Red Hat Enterprise Linux 7

システムレベルの認証ガイド

アプリケーションおよびサービスを使用したローカルシステムでの認証の設定

Last Updated: 2024-05-01

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

アプリケーションおよびサービスを使用したローカルシステムでの認証の設定

Florian Delehaye
Red Hat Customer Content Services
fdelehay@redhat.com

Marc Muehlfeld
Red Hat Customer Content Services

Filip Hanzelka
Red Hat Customer Content Services

Lucie Maňásková
Red Hat Customer Content Services

Aneta Šteflová Petrová
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

概要

本ガイドでは、ローカルシステムで認証を設定するのに利用できるアプリケーションおよびサービスを説明します。 本ガイドに加えて、Red Hat Enterprise Linux Identity Management に関連する機能とサービスに関するドキュメントは、以下のガイドを参照してください。 Linux ドメイン ID、認証、およびポリシーガイド では、Linux ベースのドメインで ID ストアと認証および承認ポリシーを一元管理して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 を使用して Linux ドメインを Microsoft Windows Active Directory (AD)と統合する方法を説明します。本ガイドでは、SSSD を使用した Common Internet File System (CIFS)および realmd システムへのアクセスに SSSD を使用した直接的および間接 AD 統合のさまざまな側面を説明します。

目次

第1章 システム認証の概要

セキュアなネットワーク環境を確立するための基盤の 1 つは、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が制限されるようにすることです。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** できます。つまり、ユーザーはアイデンティティーを検証できます。

Red Hat Enterprise Linux システムには、ユーザー ID を作成して識別できるさまざまなサービスがあります。これは、Kerberos や Samba などの大きな ID ドメインに接続するローカルシステムのファイルや、そのドメインを作成するツールになります。

本書では、管理者がローカルシステムの認証およびアイデンティティーを管理するために利用できる一般的なシステムサービスおよびアプリケーションを説明します。その他のガイドは、Linux ドメインの作成、および Linux システムの Windows ドメインへの統合 に関する詳細を提供する利用可能なガイドです。

1.1. ユーザーアイデンティティーの確認

認証 は、アイデンティティーを確認するプロセスです。ネットワークの対話については、認証には、別の当事者による識別が必要です。ネットワーク上で認証を使用する方法は、簡単なパスワード、証明書、ワンタイムパスワード (OTP) トークン、生体認証スキャンなどの数多くの方法があります。

反対に **承認** を行うと、認証されたユーザーが操作またはアクセスできるものを定義します。

認証では、ユーザーがなんらかの **認証情報** を提示してアイデンティティーを確認する必要があります。必要な認証情報の種類は、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には、以下のような認証があります。

- **パスワードベースの認証**。ほとんどのすべてのソフトウェアは、認識されたユーザー名とパスワードを提供することでユーザーが認証できるようにします。これは **簡易認証** とも呼ばれます。

- **証明書ベースの認証の使用**。証明書に基づくクライアント認証は、SSL プロトコルの一部です。クライアントは無作為に生成されたデータの一部に署名し、ネットワーク全体で証明書および署名されたデータの両方を送信します。サーバーは署名を検証し、証明書の有効性を確認します。

- **Kerberos 認証**。Kerberos は、**TGT (ticket-granting tickets)** と呼ばれる有効期限の短い認証情報のシステムを確立します。ユーザーは、ユーザーを特定し、ユーザーにチケットを発行できることをシステムに示す認証情報、つまりユーザー名およびパスワードを提示します。TGT は、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ために繰り返し使用できます。TGT を使用した認証では、ユーザーはこの方法で単一の認証プロセスのみを実現できます。

- **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のバリアントです。スマートカード (または **トークン**) は、ユーザー証明書を格納します。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、アクセスを付与できます。スマートカードを使用したシングルサインオンには、以下の 3 つの手順があります。

    1. ユーザーがスマートカードをカードリーダーに挿入します。Red Hat Enterprise Linux のプラグ可能な認証モジュール (PAM) は、挿入されたスマートカードを検出します。

    2. システムは、証明書をユーザーエントリーにマップし、スマートカードに表示された証明書を、証明書ベースの認証で説明されているように秘密鍵で暗号化して、ユーザーエントリーに保存されている証明書と比較します。

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、追加の識別メカニズムとして証明書を追加し、物理的なアクセス要件を追加することにより、Kerberos によって確立された単純な認証層に基づいています。

1.2. シングルサインオンの計画の追加

「ユーザーアイデンティティーの確認」 で説明されているように、認証に関するものは、すべての安全なアプリケーションが、それにアクセスするために少なくともパスワードを必要とするということです。中央のアイデンティティーストアがなく、すべてのアプリケーションはユーザーおよび認証情報の独自のセットを維持することなく、ユーザーは 1 つのサービスまたはアプリケーションごとにパスワードを入力する必要があります。1 日に数回、場合によっては数分ごとにパスワードを入力が求められる場合があります。

複数のパスワードを維持し、それらを入力し続けることは、ユーザーおよび管理者にとって困難です。**シングルサインオン** は、管理者が単一のパスワードストアを作成してユーザーが一度ログインし、単一のパスワードを使用してすべてのネットワークリソースに認証できるようにするための設定です。

Red Hat Enterprise Linux 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ロック解除、Mozilla Firefox でセキュリティーが保護された Web ページへのアクセスなど、複数のリソースのシングルサインオンをサポートします。PAM、NSS、Kerberos などの他のシステムサービスでは、これらの ID ソースを使用するように他のシステムアプリケーションを設定できます。

シングルサインオンは、ユーザーにとって便利であると同時に、サーバーおよびネットワークのセキュリティーにおけるもう 1 つの層でもあります。シングルサインオンは、セキュアで効果的な認証をオンにします。Red Hat Enterprise Linux は、シングルサインオンを有効にするために使用できる認証メカニズムを 2 つ提供します。

- Kerberos レルムと Active Directory ドメインを使用した Kerberos ベースの認証
- スマートカードベースの認証

どちらの方法でも、(Kerberos レルムまたは公開鍵インフラストラクチャーの認証局を介して) 一元化された ID ストアを作成し、ローカルシステムサービスは、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、ローカルシステムのローカルユーザーの認証を設定するのに利用可能なサービスがいくつかあります。これらには以下が含まれます。

認証設定

- 認証設定ツール(**authconfig**)は、システムにさまざまなアイデンティティーバックエンドと、そのシステムへの認証手段（パスワード、フィンガープリント、スマートカードなど）を設定します。

Identity バックエンドの設定

- SSSD (Security System Services Daemon) は、Microsoft Active Directory や Red Hat Enterprise Linux IdM などの複数のアイデンティティープロバイダー) 主に LDAP ベースのディレクトリー) を設定します。これは、ローカルシステムとアプリケーションの両方で使用できます。パスワードとチケットがキャッシュされ、認証情報を再利用することでオフライン認証とシングルサインオンが可能になります。

- **realmd** サービスは、認証バックエンド(IdM の SSSD)の設定を可能にするコマンドラインユーティリティーです。**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、システムをアカウントとしてドメインに参加させます。

- NSS (Name Service Switch) 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のメカニズムです。NSS は、必要な情報を取得するのに使用するモジュールであるソースを決定します。たとえば、ユーザー情報は **/etc/passwd** ファイルや LDAP ベースのディレクトリーなどの従来の UNIX ファイルに配置できますが、ホストアドレスは **/etc/hosts** ファイルや DNS レコードなどのファイルから読み取ることができます。NSS は情報が保存される場所を特定します。

認証メカニズム

- プラグ可能な認証モジュール (PAM) は、認証ポリシーを設定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、認証のさまざまな側面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使用する PAM モジュールは、アプリケーションの設定方法に基づいています。利用可能な PAM モジュールには、Kerberos、Winbind、またはローカルの UNIX ファイルベースの認証が含まれます。

その他のサービスやアプリケーションも利用できますが、これらは一般的な設定です。

パート I. システムログイン

このパートでは、**authconfig** ツール、**ipa-client-install** ツール、および **realmd** ツールを使用してシステム認証を設定する方法を説明します。

第2章 システム認証の設定

認証 は、ユーザーが特定され、システムに対して検証されるプロセスです。ユーザー名とパスワードなどの ID と認証情報を指定する必要があります。次に、システムは設定済みの認証サービスに対して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アクティブであれば、ユーザーは *認証*されます。

ユーザーが認証されると、情報はアクセス制御サービスに渡され、何がユーザーに許可されているかを判断します。これらは、ユーザーのアクセスが *許可されている* リソースです。認証と認可は 2 つの異なるプロセスであることに注意してください。

システムは、ユーザー認証を確認するために、有効なアカウントデータベースの一覧を設定している。ユーザーがローカルシステムに配置できるか、ローカルシステムが LDAP や Kerberos などのリモートシステムのユーザーデータベースを参照できます。ローカルシステムは、ユーザー情報にさまざまなデータストアを使用できます。たとえば、Lightweight Directory Access Protocol (LDAP)、ネットワーク情報サービス (NIS)、および Winbind があります。LDAP および NIS のデータストアは、Kerberos を使用し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部である可能性があるため、Red Hat Enterprise Linux は System Security Services Daemon (SSSD) を中央デーモンとして使用して、さまざまな ID バックエンドに対してユーザーを認証したり、ユーザーに対してチケット許可チケット (TGT) を要求したりできます。SSSD は、LDAP、Kerberos、および外部アプリケーションと対話し、ユーザーの認証情報を確認できます。

本章では、システム認証を設定するために Red Hat Enterprise Linux で利用可能なツールについて説明します。

- Identity Management システム用の **ipa-client-install** ユーティリティーおよび **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。
- **authconfig** ユーティリティーと、他のシステムの 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

ipa-client-install ユーティリティーおよび **realmd** システムを使用して、Identity Management マシンでシステム認証を自動的に設定できます。

ipa-client-install

ipa-client-install ユーティリティーは、Identity Management ドメインに参加するようにシステムを設定します。**ipa-client-install** の詳細は、『Linux ドメイン ID、認証、およびポリシーガイドの クライアントの』 インストール を参照してください。

Identity Management システムの場合、**ipa-client-install** は **realmd** よりも優先されることに注意してください。

realmd

realmd システムは、マシンを Identity Management や Active Directory ドメインなどの ID ドメインに参加させます。**realmd** の詳細は、『Windows Integration Guide』 の Using realmd to Connect to an Active Directory Domain を参照してください。

2.2. AUTHCONFIGの使用

authconfig ツールは、LDAP などのユーザー認証情報に使用するデータストアの種類を設定できます。Red Hat Enterprise Linux では、**authconfig** には GUI とコマンドラインオプションの両方があり、ユーザーデータストアを設定できます。**authconfig** ツールは、異なる形式の認証メカニズムを使用するとともに、ユーザーデータベースに特定のサービス(SSSD、LDAP、NIS、または Winbind)を使用するようにシステムを設定できます。

重要

Identity Management システムを設定するには、Red Hat は、**authconfig** の代わりに **ipa-client-install** ユーティリティーまたは **realmd** システムを使用することを推奨します。**authconfig** ユーティリティーは制限されており、柔軟性は低くなります。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。

認証設定には、以下の 3 つの **authconfig** ユーティリティーを使用できます。

- **authconfig-gtk** は完全なグラフィカルインターフェイスを提供します。
- **authconfig** は、手動設定用のコマンドラインインターフェイスを提供します。
- **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ーは非推奨となっています。

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

2.2.1. authconfig CLI の使用に関するヒント

authconfig コマンドラインツールは、スクリプトに渡された設定に従って、システム認証に必要なすべての設定ファイルとサービスを更新します。UI を介して設定できる以上の ID および認証設定オプションを提供するとともに、**authconfig** ツールを使用してバックアップファイルおよびキックスタートファイルを作成することもできます。

authconfig オプションの完全なリストは、help 出力と man ページを参照してください。

authconfig を実行する際に注意すべき事項があります。

- すべてのコマンドで、**--update** オプションまたは **--test** オプションのいずれかを使用します。コマンドを正常に実行するには、以下のオプションのいずれかが必要です。**--update** を使用すると、設定の変更が書き込まれます。**--test** オプションは変更を表示しますが、設定に変更を適用しません。

  --update オプションを使用しない場合、変更はシステム設定ファイルには書き込まれません。

- コマンドラインを使用して、既存の設定を更新したり、新しい設定を設定するのに使用することもできます。このため、コマンドラインは、必要な属性が特定の呼び出しで使用されることを強制しません (コマンドが更新されている可能性があるためで、そうでなければ設定を完了します)。

  認証設定の編集時に、設定が完了して正確である点に留意してください。認証設定を不完全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される可能性があります。--test オプションを使用して、--update オプションを使用して書き込む前に設定が適切であることを確認します。

- 有効化オプションには、それぞれ対応する無効化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ていませんが、管理者が認証設定にすばやく変更を加えると便利です。

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。これには、**authconfig** コマンドラインツール、Bash、Python などの一般的なシステムパッケージ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ています。

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

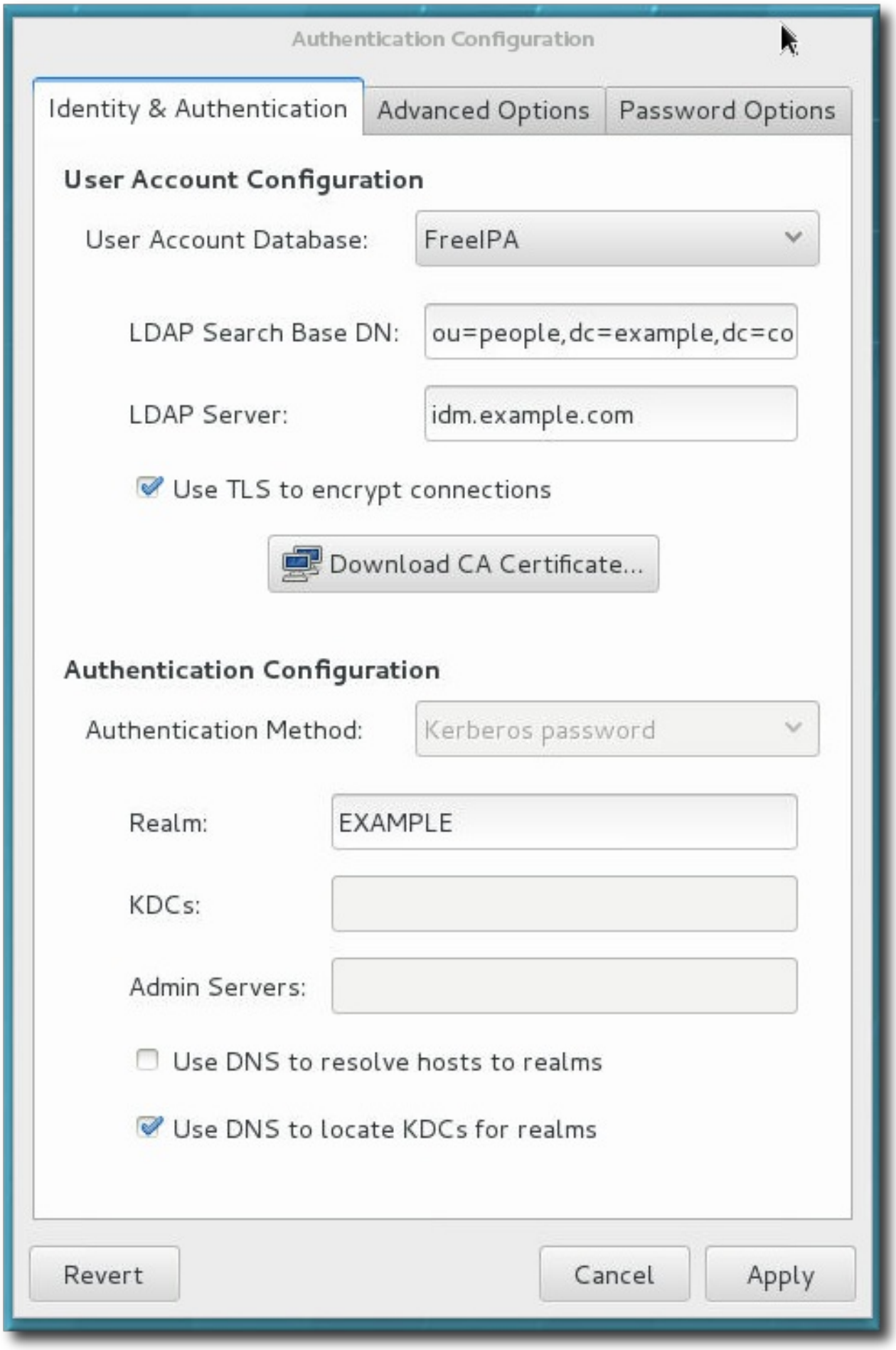
重要

authconfig UI が閉じられると、変更がすぐに有効になります。

Authentication ダイアログボックスには、設定タブが 3 つあります。

- **ID と認証**。アイデンティティーストアとして使用するリソースを設定します（ユーザー ID と対応する認証情報が保存されるデータリポジトリー）。
- **高度なオプション**。スマートカードやフィンガープリントなどのパスワードや証明書以外の認証方法を設定します。
- **パスワードオプション**。パスワード認証方法を設定します。

図2.1 認証設定ウィンドウ



2.2.4. 認証設定のテスト

認証が完全かつ適切に設定されていることが重要です。そうしないと、すべてのユーザーが (root でも) ロックアウトされたり、一部のユーザーがブロックされたりする可能性があります。

--test オプションは、考えられるすべての ID および認証メカニズムに対して、システムのすべての認証設定を出力します。これにより、有効になっている領域と無効になっている領域の両方の設定が表示されます。

test オプションを単独で実行して現在の設定をすべて表示することもできます。または、**authconfig** コマンドで使用して、（実際に設定を変更せずに） **設定を変更する方法を表示することもでき** ます。これは、提案された認証設定が完了し、正確であることの確認に非常に便利です。

```
[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および復元

認証設定の変更は問題になる可能性があります。設定を誤って変更しないと、アクセスする必要があるユーザーが誤って除外され、アイデンティティーストアへの接続が失敗するか、システムへのアクセスがすべてロックされる可能性があります。

認証設定を編集する前に、管理者がすべての設定ファイルのバックアップを作成することを強く推奨します。これは、**--savebackup** オプションを使用して行います。

```
[root@server ~]# authconfig --savebackup=/backups/authconfigbackup20200701
```

認証設定は、**--restorebackup** オプションを使用して使用するバックアップ名を指定し、以前に保存したすべてのバージョンに復元できます。

```
[root@server ~]# authconfig --restorebackup=/backups/authconfigbackup20200701
```

authconfig コマンドは、設定を変更するたびに自動バックアップを保存します。**--restorelastbackup** オプションを使用して、最後のバックアップを復元できます。

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

第3章 AUTHCONFIGで認証用のアイデンティティーストアの選択

authconfig UI の **Identity & Authentication** タブは、ユーザーを認証する方法を設定します。デフォルトでは、ローカルシステム認証を使用します。つまり、ユーザーとパスワードは、ローカルシステムのアカウントに対して確認されます。Red Hat Enterprise Linux マシンは、LDAP、NIS、Winbind などのユーザーおよび認証情報を含む外部リソースを使用することもできます。

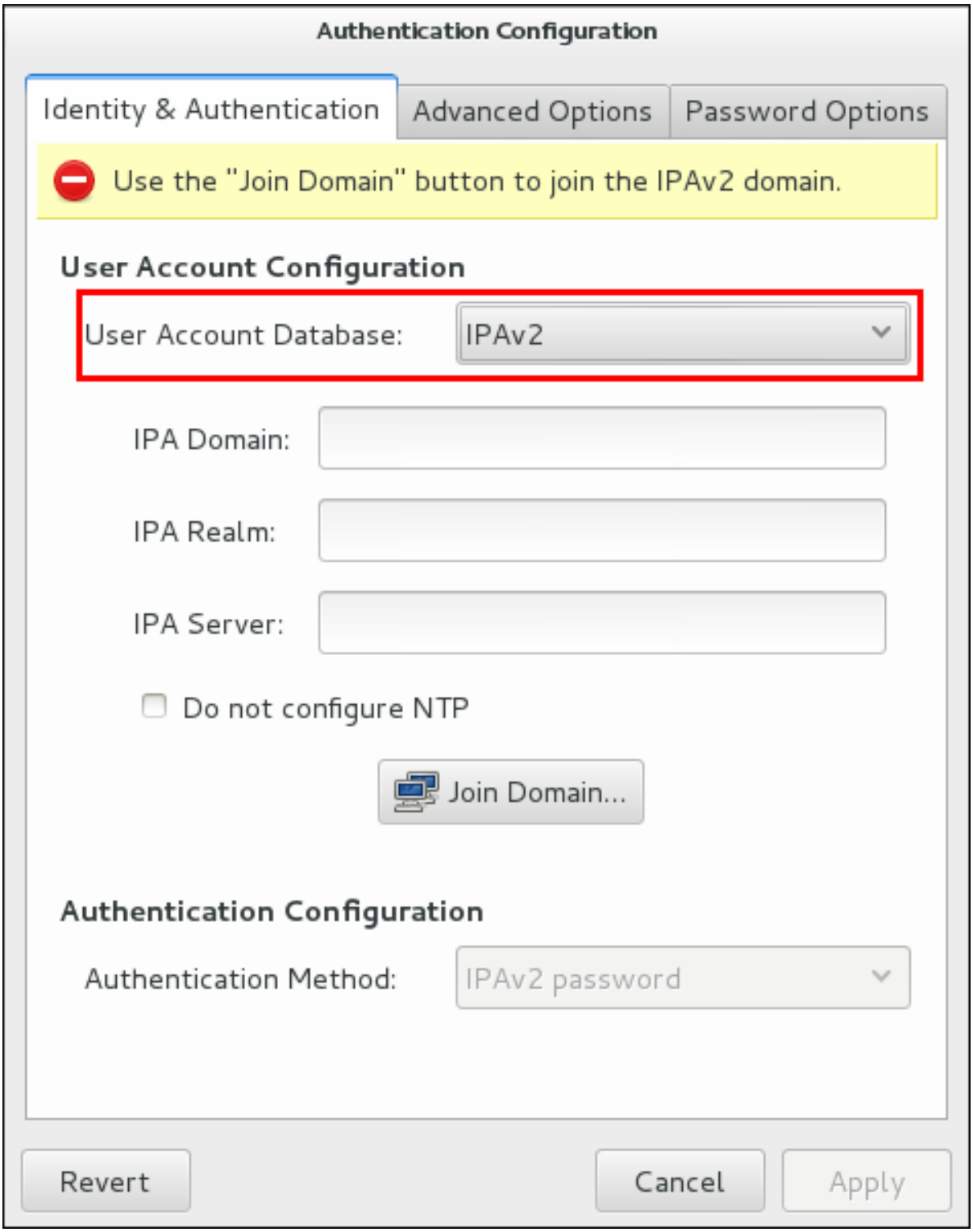
3.1. IPAV2

Identity Management サーバーをアイデンティティーバックエンドとして設定する方法は 2 つあります。IdM 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)の場合は、**authconfig** で IPAv2 プロバイダーとして設定されます。以前の IdM バージョンおよびコミュニティー FreeIPA サーバーでは、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. **User Account Database** ドロップダウンメニューで **IPAv2** を選択します。

図3.1 authentication-configuration



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** は、IdM ドメインの DNS ドメインを提供します。
   - **IPA レルム** は、IdM ドメインの Kerberos ドメインを提供します。
   - **IPA Server** は、IdM ドメイントポロジー内の IdM サーバーのホスト名を指定します。
   - **NTP を設定しないでください。**必要に応じて、クライアントの設定時に NTP サービスを無効にします。IdM サーバーおよびすべてのクライアントが Kerberos 認証および証明書を正しく機能させるには、すべてのクライアントがクロックを同期している必要があるため、

通常は推奨されません。これは、IdM サーバーがドメイン内でホストするのではなく、別の NTP サーバーを使用している場合に無効にできます。

4. **ドメインへの参加** ボタンをクリックします。

これにより **ipa-client-install** コマンドが実行され、必要に応じて **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な全システムファイルを自動的に設定し、ドメイン情報を更新するためにドメインサーバーに問い合わせます。

3.1.2. コマンドラインでの IdM の設定

IdM ドメインは、DNS や Kerberos などの 1 つの階層で、一般的なサービスと重要なサービスを一元管理します。

authconfig (*8章 **realmd** を使用した Identity ドメインへの接続* の **realmd** など)を使用して、IdM ドメインにシステムを登録することができます。これにより **ipa-client-install** コマンドを実行すると、必要に応じて **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な全システムファイルを自動的に設定し、ドメイン情報を更新するためにドメインサーバーに問い合わせます。

ドメインに参加するには、DNS ドメイン名 (**--ipav2domain**)、Kerberos レルム名 (**--ipav2realm**)、および接続する IdM サーバー (**--ipav2server**) の 3 つの情報が必要です。**--ipav2join** オプションは、IdM サーバーへの接続に使用する管理者ユーザー名を指定します。通常、これは **admin** です。

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

IdM ドメインが NTP サービスを実行していない場合は、**--disableipav2nontp** オプションを使用して、設定スクリプトが NTP サーバーとして IdM サーバーを使用しないようにすることができます。IdM サーバーおよびすべてのクライアントが Kerberos 認証および証明書を正しく機能させるには、すべてのクライアントがクロックを同期している必要があるため、通常は推奨されません。

3.2. LDAP および IDM

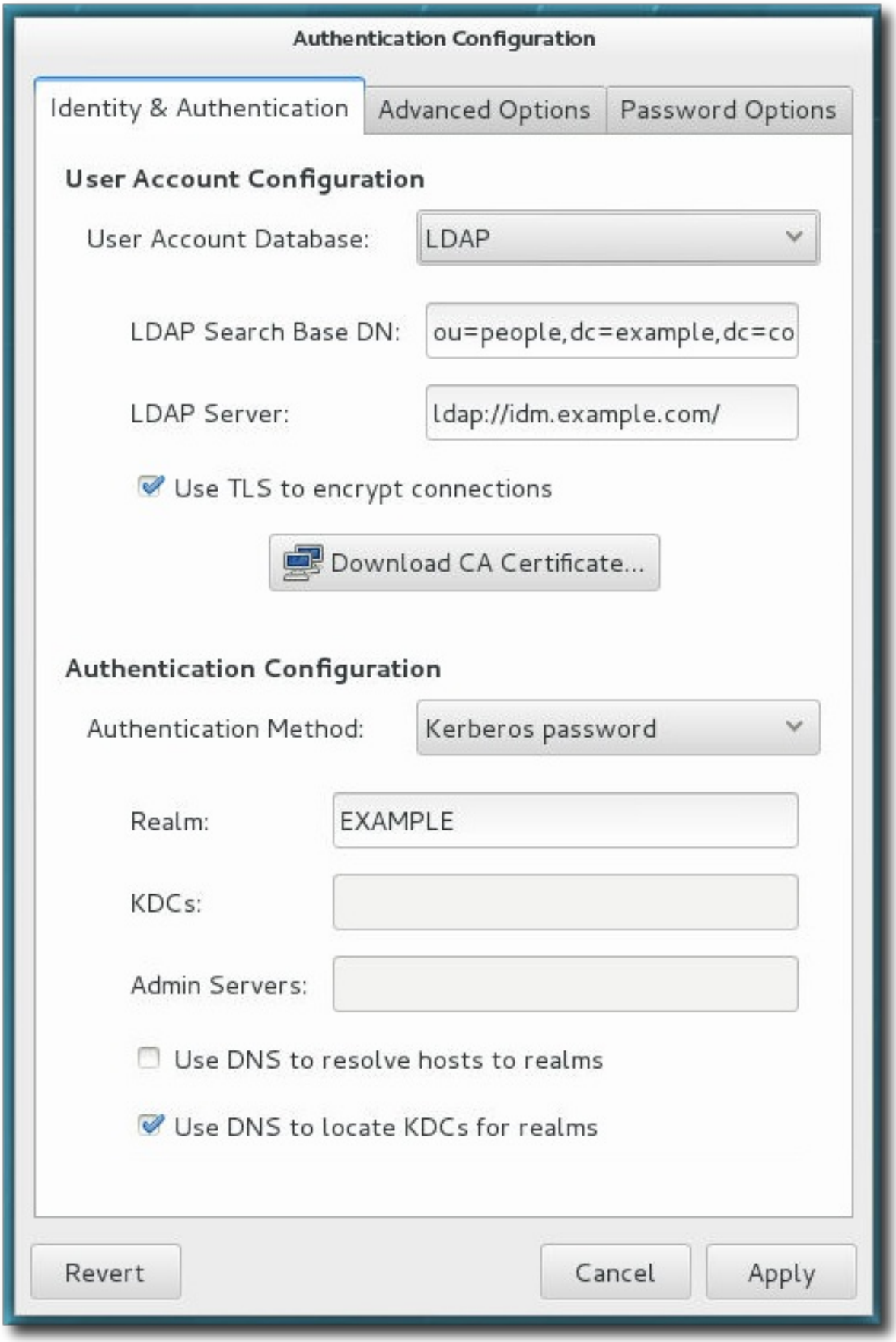
標準の LDAP ディレクトリー (OpenLDAP や Red Hat Directory Server など) の両方が LDAP アイデンティティープロバイダーとして使用できます。さらに、古い IdM バージョンと FreeIPA は、関連する Kerberos サーバーで LDAP プロバイダーとして設定することで、アイデンティティープロバイダーとして設定できます。

openldap-clients パッケージまたは sssd パッケージは、ユーザーデータベースの LDAP サーバーを設定するのに使用されます。デフォルトでは、両方のパッケージがデフォルトでインストールされます。

3.2.1. UI での LDAP 認証の設定

1. 「authconfig UI の起動」 のように、**authconfig** UI を開きます。

2. **User Account Database** ドロップダウンメニューで **LDAP** を選択します。



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** は、ユーザーディレクトリーのルート接尾辞または **識別名** (DN)を提供します。アイデンティティーまたは認証に使用されるユーザーエントリーはすべて、この親エントリーの下に存在します。たとえば、**ou=people,dc=example,dc=com** です。

このフィールドは任意です。これが指定されていない場合、System Security Services Daemon (SSSD)は LDAP サーバーの設定エントリーの ***namingContexts*** および ***defaultNamingContext*** 属性を使用して検索ベースを検出しようとします。

- **LDAP サーバー** は LDAP サーバーの URL を提供します。これには通常、**ldap://ldap.example.com:389** など、LDAP サーバーのホスト名とポート番号の両方が必要です。

  ldaps:// で始まる URL を使用してセキュアなプロトコルを入力すると、**Download CA Certificate** ボタンが有効になります。これにより、発行した認証局から LDAP サーバーの発行元 CA 証明書を取得します。CA 証明書は、プライバシーにより強化されたメール (PEM) 形式である必要があります。

- 非セキュアな標準ポート接続(**ldap://**で始まる URL)を使用する場合は、**Use TLS to encrypt connections** チェックボックスを使用し、**STARTTLS** を使用して LDAP サーバーとの通信を暗号化できます。このチェックボックスを選択すると、**Download CA Certificate** ボタンも有効になります。

注記

通信がすでに **暗号化されているため、サーバー URL が LDAPS (LDAP over SSL)セキュアプロトコルを使用している場合は、TLS を使用して接続** を暗号化する チェックボックスを選択する必要はありません。

4. 認証方法を選択します。LDAP は、簡単なパスワード認証または Kerberos 認証を許可します。

   Kerberos の使用については、「UI での Kerberos 認証の設定」 を参照してください。

   LDAP パスワード オプションは、PAM アプリケーションを使用して LDAP 認証を使用します。このオプションでは、LDAPS または TLS を使用して LDAP サーバーへ接続し、セキュアな接続を設定する必要があります。

3.2.2. コマンドラインでの LDAP ユーザーストアの設定

LDAP アイデンティティー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場合は、**--enableldapauth** を使用して、LDAP サーバー名、ユーザー接尾辞のベース DN、TLS を使用するかどうか (オプション) など、必要な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が、**authconfig** UI ではできません。

プロトコル(**ldap** または **ldaps**)およびポート番号を含む完全な LDAP URL を必ず使用してください。**--enableldaptls** オプションでセキュアな LDAP URL (**ldaps**)を使用し **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 パスワード認証に **--ldapauth** を使用する代わりに、LDAP ユーザーストアで Kerberos を使用できます。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

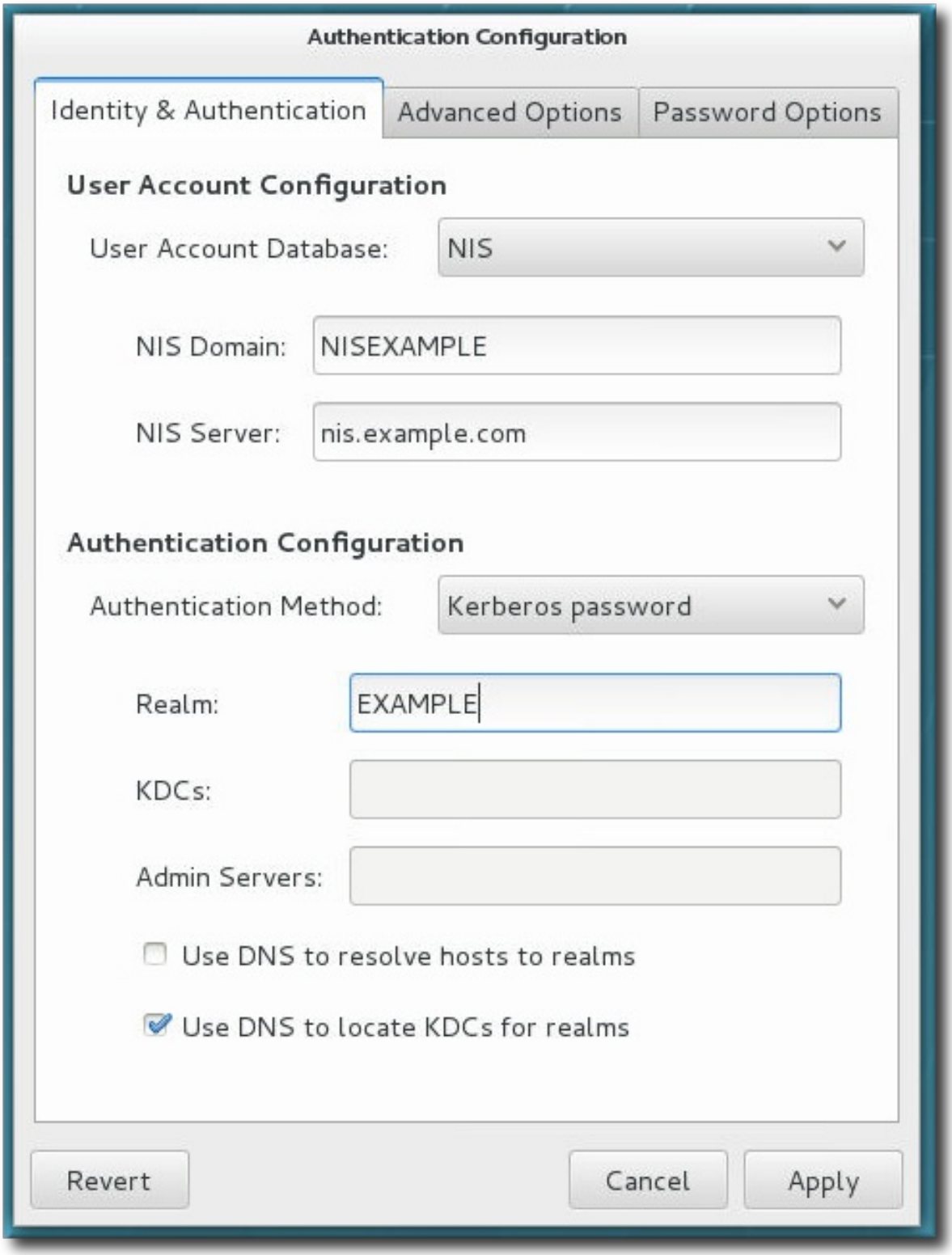
重要

NIS をアイデンティティーストアとして設定する前に、NIS 自体を環境用に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が設定されている状態で完全に設定する必要があります。
- **ypbind** パッケージがローカルシステムにインストールされている。これは NIS サービスに必要ですが、デフォルトではインストールされません。
- **portmap** サービスおよび **ypbind** サービスは起動時に開始され、起動できるようになります。これは、**ypbind** パッケージのインストールの一部として設定する必要があります。

3.3.1. UI での NIS 認証の設定

1. 「authconfig UI の起動」 のように、**authconfig** UI を開きます。
2. **User Account Database** ドロップダウンメニューで **NIS** を選択します。



3. NIS サーバーに接続する情報を設定します。つまり、NIS ドメイン名およびサーバーホスト名になります。NIS サーバーが指定されていない場合、**authconfig** デーモンは NIS サーバーをスキャンします。

4. 認証方法を選択します。NIS は、簡単なパスワード認証または Kerberos 認証を許可します。

   Kerberos の使用については、「UI での Kerberos 認証の設定」 を参照してください。

3.3.2. コマンドラインでの NIS の設定

NIS アイデンティティーストアを使用するには、**--enablenis** を使用します。Kerberos パラメーターが

明示的に設定されていない場合 (「コマンドラインでの Kerberos 認証の設定」) を除き、これは NIS 認証を自動的に使用します。NIS サーバーと NIS ドメインを識別するための唯一のパラメーターは です。これを使用しないと、**authconfig** サービスは NIS サーバーのネットワークをスキャンします。

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

3.4. WINBIND

Winbind をシステムのアイデンティティー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 サーバーをセットアップしてユーザーアカウントに使用するか、Active Directory をバックエンドの ID ストアとして使用するように Samba を設定する必要があります。

Samba の設定は、Samba プロジェクトのドキュメント で説明されています。Active Directory との統合ポイントとして Samba を設定する方法は『Windows Integration Guide』のUsing Samba for Active Directory Integrationでも説明されています。

3.4.1. authconfig GUI での Winbind の有効化

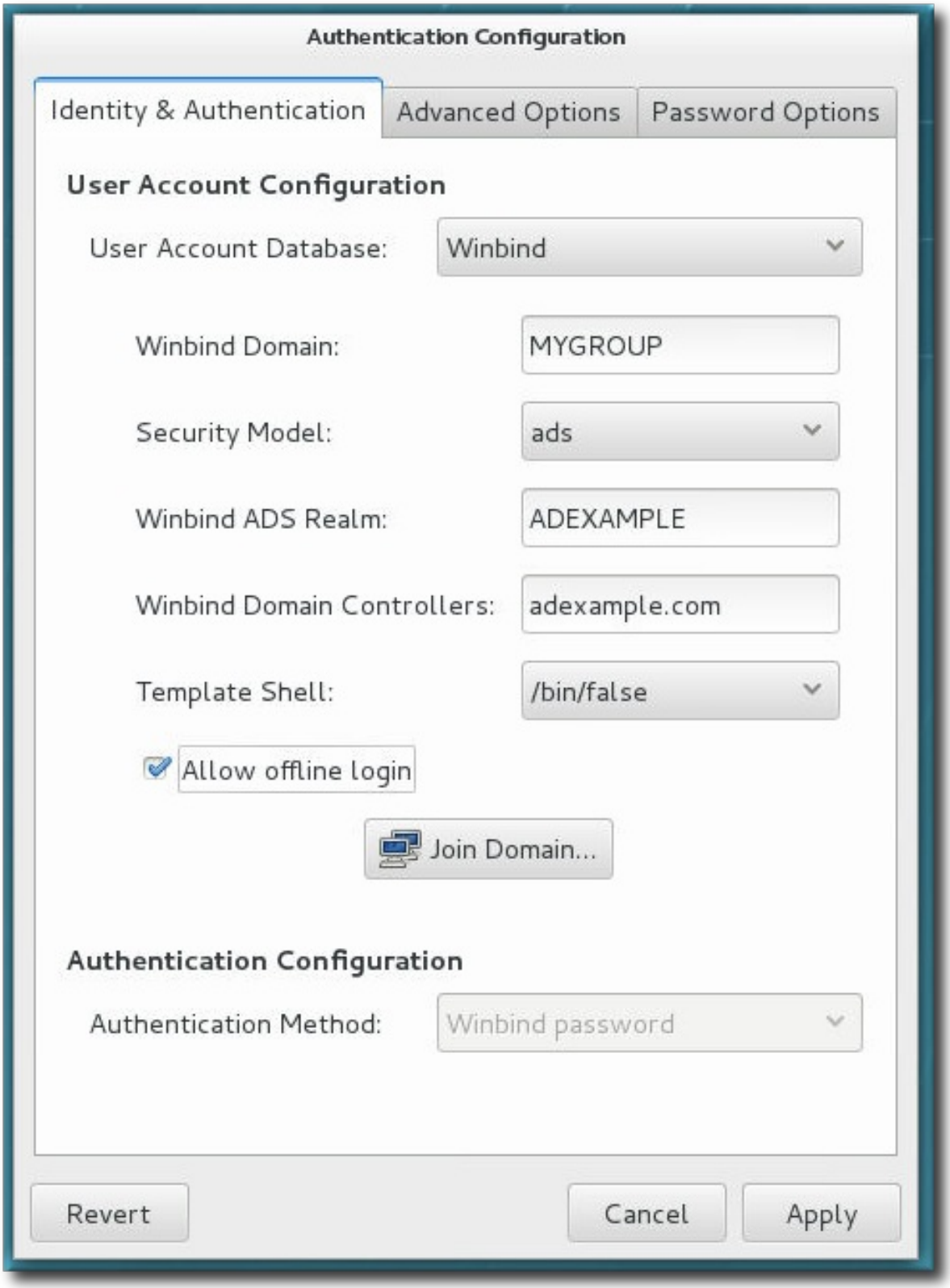
1. **samba-winbind** パッケージをインストールします。これは、Samba サービスの Windows 統合機能に必要ですが、デフォルトではインストールされません。

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

2. **authconfig** UI を開きます。

   ```
   [root2server ~]# authconfig-gtk
   ```

3. アイデンティティーおよび **認証** タブで、**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。



4. Microsoft Active Directory ドメインコントローラーへの接続に必要な情報を設定します。

   - **Winbind ドメイン** は、接続する Windows ドメインを提供します。

     これは、**DOMAIN** などの Windows 2000 形式である必要があります。

   - **セキュリティーモデル 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 種類のセキュリティーモデルをサポートします。

- ADS **は、** Samba が Active Directory Server 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** パッケージをインストールし、Kerberos を適切に設定する必要があります。

- **ドメイン** には、Windows サーバーと同様に、Windows のプライマリーまたはバックアップドメインコントローラーを使用して認証することで、Samba がユーザー名とパスワードを検証します。

- **サーバー** には、Windows サーバーなどの別のサーバーで認証することで、ローカルの Samba サーバーがユーザー名とパスワードを検証します。サーバーの認証を試みると、システムは **ユーザー** モードを使用した認証を試行します。

- **ユーザー** は、有効なユーザー名とパスワードでクライアントに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に対応します。

  ユーザー名の形式は **domain\user** である必要があります（例： **EXAMPLE\jsmith** ）。

注記

特定のユーザーが Windows ドメインに存在することを確認する場合は、**domain\user_name** 形式を使用してバックスラッシュ(\)文字をエスケープしてください。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** は、Samba サーバーが参加する Active Directory レルムを提供します。これは、**ads** セキュリティーモデルでのみ使用されます。

- **Winbind ドメインコントローラー** は、システムの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 アドレスを提供します。

- **Template Shell** は、Windows 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** すると、認証情報をローカルキャッシュに保存できます。システムがオフライン時に、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。

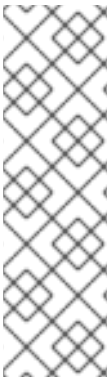
3.4.2. コマンドラインでの Winbind の有効化

Windows ドメインには複数のセキュリティーモデルがあり、ドメインで使用されるセキュリティーモデルによってローカルシステムの認証設定が決定します。ユーザーおよびサーバーのセキュリティーモデルの場合、Winbind の設定にはドメイン (またはワークグループ) 名とドメインコントローラーのホスト名のみが必要です。

--winbindjoin パラメーターは、Active Directory ドメインへの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を設定して **/etc/passwd** ファイルを確認します。

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 **domain\user** である必要があります（例： **EXAMPLE\jsmith** ）。

Windows ドメインに特定のユーザーが存在することを確認する場合は、常に **domain\user** 形式を使用し、バックスラッシュ (\) をエスケープしてください。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 および domain セキュリティーモデルの場合、Winbind は、テンプレートシェルおよびレルム (ads のみ) への追加の設定を可能にします。以下に例を示します。

