用して、**idm_user** として認証できることを意味します。

68.4.4. IdM CLI のユーザーエントリーへの証明書マッピングデータの追加

1. 管理者の認証情報を取得します。

```
# kinit admin
```

2. 以下のいずれかのオプションを選択します。

- **idm_user** の証明書をお持ちの場合は、**ipa user-add-cert** コマンドを使用して証明書をユーザーアカウントに追加します。

```
# CERT=$(openssl x509 -in idm_user_cert.pem -outform der|base64 -w0)
# ipa user-add-certmapdata idm_user --certificate $CERT
```

- **idm_user** の証明書を持っていないが、ユーザーの証明書の **Issuer** および **Subject** がわかっている場合は、以下を実行します。

```
# ipa user-add-certmapdata idm_user --subject "O=EXAMPLE.ORG,CN=test" --
issuer "CN=Smart Card CA,O=EXAMPLE.ORG"
-----------------------------------------
Added certificate mappings to user "idm_user"
-----------------------------------------
  User login: idm_user
  Certificate mapping data: X509:<I>O=EXAMPLE.ORG,CN=Smart Card
CA<S>CN=test,O=EXAMPLE.ORG
```

検証手順

必要に応じて、**.pem** 形式の証明書全体へのアクセスがある場合は、ユーザーと証明書がリンクされていることを確認します。

1. **sss_cache** ユーティリティーを使用して、SSSD キャッシュで **idm_user** の記録を無効にし、**idm_user** 情報を再読み込みします。

```
# sss_cache -u idm_user
```

2. **ipa certmap-match** コマンドに、IdM ユーザーの証明書が含まれるファイルの名前を付けて実行します。

```
# ipa certmap-match idm_user_cert.pem
--------------
1 user matched
--------------
 Domain: IDM.EXAMPLE.COM
 User logins: idm_user
----------------------------
Number of entries returned 1
----------------------------
```

この出力では、証明書マッピングデータが **idm_user** に追加され、対応するマッピングルールが存在することを確認します。これは、定義した証明書マッピングデータに一致する証明書を使用して、**idm_user** として認証できることを意味します。

68.5. ACTIVE DIRECTORY ドメインとの信頼に対する証明書マッピングルール

IdM デプロイメントが Active Directory (AD) ドメインと信頼関係にある場合、さまざまな証明書マッピングの使用例が可能です。

AD 設定によっては、以下の状況が考えられます。

- 証明書が AD Certificate System によって発行され、ユーザーと証明書が IdM に保存されている場合、認証リクエストのマッピングと処理全体が IdM 側で行われます。このシナリオの設定に関する詳細は IdM に保存されたユーザーの証明書マッピングの設定 を参照してください。
- ユーザーが AD に保存されている場合は、認証要求の処理が AD で実行されます。サブケースは 3 つあります。
  - AD ユーザーエントリーに、証明書全体が含まれる場合。このシナリオで IdM を設定する方法は、AD ユーザーエントリーに証明書全体が含まれるユーザー用の証明書マッピングの設定 を参照してください。
  - AD が、ユーザー証明書をユーザーアカウントにマップするように設定されている場合。この場合、AD ユーザーエントリーには証明書全体が含まれず、代わりに **altSecurityIdentities** と呼ばれる属性が含まれます。このシナリオで IdM を設定する方法は、AD がユーザー証明書をユーザーアカウントにマッピングするように設定している場合は、証明書マッピングの設定 を参照してください。
  - AD ユーザーエントリーに、証明書全体またはマッピングデータが含まれない場合。この場合、次の 2 つのオプションがあります。
    - ユーザー証明書が AD Certificate System によって発行された場合、証明書にはサブジェクト代替名 (SAN) としてユーザープリンシパル名が含まれるか、証明書の SID 拡張子のユーザーの SID (最新の更新が AD に適用されている場合) が含まれます。これらは両方とも、証明書をユーザーにマッピングするために使用できます。
    - ユーザー証明書がスマートカード上にある場合、スマートカードで SSH を有効にするには、SSSD は証明書から公開 SSH キーを取得する必要があるため、完全な証明書が必要です。唯一の解決策は、**ipa idoverrideuser-add** コマンドを使用して、証明書全体を IdM の AD ユーザーの ID オーバーライドに追加することです。詳細は AD ユーザーエントリーに証明書やマッピングデータが含まれていない場合の証明書マッピングの設定 を参照してください。

AD ドメイン管理者は、**altSecurityIdentities** 属性を使用して証明書を AD 内のユーザーに手動でマッピングできます。この属性には 6 つの値がサポートされていますが、3 つのマッピングは安全ではないと考えられています。2022 年 5 月 10 日のセキュリティー更新 の一環として、インストールされると、すべてのデバイスは互換モードになり、証明書がユーザーに弱くマッピングされている場合、認証は期待どおりに行われます。ただし、完全強制モードと互換性のない証明書を特定する警告メッセージがログに記録されます。2023 年 11 月 14 日以降、すべてのデバイスは完全強制モードに更新され、証明書が強力なマッピング基準を満たさない場合、認証は拒否されます。

たとえば、AD ユーザーが証明書 (PKINIT) を含む IdM Kerberos チケットをリクエストすると、AD は証明書を内部でユーザーにマップする必要があり、これに新しいマッピングルールを使用します。ただし、IdM では、IdM クライアント上のユーザーに証明書をマップするために IdM が使用されている場合、以前のルールが引き続き機能します。

IdM は新しいマッピングテンプレートをサポートしているため、AD 管理者は新しいルールを使用し、両方を維持する必要がなくなります。IdM は、Active Directory に追加された以下を含む新しいマッピングテンプレートをサポートするようになりました。

- シリアル番号: LDAPU1:(altSecurityIdentities=X509:<I>{issuer_dn!ad_x500}<SR>{serial_number!hex_ur})
- サブジェクトキー ID: LDAPU1:(altSecurityIdentities=X509:<SKI>{subject_key_id!hex_u})
- User SID: LDAPU1:(objectsid={sid})

新しい SID 拡張子を使用して証明書を再発行したくない場合は、AD のユーザーの **altSecurityIdentities** 属性に適切なマッピング文字列を追加して、手動マッピングを作成できます。

68.6. AD ユーザーエントリーに証明書全体が含まれるユーザーに証明書マッピングを設定

このユーザーストーリーでは、IdM デプロイメントが Active Directory (AD) を信頼し、そのユーザーが AD に保存され、AD のユーザーエントリーに証明書全体が含まれる場合に、IdM で証明書マッピングを有効にするのに必要な手順を説明します。

前提条件

- IdM にユーザーアカウントがない。
- ユーザーに、証明書を含む AD のアカウントがある。
- IdM 管理者が、IdM 証明書マッピングルールが基になっているデータにアクセスできる。

注記

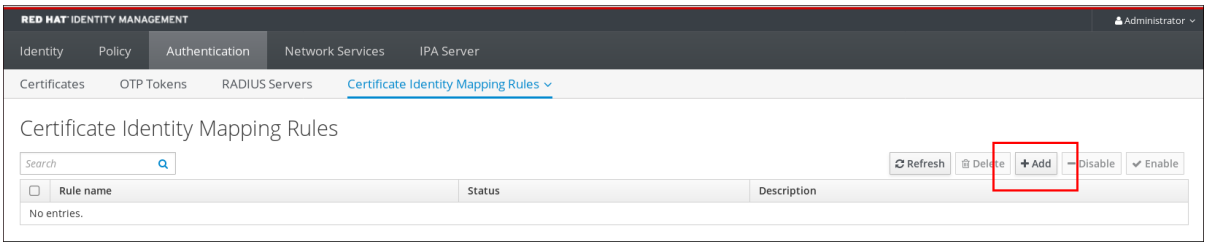
PKINIT がユーザーに対して確実に機能するには、次の条件のいずれかが適用される必要があります。

- ユーザーエントリーの証明書には、ユーザープリンシパル名またはユーザーの SID 拡張子が含まれます。
- AD のユーザーエントリーには、**altSecurityIdentities** 属性に適切なエントリーがあります。

68.6.1. IdM Web UI で証明書マッピングルールの追加

1. 管理者として IdM Web UI にログインします。
2. **Authentication → Certificate Identity Mapping Rules → Certificate Identity Mapping Rules** の順に移動します。
3. **Add** をクリックします。

図68.5 IdM Web UI で新しい証明書マッピングルールの追加



4. ルール名を入力します。
5. マッピングルールを入力します。認証のために IdM に提示された証明書全体を、AD で利用可能な証明書全体と比較するには、次のコマンドを実行します。

```
(userCertificate;binary={cert!bin})
```

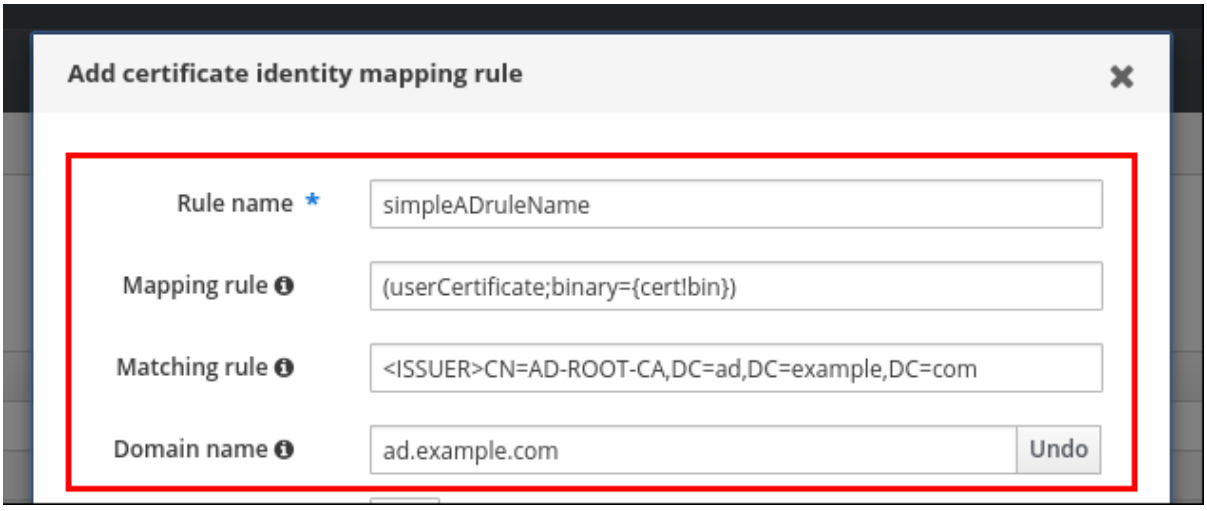
注記

完全な証明書を使用してマッピングする場合、また、証明書を更新する場合は、新しい証明書を AD ユーザーオブジェクトに必ず追加する必要があります。

6. マッチングルールを入力します。たとえば、**AD.EXAMPLE.COM** ドメインの **AD-ROOT-CA** が発行する証明書のみを認証できるようにするには、次のコマンドを実行します。

```
<ISSUER>CN=AD-ROOT-CA,DC=ad,DC=example,DC=com
```

図68.6 AD に保存されている証明書があるユーザーの証明書マッピングルール



7. **Add** をクリックします。
8. System Security Services Daemon (SSSD) は、証明書マッピングルールを定期的に再読み込みします。新たに作成したルールがすぐに読み込まれるようにするには、CLI で SSSD を再起動します。

```
# systemctl restart sssd
```

68.6.2. IdM CLI での証明書マッピングルールの追加

1. 管理者の認証情報を取得します。

```
# kinit admin
```

2. マッピングルールを入力し、マッピングルールの基となっているマッチングルールを入力します。AD で利用可能な証明書と比較する、認証用に提示される証明書全体を取得して、**AD.EXAMPLE.COM** ドメインの **AD-ROOT-CA** により発行された証明書のみの認証を許可するには、次のコマンドを実行します。

```
# ipa certmaprule-add simpleADrule --matchrule '<ISSUER>CN=AD-ROOT-CA,DC=ad,DC=example,DC=com' --maprule '(userCertificate;binary={cert!bin})' --domain ad.example.com
----------------------------------------------------
Added Certificate Identity Mapping Rule "simpleADrule"
----------------------------------------------------
  Rule name: simpleADrule
  Mapping rule: (userCertificate;binary={cert!bin})
  Matching rule: <ISSUER>CN=AD-ROOT-CA,DC=ad,DC=example,DC=com
  Domain name: ad.example.com
  Enabled: TRUE
```

注記

完全な証明書を使用してマッピングする場合、また、証明書を更新する場合は、新しい証明書を AD ユーザーオブジェクトに必ず追加する必要があります。

3. System Security Services Daemon (SSSD) は、証明書マッピングルールを定期的に再読み込みします。新たに作成したルールがすぐに読み込まれるようにする場合は、次のコマンドを実行して SSSD を再起動します。

```
# systemctl restart sssd
```

68.7. ユーザー証明書をユーザーアカウントにマッピングするように AD が設定されている場合に、証明書マッピングの設定

このユーザーストーリーでは、IdM デプロイメントが Active Directory (AD) を信頼し、そのユーザーが AD に保存され、AD のユーザーエントリーに証明書マッピングデータが含まれる場合に、IdM で証明書マッピングを有効にするのに必要な手順を説明します。

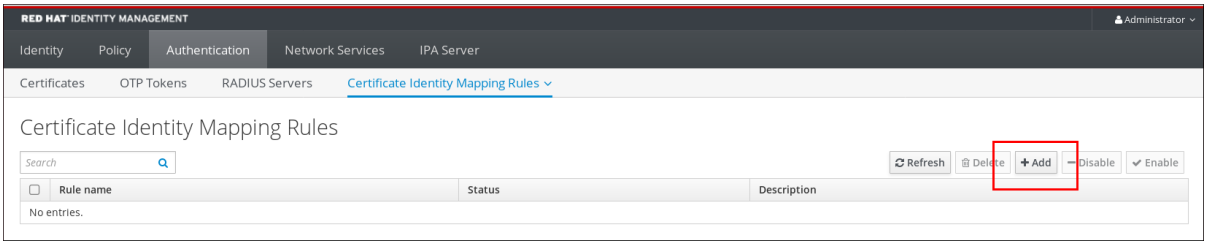
前提条件

- IdM にユーザーアカウントがない。
- このユーザーに、**altSecurityIdentities** 属性を含む AD にアカウントがある。AD は、IdM の **certmapdata** 属性に相当します。
- IdM 管理者が、IdM 証明書マッピングルールが基になっているデータにアクセスできる。

68.7.1. IdM Web UI で証明書マッピングルールの追加

1. 管理者として IdM Web UI にログインします。
2. **Authentication → Certificate Identity Mapping Rules → Certificate Identity Mapping Rules** の順に移動します。
3. **Add** をクリックします。

図68.7 IdM Web UI で新しい証明書マッピングルールの追加



4. ルール名を入力します。
5. マッピングルールを入力します。たとえば、提示された証明書で **Issuer** エントリーおよび **Subject** エントリーを AD DC で検索し、提示された証明書に含まれるこの 2 つのエントリーで見つかった情報に基づいて認証するかどうかを決定するには、次のコマンドを実行します。

```
(altSecurityIdentities=X509:<I>{issuer_dn!ad_x500}<S>{subject_dn!ad_x500})
```

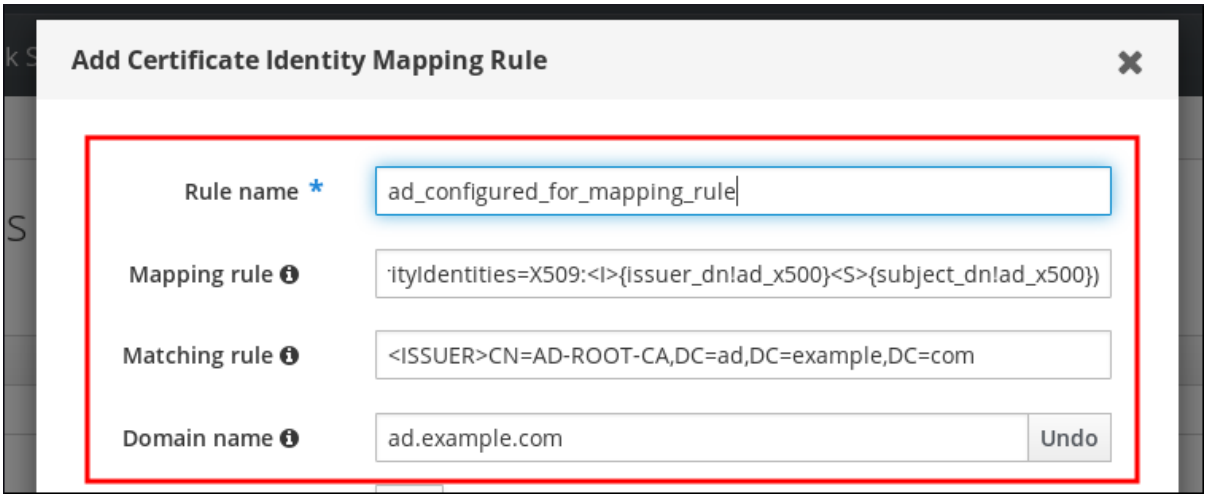
6. マッチングルールを入力します。たとえば、**AD.EXAMPLE.COM** ドメインの **AD-ROOT-CA** が発行する証明書のみを許可し、IdM に対してユーザーを認証するには、次のコマンドを実行します。

```
<ISSUER>CN=AD-ROOT-CA,DC=ad,DC=example,DC=com
```

7. ドメインを入力します。

```
ad.example.com
```

図68.8 AD がマッピング用に設定されている場合の証明書マッピングルール



8. **Add** をクリックします。

9. System Security Services Daemon (SSSD) は、証明書マッピングルールを定期的に再読み込みします。新たに作成したルールがすぐに読み込まれるようにするには、CLI で SSSD を再起動します。

```
# systemctl restart sssd
```

68.7.2. IdM CLI での証明書マッピングルールの追加

1. 管理者の認証情報を取得します。

```
# kinit admin
```

2. マッピングルールを入力し、マッピングルールの基となっているマッチングルールを入力します。たとえば、提示する証明書の **Issuer** エントリーおよび **Subject** エントリーを AD で検索し、**AD.EXAMPLE.COM** ドメインの **AD-ROOT-CA** により発行された証明書のみを許可するには、次のコマンドを実行します。

```
# ipa certmaprule-add ad_configured_for_mapping_rule --matchrule
'<ISSUER>CN=AD-ROOT-CA,DC=ad,DC=example,DC=com' --maprule
'(altSecurityIdentities=X509:<I>{issuer_dn!ad_x500}<S>{subject_dn!ad_x500})' --
domain=ad.example.com
----------------------------------------------------
Added Certificate Identity Mapping Rule "ad_configured_for_mapping_rule"
----------------------------------------------------
  Rule name: ad_configured_for_mapping_rule
  Mapping rule: (altSecurityIdentities=X509:<I>{issuer_dn!ad_x500}<S>
{subject_dn!ad_x500})
  Matching rule: <ISSUER>CN=AD-ROOT-CA,DC=ad,DC=example,DC=com
  Domain name: ad.example.com
  Enabled: TRUE
```

3. System Security Services Daemon (SSSD) は、証明書マッピングルールを定期的に再読み込みします。新たに作成したルールがすぐに読み込まれるようにする場合は、次のコマンドを実行して SSSD を再起動します。

```
# systemctl restart sssd
```

68.7.3. AD で証明書マッピングデータの確認

altSecurityIdentities 属性は、IdM の **certmapdata** ユーザー属性と同等の Active Directory (AD) です。信頼されている AD ドメインが、ユーザーアカウントにユーザー証明書をマッピングするように設定されている時に IdM で証明書マッピングを設定する場合は、IdM システム管理者が、AD のユーザーエントリーに **altSecurityIdentities** 属性が正しく設定されていることを確認する必要があります。

前提条件

- ユーザーアカウントにはユーザー管理アクセス権が必要です。

手順

- AD に保存されているユーザーの適切な情報が AD に含まれていることを確認するには、**ldapsearch** コマンドを使用します。たとえば、次のコマンドを入力して、以下の条件が適用される **adserver.ad.example.com** サーバーで、チェックします。

- **altSecurityIdentities** 属性は、**ad_user** のユーザーエントリーに設定されます。
- matchrule では、以下の条件が適用されるように指定します。
  - **ad_user** が AD への認証に使用する証明書が **ad.example.com** ドメインの **AD-ROOT-CA** により発行されている。
  - 発行者が **<S>DC=com,DC=example,DC=ad,CN=Users,CN=ad_user** である。

```
$ ldapsearch -o ldif-wrap=no -LLL -h adserver.ad.example.com \
-p 389 -D cn=Administrator,cn=users,dc=ad,dc=example,dc=com \
-W -b cn=users,dc=ad,dc=example,dc=com "(cn=ad_user)" \
altSecurityIdentities
Enter LDAP Password:
dn: CN=ad_user,CN=Users,DC=ad,DC=example,DC=com
altSecurityIdentities: X509:<I>DC=com,DC=example,DC=ad,CN=AD-ROOT-
CA<S>DC=com,DC=example,DC=ad,CN=Users,CN=ad_user
```

68.8. AD ユーザーエントリーに証明書やマッピングデータが含まれていない場合に、証明書マッピングの設定

このユーザーストーリーでは、IdM デプロイメントが Active Directory (AD) を信頼し、そのユーザーが AD に保存され、AD のユーザーエントリーに証明書全体または証明書マッピングデータが含まれる場合に、IdM で証明書マッピングを有効にするのに必要な手順を説明します。

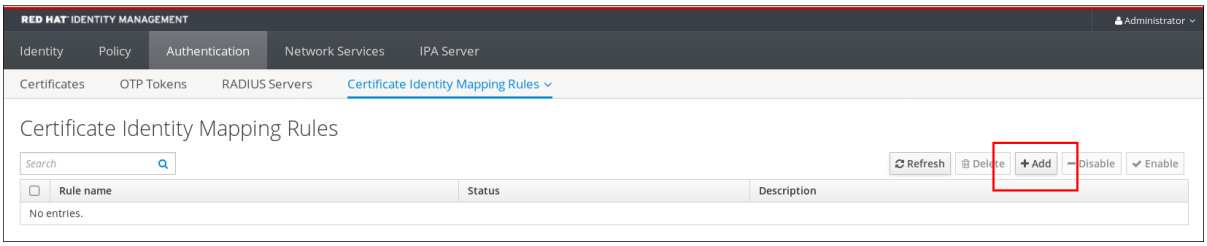
前提条件

- IdM にユーザーアカウントがない。
- ユーザーのアカウントがある AD に、証明書全体、または **altSecurityIdentities** 属性、IdM **certmapdata** 属性で AD に相当するものがない。
- IdM 管理者は次のいずれかを実行しました。
  - AD ユーザー証明書全体を IdM の AD ユーザーの **user ID override** に追加しました。
  - サブジェクト代替名やユーザーの SID など、証明書内の代替フィールドにマップする証明書マッピングルールを作成しました。

68.8.1. IdM Web UI で証明書マッピングルールの追加

1. 管理者として IdM Web UI にログインします。
2. **Authentication → Certificate Identity Mapping Rules → Certificate Identity Mapping Rules** の順に移動します。
3. **Add** をクリックします。

図68.9 IdM Web UI で新しい証明書マッピングルールの追加



4. ルール名を入力します。

5. マッピングルールを入力します。認証するために IdM に提示された証明書全体を、IdM の AD ユーザーエントリーのユーザー ID オーバーライドエントリーに保存されている証明書と比較できるようにするには、次のコマンドを実行します。

```
(userCertificate;binary={cert!bin})
```

注記

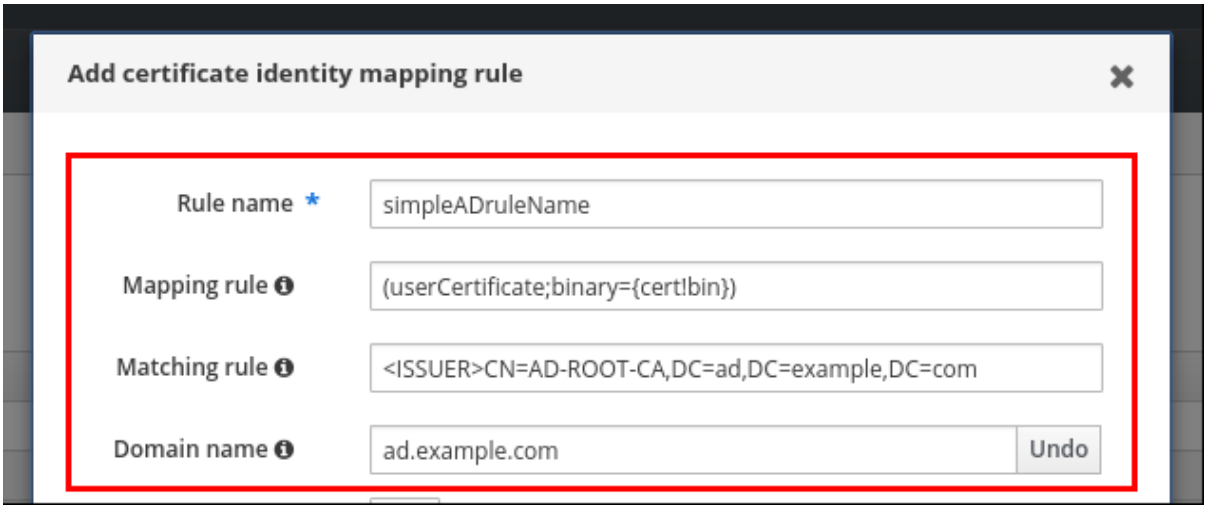
証明書には SAN としてのユーザープリンシパル名も含まれており、最新の更新では証明書の SID 拡張子にユーザーの SID も含まれているため、これらのフィールドを使用して証明書をユーザーにマップすることもできます。たとえば、ユーザーの SID を使用する場合は、このマッピングルールを **LDAPU1: (objectsid={sid})** に置き換えます。証明書マッピングの詳細は、**sss-certmap** の man ページを参照してください。

6. マッチングルールを入力します。たとえば、**AD.EXAMPLE.COM** ドメインの **AD-ROOT-CA** が発行する証明書のみを認証できるようにするには、次のコマンドを実行します。

```
<ISSUER>CN=AD-ROOT-CA,DC=ad,DC=example,DC=com
```

7. ドメイン名を入力します。たとえば、**ad.example.com** ドメインでユーザーを検索するには、以下を実行します。

図68.10 AD に証明書やマッピングデータが保存されていないユーザーに対する証明書マッピングルール



8. **Add** をクリックします。

9. System Security Services Daemon (SSSD) は、証明書マッピングルールを定期的に再読み込みします。新たに作成したルールがすぐに読み込まれるようにするには、CLI で SSSD を再起動します。

```
# systemctl restart sssd
```

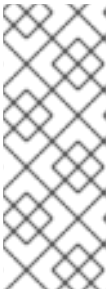
68.8.2. IdM CLI での証明書マッピングルールの追加

1. 管理者の認証情報を取得します。

```
# kinit admin
```

2. マッピングルールを入力し、マッピングルールの基となっているマッチングルールを入力します。IdM の AD ユーザーエントリーのユーザー ID オーバーライドエントリーに保存されている証明書と比較する、認証用に提示される証明書全体を取得して、**AD.EXAMPLE.COM** ドメインの **AD-ROOT-CA** により発行された証明書のみを認証できるようにするには、以下のコマンドを実行します。

```
# ipa certmaprule-add simpleADrule --matchrule '<ISSUER>CN=AD-ROOT-CA,DC=ad,DC=example,DC=com' --maprule '(userCertificate;binary={cert!bin})' --domain ad.example.com
----------------------------------------------------
Added Certificate Identity Mapping Rule "simpleADrule"
----------------------------------------------------
  Rule name: simpleADrule
  Mapping rule: (userCertificate;binary={cert!bin})
  Matching rule: <ISSUER>CN=AD-ROOT-CA,DC=ad,DC=example,DC=com
  Domain name: ad.example.com
  Enabled: TRUE
```



注記

証明書には SAN としてのユーザープリンシパル名も含まれており、最新の更新では証明書の SID 拡張子にユーザーの SID も含まれているため、これらのフィールドを使用して証明書をユーザーにマップすることもできます。たとえば、ユーザーの SID を使用する場合は、このマッピングルールを **LDAPU1:(objectsid={sid})** に置き換えます。証明書マッピングの詳細は、**sss-certmap** の man ページを参照してください。

3. System Security Services Daemon (SSSD) は、証明書マッピングルールを定期的に再読み込みします。新たに作成したルールがすぐに読み込まれるようにする場合は、次のコマンドを実行して SSSD を再起動します。

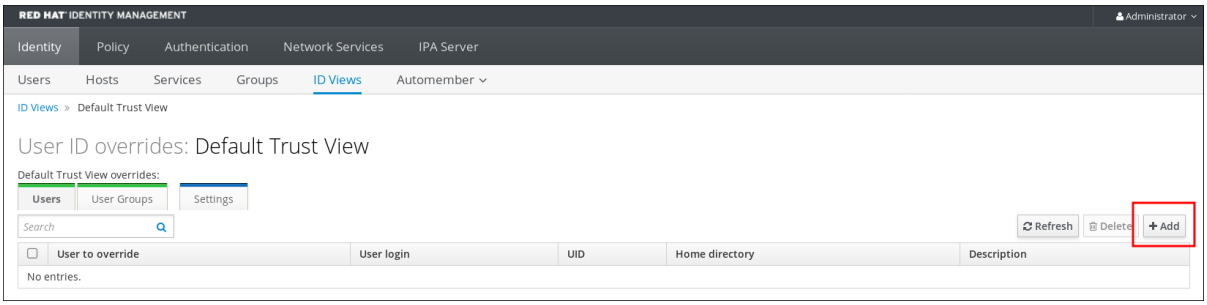
```
# systemctl restart sssd
```

68.8.3. IdM Web UI で、AD ユーザーの ID オーバーライドに証明書を追加

1. **Identity → ID Views → Default Trust View** の順に選択します。

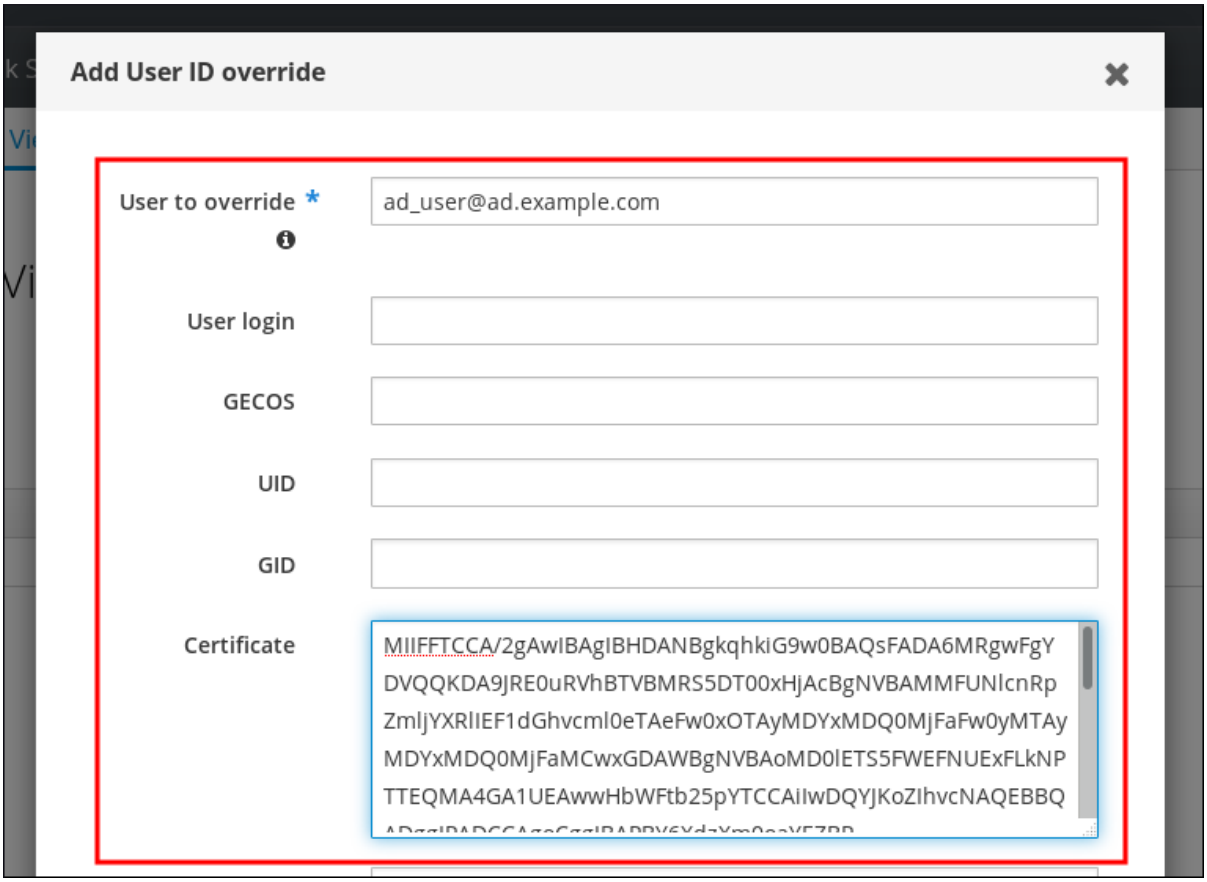
2. **Add** をクリックします。

図68.11 IdM Web UI で新規ユーザー ID オーバーライドの追加



3. **User to override** フィールドに、**ad_user@ad.example.com** と入力します。
4. **ad_user** の証明書を、**Certificate** フィールドにコピーアンドペーストします。

図68.12 AD ユーザーにユーザー ID オーバーライドの設定



5. **Add** をクリックします。

検証手順

ユーザーと証明書がリンクしていることを確認します。

1. **sss_cache** ユーティリティーを使用して、SSSD キャッシュで **ad_user@ad.example.com** のレコードを無効にし、**ad_user@ad.example.com** 情報の再読み込みを強制します。

```
# sss_cache -u ad_user@ad.example.com
```

2. AD ユーザーの証明書が含まれるファイルの名前で、**ipa certmap-match** コマンドを実行します。

```
# ipa certmap-match ad_user_cert.pem
--------------
1 user matched
--------------
 Domain: AD.EXAMPLE.COM
 User logins: ad_user@ad.example.com
----------------------------
Number of entries returned 1
----------------------------
```

この出力では、**ad_user@ad.example.com** に追加した証明書マッピングデータがあり、Adding a certificate mapping rule if the AD user entry contains no certificate or mapping data で定義した対応するマッピングルールが存在することを確認します。これは、定義した証明書マッピングデータに一致する証明書を使用して、**ad_user@ad.example.com** として認証できることを意味します。

関連情報

- Active Directory ユーザーの ID ビューの使用

68.8.4. IdM CLI で、AD ユーザーの ID オーバーライドに証明書を追加する

1. 管理者の認証情報を取得します。

```
# kinit admin
```

2. 証明書 blob を **CERT** という新しい変数に格納します。

```
# CERT=$(openssl x509 -in /path/to/certificate -outform der|base64 -w0)
```

3. **ipa idoverrideuser-add-cert** コマンドを使用して、**ad_user@ad.example.com** の証明書をユーザーアカウントに追加します。

```
# ipa idoverrideuser-add-cert ad_user@ad.example.com --certificate $CERT
```

検証手順

ユーザーと証明書がリンクしていることを確認します。

1. **sss_cache** ユーティリティーを使用して、SSSD キャッシュで **ad_user@ad.example.com** のレコードを無効にし、**ad_user@ad.example.com** 情報の再読み込みを強制します。

```
# sss_cache -u ad_user@ad.example.com
```

2. AD ユーザーの証明書が含まれるファイルの名前で、**ipa certmap-match** コマンドを実行します。

```
# ipa certmap-match ad_user_cert.pem
--------------
1 user matched
--------------
 Domain: AD.EXAMPLE.COM
 User logins: ad_user@ad.example.com
```

```
----------------------------
Number of entries returned 1
----------------------------
```

この出力では、**ad_user@ad.example.com** に追加した証明書マッピングデータがあり、Adding a certificate mapping rule if the AD user entry contains no certificate or mapping data で定義した対応するマッピングルールが存在することを確認します。これは、定義した証明書マッピングデータに一致する証明書を使用して、**ad_user@ad.example.com** として認証できることを意味します。

関連情報

- Active Directory ユーザーの ID ビューの使用

68.9. 複数のアイデンティティーマッピングルールを 1 つに結合

複数の ID マッピングルールを 1 つのルールに結合するには、個々のマッピングルールの前に | (or) 文字を追加し、括弧 **()** で区切ります。以下に例を示します。

証明書マッピングフィルターの例 1

```
$ ipa certmaprule-add ad_cert_for_ipa_and_ad_users \ --maprule='(|(ipacertmapdata=X509:<I>
{issuer_dn!nss_x500}<S>{subject_dn!nss_x500})(altSecurityIdentities=X509:<I>
{issuer_dn!ad_x500}<S>{subject_dn!ad_x500}))' \ --matchrule='<ISSUER>CN=AD-ROOT-
CA,DC=ad,DC=example,DC=com' \ --domain=ad.example.com
```

上記の例では、**--maprule** オプションのフィルター定義には、以下の基準が含まれます。

- **ipacertmapdata=X509:<I>{issuer_dn!nss_x500}<S>{subject_dn!nss_x500}** は、IdM での証明書マッピングルールの追加 の説明のとおりに、IdM ユーザーアカウントの **ipacertmapdata** 属性の値に、スマートカードの発行先および発行者をリンクさせるフィルターです。
- **altSecurityIdentities=X509:<I>{issuer_dn!ad_x500}<S>{subject_dn!ad_x500}** は、スマートカード証明書から発行先および発行者を、AD ユーザーアカウントの **altSecurityIdentities** の値にリンクするフィルターです。これは、信頼された AD ドメインがユーザー証明書をマッピングするように設定されている場合には、証明書マッピングルールの追加 で説明されています。
- **--domain=ad.example.com** オプションを追加すると、指定した証明書にマッピングされたユーザーが、ローカルの **idm.example.com** ドメインだけでなく、ad.example.com**ad.example.com** ドメイン内でも検索されます。

--maprule オプションのフィルターの定義では、論理演算子 | (or) が使用できるため、複数の基準を指定できます。この場合、ルールは、1 つ以上の基準を満たすユーザーアカウントをすべてマップします。

証明書マッピングフィルターの例 2

```
$ ipa certmaprule-add ipa_cert_for_ad_users \
  --maprule='(|(userCertificate;binary={cert!bin})(ipacertmapdata=X509:<I>
{issuer_dn!nss_x500}<S>{subject_dn!nss_x500})(altSecurityIdentities=X509:<I>
{issuer_dn!ad_x500}<S>{subject_dn!ad_x500}))' \
  --matchrule='<ISSUER>CN=Certificate Authority,O=REALM.EXAMPLE.COM' \
  --domain=idm.example.com --domain=ad.example.com
```

上記の例では、**--maprule** オプションのフィルター定義には、以下の基準が含まれます。

- **userCertificate;binary={cert!bin}** は、証明書全体を含むユーザーエントリーを返すフィルターです。AD ユーザーが、この種のフィルターを作成する場合は、AD ユーザーエントリーに証明書またはマッピングデータがない場合の証明書マッピングルールの追加 を参照してください。

- **ipacertmapdata=X509:<I>{issuer_dn!nss_x500}<S>{subject_dn!nss_x500}** は、IdM での証明書マッピングルールの追加 の説明のとおりに、IdM ユーザーアカウントの **ipacertmapdata** 属性の値に、スマートカードの発行先および発行者をリンクさせるフィルターです。

- **altSecurityIdentities=X509:<I>{issuer_dn!ad_x500}<S>{subject_dn!ad_x500}** は、スマートカード証明書から発行先および発行者を、AD ユーザーアカウントの **altSecurityIdentities** の値にリンクするフィルターです。これは、信頼された AD ドメインがユーザー証明書をマッピングするように設定されている場合には、証明書マッピングルールの追加 で説明されています。

--maprule オプションのフィルターの定義では、論理演算子 | (or) が使用できるため、複数の基準を指定できます。この場合、ルールは、1 つ以上の基準を満たすユーザーアカウントをすべてマップします。

68.10. 関連情報

- **sss-certmap(5)** の man ページを参照してください。

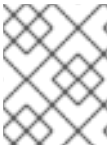
第69章 IDM クライアントのデスクトップに保存されている証明書を使用した認証の設定

Identity Management (IdM) を設定すると、IdM システム管理者は、認証局 (CA) がユーザーに発行した証明書を使用して、IdM Web UI とコマンドラインインターフェイス (CLI) に対してユーザーを認証できます。証明書は IdM クライアントのデスクトップに保存されます。

Web ブラウザーは、IdM ドメイン外のシステムで実行できます。

証明書を使用した認証設定中に、以下の点に注意してください。

- 証明書を使用して認証するユーザーに証明書がすでにある場合は、新しいユーザー証明書を要求し、クライアントにエクスポート を省略できます。
- ユーザーの証明書が IdM CA により発行された場合は、証明書とユーザーが互いにリンクしていることの確認 を省略できます。



注記

Identity Management ユーザーのみが、証明書を使用して Web UI にログインできます。Active Directory ユーザーは、ユーザー名とパスワードを使用してログインできます。

69.1. WEB UI での証明書認証用の IDENTITY MANAGEMENT SERVER の設定

Identity Management (IdM) 管理者は、ユーザーが、証明書を使用して IdM 環境で認証できるように設定できます。

手順

Identity Management 管理者が、以下を行います。

1. Identity Management サーバーで管理者権限を取得し、サーバーを設定するシェルスクリプトを作成します。
   a. **ipa-advise config-server-for-smart-card-auth** コマンドを実行し、その出力をファイル (例: **server_certificate_script.sh**) に保存します。

```
# kinit admin
# ipa-advise config-server-for-smart-card-auth > server_certificate_script.sh
```

   b. **chmod** ユーティリティーを使用して、実行パーミッションをファイルに追加します。

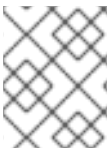
```
# chmod +x server_certificate_script.sh
```

2. Identity Management ドメインの全サーバーで、**server_certificate_script.sh** スクリプトを実行します。
   a. 証明書認証を有効にするユーザーの証明書を発行した唯一の認証局が IdM CA である場合は、IdM Certificate Authority 証明書へのパス (**/etc/ipa/ca.crt**) を使用します。

```
# ./server_certificate_script.sh /etc/ipa/ca.crt
```

b. 証明書認証を有効にするユーザーの証明書を外部の複数の CA が署名した場合は、関連する CA 証明書へのパスを使用します。

```
# ./server_certificate_script.sh /tmp/ca1.pem /tmp/ca2.pem
```



注記

トポロジー全体でユーザーの証明書認証を有効にする場合は、今後新たにシステムに追加する各レプリカに対して、必ずスクリプトを実行してください。

69.2. 新しいユーザー証明書を要求し、クライアントにエクスポート

Identity Management (IdM) 管理者は、IdM 環境でユーザーの証明書を作成し、作成した証明書を、ユーザーの証明書認証を有効にする IdM クライアントにエクスポートできます。

注記

証明書を使用して認証するユーザーがすでに証明書を持っている場合は、この手順を実行する必要はありません。

手順

1. 必要に応じて、新しいディレクトリー (例: ~/**certdb**/) を作成し、証明書の一時データベースを作成します。要求されたら、NSS 証明書の DB パスワードを作成し、後続の手順で生成される証明書への鍵を暗号化します。

```
# mkdir ~/certdb/
# certutil -N -d ~/certdb/
Enter a password which will be used to encrypt your keys.
The password should be at least 8 characters long,
and should contain at least one non-alphabetic character.

Enter new password:
Re-enter password:
```

2. 証明書署名要求 (CSR) を作成し、その出力をファイルにリダイレクトします。たとえば、**IDM.EXAMPLE.COM** レルムの **idm_user** ユーザーの **4096** ビット証明書に対して、**certificate_request.csr** という名前の CSR を作成する場合は、判別を簡単にするために、証明書の秘密鍵のニックネームを **idm_user** に設定し、発行先を **CN=idm_user,O=IDM.EXAMPLE.COM** に設定します。

```
# certutil -R -d ~/certdb/ -a -g 4096 -n idm_user -s "CN=idm_user,O=IDM.EXAMPLE.COM"
> certificate_request.csr
```

3. プロンプトが表示されたら、**certutil** を使用して一時データベースを作成したときに入力したパスワードを入力します。その後、止めるように言われるまで、ランダムにタイピングし続けます。

```
Enter Password or Pin for "NSS Certificate DB":

A random seed must be generated that will be used in the
creation of your key.  One of the easiest ways to create a
random seed is to use the timing of keystrokes on a keyboard.
```

```
To begin, type keys on the keyboard until this progress meter
is full.  DO NOT USE THE AUTOREPEAT FUNCTION ON YOUR KEYBOARD!


Continue typing until the progress meter is full:
```

4. 証明書要求ファイルをサーバーに送信します。新しく発行した証明書に関連付ける Kerberos プリンシパルと、証明書を保存する出力ファイルを指定し、必要に応じて証明書のプロファイルを指定します。たとえば、**IECUserRoles** プロファイル (**idm_user@IDM.EXAMPLE.COM** プリンシパルに追加したユーザーロール拡張を持つプロファイル) の証明書を取得して、それを **~/idm_user.pem** ファイルに保存する場合は、次のコマンドを実行します。

```
# ipa cert-request certificate_request.csr --principal=idm_user@IDM.EXAMPLE.COM --
profile-id=IECUserRoles --certificate-out=~/idm_user.pem
```

5. 証明書を NSS データベースに追加します。証明書が NSS データベースの秘密鍵に一致するように、CSR を作成する際に使用したニックネームを設定するには、**-n** オプションを使用します。**-t** オプションは信頼レベルを設定します。詳細は、certutil(1) man ページを参照してください。**-i** オプションは、入力証明書ファイルを指定します。たとえば、**idm_user** ニックネームを持つ証明書を NSS データベースに追加するには、次のコマンドを実行します。証明書は、**~/certdb/** データベースの **~/idm_user.pem** ファイルに保存されます。

```
# certutil -A -d ~/certdb/ -n idm_user -t "P,," -i ~/idm_user.pem
```

6. NSS データベースの鍵で、ニックネームが **(orphan)** と表示されていないことを確認します。たとえば、**~/certdb/** データベースに保存されている証明書で、対応する鍵が存在することを確認するには、以下のコマンドを実行します。

```
# certutil -K -d ~/certdb/
< 0> rsa      5ad14d41463b87a095b1896cf0068ccc467df395   NSS Certificate
DB:idm_user
```

7. 証明書を、NSS データベースから PKCS12 形式にエクスポートするには、**pk12util** コマンドを使用します。たとえば、NSS データベース **/root/certdb** から **~/idm_user.p12** ファイルへ、**idm_user** ニックネームを持つ証明書をエクスポートする場合は、次のコマンドを実行します。

```
# pk12util -d ~/certdb -o ~/idm_user.p12 -n idm_user
Enter Password or Pin for "NSS Certificate DB":
Enter password for PKCS12 file:
Re-enter password:
pk12util: PKCS12 EXPORT SUCCESSFUL
```

8. **idm_user** の証明書認証を有効にするホストに、証明書を転送します。

```
# scp ~/idm_user.p12 idm_user@client.idm.example.com:/home/idm_user/
```

9. セキュリティー上の理由から、証明書が転送されたホストの、.pkcs12 ファイルが格納されているディレクトリーに、other グループがアクセスできないようにします。

```
# chmod o-rwx /home/idm_user/
```

10. セキュリティー上の理由から、一時 NSS データベースおよび .pkcs12 ファイルを、サーバーから削除します。

```
# rm ~/certdb/
# rm ~/idm_user.p12
```

69.3. 証明書とユーザーが互いにリンクしていることの確認

注記

ユーザーの証明書が IdM CA によって発行された場合は、この手順を実行する必要はありません。

証明書が機能するには、証明書が、それを使用して Identity Management (IdM) に認証を受けるユーザーにリンクされていることを確認する必要があります。

- 証明書が、Identity Management 環境外の認証局から提供されている場合は、ユーザーアカウントの証明書へのリンク に記載されている手順に従って、ユーザーと証明書をリンクします。
- 証明書が Identity Management CA により提供されている場合は、その証明書がユーザーエントリーに自動的に追加されているため、証明書をユーザーアカウントにリンクする必要はありません。IdM で新しい証明書を作成する方法の詳細は、新しいユーザー証明書を要求し、クライアントにエクスポート を参照してください。

69.4. 証明書認証を有効にするためのブラウザーの設定

Web UI を使用した Identity Management (IdM) へのログイン時に証明書で認証できるようにするには、ユーザーおよび関連の認証局 (CA) 証明書を Mozilla Firefox または Google Chrome ブラウザーにインポートする必要があります。ブラウザーが実行しているホスト自体は、IdM ドメインの一部である必要はありません。

IdM が Web UI への接続をサポートしているブラウザーは以下のとおりです。

- Mozilla Firefox 38 以降
- Google Chrome 46 以降

次の手順は、Mozilla Firefox 57.0.1 ブラウザーを設定する方法を説明します。

前提条件

- PKCS #12 形式で自由にブラウザーにインポートできる ユーザー証明書 がある。

手順

1. Firefox を開き、**設定 → プライバシーとセキュリティー** に移動します。

図69.1 設定のプライバシーおよびセキュリティーセクション

2. **証明書を表示** をクリックします。

図69.2 プライバシーおよびセキュリティーで証明書を表示

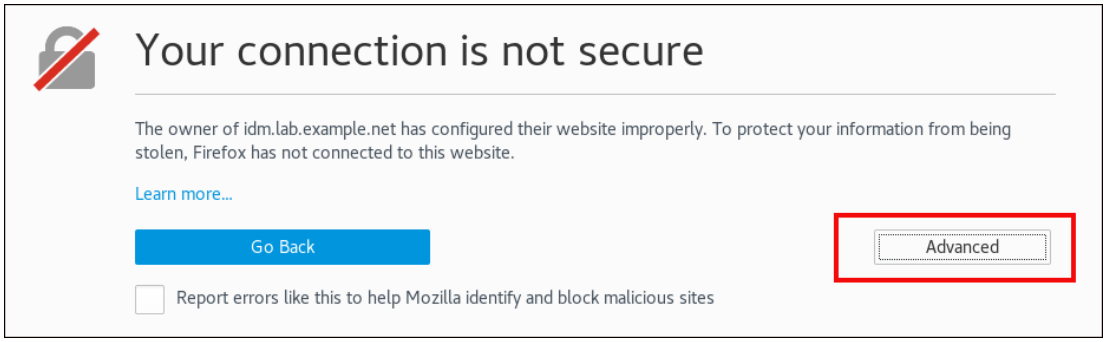
3. **あなたの証明書** タブで、**インポート** をクリックします。PKCS12 形式のユーザー証明書を見つけて開きます。**OK** をクリックし、**OK** をクリックします。

4. Identity Management 認証局が、Firefox で信頼できる認証局として認識されていることを確認します。

   a. IdM CA 証明書をローカルに保存します。

   - Firefox アドレスバーに IdM サーバーの名前を入力し、IdM の Web UI に移動します。接続が安全ではないことを警告するページで、**詳細** をクリックします。

     図69.3 安全ではない接続

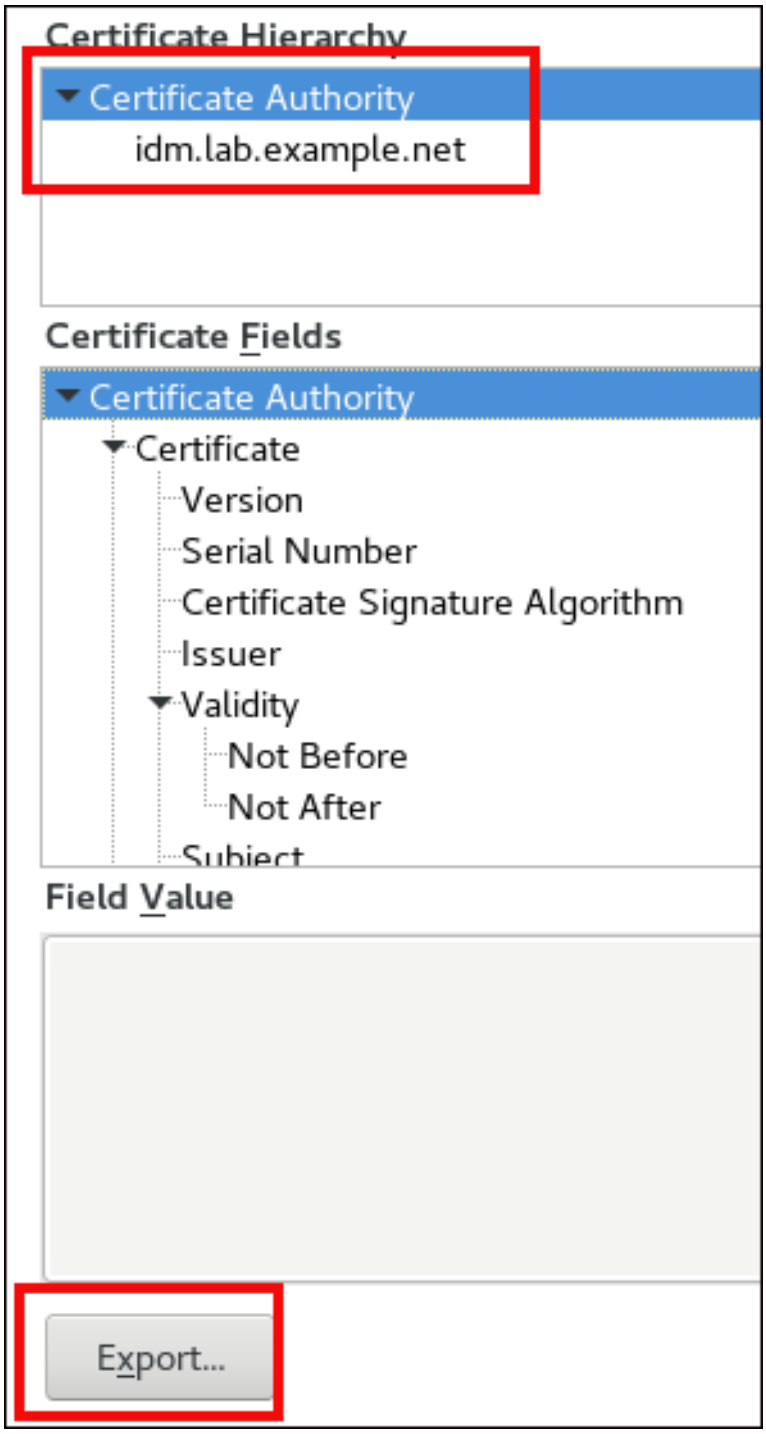


   - **例外を追加** します。**表示** をクリックします。

図69.4 証明書の詳細の表示

- **詳細** タブで、**認証局** フィールドを強調表示します。

図69.5 CA 証明書のエクスポート



- **Export** をクリックします。CA 証明書を、**CertificateAuthority.crt** ファイルとして保存し、**閉じる** をクリックして、**キャンセル** をクリックします。

b. IdM CA 証明書を、信頼できる認証局の証明書として Firefox にインポートします。

- Firefox を起動し、設定に移動して、**プライバシーおよびセキュリティー**に移動します。

図69.6 設定のプライバシーおよびセキュリティーセクション

- **証明書を表示** をクリックします。

図69.7 プライバシーおよびセキュリティーで証明書を表示

- **認証機関** タブで、**インポート** をクリックします。**CertificateAuthority.crt** ファイルで、上の手順で保存した CA 証明書を見つけて開きます。証明書を信頼し、Web サイトを識別したら、**OK** をクリックし、**OK** をクリックします。

5. Identity Management ユーザーとして証明書を使用した Identity Management Web UI の認証に進みます。

69.5. IDENTITY MANAGEMENT ユーザーとして証明書を使用した IDENTITY MANAGEMENT WEB UI の認証

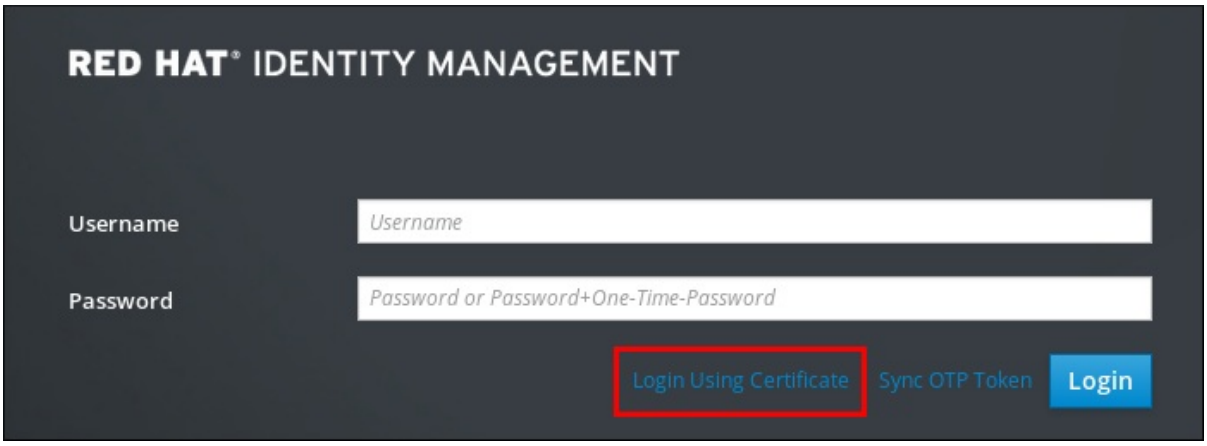
Identity Management クライアントのデスクトップに保存されている証明書を使用して、ユーザーが Identity Management (IdM) の Web UI を認証するには、次の手順に従います。

手順

1. ブラウザーで、Identity Management の Web UI (例: **https://server.idm.example.com/ipa/ui**) に移動します。

2. **Login Using Certificate** をクリックします。

図69.8 Identity Management の Web UI でLogin Using Certificate



3. ユーザーの証明書がすでに選択されているはずです。**Remember this decision** の選択を解除して、**OK** をクリックします。

これで、証明書に対応するユーザーとして認証されました。

関連情報

- Configuring Identity Management for smart card authentication を参照してください。

69.6. 証明書を使用して CLI への認証を可能にするように IDM クライアントを設定

IdM クライアントのコマンドラインインターフェイス (CLI) で、IdM ユーザーに対して証明書認証を有効にするには、IdM ユーザーの証明書および秘密鍵を IdM クライアントにインポートします。ユーザー証明書の作成と転送方法の詳細は、新しいユーザー証明書を要求し、クライアントにエクスポート を参照してください。

手順

- IdM クライアントにログインし、ユーザーの証明書と秘密鍵を含む .p12 ファイルを用意します。Kerberos TGT (Ticket Granting Ticket) ファイルを取得してキャッシュを取得するには、**-X** オプションに **X509_username:/path/to/file.p12** 属性を使用し、ユーザーのプリンシパルで **kinit** コマンドを実行して、ユーザーの X509 識別情報の場所を指定します。たとえば、**~/idm_user.p12** ファイルに保存されている識別情報を使用して **idm_user** の TGT を取得するには、次のコマンドを実行します。

```
$ kinit -X X509_idm_user='PKCS12:~/idm_user.p12' idm_user
```

注記

このコマンドは、.pem ファイル形式 (**kinit -X X509_username='FILE:/path/to/cert.pem,/path/to/key' user_principal**) にも対応しています。

第70章 IDM CA 更新サーバーの使用

70.1. IDM CA 更新サーバーの説明

組み込みの認証局 (CA) を使用する Identity Management (IdM) デプロイメントでは、CA 更新サーバーが IdM システム証明書を維持および更新します。IdM デプロイメントを確実に堅牢化します。

IdM システム証明書には、以下が含まれます。

- **IdM CA** 認証
- **OCSP** 署名証明書
- **IdM CA サブシステム** 証明書
- **IdM CA 監査署名** 証明書
- **IdM 更新エージェント** (RA) 証明書
- **KRA** トランスポート証明書およびストレージ証明書

システム証明書の特徴は、鍵がすべての CA レプリカで共有されることです。一方、IdM サービス証明書 (**LDAP** 証明書、**HTTP** 証明書、**PKINIT** 証明書など) には、さまざまな IdM CA サーバーに、さまざまな鍵ペアと発行先名があります。

IdM トポロジーでは、デフォルトで、最初の IdM CA サーバーが CA 更新サーバーになります。

注記

IdM CA は、アップストリームのドキュメントでは **Dogtag** と呼ばれています。

CA 更新サーバーのロール

IdM CA 証明書、**IdM CA サブシステム** 証明書、および **IdM RA** 証明書は、IdM デプロイメントに重要です。各証明書は、**/etc/pki/pki-tomcat/** ディレクトリーの NSS データベースおよび LDAP データベースのエントリーに保存されます。LDAP に保存されている証明書は、NSS データベースに保存されている証明書に一致する必要があります。一致しない場合は、IdM フレームワークと IdM CA、および IdM CA と LDAP との間に認証エラーが発生します。

すべての IdM CA レプリカは、すべてのシステム証明書への追跡リクエストがあります。統合 CA を備えた IdM デプロイメントに CA 更新サーバーが含まれていない場合には、各 IdM CA サーバーはシステム証明書の更新を個別に要求します。これにより、異なる CA レプリカにさまざまなシステム証明書が含まれるようになり、認証に失敗します。

CA レプリカの 1 つを更新サーバーとすることで、システム証明書が必要に応じて一度だけ更新されるため、認証が失敗しないようになります。

CA レプリカ上の certmonger サービスのロール

すべての IdM CA レプリカで実行している **certmonger** サービスは、**dogtag-ipa-ca-renew-agent** 更新ヘルパーを使用して、IdM システム証明書を追跡します。更新ヘルパープログラムは、CA 更新マスター設定を読み取ります。CA 更新サーバー以外の各 CA レプリカで、更新ヘルパーが **ca_renewal** LDAP エントリーから最新のシステム証明書を取得します。**certmonger** の更新の試みが正確に行われる際に、非決定論により、CA 更新サーバーが実際に証明書を更新する前に、**dogtag-ipa-ca-renew-agent** ヘルパーがシステム証明書の更新を試みることがあります。この状況が発生すると、CA レプリ

カの **certmonger** サービスに、すぐに有効期限が切れる古い証明書が提供されます。**certmonger** は、これデータベースにすでに保存されているのと同じ証明書であることを認識し、CA 更新サーバーから更新された証明書を取得できるまで、個々の試行の間に少し遅れて証明書の更新を試行し続けます。

IdM CA 更新サーバーの適切な機能

埋め込み CA のある IdM デプロイメントは、IdM CA でインストールされた、または後で IdM CA サーバーがインストールされた IdM デプロイメントです。組み込み CA を使用した IdM デプロイメントでは、常に更新サーバーとして設定された CA レプリカが 1 つだけ必要になります。更新サーバーはオンラインで完全に機能し、その他のサーバーで正しく複製する必要があります。

ipa server-del コマンド、**ipa-replica-manage del** コマンド、**ipa-csreplica-manage del** コマンド、または **ipa-server-install --uninstall** コマンドを使用して、現在の CA 更新サーバーを削除すると、別の CA レプリカが CA 更新サーバーとして自動的に割り当てられます。このポリシーは、更新されたサーバー設定を有効に保つようにします。

このポリシーは、以下の状況を対象としません。

- **オフライン更新サーバー**
  更新サーバーが長期間オフラインであると、更新ウィンドウが表示されない場合があります。この場合、更新されていないすべてのサーバーでは、証明書が期限切れになるまで現在のシステム証明書を再インストールし続けます。これが発生すると、証明書が失効した場合でも、その他の証明書の更新が失敗する可能性があるため、IdM デプロイメントが中断されます。

  現行の更新サーバーがオフラインになり、長期間利用できない状況を防ぐには、新規の CA 更新サーバーを手動で割り当てる ことを検討してください。

- **レプリケーションの問題**
  更新サーバーと他の CA レプリカとの間でレプリケーションの問題が存在する場合は、更新が成功する可能性もありますが、その他の CA レプリカが、期限切れになる前に更新された証明書を取得できなくなる可能性があります。

  この問題を回避するには、レプリカ合意が正しく機能することを確認してください。詳細は、RHEL 7 の **Linux ドメイン ID、認証、およびポリシーガイド** の 一般的 または 特定 のレプリケーションのトラブルシューティングガイドラインを参照してください。

70.2. IDM CA 更新サーバーの変更およびリセット

CA 更新サーバーの使用を止めると、Identity Management (IdM) により、IdM CA サーバーのリストから新しい CA 更新サーバーが自動的に選択されます。システム管理者は、選択に影響を与えることはできません。

新しい IdM CA 更新サーバーを選択できるようにするには、システム管理者が交換を手動で実行する必要があります。CA 更新サーバーを選択してから、現在の更新サーバーの使用を停止するプロセスを開始します。

現在の CA 更新サーバー設定が無効な場合は、IdM CA 更新サーバーをリセットします。

この手順では、CA 更新サーバーを変更またはリセットします。

前提条件

- IdM 管理者認証情報がある。

手順

1. IdM 管理者認証情報を取得します。

```
~]$ kinit admin
Password for admin@IDM.EXAMPLE.COM:
```

2. 任意で、デプロイメント内のどの IdM サーバーが新しい CA 更新サーバーになるのに必要な CA のロールを持っているかを確認するには、次のコマンドを実行します。

```
~]$ ipa server-role-find --role 'CA server'
----------------------
2 server roles matched
----------------------
  Server name: server.idm.example.com
  Role name: CA server
  Role status: enabled

  Server name: replica.idm.example.com
  Role name: CA server
  Role status: enabled
----------------------------
Number of entries returned 2
----------------------------
```

   デプロイメントには、2 つの CA サーバーがあります。

3. 必要に応じて、どの CA サーバーが現在の CA 更新サーバーであるかを確認するには、次のコマンドを実行します。

```
~]$ ipa config-show | grep 'CA renewal'
  IPA CA renewal master: server.idm.example.com
```

   現在の更新サーバーは **server.idm.example.com** です。

4. 更新サーバー設定を変更するには、**--ca-renewal-master-server** オプションを指定して **ipa config-mod** ユーティリティーを使用します。

```
~]$ ipa config-mod --ca-renewal-master-server replica.idm.example.com | grep 'CA renewal'
  IPA CA renewal master: replica.idm.example.com
```

重要

以下を使用して、新しい CA 更新サーバーに切り替えることもできます。

- **ipa-cacert-manage --renew** コマンド。このコマンドは、CA 証明書を更新し、**かつ**コマンドを実行する CA サーバーを新しい CA 更新サーバーにします。
- **ipa-cert-fix** コマンド。このコマンドは、期限切れの証明書が失敗の原因になっている場合にデプロイメントを回復します。また、コマンドを実行する CA サーバーを新しい CA 更新サーバーにします。
  詳細は IdM がオフライン時に期限切れのシステム証明書の更新 を参照してください。

第71章 外部署名された CA 証明書の管理

アイデンティティー Management (IdM) は、さまざまなタイプの認証局 (CA) 設定を提供します。IdM を統合 CA を使用してインストールするか、外部 CA を使用してインストールするかを選択できます。インストール時に使用する CA のタイプを指定する必要があります。ただし、インストールすると、外部署名 CA から自己署名 CA に、またはその逆に移行できます。さらに、自己署名 CA は自動的に更新されますが、外部署名 CA 証明書を必ず更新する必要があります。外部署名された CA 証明書を管理するには、必要に応じて関連セクションを参照してください。

- 外部署名された CA を使用した IdM のインストール:
  - IdM サーバーのインストール: 統合 DNS と外部 CA を root CA として使用する場合
  - IdM サーバーのインストール: 統合 DNS なしで外部 CA を root CA として使用する場合
- 外部署名 CA から自己署名 CA への切り替え。
- 自己署名 CA から外部署名 CA への切り替え。
- 外部署名された CA 証明書を更新します。

71.1. IDM での外部署名 CA から自己署名 CA への切り替え

この手順は、外部署名から、Identity Management (IdM) 認証局 (CA) の自己署名証明書に切り替えます。自己署名の CA の場合、CA 証明書の更新は自動的に管理されます。システム管理者は、外部認証局に証明書署名リクエスト (CSR) を提出する必要はありません。

外部署名から自己署名の CA へ切り替える場合は、CA 証明書を置き換えます。以前の CA が署名する証明書は有効のままで、今でも使用されています。たとえば、**LDAP** 証明書の証明書チェーンは、自己署名の CA に移動した後も変更されません。

> **external_CA** certificate > **IdM CA** certificate > **LDAP** certificate

前提条件

- IdM CA 更新サーバーおよびすべての IdM クライアントとサーバーへの **root** アクセス権がある。

手順

1. IdM CA 更新サーバーで、CA 証明書を自己署名として更新します。

```
# ipa-cacert-manage renew --self-signed
Renewing CA certificate, please wait
CA certificate successfully renewed
The ipa-cacert-manage command was successful
```

2. **root** として、残りのすべての IdM サーバーとクライアントに **SSH** で接続します。以下に例を示します。

```
# ssh root@idmclient01.idm.example.com
```

3. IdM クライアントで、サーバーからの証明書を使用して、ローカルの IdM 証明書データベースを更新します。

```
[idmclient01 ~]# ipa-certupdate
Systemwide CA database updated.
Systemwide CA database updated.
The ipa-certupdate command was successful
```

4. 必要に応じて、更新が成功したかどうかを確認して、新しい CA 証明書を **/etc/ipa/ca.crt** ファイルに追加します。

```
[idmclient01 ~]$ openssl crl2pkcs7 -nocrl -certfile /etc/ipa/ca.crt | openssl pkcs7 -
print_certs -text -noout
[...]
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 39 (0x27)
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: O=IDM.EXAMPLE.COM, CN=Certificate Authority
        Validity
            Not Before: Jul  1 16:32:45 2019 GMT
            Not After : Jul  1 16:32:45 2039 GMT
        Subject: O=IDM.EXAMPLE.COM, CN=Certificate Authority
[...]
```

この出力は、新規 CA 証明書が古い CA 証明書と共にリストされているため、更新が正常に行われたことを示しています。

71.2. IDM での自己署名 CA から外部署名 CA への切り替え

IdM で自己署名 CA から外部署名 CA に切り替えることができます。IdM で外部署名された CA に切り替えると、IdM CA サーバーは外部 CA のサブ CA になります。また、CA 証明書の更新は自動的に管理されないため、システム管理者は外部機関に証明書署名要求 (CSR) を送信する必要があります。

外部署名された CA に切り替えるには、CSR が外部 CA によって署名される必要があります。外部 CA を使用した IdM CA 更新サーバー証明書の更新 の手順に従って、IdM で自己署名 CA に切り替えます。

71.3. 外部 CA を使用した IDM CA 更新サーバー証明書の更新

以下の手順に従って、外部 CA を使用して Identity Management (IdM) 認証局 (CA) 証明書を更新し、証明書署名要求 (CSR) に署名します。この設定では、IdM CA サーバーは、外部 CA のサブ CA です。外部の CA は、Active Directory Certificate Server (AD CS) を使用することができますが、必須ではありません。

外部認証局が AD CS の場合は、CSR に、IdM CA 証明書に必要なテンプレートを指定できます。証明書テンプレートは、証明書要求の受信時に CA が使用するポリシーおよびルールを定義します。AD の証明書テンプレートは、IdM の証明書プロファイルに対応します。

オブジェクト識別子 (OID) で特定の AD CS テンプレートを定義できます。OID は、分散アプリケーションのデータ要素、構文などを一意に識別するために、さまざまな発行機関が発行する一意の数値です。

または、特定の AD CS テンプレートを名前で定義することもできます。たとえば、IdM CA から AD CS に送信された CSR で使用されるデフォルトプロファイルの名前は **subCA** です。

CSR で OID または名前を指定してプロファイルを定義するには、**external-ca-profile** オプションを使用します。詳細は、**ipa-cacert-manage** man ページを参照してください。

既製の証明書テンプレートを使用する以外に、AD CS でカスタム証明書テンプレートを作成し、CSR で使用できます。

前提条件

- IdM CA 更新サーバーへの root のアクセス権がある。

手順

この手順では、現在の CA 証明書が自己署名の証明書であるか、外部署名であるかに関係なく、外部署名を使用して IdM CA の証明書を更新します。

1. 外部 CA に送信される CSR を作成します。

   - 外部 CA が AD CS の場合は、**--external-ca-type=ms-cs** オプションを使用します。デフォルトの **subCA** テンプレート以外のテンプレートが必要な場合は、**--external-ca-profile** を使用してこれを指定します。

     ```
     ~]# ipa-cacert-manage renew --external-ca --external-ca-type=ms-cs [--external-ca-profile=PROFILE]
     Exporting CA certificate signing request, please wait
     The next step is to get /var/lib/ipa/ca.csr signed by your CA and re-run ipa-cacert-manage as:
     ipa-cacert-manage renew --external-cert-file=/path/to/signed_certificate --external-cert-file=/path/to/external_ca_certificate
     The ipa-cacert-manage command was successful
     ```

   - 外部 CA が AD CS ではない場合は、以下のようになります。

     ```
     ~]# ipa-cacert-manage renew --external-ca
     Exporting CA certificate signing request, please wait
     The next step is to get /var/lib/ipa/ca.csr signed by your CA and re-run ipa-cacert-manage as:
     ipa-cacert-manage renew --external-cert-file=/path/to/signed_certificate --external-cert-file=/path/to/external_ca_certificate
     The ipa-cacert-manage command was successful
     ```

     この出力は、CSR が作成され、**/var/lib/ipa/ca.csr** ファイルに保存されていることを示しています。

2. **/var/lib/ipa/ca.csr** にある CSR を外部 CA に送信します。このプロセスは、外部 CA として使用するサービスにより異なります。

3. 発行した証明書と、ベース 64 でエンコードされたブロブで CA を発行する CA 証明書チェーンを取得します。以下に例を示します。

   - 外部 CA が AD CS ではない場合の PEM ファイル

   - 外部 CA が AD CS の場合の Base_64 証明書
     プロセスは、各証明書サービスによって異なります。通常は Web ページか通知メールにダウンロードリンクがあり、管理者が、必要なすべての証明書をダウンロードできるようになっています。

外部 CA が AD CS で、Microsoft Windows 認証局管理ウィンドウから既知のテンプレートで CSR を送信した場合、AD CS は証明書を直ちに発行し、その証明書を保存する証明書の保存ダイアログが AD CS Web インターフェイスに表示されます。

4. **ipa-cacert-manage renew** コマンドを再度実行し、完全な証明書チェーンを提供するのに必要な CA 証明書ファイルをすべて追加します。**--external-cert-file** オプションを複数回使用して、必要な数だけファイルを指定します。

```
~]# ipa-cacert-manage renew --external-cert-file=/path/to/signed_certificate --external-cert-file=/path/to/external_ca_certificate_1 --external-cert-file=/path/to/external_ca_certificate_2
```

5. すべての IdM サーバーおよびクライアントで、サーバーの証明書でローカルの IdM 証明書データベースを更新します。

```
[client ~]$ ipa-certupdate
Systemwide CA database updated.
Systemwide CA database updated.
The ipa-certupdate command was successful
```

6. 必要に応じて、更新が成功したかどうかを確認して、新しい CA 証明書を **/etc/ipa/ca.crt** ファイルに追加します。

```
[client ~]$ openssl crl2pkcs7 -nocrl -certfile /etc/ipa/ca.crt | openssl pkcs7 -print_certs -text -noout
[...]
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 39 (0x27)
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: O=IDM.EXAMPLE.COM, CN=Certificate Authority
        Validity
            Not Before: Jul  1 16:32:45 2019 GMT
            Not After : Jul  1 16:32:45 2039 GMT
        Subject: O=IDM.EXAMPLE.COM, CN=Certificate Authority
[...]
```

この出力は、新規 CA 証明書が古い CA 証明書と共にリストされているため、更新が正常に行われたことを示しています。

第72章 IDM がオフライン時に期限切れのシステム証明書の更新

システム証明書の期限が切れると、Identity Management (IdM) が起動できません。IdM は、**ipa-cert-fix** ツールを使用して、このような状況であってもシステム証明書の更新に対応します。

前提条件

- IdM が、Red Hat Enterprise Linux 8.1 以降にのみインストールされている。
- ホストで **ipactl start --ignore-service-failures** コマンドを入力して、LDAP サービスが実行中であることを確認します。

72.1. CA 更新サーバーでの期限切れのシステム証明書の更新

以下の手順に従って、期限切れの IdM 証明書に **ipa-cert-fix** ツールを適用します。

重要

CA 更新サーバーではない CA (認証局) ホストで **ipa-cert-fix** ツールを実行し、ユーティリティーが共有証明書を更新すると、そのホストは自動的にドメイン内の新しい CA 更新サーバーになります。不整合を避けるために、ドメインには常に CA 更新サーバー 1 つだけを設定する必要があります。

前提条件

- 管理者権限でサーバーにログインしている。

手順

1. (オプション) システムをバックアップします。これは、**ipa-cert-fix** が **nssdbs** に対して元に戻せない変更を行うため、強く推奨されます。**ipa-cert-fix** は LDAP に対しても変更を行うため、クラスター全体をバックアップすることも推奨されます。
2. **ipa-cert-fix** ツールを起動して、システムを分析し、更新を必要とする期限切れの証明書のリストを表示します。

```
# ipa-cert-fix
...
The following certificates will be renewed:

Dogtag sslserver certificate:
  Subject: CN=ca1.example.com,O=EXAMPLE.COM 201905222205
  Serial:  13
  Expires: 2019-05-12 05:55:47
...
Enter "yes" to proceed:
```

3. 更新プロセスを開始するには、**yes** を入力します。

```
Enter "yes" to proceed: yes
Proceeding.
Renewed Dogtag sslserver certificate:
  Subject: CN=ca1.example.com,O=EXAMPLE.COM 201905222205
  Serial:  268369925
```

```
  Expires: 2021-08-14 02:19:33
...

Becoming renewal master.
The ipa-cert-fix command was successful
```

ipa-cert-fix が期限切れの証明書をすべて更新する前に、最大 1 分かかる場合があります。

4. 必要に応じて、すべてのサービスが現在実行していることを確認します。

```
# ipactl status
Directory Service: RUNNING
krb5kdc Service: RUNNING
kadmin Service: RUNNING
httpd Service: RUNNING
ipa-custodia Service: RUNNING
pki-tomcatd Service: RUNNING
ipa-otpd Service: RUNNING
ipa: INFO: The ipactl command was successful
```

この時点で、証明書が更新され、サービスが実行しています。次の手順は、IdM ドメイン内のその他のサーバーを確認します。

注記

複数の CA サーバーで証明書を修復する必要がある場合は、次のコマンドを実行します。

1. トポロジー全体で LDAP レプリケーションが機能していることを確認したら、上記の手順に従って、最初に 1 つの CA サーバーで **ipa-cert-fix** を実行します。
2. 別の CA サーバーで **ipa-cert-fix** を実行する前に、(別の CA サーバーの) **getcert-resubmit** を介して共有証明書の Certmonger 更新をトリガーして、共有証明書の不必要な更新を回避します。

72.2. 更新後の IDM ドメイン内の他の IDM サーバーの検証

ipa-cert-fix ツールで、CA 更新サーバーを更新したら、以下を行う必要があります。

- ドメインのその他の Identity Management (IdM) サーバーをすべて再起動する。
- certmonger が証明書を更新したかどうかを確認する。
- 期限切れのシステム証明書でその他の認証局 (CA) レプリカがある場合は、証明書も **ipa-cert-fix** ツールで更新する。

前提条件

- 管理者権限でサーバーにログインしている。

手順

1. **--force** パラメーターで IdM を再起動します。

```
# ipactl restart --force
```

- **--force** パラメーターを使用すると、**ipactl** ユーティリティーは個々のサービスの起動失敗を無視します。たとえば、サーバーに期限切れの証明書を持つ CA もあると、**pki-tomcat** サービスが起動に失敗します。**--force** パラメーターを使用しているため、これが予想され、無視されます。

2. 再起動後に、**certmonger** サービスが証明書を更新することを確認します (証明書の状態は MONITORING になります)。

```
# getcert list | egrep '^Request|status:|subject:'
Request ID '20190522120745':
        status: MONITORING
        subject: CN=IPA RA,O=EXAMPLE.COM 201905222205
Request ID '20190522120834':
        status: MONITORING
        subject: CN=Certificate Authority,O=EXAMPLE.COM 201905222205
...
```

certmonger がレプリカ上で共有証明書を更新する前に時間がかかる場合があります。

3. サーバーも CA の場合、上記のコマンドは、**pki-tomcat** サービスが使用する証明書の **CA_UNREACHABLE** を報告します。

```
Request ID '20190522120835':
        status: CA_UNREACHABLE
        subject: CN=ca2.example.com,O=EXAMPLE.COM 201905222205
...
```

4. この証明書を更新するには、**ipa-cert-fix** ユーティリティーを使用します。

```
# ipa-cert-fix
Dogtag sslserver certificate:
  Subject: CN=ca2.example.com,O=EXAMPLE.COM
  Serial:  3
  Expires: 2019-05-11 12:07:11

Enter "yes" to proceed: yes
Proceeding.
Renewed Dogtag sslserver certificate:
  Subject: CN=ca2.example.com,O=EXAMPLE.COM 201905222205
  Serial:  15
  Expires: 2019-08-14 04:25:05

The ipa-cert-fix command was successful
```

これで、すべての IdM 証明書が更新され、正常に機能するようになりました。

第73章 IDM レプリカでまだ有効期限が切れていない場合の WEB サーバーと LDAP サーバーの証明書の置き換え

Identity Management (IdM) システム管理者は、IdM サーバーで実行している Web (または **httpd**) サービスおよび LDAP (または **Directory**) サービスの証明書を手動で置き換えることができます。たとえば、証明書の有効期限が近づいていて、**certmonger** ユーティリティーが証明書を自動的に更新するように設定されていない場合、または証明書が外部の認証局 (CA) によって署名されている場合に、これが必要になることがあります。

この例では、**server.idm.example.com** IdM サーバーで実行しているサービスの証明書をインストールします。外部 CA から証明書を取得する。

注記

HTTP サービス証明書と LDAP サービス証明書には異なる IdM サーバーに異なるキーペアとサブジェクト名があるため、各 IdM サーバーで証明書を個別に更新する必要があります。

前提条件

- IdM サーバーが複製合意を持つトポロジー内の少なくとも 1 つの他の IdM レプリカで、Web および LDAP 証明書は引き続き有効です。これは **ipa-server-certinstall** コマンドの前提条件です。このコマンドは、他の IdM レプリカと通信するために **TLS** 接続を必要とします。ただし、証明書が無効な場合、そのような接続は確立できず、**ipa-server-certinstall** コマンドは失敗します。その場合は、Web サーバーと LDAP サーバーの証明書が IdM デプロイメント全体で期限切れになった場合の置き換え を参照してください。
- IdM サーバーへの **root** アクセス権限がある。
- **Directory Manager** パスワードを把握している。
- 外部 CA の CA 証明書チェーンを保存しているファイル (**ca_certificate_chain_file.crt**) にアクセスできる。

手順

1. **ca_certificate_chain_file.crt** に含まれる証明書を、追加の CA 証明書として IdM にインストールします。

```
# ipa-cacert-manage install
```

2. **ca_certicate_chain_file.crt** からの証明書を使用して、ローカルの IdM 証明書データベースを更新します。

```
# ipa-certupdate
```

3. **OpenSSL** ユーティリティーを使用して、秘密鍵と証明書署名要求 (CSR) を生成します。

```
$ openssl req -new -newkey rsa:4096 -days 365 -nodes -keyout new.key -out new.csr -addext "subjectAltName = DNS:server.idm.example.com" -subj '/CN=server.idm.example.com,O=IDM.EXAMPLE.COM'
```

CSR を外部 CA に送信します。このプロセスは、外部 CA として使用するサービスにより異なります。CA が証明書に署名したら、証明書を IdM サーバーにインポートします。

4. IdM サーバーで、Apache Web サーバーの古い秘密鍵および証明書を、新しい鍵と、新しく署名した証明書に置き換えます。

```
# ipa-server-certinstall -w --pin=password new.key new.crt
```

上記のコマンドでは、以下のようになります。

- **-w** オプションは、Web サーバーに証明書をインストールすることを指定します。
- **--pin** オプションは、秘密鍵を保護するパスワードを指定します。

5. プロンプトが表示されたら、**Directory Manager** パスワードを入力します。

6. LDAP サーバーの古い秘密鍵および証明書を、新しい鍵と、新しく署名した証明書に置き換えます。

```
# ipa-server-certinstall -d --pin=password new.key new.cert
```

上記のコマンドでは、以下のようになります。

- **-d** オプションは、LDAP サーバーに証明書をインストールすることを指定します。
- **--pin** オプションは、秘密鍵を保護するパスワードを指定します。

7. プロンプトが表示されたら、**Directory Manager** パスワードを入力します。

8. **httpd** サービスを再起動します。

```
# systemctl restart httpd.service
```

9. **Directory** サービスを再起動します。

```
# systemctl restart dirsrv@IDM.EXAMPLE.COM.service
```

10. サーバー上でサブ CA が削除または置き換えられている場合は、クライアントを更新します。

```
# ipa-certupdate
```

関連情報

- IdM と機能する証明書形式への変換
- **ipa-server-certinstall(1)** の man ページ

第74章 IDM デプロイメント全体で期限切れになった WEB サーバーと LDAP サーバーの証明書を置き換える

Identity Management (IdM) は、以下のサービス証明書を使用します。

- LDAP (または **Directory**) サーバー証明書
- Web (または **httpd**) サーバー証明書
- PKINIT 証明書

CA を使用しない IdM デプロイメントでは、**certmonger** はデフォルトで IdM サービス証明書を追跡したり、その有効期限を通知したりしません。IdM システム管理者がこれらの証明書の通知を手動で設定しない場合、または証明書を追跡するように **certmonger** を設定しない場合、証明書は予告なしに期限切れになります。

以下の手順に従って、**server.idm.example.com** IdM サーバーで実行している **httpd** および LDAP サービスの期限切れの証明書を手動で置き換えます。

注記

HTTP および LDAP サービス証明書は、異なる IdM サーバーで異なるキーペアとサブジェクト名を持ちます。したがって、各 IdM サーバーで個別に証明書を更新する必要があります。

前提条件

- トポロジー内の **すべての** IdM レプリカで、HTTP および LDAP 証明書の有効期限が切れています。そうでない場合は、Web サーバーと LDAP サーバーの証明書が IdM レプリカで期限切れになっていない場合の置き換え を参照してください。
- IdM サーバーとレプリカへの **root** アクセス権がある。
- **Directory Manager** パスワードを把握している。
- 次のディレクトリーとファイルのバックアップを作成しました:
  - **/etc/dirsrv/slapd-IDM-EXAMPLE-COM/**
  - **/etc/httpd/alias**
  - **/var/lib/certmonger**
  - **/var/lib/ipa/certs/**

手順

1. 新しい証明書の署名に同じ CA を使用していない場合、またはすでにインストールされている CA 証明書が無効になっている場合は、ローカルデータベース内の外部 CA に関する情報を、外部 CA の有効な CA 証明書チェーンを含むファイルで更新します。.このファイルは、PEM および DER 証明書、PKCS#7 証明書チェーン、PKCS#8、生の秘密鍵、および PKCS#12 形式で受け入れられます。
   a. **ca_certificate_chain_file.crt** で利用可能な証明書を追加の CA 証明書として IdM にインストールします。
      ■

```
# ipa-cacert-manage install ca_certificate_chain_file.crt
```

b. ca_certicate_chain_file.crt からの証明書を使用して、ローカルの IdM 証明書データベースを更新します。

```
# ipa-certupdate
```

2. **httpd** および LDAP の証明書を要求します。

a. **OpenSSL** ユーティリティーを使用して、IdM インスタンスで実行している Apache Web サーバーの証明書署名要求 (CSR) をサードパーティー CA に作成します。

```
$ openssl req -new -newkey rsa:2048 -nodes -keyout /var/lib/ipa/private/httpd.key -
out /tmp/http.csr -addext 'subjectAltName = DNS:server.idm.example.com,
otherName:1.3.6.1.4.1.311.20.2.3;UTF8:HTTP/server.idm.example.com@IDM.EXAM
PLE.COM' -subj '/O=IDM.EXAMPLE.COM/CN=server.idm.example.com'
```

新しい秘密鍵の作成は任意です。元の秘密鍵がまだある場合は、**openssl req** コマンドで **-in** オプションを使用して、要求を読み取る入力ファイル名を指定できます。

b. **OpenSSL** ユーティリティーを使用して、IdM インスタンスで実行している LDAP サーバーの証明書署名要求 (CSR) をサードパーティー CA に作成します。

```
$ openssl req -new -newkey rsa:2048 -nodes -keyout ~/ldap.key -out /tmp/ldap.csr -
addext 'subjectAltName = DNS:server.idm.example.com,
otherName:1.3.6.1.4.1.311.20.2.3;UTF8:ldap/server.idm.example.com@IDM.EXAMP
LE.COM' -subj '/O=IDM.EXAMPLE.COM/CN=server.idm.example.com'
```

新しい秘密鍵の作成は任意です。元の秘密鍵がまだある場合は、**openssl req** コマンドで **-in** オプションを使用して、要求を読み取る入力ファイル名を指定できます。

c. CSR、**/tmp/http.csr** および **tmp/ldap.csr** を外部 CA に送信し、**httpd** の証明書と LDAP の証明書を取得します。このプロセスは、外部 CA として使用するサービスにより異なります。

3. **httpd** の証明書をインストールします。

```
# cp /path/to/httpd.crt /var/lib/ipa/certs/
```

4. LDAP 証明書を NSS データベースにインストールします。

a. [オプション] 利用可能な証明書を一覧表示します。

```
# certutil -d /etc/dirsrv/slapd-IDM-EXAMPLE-COM/ -L
Certificate Nickname                                         Trust Attributes
                                                             SSL,S/MIME,JAR/XPI

Server-Cert                                                  u,u,u
```

デフォルトの証明書のニックネームは **Server-Cert** ですが、別の名前が適用された可能性があります。

b. 前の手順の証明書のニックネームを使用して、古い無効な証明書を NSS データベース (**NSSDB**) から削除します。

```
# certutil -D -d /etc/dirsrv/slapd-IDM-EXAMPLE-COM/ -n 'Server-Cert' -f
/etc/dirsrv/slapd-IDM-EXAMPLE-COM/pwdfile.txt
```

c. **NSSDB** へのインポートプロセスを容易にするために、PKCS12 ファイルを作成します。

```
# openssl pkcs12 -export -in ldap.crt -inkey ldap.key -out ldap.p12 -name Server-
Cert
```

d. 作成した PKCS#12 ファイルを **NSSDB** にインストールします。

```
# pk12util -i ldap.p12 -d /etc/dirsrv/slapd-IDM-EXAMPLE-COM/ -k
/etc/dirsrv/slapd-IDM-EXAMPLE-COM/pwdfile.txt
```

e. 新しい証明書が正常にインポートされたことを確認します。

```
# certutil -L -d /etc/dirsrv/slapd-IDM-EXAMPLE-COM/
```

