



Red Hat Enterprise Linux 7

システムレベルの認証ガイド

認証および Identity Management に関するシステムレベルのサービス

Red Hat Enterprise Linux 7 システムレベルの認証ガイド

認証および Identity Management に関するシステムレベルのサービス

Filip Hanzelka

Red Hat Customer Content Services

fhanzelk@redhat.com

Lucie Maňásková

Red Hat Customer Content Services

lmanasko@redhat.com

Aneta Šteflová Petrová

Red Hat Customer Content Services

Marc Muehlfeld

Red Hat Customer Content Services

Tomáš Čapek

Red Hat Customer Content Services

Ella Deon Ballard

Red Hat Customer Content Services

法律上の通知

Copyright © 2018 Red Hat, Inc.

This document is licensed by Red Hat under the [Creative Commons Attribution-ShareAlike 3.0 Unported License](#). If you distribute this document, or a modified version of it, you must provide attribution to Red Hat, Inc. and provide a link to the original. If the document is modified, all Red Hat trademarks must be removed.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux ® is the registered trademark of Linus Torvalds in the United States and other countries.

Java ® is a registered trademark of Oracle and/or its affiliates.

XFS ® is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL ® is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js ® is an official trademark of Joyent. Red Hat Software Collections is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack ® Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

概要

本ガイドでは、ローカルシステム上で認証設定に利用可能な様々なアプリケーションとサービスを扱っています。Red Hat Enterprise Linux Identity Management に関する機能およびサービスについての資料は、本ガイドの他に以下のガイドがあります。Linux ドメイン ID、認証、およびポリシーガイドでは、Linux ベースのドメイン内における認証および承認ポリシーに加え、ID ストアを集中管理するソリューションである Red Hat Identity Management について説明しています。

Windows 統合ガイドでは、Identity Management を使って Linux ドメインと Microsoft Windows Active Directory (AD) を統合する方法について説明しています。この他にも、直接および間接的 AD 統合の様々な側面、SSSD を使って Common Internet File System (CIFS) にアクセスする方法、そして realmd システムなどについて説明しています。

目次

第1章 システム認証について	4
1.1. ユーザー ID の確認	4
1.2. シングルサインオンのプランニング	5
1.3. 利用可能なサービス	5
パート I. システムログイン	7
第2章 システム認証の設定	8
2.1. システム認証用の IDENTITY MANAGEMENT ツール	8
2.2. AUTHCONFIG の使用	8
第3章 AUTHCONFIG を使用して認証用に ID ストアを選択する手順	14
3.1. IPA V2	14
3.2. LDAP と IDM	16
3.3. NIS	19
3.4. WINBIND	21
第4章 認証メカニズムの設定	25
4.1. AUTHCONFIG を使用したローカル認証の設定	25
4.2. AUTHCONFIG を使用したシステムパスワードの設定	27
4.3. AUTHCONFIG を使用した KERBEROS (LDAP または NIS 認証) の設定	31
4.4. スマートカード	34
4.5. ワンタイムパスワード	41
4.6. AUTHCONFIG を使用した指紋の設定	41
第5章 AUTHCONFIG を使用したキックスタートと設定ファイルの管理	44
第6章 AUTHCONFIG を使用したカスタムホームディレクトリーの有効化	45
パート II. ID と認証ストア	48
第7章 SSSD の設定	49
7.1. SSSD について	49
7.2. クライアントごとに複数の SSSD 設定ファイルを使用	50
7.3. SSSD 向け ID プロバイダーと認証プロバイダーの設定	50
7.4. アイデンティティプロバイダーと認証プロバイダー向け追加的設定	56
7.5. SSSD のシステムサービスの設定	62
7.6. SSSD クライアント側のビュー	67
7.7. SSSD のダウングレード	70
7.8. SSSD と NSCD の使用	70
7.9. その他のリソース	70
第8章 REALMD を使った ID ドメインへの接続	72
第9章 LDAP サーバー	73
9.1. RED HAT DIRECTORY SERVER	73
9.2. OPENLDAP	73
パート III. セキュアなアプリケーション	93
第10章 PAM (プラグ可能な認証モジュール) の使用	94
10.1. PAM について	94
10.2. PAM 設定ファイルについて	94
10.3. PAM と管理認証情報のキャッシング	99
10.4. PAM サービスのドメイン制限	100

第11章 KERBEROS の使用	103
11.1. KERBEROS について	103
11.2. KERBEROS KDC の設定	109
11.3. KERBEROS クライアントの設定	114
11.4. スマートカード用の KERBEROS クライアントの設定	116
11.5. レルム間 KERBEROS 信頼の設定	117
第12章 CERTMONGER を使った作業	123
12.1. CERTMONGER と認証局	123
12.2. CERTMONGER での自己署名証明書のリクエスト	123
12.3. SCEP での CA 署名の証明書のリクエスト	124
12.4. NSS データベースでの証明書の保存	126
12.5. CERTMONGER を使った証明書の追跡	127
第13章 アプリケーションをシングルサインオン向けに設定	128
13.1. FIREFOX でシングルサインオンに KERBEROS を使用する設定	128
13.2. FIREFOX での証明書管理	129
13.3. メールクライアントの証明書管理	132
付録A トラブルシューティング	136
A.1. SSSD のトラブルシューティング	136
A.2. SSSD での SUDO のトラブルシューティングと SUDO のデバッグログ	146
A.3. FIREFOX の KERBEROS 設定のトラブルシューティング	148
付録B 改訂履歴	150

第1章 システム認証について

セキュアなネットワーク環境を確立するための第一歩は、ネットワークへのアクセス権限を持つユーザーにアクセスを限定することです。アクセスが許可されるとユーザーはシステムに対して **認証** することができます。つまり、ユーザー自身の ID を実証できることになります。

Red Hat Enterprise Linux システムでは、ユーザー ID の作成や識別に利用可能な異なるサービスが多数あります。このようなサービスには、ローカルシステムファイルのほか、**Kerberos** または **Samba** などの大型の ID ドメインに接続するサービス、あるいはこのようなドメインを作成するツールなどがあります。

本ガイドでは、ローカルシステムで管理者が認証およびアイデンティティの管理に利用できる一般的なシステムサービスとアプリケーションについて説明しています。「[creating Linux domains](#)」や「[integrating a Linux system into a Windows domain](#)」についての詳細情報は、個別のガイドが用意されています。

1.1. ユーザー ID の確認

認証 とは、ID を確認するプロセスのことです。ネットワークの対話では、一方が他方を特定することで認証が行われます。ネットワーク上での認証には多くの方法があります。簡単なパスワードや証明書、ワンタイムパスワード (OTP) トークンや生体認証スキャンなどです。

一方で **承認** では、認証された関係者が許可される動作やアクセスを定義します。

認証においては、ユーザーが自身の ID を証明するためになんらかの **認証情報** を提示することが求められます。求められる認証情報の種類は、使用される認証メカニズムによって定義されます。システム上のローカルユーザーは、以下のような認証が利用できます。

- **パスワードベースの認証** ほとんどすべてのソフトウェアでは、ユーザーが提供する名前とパスワードによる認証を許可しています。これは **簡易認証** とも呼ばれます。
- **証明書ベースの認証** 証明書をベースとしたクライアントの認証は、**SSL** プロトコルの一部です。クライアントはランダムに生成されたデータにデジタル処理で署名し、証明書と証明済みのデータをネットワーク経由で送信します。サーバーは署名を確認して証明書の有効性を確認します。
- **Kerberos 認証** Kerberos は **ticket-granting tickets (TGT)** と呼ばれる短期の認証情報システムを確立します。ユーザーがユーザー名とパスワードという認証情報を提示することでユーザーが特定され、このユーザーにチケットを発行可能であることをシステムに対して示します。TGT はその後、**Web** サイトや電子メールなどの他のサービスへのアクセスチケットを要求する際に繰り返し使用することができます。このように TGT を使用した認証では、ユーザーの認証プロセスは一度で済みます。
- **スマートカードベースの認証** これは証明書ベースの認証とわずかに異なるものです。スマートカード (または トークン) にはユーザーの証明書が保存されています。ユーザーがトークンをシステムに挿入すると、システムは証明書を読み取り、アクセスを許可することができます。スマートカードを使ったシングルサインオンには、以下の 3 つのステップがあります。
 1. ユーザーがスマートカードをカードリーダーに差し込みます。**PAM** (プラグ可能な認証モジュール) が差し込まれたスマートカードを検出します。
 2. システムが証明書をユーザーエントリーにマッピングし、スマートカード上で提示された証明書とユーザーエントリーで保存されている証明書を比較します。前者は、証明書ベースの認証で説明されている秘密鍵で暗号化されています。

3. 証明書がキー配布センター (KDC) に対して正常に確認されると、ユーザーはログインを許可されます。

スマートカードベースの認証は、Kerberos によって確立された簡易認証の層に物理的アクセス要件と認証情報を新たな識別メカニズムとして追加することで構築されます。

1.2. シングルサインオンのプランニング

「[ユーザー ID の確認](#)」で説明した認証では、セキュアなアプリケーションにアクセスするには最低でもパスワードが必要になります。中央の ID ストアがない場合や、各アプリケーションが独自にユーザーと認証情報を維持している場合に、ユーザーはサービスやアプリケーションを開くたびにパスワードの入力を求められます。こうなると一日に何度も、場合によっては数分ごとにパスワードの入力が必要になる可能性があります。

複数のパスワードを維持してそれらを何度も入力することは、ユーザーおよび管理者にとって大変な負担です。シングルサインオンを使うと、管理者は単一のパスワードストアを作成できるので、ユーザーは単一のパスワードを使ってログインして、すべてのネットワークリソースに認証されることが可能になります。

Red Hat Enterprise Linux では、ワークステーションへのログインやスクリーンセーバーの解除、Mozilla Firefox を使った安全なウェブページへのアクセスなど、複数のリソースに対してシングルサインオンをサポートしています。PAM、NSS、および Kerberos など、他の利用可能なシステムサービスを使うと、他のシステムアプリケーションもこれらの ID ソースを使用するように設定できます。

シングルサインオンはユーザーの利便性を高めると同時に、サーバーおよびネットワークの新たなセキュリティ層の役割も果たします。シングルサインオンはセキュアで効果的な認証の要所となります。Red Hat Enterprise Linux では、シングルサインオンを有効にする以下の 2 つの認証メカニズムを提供しています。

- Kerberos レalmと Active Directory ドメインの両方を使った Kerberos ベースの認証
- スマートカードベースの認証

これらのメカニズムは両方とも (Kerberos レalmまたは公開鍵インフラストラクチャーの認証局により) 中央 ID ストアを作成します。ローカルシステムのサービスは複数のローカルストアを維持するのではなく、これらの ID ドメインを使用します。

1.3. 利用可能なサービス

すべての Red Hat Enterprise Linux システムには、ローカルシステム上のローカルユーザーの認証が設定可能なサービスがあります。以下のものが含まれます。

認証セットアップ

- 認証設定ツール (**authconfig**) はシステム用に異なる ID バックエンドと認証方法 (パスワードや指紋、スマートカードなど) をセットアップします。

ID バックエンドセットアップ

- Security System Services Daemon (SSSD) は複数のアイデンティティプロバイダー (主に Microsoft Active Directory や Red Hat Enterprise Linux IdM などの LDAP ベースのディレクトリ) をセットアップし、これをローカルシステムとアプリケーションの両方のユーザーに使用することができます。パスワードとチケットはキャッシュされるので、認証情報を再利用してオフライン認証とシングルサインオンの両方が可能になります。
- **realmd** サービスはコマンドラインユーティリティーで、IdM 用の SSSD である認証バック

エンドの設定を可能にします。**realmd** サービスは **DNS** レコードに基づいて利用可能な **IdM** ドメインを検出し、**SSSD** を設定してからドメインのアカウントとしてシステムに参加します。

- **Name Service Switch (NSS)** は、ユーザー、グループ、またはホストの情報を返す低レベルのシステムコール用のメカニズムです。**NSS** は、必要な情報を取得するためにどのソース、つまりどのモジュールを使用すべきか判断します。たとえば、ユーザー情報は **/etc/passwd** ファイルなどの従来の **UNIX** ファイルか **LDAP** ベースのディレクトリーで見つかりますが、ホストアドレスは **/etc/hosts** ファイルなどのファイルか **DNS** レコードから読み取ります。**NSS** は情報の格納場所を見つけます。

認証メカニズム

- **PAM** (プラグ可能な認証モジュール) は、認証ポリシーをセットアップするシステムを提供します。認証に **PAM** を使用するアプリケーションは、認証における異なる要素を制御する異なるモジュールを読み込みます。アプリケーションがどの **PAM** モジュールを使用するかは、そのアプリケーションの設定方法に基づきます。利用可能な **PAM** には、**Kerberos**、**Winbind**、ローカルの **UNIX** ファイルベースの認証などがあります。

他のサービスやアプリケーションも利用可能ですが、上記のものが一般的です。

パート I. システムログイン

第2章 システム認証の設定

認証とは、あるシステムに対するユーザーの特定および検証のプロセスのことです。認証を行うには、ユーザー名やパスワードなど、ある種のアイデンティティおよび認証情報を提示する必要があります。システムは、設定した認証サービスと認証情報を比較します。認証情報が一致し、ユーザーアカウントが有効な場合には、ユーザーは **認証済み**になります。

ユーザーが認証済みになると、そのユーザー情報がアクセス制御サービスに渡されてユーザーに何か許可されているかを決定します。これが、ユーザーがアクセスする **認証済み** リソースです。認証および承認は 2 つの異なるプロセスとなっています。

システムには、ユーザー認証の確認のために、システム用の有効なアカウントデータベースの設定済みリストが必要です。ローカルシステムはユーザー情報に様々な異なるデータストアを使用することができます。それには **Lightweight Directory Access Protocol (LDAP)**、**Network Information Service (NIS)**、および **Winbind** が含まれます。さらに LDAP と NIS の両データストアは **Kerberos** を使ってユーザーを認証できます。

利便性とシングルサインオンの一環のため、**Red Hat Enterprise Linux** は **System Security Services Daemon (SSSD)** を中心となるデーモンとして使用し、異なる ID バックエンドに対するユーザーの認証や、ユーザー用に **ticket-granting ticket (TGT)** を依頼することもできます。SSSD は、LDAP や **Kerberos**、外部のアプリケーションと対話して、ユーザーの認証情報を確認することができます。

この章では、システム認証に利用可能な **Red Hat Enterprise Linux** のツールについて説明します。

- **Identity Management** システム向けの **ipa-client-install** ユーティリティおよび **realmd** システム。詳細は、「[システム認証用の Identity Management ツール](#)」を参照してください。
- 他のシステム向けの **authconfig** ユーティリティおよび **authconfig UI**。詳細は、「[authconfig の使用](#)」を参照してください。

2.1. システム認証用の IDENTITY MANAGEMENT ツール

Identity Management マシンで自動的にシステム認証を設定するには、**ipa-client-install** ユーティリティと **realmd** システムを使用することができます。

ipa-client-install

ipa-client-install ユーティリティは、システムがクライアントマシンとして **Identity Management** ドメインに参加するように設定します。**ipa-client-install** に関する詳細情報は、『[Linux ドメイン ID、認証、およびポリシーガイド](#)』を参照してください。

Identity Management システムには、**realmd** よりも **ipa-client-install** が推奨される点にご注意ください。

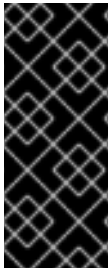
realmd

realmd システムは、**Identity Management** または **Active Directory** ドメインなどのアイデンティティドメインにマシンを参加させます。**realmd** に関する詳細情報は『[Windows 統合ガイド](#)』を参照してください。

2.2. AUTHCONFIG の使用

authconfig ツールは、LDAP など、どのようなデータストアをユーザー認証情報に使用するかを設定

する上で役立つことができます。Red Hat Enterprise Linux の **authconfig** には、ユーザーのデータストアを設定するオプションとして、GUI とコマンドラインの両方を使用できます。**authconfig** ツールは、システムで異なる形式の認証メカニズムを使用するのに加え、固有のサービス (SSSD、LDAP、NIS または Winbind) をユーザーデータベースとして使用するよう設定することができます。



重要

Red Hat は、Identity Management システムの設定には **authconfig** ではなく **ipa-client-install** ユーティリティまたは **realmd** システムを使用することを推奨します。**authconfig** ユーティリティは機能に制約があり、柔軟性が大幅に低下します。詳細情報は、「[システム認証用の Identity Management ツール](#)」を参照してください。

認証セッティングの設定には、以下の 3 つの **authconfig** ユーティリティが使用できます。

- **authconfig-gtk** は、完全なグラフィカルインターフェースを提供します。
- **authconfig** は、手動での設定に使用するコマンドラインインターフェースを提供します。
- **authconfig-tui** はテキストベースの UI を提供します。このユーティリティは非推奨であることに注意してください。

これらの設定ユーティリティはすべて **root** として実行する必要があります。

2.2.1. authconfig CLI 使用時のヒント

authconfig コマンドラインツールは、スクリプトに渡されたセッティングにしたがい、システム認証に必要な設定ファイルとサービスのすべてを更新します。UI で設定可能な ID と認証設定オプションよりもさらに多くを提供すると共に、**authconfig** ツールを使うとバックアップファイルとキックスタートファイルも作成できます。

authconfig オプションの完全なリストについては、ヘルプの出力と **man** ページを参照してください。

authconfig の実行に際しては、以下の点に留意してください。

- すべてのコマンドで **--update** か **--test** のオプションを使用します。これらのオプションのいずれかがコマンドを正しく実行するために必要になります。**--update** を使用すると設定の変更を書き込みます。**--test** オプションは設定への変更を表示しますが、適用はしません。

--update オプションを使用しないと、システム設定ファイルに変更が書き込まれません。
- コマンドラインは、既存設定の更新のほか、新規設定にも使用することができます。このためコマンドラインでは、(強制すると、コマンドが設定すべてを更新する可能性があるため) 指定の呼び出しに必須属性が強制的に使用されることはありません。

認証設定を編集する際は、設定が完全かつ正確であることを確認してください。認証設定を不完全なものまたは間違った値に変更すると、ユーザーがシステムからロックアウトされてしまう可能性があります。**--update** オプションを使用して変更を書き込む前に、**--test** オプションで設定が適切であることを確認してください。

- それぞれの「**enable**」オプションには対応する「**disable**」オプションがあります。

2.2.2. authconfig UI のインストール

authconfig UI は、デフォルトではインストールされませんが、管理者が認証設定に簡単な変更を加える際には役立ちます。

UI をインストールするには、**authconfig-gtk** パッケージをインストールします。このパッケージには、**authconfig** コマンドラインツールや **Bash**、**Python** など一般的なシステムパッケージへの依存関係があります。これらのほとんどは、デフォルトでインストールされます。

```
[root@server ~]# yum install authconfig-gtk
Loaded plugins: langpacks, product-id, subscription-manager
Resolving Dependencies
--> Running transaction check
---> Package authconfig-gtk.x86_64 0:6.2.8-8.el7 will be installed
--> Finished Dependency Resolution
```

Dependencies Resolved

```
=====
=====
Package                Arch          Version           Repository
Size
=====
=====
Installing:
authconfig-gtk         x86_64        6.2.8-8.el7       RHEL-Server
105 k
```

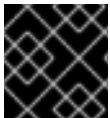
Transaction Summary

```
=====
=====
Install  1 Package

... 8< ...
```

2.2.3. authconfig UI の起動

1. 端末を開いて、**root** でログインします。
2. **system-config-authentication** コマンドを実行します。



重要

authconfig UI を閉じると、すぐに変更が反映されます。

認証の設定 ダイアログボックスには以下の 3 つの設定タブがあります。

- **識別と認証 ID ストア** (ユーザー ID と対応する認証情報が保存されるデータレポジトリ) として使用されるリソースを設定します。
- **高度なオプション** スマートカードや指紋などのパスワードや認証情報以外の認証方法を設定します。
- **パスワードオプション** パスワードの認証方法を設定します。

The screenshot shows the 'Authentication Configuration' window with three tabs: 'Identity & Authentication' (selected), 'Advanced Options', and 'Password Options'. The window is divided into two main sections: 'User Account Configuration' and 'Authentication Configuration'.

User Account Configuration

- User Account Database:
- LDAP Search Base DN:
- LDAP Server:
- ☒ Use TLS to encrypt connections
-

Authentication Configuration

- Authentication Method:
- Realm:
- KDCs:
- Admin Servers:
- ☐ Use DNS to resolve hosts to realms
- ☒ Use DNS to locate KDCs for realms

At the bottom, there are three buttons: 'Revert', 'Cancel', and 'Apply'.

図2.1 authconfig ウィンドウ

2.2.4. 認証設定のテスト

認証設定を完全かつ適切に行うことは極めて重要です。これが行われないと、ユーザー (root さえも) すべてがシステムからロックアウトされたり、一部のユーザーがブロックされてしまったりする可能性があります。

--test オプションは、システム用のすべての ID および認証メカニズムに関する認証設定をプリントします。これには、有効なものおよび無効なエリアの設定の両方が表示されます。

test オプションはそれのみで使用する完全な現行設定を表示するか、**authconfig** コマンドと使用して (実際に変更せずに) 設定がどのように変更されるかを表示することができます。これは、提示された認証設定が完全かつ正確かどうかを確認する上で非常に便利なものです。

```
[root@server ~]# authconfig --test
caching is disabled
nss_files is always enabled
nss_compat is disabled
nss_db is disabled
nss_hesiod is disabled
  hesiod LHS = ""
  hesiod RHS = ""
nss_ldap is disabled
  LDAP+TLS is disabled
  LDAP server = ""
  LDAP base DN = ""
nss_nis is disabled
  NIS server = ""
  NIS domain = ""
nss_nisplus is disabled
nss_winbind is disabled
  SMB workgroup = "MYGROUP"
  SMB servers = ""
  SMB security = "user"
  SMB realm = ""
  Winbind template shell = "/bin/false"
  SMB idmap range = "16777216-33554431"
nss_sss is enabled by default
nss_wins is disabled
nss_mdns4_minimal is disabled
DNS preference over NSS or WINS is disabled
pam_unix is always enabled
  shadow passwords are enabled
  password hashing algorithm is sha512
pam_krb5 is disabled
  krb5 realm = "#"
  krb5 realm via dns is disabled
  krb5 kdc = ""
  krb5 kdc via dns is disabled
  krb5 admin server = ""
pam_ldap is disabled
  LDAP+TLS is disabled
  LDAP server = ""
  LDAP base DN = ""
  LDAP schema = "rfc2307"
pam_pkcs11 is disabled
  use only smartcard for login is disabled
```



```

smartcard module = ""
smartcard removal action = ""
pam_fprintd is disabled
pam_ecryptfs is disabled
pam_winbind is disabled
SMB workgroup = "MYGROUP"
SMB servers = ""
SMB security = "user"
SMB realm = ""
pam_sss is disabled by default
credential caching in SSSD is enabled
SSSD use instead of legacy services if possible is enabled
IPAv2 is disabled
IPAv2 domain was not joined
IPAv2 server = ""
IPAv2 realm = ""
IPAv2 domain = ""
pam_pwquality is enabled (try_first_pass local_users_only retry=3
authtok_type=)
pam_passwdqc is disabled ()
pam_access is disabled ()
pam_mkhomedir or pam_oddjob_mkhomedir is disabled (umask=0077)
Always authorize local users is enabled ()
Authenticate system accounts against network services is disabled

```

2.2.5. authconfig を使用した設定の保存と復元

認証設定を変更すると、問題が発生する場合があります。不適切な変更を行うと、アクセスがあるべきユーザーを除外したり、ID ストアへの接続が失敗したり、システムへのすべてのアクセスがロックアウトされてしまうことすらあります。

管理者は認証設定を変更する前に、すべての設定ファイルのバックアップを作成することが強く推奨されます。これは **--savebackup** オプションで実行できます。

```
[root@server ~]# authconfig --savebackup=/backups/authconfigbackup20180701
```

--restorebackup オプションと適用するバックアップ名を使用することで、以前に保存されたあらゆるバージョンの認証設定をも復元することができます。

```
[root@server ~]# authconfig --
restorebackup=/backups/authconfigbackup20180701
```

authconfig コマンドは、設定が変更されるたびに自動的にバックアップを保存します。**--restorelastbackup** オプションを使用すると、最新の自動バックアップを復元できます。

```
[root@server ~]# authconfig --restorelastbackup
```

第3章 `AUTHCONFIG` を使用して認証用に ID ストアを選択する手順

`authconfig` UI の **識別と認証** タブは、ユーザーの認証方法を設定します。デフォルトではローカルシステムの認証を使用します。つまり、ユーザーとユーザーのパスワードがローカルシステムのアカウントに対してチェックされます。**Red Hat Enterprise Linux** のマシンは、ユーザーと認証情報を含む LDAP、NIS、および Winbind などの外部リソースを使用することもできます。

3.1. IPA V2

Identity Management サーバーを ID バックエンドとして設定するには、2 つの方法があります。IdM のバージョン 2 (**Red Hat Enterprise Linux** バージョン 6.3 以前)、バージョン 3 (**Red Hat Enterprise Linux** 6.4 以降)、およびバージョン 4 (**Red Hat Enterprise Linux** 7.1 以降) では、`authconfig` で IPA v2 プロバイダーとして設定されます。これよりも旧式の IdM バージョンおよびコミュニティの FreeIPA サーバーの場合は、LDAP プロバイダーとして設定されます。

3.1.1. UI での IdM の設定

1. `authconfig` UI を開きます。
2. ユーザーアカウントデータベース ドロップダウンメニュー内で **IPA v2** を選択します。

Authentication Configuration

Identity & Authentication | Advanced Options | Password Options

Use the "Join Domain" button to join the IPAv2 domain.

User Account Configuration

User Account Database: IPAv2

IPA Domain:

IPA Realm:

IPA Server:

☐ Do not configure NTP

Join Domain...

Authentication Configuration

Authentication Method: IPAv2 password

Revert Cancel Apply

図3.1 認証の設定

3. IdM サーバーへの接続に必要な情報を設定します。

- **IPA ドメイン** には、IdM ドメインの DNS ドメインを入力します。
- **IPA レalm** には、IdM ドメインの Kerberos ドメインを入力します。
- **IPA サーバー** には、IdM ドメイントポロジー内のいずれかの IdM サーバーのホスト名を入力します。
- **NTP を設定しない** チェックボックスを選択すると、クライアント設定時に NTP サービスを無効にします。IdM サーバーとすべてのクライアントは、Kerberos 認証と認証情報が正

常に機能するために同期されたクロックを必要とするため、この設定は通常推奨されません。IdM サーバーがドメイン内でホスティングしている NTP サーバー以外のものを使用している場合は、これを無効にすることができます。

4. ドメインへ参加 ボタンをクリックします。

これで **ipa-client-install** コマンドが実行され、必要な場合は **ipa-client** パッケージがインストールされます。インストールスクリプトは、ローカルシステムに必要となるすべてのシステムファイルを自動的に設定し、ドメイン情報更新のためにドメインサーバーに接続します。

3.1.2. コマンドラインを使用した IdM の設定

IdM ドメインは、特に DNS や Kerberos といった一般的かつ必須のサービスを単一階層に集約します。

(「[8章 realmd を使った ID ドメインへの接続](#)」の **realmd** のように) **authconfig** を使うと IdM ドメインにシステムを登録することができます。このコマンドは **ipa-client-install** コマンドを実行し、必要な場合は **ipa-client** パッケージをインストールします。インストールスクリプトは、ローカルシステムに必要となるすべてのシステムファイルを自動的に設定し、ドメイン情報更新のためにドメインサーバーに接続します。

ドメインへの参加には、DNS ドメイン名 (**--ipav2domain**)、Kerberos レalm 名 (**--ipav2realm**)、および接続する IdM サーバー (**--ipav2server**) という 3 つの情報が必要になります。**--ipav2join** オプションは、IdM サーバーへの接続に管理者が使用するユーザー名を指定し、通常は **admin** とします。

```
[root@server ~]# authconfig --enableipav2 --ipav2domain=IPAEXAMPLE --  
ipav2realm=IPAEXAMPLE --ipav2server=ipaexample.com --ipav2join=admin
```

IdM ドメインが独自の NTP サービスを実行していない場合、設定スクリプトが IdM サーバーを NTP サーバーとして使用しないように **--disableipav2nntp** オプションを使うことが可能です。IdM サーバーとすべてのクライアントは、Kerberos 認証と認証情報が正常に機能するために同期されたクロックを必要とするため、この設定は通常推奨されません。

3.2. LDAP と IDM

(OpenLDAP や Red Hat Directory Server などの) LDAP ディレクトリーは、LDAP アイデンティティプロバイダーとして使用することができます。また、古いバージョンの IdM と FreeIPA は、それらを関連する Kerberos サーバーと LDAP プロバイダーとして設定することで、アイデンティティプロバイダーとすることができます。

ユーザーデータベースの LDAP サーバー設定には、**openldap-clients** パッケージか **sssd** パッケージを使用します。両パッケージともデフォルトでインストールされています。

3.2.1. UI での LDAP 認証の設定

1. 「[authconfig UI の起動](#)」にあるように、**authconfig UI** を開きます。
2. ユーザーアカウントデータベースのドロップダウンメニューで **LDAP** を選択します。

Authentication Configuration

Identity & Authentication Advanced Options Password Options

User Account Configuration

User Account Database: LDAP

LDAP Search Base DN: ou=people,dc=example,dc=com

LDAP Server: ldap://idm.example.com/

☒ Use TLS to encrypt connections

Download CA Certificate...

Authentication Configuration

Authentication Method: Kerberos password

Realm: EXAMPLE

KDCs:

Admin Servers:

☐ Use DNS to resolve hosts to realms

☒ Use DNS to locate KDCs for realms

Revert Cancel Apply

3. LDAP サーバーへの接続に必要な情報を設定します。

- **LDAP 検索ベース DN** で、ユーザーディレクトリーの root のサフィックスまたは識別名 (DN) を指定します。アイデンティティまたは認証に使用するユーザーエントリーはすべて、**ou=people,dc=example,dc=com** などのように、この親エントリーの下に存在しま

す。

このフィールドはオプションです。指定しない場合は、LDAP サーバーの設定エントリーにある **namingContexts** および **defaultNamingContext** 属性を使って System Security Services Daemon (SSSD) が検索ベースの検出を試行します。

- **LDAP サーバー** には、LDAP サーバーの URL を入力します。これには通常、**ldap://ldap.example.com:389** のように、LDAP サーバーのホスト名とポート番号の両方が必要になります。

ldaps:// で開始する URL を使用してセキュアなプロトコルを入力すると、**CA 証明書をダウンロードする** ボタンが有効になり、このボタンで LDAP サーバーの発行 CA 証明書を発行元の証明局から取得します。CA 証明書はプライバシー強化メール (PEM: privacy enhanced mail) 形式でなければなりません。

- セキュアでない標準のポート接続 (**ldap://** で始まる URL) を使用する場合は、**TLS を使用して接続を暗号化する** チェックボックスを使用して **STARTTLS** で LDAP サーバーとの通信を暗号化します。このチェックボックスを選択すると、**CA 証明書をダウンロードする...** ボタンが有効になります。



注記

サーバーの URL で LDAPS (LDAP および SSL) のセキュアなプロトコルが使用されている場合には、通信がすでに暗号化されているので **TLS を使用して接続を暗号化する** チェックボックスは選択する必要はありません。

4. 認証の方法を選択します。LDAP は、単純なパスワード認証または Kerberos 認証を受け付けます。

Kerberos の使用方法は、「[UI での Kerberos 認証の設定](#)」で説明しています。

LDAP パスワード オプションでは、LDAP 認証を使用するために PAM アプリケーションを使用します。このオプションは、LDAP サーバーへの接続に LDAPS または TLS のいずれかを設定したセキュアな接続が必要です。

3.2.2. コマンドラインでの LDAP ユーザストアの設定

LDAP ID ストアを使用するには、**--enableldap** を使用します。LDAP を認証ソースとして使用するには、**--enableldapauth** を使用して、それから LDAP サーバー名、ユーザー接尾辞のベース DN、および (オプションとして) TLS を使用するかどうか、などの必須となる接続情報を使用します。**authconfig** コマンドには、ユーザーエントリーの RFC 2307bis スキーマを有効または無効にするオプションもあります。これは、**authconfig** UI では利用できません。

プロトコル (**ldap** または **ldaps**) とポート番号を含む完全な LDAP URL を使うようにしてください。**--enableldaptls** オプションとセキュアな LDAP URL (**ldaps**) を一緒に使用しないでください。

```
authconfig --enableldap --enableldapauth --
ldapserver=ldap://ldap.example.com:389,ldap://ldap2.example.com:389 --
ldapbasedn="ou=people,dc=example,dc=com" --enableldaptls --
ldaploadcacert=https://ca.server.example.com/caCert.crt --update
```

LDAP のパスワード認証に **--ldapauth** ではなく、LDAP ユーザストアで Kerberos を使用することもできます。これらのオプションは「[コマンドラインでの Kerberos 認証の設定](#)」で説明されています。

3.3. NIS



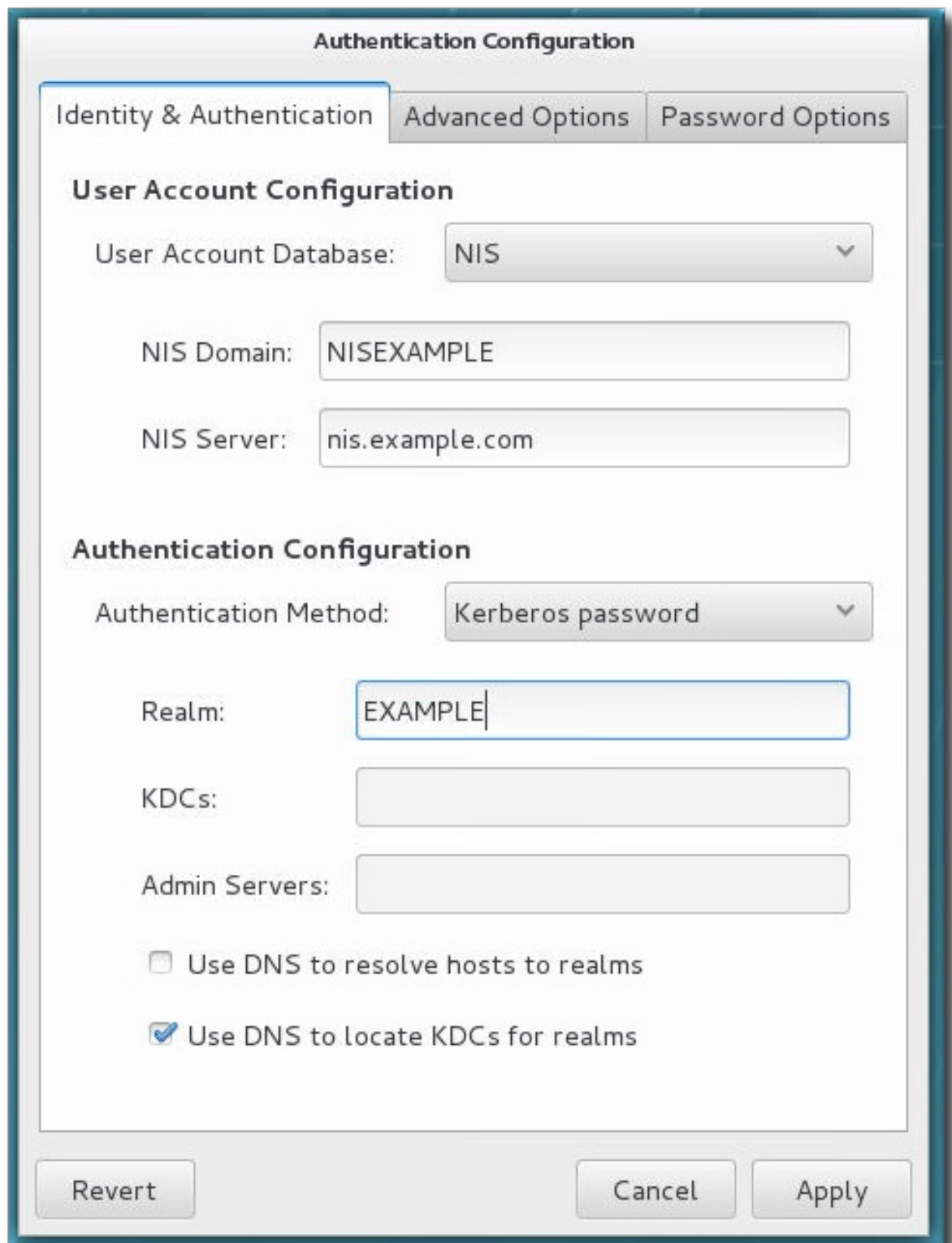
重要

NIS を ID ストアとして設定する前に、NIS 自体を環境にあわせて設定する必要があります。

- NIS サーバーは、ユーザーアカウント設定で完全に設定する必要があります。
- ローカルシステムに **ypbind** パッケージをインストールする必要があります。これは NIS サービスに必要なものですが、デフォルトではインストールされません。
- **portmap** と **ypbind** サービスが起動されており、ブート時に起動するように設定します。これは、**ypbind** パッケージのインストール時に設定します。

3.3.1. UI での NIS 認証の設定

1. 「[authconfig UI の起動](#)」にあるように、**authconfig UI** を開きます。
2. ユーザーアカウントデータベースのドロップダウンメニューで **NIS** を選択します。



The image shows a 'Authentication Configuration' dialog box with three tabs: 'Identity & Authentication' (selected), 'Advanced Options', and 'Password Options'. Under 'Identity & Authentication', there are two sections. The first section, 'User Account Configuration', includes a 'User Account Database' dropdown set to 'NIS', an 'NIS Domain' text field with 'NISEXAMPLE', and an 'NIS Server' text field with 'nis.example.com'. The second section, 'Authentication Configuration', includes an 'Authentication Method' dropdown set to 'Kerberos password', a 'Realm' text field with 'EXAMPLE', and empty text fields for 'KDCs' and 'Admin Servers'. Below these are two checkboxes: 'Use DNS to resolve hosts to realms' (unchecked) and 'Use DNS to locate KDCs for realms' (checked). At the bottom are 'Revert', 'Cancel', and 'Apply' buttons.

Authentication Configuration

Identity & Authentication | Advanced Options | Password Options

User Account Configuration

User Account Database: NIS

NIS Domain: NISEXAMPLE

NIS Server: nis.example.com

Authentication Configuration

Authentication Method: Kerberos password

Realm: EXAMPLE

KDCs:

Admin Servers:

☐ Use DNS to resolve hosts to realms

☒ Use DNS to locate KDCs for realms

Revert Cancel Apply

3. NIS サーバーに接続するための情報として、NIS ドメイン名とサーバーのホスト名を設定します。NIS サーバーが指定されない場合は、**authconfig** デーモンが NIS サーバーをスキャンして検索します。
4. 認証の方法を選択します。NIS は、単純なパスワード認証または Kerberos 認証を受け付けます。

Kerberos の使用方法は、「[UI での Kerberos 認証の設定](#)」で説明しています。

3.3.2. コマンドラインを使用した NIS の設定

NIS ID ストアを使用するには、**--enablenis** を使います。Kerberos パラメーターが明示的に設定されている場合 (「[コマンドラインでの Kerberos 認証の設定](#)」) を除き、ここでは自動的に NIS 認証が使用されます。唯一のパラメーターは、NIS サーバーと NIS ドメインを特定するためのものです。これらが使用されない場合は、**authconfig** サービスはネットワークをスキャンして NIS サーバーを探します。

```
[root@server ~]# authconfig --enablenis --nisdomain=EXAMPLE --
nisserver=nis.example.com --update
```

3.4. WINBIND

Winbind をシステムに ID ストアとして設定する前に、Samba を設定する必要があります。Samba サーバーはユーザーアカウント用にセットアップするか、バックエンドの ID ストアとして Active Directory を使用するように設定する必要があります。

Samba の設定については、[Samba project documentation](#) で説明されています。Samba を Active Directory との統合ポイントとして設定する方法については、『[Red Hat Enterprise Linux Windows 統合ガイド](#)』で説明しています。

3.4.1. authconfig GUI での Winbind の有効化

1. **samba-winbind** パッケージをインストールします。これは、Samba サービスの Windows 統合機能に必要なものですが、デフォルトではインストールされていません。

```
[root@server ~]# yum install samba-winbind
```

2. **authconfig** UI を開きます。

```
[root@server ~]# authconfig-gtk
```

3. **識別と認証** タブの **ユーザーアカウントデータベース** ドロップダウンメニューで **Winbind** を選択します。

Authentication Configuration

Identity & Authentication | Advanced Options | Password Options

User Account Configuration

User Account Database: Winbind

Winbind Domain: MYGROUP

Security Model: ads

Winbind ADS Realm: AEXAMPLE

Winbind Domain Controllers: adexample.com

Template Shell: /bin/false

☒ Allow offline login

Join Domain...