```
[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の情報を設定するためのオプションは他にもたくさんあります。たとえば、名前の形式、ユーザー名とともにドメイン名を要求するかどうか、UID の範囲などです。これらのオプションは **authconfig** ヘルプに一覧表示されます。

第4章 認証メカニズムの設定

Red Hat Enterprise Linux は、複数の認証方法に対応します。**authconfig** ツールを使用して設定することも、場合によっては Identity Management ツールを使用して設定することもできます。

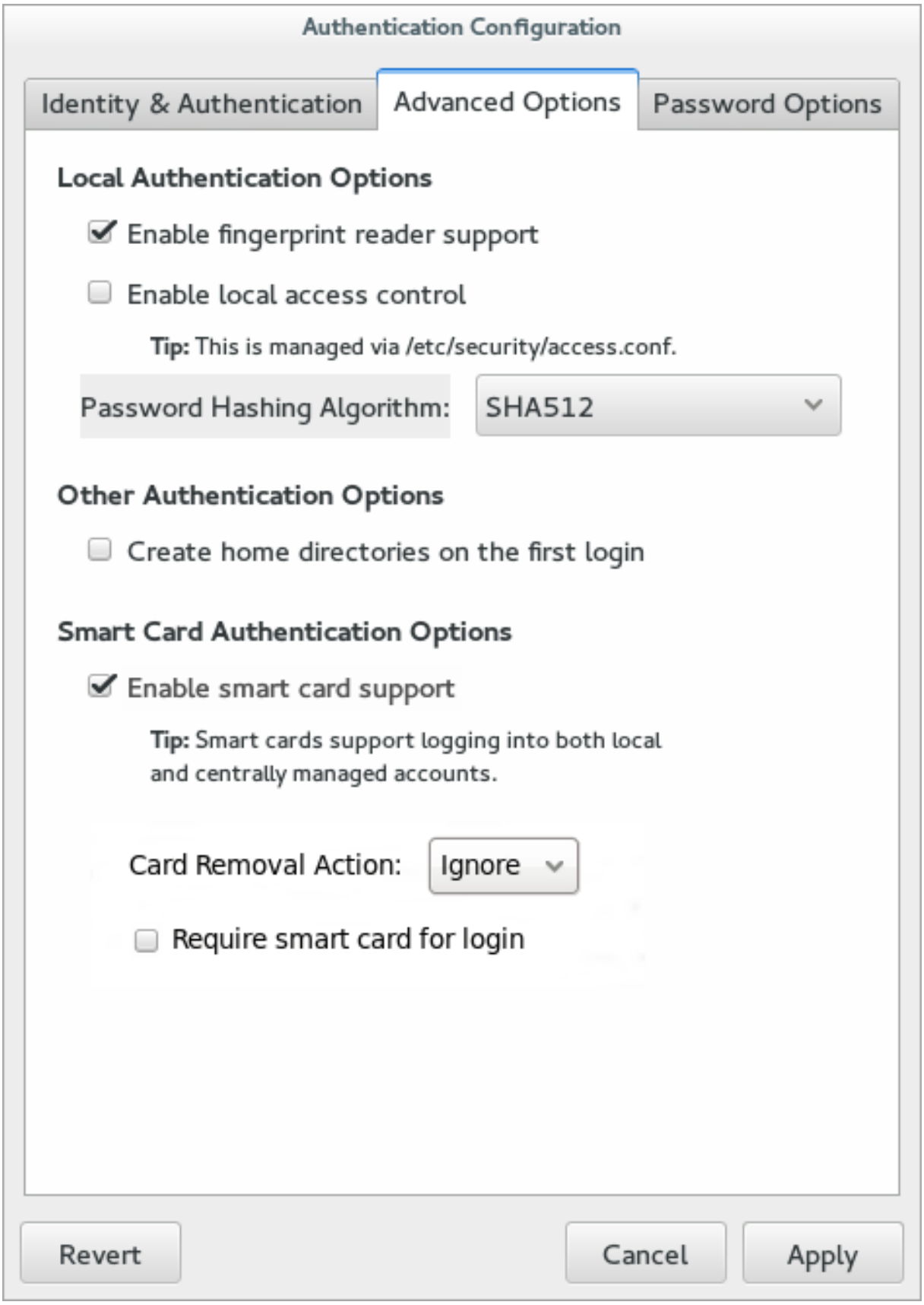
4.1. AUTHCONFIGを使用したローカル認証の設定

ローカル **認証オプション** エリアは、バックエンドに保存されているユーザーではなく、ローカルシステムアカウントの設定を定義します。この設定により、(**/etc/security/access.conf**で定義されているように)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。それ以外の場合は、認証ポリシーはアイデンティティープロバイダーまたはサービス自体内で定義できます。

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 にすると、ローカルユーザー認可ルールの **/etc/security/access.conf** ファイルを確認するようにシステムが設定されます。これは PAM 認証です。

図4.1 ローカルアカウントフィールド



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 には、ローカル承認制御を有効にするオプションが 2 つあります。**--enablelocauthorize** はネットワーク認証を省略し、システムユーザーの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** でシステム認可ポリシーを検索するように設定します。

```
[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、または改ざんに対して脆弱になります。これを防ぐため、暗号ハッシュアルゴリズムを使用してパスワードハッシュダイジェストをセキュアに保存できます。IdM でサポートされている推奨 (およびデフォルト) のハッシュアルゴリズムは SHA-512 です。これは、64 ビットのワードを使用し、セキュリティーを強化するためにソルトとストレッチも使用します。後方互換性を確保するために、SHA-256、DES、BigCrypt、および MD5 ハッシュアルゴリズムもサポートされます。

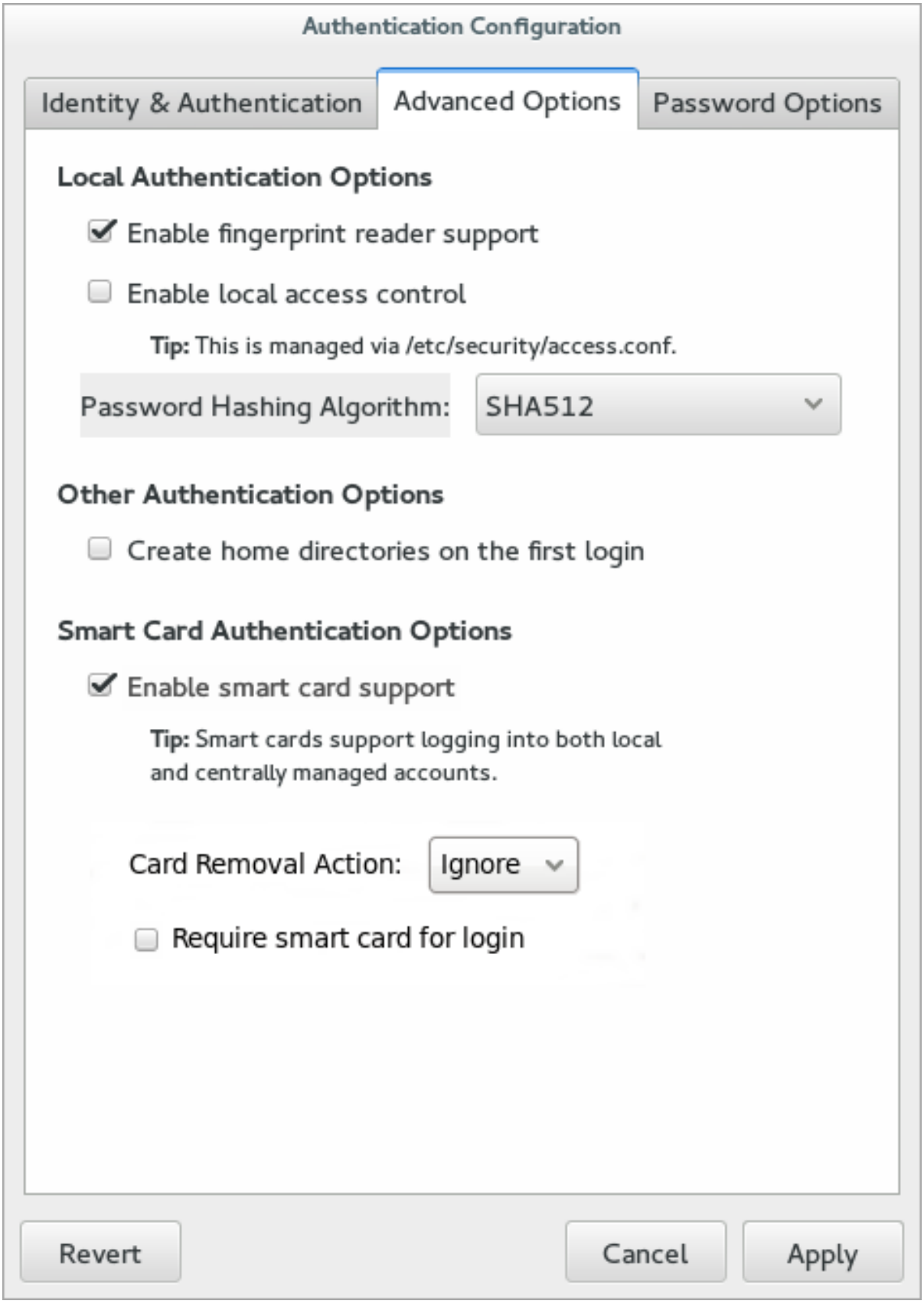
重要

下位互換性が必要ない場合は、より安全な SHA-512 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

Local Authentication Options タブは、ローカルパスワードがシステムに保存される方法を設定します。パスワードハッシュ **アルゴリズム ドロップダウン** メニューは、パスワードハッシュをセキュアに保存するようにアルゴリズムを設定します。

1. 「authconfig UI の起動」 のように、**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 ドロップダウンメニューで、使用するアルゴリズムを選択します。



4. **Apply** 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の設定

ユーザー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または変更するには、**--passalgo** オプションとアルゴリズムの短縮名を使用します。以下の例では、SHA-512 アルゴリズムを使用しています。

-

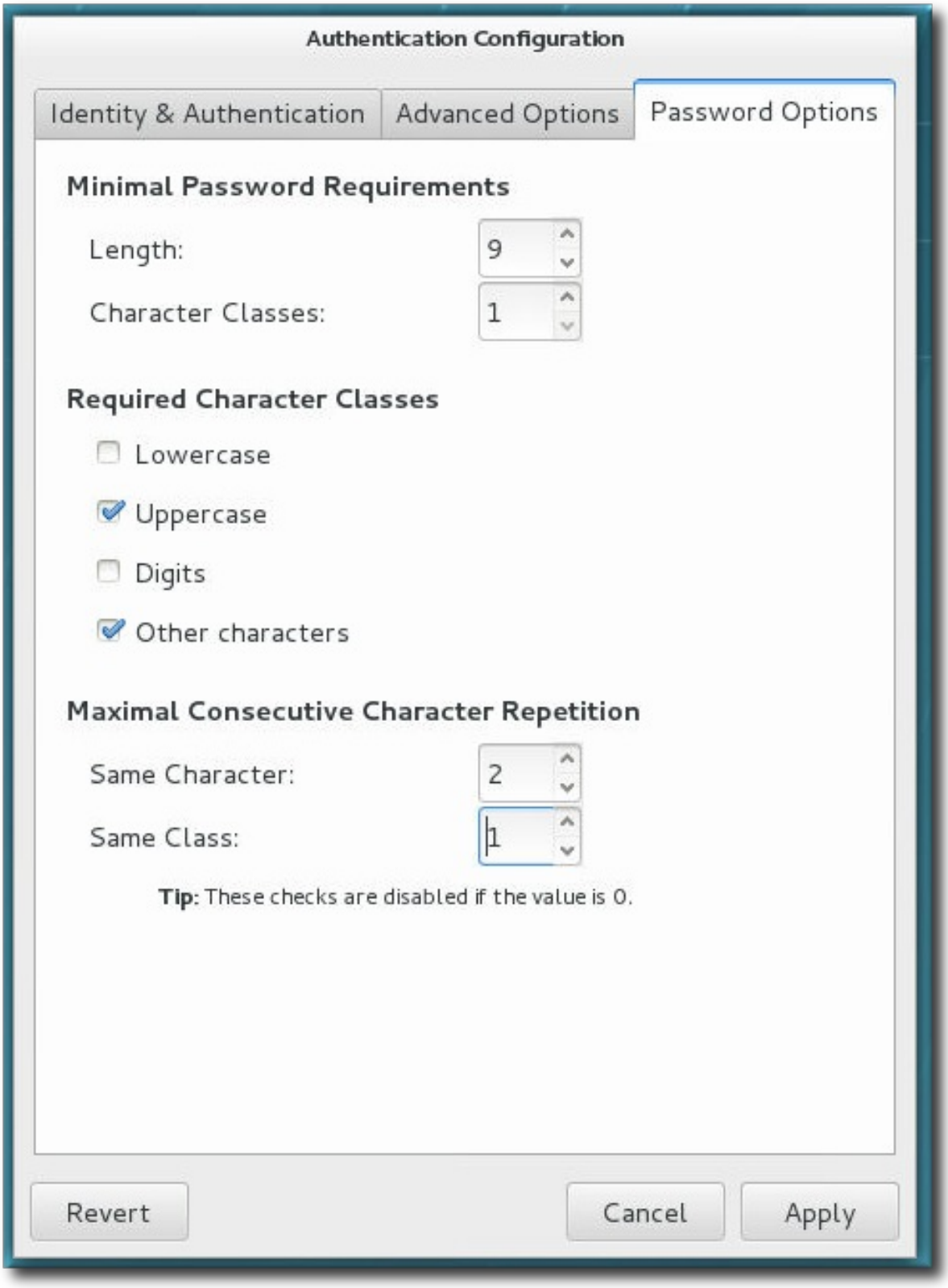
```
[root@server ~]# authconfig --passalgo=sha512 --update
```

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に対してパスワードを設定する方法を設定します。複雑さは、長さと文字クラスのバリエーションの組み合わせです。これを確認する 1 つの方法は、複雑なパスワードのポリシーを設定するには、2 つの要素があるということです。パスワードに使用できる文字の種類 (大文字、小文字、特殊文字など) を特定することと、この文字をパスワードでどのように使用するか (どのくらいの長さが必要か、どのくらいの頻度で文字を繰り返すことができるかを特定することです。

4.2.2.1. UI でのパスワードの複雑性の設定

1. 「authconfig UI の起動」 のように、**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。



3. パスワードの **最小長** を設定します。
   - パスワードの最小長
   - パスワードで使用する必要のある文字クラスの最小数。
4. パスワードに使用する **必要のある** 文字クラスを有効にします。たとえば、任意のパスワードで大文字を使用できますが、**Uppercase** チェックボックスを選択した場合は、すべてのパスワードで大文字を使用する必要があります。

5. 文字または文字クラスが連続して繰り返すことができる回数を設定します。(ゼロに設定すると、制限が繰り返されません。)

   同じ文字 フィールドの場合、1文字または文字を繰り返す頻度を設定します。たとえば、これを 2 に設定すると、**ssecret** は使用できますが、**sssecret** は使用できません。

   同様に、**同じクラス** は、文字クラス（大文字、番号、特殊文字）から文字を繰り返すことができる回数に制限を設定します。これを 3 に設定すると、たとえば **secret!!** は使用できますが、**secret!!@** または **secret1234** は使用できません。

6. **Apply** 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

コメント行でパスワードの複雑性を定義する場合、要件の設定には 2 つの部分があります。1つ目は、パスワードの作成方法に関する要件を設定することです。パスワードの長さ、文字を繰り返すことができるか、使用する必要がある文字の種類です。

- 最小長 (**--passminlen**)。
- 使用する必要のあるさまざまなタイプの文字の最小数 (**--passminclass**)。
- 文字が連続して繰り返すことができる回数 (**--passmaxrepeat**)。このパラメーターをゼロに設定すると、繰り返し制限がなくなります。
- (数字など) 同じタイプの文字が1行で使用できる回数 (**--passmaxclassrepeat**)。このパラメーターをゼロに設定すると、繰り返し制限がなくなります。

もう1つは、パスワードに使用できるタイプまたは文字クラスを定義します。すべての文字タイプは暗黙的に許可されます。**--enablereqType** オプションを使用すると、指定したクラスが絶対に必要であるか、パスワードが拒否されます。(逆に、型を明示的に拒否することもできます))

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

たとえば、これは 9 文字の最小長を設定し、文字またはクラスを 2 回以上繰り返すことを許可せず、大文字と特殊文字の両方を必要とします。

```
[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 を使用)の設定

LDAP および NIS の認証ストアは、Kerberos 認証方法をサポートします。Kerberos の使用には、以下のような利点があります。

- これは、標準ポートを介した接続を許可する一方で、通信にセキュリティーレイヤーを使用します。
- SSSD で認証情報キャッシュを自動的に使用します。これにより、オフラインログインが可能になります。

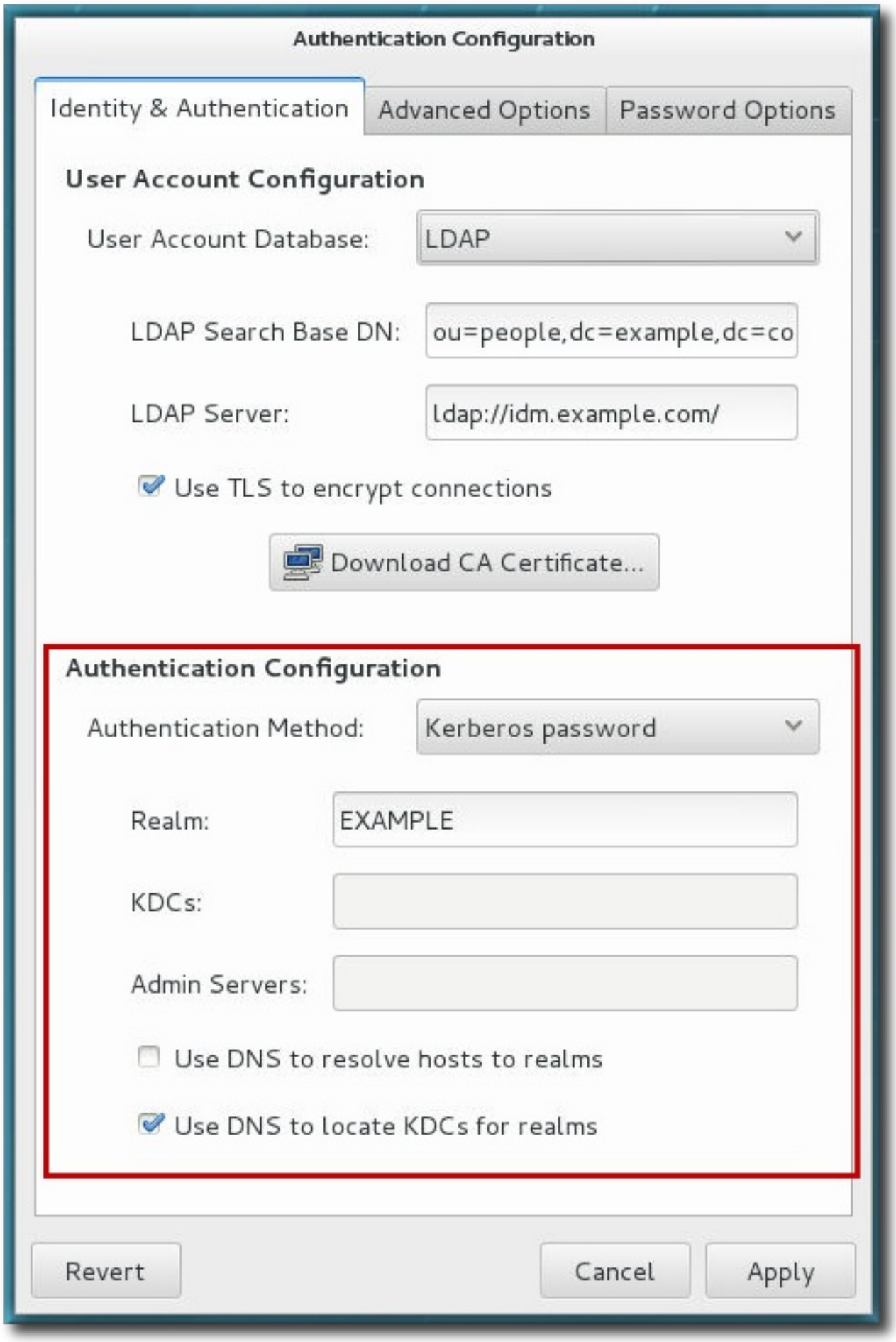
注記

Kerberos 認証を使用するには、**krb5-libs** パッケージおよび **krb5-workstation** パッケージが必要です。

4.3.1. UI での Kerberos 認証の設定

Authentication Method ドロップダウンメニューの **Kerberos password** オプションは、Kerberos レルムへの接続に必要なフィールドを自動的に開きます。

図4.2 Kerberos フィールド



- **realm** は、Kerberos サーバーのレルムの名前を指定します。レルムは、Kerberos を使用するネットワークで、1 つ以上の **キー配布センター** (KDC) と、潜在的に多数のクライアントで設定されます。
- **KDC は**、Kerberos チケットを発行するサーバーのコンマ区切りリストを提供します。
- **管理サーバー に** は、レルムで **kadmind** プロセスを実行している管理サーバーのリストが表示されます。
- 必要に応じて、DNS を使用してサーバーのホスト名を解決し、レルム内で追加の KDC を見つけます。

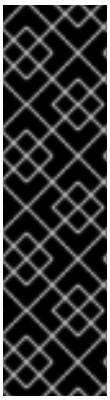
4.3.2. コマンドラインでの Kerberos 認証の設定

LDAP および NIS の両方で、Kerberos 認証をネイティブ認証メカニズムの代わりに使用できます。少なくとも、Kerberos 認証を使用するには、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。また、DNS を使用してクライアント名を解決し、追加の管理サーバーを検索す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

スマートカードに基づいた認証は、パスワードベースの認証の代替手段です。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 コードを指定する必要があります。



