Red Hat Enterprise Linux 8

RHEL での認証と認可の構成

SSSD、authselect、sssctl、およびスマートカードを使用した認証と認可の設定

Last Updated: 2020-09-07

Red Hat Enterprise Linux 8 RHEL での認証と認可の構成

SSSD、authselect、sssctl、およびスマートカードを使用した認証と認可の設定

法律上の通知

概要

このドキュメントコレクションでは、Red Hat Enterprise Linux 8ホストで認証および認可を設定する方法を説明します。

目次

RED HAT ドキュメントへのフィードバック (英語のみ)

ご意見ご要望をお聞かせください。ドキュメントの改善点はございませんか。改善点を報告する場合は、以下のように行います。

- 特定の文章に簡単なコメントを記入する場合は、以下の手順を行います。
    1. ドキュメントの表示が **Multi-page HTML** 形式になっていて、ドキュメントの右上端に **Feedback** ボタンがあることを確認してください。
    2. マウスカーソルで、コメントを追加する部分を強調表示します。
    3. そのテキストの下に表示される **Add Feedback** ポップアップをクリックします。
    4. 表示される手順に従ってください。
- より詳細なフィードバックを行う場合は、Bugzilla のチケットを作成します。
    1. Bugzilla の Web サイトにアクセスします。
    2. Component で **Documentation** を選択します。
    3. **Description** フィールドに、ドキュメントの改善に関するご意見を記入してください。ドキュメントの該当部分へのリンクも記入してください。
    4. **Submit Bug** をクリックします。

第1章 AUTHSELECT でユーザー認証の設定

1.1. AUTHSELECT の使用方法

authselect は、Red Hat Enterprise Linux ホストで、ユーザー認証の設定を簡略化するユーティリティーです。**authselect** は、最新のすべての ID 管理システムで普遍的に使用できる 2 つの既製プロファイルを提供します。

- **sssd** プロファイル
- **winbind** プロファイル

従来の機能との互換性により、**nis** プロファイルも利用できます。

Red Hat は、たとえば、ドメイン内のサービスを使用するために、データベースの LDAP、winbind、または nis を使用してユーザーを認証している場合など、半集中型の ID 管理環境での **authselect** の使用を推奨しています。

警告

お使いのホストが Red Hat Enterprise Linux Identity Management または Active Directory に含まれる場合は、**authselect** を使用しないでください。**ipa-client-install** コマンドは、ホストを Red Hat Identity Management ドメインに参加させるときに呼び出され、ホスト上で認証を設定します。同様に、**realm join** コマンドは、Active Directory ドメインにホストを参加させるときに呼び出され、ホストで認証を構成します。

authconfig ユーティリティーは、以前の Red Hat Enterprise Linux バージョンで、さまざまな設定ファイルの作成および変更するために使用されていたため、トラブルシューティングが困難になっていました。**authselect** は、このディレクトリーのファイルのみを変更するため、テストとトラブルシューティングが容易になります。

- **/etc/nsswitch.conf**
- **/etc/pam.d/*** ファイル
- **/etc/dconf/db/distro.d/*** ファイル

NSS (Name Service Switch) の構成ファイル (**/etc/nsswitch.conf**) は、GNU C ライブラリーおよびその他の特定のアプリケーションで、さまざまなカテゴリーの名前サービス情報を、どのソースから、どの順番で取得するかを決定するのに使用されます。情報の各カテゴリーは、データベース名で識別されます。

Linux-PAM (Pluggable Authentication Modules) は、システムのアプリケーション (サービス) の認証タスクを処理するモジュールのシステムです。認証の性質は動的に設定できます。システム管理者は、個々のサービス提供アプリケーションがユーザーを認証する方法を選択できます。この動的構成は、**/etc/pam.d/** ディレクトリーの設定ファイルの内容により設定されます。このディレクトリーには、このサービスに必要な認証タスクを実行する PAM の一覧と、個々の PAM が失敗した場合の PAM-API の適切な動作が表示されます。

authselect プロファイルが特定のホストに対して選択されると、そのプロファイルは、そのホストにログインしているすべてのユーザーに適用されます。

1.2. AUTHSELECT プロファイルの選択

システム管理者は、特定のホストの **authselect** ユーティリティーにプロファイルを選択できます。そのプロファイルはそのホストにログインしているすべてのユーザーに適用されます。

手順

1. 認証プロバイダーに適した **authselect** プロファイルを選択します。たとえば、LDAP を使用している企業のネットワークにログインするには、**sssd** を選択します。root で、次のコマンドを実行します。

```
# authselect select sssd
```

2. 必要に応じて、**/etc/nsswitch.conf** ファイルの内容を確認します。

```
passwd:     sss files
group:      sss files
netgroup:   sss files
automount:  sss files
services:   sss files
...
```

/etc/nsswitch.conf ファイルの内容は、**sssd** プロファイルを選択すると、最初の 5 つの項目のうちの 1 つに関する情報が要求されている場合に、システムが最初に **sssd** を使用することを示しています。**sssd** キャッシュ、および認証を提供するサーバーで、要求された情報が見つからない、または **sssd** を実行していないと、システムはローカルファイル (**/etc/***) を調べます。

たとえば、ユーザー ID に関する情報が要求されると、そのユーザー ID は、最初に **sssd** キャッシュで検索されます。そこで見つからない場合は、**/etc/passwd** ファイルが参照されます。同様に、ユーザーのグループ所属が要求されると、最初に **sssd** キャッシュで検索され、そこに見つからない場合に限り、**/etc/group** ファイルが参照されます。

実際には、ローカルの **files** データベースが検索されることはありません。唯一の例外は、**root** ユーザーの場合です。これは、**sssd** で処理されることはありませんが、**files** で処理されます。

3. 必要に応じて、**/etc/pam.d/system-auth** ファイルの内容を確認します。

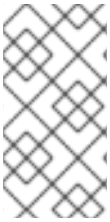
```
# Generated by authselect on Tue Sep 11 22:59:06 2018
# Do not modify this file manually.

auth        required      pam_env.so
auth        required      pam_faildelay.so delay=2000000
auth        [default=1 ignore=ignore success=ok]    pam_succeed_if.so uid >= 1000 quiet
auth        [default=1 ignore=ignore success=ok]    pam_localuser.so
auth        sufficient    pam_unix.so nullok try_first_pass
auth        requisite     pam_succeed_if.so uid >= 1000 quiet_success
auth        sufficient    pam_sss.so forward_pass
auth        required      pam_deny.so
```

```
account     required      pam_unix.so
account     sufficient    pam_localuser.so
...
```

たとえば、**/etc/pam.d/system-auth** ファイルには次の情報が含まれています。

- ユーザーパスワードのロックアウト条件
- スマートカードによる認証の可能性
- 指紋による認証の可能性
  authselect select sssd コマンドまたは **authselect select winbind** コマンドに次のオプションを追加して、デフォルトのプロファイル設定を変更できます。
  - **with-faillock**
  - **with-smartcard**
  - **with-fingerprint**

利用可能なオプションの一覧を確認する場合は、「authconfig から authselect へのスクリプトの変換」か、man ページの authselect-migration(7) を参照してください。



注記

authselect 選択手順を完了する前に、プロファイルに関連する設定ファイルが正しく設定されていることを確認してください。たとえば、**sssd** デーモンが正しく構成されておらずアクティブではない場合に **authselect select** を実行すると、ローカルユーザーのみが、pam_unix を使用して認証できるようになります。

お使いのユースケースで既製のプロファイルを調整する際に、上述のいずれかの **authselect select** コマンドラインオプションでは十分ではない場合は、次を行うことができます。