5. **httpd** サービスを再起動します。

```
# systemctl restart httpd.service
```

6. **Directory** サービスを再起動します。

```
# systemctl restart dirsrv@IDM-EXAMPLE-COM.service
```

7. すべての IdM レプリカで前のすべての手順を実行します。これは、レプリカ間の **TLS** 接続を確立するための前提条件です。

8. 新しい証明書を LDAP ストレージに登録します。

a. Apache サーバーの古い秘密鍵および証明書を、新しい鍵と、新しく署名した証明書に置き換えます。

```
# ipa-server-certinstall -w --pin=password /var/lib/ipa/private/httpd.key
/var/lib/ipa/certs/httpd.crt
```

上記のコマンドでは、以下のようになります。

- **-w** オプションは、Web サーバーに証明書をインストールすることを指定します。
- **--pin** オプションは、秘密鍵を保護するパスワードを指定します。

b. プロンプトが表示されたら、**Directory Manager** パスワードを入力します。

c. LDAP サーバーの古い秘密鍵および証明書を、新しい鍵と、新しく署名した証明書に置き換えます。

```
# ipa-server-certinstall -d --pin=password /etc/dirsrv/slapd-IDM-EXAMPLE-
COM/ldap.key /path/to/ldap.crt
```

上記のコマンドでは、以下のようになります。

- **-d** オプションは、LDAP サーバーに証明書をインストールすることを指定します。

- **--pin** オプションは、秘密鍵を保護するパスワードを指定します。

d. プロンプトが表示されたら、**Directory Manager** パスワードを入力します。

e. **httpd** サービスを再起動します。

```
# systemctl restart httpd.service
```

f. **Directory** サービスを再起動します。

```
# systemctl restart dirsrv@IDM-EXAMPLE-COM.service
```

9. 影響を受ける他のすべてのレプリカで、前の手順のコマンドを実行します。

関連情報

IdM で動作するように証明書形式を変換する * man **ipa-server-certinstall (1)** * How do I manually renew Identity Management (IPA) certificates on RHEL 8 after they have expired?(CA のない IPA)

第75章 IDM CA サーバーでの CRL の生成

IdM デプロイメントで埋め込み認証局 (CA) を使用する場合は、証明書失効リスト (CRL) の生成を 1 つの Identity Management (IdM) サーバーから別のサーバーに移動する必要になることがあります。たとえば、サーバーを別のシステムに移行する場合に必要になる場合があります。

CRL を生成するサーバーは 1 台だけ設定します。CRL パブリッシャーロールを実行する IdM サーバーは通常、CA 更新サーバーロールを実行するサーバーと同じですが、この設定は必須ではありません。CRL パブリッシャーサーバーの使用を停止する前に、別のサーバーを選択して CRL パブリッシャーサーバーロールを実行するように設定します。

75.1. IDM サーバーでの CRL 生成の停止

IdM CRL パブリッシャーサーバーで証明書失効リスト (CRL) の生成を停止するには、**ipa-crlgen-manage** コマンドを使用します。生成を無効にする前に、サーバーで実際に CRL が生成されることを確認してください。その後、無効にすることができます。

前提条件

- Identity Management (IdM) サーバーが、RHEL 8.1 システム以降にインストールされている。
- root としてログインしている。

手順

1. サーバーが CRL を生成しているかどうかを確認します。

```
[root@server ~]# ipa-crlgen-manage status
CRL generation: enabled
Last CRL update: 2019-10-31 12:00:00
Last CRL Number: 6
The ipa-crlgen-manage command was successful
```

2. サーバーで CRL の生成を停止します。

```
[root@server ~]# ipa-crlgen-manage disable
Stopping pki-tomcatd
Editing /var/lib/pki/pki-tomcat/conf/ca/CS.cfg
Starting pki-tomcatd
Editing /etc/httpd/conf.d/ipa-pki-proxy.conf
Restarting httpd
CRL generation disabled on the local host. Please make sure to configure CRL generation on another master with ipa-crlgen-manage enable.
The ipa-crlgen-manage command was successful
```

3. サーバーが CRL の生成を停止したかどうかを確認します。

```
[root@server ~]# ipa-crlgen-manage status
```

サーバーで CRL の生成が停止されました。次の手順は、IdM レプリカで CRL 生成を有効にすることです。

75.2. IDM レプリカサーバーでの CRL 生成の開始

ipa-crlgen-manage コマンドを使用して、IdM CA サーバーで証明書失効リスト (CRL) の生成を開始できます。

前提条件

- Identity Management (IdM) サーバーが、RHEL 8.1 システム以降にインストールされている。
- RHEL システムは、IdM 認証局サーバーが必要である。
- root としてログインしている。

手順

1. CRL の生成を開始します。

```
[root@replica1 ~]# ipa-crlgen-manage enable
Stopping pki-tomcatd
Editing /var/lib/pki/pki-tomcat/conf/ca/CS.cfg
Starting pki-tomcatd
Editing /etc/httpd/conf.d/ipa-pki-proxy.conf
Restarting httpd
Forcing CRL update
CRL generation enabled on the local host. Please make sure to have only a single CRL generation master.
The ipa-crlgen-manage command was successful
```

2. CRL が生成されているかどうかを確認します。

```
[root@replica1 ~]# ipa-crlgen-manage status
CRL generation: enabled
Last CRL update: 2019-10-31 12:10:00
Last CRL Number: 7
The ipa-crlgen-manage command was successful
```

75.3. CRL 更新間隔の変更

証明書失効リスト (CRL) ファイルは、デフォルトでは、アイデンティティー Management Certificate Authority (Idm CA) によって 4 時間ごとに自動的に生成されます。この間隔は以下の手順で変更できます。

手順

1. CRL 生成サーバーを停止します。

```
# systemctl stop pki-tomcatd@pki-tomcat.service
```

2. **/var/lib/pki/pki-tomcat/conf/ca/CS.cfg** ファイルを開き、**ca.crl.MasterCRL.autoUpdateInterval** の値を新しい間隔設定に変更します。たとえば、60 分ごとに CRL を生成するには、次のコマンドを実行します。

```
ca.crl.MasterCRL.autoUpdateInterval=60
```

注記

ca.crl.MasterCRL.autoUpdateInterval パラメーターを更新すると、すでにスケジュールされている CRL の次回更新後に変更が有効になります。

3. CRL 生成サーバーを起動します。

```
# systemctl start pki-tomcatd@pki-tomcat.service
```

関連情報

- IdM レプリカサーバーでの CRL 生成の詳細は、IdM レプリカサーバーでの CRL 生成の開始 を参照してください。

第76章 CA 更新サーバーと CRL パブリッシャーのロールを実行するサーバーの使用の停止

認証局 (CA) 更新サーバーロールおよび 証明書失効リスト (CRL) パブリッシャーロールの両方を実行しているサーバーが 1 台あるとします。このサーバーをオフラインにするか、使用を停止する必要がある場合は、別の CA サーバーを選択して、このロールを実行するように設定します。

この例では、CA 更新サーバーと CRL パブリッシャーのロールに対応しているホスト **server.idm.example.com** の使用を停止する必要があります。この手順では、CA 更新サーバーロールおよび CRL パブリッシャーロールをホスト **replica.idm.example.com** に転送し、IdM 環境から **server.idm.example.com** を削除します。

注記

CA 更新サーバーと CRL パブリッシャーの両方のロールを実行するために同じサーバーを設定する必要はありません。

前提条件

- IdM 管理者認証情報がある。
- 使用を停止しているサーバーの root パスワードがある。
- IdM 環境に、少なくとも 2 つの CA レプリカがある。

手順

1. IdM 管理者認証情報を取得します。

```
[user@server ~]$ kinit admin
Password for admin@IDM.EXAMPLE.COM:
```

2. **(オプション)** どのサーバーが CA 更新サーバーおよび CRL パブリッシャーのロールを実行しているかわからない場合は、以下を実行します。

   a. 現在の CA 更新サーバーを表示します。次のコマンドは、任意の IdM サーバーから実行できます。

```
[user@server ~]$ ipa config-show | grep 'CA renewal'
  IPA CA renewal master: server.idm.example.com
```

   b. ホストが現在の CRL パブリッシャーであるかどうかをテストします。

```
[user@server ~]$ ipa-crlgen-manage status
CRL generation: enabled
Last CRL update: 2019-10-31 12:00:00
Last CRL Number: 6
The ipa-crlgen-manage command was successful
```

   CRL を生成しない CA サーバーは、**CRL generation: disabled** を表示します。

```
[user@replica ~]$ ipa-crlgen-manage status
CRL generation: disabled
The ipa-crlgen-manage command was successful
```

CRL パブリッシャーサーバーが見つかるまで、CA サーバーでこのコマンドを入力し続けます。

c. このロールに対応するために昇格できるその他の CA サーバーをすべて表示します。この環境には 2 台の CA サーバーがあります。

```
[user@server ~]$ ipa server-role-find --role 'CA server'
---------------------
2 server roles matched
---------------------
  Server name: server.idm.example.com
  Role name: CA server
  Role status: enabled
  Server name: replica.idm.example.com
  Role name: CA server
  Role status: enabled
----------------------------
Number of entries returned 2
----------------------------
```

3. **replica.idm.example.com** を CA 更新サーバーとして設定します。

```
[user@server ~]$ ipa config-mod --ca-renewal-master-server replica.idm.example.com
```

4. **server.idm.example.com** で、以下を実行します。

a. 証明書更新タスクを無効にします。

```
[root@server ~]# pki-server ca-config-set ca.certStatusUpdateInterval 0
```

b. IdM サービスを再起動します。

```
[root@server ~]# ipactl restart
```

5. **replica.idm.example.com** で、以下を実行します。

a. 証明書更新タスクを有効にします。

```
[root@replica ~]# pki-server ca-config-unset ca.certStatusUpdateInterval
```

b. IdM サービスを再起動します。

```
[root@replica ~]# ipactl restart
```

6. **server.idm.example.com** で、CRL の生成を中止します。

```
[user@server ~]$ ipa-crlgen-manage disable
Stopping pki-tomcatd
Editing /var/lib/pki/pki-tomcat/conf/ca/CS.cfg
```

```
Starting pki-tomcatd
Editing /etc/httpd/conf.d/ipa-pki-proxy.conf
Restarting httpd
CRL generation disabled on the local host. Please make sure to configure CRL generation on
another master with ipa-crlgen-manage enable.
The ipa-crlgen-manage command was successful
```

7. **replica.idm.example.com** で、CRL の生成を開始します。

```
[user@replica ~]$ ipa-crlgen-manage enable
Stopping pki-tomcatd
Editing /var/lib/pki/pki-tomcat/conf/ca/CS.cfg
Starting pki-tomcatd
Editing /etc/httpd/conf.d/ipa-pki-proxy.conf
Restarting httpd
Forcing CRL update
CRL generation enabled on the local host. Please make sure to have only a single CRL
generation master.
The ipa-crlgen-manage command was successful
```

8. **server.idm.example.com** で IdM サービスを停止します。

```
[root@server ~]# ipactl stop
```

9. **replica.idm.example.com** で、IdM 環境から **server.idm.example.com** を削除します。

```
[user@replica ~]$ ipa server-del server.idm.example.com
```

10. **server.idm.example.com** で、**ipa-server-install --uninstall** コマンドを root アカウントとして使用します。

```
[root@server ~]# ipa-server-install --uninstall
...
Are you sure you want to continue with the uninstall procedure? [no]: yes
```

検証手順

- 現在の CA 更新サーバーを表示します。

```
[user@replica ~]$ ipa config-show | grep 'CA renewal'
  IPA CA renewal master: replica.idm.example.com
```

- **replica.idm.example.com** ホストが CRL を生成していることを確認します。

```
[user@replica ~]$ ipa-crlgen-manage status
CRL generation: enabled
Last CRL update: 2019-10-31 12:10:00
Last CRL Number: 7
The ipa-crlgen-manage command was successful
```

関連情報

- IdM CA 更新サーバーの変更およびリセット

- IdM CA サーバーでの CRL の生成
- IdM レプリカのアンインストール

第77章 CERTMONGER を使用したサービスの IDM 証明書の取得

77.1. CERTMONGER の概要

Identity Management (IdM) が統合 IdM 認証局 (CA) とともにインストールされていると、**certmonger** サービスを使用してシステムおよびサービス証明書を追跡し、更新します。証明書の期限が切れると、**certmonger** は次の方法で更新プロセスを管理します。

- 元の要求で提供されたオプションを使用して、証明書署名要求 (CSR) を再生成する。
- IdM API の cert-request コマンドを使用して、CSR を IdM CA に送信する。
- IdM CA から証明書を受け取る。
- 元の要求で指定されている場合は、事前保存コマンドを実行する。
- 更新要求で指定された場所 (NSS データベースまたはファイル内) に新しい証明書をインストールする。
- 元の要求で指定されている場合は、post-save コマンドを実行する。たとえば、post-save コマンドは、関連するサービスを再起動するように **certmonger** に指示し、サービスが新しい証明書が取得できるようにします。

certmonger が追跡する証明書の種類

証明書は、システム証明書とサービス証明書に分けることができます。

さまざまなサーバーのさまざまなキーペアと発行名を持つサービス証明書 (**HTTP**、**LDAP**、**PKINIT** など) とは異なり、IdM システム証明書とその鍵はすべての CA レプリカで共有されます。IdM システム証明書には、以下が含まれます。

- **IdM CA** 認証
- **OCSP** 署名証明書
- **IdM CA サブシステム** 証明書
- **IdM CA 監査署名** 証明書
- **IdM 更新エージェント** (RA) 証明書
- **KRA** トランスポート証明書およびストレージ証明書

certmonger サービスは、統合 CA を使用した IdM 環境のインストール時に要求された IdM システム証明書およびサービス証明書を追跡します。**certmonger** は、IdM ホストで実行しているその他のサービスに対して、システム管理者が手動で要求した証明書を追跡します。**Certmonger** では、外部 CA 証明書またはユーザー証明書は追跡されません。

Certmonger のコンポーネント

certmonger サービスは、2 つの主要コンポーネントで設定されています。

- **certmonger デーモン** - 証明書のリストを追跡し、更新コマンドを実行するエンジンです。
- **コマンドラインインターフェイス** (CLI) の **getcert** ユーティリティー。これにより、システム管理者は **certmonger** デーモンにアクティブにコマンドを送信できます。

具体的には、システム管理者は、**getcert** ユーティリティーを使用して以下のことができます。

- 新しい証明書を要求する
- **certmonger** が追跡する証明書のリストを表示する
- 証明書の追跡を開始または停止する
- 証明書を更新する

77.2. CERTMONGER を使用したサービスの IDM 証明書の取得

ブラウザーと、Identity Management (IdM) クライアントで実行している Web サービスとの間の通信が安全で暗号化されていることを確認するには、TLS 証明書を使用します。IdM 認証局 (CA) から Web サービスの TLS 証明書を取得します。

以下の手順に従って、**certmonger** を使用して、IdM クライアントで実行しているサービス (**HTTP/my_company.idm.example.com@IDM.EXAMPLE.COM**) の IdM 証明書を取得します。

certmonger 使用して証明書を自動的に要求するということは、**certmonger** 更新の期限が切れたときに証明書を管理および更新することを意味します。

certmonger がサービス証明書を要求したときの動作の視覚的な表現については、サービス証明書を要求する certmonger の通信フロー を参照してください。

前提条件

- Web サーバーが、IdM クライアントとして登録されている。
- 手順を実行している IdM クライアントへのルートアクセス権限がある。
- 証明書を要求しているサービスは、前もって IdM に用意する必要はない。

手順

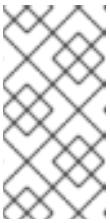
1. **HTTP** サービスが稼働している IdM クライアント **my_company.idm.example.com** で、以下を指定する **HTTP/my_company.idm.example.com@IDM.EXAMPLE.COM** プリンシパルに対応するサービスの証明書を要求します。
   - 証明書は、ローカルの **/etc/pki/tls/certs/httpd.pem** ファイルに保存されます。
   - 秘密鍵は、ローカルの **/etc/pki/tls/private/httpd.key** ファイルに保存されます。
   - **SubjectAltName** の extensionRequest が、 **my_company.idm.example.com** の DNS 名の署名要求に追加されます。

     ```
     # ipa-getcert request -K HTTP/my_company.idm.example.com -k
     /etc/pki/tls/private/httpd.key -f /etc/pki/tls/certs/httpd.pem -g 2048 -D
     my_company.idm.example.com -C "systemctl restart httpd"
     New signing request "20190604065735" added.
     ```

     上記のコマンドでは、以下のようになります。

     - **ipa-getcert request** コマンドは、証明書が IdM CA から取得することを示しています。**ipa-getcert request** コマンドは、**getcert request -c IPA** のショートカットです。

- **-g** オプションは、生成先のキーのサイズ (設定されていない場合) を指定します。
- **-D** オプションは、要求に追加する DNS 値 **SubjectAltName** を指定します。
- **-C** オプションは、証明書の取得後に **httpd** サービスを再起動するように **certmonger** に指示します。
- 特定のプロファイルで証明書を発行するように指定する場合は、**-T** オプションを使用します。
- 指定した CA から名前付き発行者を使用して証明書を要求するには、**-X ISSUER** オプションを使用します。



注記

RHEL 8 は、RHEL 7 で使用されるものとは異なる SSL モジュール (Apache) を使用します。SSL モジュールは、NSS ではなく OpenSSL に依存しています。このため、RHEL 8 では、NSS データベースを使用して **HTTPS** 証明書と秘密鍵を保存することができません。

2. 必要に応じて、リクエストの状況を確認するには、次のコマンドを実行します。

```
# ipa-getcert list -f /etc/pki/tls/certs/httpd.pem
Number of certificates and requests being tracked: 3.
Request ID '20190604065735':
    status: MONITORING
    stuck: no
    key pair storage: type=FILE,location='/etc/pki/tls/private/httpd.key'
    certificate: type=FILE,location='/etc/pki/tls/certs/httpd.crt'
    CA: IPA
[...]
```

この出力は、要求が **MONITORING** 状況であることを表しています。これは、証明書が取得されていることを示しています。キーペアと証明書の場所は、要求された場所です。

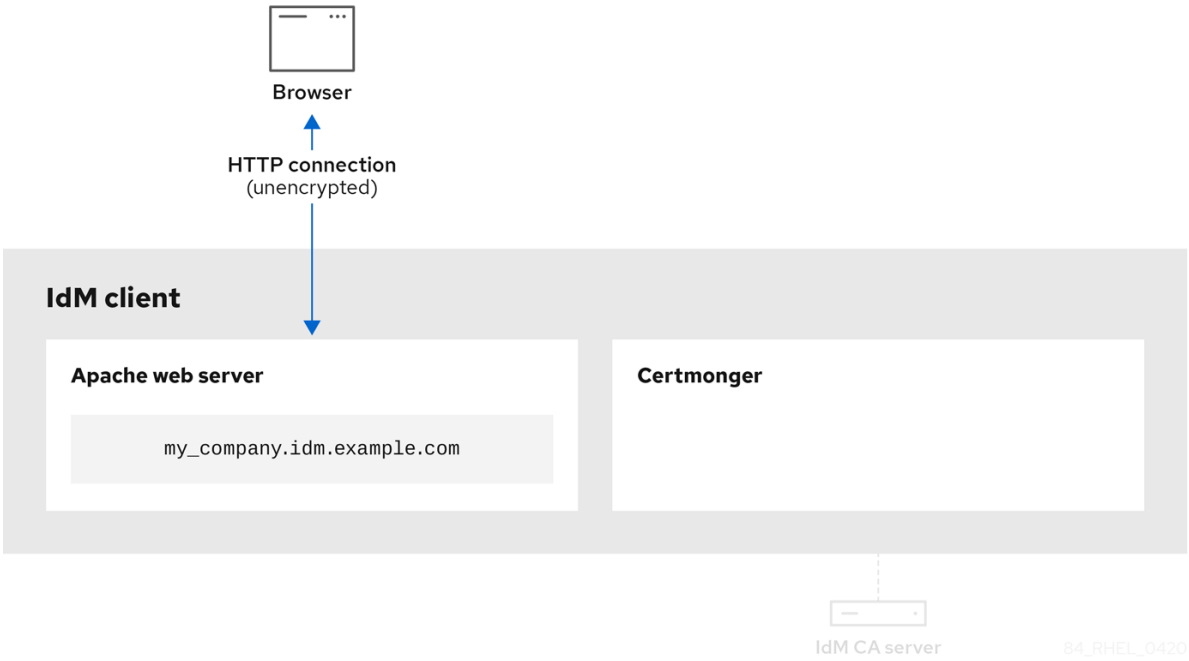
77.3. サービス証明書を要求する CERTMONGER の通信フロー

これらの図では、**certmonger** が Identity Management (IdM) 認証局 (CA) サーバーからサービス証明書を要求したときに何が起こるかを段階をおって紹介しています。シーケンスは次の図で設定されています。

- 暗号化されていない通信
- サービス証明書を要求する certmonger
- サービス証明書を発行する IdM CA
- Certmonger によるサービス証明書の適用
- 古い証明書が有効期限に近づいているときに新しい証明書を要求する certmonger

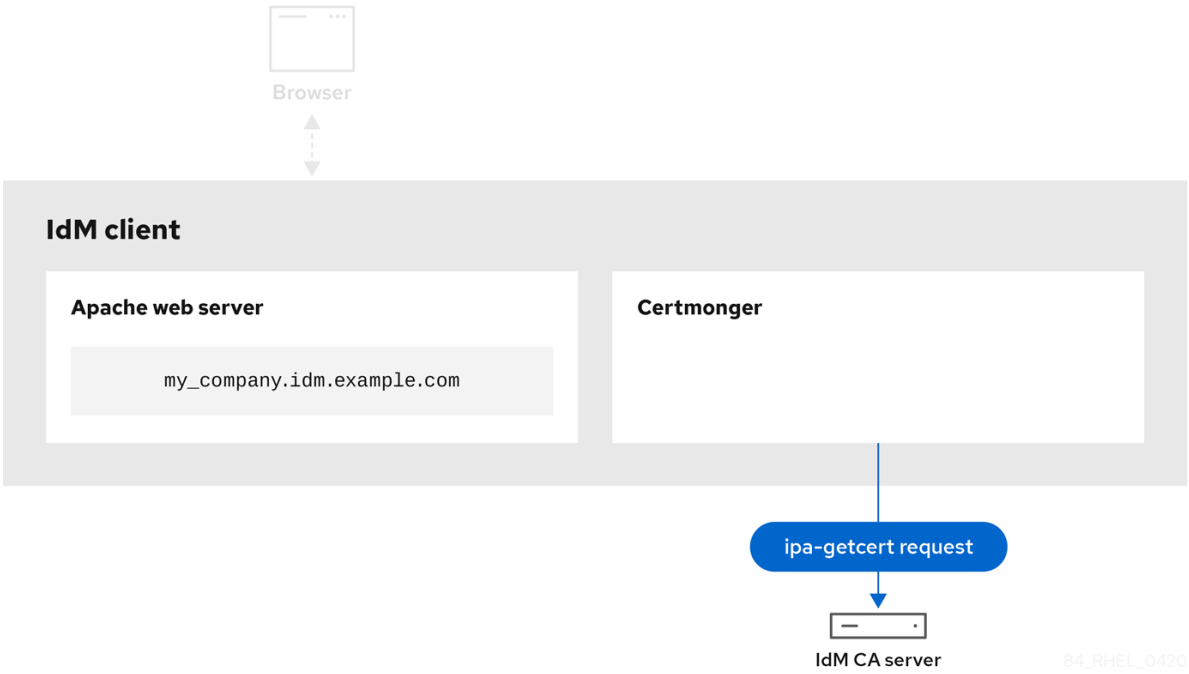
暗号化されていない通信 は、初期状態を示しています。HTTPS 証明書がないと、Web サーバーとブラウザー間の通信は暗号化されません。

図77.1暗号化されていない通信



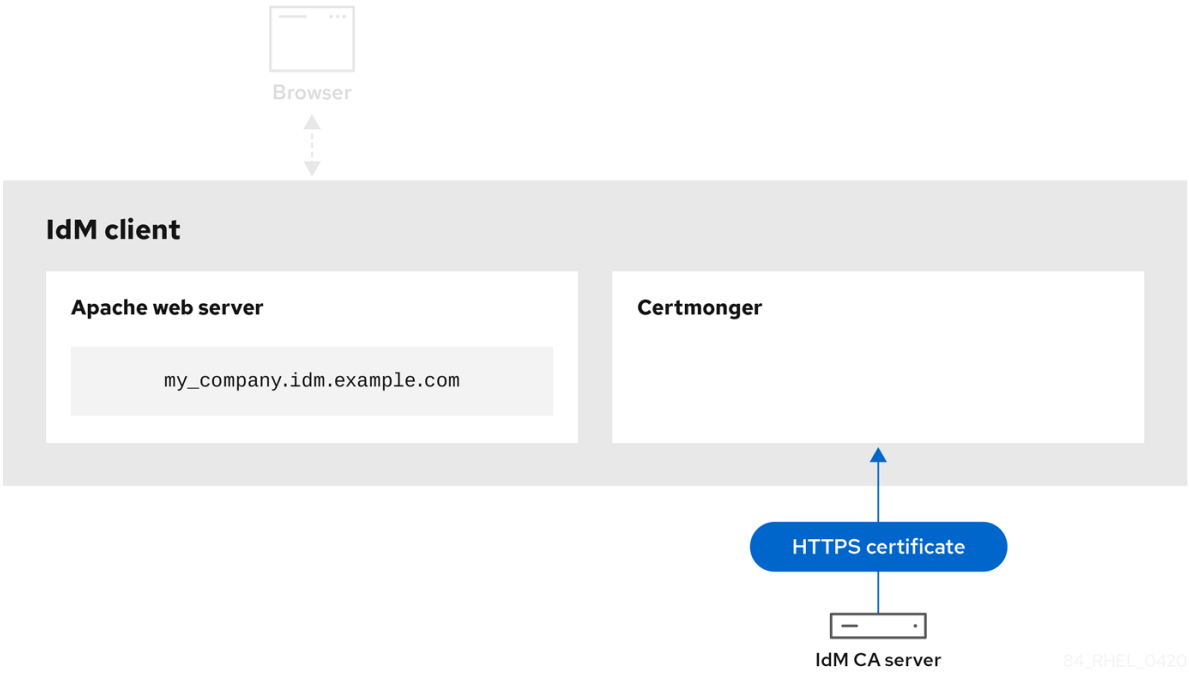
サービス証明書を要求する certmonger は、システム管理者が **certmonger** を使用して Apache Web サーバーの HTTPS 証明書を手動で要求していることを示しています。Web サーバー証明書を要求する場合、certmonger は CA と直接対話しないことに注意してください。IdM 経由でプロキシーが設定されます。

図77.2 サービス証明書を要求する certmonger



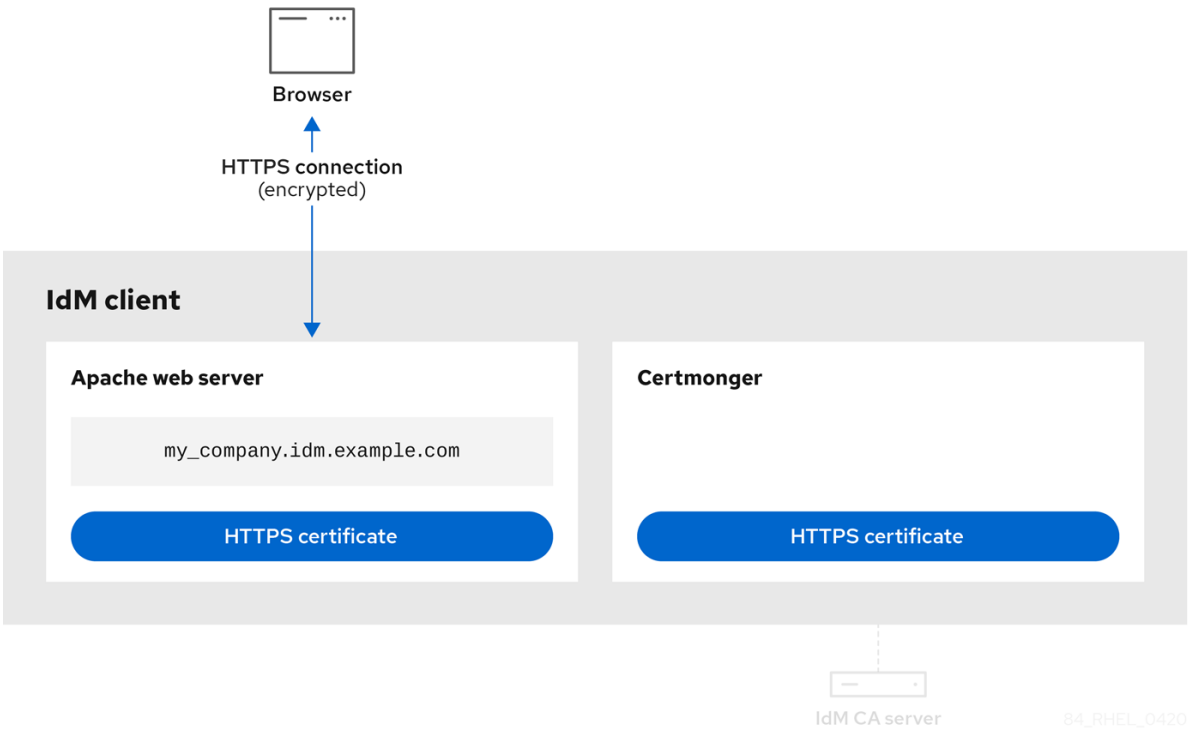
サービス証明書を発行する IdM CA は、Web サーバーで HTTPS 証明書を発行する IdM CA を示しています。

図77.3 サービス証明書を発行する IdM CA



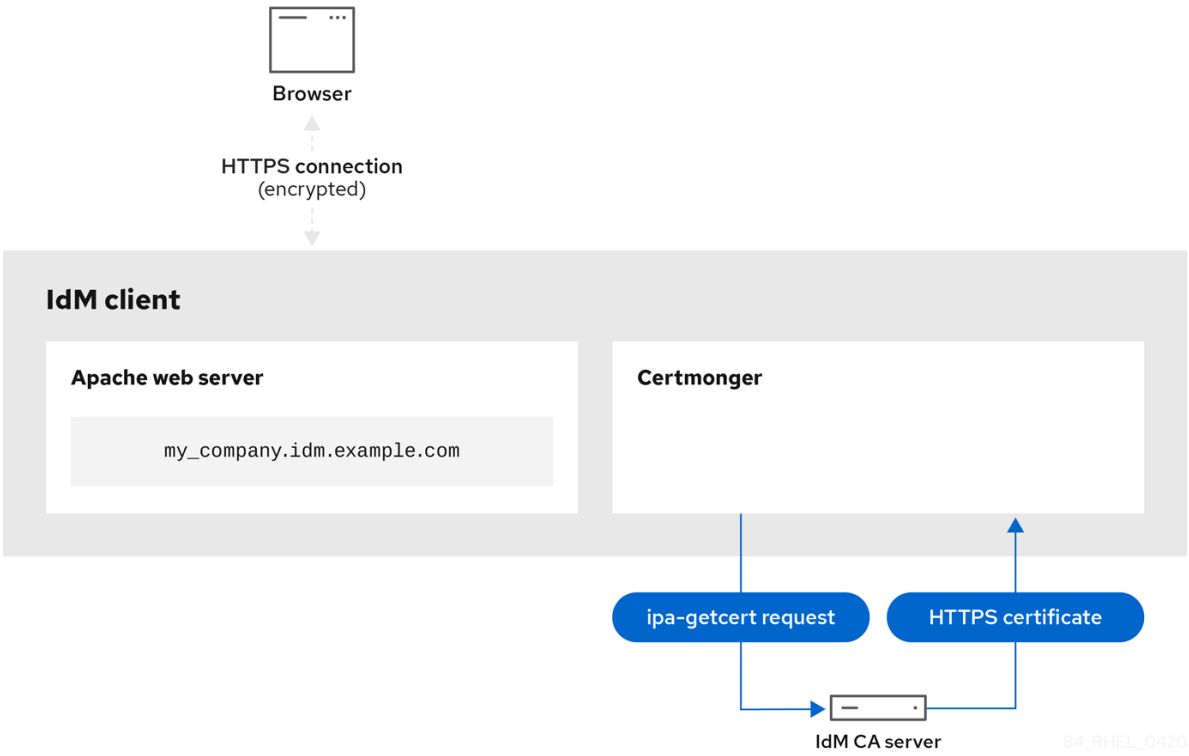
Certmonger によるサービス証明書の適用 は、**certmonger** が HTTPS 証明書を Id M クライアントの適切な場所に配置し、指示された場合は **httpd** サービスを再起動することを示します。その後、Apache サーバーは HTTPS 証明書を使用して、Apache サーバーとブラウザー間のトラフィックを暗号化します。

図77.4 Certmonger によるサービス証明書の適用



古い証明書が有効期限に近づいているときに新しい証明書を要求する certmonger は、証明書の有効期限が切れる前に、**certmonger** が Id MCA からのサービス証明書の更新を自動的に要求していることを示しています。IdM CA は、新しい証明書を発行します。

図77.5 古い証明書が有効期限に近づいているときに新しい証明書を要求する certmonger



77.4. CERTMONGER が追跡する証明書要求の詳細を表示

certmonger サービスは、証明書要求を監視します。証明書要求が正常に署名されると、証明書が作成されます。**certmonger** は、生成される証明書を含む証明書要求を管理します。以下の手順に従って、**certmonger** が管理する特定の証明書要求の詳細を表示します。

手順

- 証明書要求の指定方法が分からない場合は、特定の証明書要求のみの詳細をリスト表示します。たとえば、次を指定できます。
  - リクエスト ID
  - 証明書の場所
  - 証明書のニックネーム
    たとえば、要求 ID が 20190408143846 である証明書の詳細を表示するには、**-v** オプションを使用して、証明書のリクエストが失敗した場合にエラーの詳細をすべて表示します。

```
# getcert list -i 20190408143846 -v
Number of certificates and requests being tracked: 16.
Request ID '20190408143846':
 status: MONITORING
 stuck: no
 key pair storage: type=NSSDB,location='/etc/dirsrv/slapd-IDM-EXAMPLE-
```

```
COM',nickname='Server-Cert',token='NSS Certificate DB',pinfile='/etc/dirsrv/slapd-IDM-EXAMPLE-COM/pwdfile.txt'
 certificate: type=NSSDB,location='/etc/dirsrv/slapd-IDM-EXAMPLE-COM',nickname='Server-Cert',token='NSS Certificate DB'
 CA: IPA
 issuer: CN=Certificate Authority,O=IDM.EXAMPLE.COM
 subject: CN=r8server.idm.example.com,O=IDM.EXAMPLE.COM
 expires: 2021-04-08 16:38:47 CEST
 dns: r8server.idm.example.com
 principal name: ldap/server.idm.example.com@IDM.EXAMPLE.COM
 key usage: digitalSignature,nonRepudiation,keyEncipherment,dataEncipherment
 eku: id-kp-serverAuth,id-kp-clientAuth
 pre-save command:
 post-save command: /usr/libexec/ipa/certmonger/restart_dirsrv IDM-EXAMPLE-COM
 track: yes
 auto-renew: yes
```

この出力では、証明書に関する情報の一部が表示されます。以下に例を示します。

- 証明書の場所 - 上記の例では、**/etc/dirsrv/slapd-IDM-EXAMPLE-COM** ディレクトリーの NSS データベースです。
- 証明書のニックネーム - 上記の例では **Server-Cert** になります。
- ピンを保存しているファイル - 上記の例では **/etc/dirsrv/slapd-IDM-EXAMPLE-COM/pwdfile.txt** になります。
- 証明書の更新に使用される認証局 (CA) - 上記の例では **IPA** CA です。
- 有効期限 - 上記の例では **2021-04-08 16:38:47 CEST** になります。
- 証明書のステータス - 上記の例の **MONITORING** 状態は、証明書が有効であり、追跡されていることを意味します。
- post-save コマンド - 上記の例では、**LDAP** サービスの再起動です。

- 証明書要求の指定方法が分からない場合は、**certmonger** が監視または取得しようとしているすべての証明書の詳細をリスト表示します。

```
# getcert list
```

関連情報

- **getcert list** man ページを参照してください。

77.5. 証明書追跡の開始および停止

以下の手順に従って、**getcert stop-tracking** コマンドおよび **getcert start-tracking** コマンドを使用して証明書を監視します。この 2 つのコマンドは、**certmonger** サービスで利用できます。証明書追跡を有効にすると、Identity Management (IdM) 認証局 (CA) が発行する証明書を、別の IdM クライアントのマシンにインポートすると特に便利です。証明書の追跡を有効にすることは、次のプロビジョニングシナリオの最終ステップでもあります。

1. IdM サーバーでは、存在していないシステムの証明書を作成する。

2. 新しいシステムを作成する。
3. 新しいシステムを IdM クライアントとして登録する。
4. IdM クライアントの IdM サーバーから、証明書および鍵をインポートする。
5. **certmonger** を使用して証明書の追跡を開始し、有効期限が切れる時に証明書を更新するようにする。

手順

- 要求 ID が 20190408143846 の証明書の監視を無効にするには、次のコマンドを実行します。

```
# getcert stop-tracking -i 20190408143846
```

その他のオプションは、**getcert stop-tracking** man ページを参照してください。

- **/tmp/some_cert.crt** ファイルに保存されている証明書の監視を有効にするには、次のコマンドを実行します。秘密鍵が **/tmp/some_key.key** ファイルに保存されます。

```
# getcert start-tracking -c IPA -f /tmp/some_cert.crt -k /tmp/some_key.key
```

certmonger は、証明書を発行した CA タイプを自動的に特定できません。そのため、IdM CA が証明書を発行した場合は、**getcert start-tracking** コマンドに **IPA** 値を付けて **-c** オプションを追加します。**-c** オプションを追加しないと、**certmonger** が NEED_CA 状態になります。

その他のオプションは、**getcert start-tracking** man ページを参照してください。

注記

この 2 つのコマンドは証明書を操作しません。たとえば、**getcert stop-tracking** は、証明書を削除しなかったり、NSS データベースまたはファイルシステムから削除したりせず、監視する証明書のリストから証明書を削除します。同様に、**getcert start-tracking** は、監視された証明書のリストに証明書のみを追加します。

77.6. 証明書を手動で更新

証明書が失効日近くになると、**certmonger** デーモンは、認証局 (CA) ヘルパーを使用して更新コマンドを自動的に発行し、更新された証明書を取得し、以前の証明書を新しい証明書に置き換えます。

getcert resubmit コマンドを使用して、事前に証明書を手動で更新することもできます。これにより、SAN (Subject Alternative Name) を追加したりすると、証明書に含まれる情報を更新できます。

次の手順に従って、証明書を手動で更新します。

手順

- リクエスト ID が 20190408143846 の証明書を更新するには、次のコマンドを実行します。

```
# getcert resubmit -i 20190408143846
```

特定の証明書の要求 ID を取得するには、**getcert list** コマンドを使用します。詳細は、**getcert list** man ページを参照してください。

77.7. CERTMONGER が CA レプリカでの IDM 証明書の追跡を再開

この手順は、証明書の追跡が中断された後、**certmonger** が統合認証局で Identity Management (IdM) デプロイメントに不可欠な IdM システム証明書の追跡を再開する方法を説明します。中断は、システム証明書の更新中に IdM ホストが IdM から登録解除されたか、レプリケーショントポロジーが正しく機能しないことが原因である可能性があります。この手順では、**certmonger** が IdM サービス証明書 (**HTTP**、**LDAP**、**PKINIT**) の追跡を再開する方法も説明します。

前提条件

- システム証明書の追跡を再開するホストは、IdM 認証局 (CA) でもある IdM サーバーですが、IdM CA 更新サーバーではありません。

手順

1. サブシステムの CA 証明書の PIN を取得します。

```
# grep 'internal=' /var/lib/pki/pki-tomcat/conf/password.conf
```

2. サブシステムの CA 証明書に追跡を追加します。次のコマンドの **[internal PIN]** を、直前の手順で取得した PIN に置き換えます。

```
# getcert start-tracking -d /etc/pki/pki-tomcat/alias -n "caSigningCert cert-pki-ca" -c
'dogtag-ipa-ca-renew-agent' -P [internal PIN] -B
/usr/libexec/ipa/certmonger/stop_pkicad -C '/usr/libexec/ipa/certmonger/renew_ca_cert
"caSigningCert cert-pki-ca"' -T caCACert

# getcert start-tracking -d /etc/pki/pki-tomcat/alias -n "auditSigningCert cert-pki-ca" -c
'dogtag-ipa-ca-renew-agent' -P [internal PIN] -B
/usr/libexec/ipa/certmonger/stop_pkicad -C '/usr/libexec/ipa/certmonger/renew_ca_cert
"auditSigningCert cert-pki-ca"' -T caSignedLogCert

# getcert start-tracking -d /etc/pki/pki-tomcat/alias -n "ocspSigningCert cert-pki-ca" -c
'dogtag-ipa-ca-renew-agent' -P [internal PIN] -B
/usr/libexec/ipa/certmonger/stop_pkicad -C '/usr/libexec/ipa/certmonger/renew_ca_cert
"ocspSigningCert cert-pki-ca"' -T caOCSPCert

# getcert start-tracking -d /etc/pki/pki-tomcat/alias -n "subsystemCert cert-pki-ca" -c
'dogtag-ipa-ca-renew-agent' -P [internal PIN] -B
/usr/libexec/ipa/certmonger/stop_pkicad -C '/usr/libexec/ipa/certmonger/renew_ca_cert
"subsystemCert cert-pki-ca"' -T caSubsystemCert

# getcert start-tracking -d /etc/pki/pki-tomcat/alias -n "Server-Cert cert-pki-ca" -c
'dogtag-ipa-ca-renew-agent' -P [internal PIN] -B
/usr/libexec/ipa/certmonger/stop_pkicad -C '/usr/libexec/ipa/certmonger/renew_ca_cert
"Server-Cert cert-pki-ca"' -T caServerCert
```

3. 残りの IdM 証明書、**HTTP** 証明書、**LDAP** 証明書、**IPA 更新エージェント** 証明書、および **PKINIT** 証明書の追跡を追加します。

```
# getcert start-tracking -f /var/lib/ipa/certs/httpd.crt -k /var/lib/ipa/private/httpd.key -p
/var/lib/ipa/passwds/idm.example.com-443-RSA -c IPA -C
/usr/libexec/ipa/certmonger/restart_httpd -T calPAserviceCert

# getcert start-tracking -d /etc/dirsrv/slapd-IDM-EXAMPLE-COM -n "Server-Cert" -c IPA
```

```
-p /etc/dirsrv/slapd-IDM-EXAMPLE-COM/pwdfile.txt -C
'/usr/libexec/ipa/certmonger/restart_dirsrv "IDM-EXAMPLE-COM"' -T calPAserviceCert

# getcert start-tracking -f /var/lib/ipa/ra-agent.pem -k /var/lib/ipa/ra-agent.key -c
dogtag-ipa-ca-renew-agent -B /usr/libexec/ipa/certmonger/renew_ra_cert_pre -C
/usr/libexec/ipa/certmonger/renew_ra_cert -T caSubsystemCert

# getcert start-tracking -f /var/kerberos/krb5kdc/kdc.crt -k
/var/kerberos/krb5kdc/kdc.key -c dogtag-ipa-ca-renew-agent -B
/usr/libexec/ipa/certmonger/renew_ra_cert_pre -C
/usr/libexec/ipa/certmonger/renew_kdc_cert -T KDCs_PKINIT_Certs
```

4. **certmonger** を再起動します。

```
# systemctl restart certmonger
```

5. **certmonger** の起動後 1 分待ってから、新しい証明書の状況を確認します。

```
# getcert list
```

関連情報

- IdM システム証明書の有効期限が切れている場合は、KCS (Knowledge Centered Support) ソリューション を参照して、CA 更新サーバー、および CRL パブリッシャーサーバーでもある IdM CA サーバーで IdM システム証明書を手動で更新します。次に、How do I manually renew Identity Management (IPA) certificates on RHEL7 after they have expired? (Replica IPA Server) で説明されている手順に従って、トポロジーにあるその他のすべての CA サーバーで IdM システム証明書を手動で更新します。

77.8. CERTMONGER での SCEP の使用

Simple Certificate Enrollment Protocol (SCEP) は、さまざまなデバイスやオペレーティングシステムで使用できる証明書管理プロトコルです。環境内で SCEP サーバーを外部認証局 (CA) として使用している場合、**certmonger** を使用して Identity Management (IdM) クライアントの証明書を取得できます。

77.8.1. SCEP の概要

Simple Certificate Enrollment Protocol (SCEP) は、さまざまなデバイスやオペレーティングシステムで使用できる証明書管理プロトコルです。SCEP サーバーを外部認証局 (CA) として使用できます。

Identity Management (IdM) クライアントを設定して、CA SCEP サービスから直接 HTTP 経由で証明書を要求および取得できます。このプロセスは、通常、限られた時間でのみ有効な共有シークレットで保護されます。

クライアント側で、SCEP は以下のコンポーネントを提供する必要があります。

- SCEP URL: CA SCEP インターフェイスの URL。
- SCEP 共有シークレット: CA と SCEP クライアントの間で共有される、証明書を取得するために使用される **challengePassword** PIN。

その後、クライアントは SCEP 経由で CA 証明書チェーンを取得し、CA に証明書署名要求を送信します。

certmonger で SCEP を設定する場合は、発行した証明書パラメーターを指定する新しい CA 設定プロファイルを作成します。

77.8.2. SCEP 経由での IdM CA 署名証明書の要求

以下の例では、**SCEP_example** SCEP CA 設定を **certmonger** に追加し、IdM クライアント **client.idm.example.com** で新しい証明書を要求します。**certmonger** は、NSS 証明書データベース形式と、OpenSSL などのファイルベース (PEM) 形式の両方をサポートしています。

前提条件

- SCEP URL を知っている。
- **challengePassword** PIN 共有シークレットがある。

手順

1. CA 設定を **certmonger** に追加します。

```
[root@client.idm.example.com ~]# getcert add-scep-ca -c SCEP_example -u SCEP_URL
```

   - **-c**: CA 設定に必要なニックネーム。後で同じ値を、他の **getcert** コマンドと合わせて使用できます。
   - **-u**: サーバーの SCEP インターフェイスへの URL。

重要

HTTPS URL を使用する場合は、**-R** オプションを使用して SCEP サーバー CA 証明書の PEM 形式のコピーの場所も指定する必要があります。

2. CA 設定が正常に追加されたことを確認します。

```
[root@client.idm.example.com ~]# getcert list-cas -c SCEP_example
CA 'SCEP_example':
        is-default: no
        ca-type: EXTERNAL
        helper-location: /usr/libexec/certmonger/scep-submit -u
http://SCEP_server_enrollment_interface_URL
        SCEP CA certificate thumbprint (MD5): A67C2D4B 771AC186 FCCA654A 5E55AAF7
        SCEP CA certificate thumbprint (SHA1): FBFF096C 6455E8E9 BD55F4A5 5787C43F
1F512279
```

   設定が正常に追加された場合、certmonger はリモート CA から CA チェーンを取得します。CA チェーンは、コマンド出力でサムプリントとして表示されます。暗号化されていない HTTP でサーバーにアクセスすると、中間者攻撃を防ぐため、サムプリントを SCEP サーバーに表示されるものと手動で比較します。

3. CA から証明書を要求します。
   - NSS を使用している場合:

```
[root@client.idm.example.com ~]# getcert request -I Example_Task -c SCEP_example
-d /etc/pki/nssdb -n ExampleCert -N cn="client.idm.example.com" -L one-time_PIN -D
client.idm.example.com
```

オプションを使用して、証明書要求の以下のパラメーターを指定できます。

- **-I** : **(オプション)** タスクの名前: リクエストの追跡 ID。後で **getcert list** コマンドで同じ値を使用できます。
- **-c**: 要求を送信する CA 設定。
- **-d**: 証明書およびキーを保存する NSS データベースを備えたディレクトリー。
- **-n**: NSS データベースで使用される証明書のニックネーム。
- **-N**: CSR のサブジェクト名。
- **-L** : CA が発行する期限付きの 1 回限りの **challengePassword** PIN。
- **-d**: 証明書のサブジェクト代替名。通常はホスト名と同じです。

- OpenSSL を使用している場合は、以下を行います。

```
[root@client.idm.example.com ~]# getcert request -I Example_Task -c SCEP_example
-f /etc/pki/tls/certs/server.crt -k /etc/pki/tls/private/private.key -N
cn="client.idm.example.com" -L one-time_PIN -D client.idm.example.com
```

オプションを使用して、証明書要求の以下のパラメーターを指定できます。

- **-I** : **(オプション)** タスクの名前: リクエストの追跡 ID。後で **getcert list** コマンドで同じ値を使用できます。
- **-c**: 要求を送信する CA 設定。
- **-f**: 証明書へのストレージパス。
- **-k**: キーへのストレージパス。
- **-N**: CSR のサブジェクト名。
- **-L** : CA が発行する期限付きの 1 回限りの **challengePassword** PIN。
- **-d**: 証明書のサブジェクト代替名。通常はホスト名と同じです。

検証

1. 証明書が発行され、ローカルデータベースに正しく保存されていることを確認します。

- NSS を使用している場合は、以下を入力します。

```
[root@client.idm.example.com ~]# getcert list -I Example_Task
 Request ID 'Example_Task':
        status: MONITORING
        stuck: no
        key pair storage:
type=NSSDB,location='/etc/pki/nssdb',nickname='ExampleCert',token='NSS Certificate
```

```
DB'
        certificate:
type=NSSDB,location='/etc/pki/nssdb',nickname='ExampleCert',token='NSS Certificate
DB'
        signing request thumbprint (MD5): 503A8EDD DE2BE17E 5BAA3A57 D68C9C1B
        signing request thumbprint (SHA1): B411ECE4 D45B883A 75A6F14D 7E3037F1
D53625F4
        CA: IPA
        issuer: CN=Certificate Authority,O=EXAMPLE.COM
        subject: CN=client.idm.example.com,O=EXAMPLE.COM
        expires: 2018-05-06 10:28:06 UTC
        key usage: digitalSignature,keyEncipherment
        eku: iso.org.dod.internet.security.mechanisms.8.2.2
        certificate template/profile: IPSECIntermediateOffline
        pre-save command:
        post-save command:
        track: yes
 auto-renew: yes
```

- OpenSSL を使用している場合は、以下を入力します。

```
[root@client.idm.example.com ~]# getcert list -I Example_Task
Request ID 'Example_Task':
        status: MONITORING
        stuck: no
        key pair storage: type=FILE,location='/etc/pki/tls/private/private.key'
        certificate: type=FILE,location='/etc/pki/tls/certs/server.crt'
        CA: IPA
        issuer: CN=Certificate Authority,O=EXAMPLE.COM
        subject: CN=client.idm.example.com,O=EXAMPLE.COM
        expires: 2018-05-06 10:28:06 UTC
        eku: id-kp-serverAuth,id-kp-clientAuth
        pre-save command:
        post-save command:
        track: yes
        auto-renew: yes
```

ステータス **MONITORING** は、発行した証明書の取得に成功したことを表します。**getcert-list(1)** の man ページには、その他の状態とその意味が記載されています。

関連情報

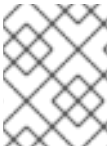
- 証明書を要求する場合の他のオプションは、**getcert-request(1)** の man ページを参照してください。

77.8.3. certmonger による AD SCEP 証明書の自動更新

certmonger が SCEP 証明書の更新要求を送信すると、この要求は既存の証明書の秘密鍵を使用して署名されます。ただし、**certmonger** によってデフォルトで送信される更新要求には、最初に証明書を取得するために使用された **challengePassword** PIN も含まれています。

SCEP サーバーとして機能する Active Directory (AD) Network Device Enrollment Service (NDES) サーバーは、元の **challengePassword** PIN を含む更新要求を自動的に拒否します。そのため、更新に失敗します。

AD での更新を機能させるには、**challengePassword** PIN なしで署名済みの更新要求を送信するように **certmonger** を設定する必要があります。また、更新時にサブジェクト名を比較しないように AD サーバーを設定する必要があります。



注記

challengePassword が含まれるリクエストも拒否する AD 以外の SCEP サーバーが存在する場合があります。この場合は、**certmonger** 設定を変更する必要もあります。

前提条件

- RHEL サーバーは RHEL 8.6 以降を実行している必要がある。

手順

1. AD サーバーで **regedit** を開きます。
2. **HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Cryptography\MSCEP** サブキーに、新しい 32 ビットの REG_DWORD エントリー **DisableRenewalSubjectNameMatch** を追加し、その値を **1** に設定します。
3. **certmonger** が実行されているサーバーで、**/etc/certmonger/certmonger.conf** ファイルを開き、次のセクションを追加します。

```
[scep]
challenge_password_otp = yes
```

4. certmonger を再起動します。

```
# systemctl restart certmonger
```

第78章 RHEL システムロールを使用して証明書を要求する

certificate システムロールを使用すると、証明書を発行および管理できます。

78.1. 証明書 RHEL システムロール

certificate システムロールを使用すると、Ansible Core を使用して TLS および SSL 証明書の発行と更新を管理できます。

ロールは **certmonger** を証明書プロバイダーとして使用し、自己署名証明書の発行と更新、および IdM 統合認証局 (CA) の使用を現時点でサポートしています。

certificate システムロールを含む Ansible Playbook では、次の変数を使用できます。

certificate_wait
タスクが証明書を発行するまで待機するかどうかを指定します。

certificate_requests
発行する各証明書とそのパラメーターを表すには、次のコマンドを実行します。

関連情報

- **/usr/share/ansible/roles/rhel-system-roles.certificate/README.md** ファイル
- **/usr/share/doc/rhel-system-roles/certificate/** ディレクトリー

78.2. CERTIFICATE システムロールを使用した新しい自己署名証明書の要求

certificate システムロールを使用すると、Ansible Core を使用して自己署名証明書を発行できます。

このプロセスは、**certmonger** プロバイダーを使用し、**getcert** コマンドで証明書を要求します。

前提条件

- 制御ノードと管理ノードを準備している
- 管理対象ノードで Playbook を実行できるユーザーとしてコントロールノードにログインしている。
- 管理対象ノードへの接続に使用するアカウントに、そのノードに対する **sudo** 権限がある。

手順

1. 次の内容を含む Playbook ファイル (例: ~/**playbook.yml**) を作成します。

```
---
- hosts: managed-node-01.example.com
  roles:
    - rhel-system-roles.certificate
  vars:
    certificate_requests:
      - name: mycert
        dns: "*.example.com"
        ca: self-sign
```

- **name** パラメーターを希望する証明書の名前 (**mycert** など) に設定します。
- **dns** パラメーターを ***.example.com** などの証明書に含むドメインに設定します。
- **ca** パラメーターを **self-sign** に設定します。

デフォルトでは、**certmonger** は有効期限が切れる前に証明書の更新を自動的に試行します。これは、Ansible Playbook の **auto_renew** パラメーターを **no** に設定すると無効にできます。

2. Playbook の構文を検証します。

```
$ ansible-playbook --syntax-check ~/playbook.yml
```

このコマンドは構文を検証するだけであり、有効だが不適切な設定から保護するものではないことに注意してください。

3. Playbook を実行します。

```
$ ansible-playbook ~/playbook.yml
```

関連情報

- **/usr/share/ansible/roles/rhel-system-roles.certificate/README.md** ファイル
- **/usr/share/doc/rhel-system-roles/certificate/** ディレクトリー

78.3. CERTIFICATE システムロールを使用した IDM CA からの新しい証明書の要求

certificate システムロールを使用すると、統合認証局 (CA) を持つ IdM サーバーを使用しながら、**ansible-core** を使用して証明書を発行できます。したがって、IdM を CA として使用する場合に、複数のシステムの証明書トラストチェーンを効率的かつ一貫して管理できます。

このプロセスは、**certmonger** プロバイダーを使用し、**getcert** コマンドで証明書を要求します。

前提条件

- 制御ノードと管理ノードを準備している
- 管理対象ノードで Playbook を実行できるユーザーとしてコントロールノードにログインしている。
- 管理対象ノードへの接続に使用するアカウントに、そのノードに対する **sudo** 権限がある。

手順

1. 次の内容を含む Playbook ファイル (例: **~/playbook.yml**) を作成します。

```
---
- hosts: managed-node-01.example.com
  roles:
    - rhel-system-roles.certificate
  vars:
    certificate_requests:
```

```
    - name: mycert
      dns: www.example.com
      principal: HTTP/www.example.com@EXAMPLE.COM
      ca: ipa
```

- **name** パラメーターを希望する証明書の名前 (**mycert** など) に設定します。
- **dns** パラメーターをドメインに設定し、証明書に追加します (例: **www.example.com**)。
- **principal** パラメーターを設定し、Kerberos プリンシパルを指定します (例: **HTTP/www.example.com@EXAMPLE.COM**)。
- **ca** パラメーターを **ipa** に設定します。

デフォルトでは、**certmonger** は有効期限が切れる前に証明書の更新を自動的に試行します。これは、Ansible Playbook の **auto_renew** パラメーターを **no** に設定すると無効にできます。

2. Playbook の構文を検証します。

```
$ ansible-playbook --syntax-check ~/playbook.yml
```

このコマンドは構文を検証するだけであり、有効だが不適切な設定から保護するものではないことに注意してください。

3. Playbook を実行します。

```
$ ansible-playbook ~/playbook.yml
```

関連情報

- **/usr/share/ansible/roles/rhel-system-roles.certificate/README.md** ファイル
- **/usr/share/doc/rhel-system-roles/certificate/** ディレクトリー

78.4. CERTIFICATE システムロールを使用して証明書発行前または発行後に実行するコマンドを指定する

certificate ロールでは、Ansible Core を使用して、証明書の発行または更新の前後にコマンドを実行できます。

以下の例では、管理者が **www.example.com** の自己署名証明書を発行または更新する前に **httpd** サービスを停止し、後で再起動します。

前提条件

- 制御ノードと管理ノードを準備している
- 管理対象ノードで Playbook を実行できるユーザーとしてコントロールノードにログインしている。
- 管理対象ノードへの接続に使用するアカウントに、そのノードに対する **sudo** 権限がある。

手順

1. 次の内容を含む Playbook ファイル (例: ~/**playbook.yml**) を作成します。

```
---
- hosts: managed-node-01.example.com
  roles:
    - rhel-system-roles.certificate
  vars:
    certificate_requests:
      - name: mycert
        dns: www.example.com
        ca: self-sign
        run_before: systemctl stop httpd.service
        run_after: systemctl start httpd.service
```

   - **name** パラメーターを希望する証明書の名前 (**mycert** など) に設定します。
   - **dns** パラメーターをドメインに設定し、証明書に追加します (例: **www.example.com**)。
   - **ca** パラメーターを証明書を発行する際に使用する CA に設定します (例: **self-sign**)。
   - この証明書を発行または更新する前に、**run_before** パラメーターを実行するコマンドに設定します (例: **systemctl stop httpd.service**)。
   - この証明書を発行または更新した後に、**run_after** パラメーターを実行するコマンドに設定します (例: **systemctl start httpd.service**)。

   デフォルトでは、**certmonger** は有効期限が切れる前に証明書の更新を自動的に試行します。これは、Ansible Playbook の **auto_renew** パラメーターを **no** に設定すると無効にできます。

2. Playbook の構文を検証します。

```
$ ansible-playbook --syntax-check ~/playbook.yml
```

   このコマンドは構文を検証するだけであり、有効だが不適切な設定から保護するものではないことに注意してください。

3. Playbook を実行します。

```
$ ansible-playbook ~/playbook.yml
```

関連情報

- **/usr/share/ansible/roles/rhel-system-roles.certificate/README.md** ファイル
- **/usr/share/doc/rhel-system-roles/certificate/** ディレクトリー

第79章 証明書のサブセットだけに信頼するアプリケーションを制限する手順

Identity Management (IdM) インストールが統合証明書システム (CS) 認証局 (CA) で設定されている場合は、軽量のサブ CA を作成できます。作成するすべてのサブ CA は、証明書システムのプライマリー CA である **ipa** CA に従属します。

このコンテキストでの **軽量サブ CA** は、**証明書を特定の目的で発行するサブ CA** になります。たとえば、軽量のサブ CA では、仮想プライベートネットワーク (VPN) ゲートウェイや Web ブラウザーなどのサービスを設定して、 **sub-CA A** が発行する証明書のみを受け入れることができます。**sub-CA B** が発行する証明書だけを受け入れるように他のサービスを設定すると、**sub-CA A**、プライマリー CA、**ipa** CA、およびこの 2 つの間にある中間サブ CA が発行する証明書を受け付けないようにすることができます。

クライアントが正しく設定されている場合に、サブ CA の中間証明書を取り消すと、このサブ CA で発行されたすべての証明書 は自動的に無効とみなされます。ルート CA (**ipa**)、または別のサブ CA が直接発行するその他の証明書はすべて有効のままになります。

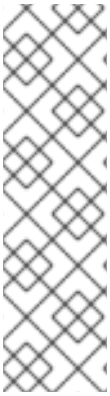
本セクションでは、Apache Web サーバーの例を使用して、アプリケーションが証明書のサブセットだけを信頼するように制限する方法を説明します。本セクションでは、IdM クライアントで実行している Web サーバーを、IdM サブ CA **webserver-ca** が発行する証明書を使用するように制限して、ユーザーに IdM サブ CA **webclient-ca** が発行するユーザー証明書を使用して、Web サーバーへの認証を行うようにするには、以下のセクションを実行します。

実行する必要がある手順は以下の通りです。

1. IdM サブ CA の作成
2. IdM WebUI からのサブ CA 証明書のダウンロード
3. 使用するユーザー、サービス、CA、および証明書プロファイルの正しい組み合わせを指定する CA ACL の作成
4. IdM サブ CA からの IdM クライアントで実行している Web サービスの証明書要求
5. シングルインスタンスの Apache HTTP サーバーの設定
6. Apache HTTP Server への TLS 暗号化の追加
7. Apache HTTP サーバーで対応している TLS プロトコルバージョンの設定
8. Apache HTTP サーバーで対応している暗号の設定
9. Web サーバーで TLS クライアント証明書認証の設定
10. IdM のサブ CA からのユーザー証明書の要求およびクライアントへのエクスポート
11. ブラウザーへのユーザー証明書のインポートおよびサブ CA 証明書を信頼するようなブラウザーの設定

79.1. 軽量サブ CA の管理

本セクションでは、軽量の従属認証局 (サブ CA) を管理する方法を説明します。作成するすべてのサブ CA は、証明書システムのプライマリー CA である **ipa** CA に従属します。サブ CA を無効にしたり、削除したりすることもできます。



注記

- サブ CA を削除すると、そのサブ CA の失効確認は機能しなくなります。**notAfter** の有効期限が近づいていて、そのサブ CA が発行した証明書がなくなった場合に限り、サブ CA を削除してください。
- サブ CA が発行した期限切れではない証明書がまだ存在する場合に限り、サブ CA を無効にしてください。サブ CA により発行された証明書がすべて期限切れになると、そのサブ CA を削除できます。
- IdM CA を無効にしたり、削除したりすることはできません。

サブ CA の管理の詳細は、次を参照してください。

- IdM WebUI からのサブ CA の作成
- IdM WebUI からのサブ CA の削除
- IdM CLI からのサブ CA の作成
- IdM CLI からのサブ CA の無効化
- IdM CLI からのサブ CA の削除

79.1.1. IdM WebUI からのサブ CA の作成

以下の手順に従って、IdM WebUI を使用して **webserver-ca** および **webclient-ca** という名前の新しいサブ CA を作成します。

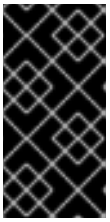
前提条件

- 管理者の認証情報を取得していることを確認している。

手順

1. **Authentication** メニューで、**Certificates** をクリックします。
2. **Certificate Authorities** を選択し、**Add** をクリックします。
3. **webserver-ca** サブ CA の名前を入力します。サブジェクト DN フィールドにサブジェクト DN (例: **CN=WEBSERVER,O=IDM.EXAMPLE.COM**) を入力します。サブジェクト DN は、IdM CA インフラストラクチャー内で一意である必要があります。
4. **webclient-ca** サブ CA の名前を入力します。サブジェクト DN **CN=WEBCLIENT,O=IDM.EXAMPLE.COM** をサブジェクト DN フィールドに入力します。
5. コマンドラインインターフェイスで、**ipa-certupdate** コマンドを実行して、**webserver-ca** サブ CA 証明書および **webclient-ca** サブ CA 証明書の **certmonger** 追跡リクエストを作成します。

```
[root@ipaserver ~]# ipa-certupdate
```



重要

サブ CA の作成後に **ipa-certupdate** コマンドを実行しておかないと、サブ CA 証明書の有効期限が切れたときに、エンドエンティティー証明書の有効期限が切れていなくても、サブ CA が発行したエンドエンティティー証明書は無効と見なされます。

検証

- 新しいサブ CA の署名証明書が IdM データベースに追加されたことを確認します。

```
[root@ipaserver ~]# certutil -d /etc/pki/pki-tomcat/alias/ -L

Certificate Nickname                    Trust Attributes
                                        SSL,S/MIME,JAR/XPI

caSigningCert cert-pki-ca               CTu,Cu,Cu
Server-Cert cert-pki-ca                 u,u,u
auditSigningCert cert-pki-ca            u,u,Pu
caSigningCert cert-pki-ca ba83f324-5e50-4114-b109-acca05d6f1dc u,u,u
ocspSigningCert cert-pki-ca             u,u,u
subsystemCert cert-pki-ca               u,u,u
```

注記

新しいサブ CA 証明書は、証明書システムインスタンスがインストールされているすべてのレプリカに自動的に転送されます。

79.1.2. IdM WebUI からのサブ CA の削除

以下の手順に従って、IdM WebUI で軽量のサブ CA を削除します。

注記

- サブ CA を削除すると、そのサブ CA の失効確認は機能しなくなります。**notAfter** の有効期限が近づいていて、そのサブ CA が発行した証明書がなくなった場合に限り、サブ CA を削除してください。
- サブ CA が発行した期限切れではない証明書がまだ存在する場合に限り、サブ CA を無効にしてください。サブ CA により発行された証明書がすべて期限切れになると、そのサブ CA を削除できます。
- IdM CA を無効にしたり、削除したりすることはできません。

前提条件

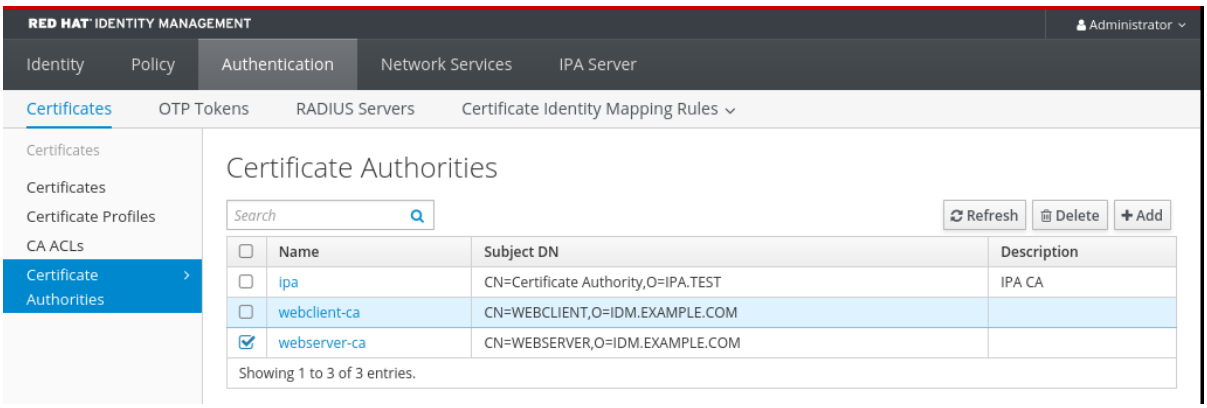
- 管理者の認証情報を取得していることを確認している。
- IdM CLI でサブ CA を無効にしている。IdM CLI からのサブ CA の無効化 参照

手順

1. IdM WebUI で、**Authentication** タブを開き、**Certificates** サブタブを選択します。

2. **Certificate Authorities** を選択します。
3. 削除するサブ CA を選択し、**Delete** をクリックします。

図79.1 IdM Web UI でのサブ CA の削除



4. **Delete** をクリックして確定します。

サブ CA が **Certificate Authorities** のリストから削除されます。

79.1.3. IdM CLI からのサブ CA の作成

以下の手順に従って、IdM CLI を使用して **webserver-ca** および **webclient-ca** という名前の新しいサブ CA を作成します。

前提条件

- 管理者の認証情報を取得していることを確認している。
- CA サーバーである IdM サーバーにログインしている。

手順

1. **ipa ca-add** コマンドを入力し、サブ CA **webserver-ca** の名前とそのサブジェクト識別名 (DN) を指定します。

```
[root@ipaserver ~]# ipa ca-add webserver-ca --
subject="CN=WEBSERVER,O=IDM.EXAMPLE.COM"
-------------------
Created CA "webserver-ca"
-------------------
  Name: webserver-ca
  Authority ID: ba83f324-5e50-4114-b109-acca05d6f1dc
  Subject DN: CN=WEBSERVER,O=IDM.EXAMPLE.COM
  Issuer DN: CN=Certificate Authority,O=IDM.EXAMPLE.COM
```

名前

CA の名前

認証局 ID

CA 用に自動作成される個別 ID。

発行先 DN

サブジェクト識別名 (DN)サブジェクト DN は、IdM CA インフラストラクチャー内で一意である必要があります。

発行者 DN

サブ CA 証明書を発行した親 CA。サブ CA はすべて、IdM のルート CA の子として作成されます。

2. Web クライアントに証明書を発行するサブ CA **webclient-ca** を作成します。

```
[root@ipaserver ~]# ipa ca-add webclient-ca --
subject="CN=WEBCLIENT,O=IDM.EXAMPLE.COM"
-------------------
Created CA "webclient-ca"
-------------------
  Name: webclient-ca
  Authority ID: 8a479f3a-0454-4a4d-8ade-fd3b5a54ab2e
  Subject DN: CN=WEBCLIENT,O=IDM.EXAMPLE.COM
  Issuer DN: CN=Certificate Authority,O=IDM.EXAMPLE.COM
```

3. **ipa-certupdate** コマンドを実行して、**webserver-ca** および **webclient-ca** のサブ CA 証明書の **certmonger** 追跡リクエストを作成します。

```
[root@ipaserver ~]# ipa-certupdate
```

重要

サブ CA の作成後に **ipa-certupdate** コマンドを実行し忘れ、サブ CA 証明書が期限切れになった場合、そのサブ CA が発行したエンドエンティティー証明書は、エンドエンティティー証明書の有効期限が切れていなくても無効と見なされます。

検証手順

- 新しいサブ CA の署名証明書が IdM データベースに追加されたことを確認します。

```
[root@ipaserver ~]# certutil -d /etc/pki/pki-tomcat/alias/ -L

Certificate Nickname                    Trust Attributes
                                        SSL,S/MIME,JAR/XPI

caSigningCert cert-pki-ca               CTu,Cu,Cu
Server-Cert cert-pki-ca                 u,u,u
auditSigningCert cert-pki-ca            u,u,Pu
caSigningCert cert-pki-ca ba83f324-5e50-4114-b109-acca05d6f1dc u,u,u
ocspSigningCert cert-pki-ca             u,u,u
subsystemCert cert-pki-ca               u,u,u
```

注記

新しいサブ CA 証明書は、証明書システムインスタンスがインストールされているすべてのレプリカに自動的に転送されます。

79.1.4. IdM CLI からのサブ CA の無効化

以下の手順に従って、IdM CLI からサブ CA を無効にします。サブ CA が発行した期限切れでない証明書が依然として存在する場合は、証明書を削除しないでください。ただし、無効にすることはできます。サブ CA を削除すると、そのサブ CA に対する取消しの確認が機能しなくなります。

前提条件

- 管理者の認証情報を取得していることを確認している。

手順

1. **ipa ca-find** コマンドを実行して、削除するサブ CA の名前を確認します。

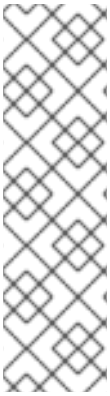
```
[root@ipaserver ~]# ipa ca-find
-------------
3 CAs matched
-------------
  Name: ipa
  Description: IPA CA
  Authority ID: 5195deaf-3b61-4aab-b608-317aff38497c
  Subject DN: CN=Certificate Authority,O=IPA.TEST
  Issuer DN: CN=Certificate Authority,O=IPA.TEST

  Name: webclient-ca
  Authority ID: 605a472c-9c6e-425e-b959-f1955209b092
  Subject DN: CN=WEBCLIENT,O=IDM.EXAMPLE.COM
  Issuer DN: CN=Certificate Authority,O=IPA.TEST

 Name: webserver-ca
  Authority ID: 02d537f9-c178-4433-98ea-53aa92126fc3
  Subject DN: CN=WEBSERVER,O=IDM.EXAMPLE.COM
  Issuer DN: CN=Certificate Authority,O=IPA.TEST
----------------------------
Number of entries returned 3
----------------------------
```

2. **ipa ca-disable** コマンドを実行して、サブ CA (この例では **webserver-ca**) を無効にします。

```
ipa ca-disable webserver-ca
---------------------------
Disabled CA "webserver-ca"
---------------------------
```

79.1.5. IdM CLI からのサブ CA の削除

以下の手順に従って、IdM CLI から軽量のサブ CA を削除します。



注記

- サブ CA を削除すると、そのサブ CA の失効確認は機能しなくなります。**notAfter** の有効期限が近づいていて、そのサブ CA が発行した証明書がなくなった場合に限り、サブ CA を削除してください。
- サブ CA が発行した期限切れではない証明書がまだ存在する場合に限り、サブ CA を無効にしてください。サブ CA により発行された証明書がすべて期限切れになると、そのサブ CA を削除できます。
- IdM CA を無効にしたり、削除したりすることはできません。

前提条件

- 管理者の認証情報を取得していることを確認している。

手順

1. サブ CA および CA のリストを表示するには、**ipa ca-find** コマンドを実行します。

```
# ipa ca-find
-------------
3 CAs matched
-------------
  Name: ipa
  Description: IPA CA
  Authority ID: 5195deaf-3b61-4aab-b608-317aff38497c
  Subject DN: CN=Certificate Authority,O=IPA.TEST
  Issuer DN: CN=Certificate Authority,O=IPA.TEST

  Name: webclient-ca
  Authority ID: 605a472c-9c6e-425e-b959-f1955209b092
  Subject DN: CN=WEBCLIENT,O=IDM.EXAMPLE.COM
  Issuer DN: CN=Certificate Authority,O=IPA.TEST

 Name: webserver-ca
  Authority ID: 02d537f9-c178-4433-98ea-53aa92126fc3
  Subject DN: CN=WEBSERVER,O=IDM.EXAMPLE.COM
  Issuer DN: CN=Certificate Authority,O=IPA.TEST
----------------------------
Number of entries returned 3
----------------------------
```

2. **ipa ca-disable** コマンドを実行して、サブ CA (この例では **webserver-ca**) を無効にします。

```
# ipa ca-disable webserver-ca
--------------------------
Disabled CA "webserver-ca"
--------------------------
```

3. サブ CA (この例では **webserver-ca**) を削除します。

```
# ipa ca-del webserver-ca
-------------------------
Deleted CA "webserver-ca"
```

```
-------------------------
```

検証

- **ipa ca-find** を実行して、CA およびサブ CA のリストを表示します。**webserver-ca** がリストに表示されなくなりました。

```
# ipa ca-find
-------------
2 CAs matched
-------------
  Name: ipa
  Description: IPA CA
  Authority ID: 5195deaf-3b61-4aab-b608-317aff38497c
  Subject DN: CN=Certificate Authority,O=IPA.TEST
  Issuer DN: CN=Certificate Authority,O=IPA.TEST

  Name: webclient-ca
  Authority ID: 605a472c-9c6e-425e-b959-f1955209b092
  Subject DN: CN=WEBCLIENT,O=IDM.EXAMPLE.COM
  Issuer DN: CN=Certificate Authority,O=IPA.TEST
----------------------------
Number of entries returned 2
----------------------------
```

79.2. IDM WEB UI からのサブ CA 証明書のダウンロード

前提条件

- IdM 管理者の認証情報を取得していることを確認している。

手順

1. **Authentication** メニューで、**Certificates** > **Certificates** をクリックします。

   図79.2 証明書リストに含まれるサブ CA 証明書

☐	268173326	CN=WEBSERVER,O=IDM.EXAMPLE.COM	ipa	VALID
☐	268238849	CN=idm_user,O=IDM.EXAMPLE.COM	ipa	VALID

2. サブ CA 証明書のシリアル番号をクリックして、証明書情報ページを開きます。
3. 証明書情報ページで、**Actions** > **Download** をクリックします。
4. CLI で、サブ CA 証明書を **/etc/pki/tls/private/** ディレクトリーに移動します。

```
# mv path/to/the/downloaded/certificate /etc/pki/tls/private/sub-ca.crt
```

79.3. WEB サーバーおよびクライアント認証用の CA ACL の作成

認証局のアクセス制御リスト (CA ACL) ルールは、どのユーザー、サービス、またはホストにどのプロファイルを使用して証明書を発行するかを定義します。CA ACL は、プロファイル、プリンシパル、およびグループを関連付けることで、特定のプロファイルを使用した証明書をプリンシパルまたはグルー

プが要求できるようにします。

たとえば、管理者は CA ACL を使用して、ロンドンオフィス関連のグループに所属するユーザーだけが、ロンドンのオフィスから作業する社員向けのプロファイルを使用するように限定できます。

79.3.1. IdM CLI での CA ACL の表示

IdM デプロイメントで利用可能な認証局のアクセス制御リスト (CA ACL) の一覧と、特定の CA ACL の詳細を表示するには、次の手順に従います。

手順

1. IdM 環境内のすべての CA ACL を表示するには、**ipa caacl-find** コマンドを入力します。

```
$ ipa caacl-find
-----------------
1 CA ACL matched
-----------------
  ACL name: hosts_services_caIPAserviceCert
  Enabled: TRUE
```

2. CA ACL の詳細を表示するには、**ipa caacl-show** コマンドを入力して、CA ACL 名を指定します。たとえば、CA ACL **hosts_services_caIPAserviceCert** の詳細を表示するには、次のコマンドを実行します。

```
$ ipa caacl-show hosts_services_caIPAserviceCert
  ACL name: hosts_services_caIPAserviceCert
  Enabled: TRUE
  Host category: all
  Service category: all
  CAs: ipa
  Profiles: caIPAserviceCert
  Users: admin
```

79.3.2. webserver-ca で発行される証明書を使用して Web クライアントを認証する Web サーバー用の CA ACL の作成

システム管理者が **HTTP/my_company.idm.example.com@IDM.EXAMPLE.COM** サービス証明書の要求時に、サブ CA **webserver-ca** および **caIPAserviceCert** プロファイルを使用するように要求する CA ACL を作成するには、次の手順に従います。ユーザーが別のサブ CA または別のプロファイルの証明書を要求すると、その要求は失敗します。唯一の例外は、別に一致する CA ACL があり、その ACL が有効な場合です。利用可能な CA ACL を表示するには、IdM CLI で CA ACL の表示 を参照してください。

前提条件

- **HTTP/my_company.idm.example.com@IDM.EXAMPLE.COM** サービスが IdM に含まれている。
- IdM 管理者の認証情報を取得していることを確認している。

手順

1. **ipa caacl** コマンドを使用して CA ACL を作成し、その名前を指定します。

```
$ ipa caacl-add TLS_web_server_authentication
----------------------------------------
Added CA ACL "TLS_web_server_authentication"
----------------------------------------
  ACL name: TLS_web_server_authentication
  Enabled: TRUE
```

2. **ipa caacl-mod** コマンドを使用して CA ACL を変更し、CA ACL の説明を指定します。

```
$ ipa caacl-mod TLS_web_server_authentication --desc="CAACL for web servers
authenticating to web clients using certificates issued by webserver-ca"
----------------------------------------------
Modified CA ACL "TLS_web_server_authentication"
----------------------------------------------
  ACL name: TLS_web_server_authentication
  Description: CAACL for web servers authenticating to web clients using certificates issued
by webserver-ca
  Enabled: TRUE
```

3. **webserver-ca** サブ CA を CA ACL に追加します。

```
$ ipa caacl-add-ca TLS_web_server_authentication --ca=webserver-ca
  ACL name: TLS_web_server_authentication
  Description: CAACL for web servers authenticating to web clients using certificates issued
by webserver-ca
  Enabled: TRUE
  CAs: webserver-ca
-------------------------
Number of members added 1
-------------------------
```

4. **ipa caacl-add-service** を使用して、プリンシパルで証明書を要求できるサービスを指定します。

```
$ ipa caacl-add-service TLS_web_server_authentication --
service=HTTP/my_company.idm.example.com@IDM.EXAMPLE.COM
  ACL name: TLS_web_server_authentication
  Description: CAACL for web servers authenticating to web clients using certificates issued
by webserver-ca
  Enabled: TRUE
  CAs: webserver-ca
  Services: HTTP/my_company.idm.example.com@IDM.EXAMPLE.COM
-------------------------
Number of members added 1
-------------------------
```

5. **ipa caacl-add-profile** コマンドを使用して、要求された証明書の証明書プロファイルを指定します。

```
$ ipa caacl-add-profile TLS_web_server_authentication --
certprofiles=caIPAserviceCert
  ACL name: TLS_web_server_authentication
  Description: CAACL for web servers authenticating to web clients using certificates issued
by webserver-ca
```

```
  Enabled: TRUE
  CAs: webserver-ca
  Profiles: caIPAserviceCert
  Services: HTTP/my_company.idm.example.com@IDM.EXAMPLE.COM
-------------------------
Number of members added 1
-------------------------
```

新たに作成した CA ACL は直接使用できます。CA ACL はデフォルトで作成後に有効になります。

注記

CA ACL は、特定のプリンシパルまたはグループから送信される要求への対応が許可されているのは、どの CA またはプロファイルの組み合わせであるかを指定することが目的です。CA ACL は、証明書の検証や信頼には適用されず、発行された証明書の使用方法にも影響はありません。

79.3.3. webclient-ca が発行する証明書を使用して Web サーバーに対して認証するユーザーの Web ブラウザー用に CA ACL を作成する手順

システム管理者が証明書の要求時に サブ CA **webclient-ca** と **IECUserRoles** プロファイルを使用する必要がある CA ACL を作成するには、次の手順に従います。ユーザーが別のサブ CA または別のプロファイルの証明書を要求すると、その要求は失敗します。唯一の例外は、別に一致する CA ACL があり、その ACL が有効な場合です。利用可能な CA ACL を表示するには、IdM CLI で CA ACL の表示 を参照してください。

前提条件

- IdM 管理者の認証情報を取得していることを確認している。

手順

1. **ipa caacl** コマンドを使用して CA ACL を作成し、その名前を指定します。

```
$ ipa caacl-add TLS_web_client_authentication
--------------------------------------------
Added CA ACL "TLS_web_client_authentication"
--------------------------------------------
  ACL name: TLS_web_client_authentication
  Enabled: TRUE
```

2. **ipa caacl-mod** コマンドを使用して CA ACL を変更し、CA ACL の説明を指定します。

```
$ ipa caacl-mod TLS_web_client_authentication --desc="CAACL for user web
browsers authenticating to web servers using certificates issued by webclient-ca"
--------------------------------------------
Modified CA ACL "TLS_web_client_authentication"
--------------------------------------------
  ACL name: TLS_web_client_authentication
  Description: CAACL for user web browsers authenticating to web servers using certificates
issued by webclient-ca
  Enabled: TRUE
```

3. **webclient-ca** サブ CA を CA ACL に追加します。

```
$ ipa caacl-add-ca TLS_web_client_authentication --ca=webclient-ca
  ACL name: TLS_web_client_authentication
  Description: CAACL for user web browsers authenticating to web servers using certificates issued by webclient-ca
  Enabled: TRUE
  CAs: webclient-ca
-------------------------
Number of members added 1
-------------------------
```

4. **ipa caacl-add-profile** コマンドを使用して、要求された証明書の証明書プロファイルを指定します。

```
$ ipa caacl-add-profile TLS_web_client_authentication --certprofiles=IECUserRoles
  ACL name: TLS_web_client_authentication
  Description: CAACL for user web browsers authenticating to web servers using certificates issued by webclient-ca
  Enabled: TRUE
  CAs: webclient-ca
  Profiles: IECUserRoles
-------------------------
Number of members added 1
-------------------------
```

5. **ipa caacl-mod** コマンドを使用して CA ACL を変更し、CA ACL がすべての IdM ユーザーに適用されるように指定します。

```
$ ipa caacl-mod TLS_web_client_authentication --usercat=all
-----------------------------------------------
Modified CA ACL "TLS_web_client_authentication"
-----------------------------------------------
  ACL name: TLS_web_client_authentication
  Description: CAACL for user web browsers authenticating to web servers using certificates issued by webclient-ca
  Enabled: TRUE
  User category: all
  CAs: webclient-ca
  Profiles: IECUserRoles
```

新たに作成した CA ACL は直接使用できます。CA ACL はデフォルトで作成後に有効になります。

注記

CA ACL は、特定のプリンシパルまたはグループから送信される要求への対応が許可されているのは、どの CA またはプロファイルの組み合わせであるかを指定することが目的です。CA ACL は、証明書の検証や信頼には適用されず、発行された証明書の使用方法にも影響はありません。

79.4. CERTMONGER を使用したサービスの IDM 証明書の取得

ブラウザーと、IdM クライアントで実行している Web サービスとの間の通信が安全で暗号化されてい

ることを確認するには、TLS 証明書を使用します。サブ CA **webserver-ca** が発行する証明書を信頼し、その他の IdM サブ CA は信頼しないように制限して Web ブラウザーを設定する場合にはサブ CA **webserver-ca** から Web サービスの TLS 証明書を取得します。

以下の手順に従って、**certmonger** を使用して、IdM クライアントで実行しているサービス (**HTTP/my_company.idm.example.com@IDM.EXAMPLE.COM**) の IdM 証明書を取得します。

certmonger 使用して証明書を自動的に要求するということは、**certmonger** 更新の期限が切れたときに証明書を管理および更新することを意味します。

certmonger がサービス証明書を要求したときの動作の視覚的な表現については、サービス証明書を要求する certmonger の通信フロー を参照してください。

前提条件

- Web サーバーが、IdM クライアントとして登録されている。
- 手順を実行している IdM クライアントへのルートアクセス権限がある。
- 証明書を要求しているサービスは、前もって IdM に用意する必要はない。

手順

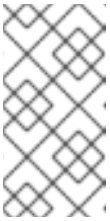
1. **HTTP** サービスが稼働している IdM クライアント **my_company.idm.example.com** で、以下を指定する **HTTP/my_company.idm.example.com@IDM.EXAMPLE.COM** プリンシパルに対応するサービスの証明書を要求します。
   - 証明書は、ローカルの **/etc/pki/tls/certs/httpd.pem** ファイルに保存されます。
   - 秘密鍵は、ローカルの **/etc/pki/tls/private/httpd.key** ファイルに保存されます。
   - サブ CA **webserver-ca** は、発行元の認証局になります。
   - **SubjectAltName** の extensionRequest が、 **my_company.idm.example.com** の DNS 名の署名要求に追加されます。

     ```
     # ipa-getcert request -K HTTP/my_company.idm.example.com -k
     /etc/pki/tls/private/httpd.key -f /etc/pki/tls/certs/httpd.pem -g 2048 -D
     my_company.idm.example.com -X webserver-ca -C "systemctl restart httpd"
     New signing request "20190604065735" added.
     ```

     上記のコマンドでは、以下のようになります。

     - **ipa-getcert request** コマンドは、証明書が IdM CA から取得することを示しています。**ipa-getcert request** コマンドは、**getcert request -c IPA** のショートカットです。
     - **-g** オプションは、生成先のキーのサイズ (設定されていない場合) を指定します。
     - **-D** オプションは、要求に追加する DNS 値 **SubjectAltName** を指定します。
     - **-X** オプションは、証明書の発行者が、**ipa** ではなく **webserver-ca** でなければならないことを指定します。
     - **-C** オプションは、証明書の取得後に **httpd** サービスを再起動するように **certmonger** に指示します。

- 特定のプロファイルで証明書を発行するように指定する場合は、**-T** オプションを使用します。



注記

RHEL 8 は、RHEL 7 で使用されるものとは異なる SSL モジュール (Apache) を使用します。SSL モジュールは、NSS ではなく OpenSSL に依存しています。このため、RHEL 8 では、NSS データベースを使用して **HTTPS** 証明書と秘密鍵を保存することができません。

2. 必要に応じて、リクエストの状況を確認するには、次のコマンドを実行します。

```
# ipa-getcert list -f /etc/pki/tls/certs/httpd.pem
Number of certificates and requests being tracked: 3.
Request ID '20190604065735':
    status: MONITORING
    stuck: no
    key pair storage: type=FILE,location='/etc/pki/tls/private/httpd.key'
    certificate: type=FILE,location='/etc/pki/tls/certs/httpd.crt'
    CA: IPA
    issuer: CN=WEBSERVER,O=IDM.EXAMPLE.COM

[...]
```

この出力は、要求が **MONITORING** 状況であることを表しています。これは、証明書が取得されていることを示しています。キーペアと証明書の場所は、要求された場所です。