重要

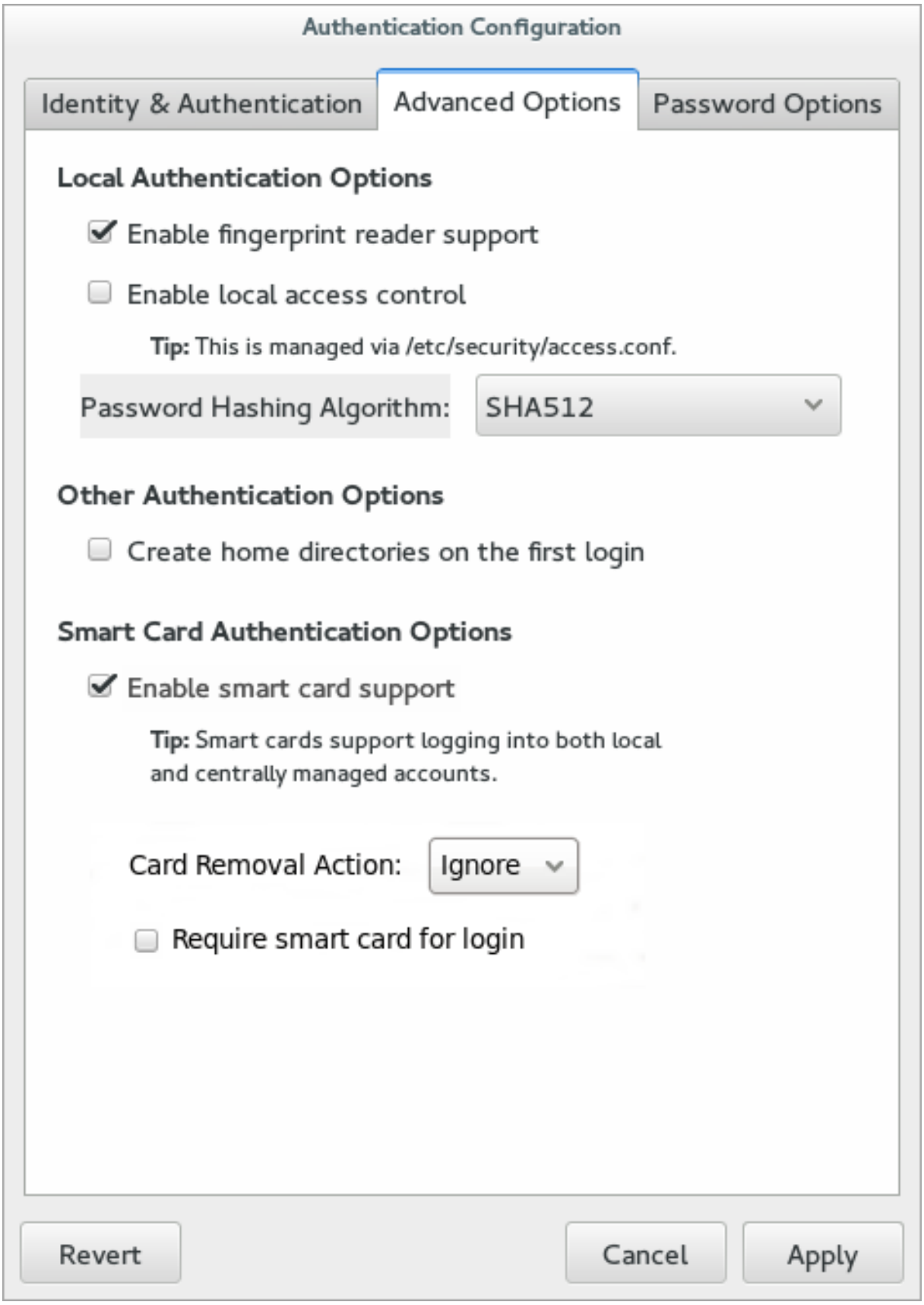
次のセクションでは、pam_pkcs11 パッケージおよび pam_krb5 パッケージを使用して、ローカルユーザーでスマートカード認証に単一のシステムを設定する方法を説明します。これらのパッケージは、『7.4 リリースノート』 の 非推奨の機能 で説明されているように、非推奨になりました。

スマートカード認証を一元的に設定するには、System Security Services Daemon(SSSD) が提供する拡張スマートカード機能を使用します。詳細は、『Linux Domain Identity, Authentication, and Policy Guide』のSmart-card Authentication in Identity Management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

Enable smart card support オプションを選択すると、スマートカードの動作を設定するための追加の制御が表示されます。

図4.3 スマートカードオプション



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

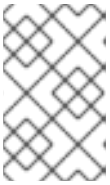
Red Hat Enterprise Linux にログインする際にシングルサインオンを使用するには、以下のパッケージが必要です。

- 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. ネットワーク用のルート 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。証明書は、**certutil** コマンドを使用して適切なシステムデータベースにインストールされます。以下に例を示します。

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



注記

インポートされた証明書は、プロセスの後で **authconfig** UI に表示されないことに注意してください。UI に証明書は表示されません。認証中に **/etc/pki/nssdb/** ディレクトリーから証明書を取得します。

3. トップメニューで **Application** メニューを選択し、**Sundry** を選択してから **Authentication** 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. **Enable Smart Card Support** 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 **削除アクション** メニューは、アクティブなセッション中にスマートカードが削除された場合にシステムが取得する応答を設定します。**無視** オプションは、スマートカードが削除され、**Lock** が画面を直ちにロックしている間に、システムが通常通りに機能し続けることを意味します。

   - **Require smart card for login** チェックボックスは、ログインにスマートカードが必要であるかどうかを設定します。このオプションを選択すると、認証の他の方法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正常にログイン **するまで** は、これを選択しないでください。

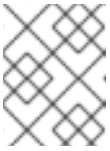
7. デフォルトでは、証明書が失効したかどうかを確認するメカニズム (オンライン証明書ステータスプロトコル、OCSP、応答) は無効です。有効期限が切れる前に証明書を失効したかどうかを検証するには、***cert_policy*** ディレクティブに **ocsp_on** オプションを追加して OCSP チェックを有効にします。

   a. **pam_pkcs11.conf** ファイルを開きます。

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

   b. **ocsp_on** オプションが含まれるように、すべての ***cert_policy*** 行を変更します。

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



注記

ファイルの解析方法により、**cert_policy** と等号記号の間にスペースが **必要** です。そうでない場合は、パラメーターの解析に失敗します。

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 カードの場合は、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。CAC カードに Microsoft プリンシパル名を使用するには、**.k5login** ファイルが必要です。

10. 以下の行を **/etc/pam.d/smartcard-auth** ファイルおよび **/etc/pam.d/system-auth** 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 モジュールが想定どおりに機能しない場合は、coolkey パッケージの モジュール(**/usr/lib64/pkcs11/libcoolkeypk11.so**)を使用します。この場合は、Red Hat テクニカルサポートへの問い合わせ、または問題に関する Bugzilla レポートの作成を検討してください。

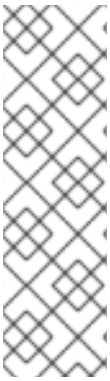
11. **/etc/krb5.conf** ファイルを設定します。この設定は、CAC カードか Gemalto 64K カードを使用しているかによって異なります。

- CAC カードでは、CAC カードの使用に関連するすべてのルート証明書を **pkinit_anchors** で指定します。CAC カードを設定するための **/etc/krb5.conf** ファイルの例では、*EXAMPLE.COM* は CAC カードのレルム名で、*kdc.server.hostname.com* は KDC サーバーのホスト名です。

```
[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
    }
```

- 以下の Gemalto 64K カードを設定するための **/etc/krb5.conf** ファイルの例では、*EXAMPLE.COM* は KDC サーバー上に作成されたレルム、*kdc-ca.pem* は CA 証明書、*kdc.server.hostname.com* は KDC サーバーのホスト名です。

```
[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、デバッグモードで実行する際に **pklogin_finder** ユーティリティーがログイン ID をカード上の証明書にマッピングしてから、証明書の有効性に関する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用し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ために必要なのは 、**--enablesmartcard** オプションを設定することだけです。

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

スマートカードの他の設定オプションには、その他にもデフォルトのスマートカードモジュールの変更、スマートカードの削除時にシステムの動作の設定、ログイン用のスマートカードの要求などの設定オプションがあります。

値が **0** の場合は、スマートカードが削除された場合にすぐにユーザーをロックするようにシステムに指示します。**1** を設定すると、スマートカードが削除された場合にこれを無視します。

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

スマートカード認証が正常に設定およびテストされたら、シンプルなパスワードベースの認証ではなく、ユーザーにスマートカード認証を要求するようにシステムを設定できます。

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

警告

スマートカードを使用してシステムに正常に認証されるまで **--enablerequiresmartcard** オプションを使用しないでください。それ以外の場合は、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。詳細は、『Linux Domain Identity, Authentication, and Policy Guide 』のSmart-card Authentication in Identity Managementを参照してください。。

4.4.3. 対応するスマートカード

Red Hat Enterprise Linux では、以下のスマートカードとリーダーがサポートされます。

スマートカード

- Athena ASECard Crypto スマート、pkcs15-unit
- ATOS (Siemens) CardOS 5.0
- Gemalto ID Classic 230 / TOP IM CY2 64kv2
- Gemalto Cyberflex Access 64k V2c
- Gemalto GemPCKey USB フォームファクターキー
- 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

- スマートカードリーダーを内臓した HP キーボード KUS1206
- Omnikey 3121 リーダー
- PID 0x3022 リーダーを備えた Omnikey 3121
- Reiner SCT cyberJack RFID komfort リーダー
- SCR331 CCID リーダー

4.5. ワンタイムパスワード

ワンタイムパスワード (OTP) は、認証セッションを 1 つだけ有効なパスワードで、使用後に無効になります。長期間同じになる従来の静的パスワードとは異なり、OTP は変更を維持します。OTP は、2 要素認証の一部として使用されます。最初のステップでは、ユーザーが従来の静的パスワードで認証する必要があり、2 番目のステップでは、認識された認証トークンによって発行された OTP の入力が求められます。

OTP と静的パスワードの組み合わせを使用した認証は、静的パスワードのみを使用した認証よりも安全であると考えられます。OTP は、ログイン中に OTP をインターセプトする可能性がある場合でも、正常な認証を一度だけ使用できるため、傍受された OTP はすでにそのポイントにより無効になります。

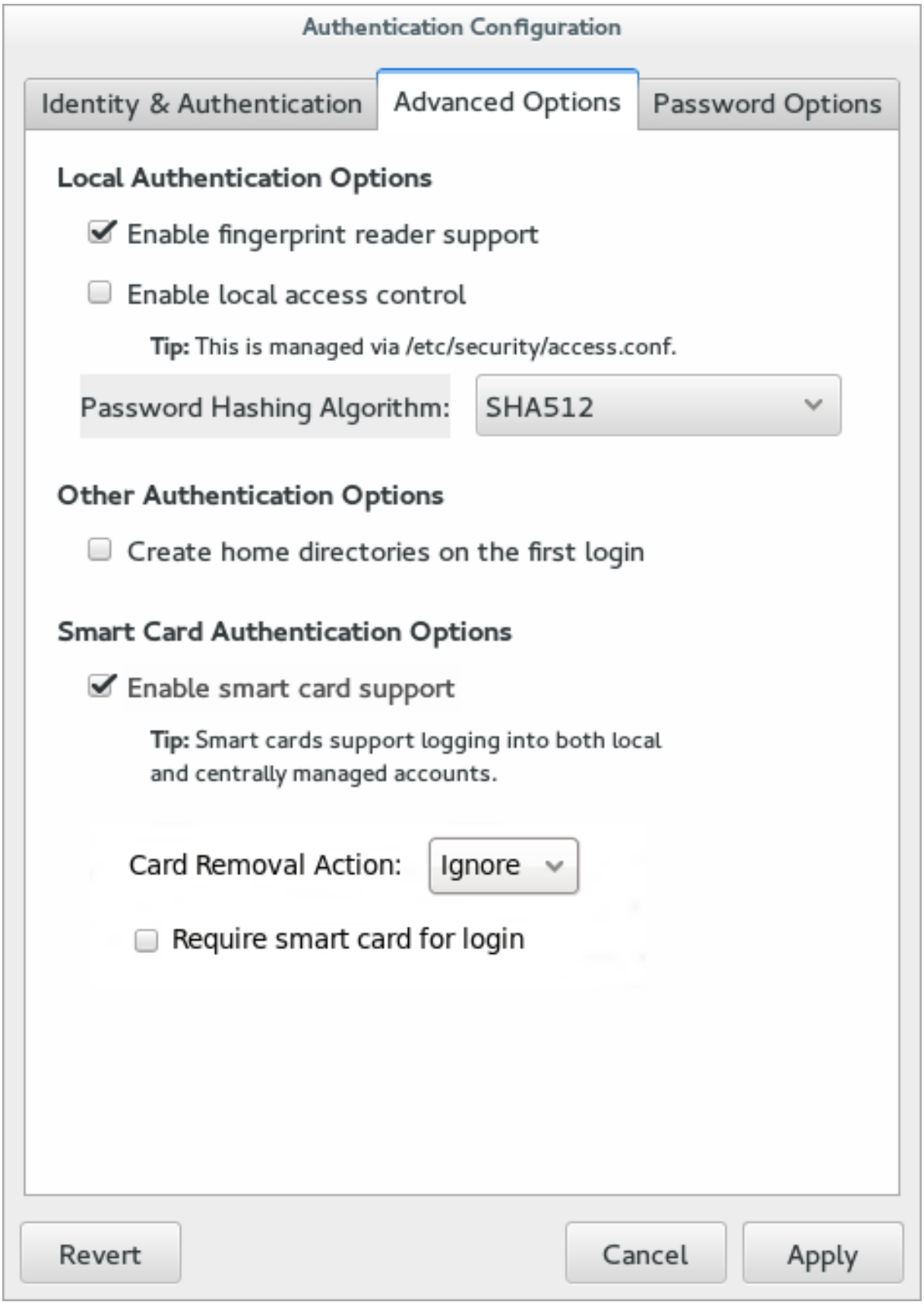
Red Hat Enterprise Linux のワンタイムパスワード
Red Hat Identity Management は、IdM ユーザーの OTP 認証をサポートします。詳細は、『Linux Domain Identity, Authentication, and Policy Guide』のOne-Time Passwordsを参照してください。

4.6. AUTHCONFIGを使用したフィンガープリントの設定

4.6.1. UI でのフィンガープリント認証の使用

適切なハードウェアが利用可能な場合は、**フィンガープリントリーダーのサポート** オプションを使用すると、他の認証情報に加えて、フィンガープリントスキャンを使用してローカルユーザーを認証できます。

図4.4 フィンガープリントのオプション



4.6.2. コマンドラインでのフィンガープリント認証の設定

フィンガープリントリーダーのサポートを有効にするオプションは 1 つあります。このオプションは、単独または LDAP ユーザーストアなどの他の **authconfig** 設定と併用できます。

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

第5章 AUTHCONFIGを使用したキックスタートファイルと設定ファイルの管理

--update オプションは、設定の変更ですべての設定ファイルを更新します。いくつかの代替オプションがあり、動作が若干異なります。

- **--kickstart** は、更新された設定をキックスタートファイルに書き込みます。
- **--test** は変更と共に完全な設定を表示しますが、設定ファイルは編集しません。

さらに、**authconfig** を使用して以前の設定をバックアップおよび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** オプションは、バックアップディレクトリーを **2011-07-01** として指定します。

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

これにより、**/var/lib/authconfig/backup-2011-07-01 ディレクトリーの下にあるすべての認証設定ファイルがバックアップ** されます。

保存したバックアップは、**--restorebackup** オプションを使用して設定を復元し、手動で保存した設定の名前を指定します。

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

また、**authconfig** は、変更を適用する前に設定のバックアップを作成します(**--update** オプションを使用)。設定は、**--restorelastbackup** オプションを使用して完全バックアップを指定することなく、最新の自動バックアップから復元できます。

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 ユーザーに **/home** にないホームディレクトリーがあり、ユーザーが初めてログインしたときにホームディレクトリーを作成するようにシステムが設定されている場合、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ます。

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

   ```
   [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、**authconfig** コマンドがホームディレクトリーの作成に使用する **pam_oddjob_mkhomedir.so** ライブラリーを提供します。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は自動的に使用します。それ以外の場合は、デフォルトで **pam_mkhomedir.so** が使用されます。

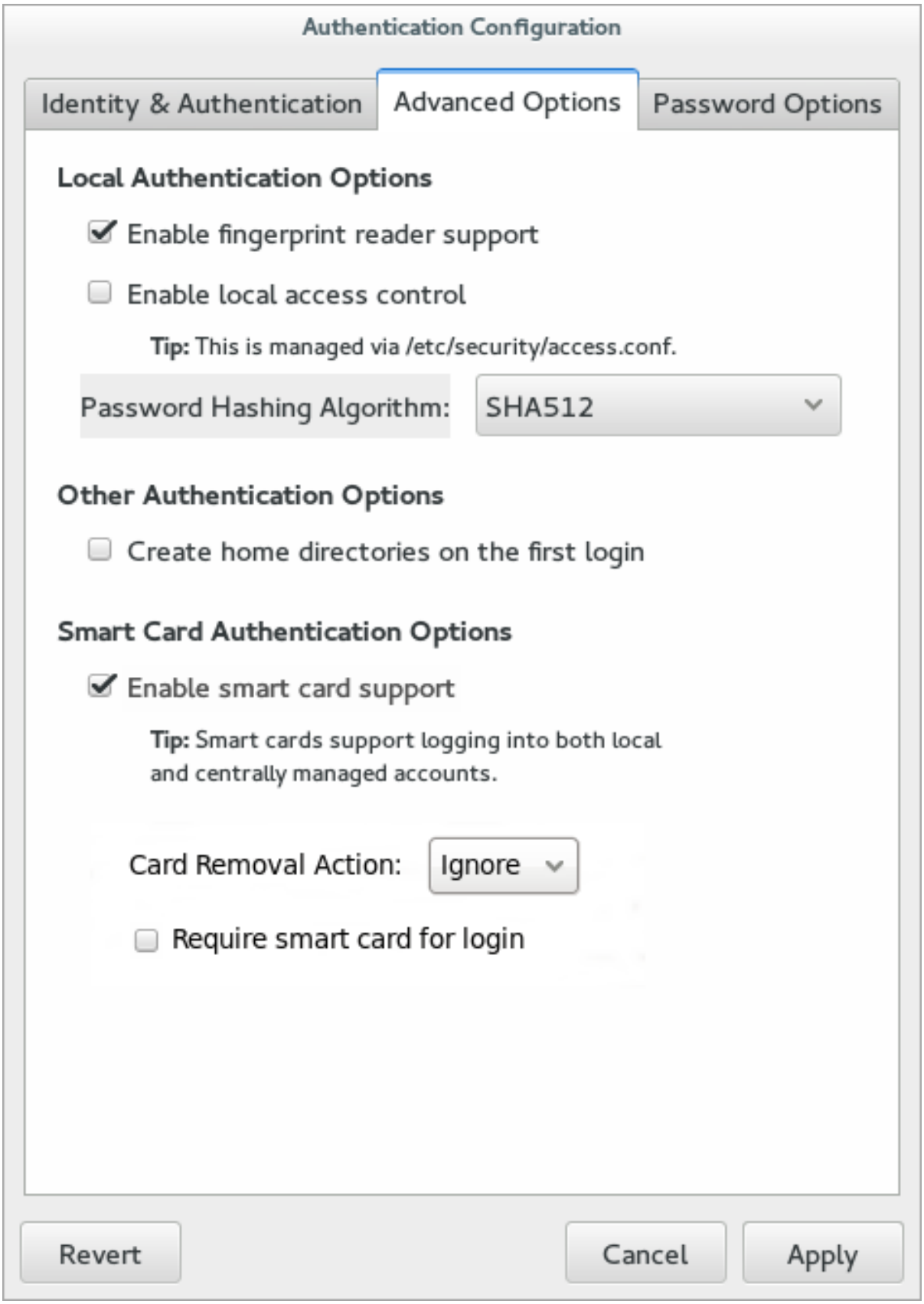
3. **oddjobd** サービスが実行中であることを確認します。

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** オプションを指定して実行します。

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

   UI には、**Advanced Options** タブ(**最初のログインでホームディレクトリーの作成)に、ユーザーが初めてログイン**したときにホームディレクトリーを自動的に作成するオプションがあります。

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、集中管理されているアカウントでは利点があります。ただし、自動マウントなどのシステムがユーザーのホームディレクトリーの管理に使用される場合、このオプションは選択しないでください。

ホームディレクトリーの設定を変更する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 コンテキストを修正します。以下に例を示します。

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

パート II. ID ストアと認証ストア

このパートでは、**System Security Services Daemon** (**SSSD**)を設定する方法、**realmd** ツールを使用して ID ドメインに接続する方法、および **OpenLDAP** サーバーをインストール、設定、および実行する方法を説明します。

第7章 SSSD の設定

7.1. SSSD の概要

7.1.1. 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。ローカルシステム (SSSD *クライアント*) を外部のバックエンドシステム (*プロバイダー*) に接続します。これにより、SSSD クライアントに SSSD プロバイダーを使用した ID および認証のリモートサービスへのアクセスが提供されます。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、または Kerberos レルムが含まれます。

このため、SSSD は以下のようになります。

1. クライアントをアイデンティティーストアに接続して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、外部バックエンドシステムに保存されているユーザーアカウントを使用して認証を行うことができます。

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。代わりに、外部データストアからのアイデンティティーを使用し、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、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。

7.1.2. SSSD を使用する利点

ID および認証サーバーへの負荷の削減

情報を要求すると、SSSD クライアントは SSSD に接続し、SSSD はキャッシュをチェックします。SSSD は、キャッシュで情報が利用できない場合に限り、サーバーに問い合わせます。

オフライン認証

SSSD は、必要に応じて、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。この設定では、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、ユーザーがリソースに対して正常に認証できるようになります。

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。

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あります。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。

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。

7.2. クライアントごとに複数の SSSD 設定ファイルの使用

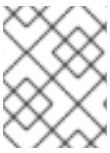
SSSD のデフォルトの設定ファイルは **/etc/sssd/sssd.conf** です。このファイルとは別に、SSSD は **/etc/sssd/conf.d/** ディレクトリー内のすべての ***.conf** ファイルから設定を読み取ることができます。

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** ディレクトリーの隠しファイル(**.**で始まるファイル)を読み取りません。

7.3. SSSD の ID および認証プロバイダーの設定

7.3.1. SSSD の ID プロバイダーおよび認証プロバイダーの概要

SSSD ドメイン。ID プロバイダー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 設定ファイルで *ドメイン* として設定されます。1 つのドメインを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* (ユーザー情報用)
- *認証プロバイダー* (認証要求用)
- *アクセス制御プロバイダー* (承認要求用)
- (対応するすべての操作が単一のサーバー内で実行する場合の) これらのプロバイダーの組み合わせ

SSSD 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。設定しないと、SSSD は起動しません。

/etc/sssd/sssd.conf ファイルの **access_provider** オプションは、ドメインに使用されるアクセス制御プロバイダーを設定します。デフォルトでは、オプションは を **許可** し、常にすべてのアクセスを許可します。詳細は sssd.conf(5) の man ページを参照してください。

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。

プロキシープロバイダーを使用して、SSSD を以下を使用するように設定できます。

- 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** およびリモート認証で定義されたローカルシステムアカウント

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

表7.1 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 [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	

このガイドでは、すべてのプロバイダータイプについて説明しているわけではないことに注意してください。詳細情報は、以下の追加リソースを参照してください。

- Identity Management に SSSD クライアントを設定するには、Red Hat は **ipa-client-install** ユーティリティーを使用することを推奨します。『Linux Domain Identity, Authentication, and Policy Guide』のInstalling and Uninstalling Identity Management Clients
- **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 で使用する Active Directory を設定するには、『Windows 統合ガイド』のActive Directory を SSSD の アイデンティティープロバイダーとして使用を参照してください。

7.3.2. SSSD の 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
```

LDAP ドメインを検出するように SSSD を設定

1. **/etc/sssd/sssd.conf** ファイルを開きます。
2. LDAP ドメインの **[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ープロバイダー、認証プロバイダー、またはその両方として使用するかどうかを指定します。
   a. LDAP サーバーをアイデンティティープロバイダーとして使用するには、***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には、***auth_provider*** オプションを **ldap** に設定します。

   たとえば、両方に LDAP サーバーを使用するには、以下を実行します。

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

4. LDAP サーバーを指定します。以下のいずれかを選択します。
   a. サーバーを明示的に定義するには、***ldap_uri*** オプションでサーバーの URI を指定します。

```
[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 アドレスも受け入れます。ただし、サーバー名の代わりに IP アドレスを使用すると、TLS/SSL 接続が失敗する場合があります。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. DNS サービス検出を使用してサーバーを動的に検出するように SSSD を設定するには、「DNS サービスディスカバリーの設定」 を参照してください。

   必要に応じて、***ldap_backup_uri*** オプションで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 サーバーへのセキュアな接続を確立する方法を指定します。TLS 接続を使用することが推奨されます。これを行うには、***ldap_id_use_start_tls*** オプションを有効にし、以下の 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、および証明書で実行されるチェックを行うかどうかを指定します
   - ***ldap_tls_cacert*** 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 = ldap://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、認証に暗号化されたチャンネルを常に使用します。これにより、暗号化されていないネットワーク上でパスワードが送信されなくなります。**ldap_id_use_start_tls = true** を使用すると、ID ルックアップ(**id** ユーティリティーまたは **getent** ユーティリティーに基づくコマンドなど)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。このオプションは、SSSD がクエリーするドメインを一覧表示します。以下に例を示します。

```
domains = LDAP_domain_name, domain2
```

関連情報

上記の手順は、LDAP プロバイダーの基本オプションを示しています。詳細は、以下を参照してください。

- sssd.conf(5) の man ページ。ここでは、すべてのタイプのドメインで利用可能なグローバルオプションを説明します。
- sssd-ldap(5) の man ページ。ここでは、LDAP に固有のオプションを説明します。

7.3.3. SSSD のファイルプロバイダーの設定

files プロバイダーは、**/etc/passwd** および **/etc/groups** ファイルの内容をミラーリングし、SSSD でこれらのファイルからのユーザーおよびグループを利用できるようにします。これにより、**sss** データベースを、**/etc/nsswitch.conf** ファイルのユーザーおよびグループの最初のソースとして設定できます。

■

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

この設定では、**files** プロバイダーが **/etc/sssd/sssd.conf** に設定されている場合、Red Hat Enterprise Linux は最初にユーザーおよびグループのすべてのクエリーを SSSD に送信します。SSSD が実行していない場合、または SSSD が要求されたエントリーが見つからない場合、システムはフォールバックしてローカルファイルでユーザーおよびグループを検索します。ほとんどのユーザーとグループを LDAP ディレクトリーなどの中央データベースに保存する場合、この設定により、ユーザーとグループの検索速度が向上します。

前提条件

- SSSD をインストールします。

```
# yum install sssd
```

ファイルドメインを検出するように SSSD を設定

1. **/etc/sssd/sssd.conf** ファイルに以下のセクションを追加します。

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

2. 必要に応じて、**sss** データベースを、**/etc/sssd/sssd.conf** ファイルでユーザーおよびグループの検索の最初のソースとして設定します。

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

3. システムの起動時に **sssd** サービスが起動する方法でシステムを設定します。

```
# systemctl enable sssd
```

4. **sssd** サービスを再起動します。

```
# systemctl restart sssd
```

関連情報

上記の手順では、**files** プロバイダーの基本オプションを示しています。詳細は、以下を参照してください。

- sssd.conf(5) の man ページ。ここでは、すべてのタイプのドメインで利用可能なグローバルオプションを説明します。
- sssd-files(5) の man ページ。これは、**files** プロバイダーに固有のオプションを説明します。

7.3.4. SSSD の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
```

プロキシードメインを検出するように SSSD を設定

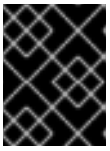
1. **/etc/sssd/sssd.conf** ファイルを開きます。
2. プロキシープロバイダーの **[domain]** セクションを作成します。

```
[domain/proxy_name]
```

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

   以下に例を示します。

```
[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** が再帰的に含まれて **いない** ことを確認します。

4. アイデンティティー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 **プロキシー** に設定します。
   b. NSS ライブラリーを ID プロキシーとして指定するには、***proxy_lib_name*** オプションを使用します。

   以下に例を示します。

```
[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。このオプションは、SSSD がクエリーするドメインを一覧表示します。以下に例を示します。

```
domains = proxy_name, domain2
```

関連情報

上記の手順では、プロキシープロバイダーの基本オプションを説明します。詳細は sssd.conf(5) の man ページを参照してください。これは、すべてのタイプのドメインや他のプロキシー関連のオプションで利用可能なグローバルオプションを説明します。

7.3.5. 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
```

Kerberos ドメインを検出するように SSSD を設定

1. **/etc/sssd/sssd.conf** ファイルを開きます。
2. SSSD ドメインの **[domain]** セクションを作成します。

```
[domain/Kerberos_domain_name]
```

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

   Kerberos プリンシパル名が指定のアイデンティティープロバイダーで利用できない場合、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。このオプションは、サーバーのホスト名または IP アドレスを受け入れます。

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

krb5_server = kdc.example.com
```

      ii. DNS サービス検出を使用してサーバーを動的に検出するように SSSD を設定するには、「DNS サービスディスカバリーの設定」 を参照してください。

   必要に応じて、***krb5_backup_*** server オプションでバックアップサーバーを指定します。

   c. krb5_server または ***krb5_ backup_server で指定された KDC で Change Password サービスが実行されていない場合は、krb5_ passwd*** オプションを使用して、サービスが実行しているサーバーを指定します。

```
[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 を使用しないと、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_realm*** オプションを使用して、Kerberos レルムの名前を指定します。

```
[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_realm = 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。このオプションは、SSSD がクエリーするドメインを一覧表示します。以下に例を示します。

```
domains = Kerberos_domain_name, domain2
```

関連情報

上記の手順では、Kerberos プロバイダーの基本オプションを説明します。詳細は、以下を参照してください。

- sssd.conf(5) の man ページ。ここでは、すべてのタイプのドメインで利用可能なグローバルオプションを説明します。
- sssd-krb5(5) の man ページ。ここでは、Kerberos 固有のオプションを説明します。

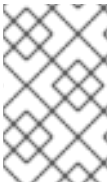
7.4. ID プロバイダーおよび認証プロバイダーの追加設定

7.4.1. ユーザー名の形式の調整

7.4.1.1. 完全なユーザー名を解析するための正規表現の定義

SSSD は、完全なユーザー名の文字列を解析して、ユーザー名とドメインコンポーネントにします。デフォルトでは、SSSD は Python 構文の以下の正規表現に基づいて、**user_name@domain_name** 形式の完全なユーザー名を解釈します。

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



注記

Identity Management プロバイダーおよび Active Directory プロバイダーでは、デフォルトのユーザー名形式は ***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/sssd/sssd.conf** ファイルを開きます。
2. ***re_expression*** オプションを使用して、カスタムの正規表現を定義します。

a. すべてのドメインに対してグローバルに正規表現を定義するには、**sssd.conf** の **[sssd]** セクションに ***re_expression*** を追加します。

b. 特定のドメインに対して個別に正規表現を定義するには、**sssd.conf** の対応するドメインセクションに ***re_expression*** を追加します。

たとえば、*LDAP* ドメインの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>[^\\]+$)
```

詳細は、sssd.conf(5) man ページの **SPECIAL SECTIONS** および **DOMAIN SECTIONS** 部分の ***re_expression*** の説明を参照してください。

7.4.1.2. SSSD 出力の完全ユーザー名の定義

/etc/sssd/sssd.conf ファイルで **use_fully_qualified_names** オプションが有効になっている場合、SSSD は、デフォルトで以下の拡張に基づいて **@domain** 形式で完全なユーザー名を出力します。

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

注記

use_fully_qualified_names が設定されていない場合や、信頼されるドメインに対して明示的に **false** に設定されている場合には、ドメインコンポーネントなしでユーザー名のみが出力されます。

SSSD が完全なユーザー名を出力する形式を調整するには、次のコマンドを実行します。

1. **/etc/sssd/sssd.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拡張を定義します。

   a. すべてのドメインに対してグローバルに拡張を定義するには、**sssd.conf** の **[sssd]** セクションに ***full_name_format*** を追加します。

   b. 特定のドメインに拡張を個別に定義するには、**sssd.conf** の対応するドメインセクションに ***full_name_format*** を追加します。

詳細は、sssd.conf(5) man ページの **SPECIAL SECTIONS** および **DOMAIN SECTIONS** 部分の ***full_name_format*** の説明を参照してください。

名前設定によっては、SSSD は名前のドメインコンポーネントを削除できるため、認証エラーが発生する可能性があります。このため、***full_name_format*** を標準以外の値に設定すると、警告により、これをより標準形式に変更するように求められます。

7.4.2. オフライン認証の有効化

SSSD は、デフォルトでは、ユーザーの認証情報をキャッシュしません。認証要求の処理時に、SSSD は常にアイデンティティープロバイダーに問い合わせします。プロバイダーが利用できない場合は、ユーザー認証に失敗します。

重要

SSSD は、パスワードをプレーンテキストでキャッシュしません。パスワードのハッシュのみを保存します。

アイデンティティープロバイダーが利用できない場合にユーザーが認証できるようにするには、認証情報キャッシュを有効にします。

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

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

3. **任意ですが、推奨されます。**アイデンティティープロバイダーが利用できない場合に SSSD がオフライン認証の許可期間に制限を設定します。
   a. SSSD と連携するように 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時間制限を指定します。たとえば、最終ログインに成功してから 3 日間、オフライン認証を可能にするには、次のようにします。

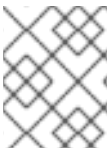
```
[pam]
offline_credentials_expiration = 3
```

offline_credentials_expiration の詳細は、sssd.conf(5) の man ページを参照してください。

7.4.3. DNS サービスディスカバリーの設定

アイデンティティーまたは認証サーバーが **/etc/sssd/sssd.conf** ファイルで明示的に定義されていない場合、SSSD は *DNS サービス*検出を使用してサーバーを動的に検出できます。[1]をクリックします。

たとえば、**sssd.conf** に ***id_provider = ldap*** 設定が含まれているものの、***ldap_uri*** オプションでホスト名または IP アドレスが指定されていない場合、SSSD は DNS サービス検出を使用してサーバーを動的に検出します。