- **/etc/authselect/user-nsswitch.conf** ファイルを変更して、既製のプロファイルを変更します。詳細は「既製の authselect プロファイルの変更」を参照してください。
- カスタムプロファイルを作成します。詳細は「独自の authselect プロファイルの作成とデプロイメント」を参照してください。

1.3. 既製の AUTHSELECT プロファイルの変更

システム管理者は、デフォルトのプロファイル **sssd**、**winbind**、または **nis** のいずれかを、ニーズに合わせて変更できます。以下の項目を除き、**/etc/authselect/user-nsswitch.conf** ファイルを変更できます。

- passwd
- group
- netgroup
- automount
- services

その後、**authselect select profile_name** を実行すると、プロファイルの変更が **/etc/authselect/user-nsswitch.conf** ファイルから **/etc/nsswitch.conf** ファイルに転送されますが、受け入れられない変更はデフォルトのプロファイル設定により上書きされます。

重要

/etc/nsswitch.conf ファイルを直接編集しないでください。

手順

1. **authselect** プロファイルを選択します。以下に例を示します。

```
# authselect select sssd
```

2. **/etc/authselect/user-nsswitch.conf** ファイルを編集します。

3. **/etc/authselect/user-nsswitch.conf** ファイルから変更を適用します。

```
# authselect apply-changes
```

4. 必要に応じて、**/etc/nsswitch.conf** ファイルで、**/etc/authselect/user-nsswitch.conf** からの変更が伝播されているのを確認してください。

1.4. 独自の AUTHSELECT プロファイルの作成とデプロイメント

システム管理者は、デフォルトプロファイル (**sssd**、**winbind**、または **nis**) のいずれかをカスタマイズすることで、カスタムプロファイルを作成してデプロイできます。これは、「既製の authselect プロファイルの変更」では十分ではない場合に特に便利です。カスタムプロファイルをデプロイすると、そのプロファイルは指定したホストにログインしているすべてのユーザーに適用されます。

手順

1. **authselect create-profile** コマンドを使用してカスタムプロファイルを作成します。たとえば、既製の **sssd** プロファイルに基づく **user-profile** というカスタムプロファイルを作成し、**/etc/nsswitch.conf** ファイルで項目を設定するには、以下のコマンドを実行します。

```
# authselect create-profile user-profile -b sssd --symlink-meta --symlink-pam
New profile was created at /etc/authselect/custom/user-profile
```

コマンドで **--symlink-pam** オプションを使用すると、PAM テンプレートが、コピーではなく元のプロファイルファイルへのシンボリックリンクになります。**--symlink-meta** オプションを使用すると、README、REQUIREMENTS などのメタファイルが、コピーではなく元のプロファイルファイルへのシンボリックリンクになります。これにより、元のプロファイルの PAM テンプレートおよびメタファイルへの今後の更新が、カスタムプロファイルにも反映されます。

このコマンドにより、**/etc/authselect/custom/user-profile/** ディレクトリーの **/etc/nsswitch.conf** ファイルのコピーが作成されました。

2. **/etc/authselect/custom/user-profile/nsswitch.conf** ファイルを設定します。

3. **authselect select** コマンドを実行してカスタムプロファイルを選択し、custom/**name_of_the_profile** をパラメーターとして追加します。たとえば、**user-profile** プロファイルを選択するには、以下のコマンドを実行します。

```
# authselect select custom/user-profile
```

お使いのマシンで **user-profile** プロファイルを選択すると、その後 Red Hat が **sssd** プロファイルを更新した場合に、**/etc/nsswitch.conf** ファイルに行った更新以外のすべての更新を利用できるようになります。

例
次の手順は、**sssd** プロファイルに基づいてプロファイルを作成する方法を示しています。ここでは、ホスト名に対するローカルの静的テーブルルックアップを、**/etc/hosts** ファイルでのみ参照し、**dns** データベースまたは **myhostname** データベースは参照しません。

1. **/etc/nsswitch.conf** ファイルで、次の行を編集します。

```
hosts:      files
```

2. **sssd** に基づいてカスタムプロファイルを作成します。**/etc/nsswitch.conf** に対する変更は除外します。

```
# authselect create-profile user-profile -b sssd --symlink-meta --symlink-pam
```

3. プロファイルを選択します。

```
# authselect select custom/user-profile
```

4. 必要に応じて、カスタムプロファイルで、次の点を確認します。

- 選択した **sssd** プロファイルに応じて **/etc/pam.d/system-auth** ファイルが作成されている。
- **/etc/nsswitch.conf** の設定は変更されていない。

```
hosts:      files
```

注記

一方、**authselect select** を実行すると、**sssd** は次のようになります。

```
hosts:      files dns myhostname
```

1.5. AUTHCONFIG から AUTHSELECT へのスクリプトの変換

ipa-client-install または **realm join** を使用してドメインに参加する場合は、スクリプトの **authconfig** 呼び出しを削除しても問題はありません。これができない場合は、各 **authconfig** コールを、同等の **authselect** コールに置き換えてください。その場合は、正しいプロファイルと適切なオプションを選択します。さらに、必要な設定ファイルを編集します。

- **/etc/krb5.conf**
- **/etc/sssd/sssd.conf** (**sssd** プロファイルの場合) または **/etc/samba/smb.conf** (**winbind** プロファイルの場合)

] および xref:tab-relation-of-authconfig-options-to-authselect-profile-options_configuring-user-authentication-using-authselect[は、**authconfig** オプションに相当する **authselect** を示しています。

表1.1 authconfig オプションと authselect プロファイルの関係

authconfig オプション	authselect プロファイル
--enableldap --enableldapauth	**sssd**
--enablesssd --enablesssdauth	**sssd**
--enablekrb5	**sssd**
--enablewinbind --enablewinbindauth	**winbind**
--enablenis	**nis**

表1.2 authconfig オプションと同等の authselect プロファイルオプション

authconfig オプション	authselect プロファイル機能
--enablesmartcard	with-smartcard
--enablefingerprint	with-fingerprint
--enableecryptfs	with-ecryptfs
--enablemkhomedir	with-mkhomedir
--enablefaillock	with-faillock
--enablepamaccess	with-pamaccess
--enablewinbindkrb5	with-krb5

表1.3「authconfig コマンドと同等の authselect コマンドの例」は、**authconfig** へのキックスタートコールを、**authselect** へのキックスタートコールに変換した例となります。

表1.3 authconfig コマンドと同等の authselect コマンドの例

authconfig コマンド	同等の authselect コマンド
authconfig --enableldap --enableldapauth --enablefaillock --updateall	authselect select sssd with-faillock
authconfig --enablesssd --enablesssdauth --enablesmartcard --smartcardmodule=sssd --updateall	authselect select sssd with-smartcard
authconfig --enableecryptfs --enablepamaccess --updateall	authselect select sssd with-ecryptfs with-pamaccess

authconfig --enablewinbind --enablewinbindauth --winbindjoin=Administrator --updateall	realm join -U Administrator --client-software=winbind **WINBINDDOMAIN**

第2章 SSSD とその利点について

2.1. SSSD の仕組み

システムセキュリティーサービスデーモン (System Security Services Daemon: SSSD) は、リモートディレクトリーと認証メカニズムにアクセスするシステムサービスです。SSSD **クライアント** であるローカルシステムを、外部のバックエンドシステム (**プロバイダー**) に接続できます。以下に例を示します。

- LDAP ディレクトリー
- IdM (Identity Management) ドメイン
- AD (Active Directory) ドメイン
- Kerberos レルム