79.5. サービス証明書を要求する CERTMONGER の通信フロー

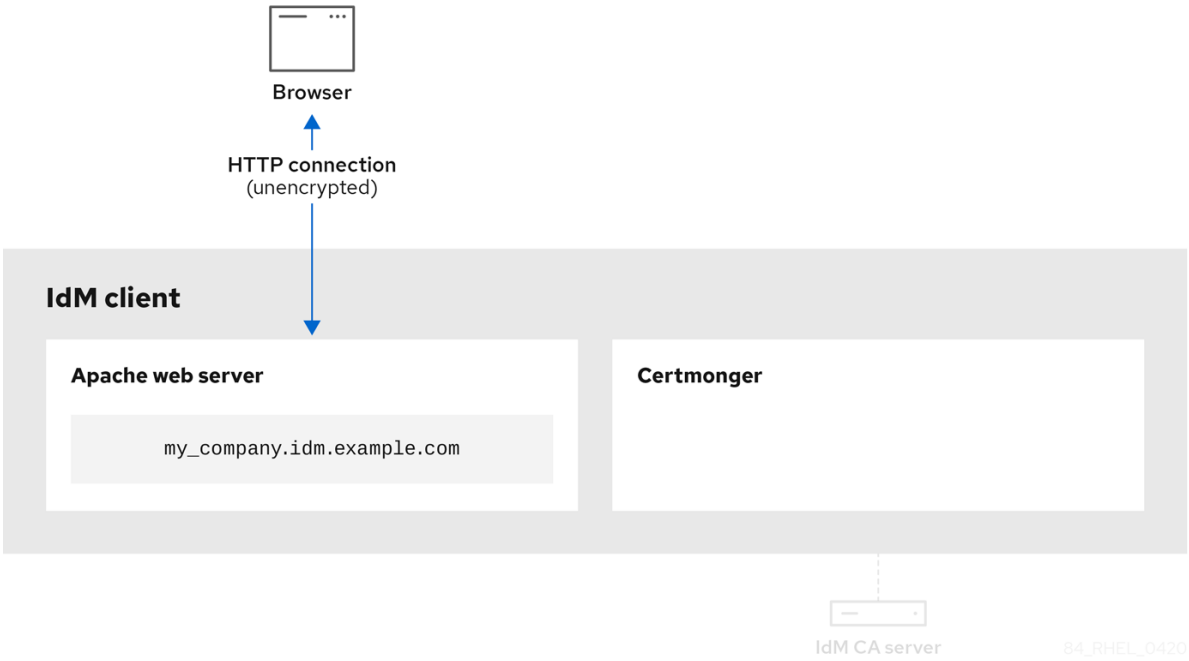
これらの図では、**certmonger** が Identity Management (IdM) 認証局 (CA) サーバーからサービス証明書を要求したときに何が起こるかを段階をおって紹介しています。シーケンスは次の図で設定されています。

- 暗号化されていない通信
- サービス証明書を要求する certmonger
- サービス証明書を発行する IdM CA
- Certmonger によるサービス証明書の適用
- 古い証明書が有効期限に近づいているときに新しい証明書を要求する certmonger

この図では、サブ CA **webserver-ca** は汎用 **IdM CA サーバー** で表現されます。

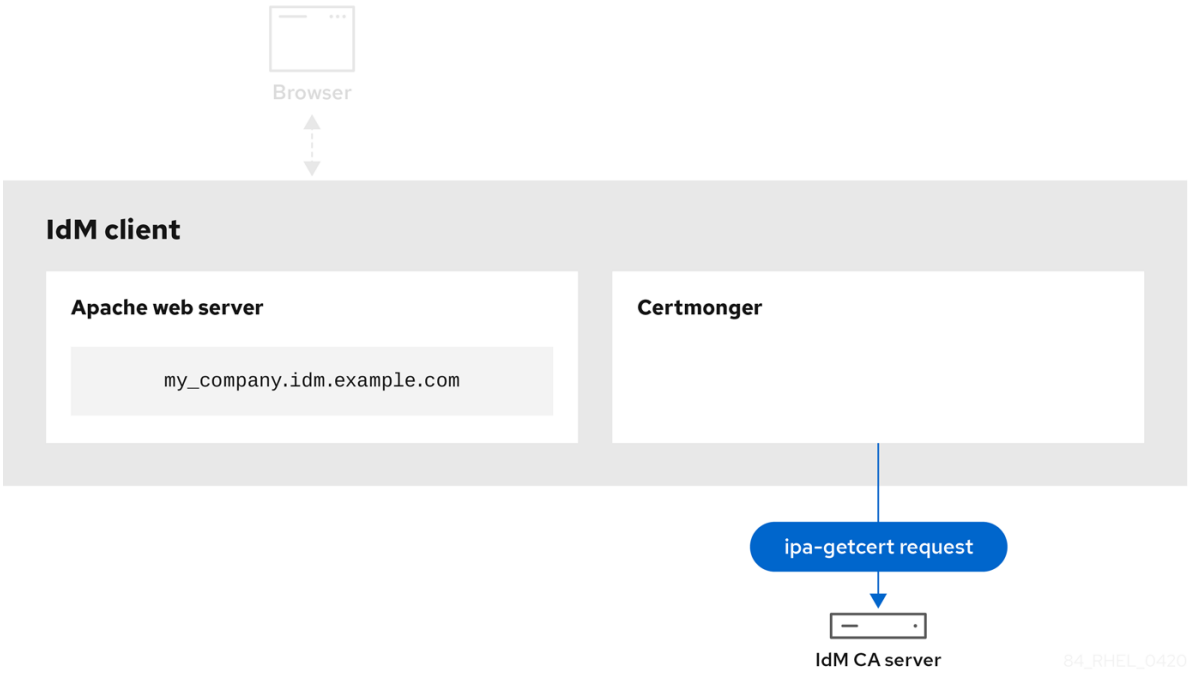
暗号化されていない通信 は、初期状態を示しています。HTTPS 証明書がないと、Web サーバーとブラウザー間の通信は暗号化されません。

図79.3 暗号化されていない通信



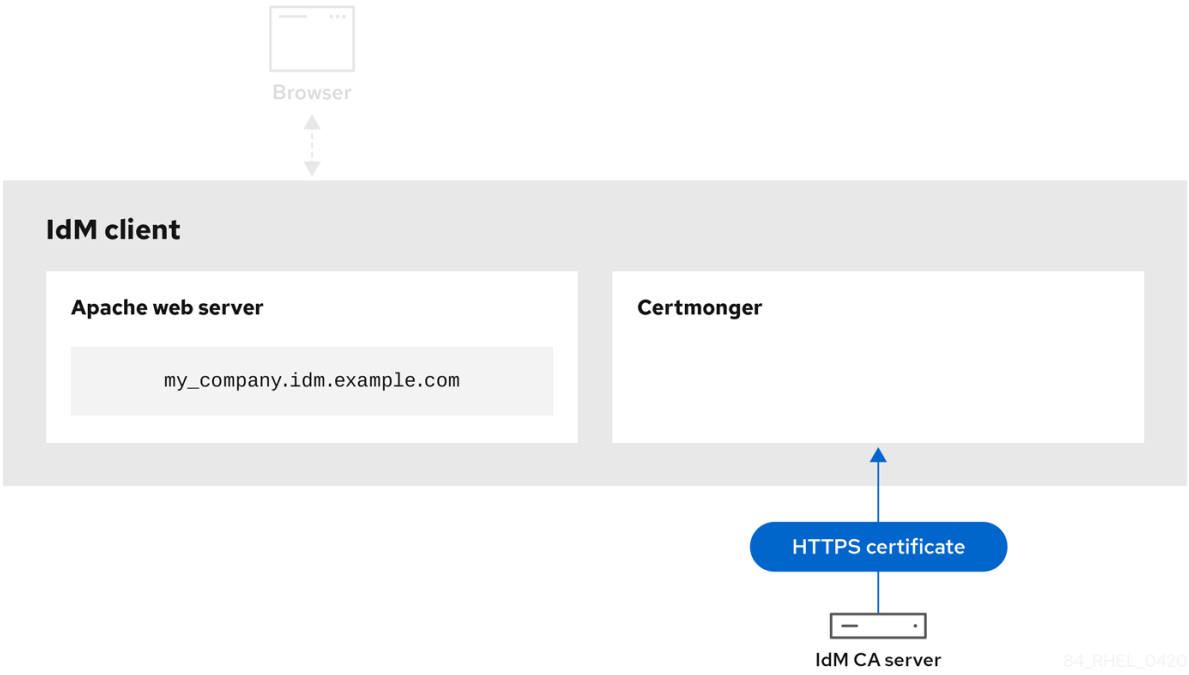
サービス証明書を要求する certmonger は、システム管理者が **certmonger** を使用して Apache Web サーバーの HTTPS 証明書を手動で要求していることを示しています。Web サーバー証明書を要求する場合、certmonger は CA と直接対話しないことに注意してください。IdM 経由でプロキシーが設定されます。

図79.4 サービス証明書を要求する certmonger



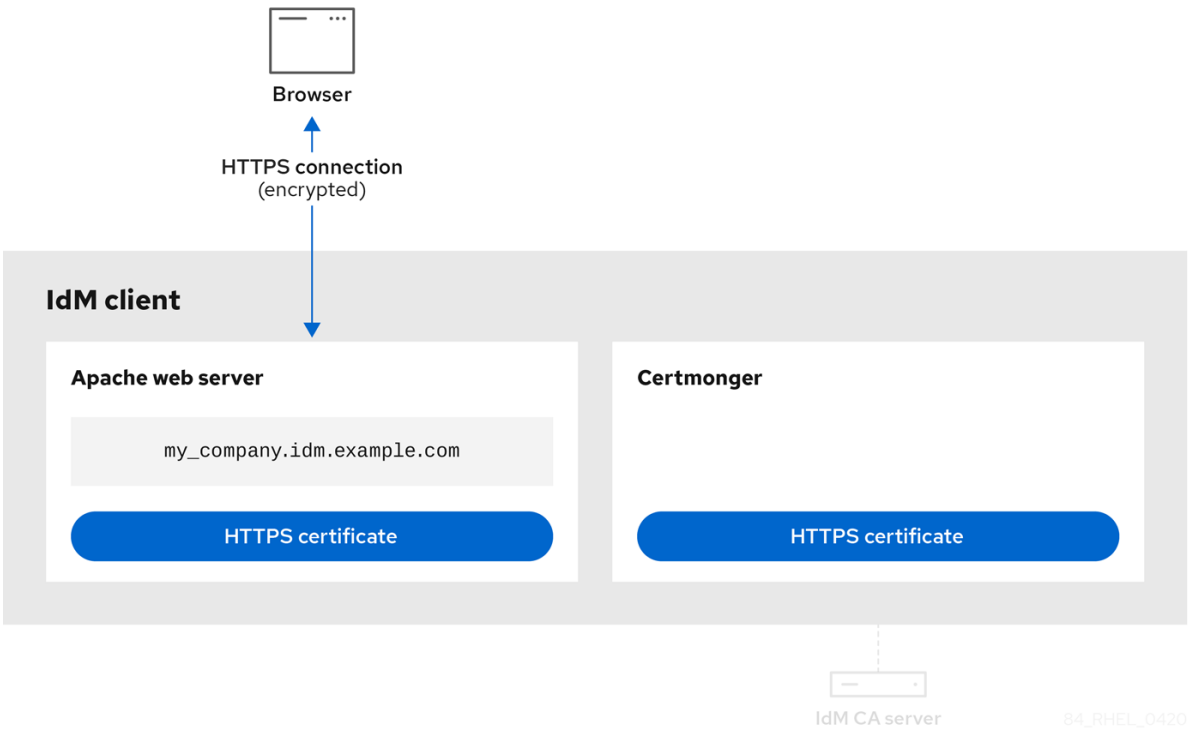
サービス証明書を発行する IdM CA は、Web サーバーで HTTPS 証明書を発行する IdM CA を示しています。

図79.5 サービス証明書を発行する IdM CA



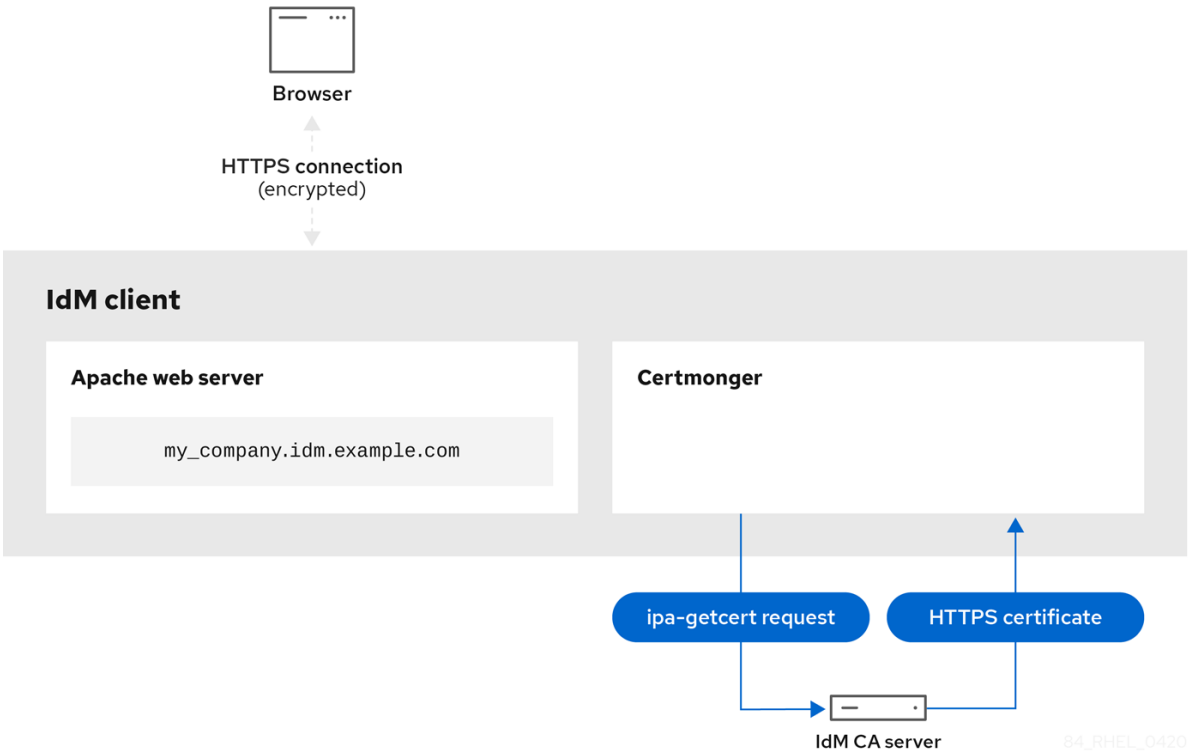
Certmonger によるサービス証明書の適用 は、**certmonger** が HTTPS 証明書を Id M クライアントの適切な場所に配置し、指示された場合は **httpd** サービスを再起動することを示します。その後、Apache サーバーは HTTPS 証明書を使用して、Apache サーバーとブラウザー間のトラフィックを暗号化します。

図79.6 Certmonger によるサービス証明書の適用



古い証明書が有効期限に近づいているときに新しい証明書を要求する certmonger は、証明書の有効期限が切れる前に、**certmonger** が Id MCA からのサービス証明書の更新を自動的に要求していることを示しています。IdM CA は、新しい証明書を発行します。

図79.7 古い証明書が有効期限に近づいているときに新しい証明書を要求する certmonger



79.6. シングルインスタンスの APACHE HTTP SERVER 設定

シングルインスタンスの Apache HTTP Server を設定して、静的 HTML コンテンツを提供できます。

Web サーバーに関連付けられた全ドメインにサーバーから同じコンテンツを提供する必要がある場合は、この手順に従います。異なるドメインに異なるコンテンツを提供する場合は、名前ベースの仮想ホストを設定します。詳細は Apache 名ベースの仮想ホストの設定 を参照してください。

手順

1. **httpd** パッケージをインストールします。

```
# yum install httpd
```

2. **firewalld** を使用する場合は、ローカルのファイアウォールで TCP ポート **80** を開きます。

```
# firewall-cmd --permanent --add-port=80/tcp
# firewall-cmd --reload
```

3. **httpd** サービスを有効にして起動します。

```
# systemctl enable --now httpd
```

4. 必要に応じて、HTML ファイルを **/var/www/html/** ディレクトリーに追加します。

注記

/var/www/html/ にコンテンツを追加する場合には、**httpd** を実行するユーザーが、デフォルトでファイルとディレクトリーを読み取れるようにする必要があります。コンテンツの所有者は、 **root** ユーザーおよび **root** ユーザーグループ、または管理者別のユーザーまたはグループのいずれかになります。コンテンツの所有者が **root** ユーザーおよび **root** ユーザーグループの場合には、他のユーザーがファイルを読み取れるようにする必要があります。すべてのファイルとディレクトリーの SELinux コンテキストは **httpd_sys_content_t** である必要があります。これはデフォルトで **/var/www** ディレクトリー内の全コンテンツに適用されます。

検証手順

- Web ブラウザーで **http://my_company.idm.example.com/** または **http://server_IP/** に移動します。
  /var/www/html/ ディレクトリーが空であるか、**index.html** または **index.htm** ファイルが含まれていない場合は、Apache が **Red Hat Enterprise Linux Test Page** を表示します。**/var/www/html/** に異なる名前の HTML ファイルが含まれている場合は、**http://server_IP/example.html** や **http://my_company.idm.example.com/example.html** などの URL をファイルに指定してそのファイルを読み込むことができます。

関連情報

- Apache マニュアル: Apache HTTP Server マニュアルのインストール
- **httpd.service(8)** man ページを参照してください。

79.7. APACHE HTTP SERVER への TLS 暗号化の追加

idm.example.com ドメイン向けに、Apache HTTP Server **my_company.idm.example.com** で TLS 暗号化を有効にできます。

前提条件

- Apache HTTP Server **my_company.idm.example.com** をインストールして、実行している。
- certmonger を使用したサービスの IdM 証明書の取得 で説明されているように、**webserver-ca** サブ CA から TLS 証明書を取得し、**/etc/pki/tls/certs/httpd.pem** ファイルに保存しました。別のパスを使用する場合は、この手順で対応する手順を調整します。
- 対応する秘密鍵を **/etc/pki/tls/private/httpd.key** ファイルに保存している。別のパスを使用する場合は、この手順で対応する手順を調整します。
- CA 証明書 **webserver-ca** が **/etc/pki/tls/private/sub-ca.crt** ファイルに保存されている。別のパスを使用する場合は、この手順で対応する手順を調整します。
- クライアントおよび Web サーバー **my_company.idm.example.com** を使用してサーバーのホスト名が Web サーバーのホスト名に対して解決されている。

手順

1. **mod_ssl** パッケージをインストールします。

```
# yum install mod_ssl
```

2. **/etc/httpd/conf.d/ssl.conf** ファイルを編集し、以下の設定を **<VirtualHost _default_:443>** ディレクティブに追加します。

   a. サーバー名を設定します。

   ```
   ServerName my_company.idm.example.com
   ```

   重要

   サーバー名は、証明書の **Common Name** フィールドに設定されているエントリーと一致している必要があります。

   a. 必要に応じて、証明書の **Subject Alt Names** (SAN) フィールドに追加のホスト名が含まれる場合に、これらのホスト名にも TLS 暗号化を提供するように **mod_ssl** を設定できます。これを設定するには、**ServerAliases** パラメーターと対応する名前を追加します。

   ```
   ServerAlias www.my_company.idm.example.com
   server.my_company.idm.example.com
   ```

   b. 秘密鍵、サーバー証明書、および CA 証明書へのパスを設定します。

   ```
   SSLCertificateKeyFile "/etc/pki/tls/private/httpd.key"
   SSLCertificateFile "/etc/pki/tls/certs/httpd.pem"
   SSLCACertificateFile "/etc/pki/tls/certs/ca.crt"
   ```

3. セキュリティー上の理由から、**root** ユーザーのみが秘密鍵ファイルにアクセスできるように設定します。

   ```
   # chown root:root /etc/pki/tls/private/httpd.key
   # chmod 600 //etc/pki/tls/private/httpd.key
   ```

   警告

   秘密鍵に権限のないユーザーがアクセスした場合は、証明書を取り消し、新しい秘密鍵を作成し、新しい証明書を要求します。そうでない場合は、TLS 接続が安全ではなくなります。

4. **firewalld** を使用する場合は、ローカルのファイアウォールでポート **443** を開きます。

   ```
   # firewall-cmd --permanent --add-port=443/tcp
   # firewall-cmd --reload
   ```

5. **httpd** サービスを再起動します。

   ```
   # systemctl restart httpd
   ```

注記

パスワードで秘密鍵ファイルを保護した場合は、**httpd** サービスの起動時に毎回このパスワードを入力する必要があります。

- ブラウザーを使用して **https://my_company.idm.example.com** に接続します。

関連情報

- SSL/TLS 暗号化
- RHEL 8 における TLS のセキュリティー上の検討事項

79.8. APACHE HTTP サーバーでサポートされる TLS プロトコルバージョンの設定

デフォルトでは、RHEL の Apache HTTP Server は、最新のブラウザーにも互換性のある安全なデフォルト値を定義するシステム全体の暗号化ポリシーを使用します。たとえば、**DEFAULT** ポリシーでは、**TLSv1.2** および **TLSv1.3** プロトコルバージョンのみが Apache で有効になるように定義します。

Apache HTTP Server **my_company.idm.example.com** がサポートする TLS プロトコルのバージョンを手動で設定できます。たとえば、環境が特定の TLS プロトコルバージョンのみを有効にする必要がある場合には、以下の手順に従います。

- お使いの環境のクライアントで、セキュリティーの低い **TLS1** (TLSv1.0) プロトコルまたは **TLS1.1** プロトコルも使用できるようにする必要がある場合。
- Apache が **TLSv1.2** プロトコルまたは **TLSv1.3** プロトコルのみに対応するように設定する場合。

前提条件

- TLS 暗号化は、Apache HTTP Server への TLS 暗号化の追加 で説明されているように、**my_company.idm.example.com** サーバーで有効になっています。

手順

1. **/etc/httpd/conf/httpd.conf** ファイルを編集し、TLS プロトコルバージョンを設定する **<VirtualHost>** ディレクティブに以下の設定を追加します。たとえば、**TLSv1.3** プロトコルのみを有効にするには、以下を実行します。

```
SSLProtocol -All TLSv1.3
```

2. **httpd** サービスを再起動します。

```
# systemctl restart httpd
```

検証手順

1. 以下のコマンドを使用して、サーバーが **TLSv1.3** に対応していることを確認します。

```
# openssl s_client -connect example.com:443 -tls1_3
```

2. 以下のコマンドを使用して、サーバーが **TLSv1.2** に対応していないことを確認します。

```
# openssl s_client -connect example.com:443 -tls1_2
```

サーバーがプロトコルに対応していない場合には、このコマンドは以下のエラーを返します。

```
140111600609088:error:1409442E:SSL routines:ssl3_read_bytes:tlsv1 alert protocol
version:ssl/record/rec_layer_s3.c:1543:SSL alert number 70
```

3. 必要に応じて、他の TLS プロトコルバージョンのコマンドを繰り返し実行します。

関連情報

- **update-crypto-policies(8)** の man ページ
- Using system-wide cryptographic policies.
- **SSLProtocol** パラメーターの詳細については、Apache マニュアルの **mod_ssl** のドキュメント Apache HTTP Server マニュアルのインストール を参照してください。

79.9. APACHE HTTP サーバーで対応している暗号の設定

デフォルトでは、Apache HTTP サーバーは、安全なデフォルト値を定義するシステム全体の暗号化ポリシーを使用します。これは、最近のブラウザーとも互換性があります。システム全体の暗号化で使用可能な暗号化のリストは、**/etc/crypto-policies/back-ends/openssl.config** ファイルを参照してください。

Apache HTTP Server **my_company.idm.example.com** がサポートする暗号を手動で設定できます。お使いの環境で特定の暗号が必要な場合は、以下の手順に従います。

前提条件

- TLS 暗号化は、Apache HTTP Server への TLS 暗号化の追加 で説明されているように、**my_company.idm.example.com** サーバーで有効になっています。

手順

1. **/etc/httpd/conf/httpd.conf** ファイルを編集し、TLS 暗号を設定する **<VirtualHost>** ディレクティブに **SSLCipherSuite** パラメーターを追加します。

```
SSLCipherSuite
"EECDH+AESGCM:EDH+AESGCM:AES256+EECDH:AES256+EDH:!SHA1:!SHA256"
```

この例では、**EECDH+AESGCM**、**EDH+AESGCM**、**AES256+EECDH**、および **AES256+EDH** 暗号のみを有効にし、**SHA1** および **SHA256** メッセージ認証コード (MAC) を使用するすべての暗号を無効にします。

2. **httpd** サービスを再起動します。

```
# systemctl restart httpd
```

検証手順

1. Apache HTTP Server が対応する暗号化のリストを表示するには、以下を行います。

   a. **nmap** パッケージをインストールします。

   ```
   # yum install nmap
   ```

   b. **nmap** ユーティリティーを使用して、対応している暗号を表示します。

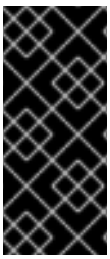
   ```
   # nmap --script ssl-enum-ciphers -p 443 example.com
   ...
   PORT    STATE SERVICE
   443/tcp open  https
   | ssl-enum-ciphers:
   |   TLSv1.2:
   |     ciphers:
   |       TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (ecdh_x25519) - A
   |       TLS_DHE_RSA_WITH_AES_256_GCM_SHA384 (dh 2048) - A
   |       TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256 (ecdh_x25519) - A
   ...
   ```

関連情報

- **update-crypto-policies(8)** の man ページ
- Using system-wide cryptographic policies.
- Apache HTTP Server マニュアルのインストール - SSLCipherSuite

79.10. TLS クライアント証明書認証の設定

クライアント証明書認証を使用すると、管理者は、証明書を使用して認証したユーザーだけが Web サーバー **my_company.idm.example.com** のリソースにアクセスできるようにすることが可能です。**/var/www/html/Example/** ディレクトリーにクライアント証明書認証を設定できます。



重要

Apache サーバー **my_company.idm.example.com** が TLS 1.3 プロトコルを使用する場合は、一部のクライアントに追加の設定が必要です。たとえば、Firefox で、**about:config** メニューの **security.tls.enable_post_handshake_auth** パラメーターを **true** に設定します。詳細は、Transport Layer Security version 1.3 in Red Hat Enterprise Linux 8 を参照してください。

前提条件

- TLS 暗号化は、Apache HTTP Server への TLS 暗号化の追加 で説明されているように、**my_company.idm.example.com** サーバーで有効になっています。

手順

1. **/etc/httpd/conf/httpd.conf** ファイルを編集し、以下の設定をクライアント認証を設定する **<VirtualHost>** ディレクティブに追加します。

```
<Directory "/var/www/html/Example/">
  SSLVerifyClient require
</Directory>
```

SSLVerifyClient require の設定では、**/var/www/html/Example/** ディレクトリーのコンテンツにクライアントがアクセスする前に、サーバーがクライアント証明書を正常に検証する必要があることを定義します。

2. **httpd** サービスを再起動します。

```
# systemctl restart httpd
```

検証手順

1. **curl** ユーティリティーを使用して、クライアント認証なしで URL **https://my_company.idm.example.com/Example/** にアクセスします。

```
$ curl https://my_company.idm.example.com/Example/
curl: (56) OpenSSL SSL_read: error:1409445C:SSL routines:ssl3_read_bytes:tlsv13 alert certificate required, errno 0
```

このエラーから、Web サーバー **my_company.idm.example.com** にクライアント証明書認証が必要であることが分かります。

2. クライアントの秘密鍵と証明書、および CA 証明書を **curl** に指定して、クライアント認証で同じ URL にアクセスします。

```
$ curl --cacert ca.crt --key client.key --cert client.crt
https://my_company.idm.example.com/Example/
```

要求に成功すると、**curl** は **/var/www/html/Example/** ディレクトリーに保存されている **index.html** ファイルを表示します。

関連情報

- Apache HTTP Server マニュアルのインストール - mod_ssl 設定

79.11. 新しいユーザー証明書を要求し、クライアントにエクスポート

Identity Management (IdM) 管理者は、Web ブラウザーを使用してサーバーにアクセスするユーザーに対して、特定の IdM サブ CA が発行する証明書での認証を求めるように IdM クライアントで実行中の Web サーバーを設定できます。特定の IdM サブ CA からユーザー証明書を要求し、ユーザーが Web ブラウザー経由で Web サーバーにアクセスするホストにその証明書と対応する秘密鍵をエクスポートするには、本セクションを実行するには、次の手順に従います。その後、ブラウザーに証明書と秘密鍵をインポート します。

手順

1. 必要に応じて、新しいディレクトリー (例: ~/**certdb**/) を作成し、証明書の一時データベースを作成します。要求されたら、NSS 証明書の DB パスワードを作成し、後続の手順で生成される証明書への鍵を暗号化します。

```
# mkdir ~/certdb/
```

```
# certutil -N -d ~/certdb/
Enter a password which will be used to encrypt your keys.
The password should be at least 8 characters long,
and should contain at least one non-alphabetic character.

Enter new password:
Re-enter password:
```

2. 証明書署名要求 (CSR) を作成し、その出力をファイルにリダイレクトします。たとえば、**IDM.EXAMPLE.COM** レルムの **idm_user** ユーザーの **4096** ビット証明書に対して、**certificate_request.csr** という名前の CSR を作成する場合は、判別を簡単にするために、証明書の秘密鍵のニックネームを **idm_user** に設定し、発行先を **CN=idm_user,O=IDM.EXAMPLE.COM** に設定します。

```
# certutil -R -d ~/certdb/ -a -g 4096 -n idm_user -s "CN=idm_user,O=IDM.EXAMPLE.COM"
> certificate_request.csr
```

3. プロンプトが表示されたら、**certutil** を使用して一時データベースを作成したときに入力したパスワードを入力します。その後、止めるように言われるまで、ランダムにタイピングし続けます。

```
Enter Password or Pin for "NSS Certificate DB":

A random seed must be generated that will be used in the
creation of your key.  One of the easiest ways to create a
random seed is to use the timing of keystrokes on a keyboard.

To begin, type keys on the keyboard until this progress meter
is full.  DO NOT USE THE AUTOREPEAT FUNCTION ON YOUR KEYBOARD!


Continue typing until the progress meter is full:
```

4. 証明書要求ファイルをサーバーに送信します。新しく発行した証明書に関連付ける Kerberos プリンシパルと、証明書を保存する出力ファイルを指定し、必要に応じて証明書のプロファイルを指定します。証明書を発行する IdM サブ CA を指定します。たとえば、**idm_user@IDM.EXAMPLE.COM** プリンシパルの **IECUserRoles** プロファイル (ユーザーロール拡張を追加したプロファイル) の証明書を **webclient-ca** から取得して、**~/idm_user.pem** ファイルに保存するには、次のコマンドを実行します。

```
# ipa cert-request certificate_request.csr --principal=idm_user@IDM.EXAMPLE.COM --
profile-id=IECUserRoles --ca=webclient-ca --certificate-out=~/idm_user.pem
```

5. 証明書を NSS データベースに追加します。証明書が NSS データベースの秘密鍵に一致するように、CSR を作成する際に使用したニックネームを設定するには、**-n** オプションを使用します。**-t** オプションは信頼レベルを設定します。詳細は、certutil(1) man ページを参照してください。**-i** オプションは、入力証明書ファイルを指定します。たとえば、**idm_user** ニックネームを持つ証明書を NSS データベースに追加するには、次のコマンドを実行します。証明書は、**~/certdb/** データベースの **~/idm_user.pem** ファイルに保存されます。

```
# certutil -A -d ~/certdb/ -n idm_user -t "P,," -i ~/idm_user.pem
```

6. NSS データベースの鍵で、ニックネームが **(orphan)** と表示されていないことを確認します。たとえば、~/**certdb**/ データベースに保存されている証明書で、対応する鍵が存在することを確認するには、以下のコマンドを実行します。

```
# certutil -K -d ~/certdb/
< 0> rsa      5ad14d41463b87a095b1896cf0068ccc467df395   NSS Certificate DB:idm_user
```

7. 証明書を、NSS データベースから PKCS12 形式にエクスポートするには、**pk12util** コマンドを使用します。たとえば、NSS データベース **/root/certdb** から ~/**idm_user.p12** ファイルへ、**idm_user** ニックネームを持つ証明書をエクスポートする場合は、次のコマンドを実行します。

```
# pk12util -d ~/certdb -o ~/idm_user.p12 -n idm_user
Enter Password or Pin for "NSS Certificate DB":
Enter password for PKCS12 file:
Re-enter password:
pk12util: PKCS12 EXPORT SUCCESSFUL
```

8. **idm_user** の証明書認証を有効にするホストに、証明書を転送します。

```
# scp ~/idm_user.p12 idm_user@client.idm.example.com:/home/idm_user/
```

9. セキュリティー上の理由から、証明書が転送されたホストの、.pkcs12 ファイルが格納されているディレクトリーに、other グループがアクセスできないようにします。

```
# chmod o-rwx /home/idm_user/
```

10. セキュリティー上の理由から、一時 NSS データベースおよび .pkcs12 ファイルを、サーバーから削除します。

```
# rm ~/certdb/
# rm ~/idm_user.p12
```

79.12. 証明書認証を有効にするためのブラウザーの設定

Web UI を使用した Identity Management (IdM) へのログイン時に証明書で認証できるようにするには、ユーザーおよび関連の認証局 (CA) 証明書を Mozilla Firefox または Google Chrome ブラウザーにインポートする必要があります。ブラウザーが実行しているホスト自体は、IdM ドメインの一部である必要はありません。

IdM が Web UI への接続をサポートしているブラウザーは以下のとおりです。

- Mozilla Firefox 38 以降
- Google Chrome 46 以降

次の手順は、Mozilla Firefox 57.0.1 ブラウザーを設定する方法を説明します。

前提条件

- PKCS #12 形式で自由にブラウザーにインポートできる ユーザー証明書 がある。
- サブ CA 証明書をダウンロード し、PEM 形式で自由に使用できるようにしている。

手順

1. Firefox を開き、**設定** → **プライバシーとセキュリティー** に移動します。

図79.8 設定のプライバシーおよびセキュリティーセクション

2. **証明書を表示** をクリックします。

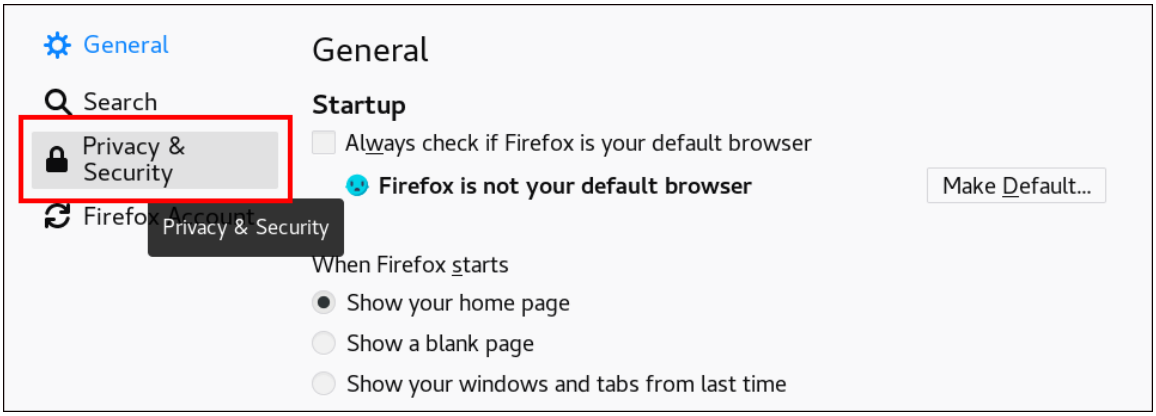
図79.9 プライバシーおよびセキュリティーで証明書を表示

3. **あなたの証明書** タブで、**インポート** をクリックします。PKCS12 形式のユーザー証明書を見つけて開きます。**OK** をクリックし、**OK** をクリックします。

4. IdM サブ CA が信頼された認証局として Firefox に認識されていることを確認するには、IdM WebUI からのサブ CA 証明書のダウンロード に保存した IdM サブ CA 証明書を信頼された認証局証明書としてインポートします。

   a. Firefox を起動し、設定に移動して、**プライバシーおよびセキュリティー**に移動します。

図79.10 設定のプライバシーおよびセキュリティーセクション



   b. **証明書を表示** をクリックします。

図79.11 プライバシーおよびセキュリティーで証明書を表示

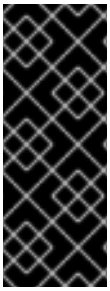
c. **認証機関** タブで、**インポート** をクリックします。サブ CA 証明書を探して開きます。証明書を信頼し、Web サイトを識別したら、**OK** をクリックし、**OK** をクリックします。

第80章 関連する証明書の特定グループの迅速な無効化

システム管理者は任意で、関連する証明書の特定のグループをすばやく無効にできます。

- 特定の軽量の Identity Management (IdM) サブ CA が発行した証明書のみを信頼するようにアプリケーションを設計します。その後、これらの証明書を発行した Identity Management (IdM) サブ CA の証明書を取り消すだけで、この証明書をすべて無効にできます。IdM で軽量のサブ CA を作成して使用する方法の詳細は、関連する証明書の特定グループの迅速な無効化 を参照してください。

- 失効予定の IdM サブ CA が発行した全証明書がすぐに無効となっていることを確認するには、このような証明書に依存するアプリケーションが IdM OCSP レスポンダーを使用するように設定します。たとえば、Firefox ブラウザーが OCSP レスポンダーを使用するように設定するには、Firefox の設定で **Query OCSP responder servers to confirm the current validity of certificates** チェックボックスが選択されているようにします。
  IdM では、証明書失効リスト (CRL) が 4 時間ごとに更新されます。IdM サブ CA によって発行されたすべての証明書を無効にするには、IdM サブ CA 証明書を失効 させます。さらに、関連する CA ACL を無効 にして、IdM サブ CA の無効化 の検討も行ってください。サブ CA を無効にするとサブ CA が新しい証明書を発行できなくなりますが、サブ CA の署名キーが保持されるため、以前に発行した証明書に、オンライン証明書ステータスプロトコル (OCSP) の応答を生成することができます。



重要

お使いの環境で OCSP を使用する場合は、サブ CA を削除しないでください。サブ CA を削除するとサブ CA の署名キーが削除されるため、そのサブ CA が発行する証明書の OCSP 応答を生成できなくなります。

サブ CA の無効化が唯一推奨されるのは、署名キーを新しくしつつも、同じサブジェクト識別名 (DN) を使用して、サブ CA を新規作成するシナリオの場合のみです。

80.1. IDM CLI での CA ACL の無効化

IdM サービスまたは IdM サービスのグループを終了する場合は、対応する既存の CA ACL を無効にすることを検討してください。

Web サーバーが IdM クライアントで実行中の場合には IdM サブ CA (**webserver-ca**) 発行の証明書を要求するように制限する TLS_web_server_authentication CA ACL と、IdM ユーザーの場合には IdM サブ CA (**webclient-ca**) 発行のユーザー証明書を要求するように制限する TLS_web_client_authentication CA ACL を無効にするには、次の手順に従います。

手順

1. 必要に応じて、IdM 環境内の CA ACL をすべて表示するには、**ipa caacl-find** コマンドを入力します。

```
$ ipa caacl-find
-----------------
3 CA ACLs matched
-----------------
  ACL name: hosts_services_caIPAserviceCert
  Enabled: TRUE

  ACL name: TLS_web_server_authentication
  Enabled: TRUE
```

```
  ACL name: TLS_web_client_authentication
  Enabled: TRUE
```

2. 必要に応じて、CA ACL の詳細を表示するには、**ipa caacl-show** コマンドを入力して、CA ACL 名を指定します。

```
$ ipa caacl-show TLS_web_server_authentication
  ACL name: TLS_web_server_authentication
  Description: CAACL for web servers authenticating to web clients using certificates issued by webserver-ca
  Enabled: TRUE
  CAs: webserver-ca
  Profiles: caIPAserviceCert
  Services: HTTP/rhel8server.idm.example.com@IDM.EXAMPLE.COM
```

3. CA ACL を無効にするには、**ipa caacl-disable** コマンドを入力して、CA ACL 名を指定します。

   - **TLS_web_server_authentication** CA ACL を無効にするには、以下を入力します。

```
$ ipa caacl-disable TLS_web_server_authentication
-----------------------------------------------
Disabled CA ACL "TLS_web_server_authentication"
-----------------------------------------------
```

   - **TLS_web_client_authentication** CA ACL を無効にするには、以下を入力します。

```
$ ipa caacl-disable TLS_web_client_authentication
-----------------------------------------------
Disabled CA ACL "TLS_web_client_authentication"
-----------------------------------------------
```

有効な CA ACL は **hosts_services_caIPAserviceCert** CA ACL のみです。

重要

CA ACL **hosts_services_caIPAserviceCert** を無効にする場合には、細心の注意を払ってください。プロファイルが **caIPAserviceCert** の **ipa** CA を IdM サーバーで使用できるように指定する別の CA ACL を用意せずに **hosts_services_caIPAserviceCert** を無効にすると、IdM の **HTTP** と **LDAP** 証明書の更新に失敗します。IdM **HTTP** 証明書および **LDAP** 証明書の期限が切れると、最終的には IdM システムに問題が発生します。

80.2. IDM サブ CA の無効化

IdM サブ CA の CA 証明書を無効にして、そのサブ CA によって発行されたすべての証明書を無効にした後、IdM サブ CA が必要なくなった場合は、IdM サブ CA を無効にすることを検討してください。サブ CA は後で再度有効にできます。

サブ CA を無効にするとサブ CA が新しい証明書を発行できなくなりますが、サブ CA の署名キーが保持されるため、以前に発行した証明書に、オンライン証明書ステータスプロトコル (OCSP) の応答を生成することができます。

前提条件

- IdM 管理者としてログインしている。

手順

- **ipa ca-disable** コマンドを入力し、サブ CA の名前を指定します。

```
$ ipa ca-disable webserver-CA
--------------------
Disabled CA "webserver-CA"
--------------------
```

第81章 IDM の VAULT

本章では、Identity Management(IdM) の vault について説明します。本章では、以下のトピックを紹介します。

- vault の概念。
- vault に関連付けられる各種ロール。
- IdM で利用可能な各種 vault - セキュリティーおよびアクセス制御のレベル別 。
- IdM で利用可能な各種 vault - 所有権別 。
- vault コンテナーの概念。
- IdM での vault 管理向けの基本的なコマンド 。
- IdM で vault を使用するのに必要な KPA (Key Recovery Authority) のインストール 。

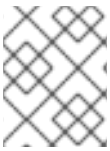
81.1. VAULT およびその利点

vault は、機密データをすべてセキュアに保存しつつも、1 箇所で都合よく Identity Management (IdM) を使用するのに便利な機能です。 vault にはさまざまなタイプがあり、要件に応じて使用する vault を選択する必要があります。

vault とは、シークレットの保存、取得、共有、および復旧を行うための (IdM の) セキュアな場所を指し、シークレットは、通常は一部のユーザーまたはエンティティーグループのみがアクセスできる、認証情報などの機密データを指します。たとえば、シークレットには以下が含まれます。

- パスワード
- 暗証番号
- SSH 秘密鍵

vault はパスワードマネージャーと類似しています。valut を使用する場合、通常、パスワードマネージャーと同様に、ロックを解除するためのプライマリーのパスワードを 1 つ生成し、記憶して、vault に保存されている情報にアクセスする必要があります。ただし、標準の vault を指定することも可能です。標準の vault では、vault に保存されているシークレットにアクセスするためにパスワードを入力する必要はありません。



注記

IdM の vault は、認証情報を保存して、IdM 関連以外の外部サービスに対して認証を可能にすることを目的としています。

IdM vault には他にも、次のような重要な特徴があります。

- vault にアクセスできるのは、vault の所有者と、vault メンバーとして vault の所有者が選択した IdM ユーザーだけです。また、IdM 管理者も vault にアクセスできます。
- ユーザーに vault を作成する権限がない場合には、IdM 管理者が vault を作成し、そのユーザーを所有者として設定できます。

- ユーザーおよびサービスは、IdM ドメインに登録されているマシンからであれば、vault に保存されているシークレットにアクセスできます。
- vault 1 つに追加できるシークレットは 1 つのみです (例: ファイル 1 つ)。ただし、ファイル自体には、パスワード、キータブ、証明書など複数のシークレットを含めることができます。

注記

Vault は、IdM Web UI ではなく、IdM コマンドライン (CLI) からしか利用できません。

81.2. VAULT の所有者、メンバー、および管理者

Identity Management (IdM) で識別される vault ユーザータイプは以下のとおりです。

Vault 所有者

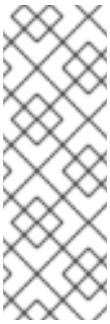
vault 所有者は、vault の基本的な管理権限のあるユーザーまたはサービスです。たとえば、vault の所有者は vault のプロパティーを変更したり、新しい vault メンバーを追加したりできます。
各 vault には最低でも所有者が 1 人必要です。vault には複数の所有者を指定することもできます。

Vault メンバー

vault メンバーは、別のユーザーまたはサービスが作成した vault にアクセスできるユーザーまたはサービスです。

Vault 管理者

vault 管理者は全 vault に制限なくアクセスでき、vault の操作をすべて実行できます。



注記

対称と非対称 vault は、パスワードまたは鍵で保護されており、特別なアクセス制御ルールが適用されます (Vault タイプ を参照)。管理者は、以下を行うためにこの特別なルールを満たす必要があります。

- 対称および非対称 vault のシークレットにアクセスする。
- vault パスワードまたはキーを変更またはリセットする。

vault 管理者は、**vault administrators** 特権を持つユーザーです。IdM のロールベースアクセス制御 (RBAC) のコンテキストでの特権とは、ロールに適用できるパーミッションのグループのことです。

Vault ユーザー

vault ユーザーは、vault のあるコンテナー内のユーザーです。**Vault ユーザー** 情報は、**ipa vault-show** などの特定のコマンドの出力に表示されます。

```
$ ipa vault-show my_vault
  Vault name: my_vault
  Type: standard
  Owner users: user
  Vault user: user
```

vault コンテナーおよびユーザー vault の詳細は、Vault コンテナー を参照してください。

関連情報

- vault のタイプの詳細は、標準、対称および非対称 vault を参照してください。

81.3. 標準、対称および非対称 VAULT

IdM では、セキュリティーおよびアクセス制御のレベルをもとに vault を以下のタイプに分類します。

標準 vault

Vault の所有者と vault メンバーは、パスワードやキーを使用せずにシークレットをアーカイブして取得できます。

対称 vault

vault のシークレットは対称キーを使用して保護されます。Vault の所有者とメンバーは、シークレットをアーカイブして取得できますが、vault パスワードを指定する必要があります。

非対称 vault

vault のシークレットは非対称キーを使用して保護されます。ユーザーは公開鍵でシークレットをアーカイブし、秘密鍵でシークレットを取得します。vault メンバーはシークレットのアーカイブのみが可能ですが、vault 所有者はシークレットのアーカイブと取得の両方が可能です。

81.4. ユーザー、サービスおよび共有 VAULT

IdM では、所有権をもとに vault を複数のタイプに分類します。以下の表 には、各タイプ、所有者、および使用方法に関する情報が含まれます。

表81.1 所有権に基づく IdM vault

タイプ	説明	所有者	備考
ユーザー vault	ユーザーのプライベート vault	ユーザー x 1	IdM 管理者が許可すれば、ユーザーは 1 つまたは複数のユーザー vault を所有できます。
サービス vault	サービスのプライベート vault	サービス x 1	IdM 管理者が許可すれば、ユーザーは 1 つまたは複数のサービス vault を所有できます。
共有 vault	複数のユーザーおよびグループで共有される vault	vault を作成した vault の管理者	IdM 管理者が許可すれば、ユーザーおよびサービスは 1 つまたは複数のユーザー vault を所有できます。vault の作成者以外に、vault 管理者が vault に対して完全なアクセス権があります。

81.5. VAULT コンテナー

vault コンテナーは vault のコレクションです。以下の表 は、Identity Management (IdM) が提供するデフォルトの vault コンテナーのリストです。

表81.2 IdM のデフォルトの vault コンテナー

タイプ	説明	目的

タイプ	説明	目的
ユーザーコンテナー	ユーザーのプライベートコンテナー	特定ユーザーのユーザー vault を格納します。
サービスコンテナー	サービスのプライベートコンテナー	特定のサービスのサービス vault を格納します。
共有コンテナー	複数のユーザーおよびサービスのコンテナー	複数のユーザーまたはサービスで共有可能な vault を格納します。

IdM では、ユーザーまたはサービスのプライベート vault が初めて作成されると、ユーザーまたはサービスごとにユーザーコンテナーおよびサービスコンテナーを自動的に作成します。ユーザーまたはサービスが削除されると、IdM はコンテナーとそのコンテンツを削除します。

81.6. 基本的な IDM VAULT コマンド

以下に概説する基本的なコマンドを使用して、Identity Management (IdM) vault を管理できます。以下の表 には、**ipa vault-*** コマンドとその目的が記載されています。

注記

ipa vault-* コマンドを実行する前に、IdM ドメインのサーバー 1 台以上に Key Recovery Authority (KRA) 証明書システムコンポーネントをインストールします。詳細は IdM での Key Recovery Authority (KRA) のインストール を参照してください。

表81.3 基本的な IdM vault コマンドおよび説明

コマンド	目的
ipa help vault	IdM vault コマンドおよびサンプル vault コマンドの概念などの情報を表示します。
ipa vault-add --help、ipa vault-find --help	特定の **ipa vault-*** コマンドに **--help** オプションを追加すると、このコマンドで利用可能なオプションと詳細なヘルプが表示されます。

ipa vault-show user_vault --user idm_user	vault メンバーとして vault にアクセスする場合は、vault 所有者を指定する必要があります。vault 所有者を指定しない場合には、IdM により vault が見つからない旨が通知されます。
	`[admin@server ~]$ ipa vault-show user_vault`
	`ipa: ERROR: user_vault: vault not found`

コマンド	目的
ipa vault-show shared_vault --shared	共有 vault にアクセスする場合には、アクセスする vault が共有 vault であることを指定する必要があります。それ以外の場合は、IdM により vault が見つからない旨が通知されます。 [admin@server ~]$ ipa vault-show shared_vault ipa: ERROR: shared_vault: vault not found

81.7. IDM での KEY RECOVERY AUTHORITY (KRA) のインストール

以下の手順に従って、特定の IdM サーバーに Key Recovery Authority (KRA) Certificate System (CS) コンポーネントをインストールして、Identity Management (IdM) の vault を有効にします。

前提条件

- IdM サーバーに **root** としてログインしている。
- IdM 認証局が IdM サーバーにインストールされている。
- **Directory Manager** 認証情報がある。

手順

- KRA をインストールします。

```
# ipa-kra-install
```

重要

非表示のレプリカに、IdM クラスターの最初の KRA をインストールできます。ただし、追加の KRA をインストールするには、非表示レプリカを一時的にアクティベートしてから、表示されているレプリカに KRA のクローンをインストールする必要があります。その後に、最初に非表示レプリカを再度非表示にできます。

注記

vault サービスの可用性と耐障害性を高めるには、2 台以上の IdM サーバーに KRA をインストールします。複数の KRA サーバーを保持することで、データの損失を防ぎます。

関連情報

- Demoting or promoting hidden replicas を参照してください。
- The hidden replica mode を参照してください。

第82章 IDM ユーザー VAULT の使用: シークレットの保存および取得

本章では、Identity Management でユーザー vault を使用する方法を説明します。具体的には、IdM vault にシークレットを保存する方法と、シークレットを取得する方法を説明します。異なる IdM クライアント 2 台から保存と取得が可能です。

前提条件

- Key Recovery Authority (KRA) Certificate System コンポーネントが IdM ドメインの 1 つ以上のサーバーにインストールされている。詳細は IdM での Key Recovery Authority (KRA) のインストール を参照してください。

82.1. ユーザー VAULT でのシークレットの保存

この手順に従って、機密情報を含むファイルを安全に保存するための 1 つ以上のプライベート vault を含む vault コンテナーを作成します。以下の手順で使用する例では、**idm_user** ユーザーが標準タイプの vault を作成します。標準タイプの vault では、ファイルへのアクセス時に **idm_user** を認証する必要がありません。**IdM_user** は、ユーザーがログインしている IdM クライアントからファイルを取得できます。

この手順では、以下を想定しています。

- **IdM_user** は vault を作成するユーザーである。
- **my_vault** はユーザーの証明書保存に使用する vault である。
- アーカイブした証明書にアクセスするのに vault のパスワードを指定しなくてもいいように vault タイプが **standard** に設定されている。
- **secret.txt** は vault に保存する証明書が含まれるファイルです。

前提条件

- **idm_user** のパスワードを知っている。
- IdM クライアントであるホストにログインしている。

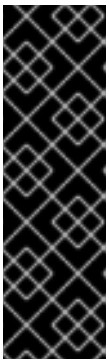
手順

1. **idm_user** の Kerberos Ticket Granting Ticket (TGT) を取得します。

```
$ kinit idm_user
```

2. **ipa vault-add** コマンドに **--type standard** オプションを指定して、標準 vault を作成します。

```
$ ipa vault-add my_vault --type standard
----------------------
Added vault "my_vault"
----------------------
  Vault name: my_vault
  Type: standard
  Owner users: idm_user
  Vault user: idm_user
```



重要

最初のユーザー vault の作成には、同じユーザーが使用されているようにしてください。ユーザーの最初の vault を作成すると、ユーザーの vault コンテナーも作成されます。作成エージェントは vault コンテナーの所有者になります。

たとえば、**admin** などの別のユーザーが **user1** の最初のユーザー vault を作成する場合には、ユーザーの vault コンテナーの所有者も **admin** になり、**user1** はユーザー vault にアクセスしたり、新しいユーザー vault を作成したりできません。

3. **ipa vault-archive** コマンドに **--in** オプションを指定して、**secret.txt** ファイルを vault にアーカイブします。

```
$ ipa vault-archive my_vault --in secret.txt
----------------------------------
Archived data into vault "my_vault"
----------------------------------
```

82.2. ユーザー VAULT からのシークレットの取得

Identity Management (IdM) では、ユーザープライベート vault からシークレットを取得して、ログインしている IdM クライアントに配置できます。

この手順に従って、**idm_user** という名前の IdM ユーザーが **my_vault** という名前のユーザープライベート vault からシークレットを取得して **idm_client.idm.example.com** に配置します。

前提条件

- **IdM_user** が **my_vault** の所有者である。
- **IdM_user** が vault にシークレットをアーカイブしてある。
- **my_vault** は標準の vault であるため、**idm_user** は vault のコンテンツへのアクセスにパスワードを入力する必要はありません。

手順

1. **idm_client** に **idm_user** として SSH 接続します。

```
$ ssh idm_user@idm_client.idm.example.com
```

2. **idm_user** としてログインします。

```
$ kinit user
```

3. **--out** オプションを指定して **ipa vault-retrieve --out** コマンドを使用し、vault のコンテンツを取得して、**secret_exported.txt** ファイルに保存します。

```
$ ipa vault-retrieve my_vault --out secret_exported.txt
-------------------------------------
Retrieved data from vault "my_vault"
-------------------------------------
```

82.3. 関連情報

- Ansible を使用した IdM ユーザー vault の管理: シークレットの保存および取得 を参照してください。

第83章 ANSIBLE を使用した IDM ユーザー VAULT の管理: シークレットの保存および取得

本章では、Ansible **vault** モジュールを使用して Identity Management でユーザー vault を管理する方法を説明します。具体的には、ユーザーが Ansible Playbook を使用して以下の 3 つのアクションを実行する方法を説明しています。

- IdM でユーザーコンテナーを作成する。
- シークレットを vault に保存する。
- vault からシークレットを取得する。

異なる IdM クライアント 2 台から保存と取得が可能です。

前提条件

- Key Recovery Authority (KRA) Certificate System コンポーネントが IdM ドメインの 1 つ以上のサーバーにインストールされている。詳細は IdM での Key Recovery Authority (KRA) のインストール を参照してください。

83.1. ANSIBLE を使用して IDM に標準ユーザー VAULT を存在させる手順

以下の手順に従って、Ansible Playbook を使用して 1 つ以上のプライベート vault を持つ vault コンテナーを作成し、機密情報を安全に保存します。以下の手順で使用する例では、**idm_user ユーザー** は **my_vault** という名前の標準タイプの vault を作成します。標準タイプの vault では、ファイルへのアクセス時に **idm_user** を認証する必要がありません。**IdM_user** は、ユーザーがログインしている IdM クライアントからファイルを取得できます。

前提条件

- Ansible コントローラー (手順の内容を実行するホスト) に ansible-freeipa パッケージがインストールされている。
- **idm_user** のパスワードを知っている。

手順

1. **/usr/share/doc/ansible-freeipa/playbooks/vault** ディレクトリーに移動します。

```
$ cd /usr/share/doc/ansible-freeipa/playbooks/vault
```

2. **inventory.file** などのインベントリーファイルを作成します。

```
$ touch inventory.file
```

3. **inventory.file** を開き、**[ipaserver]** セクションに、設定する IdM サーバーを定義します。たとえば、Ansible に対して **server.idm.example.com** を設定するように指示するには、次のコマンドを実行します。

```
[ipaserver]
server.idm.example.com
```

4. Ansible Playbook の **ensure-standard-vault-is-present.yml** ファイルのコピーを作成します。以下に例を示します。

```
$ cp ensure-standard-vault-is-present.yml ensure-standard-vault-is-present-copy.yml
```

5. **ensure-standard-vault-is-present-copy.yml** ファイルを開いて編集します。

6. **ipavault** タスクセクションに以下の変数を設定して、ファイルを調整します。

- **ipaadmin_principal** 変数は idm_user に設定します。
- **ipaadmin_password** 変数は idm_user のパスワードに設定します。
- **user** 変数は idm_user に設定します。
- **name** 変数は my_vault に設定します。
- **vault_type** 変数は standard に設定します。
  今回の例で使用するように変更した Ansible Playbook ファイル:

```
---
- name: Tests
  hosts: ipaserver
  gather_facts: false

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - ipavault:
      ipaadmin_principal: idm_user
      ipaadmin_password: idm_user_password
      user: idm_user
      name: my_vault
      vault_type: standard
```

7. ファイルを保存します。

8. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file ensure-standard-vault-is-present-copy.yml
```

83.2. ANSIBLE を使用して IDM の標準ユーザー VAULT でシークレットをアーカイブする手順

以下の手順に従って、Ansible Playbook を使用してパーソナル vault に機密情報を保存します。この例では、**idm_user** ユーザーは **my_vault** という名前の vault に **password.txt** という名前で機密情報が含まれるファイルをアーカイブします。

前提条件

- Ansible コントローラー (手順の内容を実行するホスト) に ansible-freeipa パッケージがインストールされている。

- idm_user のパスワードを知っている。
- IdM_user が所有者であるか、my_vault のメンバーユーザーである。
- password.txt (my_vault にアーカイブするシークレット) にアクセスできる。

手順

1. **/usr/share/doc/ansible-freeipa/playbooks/vault** ディレクトリーに移動します。

```
$ cd /usr/share/doc/ansible-freeipa/playbooks/vault
```

2. インベントリーファイルを開き、設定する IdM サーバーが **[ipaserver]** セクションに記載されていることを確認します。たとえば、Ansible に対して **server.idm.example.com** を設定するように指示するには、次のコマンドを実行します。

```
[ipaserver]
server.idm.example.com
```

3. Ansible Playbook の **data-archive-in-symmetric-vault.yml** ファイルのコピーを作成して symmetric を standard に置き換えます。以下に例を示します。

```
$ cp data-archive-in-symmetric-vault.yml data-archive-in-standard-vault-copy.yml
```

4. **data-archive-in-standard-vault-copy.yml** ファイルを開いて編集します。
5. **ipavault** タスクセクションに以下の変数を設定して、ファイルを調整します。
   - **ipaadmin_principal** 変数は idm_user に設定します。
   - **ipaadmin_password** 変数は idm_user のパスワードに設定します。
   - **user** 変数は idm_user に設定します。
   - **name** 変数は my_vault に設定します。
   - **in** 変数は機密情報が含まれるファイルへのパスに設定します。
   - **action** 変数は member に設定します。
     今回の例で使用するように変更した Ansible Playbook ファイル:

```
---
- name: Tests
  hosts: ipaserver
  gather_facts: false

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - ipavault:
      ipaadmin_principal: idm_user
      ipaadmin_password: idm_user_password
      user: idm_user
```

```
    name: my_vault
    in: /usr/share/doc/ansible-freeipa/playbooks/vault/password.txt
    action: member
```

6. ファイルを保存します。
7. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file data-archive-in-standard-vault-copy.yml
```

83.3. ANSIBLE を使用して IDM の標準ユーザー VAULT からシークレットを取得する手順

以下の手順に従って、Ansible Playbook を使用してユーザーのパーソナル vault からシークレットを取得します。以下の手順で使用する例では、**idm_user** ユーザーは、**my_vault** という名前の標準タイプの vault から機密データを含むファイルを取得して、**host01** という名前の IdM クライアントに配置します。ファイルへのアクセス時に **IdM_user** を認証する必要はありません。**IdM_user** は Ansible を使用して、Ansible がインストールされている IdM クライアントからファイルを取得できます。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- **idm_user** のパスワードを知っている。
- **IdM_user** が **my_vault** の所有者である。
- **IdM_user** が **my_vault** にシークレットを保存している。
- Ansible が、シークレットを取得して配置する先の IdM ホストのディレクトリーに書き込みができる。
- **IdM_user** はシークレットを取得して配置する先の IdM ホストのディレクトリーから読み取りができる。

手順

1. **/usr/share/doc/ansible-freeipa/playbooks/vault** ディレクトリーに移動します。

```
$ cd /usr/share/doc/ansible-freeipa/playbooks/vault
```

2. インベントリーファイルを開き、明確に定義されたセクションで、シークレットを取得する IdM クライアントを記載します。たとえば、Ansible に対して **host01.idm.example.com** にシークレットを取得して配置するよう指示するには、次のコマンドを実行します。

```
[ipahost]
host01.idm.example.com
```

3. Ansible Playbook ファイル (**retrive-data-symmetric-vault.yml**) のコピーを作成します。symbolicmetric を Standard に置き換えます。以下に例を示します。

```
$ cp retrive-data-symmetric-vault.yml retrieve-data-standard-vault.yml-copy.yml
```

4. **retrieve-data-standard-vault.yml-copy.yml** ファイルを開いて編集します。
5. **hosts** 変数は **ipahost** に設定して、ファイルを調整します。
6. **ipavault** タスクセクションに以下の変数を設定して、ファイルを調整します。
   - **ipaadmin_principal** 変数は **idm_user** に設定します。
   - **ipaadmin_password** 変数は **idm_user** のパスワードに設定します。
   - **user** 変数は **idm_user** に設定します。
   - **name** 変数は **my_vault** に設定します。
   - **out** 変数は、シークレットをエクスポートするファイルの完全パスに設定します。
   - **state** 変数は **retrieved** に設定します。
     今回の例で使用するように変更した Ansible Playbook ファイル:

```
---
- name: Tests
  hosts: ipahost
  gather_facts: false

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - ipavault:
      ipaadmin_principal: idm_user
      ipaadmin_password: idm_user_password
      user: idm_user
      name: my_vault
      out: /tmp/password_exported.txt
      state: retrieved
```

7. ファイルを保存します。
8. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file retrieve-data-standard-vault.yml-copy.yml
```

検証手順

1. **host01** に **user01** として **SSH** 接続します。

```
$ ssh user01@host01.idm.example.com
```

2. Ansible Playbook ファイルに **out** 変数で指定したファイルを表示します。

```
$ vim /tmp/password_exported.txt
```

これで、エクスポートされたシークレットが表示されます。

- Ansible を使用して IdM vault およびユーザーシークレットを管理する方法および、Playbook 変数の情報は、**/usr/share/doc/ansible-freeipa/** ディレクトリーで利用可能な README-vault.md Markdown ファイルおよび **/usr/share/doc/ansible-freeipa/playbooks/vault/** で利用可能なサンプルの Playbook を参照してください。

第84章 IDM サービスシークレットの管理: シークレットの保存と取得

本セクションでは、管理者が **ansible-freeipa vault** モジュールを使用してサービスシークレットを一元的にセキュアに保存する方法を説明します。この例で使用される vault は非対称であるため、これを使用する場合は、管理者は以下の手順を実行する必要があります。

1. **openssl** ユーティリティーなどを使用して秘密鍵を生成する。
2. 秘密鍵をもとに公開鍵を生成する。

サービスシークレットは、管理者が vault にアーカイブする時に公開鍵を使用して暗号化されます。その後、ドメイン内の特定のマシンでホストされるサービスインスタンスが、秘密鍵を使用してシークレットを取得します。シークレットにアクセスできるのは、サービスと管理者のみです。

シークレットが漏洩した場合には、管理者はサービス Vault でシークレットを置き換えて、漏洩されていないサービスインスタンスに配布しなおすことができます。

前提条件

- Key Recovery Authority (KRA) Certificate System コンポーネントが IdM ドメインの 1 つ以上のサーバーにインストールされている。詳細は IdM での Key Recovery Authority (KRA) のインストール を参照してください。

このセクションでは、以下の手順について説明します。

1. 非対称 vault での IdM サービスシークレットの保存
2. IdM サービスインスタンスのサービスシークレットの取得
3. シークレットが漏洩した場合の IdM サービス vault シークレットの変更

使用される用語

本手順での以下の用語について説明します。

- **admin** は、サービスパスワードを管理する管理者です。
- **private-key-to-an-externally-signed-certificate.pem** は、サービスシークレットを含むファイルです (ここでは外部署名証明書への秘密鍵)。この秘密鍵と、vault からのシークレットの取得に使用する秘密鍵と混同しないようにしてください。
- **secret_vault** は、サービス向けに作成された vault です。
- **HTTP/webserver.idm.example.com** は、シークレットがアーカイブされるサービスです。
- **service-public.pem** は、**password_vault** に保存されているパスワードの暗号化に使用するサービスの公開鍵です。
- **service-private.pem** は、**secret_vault** に保存されているパスワードの復号化に使用するサービスの秘密鍵です。

84.1. 非対称 VAULT での IDM サービスシークレットの保存

この手順に従って非対称 vault を作成し、それを使用してサービスシークレットをアーカイブします。

前提条件

- IdM 管理者パスワードを把握している。

手順

1. 管理者としてログインします。

```
$ kinit admin
```

2. サービスインスタンスの公開鍵を取得します。たとえば、**openssl** ユーティリティーを使用する場合は以下を行います。

   a. **service-private.pem** 秘密鍵を生成します。

```
$ openssl genrsa -out service-private.pem 2048
Generating RSA private key, 2048 bit long modulus
.+++
.....................................+++
e is 65537 (0x10001)
```

   b. 秘密鍵をもとに **service-public.pem** 公開鍵を生成します。

```
$ openssl rsa -in service-private.pem -out service-public.pem -pubout
writing RSA key
```

3. サービスインスタンス vault として非対称 vault を作成し、公開鍵を指定します。

```
$ ipa vault-add secret_vault --service HTTP/webserver.idm.example.com --type
asymmetric --public-key-file service-public.pem
----------------------------
Added vault "secret_vault"
----------------------------
Vault name: secret_vault
Type: asymmetric
Public key: LS0tLS1C...S0tLS0tCg==
Owner users: admin
Vault service: HTTP/webserver.idm.example.com@IDM.EXAMPLE.COM
```

   vault にアーカイブされたパスワードはこの鍵で保護されます。

4. サービスシークレットをサービス vault にアーカイブします。

```
$ ipa vault-archive secret_vault --service HTTP/webserver.idm.example.com --in
private-key-to-an-externally-signed-certificate.pem
----------------------------------
Archived data into vault "secret_vault"
----------------------------------
```

   これにより、サービスインスタンスの公開鍵でシークレットが暗号化されます。

上記の手順を、シークレットを必要とする全サービスインスタンスで繰り返します。サービスインスタンスごとに新規の非対称 vault を作成します。

84.2. IDM サービスインスタンスのサービスシークレットの取得

サービスインスタンスを使用して、ローカルに保存されたサービス秘密キーを使用してサービスコンテナーのシークレットを取得するには、次の手順に従います。

前提条件

- サービス vault を所有するサービスプリンシパルのキータブにアクセスできる (例: HTTP/webserver.idm.example.com)。
- 非対称 vault を作成して vault にシークレットをアーカイブしている。
- サービス vault のシークレットの取得に使用する秘密鍵を使用できる。

手順

1. 管理者としてログインします。

```
$ kinit admin
```

2. サービスの Kerberos チケットを取得します。

```
# kinit HTTP/webserver.idm.example.com -k -t /etc/httpd/conf/ipa.keytab
```

3. サービス vault パスワードを取得します。

```
$ ipa vault-retrieve secret_vault --service HTTP/webserver.idm.example.com --private-key-file service-private.pem --out secret.txt
----------------------------------
Retrieved data from vault "secret_vault"
----------------------------------
```

84.3. シークレットが漏洩した場合の IDM サービス VAULT シークレットの変更

サービスコンテナーのシークレットを変更して、侵害されたサービスインスタンスを隔離するには、次の手順に従います。

前提条件

- **IdM 管理者** パスワードが分かっている。
- サービスシークレットの保存先の 非対称 vault を作成している。
- 新しいシークレットを生成し、そのシークレットにアクセスできる (例: **new-private-key-to-an-externally-signed-certificate.pem** ファイル)。

手順

1. 新規シークレットをサービスインスタンス vault にアーカイブします。

```
$ ipa vault-archive secret_vault --service HTTP/webserver.idm.example.com --in new-private-key-to-an-externally-signed-certificate.pem
```

```
---------------------------------
Archived data into vault "secret_vault"
---------------------------------
```

これにより、vault に保存されている現在のシークレットが上書きされます。

2. 不正アクセスがされていないサービスインスタンスのみで新規シークレットを取得します。詳細は IdM サービスインスタンスのサービスシークレットの取得 を参照し てください。

84.4. 関連情報

- Ansible を使用した IdM サービス vault の管理: シークレットの保存および取得 を参照してください。

第85章 ANSIBLE を使用した IDM サービス VAULT の管理: シークレットの保存および取得

本セクションでは、管理者が **ansible-freeipa vault** モジュールを使用してサービスシークレットを一元的にセキュアに保存する方法を説明します。この例で使用される vault は非対称であるため、これを使用する場合は、管理者は以下の手順を実行する必要があります。

1. **openssl** ユーティリティーなどを使用して秘密鍵を生成する。
2. 秘密鍵をもとに公開鍵を生成する。

サービスシークレットは、管理者が vault にアーカイブする時に公開鍵を使用して暗号化されます。その後、ドメイン内の特定のマシンでホストされるサービスインスタンスが、秘密鍵を使用してシークレットを取得します。シークレットにアクセスできるのは、サービスと管理者のみです。

シークレットが漏洩した場合には、管理者はサービス Vault でシークレットを置き換えて、漏洩されていないサービスインスタンスに配布しなおすことができます。

前提条件

- Key Recovery Authority (KRA) Certificate System コンポーネントが IdM ドメインの 1 つ以上のサーバーにインストールされている。詳細は IdM での Key Recovery Authority (KRA) のインストール を参照してください。

このセクションでは、以下の手順について説明します。

- Ansible を使用して IdM に非対称サービス vault を存在させる手順
- Ansible を使用した非対称 vault への IdM サービスシークレットの保存
- Ansible を使用した IdM サービスのサービスシークレットの取得
- シークレットが漏洩した場合の Ansible での IdM サービス vault シークレットの変更

本手順での以下の用語について説明します。

- **admin** は、サービスパスワードを管理する管理者です。
- **private-key-to-an-externally-signed-certificate.pem** は、サービスシークレットを含むファイルです (ここでは外部署名証明書への秘密鍵)。この秘密鍵と、vault からのシークレットの取得に使用する秘密鍵と混同しないようにしてください。
- **secret_vault** は、サービスシークレット保存向けに作成された vault です。
- **HTTP/webserver1.idm.example.com** は vault の所有者となるサービスです。
- **HTTP/webserver2.idm.example.com** および **HTTP/webserver3.idm.example.com** は vault メンバーサービスです。
- **service-public.pem** は、**password_vault** に保存されているパスワードの暗号化に使用するサービスの公開鍵です。
- **service-private.pem** は、**secret_vault** に保存されているパスワードの復号化に使用するサービスの秘密鍵です。

85.1. ANSIBLE を使用して IDM に非対称サービス VAULT を存在させる手順

以下の手順に従って、Ansible Playbook を使用して 1 つ以上のプライベート Vault を持つサービス vault コンテナーを作成し、機密情報を安全に保存します。以下の手順で使用する例では、管理者は **secret_vault** という名前の非対称 vault を作成します。こうすることで、vault メンバーは、vault のシークレットを取得する際に、秘密鍵を使用して必ず認証することになります。vault メンバーは、IdM クライアントからファイルを取得できます。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- **IdM 管理者** パスワードが分かっている。

手順

1. **/usr/share/doc/ansible-freeipa/playbooks/vault** ディレクトリーに移動します。

```
$ cd /usr/share/doc/ansible-freeipa/playbooks/vault
```

2. サービスインスタンスの公開鍵を取得します。たとえば、**openssl** ユーティリティーを使用する場合は以下を行います。
   a. **service-private.pem** 秘密鍵を生成します。

```
$ openssl genrsa -out service-private.pem 2048
Generating RSA private key, 2048 bit long modulus
.+++
.....................................+++
e is 65537 (0x10001)
```

   b. 秘密鍵をもとに **service-public.pem** 公開鍵を生成します。

```
$ openssl rsa -in service-private.pem -out service-public.pem -pubout
writing RSA key
```

3. オプション: **inventory.file** など、存在しない場合はインベントリーファイルを作成します。

```
$ touch inventory.file
```

4. インベントリーファイルを開き、**[ipaserver]** セクションに、設定する IdM サーバーを定義します。たとえば、Ansible に対して **server.idm.example.com** を設定するように指示するには、次のコマンドを実行します。

```
[ipaserver]
server.idm.example.com
```

5. Ansible Playbook ファイル **ensure-asymmetric-vault-is-present.yml** のコピーを作成します。以下に例を示します。

```
$ cp ensure-asymmetric-vault-is-present.yml ensure-asymmetric-service-vault-is-present-copy.yml
```

6. **ensure-asymmetric-vault-is-present-copy.yml** ファイルを開いて編集します。

7. Ansible コントローラーから **server.idm.example.com** サーバーに **service-public.pem** の公開鍵をコピーするタスクを追加します。

8. **ipavault** タスクセクションに以下の変数を設定して、残りのファイルを変更します。

   - **ipaadmin_password** 変数は IdM 管理者パスワードに設定します。
   - **secret_vault** などの **name** 変数を使用して vault の名前を定義します。
   - **vault_type** 変数は **asymmetric** に設定します。
   - **service** 変数は、vault を所有するサービスのプリンシパル (例: **HTTP/webserver1.idm.example.com**) に設定します。
   - **public_key_file** は、公開鍵の場所に設定します。
     以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Tests
  hosts: ipaserver
  gather_facts: false
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Copy public key to ipaserver.
    copy:
      src: /path/to/service-public.pem
      dest: /usr/share/doc/ansible-freeipa/playbooks/vault/service-public.pem
      mode: 0600
  - name: Add data to vault, from a LOCAL file.
    ipavault:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: secret_vault
      vault_type: asymmetric
      service: HTTP/webserver1.idm.example.com
      public_key_file: /usr/share/doc/ansible-freeipa/playbooks/vault/service-public.pem
```

9. ファイルを保存します。

10. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file ensure-asymmetric-service-vault-is-present-copy.yml
```

85.2. ANSIBLE を使用した非対称 VAULT へのメンバーサービスの追加

以下の手順に従って、Ansible Playbook を使用してメンバーサービスをサービス vault に追加し、これらすべてが vault に保存されているシークレットを取得できるようにします。以下の手順で使用する例では、IdM の管理者は **HTTP/webserver2.idm.example.com** と **HTTP/webserver3.idm.example.com** のサービスプリンシパルを **HTTP/webserver1.idm.example.com** が所有する **secret_vault** vault に追加します。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- **IdM 管理者** パスワードが分かっている。
- サービスシークレットの保存先の 非対称 vault を作成している。

手順

1. **/usr/share/doc/ansible-freeipa/playbooks/vault** ディレクトリーに移動します。

```
$ cd /usr/share/doc/ansible-freeipa/playbooks/vault
```

2. オプション: **inventory.file** など、存在しない場合はインベントリーファイルを作成します。

```
$ touch inventory.file
```

3. インベントリーファイルを開き、**[ipaserver]** セクションに、設定する IdM サーバーを定義します。たとえば、Ansible に対して **server.idm.example.com** を設定するように指示するには、次のコマンドを実行します。

```
[ipaserver]
server.idm.example.com
```

4. Ansible Playbook ファイル (**data-archive-in-asymmetric-vault.yml**) を作成します。以下に例を示します。

```
$ cp data-archive-in-asymmetric-vault.yml add-services-to-an-asymmetric-vault.yml
```

5. **data-archive-in-asymmetric-vault-copy.yml** ファイルを開いて編集します。
6. ファイルを変更するには、**ipavault** タスクセクションに以下の変数を設定します。
   - **ipaadmin_password** 変数は IdM 管理者パスワードに設定します。
   - **name** 変数は vault の名前 (例: **secret_vault**) に設定します。
   - **service** 変数は vault のサービス所有者に設定します (例: **HTTP/webserver1.idm.example.com**)。
   - services 変数を使用して、vault シークレットにアクセスできる **サービス** を定義します。
   - **action** 変数は **member** に設定します。
     今回の例で使用するように変更した Ansible Playbook ファイル:

```
---
- name: Tests
  hosts: ipaserver
  gather_facts: false

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - ipavault:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: secret_vault
      service: HTTP/webserver1.idm.example.com
      services:
      - HTTP/webserver2.idm.example.com
      - HTTP/webserver3.idm.example.com
      action: member
```

7. ファイルを保存します。
8. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file add-services-to-an-asymmetric-vault.yml
```

85.3. ANSIBLE を使用した非対称 VAULT への IDM サービスシークレットの保存

以下の手順に従って、Ansible Playbook を使用して、サービス vault にシークレットを保存し、サービスが後で取得できるようにします。以下の手順で使用する例では、管理者は **secret_vault** という名前の非対称 vault にシークレットが含まれる **PEM** ファイルを保存します。こうすることで、サービスは、vault からシークレットを取得する際に、秘密鍵を使用して必ず認証することになります。vault メンバーは、IdM クライアントからファイルを取得できます。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。

- Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
- この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
- この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。

- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- **IdM 管理者** パスワードが分かっている。
- サービスシークレットの保存先の 非対称 vault を作成している。
- シークレットが **/usr/share/doc/ansible-freeipa/playbooks/vault/private-key-to-an-externally-signed-certificate.pem** ファイルなど、Ansible コントローラーのローカルに保存されている。

手順

1. **/usr/share/doc/ansible-freeipa/playbooks/vault** ディレクトリーに移動します。

```
$ cd /usr/share/doc/ansible-freeipa/playbooks/vault
```

2. オプション: **inventory.file** など、存在しない場合はインベントリーファイルを作成します。

```
$ touch inventory.file
```

3. インベントリーファイルを開き、**[ipaserver]** セクションに、設定する IdM サーバーを定義します。たとえば、Ansible に対して **server.idm.example.com** を設定するように指示するには、次のコマンドを実行します。

```
[ipaserver]
server.idm.example.com
```

4. Ansible Playbook ファイル (**data-archive-in-asymmetric-vault.yml**) を作成します。以下に例を示します。

```
$ cp data-archive-in-asymmetric-vault.yml data-archive-in-asymmetric-vault-copy.yml
```

5. **data-archive-in-asymmetric-vault-copy.yml** ファイルを開いて編集します。
6. ファイルを変更するには、**ipavault** タスクセクションに以下の変数を設定します。
   - **ipaadmin_password** 変数は IdM 管理者パスワードに設定します。
   - **name** 変数は vault の名前 (例: **secret_vault**) に設定します。
   - **service** 変数は vault のサービス所有者に設定します (例: **HTTP/webserver1.idm.example.com**)。
   - **in** 変数は "{{ lookup('file', 'private-key-to-an-externally-signed-certificate.pem')| b64encode }}" に設定します。この設定では、Ansible が IdM サーバーではなく、Ansible コントローラーの作業ディレクトリーから秘密鍵のあるファイルを取得するようになりま

す。

- **action** 変数は **member** に設定します。
  今回の例で使用するように変更した Ansible Playbook ファイル:

```
---
- name: Tests
  hosts: ipaserver
  gather_facts: false

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - ipavault:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: secret_vault
      service: HTTP/webserver1.idm.example.com
      in: "{{ lookup('file', 'private-key-to-an-externally-signed-certificate.pem') | b64encode }}"
      action: member
```

7. ファイルを保存します。

8. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file data-archive-in-asymmetric-vault-copy.yml
```

85.4. ANSIBLE を使用した IDM サービスのサービスシークレットの取得

以下の手順に従って、Ansible Playbook を使用してサービスの代わりにサービス vault からシークレットを取得します。以下の手順で使用する例では、Playbook を実行して **secret_vault** という名前の **PEM** ファイルを取得し、Ansible インベントリーファイルに **ipaservers** として記載されている全ホストの指定の場所に保存します。

サービスは、キータブを使用して IdM に対して、さらに秘密鍵を使用して vault に対して認証を実行します。**ansible-freeipa** がインストールされている IdM クライアントから、サービスの代わりにファイルを取得できます。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

- IdM 管理者パスワードを把握している。
- サービスシークレットの保存先の 非対称 vault を作成している。
- vault にシークレットをアーカイブ している。
- Ansible コントローラーで **private_key_file** 変数が指定する場所にサービス vault のシークレットの取得に使用する秘密鍵が保存されている。

手順

1. **/usr/share/doc/ansible-freeipa/playbooks/vault** ディレクトリーに移動します。

```
$ cd /usr/share/doc/ansible-freeipa/playbooks/vault
```

2. オプション: **inventory.file** など、存在しない場合はインベントリーファイルを作成します。

```
$ touch inventory.file
```

3. インベントリーファイルを開き、以下のホストを定義します。
   - **[ipaserver]** セクションで、使用する IdM サーバーを定義します。
   - **[webservers]** セクションで、シークレットを取得するホストを定義します。たとえば Ansible に対して **webserver1.idm.example.com**、**webserver2.idm.example.com** および **webserver3.idm.example.com** にシークレットを取得するように指示するには以下を実行します。

```
[ipaserver]
server.idm.example.com

[webservers]
webserver1.idm.example.com
webserver2.idm.example.com
webserver3.idm.example.com
```

4. Ansible Playbook ファイル (**retrieve-data-asymmetric-vault.yml**) のコピーを作成します。以下に例を示します。

```
$ cp retrieve-data-asymmetric-vault.yml retrieve-data-asymmetric-vault-copy.yml
```

5. **retrieve-data-asymmetric-vault-copy.yml** ファイルを開いて編集します。
6. ファイルを変更するには、**ipavault** タスクセクションに以下の変数を設定します。
   - **ipaadmin_password** 変数は IdM 管理者パスワードに設定します。
   - **name** 変数は vault の名前 (例: **secret_vault**) に設定します。
   - **service** 変数は vault のサービス所有者に設定します (例: **HTTP/webserver1.idm.example.com**)。
   - **private_key_file** 変数は、サービス vault のシークレットの取得に使用する秘密鍵の場所に設定します。

- **out** 変数は、現在の作業ディレクトリーなど、**private-key-to-an-externally-signed-certificate.pem** シークレットの取得先となる IdM サーバーの場所に設定します。

- **action** 変数は **member** に設定します。
  今回の例で使用するように変更した Ansible Playbook ファイル:

```
---
- name: Retrieve data from vault
  hosts: ipaserver
  become: no
  gather_facts: false

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Retrieve data from the service vault
    ipavault:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: secret_vault
      service: HTTP/webserver1.idm.example.com
      vault_type: asymmetric
      private_key: "{{ lookup('file', 'service-private.pem') | b64encode }}"
      out: private-key-to-an-externally-signed-certificate.pem
      state: retrieved
```

7. Playbook に、IdM サーバーから Ansible コントローラーにデータファイルを取得するセクションを追加します。

```
---
- name: Retrieve data from vault
  hosts: ipaserver
  become: no
  gather_facts: false
  tasks:
[...]
  - name: Retrieve data file
    fetch:
      src: private-key-to-an-externally-signed-certificate.pem
      dest: ./
      flat: yes
      mode: 0600
```

8. インベントリーファイルの **webservers** セクションに記載されている Web サーバーに、Ansible コントローラーから取得した **private-key-to-an-externally-signed-certificate.pem** ファイルを転送するセクションを Playbook に追加します。

```
---
- name: Send data file to webservers
  become: no
  gather_facts: no
  hosts: webservers
  tasks:
  - name: Send data to webservers
    copy:
```

```
    src: private-key-to-an-externally-signed-certificate.pem
    dest: /etc/pki/tls/private/httpd.key
    mode: 0444
```

9. ファイルを保存します。
10. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file retrieve-data-asymmetric-vault-copy.yml
```

85.5. シークレットが漏洩した場合の ANSIBLE での IDM サービス VAULT シークレットの変更

以下の手順に従って、サービスインスタンスが危険にさらされたときに Ansible Playbook を再利用し、サービス vault に保存されているシークレットを変更します。以下の例のシナリオでは、シークレットを保存する非対称 vault への鍵は漏洩されておらず、**webserver3.idm.example.com** で取得したシークレットの情報が漏洩した場合を前提としています。この例では、管理者は 非対称 vault でシークレットを保存する時 および IdM ホストに非対称 vault からシークレットを取得する時 に Ansible Playbook を再利用します。この手順のはじめに、IdM 管理者は新規シークレットを含む新しい **PEM** ファイルを非対称 vault に保存し、不正アクセスのあった Web サーバー (**webserver3.idm.example.com**) に新しいシークレットを配置しないようにインベントリーファイルを調節し、もう一度この 2 つの手順を実行します。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- **IdM 管理者** パスワードが分かっている。
- サービスシークレットの保存先の 非対称 vault を作成している。
- IdM ホストで実行中の Web サービスの **httpd** 鍵を新たに生成し、不正アクセスのあった以前の鍵を置き換えている。
- 新しい **httpd** キーが **/usr/share/doc/ansible-freeipa/playbooks/vault/private-key-to-an-externally-signed-certificate.pem** ファイルなど、Ansible コントローラーのローカルに保存されている。

手順

1. **/usr/share/doc/ansible-freeipa/playbooks/vault** ディレクトリーに移動します。

```
$ cd /usr/share/doc/ansible-freeipa/playbooks/vault
```

2. インベントリーファイルを開き、以下のホストが正しく定義されていることを確認します。

   - **[ipaserver]** セクションの IdM サーバー
   - **[webservers]** セクションのシークレット取得先のホスト。たとえば Ansible に対して **webserver1.idm.example.com** と **webserver2.idm.example.com** にシークレットを取得するように指示するには、以下を入力します。

```
[ipaserver]
server.idm.example.com

[webservers]
webserver1.idm.example.com
webserver2.idm.example.com
```

重要

このリストに不正アクセスのあった Web サーバーが含まれないようにしてください (現在の例では **webserver3.idm.example.com**)。

3. **data-archive-in-asymmetric-vault-copy.yml** ファイルを開いて編集します。
4. ファイルを変更するには、**ipavault** タスクセクションに以下の変数を設定します。

   - **ipaadmin_password** 変数は IdM 管理者パスワードに設定します。
   - **name** 変数は vault の名前 (例: **secret_vault**) に設定します。
   - **service** 変数は vault のサービス所有者に設定します (例: **HTTP/webserver.idm.example.com**)。
   - **in** 変数は "**{{ lookup('file', 'new-private-key-to-an-externally-signed-certificate.pem')| b64encode }}**" に設定します。この設定では、Ansible が IdM サーバーではなく、Ansible コントローラーの作業ディレクトリーから秘密鍵のあるファイルを取得するようになります。
   - **action** 変数は **member** に設定します。
     今回の例で使用するように変更した Ansible Playbook ファイル:

```
---
- name: Tests
  hosts: ipaserver
  gather_facts: false

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - ipavault:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: secret_vault
      service: HTTP/webserver.idm.example.com
```

```
    in: "{{ lookup('file', 'new-private-key-to-an-externally-signed-certificate.pem') | b64encode
}}"
    action: member
```

5. ファイルを保存します。
6. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file data-
archive-in-asymmetric-vault-copy.yml
```

7. **retrieve-data-asymmetric-vault-copy.yml** ファイルを開いて編集します。
8. ファイルを変更するには、**ipavault** タスクセクションに以下の変数を設定します。
   - **ipaadmin_password** 変数は IdM 管理者パスワードに設定します。
   - **name** 変数は vault の名前 (例: **secret_vault**) に設定します。
   - **service** 変数は vault のサービス所有者に設定します (例: **HTTP/webserver1.idm.example.com**)。
   - **private_key_file** 変数は、サービス vault のシークレットの取得に使用する秘密鍵の場所に設定します。
   - **out** 変数は、現在の作業ディレクトリーなど、**new-private-key-to-an-externally-signed-certificate.pem** シークレットの取得先となる IdM サーバーの場所に設定します。
   - **action** 変数は **member** に設定します。
     今回の例で使用するように変更した Ansible Playbook ファイル:

```
---
- name: Retrieve data from vault
  hosts: ipaserver
  become: no
  gather_facts: false

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Retrieve data from the service vault
    ipavault:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: secret_vault
      service: HTTP/webserver1.idm.example.com
      vault_type: asymmetric
      private_key: "{{ lookup('file', 'service-private.pem') | b64encode }}"
      out: new-private-key-to-an-externally-signed-certificate.pem
      state: retrieved
```

9. Playbook に、IdM サーバーから Ansible コントローラーにデータファイルを取得するセクションを追加します。

```
---
- name: Retrieve data from vault
```

```
  hosts: ipaserver
  become: true
  gather_facts: false
  tasks:
[...]
  - name: Retrieve data file
    fetch:
      src: new-private-key-to-an-externally-signed-certificate.pem
      dest: ./
      flat: yes
      mode: 0600
```

10. インベントリーファイルの **webservers** セクションに記載されている Web サーバーに、Ansible コントローラーから取得した **new-private-key-to-an-externally-signed-certificate.pem** ファイルを転送するセクションを Playbook に追加します。

```
---
- name: Send data file to webservers
  become: true
  gather_facts: no
  hosts: webservers
  tasks:
  - name: Send data to webservers
    copy:
      src: new-private-key-to-an-externally-signed-certificate.pem
      dest: /etc/pki/tls/private/httpd.key
      mode: 0444
```

11. ファイルを保存します。

12. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file retrieve-data-asymmetric-vault-copy.yml
```

85.6. 関連情報

- **/usr/share/doc/ansible-freeipa/** ディレクトリーの README-vault.md Markdown ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/vault/** ディレクトリーのサンプルの Playbook を参照してください。

第86章 ANSIBLE を使用して IDM にサービスを配置させる手順およびさせない手順

Ansible **サービス** モジュールを使用すると、Identity Management (IdM) 管理者は、IdM ネイティブではない特定のサービスが IdM に存在するか、存在しないかを確認できます。たとえば、**サービス** モジュールを使用すると以下を行うことができます。

