Red Hat Enterprise Linux 9

IdM ユーザー、グループ、ホスト、およびアクセス制御ルールの管理

ユーザーとホストの設定、グループでの管理、およびホストベースおよびロールベースのアクセス制御ルールによるアクセスの制御

Last Updated: 2024-06-26

Red Hat Enterprise Linux 9 IdM ユーザー、グループ、ホスト、およびアクセス制御ルールの管理

ユーザーとホストの設定、グループでの管理、およびホストベースおよびロールベースのアクセス制御ルールによるアクセスの制御

法律上の通知

概要

Red Hat Identity Management (IdM) の主な機能は、ユーザー、グループ、ホスト、およびホストベースのアクセス制御 (HBAC) やロールベースのアクセス制御 (RBAC) などのアクセス制御ルールの管理です。これらは、コマンドライン、IdM Web UI、および Ansible Playbook を使用して設定できます。 管理タスクには、Kerberos ポリシーとセキュリティーの設定、グループメンバーシップの自動化、権限の委譲などがあります。

目次

RED HAT ドキュメントへのフィードバック (英語のみ)

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

Jira からのフィードバック送信 (アカウントが必要)

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

第1章 IDM コマンドラインユーティリティーの概要

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

前提条件

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

1.1. IPA コマンドラインインターフェイスとは

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

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

IPA CLI では以下を行うことができます。

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

1.2. IPA のヘルプとは

IPA ヘルプは、IdM サーバー用の組み込みドキュメントシステムです。

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

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

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

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

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

- **[]** - 括弧は、すべてのパラメーターが任意であることを示しており、**ipa help** のみを入力すれば、コマンドが実行できます。

- | - パイプ文字は **または** の意味になります。したがって、基本的な **ipa help** コマンドを使用して、**TOPIC**、**COMMAND**、**topics** または **commands** を指定できます。
  - **topics** – コマンド **ipa help topics** を実行して、IPA ヘルプでカバーされている **user**、**cert**、**server** などのトピックのリストを表示できます。
  - **TOPIC** – 大文字の TOPIC は変数になります。したがって、特定のトピック (**ipa help user** など) を指定できます。
  - **commands** – コマンド **ipa help commands** を入力して、**user-add**、**ca-enable**、**server-show** などの IPA ヘルプでカバーされているコマンドのリストを表示できます。
  - **COMMAND** – 大文字の COMMAND は変数になります。したがって、**ipa help user-add** などの特定のコマンドを指定できます。

1.3. IPA ヘルプトピックの使用

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

手順

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

```
$ ipa help topics
```

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

```
$ ipa help user
```

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

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

スクロールダウンすれば、ヘルプ全体を表示できます

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

1.4. IPA HELP コマンドの使用

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

手順

1. 端末を開き、IdM サーバーに接続します。
2. ヘルプで使用できるコマンドのリストを表示するには、**ipa help commands** コマンドを実行します。

```
$ ipa help commands
```

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

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

関連情報

- **ipa** の man ページ

1.5. IPA コマンドの構造

IPA CLI は、以下のタイプのコマンドを区別します。

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

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

- ユーザー
- ホスト
- DNS レコード
- 証明書

その他にも多数あります。

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

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

コマンドの構造は次のとおりです。

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

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

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

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

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

1.6. IPA コマンドを使用した IDM へのユーザーアカウントの追加

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

前提条件

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

手順

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

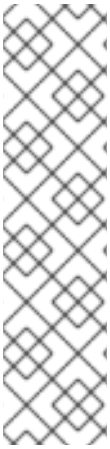
   ```
   $ ipa user-add
   ```

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

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

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

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



注記

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

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

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

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

関連情報

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

1.7. IPA コマンドで IDM のユーザーアカウントの変更

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

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

前提条件

- ユーザーアカウントを変更するには、管理者権限が必要です。

手順

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

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

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

IPA CLI は次の出力を出力します。

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

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

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

関連情報

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

1.8. IDM ユーティリティーに値をリスト形式で提供する方法

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

IdM は、多値リストを提供する次の方法に対応します。

- 同じコマンド呼び出しで、同じコマンドライン引数を複数回指定します。

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

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

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

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

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

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

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

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

または

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

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

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

または

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

1.9. IDM ユーティリティーで特殊文字を使用する方法

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

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

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

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

第2章 コマンドラインでユーザーアカウントの管理

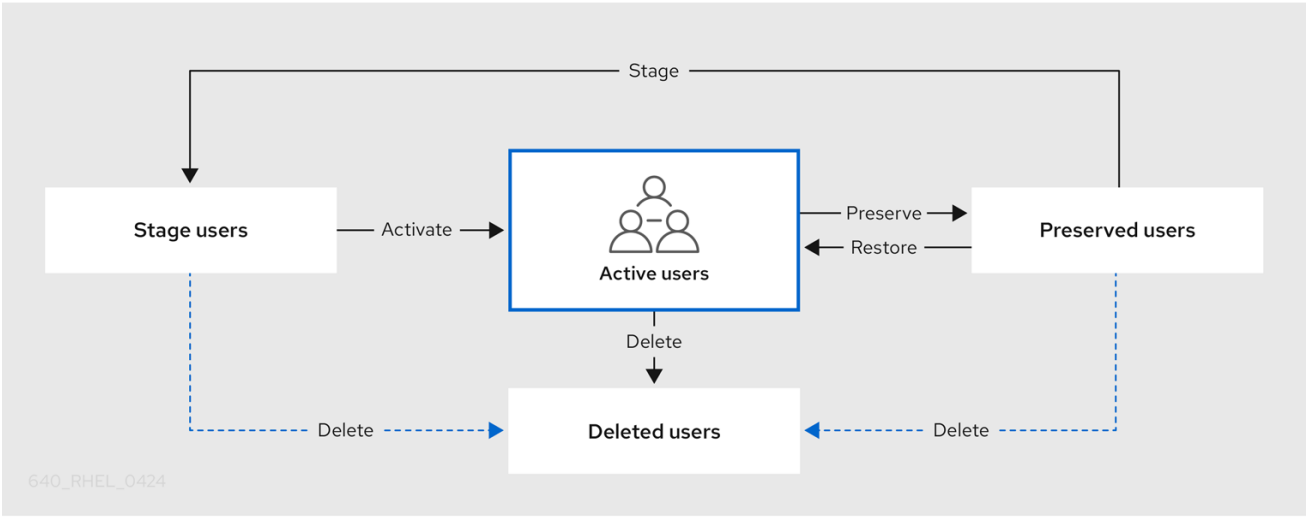
IdM (Identity Management) のユーザーライフサイクルには、次のようないくつかの段階があります。

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

2.1. ユーザーのライフサイクル

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

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



IdM データベースからユーザーエントリーを完全に削除できます。

重要

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

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

警告

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

警告

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

2.2. コマンドラインを使用したユーザーの追加

次のユーザーを追加できます。

- **アクティブ** - ユーザーがアクティブに使用できるユーザーアカウント
- **ステージ** - ユーザーは、このアカウントを使用できません。新しいユーザーアカウントを準備する場合は、ステージユーザーを作成します。ユーザーがアカウントを使用する準備ができると、アクティベートできます。

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

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

警告

IdM は、新しいユーザーアカウントに一意のユーザー ID (UID) を自動的に割り当てます。**ipa user-add** コマンドで **--uid=INT** オプションを使用すると、UID を手動で割り当てることができます。ただし、UID 番号が一意であるかどうかは、サーバーによって検証されません。そのため、複数のユーザーエントリーに同じ UID 番号が割り当てられる可能性があります。**--gidnumber=INT** オプションを使用してユーザーアカウントに GID を手動で割り当てる場合も、ユーザーのプライベートグループ ID (GID) で同様の問題が発生する可能性があります。同じ ID を持つユーザーエントリーが複数あるかどうかを確認するには、**ipa user-find --uid=<uid>** または **ipa user-find --gidnumber=<gidnumber>** と入力します。

Red Hat は、複数のエントリーに同じ UID または GID を割り当てることがないようにすることを推奨します。ID が重複するオブジェクトがある、セキュリティー識別子 (SID) が正しく生成されません。SID は、IdM と Active Directory 間の信頼と、Kerberos 認証の正常な動作に不可欠です。

前提条件

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

手順

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

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

IdM は、以下の正規表現で説明できるユーザー名をサポートします。

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

注記

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

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

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

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

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

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

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

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

```
$ ipa user-find
```

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

関連情報

- PAC 情報による Kerberos セキュリティーの強化
- Are user/group collisions supported in Red Hat Enterprise Linux?
- SID のないユーザーは、アップグレード後に IdM にログインできない

2.3. コマンドラインでユーザーのアクティベート

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

前提条件

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

手順

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

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

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

```
$ ipa user-find
```

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

2.4. コマンドラインでユーザーの保存

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

前提条件

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

手順

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

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

注記

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

2.5. コマンドラインを使用したユーザーの削除

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

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

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

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

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

前提条件

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

手順

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

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

ユーザーアカウントは、IdM から完全に削除されました。

2.6. コマンドラインでユーザーの復元

保存済みユーザーは、以下のステータスに復元できます。

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

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

前提条件

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

手順

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

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

または、ユーザーアカウントをステージユーザーとして復元することもできます。

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

検証手順

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

  ```
  $ ipa user-find
  ```

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

第3章 IDM WEB UI でユーザーアカウントの管理

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

ユーザーアカウントの作成

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

ユーザーアカウントをアクティベートする

従業員の最初の就業日に アカウントをアクティベート します。

ユーザーアカウントを無効にする

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

ユーザーアカウントを有効にする

ユーザーが戻ってきたら、アカウントを再度有効にする 必要があります。

ユーザーアカウントを保存する

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

ユーザーアカウントを復元する

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

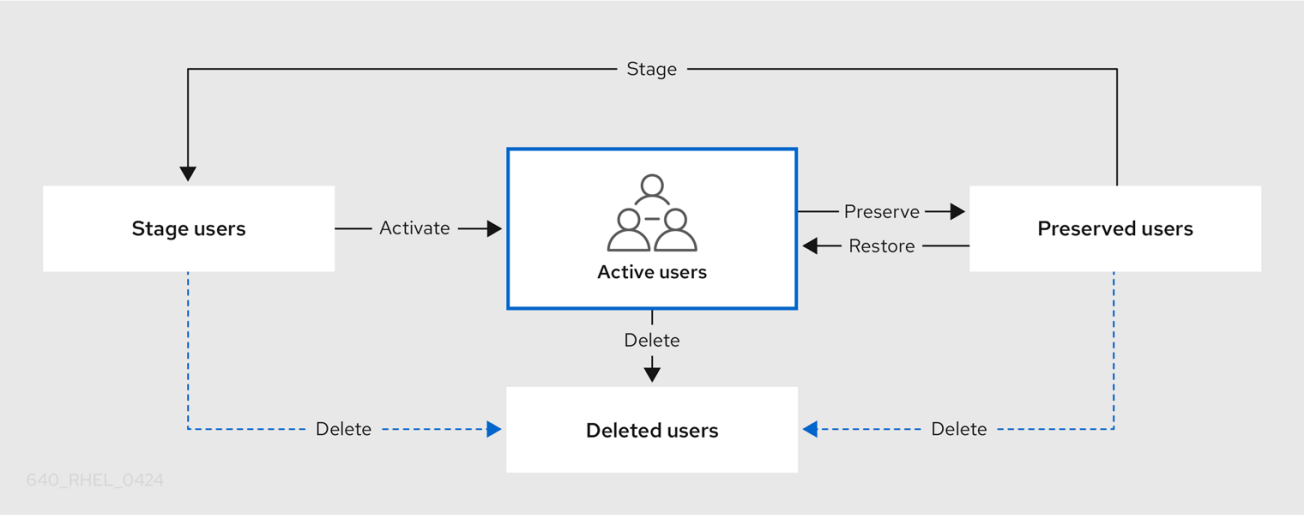
ユーザーアカウントを削除する

従業員が解雇された場合は、バックアップなしで アカウントを削除します。

3.1. ユーザーのライフサイクル

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

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



IdM データベースからユーザーエントリーを完全に削除できます。

重要

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

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

警告

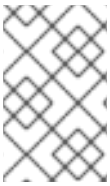
admin ユーザーを削除しないでください。**admin** は IdM で必要な事前定義ユーザーであるため、この操作では特定のコマンドで問題が生じます。別の **admin** ユーザーを定義して使用する場合は、管理者権限を少なくとも 1 つのユーザーに付与してから、**ipa user-disable admin** を使用して、事前定義された admin ユーザーを無効にします。

警告

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

3.2. WEB UI でユーザーの追加

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



注記

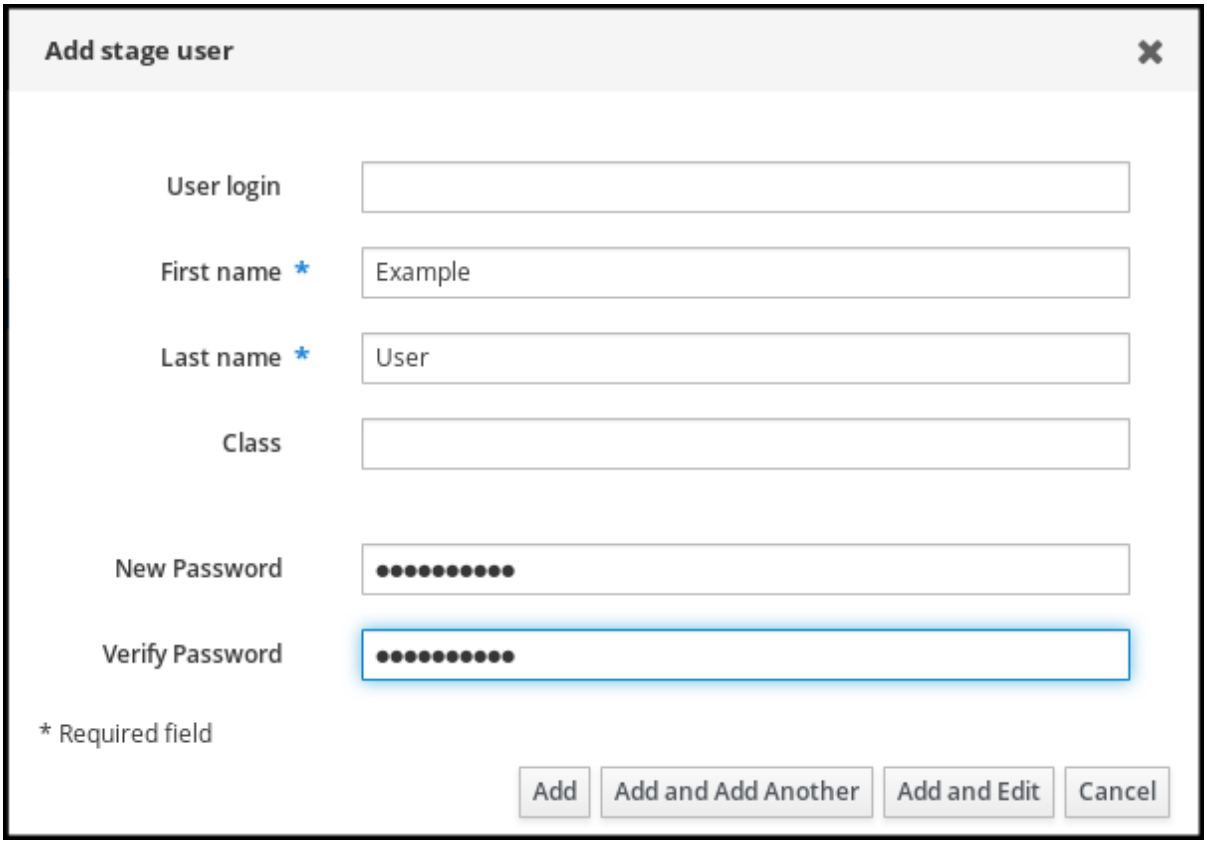
または、直接、アクティブなユーザーアカウントを作成できます。アクティブユーザーを追加する場合は、以下の手順に従って、**アクティブユーザー** タブでユーザーアカウントを追加します。

前提条件

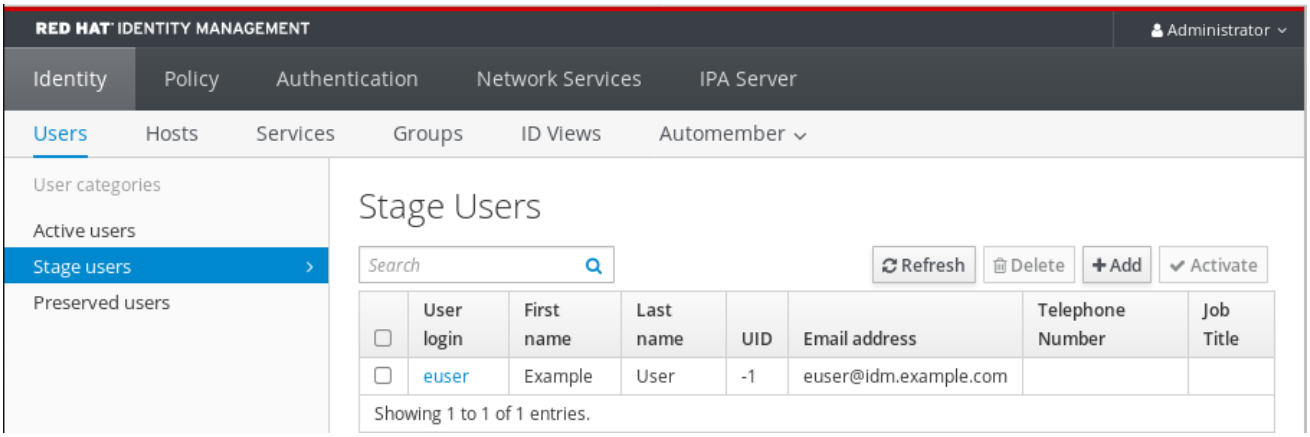
- IdM、またはユーザー管理者ロールを管理する管理者権限

手順

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



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



注記

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

警告

IdM は、新しいユーザーアカウントに一意のユーザー ID (UID) を自動的に割り当てます。管理者は、UID を手動で割り当てるだけでなく、既存の UID を変更することもできます。ただし、新しい UID 番号が一意であるかどうかは、サーバーによって検証されません。そのため、複数のユーザーエントリーに同じ UID 番号が割り当てられる可能性があります。ユーザーアカウントに GID (ユーザープライベートグループ ID) を手動で割り当てる場合も、ユーザープライベートグループ ID (GID) で同様の問題が発生する可能性があります。IdM CLI で **ipa user-find --uid=<uid>** または **ipa user-find --gidnumber=<gidnumber>** コマンドを使用すると、同じ ID を持つユーザーエントリーが複数あるかどうかを確認できます。

Red Hat は、複数のエントリーに同じ UID または GID を割り当てることがないようにすることを推奨します。ID が重複するオブジェクトがある、セキュリティー識別子 (SID) が正しく生成されません。SID は、IdM と Active Directory 間の信頼と、Kerberos 認証の正常な動作に不可欠です。

関連情報

- PAC 情報による Kerberos セキュリティーの強化
- Are user/group collisions supported in Red Hat Enterprise Linux?
- SID のないユーザーは、アップグレード後に IdM にログインできない

3.3. IDM WEB UI でステージユーザーのアクティベート

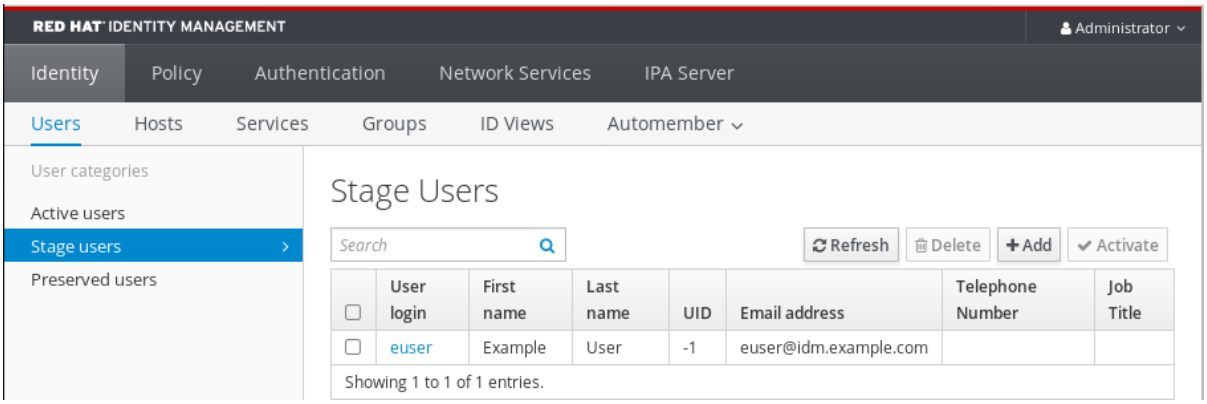
ユーザーが IdM にログインする前、およびユーザーを IdM グループに追加する前に、この手順に従ってステージユーザーアカウントをアクティベートする必要があります。

前提条件

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

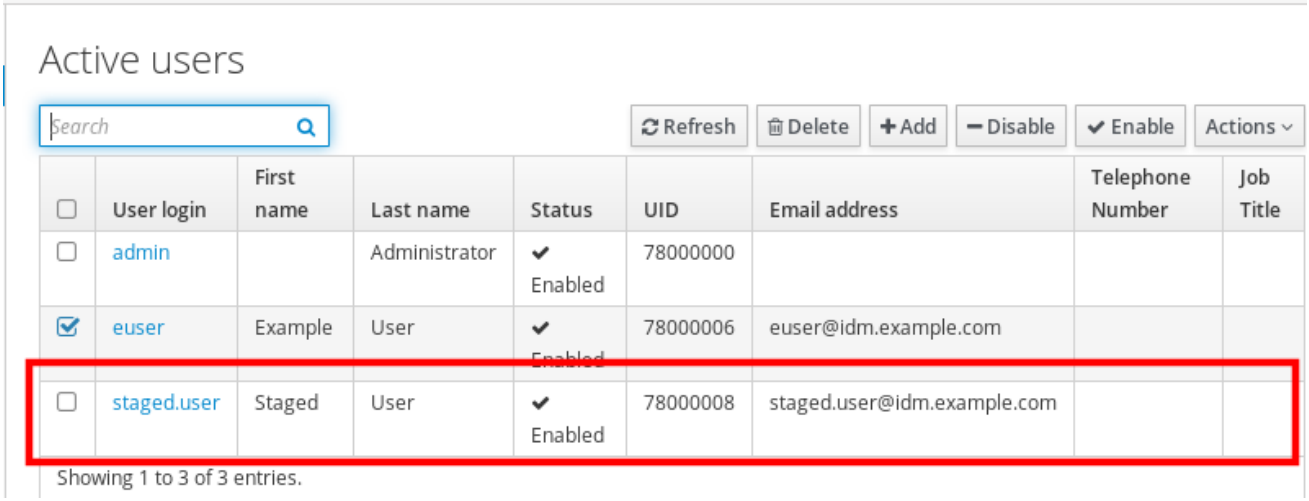
手順

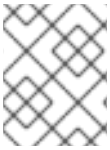
1. IdM Web UI にログインします。
2. **ユーザー → ステージユーザー** タブに移動します。
3. 有効にするユーザーアカウントのチェックボックスをクリックします。
4. **アクティベート** ボタンをクリックします。



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

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





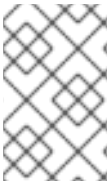
注記

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

3.4. WEB UI でのユーザーアカウントの無効化

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

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



注記

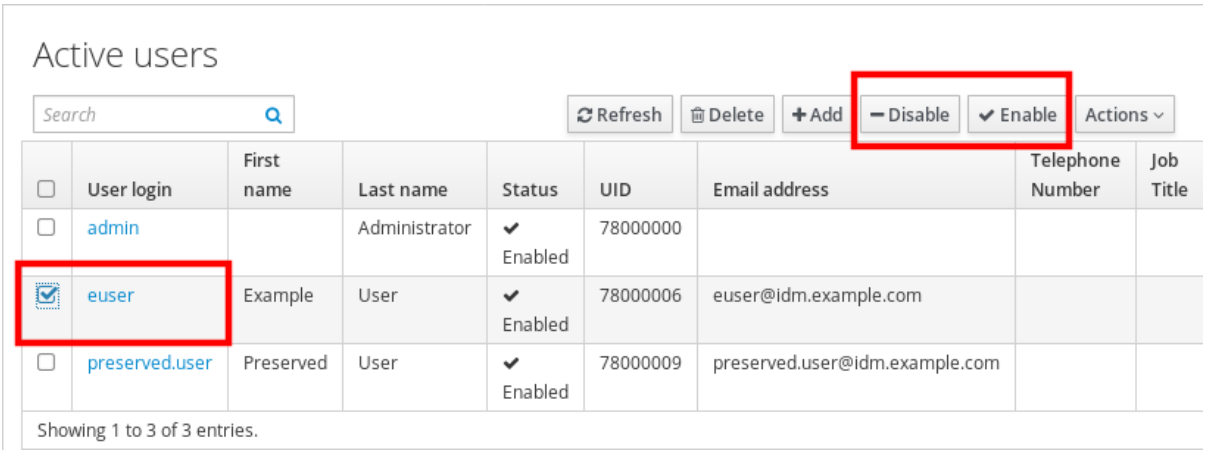
ユーザーアカウントを無効にした後、既存の接続はユーザーの Kerberos TGT や他のチケットの有効期限が切れるまで有効です。チケットの期限が切れると、ユーザーが更新できなくなります。

前提条件

- IdM Web UI、またはユーザー管理者ロールを管理する管理者権限

手順

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



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

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

User login	First name	Last name	Status	UID	Email address	Telephone Number
admin		Administrator	Enabled	78000000		
euser	Example	User	Disabled	78000006	euser@idm.example.com	
preserved.user	Preserved	User	Enabled	78000009	preserved.user@idm.example.com	

3.5. WEB UI でユーザーアカウントの有効化

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

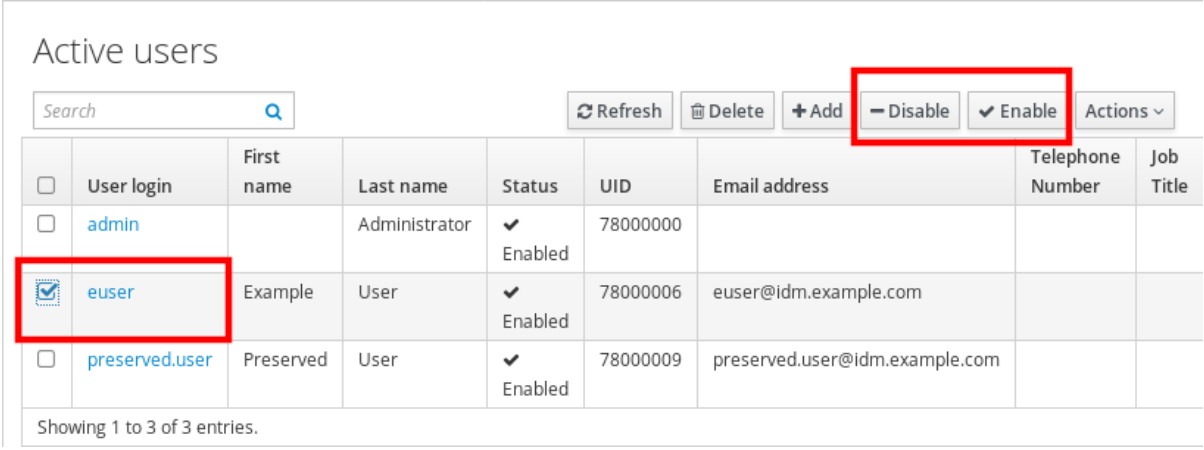
前提条件

- IdM Web UI、またはユーザー管理者ロールを管理する管理者権限

手順

1. IdM Web UI にログインします。
2. **ユーザー → アクティブユーザー** タブに移動します。

3. 有効にするユーザーアカウントのチェックボックスをクリックします。
4. **有効** ボタンをクリックします。



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

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

3.6. IDM WEB UI でアクティブなユーザーの保存

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

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

復元したアカウントをアクティブモードに戻すことができます。

注記

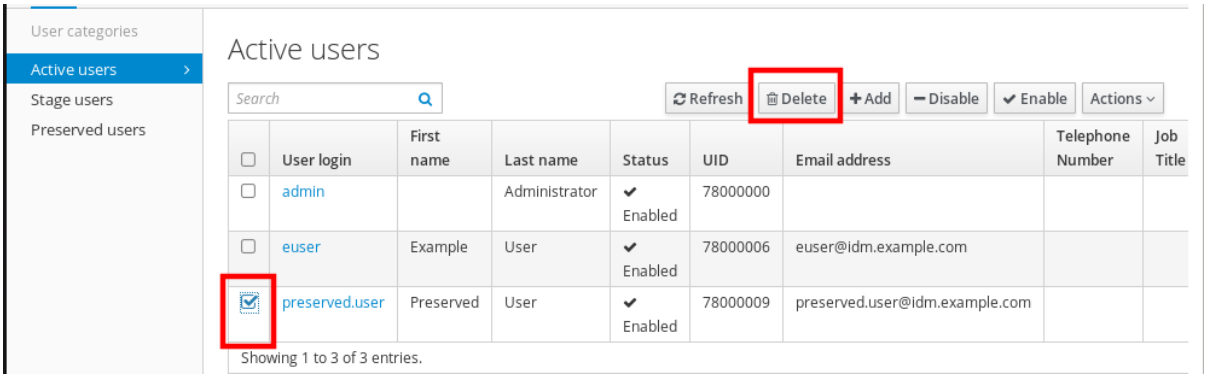
保存済みユーザーのリストは、以前のユーザーアカウントの履歴を提供します。

前提条件

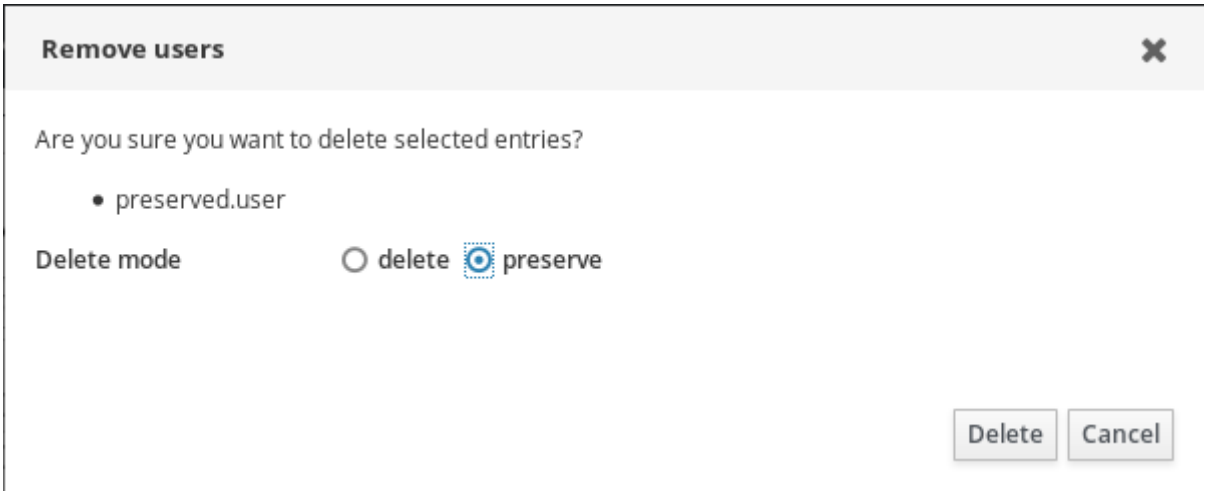
- IdM (Identity Management) Web UI、またはユーザー管理者ロールを管理する管理者権限

手順

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



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



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

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

3.7. IDM WEB UI でユーザーの復元

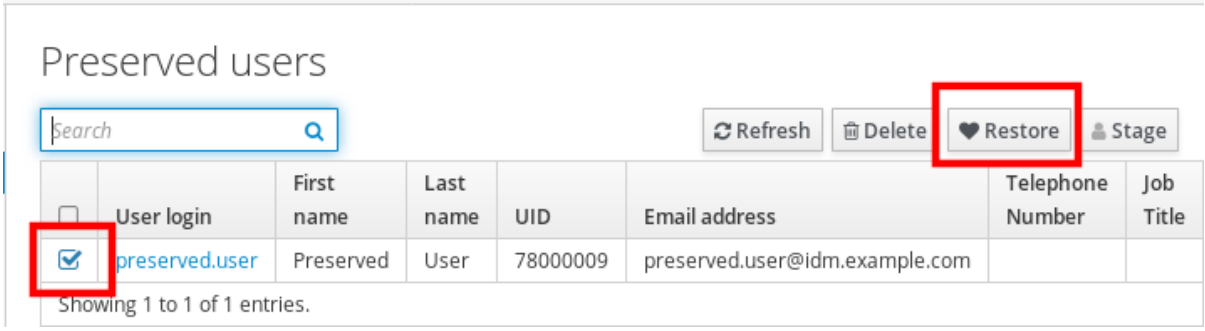
IdM (Identity Management) を使用すると、保存済みユーザーアカウントをアクティブな状態で復元できます。保存済みユーザーをアクティブなユーザーまたはステージユーザーに復元できます。

前提条件

- IdM Web UI、またはユーザー管理者ロールを管理する管理者権限

手順

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



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

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

3.8. IDM WEB UI でユーザーの削除

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

以下を削除できます。

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

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

- **ステージユーザー** タブ
- **保存済みユーザー** タブ

前提条件

- IdM Web UI、またはユーザー管理者ロールを管理する管理者権限

手順

1. IdM Web UI にログインします。
2. **ユーザー → アクティブユーザー** タブに移動します。
   ユーザー → ステージユーザー または **ユーザー → 保存済みユーザー** でも、ユーザーアカウントを削除できます。
3. **削除** アイコンをクリックします。
4. **ユーザーの削除** ダイアログボックスで、**モードの削除** ラジオボタンを、**削除** に切り替えます。

5. **削除** ボタンをクリックします。

ユーザーアカウントが、IdM から完全に削除されました。

第4章 ANSIBLE PLAYBOOK を使用したユーザーアカウントの管理

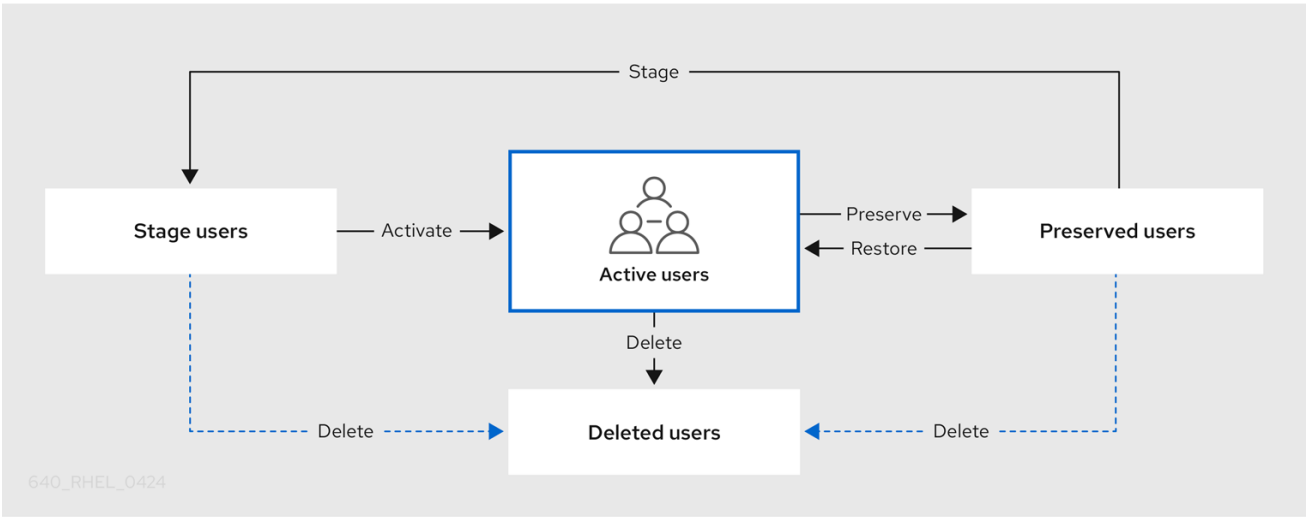
Ansible Playbook を使用して IdM のユーザーを管理できます。ユーザーのライフサイクル を示した後、本章では以下の操作に Ansible Playbook を使用する方法を説明します。

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

4.1. ユーザーのライフサイクル

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

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



IdM データベースからユーザーエントリーを完全に削除できます。

重要

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

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

警告

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

警告

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

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

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

前提条件

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

手順

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

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

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

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

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

ユーザーを追加するには、以下のオプションを使用する必要があります。

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

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

注記

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

3. Playbook を実行します。

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

検証手順

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

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

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

  2. admin の Kerberos チケットを要求します。

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

  3. **idm_user** に関する情報を要求します。

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

  idm_userという名前のユーザー が IdM に存在しています。

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

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

前提条件

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

手順

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

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

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

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

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

注記

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

3. Playbook を実行します。

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

検証手順

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

1. 管理者として **ipaserver** にログインします。

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

2. **idm_user_1** に関する情報を表示します。

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

idm_user_1 という名前のユーザーが IdM に存在しています。

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

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

前提条件

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

手順

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

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

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

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

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

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

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

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

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

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

検証手順

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

  1. 管理者として **ipaserver** にログインします。

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

2. **idm_user_1** に関する情報を表示します。

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

idm_user_1 という名前のユーザーが IdM に存在しています。

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

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

前提条件

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

手順

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

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

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

```
---
```

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

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

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

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

検証手順

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

1. 管理者として **ipaserver** にログインします。

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

2. **idm_user_1** に関する要求情報:

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

idm_user_1 という名前のユーザーは IdM に存在しません。

4.6. 関連情報

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

第5章 IDM でのユーザーパスワードの管理

5.1. IDM ユーザーパスワードは誰がどのように変更するのか

他のユーザーのパスワードを変更するパーミッションのない通常ユーザーは、独自の個人パスワードのみを変更できます。新しいパスワードは、そのユーザーがメンバーとなっているグループに適用される IdM パスワードポリシーに合致する必要があります。パスワードポリシーの設定方法の詳細は、IdM パスワードポリシーの定義 を参照してください。

管理者およびパスワード変更権限を持つユーザーは、新しいユーザーに初期パスワードを設定し、既存のユーザーのパスワードをリセットできます。これらのパスワードには、以下が該当します。

- IdM パスワードポリシーを満たす必要はありません。
- 最初のログインに成功したら失効します。このような場合、IdM はユーザーが期限切れのパスワードを直ちに変更するよう要求します。この動作を無効にするには、次回のログイン時にパスワード変更を求められることなく、IdM でパスワードリセットを有効にする を参照してください。

注記

LDAP Directory Manager(DM) ユーザーは、LDAP ツールを使用してユーザーパスワードを変更できます。新しいパスワードは、任意の IdM パスワードポリシーを上書きできます。DM によって設定されたパスワードは最初のログイン後に有効期限が切れません。

5.2. IDM WEB UI でのユーザーパスワードの変更

Identity Management (IdM) ユーザーは、IdM Web UI でユーザーパスワードを変更できます。

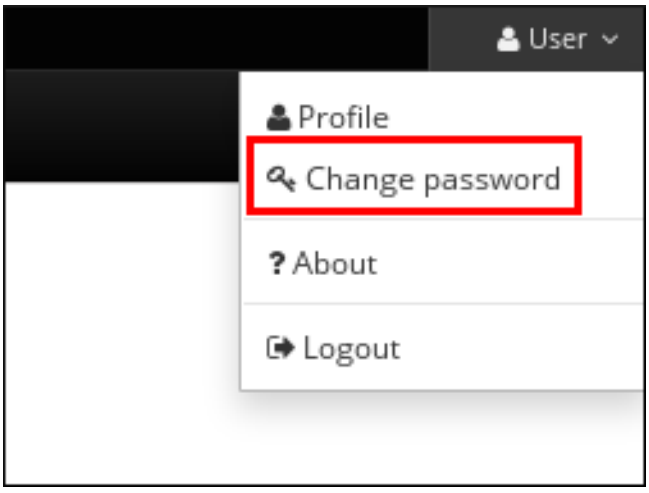
前提条件

- IdM Web UI にログインしている。

手順

1. 右上隅の **User name → Change password**をクリックします。

図5.1 パスワードのリセット



2. 現在のパスワードおよび新しいパスワードを入力します。

5.3. IDM WEB UI での別のユーザーのパスワードのリセット

Identity Management (IdM) の管理ユーザーは、IdM Web UI で他のユーザーのパスワードを変更できます。

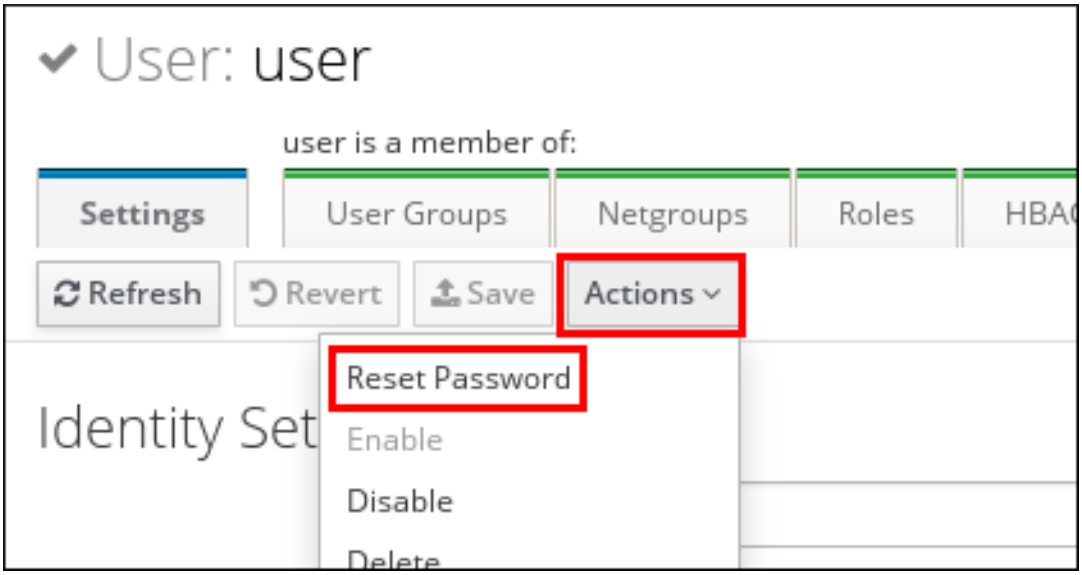
前提条件

- 管理ユーザーとして IdM Web UI にログインしている。

手順

1. **Identity → Users** を選択します。
2. 編集するユーザー名をクリックします。
3. **Actions → Reset password** をクリックします。

図5.2 パスワードのリセット



4. 新しいパスワードを入力し、**Reset Password** をクリックします。

図5.3 新しいパスワードの確認

5.4. DIRECTORY MANAGER ユーザーパスワードのリセット

Identity Management (IdM) Directory Manager のパスワードを紛失した場合は、リセットできます。

前提条件

- IdM サーバーに **root** にアクセスできる。

手順

1. **pwdhash** コマンドを使用して、新しいパスワードハッシュを生成します。以下に例を示します。

```
# pwdhash -D /etc/dirsrv/slapd-IDM-EXAMPLE-COM password
{PBKDF2_SHA256}AAAgABU0bKhyjY53NcxY33ueoPjOUWtl4iyYN5uW...
```

   Directory Server 設定へのパスを指定すると、**nsslapd-rootpwstoragescheme** 属性に設定されたパスワードストレージスキームが自動的に使用され、新しいパスワードを暗号化します。

2. トポロジー内のすべての IdM サーバーで、以下の手順を実行します。

   a. サーバーにインストールされている IdM サービスをすべて停止します。

```
# ipactl stop
```

   b. **/etc/dirsrv/IDM-EXAMPLE-COM/dse.ldif** ファイルーを編集し、**nsslapd-rootpw** 属性を、**pwdhash** コマンドで生成された値に設定します。

```
nsslapd-rootpw:
{PBKDF2_SHA256}AAAgABU0bKhyjY53NcxY33ueoPjOUWtl4iyYN5uW...
```

   c. サーバーにインストールされている IdM サービスをすべて起動します。

```
# ipactl start
```

5.5. IDM CLI でのユーザーパスワードの変更または別のユーザーのパスワードのリセット

Identity Management (IdM) コマンドラインインターフェイス (CLI) を使用して、ユーザーパスワードを変更できます。管理ユーザーの場合は、CLI を使用して別のユーザーのパスワードをリセットできます。

前提条件

- IdM ユーザーの TGT (Ticket-Granting Ticket) を取得している。
- 別のユーザーのパスワードをリセットする場合は、IdM の管理ユーザーの TGT を取得している必要がある。

手順

- ユーザーの名前と **--password** オプションを指定して、**ipa user-mod** コマンドを入力します。このコマンドにより、新しいパスワードの入力が求められます。

```
$ ipa user-mod idm_user --password
Password:
Enter Password again to verify:
```

```
--------------------
Modified user "idm_user"
--------------------
...
```

注記

ipa user-mod の代わりに **ipa passwd idm_user** を使用することもできます。

5.6. 次回のログイン時にパスワード変更を求められることなく、IDM でパスワードリセットを有効にする

デフォルトでは、管理者が別のユーザーのパスワードをリセットすると、初回のログインに成功したらパスワードが期限切れになります。IdM Directory Manager では、各 IdM 管理者に次の特権を指定できます。

- 初回ログイン後にパスワードの変更をユーザーに要求することなく、パスワードの変更操作を行うことができます。
- 強度や履歴の強制が適用されないようにパスワードポリシーをバイパスします。

警告

パスワードポリシーをバイパスすると、セキュリティー上の脅威になる可能性があります。これらの追加の特権を付与するユーザーを選択するときは注意してください。

前提条件

- Directory Manager のパスワードを把握している。

手順

1. ドメイン内のすべての Identity Management (IdM) サーバーで、次の変更を行います。
   a. **ldapmodify** コマンドを実行して、LDAP エントリーを変更します。IdM サーバーの名前と 389 ポートを指定し、Enter キーを押します。

   ```
   $ ldapmodify -x -D "cn=Directory Manager" -W -h server.idm.example.com -p 389
   Enter LDAP Password:
   ```

   b. Directory Manager パスワードを入力します。
   c. **ipa_pwd_extop** パスワード同期エントリーの識別名を入力し、Enter キーを押します。

   ```
   dn: cn=ipa_pwd_extop,cn=plugins,cn=config
   ```

   d. 変更の **modify** 型を指定し、Enter キーを押します。

```
changetype: modify
```

e. LDAP が実行する修正のタイプと、その属性を指定します。Enter キーを押します。

```
add: passSyncManagersDNs
```

f. **passSyncManagersDNs** 属性に管理ユーザーアカウントを指定します。属性は多値です。たとえば、**admin** ユーザーに、Directory Manager の電源をリセットするパスワードを付与するには、次のコマンドを実行します。

```
passSyncManagersDNs: \
uid=admin,cn=users,cn=accounts,dc=example,dc=com
```

g. Enter キーを 2 回押して、エントリーの編集を停止します。

手順全体を以下に示します。

```
$ ldapmodify -x -D "cn=Directory Manager" -W -h server.idm.example.com -p 389
Enter LDAP Password:
dn: cn=ipa_pwd_extop,cn=plugins,cn=config
changetype: modify
add: passSyncManagersDNs
passSyncManagersDNs: uid=admin,cn=users,cn=accounts,dc=example,dc=com
```

passSyncManagerDNs にリスト表示されている **admin** ユーザーに、追加特権が追加されました。

5.7. IDM ユーザーのアカウントがロックされているかどうかの確認

Identity Management (IdM) 管理者は、IdM ユーザーのアカウントがロックされているかどうかを確認できます。そのためには、ユーザーの最大許容ログイン試行回数と、ユーザーの実際の失敗ログイン回数を比較する必要があります。

前提条件

- IdM の管理ユーザーの TGT (Ticket-Granting Ticket) を取得している。

手順

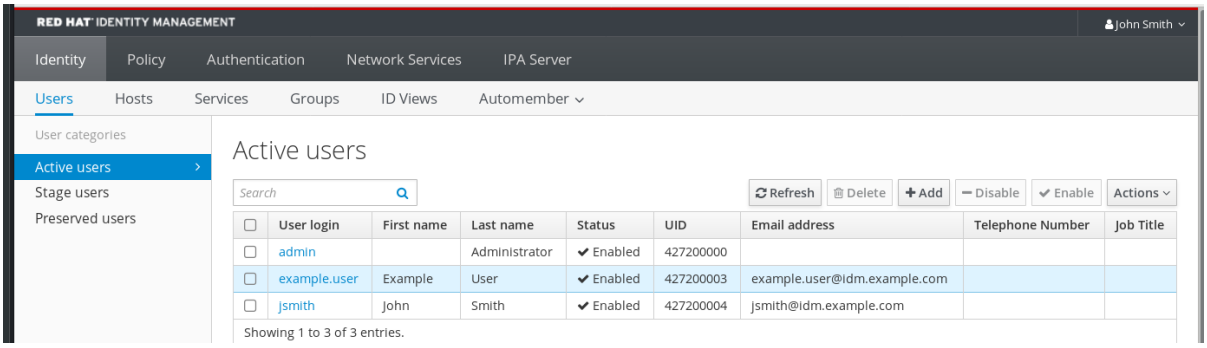
1. ユーザーアカウントのステータスを表示して、失敗したログインの数を確認します。

```
$ ipa user-status example_user
-----------------------
Account disabled: False
-----------------------
  Server: idm.example.com
  Failed logins: 8
  Last successful authentication: N/A
  Last failed authentication: 20220229080317Z
  Time now: 2022-02-29T08:04:46Z
----------------------------
Number of entries returned 1
----------------------------
```

2. 特定のユーザーに許可されたログイン試行回数を表示します。

   a. IdM 管理者として IdM Web UI にログインします。

   b. **Identity → Users → Active users**タブを開きます。



   a. ユーザー名をクリックして、ユーザー設定を開きます。

   b. **パスワードポリシー** セクションで、**Max failures** アイテムを探します。

3. **ipa user-status** コマンドの出力に表示されているログイン失敗回数と、IdM Web UI に表示されている**Max failures** 数を比較します。ログインに失敗した回数が、許可されている最大ログイン試行回数と等しい場合、ユーザーアカウントはロックされます。

関連情報

- IIdM でのパスワード失敗後のユーザーアカウントのロック解除

5.8. IIDM でのパスワード失敗後のユーザーアカウントのロック解除

ユーザーが間違ったパスワードを一定回数使用してログインしようとすると、Identity Management (IdM) はユーザーアカウントをロックするため、ユーザーはログインできなくなります。セキュリティー上の理由から、IdM では、ユーザーアカウントがロックされていることを示す警告メッセージは表示されません。代わりに、CLI プロンプトがユーザーにパスワードを何度も要求し続ける場合があります。

IdM は、指定した時間が経過した後にユーザーアカウントを自動的にアンロックします。または、以下の手順でユーザーアカウントのロックを手動で解除することもできます。

前提条件

- IdM 管理ユーザーの Ticket-Granting Ticket を取得している。

手順

- ユーザーアカウントのロックを解除するには、**ipa user-unlock** コマンドを実行します。

```
$ ipa user-unlock idm_user
-----------------------
Unlocked account "idm_user"
-----------------------
```

この後、ユーザーは再度ログインできるようになります。

関連情報

- IdM ユーザーのアカウントがロックされているかどうかの確認

5.9. IDM のユーザーに対する、最後に成功した KERBEROS 認証の追跡の有効化

パフォーマンス上の理由から、Red Hat Enterprise Linux 8 で実行している Identity Management (IdM) には、ユーザーが最後に成功した Kerberos 認証のタイムスタンプが保存されません。そのため、**ipa user-status** などの特定のコマンドではタイムスタンプが表示されません。

前提条件

- IdM の管理ユーザーの TGT (Ticket-Granting Ticket) を取得している。
- 手順を実行している IdM サーバーへの **root** アクセス権限がある。

手順

1. 現在有効なパスワードプラグイン機能を表示します。

```
# ipa config-show | grep "Password plugin features"
  Password plugin features: AllowNThash, KDC:Disable Last Success
```

この出力は、**KDC:Disable Last Success** プラグインが有効になっていることを示しています。このプラグインにより、最後に成功した Kerberos 認証試行が ipa user-status 出力に表示されなくなります。

2. 現在有効な **ipa config-mod** コマンドに、すべての機能の **--ipaconfigstring=feature** パラメーターを追加します (**KDC:Disable Last Success** を除く)。

```
# ipa config-mod --ipaconfigstring='AllowNThash'
```

このコマンドは、**AllowNThash** プラグインのみを有効にします。複数の機能を有効にするには、機能ごとに **--ipaconfigstring=feature** パラメーターを個別に指定します。

3. IdM を再起動します。

```
# ipactl restart
```

第6章 IDM パスワードポリシーの定義

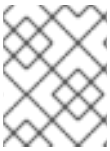
本章では、Identity Management (IdM) パスワードポリシーについて、また、Ansible Playbook を使用して IdM に新規パスワードポリシーを追加する方法を説明します。

6.1. パスワードポリシーとは

パスワードポリシーは、パスワードが満たす必要のある一連のルールです。たとえば、パスワードポリシーでは、パスワードの最小長と最大有効期間を定義できます。このポリシーの対象となる全ユーザーには、十分に長いパスワードを設定して、指定の条件を満たす頻度でパスワードを変更する必要があります。このようにパスワードポリシーを使用することで、ユーザーのパスワードが検出されて悪用されるリスクが軽減されます。

6.2. IDM のパスワードポリシー

パスワードは、Identity Management (IdM) ユーザーが IdM Kerberos ドメインに対して認証する最も一般的な方法です。パスワードポリシーでは、このような IdM ユーザーのパスワードが満たす必要条件を定義します。



注記

IdM パスワードポリシーは基礎となる LDAP ディレクトリーで設定されますが、Kerberos Key Distribution Center (KDC) はパスワードポリシーを強制的に使用します。

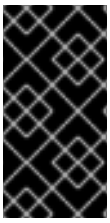
パスワードポリシー属性 は、IdM でのパスワードポリシーの定義に使用できる属性をリスト表示します。

表6.1 パスワードポリシーの属性

属性	説明	例
Max lifetime	パスワードのリセットが必要になるまでの、パスワードの有効期間 (日) の上限です。デフォルト値は 90 日です。 この属性が 0 に設定されている場合、パスワードの有効期限は切れないことに注意してください。	Max lifetime = 180 ユーザーパスワードは 180 日間のみ有効です。有効期限が経過すると、IdM は変更を求めるプロンプトを表示します。
Min lifetime	パスワードを変更してから次に変更操作を行うまでに最小限開ける必要のある時間。	Min lifetime = 1 ユーザーがパスワードの変更後に、次に変更するまでに最低でも 1 時間待機する必要があります。
History size	保存される以前のパスワード数。パスワードの履歴にあるパスワードを再利用できませんが、保存されていない以前のものは使用できます。	History size = 0 この場合、パスワード履歴は空になり、ユーザーは以前のパスワードをどれでも再利用できます。

属性	説明	例
Character classes	パスワードで使用する必要のある文字クラスの数。文字クラスは次のとおりです。 * 大文字 * 小文字 * 数字 * コンマ (,)、ピリオド (.)、アスタリスク (*) などの特殊文字 * 他の UTF-8 文字 1 つの文字を複数回連続で使用すると、文字クラスが 1 つ減少します。以下に例を示します。 * **Secret1** には、大文字、小文字、数字の 3 つの文字クラスがあります。 * **Secret111** には、大文字、小文字、数字が含まれていますが、**1** を 繰り返し使用使用したため、ペナルティが -1 で文字クラスが 2 つになりります。	Character classes = 0 必要なクラスのデフォルト数は 0 です。番号を設定するには、**--minclasses** オプションを指定して **ipa pwpolicy-mod** コマンドを実行します。 この表の下に記載されている 重要 の注意事項も併せて参照してください。
Min length	パスワードの最小長。 追加のパスワードポリシーオプション のいずれかが設定されていると、パスワードの最小長は 6 文字です。	Min length = 8 8 文字未満のパスワードは使用できません。
Max failures	IdM がユーザーアカウントをロックするまでのログイン試行の最大失敗数。	Max failures = 6 ユーザーがパスワードを誤って 7 回入力すると、IdM はユーザーアカウントをロックします。
Failure reset interval	失敗したログイン試行回数を IdM がリセットするまでの時間 (秒単位)。	Failure reset interval = 60 **Max failures** で定義されたログイン試行回数が 1 分以上経過すると、ユーザーはユーザーアカウントがロックされる心配なく再ログインを試みることができます。

属性	説明	例
Lockout duration	**Max failures** で定義された回数のログイン試行に失敗した後にユーザーアカウントがロックされる時間 (秒単位)。	Lockout duration = 600 アカウントがロックされると、10 分間ログインできません。



重要

国際文字や記号を使用できないハードウェアセットが各種ある場合には、文字クラス要件に英語と共通記号を使用してください。パスワードの文字クラスポリシーの詳細は、Red Hat ナレッジベースの What characters are valid in a password? を参照してください。

6.3. ANSIBLE PLAYBOOK を使用して IDM にパスワードポリシーを存在させる手順

Ansible Playbook を使用して Identity Management (IdM) にパスワードポリシーを存在させるには、次の手順に従います。

IdM におけるデフォルトの **global_policy** パスワードポリシーでは、パスワード内の異なる文字クラスの数は 0 に設定されています。履歴サイズも 0 に設定されています。

以下の手順に従って、Ansible Playbook を使用して、IdM グループにより強力なパスワードポリシーを適用します。

注記

IdM グループのパスワードポリシーのみを定義できます。個別ユーザーにパスワードポリシーを定義できません。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
  - この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。
- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。
- IdM 管理者パスワードを把握している。
- IdM にパスワードポリシーが存在することを確認するグループ。

手順

1. **inventory.file** などのインベントリーファイルを作成し、**[ipaserver]** セクションに IdM サーバーの **FQDN** を定義します。

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

2. Ansible Playbook を作成して、存在させるパスワードポリシーを定義します。この手順は、**/usr/share/doc/ansible-freeipa/playbooks/pwpolicy/pwpolicy_present.yml** ファイルの例をコピーして変更し、簡素化できます。

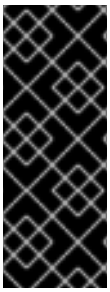
```
---
- name: Tests
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure presence of pwpolicy for group ops
    ipapwpolicy:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: ops
      minlife: 7
      maxlife: 49
      history: 5
      priority: 1
      lockouttime: 300
      minlength: 8
      minclasses: 4
      maxfail: 3
      failinterval: 5
```

   各変数の意味については、パスワードポリシーの属性 を参照してください。

3. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file
path_to_playbooks_directory_/new_pwpolicy_present.yml
```

Ansible Playbook を使用して、**ops** グループのパスワードポリシーを IdM に存在させることができました。



重要

ops パスワードポリシーの優先度は 1に設定されますが、**global_policy** パスワードポリシーには優先度が設定されません。上記の理由から、**ops** ポリシーは **ops** グループの**global_policy** より自動的に優先され、すぐに有効になります。

global_policy は、ユーザーにポリシーが設定されていない場合のフォールバックポリシーとして機能し、グループポリシーよりも優先されることはありません。

関連情報

- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-pwpolicy.md** ファイルを参照してください。

- Password policy priorities を参照してください。

6.4. IDM での追加のパスワードポリシーオプション

Identity Management (IdM) 管理者は、**libpwquality** 機能セットに基づく追加のパスワードポリシーオプションを有効にすることで、デフォルトのパスワード要件を強化できます。追加のパスワードポリシーオプションには、以下が含まれます。

--maxrepeat

新しいパスワードに使用できる、連続する同一文字数の上限を指定します。

--maxsequence

新しいパスワードにおける単調な文字シーケンスの最大長を指定します。このような配列の例は、**12345** または **fedcb** です。このようなパスワードのほとんどは、簡素化チェックに合格しません。

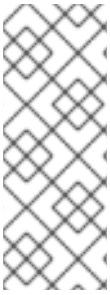
--dictcheck

ゼロ以外の場合は、パスワード (修正可能) が辞書の単語と一致するかどうかを確認します。現在、**libpwquality** は、**cracklib** ライブラリーを使用してディクショナリーの確認を実行しています。

--usercheck

ゼロ以外の場合は、パスワード (修正可能) に、何らかの形式でユーザー名が含まれているかどうかを確認します。ユーザー名が 3 文字より短い場合は実行されません。

既存のパスワードには、追加のパスワードポリシーオプションを適用できません。追加オプションのいずれかを適用すると、IdM は、パスワードの最小文字数である **--minlength** オプションを自動的に 6 文字に設定します。



注記

RHEL 7、RHEL 8、RHEL 9 サーバーが混在する環境では、RHEL 8.4 以降で実行されているサーバーにのみ追加のパスワードポリシー設定を適用できます。ユーザーが IdM クライアントにログインし、IdM クライアントが RHEL 8.3 以前で実行している IdM サーバーと通信している場合は、システム管理者が設定した新しいパスワードポリシーの要件は適用されません。一貫した動作を確認するには、すべてのサーバーを RHEL 8.4 以降にアップグレードまたは更新します。

関連情報:

- IdM グループへの追加のパスワードポリシーの適用
- **pwquality(3)** man ページ

6.5. IDM グループへの追加のパスワードポリシーオプションの適用

Identity Management (IdM) で追加のパスワードポリシーオプションを適用するには、次の手順に従います。ここでは、新しいパスワードにユーザー名が含まれていないことと、パスワードに同じ文字が連続して 2 文字以内になるようにすることで、**マネージャー** グループのパスワードポリシーを強化する方法を説明します。

前提条件

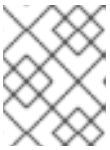
- IdM 管理者としてログインしている。

- **マネージャー** グループが IdM に存在している。
- **マネージャー** パスワードポリシーが IdM に存在している。

手順

1. **マネージャー** グループのユーザーが提案するすべての新しいパスワードに、ユーザー名の確認を適用します。

```
$ ipa pwpolicy-mod --usercheck=True managers
```



注記

パスワードポリシーの名前を指定しないと、デフォルトの **global_policy** が変更されます。

2. **マネージャー** パスワードポリシーで、同一の連続した文字の上限を 2 に設定します。

```
$ ipa pwpolicy-mod --maxrepeat=2 managers
```

パスワードに、同一の連続した文字が 2 文字を超える場合は、パスワードが使用できなくなります。たとえば、**eR873mUi111YJQ** の組み合わせは、連続して 3 つの **1** を含むため、使用できません。

検証

1. **test_user** という名前のテストユーザーを追加します。

```
$ ipa user-add test_user
First name: test
Last name: user
----------------------------
Added user "test_user"
----------------------------
```

2. テストユーザーを **マネージャー** グループに追加します。
   a. IdM Web UI で、**Identity → Groups → User Groups** をクリックします。
   b. **managers** をクリックします。
   c. **Add** をクリックします。
   d. **Add users into user group 'managers'**ページで、**test_user** をチェックします。
   e. **>** 矢印をクリックして、ユーザーを **Prospective** 列に移動します。
   f. **Add** をクリックします。
3. テストユーザーのパスワードをリセットします。
   a. **Identity → Users** に移動します。
   b. **test_user** をクリックします。

c. **Actions** メニューで、**Reset Password** をクリックします。

d. ユーザーの一時パスワードを入力します。

4. コマンドラインで、**test_user** の Kerberos Ticket-Granting Ticket (TGT) を取得してみてください。

```
$ kinit test_user
```

a. 一時パスワードを入力します。

b. パスワードを変更する必要があることがシステムから通知されます。**test_user** のユーザー名を含むパスワードを入力します。

```
Password expired. You must change it now.
Enter new password:
Enter it again:
Password change rejected: Password not changed.
Unspecified password quality failure while trying to change password.
Please try again.
```

注記

Kerberos には、詳細なエラーパスワードポリシーの報告はなく、特定のケースでは、パスワードが拒否された理由を明確に示していません。

c. 入力したパスワードが拒否されたことがシステムから通知されます。連続して 3 文字以上の同一文字を含むパスワードを入力します。

```
Password change rejected: Password not changed.
Unspecified password quality failure while trying to change password.
Please try again.

Enter new password:
Enter it again:
```

d. 入力したパスワードが拒否されたことがシステムから通知されます。**マネージャー** パスワードポリシーの基準を満たすパスワードを入力します。

```
Password change rejected: Password not changed.
Unspecified password quality failure while trying to change password.
Please try again.