Authentication Configuration

Authentication Method: Winbind password

Revert Cancel Apply

4. Microsoft Active Directory ドメインコントローラーへ接続するために必要な情報を設定します。

- **Winbind** ドメイン には、接続先の Windows ドメインを入力します。

これは、**DOMAIN** のような Windows 2000 形式にします。

- **セキュリティーモデル** では、Samba クライアントに使用するセキュリティーモデルを設定します。**authconfig** は、以下の 4 つのタイプのセキュリティーモデルをサポートしています。

- **ads** は、Samba が Active Directory Server (ADS) レルムのドメインメンバーとして機能するように設定します。このモードで操作するには、**krb5-server** パッケージがインストールされ、Kerberos が適切に設定されている必要があります。
- **domain** では、Windows サーバーと同様に Windows のプライマリまたはバックアップドメインコントローラーがユーザー名およびパスワードを認証することで、Samba が検証します。
- **server** では、別のサーバー (例: Windows Server) で認証することにより、ローカルの Samba サーバーがユーザー名およびパスワードを確認します。サーバー認証に失敗した場合には、システムは **user** モードで認証を試みます。
- **user** では、クライアントが有効なユーザー名とパスワードでログインする必要があります。このモードは暗号化されたパスワードをサポートします。

ユーザー名の形式は、**EXAMPLE\jsmith** のように **ドメイン\ユーザー** とする必要があります。



注記

任意のユーザーが Windows ドメイン内に存在することを確認する際は、常に **domain\user_name** の形式を使用して、バックスラッシュ文字 (\) でエスケープします。以下に例を示します。

```
[root@server ~]# getent passwd domain\\user
DOMAIN\user:*:16777216:16777216:Name
Surname:/home/DOMAIN/user:/bin/bash
```

以下はデフォルトのオプションになります。

- **Winbind ADS レルム** には、Samba サーバーが参加する Active Directory レルムを入力します。これは **ads** セキュリティーモデルの場合にのみ、使用されます。
- **Winbind ドメインコントローラー** には、システム登録に使用するドメインコントローラーのホスト名または IP アドレスを入力します。
- **テンプレートシェル** では、Windows のユーザーアカウント設定に使用するログインシェルを設定します。
- **オフラインログインを許可** は、ローカルキャッシュでの認証情報の保存を許可します。システムがオフラインの時にユーザーがシステムリソースに認証を試みると、キャッシュが参照されます。

3.4.2. コマンドラインでの Winbind の有効化

Windows のドメインには複数の異なるセキュリティーモデルがあり、ドメインで使われるセキュリティーモデルがローカルシステムの認証設定を決定します。ユーザーとサーバーのセキュリティーモデルでは、Winbind 設定で必要となるのはドメイン (またはワークグループ) の名前とドメインコントローラーのホスト名のみです。

--winbindjoin パラメーターは Active Directory ドメイン接続に使用するユーザーを設定し、**--enablelocalauthorize** はローカルの承認操作で **/etc/passwd** ファイルを確認するよう設定します。

authconfig コマンドの実行後に、Active Directory ドメインに参加します。

```
[root@server ~]# authconfig --enablewinbind --enablewinbindauth --
smbsecurity=user|server --enablewinbindoffline --
smbservers=ad.example.com --smbworkgroup=EXAMPLE --update --
enablelocauthorize --winbindjoin=admin
[root@server ~]# net join ads
```



注記

ユーザー名の形式は、**EXAMPLE\jsmith**のように **ドメイン\ユーザー** とする必要があります。

任意のユーザーが Windows ドメイン内に存在することを検証する際は、常に **domain\user** の形式を使い、バックスラッシュ文字 (\) でエスケープします。以下に例を示します。

```
[root@server ~]# getent passwd domain\\user
DOMAIN\user:*:16777216:16777216:Name
Surname:/home/DOMAIN/user:/bin/bash
```

ads と **domain** ドメインのセキュリティーモデルでは、Winbind 設定はテンプレートシェルとレルム (**ads** のみ) の追加設定を許可します。例を示します。

```
[root@server ~]# authconfig --enablewinbind --enablewinbindauth --
smbsecurity ads --enablewinbindoffline --smbservers=ad.example.com --
smbworkgroup=EXAMPLE --smbrealm EXAMPLE.COM --winbindtemplateshell=/bin/sh
--update
```

Windows ベースの認証と Windows ユーザーアカウント情報の設定では、名前形式、ドメイン名をユーザー名と一緒に要求するかどうか、UID の範囲など、数多くの他のオプションがあります。これらのオプションは **authconfig** ヘルプ内に記載されています。

第4章 認証メカニズムの設定

Red Hat Enterprise Linux は、複数の異なる認証メソッドをサポートします。**authconfig** ツールまたは、場合によっては Identity Management ツールを使用して設定することができます。

4.1. AUTHCONFIG を使用したローカル認証の設定

ローカル認証のオプションでは、バックエンドに保存されているユーザーではなく、ローカルシステムのアカウントの設定を定義します。この設定では、システムサービスへのユーザーベースの承認を定義します (**/etc/security/access.conf** で定義)。そうでない場合は、承認ポリシーはアイデンティティプロバイダー内もしくはサービス自体で定義できます。

4.1.1. UI でのローカルアクセス制御の有効化

ローカルアクセス制御を有効にする は、ローカルユーザーの承認ルールをシステムが **/etc/security/access.conf** ファイルで確認するように設定します。これは PAM 承認です。

The image shows a window titled "Authentication Configuration" with three tabs: "Identity & Authentication", "Advanced Options" (which is selected), and "Password Options".

Local Authentication Options

- ☒ Enable fingerprint reader support
- ☐ Enable local access control

Tip: This is managed via `/etc/security/access.conf`.

Password Hashing Algorithm: SHA512 ▼

Other Authentication Options

- ☐ Create home directories on the first login

Smart Card Authentication Options

- ☒ Enable smart card support

Tip: Smart cards support logging into both local and centrally managed accounts.

Card Removal Action: Ignore ▼

- ☐ Require smart card for login

At the bottom of the window are three buttons: "Revert", "Cancel", and "Apply".

図4.1 ローカルアカウントのフィールド

4.1.2. コマンドラインでのローカルアクセス制御の設定

`authconfig` ではローカル承認制御を有効にする 2 つのオプションがあります。--
`enablelocalauthorize` はネットワーク認証を省略して、システムユーザーに関してローカルファイルのみを確認します。--`enablepamaccess` は、システムが `/etc/security/access.conf` でシステ

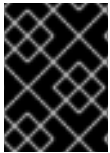
ム承認ポリシーを検索するよう設定します。

```
[root@server ~]# authconfig --enablelocauthorize --enablepamaccess --update
```

4.2. AUTHCONFIG を使用したシステムパスワードの設定

4.2.1. パスワードのセキュリティー

パスワードがプレーンテキスト形式で保存されている場合は、クラッキング、不正アクセスや改ざんに対して脆弱になります。これを回避するには、パスワードハッシュダイジェストをセキュアに保存するための暗号化ハッシュアルゴリズムを使用することができます。IdM でサポートされる推奨 (およびデフォルト) のハッシュアルゴリズムは **SHA-512** で、これは追加のセキュリティーとして **64** ビット文字と、ソルトおよびストレッチングも使用します。後方互換性を確保するために、**SHA-256**、**DES**、**BigCrypt** および **MD5** もサポートします。



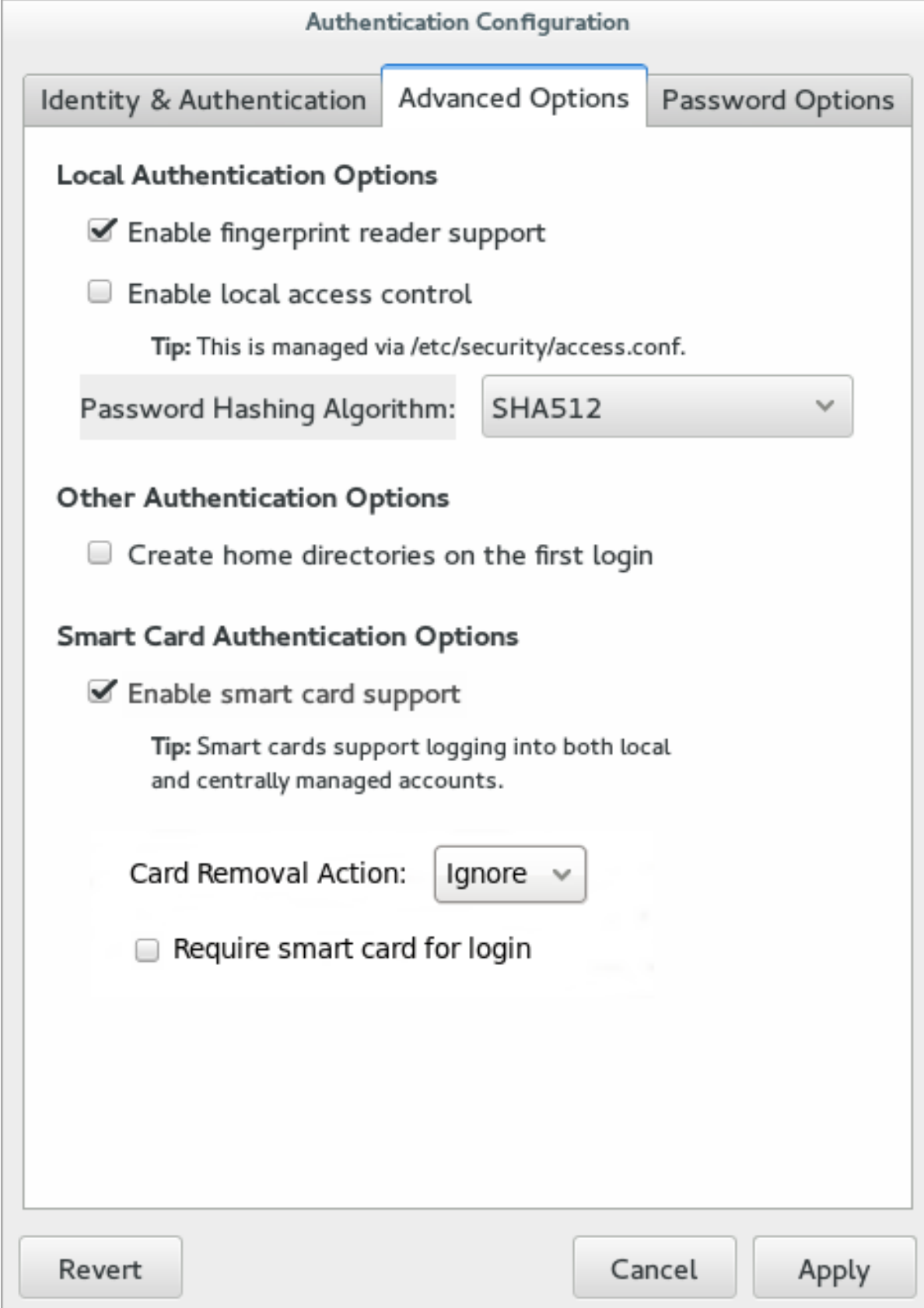
重要

後方互換性が不要な場合は、**SHA-512** の方がよりセキュアであるため、こちらのみを使用してください。

4.2.1.1. UI でのパスワードハッシュの設定

ローカル認証のオプション タブでは、ローカルパスワードをシステムに保存する方法を設定します。パスワードハッシュアルゴリズムのドロップダウンメニューでは、パスワードハッシュをセキュアに保存するアルゴリズムを設定します。

1. 「[authconfig UI の起動](#)」にあるように、**authconfig UI** を開きます。
2. **高度なオプション** タブを開きます。
3. **パスワードハッシュアルゴリズム** のドロップダウンメニューから使用するアルゴリズムを選択します。



The image shows a 'Authentication Configuration' dialog box with three tabs: 'Identity & Authentication', 'Advanced Options' (which is selected), and 'Password Options'. Under the 'Advanced Options' tab, there are three sections: 'Local Authentication Options' with checkboxes for 'Enable fingerprint reader support' (checked) and 'Enable local access control' (unchecked), followed by a tip about /etc/security/access.conf and a 'Password Hashing Algorithm' dropdown set to 'SHA512'; 'Other Authentication Options' with a checkbox for 'Create home directories on the first login' (unchecked); and 'Smart Card Authentication Options' with a checked 'Enable smart card support' checkbox, a tip about smart cards, a 'Card Removal Action' dropdown set to 'Ignore', and an unchecked 'Require smart card for login' checkbox. At the bottom are 'Revert', 'Cancel', and 'Apply' buttons.

Authentication Configuration

Identity & Authentication **Advanced Options** Password Options

Local Authentication Options

☒ Enable fingerprint reader support

☐ Enable local access control

Tip: This is managed via /etc/security/access.conf.

Password Hashing Algorithm: SHA512 ▼

Other Authentication Options

☐ Create home directories on the first login

Smart Card Authentication Options

☒ Enable smart card support

Tip: Smart cards support logging into both local and centrally managed accounts.

Card Removal Action: Ignore ▼

☐ Require smart card for login

Revert Cancel Apply

4. **適用** ボタンをクリックします。

4.2.1.2. コマンドラインでのパスワードハッシュ化の設定

ユーザーのパスワードダイジェストをセキュアに保存するために使用するハッシュアルゴリズムを設定もしくは変更するには、**--passalgo** オプションとハッシュの略名を使用します。たとえば、SHA-512 アルゴリズムには以下を使用します。

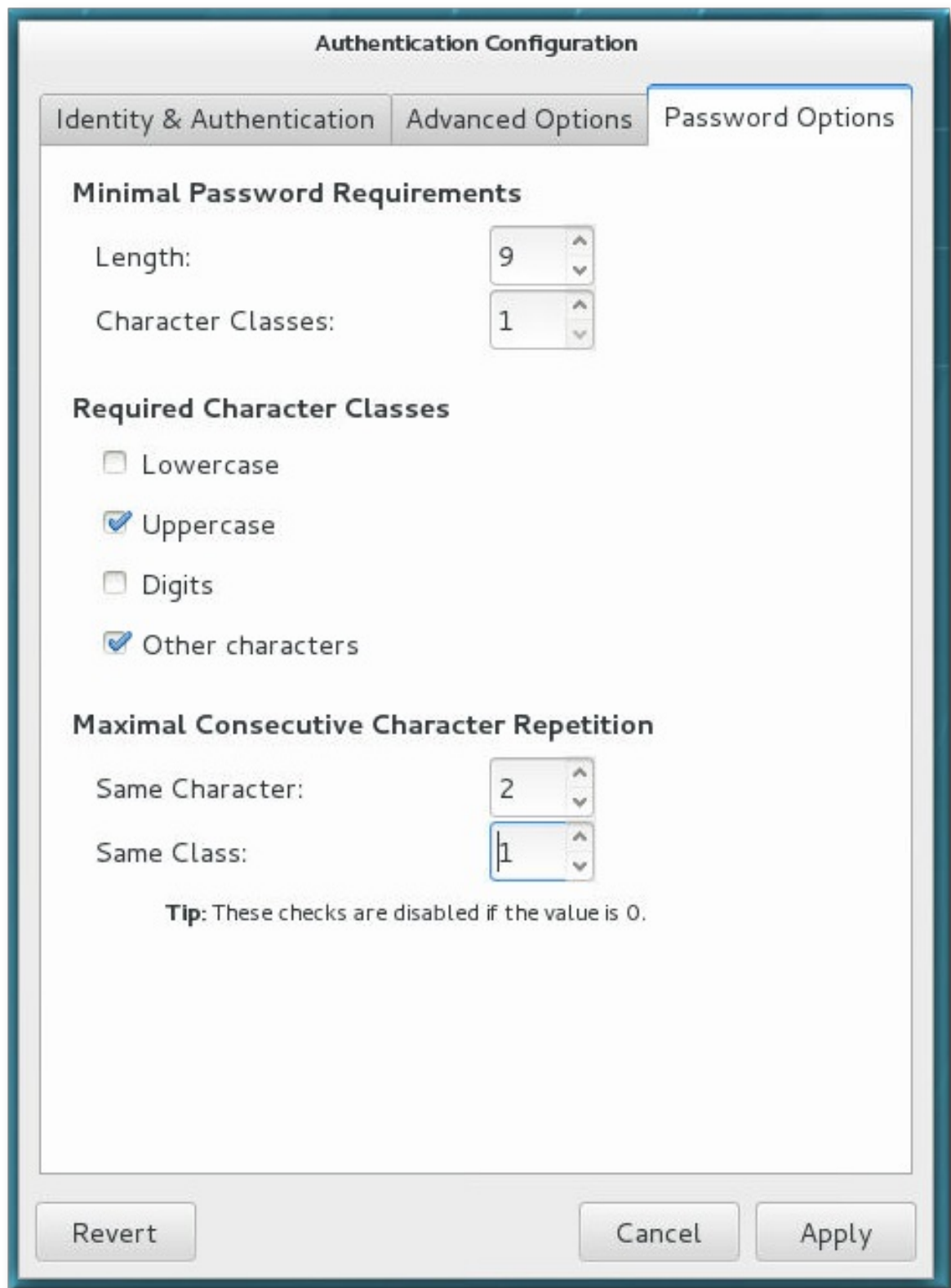

```
[root@server ~]# authconfig --passalgo=sha512 --update
```

4.2.2. パスワードの複雑性

パスワードの複雑性は、ローカルユーザーアカウントの設定に使用可能なパスワードの強度を設定します。複雑性は、長さと文字クラスの組み合わせで決められます。パスワードの複雑性を決定するポリシーは、2つの要素から構成されます。ひとつはパスワードで使用可能な文字タイプの特定です (大文字と小文字や特別な文字など)。もうひとつはパスワード内でこれらの文字がどのように使用できるかです (長さおよび連続文字数)。

4.2.2.1. UI を使用したパスワードの強度設定

1. 「[authconfig UI の起動](#)」にあるように、**authconfig UI** を開きます。
2. パスワードオプション タブを開きます。



The image shows a screenshot of the 'Authentication Configuration' dialog box, specifically the 'Password Options' tab. The dialog has three tabs: 'Identity & Authentication', 'Advanced Options', and 'Password Options'. The 'Password Options' tab is active and contains the following settings:

- Minimal Password Requirements**
 - Length: 9
 - Character Classes: 1
- Required Character Classes**
 - ☐ Lowercase
 - ☒ Uppercase
 - ☐ Digits
 - ☒ Other characters
- Maximal Consecutive Character Repetition**
 - Same Character: 2
 - Same Class: 1

Below these settings is a tip: **Tip:** These checks are disabled if the value is 0.

At the bottom of the dialog are three buttons: 'Revert', 'Cancel', and 'Apply'.

3. 以下の**最小パスワード要件**を設定します。
 - パスワードの最低限の長さ
 - パスワードで使用する文字クラスの最低数
4. パスワードに**必要な文字クラス**を有効にします。たとえば、パスワードには大文字を使用することができますが、**大文字** チェックボックスを選択すると、パスワードはすべて大文字を使用する必要があります。

5. 同じ文字もしくは同じクラスを連続させることができる最大数を設定します (これをゼロに設定すると、連続する数は制限されません)。

同じ文字 フィールドでは、単一文字の連続可能な最大数を設定します。たとえばこれを **2** に設定すると、**ssecret** は許可されますが **sssecret** は許可されません。

同様に **同じクラス** では、(大文字、特別文字といった) 文字クラスからの文字の連続可能な最大数を設定します。たとえばこれを **3** に設定すると、**secret!!** は許可されますが、**secret!!@** や **secret1234** は許可されません。

6. **適用** ボタンをクリックします。

4.2.2.2. コマンドラインでのパスワードの複雑性の設定

パスワードの複雑性をコマンドラインで定義する際には、設定要件は 2 つの要素から構成されます。ひとつは、パスワードの構成方法です。つまり、長さや連続文字数、必要な文字クラスの数などです。

- パスワードの最低限の長さ (**--passminlen**)。
- パスワードで使用する文字クラスの最低数 (**--passminclass**)。
- 同じ文字を連続させることができる最大数 (**--passmaxrepeat**)。これをゼロに設定すると、連続する数は制限されません。
- 同じクラスの文字 (数字など) を連続させることができる最大数 (**--passmaxclassrepeat**)。これをゼロに設定すると、連続する数は制限されません。

もうひとつは、パスワードに使用できる文字クラスを定義します。すべてのクラスが暗示的に使用可能ですが、**--enablereqType** オプションを使うと、特定のクラスが必須となり、そのクラスがないパスワードは許可されません。(反対に、クラスを明示的に拒否する設定も可能です。)

- 大文字 (**--enablerequpper**)
- 小文字 (**--enablereqlower**)
- 数字 (**--enablereqdigit**)
- 特殊文字 (**--enablereqother**)

たとえば、以下の設定では最小文字数を 9 文字とし、文字およびクラスの最大連続数は 2 となり、大文字と特殊文字の両方を必須とします。

```
[root@server ~]# authconfig --passminlen=9 --passminclass=3 --
passmaxrepeat=2 --passmaxclassrepeat=2 --enablerequpper --enablereqother --
update
```

4.3. AUTHCONFIG を使用した KERBEROS (LDAP または NIS 認証) の設定

LDAP と NIS の認証ストアは両方とも Kerberos の認証メソッドをサポートします。Kerberos を使用すると以下の利点があります。

- 標準ポート上での接続を許可しながら、通信にセキュリティー層を使用します。
- オフラインログインを可能にする SSSD を使用した認証情報キャッシングを自動的に使用します。



注記

Kerberos 認証の使用には、**krb5-libs** と **krb5-workstation** の両パッケージが必要です。

4.3.1. UI での Kerberos 認証の設定

認証の方法 のドロップダウンメニューから **Kerberos パスワード** を選ぶと、自動的に Kerberos レルムへの接続に必要なフィールドが開きます。

Authentication Configuration

Identity & Authentication Advanced Options Password Options


User Account Configuration

User Account Database: LDAP

LDAP Search Base DN: ou=people,dc=example,dc=co

LDAP Server: ldap://idm.example.com/

☒ Use TLS to encrypt connections

 Download CA Certificate...

Authentication Configuration

Authentication Method: Kerberos password

Realm: EXAMPLE

KDCs:

Admin Servers:

☐ Use DNS to resolve hosts to realms

☒ Use DNS to locate KDCs for realms

Revert Cancel Apply

図4.2 Kerberos フィールド

- **レルム** には、Kerberos サーバー用のレルム名を入力します。レルムとは、1つまたはそれ以上の **キー配布センター (KDC)** と多数のクライアントで構成される、Kerberos を使用するネットワークのことです。
- **KDCs** には、Kerberos チケットを発行するサーバーのコンマ区切りの一覧を入力します。
- **管理サーバー** には、レルム内で **kadmind** プロセスを実行している管理サーバーの一覧を入力します。
- オプションとして、DNS を使用してサーバーのホスト名を解決し、レルム内の新たな KDC を見つけることができます。

4.3.2. コマンドラインでの Kerberos 認証の設定

LDAP と NIS は両方とも、それらのネイティブな認証メカニズムの代わりに Kerberos 認証を使用することができます。Kerberos 認証を使用する場合、最低でもレルム、KDC、および管理サーバーを指定する必要があります。DNS を使用してクライアント名を解決、追加の管理サーバーを見つけるオプションもあります。

```
[root@server ~]# authconfig NIS or LDAP options --enablekrb5 --krb5realm  
EXAMPLE --krb5kdc kdc.example.com:88,server.example.com:88 --  
krb5adminserver server.example.com:749 --enablekrb5kdcdns --  
enablekrb5realmdns --update
```

4.4. スマートカード

パスワードベースの認証の代わりに、スマートカードベースの認証があります。ユーザーの認証情報がスマートカードに格納され、特別なソフトウェアやハードウェアを使用して、その情報にアクセスします。スマートカードを使用して認証するには、ユーザーはスマートカードリーダーにスマートカードを設置してから、そのスマートカードの PIN コードを提示する必要があります。



重要

以下のセクションでは、**pam_pkcs11** パッケージおよび **pam_krb5** パッケージを使用してローカルユーザーに対するスマートカード認証用のシングルシステムの設定方法について説明しています。『7.4 Release Notes』の [非推奨の機能](#) で説明しているように、これらのパッケージは現在は非推奨となっていることに留意してください。

スマートカード認証の一元化を設定するには、**System Security Services Daemon (SSSD)** が提供する強化されたスマートカードの機能を使用します。詳細は、Red Hat Identity Management の『Linux ドメイン ID、認証、およびポリシーガイド』にある [Identity Management でのスマートカード認証](#) を参照してください。

4.4.1. authconfig を使用したスマートカードの設定

スマートカードサポートを有効にする オプションを選択すると、スマートカードの設定動作を追加で制御できるようになります。

The image shows a 'Authentication Configuration' dialog box with three tabs: 'Identity & Authentication', 'Advanced Options' (which is selected), and 'Password Options'. Under the 'Advanced Options' tab, there are three sections: 'Local Authentication Options' with checkboxes for 'Enable fingerprint reader support' (checked) and 'Enable local access control' (unchecked), followed by a tip and a 'Password Hashing Algorithm' dropdown set to 'SHA512'; 'Other Authentication Options' with a checkbox for 'Create home directories on the first login' (unchecked); and 'Smart Card Authentication Options' with a checked 'Enable smart card support' checkbox, a tip, a 'Card Removal Action' dropdown set to 'Ignore', and an unchecked 'Require smart card for login' checkbox. At the bottom are 'Revert', 'Cancel', and 'Apply' buttons.

Authentication Configuration

Identity & Authentication **Advanced Options** Password Options

Local Authentication Options

- ☒ Enable fingerprint reader support
- ☐ Enable local access control

Tip: This is managed via `/etc/security/access.conf`.

Password Hashing Algorithm: SHA512 ▼

Other Authentication Options

- ☐ Create home directories on the first login

Smart Card Authentication Options

- ☒ Enable smart card support

Tip: Smart cards support logging into both local and centrally managed accounts.

Card Removal Action: Ignore ▼

- ☐ Require smart card for login

Revert Cancel Apply

図4.3 スマートカード認証のオプション

Red Hat Enterprise Linux サーバーおよびワークステーションにおけるスマートカードでのログインはデフォルトでは有効になっておらず、システム設定で有効にする必要があります。



注記

Red Hat Enterprise Linux へのログイン時にシングルサインオンを使用する場合は、以下のパッケージが必要になります。

- nss-tools
- nss-pam-ldapd
- esc
- pam_pkcs11
- pam_krb5
- opensc
- pcsc-lite-ccid
- gdm
- authconfig
- authconfig-gtk
- krb5-libs
- krb5-workstation
- krb5-pkinit
- pcsc-lite
- pcsc-lite-libs

4.4.1.1. UIでのスマートカード認証の有効化

1. **root** でシステムにログインします。
2. ネットワーク用の **root CA** 証明書をベース 64 形式でダウンロードし、サーバーにインストールします。**certutil** コマンドを使うと、証明書は適切なシステムデータベースにインストールされます。例を示します。

```
[root@server ~]# certutil -A -d /etc/pki/nssdb -n "root CA cert" -t "CT,C,C" -i /tmp/ca_cert.crt
```



注記

このプロセスの後半で **authconfig** UI にインポートされた証明書が表示されなくても、心配はいりません。UI では証明書は表示されません。認証時に **/etc/pki/nssdb/** ディレクトリーから取得されます。

3. トップメニューでアプリケーションから **諸ツール** を選択し、**認証** をクリックします。
4. **高度なオプション** タブを開きます。

5. スマートカードサポートを有効にする チェックボックスをクリックします。

6. スマートカードでは 2 つの動作が設定可能です。

- **カード削除のアクション** では、アクティブセッション中にカードが取り出された時のシステムの対応方法を設定します。**無視する** オプションの場合、カードが取り出されてもシステムは通常の機能を続けます。**ロックする** の場合は直ちに画面をロックします。
- **スマートカードログインを要求** のチェックボックスでは、スマートカードがログインで必要かどうかを設定します。このオプションが選択されると、他の認証メソッドはすべてブロックされます。



警告

スマートカードを使用してシステムに正常にログインする までは、このオプションは選択しないでください。

7. デフォルトでは、証明書が失効となったかどうかを確認するメカニズム (オンライン証明書ステータスプロトコル、OCSP の反応) は、無効になっています。有効期限が切れる前に証明書が失効したかどうかを検証するには、**cert_policy** ディレクティブに **ocsp_on** オプションを追加して OCSP のチェックを有効にします。

1. **pam_pkcs11.conf** ファイルを開きます。

```
vim /etc/pam_pkcs11/pam_pkcs11.conf
```

2. **cert_policy** 行すべてに **ocsp_on** オプションを追加します。

```
cert_policy = ca, ocsp_on, signature;
```



注記

このファイルの解析方法が原因で、**cert_policy** とイコール記号の間には空白が **必要になります**。これがないと、パラメーターの解析が失敗します。

8. (個人証明書とキーによる設定で) スマートカードが登録されていない場合、スマートカードを登録します。

9. スマートカードが CAC カードの場合、CAC ユーザーのホームディレクトリーに **.k5login** ファイルを作成します。**.k5login** ファイルは、CAC カード上に Microsoft Principal Name を記載するために必要となります。

10. 以下の行を **/etc/pam.d/smartcard-auth** と **/etc/pam.d/system-auth** の各ファイルに追加します。

```
auth    optional    pam_krb5.so use_first_pass no_subsequent_prompt
preauth_options=X509_user_identity=PKCS11:/usr/lib64/pkcs11/opensc-pkcs11.so
```

-

OpenSC モジュールが予想どおりに動作しない場合、**coolkey** パッケージのモジュール **/usr/lib64/pkcs11/libcoolkeypk11.so** を使用します。この場合、この問題について Red Hat テクニカルサポートへ問い合わせるか、**Bugzilla** に報告することを検討してください。

11. **/etc/krb5.conf** ファイルを設定します。この設定は、CAC カードか Gemalto 64K カードを使っているかによって異なります。

- CAC カードの場合、CAC カード使用に関連するすべての root 証明書を **pkinit_anchors** で指定します。以下の **/etc/krb5.conf** ファイルで CAC カードを設定する例では、**EXAMPLE.COM** が CAC カードのレルム名になり、**kdc.server.hostname.com** が KDC サーバーのホスト名になります。

```
[logging]
    default = FILE:/var/log/krb5libs.log
    kdc = FILE:/var/log/krb5kdc.log
    admin_server = FILE:/var/log/kadmind.log

[libdefaults]
    dns_lookup_realm = false
    dns_lookup_kdc = false
    ticket_lifetime = 1h
    renew_lifetime = 6h
    forwardable = true

    default_realm = EXAMPLE.COM
[realms]
    EXAMPLE.COM = {
        kdc = kdc.server.hostname.com
        admin_server = kdc.server.hostname.com
        pkinit_anchors = FILE:/etc/pki/nssdb/ca_cert.pem
        pkinit_anchors = FILE:/etc/pki/nssdb/CAC_CA_cert.pem
        pkinit_anchors = FILE:/etc/pki/nssdb/CAC_CA_email_cert.pem
        pkinit_anchors = FILE:/etc/pki/nssdb/CAC_root_ca_cert.pem
        pkinit_cert_match = CAC card specific information
    }

[domain_realm]
    EXAMPLE.COM = EXAMPLE.COM
    .EXAMPLE.COM = EXAMPLE.COM

    .kdc.server.hostname.com = EXAMPLE.COM
    kdc.server.hostname.com = EXAMPLE.COM

[appdefaults]
    pam = {
        debug = true
        ticket_lifetime = 1h
        renew_lifetime = 3h
        forwardable = true
        krb4_convert = false
        mappings = username on the CAC card      Principal name on
the card
    }
```

- Gemalto 64K カードを設定する以下の `/etc/krb5.conf` ファイルの場合、`EXAMPLE.COM` は KDC サーバー上で作成されたレルムになり、`kdc-ca.pem` は CA 証明書、`kdc.server.hostname.com` が KDC サーバーのホスト名になります。

```
[logging]
default = FILE:/var/log/krb5libs.log
kdc = FILE:/var/log/krb5kdc.log
admin_server = FILE:/var/log/kadmind.log

[libdefaults]
dns_lookup_realm = false
dns_lookup_kdc = false
ticket_lifetime = 15m
renew_lifetime = 6h
forwardable = true

default_realm = EXAMPLE.COM
[realms]
EXAMPLE.COM = {
    kdc = kdc.server.hostname.com
    admin_server = kdc.server.hostname.com
    pkinit_anchors = FILE:/etc/pki/nssdb/kdc-ca.pem
    pkinit_cert_match = <KU>digitalSignature
    pkinit_kdc_hostname = kdc.server.hostname.com
}

[domain_realm]
EXAMPLE.COM = EXAMPLE.COM
.EXAMPLE.COM = EXAMPLE.COM

.kdc.server.hostname.com = EXAMPLE.COM
kdc.server.hostname.com = EXAMPLE.COM

[appdefaults]
pam = {
    debug = true
    ticket_lifetime = 1h
    renew_lifetime = 3h
    forwardable = true
    krb4_convert = false
}
```

注記

スマートカードが挿入されると、**pklogin_finder** ユーティリティーがデバッグモードで実行されている場合、まずログイン ID をカード上の証明書にマッピングし、証明書の有効性についての情報の出力を試みます。

```
pklogin_finder debug
```

このコマンドは、スマートカードを使ってシステムにログインする際の問題を診断する上で役立ちます。

4.4.1.2. コマンドラインでのスマートカード認証の設定

システムでスマートカードを使用する際に必要となるのは **--enablesmartcard** オプションの設定のみです。

```
[root@server ~]# authconfig --enablesmartcard --update
```

スマートカードには、デフォルトのスマートカードモジュールの変更、スマートカードが取り出された時のシステムの動作のセッティング、ログイン時にスマートカードを要求するなど、他の設定オプションがあります。

以下のコマンドの値 **0** は、スマートカードが取り出された場合、システムがユーザーをロックアウトするように指示します。設定が **1** の場合は、スマートカードが取り出されても、これを無視します。

```
[root@server ~]# authconfig --enablesmartcard --smartcardaction=0 --update
```

スマートカード認証が正常に設定されテストが終わると、単純なパスワードベースの認証ではなく、スマートカード認証をユーザーに要求するようにシステムを設定することができます。

```
[root@server ~]# authconfig --enablerequiresmartcard --update
```



警告

スマートカードを使用して正常にシステム認証が終了するまで、**--enablerequiresmartcard** オプションは使用しないでください。終了前に使用すると、ユーザーがシステムにログインできなくなる可能性があります。

4.4.2. Identity Management でのスマートカード認証

Red Hat Identity Management は、IdM ユーザーのスマートカード認証をサポートします。詳細情報は [Linux ドメイン ID、認証、およびポリシーガイド](#) を参照してください。

4.4.3. 対応するスマートカード

以下のスマートカードおよびリーダーは、Red Hat Enterprise Linux に対応しています。

スマートカード

- Athena ASECard Crypto smart, pkcs15-unit
- ATOS (Siemens) CardOS 5.0
- Gemalto ID Classic 230 / TOP IM CY2 64kv2
- Gemalto Cyberflex Access 64k V2c
- Gemalto GemPCKey USB form factor key
- Giesecke & Devrient (G&D) SmartCafe Expert 6.0 (SCP03)
- Giesecke & Devrient (G&D) SmartCafe Expert 7.0 (SCP03)

- Safenet 330J
- Safenet SC650 (SCP01)
- Siemens Card CardOS M4.4
- Yubikey 4

リーダー

- HP Keyboard KUS1206 with built in Smart Card reader
- Omnikey 3121 reader
- Omnikey 3121 with PID 0x3022 reader
- Reiner SCT cyberJack RFID komfort reader
- SCR331 CCID reader

4.5. ワンタイムパスワード

ワンタイムパスワード (OTP) は、1 回の認証セッションのみで有効なパスワードです。これは一度使用すると、無効になります。長期間に渡り変わることがない従来の静的なパスワードと違って、OTP は常に変更されます。OTP は 2 要素認証の一部として使用されます。まず、ユーザーは従来の静的パスワードで認証を行い、次に認識された認証トークンが発行する OTP が求められます。

静的パスワードと OTP を組み合わせた認証は、静的パスワードのみの認証よりも安全だと考えられています。正常な認証に OTP が使えるのは 1 度だけなので、ログイン中に侵入者が OTP を傍受したとしても、その OTP はその時点ですでに無効になっているためです。

Red Hat Enterprise Linux でのワンタイムパスワード

Red Hat Identity Management は、IdM ユーザーの OTP 認証をサポートします。詳細情報は [Linux ドメイン ID、認証、およびポリシーガイド](#) を参照してください。

4.6. AUTHCONFIG を使用した指紋の設定

4.6.1. UI での指紋認証の使用

適切なハードウェアが利用可能である場合は、他の認証情報に加えて **指紋リーダーサポートを有効にする** オプションでローカルユーザーの認証に指紋スキャンを使用することができます。

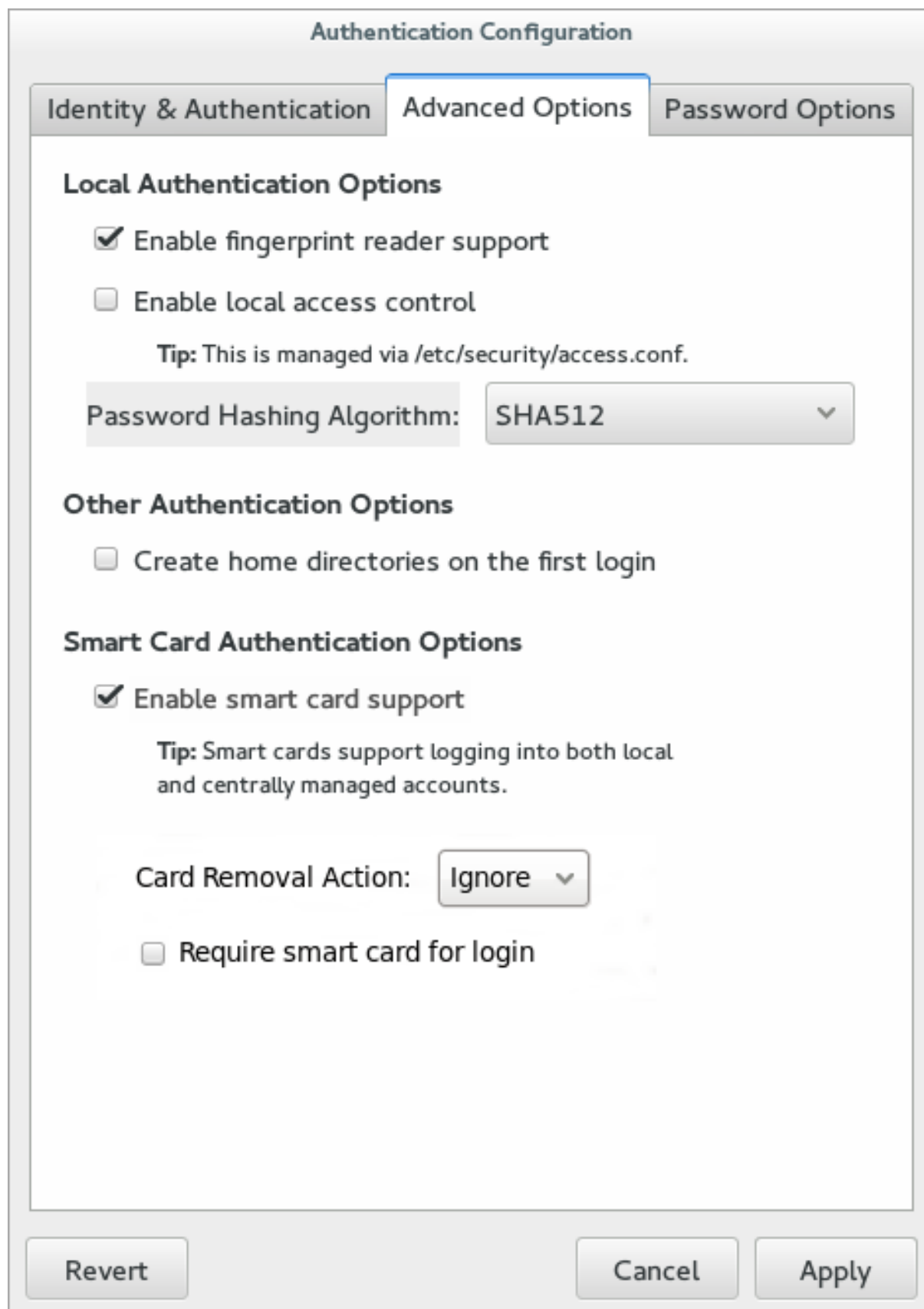


図4.4 指紋オプション

4.6.2. コマンドラインでの指紋認証の設定

指紋リーダーのサポートを有効にするには、オプションが1つあります。このオプションは単独で、または LDAP ユーザストアのような他の **authconfig** 設定と共に使用することができます。

```
[root@server ~]# authconfig --enablefingerprint --update
```

第5章 AUTHCONFIG を使用したキックスタートと設定ファイルの管理

--update オプションは、変更されたすべての設定ファイルを更新します。以下の代替オプションでは、動作が多少異なります。

- **--kickstart** は、更新した設定をキックスタートファイルに書き込みます。
- **--test** は、変更後の設定全域を標準出力に表示しますが、設定ファイルはどれも編集しません。