- 手動でインストールしたサービスが IdM クライアントに存在していることを確認します。サービスが存在しない場合には自動的にインストールされます。詳細は、次を参照してください。
  - IdM クライアントの IdM で HTTP サービスが存在することを確認する。
  - 単一の Ansible タスクを使用して、IdM クライアント上の IdM に複数のサービスが存在することを確認する。
  - IdM 以外のクライアントの IdM で HTTP サービスが存在することを確認する。
  - DNS のない IdM クライアントに HTTP サービスが存在することを確認する。
- IdM に登録したサービスに、証明書がアタッチされていることを確認して、証明書がない場合には自動的にその証明書がインストールされます。詳細は、次を参照してください。
  - IdM サービスエントリーで外部署名の証明書が存在することを確認する。
- IdM ユーザーとホストが、サービスキータブを取得して作成できるようにします。詳細は、次を参照してください。
  - IdM ユーザー、グループ、ホスト、またはホストグループがサービスのキータブを作成できるようにする。
  - IdM ユーザー、グループ、ホスト、またはホストグループがサービスのキータブを取得できるようにする。
- IdM ユーザーとホストが Kerberos エイリアスをサービスに追加できるようにします。詳細は、次を参照してください。
  - サービスの Kerberos プリンシパルエイリアスを存在させる手順
- サービスが IdM クライアントにないかを確認して、そのサービスが存在する場合は自動的にそのサービスを削除します。詳細は、次を参照してください。
  - IdM クライアントの IdM で HTTP サービスがないことを確認する。

86.1. ANSIBLE PLAYBOOK を使用して IDM に HTTP サービスを存在させる手順

以下の手順に従って、Ansible Playbook を使用して IdM に HTTP サーバーが存在することを確認します。

前提条件

- HTTP サービスをホストするシステムが IdM クライアントである。
- IdM 管理者パスワードがある。

手順

1. **inventory.file** などのインベントリーファイルを作成します。

```
$ touch inventory.file
```

2. **inventory.file** を開き、**[ipaserver]** セクションに、設定する IdM サーバーを定義します。たとえば、Ansible に対して **server.idm.example.com** を設定するように指示するには、次のコマンドを実行します。

```
[ipaserver]
server.idm.example.com
```

3. Ansible Playbook ファイル (**/usr/share/doc/ansible-freeipa/playbooks/service/service-is-present.yml**) のコピーを作成します。以下に例を示します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/service/service-is-present.yml
/usr/share/doc/ansible-freeipa/playbooks/service/service-is-present-copy.yml
```

4. Ansible Playbook ファイル (**/usr/share/doc/ansible-freeipa/playbooks/service/service-is-present-copy.yml**) を開きます。

```
---
- name: Playbook to manage IPA service.
  hosts: ipaserver
  gather_facts: false

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  # Ensure service is present
  - ipaservice:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: HTTP/client.idm.example.com
```

5. ファイルを調整します。

   - **ipaadmin_password** 変数で定義されている IdM 管理者パスワードを変更します。
   - **ipaservice** タスクの **name** 変数で定義されているように、HTTP サービスが実行されている IdM クライアントの名前を変更します。

6. ファイルを保存し、終了します。

7. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file /usr/share/doc/ansible-
freeipa/playbooks/service/service-is-present-copy.yml
```

検証手順

1. 管理者として IdM Web UI にログインします。

2. **Identity** → **Services** に移動します。

Services リストに **HTTP/client.idm.example.com@IDM.EXAMPLE.COM** がある場合は、Ansible Playbook は IdM に正常に追加されています。

関連情報

- HTTP サーバーとブラウザークライアント間の通信を保護する場合は、Apache HTTP Server への TLS 暗号化の追加 を参照してください。
- HTTP サービスの証明書を要求する場合は、certmonger を使用したサービスの IdM 証明書の取得 に記載されている手順を参照してください。

86.2. 単一の ANSIBLE タスクを使用して、IDM クライアント上の IDM に複数のサービスが存在することを確認する

ansible-freeipa ipaservice モジュールを使用すると、単一の Ansible タスクで複数の Identity Management (IdM) サービスを追加、変更、削除できます。そのためには、**ipaservice** モジュールの **services** オプションを使用します。

services オプションを使用すると、特定のサービスにのみ適用される複数のサービス変数を指定することもできます。このサービスを **name** 変数で定義します。これは、**services** オプションの唯一の必須変数です。

この手順を完了して、単一タスクで IdM に **HTTP/client01.idm.example.com@IDM.EXAMPLE.COM** サービスと **ftp/client02.idm.example.com@IDM.EXAMPLE.COM** サービスが存在することを確認します。

前提条件

- コントロールノードでは、
  - Ansible バージョン 2.14 以降を使用している。
  - **ansible-freeipa** パッケージをインストールしている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成した。
  - RHEL 8.9 以降を使用している。
  - **ipaadmin_password** を **Secret.yml** Ansible Vault に保存している。

手順

1. 次の内容を含む Ansible Playbook ファイル **add-http-and-ftp-services.yml** を作成します。

```
---
- name: Playbook to add multiple services in a single task
  hosts: ipaserver
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml

  tasks:
  - name: Add HTTP and ftp services
```

```
  ipaservice:
    ipaadmin_password: "{{ ipaadmin_password }}"
    services:
    - name:  HTTP/client01.idm.example.com@IDM.EXAMPLE.COM
    - name: ftp/client02.idm.example.com@IDM.EXAMPLE.COM
```

2. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory add-http-and-ftp-services.yml
```

関連情報

- **ansible-freeipa** アップストリームドキュメントのサービスモジュール

86.3. ANSIBLE PLAYBOOK を使用して、IDM 以外のクライアントの IDM で HTTP サービスを存在させる手順

以下の手順に従って、Ansible Playbook を使用して IdM クライアントではないホストの IdM に HTTP サーバーが存在することを確認します。HTTP サーバーを IdM に追加して、ホストを IdM に追加することもできます。

前提条件

- ホストに HTTP サービスをインストール している。
- HTTP を設定したホストが IdM クライアントではない。それ以外の場合は、Ansible Playbook を使用して IdM でに HTTP サービスを存在させる手順 に従います。
- IdM 管理者パスワードがある。
- ホスト向けの DNS A レコード (または IPv6 を使用している場合は AAAA レコード) がある。

手順

1. **inventory.file** などのインベントリーファイルを作成します。

```
$ touch inventory.file
```

2. **inventory.file** を開き、**[ipaserver]** セクションに、設定する IdM サーバーを定義します。たとえば、Ansible に対して **server.idm.example.com** を設定するように指示するには、次のコマンドを実行します。

```
[ipaserver]
server.idm.example.com
```

3. Ansible Playbook ファイル (**/usr/share/doc/ansible-freeipa/playbooks/service/service-is-present-using-host-check.yml**) のコピーを作成します。以下に例を示します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/service/service-is-present-without-host-check.yml /usr/share/doc/ansible-freeipa/playbooks/service/service-is-present-without-host-check-copy.yml
```

4. コピーしたファイル (**/usr/share/doc/ansible-freeipa/playbooks/service/service-is-present-using-host-check-copy.yml**) を開きます。**ipaservice** タスクで **ipaadmin_password** と **name** 変数の場所を特定します。

```
---
- name: Playbook to manage IPA service.
  hosts: ipaserver
  gather_facts: false

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  # Ensure service is present
  - ipaservice:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: HTTP/www2.example.com
      skip_host_check: yes
```

5. ファイルを調整します。

   - **ipaadmin_password** 変数は IdM 管理者パスワードに設定します。
   - **name** 変数は、HTTP サービスが実行されているホストの名前に設定します。

6. ファイルを保存し、終了します。

7. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file /usr/share/doc/ansible-
freeipa/playbooks/service/service-is-present-without-host-check-copy.yml
```

検証手順

1. 管理者として IdM Web UI にログインします。
2. **Identity → Services** に移動します。

HTTP/client.idm.example.com@IDM.EXAMPLE.COM が **Services** リストに表示されています。

関連情報

- 通信の安全を確保する方法は、Apache HTTP Server への TLS 暗号化の追加 を参照してください。

86.4. ANSIBLE PLAYBOOK を使用して、DNS を使用せずに IDM クライアントで HTTP サービスを存在させる手順

以下の手順に従って、Ansible Playbook を使用して DNS エントリーがない IdM クライアントで HTTP サーバーが存在することを確認します。このシナリオでは、IdM ホストに DNS A エントリーがないことを前提としています (IPv4 の代わりに IPv6 を使用する場合には DNS AAAA エントリーがない)。

前提条件

- HTTP サービスをホストするシステムが IdM に登録されている。
- ホストの DNS A または DNS AAAA レコードを存在させない。ホストの DNS レコードが存在する場合には、ホストの DNS レコードが存在する場合は、Ansible Playbook を使用して IdM に HTTP サービスを存在させる手順 に従うようにしてください。
- IdM 管理者パスワードがある。

手順

1. **inventory.file** などのインベントリーファイルを作成します。

```
$ touch inventory.file
```

2. **inventory.file** を開き、**[ipaserver]** セクションに、設定する IdM サーバーを定義します。たとえば、Ansible に対して **server.idm.example.com** を設定するように指示するには、次のコマンドを実行します。

```
[ipaserver]
server.idm.example.com
```

3. Ansible Playbook ファイル (**/usr/share/doc/ansible-freeipa/playbooks/service/service-is-present-with-host-force.yml**) のコピーを作成します。以下に例を示します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/service/service-is-present-with-host-force.yml /usr/share/doc/ansible-freeipa/playbooks/service/service-is-present-with-host-force-copy.yml
```

4. コピーしたファイル **/usr/share/doc/ansible-freeipa/playbooks/service/service-is-present-with-host-force-copy.yml** を開きます。**ipaservice** タスクで **ipaadmin_password** と **name** 変数の場所を特定します。

```
---
- name: Playbook to manage IPA service.
  hosts: ipaserver
  gather_facts: false

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  # Ensure service is present
  - ipaservice:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: HTTP/ihavenodns.info
      force: yes
```

5. ファイルを調整します。
   - **ipaadmin_password** 変数は IdM 管理者パスワードに設定します。
   - **name** 変数は、HTTP サービスが実行されているホストの名前に設定します。
6. ファイルを保存し、終了します。

7. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file /usr/share/doc/ansible-
freeipa/playbooks/service/service-is-present-with-host-force-copy.yml
```

検証手順

1. 管理者として IdM Web UI にログインします。
2. **Identity → Services** に移動します。

HTTP/client.idm.example.com@IDM.EXAMPLE.COM が Services リストに表示されています。

関連情報

- 通信の安全を確保する方法は、Apache HTTP Server への TLS 暗号化の追加 を参照してください。

86.5. ANSIBLE PLAYBOOK を使用して IDM サービスエントリーに外部署名証明書を存在させる手順

この手順に従い、**ansible-freeipa service** モジュールを使用して、外部の認証局 (CA) が HTTP サービスの IdM エントリーにアタッチされていることを確認します。IdM CA が自己署名の証明書を使用する場合には、IdM CA ではなく外部 CA が署名した HTTP サービスの証明書を使用すると便利です。

前提条件

- ホストに HTTP サービスをインストール している。
- HTTP サービスを IdM に登録している。
- IdM 管理者パスワードがある。
- 発行元が HTTP サービスのプリンシパルに対応する外部署名証明書がある。

手順

1. **inventory.file** などのインベントリーファイルを作成します。

```
$ touch inventory.file
```

2. **inventory.file** を開き、**[ipaserver]** セクションに、設定する IdM サーバーを定義します。たとえば、Ansible に対して **server.idm.example.com** を設定するように指示するには、次のコマンドを実行します。

```
[ipaserver]
server.idm.example.com
```

3. 以下のように **/usr/share/doc/ansible-freeipa/playbooks/service/service-member-certificate-present.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/service/service-member-certificate-present.yml /usr/share/doc/ansible-freeipa/playbooks/service/service-member-certificate-present-copy.yml
```

4. オプション: 証明書の形式が Privacy Enhanced Mail (PEM) の場合には、証明書を識別名エンコーディングルール (DER) 形式に変換し、コマンドラインインターフェイス (CLI) でより簡単に処理できるようにします。

```
$ openssl x509 -outform der -in cert1.pem -out cert1.der
```

5. **base64** を使用して **DER** ファイルを復号化します。ラッピングを無効にするには、**-w0** オプションを使用します。

```
$ base64 cert1.der -w0
MIIC/zCCAeegAwIBAgIUV74O+4kXeg21o4vxfRRtyJm...
```

6. 標準出力からクリップボードに証明書をコピーします。

7. **/usr/share/doc/ansible-freeipa/playbooks/service/service-member-certificate-present-copy.yml** ファイルを開き、このファイルの内容を編集および表示します。

```
---
- name: Service certificate present.
  hosts: ipaserver
  gather_facts: false

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  # Ensure service certificate is present
  - ipaservice:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: HTTP/client.idm.example.com
      certificate: |
        - MIICBjCCAW8CFHnm32VcXaUDGfEGdDL/...
      [...]
      action: member
      state: present
```

8. ファイルを調整します。

   - **certificate** 変数を使用して定義した証明書は、CLI からコピーした証明書に置き換えます。上記のように | パイプ文字と **certificate:** 変数を併用する場合は、1 行に入力せずに、このように証明書を入力してください。これで、証明書の読み取りが容易になります。

   - **ipaadmin_password** 変数で定義されている IdM 管理者パスワードを変更します。

   - HTTP サービスを実行する IdM クライアントの名前 (**name** 変数で定義) を変更します。

   - その他の関連する変数を変更します。

9. ファイルを保存し、終了します。

10. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file /usr/share/doc/ansible-
freeipa/playbooks/service/service-member-certificate-present-copy.yml
```

検証手順

1. 管理者として IdM Web UI にログインします。
2. **Identity → Services** に移動します。
3. **HTTP/client.idm.example.com** など、新しく追加した証明書が指定されたサービス名をクリックします。

右側の **サービス証明書** セクションで、新たに追加した証明書を確認できるようになりました。

86.6. ANSIBLE PLAYBOOK を使用して IDM ユーザー、グループ、ホスト、またはホストグループでサービスのキータブを作成できるようにする手順

キータブは、Kerberos プリンシパルと暗号化鍵のペアを含むファイルです。キータブファイルは通常、人の介入なしで、またはプレーンテキストファイルに保存されたパスワードを使用せずに、スクリプトで自動的に Kerberos で認証を行えるようにするために使用します。その後、スクリプトは取得した認証情報を使用してリモートシステムに保存されているファイルにアクセスできます。

Identity Management (IdM) 管理者は、他のユーザーが、IdM で実行しているサービスのキータブを取得したり、作成したりできるようにします。特定のユーザーおよびユーザーグループがキータブを作成できるようにすると、IdM 管理者パスワードを共有せずにサービスの管理を委譲できます。このように委譲することで、システムを詳細にわたり管理できます。

以下の手順に従って、特定の IdM ユーザー、ユーザーグループ、ホスト、およびホストグループが、IdM クライアントで実行している HTTP サービスのキータブを作成できるようにします。具体的には、IdM ユーザー **user01** が、**client.idm.example.com** という名前の IdM クライアントで実行中の HTTP サービスのキータブを作成できるように設定する方法を説明します。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- HTTP サービスを IdM に登録している。

- HTTP サービスをホストするシステムが IdM クライアントである。
- キータブの作成を許可する IdM ユーザーおよびユーザーグループが IdM に存在する。
- キータブの作成を許可する IdM ホストおよびホストグループが IdM に存在する。

手順

1. **inventory.file** などのインベントリーファイルを作成します。

```
$ touch inventory.file
```

2. **inventory.file** を開き、**[ipaserver]** セクションに、設定する IdM サーバーを定義します。たとえば、Ansible に対して **server.idm.example.com** を設定するように指示するには、次のコマンドを実行します。

```
[ipaserver]
server.idm.example.com
```

3. Ansible Playbook ファイル (**/usr/share/doc/ansible-freeipa/playbooks/service/service-member-allow_create_keytab-present.yml**) のコピーを作成します。以下に例を示します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/service/service-member-allow_create_keytab-present.yml /usr/share/doc/ansible-freeipa/playbooks/service/service-member-allow_create_keytab-present-copy.yml
```

4. Ansible Playbook ファイル (**/usr/share/doc/ansible-freeipa/playbooks/service/service-member-allow_create_keytab-present-copy.yml**) を開きます。

5. 以下を変更してファイルを調整します。

   - **ipaadmin_password** 変数で指定した IdM 管理者パスワード。
   - HTTP サービスが実行されている IdM クライアントの名前。現在の例では、**HTTP/client.idm.example.com** です。
   - **allow_create_keytab_user:** セクションに記載されている IdM ユーザー名。現在の例では **user01** です。
   - **allow_create_keytab_group:** セクションに記載されている IdM ユーザーグループ名。
   - **allow_create_keytab_host:** セクションに記載されている IdM ホスト名。
   - **allow_create_keytab_hostgroup:** セクションに記載されている IdM ホストグループ名。
   - **tasks** セクションの **name** 変数で指定されているタスク名。
     現在の例に合わせて調節すると、コピーされたファイルは以下のようになります。

```
---
- name: Service member allow_create_keytab present
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
```

```
  - name: Service HTTP/client.idm.example.com members allow_create_keytab present for
user01
    ipaservice:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: HTTP/client.idm.example.com
      allow_create_keytab_user:
      - user01
      action: member
```

6. ファイルを保存します。

7. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file /usr/share/doc/ansible-
freeipa/playbooks/service/service-member-allow_create_keytab-present-copy.yml
```

検証手順

1. 特定の HTTP サービスのキータブを作成する権限のある IdM ユーザーで、IdM サーバーに SSH 接続します。

```
$ ssh user01@server.idm.example.com
Password:
```

2. **ipa-getkeytab** コマンドを使用して、HTTP サービスの新規キータブを生成します。

```
$ ipa-getkeytab -s server.idm.example.com -p HTTP/client.idm.example.com -k
/etc/httpd/conf/krb5.keytab
```

-s オプションは、キータブを生成するキー配布センター (KDC) サーバーを指定します。

-p オプションは、作成するキータブのプリンシパルを指定します。

-k オプションは、新規キーを追加するキータブファイルを指定します。ファイルが存在しない場合には、作成されます。

このコマンドでエラーが表示されない場合は、**user01** で、**HTTP/client.idm.example.com** のキータブが正常に作成されています。

86.7. ANSIBLE PLAYBOOK を使用して IDM ユーザー、グループ、ホスト、またはホストグループでサービスのキータブを取得できるようにする手順

キータブは、Kerberos プリンシパルと暗号化鍵のペアを含むファイルです。キータブファイルは通常、人の介入なしで、またはプレーンテキストファイルに保存されたパスワードを使用せずに、スクリプトで自動的に Kerberos で認証を行えるようにするために使用します。その後、スクリプトは取得した認証情報を使用してリモートシステムに保存されているファイルにアクセスできます。

IdM 管理者は、他のユーザーが、IdM で実行しているサービスのキータブを取得したり、作成したりできるようにします。

以下の手順に従って、特定の IdM ユーザー、ユーザーグループ、ホスト、およびホストグループが、

IdM クライアントで実行している HTTP サービスのキータブを取得できるようにします。具体的には IdM ユーザー **user01** が、**client.idm.example.com** で実行する HTTP サービスのキータブを取得できるように設定する方法を説明します。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- HTTP サービスを IdM に登録している。
- キータブの取得を許可する IdM ユーザーおよびグループが IdM に存在する。
- キータブの取得を許可する IdM ホストおよびホストグループが IdM に存在する。

手順

1. **inventory.file** などのインベントリーファイルを作成します。

```
$ touch inventory.file
```

2. **inventory.file** を開き、**[ipaserver]** セクションに、設定する IdM サーバーを定義します。たとえば、Ansible に対して **server.idm.example.com** を設定するように指示するには、次のコマンドを実行します。

```
[ipaserver]
server.idm.example.com
```

3. Ansible Playbook ファイル (**/usr/share/doc/ansible-freeipa/playbooks/service/service-member-allow_retrieve_keytab-present.yml**) のコピーを作成します。以下に例を示します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/service/service-member-allow_retrieve_keytab-present.yml /usr/share/doc/ansible-freeipa/playbooks/service/service-member-allow_retrieve_keytab-present-copy.yml
```

4. コピーしたファイル **/usr/share/doc/ansible-freeipa/playbooks/service/service-member-allow_retrieve_keytab-present-copy.yml** を開きます。
5. ファイルを調整します。
   - **ipaadmin_password** 変数は IdM 管理者パスワードに設定します。

- **ipaservice** タスクの **name** 変数は、HTTP サービスのプリンシパルに設定します。現在の例では、**HTTP/client.idm.example.com** です。
- **allow_retrieve_keytab_group:** セクションで IdM ユーザーの名前を指定します。現在の例では **user01** です。
- **allow_retrieve_keytab_group:** セクションで、IdM ユーザーグループの名前を指定します。
- **allow_retrieve_keytab_group:** セクションで IdM ホストの名前を指定します。
- **allow_retrieve_keytab_group:** セクションで IdM ホストグループの名前を指定します。
- **tasks** セクションの **name** 変数を使用して、タスク名を指定します。
現在の例に合わせて調節すると、コピーされたファイルは以下のようになります。

```
---
- name: Service member allow_retrieve_keytab present
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Service HTTP/client.idm.example.com members allow_retrieve_keytab present for
user01
    ipaservice:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: HTTP/client.idm.example.com
      allow_retrieve_keytab_user:
      - user01
      action: member
```

6. ファイルを保存します。
7. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file /usr/share/doc/ansible-
freeipa/playbooks/service/service-member-allow_retrieve_keytab-present-copy.yml
```

検証手順

1. HTTP サービスのキータブ取得権限がある IdM ユーザーとして、IdM サーバーに SSH 接続します。

```
$ ssh user01@server.idm.example.com
Password:
```

2. **-r** オプションを指定して **ipa-getkeytab** コマンドを使用してキータブを取得します。

```
$ ipa-getkeytab -r -s server.idm.example.com -p HTTP/client.idm.example.com -k
/etc/httpd/conf/krb5.keytab
```

-s オプションは、キータブの取得元となるキー配布センター (KDC) サーバーを指定します。

-p オプションは、キータブを取得するプリンシパルを指定します。

-k オプションは、取得した鍵を追加するキータブファイルを指定します。ファイルが存在しない場合には、作成されます。

このコマンドでエラーが表示されない場合は、**user01** で **HTTP/client.idm.example.com** のキータブが正常に取得されています。

86.8. ANSIBLE PLAYBOOK を使用してサービスの KERBEROS プリンシパルのエイリアスを存在させる手順

シナリオによっては、IdM ユーザー、ホストまたはサービスが Kerberos アプリケーションに対して Kerberos プリンシパルエイリアスを使用して認証できるように IdM 管理者が設定すると便利です。次のようなシナリオになります。

- ユーザー名の変更後に、ユーザーが以前のユーザー名と新しいユーザー名の両方でシステムにログインできるようにする。
- IdM Kerberos レルムがメールドメインと異なる場合でも、ユーザーはメールアドレスを使用してログインする必要がある。

以下の手順に従って、**client.idm.example.com** で実行される HTTP サービスの **HTTP/mycompany.idm.example.com** のプリンシパルエイリアスを作成します。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- HTTP サービスを設定 している。
- HTTP サービスを IdM に登録している。
- HTTP を設定しているホストが IdM クライアントである。

手順

1. **inventory.file** などのインベントリーファイルを作成します。

```
$ touch inventory.file
```

2. **inventory.file** を開き、**[ipaserver]** セクションに、設定する IdM サーバーを定義します。たとえば、Ansible に対して **server.idm.example.com** を設定するように指示するには、次のコマンドを実行します。

```
[ipaserver]
server.idm.example.com
```

3. Ansible Playbook ファイル (**/usr/share/doc/ansible-freeipa/playbooks/service/service-member-principal-present.yml**) のコピーを作成します。以下に例を示します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/service/service-member-principal-present.yml /usr/share/doc/ansible-freeipa/playbooks/service/service-member-principal-present-copy.yml
```

4. Ansible Playbook ファイル (**/usr/share/doc/ansible-freeipa/playbooks/service/service-member-principal-present-copy.yml**) を開きます。

5. 以下を変更してファイルを調整します。

   - **ipaadmin_password** 変数で指定した IdM 管理者パスワード。
   - **name** 変数で指定したサービス名。これは、サービスの標準プリンシパル名です。現在の例では **HTTP/client.idm.example.com** です。
   - **principal** 変数で指定した Keroberos プリンシパルエイリアス。これは、 **name** 変数で定義したサービスに追加するエイリアスです。現在の例では、**host/mycompany.idm.example.com** です。
   - **tasks** セクションの **name** 変数で指定されているタスク名。
   現在の例に合わせて調節すると、コピーされたファイルは以下のようになります。

```
---
- name: Service member principal present
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Service HTTP/client.idm.example.com member principals
host/mycompany.idm.exmaple.com present
    ipaservice:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: HTTP/client.idm.example.com
      principal:
        - host/mycompany.idm.example.com
      action: member
```

6. ファイルを保存します。

7. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i path_to_inventory_directory/inventory.file /usr/share/doc/ansible-freeipa/playbooks/service/service-member-principal-present-copy.yml
```

Playbook を実行すると、到達できないタスクが 0 件、失敗したタスクが 0 件になる場合には、**HTTP/client.idm.example.com** サービスの **host/mycompany.idm.example.com** Kerberos プリンシパルが正常に作成されています。

関連情報

- Managing Kerberos principal aliases for users, hosts, and services を参照してください。

86.9. ANSIBLE PLAYBOOK を使用して IDM に HTTP サービスを存在させないようにする手順

以下の手順に従って、IdM からサービスの登録を解除します。より具体的には、Ansible Playbook を使用して、IdM にある **HTTP/client.idm.example.com** という名前の HTTP サーバーを削除する方法を説明します。

前提条件

- IdM 管理者パスワードがある。

手順

1. **inventory.file** などのインベントリーファイルを作成します。

```
$ touch inventory.file
```

2. **inventory.file** を開き、**[ipaserver]** セクションに、設定する IdM サーバーを定義します。たとえば、Ansible に対して **server.idm.example.com** を設定するように指示するには、次のコマンドを実行します。

```
[ipaserver]
server.idm.example.com
```

3. Ansible Playbook ファイル (**/usr/share/doc/ansible-freeipa/playbooks/service/service-is-absent.yml**) のコピーを作成します。以下に例を示します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/service/service-is-absent.yml
/usr/share/doc/ansible-freeipa/playbooks/service/service-is-absent-copy.yml
```

4. Ansible Playbook ファイル (**/usr/share/doc/ansible-freeipa/playbooks/service/service-is-absent-copy.yml**) を開きます。

5. 以下を変更してファイルを調整します。

   - **ipaadmin_password** 変数で定義されている IdM 管理者パスワード。
   - **ipaservice** タスクの **name** 変数で定義されている、HTTP サービスの Kerberos プリンシパル。
     現在の例に合わせて調節すると、コピーされたファイルは以下のようになります。

```
---
- name: Playbook to manage IPA service.
  hosts: ipaserver
  gather_facts: false
```

```
vars_files:
- /home/user_name/MyPlaybooks/secret.yml
tasks:
# Ensure service is absent
- ipaservice:
    ipaadmin_password: "{{ ipaadmin_password }}"
    name: HTTP/client.idm.example.com
    state: absent
```

6. ファイルを保存し、終了します。

7. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file /usr/share/doc/ansible-
freeipa/playbooks/service/service-is-absent-copy.yml
```

検証手順

1. 管理者として IdM Web UI にログインします。

2. **Identity → Services** に移動します。

Services リストに **HTTP/client.idm.example.com@IDM.EXAMPLE.COM** サービスが表示されていない場合には、IdM から正常に削除されています。

86.10. 関連情報

- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-service.md** Markdown ファイルを参照してください。

- **/usr/share/doc/ansible-freeipa/playbooks/config** ディレクトリーのサンプルの Playbook を参照してください。

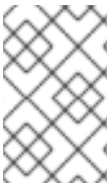
第87章 IDM を管理する AD ユーザーの有効化

87.1. AD ユーザーの ID のオーバーライド

Red Hat Enterprise Linux (RHEL) 7 では、System Security Services Daemon (SSSD) で外部グループメンバーシップを使用して、AD (Active Directory) ユーザーとグループが POSIX 環境の Identity Management (IdM) リソースにアクセスするのを許可します。

IdM LDAP サーバーには、アクセス制御を付与する独自のメカニズムがあります。RHEL 8 には、AD ユーザーに対する ID ユーザーのオーバーライドを、IdM グループのメンバーとして追加できるようにする更新が導入されました。ID オーバーライドは、特定の Active Directory ユーザーまたはグループのプロパティーが特定の ID ビュー (この場合は **Default Trust View**) 内でどのように見えるかを記述するレコードです。この更新により、IdM LDAP サーバーは、IdM グループのアクセス制御ルールを AD ユーザーに適用できます。

AD ユーザーは、IdM UI のセルフサービス機能 (SSH キーのアップロード、個人のデータの変更など) を使用できるようになりました。AD 管理者は、アカウントおよびパスワードを 2 つ使用しなくても、IdM を完全に管理できるようになります。



注記

IdM の一部の機能は、AD ユーザーには現在利用できません。たとえば、IdM の **admins** グループに所属する AD ユーザーが、IdM ユーザーのパスワードを設定することはできません。

重要

IdM の **sudo** ルールに AD ユーザーの ID オーバーライドを使用 **しない** でください。AD ユーザーの ID オーバーライドは、AD ユーザー自体ではなく、AD ユーザーの POSIX 属性のみを表します。

関連情報

- Active Directory ユーザーの ID ビューの使用

87.2. IDM を管理する AD ユーザーを有効にする ID オーバーライドの使用

AD ユーザーの ID オーバーライドを作成して使用し、そのユーザーに IdM ユーザーと同じ権限を付与するには、次の手順に従います。この手順は、信頼コントローラーまたは信頼エージェントとして設定した IdM サーバーで行います。

前提条件

- Identity Management (IdM) サーバーで **idm:DL1** ストリームが有効になっていて、このストリームで配信される RPM に切り替えている。

```
# yum module enable idm:DL1
# yum distro-sync
```

- **idm:DL1/adtrust** プロファイルが IdM サーバーにインストールされている。

```
# yum module install idm:DL1/adtrust
```

プロファイルには、Active Directory (AD) との信頼関係を持つ IdM サーバーのインストールに必要なすべてのパッケージが含まれています。

- 作業用の IdM 環境が設定されている。詳細は、Identity Management のインストール を参照してください。
- IdM 環境と AD との間の信頼関係が設定されて動作している。

手順

1. IdM 管理者として、**Default Trust View** で AD ユーザーの ID オーバーライドを作成します。たとえば、ユーザー **ad_user@ad.example.com** の ID オーバーライドを作成するには、次のコマンドを実行します。

```
# kinit admin
# ipa idoverrideuser-add 'default trust view' ad_user@ad.example.com
```

2. **Default Trust View** から IdM グループのメンバーとして ID オーバーライドを追加します。これは、Active Directory と対話するため、POSIX 以外のグループである必要があります。
対象のグループが IdM ロールのメンバーである場合、ID オーバーライドによって表される AD ユーザーは、IdM API (コマンドラインインターフェイス、IdM の Web UI など) の使用時に、ロールにより付与されたすべての権限を取得します。

   たとえば、**ad_user@ad.example.com** ユーザーの ID オーバーライドを IdM **admins** グループに追加するには、次のコマンドを実行します。

```
# ipa group-add-member admins --idoverrideusers=ad_user@ad.example.com
```

3. または、**User Administrator** ロールなどのロールに ID オーバーライドを追加することもできます。

```
# ipa role-add-member 'User Administrator' --
idoverrideusers=ad_user@ad.example.com
```

関連情報

- Active Directory ユーザーの ID ビューの使用

87.3. ANSIBLE を使用して AD ユーザーが IDM を管理できるようにする手順

Ansible Playbook を使用してユーザー ID オーバーライドが Identity Management (IdM) グループに存在することを確認するには、次の手順に従います。ユーザー ID オーバーライドは、Active Directory (AD) とのトラストを確立した後に、デフォルトの Trust View で作成した AD ユーザーのオーバーライドです。Playbook を実行すると、AD 管理者などの AD ユーザーは、2 つの異なるアカウントとパスワードを持たなくても IdM を完全に管理できるようになります。

前提条件

- IdM **admin** のパスワードを把握している。
- AD とのトラストをインストール している。

- IdM に AD ユーザーのユーザー ID オーバーライドがすでに存在する。存在しない場合は、**ipa idoverrideuser-add 'default trust view' ad_user@ad.example.com** コマンドで作成します。
- ユーザー ID オーバーライドを追加しようとしているグループが、IdM にすでに存在する 。
- IdM 4.8.7 バージョン以降の IdM を使用している。サーバーにインストールされている IdM のバージョンを表示するには、**ipa --version** を実行します。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **~/MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. 次の内容で **add-useridoverride-to-group.yml** playbook を作成します。

```
---
- name: Playbook to ensure presence of users in a group
  hosts: ipaserver


  - name: Ensure the ad_user@ad.example.com user ID override is a member of the admins
group:
    ipagroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: admins
      idoverrideuser:
      - ad_user@ad.example.com
```

上記の例では、以下のようになります。

- Secret123 は IdM **admin** パスワードです。
- **admins** は、**ad_user@ad.example.com** ID オーバーライドを追加する IdM POSIX グループの名前です。このグループのメンバーには、完全な管理者権限があります。
- **ad_user@ad.example.com** は、AD 管理者のユーザー ID オーバーライドです。ユーザーは、信頼が確立された AD ドメインに保存されます。

3. ファイルを保存します。

4. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory add-useridoverride-to-group.yml
```

関連情報

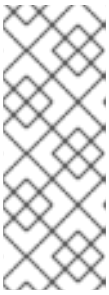
- AD ユーザーの ID のオーバーライド
- /usr/share/doc/ansible-freeipa/README-group.md
- /usr/share/doc/ansible-freeipa/playbooks/user
- Using ID views in Active Directory environments

87.4. AD ユーザーが IDM CLI で正しいコマンドを実行できることの確認

Active Directory (AD) ユーザーが Identity Management (IdM) コマンドラインインターフェイス (CLI) にログインし、自分のロールに適したコマンドを実行できることを確認します。

1. IdM 管理者の、現在の Kerberos チケットを破棄します。

```
# kdestroy -A
```



注記

MIT Kerberos の GSSAPI 実装が優先的にターゲットサービスの領域 (この場合は IdM レルム) から認証情報を選択するため、Kerberos チケットの破棄が必要です。これは、認証情報のキャッシュコレクションを意味します。つまり、タイプが **KCM:**、**KEYRING:**、または **DIR:** の認証情報キャッシュが使用されている場合、AD ユーザーの認証情報の代わりに、以前に取得した **admin** またはその他の IdM プリンシパルの認証情報が、IdM API にアクセスするために使用されます。

2. ID オーバーライドが作成された AD ユーザーの Kerberos 認証情報を入手します。

```
# kinit ad_user@AD.EXAMPLE.COM
Password for ad_user@AD.EXAMPLE.COM:
```

3. AD ユーザーの ID オーバーライド使用する、IdM グループのメンバーシップから生じるパーミッションが、そのグループ内の任意の IdM ユーザーと同じものであることをテストします。AD ユーザーの ID オーバーライドが **admins** グループに追加されている場合、AD ユーザーは、たとえば IdM にグループを作成できます。

```
# ipa group-add some-new-group
---------------------------
Added group "some-new-group"
---------------------------
  Group name: some-new-group
  GID: 1997000011
```

87.5. ANSIBLE を使用して AD ユーザーが IDM を管理できるようにする

ansible-freeipa idoverrideuser および **group** モジュールを使用して、信頼された AD ドメインの Active Directory (AD) ユーザーのユーザー ID オーバーライドを作成し、そのユーザーに IdM ユーザーと同じ権限を付与することができます。この手順では、最初の Playbook タスクで **administrator@addomain.com** ID オーバーライドが追加された **デフォルトの信頼ビュー** ID ビューの例を使用します。次の Playbook タスクでは、administrator@addomain.com ID オーバーライドが IdM **管理者** グループにメンバーとして追加されます。その結果、AD 管理者は 2 つの異なるアカウントとパスワードを持たずに IdM を管理できるようになります。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - RHEL 8.10 以降を使用しています。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- AD フォレストは IdM と信頼関係にあります。この例では、AD ドメインの名前は **addomain.com** であり、AD 管理者の完全修飾ドメイン名 (FQDN) は **administrator@addomain.com** です。
- インベントリーファイル内の **ipaserver** ホストは、信頼コントローラーまたは信頼エージェントとして設定されています。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. Ansible コントロールノードで、**administrator@addomain.com** ユーザーオーバーライドをデフォルトの信頼ビューに追加するタスクを含む **enable-ad-admin-to-administer-idm.yml** Playbook を作成します。

```
---
- name: Enable AD administrator to act as a FreeIPA admin
  hosts: ipaserver
  become: false
  gather_facts: false

  tasks:
  - name: Ensure idoverride for administrator@addomain.com in 'default trust view'
    ipaidoverrideuser:
      ipaadmin_password: "{{ ipaadmin_password }}"
      idview: "Default Trust View"
      anchor: administrator@addomain.com
```

2. 同じ Playbook 内の別の Playbook タスクを使用して、AD 管理者ユーザー ID オーバーライドを **admins** グループに追加します。

```
- name: Add the AD administrator as a member of admins
  ipagroup:
    ipaadmin_password: "{{ ipaadmin_password }}"
    name: admins
    idoverrideuser:
    - administrator@addomain.com
```

3. ファイルを保存します。

4. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory enable-ad-admin-to-administer-idm.yml
```

検証

1. AD 管理者として IdM クライアントにログインします。

```
$ ssh administrator@addomain.com@client.idm.example.com
```

2. 有効なチケット保証チケット (TGT) を取得したことを確認します。

```
$ klist
Ticket cache: KCM:325600500:99540
Default principal: Administrator@ADDOMAIN.COM
Valid starting Expires Service principal
02/04/2024 11:54:16 02/04/2024 21:54:16 krbtgt/ADDOMAIN.COM@ADDOMAIN.COM
renew until 02/05/2024 11:54:16
```

3. IdM で **管理者** 権限を確認します。

```
$ ipa user-add testuser --first=test --last=user
----------------------
Added user "tuser"
----------------------
  User login: tuser
  First name: test
  Last name: user
  Full name: test user
[...]
```

関連情報

- idoverrideuser および ipagroup **ansible-freeipa** アップストリームドキュメント
- IdM を管理する AD ユーザーの有効化

第88章 ドメイン解決順序を設定して AD ユーザーの短縮名を解決する手順

デフォルトでは、**user_name@domain.com** または **domain.com\user_name** の形式で完全修飾名を指定して、Active Directory (AD) 環境からユーザーおよびグループを解決し、認証する必要があります。以下のセクションでは、AD ユーザーおよびグループの短縮名を解決するように IdM サーバーおよびクライアントを設定する方法を説明します。

- ドメイン解決順序の仕組み
- IdM サーバーでのグローバルドメイン解決順序設定
- IdM サーバーの ID ビューのドメイン解決順序設定
- Ansible を使用してドメイン解決順序を持つ ID ビューを作成する
- IdM クライアントの SSSD でのドメイン解決順序設定

88.1. ドメイン解決順序の仕組み

Red Hat では、Active Directory (AD) の信頼を使用する Identity Management (IdM) 環境では、完全修飾名を指定してユーザーおよびグループを解決し、認証することを推奨します。以下に例を示します。

- **idm.example.com** ドメインからの IdM ユーザーの場合は **<idm_username>@idm.example.com**
- **ad.example.com** ドメインからの AD ユーザーの場合は **<ad_username>@ad.example.com**

デフォルトでは、**ad_username** などの **短縮名** 形式を使用してユーザーまたはグループの検索を実行すると、IdM は IdM ドメインのみを検索する場合には、AD ユーザーまたはグループの検索に失敗します。短縮名を使用して AD ユーザーまたはグループを解決するには、**ドメイン解決順序** オプションを設定して、IdM が複数のドメインを検索する順番を変更します。

IdM データベースまたは個々のクライアントの SSSD 設定で、ドメイン解決の順序を一元的に設定できます。IdM は、以下の優先順位でドメイン解決の順序を評価します。

- ローカルの **/etc/sssd/sssd.conf** 設定。
- ID ビューの設定。
- グローバル IdM 設定。

備考

- ホストの SSSD 設定に **default_domain_suffix** オプションが含まれ、このオプションを指定せずにドメインへの要求を行う場合は、完全修飾ユーザー名を使用する必要があります。
- **ドメイン解決順序** オプションを使用して **compat** ツリーをクエリーすると、複数のユーザー ID (UID) が返される可能性があります。これで問題がある場合には、Pagure バグレポート Inconsistent compat user objects for AD users when domain resolution order is set を参照してください。

重要

IdM クライアントまたは IdM サーバーで **full_name_format** SSSD オプションは使用しないでください。このオプションにデフォルト値以外の値を使用すると、ユーザー名が表示される方法が変更され、IdM 環境での検索が中断される可能性があります。

関連情報

- Active Directory Trust for Legacy Linux Clients .

88.2. IDM サーバーでのグローバルドメイン解決順序設定

この手順では、IdM ドメイン内の全クライアントにドメイン解決の順序を設定します。この例では、ドメインの解決順序を設定して、以下の順番でユーザーとグループを検索します。

1. Active Directory (AD) root ドメイン **ad.example.com**
2. AD 子ドメイン **subdomain1.ad.example.com**
3. IdM ドメイン **idm.example.com**

前提条件

- AD 環境で信頼を設定している。

手順

- **ipa config-mod --domain-resolution-order** コマンドを使用して、希望の順序で検索するドメインをリスト表示します。コロン (**:**) でドメインを区切ります。

```
[user@server ~]$ ipa config-mod --domain-resolution-
order='ad.example.com:subdomain1.ad.example.com:idm.example.com'
Maximum username length: 32
Home directory base: /home
...
  Domain Resolution Order:
ad.example.com:subdomain1.ad.example.com:idm.example.com
...
```

検証手順

- 短縮名だけを使用して、**ad.example.com** ドメインからユーザー情報を取得できることを確認します。

```
[root@client ~]# id <ad_username>
uid=1916901102(ad_username) gid=1916900513(domain users)
groups=1916900513(domain users)
```

88.3. IDM サーバーの ID ビューのドメイン解決順序設定

この手順では、IdM サーバーおよびクライアントの特定のセットに適用できるように、ID ビューのドメイン解決の順序を設定します。この例では、IdM ホスト **client1.idm.example.com** に **ADsubdomain1_first** という名前の ID ビューを作成して、ドメインの解決順序を設定し、以下の順番

でユーザーとグループを検索します。

1. Active Directory (AD) 子ドメイン **subdomain1.ad.example.com**
2. AD root ドメイン **ad.example.com**
3. IdM ドメイン **idm.example.com**

注記

ID ビューで設定したドメイン解決の順序は、グローバルなドメイン解決順序より優先されますが、SSSD 設定でローカルに設定されたドメイン解決順序をオーバーライドすることはありません。

前提条件

- AD 環境で信頼を設定している。

手順

1. **--domain-resolution-order** オプションを指定して ID ビューを作成します。

```
[user@server ~]$ ipa idview-add ADsubdomain1_first --desc "ID view for resolving AD
subdomain1 first on client1.idm.example.com" --domain-resolution-order
subdomain1.ad.example.com:ad.example.com:idm.example.com
---------------------------------
Added ID View "ADsubdomain1_first"
---------------------------------
ID View Name: ADsubdomain1_first
Description: ID view for resolving AD subdomain1 first on client1.idm.example.com
Domain Resolution Order:
subdomain1.ad.example.com:ad.example.com:idm.example.com
```

2. ID ビューを IdM ホストに適用します。

```
[user@server ~]$ ipa idview-apply ADsubdomain1_first --hosts
client1.idm.example.com
---------------------------------
Applied ID View "ADsubdomain1_first"
---------------------------------
  hosts: client1.idm.example.com
-----------------------------------------------
Number of hosts the ID View was applied to: 1
-----------------------------------------------
```

検証手順

- ID ビューの詳細を表示します。

```
[user@server ~]$ ipa idview-show ADsubdomain1_first --show-hosts
  ID View Name: ADsubdomain1_first
  Description: ID view for resolving AD subdomain1 first on client1.idm.example.com
```

```
  Hosts the view applies to: client1.idm.example.com
  Domain resolution order:
subdomain1.ad.example.com:ad.example.com:idm.example.com
```

- 短縮名だけを使用して、**subdomain1.ad.example.com** ドメインからユーザー情報を取得できることを確認します。

```
[root@client1 ~]# id <user_from_subdomain1>
uid=1916901106(user_from_subdomain1) gid=1916900513(domain users)
groups=1916900513(domain users)
```

88.4. ANSIBLE を使用してドメイン解決順序を持つ ID ビューを作成する

ansible-freeipa idview モジュールを使用して、アイデンティティー Management (IdM) デプロイメントで ID ビューを追加、変更、および削除できます。たとえば、ドメイン解決順序を持つ ID ビューを作成して、短い名前表記を有効にすることができます。

短縮名表記では、**aduser05@ad.example.com** などの Active Directory (AD) の完全なユーザー名が、短縮ログイン (この場合は **aduser05**) に置き換えられます。つまり、**SSH** を使用して IdM クライアントにログインする場合、**aduser05 はssh** aduser05@ad.example.com@client.idm.example.com ではなく **ssh** aduser05@client.idm.example.com と入力できます。**id** などの他のコマンドにも同じことが当てはまります。

Ansible を使用して次のことを行うには、この手順を完了します。

- 短い名前の修飾に使用されるコロンで区切られたドメインの文字列を定義します。この例では、文字列は **ad.example.com:idm.example.com** です。
- 文字列で識別される最初のドメインでユーザー名を最初に検索するように SSSD に指示する ID ビューを作成します。この例では、**ad.example.com** です。
- ID ビューを特定のホストに適用します。この例では、これは **testhost.idm.example.com** です。

注記

IdM クライアントには 1 つの ID ビューのみを適用できます。新しい ID ビューを適用すると、該当する場合は以前の ID ビューが自動的に削除されます。

前提条件

- コントロールノードでは、
  - Ansible バージョン 2.14 以降を使用している。
  - **ansible-freeipa** パッケージをインストールしている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成した。
  - RHEL 8.9 以降を使用している。
  - **ipaadmin_password** を **Secret.yml** Ansible Vault に保存している。
- **testhost.idm.example.com** は IdM クライアントです。

- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **~/MyPlaybooks/** ディレクトリーに移動し、次の内容を含む Ansible Playbook ファイル **add-id-view-with-domain-resolution-order.yml** を作成します。

```
---
- name: Playbook to add idview and apply it to an IdM client
  hosts: ipaserver
  vars_files:
  - /home/<user_name>/MyPlaybooks/secret.yml
  become: false
  gather_facts: false

  tasks:
  - name: Add idview and apply it to testhost.idm.example.com
    ipaidview:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: test_idview
      host: testhost.idm.example.com
      domain_resolution_order: "ad.example.com:ipa.example.com"
```

2. Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory add-id-view-with-domain-resolution-order.yml
```

検証

1. **testhost.idm.example.com** に SSH で接続します。

2. 短縮名だけを使用して、**ad.example.com** ドメインからユーザー情報を取得できることを確認します。

```
[root@testhost ~]# id aduser05
uid=1916901102(aduser05) gid=1916900513(domain users) groups=1916900513(domain users)
```

関連情報

- **ansible-freeipa** アップストリームドキュメントの idview モジュール

88.5. IDM クライアントの SSSD でのドメイン解決順序設定

この手順では、IdM クライアントの SSSD 設定でドメイン解決の順序を設定します。この例では、IdM ホスト **client2.idm.example.com** が、以下の順番でユーザーとグループを検索するように設定します。

1. Active Directory (AD) 子ドメイン **subdomain1.ad.example.com**

2. AD root ドメイン **ad.example.com**

3. IdM ドメイン **idm.example.com**

注記

ローカルの SSSD 設定のドメイン解決順序は、グローバルおよび ID ビュードメインの解決順序よりも優先されます。

前提条件

- AD 環境で信頼を設定している。

手順

1. テキストエディターで **/etc/sssd/sssd.conf** ファイルを開きます。
2. このファイルの **[sssd]** セクションに **domain_resolution_order** オプションを設定します。

```
domain_resolution_order = subdomain1.ad.example.com, ad.example.com,
idm.example.com
```

3. ファイルを保存してから閉じます。
4. SSSD サービスを再起動して、新しい設定を読み込みます。

```
[root@client2 ~]# systemctl restart sssd
```

検証手順

- 短縮名だけを使用して、**subdomain1.ad.example.com** ドメインからユーザー情報を取得できることを確認します。

```
[root@client2 ~]# id <user_from_subdomain1>
uid=1916901106(user_from_subdomain1) gid=1916900513(domain users)
groups=1916900513(domain users)
```

88.6. 関連情報

- ID ビューを使用した IdM クライアントのユーザー属性値をオーバーライドする

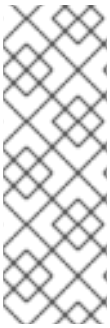
第89章 IDM での AD ユーザープリンシパル名を使用した認証の有効化

89.1. IDM で信頼される AD フォレストのユーザープリンシパル名

Identity Management (IdM) 管理者は、AD ユーザーが別の **ユーザープリンシパル名** (UPN) を使用して IdM ドメインのリソースにアクセスできるようにできます。UPN は、AD ユーザーが認証に使用する別のユーザーログインで、形式は **user_name@KERBEROS-REALM** です。AD フォレストでは、追加の Kerberos エイリアスと UPN 接尾辞の両方を設定することができるため、AD 管理者は **user_name** と **KERBEROS-REALM** の両方に別の値を設定できます。

たとえば、ある会社が **AD.EXAMPLE.COM** Kerberos レルムを使用する場合に、ユーザーのデフォルトの UPN は **user@ad.example.com** です。**user@example.com** などのメールアドレスを使用してユーザーがログインできるようにするには、AD で **EXAMPLE.COM** を別の UPN として設定できます。または、最近企業で合併が行われ、ログインの名前空間が統一して提供されるようにする場合に、UPN (**エンタープライズ UPN とも呼ばれる**) は特に便利です。

UPN 接尾辞は、AD フォレストルートで定義された場合に IdM にだけ表示されます。AD 管理者は、**Active Directory Domain and Trust** ユーティリティーまたは **PowerShell** コマンドラインツールで UPN を定義できます。



注記

Red Hat は、ユーザーへの UPN 接尾辞の設定には、**Active Directory Domain and Trust** ユーティリティーなどのエラー検証を実行するツールを使用することを推奨します。

Active Directory ではこのような操作は検証されないため、**ldapmodify** コマンドを使用してユーザーの **userPrincipalName** 属性を設定するなど、低レベルの変更で UPN を設定することを推奨します。

AD 側で新しい UPN を定義したら、IdM サーバーで **ipa trust-fetch-domains** コマンドを実行して、更新された UPN を取得します。AD UPN が IdM で最新であることを確認する手順 を参照してください。

IdM は、**cn=trusted_domain_name,cn=ad,cn=trusts,dc=idm,dc=example,dc=com** サブツリーの **ipaNTAdditionalSuffixes** 複数値の属性にドメインの UPN 接尾辞を保存します。

関連情報

- AD フォレストルートでの UPN 接尾辞設定の実装方法
- AD ユーザーエントリーを手動で修正して UPN 接尾辞の検証を省略する方法
- 信頼コントローラーおよび信頼エージェント

89.2. AD UPN が IDM で最新であることを確認する手順

信頼される Active Directory (AD) フォレストで、ユーザープリンシパル名 (UPN) 接尾辞を追加または削除してから、IdM サーバーで信頼されるフォレストの情報を更新します。

前提条件

- IdM 管理者の認証情報

手順

- **ipa trust-fetch-domains** コマンドを入力します。空白のような出力も想定範囲である点に注意してください。

```
[root@ipaserver ~]# ipa trust-fetch-domains
Realm-Name: ad.example.com
------------------------------
No new trust domains were found
------------------------------
---------------------------
Number of entries returned 0
---------------------------
```

検証手順

- **ipa trust-show** コマンドを入力し、サーバーが新しい UPN をフェッチしていることを確認します。プロンプトが表示されたら、AD レルムの名前を指定します。

```
[root@ipaserver ~]# ipa trust-show
Realm-Name: ad.example.com
  Realm-Name: ad.example.com
  Domain NetBIOS name: AD
  Domain Security Identifier: S-1-5-21-796215754-1239681026-23416912
  Trust direction: One-way trust
  Trust type: Active Directory domain
  UPN suffixes: example.com
```

この出力では、**example.com** UPN 接尾辞が **ad.example.com** レルムエントリーに含まれることが分かります。

89.3. AD UPN 認証問題のトラブルシューティングデータの収集

Active Directory (AD) 環境および IdM 環境からユーザープリンシパル名 (UPN) 設定に関するトラブルシューティングデータを収集するには、次の手順に従います。AD ユーザーが別の UPN を使用してログインできない場合は、こでの情報を使用してトラブルシューティング作業を絞り込むことができます。

前提条件

- AD ドメインコントローラーから情報を取得できるように IdM 信頼コントローラーまたは信頼エージェントにログインしておく。
- 以下の設定ファイルを変更し、IdM サービスを再起動できるように **root** 権限がある。

手順

1. テキストエディターで **/usr/share/ipa/smb.conf.empty** 設定ファイルを開きます。
2. 以下の内容をファイルに追加します。

```
[global]
log level = 10
```

3. **/usr/share/ipa/smb.conf.empty** ファイルを保存して閉じます。

4. テキストエディターで **/etc/ipa/server.conf** 設定ファイルを開きます。このファイルがない場合は作成します。

5. 以下の内容をファイルに追加します。

```
[global]
debug = True
```

6. **/etc/ipa/server.conf** ファイルを保存して終了します。

7. Apache Web サーバーのサービスを再起動して、設定の変更を適用します。

```
[root@server ~]# systemctl restart httpd
```

8. AD ドメインから信頼情報を取得します。

```
[root@server ~]# ipa trust-fetch-domains <ad.example.com>
```

9. 以下のログファイルでデバッグの出力とトラブルシューティング情報を確認します。

   - **/var/log/httpd/error_log**
   - **/var/log/samba/log.***

関連情報

- Using rpcclient to gather troubleshooting data for AD UPN authentication issues を参照してください。

第90章 IDM で標準 DNS ホスト名の使用

DNS 正規化は、潜在的なセキュリティーリスクを回避するために、Identity Management (IdM) クライアントでデフォルトで無効になっています。たとえば、攻撃者がドメインの DNS サーバーとホストを制御している場合、攻撃者は短いホスト名 (**demo** など) を、侵害されたホスト (**malicious.example.com** など) に解決させることができます。この場合、ユーザーは想定とは異なるサーバーに接続します。

この手順では、IdM クライアントで正規化されたホスト名を使用する方法について説明します。

90.1. ホストプリンシパルへのエイリアスの追加

デフォルトでは、**ipa-client-install** コマンドを使用して登録した Identity Management (IdM) クライアントでは、サービスプリンシパルで短縮ホスト名を使用することができません。たとえば、ユーザーがサービスにアクセスするときに、**host/demo@EXAMPLE.COM** ではなく、**host/demo.example.com@EXAMPLE.COM** のみを使用できます。

Kerberos プリンシパルにエイリアスを追加するには、次の手順に従います。または、**/etc/krb5.conf** ファイルでホスト名の正規化を有効にできます。詳細は、 クライアントのサービスプリンシパルでのホスト名の正規化の有効化 を参照してください。

前提条件

- IdM クライアントがインストールされている。
- ホスト名が、ネットワーク内で一意の名前である。

手順

1. **admin** ユーザーとして、IdM に対して認証します。

```
$ kinit admin
```

2. エイリアスをホストプリンシパルに追加します。たとえば、**demo** エイリアスを、**demo.examle.com** ホストプリンシパルに追加するには、次のコマンドを実行します。

```
$ ipa host-add-principal demo.example.com --principal=demo
```

90.2. クライアントのサービスプリンシパルでのホスト名の正規化の有効化

クライアント上のサービスプリンシパルのホスト名の正規化を有効にするには、次の手順に従います。

ホストプリンシパルへのエイリアスの追加 の説明に従って、ホストプリンシパルのエイリアスを使用する場合は、正規化を有効にする必要がないことに注意してください。

前提条件

- Identity Management (IdM) クライアントがインストールされている。
- **root** ユーザーとして IdM クライアントにログインしている。
- ホスト名が、ネットワーク内で一意の名前である。

手順

1. **/etc/krb5.conf** ファイルの **[libdefaults]** セクションで、**dns_canonicalize_hostname** パラメーターを **false** に設定します。

```
[libdefaults]
...
dns_canonicalize_hostname = true
```

90.3. DNS ホスト名の正規化を有効にしてホスト名を使用するためのオプション

クライアントのサービスプリンシパルでのホスト名の正規化の有効化 の説明に従って、**/etc/krb5.conf** ファイルに **dns_canonicalize_hostname = true** を設定すると、サービスプリンシパルでホスト名を使用する際に、以下のオプションがあります。

- Identity Management (IdM) 環境では、**host/demo.example.com@EXAMPLE.COM** などのサービスプリンシパルで完全なホスト名を使用できます。
- IdM がない環境では、RHEL ホストを Active Directory (AD) ドメインのメンバーとする場合に、AD ドメインコントローラー (DC) が、AD に登録されているマシンの NetBIOS 名のサービスプリンシパルを自動的に作成するため、これ以上考慮が必要な事項はありません。

第91章 ANSIBLE PLAYBOOK を使用した IDM でのグローバル DNS 設定の管理

Red Hat Ansible Engine の **dnsconfig** モジュールを使用して、Identity Management (IdM) DNS のグローバル設定を設定できます。グローバル DNS 設定で定義したオプションは、すべての IdM DNS サーバーに適用されます。ただし、グローバル設定は、特定の IdM DNS ゾーンの設定よりも優先度が低くなります。

dnsconfig モジュールは以下の変数をサポートします。

- グローバルフォワーダー (特に通信に使用する IP アドレスとポート)
- グローバル転送ポリシー: only、first、または noneDNS 転送ポリシーの上記のタイプの詳細は、IdM の DNS 転送ポリシー を参照してください。
- 正引きルックアップおよび逆引きルックアップゾーンの同期。

前提条件

- DNS サービスが IdM サーバーにインストールされている。統合 DNS のある IdM サーバーをインストールする方法は、以下のリンクのいずれかを参照してください。
  - IdM サーバーのインストール: 統合 DNS と統合 CA を root CA として使用する場合
  - IdM サーバーのインストール: 統合 DNS と外部 CA を root CA として使用する場合
  - IdM サーバーのインストール: 統合 DNS があり外部 CA がない場合

本章では、以下のセクションを説明します。

- IdM を使用して /etc/resolv.conf のグローバルフォワーダーが NetworkManager に削除されないようにする方法
- Ansible を使用して IdM に DNS グローバルフォワーダーを存在させる手順
- Ansible を使用して IdM に DNS グローバルフォワーダーを存在させないようにする手順
- ipadnsconfig ansible-freeipa モジュールの **action: member** オプション
- IdM の DNS 転送ポリシーの 概要
- Ansible Playbook を使用して forward first ポリシーを IdM DNS グローバル設定で指定する手順
- Ansible Playbook を使用して IdM DNS でグローバルフォワーダーを無効にする手順
- Ansible Playbook を使用して IdM DNS で正引きおよび逆引きルックアップゾーンの同期を無効にする手順

91.1. IDM を使用して /ETC/RESOLV.CONF のグローバルフォワーダーが NETWORKMANAGER に削除されないようにする方法

統合 DNS で Identity Management (IdM) をインストールすると、**/etc/resolv.conf** ファイルが localhost アドレス (**127.0.0.1**) を参照するように設定されます。

```
# Generated by NetworkManager
search idm.example.com
nameserver 127.0.0.1
```

DHCP (**Dynamic Host Configuration Protocol**) を使用するネットワークなど、環境によっては、**/etc/resolv.conf** ファイルへの変更が **NetworkManager** サービスにより元に戻されてしまう場合があります。IdM DNS のインストールプロセスでは、以下のように **NetworkManager** サービスも設定し、DNS 設定を永続化します。

1. DNS インストールスクリプトを使用して、**/etc/NetworkManager/conf.d/zzzz-ipa.conf NetworkManager** 設定ファイルを作成し、検索の順序と DNS サーバーリストを制御します。

```
# auto-generated by IPA installer
[main]
dns=default

[global-dns]
searches=$DOMAIN

[global-dns-domain-*]
servers=127.0.0.1
```

2. **NetworkManager** サービスが再読み込みされ、**/etc/NetworkManager/conf.d/** ディレクトリーにある以前のファイルの設定を使用して **/etc/resolv.conf** ファイルを作成します。今回の場合は、**zzz-ipa.conf** ファイルです。

重要

/etc/resolv.conf ファイルは手動で変更しないでください。

91.2. ANSIBLE を使用して IDM に DNS グローバルフォワーダーを存在させる手順

以下の手順に従って、Ansible Playbook を使用して、IdM に DNS グローバルフォワーダーを追加します。以下の例では、IdM 管理者は、ポート **53** にインターネットプロトコル (IP) v4 アドレスが **7.7.9.9**、IPv6 アドレスが **2001:db8::1:0** で指定されている DNS サーバーに、DNS グローバルフォワーダーが配置されるようにします。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

- IdM 管理者パスワードを把握している。

手順

1. **/usr/share/doc/ansible-freeipa/playbooks/dnsconfig** ディレクトリーに移動します。

```
$ cd /usr/share/doc/ansible-freeipa/playbooks/dnsconfig
```

2. インベントリーファイルを開き、設定する IdM サーバーが **[ipaserver]** セクションに記載されていることを確認します。たとえば、Ansible に対して **server.idm.example.com** を設定するように指示するには、次のコマンドを実行します。

```
[ipaserver]
server.idm.example.com
```

3. Ansible Playbook ファイル (**forwarders-absent.yml**) のコピーを作成します。以下に例を示します。

```
$ cp forwarders-absent.yml ensure-presence-of-a-global-forwarder.yml
```

4. **ensure-presence-of-a-global-forwarder.yml** ファイルを開いて編集します。

5. 以下の変数を設定してファイルを調整します。

   a. Playbook の **name** 変数は、**IdM DNS にグローバルフォワーダーを追加する Playbook** の設定に変更します。

   b. **tasks** セクションで、タスクの **name** を **Ensure the presence of a DNS global forwarder to 7.7.9.9 and 2001:db8::1:0 on port 53** に変更します。

   c. **ipadnsconfig** の **forwarders** セクションで以下を行います。

      i. 最初の **ip_address** の値は、グローバルフォワーダーの IPv4 アドレス (**7.7.9.9**) に変更します。

      ii. 2 番目の **ip_address** の値は、グローバルフォワーダーの IPv6 アドレス (**2001:db8::1:0**) に変更します。

      iii. **port** の値が **53** に設定されていることを確認します。

   d. **state** を **present** に変更します。
   今回の例で使用するように変更した Ansible Playbook ファイル:

```
---
- name: Playbook to ensure the presence of a global forwarder in IdM DNS
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure the presence of a DNS global forwarder to 7.7.9.9 and 2001:db8::1:0 on port
53
    ipadnsconfig:
      forwarders:
        - ip_address: 7.7.9.9
```

```
    - ip_address: 2001:db8::1:0
      port: 53
  state: present
```

6. ファイルを保存します。

7. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file ensure-presence-of-a-global-forwarder.yml
```

関連情報

- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-dnsconfig.md** ファイルを参照してください。

91.3. ANSIBLE を使用して IDM に DNS グローバルフォワーダーを存在させないようにする手順

以下の手順に従って、Ansible Playbook を使用して IdM で DNS グローバルフォワーダーを削除します。以下の手順では、IdM 管理者が、ポート **53** で、IP (Internet Protocol) v4 アドレス **8.8.6.6** および IP v6 アドレス **2001:4860:4860::8800** を持つ DNS グローバルフォワーダーが存在しないことを確認します。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- IdM 管理者パスワードを把握している。

手順

1. **/usr/share/doc/ansible-freeipa/playbooks/dnsconfig** ディレクトリーに移動します。

```
$ cd /usr/share/doc/ansible-freeipa/playbooks/dnsconfig
```

2. インベントリーファイルを開き、設定する IdM サーバーが **[ipaserver]** セクションに記載されていることを確認します。たとえば、Ansible に対して **server.idm.example.com** を設定するように指示するには、次のコマンドを実行します。

```
[ipaserver]
server.idm.example.com
```

3. Ansible Playbook ファイル (**forwarders-absent.yml**) のコピーを作成します。以下に例を示します。

```
$ cp forwarders-absent.yml ensure-absence-of-a-global-forwarder.yml
```

4. **ensure-absence-of-a-global-forwarder.yml** ファイルを開いて編集します。

5. 以下の変数を設定してファイルを調整します。

   a. Playbook の **name** 変数は、**IdM DNS でグローバルフォワーダーを配置しない Playbook** の設定に変更します。

   b. **tasks** セクションで、タスクの **name** を **Ensure the absence of a DNS global forwarder to 8.8.6.6 and 2001:4860:4860::8800 on port 53** に変更します。

   c. **ipadnsconfig** の **forwarders** セクションで以下を行います。

      i. 最初の **ip_address** の値は、グローバルフォワーダーの IPv4 アドレス (**8.8.6.6**) に変更します。

      ii. 2 番目の **ip_address** の値は、グローバルフォワーダーの IPv6 アドレス (**2001:4860:4860::8800**) に変更します。

      iii. **port** の値が **53** に設定されていることを確認します。

   d. **action** 変数は **member** に設定します。

   e. **state** が **absent** に設定されていることを確認します。

   今回の例で使用するように変更した Ansible Playbook ファイル:

```
---
- name: Playbook to ensure the absence of a global forwarder in IdM DNS
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure the absence of a DNS global forwarder to 8.8.6.6 and
2001:4860:4860::8800 on port 53
    ipadnsconfig:
      forwarders:
        - ip_address: 8.8.6.6
        - ip_address: 2001:4860:4860::8800
          port: 53
      action: member
      state: absent
```

重要

Playbook で **action: member** を使用せずに **state: absent** オプションだけを使用すると、その Playbook は失敗します。

6. ファイルを保存します。

7. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file ensure-absence-of-a-global-forwarder.yml
```

関連情報

- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-dnsconfig.md** ファイル
- ipadnsconfig ansible-freeipa モジュールの **action: member** オプション

91.4. IPADNSCONFIG ANSIBLE-FREEIPA モジュールの ACTION: MEMBER オプション

ansible-freeipa ipadnsconfig モジュールを使用して Identity Management (IdM) のグローバルフォワーダーを除外するには、**state: absent** オプションの他に **action: member** オプションを使用する必要があります。Playbook で **action: member** を使用せずに **state: absent** だけを使用すると、その Playbook は失敗します。そのため、すべてのグローバルフォワーダーを削除するには、Playbook でこれらをすべて個別に指定する必要があります。一方、**state: present** オプションに **action: member** は必要ありません。

次の表 に、action: member オプションの正しい使用法を示す DNS グローバルフォワーダーの追加と削除の両方の設定例を示します。この表の各行には、以下が含まれます。

- Playbook を実行する前に設定されたグローバルフォワーダー
- Playbook からの抜粋
- Playbook の実行後に設定されたグローバルフォワーダー

表91.1 グローバルフォワーダーの ipadnsconfig 管理

以前のフォワーダー	Playbook の抜粋	後のフォワーダー

8.8.6.6	[...]
	tasks:
	- name: Ensure the presence of DNS global forwarder 8.8.6.7
	  ipadnsconfig:
	  forwarders:
	  - ip_address: 8.8.6.7
	  state: present
		8.8.6.7

以前のフォワーダー	Playbook の抜粋	後のフォワーダー

8.8.6.6	[...]
	tasks:
	- name: Ensure the presence of DNS global forwarder 8.8.6.7
	  ipadnsconfig:
	  forwarders:
	  - ip_address: 8.8.6.7
	  action: member
	  state: present
		8.8.6.6、8.8.6.7

8.8.6.6、8.8.6.7	[...]
	tasks:
	- name: Ensure the absence of DNS global forwarder 8.8.6.7
	  ipadnsconfig:
	  forwarders:
	  - ip_address: 8.8.6.7
	  state: absent
		Playbook を実行しようとすると、エラーが発生します。元の設定 (8.8.6.6、8.8.6.7) は変更されません。

8.8.6.6、8.8.6.7	[...]
	tasks:
	- name: Ensure the absence of DNS global forwarder 8.8.6.7
	  ipadnsconfig:
	  forwarders:
	  - ip_address: 8.8.6.7
	  action: member
	  state: absent
		8.8.6.6

91.5. IDM での DNS 転送ポリシー

IdM は、**first** および **only** の BIND 転送ポリシーと、IdM 固有の転送ポリシー **none** をサポートします。

forward first (デフォルト)

IdM BIND サービスは、DNS クエリーを設定済みのフォワーダーに転送します。サーバーエラーやタイムアウトが原因でクエリーに失敗すると、BIND はインターネット上のサーバーを使用して再帰解決にフォールバックします。**forward first** ポリシーはデフォルトのポリシーで、DNS トラフィックの最適化に適しています。

Forward only

IdM BIND サービスは、DNS クエリーを設定済みのフォワーダーに転送します。サーバーエラーやタイムアウトが原因でクエリーに失敗すると、BIND はエラーをクライアントに返します。分割された DNS 設定の環境では、**forward only** ポリシーが推奨されます。

None (転送の無効化)

DNS クエリーは、**none** 転送ポリシーで転送されません。グローバル転送設定をゾーン別にオーバーライドする場合にのみ、転送の無効化は有用です。このオプションは、IdM の BIND 設定で空のフォワーダーリストを指定するのと同じです。

注記

転送を使用して、IdM のデータと、他の DNS サーバーのデータと統合できません。IdM DNS のプライマリーゾーン内にある特定のサブゾーンのクエリーのみを転送できます。

デフォルトでは、IdM サーバーが権威サーバーとなっているゾーンに、クエリーされた DNS 名が所属する場合には、BIND サービスは、クエリーを別のサーバーに転送しません。このような場合は、クエリーされた DNS 名が IdM データベースに見つからない場合は、**NXDOMAIN** との応答が返されます。転送は使用されません。

例91.1 サンプルシナリオ

IdM サーバーは、**test.example** の権威サーバーです。DNS ゾーン。BIND は、IP アドレス **192.0.2.254** でクエリーを DNS サーバーに転送するように設定されています。

クライアントが **nonexistent.test.example** のクエリーを送信する場合DNS 名である BIND は、IdM サーバーが **test.example.** ゾーンの権威サーバーであることを検出して、クエリーを **192.0.2.254.** サーバーには転送しません。その結果、DNS クライアントは **NXDomain** エラーメッセージを受け取り、クエリーされたドメインが存在しないことをユーザーに通知します。

91.6. ANSIBLE PLAYBOOK を使用して FORWARD FIRST ポリシーを IDM DNS グローバル設定で指定する手順

以下の手順に従って、Ansible Playbook を使用して、IdM DNS のグローバル転送ポリシーが **forward first** に設定されていることを確認します。

forward first DNS 転送ポリシーを使用する場合には、DNS クエリーは設定済みのフォワーダーに転送されます。サーバーエラーやタイムアウトが原因でクエリーに失敗すると、BIND はインターネット上のサーバーを使用して再帰解決にフォールバックします。Forward first ポリシーはデフォルトのポリシーです。トラフィックの最適化に適しています。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

- IdM 管理者パスワードを把握している。
- IdM 環境に統合 DNS サーバーが含まれている。

手順

1. **/usr/share/doc/ansible-freeipa/playbooks/dnsconfig** ディレクトリーに移動します。

```
$ cd /usr/share/doc/ansible-freeipa/playbooks/dnsconfig
```

2. インベントリーファイルを開き、設定する IdM サーバーが **[ipaserver]** セクションに記載されていることを確認します。たとえば、Ansible に対して **server.idm.example.com** を設定するように指示するには、次のコマンドを実行します。

```
[ipaserver]
server.idm.example.com
```

3. Ansible Playbook ファイル (**set-configuration.yml**) のコピーを作成します。以下に例を示します。

```
$ cp set-configuration.yml set-forward-policy-to-first.yml
```

4. **set-forward-policy-to-first.yml** ファイルを開いて編集します。
5. **ipadnsconfig** タスクセクションに以下の変数を設定して、ファイルを調整します。
   - **ipaadmin_password** 変数は IdM 管理者パスワードに設定します。
   - **forward_policy** 変数は **first** に設定します。
     元の Playbook で関連性の他の行はすべて削除します。以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Playbook to set global forwarding policy to first
  hosts: ipaserver
  become: true

  tasks:
  - name: Set global forwarding policy to first.
    ipadnsconfig:
      ipaadmin_password: "{{ ipaadmin_password }}"
      forward_policy: first
```

6. ファイルを保存します。
7. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file set-forward-policy-to-first.yml
```

関連情報

- IdM での DNS 転送ポリシー を参照してください。

- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-dnsconfig.md** ファイルを参照してください。
- サンプルの Playbook は、**/usr/share/doc/ansible-freeipa/playbooks/dnsconfig** ディレクトリーを参照してください。

91.7. ANSIBLE PLAYBOOK を使用して IDM DNS でグローバルフォワーダーを無効にする手順

以下の手順に従って、Ansible Playbook を使用して、IdM DNS でグローバルフォワーダーが無効になっていることを確認します。グローバルフォワーダーの無効化は、**forward_policy** 変数を **none** に設定します。

グローバルフォワーダーを無効にすると、DNS クエリーは転送されません。グローバル転送設定をゾーン別にオーバーライドする場合にのみ、転送の無効化は有用です。このオプションは、IdM の BIND 設定で空のフォワーダーリストを指定するのと同じです。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- IdM 管理者パスワードを把握している。
- IdM 環境に統合 DNS サーバーが含まれている。

手順

1. **/usr/share/doc/ansible-freeipa/playbooks/dnsconfig** ディレクトリーに移動します。

```
$ cd /usr/share/doc/ansible-freeipa/playbooks/dnsconfig
```

2. インベントリーファイルを開き、設定する IdM サーバーが **[ipaserver]** セクションに記載されていることを確認します。たとえば、Ansible に対して **server.idm.example.com** を設定するように指示するには、次のコマンドを実行します。

```
[ipaserver]
server.idm.example.com
```

3. Ansible Playbook ファイル (**disable-global-forwarders.yml**) のコピーを作成します。以下に例を示します。

```
$ cp disable-global-forwarders.yml disable-global-forwarders-copy.yml
```

4. **disable-global-forwarders-copy.yml** ファイルを開いて編集します。
5. **ipadnsconfig** タスクセクションに以下の変数を設定して、ファイルを調整します。
   - **ipaadmin_password** 変数は IdM 管理者パスワードに設定します。
   - **forward_policy** 変数を **none** に設定します。
     以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Playbook to disable global DNS forwarders
  hosts: ipaserver
  become: true

  tasks:
  - name: Disable global forwarders.
    ipadnsconfig:
      ipaadmin_password: "{{ ipaadmin_password }}"
      forward_policy: none
```

6. ファイルを保存します。
7. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file disable-global-forwarders-copy.yml
```

関連情報

- IdM での DNS 転送ポリシー を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-dnsconfig.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/dnsconfig** ディレクトリーにあるその他のサンプル Playbook を参照してください。

91.8. ANSIBLE PLAYBOOK を使用して IDM DNS で正引きおよび逆引きルックアップゾーンの同期を無効にする手順

以下の手順に従って、Ansible Playbook を使用して、正引きおよび逆引きルックアップゾーンが IdM DNS で同期されないようにします。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。

    - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。

- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- IdM 管理者パスワードを把握している。
- IdM 環境に統合 DNS サーバーが含まれている。

手順

1. **/usr/share/doc/ansible-freeipa/playbooks/dnsconfig** ディレクトリーに移動します。

```
$ cd /usr/share/doc/ansible-freeipa/playbooks/dnsconfig
```

2. インベントリーファイルを開き、設定する IdM サーバーが **[ipaserver]** セクションに記載されていることを確認します。たとえば、Ansible に対して **server.idm.example.com** を設定するように指示するには、次のコマンドを実行します。

```
[ipaserver]
server.idm.example.com
```

3. Ansible Playbook ファイル (**disallow-reverse-sync.yml**) のコピーを作成します。以下に例を示します。

```
$ cp disallow-reverse-sync.yml disallow-reverse-sync-copy.yml
```

4. **disallow-reverse-sync-copy.yml** ファイルを開きます。
5. **ipadnsconfig** タスクセクションに以下の変数を設定して、ファイルを調整します。
    - **ipaadmin_password** 変数は IdM 管理者パスワードに設定します。
    - **allow_sync_ptr** 変数を **no** に設定します。
      以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Playbook to disallow reverse record synchronization
  hosts: ipaserver
  become: true

  tasks:
  - name: Disallow reverse record synchronization.
    ipadnsconfig:
      ipaadmin_password: "{{ ipaadmin_password }}"
      allow_sync_ptr: no
```

6. ファイルを保存します。
7. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file disallow-reverse-sync-copy.yml
```

関連情報

- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-dnsconfig.md** ファイルを参照してください。
- サンプルの Playbook は、**/usr/share/doc/ansible-freeipa/playbooks/dnsconfig** ディレクトリーを参照してください。

第92章 IDM での DNS ゾーンの管理

Identity Management (IdM) 管理者は、IdM DNS ゾーンの動作を管理できます。本章では、以下のトピックおよび手順を説明します。

- IdM でサポートされる DNS ゾーンタイプ
  - IdM Web UI を使用してプライマリー IdM DNS ゾーンを追加する方法
  - IdM CLI を使用してプライマリー IdM DNS ゾーンを追加する方法
  - IdM Web UI を使用してプライマリー IdM DNS ゾーンを削除する方法
  - IdM CLI を使用してプライマリー IdM DNS ゾーンを削除する方法
- IdM で設定できる DNS 属性
  - IdM Web UI で DNS 属性を設定する方法
  - IdM CLI で DNS 属性を設定する方法
- IdM でのゾーン転送の仕組み
  - IdM Web UI でゾーン転送を許可する方法
  - IdM CLI でゾーン転送を許可する方法

前提条件

- DNS サービスが IdM サーバーにインストールされている。統合 DNS のある IdM サーバーをインストールする方法は、以下のリンクのいずれかを参照してください。
  - IdM サーバーのインストール: 統合 DNS と統合 CA を root CA として使用する場合
  - IdM サーバーのインストール: 統合 DNS と外部 CA を root CA として使用する場合
  - IdM サーバーのインストール: 統合 DNS があり外部 CA がない場合

92.1. サポート対象の DNS ゾーンタイプ

Identity Management (IdM) は、2 種類の DNS ゾーン (**primary** および **forward**) をサポートします。ここでは、DNS 転送のシナリオ例を含め、2 種類のゾーンについて説明します。

注記

本ガイドでは、ゾーンタイプには BIND の用語を使用し、Microsoft Windows DNS で使用する用語とは異なります。BIND のプライマリーゾーンは、Microsoft Windows DNS の **正引きルックアップゾーン** と **逆引きルックアップゾーン** と同じ目的で使用されます。BIND の正引きゾーンは、Microsoft Windows DNS の **条件付きフォワーダー** と同じ目的で使用されます。

プライマリー DNS ゾーン

プライマリー DNS ゾーンには、権威 DNS データが含まれ、DNS を動的に更新できます。この動作は、標準 BIND 設定の **type master** 設定と同じです。プライマリーゾーンは、**ipa dnszone-*** コマンドを使用して管理できます。

標準 DNS ルールに準拠するには、プライマリーゾーンすべてに **start of authority** (SOA) と **nameserver** (NS) レコードを含める必要があります。IdM では、DNS ゾーンの作成時にこれらのレコードが自動的に生成されますが、NS レコードを親ゾーンに手動でコピーして適切な委譲を作成する必要があります。

標準の BIND の動作に合わせて、権威サーバーではない名前のクエリーは、他の DNS サーバーに転送されます。DNS サーバー (別称: フォワーダー) は、クエリーに対して権威がある場合と、ない場合があります。

> **例92.1 DNS 転送のシナリオ例**
>
> IdM サーバーには **test.example.** プライマリーゾーンが含まれています。このゾーンには、**sub.test.example.** 名前の NS 委譲レコードが含まれます。さらに、**test.example.** ゾーンは、**sub.text.example** サブゾーンのフォワーダー IP アドレス **192.0.2.254** で設定されます。
>
> クライアントが **nonexistent.test.example.** の名前をクエリーすると、**NXDomain** の応答を受け取りますが、IdM サーバーはこの名前に対して権威があるため、転送は発生しません。
>
> 反対に、**host1.sub.test.example.** の名前をクエリーすると、IdM サーバーはこの名前に対して権威がないので、設定済みのフォワーダー (**192.0.2.254**) に転送されます。

正引き DNS ゾーン

IdM の観点からは、正引き DNS ゾーンには権威データは含まれません。実際、正引きのゾーンには、通常以下情報 2 つのみが含まれます。

- ドメイン名
- ドメインに関連付けられた DNS サーバーの IP アドレス

定義済みのドメインに所属する名前のクエリーはすべて、指定の IP アドレスに転送されます。この動作は、標準 BIND 設定の **type forward** 設定と同じです。正引きゾーンは、**ipa dnsforwardzone-*** コマンドを使用して管理できます。

正引き DNS ゾーンは、IdM-Active Directory (AD) 信頼のコンテキストで特に便利です。IdM DNS サーバーが **idm.example.com** ゾーンに対して、AD DNS サーバーが **ad.example.com** ゾーンに対して権威がある場合には、**ad.example.com** が **idm.example.com** プライマリーゾーンの DNS 正引きゾーンになります。つまり、IP アドレスが **somehost.ad.example.com** の IdM クライアントからクエリーが送信されると、**ad.example.com** IdM DNS 正引きゾーンに指定の AD ドメインコントローラーに転送されます。

92.2. IDM WEB UI でのプライマリー DNS ゾーンの追加

Identity Management (IdM) Web UI を使用してプライマリー DNS ゾーンを追加するには、次の手順に従います。

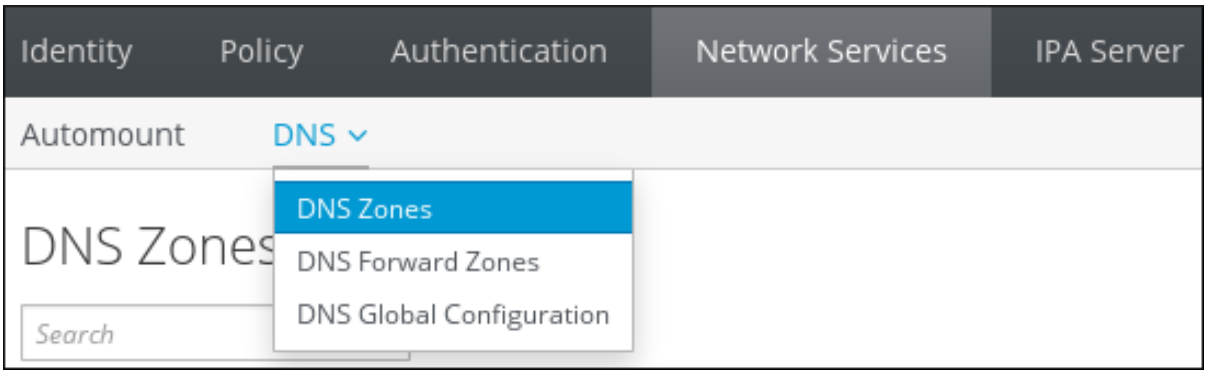
前提条件

- IdM 管理者としてログインしている。

手順

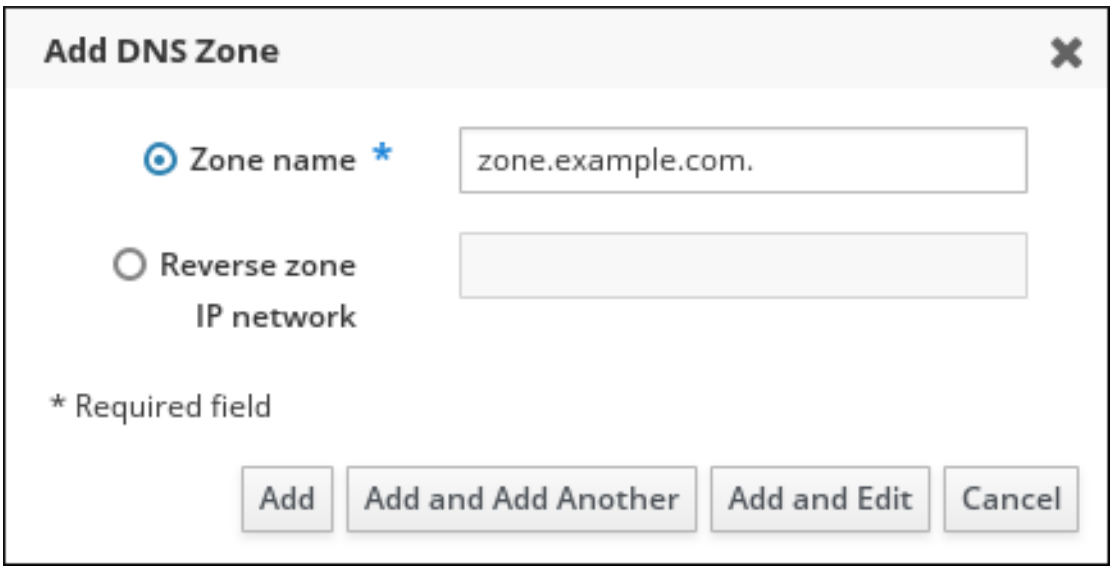
1. IdM Web UI で、**Network Services → DNS → DNS Zones** の順にクリックします。

図92.1 IdM DNS プライマリーゾーンの管理



2. すべてのゾーンリストの上部にある **追加** をクリックします。
3. ゾーン名を指定します。

図92.2 新しい IdM プライマリーゾーンの入力



4. **Add** をクリックします。

92.3. IDM CLI でのプライマリー DNS ゾーンの追加

Identity Management (IdM) コマンドラインインターフェイス (CLI) を使用してプライマリー DNS ゾーンを追加するには、次の手順に従います。

前提条件

- IdM 管理者としてログインしている。

手順

- **ipa dnszone-add** コマンドは、新しいゾーンを DNS ドメインに追加します。新しいゾーンを追加するには、新しいサブドメイン名を指定する必要があります。サブドメイン名を直接指定するには、以下のコマンドを実行します。

```
$ ipa dnszone-add newzone.idm.example.com
```

ipa dnszone-add に名前を指定しない場合には、スクリプトにより自動的に名前を求めるプロンプトが表示されます。

関連情報

- **ipa dnszone-add --help** を参照してください。

92.4. IDM WEB UI でのプライマリー DNS ゾーンの削除

IdM Web UI を使用して Identity Management (IdM) からプライマリー DNS ゾーンを削除するには、この手順に従います。

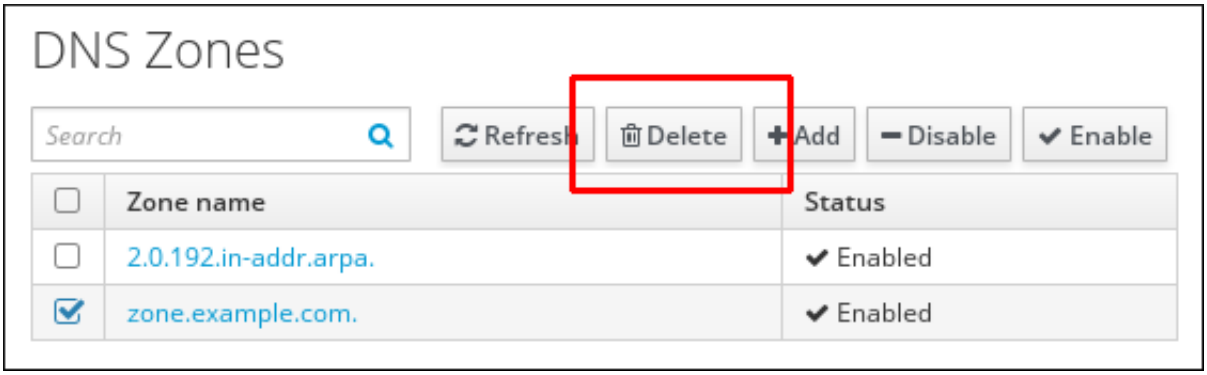
前提条件

- IdM 管理者としてログインしている。

手順

1. IdM Web UI で、**Network Services → DNS → DNS Zones** の順にクリックします。
2. ゾーン名の横にあるチェックボックスを選択し、**削除** をクリックします。

図92.3 プライマリー DNS ゾーンの削除



3. **DNS ゾーンの削除** ダイアログウィンドウで、選択したゾーンの削除を確定します。

92.5. IDM CLI でのプライマリー DNS ゾーンの削除

IdM コマンドラインインターフェイス (CLI) を使用して Identity Management (IdM) からプライマリー DNS ゾーンを削除するには、次の手順に従います。

前提条件

- IdM 管理者としてログインしている。

手順

- プライマリー DNS ゾーンを削除するには、**ipa dnszone-del** コマンドの後に、削除するゾーンの名前を入力します。以下に例を示します。

```
$ ipa dnszone-del idm.example.com
```

92.6. DNS 設定の優先順位

多くの DNS 設定オプションは、次のレベルで設定できます。優先度は、レベルごとに異なります。

ゾーン固有の設定

IdM に定義されている特定のゾーンに固有の設定は、優先度が最も高いレベルです。**ipa dnszone-*** コマンドおよび **ipa dnsforwardzone-*** コマンドを使用して、ゾーン固有の設定を管理できます。

サーバーごとの設定

IdM サーバーのインストール時に、サーバーごとのフォワーダーを定義するように求められます。サーバーごとのフォワーダーは、**ipa dnsserver-*** コマンドを使用して管理できます。レプリカのインストール時にサーバーごとのフォワーダーを設定しない場合は、**--no-forwarder** オプションを使用できます。

グローバル DNS 設定

ゾーン固有の設定が定義されていない場合は、IdM は LDAP に保存されているグローバル DNS 設定を使用します。グローバル DNS 設定は、**ipa dnsconfig-*** コマンドを使用して管理できます。グローバル DNS 設定で定義したオプションは、すべての IdM DNS サーバーに適用されます。

/etc/named.conf の設定

IdM DNS サーバーごとに **/etc/named.conf** ファイルで定義されている設定の優先度は、最も低くなります。これは各サーバーに固有のものであり、手動で編集する必要があります。

/etc/named.conf ファイルを使用するのは通常、ローカル DNS キャッシュへの DNS 転送を指定する場合のみです。他のオプションは、上記のゾーン固有の設定と、グローバル DNS 設定のコマンドを使用して管理します

DNS オプションは、同時に複数のレベルで設定できます。このような場合に、最も優先度が高い設定は、レベルが低い設定よりも優先されます。

関連情報

- Per Server Config in LDAP の **Priority order of configuration** セクション

92.7. プライマリー IDM DNS ゾーンの設定属性

Identity Management (IdM) は、更新期間、転送設定、キャッシュ設定など、特定のデフォルト設定を指定して新しいゾーンを作成します。IdM DNS ゾーン属性 には、デフォルトのゾーン設定属性があります。これは、以下のオプションのいずれかを使用して変更できます。