Enter new password:
Enter it again:
```

5. 取得した TGT を表示します。

```
$ klist
Ticket cache: KCM:0:33945
Default principal: test_user@IDM.EXAMPLE.COM
```

```
Valid starting       Expires              Service principal
07/07/2021 12:44:44  07/08/2021 12:44:44
krbtgt@IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
```

マネージャー のパスワードポリシーが、**マネージャー** グループのユーザーに対して正しく機能するようになりました。

関連情報

- IdM での追加のパスワードポリシー

6.6. ANSIBLE PLAYBOOK を使用して追加のパスワードポリシーオプションを IDM グループに適用する

Ansible Playbook を使用して追加のパスワードポリシーオプションを適用し、特定の IdM グループのパスワードポリシー要件を強化できます。この目的には、**maxrepeat**、**maxsequence**、**dictcheck**、および **usercheck** パスワードポリシーオプションを使用できます。この例では、**managers** グループに次の要件を設定する方法を説明します。

- ユーザーの新しいパスワードに、ユーザーのそれぞれのユーザー名が含まれていない。
- パスワードに含まれる連続する同一の文字が 2 文字以下である。
- パスワードに含まれる単調な文字列が 3 文字以内である。これは、システムが **1234** や **abcd** などの文字列を含むパスワードを受け入れないことを意味します。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している。
  - **secret.yml** Ansible vault に **ipaadmin_password** が保存されている。
- IdM にパスワードポリシーが存在することを確認するグループ。

手順

1. Ansible Playbook ファイル **manager_pwpolicy_present.yml** を作成して、存在させるパスワードポリシーを定義します。この手順を簡素化するには、次の例をコピーして変更します。

```
---
- name: Tests
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure presence of usercheck and maxrepeat pwpolicy for group managers
    ipapwpolicy:
```

```
    ipaadmin_password: "{{ ipaadmin_password }}"
    name: managers
    usercheck: True
    maxrepeat: 2
    maxsequence: 3
```

2. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file
path_to_playbooks_directory_/manager_pwpolicy_present.yml
```

検証

1. **test_user** という名前のテストユーザーを追加します。

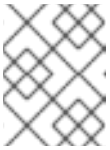
```
$ ipa user-add test_user
First name: test
Last name: user
----------------------------
Added user "test_user"
----------------------------
```

2. テストユーザーを **マネージャー** グループに追加します。
   a. IdM Web UI で、**Identity → Groups → User Groups** をクリックします。
   b. **managers** をクリックします。
   c. **Add** をクリックします。
   d. **Add users into user group 'managers'**ページで、**test_user** をチェックします。
   e. **>** 矢印をクリックして、ユーザーを **Prospective** 列に移動します。
   f. **Add** をクリックします。
3. テストユーザーのパスワードをリセットします。
   a. **Identity → Users** に移動します。
   b. **test_user** をクリックします。
   c. **Actions** メニューで、**Reset Password** をクリックします。
   d. ユーザーの一時パスワードを入力します。
4. コマンドラインで、**test_user** の Kerberos Ticket-Granting Ticket (TGT) を取得してみてください。

```
$ kinit test_user
```

   a. 一時パスワードを入力します。
   b. パスワードを変更する必要があることがシステムから通知されます。**test_user** のユーザー名を含むパスワードを入力します。

```
Password expired. You must change it now.
Enter new password:
Enter it again:
Password change rejected: Password not changed.
Unspecified password quality failure while trying to change password.
Please try again.
```



注記

Kerberos には、詳細なエラーパスワードポリシーの報告はなく、特定のケースでは、パスワードが拒否された理由を明確に示していません。

c. 入力したパスワードが拒否されたことがシステムから通知されます。連続して 3 文字以上の同一文字を含むパスワードを入力します。

```
Password change rejected: Password not changed.
Unspecified password quality failure while trying to change password.
Please try again.

Enter new password:
Enter it again:
```

d. 入力したパスワードが拒否されたことがシステムから通知されます。3 文字を超える単調な文字列を含むパスワードを入力します。たとえば、**1234** や **fedc** などの文字列です。

```
Password change rejected: Password not changed.
Unspecified password quality failure while trying to change password.
Please try again.

Enter new password:
Enter it again:
```

e. 入力したパスワードが拒否されたことがシステムから通知されます。**マネージャー** パスワードポリシーの基準を満たすパスワードを入力します。

```
Password change rejected: Password not changed.
Unspecified password quality failure while trying to change password.
Please try again.

Enter new password:
Enter it again:
```

5. TGT を取得したことを確認します。これは、有効なパスワードを入力した後にのみ可能です。

```
$ klist
Ticket cache: KCM:0:33945
Default principal: test_user@IDM.EXAMPLE.COM

Valid starting       Expires              Service principal
07/07/2021 12:44:44  07/08/2021 12:44:44
krbtgt@IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
```

関連情報

- IdM での追加のパスワードポリシー
- **/usr/share/doc/ansible-freeipa/README-pwpolicy.md**
- **/usr/share/doc/ansible-freeipa/playbooks/pwpolicy**

第7章 パスワード失効に関する通知の管理

ipa-client-epn パッケージに含まれる Expiring Password Notification (EPN) ツールを使用して、設定期間内にパスワードが失効する Identity Management (IdM) ユーザーのリストを構築できます。EPN ツールをインストール、設定、および使用するには、該当のセクションを参照してください。

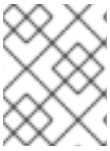
- Expiring Password Notification (EPN) ツールの概要
- Expiring Password Notification ツールのインストール
- EPN ツールを実行してパスワードが失効するユーザーへのメール送信
- ipa-epn.timer を有効にして、パスワードが失効する全ユーザーへのメールの送信
- Expiring Password Notification (EPN) のメールテンプレートの変更

7.1. EXPIRING PASSWORD NOTIFICATION (EPN) ツールの概要

Expiring Password Notification (EPN) ツールは、設定期間内にパスワードが失効する Identity Management (IdM) ユーザーのリストの作成に使用可能なスタンドアロンツールです。

IdM 管理者は、EPN を使用して以下を行うことができます。

- 対象ユーザーのリストを JSON 形式で表示する。これは、ドライランモードを実行時に作成されます。
- 特定の日または日付の範囲に送信される電子メール数を計算する。
- パスワード期限切れのメール通知をユーザーに送信する。
- **lpa-epn.timer** が EPN ツールを毎日実行し、定義済みの未来の日付範囲内にパスワードが執行するユーザーに対してメールを送信するように設定する。
- メール通知をカスタマイズして、ユーザーに送信する。



注記

ユーザーアカウントが無効な場合には、パスワードが期限切れなってもメール通知は送信されません。

7.2. EXPIRING PASSWORD NOTIFICATION ツールのインストール

Expiring Password Notification (EPN) ツールをインストールするには、次の手順に従います。

前提条件

- スマートホストで設定したローカルの Postfix SMTP サーバーを使用して、Identity Management (IdM) レプリカまたは IdM クライアントに EPN ツールをインストールします。

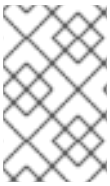
手順

- EPN ツールをインストールします。

```
# dnf install ipa-client-epn
```

7.3. EPN ツールを実行してパスワードが失効するユーザーへのメール送信

Expiring Password Notification (EPN) ツールを実行してパスワードの期限が切れるユーザーにメールを送信するには、次の手順に従います。



注記

EPN ツールはステートレスです。特定の日付にパスワードが失効するユーザーに対してメールの送信に失敗した場合には、EPN ツールには失敗したユーザーのリストは保存されません。

前提条件

- **ipa-client-epn** パッケージがインストールされている。Expiring Password Notification ツールのインストール を参照してください。
- 必要に応じて、**ipa-epn** メールテンプレートをカスタマイズする。期限切れのパスワード通知テンプレートの変更 を参照してください。

手順

1. **epn.conf** 設定ファイルを更新して、今後パスワードが失効するユーザーに通知されるように、EPN ツールのオプションを設定します。

```
# vi /etc/ipa/epn.conf
```

2. 必要に応じて **notify_ttls** を更新します。デフォルトでは、28、14、7、3 および 1 日以内にパスワードが期限切れになるユーザーに通知します。

```
notify_ttls = 28, 14, 7, 3, 1
```

3. SMTP サーバーおよびポートを設定します。

```
smtp_server = localhost
smtp_port = 25
```

4. メールで失効通知を送信するメールアドレスを指定します。配信に失敗したメールは以下のアドレスに返されます。

```
mail_from =admin-email@example.com
```

5. **/etc/ipa/epn.conf** ファイルを保存します。

6. **--dry-run** オプションなしでツールを実行した場合には、EPN ツールをドライランモードで実行し、パスワード失効メールの通知を送信するユーザーのリストを生成します。

```
ipa-epn --dry-run
[
    {
     "uid": "user5",
     "cn": "user 5",
     "krbpasswordexpiration": "2020-04-17 15:51:53",
     "mail": "['user5@ipa.test']"
```

```
    }
]
[
    {
     "uid": "user6",
     "cn": "user 6",
     "krbpasswordexpiration": "2020-12-17 15:51:53",
     "mail": "['user5@ipa.test']"
     }
]
The IPA-EPN command was successful
```

注記

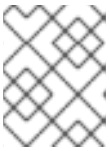
返されたユーザーのリストが非常に大きく、かつ **--dry-run** オプションなしでツールを実行すると、メールサーバーで問題が発生する可能性があります。

7. ドライランモードで EPN ツールを実行時に返された全ユーザーのリストに失効メールを送信するには、**--dry-run** オプションをなしで EPN ツールを実行します。

```
ipa-epn
[
  {
     "uid": "user5",
     "cn": "user 5",
     "krbpasswordexpiration": "2020-10-01 15:51:53",
     "mail": "['user5@ipa.test']"
  }
]
[
  {
    "uid": "user6",
    "cn": "user 6",
    "krbpasswordexpiration": "2020-12-17 15:51:53",
    "mail": "['user5@ipa.test']"
  }
]
The IPA-EPN command was successful
```

8. EPN を監視システムに追加して、**--from-nbdays** および **--to-nbdays** オプションで EPN を呼び出し、特定の時間内に期限切れになるユーザーパスワード数を確認できます。

```
# ipa-epn --from-nbdays 8 --to-nbdays 12
```



注記

--from-nbdays および **--to-nbdays** で EPN ツールを呼び出すと、自動的にドライランモードで実行されます。

検証手順

- EPN ツールを実行し、メール通知が送信されていることを確認します。

関連情報

関連情報

- **ipa-epn** の man ページを参照してください。
- **epn.conf** の man ページを参照してください。

7.4. IPA-EPN.TIMER を有効にして、パスワードが失効する全ユーザーへのメールの送信

ipa-epn.timer を使用して Expiring Password Notification (EPN) ツールを実行し、パスワードの期限が切れるユーザーにメールを送信するには、次の手順に従います。**ipa-epn.timer** は **epn.conf** ファイルを解析し、そのファイルに設定された未来の日付範囲内にパスワードの期限が切れるユーザーにメールを送信します。

前提条件

- **ipa-client-epn** パッケージがインストールされている。 Expiring Password Notification (EPN) ツールのインストール を参照してください。
- 必要に応じて、**ipa-epn** メールテンプレートをカスタマイズする。Expiring Password Notification (EPN) のメールテンプレートの変更 を参照してください。

手順

- **ipa-epn.timer** を起動します。

```
systemctl start ipa-epn.timer
```

タイマーを起動すると、デフォルトでは、EPN ツールは毎日午前 1 時に実行されます。

関連情報

- **ipa-epn** の man ページを参照してください。

7.5. EXPIRING PASSWORD NOTIFICATION (EPN) のメールテンプレートの変更

Expiring Password Notification (EPN) のメールメッセージのテンプレートをカスタマイズするには、次の手順に従います。

前提条件

- **ipa-client-epn** パッケージがインストールされている。

手順

1. EPN メッセージテンプレートを開きます。

```
# vi /etc/ipa/epn/expire_msg.template
```

2. 必要に応じてテンプレートテキストを更新します。

```
Hi {{ fullname }},
```

```
Your password will expire on {{ expiration }}.

Please change it as soon as possible.
```

テンプレートでは以下の変数を使用できます。

- User ID: uid
- Full name: fullname
- First name: first
- Last name: last
- Password expiration date: expiration

3. メッセージテンプレートファイルを保存します。

検証手順

- EPN ツールを実行し、メール通知に更新したテキストが含まれていることを確認します。

関連情報

- **ipa-epn** の man ページを参照してください。

第8章 IDM クライアントの IDM ユーザーへの SUDO アクセスの許可

Identity Management でユーザーに **sudo** アクセス権を付与する方法を詳しく説明します。

8.1. IDM クライアントの SUDO アクセス

システム管理者は、root 以外のユーザーに、通常 **root** ユーザー用に予約されている管理コマンドを実行できるようにする **sudo** アクセスを付与できます。その結果、ユーザーが、通常、**root** ユーザー用に予約される管理コマンドを実行する場合は、コマンドの前に **sudo** を付けることができます。パスワードを入力すると、そのコマンドは **root** ユーザーとして実行されます。データベースサービスアカウントなどの別のユーザーまたはグループとして **sudo** コマンドを実行するには、**sudo** ルールの **RunAs エイリアス** を設定できます。

Red Hat Enterprise Linux (RHEL) 8 ホストが Identity Management (IdM) クライアントとして登録されている場合は、以下の方法で、どの IdM ユーザーがホストでどのコマンドを実行できるかを定義する **sudo** ルールを指定できます。

- ローカルの **/etc/sudoers** ファイル
- IdM での一元設定

コマンドラインインターフェイス (CLI) と IdM Web UI を使用して、IdM クライアントの **sudo 集約ルール** を作成できます。

Generic Security Service Application Programming Interface (GSSAPI) を使用して **sudo** のパスワードレス認証を設定することもできます。これは、UNIX ベースのオペレーティングシステムがネイティブで Kerberos サービスにアクセスして認証する方法です。**pam_sss_gss.so** Pluggable Authentication Module (PAM) を使用して SSSD サービスを介して GSSAPI 認証を呼び出し、有効な Kerberos チケットを使用して **sudo** コマンドに対して認証を行うことができます。

関連情報

- Managing sudo access を参照してください。

8.2. CLI での IDM クライアントの IDM ユーザーへの SUDO アクセス許可

Identity Management (IdM) では、特定の IdM ホストで IdM ユーザーアカウントの特定コマンドに **sudo** アクセスを付与できます。最初に **sudo** コマンドを追加してから、1 つまたは複数のコマンドに対して **sudo** ルールを作成します。

たとえば、**idmclient** マシンで **/usr/sbin/reboot** コマンドを実行する権限を **idm_user** に付与する **idm_user_reboot** の **sudo** ルールを作成するには、以下の手順を実行します。

前提条件

- IdM 管理者としてログインしている。
- IdM で **idm_user** のユーザーアカウントを作成し、ユーザーのパスワードを作成してそのアカウントのロックを解除している。CLI を使用して新しい IdM ユーザーを追加する方法の詳細は、コマンドラインを使用したユーザーの追加 を参照してください。
- **idmclient** ホストにローカル **idm_user** アカウントが存在しない。**idm_user** ユーザーは、ローカルの **/etc/passwd** ファイルには表示されません。

手順

1. IdM の **管理者** として Kerberos チケットを取得します。

```
[root@idmclient ~]# kinit admin
```

2. **sudo** コマンドの IdM データベースに **/usr/sbin/reboot** コマンドを追加します。

```
[root@idmclient ~]# ipa sudocmd-add /usr/sbin/reboot
----------------------------------------
Added Sudo Command "/usr/sbin/reboot"
----------------------------------------
  Sudo Command: /usr/sbin/reboot
```

3. **idm_user_reboot** という名前の **sudo** ルールを作成します。

```
[root@idmclient ~]# ipa sudorule-add idm_user_reboot
--------------------------------
Added Sudo Rule "idm_user_reboot"
--------------------------------
  Rule name: idm_user_reboot
  Enabled: TRUE
```

4. **/usr/sbin/reboot** コマンドを **idm_user_reboot** ルールに追加します。

```
[root@idmclient ~]# ipa sudorule-add-allow-command idm_user_reboot --sudocmds
'/usr/sbin/reboot'
  Rule name: idm_user_reboot
  Enabled: TRUE
  Sudo Allow Commands: /usr/sbin/reboot
-------------------------
Number of members added 1
-------------------------
```

5. **idm_user_reboot** ルールを IdM **idmclient** ホストに適用します。

```
[root@idmclient ~]# ipa sudorule-add-host idm_user_reboot --hosts
idmclient.idm.example.com
Rule name: idm_user_reboot
Enabled: TRUE
Hosts: idmclient.idm.example.com
Sudo Allow Commands: /usr/sbin/reboot
-------------------------
Number of members added 1
-------------------------
```

6. **idm_user** アカウントを **idm_ user_reboot** ルールに追加します。

```
[root@idmclient ~]# ipa sudorule-add-user idm_user_reboot --users idm_user
Rule name: idm_user_reboot
Enabled: TRUE
Users: idm_user
Hosts: idmclient.idm.example.com
Sudo Allow Commands: /usr/sbin/reboot
```

```
-------------------------
Number of members added 1
-------------------------
```

7. 必要に応じて、**idm_user_reboot** ルールの有効性を定義します。

   a. **sudo** ルールが有効である時間を定義するには、**--setattr sudonotbefore=DATE** オプションを指定して **ipa sudorule-mod sudo_rule_name** コマンドを使用します。**DATE** 値は、**yyyymmddHHMMSSZ** 形式に準拠し、明示的に指定される秒数である必要があります。たとえば、**idm_user_reboot** ルールの有効性の開始を 2025 12:34:00 年 12 月 31 に設定するには、次のコマンドを実行します。

   ```
   [root@idmclient ~]# ipa sudorule-mod idm_user_reboot --setattr
   sudonotbefore=20251231123400Z
   ```

   b. sudo ルールが有効な停止時間を定義するには、**--setattr sudonotafter=DATE** オプションを使用します。たとえば、**idm_user_reboot** ルールの有効期間の最後を 2026 12:34:00 年 12 月 31 に設定するには、次のコマンドを実行します。

   ```
   [root@idmclient ~]# ipa sudorule-mod idm_user_reboot --setattr
   sudonotafter=20261231123400Z
   ```

注記

サーバーからクライアントへの変更の伝播には数分かかる場合があります。

検証手順

1. **idmclient** ホストに **idm_user** アカウントとしてログインします。

2. **idm_user** アカウントが実行可能な **sudo** ルールを表示します。

```
[idm_user@idmclient ~]$ sudo -l
Matching Defaults entries for idm_user on idmclient:
    !visiblepw, always_set_home, match_group_by_gid, always_query_group_plugin,
    env_reset, env_keep="COLORS DISPLAY HOSTNAME HISTSIZE KDEDIR
LS_COLORS",
    env_keep+="MAIL PS1 PS2 QTDIR USERNAME LANG LC_ADDRESS LC_CTYPE",
    env_keep+="LC_COLLATE LC_IDENTIFICATION LC_MEASUREMENT
LC_MESSAGES",
    env_keep+="LC_MONETARY LC_NAME LC_NUMERIC LC_PAPER LC_TELEPHONE",
    env_keep+="LC_TIME LC_ALL LANGUAGE LINGUAS _XKB_CHARSET XAUTHORITY
KRB5CCNAME",
    secure_path=/sbin\:/bin\:/usr/sbin\:/usr/bin

User idm_user may run the following commands on idmclient:
    (root) /usr/sbin/reboot
```

3. **sudo** を使用してマシンを再起動します。プロンプトが表示されたら、**idm_user** のパスワードを入力します。

```
[idm_user@idmclient ~]$ sudo /usr/sbin/reboot
[sudo] password for idm_user:
```

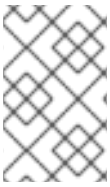
8.3. AD での IDM クライアントの IDM ユーザーへの SUDO アクセス許可

Identity Management (IdM) システム管理者は、IdM ユーザーグループを使用して、アクセス許可、ホストベースのアクセス制御、**sudo** ルール、および IdM ユーザーに対するその他の制御を設定できます。IdM ユーザーグループは、IdM ドメインリソースへのアクセスを許可および制限します。

Active Directory (AD) **ユーザー** と AD **グループ** の両方を IdM ユーザーグループに追加できます。これを実行するには、以下を行います。

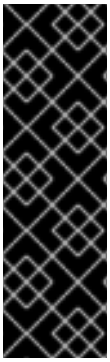
1. AD ユーザーまたはグループを **非 POSIX** 外部 IdM グループに追加します。
2. 非 POSIX 外部 IdM グループを IdM **POSIX** グループに追加します。

その後、POSIX グループの権限を管理することで、AD ユーザーの権限を管理できます。例えば、特定のコマンドの **sudo** アクセスを、特定の IdM ホストの IdM POSIX ユーザーグループに付与できます。



注記

AD ユーザーグループを、IdM 外部グループにメンバーとして追加することもできます。これにより、1 つの AD レルムにユーザーおよびグループの管理を維持することで、Windows ユーザーのポリシーの定義が容易になります。



重要

IdM の SUDO ルールに AD ユーザーの ID オーバーライドを使用 **しない** でください。AD ユーザーの ID オーバーライドは、AD ユーザー自体ではなく、AD ユーザーの POSIX 属性のみを表します。

ID オーバーライドをグループメンバーとして追加できます。ただし、この機能は IdM API で IdM リソースを管理するためにのみ使用できます。グループメンバーとして ID オーバーライドを追加する可能性は POSIX 環境に拡張されていないため、**sudo** またはホストベースのアクセス制御 (HBAC) ルールのメンバーシップには使用できません。

この手順では、**ad_users_reboot sudo** ルールを作成して、**administrator@ad-domain.com** AD ユーザーに、**idmclient** IdM ホストで **/usr/sbin/reboot** コマンドを実行するパーミッションを付与します。これは通常、**root** ユーザー用に予約されています。**administrator@ad-domain.com** は **ad_users_external** 非 POSIX グループのメンバーであり、これは **ad_users** POSIX グループのメンバーでもあります。

前提条件

- IdM **admin** Kerberos の チケット許可チケット (TGT) を取得しました。
- IdM ドメインと **ad-domain.com** AD ドメインの間にフォレスト間の信頼が存在します。
- **idmclient** ホストにローカル **管理者** アカウントが存在しません。**管理者** ユーザーがローカルの **/etc/passwd** ファイルにリストされていません。

手順

1. **administrator@ad-domain** メンバーを持つ **ad_users_external** グループを含む **ad_users** グループを作成します。

a. **オプション**: AD ドメインで対応するグループを作成または選択して、IdM レルムで AD ユーザーを管理するために使用します。複数の AD グループを使用して、それらを IdM 側の異なるグループに追加できます。

b. **ad_users_external** グループを作成し、**--external** オプションを追加して、IdM ドメイン外のメンバーが含まれていることを示します。

```
[root@ipaserver ~]# ipa group-add --desc='AD users external map'
ad_users_external --external
-----------------------------
Added group "ad_users_external"
-----------------------------
  Group name: ad_users_external
  Description: AD users external map
```

注記

ここで指定する外部グループが、Active Directory セキュリティーグループドキュメントで定義されているように、**global** または **universal** グループスコープを持つ AD セキュリティーグループであることを確認してください。たとえば、グループスコープが **domain local** であるため、**Domain users** または **Domain admins** AD セキュリティーグループは使用できません。

c. **ad_users** グループを作成します。

```
[root@ipaserver ~]# ipa group-add --desc='AD users' ad_users
---------------------
Added group "ad_users"
---------------------
  Group name: ad_users
  Description: AD users
  GID: 129600004
```

d. **administrator@ad-domain.com** AD ユーザーを外部メンバーとして **ad_users_external** に追加します。

```
[root@ipaserver ~]# ipa group-add-member ad_users_external --external
"administrator@ad-domain.com"
 [member user]:
 [member group]:
  Group name: ad_users_external
  Description: AD users external map
  External member: S-1-5-21-3655990580-1375374850-1633065477-513
------------------------
Number of members added 1
------------------------
```

AD ユーザーは、**DOMAIN\user_name** または **user_name@DOMAIN** などの完全修飾名で識別される必要があります。次に、AD ID がユーザーの AD SID にマップされます。同じことが AD グループの追加にも当てはまります。

e. **ad_users_external** を **ad_users** にメンバーとして追加します。

```
[root@ipaserver ~]# ipa group-add-member ad_users --groups ad_users_external
```

```
  Group name: ad_users
  Description: AD users
  GID: 129600004
  Member groups: ad_users_external
-------------------------
Number of members added 1
-------------------------
```

2. **ad_users** のメンバーに、**idmclient** ホストで **/usr/sbin/reboot** を実行する権限を付与します。

   a. **sudo** コマンドの IdM データベースに **/usr/sbin/reboot** コマンドを追加します。

```
[root@idmclient ~]# ipa sudocmd-add /usr/sbin/reboot
-----------------------------------
Added Sudo Command "/usr/sbin/reboot"
-----------------------------------
  Sudo Command: /usr/sbin/reboot
```

   b. **ad_users_reboot** という名前の **sudo** ルールを作成します。

```
[root@idmclient ~]# ipa sudorule-add ad_users_reboot
--------------------------------
Added Sudo Rule "ad_users_reboot"
--------------------------------
  Rule name: ad_users_reboot
  Enabled: True
```

   c. **/usr/sbin/reboot** コマンドを **ad_users_reboot** ルールに追加します。

```
[root@idmclient ~]# ipa sudorule-add-allow-command ad_users_reboot --sudocmds
'/usr/sbin/reboot'
  Rule name: ad_users_reboot
  Enabled: True
  Sudo Allow Commands: /usr/sbin/reboot
-------------------------
Number of members added 1
-------------------------
```

   d. **ad_users_reboot** ルールを IdM **idmclient** ホストに適用します。

```
[root@idmclient ~]# ipa sudorule-add-host ad_users_reboot --hosts
idmclient.idm.example.com
Rule name: ad_users_reboot
Enabled: True
Hosts: idmclient.idm.example.com
Sudo Allow Commands: /usr/sbin/reboot
-------------------------
Number of members added 1
-------------------------
```

   e. **ad_users** グループを **ad_users_reboot** ルールに追加します。

```
[root@idmclient ~]# ipa sudorule-add-user ad_users_reboot --groups ad_users
Rule name: ad_users_reboot
```

```
Enabled: TRUE
User Groups: ad_users
Hosts: idmclient.idm.example.com
Sudo Allow Commands: /usr/sbin/reboot
-------------------------
Number of members added 1
-------------------------
```

注記

サーバーからクライアントへの変更の伝播には数分かかる場合があります。

検証手順

1. **ad_users** グループの間接メンバーである **administrator@ad-domain.com** で **idmclient** ホストにログインします。

```
$ ssh administrator@ad-domain.com@ipaclient
Password:
```

2. オプションで、**administrator@ad-domain.com** が実行できる **sudo** コマンドを表示します。

```
[administrator@ad-domain.com@idmclient ~]$ sudo -l
Matching Defaults entries for administrator@ad-domain.com on idmclient:
    !visiblepw, always_set_home, match_group_by_gid, always_query_group_plugin,
    env_reset, env_keep="COLORS DISPLAY HOSTNAME HISTSIZE KDEDIR
LS_COLORS",
    env_keep+="MAIL PS1 PS2 QTDIR USERNAME LANG LC_ADDRESS LC_CTYPE",
    env_keep+="LC_COLLATE LC_IDENTIFICATION LC_MEASUREMENT
LC_MESSAGES",
    env_keep+="LC_MONETARY LC_NAME LC_NUMERIC LC_PAPER LC_TELEPHONE",
    env_keep+="LC_TIME LC_ALL LANGUAGE LINGUAS _XKB_CHARSET XAUTHORITY
KRB5CCNAME",
    secure_path=/sbin\:/bin\:/usr/sbin\:/usr/bin

User administrator@ad-domain.com may run the following commands on idmclient:
    (root) /usr/sbin/reboot
```

3. **sudo** を使用してマシンを再起動します。プロンプトが表示されたら、**administrator@ad-domain.com** のパスワードを入力します。

```
[administrator@ad-domain.com@idmclient ~]$ sudo /usr/sbin/reboot
[sudo] password for administrator@ad-domain.com:
```

関連情報

- Active Directory ユーザーおよび Identity Management グループ
- Include users and groups from a trusted Active Directory domain into SUDO rules

8.4. IDM WEB UI を使用した IDM クライアントでの IDM ユーザーへの SUDO アクセス権の付与

Identity Management (IdM) では、特定の IdM ホストで IdM ユーザーアカウントの特定コマンドに **sudo** アクセスを付与できます。最初に **sudo** コマンドを追加してから、1 つまたは複数のコマンドに対して **sudo** ルールを作成します。

idmclient マシンで **/usr/sbin/reboot** コマンドを実行する権限を **idm_user** に付与する **idm_user_reboot** の sudo ルールを作成するには、以下の手順を実行します。

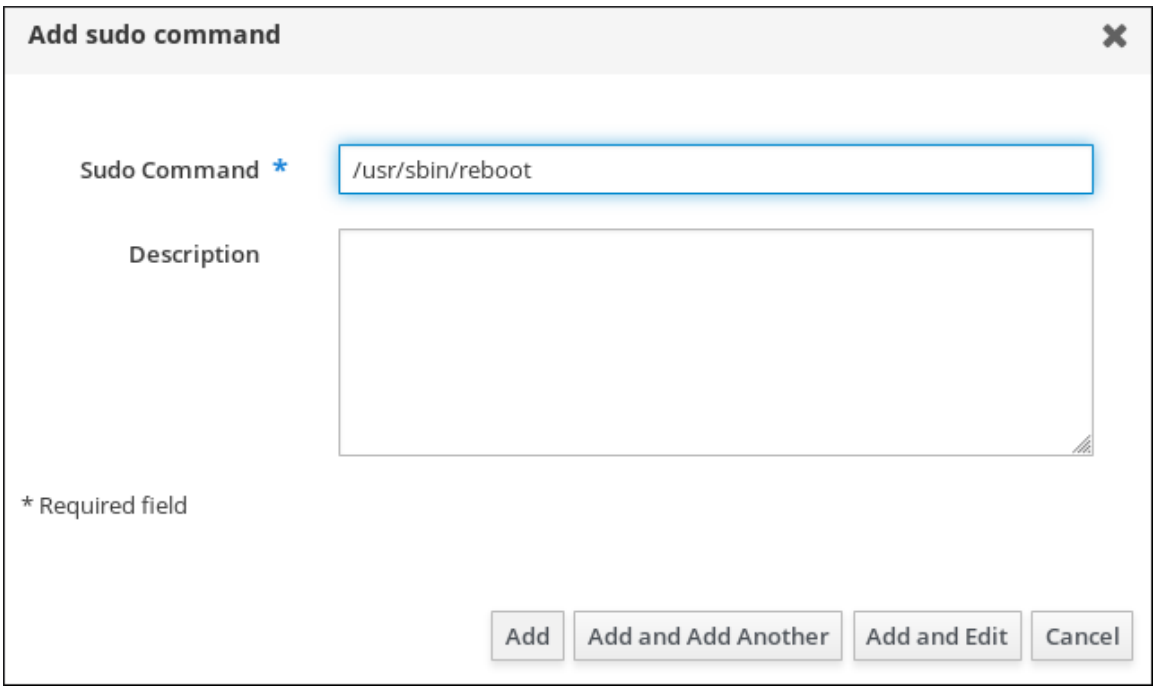
前提条件

- IdM 管理者としてログインしている。
- IdM で **idm_user** のユーザーアカウントを作成し、ユーザーのパスワードを作成してそのアカウントのロックを解除している。コマンドラインインターフェイスを使用して新しい IdM ユーザーを追加する方法の詳細は、コマンドラインを使用したユーザーの追加 を参照してください。
- **idmclient** ホストにローカル **idm_user** アカウントが存在しない。**idm_user** ユーザーは、ローカルの **/etc/passwd** ファイルには表示されません。

手順

1. **sudo** コマンドの IdM データベースに **/usr/sbin/reboot** コマンドを追加します。
   a. **Policy → Sudo → Sudo Commands** の順に移動します。
   b. 右上にある **Add** をクリックして、**Add sudo command** ダイアログボックスを開きます。
   c. **sudo**: **/usr/sbin/reboot** を使用してユーザーが実行できるコマンドを入力します。

   図8.1 IdM sudo コマンドの追加



   d. **Add** をクリックします。
2. 新しい **sudo** コマンドエントリーを使用して sudo ルールを作成し、**idm_user** が **idmclient** マシンを再起動できるようにします。
   a. **Policy → Sudo → Sudo** ルールに移動します。

b. 右上にある Add をクリックして、Add sudo rule ダイアログボックスを開きます。

c. **sudo** ルールの名前を入力します (idm_user_reboot)。

d. Add and Edit をクリックします。

e. ユーザーを指定します。

   i. Who セクションで、Specified Users and Groups のラジオボタンを選択します。

   ii. User category the rule applies to のサブセクションで Add をクリックして、Add users into sudo rule "idm_user_reboot" ダイアログボックスを開きます。

   iii. Add users into sudo rule "idm_user_reboot" ダイアログボックスにある Available 列で、idm_user チェックボックスを選択し、これを Prospective 列に移動します。

   iv. Add をクリックします。

f. ホストを指定します。

   i. Access this host セクションで、Specified Hosts and Groups ラジオボタンを確認します。

   ii. Host category this rule applies to サブセクションで Add をクリックして、Add hosts into sudo rule "idm_user_reboot" ダイアログボックスを開きます。

   iii. Add hosts into sudo rule "idm_user_reboot" ダイアログボックスにある Available 列で、idmclient.idm.example.com チェックボックスを選択し、これを Prospective 列に移動します。

   iv. Add をクリックします。

g. コマンドを指定します。

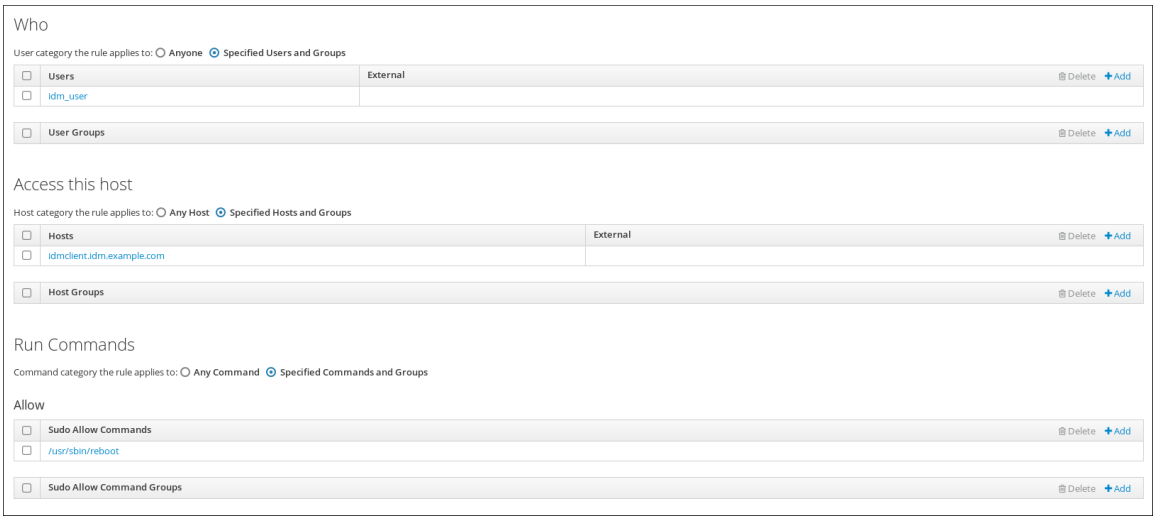
   i. Run Commands セクションの Command category the rule applies to サブセクションで、Specified Commands and Groups ラジオボタンにチェックを入れます。

   ii. Sudo Allow Commands サブセクションで Add をクリックして、Add allow sudo commands into sudo rule "idm_user_reboot" ダイアログボックスを開きます。

   iii. Add allow sudo commands into sudo rule "idm_user_reboot" ダイアログボックスにある Available 列で、**/usr/sbin/reboot** チェックボックスを選択し、これを Prospective 列に移動します。

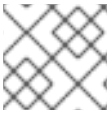
   iv. Add をクリックして、idm_sudo_reboot ページに戻ります。

図8.2 IdM sudo ルールの追加



h. 左上隅にある **Save** をクリックします。

新しいルールはデフォルトで有効になります。



注記

サーバーからクライアントへの変更の伝播には数分かかる場合があります。

検証手順

1. **idmclient** に **idm_user** としてログインします。
2. **sudo** を使用してマシンを再起動します。プロンプトが表示されたら、**idm_user** のパスワードを入力します。

```
$ sudo /usr/sbin/reboot
[sudo] password for idm_user:
```

sudo ルールが正しく設定されている場合には、マシンが再起動します。

8.5. IDM クライアントでサービスアカウントとしてコマンドを実行する CLI での SUDO ルールの作成

IdM では、**RunAs エイリアス** を使用して、**sudo** ルールを設定し、別のユーザーまたはグループとして **sudo** コマンドを実行できます。たとえば、データベースアプリケーションをホストする IdM クライアントが存在し、そのアプリケーションに対応するローカルサービスアカウントとしてコマンドを実行する必要があるとします。

この例を使用して、**run_third-party-app_report** と呼ばれるコマンドラインに **sudo** ルールを作成し、**idm_user** アカウントが **idmclient** ホストの **thirdpartyapp** サービスアカウントとして **/opt/third-party-app/bin/report** コマンドを実行できるようにします。

前提条件

- IdM 管理者としてログインしている。

- IdM で **idm_user** のユーザーアカウントを作成し、ユーザーのパスワードを作成してそのアカウントのロックを解除している。CLI を使用して新しい IdM ユーザーを追加する方法の詳細は、コマンドラインを使用したユーザーの追加 を参照してください。
- **idmclient** ホストにローカル **idm_user** アカウントが存在しない。**idm_user** ユーザーは、ローカルの **/etc/passwd** ファイルには表示されません。
- **idmclient** ホストに、**third-party-app** という名前のカスタムアプリケーションがインストールされている。
- **third-party-app** アプリケーションの **report** コマンドが、**/opt/third-party-app/bin/report** ディレクトリーにインストールされている。
- **third-party-app** アプリケーションにコマンドを実行するために、**thirdpartyapp** という名前のローカルサービスアカウントを作成している。

手順

1. IdM の **管理者** として Kerberos チケットを取得します。

```
[root@idmclient ~]# kinit admin
```

2. **/opt/third-party-app/bin/report** コマンドを、**sudo** コマンドの IdM データベースに追加します。

```
[root@idmclient ~]# ipa sudocmd-add /opt/third-party-app/bin/report
----------------------------------------------------
Added Sudo Command "/opt/third-party-app/bin/report"
----------------------------------------------------
  Sudo Command: /opt/third-party-app/bin/report
```

3. **run_third-party-app_report** という名前の **sudo** ルールを作成します。

```
[root@idmclient ~]# ipa sudorule-add run_third-party-app_report
-------------------------------------------
Added Sudo Rule "run_third-party-app_report"
-------------------------------------------
  Rule name: run_third-party-app_report
  Enabled: TRUE
```

4. **--users=<user>** オプションを使用して、**sudorule-add-runasuser** コマンドに RunAs ユーザーを指定します。

```
[root@idmclient ~]# ipa sudorule-add-runasuser run_third-party-app_report --
users=thirdpartyapp
  Rule name: run_third-party-app_report
  Enabled: TRUE
  RunAs External User: thirdpartyapp
------------------------
Number of members added 1
------------------------
```

ユーザー (または **--groups=*** オプションで指定したグループ) は、ローカルサービスアカウントや Active Directory ユーザーなどの IdM の外部に配置できます。グループ名には **%** 接頭辞を追加しないでください。

5. **/opt/third-party-app/bin/report** コマンドを **run_third-party-app_report** ルールに追加します。

```
[root@idmclient ~]# ipa sudorule-add-allow-command run_third-party-app_report --sudocmds '/opt/third-party-app/bin/report'
Rule name: run_third-party-app_report
Enabled: TRUE
Sudo Allow Commands: /opt/third-party-app/bin/report
RunAs External User: thirdpartyapp
-------------------------
Number of members added 1
-------------------------
```

6. **run_third-party-app_report** ルールを IdM **idmclient** ホストに適用します。

```
[root@idmclient ~]# ipa sudorule-add-host run_third-party-app_report --hosts idmclient.idm.example.com
Rule name: run_third-party-app_report
Enabled: TRUE
Hosts: idmclient.idm.example.com
Sudo Allow Commands: /opt/third-party-app/bin/report
RunAs External User: thirdpartyapp
-------------------------
Number of members added 1
-------------------------
```

7. **idm_user** アカウントーを **run_third-party-app_report** ルールに追加します。

```
[root@idmclient ~]# ipa sudorule-add-user run_third-party-app_report --users idm_user
Rule name: run_third-party-app_report
Enabled: TRUE
Users: idm_user
Hosts: idmclient.idm.example.com
Sudo Allow Commands: /opt/third-party-app/bin/report
RunAs External User: thirdpartyapp
-------------------------
Number of members added 1
```

注記

サーバーからクライアントへの変更の伝播には数分かかる場合があります。

検証手順

1. **idmclient** ホストに **idm_user** アカウントとしてログインします。
2. 新しい sudo ルールをテストします。
   a. **idm_user** アカウントが実行可能な **sudo** ルールを表示します。

```
[idm_user@idmclient ~]$ sudo -l
Matching Defaults entries for idm_user@idm.example.com on idmclient:
    !visiblepw, always_set_home, match_group_by_gid, always_query_group_plugin,
    env_reset, env_keep="COLORS DISPLAY HOSTNAME HISTSIZE KDEDIR
```

```
LS_COLORS",
    env_keep+="MAIL PS1 PS2 QTDIR USERNAME LANG LC_ADDRESS LC_CTYPE",
    env_keep+="LC_COLLATE LC_IDENTIFICATION LC_MEASUREMENT
LC_MESSAGES",
    env_keep+="LC_MONETARY LC_NAME LC_NUMERIC LC_PAPER
LC_TELEPHONE",
    env_keep+="LC_TIME LC_ALL LANGUAGE LINGUAS _XKB_CHARSET
XAUTHORITY KRB5CCNAME",
    secure_path=/sbin\:/bin\:/usr/sbin\:/usr/bin

User idm_user@idm.example.com may run the following commands on idmclient:
    (thirdpartyapp) /opt/third-party-app/bin/report
```

b. **report** コマンドを **thirdpartyapp** サービスアカウントとして実行します。

```
[idm_user@idmclient ~]$ sudo -u thirdpartyapp /opt/third-party-app/bin/report
[sudo] password for idm_user@idm.example.com:
Executing report...
Report successful.
```

8.6. IDM クライアントでサービスアカウントとしてコマンドを実行する IDM WEBUI での SUDO ルールの作成

IdM では、**RunAs エイリアス** を使用して、**sudo** ルールを設定し、別のユーザーまたはグループとして **sudo** コマンドを実行できます。たとえば、データベースアプリケーションをホストする IdM クライアントが存在し、そのアプリケーションに対応するローカルサービスアカウントとしてコマンドを実行する必要があるとします。

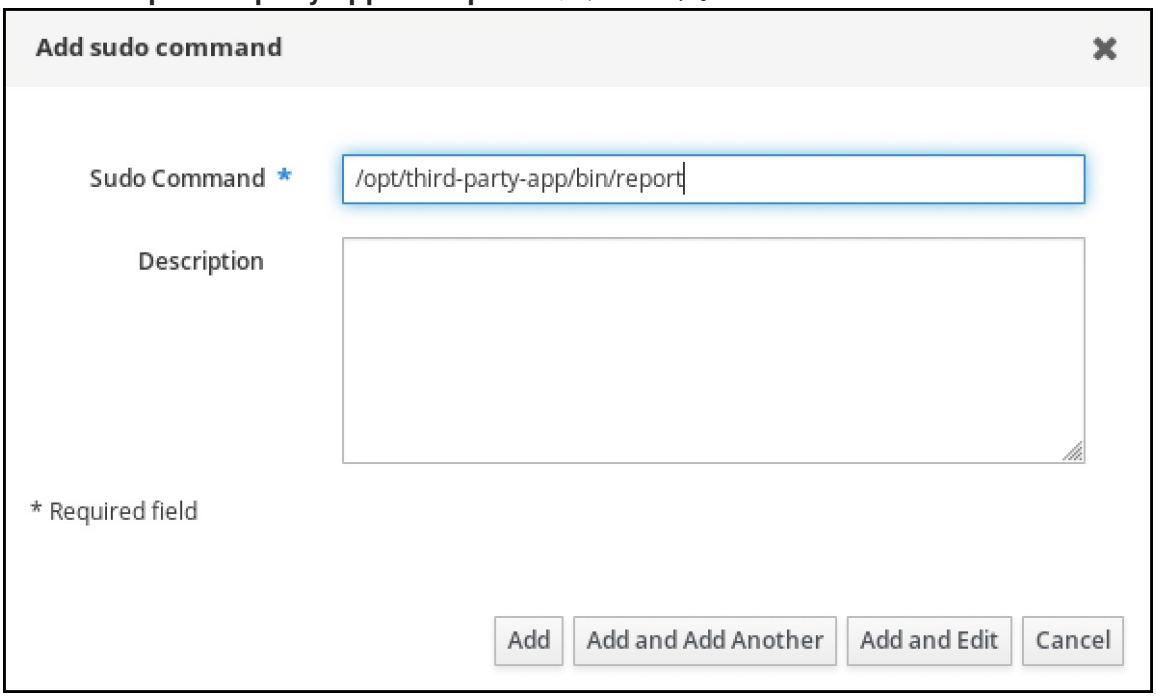
この例を使用して、**run_third-party-app_report** という IdM WebUI に **sudo** ルールを作成し、**idm_user** アカウントが **idmclient** ホストで **thirdpartyapp** サービスアカウントとして **/opt/third-party-app/bin/report** コマンドを実行できるようにします。

前提条件

- IdM 管理者としてログインしている。
- IdM で **idm_user** のユーザーアカウントを作成し、ユーザーのパスワードを作成してそのアカウントのロックを解除している。CLI を使用して新しい IdM ユーザーを追加する方法の詳細は、コマンドラインを使用したユーザーの追加 を参照してください。
- **idmclient** ホストにローカル **idm_user** アカウントが存在しない。**idm_user** ユーザーは、ローカルの **/etc/passwd** ファイルには表示されません。
- **idmclient** ホストに、**third-party-app** という名前のカスタムアプリケーションがインストールされている。
- **third-party-app** アプリケーションの **report** コマンドが、**/opt/third-party-app/bin/report** ディレクトリーにインストールされている。
- **third-party-app** アプリケーションにコマンドを実行するために、**thirdpartyapp** という名前のローカルサービスアカウントを作成している。

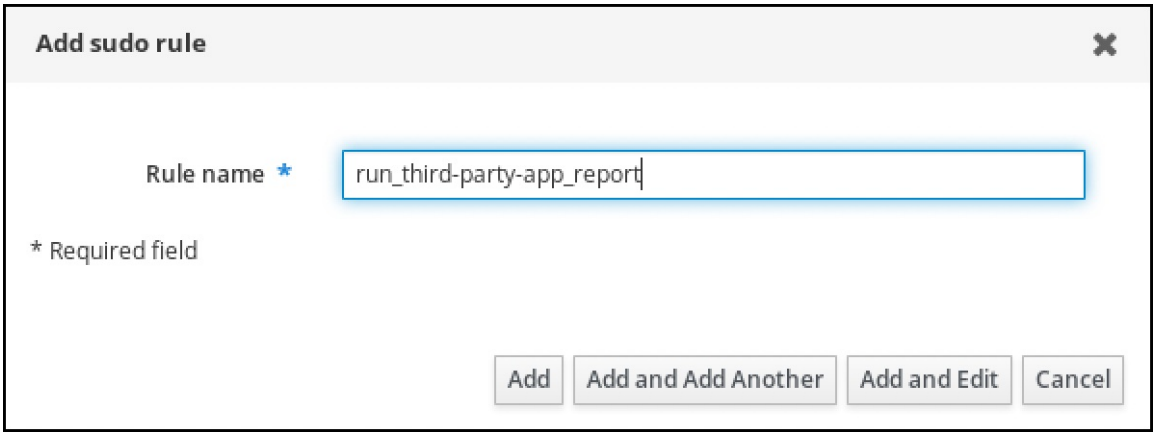
手順

1. **/opt/third-party-app/bin/report** コマンドを、**sudo** コマンドの IdM データベースに追加します。
   a. **Policy → Sudo → Sudo Commands** の順に移動します。
   b. 右上にある **Add** をクリックして、**Add sudo command** ダイアログボックスを開きます。
   c. コマンド **/opt/third-party-app/bin/report** を入力します。



   d. **Add** をクリックします。

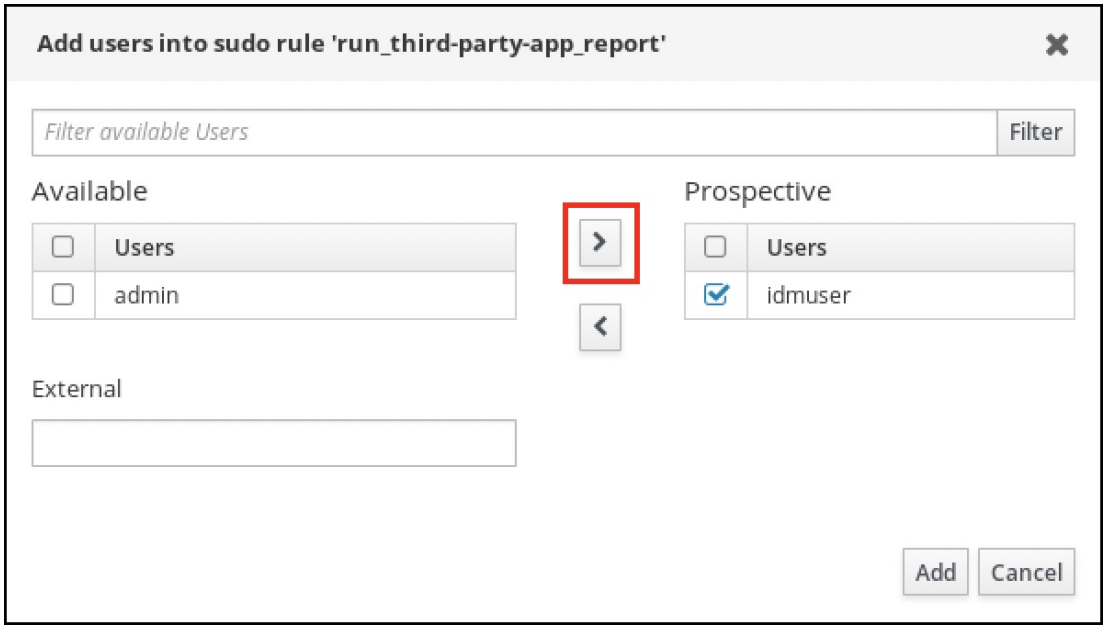
2. 新しい **sudo** コマンドエントリーを使用して、新しい **sudo** ルールを作成します。
   a. **Policy → Sudo → Sudo ルール**に移動します。
   b. 右上にある **Add** をクリックして、**Add sudo rule** ダイアログボックスを開きます。
   c. **sudo** ルールの名前 **run_third-party-app_report** を入力します。



   d. **Add and Edit** をクリックします。
   e. ユーザーを指定します。
      i. **Who** セクションで、**Specified Users and Groups** のラジオボタンを選択します。

ii. User category the rule applies toのサブセクションで Add をクリックして、Add users into sudo rule "run_third-party-app_report" ダイアログボックスを開きます。

iii. Available 列の Add users into sudo rule "run_third-party-app_report" ダイアログボックスで、idm_user チェックボックスをオンにして、これを Prospective 列に移動します。



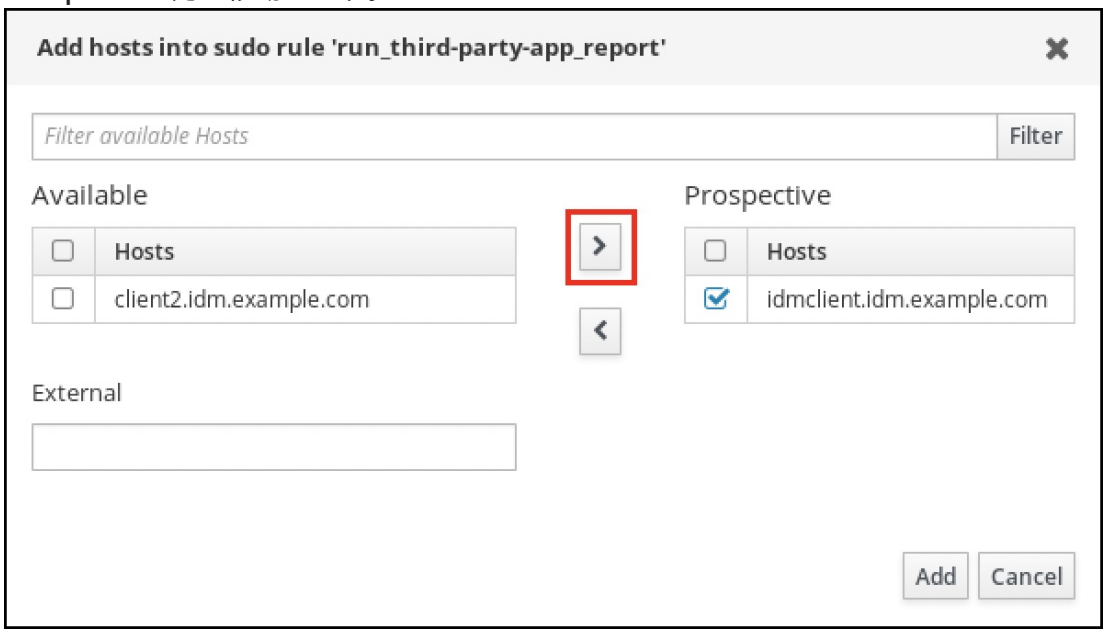
iv. Add をクリックします。

f. ホストを指定します。

i. Access this host セクションで、Specified Hosts and Groups ラジオボタンを確認します。

ii. Host category this rule applies to サブセクションで Add をクリックして、 Add hosts into sudo rule "run_third-party-app_report" ダイアログボックスを開きます。

iii. Available 列の Add hosts into sudo rule "run_third-party-app_report" ダイアログボックスで、idmclient.idm.example.com チェックボックスをオンにして、これを Prospective 列に移動します。



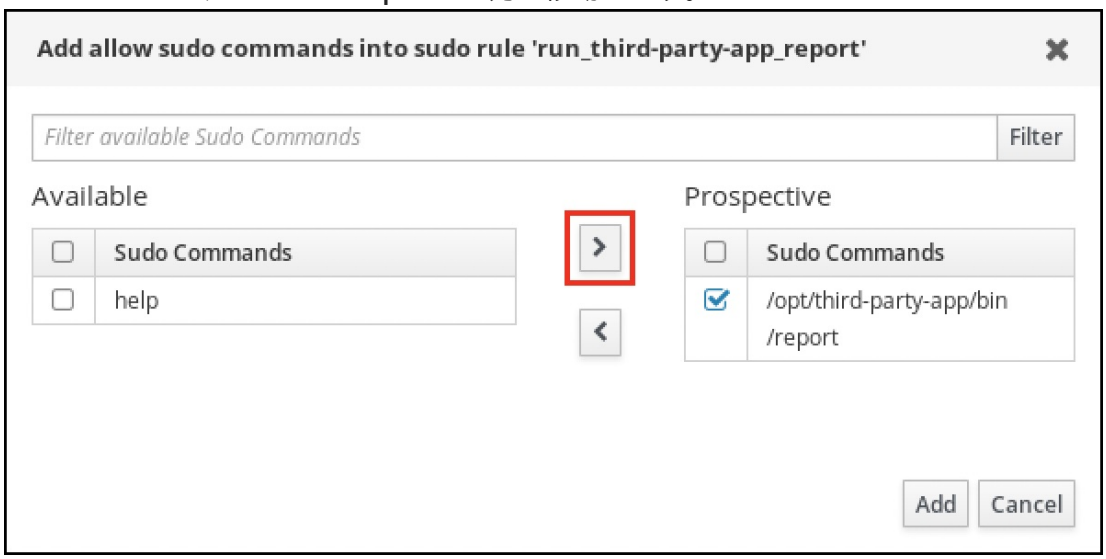
iv. Add をクリックします。

g. コマンドを指定します。

i. Run Commands セクションの Command category the rule applies to サブセクションで、Specified Commands and Groups ラジオボタンにチェックを入れます。

ii. Sudo Allow Commands サブセクションで Add をクリックして、Add allow sudo commands into sudo rule "run_third-party-app_report" ダイアログボックスを開きます。

iii. Available 列の Add allow sudo commands into sudo rule "run_third-party-app_report" ダイアログボックスで、**/opt/third-party-app/bin/report** チェックボックスをオンにして、これを Prospective 列に移動します。



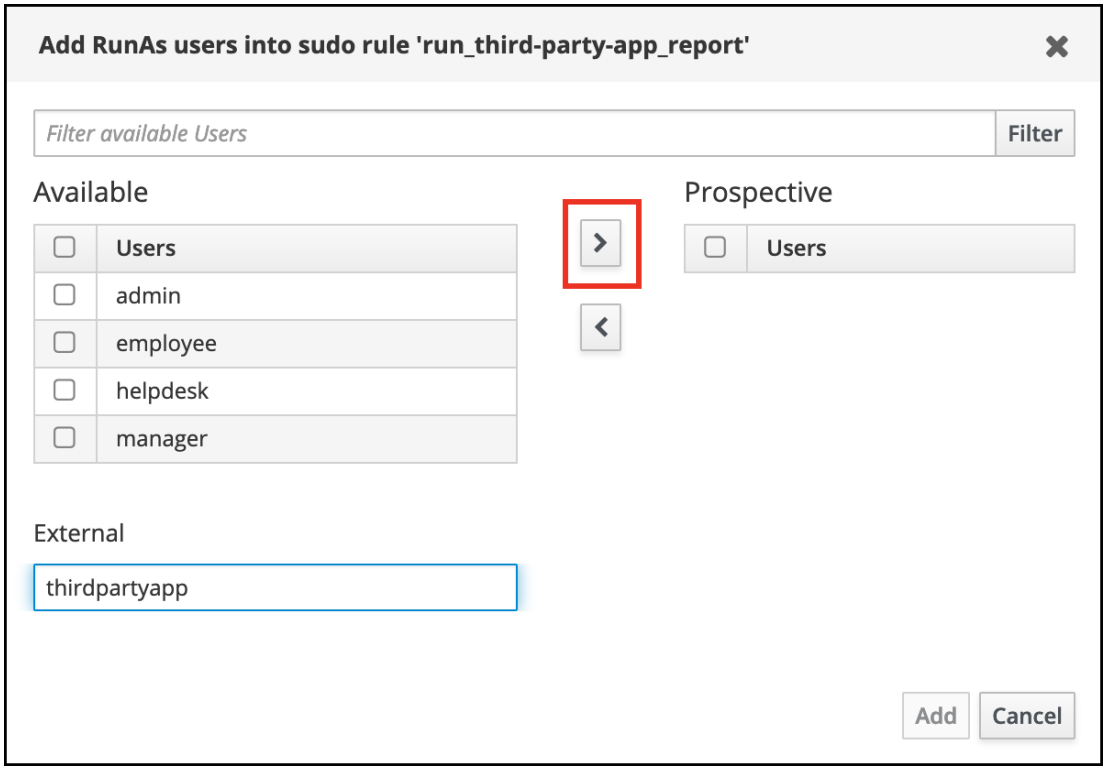
iv. Add をクリックして、run_third-party-app_report のページに戻ります。

h. RunAs ユーザーを指定します。

i. As Whom で、**指定したユーザーとグループ** のラジオボタンを確認します。

ii. RunAs **ユーザー** サブセクションで Add をクリックして、Add RunAs users into sudo rule "run_third-party-app_report" ダイアログボックスを開きます。

iii. Add RunAs users into sudo rule "run_third-party-app_report" ダイアログボックスで、External ボックスに **thirdpartyapp** サービスアカウントを入力し、これを Prospective 列に移動します。

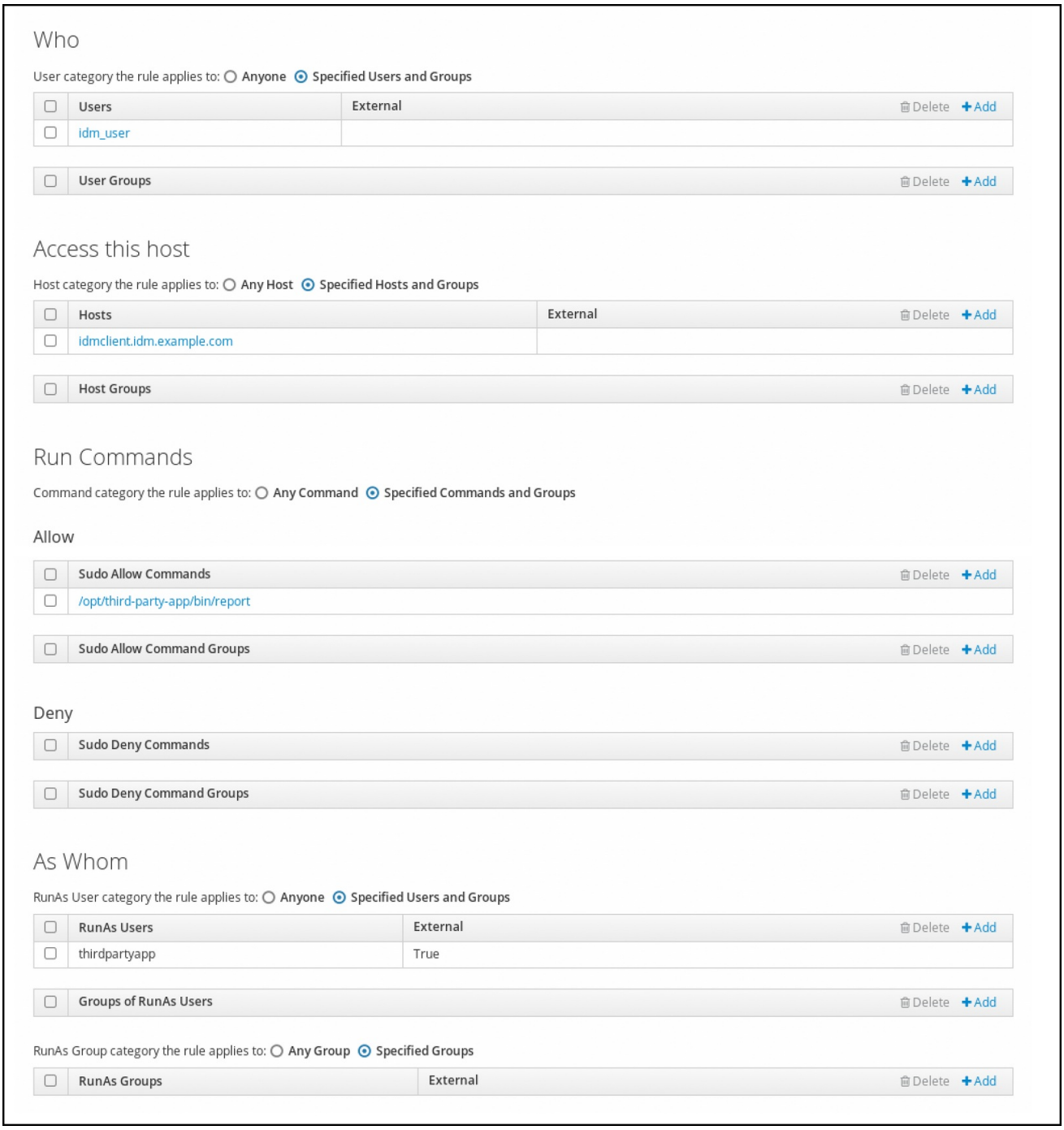


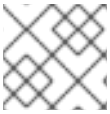
iv. **Add** をクリックして、**run_third-party-app_report** のページに戻ります。

i. 左上隅にある **Save** をクリックします。

新しいルールはデフォルトで有効になります。

図8.3 sudo ルールの詳細





注記

サーバーからクライアントへの変更の伝播には数分かかる場合があります。

検証手順

1. **idmclient** ホストに **idm_user** アカウントとしてログインします。
2. 新しい sudo ルールをテストします。
   a. **idm_user** アカウントが実行可能な **sudo** ルールを表示します。

```
[idm_user@idmclient ~]$ sudo -l
Matching Defaults entries for idm_user@idm.example.com on idmclient:
    !visiblepw, always_set_home, match_group_by_gid, always_query_group_plugin,
```

```
    env_reset, env_keep="COLORS DISPLAY HOSTNAME HISTSIZE KDEDIR
LS_COLORS",
    env_keep+="MAIL PS1 PS2 QTDIR USERNAME LANG LC_ADDRESS LC_CTYPE",
    env_keep+="LC_COLLATE LC_IDENTIFICATION LC_MEASUREMENT
LC_MESSAGES",
    env_keep+="LC_MONETARY LC_NAME LC_NUMERIC LC_PAPER
LC_TELEPHONE",
    env_keep+="LC_TIME LC_ALL LANGUAGE LINGUAS _XKB_CHARSET
XAUTHORITY KRB5CCNAME",
    secure_path=/sbin\:/bin\:/usr/sbin\:/usr/bin

User idm_user@idm.example.com may run the following commands on idmclient:
    (thirdpartyapp) /opt/third-party-app/bin/report
```

b. **report** コマンドを **thirdpartyapp** サービスアカウントとして実行します。

```
[idm_user@idmclient ~]$ sudo -u thirdpartyapp /opt/third-party-app/bin/report
[sudo] password for idm_user@idm.example.com:
Executing report...
Report successful.
```