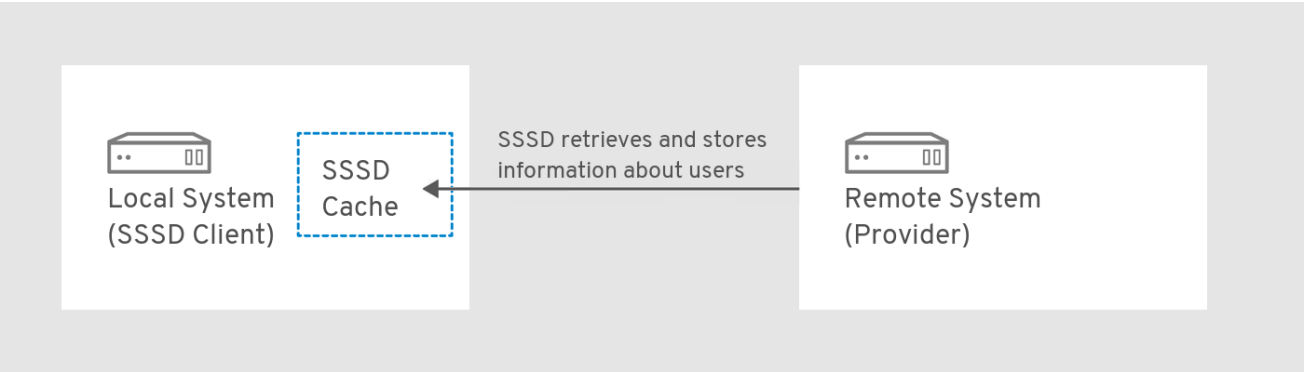
SSSD は、以下の 2 段階で機能します。

1. クライアントをリモートプロバイダーに接続し、ID 情報および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。

ローカルシステムのユーザーは、リモートプロバイダーに保存されているユーザーアカウントを使用して認証できます。

SSSD は、ローカルシステムでユーザーアカウントを作成しません。ただし、SSSD は、IdM ユーザーのホームディレクトリーを作成するように設定できます。作成が完了すると、IdM ユーザーのホームディレクトリーと、クライアント上のコンテンツは、ユーザーがログアウトしても削除されません。

図2.1 SSSD の仕組み



SSSD は、NSS (Name Service Switch) や PAM (Pluggable Authentication Modules) などの複数のシステムサービスのキャッシュを提供することもできます。

2.2. SSSD を使用する利点

SSSD (System Security Services Daemon) を使用すると、ユーザー ID の取得とユーザー認証に複数の利点があります。

オフライン認証

SSSD は、必要に応じて、リモートプロバイダー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。この設定では、セッションの開始時にすでにリモートプロバイダーに対して一

度認証されている場合は、リモートプロバイダーまたはクライアントがオフラインであってもリソースに対して正常に認証できます。

単一のユーザーアカウント: 認証プロセスの一貫性の向上

SSSD では、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。条件は次のとおりです。

- 特定のセッションでは、ユーザーが最低でも一度ログインしている必要があります。ユーザーが初めてログインしたときに、クライアントはリモートプロバイダーに接続する必要があります。

- SSSD でキャッシュを有効にする必要があります。
  SSSD を使用しないと、リモートユーザーには、多くの場合、複数のユーザーアカウントが存在します。たとえば、仮想プライベートネットワーク (VPN) に接続するには、リモートユーザーが、ローカルシステム用のアカウントのほかに、VPN システム用の別のアカウントが必要になります。このシナリオでは、最初にプライベートネットワーク上で認証して、リモートサーバーからユーザーを取得し、ユーザー認証情報をローカルでキャッシュする必要があります。

  SSSD では、キャッシュおよびオフライン認証により、リモートユーザーはローカルマシンに認証することで、ネットワークリソースに接続できます。SSSD は次にネットワークの認証情報を維持します。

ID プロバイダーおよび認証プロバイダーへの負荷の軽減

情報をリクエストすると、クライアントはまずローカルの SSSD キャッシュを確認します。SSSD は、キャッシュで情報が利用できない場合に限り、リモートプロバイダーに問い合わせます。

2.3. クライアントごとに複数の SSSD 設定ファイル

SSSD のデフォルト設定ファイルは **/etc/sssd/sssd.conf** です。このファイルとは別に、SSSD は、**/etc/sssd/conf.d/** ディレクトリーが ***.conf** ファイルのすべてからその設定を読み取ることができます。

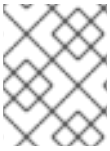
この組み合わせにより、すべてのクライアントでデフォルトの **/etc/sssd/sssd.conf** ファイルを使用し、追加の設定ファイルに追加設定を追加して、クライアントごとに機能を個別に拡張できます。

SSSD が設定ファイルを処理する方法
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/sssd/sssd.conf** ファイル

2. **/etc/sssd/conf.d/** の他の ***.conf** ファイル (アルファベット順)

同じパラメーターが複数の設定ファイルに表示されると、SSSD は最後に読み取る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) を読み込みません。

2.4. SSSD の ID プロバイダーおよび認証プロバイダー

SSSD ドメインとしての ID および認証プロバイダー

ID および認証プロバイダーは、SSSD 設定ファイル **/etc/sssd/sssd.conf** で **ドメイン** として設定されます。プロバイダーは、ファイル **[domain/name of the domain]** セクションまたは **[domain/default]** セクションに登録されます。

1 つのドメインを、以下のプロバイダーのいずれかとして設定できます。

- UID や GID などのユーザー情報を提供する **ID プロバイダー**
  - **/etc/sssd/sssd.conf** ファイルの **[domain/name of the domain]** セクションの **id_provider** オプションを使用して、ドメインを **ID プロバイダー** として指定します。
- 認証要求を処理する **認証プロバイダー**
  - **/etc/sssd/sssd.conf** の **[domain/name of the domain]** セクションの **auth_provider** オプションを使用して、ドメインを **認証プロバイダー** として指定します。
- 認可要求を処理する **アクセス制御プロバイダー**
  - **/etc/sssd/sssd.conf** の **[domain/name of the domain]** セクションの **access_provider** オプションを使用して、ドメインを **アクセス制御プロバイダー** として指定します。デフォルトでは、オプションは **permit** に設定されており、常にすべてのアクセスを許可します。詳細は man ページの **sssd.conf**(5) を参照してください。
- 対応するすべての操作が 1 台のサーバー内で実行される場合など、これらのプロバイダーの組み合わせ
  - この場合、**id_provider**オプション、**auth_provider**オプション、および **access_provider** オプションはすべて、**/etc/sssd/sssd.conf** の **[domain/name of the domain]** または **[domain/default]** セクションに登録されます。

注記

SSSD に複数のドメインを設定できます。少なくともいずれかのドメインを設定する必要があります。設定しないと、SSSD は起動しません。

プロキシープロバイダー

プロキシープロバイダーは、SSSD と、SSSD が使用できないリソースとの間の中間リレーとして機能します。プロキシープロバイダーを使用する場合、SSSD はプロキシーサービスに接続し、プロキシーは指定されたライブラリーを読み込みます。

以下を有効にするために、プロキシープロバイダーを使用するように SSSD を設定できます。

- 指紋スキャナーなどの別の認証方法
- NIS などのレガシーシステム
- **/etc/passwd** ファイルで ID プロバイダーとして定義されるローカルシステムアカウントと、Kerberos などのリモート認証プロバイダー

ID プロバイダーおよび認証プロバイダーの利用可能な組み合わせ

SSSD が、以下の ID プロバイダーと認証プロバイダーの組み合わせを使用するように設定できます。

表2.1 ID プロバイダーおよび認証プロバイダーの利用可能な組み合わせ

ID プロバイダー	認証プロバイダー
Identity Management [a]	Identity Management
Active Directory	Active Directory
LDAP	LDAP
LDAP	Kerberos
Proxy	Proxy
Proxy	LDAP
Proxy	Kerberos
[a] LDAP プロバイダータイプの拡張	