また、**authconfig** を使用してバックアップを作成したり、以前の設定に復元することもできます。すべてのアーカイブは **/var/lib/authconfig/** ディレクトリー内の一意のサブディレクトリーに保存されます。たとえば、**--savebackup** オプションを使って **2011-07-01** といったバックアップディレクトリーを作成します。

```
[root@server ~]# authconfig --savebackup=2011-07-01
```

これにより、**/var/lib/authconfig/backup-2011-07-01** ディレクトリーに認証設定ファイルすべてのバックアップが作成されます。

保存したバックアップはどれも **--restorebackup** オプションで手動保存した設定の名前を入力して、設定の復元に使用できます。

```
[root@server ~]# authconfig --restorebackup=2011-07-01
```

また、**authconfig** は、変更を (**--update** オプションで) 適用する前に、自動的に設定のバックアップを作成します。**--restorelastbackup** オプションを使用すると、バックアップを指定しない場合は最新の自動バックアップから設定を復元できます。

第6章 AUTHCONFIG を使用したカスタムホームディレクトリーの有効化

LDAP ユーザーのホームディレクトリーが **/home** 以外の場所にあり、かつユーザーの初回ログイン時にシステムがホームディレクトリーを作成するように設定されている場合は、これらのディレクトリーは間違ったパーミッションで作成されてしまいます。

1. **/home** ディレクトリーから正しい SELinux コンテキストとパーミッションをローカルシステム上で作成されたホームディレクトリーに適用します。例を示します。

```
[root@server ~]# semanage fcontext -a -e /home /home/locale
```

2. **oddjob-mkhomedir** パッケージをシステムにインストールします。

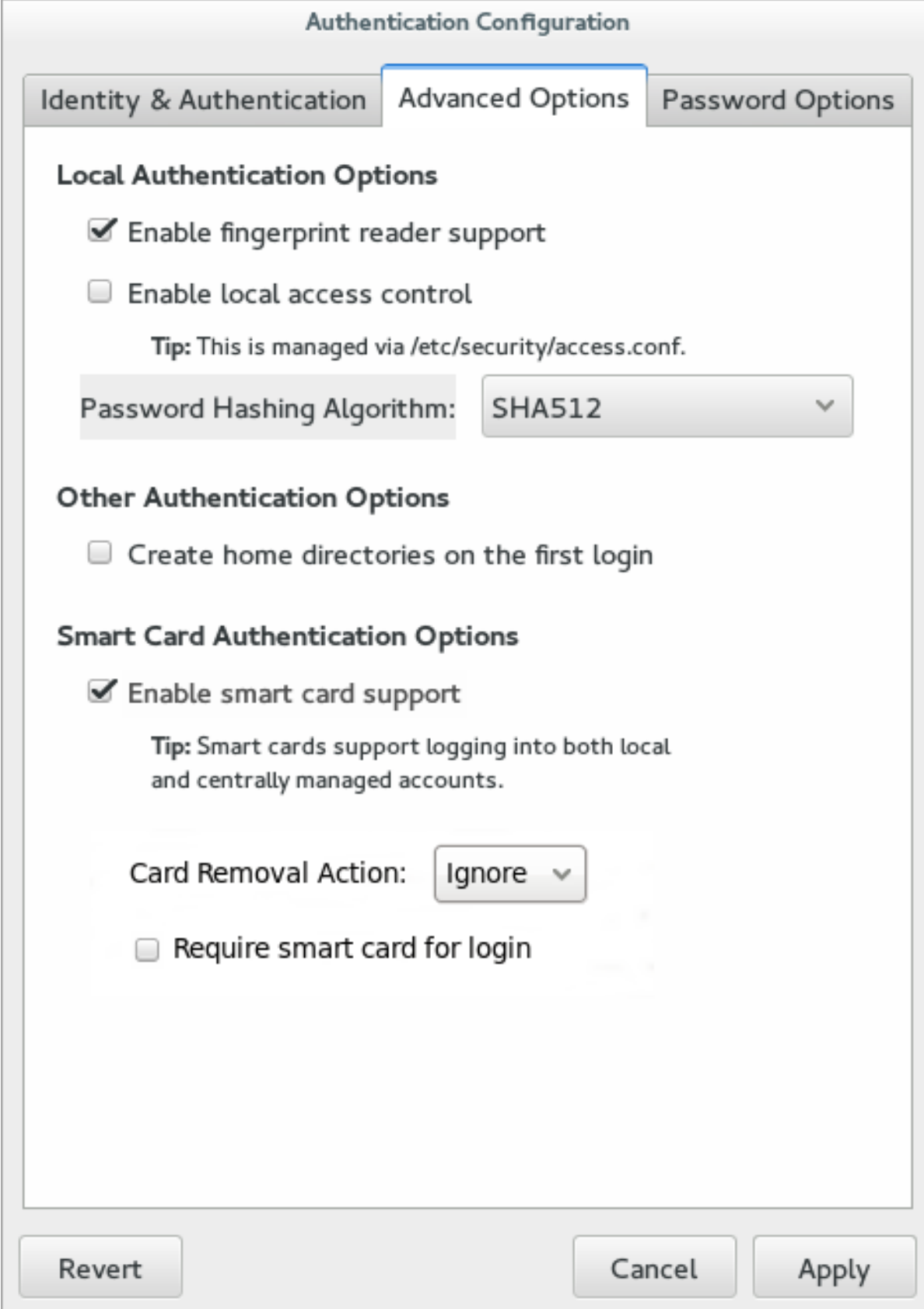
このパッケージは、**pam_oddjob_mkhomedir.so** ライブラリーを提供し、**authconfig** コマンドはこのライブラリーを使用してホームディレクトリーを作成します。デフォルトの **pam_mkhomedir.so** ライブラリーとは異なり、**pam_oddjob_mkhomedir.so** ライブラリーは SELinux ラベルを作成できます。

authconfig コマンドは、**pam_oddjob_mkhomedir.so** ライブラリーが利用可能な場合には、自動的にこれを使用します。利用可能でない場合には、デフォルトの **pam_mkhomedir.so** ライブラリーを使用します。

3. **oddjobd** サービスが実行されていることを確認します。
4. **authconfig** コマンドを実行して、ホームディレクトリーを有効にします。コマンドラインでは、**--enablemkhomedir** オプションを使うとこれが実行できます。

```
[root@server ~]# authconfig --enablemkhomedir --update
```

UI では、**高度なオプション タブ (利用者の最初のログイン時にホームディレクトリーを作成する)** オプションを選択するとユーザーの初回ログイン時に自動的にホームディレクトリーが作成されます。



The image shows a 'Authentication Configuration' dialog box with three tabs: 'Identity & Authentication', 'Advanced Options' (which is selected), and 'Password Options'. Under the 'Advanced Options' tab, there are three sections: 'Local Authentication Options', 'Other Authentication Options', and 'Smart Card Authentication Options'. In 'Local Authentication Options', 'Enable fingerprint reader support' is checked, 'Enable local access control' is unchecked, and a tip states it is managed via /etc/security/access.conf. The 'Password Hashing Algorithm' is set to 'SHA512'. In 'Other Authentication Options', 'Create home directories on the first login' is unchecked. In 'Smart Card Authentication Options', 'Enable smart card support' is checked, with a tip stating it supports logging into both local and centrally managed accounts. The 'Card Removal Action' is set to 'Ignore', and 'Require smart card for login' is unchecked. At the bottom are 'Revert', 'Cancel', and 'Apply' buttons.

Authentication Configuration

Identity & Authentication | **Advanced Options** | Password Options

Local Authentication Options

- ☒ Enable fingerprint reader support
- ☐ Enable local access control

Tip: This is managed via /etc/security/access.conf.

Password Hashing Algorithm: SHA512

Other Authentication Options

- ☐ Create home directories on the first login

Smart Card Authentication Options

- ☒ Enable smart card support

Tip: Smart cards support logging into both local and centrally managed accounts.

Card Removal Action: Ignore

- ☐ Require smart card for login

Revert Cancel Apply

図6.1 ホームディレクトリーオプション

このオプションは、LDAP などアカウントを集中的に管理している場合に便利です。しかし、ユーザーのホームディレクトリー管理に `automount` のようなシステムを使用している場合は、このオプションを選択しないでください。

ホームディレクトリーの設定変更前にホームディレクトリーが作成された場合は、パーミッションと SELinux コンテキストを訂正します。例を示します。

```
[root@server ~]# semanage fcontext -a -e /home /home/locale  
# restorecon -R -v /home/locale
```

パート II. ID と認証ストア

第7章 SSSD の設定

7.1. SSSD について

7.1.1. SSSD の仕組み

System Security Services Daemon (SSSD) は、リモートディレクトリーおよび認証メカニズムへアクセスするシステムサービスです。このサービスは、ローカルシステム (**SSSD client**) を外部のバックエンドシステム (**provider**) に接続します。これにより、**SSSD クライアント**は、**SSSD プロバイダー**を使用してアイデンティティおよび認証リモートサービスへアクセスできます。たとえば、これらのリモートサービスには、**LDAP ディレクトリー**、**Identity Management (IdM)** または **Active Directory (AD) ドメイン**、もしくは **Kerberos レalm**などがあります。

このような目的で **SSSD** は以下を実行します。

1. クライアントを ID ストアに接続し、認証情報を取得します。
2. 取得した認証情報を使用して、クライアントにユーザーと認証情報のローカルキャッシュを作成します。

続いて、ローカルシステムのユーザーは、外部のバックエンドシステムに保存されたユーザーアカウントを使用して認証ができます。

SSSD は、ローカルシステム上にユーザーアカウントを作成しません。代わりに、外部のデータストアのアイデンティティを使用して、ユーザーがローカルシステムにアクセスできるようにします。



図7.1 SSSD の仕組み

SSSD は、**Name Service Switch (NSS)** またはプラグ可能な認証モジュール (**PAM**) などの、いくつかのシステムサービスにキャッシュを提供することもできます。

7.1.2. SSSD を使用する利点

アイデンティティおよび認証サーバーへの負荷を軽減

情報をリクエストする際、**SSSD クライアント**は **SSSD** に連絡し、**SSSD** はキャッシュを確認します。**SSSD** は、キャッシュに情報がない場合だけサーバーに連絡します。

オフライン認証

SSSD はオプションで、リモートサービスから取得したユーザー ID および認証情報のキャッシュを保持します。このセットアップでは、リモートサーバーまたは **SSSD クライアント**がオフラインであっても、リソースに正常に認証することができます。

認証プロセスの一貫性が改善された単一ユーザーアカウント

SSSD があれば、オフライン認証用に中央アカウントとローカルユーザーアカウントの両方を維持する必要はありません。

多くの場合、リモートユーザーは複数のユーザーアカウントを持っています。たとえば、仮想プライベートネットワーク (VPN) に接続するためには、リモートユーザーはローカルシステム用のアカウントのほか、VPN システム用のアカウントも持っています。

キャッシングおよびオフライン認証により、リモートユーザーは自身のローカルマシンに対する認証を行うだけでネットワークリソースに接続できます。その後は SSSD がそれらのネットワーク認証情報を維持します。

7.2. クライアントごとに複数の SSSD 設定ファイルを使用

SSSD のデフォルトの設定ファイルは、`/etc/sss/sss.conf` です。このファイルとは別に、SSSD は `/etc/sss/conf.d/` ディレクトリー内のすべての `*.conf` ファイルの設定を読み取ることができます。

たとえば、これにより、すべてのクライアント上にあるデフォルトの `/etc/sss/sss.conf` ファイルを使用して、別の設定ファイルに追加設定を追加し、クライアントごとに機能を個別に拡張することができます。

SSSD が設定ファイル进行处理する方法

SSSD は、以下の順番で設定ファイルを読み取ります。

1. プライマリー `/etc/sss/sss.conf` ファイル
2. `/etc/sss/conf.d/` 内にあるその他の `*.conf` ファイルをアルファベット順で読み取り

同じパラメーターが複数の設定ファイルに出現すると、SSSD は最後に読み取ったパラメーターを使用します。



注記

SSSD は、`conf.d` ディレクトリー内の隠しファイル (`.` で始まるファイル) は読み取りません。

7.3. SSSD 向け ID プロバイダーと認証プロバイダーの設定

7.3.1. SSSD の ID プロバイダーと認証プロバイダーについて

SSSD ドメイン ID プロバイダーと認証プロバイダー

ID および認証プロバイダーは、SSSD 設定ファイル内で `domains` として設定されます。単一ドメインは、以下のように使用することができます。

- アイデンティティープロバイダー(ユーザー情報向け)
- 認証プロバイダー(認証リクエスト向け)
- アクセス制御プロバイダー(承認リクエスト向け)
- これらのプロバイダーの組み合わせ (対応するすべての操作が単一サーバー内で実行される場合)

SSSD 用に複数のドメインを設定できます。少なくとも1つのドメインを設定する必要があります。そうでなければ、SSSD は開始しません。

`/etc/sss/sss.conf` ファイル内の `access_provider` オプションが、ドメイン用に使われるアクセス制御プロバイダーを設定します。デフォルトで、オプションは `permit` に設定されます。これにより、すべてのアクセスが常に許可されます。詳細は、`sss.conf(5)` の `man` ページを参照してください。

プロキシプロバイダー

プロキシプロバイダーは、SSSD とリソースとの間の中間的中継点の役割を持っています。このリソースがなければ SSSD は使用できません。プロキシプロバイダーを使用する際、SSSD はプロキシサービスに接続し、プロキシは指定されたライブラリーを読み込みます。

プロキシプロバイダーを使用して SSSD を設定すると、以下を使用できます。

- 指紋スキャナーなどの別の認証メソッド
- NIS などのレガシーシステム
- `/etc/passwd` で定義されたローカルシステムアカウントおよびリモート認証

ID および認証プロバイダーの利用可能な組み合わせ

表7.1 ID および認証プロバイダーの利用可能な組み合わせ

アイデンティティプロバイダー	認証プロバイダー
Identity Management [a]	Identity Management [a]
Active Directory [a]	Active Directory [a]
LDAP	LDAP
LDAP	Kerberos
proxy	proxy
proxy	LDAP
proxy	Kerberos
[a] LDAP プロバイダータイプの拡張子	

本ガイドでは、すべてのプロバイダータイプを説明しているわけではないことに留意してください。詳細情報は、以下を参照してください。

- Identity Management の SSSD クライアントを設定するには、Red Hat は `ipa-client-install` ユーティリティの使用を推奨しています。『Linux ドメイン ID、認証、およびポリシーガイド』の [Identity Management クライアントのインストールおよびアンインストール](#) を参照してください。

- **ipa-client-install** を使用せずに Identity Management の SSSD クライアントを手動で設定するには、Red Hat ナレッジベースの [Installing and Uninstalling an Identity Management Client Manually](#) を参照してください。
- SSSD と共に使用する Active Directory の設定については、『Windows 統合ガイド』の [Active Directory を SSSD のアイデンティティプロバイダーとして使用する](#) を参照してください。

7.3.2. SSSD 向け LDAP ドメインの設定

前提条件

- SSSD をインストールします。

```
# yum install sssd
```

SSSD を設定して LDAP ドメインを発見

1. `/etc/sss/sss.conf` ファイルを開きます。
2. LDAP ドメイン用に **[domain]** セクションを作成します。

```
[domain/LDAP_domain_name]
```

3. LDAP サーバーをアイデンティティプロバイダーとして、または認証プロバイダーとして使用したいのか、もしくは両方として使用したいのかを指定します。
 - a. LDAP サーバーをアイデンティティプロバイダーとして使用する場合は、**id_provider** オプションを **ldap** に設定します。
 - b. LDAP サーバーを認証プロバイダーとして使用する場合は、**auth_provider** オプションを **ldap** に設定します。

たとえば、LDAP サーバーを両方として使用する場合は、以下のとおりです。

```
[domain/LDAP_domain_name]
id_provider = ldap
auth_provider = ldap
```

4. LDAP サーバーを指定します。以下から1つ選択します。
 - a. サーバーを明示的に定義するには、サーバーの URI を **ldap_uri** オプションで指定します。

```
[domain/LDAP_domain_name]
id_provider = ldap
auth_provider = ldap

ldap_uri = ldap://ldap.example.com
```

ldap_uri オプションは、サーバーの IP アドレスも受信します。しかし、サーバー名の代わりに IP アドレスを使用すると、TLS/SSL 接続の失敗につながる場合があります。Red Hat ナレッジベースの [Configuring an SSSD Provider to Use an IP Address in the Certificate Subject Name](#) を参照してください。

- b. SSSD を設定し、DNS service discovery を動的に使用してサーバーを発見するには、「[DNS Service Discovery の設定](#)」を参照してください。

オプションとして、**ldap_backup_uri** オプションにおいてもバックアップサーバーを指定します。

5. **ldap_search_base** オプションで、LDAP サーバーの検索ベースを指定します。

```
[domain/LDAP_domain_name]
id_provider = ldap
auth_provider = ldap

ldap_uri = ldap://ldap.example.com
ldap_search_base = dc=example,dc=com
```

6. LDAP サーバーへの安全な接続を確立するための方法を指定します。推奨されているのは、TLS 接続を使用する方法です。そのためには、**ldap_id_use_start_tls** オプションを有効にし、これらの CA 証明書関連のオプションを使用します。

- **ldap_tls_reqcert** は、クライアントがサーバー証明書を要求するかどうか、そしてその証明書でどのような確認が行われるのかを指定します。
- **ldap_tls_cacert** は、証明書を含むファイルを指定します。

```
[domain/LDAP_domain_name]
id_provider = ldap
auth_provider = ldap

ldap_uri = ldaps://ldap.example.com
ldap_search_base = dc=example,dc=com

ldap_id_use_start_tls = true
ldap_tls_reqcert = demand
ldap_tls_cacert = /etc/pki/tls/certs/ca-bundle.crt
```



注記

SSSD は常に、認証用に暗号化されたチャンネルを使用します。これにより、パスワードが暗号化されずにネットワーク上に送信されることが決していないようにします。**ldap_id_use_start_tls = true** により、アイデンティティルックアップ (**id** または **getent** のユーティリティーをベースとするコマンドなど) も暗号化されます。

7. **[sssd]** セクションの **domains** オプションに新しいドメインを追加します。オプションは、SSSD がクエリーするドメインを一覧表示します。例を示します。

```
domains = LDAP_domain_name, domain2
```

その他のリソース

上記の手順は、LDAP プロバイダー向けの基本的なオプションを示しています。詳細については以下を参照してください。

- `sssd.conf(5)` の `man` ページでは、すべてのドメインタイプで利用可能なグローバルオプションを説明しています。
- `sssd-ldap(5)` の `man` ページでは、LDAP 固有のオプションを説明しています。

7.3.3. SSSD 向けプロキシープロバイダーの設定

前提条件

- SSSD をインストールします。

```
# yum install sssd
```

SSSD を設定してプロキシードメインを発見

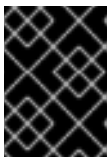
1. `/etc/sss/sss.conf` ファイルを開きます。
2. プロキシープロバイダー用に `[domain]` セクションを作成します。

```
[domain/proxy_name]
```

3. 認証プロバイダーを特定するには、以下を実行します。
 - a. **`auth_provider`** オプションを **`proxy`** に設定します。
 - b. **`proxy_pam_target`** オプションを使用して、PAM サービスを認証プロキシーに指定します。

例を示します。

```
[domain/proxy_name]
auth_provider = proxy
proxy_pam_target = sssdpamproxy
```



重要

プロキシー PAM スタックに **`pam_sss.so`** が再帰的に **格納されていない** ことを確認してください。

4. アイデンティティプロバイダーを指定するには、以下を実行します。
 - a. **`id_provider`** オプションを **`proxy`** に設定します。
 - b. **`proxy_lib_name`** オプションを使用して、NSS ライブラリーをアイデンティティプロキシーに指定します。

例を示します。

```
[domain/proxy_name]
id_provider = proxy
proxy_lib_name = nis
```

5. **[sssd]** セクションの **domains** オプションに新しいドメインを追加します。オプションは、SSSD がクエリーするドメインを一覧表示します。例を示します。

```
domains = proxy_name, domain2
```

その他のリソース

上記の手順は、プロキシプロバイダーの基本的なオプションを示しています。詳細については、**sssd.conf(5)** の **man** ページを参照してください。ここでは、すべてのドメインタイプで利用可能なグローバルオプションと、その他のプロキシ関連オプションについて説明しています。

7.3.4. Kerberos 認証プロバイダーの設定

前提条件

- SSSD をインストールします。

```
# yum install sssd
```

SSSD の設定による Kerberos ドメインの発見

1. **/etc/sss/sss.conf** ファイルを開きます。
2. SSSD ドメイン向けに **[domain]** セクションを作成

```
[domain/Kerberos_domain_name]
```

3. アイデンティティプロバイダーを指定します。たとえば、LDAP アイデンティティプロバイダーの設定に関する詳細は、「[SSSD 向け LDAP ドメインの設定](#)」を参照してください。

指定したアイデンティティプロバイダーで Kerberos のプリンシパル名を利用できない場合、SSSD は **username@REALM** 形式を使用してプリンシパルを構築します。

4. Kerberos 認証プロバイダーの詳細を指定します。

- a. **auth_provider** オプションを **krb5** に設定します。

```
[domain/Kerberos_domain_name]
id_provider = ldap
auth_provider = krb5
```

- b. Kerberos サーバーを指定します。

- i. サーバーを明示的に定義するには、**krb5_server** オプションを使用します。オプションは、サーバーのホスト名または IP アドレスを受信します。

```
[domain/Kerberos_domain_name]
id_provider = ldap
auth_provider = krb5

krb5_server = kdc.example.com
```

- ii. SSSD を設定し、DNS service discovery を動的に使用してサーバーを発見するには、「[DNS Service Discovery の設定](#)」を参照してください。

オプションとして、**krb5_backup_server** オプションにおいてもバックアップサーバーを指定します。

- c. パスワード変更サービスが、**krb5_server** または **krb5_backup_server** で指定した KDC 上で稼働していない場合、**krb5_passwd** オプションを使用してサーバーにサービスが稼働している場所を指定します。

```
[domain/Kerberos_domain_name]
id_provider = ldap
auth_provider = krb5

krb5_server = kdc.example.com
krb5_backup_server = kerberos.example.com
krb5_passwd = kerberos.admin.example.com
```

krb5_passwd が使用されていない場合、SSSD は **krb5_server** または **krb5_backup_server** で指定された KDC を使用します。

- d. **krb5_realms** オプションを使用して、Kerberos レalm 名を指定します。

```
[domain/Kerberos_domain_name]
id_provider = ldap
auth_provider = krb5

krb5_server = kerberos.example.com
krb5_backup_server = kerberos2.example.com
krb5_passwd = kerberos.admin.example.com
krb5_realms = EXAMPLE.COM
```

5. **[sssd]** セクションの **domains** オプションに新しいドメインを追加します。オプションは、SSSD がクエリーするドメインを一覧表示します。例を示します。

```
domains = Kerberos_domain_name, domain2
```

その他のリソース

上記の手順は、Kerberos プロバイダー向けの基本的なオプションを示しています。詳細については以下を参照してください。

- **sssd.conf(5)** の **man** ページでは、すべてのドメインタイプで利用可能なグローバルオプションを説明しています。
- **sssd-krb5(5)** の **man** ページでは、Kerberos 固有のオプションを説明しています。

7.4. アイデンティティプロバイダーと認証プロバイダー向け追加的設定

7.4.1. ユーザー名形式の調整

7.4.1.1. 完全なユーザー名の解析向け正規表現の定義

SSSD は、完全なユーザー名のストリングをユーザー名およびドメインのコンポーネントに解析します。SSSD はデフォルトにより、以下の Python シンタックスの正規表現を基本とする **user_name@domain_name** 形式で完全なユーザー名を解釈します。

■

```
(?P<name>[^\@]+)@?(?P<domain>[^\@]*$)
```



注記

Identity Management および Active Directory プロバイダーの場合、デフォルトによるユーザー名の形式は **user_name@domain_name** または **NetBIOS_name\user_name** です。

SSSD が完全なユーザー名を解釈する方法を調整するには、以下を実行します。

1. **/etc/sss/sss.conf** ファイルを開きます。
2. **re_expression** オプションを使用して、カスタム正規表現を定義します。
 - a. すべてのドメイン向けに正規表現をグローバルに定義するには、**re_expression** を **sss.conf** の **[sss]** セクションに追加します。
 - b. 特定のドメイン向けに正規表現を個々に定義するには、**re_expression** を **sss.conf** の対応するドメインセクションに追加します。

たとえば、LDAP ドメイン向けに正規表現を設定するには、以下を実行します。

```
[domain/LDAP]
[... file truncated ...]
re_expression = (?P<domain>[^\@]*?)\@?(?P<name>[^\@]+)$
```

詳細については、**sss.conf(5)** の man ページにある **SPECIAL SECTIONS** および **DOMAIN SECTIONS** 部分の **re_expression** を参照してください。

7.4.1.2. SSSD による完全なユーザー名のプリント方法の定義

/etc/sss/sss.conf ファイル内で **use_fully_qualified_names** オプションが有効となっている場合、SSSD はデフォルトにより、以下の展開を基本とする **name@domain** 形式で完全なユーザー名をプリントします。

```
%1$s@%2$s
```



注記

use_fully_qualified_names が設定されていない場合、または信頼できるドメイン向けに **false** に明示的に設定されている場合、ドメインコンポーネントなしにユーザー名のみがプリントされます。

SSSD が完全なユーザー名をプリントする形式を調整するには、以下を実行します。

1. **/etc/sss/sss.conf** ファイルを開きます。
2. **full_name_format** オプションを使用して、完全なユーザー名形式の展開を定義します。
 - a. すべてのドメイン向けの展開をグローバルに定義するには、**full_name_format** を **sss.conf** の **[sss]** セクションに追加します。

- b. 特定のドメイン向けに展開を個々に定義するには、**full_name_format** を **sssd.conf** の対応するドメインセクションに追加します。

詳細については、**sssd.conf(5)** の man ページにある **SPECIAL SECTIONS** および **DOMAIN SECTIONS** 部分の **full_name_format** を参照してください。

名前の設定によっては、**SSSD** が名前のドメインコンポーネントを削除し、認証エラーが発生する場合があります。このため、**full_name_format** を標準以外の値に設定すると、より標準的な値への変更を促す警告が表示されます。

7.4.2. オフライン認証の有効化

SSSD はデフォルトで、ユーザーの認証情報をキャッシュしません。認証要求の処理の際、**SSSD** は常にアイデンティティプロバイダーと連絡を取ります。プロバイダーと連絡が取れない場合、ユーザー認証に失敗することになります。



重要

SSSD はプレーンテキストでパスワードをキャッシュすることはありません。ハッシュ化されたパスワードのみを保存します。

アイデンティティプロバイダーを利用できない場合であっても、ユーザーが確実に認証できるようにするために、認証情報のキャッシングを有効にします。

1. **/etc/sss/sss.conf** ファイルを開きます。
2. ドメインセクションにおいて、**cache_credentials = true** 設定を追加します。

```
[domain/domain_name]
cache_credentials = true
```

3. **推奨 (オプション)** アイデンティティプロバイダーを利用できない場合、**SSSD** がオフライン認証を許容する制限時間を設定します。
 - a. **SSSD** と機能する **PAM** サービスを設定します。「[サービスの設定: PAM](#)」を参照してください。
 - b. **offline_credentials_expiration** オプションを使用して制限時間を指定します。たとえば、ユーザーが最後にログインした時から **3 日間** はオフラインでの認証が可能と指定する場合、以下ようになります。

```
[pam]
offline_credentials_expiration = 3
```

offline_credentials_expiration の詳細については、**sssd.conf(5)** の man ページを参照してください。

7.4.3. DNS Service Discovery の設定

アイデンティティまたは認証サーバーが **/etc/sss/sss.conf** ファイルで明示的に定義されていない場合、**SSSD** は **DNS service discovery** を使用してサーバーを動的に発見することができます。^[1]

たとえば、**sssd.conf** に **id_provider = ldap** 設定が格納されているが、**ldap_uri** オプションがホスト名または IP アドレスを何も指定しない場合、SSSD は **DNS service discovery** を使用してサーバーを動的に発見します。



注記

SSSD はバックアップサーバーを動的に発見することはできません。プライマリーサーバーのみの発見となります。

DNS Service Discovery 用に SSSD を設定

1. **/etc/sss/sss.conf** ファイルを開きます。
2. プライマリーサーバー値を **_srv_** に設定します。LDAP プロバイダーの場合、**ldap_uri** オプションを使用してプライマリーサーバーを設定します。

```
[domain/domain_name]
id_provider = ldap
ldap_uri = _srv_
```

3. サービスタイプを設定することで、パスワード変更プロバイダーで **service discovery** を有効にします。

```
[domain/domain_name]
id_provider = ldap
ldap_uri = _srv_

chpass_provider = ldap
ldap_chpass_dns_service_name = ldap
```

4. オプション デフォルトでは、**service discovery** はシステムホスト名のドメイン部分をドメイン名に使用します。別の DNS ドメインを使用するには、**dns_discovery_domain** オプションでドメイン名を指定します。
5. オプション デフォルトでは、**service discovery** は LDAP サービスタイプをスキャンします。別のサービスタイプを使用するには、**ldap_dns_service_name** オプションでタイプを指定します。
6. オプション デフォルトでは、SSSD は IPv4 アドレスのルックアップを試みます。この試みに失敗した場合、SSSD は IPv6 アドレスのルックアップを試みます。この動作をカスタマイズするには、**lookup_family_order** オプションを使用します。詳細は、**sss.conf(5)** の **man** ページを参照してください。
7. **service discovery** を使用したいすべてのサービスで、DNS レコードを DNS サーバーに追加します。

```
_service._protocol._domain TTL priority weight port host_name
```

7.4.4. simple アクセスプロバイダーを使用したアクセス制御の定義

simple アクセスプロバイダーは、ユーザー名またはグループのリストを基にアクセスを許可または拒否します。つまり、特定のマシンへのアクセスを制限することができます。

たとえば、企業のノートパソコンに **simple** アクセスプロバイダーを使用すると、特定のユーザーまたは特定のグループのみに限定したアクセスが可能となります。他のユーザーまたはグループは、設定された認証プロバイダーに対して正しく認証されても、ログインできません。

simple アクセスプロバイダーのルールの設定

1. `/etc/sss/sss.conf` ファイルを開きます。
2. `access_provider` オプションを **simple** に設定します。

```
[domain/domain_name]
access_provider = simple
```

3. ユーザーのアクセス制御のルールを定義します。以下のうち1つを選択してください。
 - a. ユーザーのアクセスを許可する場合、**simple_allow_users** オプションを使用します。
 - b. ユーザーのアクセスを拒否する場合、**simple_deny_users** オプションを使用します。



重要

特定のユーザーにアクセスを許可する方が、拒否するよりも安全だと考えられています。特定のユーザーへのアクセスを拒否すると、自動的にその他全員に対してアクセスを許可することになります。

4. グループのアクセス制御のルールを定義します。以下のうち1つを選択してください。
 - a. グループのアクセスを許可するには、**simple_allow_groups** オプションを使用します。
 - b. グループのアクセスを拒否するには、**simple_deny_groups** オプションを使用します。



重要

特定のグループにアクセスを許可する方が、拒否するよりも安全だと考えられています。特定のグループへのアクセスを拒否すると、自動的にその他全員に対してアクセスを許可することになります。

以下の例では、**user1**、**user2** および **group1** のメンバーはアクセスが許可され、その他すべてのユーザーはアクセスを拒否されています。

```
[domain/domain_name]
access_provider = simple
simple_allow_users = user1, user2
simple_allow_groups = group1
```

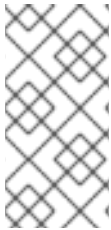
詳細は、`sss-simple(5)` の `man` ページを参照してください。

7.4.5. LDAP アクセスフィルターを使用したアクセス制御の定義

`access_provider` オプションが `/etc/sss/sss.conf` に設定されている場合、SSSD は指定されたアクセスプロバイダーを使用して、どのユーザーがシステムへのアクセスを許可されたのか判断します。使用中のアクセスプロバイダーが、LDAP プロバイダータイプの拡張子の場合、LDAP アクセス制

御フィルターも指定できます。ユーザーは、システムへのアクセス許可を得るためには、指定されたフィルターと一致しなければなりません。

たとえば、**Active Directory (AD)** サーバーをアクセスプロバイダーとして使用する場合、**Linux** システムへのアクセスを指定した **AD ユーザー** だけに限定することができます。指定したフィルターと一致しないその他すべてのユーザーは、アクセスを拒否されます。



注記

アクセスフィルターは、**LDAP ユーザーエントリ** のみに適用されます。したがって、このタイプのアクセス制御をネスト化されたグループに使用しても機能しない場合があります。アクセス制御をネスト化されたグループに適用するには、「[simple アクセスプロバイダーを使用したアクセス制御の定義](#)」を参照してください。



重要

オフラインキャッシュを使用する場合、**SSSD** はユーザーの前回のオンラインでのログイン試行が正常だったかどうかを確認します。前回のオンラインでのログインが正常だったユーザーは、アクセスフィルターと一致しなくても、オフラインでもそのままログインできます。

SSSD を設定し、LDAP アクセスフィルターを適用

1. `/etc/sss/sss.conf` ファイルを開きます。
2. **[domain]** セクションで、LDAP アクセス制御フィルターを指定します。
 - LDAP アクセスプロバイダーには、**`ldap_access_filter`** オプションを使用します。詳細については `sss-ldap(5)` の `man` ページを参照してください。
 - AD アクセスプロバイダーには、**`ad_access_filter`** オプションを使用します。詳細については、`sss-ad(5)` の `man` ページを参照してください。

たとえば、**admins** ユーザーグループに属し、**unixHomeDirectory** 属性セットを持つ **AD ユーザー** のみにアクセスを許可する場合、以下の設定を実行します。

```
[domain/AD_domain_name]
access_provider = ad
[... file truncated ...]
ad_access_filter = (&
(memberOf=cn=admins,ou=groups,dc=example,dc=com)
(unixHomeDirectory=*))
```

SSSD は、エントリ内の **authorizedService** や **host** 属性で結果をチェックすることもできます。実際にはユーザーエントリや設定によって、LDAP フィルター、**authorizedService**、**host** のすべての評価が可能です。**ldap_access_order** パラメーターは、使用するすべてのアクセス制御方法を評価方法の順番で表示します。

```
[domain/example.com]
access_provider = ldap
ldap_access_filter = memberOf=cn=allowedusers,ou=Groups,dc=example,dc=com
ldap_access_order = filter, host, authorized_service
```

認証済みサービスや許可されたホストの評価に使用するユーザーエントリ内の属性は、カスタマイズ

が可能です。追加のアクセス制御パラメーターは、**sssd-ldap(5)** の **man** ページに記載されています。

7.5. SSSD のシステムサービスの設定

SSSD は、複数のシステムサービスへのインターフェースを提供します。特に以下が知られています。

Name Service Switch (NSS)

「サービスの設定:**NSS**」を参照してください。

PAM (プラグ可能な認証モジュール)

「サービスの設定:**PAM**」を参照してください。

OpenSSH

『Linux ドメイン ID、認証、およびポリシーガイド』の「**SSSD が OpenSSH サービス用にキャッシュを提供するように設定する方法**」を参照してください。

autofs

「サービスの設定:**autofs**」を参照してください。

sudo

「サービスの設定:**sudo**」を参照してください。

7.5.1. サービスの設定: NSS

SSSD が NSS と機能する方法

Name Service Switch (NSS) サービスは、システムアイデンティティおよびサービスを設定ソースとマッピングします。つまり、サービスが様々な設定ソースおよび名前解決メカニズムをルックアップできる中央設定ストアを提供します。

SSSD は、数種類の NSS マップのプロバイダーとして、NSS を使用することができます。

- ユーザー情報 (**passwd** のマッピング)
- グループ (**groups** のマッピング)
- ネットグループ (**netgroups** のマッピング)
- サービス (**services** のマッピング)

前提条件

- SSSD をインストールします。

```
# yum install sssd
```

NSS サービスを設定して SSSD を使用

1. **authconfig** ユーティリティーを使用して SSSD を有効にします。

```
[root@server ~]# authconfig --enablesssd --update
```

■

これにより **/etc/nsswitch.conf** ファイルがアップデートされ、SSSD を使うために以下の NSS のマッピングが有効になります。

```
passwd:      files sss
shadow:      files sss
group:        files sss

netgroup:     files sss
```

2. **/etc/nsswitch.conf** を開いて、**sss** を **services** マッピングラインに追加します。

```
services: file sss
```

NSS と機能するように SSSD を設定

1. **/etc/sss/sss.conf** ファイルを開きます。
2. **[sss]** セクションでは、NSS が SSSD と共に機能するサービスの1つとして記載されていることを確認します。

```
[sss]
[... file truncated ...]
services = nss, pam
```

3. **[nss]** セクションでは、SSSD が NSS と対話する方法を設定します。以下に例を示します。

```
[nss]
filter_groups = root
filter_users = root
entry_cache_timeout = 300
entry_cache_nowait_percentage = 75
```

利用可能なオプションの完全な一覧については、**sss.conf(5)** の man ページの **NSS configuration options** を参照してください。

4. SSSD を再起動します。

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

統合が正常に機能するかをテスト

以下のコマンドを使用して、ユーザー情報を表示します。

- **id user**
- **getent passwd user**

7.5.2. サービスの設定: PAM



警告

PAM 設定ファイルに誤りがあると、ユーザーはシステムから完全にロックアウトされてしまいます。設定ファイルは変更を行う前に必ずバックアップを作成し、変更を元に戻すことができるようにセッションをオープンのままにしてください。

PAM を設定して SSSD を使用

- **authconfig** ユーティリティーを使用して SSSD を有効にします。

```
# authconfig --enablesssdauth --update
```

これにより、通常は **/etc/pam.d/system-auth** ファイルおよび **/etc/pam.d/password-auth** ファイルで PAM 設定がアップデートされ、SSSD モジュールを参照します。以下に例を示します。

```
[... file truncated ...]
auth    required pam_env.so
auth    sufficient pam_unix.so nullok try_first_pass
auth    requisite pam_succeed_if.so uid >= 500 quiet
auth    sufficient pam_sss.so use_first_pass
auth    required pam_deny.so
[... file truncated ...]
```

詳細は、**pam.conf(5)** または **pam(8)** の各 man ページを参照してください。

PAM と機能するように SSSD を設定

1. **/etc/sss/sss.conf** ファイルを開きます。
2. **[sss]** セクションでは、NSS が SSSD と共に機能するサービスの1つとして記載されていることを確認します。

```
[sss]
[... file truncated ...]
services = nss, pam
```

3. **[pam]** セクションでは、SSSD が PAM と対話する方法を設定します。以下に例を示します。

```
[pam]
offline_credentials_expiration = 2
offline_failed_login_attempts = 3
offline_failed_login_delay = 5
```

利用可能なオプションの完全な一覧については、**sss.conf(5)** の man ページの **PAM configuration options** を参照してください。

4. SSSD を再起動します。

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

統合が正常に機能するかをテスト

- ユーザーとしてログインを試行します。
- **sssctl user-checks user_name auth** コマンドを使用して、SSSD 設定を確認します。詳細については、**sssctl user-checks --help** コマンドを使用します。

7.5.3. サービスの設定:autofs

SSSD が automount と機能する方法

automount ユーティリティーでは、NFS ファイルシステムの自動マウントおよび自動アンマウントが可能なため (オンデマンドによるマウント機能)、システムのリソースを節約することができます。**automount** の詳細は、ストレージ管理ガイドの **autofs** を参照してください。

automount を SSSD に向けるよう設定することができます。以下のようなセットアップで可能です。

1. ユーザーがディレクトリーのマウントを試行する場合、SSSD は LDAP に連絡し、現行の **automount** 設定に関する必要な情報を取得します。
2. SSSD は、**automount** が必要とする情報をキャッシュに保存するので、LDAP サーバーがオフラインになっても、ユーザーはディレクトリーをマウントすることができます。

autofs を設定して SSSD を使用

1. **autofs** パッケージをインストールします。

```
# yum install autofs
```

2. **/etc/nsswitch.conf** ファイルを開きます。
3. **automount** ライン上で、**automount** のマップ情報を探すための場所を **ldap** から **sss** へと変更します。

```
automount: files sss
```

autofs と機能するように SSSD を設定

1. **/etc/sss/sss.conf** ファイルを開きます。
2. **[sss]** セクションで、SSSD が管理するサービスの一覧に **autofs** を追加します。

```
[sss]
services = nss,pam,autofs
```

3. 新しい **[autofs]** セクションを作成します。空のままでかまいません。

```
[autofs]
```

利用可能なオプションの一覧については、**sss.conf(5)** の man ページの **AUTOFS configuration options** を参照してください。