注記

SSSD が検出するのは、プライマリーサーバーのみで、バックアップサーバーを動的には検出できません。

DNS サービス検出のための SSSD の設定

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

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

3. パスワード変更プロバイダーでサービス検出を有効にするには、サービスタイプを設定します。

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

chpass_provider = ldap
ldap_chpass_dns_service_name = ldap
```

4. **オプション:**サービス検出は、システムホスト名のドメイン部分をドメイン名として使用します。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、サービス検出は 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 アドレスの検索を試行します。試行に失敗すると、SSSD は IPv6 アドレスの検索を試行します。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 sssd.conf(5) の man ページを参照してください。

7. サービス検出を使用するすべてのサービスについて、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一覧に基づいてアクセスを許可または拒否します。これにより、特定のマシンへのアクセスを制限できます。

たとえば、企業のラップトップでは、**Simple アクセスプロバイダーを使用して**、特定のユーザーまたは特定のグループのみへのアクセスを制限できます。他のユーザーまたはグループは、設定済みの認証プロバイダーに対して正常に認証されている場合でもログインできません。

Simple Access Provider ルールの設定

1. **/etc/sssd/sssd.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ルールを定義します。以下のいずれかを選択します。

   a. ユーザーへのアクセスを許可するには、***simple_allow_users*** オプションを使用します。

   b. ユーザーへのアクセスを拒否するには、***simple_deny_users*** オプションを使用します。

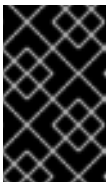
重要

特定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であると考えられます。特定のユーザーへのアクセスを拒否する場合には、他のユーザーすべてにアクセスを自動的に許可します。

4. グループのアクセス制御ルールを定義します。以下のいずれかを選択します。

    a. グループへ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. グループへ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であると考えられます。特定のグループへのアクセスを拒否する場合には、他のグループすべてに、アクセスを自動的に許可します。

次の例では、**group1** の **user1**、**user2**、および メンバーへのアクセスを許可し、他のすべてのユーザーへのアクセスを拒否します。

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

詳細は sssd-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

/etc/sssd/sssd.conf で ***access_provider*** オプションが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、システムにアクセスできるユーザーを評価します。使用しているアクセスプロバイダーが LDAP プロバイダータイプの拡張の場合には、システムへのアクセス許可に必要な LDAP アクセス制御フィルターを指定することもできます。

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は、Linux システムへのアクセスを特定の AD ユーザーに制限できます。指定のフィルターに一致しない他のユーザーはすべてアクセスが拒否されます。

注記

アクセスフィルターは LDAP ユーザーエントリーにのみ適用されます。そのため、ネスト化されたグループでこのタイプのアクセス制御を使用すると機能しない可能性があります。ネストされたグループにアクセス制御を適用するには、「**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、ユーザーが最後にオンラインログインの試行に成功したかどうかを確認します。直近のオンラインログイン中に正常にログインしたユーザーは、アクセスフィルターに一致しない場合でも、オフラインでログインできるようになります。

LDAP アクセスフィルターを適用するための SSSD 設定

1. **/etc/sssd/sssd.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の場合は、***ldap_access_filter*** オプションを使用します。詳細は sssd-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の場合は、***ad_access_filter*** オプションを使用します。詳細は sssd-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*** 属性が設定されている AD ユーザーにのみアクセスを許可するには、次のコマンドを実行します。

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

SSSD は、エントリーの ***authorizedService*** または ***host*** 属性で結果を確認することもできます。実際、ユーザーエントリーおよび設定に応じて、すべてのオプション(LDAP フィルター、***authorizedService***、および ***host***)を評価できます。**ldap_access_order** パラメーターは、評価すべき順に、使用するアクセス制御の手法をすべて表示します。

```
[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
```

認可されたサービスまたは許可されたホストの評価に使用するユーザーエントリーの属性をカスタマイズできます。追加のアクセス制御パラメーターは、**sssd-ldap (5)** man ページに一覧表示されます。

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イスを提供します。以下に例を示します。

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OpenSSH サービスのキャッシュを提供するように SSSD の設定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

NSS での SSSD の仕組み

Name Service Switch (NSS) サービスは、システム ID およびサービスを設定ソースとマッピングします。これは、サービスがさまざまな設定および名前解決メカニズムのソースを検索できる中央の設定ストアを提供します。

SSSD は、NSS を、いくつかのタイプの NSS マップのプロバイダーとして使用できます。以下に例を示します。

- ユーザー情報(**passwd** マップ)
- グループ(**groups** マップ)
- netgroups (**netgroups** マップ)
- サービス(**services** マップ)

前提条件

- SSSD をインストールします。

```
# yum install sssd
```

SSSD を使用するように NSS サービスを設定する

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** ファイルが更新され、以下の NSS マップが SSSD を使用できるようになります。

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

netgroup:   files sss
```

2. **/etc/nsswitch.conf** を開き、**services** マップ行に **sss** を追加します。

```
services: files sss
```

NSS で動作するように SSSD を設定する

1. **/etc/sssd/sssd.conf** ファイルを開きます。
2. **[sssd]** セクションで、NSS が SSSD と連携するサービスの 1 つとしてリストされていることを確認します。

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

3. **[nss]** セクションで、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一覧は、sssd.conf(5) 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
```

インテグレーションの動作が正しく行われることのテスト
以下のコマンドを使用して、ユーザーに関する情報を表示します。

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

7.5.2. サービスの設定: PAM

警告

PAM 設定ファイルの間違いにより、ユーザーがシステムから完全にロックされる可能性があります。変更を実行する前に設定ファイルを常にバックアップし、セッションを開いたままにして、変更を元に戻すことができます。

SSSD を使用するように PAM を設定する

- **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、SSSD モジュール（通常は **/etc/pam.d/system-auth** ファイルおよび **/etc/pam.d/password-auth** ファイル）を参照する PAM 設定が更新されます。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 man ページを参照してください。

SSSD が PAM と連携するように設定する

1. **/etc/sssd/sssd.conf** ファイルを開きます。
2. **[sssd]** セクションで、PAM が SSSD と連携するサービスの 1 つとしてリストされていることを確認します。

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

3. **[pam]** セクションで、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一覧は、sssd.conf(5) man ページの **PAM 設定オプション** を参照してください。

4. SSSD を再起動します。

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

インテグレーションの動作が正しく行われることのテスト

- ユーザーとしてログインしてみてください。
- **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

automountでの SSSD の仕組み

automount ユーティリティーは、NFS ファイルシステムを自動的にマウントおよびアンマウントできます（オンデマンドマウント）。これにより、システムリソースが保存されます。**自動マウント** 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 が SSSD をポイントするように設定できます。この設定では、以下が行われます。

1. ユーザーがディレクトリーのマウントを試みると、SSSD は LDAP に接続して、現在の **自動** マウント設定に関する必要な情報を取得します。
2. SSSD は、**自動** マウントで必要な情報をキャッシュに保存します。これにより、LDAP サーバーがオフラインであっても、ユーザーはディレクトリーをマウントできます。

SSSD を使用するように autofs を設定する

1. autofs パッケージをインストールしている。

```
# yum install autofs
```

2. **/etc/nsswitch.conf** ファイルを開きます。

3. **automount** の行で、自動マウントマップ情報を検索する場所を **ldap** から **sss** に変更します。

```
automount: files sss
```

autofsと連携するように SSSD を設定

1. **/etc/sssd/sssd.conf** ファイルを開きます。

2. **[sssd]** セクションで、SSSD が管理するサービス一覧に **autofs** を追加します。

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

3. **[autofs]** セクションを新規作成します。空欄のままにすることができます。

```
[autofs]
```

利用可能なオプションの一覧は、sssd.conf(5) man ページの **AUTOFS 設定オプション** を参照してください。

4. SSSD が LDAP から **自動マウント** 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。「SSSD の 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、いくつかの **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一覧は、sssd.conf(5) man ページの **DOMAIN SECTIONS** を参照してください。

autofs オプションを追加で指定しない場合、設定はアイデンティティープロバイダーの設定によって異なります。

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 からマップを出力します。

7.5.4. サービスの設定： sudo

sudoでの SSSD の仕組み

sudo ユーティリティーは、指定したユーザーに管理アクセスを提供します。**sudo** の詳細 は、『システム管理者のガイドの **sudo** ユーティリティーのドキュメント』 を参照し てください。

sudo が SSSD を参照するように設定できます。この設定では、以下が行われます。

1. ユーザーが **sudo** 操作を試みると、SSSD は LDAP または AD に問い合わせ、現在の **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します。これにより、LDAP または AD サーバーがオフラインであっても **sudo** 操作を実行できます。

SSSD は、 ***sudo Host*** 属性の値に応じて、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。詳細は sssd-sudo(5) の man ページを参照してください。

SSSD を使用するように sudo を設定する

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

```
sudoers: files sss
```

sudoで動作するように SSSD を設定

1. **/etc/sssd/sssd.conf** ファイルを開きます。
2. **[sssd]** セクションで、**sudo** を SSSD が管理するサービス一覧に追加します。

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

3. **[sudo]** セクションを新たに作成します。空欄のままにすることができます。

```
[sudo]
```

利用可能なオプションの一覧は、sssd.conf(5) man ページの **SUDO 設定オプション** を参照してください。

4. SSSD がディレクトリーから **sudo** 情報を読み取れるように、LDAP または AD 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、次を参照してください。
   - 「SSSD の LDAP ドメインの設定」
   - 『Windows 統合ガイド』のSSSD の ID プロバイダーとして Active Directory の使用

   LDAP または AD ドメインの **[domain]** セクションには、以下の **sudo**関連のパラメーターを含める必要があります。

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

注記

ID プロバイダーとして Identity Management または AD を設定すると、**sudo** プロバイダーが自動的に有効になります。この場合、***sudo_provider*** パラメーターを指定する必要はありません。

利用可能なオプションの一覧は、sssd.conf(5) man ページの **DOMAIN SECTIONS** を参照してください。

sudo プロバイダーに利用可能なオプションは、sssd-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
```

AD をプロバイダーとして使用する場合は、AD スキーマを拡張して **sudo** ルールをサポートする必要があります。詳細は **sudo** ドキュメントを参照してください。

LDAP または AD で **sudo** ルールを提供する方法は、sudoers.ldap(5) の man ページを参照してください。

7.6. SSSD クライアント側のビュー

SSSD を使用すると、クライアント側ビューを作成して、POSIX ユーザーまたはグループ属性に新しい値を指定できます。このビューは、オーバーライドが設定されているローカルマシンでのみ有効になります。**ipa** を除き、すべての **id_provider** 値にクライアント側の上書きを設定できます。**ipa** プロバイダーを使用している場合は、IdM で ID ビューを一元的に定義します。Linux Domain Identity, Authentication, and Policy Guideの該当するセクションを参照してください。

詳細は、『Linux Domain Identity, Authentication, and Policy Guide 』のPotential Negative Impact on SSSD Performance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オーバーライドを作成したら、SSSD を再起動して変更を有効にします。

```
# systemctl restart sssd
```

7.6.1. ユーザーアカウントの異なる属性値の定義

管理者は、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ている。ただし、LDAP でのユーザーの新しい ID は、ローカルシステムでのユーザーの以前の ID とは異なります。クライアント側のビューを、既存のファイルの権限を変更する代わりに 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* で上書きするには、以下を実行します。

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の有効期限が切れるまで待ちます。手動で期限切れにするには、以下を実行します。

```
# 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:::::
```

上書きできる属性の一覧は、コマンドに ***--help*** を追加して、コマンドラインオプションを一覧表示します。

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

7.6.2. ホストですべての上書きの一覧表示

管理者は、ホスト上の全ユーザーおよびグループの上書きを一覧表示し、正しい属性が上書きされたことを確認できます。

全ユーザーの上書きを一覧表示するには、以下を行います。

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

全グループの上書きを一覧表示するには、以下を行います。

```
# 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

以前に、グローバル LDAP ディレクトリーで定義されている *user* アカウントのシェルの上書きを作成していました。アカウントの上書きを削除するには、以下のコマンドを実行します。

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

変更はすぐに有効になります。

グループの上書きを削除するには、次を実行します。

```
# 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と、このオブジェクトの上書きがすべて削除されます。

7.6.4. ローカルビューのエクスポートおよびインポート

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ます。キャッシュから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して、バックアップを作成できます。たとえば、SSSD キャッシュを削除すると、後でビューを再度復元できます。

ユーザーおよびグループビューのバックアップを作成するには、以下を行います。

```
# sss_override user-export /var/lib/sss/backup/sssd_user_overrides.bak
# sss_override group-export /var/lib/sss/backup/sssd_group_overrides.bak
```

ユーザーおよびグループビューを復元するには、以下を行います。

```
# sss_override user-import /var/lib/sss/backup/sssd_user_overrides.bak
# sss_override group-import /var/lib/sss/backup/sssd_group_overrides.bak
```

7.7. SSSD のダウングレード

ダウングレード時 – SSSD のバージョンをダウングレードするか、オペレーティングシステム自体をダウングレードする場合、既存の SSSD キャッシュを削除する必要があります。キャッシュが削除されないと、SSSD プロセスは停止しますが、PID ファイルは残ります。SSSD ログは、キャッシュバージョンが認識されないため、関連付けられたドメインのいずれかに接続できないことを示します。

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

その後、ユーザーは認識されなくなり、ドメインサービスおよびホストに対して認証できなくなります。

SSSD バージョンをダウングレードしてから以下を行います。

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 での NSCD の使用

SSSD は、NSCD デーモンで使用するように設計されていません。SSSD は NSCD と直接競合しませんが、両サービスを使用すると、特にエントリーがキャッシュされる期間において予期しない動作が発生する可能性があります。

問題の最も一般的な証拠は NFS と競合します。Network Manager を使用してネットワーク接続を管理する場合、ネットワークインターフェイスが起動するまでに数分かかる場合があります。この間、さまざまなサービスが起動しようとします。これらのサービスがネットワークが稼働している前に起動し、DNS サーバーが利用できる場合には、これらのサービスは必要な正引きまたは逆引き DNS エントリーを特定できません。これらのサービスは、誤った resolv.conf ファイルまたは空の **resolv.conf** ファイルを読み取ります。このファイルは、通常 1 回だけ読み取るため、このファイルへの変更は自動的に適用されません。これにより、サービスを手動で再起動しない限り、NSCD サービスが実行されているマシンで NFS ロックが失敗する可能性があります。

この問題を回避するには、**/etc/nscd.conf** ファイルの **ホスト** でのみキャッシュを有効にし、**passwd**、**group**、**services**、および **netgroup** エントリーの SSSD キャッシュに依存します。

/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
```

ホストの要求に回答する NSCD により、これらのエントリーは NSCD によりキャッシュされ、ブートプロセス中に NSCD によって返されます。その他のエントリーはすべて SSSD により処理されます。

7.9. 関連情報

- SSSD 関連の man ページの完全なリストは、sssd(8) man ページの **SEE ALSO** セクションを参照してください。
- トラブルシューティングに関するアドバイス: 「SSSD のトラブルシューティング」
- SSSD がサーバーから送信されるパスワード有効期限の警告を処理し、ローカルシステムでユーザーに表示される手順: Red Hat ナレッジベースのSetting Password Expiry
- SSSD クライアントは、LDAP サーバーから取得した全ユーザーに対して GID を自動的に作成でき、GID 番号がすでに取得されていない限り、GID がユーザーの UID と一致することを確認します。Active Directory に直接統合される SSSD クライアントで GID の自動作成を行う方法を確認するには、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 サービス検出を使用すると、アプリケーションが特定タイプの特定サービスに対して指定のドメインの SRV レコードを確認し、必要なタイプのサーバーを返すことができます。DNS サービス検出は RFC 2782 で定義されています。

第8章 REALMD を使用した IDENTITY ドメインへの接続

realmd システムは、アイデンティティードメインを検出して参加する明確で簡単な方法を提供します。ドメイン自体に接続されませんが、ドメインに接続するように SSSD や Winbind などの基礎となる Linux システムサービスを設定します。

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ます。**realmd** を使用して AD 以外の ID ドメインに接続する場合も同じ手順が適用されます。『Windows Integration Guide』のUsing realmd to Connect to an Active Directory Domain を参照してください。

第9章 LDAP サーバー

LDAP (Lightweight Directory Access Protocol)は、ネットワーク上で一元的に保存された情報にアクセスするために使用されるオープンプロトコルのセットです。ディレクトリー共有の **X.500 標準に** 基づいていますが、より複雑でリソースを大量に消費します。このため、LDAP は 「X.500 Lite」 と呼ばれることもあります。

X.500 と同様に、LDAP はディレクトリーを使用して階層的な方法で情報を編成します。これらのディレクトリーは、名前、アドレス、電話番号などのさまざまな情報を保存し、*Network Information Service* (NIS) と同様に使用でき、ユーザーが LDAP 対応のネットワーク上のマシンからアカウントにアクセスできるようにすることもできます。

LDAP は通常、一元管理されたユーザーおよびグループ、ユーザー認証、またはシステム設定に使用されます。また、ユーザーは仮想電話ディレクトリーとしても提供でき、ユーザーは他のユーザーの連絡先情報に簡単にアクセスすることができます。さらに、ユーザーが世界中の他の LDAP サーバーを参照できるようにするため、情報のアドホックなグローバルリポジトリーを提供できます。ただし、大学、政府機関、民間企業などの個々の組織で最も頻繁に使用されます。

9.1. RED HAT DIRECTORY SERVER

Red Hat Directory Server は、ユーザー ID とアプリケーション情報を一元化する LDAP 準拠のサーバーです。アプリケーション設定、ユーザープロファイル、グループデータ、ポリシー、アクセス制御情報を保存するためのオペレーティングシステムに依存しないレジストリーおよびネットワークベースのレジストリーを提供します。

注記

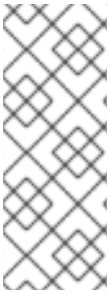
Directory Server をインストールし、更新するには、現在の Red Hat Directory Server サブスクリプションが必要です。

Directory Server の設定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 デプロイメントガイド』
- 『Red Hat Directory Server Administration Guide』
- 『Red Hat Directory Server の設定、コマンド、およびファイルリファレンス』
- 『Red Hat Directory Server Performance Tuning Guide』

9.2. OPENLDAP

このセクションでは、LDAPv2 プロトコルおよび LDAPv3 プロトコルのオープンソース実装である **OpenLDAP 2.**4 のインストールおよび設定について説明します。

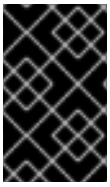


注記

Red Hat Enterprise Linux 7.4 以降、openldap-server パッケージは非推奨となり、Red Hat Enterprise Linux の今後のメジャーリリースには含まれません。このため、Red Hat Enterprise Linux または Red Hat Directory Server に含まれる Identity Management に移行します。Identity Management の詳細は、『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 の詳細は、「Red Hat Directory Server」 を参照してください。

9.2.1. LDAP の概要

クライアントサーバーアーキテクチャーを使用すると、LDAP は、ネットワークからアクセスできる中央情報ディレクトリー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ー内で情報の修正を試みると、サーバーは、ユーザーに変更を行うパーミッションを検証し、要求された時にエントリーを追加または更新します。通信が保護されるようにするには、*Transport Layer Security* (TLS) 暗号プロトコルを使用して、攻撃者が送信を傍受しないようにすることができます。



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、引き続き既存の NSS データベース設定で動作します。

重要

Resolution for POODLE SSLv3.0 vulnerability (CVE-2014-3566) for components that do not allow SSLv3 to be disabled via configuration settings に記載の脆弱性により、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** を効果的に無効にできるよう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。リスクを軽減するには、**stunnel** コマンドを使用してセキュアなトンネルを提供し、**SSLv3** の使用から **stunnel** を無効にすることが推奨されます。**stunnel** の使用に関する詳細は、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、いくつかのデータベースシステムをサポートしているため、管理者は、提供する予定の情報の種類に最適なソリューションを柔軟に選択できます。明確に定義されたクライアントの *アプリケーションプログラミングインターフェイス* (API) により、LDAP サーバーと通信できるアプリケーションの数は多数であり、数量と品質の両方で増加します。

9.2.1.1. LDAP の用語

以下は、本章で使用される LDAP 固有の用語の一覧です。

entry

LDAP ディレクトリー内の単一のユニット。各エントリーは、固有の *識別名* (DN) で識別されます。

attribute

エントリーに直接関連付けられた情報。たとえば、組織が LDAP エントリーとして表される場合、この組織に関連付けられている属性にはアドレス、fax 番号などが含まれる場合があります。同様に、個人の電話番号やメールアドレスなどの一般的な属性のエントリーとして、ユーザーを表示することもできます。

属性は、単一の値、または順序付けられていないスペースで区切られた値のリストのいずれかを持

つことができます。特定の属性は任意ですが、その他は必須です。必要な属性は **objectClass** 定義を使用して指定し、**/etc/openldap/slapd.d/cn=config/cn=schema/** ディレクトリーにあるスキーマファイルにあります。

属性とそれに対応する値のアサーションは、RDN (*Relative Distinguished Name*) とも呼ばれます。グローバルで一意となる識別名とは異なり、相対識別名はエントリーごとに一意のみになります。

LDIF

LDAP データ交換形式 (LDIF) は LDAP エントリーのプレーンテキスト表現です。以下の形式を取ります。

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

任意の *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って決定される数値です。各エントリーには、対応するスキーマファイルにすべて定義されている限り、必要が数の *attribute_type* と *attribute_value* のペアを含めることができます。空白行は、エントリーの最後を示します。

9.2.1.2. OpenLDAP の機能

OpenLDAP スイートは、以下の重要な機能を提供します。

- **LDAPv3 サポート**: LDAP バージョン 2 以降のプロトコルの変更の多くは、LDAP よりセキュアにするように設計されています。そのため、これには SASL (Simple Authentication and Security Layer)、TLS (Transport Layer Security)、および SSL (Secure Sockets Layer) プロトコルのサポートが含まれます。
- **LDAP Over IPC**: プロセス間の通信 (IPC) を使用すると、ネットワーク上で通信する必要がなくなります。
- **IPv6 サポート**: OpenLDAP は、インターネットプロトコルの次世代である IPv6 (Internet Protocol version 6) 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**: 現在の 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、使用する方法を向上させます。
- **強化されたスタンドアロン LDAP サーバー**: これには、更新されたアクセス制御システム、スレッドプール、より良いツールなどが含まれています。

9.2.1.3. OpenLDAP サーバーの設定

Red Hat Enterprise Linux で LDAP サーバーを設定する通常の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詳細は、「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」 の説明に従って設定をカスタマイズします。
3. 「OpenLDAP サーバーの実行」 の説明に従って **slapd** サービスを起動します。

4. **ldapadd** ユーティリティーを使用して、エントリーを LDAP ディレクトリーに追加します。
5. **ldapsearch** ユーティリティーを使用して、**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 ライブラリーおよびツールのスイートは、以下のパッケージで提供されます。

表9.1 OpenLDAP パッケージの一覧

パッケージ	Description
openldap	OpenLDAP サーバーとクライアントアプリケーションの実行に必要なライブラリーを含むパッケージ。
openldap-clients	LDAP サーバーのディレクトリーを表示および変更するコマンドラインユーティリティーを含むパッケージ。
openldap-servers	LDAP サーバーを設定し、実行するサービスとユーティリティーの両方を含むパッケージ。これには、*スタンドアロンの LDAP デーモン* **slapd** が含まれます。
compat-openldap	OpenLDAP 互換性ライブラリーを含むパッケージ。

また、以下のパッケージは、一般的に LDAP サーバーで使用されます。

表9.2 一般的にインストールされている追加 LDAP パッケージの一覧

パッケージ	Description
nss-pam-ldapd	**nslcd** を含むパッケージ。ユーザーがローカル LDAP クエリーを実行できるようにするローカルの LDAP 名サービス。
mod_ldap	**mod_authnz_ldap** モジュールおよび **mod_ldap** モジュールを含むパッケージ。**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、完全な 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強制できます。同じパッケージに含まれる **mod_ldap** モジュールは、設定可能な共有メモリーキャッシュを提供し、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回避し、SSL/TLS もサポートします。このパッケージは Optional チャンネルにより提供されることに注意してください。Red Hat 追加チャンネルの詳細は、『System Administrator's Guide』の Adding the Optional and Supplementary Repositories 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 サーバーインストールを実行するには、シェルプロン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が必要です（つまり、**root**で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は、『System Administrator's Guide』のInstalling Packages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には、openldap-servers パッケージにより、**slapd** サービスとともに以下のユーティリティーがインストールされます。

表9.3 OpenLDAP サーバーユーティリティーの一覧

コマンド	Description
slapacl	属性の一覧へのアクセスを確認できます。
slapadd	LDIF ファイルから LDAP ディレクトリーにエントリーを追加できます。
slapauth	認証および承認権限の ID のリストを確認できます。
slapcat	デフォルト形式の LDAP ディレクトリーからエントリーを取得し、LDIF ファイルに保存できます。
slapdn	利用可能なスキーマ構文に基づいて、識別名 (DN) の一覧を確認できます。
slapindex	現在のコンテンツに基づいて **slapd** ディレクトリーを再インデックス化できます。設定ファイルのインデックスオプションを変更する場合に、このユーティリティーを実行します。
slappasswd	**ldapmodify** ユーティリティーまたは **slapd** 設定ファイルで使用する暗号化されたユーザーパスワードを作成できます。
slapschema	対応するスキーマでデータベースのコンプライアンスを確認できます。
slaptest	LDAP サーバー設定を確認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細な説明は、「インストールされているドキュメント」に記載されている、対応する man ページを参照してください。

重要

slapadd を実行できるのは **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** で作成したファイルを変更できません。この問題を修正するには、**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

データの整合性を保持するには、**slapadd**、**slapcat**、または **slapindex** を使用する前に **slapd** サービスを停止します。これを行うには、シェルプロンプトで以下を実行できます。

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

slapd サービスの現在の状態の開始、停止、再起動、および確認の方法は、「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 ディレクトリーのエントリーの追加、変更、および削除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	Description
ldapadd	エントリーは、ファイルまたは標準入力から LDAP ディレクトリーに追加できます。これは、**ldapmodify -a** へのシンボリックリンクです。
ldapcompare	指定属性を LDAP ディレクトリーエントリーと比較できます。
ldapdelete	LDAP ディレクトリーからエントリーを削除できます。
ldapexop	拡張 LDAP 操作を実行できます。
ldapmodify	LDAP ディレクトリー (ファイルまたは標準入力のいずれか) のエントリーを変更できます。
ldapmodrdn	LDAP ディレクトリーエントリーの RDN 値を変更できます。
ldappasswd	LDAP ユーザーのパスワードを設定または変更できるようにします。
ldapsearch	LDAP ディレクトリーエントリーを検索できます。
ldapurl	LDAP URL の組み立てまたは分解を可能にします。
ldapwhoami	LDAP サーバーで **whoami** 操作を実行できます。

ldapsearch を除き、各ユーティリティーは、LDAP ディレクトリー内で変更する各エントリーのコマンドを入力するのではなく、加えられる変更を含むファイルを参照することで簡単に使用できます。このようなファイルの形式は、各ユーティリティーの man ページで説明されています。

9.2.2.3. 共通 LDAP クライアントアプリケーションの概要

サーバー上にディレクトリーを作成および変更できるさまざまなグラフィカル LDAP クライアントがありますが、Red Hat EnterpriseLinux には含まれていません。読み取り専用モードのディレクトリーにアクセスできる一般的なアプリケーションには、**Mozilla Thunderbird**、**Evolution**、**Ekiga** などがあります。

9.2.3. OpenLDAP サーバーの設定

デフォルトでは、OpenLDAP 設定は **/etc/openldap/** ディレクトリーに保存されます。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および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	Description
/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の設定ファイルこれには、**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。

OpenLDAP は、**/etc/openldap/slapd.conf** ファイルから設定を読み取らなくなりました。代わりに、**/etc/openldap/slapd.d/** ディレクトリーにある設定データベースを使用します。以前のインストールの既存の **slapd.conf** ファイルがある場合は、以下のコマンドを実行して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、階層的なディレクトリー構造で整理された LDIF エントリーで設定され、これらのエントリーを編集する方法として、「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が推奨されます。

重要

LDIF ファイルのエラーにより、**slapd** サービスが起動しないことがあります。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が強く推奨されます。

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** ファイルに保存されます。一般的には、以下のディレクティブが使用されます。

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…
```

表9.6「利用可能な **olcAllows** オプション」 に記載されている、スペースで区切られた機能のリストを受け入れます。デフォルトオプションは **bind_v2** です。

表9.6 利用可能な olcAllows オプション

オプション	Description
bind_v2	LDAP バージョン 2 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) が空でない場合は匿名バインドを有効にします。
bind_anon_dn	識別名 (DN) が空 **でない** 場合は匿名バインドを有効にします。
update_anon	匿名更新操作の処理を有効にします。
proxy_authz_anon	匿名プロキシーの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、匿名セッションの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、認証されたセッションの保留中のリクエスト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…
```

表9.7「利用可能な **olcDisallows** オプション」 に記載されている、スペースで区切られた機能のリストを受け入れます。デフォルトでは、機能は無効になりません。

表9.7 利用可能な olcDisallows オプション

オプション	Description
bind_anon	匿名バインド要求の受け入れを無効にします。
bind_simple	簡単なバインド認証メカニズムを無効にします。
tls_2_anon	STARTTLS コマンドを受け取ると、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閉じる前に待機する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、デフォルトでは無効になっています (つまり **0** に設定されます)。

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 オプションでは、サーバーがこれを処理できない場合に、要求を処理するサーバーの 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になっています。

例9.7 olcReferral ディレクティブの使用

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

olcWriteTimeout

olcWriteTimeout オプションでは、未処理の書き込み要求との接続を閉じる前に待機する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、デフォルトでは無効になっています (つまり **0** に設定されます)。

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)などのグローバルデータベースオプションを定義します。詳細は、slapd-config(5) の man ページの Global Database Options セクションを参照してください。

9.2.3.3. Monitor バックエンド

/etc/openldap/slapd.d/cn=config/olcDatabase=psycmonitor.ldif ファイルは、OpenLDAP モニターのバックエンドを制御します。これを有効にすると、デーモンの実行ステータスに関する情報で、OpenLDAP により自動生成され、動的に更新されます。接尾辞は **cn=Monitor** で、変更できません。詳細は、slapd-monitor(5) の man ページをご覧ください。

9.2.3.4. データベース固有の設定

デフォルトでは、OpenLDAP サーバーは **hdb** データベースバックエンドを使用します。サブツリーの名前をサポートする階層データベースレイアウトを使用する他に、**bdb** バックエンドと同じであり、同じ設定オプションを使用します。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。

その他のバックエンドデータベースの一覧は、slapd.backends(5) の man ページを参照してください。個々のバックエンドの man ページにあるデータベース固有の設定。以下に例を示します。

```
# man slapd-hdb
```

注記

bdb バックエンドおよび **hdb** バックエンドは非推奨になりました。代わりに、新規インストールに **mdb** バックエンドを使用すること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されます。

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
```

TRUE (読み取り専用モードを有効) または **FALSE** (データベースの変更を有効) のいずれかを受け入れます。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、LDAP ディレクトリー上の操作に設定されたアクセス制御または管理制限パラメーターが無制限のユーザーを指定できます。以下の形式を取ります。

```
olcRootDN: distinguished_name
```

識別名 (DN) を受け入れます。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用して指定されるユーザー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
```

プレーンテキストの文字列またはハッシュのいずれかを指定できます。ハッシュを生成するには、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 ディレクティブでは、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
```

完全修飾ドメイン名 (FQDN) を受け入れます。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、**/etc/openldap/schema/** に以前あった LDAP 定義も含まれます。OpenLDAP で使用されるスキーマを拡張して、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、https://openldap.org/doc/admin24/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して保護できます。TLS は、ネットワーク上の通信セキュリティーを提供するために設計された暗号プロトコルです。Red Hat Enterprise Linux 7 の OpenLDAP スイートは、TLS 実装として OpenSSL を使用します。

TLS を使用してセキュアな接続を確立するには、必要な証明書を取得します。次に、クライアントとサーバーの両方で複数のオプションを設定する必要があります。最低でも、認証局 (CA) 証明書と、独自のサーバー証明書および秘密鍵を使用してサーバーを設定する必要があります。クライアントは、すべての信頼される CA 証明書を含むファイルの名前で設定する必要があります。

通常、サーバーは単一の CA 証明書に署名することだけが必要になります。クライアントはさまざまな安全なサーバーに接続したい場合があるため、設定で複数の信頼できる CA のリストを指定するのが一般的です。

サーバー設定

本セクションでは、TLS を確立するために OpenLDAP サーバーの **/etc/openldap/slapd.d/cn=config.ldif** ファイルで指定する必要がある slapd のグローバル設定ディレクティブを一覧表示します。

古いスタイルの設定は単一のファイルを使用しますが、通常は **/usr/local/etc/openldap/slapd.conf** としてインストールされますが、新しいスタイルは **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は、SSL を確立するためにも有効です。TLS ディレクティブの他に、サーバー側で SSL 専用のポートを有効にする必要があります。通常はポート 636 です。これを行うには、**/etc/sysconfig/slapd** ファイルを編集し、**SLAPD_URLS** ディレクティブで指定された URL のリストに **ldaps:**/// 文字列を追加します。

olcTLSCACertificateFile

olcTLSCACertificateFile ディレクティブは、信頼された CA 証明書が含まれる PEM (Privacy-enhanced mail) スキーマでエンコードされるファイルを指定します。ディレクティブは以下の形式になります。

> **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、個別の CA 証明書が含まれる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、実際の証明書ファイルを参照するハッシュ化された名前でシンボリックリンクを生成する OpenSSL **c_rehash** ユーティリティーで特別に管理する必要があります。通常、代わりに **olcTLSCACertificateFile** ディレクティブを使用することが推奨されます。

ディレクティブは以下の形式になります。

> **olcTLSCACertificatePath**: *path*

path を、CA 証明書ファイルを含むディレクトリーのパスに置き換えます。指定し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。ディレクティブは以下の形式になります。