8.7. IDM クライアントでの SUDO の GSSAPI 認証の有効化

以下の手順では、**pam_sss_gss.so** PAM モジュールを介して、**sudo** コマンドおよび **sudo -i** コマンドの IdM クライアントで、Generic Security Service Application Program Interface (GSSAPI) 認証を有効にする方法を説明します。この設定により、IdM ユーザーは Kerberos チケットを使用して **sudo** コマンドに対する認証が可能になります。

前提条件

- IdM ホストに適用する IdM ユーザーの **sudo** ルールを作成している。この例では、**idmclient** ホストで **/usr/sbin/reboot** コマンドを実行するパーミッションを **idm_user** アカウントに付与する **idm_user_reboot sudo** ルールが作成済みです。
- **/etc/sssd/sssd.conf** ファイルと、**/etc/pam.d/** ディレクトリーの PAM ファイルを変更するための **root** 特権がある。

手順

1. **/etc/sssd/sssd.conf** 設定ファイルを開きます。
2. **[domain/<domain_name>]** セクションに以下のエントリーを追加します。

```
[domain/<domain_name>]
pam_gssapi_services = sudo, sudo-i
```

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

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

5. RHEL 9.2 以降を実行している場合:

a. (オプション) **sssd authselect** プロファイルを選択したかどうかを確認します。

```
# authselect current
Profile ID: sssd
```

出力に、**sssd authselect** プロファイルが選択されていることが示されます。

b. **sssd authselect** プロファイルが選択されている場合は、GSSAPI 認証を有効にします。

```
# authselect enable-feature with-gssapi
```

c. **sssd authselect** プロファイルが選択されていない場合は、それを選択して GSSAPI 認証を有効にします。

```
# authselect select sssd with-gssapi
```

6. RHEL 9.1 以前を実行している場合:

a. **/etc/pam.d/sudo** の PAM 設定ファイルを開きます。

b. 以下のエントリーを、**/etc/pam.d/sudo** ファイルの **auth** セクションの最初の行に追加します。

```
#%PAM-1.0
auth sufficient pam_sss_gss.so
auth       include      system-auth
account    include      system-auth
password   include      system-auth
session    include      system-auth
```

c. **/etc/pam.d/sudo** ファイルを保存して閉じます。

検証手順

1. **idm_user** アカウントとしてホストにログインします。

```
[root@idm-client ~]# ssh -l idm_user@idm.example.com localhost
idm_user@idm.example.com's password:
```

2. **idm_user** アカウントで Ticket-Granting Ticket があることを確認します。

```
[idmuser@idmclient ~]$ klist
Ticket cache: KCM:1366201107
Default principal: idm_user@IDM.EXAMPLE.COM

Valid starting       Expires              Service principal
01/08/2021 09:11:48  01/08/2021 19:11:48
krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
 renew until 01/15/2021 09:11:44
```

3. **(オプション) idm_user** アカウントの Kerberos 認証情報がない場合は、現在の Kerberos 認証情報を削除し、正しい認証情報を要求します。

```
[idm_user@idmclient ~]$ kdestroy -A
```

```
[idm_user@idmclient ~]$ kinit idm_user@IDM.EXAMPLE.COM
Password for idm_user@idm.example.com:
```

4. パスワードを指定せずに **sudo** を使用してマシンを再起動します。

```
[idm_user@idmclient ~]$ sudo /usr/sbin/reboot
```

関連情報

- IdM の用語 一覧の GSSAPI エントリー
- IdM Web UI で IdM クライアントの IdM ユーザーへの sudo アクセスの許可
- CLI での IdM クライアントの IdM ユーザーへの sudo アクセス許可
- **pam_sss_gss(8)** の man ページ
- **sssd.conf (5)** の man ページ

8.8. IDM クライアントでの GSSAPI 認証の有効化および SUDO の KERBEROS 認証インジケーターの有効化

以下の手順では、**pam_sss_gss.so** PAM モジュールを介して、**sudo** コマンドおよび **sudo -i** コマンドの IdM クライアントで、Generic Security Service Application Program Interface (GSSAPI) 認証を有効にする方法を説明します。また、スマートカードを使用してログインしたユーザーのみが Kerberos チケットでこれらのコマンドに対して認証されます。

注記

この手順をテンプレートとして使用し、他の PAM 対応サービスに対して SSSD で GSSAPI 認証を設定して、さらに特定の認証インジケーターが Kerberos チケットにアタッチされているユーザーだけにアクセスを限定することができます。

前提条件

- IdM ホストに適用する IdM ユーザーの **sudo** ルールを作成している。この例では、**idmclient** ホストで **/usr/sbin/reboot** コマンドを実行するパーミッションを **idm_user** アカウントに付与する **idm_user_reboot sudo** ルールが作成済みです。
- **idmclient** ホストにスマートカード認証を設定している。
- **/etc/sssd/sssd.conf** ファイルと、**/etc/pam.d/** ディレクトリーの PAM ファイルを変更するための **root** 特権がある。

手順

1. **/etc/sssd/sssd.conf** 設定ファイルを開きます。
2. **[domain/<domain_name>]** セクションに以下のエントリーを追加します。

```
[domain/<domain_name>]
pam_gssapi_services = sudo, sudo-i
pam_gssapi_indicators_map = sudo:pkinit, sudo-i:pkinit
```

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

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

5. **/etc/pam.d/sudo** の PAM 設定ファイルを開きます。
6. 以下のエントリーを、**/etc/pam.d/sudo** ファイルの **auth** セクションの最初の行に追加します。

```
#%PAM-1.0
auth sufficient pam_sss_gss.so
auth       include      system-auth
account    include      system-auth
password   include      system-auth
session    include      system-auth
```

7. **/etc/pam.d/sudo** ファイルを保存して閉じます。
8. **/etc/pam.d/sudo-i** の PAM 設定ファイルを開きます。
9. 以下のエントリーを、**/etc/pam.d/sudo-i** ファイルの **auth** セクションの最初の行に追加します。

```
#%PAM-1.0
auth sufficient pam_sss_gss.so
auth       include      sudo
account    include      sudo
password   include      sudo
session    optional     pam_keyinit.so force revoke
session    include      sudo
```

10. **/etc/pam.d/sudo-i** ファイルを保存して閉じます。

検証手順

1. **idm_user** アカウントとしてホストにログインし、スマートカードで認証します。

```
[root@idmclient ~]# ssh -l idm_user@idm.example.com localhost
PIN for smart_card
```

2. スマートカードユーザーを使用して Ticket-Granting Ticket があることを確認します。

```
[idm_user@idmclient ~]$ klist
Ticket cache: KEYRING:persistent:1358900015:krb_cache_TObtNMd
Default principal: idm_user@IDM.EXAMPLE.COM

Valid starting       Expires              Service principal
02/15/2021 16:29:48  02/16/2021 02:29:48
krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
 renew until 02/22/2021 16:29:44
```

3. **idm_user** アカウントが実行可能な **sudo** ルールを表示します。

```
[idm_user@idmclient ~]$ sudo -l
Matching Defaults entries for idmuser on idmclient:
    !visiblepw, always_set_home, match_group_by_gid, always_query_group_plugin,
    env_reset, env_keep="COLORS DISPLAY HOSTNAME HISTSIZE KDEDIR
LS_COLORS",
    env_keep+="MAIL PS1 PS2 QTDIR USERNAME LANG LC_ADDRESS LC_CTYPE",
    env_keep+="LC_COLLATE LC_IDENTIFICATION LC_MEASUREMENT
LC_MESSAGES",
    env_keep+="LC_MONETARY LC_NAME LC_NUMERIC LC_PAPER LC_TELEPHONE",
    env_keep+="LC_TIME LC_ALL LANGUAGE LINGUAS _XKB_CHARSET XAUTHORITY
KRB5CCNAME",
    secure_path=/sbin\:/bin\:/usr/sbin\:/usr/bin

User idm_user may run the following commands on idmclient:
    (root) /usr/sbin/reboot
```

4. パスワードを指定せずに **sudo** を使用してマシンを再起動します。

```
[idm_user@idmclient ~]$ sudo /usr/sbin/reboot
```

関連情報

- PAM サービスの GSSAPI 認証を制御する SSSD オプション
- IdM の用語 一覧の GSSAPI エントリー
- スマートカード認証用の Identity Management の設定
- Kerberos 認証インジケーター
- IdM Web UI で IdM クライアントの IdM ユーザーへの sudo アクセスの許可
- CLI での IdM クライアントの IdM ユーザーへの sudo アクセス許可
- **pam_sss_gss(8)** の man ページ
- **sssd.conf (5)** の man ページ

8.9. PAM サービスの GSSAPI 認証を制御する SSSD オプション

/etc/sssd/sssd.conf 設定ファイルに以下のオプションを使用すると、SSSD サービス内の GSSAPI 設定を調整できます。

pam_gssapi_services

SSSD を使用した GSSAPI 認証はデフォルトで無効になっています。このオプションを使用すると、PAM モジュール **pam_sss_gss.so** を使用して GSSAPI 認証を試すことができる PAM サービスをコンマ区切りのリストで指定できます。GSSAPI 認証を明示的に無効にするには、このオプションを **-** に設定します。

pam_gssapi_indicators_map

このオプションは、Identity Management (IdM) ドメインにのみ適用されます。このオプションを使用して、サービスへの PAM のアクセスを付与するのに必要な Kerberos 認証インジケーターをリスト表示します。ペアの形式は **<PAM_service>:_<required_authentication_indicator>_** でなければなりません。
有効な認証インジケーターは以下のとおりです。

- **OTP** - 2 要素認証
- **radius** - RADIUS 認証
- **pkinit** - PKINIT、スマートカード、または証明書での認証
- **hardened** - 強化パスワード

pam_gssapi_check_upn

このオプションはデフォルトで有効となっており、**true** に設定されています。このオプションを有効にすると、SSSD サービスでは Kerberos 認証情報と同じユーザー名が必要になります。**false** の場合には、**pam_sss_gss.so** の PAM モジュールは、必要なサービスチケットを取得できるすべてのユーザーを認証します。

例

次のオプションでは、**sudo** と **sudo-i** サービスの Kerberos 認証を有効にします。この認証では、**sudo** ユーザーはワンタイムパスワードで認証する必要があり、ユーザー名と Kerberos プリンシパルが同じでなければなりません。この設定は **[pam]** セクションにあるため、すべてのドメインに適用されます。

```
[pam]
pam_gssapi_services = sudo, sudo-i
pam_gssapi_indicators_map = sudo:otp
pam_gssapi_check_upn = true
```

これらのオプションを個別の **[domain]** セクションで設定して、**[pam]** セクションのグローバル値を上書きすることもできます。次のオプションは、異なる GSSAPI 設定を各ドメインに適用します。

idm.example.com ドメインの場合

- **sudo** と **sudo -i** サービスの GSSAPI 認証を有効にする。
- **sudo** コマンドには、証明書またはスマートカード認証オーセンティケーターが必要である。
- **sudo -i** コマンドにはワンタイムパスワード認証が必要である。
- ユーザー名と Kerberos プリンシパルを常に一致させる必要がある。

ad.example.com ドメインの場合

- **sudo** サービスに対してのみ GSSAPI 認証を有効にする。
- ユーザー名とプリンシパルを強制的に一致させない。

```
[domain/idm.example.com]
pam_gssapi_services = sudo, sudo-i
pam_gssapi_indicators_map = sudo:pkinit, sudo-i:otp
pam_gssapi_check_upn = true
...

[domain/ad.example.com]
pam_gssapi_services = sudo
pam_gssapi_check_upn = false
...
```

関連情報

- Kerberos 認証インジケーター

8.10. SUDO の GSSAPI 認証のトラブルシューティング

IdM から Kerberos チケットを使用して **sudo** サービスに対する認証できない場合は、以下のシナリオを使用して設定のトラブルシューティングを行います。

前提条件

- **sudo** サービスの GSSAPI 認証が有効化されている。IdM クライアントでの sudo の GSSAPI 認証の有効化 を参照してください。
- **/etc/sssd/sssd.conf** ファイルと、**/etc/pam.d/** ディレクトリーの PAM ファイルを変更するための **root** 特権がある。

手順

- 以下のエラーが表示された場合、Kerberos サービスはホスト名をもとに、サービスチケットに合わせて正しいレルムを解決できない可能性があります。

```
Server not found in Kerberos database
```

このような場合は、**/etc/krb5.conf** の Kerberos 設定ファイルの **[domain_realm]** セクションにホスト名を直接追加します。

```
[idm-user@idm-client ~]$ cat /etc/krb5.conf
...

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

- 以下のエラーが表示される場合には、Kerberos 認証情報がありません。

```
No Kerberos credentials available
```

このような場合は、**kinit** ユーティリティーを使用して Kerberos 認証情報を取得するか、SSSD で認証します。

```
[idm-user@idm-client ~]$ kinit idm-user@IDM.EXAMPLE.COM
Password for idm-user@idm.example.com:
```

- **/var/log/sssd/sssd_pam.log** ログファイルに以下のエラーのいずれかが表示される場合には、Kerberos 認証情報と、現在ログインしたユーザーのユーザー名とが一致しません。

```
User with UPN [<UPN>] was not found.

UPN [<UPN>] does not match target user [<username>].
```

このような場合は、SSSD で認証されたことを確認するか、**/etc/sssd/sssd.conf** ファイルで **pam_gssapi_check_upn** オプションを無効にすることを検討してください。

```
[idm-user@idm-client ~]$ cat /etc/sssd/sssd.conf
...

pam_gssapi_check_upn = false
```

- 他のトラブルシューティングを行う場合は、PAM モジュール **pam_sss_gss.so** のデバッグ出力を有効してください。
  - **/etc/pam.d/sudo** や **/etc/pam.d/sudo-i** など、PAM ファイルに **pam_sss_gss.so** の全エントリーの最後に **debug** オプションを追加します。

```
[root@idm-client ~]# cat /etc/pam.d/sudo
#%PAM-1.0
auth       sufficient   pam_sss_gss.so   debug
auth       include      system-auth
account    include      system-auth
password   include      system-auth
session    include      system-auth
```

```
[root@idm-client ~]# cat /etc/pam.d/sudo-i
#%PAM-1.0
auth       sufficient   pam_sss_gss.so   debug
auth       include      sudo
account    include      sudo
password   include      sudo
session    optional     pam_keyinit.so force revoke
session    include      sudo
```

  - **pam_sss_gss.so** モジュールで認証を試み、コンソールの出力を確認します。この例では、ユーザーには Kerberos 認証情報がありません。

```
[idm-user@idm-client ~]$ sudo ls -l /etc/sssd/sssd.conf
pam_sss_gss: Initializing GSSAPI authentication with SSSD
pam_sss_gss: Switching euid from 0 to 1366201107
pam_sss_gss: Trying to establish security context
pam_sss_gss: SSSD User name: idm-user@idm.example.com
pam_sss_gss: User domain: idm.example.com
pam_sss_gss: User principal:
pam_sss_gss: Target name: host@idm.example.com
pam_sss_gss: Using ccache: KCM:
pam_sss_gss: Acquiring credentials, principal name will be derived
pam_sss_gss: Unable to read credentials from [KCM:] [maj:0xd0000, min:0x96c73ac3]
pam_sss_gss: GSSAPI: Unspecified GSS failure.  Minor code may provide more
information
pam_sss_gss: GSSAPI: No credentials cache found
pam_sss_gss: Switching euid from 1366200907 to 0
pam_sss_gss: System error [5]: Input/output error
```

8.11. ANSIBLE PLAYBOOK を使用して IDM クライアントでの IDM ユーザーの SUDO アクセスを確認する

Identity Management (IdM) では、特定の IdM ホストの IdM ユーザーアカウントに **sudo** アクセスが付与されるようにできます。

この手順では、**idm_user_reboot** という名前の **sudo** ルールが存在するように設定します。このルールは、**idmclient** マシンで **/usr/sbin/reboot** コマンドを実行するパーミッションを **idm_user** に付与します。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
  - この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。
- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。
- IdM に **idm_user** のユーザーアカウントが存在することを確認し、そのユーザーのパスワードを作成してアカウントのロックを解除している。コマンドラインインターフェイスを使用して新しい IdM ユーザーを追加する方法の詳細は、コマンドラインを使用したユーザーの追加 を参照してください。
- **idmclient** にローカルの **idm_user** アカウントがない。(**idm_user** ユーザーは **idmclient** の **/etc/passwd** ファイルに表示されていない)。

手順

1. **inventory.file** などのインベントリーファイルを作成し、そこに **ipaservers** を定義します。

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

2. **sudo** コマンドを 1 つまたは複数追加します。
   a. **ensure-reboot-sudocmd-is-present.yml** Ansible Playbook を作成し、**sudo** コマンドの IdM データベースに **/usr/sbin/reboot** コマンドが存在するようにします。この手順は、**/usr/share/doc/ansible-freeipa/playbooks/sudocmd/ensure-sudocmd-is-present.yml** ファイルのサンプルをコピーして変更し、簡素化できます。

```
---
- name: Playbook to manage sudo command
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  # Ensure sudo command is present
  - ipasudocmd:
```

```
    ipaadmin_password: "{{ ipaadmin_password }}"
    name: /usr/sbin/reboot
    state: present
```

b. Playbook を実行します。

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

3. コマンドを参照する **sudo** ルールを作成します。

a. **sudo** コマンドエントリーを使用して sudo ルールが存在することを確認する **ensure-sudorule-for-idmuser-on-idmclient-is-present.yml** Ansible Playbook を作成します。sudo ルールは、 **idm_user** が **idmclient** マシンを再起動することを許可します。この手順は、**/usr/share/doc/ansible-freeipa/playbooks/sudorule/ensure-sudorule-is-present.yml** ファイルのサンプルをコピーして変更し、簡素化できます。

```
---
- name: Tests
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  # Ensure a sudorule is present granting idm_user the permission to run /usr/sbin/reboot
on idmclient
  - ipasudorule:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: idm_user_reboot
      description: A test sudo rule.
      allow_sudocmd: /usr/sbin/reboot
      host: idmclient.idm.example.com
      user: idm_user
      state: present
```

b. Playbook を実行します。

```
$ ansible-playbook -v -i path_to_inventory_directory/inventory.file
path_to_playbooks_directory/ensure-sudorule-for-idmuser-on-idmclient-is-
present.yml
```

検証手順

idm_user が **sudo** を使用して **idmclient** を再起動できることを確認し、IdM サーバーに存在するように設定した **sudo** ルールが **idmclient** で機能することをテストします。サーバーに加えられた変更がクライアントで反映されるまで数分かかる場合があります。

1. **idmclient** に **idm_user** としてログインします。

2. **sudo** を使用してマシンを再起動します。プロンプトが表示されたら、**idm_user** のパスワードを入力します。

```
$ sudo /usr/sbin/reboot
[sudo] password for idm_user:
```

sudo が正しく設定されている場合には、マシンが再起動します。

関連情報

- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-sudocmd.md** ファイル、**README-sudocmdgroup.md** ファイル、および **README-sudorule.md** ファイルを参照してください。

第9章 LDAPMODIFY を使用した IDM ユーザーの外部管理

IdM 管理者は、**ipa** コマンドを使用してディレクトリーの内容を管理できます。または、**ldapmodify** コマンドを使用して同様に管理することもできます。このコマンドを対話的に使用して、すべてのデータをコマンドラインで直接指定できます。ファイル内のデータを LDAP データ交換形式 (LDIF) で **ldapmodify** コマンドに提供することもできます。

9.1. IDM ユーザーアカウントを外部で管理するためのテンプレート

次のテンプレートは、IdM でのさまざまなユーザー管理操作に使用できます。これらのテンプレートでは、以下の目的を達成するために **ldapmodify** を使用して変更する必要のある属性がどれであるかがわかります。

- 新規ステージユーザーの追加
- ユーザーの属性変更
- ユーザーの有効化
- ユーザーの無効化
- ユーザーの保存

テンプレートは LDAP データ交換形式 (LDIF) でフォーマットされます。LDIF は、LDAP ディレクトリーのコンテンツおよび更新リクエストを表す標準的なプレーンテキストデータ交換形式です。

テンプレートを使用し、プロビジョニングシステムの LDAP プロバイダーを設定して、IdM ユーザーアカウントを管理できます。

詳細な手順例は、以下のセクションを参照してください。

- LDIF ファイルで定義されている IdM ステージユーザーの追加
- ldapmodify を使用した CLI からの IdM ステージユーザーの直接追加
- ldapmodify での IdM ユーザーの保存

新規ステージユーザーを追加するためのテンプレート

- **UID および GID が自動的に割り当てられた** ユーザーを追加するためのテンプレート。作成したエントリーの識別名 (DN) は **uid=user_login** で開始する必要があります。

```
dn: uid=user_login,cn=staged
users,cn=accounts,cn=provisioning,dc=idm,dc=example,dc=com
changetype: add
objectClass: top
objectClass: inetorgperson
uid: user_login
sn: surname
givenName: first_name
cn: full_name
```

- **UID と GID が静的に割り当てられている** ユーザーを追加するためのテンプレート

```
dn: uid=user_login,cn=staged
```

```
users,cn=accounts,cn=provisioning,dc=idm,dc=example,dc=com
changetype: add
objectClass: top
objectClass: person
objectClass: inetorgperson
objectClass: organizationalperson
objectClass: posixaccount
uid: user_login
uidNumber: UID_number
gidNumber: GID_number
sn: surname
givenName: first_name
cn: full_name
homeDirectory: /home/user_login
```

ステージユーザーの追加時に IdM オブジェクトクラスを指定する必要はありません。IdM は、ユーザーのアクティベート後にこれらのクラスを自動的に追加します。

既存ユーザーを変更するためのテンプレート

- **ユーザーの属性の変更**:

```
dn: distinguished_name
changetype: modify
replace: attribute_to_modify
attribute_to_modify: new_value
```

- **ユーザーの無効化**:

```
dn: distinguished_name
changetype: modify
replace: nsAccountLock
nsAccountLock: TRUE
```

- **ユーザーの有効化**

```
dn: distinguished_name
changetype: modify
replace: nsAccountLock
nsAccountLock: FALSE
```

nssAccountLock 属性を更新してもステージユーザーおよび保存済みユーザーには影響はありません。更新操作が正常に完了しても、属性値は **nssAccountLock: TRUE** のままになります。

- **ユーザーの保持**

```
dn: distinguished_name
changetype: modrdn
newrdn: uid=user_login
deleteoldrdn: 0
newsuperior: cn=deleted users,cn=accounts,cn=provisioning,dc=idm,dc=example,dc=com
```

注記

ユーザーの変更前に、ユーザーのログインを検索してユーザーの識別名 (DN) を取得します。以下の例では、**user_allowed_to_modify_user_entries** ユーザーは、**activator** または IdM 管理者など、ユーザーおよびグループの情報の変更を許可されたユーザーです。以下の例のパスワードは、このユーザーのパスワードです。

```
[...]
# ldapsearch -LLL -x -D
"uid=user_allowed_to_modify_user_entries,cn=users,cn=accounts,dc=idm,dc=example,dc=com" -w "Secret123" -H ldap://r8server.idm.example.com -b
"cn=users,cn=accounts,dc=idm,dc=example,dc=com" uid=test_user
dn: uid=test_user,cn=users,cn=accounts,dc=idm,dc=example,dc=com
memberOf: cn=ipausers,cn=groups,cn=accounts,dc=idm,dc=example,dc=com
```

9.2. IDM グループアカウントを外部で管理するためのテンプレート

次のテンプレートは、IdM でのさまざまなユーザーグループ管理操作に使用できます。これらのテンプレートでは、以下の目的を達成するために **ldapmodify** を使用して変更する必要のある属性がどれであるかがわかります。

- 新規グループの作成
- 既存グループの削除
- グループへのメンバーの追加
- グループからメンバーの削除

テンプレートは LDAP データ交換形式 (LDIF) でフォーマットされます。LDIF は、LDAP ディレクトリーのコンテンツおよび更新リクエストを表す標準的なプレーンテキストデータ交換形式です。

テンプレートを使用して、プロビジョニングシステムの LDAP プロバイダーを設定して、IdM グループアカウントを管理できます。

新規グループの作成

```
dn: cn=group_name,cn=groups,cn=accounts,dc=idm,dc=example,dc=com
changetype: add
objectClass: top
objectClass: ipaobject
objectClass: ipausergroup
objectClass: groupofnames
objectClass: nestedgroup
objectClass: posixgroup
uid: group_name
cn: group_name
gidNumber: GID_number
```

グループの変更

- **既存グループの削除**

```
dn: group_distinguished_name
changetype: delete
```

- **グループへのメンバーの追加**

```
dn: group_distinguished_name
changetype: modify
add: member
member: uid=user_login,cn=users,cn=accounts,dc=idm,dc=example,dc=com
```

ステージまたは保存済みユーザーをグループに追加しないでください。更新操作が正常に完了しても、ユーザーはグループのメンバーとしては更新されません。アクティブなユーザーのみがグループに所属できます。

- **グループからのメンバーの削除**

```
dn: distinguished_name
changetype: modify
delete: member
member: uid=user_login,cn=users,cn=accounts,dc=idm,dc=example,dc=com
```

注記

グループの変更前に、グループ名で検索してグループの識別名 (DN) を取得します。

```
# ldapsearch -YGSSAPI -H ldap://server.idm.example.com -b
"cn=groups,cn=accounts,dc=idm,dc=example,dc=com" "cn=group_name"
dn: cn=group_name,cn=groups,cn=accounts,dc=idm,dc=example,dc=com
ipaNTSecurityIdentifier: S-1-5-21-1650388524-2605035987-2578146103-11017
cn: testgroup
objectClass: top
objectClass: groupofnames
objectClass: nestedgroup
objectClass: ipausergroup
objectClass: ipaobject
objectClass: posixgroup
objectClass: ipantgroupattrs
ipaUniqueID: 569bf864-9d45-11ea-bea3-525400f6f085
gidNumber: 1997010017
```

9.3. LDAPMODIFY コマンドの対話的な使用

対話モードで LDAP (Lightweight Directory Access Protocol) エントリーを変更できます。

手順

1. コマンド行で、**ldapmodify** コマンドの後に LDAP Data Interchange Format (LDIF) ステートメントを入力します。

例9.1testuser の電話番号の変更

```
# ldapmodify -Y GSSAPI -H ldap://server.example.com
dn: uid=testuser,cn=users,cn=accounts,dc=example,dc=com
```

```
changetype: modify
replace: telephoneNumber
telephonenumber: 88888888
```

-Y オプションを使用するには、Kerberos チケットを取得する必要があることに注意してください。

2. **Ctlr+D** を押してインタラクティブモードを終了します。
3. または、**ldapmodify** コマンドの後に LDIF ファイルを指定します。

例9.2 ldapmodify コマンドは、LDIF ファイルから変更データを読み取ります。

```
# ldapmodify -Y GSSAPI -H ldap://server.example.com -f ~/example.ldif
```

関連情報

- **ldapmodify** コマンドの使用方法に関する詳細は、**ldapmodify(1)** の man ページを参照してください。
- **LDIF** 構造の詳細は、**ldif(5)** man ページを参照してください。

9.4. LDAPMODIFY での IDM ユーザーの保存

ldapmodify を使用して IdM ユーザーを保存する (従業員が退職した後にユーザーアカウントを非アクティブ化する) には、次の手順に従います。

前提条件

- ユーザーを保存するロールが割り当てられた IdM ユーザーとして認証できる。

手順

1. ユーザーを保存するロールを持つ IdM ユーザーとしてログインします。

```
$ kinit admin
```

2. **ldapmodify** コマンドを入力し、Generic Security Services API (GSSAPI) を認証に使用する Simple Authentication and Security Layer (SASL) メカニズムとして指定します。

```
# ldapmodify -Y GSSAPI
SASL/GSSAPI authentication started
SASL username: admin@IDM.EXAMPLE.COM
SASL SSF: 256
SASL data security layer installed.
```

3. 保存するユーザーの **dn** を入力します。

```
dn: uid=user1,cn=users,cn=accounts,dc=idm,dc=example,dc=com
```

4. 実行する変更のタイプとして **modrdn** を入力します。

```
changetype: modrdn
```

5. ユーザーの **newrdn** を指定します。

```
newrdn: uid=user1
```

6. 以下のようにユーザーの保存を指定します。

```
deleteoldrdn: 0
```

7. **新しい上位 DN** を指定します。

```
newsuperior: cn=deleted users,cn=accounts,cn=provisioning,dc=idm,dc=example,dc=com
```

ユーザーを保存すると、そのエントリーをディレクトリー情報ツリー (DIT) 内の新しい場所に移動します。上記の理由から、新しい親エントリーの DN を新しい上位 DN として指定する必要があります。

8. **Enter** を再度押して、これがエントリーの最後であることを確認します。

```
[Enter]

modifying rdn of entry "uid=user1,cn=users,cn=accounts,dc=idm,dc=example,dc=com"
```

9. **Ctrl** + **C** を使用して接続を終了します。

検証手順

- 保存済みユーザーをリスト表示して、ユーザーが保存されていることを確認します。

```
$ ipa user-find --preserved=true
--------------
1 user matched
--------------
  User login: user1
  First name: First 1
  Last name: Last 1
  Home directory: /home/user1
  Login shell: /bin/sh
  Principal name: user1@IDM.EXAMPLE.COM
  Principal alias: user1@IDM.EXAMPLE.COM
  Email address: user1@idm.example.com
  UID: 1997010003
  GID: 1997010003
  Account disabled: True
  Preserved user: True
----------------------------
Number of entries returned 1
----------------------------
```

第10章 LDAPSEARCH コマンドを使用した IDM エントリーの検索

ipa find コマンドを使用して、アイデンティティー管理エントリーを検索できます。**ipa** コマンドの詳細は、IPA コマンドの構造 セクションを参照してください。

このセクションでは、**ldapsearch** コマンドラインコマンドを使用し、アイデンティティー管理エントリー経由で検索する代替オプションの基本を紹介します。

10.1. LDAPSEARCH コマンドの使用

ldapsearch コマンドの形式は次のとおりです。

```
# ldapsearch [-x | -Y mechanism] [options] [search_filter] [list_of_attributes]
```

- 認証方法を設定するには、**-x** オプションを指定して簡易バインドを使用するか、**-Y** オプションを指定して Simple Authentication and Security Layer (SASL) メカニズムを設定します。**-Y GSSAPI** オプションを使用している場合は、Kerberos チケットを取得する必要があることに注意してください。
- **オプション** は、以下の表で説明されている **ldapsearch** コマンドオプションです。
- **search_filter** は LDAP 検索フィルターです。
- **list_of_attributes** は、検索結果が返す属性のリストです。

たとえば、ベース LDAP ツリーのすべてのエントリーでユーザー名 **user01** を検索するとします。

```
# ldapsearch -x -H ldap://ldap.example.com -s sub "(uid=user01)"
```

- **-x** オプションは、簡易バインドで認証するように **ldapsearch** コマンドに指示します。**-D** オプションで識別名 (DN) を指定しない場合、認証は匿名になることに注意してください。
- **-H** オプションは、**ldap://ldap.example.com** に接続します。
- **-s sub** オプションは、**ldapsearch** コマンドに、ベース DN から始まるすべてのエントリーを検索して、名前が **user01** のユーザーを検索するように指示します。**"(uid=user01)"** はフィルターです。

-b オプションを使用して検索の開始点を指定しない場合、コマンドはデフォルトのツリーを検索することに注意してください。これは、**etc/openldap/ldap.conf** ファイルの BASE パラメーターで指定されます。

表10.1 ldapsearch コマンドのオプション

オプション	説明
-b	検索の開始点。コマンドラインがコードに解釈できるアスタリスク (*) またはその他の文字が検索パラメーターに含まれている場合は、値を一重引用符または二重引用符で囲む必要があります。たとえば、**-b cn=user,ou=Product Development,dc=example,dc=com** です。

オプション	説明
-D	認証に使用する識別名 (DN)。
-H	サーバーに接続するための LDAP URL。**-H** オプションは、**-h** および **-p** オプションを置き換えます。
-l	検索要求が完了するまで待機する制限時間 (秒単位)。
-s **scope**	検索の範囲です。スコープには、次のいずれかを選択できます。 ● **base** は、**-b** オプションからのエントリー、または **LDAP_BASEDN** 環境変数によって定義されたエントリーのみを検索します。 ● **one** は、**-b** オプションからエントリーの子のみ検索します。 ● **sub** は、-b オプションの開始点からサブツリーを検索します。
-W	パスワードを要求します。
-x	単純なバインドを許可するためにデフォルトの SASL 接続を無効にします。
-Y **SASL_mechanism**	認証用の SASL メカニズムを設定します。
-z **number**	検索結果の最大エントリー数。

ldapsearch コマンドの **-x** または **-Y** オプションを使用して、いずれかの認証メカニズムを指定する必要があることに注意してください。

関連情報

- **ldapsearch** の使用方法について、詳しくは **ldapsearch (1)** の man ページを参照してください。

10.2. LDAPSEARCH フィルターの使用

ldapsearch フィルターを使用すると、検索結果を絞り込むことができます。

たとえば、共通名が **example** に設定されたすべてのエントリーが検索結果に含まれるようにするには、次のようにします。

```
"(cn=example)"
```

この場合、**等号 (=)** は Operator であり、 **example** は値です。

表10.2 ldapsearch フィルター Operator

検索タイプ	演算子	説明
等号	=	値と完全に一致するエントリーを返します。(例: **cn=example**)。
部分文字列	**=string* string**	substring が一致するすべてのエントリーを返します。(例: **cn=exa*l**)。アスタリスク (*) はゼロ (0) 以上の文字を示します。
以上	>=	値以上の属性を持つすべてのエントリーを返します。(例: **uidNumber >= 5000**)。
より小か等しい	<=	値以下の属性を持つすべてのエントリーを返します。(例: **uidNumber <= 5000**)。
存在	=*	1 つ以上の属性を持つすべてのエントリーを返します。(例: **cn=***)。
概算値	~=	値属性に類似するすべてのエントリーを返します。たとえば、**l~=san fransico** は **l=san francisco** を返すことができます。

ブール 演算子を使用して、**ldapsearch** コマンドに複数のフィルターを組み合わせることができます。

表10.3 ldapsearch フィルターのブール演算子

検索タイプ	演算子	説明
AND	&	フィルター内のすべてのステートメントが true のエントリーをすべて返します。例: **(&(filter)(filter)(filter)...)**
OR	|	フィルター内の少なくとも 1 つのステートメントが true のエントリーをすべて返します。例: **(|(filter)(filter)(filter)...)**
NOT	!	フィルター内のステートメントが true でないエントリーをすべて返します。例: **(!(filter))**

第11章 ユーザーの外部プロビジョニングのための IDM 設定

システム管理者は、Identity Management (IdM) が、ID 管理用の外部ソリューションでユーザーのプロビジョニングをサポートするように設定できます。

ipa ユーティリティーを使用する代わりに、外部プロビジョニングシステムの管理者は **ldapmodify** ユーティリティーを使用して IdM LDAP にアクセスできます。管理者は、ldapmodify を使用して CLI から、または LDIF ファイル を使用して、個々のステージユーザーを追加できます。

IdM 管理者が外部プロビジョニングシステムを完全に信頼して、検証済みのユーザーだけを追加することを前提とします。ただし、新たにアクティブユーザーを直接追加できるように、外部プロビジョニングシステムの管理者に **User Administrator** の IdM ロールを割り当てなくても構いません。

外部プロビジョニングシステムで作成されたステージユーザーを自動的にアクティブユーザーに移動するように スクリプトを設定 できます。

本章には、以下の項が含まれます。

1. Identity Management (IdM) の準備 して外部プロビジョニングシステムを使用してステージユーザーを IdM に追加する。
2. 外部プロビジョニングシステムが追加したユーザーをステージユーザーからアクティブユーザーに移動する スクリプトを作成 する。
3. 外部プロビジョニングシステムを使用して IdM ステージユーザーを追加する。これには 2 つの方法があります。
   - LDIF ファイルを使用した IdM ステージユーザーの追加
   - ldapmodify を使用した CLI からの IdM ステージユーザーの直接追加

11.1. ステージユーザーアカウントの自動アクティベーションに向けた IDM アカウントの準備

以下の手順では、外部プロビジョニングシステムが使用する 2 つの IdM ユーザーアカウントを設定する方法を説明します。適切なパスワードポリシーが指定されたグループにアカウントを追加すると、外部プロビジョニングシステムが IdM でユーザーのプロビジョニングを管理できるようになります。以下では、ステージユーザーの追加用に外部システムが使用するユーザーアカウントには **provisionator** という名前が付けられます。ステージユーザーを自動アクティベートする時に使用されるユーザーアカウントは **activator** という名前です。

前提条件

- 以下の手順を実行するホストが IdM に登録されている。

手順

1. IdM 管理者としてログインします。

```
$ kinit admin
```

2. ステージユーザーを追加する特権指定して **provisionator** という名前のユーザーを作成します。
   a. provisionator ユーザーアカウントを追加します。

```
$ ipa user-add provisionator --first=provisioning --last=account --password
```

a. provisionator ユーザーに必要な特権を割り当てます。

i. ステージユーザーの追加を管理する **System Provisioning** というカスタムロールを作成します。

```
$ ipa role-add --desc "Responsible for provisioning stage users" "System
Provisioning"
```

ii. **Stage User Provisioning** の特権をロールに追加します。この特権により、ステージユーザーを追加できます。

```
$ ipa role-add-privilege "System Provisioning" --privileges="Stage User Provisioning"
```

iii. provisionator ユーザーをロールに追加します。

```
$ ipa role-add-member --users=provisionator "System Provisioning"
```

iv. provisionator が IdM に存在することを確認します。

```
$ ipa user-find provisionator --all --raw
--------------
1 user matched
--------------
  dn: uid=provisionator,cn=users,cn=accounts,dc=idm,dc=example,dc=com
  uid: provisionator
  [...]
```

3. ユーザーアカウントを管理する特権を指定して **activator** ユーザーを作成します。

a. activator ユーザーアカウントを追加します。

```
$ ipa user-add activator --first=activation --last=account --password
```

b. デフォルトの **User Administrator** ロールにユーザーを追加して、activator ユーザーに必要な特権を付与します。

```
$ ipa role-add-member --users=activator "User Administrator"
```

4. アプリケーションアカウントのユーザーグループを作成します。

```
$ ipa group-add application-accounts
```

5. グループのパスワードポリシーを更新します。以下のポリシーは、アカウントのパスワードの有効期限やロックアウトを防ぎますが、複雑なパスワードを必要とすることでリスクの可能性を低減します。

```
$ ipa pwpolicy-add application-accounts --maxlife=10000 --minlife=0 --history=0 --
minclasses=4 --minlength=8 --priority=1 --maxfail=0 --failinterval=1 --lockouttime=0
```

6. (必要に応じて) パスワードポリシーが IdM に存在することを確認します。

```
$ ipa pwpolicy-show application-accounts
  Group: application-accounts
  Max lifetime (days): 10000
  Min lifetime (hours): 0
  History size: 0
[...]
```

7. アプリケーションアカウントのグループにプロビジョニングアカウントおよびアクティベーションアカウントを追加します。

```
$ ipa group-add-member application-accounts --users={provisionator,activator}
```

8. ユーザーアカウントのパスワードを変更します。

```
$ kpasswd provisionator
$ kpasswd activator
```

新しい IdM ユーザーのパスワードはすぐに失効するため、パスワードの変更が必要になります。

関連情報:

- Managing user accounts using the command line を参照してください。
- Delegating Permissions over Users を参照してください。
- IdM パスワードポリシーの定義 を参照してください。

11.2. IDM ステージユーザーアカウントの自動アクティベーションの設定

この手順では、ステージユーザーをアクティベートするスクリプトを作成する方法を説明します。システムは、指定した間隔でスクリプトを自動的に実行します。これにより、新規ユーザーアカウントが自動的にアクティベートされ、作成直後に使用できます。

重要

以下の手順では、外部プロビジョニングシステムの所有者がユーザーをすでに検証しており、スクリプトが IdM への追加前に IdM 側で追加の検証を必要としていないことを前提としています。

IdM サーバーの 1 つでのみアクティベーションプロセスを有効にするだけで十分です。

前提条件

- **provisionator** アカウントおよび **activator** アカウントが IdM に存在している。詳細は ステージユーザーアカウントの自動アクティベーション用の IdM アカウントの準備 を参照してください。
- この手順を実行する IdM サーバーで root 権限がある。
- IdM 管理者としてログインしている。
- 外部プロビジョニングシステムを信頼している。

手順

1. アカウントのアクティベーション用に keytab ファイルを生成します。

```
# ipa-getkeytab -s server.idm.example.com -p "activator" -k /etc/krb5.ipa-activation.keytab
```

複数の IdM サーバーでアクティベーションプロセスを有効にする場合は、1 つのサーバーだけで keytab ファイルを生成します。次に、keytab ファイルを他のサーバーにコピーします。

2. 以下の内容を含む **/usr/local/sbin/ipa-activate-all** スクリプトを作成して全ユーザーをアクティベートします。

```
#!/bin/bash

kinit -k -i activator

ipa stageuser-find --all --raw | grep "  uid:" | cut -d ":" -f 2 | while read uid; do ipa stageuser-activate ${uid}; done
```

3. **ipa-activate-all** スクリプトのパーミッションおよび所有権を編集して、実行可能なファイルに変更します。

```
# chmod 755 /usr/local/sbin/ipa-activate-all
# chown root:root /usr/local/sbin/ipa-activate-all
```

4. systemd ユニットファイル **/etc/systemd/system/ipa-activate-all.service** を作成して、以下の内容を追加します。

```
[Unit]
Description=Scan IdM every minute for any stage users that must be activated

[Service]
Environment=KRB5_CLIENT_KTNAME=/etc/krb5.ipa-activation.keytab
Environment=KRB5CCNAME=FILE:/tmp/krb5cc_ipa-activate-all
ExecStart=/usr/local/sbin/ipa-activate-all
```

5. systemd タイマー **/etc/systemd/system/ipa-activate-all.timer** を作成して、以下の内容を追加します。

```
[Unit]
Description=Scan IdM every minute for any stage users that must be activated

[Timer]
OnBootSec=15min
OnUnitActiveSec=1min

[Install]
WantedBy=multi-user.target
```

6. 新しい設定を再読み込みします。

```
# systemctl daemon-reload
```

7. **ipa-activate-all.timer** を有効にします。

```
# systemctl enable ipa-activate-all.timer
```

8. **ipa-activate-all.timer** を起動します。

```
# systemctl start ipa-activate-all.timer
```

9. (必要に応じて) **ipa-activate-all.timer** デーモンが実行していることを確認します。

```
# systemctl status ipa-activate-all.timer
● ipa-activate-all.timer - Scan IdM every minute for any stage users that must be activated
   Loaded: loaded (/etc/systemd/system/ipa-activate-all.timer; enabled; vendor preset:
disabled)
   Active: active (waiting) since Wed 2020-06-10 16:34:55 CEST; 15s ago
  Trigger: Wed 2020-06-10 16:35:55 CEST; 44s left

Jun 10 16:34:55 server.idm.example.com systemd[1]: Started Scan IdM every minute for any
stage users that must be activated.
```

11.3. LDIF ファイルで定義されている IDM ステージユーザーの追加

IdM LDAP にアクセスし、LDIF ファイルを使用してステージユーザーを追加するには、次の手順に従います。以下の例では、ユーザーを 1 つ追加していますが、一括モードで 1 つのファイルに複数のユーザーを追加できます。

前提条件

- IdM 管理者が、**provisionator** アカウントとパスワードを作成している。詳細は ステージユーザーアカウントの自動アクティベーション用の IdM アカウントの準備 を参照してください。
- 外部管理者が **provisionator** アカウントのパスワードを知っている。
- LDAP サーバーから IdM サーバーに SSH 接続できる。
- 以下のような、ユーザーのライフサイクルを正しく処理できるように IdM ステージユーザーに割り当てる必要のある最小限の属性セットを提供できる。
  - **識別名** (dn)
  - **共通名** (cn)
  - **名前 (姓)** (sn)
  - **uid**

手順

1. 外部サーバーで、新規ユーザーに関する情報が含まれる LDIF ファイルを作成します。

```
dn: uid=stageidmuser,cn=staged
users,cn=accounts,cn=provisioning,dc=idm,dc=example,dc=com
changetype: add
objectClass: top
objectClass: inetorgperson
uid: stageidmuser
```

```
sn: surname
givenName: first_name
cn: full_name
```

2. 外部サーバーから IdM サーバーへの LDIF ファイルを転送します。

```
$ scp add-stageidmuser.ldif provisionator@server.idm.example.com:/provisionator/
Password:
add-stageidmuser.ldif                                         100%  364
217.6KB/s   00:00
```

3. **SSH** プロトコルを使用して、**provisionator** として IdM サーバーに接続します。

```
$ ssh provisionator@server.idm.example.com
Password:
[provisionator@server ~]$
```

4. IdM サーバーで、provisionator アカウントの Kerberos ticket-granting ticket (TGT) を取得します。

```
[provisionator@server ~]$ kinit provisionator
```

5. -f オプションと LDIF ファイルの名前を指定して **ldapadd** コマンドを入力します。IdM サーバー名とポート番号を指定します。

```
~]$ ldapadd -h server.idm.example.com -p 389 -f  add-stageidmuser.ldif
SASL/GSSAPI authentication started
SASL username: provisionator@IDM.EXAMPLE.COM
SASL SSF: 256
SASL data security layer installed.
adding the entry "uid=stageidmuser,cn=staged
users,cn=accounts,cn=provisioning,dc=idm,dc=example,dc=com"
```

11.4. LDAPMODIFY を使用した CLI からの IDM ステージユーザーの直接追加

Identity Management (IdM) LDAP にアクセスし、**ldapmodify** ユーティリティーを使用してステージユーザーを追加するには、次の手順に従います。

前提条件

- IdM 管理者が、**provisionator** アカウントおよびパスワードを作成している。詳細は ステージユーザーアカウントの自動アクティベーション用の IdM アカウントの準備 を参照してください。
- 外部管理者が **provisionator** アカウントのパスワードを知っている。
- LDAP サーバーから IdM サーバーに SSH 接続できる。
- 以下のような、ユーザーのライフサイクルを正しく処理できるように IdM ステージユーザーに割り当てる必要のある最小限の属性セットを提供できる。
  - **識別名** (dn)

- **共通名** (cn)
- **名前 (姓)** (sn)
- **uid**

手順

1. IdM の ID および認証情報を使用して、**SSH** プロトコルを使用して IdM サーバーに接続します。

```
$ ssh provisionator@server.idm.example.com
Password:
[provisionator@server ~]$
```

2. 新規ステージユーザーを追加するロールを持つ IdM ユーザーである **provisionator** アカウントの TGT を取得します。

```
$ kinit provisionator
```

3. **ldapmodify** コマンドを入力し、Generic Security Services API (GSSAPI) を認証に使用する Simple Authentication and Security Layer (SASL) メカニズムとして指定します。IdM サーバーとポート名を指定します。

```
# ldapmodify -h server.idm.example.com -p 389 -Y GSSAPI
SASL/GSSAPI authentication started
SASL username: provisionator@IDM.EXAMPLE.COM
SASL SSF: 56
SASL data security layer installed.
```

4. 追加するユーザーの **dn** を入力します。

```
dn: uid=stageuser,cn=staged
users,cn=accounts,cn=provisioning,dc=idm,dc=example,dc=com
```

5. 実行する変更の種類として **add** を入力します。

```
changetype: add
```

6. ユーザーのライフサイクルを正しく処理できるようにするために必要な LDAP オブジェクトクラスのカテゴリーを指定します。

```
objectClass: top
objectClass: inetorgperson
```

追加のオブジェクトクラスを指定できます。

7. ユーザーの **uid** を入力します。

```
uid: stageuser
```

8. ユーザーの **cn** を入力します。

```
cn: Babs Jensen
```

9. ユーザーの名前 (姓) を入力します。

```
sn: Jensen
```

10. **Enter** を再度押して、これがエントリーの最後であることを確認します。

```
[Enter]

adding new entry "uid=stageuser,cn=staged
users,cn=accounts,cn=provisioning,dc=idm,dc=example,dc=com"
```

11. **Ctrl** + **C** を使用して接続を終了します。

検証手順

ステージエントリーの内容を検証し、プロビジョニングシステムにより、必要な全 POSIX 属性が追加され、ステージエントリーのアクティベートの準備ができていることを確認します。

- 新規ステージユーザーの LDAP 属性を表示するには、**ipa stageuser-show --all --raw** コマンドを実行します。

```
$ ipa stageuser-show stageuser --all --raw
  dn: uid=stageuser,cn=staged
users,cn=accounts,cn=provisioning,dc=idm,dc=example,dc=com
  uid: stageuser
  sn: Jensen
  cn: Babs Jensen
  has_password: FALSE
  has_keytab: FALSE
  nsaccountlock: TRUE
  objectClass: top
  objectClass: inetorgperson
  objectClass: organizationalPerson
  objectClass: person
```

1. ユーザーは、**nsaccountlock** 属性を使用して明示的に無効化されている点に注意してください。

11.5. 関連情報

- ldapmodify を使用した IdM ユーザーの外部管理 を参照してください。

第12章 ユーザー、ホスト、およびサービス用の KERBEROS プリンシパルエイリアスの管理

新しいユーザー、ホスト、またはサービスを作成すると、以下の形式で Kerberos プリンシパルが自動的に追加されます。

- **user_name@REALM**
- host/**host_name@REALM**
- **service_name/host_name@REALM**

管理者は、エイリアスを使用して、ユーザー、ホスト、またはサービスが Kerberos アプリケーションに対して認証できるようにすることができます。これは、次のような状況で役立ちます。

- ユーザー名の変更後に、ユーザーが以前のユーザー名と新しいユーザー名の両方でログインする必要がある。
- IdM Kerberos レルムがメールドメインと異なる場合でも、ユーザーはメールアドレスを使用してログインする必要がある。

ユーザーの名前を変更すると、オブジェクトはエイリアスと以前の正規プリンシパル名を保持することに注意してください。

12.1. KERBEROS プリンシパルエイリアスの追加

Identity Management (IdM) 環境では、エイリアス名を既存の Kerberos プリンシパルに関連付けることができます。これにより、セキュリティーが強化され、IdM ドメイン内の認証プロセスが簡素化されます。

手順

- エイリアス名 **useralias** をアカウント **user** に追加するには、次のように入力します。

```
# ipa user-add-principal <user> <useralias>
--------------------------------
Added new aliases to user "user"
--------------------------------
        User login: user
  Principal alias: user@IDM.EXAMPLE.COM, useralias@IDM.EXAMPLE.COM
```

ホストまたはサービスにエイリアスを追加するには、代わりにそれぞれ **ipa host-add-principal** または **ipa service-add-principal** コマンドを使用します。

エイリアス名を使用して認証する場合は、**kinit** コマンドで **-C** オプションを使用します。

```
# kinit -C <useralias>
Password for <user>@IDM.EXAMPLE.COM:
```

12.2. KERBEROS プリンシパルエイリアスの削除

Identity Management (IdM) 環境で Kerberos プリンシパルに関連付けられているエイリアス名を削除できます。

手順

- アカウント **user** からエイリアス **useralias** を削除するには、次のように入力します。

```
# ipa user-remove-principal <user> <useralias>
--------------------------------
Removed aliases from user "user"
--------------------------------
  User login: user
  Principal alias: user@IDM.EXAMPLE.COM
```

ホストまたはサービスからエイリアスを削除するには、代わりにそれぞれ **ipa host-remove-principal** または **ipa service-remove-principal** コマンドを使用します。

正規のプリンシパル名は削除できないことに注意してください。

```
# ipa user-show <user>
  User login: user
  ...
  Principal name: user@IDM.EXAMPLE.COM
  ...

# ipa user-remove-principal user user
ipa: ERROR: invalid 'krbprincipalname': at least one value equal to the canonical principal name must be present
```

12.3. KERBEROS エンタープライズプリンシパルエイリアスの追加

Identity Management (IdM) 環境では、エンタープライズプリンシパルエイリアス名を既存の Kerberos エンタープライズプリンシパルに関連付けることができます。エンタープライズプリンシパルエイリアスは、ユーザープリンシパル名 (UPN) 接尾辞、NetBIOS 名、または信頼された Active Directory フォレストドメインのドメイン名以外の任意のドメイン接尾辞を使用できます。

注記

エンタープライズプリンシパルエイリアスを追加または削除する場合は、2 つのバックスラッシュ (\\) を使用して @ 記号をエスケープします。エスケープしないと、シェルが @ 記号を Kerberos レルム名の一部として解釈し、次のエラーが発生します。

```
ipa: ERROR: The realm for the principal does not match the realm for this IPA server
```

手順

- エンタープライズプリンシパルエイリアス **user@example.com** を **user** アカウントに追加するには、以下を実行します。

```
# ipa user-add-principal <user> <user\\@example.com>
--------------------------------
Added new aliases to user "user"
--------------------------------
        User login: user
  Principal alias: user@IDM.EXAMPLE.COM, user\@example.com@IDM.EXAMPLE.COM
```

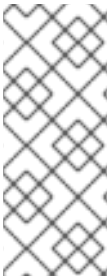
ホストまたはサービスにエンタープライズエイリアスを追加するには、代わりにそれぞれ **ipa host-add-principal** または **ipa service-add-principal** コマンドを使用します。

エンタープライズプリンシパル名を使用して認証する場合は、**kinit** コマンドで **-E** オプションを使用します。

```
# kinit -E <user@example.com>
Password for user\@example.com@IDM.EXAMPLE.COM:
```

12.4. KERBEROS エンタープライズプリンシパルエイリアスの削除

Identity Management (IdM) 環境で Kerberos エンタープライズプリンシパルに関連付けられているエンタープライズプリンシパルエイリアス名を削除できます。



注記

エンタープライズプリンシパルエイリアスを追加または削除する場合は、2 つのバックスラッシュ (\\) を使用して @ 記号をエスケープします。エスケープしないと、シェルが @ 記号を Kerberos レルム名の一部として解釈し、次のエラーが発生します。

```
ipa: ERROR: The realm for the principal does not match the realm for this IPA server
```

手順

- エンタープライズプリンシパルエイリアス **user@example.com** をアカウント **user** から削除するには、次のように入力します。

```
# ipa user-remove-principal <user> <user\\@example.com>
--------------------------------
Removed aliases from user "user"
--------------------------------
  User login: user
  Principal alias: user@IDM.EXAMPLE.COM
```

ホストまたはサービスからエイリアスを削除するには、代わりにそれぞれ **ipa host-remove-principal** または **ipa service-remove-principal** コマンドを使用します。

第13章 PAC 情報による KERBEROS セキュリティーの強化

RHEL 8.5 以降の Identity Management (IdM) では、特権属性証明書 (PAC) 情報をデフォルトで使用できます。また、RHEL 8.5 より前にインストールされた IdM デプロイメントで、セキュリティー識別子 (SID) を有効にすることもできます。

13.1. IDM での特権属性証明書 (PAC) の使用

セキュリティーを強化するために、RHEL Identity Management (IdM) は、新しいデプロイメントでデフォルトで特権属性証明書 (PAC) 情報を含む Kerberos チケットを発行するようになりました。PAC には、セキュリティー識別子 (SID)、グループメンバーシップ、ホームディレクトリー情報など、Kerberos プリンシパルに関する豊富な情報が含まれています。

Microsoft Active Directory (AD) がデフォルトで使用する SID は、再利用されることのないグローバルに一意の識別子です。SID は複数の名前空間を表します。各ドメインには、各オブジェクトの SID の接頭辞である SID があります。

RHEL 8.5 以降、IdM サーバーまたはレプリカをインストールすると、インストールスクリプトはデフォルトでユーザーおよびグループの SID を生成します。これにより、IdM が PAC データを操作できるようになります。RHEL 8.5 より前に IdM をインストールし、AD ドメインとの信頼を設定していない場合は、IdM オブジェクトの SID が生成されていない可能性があります。IdM オブジェクトの SID の生成に関する詳細は、IdM でのセキュリティー識別子 (SID) の有効化 を参照してください。

Kerberos チケットの PAC 情報を評価することで、リソースアクセスをより詳細に制御できます。たとえば、あるドメインの管理者アカウントは、他のドメインの管理者アカウントとは一意に異なる SID を持っています。AD ドメインへの信頼がある IdM 環境では、UID が 0 のすべての Linux **root** アカウントなど、さまざまな場所で繰り返される可能性のある単純なユーザー名や UID ではなく、グローバルに一意の SID に基づいてアクセス制御を設定できます。

13.2. IDM でのセキュリティー識別子 (SID) の有効化

RHEL 8.5 より前に IdM をインストールし、AD ドメインとの信頼を設定していない場合は、IdM オブジェクトのセキュリティー識別子 (SID) が生成されていない可能性があります。これは、以前は SID を生成する場合、**ipa-adtrust-install** コマンドを実行して **信頼コントローラー** ロールを IdM サーバーに追加することが唯一の方法だったためです。

RHEL 8.6 以降、IdM の Kerberos では、IdM オブジェクトに SID が必要です。これは、特権属性証明書 (PAC) 情報に基づくセキュリティーに必要です。

前提条件

- RHEL 8.5 より前に IdM をインストールしている。
- Active Directory ドメインとの信頼の設定の一部である **ipa-sidgen** タスクを実行していない。
- IdM 管理者アカウントとして認証可能である。

手順

- SID の使用を有効にし、**SIDgen** タスクをトリガーして、既存のユーザーとグループの SID を生成します。このタスクはリソースを大量に消費する可能性があります。

```
[root@server ~]# ipa config-mod --enable-sid --add-sids
```

検証

- IdM **admin** ユーザーアカウントエントリーに、**-500** で終わる SID (ドメイン管理者用に予約されている SID) のある **ipantsecurityidentifier** 属性があることを確認します。