関連情報

- 必要に応じて、**authselect** ユーティリティーを使用して SSSD を設定します。**authselect** の使用に関する詳細は、1章*authselect でユーザー認証の設定*を参照してください。
- ホストが Active Directory (AD) フォレストとの信頼関係にある Identity Management (IdM) に登録されている場合は、**sssctl** ユーティリティーを使用してドメインのステータスを一覧表示し、確認できます。詳細は8章*SSSD を使用したドメイン情報のクエリー*を参照してください。
- **sssctl** ユーティリティーを使用して、アクセス制御レポートを作成し、ユーザーデータを表示できます。詳細は7章*SSSD を使用したホストのユーザーアクセスに関するレポート*を参照してください。

第3章 LDAP を使用し、TLS 認証を必要とする SSSD の設定

3.1. SSSD を使用して、暗号化された方法で LDAP からデータを取得する OPENLDAP クライアント

SSSD (System Security Services Daemon) は、RHEL 8 ホストで ID データの取得と認証を管理するデーモンです。システム管理者は、スタンドアロンの LDAP サーバーデータベースをユーザーアカウントデータベースとして使用するようにホストの SSSD を設定できます。LDAP サーバーの例には、OpenLDAP サーバーと Red Hat 389 Directory Server が含まれます。本章のシナリオには、LDAP サーバーとの接続を TLS 証明書で暗号化する必要があるという要件も含まれています。

LDAP オブジェクトの認証方法は、Kerberos パスワードまたは LDAP パスワードのいずれかになります。LDAP オブジェクトの認証および認可に関する質問は、本章では扱いません。

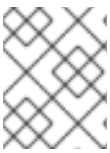
重要

LDAP で SSSD を設定するのは、SSSD および LDAP で高度な専門知識を必要とする複雑な手順です。代わりに、Active Directory や Red Hat Identity Management (IdM) などの統合型の自動ソリューションを使用することを検討してください。IdM の詳細は『Identity Management の計画』を参照してください。

3.2. LDAP を使用し、TLS 認証を必要とする SSSD の設定

以下の手順に従って、Red Hat Enterprise Linux (RHEL) システムを OpenLDAP クライアントとして設定し、以下のクライアント設定を指定します。

- RHEL システムは、OpenLDAP サーバーをユーザーアカウントデータベースとして使用します。
- RHEL システムは、ユーザーデータを取得するサービスとして SSSD (System Security Services Daemon) を使用します。
- RHEL システムは、TLS 証明書を使用して、OpenLDAP サーバーとの接続を暗号化します。



注記

この手順に従って、RHEL システムを Red Hat 389 Directory Server のクライアントとして設定することもできます。

前提条件

- OpenLDAP サーバーがインストールされている。
- OpenLDAP サーバーのクライアントとなるホストで、root 認証情報がある。
- OpenLDAP サーバーのクライアントとなるホストに **/etc/sssd/sssd.conf** ファイルが作成され、**ldap** を **autofs_provider** および **id_provider** として指定するように設定されている。
- **PEM** 形式で保存された OpenLDAP サーバーの TLS 証明書がある。

手順

1. 必要なパッケージをインストールします。
   -

```
# dnf -y install openldap-clients sssd sssd-ldap oddjob-mkhomedir
```

2. 認証プロバイダーを **sssd** に切り替えます。

```
# authselect select sssd with-mkhomedir
```

3. LDAP サーバー証明書が含まれる **core-dirsrv.ca.pem** ファイルを **/etc/openldap/cacerts** ディレクトリーにコピーします。

4. LDAP サーバーの URL と接尾辞を **/etc/openldap/ldap.conf** ファイルに追加します。

```
URI ldap://ldap-server.example.com/
BASE dc=example,dc=com
```

5. **/etc/openldap/ldap.conf** で、**TLS_CACERT** パラメーターを指定する行を **/etc/openldap/cacerts/core-dirsrv.ca.pem** に追加して、OpenLDAP サーバー証明書の場所を指定します。

```
# When no CA certificates are specified the Shared System Certificates
# are in use. In order to have these available along with the ones specified
# by TLS_CACERTDIR one has to include them explicitly:
TLS_CACERT /etc/openldap/certs/core-dirsrv.ca.pem
```

6. **/etc/sssd/sssd.conf** ファイルで、環境の値を **ldap_uri** パラメーターおよび **ldap_search_base** パラメーターに追加します。

```
[domain/default]
id_provider = ldap
autofs_provider = ldap
auth_provider = ldap
chpass_provider = ldap
ldap_uri = ldap://ldap-server.example.com/
ldap_search_base = dc=example,dc=com
ldap_id_use_start_tls = True
cache_credentials = True
ldap_tls_cacertdir = /etc/openldap/certs
ldap_tls_reqcert = allow

[sssd]
services = nss, pam, autofs
domains = default

[nss]
homedir_substring = /home
…
```

7. **/etc/sssd/sssd.conf** で、**[domain/default]** セクションの **ldap_tls_cacert** および **ldap_tls_reqcert** の値を変更して TLS 認証要件を指定します。

```
…
cache_credentials = True
ldap_tls_cacert = /etc/openldap/certs/core-dirsrv.ca.pem
ldap_tls_reqcert = hard
…
```

8. **/etc/sssd/sssd.conf** ファイルの権限を変更します。

```
# chmod 600 /etc/sssd/sssd.conf
```

9. SSSD を再起動して有効にします。

```
# systemctl restart sssd oddjobd
# systemctl enable sssd oddjobd
```

10. (必要に応じて) LDAP サーバーが非推奨の TLS 1.0 プロトコルまたは TLS 1.1 プロトコルを使用している場合は、クライアントシステムでシステム全体の暗号化ポリシーを LEGACY レベルに切り替えて、RHEL 8 がこのプロトコルを使用して通信できるようにします。

```
# update-crypto-policies --set LEGACY
```

詳細は、『RHEL 8.0 リリースノート』の「非推奨の機能」セクションを参照してください。

検証手順

- **id** コマンドで LDAP ユーザーを指定して、ログインを確認します。

```
# id ldap_user
uid=17388(ldap_user) gid=45367(sysadmins)
groups=45367(sysadmins),25395(engineers),10(wheel),1202200000(admins)
```

システム管理者は、**id** コマンドを使用して LDAP からユーザーをクエリーできるようになりました。このコマンドは、正しいユーザー ID とグループメンバーシップを返します。

第4章 AD を認証プロバイダーとして使用するように RHEL を設定

4.1. AD を認証プロバイダーとして使用するスタンドアロンの RHEL ホスト

システム管理者は、Red Hat Enterprise Linux (RHEL) ホストの認証プロバイダーに Active Directory (AD) を使用できます。この場合は、ホストが AD に参加しません。以下に例を示します。

- AD 管理者に対し、ホストの有効化および無効化の制御を付与したくない場合。
- そのホスト (企業 PC) は、社内で 1 人のユーザーのみが使用する予定である。

重要

この手順は、このアプローチが推奨される場合に限り実装してください。通常は推奨されません。

代わりに、システムを AD または Red Hat Identity Management (IdM) に完全に参加させることを検討してください。RHEL ホストをドメインに参加させると、設定を簡単に管理できます。クライアントを直接 AD に参加させることに関連するクライアントアクセスライセンスについて懸念がある場合は、AD との信頼関係にある IdM サーバーを活用することを検討してください。IdM-AD 信頼の詳細は、「IdM と AD との間のフォレスト間の信頼の計画」と「IdM と AD との間のフォレスト間の信頼のインストール」を参照してください。