> **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た証明書に一致する秘密鍵が含まれるファイルを指定します。現在の実装は暗号化された秘密鍵に対応していないため、そのファイルが十分に保護されている必要がある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

クライアントシステムの **/etc/openldap/ldap.conf** 設定ファイルで以下のディレクティブを指定しま

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して行いま す。**/etc/openldap/ldap.conf** のディレクティブはシステム全体で設定されますが、各ユーザーは **~/.ldaprc** ファイルで上書きすることができます。

同じディレクティブを使用して SSL 接続を確立できます。**ldaps://** 文字列は、**ldapsearch** などの OpenLDAP コマンドの **ldap://** の代わりに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た SSL のデフォルトポートであるポート 636 を使用するように強制されます。

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認証局の証明書を含むファイルを指定します。これは、サーバーの **olcTLSCACertificateFile** ディレクティブと同じです。**TLS_CACERT** は、**/etc/openldap/ldap.conf** の **TLS_CACERTDIR** の前に常に指定する必要があります。ディレクティブは以下の形式になります。

```
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、別のファイルに認証局証明書が含まれるディレクトリーへのパスを指定します。サーバーの **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のパスに置き換えます。

TLS_CERT

TLS_CERT は、クライアント証明書が含まれるファイルを指定します。このディレクティブは、ユーザーの **~/.ldaprc** ファイルでのみ指定できます。ディレクティブは以下の形式になります。

```
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一致する秘密鍵が含まれるファイルを指定します。サーバーで **olcTLSCertificateFile** と同様に、暗号化された鍵ファイルはサポートされません。ファイル自体は注意して保護する必要があります。このオプションは、ユーザーの **~/.ldaprc** ファイルでのみ設定できます。

TLS_KEY ディレクティブは以下の形式になります。

```
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 1 つ以上の他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の更新をコ

ンシューマーに複製し、受信した更新はコンシューマーによって他のサーバーにさらに伝播されるため、コンシューマーは同時にプロバイダーとして機能することもできます。また、コンシューマーは LDAP サーバーである必要がなく、LDAP クライアントである必要はありません。OpenLDAP では、複数のレプリケーションモードを使用できます。ほとんどの場合は **ミラー** および **同期** です。OpenLDAP レプリケーションモードの詳細は、openldap-servers パッケージ (「インストールされているドキュメント」を参照) でインストールされる **OpenLDAP Software Administrator's Guide** を参照してください。

選択したレプリケーション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で、**/etc/openldap/slapd.d/** で以下のいずれかのディレクティブを使用します。

olcMirrorMode

olcMirrorMode ディレクティブは、ミラーのレプリケーションモードを有効にします。以下の形式を取ります。

```
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。**serverID** は、**syncrepl** オプションと共に指定する必要があります。詳細は、**18.3.4.OpenLDAP ソフトウェア管理ガイド** の MirrorMode セクション (「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 レプリケーションモードを有効にします。以下の形式を取ります。

```
olcSyncrepl on
```

同期のレプリケーションモードでは、プロバイダーとコンシューマーの両方に特定の設定が必要になります。この設定は、**OpenLDAP ソフトウェア管理ガイド** の **18.3.1. Syncrepl** セクション (「インストールされているドキュメント」) を参照してください。

9.2.3.8. モジュールとバックエンドの読み込み

動的にロードされたモジュールを使用して **slapd** サービスを強化できます。これらのモジュールのサポートは、**slapd** の設定時に **--enable-modules** オプションを使用して有効にする必要があります。モジュールは、拡張子が **.la** のファイルに保存されます。

```
module_name.la
```

LDAP 要求に対応して、*バックエンド* がデータを保存または取得します。バックエンドは、静的に **slapd** にコンパイルするか、モジュールサポートが有効になっている場合は、動的に読み込むことができます。後者の場合は、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、**/etc/openldap/slapd.d/** で以下のディレクティブを使用します。

olcModuleLoad

olcModuleLoad ディレクティブは、読み込むモジュール (動的に読み込み可能) を指定します。以下の形式を取ります。

```
olcModuleLoad: module
```

ここで、*module* は、読み込むモジュールまたはバックエンドを含むファイルを表します。

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。デフォルトでは、SELinux は、アプリケーションが OpenLDAP サーバーにアクセスできないようにします。複数のアプリケーションで必要な LDAP による認証を有効にするには、SELinux ブール値 **allow_ypbind** を有効にする必要があります。特定のアプリケーションでは、このシナリオで有効な **authlogin_nsswitch_use_ldap** ブール値も要求します。次のコマンドを実行して、前述のブール値を有効にします。

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

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

-P オプションを使用すると、システムを再起動してもこの設定が維持されます。SELinux に関する詳細情報は、Red Hat Enterprise Linux 7 SELinux User's and Administrator's Guide も併せて参照してください。

9.2.5. OpenLDAP サーバーの実行

本セクションでは、**スタンドアロン LDAP デーモン の開始、停止、再起動、および現在のステータスの確認方法を説明します**。システムサービス全般を管理する方法は、『System Administrator's Guide』のManaging Services with systemdを参照してください。

9.2.5.1. サービスの起動

現行セッションで **slapd** サービスを起動するには、**root** で次のコマンドを実行します。

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

システムの起動時にサービスが自動的に起動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

現行セッションで実行中の **slapd** サービスを停止するには、**root** で次のコマンドを実行します。

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

システムの起動時にサービスが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
```

これにより、サービスが停止し、すぐに再起動します。以下のコマンドを使用して、設定を再読み込みします。

9.2.5.4. サービスステータスの確認

slapd サービスが実行していることを確認するには、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 を使用してシステムを認証するためのシステムの設定

OpenLDAP を使用してシステムを認証するように設定するには、適切なパッケージが LDAP サーバーとクライアントマシンの両方にインストールされていることを確認してください。サーバーの設定方法は、「OpenLDAP スイートのインストール」 および 「OpenLDAP サーバーの設定」 の手順に従います。クライアントで、シェルプロンプトで以下を入力します。

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

9.2.6.1. 以前の認証情報の LDAP 形式への移行

migrationtools パッケージは、認証情報を LDAP 形式に移行するのに役立つシェルおよび 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、シェルプロンプトで以下を入力します。

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

これにより、スクリプトが **/usr/share/migrationtools/** ディレクトリーにインストールされます。インストールが完了したら、**/usr/share/migrationtools/migrate_common.ph** ファイルを編集し、以下の行を変更して正しいドメインを反映させます。

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

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

または、コマンドラインで直接環境変数を指定することもできます。たとえば、デフォルトのベースを **dc=example,dc=com** に設定して **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を移行するために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 が実行しているか？	使用するスクリプト
/etc フラットファイル	はい	**migrate_all_online.sh**
/etc フラットファイル	いいえ	**migrate_all_offline.sh**
NetInfo	はい	**migrate_all_netinfo_online.sh**
NetInfo	いいえ	**migrate_all_netinfo_offline.sh**
NIS (YP)	はい	**migrate_all_nis_online.sh**
NIS (YP)	いいえ	**migrate_all_nis_offline.sh**

これらのスクリプトの使用方法は、**/usr/share/doc/migrationtools-*version*/** ディレクトリーの **README** ファイルおよび **migration-tools.txt** ファイルを参照してください。

9.2.7. 関連情報

以下のリソースは、Lightweight Directory Access Protocol に関する追加情報を提供します。システムで LDAP を設定する前に、『OpenLDAP Software 管理者ガイド』 など、これらのリソース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 ソフトウェア管理者ガイド』 のコピー。
- **/usr/share/doc/openldap-servers-*version*/README.schema**: インストールされたスキーマファイルの説明が含まれる README ファイル。

また、パッケージ openldap、openldap-servers、および openldap-clients でインストールされる man ページも多数あります。

クライアントアプリケーション

- ldapadd(1) : **ldapadd** コマンドの man ページでは、LDAP ディレクトリーにエントリーを追加する方法を説明します。
- ldapdelete(1) : **ldapdelete** コマンドの man ページでは、LDAP ディレクトリー内のエントリーを削除する方法を説明します。
- ldapmodify(1) : **ldapmodify** コマンドの man ページでは、LDAP ディレクトリー内のエントリーを変更する方法を説明します。
- ldapsearch(1) : **ldapsearch** コマンドの man ページでは、LDAP ディレクトリー内のエントリーを検索する方法を説明します。

- ldappasswd(1) : **ldappasswd** コマンドの man ページでは、LDAP ユーザーのパスワードを設定または変更する方法を説明します。
- ldapcompare(1) : **ldapcompare** ツールの使用方法を説明します。
- ldapwhoami(1) : **ldapwhoami** ツールの使用方法を説明します。
- ldapmodrdn(1): エントリーの RDN を変更する方法を説明します。

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を説明します。

管理アプリケーション

- slapadd(8C) : **slapd** データベースにエントリーを追加するために使用されるコマンドラインオプションを説明します。
- slapcat(8C) : **slapd** データベースから LDIF ファイル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を説明します。
- slapindex(8C) : **slapd** データベースの内容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を説明します。
- 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のに使用されるコマンドラインオプションを説明します。

設定ファイル

- ldap.conf(5) : **ldap.conf** ファイルの man ページでは、LDAP クライアントの設定ファイル内で利用可能な形式およびオプションが説明されています。
- slapd-config(5) : **/etc/openldap/slapd.d** 設定ディレクトリーで利用可能な形式およびオプションを説明します。

その他リソース

- 『OpenLDAP および Mozilla NSS 互換性レイヤー』 NSS データベース後方互換性に関する実装の詳細。
- 『How do I use TLS/SSL?』OpenLDAP が OpenSSL を使用するように設定する方法に関する情報

パート III. セキュアなアプリケーション

このパートでは、**PAM**(**Pluggable Authentication Modules**)の使用方法、**Kerberos** 認証プロトコルと **certmonger** デーモンの使用方法、最後に **シングルサインオン** (**SSO**)用にアプリケーションを設定する方法の詳細を提供します。

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) は、認証および認可の一般的なフレームワークです。Red Hat Enterprise Linux のシステムアプリケーションの多くは、認証および承認の基礎となる PAM 設定により異なります。

10.1. PAM について

Pluggable Authentication Module(PAM) は、システムアプリケーションが中央で設定されたフレームワークに認証を中継するのに使用できる集中認証メカニズムを提供します。

PAM は、さまざまな種類の認証ソース (Kerberos、SSSD、NIS、またはローカルファイルシステムなど) に PAM モジュールがあるため、プラグ可能な動作になります。異なる認証ソースの優先順位を設定できます。

このモジュラーアーキテクチャーにより、管理者はシステムの認証ポリシーを柔軟に設定することができます。PAM は、開発者および管理者にとって以下のような便利なシステムです。

- 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、システム管理者に対して、優れた柔軟性と制御性を提供します。
- 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. その他の PAM リソース

PAM には、PAM の使用と PAM を他のアプリケーションと統合するためのモジュールの作成に関する詳細が記載されている広範囲なドキュメントセットがあります。PAM を使用する主要なモジュールと設定ファイルのほとんどすべてには、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、『システム管理者ガイド』、『モジュールライターのマニュアル』、および『アプリケーション開発者のマニュアル』、および PAM 標準 DCE-RFC 86.0 のコピーが含まれています。

http://www.linux-pam.org で PAM 用のライブラリーを利用できます。これは、Linux-PAM プロジェクトの主なディストリビューション web サイトです。これには、さまざまな PAM モジュールに関する情報、よくある質問、および追加の PAM ドキュメンテーションが含まれています。

10.1.2. カスタム PAM モジュール

新しい PAM モジュールは、PAM 対応アプリケーションで使用するためにいつでも作成または追加できます。PAM 対応プログラムは新しいモジュールとメソッドをすぐに使用できます。これは、コンパイルなし、あるいは変更なしで定義します。これにより、開発者およびシステム管理者は、再コンパイルせずに異なるプログラムに認証モジュールの選択やテストを使用できます。

モジュール作成に関するドキュメントは、**/usr/share/doc/pam-devel-version#/** ディレクトリーに含まれています。

10.2. PAM 設定ファイルについて

PAM 対応のアプリケーションまたは *サービス* ごとに、**/etc/pam.d/** ディレクトリーに ファイルがあります。このディレクトリーの各ファイルは、アクセスを制御するサービスと同じ名前を持ちます。たとえば、 **login** プログラムはログインとしてサービス名を定義し、**/etc/pam.d/login** PAM 設定ファイルをインストールします。

警告

PAM 設定ファイルを手動で編集するのではなく、**authconfig** ツールを使用して PAM を設定することを強く推奨します。

10.2.1. PAM 設定ファイル形式

各 PAM 設定ファイルには、モジュール (認証設定領域) とその制御または引数を定義するディレクティブが含まれます。

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 (インターフェイス) 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イスは以下のように最初のフィールドで定義されます。以下に例を示します。

```
auth required pam_unix.so
```

PAM **インターフェイス** は、基本的に特定のモジュールを実行できる認証アクションのタイプです。4 種類の PAM モジュールインターフェイスが利用できます。それぞれは、認証および承認プロセスのさまざまな側面に対応します。

- **auth**: このモジュールインターフェイ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、検証します。このインターフェイスを持つモジュールは、グループメンバーシップなどの認証情報を設定することもできます。
- **account**: このモジュールインターフェイスは、アクセスが許可されていることを確認します。たとえば、ユーザーアカウントの有効期限が切れたか、または特定の時間にユーザーがログインできるかどうかを確認します。
- **password**: このモジュールインターフェイスは、ユーザーパスワードの変更に使用されます。
- **session**: このモジュールインターフェイスはユーザーセッションを設定および管理します。このインターフェイスのあるモジュールは、ユーザーのホームディレクトリーをマウントしたり、ユーザーのメールボックスを利用可能にするなど、アクセスを許可するために必要な追加の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イスまたはすべてのモジュールインターフェイスを提供できます。たとえば、**pam_unix.so** は 4 つのモジュールインターフェイスをすべて提供します。

pam_unix.so などのモジュール名は、PAM に、指定されたモジュールインターフェイスを含むライブラリーの名前を提供します。アプリケーションが適切なバージョンの **libpam** にリンクされているため、ディレクトリー名は省略され、モジュールの正しいバージョンを見つけることができます。

すべての PAM モジュールは、呼び出されると成功または失敗の結果を生成します。**制御フラグ** は、結果をどう処理するかを PAM に指示します。モジュールは特定の順序で一覧表示 (**スタック**) でき、制御フラグは特定モジュールの成功または失敗の重要性を判断し、ユーザーをサービスに対して認証する際の全体的な目標を決定します。

単純なフラグがいくつかあります。[2]設定にはキーワードのみを使用します。

- required: 認証を続行するには、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、そのインターフェイ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。

- **必須** - 認証を続行するには、モジュールの結果が正常に実行される必要があります。ただし、この時点でテストが失敗すると、最初に失敗した **required** または **requisite** モジュールテストを反映したメッセージとともに、すぐにユーザーに通知されます。

- **sufficient** - モジュールが失敗する場合、結果は無視されます。ただし、モジュールフラグ付きの **sufficient** の結果が成功し、以前のモジュールフラグ付きの **required** が失敗していない場合は、その他の結果は不要で、ユーザーはサービスに対して認証されます。

- ***オプション*** - *モジュールの結果は無視されます。他のモジュールがインターフェイスを参照していない場合に、認証を成功させるには、**optional** としてフラグが付いたモジュールが必要です。*

- ***include***: *他の制御とは異なり、モジュールの結果の処理方法とは関係ありません。このフラグは、指定のパラメーターに一致する設定ファイルのすべての行でプルし、それらをモジュールに引数として追加します。*

*モジュールインターフェイ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で **sufficient** 値または **requisite** 値が使用される場合、モジュールを一覧表示する順序は認証プロセスで重要になります。*

*管理者は、スタッキングを使用して、ユーザーが認証を許可される前に、特定の条件が存在することを要求できます。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、いくつかのスタック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、UID が 0 であることを確認することで、**pam_rootok.so** モジュールを使用して現在のユーザーがroot かどうかを確認します。このテストに成功すると、他のモジュールは参照されず、コマンドが実行されます。このテストが失敗すると、次のモジュールが参照されます。*

- ***auth include system-auth***: *この行には、**/etc/pam.d/system-auth** モジュールの内容が含まれ、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** モジュールを使用して、root ユーザーまたはコンソールにログインしているすべてのユーザーがシステムを再起動します。*

- ***session required pam_permit.so*** - *この行はセッション設定に関連します。**pam_permit.so** を使用すると、**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、いくつかの引数を取ることができます。以下の例では、**enforce_for_root** は、root ユーザーのパスワードでも強度チェックに合格する必要があることを指定し、**retry** は、ユーザーが強力なパスワードを入力する 3 つの機会を受け取ることを定義します。

```
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を及ぼしません。ただし、一部のモジュールは無効な引数で失敗する可能性があります。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** および関連する **journalctl** ツールの使用方法は、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。以前のバージョンの Red Hat Enterprise Linux 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. アノテーション付き 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
```

- 最初の行はコメントで、行頭のハッシュマーク(**#**)で示されます。
- 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。

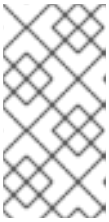
  auth required pam_securetty.so - このモジュールは、ユーザーが root としてログインしようとしている **場合** に、そのファイルが存在する **場合** は、ユーザーがログインする TTY が **/etc/securetty** ファイルに一覧表示されます。

  TTY がファイルに記載されていない場合は、root でログインしようとすると、**Login incorrect** メッセージを表示して失敗します。

  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** に保存されている情報を使用してパスワードをチェックし、存在する場合は **/etc/shadow**。

  引数は、**pam_unix.so** モジュールに空のパスワードを許可するように **nullok** に指示します。

- ***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。ユーザーが存在して 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 つの ***auth*** モジュールがすべてチェックされます。これにより、ユーザーは認証に失敗したステージを把握できません。攻撃者のこのような知識により、システムのクラッキング方法がより簡単に推測される可能性があります。

- ***account required pam_unix.so*** - このモジュールは、必要なアカウントの検証を実行します。たとえば、シャドウパスワードが有効になっている場合、***pam_unix.so*** モジュールのアカウントインターフェイスは、アカウントの有効期限が切れたかどうか、または許可された猶予期間内にユーザーがパスワードを変更していないかどうかを確認します。

- ***password required pam_pwquality.so retry=3*** - パスワードの有効期限が切れると、***pam_pwquality.so*** モジュールのパスワードコンポーネントは新しいパスワードを要求します。その後、新たに作成されたパスワードをテストして、辞書ベースのパスワードクラッキングプログラムで簡単に判別できるかどうかを確認します。

  引数 ***retry=3*** は、テストが初めて失敗すると、強力なパスワードを作成する可能性が 2 つあります。

- ***password required pam_unix.so shadow nullok use_authtok*** - この行は、***pam_unix.so*** モジュールの password インターフェイスを使用して、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を更新する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、ユーザーが空のパスワード ***から*** パスワードを変更できるようにするようにモジュールに指示します。それ以外の場合は、null パスワードはアカウントロックとして扱われます。

  - この行の最後の引数 ***use_authtok*** は、PAM モジュールをスタックする際の順序の重要性の例を提供します。この引数は、ユーザーに新しいパスワードを要求しないように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で記録されたパスワードを受け入れます。このようにして、新しいパスワードはすべて、受け入れられる前に、安全なパスワードについて ***pam_pwquality.so*** テストを渡す必要があります。

- ***session required pam_unix.so*** - 最後の行は、***pam_unix.so*** モジュールのセッションインターフェイスにセッションを管理します。このモジュールは、各セッションの開始と終了時に、ユーザー名とサービスタイプを ***/var/log/secure*** に記録します。このモジュールは、追加機能のために他のセッションモジュールとスタックすることで補足できます。

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 の多くのグラフィカル管理ツールは、***pam_timestamp.so*** モジュールを使用して最大 5 分間ユーザーに昇格された特権を提供します。このメカニズムの仕組みを理解することが重要です。これは、***pam_timestamp.so*** が有効であるときに端末から離れたユーザーが、コンソールに物理的にアクセスできるユーザーによってマシンを開いたままにするためです。

PAM タイムスタンプスキームでは、グラフィカル管理アプリケーションにより、起動時に root パスワードの入力が求められます。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。タイムスタンプファイルがすでに存在する場合は、グラフィカル管理プログラムではパスワードの入力が求められません。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最新の状態にし、ユーザーの不完全な管理アクセスを 5 分追加で保持します。

/var/run/sudo/ユーザー ディレクトリーのファイルを確認して、タイムスタンプファイルの実際の状態を確認できます。デスクトップでは、関連するファイルは **unknown:root** です。これが存在し、タイムスタンプが 5 分未満の場合は、認証情報が有効です。

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

[D]

10.3.1. 一般的な pam_timestamp ディレクティブ

pam_timestamp.so モジュールは、以下の 2 つのインターフェイスを提供します。

- **auth**
- **session**

さらに、**pam_timestamp.so** では、以下のオプションを利用できます。

- **timestamp_timeout**: タイムスタンプファイルの有効期間（秒単位）を指定します。デフォルトは 300 (5 分) です。
- **timestampdir**: タイムスタンプファイルを保存するディレクトリーを指定します。デフォルトは **/var/run/sudo/** です。
- 詳細メッセージには **verbose** または **debug** を使用することもできます。

以下に例を示します。

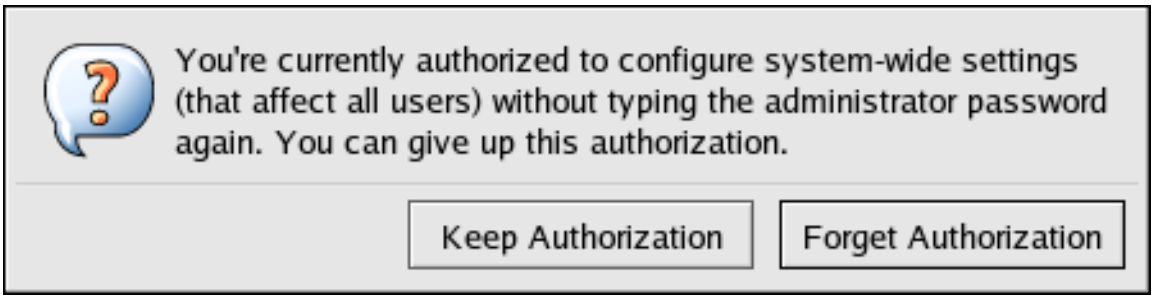
```
auth       sufficient    pam_timestamp.so timestamp_timeout=600
session    optional      pam_timestamp.so
```

PAM のディレクティブの使用および設定方法は、「PAM 設定ファイルについて」 を参照してください。**pam_timestamp (8)** man ページと **pam.conf (5**) の man ページも参照してください。

10.3.2. 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コンソールを利用する前に、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行うには、パネルの認証アイコンをクリックします。これにより、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



[D]

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。*
- *特権アプリケーションが起動されたものと同じターミナルウィンドウから **/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。このコマンドは、root で実行しないでください。*
- *アイコン上の **Forget Authorization** アクションを使用せずにデスクトップで認証情報を強制終了するには、**/sbin/pam_timestamp_check** 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方法は、**pam_timestamp_check** の man ページを参照してください。*

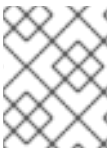
10.4. PAM サービスのドメイン制限

重要

この機能では、システムで SSSD を実行する必要があります。

SSSD を使用すると、PAM サービスがアクセスできるドメインを制限できます。SSSD は、特定の PAM サービスが実行中のユーザーに基づいて 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 サービスユーザーがドメインにアクセスできるかどうかによって異なります。

サンプルユースケースは、外部ユーザーが FTP サーバーへの認証を行える環境です。FTP サーバーは、権限のない別のユーザーとして実行します。このユーザーは、内部の企業アカウントとは別に、選択した SSSD ドメインに対してのみ認証できます。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_ldap** などのレガシー PAM モジュールと似ていますが、個別の設定ファイルを PAM モジュールのパラメーターとして使用できました。*

ドメインへのアクセスを制限するオプション
選択したドメインへのアクセスを制限するには、以下のオプションを使用できます。

pam_trusted_users* in */etc/sssd/sssd.conf

*このオプションは、SSSD が信頼する PAM サービスを表す数値の UID またはユーザー名の一覧を受け入れます。デフォルト設定は all です。これは、**すべて** のサービスユーザーが信頼され、任意のドメインにアクセスできることを意味します。*

pam_public_domains* in */etc/sssd/sssd.conf

*このオプションは、パブリック 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、信頼できない PAM サービスユーザーであってもドメインにアクセスできます。オプションは、**all** および **none** の値も受け入れます。デフォルト値は **none** です。つまり、ドメインは公開されておらず、信頼でき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 サービスが認証できるドメインの一覧を指定します。ドメインを指定せずに **domains** を使用する場合、PAM サービスはドメインに対して認証できなくなります。*

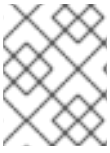
```
auth     required   pam_sss.so domains=
```

***ドメイン** が PAM 設定ファイルで使用されていない場合、PAM サービスは、サービスが信頼できるユーザーで実行している条件で、すべてのドメインに対して認証できます。*

***/etc/sssd/sssd.conf** SSSD 設定ファイルの **domains** オプションは、SSSD が認証を試行するドメインの一覧も指定します。PAM 設定ファイルの **domains** オプションは、**sssd.conf** のドメイン一覧を拡張することができないことに注意してください。短いリストを指定することで、ドメインの **sssd.conf** リストを制限することしかできません。したがって、ドメインが PAM ファイルで指定され、**sssd.conf** で指定されていない場合、PAM サービスは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 **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、任意のドメインにアクセスできることを示しています。この場合、PAM 設定ファイルの **domains** オプションを使用して、アクセスできるドメインを制限することができます。*

***sssd.conf** に **pam_public_ domains** が含まれているときに PAM 設定ファイルでドメインを使用してドメインを指定する場合は、**pam_public_domains** でもドメインを指定する必要がある場合があります。**pam_public_domains** が使用されていても、必要なドメインが含まれていない場合、PAM サービスは信頼できないユーザーで実行していると、ドメインに対して正常に認証できなくなります。*



注記

PAM 設定ファイルで定義されるドメイン制限は、ユーザールックアップではなく、認証アクションにのみ適用されます。

***pam_trusted_users** オプションおよび **pam_public_domains** オプションの詳細は、sssd.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される **domains** オプションの詳細は、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の制限

PAM サービスが認証できるドメインを制限するには、次のコマンドを実行します。

1. *必要なドメインにアクセスするように SSSD が設定されていることを確認してください。SSSD が認証できるドメインは、**/etc/sssd/sssd.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 サービスが認証できるドメインを指定します。これには、PAM 設定ファイルに **domains** オプションを設定します。以下に例を示します。*

```
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** に対してのみ認証できるようになりました。*

---

*[2] 設定可能な制御フラグは数多くあります。これらは attribute=value ペアで設定されます。属性の完全なリストは **pam.d** の man ページにあります。*

第11章 KERBEROS の使用

ネットワーク内でシステムのセキュリティーと整合性を維持することは重要です。また、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、およびサーバーが含まれます。これには、ネットワーク上で実行中のすべての内容と、これらのサービスが使用される仕組みを理解する必要があります。このセキュリティーの維持の中核となるのは、これらのアプリケーションおよびサービスへの **アクセス** の維持と、そのアクセスの実施です。

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な認証プロトコルです。Kerberos では、他のマシンでサービスにアクセスした場合でも、パスワードがネットワーク経由で送信されることはありません。

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およびサービスへの定義済みかつ制限されたアクセスを受け取れるようにするメカニズムを提供します。Kerberos はアイデンティティーを確認してエンティティーを **認証** します。また、Kerberos はこの認証データも保護し、外部からアクセス、使用、改ざんされないようにします。

11.1. KERBEROS について

Kerberos は対称キー暗号を使用して[3] 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上で送信されることはありません。

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 の仕組みの基本

従来の多くのネットワークサービスは、パスワードベースの認証スキームを使用しており、ユーザーは特定のネットワークサーバーにアクセスするためのパスワードを提供します。ただし、多くのサービスに対する認証情報の送信は暗号化されません。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できないようにする必要があり、ネットワーク上のすべてのコンピューターおよびユーザーが信頼でき、信頼でき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では、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想定することはできません。ネットワークへのアクセスを取得する攻撃者は、シンプルなパケットアナライザー (**パケットスニッファー**) 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を破ります。そのため、セキュリティーインフラストラクチャー全体の整合性が保たれます。

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各ユーザーを個別に認証するのではなく、Kerberos は対称暗号化と信頼できるサードパーティー (**キー配布センター** KDC) を使用し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** を設定します。

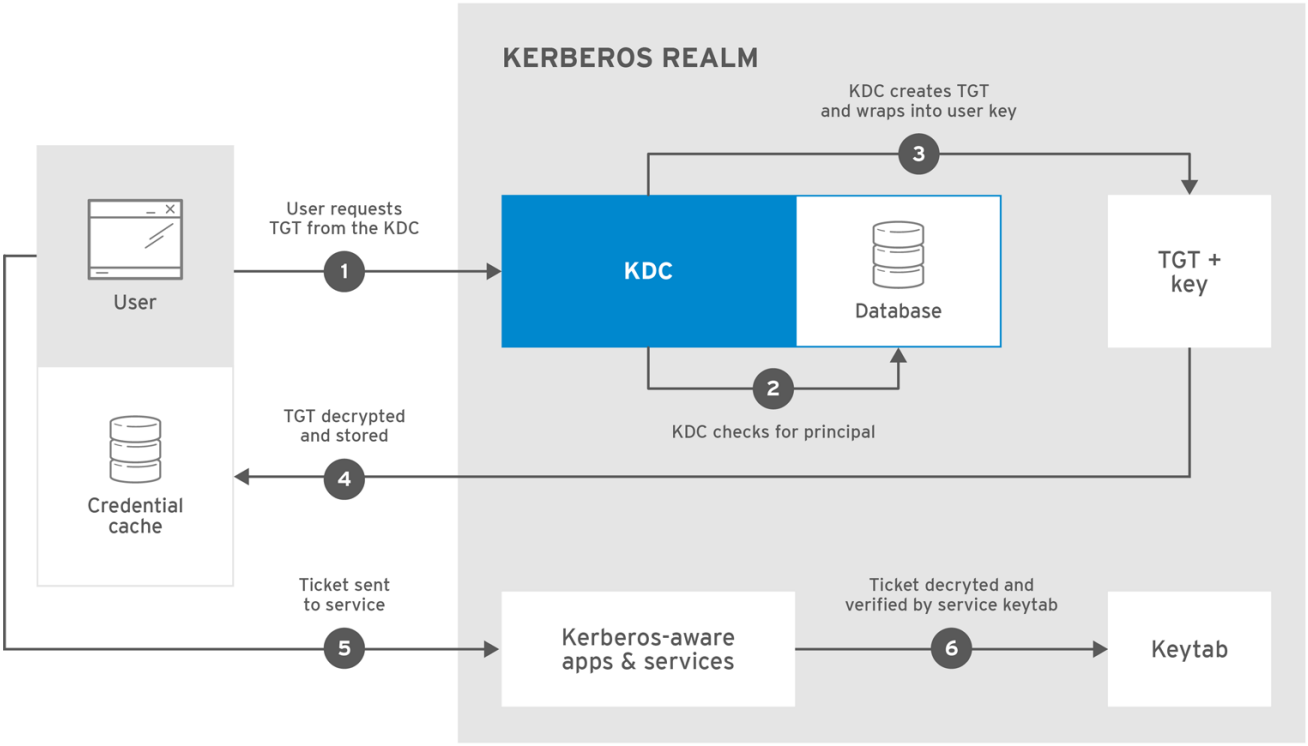
ユーザーがKDC に対して認証を行うと、KDC はそのセッションに特定した認証情報のセット (**チケット**) をユーザーのマシンに送り返します。Kerberos 対応のサービスでは、ユーザーがパスワードを使用して認証する必要はなく、サービスすべてがユーザーのマシン上でこのチケットを探します。

図11.1「Kerberos 認証」 に示すように、各ユーザーは **プリンシパル** と呼ばれる一意のアイデンティティーでKDC に識別されます。Kerberos 対応のネットワーク上のユーザーがワークステーションにログインすると、認証サーバーから **ticket-granting ticket** (TGT) の要求の一部としてプリンシパルが

KDC に送信されます。この要求は、ログインプログラムによりユーザーに透過的となるようにしたり、ユーザーがログインした後に **kinit** プログラムを介してユーザーが手動で送信したりすることができます。

次に KDC はデータベース内でプリンシパルをチェックします。プリンシパルが見つかると、KDC は TGT を作成し、ユーザーのキーを使用してこれを暗号化し、TGT をそのユーザーに送信します。

図11.1 Kerberos 認証



クライアント上のログインまたは **kinit** プログラムは、ユーザーのパスワードから計算するユーザーのキーを使用して TGT を復号化します。ユーザーのキーはクライアントマシン上でのみ使用され、ネットワーク上では送信**されません**。KDC によって送信されるチケット (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、Kerberos 対応のサービスで確認できます。Red Hat Enterprise Linux 7 では、以下のタイプの認証情報キャッシュがサポートされます。

- Red Hat Enterprise Linux 7 のデフォルトのキャッシュである KEYRING ccache タイプ
- System Security Services Daemon (SSSD) Kerberos Credential Manager (KCM) (Red Hat Enterprise Linux 7.4 の代替オプション)
- FILE
- DIR
- MEMORY

SSSD KCM では、Kerberos キャッシュはパッシブストアに保存されず、デーモンにより管理されます。この設定では、**kinit** などのアプリケーションによって通常使用される Kerberos ライブラリーは KCM クライアントであり、デーモンは KCM サーバーと呼ばれます。