4. SSSD が LDAP から **automount** 情報を読み取れるように、LDAP ドメインが **sssd.conf** で利用可能であることを確認してください。詳細は、「[SSSD 向け LDAP ドメインの設定](#)」を参照してください。

sssd.conf の **[domain]** セクションは、複数の **autofs** 関連のオプションを受け入れます。以下に例を示します。

```
[domain/LDAP]
[...] file truncated ...]
autofs_provider=ldap
ldap_autofs_search_base=cn=automount,dc=example,dc=com
ldap_autofs_map_object_class=automountMap
ldap_autofs_entry_object_class=automount
ldap_autofs_map_name=automountMapName
ldap_autofs_entry_key=automountKey
ldap_autofs_entry_value=automountInformation
```

利用可能なオプションの完全な一覧については、**sssd.conf(5)** の man ページの **DOMAIN SECTIONS** を参照してください。

追加の **autofs** オプションを提供しない場合は、設定はアイデンティティプロバイダーの設定に依存します。

5. SSSD を再起動します。

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

設定のテスト

- **automount -m** コマンドを使用して、SSSD からマップをプリントします。

7.5.4. サービスの設定:sudo

SSSD が sudo と機能する方法

sudo ユーティリティーでは、指定されたユーザーに管理アクセスが与えられます。**sudo** の詳細情報については、『システム管理者のガイド』の [The sudo コマンド](#) を参照してください。

sudo を SSSD に向けるよう設定することができます。以下のようなセットアップで可能です。

1. ユーザーが **sudo** オペレーションを試行した場合、SSSD は LDAP に連絡し、現行の **sudo** 設定に関する必要な情報を取得します。
2. SSSD は **sudo** 情報をキャッシュに保存するので、LDAP サーバーがオフラインになっても、ユーザーは **sudo** 操作を実行できます。

SSSD は、**sudoHost** 属性の値によって、ローカルシステムに適用される **sudo** ルールのみをキャッシュします。詳細については、**sssd-sudo(5)** の man ページを参照してください。

sudo を設定して SSSD を使用

1. **/etc/nsswitch.conf** ファイルを開きます。
2. SSSD を **sudoers** ライン上の一覧に追加します。

```
sudoers: files sss
```

sudo と機能するように SSSD を設定

1. `/etc/sss/sss.conf` ファイルを開きます。
2. `[sss]` セクションで、SSSD が管理するサービスの一覧に **sudo** を追加します。

```
[sss]
services = nss,pam,sudo
```

3. 新しい `[sudo]` セクションを作成します。空のままでかまいません。

```
[sudo]
```

利用可能なオプションの一覧については、`sss.conf(5)` の man ページの **SUDO configuration options** を参照してください。

4. SSSD が LDAP から **sudo** 情報を読み取れるように、LDAP ドメインが `sss.conf` で利用可能であることを確認してください。詳細は「[SSSD 向け LDAP ドメインの設定](#)」を参照してください。

LDAP ドメインの `[domain]` セクションは、これらの **sudo** 関連のパラメーターを格納する必要があります。

```
[domain/LDAP]
[... file truncated ...]
sudo_provider = ldap
ldap_sudo_search_base = ou=sudoers,dc=example,dc=com
```



注記

Identity Management を ID プロバイダーとして設定すると、**sudo** プロバイダーは自動的に有効となります。この場合、**sudo_provider = ipa** を指定する必要はありません。

利用可能なオプションの完全な一覧については、`sss.conf(5)` の man ページの **DOMAIN SECTIONS** を参照してください。

sudo プロバイダーの利用可能なオプションについては、`sss-ldap(5)` の man ページを参照してください。

5. SSSD を再起動します。

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

7.6. SSSD クライアント側のビュー

SSSD は、クライアント側のビューを作成して、POSIX ユーザーまたはグループ属性に新しい値を指定することができます。ビューは、上書きが設定されたローカルマシンでのみ有効になります。**ipa** 以外のすべての **id_provider** の値にクライアント側の上書きが設定されます。**ipa** プロバイダーを使用す

る場合には、IdM で集約的に ID ビューを定義します。『Linux ドメイン ID、認証およびポリシーガイド』の該当するセクションを参照してください。

SSSD パフォーマンスに対する潜在的なマイナス要因については『Linux ドメイン ID、認証およびポリシーガイド』の該当するセクションを参照してください。



注記

sss_override user-add、**sss_override group-add** または **sss_override user-import** コマンドを使用して最初の上書きを作成した後に、SSSD を再起動して変更を適用します。

```
# systemctl restart sssd
```

7.6.1. ユーザーアカウントに対する異なる属性値の定義

管理者として、既存のホストが LDAP からのアカウントを使用するように設定しましたが、LDAP のユーザーの新規 ID がローカルシステムの以前の ID と異なる場合に、既存ファイルのパーミッションを変更するのではなく、クライアント側のビューが UID を上書きするように設定できます。

user アカウントの UID を UID **6666** に上書きするには以下を行います。

1. オプション **user** アカウントの現在の UID を表示します。

```
# id user
uid=1241400014(user_name) gid=1241400014(user_name)
Groups=1241400014(user_name)
```

2. アカウントの UID を **6666** に上書きします。

```
# sss_override user-add user -u 6666
```

3. インメモリーキャッシュが失効するまで待ちます。手動で失効させるには以下を行います。

```
# sss_cache --users
```

4. 新しい UID が適用されていることを確認します。

```
# id user
uid=6666(user_name) gid=1241400014(user_name)
Groups=1241400014(user_name)
```

5. オプション **ユーザー** の上書きを表示します。

```
# sss_override user-show user
user@ldap.example.com::6666:::~:
```

上書き可能な属性の一覧については **--help** をコマンドに追加してコマンドラインオプションを表示します。

```
# sss_override user-add --help
```


7.6.2. ホスト上の全上書きの表示

管理者として、ホスト上のすべてのユーザーおよびグループの上書きを表示して、正しい属性により上書きされたことを確認します。

ユーザーの上書きをすべて表示するには以下を実行します。

```
# sss_override user-find
user1@ldap.example.com::8000::::/bin/zsh:
user2@ldap.example.com::8001::::/bin/bash:
...
```

グループの上書きをすべて表示するには以下を実行します。

```
# sss_override group-find
group1@ldap.example.com::7000
group2@ldap.example.com::7001
...
```

7.6.3. ローカルの上書きの削除

グローバル LDAP ディレクトリーに定義されている **user** アカountのシェルの上書きを過去に作成しており、このアカウントの上書きを削除するには以下を実行します。

```
# sss_override user-del user
```

変更はすぐに適用されます。

グループの上書きを削除するには以下を実行します。

```
# sss_override group-del group
```



注記

ユーザーまたはグループの上書きを削除する場合に、このオブジェクトの上書きをすべて削除します。

7.6.4. ローカルビューのエクスポートとインポート

クライアント側のビューは、ローカルの SSSD キャッシュに保存されています。キャッシュからバックアップを作成するファイルにユーザーおよびグループビューをエクスポートすることができます。たとえば、SSSD キャッシュを削除すると、後ほどこのビューを復元することができます。

ユーザーおよびグループビューをバックアップするには以下を実行します。

```
# sss_override user-export /var/lib/sss/backup/sss_user_overrides.bak
# sss_override group-export /var/lib/sss/backup/sss_group_overrides.bak
```

ユーザーおよびグループビューを復元するには以下を実行します。

```
# sss_override user-import /var/lib/sss/backup/sss_user_overrides.bak
# sss_override group-import /var/lib/sss/backup/sss_group_overrides.bak
```

7.7. SSSD のダウングレード

SSSD のバージョンもしくはオペレーティングシステム自体をダウングレードする際には、既存の SSSD キャッシュを削除する必要があります。これが削除されないと、SSSD プロセスは停止しますが PID ファイルは残ります。以下の SSSD ログは、キャッシュのバージョンを認識できないので関連ドメインに接続できないことを示しています。

```
(Wed Nov 28 21:25:50 2012) [sssd] [sysdb_domain_init_internal] (0x0010):  
Unknown DB version [0.14], expected [0.10] for domain AD!
```

するとユーザーは認識されず、ドメインサーバーとホストに認証されなくなります。

SSSD バージョンをダウングレードしたら、以下を実行します。

1. 既存のキャッシュデータベースファイルを削除します。

```
[root@server ~]# rm -rf /var/lib/sss/db/*
```

2. SSSD プロセスを再起動します。

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

7.8. SSSD と NSCD の使用

SSSD は、NSCD デーモンと併用するようには設計されていません。SSSD は直接的に NSCD と競合することはありませんが、両方のサービスを併用すると、特にエントリーがキャッシュされる期間に関して予期しない動作が発生する可能性があります。

最も一般的な問題は NFS との競合です。ネットワーク接続の管理に **Network Manager** を使用していると、ネットワークインターフェースが立ち上がるのに数分かかる場合があります。この時間内に各種のサービスが起動を試みます。ネットワークが起動して DNS サーバーが利用可能になる前にこれらのサービスが起動した場合、これらのサービスは必要となる順引きまたは逆引きの DNS エントリーの特定に失敗します。これらのサービスは、不正確かおそらく空の **resolv.conf** ファイルを読み取ることになります。このファイルは通常 1 回しか読み取られないため、このファイルへの変更は自動的に適用されません。NSCD サービスが手動で再起動されないと、この状態は NSCD サービスが実行しているマシン上で NFS 失敗の原因になります。

この問題を避けるには、**/etc/nscd.conf** ファイルでホストとサービスのキャッシングを有効にし、**passwd**、**group**、**services** および **netgroup** のエントリーで SSSD キャッシュに依存するようにします。

/etc/nscd.conf ファイルを変更します。

```
enable-cache hosts yes  
enable-cache passwd no  
enable-cache group no  
enable-cache netgroup no  
enable-cache services no
```

NSCD がホスト要求に応答するようにすると、これらのエントリーは NSCD がキャッシュし、ブートプロセス中に NSCD によって返されます。他のエントリーはすべて SSSD が処理します。

7.9. その他のリソース

- SSSD 関連の man ページの完全な一覧は、`sssd(8)` の man ページの **SEE ALSO** を参照してください。
- トラブルシューティングのアドバイス: 「[SSSD のトラブルシューティング](#)」
- サーバーから送信されるパスワード有効期限警告を処理し、ローカルシステムでユーザーに警告を表示する手順を SSSD で設定するには、Red Hat のナレッジベースにある [Setting Password Expiry](#) を参照してください。
- SSSD クライアントは、LDAP サーバーから取得した各ユーザーに対して自動的に GID を作成できます。また同時に、GID 番号がすでに取りられていない限り、GID とユーザーの UID が一致するよう確認します。Active Directory に直接統合される SSSD クライアントで GID の自動作成が可能となる方法については、[Windows 統合ガイド](#) の該当するセクションを参照してください。

[1] DNS service discovery は、アプリケーションが特定のタイプの特定のサービスについて、あるドメイン内の SRV 記録をチェックできるようにします。そして、必要とされるタイプと一致するサーバーを返します。DNS service discovery は、[RFC 2782](#) で定義されています。

第8章 **REALMD** を使った **ID** ドメインへの接続

realmd システムは、ID ドメインを発見して参加する明確かつ簡単な方法を提供します。これはドメイン自体には接続しませんが、**SSSD** や **Winbind** といった基礎となる Linux システムサービスがドメインに接続するよう設定します。

Windows 統合ガイドでは、**realmd** を使用して **Microsoft Active Directory (AD)** ドメインに接続する方法を説明しています。**realmd** を使用して AD 以外の ID ドメインに接続する方法には、同様の手順が適用されます。対応する『[Windows 統合ガイド](#)』の章を参照してください。

第9章 LDAP サーバー

LDAP (Lightweight Directory Access Protocol) は、ネットワーク上で中央に格納されている情報へアクセスするために使用するオープンプロトコルのセットです。これは、ディレクトリー共有のために **X.500** 標準をベースとしていますが、それほど複雑ではなくリソース集約型です。このため LDAP は「**X.500 Lite**」と呼ばれることがあります。

X.500 のように、LDAP はディレクトリーを使用して階層式で情報を編成します。ディレクトリーには、名前やアドレス、電話番号などの各種情報を格納することができ、また ネットワーク情報サービス (NIS) と似た方法で 사용할 こともできます。このため、誰でも LDAP 対応ネットワーク上にあるマシンから自分のアカウントにアクセスできます。

LDAP の一般的な使用目的は、中央で管理されるユーザーとグループ、ユーザー認証、システム設定です。また、仮想の電話帳としても使用できるため、ユーザーは他のユーザーの連絡先情報に容易にアクセスできます。さらに、LDAP は世界各地にある他の LDAP サーバーにユーザーを照会して、グローバルレポジトリに保存されているアドホックな情報を提供することが可能です。しかし LDAP が最もよく使用されているのは、大学や政府省庁、民間企業など個々の組織内です。

9.1. RED HAT DIRECTORY SERVER

Red Hat Directory Server は、ユーザーアイデンティティーおよびアプリケーション情報を一元化する LDAP に準拠したサーバーです。このサーバーは、アプリケーションの設定、ユーザープロファイル、グループデータ、ポリシー、およびアクセス制御情報の保存用にオペレーティングシステムから独立したネットワークベースのレジストリーを提供します。



注記

Directory Server のインストールおよびアップデートには、最新の Red Hat Directory Server サブスクリプションが必要です。

Directory Server のセットアップおよび使用に関する詳細は、以下を参照してください。

- 『[Red Hat Directory Server Installation Guide](#)』
- 『[Red Hat Directory Server Deployment Guide](#)』
- 『[Red Hat Directory Server Administration Guide](#)』
- 『[Red Hat Directory Server Configuration, Command, and File Reference](#)』
- 『[Red Hat Directory Server Performance Tuning Guide](#)』
- 『[Red Hat Directory Server Plug-in Guide](#)』

9.2. OPENLDAP

本セクションでは、LDAPv2 および LDAPv3 プロトコルのオープンソース実装である **OpenLDAP 2.4** をインストール/設定する方法について取り上げます。



注記

openldap-server パッケージは、Red Hat Enterprise Linux 7.4 以降は非推奨とされ、Red Hat Enterprise Linux の今後のメジャーリリースには同梱されません。この理由から、Red Hat Enterprise Linux に同梱された Identity Management または Red Hat Directory Server へ移行してください。Identity Management の詳細情報は『[Linux ドメイン ID、認証、およびポリシーガイド](#)』を参照してください。Directory Server の詳細情報は「[Red Hat Directory Server](#)」を参照してください。

9.2.1. LDAP の概要

LDAP は、クライアント/サーバーのアーキテクチャーを使用して、ネットワークからアクセス可能な中央情報ディレクトリを作成する、信頼できる手段を提供します。クライアントがこのディレクトリ内の情報を修正しようとする、サーバーはこのユーザーが修正できる権限を持っていることを確認します。その後、要求されたとおりにエントリを追加、更新します。通信の安全性を確保するために、*Transport Layer Security* (TLS) 暗号化プロトコルを使用して攻撃者が送信を傍受できないようにします。



重要

Red Hat Enterprise Linux 7.5 以降の OpenLDAP スイートは、ネットワークセキュリティーサービス (NSS) の Mozilla 実装を使用しなくなりました。代わりに、*OpenSSL* を使用します。OpenLDAP は、現行の NSS データベース設定による機能を継続します。



重要

[Resolution for POODLE SSLv3.0 vulnerability \(CVE-2014-3566\) for components that do not allow SSLv3 to be disabled via configuration settings](#) (設定から SSLv3 を無効にできないコンポーネントで POODLE SSLv3.0 脆弱性 (CVE-2014-3566) を解決する方法) に説明されている脆弱性により、Red Hat はセキュリティー保護のために **SSLv3** プロトコルに依存しないことを推奨しています。OpenLDAP は、**SSLv3** を効果的に無効にする設定パラメーターを提供しないシステムコンポーネントの1つです。リスクの軽減策として、**stunnel** コマンドを使用してセキュアなトンネルを提供し、**stunnel** で **SSLv3** を使用できないようにすることが推奨されます。**stunnel** の使用方法についての詳細は、[Red Hat Enterprise Linux 7 セキュリティーガイド](#) を参照してください。

LDAP サーバーは複数のデータベースシステムをサポートするため、管理者は適用する予定の情報のタイプに最適なソリューションを柔軟に選択できます。明確に定義された *アプリケーションプログラミングインターフェース* (API) クライアントにより、LDAP サーバーと通信できるアプリケーションは多く、その数と質ともに向上しています。

9.2.1.1. LDAP の用語

以下は、本章で使用されている LDAP 特有の用語一覧です。

エントリ

LDAP ディレクトリ内の単一ユニットです。各エントリは一意の DN (*Distinguished Name*: 識別名) で識別されます。

属性

エントリに直接関連付けられた情報です。たとえば、ある組織が LDAP エントリとして表示されている場合に、アドレス、ファックス番号などがこの組織に関連付けられた属性です。同様に、人も電話番号や電子メールアドレスのような共通の属性を持つエントリとして表示することがで

きます。

属性には、単一値または順不同の空白で区切られた値の一覧のどちらかがあります。一部の属性はオプションですが、その他は必須です。必須の属性は **objectClass** 定義を使用して指定され、**/etc/openldap/slapd.d/cn=config/cn=schema/** ディレクトリー内のスキーマファイル内にあります。

属性とそれに対応する値のアサーションは、**RDN (Relative Distinguished Name: 相対識別名)** と呼ばれます。グローバルで一意の識別名とは異なり、相対識別名は エントリーに対してのみ一意なものです。

LDIF

LDAP データ交換形式(LDIF)は LDAP エントリーをプレーンテキスト形式で表示したものです。以下のような形式を取ります。

```
[id] dn: distinguished_name
attribute_type: attribute_value...
attribute_type: attribute_value...
...
```

オプションの *id* は、エントリーの編集に使用されるアプリケーションにより決定される番号です。*attribute_type* と *attribute_value* のペアが対応するスキーマファイルですべて定義されている限り、各エントリーはそれらを必要な数だけ含むことができます。空白の行はエントリーの終了を意味しています。

9.2.1.2. OpenLDAP の機能

OpenLDAP スイートは多くの重要な機能を提供します。

- **LDAPv3 サポート** – LDAP バージョン 2 以降に行われたプロトコルへの変更の多くは、LDAP をよりセキュアにするように設計されています。その他の改善点としては、**Simple Authentication and Security Layer (SASL)**、**Transport Layer Security (TLS)** および **Secure Sockets Layer (SSL)** プロトコルに対するサポートがあります。
- **IPC 上の LDAP** – プロセス間通信 (IPC) の使用により、ネットワーク上で通信する必要性をなくすことでセキュリティーを強化します。
- **IPv6 サポート** – OpenLDAP は、次世代のインターネットプロトコルである **Internet Protocol version 6 (IPv6)** に準拠しています。
- **LDIFv1 サポート** – OpenLDAP は LDIF バージョン 1 に完全に準拠しています。
- **更新された C API** – 最新の C API はプログラマーが LDAP ディレクトリーサーバーに接続して使用する方法を改善します。
- **スタンドアロン LDAP サーバーの機能強化** – これには更新されたアクセス制御システム、スレッドプーリング、改善されたツール、その他多くが含まれます。

9.2.1.3. OpenLDAP サーバーの設定

Red Hat Enterprise Linux における LDAP サーバーの標準的な設定手順は、以下のとおりです。

1. OpenLDAP スイートをインストールします。必要なパッケージのインストールに関する詳しい情報は、「[OpenLDAP スイートのインストール](#)」を参照してください。

2. 「[OpenLDAP サーバーの設定](#)」で説明のとおり設定をカスタマイズします。
3. 「[OpenLDAP サーバーの実行](#)」で説明のとおり **slapd** サービスを開始します。
4. **ldapadd** ユーティリティを使用して、エントリーを LDAP ディレクトリーに追加します。
5. **ldapsearch** ユーティリティを使用して、**slapd** サービスが情報に正しくアクセスしていることを確認します。

9.2.2. OpenLDAP スイートのインストール

OpenLDAP ライブラリーとツールのスイートは、以下のパッケージにより提供されます。

表9.1 OpenLDAP パッケージの一覧

パッケージ	詳細
openldap	OpenLDAP サーバー/クライアントアプリケーションを実行するために必要なライブラリーを含むパッケージです。
openldap-clients	LDAP サーバー上のディレクトリーを表示、修正するためのコマンドラインユーティリティを含むパッケージです。
openldap-servers	LDAP サーバーを設定して実行するためのサービスとユーティリティの両方を含むパッケージです。これには、スタンドアロン LDAP デーモンである slapd が含まれます。
compat-openldap	OpenLDAP 互換ライブラリーを含むパッケージです。

また、LDAP サーバーと共によく使用されるパッケージは以下のとおりです。

表9.2 一般的にインストールされるその他の LDAP パッケージの一覧

パッケージ	詳細
nss-pam-ldapd	ユーザーがローカルの LDAP クエリーを実行できるようにするローカル LDAP ネームサービスである、 nsldap を含むパッケージです。
mod_ldap	mod_authnz_ldap および mod_ldap モジュールを含むパッケージです。 mod_authnz_ldap モジュールは Apache HTTP Server の LDAP 承認モジュールです。このモジュールは、LDAP ディレクトリーに対してユーザーの認証情報を認証でき、ユーザー名、完全 DN、グループメンバーシップ、任意の属性、または完全なフィルター文字列に基づいてアクセス制御を行います。同じパッケージに含まれる mod_ldap モジュールは、数多くの HTTP リクエストでのディレクトリーアクセスの繰り返しを防ぐために設定可能な共有メモリーキャッシュ、および SSL/TLS のサポートを提供します。このパッケージは Optional チャンネルで提供されることに注意してください。Red Hat の追加チャンネルの詳細は、『システム管理者のガイド』の「 Optional および Supplementary リポジトリの追加 」を参照してください。

これらのパッケージをインストールするには、以下の形式で **yum** コマンドを使用します。


```
yum install package...
```

たとえば、基本的な LDAP サーバーをインストールするには、シェルスクリプトで以下を入力します。

```
~]# yum install openldap openldap-clients openldap-servers
```

このコマンドを実行するには、スーパーユーザー権限を持っている (**root** としてログインしている) 必要があります。Red Hat Enterprise Linux に新しいパッケージをインストールする方法の詳細については、『システム管理者のガイド』の「[パッケージのインストール](#)」を参照してください。

9.2.2.1. OpenLDAP サーバーユーティリティーの概要

管理タスクを実行するために、**openldap-servers** パッケージは **slapd** サービスと共に以下のユーティリティーをインストールします。

表9.3 OpenLDAP サーバーユーティリティーの一覧

コマンド	詳細
slapacl	属性の一覧へのアクセスをチェックできるようにします。
slapadd	LDIF ファイルから LDAP ディレクトリーへエントリーを追加できるようにします。
slapauth	認証/承認のパーミッション用に ID の一覧をチェックできるようにします。
slapcat	LDAP ディレクトリーからデフォルト形式でエントリーをプルして、LDIF ファイル内に保存できるようにします。
slapdn	利用可能なスキーマ構文を基に、識別名 (DN) の一覧をチェックできるようにします。
slapindex	現在の内容を基に、 slapd ディレクトリーのインデックスを再構築できるようにします。設定ファイル内のインデックスオプションを変更する時は常にこのユーティリティーを実行します。
slappasswd	暗号化されたユーザーパスワードを作成できるようにします。このパスワードは、 ldapmodify ユーティリティーと共に、または slapd 設定ファイル内で使用できます。
slapschema	対応するスキーマとのデータベースの整合性をチェックできるようにします。
slaptest	LDAP サーバーの設定をチェックできるようにします。

これらのユーティリティーとその使用方法の詳しい説明については、「[インストールされているドキュメント](#)」にある各 man ページを参照してください。



重要

slapadd が実行可能なのは **root** のみですが、**slapd** サービスは **ldap** ユーザーとして実行します。このため、ディレクトリーサーバーは **slapadd** によって作成されたファイルは修正することができません。この問題を解決するには、**slapdadd** ユーティリティーの実行後に、シェルプロンプトで以下を入力します。

```
~]# chown -R ldap:ldap /var/lib/ldap
```



警告

データの整合性を維持するために、**slapadd**、**slapcat**、**slapindex** を使用する前に **slapd** サービスを停止します。シェルプロンプトで以下を入力します：

```
~]# systemctl stop slapd.service
```

slapd サービスの起動、停止、再起動および現在のステータスの確認を行う方法の詳細については、「[OpenLDAP サーバーの実行](#)」を参照してください。

9.2.2.2. OpenLDAP クライアントユーティリティーの概要

openldap-clients パッケージは、LDAP ディレクトリー内のエントリーを追加/修正/削除するために使用できる以下のユーティリティーをインストールします。

表9.4 OpenLDAP クライアントユーティリティーの一覧

コマンド	詳細
ldapadd	ファイルまたは標準入力からエントリーを LDAP ディレクトリーに追加できるようにします。これは ldapmodify -a へのシンボリックリンクです。
ldapcompare	任意の属性と LDAP ディレクトリーのエントリーを比較できるようにします。
ldapdelete	LDAP ディレクトリーからエントリーを削除できるようにします。
ldapexop	LDAP の拡張操作を実行できるようにします。
ldapmodify	LDAP ディレクトリー内のエントリーをファイルまたは標準入力から修正できるようにします。
ldapmodrdn	LDAP ディレクトリーエントリーの RDN 値を修正できるようにします。
ldappasswd	LDAP ユーザー用のパスワードを設定/変更できるようにします。

コマンド	詳細
ldapsearch	LDAP ディレクトリーエントリーを検索できるようにします。
ldapurl	LDAP URL を生成/分解できるようにします。
ldapwhoami	LDAP サーバー上で whoami 操作を実行できるようにします。

ldapsearch は例外ですが、これらのユーティリティーをより簡単に利用するには、LDAP ディレクトリー内で変更を加えるエントリーごとにコマンドを入力するのではなく、変更が含まれるファイルを参照します。このようなファイルの形式は、各ユーティリティーの **man** ページに要約されています。

9.2.2.3. 一般的な LDAP クライアントアプリケーションの概要

サーバー上のディレクトリーを作成/修正できるグラフィカルな LDAP クライアントは各種存在しますが、どれも Red Hat Enterprise Linux には含まれていません。読み取り専用モードでディレクトリーにアクセスできる人気のアプリケーションとしては、**Mozilla Thunderbird**、**Evolution**、**Ekiga** があります。

9.2.3. OpenLDAP サーバーの設定

デフォルトでは、OpenLDAP 設定は **/etc/openldap/** ディレクトリーに格納されています。以下の表は、このディレクトリー内の最も重要なディレクトリーとファイルを示しています。

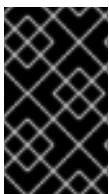
表9.5 OpenLDAP 設定ファイルとディレクトリーの一覧

パス	詳細
/etc/openldap/ldap.conf	OpenLDAP ライブラリーを使用するクライアントアプリケーション用の設定ファイルです。これには ldapadd 、 ldapsearch 、 Evolution などが含まれます。
/etc/openldap/slapd.d/	slapd 設定を含むディレクトリーです。

OpenLDAP は設定の **/etc/openldap/slapd.conf** ファイルからの読み取りを実行しなくなった点に注意してください。代わりに、**/etc/openldap/slapd.d/** ディレクトリーに存在する設定データベースを使用します。以前のインストールから **slapd.conf** ファイルが残っている場合は、以下のコマンドを実行することで新しい形式に変換できます。

```
~]# slaptest -f /etc/openldap/slapd.conf -F /etc/openldap/slapd.d/
```

slapd 設定は、階層ディレクトリー構造で編成されている LDIF エントリーで構成されています。これらのエントリーの編集方法として推奨されるのは、「[OpenLDAP サーバーユーティリティーの概要](#)」で説明されているサーバーユーティリティーを使用することです。



重要

LDIF ファイル内でエラーが発生すると **slapd** サービスが起動できなくなることがあります。このため、**/etc/openldap/slapd.d/** 内の LDIF ファイルを直接編集しないことを強くお勧めします。

9.2.3.1. グローバル設定の変更

LDAP サーバーのグローバル設定オプションは、`/etc/openldap/slapd.d/cn=config.ldif` ファイル内に格納されています。一般的に使用されるディレクティブは、以下のとおりです。

olcAllows

olcAllows ディレクティブを使用すると、有効にする機能を指定できます。以下の形式を取ります。

```
olcAllows: feature...
```

これは、表9.6「利用可能な **olcAllows** のオプション」で説明のとおり、空白で区切られた機能の一覧を使用します。デフォルトオプションは **bind_v2** です。

表9.6 利用可能な **olcAllows** のオプション

オプション	詳細
bind_v2	LDAP バージョン 2 のバインド要求の受け入れを有効にします。
bind_anon_cred	識別名(DN) が空欄である場合の匿名バインドを有効にします。
bind_anon_dn	識別名(DN) が空欄でない場合の匿名バインドを有効にします。
update_anon	匿名による更新操作の処理を有効にします。
proxy_authz_anon	匿名によるプロキシ承認制御の処理を有効にします。

例9.1 **olcAllows** ディレクティブの使用

```
olcAllows: bind_v2 update_anon
```

olcConnMaxPending

olcConnMaxPending ディレクティブを使用すると、匿名セッションに対する保留中の要求の最大数を指定できます。以下の形式を取ります。

```
olcConnMaxPending: number
```

デフォルトオプションは **100** です。

例9.2 **olcConnMaxPending** ディレクティブの使用

```
olcConnMaxPending: 100
```

olcConnMaxPendingAuth

olcConnMaxPendingAuth ディレクティブを使用すると、認証済みセッションに対する保留中の要求の最大数を指定できます。以下の形式を取ります。

```
olcConnMaxPendingAuth: number
```

デフォルトオプションは **1000** です。

例9.3 olcConnMaxPendingAuth ディレクティブの使用

```
olcConnMaxPendingAuth: 1000
```

olcDisallows

olcDisallows ディレクティブを使用すると、無効にする機能を指定できます。以下の形式を取ります。

```
olcDisallows: feature...
```

これは、表9.7「利用可能な **olcDisallows** のオプション」で説明のとおり、空白で区切られた機能の一覧を使用します。デフォルトでは、どの機能も無効ではありません。

表9.7 利用可能な **olcDisallows** のオプション

オプション	詳細
bind_anon	匿名バインド要求の受け入れを無効にします。
bind_simple	簡易バインド認証のメカニズムを無効にします。
tls_2_anon	STARTTLS コマンドの受け取り時に、匿名セッションの強制を無効にします。
tls_authc	認証時に、STARTTLS コマンドを許可しません。

例9.4 olcDisallows ディレクティブの使用

```
olcDisallows: bind_anon
```

olcIdleTimeout

olcIdleTimeout ディレクティブを使用すると、アイドル状態の接続を終了するまでの待機秒数を指定できます。以下の形式を取ります。

```
olcIdleTimeout: number
```

このオプションは、デフォルトで無効です (**0** に設定されています)。

例9.5 olcIdleTimeout ディレクティブの使用

```
olcIdleTimeout: 180
```

olcLogFile

olcLogFile ディレクティブを使用すると、ログメッセージを書き込むファイルを指定できます。以下の形式を取ります。

```
olcLogFile: file_name
```

ログメッセージは、デフォルトで標準エラーに書き込まれます。

例9.6 olcLogFile ディレクティブの使用

```
olcLogFile: /var/log/slapd.log
```

olcReferral

olcReferral オプションを使用すると、サーバーが要求を処理できない場合に処理する別のサーバーの URL を指定できます。以下の形式を取ります。

```
olcReferral: URL
```

このオプションは、デフォルトで無効です。

例9.7 olcReferral ディレクティブの使用

```
olcReferral: ldap://root.openldap.org
```

olcWriteTimeout

olcWriteTimeout オプションを使用すると、未処理の書き込み要求がある接続を終了するまでの待機秒数を指定できます。以下の形式を取ります。

```
olcWriteTimeout
```

このオプションは、デフォルトで無効です (0 に設定されています)。

例9.8 olcWriteTimeout ディレクティブの使用

```
olcWriteTimeout: 180
```

9.2.3.2. フロントエンドの設定

OpenLDAP フロントエンド設定は **etc/openldap/slapd.d/cn=config/olcDatabase={-1}frontend.ldif** ファイルに保存され、アクセス制御リスト (ACL) など、グローバルのデータベースオプションを定義します。詳細は、**slapd-config(5)** man ページの **Global Database Options** のセク

ションを参照してください。

9.2.3.3. 監視のバックエンド

`/etc/openldap/slapd.d/cn=config/olcDatabase={1}monitor.ldif` ファイルは、OpenLDAP 監視バックエンドを制御します。有効な場合には、このファイルは OpenLDAP により自動的に生成され、デーモンの実行状況に関する情報で動的に更新されます。サフィックスは `cn=Monitor` で、変更できません。詳細情報は `slapd-monitor(5)` の man ページを参照してください。

9.2.3.4. データベース固有の設定

OpenLDAP サーバーはデフォルトで、**hdb** データベースのバックエンドを使用します。OpenLDAP サーバーは、サブソリィの名前変更をサポートする階層データベースのレイアウトを使用しているのに加え、**bdb** バックエンドと同一のため、同じ設定オプションを使用しています。このデータベースバックエンドの設定は `/etc/openldap/slapd.d/cn=config/olcDatabase={2}hdb.ldif` ファイルに保存されます。

他のバックエンドのデータベース一覧は `slapd.backends(5)` の man ページを参照してください。データベース固有の設定は、個別のバックエンドの man ページを確認してください。以下に例を示します。

```
# man slapd-hdb
```



注記

bdb および **hdb** バックエンドは非推奨です。代わりに、新規インストールには **mdb** バックエンドの使用を検討してください。

以下のディレクティブは、データベース固有の設定で一般的に使用します。

olcReadOnly

olcReadOnly ディレクティブを使用すると、データベースを読み取り専用モードで使用できます。以下の形式を取ります。

```
olcReadOnly: boolean
```

これは、**TRUE** (読み取り専用モードを有効) / **FALSE** (データベースの修正を有効) を使用します。デフォルトのオプションは **FALSE** です。

例9.9 **olcReadOnly** ディレクティブの使用

```
olcReadOnly: TRUE
```

olcRootDN

olcRootDN ディレクティブを使用すると、アクセス制御により制限されないユーザーまたは LDAP ディレクトリーの操作用に設定される管理制限パラメーターを指定できます。以下の形式を取ります。

```
olcRootDN: distinguished_name
```

これは、**識別名 (DN)** を使用します。デフォルトのオプションは **cn=Manager, dn=my-domain, dc=com** です。

例9.10 **olcRootDN** ディレクティブの使用

```
olcRootDN: cn=root, dn=example, dn=com
```

olcRootPW

olcRootPW ディレクティブを使用すると、**olcRootDN** ディレクティブを使って指定されたユーザー用のパスワードを設定できます。以下の形式を取ります。

```
olcRootPW: password
```

これは、プレーンテキスト文字列かハッシュを使用します。ハッシュを生成するためには、シェルプロンプトで以下を入力します。

```
~]$ slappaswd
New password:
Re-enter new password:
{SSHA}WczWsyPEnMchFf1GRTweq2q7XJcvmSxD
```

例9.11 **olcRootPW** ディレクティブの使用

```
olcRootPW: {SSHA}WczWsyPEnMchFf1GRTweq2q7XJcvmSxD
```

olcSuffix

olcSuffix ディレクティブを使用すると、情報を提供するドメインを指定できます。以下の形式を取ります。

```
olcSuffix: domain_name
```

これは、**完全修飾ドメイン名 (FQDN)** を使用します。デフォルトのオプションは **dc=my-domain, dc=com** です。

例9.12 **olcSuffix** ディレクティブの使用

```
olcSuffix: dc=example, dc=com
```

9.2.3.5. スキーマの拡張

OpenLDAP 2.3 以降、**/etc/openldap/slapd.d/** ディレクトリーには、以前 **/etc/openldap/schema/** に配置されていた LDAP 定義も含まれています。デフォルトのスキーマファイルをガイドとして使用して追加の属性タイプとオブジェクトクラスをサポートするために、

OpenLDAP により使用されるスキーマを拡張することができます。ただし、このタスクは本章の範囲外です。このトピックの詳細については、<http://www.openldap.org/doc/admin/schema.html> を参照してください。

9.2.3.6. セキュアな接続の確立

OpenLDAP スイートとサーバーは、Transport Layer Security (TLS) フレームワークを使用することでセキュリティ保護ができます。TLS は、ネットワーク上の通信セキュリティを提供するように設計された暗号化プロトコルです。Red Hat Enterprise Linux 7 の OpenLDAP スイートは、OpenSSL を TLS 実装として使用します。

TLS を使用して安全な接続を確立するには、必要な証明書を獲得します。そして、クライアントとサーバーの両方で多くのオプションを設定する必要があります。少なくとも、認証局 (CA) の証明書とそれ自体のサーバー証明書および秘密鍵で、サーバーを設定する必要があります。クライアントは、信頼できる CA 証明書すべてを含むファイル名で設定する必要があります。

サーバーは通常、1つの CA 証明書にのみ署名する必要があります。クライアントは種々の安全なサーバーに接続する可能性があるため、一般的には設定で信頼できるいくつかの CA 一覧を指定します。

サーバー設定

このセクションでは、TLS を確立するために OpenLDAP サーバー上の `/etc/openldap/slapd.d/cn=config.ldif` ファイルで指定する必要のある `slapd` のグローバル設定ディレクティブを表示します。

古いスタイルの設定では通常 `/usr/local/etc/openldap/slapd.conf` としてインストールされる単一ファイルを使用しますが、新規のスタイルでは `slapd` バックエンドデータベースを使用して設定を保存します。設定データベースは通常 `/usr/local/etc/openldap/slapd.d/` ディレクトリーにあります。

以下のディレクティブも SSL を確立するのに有効です。TLS ディレクティブの他にも、サーバー側で SSL 専用ポートを有効にする必要があります (通常はポート 636 になります)。これを実行するには、`/etc/sysconfig/slapd` ファイルを編集し、`ldaps:///` 文字列を `SLAPD_URLS` ディレクティブで指定された URL の一覧に追加します。

`olcTLSCACertificateFile`

`olcTLSCACertificateFile` ディレクティブは、信頼できる CA 証明書を含む PEM (Privacy-Enhanced Mail) スキーマでエンコードされたファイルを指定します。ディレクティブの形式は以下のようになります。

```
olcTLSCACertificateFile: path
```

path を CA 証明書ファイルへのパスに置き換えます。

`olcTLSCACertificatePath`

`olcTLSCACertificatePath` ディレクティブは、個別ファイルの CA 証明書を含むディレクトリーへのパスを指定します。このディレクトリーは、OpenSSL `c_rehash` ユーティリティーで特別に管理する必要があり、このユーティリティーは実際の証明書ファイルを指すハッシュ化された名前のシンボリックリンクを生成します。一般的に、`olcTLSCACertificateFile` ディレクティブを使用する方が、より簡易です。

ディレクティブの形式は以下のとおりです。

```
olcTLSCACertificatePath: path
```

path を CA 証明書ファイルを含むディレクトリーへのパスに置き換えます。指定されたディレクトリーは OpenSSL `c_rehash` ユーティリティーで管理する必要があります。

olcTLSCertificateFile

olcTLSCertificateFile ディレクティブは、**slapd** サーバー証明書を含むファイルを指定します。ディレクティブの形式は以下のとおりです。

olcTLSCertificateFile: *path*

path を **slapd** サービスのサーバー証明書ファイルへのパスに置き換えます。

olcTLSCertificateKeyFile

olcTLSCertificateKeyFile ディレクティブは、**olcTLSCertificateFile** で指定されたファイルに保存されている証明書に適合する秘密鍵を含むファイルを指定します。現在の実装では秘密鍵の暗号化はサポートされていないことから、この鍵を含むファイルは十分に保護される必要がありますことに注意してください。ディレクティブは以下の形式になります。

olcTLSCertificateKeyFile: *path*

path を秘密鍵ファイルへのパスに置き換えます。

クライアント設定

以下のディレクティブを、クライアントシステム上の `/etc/openldap/ldap.conf` 設定ファイルで指定します。これらのディレクティブのほとんどは、サーバー設定オプションと並行するものです。`/etc/openldap/ldap.conf` 内のディレクティブは、システム全体ベースで設定されていますが、個別のユーザーは `~/ldaprc` ファイルでこれを上書きすることができます。

SSL 接続を確立するには同じディレクティブを使用できます。**ldapsearch** などの OpenLDAP コマンドでは、**ldap://** ではなく **ldaps://** 文字列を使用する必要があります。これにより、コマンドはサーバーで設定される SSL のデフォルトポートであるポート **636** を使用するよう強制されます。

TLS_CACERT

TLS_CACERT ディレクティブは、クライアントが認識するすべての CA 証明書を格納しているファイルを指定します。これは、サーバー上の **olcTLSCACertificateFile** ディレクティブに相当します。**TLS_CACERT** は常に、`/etc/openldap/ldap.conf` 内の **TLS_CACERTDIR** の前に指定されなければなりません。ディレクティブの形式は以下のとおりです。

TLS_CACERT *path*

path を CA 証明書ファイルへのパスに置き換えます。

TLS_CACERTDIR

TLS_CACERTDIR ディレクティブは、個別ファイルにある CA 証明書を含むディレクトリーへのパスを指定します。サーバー上の **olcTLSCACertificatePath** と同様に、指定されたディレクトリーは OpenSSL `c_rehash` ユーティリティーで管理する必要があります。