4.2. AD を認証プロバイダーとして使用する RHEL ホストの設定

この手順を実行すると、**AD_user** という名前のユーザーが、**example.com** ドメインの Active Directory AD ユーザーデータベースに設定されたパスワードを使用して、**rhel8_host** システムにログインできるようになります。この例では、Kerberos レルム **EXAMPLE.COM** は **example.com** ドメインに対応します。

前提条件

- **rhel8_host** への root アクセスがある。
- **AD_user** ユーザーアカウントが **example.com** ドメインにある。
- Kerberos レルムが **EXAMPLE.COM** である。
- **realm join** コマンドを使用して、**rhel8_host** が AD に参加していない。

手順

1. パスワードを割り当てずに、**AD_user** ユーザーアカウントをローカルに作成します。

```
# useradd AD_user
```

2. **/etc/nsswitch.conf** ファイルを開いて編集し、以下の行が含まれていることを確認します。

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

3. **/etc/krb5.conf** ファイルを開いて編集し、以下のセクションと項目が含まれるようにします。

```
# To opt out of the system crypto-policies configuration of krb5, remove the
# symlink at /etc/krb5.conf.d/crypto-policies which will not be recreated.
includedir /etc/krb5.conf.d/

[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
    ticket_lifetime = 24h
    renew_lifetime = 7d
    forwardable = true
    rdns = false
    pkinit_anchors = /etc/pki/tls/certs/ca-bundle.crt
    spake_preauth_groups = edwards25519
    default_realm = EXAMPLE.COM
    default_ccache_name = KEYRING:persistent:%{uid}

[realms]
 EXAMPLE.COM = {
     kdc = ad.example.com
     admin_server = ad.example.com
 }

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

4. **/etc/sssd/sssd.conf** ファイルを作成し、以下のセクションと行をそのファイルに追加します。

```
[sssd]
    services = nss, pam
    domains = EXAMPLE.COM

[domain/EXAMPLE.COM]
    id_provider = files
    auth_provider = krb5
    krb5_realm = EXAMPLE.COM
    krb5_server = ad.example.com
```

5. **/etc/sssd/sssd.conf** ファイルの権限を変更します。

```
# chmod 600 /etc/sssd/sssd.conf
```

6. SSSD (Security System Services Daemon) を起動します。

```
# systemctl start sssd
```

7. SSSD を有効にします。

```
# systemctl enable sssd
```

8. **/etc/pam.d/system-auth** ファイルを開き、以下のセクションと行が含まれるように変更します。

```
# Generated by authselect on Wed May  8 08:55:04 2019
# Do not modify this file manually.

auth        required                                 pam_env.so
auth        required                                 pam_faildelay.so delay=2000000
auth        [default=1 ignore=ignore success=ok]         pam_succeed_if.so uid >= 1000 quiet
auth        [default=1 ignore=ignore success=ok]         pam_localuser.so
auth        sufficient                               pam_unix.so nullok try_first_pass
auth        requisite                                pam_succeed_if.so uid >= 1000 quiet_success
auth        sufficient                               pam_sss.so forward_pass
auth        required                                 pam_deny.so

account     required                                  pam_unix.so
account     sufficient                               pam_localuser.so
account     sufficient                               pam_succeed_if.so uid < 1000 quiet
account     [default=bad success=ok user_unknown=ignore] pam_sss.so
account     required                                 pam_permit.so

password    requisite                                 pam_pwquality.so try_first_pass local_users_only
password    sufficient                               pam_unix.so sha512 shadow nullok try_first_pass
use_authtok
password    sufficient                               pam_sss.so use_authtok
password    required                                  pam_deny.so

session     optional                                 pam_keyinit.so revoke
session     required                                 pam_limits.so
-session    optional                                 pam_systemd.so
session     [success=1 default=ignore]                    pam_succeed_if.so service in crond quiet
use_uid
session     required                                 pam_unix.so
session     optional                                 pam_sss.so
```

9. **/etc/pam.d/system-auth** ファイルの内容を **/etc/pam.d/password-auth** ファイルにコピーします。**yes** と入力して、ファイルの現在の内容を上書きします。

```
# cp /etc/pam.d/system-auth /etc/pam.d/password-auth
cp: overwrite '/etc/pam.d/password-auth'? yes
```

検証手順

1. **AD_user** 用の Kerberos チケット保証チケット (TGT) を要求します。 **AD_user** のパスワードを必要に応じて入力します。

```
# kinit AD_user
Password for AD_user@EXAMPLE.COM:
```

2. 取得した TGT を表示します。

```
# klist
```

```
Ticket cache: KEYRING:persistent:0:0
Default principal: AD_user@EXAMPLE.COM

Valid starting     Expires            Service principal
11/02/20 04:16:38  11/02/20 14:16:38  krbtgt/EXAMPLE.COM@EXAMPLE.COM
 renew until 18/02/20 04:16:34
```

AD_user は、Kerberos ドメイン **EXAMPLE.COM** の認証情報を使用して **rhel8_host** に正常にログインしました。

第5章 AUTHSELECT でスマートカードの設定

本章では、以下のいずれかを達成するためのスマートカードの設定方法を説明します。

- パスワードとスマートカード認証の両方を有効化
- パスワードの無効化およびスマートカード認証の有効化
- 削除時にロックの有効化

前提条件

- authselect がインストールされている。
  authselect ツールは、Linux ホストでユーザー認証を設定し、スマートカード認証パラメーターを設定するのに使用できます。authselect の詳細は、「authselect でユーザー認証の設定」を参照してください。
- RHEL 8 で対応しているスマートカードまたは USB デバイス
  詳細は、「Smart Card support in RHEL8」 を参照してください。

5.1. スマートカードの対象となる証明書

authselect を使用してスマートカードを設定する前に、証明書をカードにインポートする必要があります。以下のツールを使用して証明書を生成できます。

- Active Directory (AD)
- Identity Management (IdM)
  IdM 証明書を作成する方法は、「新しいユーザー証明書を要求し、クライアントにエクスポート」を参照してください。
- Red Hat Certificate System (RHCS)
  詳細は、「Managing Smart Cards with the Enterprise Security Client」 を参照してください。
- ローカルの認証局ユーザーがドメインの一部ではない場合、またはテスト目的である場合は、ローカル認証局が生成した証明書を使用可能。
  ローカル認証を作成してスマートカードにインポートする方法の詳細は、「ローカル証明書のスマートカードへの設定とインポート」を参照してください。

5.2. ユーザーパスワード認証を有効にしてスマートカード認証を設定

本セクションでは、システムでスマートカードとパスワードの両方認証を有効にする方法を説明します。

前提条件

- スマートカードに、証明書と秘密鍵が含まれる。
- そのカードがリーダーに挿入され、コンピューターに接続されている。
- **authselect** ツールがシステムにインストールされている。

手順

- 以下のコマンドを実行して、スマートカードとパスワード認証を許可します。

```
# authselect select sssd with-smartcard --force
```

この時点で、スマートカード認証が有効になります。ただし、スマートカードを忘れてしまった場合は、パスワード認証が動作します。

5.3. スマートカード認証を強制するための AUTHSELECT の設定

authselect ツールを使用すると、システムでスマートカード認証を設定でき、デフォルトのパスワード認証を無効にできます。**authselect** コマンドには以下のオプションが含まれている必要があります。

- **with-smartcard** - スマートカード認証の有効化
- **with-smartcard-required** - 排他的なスマートカード認証の有効化 (パスワードによる認証は無効になっています)

前提条件

- スマートカードに、証明書と秘密鍵が含まれている。
- そのカードがリーダーに挿入され、コンピューターに接続されている。
- **authselect** ツールがローカルシステムにインストールされている。

手順

- 以下のコマンドを実行して、スマートカード認証を強制します。