SSSD KCM デーモンが管理する Kerberos 認証情報キャッシュには、以下のような利点があります。

- デーモンはステートフルであり、Kerberos 認証情報キャッシュの更新や古い ccache の取得などのタスクを実行できます。更新と追跡は、SSSD 自体が取得したチケット（通常は

pam_sss.so を介したログインを介して）や、kinit などで取得したチケットに対しても可能となります。

- *プロセスはユーザースペースで実行されるため、カーネル KEYRING とは異なり、UID 名前空間の影響を受けます。*
- *呼び出し元の UID に完全に依存し、コンテナー化された環境ではすべてのコンテナー間で共有されるカーネル KEYRING ベースのキャッシュとは異なり、KCM サーバーのエントリーポイントは、選択したコンテナーにのみバインドマウントできる UNIX ソケットです。*

認証後、サーバーは kinit をチェックするのではなく、認識されたプリンシパルとそのキーの暗号化されていない一覧を確認できます。これはキータブに保持さ れ ます。

TGT は、一定期間 (通常は 10 から 24 時間) の後に期限切れに設定され、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。セキュリティーの破られた TGT が攻撃者に利用される時間を短くするために、有効期限が設定されています。TGT の発行後、TGT の有効期限が切れるまで、もしくは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はありません。

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、クライアントソフトウェアは TGT を使用して ticket-granting サーバー (TGS) からその特定のサービスの新しいチケットを要求します。サービスチケットはその後、そのサービスに対して透過的にユーザーを認証するために使用されます。

11.1.2. Kerberos プリンシパル名について

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、識別子 は Kerberos ユーザー名のみになります。サービスの場合、識別子 はサービス名と、それが実行するマシンのホスト名の組み合わせです。

```
service/FQDN@REALM
```

サービス 名は、ホスト、ldap、http、DNS など、サービスタイプに固有の大文字と小文字を区別する文字列です。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。

以下に例を示します。

```
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 サーバーは解決されたホスト名 web-01.example.com@EXAMPLE.COM のチケットを要求するため、ホストプリンシパルは host/ web-01.example.com@EXAMPLE.COM である必要があります。

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、これで十分ですが、派生したレルム名は存在しないレルムの名前になります。このような場合、サーバーの DNS ドメイン名からレルムの名前へのマッピングは、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。以下に例を示します。

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

この設定では、2 つのマッピングを指定します。最初のマッピングは、example.com DNS ドメイン内のシステムが **EXAMPLE.COM** レルムに属することを指定します。2 つ目は、名前が example.com のシステムがレルムにあることを指定します。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マークされます。たとえば、"_kerberos TXT" レコードを使用して、マッピングを DNS に直接保存することもできます。

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

11.1.4. 環境要件

Kerberos ではマシン名を解決できる必要があります。そのため、DNS(ドメインネームサービス) が必要です。ネットワーク上の DNS エントリーとホストの両方を適切に設定する必要があります。これは、**/usr/share/doc/krb5-server-**version- **number**の Kerberos ドキュメントで説明されています。

Kerberos 認証を受け入れるアプリケーションには、時間同期が必要です。**ntpd** などのサービスを使用して、ネットワーク上のマシン間のクロック同期を概算することができます。**ntpd** サービスの詳細は、**/usr/share/doc/ntp-**version-number**/html/index.html** または ntpd(8) の man ページを参照してください。

注記

Red Hat Enterprise Linux 7 を実行している Kerberos クライアントは、KDC の自動調整をサポートします。したがって、タイミングの要件は厳格ではありません。これにより、Red Hat Enterprise Linux 7 で IdM クライアントをデプロイする際のクロッキングの違いに対する耐性が向上します。

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ー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、各ユーザーが信頼されていて、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前提としています。その主な目的は、暗号化されていないパスワードがそのネットワーク上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ホスト (KDC) に適切なユーザー以外のユーザーがアクセスすると、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、そのソースを変更して 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。この方法で変更したアプリケーションは、Kerberos 対応 として考慮されます。アプリケーションによっては、アプリケーションのサイズや設計により、非常に問題になる場合があります。その他の互換性のない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に変更を加える必要があります。ここでも、大幅なプログラミングが必要になる場合があります。デフォルトでは Kerberos サポートのないクローズソースアプリケーションは、多くの場合最も問題となります。*

- *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*** *クライアントアプリケーションおよび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アプリケーションおよび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*** *などの標準の 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にユーザーパスワードを移行するのは面倒です。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 のデプロイ方法によって大きく異なる可能性があります。このため、Identity Management 機能を使用することが推奨されます。これには、移行用の特別なツールおよびメソッドがあります。*

警告

ネットワーク上のユーザーが Kerberos 以外の対応サービスに対してパスワードをプレーンテキストで送信して認証すると、Kerberos システムが危険にさらされる可能性があります。Kerberos 以外の対応サービス (telnet や FTP など) の使用は強く推奨されません。SSH や SSL で保護されたサービスなどの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も推奨されますが、これは理想的ではありません。

11.1.6. Kerberos に関するその他のリソース

Kerberos は、デプロイ方法に柔軟性が高い、実装する複雑なサービスになります。表11.1「外部の Kerberos 資料」 また、表11.2「重要な Kerberos の man ページ」 には、Kerberos の使用に関する詳細情報として、最も重要なソースまたは最も有用なソースの一覧が記載されています。

表11.1 外部の Kerberos 資料

資料	場所
Kerberos V5 Installation Guide (PostScript と HTML の両方)	**/usr/share/doc/krb5-server-*version-number***
Kerberos V5 System Administrator's Guide (PostScript と HTML の両方)	**/usr/share/doc/krb5-server-*version-number***

資料	場所
Kerberos V5 UNIX User's Guide (PostScript と HTML の両方)	**/usr/share/doc/krb5-workstation-*version-number***
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』。原板は Bill Bryant により 1988 年にリリースされ、1997 年に Theodore Ts'o によって改訂されています。本書は、Kerberos スタイルの認証システムの作成を考慮している 2 人の開発者の会話を表しています。この対話形式のディスカッションを参照することで、Kerberos に完全に精通していない人にとって、良い開始地点となります。	http://web.mit.edu/kerberos/www/dialogue.html
ネットワーク Kerberos に対応するための記事。	http://www.ornl.gov/~jar/HowToKerb.html

man **command_name** *を実行すると、いずれかの man ページファイルを開くことができます。*

表11.2 重要な Kerberos の man 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 システムの概要では、認証情報がどのように機能するかを説明し、Kerberos チケットの取得および破棄に関する推奨事項を提供します。man ページの下部では、関連する man ページが多数参照されています。
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。

man ページ	説明
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。
kdc.conf	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。

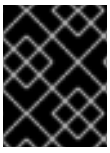
11.2. KERBEROS KDC の設定

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 を導入する方法として、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である必要があります。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、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** および **/var/kerberos/krb5kdc/kdc.conf** 設定ファイルを編集して、レルム名とドメインからレルムへのマッピングを反映します。以下に例を示します。*

```
[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 のインスタンスを正しいドメイン名で置き換えることで設定できます。これは、正しい形式で大文字と小文字の名前を維持することが確実にでき、KDC を kerberos.example.com から Kerberos サーバーの名前に変更することで設定できます。通常、レルム名はすべて大文字で、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** ユーティリティーを使用してデータベースを作成します。*

```
[root@server ~]# kdb5_util create -s
```

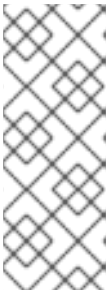
***create** コマンドは、Kerberos レルムのキーを保存するデータベースを作成します。**-s** 引数は、マスターサーバーキーが保存される stash ファイルを作成します。キーの読み取り元となる stash ファイルがない場合、Kerberos サーバー(**krb5kdc**)は起動時に毎回マスターサーバーパスワード（キーを再生成するために使用できる）を要求します。*

4. ***/var/kerberos/krb5kdc/kadm5.acl** ファイルを編集します。このファイルは、Kerberos データベースへの管理アクセス権とそのアクセスレベルを決定するために **kadmind** によって使用されます。以下に例を示します。*

```
*/admin@EXAMPLE.COM  *
```

多くのユーザーは、データベース内で単一のプリンシパルで表されます (joe@EXAMPLE.COM などの NULL または空のインスタンス)。この設定では、admin (例: joe/admin@EXAMPLE.COM) のインスタンスを持つ 2 番目のプリンシパルを持つユーザーは、レルムの Kerberos データベースに対する完全な管理制御を強化できます。

***kadmind** がサーバーで起動した後、ユーザーはレルム内のいずれかのクライアントまたはサーバーで **kadmin** を実行してサービスにアクセスできます。ただし、**kadm5.acl** ファイルにリストされているユーザーのみが、自身のパスワードを変更することを除いて、いつでもデータベースを変更できます。*



注記

kadmin ユーティリティーはネットワーク経由で **kadmind** サーバーと通信し、Kerberos を使用して認証を処理します。したがって、ネットワーク経由でサーバーに接続してサーバーを管理するには、最初のプリンシパルがすでに存在している必要があります。**kadmin.local** コマンドを使用して最初のプリンシパルを作成します。これは、KDC と同じホストで使用するように特別に設計されており、認証に Kerberos を使用しません。

5. KDC 端末で **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 kadmin.service
```

7. **kadmin** 内で **addprinc** コマンドを使用してユーザーのプリンシパルを追加します。**kadmin** および **kadmin.local** は、KDC へのコマンドラインインターフェイスです。そのため、**addprinc** などの多くのコマンドは、**kadmin** プログラムの起動後に利用できます。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** を使用してキャッシュ内の認証情報の一覧を表示し、**kdestroy** を使用してキャッシュと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、(Kerberos サーバーではなく)同じシステムログインユーザー名を使用して認証を試みます。ユーザー名が Kerberos データベースのプリンシパルに対応しない場合は、**kinit** がエラーメッセージを発行します。その場合は、コマンドラインの引数として、正しいプリンシパルの名前で **kinit** を指定します。

```
kinit principal
```

11.2.2. セカンダリー KDC の設定

特定のレルムに複数の KDC がある場合は、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書き込み可能なコピーを保持し、**kadmind** を実行します。マスター KDC もレルムの **管理サーバー** です。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、**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
```

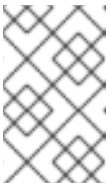
2. マスターKDC の **krb5.conf および kdc.conf** ファイルをセカンダリーKDC にコピーします。

3. マスターKDC の root シェルから **kadmin.local** を起動します。

   a. **kadmin.local add_principal** コマンドを使用して、マスターKDC の **host** サービスの新規エントリーを作成します。

```
[root@masterkdc ~]# 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 WRFILE:/etc/krb5.keytab.
Entry for principal host/masterkdc.example.com with kvno 3, encryption type DES with
HMAC/sha1 added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal host/masterkdc.example.com with kvno 3, encryption type DES cbc
mode with RSA-MD5 added to keytab WRFILE:/etc/krb5.keytab.
kadmin: quit
```

   b. **kadmin.local ktadd** コマンドを使用してサービスにランダムにキーを設定し、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

このキーは、**kprop** コマンドがセカンダリーサーバーに認証するために使用されます。インストールするセカンダリーKDC サーバーの数にかかわらず、これは一度だけ実行する必要があります。

4. セカンダリーKDC で root シェルから **kadmin** を起動します。

   a. **kadmin.local add_principal** コマンドを使用して、セカンダリーKDC の **host** サービス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 WRFILE:/etc/krb5.keytab.
Entry for principal host/slavekdc.example.com with kvno 3, encryption type ArcFour with
HMAC/md5 added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal host/slavekdc.example.com with kvno 3, encryption type DES with
HMAC/sha1 added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal host/slavekdc.example.com with kvno 3, encryption type DES cbc
mode with RSA-MD5 added to keytab WRFILE:/etc/krb5.keytab.
kadmin: quit
```

b. ***kadmin.local ktadd*** コマンドを使用してサービス用のランダムキーを設定し、ランダムキーをセカンダリー KDC サーバーのデフォルト keytab ファイルに保存します。このキーは、クライアントの認証時に ***kpropd*** サービスによって使用されます。

5. セカンダリー KDC はサービスキーを使用して、接続するクライアントをすべて認証できます。すべてのクライアントが新しいレルムデータベースで ***kprop*** サービスを提供することができるわけではありません。アクセスを制限するために、セカンダリー KDC 上の ***kprop*** サービスは、プリンシパル名が ***/var/kerberos/krb5kdc/kpropd.acl*** に記載されているクライアントからの更新のみを受け入れます。

   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.acl
   ```

6. セカンダリー KDC がデータベースのコピーを取得したら、暗号化に使用したマスターキーも必要になります。KDC データベースのマスターキーがマスター KDC の stash ファイルに保存されている場合（通常は ***/var/kerberos/krb5kdc/.k5.REALM***）、利用可能な安全な方法を使用してセカンダリー KDC にコピーするか、***kdb5_util create を実行して同じパスワードを指定して、セカンダリー KDC にダミーデータベースと同一の*** stash ファイルを作成します。ダミーデータベースは、最初に成功したデータベースの伝播によって上書きされます。

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 (**krb5_prop**)**で TCP を使用して接続し、kprop** サービスを起動します。

8. ***kadmin*** サービスが を **無効** にしていることを確認します。

9. マスター KDC 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、手動でデータベース伝搬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@masterkdc ~]# kprop slavekdc.example.com
    ```

11. ***kinit*** を使用して、クライアントシステムが KDC から正しく初期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 の一覧のセカンダリー KDC のみを一覧表示する必要があります。

    ```
    [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*** コマンドを実行して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、***cron*** サービスがスクリプトを定期的に実行するように設定します。

11.2.3. Kerberos キー配布センタープロキシー

一部の管理者は、デプロイメントでデフォルトの Kerberos ポートにアクセスできないようにすることを選択する場合があります。ユーザー、ホスト、およびサービスが Kerberos 認証情報を取得できるようにするには、**HTTPS** ポート 443 経由で Kerberos と通信するプロキシーとして **HTTPS** サービスを使用できます。

Identity Management (IdM) では、**Kerberos Key Distribution Center Proxy** (KKDCP) がこの機能を提供します。

KKDCP サーバー

IdM サーバーでは、KKDCP がデフォルトで有効になります。属性と値のペア **ipaConfigString=kdcProxyEnabled** がディレクトリーに存在する場合、KKDCP は Apache Web サーバーが起動するたびに自動的に有効になります。この状況では、シンボリックリンク **/etc/httpd/conf.d/ipa-kdc-proxy.conf** が作成されます。したがって、シンボリックリンクが存在することを確認することで、IdM サーバーで KKDCP が有効になっていることを確認できます。

```
$ 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
```

詳細は、以下のサーバー設定の例を参照してください。

例11.1 KKDCP サーバー I の設定

以下の設定例により、複数の Kerberos サーバーが使用される IdM KKDCP と Active Directory レルム間のトランスポートプロトコルとして TCP を使用できるようになります。

1. **/etc/ipa/kdcproxy/kdcproxy.conf** ファイルで、[global] セクションの **use_dns** パラメーターを **false** に設定します。

```
[global]
use_dns = false
```

2. プロキシー化されたレルム情報を **/etc/ipa/kdcproxy/kdcproxy.conf** ファイルに配置します。プロキシーを使用する [AD.**EXAMPLE.COM**] レルムの場合は、たとえば次のようにレルム設定パラメーターを一覧表示します。

```
[AD.EXAMPLE.COM]
kerberos = kerberos+tcp://1.2.3.4:88 kerberos+tcp://5.6.7.8:88
kpasswd = kpasswd+tcp://1.2.3.4:464 kpasswd+tcp://5.6.7.8:464
```

重要

レルム設定パラメーターは、特定のオプションを複数回指定する可能性がある **/etc/krb5.conf および kdc.conf** ではなく、スペースで区切られた複数のサーバーをリストする必要があります。

3. IdM サービスを再起動します。

```
# ipactl restart
```

例11.2 KKDCP サーバー II の設定

このサーバー設定の例は、DNS サービスレコードに依存して、通信する AD サーバーを見つけます。

1. **/etc/ipa/kdcproxy/kdcproxy.conf** ファイルの [global] セクションで、**use_dns** パラメーターを **true** に設定します。

```
[global]
configs = mit
use_dns = true
```

configs パラメーターを使用すると、他の設定モジュールをロードできます。この場合、設定は MIT **libkrb5** ライブラリーから読み込まれます。

2. **オプション：** DNS サービスレコードを使用しない場合は、明示的な AD サーバーを **/etc/krb5.conf** ファイルの [realms] セクションに追加します。たとえば、プロキシーを使用するレルムが AD.EXAMPLE.COM の場合は、以下を追加します。

```
[realms]
AD.EXAMPLE.COM = {
    kdc = ad-server.ad.example.com
    kpasswd_server = ad-server.ad.example.com
}
```

3. IdM サービスを再起動します。

```
# ipactl restart
```

KKDCP クライアント

クライアントシステムは、KDC プロキシーを **/etc/krb5.conf** ファイルを介してポイントします。この手順に従って、AD サーバーに到達します。

1. クライアントで **/etc/krb5.conf** ファイルを開き、AD レルムの名前を [realms] セクションに追加します。

```
[realms]
AD.EXAMPLE.COM {
    kdc = https://ipa-server.example.com/KdcProxy
    kdc = https://ipa-server2.example.com/KdcProxy
    kpasswd_server = https://ipa-server.example.com/KdcProxy
    kpasswd_server = https://ipa-server2.example.com/KdcProxy
}
```

2. **/etc/sssd/sssd.conf** ファイルを開き、krb5_use_kdcinfo = False 行を IdM ドメインセクションに追加します。

```
[domain/example.com]
krb5_use_kdcinfo = False
```

3. **SSSD** サービスを再起動します。

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

関連情報

- Active Directory レルムの 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** 設定ファイルを指定することです。**ssh** および **slogin** は、クライアントシステムにリモートでログインする方法が推奨されますが、Kerberos 対応のバージョンの **rsh** および **rlogin** は引き続き利用でき、追加の設定変更が可能です。

1. すべてのクライアントマシンに **krb5-libs** パッケージおよび **krb5-workstation** パッケージを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** ファイルを指定します。通常、これは Kerberos Distribution Center (KDC) が使用する同じ **krb5.conf** ファイルになります。以下に例を示します。

```
[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
```

一部の環境では、KDC は HTTPS Kerberos Key Distribution Center Proxy (KKDCP) を使用してのみアクセスできます。この場合は、以下の変更を加えます。

   a. ホスト名の代わりに KKDCP の URL を **[realms]** セクションの **kdc** オプションおよび **admin_server** オプションに割り当てます。

```
[realms]
EXAMPLE.COM = {
  kdc = https://kdc.example.com/KdcProxy
  admin_server = https://kdc.example.com/KdcProxy
  kpasswd_server = https://kdc.example.com/KdcProxy
  default_domain = example.com
}
```

*冗長性を確保するために、パラメーター **kdc**、**admin_server**、および **kpasswd_server** は、異なる KKDCP サーバーを使用して複数回追加できます。*

b. *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 対応の **rsh** および **rlogin** サービスを使用するには、**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が Kerberos を使用して、**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です。**sshd**、**kshd**、および **klogind** 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キーへのアクセスが必要になります。*

a. ***kadmin** を使用して、KDC 上のワークステーションのホストプリンシパルを追加します。この場合のインスタンスはワークステーションのホスト名です。**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、ランダムなキーを割り当てます。*

```
addprinc -randkey host/server.example.com
```

b. *ワークステーション **自体で kadmin を実行し、ktadd コマンドを使用すると、鍵をワークステーション** 用に抽出できます。*

```
ktadd -k /etc/krb5.keytab host/server.example.com
```

5. *その他の Kerberos 対応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 対応のサービスは、表11.3「一般的な Kerberos 対応のサービス」 に一覧表示されます。*

表11.3 一般的な Kerberos 対応のサービス

サービス名	使用方法
ssh	クライアントとサーバー両方の設定で **GSSAPIAuthentication** が有効な場合に、OpenSSH は GSS-API を使用してサーバーにユーザーを認証します。クライアントでも **GSSAPIDelegateCredentials** が有効な場合は、ユーザーの認証情報がリモートシステムで利用可能になります。OpenSSH には **sftp** ツールも含まれています。このツールは SFTP サーバーに FTP のようなインターフェイスを提供し、GSS-API を使用できます。

サービス名	使用方法
IMAP	**cyrus-imap** パッケージもインストールされている場合、**cyrus-sasl-gssapi** パッケージは Kerberos 5 を使用します。**cyrus-sasl-gssapi** パッケージには、GSS-API 認証をサポートする Cyrus SASL プラグインが含まれています。Cyrus IMAP 関数は、**cyrus** ユーザーが **/etc/krb5.keytab** で適切な鍵を見つけ、プリンシパルの root が **imap** (**kadmin**で作成した)に設定されている限り Kerberos で適切に機能します。 **cyrus-imap** の代替は、**dovecot** パッケージにあります。これは、Red Hat Enterprise Linux にも含まれています。このパッケージには IMAP サーバーが含まれていますが、現在のところ GSS-API および 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が、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て、公開鍵インフラストラクチャー(PKI)のパラメーターを設定の **[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**)とシステム認証(**/etc/pam.d/system-auth**)の両方の PAM 設定に PKI モジュール情報を追加します。両方のファイルに追加する行は、以下のとおりです。*

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

*OpenSC モジュールが想定どおりに機能しない場合は、coolkey パッケージの モジュール(**/usr/lib64/pkcs11/libcoolkeypk11.so**)を使用します。この場合は、Red Hat テクニカルサポートへの問い合わせ、または問題に関する Bugzilla レポートの作成を検討してください。*

11.5. クロスレルムの KERBEROS 信頼の設定

Kerberos V5 レルムは、接続されたすべてのマスターとスレーブ上の Kerberos データベースに定義されている Kerberos プリンシパルのセットです。異なるレルムからのプリンシパルが相互に通信する場合は、レルム間の Kerberos 信頼を設定する必要があります。

多くの Linux 環境、および混合環境には、シングルサインオン、アプリケーション認証、およびユーザー管理のために Kerberos レルムが既に展開されています。これにより、Kerberos は、特に Linux 環境が Identity Management などのより構造化されたドメイン設定を使用していない場合に、さまざまなドメインおよび混合システム (Windows や Linux など) 環境で共通の統合パスになる可能性があります。

11.5.1. 信頼関係

信頼 *とは、あるレルム内のユーザーが、**そのレルムに属するかのように**、別のドメインのリソースにアクセスするために信頼されることを意味します。これは、両方のドメインで共通に保持される単一のプリンシパルに共有キーを作成して行います。*

図11.2 基本的な信頼

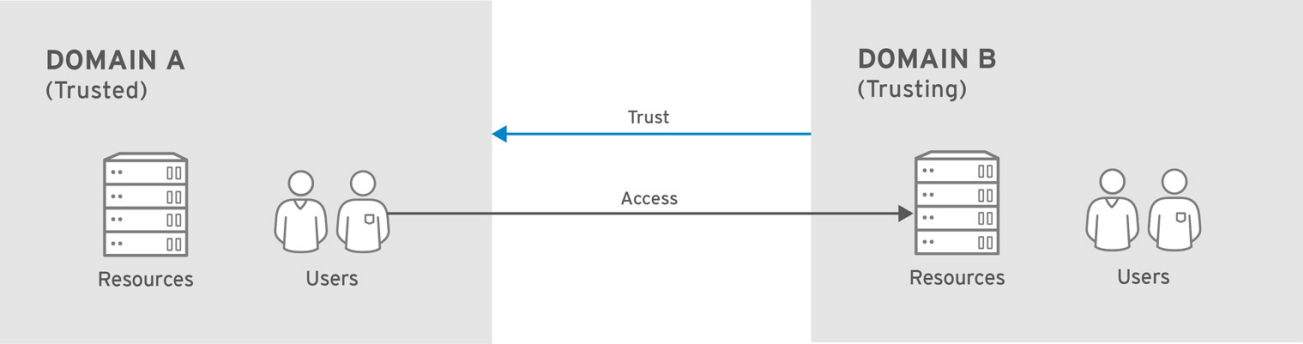


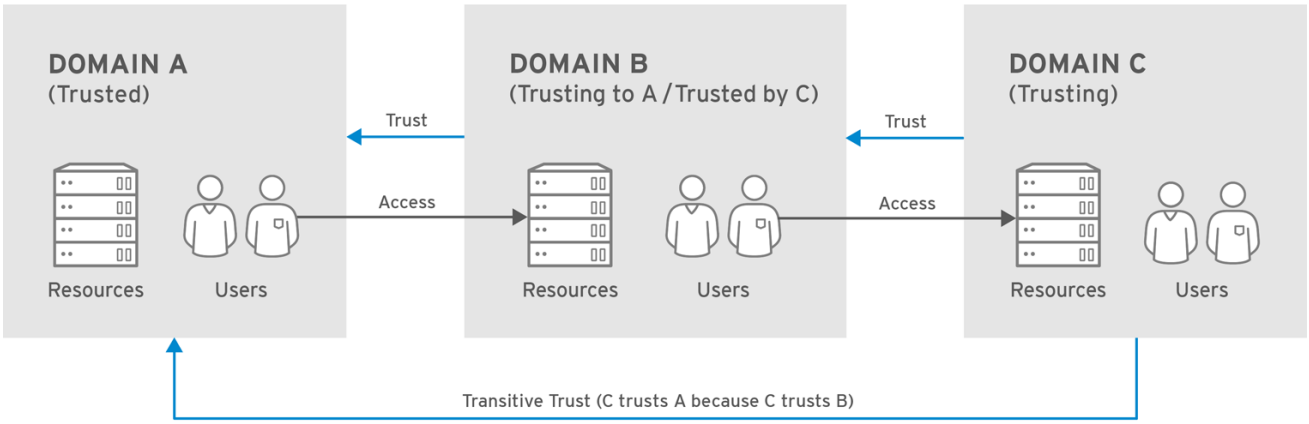
*図11.2「基本的な信頼」 では、共有プリンシパルはドメイン B (**krbtgt/B.EXAMPLE.COM@A.EXAMPLE.COM**) に属します。プリンシパルが Domain A に追加されると、Domain A のクライアントは Domain B のリソースにアクセスできます。設定されたプリンシパルは両方のレルムに存在します。共有プリンシパルには、以下の 3 つの特徴があります。*

- *両方のレルムに存在します。*
- *キーが作成されると、同じパスワードが両方のレルムで使用されます。*
- *キーには、同じキーバージョン番号(**kvno**)があります。*

***レルム間の信頼は、デフォルトで一方向** です。この信頼は自動的に再処理されないため、**B.EXAMPLE.COM** レルムは **A.EXAMPLE.COM** レルムのサービスに対して認証するために信頼されています。反対方向の信頼を確立するには、両方のレルムが **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、前述の例とは反対のマッピングのエントリーになります。*

*レルムは、信頼するレルムと信頼されるレルムの両方で、複数の信頼を持つことができます。Kerberos の信頼を使用すると、信頼は連鎖的に流れることができます。レルム A がレルム B を信頼し、レルム B がレルム C を信頼する場合、レルム A は暗黙的にレルム C を信頼します。信頼フローとレルム。これは **推移的** 信頼です。*

図11.3 推移的な信頼

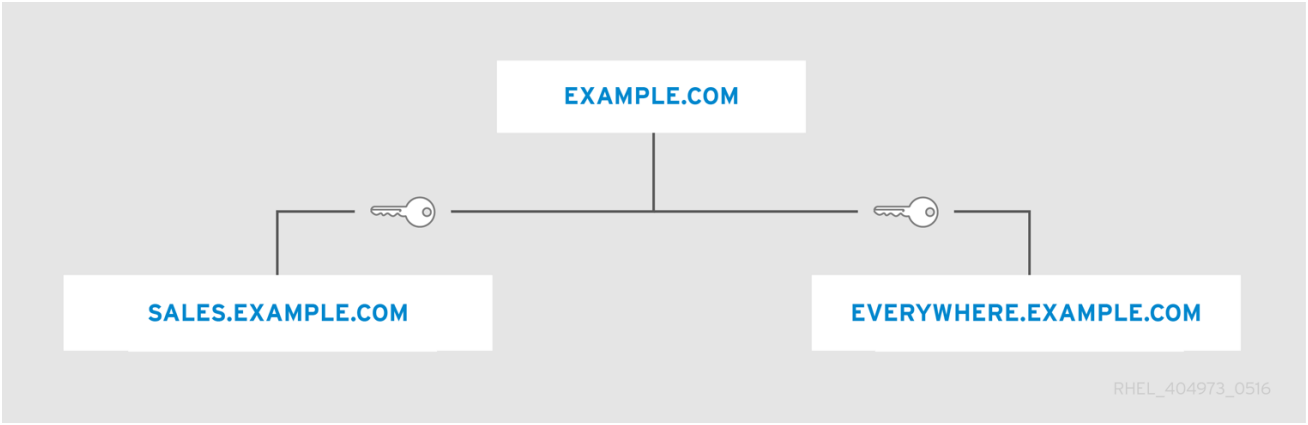


推移的信頼の方向は **信頼フロー** *です。信頼フローは、最初にサービスが属するレルムを認識し、次にクライアントがそのサービスにアクセスするのに接続する必要のあるレルムを識別することで定義する必要があります。*

Kerberos プリンシパル名は **service/hostname@REALM** *の形式で設定されます。***サービス** *は通常 LDAP、IMAP、HTTP などのプロトコル、またはホストです。***hostname** *は、ホストシステムの完全修飾ドメイン名で、***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なります。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、ドメイン名が大文字に変換されます。変換された名前は、検索する Kerberos レルムであると想定されます。*

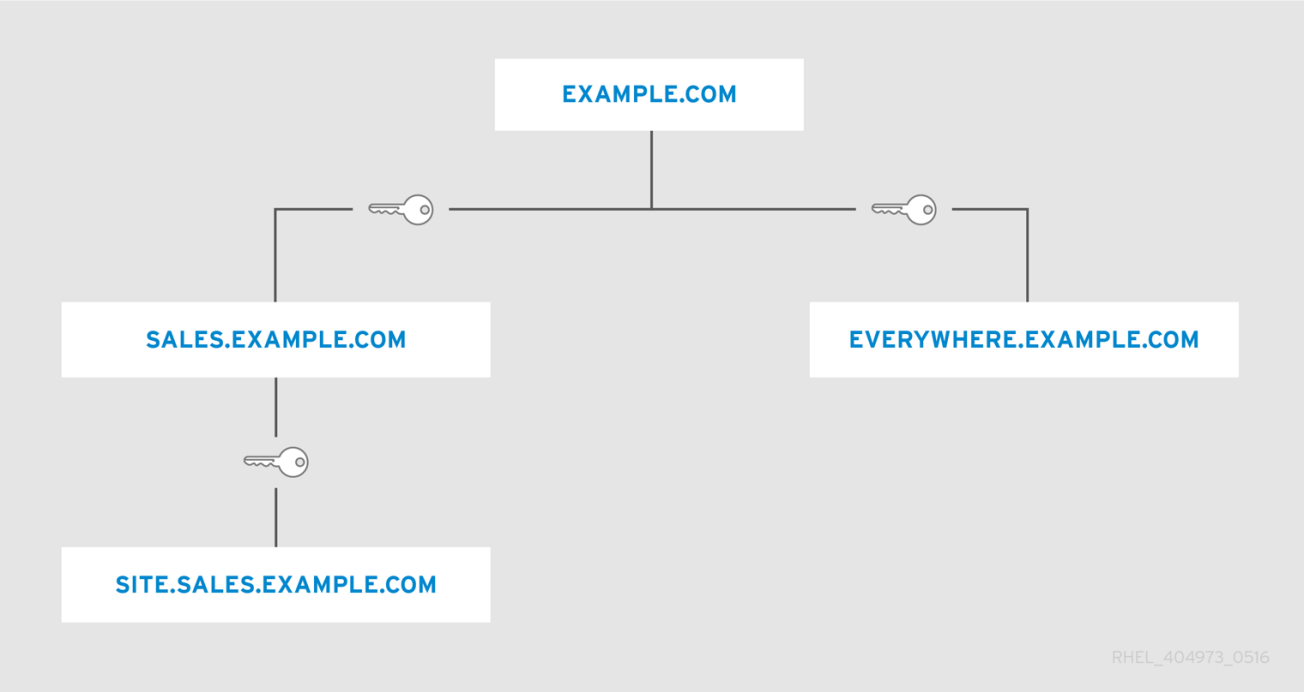
信頼をスキャンする場合、Kerberos は、各レルムが階層 DNS ドメインと同様に設定され、ルートドメインおよびサブドメインを持つことが前提となります。これは、信頼のフローが共有ルートに至ることを意味します。各ステップまたは **ホップ** *には、共有キーがあります。図11.4「同じドメインの信頼」 では、SALES.EXAMPLE.COM が EXAMPLE.COM の鍵を共有し、EXAMPLE.COM が EVERYWHERE.EXAMPLE.COM と鍵を共有します。*

図11.4 同じドメインの信頼



クライアントはレルム名を DNS 名として扱い、ルート名に到達するまで自身のレルム名の要素を取り除くことによって信頼パスを決定します。その後、サービスのレルムに到達するまで、名前を先頭に追加します。