```
[root@server ~]# ipa user-show admin --all | grep ipantsecurityidentifier
  ipantsecurityidentifier: S-1-5-21-2633809701-976279387-419745629-500
```

関連情報

- IdM での特権属性証明書 (PAC) の使用
- How to solve users unable to authenticate to IPA/IDM with PAC issues - S4U2PROXY_EVIDENCE_TKT_WITHOUT_PAC error KCS ソリューション
- 信頼コントローラーおよび信頼エージェント
- Integrate SID configuration into base IPA installers

第14章 KERBEROS チケットポリシーの管理

Identity Management (IdM) の Kerberos チケットポリシーは、Kerberos チケットアクセス、期間、および更新に対する制限を設定します。IdM サーバーで実行している Key Distribution Center (KDC) の Kerberos チケットポリシーを設定できます。

Kerberos チケットポリシーを管理する場合、次の概念や操作を実行します。

- IdM KDC のロール
- IdM Kerberos チケットポリシータイプ
- Kerberos 認証インジケーター
- IdM サービスの認証インジケーターの有効化
- グローバルチケットライフサイクルポリシーの設定
- 認証インジケーターごとのグローバルチケットポリシーの設定
- ユーザーのデフォルトチケットポリシーの設定
- ユーザーの個別認証インジケーターチケットポリシーの設定
- **krbtpolicy-mod** コマンドの認証インジケーターオプション

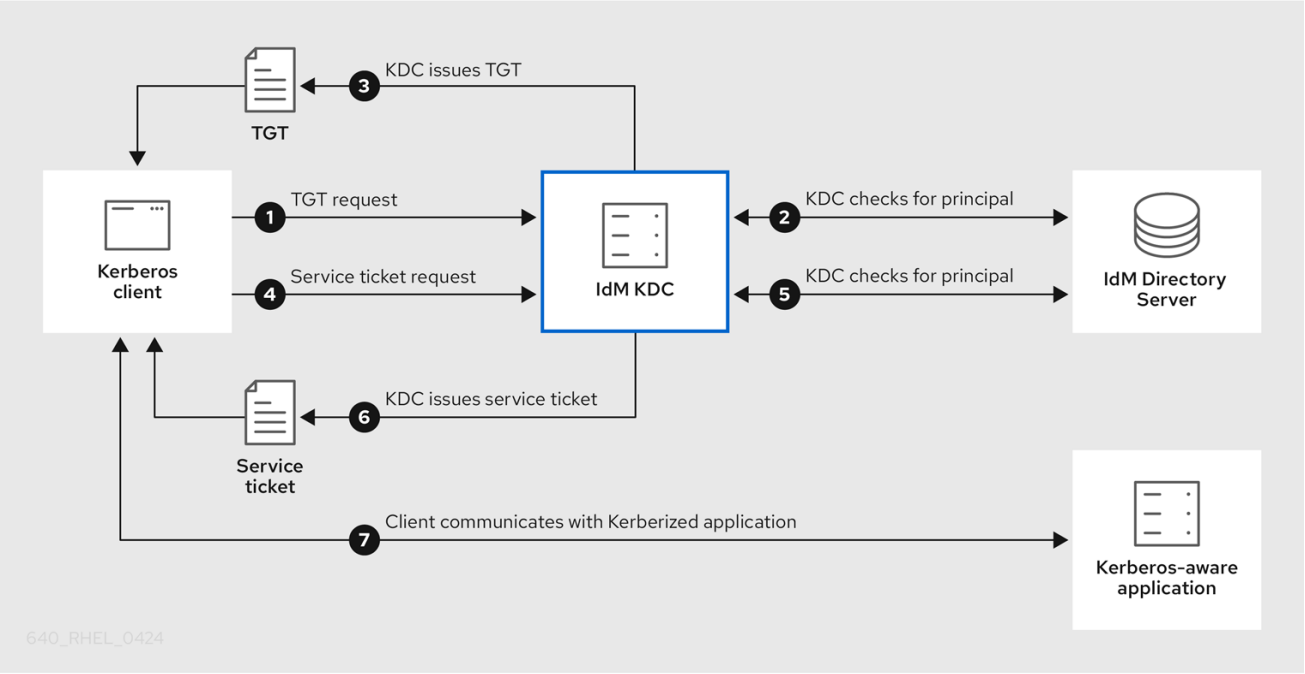
14.1. IDM KDC のロール

Identity Management の認証メカニズムは、Key Distribution Center (KDC) が設定する Kerberos インフラストラクチャーを使用します。KDC は、認証情報を保存し、IdM ネットワーク内のエンティティーから発信されるデータの信頼性を確保する信頼できる認証局です。

各 IdM ユーザー、サービス、およびホストは Kerberos クライアントとして機能し、一意の Kerberos **プリンシパル** で識別されます。

- ユーザーの場合: **identifier@REALM** (例: **admin@EXAMPLE.COM**)
- サービスの場合: **service/fully-qualified-hostname@REALM** (例: **http/server.example.com@EXAMPLE.COM**)
- ホストの場合: **host/fully-qualified-hostname@REALM** (例: **host/client.example.com@EXAMPLE.COM**)

以下の図では、Kerberos クライアント、KDC、およびクライアントの通信先となる Kerberos を使用するアプリケーション間の通信を簡単にまとめています。



1. Kerberos クライアントは、Kerberos プリンシパルとして認証することで KDC に対して身分を証明します。たとえば、IdM ユーザーは **kinit ユーザー名** を実行し、パスワードを指定します。

2. KDC はデータベースのプリンシパルを確認し、クライアントを認証し、Kerberos チケットポリシー を評価してリクエストを付与するかどうかを判断します。

3. KDC は、適切なチケットポリシーに従って、ライフサイクルおよび 認証インジケーター で Ticket-Granting Ticket (TGT) を発行します。

4. TGT により、クライアントは KDC から **サービスチケット** を要求し、ターゲットホストで Kerberos を使用するサービスと通信します。

5. KDC は、クライアントの TGT が有効であるかどうかを確認し、チケットポリシーに対してサービスチケット要求を評価します。

6. KDC はクライアントに **サービスチケット** を発行します。

7. サービスチケットを使用すると、クライアントはターゲットホストのサービスと暗号化された通信を開始できます。

14.2. IDM KERBEROS チケットポリシータイプ

IdM Kerberos チケットポリシーは、以下のチケットポリシータイプを実装します。

接続ポリシー

異なるレベルのセキュリティーで Kerberos を使用するサービスを保護するには、接続ポリシーを定義して、TGT (Ticket-granting ticket) の取得にクライアントが使用する事前認証メカニズムをもとにルールを有効にすることができます。
たとえば **client1.example.com** に接続するには、スマートカード認証が、**client2.example.com** の **testservice** アプリケーションにアクセスするには、2 要素認証が必要です。

接続ポリシーを有効するには、**認証インジケーター** をサービスに関連付けます。必要な認証インジケーターがサービスチケット要求に含まれるクライアントのみがこれらのサービスにアクセスできます。詳細は、Kerberos 認証インジケーター を参照してください。

チケットライフサイクルポリシー

各 Kerberos チケットには **有効期間** と潜在的な **更新期間** があります。チケットは最長期間に達する前に更新できますが、更新期間の超過後には更新できません。
デフォルトのグローバルチケットの有効期間は 1 日 (86400 秒) で、デフォルトのグローバル最大更新期間は 1 週間 (604800 秒) です。これらのグローバル値を調整するには、グローバルチケットライフサイクルポリシーの設定 を参照してください。

独自のチケットライフサイクルポリシーを定義することもできます。

- 各認証インジケーターに異なるグローバルチケットライフサイクル値を設定するには、認証インジケーターごとのグローバルチケットポリシーの設定 を参照してください。
- 使用する認証方法に関係なく適用する単一のユーザーのチケットライフサイクル値を定義するには、ユーザーのデフォルトチケットポリシーの設定 を参照してください。
- 単一ユーザーにのみ適用される認証インジケーター別のチケットライフサイクル値を定義するには、ユーザーの個別認証インジケーターチケットポリシーの設定 を参照してください。

14.3. KERBEROS 認証インジケーター

Kerberos Key Distribution Center (KDC) は、**認証インジケーター** を、ID の証明にクライアントが使用する事前認証メカニズムに基づいて TGT (Ticket-granting Ticket) に割り当てます。

otp

2 要素認証 (パスワード + ワンタイムパスワード)

radius

radius 認証 (通常は 802.1x 認証)

pkinit

PKINIT 、スマートカード、または証明書での認証

hardened

強化パスワード (SPAKE または FAST)[1]

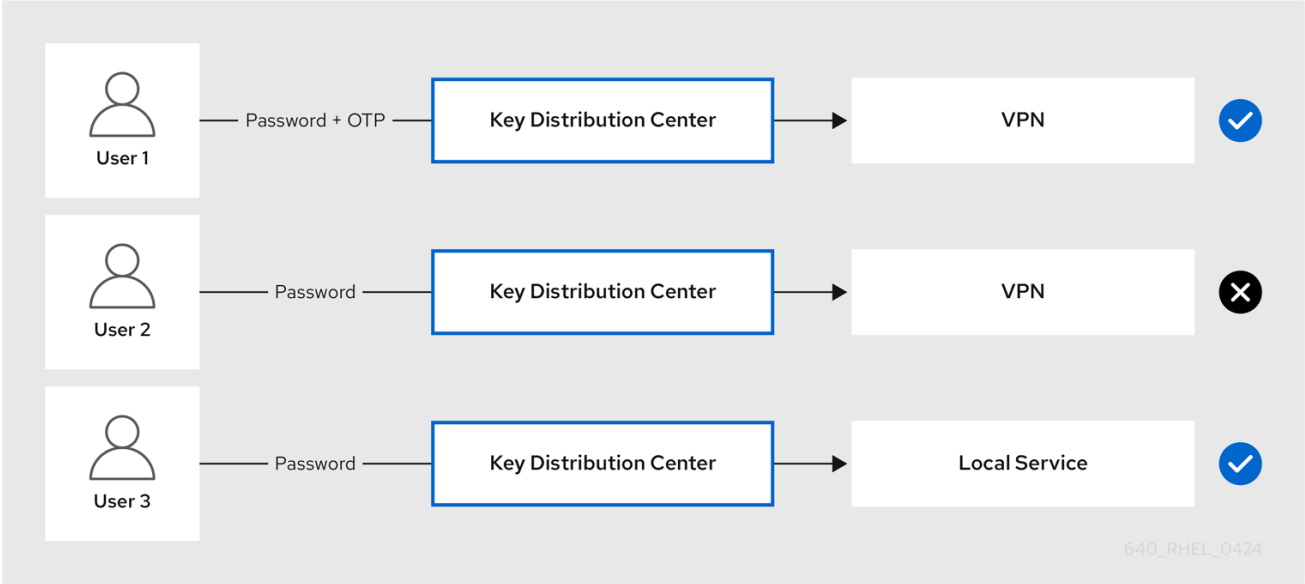
次に KDC は、TGT からの認証インジケーターを、TGT から取得するサービスチケット要求に割り当てます。KDC は、認証インジケーターに基づいて、サービスアクセス制御、チケットの最大有効期間、および更新可能な期間などのポリシーを有効にします。

認証インジケーターおよび IdM サービス

サービスまたはホストを認証インジケーターに関連付けると、対応する認証メカニズムを使用して TGT を取得したクライアントのみがアクセスできるようになります。KDC はアプリケーションやサービスではなく、サービスチケット要求の認証インジケーターをチェックし、Kerberos 接続ポリシーに基づいて要求を付与または拒否します。

たとえば、仮想プライベートネットワーク (VPN) に接続するために 2 要素認証を要求するには、**otp** 認証インジケーターをそのサービスに関連付けます。KDC から最初の TGT を取得するのにワンタイムパスワードを使用したユーザーのみが VPN にログインできます。

図14.1 otp 認証インジケータを必要とする VPN サービスの例



サービスまたはホストに認証インジケーターが割り当てられていない場合には、メカニズムに関係なく認証されたチケットを受け入れます。

関連情報

- IdM サービスの認証インジケーターの有効化
- IdM クライアントでの GSSAPI 認証の有効化および sudo の Kerberos 認証インジケーターの有効化

14.4. IDM サービスの認証インジケーターの有効化

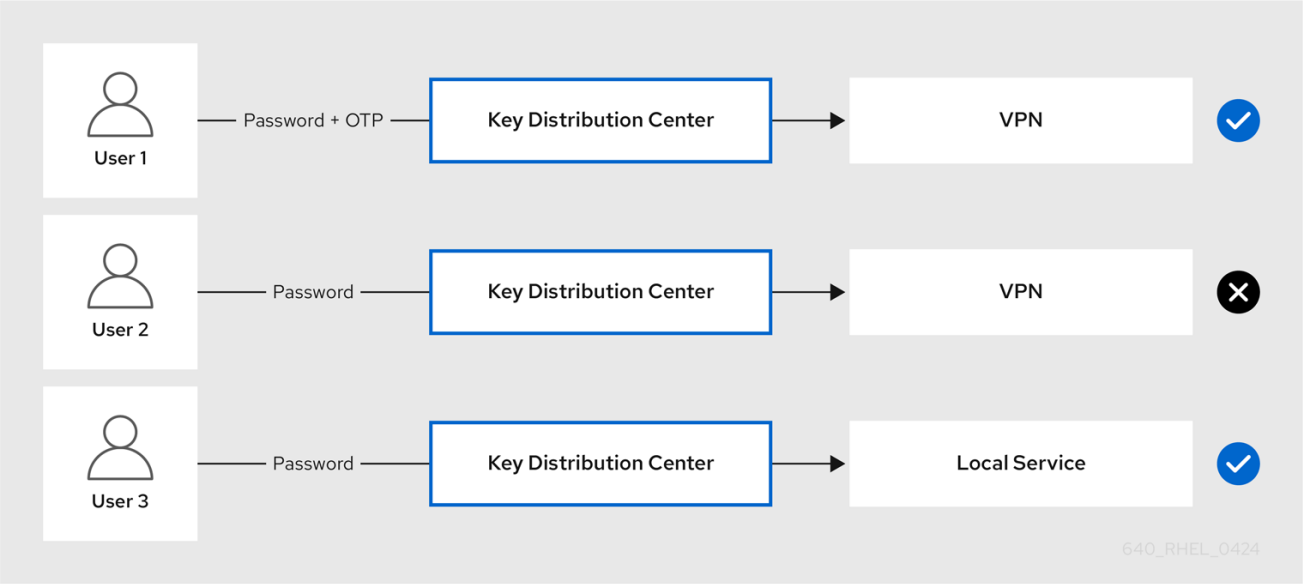
Identity Management (IdM) がサポートする認証メカニズムは、認証強度によって異なります。たとえば、標準パスワードと組み合わせて、ワンタイムパスワード (OTP) を使用して、最初の Kerberos Ticket-Granting Ticket (TGT) を取得することは、標準パスワードのみを使用する認証よりも安全と見なされます。

認証インジケーターを特定の IdM サービスに関連付けることで、IdM 管理者として、これらの特定の事前認証メカニズムを使用して最初の Ticket-Granting Ticket (TGT) を取得したユーザーのみがサービスにアクセスできるようにサービスを設定できます。

この方法では、以下のように異なる IdM サービスを設定できます。

- ワンタイムパスワード (OTP) などのより強力な認証方法を使用して最初の TGT を取得したユーザーのみが、VPN などのセキュリティーにとって重要なサービスにアクセスできます。
- パスワードなど、より簡単な認証方法を使用して初期の TGT を取得したユーザーは、ローカルログインなどの重要でないサービスにのみアクセスできます。

図14.2 異なる技術を使用した認証の例



以下の手順では、IdM サービスを作成し、着信サービスチケット要求から特定の Kerberos 認証インジケーターを必要とするように設定する方法を説明します。

14.4.1. IdM サービスエントリーおよびその Kerberos キータブの作成

IdM ホストで実行しているサービスの IdM に **IdM サービス** エントリーを追加すると、対応する Kerberos プリンシパルが作成され、サービスが SSL 証明書、Kerberos キータブ、またはその両方を要求できるようになります。

以下の手順では、IdM サービスエントリーを作成して、関連の Kerberos キータブを生成し、対象のサービスとの通信を暗号化する方法を説明します。

前提条件

- サービスが、Kerberos プリンシパル、SSL 証明書、またはその両方を保存できる。

手順

1. **ipa service-add** コマンドで IdM サービスを追加して、これに関連付けられた Kerberos プリンシパルを作成します。たとえば、ホスト **client.example.com** で実行する **testservice** アプリケーションの IdM サービスエントリーを作成するには、次のコマンドを実行します。

```
[root@client ~]# ipa service-add testservice/client.example.com
---------------------------------------------------------------
Modified service "testservice/client.example.com@EXAMPLE.COM"
---------------------------------------------------------------
  Principal name: testservice/client.example.com@EXAMPLE.COM
  Principal alias: testservice/client.example.com@EXAMPLE.COM
  Managed by: client.example.com
```

2. クライアント上でサービスの Kerberos キータブを生成し、保存します。

```
[root@client ~]# ipa-getkeytab -k /etc/testservice.keytab -p
testservice/client.example.com
Keytab successfully retrieved and stored in: /etc/testservice.keytab
```

検証手順

1. **ipa service-show** コマンドを使用して、IdM サービスに関する情報を表示します。

```
[root@server ~]# ipa service-show testservice/client.example.com
  Principal name: testservice/client.example.com@EXAMPLE.COM
  Principal alias: testservice/client.example.com@EXAMPLE.COM
  Keytab: True
  Managed by: client.example.com
```

2. **klist** コマンドを使用して、サービスの Kerberos キータブの内容を表示します。

```
[root@server etc]# klist -ekt /etc/testservice.keytab
Keytab name: FILE:/etc/testservice.keytab
KVNO Timestamp           Principal
---- ------------------- ------------------------------------------------------
   2 04/01/2020 17:52:55 testservice/client.example.com@EXAMPLE.COM (aes256-cts-
hmac-sha1-96)
   2 04/01/2020 17:52:55 testservice/client.example.com@EXAMPLE.COM (aes128-cts-
hmac-sha1-96)
   2 04/01/2020 17:52:55 testservice/client.example.com@EXAMPLE.COM (camellia128-cts-
cmac)
   2 04/01/2020 17:52:55 testservice/client.example.com@EXAMPLE.COM (camellia256-cts-
cmac)
```

14.4.2. IdM CLI を使用した IdM サービスへの認証インジケーターの関連付け

Identity Management (Id M) 管理者は、クライアントアプリケーションによって提示されるサービスチケットに特定の認証インジケーターが含まれていることを要求するようにホストまたはサービスを設定できます。たとえば、Kerberos Ticket-Granting Ticket (TGT) を取得する際に、パスワードで有効な IdM 2 要素認証トークンを使用したユーザーのみが、そのホストまたはサービスにアクセスできるようにできます。

受信サービスチケット要求からの特定の Kerberos 認証インジケーターを要求するようにサービスを設定するには、次の手順に従います。

前提条件

- IdM ホストで実行しているサービスの IdM サービスエントリーを作成している。IdM サービスエントリーおよびその Kerberos キータブの作成 を参照してください。
- IdM の管理ユーザーの Ticket-Granting Ticket (TGT) を取得している。

警告

内部 IdM サービスに認証インジケーターを割り当て **ない** でください。以下の IdM サービスでは、PKINIT およびマルチファクター認証方式で必要なインタラクティブな認証ステップを実行できません。

> **host**/server.example.com@EXAMPLE.COM
> **HTTP**/server.example.com@EXAMPLE.COM
> **ldap**/server.example.com@EXAMPLE.COM
> **DNS**/server.example.com@EXAMPLE.COM
> **cifs**/server.example.com@EXAMPLE.COM

手順

- **ipa service-mod** コマンドを使用して、サービスに必要な認証インジケーターを **--auth-ind** 引数で識別して指定します。

認証方法	--auth-ind 値
2 要素認証	**otp**
radius 認証	**radius**
PKINIT 、スマートカード、または証明書での認証	**pkinit**
強化パスワード (SPAKE または FAST)	**hardened**

たとえば、スマートカードまたは OTP 認証で認証したユーザーには **client.example.com** ホストの **testservice** プリンシパルを取得させるようにするには、以下を実行します。

```
[root@server ~]# ipa service-mod testservice/client.example.com@EXAMPLE.COM --auth-ind otp --auth-ind pkinit
-----------------------------------------------------------
Modified service "testservice/client.example.com@EXAMPLE.COM"
-----------------------------------------------------------
  Principal name: testservice/client.example.com@EXAMPLE.COM
  Principal alias: testservice/client.example.com@EXAMPLE.COM
  Authentication Indicators: otp, pkinit
  Managed by: client.example.com
```

注記

サービスからすべての認証インジケーターを削除するには、インジケーターの空のリストを指定します。

```
[root@server ~]# ipa service-mod
testservice/client.example.com@EXAMPLE.COM --auth-ind ''
-----------------------------------------------------
Modified service "testservice/client.example.com@EXAMPLE.COM"
-----------------------------------------------------
  Principal name: testservice/client.example.com@EXAMPLE.COM
  Principal alias: testservice/client.example.com@EXAMPLE.COM
  Managed by: client.example.com
```

検証手順

- **ipa service-show** コマンドを使用して、必要な認証インジケーターなど、IdM サービスに関する情報を表示します。

```
[root@server ~]# ipa service-show testservice/client.example.com
  Principal name: testservice/client.example.com@EXAMPLE.COM
  Principal alias: testservice/client.example.com@EXAMPLE.COM
  Authentication Indicators: otp, pkinit
  Keytab: True
  Managed by: client.example.com
```

関連情報

- IdM サービスの Kerberos サービスチケットの取得
- IdM クライアントでの GSSAPI 認証の有効化および sudo の Kerberos 認証インジケーターの有効化

14.4.3. IdM Web UI を使用した IdM サービスへの認証インジケーターの関連付け

Identity Management (IdM) 管理者は、ホストまたはサービスが、クライアントアプリケーションが提示するサービスチケットを特定の認証インジケーターを含むように設定できます。たとえば、Kerberos Ticket-Granting Ticket (TGT) を取得する際に、パスワードで有効な IdM 2 要素認証トークンを使用したユーザーのみが、そのホストまたはサービスにアクセスできるようにできます。

IdM Web UI を使用して、受信チケット要求からの特定の Kerberos 認証インジケーターを要求するようにホストまたはサービスを設定するには、次の手順に従います。

前提条件

- 管理ユーザーとして IdM Web UI にログインしている。

手順

1. **Identity → Hosts** または **Identity → Services** を選択します。
2. 必要なホストまたはサービスの名前をクリックします。
3. **Authentication indicators** で、必要な認証方法を選択します。

- たとえば、**OTP** を選択すると、Kerberos TGT を取得する際に、パスワードで有効な IdM 2 要素認証トークンを使用したユーザーのみが、ホストまたはサービスにアクセスできるようになります。
- **OTP** と **RADIUS** の両方を選択した場合は、Kerberos TGT を取得する際に有効な IdM 二要素認証トークンとパスワードを使用したユーザー **および** Kerberos TGT の取得に RADIUS サーバーを使用したユーザーの両方が、アクセスを許可されます。

4. ページ上部にある **Save** をクリックします。

関連情報

- IdM サービスの Kerberos サービスチケットの取得
- IdM クライアントでの GSSAPI 認証の有効化および sudo の Kerberos 認証インジケーターの有効化

14.4.4. IdM サービスの Kerberos サービスチケットの取得

以下の手順では、IdM サービスの Kerberos サービスチケットを取得する方法を説明します。この手順を使用して、特定の Kerberos 認証インジケーターが Ticket-Granting Ticket (TGT) に存在することを強制するなど、Kerberos チケットポリシーをテストできます。

前提条件

- 対応する **IdM サービス** エントリーを作成している (使用しているサービスが内部 IdM サービスではない場合)。IdM サービスエントリーおよびその Kerberos キータブの作成 を参照してください。
- Kerberos Ticket-Granting Ticket (TGT) がある。

手順

- サービスチケットを取得するには、**kvno** コマンドに **-S** オプションを指定して、IdM サービスの名前と管理するホストの完全修飾ドメイン名を指定します。

```
[root@server ~]# kvno -S testservice client.example.com
testservice/client.example.com@EXAMPLE.COM: kvno = 1
```

注記

IdM サービスにアクセスする必要があり、現在の Ticket-Granting Ticket (TGT) に必要な Kerberos 認証インジケーターが関連付けられていない場合は、**kdestroy** コマンドで現在の Kerberos 認証情報キャッシュを消去し、新しい TGT を取得します。

```
[root@server ~]# kdestroy
```

たとえば、パスワードを使用して認証し、**pkinit** 認証インジケーターが関連付けられた IdM サービスにアクセスする必要がある場合は、現在の認証情報キャッシュを破棄し、スマートカードで再認証します。Kerberos 認証インジケーター を参照してください。

検証手順

- **klist** コマンドを使用して、サービスチケットがデフォルトの Kerberos 認証情報キャッシュにあることを確認します。

```
[root@server etc]# klist_
Ticket cache: KCM:1000
Default principal: admin@EXAMPLE.COM

Valid starting       Expires              Service principal
04/01/2020 12:52:42  04/02/2020 12:52:39  krbtgt/EXAMPLE.COM@EXAMPLE.COM
04/01/2020 12:54:07 04/02/2020 12:52:39
testservice/client.example.com@EXAMPLE.COM
```

14.4.5. 関連情報

- Kerberos 認証インジケーター を参照してください。

14.5. グローバルチケットライフサイクルポリシーの設定

グローバルチケットポリシーは、すべてのサービスチケットとユーザーごとのチケットポリシーが定義されていないユーザーに適用されます。

以下の手順では、**ipa krbtpolicy-mod** コマンドを使用して、グローバル Kerberos チケットポリシーのチケットの最大有効期間と最大更新期間を調整する方法を説明します。

ipa krbtpolicy-mod コマンドを使用する場合は、以下の引数のいずれかを指定します。

- **--maxlife** - チケットの最大有効期間 (秒単位)
- **--maxrenew** - 更新可能な最大期間 (秒単位)

手順

1. グローバルチケットポリシーを変更するには、以下を実行します。

```
[root@server ~]# ipa krbtpolicy-mod --maxlife=$((8*60*60)) --maxrenew=$((24*60*60))
  Max life: 28800
  Max renew: 86400
```

この例では、最大有効期間は 8 時間 (8 * 60 分 * 60 秒) に設定され、最大の更新可能期間は 1 日 (24 * 60 分 * 60 秒) に設定されています。

2. オプション: グローバルの Kerberos チケットポリシーをデフォルトのインストール値にリセットするには、以下を実行します。

```
[root@server ~]# ipa krbtpolicy-reset
  Max life: 86400
  Max renew: 604800
```

検証手順

- グローバルチケットポリシーを表示します。

```
[root@server ~]# ipa krbtpolicy-show
  Max life: 28800
  Max renew: 86640
```

関連情報

- ユーザーのデフォルトチケットポリシーの設定 を参照してください。
- ユーザーの個々の認証インジケーターチケットポリシーの設定 を参照してください。

14.6. 認証インジケーターごとのグローバルチケットポリシーの設定

各認証インジケーターのグローバルのチケット最大有効期間と最大更新可能期間を調整するには、次の手順に従います。この設定は、ユーザー別のチケットポリシーが定義されていないユーザーに適用されます。

ipa krbtpolicy-mod コマンドを使用して、それぞれに割り当てられた 認証インジケーター に合わせて、Kerberos チケットのグローバルの最大有効期間または更新可能な期間を指定します。

手順

- たとえば、グローバルな 2 要素のチケットの有効期間と更新期間の値を 1 週間に、グローバルスマートカードチケットの有効期間と更新期間の値を 2 週間に設定するには以下を実行します。

```
[root@server ~]# ipa krbtpolicy-mod --otp-maxlife=604800 --otp-maxrenew=604800 --
pkinit-maxlife=172800 --pkinit-maxrenew=172800
```

検証手順

- グローバルチケットポリシーを表示します。

```
[root@server ~]# ipa krbtpolicy-show
  Max life: 86400
  OTP max life: 604800
  PKINIT max life: 172800
  Max renew: 604800
  OTP max renew: 604800
  PKINIT max renew: 172800
```

OTP および PKINIT の値は、グローバルなデフォルトの **Max life** および **Max renew** 値とは異なることに注意してください。

関連情報

- **krbtpolicy-mod** コマンドの認証インジケーターオプション を参照してください。
- ユーザーのデフォルトチケットポリシーの設定 を参照してください。
- ユーザーの個々の認証インジケーターチケットポリシーの設定 を参照してください。

14.7. ユーザーのデフォルトチケットポリシーの設定

一意の Kerberos チケットポリシーを定義して、単一のユーザーだけに適用できます。これらのユーザーごとの設定は、すべての認証インジケーターに対してグローバルチケットポリシーをオーバーライドします。

ipa krbtpolicy-mod username コマンドを使用して、最低でも以下のいずれかの引数を指定します。

- **--maxlife** - チケットの最大有効期間 (秒単位)
- **--maxrenew** - 更新可能な最大期間 (秒単位)

手順

1. たとえば、IdM **管理者** ユーザーの最大チケット期間を 2 日に、最大更新期間を 2 週に設定するには、次のコマンドを実行します。

```
[root@server ~]# ipa krbtpolicy-mod admin --maxlife=172800 --maxrenew=1209600
  Max life: 172800
  Max renew: 1209600
```

2. オプション: ユーザーのチケットポリシーをリセットするには、以下を実行します。

```
[root@server ~]# ipa krbtpolicy-reset admin
```

検証手順

- ユーザーに適用される有効な Kerberos チケットポリシーを表示します。

```
[root@server ~]# ipa krbtpolicy-show admin
  Max life: 172800
  Max renew: 1209600
```

関連情報

- グローバルチケットライフサイクルポリシーの設定 を参照してください。
- 認証インジケーターごとのグローバルチケットポリシーの設定 を参照してください。

14.8. ユーザーの個別認証インジケーターチケットポリシーの設定

管理者は、ユーザーに、認証インジケーター別に異なる Kerberos チケットポリシーを定義できます。たとえば、IdM **管理者** ユーザーは、チケットを OTP 認証で取得した場合には 2 日間、スマートカード認証で取得した場合には 1 週間更新できるようにポリシーを設定できます。

これらの認証インジケーター設定は、**ユーザーの** デフォルトのチケットポリシー、**グローバル** デフォルトチケットポリシー、および **グローバル** 認証インジケーターチケットポリシーをオーバーライドします。

ipa krbtpolicy-mod username コマンドを使用して、それぞれに割り当てられた 認証インジケーター に合わせて、ユーザー Kerberos チケットのカスタム最大有効期間および最大の更新可能期間を設定します。

手順

1. たとえば、ワンタイムパスワード認証でチケットを取得した場合に IdM **admin** ユーザーが 2 日間 Kerberos チケットを更新できるようにするには、**--otp-maxrenew** オプションを設定します。

```
[root@server ~]# ipa krbtpolicy-mod admin --otp-maxrenew=$((2*24*60*60))
  OTP max renew: 172800
```

2. オプション: ユーザーのチケットポリシーをリセットするには、以下を実行します。

```
[root@server ~]# ipa krbtpolicy-reset username
```

検証手順

- ユーザーに適用される有効な Kerberos チケットポリシーを表示します。

```
[root@server ~]# ipa krbtpolicy-show admin
  Max life: 28800
  Max renew: 86640
```

関連情報

- **krbtpolicy-mod** コマンドの認証インジケーターオプション を参照してください。
- ユーザーのデフォルトチケットポリシーの設定 を参照してください。
- グローバルチケットライフサイクルポリシーの設定 を参照してください。
- 認証インジケーターごとのグローバルチケットポリシーの設定 を参照してください。

14.9. KRBTPOLICY-MOD コマンドの認証インジケーターオプション

以下の引数で認証インジケーターの値を指定します。

表14.1 krbtpolicy-mod コマンドの認証インジケーターオプション

認証インジケーター	最大有効期間の引数	最大更新期間の引数
otp	**--otp-maxlife**	**--otp-maxrenew**
radius	**--radius-maxlife**	**--radius-maxrenew**
pkinit	**--pkinit-maxlife**	**--pkinit-maxrenew**
hardened	**--hardened-maxlife**	**--hardened-maxrenew**

[1] 強化されたパスワードは、Single-Party Public-Key Authenticated Key Exchange (SPAKE) の事前認証または Flexible Authentication via Secure Tunneling (FAST) 防御を使用して、総当たりパスワード辞書攻撃を受けないようにセキュリティー保護されています。

第15章 IDM の KERBEROS PKINIT 認証

Kerberos(PKINIT) の初期認証の公開鍵暗号化は、Kerberos の事前認証メカニズムです。Identity Management (IdM) サーバーには、Kerberos PKINIT 認証のメカニズムが含まれています。

15.1. デフォルトの PKINIT 設定

IdM サーバーのデフォルトの PKINIT 設定は、認証局 (CA) 設定によって異なります。

表15.1 IdM のデフォルトの PKINIT 設定

CA 設定	PKINIT の設定
CA なし、外部 PKINIT 証明書の提供なし	ローカル PKINIT: IdM はサーバーの内部用途でのみ PKINIT を使用します。
CA なし、外部 PKINIT 証明書の IdM への提供あり	IdM は、外部の Kerberos 鍵配布センター (KDC) 証明書と CA 証明書を使用して PKINIT を設定します。
統合 CA あり	IdM は、IdM CA が署名した証明書を使用して PKINIT を設定します。

15.2. 現在の PKINIT 設定の表示

IdM には、ドメインの PKINIT 設定をクエリーするのに使用できるコマンドが複数用意されています。

手順

- ドメインの PKINIT のステータスを確認するには、**ipa pkinit-status** コマンドを使用します。

```
$ ipa pkinit-status
  Server name: server1.example.com
  PKINIT status: enabled
  [...output truncated...]
  Server name: server2.example.com
  PKINIT status: disabled
  [...output truncated...]
```

  このコマンドは、**enabled** または **disabled** として PKINIT 設定の状態を表示します。

  - **Enabled**: PKINIT は、統合 IdM CA または外部 PKINIT 証明書により署名された証明書を使用して設定されます。
  - **Disabled**: IdM は、IdM サーバーでの内部目的でのみ PKINIT を使用します。

- IdM クライアントの PKINIT に対応するアクティブな Kerberos 鍵配布センター (KDC) がある IdM サーバーをリスト表示するには、任意のサーバーで **ipa config-show** コマンドを使用します。

```
$ ipa config-show
  Maximum username length: 32
  Home directory base: /home
```

```
Default shell: /bin/sh
Default users group: ipausers
[...output truncated...]
IPA masters capable of PKINIT: server1.example.com
[...output truncated...]
```

15.3. IDM での PKINIT の設定

IdM サーバーが PKINIT を無効にした状態で動作している場合は、以下の手順に従って有効にします。たとえば、**--no-pkinit** オプションを **ipa-server-install** ユーティリティーまたは **ipa-replica-install** ユーティリティーで渡した場合には、サーバーは PKINIT が無効な状態で動作します。

前提条件

- 認証局 (CA) がインストールされているすべての IdM サーバーが、同じドメインレベルで稼働していることを確認します。

手順

1. PKINIT がサーバーで有効になっているかどうかを確認します。

```
# kinit admin

Password for admin@IDM.EXAMPLE.COM:
# ipa pkinit-status --server=server.idm.example.com
1 server matched
----------------
Server name: server.idm.example.com
PKINIT status:enabled
----------------------------
Number of entries returned 1
----------------------------
```

PKINIT が無効になっている場合は、次の出力が表示されます。

```
# ipa pkinit-status --server server.idm.example.com
-----------------
0 servers matched
-----------------
----------------------------
Number of entries returned 0
----------------------------
```

--server <server_fqdn> パラメーターを省略することで、このコマンドを使用して、PKINIT が有効になっているすべてのサーバーを検索することもできます。

2. CA なしで IdM を使用している場合は、以下の手順を実行します。

   a. IdM サーバーに、Kerberos Key Distribution Center (KDC) 証明書に署名した CA 証明書をインストールします。

```
# ipa-cacert-manage install -t CT,C,C ca.pem
```

b. すべての IPA ホストを更新するには、すべてのレプリカとクライアントで **ipa-certupdate** コマンドを繰り返し実行します。

```
# ipa-certupdate
```

c. **ipa-cacert-manage list** コマンドを使用して、CA 証明書がすでに追加されているかどうかを確認します。以下に例を示します。

```
# ipa-cacert-manage list
CN=CA,O=Example Organization
The ipa-cacert-manage command was successful
```

d. **ipa-server-certinstall** ユーティリティーを使用して、外部 KDC 証明書をインストールします。KDC 証明書は以下の条件を満たしている必要があります。

- コモンネーム **CN=fully_qualified_domain_name,certificate_subject_base** で発行されている。
- Kerberos プリンシパル **krbtgt/REALM_NAME@REALM_NAME** を含む。
- KDC 認証のオブジェクト識別子 (OID) **1.3.6.1.5.2.3.5** を含む。

```
# ipa-server-certinstall --kdc kdc.pem kdc.key

# systemctl restart krb5kdc.service
```

e. PKINIT のステータスを確認します。

```
# ipa pkinit-status
  Server name: server1.example.com
  PKINIT status: enabled
  [...output truncated...]
  Server name: server2.example.com
  PKINIT status: disabled
  [...output truncated...]
```

3. CA 証明書のある IdM を使用している場合は、次のように PKINIT を有効にします。

```
# ipa-pkinit-manage enable
  Configuring Kerberos KDC (krb5kdc)
  [1/1]: installing X509 Certificate for PKINIT
  Done configuring Kerberos KDC (krb5kdc).
  The ipa-pkinit-manage command was successful
```

IdM CA を使用している場合、コマンドは CA から PKINIT KDC 証明書を要求します。

関連情報

- **ipa-server-certinstall(1)** の man ページ

15.4. 関連情報

- Kerberos PKINIT の詳細は、MIT Kerberos ドキュメントの PKINIT 設定

第16章 IDM KERBEROS キータブファイルの維持

Kerberos キータブファイルとは何か、および Identity Management (IdM) がそれを使用してサービスが Kerberos で安全に認証できるようにする方法について詳しく説明します。

この情報を使用して、これらの機密ファイルを保護する必要がある理由を理解し、IdM サービス間の通信の問題をトラブルシューティングできます。

詳細は、以下のトピックを参照してください。

- Identity Management が Kerberos キータブファイルを使用する方法
- Kerberos キータブファイルが IdM データベースと同期していることの確認
- IdM Kerberos キータブファイルとその内容のリスト
- IdM マスターキーの暗号化タイプの表示

16.1. IDENTITY MANAGEMENT が KERBEROS キータブファイルを使用する方法

Kerberos キータブは、Kerberos プリンシパルとそれに対応する暗号化キーを含むファイルです。ホスト、サービス、ユーザー、およびスクリプトは、キータブを使用して、人間の介入を必要とせずに、Kerberos Key Distribution Center (KDC) に対して安全に認証することができます。

IdM サーバー上のすべての IdM サービスには、Kerberos データベースに格納された一意の Kerberos プリンシパルがあります。たとえば、IdM サーバー **east.idm.example.com** および **west.idm.example.com** が DNS サービスを提供する場合、IdM はこれらのサービスを識別するために 2 つの一意の DNS Kerberos プリンシパルを作成します。これらは、命名規則 **<service>/host.domain.com@REALM.COM** に従います:

- **DNS/east.idm.example.com@IDM.EXAMPLE.COM**
- **DNS/west.idm.example.com@IDM.EXAMPLE.COM**

IdM は、Kerberos キーのローカルコピーをキーバージョン番号 (KVNO) とともに保存するために、これらのサービスごとにサーバー上にキータブを作成します。たとえば、デフォルトのキータブファイル **/etc/krb5.keytab** には **host** プリンシパルが格納されます。このプリンシパルは、Kerberos レルム内のそのマシンを表し、ログイン認証に使用されます。KDC は、**aes256-cts-hmac-sha1-96** および **aes128-cts-hmac-sha1-96** など、サポートするさまざまな暗号化アルゴリズムの暗号化キーを生成します。

klist コマンドを使用して、キータブファイルの内容を表示できます。

```
[root@idmserver ~]# klist -ekt /etc/krb5.keytab
Keytab name: FILE:/etc/krb5.keytab
KVNO Timestamp           Principal
---- ------------------- ------------------------------------------------------
   2 02/24/2022 20:28:09 host/idmserver.idm.example.com@IDM.EXAMPLE.COM (aes256-cts-hmac-sha1-96)
   2 02/24/2022 20:28:09 host/idmserver.idm.example.com@IDM.EXAMPLE.COM (aes128-cts-hmac-sha1-96)
   2 02/24/2022 20:28:09 host/idmserver.idm.example.com@IDM.EXAMPLE.COM (camellia128-cts-
```

```
cmac)
   2 02/24/2022 20:28:09 host/idmserver.idm.example.com@IDM.EXAMPLE.COM (camellia256-cts-
cmac)
```

関連情報

- Kerberos キータブファイルが IdM データベースと同期していることの確認
- IdM Kerberos キータブファイルとその内容のリスト

16.2. KERBEROS キータブファイルが IDM データベースと同期していることの確認

Kerberos パスワードを変更すると、IdM は対応する新しい Kerberos キーを自動的に生成し、そのキーバージョン番号 (KVNO) を増やします。Kerberos キータブが新しいキーと KVNO で更新されていない場合、そのキータブに依存して有効なキーを取得するサービスは、Kerberos キー配布センター (KDC) に対して認証できない可能性があります。

IdM サービスの 1 つが別のサービスと通信できない場合は、次の手順を使用して、Kerberos キータブファイルが IdM データベースに保存されているキーと同期していることを確認します。それらが同期していない場合は、更新されたキーと KVNO を使用して Kerberos キータブを取得します。この例では、IdM サーバーの更新された **DNS** プリンシパルを比較して取得します。

前提条件

- キータブファイルを取得するには、IdM 管理者アカウントとして認証する必要がある。
- 他のユーザーが所有するキータブファイルを変更するには、**root** アカウントとして認証する必要がある。

手順

1. 検証しているキータブでプリンシパルの KVNO を表示します。次の例では、**/etc/named.keytab** ファイルに、KVNO が 2 の **DNS/server1.idm.example.com@EXAMPLE.COM** プリンシパルのキーがあります。

```
[root@server1 ~]# klist -ekt /etc/named.keytab
Keytab name: FILE:/etc/named.keytab
KVNO Timestamp           Principal
---- ------------------- ------------------------------------------------------
   2 11/26/2021 13:51:11 DNS/server1.idm.example.com@EXAMPLE.COM (aes256-cts-
hmac-sha1-96)
   2 11/26/2021 13:51:11 DNS/server1.idm.example.com@EXAMPLE.COM (aes128-cts-
hmac-sha1-96)
   2 11/26/2021 13:51:11 DNS/server1.idm.example.com@EXAMPLE.COM (camellia128-cts-
cmac)
   2 11/26/2021 13:51:11 DNS/server1.idm.example.com@EXAMPLE.COM (camellia256-cts-
cmac)
```

2. IdM データベースに保存されているプリンシパルの KVNO を表示します。この例では、IdM データベースのキーの KVNO がキータブの KVNO と一致しません。

```
[root@server1 ~]# kvno DNS/server1.idm.example.com@EXAMPLE.COM
DNS/server1.idm.example.com@EXAMPLE.COM: kvno = 3
```

3. IdM 管理者アカウントとして認証します。

```
[root@server1 ~]# kinit admin
Password for admin@IDM.EXAMPLE.COM:
```

4. プリンシパルの更新された Kerberos キーを取得し、キータブに保存します。**root** ユーザーとしてこのステップを実行して、**named** ユーザーが所有する **/etc/named.keytab** ファイルを変更できるようにします。

```
[root@server1 ~]# ipa-getkeytab -s server1.idm.example.com -p
DNS/server1.idm.example.com -k /etc/named.keytab
```

検証

1. プリンシパルの更新された KVNO をキータブに表示します。

```
[root@server1 ~]# klist -ekt /etc/named.keytab
Keytab name: FILE:/etc/named.keytab
KVNO Timestamp           Principal
---- ------------------- ------------------------------------------------------
   4 08/17/2022 14:42:11 DNS/server1.idm.example.com@EXAMPLE.COM (aes256-cts-
hmac-sha1-96)
   4 08/17/2022 14:42:11 DNS/server1.idm.example.com@EXAMPLE.COM (aes128-cts-
hmac-sha1-96)
   4 08/17/2022 14:42:11 DNS/server1.idm.example.com@EXAMPLE.COM (camellia128-cts-
cmac)
   4 08/17/2022 14:42:11 DNS/server1.idm.example.com@EXAMPLE.COM (camellia256-cts-
cmac)
```

2. IdM データベースに保存されているプリンシパルの KVNO を表示し、キータブの KVNO と一致することを確認します。

```
[root@server1 ~]# kvno DNS/server1.idm.example.com@EXAMPLE.COM
DNS/server1.idm.example.com@EXAMPLE.COM: kvno = 4
```

関連情報

- Identity Management が Kerberos キータブファイルを使用する方法
- IdM Kerberos キータブファイルとその内容のリスト

16.3. IDM KERBEROS キータブファイルとその内容のリスト

以下の表は、IdM Kerberos キータブファイルの場所、内容、および目的を示しています。

表16.1 テーブル

キータブの場所	内容	目的

キータブの場所	内容	目的
/etc/krb5.keytab	**host** プリンシパル	ログイン時にユーザーの認証情報を確認します (**nfs** プリンシパルがない場合に NFS によって使用されます)。
/etc/dirsrv/ds.keytab	**ldap** プリンシパル	IdM データベースに対してユーザーを認証し、IdM レプリカ間でデータベースの内容を安全に複製します。
/var/lib/ipa/gssproxy/http.keytab	**HTTP** プリンシパル	Apache サーバーに対して認証します。
/etc/named.keytab	**DNS** プリンシパル	DNS レコードを安全に更新します。
/etc/ipa/dnssec/ipa-dnskeysyncd.keytab	**ipa-dnskeysyncd** プリンシパル	OpenDNSSEC と LDAP の同期を維持します。
/etc/pki/pki-tomcat/dogtag.keytab	**dogtag** プリンシパル	認証局 (CA) と通信します。
/etc/samba/samba.keytab	**cifs** プリンシパルと **host** プリンシパル	Samba サービスと通信します。
/var/lib/sss/keytabs/ad-domain.com.keytab	**HOSTNAME$@AD-DOMAIN.COM** 形式の Active Directory (AD) ドメインコントローラー (DC) プリンシパル	IdM-AD 信頼を介して AD DC と通信します。

関連情報

- Identity Management が Kerberos キータブファイルを使用する方法
- Kerberos キータブファイルが IdM データベースと同期していることの確認

16.4. IDM マスターキーの暗号化タイプの表示

Identity Management (IdM) 管理者は、IdM マスターキーの暗号化タイプを表示できます。これは、IdM Kerberos Distribution Center (KDC) が保存時に他のすべてのプリンシパルを暗号化するために使用するキーです。暗号化の種類を知ることで、デプロイメントの FIPS 標準との互換性を判断するのに役立ちます。

RHEL 8.7 の時点で、暗号化タイプは **aes256-cts-hmac-sha384-192** です。この暗号化タイプは、FIPS 140-3 への準拠を目的としたデフォルトの RHEL 9 FIPS 暗号化ポリシーと互換性があります。

以前の RHEL バージョンで使用されていた暗号化タイプは、FIPS 140-3 標準に準拠する RHEL 9 システムと互換性がありません。FIPS モードの RHEL 9 システムを RHEL 8 FIPS 140-2 デプロイメントと互換性を持たせるには、RHEL 9 システムで **FIPS:AD-SUPPORT** 暗号化ポリシーを有効にします。

注記

Microsoft の Active Directory 実装は、SHA-2 HMAC を使用する RFC8009 Kerberos 暗号化タイプをまだサポートしていません。したがって、IdM-AD 信頼が設定されている場合、IdM マスターキーの暗号化タイプが **aes256-cts-hmac-sha384-192** であっても、FIPS:AD-SUPPORT 暗号サブポリシーの使用が必要になります。

前提条件

- IdM デプロイメント内の RHEL 8 レプリカのいずれかに **root** アクセス権がある。

手順

- レプリカで、コマンドラインインターフェイスで暗号化の種類を表示します。

```
# kadmin.local getprinc K/M | grep -E '^Key:'
Key: vno 1, aes256-cts-hmac-sha1-96
```

出力の **aes256-cts-hmac-sha1-96** キーは、IdM デプロイメントが RHEL 8.6 以前を実行しているサーバーにインストールされたことを示しています。出力に **aes256-cts-hmac-sha384-192** キーが存在する場合、IdM デプロイメントが RHEL 8.7 以降を実行しているサーバーにインストールされたことを示します。

第17章 IDM 環境でのパスキー認証の有効化

Fast IDentity Online 2 (FIDO2) 標準は、公開鍵暗号化をベースとしており、PIN または生体認証を使用したパスワードレスのフローを選択肢として追加するものです。IdM 環境でのパスキー認証では、**libfido2** ライブラリーでサポートされている FIDO2 互換デバイスが使用されます。

パスキー認証方式は、PIN または指紋を必要とするパスワードレス認証と多要素認証 (MFA) を組み込むことで、規制標準に準拠するための追加のセキュリティーレイヤーを提供します。Identity Management (IdM) 環境でのパスキーデバイスとパスキー有効化など、特殊なハードウェアとソフトウェアの組み合わせを使用して、データ保護が重要な環境でセキュリティーを強化します。

システムが IdM 環境のネットワークに接続されている場合、パスキー認証方法によって Kerberos チケットが自動的に発行され、IdM ユーザーのシングルサインオン (SSO) が有効になります。

パスキーを使用すると、オペレーティングシステムのグラフィカルインターフェイスを介して認証できます。システムでパスキーとパスワードによる認証が許可されている場合は、キーボードの **Space** を押してから **Enter** キーを押すことで、パスキー認証をスキップし、パスワードで認証することができます。GNOME デスクトップマネージャー (GDM) を使用する場合は、**Enter** を押してパスキー認証を回避できます。

現在、IdM 環境のパスキー認証では、特定のパスキーデバイスの識別を可能にする FIDO2 アテステーションメカニズムがサポートされていないことに注意してください。

次の手順では、IdM 環境でパスキー認証を管理および設定する方法について説明します。

17.1. 前提条件

- パスキーデバイスを用意する。
- **fido2-tools** パッケージをインストールする。

```
# dnf install fido2-tools
```

- パスキーデバイスの PIN を設定する。
  1. パスキーデバイスを USB ポートに接続します。
  2. 接続されているパスキーデバイスをリスト表示します。

```
# fido2-token -L
```

  3. コマンドプロンプトに従って、パスキーデバイスの PIN を設定します。

```
# fido2-token -C passkey_device
```

17.2. パスキーデバイスの登録

ユーザーはパスキーデバイスを使用して認証を設定できます。パスキーデバイスは、YubiKey 5 NFC などのあらゆる FIDO2 仕様デバイスと互換性があります。この認証方法を設定するには、以下の手順に従ってください。

前提条件

- パスキーデバイスの PIN が設定されている。

- IdM ユーザーに対してパスキー認証が有効になっている。

```
# ipa user-add user01 --first=user --last=01 --user-auth-type=passkey
```

既存の IdM ユーザーに対しては、上記と同じ **--user-auth-type=passkey** パラメーターを指定した **ipa user-mod** を使用します。

- ユーザーが認証する物理マシンにアクセスできる。

手順

1. パスキーデバイスを USB ポートに挿入します。
2. IdM ユーザーのパスキーを登録します。

```
# ipa user-add-passkey user01 --register
```

アプリケーションのプロンプトに従います。

a. パスキーデバイスの PIN を入力します。

b. デバイスをタッチしてアイデンティティー確認を行います。生体認証デバイスを使用している場合は、必ずデバイスの登録に使用したのと同じ指を使用します。

複数の場所またはデバイスからの認証を可能にするバックアップ手段として、複数のパスキーデバイスを設定することを推奨します。認証中に Kerberos チケットが発行されるようにするために、ユーザーに 12 個を超えるパスキーデバイスを設定することは避けてください。

検証

1. パスキー認証を使用するように設定したユーザー名でシステムにログインします。パスキーデバイスを挿入するよう求めるプロンプトが表示されます。

```
Insert your passkey device, then press ENTER.
```

2. パスキーデバイスを USB ポートに挿入し、プロンプトが表示されたら PIN を入力します。

```
Enter PIN:
Creating home directory for user01@example.com.
```

3. Kerberos チケットが発行されたことを確認します。

```
$ klist
Default principal: user01@IPA.EXAMPLE.COM
```

パスキー認証をスキップするには、プロンプトに任意の文字を入力するか、ユーザー認証が有効になっている場合は空の PIN を入力します。パスワードベースの認証にリダイレクトされます。

17.3. 認証ポリシー

認証ポリシーは、利用可能なオンライン認証方法とローカル認証方法を設定するために使用します。

オンライン接続を使用した認証

サービスがサーバー側で提供するオンライン認証方法をすべて使用します。IdM、AD、または Kerberos サービスの場合、デフォルトの認証方法は Kerberos です。

オンライン接続なしでの認証

ユーザーが利用できる認証方法を使用します。**local_auth_policy** オプションを使用して認証方法を調整できます。

使用可能なオンラインおよびオフライン認証方法を設定するには、**/etc/sssd/sssd.conf** ファイルの **local_auth_policy** オプションを使用します。デフォルトでは、サービスのサーバー側でサポートされている方法のみを使用して認証が実行されます。次の値を使用してポリシーを調整できます。

- **match** 値は、オフラインとオンラインの状態のマッチングを有効にします。たとえば、IdM サーバーがオンラインパスキー認証をサポートしている場合に **match** を使用すると、パスキー方式のオフライン認証とオンライン認証が有効になります。
- **only** 値は、オフラインの方法のみを提供し、オンラインの方法は無視します。
- **enable** および **disable** 値は、オフライン認証の方法を明示的に定義します。たとえば、**enable:passkey** は、オフライン認証のパスキーのみを有効にします。

次の設定例では、ローカルユーザーがスマートカード認証を使用してローカルで認証できるようにします。

```
[domain/shadowutils]
id_provider = proxy
proxy_lib_name = files
auth_provider = none
local_auth_policy = only
```

local_auth_policy オプションは、パスキー認証方法とスマートカード認証方法に適用されます。

17.4. パスキーユーザーとして IDM チケット許可チケットを取得する

パスキーユーザーとして Kerberos TGT (Ticket-granting ticket) を取得するには、匿名の Kerberos チケットを要求し、Secure Tunneling (FAST) チャネルを介したフレキシブル認証を有効にして、Kerberos クライアントと Kerberos ディストリビューションセンター (KDC) 間のセキュアな接続を提供します。

前提条件

- IdM クライアントと IdM サーバーが RHEL 9.1 以降を使用している。
- IdM クライアントと IdM サーバーが SSSD 2.7.0 以降を使用している。
- パスキーデバイスを登録し、認証ポリシーを設定している　。

手順

1. 次のコマンドを実行して認証情報キャッシュを初期化します。

```
[root@client ~]# kinit -n @IDM.EXAMPLE.COM -c FILE:armor.ccache
```

このコマンドは、新しい Kerberos チケットを要求するたびに指定する必要がある armor.ccache ファイルを作成することに注意してください。

2. 次のコマンドを実行して Kerberos チケットを要求します。

```
[root@client ~]# kinit -T FILE:armor.ccache <username>@IDM.EXAMPLE.COM
Enter your PIN:
```

検証

- Kerberos チケット情報を表示します。

```
[root@client ~]# klist -C
Ticket cache: KCM:0:58420
Default principal: <username>@IDM.EXAMPLE.COM

Valid starting     Expires            Service principal
05/09/22 07:48:23  05/10/22 07:03:07  krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
config: fast_avail(krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM) = yes
08/17/2022 20:22:45  08/18/2022 20:22:43
krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
config: pa_type(krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM) = 153
```

pa_type = 153 は、パスキー認証を示します。

第18章 IDM での KDC プロキシーの使用

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

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

IdM サーバーでは、KKDCP はデフォルトで有効で、**https://server.idm.example.com/KdcProxy** で利用できます。IdM クライアントでは、KDCP にアクセスするために、その Kerberos 設定を変更する必要があります。

18.1. KKDCP を使用するための IDM クライアントの設定

Identity Management (IdM) システム管理者は、IdM サーバーで Kerberos Key Distribution Center Proxy (KKDCP) を使用するように IdM クライアントを設定できます。これは、デフォルトの Kerberos ポートが IdM サーバーでアクセスできず、**HTTPS** ポート 443 が Kerberos サービスにアクセスする唯一の方法である場合に役立ちます。

前提条件

- IdM クライアントへの **root** アクセス権限がある。

手順

1. **/etc/krb5.conf** ファイルを開いて編集します。
2. **[realms]** セクションで、**kdc**、**admin_server**、および **kpasswd_server** オプションの KKDCP の URL を入力します。

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

   冗長性の場合は、パラメーター **kdc**、**admin_server**、および **kpasswd_server** を複数回追加して、別の KDCP サーバーを示すことができます。

3. **sssd** を再起動して、変更を有効にします。

```
~]# systemctl restart sssd
```

18.2. IDM サーバーで KKDCP が有効になっていることの確認

Identity Management (Id M) サーバーでは、属性と値のペア **ipaConfigString=kdcProxyEnabled** がディレクトリーに存在する場合、Apache Web サーバーが起動するたびに Kerberos Key Distribution Center Proxy (KKDCP) が自動的に有効になります。この場合は、シンボリックリンク **/etc/httpd/conf.d/ipa-kdc-proxy.conf** が作成されます。

非特権ユーザーであっても、IdM サーバーで KKDCP が有効になっているかどうかを確認できます。

手順

- シンボリックリンクが存在することを確認します。

```
$ ls -l /etc/httpd/conf.d/ipa-kdc-proxy.conf
lrwxrwxrwx. 1 root root 36 Jun 21  2020 /etc/httpd/conf.d/ipa-kdc-proxy.conf -> /etc/ipa/kdcproxy/ipa-kdc-proxy.conf
```

この出力は、KKDCP が有効になっていることを確認します。

18.3. IDM サーバーでの KDCP の無効化

Identity Management (IdM) システム管理者は、IdM サーバーで Kerberos Key Distribution Center Proxy (KKDCP) を無効にできます。

前提条件

- IdM サーバーへの **root** アクセス権限がある。

手順

1. ディレクトリーから **ipaConfigString=kdcProxyEnabled** 属性と値のペアを削除します。

```
# ipa-ldap-updater /usr/share/ipa/kdcproxy-disable.uldif
Update complete
The ipa-ldap-updater command was successful
```

2. **httpd** サービスを再起動します。

```
# systemctl restart httpd.service
```

現在の IdM サーバーで、KDCP が無効になりました。

検証手順

- シンボリックリンクが存在しないことを確認します。

```
$ ls -l /etc/httpd/conf.d/ipa-kdc-proxy.conf
ls: cannot access '/etc/httpd/conf.d/ipa-kdc-proxy.conf': No such file or directory
```

18.4. IDM サーバーでの KDCP の再有効化

IdM サーバーでは、Kerberos Key Distribution Center Proxy (KKDCP) はデフォルトで有効で、**https://server.idm.example.com/KdcProxy** で利用できます。

KDCP がサーバーで無効になっている場合は、再度有効にできます。

前提条件

- IdM サーバーへの **root** アクセス権限がある。

手順

1. **ipaConfigString=kdcProxyEnabled** 属性と値のペアをディレクトリーに追加します。

```
# ipa-ldap-updater /usr/share/ipa/kdcproxy-enable.uldif
Update complete
The ipa-ldap-updater command was successful
```

2. **httpd** サービスを再起動します。

```
# systemctl restart httpd.service
```

現在の IdM サーバーで KDCP が有効になりました。

検証手順

- シンボリックリンクが存在することを確認します。

```
$ ls -l /etc/httpd/conf.d/ipa-kdc-proxy.conf
lrwxrwxrwx. 1 root root 36 Jun 21  2020 /etc/httpd/conf.d/ipa-kdc-proxy.conf ->
/etc/ipa/kdcproxy/ipa-kdc-proxy.conf
```

18.5. KKDCP サーバー I の設定

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

前提条件

- **root** アクセスがある。

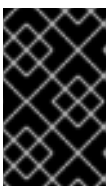
手順

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

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

2. プロキシーされたレルム情報を **/etc/ipa/kdcproxy/kdcproxy.conf** ファイルに入れます。たとえば、プロキシーを使用する [AD.**EXAMPLE.COM**] レルムの場合、次のようにレルム設定パラメーターをリスト表示します。

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



重要

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

3. Identity Management (IdM) サービスを再起動します。

```
# ipactl restart
```

関連情報

- Red Hat ナレッジベースの Configure IPA server as a KDC Proxy for AD Kerberos communication を参照してください。

18.6. KKDCP サーバー II の設定

次のサーバー設定は、DNS サービスレコードに依存して、通信する Active Directory (AD) サーバーを検索します。

前提条件

- **root** アクセスがある。

手順

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

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

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

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

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

3. Identity Management (IdM) サービスを再起動します。

```
# ipactl restart
```

関連情報

- Red Hat ナレッジベースの Configure IPA server as a KDC Proxy for AD Kerberos communication を参照してください。

第19章 CLI を使用した IDM でのセルフサービスルールの管理

Identity Management (IdM) のセルフサービスルールと、コマンドラインインターフェイス (CLI) でセルフサービスアクセスルールを作成および編集する方法について説明します。

19.1. IDM でのセルフサービスアクセス制御

セルフサービスのアクセス制御ルールは、Identity Management (IdM) エンティティーが IdM Directory Server エントリーで実行できる操作を定義します (例: IdM ユーザーが独自のパスワードを更新できるなど)。

この制御方法では、認証された IdM エンティティーがその LDAP エントリー内の特定の属性を編集できますが、エントリー全体での **追加** や **削除** の操作は許可されません。

警告

セルフサービスのアクセス制御規則を使用する場合は注意が必要です。アクセス制御ルールを誤って設定すると、エンティティーの特権が誤って昇格する可能性があります。

19.2. CLI を使用したセルフサービスルールの作成

コマンドラインインターフェイス (CLI) を使用して IdM でセルフサービスアクセスルールを作成するには、次の手順に従います。

前提条件

- IdM、または **ユーザー管理者** ロールを管理する管理者権限
- 有効な Kerberos チケット。詳細は、Using kinit to log in to IdM manually を参照してください。

手順

- セルフサービスルールを追加するには、**ipa selfservice-add** コマンドを使用して、以下の 2 つのオプションを指定します。

  --permissions

  アクセス制御命令 (ACI) が付与する **読み取り** パーミッションおよび **書き込み** パーミッションを設定します。

  --attrs

  ACI がパーミッションを付与する属性の完全なリストを設定します。

たとえば、セルフサービスルールを作成して、ユーザーが独自の名前の詳細を変更できるようにするには、次のコマンドを実行します。

```
$ ipa selfservice-add "Users can manage their own name details" --permissions=write --
attrs=givenname --attrs=displayname --attrs=title --attrs=initials
----------------------------------------------------------
```

```
Added selfservice "Users can manage their own name details"
----------------------------------------------------------
  Self-service name: Users can manage their own name details
  Permissions: write
  Attributes: givenname, displayname, title, initials
```

19.3. CLI を使用したセルフサービスルールの編集

コマンドラインインターフェイス (CLI) を使用して IdM でセルフサービスアクセスルールを編集するには、次の手順に従います。

前提条件

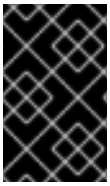
- IdM、または **ユーザー管理者** ロールを管理する管理者権限
- 有効な Kerberos チケット。詳細は、Using kinit to log in to IdM manually を参照してください。

手順

1. **オプション**: **ipa selfservice-find** コマンドを使用して、既存のセルフサービスルールを表示します。
2. **オプション**: **ipa selfservice-show** コマンドを使用して、変更するセルフサービスルールの詳細を表示します。
3. **ipa selfservice-mod** コマンドを使用してセルフサービスルールを編集します。

以下に例を示します。

```
$ ipa selfservice-mod "Users can manage their own name details" --attrs=givenname --
attrs=displayname --attrs=title --attrs=initials --attrs=surname
----------------------------------------------------------
Modified selfservice "Users can manage their own name details"
----------------------------------------------------------
Self-service name: Users can manage their own name details
Permissions: write
Attributes: givenname, displayname, title, initials
```



重要

ipa selfservice-mod コマンドを使用すると、以前に定義されたパーミッションと属性が上書きされるため、定義する必要のある新しいパーミッションおよび属性と共に、既存のパーミッションおよび属性の完全なリストも常に含めるようにしてください。

検証手順

- **ipa selfservice-show** コマンドを使用して、編集したセルフサービスルールを表示します。

```
$ ipa selfservice-show "Users can manage their own name details"
----------------------------------------------------------
Self-service name: Users can manage their own name details
Permissions: write
Attributes: givenname, displayname, title, initials
```

19.4. CLI を使用したセルフサービスルールの削除

コマンドラインインターフェイス (CLI) を使用して IdM でセルフサービスアクセスルールを削除するには、次の手順に従います。

前提条件

- IdM、または **ユーザー管理者** ロールを管理する管理者権限
- 有効な Kerberos チケット。詳細は、Using kinit to log in to IdM manually を参照してください。

手順

- **ipa selfservice-del** コマンドを使用してセルフサービスルールを削除します。

以下に例を示します。