```
# authselect select sssd with-smartcard  with-smartcard-required --force
```

この時点では、スマートカードでのみログインできます。パスワード認証は使用できなくなります。

5.4. 削除時にロックを使用したスマートカード認証の設定

authselect サービスを使用すると、スマートカードの認証を設定して、リーダーからスマートカードを削除した後も、画面を即時にロックできます。**authselect** コマンドには以下の変数が含まれている必要があります。

- **with-smartcard** - スマートカード認証の有効化
- **with-smartcard-required** - 排他的なスマートカード認証の有効化 (パスワードによる認証は無効になっています)
- **with-smartcard-lock-on-removal** - スマートカードの削除後に強制的にログアウト

前提条件

- スマートカードに、証明書と秘密鍵が含まれている。
- そのカードがリーダーに挿入され、コンピューターに接続されている。
- **authselect** ツールがローカルシステムにインストールされている。

手順

- 以下のコマンドを実行して、スマートカード認証の有効化、パスワード認証の無効化、削除時のロックの強制を行います。

```
# authselect select sssd  with-smartcard  with-smartcard-required with-smartcard-lock-on-removal --force
```

これで、カードを削除すると、画面がロックされます。ロックを解除するには、スマートカードを再度挿入する必要があります。

第6章 ローカル証明書のスマートカードへの設定およびインポート

本章では、以下のシナリオを説明します。

- ホストがドメインに接続されていない
- このホストで、スマートカードで認証する必要がある
- スマートカード認証を使用して SSH アクセスを設定する必要がある
- **authselect** を使用してスマートカードを設定する必要がある

このシナリオを行うには、以下の設定を使用します。

- スマートカードで認証するユーザーのユーザー証明書を取得する。証明書は、ドメインで使用される信頼できる認証局によって生成される必要があります。
  証明書を取得できない場合は、テスト目的で、ローカルの認証局が署名したユーザー証明書を生成します。
- スマートカードに証明書と秘密鍵を保存する。
- SSH アクセス用のスマートカード認証を設定する。

重要

ホストがドメインの一部である場合は、そのホストをドメインに追加し、Active Directory または Identity Management 認証局が生成した証明書を使用します。

スマートカードに IdM 証明書を作成する方法は、「スマートカード認証用の Identity Management の設定」を参照してください。

前提条件

- authselect がインストールされている。
  authselect ツールは、Linux ホストでユーザー認証を設定し、スマートカード認証パラメーターを設定するのに使用できます。authselect の詳細は、「authselect とは」を参照してください。
- RHEL 8 で対応しているスマートカードまたは USB デバイス
  詳細は、「Smart Card support in RHEL8」 を参照してください。

6.1. ローカル証明書の作成

本セクションでは、以下のタスクを実行する方法を説明します。

- OpenSSL 認証局の生成
- 証明書署名リクエストの作成

警告

以下の手順は、テスト目的のみを想定しています。ローカルの自己署名証明局により生成される証明書は、AD、IdM、または RHCS 認証局を使用する場合と同じように安全ではありません。ホストがドメインに含まれていない場合でも、企業の認定機関が生成した証明書を使用する必要があります。

手順

1. 証明書を生成するディレクトリーを作成します。以下に例を示します。

```
# mkdir /tmp/ca
# cd /tmp/ca
```

2. 証明書を設定します (このテキストは、**ca** ディレクトリーのコマンドラインにコピーします)。

```
cat > ca.cnf <<EOF
[ ca ]
default_ca = CA_default

[ CA_default ]
dir             = .
database        = \$dir/index.txt
new_certs_dir   = \$dir/newcerts

certificate     = \$dir/rootCA.crt
serial          = \$dir/serial
private_key     = \$dir/rootCA.key
RANDFILE        = \$dir/rand

default_days    = 365
default_crl_days = 30
default_md      = sha256

policy          = policy_any
email_in_dn     = no

name_opt        = ca_default
cert_opt        = ca_default
copy_extensions  = copy

[ usr_cert ]
authorityKeyIdentifier = keyid, issuer

[ v3_ca ]
subjectKeyIdentifier   = hash
authorityKeyIdentifier = keyid:always,issuer:always
basicConstraints       = CA:true
keyUsage               = critical, digitalSignature, cRLSign, keyCertSign

[ policy_any ]
```

```
organizationName       = supplied
organizationalUnitName = supplied
commonName             = supplied
emailAddress           = optional

[ req ]
distinguished_name = req_distinguished_name
prompt             = no

[ req_distinguished_name ]
O  = Example
OU = Example Test
CN = Example Test CA
EOF
```

3. 以下のディレクトリーを作成します。

```
# mkdir certs crl newcerts
```

4. 以下のファイルを作成します。

```
# touch index.txt crlnumber index.txt.attr
```

5. シリアルファイルに番号 01 を書き込みます。

```
# echo 01 > serial
```

このコマンドは、シリアル番号内に 01 を書き込みます。これは証明書のシリアル番号です。この CA によってリリースされる新しい証明書ごとに、数が 1 つ増えます。

6. OpenSSL root CA キーを作成します。

```
# openssl genrsa -out rootCA.key 2048
```

7. 自己署名 root 認証局証明書を作成します。

```
# openssl req -batch -config ca.cnf \
    -x509 -new -nodes -key rootCA.key -sha256 -days 10000 \
    -set_serial 0 -extensions v3_ca -out rootCA.crt
```

8. ユーザー名のキーを作成します。

```
# openssl genrsa -out example.user.key 2048
```

このキーは、セキュリティーが保護されていないローカルシステムで生成されるため、キーがカードに格納されると、キーがシステムから削除されます。

キーはスマートカードで直接作成できます。これを行う場合は、スマートカードの製造元が作成した手順に従います。

9. 証明書署名リクエスト設定ファイルを作成します (このテキストを ca ディレクトリーでコマンドラインにコピーします)。

```
cat > req.cnf <<EOF
```

```
[ req ]
distinguished_name = req_distinguished_name
prompt = no

[ req_distinguished_name ]
O = Example
OU = Example Test
CN = testuser

[ req_exts ]
basicConstraints = CA:FALSE
nsCertType = client, email
nsComment = "testuser"
subjectKeyIdentifier = hash
keyUsage = critical, nonRepudiation, digitalSignature, keyEncipherment
extendedKeyUsage = clientAuth, emailProtection, msSmartcardLogin
subjectAltName = otherName:msUPN;UTF8:testuser@EXAMPLE.COM,
email:testuser@example.com
EOF
```

10. example.user 証明書用の証明書署名リクエストを作成します。

```
# openssl req -new -nodes -key example.user.key \
    -reqexts req_exts -config req.cnf -out testuser.csr
```

11. 新しい証明書を設定します。有効期限は 1 年に設定されています。

```
# openssl ca -config ca.cnf -batch -notext \
    -keyfile rootCA.key -in example.user.csr -days 365 \
    -extensions usr_cert -out example.user.crt
```

この時点で、認証局と証明書が正常に生成され、スマートカードにインポートできる状態になります。

6.2. SSSD ディレクトリーへの証明書のコピー

GNOME デスクトップマネージャー (GDM) には SSSD が必要です。GDM を使用する場合は、**/etc/sssd/pki** ディレクトリーに PEM 証明書をコピーする必要があります。

前提条件

- ローカル CA の認証局および証明書が生成されている。

手順

1. システムに SSSD がインストールされていることを確認します。

```
# rpm -q sssd
sssd-2.0.0.43.el8_0.3.x86_64
```

2. **/etc/sssd/pki** ディレクトリーを作成します。