- コマンドラインインターフェイス (CLI) の **dnszone-mod** コマンド詳細は IdM CLI でのプライマリー DNS ゾーンの設定の編集 を参照してください。
- IdM Web UI詳細は IdM Web UI でのプライマリー DNS ゾーンの設定の編集 を参照してください。
- **ipadnszone** モジュールを使用する Ansible Playbook詳細は、IdM での DNS ゾーンの管理 を参照してください。

ここではゾーンの実際の情報を設定するほか、DNS サーバーが **start of authority** (SOA) レコードエントリーを処理する方法と、DNS ネームサーバーからの記録を更新する方法を定義します。

表92.1 IdM DNS ゾーン属性

属性	コマンドラインオプション	説明
権威ネームサーバー	**--name-server**	プライマリー DNS ネームサーバーのドメイン名 (別称: SOA MNAME) を設定します。 デフォルトでは、各 IdM サーバーは SOA MNAME フィールドで自己アドバタイズします。そのため、**--name-server** を使用して LDAP に保存されている値は無視されます。
管理者の電子メールアドレス	**--admin-email**	ゾーン管理者が使用する電子メールアドレスを設定します。デフォルトでは、ホストの root アカウントになります。
SOA serial	**--serial**	SOA レコードにシリアル番号を設定します。IdM ではバージョン番号が自動的に設定され、この番号のユーザーによる変更は想定されていません。
SOA refresh	**--refresh**	セカンダリー DNS サーバーがプライマリー DNS サーバーから更新を要求するまでの待機時間を秒単位で設定します。
SOA retry	**--retry**	失敗した更新操作を再試行するまでに待機する時間を秒単位で設定します。
SOA expire	**--expire**	セカンダリー DNS サーバーが操作の試行を終了するまでに、更新操作を実行する時間を秒単位で設定します。
SOA minimum	**--minimum**	RFC 2308 に準拠し、ネガティブキャッシュの TTL (TTL) 値を秒単位で設定します。
SOA time to live	**--ttl**	ゾーン apex のレコードの TTL を秒単位で設定します。たとえば、**example.com** ゾーンでは、名前が **example.com** のすべてのレコード (A、NS または SOA) が設定されますが、**test.example.com** などの他のドメイン名には影響はありません。
デフォルトの TTL	**--default-ttl**	これまでに個別の Time To Live (TTL) 値が設定されたことのないゾーンで、すべての値のネガティブキャッシュのデフォルト TTL を秒単位で設定します。変更を有効にするには、すべての IdM DNS サーバーで **named-pkcs11** サービスを再起動する必要があります。
BIND 更新ポリシー	**--update-policy**	DNS ゾーンでクライアントに許可されるパーミッションを設定します。
Dynamic update	**--dynamic-update**=TRUE|FALSE	クライアントの DNS レコードへの動的更新を有効にします。 false に設定すると、IdM クライアントマシンは IP アドレスを追加または更新できなくなる点に注意してください。

属性	コマンドラインオプション	説明
Allow transfer	**--allow-transfer**=string	指定のゾーンを転送できる IP アドレスまたはネットワーク名のセミコロン区切りのリストを指定します。 デフォルトでは、ゾーン転送は無効です。デフォルトの **--allow-transfer** 値は **none** です。
Allow query	**--allow-query**	DNS クエリーを発行できる IP アドレスまたはネットワーク名のセミコロン区切りのリストを指定します。
Allow PTR sync	**--allow-sync-ptr**=1|0	ゾーンの A または AAAA レコード (正引きレコード) が自動的に PTR (逆引き) レコードと同期されるかどうかを設定します。
Zone forwarder	**--forwarder**=IP_address	DNS ゾーン向けに特別に設定されたフォワーダーを指定します。これは、IdM ドメインで使用されるグローバルフォワーダーとは別のものです。 複数のフォワーダーを指定する場愛には、オプションを複数回使用します。
転送ポリシー	**--forward-policy**=none|only|first	転送ポリシーを指定します。サポート対象のポリシーに関する情報は、IdM での DNS 転送ポリシー を参照してください。

92.8. IDM WEB UI でのプライマリー DNS ゾーン設定の編集

IdM Web UI を使用してプライマリー Identity Management (IdM) DNS の設定属性を編集するには、この手順に従います。

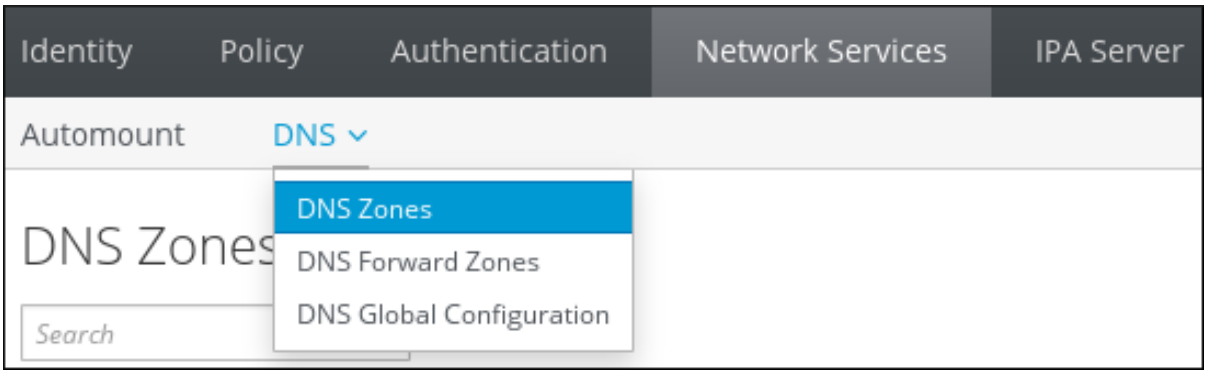
前提条件

- IdM 管理者としてログインしている。

手順

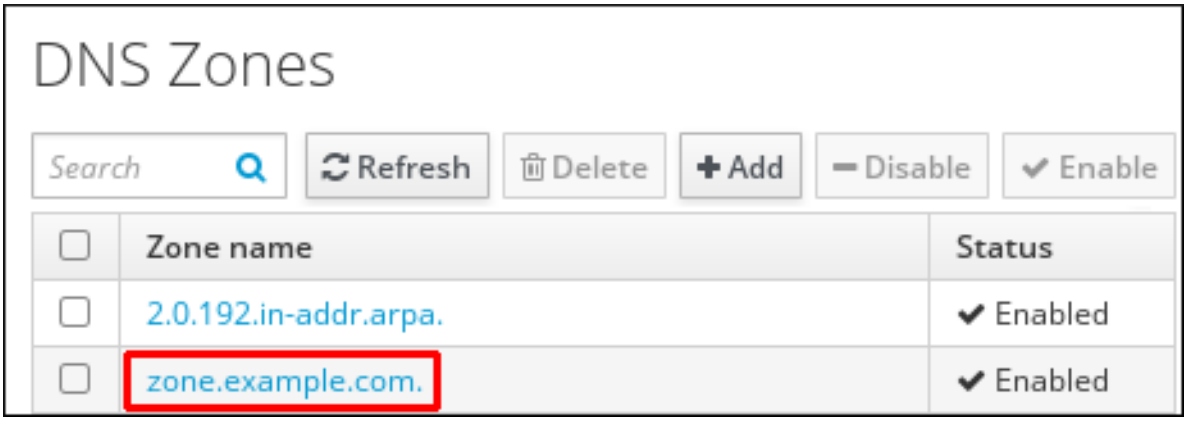
1. IdM Web UI で、**Network Services → DNS → DNS Zones** の順にクリックします。

図92.4 DNS プライマリーゾーンの管理



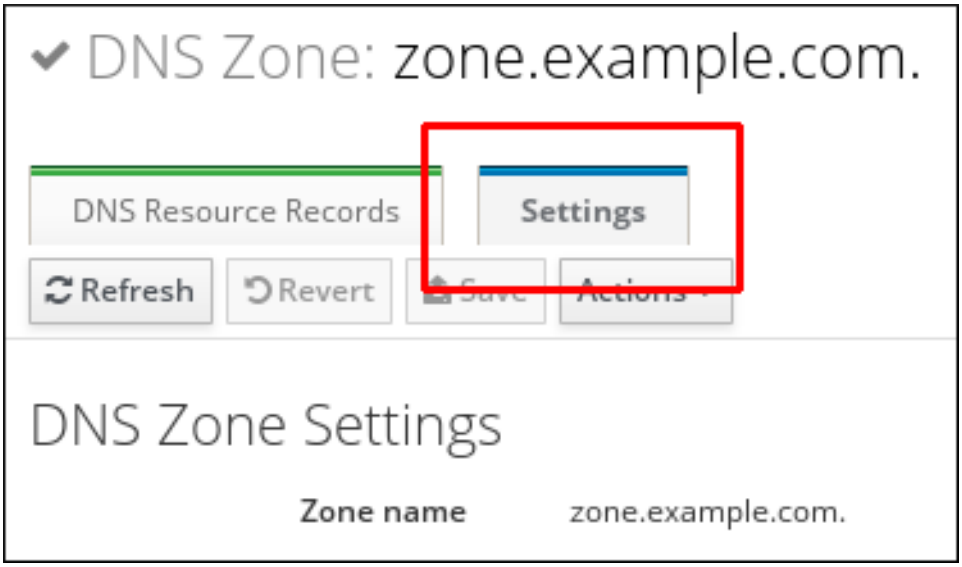
2. **DNS ゾーン** セクションで、全ゾーンのリストにあるゾーン名をクリックし、DNS ゾーンページを開きます。

図92.5 プライマリーゾーンの編集



3. **Settings** を クリックします。

図92.6 プライマリーゾーンの編集ページの設定タブ



4. 必要に応じてゾーン設定を変更します。
利用可能な設定の詳細は、IdM DNS ゾーン属性 を参照してください。

5. **Save** をクリックして、新しい設定を確定します。

注記

ゾーンのデフォルトの Time To Live (TTL) を変更する場愛には、全 IdM DNS サーバーで **named-pkcs11** サービスを再起動して、変更を適用します。他の全設定は、すぐに自動的に有効になります。

92.9. IDM CLI でのプライマリー DNS ゾーンの設定の編集

Identity Management (IdM) コマンドラインインターフェイス (CLI) を使用してプライマリー DNS ゾーンの設定を編集するには、次の手順に従います。

前提条件

- IdM 管理者としてログインしている。

手順

- 既存のプライマリー DNS ゾーンを変更するには、**ipa dnszone-mod** コマンドを使用します。たとえば、失敗した更新操作を再試行するまでに待機する時間を 1800 秒に設定します。

```
$ ipa dnszone-mod --retry 1800
```

利用可能な設定と、対応する CLI オプションの詳細は、IdM DNS ゾーン属性 を参照してください。

特定の設定で、変更する DNS ゾーンエントリーに値が指定されていない場合は、**ipa dnszone-mod** コマンドで値を追加します。設定に値がない場合は、このコマンドを実行すると、指定の値に上書きされます。

注記

ゾーンのデフォルトの Time To Live (TTL) を変更する場愛には、全 IdM DNS サーバーで **named-pkcs11** サービスを再起動して、変更を適用します。他の全設定は、すぐに自動的に有効になります。

関連情報

- **ipa dnszone-mod --help** を参照してください。

92.10. IDM でのゾーン転送

DNS が統合された Identity Management (IdM) デプロイメントでは、**ゾーン転送** を使用して、すべてのリソースレコードを 1 つのネームサーバーから別のネームサーバーにコピーできます。ネームサーバーは、ゾーンの権威データを保持します。DNS ゾーン **zone A** に権威のある DNS サーバーでゾーンに変更を加えると、**zone A** 以外にある IdM DNS ドメインの他のネームサーバーに変更を配信する必要があります。

重要

IdM 統合 DNS には、複数のサーバーで同時に記述できます。IdM ゾーンの Start of Authority (SOA) シリアル番号は、個別の IdM DNS サーバーと同期されません。このような理由から、転送予定ゾーン外にある DNS サーバーは、転送予定ゾーン内の特定の DNS サーバー 1 台だけを使用するように設定します。こうすることで、同期されていない SOA シリアル番号が原因でゾーン転送が失敗しないようにします。

IdM は、RFC 5936 (AXFR) および RFC 1995 (IXFR) 標準に準拠するゾーン転送をサポートします。

関連情報

- IdM Web UI でのゾーン転送の有効化 を参照してください。
- IdM CLI でのゾーン転送の有効化 を参照してください。

92.11. IDM WEB UI でのゾーン転送の有効化

IdM Web UI を使用して Identity Management (IdM) でゾーン転送を有効にするには、次の手順に従います。

前提条件

- IdM 管理者としてログインしている。

手順

1. IdM Web UI で、**Network Services → DNS → DNS Zones** の順にクリックします。
2. **Settings** をクリックします。
3. **Allow transfer** で、ゾーンレコードを転送するネームサーバーを指定します。

   図92.7 ゾーン転送の有効化

4. DNS ゾーンページの上部にある **Save** をクリックして、新しい設定を確定します。

92.12. IDM CLI でのゾーン転送の有効化

IdM コマンドラインインターフェイス (CLI) を使用して Identity Management (IdM) でゾーン転送を有効にするには、次の手順に従います。

前提条件

- IdM 管理者としてログインしている。
- セカンダリー DNS サーバーへの root アクセス権限がある。

手順

- **BIND** サービスでゾーン転送を有効にするには、**ipa dnszone-mod** コマンドを入力し、ゾーンレコードの転送先となる転送予定ゾーンに含まれないサーバー名のリストを **--allow-transfer** オプションを使用して指定します。以下に例を示します。

```
$ ipa dnszone-mod --allow-transfer=192.0.2.1;198.51.100.1;203.0.113.1
idm.example.com
```

検証手順

1. ゾーン転送が有効な DNS サーバーの 1 つに SSH 接続します。
   -

```
$ ssh 192.0.2.1
```

2. **dig** ユーティリティーなどのツールを使用して、IdM DNS ゾーンを転送します。

```
# dig @ipa-server zone_name AXFR
```

コマンドでエラーが返されない場合は、**zone_name** のゾーン転送が正常に有効化されています。

92.13. 関連情報

- Using Ansible playbooks to manage IdM DNS zones を参照してください。

第93章 ANSIBLE PLAYBOOK を使用した IDM DNS ゾーンの管理

Identity Management (IdM) 管理者は、**ansible-freeipa** パッケージに含まれる **dnszone** モジュールを使用して IdM DNS ゾーンの動作を管理できます。

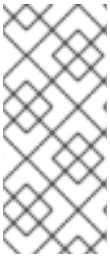
- IdM でサポートされる DNS ゾーンタイプ
- IdM で設定できる DNS 属性
- Ansible Playbook を使用して IdM DNS にプライマリーゾーンを作成する方法
- Ansible Playbook を使用して複数の変数が含まれるプライマリー IdM DNS ゾーンを存在させる手順
- IP アドレスが指定されている場合に Ansible Playbook を使用して逆引き DNS ルックアップのゾーンを存在させる手順

前提条件

- DNS サービスが IdM サーバーにインストールされている。Red Hat Ansible Engine を使用して、統合 DNS のある IdM サーバーをインストールする方法は、Ansible Playbook を使用した Identity Management サーバーのインストール を参照してください。

93.1. サポート対象の DNS ゾーンタイプ

Identity Management (IdM) は、2 種類の DNS ゾーン (**primary** および **forward**) をサポートします。ここでは、DNS 転送のシナリオ例を含め、2 種類のゾーンについて説明します。



注記

本ガイドでは、ゾーンタイプには BIND の用語を使用し、Microsoft Windows DNS で使用する用語とは異なります。BIND のプライマリーゾーンは、Microsoft Windows DNS の **正引きルックアップゾーン** と **逆引きルックアップゾーン** と同じ目的で使用されます。BIND の正引きゾーンは、Microsoft Windows DNS の **条件付きフォワーダー** と同じ目的で使用されます。

プライマリー DNS ゾーン

プライマリー DNS ゾーンには、権威 DNS データが含まれ、DNS を動的に更新できます。この動作は、標準 BIND 設定の **type master** 設定と同じです。プライマリーゾーンは、**ipa dnszone-*** コマンドを使用して管理できます。
標準 DNS ルールに準拠するには、プライマリーゾーンすべてに **start of authority** (SOA) と **nameserver** (NS) レコードを含める必要があります。IdM では、DNS ゾーンの作成時にこれらのレコードが自動的に生成されますが、NS レコードを親ゾーンに手動でコピーして適切な委譲を作成する必要があります。

標準の BIND の動作に合わせて、権威サーバーではない名前のクエリーは、他の DNS サーバーに転送されます。DNS サーバー (別称: フォワーダー) は、クエリーに対して権威がある場合と、ない場合があります。

> **例93.1 DNS 転送のシナリオ例**
>
> IdM サーバーには **test.example.** プライマリーゾーンが含まれています。このゾーンには、**sub.test.example.** 名前の NS 委譲レコードが含まれます。さらに、**test.example.** ゾーンは、**sub.text.example** サブゾーンのフォワーダー IP アドレス **192.0.2.254** で設定されます。

> クライアントが **nonexistent.test.example.** の名前をクエリーすると、**NXDomain** の応答を受け取りますが、IdM サーバーはこの名前に対して権威があるため、転送は発生しません。
>
> 反対に、**host1.sub.test.example.** の名前をクエリーすると、IdM サーバーはこの名前に対して権威がないので、設定済みのフォワーダー (**192.0.2.254**) に転送されます。

正引き DNS ゾーン

IdM の観点からは、正引き DNS ゾーンには権威データは含まれません。実際、正引きのゾーンには、通常以下情報 2 つのみが含まれます。

- ドメイン名
- ドメインに関連付けられた DNS サーバーの IP アドレス

定義済みのドメインに所属する名前のクエリーはすべて、指定の IP アドレスに転送されます。この動作は、標準 BIND 設定の **type forward** 設定と同じです。正引きゾーンは、**ipa dnsforwardzone-*** コマンドを使用して管理できます。

正引き DNS ゾーンは、IdM-Active Directory (AD) 信頼のコンテキストで特に便利です。IdM DNS サーバーが **idm.example.com** ゾーンに対して、AD DNS サーバーが **ad.example.com** ゾーンに対して権威がある場合には、**ad.example.com** が **idm.example.com** プライマリーゾーンの DNS 正引きゾーンになります。つまり、IP アドレスが **somehost.ad.example.com** の IdM クライアントからクエリーが送信されると、**ad.example.com** IdM DNS 正引きゾーンに指定の AD ドメインコントローラーに転送されます。

93.2. プライマリー IDM DNS ゾーンの設定属性

Identity Management (IdM) は、更新期間、転送設定、キャッシュ設定など、特定のデフォルト設定を指定して新しいゾーンを作成します。IdM DNS ゾーン属性 には、デフォルトのゾーン設定属性があります。これは、以下のオプションのいずれかを使用して変更できます。

- コマンドラインインターフェイス (CLI) の **dnszone-mod** コマンド詳細は IdM CLI でのプライマリー DNS ゾーンの設定の編集 を参照してください。
- IdM Web UI詳細は IdM Web UI でのプライマリー DNS ゾーンの設定の編集 を参照してください。
- **ipadnszone** モジュールを使用する Ansible Playbook詳細は、IdM での DNS ゾーンの管理 を参照してください。

ここではゾーンの実際の情報を設定するほか、DNS サーバーが **start of authority** (SOA) レコードエントリーを処理する方法と、DNS ネームサーバーからの記録を更新する方法を定義します。

表93.1 IdM DNS ゾーン属性

属性	ansible-freeipa 変数	説明

属性	ansible-freeipa 変数	説明
権威ネームサーバー	**name_server**	プライマリー DNS ネームサーバーのドメイン名 (別称: SOA MNAME) を設定します。 デフォルトでは、各 IdM サーバーは SOA MNAME フィールドで自己アドバタイズします。そのため、**--name-server** を使用して LDAP に保存されている値は無視されます。
管理者の電子メールアドレス	**admin_email**	ゾーン管理者が使用する電子メールアドレスを設定します。デフォルトでは、ホストの root アカウントになります。
SOA serial	**serial**	SOA レコードにシリアル番号を設定します。IdM ではバージョン番号が自動的に設定され、この番号のユーザーによる変更は想定されていません。
SOA refresh	**refresh**	セカンダリー DNS サーバーがプライマリー DNS サーバーから更新を要求するまでの待機時間を秒単位で設定します。
SOA retry	**retry**	失敗した更新操作を再試行するまでに待機する時間を秒単位で設定します。
SOA expire	**expire**	セカンダリー DNS サーバーが操作の試行を終了するまでに、更新操作を実行する時間を秒単位で設定します。
SOA minimum	**minimum**	RFC 2308 に準拠し、ネガティブキャッシュの TTL (TTL) 値を秒単位で設定します。
SOA time to live	**ttl**	ゾーン apex のレコードの TTL を秒単位で設定します。たとえば、**example.com** ゾーンでは、名前が **example.com** のすべてのレコード (A、NS または SOA) が設定されますが、**test.example.com** などの他のドメイン名には影響はありません。
デフォルトの TTL	**default_ttl**	これまでに個別の Time To Live (TTL) 値が設定されたことのないゾーンで、すべての値のネガティブキャッシュのデフォルト TTL を秒単位で設定します。変更を有効にするには、すべての IdM DNS サーバーで **named-pkcs11** サービスを再起動する必要があります。
BIND 更新ポリシー	**update_policy**	DNS ゾーンでクライアントに許可されるパーミッションを設定します。
Dynamic update	**dynamic_update**=TRUE|FALSE	クライアントの DNS レコードへの動的更新を有効にします。 false に設定すると、IdM クライアントマシンは IP アドレスを追加または更新できなくなる点に注意してください。

属性	ansible-freeipa 変数	説明
Allow transfer	**allow_transfer**=string	指定のゾーンを転送できる IP アドレスまたはネットワーク名のセミコロン区切りのリストを指定します。 デフォルトでは、ゾーン転送は無効です。**allow_transfer** のデフォルト値は **none** です。
Allow query	**allow_query**	DNS クエリーを発行できる IP アドレスまたはネットワーク名のセミコロン区切りのリストを指定します。
Allow PTR sync	**allow_sync_ptr**=1|0	ゾーンの A または AAAA レコード (正引きレコード) が自動的に PTR (逆引き) レコードと同期されるかどうかを設定します。
Zone forwarder	**forwarder**=IP_address	DNS ゾーン向けに特別に設定されたフォワーダーを指定します。これは、IdM ドメインで使用されるグローバルフォワーダーとは別のものです。 複数のフォワーダーを指定する場愛には、オプションを複数回使用します。
転送ポリシー	**forward_policy**=none|only|first	転送ポリシーを指定します。サポート対象のポリシーに関する情報は、IdM での DNS 転送ポリシー を参照してください。

関連情報

- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-dnszone.md** ファイルを参照してください。

93.3. ANSIBLE を使用した IDM DNS でのプライマリーゾーンの作成

以下の手順に従って、Ansible Playbook を使用して、プライマリー DNS ゾーンが存在することを確認します。以下の手順で使用する例では、**zone.idm.example.com** DNS ゾーンが存在するようにします。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- IdM 管理者パスワードを把握している。

手順

1. **/usr/share/doc/ansible-freeipa/playbooks/dnszone** ディレクトリーに移動します。

```
$ cd /usr/share/doc/ansible-freeipa/playbooks/dnszone
```

2. インベントリーファイルを開き、設定する IdM サーバーが **[ipaserver]** セクションに記載されていることを確認します。たとえば、Ansible に対して **server.idm.example.com** を設定するように指示するには、次のコマンドを実行します。

```
[ipaserver]
server.idm.example.com
```

3. Ansible Playbook ファイルのコピー (**dnszone-present.yml**) を作成します。以下に例を示します。

```
$ cp dnszone-present.yml dnszone-present-copy.yml
```

4. **dnszone-present-copy.yml** ファイルを開いて編集します。

5. **ipadnszone** タスクセクションに以下の変数を設定してファイルを調整します。

   - **ipaadmin_password** 変数は IdM 管理者パスワードに設定します。
   - **zone_name** 変数は **zone.idm.example.com** に設定します。
     以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Ensure dnszone present
  hosts: ipaserver
  become: true

  tasks:
  - name: Ensure zone is present.
    ipadnszone:
      ipaadmin_password: "{{ ipaadmin_password }}"
      zone_name: zone.idm.example.com
      state: present
```

6. ファイルを保存します。

7. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file dnszone-present-copy.yml
```

関連情報

- サポート対象の DNS ゾーンタイプ を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-dnszone.md** ファイルを参照してください。

- **/usr/share/doc/ansible-freeipa/playbooks/dnszone** ディレクトリーのサンプルの Ansible Playbook を参照してください。

93.4. ANSIBLE PLAYBOOK を使用して、変数が複数ある IDM にプライマリー DNS ゾーンを存在させる手順

以下の手順に従って、Ansible Playbook を使用して、プライマリー DNS ゾーンが存在することを確認します。以下の手順で使用する例では、IdM 管理者は **zone.idm.example.com** の DNS ゾーンを追加します。Ansible Playbook は、ゾーンのパラメーターを複数設定します。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- IdM 管理者パスワードを把握している。

手順

1. **/usr/share/doc/ansible-freeipa/playbooks/dnszone** ディレクトリーに移動します。

```
$ cd /usr/share/doc/ansible-freeipa/playbooks/dnszone
```

2. インベントリーファイルを開き、設定する IdM サーバーが **[ipaserver]** セクションに記載されていることを確認します。たとえば、Ansible に対して **server.idm.example.com** を設定するように指示するには、次のコマンドを実行します。

```
[ipaserver]
server.idm.example.com
```

3. Ansible Playbook ファイルのコピー (**dnszone-all-params.yml**) を作成します。以下に例を示します。

```
$ cp dnszone-all-params.yml dnszone-all-params-copy.yml
```

4. **dnszone-all-params-copy.yml** ファイルを開いて編集します。
5. **ipadnszone** タスクセクションに以下の変数を設定してファイルを調整します。
   - **ipaadmin_password** 変数は IdM 管理者パスワードに設定します。
   - **zone_name** 変数は zone.idm.example.com に設定します。

- 正引きレコードと逆引きレコードを同期できるように場合は (A および AAAA レコードを PTR レコードと同期)、**allow_sync_ptr** 変数を true に設定します。
- **dynamic_update** 変数は、true に設定して、IdM クライアントマシンが IP アドレスを追加または更新できるようにします。
- **dnssec** 変数は、true に設定して、ゾーン内のレコードのインラインの DNSSEC 署名を許可します。
- **allow_transfer** 変数は、ゾーン内のセカンダリーネームサーバーの IP アドレスに設定します。
- **allow_query** 変数は、クエリーを発行できる IP アドレスまたはネットワークに設定します。
- **forwarders** 変数は、グローバルフォワーダー の IP アドレスに設定します。
- **serial** 変数は SOA レコードのシリアル番号に設定します。
- ゾーン内の DNS レコードの **refresh**、**retry**、**expire**、**minimum**、**ttl** および **default_ttl** の値を定義します。
- **nsec3param_rec** 変数を使用して、ゾーンの NSEC3PARAM レコードを定義します。
- **skip_overlap_check** 変数は、true に設定して、既存のゾーンと重複していても DNS を強制的に作成します。
- **skip_nameserver_check** は、true に設定して、ネームサーバーが解決できない場合でも DNS ゾーンを強制的に作成します。
  以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Ensure dnszone present
  hosts: ipaserver
  become: true

  tasks:
  - name: Ensure zone is present.
    ipadnszone:
      ipaadmin_password: "{{ ipaadmin_password }}"
      zone_name: zone.idm.example.com
      allow_sync_ptr: true
      dynamic_update: true
      dnssec: true
      allow_transfer:
        - 1.1.1.1
        - 2.2.2.2
      allow_query:
        - 1.1.1.1
        - 2.2.2.2
      forwarders:
        - ip_address: 8.8.8.8
        - ip_address: 8.8.4.4
          port: 52
      serial: 1234
      refresh: 3600
      retry: 900
```

```
    expire: 1209600
    minimum: 3600
    ttl: 60
    default_ttl: 90
    name_server: server.idm.example.com.
    admin_email: admin.admin@idm.example.com
    nsec3param_rec: "1 7 100 0123456789abcdef"
    skip_overlap_check: true
    skip_nameserver_check: true
    state: present
```

6. ファイルを保存します。

7. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file dnszone-all-params-copy.yml
```

関連情報

- サポート対象の DNS ゾーンタイプ を参照してください。
- プライマリー IdM DNS ゾーンの設定属性 を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-dnszone.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/dnszone** ディレクトリーのサンプルの Ansible Playbook を参照してください。

93.5. IP アドレスが指定されている場合に ANSIBLE PLAYBOOK を使用して逆引き DNS ルックアップのゾーンを存在させる手順

以下の手順に従って、Ansible Playbook を使用して、逆引き DNS ゾーンが存在することを確認します。以下の手順で使用する例では、IdM 管理者は、IdM ホストの IP アドレスおよび接頭辞長を使用して、逆引き DNS ルックアップゾーンを追加します。

name_from_ip 変数を使用して DNS サーバーの IP アドレスの接頭辞の長さを指定すると、ゾーン名を制御できます。接頭辞の長さを指定しない場合には、システムが DNS サーバーにゾーンに関するクエリーを出し、192.168.1.2 の **name_from_ip** の値をもとに、このクエリーで、以下の DNS ゾーンのいずれかを返します。

- 1.168.192.in-addr.arpa.
- 168.192.in-addr.arpa.
- 192.in-addr.arpa.

クエリーが返すゾーンは想定しているゾーンとは異なる可能性があるため、ゾーンが誤って削除されないように **state** オプションが **present** に設定されている場合のみ、**name_from_ip** を使用できます。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。

- Ansible バージョン 2.14 以降を使用している。
- Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
- この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
- この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。

- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- IdM 管理者パスワードを把握している。

手順

1. **/usr/share/doc/ansible-freeipa/playbooks/dnszone** ディレクトリーに移動します。

```
$ cd /usr/share/doc/ansible-freeipa/playbooks/dnszone
```

2. インベントリーファイルを開き、設定する IdM サーバーが **[ipaserver]** セクションに記載されていることを確認します。たとえば、Ansible に対して **server.idm.example.com** を設定するように指示するには、次のコマンドを実行します。

```
[ipaserver]
server.idm.example.com
```

3. Ansible Playbook ファイルのコピー (**dnszone-reverse-from-ip.yml**) を作成します。以下に例を示します。

```
$ cp dnszone-reverse-from-ip.yml dnszone-reverse-from-ip-copy.yml
```

4. **dnszone-reverse-from-ip-copy.yml** ファイルを開いて編集します。
5. **ipadnszone** タスクセクションに以下の変数を設定してファイルを調整します。
   - **ipaadmin_password** 変数は IdM 管理者パスワードに設定します。
   - **name_from_ip** 変数は IdM ネームサーバーの IP に設定し、接頭辞の長さを指定します。以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Ensure dnszone present
  hosts: ipaserver
  become: true

  tasks:
  - name: Ensure zone for reverse DNS lookup is present.
    ipadnszone:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name_from_ip: 192.168.1.2/24
      state: present
    register: result
```

```
- name: Display inferred zone name.
  debug:
    msg: "Zone name: {{ result.dnszone.name }}"
```

この Playbook は、IP アドレス **192.168.1.2** と接頭辞長 24 をもとに、逆引き DNS ルックアップのゾーンを作成します。次に、Playbook は生成されたゾーン名を表示します。

6. ファイルを保存します。

7. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file dnszone-reverse-from-ip-copy.yml
```

関連情報

- サポート対象の DNS ゾーンタイプ を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-dnszone.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/dnszone** ディレクトリーのサンプルの Ansible Playbook を参照してください。

第94章 IDM での DNS の場所の管理

IdM Web UI および IdM コマンドラインインターフェイス (CLI) を使用して Identity Management (IdM) DNS の場所を管理する方法について詳しくは、次のトピックと手順を参照してください。

- DNS ベースのサービス検出
- DNS の場所のデプロイに関する考慮事項
- DNS の Time to live (TTL)
- IdM Web UI を使用した DNS の場所の作成
- IdM CLI を使用した DNS の場所の作成
- IdM Web UI を使用した DNS の場所への IdM サーバーの割り当て
- IdM Web UI を使用した DNS の場所への IdM サーバーの割り当て
- IdM クライアントが同じ場所にある IdM サーバーを使用するように設定する手順

94.1. DNS ベースのサービス検出

DNS ベースのサービス検出は、クライアントが DNS プロトコルを使用するプロセスで、**LDAP** や **Kerberos** など、特定のサービスを提供するネットワークでサーバーを見つけ出します。一般的な操作の 1 つとして、クライアントが最寄りのネットワークインフラストラクチャー内にある認証サーバーを特定できるようにすることが挙げられます。理由は、スループットが向上してネットワークレイテンシーが短縮されるので全体的なコスト削減を図ることができるためです。

サービス検出の主な利点は以下のとおりです。

- 近くにあるサーバーの名前を明示的に設定する必要がない。
- DNS サーバーをポリシーの中央プロバイダーとして使用する。同じ DNS サーバーを使用するクライアントは、サービスプロバイダーと優先順序に関する同じポリシーにアクセスできます。

Identity Management (IdM) ドメインには、**LDAP**、**Kerberos**、およびその他のサービスに DNS サービスレコード (SRV レコード) があります。たとえば、次のコマンドは、IdM DNS ドメインで TCP ベースの **Kerberos** サービスを提供するホストの DNS サーバーをクエリーします。

例94.1 DNS の場所に関する独立した結果

```
$ dig -t SRV +short _kerberos._tcp.idm.example.com
0 100 88 idmserver-01.idm.example.com.
0 100 88 idmserver-02.idm.example.com.
```

出力には、以下の情報が含まれます。

- **0** (優先度): ターゲットホストの優先度。値が小さいほど優先度が高くなります。
- **100** (加重)。優先順位が同じエントリーの相対的な重みを指定します。詳細は RFC 2782, section 3 を参照してください。
- **88** (ポート番号): サービスのポート番号

- サービスを提供するホストの正規名。

この例では、2 つのホスト名が返され、どちらも同じ優先順位と重みでした。この場合には、クライアントは結果リストから無作為にエントリーを使用します。

代わりに、クライアントを設定して、DNS の場所に設定されている DNS サーバーをクエリーすると、出力が異なります。場所が割り当てられた IdM サーバーの場合は、カスタマイズした値が返されます。以下の例では、クライアントは、場所 **germany** にある DNS サーバーをクエリーするように設定されています。

例94.2 DNS の場所ベースの結果

```
$ dig -t SRV +short _kerberos._tcp.idm.example.com
_kerberos._tcp.germany._locations.idm.example.com.
0 100 88 idmserver-01.idm.example.com.
50 100 88 idmserver-02.idm.example.com.
```

IdM DNS サーバーは、ローカルサーバーを優先する DNS の場所固有の SRV レコードを参照する DNS エイリアス (CNAME) を自動的に返します。この CNAME レコードは、出力の最初の行に表示されます。この例では、ホスト**idmserver-01.idm.example.com**の優先度の値が最も低いため、優先されます。**idmserver-02.idm.example.com** の優先度の値が高く、推奨されるホストが使用できない場合にバックアップとしてのみ使用されます。

94.2. DNS の場所のデプロイに関する考慮事項

Identity Management (IdM) は、統合 DNS を使用する際に、場所固有のサービス (SRV) レコードを生成できます。各 IdM DNS サーバーはロケーション固有の SRV レコードを生成するため、DNS の場所ごとに 1 つ以上の IdM DNS サーバーをインストールする必要があります。

クライアントの DNS の場所に対するアフィニティーは、クライアントが受け取った DNS レコードでのみ定義されます。そのため、DNS のサービス検出を行うクライアントが、IdM DNS サーバーからの場所固有のレコードを解決した場合には、IdM DNS サーバーと IdM 以外の DNS コンシューマーサーバーと recursor を組み合わせることができます。

IdM サービスおよび IdM DNS サービス以外のほとんどのデプロイメントでは、DNS recursor はラウンドトリップタイム (RTT) メトリックを使用して、最寄りの IdM DNS サーバーを自動的に選択します。通常、IdM DNS サーバーを使用するクライアントが、最寄りの DNS の場所のレコードを取得し、最寄りの DNS サーバーの最適なセットを使用するようになります。

94.3. DNS の TIME TO LIVE (TTL)

クライアントは、ゾーンの設定に指定された期間の DNS リソースレコードをキャッシュできます。このキャッシュにより、クライアントは Time to Live (TTL) 値の有効期限が切れるまで変更を受け取れない場合があります。Identity Management (IdM) のデフォルトの TTL 値は **1 日** です。

クライアントコンピューターがサイト間でローミングする場合には、IdM DNS ゾーンの TTL 値を調整する必要があります。この値は、クライアントがサイト間のローミングに必要とする時間よりも低い値に設定します。これにより、別のサイトに再接続する前にクライアントでキャッシュされた DNS エントリーが期限切れになり、DNS サーバーに対してクエリーを実行し、場所固有の SRV レコードを更新します。

関連情報

- プライマリー IdM DNS ゾーンの設定属性 を参照してください。

94.4. IDM WEB UI を使用した DNS の場所の作成

DNS の場所を使用すると、Identity Management (IdM) クライアントとサーバー間の通信速度を増すことができます。IdM Web UI を使用して DNS ロケーションを作成するには、この手順に従ってください。

前提条件

- IdM デプロイメントに DNS が統合されている。
- IdM で DNS の場所を作成するパーミッションがある。(例: IdM 管理者としてログイン)。

手順

1. **IPA Server** タブを開きます。
2. **Topology** サブタブを選択します。
3. ナビゲーションバーの **IPA の場所** をクリックします。
4. ロケーションリストの上部にある **追加** をクリックします。
5. ロケーション名を入力します。
6. **追加** ボタンをクリックして場所を保存します。
7. 必要に応じて、手順を繰り返して、さらに場所を追加します。

関連情報

- IdM Web UI を使用した DNS の場所への IdM サーバーの割り当て を参照してください。
- Ansible を使用して IdM の場所が存在することを確認する を参照してください。

94.5. IDM CLI を使用した DNS の場所の作成

DNS の場所を使用すると、Identity Management (IdM) クライアントとサーバー間の通信速度を増すことができます。IdM コマンドラインインターフェイス (CLI) で **ipa location-add** コマンドを使用して DNS ロケーションを作成するには、この手順に従います。

前提条件

- IdM デプロイメントに DNS が統合されている。
- IdM で DNS の場所を作成するパーミッションがある。(例: IdM 管理者としてログイン)。

手順

1. たとえば、新しい場所 **germany** を作成するには、以下を入力します。

```
$ ipa location-add germany
----------------------------
```

```
Added IPA location "germany"
----------------------------
  Location name: germany
```

2. 必要に応じて、手順を繰り返して、さらに場所を追加します。

関連情報

- IdM CLI を使用した DNS の場所への IdM サーバーの割り当て を参照してください。
- Ansible を使用して IdM の場所が存在することを確認する を参照してください。

94.6. IDM WEB UI を使用した DNS の場所への IDM サーバーの割り当て

Identity Management (IdM) の DNS の場所を使用すると、IdM クライアントとサーバー間の通信速度を増すことができます。IdM Web UI を使用して IdM サーバーを DNS ロケーションに割り当てるには、この手順に従います。

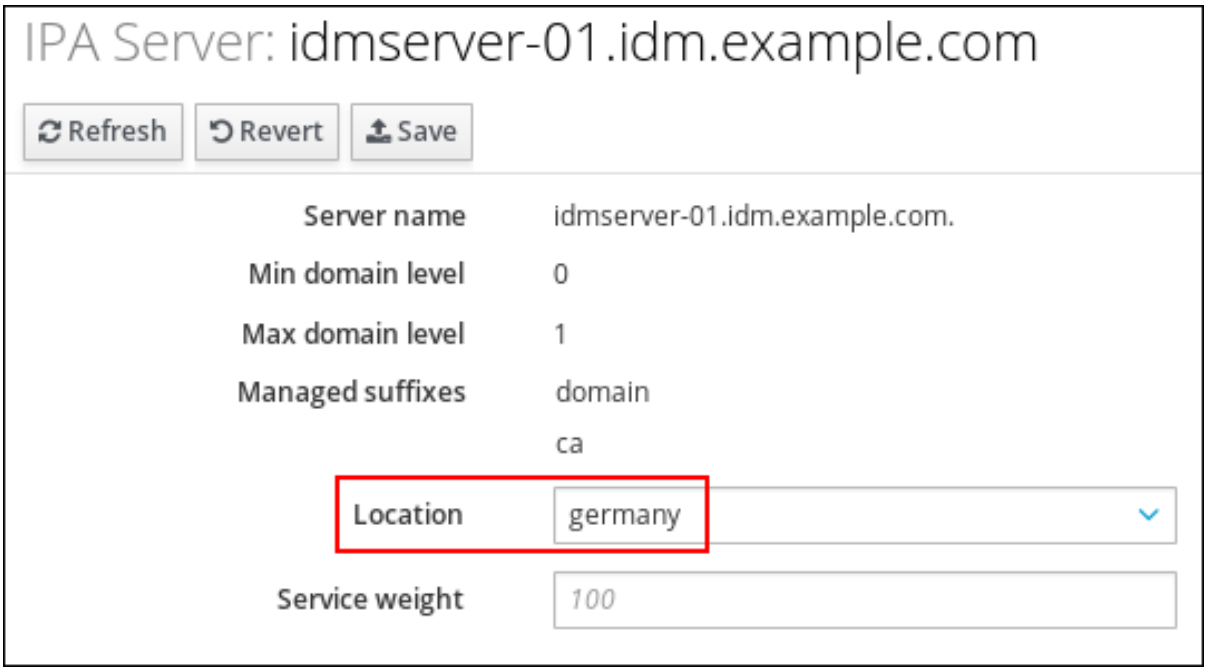
前提条件

- IdM デプロイメントに DNS が統合されている。
- たとえば、IdM admin ユーザーなど、DNS の場所を割り当てるパーミッションがあるユーザーとしてログインしている。
- DNS の場所を割り当てるホストへの **root** アクセス権がある。
- サーバーを割り当てる IdM DNS の場所を作成 している。

手順

1. **IPA Server** タブを開きます。
2. **Topology** サブタブを選択します。
3. ナビゲーションにある **IPA Servers** をクリックします。
4. IdM サーバー名をクリックします。
5. DNS の場所を選択し、必要に応じてサービスの加重を設定します。

図94.1 DNS の場所へのサーバーの割り当て



6. **Save** をクリックします。
7. 前の手順で DNS の場所を割り当てたホストのコマンドラインインターフェイス (CLI) で、**named-pkcs11** サービスを再起動します。

```
[root@idmserver-01 ~]# systemctl restart named-pkcs11
```

8. オプション: この手順を繰り返して、他の IdM サーバーに DNS の場所を割り当てます。

関連情報

- IdM クライアントが同じ場所にある IdM サーバーを使用するように設定する手順 を参照してください。

94.7. IDM CLI を使用した DNS の場所への IDM サーバーの割り当て

Identity Management (IdM) の DNS の場所を使用すると、IdM クライアントとサーバー間の通信速度を増すことができます。IdM コマンドラインインターフェイス (CLI) を使用して IdM サーバーを DNS の場所に割り当てるには、次の手順に従います。

前提条件

- IdM デプロイメントに DNS が統合されている。
- たとえば、IdM admin ユーザーなど、DNS の場所を割り当てるパーミッションがあるユーザーとしてログインしている。
- DNS の場所を割り当てるホストへの **root** アクセス権がある。
- サーバーを割り当てる IdM DNS の場所を作成 している。

手順

1. オプション: 設定済みの DNS の場所をすべて表示します。

```
[root@server ~]# ipa location-find
----------------------
2 IPA locations matched
----------------------
Location name: australia
Location name: germany
----------------------------
Number of entries returned: 2
----------------------------
```

2. サーバーを DNS の場所に割り当てます。たとえば、場所 **germany** を **idmserver-01.idm.example.com** サーバーに割り当てるには、以下を実行します。

```
# ipa server-mod idmserver-01.idm.example.com --location=germany
ipa: WARNING: Service named-pkcs11.service requires restart on IPA server
idmserver-01.idm.example.com to apply configuration changes.
-----------------------------------------------
Modified IPA server "idmserver-01.idm.example.com"
-----------------------------------------------
Servername: idmserver-01.idm.example.com
Min domain level: 0
Max domain level: 1
Location: germany
Enabled server roles: DNS server, NTP server
```

3. 前の手順で DNS の場所を割り当てたホストで **named-pkcs11** サービスを再起動します。

```
# systemctl restart named-pkcs11
```

4. オプション: この手順を繰り返して、他の IdM サーバーに DNS の場所を割り当てます。

関連情報

- IdM クライアントが同じ場所にある IdM サーバーを使用するように設定する手順 を参照してください。

94.8. IDM クライアントが同じ場所にある IDM サーバーを使用するように設定する手順

Identity Management (IdM) サーバーは、IdM Web UI を使用した DNS の場所への IdM サーバーの割り当て で説明されているように、DNS の場所に割り当てます。これで、IdM サーバーと同じ場所にある DNS サーバーを使用するようにクライアントを設定できます。

- **DHCP** サーバーが DNS サーバーの IP アドレスをクライアントに割り当てる場合は、**DHCP** サービスを設定します。**DHCP** サービスで DNS サーバーを割り当てる方法は、**DHCP** サービスのドキュメントを参照してください。

- クライアントに **DHCP** サーバーから DNS サーバーの IP アドレスが割り当てられない場合は、クライアントのネットワーク設定で IP を手動で設定します。Red Hat Enterprise Linux でネットワークを設定する方法は、**Red Hat Enterprise Linux ネットワークガイド**の ネットワー接続の設定 セクションを参照してください。

注記

別のロケーションに割り当てられた DNS サーバーを使用するようにクライアントを設定すると、クライアントは両方の場所にある IdM サーバーに接続します。

例94.3 クライアントの場所により変化するネームサーバーエントリー

以下の例は、場所が異なるクライアントの **/etc/resolv.conf** ファイルにあるさまざまなネームサーバーエントリーを示しています。

プラハのクライアント:

```
nameserver 10.10.0.1
nameserver 10.10.0.2
```

パリのクライアント:

```
nameserver 10.50.0.1
nameserver 10.50.0.3
```

オスロのクライアント:

```
nameserver 10.30.0.1
```

ベルリンのクライアント:

```
nameserver 10.30.0.1
```

各 DNS サーバーが IdM の場所に割り当てられている場合に、クライアントはその場所にある IdM サーバーを使用します。

94.9. 関連情報

- Ansible を使用した IdM での DNS の場所の管理 を参照してください。

第95章 ANSIBLE を使用した IDM での DNS の場所の管理

Identity Management (IdM) 管理者は、**ansible-freeipa** パッケージで利用可能な **location** モジュールを使用して IdM DNS の場所を管理できます。

- DNS ベースのサービス検出
- DNS の場所のデプロイに関する考慮事項
- DNS の Time to live (TTL)
- Ansible を使用して IdM の場所が存在することを確認する
- Ansible を使用して IdM の場所を削除する手順

95.1. DNS ベースのサービス検出

DNS ベースのサービス検出は、クライアントが DNS プロトコルを使用するプロセスで、**LDAP** や **Kerberos** など、特定のサービスを提供するネットワークでサーバーを見つけ出します。一般的な操作の 1 つとして、クライアントが最寄りのネットワークインフラストラクチャー内にある認証サーバーを特定できるようにすることが挙げられます。理由は、スループットが向上してネットワークレイテンシーが短縮されるので全体的なコスト削減を図ることができるためです。

サービス検出の主な利点は以下のとおりです。

- 近くにあるサーバーの名前を明示的に設定する必要がない。
- DNS サーバーをポリシーの中央プロバイダーとして使用する。同じ DNS サーバーを使用するクライアントは、サービスプロバイダーと優先順序に関する同じポリシーにアクセスできます。

Identity Management (IdM) ドメインには、**LDAP**、**Kerberos**、およびその他のサービスに DNS サービスレコード (SRV レコード) があります。たとえば、次のコマンドは、IdM DNS ドメインで TCP ベースの **Kerberos** サービスを提供するホストの DNS サーバーをクエリーします。

例95.1 DNS の場所に関する独立した結果

```
$ dig -t SRV +short _kerberos._tcp.idm.example.com
0 100 88 idmserver-01.idm.example.com.
0 100 88 idmserver-02.idm.example.com.
```

出力には、以下の情報が含まれます。

- **0** (優先度): ターゲットホストの優先度。値が小さいほど優先度が高くなります。
- **100** (加重)。優先順位が同じエントリーの相対的な重みを指定します。詳細は RFC 2782, section 3 を参照してください。
- **88** (ポート番号): サービスのポート番号
- サービスを提供するホストの正規名。

この例では、2 つのホスト名が返され、どちらも同じ優先順位と重みでした。この場合には、クライアントは結果リストから無作為にエントリーを使用します。

代わりに、クライアントを設定して、DNS の場所に設定されている DNS サーバーをクエリーすると、出力が異なります。場所が割り当てられた IdM サーバーの場合は、カスタマイズした値が返されます。以下の例では、クライアントは、場所 **germany** にある DNS サーバーをクエリーするように設定されています。

例95.2 DNS の場所ベースの結果

```
$ dig -t SRV +short _kerberos._tcp.idm.example.com
_kerberos._tcp.germany._locations.idm.example.com.
0 100 88 idmserver-01.idm.example.com.
50 100 88 idmserver-02.idm.example.com.
```

IdM DNS サーバーは、ローカルサーバーを優先する DNS の場所固有の SRV レコードを参照する DNS エイリアス (CNAME) を自動的に返します。この CNAME レコードは、出力の最初の行に表示されます。この例では、ホスト**idmserver-01.idm.example.com**の優先度の値が最も低いため、優先されます。**idmserver-02.idm.example.com** の優先度の値が高く、推奨されるホストが使用できない場合にバックアップとしてのみ使用されます。

95.2. DNS の場所のデプロイに関する考慮事項

Identity Management (IdM) は、統合 DNS を使用する際に、場所固有のサービス (SRV) レコードを生成できます。各 IdM DNS サーバーはロケーション固有の SRV レコードを生成するため、DNS の場所ごとに 1 つ以上の IdM DNS サーバーをインストールする必要があります。

クライアントの DNS の場所に対するアフィニティーは、クライアントが受け取った DNS レコードでのみ定義されます。そのため、DNS のサービス検出を行うクライアントが、IdM DNS サーバーからの場所固有のレコードを解決した場合には、IdM DNS サーバーと IdM 以外の DNS コンシューマーサーバーと recursor を組み合わせることができます。

IdM サービスおよび IdM DNS サービス以外のほとんどのデプロイメントでは、DNS recursor はラウンドトリップタイム (RTT) メトリックを使用して、最寄りの IdM DNS サーバーを自動的に選択します。通常、IdM DNS サーバーを使用するクライアントが、最寄りの DNS の場所のレコードを取得し、最寄りの DNS サーバーの最適なセットを使用するようになります。

95.3. DNS の TIME TO LIVE (TTL)

クライアントは、ゾーンの設定に指定された期間の DNS リソースレコードをキャッシュできます。このキャッシュにより、クライアントは Time to Live (TTL) 値の有効期限が切れるまで変更を受け取れない場合があります。Identity Management (IdM) のデフォルトの TTL 値は **1 日** です。

クライアントコンピューターがサイト間でローミングする場合には、IdM DNS ゾーンの TTL 値を調整する必要があります。この値は、クライアントがサイト間のローミングに必要とする時間よりも低い値に設定します。これにより、別のサイトに再接続する前にクライアントでキャッシュされた DNS エントリーが期限切れになり、DNS サーバーに対してクエリーを実行し、場所固有の SRV レコードを更新します。

関連情報

- プライマリー IdM DNS ゾーンの設定属性 を参照してください。

95.4. ANSIBLE を使用して IDM の場所が存在することを確認する

Identity Management (IdM) のシステム管理者は、クライアントが最寄りのネットワークインフラストラクチャーで認証サーバーを特定できるように IdM DNS の場所を設定できます。

以下の手順では、Ansible Playbook を使用して IdM に DNS の場所を追加する方法を説明します。この例では、DNS の場所 **germany** が IdM に存在することを確認する方法を説明します。IdM に DNS の場所を追加して、ローカルの IdM クライアントがサーバーの応答時間を短縮できるように、特定の IdM サーバーをこの場所に割り当てることができます。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- DNS の場所のデプロイメントに関する考慮事項 を理解している。

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/location/** ディレクトリーにある **location-present.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/location/location-present.yml location-present-copy.yml
```

3. Ansible Playbook の **location-present-copy.yml** ファイルを開いて編集します。
4. **ipalocation** タスクセクションに以下の変数を設定して、ファイルを調整します。
   - 使用しているユースケースに合わせて、タスクの **名前** を調節します。
   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は、場所の名前に設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: location present example
  hosts: ipaserver
```

```
vars_files:
- /home/user_name/MyPlaybooks/secret.yml
tasks:
- name: Ensure that the "germany" location is present
  ipalocation:
    ipaadmin_password: "{{ ipaadmin_password }}"
    name: germany
```

5. ファイルを保存します。

6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory location-present-copy.yml
```

関連情報

- IdM Web UI を使用した DNS の場所への IdM サーバーの割り当て または IdM CLI を使用した DNS の場所への IdM サーバーの割り当て を参照してください。

95.5. ANSIBLE を使用して IDM の場所を削除する手順

Identity Management (IdM) のシステム管理者は、クライアントが最寄りのネットワークインフラストラクチャーで認証サーバーを特定できるように IdM DNS の場所を設定できます。

以下の手順では、Ansible Playbook を使用して、IdM から DNS の場所を削除する方法を説明します。この例では、DNS の場所 (**germany**) が IdM から削除されていることを確認する方法を説明します。DNS の場所を削除すると、その場所に、特定の IdM サーバーを割り当てられず、ローカルの IdM クライアントでその場所を使用できなくなります。

前提条件

- IdM 管理者パスワードを把握している。
- **germany** DNS の場所に IdM サーバーが割り当てられていません。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- この例では、サンプルの Playbook のコピーを保存する一元管理場所として **~/MyPlaybooks/** ディレクトリーを 作成して設定 していることを前提とします。

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/location/** ディレクトリーにある **location-absent.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/location/location-absent.yml location-absent-copy.yml
```

3. Ansible Playbook ファイル (**location-absent-copy.yml**) を開きます。

4. **ipalocation** タスクセクションに以下の変数を設定して、ファイルを調整します。

   - 使用しているユースケースに合わせて、タスクの **名前** を調節します。
   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は DNS の場所の名前に設定します。
   - **state** 変数は **absent** に設定されていることを確認します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: location absent example
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure that the "germany" location is absent
    ipalocation:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: germany
      state: absent
```

5. ファイルを保存します。

6. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory location-absent-copy.yml
```

95.6. 関連情報

- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-location.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/location** ディレクトリーのサンプルの Ansible Playbook を参照してください。

第96章 IDM での DNS 転送の管理

以下の手順に従い、Identity Management (IdM) Web UI、IdM CLI、および Ansible を使用して DNS グローバルフォワーダーおよび DNS 正引きゾーンを設定します。

- IdM DNS サーバーの 2 つのロール
- IdM での DNS 転送ポリシー
- IdM Web UI でのグローバルフォワーダーの追加
- CLI でのグローバルフォワーダーの追加
- IdM Web UI での DNS 正引きゾーンの追加
- CLI での DNS 正引きゾーンの追加
- Ansible を使用した IdM での DNS グローバルフォワーダーの確立
- Ansible を使用して IdM に DNS グローバルフォワーダーを存在させる手順
- Ansible を使用して IdM に DNS グローバルフォワーダーを存在させないようにする手順
- Ansible を使用した IdM での DNS グローバルフォワーダーの無効化
- Ansible を使用して IdM に DNS 正引きゾーンを存在させる手順
- Ansible を使用して IdM で DNS 正引きゾーンを複数配置する手順
- Ansible を使用して IdM で DNS 正引きゾーンを無効にする手順
- Ansible を使用して IdM から DNS 正引きゾーンを削除する手順

96.1. IDM DNS サーバーの 2 つのロール

DNS 転送は、DNS サービスが DNS クエリーに応答する方法を左右します。デフォルトでは、IdM と統合されている Berkeley Internet Name Domain (BIND) サービスは、**権威** および **再帰** DNS サーバーの両方として機能します。

権威 DNS サーバー

IdM サーバーが権威のある DNS ゾーンに所属する名前のクエリーを DNS クライアントが出した場合に、BIND は設定済みのゾーンに含まれるデータで応答します。権威データは常に他のデータよりも優先されます。

再帰 DNS サーバー

IdM サーバーが権威のない名前のクエリーを DNS クライアントが出した場合に、BIND は他の DNS サーバーを使用してこのクエリーを解決しようとします。フォワーダーが定義されていない場合は、BIND がインターネット上のルートサーバーにクエリーを出し、再帰解決アルゴリズムを使用して DNS クエリーに応答します。

BIND を使用して他の DNS サーバーに直接問い合わせて、インターネットで利用可能なデータをもとに再帰を実行することは推奨されません。別の DNS サーバーである **フォワーダー** を使用してクエリーを解決するように BIND を設定できます。

フォワーダーを使用するように BIND を設定すると、クエリーと応答が IdM サーバーとフォワーダーの間で送受信され、IdM サーバーが権威データ以外の DNS キャッシュとして機能します。

96.2. IDM での DNS 転送ポリシー

IdM は、**first** および **only** の BIND 転送ポリシーと、IdM 固有の転送ポリシー **none** をサポートします。

forward first (デフォルト)

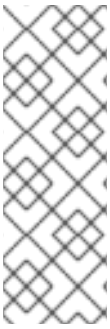
IdM BIND サービスは、DNS クエリーを設定済みのフォワーダーに転送します。サーバーエラーやタイムアウトが原因でクエリーに失敗すると、BIND はインターネット上のサーバーを使用して再帰解決にフォールバックします。**forward first** ポリシーはデフォルトのポリシーで、DNS トラフィックの最適化に適しています。

Forward only

IdM BIND サービスは、DNS クエリーを設定済みのフォワーダーに転送します。サーバーエラーやタイムアウトが原因でクエリーに失敗すると、BIND はエラーをクライアントに返します。分割された DNS 設定の環境では、**forward only** ポリシーが推奨されます。

None (転送の無効化)

DNS クエリーは、**none** 転送ポリシーで転送されません。グローバル転送設定をゾーン別にオーバーライドする場合にのみ、転送の無効化は有用です。このオプションは、IdM の BIND 設定で空のフォワーダーリストを指定するのと同じです。



注記

転送を使用して、IdM のデータと、他の DNS サーバーのデータと統合できません。IdM DNS のプライマリーゾーン内にある特定のサブゾーンのクエリーのみを転送できます。

デフォルトでは、IdM サーバーが権威サーバーとなっているゾーンに、クエリーされた DNS 名が所属する場合には、BIND サービスは、クエリーを別のサーバーに転送しません。このような場合は、クエリーされた DNS 名が IdM データベースに見つからない場合は、**NXDOMAIN** との応答が返されます。転送は使用されません。

例96.1 サンプルシナリオ

IdM サーバーは、**test.example** の権威サーバーです。DNS ゾーン。BIND は、IP アドレス **192.0.2.254** でクエリーを DNS サーバーに転送するように設定されています。

クライアントが **nonexistent.test.example** のクエリーを送信する場合DNS 名である BIND は、IdM サーバーが **test.example.** ゾーンの権威サーバーであることを検出して、クエリーを **192.0.2.254.** サーバーには転送しません。その結果、DNS クライアントは **NXDomain** エラーメッセージを受け取り、クエリーされたドメインが存在しないことをユーザーに通知します。

96.3. IDM WEB UI でのグローバルフォワーダーの追加

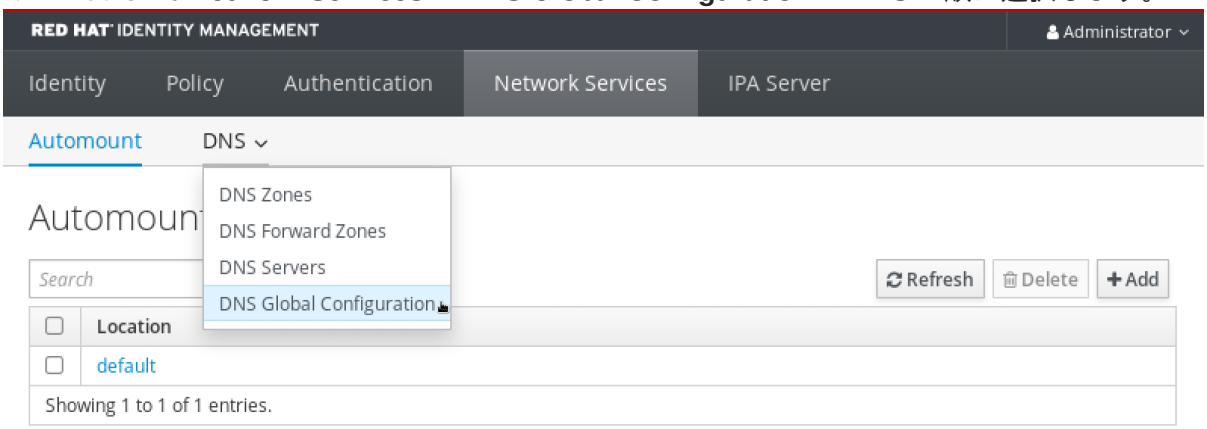
以下の手順に従って、Identity Management (IdM) Web UI でグローバル DNS フォワーダーを追加します。

前提条件

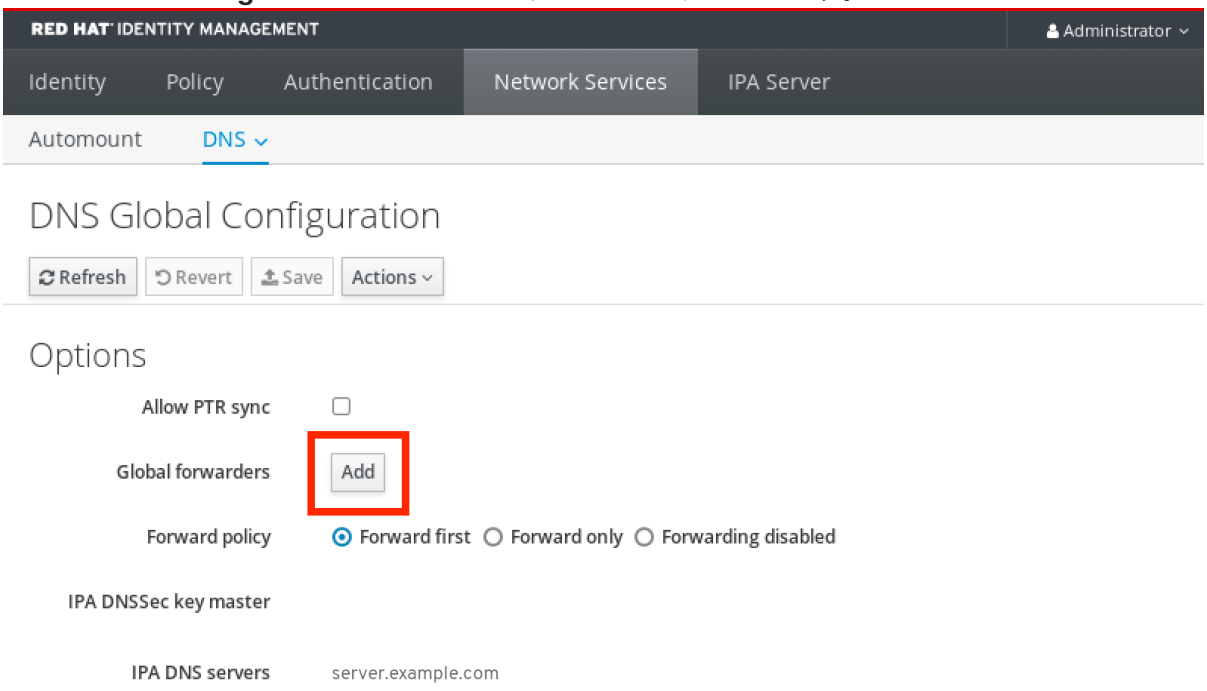
- IdM 管理者として IdM WebUI にログインしている。
- クエリーを転送する DNS サーバーのインターネットプロトコル (IP) アドレスを知っている。

手順

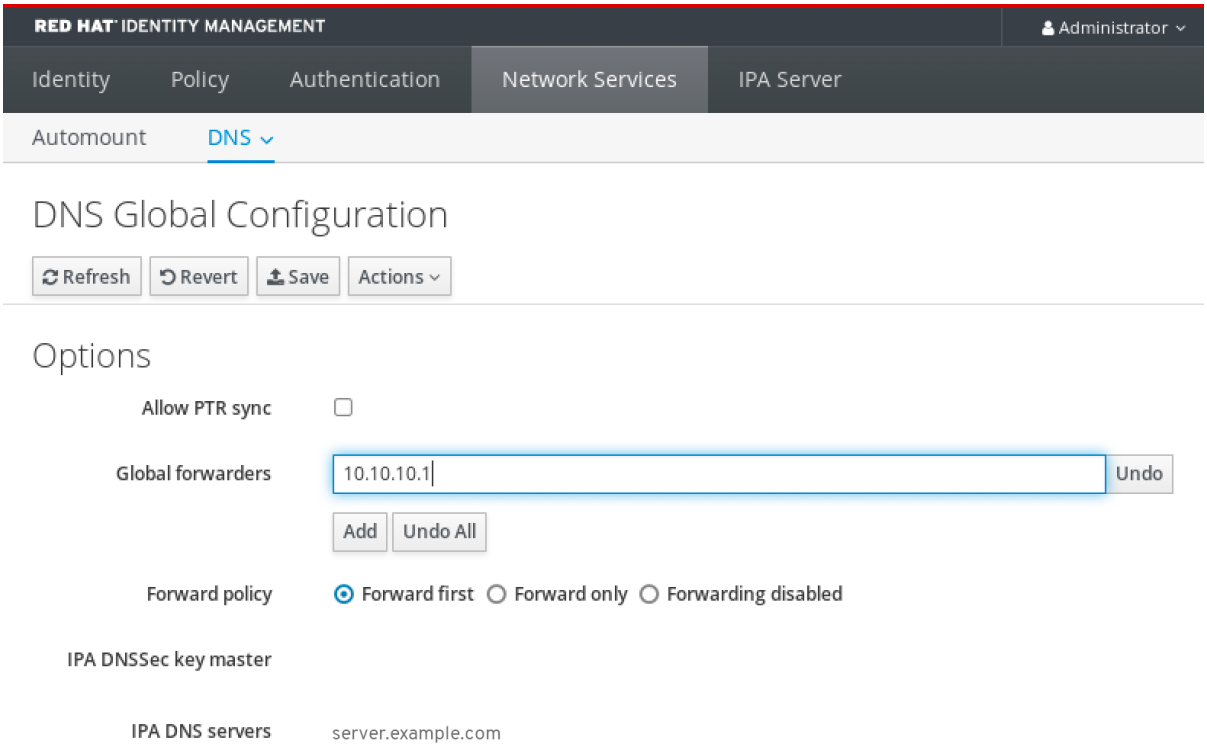
1. IdM Web UI で **Network Services → DNS Global Configuration → DNS** の順に選択します。



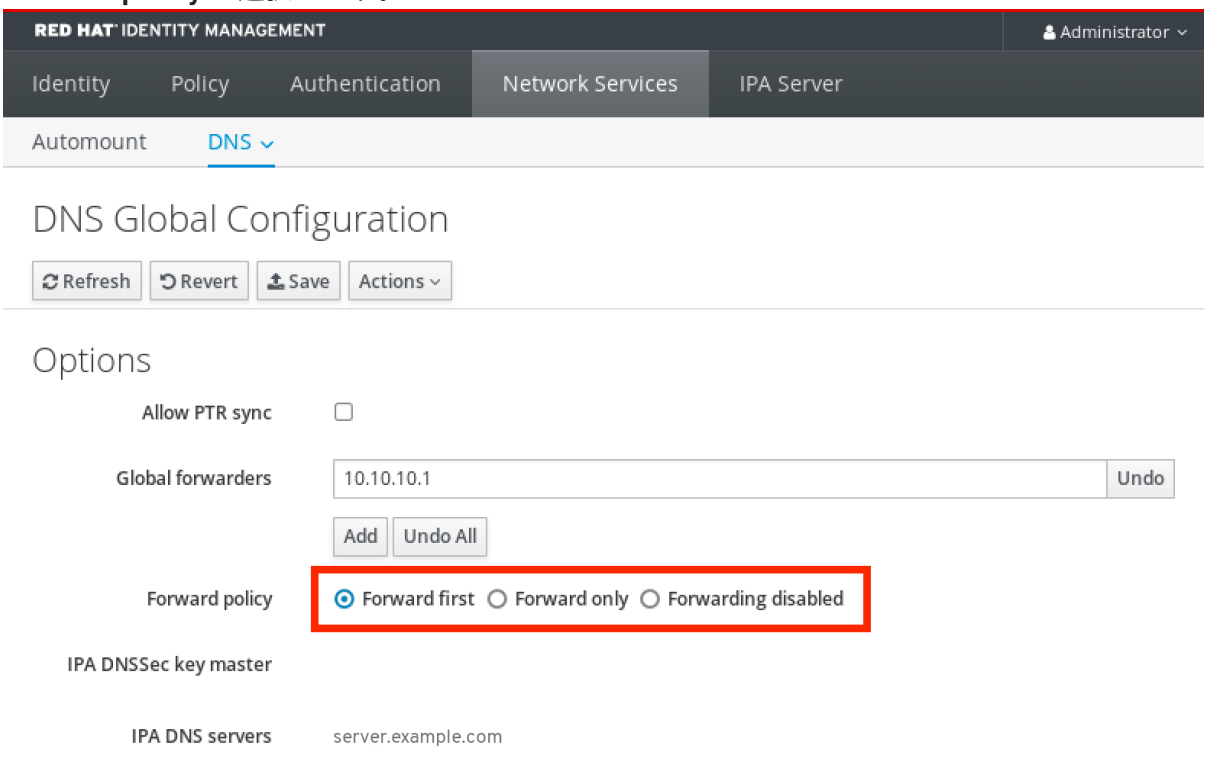
2. **DNS Global Configuration** セクションで、**Add** をクリックします。



3. 転送された DNS クエリーを受信する DNS サーバーの IP アドレスを指定します。



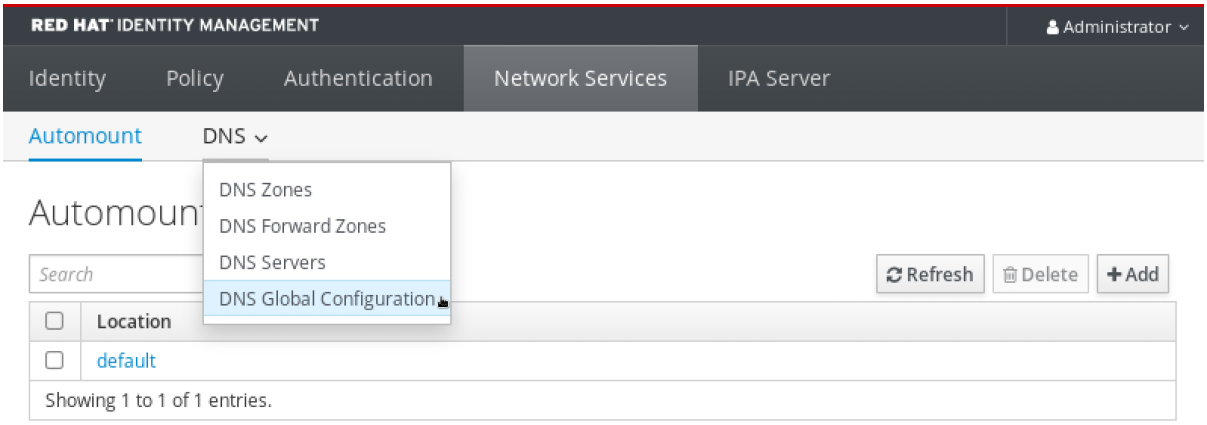
4. **Forward policy** を選択します。



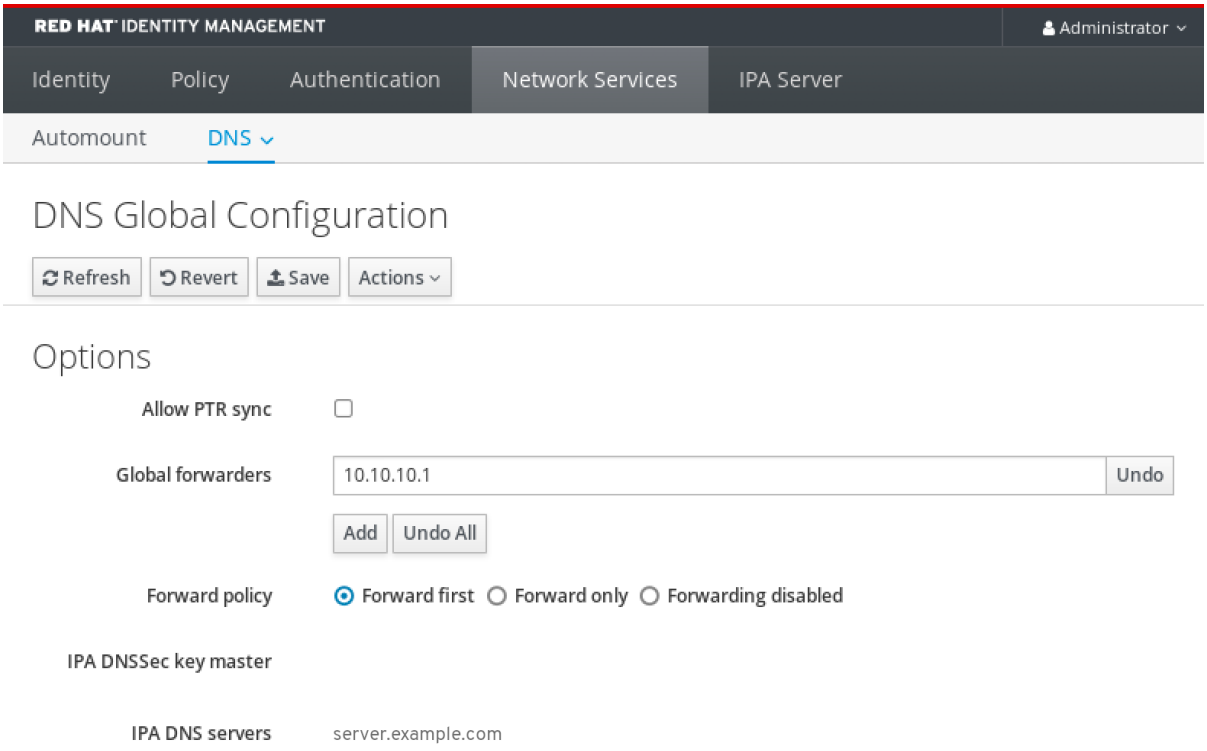
5. ウィンドウの上部にある **Save** をクリックします。

検証手順

1. **Network Services → DNS Global Configuration → DNS** の順に選択します。



2. 指定した転送ポリシーで、グローバルフォワーダーが IdM Web UI で存在し、有効化されていることを確認します。



96.4. CLI でのグローバルフォワーダーの追加

コマンドラインインターフェイス(CLI)を使用してグローバル DNS フォワーダーを追加するには、以下の手順に従います。

前提条件

- IdM 管理者としてログインしている。
- クエリーを転送する DNS サーバーのインターネットプロトコル (IP) アドレスを知っている。

手順

- **ipa dnsconfig-mod** コマンドを使用して、新しいグローバルフォワーダーを追加します。**--forwarder** オプションで DNS フォワーダーの IP アドレスを指定します。

```
[user@server ~]$ ipa dnsconfig-mod --forwarder=10.10.0.1
```

```
Server will check DNS forwarder(s).
This may take some time, please wait ...
  Global forwarders: 10.10.0.1
  IPA DNS servers: server.example.com
```

検証手順

- **dnsconfig-show** コマンドを使用して、グローバルフォワーダーを表示します。

```
[user@server ~]$ ipa dnsconfig-show
  Global forwarders: 10.10.0.1
  IPA DNS servers: server.example.com
```

96.5. IDM WEB UI での DNS 正引きゾーンの追加

以下の手順に従って、Identity Management (IdM) Web UI に DNS 正引きゾーンを追加します。

重要

絶対に必要な場合を除き、正引きゾーンは使用しないでください。正引きゾーンは、標準的な解決策ではないので、正引きゾーンを使用すると予期しない動作が発生する可能性があります。正引きゾーンを使用する必要がある場合は、グローバル転送設定が優先されるように、正引きゾーンの使用を制限します。

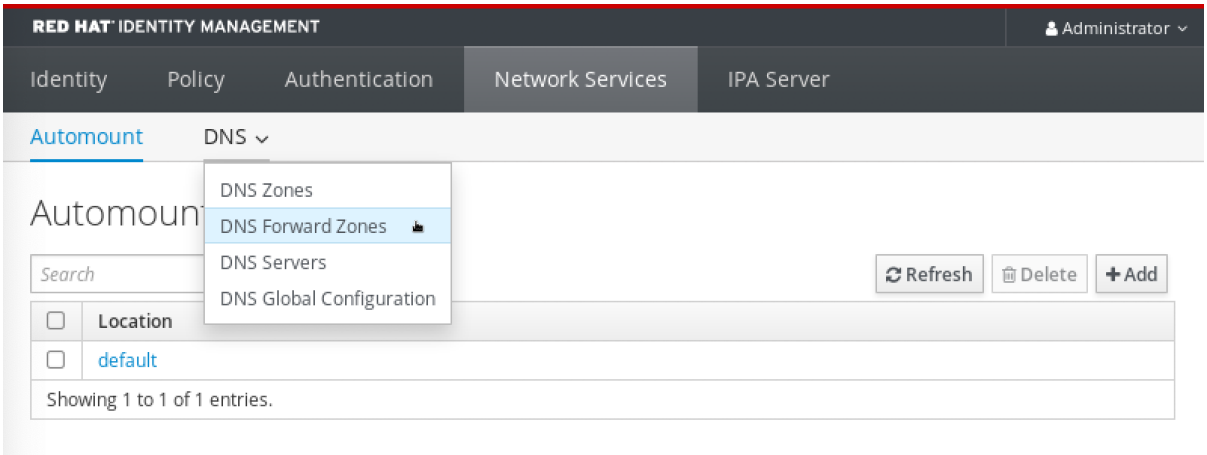
新しい DNS ゾーンを作成する場合には、Red Hat は、ネームサーバー (NS) レコードで標準の DNS 委譲を常に使用し、正引きゾーンを回避することを推奨します。多くの場合、グローバルフォワーダーを使用するだけで十分なため、正引きゾーンは必要ありません。

前提条件

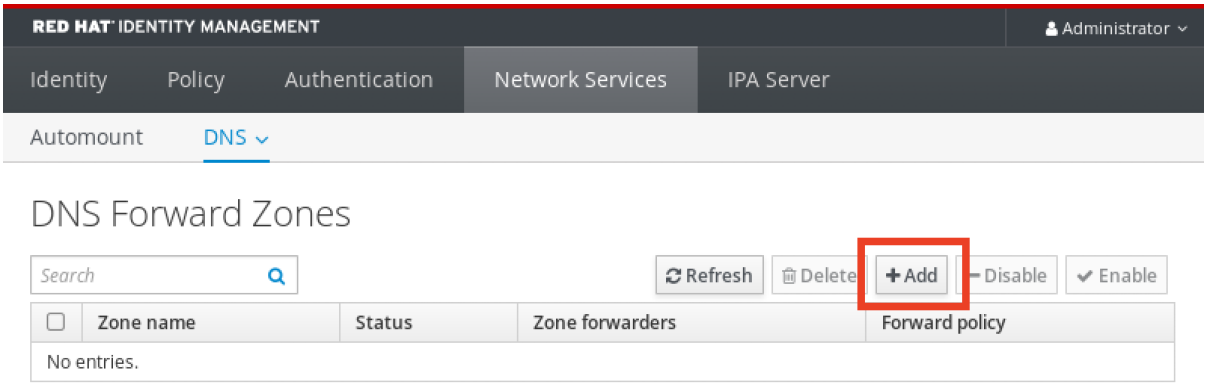
- IdM 管理者として IdM WebUI にログインしている。
- クエリーを転送する DNS サーバーのインターネットプロトコル (IP) アドレスを知っている。

手順

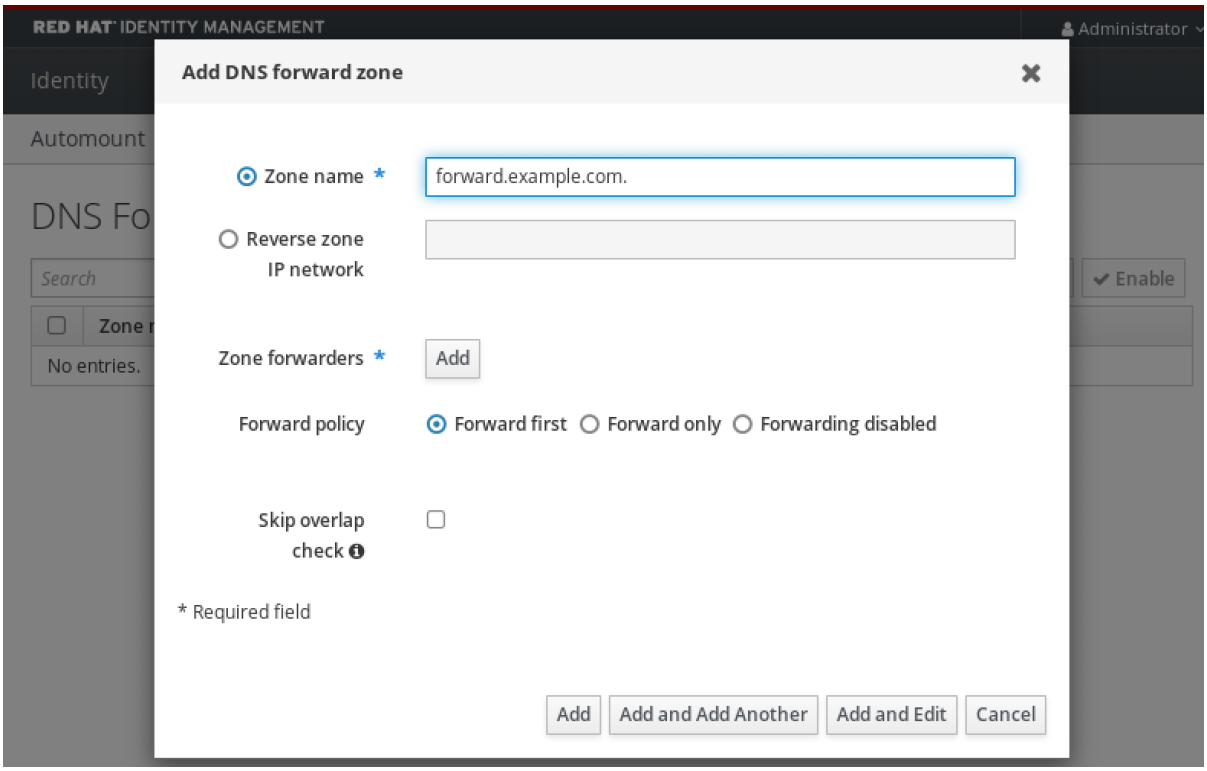
1. IdM Web UI で **Network Services → DNS Forward Zones → DNS** の順に選択します。



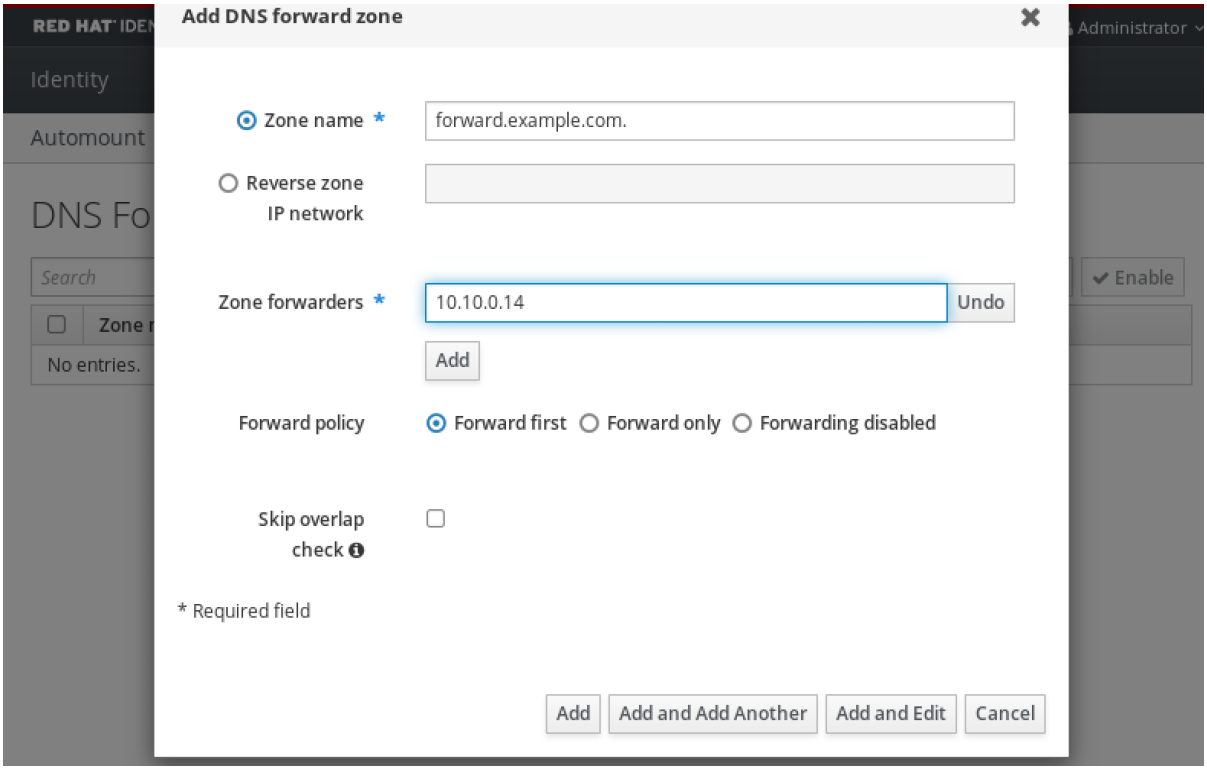
2. **DNS Forward Zones** セクションで、**Add** をクリックします。



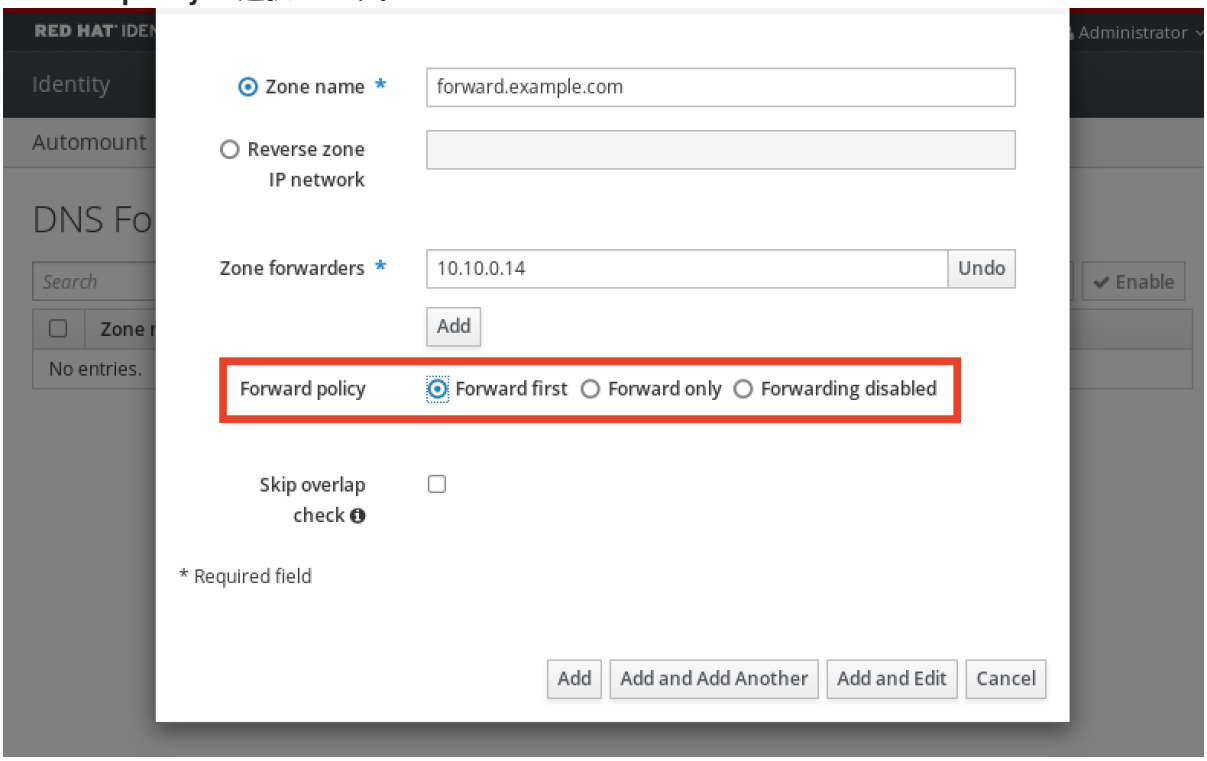
3. **Add DNS forward zone** ウィンドウで、正引きゾーン名を指定します。



4. **Add** ボタンをクリックして、転送要求を受信する DNS サーバーの IP アドレスを指定します。正引きゾーンごとに複数のフォワーダーを指定できます。