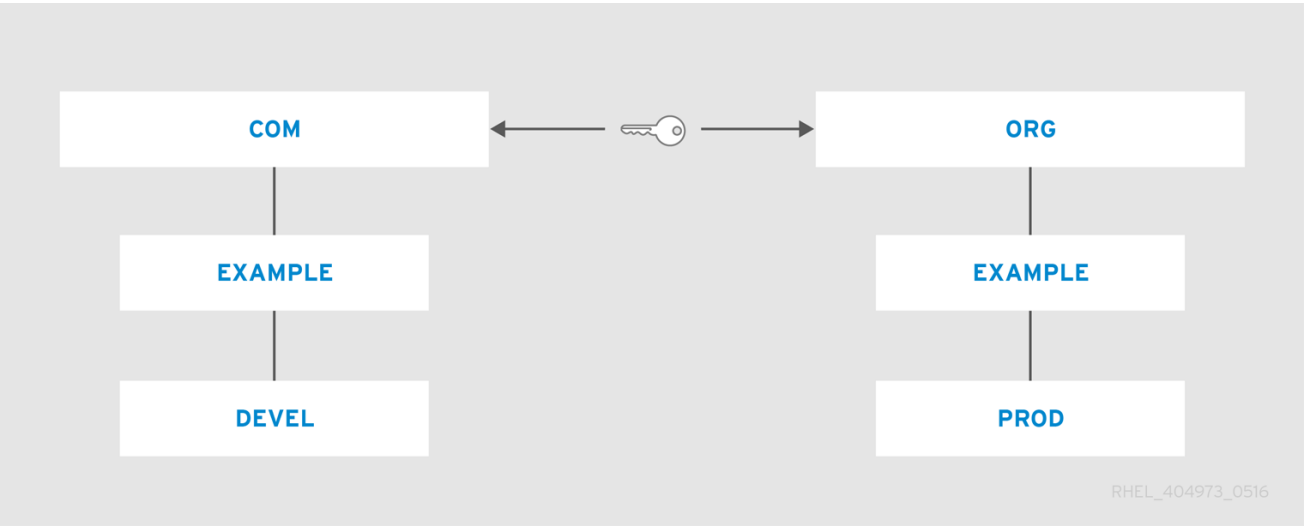
図11.5 同じドメインの子/親の信頼



これは、推移的である信頼の性質です。SITE.SALES.EXAMPLE.COM には、SALES.EXAMPLE.COM との共有キーが1 つだけあります。しかし、一連の小さな信頼のために、信頼が SITE.SALES.EXAMPLE.COM から EVERYWHERE.EXAMPLE.COM に移動できるようにする大きな信頼フローがあります。

この信頼フローは、ドメインレベルで共有鍵を作成し、サイトが共通の接尾辞を持たないことで、完全に異なるドメイン間で送信される可能性があります。

図11.6 異なるドメインの信頼



[capaths] セクション

*フローを明示的に定義することにより、ホップ数を減らし、非常に複雑な信頼フローを表すこともできます。**/etc/krb5.conf** ファイルの **[capaths]** セクションは、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簡単です。クライアントがプリンシパルを持つ各レルムにメインエントリーがあり、各レルムセクションにはクライアントが認証情報を取得する中間レルムのリストになります。*

たとえば、**[capaths]** を使用して認証情報を取得するために以下のプロセスを指定できます。

1. レルム A からのクレデンシャルでは、クライアントは Realm 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** チケットを要求します。

   Realm A の KDC が発行する **krbtgt/B@A** チケットは、レルム間のチケット付与チケット です。クライアントは、Realm B の KDC に対してチケットを Realm B のサービスプリンシパルに要求できます。

2. **krbtgt/B@A** チケットを使用すると、クライアントは **krbtgt/C@B** クロスレルムチケットを要求します。

3. Realm B の KDC が発行する **krbtgt/C@B** チケットを使用すると、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. Realm C の KDC が発行する **krbtgt/D@C** チケットを使用すると、クライアントは Realm 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。

その後、認証情報キャッシュには **krbtgt/A@A**、**krbtgt/B@A**、**krbtgt/C@B**、**krbtgt/D@C**、および **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、3 つの中間レルムチケットを取得する必要があります。

[capaths] 設定の例を含む **[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。

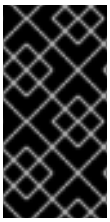
11.5.2. レルム信頼の設定

この例では、Kerberos レルムは **A.EXAMPLE.COM** および **B.EXAMPLE.COM** です。

kadmin を使用して、A レルムの B レルムの共有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
```

つまり、A レルムが B プリンシパルを信頼することを意味します。



重要

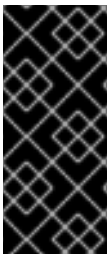
レルム間のプリンシパルに非常に強固なパスワードを選択することが推奨されます。ユーザーに 1 日に数回プロンプトが表示される他の多くのパスワードとは異なり、システムはレルム間プリンシパルのパスワードを頻繁に要求しないため、簡単に覚える必要はありません。

双方向の信頼を作成するには、逆方向のプリンシパルを作成します。**kadmin** を使用して、B レルムに A レルムのプリンシパルを作成します。

```
[root@server ~]# kadmin -r B.EXAMPLE.COM
kadmin: add_principal krbtgt/A.EXAMPLE.COM@B.EXAMPLE.COM
Enter password for principal "krbtgt/A.EXAMPLE.COM@B.EXAMPLE.COM":
```

```
Re-enter password for principal "krbtgt/A.EXAMPLE.COM@B.EXAMPLE.COM":
Principal "krbtgt/A.EXAMPLE.COM@B.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)と暗号化タイプの両方が一致することを確認します。



重要

一般的な間違った状況は、管理者がパスワードの代わりに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しいエントリーをダンプして2 番目のレルムにインポートするために **add_principal** コマンドの **-randkey** オプションを使用しようとすることです。データベースダンプに含まれる鍵自体がマスターキーを使用して暗号化されるため、レルムデータベースのマスターキーが同一でない限り動作しません。

---

[3] クライアントとサーバーの両方が共通の鍵を共有しているシステム。これは、ネットワーク通信の暗号化と復号化に使用されます。

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のライフサイクルを管理し、適切に設定されている場合は、認証局(CA)と連携して証明書を更新できます。

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、有効期限が近づいている証明書を更新できます。**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** です。

注記

certmonger デーモンは証明書を取り消すことができません。証明書は、関連する認証局のみが取り消すことができます。この認証局は、証明書を無効にし、証明書失効リストを更新する必要があります。

12.1. CERTMONGER および認証局

デフォルトでは、**certmonger** は、証明書が使用する認証局ソースで異なる 3 種類の証明書を自動的に取得できます。

- 自己署名証明書

  各証明書は証明書独自のキーを使用して署名されるため、自己署名証明書の生成には CA は全く関与しません。自己署名証明書を検証するソフトウェアは、証明書を検証するために、その証明書を直接信頼するように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 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。

- システムに存在するローカル CA により署名された証明書

  ローカル署名者が署名した証明書を検証するソフトウェアは、証明書を検証するためにこのローカル署名から証明書を信頼するように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 CA は **certmonger** を使用して証明書を管理することもできますが、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 ヘルパーの作成方法は、**certmonger** プロジェクトのドキュメント（） https://pagure.io/certmonger/blob/master/f/doc/submit.txt を参照してください。

12.2. CERTMONGER を使用した自己署名証明書の要求

certmonger で証明書を要求するには、**getcert request** ユーティリティーを使用します。

証明書とキーは、証明書のニックネームで識別される **.pem** 拡張子または NSS データベースを持つプレーンテキストファイルにローカルに保存されます。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。以下に例を示します。

```
[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 NSS データベースで、**Server-Cert** はこの証明書のニックネームです。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。*

証明書を生成するコマンドで提供できるオプションは、要求している証明書の種類、最終的な証明書に必要な設定、およびその他の設定によって異なります。

- ***-r** は、キーペアがすでに存在する場合に失効日が近づくと証明書を自動的に更新します。このオプションはデフォルトで使用されます。*
- ***-f** は、指定したファイルに証明書を保存します。*
- ***-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 プリンシパル名を指定します。**-K** は、IdM サーバーから証明書を要求する際に必要であり、自己署名の証明書またはローカルで署名された証明書を要求する場合は任意になります。*
- ***-N** はサブジェクト名を指定します。*
- ***-d** は、DNS ドメイン名を **subjectAltName** 値として証明書に含めるよう要求します。*
- ***-u** は、拡張キー使用フラグを設定します。*
- ***-A** は、**subjectAltName** 値として証明書に含まれる IP アドレスを要求します。*
- ***-I** はタスクの名前を設定します。**certmonger** はこの名前を使用して、ストレージの場所と要求オプションの組み合わせを参照し、**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** はタスクに自動生成された名前を割り当てます。*

*IdM のような実際の CA は、CA 独自のポリシーに従って、**-K**、**-N**、**-D**、**-U**、および **-A**オプションを使用して署名要求で指定したものをすべて無視できます。たとえば、IdM では、**-K** と **-N** がローカルホスト名に合意する必要があります。**selfsign-getcert** コマンドおよび **local-getcert** コマンドを使用して生成される証明書は、ポリシーを強制しないため、指定するオプションに同意します。*

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 経由での CA 署名証明書の要求

Simple Certificate Enrollment Protocol (SCEP) は、CA を使用した証明書管理プロセスを自動化し、簡素化します。クライアント要求や、CA の SCEP サービスから直接 HTTP 経由で証明書を取得できるようになります。このプロセスは、通常、限られた時間でのみ有効な 1 回限りの PIN で保護されます。

*以下の例では、SCEP CA 設定を **certmonger** に追加し、新しい証明書を要求し、ローカルの NSS データベースに追加します。*

1. *CA 設定を **certmonger** に追加します。*

```
[root@server ~]# getcert add-scep-ca -c CA_Name -u SCEP_URL
```

-

- ***-c***: CA 設定に必要なニックネーム。後で同じ値を他の **getcert** コマンドに渡すことができます。
- ***-u***: サーバーの SCEP インターフェイスへの URL。
- HTTPS URL を使用する場合は必須のパラメーターです。

  -R CA_Filename: HTTPS 暗号化に使用される SCEP サーバーの CA 証明書の PEM 形式のコピーの場所。

2. CA 設定が正常に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 で取得され、コマンドの出力に表示される際に、CA 設定が正常に追加されました。暗号化されていない HTTP でサーバーにアクセスすると、中間者攻撃を防ぐため、サムプリントを SCEP サーバーに表示されるものと手動で比較します。

3. CA から証明書を要求します。

```
[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。後で同じ値を **getcert list** コマンドに渡すことができます。
- ***-c***: 要求を送信する CA 設定。
- ***-d***: 証明書とキーを格納するための NSS データベースを備えたディレクトリー。
- ***-n***: NSS データベースで使用される証明書のニックネーム。
- ***-n***: CSR のサブジェクト名。
- ***-l***: CA が発行する Time-time PIN。

4. 要求の送信直後に、証明書が発行され、ローカルデータベースに正しく保存されていることを確認します。

```
[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
```

*ステータス **MONITORING** は、発行した証明書の取得に成功したことを表します。**getcert-list (1)** の man ページには、他の可能な状態とその意味が一覧表示されます。*

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。鍵と証明書を NSS データベースに保存するには、証明書の要求に使用するコマンドで**-d** および **-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 データベースの証明書に使用される証明書のニックネームを指定します。*

注記

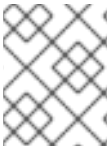
***.pem** ファイルを提供する **-f** オプションおよび **-k** オプションの代わりに、**-d** オプションおよび **-n** オプションが使用されます。*

以下に例を示します。

```
[root@server ~]# selfsign-getcert request -d /export/alias -n ServerCert ...
```

***ipa-getcert** および **local-getcert** を使用して証明書を要求すると、別の 2 つのオプションを指定できます。*

- ***-F** は、CA の証明書を保存するファイルを指定します。*
- ***-a** は、CA の証明書が保存される NSS データベースの場所を指定します。*



注記

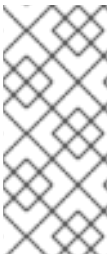
***selfsign-getcert** を使用して証明書を要求する場合、自己署名証明書の生成には CA が含まれないので、**-F** オプションおよび **-a** オプションを指定する必要はありません。*

***-F** オプション、**-a** オプション、または **local-getcert** の両方を指定すると、ローカル署名者が発行した証明書を検証するために必要な CA 証明書のコピーを取得できます。以下に例を示します。*

```
[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、有効期間の終了時に証明書を自動的に更新できます。この方法で証明書を追跡するには、**getcert start-tracking** コマンドを実行します。



注記

getcert request コマンドはデフォルトで、要求された証明書を自動的に追跡して更新するため、**getcert start-tracking** を実行した後、**getcert** start-tracking を実行する必要はありません。**getcert start-tracking** コマンドは、他のプロセスを介して鍵と証明書をすでに取得している状況を対象としています。したがって、追跡を開始するように **certmonger** に手動で指示する必要があります。

getcert start-tracking コマンドは、複数のオプションを取ります。

- **-r** は、キーペアがすでに存在する場合に失効日が近づくと証明書を自動的に更新します。このオプションはデフォルトで使用されます。
- **-I** は、追跡要求の名前を設定します。**certmonger** はこの名前を使用して、ストレージの場所と要求オプションの組み合わせを参照し、**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** はタスクの名前を自動的に生成します。

```
[root@server ~]# getcert start-tracking -I cert1-tracker -d /export/alias -n ServerCert
```

証明書の追跡を取り消すには、**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

ブラウザーや電子メールクライアントなどの一般的なアプリケーションには、ユーザーを認証する手段としてKerberos チケット、SSL 証明書、またはトークンを使用するように設定できます。

アプリケーションを設定する正確な手順は、アプリケーション自体により異なります。本章の例では (Mozilla Thunderbird およびMozilla Firefox) は、Kerberos またはその他の認証情報を使用するようにユーザーアプリケーションを設定する方法を考えています。

13.1. FIREFOX でシングルサインオンに KERBEROS を使用する設定

Firefox では、Kerberos を使用して、イントラネットサイトや他の保護されている Web サイトへのシングルサインオン (SSO) を使用できます。Firefox でKerberos を使用するには、まずKerberos 認証情報を適切なKDC に送信するように設定する必要があります。

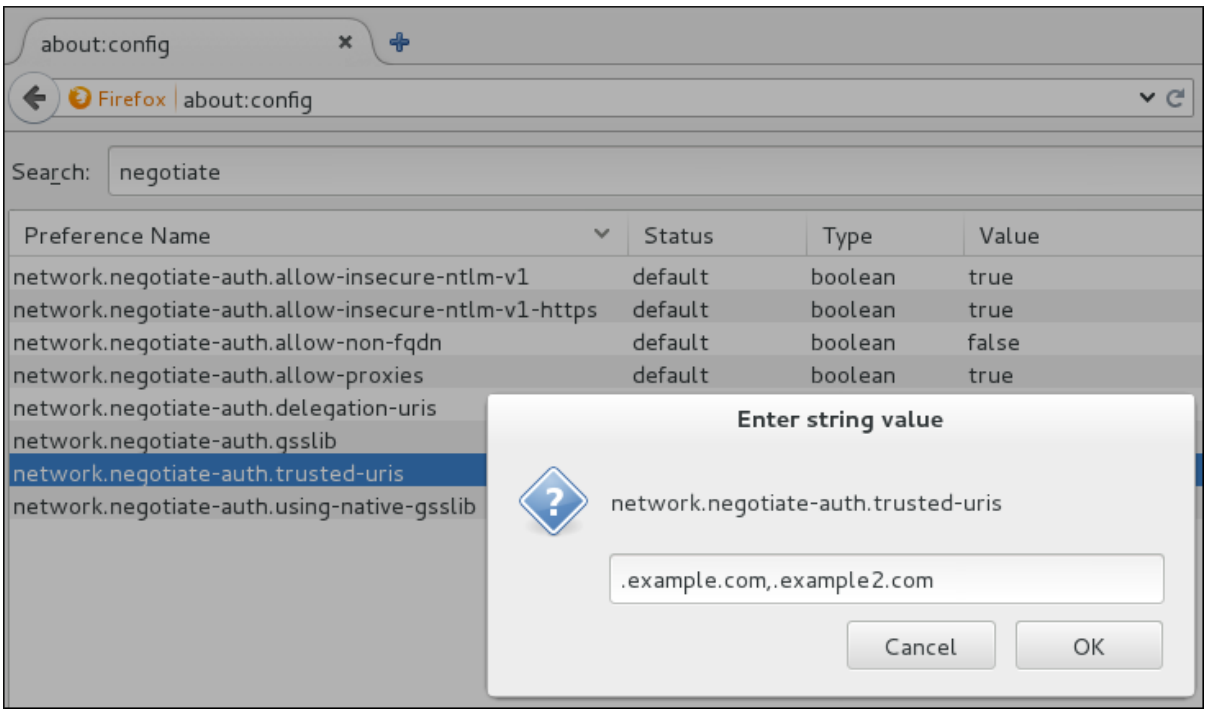
*Firefox がKerberos 認証情報を渡すように設定した後でも、有効なKerberos チケットが必要になります。Kerberos チケットを生成するには、**kinit** コマンドを使用してKDC 上のユーザーのユーザーパスワードを指定します。*

```
[jsmith@host ~] $ kinit
Password for jsmith@EXAMPLE.COM:
```

Firefox がSSO にKerberos を使用するように設定するには、以下を実行します。

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の一覧を表示します。*
2. ***Filter** フィールドに、オプションのリストを制限するために **negotiate** と入力します。*
3. ***network.negotiate-auth.trusted-uris** エントリーをダブルクリックします。*
4. *先行ピリオド (.) を含む、認証に使用する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を入力します。*

図13.1 Firefox の手動設定



重要

*Firefox 設定オプションで **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お勧めしません。これは、すべての Kerberos 対応サーバーがユーザーとして機能できるためです。*

注記

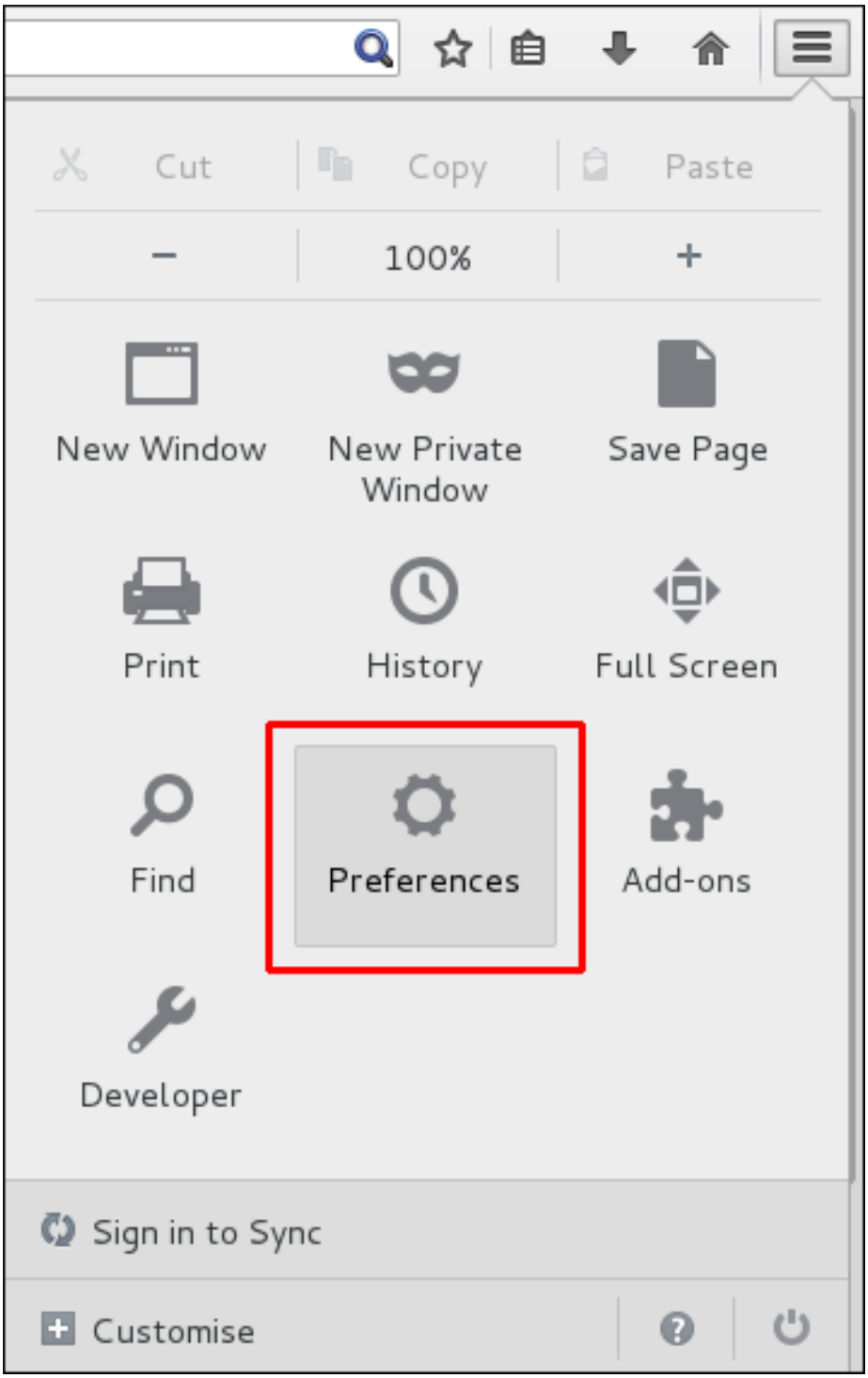
詳細は、『Linux Domain Identity, Authentication, and Policy Guide』の Configuring the Browser for Kerberos Authentication を参照してください。

13.2. FIREFOX での証明書管理

*Firefox で証明書を管理するには、**Certificate Manager** を開きます。*

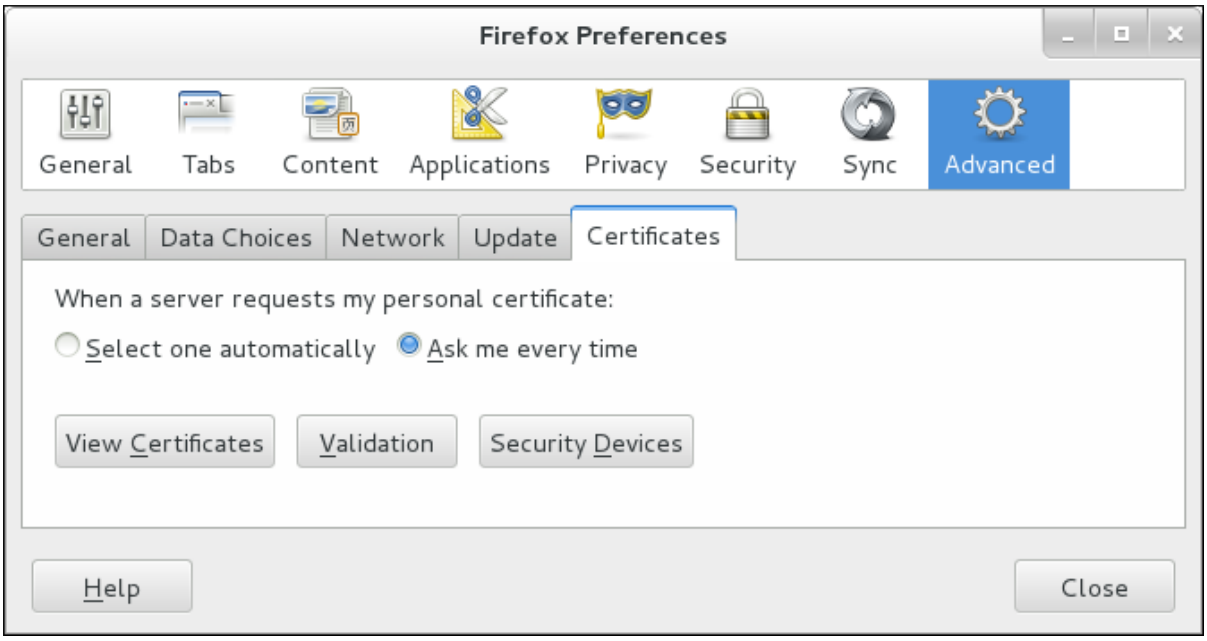
1. *Mozilla Firefox で Firefox メニューを開き、**Preferences** をクリックします。*

図13.2 Firefox の設定



2. ***Advanced*** *セクションを開き、* ***Certificates*** *タブを選択します。*

図13.3 Firefox の証明書タブ

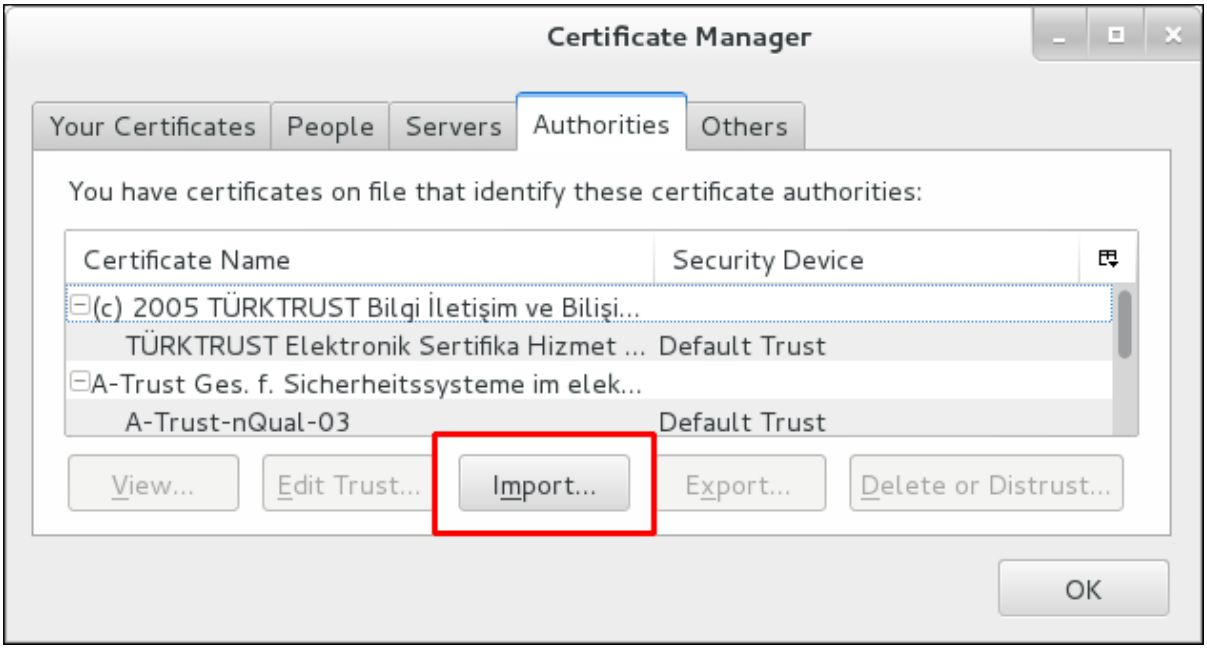


3. ***View Certificates*** *をクリックして、* ***Certificate Manager*** *を開きます。*

CA 証明書をインポートするには、以下を実行します。

1. *CA 証明書をダウンロードし、コンピューターに保存します。*
2. ***Certificate Manager*** *で* ***Authorities*** *タブを選択し、* ***Import*** *をクリックします。*

図13.4 Firefox での CA 証明書のインポート

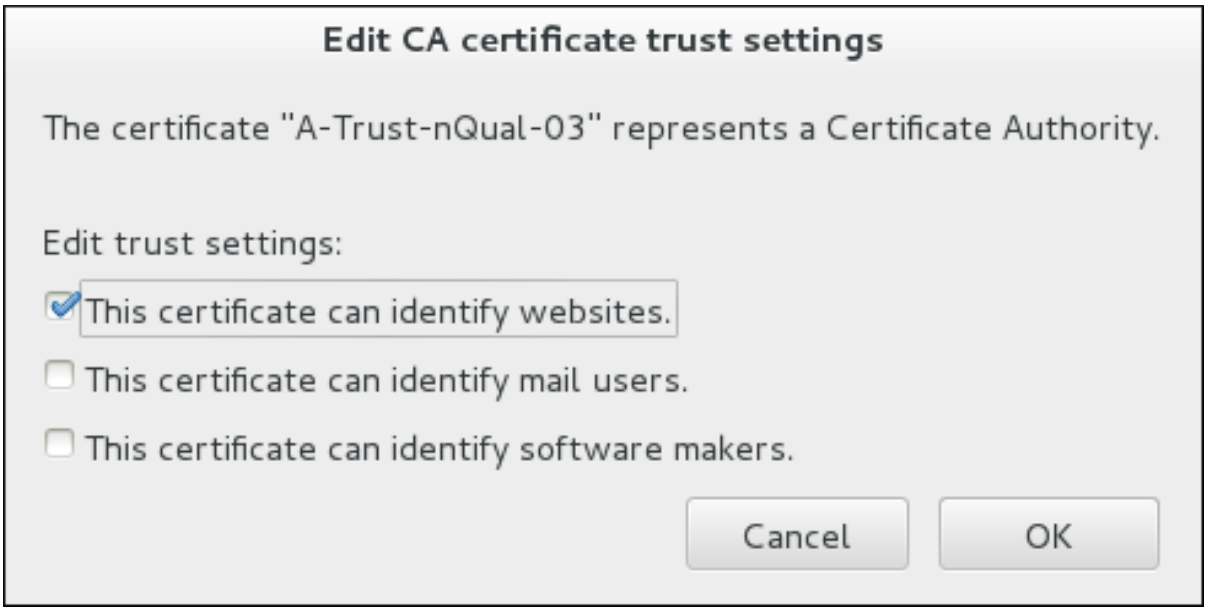


3. *ダウンロードした CA 証明書を選択します。*

証明書信頼関係を設定するには、以下を行います。

1. ***Certificate Manager*** *の* ***Authorities*** *タブで、適切な証明書を選択し、* ***Edit Trust*** *をクリックします。*
2. *証明書トラスト設定を編集します。*

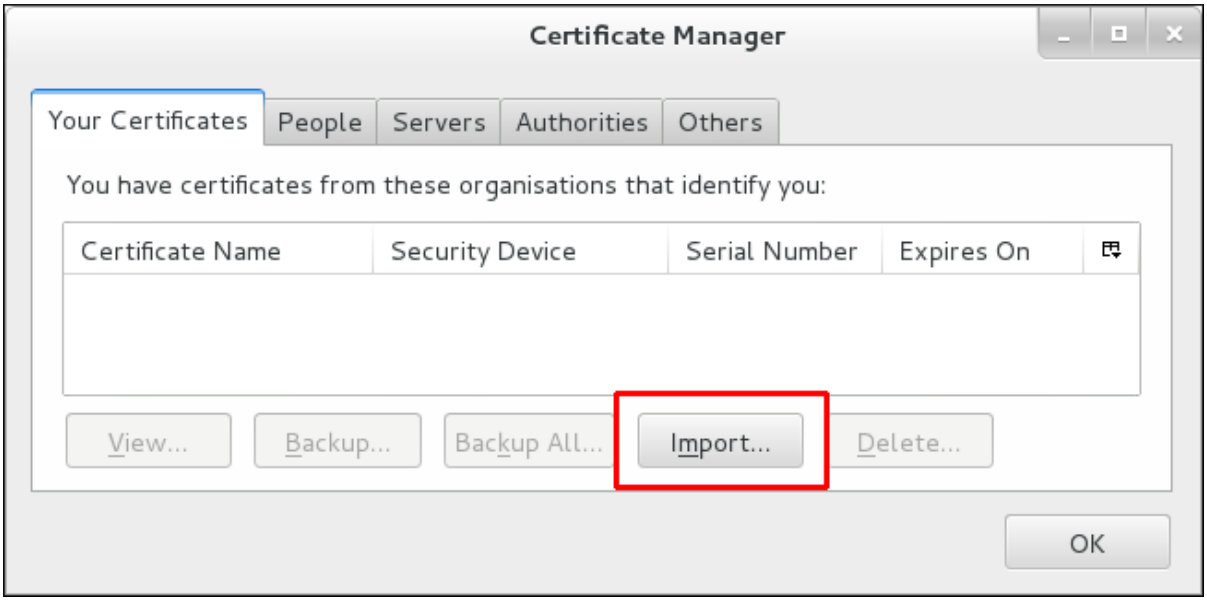
図13.5 Firefox での証明書トラスト設定の編集



認証に個人証明書を使用するには、以下を実行します。

1. ***Certificate Manager*** *の* ***Your Certificates*** *タブで、* ***Import*** *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ーから必要な証明書を選択します。*

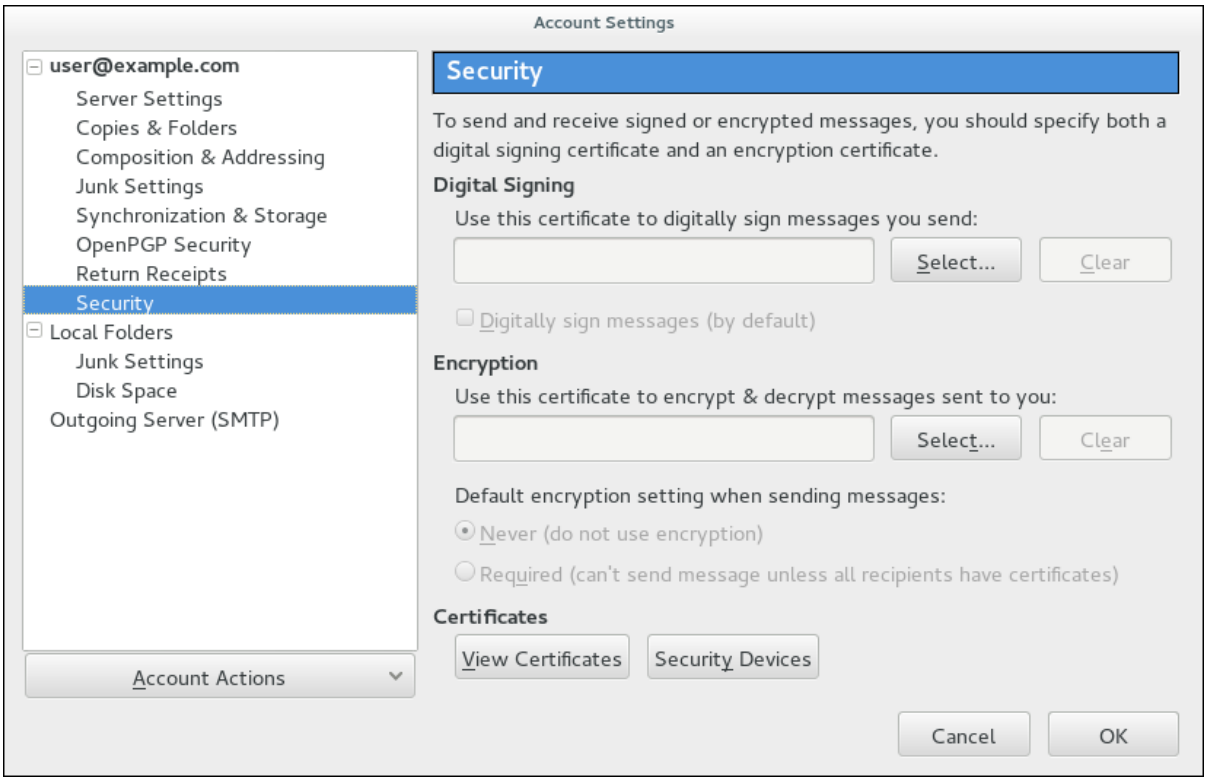
13.3. メールクライアントの証明書管理

以下の例は、Mozilla Thunderbird 電子メールクライアントで証明書を管理する方法を示しています。これは、一般的に電子メールクライアントで証明書を設定する手順を説明します。

1. *Mozilla Thunderbird で、Thunderbird のメインメニューを開き、* ***Preferences → Account Settings*** *の順に選択します。*

2. ***Security*** *項目を選択し、* ***View Certificates*** *をクリックして* ***Certificate Manager*** *を開きます。*

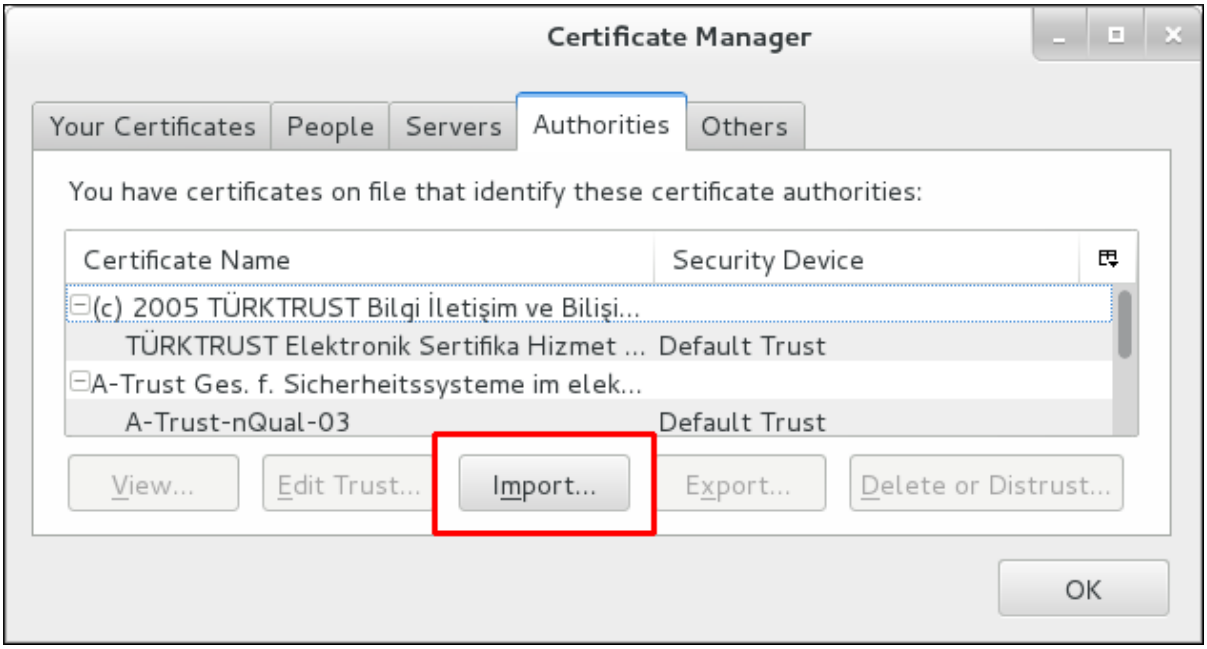
図13.7 Thunderbird のアカウント設定



CA 証明書をインポートするには、以下を実行します。

1. *CA 証明書をダウンロードし、コンピューターに保存します。*
2. ***Certificate Manager** で **Authorities** タブを選択し、**Import** をクリックします。*

   図13.8 Thunderbird での CA 証明書のインポート



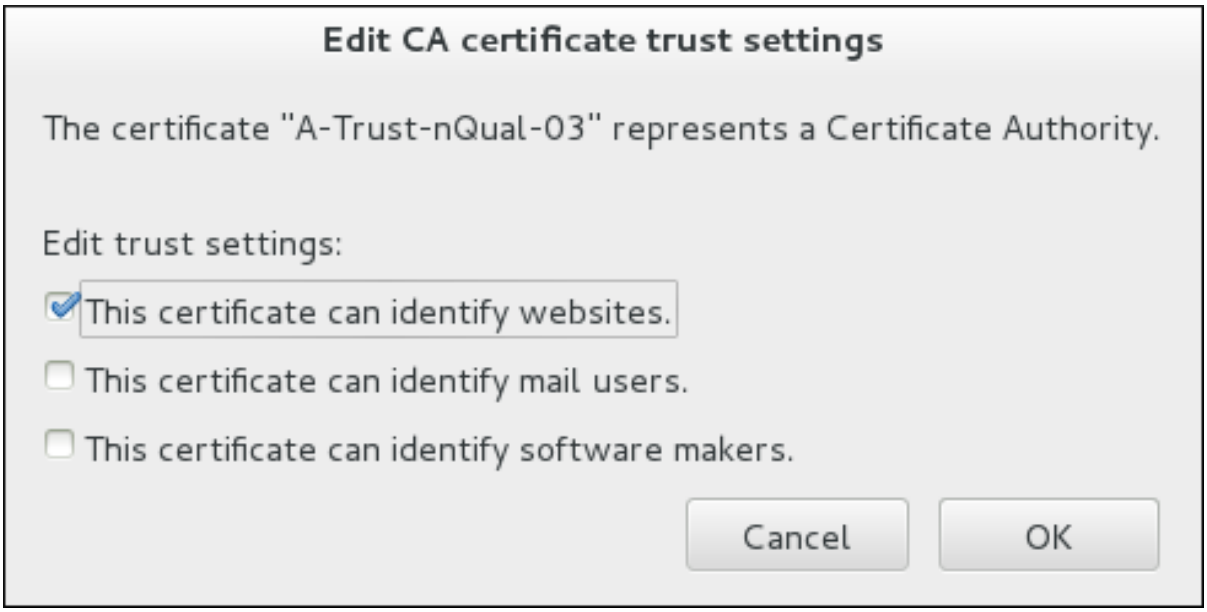
3. *ダウンロードした CA 証明書を選択します。*

証明書信頼関係を設定するには、以下を行います。

1. ***Certificate Manager** の **Authorities** タブで、適切な証明書を選択し、**Edit Trust** をクリックします。*

2. *証明書トラスト設定を編集します。*

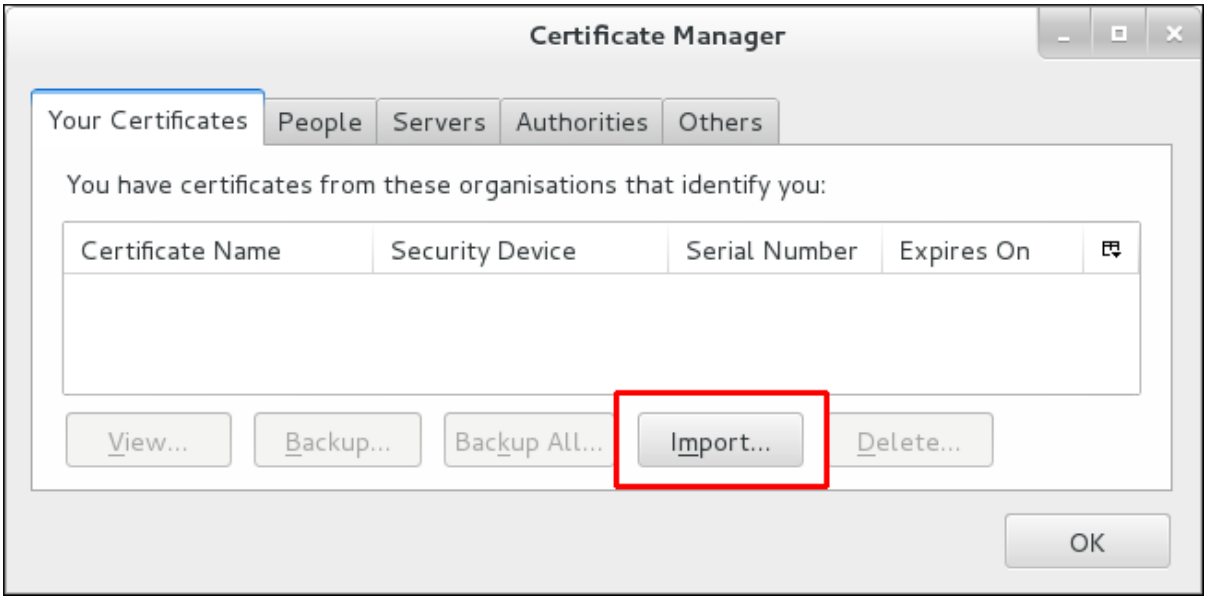
図13.9 Thunderbird での証明書トラスト設定の編集



認証に個人証明書を使用するには、以下を実行します。

1. ***Certificate Manager*** *の* ***Your Certificates*** *タブで、* ***Import*** *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ーから必要な証明書を選択します。*

3. ***Certificate Manager*** *を閉じ、* ***Account Settings*** *の* ***Security*** *項目に戻ります。*

4. *フォームの* ***Digital Signing*** *セクションで、Select 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. ***Encryption*** *で Select をクリックして、メッセージを暗号化および復号する個人証明書を* ***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 ドメインのデバッグログの設定」
- 「SSSD ログファイルの確認」
- 「SSSD 設定に関する問題」

A.1.1. SSSD ドメインのデバッグログの設定

各ドメインは、独自のデバッグログレベルを設定します。ログレベルを増やすと、SSSD またはドメイン設定の問題に関する詳細情報が提供されます。

*ログレベルを変更するには、**sssd.conf** ファイルの各セクションに **debug_level** パラメーターを設定して、追加のログを生成します。以下に例を示します。*