```
# file /etc/sssd/pki
/etc/sssd/pki/: directory
```

3. **rootCA.crt** を PEM ファイルとして **/etc/sssd/pki/** ディレクトリーにコピーします。

```
# cp /tmp/ca/rootCA.crt /etc/sssd/pki/sssd_auth_ca_db.pem
```

これで、認証局と証明書が生成され、**/etc/sssd/pki** ディレクトリーに保存されました。

注記

別のアプリケーションで認証局の証明書を共有する場合は、sssd.conf の場所を変更してください。

- SSSD PAM レスポンダー: **[pam]** セクションの **pam_cert_db_path**
- SSSD ssh レスポンダー: **[ssh]** セクションの **ca_db**

詳細は、man ページの **sssd.conf** を参照してください。

Red Hat では、デフォルトのパスを維持して、SSSD に専用の認証局証明書ファイルを使用し、認証に信頼されている認証局のみをここに登録するようにすることを推奨します。

6.3. スマートカードを管理および使用するツールのインストール

スマートカードを設定するには、証明書を生成し、スマートカードに保存するツールが必要になります。

以下を行う必要があります。

- 証明書管理に役立つ **gnutls-utils** パッケージをインストールする。
- スマートカードと連携するライブラリーおよびユーティリティーのセットを提供する **opensc** パッケージをインストールする。
- スマートカードリーダーと通信する **pcscd** サービスを開始する。

手順

1. **opensc** パッケージおよび **gnutls-utils** パッケージをインストールします。

```
# dnf -y install opensc gnutls-utils
```

2. **pcscd** サービスを開始します。

```
# systemctl start pcscd
```

pcscd サービスが稼働していることを確認します。

6.4. スマートカードでの証明書の保存

本セクションでは、設定に役立つ **pkcs15-init** によるスマートカードの設定を説明します。

- スマートカードの消去
- 新しい PIN および任意の PIN ブロック解除キー (PUK) の設定

- スマートカードでの新規スロットの作成
- スロットへの証明書、秘密鍵、および公開鍵の保存
- スマートカード設定のロック (一部のスマートカードではこのタイプのファイナライズが必要になります)

前提条件

- **pkcs15-init** ツールを含む **opensc** パッケージがインストールされている。
  詳細は「スマートカードを管理および使用するツールのインストール」を参照してください。
- カードがリーダーに挿入され、コンピューターに接続されている。
- スマートカードに保存する秘密鍵、公開鍵、および証明書がある。この手順の **testuser.key**、**testuserpublic.key**、および **testuser.crt** は、秘密鍵、公開鍵、および証明書に使用される名前です。
- 現在のスマートカードユーザーの PIN およびセキュリティー担当者 (セキュリティーオフィサー) の PIN (SO-PIN)

手順

1. スマートカードを消去して PIN で自身を認証します。

```
$ pkcs15-init --erase-card --use-default-transport-keys
Using reader with a card: Reader name
PIN [Security Officer PIN] required.
Please enter PIN [Security Officer PIN]:
```

   カードが削除されました。

2. スマートカードを初期化し、ユーザーの PIN と PUK を設定します。また、セキュリティー担当者の PIN と PUK を設定します。

```
$ pkcs15-init --create-pkcs15 --use-default-transport-keys \
    --pin 963214 --puk 321478 --so-pin 65498714 --so-puk 784123
Using reader with a card: Reader name
```

   pcks15-init ツールは、スマートカードに新しいスロットを作成します。

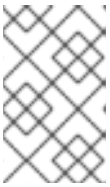
3. スロットのラベルと認証 ID を設定します。

```
$ pkcs15-init --store-pin --label testuser \
    --auth-id 01 --so-pin 65498714 --pin 963214 --puk 321478
Using reader with a card: Reader name
```

   ラベルは人間が判読できる値に設定されます (この場合は **testuser**)。**auth-id** は 16 進数の値である必要があります。この場合、**01** に設定されます。

4. スマートカードの新しいスロットに秘密鍵を保存し、ラベルを付けます。

```
$ pkcs15-init --store-private-key testuser.key --label testuser_key \
    --auth-id 01 --id 01 --pin 963214
Using reader with a card: Reader name
```



注記

--id に指定する値は、秘密鍵と証明書を保存する際に同じである必要があります。**--id** の値を指定しないと、ツールがより複雑な値を計算するため、独自の値の定義が容易になります。

5. スマートカードの新しいスロットに証明書を保存し、ラベル付けします。

```
$ pkcs15-init --store-certificate testuser.crt --label testuser_crt \
    --auth-id 01 --id 01 --format pem --pin 963214
Using reader with a card: Reader name
```

6. (必要に応じて) スマートカードの新しいスロットに公開鍵を保存し、ラベルを付けます。

```
$ pkcs15-init --store-public-key testuserpublic.key
    --label testuserpublic_key --auth-id 01 --id 01 --pin 963214
Using reader with a card: Reader name
```

注記

公開鍵が秘密鍵または証明書、もしくはその両方に対応している場合は、その秘密鍵または証明書 (もしくはその両方) と同じ ID を指定する必要があります。

7. (必要に応じて) スマートカードの中には、設定をロックしてカードをファイナライズする必要があるものもあります。

```
$ pkcs15-init -F
```

この段階では、スマートカードには、新たに作成されたスロットに証明書、秘密鍵、および公開鍵が含まれます。ユーザーの PIN と PUK、およびセキュリティー担当者の PIN と PUK も作成しました。

6.5. スマートカード認証で SSH アクセスの設定

SSH 接続には認証が必要です。パスワードまたは証明書を使用できます。本セクションでは、以下を説明します。

- スマートカードに保存されている証明書を使用して認証を有効にするのに必要な設定
- **authselect** ツールを使用した削除設定のロック

削除時ロックを設定すると、スマートカードの削除後に強制的にログアウトされます。

authselect を使用したスマートカードの設定の詳細は「authselect でスマートカードの設定」を参照してください。

前提条件

- スマートカードに、証明書と秘密鍵が含まれている。
- カードがリーダーに挿入され、コンピューターに接続されている。
- SSSD がインストールされ、設定されている。

- ユーザー名が、証明書の SUBJECT の CN (Common Name) または UID (User ID) と一致する。
- **pcscd** サービスがローカルマシンで実行している。
  詳細は「スマートカードを管理および使用するツールのインストール」を参照してください。

手順

1. スマートカード認証を使用するユーザーのホームディレクトリーで、SSH キー用の新しいディレクトリーを作成します。

```
# mkdir /home/example.user/.ssh
```

2. **opensc** ライブラリーで **ssh-keygen -D** コマンドを実行して、スマートカードの秘密鍵とペアの既存の公開鍵を取得し、そのユーザーの SSH キーディレクトリーの **authorized_keys** 一覧に追加し、スマートカード認証を使用した SSH アクセスを有効にします。

```
# ssh-keygen -D /usr/lib64/pkcs11/opensc-pkcs11.so >>
~example.user/.ssh/authorized_keys
```

3. SSH では、**/.ssh** ディレクトリーおよび **authorized_keys** ファイルのアクセス権が必要です。アクセス権を設定または変更するには、以下を入力します。

```
# chown -R example.user:example.user ~example.user/.ssh/
# chmod 700 ~example.user/.ssh/
# chmod 600 ~example.user/.ssh/authorized_keys
```

4. 必要に応じて、キーを表示します。

```
# cat ~example.user/.ssh/authorized_keys
```

端末にキーが表示されます。

5. **/etc/sssd/sssd.conf** ファイルでスマートカード認証が有効になっていることを確認します。**[pam]** セクションで、pam 証明書認証モジュールである **pam_cert_auth = True** を有効にします。

sssd.conf ファイルがまだ作成されていない場合は、以下のスクリプトをコマンドラインにコピーして、最小限の機能設定を作成できます。