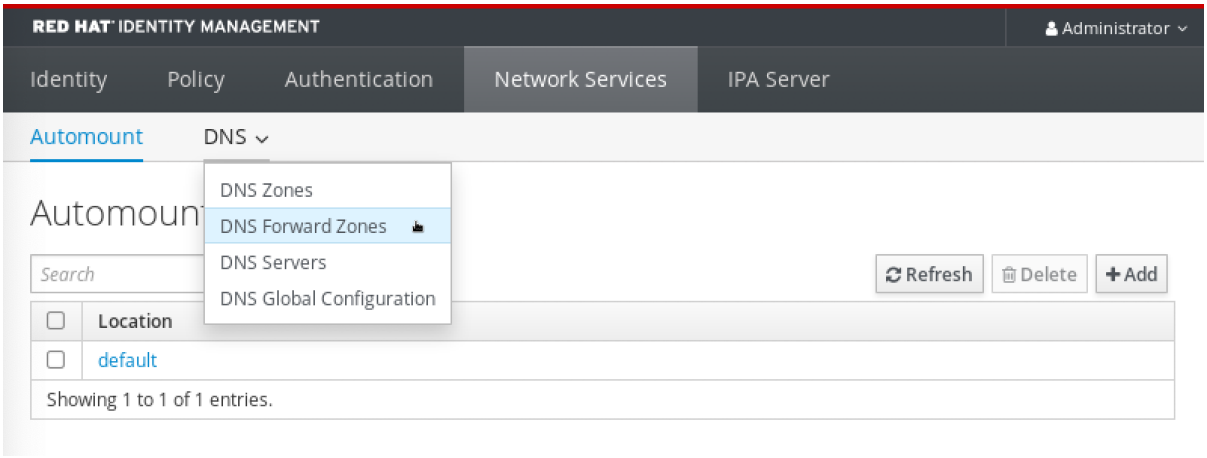
5. **Forward policy** を選択します。



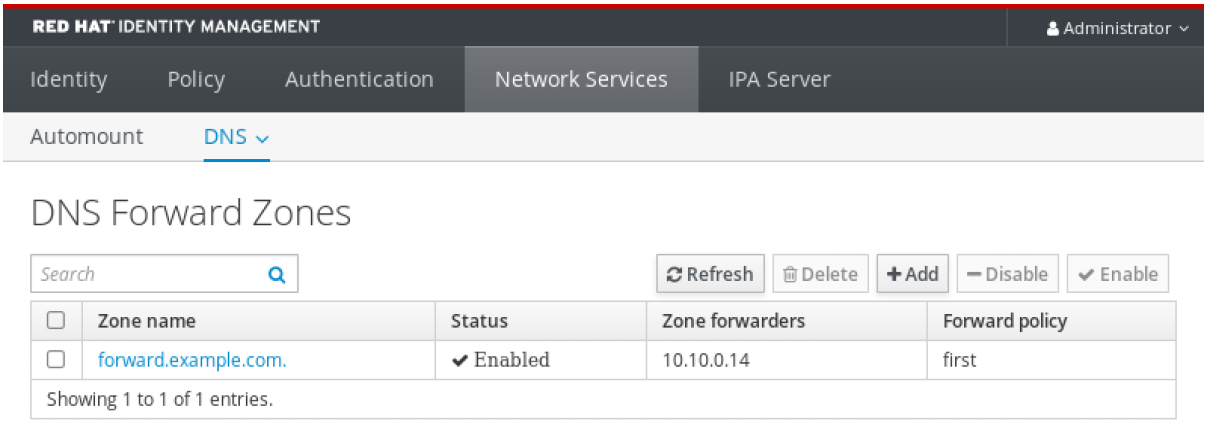
6. ウィンドウの下部にある **Add** をクリックして、新しい正引きゾーンを追加します。

検証手順

1. IdM Web UI で **Network Services** → **DNS Forward Zones** → **DNS** の順に選択します。



2. 指定したフォワーダーおよび転送ポリシーで、正引きゾーンが IdM Web UI で存在し、有効化されていることを確認します。



96.6. CLI での DNS 正引きゾーンの追加

コマンドラインインターフェイス(CLI)を使用して DNS 正引きゾーンを追加するには、以下の手順に従います。

重要

絶対に必要な場合を除き、正引きゾーンは使用しないでください。正引きゾーンは、標準的な解決策ではないので、正引きゾーンを使用すると予期しない動作が発生する可能性があります。正引きゾーンを使用する必要がある場合は、グローバル転送設定が優先されるように、正引きゾーンの使用を制限します。

新しい DNS ゾーンを作成する場合には、Red Hat は、ネームサーバー (NS) レコードで標準の DNS 委譲を常に使用し、正引きゾーンを回避することを推奨します。多くの場合、グローバルフォワーダーを使用するだけで十分なため、正引きゾーンは必要ありません。

前提条件

- IdM 管理者としてログインしている。
- クエリーを転送する DNS サーバーのインターネットプロトコル (IP) アドレスを知っている。

手順

- **dnsforwardzone-add** コマンドを使用して、新しい正引きゾーンを追加します。転送ポリシーが **none** ではない場合には、**--forwarder** オプションを使用して最低でもフォワーダーを 1 つ指定し、**--forward-policy** オプションで転送ポリシーを指定します。

```
[user@server ~]$ ipa dnsforwardzone-add forward.example.com. --
forwarder=10.10.0.14 --forwarder=10.10.1.15 --forward-policy=first

Zone name: forward.example.com.
Zone forwarders: 10.10.0.14, 10.10.1.15
Forward policy: first
```

検証手順

- **dnsforwardzone-show** コマンドを使用して、作成した DNS 正引きゾーンを表示します。

```
[user@server ~]$ ipa dnsforwardzone-show forward.example.com.

Zone name: forward.example.com.
Zone forwarders: 10.10.0.14, 10.10.1.15
Forward policy: first
```

96.7. ANSIBLE を使用した IDM での DNS グローバルフォワーダーの確立

以下の手順に従って、Ansible Playbook を使用して IdM で DNS グローバルフォワーダーを確立します。

以下の手順の例では、IdM 管理者はポート **53** にインターネットプロトコル (IP) v4 アドレスが **8.8.6.6**、IPv6 アドレスが **2001:4860:4860::8800** で指定されている DNS サーバーに DNS グローバルフォワーダーを作成します。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- IdM 管理者パスワードを把握している。

手順

1. **/usr/share/doc/ansible-freeipa/playbooks/dnsconfig** ディレクトリーに移動します。

```
$ cd /usr/share/doc/ansible-freeipa/playbooks/dnsconfig
```

2. インベントリーファイルを開き、設定する IdM サーバーが **[ipaserver]** セクションに記載されていることを確認します。たとえば、Ansible に対して **server.idm.example.com** を設定するように指示するには、次のコマンドを実行します。

```
[ipaserver]
server.idm.example.com
```

3. Ansible Playbook ファイル (**set-configuration.yml**) のコピーを作成します。以下に例を示します。

```
$ cp set-configuration.yml establish-global-forwarder.yml
```

4. **establish-global-forwarder.yml** ファイルを開いて編集します。

5. 以下の変数を設定してファイルを調整します。

   a. Playbook の **name** 変数は、**IdM DNS でグローバルフォワーダーを確立する Playbook** の設定に変更します。

   b. **tasks** セクションで、タスクの **name** を **Create a DNS global forwarder to 8.8.6.6 and 2001:4860:4860::8800** に変更します。

   c. **ipadnsconfig** の **forwarders** セクションで以下を行います。

      i. 最初の **ip_address** の値は、グローバルフォワーダーの IPv4 アドレス (**8.8.6.6**) に変更します。

      ii. 2 番目の **ip_address** の値は、グローバルフォワーダーの IPv6 アドレス (**2001:4860:4860::8800**) に変更します。

      iii. **port** の値が **53** に設定されていることを確認します。

   d. **forward_policy** を **first** に変更します。
   今回の例で使用するように変更した Ansible Playbook ファイル:

```
---
- name: Playbook to establish a global forwarder in IdM DNS
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Create a DNS global forwarder to 8.8.6.6 and 2001:4860:4860::8800
    ipadnsconfig:
      forwarders:
        - ip_address: 8.8.6.6
        - ip_address: 2001:4860:4860::8800
          port: 53
      forward_policy: first
      allow_sync_ptr: yes
```

6. ファイルを保存します。

7. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file establish-global-forwarder.yml
```

関連情報

- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-dnsconfig.md** ファイルを参照してください。

96.8. ANSIBLE を使用して IDM に DNS グローバルフォワーダーを存在させる手順

以下の手順に従って、Ansible Playbook を使用して、IdM に DNS グローバルフォワーダーを追加します。以下の例では、IdM 管理者は、ポート **53** にインターネットプロトコル (IP) v4 アドレスが **7.7.9.9**、IPv6 アドレスが **2001:db8::1:0** で指定されている DNS サーバーに、DNS グローバルフォワーダーが配置されるようにします。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- IdM 管理者パスワードを把握している。

手順

1. **/usr/share/doc/ansible-freeipa/playbooks/dnsconfig** ディレクトリーに移動します。

```
$ cd /usr/share/doc/ansible-freeipa/playbooks/dnsconfig
```

2. インベントリーファイルを開き、設定する IdM サーバーが **[ipaserver]** セクションに記載されていることを確認します。たとえば、Ansible に対して **server.idm.example.com** を設定するように指示するには、次のコマンドを実行します。

```
[ipaserver]
server.idm.example.com
```

3. Ansible Playbook ファイル (**forwarders-absent.yml**) のコピーを作成します。以下に例を示します。

```
$ cp forwarders-absent.yml ensure-presence-of-a-global-forwarder.yml
```

4. **ensure-presence-of-a-global-forwarder.yml** ファイルを開いて編集します。

5. 以下の変数を設定してファイルを調整します。

   a. Playbook の **name** 変数は、**IdM DNS にグローバルフォワーダーを追加する Playbook** の設定に変更します。

   b. **tasks** セクションで、タスクの **name** を **Ensure the presence of a DNS global forwarder to 7.7.9.9 and 2001:db8::1:0 on port 53** に変更します。

   c. **ipadnsconfig** の **forwarders** セクションで以下を行います。

      i. 最初の **ip_address** の値は、グローバルフォワーダーの IPv4 アドレス (**7.7.9.9**) に変更します。

      ii. 2 番目の **ip_address** の値は、グローバルフォワーダーの IPv6 アドレス (**2001:db8::1:0**) に変更します。

      iii. **port** の値が **53** に設定されていることを確認します。

   d. **state** を **present** に変更します。
   今回の例で使用するように変更した Ansible Playbook ファイル:

```
---
- name: Playbook to ensure the presence of a global forwarder in IdM DNS
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure the presence of a DNS global forwarder to 7.7.9.9 and 2001:db8::1:0 on port
53
    ipadnsconfig:
      forwarders:
        - ip_address: 7.7.9.9
        - ip_address: 2001:db8::1:0
          port: 53
      state: present
```

6. ファイルを保存します。

7. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file ensure-presence-
of-a-global-forwarder.yml
```

関連情報

- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-dnsconfig.md** ファイルを参照してください。

96.9. ANSIBLE を使用して IDM に DNS グローバルフォワーダーを存在させないようにする手順

以下の手順に従って、Ansible Playbook を使用して IdM で DNS グローバルフォワーダーを削除しま

す。以下の手順では、IdM 管理者が、ポート **53** で、IP (Internet Protocol) v4 アドレス **8.8.6.6** および IP v6 アドレス **2001:4860:4860::8800** を持つ DNS グローバルフォワーダーが存在しないことを確認します。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- IdM 管理者パスワードを把握している。

手順

1. **/usr/share/doc/ansible-freeipa/playbooks/dnsconfig** ディレクトリーに移動します。

```
$ cd /usr/share/doc/ansible-freeipa/playbooks/dnsconfig
```

2. インベントリーファイルを開き、設定する IdM サーバーが **[ipaserver]** セクションに記載されていることを確認します。たとえば、Ansible に対して **server.idm.example.com** を設定するように指示するには、次のコマンドを実行します。

```
[ipaserver]
server.idm.example.com
```

3. Ansible Playbook ファイル (**forwarders-absent.yml**) のコピーを作成します。以下に例を示します。

```
$ cp forwarders-absent.yml ensure-absence-of-a-global-forwarder.yml
```

4. **ensure-absence-of-a-global-forwarder.yml** ファイルを開いて編集します。
5. 以下の変数を設定してファイルを調整します。
   a. Playbook の **name** 変数は、**IdM DNS でグローバルフォワーダーを配置しない Playbook** の設定に変更します。
   b. **tasks** セクションで、タスクの **name** を **Ensure the absence of a DNS global forwarder to 8.8.6.6 and 2001:4860:4860::8800 on port 53** に変更します。
   c. **ipadnsconfig** の **forwarders** セクションで以下を行います。
      i. 最初の **ip_address** の値は、グローバルフォワーダーの IPv4 アドレス (**8.8.6.6**) に変更します。

ii. 2 番目の **ip_address** の値は、グローバルフォワーダーの IPv6 アドレス (**2001:4860:4860::8800**) に変更します。

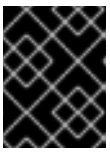
iii. **port** の値が **53** に設定されていることを確認します。

d. **action** 変数は **member** に設定します。

e. **state** が **absent** に設定されていることを確認します。

今回の例で使用するように変更した Ansible Playbook ファイル:

```
---
- name: Playbook to ensure the absence of a global forwarder in IdM DNS
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure the absence of a DNS global forwarder to 8.8.6.6 and
2001:4860:4860::8800 on port 53
    ipadnsconfig:
      forwarders:
        - ip_address: 8.8.6.6
        - ip_address: 2001:4860:4860::8800
          port: 53
      action: member
      state: absent
```



重要

Playbook で **action: member** を使用せずに **state: absent** オプションだけを使用すると、その Playbook は失敗します。

6. ファイルを保存します。

7. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file ensure-absence-of-a-global-forwarder.yml
```

関連情報

- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-dnsconfig.md** ファイル
- ipadnsconfig ansible-freeipa モジュールの **action: member** オプション

96.10. ANSIBLE を使用した IDM での DNS グローバルフォワーダーの無効化

Ansible Playbook を使用して、IdM で DNS グローバルフォワーダーを無効にするには、以下の手順に従います。以下の手順の例では、IdM の管理者がグローバルフォワーダーの転送ポリシーが **none** に設定されていることを確認し、グローバルフォワーダーを実質的に無効にします。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- IdM 管理者パスワードを把握している。

手順

1. **/usr/share/doc/ansible-freeipa/playbooks/dnsconfig** ディレクトリーに移動します。

```
$ cd /usr/share/doc/ansible-freeipa/playbooks/dnsconfig
```

2. インベントリーファイルを開き、設定する IdM サーバーが **[ipaserver]** セクションに記載されていることを確認します。たとえば、Ansible に対して **server.idm.example.com** を設定するように指示するには、次のコマンドを実行します。

```
[ipaserver]
server.idm.example.com
```

3. 全 DNS グローバルフォワーダーを無効にするように設定済みの Ansible Playbook ファイル (**disable-global-forwarders.yml**) の内容を確認します。以下に例を示します。

```
$ cat disable-global-forwarders.yml
---
- name: Playbook to disable global DNS forwarders
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Disable global forwarders.
    ipadnsconfig:
      forward_policy: none
```

4. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file disable-global-forwarders.yml
```

関連情報

- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-dnsconfig.md** ファイルを参照してください。

96.11. ANSIBLE を使用して IDM に DNS 正引きゾーンを存在させる手順

以下の手順に従って、Ansible Playbook を使用して IdM に DNS 正引きゾーンを追加します。以下の手順の例では、IdM 管理者は、インターネットプロトコル (IP) プロトコルが **8.8.8.8** の DNS サーバーに **example.com** の DNS 正引きゾーンが配置されるようにします。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- IdM 管理者パスワードを把握している。

手順

1. **/usr/share/doc/ansible-freeipa/playbooks/dnsconfig** ディレクトリーに移動します。

```
$ cd /usr/share/doc/ansible-freeipa/playbooks/dnsconfig
```

2. インベントリーファイルを開き、設定する IdM サーバーが **[ipaserver]** セクションに記載されていることを確認します。たとえば、Ansible に対して **server.idm.example.com** を設定するように指示するには、次のコマンドを実行します。

```
[ipaserver]
server.idm.example.com
```

3. Ansible Playbook ファイル (**forwarders-absent.yml**) のコピーを作成します。以下に例を示します。

```
$ cp forwarders-absent.yml ensure-presence-forwardzone.yml
```

4. **ensure-presence-forwardzone.yml** ファイルを開いて編集します。
5. 以下の変数を設定してファイルを調整します。
   a. Playbook の **name** 変数は、**IdM DNS に DNS 正引きゾーンを追加する Playbook** の設定に変更します。
   b. **tasks** セクションで、タスクの **name** を **Ensure presence of a dnsforwardzone for example.com to 8.8.8.8** に変更します。

c. **tasks** セクションで、**ipadnsconfig** のヘディングを **ipadnsforwardzone** に変更します。

d. **ipadnsforwardzone** セクションで以下を実行します。

i. **ipaadmin_password** 変数を追加して、IdM 管理者パスワードに設定します。

ii. **name** 変数を追加して **example.com** に設定します。

iii. **forwarders** セクションで、以下を実行します。

A. **ip_address** と **port** の行を削除します。

B. 転送要求を受信できるように DNS サーバーの IP アドレスをダッシュの後に指定して追加します。

```
- 8.8.8.8
```

iv. **forwardpolicy** 変数を追加して **first** に設定します。

v. **skip_overlap_check** 変数を追加し、**true** に設定します。

vi. **state** 変数は **present** に変更します。

今回の例で使用するように変更した Ansible Playbook ファイル:

```
---
- name: Playbook to ensure the presence of a dnsforwardzone in IdM DNS
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure the presence of a dnsforwardzone for example.com to 8.8.8.8
  ipadnsforwardzone:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: example.com
      forwarders:
          - 8.8.8.8
      forwardpolicy: first
      skip_overlap_check: true
      state: present
```

6. ファイルを保存します。

7. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file ensure-presence-forwardzone.yml
```

関連情報

- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-dnsforwardzone.md** ファイルを参照してください。

96.12. ANSIBLE を使用して IDM で DNS 正引きゾーンを複数配置する手順

以下の手順に従って、Ansible Playbook を使用して、IdM の DNS 正引きゾーンに複数のフォワーダーがあることを確認します。以下の手順の例では、IdM 管理者が **example.com** の DNS 正引きゾーンが **8.8.8.8** と **4.4.4.4** に転送されるようにします。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- IdM 管理者パスワードを把握している。

手順

1. **/usr/share/doc/ansible-freeipa/playbooks/dnsconfig** ディレクトリーに移動します。

```
$ cd /usr/share/doc/ansible-freeipa/playbooks/dnsconfig
```

2. インベントリーファイルを開き、設定する IdM サーバーが **[ipaserver]** セクションに記載されていることを確認します。たとえば、Ansible に対して **server.idm.example.com** を設定するように指示するには、次のコマンドを実行します。

```
[ipaserver]
server.idm.example.com
```

3. Ansible Playbook ファイル (**forwarders-absent.yml**) のコピーを作成します。以下に例を示します。

```
$ cp forwarders-absent.yml ensure-presence-multiple-forwarders.yml
```

4. **ensure-presence-multiple-forwarders.yml** ファイルを開いて編集します。
5. 以下の変数を設定してファイルを調整します。
   a. Playbook の **name** 変数は、**IdM DNS の DNS 正引きゾーンに複数のフォワーダーを配置する Playbook** の設定に変更します。
   b. **tasks** セクションで、タスクの **name** を **Ensure presence of 8.8.8.8 and 4.4.4.4 forwarders in dnsforwardzone for example.com** に変更します。
   c. **tasks** セクションで、**ipadnsconfig** のヘディングを **ipadnsforwardzone** に変更します。
   d. **ipadnsforwardzone** セクションで以下を実行します。
      i. **ipaadmin_password** 変数を追加して、IdM 管理者パスワードに設定します。

ii. **name** 変数を追加して **example.com** に設定します。

iii. **forwarders** セクションで、以下を実行します。

A. **ip_address** と **port** の行を削除します。

B. 配置する DNS サーバーの IP アドレスを、前にダッシュをつけて追加します。

```
- 8.8.8.8
- 4.4.4.4
```

iv. state 変数を present に変更します。

今回の例で使用するように変更した Ansible Playbook ファイル:

```
---
- name: name: Playbook to ensure the presence of multiple forwarders in a dnsforwardzone
in IdM DNS
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure presence of 8.8.8.8 and 4.4.4.4 forwarders in dnsforwardzone for
example.com
  ipadnsforwardzone:
      ipaadmin_password: "{{ ipaadmin_password }}"
     name: example.com
      forwarders:
          - 8.8.8.8
          - 4.4.4.4
      state: present
```

6. ファイルを保存します。

7. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file ensure-presence-
multiple-forwarders.yml
```

関連情報

- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-dnsforwardzone.md** ファイルを参照してください。

96.13. ANSIBLE を使用して IDM で DNS 正引きゾーンを無効にする手順

Ansible Playbook を使用して IdM で DNS 正引きゾーンを無効にするには、以下の手順に従います。以下の手順の例では、IdM 管理者は **example.com** の DNS 正引きゾーンが無効になっていることを確認します。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。

- Ansible バージョン 2.14 以降を使用している。
- Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
- この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
- この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。

- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- IdM 管理者パスワードを把握している。

手順

1. **/usr/share/doc/ansible-freeipa/playbooks/dnsconfig** ディレクトリーに移動します。

```
$ cd /usr/share/doc/ansible-freeipa/playbooks/dnsconfig
```

2. インベントリーファイルを開き、設定する IdM サーバーが **[ipaserver]** セクションに記載されていることを確認します。たとえば、Ansible に対して **server.idm.example.com** を設定するように指示するには、次のコマンドを実行します。

```
[ipaserver]
server.idm.example.com
```

3. Ansible Playbook ファイル (**forwarders-absent.yml**) のコピーを作成します。以下に例を示します。

```
$ cp forwarders-absent.yml ensure-disabled-forwardzone.yml
```

4. **ensure-disabled-forwardzone.yml** ファイルを開いて編集します。
5. 以下の変数を設定してファイルを調整します。
   a. Playbook の **name** 変数は、**IdM DNS に DNS 正引きゾーンを無効にする Playbook** の設定に変更します。
   b. **tasks** セクションで、タスクの **name** を **Ensure a dnsforwardzone for example.com is disabled** に変更します。
   c. **tasks** セクションで、**ipadnsconfig** のヘディングを **ipadnsforwardzone** に変更します。
   d. **ipadnsforwardzone** セクションで以下を実行します。
      i. **ipaadmin_password** 変数を追加して、IdM 管理者パスワードに設定します。
      ii. **name** 変数を追加して **example.com** に設定します。
      iii. **forwarders** セクション全体を削除します。
      iv. **state** 変数を **disabled** に変更します。

   今回の例で使用するように変更した Ansible Playbook ファイル:

```
---
- name: Playbook to ensure a dnsforwardzone is disabled in IdM DNS
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure a dnsforwardzone for example.com is disabled
  ipadnsforwardzone:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: example.com
      state: disabled
```

6. ファイルを保存します。

7. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file ensure-disabled-forwardzone.yml
```

関連情報

- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-dnsforwardzone.md** ファイルを参照してください。

96.14. ANSIBLE を使用して IDM から DNS 正引きゾーンを削除する手順

Ansible Playbook を使用して IdM に DNS 正引きゾーンを削除するには、以下の手順に従います。以下の例では、IdM 管理者は **example.com** の DNS 正引きゾーンを削除します。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- IdM 管理者パスワードを把握している。

手順

1. **/usr/share/doc/ansible-freeipa/playbooks/dnsconfig** ディレクトリーに移動します。

```
$ cd /usr/share/doc/ansible-freeipa/playbooks/dnsconfig
```

2. インベントリーファイルを開き、設定する IdM サーバーが **[ipaserver]** セクションに記載されていることを確認します。たとえば、Ansible に対して **server.idm.example.com** を設定するように指示するには、次のコマンドを実行します。

```
[ipaserver]
server.idm.example.com
```

3. Ansible Playbook ファイル (**forwarders-absent.yml**) のコピーを作成します。以下に例を示します。

```
$ cp forwarders-absent.yml ensure-absence-forwardzone.yml
```

4. **ensure-absence-forwardzone.yml** ファイルを開いて編集します。

5. 以下の変数を設定してファイルを調整します。

   a. Playbook の **name** 変数は、**IdM DNS に DNS 正引きゾーンを削除する Playbook** の設定に変更します。

   b. **tasks** セクションで、タスクの **name** を **Ensure the absence of a dnsforwardzone for example.com** に変更します。

   c. **tasks** セクションで、**ipadnsconfig** のヘディングを **ipadnsforwardzone** に変更します。

   d. **ipadnsforwardzone** セクションで以下を実行します。

      i. **ipaadmin_password** 変数を追加して、IdM 管理者パスワードに設定します。

      ii. **name** 変数を追加して **example.com** に設定します。

      iii. **forwarders** セクション全体を削除します。

      iv. **state** 変数を **absent** のままにします。

   今回の例で使用するように変更した Ansible Playbook ファイル:

```
---
- name: Playbook to ensure the absence of a dnsforwardzone in IdM DNS
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure the absence of a dnsforwardzone for example.com
    ipadnsforwardzone:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: example.com
      state: absent
```

6. ファイルを保存します。

7. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file ensure-absence-forwardzone.yml
```

関連情報

- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-dnsforwardzone.md** ファイルを参照してください。

第97章 IDM での DNS レコードの管理

本章では、Identity Management (IdM) で DNS レコードを管理する方法を説明します。IdM 管理者は、IdM で DNS レコードを追加、変更、および削除できます。本章は以下のセクションで設定されます。

- IdM の DNS レコード
- IdM Web UI からの DNS リソースレコードの追加
- IdM CLI からの DNS リソースレコードの追加
- 一般的な ipa dnsrecord-add オプション
- IdM Web UI での DNS レコードの削除
- IdM Web UI での DNS レコード全体の削除
- IdM CLI での DNS レコードの削除

前提条件

- IdM デプロイメントに統合 DNS サーバーが含まれている。統合 DNS のある IdM のインストール方法は、以下のリンクのいずれかを参照してください。
  - IdM サーバーのインストール: 統合 DNS と統合 CA を root CA として使用する場合
  - IdM サーバーのインストール: 統合 DNS と外部 CA を root CA として使用する場合

97.1. IDM の DNS レコード

Identity Management (IdM) は、多種の DNS レコードに対応します。以下の 4 つが最も頻繁に使用されます。

A

これは、ホスト名および IPv4 アドレスの基本マップです。A レコードのレコード名は、**www** などのホスト名です。A レコードの **IP アドレス** 値は、**192.0.2.1** などの IPv4 アドレスです。
A レコードの詳細は、RFC 1035 を参照してください。

AAAA

これは、ホスト名および IPv6 アドレスの基本マップです。AAAA レコードのレコード名は **www** などのホスト名です。**IP アドレス** の値は、**2001:DB8::1111** などの IPv6 アドレスです。
AAAA レコードの詳細は RFC 3596 を参照してください。

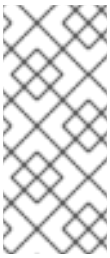
SRV

サービス (SRV) リソースレコード は、特定のサービスを提供するサーバーの DNS 名にサービス名をマッピングします。たとえば、このタイプのレコードは LDAP ディレクトリーのようなサービスを管理するサーバーに、このサービスをマッピングします。
SRV レコードのレコード名は、**_ldap._tcp** など、**_service._protocol** の形式を取ります。SRV レコードの設定オプションには、ターゲットサービスの優先順位、加重、ポート番号、およびホスト名が含まれます。

SRV レコードの詳細は、RFC 2782 を参照してください。

PTR

ポインターレコード (PTR) は、IP アドレスをドメイン名にマッピングする逆引き DNS レコードを追加します。



注記

IPv4 アドレスの逆引き DNS ルックアップはすべて、**in-addr.arpa.** ドメインで定義される逆引きエントリーを使用します。人間が判別可能な形式の逆アドレスは、通常の IP とまったく逆で、**in-addr.arpa.** ドメインが最後に付いています。たとえば、ネットワークアドレス **192.0.2.0/24** の逆引きゾーンは、**2.0.192.in-addr.arpa** になります。

PTR レコード名は、RFC 1035 (RFC 2317 および RFC 3596 で拡張) で指定の標準形式を仕様する必要があります。ホスト名の値は、レコードを作成するホストの正規のホスト名である必要があります。

注記

また、IPv6 アドレスの逆引きゾーンは、**.ip6.arpa.** ドメインのゾーンを使用して設定できます。IPv6 逆引きゾーンの詳細は、RFC 3596 を参照してください。

DNS リソースレコードの追加時には、レコードの多くで異なるデータが必要になることに注意してください。たとえば、CNAME レコードにはホスト名が必要ですが、A レコードには IP アドレスが必要です。IdM Web UI では、新しいレコードを追加するフォームのフィールドが自動的に更新され、現在選択されているレコードタイプに必要なデータが反映されます。

97.2. IDM WEB UI での DNS リソースレコードの追加

Identity Management (IdM) Web UI に DNS リソースレコードを追加するには、次の手順に従います。

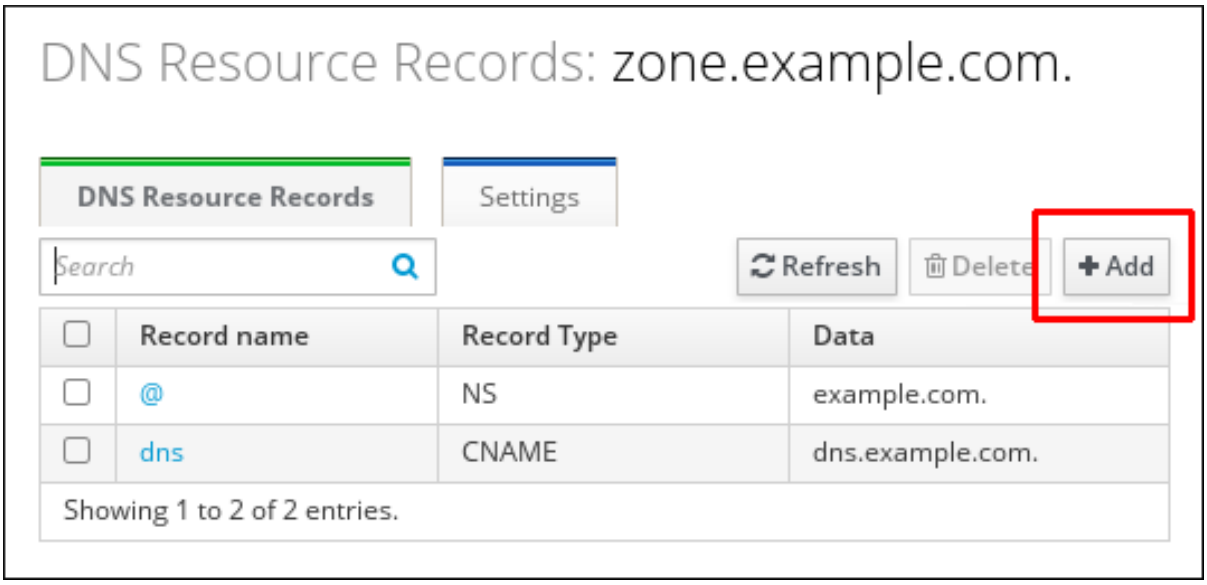
前提条件

- DNS レコードの追加先の DNS ゾーンが存在し、IdM が管理している。IdM DNS での DNS ゾーンの作成に関する詳細は、IdM の DNS ゾーンの管理 を参照してください。
- IdM 管理者としてログインしている。

手順

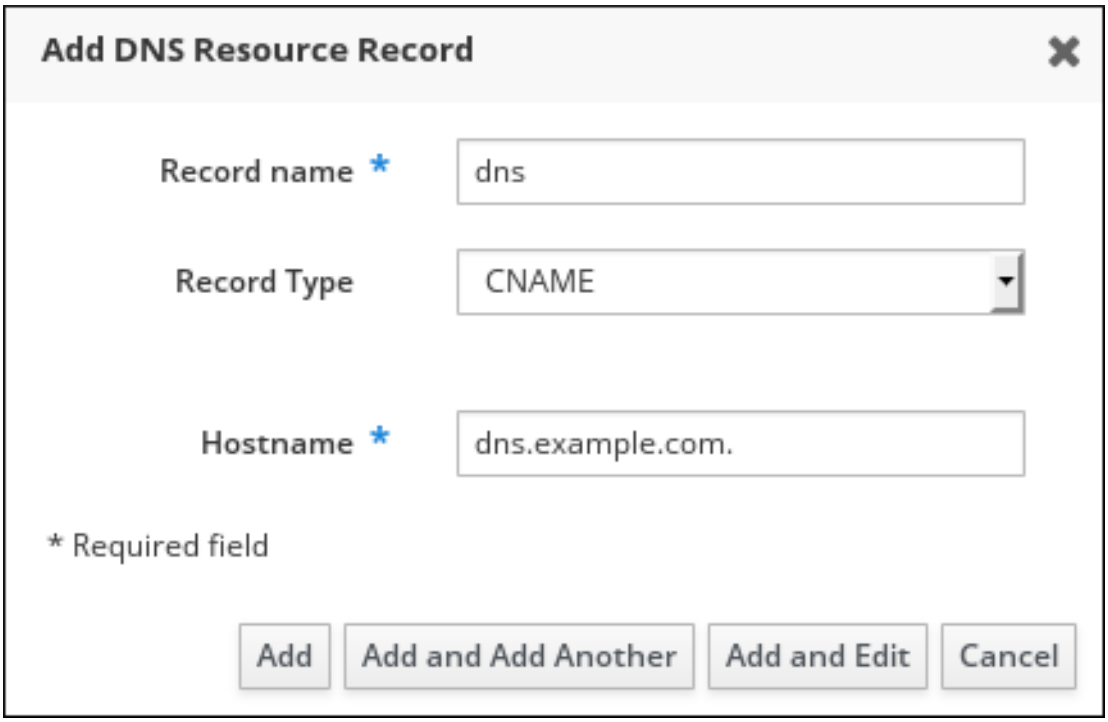
1. IdM Web UI で、**Network Services → DNS → DNS Zones** の順にクリックします。
2. DNS レコードを追加する DNS ゾーンをクリックします。
3. **DNS Resource Record** セクションで、**Add** をクリックして新規レコードを追加します。

図97.1 新しい DNS リソースレコードの追加



4. 作成するレコードのタイプを選択し、必要に応じて他のフィールドにも入力します。

図97.2 新しい DNS リソースレコードの定義



5. **Add** をクリックして、新規レコードを確定します。

97.3. IDM CLI からの DNS リソースレコードの追加

コマンドラインインターフェイス (CLI) から任意のタイプの DNS リソースレコードを追加するには、次の手順に従います。

前提条件

- DNS レコードを追加する DNS ゾーンが存在する。IdM DNS での DNS ゾーンの作成に関する詳細は、IdM の DNS ゾーンの管理 を参照してください。

- IdM 管理者としてログインしている。

手順

1. DNS リソースレコードを追加するには、**ipa dnsrecord-add** コマンドを使用します。このコマンドは、以下の構文に従います。

```
$ ipa dnsrecord-add zone_name record_name --record_type_option=data
```

上記のコマンドでは、以下のようになります。

- **zone_name** は、レコードを追加する DNS ゾーンの名前です。
- **record_name** は、新しい DNS リソースレコードの識別子です。

たとえば、**host1** の A タイプ DNS レコードを **idm.example.com** ゾーンに追加するには、次のコマンドを実行します。

```
$ ipa dnsrecord-add idm.example.com host1 --a-rec=192.168.122.123
```

97.4. 一般的な IPA DNSRECORD-* オプション

Identity Management (IdM) で最も一般的な DNS リソースレコードタイプを追加、変更、および削除する場合は、以下のオプションを使用できます。

- A (IPv4)
- AAAA (IPv6)
- SRV
- PTR

Bash では、**--option={val1,val2,val3}** など、波括弧の中にコンマ区切りで値を指定して、複数のエントリーを定義できます。

表97.1 全般的なレコードのオプション

オプション	説明
--ttl=number	レコードの有効期間を設定します。
--structured	raw DNS レコードを解析し、それらを構造化された形式で返します。

表97.2 "A" レコードのオプション

オプション	説明	例
--a-rec=ARECORD	A レコードを 1 つまたはリストで指定します。	**ipa dnsrecord-add idm.example.com host1 --a-rec=192.168.122.123**

オプション	説明	例
	指定の IP アドレスでワイルドカード A レコードを作成できます。	**ipa dnsrecord-add idm.example.com "*" --a-rec=192.168.122.123** [a]
--a-ip-address=string	レコードの IP アドレスを渡します。レコードの作成時に、**A** レコードの値を指定するオプションは **--a-rec** です。ただし **A** レコードを変更する時に、**--a-rec** オプションを使用して **A** レコードの現在の値を指定します。新しい値は、**--a-ip-address** オプションで設定します。	**ipa dnsrecord-mod idm.example.com --a-rec 192.168.122.123 --a-ip-address 192.168.122.124**

[a] この例では、IP アドレスが 192.0.2.123 のワイルドカード **A** レコードを作成します。

表97.3 "AAAA" レコードのオプション

オプション	説明	例
--aaaa-rec=AAAARECORD	AAAA (IPv6) レコードを 1 つまたはリストで指定します。	**ipa dnsrecord-add idm.example.com www --aaaa-rec 2001:db8::1231:5675**
--aaaa-ip-address=string	レコードの IPv6 アドレスを渡します。レコードの作成時に、**A** レコードの値を指定するオプションは **--aaaa-rec** です。ただし、**A** レコードを変更する時に、**--aaaa-rec** オプションを使用して **A** レコードの現在の値を指定します。新しい値は、**--a-ip-address** オプションで設定します。	**ipa dnsrecord-mod idm.example.com --aaaa-rec 2001:db8::1231:5675 --aaaa-ip-address 2001:db8::1231:5676**

表97.4 "PTR" レコードのオプション

オプション	説明	例
--ptr-rec=PTRRECORD	PTR レコードを 1 つまたはリストで指定します。逆引き DNS レコードを追加する時には、他の DNS レコードの追加の方法と比べ、**ipa dnsrecord-add** コマンドで使用するゾーン名は、逆になります。通常、ホストの IP アドレスは、指定のネットワークにおける IP アドレスの最後のオクテットを使用します。右側の最初の例では、IPv4 アドレスが 192.168.122.4. の server4.idm.example.com の PTR レコードを追加します。2 番目の例では、0.0.0.0.0.0.0.0.8.b.d.0.1.0.0.2.ip6.arpa. に逆引き DNS エントリーを追加します。IP アドレスが 2001:DB8::1111 の server2.example.com ホストの IPv6 逆引きゾーン。	**ipa dnsrecord-add 122.168.192.in-addr.arpa 4 --ptr-rec server4.idm.example.com.** **$ ipa dnsrecord-add 0.0.0.0.0.0.0.0.8.b.d.0.1.0.0.2.ip6.arpa.1.1.1.0.0.0.0.0.0.0.0.0.0.0.0 --ptr-rec server2.idm.example.com.**

オプション	説明	例
--ptr-hostname=string	レコードのホスト名を指定します。	

表97.5 "SRV" レコードのオプション

オプション	説明	例
--srv-rec=SRVRECORD	SRV レコードを 1 つまたはリストで指定します。右側の例では、_ldap._tcp は、SRV レコードのサービスタイプと接続プロトコルを定義します。**--srv-rec** オプションは、優先順位、加重、ポート、およびターゲットの値を定義します。この例では、加重値 51 と 49 が最大 100 まで加算され、特定のレコードが使用される確率を % で表します。	**# ipa dnsrecord-add idm.example.com _ldap._tcp --srv-rec="0 51 389 server1.idm.example.com."** **# ipa dnsrecord-add server.idm.example.com _ldap._tcp --srv-rec="1 49 389 server2.idm.example.com."**
--srv-priority=number	レコードの優先順位を設定します。あるサービスタイプに複数の SRV レコードがある場合もあります。優先順位 (0 - 65535) はレコードの階級を設定し、数字が小さいほど優先順位が高くなります。サービスは、優先順位の最も高いレコードを最初に使用する必要があります。	**# ipa dnsrecord-mod server.idm.example.com _ldap._tcp --srv-rec="1 49 389 server2.idm.example.com." --srv-priority=0**
--srv-weight=number	レコードの加重を設定します。これは、SRV レコードの優先順位が同じ場合に順序を判断する際に役立ちます。設定された加重は最大 100 とし、これは特定のレコードが使用される可能性をパーセンテージで示しています。	**# ipa dnsrecord-mod server.idm.example.com _ldap._tcp --srv-rec="0 49 389 server2.idm.example.com." --srv-weight=60**
--srv-port=number	ターゲットホスト上のサービスのポートを渡します。	**# ipa dnsrecord-mod server.idm.example.com _ldap._tcp --srv-rec="0 60 389 server2.idm.example.com." --srv-port=636**
--srv-target=string	ターゲットホストのドメイン名を提供します。該当サービスがドメイン内で利用可能でない場合は、単一のピリオド (.) として指定される場合があります。	

関連情報

- **ipa dnsrecord-add --help** を実行します。

97.5. IDM WEB UI での DNS レコードの削除

IdM Web UI を使用して Identity Management (IdM) の DNS レコードを削除するには、この手順に従います。

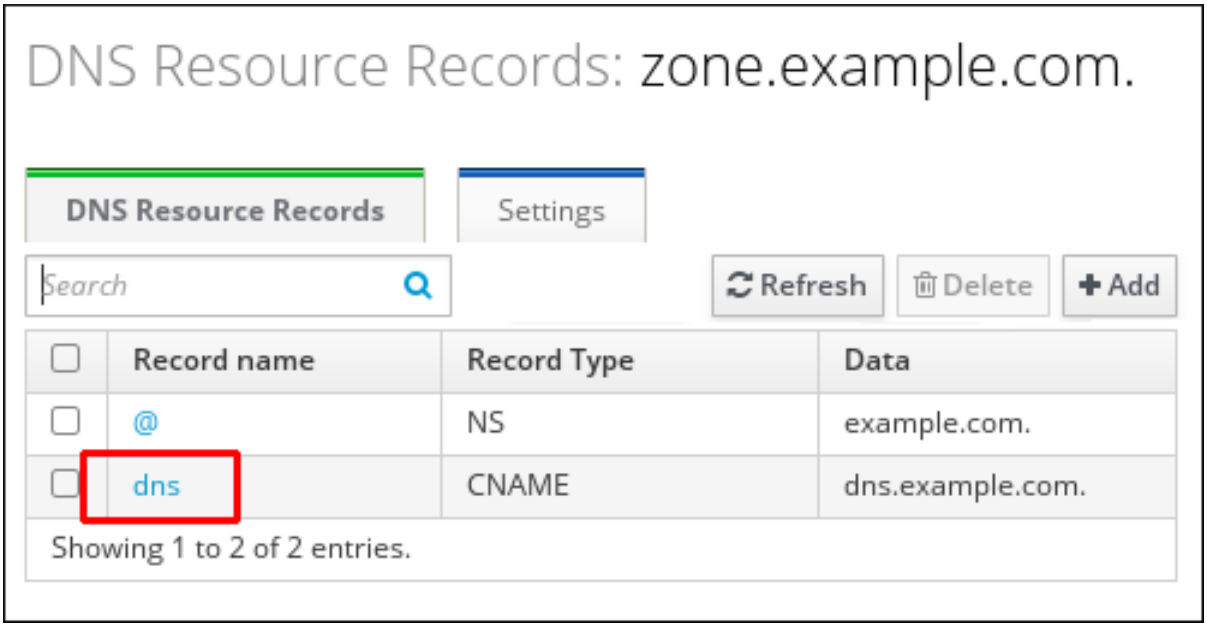
前提条件

- IdM 管理者としてログインしている。

手順

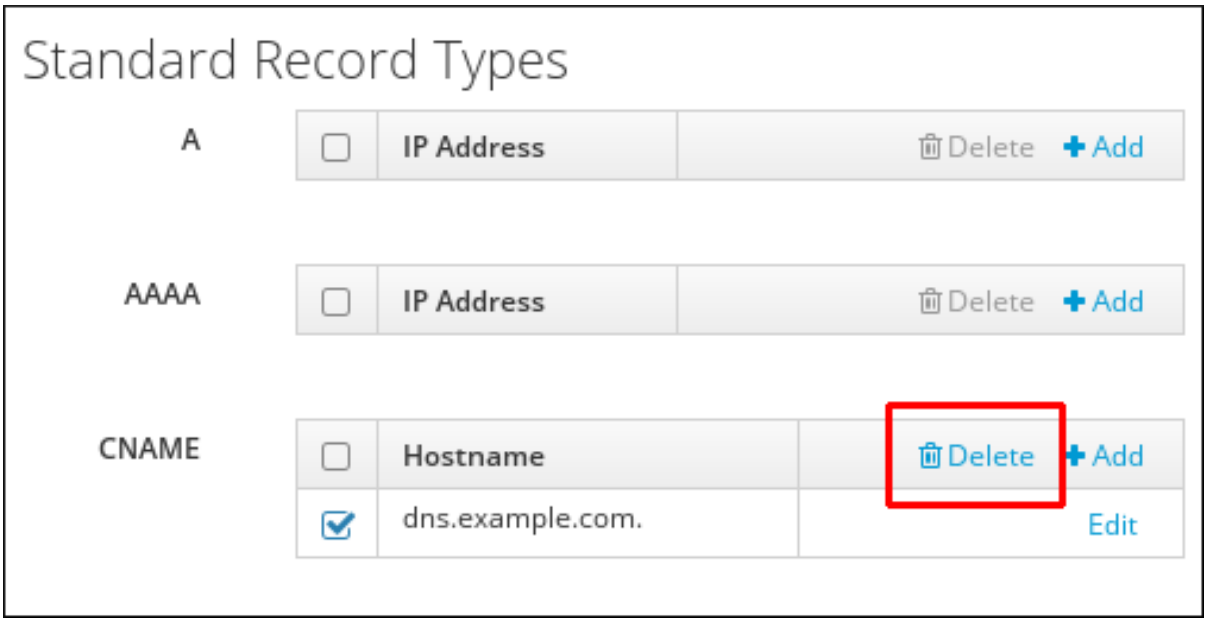
1. IdM Web UI で、**Network Services → DNS → DNS Zones** の順にクリックします。
2. DNS レコードを削除するゾーン (**example.com** など) をクリックします。
3. **DNS Resource Record** のセクションで、リソースレコードの名前をクリックします。

   図97.3 DNS リソースレコードの選択



4. 削除するレコードタイプの名前の横にあるチェックボックスを選択します。
5. **Delete** をクリックします。

図97.4 DNS リソースレコードの削除



選択したレコードタイプが削除されました。リソースレコードの他の設定はそのままになります。

関連情報

- IdM Web UI での DNS レコード全体の削除 を参照してください。

97.6. IDM WEB UI での DNS レコード全体の削除

Identity Management (IdM) Web UI を使用してゾーン内の特定のリソースのすべてのレコードを削除するには、次の手順に従います。

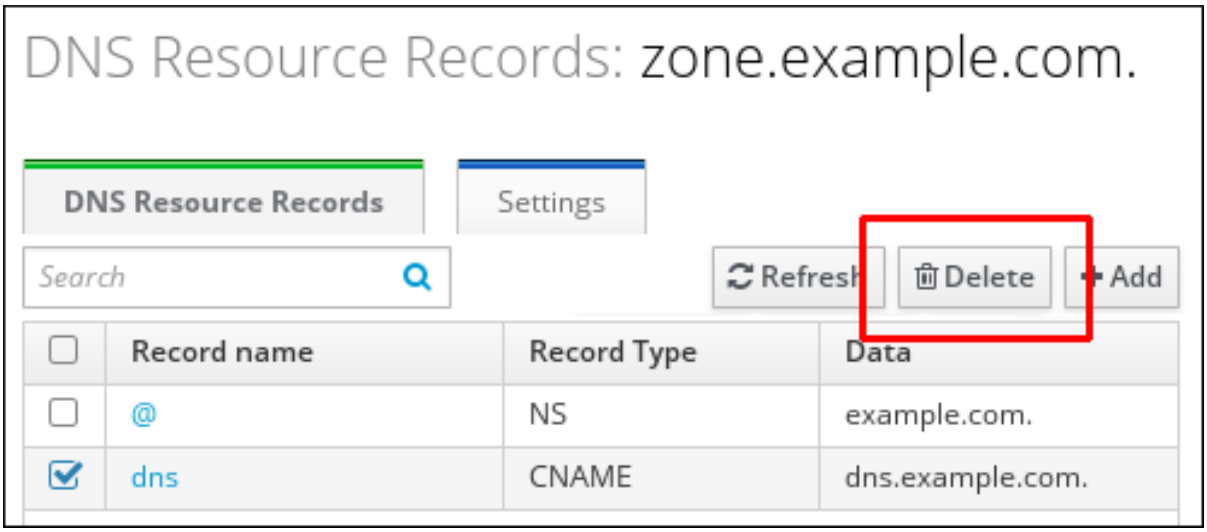
前提条件

- IdM 管理者としてログインしている。

手順

1. IdM Web UI で、**Network Services → DNS → DNS Zones** の順にクリックします。
2. DNS レコードを削除するゾーン (例: **zone.example.com**) をクリックします。
3. **DNS Resource Record** セクションで、削除するリソースレコードのチェックボックスを選択します。
4. **Delete** をクリックします。

図97.5 全リソースレコードの削除



リソースレコードがすべて削除されました。

97.7. IDM CLI での DNS レコードの削除

Identity Management (IdM) DNS によって管理されるゾーンから DNS レコードを削除するには、次の手順に従います。

前提条件

- IdM 管理者としてログインしている。

手順

- ゾーンからレコードを削除するには **ipa dnsrecord-del** コマンドを使用して、**--recordType-rec** オプションでレコードの値を指定して追加します。たとえば、A タイプのレコードを削除するには以下を実行します。

```
$ ipa dnsrecord-del example.com www --a-rec 192.0.2.1
```

オプションなしで **ipa dnsrecord-del** コマンドを実行すると、削除するレコードについての情報の入力が求められます。**--del-all** オプションを指定してコマンドを実行すると、ゾーンに関連するレコードがすべて削除されることに注意してください。

関連情報

- **ipa dnsrecord-del --help** コマンドを実行します。

97.8. 関連情報

- Ansible を使用した IdM での DNS レコードの管理 を参照してください。

第98章 外部 DNS の使用時の DNS レコードのシステム的な更新

外部 DNS を使用する場合には、Identity Management は、トポロジーの変更後に DNS レコードを自動的に更新しません。外部 DNS サービスによって管理されている DNS レコードを体系的に更新できるため、手動での DNS 更新の必要性が軽減されます。

DNS レコードを更新すると、古い DNS レコードまたは無効な DNS レコードが削除され、新しいレコードが追加されます。トポロジーの変更後に、DNS レコードを更新する必要があります。以下に例を示します。

- レプリカのインストールまたはアンインストール後
- IdM サーバーに CA、DNS、KRA、または Active Directory 信頼をインストールした後

98.1. GUI を使用した外部 DNS レコードの更新

トポロジーに変更を加えた場合は、外部 DNS GUI を使用して外部 DNS レコードを更新する必要があります。

手順

1. 更新が必要なレコードを表示します。

```
$ ipa dns-update-system-records --dry-run
 IPA DNS records:
   _kerberos-master._tcp.example.com. 86400 IN SRV 0 100 88 ipa.example.com.
   _kerberos-master._udp.example.com. 86400 IN SRV 0 100 88 ipa.example.com.
[... output truncated ...]
```

2. 外部 DNS GUI を使用して、レコードを更新します。

98.2. NSUPDATE を使用して外部 DNS レコードを更新する

nsupdate ユーティリティーを使用して外部 DNS レコードを更新できます。プロセスを自動化するために、スクリプトにコマンドを追加することもできます。**nsupdate** ユーティリティーを使用して更新するには、DNS レコードを含むファイルを生成し、TSIG を使用して保護された **nsupdate** 要求を送信するか、GSS-TSIG を使用して保護された **nsupdate** 要求を送信する必要があります。

手順

- nsupdate の DNS レコードを含むファイルを生成するには、**--out** オプションを指定した ` **ipa dns- 更新 -system-records --dry-run コマンドを使用します**。このオプションは、生成するファイルのパスを指定します。

```
$ ipa dns-update-system-records --dry-run --out dns_records_file.nsupdate
 IPA DNS records:
   _kerberos-master._tcp.example.com. 86400 IN SRV 0 100 88 ipa.example.com.
   _kerberos-master._udp.example.com. 86400 IN SRV 0 100 88 ipa.example.com.
[... output truncated ...]
```

  生成されたファイルには、**nsupdate** ユーティリティーが許可する形式で、必要な DNS レコードが含まれます。

- 生成されるレコードは、以下に依存します。

- レコードを更新するゾーンの自動検出
- ゾーンの権威サーバーの自動検出
  標準以外の DNS 設定を使用しているか、ゾーンの委譲がない場合は、**nsupdate** が正しいゾーンとサーバーを見つけられない可能性があります。この場合は、生成されるファイルの先頭に以下のオプションを追加します。
- **server** は、**nsupdate** がレコードを送信する信頼できる DNS サーバーのサーバー名またはポートを指定します。
- **zone** : **nsupdate が** レコードを配置するゾーンの名前を指定します。

例98.1 生成されたレコード

```
$ cat dns_records_file.nsupdate
zone example.com.
server 192.0.2.1
; IPA DNS records
update delete _kerberos-master._tcp.example.com. SRV
update add _kerberos-master._tcp.example.com. 86400 IN SRV 0 100 88
ipa.example.com.
[... output truncated ...]
```

98.3. TSIG を使用した NSUPDATE 要求のセキュアな送信

nsupdate を使用して要求を送信する場合は、要求を適切にセキュリティー保護してください。トランザクション署名 (TSIG) を使用すると、共有キーで **nsupdate** を使用できるようになります。

前提条件

- TSIG に DNS サーバーを設定する必要があります。
- DNS サーバーとそのクライアントの両方に、共有鍵が必要です。

手順

- **nsupdate** コマンドを実行し、次のいずれかのオプションを使用して共有シークレットを指定します。
  - **-k**: TSIG 認証キーを指定します。

    ```
    $ nsupdate -k tsig_key.file dns_records_file.nsupdate
    ```

  - **-y**: 鍵の名前と Base64 でエンコードされた共有秘密鍵から署名を生成します。

    ```
    $ nsupdate -y algorithm:keyname:secret dns_records_file.nsupdate
    ```

98.4. GSS-TSIG を使用した NSUPDATE 要求のセキュアな送信

nsupdate を使用して要求を送信する場合は、要求を適切にセキュリティー保護してください。GSS-TSIG は、GSS-API インターフェイスを使用して秘密の TSIG 鍵を取得します。GSS-TSIG は、TSIG プロトコルの拡張機能です。

前提条件

- DNS サーバーは GSS-TSIG 用に設定する必要があります。

注記

この手順では、Kerberos V5 プロトコルが GSS-API のテクノロジーとして使用されていることを前提としています。

手順

1. レコードの更新を許可されたプリンシパルで認証します。

```
$ kinit principal_allowed_to_update_records@REALM
```

2. GSS-TSIG モードを有効にするには、**-g** オプションを指定して **nsupdate** を 実行します。

```
$ nsupdate -g dns_records_file.nsupdate
```

98.5. 関連情報

- **nsupdate (8)** man ページ
- RFC 2845 では TSIG プロトコルが説明されています。
- RFC 3645 は GSS-TSIG アルゴリズムを記述します。

第99章 ANSIBLE を使用した IDM での DNS レコードの管理

本章では、Ansible Playbook を使用して Identity Management (IdM) で DNS レコードを管理する方法を説明します。IdM 管理者は、IdM で DNS レコードの追加、変更、および削除が可能です。本章は以下のセクションで設定されます。

- Ansible を使用して IdM に A および AAAA DNS レコードが存在させる手順
- Ansible を使用して IdM に A および PTR DNS レコードを存在させる手順
- Ansible を使用して IdM に複数の DNS レコードを存在させる手順
- Ansible を使用して IdM に複数の CNAME レコードを存在させる手順
- Ansible を使用して IdM に SRV レコードを存在させる手順

99.1. IDM の DNS レコード

Identity Management (IdM) は、多種の DNS レコードに対応します。以下の 4 つが最も頻繁に使用されます。

A

これは、ホスト名および IPv4 アドレスの基本マップです。A レコードのレコード名は、**www** などのホスト名です。A レコードの **IP アドレス** 値は、**192.0.2.1** などの IPv4 アドレスです。
A レコードの詳細は、RFC 1035 を参照してください。

AAAA

これは、ホスト名および IPv6 アドレスの基本マップです。AAAA レコードのレコード名は **www** などのホスト名です。**IP アドレス** の値は、**2001:DB8::1111** などの IPv6 アドレスです。
AAAA レコードの詳細は RFC 3596 を参照してください。

SRV

サービス (SRV) リソースレコード は、特定のサービスを提供するサーバーの DNS 名にサービス名をマッピングします。たとえば、このタイプのレコードは LDAP ディレクトリーのようなサービスを管理するサーバーに、このサービスをマッピングします。
SRV レコードのレコード名は、**_ldap._tcp** など、**_service._protocol** の形式を取ります。SRV レコードの設定オプションには、ターゲットサービスの優先順位、加重、ポート番号、およびホスト名が含まれます。

SRV レコードの詳細は、RFC 2782 を参照してください。

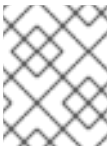
PTR

ポインターレコード (PTR) は、IP アドレスをドメイン名にマッピングする逆引き DNS レコードを追加します。

注記

IPv4 アドレスの逆引き DNS ルックアップはすべて、**in-addr.arpa.** ドメインで定義される逆引きエントリーを使用します。人間が判別可能な形式の逆アドレスは、通常の IP とまったく逆で、**in-addr.arpa.** ドメインが最後に付いています。たとえば、ネットワークアドレス **192.0.2.0/24** の逆引きゾーンは、**2.0.192.in-addr.arpa** になります。

PTR レコード名は、RFC 1035 (RFC 2317 および RFC 3596 で拡張) で指定の標準形式を仕様する必要があります。ホスト名の値は、レコードを作成するホストの正規のホスト名である必要があります。



注記

また、IPv6 アドレスの逆引きゾーンは、**.ip6.arpa.** ドメインのゾーンを使用して設定できます。IPv6 逆引きゾーンの詳細は、RFC 3596 を参照してください。

DNS リソースレコードの追加時には、レコードの多くで異なるデータが必要になることに注意してください。たとえば、CNAME レコードにはホスト名が必要ですが、A レコードには IP アドレスが必要です。IdM Web UI では、新しいレコードを追加するフォームのフィールドが自動的に更新され、現在選択されているレコードタイプに必要なデータが反映されます。

99.2. 一般的な IPA DNSRECORD-* オプション

Identity Management (IdM) で最も一般的な DNS リソースレコードタイプを追加、変更、および削除する場合は、以下のオプションを使用できます。

- A (IPv4)
- AAAA (IPv6)
- SRV
- PTR

Bash では、**--option={val1,val2,val3}** など、波括弧の中にコンマ区切りで値を指定して、複数のエントリーを定義できます。

表99.1 全般的なレコードのオプション

オプション	説明
--ttl=number	レコードの有効期間を設定します。
--structured	raw DNS レコードを解析し、それらを構造化された形式で返します。

表99.2 "A" レコードのオプション

オプション	説明	例
--a-rec=ARECORD	A レコードを 1 つまたはリストで指定します。	**ipa dnsrecord-add idm.example.com host1 --a-rec=192.168.122.123**
	指定の IP アドレスでワイルドカード A レコードを作成できます。	**ipa dnsrecord-add idm.example.com "*" --a-rec=192.168.122.123** [a]

オプション	説明	例
--a-ip-address=string	レコードの IP アドレスを渡します。レコードの作成時に、**A** レコードの値を指定するオプションは **--a-rec** です。ただし **A** レコードを変更する時に、**--a-rec** オプションを使用して **A** レコードの現在の値を指定します。新しい値は、**--a-ip-address** オプションで設定します。	**ipa dnsrecord-mod idm.example.com --a-rec 192.168.122.123 --a-ip-address 192.168.122.124**
[a] この例では、IP アドレスが 192.0.2.123 のワイルドカード **A** レコードを作成します。		

表99.3 "AAAA" レコードのオプション

オプション	説明	例
--aaaa-rec=AAAARECORD	AAAA (IPv6) レコードを 1 つまたはリストで指定します。	**ipa dnsrecord-add idm.example.com www --aaaa-rec 2001:db8::1231:5675**
--aaaa-ip-address=string	レコードの IPv6 アドレスを渡します。レコードの作成時に、**A** レコードの値を指定するオプションは **--aaaa-rec** です。ただし、**A** レコードを変更する時に、**--aaaa-rec** オプションを使用して **A** レコードの現在の値を指定します。新しい値は、**--a-ip-address** オプションで設定します。	**ipa dnsrecord-mod idm.example.com --aaaa-rec 2001:db8::1231:5675 --aaaa-ip-address 2001:db8::1231:5676**

表99.4 "PTR" レコードのオプション

オプション	説明	例
--ptr-rec=PTRRECORD	PTR レコードを 1 つまたはリストで指定します。逆引き DNS レコードを追加する時には、他の DNS レコードの追加の方法と比べ、**ipa dnsrecord-add** コマンドで使用するゾーン名は、逆になります。通常、ホストの IP アドレスは、指定のネットワークにおける IP アドレスの最後のオクテットを使用します。右側の最初の例では、IPv4 アドレスが 192.168.122.4. の server4.idm.example.com の PTR レコードを追加します。2 番目の例では、0.0.0.0.0.0.0.0.8.b.d.0.1.0.0.2.ip6.arpa. に逆引き DNS エントリーを追加します。IP アドレスが 2001:DB8::1111 の server2.example.com ホストの IPv6 逆引きゾーン。	**ipa dnsrecord-add 122.168.192.in-addr.arpa 4 --ptr-rec server4.idm.example.com.** **$ ipa dnsrecord-add 0.0.0.0.0.0.0.0.8.b.d.0.1.0.0.2.ip6.arpa.1.1.1.0.0.0.0.0.0.0.0.0.0.0.0 --ptr-rec server2.idm.example.com.**
--ptr-hostname=string	レコードのホスト名を指定します。	

表99.5 "SRV" レコードのオプション

オプション	説明	例
--srv-rec=SRVRECORD	SRV レコードを 1 つまたはリストで指定します。右側の例では、_ldap._tcp は、SRV レコードのサービスタイプと接続プロトコルを定義します。**--srv-rec** オプションは、優先順位、加重、ポート、およびターゲットの値を定義します。この例では、加重値 51 と 49 が最大 100 まで加算され、特定のレコードが使用される確率を % で表します。	**# ipa dnsrecord-add idm.example.com _ldap._tcp --srv-rec="0 51 389 server1.idm.example.com."** **# ipa dnsrecord-add server.idm.example.com _ldap._tcp --srv-rec="1 49 389 server2.idm.example.com."**
--srv-priority=number	レコードの優先順位を設定します。あるサービスタイプに複数の SRV レコードがある場合もあります。優先順位 (0 - 65535) はレコードの階級を設定し、数字が小さいほど優先順位が高くなります。サービスは、優先順位の最も高いレコードを最初に使用する必要があります。	**# ipa dnsrecord-mod server.idm.example.com _ldap._tcp --srv-rec="1 49 389 server2.idm.example.com." --srv-priority=0**
--srv-weight=number	レコードの加重を設定します。これは、SRV レコードの優先順位が同じ場合に順序を判断する際に役立ちます。設定された加重は最大 100 とし、これは特定のレコードが使用される可能性をパーセンテージで示しています。	**# ipa dnsrecord-mod server.idm.example.com _ldap._tcp --srv-rec="0 49 389 server2.idm.example.com." --srv-weight=60**
--srv-port=number	ターゲットホスト上のサービスのポートを渡します。	**# ipa dnsrecord-mod server.idm.example.com _ldap._tcp --srv-rec="0 60 389 server2.idm.example.com." --srv-port=636**
--srv-target=string	ターゲットホストのドメイン名を提供します。該当サービスがドメイン内で利用可能でない場合は、単一のピリオド (.) として指定される場合があります。	

関連情報

- **ipa dnsrecord-add --help** を実行します。

99.3. ANSIBLE を使用して IDM に A および AAAA DNS レコードが存在させる手順

Ansible Playbook を使用して、特定の IdM ホストの A および AAAA レコードが存在することを確認するには、以下の手順に従います。以下の手順で使用する例では、IdM 管理者は **idm.example.com** DNS ゾーンに **host1** の A レコードおよび AAAA レコードを追加します。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- IdM 管理者パスワードを把握している。
- **idm.example.com** ゾーンが存在しており、IdM DNS が管理する。IdM DNS にプライマリー DNS ゾーンを追加する方法は、Ansible Playbook を使用した IdM DNS ゾーンの管理 を参照してください。

手順

1. **/usr/share/doc/ansible-freeipa/playbooks/dnsrecord** ディレクトリーに移動します。

```
$ cd /usr/share/doc/ansible-freeipa/playbooks/dnsrecord
```

2. インベントリーファイルを開き、設定する IdM サーバーが **[ipaserver]** セクションに記載されていることを確認します。たとえば、Ansible に対して **server.idm.example.com** を設定するように指示するには、次のコマンドを実行します。

```
[ipaserver]
server.idm.example.com
```

3. Ansible Playbook ファイル (**ensure-A-and-AAAA-records-are-present.yml**) のコピーを作成します。以下に例を示します。

```
$ cp ensure-A-and-AAAA-records-are-present.yml ensure-A-and-AAAA-records-are-present-copy.yml
```

4. **ensure-A-and-AAAA-records-are-present-copy.yml** ファイルを開いて編集します。
5. **ipadnsrecord** タスクセクションで以下の変数を設定して、ファイルを調整します。
   - **ipaadmin_password** 変数は IdM 管理者パスワードに設定します。
   - **zone_name** 変数は **idm.example.com** に設定します。
   - **record** 変数で、**name** 変数は **host1** に、**a_ip_address** 変数は 192.168.122.123 に設定します。
   - **record** 変数で、**name** 変数は **host1** に、**aaaa_ip_address** 変数は **::1** に設定します。
     以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Ensure A and AAAA records are present
  hosts: ipaserver
  become: true
  gather_facts: false

  tasks:
  # Ensure A and AAAA records are present
  - name: Ensure that 'host1' has A and AAAA records.
    ipadnsrecord:
      ipaadmin_password: "{{ ipaadmin_password }}"
      zone_name: idm.example.com
      records:
      - name: host1
        a_ip_address: 192.168.122.123
      - name: host1
        aaaa_ip_address: ::1
```

6. ファイルを保存します。

7. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file ensure-A-and-AAAA-records-are-present-copy.yml
```

関連情報

- IdM での DNS レコード を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-dnsrecord.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/dnsrecord** ディレクトリーのサンプルの Ansible Playbook を参照してください。

99.4. ANSIBLE を使用して IDM に A および PTR DNS レコードを存在させる手順

以下の手順に従って、Ansible Playbook を使用して、特定の IdM ホストの A レコードと、対応する PTR レコードが存在することを確認します。以下の手順で使用する例では、IdM 管理者は、**idm.example.com** ゾーンで IP アドレスが **192.168.122.45** の **host1** の A レコードと PTR レコードを追加します。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。

  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。

- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- IdM 管理者パスワードを把握している。
- **idm.example.com** DNS ゾーンが存在しており、IdM DNS が管理する。IdM DNS にプライマリー DNS ゾーンを追加する方法は、Ansible Playbook を使用した IdM DNS ゾーンの管理 を参照してください。

手順

1. **/usr/share/doc/ansible-freeipa/playbooks/dnsrecord** ディレクトリーに移動します。

```
$ cd /usr/share/doc/ansible-freeipa/playbooks/dnsrecord
```

2. インベントリーファイルを開き、設定する IdM サーバーが **[ipaserver]** セクションに記載されていることを確認します。たとえば、Ansible に対して **server.idm.example.com** を設定するように指示するには、次のコマンドを実行します。

```
[ipaserver]
server.idm.example.com
```

3. Ansible Playbook ファイル (**ensure-dnsrecord-with-reverse-is-present.yml**) のコピーを作成します。以下に例を示します。

```
$ cp ensure-dnsrecord-with-reverse-is-present.yml ensure-dnsrecord-with-reverse-is-present-copy.yml
```

4. **ensure-dnsrecord-with-reverse-is-present-copy.yml** ファイルを開いて編集します。
5. **ipadnsrecord** タスクセクションで以下の変数を設定して、ファイルを調整します。
   - **ipaadmin_password** 変数は IdM 管理者パスワードに設定します。
   - **name** 変数は **host1** に設定します。
   - **zone_name** 変数は **idm.example.com** に設定します。
   - **ip_address** 変数は、**192.168.122.45** に設定します。
   - **create_reverse** 変数は **yes** に設定します。
     以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Ensure DNS Record is present.
  hosts: ipaserver
  become: true
  gather_facts: false

  tasks:
  # Ensure that dns record is present
  - ipadnsrecord:
```

```
    ipaadmin_password: "{{ ipaadmin_password }}"
    name: host1
    zone_name: idm.example.com
    ip_address: 192.168.122.45
    create_reverse: yes
    state: present
```

6. ファイルを保存します。

7. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file ensure-dnsrecord-with-reverse-is-present-copy.yml
```

関連情報

- IdM での DNS レコード を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-dnsrecord.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/dnsrecord** ディレクトリーのサンプルの Ansible Playbook を参照してください。

99.5. ANSIBLE を使用して IDM に複数の DNS レコードを存在させる手順

Ansible Playbook を使用して、複数の値が特定の IdM DNS レコードに関連付けられるようにするには、以下の手順に従います。以下の手順で使用する例では、IdM 管理者は **idm.example.com** DNS ゾーンに **host1** の A レコードを複数追加します。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- IdM 管理者パスワードを把握している。
- **idm.example.com** ゾーンが存在しており、IdM DNS が管理する。IdM DNS にプライマリー DNS ゾーンを追加する方法は、Ansible Playbook を使用した IdM DNS ゾーンの管理 を参照してください。

手順

1. **/usr/share/doc/ansible-freeipa/playbooks/dnsrecord** ディレクトリーに移動します。

```
$ cd /usr/share/doc/ansible-freeipa/playbooks/dnsrecord
```

2. インベントリーファイルを開き、設定する IdM サーバーが **[ipaserver]** セクションに記載されていることを確認します。たとえば、Ansible に対して **server.idm.example.com** を設定するように指示するには、次のコマンドを実行します。

```
[ipaserver]
server.idm.example.com
```

3. Ansible Playbook ファイル (**ensure-presence-multiple-records.yml**) のコピーを作成します。以下に例を示します。

```
$ cp ensure-presence-multiple-records.yml ensure-presence-multiple-records-copy.yml
```

4. **ensure-presence-multiple-records-copy.yml** ファイルを開いて編集します。

5. **ipadnsrecord** タスクセクションで以下の変数を設定して、ファイルを調整します。

   - **ipaadmin_password** 変数は IdM 管理者パスワードに設定します。
   - **records** セクションで、**name** 変数を **host1** に設定します。
   - **record** セクションで、**zone_name** 変数を **idm.example.com** に設定します。
   - **record** セクションで、**a_rec** 変数を 192.168.122.112 に、192.168.122.122 **に** 設定し **ます**。
   - **records** セクションの 2 番目のレコードを定義します。
     - **name** 変数は **host1** に設定します。
     - **zone_name** 変数は **idm.example.com** に設定します。
     - **aaaa_rec** 変数は **::1** に設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Test multiple DNS Records are present.
  hosts: ipaserver
  become: true
  gather_facts: false

  tasks:
  # Ensure that multiple dns records are present
  - ipadnsrecord:
      ipaadmin_password: "{{ ipaadmin_password }}"
      records:
        - name: host1
          zone_name: idm.example.com
          a_rec: 192.168.122.112
          a_rec: 192.168.122.122
```

```
    - name: host1
      zone_name: idm.example.com
      aaaa_rec: ::1
```

6. ファイルを保存します。

7. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file ensure-presence-multiple-records-copy.yml
```

関連情報

- IdM での DNS レコード を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-dnsrecord.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/dnsrecord** ディレクトリーのサンプルの Ansible Playbook を参照してください。

99.6. ANSIBLE を使用して IDM に複数の CNAME レコードを存在させる手順

Canonical Name レコード (CNAME レコード) は、DNS (Domain Name System) のリソースレコードの一種で、別の名前 (CNAME) にドメイン名、エイリアスをマッピングします。

CNAME レコードは、FTP サービスと Web サービスがそれぞれ別のポートで実行されている場合など、1 つの IP アドレスから複数のサービスを実行する場合に、役立つ可能性があります。

Ansible Playbook を使用して、複数の CNAME レコードが IdM DNS に存在することを確認するには、以下の手順に従います。以下の手順で使用する例では、**host03** は HTTP サーバーと FTP サーバーの両方として機能します。IdM 管理者は、**idm.example.com** ゾーンに **host03** A レコードの **www** および **ftp** CNAME レコードを追加します。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
- IdM 管理者パスワードを把握している。

- **idm.example.com** ゾーンが存在しており、IdM DNS が管理する。IdM DNS にプライマリー DNS ゾーンを追加する方法は、Ansible Playbook を使用した IdM DNS ゾーンの管理 を参照してください。
- **host03** A レコードが **idm.example.com** ゾーンに存在している。

手順

1. **/usr/share/doc/ansible-freeipa/playbooks/dnsrecord** ディレクトリーに移動します。

```
$ cd /usr/share/doc/ansible-freeipa/playbooks/dnsrecord
```

2. インベントリーファイルを開き、設定する IdM サーバーが **[ipaserver]** セクションに記載されていることを確認します。たとえば、Ansible に対して **server.idm.example.com** を設定するように指示するには、次のコマンドを実行します。

```
[ipaserver]
server.idm.example.com
```

3. Ansible Playbook ファイル (**ensure-CNAME-record-is-present.yml**) のコピーを作成します。以下に例を示します。

```
$ cp ensure-CNAME-record-is-present.yml ensure-CNAME-record-is-present-copy.yml
```

4. **ensure-CNAME-record-is-present-copy.yml** ファイルを開いて編集します。
5. **ipadnsrecord** タスクセクションで以下の変数を設定して、ファイルを調整します。
   - (任意) Play の **name** で提示された説明を調整します。
   - **ipaadmin_password** 変数は IdM 管理者パスワードに設定します。
   - **zone_name** 変数は **idm.example.com** に設定します。
   - **record** 変数セクションで、以下の変数および値を設定します。
     - **name** 変数は **www** に設定します。
     - **cname_hostname** 変数は **host03** に設定します。
     - **name** 変数は **ftp** に設定します。
     - **cname_hostname** 変数は **host03** に設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Ensure that 'www.idm.example.com' and 'ftp.idm.example.com' CNAME records
point to 'host03.idm.example.com'.
  hosts: ipaserver
  become: true
  gather_facts: false

  tasks:
  - ipadnsrecord:
      ipaadmin_password: "{{ ipaadmin_password }}"
```

```
  zone_name: idm.example.com
  records:
  - name: www
    cname_hostname: host03
  - name: ftp
    cname_hostname: host03
```

6. ファイルを保存します。

7. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file ensure-CNAME-record-is-present.yml
```

関連情報

- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-dnsrecord.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/dnsrecord** ディレクトリーのサンプルの Ansible Playbook を参照してください。

99.7. ANSIBLE を使用して IDM に SRV レコードを存在させる手順

DNS サービス (SRV) レコードは、ドメインで利用可能なサービスのホスト名、ポート番号、トランスポートプロトコル、優先度、および加重を定義します。Identity Management (IdM) では、SRV レコードを使用して、IdM サーバーとレプリカを特定できます。

以下の手順に従って、Ansible Playbook を使用して、SRV レコードが IdM DNS に存在することを確認します。以下の手順で使用する例では、IdM の管理者が **10 50 88 idm.example.com** の値を指定して **_kerberos._udp.idm.example.com** SRV レコードを追加します。この例では、以下の値を指定します。

- サービスの優先度を 10 に設定します。
- サービスの加重を 50 に設定します。
- サービスが使用するポートを 88 に設定します。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

- IdM 管理者パスワードを把握している。
- **idm.example.com** ゾーンが存在しており、IdM DNS が管理する。IdM DNS にプライマリー DNS ゾーンを追加する方法は、Ansible Playbook を使用した IdM DNS ゾーンの管理 を参照してください。

手順

1. **/usr/share/doc/ansible-freeipa/playbooks/dnsrecord** ディレクトリーに移動します。

```
$ cd /usr/share/doc/ansible-freeipa/playbooks/dnsrecord
```

2. インベントリーファイルを開き、設定する IdM サーバーが **[ipaserver]** セクションに記載されていることを確認します。たとえば、Ansible に対して **server.idm.example.com** を設定するように指示するには、次のコマンドを実行します。

```
[ipaserver]
server.idm.example.com
```

3. Ansible Playbook ファイル (**ensure-SRV-record-is-present.yml**) のコピーを作成します。以下に例を示します。

```
$ cp ensure-SRV-record-is-present.yml ensure-SRV-record-is-present-copy.yml
```

4. **ensure-SRV-record-is-present-copy.yml** ファイルを開いて編集します。
5. **ipadnsrecord** タスクセクションで以下の変数を設定して、ファイルを調整します。
   - **ipaadmin_password** 変数は IdM 管理者パスワードに設定します。
   - **name** 変数は **_kerberos._udp.idm.example.com** に設定します。
   - **srv_rec** 変数は **'10 50 88 idm.example.com'** に設定します。
   - **zone_name** 変数は **idm.example.com** に設定します。
     今回の例で使用するように変更した Ansible Playbook ファイル:

```
---
- name: Test multiple DNS Records are present.
  hosts: ipaserver
  become: true
  gather_facts: false

  tasks:
  # Ensure a SRV record is present
  - ipadnsrecord:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: _kerberos._udp.idm.example.com
      srv_rec: '10 50 88 idm.example.com'
      zone_name: idm.example.com
      state: present
```

6. ファイルを保存します。
7. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file ensure-SRV-record-is-present.yml
```

関連情報

- IdM での DNS レコード を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-dnsrecord.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/dnsrecord** ディレクトリーのサンプルの Ansible Playbook を参照してください。

第100章 ANSIBLE を使用した IDM サーバーの管理

Red Hat Ansible Engine を使用すると、Identity Management (IdM) トポロジーのサーバーを管理できます。**ansible-freeipa** パッケージの **server** モジュールを使用して、IdM トポロジーにサーバーの有無を確認できます。任意のレプリカを非表示にしたり、レプリカを表示したりすることもできます。

このセクションには、以下のトピックが含まれます。

- Ansible を使用した IdM サーバーの存在の確認
- Ansible を使用した IdM トポロジーに IdM サーバーが存在しないことの確認
- 最後の IdM サーバーロールをホストしているにもかかわらず IdM サーバーがないことの確認
- IdM サーバーが存在しないが、必ずしも他の IdM サーバーから切断されていないことの確認
- Ansible Playbook を使用した既存の IdM サーバーが非表示であることの確認
- Ansible Playbook を使用した既存の IdM サーバーが表示されていることの確認
- 既存の IdM サーバーに IdM DNS の場所が割り当てられていることの確認
- 既存の IdM サーバーに IdM DNS の場所が割り当てられていないことの確認

100.1. ANSIBLE を使用した IDM サーバーの存在の確認

Ansible Playbook で **ipaserver ansible-freeipa** モジュールを使用して、Identity Management (IdM) サーバーが存在することを確認できます。

注記

ipaserver Ansible モジュールは、IdM サーバーをインストールしません。

前提条件

- IdM **admin** のパスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
  - 制御ノードからインベントリーファイルに定義した IdM サーバーへの **SSH** 接続が正常に動作している。

手順

1. **~/MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/server/** ディレクトリーにある **server-present.yml** Ansible Playbook ファイルをコピーします。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/server/server-present.yml server-present-copy.yml
```

3. **server-present-copy.yml** を開いて編集します。

4. **ipaserver** タスクセクションで次の変数を設定してファイルを調整し、ファイルを保存します。

- **ipaadmin_password** 変数は IdM **admin** のパスワードに設定します。
- **name** 変数をサーバーの **FQDN** に設定します。サンプルサーバーの **FQDN** は server123.idm.example.com です。

```
---
- name: Server present example
  hosts: ipaserver
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure server server123.idm.example.com is present
    ipaserver:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: server123.idm.example.com
```

5. Ansible Playbook を実行し、Playbook ファイルとインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory server-present-copy.yml
```

関連情報

- Ansible Playbook を使用した Identity Management サーバーのインストール を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-server.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/server** ディレクトリーのサンプルの Playbook を参照してください。

100.2. ANSIBLE を使用した IDM トポロジーに IDM サーバーが存在しないことの確認

Ansible Playbook を使用して、Identity Management (IdM) サーバーが、ホストとしても IdM トポロジーに存在しないようにします。

ansible-freeipa ipaserver ロールとは対照的に、この Playbook で使用する **ipaserver** モジュールは、サーバーから IdM サービスをアンインストールしません。

前提条件

- IdM **admin** のパスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
  - 制御ノードからインベントリーファイルに定義した IdM サーバーへの **SSH** 接続が正常に動作している。

手順

1. **~/MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/server/** ディレクトリーにある **server-absent.yml** Ansible Playbook ファイルをコピーします。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/server/server-absent.yml server-absent-copy.yml
```

3. **server-absent-copy.yml** を開いて編集します。
4. **ipaserver** タスクセクションで次の変数を設定してファイルを調整し、ファイルを保存します。
   - **ipaadmin_password** 変数は IdM **admin** のパスワードに設定します。
   - **name** 変数をサーバーの **FQDN** に設定します。サンプルサーバーの **FQDN** は **server123.idm.example.com** です。
   - **state** 変数は、**absent** に設定されていることを確認します。

```
---
- name: Server absent example
  hosts: ipaserver
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure server server123.idm.example.com is absent
```

```
ipaserver:
  ipaadmin_password: "{{ ipaadmin_password }}"
  name: server123.idm.example.com
  state: absent
```

5. Ansible Playbook を実行し、Playbook ファイルとインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory server-absent-copy.yml
```

6. **server123.idm.example.com** を指定しているネームサーバー (NS) の DNS レコードがすべて DNS ゾーンから削除されていることを確認してください。使用する DNS が IdM により管理される統合 DNS であるか、外部 DNS であるかに関わらず、確認を行なってください。

関連情報

- IdM サーバーのアンインストール を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-server.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/server** ディレクトリーのサンプルの Playbook を参照してください。

100.3. 最後の IDM サーバーロールをホストしているにもかかわらず IDM サーバーがないことの確認

Ansible を使用すると、最後の IdM サービスインスタンスがサーバーで実行している場合でも、Identity Management (IdM) サーバーがないことを確認できます。認証局 (CA)、キーリカバリー認証局 (KRA)、または DNS サーバーはすべて IdM サービスの例です。

警告

CA サーバー、KRA サーバー、または DNS サーバーとして機能する最後のサーバーを削除すると、IdM 機能に深刻な不具合が生じます。**ipa service-find** を使用すると、どのサービスがどの IdM サーバーで実行されているかを手動で確認できます。認証局サーバーのプリンシパル名は **dogtag/server_name/REALM_NAME** です。

ansible-freeipa ipaserver ロールとは対照的に、この Playbook で使用する **ipaserver** モジュールは、サーバーから IdM サービスをアンインストールしません。

前提条件

- IdM **admin** のパスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。

- Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
- この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
- この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。

- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
  - 制御ノードからインベントリーファイルに定義した IdM サーバーへの **SSH** 接続が正常に動作している。

手順

1. **~/MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/server/** ディレクトリーにある **server-absent-ignore-last-of-role.yml** Ansible Playbook ファイルをコピーします。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/server/server-absent-ignore-last-of-role.yml server-absent-ignore-last-of-role-copy.yml
```

3. **server-absent-ignore-last-of-role-copy.yml** ファイルを開いて編集します。
4. **ipaserver** タスクセクションで次の変数を設定してファイルを調整し、ファイルを保存します。
   - **ipaadmin_password** 変数は IdM **admin** のパスワードに設定します。
   - **name** 変数をサーバーの **FQDN** に設定します。サンプルサーバーの **FQDN** は **server123.idm.example.com** です。
   - **ignore_last_of_role** が **yes** に設定されていることを確認します。
   - **state** 変数は **absent** に設定します。

```
---
- name: Server absent with last of role skip example
  hosts: ipaserver
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure server “server123.idm.example.com” is absent with last of role skip
    ipaserver:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: server123.idm.example.com
      ignore_last_of_role: yes
      state: absent
```

5. Ansible Playbook を実行し、Playbook ファイルとインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory server-absent-ignore-last-of-role-copy.yml
```

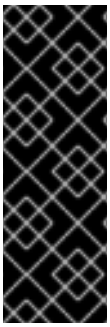
6. server123.idm.example.com を指定するネームサーバー (NS) の DNS レコードが、すべて DNS ゾーンから削除されていることを確認してください。使用する DNS が IdM により管理される統合 DNS であるか、外部 DNS であるかに関わらず、確認を行なってください。

関連情報

- IdM サーバーのアンインストール を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-server.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/server** ディレクトリーのサンプルの Playbook を参照してください。

100.4. IDM サーバーが存在しないが、必ずしも他の IDM サーバーから切断されていないことの確認

トポロジーから Identity Management (IdM) サーバーを削除する場合は、Ansible Playbook でレプリケーションアグリーメントをそのまま保持できます。Playbook では、IdM サーバーがホストとしても IdM に存在しないことも確認します。



重要

削除する際にサーバーのレプリカ合意を無視することが推奨されるのは、削除を予定している他のサーバーが機能不全のサーバーである場合のみです。トポロジーの中心点として機能するサーバーを削除すると、トポロジーが 2 つの切断されたクラスターに分割される可能性があります。

機能不全のサーバーは、**ipa server-del** コマンドを使用してトポロジーから削除できます。

注記

認証局 (CA)、キーリカバリー認証局 (KRA)、または DNS サーバーとして機能する最後のサーバーを削除すると、Identity Management (IdM) 機能に深刻な不具合が生じます。この問題を防ぐため、Playbook は、CA サーバー、KRA サーバー、または DNS サーバーとして機能するサーバーをアンインストールする前に、これらのサービスがドメインの別のサーバーで実行していることを確認します。

ansible-freeipa ipaserver ロールとは対照的に、この Playbook で使用する **ipaserver** モジュールは、サーバーから IdM サービスをアンインストールしません。

前提条件

- IdM **admin** のパスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。

- この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
- この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。

- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
  - 制御ノードからインベントリーファイルに定義した IdM サーバーへの **SSH** 接続が正常に動作している。

手順

1. **~/MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/server/** ディレクトリーにある **server-absent-ignore_topology_disconnect.yml** Ansible Playbook ファイルをコピーします。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/server/server-absent-ignore_topology_disconnect.yml server-absent-ignore_topology_disconnect-copy.yml
```

3. **server-absent-ignore_topology_disconnect-copy.yml** を開いて編集します。
4. **ipaserver** タスクセクションで次の変数を設定してファイルを調整し、ファイルを保存します。
   - **ipaadmin_password** 変数は IdM **admin** のパスワードに設定します。
   - **name** 変数をサーバーの **FQDN** に設定します。サンプルサーバーの **FQDN** は **server123.idm.example.com** です。
   - **ignore_topology_disconnect** 変数が **yes** に設定されていることを確認します。
   - **state** 変数は、**absent** に設定されていることを確認します。

```
---
- name: Server absent with ignoring topology disconnects example
  hosts: ipaserver
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure server “server123.idm.example.com” with ignoring topology disconnects
    ipaserver:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: server123.idm.example.com
      ignore_topology_disconnect: yes
      state: absent
```

5. Ansible Playbook を実行し、Playbook ファイルとインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory server-absent-ignore_topology_disconnect-copy.yml
```

6. [オプション] **server123.idm.example.com** を指すすべてのネームサーバー (NS) DNS レコードが DNS ゾーンから削除されていることを確認します。使用する DNS が IdM により管理される統合 DNS であるか、外部 DNS であるかに関わらず、確認を行なってください。

関連情報

- IdM サーバーのアンインストール を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-server.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/server** ディレクトリーのサンプルの Playbook を参照してください。

100.5. ANSIBLE PLAYBOOK を使用した既存の IDM サーバーが非表示であることの確認

Ansible Playbook の **ipaserver ansible-freeipa** モジュールを使用して、既存の Identity Management (IdM) サーバーが非表示になっていることを確認します。この Playbook では、IdM サーバーがインストールされないことに注意してください。

前提条件

- IdM **admin** のパスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
  - 制御ノードからインベントリーファイルに定義した IdM サーバーへの **SSH** 接続が正常に動作している。

手順

1. **~/MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/server/** ディレクトリーにある **server-hidden.yml** Ansible Playbook ファイルをコピーします。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/server/server-hidden.yml server-hidden-copy.yml
```

3. **server-hidden-copy.yml** ファイルを開いて編集します。
4. **ipaserver** タスクセクションで次の変数を設定してファイルを調整し、ファイルを保存します。
   - **ipaadmin_password** 変数は IdM **admin** のパスワードに設定します。
   - **name** 変数をサーバーの **FQDN** に設定します。サンプルサーバーの **FQDN** は server123.idm.example.com です。
   - **hidden** 変数が **True** に設定されていることを確認します。

```
---
- name: Server hidden example
  hosts: ipaserver
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure server server123.idm.example.com is hidden
    ipaserver:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: server123.idm.example.com
      hidden: True
```

5. Ansible Playbook を実行し、Playbook ファイルとインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory server-hidden-copy.yml
```

関連情報

- Ansible Playbook を使用した Identity Management サーバーのインストール を参照してください。
- The hidden replica mode を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-server.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/server** ディレクトリーのサンプルの Playbook を参照してください。

100.6. ANSIBLE PLAYBOOK を使用した既存の IDM サーバーが表示されていることの確認

Ansible Playbook で **ipaserver ansible-freeipa** モジュールを使用して、既存の Identity Management (IdM) サーバーが表示されていることを確認します。この Playbook では、IdM サーバーがインストールされないことに注意してください。

前提条件

- IdM **admin** のパスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。

- Ansible バージョン 2.14 以降を使用している。
- Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
- この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
- この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。

- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
  - 制御ノードからインベントリーファイルに定義した IdM サーバーへの **SSH** 接続が正常に動作している。

手順

1. **~/MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/server/** ディレクトリーにある **server-not-hidden.yml** Ansible Playbook ファイルをコピーします。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/server/server-not-hidden.yml server-not-hidden-copy.yml
```

3. **server-not-hidden-copy.yml** ファイルを開いて編集します。
4. **ipaserver** タスクセクションで次の変数を設定してファイルを調整し、ファイルを保存します。
   - **ipaadmin_password** 変数は IdM **admin** のパスワードに設定します。
   - **name** 変数をサーバーの **FQDN** に設定します。サンプルサーバーの **FQDN** は **server123.idm.example.com** です。
   - **hidden** 変数が **no** に設定されていることを確認します。

```
---
- name: Server not hidden example
  hosts: ipaserver
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure server server123.idm.example.com is not hidden
    ipaserver:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: server123.idm.example.com
      hidden: no
```