```
[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	致命的な障害。SSSD の起動を妨げる、または SSSD の実行を停止させるもの。
1	重大なエラー。SSSD を強制終了しないものの、少なくとも 1 つの主要な機能が機能しなくなったことを示すエラー。
2	深刻なエラー。特定の要求または操作が失敗したことを示すエラー。
3	マイナーな障害。これらのエラーが浸透して、2 の動作不良の原因となるのです。
4	設定設定。
5	関数データ。
6	操作関数のメッセージを追跡します。
7	内部制御関数のメッセージのトレース。
8	対象の関数内部変数の内容。
9	非常に低いレベルのトレース情報。

SSSD の実行中にデバッグレベルを変更するには、**sssd-tools** パッケージに含まれる **sss_debuglevel** ユーティリティーを使用します。仕組みの詳細は、sss_debuglevel の man ページを参照してください。

A.1.2. SSSD ログファイルの確認

SSSD は、多くのログファイルを使用して、**/var/log/sssd/** ディレクトリーにある操作に関する情報を報告します。SSSD は、各ドメインのログファイルと **sssd_pam.log** および **sssd_nss.log** ファイルを生成します。

- ***krb5_child.log***: Kerberos 認証に関連する有効期限の短いヘルパープロセスのログファイル
- ***ldap_child.log***: LDAP サーバーと通信に関連する有効期限の短いヘルパープロセスのログファイル
- ***selinux_child.log***: SELinux 情報を取得する有効期限の短いヘルパープロセスのログファイル
- ***sssd.log***: レスポンダープロセスと通信する SSSD のログファイル
- ***sssd_[domain].log***: 各 SSSD ドメインセクションは、LDAP サーバーとの通信に関する情報を別のログファイルに記録します。
- ***sssd_ifp.log***: InfoPipe レスポンダーのログファイル。システムバスを介してアクセス可能なパブリック D-Bus インターフェイスを提供します。
- ***sssd_nss.log***: ユーザーおよびグループ情報を取得する Name Services Switch (NSS) レスポンダーのログファイル。
- ***sssd_pac.log***: Active Directory ユーザーおよびグループの管理に SSSD が Kerberos と動作する方法を定義する Microsoft Privilege Attribute Certificate (PAC) レスポンダーサービスのログファイルです。
- ***sssd_pam.log***: Pluggable Authentication Module (PAM) レスポンダーのログファイル
- ***sssd_ssh.log***: SSH レスポンダープロセスのログファイルです。

さらに、**/var/log/secure** ファイルは認証の失敗と失敗の原因をログに記録します。

A.1.3. SSSD 設定に関する問題

問： SSSD が起動に失敗する

答： SSSD では、デーモンを起動する前に、必要なすべてのエントリーで設定ファイルを適切に設定する必要があります。

SSSD では、サービスが起動する前に、最低でもドメインを適切に設定する必要があります。ドメインがないと、SSSD を起動すると、ドメインが設定されていない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/sssd/sssd.conf ファイルを編集し、ドメインを 1 つ以上作成します。

SSSD は、開始する前に、少なくとも 1 つ以上の利用可能なサービスプロバイダーも必要です。問題がサービスプロバイダー設定にある場合、エラーメッセージはサービスが設定されていないことを示します。

```
[sssd] [get_monitor_config] (0): No services configured!
```

***/etc/sssd/sssd.conf** ファイルを編集し、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、サービスプロバイダーを **/etc/sssd/sssd.conf** ファイルの単一の **services** エントリーでコンマ区切りのリストとして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一覧表示されます。最後のエントリーのみが SSSD によって認識されます。*

*SSSD では、**/etc/sssd/sssd.conf** の所有権とパーミッションを正しく設定する必要があります。所有者またはパーミッションが誤って設定されている場合は、SSSD を開始しようとすると、以下のエラーメッセージが返されます。*

```
[sssd] [confdb_ldif_from_ini_file] (0x0020): Permission check on config file failed.
[sssd] [confdb_init_db] (0x0020): Cannot convert INI to LDIF [1]: [Operation not
permitted]
[sssd] [confdb_setup] (0x0010): ConfDB initialization has failed [1]: Operation not
permitted
[sssd] [load_configuration] (0x0010): Unable to setup ConfDB [1]: Operation not
permitted
[sssd] [main] (0x0020): Cannot read config file /etc/sssd/sssd.conf. Please check that
the file is accessible only by the owner and owned by root.root.
```

***/etc/sssd/sssd.conf** ファイルの正しい所有権と権限を設定します。*

```
# chmod 600 /etc/sssd/sssd.conf
# chown root:root /etc/sssd/sssd.conf
```

問： getent group の id またはグループメンバーを持つグループは表示されません。

***答：** これは、**sssd.conf** の **[domain/DOMAINNAME]** セクションに誤った **ldap_schema** 設定が原因である可能性があります。*

SSSD は RFC 2307 および 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 の相違点は、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です。RFC 2307 サーバーでは、グループメンバーはメンバーであるユーザーの名前が含まれる多値 **memberuid** 属性として保存されます。RFC2307bis サーバーでは、グループメンバーは、このグループの **メンバー** であるユーザーまたはグループの DN を含む多値 member または **uniqueMember** 属性として保存されます。RFC2307bis を使用すると、ネストされたグループも保守できます。*

グループルックアップが情報が返されない場合は、以下を行います。

1. ***ldap_schema** を **rfc2307bis** に設定します。*

2. *Delete **/var/lib/sss/db/cache_DOMAINNAME.ldb**.*

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 **sssd.conf** に追加します。*

```
ldap_group_name = uniqueMember
```

次に、キャッシュを削除し、再度 SSSD を再起動します。

問： 認証は LDAP に対して失敗します。

答： *認証を実行するには、SSSD で通信チャネルを暗号化する必要があります。これは、**sssd.conf** が標準プロトコル(**ldap://**)に接続するように設定されていると、Start TLS で通信チャネルの暗号化を試みます。**sssd.conf** が安全なプロトコル(**ldaps://**)に接続するように設定されている場合、SSSD は SSL を使用します。*

つまり、LDAP サーバーは SSL または TLS で実行する必要があります。標準の LDAP ポート (389) で TLS を有効にするか、セキュア LDAPS ポート (636) で SSL を有効にする必要があります。SSL または TLS のいずれかを使用する場合、LDAP サーバーも有効な証明書の信頼で設定する必要があります。

無効な証明書の信頼は、LDAP に対する認証に関する最も一般的な問題の 1 つです。クライアントが LDAP サーバー証明書を適切に信頼していない場合、接続を検証できず、SSSD はパスワードの送信を拒否します。LDAP プロトコルでは、パスワードをプレーンテキストで LDAP サーバーに送信する必要があります。暗号化されていない接続でプレーンテキストのパスワードを送信することは、セキュリティーの問題です。

*証明書が信頼されていない場合は、**syslog** メッセージが書き込まれ、TLS 暗号化を開始できなかったことを示します。証明書設定は、SSSD とは別に LDAP サーバーにアクセスできるかどうかを確認してテストできます。たとえば、以下は、**test.example.com** への TLS 接続で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、テストは以下のエラーを出して失敗します。

```
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、次のコマンドを実行します。

1. *LDAP サーバー証明書に署名するために使用される認証局の公開 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. *ファイルシステムの CA 証明書を参照する **sssd.conf** ファイルに行を追加します。*

   ```
   ldap_tls_cacert = /path/to/cacert
   ```

3. *LDAP サーバーが自己署名証明書を使用する場合は、**sssd.conf** ファイルから **ldap_tls_reqcert** 行を削除します。*

   このパラメーターにより、SSSD が CA 証明書により発行された証明書を信頼するように指示します。これは、自己署名の CA 証明書を使用するセキュリティーリスクになります。

問： 非標準ポートで LDAP サーバーへの接続に失敗します。

答： SELinux を Enforcing モードで実行する場合は、クライアントの SELinux ポリシーを変更して、標準以外のポートで LDAP サーバーに接続する必要があります。以下に例を示します。

```
# semanage port -a -t ldap_port_t -p tcp 1389
```

問： NSS がユーザー情報を返すことができません

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 サービスが実行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/sssd.service; enabled)
  Active: active (running) since Wed 2015-01-14 10:17:26 CET; 1min 30s ago
  Process: 683 ExecStart=/usr/sbin/sssd -D -f (code=exited, status=0/SUCCESS)
Main PID: 745 (sssd)
  CGroup: /system.slice/sssd.service
          ├─745 /usr/sbin/sssd -D -f
    ├─746 /usr/libexec/sssd/sssd_be --domain default --debug-to-files...
    ├─804 /usr/libexec/sssd/sssd_nss --debug-to-files
    └─805 /usr/libexec/sssd/sssd_pam --debug-to-files
```

SSSD が **Active: active (running)** 状態にあり、出力に **sssd_nss** が含まれると、NSS サービスが実行されている。

NSS を実行している場合は、プロバイダーが **/etc/sssd/sssd.conf** ファイルの **[nss]** セクションで適切に設定されていることを確認します。特に、**filter_users** 属性および **filter_groups** 属性を確認します。

NSS が SSSD が使用するサービスの一覧に含まれていることを確認します。

/etc/nsswitch.conf ファイルの設定を確認します。詳細は、「SSSD を使用するように NSS サービスを設定する」 を参照してください。

問： NSS が間違ったユーザー情報を返す

答： 検索が正しくないユーザー情報を返した場合は、別のドメインでユーザー名が競合していないことを確認してください。複数のドメインがある場合は、**/etc/sssd/sssd.conf** ファイルで **use_fully_qualified_domains** 属性を **true** に設定します。これは、同じ名前の異なるドメインの異なるユーザーを区別します。

問： ローカルの SSSD ユーザーのパスワードを設定すると、パスワードが 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を変更しようとすると、パスワードを 2 回求められる場合があります。

```
[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.
```

これは、PAM 設定が間違っているためです。**/etc/pam.d/system-auth** ファイルで **use_authtok** オプションが正しく設定されていることを確認します。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 ID プロバイダーは sssd.conf ファイルで適切に設定されていますが、SSSD は接続に失敗し、GSS-API エラーが発生します。

答： SSSD は、ホスト名を使用して Active Directory プロバイダーのみに接続できます。ホスト名が指定されていない場合、SSSD クライアントはホストの IP アドレスを解決できず、認証に失敗します。

たとえば、以下の設定を使用します。

```
[domain/ADEXAMPLE]
debug_level = 0xFFF0
id_provider = ad
ad_server = 172.16.0.1
ad_domain = example.com
krb5_canonicalize = False
```

SSSD クライアントはこの GSS-API 失敗を返し、認証要求に失敗します。

```
(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)]
```

このエラーを回避するには、**ad_server** を Active Directory ホストの名前に設定するか、「DNS サービスディスカバリーの設定」 で説明されているように **_srv_** キーワードを使用して DNS サービス検出を使用します。

問： SSSD が中央認証用に設定されましたが、アプリケーション (Firefox、Adobe など) が複数起動しません。

答： 64 ビットシステムでは、32 ビットアプリケーションには、パスワードおよび ID キャッシュにアクセスするのに使用する 32 ビットのバージョンの SSSD クライアントライブラリーが必要です。32 ビットバージョンの SSSD が利用できない場合は、システムが SSSD キャッシュを使用するように設定されている場合、32 ビットアプリケーションは起動に失敗する可能性があります。

たとえば、Firefox はパーミッション拒否エラーで失敗できます。

```
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 の場合は、現在のシステムユーザーが認識されていないことを示すエラーが表示されます。

```
[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は同様のユーザーまたはパーミッションエラーが表示される可能性があります。

問： SSSD は、削除された自動マウントの場所を表示します。

答： 自動マウントの場所の SSSD キャッシュは、場所が後で変更または削除しても持続します。SSSD で autofs 情報を更新するには、次のコマンドを実行します。

1. 「UID または GID の変更後にユーザーがログインできません。」 の説明に従って、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 の変更後にユーザーがログインできません。

ユーザーまたはグループ ID を変更すると、SSSD はユーザーがログインしないようにします。

エラー内容:
SSSD は、ユーザー ID(UID) とグループ ID(GID) に基づいてユーザーおよびグループを認識します。既存のユーザーまたはグループの UID または GID が変更されると、SSSD はユーザーまたはグループを認識しません。

解決方法:
sss_cache ユーティリティーを使用して 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。
2. 特定ユーザーの SSSD キャッシュをクリアし、その他のキャッシュレコードをそのまま残すには、次のコマンドを実行します。

```
# sss_cache --user user_name
```

ドメイン全体のキャッシュを消去するには、以下のコマンドを実行します。

```
# sss_cache --domain domain_name
```

このユーティリティーは、ユーザー、グループ、またはドメインの SSSD キャッシュ内のレコードを無効にします。その後、SSSD はアイデンティティープロバイダーからレコードを取得してキャッシュを更新します。

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 コントロールおよび Status ユーティリティー

SSSD は、**sssctl** ユーティリティーを提供して、ステータス情報の取得、トラブルシューティング中にデータファイル、およびその他の SSSD 関連のタスクを管理します。

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** の一部のオプションは、SSSD InfoPipe レスポンダーを使用します。これを有効にするには、**/etc/sssd/sssd.conf** の **services** オプションに **ifp** を追加します。

```
[sssd]
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 設定検証

sssctl config-check コマンドは、SSSD 設定ファイルの静的分析を実行します。これにより、**/etc/sssd/sssd.conf** ファイルおよび/ **etc/sssd/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受け取ることができます。

このコマンドは、SSSD 設定ファイルで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、パーミッションを **600** に設定する必要があります。

ファイル名

/etc/sssd/conf.d/ のファイル名は、接尾辞 **.conf** を使用し、ピリオド（非表示ファイル）で開始しないでください。

誤植

セクションとオプション名で誤字のエラーがチェックされます。値はチェックされません。

オプション

オプションは正しいセクションに配置する必要があります。

設定を確認するには、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/sssd/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/sssd/conf.d/sample.conf
```

A.1.5.2. ユーザーデータの表示

sssctl user-checks コマンドは、SSSD をユーザールックアップ、認証、および認可のバックエンドとして使用するアプリケーションの問題をデバッグするのに役立ちます。このコマンドは、NSS 経由で利用可能なユーザーデータと、D-Bus インターフェイスの InfoPipe レスポンダーを表示します。表示されるデータは、ユーザーが **system-auth** PAM 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。

たとえば、**admin** ユーザーのユーザーデータを表示するには、次のコマンドを実行します。

```
# sssctl user-checks admin
user: admin
action: acct
service: system-auth

SSSD nss user lookup result:
 - user name: admin
 - user id: 1194200000
 - group id: 1194200000
 - gecos: Administrator
 - home directory: /home/admin
 - shell: /bin/bash

SSSD InfoPipe user lookup result:
 - name: admin
 - uidNumber: 1194200000
 - gidNumber: 1194200000
 - gecos: Administrator
 - homeDirectory: /home/admin
 - loginShell: /bin/bash

testing pam_acct_mgmt

pam_acct_mgmt: Success

PAM Environment:
 - no env -
```

その他のオプションは、**sssctl user-checks --help** コマンドの出力を参照してください。

A.1.5.3. ドメイン情報

ドメインステータスには、SSSD アクセスでドメインアクセスの一覧が表示され、そのステータスを取得できるようになります。

1. 信頼されたドメインを含む、SSSD 内で利用可能なドメインの一覧を表示します。

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

2. ドメイン idm.example.com のステータスを取得します。

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

A.1.5.4. キャッシュされたエントリー情報

sssctl コマンドを使用すると、キャッシュされたエントリーに関する情報を取得して、キャッシュ関連の認証問題を調査およびデバッグできます。

ユーザーアカウント **admin** のキャッシュ情報をクエリーするには、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 netgroup のキャッシュ情報をクエリーするには、以下を使用します。

```
[root@server ~]# sssctl group-show groupname
[root@server ~]# sssctl netgroup-show netgroupname
```

A.1.5.5. ログファイルの切り捨て

問題をデバッグする場合、**sssctl logs-remove** を使用して、**/var/log/sssd/** ディレクトリー内のすべての SSSD ログファイルを切り捨て、新たに作成されたエントリーのみをキャプチャーできます。

```
[root@server ~]# sssctl logs-remove
Truncating log files...
```

A.1.5.6. SSSD キャッシュの削除

SSSD キャッシュデータベースファイルを削除するには、**sssctl** コマンドは **remove-cache** オプションを提供します。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。

以下のコマンドを使用して、ローカルデータをすべてバックアップし、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

バックアップは、ローカルオーバーライドなどのローカルデータのみを **/var/lib/sss/backup/** ディレクトリーに保存します。

バックアップされた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.1.5.7. LDAP グループに関する情報の取得には時間がかかる

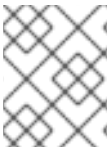
LDAP グループに関する情報を検索する操作には、特に大規模なグループまたはネストされたグループの場合は非常に長い時間がかかります。

エラー内容:
デフォルトでは、LDAP グループ情報の検索は、グループのすべてのメンバーを返します。大規模なグループまたはネスト化されたグループが含まれる操作の場合は、すべてのメンバーを返すとプロセスが長くなります。

解決方法:
ユーザーがグループに属するかどうかを評価する場合は、グループ検索で返されるメンバーシップリストは使用されません。特に ID ルックアップのパフォーマンスを改善するには、グループメンバーシップルックアップを無効にします。

1. **/etc/sssd/sssd.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
```



注記

デプロイメントで compat ツリーを持つ IdM サーバーがある場合は、このオプションを **true** に設定しないでください。

A.2. SSSD および SUDO DEBUGGING LOGS を使用した SUDO のトラブルシューティング

A.2.1. SSSD および sudo デバッグロギング

デバッグロギング機能により、SSSD および sudo に関する追加情報を記録できます。

sudo デバッグログファイル
sudo デバッグを有効にするには、以下の手順に従います。

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と環境についてどのような情報が利用できますか？

```
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=498,6004,6005,7001,106501,554800513,554801107,554801108,554801393,5548015
03,554802131,554802244,554807670
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 ルールの取得に使用されるデータソース。

```
sudo[22259] <- sudo_parseln @ ./fileops.c:178 := sudoers: files sss
```

- SSSD プラグインは以下の行で始まります。

```
sudo[22259] <- sudo_sss_open @ ./sssd.c:305 := 0
```

- SSSD が返すルールの数

```
sudo[22259] Received 3 rule(s)
```

- ルールが一致しているかどうか。

```
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 デバッグを有効にするには、以下を実行します。

1. */etc/sssd/sssd.conf* ファイルの **[sudo]** セクションおよび **[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、デバッグ情報をログファイルに書き込みます。

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/sssd/sssd_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のに役立ちます。

- SSSD が返すルールの数

```
[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。

```
[sssd[be[LDAP.PB]]] [sysdb_save_sudorule] (0x0400): Adding sudo rule demo-name
```

- マッチング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ために使用されたのはどのフィルターですか。

```
[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]
```

このフィルターを使用して、IdM データベース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 レスポンダーログファイル：/var/log/sssd/sssd_sudo.log

このログファイルは、以下のような質問に回答するのに役立ちます。

- SSSD が返すルールの数

```
[sssd[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。

```
[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るルールを検索するにはどうすればいいですか。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 ユーティリティーは、ldb-tools パッケージに含まれています。

A.3. FIREFOX 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** ファイルで、メッセージの **nsNegotiateAuth** でエラーメッセージを確認します。

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
```

これは、Kerberos チケットが利用できないことを意味します (つまり、有効期限が切れたか、生成されていません)。これを修正するには、**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。

サーバーが Kerberos データベースで見つかりません

```
-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 と通信できないことを意味します。通常、これは Kerberos 設定の問題です。正しいエントリーは、ドメインを識別するために /**etc/krb5.conf** ファイルの **[domain_realm]** セクションになければなりません。以下に例を示します。*

```
.example.com = EXAMPLE.COM
example.com = EXAMPLE.COM
```

ログにエラーはありません

HTTP プロキシーサーバーが 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 を使用してサイトへの接続を試みます。これにより、要求を変更せずに渡すことができます。

付録B 更新履歴

改訂番号はこのマニュアルの編集に関するものであり、Red Hat Enterprise Linux のバージョン番号とは関係ありません。

改訂 7.0-31 Wed Nov 11 2020 Florian Delehaye
7.9 GA 公開用のマイナーな修正で更新

改訂 7.0-30 Tue Aug 06 2019 Marc Muehlfeld
7.7 GA 公開用ドキュメントバージョン

改訂 7.0-29 Tue Apr 08 2019 Marc Muehlfeld
『SSSD のファイルプロバイダーの設定』と『ユーザーデータの表示』を追加若干の更新

改訂 7.0-28 Fri Oct 26 2018 Filip Hanzelka
7.6 GA 公開用ドキュメントの準備。

改訂 7.0-27 Mon Jun 25 2018 Filip Hanzelka
『certmonger の操作』 を更新。

改訂 7.0-26 Tue Apr 10 2018 Filip Hanzelka
7.5 GA 公開用ドキュメントの準備。

改訂 7.0-25 Mon Mar 19 2018 Lucie Maňásková
マイナー更新。

改訂 7.0-24 Mon Feb 12 2018 Aneta Šteflová Petrová
若干の更新

改訂 7.0-23 Mon Jan 29 2018 Aneta Šteflová Petrová
若干の修正。

改訂 7.0-22 Mon Dec 4 2017 Aneta Šteflová Petrová
『スマートカード』 を更新。

改訂 7.0-21 Mon Nov 20 2017 Aneta Šteflová Petrová
若干の修正。

改訂 7.0-20 Mon Nov 6 2017 Aneta Šteflová Petrová
若干の修正。

改訂 7.0-19 Mon Aug 14 2017 Aneta Šteflová Petrová
coolkey パッケージを参照する更新されたセクションを更新。

改訂 7.0-18 Tue Jul 18 2017 Aneta Šteflová Petrová
7.4 GA 公開用ドキュメントバージョン

改訂 7.0-17 Mon Mar 27 2017 Aneta Šteflová Petrová
無効になっていたリンクを修正。

改訂 7.0-16 Mon Feb 27 2017 Aneta Šteflová Petrová
Kerberos KDC プロキシーを更新しました。その他の若干の更新。

改訂 7.0-15 Wed Dec 7 2016 Aneta Šteflová Petrová
SSSD クライアント側のビューを追加その他の若干の更新。

改訂 7.0-14 Tue Oct 18 2016 Aneta Šteflová Petrová

7.3 GA リリースのバージョン

改訂 7.0-13 Wed Jul 27 2016 Marc Muehlfeld
SCEP を介して証明書を要求する Kerberos over HTTP(kdcproxy) を追加。その他の小規模更新。

改訂 7.0-11 Thu Mar 03 2016 Aneta Petrová
PAM サービスのドメイン制限を追加。

改訂 7.0-10 Tue Feb 09 2016 Aneta Petrová
authconfig の章を小さな章に分割。その他の小規模更新。

改訂 7.0-9 Thu Nov 12 2015 Aneta Petrová
7.2 GA リリース向けのバージョン。

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 Wed Feb 25 2015 Tomáš Čapek
7.1 GA リリース向けバージョン。

改訂 7.0-4 Fri Dec 05 2014 Tomáš Čapek
スプラッシュページでの分類順序を更新して再構築。

改訂 7.0-1 July 16, 2014 Ella Deon Ballard
RHEL 7.0 の初期ドラフト