TLS_CACERTDIR *directory*

directory を CA 証明書ファイルを含むディレクトリーへのパスに置き換えます。

TLS_CERT

TLS_CERT は、クライアントの証明書を格納しているファイルを指定します。この指示文の指定ができるのは、ユーザーの `~/ldaprc` ファイル内のみです。指示文の形式は以下のとおりです。

```
TLS_CERT path
```

`path` をクライアント証明書ファイルへのパスに置き換えます。

TLS_KEY

TLS_KEY は、**TLS_CERT** ディレクティブで指定されたファイルに保存されている証明書に適合する秘密鍵を格納するファイルを指定します。サーバー上の `olcTLSCertificateFile` と同様に鍵ファイルの暗号化はサポートされないため、ファイル自体は慎重に保護される必要があります。このオプションが設定できるのは、ユーザーの `~/ldaprc` ファイル内のみです。

TLS_KEY ディレクティブの形式は以下のようになります。

```
TLS_KEY path
```

`path` をクライアント証明書ファイルへのパスに置き換えます。

9.2.3.7. レプリケーションの設定

レプリケーションは、ある LDAP サーバー (プロバイダー) から別の1つ以上のサーバーまたはクライアント (コンシューマー) に更新をコピーするプロセスです。プロバイダーはディレクトリー更新をコンシューマーに複製し、この受け取られた更新はさらにコンシューマーから他のサーバーに伝達されます。このため、コンシューマーは同時にプロバイダーとしての動作も行います。また、コンシューマーは LDAP である必要はなく、単なる LDAP クライアントでも構いません。OpenLDAP では、いくつかの複製モードが使用可能で、なかでも重要なのは **mirror** と **sync** です。OpenLDAP 複製モードに関する詳細情報は、`openldap-servers` パッケージとインストールされる **OpenLDAP Software Administrator's Guide** を参照してください ([「インストールされているドキュメント」](#) を参照)。

選択した複製モードを有効にするには、プロバイダーとコンシューマーの両方の `/etc/openldap/slapd.d/` で、以下のディレクティブのいずれかを使用します。

olcMirrorMode

olcMirrorMode ディレクティブは、**mirror** 複製モードを有効にします。以下の形式になります。

```
olcMirrorMode on
```

このオプションは、プロバイダーとコンシューマーの両方で指定する必要があります。また、**serverID** を **syncrepl** オプションで指定する必要もあります。**OpenLDAP Software Administrator's Guide** の **18.3.4. MirrorMode** セクションにある詳細例を参照してください ([「インストールされているドキュメント」](#) を参照)。

olcSyncrepl

olcSyncrepl ディレクティブは、**sync** 複製モードを有効にします。以下の形式になります。

```
olcSyncrepl on
```

sync 複製モードは、プロバイダーとコンシューマーの両方で特定の設定が必要になります。この設定は、**OpenLDAP Software Administrator's Guide** の **18.3.1. Syncrepl** セクションで詳しく説明さ

れています(「[インストールされているドキュメント](#)」を参照)。

9.2.3.8. モジュールとバックエンドの読み込み

slapd サービスは、動的に読み込まれるモジュールで強化することができます。これらのモジュールのサポートは、**slapd** の設定時に **--enable-modules** オプションで有効にする必要があります。モジュールは、ファイルに **.la** 拡張子を付けて保存します。

```
module_name.la
```

バックエンドは、LDAP リクエストに応じて、データを保存または取得します。バックエンドは **slapd** に静的にコンパイルするか、モジュールサポートが有効な場合は動的に読み込むことができます。後者の場合には、以下の命名規則が適用されます。

```
back_backend_name.la
```

モジュールまたはバックエンドを読み込むには、以下のディレクティブを **/etc/openldap/slapd.d/** で使用します。

olcModuleLoad

olcModuleLoad ディレクティブは、動的に読み込まれるモジュールを指定します。以下の形式になります。

```
olcModuleLoad: module
```

ここでの *module* は、読み込まれる予定のモジュールを格納しているファイルまたはバックエンドを表します。

9.2.4. LDAP を使用したアプリケーションの SELinux ポリシー

SELinux は Linux カーネルにおける強制アクセス制御メカニズムの実装です。デフォルトで SELinux はアプリケーションが OpenLDAP サーバーにアクセスすることを防ぎます。いくつかのアプリケーションで必要となる LDAP での認証を可能にするには、**allow_ybind** SELinux ブール値を有効にする必要があります。このシナリオの一部のアプリケーションでは、有効にされた **authlogin_nsswitch_use_ldap** ブール値も必要になります。以下のコマンドを実行してこのブール値を有効にしてください。

```
~]# setsebool -P allow_ybind=1
```

```
~]# setsebool -P authlogin_nsswitch_use_ldap=1
```

-P オプションは、システムの再起動時にこの設定を永続化します。SELinux についての詳細情報は、『[Red Hat Enterprise Linux 7 SELinux ユーザーおよび管理者のガイド](#)』を参照してください。

9.2.5. OpenLDAP サーバーの実行

本セクションでは、**Standalone LDAP Daemon** の起動、停止、再起動、現行ステータスの確認方法を説明しています。システムサービス全般の詳細情報については、『[システム管理者のガイド](#)』の「[systemd でのサービスの管理](#)」を参照してください。

9.2.5.1. サービスの開始

現行のセッションで **slapd** サービスを起動するには、シェルプロンプトで **root** として以下を入力します。

```
~]# systemctl start slapd.service
```

サービスが起動時に自動的に開始するように設定するには、**root** で以下のコマンドを実行します。

```
~]# systemctl enable slapd.service
ln -s '/usr/lib/systemd/system/slapd.service' '/etc/systemd/system/multi-user.target.wants/slapd.service'
```

9.2.5.2. サービスの停止

現行のセッションで実行中の **slapd** サービスを停止するには、シェルプロンプトで **root** として以下を入力します。

```
~]# systemctl stop slapd.service
```

サービスが起動時に自動的に起動しないようにするには、**root** で以下を入力します。

```
~]# systemctl disable slapd.service
rm '/etc/systemd/system/multi-user.target.wants/slapd.service'
```

9.2.5.3. サービスの再起動

実行中の **slapd** サービスを再起動するには、シェルプロンプトで以下を入力します：

```
~]# systemctl restart slapd.service
```

これでサービスが停止し、即座に再起動します。設定を再読み込みするには、このコマンドを使用します。

9.2.5.4. サービスステータスの確認

slapd サービスが実行中かどうかを確認するには、シェルプロンプトで以下を入力します。

```
~]$ systemctl is-active slapd.service
active
```

9.2.6. OpenLDAP を使用した認証用のシステムの設定

OpenLDAP を使用して認証するようシステムを設定するには、LDAP サーバーとクライアントマシンの両方に適切な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。サーバーの設定方法に関する詳細は、「[OpenLDAP スイートのインストール](#)」と「[OpenLDAP サーバーの設定](#)」の手順に従って下さい。クライアントで、以下をシェルプロンプトで入力します。

```
~]# yum install openldap openldap-clients nss-pam-ldapd
```

9.2.6.1. 旧認証情報を LDAP 形式に移行

migrationtools パッケージは、認証情報を LDAP 形式に移行する時に役立つシェルスクリプトと Perl スクリプトのセットを提供します。このパッケージをインストールするには、シェルスクリプトで以下を入力します。

```
~]# yum install migrationtools
```

これにより、スクリプトを `/usr/share/migrationtools/` ディレクトリーにインストールします。インストール後、`/usr/share/migrationtools/migrate_common.ph` ファイルを編集して、以下の行を変更して正しいドメインを反映させます。以下に例を示します。

```
# Default DNS domain
$DEFAULT_MAIL_DOMAIN = "example.com";

# Default base
$DEFAULT_BASE = "dc=example,dc=com";
```

別の方法として、コマンドライン上で環境変数を直接指定することもできます。たとえば、`dc=example,dc=com` に設定されているデフォルトベースを使って `migrate_all_online.sh` スクリプトを実行するには、以下を入力します。

```
~]# export DEFAULT_BASE="dc=example,dc=com" \
/usr/share/migrationtools/migrate_all_online.sh
```

ユーザーデータベースの移行を実行するスクリプトを決定するには、[表9.8「一般的に使用される LDAP 移行スクリプト」](#) を参照してください。

表9.8 一般的に使用される LDAP 移行スクリプト

既存のネームサービス	LDAP は実行中ですか?	使用するスクリプト
<code>/etc</code> フラットファイル	はい	<code>migrate_all_online.sh</code>
<code>/etc</code> フラットファイル	いいえ	<code>migrate_all_offline.sh</code>
NetInfo	はい	<code>migrate_all_netinfo_online.sh</code>
NetInfo	いいえ	<code>migrate_all_netinfo_offline.sh</code>
NIS (YP)	はい	<code>migrate_all_nis_online.sh</code>
NIS (YP)	いいえ	<code>migrate_all_nis_offline.sh</code>

これらのスクリプトの使用方法に関する詳細情報は、`/usr/share/doc/migrationtools-version/` ディレクトリー内の `README` および `migration-tools.txt` ファイルを参照してください。

9.2.7. その他のリソース

以下のリソースは、LDAP (Lightweight Directory Access Protocol) に関するさらなる情報を提供します。システムに LDAP を設定する前に、以下の資料、特に『OpenLDAP Software Administrator's Guide (OpenLDAP ソフトウェア管理者ガイド)』を確認することを強く推奨します。

インストールされているドキュメント

以下のドキュメントは、`openldap-servers` パッケージでインストールされます。

- `/usr/share/doc/openldap-servers-version/guide.html` – 『OpenLDAP Software Administrator's Guide』のコピー。
- `/usr/share/doc/openldap-servers-version/README.schema` – インストールされているスキーマファイルが含まれる README ファイル。

さらに、`openldap`、`openldap-servers`、`openldap-clients` パッケージでインストールされる多くの man ページがあります。

クライアントアプリケーション

- `ldapadd(1)` – `ldapadd` コマンドの man ページで、LDAP ディレクトリーにエントリーを追加する方法について説明しています。
- `ldapdelete(1)` – `ldapdelete` コマンドの man ページで、LDAP ディレクトリー内のエントリーを削除する方法について説明しています。
- `ldapmodify(1)` – `ldapmodify` コマンドの man ページで、LDAP ディレクトリー内のエントリーを変更する方法について説明しています。
- `ldapsearch(1)` – `ldapsearch` コマンドの man ページで、LDAP ディレクトリー内でエントリーを検索する方法について説明しています。
- `ldappasswd(1)` – `ldappasswd` コマンドの man ページで、LDAP ユーザーのパスワードの設定および変更方法について説明しています。
- `ldapcompare(1)` – `ldapcompare` ツールの使用方法について説明しています。
- `ldapwhoami(1)` – `ldapwhoami` ツールの使用方法について説明しています。
- `ldapmodrdn(1)` – エントリーの RDN の変更方法について説明しています。

サーバーアプリケーション

- `slapd(8C)` – LDAP サーバーのコマンドラインオプションについて説明しています。

管理アプリケーション

- `slapadd(8C)` – エントリーを `slapd` データベースに追加するために使用されるコマンドラインオプションについて説明しています。
- `slapcat(8C)` – LDIF ファイルを `slapd` データベースから生成するために使用されるコマンドラインオプションについて説明しています。
- `slapindex(8C)` – `slapd` データベースのコンテンツに基づいてインデックスを再生成するために使用されるコマンドラインオプションについて説明しています。
- `slappasswd(8C)` – LDAP ディレクトリーのユーザーパスワードを生成するために使用されるコマンドラインオプションについて説明しています。

設定ファイル

- `ldap.conf(5)` – `ldap.conf` ファイルの `man` ページで、LDAP クライアントの設定ファイル内で利用可能なフォーマットおよびオプションについて説明しています。
- `slapd-config(5)` – `/etc/openldap/slapd.d` 設定ディレクトリ内で利用可能なフォーマットおよびオプションについて説明しています。

その他のリソース

- NSS データベースの後方互換性の実装に関する詳細は、『[OpenLDAP and Mozilla NSS Compatibility Layer](#)』を参照してください。
- OpenSSL を使用するための OpenLDAP の設定方法に関する情報は、『[How do I use TLS/SSL?](#)』を参照してください。

パート III. セキュアなアプリケーション

第10章 PAM (プラグ可能な認証モジュール) の使用

プラグ可能な認証モジュール(PAM)とは、認証と承認のための一般的なフレームワークです。Red Hat Enterprise Linux のほとんどのシステムアプリケーションは、認証および承認で基礎となる PAM 設定に依存しています。

10.1. PAM について

PAM (プラグ可能な認証モジュール) は、集中型認証メカニズムを提供します。システムアプリケーションは、これを使って認証を中央で設定されたフレームワークに中継できます。

PAM には異なるタイプの認証ソース (Kerberos や SSSD、NIS、ローカルファイルシステム) 用のモジュールがあるので、プラグ可能になっています。異なる認証ソースに優先順位を付けることができます。

PAM はモジュラー型アーキテクチャーであることから、管理者はシステム用の認証ポリシーを柔軟に設定することができます。PAM は以下のような理由で開発者および管理者にとって便利なシステムとなっています。

- PAM は、多様なアプリケーションで使用できる共通の認証スキームを提供します。
- PAM は、認証における制御と多大な柔軟性をシステム管理者に提供します。
- PAM は、完全に文書化された単一ライブラリーを提供します。開発者は、独自の認証スキームを作成することなくプログラムの作成ができます。

10.1.1. 他の PAM リソース

PAM には、PAM の使用および PAM を他のアプリケーションと統合拡張するためのモジュール作成に関する詳細かつ大規模なドキュメンテーションがあります。PAM と使用するほとんどのメジャーなモジュールおよび設定ファイルには、独自の man ページがあります。さらに、`/usr/share/doc/pam-version#/` ディレクトリーには、『System Administrators' Guide』、『Module Writers' Manual』、および『Application Developers' Manual』のほか、PAM の標準である DCE-RFC 86.0 が含まれています。

PAM 用のライブラリーは、<http://www.linux-pam.org> にあります。これは Linux-PAM プロジェクトの主要なディストリビューション Web サイトで、様々な PAM モジュールに関する情報やよくある質問、追加の PAM ドキュメンテーションがあります。

10.1.2. PAM モジュールのカスタマイズ

新規の PAM モジュールは、PAM 対応アプリケーションでいつでも作成、追加ができます。PAM 対応プログラムは、新規モジュールとそれが定義するメソッドを再編集したり修正したりすることなく、即座に使用することができます。このため開発者およびシステム管理者は、異なるプログラム用の認証メソッドを再コンパイルすることなく組み合わせて使用したりテストしたりすることができます。

モジュール作成に関するドキュメンテーションは、`/usr/share/doc/pam-devel-version#/` に格納されています。

10.2. PAM 設定ファイルについて

PAM 対応アプリケーションまたは サービスはそれぞれ `/etc/pam.d/` ディレクトリーにファイルがあり、これにはファイルがアクセスを制御するサービスと同じ名前が付けられています。たとえば、`login` プログラムはサービス名を `login` と定義し、`/etc/pam.d/login` という名前の PAM 設

定ファイルをインストールします。



警告

PAM の設定には、PAM 設定ファイルを手動で編集するのではなく、**authconfig** ツールを使用することが強く推奨されます。

10.2.1. PAM 設定ファイルの書式

各 PAM 設定ファイルには、モジュール (認証設定エリア) を定義するディレクティブとその制御もしくは属性が含まれています。

ディレクティブの構文はすべて簡単なもので、モジュールの目的 (インターフェース) とモジュールの設定を特定します。

```
module_interface control_flag module_name module_arguments
```

PAM 設定ファイルでは、モジュールインターフェースは以下のように最初のフィールドで定義されます。

```
auth required pam_unix.so
```

PAM インターフェースとは、本質的にはその特定のモジュールが実行可能な認証アクションのタイプのことです。PAM モジュールインターフェースでは 4 つのタイプが利用可能で、それぞれが認証および承認プロセスの別の要素に対応しています。

- **auth**—このモジュールインターフェースは、ユーザーを認証します。たとえば、パスワードの有効性を要求したり検証したりします。このインターフェースがあるモジュールは、グループメンバーシップといった認証情報も設定できます。
- **account**—このモジュールインターフェースは、アクセスが許可されたことを確認します。たとえば、ユーザーアカウントの期限が切れたか、またはユーザーが 1 日の特定の時間にログインを許可されるかどうかをチェックします。
- **password**—このモジュールインターフェースは、ユーザーのパスワード変更に使用されます。
- **session**—このモジュールインターフェースは、ユーザーセッションを設定、管理します。このインターフェースのあるモジュールは、ユーザーのホームディレクトリをマウントしたり、ユーザーのメールボックスを利用可能にするなど、アクセスの許可を必要とする追加タスクも実行できます。

個別のモジュールは、いずれかのインターフェース、またはすべてのインターフェースを提供できます。たとえば、**pam_unix.so** は 4 つすべてのモジュールインターフェースを提供します。

pam_unix.so といったモジュール名は、指定されたモジュールインターフェースのライブラリー名を PAM に提供します。ディレクトリー名は省略されています。アプリケーションが適切なバージョンの **libpam** にリンクされており、これがモジュールの適切なバージョンを見つけることができるためです。

PAM モジュールはすべて、呼び出されると成功か失敗の結果を生成します。**制御フラグ**がPAMに結果をどのように処理するかについて指示します。モジュールは特定の順番で記載することができ(スタック)、ユーザーのサービスへの認証全体において特定モジュールの成功もしくは失敗の重要性を制御フラグが判断します。

フラグには簡単なものはいくつかあり^[2]、これらの設定にはキーワードのみを使用します。

- **required** – 認証を継続するには、モジュール結果が成功する必要があります。この時点でテストが失敗すると、該当インターフェースを参照するすべてのモジュールテストの結果が完了するまでユーザーには通知されません。
- **requisite** – 認証を継続するには、モジュール結果が成功する必要があります。ただし、この時点でテストが失敗するとユーザーに直ちに通知され、そのメッセージには最初に失敗した **required** または **requisite** モジュールテストが反映されます。
- **sufficient** – モジュール結果は失敗した場合でも無視されます。ただし、**sufficient** のフラグの付いたモジュールが成功して、かつ **required** のフラグが付いたモジュールがそれまでに失敗していなければ、それ以上の結果は要求されず、ユーザーはサービスに認証されます。
- **optional** – モジュール結果は無視されます。**optional** のフラグのついたモジュールは、他のモジュールが該当インターフェースを参照しない場合にのみ、認証成功に必要となります。
- **include** – 他の制御とは違い、これはモジュール結果の処理には関係しません。このフラグは、特定パラメーターと合致する設定ファイル内のすべての行を取得し、それらを引数としてモジュールに追加します。

モジュールインターフェースのディレクティブは、重ねて配置することで**スタック化**が可能なので、複数のモジュールをまとめて1つの目的に使用することができます。



注記

モジュールの制御フラグが **sufficient** または **requisite** の値を使用している場合、モジュールの記載順序は認証プロセスで重要になります。

スタックを使用する場合に、管理者はユーザーを認証するための固有の条件が必要になります。たとえば、**setup** ユーティリティーは通常、PAM 設定ファイルにあるように、複数のスタック化されたモジュールを使用します。

```
[root@MyServer ~]# cat /etc/pam.d/setup
```

```
auth      sufficient pam_rootok.so
auth      include system-auth
account   required pam_permit.so
session   required pam_permit.so
```

- **auth sufficient pam_rootok.so** – この行は **pam_rootok.so** モジュールを使用して、現行ユーザーの UID が 0 であることを確認して、ユーザーが root かどうかをチェックします。テストが成功すると、他のモジュールは参照されず、コマンドが実行されます。テストが失敗すると、次のモジュールに移ります。
- **auth include system-auth** – この行には **/etc/pam.d/system-auth** モジュールのコンテンツが含まれており、認証のためにこのコンテンツを処理します。

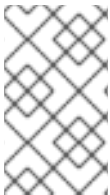
- **account required pam_permit.so**—この行は **pam_permit.so** モジュールを使って、**root** ユーザーもしくはコンソールにログインしているユーザーがシステム再起動をできるようにします。
- **session required pam_permit.so**—この行は、セッション設定に関するものです。**pam_permit.so** を使って **setup** ユーティリティーが失敗しないようにします。

PAM はいくつかのモジュール向けの認証中に 引数 を使って情報をプラグ可能なモジュールに渡します。

たとえば、**pam_pwquality.so** モジュールはパスワードの強度をチェックし、複数の引数を取ることができます。以下の例では、**enforce_for_root** で **root** ユーザーのパスワードでも強度チェックをパスする必要があることを指定し、**retry** でユーザーが強固なパスワードを 3 回入力できることを定義しています。

```
password requisite pam_pwquality.so enforce_for_root retry=3
```

無効な引数は通常無視され、PAM モジュールの成功や失敗には影響しません。ただし、モジュールのなかには無効な引数で失敗するものもあります。ほとんどのモジュールはエラーを **journald** サービスに報告します。**journald** の使用方法と関連の **journalctl** ツールに関する情報は、『[システム管理者のガイド](#)』を参照してください。



注記

journald サービスは Red Hat Enterprise Linux 7.1 で導入されました。Red Hat Enterprise Linux の以前のバージョンでは、ほとんどのモジュールはエラーを **/var/log/secure** ファイルに報告します。

10.2.2. 注釈付きの PAM 設定例

例10.1「[シンプルな PAM 設定](#)」は、PAM アプリケーション設定ファイルのサンプルです。

例10.1 シンプルな PAM 設定

```
#%PAM-1.0
auth required pam_securetty.so
auth required pam_unix.so nullok
auth required pam_nologin.so
account required pam_unix.so
password required pam_pwquality.so retry=3
password required pam_unix.so shadow nullok use_authtok
session required pam_unix.so
```

- 最初の行は、行頭のハッシュ記号 (#) が示すように、コメントになります。
- 2 行目から 4 行目は、ログイン認証用に 3 つのモジュールをスタックしています。

auth required pam_securetty.so—このモジュールは、ユーザーが **root** としてログインしようとしており、**/etc/securetty** ファイルが存在している場合に、そのユーザーのログイン先の TTY がファイルに記載されていることを確認します。

TTY がファイルに記載されていない場合は、**root** としてログインしようとする、**Login incorrect** メッセージが表示され、失敗します。

auth required pam_unix.so nullok—このモジュールはユーザーにパスワードを要求し、**/etc/passwd** と、存在する場合は **/etc/shadow** に保存されている情報を使用して、パスワードを確認します。

引数 **nullok** は **pam_unix.so** モジュールに空白のパスワードを許可するよう指示します。

- **auth required pam_nologin.so**—これは認証の最終ステップです。**/etc/nologin** ファイルが存在するかどうかを確認します。このファイルが存在してユーザーが **root** でない場合は、認証に失敗します。



注記

この例では、最初の **auth** モジュールが失敗しても、3 つすべての **auth** モジュールが確認されます。これにより、どの段階で認証に失敗したかをユーザーに知らせずに済みます。この情報が攻撃者の手に渡ると、システムの攻撃方法がより簡単に推測されてしまいます。

- **account required pam_unix.so**—このモジュールは、必要なアカウント確認を実行します。たとえば、シャドウパスワードが有効になっていると、**pam_unix.so** モジュールのアカウントインターフェースはアカウントの有効期間が切れているかどうか、またはユーザーが許可されている猶予期間内にパスワードを変更していないかを確認します。
- **password required pam_pwquality.so retry=3**—パスワードの有効期間が切れている場合は、**pam_pwquality.so** モジュールのパスワードコンポーネントが新たなパスワードを要求します。そして、新規に作成されたパスワードが辞書ベースのパスワード解読プログラムで容易に判断できるかどうかをテストします。

引数 **retry=3** は、テストに 1 回失敗しても、ユーザーは強固なパスワードを作成する機会があると 2 回あることを示しています。

- **password required pam_unix.so shadow nullok use_authtok**—この行は、**pam_unix.so** モジュールの **password** インターフェースを使って、プログラムがユーザーのパスワードを変更するかどうかを指定します。
 - 引数 **shadow** は、ユーザーのパスワード更新の際にシャドウパスワードを作成するようモジュールに指示します。
 - 引数 **nullok** は、ユーザーが空白のパスワードから変更できるようにし、それ以外の場合は **null** パスワードをアカウントロックとして扱うようモジュールに指示します。
 - この行の最後の引数 **use_authtok** は、PAM モジュールのスタック化の際における順序の重要性の例を提供します。この引数は、ユーザーに新規パスワードを要求しないようモジュールに指示します。代わりに、以前のパスワードモジュールが記録したパスワードを受け入れます。これにより、新規パスワードはすべて、受け入れ前にパスワードの安全テスト **pam_pwquality.so** をパスする必要があります。
- **session required pam_unix.so**—最後の行は、**pam_unix.so** モジュールのセッション引数にセッションを管理するよう指示します。このモジュールはユーザー名とサービスタイプを各セッションの最初と最後に **/var/log/secure** に記録します。このモジュールは他のセッションモジュールとスタック化した追加機能を補うことができます。

10.3. PAM と管理認証情報のキャッシング

GNOME の **control-center** など、Red Hat Enterprise Linux 内の多くのグラフィカル管理ツールは、システム特権を持つユーザーに最大 5 分間の **pam_timestamp.so** モジュール使用を提供します。**pam_timestamp.so** が実行中の端末をユーザーが離れると、このコンソールに物理的にアクセス可能な人物がマシンを操作できるようになってしまうため、このメカニズムの作動方法を理解することは重要になります。

PAM タイムスタンプスキームでは、グラフィカルな管理アプリケーションは起動時に **root** パスワードをユーザーに求めます。ユーザーが認証されると、**pam_timestamp.so** モジュールはタイムスタンプファイルを作成します。デフォルトでは、これは **/var/run/sudo/** ディレクトリーに作成されます。すでにタイムスタンプファイルがある場合は、グラフィカル管理プログラムはパスワードを要求しません。代わりに **pam_timestamp.so** モジュールはタイムスタンプファイルをリフレッシュして、ユーザーに無条件の管理者アクセスをさらに 5 分間確保します。

タイムスタンプファイルの実際の状態は、**/var/run/sudo/user** ディレクトリーで確認できます。デスクトップの場合は、関連ファイルは **unknown:root** になります。これが存在してそのタイムスタンプが 5 分未満の場合、認証情報は有効です。

タイムスタンプファイルが存在すると、パネルの通知スペースに認証アイコンが表示されます。



図10.1 認証アイコン

10.3.1. タイムスタンプファイルの削除

PAM タイムスタンプがアクティブな状態のコンソールは、放置する前にタイムスタンプファイルを破棄することが推奨されます。グラフィカル環境でこれを実行するには、パネルの認証アイコンをクリックします。これでダイアログボックスが開きます。**Forget Authorization** ボタンをクリックしてアクティブなタイムスタンプファイルを破棄します。

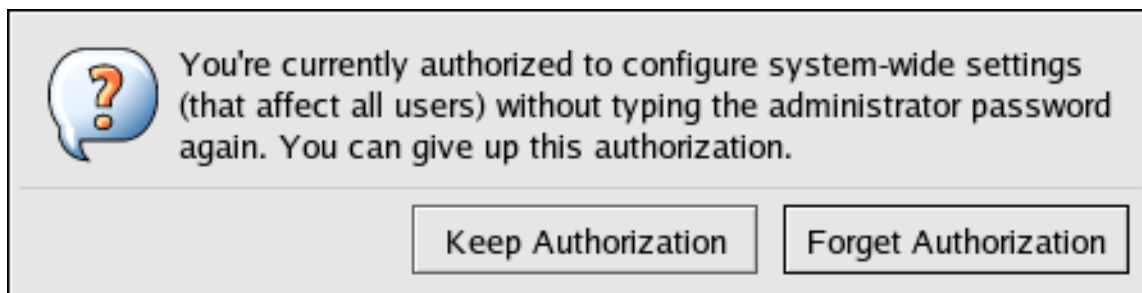


図10.2 認証ダイアログを閉じる

PAM タイムスタンプファイルには、以下の重要な特徴があります。

- **ssh** を使用してシステムにリモートでログインしている場合は、**/sbin/pam_timestamp_check -k root** コマンドを使用してタイムスタンプファイルを破棄します。
- 権限のあるアプリケーションが起動されたものと同じ端末ウィンドウから **/sbin/pam_timestamp_check -k root** コマンドを実行します。

- **pam_timestamp.so** モジュールを最初に起動したログイン済みユーザーは、**/sbin/pam_timestamp_check -k** コマンドを実行するユーザーである必要があります。root でこのコマンドを実行しないでください。
- デスクトップでアイコン上の **Forget Authorization** アクションを使用せずに認証情報を破棄するには、**/sbin/pam_timestamp_chec** コマンドを使用します。

```
/sbin/pam_timestamp_check -k root </dev/null >/dev/null 2>/dev/null
```

他の方法は、コマンドが実行される PTY から認証情報を削除するだけです。

pam_timestamp_check を使用してタイムスタンプファイルを破棄する詳細情報については、**pam_timestamp_check** の man ページを参照してください。

10.3.2. 一般的な pam_timestamp ディレクティブ

pam_timestamp.so モジュールはいくつかのディレクティブを受け付けます。最も一般的なものは以下の 2 つです。

- **timestamp_timeout** – タイムスタンプの有効期間を秒単位で指定します。デフォルト値は 300 (5 分間) です。
- **timestampdir** – タイムスタンプファイルの保存先となるディレクトリーを指定します。デフォルト値は **/var/run/sudo/** です。

10.4. PAM サービスのドメイン制限



重要

この機能を使用するには、システムで **SSSD** を実行している必要があります。

SSSD は、PAM サービスがアクセス可能なドメインを制限することができます。**SSSD** は、特定の PAM サービスの実行ユーザーをもとに、PAM サービスからの認証要求を評価します。PAM サービスが **SSSD** ドメインにアクセスできるかどうかは、PAM サービスのユーザーがそのドメインにアクセスできるかどうかにより左右されます。

ユースケースの 1 つとして、外部ユーザーが FTP サーバーに認証可能となっている環境が挙げられます。FTP サーバーは権限のない別のユーザーが実行しており、社内のアカウントと分けられた、特定の **SSSD** ドメインにのみ認証可能であるべきです。この機能を使うと、管理者は FTP ユーザーが FTP PAM 設定ファイルに指定されている特定のドメインのみに認証できるようにすることができます。



注記

この機能は、PAM モジュールのパラメーターとして別の設定ファイルを使用できた **pam_ldap** などのレガシー PAM モジュールに似ています。

ドメインへのアクセス制御オプション

以下のオプションを使用して、特定のドメインのみにアクセスを制限することができます。

/etc/sss/sss.conf の **pam_trusted_users**

このオプションは、SSSD デーモンが信頼する PAM サービスを表す数字の UID またはユーザー名のリストを受け入れます。デフォルトの設定は **all** で、すべてのユーザーを信頼し、どのドメインにもアクセスできるという意味です。

/etc/sss/sss.conf の pam_public_domains

このオプションは、パブリックの SSSD ドメインの一覧を受け入れます。パブリックドメインは、信頼されていない PAM サービスユーザーでもアクセス可能なドメインです。また、このオプションは **all** と **none** の値も受け入れます。デフォルト値は **none** で、どのドメインもパブリックではないので、信頼されていないサービスユーザーはどのドメインにもアクセスできません。

PAM 設定ファイルの domains

このオプションは、PAM サービスが認証可能なドメイン一覧を指定します。ドメインを指定せずに **domains** を使用すると、PAM サービスはどのドメインにも認証できません。以下に例を示します。

```
auth      required    pam_sss.so domains=
```

PAM 設定ファイルで **domains** を使用しない場合は、PAM サービスは、信頼済みのユーザーでサービスが実行されているという条件下であれば、すべてのドメインに対する認証を通過できます。

SSSD 設定ファイル (/etc/sss/sss.conf) の **domains** オプションでは、SSSD が認証を試行するドメイン一覧を指定します。PAM 設定ファイルの **domains** オプションは、**sss.conf** のドメイン一覧を拡張できず、もう少し短い一覧を指定して **sss.conf** のドメイン一覧を制限することができます。そのため、ドメインが PAM ファイルで指定されているが **sss.conf** で指定されていない場合には、PAM サービスはそのドメインに対して認証できなくなります。

デフォルト設定の **pam_trusted_users = all** および **pam_public_domains = none** は、すべての PAM サービスユーザーが信頼されており、どのドメインにもアクセスできることを指定します。PAM 設定ファイルの **domains** オプションは、アクセス可能なドメインを制限するような状況で使用できます。

sss.conf に **pam_public_domains** が含まれているが、PAM 設定ファイルの **domains** を使用してドメインを指定する場合は、**pam_public_domains** のドメインも指定する必要がでてくる可能性があります。**pam_public_domains** を使用するが、必要なドメインが含まれていない場合には、PAM サービスは、信頼されていないユーザーで実行されている場合はドメインへの認証が正常に実行できません。



注記

PAM 設定ファイルで定義したドメインの制限は、ユーザー検索ではなく、認証アクションにのみ適用されます。

pam_trusted_users と **pam_public_domains** オプションに関する詳細情報は **sss.conf(5)** の man ページを参照してください。PAM 設定ファイルで使用する **domains** オプションに関する詳細情報は **pam_sss(8)** の man ページを参照してください。

例10.2 PAM サービスのドメイン制限

PAM サービスが認証可能なドメインを制限するには以下を行います。

1. SSSD が必要なドメインにアクセスするように設定されていることを確認します。SSSD が認証可能なドメインは、**/etc/sss/sss.conf** ファイルの **domains** オプションで定義

されます。

```
[sssd]
domains = domain1, domain2, domain3
```

2. PAM サービスが認証可能なドメインを指定します。これには、以下のように PAM 設定ファイルの **domains** オプションを設定します。

```
auth          sufficient    pam_sss.so forward_pass domains=domain1
account       [default=bad  success=ok user_unknown=ignore]
pam_sss.so
password      sufficient    pam_sss.so use_authtok
```

PAM サービスは **domain1** に対してのみ認証可能です。

[2] 設定可能な制御フラグには複雑なものもあります。これらは、**属性=値**のペアで設定します。属性の完全な一覧は、**pam.d** の **man** ページで確認できます。

第11章 KERBEROS の使用

ネットワーク内でのシステムのセキュリティと整合性の維持は非常に重要であり、これはネットワークインフラストラクチャー内のすべてのユーザー、アプリケーション、サービスおよびサーバーに及びます。この維持には、ネットワーク上で実行中のすべてサービスやそれらがどのように使用されているかを理解していることが必要です。このセキュリティ維持の中心的なタスクは、アプリケーションやサービスへのアクセスの維持と、そのアクセスの実施になります。

Kerberos は通常のパスワードベースの認証よりもはるかに安全です。Kerberos では、他のマシン上のサービスにアクセスする場合でも、パスワードが送信されないからです。

Kerberos は、ユーザーとマシンの両方がネットワークに対して自らを識別し、管理者が設定した領域とサービスへの定義済みかつ制限されたアクセスをユーザーとマシンが受けられるようにするメカニズムを提供します。Kerberos はエンティティの ID を検証してそれらを **認証** するほか、この認証情報データを保護することで、外部の人間によるこのデータへのアクセス、使用または改ざんを防ぎます。

11.1. KERBEROS について

Kerberos は対称鍵暗号^[3]を用いてネットワークサービスに対してユーザーを認証します。つまり、パスワードがネットワーク経由で送信されることは決してありません。

そのため、ユーザーが Kerberos を使用してネットワークサービスに対して認証を行う際に、ネットワークトラフィックを監視してパスワードの収集を図っている不正なユーザーを効果的に阻止することができます。

11.1.1. Kerberos の基本的な仕組み

従来のほとんどのネットワークサービスではパスワードベースの認証スキームを使用しており、その場合はユーザーが特定のネットワークサーバーにアクセスするためにパスワードを提供します。ただし、多くのサービスにおける認証情報は暗号化されずに送信されています。このようなスキームをセキュアにするには、ネットワークを外部からアクセス不可とし、ネットワーク上のすべてのコンピューターとユーザーは信頼できるものであり、信頼されているものでなければなりません。

シンプルなパスワードベースの認証を使う際には、インターネットに接続されているネットワークが安全であるとは想定できません。ネットワークにアクセスする攻撃者は誰でもパケットアナライザーもしくはパケットスニフアーを使用してユーザー名とパスワードを傍受し、ユーザーアカウントの安全性を脅かすことができるので、セキュリティインフラストラクチャー全体の整合性が脅かされます。

Kerberos はネットワーク経由で暗号化されていないパスワード送信をなくし、攻撃者がネットワークを傍受する潜在的脅威を取り除きます。

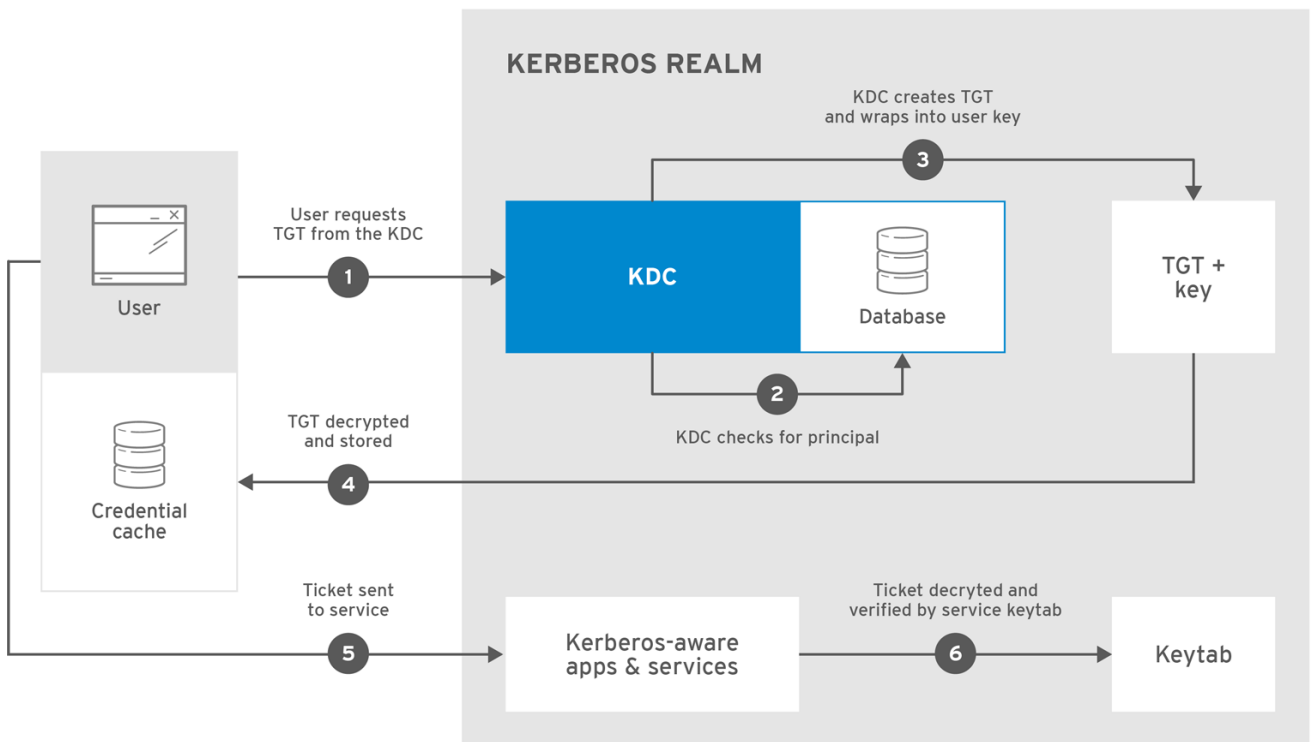
シンプルなパスワード認証で個別のユーザーが個別のネットワークサービスに対して認証を行うのではなく、Kerberos は対称暗号と信頼できるサードパーティー (**キー配布センター KDC**) を用いてユーザーをネットワークサービスのスイートに対して認証します。その KDC とセカンダリー KDC が管理するコンピューターが **レルム** を構成します。

ユーザーが KDC に対して認証を行うと、KDC はそのセッションに特定した認証情報のセット (**チケット**) をユーザーのマシンに送り返します。Kerberos 対応のサービスでは、ユーザーがパスワードを使用して認証する必要はなく、サービスすべてがユーザーのマシン上でこのチケットを探します。

図11.1 「Kerberos 認証の場合」にあるように、各ユーザーは一意の ID で KDC に識別されます。この ID は **プリンシパル**と呼ばれます。Kerberos 対応ネットワーク上でユーザーがワークステーションにログインすると、このユーザーのプリンシパルが認証サーバーから **ticket-granting ticket** (または

TGT) のリクエストの一部として KDC に送信されます。このリクエストは、ユーザーが処理を意識しなくてもよいようにログインプログラムによって送信するか、ユーザーがログイン後に手動で **kinit** プログラムを用いて送信することができます。

すると KDC はデータベース内でプリンシパルを確認します。プリンシパルが見つかったら、KDC は TGT を作成し、ユーザーの鍵を使ってこれを暗号化し、TGT をそのユーザーに送信します。