```
# cat > /etc/sssd/sssd.conf <<EOF
[sssd]
services = nss, pam
domains = shadowutils

[nss]

[pam]
pam_cert_auth = True

[domain/shadowutils]
id_provider = files
EOF
```

6. SSH キーを使用するには、**authselect** コマンドで認証を設定します。

```
# authselect select sssd with-smartcard with-smartcard-lock-on-removal --force
```

これで、次のコマンドを実行して SSH アクセスを確認できます。

```
# ssh -I /usr/lib64/opensc-pkcs11.so -l example.user localhost hostname
```

設定に成功すると、スマートカードの PIN を入力するように求められます。

設定がローカルで機能するようになりました。これで、公開鍵をコピーして、SSH を使用するすべてのサーバーにある **authorized_keys** ファイルに配布できます。

第7章 SSSD を使用したホストのユーザーアクセスに関するレポート

SSSD (Security System Services Daemon) は、どのユーザーがクライアントにアクセスできるか、またはできないかを追跡します。本章では、**sssctl** ツールを使用してアクセス制御レポートを作成し、ユーザーデータを表示する方法を説明します。

前提条件

- SSSD パッケージがネットワーク環境にインストールされている。

7.1. SSSCTL コマンド

sssctl は、Security System Services Daemon (SSSD) を使用したコマンドラインツールで、以下の情報を収集します。

- ドメインの状態
- クライアントユーザー認証
- 特定のドメインのクライアントへのユーザーアクセス
- キャッシュされたコンテンツに関する情報

sssctl ツールを使用すると、以下が可能になります。

- SSSD キャッシュの管理
- ログの管理
- 設定ファイルの確認

注記

sssctl ツールは、**sss_cache** ツールおよび **sss_debuglevel** ツールに代わるものです。

関連情報

- **sssctl** の詳細は、以下を入力します。

```
# sssctl --help
```

7.2. SSSCTL を使用したアクセス制御レポートの生成

SSSD は、クライアントにログインできるユーザーを制御するため、レポートを実行しているマシンに適用されるアクセス制御ルールを一覧表示できます。

注記

キー配布センター (KDC) がロックアウトしたユーザーをツールが追跡しないため、アクセスレポートは正確ではありません。

前提条件

- 管理者権限でログインしている。
- **sssctl** が RHEL 7 および RHEL 8 システムで利用できる。

手順

- **idm.example.com** ドメインのレポートを生成するには、以下を入力します。

```
[root@client1 ~]# sssctl access-report idm.example.com
1 rule cached

Rule name: example.user
 Member users: example.user
 Member services: sshd
```

7.3. SSSCTL でユーザー認可の詳細の表示

sssctl user-checks コマンドは、SSSD (System Security Services Daemon) をユーザールックアップ、認証、および認可に使用するアプリケーションでの問題のデバッグに役立ちます。

sssctl user-checks [USER_NAME] コマンドは、NSS (Name Service Switch) および D-Bus インターフェースの InfoPipe レスポンダーで利用可能なユーザーデータを表示します。表示されるデータは、ユーザーが **system-auth** の PAM (Pluggable Authentication Module) サービスを使用してログインすることを許可されているかどうかを示します。

コマンドには、2 つのオプションがあります。

- **-a** (PAM アクション用)
- **-s** (PAM サービス用)

-a オプションおよび **-s** オプションを定義しない場合、**sssctl** ツールはデフォルトのオプション **-a acct -s system-auth** を使用します。

前提条件

- 管理者権限でログインしている。
- **sssctl** ツールが RHEL 7 および RHEL 8 システムで利用できる。

手順

- 特定ユーザーのユーザーデータを表示するには、以下を入力します。

```
[root@client1 ~]# sssctl user-checks -a acct -s sshd example.user
user: example.user
action: acct
service: sshd
....
```

関連情報

- **sssctl user-checks** の詳細は、以下のコマンドを実行します。

```
sssctl user-checks --help
```

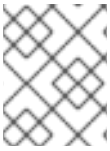
第8章 SSSD を使用したドメイン情報のクエリー

SSSD (Security System Services Daemon) は、フォレスト間の信頼の Active Directory ドメインなど、Identity Management (IdM) のドメインを一覧表示できます。一覧表示された各ドメインのステータスを確認することもできます。

- sssctl コマンドを使用したドメインの一覧表示
- sssctl コマンドを使用したドメインステータスの確認

8.1. SSSCTL を使用したドメインの一覧表示

sssctl domain-list コマンドは、ドメイントポロジーに関する問題のデバッグに役立ちます。



注記

ステータスはすぐに利用できない可能性があります。ドメインが表示されない場合は、コマンドを繰り返します。

前提条件

- 管理者権限でログインしている。
- **sssctl** が RHEL 7 および RHEL 8 システムで利用できる。

手順

1. sssctl コマンドのヘルプを表示するには、以下のコマンドを実行します。

```
[root@client1 ~]# sssctl --help
....
```

2. 利用可能なドメインの一覧を表示するには、以下を入力します。

```
[root@client1 ~]# sssctl domain-list
implicit_files
idm.example.com
ad.example.com
sub1.ad.example.com
```

一覧には、Active Directory と Identity Management との間でフォレスト間の信頼関係にあるドメインが含まれます。

8.2. SSSCTL でドメインステータスの確認

sssctl domain-status コマンドは、ドメイントポロジーに関する問題のデバッグに役立ちます。

注記

ステータスはすぐに利用できない可能性があります。ドメインが表示されない場合は、コマンドを繰り返します。

前提条件

- 管理者権限でログインしている。
- **sssctl** が RHEL 7 および RHEL 8 システムで利用できる。

手順

1. sssctl コマンドのヘルプを表示するには、以下のコマンドを実行します。

```
[root@client1 ~]# sssctl --help
```

2. 特定ドメインのユーザーデータを表示するには、以下を入力します。

```
[root@client1 ~]# sssctl domain-status idm.example.com
Online status: Online

Active servers:
IPA: master.idm.example.com

Discovered IPA servers:
- master.idm.example.com
```

ドメイン **idm.example.com** はオンラインになり、コマンドを設定したクライアントから表示されます。

ドメインが利用できないと、結果は以下のようになります。

```
[root@client1 ~]# sssctl domain-status ad.example.com
Unable to get online status
```

第9章 ローカル SSSD 設定の誤字の排除

sssctl config-check コマンドを使用して、ホストの **/etc/sssd/sssd.conf** ファイルに誤字のエラーがあるかどうかをテストできます。

前提条件

- root でログインしている。

手順

1. **sssctl config-check** コマンドを実行します。

```
# sssctl config-check

Issues identified by validators: 1
[rule/allowed_domain_options]: Attribute 'ldap_search' is not allowed in section
'domain/example1'. Check for typos.

Messages generated during configuration merging: 0

Used configuration snippet files: 0
```

2. **/etc/sssd/sssd.conf** ファイルを開き、タイプミスを修正します。前の例でエラーメッセージが発生した場合は、**ldap_search** を **ldap_search_base** に置き換えます。

```
[...]
[domain/example1]
ldap_search_base = dc=example,dc=com
[...]
```

3. ファイルを保存します。

4. SSSD を再起動します。

```
# systemctl restart sssd
```

検証手順

- **sssctl config-check** コマンドを実行します。

```
# sssctl config-check

Issues identified by validators: 0

Messages generated during configuration merging: 0

Used configuration snippet files: 0
```

/etc/sssd/sssd.conf ファイルで、誤字がなくなりました。