```
$ ipa selfservice-del "Users can manage their own name details"
--------------------------------------------------------
Deleted selfservice "Users can manage their own name details"
--------------------------------------------------------
```

検証手順

- **ipa selfservice-find** コマンドを使用して、すべてのセルフサービスルールを表示します。削除したばかりのルールがなくなっているはずです。

第20章 IDM WEB UI を使用したセルフサービスルールの管理

Identity Management (IdM) のセルフサービスルールと、Web インターフェイス (IdM Web UI) でセルフサービスアクセスルールを作成および編集する方法について説明します。

20.1. IDM でのセルフサービスアクセス制御

セルフサービスのアクセス制御ルールは、Identity Management (IdM) エンティティーが IdM Directory Server エントリーで実行できる操作を定義します (例: IdM ユーザーが独自のパスワードを更新できるなど)。

この制御方法では、認証された IdM エンティティーがその LDAP エントリー内の特定の属性を編集できますが、エントリー全体での **追加** や **削除** の操作は許可されません。

警告

セルフサービスのアクセス制御規則を使用する場合は注意が必要です。アクセス制御ルールを誤って設定すると、エンティティーの特権が誤って昇格する可能性があります。

20.2. IDM WEB UI を使用したセルフサービスルールの作成

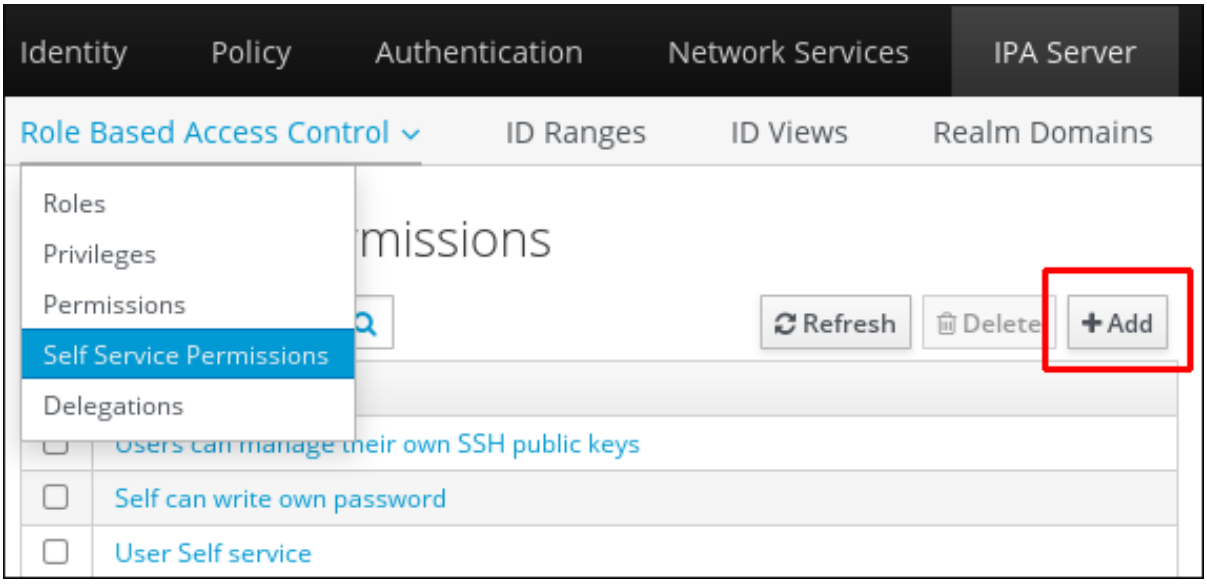
Web インターフェイス (IdM Web UI) を使用して IdM でセルフサービスアクセスルールを作成するには、次の手順に従います。

前提条件

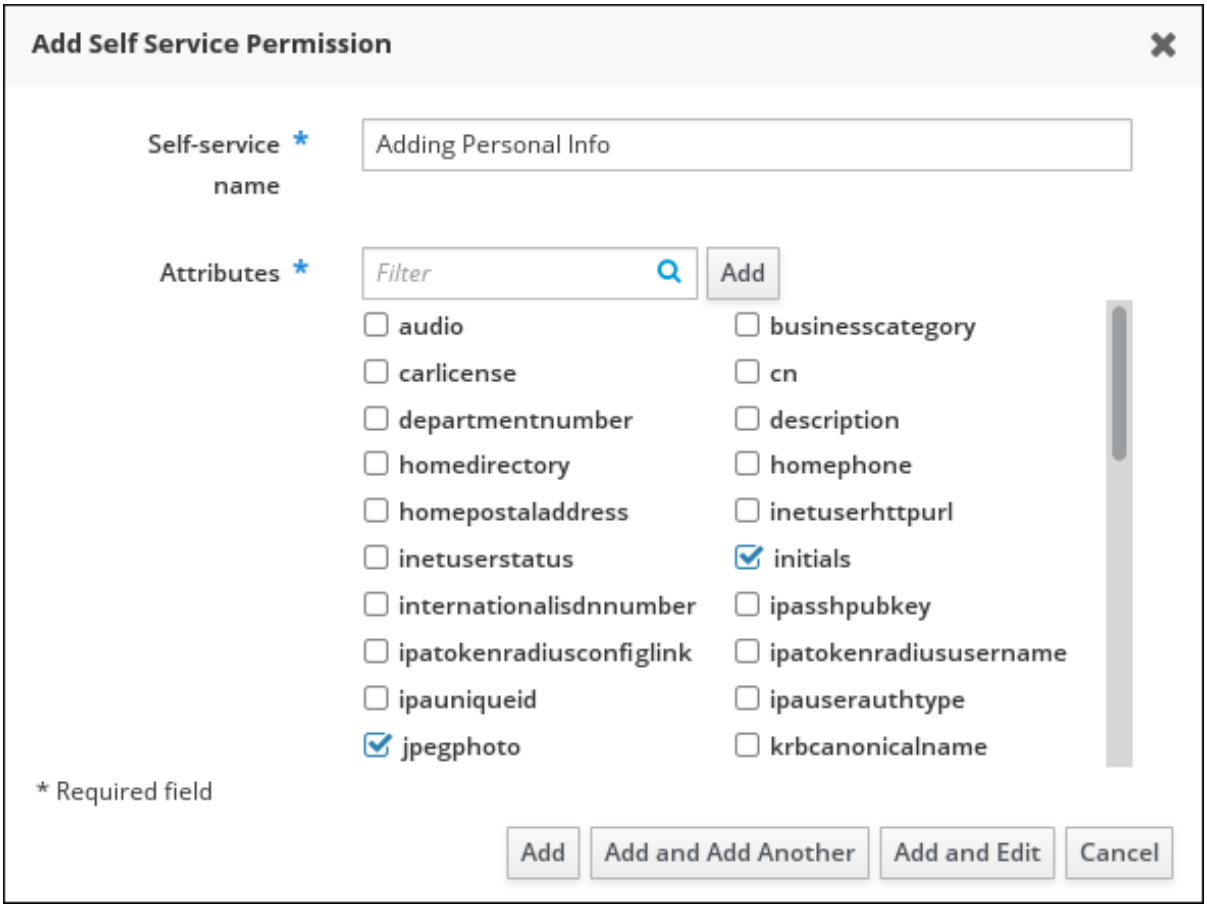
- IdM、または **ユーザー管理者** ロールを管理する管理者権限
- IdM Web UI にログインしている。詳細は、Web ブラウザーでの IdM Web UI へのアクセス を参照してください。

手順

1. **IPA Server** タブで **Role-Based Access Control** サブメニューを開き、**Self Service Permissions** を選択します。
2. セルフサービスアクセスルールの一覧の右上にある **Add** をクリックします。



3. **Add Self Service Permission** ウィンドウが開きます。**Self-service name** フィールドに新しいセルフサービスルールの名前を入力します。空白を使用できます。



4. ユーザーによる編集を可能にする属性の横にあるチェックボックスを選択します。

5. **必要に応じて** アクセス可能にする属性が一覧にない場合には、一覧に追加できます。

   a. **Add** ボタンをクリックします。

   b. 以下の **Add Custom Attribute** ウィンドウの **Attribute** テキストフィールドに属性名を入力します。

   c. **OK** ボタンをクリックして属性を追加します。

d. 新しい属性が選択されていることを確認します。

6. フォームの下部にある **Add** ボタンをクリックして、新しいセルフサービスルールを保存します。
または、**Add and Edit** ボタンをクリックしてセルフサービスルールを編集するか、**Add and Add another** ボタンをクリックしてさらにルールを保存および追加できます。

20.3. IDM WEB UI を使用したセルフサービスルールの編集

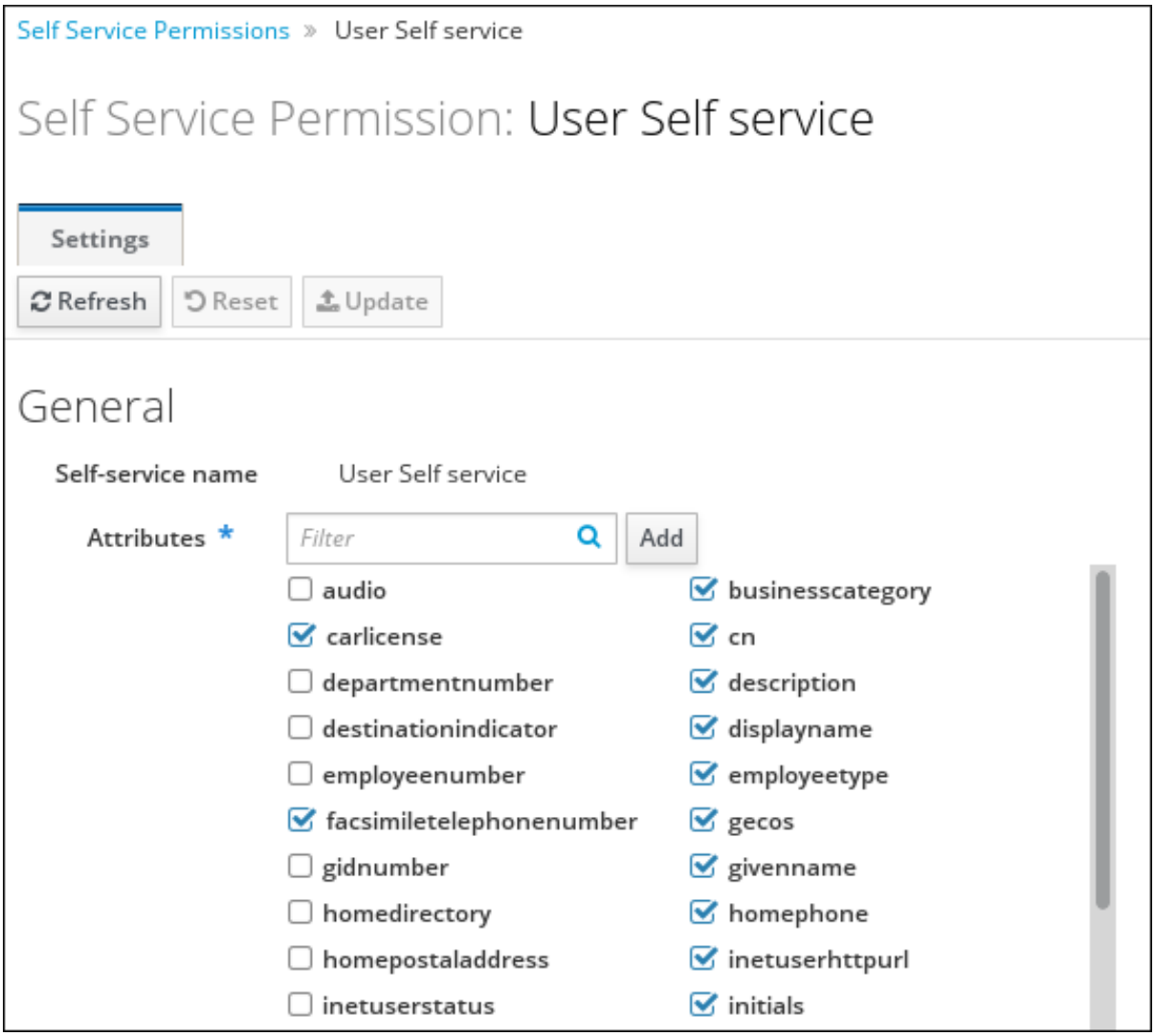
Web インターフェイス (IdM Web UI) を使用して IdM でセルフサービスアクセスルールを編集するには、次の手順に従います。

前提条件

- IdM、または **ユーザー管理者** ロールを管理する管理者権限
- IdM Web UI にログインしている。詳細は、Web ブラウザーでの IdM Web UI へのアクセス を参照してください。

手順

1. **IPA Server** タブで **Role-Based Access Control** サブメニューを開き、**Self Service Permissions** を選択します。
2. 変更するセルフサービスルールの名前をクリックします。



3. 編集ページでは、セルフサービスルールに追加または削除する属性のリストの編集だけが可能です。適切なチェックボックスを選択または選択解除します。

4. **Save** ボタンをクリックして、セルフサービスルールへの変更を保存します。

20.4. IDM WEB UI を使用したセルフサービスルールの削除

Web インターフェイス (IdM Web UI) を使用して IdM のセルフサービスアクセスルールを削除するには、次の手順に従います。

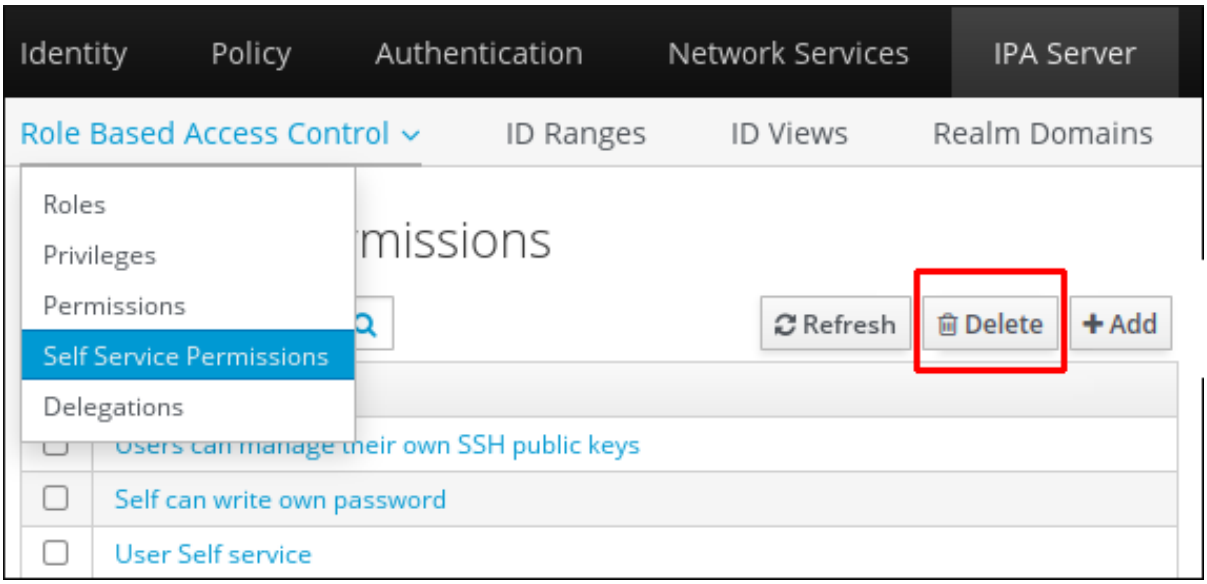
前提条件

- IdM、または **ユーザー管理者** ロールを管理する管理者権限
- IdM Web UI にログインしている。詳細は、Web ブラウザーでの IdM Web UI へのアクセス を参照してください。

手順

1. **IPA Server** タブで **Role-Based Access Control** サブメニューを開き、**Self Service Permissions** を選択します。

2. 削除するルールの横にあるチェックボックスを選択し、リストの右側にある **Delete** ボタンをクリックします。



3. ダイアログが開き、**Delete** をクリックして確認します。

第21章 ANSIBLE PLAYBOOK を使用した IDM でのセルフサービスルールの管理

本セクションでは、Identity Management (IdM) のセルフサービスルールを紹介し、Ansible Playbook を使用してセルフサービスアクセスルールを作成および編集する方法を説明します。セルフサービスのアクセス制御ルールにより、IdM エンティティーは IdM Directory Server エントリーで指定の操作を実行できます。

- IdM でのセルフサービスアクセス制御
- Ansible を使用してセルフサービスルールを存在させる手順
- Ansible を使用してセルフサービスルールがないことを確認する手順
- Ansible を使用してセルフサービスルールに固有の属性を含める手順
- Ansible を使用してセルフサービスルールに固有の属性を含めいないようにする手順

21.1. IDM でのセルフサービスアクセス制御

セルフサービスのアクセス制御ルールは、Identity Management (IdM) エンティティーが IdM Directory Server エントリーで実行できる操作を定義します (例: IdM ユーザーが独自のパスワードを更新できるなど)。

この制御方法では、認証された IdM エンティティーがその LDAP エントリー内の特定の属性を編集できますが、エントリー全体での **追加** や **削除** の操作は許可されません。

警告

セルフサービスのアクセス制御規則を使用する場合は注意が必要です。アクセス制御ルールを誤って設定すると、エンティティーの特権が誤って昇格する可能性があります。

21.2. ANSIBLE を使用してセルフサービスルールを存在させる手順

以下の手順では、Ansible Playbook を使用してセルフサービスルールを定義し、Identity Management (IdM) サーバーに存在させる方法を説明します。この例では、**ユーザーが自分の名前の情報を管理できる** 新しいルールでは、ユーザーに、自分の **givenname**、**displayname**、**title**、および **initials** 属性を変更できるよ権限を付与します。たとえば、表示名や初期などを変更することができます。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。

- ~/MyPlaybooks/ ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
- この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。

- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。

手順

1. ~/ MyPlaybooks/ ディレクトリーに移動します。

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

2. **/usr/share/doc/ansible-freeipa/playbooks/selfservice/** ディレクトリーにある **selfservice-present.yml** のコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/selfservice/selfservice-present.yml
selfservice-present-copy.yml
```

3. Ansible Playbook ファイル (**selfservice-present-copy.yml**) を開きます。

4. **ipaselfservice** タスクセクションに以下の変数を設定してファイルを調整します。
   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は、新しいセルフサービスルールの名前に設定します。
   - **permission** 変数は、付与するパーミッションをコンマ区切りのリスト (**read** および **write**) で設定します。
   - **attribute** 変数は、ユーザーが管理できる属性 (**givenname**、**displayname**、**title**、および **initials**) をリストとして設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Self-service present
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure self-service rule "Users can manage their own name details" is present
    ipaselfservice:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: "Users can manage their own name details"
      permission: read, write
      attribute:
      - givenname
      - displayname
      - title
      - initials
```

5. ファイルを保存します。

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

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory selfservice-present-copy.yml
```

関連情報

- IdM でのセルフサービスアクセス制御 を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-selfservice.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/selfservice** ディレクトリーを参照してください。

21.3. ANSIBLE を使用してセルフサービスルールがないことを確認する手順

以下の手順では、Ansible Playbook を使用して、指定したセルフサービスルールが IdM 設定に存在しないことを確認する方法を説明します。以下の例では、**ユーザーが自分の名前の詳細** のセルフサービスルールが IdM に存在しないことを確認する方法を説明します。これにより、ユーザーは自分の表示名や初期などを変更できないようにします。

前提条件

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

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

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

2. **/usr/share/doc/ansible-freeipa/playbooks/selfservice/** ディレクトリーにある **selfservice-absent.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/selfservice/selfservice-absent.yml selfservice-absent-copy.yml
```

3. Ansible Playbook ファイル (**selfservice-absent-copy.yml**) を開きます。
4. **ipaselfservice** タスクセクションに以下の変数を設定してファイルを調整します。
   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は、セルフサービスルールの名前に設定します。
   - **state** 変数は **absent** に設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Self-service absent
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure self-service rule "Users can manage their own name details" is absent
    ipaselfservice:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: "Users can manage their own name details"
      state: absent
```

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

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory selfservice-absent-copy.yml
```

関連情報

- IdM でのセルフサービスアクセス制御 を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-selfservice.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/selfservice** ディレクトリーのサンプルの Playbook を参照してください。

21.4. ANSIBLE を使用してセルフサービスルールに固有の属性を含める手順

以下の手順では、Ansible Playbook を使用して、既存のセルフサービスルールに特定の設定を追加する方法を説明します。この例では、**ユーザーは自分の名前詳細を管理できる** のセルフサービスルールに、**surname** のメンバー属性が含まれるようにします。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。

- Ansible バージョン 2.14 以降を使用している。
- Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
- **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
- この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。

- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。
- **ユーザーが独自の名前の詳細** セルフサービスルールが IdM に存在する。

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

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

2. **/usr/share/doc/ansible-freeipa/playbooks/selfservice/** ディレクトリーにある **selfservice-member-present.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/selfservice/selfservice-member-present.yml selfservice-member-present-copy.yml
```

3. Ansible Playbook ファイル (**selfservice-member-present-copy.yml**) を開きます。
4. **ipaselfservice** タスクセクションに以下の変数を設定してファイルを調整します。
   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は、変更するセルフサービスルールの名前に設定します。
   - **attribte** 変数は **surname** に設定します。
   - **action** 変数は **member** に設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Self-service member present
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure selfservice "Users can manage their own name details" member attribute surname is present
    ipaselfservice:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: "Users can manage their own name details"
      attribute:
      - surname
      action: member
```

5. ファイルを保存します。

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

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory selfservice-member-present-copy.yml
```

関連情報

- IdM でのセルフサービスアクセス制御 を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーで利用可能な **README-selfservice.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/selfservice** ディレクトリーのサンプルの Playbook を参照してください。

21.5. ANSIBLE を使用してセルフサービスルールに固有の属性を含めないようにする手順

以下の手順では、Ansible Playbook を使用して、セルフサービスルールに特定の設定が割り当てられないようにする方法を説明します。この Playbook を使用して、セルフサービスルールで不必要なアクセス権限を付与しないようにします。この例では、**ユーザーが独自の名前の詳細を管理できる** セルフサービスルールに **givenname** と **surname** のメンバー属性が含まれないようにします。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
  - この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。
- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。
- **ユーザーが独自の名前の詳細** セルフサービスルールが IdM に存在する。

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

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

2. **/usr/share/doc/ansible-freeipa/playbooks/selfservice/** ディレクトリーにある **selfservice-member-absent.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/selfservice/selfservice-member-absent.yml selfservice-member-absent-copy.yml
```

3. Ansible Playbook ファイル (**selfservice-member-absent-copy.yml**) を開きます。

4. **ipaselfservice** タスクセクションに以下の変数を設定してファイルを調整します。

   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は、変更するセルフサービスルールの名前に設定します。
   - **属性** 変数は **givenname** および **surname** に設定します。
   - **action** 変数は **member** に設定します。
   - **state** 変数は **absent** に設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Self-service member absent
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure selfservice "Users can manage their own name details" member attributes givenname and surname are absent
    ipaselfservice:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: "Users can manage their own name details"
      attribute:
      - givenname
      - surname
      action: member
      state: absent
```

5. ファイルを保存します。

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

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory selfservice-member-absent-copy.yml
```

関連情報

- IdM でのセルフサービスアクセス制御 を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-selfservice.md** ファイルを参照してください。

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

第22章 IDM CLI でのユーザーグループの管理

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

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

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

- IdM ユーザー
- 他の IdM ユーザーグループ
- 外部ユーザー (IdM の外部に存在するユーザー)

22.1. IDM のさまざまなグループタイプ

IdM は、以下のグループタイプをサポートします。

POSIX グループ (デフォルト)

POSIX グループは、メンバーの Linux POSIX 属性に対応します。Active Directory と対話するグループは POSIX 属性を使用できないことに注意してください。
POSIX 属性は、ユーザーを個別のエンティティーとして識別します。ユーザーに関連する POSIX 属性の例には、**uidNumber** (ユーザー番号 (UID))、および **gidNumber** (グループ番号 (GID)) が含まれます。

非 POSIX グループ

非 POSIX グループは、POSIX 属性に対応していません。たとえば、これらのグループには GID が定義されていません。
このタイプのグループのすべてのメンバーは、IdM ドメインに属している必要があります。

外部グループ

外部グループを使用して、以下のような IdM ドメイン外の ID ストアに存在するグループメンバーを追加できます。

- ローカルシステム
- Active Directory ドメイン
- ディレクトリーサービス

外部グループは、POSIX 属性に対応していません。たとえば、これらのグループには GID が定義されていません。

表22.1 デフォルトで作成されたユーザーグループ

グループ名	デフォルトのグループメンバー
ipausers	すべての IdM ユーザー
admins	管理権限を持つユーザー (デフォルトの **admin** ユーザーを含む)
editors	特権のないレガシーグループ

グループ名	デフォルトのグループメンバー
trust admins	Active Directory 信頼を管理する権限を持つユーザー

ユーザーをユーザーグループに追加すると、ユーザーはグループに関連付けられた権限とポリシーを取得します。たとえば、ユーザーに管理権限を付与するには、ユーザーを **admins** グループに追加します。

警告

admins グループを削除しないでください。**admins** は IdM で必要な事前定義グループであるため、この操作では特定のコマンドで問題が生じます。

さらに、IdM で新しいユーザーが作成されるたびに、IdM は、デフォルトで **ユーザーのプライベートグループ** を作成します。プライベートグループの詳細は、プライベートグループのないユーザーの追加 を参照してください。

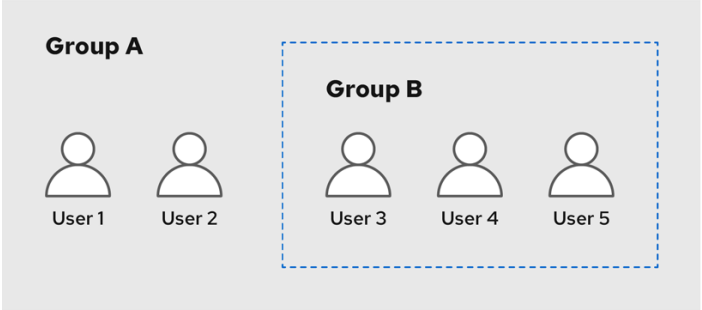
22.2. 直接および間接のグループメンバー

IdM のユーザーグループ属性は、直接メンバーと間接メンバーの両方に適用されます。グループ B がグループ A のメンバーである場合、グループ B のすべてのユーザーはグループ A の間接メンバーと見なされます。

たとえば、以下の図の場合:

- ユーザー 1 とユーザー 2 は、グループ A の **直接メンバー** です。
- ユーザー 3、ユーザー 4、およびユーザー 5 は、グループ A の **間接メンバー** です。

図22.1 直接および間接グループメンバーシップ



ユーザーグループ A にパスワードポリシーを設定すると、そのポリシーはユーザーグループ B のすべてのユーザーにも適用されます。

22.3. IDM CLI を使用したユーザーグループの追加

IdM CLI を使用してユーザーグループを追加するには、次の手順に従います。

前提条件

- 管理者としてログインしている。詳細は、Using kinit to log in to IdM manually を参照してください。

手順

- **ipa group-add group_name** コマンドを使用してユーザーグループを追加します。たとえば、group_a を作成するには、次のコマンドを実行します。

```
$ ipa group-add group_a
---------------------
Added group "group_a"
---------------------
  Group name: group_a
  GID: 1133400009
```

  デフォルトでは、**ipa group-add** は、POSIX ユーザーグループを追加します。別のグループタイプを指定するには、**ipa group-add** にオプションを追加します。

  - **--nonposix** は、非 POSIX グループを作成します。
  - **--external** は、外部グループを作成します。
    グループタイプの詳細は、IdM のさまざまなグループタイプ を参照してください。

  ユーザーグループの追加時にカスタムの GID を指定するには、**--gid=custom_GID** オプションを使用します。これを行う場合は、ID の競合を避けるように注意してください。カスタム GID を指定しない場合、IdM は使用可能な ID 範囲から GID を自動的に割り当てます。

22.4. IDM CLI を使用したユーザーグループの検索

IdM CLI を使用して既存のユーザーグループを検索するには、次の手順に従います。

手順

- **ipa group-find** コマンドを使用して、すべてのユーザーグループを表示します。グループタイプを指定するには、**ipa group-find** にオプションを追加します。
  - **ipa group-find --posix** コマンドを使用して、すべての POSIX グループを表示します。
  - **ipa group-find --nonposix** コマンドを使用して、すべての非 POSIX グループを表示します。
  - **ipa group-find --external** コマンドを使用して、すべての外部グループを表示します。
    異なるグループタイプの詳細は、IdM のさまざまなグループタイプ を参照してください。

22.5. IDM CLI を使用したユーザーグループの削除

IdM CLI を使用してユーザーグループを削除するには、には、次の手順に従います。グループを削除しても、IdM からグループメンバーは削除されないことに注意してください。

前提条件

- 管理者としてログインしている。詳細は、Using kinit to log in to IdM manually を参照してください。

手順

- **ipa group-del group_name** コマンドを使用してユーザーグループを削除します。たとえば、group_a を削除するには、次のコマンドを実行します。

```
$ ipa group-del group_a
--------------------------
Deleted group "group_a"
--------------------------
```

22.6. IDM CLI でユーザーグループにメンバーの追加

ユーザーおよびユーザーグループの両方をユーザーグループのメンバーとして追加できます。詳細は、IdM のさまざまなグループタイプ および 直接および間接のグループメンバー を参照してください。IdM CLI を使用してユーザーグループにメンバーを追加するには、次の手順に従います。

前提条件

- 管理者としてログインしている。詳細は、Using kinit to log in to IdM manually を参照してください。

手順

- **ipa group-add-member** コマンドを使用して、ユーザーグループにメンバーを追加します。次のオプションを使用して、メンバーのタイプを指定します。
  - **--users** は、IdM ユーザーを追加します
  - **--external** は、**DOMAIN\user_name** 形式または **user_name@domain** 形式で、IdM ドメイン外に存在するユーザーを追加します
  - **--groups** は、IdM ユーザーグループを追加します。

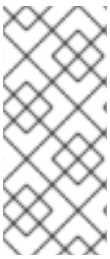
  たとえば、group_b を group_a のメンバーとして追加するには、次のコマンドを実行します。

```
$ ipa group-add-member group_a --groups=group_b
Group name: group_a
GID: 1133400009
Member users: user_a
Member groups: group_b
Indirect Member users: user_b
-------------------------
Number of members added 1
-------------------------
```

  group_b のメンバーは、group_a の間接メンバーになりました。

重要

グループを別のグループのメンバーとして追加する場合は、再帰グループを作成しないでください。たとえば、グループ A がグループ B のメンバーである場合は、グループ B をグループ A のメンバーとして追加しないでください。再帰的なグループにより予期しない動作が発生する可能性があります。



注記

ユーザーグループにメンバーを追加した後、更新が Identity Management 環境のすべてのクライアントに広がるまでに時間がかかる場合があります。これは、特定のホストがユーザー、グループ、およびネットグループを解決するときに、**System Security Services Daemon** (SSSD) が最初にキャッシュを調べて、サーバーで不足または期限切れのレコードのみを検索するためです。

22.7. ユーザープライベートグループなしでユーザーの追加

デフォルトでは、IdM で新しいユーザーが作成されるたびに、IdM がユーザーのプライベートグループ (UPG) を作成します。UPG は特定のグループタイプです。

- UPG の名前は、新しく作成されたユーザーと同じです。
- ユーザーは UPG の唯一のメンバーです。UPG には他のメンバーを含めることができません。
- プライベートグループの GID は、ユーザーの UID と一致します。

ただし、UPG を作成せずにユーザーを追加することは可能です。

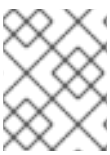
22.7.1. ユーザープライベートグループのないユーザー

NIS グループまたは別のシステムグループが、ユーザープライベートグループに割り当てられる GID をすでに使用している場合は、UPG を作成しないようにする必要があります。

これは、以下の 2 つの方法で実行できます。

- プライベートグループをグローバルに無効にすることなく、UPG なしで新しいユーザーを追加する。プライベートグループがグローバルに有効になっている場合にユーザープライベートグループのないユーザーを追加 を参照してください。
- すべてのユーザーに対して UPG をグローバルに無効にしてから、新しいユーザーを追加する。すべてのユーザーに対してユーザープライベートグループをグローバルで無効にする および ユーザープライベートグループをグローバルで無効にしている時のユーザーの追加 を参照してください。

どちらの場合も、IdM では、新しいユーザーを追加するときに GID を指定する必要があります。指定しないと、操作は失敗します。これは、IdM には新しいユーザーの GID が必要ですが、デフォルトのユーザーグループ **ipausers** は非 POSIX グループであるため、関連付けられた GID がないためです。指定する GID は、既存のグループに対応する必要がありません。



注記

GID を指定しても、新しいグループは作成されません。この属性は IdM に必要であるため、新しいユーザーに GID 属性のみを設定します。

22.7.2. プライベートグループがグローバルに有効になっている場合にユーザープライベートグループのないユーザーを追加

システムで UPG が有効になっている場合でも、ユーザープライベートグループ (UPG) を作成せずにユーザーを追加できます。これには、新しいユーザーの GID を手動で設定する必要があります。これが必要な理由の詳細については、ユーザープライベートグループのないユーザー を参照してください。

手順

- IdM が UPG を作成しないようにするには、**ipa user-add** コマンドに **--noprivate** オプションを追加します。
  コマンドを成功させるには、カスタム GID を指定する必要があることに注意してください。たとえば、GID 10000 の新しいユーザーを追加するには、次のコマンドを実行します。

```
$ ipa user-add jsmith --first=John --last=Smith --noprivate --gid 10000
```

22.7.3. すべてのユーザーに対してユーザープライベートグループをグローバルに無効にする

ユーザープライベートグループ (UPG) をグローバルに無効にできます。これにより、すべての新しいユーザーの UPG が作成されなくなります。既存のユーザーはこの変更の影響を受けません。

手順

1. 管理者権限を取得します。

```
$ kinit admin
```

2. IdM は、Directory Server Managed Entries プラグインを使用して UPG を管理します。プラグインのインスタンスをリスト表示します。

```
$ ipa-managed-entries --list
```

3. IdM が UPG を作成しないようにするには、ユーザープライベートグループを管理するプラグインインスタンスを無効にします。

```
$ ipa-managed-entries -e "UPG Definition" disable
Disabling Plugin
```

注記

後で **UPG 定義** インスタンスを再有効にするには、**ipa-managed-entries -e "UPG Definition" enable** コマンドを使用します。

4. Directory Server を再起動して、新しい設定を読み込みます。

```
$ sudo systemctl restart dirsrv.target
```

UPG が無効になった後にユーザーを追加するには、GID を指定する必要があります。詳細は、ユーザープライベートグループがグローバルで無効になっている場合のユーザーの追加 を参照してください。

検証手順

- UPG がグローバルで無効になっているかどうかを確認するには、再度 disable コマンドを使用します。

```
$ ipa-managed-entries -e "UPG Definition" disable
Plugin already disabled
```

22.7.4. ユーザープライベートグループがグローバルで無効になっている場合のユーザーの追加

ユーザープライベートグループ (UPG) がグローバルで無効になっている場合、IdM は GID を新しいユーザーに自動的に割り当てません。ユーザーを正常に追加するには、GID を手動で割り当てるか、automember を使用して割り当てる必要があります。これが必要な理由の詳細については、Users without a user private group を参照してください。

前提条件

- UPG は、すべてのユーザーに対してグローバルに無効にする必要があります。詳細は、すべてのユーザーに対してユーザープライベートグループをグローバルに無効にする をご覧ください。

手順

- UPG の作成が無効になっているときに新しいユーザーの追加が成功することを確認するには、次のいずれかを選択します。
  - 新しいユーザーを追加するときにカスタムの GID を指定します。GID は、既存のユーザーグループに対応する必要はありません。
    たとえば、コマンドラインからユーザーを追加する場合は、**--gid** オプションを **ipa user-add** コマンドに追加します。
  - automember ルールを使用して、GID のある既存のグループにユーザーを追加します。

22.8. IDM CLI を使用して IDM ユーザーグループにメンバーマネージャーとしてユーザーまたはグループを追加する手順

IdM CLI を使用してユーザーまたはグループをメンバーマネージャーとして IdM ユーザーグループに追加するには、次の手順に従います。メンバーマネージャーは、ユーザーまたはグループを IdM ユーザーグループに追加できますが、グループの属性を変更できません。

前提条件

- 管理者としてログインしている。詳細は、Using kinit to log in to IdM manually を参照してください。
- メンバーマネージャーとして追加するユーザーまたはグループの名前と、メンバーマネージャーが管理するグループ名を用意する。

手順

- **ipa group-add-member-manager** コマンドを使用して、ユーザーをメンバーマネージャーとして IdM ユーザーグループに追加します。
  たとえば、ユーザー **test** を **group_a** のメンバーマネージャーとして追加します。

```
$ ipa group-add-member-manager group_a --users=test
Group name: group_a
GID: 1133400009
Membership managed by users: test
-------------------------
Number of members added 1
-------------------------
```

ユーザー **test** は **group_a** のメンバーを管理できるようになりました。

- **ipa group-add-member-manager** コマンドを使用して、グループをメンバーマネージャーとして IdM ユーザーグループに追加します。
たとえば、 グループ **group_admins** を **group_a** のメンバーマネージャーとして追加します。

```
$ ipa group-add-member-manager group_a --groups=group_admins
Group name: group_a
GID: 1133400009
Membership managed by groups: group_admins
Membership managed by users: test
-------------------------
Number of members added 1
-------------------------
```

グループ **group_admins** は **group_a** のメンバーを管理できるようになりました。

注記

ユーザーグループにメンバーマネージャーを追加してから、更新が Identity Management 環境のすべてのクライアントに広がるまでに時間がかかる場合があります。

検証手順

- **ipa group-show** コマンドを使用して、ユーザーおよびグループがメンバーマネージャーとして追加されたことを確認します。

```
$ ipa group-show group_a
Group name: group_a
GID: 1133400009
Membership managed by groups: group_admins
Membership managed by users: test
```

関連情報

- 詳細は、**ipa group-add-member-manager --help** を参照してください。

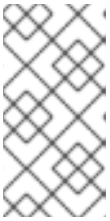
22.9. IDM CLI を使用したグループメンバーの表示

IdM CLI を使用してグループのメンバーを表示するには、次の手順に従います。直接グループメンバーと間接グループメンバーの両方を表示できます。詳細は、直接および間接のグループメンバー を参照してください。

手順

- グループのメンバーのリストを表示するには、**ipa group-show group_name** コマンドを使用します。以下に例を示します。

```
$ ipa group-show group_a
  ...
  Member users: user_a
  Member groups: group_b
  Indirect Member users: user_b
```



注記

間接メンバーのリストには、信頼された Active Directory ドメインの外部ユーザーが含まれません。Active Directory 信頼ユーザーオブジェクトは、Identity Management 内に LDAP オブジェクトとして存在しないため、Identity Management インターフェイスには表示されません。

22.10. IDM CLI を使用してユーザーグループからメンバーを削除

IdM CLI を使用してユーザーグループからメンバーを削除するには、次の手順に従います。

前提条件

- 管理者としてログインしている。詳細は、Using kinit to log in to IdM manually を参照してください。

手順

1. **オプション。ipa group-show** コマンドを使用して、削除するメンバーがグループに含まれていることを確認します。
2. **ipa group-remove-member** コマンドを使用して、ユーザーグループからメンバーを削除します。
   以下のオプションを使用して、削除するメンバーを指定します。
   - **--users** は、IdM ユーザーを削除します
   - **--external** は、**DOMAIN\user_name** または **user_name@domain** の形式で、IdM ドメイン外に存在するユーザーを削除します
   - **--groups** は、IdM ユーザーグループを削除します

   たとえば、**group_name** という名前のグループから、**user1**、**user2**、および **group1** を削除するには、次のコマンドを実行します。

```
$ ipa group-remove-member group_name --users=user1 --users=user2 --groups=group1
```

22.11. IDM CLI を使用してユーザーまたはグループを IDM ユーザーグループのメンバーマネージャーから取り消す手順

IdM CLI を使用して IdM ユーザーグループのメンバーマネージャーからユーザーまたはグループを削除するには、次の手順に従います。メンバーマネージャーは、IdM ユーザーグループからユーザーまたはグループを削除できますが、グループの属性を変更できません。

前提条件

- 管理者としてログインしている。詳細は、Using kinit to log in to IdM manually を参照してください。
- 削除する既存のメンバーマネージャーのユーザーまたはグループの名前と、そのメンバーマネージャーが管理するグループ名が必要です。

手順

- **ipa group-remove-member-manager** コマンドを使用してユーザーを IdM ユーザーグループのメンバーマネージャーから削除します。
  たとえば、ユーザー **test** を **group_a** のメンバーマネージャーから削除します。

```
$ ipa group-remove-member-manager group_a --users=test
Group name: group_a
GID: 1133400009
Membership managed by groups: group_admins
-------------------------
Number of members removed 1
-------------------------
```

  ユーザー **test** は **group_a** のメンバーを管理できなくなりました。

- **ipa group-remove-member-manager** コマンドを使用してグループを IdM ユーザーグループのメンバーマネージャーから削除します。
  たとえば、 グループ **group_admins** を **group_a** のメンバーマネージャーから削除します。

```
$ ipa group-remove-member-manager group_a --groups=group_admins
Group name: group_a
GID: 1133400009
-------------------------
Number of members removed 1
-------------------------
```

  グループ **group_admins** は **group_a** のメンバーを管理できなくなりました。

注記

ユーザーグループからメンバーマネージャーを削除してから、更新が Identity Management 環境のすべてのクライアントに広がるまでに時間がかかる場合があります。

検証手順

- **ipa group-show** コマンドを使用して、ユーザーおよびグループがメンバーマネージャーから削除されたことを確認します。

```
$ ipa group-show group_a
Group name: group_a
GID: 1133400009
```

関連情報

- 詳細は、**ipa group-remove-member-manager --help** を参照してください。

22.12. IDM でのローカルグループとリモートグループのグループマージの有効化

グループは、Identity Management (IdM) や Active Directory (AD) などのドメインによって提供されて一元管理されるか、ローカルシステムの **etc/group** ファイルで管理されます。ほとんどの場合、ユーザーは一元管理されたストアに依存しています。しかし、ソフトウェアによっては、現在も既知のグループのメンバーシップに基づいてアクセス制御を管理している場合があります。

ドメインコントローラーおよびローカルの **etc/group** ファイルのグループを管理する場合は、グループのマージを有効にすることができます。ローカルファイルとリモートサービスの両方を確認するように **nsswitch.conf** ファイルを設定できます。グループが両方に存在する場合、メンバーユーザーのリストが結合され、単一の応答で返されます。

以下の手順では、ユーザー **idmuser** のグループのマージを有効にする方法について説明します。

手順

1. **/etc/nsswitch.conf** ファイルに **[SUCCESS=merge]** を追加します。

```
# Allow initgroups to default to the setting for group.
initgroups: sss [SUCCESS=merge] files
```

2. **idmuser** を IdM に追加します。

```
# ipa user-add idmuser
First name: idm
Last name: user
----------------------
Added user "idmuser"
----------------------
User login: idmuser
First name: idm
Last name: user
Full name: idm user
Display name: idm user
Initials: tu
Home directory: /home/idmuser
GECOS: idm user
Login shell: /bin/sh
Principal name: idmuser@IPA.TEST
Principal alias: idmuser@IPA.TEST
Email address: idmuser@ipa.test
UID: 19000024
GID: 19000024
Password: False
Member of groups: ipausers
Kerberos keys available: False
```

3. ローカルの **audio** グループの GID を確認します。

```
$ getent group audio
--------------------
audio:x:63
```

4. **audio** グループを IdM に追加します。

```
$ ipa group-add audio --gid 63
-------------------
Added group "audio"
-------------------
Group name: audio
GID: 63
```

注記

audio グループを IdM に追加するときに定義する GID は、ローカルの **audio** グループの GID と同じである必要があります。

5. **idmuser** ユーザーを IdM の **audio** グループに追加します。

```
$ ipa group-add-member audio --users=idmuser
Group name: audio
GID: 63
Member users: idmuser
-------------------------
Number of members added 1
-------------------------
```

検証

1. **idmuser** としてログインします。
2. **idmuser** のセッションにローカルグループがあることを確認します。

```
$ id idmuser
uid=1867800003(idmuser) gid=1867800003(idmuser)
groups=1867800003(idmuser),63(audio),10(wheel)
```

22.13. ANSIBLE を使用して、IDM クライアントのローカルサウンドカードへのユーザー ID オーバーライドアクセス権を付与する

ansible-freeipa group および **idoverrideuser** モジュールを使用して、Identity Management (IdM) または Active Directory (AD) ユーザーを IdM クライアント上の **audio** ローカルグループのメンバーにすることができます。これにより、IdM または AD ユーザーに、ホスト上のサウンドカードへの特権アクセスが付与されます。この手順で使用する例では、最初の Playbook タスクで **Default Trust View** ID ビューに **aduser@addomain.com** ID オーバーライドを追加します。次の Playbook タスクで、RHEL ホスト上の **audio** ローカルグループの GID に対応する GID 63 の **audio** グループを IdM に作成します。同時に、aduser@addomain.com ID オーバーライドを IdM オーディオグループにメンバーとして追加します。

前提条件

- 手順の最初の部分を実行する対象である IdM クライアントへの **root** アクセス権を持っている。この例では、これは **client.idm.example.com** です。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - RHEL 9.4 以降を使用している。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
  - この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。
- AD フォレストが IdM と信頼関係にある。この例では、AD ドメインの名前は **addomain.com** であり、ローカルグループ **audio** に存在することを確認する AD ユーザーの完全修飾ドメイン名 (FQDN) は **aduser@addomain.com** です。
- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。

手順

1. **client.idm.example.com** で、**/etc/nsswitch.conf** ファイルに **[SUCCESS=merge]** を追加します。

```
[...]
# Allow initgroups to default to the setting for group.
initgroups: sss [SUCCESS=merge] files
```

2. **audio** ローカルグループの GID を特定します。

```
$ getent group audio
----------------------
audio:x:63
```

3. Ansible コントロールノードで、**aduser@addomain.com** ユーザーオーバーライドを Default Trust View に追加するタスクを含む **add-aduser-to-audio-group.yml** Playbook を作成します。

```
---
- name: Playbook to manage idoverrideuser
  hosts: ipaserver
  become: false

  tasks:
  - name: Add aduser@addomain.com user to the Default Trust View
    ipaidoverrideuser:
      ipaadmin_password: "{{ ipaadmin_password }}"
      idview: "Default Trust View"
      anchor: aduser@addomain.com
```

4. 同じ Playbook 内の別の Playbook タスクを使用して、**GID** 63 を持つグループ **audio** を IdM に追加します。aduser idoverrideuser をグループに追加します。

```
- name: Add the audio group with the aduser member and GID of 63
  ipagroup:
    ipaadmin_password: "{{ ipaadmin_password }}"
    name: audio
    idoverrideuser:
    - aduser@addomain.com
    gidnumber: 63
```

5. ファイルを保存します。

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

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory add-aduser-to-audio-group.yml
```

検証

1. AD ユーザーとして IdM クライアントにログインします。

```
$ ssh aduser@addomain.com@client.idm.example.com
```

2. AD ユーザーのグループメンバーシップを確認します。

```
$ id aduser@addomain.com
uid=702801456(aduser@addomain.com) gid=63(audio) groups=63(audio)
```

関連情報

- idoverrideuser および ipagroup に関する **ansible-freeipa** アップストリームドキュメント
- IdM でのローカルグループとリモートグループのグループマージの有効化

第23章 IDM WEG UI でのユーザーグループの管理

本章では、IdM Web UI を使用したユーザーグループ管理を説明します。

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

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

- IdM ユーザー
- 他の IdM ユーザーグループ
- 外部ユーザー (IdM の外部に存在するユーザー)

23.1. IDM のさまざまなグループタイプ

IdM は、以下のグループタイプをサポートします。

POSIX グループ (デフォルト)

POSIX グループは、メンバーの Linux POSIX 属性に対応します。Active Directory と対話するグループは POSIX 属性を使用できないことに注意してください。
POSIX 属性は、ユーザーを個別のエンティティーとして識別します。ユーザーに関連する POSIX 属性の例には、**uidNumber** (ユーザー番号 (UID))、および **gidNumber** (グループ番号 (GID)) が含まれます。

非 POSIX グループ

非 POSIX グループは、POSIX 属性に対応していません。たとえば、これらのグループには GID が定義されていません。
このタイプのグループのすべてのメンバーは、IdM ドメインに属している必要があります。

外部グループ

外部グループを使用して、以下のような IdM ドメイン外の ID ストアに存在するグループメンバーを追加できます。

- ローカルシステム
- Active Directory ドメイン
- ディレクトリーサービス

外部グループは、POSIX 属性に対応していません。たとえば、これらのグループには GID が定義されていません。

表23.1 デフォルトで作成されたユーザーグループ

グループ名	デフォルトのグループメンバー
ipausers	すべての IdM ユーザー
admins	管理権限を持つユーザー (デフォルトの **admin** ユーザーを含む)
editors	特権のないレガシーグループ

グループ名	デフォルトのグループメンバー
trust admins	Active Directory 信頼を管理する権限を持つユーザー

ユーザーをユーザーグループに追加すると、ユーザーはグループに関連付けられた権限とポリシーを取得します。たとえば、ユーザーに管理権限を付与するには、ユーザーを **admins** グループに追加します。

警告

admins グループを削除しないでください。**admins** は IdM で必要な事前定義グループであるため、この操作では特定のコマンドで問題が生じます。

さらに、IdM で新しいユーザーが作成されるたびに、IdM は、デフォルトで **ユーザーのプライベートグループ** を作成します。プライベートグループの詳細は、プライベートグループのないユーザーの追加 を参照してください。

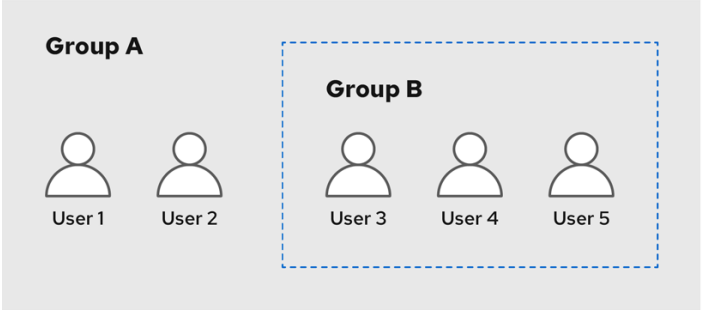
23.2. 直接および間接のグループメンバー

IdM のユーザーグループ属性は、直接メンバーと間接メンバーの両方に適用されます。グループ B がグループ A のメンバーである場合、グループ B のすべてのユーザーはグループ A の間接メンバーと見なされます。

たとえば、以下の図の場合:

- ユーザー 1 とユーザー 2 は、グループ A の **直接メンバー** です。
- ユーザー 3、ユーザー 4、およびユーザー 5 は、グループ A の **間接メンバー** です。

図23.1 直接および間接グループメンバーシップ



640_RHEL_0424

ユーザーグループ A にパスワードポリシーを設定すると、そのポリシーはユーザーグループ B のすべてのユーザーにも適用されます。

23.3. IDM WEB UI を使用したユーザーグループの追加

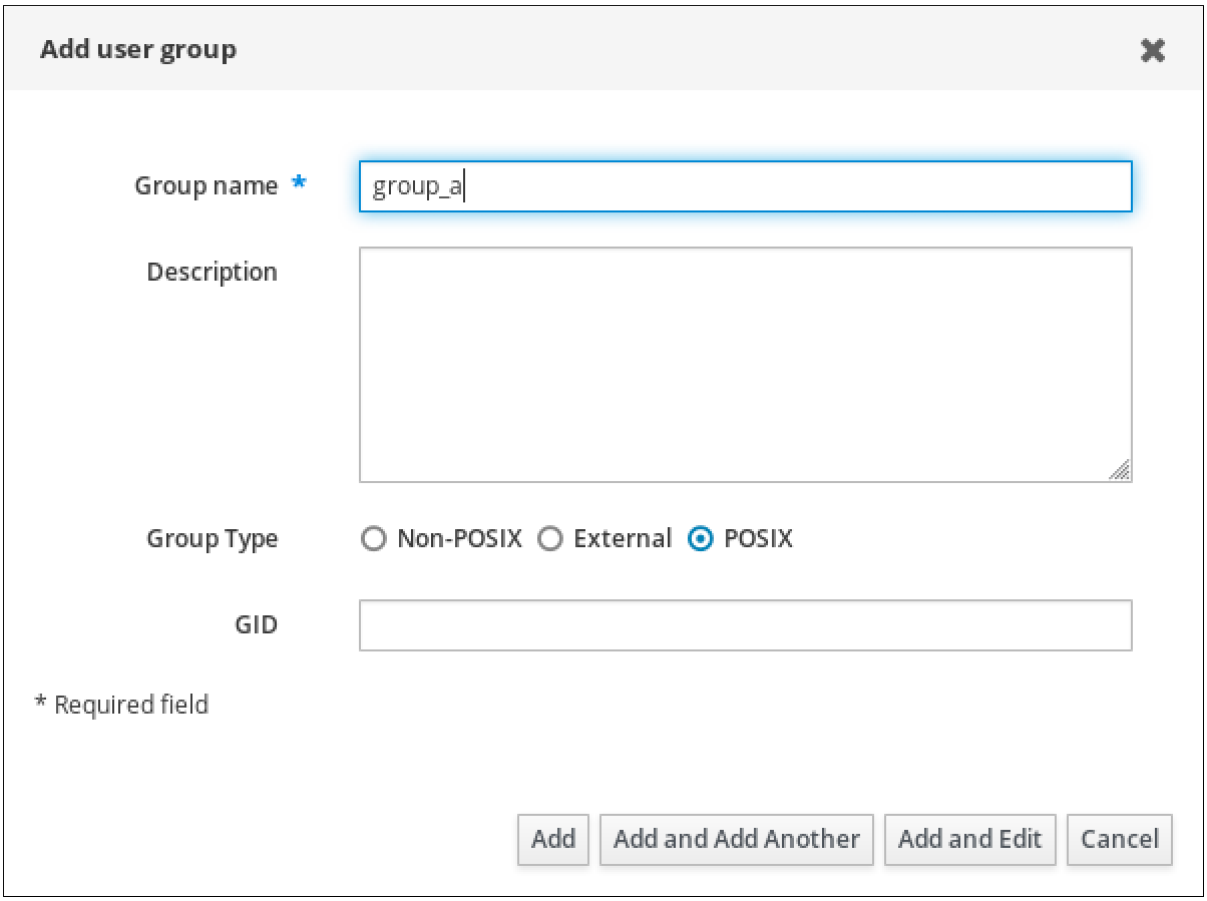
IdM Web UI を使用してユーザーグループを追加するには、次の手順に従います。

前提条件

- IdM Web UI にログインしている。

手順

1. **Identity → Groups** の順にクリックし、左のサイドバーで **User Groups** を選択します。
2. **Add** をクリックして、グループの追加を開始します。
3. グループの情報を入力します。ユーザーグループタイプの詳細は、IdM のさまざまなグループタイプ を参照してください。
   グループのカスタム GID を指定できます。これを行う場合は、ID の競合を避けるように注意してください。カスタム GID を指定しない場合、IdM は使用可能な ID 範囲から GID を自動的に割り当てます。



4. **Add** をクリックして確定します。

23.4. IDM WEB UI を使用したユーザーグループの削除

IdM Web UI を使用してユーザーグループを削除するには、次の手順に従います。グループを削除しても、IdM からグループメンバーは削除されないことに注意してください。

前提条件

- IdM Web UI にログインしている。

手順

1. **Identity → Groups** の順にクリックし、**User Groups** を選択します。
2. 削除するグループを選択します。
3. **Delete** をクリックします。
4. **Delete** をクリックして確定します。

23.5. IDM WEB UI でユーザーグループにメンバーの追加

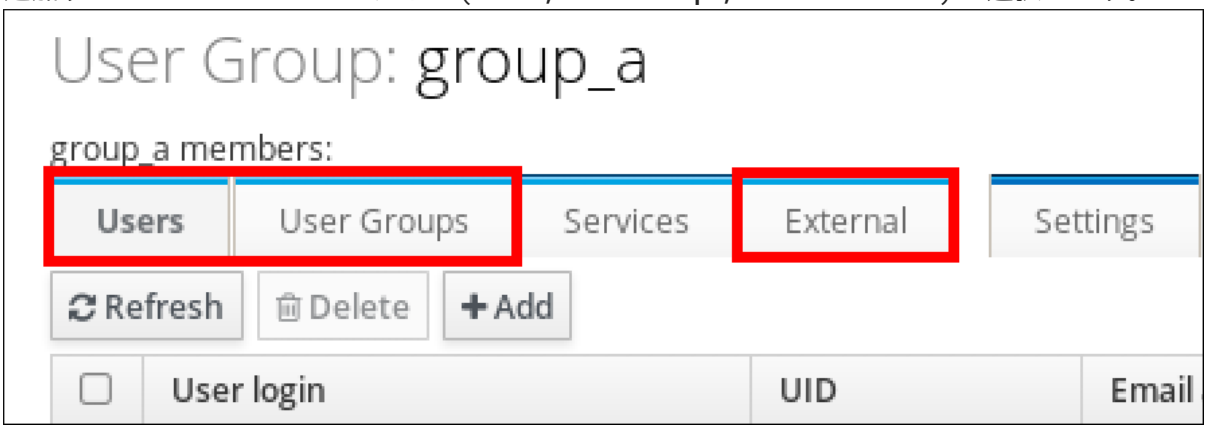
ユーザーおよびユーザーグループの両方をユーザーグループのメンバーとして追加できます。詳細は、IdM のさまざまなグループタイプ および 直接および間接のグループメンバー を参照してください。

前提条件

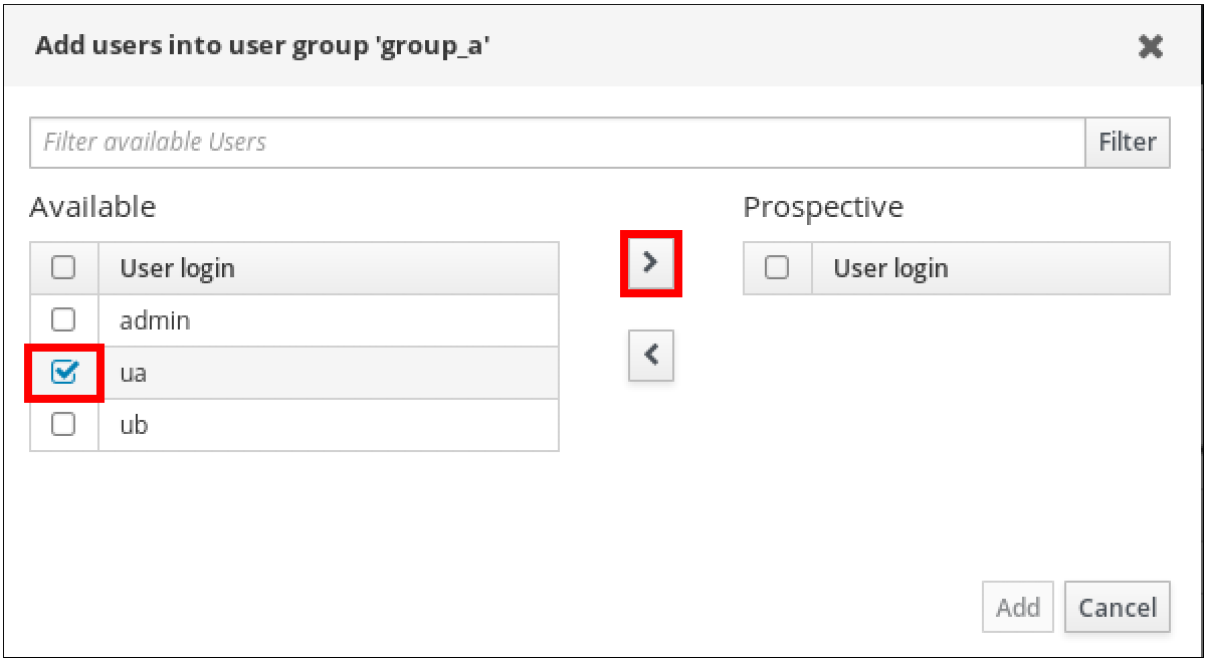
- IdM Web UI にログインしている。

手順

1. **Identity → Groups** の順にクリックし、左のサイドバーで **User Groups** を選択します。
2. グループ名をクリックします。
3. 追加するグループメンバーのタイプ (**Users**, **User Groups**, または **External**) を選択します。



4. **Add** をクリックします。
5. 追加するメンバーの横にあるチェックボックスを 1 つ以上選択します。
6. 右矢印をクリックして、選択したメンバーをグループに移動します。



7. **Add** をクリックして確定します。

23.6. WEB UI を使用して IDM ユーザーグループにメンバーマネージャーとしてユーザーまたはグループを追加する手順

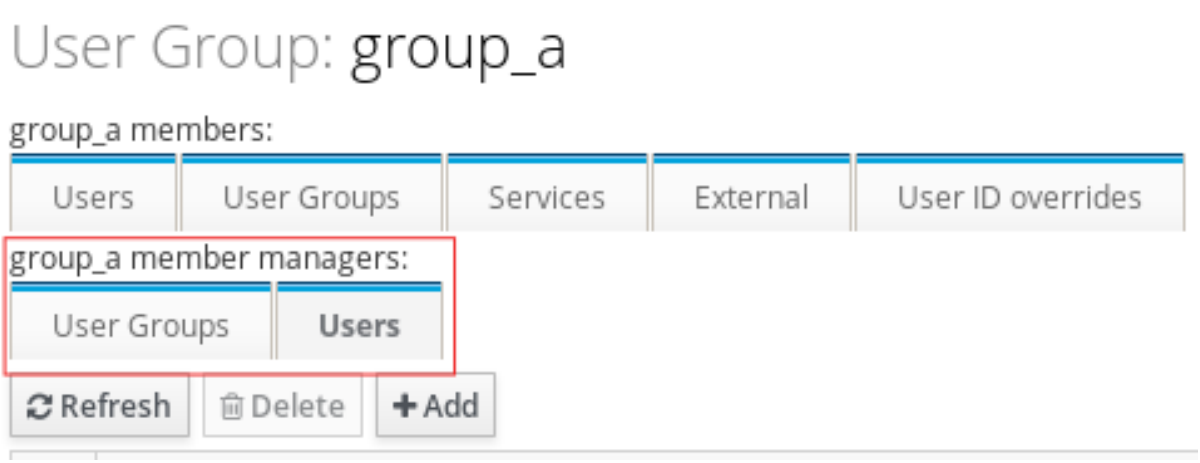
Web UI を使用してユーザーまたはグループをメンバーマネージャーとして IdM ユーザーグループに追加するには、次の手順に従います。メンバーマネージャーは、ユーザーまたはグループを IdM ユーザーグループに追加できますが、グループの属性を変更できません。

前提条件

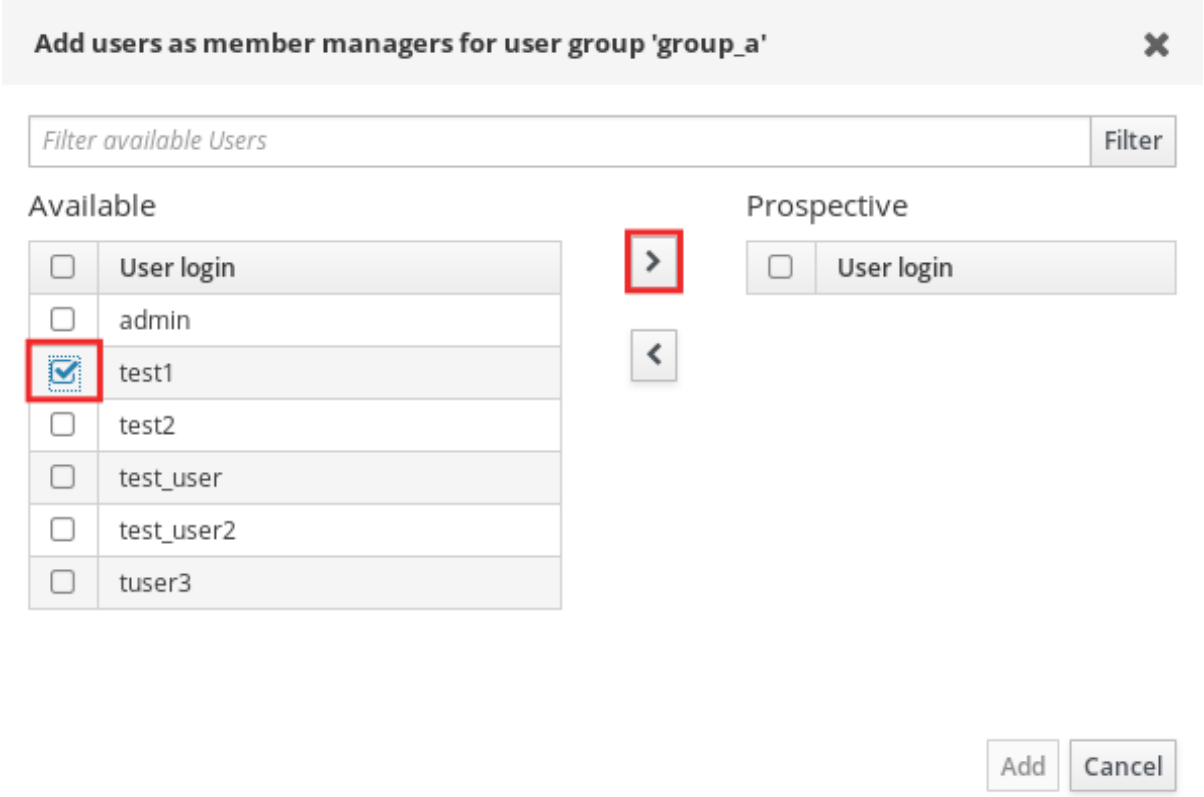
- IdM Web UI にログインしている。
- メンバーマネージャーとして追加するユーザーまたはグループの名前と、メンバーマネージャーが管理するグループ名を用意する。

手順

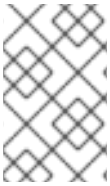
1. **Identity → Groups** の順にクリックし、左のサイドバーで **User Groups** を選択します。
2. グループ名をクリックします。
3. 追加するグループメンバーマネージャーのタイプ (**Users** または **User Groups**) を選択します。



4. **Add** をクリックします。
5. 追加するメンバーの横にあるチェックボックスを 1 つ以上選択します。
6. 右矢印をクリックして、選択したメンバーをグループに移動します。



7. **Add** をクリックして確定します。

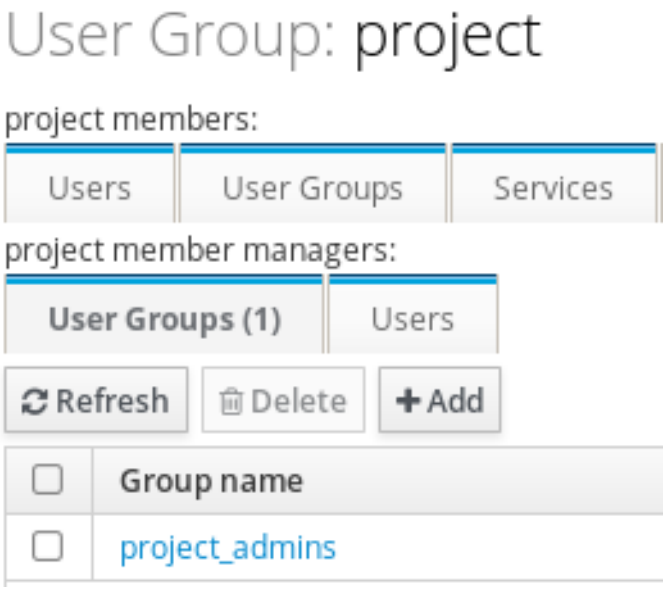


注記

ユーザーグループにメンバーマネージャーを追加してから、更新が Identity Management 環境のすべてのクライアントに広がるまでに時間がかかる場合があります。

検証手順

- 新たに追加したユーザーまたはユーザーグループが、ユーザーまたはユーザーグループのメンバーマネージャーのリストに追加されていることを確認します。



関連情報

- 詳細は、**ipa group-add-member-manager --help** を参照してください。

23.7. IDM WEB UI を使用したグループメンバーの表示

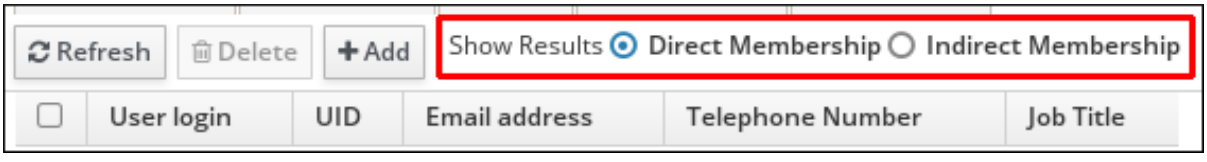
IdM Web UI を使用してグループのメンバーを表示するには、次の手順に従います。直接グループメンバーと間接グループメンバーの両方を表示できます。詳細は、直接および間接のグループメンバー を参照してください。

前提条件

- IdM Web UI にログインしている。

手順

1. **Identity → Groups** を選択します。
2. 左のサイドバーで **User Groups** を選択します。
3. 表示するグループの名前をクリックします。
4. **直接メンバーシップ** と **間接メンバーシップ** を切り替えます。