5. Ansible Playbook を実行し、Playbook ファイルとインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory server-not-hidden-copy.yml
```

関連情報

- Ansible Playbook を使用した Identity Management サーバーのインストール を参照してください。
- The hidden replica mode を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-server.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/server** ディレクトリーのサンプルの Playbook を参照してください。

100.7. 既存の IDM サーバーに IDM DNS の場所が割り当てられていることの確認

Ansible Playbook の **ipaserver ansible-freeipa** モジュールを使用して、既存の Identity Management (IdM) サーバーに特定の IdM DNS の場所が割り当てられていることを確認します。

ipaserver Ansible モジュールは、IdM サーバーをインストールしないことに注意してください。

前提条件

- IdM **admin** のパスワードを把握している。
- IdM DNS の場所が存在します。サンプルの場所は **germany** です。
- サーバーへの **root** アクセス権限がある。サンプルサーバーは **server123.idm.example.com** である。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
  - 制御ノードからインベントリーファイルに定義した IdM サーバーへの **SSH** 接続が正常に動作している。

手順

1. **~/MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/server/** ディレクトリーにある **server-location.yml** Ansible Playbook ファイルをコピーします。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/server/server-location.yml server-location-copy.yml
```

3. **server-location-copy.yml** ファイルを開いて編集します。

4. **ipaserver** タスクセクションで次の変数を設定してファイルを調整し、ファイルを保存します。

   - **ipaadmin_password** 変数は IdM **admin** のパスワードに設定します。
   - **name** 変数を **server123.idm.example.com** に設定します。
   - **location** 変数を **germany** に設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Server enabled example
  hosts: ipaserver
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure server server123.idm.example.com with location “germany” is present
    ipaserver:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: server123.idm.example.com
      location: germany
```

5. Ansible Playbook を実行し、Playbook ファイルとインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory server-location-copy.yml
```

6. **SSH** を使用して、**root** として **server123.idm.example.com** に接続します。

```
ssh root@server123.idm.example.com
```

7. 更新をすぐに有効にするには、サーバーで **named-pkcs11** サービスを再起動します。

```
[root@server123.idm.example.com ~]# systemctl restart named-pkcs11
```

関連情報

- Ansible Playbook を使用した Identity Management サーバーのインストール を参照してください。
- Ansible を使用して IdM の場所が存在することを確認する を参照してください。

- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-server.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/server** ディレクトリーのサンプルの Playbook を参照してください。

100.8. 既存の IDM サーバーに IDM DNS の場所が割り当てられていないことの確認

Ansible Playbook の **ipaserver ansible-freeipa** モジュールを使用して、既存の Identity Management (IdM) サーバーに IdM DNS の場所が割り当てられていないことを確認します。地理的な場所を頻繁に変更するサーバーに DNS の場所を割り当てないでください。Playbook では IdM サーバーがインストールされないことに注意してください。

前提条件

- IdM **admin** のパスワードを把握している。
- サーバーへの **root** アクセス権限がある。サンプルサーバーは **server123.idm.example.com** である。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
  - 制御ノードからインベントリーファイルに定義した IdM サーバーへの **SSH** 接続が正常に動作している。

手順

1. **~/MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/server/** ディレクトリーにある **server-no-location.yml** Ansible Playbook ファイルをコピーします。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/server/server-no-location.yml server-no-location-copy.yml
```

3. **server-no-location-copy.yml** を開いて編集します。
4. **ipaserver** タスクセクションで次の変数を設定してファイルを調整し、ファイルを保存します。

- **ipaadmin_password** 変数は IdM **admin** のパスワードに設定します。
- **name** 変数を server123.idm.example.com に設定します。
- **location** 変数が "" に設定されていることを確認してください。

```
---
- name: Server no location example
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure server server123.idm.example.com is present with no location
    ipaserver:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: server123.idm.example.com
      location: “”
```

5. Ansible Playbook を実行し、Playbook ファイルとインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory server-no-location-copy.yml
```

6. **SSH** を使用して、**root** として server123.idm.example.com に接続します。

```
ssh root@server123.idm.example.com
```

7. 更新をすぐに有効にするには、サーバーで **named-pkcs11** サービスを再起動します。

```
[root@server123.idm.example.com ~]# systemctl restart named-pkcs11
```

関連情報

- Ansible Playbook を使用した Identity Management サーバーのインストール を参照してください。
- Ansible を使用した IdM での DNS の場所の管理 を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-server.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/server** ディレクトリーのサンプルの Playbook を参照してください。

第101章 IDM HEALTHCHECK 情報の収集

Healthcheck は、Identity Management (IdM) で発生する可能性がある問題を特定するのを助ける手動コマンドラインツールとして設計されました。

30 日ローテーションの Healthcheck 出力に基づいてログのコレクションを作成できます。

前提条件

- Healthcheck ツールは、RHEL 8.1 以降でのみ利用できます。

101.1. IDM の HEALTHCHECK

Identity Management (IdM) の Healthcheck ツールは、IdM 環境の健全性に影響を与える可能性のある問題を検出するのに役立ちます。

注記

Healthcheck ツールは、Kerberos 認証なしで使用できるコマンドラインツールです。

独立したモジュール

Healthcheck は、以下をテストする独立したモジュールで構成されています。

- レプリケーションの問題
- 証明書の有効性
- 認証局インフラストラクチャーの問題
- IdM および Active Directory の信頼の問題
- ファイルのパーミッションと所有権の正しい設定

2 つの出力形式

Healthcheck では、以下の出力が生成されます。これは、**output-type** オプションを使用して設定できます。

- **JSON**: マシンが判読できる出力 (デフォルト)
- **human**: 人間が判読できる出力

--output-file オプションで別の出力先ファイルを指定できます。

結果

Healthcheck の各モジュールは、次のいずれかの結果を返します。

SUCCESS
: 想定どおりに設定されています。

WARNING
: エラーではありませんが、注意または評価することを推奨します。

ERROR

想定どおりに設定されていません。

CRITICAL

想定どおりに設定されておらず、影響を受ける可能性が高くなっています。

101.2. ログローテーション

ログローテーションは新しいログファイルを毎日作成します。ファイルは日付別に編成されます。ログファイルは同じディレクトリーに保存されるため、日付に応じて特定のログファイルを選択できます。

ローテーションとは、設定されたログファイルの最大数を超えると、最新のファイルによって最も古いファイルが書き換えられ、ファイルの名前が変更されることを意味します。たとえば、ローテーションの数が 30 の場合、31 番目のログファイルが 1 番目の (最も古い) ログファイルを置き換えます。

ログローテーションは、膨大なログファイルを減らして整理するため、ログの分析に役立ちます。

101.3. IDM HEALTHCHECK でのログローテーションの設定

次の手順に従って、ログローテーションを設定します。

- **systemd** タイマー
- **crond** サービス

systemd タイマーは、Healthcheck ツールを定期的に実行して、ログを生成します。デフォルト値は毎日午前 4 時に設定されています。

crond サービスは、ログローテーションに使用されます。

デフォルトのログ名は **healthcheck.log** で、ローテーションされるログは **healthcheck.log-YYYYMMDD** 形式を使用します。

前提条件

- root でコマンドを実行できる。

手順

1. **systemd** タイマーを有効にします。

```
# systemctl enable ipa-healthcheck.timer
Created symlink /etc/systemd/system/multi-user.target.wants/ipa-healthcheck.timer ->
/usr/lib/systemd/system/ipa-healthcheck.timer.
```

2. **systemd** タイマーを起動します。

```
# systemctl start ipa-healthcheck.timer
```

3. **/etc/logrotate.d/ipahealthcheck** ファイルを開いて、保存すべきログの数を設定します。
デフォルトでは、ログローテーションは 30 日間に設定されます。

4. **/etc/logrotate.d/ipahealthcheck** ファイルで、ログへのパスを設定します。
デフォルトでは、ログは **/var/log/ipa/healthcheck/** ディレクトリーに保存されます。

5. **/etc/logrotate.d/ipahealthcheck** ファイルで、ログ生成の時間を設定します。
デフォルトでは、ログは毎日午前 4 時に作成されます。

6. ログローテーションを使用するには、**crond** サービスを有効にして実行します。

```
# systemctl enable crond
# systemctl start crond
```

ログの生成を開始するには、IPA healthcheck サービスを起動します。

```
# systemctl start ipa-healthcheck
```

結果を確認するには、**/var/log/ipa/healthcheck/** に移動して、ログが正しく作成されていることを確認します。

101.4. IDM HEALTHCHECK の設定の変更

Healthcheck の設定を変更するには、目的のコマンドラインオプションを **/etc/ipahealthcheck/ipahealthcheck.conf** ファイルに追加します。これは、たとえばログローテーションを設定し、自動分析に適した形式でログが新しいタイマーを設定したくない場合に便利です。

注記

この Healthcheck 機能は、RHEL 8.7 以降でのみ利用できます。

変更後、Healthcheck が作成するすべてのログは、新しい設定に従います。この設定は、Healthcheck の手動実行にも適用されます。

注記

Healthcheck を手動で実行する場合、設定ファイルの設定は、コマンドラインで指定したオプションよりも優先されます。たとえば、設定ファイルで **output_type** が **human** に設定されている場合、コマンドラインで **json** を指定しても効果はありません。設定ファイルで指定されていないコマンドラインオプションを使用すると、通常どおり適用されます。

関連情報

- IdM Healthcheck でのログローテーションの設定

101.5. 出力ログの形式を変更するための HEALTHCHECK の設定

設定済みのタイマーを使用して Healthcheck を設定するには、次の手順に従ってください。この例では、人間が判読できる形式でログを生成し、エラーだけでなく正常な結果も含めるように Healthcheck を設定します。

前提条件

- システムで RHEL 8.7 以降を実行している。
- **root** 権限がある。
- 以前にタイマーを使用してログローテーションを設定していた。

手順

1. テキストエディターで **/etc/ipahealthcheck/ipahealthcheck.conf** ファイルを開きます。
2. **[default]** セクションに、オプション **output_type=human** と **all=True** を追加します。
3. ファイルを保存してから閉じます。

検証

1. Healthcheck を手動で実行します。

```
# ipa-healthcheck
```

2. **/var/log/ipa/healthcheck/** に移動し、ログの形式が正しいことを確認します。

関連情報

- IdM Healthcheck でのログローテーションの設定

第102章 IDM HEALTHCHECK を使用したサービスの確認

Healthcheck ツールを使用して、Identity Management (IdM) サーバーによって使用されるサービスを監視できます。

詳細は IdM のヘルスチェック を参照してください。

前提条件

- Healthcheck ツールは、RHEL 8.1 以降でのみ利用できます。

102.1. サービスの HEALTHCHECK テスト

Healthcheck ツールには、実行されていない IdM サービスがないかどうかを確認するテストが含まれています。サービスが実行されていないと他のテストでエラーが発生する可能性があるため、このテストは重要です。したがって、まずすべてのサービスが実行されていることを確認してください。その後、他のすべてのテスト結果を確認します。

すべてのサービステストを表示するには、**--list-sources** オプションを指定して、**ipa-healthcheck** を実行します。

```
# ipa-healthcheck --list-sources
```

ipahealthcheck.meta.services ソースで、Healthcheck でテストしたサービスをすべて確認できます。

- certmonger
- dirsrv
- gssproxy
- httpd
- ipa_custodia
- ipa_dnskeysyncd
- ipa_otpd
- kadmin
- krb5kdc
- named
- pki_tomcatd
- sssd

注記

問題を検出するには、すべての IdM サーバーで上記のテストを実行します。

102.2. HEALTHCHECK を使用したサービスのスクリーニング

Healthcheck ツールを使用して、Identity Management (IdM) サーバー上で実行されているサービスのスタンドアロン手動テストを実行するには、次の手順に従います。

Healthcheck ツールには多くのテストが含まれており、その結果は次の方法で短くすることができます。

- 成功したテストをすべて除外する - **--failures-only**
- サービステストのみを含める - **--source=ipahealthcheck.meta.services**

手順

- サービスに関する警告、エラー、および重大な問題について Healthcheck を実行するには、次のコマンドを実行します。

```
# ipa-healthcheck --source=ipahealthcheck.meta.services --failures-only
```

テストに成功すると、空の括弧が表示されます。

```
[ ]
```

サービスのいずれかが失敗した場合は、以下のような結果になります。

```
{
  "source": "ipahealthcheck.meta.services",
  "check": "httpd",
  "result": "ERROR",
  "kw": {
    "status": false,
    "msg": "httpd: not running"
  }
}
```

関連情報

- **man ipa-healthcheck** を参照してください。

第103章 IDM HEALTHCHECK を使用した IDM および AD 信頼設定の検証

Healthcheck ツールを使用して、Identity Management (IdM) での IdM および Active Directory 信頼に関する問題を特定する方法について詳しく説明します。

前提条件

- Healthcheck ツールは、RHEL 8.1 以降でのみ利用できます。

103.1. IDM および AD 信頼の HEALTHCHECK のテスト

Healthcheck ツールには、Identity Management (IdM) および Active Directory (AD) 信頼のステータスをテストするための複数のテストが含まれています。

すべての信頼テストを表示するには、**--list-sources** オプションを指定して **ipa-healthcheck** を実行します。

```
# ipa-healthcheck --list-sources
```

すべてのテストは、**ipahealthcheck.ipa.trust** ソースの下にあります。

IPATrustAgentCheck

このテストでは、マシンが信頼エージェントとして設定されている場合に、SSSD 設定を確認します。**/etc/sssd/sssd.conf** 内の各ドメインで、**id_provider=ipa** は、**ipa_server_mode** が **True** であることを確認します。

IPATrustDomainsCheck

このテストでは、**sssctl domain-list** のドメインのリストを、IPA ドメインを除く **ipa trust-find** のドメインのリストと比較して、信頼ドメインが SSSD ドメインと一致するかどうかを確認します。

IPATrustCatalogCheck

このテストでは、AD ユーザー **Administrator@REALM** を解決します。これにより、**sssctl domain-status** の出力に、AD Global カタログと AD Domain Controller の値が追加されます。
各信頼ドメインに対して、SID + 500 (管理者) の ID でユーザーを検索し、**sssctl domain-status <domain> --active-server** の出力を確認して、ドメインがアクティブであることを確認します。

IPAsidgenpluginCheck

このテストでは、IPA 389-ds インスタンスで **sidgen** プラグインが有効になっていることを確認します。このテストでは、**cn=plugins,cn=config** の **IPA SIDGEN** プラグインおよび **ipa-sidgen-task** プラグインに、**nsslapd-pluginEnabled** オプションが含まれていることも検証します。

IPATrustAgentMemberCheck

このテストでは、現在のホストが **cn=adtrust agents,cn=sysaccounts,cn=etc,SUFFIX** のメンバーであることを確認します。

IPATrustControllerPrincipalCheck

このテストでは、現在のホストが **cn=adtrust agents,cn=sysaccounts,cn=etc,SUFFIX** のメンバーであることを確認します。

IPATrustControllerServiceCheck

このテストでは、現在のホストが ipactl で ADTRUST サービスを開始することを確認します。

IPATrustControllerConfCheck

このテストでは、**ldapi** が、**net conf** リストの出力で passdb バックエンドに対して有効になっていることを確認します。

IPATrustControllerGroupSIDCheck

このテストでは、admins グループの SID が 512 (Domain Admins RID) で終わることを確認します。

IPATrustPackageCheck

このテストでは、信頼コントローラーと AD 信頼が有効になっていない場合に、**trust-ad** パッケージがインストールされていることを確認します。

注記

問題を確認するには、すべての IdM サーバーで上記のテストを実行してください。

103.2. HEALTHCHECK ツールを使用した信頼のスクリーニング

Healthcheck ツールを使用して Identity Management (IdM) および Active Directory (AD) の信頼ヘルスチェックのスタンドアロン手動テストを実行するには、次の手順に従います。

Healthcheck ツールには多くのテストが含まれているため、以下の方法で結果を短くすることができます。

- 成功したテストをすべて除外する - **--failures-only**
- 信頼テストのみを含める - **--source=ipahealthcheck.ipa.trust**

手順

- 信頼における警告、エラー、および重大な問題について Healthcheck を実行するには、次のコマンドを実行します。

```
# ipa-healthcheck --source=ipahealthcheck.ipa.trust --failures-only
```

テストに成功すると、空の括弧が表示されます。

```
# ipa-healthcheck --source=ipahealthcheck.ipa.trust --failures-only
[]
```

関連情報

- **man ipa-healthcheck** を参照してください。

第104章 IDM HEALTHCHECK を使用した証明書の検証

Identity Management (IdM) の Healthcheck ツールを使用し、**certmonger** によって維持されている IPA 証明書の問題を特定する方法について詳しく説明します。

詳細は IdM のヘルスチェック を参照してください。

前提条件

- Healthcheck ツールは、RHEL 8.1 以降でのみ利用できます。

104.1. IDM 証明書の HEALTHCHECK テスト

Healthcheck ツールには、Identity Management (IdM) の certmonger が維持する証明書の状況を確認するさまざまなテストが含まれています。certmonger の詳細は、certmonger を使用してサービスの IdM 証明書の取得 を参照してください。

この一連のテストでは、有効期限、検証、信頼性、その他の問題を確認します。根本的な問題 1 つに対して、複数のエラーが発生する可能性があります。

すべての証明書テストを表示するには、**--list-sources** オプションを指定して **ipa-healthcheck** を実行します。

```
# ipa-healthcheck --list-sources
```

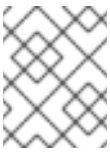
すべてのテストは、**ipahealthcheck.ipa.certs** ソースの下にあります。

IPACertmongerExpirationCheck

このテストでは、**certmonger** の有効期限を確認します。
証明書の有効期限が切れている場合は、エラーが報告されます。

証明書の有効期限が間近な場合は、警告が表示されます。デフォルトでは、このテストは、証明書の有効期限が 28 日以内のものを対象としています。

/etc/ipahealthcheck/ipahealthcheck.conf ファイルで日数を設定できます。ファイルを開いた後、デフォルトセクションにある **cert_expiration_days** オプションを変更します。

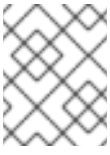


注記

certmonger は、証明書の有効期限に関する独自のビューをロードして維持します。このチェックでは、ディスク上の証明書は検証されません。

IPACertfileExpirationCheck

このテストでは、証明書ファイルまたは NSS データベースを開けないかどうかを確認します。このテストでは、有効期限も確認します。そのため、エラーまたは警告出力の **msg** 属性をよく読んでください。このメッセージは問題を特定するものです。



注記

このテストでは、ディスク上の証明書が確認されます。証明書がない、読み取りができないなどの問題が発生した場合は、別のエラーが出力される可能性があります。

IPACertNSSTrust

このテストでは、NSS データベースに保存されている証明書の信頼を比較します。NSS データベースで期待される、追跡される証明書では、期待される値と信頼が比較されます。一致しないとエラーが発生します。

IPANSSChainValidation

このテストでは、NSS 証明書の証明書チェーンを検証します。テストでは、**certutil -V -u V -e -d [dbdir] -n [nickname]** を実行します。

IPAOpenSSLChainValidation

このテストでは、OpenSSL 証明書の証明書チェーンを検証します。**NSSChain** 検証と比較するために、実行する OpenSSL コマンドを以下に示します。

```
openssl verify -verbose -show_chain -CAfile /etc/ipa/ca.crt [cert file]
```

IPARAAgent

このテストでは、ディスク上の証明書を、**uid=ipara,ou=People,o=ipaca** の LDAP の同等のレコードと比較します。

IPACertRevocation

このテストでは、certmonger を使用して、証明書が取り消されていないことを確認します。したがって、テストでは certmonger でのみメンテナンスされる証明書に接続している問題を見つけることができます。

IPACertmongerCA

このテストでは、certmonger の認証局 (CA) の設定を検証します。IdM は、CA を使用しない証明書を発行できません。
certmonger は、CA ヘルパーのセットを維持します。IdM には、IPA という名前の CA があります。IPA は、IdM を介して証明書を発行し、ホストまたはサービスの証明書に対して、ホストまたはユーザーのプリンシパルとして認証します。

また、CA サブシステム証明書を更新する **dogtag-ipa-ca-renew-agent** および **dogtag-ipa-ca-renew-agent-reuse** があります。

注記

問題を確認するには、すべての IdM サーバーで上記のテストを実行します。

104.2. HEALTHCHECK ツールを使用した証明書のスクリーニング

Healthcheck ツールを使用して Identity Management (IdM) 証明書ヘルスチェックのスタンドアロン手動テストを実行するには、次の手順に従います。

Healthcheck ツールには多くのテストが含まれているため、以下の方法で結果を短くすることができます。

- 成功したテストをすべて除外する - **--failures-only**
- 証明書テストのみを含める - **--source=ipahealthcheck.ipa.certs**

前提条件

- **root** ユーザーとして Healthcheck テストを実行する必要があります。

手順

- 証明書に関する警告、エラー、および重大な問題について Healthcheck を実行するには、次のコマンドを実行します。

```
# ipa-healthcheck --source=ipahealthcheck.ipa.certs --failures-only
```

テストに成功すると、空の括弧が表示されます。

```
[]
```

失敗したテストでは、以下の出力が表示されます。

```
{
  "source": "ipahealthcheck.ipa.certs",
  "check": "IPACertfileExpirationCheck",
  "result": "ERROR",
  "kw": {
    "key": 1234,
    "dbdir": "/path/to/nssdb",
    "error": [error],
    "msg": "Unable to open NSS database '/path/to/nssdb': [error]"
  }
}
```

上記の **IPACertfileExpirationCheck** テストは、NSS データベースを開くときに失敗しています。

関連情報

- **man ipa-healthcheck** を参照してください。

第105章 IDM HEALTHCHECK を使用したシステム証明書の検証

Healthcheck ツールを使用して Identity Management (IdM) のシステム証明書の問題を特定する方法について詳しく説明します。

詳細は IdM のヘルスチェック を参照してください。

前提条件

- Healthcheck ツールは、RHEL 8.1 以降でのみ利用できます。

105.1. システム証明書の HEALTHCHECK テスト

Healthcheck ツールには、システム (DogTag) 証明書を検証するさまざまなテストがあります。

すべてのテストを表示するには、**--list-sources** オプションを指定して **ipa-healthcheck** を実行します。

```
# ipa-healthcheck --list-sources
```

すべてのテストは、**ipahealthcheck.dogtag.ca** ソースの下にあります。

DogtagCertsConfigCheck

このテストでは、NSS データベース内の CA (認証局) 証明書を、**CS.cfg** に保存されている同じ値と比較します。一致しない場合、CA は起動に失敗します。
具体的には、以下を確認します。

- **ca.audit_signing.cert** の場合は **auditSigningCert cert-pki-ca**
- **ca.ocsp_signing.cert** の場合は **ocspSigningCert cert-pki-ca**
- **ca.signing.cert** の場合は **caSigningCert cert-pki-ca**
- **ca.subsystem.cert** の場合は **subsystemCert cert-pki-ca**
- **ca.sslserver.cert** の場合は **Server-Cert cert-pki-ca**

Key Recovery Authority (KRA) がインストールされている場合は、以下を確認します。

- **ca.connector.KRA.transportCert** の場合は **transportCert cert-pki-kra**

DogtagCertsConnectivityCheck

このテストでは、接続性を検証します。このテストは、以下の確認を行う **ipa cert-show 1** コマンドと同等です。

- Apache の PKI プロキシー設定
- IdM が CA を検出できること
- RA エージェントクライアント証明書
- 要求に対する CA 返信の正確性

このテストでは、**cert-show** を実行して CA から期待される結果 (証明書または not found) が返されることを確認する必要があるため、シリアル番号 #1 の証明書がチェックされます。

注記

問題を確認するには、すべての IdM サーバーで上記のテストを実行してください。

105.2. HEALTHCHECK を使用したシステム証明書のスクリーニング

Healthcheck ツールを使用して Identity Management (IdM) 証明書のスタンドアロン手動テストを実行するには、次の手順に従います。

Healthcheck ツールには多くのテストが含まれているため、Dogtag テスト (**--source=ipahealthcheck.dogtag.ca**) のみを含めることで結果を絞り込むことができます。

手順

- DogTag 証明書に限定して Healthcheck を実行するには、次のように入力します。

```
# ipa-healthcheck --source=ipahealthcheck.dogtag.ca
```

テストに成功すると、以下のようになります。

```
{
  "source: ipahealthcheck.dogtag.ca",
  "check: DogtagCertsConfigCheck",
  "result: SUCCESS",
  "uuid: 9b366200-9ec8-4bd9-bb5e-9a280c803a9c",
  "when: 20191008135826Z",
  "duration: 0.252280",
  "kw:" {
    "key": "Server-Cert cert-pki-ca",
    "configfile":  "/var/lib/pki/pki-tomcat/conf/ca/CS.cfg"
    }
}
```

テストに失敗すると、以下のようになります。

```
{
  "source: ipahealthcheck.dogtag.ca",
  "check: DogtagCertsConfigCheck",
  "result: CRITICAL",
  "uuid: 59d66200-1447-4b3b-be01-89810c803a98",
  "when: 20191008135912Z",
  "duration: 0.002022",
  "kw:" {
    "exception": "NSDB /etc/pki/pki-tomcat/alias not initialized",
    }
}
```

関連情報

- **man ipa-healthcheck** を参照してください。

第106章 IDM HEALTHCHECK を使用したディスク容量の確認

Healthcheck ツールを使用して、Identity Management サーバーの空きディスク容量を監視できます。

詳細は IdM のヘルスチェック を参照してください。

前提条件

- Healthcheck ツールは、RHEL 8.1 以降でのみ利用できます。

106.1. ディスク領域のヘルスチェックのテスト

Healthcheck ツールには、利用可能なディスク容量を確認するテストが含まれます。空きディスク容量が十分にないと、以下で問題が発生する可能性があります。

- ロギング
- 実行
- バックアップ

テストでは、以下のパスを確認します。

表106.1 テストされるパス

テストで確認されるパス	最小ディスク容量 (MB)
/var/lib/dirsrv/	1024
/var/lib/ipa/backup/	512
/var/log/	1024
var/log/audit/	512
/var/tmp/	512
/tmp/	512

テストのリストを表示するには、**--list-sources** オプションを指定して、**ipa-healthcheck** を実行します。

```
# ipa-healthcheck --list-sources
```

ファイルシステム容量の確認テストは、**ipahealthcheck.system.filesystemspace** ソースの下にあります。

FileSystemSpaceCheck

このテストでは、次の方法で使用可能なディスク容量を確認します。

- 最低限必要な生の空きバイト。

- パーセント - 空きディスクの最小容量は 20% にハードコーディングされています。

106.2. HEALTHCHECK ツールを使用したディスク容量のスクリーニング

Healthcheck ツールを使用して、Identity Management (IdM) サーバー上の利用可能なディスク容量のスタンドアロン手動テストを実行するには、次の手順に従います。

Healthcheck には多くのテストが含まれているため、次の方法で結果を絞り込むことができます。

- 成功したテストをすべて除外する - **--failures-only**
- 容量の確認テストのみを含める - **--source=ipahealthcheck.system.filesystemspace**

手順

- ディスク容量に関する警告、エラー、および重大な問題について Healthcheck を実行するには、次のコマンドを実行します。

```
# ipa-healthcheck --source=ipahealthcheck.system.filesystemspace --failures-only
```

テストに成功すると、空の括弧が表示されます。

```
[]
```

テストに失敗すると、たとえば、以下のような結果が表示されます。

```
{
  "source": "ipahealthcheck.system.filesystemspace",
  "check": "FileSystemSpaceCheck",
  "result": "ERROR",
  "kw": {
    "msg": "/var/lib/dirsrv: free space under threshold: 0 MiB < 1024 MiB",
    "store": "/var/lib/dirsrv",
    "free_space": 0,
    "threshold": 1024
  }
}
```

ここでは、**/var/lib/dirsrv** ディレクトリーの容量が不足しているためにテストに失敗したことが通知されています。

関連情報

- **man ipa-healthcheck** を参照してください。

第107章 HEALTHCHECK を使用した IDM 設定ファイルのパーミッションの確認

Healthcheck ツールを使用して Identity Management (IdM) 設定ファイルをテストする方法について詳しく説明します。

詳細は IdM のヘルスチェック を参照してください。

前提条件

- Healthcheck ツールは、RHEL 8.1 以降でのみ利用できます。

107.1. ファイルパーミッションの HEALTHCHECK テスト

Healthcheck ツールは、Identity Management (IdM) によりインストールまたは設定される重要なファイルの所有権とパーミッションをテストします。

テスト対象のファイルの所有権またはパーミッションが変更されていると、テスト時に **result** セクションに警告が返されます。これは必ずしも設定が機能しないことを意味しませんが、ファイルがデフォルト設定と異なることを意味します。

すべてのテストを表示するには、**--list-sources** オプションを指定して **ipa-healthcheck** を実行します。

```
# ipa-healthcheck --list-sources
```

ファイルパーミッションテストは、**ipahealthcheck.ipa.files** ソースの下にあります。

IPAFileNSSDBCheck
: このテストでは、389-ds NSS データベースと認証局 (CA) データベースを確認します。389-ds データベースは、**/etc/dirsrv/slapd-<dashed-REALM>** にあり、CA データベースは **/etc/pki/pki-tomcat/alias/** にあります。

IPAFileCheck
: このテストでは、以下のファイルを確認します。

  - **/var/lib/ipa/ra-agent.{key|pem}**
  - **/var/lib/ipa/certs/httpd.pem**
  - **/var/lib/ipa/private/httpd.key**
  - **/etc/httpd/alias/ipasession.key**
  - **/etc/dirsrv/ds.keytab**
  - **/etc/ipa/ca.crt**
  - **/etc/ipa/custodia/server.keys**
    PKINIT が有効になっている場合は、以下のファイルを確認します。
  - **/var/lib/ipa/certs/kdc.pem**
  - **/var/lib/ipa/private/kdc.key**
    DNS が設定されている場合は、以下のファイルを確認します。

- **/etc/named.keytab**
- **/etc/ipa/dnssec/ipa-dnskeysyncd.keytab**

TomcatFileCheck

このテストでは、CA が設定されている場合に、いくつかの tomcat 固有のファイルを確認します。

- **/etc/pki/pki-tomcat/password.conf**
- **/var/lib/pki/pki-tomcat/conf/ca/CS.cfg**
- **/etc/pki/pki-tomcat/server.xml**

注記

問題を確認するには、すべての IdM サーバーで上記のテストを実行します。

107.2. HEALTHCHECK を使用した設定ファイルのスクリーニング

Healthcheck ツールを使用して Identity Management (IdM) サーバーの設定ファイルのスタンドアロン手動テストを実行するには、次の手順に従います。

Healthcheck ツールには多くのテストが含まれています。結果を絞り込むには、以下を行います。

- 成功したテストをすべて除外する - **--failures-only**
- 所有者テストとパーミッションテストのみを含める - **--source=ipahealthcheck.ipa.files**

手順

1. IdM 設定ファイルの所有権とパーミッションについて Healthcheck テストを実行し、警告、エラー、重大な問題のみを表示するには、次のように入力します。

```
# ipa-healthcheck --source=ipahealthcheck.ipa.files --failures-only
```

テストに成功すると、空の括弧が表示されます。

```
# ipa-healthcheck --source=ipahealthcheck.ipa.files --failures-only
[]
```

テストに失敗すると、以下の **WARNING** のような結果が表示されます。

```
{
  "source": "ipahealthcheck.ipa.files",
  "check": "IPAFileNSSDBCheck",
  "result": "WARNING",
  "kw": {
    "key": "_etc_dirsrv_slapd-EXAMPLE-TEST_pkcs11.txt_mode",
    "path": "/etc/dirsrv/slapd-EXAMPLE-TEST/pkcs11.txt",
    "type": "mode",
    "expected": "0640",
    "got": "0666",
```

```
    "msg": "Permissions of /etc/dirsrv/slapd-EXAMPLE-TEST/pkcs11.txt are 0666 and should be 0640"
  }
}
```

関連情報

- **man ipa-healthcheck** を参照してください。

第108章 HEALTHCHECK を使用した IDM レプリケーションの確認

Healthcheck ツールを使用して、Identity Management (IdM) レプリケーションをテストできます。

詳細は IdM のヘルスチェック を参照してください。

前提条件

- Healthcheck ツールは、RHEL 8.1 以降でのみ利用できます。

108.1. レプリケーションの HEALTHCHECK テスト

Healthcheck ツールは、Identity Management (IdM) トポロジーの設定をテストして、レプリケーションの競合問題を検索します。

テストのリストを表示するには、**--list-sources** オプションを指定して、**ipa-healthcheck** を実行します。

```
# ipa-healthcheck --list-sources
```

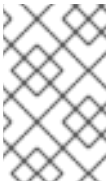
トポロジーのテストは、**ipahealthcheck.ipa.topology** ソースおよび **ipahealthcheck.ds.replication** ソースの下にあります。

IPATopologyDomainCheck

このテストでは、以下が検証されます。

- トポロジーが切断されておらず、すべてのサーバー間にレプリケーションパスがあるかどうか。
- サーバーに推奨される数以上のレプリカ合意がないかどうか。
  テストに失敗すると、接続エラーやレプリカ合意が多すぎるなどのエラーが返されます。

  テストに成功すると、設定済みのドメインが返されます。



注記

このテストでは、ドメインおよび ca 接尾辞の両方で **ipa topologysuffix-verify** コマンドを実行します (認証局がこのサーバーに設定されていることを前提としています)。

ReplicationConflictCheck

このテストでは、**(&(!(objectclass=nstombstone))(nsds5ReplConflict=*))** に一致する LDAP エントリーを検索します。

注記

問題を確認するには、すべての IdM サーバーで上記のテストを実行します。

LDAP レプリケーションの競合を解決する方法の詳細は、一般的なレプリケーションの問題解決 を参照してください。

108.2. HEALTHCHECK を使用したレプリケーションのスクリーニング

Healthcheck ツールを使用して Identity Management (IdM) レプリケーショントポロジーと設定のスタンドアロン手動テストを実行するには、次の手順に従います。

Healthcheck ツールには多くのテストが含まれているため、以下の方法で結果を短くすることができます。

- レプリケーションの競合テスト - **--source=ipahealthcheck.ds.replication**
- 正確なトポロジーテスト - **--source=ipahealthcheck.ipa.topology**

前提条件

- **root** ユーザーとして Healthcheck テストを実行する必要があります。

手順

- Healthcheck のレプリケーションの競合とトポロジーの確認を実行するには、次のコマンドを実行します。

```
# ipa-healthcheck --source=ipahealthcheck.ds.replication --
source=ipahealthcheck.ipa.topology
```

以下のような 4 つの結果が取得できます。

- SUCCESS - テストに成功

```
{
  "source": "ipahealthcheck.ipa.topology",
  "check": "IPATopologyDomainCheck",
  "result": "SUCCESS",
  "kw": {
    "suffix": "domain"
  }
}
```

- WARNING - テストには成功したが、問題の可能性あり
- ERROR - テストが失敗

```
{
  "source": "ipahealthcheck.ipa.topology",
  "check": "IPATopologyDomainCheck",
  "result": "ERROR",
  "uuid": d6ce3332-92da-423d-9818-e79f49ed321f
  "when": 20191007115449Z
  "duration": 0.005943
  "kw": {
    "msg": "topologysuffix-verify domain failed, server2 is not connected
(server2_139664377356472 in MainThread)"
  }
}
```

- CRITICAL - テストが失敗し、IdM サーバー機能に影響が及ぶ

関連情報

- **man ipa-healthcheck** を参照してください。

第109章 IDM HEALTHCHECK を使用した DNS レコードの確認

Healthcheck ツールを使用して、Identity Management (IdM) の DNS レコードの問題を特定できます。

前提条件

- Healthcheck ツールは、RHEL 8.2 以降でのみ利用できます。

109.1. DNS レコードのヘルスチェックテスト

Healthcheck ツールには、自動検出に必要な DNS レコードが解決可能であることを確認するテストが含まれます。

テストのリストを表示するには、**--list-sources** オプションを指定して、**ipa-healthcheck** を実行します。

```
# ipa-healthcheck --list-sources
```

DNS レコードの確認テストは、**ipahealthcheck.ipa.idns** ソースの下にあります。

IPADNSSystemRecordsCheck

このテストでは、**/etc/resolv.conf** ファイルで指定された最初のリゾルバーを使用して、**ipa dns-update-system-records --dry-run** コマンドで得られる DNS レコードを確認します。このレコードは IPA サーバーでテストされます。

109.2. HEALTHCHECK ツールを使用した DNS レコードのスクリーニング

Healthcheck ツールを使用して Identity Management (IdM) サーバー上で DNS レコードのスタンドアロン手動テストを実行するには、次の手順に従います。

Healthcheck ツールには多くのテストが含まれています。**--source ipahealthcheck.ipa.idns** オプションを追加して、DNS レコードテストだけを含めることで結果を絞り込むことができます。

前提条件

- **root** ユーザーとして Healthcheck テストを実行する必要があります。

手順

- DNS レコードの確認を実行するには、以下を入力します。

```
# ipa-healthcheck --source ipahealthcheck.ipa.idns
```

レコードが解決可能である場合には、テストの結果として **SUCCESS** が返されます。

```
{
    "source": "ipahealthcheck.ipa.idns",
    "check": "IPADNSSystemRecordsCheck",
    "result": "SUCCESS",
    "uuid": "eb7a3b68-f6b2-4631-af01-798cac0eb018",
    "when": "20200415143339Z",
    "duration": "0.210471",
    "kw": {
```

```
      "key": "_ldap._tcp.idm.example.com.:server1.idm.example.com."
    }
}
```

たとえば、レコードの数が想定数と一致しないなどの場合には、**WARNING** が返されます。

```
{
    "source": "ipahealthcheck.ipa.idns",
    "check": "IPADNSSystemRecordsCheck",
    "result": "WARNING",
    "uuid": "972b7782-1616-48e0-bd5c-49a80c257895",
    "when": "20200409100614Z",
    "duration": "0.203049",
    "kw": {
      "msg": "Got {count} ipa-ca A records, expected {expected}",
      "count": 2,
      "expected": 1
    }
}
```

関連情報

- **man ipa-healthcheck** を参照してください。

第110章 非表示レプリカの降格または昇格

レプリカのインストール後、レプリカの表示状態を設定できます。

非表示のレプリカの詳細は、非表示のレプリカモード を参照してください。

レプリカが CA 更新サーバーである場合は、このレプリカを非表示にする前に、サービスを別のレプリカに移動します。

詳細は IdM CA 更新サーバーの変更およびリセット を参照してください。

手順

- レプリカを非表示にするには、次のコマンドを実行します。

```
# ipa server-state replica.idm.example.com --state=hidden
```

次のコマンドを実行すれば、レプリカを表示できます

```
# ipa server-state replica.idm.example.com --state=enabled
```

トポロジー内のすべての非表示のレプリカのリストを表示するには、次のコマンドを実行します。

```
# ipa config-show
```

すべてのレプリカが有効になっている場合は、コマンドの出力に非表示のレプリカは記載されません。

第111章 IDENTITY MANAGEMENT のセキュリティー設定

Identity Management のセキュリティー関連機能について詳しく説明します。

111.1. IDENTITY MANAGEMENT がデフォルトのセキュリティー設定を適用する方法

デフォルトでは、RHEL 8 の Identity Management (IdM) はシステム全体の暗号化ポリシーを使用します。このポリシーの利点は、個々の IdM コンポーネントを手動で強化する必要がないことです。

重要

Red Hat は、システム全体の暗号化ポリシーを使用することが推奨されます。個々のセキュリティー設定を変更すると、IdM のコンポーネントが破損する可能性があります。たとえば、RHEL 8 の Java は、TLS 1.3 プロトコルに完全に対応していません。したがって、このプロトコルを使用すると、IdM でエラーが発生する可能性があります。

関連情報

- **crypto-policies(7)** man ページを参照してください。

111.2. IDENTITY MANAGEMENT の匿名 LDAP バインド

デフォルトでは、Identity Management (IdM) LDAP サーバーへの匿名バインドが有効になっています。匿名バインドは、特定の設定またはディレクトリー値を公開できます。ただし、**reald** などの一部のユーティリティーや古い RHEL クライアントでは、クライアントの登録時にドメイン設定を検出する匿名バインドを有効にする必要があります。

関連情報

- 匿名バインドの無効化

111.3. 匿名バインドの無効化

LDAP ツールを使用して **nsslapd-allow-anonymous-access** 属性をリセットすることで、Identity Management (IdM) 389 Directory Server インスタンスで匿名バインドを無効にできます。

以下は **nsslapd-allow-anonymous-access** 属性の有効な値です。

- **on**: すべての匿名バインドを許可します (デフォルト)。
- **Rootdse**: root の DSE 情報についてのみ匿名バインドを許可します。
- **off**: 匿名バインドを拒否します。

Red Hat では、属性を **off** に設定して匿名バインドを完全に拒否すると、外部クライアントによるサーバー設定のチェックもブロックするため、この設定は推奨していません。LDAP および web クライアントはドメインクライアントに限られるわけではないため、こうしたクライアントは匿名で接続を行ってルートの DSE ファイルを読み取り接続情報を取得します。

nsslapd-allow-anonymous-access 属性の値を **rootdse** に変更すると、ディレクトリーデータにアクセスせずにルート DSE およびサーバー設定へのアクセスを許可します。

警告

特定のクライアントは、匿名バインドを使用して IdM 設定を検出します。また、compat ツリーは、認証を使用していない従来のクライアントでは機能しない可能性があります。この手順は、クライアントが匿名バインドを必要としない場合にのみ実行します。

前提条件

- Directory Manager として認証して LDAP サーバーに書き込むことができる。
- **root** ユーザーとして認証して IdM サービスを再起動できる。

手順

1. **nsslapd-allow-anonymous-access** 属性を **rootdse** に変更します。

```
$ ldapmodify -x -D "cn=Directory Manager" -W -h server.example.com -p 389
Enter LDAP Password:
dn: cn=config
changetype: modify
replace: nsslapd-allow-anonymous-access
nsslapd-allow-anonymous-access: rootdse

modifying entry "cn=config"
```

2. 389 Directory Server インスタンスを再起動して、新しい設定を読み込みます。

```
# systemctl restart dirsrv.target
```

検証

- **nsslapd-allow-anonymous-access** 属性の値を表示します。

```
$ ldapsearch -x -D "cn=Directory Manager" -b cn=config -W -h server.example.com  -p 389
nsslapd-allow-anonymous-access | grep nsslapd-allow-anonymous-access
Enter LDAP Password:
# requesting: nsslapd-allow-anonymous-access
nsslapd-allow-anonymous-access: rootdse
```

関連情報

- Directory Server 11 ドキュメントの nsslapd-allow-anonymous-access
- Identity Management の匿名 LDAP バインド

第112章 IDM ドメインメンバーでの SAMBA の設定

Red Hat Identity Management (IdM) ドメインに参加しているホスト上で Samba をセットアップできます。IdM のユーザー、および可能であれば、信頼された Active Directory (AD) ドメインのユーザーは、Samba が提供する共有およびプリンターサービスにアクセスできます。

重要

IdM ドメインメンバーで Samba を使用する機能は、テクノロジープレビュー機能で、特定の制限が含まれています。たとえば、IdM 信頼コントローラーは Active Directory グローバルカタログサービスをサポートしておらず、分散コンピューティング環境/リモートプロシージャコール (DCE/RPC) プロトコルを使用した IdM グループの解決をサポートしていません。結果として、AD ユーザーは、他の IdM クライアントにログインしている場合、IdM クライアントでホストされている Samba 共有とプリンターにのみアクセスできます。Windows マシンにログインしている AD ユーザーは、IdM ドメインメンバーでホストされている Samba 共有にアクセスできません。

IdM ドメインメンバーに Samba をデプロイしているお客様は、ぜひ Red Hat にフィードバックをお寄せください。

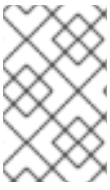
AD ドメインのユーザーが Samba によって提供される共有およびプリンターサービスにアクセスする必要がある場合は、AES 暗号化タイプが AD になっていることを確認してください。詳細は、GPO を使用した Active Directory での AES 暗号化タイプの有効化 を参照してください。

前提条件

- ホストは、クライアントとして IdM ドメインに参加している。
- IdM サーバーとクライアントの両方が RHEL 8.1 以降で実行されている必要がある。

112.1. SAMBA をドメインメンバーにインストールするための IDM ドメインの準備

IdM クライアントに Samba を設定する前に、IdM サーバーで **ipa-adtrust-install** ユーティリティーを使用して IdM ドメインを準備する必要があります。



注記

ipa-adtrust-install コマンドを自動的に実行するシステムは、AD 信頼コントローラーになります。ただし、**ipa-adtrust-install** は、IdM サーバーで 1 回のみ実行する必要があります。

前提条件

- IdM サーバーがインストールされている。
- パッケージをインストールし、IdM サービスを再起動するには、root 権限が必要です。

手順

1. 必要なパッケージをインストールします。

```
[root@ipaserver ~]# yum install ipa-server-trust-ad samba-client
```

2. IdM 管理ユーザーとして認証します。

```
[root@ipaserver ~]# kinit admin
```

3. **ipa-adtrust-install** ユーティリティーを実行します。

```
[root@ipaserver ~]# ipa-adtrust-install
```

統合 DNS サーバーとともに IdM がインストールされていると、DNS サービスレコードが自動的に作成されます。

IdM が統合 DNS サーバーなしで IdM をインストールすると、**ipa-adtrust-install** は、続行する前に DNS に手動で追加する必要があるサービスレコードのリストを出力します。

4. スクリプトにより、**/etc/samba/smb.conf** がすでに存在し、書き換えられることが求められます。

```
WARNING: The smb.conf already exists. Running ipa-adtrust-install will break your existing
Samba configuration.

Do you wish to continue? [no]: yes
```

5. このスクリプトは、従来の Linux クライアントが信頼できるユーザーと連携できるようにする互換性プラグインである **slapi-nis** プラグインを設定するように求めるプロンプトを表示します。

```
Do you want to enable support for trusted domains in Schema Compatibility plugin?
This will allow clients older than SSSD 1.9 and non-Linux clients to work with trusted users.

Enable trusted domains support in slapi-nis? [no]: yes
```

6. プロンプトが表示されたら、IdM ドメインの NetBIOS 名を入力するか、**Enter** を押して提案された名前を使用します。

```
Trust is configured but no NetBIOS domain name found, setting it now.
Enter the NetBIOS name for the IPA domain.
Only up to 15 uppercase ASCII letters, digits and dashes are allowed.
Example: EXAMPLE.

NetBIOS domain name [IDM]:
```

7. SID 生成タスクを実行して、既存ユーザーに SID を作成するように求められます。

```
Do you want to run the ipa-sidgen task? [no]: yes
```

これはリソースを集中的に使用するタスクであるため、ユーザー数が多い場合は別のタイミングで実行できます。

8. **(必要に応じて)** デフォルトでは、Windows Server 2008 以降での動的 RPC ポートの範囲は **49152-65535** として定義されます。ご使用の環境に異なる動的 RPC ポート範囲を定義する必要がある場合は、Samba が異なるポートを使用するように設定し、ファイアウォール設定でそのポートを開くように設定します。以下の例では、ポート範囲を **55000-65000** に設定します。

```
[root@ipaserver ~]# net conf setparm global 'rpc server dynamic port range' 55000-
```

```
65000
[root@ipaserver ~]# firewall-cmd --add-port=55000-65000/tcp
[root@ipaserver ~]# firewall-cmd --runtime-to-permanent
```

9. **ipa** サービスを再起動します。

```
[root@ipaserver ~]# ipactl restart
```

10. **smbclient** ユーティリティーを使用して、Samba が IdM からの Kerberos 認証に応答することを確認します。

```
[root@ipaserver ~]# smbclient -L server.idm.example.com -U user_name --use-kerberos=required
lp_load_ex: changing to config backend registry
    Sharename       Type      Comment
    ---------       ----      -------
    IPC$            IPC       IPC Service (Samba 4.15.2)
...
```

112.2. IDM クライアントでの SAMBA サーバーのインストールおよび設定

IdM ドメインに登録されたクライアントに Samba をインストールして設定できます。

前提条件

- IdM サーバーとクライアントの両方が RHEL 8.1 以降で実行されている必要がある。
- IdM ドメインは、ドメインメンバーに Samba をインストールするための IdM ドメインの準備 の説明に従って準備されます。
- IdM に AD で設定された信頼がある場合は、Kerberos の AES 暗号化タイプを有効にします。たとえば、グループポリシーオブジェクト (GPO) を使用して、AES 暗号化の種類を有効にします。詳細は、GPO を使用した Active Directory での AES 暗号化の有効化 を参照してください。

手順

1. **ipa-client-samba** パッケージをインストールします。

```
[root@idm_client]# yum install ipa-client-samba
```

2. **ipa-client-samba** ユーティリティーを使用して、クライアントを準備し、初期 Samba 設定を作成します。

```
[root@idm_client]# ipa-client-samba
Searching for IPA server...
IPA server: DNS discovery
Chosen IPA master: idm_server.idm.example.com
SMB principal to be created: cifs/idm_client.idm.example.com@IDM.EXAMPLE.COM
NetBIOS name to be used: IDM_CLIENT
Discovered domains to use:

 Domain name: idm.example.com
NetBIOS name: IDM
```

```
        SID: S-1-5-21-525930803-952335037-206501584
   ID range: 212000000 - 212199999

 Domain name: ad.example.com
NetBIOS name: AD
        SID: None
   ID range: 1918400000 - 1918599999

Continue to configure the system with these values? [no]: yes
Samba domain member is configured. Please check configuration at /etc/samba/smb.conf
and start smb and winbind services
```

3. デフォルトでは、**ipa-client-samba** は、ユーザーが接続したときにそのユーザーのホームディレクトリーを動的に共有するために、**/etc/samba/smb.conf** ファイルに **[homes]** セクションが自動的に追加されます。ユーザーがこのサーバーにホームディレクトリーがない場合、または共有したくない場合は、**/etc/samba/smb.conf** から次の行を削除します。

```
[homes]
   read only = no
```

4. ディレクトリーとプリンターを共有します。詳細は、次のセクションを参照してください。
   - POSIX ACL を使用した Samba ファイル共有の設定
   - Windows ACL で共有の設定
   - プリントサーバーとしての Samba の設定
5. ローカルファイアウォールで Samba クライアントに必要なポートを開きます。

```
[root@idm_client]# firewall-cmd --permanent --add-service=samba-client
[root@idm_client]# firewall-cmd --reload
```

6. **smb** サービスおよび **winbind** サービスを有効にして開始します。

```
[root@idm_client]# systemctl enable --now smb winbind
```

検証手順

samba-client パッケージがインストールされている別の IdM ドメインメンバーで次の検証手順を実行します。

- Kerberos 認証を使用して、Samba サーバー上の共有をリスト表示します。

```
$ smbclient -L idm_client.idm.example.com -U user_name --use-kerberos=required
lp_load_ex: changing to config backend registry

   Sharename       Type      Comment
   ---------       ----      -------
   example         Disk
   IPC$            IPC       IPC Service (Samba 4.15.2)
...
```

関連情報

- **ipa-client-samba(1)** の man ページ

112.3. IDM が新しいドメインを信頼する場合は、ID マッピング設定を手動で追加

Samba では、ユーザーがリソースにアクセスする各ドメインの ID マッピング設定が必要です。IdM クライアントで実行している既存の Samba サーバーでは、管理者が Active Directory (AD) ドメインに新しい信頼を追加した後、ID マッピング設定を手動で追加する必要があります。

前提条件

- IdM クライアントで Samba を設定している。その後、IdM に新しい信頼が追加されている。
- Kerberos の暗号化タイプ DES および RC4 は、信頼できる AD ドメインで無効にしている。セキュリティー上の理由から、RHEL 8 はこのような弱い暗号化タイプに対応していません。

手順

1. ホストのキータブを使用して認証します。

```
[root@idm_client]# kinit -k
```

2. **ipa idrange-find** コマンドを使用して、新しいドメインのベース ID と ID 範囲のサイズの両方を表示します。たとえば、次のコマンドは **ad.example.com** ドメインの値を表示します。

```
[root@idm_client]# ipa idrange-find --name="AD.EXAMPLE.COM_id_range" --raw
---------------
1 range matched
---------------
  cn: AD.EXAMPLE.COM_id_range
  ipabaseid: 1918400000
  ipaidrangesize: 200000
  ipabaserid: 0
  ipanttrusteddomainsid: S-1-5-21-968346183-862388825-1738313271
  iparangetype: ipa-ad-trust
----------------------------
Number of entries returned 1
----------------------------
```

   次の手順で、**ipabaseid** 属性および **ipaidrangesize** 属性の値が必要です。

3. 使用可能な最高の ID を計算するには、次の式を使用します。

```
maximum_range = ipabaseid + ipaidrangesize - 1
```

   前の手順の値を使用すると、**ad.example.com** ドメインで使用可能な最大 ID は **1918599999** (1918400000 + 200000 - 1) です。

4. **/etc/samba/smb.conf** ファイルを編集し、ドメインの ID マッピング設定を **[global]** セクションに追加します。

```
idmap config AD : range = 1918400000 - 1918599999
idmap config AD : backend = sss
```

ipabaseid 属性の値を最小値として指定し、前の手順で計算された値を範囲の最大値として指定します。

5. **smb** サービスおよび **winbind** サービスを再起動します。

```
[root@idm_client]# systemctl restart smb winbind
```

検証手順

- Kerberos 認証を使用して、Samba サーバー上の共有をリスト表示します。

```
$ smbclient -L idm_client.idm.example.com -U user_name --use-kerberos=required
lp_load_ex: changing to config backend registry

   Sharename       Type      Comment
   ---------       ----      -------
   example         Disk
   IPC$            IPC       IPC Service (Samba 4.15.2)
...
```

112.4. 関連情報

- Installing an Identity Management client

第113章 外部 ID プロバイダーを使用した IDM に対する認証

ユーザーを OAuth 2 デバイス承認フローをサポートする外部アイデンティティープロバイダー (IdP) に関連付けることができます。これらのユーザーが、RHEL 8.7 以降で利用可能な SSSD バージョンで認証すると、外部 IdP で認証と承認を実行した後に Kerberos チケットを使用した RHEL Identity Management (IdM) Single Sign-On 機能を受け取ります。

主な変更には以下のものがあります。

- **ipa idp-*** コマンドによる外部 IdP への参照の追加、変更、および削除
- **ipa user-mod --user-auth-type=idp** コマンドを使用したユーザーの IdP 認証の有効化

113.1. IDM を外部 IDP に接続する利点

管理者は、クラウドサービスプロバイダーなどの外部 ID ソースに保存されているユーザーが、Identity Management (IdM) 環境に追加された RHEL システムにアクセスできるようにすることができます。そのために、これらのユーザーの Kerberos チケットを発行する認証および認可プロセスをその外部エンティティーに委任できます。

この機能を使用して IdM の機能を拡張し、外部 ID プロバイダー (IdP) に保存されているユーザーが IdM によって管理される Linux システムにアクセスできるようにすることができます。

113.2. IDM が外部 IDP を介してログインを組み込む方法

SSSD 2.7.0 には、**idp** Kerberos 事前認証方法を実装する **sssd-idp** パッケージが含まれています。この認証方法は、OAuth 2.0 Device Authorization Grant フローに従って、認可の判断を外部 IdP に委任します。

1. IdM クライアントユーザーは、コマンドラインで **kinit** ユーティリティーを使用して Kerberos TGT の取得を試行するなどして、OAuth 2.0 デバイス認可付与フローを開始します。
2. 特別なコードと Web サイトのリンクが認可サーバーから IdM KDC バックエンドに送信されます。
3. IdM クライアントは、リンクとコードをユーザーに表示します。この例では、IdM クライアントはコマンドラインにリンクとコードを出力します。
4. ユーザーは、別のホストや携帯電話などのブラウザーで Web サイトのリンクを開きます。
   a. ユーザーは特別なコードを入力します。
   b. 必要に応じて、ユーザーは OAuth 2.0 ベースの IdP にログインします。
   c. ユーザーは、クライアントによる情報へのアクセスを許可するよう求められます。
5. ユーザーは、元のデバイスのプロンプトでアクセスを確認します。この例では、ユーザーはコマンドラインで **Enter** キーを押します。
6. IdM KDC バックエンドは、ユーザー情報にアクセスするために OAuth 2.0 認可サーバーをポーリングします。

サポート対象:

- Pluggable Authentication Module (PAM) ライブラリーの呼び出しを可能にする **keyboard-interactive** 認証方法を有効にして、SSH 経由でリモートからログインする場合。
- **logind** サービスを介してコンソールでローカルにログインする場合。
- **kinit** ユーティリティーを使用して Kerberos TGT (Ticket-granting ticket) を取得する場合。

現在のサポート対象外:

- IdM WebUI に直接ログインする場合。IdM WebUI にログインするには、最初に Kerberos チケットを取得する必要があります。
- Cockpit WebUI に直接ログインする場合。Cockpit WebUI にログインするには、最初に Kerberos チケットを取得する必要があります。

関連情報

- Authentication against external Identity Providers
- RFC 8628: OAuth 2.0 Device Authorization Grant

113.3. 外部 ID プロバイダーへの参照の作成

外部 ID プロバイダー (IdP) を Identity Management (IdM) 環境に接続するには、IdM で IdP 参照を作成します。Keycloak テンプレートに基づいて IdP への **my-keycloak-idp** という参照を作成するには、この手順を実行します。その他の参照テンプレートについては、IdM でのさまざまな外部 IdP への参照例を 参照してください。

前提条件

- IdM を OAuth アプリケーションとして外部 IdP に登録し、クライアント ID を取得している。
- IdM 管理者アカウントとして認証可能である。
- IdM サーバーで RHEL 8.7 以降を使用している。
- IdM サーバーで SSSD 2.7.0 以降を使用している。

手順

1. IdM サーバーで IdM 管理者として認証します。

```
[root@server ~]# kinit admin
```

2. Keycloak テンプレートに基づいて、IdP への **my-keycloak-idp** という参照を作成します。ここで、**--base-url** オプションは 、**server-name.$DOMAIN:$PORT/prefix の形式で Keycloak サーバーへの URL を指定します。**

```
[root@server ~]# ipa idp-add my-keycloak-idp \
                 --provider keycloak --organization main \
                 --base-url keycloak.idm.example.com:8443/auth \
                 --client-id id13778
----------------------------------------------
Added Identity Provider reference "my-keycloak-idp"
----------------------------------------------
```

```
  Identity Provider reference name: my-keycloak-idp
  Authorization URI:
https://keycloak.idm.example.com:8443/auth/realms/main/protocol/openid-connect/auth
  Device authorization URI:
https://keycloak.idm.example.com:8443/auth/realms/main/protocol/openid-
connect/auth/device
  Token URI: https://keycloak.idm.example.com:8443/auth/realms/main/protocol/openid-
connect/token
  User info URI: https://keycloak.idm.example.com:8443/auth/realms/main/protocol/openid-
connect/userinfo
  Client identifier: ipa_oidc_client
  Scope: openid email
  External IdP user identifier attribute: email
```

検証

- **ipa idp-show** コマンドの出力に、作成した IdP 参照が表示されていることを確認します。

```
[root@server ~]# ipa idp-show my-keycloak-idp
```

関連情報

- IdM におけるさまざまな外部 IdP への参照例
- IdM で外部アイデンティティープロバイダーを管理するための ipa idp-* コマンドのオプション
- ipa idp-* コマンドの --provider オプション
- **ipa ヘルプ idp-add**

113.4. IDM におけるさまざまな外部 IDP への参照例

次の表は、IdM 内のさまざまな IdP への参照を作成するための **ipa idp-add** コマンドの例を示しています。

アイデンティティープロバイダー	重要なオプション	コマンド例
Microsoft Identity Platform、Azure AD	**--provider microsoft --organization**	# ipa idp-add **my-azure-idp** \ **--provider microsoft** \ --organization **main** \ --client-id **<azure_client_id>**
Google	**--provider google**	# ipa idp-add **my-google-idp** \ **--provider google** \ --client-id **<google_client_id>**

アイデンティティープロバイダー	重要なオプション	コマンド例
GitHub	**--provider github**	# ipa idp-add my-github-idp \ **--provider github** \ --client-id **<github_client_id>**
Keycloak、Red Hat Single Sign-On	**--provider keycloak** **--organization** **--base-url**	# ipa idp-add my-keycloak-idp \ **--provider keycloak** \ --organization main \ --base-url **keycloak.idm.example.com:8443/auth** \ --client-id **<keycloak_client_id>** **注記** Keycloak 17 以降の Quarkus バージョンでは、URI の **/auth/** 部分が削除されています。デプロイメントで Keycloak の非 Quarkus ディストリビューションを使用する場合は、**--base-url** オプションに **/auth/** を含めます。
Okta	**--provider okta**	# ipa idp-add my-okta-idp \ **--provider okta** --base-url **dev-12345.okta.com** \ --client-id **<okta_client_id>**

関連情報

- 外部 ID プロバイダーへの参照の作成
- IdM で外部アイデンティティープロバイダーを管理するための ipa idp-* コマンドのオプション
- ipa idp-* コマンドの --provider オプション

113.5. IDM で外部アイデンティティープロバイダーを管理するための IPA IDP-* コマンドのオプション

これらの例は、さまざまな IdP テンプレートに基づいて外部 IdP への参照を設定する方法を示しています。次のオプションを使用して設定を指定します。

--provider

既知の ID プロバイダーのいずれかの定義済みテンプレート

--client-id

アプリケーション登録時に IdP によって発行された OAuth 2.0 クライアント識別子。アプリケーションの登録手順は IdP ごとに異なるため、詳細については各 IdP のドキュメントを参照してください。外部 IdP が Red Hat Single Sign-On (SSO) の場合は、OpenID Connect クライアントの作成 を参照してください。

--base-url

Keycloak と Okta で必要な IdP テンプレートのベース URL

organization

Microsoft Azure で必要な IdP からのドメインまたは組織 ID

--secret

(オプション) 機密 OAuth 2.0 クライアントからのシークレットを要求するように、外部 IdP を設定した場合は、このオプションを使用します。IdP 参照を作成するときにこのオプションを使用すると、シークレットを対話的に求めるプロンプトが表示されます。クライアントシークレットをパスワードとして保護します。

注記

RHEL 8.7 の SSSD は、クライアントシークレットを使用しない非機密 OAuth 2.0 クライアントのみをサポートします。機密クライアントからのクライアントシークレットを必要とする外部 IdP を使用する場合は、RHEL 8.8 以降で SSSD を使用する必要があります。

関連情報

- 外部 ID プロバイダーへの参照の作成
- IdM におけるさまざまな外部 IdP への参照例
- ipa idp-* コマンドの --provider オプション

113.6. 外部 IDP への参照の管理

外部 ID プロバイダー (IdP) への参照を作成したら、その参照を検索、表示、変更、および削除できます。この例では、**keycloak-server1** という名前の外部 IdP への参照を管理する方法を示します。

前提条件

- IdM 管理者アカウントとして認証可能である。
- IdM サーバーで RHEL 8.7 以降を使用している。
- IdM サーバーで SSSD 2.7.0 以降を使用している。
- IdM で外部 IdP への参照を作成しました。外部 ID プロバイダーへの参照の作成 を参照してください。

手順

1. IdM サーバーで IdM 管理者として認証します。

```
[root@server ~]# kinit admin
```

2. IdP 参照を管理します。

   - エントリーに文字列 **keycloak** が含まれる IdP 参照を見つけるには、以下を実行します。

     ```
     [root@server ~]# ipa idp-find keycloak
     ```

   - **my-keycloak-idp** という名前の IdP 参照を表示するには、以下を実行します。

     ```
     [root@server ~]# ipa idp-show my-keycloak-idp
     ```

   - IdP 参照を変更するには、**ipa idp-mod** コマンドを使用します。たとえば、**my-keycloak-idp** という名前の IdP 参照のシークレットを変更するには、**--secret** オプションを指定してシークレットの入力を求めます。

     ```
     [root@server ~]# ipa idp-mod my-keycloak-idp --secret
     ```

   - **my-keycloak-idp** という名前の IdP 参照を削除するには、以下を実行します。

     ```
     [root@server ~]# ipa idp-del my-keycloak-idp
     ```

113.7. 外部 IDP 経由での IDM ユーザーの認証を有効にする方法

外部 ID プロバイダー (IdP) 経由で IdM ユーザーを認証できるようにするには、以前に作成した外部 IdP 参照をユーザーアカウントに関連付けます。この例では、外部 IdP 参照 **keycloak-server1** をユーザー **idm-user-with-external-idp** に関連付けます。

前提条件

- IdM クライアントと IdM サーバーで RHEL 8.7 以降を使用している。
- IdM クライアントと IdM サーバーで SSSD 2.7.0 以降を使用している。
- IdM で外部 IdP への参照を作成しました。外部 ID プロバイダーへの参照の作成 を参照してください。

手順

- IdM ユーザーエントリーを変更して、IdP 参照をユーザーアカウントに関連付けます。

```
[root@server ~]# ipa user-mod idm-user-with-external-idp \
                 --idp my-keycloak-idp \
                 --idp-user-id idm-user-with-external-idp@idm.example.com \
                 --user-auth-type=idp
--------------------------------
Modified user "idm-user-with-external-idp"
--------------------------------
  User login: idm-user-with-external-idp
  First name: Test
  Last name: User1
  Home directory: /home/idm-user-with-external-idp
  Login shell: /bin/sh
  Principal name: idm-user-with-external-idp@idm.example.com
  Principal alias: idm-user-with-external-idp@idm.example.com
  Email address: idm-user-with-external-idp@idm.example.com
```

```
  UID: 35000003
  GID: 35000003
  User authentication types: idp
  External IdP configuration: keycloak
  External IdP user identifier: idm-user-with-external-idp@idm.example.com
  Account disabled: False
  Password: False
  Member of groups: ipausers
  Kerberos keys available: False
```

検証

- そのユーザーの **ipa user-show** コマンド出力に IdP への参照が表示されることを確認します。

```
[root@server ~]# ipa user-show idm-user-with-external-idp
  User login: idm-user-with-external-idp
  First name: Test
  Last name: User1
  Home directory: /home/idm-user-with-external-idp
  Login shell: /bin/sh
  Principal name: idm-user-with-external-idp@idm.example.com
  Principal alias: idm-user-with-external-idp@idm.example.com
  Email address: idm-user-with-external-idp@idm.example.com
  ID: 35000003
  GID: 35000003
  User authentication types: idp
  External IdP configuration: keycloak
  External IdP user identifier: idm-user-with-external-idp@idm.example.com
  Account disabled: False
  Password: False
  Member of groups: ipausers
  Kerberos keys available: False
```

113.8. 外部 IDP ユーザーとして IDM チケット許可チケットを取得する

アイデンティティー管理 (IdM) ユーザーの認証を外部アイデンティティープロバイダー (IdP) に委任している場合、IdM ユーザーは外部 IdP に対して認証することで Kerberos チケット保証チケット (TGT) を要求できます。

次の操作を行うには、この手順を完了してください。

1. 匿名の Kerberos チケットを取得してローカルに保存します。
2. **-T** オプションを指定した **kinit** を使用して **idm-user-with-external-idp** ユーザーの TGT を要求し、Flexible Authentication via Secure Tunneling (FAST) チャネルを有効にして、Kerberos クライアントと Kerberos Distribution Center (KDC) 間の安全な接続を提供します。

前提条件

- IdM クライアントと IdM サーバーで RHEL 8.7 以降を使用している。
- IdM クライアントと IdM サーバーで SSSD 2.7.0 以降を使用している。
- IdM で外部 IdP への参照を作成しました。外部 ID プロバイダーへの参照の作成 を参照してください。

- 外部 IdP 参照をユーザーアカウントに関連付けている。外部 IdP 経由での IdM ユーザーの認証を有効にする方法 を参照してください。
- 最初にログインしたユーザーには、ローカルファイルシステム内のディレクトリーに対する書き込み権限があります。

手順

1. 匿名 PKINIT を使用して Kerberos チケットを取得し、それを **./fast.ccache** という名前のファイルに保存します。

```
$ kinit -n -c ./fast.ccache
```

2. [オプション] 取得したチケットを表示します。

```
$ klist -c fast.ccache
Ticket cache: FILE:fast.ccache
Default principal: WELLKNOWN/ANONYMOUS@WELLKNOWN:ANONYMOUS

Valid starting       Expires              Service principal
03/03/2024 13:36:37  03/04/2024 13:14:28
krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
```

3. **-T** オプションを使用してユーザーとして認証を開始し、FAST 通信チャネルを有効にします。

```
[root@client ~]# kinit -T ./fast.ccache idm-user-with-external-idp
Authenticate at https://oauth2.idp.com:8443/auth/realms/master/device?user_code=YHMQ-XKTL and press ENTER.:
```

4. ブラウザーで、コマンド出力に提供される Web サイトでユーザーとして認証します。
5. コマンドラインで **Enter** キーを押して、認証プロセスを終了します。

検証

- Kerberos チケット情報を表示し、**config: pa_type** の行が外部 IdP による事前認証の **152** を示していることを確認します。

```
[root@client ~]# klist -C
Ticket cache: KCM:0:58420
Default principal: idm-user-with-external-idp@IDM.EXAMPLE.COM

Valid starting     Expires            Service principal
05/09/22 07:48:23  05/10/22 07:03:07  krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
config: fast_avail(krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM) = yes
08/17/2022 20:22:45  08/18/2022 20:22:43
krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
config: pa_type(krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM) = 152
```

113.9. 外部 IDP ユーザーとして SSH 経由で IDM クライアントにログインする

外部 ID プロバイダー (IdP) ユーザーとして SSH 経由で IdM クライアントにログインするには、コマンドラインでログインプロセスを開始します。プロンプトが表示されたら、IdP に関連付けられた Web サイトで認証プロセスを実行し、Identity Management (IdM) クライアントでプロセスを終了します。

前提条件

- IdM クライアントと IdM サーバーで RHEL 8.7 以降を使用している。
- IdM クライアントと IdM サーバーで SSSD 2.7.0 以降を使用している。
- IdM で外部 IdP への参照を作成しました。外部 ID プロバイダーへの参照の作成 を参照してください。
- 外部 IdP 参照をユーザーアカウントに関連付けている。外部 IdP 経由での IdM ユーザーの認証を有効にする方法 を参照してください。

手順

1. SSH 経由で IdM クライアントへのログインを試みます。

```
[user@client ~]$ ssh idm-user-with-external-idp@client.idm.example.com
(idm-user-with-external-idp@client.idm.example.com) Authenticate at
https://oauth2.idp.com:8443/auth/realms/main/device?user_code=XYFL-ROYR and press
ENTER.
```

2. ブラウザーで、コマンド出力に提供される Web サイトでユーザーとして認証します。
3. コマンドラインで **Enter** キーを押して、認証プロセスを終了します。

検証

- Kerberos チケット情報を表示し、**config: pa_type** の行が外部 IdP による事前認証の **152** を示していることを確認します。

```
[idm-user-with-external-idp@client ~]$ klist -C
Ticket cache: KCM:0:58420
Default principal: idm-user-with-external-idp@IDM.EXAMPLE.COM

Valid starting     Expires            Service principal
05/09/22 07:48:23  05/10/22 07:03:07  krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
config: fast_avail(krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM) = yes
08/17/2022 20:22:45  08/18/2022 20:22:43
krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
config: pa_type(krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM) = 152
```

113.10. IPA IDP-* コマンドの --PROVIDER オプション

次の ID プロバイダー (IdP) は、OAuth 2.0 デバイス認可グラントフローをサポートしています。

- Azure AD を含む Microsoft Identity Platform
- Google
- GitHub

- Red Hat Single Sign-On (SSO) を含む Keycloak
- Okta

ipa idp-add コマンドを使用してこれらの外部 IdP のいずれか 1 つへの参照を作成する場合、**--provider** オプションで IdP タイプを指定できます。これは、以下で説明する追加オプションに拡張されます。

--provider=microsoft

Microsoft Azure IdP では、**--organization** オプションで **ipa idp-add** に指定できる Azure テナント ID に基づくパラメーター化が可能です。live.com IdP のサポートが必要な場合は、**--organization common** オプションを指定します。
--provider=microsoft を選択すると、次のオプションを使用するように拡張されます。**--organization** オプションの値は、表内の文字列 **${ipaidporg}** を置き換えます。

オプション	値
--auth-uri=URI	**https://login.microsoftonline.com/${ipaidporg}/oauth2/v2.0/authorize**
--dev-auth-uri=URI	**https://login.microsoftonline.com/${ipaidporg}/oauth2/v2.0/devicecode**
--token-uri=URI	**https://login.microsoftonline.com/${ipaidporg}/oauth2/v2.0/token**
--userinfo-uri=URI	**https://graph.microsoft.com/oidc/userinfo**
--keys-uri=URI	**https://login.microsoftonline.com/common/discovery/v2.0/keys**
--scope=STR	**openid email**
--idp-user-id=STR	**email**

--provider=google

--provider=google を選択すると、次のオプションを使用するように拡張されます。

オプション	値
--auth-uri=URI	**https://accounts.google.com/o/oauth2/auth**
--dev-auth-uri=URI	**https://oauth2.googleapis.com/device/code**
--token-uri=URI	**https://oauth2.googleapis.com/token**
--userinfo-uri=URI	**https://openidconnect.googleapis.com/v1/userinfo**
--keys-uri=URI	**https://www.googleapis.com/oauth2/v3/certs**

オプション	値
--scope=STR	**openid email**
--idp-user-id=STR	**email**

--provider=github

--provider=github を選択すると、次のオプションを使用するように拡張されます。

オプション	値
--auth-uri=URI	**https://github.com/login/oauth/authorize**
--dev-auth-uri=URI	**https://github.com/login/device/code**
--token-uri=URI	**https://github.com/login/oauth/access_token**
--userinfo-uri=URI	**https://openidconnect.googleapis.com/v1/userinfo**
--keys-uri=URI	**https://api.github.com/user**
--scope=STR	**user**
--idp-user-id=STR	**login**

--provider=keycloak

Keycloak を使用すると、複数のレルムまたは組織を定義できます。多くの場合、これはカスタムデプロイメントの一部であるため、ベース URL とレルム ID の両方が必要です。これらは、**ipa idp-add** コマンドの **--base-url** および **--organization** オプションで指定できます。

```
[root@client ~]# ipa idp-add MySSO --provider keycloak \
   --org main --base-url keycloak.domain.com:8443/auth \
   --client-id <your-client-id>
```

--provider=keycloak を選択すると、次のオプションを使用するように拡張されます。**--base-url** オプションで指定した値は、表内の文字列 **${ipaidpbaseurl}** を置き換え、指定した値は **--organization `option replaces the string `${ipaidporg}** となります。

オプション	値
--auth-uri=URI	**https://${ipaidpbaseurl}/realms/${ipaidporg}/protocol/openid-connect/auth**
--dev-auth-uri=URI	**https://${ipaidpbaseurl}/realms/${ipaidporg}/protocol/openid-connect/auth/device**

オプション	値
--token-uri=URI	**https://${ipaidpbaseurl}/realms/${ipaidporg}/protocol/openid-connect/token**
--userinfo-uri=URI	**https://${ipaidpbaseurl}/realms/${ipaidporg}/protocol/openid-connect/userinfo**
--scope=STR	**openid email**
--idp-user-id=STR	**email**

--provider=okta

Okta に新しい組織を登録すると、新しいベース URL が関連付けられます。このベース URL は、**ipa idp-add** コマンドの **--base-url** オプションで指定できます。

```
[root@client ~]# ipa idp-add MyOkta --provider okta --base-url dev-12345.okta.com --client-id
<your-client-id>
```

--provider=okta を選択すると、次のオプションを使用するように拡張されます。**--base-url** オプションに指定した値は、表内の文字列 **${ipaidpbaseurl}** を置き換えます。

オプション	値
--auth-uri=URI	**https://${ipaidpbaseurl}/oauth2/v1/authorize**
--dev-auth-uri=URI	**https://${ipaidpbaseurl}/oauth2/v1/device/authorize**
--token-uri=URI	**https://${ipaidpbaseurl}/oauth2/v1/token**
--userinfo-uri=URI	**https://${ipaidpbaseurl}/oauth2/v1/userinfo**
--scope=STR	**openid email**
--idp-user-id=STR	**email**

関連情報

- 事前入力された IdP テンプレート

第114章 ANSIBLE を使用して IDM ユーザーの認証を外部アイデンティティープロバイダーに委任する

idp ansible-freeipa モジュールを使用して、OAuth 2 デバイス認証フローをサポートする外部アイデンティティープロバイダー (IdP) にユーザーを関連付けることができます。IdP 参照と関連付けられた IdP ユーザー ID が存在する場合は、それらを使用して、**ユーザーansible-freeipa** モジュールを持つ IdM ユーザーの IdP 認証を有効にすることができます。

これらのユーザーが、RHEL 9.1 以降で利用可能な SSSD バージョンで認証すると、外部 IdP で認証と承認を実行した後に Kerberos チケットを使用した RHEL Identity Management (IdM) Single Sign-On 機能を受け取ります。

114.1. IDM を外部 IDP に接続する利点

管理者は、クラウドサービスプロバイダーなどの外部 ID ソースに保存されているユーザーが、Identity Management (IdM) 環境に追加された RHEL システムにアクセスできるようにすることができます。そのために、これらのユーザーの Kerberos チケットを発行する認証および認可プロセスをその外部エンティティーに委任できます。

この機能を使用して IdM の機能を拡張し、外部 ID プロバイダー (IdP) に保存されているユーザーが IdM によって管理される Linux システムにアクセスできるようにすることができます。

114.2. IDM が外部 IDP を介してログインを組み込む方法

SSSD 2.7.0 には、**idp** Kerberos 事前認証方法を実装する **sssd-idp** パッケージが含まれています。この認証方法は、OAuth 2.0 Device Authorization Grant フローに従って、認可の判断を外部 IdP に委任します。

1. IdM クライアントユーザーは、コマンドラインで **kinit** ユーティリティーを使用して Kerberos TGT の取得を試行するなどして、OAuth 2.0 デバイス認可付与フローを開始します。
2. 特別なコードと Web サイトのリンクが認可サーバーから IdM KDC バックエンドに送信されます。
3. IdM クライアントは、リンクとコードをユーザーに表示します。この例では、IdM クライアントはコマンドラインにリンクとコードを出力します。
4. ユーザーは、別のホストや携帯電話などのブラウザーで Web サイトのリンクを開きます。
   a. ユーザーは特別なコードを入力します。
   b. 必要に応じて、ユーザーは OAuth 2.0 ベースの IdP にログインします。
   c. ユーザーは、クライアントによる情報へのアクセスを許可するよう求められます。
5. ユーザーは、元のデバイスのプロンプトでアクセスを確認します。この例では、ユーザーはコマンドラインで **Enter** キーを押します。
6. IdM KDC バックエンドは、ユーザー情報にアクセスするために OAuth 2.0 認可サーバーをポーリングします。

サポート対象:

- Pluggable Authentication Module (PAM) ライブラリーの呼び出しを可能にする **keyboard-interactive** 認証方法を有効にして、SSH 経由でリモートからログインする場合。

- **logind** サービスを介してコンソールでローカルにログインする場合。
- **kinit** ユーティリティーを使用して Kerberos TGT (Ticket-granting ticket) を取得する場合。

現在のサポート対象外:

- IdM WebUI に直接ログインする場合。IdM WebUI にログインするには、最初に Kerberos チケットを取得する必要があります。
- Cockpit WebUI に直接ログインする場合。Cockpit WebUI にログインするには、最初に Kerberos チケットを取得する必要があります。

関連情報

- Authentication against external Identity Providers
- RFC 8628: OAuth 2.0 Device Authorization Grant

114.3. ANSIBLE を使用して外部アイデンティティープロバイダーへの参照を作成する

外部 ID プロバイダー (IdP) を Identity Management (IdM) 環境に接続するには、IdM で IdP 参照を作成します。**idp ansible-freeipa** モジュールを使用して **github** 外部 IdP への参照を設定するには、この手順を完了します。

前提条件

- IdM を OAuth アプリケーションとして外部 IdP に登録し、IdM ユーザーが IdM への認証に使用するデバイス上でクライアント ID とクライアントシークレットを生成しました。この例では、次のことを前提としています。
  - **my_github_account_name** は、IdM ユーザーが IdM への認証に使用するアカウントを持つ github ユーザーです。
  - **クライアント ID は2efe1acffe9e8ab869f4** です。
  - **クライアントシークレット** は **656a5228abc5f9545c85fa626aecbf69312d398c** です。
- IdM サーバーで RHEL 8.7 以降を使用している。
- IdM サーバーで SSSD 2.7.0 以降を使用している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - RHEL 8.10 以降を使用しています。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。

手順

1. Ansible コントロールノードで、**configure-external-idp-reference.yml** Playbook を作成します。

```
---
- name: Configure external IdP
  hosts: ipaserver
  become: false
  gather_facts: false

  tasks:
  - name: Ensure a reference to github external provider is available
    ipaidp:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: github_idp
      provider: github
      client_ID: 2efe1acffe9e8ab869f4
      secret: 656a5228abc5f9545c85fa626aecbf69312d398c
      idp_user_id: my_github_account_name
```

2. ファイルを保存します。

3. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory configure-external-idp-reference.yml
```

検証

- IdM クライアントで、**ipa idp-show** コマンドの出力に、作成した IdP 参照が表示されていることを確認します。

```
[idmuser@idmclient ~]$ ipa idp-show github_idp
```

次のステップ

- Ansible を使用して IdM ユーザーが外部 IdP 経由で認証できるようにする

関連情報

- idp **ansible-freeipa** アップストリームドキュメント

114.4. ANSIBLE を使用して IDM ユーザーが外部 IDP 経由で認証できるようにする

ユーザー ansible-freeipa モジュールを使用すると、アイデンティティー管理 (IdM) ユーザーが外部アイデンティティープロバイダー (IdP) 経由で認証できるようになります。これを行うには、以前に作成した外部 IdP 参照を IdM ユーザーアカウントに関連付けます。Ansible を使用して、**github_idp** という名前の外部 IdP 参照 **を idm-user-with-external-idp** という名前の IdM ユーザーに関連付けるには、この手順を実行します。この手順の結果、ユーザーは **my_github_account_name** github アイデンティティーを使用して 、**idm-user-with-external-idp** として IdM に認証できるようになります。

前提条件

- IdM クライアントと IdM サーバーで RHEL 8.7 以降を使用している。
- IdM クライアントと IdM サーバーで SSSD 2.7.0 以降を使用している。
- IdM で外部 IdP への参照を作成しました。Ansible を使用して外部アイデンティティープロバイダーへの参照を作成するを 参照してください。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - RHEL 8.10 以降を使用しています。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。

手順

1. Ansible コントロールノードで、**enable-user-to-authenticate-via-external-idp.yml** Playbook を作成します。

```
---
- name: Ensure an IdM user uses an external IdP to authenticate to IdM
  hosts: ipaserver
  become: false
  gather_facts: false

  tasks:
  - name: Retrieve Github user ID
    ansible.builtin.uri:
      url: “https://api.github.com/users/my_github_account_name”
      method: GET
      headers:
        Accept: “application/vnd.github.v3+json”
    register: user_data

  - name: Ensure IdM user exists with an external IdP authentication
    ipauser:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: idm-user-with-external-idp
      first: Example
      last: User
      userauthtype: idp
      idp: github_idp
      idp_user_id: my_github_account_name
```

2. ファイルを保存します。

3. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory enable-user-to-authenticate-via-external-idp.yml
```

検証

- IdM クライアントにログインし、**idm-user-with-external-idp** ユーザーの **ipa user-show** コマンドの出力に IdP への参照が表示されていることを確認します。

```
$ ipa user-show idm-user-with-external-idp
User login: idm-user-with-external-idp
First name: Example
Last name: User
Home directory: /home/idm-user-with-external-idp
Login shell: /bin/sh
Principal name: idm-user-with-external-idp@idm.example.com
Principal alias: idm-user-with-external-idp@idm.example.com
Email address: idm-user-with-external-idp@idm.example.com
ID: 35000003
GID: 35000003
User authentication types: idp
External IdP configuration: github
External IdP user identifier: idm-user-with-external-idp@idm.example.com
Account disabled: False
Password: False
Member of groups: ipausers
Kerberos keys available: False
```

関連情報

- idp **ansible-freeipa** アップストリームドキュメント

114.5. 外部 IDP ユーザーとして IDM チケット許可チケットを取得する

アイデンティティー管理 (IdM) ユーザーの認証を外部アイデンティティープロバイダー (IdP) に委任している場合、IdM ユーザーは外部 IdP に対して認証することで Kerberos チケット保証チケット (TGT) を要求できます。

次の操作を行うには、この手順を完了してください。

1. 匿名の Kerberos チケットを取得してローカルに保存します。
2. **-T** オプションを指定した **kinit** を使用して **idm-user-with-external-idp** ユーザーの TGT を要求し、Flexible Authentication via Secure Tunneling (FAST) チャネルを有効にして、Kerberos クライアントと Kerberos Distribution Center (KDC) 間の安全な接続を提供します。

前提条件

- IdM クライアントと IdM サーバーで RHEL 8.7 以降を使用している。
- IdM クライアントと IdM サーバーで SSSD 2.7.0 以降を使用している。

- IdM で外部 IdP への参照を作成しました。Ansible を使用して外部アイデンティティープロバイダーへの参照を作成するを 参照してください。
- 外部 IdP 参照をユーザーアカウントに関連付けている。Ansible を使用して IdM ユーザーが外部 IdP 経由で認証できるようにするを 参照してください。
- 最初にログインしたユーザーには、ローカルファイルシステム内のディレクトリーに対する書き込み権限があります。

手順

1. 匿名 PKINIT を使用して Kerberos チケットを取得し、それを **./fast.ccache** という名前のファイルに保存します。

```
$ kinit -n -c ./fast.ccache
```

2. [オプション] 取得したチケットを表示します。

```
$ klist -c fast.ccache
Ticket cache: FILE:fast.ccache
Default principal: WELLKNOWN/ANONYMOUS@WELLKNOWN:ANONYMOUS

Valid starting       Expires              Service principal
03/03/2024 13:36:37  03/04/2024 13:14:28
krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
```

3. **-T** オプションを使用してユーザーとして認証を開始し、FAST 通信チャネルを有効にします。

```
[root@client ~]# kinit -T ./fast.ccache idm-user-with-external-idp
Authenticate at https://oauth2.idp.com:8443/auth/realms/master/device?user_code=YHMQ-
XKTL and press ENTER.:
```

4. ブラウザーで、コマンド出力に提供される Web サイトでユーザーとして認証します。
5. コマンドラインで **Enter** キーを押して、認証プロセスを終了します。

検証

- Kerberos チケット情報を表示し、**config: pa_type** の行が外部 IdP による事前認証の **152** を示していることを確認します。

```
[root@client ~]# klist -C
Ticket cache: KCM:0:58420
Default principal: idm-user-with-external-idp@IDM.EXAMPLE.COM

Valid starting     Expires            Service principal
05/09/22 07:48:23  05/10/22 07:03:07  krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
config: fast_avail(krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM) = yes
08/17/2022 20:22:45  08/18/2022 20:22:43
krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
config: pa_type(krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM) = 152
```

114.6. 外部 IDP ユーザーとして SSH 経由で IDM クライアントにログインする

外部 ID プロバイダー (IdP) ユーザーとして SSH 経由で IdM クライアントにログインするには、コマンドラインでログインプロセスを開始します。プロンプトが表示されたら、IdP に関連付けられた Web サイトで認証プロセスを実行し、Identity Management (IdM) クライアントでプロセスを終了します。

前提条件

- IdM クライアントと IdM サーバーで RHEL 8.7 以降を使用している。
- IdM クライアントと IdM サーバーで SSSD 2.7.0 以降を使用している。
- IdM で外部 IdP への参照を作成しました。Ansible を使用して外部アイデンティティープロバイダーへの参照を作成するを 参照してください。
- 外部 IdP 参照をユーザーアカウントに関連付けている。Ansible を使用して IdM ユーザーが外部 IdP 経由で認証できるようにするを 参照してください。

手順

1. SSH 経由で IdM クライアントへのログインを試みます。

```
[user@client ~]$ ssh idm-user-with-external-idp@client.idm.example.com
(idm-user-with-external-idp@client.idm.example.com) Authenticate at
https://oauth2.idp.com:8443/auth/realms/main/device?user_code=XYFL-ROYR and press
ENTER.
```

2. ブラウザーで、コマンド出力に提供される Web サイトでユーザーとして認証します。
3. コマンドラインで **Enter** キーを押して、認証プロセスを終了します。

検証

- Kerberos チケット情報を表示し、**config: pa_type** の行が外部 IdP による事前認証の **152** を示していることを確認します。

```
[idm-user-with-external-idp@client ~]$ klist -C
Ticket cache: KCM:0:58420
Default principal: idm-user-with-external-idp@IDM.EXAMPLE.COM

Valid starting     Expires            Service principal
05/09/22 07:48:23  05/10/22 07:03:07  krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
config: fast_avail(krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM) = yes
08/17/2022 20:22:45  08/18/2022 20:22:43
krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
config: pa_type(krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM) = 152
```

114.7. IPAIDP ANSIBLE モジュールのプロバイダーオプション

次の ID プロバイダー (IdP) は、OAuth 2.0 デバイス認可グラントフローをサポートしています。

- Azure AD を含む Microsoft Identity Platform

- Google
- GitHub
- Red Hat Single Sign-On (SSO) を含む Keycloak
- Okta

idp ansible-freeipa モジュールを使用してこれらの外部 IdP の 1 つへの参照を作成する場合、**ipaidp ansible-freeipa** Playbook タスクの **provider** オプションで IdP タイプを指定できます。これは、以下に説明する追加オプションに拡張されます。

--provider=microsoft

Microsoft Azure IdP では、**組織** オプションで指定できる Azure テナント ID に基づいてパラメーター化を行うことができます。live.com IdP のサポートが必要な場合は、--organization common オプションを指定します。

--provider=microsoft を選択すると、次のオプションを使用するように拡張されます。--organization オプションの値は、表内の文字列 **${ipaidporg}** を置き換えます。

オプション	値
--auth-uri=URI	**https://login.microsoftonline.com/${ipaidporg}/oauth2/v2.0/authorize**
dev_auth_uri: URI	**https://login.microsoftonline.com/${ipaidporg}/oauth2/v2.0/devicecode**
token_uri: URI	**https://login.microsoftonline.com/${ipaidporg}/oauth2/v2.0/token**
userinfo_uri: URI	**https://graph.microsoft.com/oidc/userinfo**
keys_uri: URI	**https://login.microsoftonline.com/common/discovery/v2.0/keys**
--scope=STR	**openid email**
idp_user_id: STR	**email**

--provider=google

--provider=google を選択すると、次のオプションを使用するように拡張されます。

オプション	値
--auth-uri=URI	**https://accounts.google.com/o/oauth2/auth**
dev_auth_uri: URI	**https://oauth2.googleapis.com/device/code**
token_uri: URI	**https://oauth2.googleapis.com/token**

オプション	値
userinfo_uri: URI	**https://openidconnect.googleapis.com/v1/userinfo**
keys_uri: URI	**https://www.googleapis.com/oauth2/v3/certs**
--scope=STR	**openid email**
idp_user_id: STR	**email**