RHEL_404973_1016

図11.1 Kerberos 認証の場合

次にクライアント上のログインもしくは **kinit** プログラムがユーザーの鍵を使って TGT を暗号解除します。ユーザーの鍵は、ログインもしくは **kinit** プログラムがユーザーのパスワードから計算します。ユーザーの鍵はクライアントマシン上でのみ使用され、ネットワーク経由では送信されません。KDC が送信したチケット (または認証情報) はローカルストアである **認証情報キャッシュ (ccache)** に保存されます。Kerberos 対応サービスはこのキャッシュをチェックすることができます。Red Hat Enterprise Linux 7 では、以下のタイプの認証情報キャッシュに対応しています。

- **KEYRING** 永続性のある **KEYRING ccache** タイプが Red Hat Enterprise Linux 7 のデフォルトになります。
- **FILE**
- **DIR**
- **MEMORY**

認証後は、サーバーは **kinit** を確認するのではなく、認識されたプリンシパルとそれらの鍵の暗号化されていないリストをチェックできます。このリストは **keytab** に保持されます。

TGT は特定期間の後に (通常は 10 から 24 時間) 有効期限が切れるように設定され、クライアントマシンの認証情報キャッシュに保存されます。TGT に期限が設定されているのは、攻撃者が TGT を使用できる期間を短時間にするためです。TGT の発行後は、TGT の有効期限が切れるまで、またはユーザーがログアウトして再度ログインするまで、ユーザーはパスワードを再入力する必要がありません。

ユーザーがネットワークサービスにアクセスする必要がある場合は、クライアントのソフトウェアは

TGT を使って **ticket-granting** サーバー (TGS) から該当サービス用の新規チケットを要求します。サービスチケットは該当サービスに対するユーザーの認証を透過的に行うために使用されます。

11.1.2. Kerberos プリンシパル名

プリンシパルはユーザーやサービスだけでなく、そのエンティティーが属するレルムも特定します。プリンシパル名は、識別子とレルムの 2 つからなります。

```
identifier@REALM
```

ユーザーの場合は **identifier** は Kerberos ユーザー名のみで、サービスの場合は **identifier** はサービス名と当該サービスが実行されているマシンのホスト名を組み合わせたものです。

```
service/FQDN@REALM
```

service 名は、**host**、**ldap**、**http**、および **DNS** といったようにサービスタイプに固有の、大文字と小文字を区別した文字列です。すべてのサービスに明らかなプリンシパル識別子があるわけではありません。たとえば、**sshd** デーモンはホストサービスプリンシパルを使用します。

ホストプリンシパルは通常、**/etc/krb5.keytab** に保存されています。

Kerberos がチケットを要求する際は常に、ドメイン名のエイリアス (DNS CNAME レコード) を対応する DNS アドレス (A または AAAA レコード) に解決します。アドレスレコードからのホスト名は、サービスまたはホストプリンシパルが作成される際に使用されます。

例を示します。

```
www.example.com CNAME web-01.example.com
web-01.example.com A 192.0.2.145
```

サービスは、ホストの CNAME エイリアスを使ってホストに接続を試みます。

```
$ ssh www.example.com
```

Kerberos サーバーは解決済みホスト名 **web-01.example.com@EXAMPLE.COM** のチケットを要求するので、ホストプリンシパルは **host/web-01.example.com@EXAMPLE.COM** となります。

11.1.3. ドメインからレルムへのマッピング

クライアントが特定サーバー上で実行中のサービスにアクセスしようとする際は、サービス名 (ホスト) とサーバー名 (**foo.example.com**) は分かっていますが、ネットワーク上には複数のレルムが導入されることが可能なので、クライアントはサービスが存在する Kerberos レルムの名前を推測する必要があります。

デフォルトでは、レルム名はサーバーのドメイン名をすべて大文字にしたものになります。

```
foo.example.org → EXAMPLE.ORG
foo.example.com → EXAMPLE.COM
foo.hq.example.com → HQ.EXAMPLE.COM
```

設定によってはこれで十分な場合もありますが、派生したレルム名が存在しないレルムの名前になることもあります。そのような場合には、サーバーの DNS ドメイン名からレルム名へのマッピングをクライアントシステムの **/etc/krb5.conf** ファイルの **domain_realm** セクションで指定する必要があります。

ます。例を示します。

```
[domain_realm]
.example.com = EXAMPLE.COM
example.com = EXAMPLE.COM
```

上記の設定では、2つのマッピングを指定しています。最初のマッピングでは、**example.com** DNS ドメインのシステムはすべて **EXAMPLE.COM** レルムに所属すると指定しています。2つ目では、**example.com** という名前そのものも同じレルムに所属することを指定しています。ドメインと特定ホストの違いは、最初のピリオド記号の有無で決まります。このマッピングは "**_kerberos TXT**" レコードを使って DNS に直接保存することもできます。例を示します。

```
$ORIGIN example.com
_kerberos TXT "EXAMPLE.COM"
```

11.1.4. 環境要件

Kerberos はマシン名の解決に依存していることから、作業中のドメインネームサービス (DNS) を必要とします。ネットワーク上の DNS エントリーとホストは両方とも適切に設定する必要があります。これは **/usr/share/doc/krb5-server-version-number** にある Kerberos ドキュメンテーションで説明されています。

Kerberos 認証を許可するアプリケーションは、時間の同期を必要とします。ネットワーク上のマシン間におけるおおよそのクロック同期は、**ntpd** などのサービスを使って設定できます。**ntpd** サービスに関する情報は、**/usr/share/doc/ntp-version-number/html/index.html** にあるドキュメンテーションか **ntpd(8)** の man ページを参照してください。



注記

Red Hat Enterprise Linux 7 上で実行中の Kerberos クライアントは KDC との自動時間調整をサポートしており、厳密な時間要件はありません。これにより、Red Hat Enterprise Linux 7 で IdM クライアントを導入する際には時間の差異に対する耐性が高くなります。

11.1.5. Kerberos 導入における考慮点

Kerberos は一般的かつ重大なセキュリティの脅威を排除しますが、以下のような理由から実装は容易ではありません。

- Kerberos は、各ユーザーは信頼されているものの、信頼できないネットワーク上で信頼できないホストを使用していることを想定しています。Kerberos の第一の目的は、暗号化されていないパスワードをネットワーク経由で送信されないようにすることです。ただし、認証に使用されるチケットを発行するあるホスト (KDC) に正当なユーザー以外の誰かがアクセスすると、Kerberos 認証システム全体が危険にさらされます。
- アプリケーションが Kerberos を使用するには、そのソースを修正して Kerberos ライブラリーに適切な呼び出しを行う必要があります。このような修正を受けたアプリケーションは、Kerberos に対応しているとみなされます。アプリケーションのなかには、そのサイズやデザインが理由でこれが問題になるものもあります。また、互換性のない他のアプリケーションでは、サーバーとクライアントが通信する方法を変更する必要があります。これはかなり大量のプログラミングが必要になる可能性があります。多くの場合、デフォルトで Kerberos 対応となっていないクローズソースのアプリケーションが最も問題のあるものとなります。
- Kerberos でネットワークの安全を図るには、暗号化されていないパスワードを送信するすべて

のクライアント/サーバーアプリケーションのバージョンで Kerberos 対応のものを使用するか、そのようなクライアント/サーバーアプリケーションをまったく使用しないかのどちらかにする必要があります。

- **/etc/passwd** や **/etc/shadow** といった標準 UNIX パスワードデータベースから Kerberos パスワードデータベースへのユーザーパスワードの移行は、面倒な作業になります。このタスクを実行する自動メカニズムはありません。移行方法は、Kerberos を導入する方法によって大幅に異なります。Identity Management 機能の使用が推奨されるのは、このためです。これには移行のための特別のツールとメソッドが備わっています。



警告

ネットワーク上のユーザーが Kerberos 非対応サービスに対してプレーンテキストでパスワードを送信すると、Kerberos システムは危険にさらされます。(telnet および FTP を含む) Kerberos 非対応サービスを使用しないことが強く推奨されます。SSH や SSL でセキュアとなったサービスのような他の暗号化プロトコルは暗号化されていないサービスよりは優れているものの、それでも理想的なものではありません。

11.1.6. Kerberos に関するその他のリソース

Kerberos の導入には柔軟性があることから、その実装は複雑なサービスとすることが可能です。表 11.1 「外部の Kerberos 資料」と表 11.2 「Kerberos man の重要なページ」では、Kerberos の使用に関する詳細情報で最重要もしくは最も有益なソースを一覧表示しています。

表11.1 外部の Kerberos 資料

資料	場所
Kerberos V5 Installation Guide (PostScript と HTML の両方)	<code>/usr/share/doc/krb5-server-version-number</code>
Kerberos V5 System Administrator's Guide (PostScript と HTML の両方)	<code>/usr/share/doc/krb5-server-version-number</code>
Kerberos V5 UNIX User's Guide (PostScript と HTML の両方)	<code>/usr/share/doc/krb5-workstation-version-number</code>
Kerberos: The Network Authentication Protocol (MIT のウェブページ)	http://web.mit.edu/kerberos/www/
『Designing an Authentication System: a Dialogue in Four Scenes』、Bill Bryant 著 1988。Theodore Ts'o による修正 1997。このドキュメントは、2 人の開発者による Kerberos スタイルの認証システムについての考察です。2 人の議論は会話形式となっており、Kerberos をまったく知らない読者にとっても分かりやすいものになっています。	http://web.mit.edu/kerberos/www/dialogue.html

資料	場所
ネットワークの Kerberos 対応化に関する記事	http://www.ornl.gov/~jar/HowToKerb.html

man ページのファイルは、**man command_name** を実行すると開きます。

表11.2 Kerberos man の重要なページ

Man ページ	説明
クライアントアプリケーション	
kerberos	Kerberos システムへの導入では、認証情報がどのように機能するかが説明され、Kerberos チケットの取得および破棄に関する推奨事項が提供されます。 man ページの下部では、関連する man ページが多く参照されています。
kinit	このコマンドを使用して ticket-granting ticket を取得、キャッシュする方法が説明されています。
kdestroy	このコマンドを使用して Kerberos 認証情報を破棄する方法が説明されています。
klist	このコマンドを使用して、キャッシュされた Kerberos 認証情報を一覧表示する方法が説明されています。
管理アプリケーション	
kadmin	このコマンドを使用して Kerberos V5 データベースを管理する方法が説明されています。
kdb5_util	このコマンドを使用して Kerberos V5 データベース上で低レベルの管理機能を作成、実行する方法が説明されています。
サーバーアプリケーション	
krb5kdc	Kerberos V5 KDC に利用可能なコマンドラインオプションが説明されています。
kadmind	Kerberos V5 管理サーバー に利用可能なコマンドラインオプションが説明されています。
設定ファイル	
krb5.conf	Kerberos V5 ライブラリー用の設定ファイル内で利用可能な形式とオプションを説明しています。

Man ページ	説明
<code>kdc.conf</code>	Kerberos V5 AS および KDC 用の設定ファイル内で利用可能な形式とオプションを説明しています。

11.2. KERBEROS KDC の設定

マスター KDC を最初にインストールして設定した後に、必要なセカンダリーサーバーをインストールします。



重要

Kerberos KDC を手動で設定することは推奨されません。Red Hat Enterprise Linux 環境に Kerberos を導入する際は、Identity Management 機能を使用することが推奨されます。

11.2.1. マスター KDC サーバーの設定



重要

KDC システムは専用マシンにしてください。このマシンは非常に安全である必要があります。可能な場合は、このマシン上で KDC 以外のサービスを実行しないでください。

1. KDC に必要なパッケージをインストールします。

```
[root@server ~]# yum install krb5-server krb5-libs krb5-workstation
```

2. `/etc/krb5.conf` と `/var/kerberos/krb5kdc/kdc.conf` の設定ファイルをレルム名とドメインからレルムへのマッピングを反映させるように編集します。例を示します。

```
[logging]
default = FILE:/var/log/krb5libs.log
kdc = FILE:/var/log/krb5kdc.log
admin_server = FILE:/var/log/kadmind.log

[libdefaults]
default_realm = EXAMPLE.COM
dns_lookup_realm = false
dns_lookup_kdc = false
ticket_lifetime = 24h
renew_lifetime = 7d
forwardable = true
allow_weak_crypto = true

[realms]
EXAMPLE.COM = {
    kdc = kdc.example.com.:88
    admin_server = kdc.example.com
    default_domain = example.com
}
```

```
[domain_realm]
.example.com = EXAMPLE.COM
example.com = EXAMPLE.COM
```

シンプルなレルムは、**EXAMPLE.COM**と **example.com**を適切なドメイン名で置き換え(大文字と小文字を正確な形式になるよう確認してください)、KDC を **kerberos.example.com** から Kerberos サーバーに変更することで構成できます。慣例では、レルム名はすべて大文字、DNS ホスト名およびドメイン名はすべて小文字になります。これらの設定ファイルの **man** ページには、ファイル形式に関する詳細が記載されています。

3. **kdb5_util** ユーティリティを使用してデータベースを作成します。

```
[root@server ~]# kdb5_util create -s
```

create コマンドは、Kerberos レルム用の鍵を保存するデータベースを作成します。**-s** 引数は、マスター鍵を保存する **stash** ファイルを作成します。鍵を読み取る **stash** ファイルがない場合、Kerberos サーバー (**krb5kdc**) は起動時に毎回、ユーザーに対してマスターサーバーのパスワード(これを使って鍵を再生成することが可能)を要求します。

4. **/var/kerberos/krb5kdc/kadm5.ac1** ファイルを編集します。**kadmind** はこのファイルを使って Kerberos データベースへの管理アクセスがあるプリンシパルとそのアクセスレベルを判断します。以下に例を示します。

```
*/admin@EXAMPLE.COM *
```

ほとんどのユーザーはデータベース内で単一プリンシパル(**joe@EXAMPLE.COM**のような **NULL** または空白のインスタンス)により表示されます。この設定では、**admin** のインスタンスがある 2 番目のプリンシパル(たとえば **joe/admin@EXAMPLE.COM**)を持つユーザーは、レルムの Kerberos データベースに対して完全な管理制御を行使することができます。

kadmind がサーバー上で開始されると、ユーザーはレルム内のクライアントもしくはサーバー上で **kadmin** を実行することでこのサービスにアクセスできます。ただし、データベースの編集が可能なのは(自身のパスワード変更を除く) **kadm5.ac1** ファイルに名前が記載されているユーザーのみになります。



注記

kadmin ユーティリティはネットワーク経由で **kadmind** サーバーと通信し、Kerberos を使って認証を処理します。このため、ネットワーク経由でサーバーに接続してこの処理を行う前に、最初のプリンシパルが存在している必要があります。最初のプリンシパルは、**kadmin.local** コマンドで作成します。このコマンドは、KDC と同じホスト上で使用するよう特別に設計されており、認証に Kerberos を使用しません。

5. KDC ターミナルで **kadmin.local** を使用して最初のプリンシパルを作成します。

```
[root@server ~]# kadmin.local -q "addprinc username/admin"
```

6. 以下のコマンドを使用して Kerberos を起動します。

```
[root@server ~]# systemctl start krb5kdc.service
[root@server ~]# systemctl start kadmind.service
```

7. **kadmin** 内で **addprinc** コマンドを使用してユーザーにプリンシパルを追加します。**kadmin** および **kadmin.local** は、KDC のコマンドラインインターフェースになります。このため、**addprinc** のようなコマンドの多くは、**kadmin** プログラムの起動後に利用可能になります。詳細は、**kadmin** の **man** ページを参照してください。
8. KDC がチケットを発行していることを確認します。まず、**kinit** を実行してチケットを取得し、認証情報キャッシュファイルに保存します。次に **klist** を使ってキャッシュ内の認証情報一覧を表示し、**kdestroy** を使ってキャッシュとそこに含まれる認証情報を破棄します。



注記

デフォルトでは、**kinit** は同一のシステムログインユーザー名 (Kerberos サーバーではなく) を使って認証を試みます。このユーザー名が Kerberos データベース内のプリンシパルに対応しない場合は、**kinit** はエラーメッセージを発行します。この場合は、コマンドラインで **kinit** の引数として適切なプリンシパルの名前を指定してください。

```
kinit principal
```

11.2.2. セカンダリー KDC の設定

あるレルムに複数の KDC がある場合、1 つの KDC (マスター KDC) が書き込み可能なレルムデータベースのコピーを保持し、**kadmind** を実行します。マスター KDC はレルムの管理サーバーでもあります。追加のセカンダリー KDC はデータベースの読み取り専用コピーを保持して、**kpropd** を実行します。

マスターおよびスレーブを伝達するステップでは、マスター KDC がデータベースを一時ダンプファイルにダンプして、そのファイルを各スレーブに送信する必要があります。このファイルは、そのダンプファイルのコンテンツでこれ以前に受信したデータベースの読み取り専用コピーを上書きします。

セカンダリー KDC を設定するには、以下の手順にしたがいます。

1. KDC に必要なパッケージをインストールします。

```
[root@slavekdc ~]# yum install krb5-server krb5-libs krb5-workstation
```

2. マスター KDC の **krb5.conf** と **kdc.conf** のファイルをセカンダリー KDC にコピーします。
3. マスター KDC で root シェルから **kadmin.local** を起動します。

1. **kadmin.local add_principal** コマンドを使ってマスター KDC の **host** サービス用の新規エントリーを作成します。

```
[root@slavekdc ~]# kadmin.local -r EXAMPLE.COM
Authenticating as principal root/admin@EXAMPLE.COM with
password.
kadmin: add_principal -randkey host/masterkdc.example.com
Principal "host/masterkdc.example.com@EXAMPLE.COM" created.
kadmin: ktadd host/masterkdc.example.com
Entry for principal host/masterkdc.example.com with kvno 3,
encryption type Triple DES cbc mode with HMAC/sha1 added to
keytab WRFILE:/etc/krb5.keytab.
Entry for principal host/masterkdc.example.com with kvno 3,
```

```

encryption type ArcFour with HMAC/md5 added to keytab
WRFFILE:/etc/krb5.keytab.
Entry for principal host/masterkdc.example.com with kvno 3,
encryption type DES with HMAC/sha1 added to keytab
WRFFILE:/etc/krb5.keytab.
Entry for principal host/masterkdc.example.com with kvno 3,
encryption type DES cbc mode with RSA-MD5 added to keytab
WRFFILE:/etc/krb5.keytab.
kadmin: quit

```

2. **kadmin.local ktadd** コマンドを使ってサービス用にランダムな鍵を設定し、その鍵をマスターのデフォルト **keytab** ファイルに保存します。



注記

この鍵は **kprop** コマンドでセカンダリーサーバーへの認証に使用します。インストールされているセカンダリー KDC サーバーの数にかかわらず、この設定が必要なのは1回のみです。

4. セカンダリー KDC で root シェルから **kadmin** を起動します。

1. **kadmin.local add_principal** コマンドを使ってセカンダリー KDC の **host** サービス用の新規エントリーを作成します。

```

[root@slavekdc ~]# kadmin -p jsmith/admin@EXAMPLE.COM -r
EXAMPLE.COM
Authenticating as principal jsmith/admin@EXAMPLE.COM with
password.
Password for jsmith/admin@EXAMPLE.COM:
kadmin: add_principal -randkey host/slavekdc.example.com
Principal "host/slavekdc.example.com@EXAMPLE.COM" created.
kadmin: ktadd host/slavekdc.example.com@EXAMPLE.COM
Entry for principal host/slavekdc.example.com with kvno 3,
encryption type Triple DES cbc mode with HMAC/sha1 added to
keytab WRFFILE:/etc/krb5.keytab.
Entry for principal host/slavekdc.example.com with kvno 3,
encryption type ArcFour with HMAC/md5 added to keytab
WRFFILE:/etc/krb5.keytab.
Entry for principal host/slavekdc.example.com with kvno 3,
encryption type DES with HMAC/sha1 added to keytab
WRFFILE:/etc/krb5.keytab.
Entry for principal host/slavekdc.example.com with kvno 3,
encryption type DES cbc mode with RSA-MD5 added to keytab
WRFFILE:/etc/krb5.keytab.
kadmin: quit

```

2. **kadmin.local ktadd** コマンドを使ってサービス用にランダムな鍵を設定し、その鍵をセカンダリーサーバーのデフォルト **keytab** ファイルに保存します。この鍵は、クライアントの認証時に **kpropd** で使用します。
5. セカンダリー KDC はサービス鍵を使って、接続するクライアントを認証することができます。もちろん、すべてのクライアントが新規レルムデータベースで **kprop** サービスの提供を許可されるべきではありません。アクセスを制限するために、セカンダリー KDC 上の **kprop** サービス

スは、そのプリンシパル名が `/var/kerberos/krb5kdc/kpropd.ac1` に記載されているクライアントからの更新しか受け付けません。

マスター KDC のホストサービス名をこのファイルに追加します。

```
[root@slavekdc ~]# echo host/masterkdc.example.com@EXAMPLE.COM >
/var/kerberos/krb5kdc/kpropd.ac1
```

6. セカンダリー KDC がデータベースのコピーを取得すると、その暗号化に使用されたマスター鍵も必要になります。KDC データベースのマスター鍵がマスター KDC 上の `stash` ファイル (通常、`/var/kerberos/krb5kdc/.k5.REALM`) に保存されている場合は、これを安全な方法でセカンダリー KDC にコピーするか、`kdb5_util create -s` を実行してセカンダリー KDC 上にダミーデータベースおよび同一の `stash` ファイルを作成し、同じパスワードを提供します。ダミーデータベースは、データベースが最初に正常に伝達される際に上書きされます。
7. セカンダリー KDC のファイアウォールで、マスター KDC がポート 754 上の TCP (`krb5_prop`) を使用して接続し、`kprop` サービスを開始できるように許可されていることを確認します。
8. `kadmin` サービスが無効になっていることを確認します。
9. マスター KDC 上のレルムデータベースを、`kprop` コマンドが読み取るデフォルトのデータファイル (`/var/kerberos/krb5kdc/slave_datatrans`) にダンプして、手動でのデータベース伝達テストを実行します。

```
[root@masterkdc ~]# kdb5_util dump
/var/kerberos/krb5kdc/slave_datatrans
```

10. `kprop` コマンドを使用して、そのコンテンツをセカンダリー KDC に送信します。

```
[root@slavekdc ~]# kprop slavekdc.example.com
```

11. `kinit` を使用して、クライアントシステムが KDC から正常に初回認証情報を取得できることを確認します。

クライアントの `/etc/krb5.conf` は、KDC 一覧内にセカンダリー KDC のみを記載しているはずです。

```
[realms]
EXAMPLE.COM = {
    kdc = slavekdc.example.com.:88
    admin_server = kdc.example.com
    default_domain = example.com
}
```

12. レルムデータベースをダンプし、`kprop` コマンドの実行によりデータベースを各セカンダリー KDC に送信するスクリプトを作成します。このスクリプトを定期的に行うように `cron` サービスを設定します。

11.2.3. Kerberos キー配布センター (KDC) プロキシ

デプロイメントによっては、デフォルトの Kerberos ポートではなく HTTPS ポート (TCP を使用した 443) のみがアクセス可能です。クライアントは、プロキシとして IdM HTTPS サービスを使用して Kerberos 認証情報を取得できます。このリバースプロキシにより、HTTPS 経由で Kerberos で認証

されたサービスにアクセスすることができます。

Kerberos キー配布センタープロキシ (KKDCP) により、IdM にこの機能を提供します。

デプロイメントでの KKDCP の設定

IdM サーバーでは、KKDCP はデフォルトで有効になっています。

IdM クライアントでは、KKDCP を有効化する必要があります。

1. 「[Kerberos クライアントの設定](#)」の説明にあるように、`/etc/krb5.conf` ファイルを再設定します。
2. SSSD サービスを再起動します。

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

KKDCP が IdM サーバーで有効化されていることの確認

`ipaConfigString=kdcProxyEnabled` の属性と値のペアがディレクトリーに存在する場合に、KKDCP は Apache Web サーバーが起動するたびに自動的に有効化されます。このような場合には、`/etc/httpd/conf.d/ipa-kdc-proxy.conf` のシンボリックリンクが作成されます。

KKDCP 機能が有効化されているかどうかを確認するには、このシンボリックリンクが存在するかどうかを確認します。

```
$ ls -l /etc/httpd/conf.d/ipa-kdc-proxy.conf
lrwxrwxrwx. 1 root root 36 Aug 15 09:37 /etc/httpd/conf.d/ipa-kdc-proxy.conf -> /etc/ipa/kdcproxy/ipa-kdc-proxy.conf
```

IdM サーバーの KKDCP の無効化

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
```

2. IdM サーバーで `httpd` サービスを再起動します。

```
# systemctl restart httpd.service
```

その他のリソース

- Active Directory レルム向けの KKDCP の設定に関する詳細は、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。

11.3. KERBEROS クライアントの設定

Kerberos 5 クライアントの設定に必要なのは、クライアントパッケージをインストールし、各クライアントに有効な `krb5.conf` 設定ファイルを提供することです。`ssh` および `slogin` がクライアントシステムへのリモートでのログイン方法として推奨されますが、`rsh` および `rlogin` の Kerberos 対応バージョンも追加の設定変更で利用可能になります。

1. `krb5-libs` および `krb5-workstation` のパッケージをすべてのクライアントマシンにインストールします。

```
[root@server ~]# yum install krb5-workstation krb5-libs
```

2. 各クライアントに有効な **/etc/krb5.conf** ファイルを提供します。これは通常、Kerberos 配信センター (KDC) が使用する **krb5.conf** ファイルと同じ場合が多いです。例を示します。

```
[logging]
default = FILE:/var/log/krb5libs.log
kdc = FILE:/var/log/krb5kdc.log
admin_server = FILE:/var/log/kadmind.log

[libdefaults]
default_realm = EXAMPLE.COM
dns_lookup_realm = false
dns_lookup_kdc = false
ticket_lifetime = 24h
renew_lifetime = 7d
forwardable = true
allow_weak_crypto = true

[realms]
EXAMPLE.COM = {
    kdc = kdc.example.com:88
    admin_server = kdc.example.com
    default_domain = example.com
}

[domain_realm]
.example.com = EXAMPLE.COM
example.com = EXAMPLE.COM
```

環境によっては、KDC は HTTPS Kerberos 配信センタープロキシ (KKDCP) を使用する場合があります。このような場合には、以下の変更を加えてください。

1. **[realms]** セクションの **kdc** および **admin_server** オプションには、ホスト名の代わりに KKDCP の URL を割り当ててください。

```
[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
}
```

冗長化に向け、異なる KKDCP サーバーを使用して **kdc**、**admin_server**、**kpasswd_server** は複数回追加することができます。

2. IdM クライアントで **sssd** サービスを再起動して変更を適用します。

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

3. Kerberos に対応した **rsh** と **rlogin** のサービスを使用するために、**rsh** パッケージをインストールします。

4. ワークステーションで **Kerberos** を使って、**ssh**、**rsh**、または **rlogin** を使用して接続するユーザーを認証する前に、**Kerberos** データベースに独自のホストプリンシパルが必要になります。**sshd**、**kshd**、および **klogind** の各サーバープログラムは、すべてホストサービスのプリンシパルの鍵へのアクセスが必要になります。

1. **kadmin** を使ってワークステーション用のホストプリンシパルを **KDC** 上に追加します。このケースでのインスタンスは、ワークステーションのホスト名になります。**kadmin** の **addprinc** コマンドに **-randkey** オプションを使用してプリンシパルを作成し、それをランダムな鍵に割り当てます。

```
addprinc -randkey host/server.example.com
```

2. ワークステーション用の鍵は、**kadmin** をワークステーション上で実行して、**ktadd** コマンドを使用すると抽出できます。

```
ktadd -k /etc/krb5.keytab host/server.example.com
```

5. **Kerberos** に対応した他のネットワークサービスを使用するには、**krb5-server** パッケージをインストールしてサービスを起動します。**Kerberos** に対応したサービスは [表11.3「一般的な Kerberos 対応サービス」](#) に一覧表示されます。

表11.3 一般的な Kerberos 対応サービス

サービス名	使用方法
ssh	クライアントおよびサーバー両方の設定で GSSAPIAuthentication が有効になっている場合、 OpenSSH は GSS-API を使ってサーバーへのユーザー認証を行います。クライアントの GSSAPIDelegateCredentials も有効になっている場合は、ユーザーの認証情報はリモートシステムでも利用可能になります。 OpenSSH には sftp ツールも含まれます。このツールは、FTP に似たインターフェースを SFTP サーバーに提供し、 GSS-API を使用することができます。
IMAP	cyrus-sasl-gssapi パッケージもインストールされていれば、 cyrus-imap パッケージは Kerberos 5 を使用します。 cyrus-sasl-gssapi パッケージには Cyrus SASL プラグインが含まれており、これは GSS-API 認証に対応しています。 Cyrus IMAP は、 cyrus ユーザーが /etc/krb5.keytab で適切な鍵を見つけることができ、プリンシパルの root が imap (kadmin で作成) に設定されていれば、 Kerberos と正常に機能します。 cyrus-imap の代わりは dovecot パッケージ内にあり、これは Red Hat Enterprise Linux に含まれています。このパッケージには IMAP サーバーは含まれていますが、現在のところ GSS-API と Kerberos には対応していません。

11.4. スマートカード用の **KERBEROS** クライアントの設定

スマートカードは Kerberos との使用が可能です。スマートカード上で X.509 (SSL) ユーザー証明書を認識するための追加設定が必要になります。

1. 他のクライアントパッケージと共に、必要となる PKI/OpenSSL パッケージをインストールします。

```
[root@server ~]# yum install krb5-pkinit
[root@server ~]# yum install krb5-workstation krb5-libs
```

2. `/etc/krb5.conf` 設定ファイルを編集し、公開鍵インフラストラクチャー (PKI) 用のパラメーターを設定ファイルの `[realms]` セクションに追加します。`pkinit_anchors` パラメーターは CA 証明書バンドルファイルの場所を設定します。

```
[realms]
EXAMPLE.COM = {
    kdc = kdc.example.com.:88
    admin_server = kdc.example.com
    default_domain = example.com
    ...
    pkinit_anchors = FILE:/usr/local/example.com.crt
}
```

3. スマートカード認証 (`/etc/pam.d/smartcard-auth`) およびシステム認証 (`/etc/pam.d/system-auth`) 用の PAM 設定に PKI モジュール情報を追加します。両ファイルに追加する行は、以下のようになります。

```
auth    optional    pam_krb5.so use_first_pass no_subsequent_prompt
preauth_options=X509_user_identity=PKCS11:/usr/lib64/pkcs11/opensc-pkcs11.so
```

OpenSC モジュールが予想どおりに動作しない場合、`coolkey` パッケージのモジュール `/usr/lib64/pkcs11/libcoolkeypk11.so` を使用します。この場合、この問題について Red Hat テクニカルサポートへ問い合わせるか、Bugzilla に報告することを検討してください。

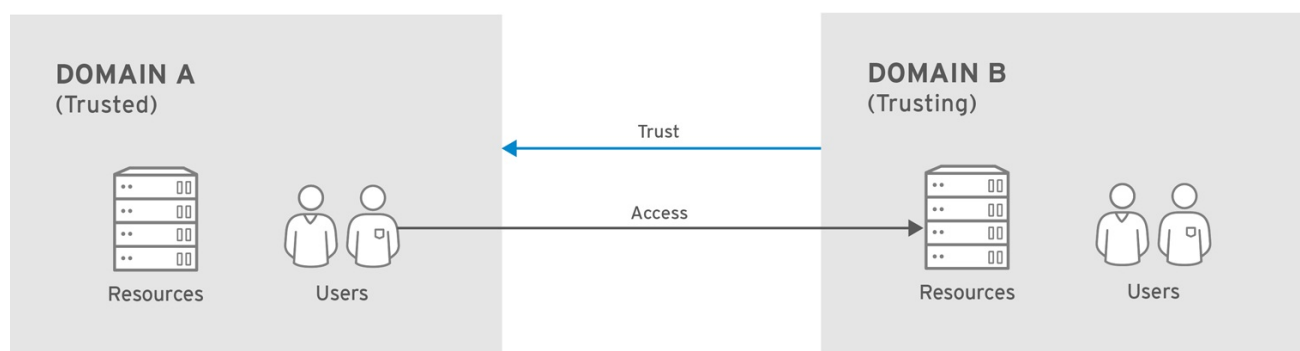
11.5. レルム間 KERBEROS 信頼の設定

Kerberos v5 レルムは、接続されているすべてのマスターおよびスレーブ上の Kerberos データベースで定義されている Kerberos プリンシパルのセットです。異なるレルム間のプリンシパルが相互に通信できるようにするには、レルム間 Kerberos 信頼を設定する必要があります。

混在環境の場合と同様に、多くの Linux 環境でシングルサインオンやアプリケーション認証、ユーザー管理用に Kerberos レルムが導入されています。この場合、Kerberos は異なるドメインや (Windows および Linux などの) 混在環境における共通の統合パスとなり得ます。Linux 環境で Identity Management のようなより構成化されたドメイン設定が使用されていない場合は、特にこれが該当します。

11.5.1. 信頼関係

信頼とは、あるレルム内のユーザーが別のドメイン内のリソースにアクセスする際にこの別のレルムに所属しているかのように 信頼されていることを指します。これは、両方のドメインで共通して保持される単一のプリンシパル用に共有鍵を作成することで実現します。



RHEL_404973_0516

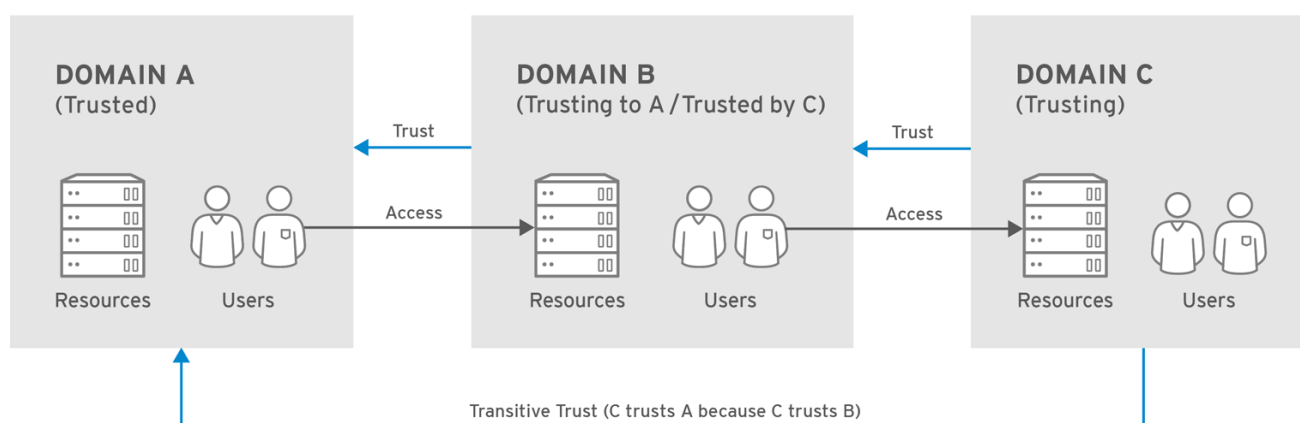
図11.2 基本的な信頼

図11.2「基本的な信頼」では、共有プリンシパルはドメイン B (`krbtgt/B.EXAMPLE.COM@A.EXAMPLE.COM`) に所属します。このプリンシパルがドメイン A にも追加されると、ドメイン A のクライアントはドメイン B 内のリソースにアクセスできるようになります。設定済みのプリンシパルは両方のレルムに存在することになります。この共有プリンシパルには以下の 3 つの特徴があります。

- 両方のレルムに存在します。
- 鍵が作成されると、両方のレルムで同じパスワードが使えます。
- キーのキーバージョン番号は同一です (`kvno`)。

デフォルトでは、レルム間信頼は一方方向です。この信頼は自動で往復しないので、`B.EXAMPLE.COM` レルムは `A.EXAMPLE.COM` レルム内のサービスに対して認証するよう信頼されています。反対方向の信頼を確立するには、両方のレルムで `krbtgt/A.EXAMPLE.COM@B.EXAMPLE.COM` サービスの鍵を共有する必要があります。これは、上記の例とは反対のマッピングを持つエントリーになります。

レルムには複数の信頼関係を確立することが可能です。信頼関係は、レルムが他のレルムを信頼することと、他のレルムが当該レルムを信頼することの両方を定義できます。Kerberos の信頼を使うと、信頼は連鎖して流れることが可能になります。たとえば、レルム A がレルム B を信頼し、レルム B がレルム C を信頼していれば、暗示的にレルム A はレルム C を信頼します。信頼がレルムに沿って流れることになります。これが **推移的** 信頼です。



RHEL_404973_0516

図11.3 推移的信頼

推移的信頼の方向は、**trust flow** (トラストフロー) と呼ばれます。トラストフローは、まずサービスが所属するレルムを認識し、次にそのサービスにアクセスするためにクライアントが連絡する必要のあるレルムを特定することで定義されます。

Kerberos プリンシパル名は、**service/hostname@REALM** という形式になります。**service** は通常、LDAP や IMAP、HTTP、**host** といったプロトコルになります。**hostname** はホストシステムの完全修飾ドメイン名になります。**REALM** は所属する Kerberos レルムです。Kerberos クライアントは通常、Kerberos レルムマッピングのホスト名または DNS ドメイン名を使用します。このマッピングは、明示的または暗黙的に指定できます。明示的なマッピングは、`/etc/krb5.conf` ファイルの `[domain_realm]` セクションを使用します。暗黙的なマッピングでは、ドメイン名は大文字に変換され、変換された名前が検索する Kerberos レルムであるとみなされます。

信頼関係をスキャンする際に、Kerberos は各レルムがルートドメインとサブドメインからなる階層的な DNS ドメインのように構成されていると仮定します。つまり、信頼は共有ルートまで移動します。ホップと呼ばれる各ステップには共有キーがあります。図11.4「同ドメイン内の信頼」では、SALES.EXAMPLE.COM は EXAMPLE.COM とキーを共有し、EXAMPLE.COM は EVERYWHERE.EXAMPLE.COM とキーを共有しています。

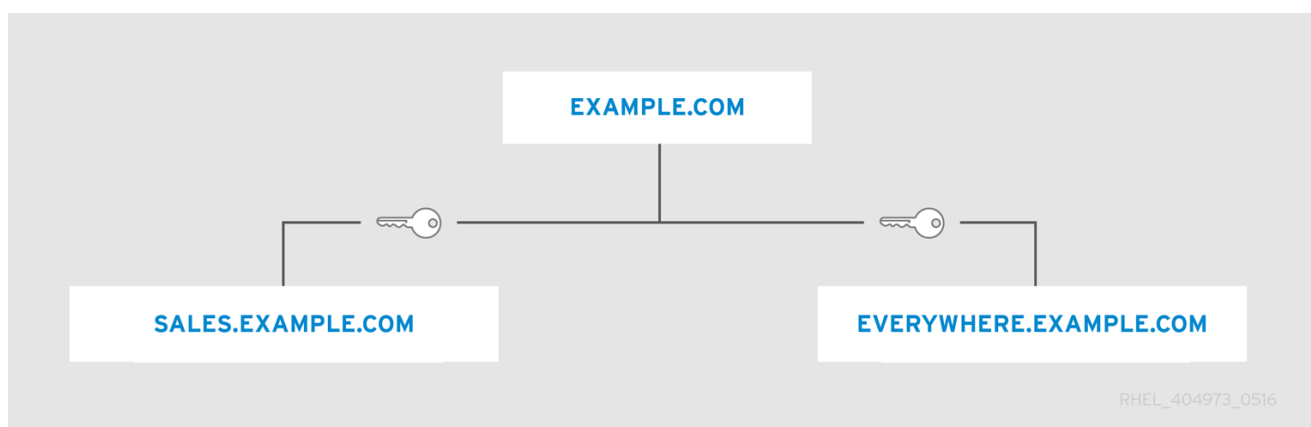


図11.4 同ドメイン内の信頼

クライアントはレルム名を DNS 名として扱い、**root** 名に達するまで自身のレルム名の要素を取り除くことで信頼パスを判断します。そして、サービスのレルムに到達するまで名前を先頭につけ始めます。

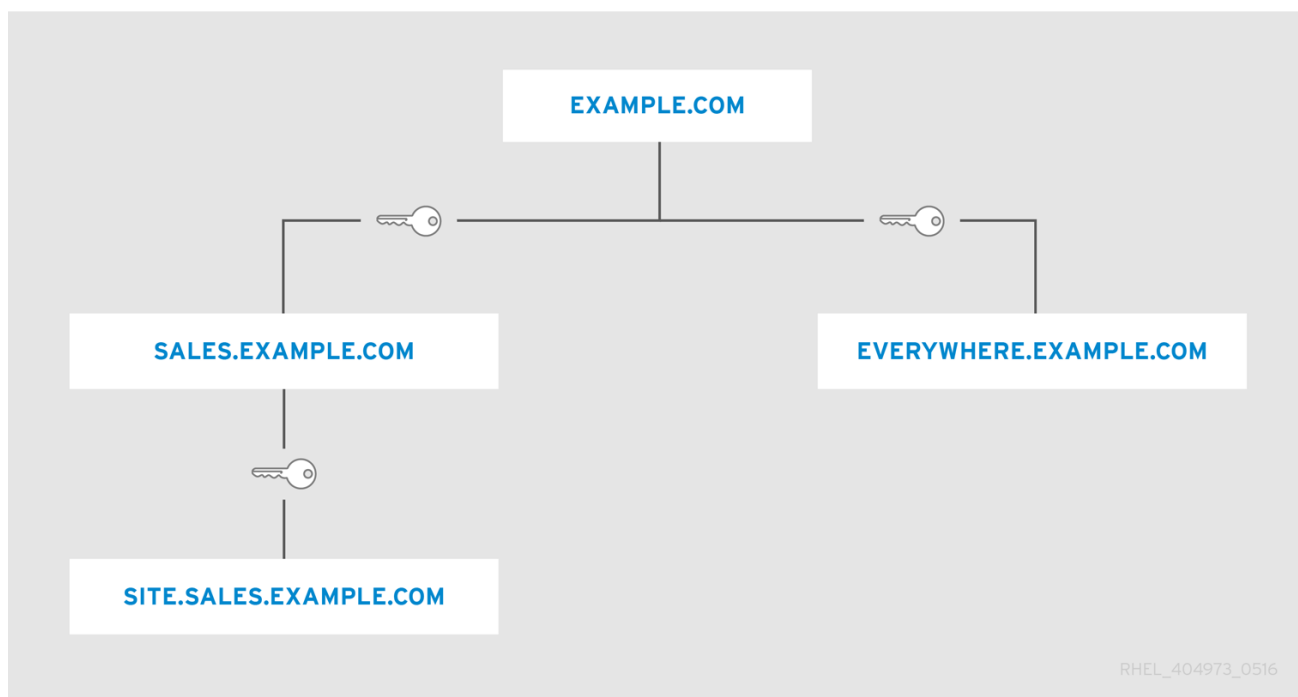


図11.5 同一ドメイン内の子/親の信頼

信頼の推移的な性質は以下のようになります。SITE.SALES.EXAMPLE.COM には SALES.EXAMPLE.COM との共有鍵が1つだけあります。しかし、小さい信頼が連続することで、SITE.SALES.EXAMPLE.COM から EVERYWHERE.EXAMPLE.COM に信頼が移動するという大きなトラストフローが可能になります。

このトラストフローは、共有鍵をドメインレベルで作成することで、完全に異なるドメイン間での行き来が可能になります。この場合、サイト間で共有される共通の接尾辞はありません。

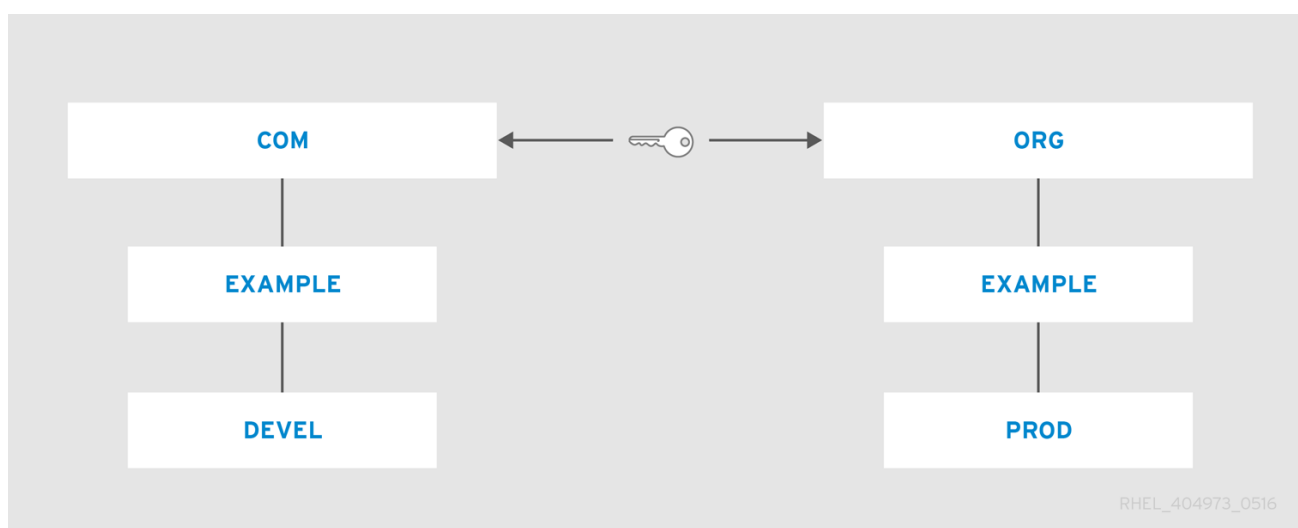


図11.6 異なるドメインでの信頼

[capaths] セクション

フローを明示的に定義することで、ホップ数を減らして非常に複雑な信頼を表示することもできます。/etc/krb5.conf ファイルの [capaths] セクションでは異なるレルム間の信頼フローを定義します。

[capaths] セクションの形式は比較的単純です。各レルムのメインエントリーがあり、ここではクライアントにはプリンシパルが含まれます。次に各レルムセクションの中には、クライアントの資格情報の取得元となる中間レルムの一覧があります。

たとえば **[capaths]** を使用して、認証情報の取得に以下のプロセスを指定することができます。

1. レルム A からの認証情報で、クライアントはレルム A の KDC から **krbtgt/A@A** チケットを取得します。このチケットを使用して、クライアントは **krbtgt/B@A** のチケットを要求します。

レルム A の KDC が発行する **krbtgt/B@A** チケットは クロスレルムの TGT (*Ticket-granting ticket*) です。これにより、クライアントはレルム B の KDC に、レルム B のサービスプリンシパルへのサービスを要求することができます。

2. **krbtgt/B@A** チケットを使用して、クライアントは **krbtgt/C@B** のクロスレルムチケットを要求します。
3. レルム B が発行する **krbtgt/C@B** チケットを使用して、クライアントは **krbtgt/D@C** クロスレルムチケットを要求します。
4. レルム C の KDC が発行する **krbtgt/D@C** チケットを使用して、クライアントはレルム D の KDC に、レルム D のサービスプリンシパルへのチケットを要求します。

このあと、認証情報キャッシュに

は、**krbtgt/A@A**、**krbtgt/B@A**、**krbtgt/C@B**、**krbtgt/D@C**、**service/hostname@D** のチケットが含まれます。**service/hostname@D** チケットを取得するには、中間のクロスレルムチケットを 3 つ取得する必要があります。

[capaths] 設定の実例を含む **[capaths]** セクションの詳細は、**krb5.conf(5)** の **man** ページを参照してください。

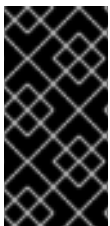
11.5.2. レルム信頼の設定

以下の例では、Kerberos レルムは **A.EXAMPLE.COM** および **B.EXAMPLE.COM** とします。

kadmin を使って **A** レルム内に **B** レルム用に共通プリンシパルのエントリーを作成します。