23.8. IDM WEB UI を使用してユーザーグループからメンバーの削除

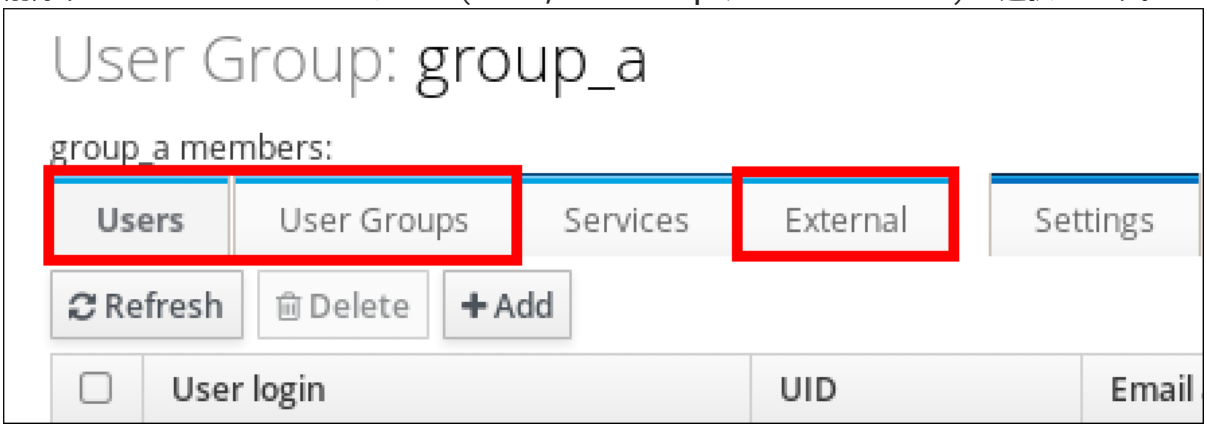
IdM Web UI を使用してユーザーグループからメンバーを削除するには、次の手順に従います。

前提条件

- IdM Web UI にログインしている。

手順

1. **Identity → Groups** の順にクリックし、左のサイドバーで **User Groups** を選択します。
2. グループ名をクリックします。
3. 削除するグループメンバーのタイプ (**Users, User Groups**、または **External**) を選択します。



4. 削除するメンバーの横にあるチェックボックスを選択します。
5. **Delete** をクリックします。
6. **Delete** をクリックして確定します。

23.9. WEB UI を使用してユーザーまたはグループを IDM ユーザーグループのメンバーマネージャーから取り消す手順

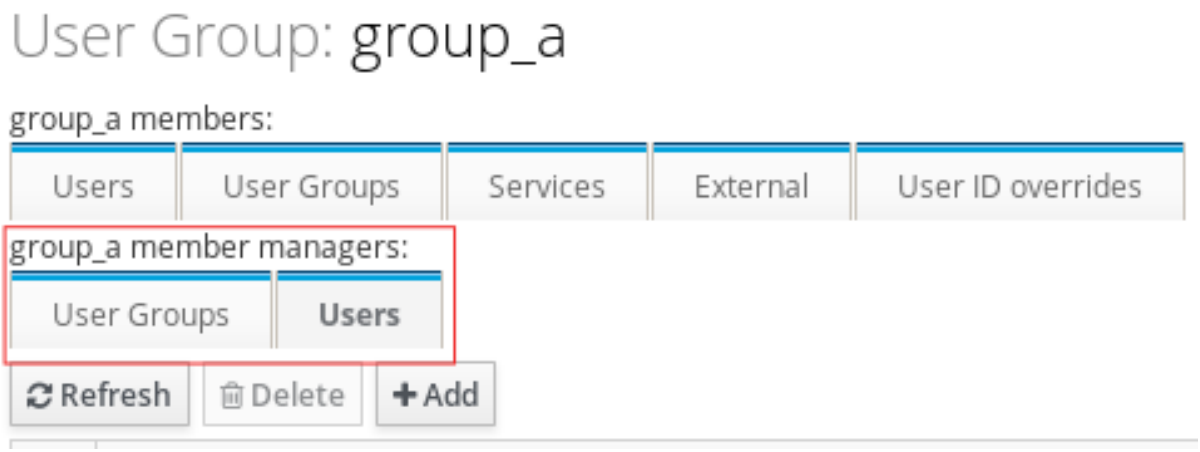
Web UI を使用して IdM ユーザーグループのメンバーマネージャーからユーザーまたはグループを削除するには、次の手順に従います。メンバーマネージャーは、IdM ユーザーグループからユーザーまたはグループを削除できますが、グループの属性を変更できません。

前提条件

- IdM Web UI にログインしている。
- 削除する既存のメンバーマネージャーのユーザーまたはグループの名前と、そのメンバーマネージャーが管理するグループ名が必要です。

手順

1. **Identity → Groups** の順にクリックし、左のサイドバーで **User Groups** を選択します。
2. グループ名をクリックします。
3. 削除するメンバーマネージャーのタイプ (**Users** または **User Groups**) を選択します。



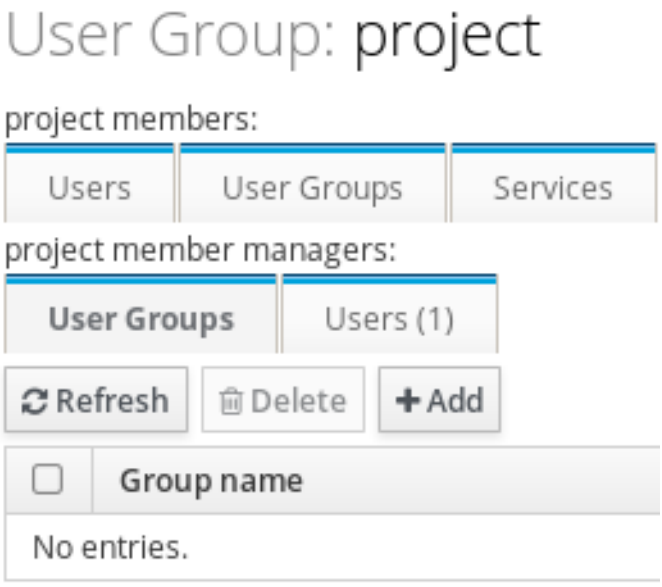
4. 削除するメンバーマネージャーの横にあるチェックボックスを選択します。
5. **Delete** をクリックします。
6. **Delete** をクリックして確定します。

注記

ユーザーグループからメンバーマネージャーを削除してから、更新が Identity Management 環境のすべてのクライアントに広がるまでに時間がかかる場合があります。

検証手順

- ユーザーまたはユーザーグループが、ユーザーまたはユーザーグループのメンバーマネージャーリストから削除されていることを確認します。



関連情報

- 詳細は、**ipa group-add-member-manager --help** を参照してください。

第24章 ANSIBLE PLAYBOOK を使用したユーザーグループの管理

本セクションでは、Ansible Playbook を使用したユーザーグループの管理について紹介します。

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

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

- IdM ユーザー
- 他の IdM ユーザーグループ
- 外部ユーザー (IdM の外部に存在するユーザー)

このセクションには、以下のトピックが含まれます。

- IdM のさまざまなグループタイプ
- 直接および間接のグループメンバー
- Ansible Playbook を使用して IdM グループおよびグループメンバーの存在を確認する
- Ansible を使用して AD ユーザーが IdM を管理できるようにする手順
- Ansible Playbook を使用して IDM ユーザーグループにメンバーマネージャーを存在させる手順
- Ansible Playbook を使用して IDM ユーザーグループにメンバーマネージャーを存在させないようにする手順

24.1. IDM のさまざまなグループタイプ

IdM は、以下のグループタイプをサポートします。

POSIX グループ (デフォルト)

POSIX グループは、メンバーの Linux POSIX 属性に対応します。Active Directory と対話するグループは POSIX 属性を使用できないことに注意してください。
POSIX 属性は、ユーザーを個別のエンティティーとして識別します。ユーザーに関連する POSIX 属性の例には、**uidNumber** (ユーザー番号 (UID))、および **gidNumber** (グループ番号 (GID)) が含まれます。

非 POSIX グループ

非 POSIX グループは、POSIX 属性に対応していません。たとえば、これらのグループには GID が定義されていません。
このタイプのグループのすべてのメンバーは、IdM ドメインに属している必要があります。

外部グループ

外部グループを使用して、以下のような IdM ドメイン外の ID ストアに存在するグループメンバーを追加できます。

- ローカルシステム
- Active Directory ドメイン
- ディレクトリーサービス

外部グループは、POSIX 属性に対応していません。たとえば、これらのグループには GID が定義されていません。

表24.1 デフォルトで作成されたユーザーグループ

グループ名	デフォルトのグループメンバー
ipausers	すべての IdM ユーザー
admins	管理権限を持つユーザー (デフォルトの **admin** ユーザーを含む)
editors	特権のないレガシーグループ
trust admins	Active Directory 信頼を管理する権限を持つユーザー

ユーザーをユーザーグループに追加すると、ユーザーはグループに関連付けられた権限とポリシーを取得します。たとえば、ユーザーに管理権限を付与するには、ユーザーを **admins** グループに追加します。

警告

admins グループを削除しないでください。**admins** は IdM で必要な事前定義グループであるため、この操作では特定のコマンドで問題が生じます。

さらに、IdM で新しいユーザーが作成されるたびに、IdM は、デフォルトで **ユーザーのプライベートグループ** を作成します。プライベートグループの詳細は、プライベートグループのないユーザーの追加 を参照してください。

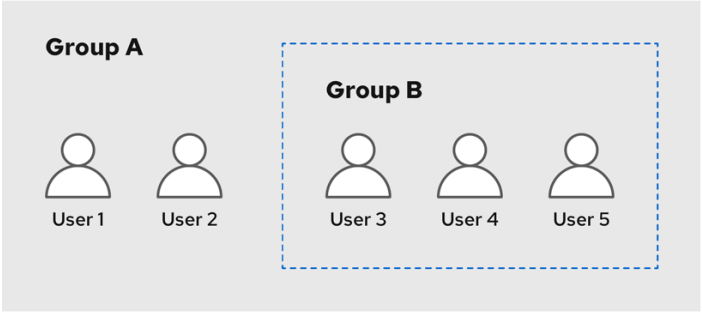
24.2. 直接および間接のグループメンバー

IdM のユーザーグループ属性は、直接メンバーと間接メンバーの両方に適用されます。グループ B がグループ A のメンバーである場合、グループ B のすべてのユーザーはグループ A の間接メンバーと見なされます。

たとえば、以下の図の場合:

- ユーザー 1 とユーザー 2 は、グループ A の **直接メンバー** です。
- ユーザー 3、ユーザー 4、およびユーザー 5 は、グループ A の **間接メンバー** です。

図24.1 直接および間接グループメンバーシップ



ユーザーグループ A にパスワードポリシーを設定すると、そのポリシーはユーザーグループ B のすべてのユーザーにも適用されます。

24.3. ANSIBLE PLAYBOOK を使用して IDM グループおよびグループメンバーの存在を確認する

以下の手順では、Ansible Playbook を使用して、IdM グループとグループメンバー (ユーザーとユーザーグループの両方) を存在させる方法を説明します。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
  - この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。
- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。
- Ansible Playbook で参照するユーザーが IdM に存在する。Ansible を使用してユーザーの存在を確認する方法は、Ansible Playbook を使用したユーザーアカウントの管理 を参照してください。

手順

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

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

2. 必要なユーザーおよびグループ情報を使用して Ansible Playbook ファイルを作成します。

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

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Create group ops with gid 1234
    ipagroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: ops
      gidnumber: 1234

  - name: Create group sysops
    ipagroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: sysops
      user:
      - idm_user

  - name: Create group appops
    ipagroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: appops

  - name: Add group members sysops and appops to group ops
    ipagroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: ops
      group:
      - sysops
      - appops
```

3. Playbook を実行します。

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

検証手順

ipa group-show コマンドを使用すると、ops グループに sysops および appops がダイレクトメンバーとして追加され、**idm_user** が間接メンバーとして追加されているかどうかを確認できます。

1. 管理者として **ipaserver** にログインします。

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

2. ops についての情報を表示します。

```
ipaserver]$ ipa group-show ops
  Group name: ops
  GID: 1234
```

```
Member groups: sysops, appops
Indirect Member users: idm_user
```

appops および **sysops** グループ (後者には **idm_user** ユーザーを含む) が IdM に存在します。

関連情報

- **/usr/share/doc/ansible-freeipa/README-group.md** Markdown ファイルを参照してください。

24.4. ANSIBLE を使用して単一のタスクで複数の IDM グループを追加する

ansible-freeipa ipagroup モジュールを使用すると、単一の Ansible タスクで複数の Identity Management (IdM) ユーザーグループを追加、変更、削除できます。そのためには、**ipagroup** モジュールの **groups** オプションを使用します。

groups オプションを使用すると、特定のグループにのみ適用される複数のグループ変数を指定することもできます。このグループは **name** 変数で定義します。これは、**groups** オプションの唯一の必須変数です。

この手順を単一のタスクで完了して、IdM に **sysops** グループと **appops** グループが存在することを確認します。sysops グループは非 posix グループとして定義し、appops グループは外部グループとして定義します。

前提条件

- コントロールノードでは、
  - Ansible バージョン 2.14 以降を使用している。
  - **ansible-freeipa** パッケージがインストールされている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している。
  - RHEL 9.3 以降を使用している。
  - **secret.yml** Ansible vault に **ipaadmin_password** が保存されている。

手順

1. 次の内容を含む Ansible Playbook ファイル **add-nonposix-and-external-groups.yml** を作成します。

```
---
- name: Playbook to add nonposix and external groups
  hosts: ipaserver
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml

  tasks:
  - name: Add nonposix group sysops and external group appops
    ipagroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      groups:
```

```
    - name: sysops
      nonposix: true
    - name: appops
      external: true
```

2. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
<path_to_inventory_directory>/hosts <path_to_playbooks_directory>/add-nonposix-
and-external-groups.yml
```

関連情報

- **ansible-freeipa** アップストリームドキュメントの group モジュール

24.5. ANSIBLE を使用して AD ユーザーが IDM を管理できるようにする手順

Ansible Playbook を使用してユーザー ID オーバーライドが Identity Management (IdM) グループに存在することを確認するには、次の手順に従います。ユーザー ID オーバーライドは、Active Directory (AD) とのトラストを確立した後に、デフォルトの Trust View で作成した AD ユーザーのオーバーライドです。Playbook を実行すると、AD 管理者などの AD ユーザーは、2 つの異なるアカウントとパスワードを持たなくても IdM を完全に管理できるようになります。

前提条件

- IdM **admin** のパスワードを把握している。
- AD とのトラストをインストール している。
- IdM に AD ユーザーのユーザー ID オーバーライドがすでに存在する。存在しない場合は、**ipa idoverrideuser-add 'default trust view' ad_user@ad.example.com** コマンドで作成します。
- ユーザー ID オーバーライドを追加しようとしているグループが、IdM にすでに存在する 。
- IdM 4.8.7 バージョン以降の IdM を使用している。サーバーにインストールされている IdM のバージョンを表示するには、**ipa --version** を実行します。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
  - この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。
- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。

手順

1. **~/MyPlaybooks/** ディレクトリーに移動します。

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

2. 次の内容で **add-useridoverride-to-group.yml** playbook を作成します。

```
---
- name: Playbook to ensure presence of users in a group
  hosts: ipaserver


  - name: Ensure the ad_user@ad.example.com user ID override is a member of the admins
group:
    ipagroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: admins
      idoverrideuser:
      - ad_user@ad.example.com
```

上記の例では、以下のようになります。

- Secret123 は IdM **admin** パスワードです。
- **admins** は、**ad_user@ad.example.com** ID オーバーライドを追加する IdM POSIX グループの名前です。このグループのメンバーには、完全な管理者権限があります。
- **ad_user@ad.example.com** は、AD 管理者のユーザー ID オーバーライドです。ユーザーは、信頼が確立された AD ドメインに保存されます。

3. ファイルを保存します。

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

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory add-
useridoverride-to-group.yml
```

関連情報

- AD ユーザーの ID のオーバーライド
- /usr/share/doc/ansible-freeipa/README-group.md
- /usr/share/doc/ansible-freeipa/playbooks/user
- Using ID views in Active Directory environments
- IdM を管理する AD ユーザーの有効化

24.6. ANSIBLE PLAYBOOK を使用して IDM ユーザーグループにメンバーマネージャーを存在させる手順

以下の手順では、Ansible Playbook を使用して、ユーザーとユーザーグループ両方の IdM メンバーマネージャーが存在させる方法を説明します。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
  - この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。
- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。
- メンバーマネージャーとして追加するユーザーまたはグループの名前と、メンバーマネージャーが管理するグループ名を用意する。

手順

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

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

2. 必要なユーザーおよびグループメンバー管理情報を使用して、Ansible Playbook ファイルを作成します。

```
---
- name: Playbook to handle membership management
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure user test is present for group_a
    ipagroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: group_a
      membermanager_user: test

  - name: Ensure group_admins is present for group_a
    ipagroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: group_a
      membermanager_group: group_admins
```

3. Playbook を実行します。

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

検証手順

ipa group-show コマンドを使用すると、**group_a** グループにメンバーマネージャーの **test** が含まれており、**group_admins** が **group_a** のメンバーであることが確認できます。

1. 管理者として **ipaserver** にログインします。

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

2. **managergroup1** についての情報を表示します。

```
ipaserver]$ ipa group-show group_a
  Group name: group_a
  GID: 1133400009
  Membership managed by groups: group_admins
  Membership managed by users: test
```

関連情報

- **ipa host-add-member-manager --help** を参照してください。
- **ipa** の man ページを参照してください。

24.7. ANSIBLE PLAYBOOK を使用して IDM ユーザーグループにメンバーマネージャーを存在させないようにする手順

以下の手順では、Ansible Playbook を使用して、ユーザーとユーザーグループ両方の IdM メンバーマネージャーが存在させないようにする方法を説明します。

前提条件

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

- 削除する既存のメンバーマネージャーのユーザーまたはグループの名前と、そのメンバーマネージャーが管理するグループ名が必要です。

手順

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

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

2. 必要なユーザーおよびグループメンバー管理情報を使用して、Ansible Playbook ファイルを作成します。

```
---
- name: Playbook to handle membership management
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure member manager user and group members are absent for group_a
    ipagroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: group_a
      membermanager_user: test
      membermanager_group: group_admins
      action: member
      state: absent
```

3. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file path_to_playbooks_directory/ensure-
member-managers-are-absent.yml
```

検証手順

ipa group-show コマンドを使用すると、**group_a** グループにメンバーマネージャーの **test** が含まれておらず、**group_admins** が **group_a** のメンバーであることが確認できます。

1. 管理者として **ipaserver** にログインします。

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

2. group_a についての情報を表示します。

```
ipaserver]$ ipa group-show group_a
  Group name: group_a
  GID: 1133400009
```

関連情報

- **ipa host-remove-member-manager --help** を参照してください。
- **ipa** の man ページを参照してください。

第25章 IDM CLI を使用したグループメンバーシップの自動化

自動グループメンバーシップを使用すると、属性に基づいてユーザーとホストをグループに自動割り当てできます。たとえば、以下を行うことができます。

- 従業員のマネージャー、ロケーション、またはその他の属性に基づいて従業員のユーザーエントリーをグループに分類する。
- クラス、ロケーション、またはその他の属性に基づいてホストを分類する。
- 全ユーザーまたは全ホストを 1 つのグローバルグループに追加する。

本章では、以下のトピックについて説明します。

25.1. 自動グループメンバーシップの利点

ユーザーの自動メンバーシップを使用すると、以下が可能になります。

- **グループメンバーシップの手動管理してオーバーヘッドを削減する**
  すべてのユーザーおよびグループを手作業で割り当てる必要がなくなります。
- **ユーザーおよびホスト管理における一貫性を向上する**
  ユーザーとホストは、厳密に定義された基準かつ自動評価された基準をもとにグループに割り当てられます。
- **グループベースの設定管理を簡素化する**
  さまざまな設定がグループに定義され、**sudo** ルール、自動マウント、またはアクセス制御などの個別のグループメンバーに適用されます。ユーザーとホストをグループに自動的に追加すると、この設定の管理が容易になります。

25.2. AUTOMEMBER ルール

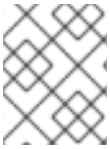
自動グループメンバーシップを設定するには、管理者は automember ルールを定義します。automember ルールは、特定のユーザーまたはホストターゲットグループに適用されます。これは、一度に複数のグループに適用できません。

管理者はルールの作成後に条件を追加します。条件では、ユーザーまたはホストをターゲットグループに含めるか、除外するかを指定します。

- **包含条件**
  ユーザーまたはホストのエントリーが包含条件を満たす場合には、ターゲットグループに含まれます。

- **除外条件**
  ユーザーまたはホストのエントリーが除外条件を満たす場合には、ターゲットグループには含まれません。

この条件は、Perl 互換正規表現 (PCRE) 形式の正規表現として指定します。PCRE の詳細は、**pcresyntax(3)** の man ページを参照してください。



注記

IdM は、包含条件よりも除外条件を先に評価します。競合が発生した場合は、包含条件よりも除外条件が優先されます。

automember ルールは、今後作成されるすべてのエントリーに適用されます。このエントリーは指定されたターゲットグループに自動的に追加されます。エントリーが複数の automember ルールで指定された条件を満たす場合には、該当するグループすべてに追加されます。

既存のエントリーは新規ルールの影響を **受けません**。既存のエントリーを変更する場合は、IdM CLI を使用した既存のエントリーへの automember ルールの適用 を参照してください。

25.3. IDM CLI を使用した AUTOMEMBER ルールの追加

IdM CLI を使用して automember ルールを追加するには、次の手順に従います。automember ルールの詳細は、automember ルール を参照してください。

automember ルールを追加した後に、automember ルールへの条件の追加 で説明されている手順に従って条件を追加できます。

注記

既存のエントリーは新規ルールの影響を **受けません**。既存のエントリーを変更する場合は、IdM CLI を使用した既存のエントリーへの automember ルールの適用 を参照してください。

前提条件

- 管理者としてログインしている。詳細は、Using kinit to log in to IdM manually を参照してください。
- 新しいルールの対象グループは IdM に存在している必要があります。

手順

1. **ipa automember-add** コマンドを入力して、automember ルールを追加します。
2. プロンプトが表示されたら、以下を指定します。
   - **Automember rule**。これはターゲットグループ名です。
   - **Grouping Type**。これは、ルールがユーザーグループまたはホストグループを対象にするかどうかを指定します。ユーザーグループを対象に設定するには、**group** と入力します。ホストグループを対象に設定するには、**hostgroup** と入力します。

たとえば、**user_group** という名前のユーザーグループの automember ルールを追加するには、以下を実行します。

```
$ ipa automember-add
Automember Rule: user_group
Grouping Type: group
---------------------------------
Added automember rule "user_group"
---------------------------------
  Automember Rule: user_group
```

検証手順

- IdM CLI を使用して既存の automember ルールを表示 して、IdM に既存の automember ルールと条件を表示できます。

25.4. IDM CLI を使用した AUTOMEMBER ルールへの条件追加

automember ルールを設定した後、IdM CLI を使用してその automember ルールに条件を追加できます。automember ルールの詳細は、automember ルール を参照してください。

前提条件

- 管理者としてログインしている。詳細は、Using kinit to log in to IdM manually を参照してください。
- ターゲットルールは IdM に存在している必要があります。詳細は IdM CLI を使用した automember ルールの追加 を参照してください。

手順

1. **ipa automember-add-condition** コマンドを使用して、包含または除外の条件を定義します。
2. プロンプトが表示されたら、以下を指定します。
   - **Automember rule**。これはターゲットルール名です。詳細は、Automember ルール を参照してください。
   - **Attribute Key**.これは、フィルターの適用先となるエントリー属性を指定します。たとえば、ユーザーの **uid** です。
   - **Grouping Type**。これは、ルールがユーザーグループまたはホストグループを対象にするかどうかを指定します。ユーザーグループを対象に設定するには、**group** と入力します。ホストグループを対象に設定するには、**hostgroup** と入力します。
   - **Inclusive regex** および **Exclusive regex**。これらは、正規表現として条件を指定します。条件を 1 つだけ指定する場合は、他の条件の入力を求められたら **Enter** を押します。

   たとえば、以下の条件は、ユーザーログイン属性 (**uid**) に値 (.*) が指定されている全ユーザーを対象にしています。

```
$ ipa automember-add-condition
Automember Rule: user_group
Attribute Key: uid
Grouping Type: group
```

```
[Inclusive Regex]: .*
[Exclusive Regex]:
--------------------------------
Added condition(s) to "user_group"
--------------------------------
  Automember Rule: user_group
  Inclusive Regex: uid=.*
----------------------------
Number of conditions added 1
----------------------------
```

別の例として、automembership ルールを使用して、Active Directory (AD) から同期した全 Windows ユーザーを対象にできます。これには、**objectClass** 属性で **ntUser** が指定された全ユーザーを対象にし、全 AD ユーザーに共有される条件を作成します。

```
$ ipa automember-add-condition
Automember Rule: ad_users
Attribute Key: objectclass
Grouping Type: group
[Inclusive Regex]: ntUser
[Exclusive Regex]:
-----------------------------------
Added condition(s) to "ad_users"
-----------------------------------
  Automember Rule: ad_users
  Inclusive Regex: objectclass=ntUser
---------------------------
Number of conditions added 1
---------------------------
```

検証手順

- IdM CLI を使用して既存の automember ルールを表示 して、IdM に既存の automember ルールと条件を表示できます。

25.5. IDM CLI を使用した既存の AUTOMEMBER ルールの表示

IdM CLI を使用して既存の automember ルールを表示するには、次の手順に従います。

前提条件

- 管理者としてログインしている。詳細は、Using kinit to log in to IdM manually を参照してください。

手順

1. **ipa automember-find** コマンドを入力します。
2. プロンプトが表示されたら、**Grouping type** を指定します。
   - ユーザーグループを対象に設定するには、**group** と入力します。
   - ホストグループを対象に設定するには、**hostgroup** と入力します。
     以下に例を示します。

```
$ ipa automember-find
Grouping Type: group
---------------
1 rules matched
---------------
  Automember Rule: user_group
  Inclusive Regex: uid=.*
----------------------------
Number of entries returned 1
----------------------------
```

25.6. IDM CLI を使用した AUTOMEMBER ルールの削除

IdM CLI を使用して automember ルールを削除するには、次の手順に従います。

automember ルールを削除すると、そのルールに関連付けられた条件もすべて削除されます。ルールから特定の条件のみを削除するには、IdM CLI を使用した automember ルールからの条件削除 を参照してください。

前提条件

- 管理者としてログインしている。詳細は、Using kinit to log in to IdM manually を参照してください。

手順

1. **ipa automember-del** コマンドを実行します。
2. プロンプトが表示されたら、以下を指定します。
   - **Automember rule**。これは、削除するルールです。
   - **Grouping rule**。これは、削除するルールがユーザーグループのルールであるか、ホストグループのルールであるかどうかを指定します。**group** または **hostgroup** を入力します。

25.7. IDM CLI を使用した AUTOMEMBER ルールからの条件削除

automember ルールから特定の条件を削除するには、次の手順に従います。

前提条件

- 管理者としてログインしている。詳細は、Using kinit to log in to IdM manually を参照してください。

手順

1. **ipa automember-remove-condition** コマンドを実行します。
2. プロンプトが表示されたら、以下を指定します。
   - **Automember rule**。これは、条件を削除するルールの名前です。
   - **Attribute Key**.これはターゲットエントリー属性です。たとえば、ユーザーの **uid** です。

- **Grouping Type**。これは、ユーザーグループまたはホストグループのどちらの条件を削除するかを指定します。**group** または **hostgroup** を入力します。
- **Inclusive regex** および **Exclusive regex**。この正規表現で、削除する条件を指定します。条件を 1 つだけ指定する場合は、他の条件の入力を求められたら **Enter** を押します。
  以下に例を示します。

```
$ ipa automember-remove-condition
Automember Rule: user_group
Attribute Key: uid
Grouping Type: group
[Inclusive Regex]: .*
[Exclusive Regex]:
--------------------------------
Removed condition(s) from "user_group"
--------------------------------
  Automember Rule: user_group
------------------------------
Number of conditions removed 1
------------------------------
```

25.8. IDM CLI を使用した AUTOMEMBER ルールの既存のエントリーへの適用

Automember ルールは、ルールの追加後に作成されたユーザーおよびホストエントリーに自動的に適用されます。Automember ルールは、ルールの追加前に既存のエントリーに対して遡って適用されることはありません。

以前に追加したエントリーに automember ルールを適用するには、自動メンバーシップを手動で再構築する必要があります。自動メンバーシップを再構築すると、既存の automember ルールがすべて再評価され、すべてのユーザーまたはホストエントリーまたは特定のエントリーに適用されます。

注記

エントリーがグループの包含条件に一致しない場合でも、自動メンバーシップを再構築しても、グループからユーザーまたはホストエントリーは削除 **されません**。手動で削除するには、IdM CLI を使用してユーザーグループからメンバーを削除 または CLI で IdM ホストグループメンバーの削除 を参照してください。

前提条件

- 管理者としてログインしている。詳細は、kinit を使用して IdM に手動でログインする を参照してください。

手順

- 自動メンバーシップを再構築するには、**ipa automember-rebuild** コマンドを実行します。以下のオプションを指定して、ターゲットにエントリーを指定します。
  - 全ユーザーの自動メンバーシップを再構築するには、**--type=group** オプションを使用します。

```
$ ipa automember-rebuild --type=group
--------------------------------------------------------
```

```
Automember rebuild task finished. Processed (9) entries.
-----------------------------------------------------
```

- 全ホストの自動メンバーシップを再構築するには、**--type=hostgroup** オプションを使用します。
- 指定したユーザーの自動メンバーシップを再構築するには、**--users=target_user** オプションを使用します。

```
$ ipa automember-rebuild --users=target_user1 --users=target_user2
-----------------------------------------------------
Automember rebuild task finished. Processed (2) entries.
-----------------------------------------------------
```

- 指定したホストの自動メンバーシップを再構築するには、**--hosts=client.idm.example.com** を使用します。

25.9. デフォルトの AUTOMEMBER グループの設定

デフォルトの automember グループを設定すると、automember ルールに一致しない新規ユーザーまたはホストエントリーは自動的にこのデフォルトグループに追加されます。

前提条件

- 管理者としてログインしている。詳細は、Using kinit to log in to IdM manually を参照してください。
- デフォルトとして設定されるターゲットグループが IdM にある。

手順

1. **ipa automember-default-group-set** コマンドを入力して、デフォルトの automember グループを設定します。
2. プロンプトが表示されたら、以下を指定します。
   - **Default (fallback) Group**。ターゲットグループ名を指定します。
   - **Grouping Type**。ターゲットがユーザーグループか、ホストグループであるかを指定します。ユーザーグループを対象に設定するには、**group** と入力します。ホストグループを対象に設定するには、**hostgroup** と入力します。
     以下に例を示します。

```
$ ipa automember-default-group-set
Default (fallback) Group: default_user_group
Grouping Type: group
-------------------------------------------------
Set default (fallback) group for automember "default_user_group"
-------------------------------------------------
  Default (fallback) Group:
cn=default_user_group,cn=groups,cn=accounts,dc=example,dc=com
```

注記

現在のデフォルトの automember グループを削除するには、**ipa automember-default-group-remove** コマンドを実行します。

検証手順

- グループが正しく設定されていることを確認するには、**ipa automember-default-group-show** コマンドを実行します。コマンドは、現在のデフォルトの automember グループを表示します。以下に例を示します。

```
$ ipa automember-default-group-show
Grouping Type: group
  Default (fallback) Group:
cn=default_user_group,cn=groups,cn=accounts,dc=example,dc=com
```

第26章 IDM WEB UI を使用したグループメンバーシップの自動化

自動グループメンバーシップを使用すると、属性に基づいてユーザーとホストをグループに自動的に割り当てることができます。たとえば、以下を行うことができます。

- 従業員のマネージャー、ロケーション、またはその他の属性に基づいて従業員のユーザーエントリーをグループに分類する。
- クラス、ロケーション、またはその他の属性に基づいてホストを分類する。
- 全ユーザーまたは全ホストを 1 つのグローバルグループに追加する。

本章では、以下のトピックについて説明します。

26.1. 自動グループメンバーシップの利点

ユーザーの自動メンバーシップを使用すると、以下が可能になります。

- **グループメンバーシップの手動管理してオーバーヘッドを削減する**
  すべてのユーザーおよびグループを手作業で割り当てる必要がなくなります。
- **ユーザーおよびホスト管理における一貫性を向上する**
  ユーザーとホストは、厳密に定義された基準かつ自動評価された基準をもとにグループに割り当てられます。
- **グループベースの設定管理を簡素化する**
  さまざまな設定がグループに定義され、**sudo** ルール、自動マウント、またはアクセス制御などの個別のグループメンバーに適用されます。ユーザーとホストをグループに自動的に追加すると、この設定の管理が容易になります。

26.2. AUTOMEMBER ルール

自動グループメンバーシップを設定するには、管理者は automember ルールを定義します。automember ルールは、特定のユーザーまたはホストターゲットグループに適用されます。これは、一度に複数のグループに適用できません。

管理者はルールの作成後に条件を追加します。条件では、ユーザーまたはホストをターゲットグループに含めるか、除外するかを指定します。

- **包含条件**
  ユーザーまたはホストのエントリーが包含条件を満たす場合には、ターゲットグループに含まれます。

- **除外条件**
  ユーザーまたはホストのエントリーが除外条件を満たす場合には、ターゲットグループには含まれません。

この条件は、Perl 互換正規表現 (PCRE) 形式の正規表現として指定します。PCRE の詳細は、**pcresyntax(3)** の man ページを参照してください。

注記

IdM は、包含条件よりも除外条件を先に評価します。競合が発生した場合は、包含条件よりも除外条件が優先されます。

automember ルールは、今後作成されるすべてのエントリーに適用されます。このエントリーは指定されたターゲットグループに自動的に追加されます。エントリーが複数の automember ルールで指定された条件を満たす場合には、該当するグループすべてに追加されます。

既存のエントリーは新規ルールの影響を **受けません**。既存のエントリーを変更する場合は、IdM Web UI を使用した automember ルールの既存エントリーへの適用 を参照してください。

26.3. IDM WEB UI を使用した AUTOMEMBER ルールの追加

IdM Web UI を使用して automember ルールを追加するには、次の手順に従います。automember ルールの詳細は、automember ルール を参照してください。

注記

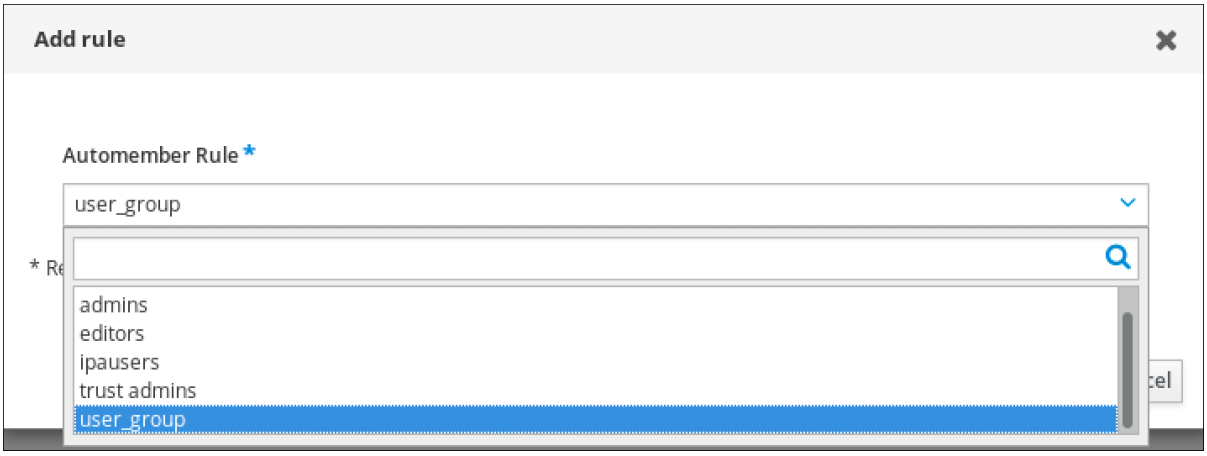
既存のエントリーは新規ルールの影響を **受けません**。既存のエントリーを変更する場合は、IdM Web UI を使用した automember ルールの既存エントリーへの適用 を参照してください。

前提条件

- IdM Web UI にログインしている。
- **admins** グループのメンバーである。
- 新しいルールのターゲットグループが IdM に存在する。

手順

1. **Identity → Automember** をクリックして、**User group rule** または **Host group rules** を選択します。
2. **Add** をクリックします。
3. **Automember rule** フィールドで、ルールを適用するグループを選択します。これはターゲットグループ名です。



4. **Add** をクリックして確定します。
5. 必要に応じて、IdM Web UI を使用した automember ルールへの条件の追加 で説明されている手順に従って、新しいルールに条件を追加できます。

26.4. IDM WEB UI を使用した AUTOMEMBER ルールへの条件の追加

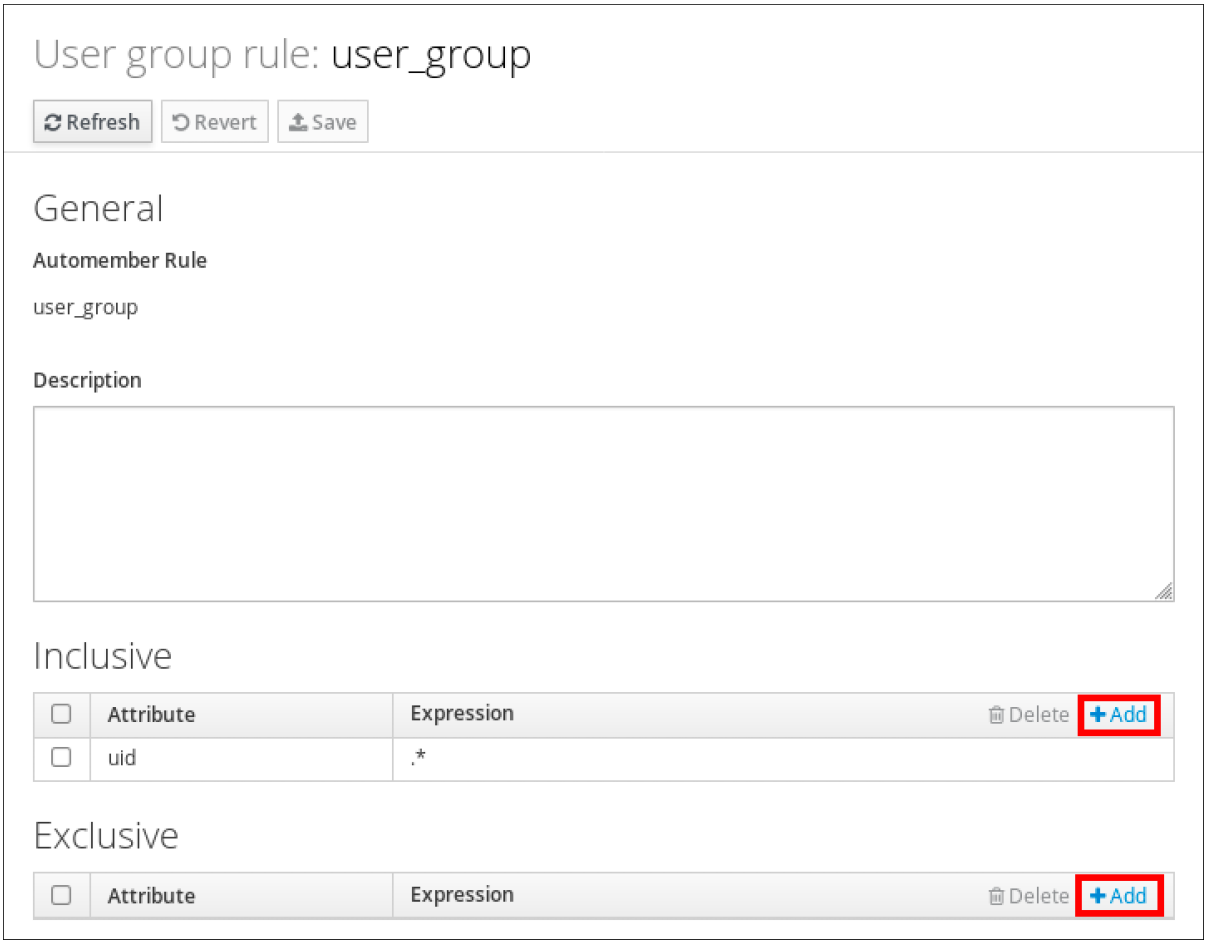
automember ルールを設定した後、IdM Web UI を使用してその automember ルールに条件を追加できます。automember ルールの詳細は、automember ルール を参照してください。

前提条件

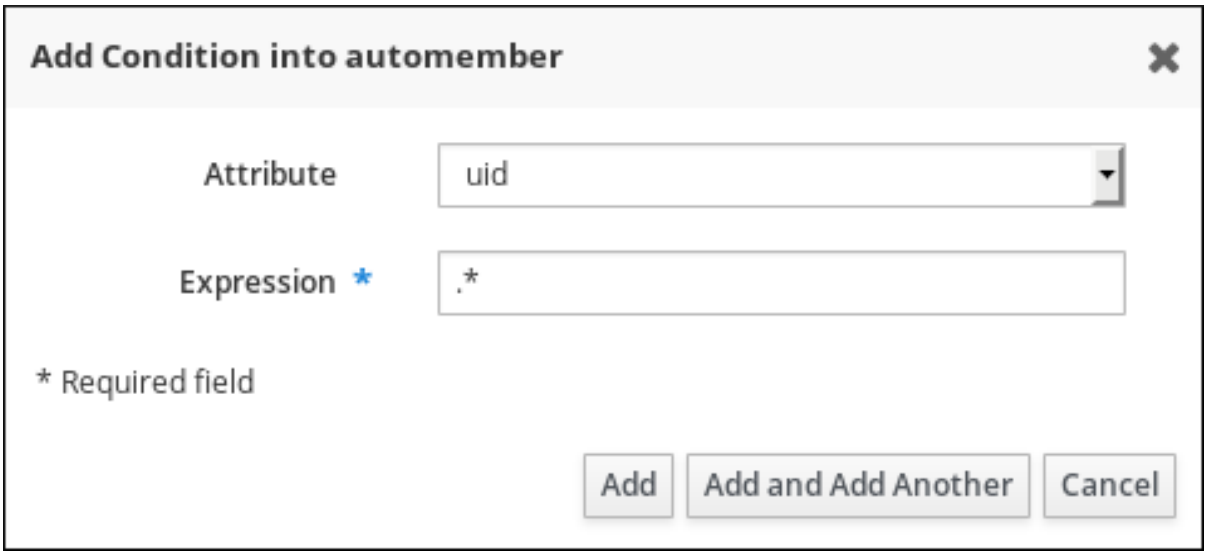
- IdM Web UI にログインしている。
- **admins** グループのメンバーである。
- ターゲットルールが IdM に存在する。

手順

1. **Identity → Automember** をクリックして、**User group rule** または **Host group rules** を選択します。
2. 条件を追加するルールをクリックします。
3. **Inclusive** セクションまたは **Exclusive** セクションで、Add をクリックします。



4. **Attribute** フィールドで、必要な属性 (**uid** など) を選択します。
5. **Expression** フィールドに正規表現を定義します。
6. **Add** をクリックします。
   たとえば、以下の条件は、ユーザー ID (uid) 属性に値 (.*) が指定されているすべてのユーザーを対象とします。



26.5. IDM WEB UI を使用した既存の AUTOMEMBER ルールおよび条件の表示

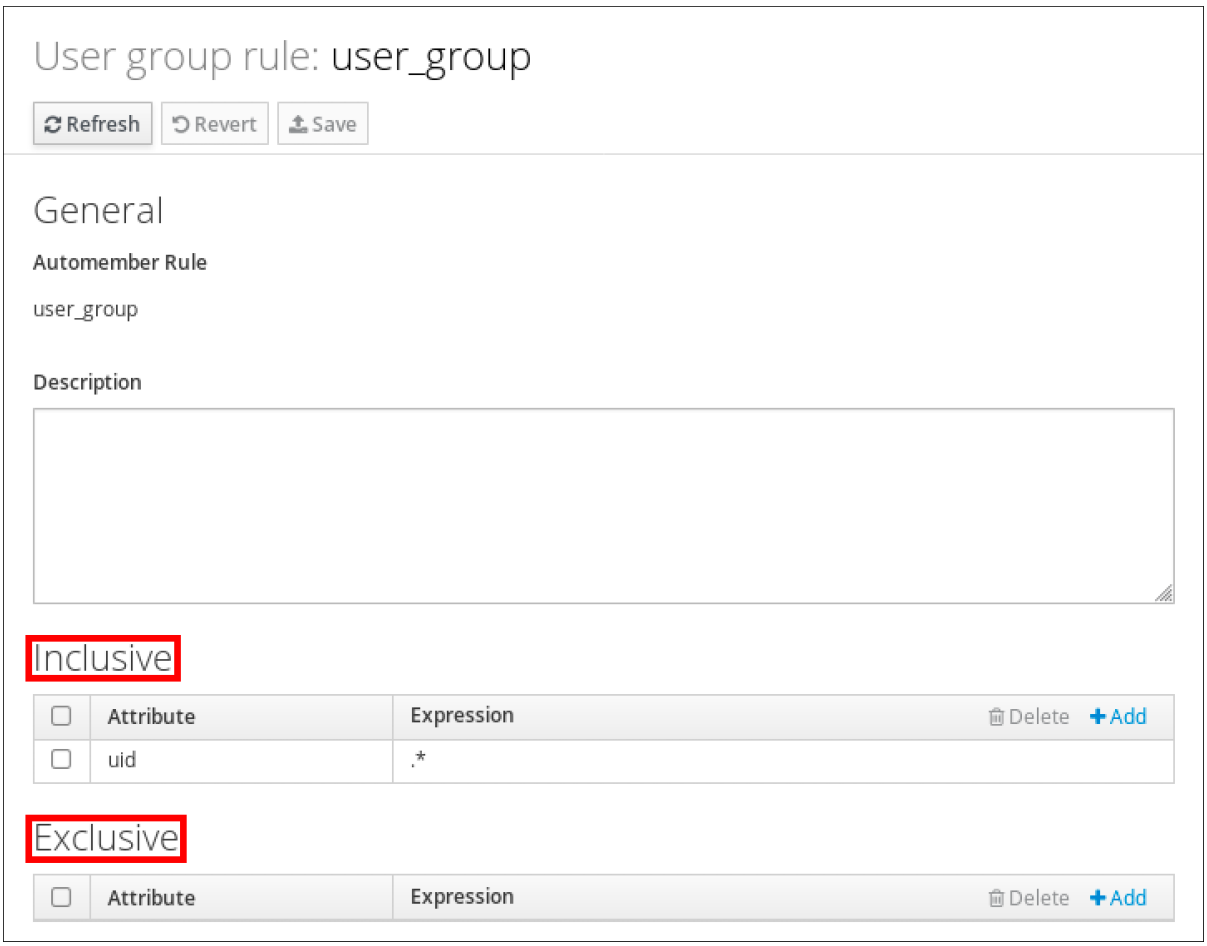
IdM Web UI を使用して既存の automember ルールと条件を表示するには、次の手順に従います。

前提条件

- IdM Web UI にログインしている。
- **admins** グループのメンバーである。

手順

1. **Identity → Automember** をクリックして、**User group rule** または **Host group rules** を選択して、それぞれの automember ルールを表示します。
2. 必要に応じて、ルールをクリックして、**Inclusive** セクションまたは **Exclusive** セクションにそのルールの条件を表示します。



26.6. IDM WEB UI を使用した AUTOMEMBER ルールの削除

IdM Web UI を使用して automember ルールを削除するには、次の手順に従います。

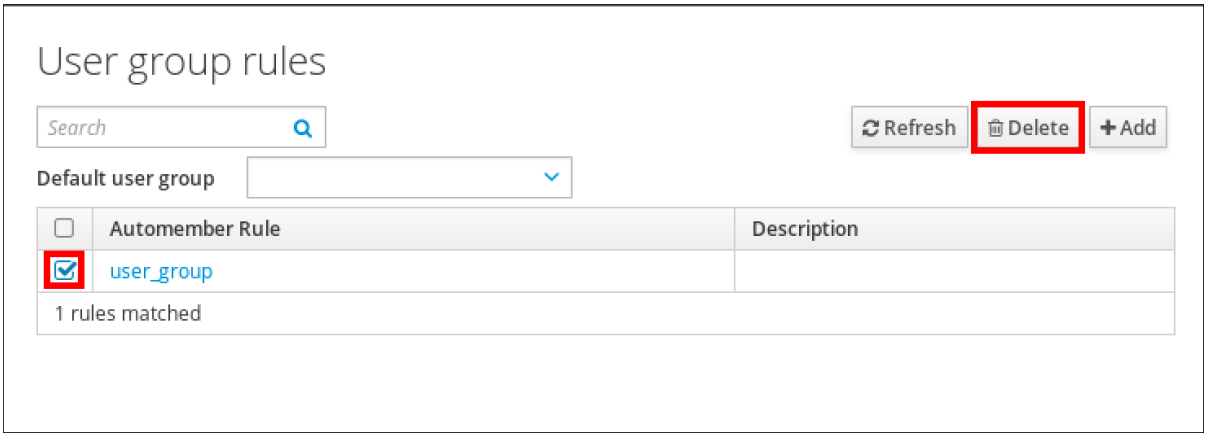
automember ルールを削除すると、そのルールに関連付けられた条件もすべて削除されます。ルールから特定の条件のみを削除するには、IdM Web UI を使用した automember ルールからの条件削除 を参照してください。

前提条件

- IdM Web UI にログインしている。
- **admins** グループのメンバーである。

手順

1. **Identity → Automember** をクリックして、**User group rule** または **Host group rules** を選択して、それぞれの automember ルールを表示します。
2. 削除するルールの横にあるチェックボックスを選択します。
3. **Delete** をクリックします。



4. **Delete** をクリックして確定します。

26.7. IDM WEB UI を使用した AUTOMEMBER ルールからの条件削除

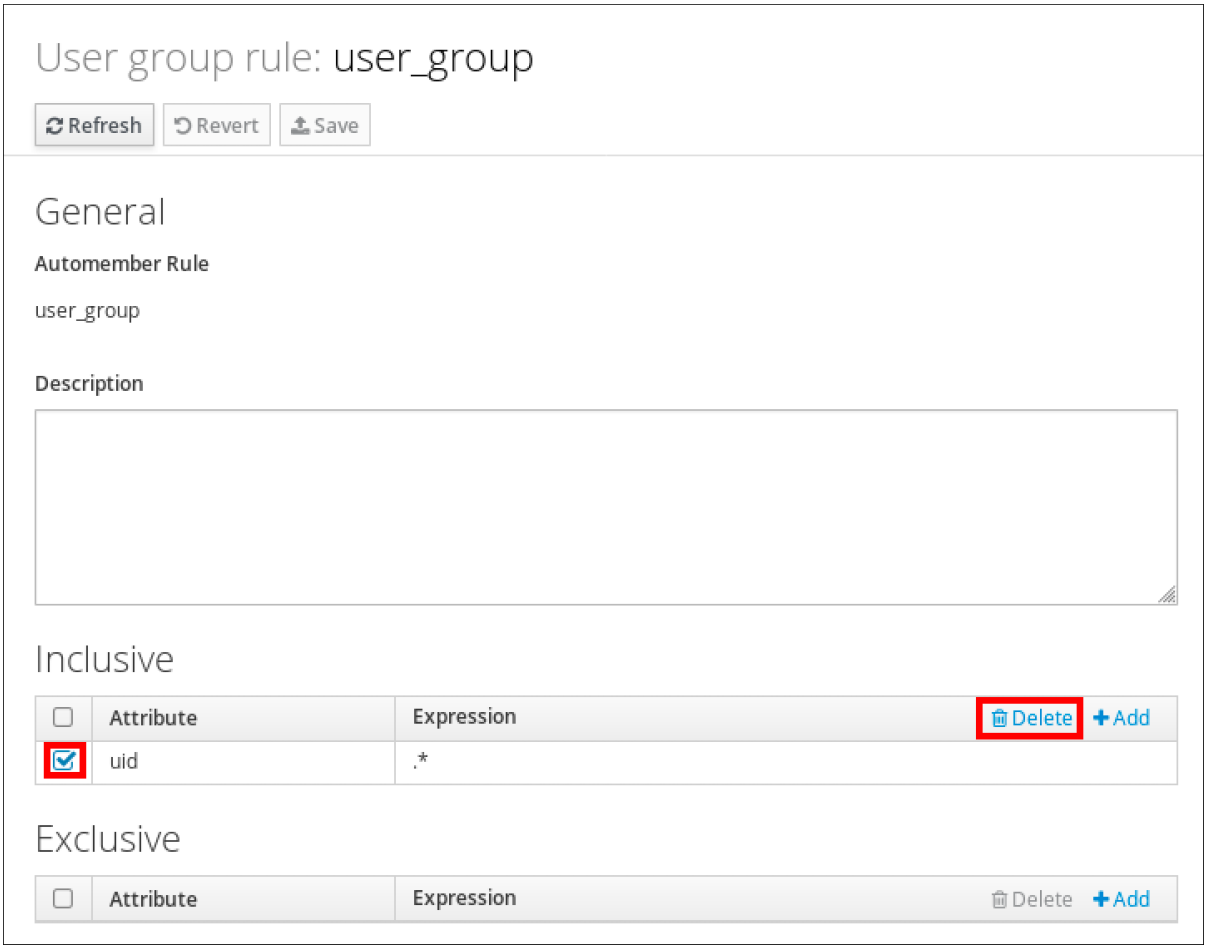
IdM Web UI を使用して automember ルールから特定の条件を削除するには、次の手順に従います。

前提条件

- IdM Web UI にログインしている。
- **admins** グループのメンバーである。

手順

1. **Identity → Automember** をクリックして、**User group rule** または **Host group rules** を選択して、それぞれの automember ルールを表示します。
2. ルールをクリックして、**Inclusive** セクションまたは **Exclusive** セクションでそのルールの条件を表示します。
3. 削除する条件の横にあるチェックボックスを選択します。
4. **Delete** をクリックします。

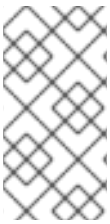


5. **Delete** をクリックして確定します。

26.8. IDM WEB UI を使用した AUTOMEMBER ルールの既存エントリーへの適用

Automember ルールは、ルールの追加後に作成されたユーザーおよびホストエントリーに自動的に適用されます。Automember ルールは、ルールの追加前に既存のエントリーに対して遡って適用されることはありません。

以前に追加したエントリーに automember ルールを適用するには、自動メンバーシップを手動で再構築する必要があります。自動メンバーシップを再構築すると、既存の automember ルールがすべて再評価され、すべてのユーザーまたはホストエントリーまたは特定のエントリーに適用されます。



注記

エントリーがグループの包含条件に一致しない場合でも、自動メンバーシップを再構築しても、グループからユーザーまたはホストエントリーは削除 **されません**。手動で削除するには、IdM Web UI を使用してユーザーグループからメンバーの削除 または IdM Web UI でホストグループメンバーの削除 を参照してください。

26.8.1. 全ユーザーまたは全ホストの自動メンバーシップの再構築

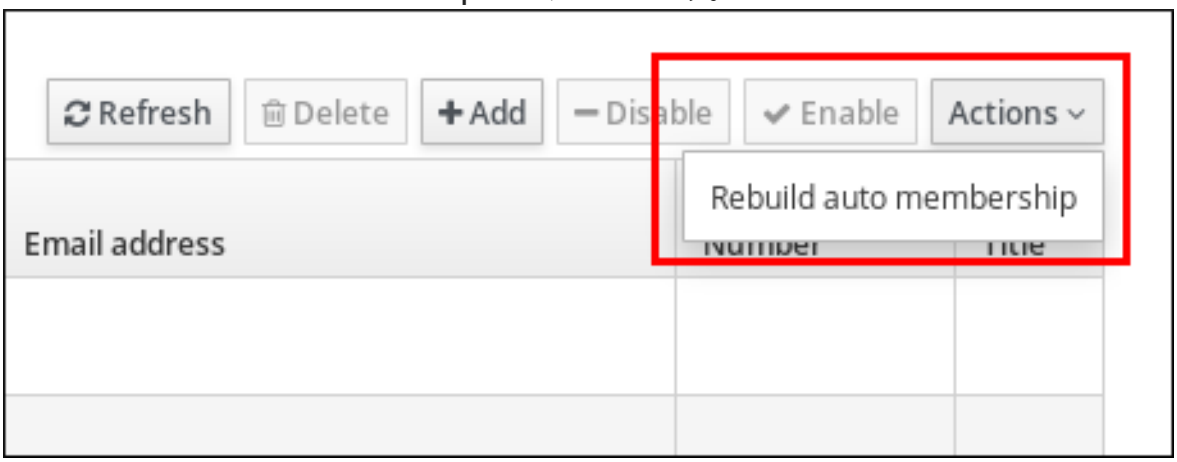
すべてのユーザーまたはホストエントリーの自動メンバーシップを再構築するには、次の手順に従います。

前提条件

- IdM Web UI にログインしている。
- **admins** グループのメンバーである。

手順

1. **Identity → Users** または **Hosts** を選択します。
2. **Actions → Rebuild auto membership** をクリックします。



26.8.2. ユーザーまたはホスト 1 つに対する自動メンバーシップの再構築

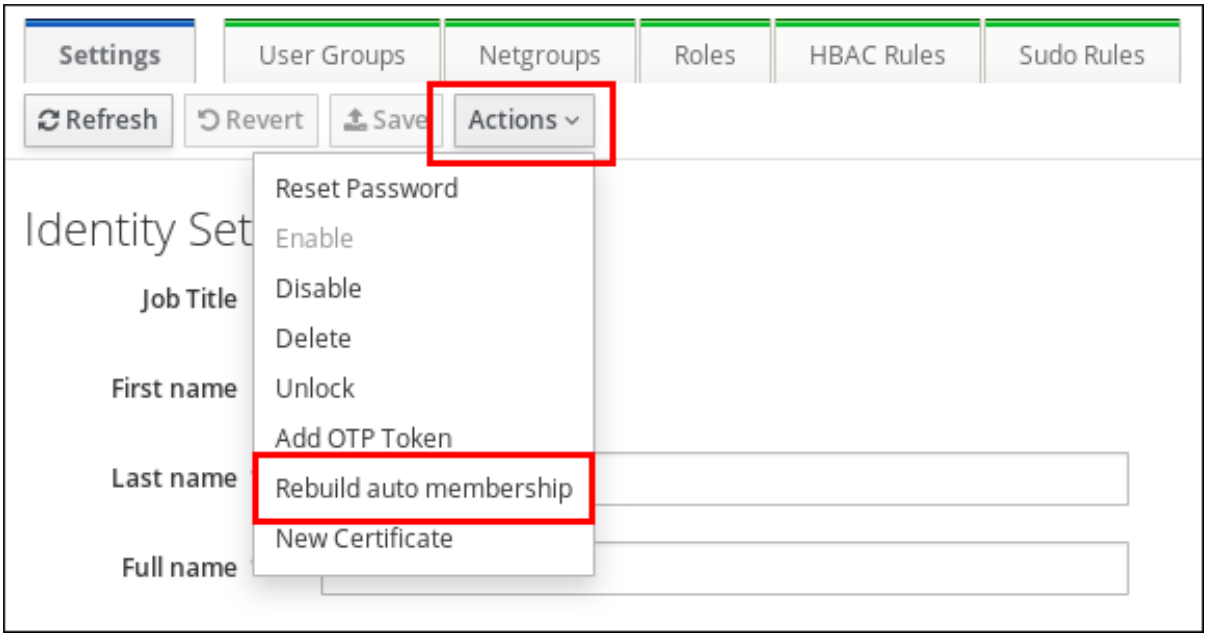
特定のユーザーまたはホストエントリーの自動メンバーシップを再構築するには、次の手順に従います。

前提条件

- IdM Web UI にログインしている。
- **admins** グループのメンバーである。

手順

1. **Identity → Users** または **Hosts** を選択します。
2. 必要なユーザー名またはホスト名をクリックします。
3. **Actions → Rebuild auto membership** をクリックします。



26.9. IDM WEB UI を使用したデフォルトのユーザーグループの設定

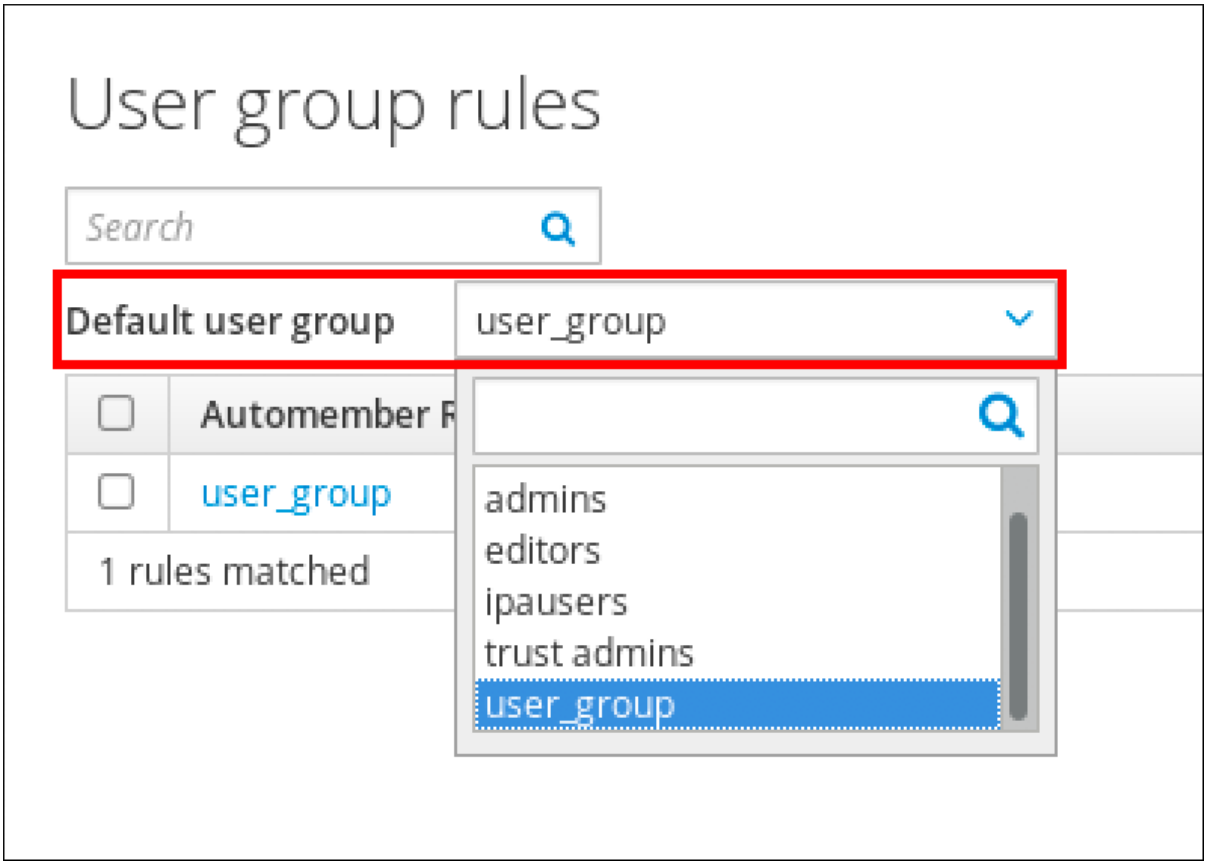
デフォルトのユーザーグループの設定時に、automember ルールに一致しない新規ユーザーエントリーは自動的にこのデフォルトグループに追加されます。

前提条件

- IdM Web UI にログインしている。
- **admins** グループのメンバーである。
- デフォルトとして設定するターゲットユーザーグループが IdM にある。

手順

1. **Identity → Automember** をクリックして、**User group rules** を選択します。
2. **Default user group** フィールドで、デフォルトのユーザーグループとして設定するグループを選択します。