--provider=github

--provider=github を選択すると、次のオプションを使用するように拡張されます。

オプション	値
--auth-uri=URI	**https://github.com/login/oauth/authorize**
dev_auth_uri: URI	**https://github.com/login/device/code**
token_uri: URI	**https://github.com/login/oauth/access_token**
userinfo_uri: URI	**https://openidconnect.googleapis.com/v1/userinfo**
keys_uri: URI	**https://api.github.com/user**
--scope=STR	**user**
idp_user_id: STR	**login**

--provider=keycloak

Keycloak を使用すると、複数のレルムまたは組織を定義できます。これはカスタムデプロイメントの一部であることが多いため、ベース URL とレルム ID の両方が必要であり、**ipaidp** Playbook タスクの **base_url** および **organization** オプションを使用して指定できます。

```
---
- name: Playbook to manage IPA idp
  hosts: ipaserver
  become: false

  tasks:
  - name: Ensure keycloak idp my-keycloak-idp is present using provider
    ipaidp:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: my-keycloak-idp
      provider: keycloak
      organization: main
      base_url: keycloak.domain.com:8443/auth
      client_id: my-keycloak-client-id
```

–

--provider=keycloak を選択すると、次のオプションを使用するように拡張されます。--base-url オプションで指定した値は、表内の文字列 **${ipaidpbaseurl}** を置き換え、指定した値は **--organization `option replaces the string `${ipaidporg}** となります。

オプション	値
--auth-uri=URI	**https://${ipaidpbaseurl}/realms/${ipaidporg}/protocol/openid-connect/auth**
dev_auth_uri: URI	**https://${ipaidpbaseurl}/realms/${ipaidporg}/protocol/openid-connect/auth/device**
token_uri: URI	**https://${ipaidpbaseurl}/realms/${ipaidporg}/protocol/openid-connect/token**
userinfo_uri: URI	**https://${ipaidpbaseurl}/realms/${ipaidporg}/protocol/openid-connect/userinfo**
--scope=STR	**openid email**
idp_user_id: STR	**email**

--provider=okta

Okta に新しい組織を登録すると、新しいベース URL が関連付けられます。このベース URL は、**ipaidp** Playbook タスクの **base_url** オプションで指定できます。

```
---
- name: Playbook to manage IPA idp
  hosts: ipaserver
  become: false

  tasks:
  - name: Ensure okta idp my-okta-idp is present using provider
    ipaidp:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: my-okta-idp
      provider: okta
      base_url: dev-12345.okta.com
      client_id: my-okta-client-id
```

--provider=okta を選択すると、次のオプションを使用するように拡張されます。--base-url オプションに指定した値は、表内の文字列 **${ipaidpbaseurl}** を置き換えます。

オプション	値
--auth-uri=URI	**https://${ipaidpbaseurl}/oauth2/v1/authorize**
dev_auth_uri: URI	**https://${ipaidpbaseurl}/oauth2/v1/device/authorize**

オプション	値
token_uri: URI	**https://${ipaidpbaseurl}/oauth2/v1/token**
userinfo_uri: URI	**https://${ipaidpbaseurl}/oauth2/v1/userinfo**
--scope=STR	**openid email**
idp_user_id: STR	**email**

関連情報

- 事前入力された IdP テンプレート

第115章 IDM とその他の RED HAT 製品の統合

次のリンクは、IdM と統合される他の Red Hat 製品のドキュメントへのリンクです。IdM ユーザーがサービスにアクセスできるように、これらの製品を設定することができます。

Ansible Automation Platform

Setting up LDAP authentication

OpenShift Container Platform

LDAP アイデンティティープロバイダーの設定

OpenStack Platform

OpenStack Identity (keystone) と Red Hat Identity Manager (IdM) の統合

Satellite

Red Hat Identity Management の使用

Single Sign-On

SSSD および FreeIPA Identity Management の統合

仮想化

Configuring an external LDAP provider

第116章 ANSIBLE を使用して IDM を NIS ドメインおよびネットグループと統合する

116.1. NIS とその利点

UNIX 環境では、ネットワーク情報サービス (NIS) は ID と認証を一元管理する一般的な方法です。当初は **Yellow Pages** (YP) という名前が付けられていた NIS は、以下のような認証および ID 情報を一元管理します。

- ユーザーおよびパスワード
- ホスト名および IP アドレス
- POSIX グループ

今日のネットワークインフラストラクチャーでは、NIS は、ホスト認証を提供しておらず、データが暗号化せずにネットワークに送信されるため、セキュリティーが非常に低いと見なされます。この問題を回避するため、NIS はセキュリティーを強化するために他のプロトコルと統合されることが多くあります。

Identity Management(IdM) を使用する場合は、NIS サーバープラグインを使用して、IdM に完全に移行することができないクライアントに接続できます。IdM は、ネットグループおよびその他の NIS データを IdM ドメインに統合します。また、NIS ドメインから IdM にユーザーおよびホストの ID を簡単に移行することもできます。

ネットグループは、NIS グループが想定されるあらゆる場所で使用できます。

関連情報

- IdM の NIS
- IdM の NIS ネットグループ
- NIS から Identity Management への移行

116.2. IDM の NIS

IdM の NIS オブジェクト

NIS オブジェクトは、RFC 2307 に準拠し、Directory Server バックエンドに統合され、保存されます。IdM は、LDAP ディレクトリーに NIS オブジェクトを作成し、クライアントは、たとえば System Security Services Daemon (SSSD) または暗号化された LDAP 接続を使用する **nss_ldap** を通じてそのオブジェクトを取得します。

IdM は、ネットグループ、アカウント、グループ、ホスト、およびその他のデータを管理します。IdM は NIS リスナーを使用してパスワード、グループ、およびネットグループを IdM エントリーにマッピングします。

IdM の NIS プラグイン

NIS サポートの場合、IdM は **slapi-nis** パッケージで提供される以下のプラグインを使用します。

NIS サーバープラグイン

NIS サーバープラグインにより、IdM 統合 LDAP サーバーがクライアントの NIS サーバーとして機能できるようになります。このロールでは、Directory Server は設定に応じて NIS マップを動的に生

成し、更新します。プラグインを使用すると、IdM は NIS プロトコルを使用するクライアントに対して NIS サーバーとして機能します。

スキーマ互換性プラグイン

スキーマ互換性プラグインを使用すると、Directory Server バックエンドは、ディレクトリー情報ツリー (DIT) の一部に保存されたエントリーの代替ビューを提供できるようになります。これには、属性値の追加、ドロップ、名前変更、およびオプションでツリー内の複数のエントリーからの属性値の取得が含まれます。
詳細は、**/usr/share/doc/slapi-nis-version/sch-getting-started.txt** ファイルを参照してください。

116.3. IDM の NIS ネットグループ

NIS エンティティーはネットグループに保存できます。UNIX グループと比較すると、ネットグループは以下のサポートを提供します。

- ネスト化されたグループ (他のグループのメンバーとしてのグループ)。
- ホストのグループ化

ネットグループは、ホスト、ユーザー、およびドメインなどの一連の情報を定義します。このセットは **トリプル** と呼ばれています。以下の 3 つのフィールドを含めることができます。

- 値。
- 「有効な値なし」を指定するダッシュ (**-**)
- 値なし。空のフィールドはワイルドカードを指定します。

(**host.example.com**,,**nisdomain.example.com**)
(-,**user**,**nisdomain.example.com**)

クライアントが NIS ネットグループを要求すると、IdM は以下の項目に LDAP エントリーを変換します。

- 従来の NIS マップへと変換し、これを NIS プラグインを使用して NIS プロトコル経由でクライアントに送信します。
- RFC 2307 または RFC 2307bis に準拠する LDAP 形式に変換します。

116.4. ANSIBLE を使用してネットグループが存在することを確認する

Ansible Playbook を使用して、IdM ネットグループが存在することを確認できます。この例では、**TestNetgroup1** グループが存在することを確認する方法を説明します。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成した。

- **ipaadmin_password** を **Secret.yml** Ansible Vault に保存している。

手順

1. 次の内容を含む Ansible Playbook ファイル **netgroup-present.yml** を作成します。

```
---
- name: Playbook to manage IPA netgroup.
  hosts: ipaserver
  become: no

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure netgroup members are present
    ipanetgroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: TestNetgroup1
```

2. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file path_to_playbooks_directory_/netgroup-
present.yml
```

関連情報

- IdM の NIS
- **/usr/share/doc/ansible-freeipa/README-netgroup.md**
- **/usr/share/doc/ansible-freeipa/playbooks/netgroup**

116.5. ANSIBLE を使用してメンバーがネットグループに存在していることを確認する

Ansible Playbook を使用すると、IdM ユーザー、グループ、およびネットグループがネットグループのメンバーであることを確認できます。この例では、**TestNetgroup1** グループに次のメンバーが含まれていることを確認する方法を説明します。

- **user1** および **user2** IdM ユーザー
- **group1** IdM グループ
- **admins** ネットグループ
- IdM クライアントである **idmclient1** ホスト

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。

- Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
- **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成した。
- **ipaadmin_password** を **Secret.yml** Ansible Vault に保存している。

- **TestNetgroup1** IdM ネットグループが存在します。
- **user1** と **user2** の IdM ユーザーが存在します。
- **group1** IdM グループが存在します。
- **admins** IdM ネットグループが存在します。

手順

1. 次の内容を含む Ansible Playbook ファイル **IdM-members-present-in-a-netgroup.yml** を作成します。

```
---
- name: Playbook to manage IPA netgroup.
  hosts: ipaserver
  become: no

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure netgroup members are present
    ipanetgroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: TestNetgroup1
      user: user1,user2
      group: group1
      host: idmclient1
      netgroup: admins
      action: member
```

2. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file path_to_playbooks_directory_/IdM-
members-present-in-a-netgroup.yml
```

関連情報

- IdM の NIS
- **/usr/share/doc/ansible-freeipa/README-netgroup.md**
- **/usr/share/doc/ansible-freeipa/playbooks/netgroup**

116.6. ANSIBLE を使用してメンバーがネットグループに存在しないことを確認する

Ansible Playbook を使用して、IdM ユーザーがネットグループのメンバーであることを確認できます。この例では、**TestNetgroup1** グループのメンバーに IdM ユーザー **user1** が含まれていないことを確認する方法を説明します。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成した。
  - **ipaadmin_password** を **Secret.yml** Ansible Vault に保存している。
- **TestNetgroup1** ネットグループが存在します。

手順

1. 次の内容を含む Ansible Playbook ファイル **IdM-member-absent-from-a-netgroup.yml** を作成します。

```
---
- name: Playbook to manage IPA netgroup.
  hosts: ipaserver
  become: no

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure netgroup user, "user1", is absent
    ipanetgroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: TestNetgroup1
      user: "user1"
      action: member
      state: absent
```

2. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file path_to_playbooks_directory_/IdM-
member-absent-from-a-netgroup.yml
```

関連情報

- IdM の NIS
- **/usr/share/doc/ansible-freeipa/README-netgroup.md**
- **/usr/share/doc/ansible-freeipa/playbooks/netgroup**

116.7. ANSIBLE を使用してネットグループが存在しないことを確認する

Ansible Playbook を使用して、Identity Management (IdM) にネットグループが存在しないことを確認できます。この例では、**TestNetgroup1** グループが IdM ドメインに存在しないことを確認する方法を説明します。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成した。
  - **ipaadmin_password** を **Secret.yml** Ansible Vault に保存している。

手順

1. 次の内容を含む Ansible Playbook ファイル **netgroup-absent.yml** を作成します。

```
---
- name: Playbook to manage IPA netgroup.
  hosts: ipaserver
  become: no

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure netgroup my_netgroup1 is absent
    ipanetgroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: my_netgroup1
      state: absent
```

2. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file path_to_playbooks_directory_/netgroup-
absent.yml
```

関連情報

- IdM の NIS
- **/usr/share/doc/ansible-freeipa/README-netgroup.md**
- **/usr/share/doc/ansible-freeipa/playbooks/netgroup**

第117章 NIS から IDENTITY MANAGEMENT への移行

ネットワーク情報サービス (NIS) サーバーには、ユーザー、グループ、ホスト、netgroups、および自動マウントマップに関する情報を追加できます。システム管理者は、全ユーザー管理操作が IdM サーバーで実行されるように、これらのエントリータイプ、認証、および承認を NIS サーバーから Identity Management (IdM) サーバーに移行できます。NIS から IdM に移行すると、Kerberos などのよりセキュアなプロトコルを利用できます。

117.1. IDM での NIS の有効化

NIS と Identity Management (IdM) サーバー間で通信できるようにするには、IdM サーバーで NIS 互換性オプションを有効にする必要があります。

前提条件

- IdM サーバーの root アクセス権限がある。

手順

1. IdM サーバーで NIS リスナーと互換性プラグインを有効にします。

```
[root@ipaserver ~]# ipa-nis-manage enable
[root@ipaserver ~]# ipa-compat-manage enable
```

2. **必要に応じて**、より厳密なファイアウォール設定を行うには、固定ポートを設定します。たとえば、ポートを未使用のポート **514** に設定するには、次のコマンドを実行します。

```
[root@ipaserver ~]# ldapmodify -x -D 'cn=directory manager' -W
dn: cn=NIS Server,cn=plugins,cn=config
changetype: modify
add: nsslapd-pluginarg0
nsslapd-pluginarg0: 514
```

警告

他のサービスとの競合を回避するため、1024 を超えるポート番号は使用しないようにしてください。

3. ポートマッパーサービスを有効にして起動します。

```
[root@ipaserver ~]# systemctl enable rpcbind.service
[root@ipaserver ~]# systemctl start rpcbind.service
```

4. Directory Server を再起動します。

```
[root@ipaserver ~]# systemctl restart dirsrv.target
```

117.2. NIS から IDM へのユーザーエントリーの移行

NIS の **passwd** マップには、名前、UID、プライマリーグループ、GECOS、シェル、ホームディレクトリーなどのユーザーに関する情報が含まれます。このデータを使用して、NIS ユーザーアカウントを Identity Management (IdM) に移行します。

前提条件

- NIS サーバーの root アクセス権限がある。
- IdM で NIS が有効になっている。
- NIS サーバーが IdM に登録されている。

手順

1. **yp-tools** パッケージをインストールします。

```
[root@nis-server ~]# yum install yp-tools -y
```

2. NIS サーバーで、以下の内容を含む **/root/nis-users.sh** スクリプトを作成します。

```
#!/bin/sh
# $1 is the NIS domain, $2 is the primary NIS server
ypcat -d $1 -h $2 passwd > /dev/shm/nis-map.passwd 2>&1

IFS=$'\n'
for line in $(cat /dev/shm/nis-map.passwd) ; do
 IFS=' '
 username=$(echo $line | cut -f1 -d:)
 # Not collecting encrypted password because we need cleartext password
 # to create kerberos key
 uid=$(echo $line | cut -f3 -d:)
 gid=$(echo $line | cut -f4 -d:)
 gecos=$(echo $line | cut -f5 -d:)
 homedir=$(echo $line | cut -f6 -d:)
 shell=$(echo $line | cut -f7 -d:)

 # Now create this entry
 echo passw0rd1 | ipa user-add $username --first=NIS --last=USER \
     --password --gidnumber=$gid --uid=$uid --gecos="$gecos" --homedir=$homedir \
     --shell=$shell
 ipa user-show $username
done
```

3. IdM **admin** ユーザーとして認証します。

```
[root@nis-server ~]# kinit admin
```

4. スクリプトを実行します。以下に例を示します。

```
[root@nis-server ~]# sh /root/nis-users.sh nisdomain nis-server.example.com
```

重要

このスクリプトは、名、姓にハードコードされた値を使用し、パスワードを **passw0rd1** に設定します。ユーザーは、次回ログイン時に一時パスワードを変更する必要があります。

117.3. ユーザーグループの NIS から IDM への移行

NIS **グループ** マップには、グループ名、GID、グループメンバーなどのグループ情報が含まれます。このデータを使用して、NIS グループを Identity Management (IdM) に移行します。

前提条件

- NIS サーバーの root アクセス権限がある。
- IdM で NIS が有効になっている。
- NIS サーバーが IdM に登録されている。

手順

1. **yp-tools** パッケージをインストールします。

```
[root@nis-server ~]# yum install yp-tools -y
```

2. 以下の内容を含めて、NIS サーバーに **/root/nis-groups.sh** スクリプトを作成します。

```
#!/bin/sh
# $1 is the NIS domain, $2 is the primary NIS server
ypcat -d $1 -h $2 group > /dev/shm/nis-map.group 2>&1

IFS=$'\n'
for line in $(cat /dev/shm/nis-map.group); do
 IFS=' '
 groupname=$(echo $line | cut -f1 -d:)
 # Not collecting encrypted password because we need cleartext password
 # to create kerberos key
 gid=$(echo $line | cut -f3 -d:)
 members=$(echo $line | cut -f4 -d:)

 # Now create this entry
 ipa group-add $groupname --desc=NIS_GROUP_$groupname --gid=$gid
 if [ -n "$members" ]; then
  ipa group-add-member $groupname --users={$members}
 fi
 ipa group-show $groupname
done
```

3. IdM **admin** ユーザーとして認証します。

```
[root@nis-server ~]# kinit admin
```

4. スクリプトを実行します。以下に例を示します。

```
[root@nis-server ~]# sh /root/nis-groups.sh nisdomain nis-server.example.com
```

117.4. ホストエントリーの NIS から IDM への移行

NIS **ホスト** マップには、ホスト名や IP アドレスなどのホストに関する情報が含まれます。このデータを使用して、NIS ホストエントリーを Identity Management (IdM) に移行します。

注記

IdM でホストグループを作成すると、対応するシャドウの NIS グループが自動的に作成されます。これらのシャドウ NIS グループに **ipa netgroup-*** コマンドを使用しないでください。**ipa netgroup-*** コマンドは、**netgroup-add** コマンドで作成されたネイティブの netgroups の管理にだけ使用します。

前提条件

- NIS サーバーの root アクセス権限がある。
- IdM で NIS が有効になっている。
- NIS サーバーが IdM に登録されている。

手順

1. **yp-tools** パッケージをインストールします。

```
[root@nis-server ~]# yum install yp-tools -y
```

2. 以下の内容を含めて、NIS サーバーに **/root/nis-hosts.sh** スクリプトを作成します。

```
#!/bin/sh
# $1 is the NIS domain, $2 is the primary NIS server
ypcat -d $1 -h $2 hosts | egrep -v "localhost|127.0.0.1" > /dev/shm/nis-map.hosts 2>&1

IFS=$'\n'
for line in $(cat /dev/shm/nis-map.hosts); do
 IFS=' '
 ipaddress=$(echo $line | awk '{print $1}')
 hostname=$(echo $line | awk '{print $2}')
 primary=$(ipa env xmlrpc_uri | tr -d '[:space:]' | cut -f3 -d: | cut -f3 -d/)
 domain=$(ipa env domain | tr -d '[:space:]' | cut -f2 -d:)
 if [ $(echo $hostname | grep "\." |wc -l) -eq 0 ] ; then
  hostname=$(echo $hostname.$domain)
 fi
 zone=$(echo $hostname | cut -f2- -d.)
 if [ $(ipa dnszone-show $zone 2>/dev/null | wc -l) -eq 0 ] ; then
  ipa dnszone-add --name-server=$primary --admin-email=root.$primary
 fi
 ptrzone=$(echo $ipaddress | awk -F. '{print $3 "." $2 "." $1 ".in-addr.arpa."}')
 if [ $(ipa dnszone-show $ptrzone 2>/dev/null | wc -l) -eq 0 ] ; then
  ipa dnszone-add  $ptrzone --name-server=$primary --admin-email=root.$primary
 fi
 # Now create this entry
```

```
 ipa host-add $hostname --ip-address=$ipaddress
 ipa host-show $hostname
done
```

3. IdM **admin** ユーザーとして認証します。

```
[root@nis-server ~]# kinit admin
```

4. スクリプトを実行します。以下に例を示します。

```
[root@nis-server ~]# sh /root/nis-hosts.sh nisdomain nis-server.example.com
```

注記

このスクリプトでは、エイリアスなどの特別なホスト設定は移行されません。

117.5. NETGROUP エントリーの NIS から IDM への移行

NIS **netgroup** マップには、netgroup に関する情報が含まれます。このデータを使用して、NIS netgroup を Identity Management (IdM) に移行します。

前提条件

- NIS サーバーの root アクセス権限がある。
- IdM で NIS が有効になっている。
- NIS サーバーが IdM に登録されている。

手順

1. **yp-tools** パッケージをインストールします。

```
[root@nis-server ~]# yum install yp-tools -y
```

2. 以下の内容を含めて NIS サーバーに **/root/nis-netgroups.sh** スクリプトを作成します。

```
#!/bin/sh
# $1 is the NIS domain, $2 is the primary NIS server
ypcat -k -d $1 -h $2 netgroup > /dev/shm/nis-map.netgroup 2>&1

IFS=$'\n'
for line in $(cat /dev/shm/nis-map.netgroup); do
 IFS=' '
 netgroupname=$(echo $line | awk '{print $1}')
 triples=$(echo $line | sed "s/^$netgroupname //")
 echo "ipa netgroup-add $netgroupname --desc=NIS_NG_$netgroupname"
 if [ $(echo $line | grep "(," | wc -l) -gt 0 ]; then
  echo "ipa netgroup-mod $netgroupname --hostcat=all"
 fi
 if [ $(echo $line | grep ",," | wc -l) -gt 0 ]; then
  echo "ipa netgroup-mod $netgroupname --usercat=all"
 fi
```

```
 for triple in $triples; do
  triple=$(echo $triple | sed -e 's/-//g' -e 's/(//' -e 's/)//')
  if [ $(echo $triple | grep ",.*," | wc -l) -gt 0 ]; then
   hostname=$(echo $triple | cut -f1 -d,)
   username=$(echo $triple | cut -f2 -d,)
   domain=$(echo $triple | cut -f3 -d,)
   hosts=""; users=""; doms="";
   [ -n "$hostname" ] && hosts="--hosts=$hostname"
   [ -n "$username" ] && users="--users=$username"
   [ -n "$domain"   ] && doms="--nisdomain=$domain"
   echo "ipa netgroup-add-member $netgroup $hosts $users $doms"
  else
   netgroup=$triple
   echo "ipa netgroup-add $netgroup --desc=<NIS_NG>_$netgroup"
  fi
 done
done
```

3. IdM **admin** ユーザーとして認証します。

```
[root@nis-server ~]# kinit admin
```

4. スクリプトを実行します。以下に例を示します。

```
[root@nis-server ~]# sh /root/nis-netgroups.sh nisdomain nis-server.example.com
```

117.6. NIS から IDM への自動マウントマップの移行

自動マウントマップは、場所 (親エントリー)、関連のキー、およびマップを定義する入れ子および相互関連のエントリーです。NIS 自動マウントマップを Identity Management (IdM) に移行するには、以下を実行します。

前提条件

- NIS サーバーの root アクセス権限がある。
- IdM で NIS が有効になっている。
- NIS サーバーが IdM に登録されている。

手順

1. **yp-tools** パッケージをインストールします。

```
[root@nis-server ~]# yum install yp-tools -y
```

2. 以下の内容を含めて、NIS サーバーに **/root/nis-automounts.sh** スクリプトを作成します。

```
#!/bin/sh
# $1 is for the automount entry in ipa

ipa automountlocation-add $1
```

```
# $2 is the NIS domain, $3 is the primary NIS server, $4 is the map name

ypcat -k -d $2 -h $3 $4 > /dev/shm/nis-map.$4 2>&1

ipa automountmap-add $1 $4

basedn=$(ipa env basedn | tr -d '[:space:]' | cut -f2 -d:)
cat > /tmp/amap.ldif <<EOF
dn: nis-domain=$2+nis-map=$4,cn=NIS Server,cn=plugins,cn=config
objectClass: extensibleObject
nis-domain: $2
nis-map: $4
nis-base: automountmapname=$4,cn=$1,cn=automount,$basedn
nis-filter: (objectclass=\*)
nis-key-format: %{automountKey}
nis-value-format: %{automountInformation}
EOF
ldapadd -x -h $3 -D "cn=Directory Manager" -W -f /tmp/amap.ldif

IFS=$'\n'
for line in $(cat /dev/shm/nis-map.$4); do
 IFS=" "
 key=$(echo "$line" | awk '{print $1}')
 info=$(echo "$line" | sed -e "s^$key[ \t]*")
 ipa automountkey-add nis $4 --key="$key" --info="$info"
done
```

注記

このスクリプトでは、NIS 自動マウント情報のエクスポート、自動マウントの場所と関連マップの LDAP データ交換形式 (LDIF) の生成、IdM Directory Server への LDIF ファイルのインポートが行われます。

3. IdM **admin** ユーザーとして認証します。

```
[root@nis-server ~]# kinit admin
```

4. スクリプトを実行します。以下に例を示します。

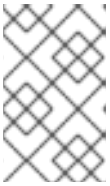
```
[root@nis-server ~]# sh /root/nis-automounts.sh location nisdomain
    nis-server.example.com map_name
```

第118章 IDM で自動マウントの使用

自動マウントは、複数のシステムにわたってディレクトリーを管理、整理、およびアクセスする方法です。Automount は、ディレクトリーへのアクセスが要求されるたびに、そのディレクトリーを自動的にマウントします。これは、ドメイン内のクライアント上のディレクトリーを簡単に共有できるため、Identity Management (IdM) ドメイン内でうまく機能します。

この例では、以下のシナリオを使用します。

- **nfs-server.idm.example.com** は、ネットワークファイルシステム (NFS) サーバーの完全修飾ドメイン名 (FQDN) です。
- 便宜上、**nfs-server.idm.example.com** は、**raleigh** 自動マウントの場所のマップを提供する IdM クライアントとします。



注記

自動マウントの場所は、NFS マップの一意のセットです。たとえば、クライアントが高速接続の恩恵を受けられるように、これらの NFS マップがすべて同じ地理的地域にあることが理想的ですが、これは必須ではありません。

- NFS サーバーは、**/exports/project** ディレクトリーを読み取り/書き込みとしてエクスポートします。
- **developers** グループに属する IdM ユーザーは、**raleigh**の自動マウントの場所を使用する IdM クライアントであれば、**/devel/project/** として、エクスポートされたディレクトリーのコンテンツにアクセスできます。
- **idm-client.idm.example.com** は、**raleigh** の自動マウントの場所を使用する IdM クライアントです。

重要

NFS サーバーの代わりに Samba サーバーを使用して IdM クライアントに共有を提供する場合は、以下の How do I configure kerberized CIFS mounts with Autofs in an IPA environment? を参照してください。を参照してください。

118.1. IDM の AUTOFS と AUTOMOUNT

autofs サービスは、アクセス時にディレクトリーをマウントするように **automount** デーモンに指示することにより、必要に応じてディレクトリーのマウントを自動化します。また、しばらく操作を行わないと、**autofs** は、**automount** に自動マウントされたディレクトリーのマウントを解除するように指示します。静的マウントとは異なり、オンデマンドマウントはシステムリソースを節約します。

マップの自動マウント

autofs を使用するシステムでは、**automount** 設定は複数のファイルに保存されます。プライマリー **automount** 設定ファイルは **/etc/auto.master** です。これには、システムの **automount** マウントポイントのマスターマッピングと、その関連リソースが含まれます。このマッピングは **自動マウントマップ** として知られています。
/etc/auto.master 設定ファイルには、**マスターマップ** が含まれます。他のマップへの参照を含めることができます。このマップは、直接または間接のいずれかになります。ダイレクトマップではマウントポイントに絶対パス名を使用し、間接マップでは相対パス名を使用します。

IdM の自動マウント設定

automount は通常、ローカルの **/etc/auto.master** と関連ファイルからマップデータを取得しますが、他のソースからマップデータを取得することもできます。一般的なソースの 1 つが LDAP サーバーです。Identity Management (Id M) のコンテキストでは、これは 389 Directory Server です。**autofs** を使用するシステムが IdM ドメインのクライアントである場合、**automount** 設定はローカル設定ファイルに保存されません。代わりに、マップ、場所、キーなどの **autofs** 設定は、LDAP エントリーとして IdM ディレクトリーに保存されます。たとえば、**idm.example.com** IdM ドメインの場合、デフォルトの **マスターマップ** は以下のように保存されます。

```
dn:
automountmapname=auto.master,cn=default,cn=automount,dc=idm,dc=example,dc=com
objectClass: automountMap
objectClass: top
automountMapName: auto.master
```

関連情報

- オンデマンドでのファイルシステムのマウント

118.2. RED HAT IDENTITY MANAGEMENT ドメインで KERBEROS を使用する NFS サーバーを設定する

Red Hat Identity Management (IdM) を使用すると、NFS サーバーを IdM ドメインに参加させることができます。これにより、ユーザーとグループを一元管理し、認証、整合性保護、トラフィック暗号化に Kerberos を使用できるようになります。

前提条件

- NFS サーバーが Red Hat Identity Management (IdM) ドメインに 登録 されている。
- NFS サーバーが実行および設定されている。

手順

1. IdM 管理者として Kerberos チケットを取得します。

```
# kinit admin
```

2. **nfs/<FQDN>** サービスプリンシパルを作成します。

```
# ipa service-add nfs/nfs_server.idm.example.com
```

3. IdM から **nfs** サービスプリンシパルを取得し、**/etc/krb5.keytab** ファイルに保存します。

```
# ipa-getkeytab -s idm_server.idm.example.com -p nfs/nfs_server.idm.example.com -k
/etc/krb5.keytab
```

4. オプション: **/etc/krb5.keytab** ファイル内のプリンシパルを表示します。

```
# klist -k /etc/krb5.keytab
Keytab name: FILE:/etc/krb5.keytab
KVNO Principal
---- --------------------------------------------------------------------------
```

```
1 nfs/nfs_server.idm.example.com@IDM.EXAMPLE.COM
1 nfs/nfs_server.idm.example.com@IDM.EXAMPLE.COM
1 nfs/nfs_server.idm.example.com@IDM.EXAMPLE.COM
1 nfs/nfs_server.idm.example.com@IDM.EXAMPLE.COM
7 host/nfs_server.idm.example.com@IDM.EXAMPLE.COM
7 host/nfs_server.idm.example.com@IDM.EXAMPLE.COM
7 host/nfs_server.idm.example.com@IDM.EXAMPLE.COM
7 host/nfs_server.idm.example.com@IDM.EXAMPLE.COM
```

デフォルトでは、ホストを IdM ドメインに参加させると、IdM クライアントがホストプリンシパルを **/etc/krb5.keytab** ファイルに追加します。ホストプリンシパルがない場合は、**ipa-getkeytab -s idm_server.idm.example.com -p host/nfs_server.idm.example.com -k /etc/krb5.keytab** コマンドを使用して追加します。

5. **ipa-client-automount** ユーティリティーを使用して、IdM ID のマッピングを設定します。

```
# ipa-client-automount
Searching for IPA server...
IPA server: DNS discovery
Location: default
Continue to configure the system with these values? [no]: yes
Configured /etc/idmapd.conf
Restarting sssd, waiting for it to become available.
Started autofs
```

6. **/etc/exports** ファイルを更新し、クライアントオプションに Kerberos セキュリティー方式を追加します。以下に例を示します。

```
/nfs/projects/      192.0.2.0/24(rw,sec=krb5i)
```

クライアントが複数のセキュリティー方式を選択できるようにするには、それらをコロンで区切って指定します。

```
/nfs/projects/      192.0.2.0/24(rw,sec=krb5:krb5i:krb5p)
```

7. エクスポートされたファイルシステムを再ロードします。

```
# exportfs -r
```

118.3. IDM CLI を使用した IDM での自動マウントの場所とマップの設定

場所はマップのセットで、すべて **auto.master** に保存されます。1 つの場所に複数のマップを保存できます。また、場所には複数のマップを保存できます。場所のエントリーは、マップエントリーのコンテナーとしてのみ機能します。それ自体は、自動マウント設定ではありません。

Identity Management (IdM) のシステム管理者は、IdM で自動マウントの場所とマップを設定できます。これにより、指定した場所の IdM ユーザーが、ホストの特定のマウントポイントに移動して、NFS サーバーがエクスポートした共有にアクセスできるようになります。エクスポートされた NFS サーバーディレクトリーとマウントポイントの両方が、マップで指定されます。この例では、**raleigh** の場所と、IdM クライアントの**/devel/** マウントポイントにある **nfs-server.idm.example.com:/exports/project** 共有を読み取り/書き込みディレクトリーとしてマウントするマップを設定する方法を説明します。

前提条件

- IdM に登録されているホストに IdM 管理者としてログインしている。

手順

1. **raleigh** の自動マウントの場所を作成します。

```
$ ipa automountlocation-add raleigh
----------------------------------
Added automount location "raleigh"
----------------------------------
  Location: raleigh
```

2. **raleigh**の場所に、**auto.devel** 自動マウントマップを作成します。

```
$ ipa automountmap-add raleigh auto.devel
--------------------------------
Added automount map "auto.devel"
--------------------------------
  Map: auto.devel
```

3. **exports/**共有のキーとマウント情報を追加します。

   a. **auto.devel** マップのキーとマウント情報を追加します。

```
$ ipa automountkey-add raleigh auto.devel --key='*' --info='-sec=krb5p,vers=4 nfs-server.idm.example.com:/exports/&'
-----------------------
Added automount key "*"
-----------------------
  Key: *
  Mount information: -sec=krb5p,vers=4 nfs-server.idm.example.com:/exports/&
```

   b. **auto.master** マップのキーとマウント情報を追加します。

```
$ ipa automountkey-add raleigh auto.master --key=/devel --info=auto.devel
---------------------------
Added automount key "/devel"
---------------------------
  Key: /devel
  Mount information: auto.devel
```

118.4. IDM クライアントでの自動マウントの設定

Identity Management (IdM) システム管理者は、IdM クライアントに自動マウントサービスを設定して、クライアントが追加された場所に設定した NFS 共有に、ユーザーがクライアントにログインしたときに IdM ユーザーが自動的にアクセスできるようにすることができます。この例では、**raleigh** の場所で利用可能な自動マウントサービスを使用するように IdM クライアントを設定する方法を説明します。

前提条件

- IdM クライアントへの **root** アクセス権限がある。

- IdM 管理者としてログインしている。
- 自動マウントの場所が存在します。サンプルの場所は **raleigh** です。

手順

1. IdM クライアントで、**ipa-client-automount** コマンドを入力して場所を指定します。**-U** オプションを使用して、スクリプトを無人で実行します。

```
# ipa-client-automount --location raleigh -U
```

2. autofs サービスを停止し、SSSD キャッシュをクリアし、autofs サービスを開始して新しい設定をロードします。

```
# systemctl stop autofs ; sss_cache -E ; systemctl start autofs
```

118.5. IDM クライアントで、IDM ユーザーが NFS 共有にアクセスできることの確認

Identity Management (IdM) システム管理者は、特定のグループのメンバーである IdM ユーザーが、特定の IdM クライアントにログインしたときに NFS 共有にアクセスできるかどうかをテストできます。

この例では、以下のシナリオがテストされています。

- **developers** グループに属する **idm_user** という名前の IdM ユーザーは、 **raleigh** automount ロケーションにある IdM クライアントである **idm-client.idm.example.com** に自動マウントされた **/devel/project** ディレクトリー内のファイルの内容を読み書きできます。

前提条件

- IdM ホスト上で Kerberos を使用して NFS サーバーをセットアップしまし た。
- IdM で自動マウントのロケーション、マップ、マウントポイントを設定 し、そこで IdM ユーザーが NFS共有にアクセスできるように設定している。
- IdM クライアントに自動マウントを設定 している。

手順

1. IdM ユーザーが **read-write** ディレクトリーにアクセスできることを確認します。
   a. IdM ユーザーとして IdM クライアントに接続します。

```
$ ssh idm_user@idm-client.idm.example.com
Password:
```

   b. IdM ユーザーの Ticket Granting Ticket (TGT) を取得します。

```
$ kinit idm_user
```

   c. [オプション] Id M ユーザーのグループメンバーシップを表示します。

```
$ ipa user-show idm_user
```

```
User login: idm_user
[...]
Member of groups: developers, ipausers
```

d. **/devel/project** ディレクトリーに移動します。

```
$ cd /devel/project
```

e. ディレクトリーの内容をリスト表示します。

```
$ ls
rw_file
```

f. ディレクトリーのファイルに行を追加し、**write** パーミッションをテストします。

```
$ echo "idm_user can write into the file" > rw_file
```

g. [オプション] ファイルの更新された内容を表示します。

```
$ cat rw_file
this is a read-write file
idm_user can write into the file
```

出力は、**idm_user** がファイルに書き込めることを確認します。

第119章 ANSIBLE を使用して IDM ユーザーの NFS 共有を自動マウントする

自動マウントは、複数のシステムにわたってディレクトリーを管理、整理、およびアクセスする方法です。Automount は、ディレクトリーへのアクセスが要求されるたびに、そのディレクトリーを自動的にマウントします。これは、ドメイン内のクライアント上のディレクトリーを簡単に共有できるため、Identity Management (IdM) ドメイン内でうまく機能します。

Ansible を使用して、IdM ロケーションの IdM クライアントにログインしている IdM ユーザーに対して NFS 共有が自動的にマウントされるように設定できます。

この章の例では、次のシナリオを使用します。

- **nfs-server.idm.example.com** は、ネットワークファイルシステム (NFS) サーバーの完全修飾ドメイン名 (FQDN) です。
- **nfs-server.idm.example.com** は、**raleigh** の自動マウントの場所にある IdM クライアントです。
- NFS サーバーは、**/exports/project** ディレクトリーを読み取り/書き込みとしてエクスポートします。
- **開発者** グループに属する IdM ユーザーは、NFS サーバーと同じ **raleigh**の自動マウントの場所にある IdM クライアントであれば、**/devel/project/** としてエクスポートしたディレクトリーのコンテンツにアクセスできます。
- **idm-client.idm.example.com** は、**raleigh** の自動マウントの場所にある IdM クライアントです。

重要

NFS サーバーの代わりに Samba サーバーを使用して IdM クライアントに共有を提供する場合は、以下の How do I configure kerberized CIFS mounts with Autofs in an IPA environment? を参照してください。を参照してください。

本章は以下のセクションで設定されます。

1. IdM の Autofs と automount
2. IdM での Kerberos を使用した NFS サーバーのセットアップ
3. Ansible を使用した IdM での自動マウントの場所、マップ、およびキーの設定
4. Ansible を使用して IdM ユーザーを NFS 共有を所有するグループに追加する
5. IdM クライアントでの自動マウントの設定
6. IdM クライアントで、IdM ユーザーが NFS 共有にアクセスできることの確認

119.1. IDM の AUTOFS と AUTOMOUNT

autofs サービスは、アクセス時にディレクトリーをマウントするように **automount** デーモンに指示することにより、必要に応じてディレクトリーのマウントを自動化します。また、しばらく操作を行わな

いと、**autofs** は、**automount** に自動マウントされたディレクトリーのマウントを解除するように指示します。静的マウントとは異なり、オンデマンドマウントはシステムリソースを節約します。

マップの自動マウント

autofs を使用するシステムでは、**automount** 設定は複数のファイルに保存されます。プライマリー **automount** 設定ファイルは **/etc/auto.master** です。これには、システムの **automount** マウントポイントのマスターマッピングと、その関連リソースが含まれます。このマッピングは **自動マウントマップ** として知られています。
/etc/auto.master 設定ファイルには、**マスターマップ** が含まれます。他のマップへの参照を含めることができます。このマップは、直接または間接のいずれかになります。ダイレクトマップではマウントポイントに絶対パス名を使用し、間接マップでは相対パス名を使用します。

IdM の自動マウント設定

automount は通常、ローカルの **/etc/auto.master** と関連ファイルからマップデータを取得しますが、他のソースからマップデータを取得することもできます。一般的なソースの 1 つが LDAP サーバーです。Identity Management (Id M) のコンテキストでは、これは 389 Directory Server です。
autofs を使用するシステムが IdM ドメインのクライアントである場合、**automount** 設定はローカル設定ファイルに保存されません。代わりに、マップ、場所、キーなどの **autofs** 設定は、LDAP エントリーとして IdM ディレクトリーに保存されます。たとえば、**idm.example.com** IdM ドメインの場合、デフォルトの **マスターマップ** は以下のように保存されます。

```
dn:
automountmapname=auto.master,cn=default,cn=automount,dc=idm,dc=example,dc=com
objectClass: automountMap
objectClass: top
automountMapName: auto.master
```

関連情報

- オンデマンドでのファイルシステムのマウント

119.2. RED HAT IDENTITY MANAGEMENT ドメインで KERBEROS を使用する NFS サーバーを設定する

Red Hat Identity Management (IdM) を使用すると、NFS サーバーを IdM ドメインに参加させることができます。これにより、ユーザーとグループを一元管理し、認証、整合性保護、トラフィック暗号化に Kerberos を使用できるようになります。

前提条件

- NFS サーバーが Red Hat Identity Management (IdM) ドメインに 登録 されている。
- NFS サーバーが実行および設定されている。

手順

1. IdM 管理者として Kerberos チケットを取得します。

```
# kinit admin
```

2. **nfs/<FQDN>** サービスプリンシパルを作成します。

```
# ipa service-add nfs/nfs_server.idm.example.com
```

3. IdM から **nfs** サービスプリンシパルを取得し、**/etc/krb5.keytab** ファイルに保存します。

```
# ipa-getkeytab -s idm_server.idm.example.com -p nfs/nfs_server.idm.example.com -k
/etc/krb5.keytab
```

4. オプション: **/etc/krb5.keytab** ファイル内のプリンシパルを表示します。

```
# klist -k /etc/krb5.keytab
Keytab name: FILE:/etc/krb5.keytab
KVNO Principal
---- --------------------------------------------------------------------------
   1 nfs/nfs_server.idm.example.com@IDM.EXAMPLE.COM
   1 nfs/nfs_server.idm.example.com@IDM.EXAMPLE.COM
   1 nfs/nfs_server.idm.example.com@IDM.EXAMPLE.COM
   1 nfs/nfs_server.idm.example.com@IDM.EXAMPLE.COM
   7 host/nfs_server.idm.example.com@IDM.EXAMPLE.COM
   7 host/nfs_server.idm.example.com@IDM.EXAMPLE.COM
   7 host/nfs_server.idm.example.com@IDM.EXAMPLE.COM
   7 host/nfs_server.idm.example.com@IDM.EXAMPLE.COM
```

デフォルトでは、ホストを IdM ドメインに参加させると、IdM クライアントがホストプリンシパルを **/etc/krb5.keytab** ファイルに追加します。ホストプリンシパルがない場合は、**ipa-getkeytab -s idm_server.idm.example.com -p host/nfs_server.idm.example.com -k /etc/krb5.keytab** コマンドを使用して追加します。

5. **ipa-client-automount** ユーティリティーを使用して、IdM ID のマッピングを設定します。

```
#  ipa-client-automount
Searching for IPA server...
IPA server: DNS discovery
Location: default
Continue to configure the system with these values? [no]: yes
Configured /etc/idmapd.conf
Restarting sssd, waiting for it to become available.
Started autofs
```

6. **/etc/exports** ファイルを更新し、クライアントオプションに Kerberos セキュリティー方式を追加します。以下に例を示します。

```
/nfs/projects/       192.0.2.0/24(rw,sec=krb5i)
```

クライアントが複数のセキュリティー方式を選択できるようにするには、それらをコロンで区切って指定します。

```
/nfs/projects/       192.0.2.0/24(rw,sec=krb5:krb5i:krb5p)
```

7. エクスポートされたファイルシステムを再ロードします。

```
# exportfs -r
```

119.3. ANSIBLE を使用した IDM での自動マウントの場所、マップ、およびキーの設定

Identity Management (IdM) のシステム管理者は、IdM で自動マウントの場所とマップを設定できます。これにより、指定した場所の IdM ユーザーが、ホストの特定のマウントポイントに移動して、NFS サーバーがエクスポートした共有にアクセスできるようになります。エクスポートされた NFS サーバーディレクトリーとマウントポイントの両方が、マップで指定されます。LDAP 用語では、ロケーションはそのようなマップエントリーのコンテナーです。

この例では、Ansible を使用して、**raleigh** の場所と、IdM クライアント上の **/devel/project** マウントポイントに **nfs-server.idm.example.com:/exports/project** 共有を読み取り/書き込みディレクトリーとしてマウントするマップを設定する方法を説明します。

前提条件

- IdM **admin** のパスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. Ansible コントロールノードで、**~/MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/automount/** ディレクトリーにある **automount-location-present.yml** Ansible Playbook ファイルをコピーします。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/automount/automount-location-present.yml automount-location-map-and-key-present.yml
```

3. **automount-location-map-and-key-present.yml** ファイルを編集用に開きます。
4. **ipaautomountlocation** タスクセクションで次の変数を設定して、ファイルを調整します。
   - **ipaadmin_password** 変数は IdM **admin** のパスワードに設定します。
   - **name** 変数を **raleigh** に設定します。
   - **state** 変数は **present** に設定されていることを確認します。
     以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Automount location present example
  hosts: ipaserver
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure automount location is present
    ipaautomountlocation:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: raleigh
      state: present
```

5. **automount-location-map-and-key-present.yml** ファイルの編集を続けます。

   a. **tasks** セクションで、自動マウントマップの存在を確認するタスクを追加します。

```
[...]
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
[...]
  - name: ensure map named auto.devel in location raleigh is created
    ipaautomountmap:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: auto.devel
      location: raleigh
      state: present
```

   b. 別のタスクを追加して、マウントポイントと NFS サーバー情報をマップに追加します。

```
[...]
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
[...]
  - name: ensure automount key /devel/project is present
    ipaautomountkey:
      ipaadmin_password: "{{ ipaadmin_password }}"
      location: raleigh
      mapname: auto.devel
      key: /devel/project
      info: nfs-server.idm.example.com:/exports/project
      state: present
```

   c. **auto.devel** が **auto.master** に接続されていることを確認する別のタスクを追加します。

```
[...]
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
[...]
- name: Ensure auto.devel is connected in auto.master:
  ipaautomountkey:
    ipaadmin_password: "{{ ipaadmin_password }}"
    location: raleigh
```

```
mapname: auto.map
key: /devel
info: auto.devel
state: present
```

6. ファイルを保存します。

7. Ansible Playbook を実行し、Playbook とインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory automount-location-map-and-key-present.yml
```

119.4. ANSIBLE を使用した NFS 共有を所有するグループへの IDM ユーザーの追加

Identity Management (IdM) システム管理者は、Ansible を使用して、NFS 共有にアクセスできるユーザーのグループを作成し、IdM ユーザーをこのグループに追加できます。

この例では、Ansible Playbook を使用して **idm_user** アカウントが **developers** グループに属していることを確認し、**idm_user** が **/exports/project** NFS 共有にアクセスできるようにする方法について説明します。

前提条件

- **nfs-server.idm.example.com** NFS サーバーへの **root** アクセス権があり、これは **raleigh** automount の場所にある IdM クライアントです。
- IdM **admin** のパスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。
  - **~/MyPlaybooks/** で、Ansible を使用して IdM で automount の場所、マップ、およびキーを設定 するからのタスクがすでに含まれている **automount-location-map-and-key-present.yml** ファイルを作成しました。

手順

1. Ansible コントロールノードで、**~/MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. **automount-location-map-and-key-present.yml** ファイルを編集用に開きます。

3. **tasks** セクションで、IdM **developers** グループが存在し、**idm_user** がこのグループに追加されていることを確認するタスクを追加します。

```
[...]
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
[...]
  - ipagroup:
     ipaadmin_password: "{{ ipaadmin_password }}"
      name: developers
      user:
      - idm_user
      state: present
```

4. ファイルを保存します。

5. Ansible Playbook を実行し、Playbook とインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory automount-location-map-and-key-present.yml
```

6. NFS サーバーで、**/exports/project** ディレクトリーのグループ所有権を **developers** に変更して、グループ内のすべての IdM ユーザーがディレクトリーにアクセスできるようにします。

```
# chgrp developers /exports/project
```

119.5. IDM クライアントでの自動マウントの設定

Identity Management (IdM) システム管理者は、IdM クライアントに自動マウントサービスを設定して、クライアントが追加された場所に設定した NFS 共有に、ユーザーがクライアントにログインしたときに IdM ユーザーが自動的にアクセスできるようにすることができます。この例では、**raleigh** の場所で利用可能な自動マウントサービスを使用するように IdM クライアントを設定する方法を説明します。

前提条件

- IdM クライアントへの **root** アクセス権限がある。
- IdM 管理者としてログインしている。
- 自動マウントの場所が存在します。サンプルの場所は **raleigh** です。

手順

1. IdM クライアントで、**ipa-client-automount** コマンドを入力して場所を指定します。**-U** オプションを使用して、スクリプトを無人で実行します。

```
# ipa-client-automount --location raleigh -U
```

2. autofs サービスを停止し、SSSD キャッシュをクリアし、autofs サービスを開始して新しい設定をロードします。

```
# systemctl stop autofs ; sss_cache -E ; systemctl start autofs
```

119.6. IDM クライアントで、IDM ユーザーが NFS 共有にアクセスできることの確認

Identity Management (IdM) システム管理者は、特定のグループのメンバーである IdM ユーザーが、特定の IdM クライアントにログインしたときに NFS 共有にアクセスできるかどうかをテストできます。

この例では、以下のシナリオがテストされています。

- **developers** グループに属する **idm_user** という名前の IdM ユーザーは、 **raleigh** automount ロケーションにある IdM クライアントである **idm-client.idm.example.com** に自動マウントされた **/devel/project** ディレクトリー内のファイルの内容を読み書きできます。

前提条件

- IdM ホスト上で Kerberos を使用して NFS サーバーをセットアップしまし た。
- IdM で自動マウントのロケーション、マップ、マウントポイントを設定 し、そこで IdM ユーザーが NFS共有にアクセスできるように設定している。
- Ansible を使用して、NFS 共有を所有する開発者グループに IdM ユーザーを追加しました 。
- IdM クライアントに自動マウントを設定 している。

手順

1. IdM ユーザーが **read-write** ディレクトリーにアクセスできることを確認します。
   a. IdM ユーザーとして IdM クライアントに接続します。

   ```
   $ ssh idm_user@idm-client.idm.example.com
   Password:
   ```

   b. IdM ユーザーの Ticket Granting Ticket (TGT) を取得します。

   ```
   $ kinit idm_user
   ```

   c. [オプション] Id M ユーザーのグループメンバーシップを表示します。

   ```
   $ ipa user-show idm_user
     User login: idm_user
     [...]
     Member of groups: developers, ipausers
   ```

   d. **/devel/project** ディレクトリーに移動します。

   ```
   $ cd /devel/project
   ```

   e. ディレクトリーの内容をリスト表示します。

   ```
   $ ls
   rw_file
   ```

f. ディレクトリーのファイルに行を追加し、**write** パーミッションをテストします。

```
$ echo "idm_user can write into the file" > rw_file
```

g. [オプション] ファイルの更新された内容を表示します。

```
$ cat rw_file
this is a read-write file
idm_user can write into the file
```

出力は、**idm_user** がファイルに書き込めることを確認します。

第120章 IDM ログファイルおよびディレクトリー

以下のセクションを使用して、Identity Management (IdM) の個々のコンポーネントを監視、分析、およびトラブルシューティングします。

- LDAP
- Apache Web Server
- Certificate System
- Kerberos
- DNS
- custodia

さらに、IdM サーバーおよびクライアント の監視、分析、トラブルシューティングを行い、IdM サーバーの監査ロギングの有効化 ができます。

120.1. IDM サーバーおよびクライアントのログファイルおよびディレクトリー

以下の表は、Identity Management (IdM) サーバーおよびクライアントが情報のログ記録に使用するディレクトリーおよびファイルを示しています。インストールエラーのトラブルシューティングには、ファイルおよびディレクトリーを使用できます。

ディレクトリーまたはファイル	説明
/var/log/ipaserver-install.log	IdM サーバーのインストールログ。
/var/log/ipareplica-install.log	IdM レプリカのインストールログ
/var/log/ipaclient-install.log	IdM クライアントのインストールログ
/var/log/sssd/	SSSD のログファイル。sssd.conf ファイル または sssctl コマンド で SSSD の詳細ロギングを有効化できます。
~/.ipa/log/cli.log	**ipa** ユーティリティーによる応答、リモートプロシージャーコール (RPC) で返されるエラーのログファイル。ツールを実行する **実効ユーザー** のホームディレクトリーに作成されます。このユーザーは、IdM ユーザープリンシパルとはユーザー名が異なる可能性があります。(これは、**ipa** コマンドの実行を試みて失敗する前にチケット保証チケット (TGT) 取得した IdM ユーザーです。)たとえば、**root** でシステムにログインし、IdM **管理者** の TGT を取得している場合は、エラーが **/root/.ipa/log/cli.log** ファイルに記録されます。
/etc/logrotate.d/	DNS、SSSD、Apache、Tomcat、および Kerberos のログローテーションのポリシー

ディレクトリーまたはファイル	説明
/etc/pki/pki-tomcat/logging.properties	このリンクは、**/usr/share/pki/server/conf/logging.properties** でデフォルトの認証局ロギング設定を参照します。

関連情報

- IdM サーバーのインストールに関するトラブルシューティング
- IdM クライアントのインストールに関するトラブルシューティング
- IdM レプリカのインストールに関するトラブルシューティング
- IdM で SSSD を使用した認証のトラブルシューティング

120.2. DIRECTORY SERVER のログファイル

以下の表は、Identity Management (IdM) Directory Server (DS) インスタンスが情報をログ記録に使用するディレクトリーおよびファイルを示しています。DS 関連の問題のトラブルシューティングには、ファイルおよびディレクトリーを使用できます。

表120.1 Directory Server のログファイル

ディレクトリーまたはファイル	説明
/var/log/dirsrv/slapd-REALM_NAME/	IdM サーバーが使用する DS インスタンスに関連付けられたログファイル。ここに記録されるほとんどの運用データは、サーバーとレプリカの相互作用に関連しています。
/var/log/dirsrv/slapd-REALM_NAME/audit	DS 設定で監査を有効にした場合のすべての DS 操作の監査証跡が含まれます。 **注記** IdM API がアクセスを記録する Apache エラーログを監査することもできます。ただし、LDAP 経由で直接変更できるため、Red Hat は監査目的でより包括的な **/var/log/dirsrv/slapd-REALM_NAME/audit** ログを有効にすることを推奨します。
/var/log/dirsrv/slapd-REALM_NAME/access	ドメイン DS インスタンスの試行したアクセスに関する詳細情報が含まれています。
/var/log/dirsrv/slapd-REALM_NAME/errors	ドメイン DS インスタンスの失敗した操作に関する詳細情報が含まれます。

関連情報

- サーバーおよびデータベースアクティビティーの監視
- ログファイルのリファレンス

120.3. IDM サーバーでの監査ロギングの有効化

監査目的で Identity Management (IdM) サーバーでのロギングを有効にするには、次の手順に従います。詳細なログを使用すると、データの監視、問題のトラブルシューティング、ネットワーク上の疑わしいアクティビティーを確認できます。

注記

特に値が大きい場合など、多くの LDAP 変更がログに記録されている場合、LDAP サービスが遅くなることがあります。

前提条件

- Directory Manager のパスワード

手順

1. LDAP サーバーにバインドします。

```
$ ldapmodify -D "cn=Directory Manager" -W << EOF
```

2. [Enter] を押します。
3. 作成するすべての変更を指定します。以下に例を示します。

```
dn: cn=config
changetype: modify
replace: nsslapd-auditlog-logging-enabled
nsslapd-auditlog-logging-enabled: on
-
replace:nsslapd-auditlog
nsslapd-auditlog: /var/log/dirsrv/slapd-REALM_NAME/audit
-
replace:nsslapd-auditlog-mode
nsslapd-auditlog-mode: 600
-
replace:nsslapd-auditlog-maxlogsize
nsslapd-auditlog-maxlogsize: 100
-
replace:nsslapd-auditlog-logrotationtime
nsslapd-auditlog-logrotationtime: 1
-
replace:nsslapd-auditlog-logrotationtimeunit
nsslapd-auditlog-logrotationtimeunit: day
```

4. 新しい行で **EOF** を入力して、**ldapmodify** コマンドの最後を示します。
5. [Enter] を 2 回押します。
6. 監査ロギングを有効にする他のすべての IdM サーバーで直前の手順を繰り返します。

検証

- **/var/log/dirsrv/slapd-REALM_NAME/audit** ファイルを開きます。

```
389-Directory/1.4.3.231 B2021.322.1803
server.idm.example.com:636 (/etc/dirsrv/slapd-IDM-EXAMPLE-COM)

time: 20220607102705
dn: cn=config
result: 0
changetype: modify
replace: nsslapd-auditlog-logging-enabled
nsslapd-auditlog-logging-enabled: on
[...]
```

ファイルが空ではない場合には、監査が有効になっていることが分かります。

重要

システムは、変更を行うエントリーのバインドされた LDAP 識別名 (DN) をログに記録します。このため、ログを後処理する必要がある場合があります。たとえば、IdM Directory Server では、レコードを変更する AD ユーザーの ID を表す ID オーバーライド DN になります。

```
$ modifiersName: ipaanchoruuid=:sid:s-1-5-21-19610888-1443184010-1631745340-279100,cn=default trust view,cn=views,cn=accounts,dc=idma,dc=idm,dc=example,dc=com
```

SID ユーザーがある場合は、**pysss_nss_idmap.getnamebysid** Python コマンドを使用して AD ユーザーを検索します。

```
>>> import pysss_nss_idmap
>>> pysss_nss_idmap.getnamebysid('S-1-5-21-1273159419-3736181166-4190138427-500'))
{'S-1-5-21-1273159419-3736181166-4190138427-500': {'name': 'administrator@ad.vm', 'type': 3}}
```

関連情報

- Red Hat Directory Server ドキュメントの Core サーバー設定属性 の監査ログ設定オプション
- IPA/IDM サーバーおよびレプリカサーバーの KCS で監査ロギングを有効にする方法
- Directory Server のログファイル

120.4. IDM サーバーでのエラーログの変更

特定の種類のエラーに関するデバッグ情報を取得するには、次の手順に従います。この例では、エラーログレベルを 8192 に設定して、レプリケーションに関する詳細なエラーログを取得することに重点を置いています。異なるタイプの情報を記録するには、Red Hat Directory Server ドキュメントの Error Log Logging Levels の表から別の番号を選択します。

注記

特に値が大きい場合に、LDAP サービスが多数のタイプをログに記録すると、処理が遅くなる可能性があります。

前提条件

- Directory Manager のパスワード。

手順

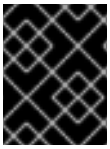
1. LDAP サーバーにバインドします。

```
$ ldapmodify -x -D "cn=directory manager" -w <password>
```

2. [Enter] を押します。
3. 変更する変更を指定します。たとえば、レプリケーションに関連するログのみを収集するには、以下を実行します。

```
dn: cn=config
changetype: modify
add: nsslapd-errorlog-level
nsslapd-errorlog-level: 8192
```

4. [Enter] を 2 回押して、**ldapmodify** 命令の最後を示します。これにより、**modifying entry "cn=config"** メッセージが表示されます。
5. [Ctrl+C] を押して **ldapmodify** コマンドを終了します。
6. レプリケーションエラーに関する詳細なログを収集する他のすべての IdM サーバーで直前の手順を繰り返します。



重要

トラブルシューティングが完了したら、**nsslapd-errorlog-level** を 0 に戻し、パフォーマンスの問題を防ぎます。

関連情報

- Directory Server エラーログレベル

120.5. IDM APACHE サーバーのログファイル

以下の表は、Identity Management (IdM) Apache Server が情報をログに記録するために使用するディレクトリーおよびファイルを示しています。

表120.2 Apache サーバーのログファイル

ディレクトリーまたはファイル	説明
/var/log/httpd/	Apache Web サーバーのログファイル。

ディレクトリーまたはファイル	説明
/var/log/httpd/access_log	これは、Apache サーバーの標準的なアクセスログおよびエラーログです。IdM Web UI および RPC コマンドラインインターフェイスは Apache を使用するため、IdM 固有のメッセージは Apache メッセージとともに記録されます。アクセスログは、主にユーザープリンシパルと使用される URI のみをログに記録します。多くの場合、これは RPC エンドポイントです。エラーログには IdM サーバーログが含まれます。
/var/log/httpd/error_log	

関連情報

- Apache ドキュメントの ログファイル

120.6. IDM の CERTIFICATE SYSTEM のログファイル

以下の表は、Identity Management (IdM) Certificate System が情報のログ記録に使用するディレクトリーおよびファイルを示しています。

表120.3 Certificate System のログファイル

ディレクトリーまたはファイル	説明
/var/log/pki/pki-ca-spawn.time_of_installation.log	IdM 認証局 (CA) のインストールログ。
/var/log/pki/pki-kra-spawn.time_of_installation.log	IdM Key Recovery Authority (KRA) のインストールログ。
/var/log/pki/pki-tomcat/	PKI 操作ログのトップレベルディレクトリー。CA ログおよび KRA ログが含まれます。
/var/log/pki/pki-tomcat/ca/	証明書の操作に関連するログを含むディレクトリー。IdM では、このログは証明書を使用するサービスプリンシパル、ホスト、およびその他のエンティティーに使用されます。
/var/log/pki/pki-tomcat/kra	KRA に関連するログを含むディレクトリー。
/var/log/messages	証明書のエラーメッセージがその他のシステムメッセージに含まれます。

関連情報

- Red Hat Certificate System **管理ガイド** の サブシステムのログの設定

120.7. IDM の KERBEROS ログファイル

以下の表は、Kerberos が Identity Management (IdM) に情報をログに記録するために使用するディレクトリーおよびファイルを示しています。

表120.4 Kerberos ログファイル

ディレクトリーまたはファイル	説明
/var/log/krb5kdc.log	これは、Kerberos KDC サーバーの主なログファイルです。
/var/log/kadmind.log	Kerberos 管理システムサーバーの主なログファイルです。
これらのファイルの場所は **krb5.conf** ファイルで設定されます。システムによっては異なる場合があります。	

120.8. IDM の DNS ログファイル

以下の表は、DNS が Identity Management (IdM) に情報をログに記録するために使用するディレクトリーおよびファイルを示しています。

表120.5 DNS ログファイル

ディレクトリーまたはファイル	説明
/var/log/messages	DNS エラーメッセージおよびその他のシステムメッセージが含まれます。このファイルの DNS ロギングは、デフォルトでは有効になりません。これを有効にするには、**# /usr/sbin/rndc querylog** コマンドを実行します。このコマンドを実行すると、次の行が **var/log/messages** に追加されます。 **Jun 26 17:37:33 r8server named-pkcs11[1445]: received control channel command 'querylog'** **Jun 26 17:37:33 r8server named-pkcs11[1445]: query logging is now on** ロギングを無効にするには、コマンドを再度実行します。

120.9. IDM の CUSTODIA ログファイル

以下の表は、Custodia が Identity Management (IdM) に情報をログに記録するために使用するディレクトリーおよびファイルを示しています。

表120.6 Custodia ログファイル

ディレクトリーまたはファイル	説明
/var/log/custodia/	Custodia サービスのログファイルディレクトリー。

120.10. 関連情報

- ログファイルの表示**journalctl** を使用すると、**systemd** ユニットファイルのログ出力を表示できます。

第121章 IDM ドメインで RHEL 8 WEB コンソールにシングルサインオンを設定

RHEL 8 Web コンソールでの Identity Management (IdM) が提供する SSO (シングルサインオン) 認証を使用する方法を学びます。

利点:

- IdM ドメインの管理者は、RHEL 8 Web コンソールを使用して、ローカルマシンを管理できます。
- IdM ドメインで Kerberos チケットを使用すると、Web コンソールにアクセスする際にログイン認証情報を指定する必要がなくなりました。
- IdM ドメインが認識しているすべてのホストは、RHEL 8 Web コンソールのローカルインスタンスから SSH 経由でアクセスできます。
- 証明書設定は必須ではありません。コンソールの Web サーバーでは、IdM 認証局が発行した証明書に自動的に切り替わり、ブラウザーに許可されます。

本章は、RHEL Web コンソールにログインするために SSO を設定する手順を説明します。

1. RHEL 8 Web コンソールを使用して IdM ドメインにマシンを追加します。
   詳細は、Joining a RHEL 8 system to an IdM domain using the web console を参照してください。
2. 認証に Kerberos を使用する場合は、マシンで Kerberos チケットを取得する必要があります。
   詳細は、Kerberos 認証を使用した Web コンソールへのログイン を参照してください。
3. IdM サーバーの管理者が、任意のホストで任意のコマンドを実行できます。
   詳細は、管理者の sudo で IdM サーバーのドメイン管理者にアクセス可能に を参照してください。

前提条件

- RHEL Web コンソールが RHEL 8 システムにインストールされている。
  詳細は、Web コンソールのインストールおよび有効化 を参照してください。
- RHEL Web コンソールを使用して IdM クライアントがシステムにインストールされている。
  詳細は IdM クライアントのインストール を参照してください。

121.1. WEB コンソールを使用した RHEL 8 システムの IDM ドメインへの参加

Web コンソールを使用して、Red Hat Enterprise Linux 8 システムを Identity Management (IdM) ドメインに参加させることができます。

前提条件

- IdM ドメインが実行中で参加するクライアントから到達可能
- IdM ドメインの管理者認証情報がある。

手順

1. RHEL Web コンソールにログインします。
詳細は、Web コンソールへのログイン を参照してください。
2. **Overview** タブの **Configuration** フィールドで、**Join Domain** をクリックします。
3. **ドメイン参加** ダイアログボックスの **ドメインアドレス** フィールドに、IdM サーバーのホスト名を入力します。
4. **ドメイン管理者名** フィールドで、IdM 管理アカウントのユーザー名を入力します。
5. **Domain administrator password** にパスワードを追加します。
6. **参加** をクリックします。

検証手順

1. システムが IdM ドメインに参加していると、RHEL 8 Web コンソールにエラーが表示されず、**システム** 画面でドメイン名を確認できます。
2. ユーザーがドメインのメンバーであることを確認するには、Terminal ページをクリックし、**id** コマンドを実行します。

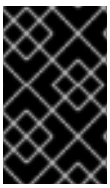
```
$ id
euid=548800004(example_user) gid=548800004(example_user)
groups=548800004(example_user) context=unconfined_u:unconfined_r:unconfined_t:s0-
s0:c0.c1023
```

関連情報

- Identity Management の計画
- Identity Management のインストール

121.2. KERBEROS 認証を使用して WEB コンソールにログイン

次の手順は、Kerberos 認証を使用するように RHEL 8 システムを設定する方法を説明します。



重要

SSO を使用した場合は、通常、Web コンソールに管理者権限がありません。これは、パスワードがない sudo を設定した場合に限り機能します。Web コンソールは、対話的に sudo パスワードを要求しません。

前提条件

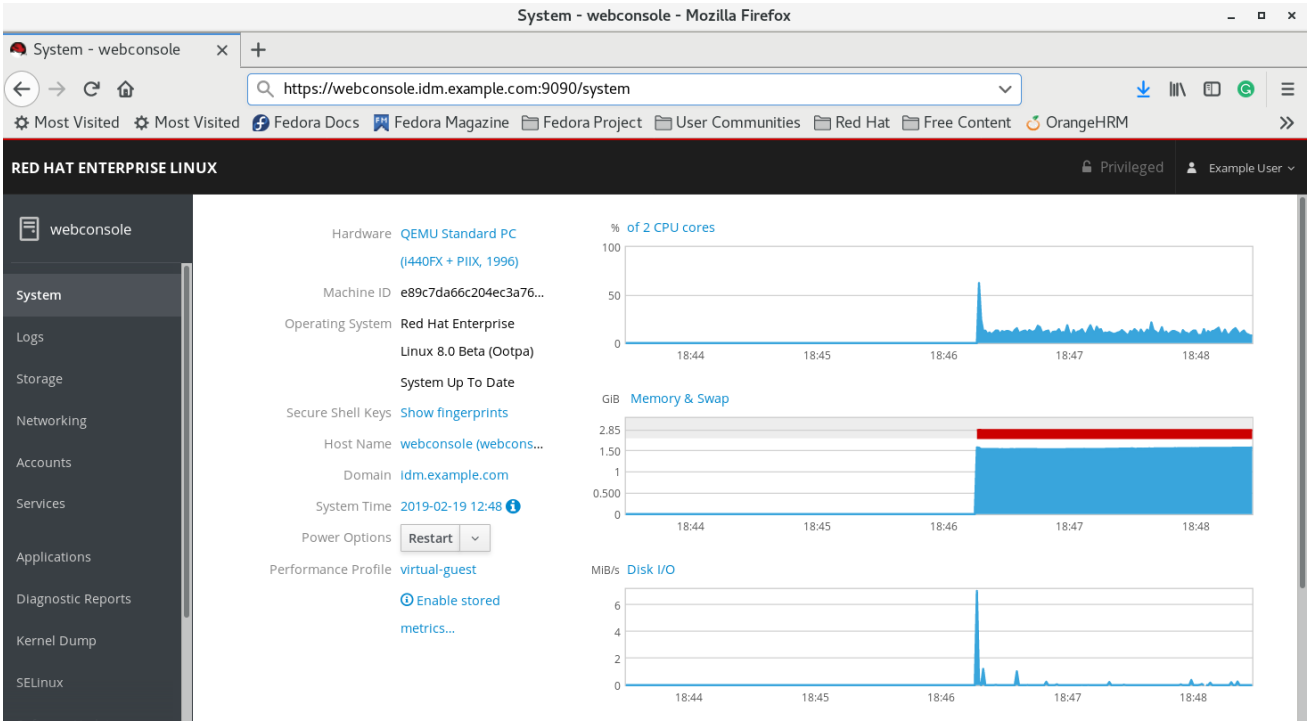
- 稼働中で、会社の環境で到達可能な IdM ドメイン
詳細は、Joining a RHEL 8 system to an IdM domain using the web console を参照してください。
- リモートシステムで、RHEL Web コンソールで接続して管理する **cockpit.socket** サービスを有効にしている。
詳細は、Web コンソールのインストール を参照してください。

- システムが、SSSD クライアントが管理する Kerberos チケットを使用しない場合は、**kinit** ユーティリティーを使用して手動でチケットを要求してみる。

手順

https://dns_name:9090 から、RHEL Web コンソールにログインします。

この時点で、RHEL Web コンソールへの接続に成功しており、設定を開始できます。



121.3. 管理者の SUDO で IDM サーバーのドメイン管理者にアクセス可能に

RHEL Web コンソールを使用すると、ドメイン管理者がアイデンティティー Management (IdM) ドメイン内の任意のホストで任意のコマンドを使用できるようにすることができます。

これを可能にするために、IdM サーバーのインストール時に自動的に作成された **admins** ユーザーグループに sudo がアクセスできるようにします。グループで **ipa-advise** スクリプトを実行すると、admins グループに追加したすべてのユーザーに sudo アクセスが付与されます。

前提条件

- サーバーが、IdM 4.7.1 以降を実行している。

手順

1. IdM サーバーに接続します。
2. ipa-advise スクリプトを実行します。

```
$ ipa-advise enable-admins-sudo | sh -ex
```

コンソールにエラーが表示されない場合、**admins** グループには IdM ドメイン内のすべてのマシンに対する sudo 権限があります。

第122章 IDM での制約付き委任の使用

Identity Management (IdM) で制約付き委任機能を使用する方法を説明します。

- Identity Managementの制約付き委任 は、制約付き委任がどのように機能するかを説明しています。
- スマートカードで認証されたユーザーが再認証を求められることなくリモートホストに SSH 接続できるように Web コンソールを設定する では、認証を必要とせずに、Red Hat Enterprise Linux Web コンソールを使用してリモートホストに **SSH 接続** するコンテキストでの制約付き委任のユースケースについて説明します。
- 認証を必要とせずに、Ansible を使用して スマートカードで認証されたユーザーが再認証を求められることなくリモートホストに SSH 接続できるように Web コンソールを設定する では、Red Hat Enterprise Linux Web コンソールを使用してリモートホストに **SSH** 接続するコンテキストでの制約付き委任のユースケースについて説明します。
- スマートカードで認証されたユーザーが認証を求められずに sudo を実行できるように Web コンソールクライアントを設定 では、Red Hat Enterprise Linux Web コンソールを使用して認証を必要とせずに **sudo** を実行するコンテキストでの制約付き委任のユースケースについて説明します。
- Ansible を使用して Web コンソールを設定し、スマートカードで認証されたユーザーが再認証を求められることなく sudo を実行できるようにする では、Ansible を使用して Red Hat Enterprise Linux Web コンソールの使用を設定するコンテキストで、認証を必要とせずに **sudo** を実行する制約付き委任のユースケースについて説明します。

122.1. アイデンティティー管理における制約付き委任

Service for User to Proxy (**S4U2proxy**) 拡張機能は、ユーザーに代わって他のサービスに対するサービスチケットを取得するサービスを提供します。この機能は、**制約付き委任と** 呼ばれています。2 番目のサービスは通常、ユーザーの承認コンテキストの下で、最初のサービスに代わって何らかの作業を実行するプロキシーです。制約付き委任を使用することで、ユーザーが Ticket Granting Ticket (TGT) を完全に委任する必要がなくなります。

Identity Management (IdM) は従来、Kerberos **S4U2proxy** 機能を使用して、Web サーバーフレームワークがユーザーの代わりに LDAP サービスチケットを取得することを可能にするものです。また、IdM-AD の信頼システムも、**cifs** プリンシパルを取得するために制約付き委任を使用しています。

S4U2proxy 機能を使用して Web コンソールクライアントを設定し、スマートカードで認証された IdM ユーザーが以下を達成できるようにすることができます。

- Web コンソールサービスが実行されている RHEL ホストで、再度認証を求められることなく、スーパーユーザー権限でコマンドを実行します。
- **SSH** を使用してリモートホストにアクセスし、再度認証を求められることなくホスト上のサービスにアクセスします。

関連情報

- Ansible を使用して Web コンソールを設定し、スマートカードで認証されたユーザーが再認証を求められることなくリモートホストに SSH 接続できるようにする
- Ansible を使用して Web コンソールを設定し、スマートカードで認証されたユーザーが再認証を求められることなく sudo を実行できるようにする

- S4U2proxy
- サービスの制約付き委任

122.2. スマートカードで認証されたユーザーが、再度認証を要求されることなくリモートホストに SSH 接続できるようにするための WEB コンソールの設定

RHEL の Web コンソールでユーザーアカウントにログインした後、Identity Management (IdM) システム管理者として、**SSH** プロトコルを使用してリモートマシンに接続する必要がある場合があります。制約付き委任 機能を使用すると、再度認証を求められることなく **SSH** を使用することができます。

制約付き委任を使用するように Web コンソールを設定するには、次の手順に従います。以下の例では、Web コンソールセッションは **myhost.idm.example.com** ホストで実行され、認証されたユーザーの代わりに **SSH** を使用して **remote.idm.example.com** ホストにアクセスするように設定されています。

前提条件

- IdM **admin** Ticket-Granting Ticket (TGT) を取得している
- **remote.idm.example.com** への **root** アクセス権がある
- Web コンソールサービスが IdM に存在する
- **remote.idm.example.com** ホストが IdM に存在する
- Web コンソールは、ユーザーセッションに **S4U2Proxy** Kerberos チケットを作成している。これを確認するために、IdM ユーザーで Web コンソールにログインし、**Terminal** ページを開き、以下を入力します。

```
$ klist
Ticket cache: FILE:/run/user/1894000001/cockpit-session-3692.ccache
Default principal: user@IDM.EXAMPLE.COM

Valid starting     Expires            Service principal
07/30/21 09:19:06 07/31/21 09:19:06
HTTP/myhost.idm.example.com@IDM.EXAMPLE.COM
07/30/21 09:19:06  07/31/21 09:19:06  krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
        for client HTTP/myhost.idm.example.com@IDM.EXAMPLE.COM
```

手順

1. 委任ルールでアクセス可能な対象ホストのリストを作成します。

   a. サービス委任ターゲットを作成します。

   ```
   $ ipa servicedelegationtarget-add cockpit-target
   ```

   b. 委任対象に対象ホストを追加します。

   ```
   $ ipa servicedelegationtarget-add-member cockpit-target \ --
   principals=host/remote.idm.example.com@IDM.EXAMPLE.COM
   ```

2. サービス委任ルールを作成し、**HTTP** サービスの Kerberos プリンシパルを追加することで、**cockpit** セッションが対象ホストのリストにアクセスできるようにします。

   a. サービス委任ルールを作成します。

   ```
   $ ipa servicedelegationrule-add cockpit-delegation
   ```

   b. Web コンソールクライアントを委任ルールに追加します。

   ```
   $ ipa servicedelegationrule-add-member cockpit-delegation \ --
   principals=HTTP/myhost.idm.example.com@IDM.EXAMPLE.COM
   ```

   c. 委任対象を委任ルールに追加します。

   ```
   $ ipa servicedelegationrule-add-target cockpit-delegation \ --
   servicedelegationtargets=cockpit-target
   ```

3. **remote.idm.example.com** ホストで Kerberos 認証を有効にします。

   a. **root** として **remote.idm.example.com** に **SSH** 接続します。

   b. **/etc/ssh/sshd_config** ファイルを開いて編集します。

   c. **GSSAPIAuthentication no** 行のコメントを外し、**GSSAPIAuthentication yes** に置き換えて、**GSSAPIAuthentication** を有効にします。

4. 上記の変更がすぐに有効になるように、**remote.idm.example.com** の **SSH** サービスを再起動します。

   ```
   $ systemctl try-restart sshd.service
   ```

関連情報

- スマートカードを使用して Web コンソールへのログイン
- アイデンティティー管理における制約付き委任

122.3. ANSIBLE を使用して WEB コンソールを設定し、スマートカードで認証されたユーザーが再認証を求められることなくリモートホストに SSH 接続できるようにする

RHEL の Web コンソールでユーザーアカウントにログインした後、Identity Management (IdM) システム管理者として、**SSH** プロトコルを使用してリモートマシンに接続する必要がある場合があります。制約付き委任 機能を使用すると、再度認証を求められることなく **SSH** を使用することができます。

servicedelegationrule および **servicedelegationtarget ansible-freeipa** モジュールを使用して、制約付き委任を使用するように Web コンソールを設定するには、この手順に従います。以下の例では、Web コンソールセッションは **myhost.idm.example.com** ホストで実行され、認証されたユーザーの代わりに **SSH** を使用して **remote.idm.example.com** ホストにアクセスするように設定されています。

前提条件

- IdM **admin** パスワードがある
- **remote.idm.example.com** への **root** アクセスがある
- Web コンソールサービスが IdM に存在する
- **remote.idm.example.com** ホストが IdM に存在する
- Web コンソールは、ユーザーセッションに **S4U2Proxy** Kerberos チケットを作成している。これを確認するために、IdM ユーザーで Web コンソールにログインし、**Terminal** ページを開き、以下を入力します。

```
$ klist
Ticket cache: FILE:/run/user/1894000001/cockpit-session-3692.ccache
Default principal: user@IDM.EXAMPLE.COM

Valid starting     Expires            Service principal
07/30/21 09:19:06 07/31/21 09:19:06
HTTP/myhost.idm.example.com@IDM.EXAMPLE.COM
07/30/21 09:19:06  07/31/21 09:19:06  krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
        for client HTTP/myhost.idm.example.com@IDM.EXAMPLE.COM
```

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。
- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. **~/MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. 以下の内容で **web-console-smart-card-ssh.yml** Playbook を作成します。
   a. 委任対象の存在を確認するタスクを作成します。

```
---
- name: Playbook to create a constrained delegation target
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure servicedelegationtarget web-console-delegation-target is present
```

```
    ipaservicedelegationtarget:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: web-console-delegation-target
```

b. 対象ホストを委任ターゲットに追加するタスクを追加します。

```
  - name: Ensure servicedelegationtarget web-console-delegation-target member
principal host/remote.idm.example.com@IDM.EXAMPLE.COM is present
    ipaservicedelegationtarget:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: web-console-delegation-target
      principal: host/remote.idm.example.com@IDM.EXAMPLE.COM
      action: member
```

c. 委任ルールの存在を確認するタスクを追加します。

```
  - name: Ensure servicedelegationrule delegation-rule is present
    ipaservicedelegationrule:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: web-console-delegation-rule
```

d. Web コンソールクライアントサービスの Kerberos プリンシパルが制約付き委任ルールのメンバーであることを確認するタスクを追加します。

```
  - name: Ensure the Kerberos principal of the web console client service is added to the
servicedelegationrule web-console-delegation-rule
    ipaservicedelegationrule:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: web-console-delegation-rule
      principal: HTTP/myhost.idm.example.com
      action: member
```

e. 制約付き委任ルールが web-console-delegation-target 委任対象と関連付けられることを確認するタスクを追加します。

```
  - name: Ensure a constrained delegation rule is associated with a specific delegation
target
    ipaservicedelegationrule:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: web-console-delegation-rule
      target: web-console-delegation-target
      action: member
```

3. ファイルを保存します。

4. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory web-console-
smart-card-ssh.yml
```

5. **remote.idm.example.com** で Kerberos 認証を有効にします。

a. **root** として **remote.idm.example.com** に **SSH** 接続します。

b. **/etc/ssh/sshd_config** ファイルを開いて編集します。

c. **GSSAPIAuthentication no** 行のコメントを外し、**GSSAPIAuthentication yes** に置き換えて、**GSSAPIAuthentication** を有効にします。

関連情報

- スマートカードを使用して Web コンソールへのログイン
- アイデンティティー管理における制約付き委任
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-servicedelegationrule.md** および **README-servicedelegationtarget.md**
- **/usr/share/doc/ansible-freeipa/playbooks/servicedelegationtarget** および **/usr/share/doc/ansible-freeipa/playbooks/servicedelegationrule** ディレクトリーのサンプル Playbook

122.4. スマートカードで認証されたユーザーが再認証を求められることなく SUDO を実行できるように WEB コンソールを設定する

RHEL Web コンソールでユーザーアカウントにログインした後、Identity Management (IdM) システム管理者として、スーパーユーザー権限でコマンドを実行することが必要になる場合があります。制約付き委任 機能を使用すると、再度認証を求められることなく、システムで **sudo** を実行できます。

制約付き委任を使用するように Web コンソールを設定するには、次の手順に従います。以下の例では、Web コンソールセッションは **myhost.idm.example.com** ホストで実行されます。

前提条件

- IdM **admin** Ticket-Granting Ticket (TGT) を取得済みである。
- Web コンソールサービスが IdM に存在する
- **myhost.idm.example.com** ホストは IdM に存在します。
- IdM サーバーのドメイン管理者への **admin sudo** アクセスを有効 にしている。
- Web コンソールは、ユーザーセッションに **S4U2Proxy** Kerberos チケットを作成している。これを確認するために、IdM ユーザーで Web コンソールにログインし、**Terminal** ページを開き、以下を入力します。

```
$ klist
Ticket cache: FILE:/run/user/1894000001/cockpit-session-3692.ccache
Default principal: user@IDM.EXAMPLE.COM

Valid starting     Expires            Service principal
07/30/21 09:19:06 07/31/21 09:19:06
HTTP/myhost.idm.example.com@IDM.EXAMPLE.COM
07/30/21 09:19:06  07/31/21 09:19:06  krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
        for client HTTP/myhost.idm.example.com@IDM.EXAMPLE.COM
```

手順

1. 委任ルールでアクセス可能な対象ホストのリストを作成します。

a. サービス委任ターゲットを作成します。

```
$ ipa servicedelegationtarget-add cockpit-target
```

b. 委任対象に対象ホストを追加します。

```
$ ipa servicedelegationtarget-add-member cockpit-target \ --
principals=host/myhost.idm.example.com@IDM.EXAMPLE.COM
```

2. サービス委任ルールを作成し、**HTTP** サービスの Kerberos プリンシパルを追加することで、**cockpit** セッションが対象ホストのリストにアクセスできるようにします。

a. サービス委任ルールを作成します。

```
$ ipa servicedelegationrule-add cockpit-delegation
```

b. Web コンソールサービスを委任ルールに追加します。

```
$ ipa servicedelegationrule-add-member cockpit-delegation \ --
principals=HTTP/myhost.idm.example.com@IDM.EXAMPLE.COM
```

c. 委任対象を委任ルールに追加します。

```
$ ipa servicedelegationrule-add-target cockpit-delegation \ --
servicedelegationtargets=cockpit-target
```

3. System Security Services Daemon (SSSD) と連携して Generic Security Service Application Program Interface (GSSAPI) を介してユーザーを認証するための PAM モジュールである **pam_sss_gss** を有効にします。

a. **/etc/sssd/sssd.conf** ファイルを開いて編集します。

b. ドメインの IdM で **pam_sss_gss** が **sudo** および **sudo -i** コマンドの認証を提供できるように指定します。

```
[domain/idm.example.com]
pam_gssapi_services = sudo, sudo-i
```

c. ファイルを保存し、終了します。

d. **/etc/pam.d/sudo** ファイルを編集用に開きます。

e. 次の行を **#%PAM-1.0** リストの先頭に挿入して、**sudo** コマンドの GSSAPI 認証を許可しますが、必須ではありません。

```
auth sufficient pam_sss_gss.so
```

f. ファイルを保存し、終了します。

4. 上記の変更がすぐに有効になるように、**SSSD** サービスを再起動します。

```
$ systemctl restart sssd
```

関連情報

- スマートカードを使用して Web コンソールへのログイン
- アイデンティティー管理における制約付き委任

122.5. ANSIBLE を使用して WEB コンソールを設定し、スマートカードで認証されたユーザーが再認証を求められることなく SUDO を実行できるようにする

RHEL Web コンソールでユーザーアカウントにログインした後、Identity Management (IdM) システム管理者として、スーパーユーザー権限でコマンドを実行することが必要になる場合があります。制約付き委任 機能を使用すると、再度認証を求められることなく、システムで **sudo** を実行できます。

この手順に従って、**ipaservicedelegationrule** モジュールおよび **ipaservicedelegationtarget ansible-freeipa** モジュールを使用して、制約付き委任を使用するように Web コンソールを設定します。以下の例では、Web コンソールセッションは **myhost.idm.example.com** ホストで実行されます。

前提条件

- スマートカードを使用して Web コンソールセッションを認証することにより、IdM **admin** のチケット認可チケット (TGT) を取得しました。
- Web コンソールサービスが IdM に登録されました。
- **myhost.idm.example.com** ホストは IdM に存在します。
- IdM サーバーのドメイン管理者への **admin sudo** アクセスを有効 にしている。
- Web コンソールは、ユーザーセッションに **S4U2Proxy** Kerberos チケットを作成している。これを確認するために、IdM ユーザーで Web コンソールにログインし、**Terminal** ページを開き、以下を入力します。

```
$ klist
Ticket cache: FILE:/run/user/1894000001/cockpit-session-3692.ccache
Default principal: user@IDM.EXAMPLE.COM

Valid starting     Expires            Service principal
07/30/21 09:19:06 07/31/21 09:19:06
HTTP/myhost.idm.example.com@IDM.EXAMPLE.COM
07/30/21 09:19:06  07/31/21 09:19:06  krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
        for client HTTP/myhost.idm.example.com@IDM.EXAMPLE.COM
```

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例は、**~/MyPlaybooks/** ディレクトリーに、制約付き委任を設定する IdM サーバーの完全修飾ドメイン名 (FQDN) を含む Ansible インベントリーファイル を作成したことを前提としています。
  - この例では、**secret.yml** Ansible ボールトに **ipaadmin_password** が保存されていることを前提としている。

- **ansible-freeipa** モジュールが実行されるノードであるターゲットノードは、IdM クライアント、サーバー、またはレプリカとしての IdM ドメインの一部です。

手順

1. Ansible コントロールノードで、**~/MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks/
```

2. 以下の内容で **web-console-smart-card-sudo.yml** Playbook を作成します。

a. 委任対象の存在を確認するタスクを作成します。

```
---
- name: Playbook to create a constrained delegation target
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure servicedelegationtarget named sudo-web-console-delegation-target is
present
    ipaservicedelegationtarget:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: sudo-web-console-delegation-target
```

b. 対象ホストを委任ターゲットに追加するタスクを追加します。

```
  - name: Ensure that a member principal named
host/myhost.idm.example.com@IDM.EXAMPLE.COM is present in a service delegation
target named sudo-web-console-delegation-target
    ipaservicedelegationtarget:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: sudo-web-console-delegation-target
      principal: host/myhost.idm.example.com@IDM.EXAMPLE.COM
      action: member
```

c. 委任ルールの存在を確認するタスクを追加します。

```
  - name: Ensure servicedelegationrule named sudo-web-console-delegation-rule is
present
    ipaservicedelegationrule:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: sudo-web-console-delegation-rule
```

d. Web コンソールサービスの Kerberos プリンシパルが制約付き委任ルールのメンバーであることを確認するタスクを追加します。

```
  - name: Ensure the Kerberos principal of the web console service is added to the
service delegation rule named sudo-web-console-delegation-rule
    ipaservicedelegationrule:
      ipaadmin_password: "{{ ipaadmin_password }}"
```

```
    name: sudo-web-console-delegation-rule
    principal: HTTP/myhost.idm.example.com
    action: member
```

e. 制約付き委任ルールが sudo-web-console-delegation-target 委任対象と関連付けられることを確認するタスクを追加します。

```
 - name: Ensure a constrained delegation rule is associated with a specific delegation
target
   ipaservicedelegationrule:
     ipaadmin_password: "{{ ipaadmin_password }}"
     name: sudo-web-console-delegation-rule
     target: sudo-web-console-delegation-target
     action: member
```

3. ファイルを保存します。

4. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory web-console-
smart-card-sudo.yml
```

5. System Security Services Daemon (SSSD) と連携して Generic Security Service Application Program Interface (GSSAPI) を介してユーザーを認証するための PAM モジュールである **pam_sss_gss** を有効にします。

   a. **/etc/sssd/sssd.conf** ファイルを開いて編集します。

   b. ドメインの IdM で **pam_sss_gss** が **sudo** および **sudo -i** コマンドの認証を提供できるように指定します。

```
[domain/idm.example.com]
pam_gssapi_services = sudo, sudo-i
```

   c. ファイルを保存し、終了します。

   d. **/etc/pam.d/sudo** ファイルを編集用に開きます。

   e. 次の行を **#%PAM-1.0** リストの先頭に挿入して、**sudo** コマンドの GSSAPI 認証を許可しますが、必須ではありません。

```
auth sufficient pam_sss_gss.so
```

   f. ファイルを保存し、終了します。

6. 上記の変更がすぐに有効になるように、**SSSD** サービスを再起動します。

```
$ systemctl restart sssd
```

関連情報

- アイデンティティー管理における制約付き委任

- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-servicedelegationrule.md** および **README-servicedelegationtarget.md**
- **/usr/share/doc/ansible-freeipa/playbooks/servicedelegationtarget** および **/usr/share/doc/ansible-freeipa/playbooks/servicedelegationrule** ディレクトリーのサンプル Playbook

122.6. 関連情報

- Web コンソールでリモートシステムの管理