```
[root@server ~]# kadmin -r A.EXAMPLE.COM
kadmin: add_principal krbtgt/B.EXAMPLE.COM@A.EXAMPLE.COM
Enter password for principal "krbtgt/B.EXAMPLE.COM@A.EXAMPLE.COM":
Re-enter password for principal "krbtgt/B.EXAMPLE.COM@A.EXAMPLE.COM":
Principal "krbtgt/B.EXAMPLE.COM@A.EXAMPLE.COM" created.
quit
```

上記では、**A** レルムが **B** プリンシパルを信頼していることを示しています。



重要

レルム間プリンシパルのパスワードは、非常に強固なものにすることが推奨されます。1 日に何度も入力を求められる他の多くのパスワードとは異なり、システムはレルム間プリンシパルのパスワードを頻繁に要求しません。このため、このパスワードを簡単に記憶できるものにする必要はありません。

双方向の信頼を作成するには、反対方向に移動するプリンシパルを作成します。**kadmin** を使って **B** レルム内に **A** レルム用のプリンシパルを作成します。

```
[root@server ~]# kadmin -r B.EXAMPLE.COM
kadmin: add_principal krbtgt/B.EXAMPLE.COM@A.EXAMPLE.COM
```

```
Enter password for principal "krbtgt/B.EXAMPLE.COM@A.EXAMPLE.COM":  
Re-enter password for principal "krbtgt/B.EXAMPLE.COM@A.EXAMPLE.COM":  
Principal "krbtgt/B.EXAMPLE.COM@A.EXAMPLE.COM" created.  
quit
```

get_principal コマンドを使用して、鍵バージョン番号 (**kvno** の値) と暗号化タイプが両方のエントリーで一致するか確認します。



重要

よくある間違っただけでは、管理者が **add_principal** コマンドを **-randkey** オプションと使用して、パスワードではなくランダムな鍵を割り当て、最初のレルムのデータベースから新規エントリーをダンプし、これを 2 番目にインポートしてしまいます。これは、レルムデータベースのマスター鍵が同じものでなければ、機能しません。データベースダンプに含まれている鍵自体が、マスター鍵を使って暗号化されているためです。

[3] これは、クライアントとサーバーの両方が共通の鍵を共有し、この鍵を用いてネットワーク通信を暗号化、暗号解除するシステムです。

第12章 CERTMONGER を使った作業

マシンの認証管理には、マシン証明書の管理が含まれます。**certmonger** サービスはアプリケーションの証明書ライフサイクルを管理し、正常に設定されていれば、認証局 (CA) と共に証明書を更新、取り消すことができます。

certmonger デーモンとそのコマンドラインクライアントを使うと、公開鍵と秘密鍵のペア生成や証明書リクエストの作成、CA に対する署名のリクエスト提出といった処理を簡素化することができます。**certmonger** デーモンは証明書の有効期限を監視し、期限が切れそうになった証明書を更新することができます。**certmonger** が監視する証明書はファイルで追跡しており、このファイルは設定可能なディレクトリー内に保存します。デフォルトの場所は、**/var/lib/certmonger/requests** になります。

12.1. CERTMONGER と認証局

certmonger はデフォルトで、以下の 3 種類の証明書を自動的に取得することができます。これらは、証明書が用いるソースによって異なります。

- 自己署名証明書

自己署名証明書は証明書自体の鍵を使って署名されるので、この生成には CA は関わりません。自己署名証明書を確認するソフトウェアには、その確認で証明書を直接信頼するよう指示する必要があります。

自己署名証明書を取得するには、**selfsign-getcert** コマンドを実行します。

- Red Hat Enterprise Linux IdM の一部としての Dogtag Certificate System CA からの証明書

IdM サーバーを使用して証明書を取得するには、**ipa-getcert** コマンドを実行します。

- システム上にあるローカルの CA が署名する証明書

ローカル署名の証明書を確認するソフトウェアには、その確認でこのローカル署名者からの証明書を信頼するよう指示する必要があります。

ローカル署名の証明書を取得するには、**local-getcert** コマンドを実行します。

他の証明局 (CA) も **certmonger** を使用して証明書を管理できますが、特別な **CA ヘルパー** を作成して、**certmonger** にサポートを追加する必要があります。CA ヘルパーの作成に関する詳細情報は、**certmonger** プロジェクトのドキュメンテーションを <https://pagure.io/certmonger/blob/master/f/doc/submit.txt> で参照してください。

12.2. CERTMONGER での自己署名証明書のリクエスト

certmonger で証明書をリクエストするには、**getcert request** ユーティリティーを使用します。

証明書と鍵は **.pem** 拡張子を付けてプレーンテキストファイル内でローカルに保存されるか **NSS** データベースで保存され、証明書のニックネームで識別されます。証明書をリクエストする際には、リクエストで証明書の保存場所とニックネームを特定します。例を示します。

```
[root@server ~]# selfsign-getcert request -d /etc/pki/nssdb -n Server-Cert
```

`/etc/pki/nssdb` ファイルはグローバルの NSS データベースで、**Server-Cert** はこの証明書のニックネームになります。証明書のニックネームはこのデータベース内で一意のものである必要があります。

証明書を生成するこのコマンドで使用可能なオプションは、他の設定に加えて、リクエストしている証明書の種類や最終的な証明書で必要な設定によって異なります。

- **-r** は、鍵のペアがすでに存在している場合で、証明書の有効期限が切れそうになると、自動的に証明書を更新します。このオプションはデフォルトで使用されます。
- **-f** は、証明書を特定ファイルに保存します。
- **-k** は、鍵を特定ファイルに保存するか、鍵のファイルが存在する場合は、ファイル内の鍵を使用します。
- **-K** は、証明書を使用するサービスの Kerberos プリンシパル名を提供します。IdM サーバーからの証明書をリクエストしている場合は **-K** は必須となり、自己署名またはローカル署名の証明書をリクエストしている場合は任意となります。
- **-N** は、サブジェクト名を提供します。
- **-D** は、**subjectAltName** 値として DNS ドメイン名が証明書に含まれるようリクエストします。
- **-U** は、拡張鍵使用フラグを設定します。
- **-A** は、**subjectAltName** 値として IP アドレスが証明書に含まれるようリクエストします。
- **-I** はタスクの名前を設定します。**certmonger** はこの名前を使ってストレージの場所とリクエストオプションの組み合わせを参照します。この名前は、**getcert list** コマンドの出力にも表示されます。このオプションを指定しないと、**certmonger** は自動生成の名前を割り当てます。

IdM にあるような実際の CA は、CA 自体のポリシーにしたがって、署名リクエストで指定した **-K**、**-N**、**-D**、**-U**、**-A** などのオプションを無視することができます。たとえば、IdM では **-K** と **-N** がローカルホスト名と一致している必要があります。一方で、**selfsign-getcert** および **local-getcert** コマンドを使用して生成した証明書は、これらのコマンドがポリシーを強制しないので、指定したオプションに一致したものになります。

例12.1 サービスにおける **certmonger** の使用

```
[root@server ~]# selfsign-getcert request -f
/etc/httpd/conf/ssl.crt/server.crt -k /etc/httpd/conf/ssl.key/server.key
-N CN=`hostname` --fqdn` -D `hostname` -U id-kp-serverAuth
```

12.3. SCEP での CA 署名の証明書のリクエスト

Simple Certificate Enrollment Protocol (SCEP) は、CA での証明書管理のプロセスを自動化、簡素化します。SCEP により、クライアントの要求を行い、HTTP 経由で証明書を CA の SCEP サービスから直接取得します。このプロセスは、通常、限定された期間のみ有効なワンタイム PIN でセキュリティが確保されます。

以下の例では、SCEP CA 設定の **certmonger** への追加、新規証明書の要求、ローカルの NSS データベースへの追加を行います。

1. CA の設定を **certmonger** に追加します。

```
[root@server ~]# getcert add-scep-ca -c CA_Name -u SCEP_URL
```

- **-c:** CA 設定の必須のニックネーム。同じ値を他の **getcert** コマンドに後から渡すことができます。
- **-u:** サーバーの SCEP インターフェースへの URL
- HTTPS URL を使用する場合は任意のパラメーター
-R CA_Filename: PEM 形式を使用した SCEP サーバーの CA 証明書コピーがある場所。
 これは、HTTPS 暗号化に使用します。

2. CA 設定が正しく追加されたことを確認します。

```
[root@server ~]# getcert list-cas -c CA_Name
CA 'CA_Name':
    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
```

CA 証明書のサムプリントが SCEP 経由で取得され、コマンドの出力に表示されると、CA 設定が正しく追加されたことになります。暗号化されていない HTTP 経由でサーバーにアクセスする場合には、中間者攻撃を防止するために手動で SCEP サーバーに表示されるものと、このサムプリントを比較してください。

3. CA から証明書を要求します。

```
[root@server ~]# getcert request -I Task_Name -c CA_Name -d
/etc/pki/nssdb -n Certificate_Name -N cn="Subject Name" -L one-
time_PIN
```

- **-I:** タスクの名前。同じコマンドを **getcert list** コマンドに後から渡すことができます。
- **-c:** 要求を送信するための CA 設定
- **-d:** 証明書と鍵を保存する NSS データベースを含むディレクトリー
- **-n:** NSS データベースで使用する証明書のニックネーム
- **-N:** CSR の件名
- **-L:** CA が発行する期間限定のワンタイム PIN

4. リクエスト送信直後に、証明書が発行され、ローカルのデータベースに正しく保存されていることを確認することができます。

```
[root@server ~]# getcert list -I TaskName
Request ID 'Task_Name':
    status: MONITORING
    stuck: no
    key pair storage:
type=NSSDB,location='/etc/pki/nssdb',nickname='TestCert',token='NSS
Certificate DB'
    certificate:
type=NSSDB,location='/etc/pki/nssdb',nickname='TestCert',token='NSS
Certificate DB'
    signing request thumbprint (MD5): 503A8EDD DE2BE17E 5BAA3A57
D68C9C1B
    signing request thumbprint (SHA1): B411ECE4 D45B883A
75A6F14D 7E3037F1 D53625F4
    CA: AD-Name
    issuer: CN=windows-CA,DC=ad,DC=example,DC=com
    subject: CN=Test Certificate
    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
```

MONITORING のステータスは、発行した証明書が正しく取得されたことを示します。**getcert-list(1)** の **man** ページでは、その他の利用可能な状態と、その意味が記載されています。

12.4. NSS データベースでの証明書の保存

デフォルトでは、**certmonger** は **.pem** ファイルを使用して鍵と証明書を保存します。NSS データベースに鍵と証明書を保存するには、証明書をリクエストするコマンドで **-d** と **-n** を指定します。

- **-d** は、セキュリティーデータベースの場所を設定します。
- **-n** は、NSS データベースで使用する証明書のニックネームを指定します。



注記

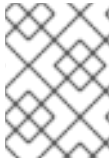
-d および **-n** のオプションは、**.pem** ファイルに提供する **-f** および **-k** オプションの代わりに使用します。

例を示します。

```
[root@server ~]# selfsign-getcert request -d /export/alias -n ServerCert
...
```

ipa-getcert および **local-getcert** を使って証明書をリクエストすると、さらに 2 つのオプションが指定できます。

- **-F** では、CA の証明書が保存されているファイルを指定します。
- **-a** では、CA の証明書が保存されている NSS データベースの場所を指定します。



注記

selfsign-getcert を使って証明書をリクエストする場合は、自己署名証明書の生成に CA が関与しないので、**-F** および **-a** オプションを指定する必要はありません。

local-getcert で **-F** または **-a** オプションのいずれか、もしくはこれら両方を指定すると、ローカル署名者が発行した証明書を確認するために必要な CA 証明書のコピーを取得することができます。例を示します。

```
[root@server ~]# local-getcert request -F /etc/httpd/conf/ssl.crt/ca.crt -
n ServerCert -f /etc/httpd/conf/ssl.crt/server.crt -k
/etc/httpd/conf/ssl.key/server.key
```

12.5. CERTMONGER を使った証明書の追跡

certmonger は、証明書の有効期限を監視し、期限終了時に証明書を更新することができます。証明書をこの方法で追跡するには、**getcert start-tracking** コマンドを実行します。



注記

getcert request の実行後に **getcert start-tracking** を実行することは、必須ではありません。**getcert request** コマンドはデフォルトで、自動的にリクエストした証明書を追跡、更新します。**getcert start-tracking** コマンドは、別のプロセスで鍵と証明書をすでに取得しており、このため手動で **certmonger** に追跡を開始するよう指示する必要がある場合に使用します。

getcert start-tracking コマンドは以下のオプションを取ります。

- **-r** は、鍵のペアがすでに存在している場合で、証明書の有効期限が切れそうになると、自動的に証明書を更新します。このオプションはデフォルトで使用されます。
- **-I** は追跡中の要求名を設定します。**certmonger** はこの名前を使ってストレージの場所とリクエストオプションの組み合わせを参照します。この名前は、**getcert list** コマンドの出力にも表示されます。このオプションを指定しないと、**certmonger** は自動生成の名前を割り当てます。

```
[root@server ~]# getcert start-tracking -I cert1-tracker -d /export/alias
-n ServerCert
```

証明書の追跡をキャンセルするには、**stop-tracking** コマンドを実行します。

第13章 アプリケーションをシングルサインオン向けに設定

ブラウザーやメールクライアントなどの一般的なアプリケーションのなかには、Kerberos チケットや SSL 証明書、トークンなどをユーザー認証の手段として使用するよう設定可能なものもあります。

これらアプリケーションの設定手順は、それぞれ異なります。本章での例 (Mozilla Thunderbird と Mozilla Firefox) では、ユーザーアプリケーションが Kerberos や他の認証情報を使用する設定方法についての考え方を提供します。

13.1. FIREFOX でシングルサインオンに KERBEROS を使用する設定

Firefox は Kerberos を使って、イントラネットのサイトや他の保護された Web サイトにシングルサインオン (SSO) を使用することができます。Firefox が Kerberos を使用するには、まず Kerberos 認証情報を適切な KDC に送信するよう設定する必要があります。

Firefox は Kerberos 認証情報を渡すように設定した後でも、有効な Kerberos チケットを必要とします。Kerberos チケットの生成には `kinit` コマンドを使用し、KDC 上のユーザーにユーザーパスワードを提供します。

```
[jsmith@host ~] $ kinit
Password for jsmith@EXAMPLE.COM:
```

Firefox で SSO (シングルサインオン) に Kerberos を使用する設定

1. Firefox のアドレスバーに **about:config** と入力して、現在の設定オプションを表示します。
2. **検索** フィールドに **negotiate** と入力して、オプション一覧を絞り込みます。
3. **network.negotiate-auth.trusted-uris** のエントリーをダブルクリックします。
4. 先頭の . (ピリオド) など認証対象のドメイン名を入力します。複数のドメインを追加する場合は、コンマ区切りのリストとして入力します。

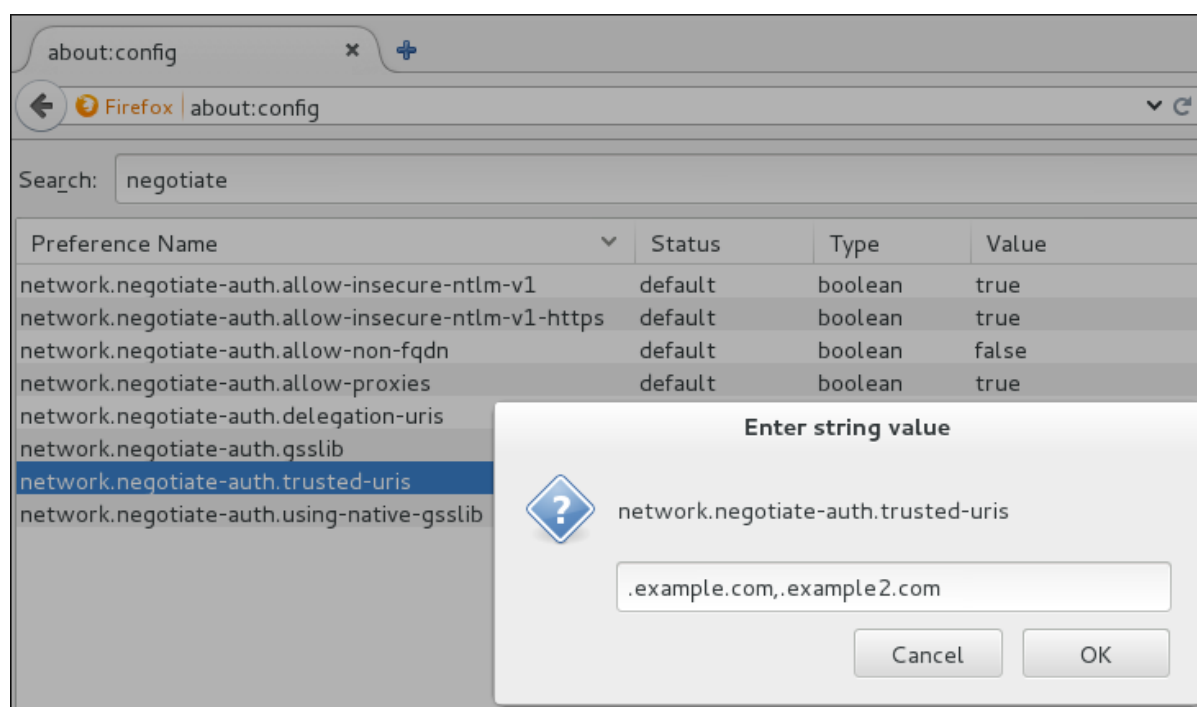
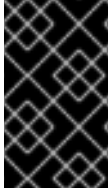


図13.1 手動での Firefox の設定



重要

Firefox の設定オプションの **network.negotiate-auth.delegation-uris** エントリーを使用して、委譲を設定することは推奨していません。これは、Kerberos に対応するサーバーすべてがユーザーとして機能できるようになるためです。



注記

Identity Management で Kerberos を使用するように Firefox を設定する方法については『[Linux ドメイン ID、認証、およびポリシーガイド](#)』の該当するセクションを参照してください。

13.2. FIREFOX での証明書管理

Firefox で証明書を管理するには、**証明書マネージャー** を開きます。

1. Mozilla Firefox で Firefox メニューを開き、**設定** をクリックします。

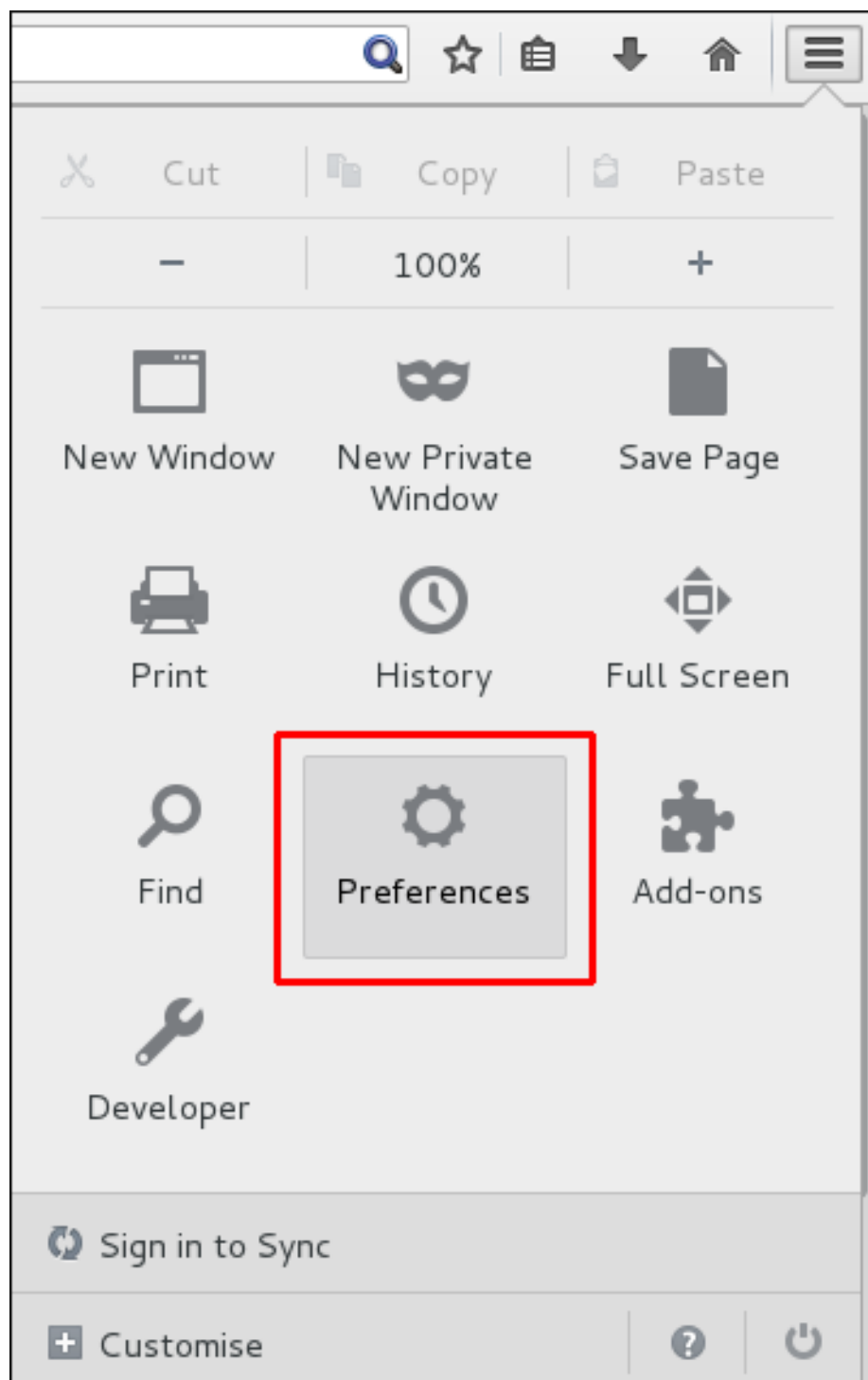


図13.2 Firefox の設定

2. 詳細 セクションを開き、**証明書** タブを選択します。

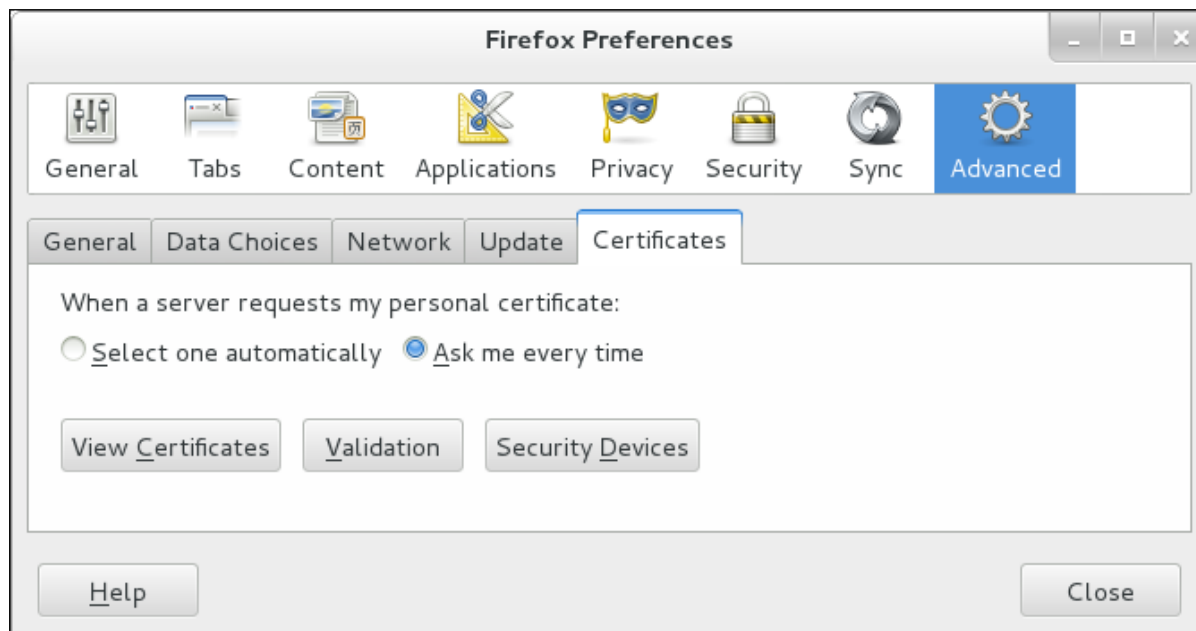


図13.3 Firefox の証明書タブ

3. **証明書を表示** をクリックして **証明書マネージャー** を開きます。

CA 証明書をインポートする手順:

1. CA 証明書をお使いのコンピューターにダウンロードして、保存します。
2. **証明書マネージャー** で、**認証局証明** のタブを選択して **インポート** をクリックします。

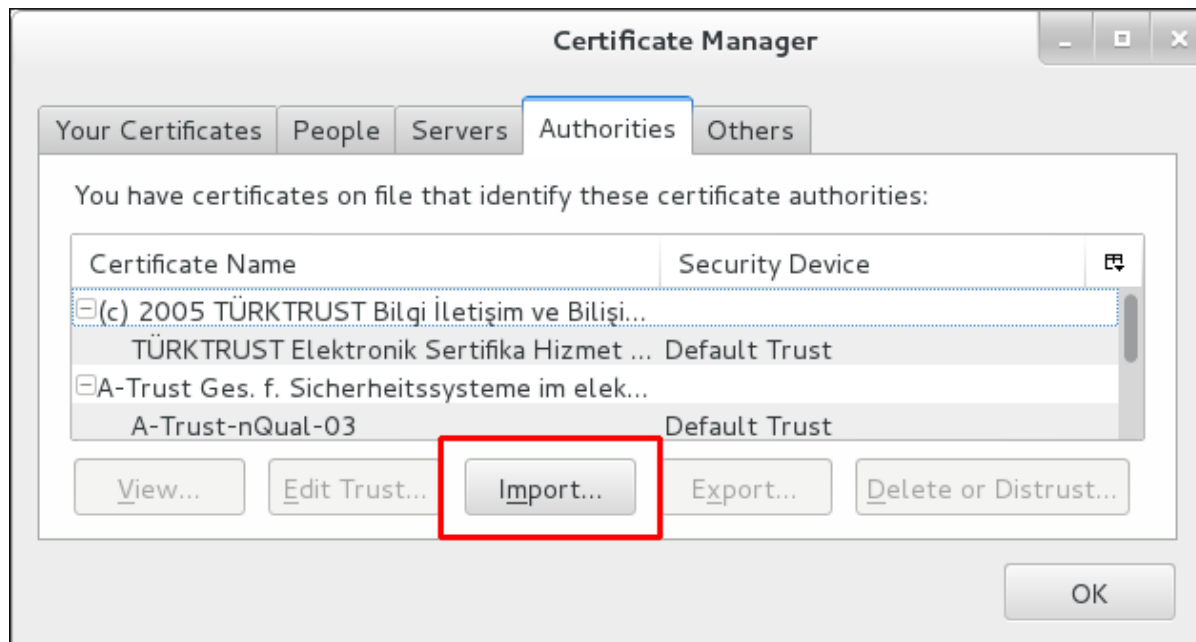


図13.4 Firefox での CA 証明書のインポート

3. ダウンロードした CA の証明書を選択します。

証明書信頼関係を設定する手順:

1. **証明書マネージャー** の **認証局証明書** タブで、適切な証明書を選択して **信頼性の設定** をクリックします。

2. 証明書の信頼性の設定を編集します。

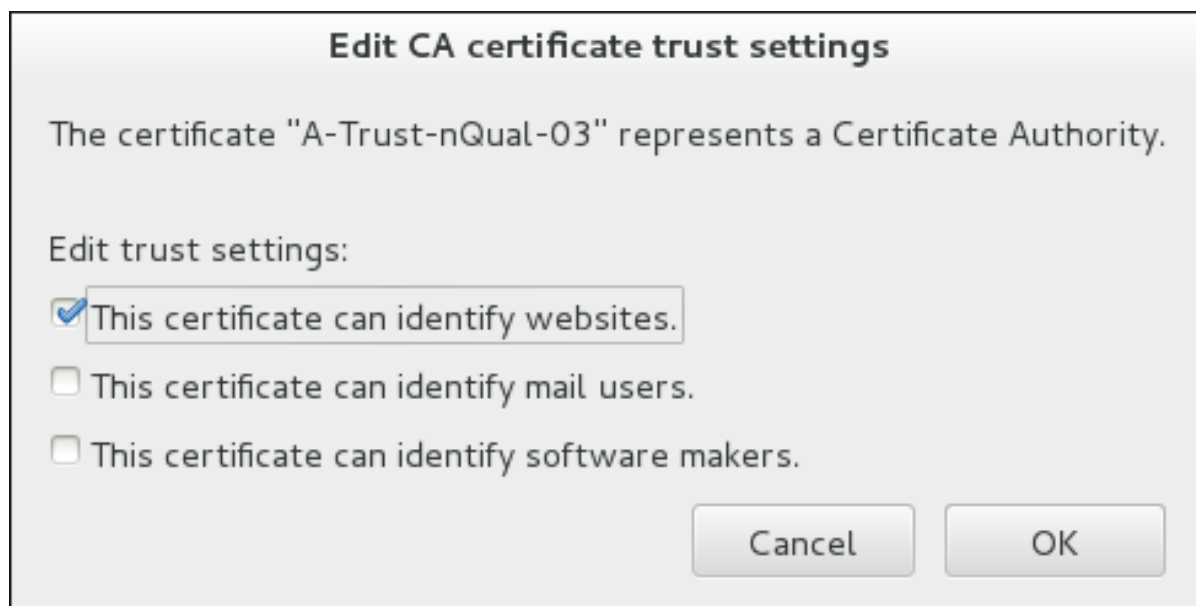


図13.5 Firefox での証明書の信頼性設定の編集

認証用の個人の証明書を使用する手順:

1. 証明書マネージャーの **あなたの証明書** タブで、**インポート** をクリックします。

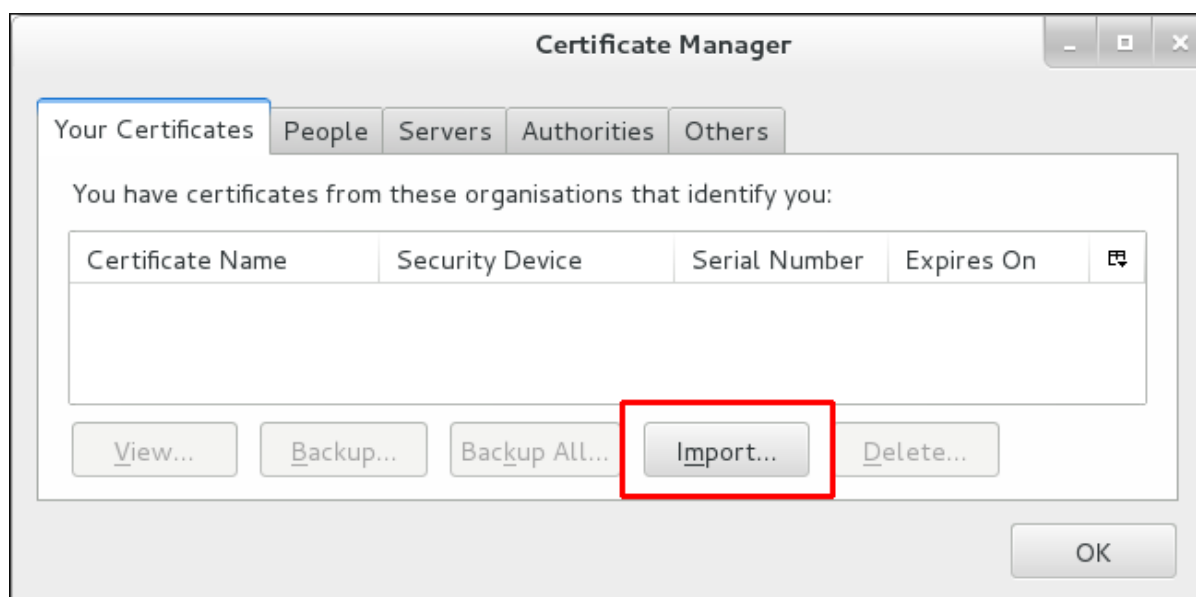


図13.6 Firefox での認証用の個人証明書のインポート

2. お使いのコンピュータで、必要な証明書を選択します。

13.3. メールクライアントの証明書管理

以下の例では、Mozilla Thunderbird のメールクライアントで証明書を管理する方法を説明します。ここでは、一般的なメールクライアントでの証明書の設定手順を説明します。

1. Mozilla Thunderbird で Thunderbird のメインメニューを開き、**設定 → アカウント設定** を選択します。

2. セキュリティー アイテムを選択し、**証明書を表示** をクリックして **証明書マネージャー** を開きます。

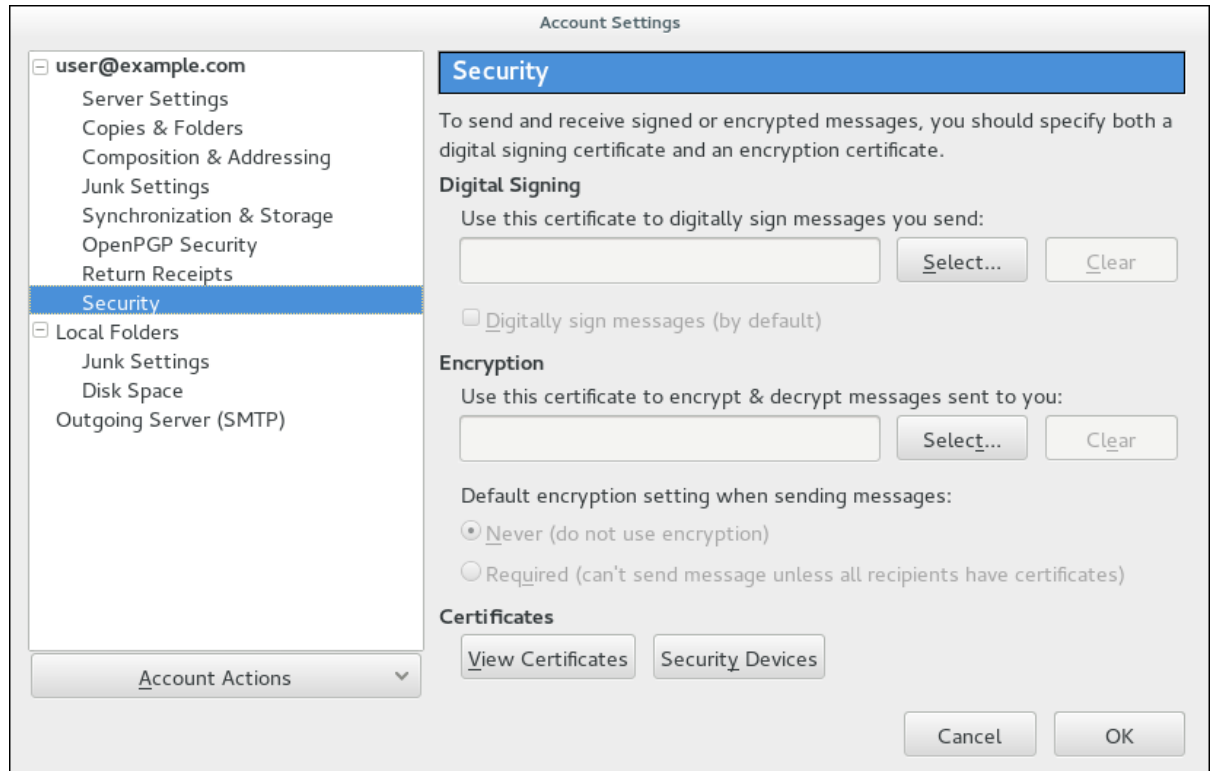


図13.7 Thunderbird でのアカウント設定

CA 証明書をインポートする手順:

1. CA 証明書をお使いのコンピューターにダウンロードして、保存します。
2. **証明書マネージャー** で、**認証局証明** のタブを選択して **インポート** をクリックします。

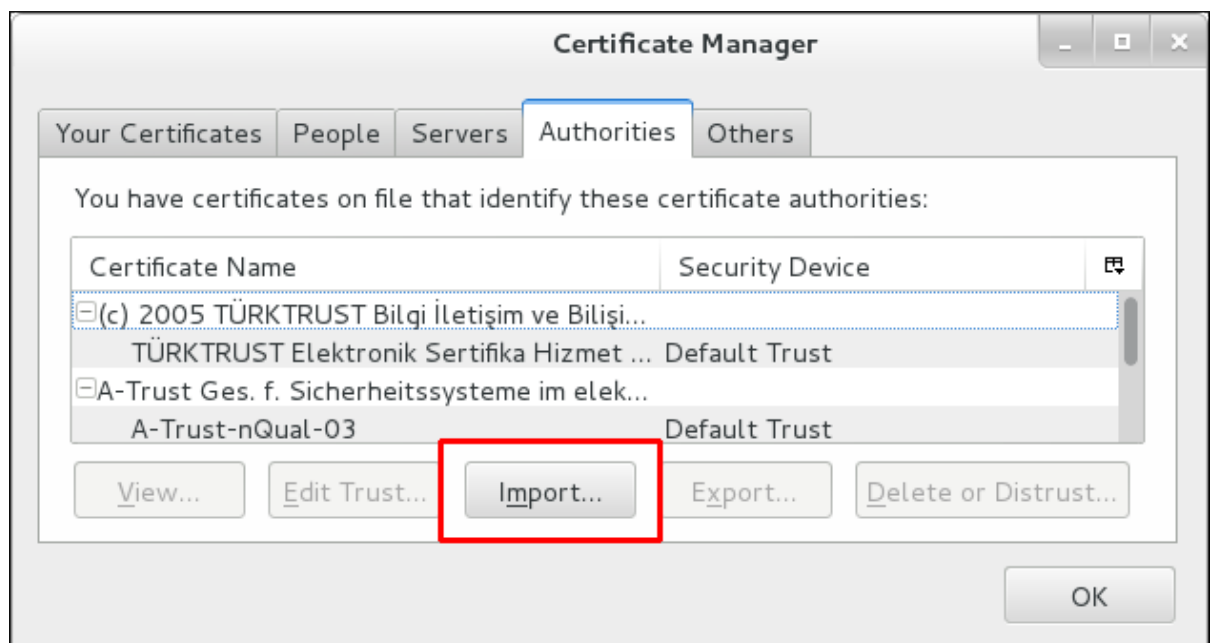


図13.8 Thunderbird での CA 証明書のインポート

3. ダウンロードした CA の証明書を選択します。

証明書信頼関係を設定する手順:

1. 証明書マネージャーの **認証局証明書** タブで、適切な証明書を選択して **信頼性の設定** をクリックします。
2. 証明書の信頼性の設定を編集します。

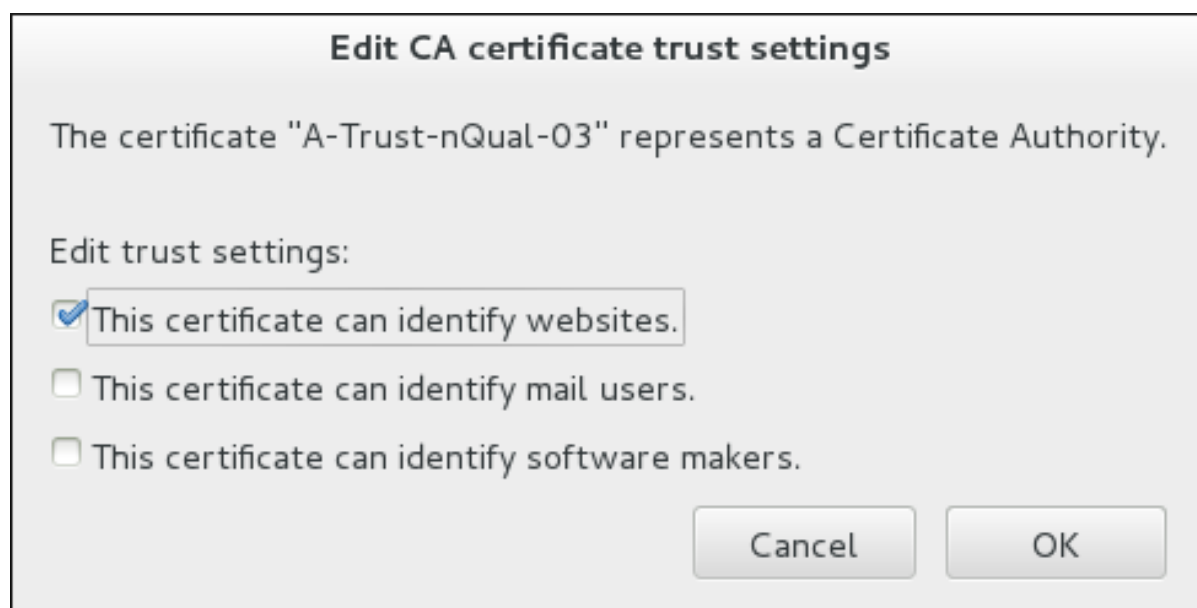


図13.9 Thunderbird での証明書の信頼性設定の編集

認証用の個人の証明書を使用する手順:

1. 証明書マネージャーの **あなたの証明書** タブで、**インポート** をクリックします。

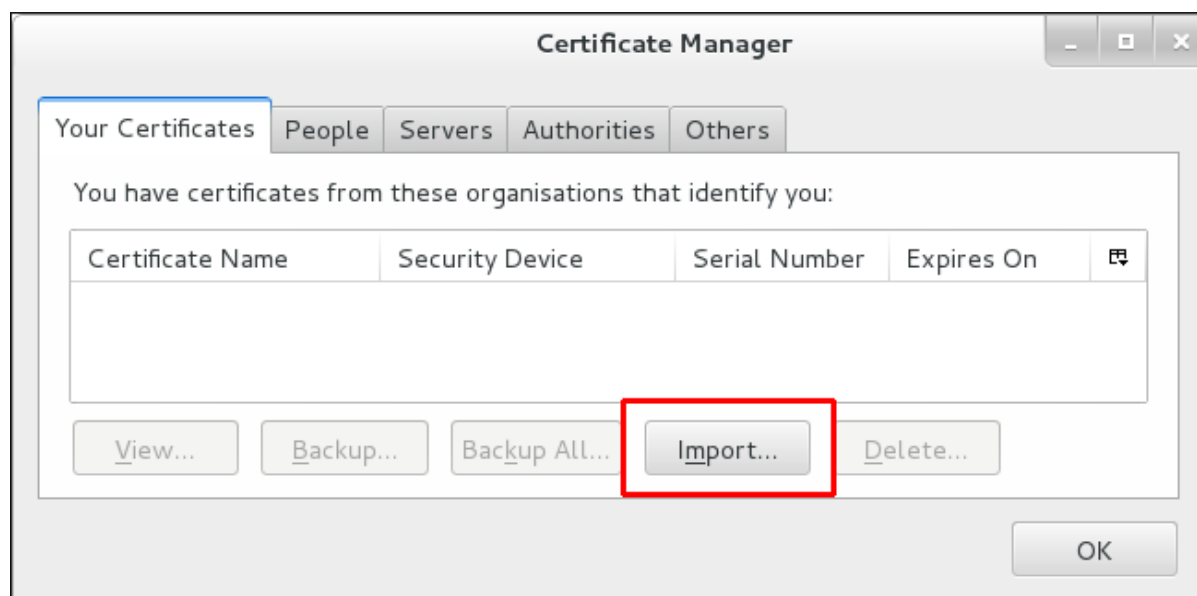


図13.10 Thunderbird での認証用の個人証明書のインポート

2. お使いのコンピュータで、必要な証明書を選択します。
3. 証明書マネージャー を閉じて、**アカウント設定** の **セキュリティ** アイテムに戻ります。
4. このフォームの **デジタル署名** セクションで、**選択** ボタンをクリックしてメッセージの署名に使用する個人証明書を選択します。

5. **暗号化** の配下で、**選択** ボタンをクリックしてメッセージの暗号化および暗号解除に使用する個人証明書を選択します。

付録A トラブルシューティング

A.1. SSSD のトラブルシューティング

- 「[SSSD ドメイン用のデバッグログの設定](#)」
- 「[SSSD ログファイルのチェック](#)」
- 「[SSSD 設定に伴う問題](#)」

A.1.1. SSSD ドメイン用のデバッグログの設定

ドメインでは、それぞれのデバッグログレベルを設定します。ログレベルを上げると、SSSD やドメイン設定での問題についての情報をより多く収集できます。

ログレベルを変更するには、追加のログを作成する **sssd.conf** ファイルで各セクションの **debug_level** パラメーターを設定します。例を示します。