26.10. IDM WEB UI を使用したデフォルトのホストグループの設定

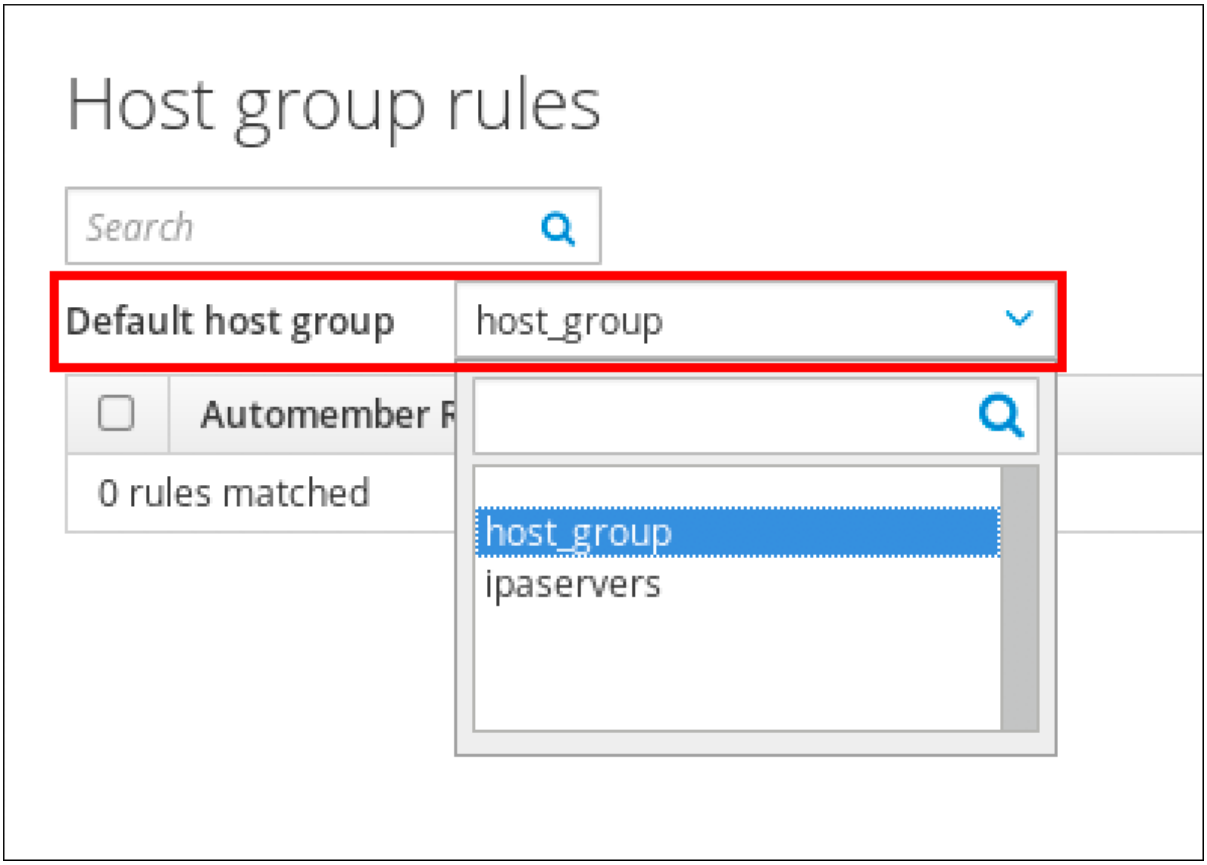
デフォルトのホストグループの設定時に、automember ルールに一致しない新規ホストエントリーが自動的にこのデフォルトグループに追加されます。

前提条件

- IdM Web UI にログインしている。
- **admins** グループのメンバーである。
- デフォルトとして設定されるターゲットホストグループが IdM にある。

手順

1. **Identity → Automember** をクリックして、**Host group rules** を選択します。
2. **Default host group** フィールドで、デフォルトのホストグループとして設定するグループを選択します。


Host group rules
Search
Default host group
host_group
Automember R
0 rules matched
host_group
ipaservers

第27章 ANSIBLE を使用した IDM のグループメンバーシップの自動化

自動グループメンバーシップを使用すると、ユーザーとホストのユーザーグループとホストグループを、その属性に基づいて自動的に割り当てることができます。たとえば、以下を行うことができます。

- 従業員のユーザーエントリーを、従業員のマネージャー、場所、役職などの属性に基づいてグループに分割します。コマンドラインに **ipa user-add --help** と入力すると、すべての属性をリスト表示できます。
- ホストを、クラス、場所、またはその他の属性に基づいてグループに分割します。コマンドラインに **ipa host-add --help** と入力すると、すべての属性をリスト表示できます。
- 全ユーザーまたは全ホストを 1 つのグローバルグループに追加する。

Red Hat Ansible Engine を使用すると、Identity Management (IdM) で自動グループメンバーシップの管理を自動化できます。

このセクションでは、以下のトピックについて説明します。

- IdM 管理用の Ansible コントロールノードの準備
- Ansible を使用した IdM ユーザーグループの automember ルールが存在することの確認
- Ansible を使用した IdM ユーザーグループの automember ルールに条件が存在することの確認
- Ansible を使用した IdM ユーザーグループの automember ルールに条件がないことの確認
- Ansible を使用した IdM グループの automember ルールがないことの確認
- Ansible を使用した IdM ホストグループの automember ルールに条件が存在することの確認

27.1. IDM 管理用の ANSIBLE コントロールノードの準備

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

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

この方法に従うことで、すべての Playbook を 1 カ所で見つけることができます。また、root 権限を呼び出さなくても Playbook を実行できます。

注記

ipaserver、**ipareplica**、**ipaclient**、**ipabackup**、**ipasmartcard_server**、および **ipasmartcard_client ansible-freeipa** のロールを実行するために必要なのは、管理対象ノードでの **root** 権限のみです。これらのロールには、ディレクトリーおよび **dnf** ソフトウェアパッケージマネージャーへの特権アクセスが必要です。

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

前提条件

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

手順

1. Ansible 設定および Playbook のディレクトリーをホームディレクトリーに作成します。

```
$ mkdir ~/MyPlaybooks/
```

2. **~/MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks
```

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

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

[privilege_escalation]
become=True
```

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

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

[ipareplicas]
replica1.idm.example.com
replica2.idm.example.com

[ipacluster:children]
ipaserver
ipareplicas

[ipacluster:vars]
ipaadmin_password=SomeADMINpassword

[ipaclients]
ipaclient1.example.com
ipaclient2.example.com

[ipaclients:vars]
ipaadmin_password=SomeADMINpassword
```

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

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

```
$ ssh-keygen
```

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

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

これらのコマンドを入力する場合は、IdM **admin** パスワードを入力する必要があります。

関連情報

- Ansible Playbook で Identity Management サーバーのインストール
- インベントリーの構築方法.

27.2. ANSIBLE を使用した IDM ユーザーグループの AUTOMEMBER ルールが存在することの確認

以下の手順では、Ansible Playbook を使用して、Identity Management (IdM) グループの **automember** ルールが存在することを確認する方法について説明しますこの例では、**testing_group** ユーザーグループに対して **automember** ルールの存在が保証されます。

前提条件

- IdM **admin** のパスワードを把握している。
- **testing_group** ユーザーグループが IdM に存在します。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
  - この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。
- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。

手順

1. **~/MyPlaybooks/** ディレクトリーに移動します。

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

2. **/usr/share/doc/ansible-freeipa/playbooks/automember/** ディレクトリーにある **automember-group-present.yml** Ansible Playbook ファイルをコピーします。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/automember/automember-group-present.yml automember-group-present-copy.yml
```

3. **automember-group-present-copy.yml** ファイルを開いて編集します。

4. **ipaautomember** タスクセクションで次の変数を設定して、ファイルを調整します。

   - **ipaadmin_password** 変数は IdM **admin** のパスワードに設定します。
   - **name** 変数を **testing_group** に設定します。
   - **automember_type** 変数を **group** に設定します。
   - **state** 変数は **present** に設定されていることを確認します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Automember group present example
  hosts: ipaserver
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure group automember rule admins is present
    ipaautomember:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: testing_group
      automember_type: group
      state: present
```

5. ファイルを保存します。

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

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory automember-group-present-copy.yml
```

関連情報

- 自動グループメンバーシップの利点 および automember ルール を参照してください。
- Ansible を使用した IdM ユーザーグループの automember ルールが存在することの確認 を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-automember.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/automember** ディレクトリーを参照してください。

27.3. ANSIBLE を使用した、IDM ユーザーグループの AUTOMEMBER ルールに指定した条件が存在することの確認

以下の手順では、Ansible Playbook を使用して、Identity Management (IdM) グループの **automember** ルールに、指定した条件が存在することを確認する方法について説明しますこの例では、**testing_group** グループに対して、**automember** ルールに UID 関連の条件が存在することが保証されます。.* 条件を指定することで、今後使用する IdM ユーザーがすべて自動的に **testing_group** のメンバーになるようにします。

前提条件

- IdM **admin** のパスワードを把握している。
- **testing_group** ユーザーグループおよび automember ユーザーグループルールが IdM に存在します。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
  - この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。
- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。

手順

1. **~/MyPlaybooks/** ディレクトリーに移動します。

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

2. **/usr/share/doc/ansible-freeipa/playbooks/automember/** ディレクトリーにある **automember-hostgroup-rule-present.yml** Ansible Playbook ファイルをコピーして、名前を付けます (**automember-usergroup-rule-present.yml** など)。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/automember/automember-hostgroup-rule-present.yml automember-usergroup-rule-present.yml
```

3. **automember-usergroup-rule-present.yml** を開いて編集します。
4. 次のパラメーターを変更して、ファイルを調整します。
   - ユースケースに対応するように Playbook の名前を変更します (例: **Automember user group rule member present**)。
   - ユースケースに合わせて、タスクの名前を変更します (例: **Ensure an automember condition for a user group is present**)。
   - **ipaautomember** タスクセクションで、以下の変数を設定します。

- **ipaadmin_password** 変数は IdM **admin** のパスワードに設定します。
- **name** 変数を **testing_group** に設定します。
- **automember_type** 変数を **group** に設定します。
- **state** 変数は **present** に設定されていることを確認します。
- **action** 変数が **member** に設定されていることを確認します。
- **inclusive key** 変数を **UID** に設定します。
- **inclusive expression** 変数を .* に設定します。

以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Automember user group rule member present
  hosts: ipaserver
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure an automember condition for a user group is present
    ipaautomember:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: testing_group
      automember_type: group
      state: present
      action: member
      inclusive:
        - key: UID
          expression: .*
```

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

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory automember-usergroup-rule-present.yml
```

検証手順

1. IdM 管理者としてログインします。

```
$ kinit admin
```

2. ユーザーを追加します。以下に例を示します。

```
$ ipa user-add user101 --first user --last 101
----------------------
Added user "user101"
----------------------
  User login: user101
  First name: user
```

```
Last name: 101
...
Member of groups: ipausers, testing_group
...
```

関連情報

- IdM CLI を使用した既存エントリーへの automember ルールの適用 を参照してください。
- 自動グループメンバーシップの利点 および automember ルール を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-automember.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/automember** ディレクトリーを参照してください。

27.4. ANSIBLE を使用した IDM ユーザーグループの AUTOMEMBER ルールに条件がないことの確認

以下の手順では、Ansible Playbook を使用して、Identity Management (IdM) グループの **automember** ルールに条件がないことを確認する方法を説明します。この例では、**automember** ルールに条件がないことが保証されており、**initials** が**dp** であるユーザーを含める必要があることを指定しています。automember ルールが **testing_group** グループに適用されます。条件を適用することにより、今後は、イニシャルが**dp**である IdM ユーザーが **testing_group**のメンバーにならないようにします。

前提条件

- IdM **admin** のパスワードを把握している。
- **testing_group** ユーザーグループおよび automember ユーザーグループルールが IdM に存在します。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
  - この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。
- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。

手順

1. **~/MyPlaybooks/** ディレクトリーに移動します。

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

2. **/usr/share/doc/ansible-freeipa/playbooks/automember/** ディレクトリーにある **automember-hostgroup-rule-absent.yml** Ansible Playbook ファイルをコピーして、名前を付けます (**automember-usergroup-rule-absent.yml** など)。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/automember/automember-hostgroup-rule-absent.yml automember-usergroup-rule-absent.yml
```

3. **automember-usergroup-rule-absent.yml** を開いて編集します。

4. 次のパラメーターを変更して、ファイルを調整します。

   - ユースケースに対応するように Playbook の名前を変更します (例: **Automember user group rule member absent**)。
   - ユースケースに合わせて、タスクの名前を変更します (例: **Ensure an automember condition for a user group is absent**)。
   - **ipaautomember** タスクセクションで、以下の変数を設定します。
     - **ipaadmin_password** 変数は IdM **admin** のパスワードに設定します。
     - **name** 変数を **testing_group** に設定します。
     - **automember_type** 変数を **group** に設定します。
     - **state** 変数は、**absent** に設定されていることを確認します。
     - **action** 変数が **member** に設定されていることを確認します。
     - **inclusive key** 変数を **initials** に設定します。
     - **inclusive expression** 変数を **dp** に設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Automember user group rule member absent
  hosts: ipaserver
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure an automember condition for a user group is absent
    ipaautomember:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: testing_group
      automember_type: group
      state: absent
      action: member
      inclusive:
        - key: initials
          expression: dp
```

5. ファイルを保存します。

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

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory automember-usergroup-rule-absent.yml
```

検証手順

1. IdM 管理者としてログインします。

```
$ kinit admin
```

2. automember グループを表示します。

```
$ ipa automember-show --type=group testing_group
 Automember Rule: testing_group
```

出力に **Inclusive Regex: initials=dp** エントリーがない場合は、**testing_group** automember ルールに指定した条件が含まれていないことを確認します。

関連情報

- IdM CLI を使用した既存エントリーへの automember ルールの適用 を参照してください。
- 自動グループメンバーシップの利点 および automember ルール を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-automember.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/automember** ディレクトリーを参照してください。

27.5. ANSIBLE を使用した IDM ユーザーグループの AUTOMEMBER ルールがないことの確認

以下の手順では、Ansible Playbook を使用して、Identity Management (IdM) グループに **automember** ルールがないことを確認する方法を説明します。この例では、**testing_group** グループに **automember** ルールがないことが保証されます。

注記

automember ルールを削除すると、そのルールに関連付けられた条件もすべて削除されます。ルールから特定の条件のみを削除するには、Ansible を使用した IdM ユーザーグループの automember ルールに条件がないことの確認 を参照してください。

前提条件

- IdM **admin** のパスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。

- この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。

- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。

手順

1. **~/MyPlaybooks/** ディレクトリーに移動します。

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

2. **/usr/share/doc/ansible-freeipa/playbooks/automember/** ディレクトリーにある **automember-group-absent.yml** Ansible Playbook ファイルをコピーします。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/automember/automember-group-absent.yml automember-group-absent-copy.yml
```

3. **automember-group-absent-copy.yml** を開いて編集します。

4. **ipaautomember** タスクセクションで次の変数を設定して、ファイルを調整します。

- **ipaadmin_password** 変数は IdM **admin** のパスワードに設定します。
- **name** 変数を testing_group に設定します。
- **automember_type** 変数を group に設定します。
- **state** 変数は、**absent** に設定されていることを確認します。

以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Automember group absent example
  hosts: ipaserver
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure group automember rule admins is absent
    ipaautomember:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: testing_group
      automember_type: group
      state: absent
```

5. ファイルを保存します。

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

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory automember-group-absent.yml
```

関連情報

- 自動グループメンバーシップの利点 および automember ルール を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-automember.md** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/automember** ディレクトリーを参照してください。

27.6. ANSIBLE を使用した IDM ホストグループの AUTOMEMBER ルールに条件が存在することの確認

以下の手順に従って、Ansible を使用して、IdM ホストグループの automember ルールに条件が存在することを確認します。この例では、**FQDN** が **.*.idm.example.com** のホストが、**primary_dns_domain_hosts** ホストグループのメンバーであることと、**FQDN** が **.*.example.org** であるホストが **primary_dns_domain_hosts** ホストグループのメンバーではないことを確認する方法を説明します。

前提条件

- IdM **admin** のパスワードを把握している。
- **primary_dns_domain_hosts** ホストグループおよび automember ホストグループルールが IdM に存在します。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
  - この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。
- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。

手順

1. **~/MyPlaybooks/** ディレクトリーに移動します。

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

2. **/usr/share/doc/ansible-freeipa/playbooks/automember/** ディレクトリーにある **automember-hostgroup-rule-present.yml** Ansible Playbook ファイルをコピーします。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/automember/automember-hostgroup-rule-present.yml automember-hostgroup-rule-present-copy.yml
```

3. **automember-hostgroup-rule-present-copy.yml** を開いて編集します。
4. **ipaautomember** タスクセクションで次の変数を設定して、ファイルを調整します。

- **ipaadmin_password** 変数は IdM **admin** のパスワードに設定します。
- **name** 変数を primary_dns_domain_hosts に設定します。
- **automember_type** を hostgroup に設定します。
- **state** 変数は **present** に設定されていることを確認します。
- **action** 変数が **member** に設定されていることを確認します。
- **inclusive key** 変数が **fqdn** に設定されていることを確認します。
- 対応する **inclusive expression** 変数を**.*.idm.example.com**に設定します。
- **exclusive key** 変数を **UID** に設定します。
- 対応する **exclusive expression** 変数を **.*.example.org** に設定します。

以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Automember user group rule member present
  hosts: ipaserver
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure an automember condition for a user group is present
    ipaautomember:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: primary_dns_domain_hosts
      automember_type: hostgroup
      state: present
      action: member
      inclusive:
        - key: fqdn
          expression: .*.idm.example.com
      exclusive:
        - key: fqdn
          expression: .*.example.org
```

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

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory automember-hostgroup-rule-present-copy.yml
```

関連情報

- IdM CLI を使用した既存エントリーへの automember ルールの適用 を参照してください。
- 自動グループメンバーシップの利点 および automember ルール を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-automember.md** ファイルを参照してください。

- **/usr/share/doc/ansible-freeipa/playbooks/automember** ディレクトリーを参照してください。

27.7. 関連情報

- Ansible Playbook を使用したユーザーアカウントの管理
- Ansible Playbook を使用したホストの管理
- Ansible Playbook を使用したユーザーグループの管理
- IdM CLI を使用したホストグループの管理

第28章 IDM CLI を使用してユーザーグループにパーミッションを委譲してユーザーを管理する手順

委譲は、セルフサービスルールおよびロールベースのアクセス制御 (RBAC) などの IdM のアクセス制御メソッドの 1 つです。委譲を使用して、あるユーザーのグループにパーミッションを割り当てて別のユーザーのグループのエントリーを管理できます。

このセクションでは、以下のトピックについて説明します。

28.1. 委譲ルール

委譲ルール を作成して、ユーザーグループにパーミッションを委譲してユーザーを管理できます。

委譲ルールを使用すると、特定のユーザーグループが、別のユーザーグループ内のユーザーの特定の属性に対して書き込み (編集) 操作を実行できます。このようなアクセス制御ルールは、委譲ルールで指定された属性のサブセットの編集に限定されており、エントリー全体の追加や削除、未指定の属性の制御はできません。

委譲ルールにより、IdM の既存のユーザーグループにパーミッションが付与されます。委任を使用すると、**managers** ユーザーグループで **employees** ユーザーグループでユーザーの選択された属性を管理できます。

28.2. IDM CLI を使用した委譲ルールの作成

IdM CLI を使用して委譲ルールを作成するには、次の手順に従います。

前提条件

- **admins** グループのメンバーとしてログインしている。

手順

- **ipa delegation-add** コマンドを入力します。以下のオプションを指定します。
  - **--group**: ユーザーグループ内のユーザーのエントリーに対する **パーミッションを付与されている** グループです。
  - **--membergroup**: 委譲グループのメンバーが **エントリーを編集できる** グループです。
  - **--permissions**: 指定の属性を表示 (**read**) して、追加または変更 (**write**) する権限をユーザーに指定するかどうか。パーミッションを指定しないと、**書き込み** パーミッションのみが追加されます。
  - **--attrs**: メンバーグループのユーザーが表示または編集できる属性です。

以下に例を示します。

```
$ ipa delegation-add "basic manager attributes" --permissions=read --permissions=write --
attrs=businesscategory --attrs=departmentnumber --attrs=employeetype --
attrs=employeenumber --group=managers --membergroup=employees
-----------------------------------------
Added delegation "basic manager attributes"
-----------------------------------------
  Delegation name: basic manager attributes
  Permissions: read, write
  Attributes: businesscategory, departmentnumber, employeetype, employeenumber
  Member user group: employees
  User group: managers
```

28.3. IDM CLI を使用した既存の委譲ルールの表示

IdM CLI を使用して既存の委譲ルールを表示するには、次の手順に従います。

前提条件

- **admins** グループのメンバーとしてログインしている。

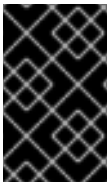
手順

- **ipa delegation-find** コマンドを入力します。

```
$ ipa delegation-find
--------------------
1 delegation matched
--------------------
  Delegation name: basic manager attributes
  Permissions: read, write
  Attributes: businesscategory, departmentnumber, employeenumber, employeetype
  Member user group: employees
  User group: managers
----------------------------
Number of entries returned 1
----------------------------
```

28.4. IDM CLI を使用した委譲ルールの変更

IdM CLI を使用して既存の委譲ルールを変更するには、次の手順に従います。



重要

--attrs オプションはそれまでにサポートされていた属性をすべて上書きするので、新規属性に加えて属性の完全リストを常に含めるようにしてください。これは、**--permissions** オプションにも適用されます。

前提条件

- **admins** グループのメンバーとしてログインしている。

手順

- 必要に応じて、任意の変更を加えて **ipa delegation-mod** コマンドを実行します。たとえば、**displayname** 属性を **basic manager attributes** のルールの例に追加するには、次のコマンドを実行します。

```
$ ipa delegation-mod "basic manager attributes" --attrs=businesscategory --attrs=departmentnumber --attrs=employeetype --attrs=employeenumber --attrs=displayname
---------------------------------------------
Modified delegation "basic manager attributes"
---------------------------------------------
  Delegation name: basic manager attributes
  Permissions: read, write
  Attributes: businesscategory, departmentnumber, employeetype, employeenumber, displayname
  Member user group: employees
  User group: managers
```

28.5. IDM CLI を使用した委譲ルールの削除

IdM CLI を使用して既存の委譲ルールを削除するには、この手順に従います。

前提条件

- **admins** グループのメンバーとしてログインしている。

手順

- **ipa delegation-del** コマンドを入力します。
- プロンプトが表示されたら、削除する委譲ルールの名前を入力します。

```
$ ipa delegation-del
Delegation name: basic manager attributes
---------------------------------------------
Deleted delegation "basic manager attributes"
---------------------------------------------
```

第29章 WEB UI を使用してユーザーグループにパーミッションを委譲してユーザーを管理する手順

委譲は、セルフサービスルールおよびロールベースのアクセス制御 (RBAC) などの IdM のアクセス制御メソッドの 1 つです。委譲を使用して、あるユーザーのグループにパーミッションを割り当てて別のユーザーのグループのエントリーを管理できます。

このセクションでは、以下のトピックについて説明します。

29.1. 委譲ルール

委譲ルール を作成して、ユーザーグループにパーミッションを委譲してユーザーを管理できます。

委譲ルールを使用すると、特定のユーザーグループが、別のユーザーグループ内のユーザーの特定の属性に対して書き込み (編集) 操作を実行できます。このようなアクセス制御ルールは、委譲ルールで指定された属性のサブセットの編集に限定されており、エントリー全体の追加や削除、未指定の属性の制御はできません。

委譲ルールにより、IdM の既存のユーザーグループにパーミッションが付与されます。委任を使用すると、**managers** ユーザーグループで **employees** ユーザーグループでユーザーの選択された属性を管理できます。

29.2. IDM WEBUI を使用した委譲ルールの作成

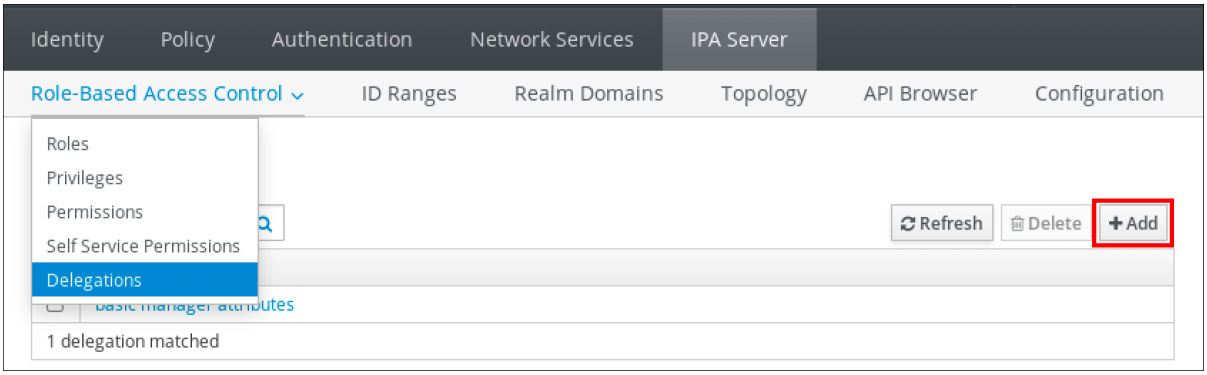
IdM WebUI を使用して委譲ルールを作成するには、次の手順に従います。

前提条件

- **admins** グループのメンバーとして IdM Web UI にログインしている。

手順

1. **IPA Server** メニューから、**Role-Based Access Control** → **Delegations** をクリックします。
2. **Add** をクリックします。



3. **Add delegation** ウィンドウで、以下を実行します。

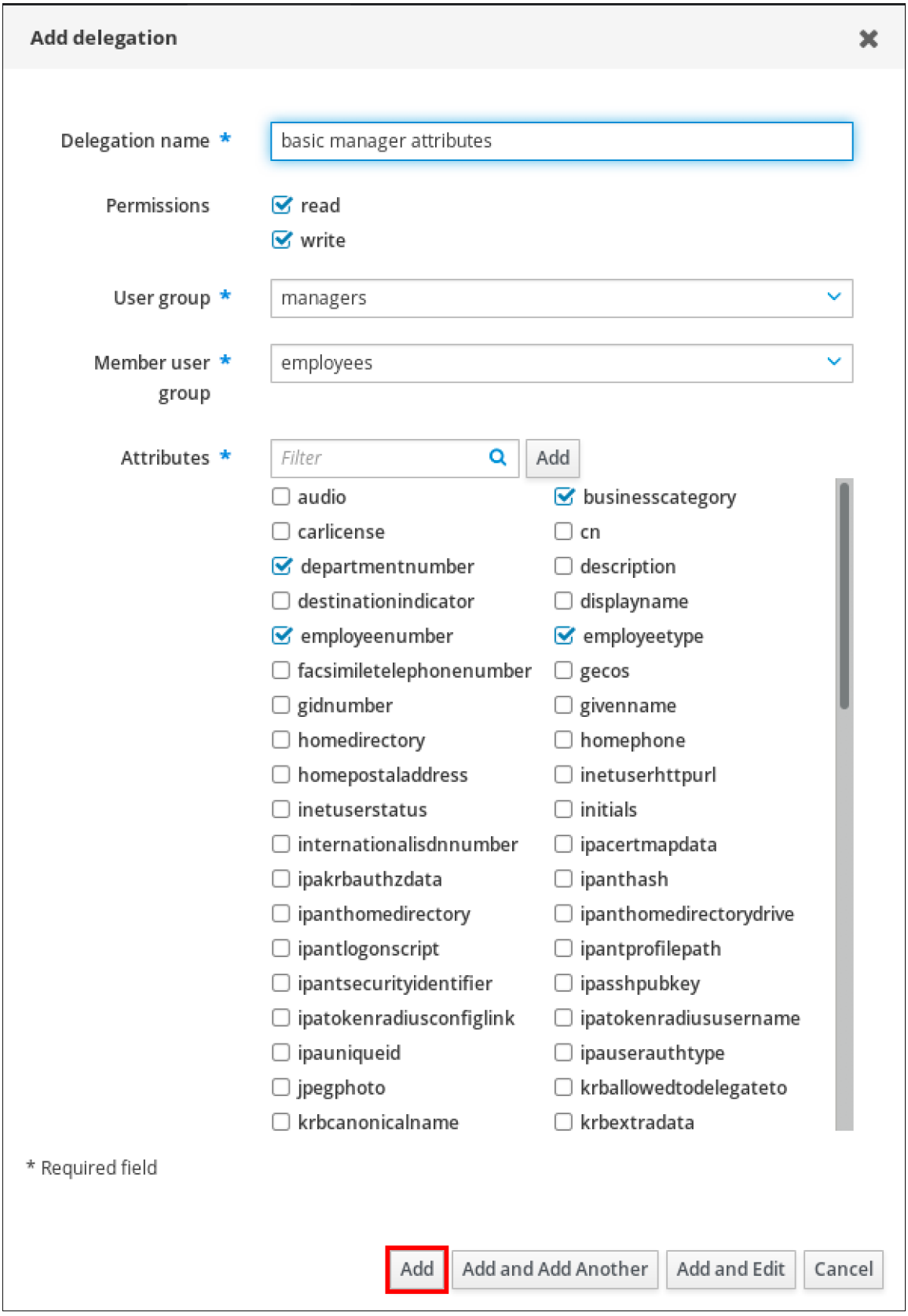
   a. 新しい委譲ルールに名前を付けます。

   b. 指定の属性を表示する権限 (**read**)、および指定の属性を追加または変更する権限 (**write**) をユーザーに指定しているかどうかを示すチェックボックスを選択して、パーミッションを設定します。

   c. ユーザーグループのドロップダウンメニューで、**パーミッションを付与している** グループを選択し、メンバーグループのユーザーエントリーを表示または編集します。

   d. **Member user group** ドロップダウンメニューで、委譲グループのメンバーが **エントリーを編集できる** グループを選択します。

   e. 属性ボックスで、パーミッションを付与する属性のチェックボックスを選択します。



f. **Add** ボタンをクリックして、新規委譲ルールを保存します。

29.3. IDM WEBUI を使用した既存の委譲ルールの表示

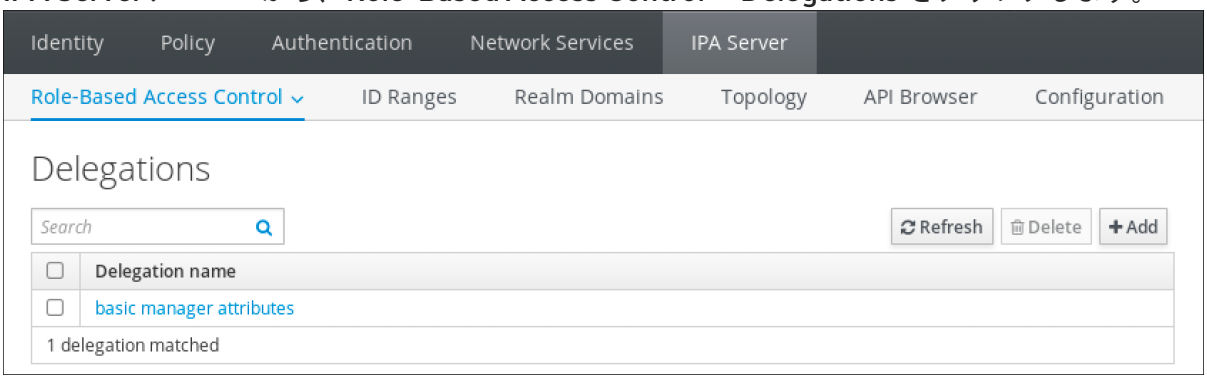
IdM WebUI を使用して既存の委譲ルールを表示するには、次の手順に従います。

前提条件

- **admins** グループのメンバーとして IdM Web UI にログインしている。

手順

- **IPA Server** メニューから、**Role-Based Access Control → Delegations** をクリックします。



29.4. IDM WEBUI を使用した委譲ルールの変更

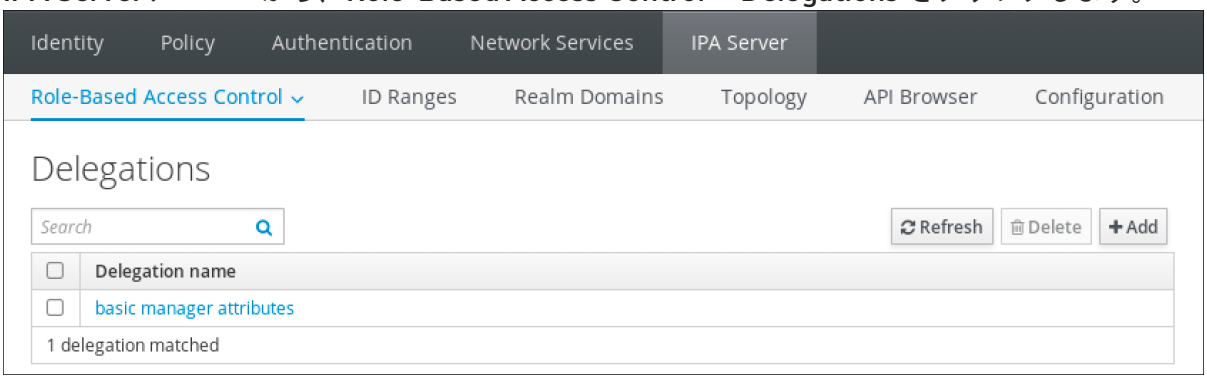
IdM WebUI を使用して既存の委譲ルールを変更するには、次の手順に従います。

前提条件

- **admins** グループのメンバーとして IdM Web UI にログインしている。

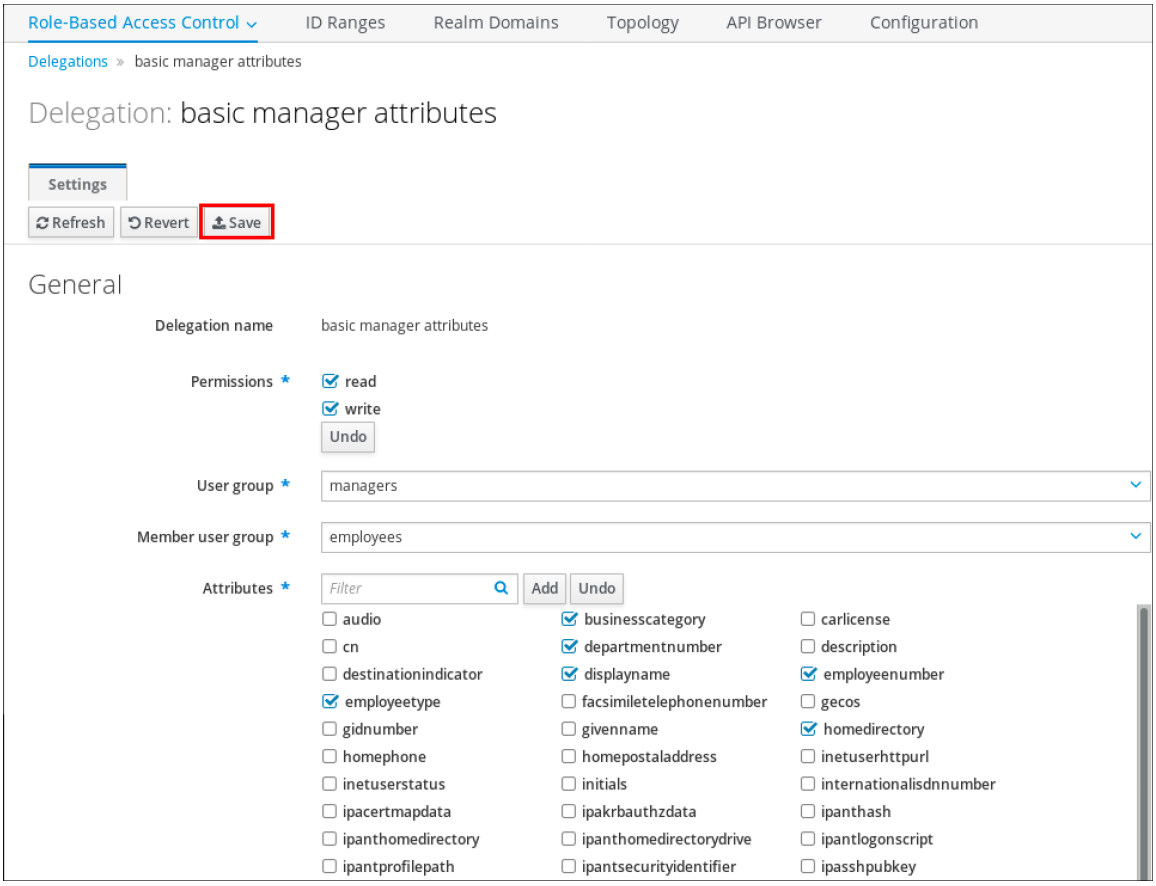
手順

1. **IPA Server** メニューから、**Role-Based Access Control → Delegations** をクリックします。



2. 変更するルールをクリックします。
3. 必要な変更を行います。
   - ルールの名前を変更します。
   - 指定の属性を表示する権限 (**read**)、および指定の属性を追加または変更する権限 (**write**) をユーザーに指定しているかどうかを示すチェックボックスを選択して、付与されたパーミッションを変更します。
   - ユーザーグループのドロップダウンメニューで、**パーミッションを付与している** グループを選択し、メンバーグループのユーザーエントリーを表示または編集します。
   - **Member user group** ドロップダウンメニューで、委譲グループのメンバーが **エントリーを編集できる** グループを選択します。

- 属性ボックスで、パーミッションを付与する属性のチェックボックスを選択します。属性のパーミッションを削除するには、関連するチェックボックスの選択を解除します。



- Save ボタンをクリックして変更を保存します。

29.5. IDM WEBUI を使用した委譲ルールの削除

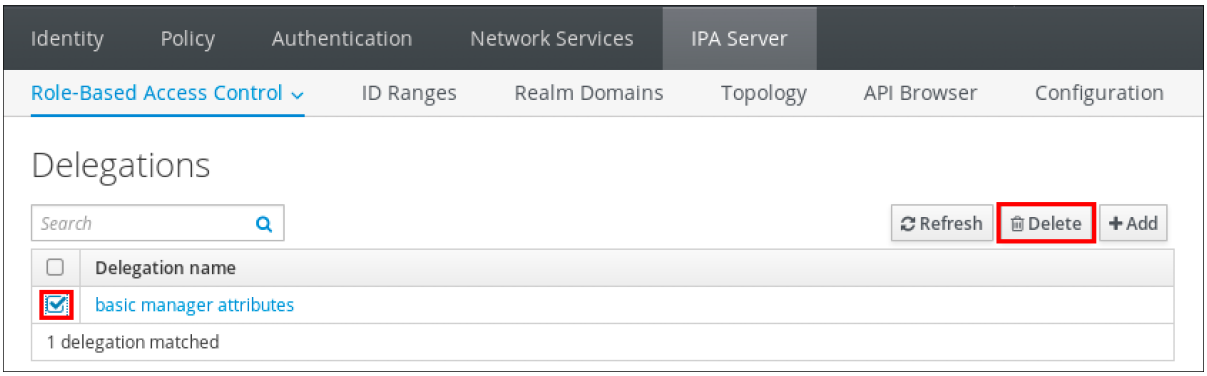
IdM WebUI を使用して既存の委譲ルールを削除するには、次の手順に従います。

前提条件

- **admins** グループのメンバーとして IdM Web UI にログインしている。

手順

1. **IPA Server** メニューから、**Role-Based Access Control** → **Delegations** をクリックします。
2. 削除するルールの横にあるチェックボックスを選択します。
3. **Delete** をクリックします。



4. **Delete** をクリックして確定します。

第30章 ANSIBLE PLAYBOOK を使用してユーザーグループにパーミッションを委譲してユーザーを管理する手順

委譲は、セルフサービスルールおよびロールベースのアクセス制御 (RBAC) などの IdM のアクセス制御メソッドの 1 つです。委譲を使用して、あるユーザーのグループにパーミッションを割り当てて別のユーザーのグループのエントリーを管理できます。

このセクションでは、以下のトピックについて説明します。

- 委譲ルール
- IdM の Ansible インベントリーファイルの作成
- Ansible を使用して委譲ルールを存在させる手順
- Ansible を使用して委譲ルールがないことを確認する手順
- Ansible を使用して委譲ルールに特定の属性を含める手順
- Ansible を使用して委譲ルールに特定の属性を含めないようにする手順

30.1. 委譲ルール

委譲ルール を作成して、ユーザーグループにパーミッションを委譲してユーザーを管理できます。

委譲ルールを使用すると、特定のユーザーグループが、別のユーザーグループ内のユーザーの特定の属性に対して書き込み (編集) 操作を実行できます。このようなアクセス制御ルールは、委譲ルールで指定された属性のサブセットの編集に限定されており、エントリー全体の追加や削除、未指定の属性の制御はできません。

委譲ルールにより、IdM の既存のユーザーグループにパーミッションが付与されます。委任を使用すると、**managers** ユーザーグループで **employees** ユーザーグループでユーザーの選択された属性を管理できます。

30.2. IDM の ANSIBLE インベントリーファイルの作成

Ansible を使用する場合は、ホームディレクトリーに Ansible Playbook 専用のサブディレクトリーを作成して、**/usr/share/doc/ansible-freeipa/*** と **/usr/share/doc/rhel-system-roles/*** サブディレクトリーからコピーして調整できるようにします。この方法には、以下の利点があります。

- すべての Playbook を 1 カ所で見つけることがでる。
- **root** 権限を呼び出さずに Playbook を実行できる。

手順

1. Ansible 設定および Playbook のディレクトリーをホームディレクトリーに作成します。

```
$ mkdir ~/MyPlaybooks/
```

2. **~/MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks
```

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

```
[defaults]
inventory = /home/<username>/MyPlaybooks/inventory

[privilege_escalation]
become=True
```

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

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

[us]
replica.idm.example.com

[ipaserver:children]
eu
us
```

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

30.3. ANSIBLE を使用して委譲ルールを存在させる手順

以下の手順では、Ansible Playbook を使用して、新しい IdM 委譲ルールの特権を定義して、その存在を確認する方法を説明します。この例では、新しい **basic manager attributes** 委譲ルールにより、**managers** グループが **employees** グループのメンバーに対して以下の属性の読み取りと書き込みを行うことができます。

- **businesscategory**
- **departmentnumber**
- **employeenumber**
- **employeetype**

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
  - この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。

- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

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

2. **/usr/share/doc/ansible-freeipa/playbooks/delegation/** にある **delegation-present.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/delegation/delegation-present.yml
delegation-present-copy.yml
```

3. Ansible Playbook ファイル **delegation-present-copy.yml** を開きます。

4. **ipadelegation** タスクセクションに以下の変数を設定して、ファイルを調整します。

   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は新しい委譲ルールの名前に設定します。
   - **permission** 変数は、付与するパーミッションをコンマ区切りのリスト (**read** および **write**) で設定します。
   - **attribute** 変数は、委譲されたユーザーグループが管理できる属性のリスト (**businesscategory**、**departmentnumber**、**employeenumber** および **employeetype**) に変数を設定します。
   - **group** 変数は、属性の表示や変更権限を付与したグループの名前に設定します。
   - **membergroup** 変数は、属性の表示または変更が可能なグループ名に設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Playbook to manage a delegation rule
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure delegation "basic manager attributes" is present
    ipadelegation:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: "basic manager attributes"
      permission: read, write
      attribute:
      - businesscategory
      - departmentnumber
      - employeenumber
      - employeetype
      group: managers
      membergroup: employees
```

5. ファイルを保存します。

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

```
$ ansible-playbook --vault-password-file=password_file -v -i ~/MyPlaybooks/inventory
delegation-present-copy.yml
```

関連情報

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

30.4. ANSIBLE を使用して委譲ルールがないことを確認する手順

以下の手順では、Ansible Playbook を使用して、指定した委譲ルールが IdM 設定に存在しないことを確認する方法を説明します。以下の例では、カスタムの **basic manager attributes** 委譲ルールが IdM に存在しないことを確認する方法を説明します。

前提条件

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

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

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

2. **/usr/share/doc/ansible-freeipa/playbooks/delegation/** ディレクトリーにある **delegation-absent.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/delegation/delegation-present.yml
delegation-absent-copy.yml
```

3. Ansible Playbook ファイル **delegation-absent-copy.yml** を開きます。
4. **ipadelegation** タスクセクションに以下の変数を設定して、ファイルを調整します。
   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は委譲ルールの名前に設定します。
   - **state** 変数は **absent** に設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Delegation absent
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure delegation "basic manager attributes" is absent
    ipadelegation:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: "basic manager attributes"
      state: absent
```

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

```
$ ansible-playbook --vault-password-file=password_file -v -i ~/MyPlaybooks/inventory
delegation-absent-copy.yml
```

関連情報

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

30.5. ANSIBLE を使用して委譲ルールに特定の属性を含める手順

以下の手順では、Ansible Playbook を使用して、委譲ルールに特定の設定を指定する方法を説明します。この Playbook を使用して、以前に作成した委譲ロールを変更できます。この例では、**basic manager attributes** 委譲ルールに **departmentnumber** メンバー属性のみが含まれるようにします。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。

- Ansible バージョン 2.14 以降を使用している。
- Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
- **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
- この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。

- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。
- **basic manager attributes** 委譲ルールが IdM に存在する。

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

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

2. **/usr/share/doc/ansible-freeipa/playbooks/delegation/** にある **delegation-member-present.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/delegation/delegation-member-present.yml delegation-member-present-copy.yml
```

3. Ansible Playbook ファイル **delegation-member-present-copy.yml** を開きます。
4. **ipadelegation** タスクセクションに以下の変数を設定して、ファイルを調整します。
   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は、変更する委譲ルールの名前に設定します。
   - **attribute** 変数は **departmentnumber** に設定します。
   - **action** 変数は **member** に設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Delegation member present
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure delegation "basic manager attributes" member attribute departmentnumber
is present
    ipadelegation:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: "basic manager attributes"
      attribute:
      - departmentnumber
      action: member
```

5. ファイルを保存します。

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

```
$ ansible-playbook --vault-password-file=password_file -v -i ~/MyPlaybooks/inventory
delegation-member-present-copy.yml
```

関連情報

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

30.6. ANSIBLE を使用して委譲ルールに特定の属性を含めないようにする手順

以下の手順では、Ansible Playbook を使用して、委譲ルールに特定の設定が割り当てられないようにする方法を説明します。この Playbook を使用して、委譲ロールが不必要なアクセス権限を付与しないようします。この例では、**basic manager attributes** 委譲ルールに **employeenumber** および **employeetype** メンバー属性が含まれないようにします。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
  - この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。
- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。
- **basic manager attributes** 委譲ルールが IdM に存在する。

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

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

2. **/usr/share/doc/ansible-freeipa/playbooks/delegation/** にある **delegation-member-absent.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/delegation/delegation-member-absent.yml delegation-member-absent-copy.yml
```

3. Ansible Playbook ファイル **delegation-member-absent-copy.yml** を開きます。

4. **ipadelegation** タスクセクションに以下の変数を設定して、ファイルを調整します。

   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は、変更する委譲ルールの名前に設定します。
   - **attribute** 変数は **employeenumber** および **employeetype** に設定します。
   - **action** 変数は **member** に設定します。
   - **state** 変数は **absent** に設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Delegation member absent
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure delegation "basic manager attributes" member attributes employeenumber and employeetype are absent
    ipadelegation:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: "basic manager attributes"
      attribute:
      - employeenumber
      - employeetype
      action: member
      state: absent
```

5. ファイルを保存します。

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

```
$ ansible-playbook --vault-password-file=password_file -v -i ~/MyPlaybooks/inventory delegation-member-absent-copy.yml
```

関連情報

- 委譲ルール を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-delegation.md** ファイルを参照してください。

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

第31章 CLI で IDM でのロールベースのアクセス制御の管理

Identity Management (IdM) のロールベースのアクセス制御と、コマンドラインインターフェイス (CLI) で実行する次の操作について詳しく説明します。

- パーミッションの管理
- 特権管理
- ロールの管理

31.1. IDM のロールベースのアクセス制御

IdM のロールベースアクセス制御 (RBAC) は、セルフサービス制御および委譲アクセス制御とは非常に異なる種類の権限をユーザーに付与します。

ロールベースのアクセス制御は、以下の 3 つの部分で設定されます。

- **パーミッション** は、ユーザーの追加または削除、グループの変更、読み取りアクセスの有効化など、特定のタスクを実行する権限を付与します。
- **特権** は、新規ユーザーの追加に必要な全権限など、権限を組み合わせます。
- **ロール** は、ユーザー、ユーザーグループ、ホスト、またはホストグループに特権のセットを付与します。

31.1.1. IdM のパーミッション

パーミッションは、ロールベースのアクセス制御の中で最も低いレベルの単位で、操作を適用する LDAP エントリーと合わせて操作を定義します。ブロックの構築と同様に、パーミッションは必要に応じて多くの権限に割り当てることができます。
1 つ以上の **権限** を使用して、許容される操作を定義します。

- **write**
- **read**
- **search**
- **compare**
- **add**
- **delete**
- **all**

上記の操作は、3 つの基本的な **ターゲット** に適用されます。

- **subtree**: ドメイン名 (DN) (この DN のサブツリー)
- **target filter**: LDAP フィルター
- **target**: DN。ワイルドカードでエントリーを指定可能。

また、以下の便利なオプションは、対応する属性を設定します。

- **type**: オブジェクトのタイプ (ユーザー、グループなど) (**subtree** および **target filter** を設定します)。
- **memberof**: グループのメンバー。**target filter** を設定します。
- **targetgroup**: 特定のグループを変更する権限 (グループメンバーシップの管理権限の付与など) を付与します (**target** を設定します)。

IdM パーミッションを使用すると、どのユーザーがどのオブジェクトにアクセスできるか、さらにこのようなオブジェクトの属性にアクセスできるかどうかを制御できます。IdM を使用すると、個別の属性を許可または拒否したり、ユーザー、グループ、sudo などの特定の IdM 機能を、全匿名ユーザー、全認証済みユーザー、または特定の特権ユーザーグループ限定などと、全体的な表示設定を変更したりできます。
たとえば、このアプローチではパーミッション指定に柔軟性があるので、アクセスが必要な特定のセクションのみにユーザーまたはグループのアクセスを制限し、他のセクションをこれらのユーザーまたはグループには完全に表示されないように設定する場合に、管理者にとって便利です。

注記

パーミッションには他のパーミッションを含めることはできません。

31.1.2. デフォルトの管理パーミッション

管理パーミッションは、IdM にデフォルトで含まれているパーミッションです。このパーミッションはユーザーが作成した他のパーミッションと同様に機能しますが、以下の相違点があります。

- この管理パーミッションは削除できず、名前、場所、ターゲットの属性を変更できません。
- このパーミッションには 3 つの属性セットがあります。
  - **デフォルト** の属性。IdM で管理されているため、ユーザーは変更できません。
  - **包含** 属性。ユーザーが別途追加する属性。
  - **除外** 属性。ユーザーが削除する属性。

管理パーミッションは、デフォルトおよび包含属性セットに表示されている属性すべてに適用されますが、除外セットに表示されている属性には適用されません。

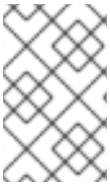
注記

管理パーミッションを削除できませんが、パーミッションにバインドタイプを設定し、すべての特権から管理パーミッションを削除して管理パーミッションを効果的に無効にできます。

管理パーミッションの名前はすべて **System:** から始まります (例: **System: Add Sudo rule** または **System: Modify Services**)。以前のバージョンの IdM では、デフォルトのパーミッションに異なるスキームを使用していました。たとえば、ユーザーはパーミッションの削除はできず、特権に割り当てるしかできませんでした。これらのデフォルトパーミッションのほとんどは、管理パーミッションに切り替わっていますが、以下のパーミッションは引き続き以前のスキームを使用します。

- Automember Rebuild メンバーシップタスクの追加
- 設定サブエントリーの追加
- レプリカ合意の追加

- 証明書削除保留
- CA から証明書のステータス取得
- DNA 範囲の読み取り
- DNA 範囲の変更
- PassSync Manager の設定の読み取り
- PassSync Manager 設定の変更
- レプリカ合意の読み込み
- レプリカ合意の修正
- レプリカ合意の削除
- LDBM データベース設定の読み取り
- 証明書の要求
- CA ACL を無視する証明書の要求
- 別のホストからの証明書の要求
- CA からの証明書の取得
- 証明書の取り消し
- IPA 設定の書き込み



注記

コマンドラインから管理パーミッションを変更しようとし、変更不可な属性の変更をシステム側が許可しない場合には、コマンドに失敗します。Web UI から管理パーミッションを変更しようとした場合には、変更できない属性が無効になります。

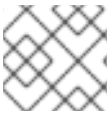
31.1.3. IdM の特権

特権は、ロールに適用されるパーミッションのグループです。
パーミッションは単一の操作を実行する権限を提供しますが、IdM タスクを成功させるには、複数のパーミッションが必要なものがあります。したがって、特権は、特定のタスクを実行するために必要な異なるパーミッションを組み合わせたものです。
たとえば、新しい IdM ユーザーにアカウントを設定するには、以下の権限が必要です。

- 新規ユーザーエントリーの作成
- ユーザーパスワードのリセット
- 新規ユーザーのデフォルト IPA ユーザーグループへの追加

これらの 3 つの低レベルのタスクを、**ユーザーの追加** という名前のカスタム特権の形式で、権限がより高いレベルのタスクに組み合わせることで、システム管理者はロールを管理しやすくなります。IdM には、すでにいくつかのデフォルト特権が含まれています。ユーザーとユーザーグループとは別に、権

限はホストおよびホストグループ、およびネットワークサービスにも割り当てられます。これにより、特定のネットワークサービスを使用するホストセットのユーザーセットによって、操作をきめ細かく制御できます。

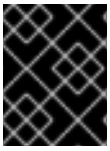


注記

特権には、他の特権を含めることはできません。

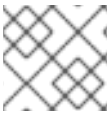
31.1.4. IdM のロール

ロールは、ロールに指定したユーザーが所有する権限のリストです。
実際には、パーミッションでは、指定の低階層のタスク (ユーザーエントリーの作成、グループへのエントリーの追加など) を実行する権限を付与し、特権では、高階層のタスク (指定のグループへの新規ユーザーの作成など) に必要なこれらのパーミッションの 1 つ以上を組み合わせます。ロールは必要に応じて、管理者ロールでユーザーの追加、変更、削除ができるなど、特権をまとめます。



重要

ロールは、許可されたアクションを分類するために使用されます。ロールは、特権昇格されないようにしたり、特権の分離を実装するツールとしては使用しません。



注記

ロールに他のロールを含めることはできません。

31.1.5. Identity Management で事前定義されたロール

Red Hat Identity Management には、以下の事前定義済みのロールが含まれています。

表31.1 Identity Management の定義済みロール

ロール	特権	説明
登録管理者	ホストの登録	クライアントまたはホストの登録を行います。
helpdesk	Modify Users、Reset passwords、Modify Group membership	簡単なユーザー管理タスクを実行します。
IT Security Specialist	Netgroups Administrators、HBAC Administrator、Sudo Administrator	ホストベースのアクセス制御、sudo ルールなどのセキュリティーポリシーを管理します。
IT Specialist	Host Administrators、Host Group Administrators、Service Administrators、Automount Administrators	ホストの管理を行います

ロール	特権	説明
Security Architect	Delegation Administrator、Replication Administrators、Write IPA Configuration、Password Policy Administrator	Identity Management 環境の管理、信頼の作成、レプリカ合意を作成します。
User Administrator	User Administrators、Group Administrators、Stage User Administrators	ユーザーおよびグループの作成を行います

31.2. CLI での IDM パーミッションの管理

コマンドラインインターフェイス (CLI) を使用して Identity Management (IdM) のパーミッションを管理するには、次の手順に従います。

前提条件

- IdM、または **ユーザー管理者** ロールを管理する管理者権限
- 有効な Kerberos チケット。詳細は、Using kinit to log in to IdM manually を参照してください。

手順

1. **ipa permission-add** コマンドを使用して、新しいパーミッションエントリーを作成します。たとえば、**dns admin** という名前のパーミッションを追加するには、次のコマンドを実行します。

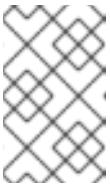
```
$ ipa permission-add "dns admin"
```

2. 以下のオプションでパーミッションのプロパティーを指定します。

- **--bindtype** は、バインドルールの種別を指定します。このオプションでは、**all**、**anonymous** および **permission** 引数を使用できます。**permission** バインドタイプは、ロールを使用してこのパーミッションを付与されたユーザーのみが実行できます。以下に例を示します。

```
$ ipa permission-add "dns admin" --bindtype=all
```

--bindtype を指定しないと、**permission** がデフォルト値になります。



注記

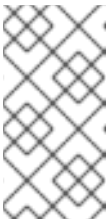
特権には、デフォルト以外のバインドルールタイプが指定されたパーミッションを追加できません。特権に既存のパーミッションは、デフォルト以外のバインドルールタイプには設定できません。

- **--right** は、パーミッションが付与する権限をリスト表示します。これは、非推奨の **--permissions** オプションに代わるものです。使用できる値は **add**、**delete**、**read**、**search**、**compare**、**write**、**all** になります。

複数の属性を設定するには、複数の **--right** オプションを使用するか、中括弧内にコンマ区切りリストを使用します。以下に例を示します。

```
$ ipa permission-add "dns admin" --right=read --right=write
```

```
$ ipa permission-add "dns admin" --right={read,write}
```



注記

add および **delete** はエントリーレベルの操作 (ユーザーの削除、グループの追加など) ですが、**read**、**search**、**compare**、**write** は、属性レベル (**userCertificate** への書き込み可、**userPassword** の読み込み不可) に近いです。

- **--attrs** は、パーミッションが付与される属性のリストを提供します。
複数の属性を設定するには、複数の **--attrs** オプションを使用するか、オプションを中括弧内にコンマ区切りリストでリスト表示します。以下に例を示します。

```
$ ipa permission-add "dns admin" --attrs=description --attrs=automountKey
```

```
$ ipa permission-add "dns admin" --attrs={description,automountKey}
```

--attrs で指定の属性が存在し、指定のオブジェクトタイプで使用可能な属性である必要があります。この条件を満たさない場合には、コマンドがスキーマ構文エラーで失敗します。

- **--type** は、ユーザー、ホスト、サービスなどのパーミッションが適用されるエントリーオブジェクトタイプを定義します。各タイプには、独自の許可属性セットがあります。
以下に例を示します。

```
$ ipa permission-add "manage service" --right=all --type=service --attrs=krbprincipalkey --attrs=krbprincipalname --attrs=managedby
```

- **--subtree** ではサブツリーエントリーを指定します。フィルターはこのサブツリーエントリーの配下にあるエントリーを対象とします。既存のサブツリーエントリーを指定します。**--subtree** ではワイルドカードや存在しないドメイン名 (DN) は使用できません。ディレクトリーに DN を追加します。
IdM は簡素化されたフラットディレクトリーツリー構造を使用しているため、**--subtree** を使用して自動マウントの場所 (他の設定のコンテナーまたは親エントリー) など、一部のエントリーを対象にできます。以下に例を示します。

```
$ ipa permission-add "manage automount locations" --subtree="ldap://ldap.example.com:389/cn=automount,dc=example,dc=com" --right=write --attrs=automountmapname --attrs=automountkey --attrs=automountInformation
```

注記

--type オプションおよび **--subtree** オプションを同時に使用できません。**--subtree** を簡素化したものとして、**--type** のフィルターに含まれている内容を確認できます (これにより、管理者の作業が簡単になります)。

- **--filter** は LDAP フィルターを使用して、パーミッションが適用されるエントリーを特定し

ます。
IdM は、指定のフィルターの有効性を自動的に確認します。このフィルターには、有効な LDAP フィルターを使用できます。以下に例を示します。

```
$ ipa permission-add "manage Windows groups" --filter="(!(objectclass=posixgroup))" --right=write --attrs=description
```

- **--memberof** は、グループが存在することを確認した後に、指定したグループのメンバーにターゲットフィルターを設定します。たとえば、このパーミッショを持つユーザーがエンジニアリンググループのメンバーのログインシェルを変更できるようにするには、以下を実行します。

```
$ ipa permission-add ManageShell --right="write" --type=user --attr=loginshell --memberof=engineers
```

- **--targetgroup** は、グループが存在することを確認した後に、ターゲットを指定のユーザーグループに設定します。たとえば、このパーミッションを持つグループが、エンジニアリンググループのメンバー属性に (メンバーの追加や削除ができるように) 書き込みできるようにするには、以下を実行します。

```
$ ipa permission-add ManageMembers --right="write" --subtree=cn=groups,cn=accounts,dc=example,dc=test --attr=member --targetgroup=engineers
```

- 必要に応じて、ターゲットドメイン名 (DN) を指定できます。
  - **--target** は、パーミッションを適用する DN を指定します。ワイルドカードは使用できます。
  - **--targetto** は、エントリーの移動先に設定できる DN サブツリーを指定します。
  - **--targetfrom** は、エントリーの移動元に設定できる DN サブツリーを指定します。

31.3. 既存のパーミッションのコマンドオプション

必要に応じて、既存のパーミッションを変更するには、以下のバリアントを使用します。

- 既存のパーミッションを編集するには、**ipa permission-mod** コマンドを使用します。パーミッションの追加と同じコマンドオプションを使用できます。
- 既存のパーミッションを見つけるには、**ipa permission-find** コマンドを使用します。パーミッションの追加と同じコマンドオプションを使用できます。
- 特定のパーミッションを表示するには、**ipa permission-show** コマンドを使用します。**--raw** 引数は、生成される未編集の 389-ds ACI を表示します。以下に例を示します。

```
$ ipa permission-show <permission> --raw
```

- **ipa permission-del** コマンドは、パーミッションを完全に削除します。

関連情報

- **ipa** の man ページを参照してください。

- **ipa help** コマンドを参照してください。

31.4. CLI での IDM 権限の管理

コマンドラインインターフェイス (CLI) を使用して Identity Management (IdM) の特権を管理するには、次の手順に従います。

前提条件

- IdM、または **ユーザー管理者** ロールを管理する管理者権限
- 有効な Kerberos チケット。詳細は、kinit を使用して IdM に手動でログインする を参照してください。
- 既存のパーミッション。パーミッションの詳細は、CLI での IdM パーミッションの管理 を参照してください。

手順

1. **ipa privilege-add** コマンドを使用して、特権エントリーを追加します。
たとえば、名前が **managing filesystems** の特権を説明を付けて追加するには、次のコマンドを実行します。

```
$ ipa privilege-add "managing filesystems" --desc="for filesystems"
```

2. **privilege-add-permission** コマンドを使用して必要なパーミッションを特権グループに割り当てます。
たとえば、パーミッション **managing automount** および **managing ftp services** を、**managing filesystems** 権限に追加するには、次のコマンドを実行します。

```
$ ipa privilege-add-permission "managing filesystems" --permissions="managing automount"
--permissions="managing ftp services"
```

31.5. 既存の特権のコマンドオプション

必要に応じて、既存の特権を変更するには、以下のバリアントを使用します。

- 既存の特権を変更するには、**ipa privilege-mod** コマンドを使用します。
- 既存の特権を見つけるには、**ipa privilege-find** コマンドを使用します。
- 特定の特権を表示するには、**ipa privilege-show** コマンドを使用します。
- **ipa privilege-remove-permission** コマンドは、特権から 1 つ以上のパーミッションを削除します。
- **ipa privilege-del** コマンドは、特権を完全に削除します。

関連情報

- **ipa** の man ページを参照してください。
- **ipa help** コマンドを参照してください。

31.6. CLI での IDM ロールの管理

コマンドラインインターフェイス (CLI) を使用して Identity Management (IdM) のロールを管理するには、次の手順に従います。

前提条件

- IdM、または **ユーザー管理者** ロールを管理する管理者権限
- 有効な Kerberos チケット。詳細は、Using kinit to log in to IdM manually を参照してください。
- 既存の特権。特権の詳細は、CLI での IdM 特権の管理 を参照してください。

手順

1. **ipa role-add** コマンドを使用して、新規ロールエントリーを追加します。

```
$ ipa role-add --desc="User Administrator" useradmin
-----------------------
Added role "useradmin"
-----------------------
Role name: useradmin
Description: User Administrator
```

2. **ipa role-add-privilege** コマンドを使用して、必要な特権をロールに追加します。

```
$ ipa role-add-privilege --privileges="user administrators" useradmin
Role name: useradmin
Description: User Administrator
Privileges: user administrators
----------------------------
Number of privileges added 1
----------------------------
```

3. **ipa role-add-member** コマンドを使用して、必要なメンバーをロールに追加します。使用可能なメンバータイプ: ユーザー、グループ、ホスト、およびホストグループ。
たとえば、**useradmins** という名前のグループを、以前に作成した **useradmin** ロールに追加するには、次のコマンドを実行します。

```
$ ipa role-add-member --groups=useradmins useradmin
Role name: useradmin
Description: User Administrator
Member groups: useradmins
Privileges: user administrators
------------------------
Number of members added 1
------------------------
```

31.7. 既存ロールのコマンドオプション

必要に応じて、既存のロールを変更するには、以下のバリアントを使用します。

- 既存のロールを変更するには、**ipa role-mod** コマンドを使用します。

- 既存のロールを見つけるには、**ipa role-find** コマンドを使用します。
- 特定のロールを表示するには、**ipa role-show** コマンドを使用します。
- ロールからメンバーを削除するには、**ipa role-remove-member** コマンドを使用します。
- **ipa role-remove-privilege** コマンドは、ロールから 1 つ以上の権限を削除します。
- **ipa role-del** コマンドは、ロールを完全に削除します。

関連情報

- **ipa** man ページを参照してください。
- **ipa help** コマンドを参照してください。

第32章 IDM WEB UI を使用したロールベースのアクセス制御の管理

Identity Management (IdM) のロールベースのアクセス制御と、Web インターフェイス (Web UI) で実行する次の操作について詳しく説明します。