```
[domain/LDAP]
cache_credentials = true
debug_level = 9
```

表A.1 デバッグログレベル

レベル	説明
0	致命的な失敗。SSSD の起動を妨げたり、SSSD の実行を停止させるもの。
1	重大な失敗。SSSD を強制終了するわけではないが、少なくとも1つのメジャーな機能が適切に動作していないことを示すエラー。
2	深刻な失敗。特定の要求や操作が失敗したことを知らせるエラー。
3	マイナーな失敗。レベル2の操作の失敗を引き起こしかねないエラー。
4	設定
5	機能データ
6	操作機能のメッセージを追跡します。
7	内部制御機能のメッセージを追跡します。
8	注意を引く可能性がある関数-内部変数のコンテンツ。

レベル	説明
9	非常に低いレベルの追跡情報。

SSSD の稼働中にデバッグレベルを変更するには、**sss_debuglevel** ユーティリティを使用します。これは、**sssd-tools** パッケージの一部です。このユーティリティの機能方法に関する情報は、**sss_debuglevel** の **man** ページを参照してください。

A.1.2. SSSD ログファイルのチェック

SSSD は、その操作に関する情報をレポートするために、**/var/log/sss/** ディレクトリーにある多くのログファイルを使用します。SSSD は各ドメイン用のログファイルだけでなく、**sssd_pam.log** ファイルと **sssd_nss.log** ファイルも作成します。

また、**/var/log/secure** ファイルには認証の失敗とその失敗の理由も記録されます。

A.1.3. SSSD 設定に伴う問題

問：SSSD が起動に失敗します。

答：SSSD はデーモンが起動する前に、すべての要求されたエントリーで設定ファイルが適切にセットアップされることが必要です。

SSSD はサービス起動前に、少なくとも1つのドメインが適切に設定されていることが必要です。このようなドメインがない場合には、SSSD の起動を試みると、ドメインが設定されていないという以下のようなエラーが返されます。

```
# sssd -d4 -i

[sssd] [ldb] (3): server_sort:Unable to register control with rootdse!
[sssd] [confdb_get_domains] (0): No domains configured, fatal error!
[sssd] [get_monitor_config] (0): No domains configured.
```

/etc/sss/sss.conf のファイルを編集して、少なくとも1つのドメインを作成します。

また SSSD は起動前に、少なくとも1つの利用可能なサービスプロバイダーが必要です。問題がサービスプロバイダーの設定にある場合、サービスが設定されていないことを示す以下のエラーメッセージが表示されます。

```
[sssd] [get_monitor_config] (0): No services configured!
```

/etc/sss/sss.conf ファイルを編集して、少なくとも1つのサービスプロバイダーを設定します。



重要

SSSD では、`/etc/sss/sss.conf` ファイル内の単一の **services** エントリー内に、サービスプロバイダーをコンマ区切りのリストで設定する必要があります。サービスが複数のエントリーに記載されている場合、SSSD が認識するのは最後のエントリーのみです。

SSSD には、`/etc/sss/sss.conf` の所有権とパーミッションが正確に設定されていることも必要です。所有権またはパーミッションが正確に設定されていない場合、SSSD の起動を試みた際に以下のようなエラーメッセージが返ってきます。

```
[sss] [confdb_ldif_from_ini_file] (0x0020): Permission check on
config file failed.
[sss] [confdb_init_db] (0x0020): Cannot convert INI to LDIF
[1]: [Operation not permitted]
[sss] [confdb_setup] (0x0010): ConfDB initialization has failed
[1]: Operation not permitted
[sss] [load_configuration] (0x0010): Unable to setup ConfDB
[1]: Operation not permitted
[sss] [main] (0x0020): Cannot read config file
/etc/sss/sss.conf. Please check that the file is accessible
only by the owner and owned by root.root.
```

正確な所有権とパーミッションを `/etc/sss/sss.conf` ファイルに設定します。

```
# chmod 600 /etc/sss/sss.conf
# chown root:root /etc/sss/sss.conf
```

問： `id` のあるグループ、または `getent group` のあるグループメンバーが見当たりません。

答： `sss.conf` の `[domain/DOMAINNAME]` セクションにある `ldap_schema` の設定が間違っている可能性があります。

SSSD は、RFC 2307 と RFC 2307bis スキーマタイプをサポートします。デフォルトでは、SSSD はより一般的な RFC 2307 スキーマを使用します。

RFC 2307 と RFC 2307bis の違いは、グループメンバーシップが LDAP サーバーに保存される方法の違いです。RFC 2307 サーバーでは、グループメンバーは複数値の **memberuid** 属性として保存され、これにはメンバーであるユーザー名が含まれます。RFC2307bis サーバーでは、グループメンバーは複数値の **member** または **uniqueMember** 属性として保存され、これにはこのグループのメンバーであるユーザーもしくはグループの DN が含まれます。RFC2307bis では、ネスト化されたグループの保持も可能になります。

グループルックアップで情報が返されない場合は、以下を実行します。

1. `ldap_schema` を `rfc2307bis` に設定します。
2. `/var/lib/sss/db/cache_DOMAINNAME.ldb` を削除します。
3. SSSD を再起動します。

これが機能しない場合は、以下の行を `sss.conf` に追加します。

```
ldap_group_name = uniqueMember
```

その後にキャッシュを削除して、SSSD を再起動します。

問： LDAP に対する認証が失敗します。

答： 認証を実行するには、SSSD では通信チャンネルの暗号化が必要になります。このため、**sssd.conf** で標準プロトコル (**ldap://**) による接続が設定されている場合、SSSD は Start TLS で通信チャンネルの暗号化を試みます。**sssd.conf** でセキュアなプロトコル (**ldaps://**) による接続が設定されている場合は、SSSD は SSL を使用します。

つまり、LDAP サーバーは SSL または TLS で実行するように設定される必要があります。TLS は標準 LDAP ポート (389) で、SSL はセキュアな LDAPS ポート (636) で有効にする必要があります。SSL と TLS のいずれの場合も、LDAP サーバーは有効な証明書信頼で設定されている必要があります。

無効な証明書信頼は、LDAP に対する認証における最も一般的な問題の1つです。クライアントに LDAP サーバー証明書の適切な信頼がないと、接続を確認できず、SSSD はパスワードの送信を拒否します。LDAP プロトコルでは、パスワードがプレーンテキストで LDAP サーバーに送信されることが要求されます。暗号化されていない接続でパスワードをプレーンテキストで送信することは、セキュリティ上の問題となります。

証明書が信頼されないと、**syslog** メッセージが書き込まれ、TLS 暗号化が開始できません。証明書の設定は、LDAP サーバーが SSSD 以外からアクセス可能かどうかをチェックすることでテストできます。以下の例では、TLS 接続で **test.example.com** への匿名バインドをテストします。

```
$ ldapsearch -x -ZZ -h test.example.com -b dc=example,dc=com
```

証明書信頼が適切に設定されていない場合、以下のエラーが出てテストは失敗します。

```
ldap_start_tls: Connect error (-11) additional info: TLS error -
8179:Unknown code ____f 13
```

証明書を信頼するには、以下を実行します。

1. 認証機関が LDAP サーバー証明書の署名に使用する公開 CA 証明書のコピーを取得して、ローカルシステムに保存します。
2. ファイルシステム上の CA 証明書に向ける以下の行を **sssd.conf** ファイルに追加します。

```
ldap_tls_cacert = /path/to/cacert
```

3. LDAP サーバーが自己署名証明書を使用している場合は、**sssd.conf** ファイルから **ldap_tls_reqcert** の行を削除します。

このパラメーターは、SSSD が認証局による証明書を信頼するように指示しますが、これは自己署名型の CA 証明書ではセキュリティリスクとなります。

問： 非標準ポートでの LDAP サーバーへの接続が失敗します。

答： enforcing モードで SELinux を実行する場合、クライアントの SELinux ポリシーは非標準ポート

でLDAPサーバーに接続するように修正する必要があります。以下に例を挙げます。

```
# semanage port -a -t ldap_port_t -p tcp 1389
```

問：NSSがユーザー情報提供に失敗します。

答：これは通常、SSSDがNSSサービスに接続できないことを意味します。

NSSサービスが稼働していることを確認します。

```
# service sssd status
Redirecting to /bin/systemctl status sssd.service
sssd.service - System Security Services Daemon
   Loaded: loaded (/usr/lib/systemd/system/sss.service;
   enabled)
   Active: active (running) since Wed 2015-01-14 10:17:26 CET;
   1min 30s ago
     Process: 683 ExecStart=/usr/sbin/sss -D -f (code=exited,
   status=0/SUCCESS)
    Main PID: 745 (sss)
       CGroup: /system.slice/sss.service
               └─745 /usr/sbin/sss -D -f
                 └─746 /usr/libexec/sss/sss_be --domain default --debug-
   to-files...
                   └─804 /usr/libexec/sss/sss_nss --debug-to-files
                     └─805 /usr/libexec/sss/sss_pam --debug-to-files
```

SSSDが**Active: active (running)**状態で、かつ出力に**sss_nss**が含まれていれば、NSSサービスは稼働しています。

NSSが稼働している場合は、プロバイダーが**/etc/sss/sss.conf**ファイル内の**[nss]**セクションで正しく設定されていることを確認してください。特に、**filter_users**属性と**filter_groups**属性をチェックしてください。

SSSDが使用するサービスのリスト内にNSSが含まれていることを確認します。

/etc/nsswitch.confファイル内の設定を確認します。詳細は、「[NSSサービスを設定してSSSDを使用](#)」を参照してください。

問：NSSが誤ったユーザー情報を返します。

答：検索で間違ったユーザー情報が返された場合には、別のドメインで競合するユーザー名がないかをチェックしてください。複数のドメインを使用している場合には、**/etc/sss/sss.conf**ファイル内の**use_fully_qualified_domains**属性を**true**に設定します。これで、異なるドメインで同じ名前を持つ異なるユーザーが区別されます。

問：ローカルのSSSDユーザー用のパスワード設定で、パスワードの入力が2回要求されます。

答：ローカルのSSSDユーザーのパスワードの変更を試みる際には、パスワードが2回要求されることがあります。

```
[root@clientF11 tmp]# passwd user1000
```



```
Changing password for user user1000.
New password:
Retype new password:
New Password:
Reenter new Password:
passwd: all authentication tokens updated successfully.
```

この原因は、PAM 設定が間違っているためです。`/etc/pam.d/system-auth` ファイル内の `use_authtok` オプションが正しく設定されていることを確認してください。正しい設定例については、「[サービスの設定: PAM](#)」を参照してください。

問： **Active Directory** アイデンティティプロバイダーは `sssd.conf` ファイルで適切に設定されているのに、**SSSD** は接続に失敗し、**GSS-API** エラーがでます。

答： **SSSD** は、ホスト名を使用しないと **Active Directory** プロバイダーに接続できません。ホスト名が提供されないと、**SSSD** クライアントはホストへの IP アドレスが解決できず、認証に失敗します。

たとえば、以下の設定の場合、

```
[domain/ADEXAMPLE]
debug_level = 0xFFFF0
id_provider = ad
ad_server = 172.16.0.1
ad_domain = example.com
krb5_canonicalize = False
```

SSSD クライアントは以下の **GSS-API** エラーを返して、認証要求が失敗します。

```
(Fri Jul 27 18:27:44 2012) [sssd[be[ADTEST]]] [sasl_bind_send]
(0x0020): ldap_sasl_bind failed (-2)[Local error]
(Fri Jul 27 18:27:44 2012) [sssd[be[ADTEST]]] [sasl_bind_send]
(0x0080): Extended failure message: [SASL(-1): generic failure: GSSAPI
Error: Unspecified GSS failure. Minor code may provide more
information (Cannot determine realm for numeric host address)]
```

このエラーを避けるには、`ad_server` を **Active Directory** ホストの名前に設定するか、「[DNS Service Discovery の設定](#)」にあるように `_srv_` キーワードを **DNS** サービスディスカバリーに使用します。

問： **SSSD** を中央認証に設定しましたが、**Firefox** や **Adobe** などいくつかのアプリケーションが起動しません。

答： 64 ビットシステム上でも、32 ビットのアプリケーションはパスワードや ID キャッシュへのアクセスに 32 ビットバージョンの **SSSD** を必要とします。32 ビットバージョンの **SSSD** が利用できない場合でも、システムは **SSSD** キャッシュを使うように設定されており、したがって 32 ビットのアプリケーションは起動に失敗します。

たとえば、**Firefox** はパーミッション拒否のエラーで失敗します。

```
Failed to contact configuration server. See
http://www.gnome.org/projects/gconf/
for information. (Details - 1: IOR file '/tmp/gconfd-
```

```
somebody/lock/ior'
not opened successfully, no gconfd located: Permission denied 2: IOR
file '/tmp/gconfd-somebody/lock/ior' not opened successfully, no
gconfd
located: Permission denied)
```

Adobe Reader の場合、以下のエラーでは現行のシステムユーザーが認識されていないことを示しています。

```
[jsmith@server ~]$ acroread
(acroread:12739): GLib-WARNING **: getpwuid_r(): failed due to unknown
user id (366)
```

他のアプリケーションでも、同様のユーザーもしくはパーミッションエラーが表示される場合があります。

問：SSSD は、削除した自動マウントの場所を示しています。

答：自動マウントの場所の SSSD キャッシュは、その場所が後で変更されたり削除されたりしても、消えずに残ります。SSSD の autofs 情報を更新するには、以下を実行します。

1. 「UID または GID の変更後、ユーザーのログインは不可」の説明にあるように、autofs キャッシュを削除します。
2. SSSD を再起動します。

```
# systemctl restart sssd
```

A.1.4. UID または GID の変更後、ユーザーのログインは不可

ユーザーまたはグループ ID の変更後、SSSD はユーザーのログインを阻止します。

エラー内容:

SSSD は、ユーザー ID (UID) およびグループ ID (GID) ベースでユーザーとグループを認識します。既存のユーザーまたはグループが UID または GID を変更すると、SSSD はそのユーザーまたはグループを認識しません。

解決方法:

sss_cache ユーティリティを使用して SSSD キャッシュを削除します。

1. sssd-tools がインストールされていることを確認します。
2. 特定のユーザーの SSSD キャッシュを削除し、残りのキャッシュのレコードをそのまま残しておくには、以下を実行します。

```
# sss_cache --user user_name
```

ドメイン全体のキャッシュを削除するには、以下を実行します。

```
# sss_cache --domain domain_name
```

ユーティリティーは、SSSD キャッシュにあるユーザーやグループ、ドメインのレコードを無効にします。続いて、SSSD はアイデンティティプロバイダーからレコードを取得し、キャッシュをリフレッシュします。

sss_cacheの詳細は、**sss_cache(8)**の man ページを参照してください。

A.1.5. SSSD コントロールとステータスユーティリティー

SSSD には、ステータス情報の取得、トラブルシューティング時のデータファイルやその他の SSSD 関連タスクの管理を行う **sssctl** ユーティリティーが含まれます。

1. **sssctl** を使用するには、**sssd-tools** パッケージをインストールします。

```
[root@server ~]# yum install sssd-tools
```

2. **sssctl** のオプションは、SSSD InfoPipe 応答を使用します。これを有効化するには **/etc/sss/sss.conf** の **services** オプションに **ifp** を追加します。

```
[sss]
services = nss, sudo, pam, ssh, ifp
```

3. SSSD を再起動します。

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

A.1.5.1. SSSD 設定の検証

sssctl config-check コマンドは、SSSD 設定ファイルの静的な解析を実行します。これにより、**/etc/sss/sss.conf** ファイルと **/etc/sss/conf.d/*** ファイルを検証して、SSSD を再起動する前にレポートを取得することができます。

このコマンドは、SSSD 設定ファイルに以下のチェックを実行します。

パーミッション

所有者およびグループの所有者は **root:root** に、パーミッションは **600** に設定する必要があります。

ファイル名

/etc/sss/conf.d/ のファイル名は **.conf** というサフィックスを使用する必要があり、.(ピリオド)で開始できないものとします。

誤字誤植

セクションとオプション名の誤字誤植がチェックされます。値はチェックされない点に注意してください。

オプション

オプションは、正しいセクションに配置するようにしてください。

設定を検証するには、以下を実行します。

```
# sssctl config-check
Issues identified by validators: 3
[rule/allowed_sections]: Section [paM] is not allowed. Check for typos.
[rule/allowed_domain_options]: Attribute 'offline_timeoutX' is not allowed
in section 'domain/idm.example.com'. Check for typos.
[rule/allowed_sudo_options]: Attribute 'homedir_substring' is not allowed
in section 'sudo'. Check for typos.

Messages generated during configuration merging: 2
File /etc/sss/conf.d/wrong-file-permissions.conf did not pass access
check. Skipping.
File configuration.conf.disabled did not match provided patterns.
Skipping.

Used configuration snippet files: 1
/etc/sss/conf.d/sample.conf
```

A.1.5.2. ドメイン情報

ドメインのステータスにより、SSSD がアクセスするドメイン一覧を表示し、ステータスを取得することができます。

1. 信頼済みのドメインなど、SSSD 内で利用可能なドメインをすべて表示します。

```
[root@server ~]# sssctl domain-list
idm.example.com
ad.example.com
```

2. *idm.example.com* ドメインのステータスを取得します。

```
[root@server ~]# sssctl domain-status idm.example.com
Online status: Online
```

A.1.5.3. キャッシュされたエントリーの情報

sssctl コマンドにより、キャッシュされたエントリーの情報を取得して、キャッシュ関連の認証問題を調査、デバッグすることができます。

ユーザーアカウント **admin** のキャッシュ情報を照会するには、以下を実行します。

```
[root@server ~]# sssctl user-show admin
Name: admin
Cache entry creation date: 07/10/16 16:09:18
Cache entry last update time: 07/14/16 10:13:32
Cache entry expiration time: 07/14/16 11:43:32
Initgroups expiration time: Expired
Cached in InfoPipe: No
```

グループまたはネットグループのキャッシュ情報を照会するには、以下を使用します。

```
[root@server ~]# sssctl group-show groupname
[root@server ~]# sssctl netgroup-show netgroupname
```

A.1.5.4. ログファイルの省略

問題をデバッグする場合には **sssctl logs-remove** を使用して **/var/log/sss/** ディレクトリーのすべての SSSD ログファイルを省略して、新たに作成されたエントリーのみを取得することができます。

```
[root@server ~]# sssctl logs-remove
Truncating log files...
```

A.1.5.5. SSSD キャッシュの削除

SSSD キャッシュのデータベースを削除するには **sssctl** コマンドを使用して **remove-cache** オプションを指定します。データベースを削除する前に、このコマンドにより自動的にバックアップが作成されます。

以下のコマンドを実行して、すべてのローカルデータをバックアップし、SSSD キャッシュを削除します。

```
[root@server ~]# sssctl cache-remove
SSSD must not be running. Stop SSSD now? (yes/no) [yes]
Creating backup of local data...
Removing cache files...
SSSD needs to be running. Start SSSD now? (yes/no) [yes]
```



注記

このバックアップでは、**/var/lib/sss/backup/** ディレクトリーに、ローカルの上書きなどのローカルデータのみを保存します。

バックエンドのコンテンツを自動的にインポートするには **sssctl restore-local-data** を実行します。

A.1.5.6. LDAP グループの情報取得には長時間必要

LDAP グループについての情報をルックアップする操作は、非常に時間がかかります。大規模なグループやネスト化されたグループの場合は、特に時間がかかります。

エラー内容:

デフォルトでは、LDAP グループについての情報をルックアップしている際には、そのグループのすべてのメンバーが返されます。大規模なグループやネスト化されたグループの操作の場合、すべてのメンバーを返すため、処理に時間がかかります。

解決方法:

グループのルックアップに返されたメンバーシップ一覧は、ユーザーがグループに所属しているかどうかを評価する際には使用されません。パフォーマンスの向上、特にアイデンティティルックアップのパフォーマンスの向上には、グループメンバーシップのルックアップを無効にします。

1. **/etc/sss/sss.conf** ファイルを開きます。
2. **[domain]** セクションで、**ignore_group_members** オプションを **true** に設定します。

```
[domain/domain_name]
[... file truncated ...]
ignore_group_members = true
```



A.2. SSSD での SUDO のトラブルシューティングと SUDO のデバッグログ

A.2.1. SSSD と sudo デバッグロギング

デバッグロギング機能では、SSSD および sudo に関する追加情報をロギングできます。

sudo デバッグログファイル

sudo デバッグを有効にするには以下を行います。

1. 以下の行を `/etc/sudo.conf` に追加します。

```
Debug sudo /var/log/sudo_debug.log all@debug
Debug sudoers.so /var/log/sudo_debug.log all@debug
```

2. デバッグするユーザーとして **sudo** コマンドを実行します。

`/var/log/sudo_debug.log` ファイルが自動的に作成され、以下のような質問に回答するための詳細情報が提供されます。

- **sudo** コマンドの実行時に、ユーザーおよび環境に関してどのような情報が入手できますか？

```
sudo[22259] settings: debug_flags=all@debug
sudo[22259] settings: run_shell=true
sudo[22259] settings: progname=sudo
sudo[22259] settings: network_addrs=192.0.2.1/255.255.255.0
fe80::250:56ff:feb9:7d6/ffff:ffff:ffff:ffff::
sudo[22259] user_info: user=user_name
sudo[22259] user_info: pid=22259
sudo[22259] user_info: ppid=22172
sudo[22259] user_info: pgid=22259
sudo[22259] user_info: tcpgid=22259
sudo[22259] user_info: sid=22172
sudo[22259] user_info: uid=10000
sudo[22259] user_info: euid=0
sudo[22259] user_info: gid=554801393
sudo[22259] user_info: egid=554801393
sudo[22259] user_info:
groups=498,6004,6005,7001,106501,554800513,554801107,554801108,55480
1393,554801503,554802131,554802244,554807670
sudo[22259] user_info: cwd=/
sudo[22259] user_info: tty=/dev/pts/1
sudo[22259] user_info: host=client
sudo[22259] user_info: lines=31
sudo[22259] user_info: cols=237
```

- **sudo** ルールの取得にはどのデータソースを使用しますか？

```
sudo[22259] <- sudo_parseIn @ ./fileops.c:178 := sudoers: files sss
```

- SSSD プラグインは以下の行で開始します。

```
sudo[22259] <- sudo_sss_open @ ./sssd.c:305 := 0
```

- SSSD が返したルールはいくつですか？

```
sudo[22259] Received 3 rule(s)
```

- ルールは一致しましたか？

```
sudo[22259] sssd/ldap sudoHost 'ALL' ... MATCH!
sudo[22259] <- user_in_group @ ./pwutil.c:1010 := false
```

SSSD デバッグログファイル

SSSD デバッグを有効化するには以下を実行します。

1. `/etc/sss/sss.conf` ファイルの `[sudo]` と `[domain/domain_name]` セクションに `debug_level` オプションを追加します。

```
[domain/domain_name]
debug_level = 0x3ff0
...
[sudo]
debug_level = 0x3ff0
```

2. SSSD を再起動します。

```
# systemctl restart sssd
```

3. `sudo` コマンドを実行してログファイルにデバッグ情報を記述します。

以下のログファイルが作成されます。

ドメインログファイル: `/var/log/sss/sss_domain_name.log`

このログファイルは、以下のような質問に回答する際に役立ちます。

- SSSD が返したルールはいくつですか？

```
[sdap_sudo_refresh_load_done] (0x0400): Received 4-rules rules
```

- SSSD はサーバーからどの `sudo` ルールをダウンロードしましたか？

```
[sss[be[LDAP.PB]]] [sysdb_save_sudorule] (0x0400): Adding sudo
rule demo-name
```

- 一致したルールはキャッシュに保存されますか？

```
[sdap_sudo_refresh_load_done] (0x0400): Sudoers is successfully
stored in cache
```

- サーバーからルールをダウンロードする際に使用したフィルターは何ですか？

```
[sdap_get_generic_ext_step] (0x0400): calling ldap_search_ext with
[(&(objectClass=sudoRole)(!(sudoHost=*)))(sudoHost=ALL)]
```

```
(sudoHost=client.example.com)(sudoHost=client)(sudoHost=192.0.2.1)
(sudoHost=192.0.2.0/24)
(sudoHost=2620:52:0:224e:21a:4aff:fe23:1394)
(sudoHost=2620:52:0:224e::/64)(sudoHost=fe80::21a:4aff:fe23:1394)
(sudoHost=fe80::/64)(sudoHost=+*)(|(sudoHost=*\\*)(sudoHost=?*)
(sudoHost=*\\2A*)(sudoHost=[*]*)))] [dc=example,dc=com]
```

以下のフィルターを使用して IdM のルールを検索します。

```
# ldapsearch -x -D "cn=Directory Manager" -W -H
ldap://server.example.com -b dc=example,dc=com '(&
(objectClass=sudoRole)...)'
```

sudo 応答ログファイル: /var/log/sss/sudo.log

このログファイルは、以下のような質問に回答する際に役立ちます。

- SSSD が返したルールはいくつですか？

```
[sss[sudo]] [sudosrv_get_sudorules_from_cache] (0x0400):
Returning 4-rules rules for [user@idm.example.com]
```

- SSSD のキャッシュの検索に適用されたフィルターはどれですか？

```
[sudosrv_get_sudorules_query_cache] (0x0200): Searching sysdb with
[(&(objectClass=sudoRule)(|(sudoUser=ALL)(sudoUser=user)
(sudoUser=#10001)(sudoUser=%group-1)(sudoUser=%user)
(sudoUser=+*)))]
```

- SSSD キャッシュから返されたルールをどのように検索しますか？以下のフィルターを使用してルールを検索します。

```
# ldbsearch -H /var/lib/sss/db/cache_domain_name.ldb -b cn=sysdb
'(&(objectClass=sudoRule)...)'
```



注記

ldbsearch ユーティリティは **ldb-tools** パッケージに含まれます。

A.3. FIREFOX の KERBEROS 設定のトラブルシューティング

Kerberos 認証が機能しない場合は、認証プロセスにおける詳細ロギングをオンにします。

1. Firefox のすべてのインスタンスを閉じます。
2. コマンドプロンプトで、**NSPR_LOG_*** 変数の値をエクスポートします。

```
export NSPR_LOG_MODULES=negotiateauth:5
export NSPR_LOG_FILE=/tmp/moz.log
```

3. そのシェルから Firefox を再起動し、Kerberos 認証に失敗していた Web サイトを開きます。

4. `/tmp/moz.log` ファイルにあるエラーメッセージで `nsNegotiateAuth` があるものを確認します。

Kerberos 認証では、以下のようなエラーが一般的です。

認証情報が見つかりません

```
-1208550944[90039d0]: entering nsNegotiateAuth::GetNextToken()  
-1208550944[90039d0]: gss_init_sec_context() failed: Miscellaneous  
failure  
No credentials cache found
```

つまり、Kerberos チケットがなかったことを意味します (有効期限が切れたか、生成されていなかったため)。これを解決するには、`kinit` を実行して Kerberos チケットを生成し、Web サイトを再度開きます。

Kerberos データベースにサーバーが見つかりません

```
-1208994096[8d683d8]: entering nsAuthGSSAPI::GetNextToken()  
-1208994096[8d683d8]: gss_init_sec_context() failed: Miscellaneous  
failure  
Server not found in Kerberos database
```

これは、ブラウザーが KDC に問い合わせができないという意味です。`/etc/krb5.conf` ファイルの `[domain_realm]` セクションで、正しいエントリーでドメインを指定する必要があります。例を示します。

```
.example.com = EXAMPLE.COM  
example.com = EXAMPLE.COM
```

ログにエラーがありません

HTTP プロキシサーバーが Kerberos 認証に必要な HTTP ヘッダーを削除している可能性があります。HTTPS を使用してサイトに接続することで、要求を変更せずに渡すことができます。

付録B 改訂履歴

改訂 7.0-26.1 翻訳ファイルを XML ソースバージョン 7.0-26 と同期	Mon May 28 2018	Ludek Janda
改訂 7.0-26 7.5 GA 公開用ドキュメントの準備	Tue Apr 10 2018	Filip Hanzelka
改訂 7.0-25 マイナーな更新	Mon Mar 19 2018	Lucie Maňásková
改訂 7.0-24 マイナーな修正および更新	Mon Feb 12 2018	Aneta Štefllová Petrová
改訂 7.0-23 マイナーな修正	Mon Jan 29 2018	Aneta Štefllová Petrová
改訂 7.0-22 『スマートカード』のアップデート	Mon Dec 4 2017	Aneta Štefllová Petrová
改訂 7.0-21 マイナーな修正	Mon Nov 20 2017	Aneta Štefllová Petrová
改訂 7.0-20 マイナーな修正	Mon Nov 6 2017	Aneta Štefllová Petrová
改訂 7.0-19 coolkey パッケージに関するセクションのアップデート	Mon Aug 14 2017	Aneta Štefllová Petrová
改訂 7.0-18 7.4 GA 公開用ドキュメントバージョン	Tue Jul 18 2017	Aneta Štefllová Petrová
改訂 7.0-17 リンク切れの修正	Mon Mar 27 2017	Aneta Štefllová Petrová
改訂 7.0-16 Kerberos KDC プロキシのトピック更新。その他のマイナーな更新。	Mon Feb 27 2017	Aneta Štefllová Petrová
改訂 7.0-15 SSSD クライアント側のビューのトピック追加。その他のマイナーな更新。	Wed Dec 7 2016	Aneta Štefllová Petrová
改訂 7.0-14 7.3 GA 公開用バージョン	Tue Oct 18 2016	Aneta Štefllová Petrová
改訂 7.0-13 SCEP での証明書の要求など HTTP (kdcproxy) を使用した Kerberos のトピック追加。その他のマイナーな更新。	Wed Jul 27 2016	Marc Muehlfield
改訂 7.0-11 PAM サービスからのドメインアクセス制限に関するトピックの追加。	Thu Mar 03 2016	Aneta Petrová
改訂 7.0-10 authconfig の章を複数の小さい章に分割。その他のマイナーな更新。	Tue Feb 09 2016	Aneta Petrová
改訂 7.0-9 7.2 GA リリース向けのバージョン	Thu Nov 12 2015	Aneta Petrová
改訂 7.0-8	Fri Mar 13 2015	Tomáš Čapek

7.1 向けの最終変更を含む非同期更新

改訂 7.0-6 7.1 GA リリース用バージョン	Wed Feb 25 2015	Tomáš Čapek
改訂 7.0-4 スプラッシュページでの分類順序を更新して再構築	Fri Dec 05 2014	Tomáš Čapek
改訂 7.0-1 RHEL 7.0 用初期ドラフト	July 16, 2014	Ella Deon Ballard