- パーミッションの管理
- 特権管理
- ロールの管理

32.1. IDM のロールベースのアクセス制御

IdM のロールベースアクセス制御 (RBAC) は、セルフサービス制御および委譲アクセス制御とは非常に異なる種類の権限をユーザーに付与します。

ロールベースのアクセス制御は、以下の 3 つの部分で設定されます。

- **パーミッション** は、ユーザーの追加または削除、グループの変更、読み取りアクセスの有効化など、特定のタスクを実行する権限を付与します。
- **特権** は、新規ユーザーの追加に必要な全権限など、権限を組み合わせます。
- **ロール** は、ユーザー、ユーザーグループ、ホスト、またはホストグループに特権のセットを付与します。

32.1.1. IdM のパーミッション

パーミッションは、ロールベースのアクセス制御の中で最も低いレベルの単位で、操作を適用する LDAP エントリーと合わせて操作を定義します。ブロックの構築と同様に、パーミッションは必要に応じて多くの権限に割り当てることができます。
1 つ以上の **権限** を使用して、許容される操作を定義します。

- **write**
- **read**
- **search**
- **compare**
- **add**
- **delete**
- **all**

上記の操作は、3 つの基本的な **ターゲット** に適用されます。

- **subtree**: ドメイン名 (DN) (この DN のサブツリー)
- **target filter**: LDAP フィルター
- **target**: DN。ワイルドカードでエントリーを指定可能。

また、以下の便利なオプションは、対応する属性を設定します。

- **type**: オブジェクトのタイプ (ユーザー、グループなど) (**subtree** および **target filter** を設定します)。
- **memberof**: グループのメンバー。**target filter** を設定します。
- **targetgroup**: 特定のグループを変更する権限 (グループメンバーシップの管理権限の付与など) を付与します (**target** を設定します)。

IdM パーミッションを使用すると、どのユーザーがどのオブジェクトにアクセスできるか、さらにこのようなオブジェクトの属性にアクセスできるかどうかを制御できます。IdM を使用すると、個別の属性を許可または拒否したり、ユーザー、グループ、sudo などの特定の IdM 機能を、全匿名ユーザー、全認証済みユーザー、または特定の特権ユーザーグループ限定などと、全体的な表示設定を変更したりできます。
たとえば、このアプローチではパーミッション指定に柔軟性があるので、アクセスが必要な特定のセクションのみにユーザーまたはグループのアクセスを制限し、他のセクションをこれらのユーザーまたはグループには完全に表示されないように設定する場合に、管理者にとって便利です。

注記

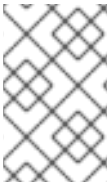
パーミッションには他のパーミッションを含めることはできません。

32.1.2. デフォルトの管理パーミッション

管理パーミッションは、IdM にデフォルトで含まれているパーミッションです。このパーミッションはユーザーが作成した他のパーミッションと同様に機能しますが、以下の相違点があります。

- この管理パーミッションは削除できず、名前、場所、ターゲットの属性を変更できません。
- このパーミッションには 3 つの属性セットがあります。
  - **デフォルト** の属性。IdM で管理されているため、ユーザーは変更できません。
  - **包含** 属性。ユーザーが別途追加する属性。
  - **除外** 属性。ユーザーが削除する属性。

管理パーミッションは、デフォルトおよび包含属性セットに表示されている属性すべてに適用されますが、除外セットに表示されている属性には適用されません。



注記

管理パーミッションを削除できませんが、パーミッションにバインドタイプを設定し、すべての特権から管理パーミッションを削除して管理パーミッションを効果的に無効にできます。

管理パーミッションの名前はすべて **System:** から始まります (例: **System: Add Sudo rule** または **System: Modify Services**)。以前のバージョンの IdM では、デフォルトのパーミッションに異なるスキームを使用していました。たとえば、ユーザーはパーミッションの削除はできず、特権に割り当てるしかできませんでした。これらのデフォルトパーミッションのほとんどは、管理パーミッションに切り替わっていますが、以下のパーミッションは引き続き以前のスキームを使用します。

- Automember Rebuild メンバーシップタスクの追加
- 設定サブエントリーの追加

- レプリカ合意の追加
- 証明書削除保留
- CA から証明書のステータス取得
- DNA 範囲の読み取り
- DNA 範囲の変更
- PassSync Manager の設定の読み取り
- PassSync Manager 設定の変更
- レプリカ合意の読み込み
- レプリカ合意の修正
- レプリカ合意の削除
- LDBM データベース設定の読み取り
- 証明書の要求
- CA ACL を無視する証明書の要求
- 別のホストからの証明書の要求
- CA からの証明書の取得
- 証明書の取り消し
- IPA 設定の書き込み

注記

コマンドラインから管理パーミッションを変更しようとし、変更不可な属性の変更をシステム側が許可しない場合には、コマンドに失敗します。Web UI から管理パーミッションを変更しようとした場合には、変更できない属性が無効になります。

32.1.3. IdM の特権

特権は、ロールに適用されるパーミッションのグループです。
パーミッションは単一の操作を実行する権限を提供しますが、IdM タスクを成功させるには、複数のパーミッションが必要なものがあります。したがって、特権は、特定のタスクを実行するために必要な異なるパーミッションを組み合わせたものです。
たとえば、新しい IdM ユーザーにアカウントを設定するには、以下の権限が必要です。

- 新規ユーザーエントリーの作成
- ユーザーパスワードのリセット
- 新規ユーザーのデフォルト IPA ユーザーグループへの追加

これらの 3 つの低レベルのタスクを、**ユーザーの追加** という名前のカスタム特権の形式で、権限がより高いレベルのタスクに組み合わせることで、システム管理者はロールを管理しやすくなります。IdM には、すでにいくつかのデフォルト特権が含まれています。ユーザーとユーザーグループとは別に、権

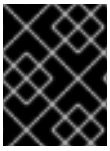
限はホストおよびホストグループ、およびネットワークサービスにも割り当てられます。これにより、特定のネットワークサービスを使用するホストセットのユーザーセットによって、操作をきめ細かく制御できます。

注記

特権には、他の特権を含めることはできません。

32.1.4. IdM のロール

ロールは、ロールに指定したユーザーが所有する権限のリストです。
実際には、パーミッションでは、指定の低階層のタスク (ユーザーエントリーの作成、グループへのエントリーの追加など) を実行する権限を付与し、特権では、高階層のタスク (指定のグループへの新規ユーザーの作成など) に必要なこれらのパーミッションの 1 つ以上を組み合わせます。ロールは必要に応じて、管理者ロールでユーザーの追加、変更、削除ができるなど、特権をまとめます。



重要

ロールは、許可されたアクションを分類するために使用されます。ロールは、特権昇格されないようにしたり、特権の分離を実装するツールとしては使用しません。

注記

ロールに他のロールを含めることはできません。

32.1.5. Identity Management で事前定義されたロール

Red Hat Identity Management には、以下の事前定義済みのロールが含まれています。

表32.1 Identity Management の定義済みロール

ロール	特権	説明
登録管理者	ホストの登録	クライアントまたはホストの登録を行います。
helpdesk	Modify Users、Reset passwords、Modify Group membership	簡単なユーザー管理タスクを実行します。
IT Security Specialist	Netgroups Administrators、HBAC Administrator、Sudo Administrator	ホストベースのアクセス制御、sudo ルールなどのセキュリティーポリシーを管理します。
IT Specialist	Host Administrators、Host Group Administrators、Service Administrators、Automount Administrators	ホストの管理を行います

ロール	特権	説明
Security Architect	Delegation Administrator、Replication Administrators、Write IPA Configuration、Password Policy Administrator	Identity Management 環境の管理、信頼の作成、レプリカ合意を作成します。
User Administrator	User Administrators、Group Administrators、Stage User Administrators	ユーザーおよびグループの作成を行います

32.2. IDM WEB UI でのパーミッションの管理

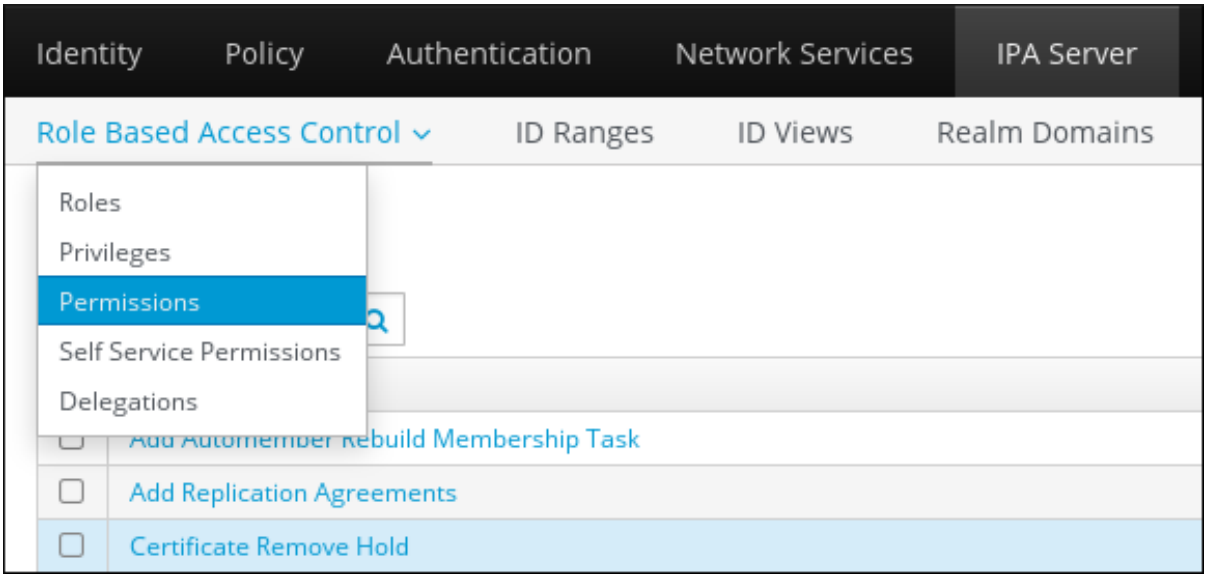
Web インターフェイス (IdM Web UI) を使用して Identity Management (IdM) のパーミッションを管理するには、次の手順に従います。

前提条件

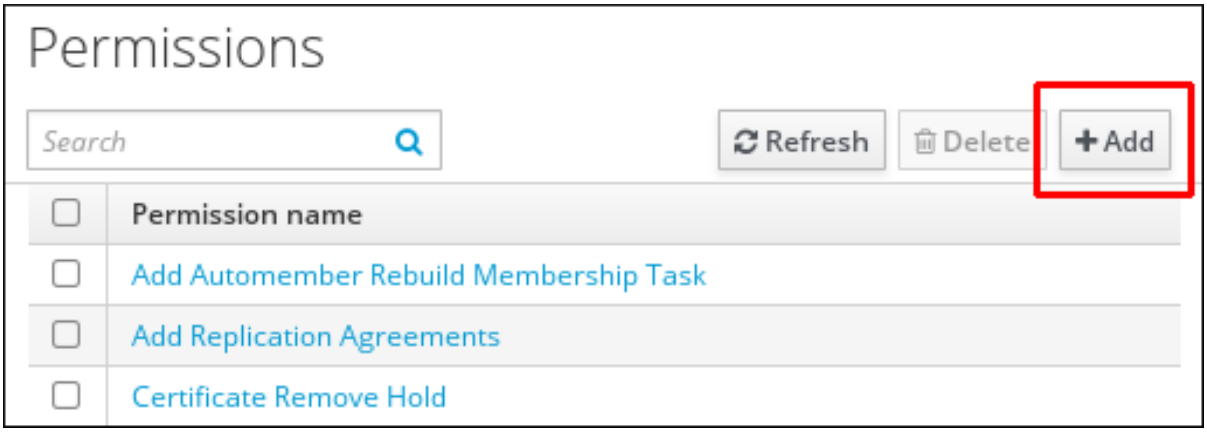
- IdM、または **ユーザー管理者** ロールを管理する管理者権限
- IdM Web UI にログインしている。詳細は、Web ブラウザーでの IdM Web UI へのアクセス を参照してください。

手順

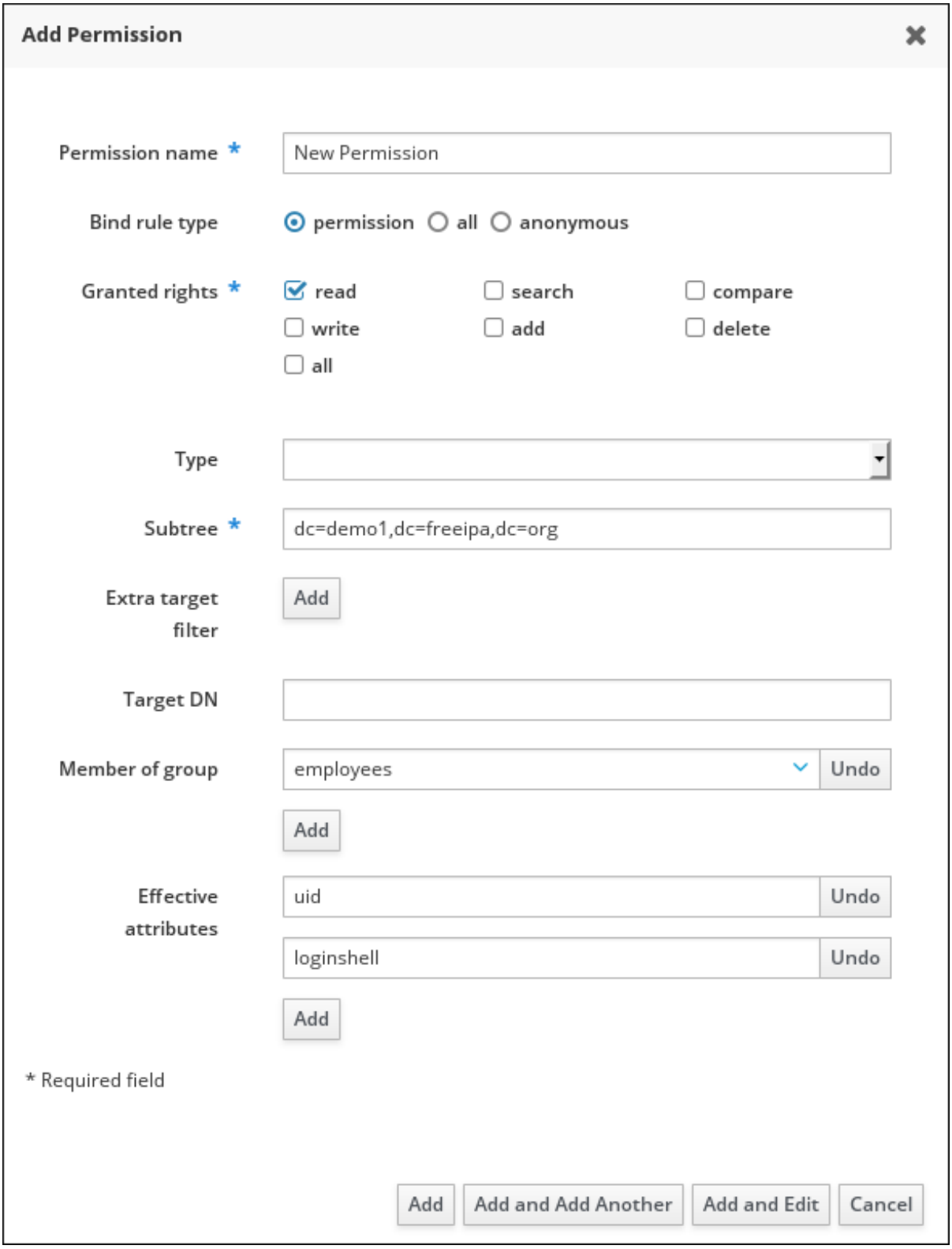
1. 新しいパーミッションを追加するには、**IPA Server** タブで **Role-Based Access Control** サブメニューを開き、**Permissions** を選択します。



2. パーミッションのリストが開きます。パーミッションのリストの上部にある **Add** ボタンをクリックします。

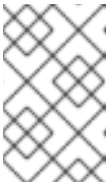


3. **Add Permission** フォームが開きます。新しいパーミッションの名前を指定し、そのプロパティーを適宜定義します。



4. 適切なバインドルールタイプを選択します。

   - **permission** はデフォルトのパーミッションタイプで、権限およびロール経由でアクセスを付与します。
   - **all**: パーミッションを全認証ユーザーに適用することを指定します。
   - **anonymous**: 認証されていないユーザーを含め、すべてのユーザーにパーミッションを適用することを指定します。



注記

特権には、デフォルト以外のバインドルールタイプが指定されたパーミッションを追加できません。特権に既存のパーミッションは、デフォルト以外のバインドルールタイプには設定できません。

5. **Granted rights** でこのパーミッションを付与する権限を選択します。

6. パーミッションのターゲットエントリーを識別する方法を定義します。

   - **Type**: ユーザー、ホスト、またはサービスなどのエントリータイプを指定します。**Type** 設定の値を選択すると、このエントリータイプの ACI でアクセス可能な対応の属性をすべて **Effective Attributes** に表示します。**Type** を定義すると、 **Subtree** および **Target DN** が事前定義された値のいずれかに設定されます。

   - **Subtree** (必須): サブツリーエントリーを指定します。このサブツリーエントリーの下にあるすべてのエントリーが対象になります。**Subtree** ではワイルドカードや存在しないドメイン名 (DN) を使用できないので、既存のサブツリーエントリーを指定します。例: **cn=automount,dc=example,dc=com**

   - **Extra target filter**: DAP フィルターを使用して、パーミッションが適用されるエントリーを特定します。フィルターには、任意の有効な LDAP フィルターを使用できます (たとえば、**(!(objectclass=posixgroup))**) IdM は指定のフィルターの有効性を自動的にチェックします。無効なフィルターを入力して、パーミッションを保存しようとすると、IdM からこの件について警告が表示されます。

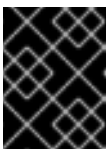
   - **Target DN**: ドメイン名 (DN) を指定し、ワイルドカードを受け入れます。例: **uid=*,cn=users,cn=accounts,dc=com**

   - **Member of group**: 指定したグループのメンバーにターゲットフィルターを設定します。フィルター設定を指定してから **Add** をクリックすると、IdM がフィルターを検証します。すべてのパーミッション設定が正しい場合は、IdM により検索が実行されます。パーミッション設定の一部が正しくない場合には、IdM により、どの設定が正しく設定されているかを示すメッセージが表示されます。

7. パーミッションに属性を追加します。

   - **Type** を設定する場合は、利用可能な ACI 属性のリストから **Effective attributes** を選択します。

   - **Type** を使用しない場合は、**Effective attributes** フィールドに属性を手動で書き込みます。一度に 1 つの属性を追加します。複数の属性を追加するには、**Add** をクリックして別の入力フィールドを追加します。



重要

パーミッションの属性を設定しない場合には、パーミッションはデフォルトですべての属性が含まれます。

8. フォーム下部の **Add** ボタンでパーミッションの追加を完了します。

   - **Add** ボタンをクリックしてパーミッションを保存し、パーミッションのリストに戻ります。

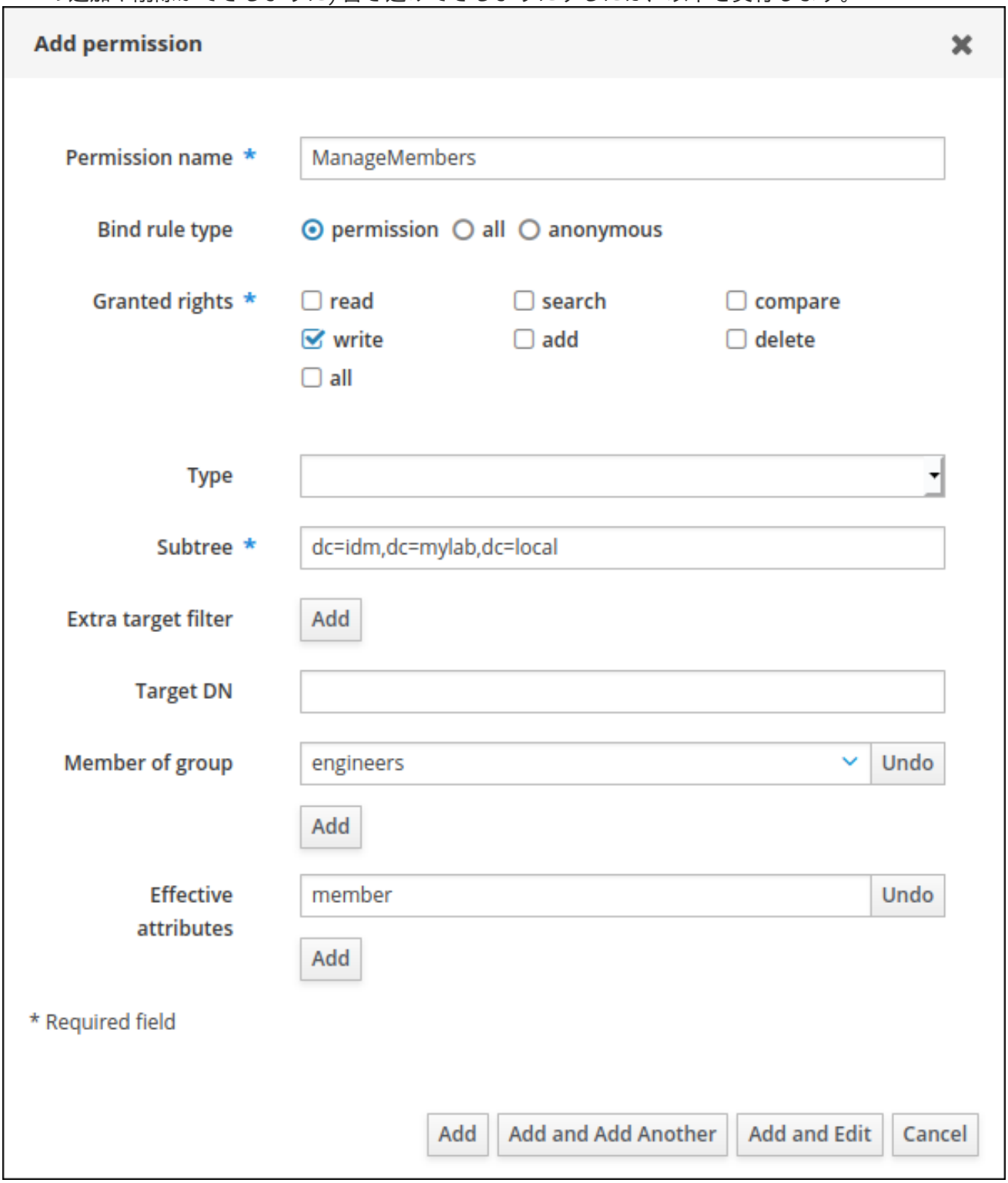
- パーミッションを保存し、**Add and Add another** ボタンをクリックして、継続して同じフォームに別のパーミッションを追加できます。
- **Add and Edit** ボタンを使用すると、新規作成したパーミッションを保存して編集を継続できます。

9. **オプション。**また、パーミッションのリストから名前をクリックして **パーミッション権限** ページを表示し、既存のパーミッションのプロパティーを編集することもできます。
10. **オプション。**既存のパーミッションを削除する必要がある場合は、リストで名前の横にあるチェックボックスにチェックマークを入れて、**Delete** ボタンをクリックして、**Remove permissions** ダイアログを表示します。

注記

デフォルトの管理パーミッションに対する操作は、変更できない属性は IdM Web UI で無効になっており、管理パーミッションを完全に削除することができないんど、制限されています。
ただし、すべての特権から管理されているパーミッションを削除して、パーミッションにバインドタイプが設定されている管理されているパーミッションを効果的に無効にすることができます。

たとえば、このパーミッションを持つグループが、エンジニアリンググループのメンバー属性に (メンバーの追加や削除ができるように) 書き込みできるようにするには、以下を実行します。



32.3. IDM WEBUI での特権の管理

Web インターフェイス (IdM Web UI) を使用して IdM の特権を管理するには、次の手順に従います。

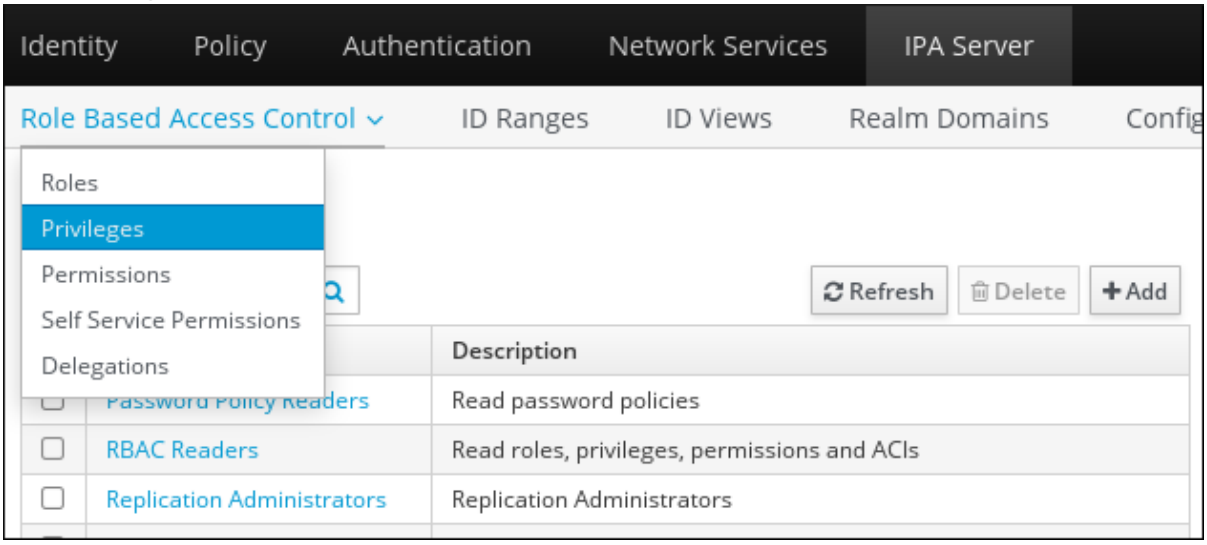
前提条件

- IdM、または **ユーザー管理者** ロールを管理する管理者権限
- IdM Web UI にログインしている。詳細は、Web ブラウザーでの IdM Web UI へのアクセス を参照してください。

- 既存のパーミッション。パーミッションの詳細は、IdM Web UI でのパーミッションの管理 を参照してください。

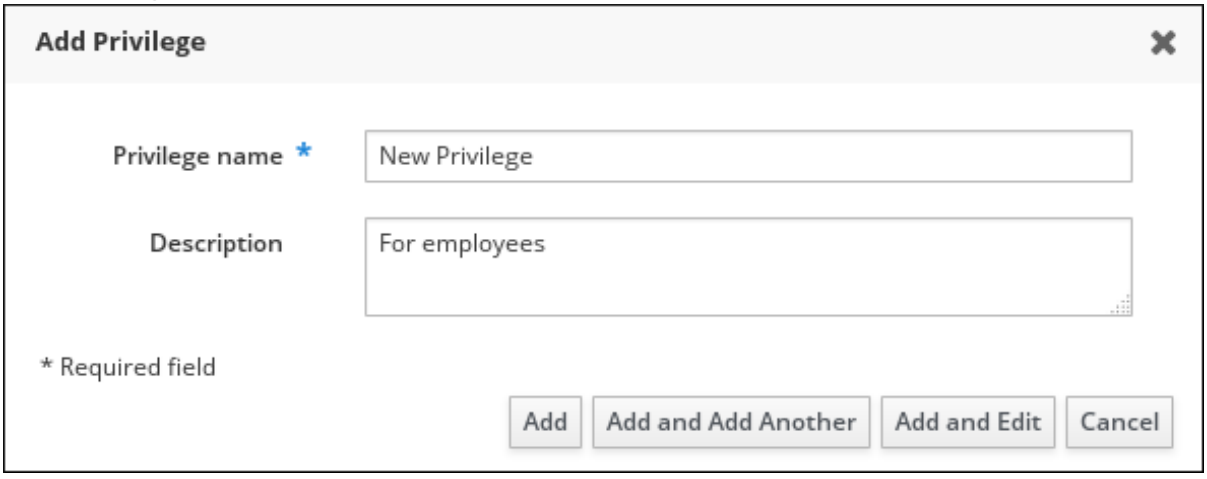
手順

1. 新しい特権を追加するには、**IPA Server** タブで **Role-Based Access Control** サブメニューを開き、**Privileges** を選択します。



2. 特権のリストが開きます。特権リストの上部にある **Add** ボタンをクリックします。

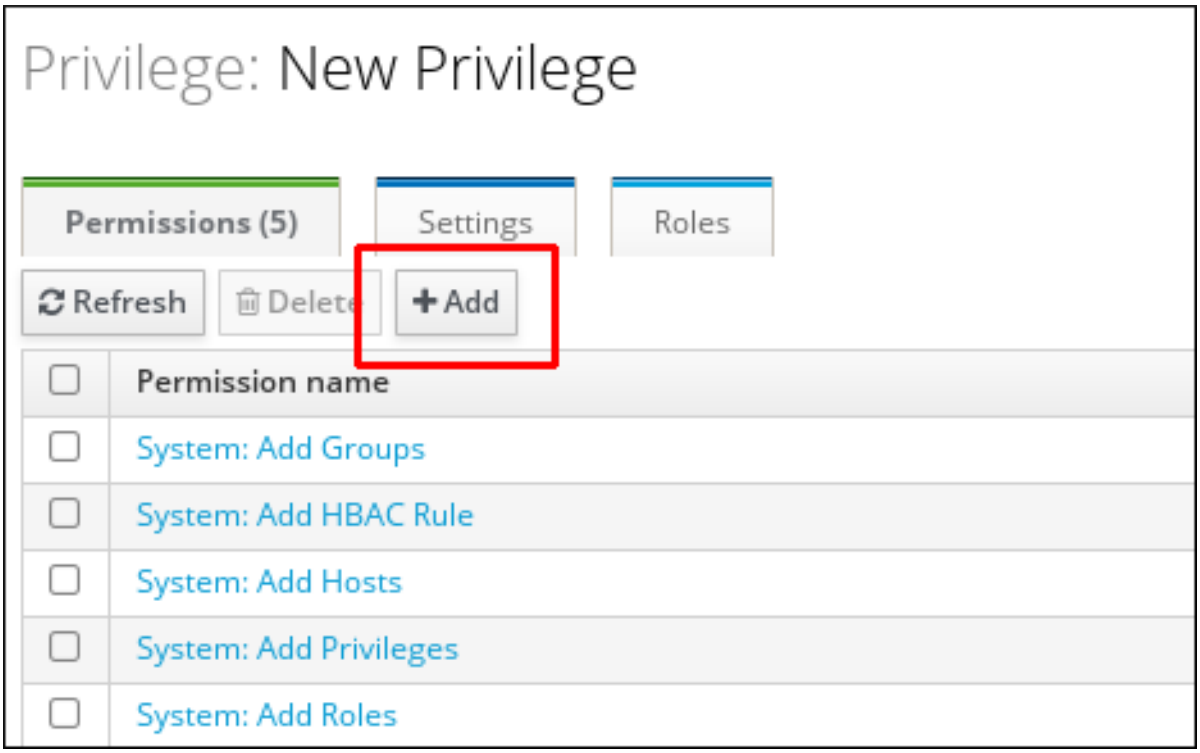
3. **Add Privilege** フォームが開きます。特権の名前と説明を入力します。



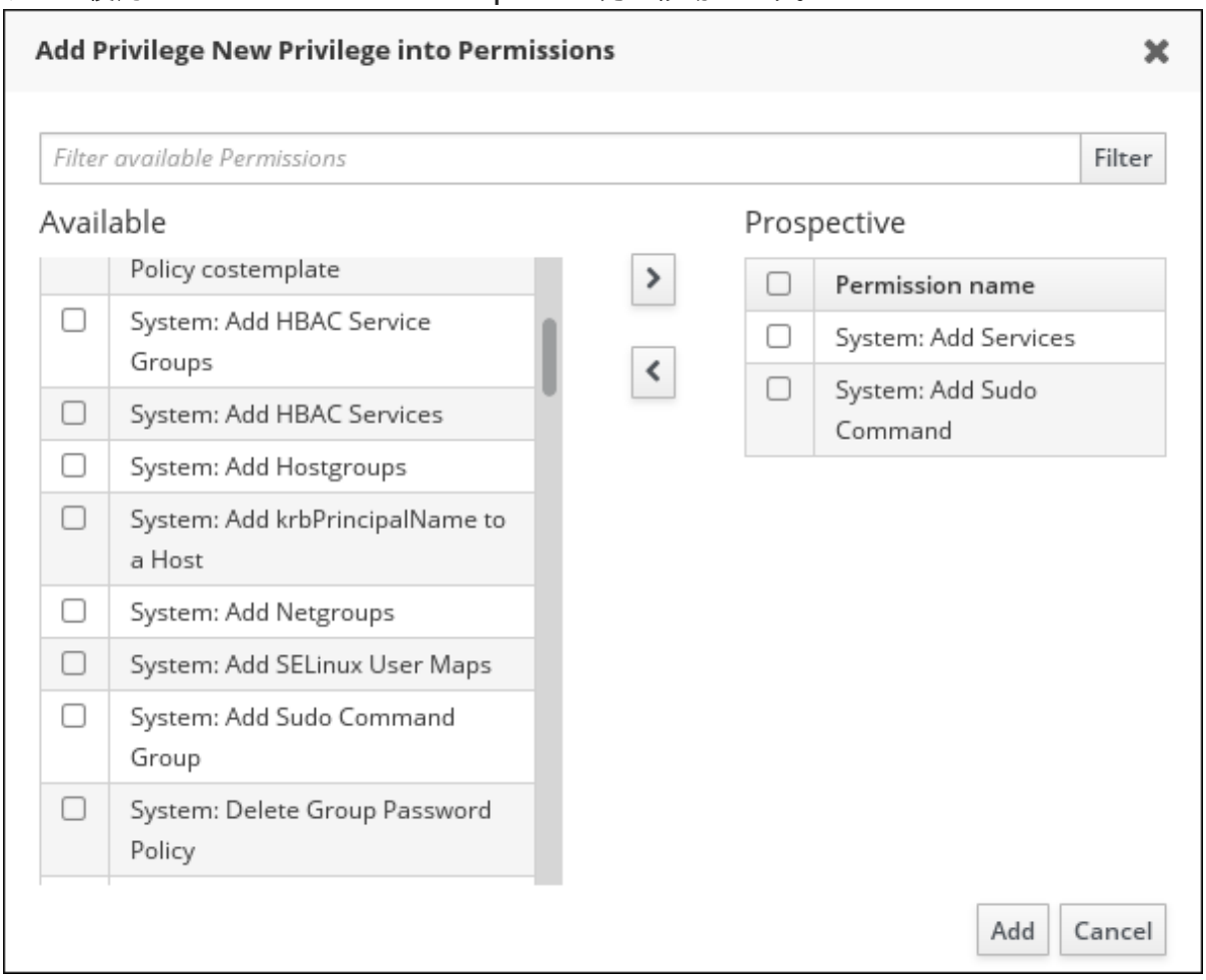
4. 新しい特権を保存し、特権設定ページに移動し、パーミッションを追加するには、**Add and Edit** ボタンをクリックします。

5. 特権リストから特権名をクリックして、特権のプロパティーを編集します。特権設定ページが開きます。

6. **Permissions** タブには、選択した特権に含まれるパーミッションのリストが表示されます。リスト上部の **Add** ボタンをクリックして、パーミッションを特権に追加します。



7. 追加する各パーミッションの名前の横にあるチェックボックスにチェックマークを入れ、> ボタンを使用してパーミッションを **Prospective** 列に移動します。



8. **Add** ボタンをクリックして確定します。

9. **オプション。**パーミッションを削除する必要がある場合は、関連するパーミッションの横にあるチェックボックス (**Remove privileges from permissions**) を表示してから、**Delete** ボタンをクリックします。

10. **オプション。**既存の特権を削除する必要がある場合は、リストで名前の横にあるチェックボックスにチェックを入れて、**Delete** ボタンをクリックすると、**Remove privileges** ダイアログが開きます。

32.4. IDM WEB UI でのロールの管理

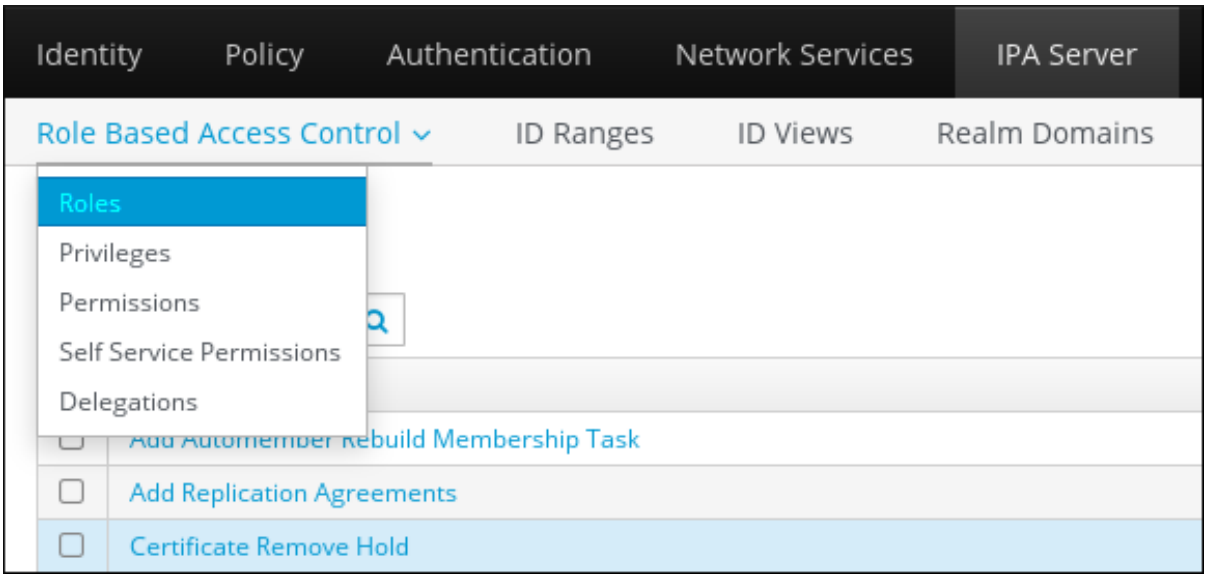
Web インターフェイス (IdM Web UI) を使用して Identity Management (IdM) のロールを管理するには、次の手順に従います。

前提条件

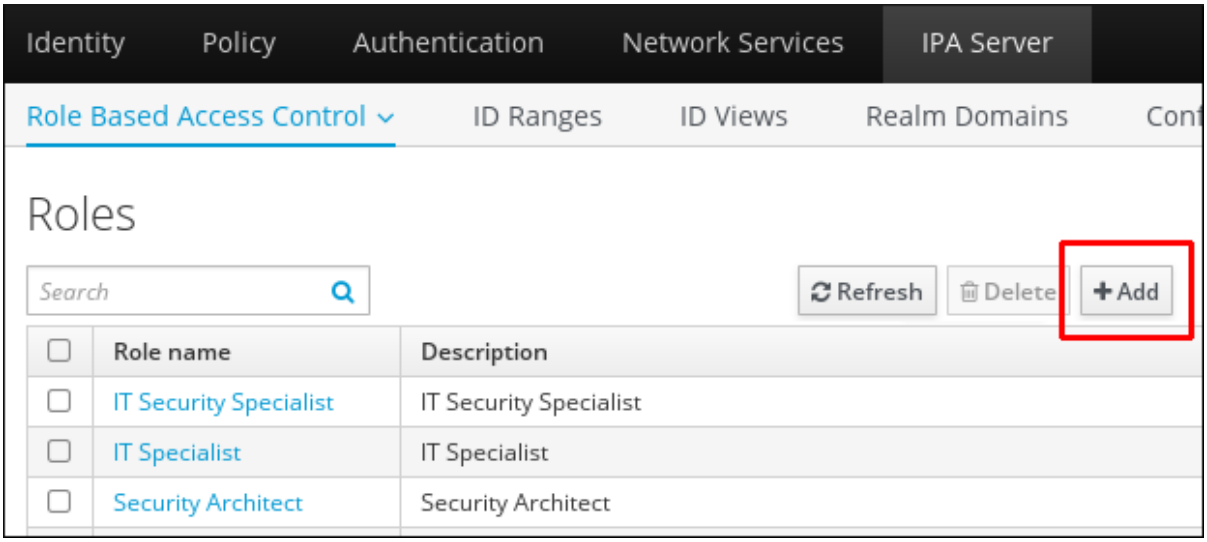
- IdM、または **ユーザー管理者** ロールを管理する管理者権限
- IdM Web UI にログインしている。詳細は、Web ブラウザーでの IdM Web UI へのアクセス を参照してください。
- 既存の特権。権限の詳細は、IdM Web UI で権限の管理 を参照してください。

手順

1. 新規ロールを追加するには、**IPA Server** タブの **Role-Based Access Control** サブメニューを開き、**Roles** を選択します。



2. ロールのリストが開きます。ロールベースのアクセス制御手順のリストの上部にある **Add** ボタンをクリックします。

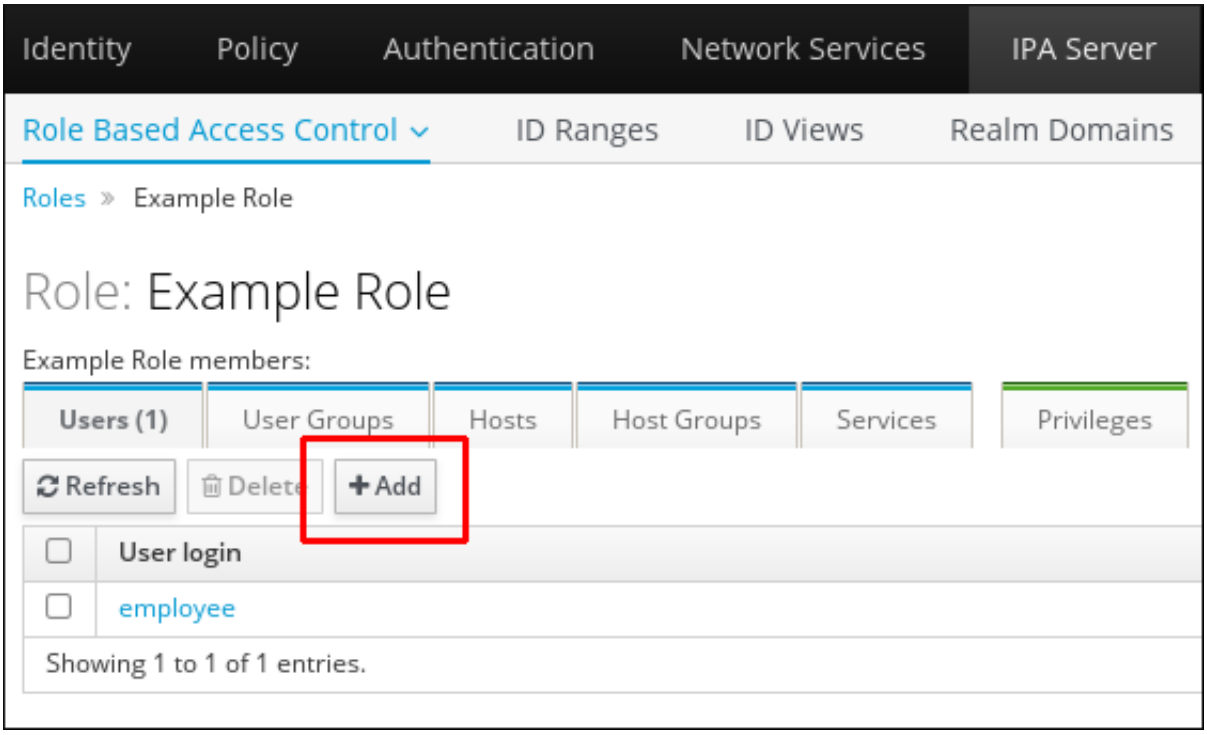


3. **Add Role** フォームが開きます。ロール名と説明を入力します。

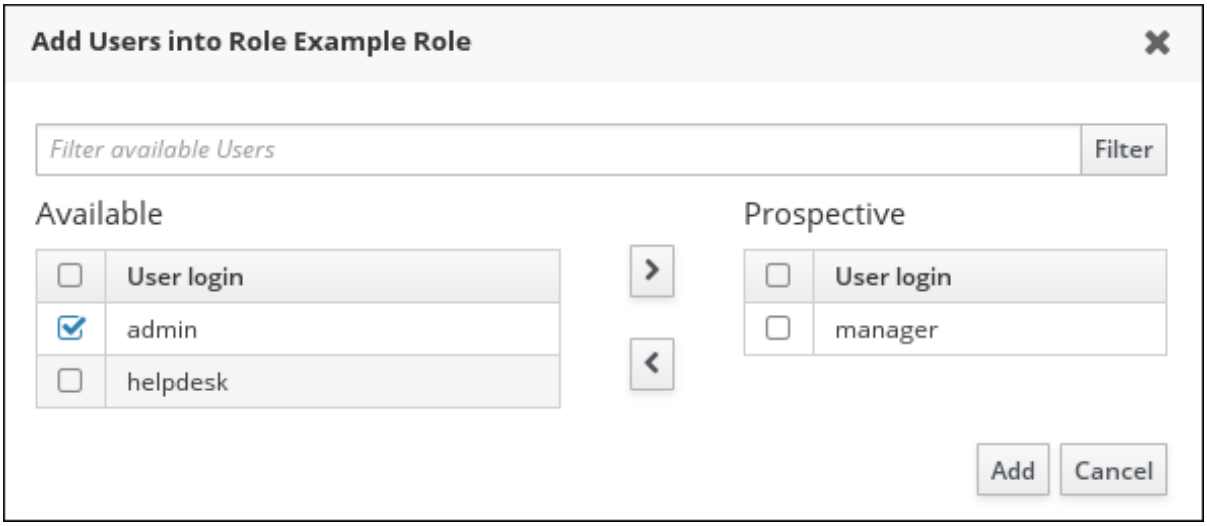
4. **Add and Edit** ボタンをクリックし、新規ロールを保存してロール設定ページに移動し、権限およびユーザーを追加します。

5. ロールリストのロール名をクリックして、ロールのプロパティーを編集します。ロール設定ページが開きます。

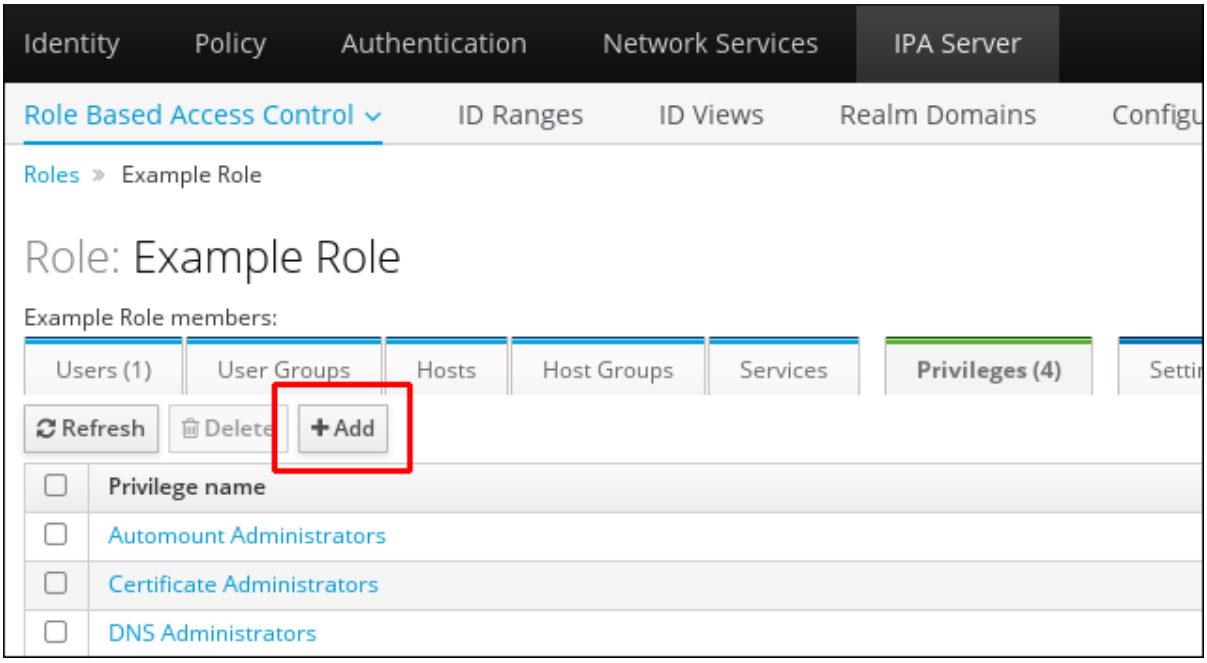
6. 関連するリストの上部にある **Add** ボタンをクリックして、**Users**、**Users Groups**、**Hosts**、**Host Groups**、または **Services** タブを使用してメンバーを追加します。



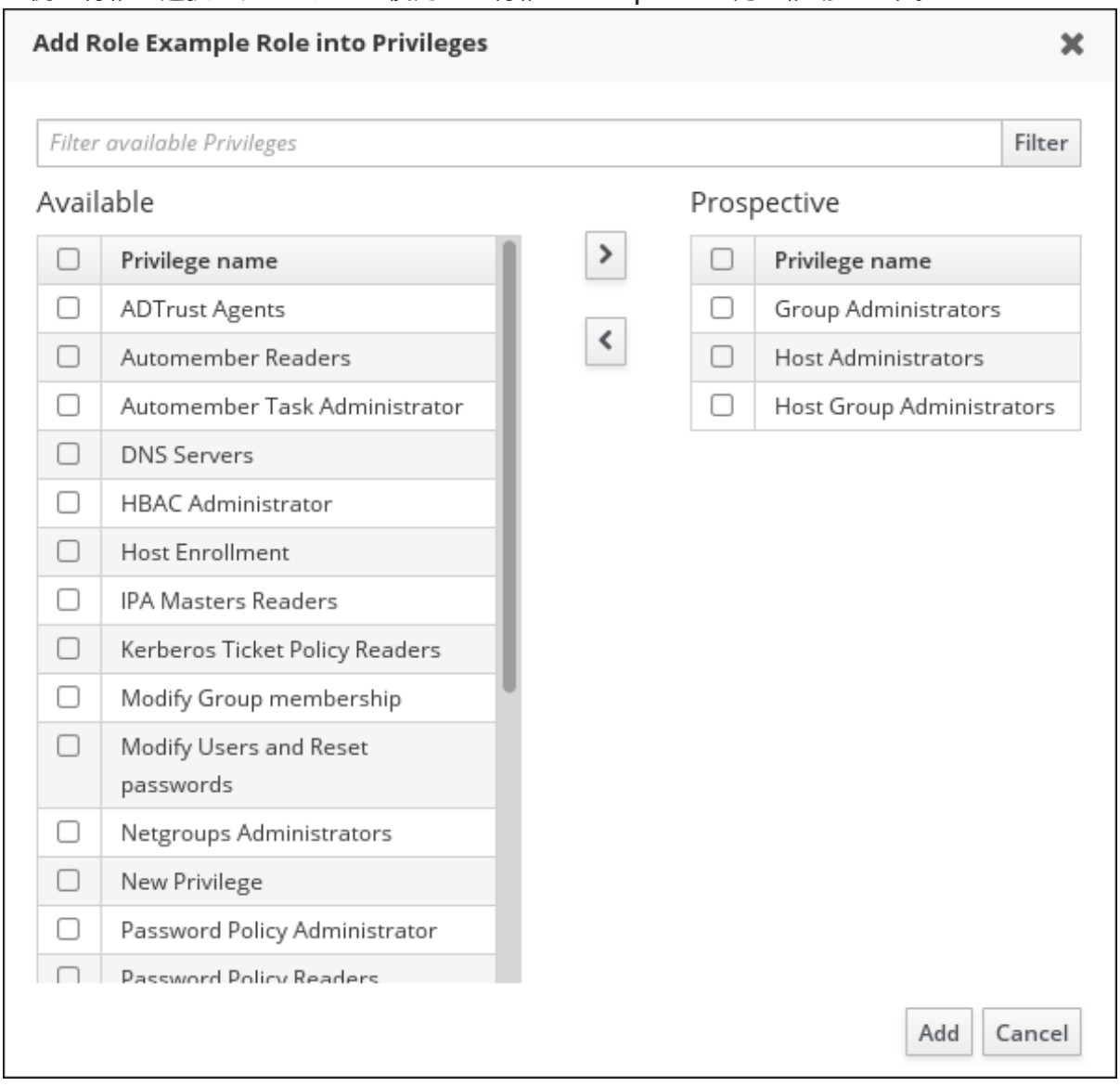
7. 開いているウィンドウで、左側のメンバーを選択し、> ボタンを使用して、メンバーを **Prospective** 列に移動します。



8. **Privileges** タブの上部にある **Add** をクリックします。



9. 左側の特権を選択し、> ボタンを使用して特権を **Prospective** 列に移動します。



10. **Add** ボタンをクリックして保存します。

11. **オプション。**ロールから特権またはメンバーを削除する必要がある場合は、削除するエンティティーの名前の横にあるチェックボックスにチェックマークを入れてから **Delete** ボタンをクリックします。ダイアログが開きます。

12. **オプション。**既存のロールを削除する必要がある場合は、リストで名前の横にあるチェックボックスを編集した後に **Delete** ボタンをクリックすると、**Remove roles** ダイアログが表示します。

第33章 ANSIBLE PLAYBOOK を使用して IDM を管理する環境の準備

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

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

このプラクティスを使用すると、すべての Playbook を 1 カ所で見つけることができます。また、root 権限を呼び出しなくても Playbook を実行できます。

注記

マネージドノードで **root** 権限があれば、**ipaserver**、**ipareplica**、**ipaclient**、および **ipabackup ansible-freeipa** ロールを実行できます。これらのロールには、ディレクトリーおよび **dnf** ソフトウェアパッケージマネージャーへの特権アクセスが必要です。

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

前提条件

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

手順

1. Ansible 設定および Playbook のディレクトリーをホームディレクトリーに作成します。

```
$ mkdir ~/MyPlaybooks/
```

2. **~/MyPlaybooks/** ディレクトリーに移動します。

```
$ cd ~/MyPlaybooks
```

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

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

```
[privilege_escalation]
become=True
```

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

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

[us]
replica.idm.example.com

[ipaserver:children]
eu
us
```

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

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

```
$ ssh-keygen
```

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

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

これらのコマンドでは、IdM **管理者** パスワードを入力します。

関連情報

- Ansible Playbook を使用した Identity Management サーバーのインストール を参照してください。
- インベントリーの構築方法 を参照してください。

第34章 ANSIBLE PLAYBOOK を使用した IDM でのロールベースアクセス制御の管理

ロールベースアクセス制御 (RBAC) は、ロールおよび権限関連を定義する、ポリシーに依存しないアクセス制御メカニズムです。Identity Management (IdM) の RBAC のコンポーネントは、ロール、権限、パーミッションです。

- **パーミッション** は、ユーザーの追加または削除、グループの変更、読み取りアクセスの有効化など、特定のタスクを実行する権限を付与します。
- **特権** は、新規ユーザーの追加に必要な全権限など、権限を組み合わせます。
- **ロール** は、ユーザー、ユーザーグループ、ホスト、またはホストグループに特権のセットを付与します。

特に大企業では、RBAC を使用すると、責任の領域を個別に設定する階層管理システムを作成できます。

本章では、Ansible Playbook を使用した RBAC の管理時に行う以下の操作について説明します。

- IdM のパーミッション
- デフォルトの管理パーミッション
- IdM の特権
- IdM のロール
- IdM の事前定義されたロール
- Ansible を使用して特権のある IdM RBAC ロールを存在させる手順
- Ansible を使用して IdM RBAC ロールを設定しないようにする手順
- Ansible を使用して、ユーザーグループに IdM RBAC ロールを割り当てる手順
- Ansible を使用して特定のユーザーに IdM RBAC ロールが割り当てられないようにする手順
- Ansible を使用してサービスを IdM RBAC ロールに所属させるように設定する手順
- Ansible を使用してホストを IdM RBAC ロールに所属させるように設定する手順
- Ansible を使用してホストグループを IdM RBAC ロールに所属させるように設定する手順

34.1. IDM のパーミッション

パーミッションは、ロールベースのアクセス制御の中で最も低いレベルの単位で、操作を適用する LDAP エントリーと合わせて操作を定義します。ブロックの構築と同様に、パーミッションは必要に応じて多くの権限に割り当てることができます。
1 つ以上の **権限** を使用して、許容される操作を定義します。

- **write**
- **read**
- **search**

- **compare**
- **add**
- **delete**
- **all**

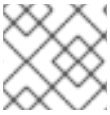
上記の操作は、3 つの基本的な **ターゲット** に適用されます。

- **subtree**: ドメイン名 (DN) (この DN のサブツリー)
- **target filter**: LDAP フィルター
- **target**: DN。ワイルドカードでエントリーを指定可能。

また、以下の便利なオプションは、対応する属性を設定します。

- **type**: オブジェクトのタイプ (ユーザー、グループなど) (**subtree** および **target filter** を設定します)。
- **memberof**: グループのメンバー。**target filter** を設定します。
- **targetgroup**: 特定のグループを変更する権限 (グループメンバーシップの管理権限の付与など) を付与します (**target** を設定します)。

IdM パーミッションを使用すると、どのユーザーがどのオブジェクトにアクセスできるか、さらにこのようなオブジェクトの属性にアクセスできるかどうかを制御できます。IdM を使用すると、個別の属性を許可または拒否したり、ユーザー、グループ、sudo などの特定の IdM 機能を、全匿名ユーザー、全認証済みユーザー、または特定の特権ユーザーグループ限定などと、全体的な表示設定を変更したりできます。
たとえば、このアプローチではパーミッション指定に柔軟性があるので、アクセスが必要な特定のセクションのみにユーザーまたはグループのアクセスを制限し、他のセクションをこれらのユーザーまたはグループには完全に表示されないように設定する場合に、管理者にとって便利です。



注記

パーミッションには他のパーミッションを含めることはできません。

34.2. デフォルトの管理パーミッション

管理パーミッションは、IdM にデフォルトで含まれているパーミッションです。このパーミッションはユーザーが作成した他のパーミッションと同様に機能しますが、以下の相違点があります。

- この管理パーミッションは削除できず、名前、場所、ターゲットの属性を変更できません。
- このパーミッションには 3 つの属性セットがあります。
  - **デフォルト** の属性。IdM で管理されているため、ユーザーは変更できません。
  - **包含** 属性。ユーザーが別途追加する属性。
  - **除外** 属性。ユーザーが削除する属性。

管理パーミッションは、デフォルトおよび包含属性セットに表示されている属性すべてに適用されますが、除外セットに表示されている属性には適用されません。

注記

管理パーミッションを削除できませんが、パーミッションにバインドタイプを設定し、すべての特権から管理パーミッションを削除して管理パーミッションを効果的に無効にできます。

管理パーミッションの名前はすべて **System:** から始まります (例: **System: Add Sudo rule** または **System: Modify Services**)。以前のバージョンの IdM では、デフォルトのパーミッションに異なるスキームを使用していました。たとえば、ユーザーはパーミッションの削除はできず、特権に割り当てるしかできませんでした。これらのデフォルトパーミッションのほとんどは、管理パーミッションに切り替わっていますが、以下のパーミッションは引き続き以前のスキームを使用します。

- Automember Rebuild メンバーシップタスクの追加
- 設定サブエントリーの追加
- レプリカ合意の追加
- 証明書削除保留
- CA から証明書のステータス取得
- DNA 範囲の読み取り
- DNA 範囲の変更
- PassSync Manager の設定の読み取り
- PassSync Manager 設定の変更
- レプリカ合意の読み込み
- レプリカ合意の修正
- レプリカ合意の削除
- LDBM データベース設定の読み取り
- 証明書の要求
- CA ACL を無視する証明書の要求
- 別のホストからの証明書の要求
- CA からの証明書の取得
- 証明書の取り消し
- IPA 設定の書き込み

注記

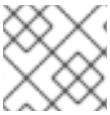
コマンドラインから管理パーミッションを変更しようとし、変更不可な属性の変更をシステム側が許可しない場合には、コマンドに失敗します。Web UI から管理パーミッションを変更しようとした場合には、変更できない属性が無効になります。

34.3. IDM の特権

特権は、ロールに適用されるパーミッションのグループです。
パーミッションは単一の操作を実行する権限を提供しますが、IdM タスクを成功させるには、複数のパーミッションが必要なものがあります。したがって、特権は、特定のタスクを実行するために必要な異なるパーミッションを組み合わせたものです。
たとえば、新しい IdM ユーザーにアカウントを設定するには、以下の権限が必要です。

- 新規ユーザーエントリーの作成
- ユーザーパスワードのリセット
- 新規ユーザーのデフォルト IPA ユーザーグループへの追加

これらの 3 つの低レベルのタスクを、**ユーザーの追加** という名前のカスタム特権の形式で、権限がより高いレベルのタスクに組み合わせることで、システム管理者はロールを管理しやすくなります。IdM には、すでにいくつかのデフォルト特権が含まれています。ユーザーとユーザーグループとは別に、権限はホストおよびホストグループ、およびネットワークサービスにも割り当てられます。これにより、特定のネットワークサービスを使用するホストセットのユーザーセットによって、操作をきめ細かく制御できます。

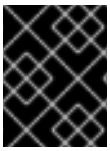


注記

特権には、他の特権を含めることはできません。

34.4. IDM のロール

ロールは、ロールに指定したユーザーが所有する権限のリストです。
実際には、パーミッションでは、指定の低階層のタスク (ユーザーエントリーの作成、グループへのエントリーの追加など) を実行する権限を付与し、特権では、高階層のタスク (指定のグループへの新規ユーザーの作成など) に必要なこれらのパーミッションの 1 つ以上を組み合わせます。ロールは必要に応じて、管理者ロールでユーザーの追加、変更、削除ができるなど、特権をまとめます。



重要

ロールは、許可されたアクションを分類するために使用されます。ロールは、特権昇格されないようにしたり、特権の分離を実装するツールとしては使用しません。

注記

ロールに他のロールを含めることはできません。

34.5. IDENTITY MANAGEMENT で事前定義されたロール

Red Hat Identity Management には、以下の事前定義済みのロールが含まれています。

表34.1 Identity Management の定義済みロール

ロール	特権	説明
登録管理者	ホストの登録	クライアントまたはホストの登録を行います。

ロール	特権	説明
helpdesk	Modify Users、Reset passwords、Modify Group membership	簡単なユーザー管理タスクを実行します。
IT Security Specialist	Netgroups Administrators、HBAC Administrator、Sudo Administrator	ホストベースのアクセス制御、sudo ルールなどのセキュリティーポリシーを管理します。
IT Specialist	Host Administrators、Host Group Administrators、Service Administrators、Automount Administrators	ホストの管理を行います
Security Architect	Delegation Administrator、Replication Administrators、Write IPA Configuration、Password Policy Administrator	Identity Management 環境の管理、信頼の作成、レプリカ合意を作成します。
User Administrator	User Administrators、Group Administrators、Stage User Administrators	ユーザーおよびグループの作成を行います

34.6. ANSIBLE を使用して特権のある IDM RBAC ロールを存在させる手順

デフォルトのロール以外で、ロールベースのアクセス制御 (RBAC) を使用して Identity Management (IdM) のリソースを詳細にわたり制御するには、カスタムロールを作成します。

以下の手順では、Ansible Playbook を使用して、新しい IdM カスタムロールの特権を定義し、その存在を確認する方法を説明します。この例では、新しい **user_and_host_administrator** ロールには、デフォルトで IdM で存在する以下の特権を一意に組み合わせたものが含まれます。

- **Group Administrators**
- **User Administrators**
- **Stage User Administrators**
- **Group Administrators**

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。

- ~/MyPlaybooks/ ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
- この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。

- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。

手順

1. **~/<MyPlaybooks>/** ディレクトリーに移動します。

```
$ cd ~/<MyPlaybooks>/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/role/** にある **role-member-user-present.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/role/role-member-user-present.yml role-member-user-present-copy.yml
```

3. Ansible Playbook ファイル (**role-member-user-present-copy.yml**) を開きます。
4. **iparole** タスクセクションに以下の変数を設定して、ファイルを調整します。
   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は新規ロールの名前に設定します。
   - **特権** リストは、新しいロールに追加する IdM 権限の名前に設定します。
   - 必要に応じて、**user** 変数は、新規ロールを付与するユーザーの名前に設定します。
   - 必要に応じて、**group** 変数は、新規ロールを付与するグループの名前に設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Playbook to manage IPA role with members.
  hosts: ipaserver
  become: true
  gather_facts: no

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - iparole:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: user_and_host_administrator
      user: idm_user01
      group: idm_group01
      privilege:
      - Group Administrators
      - User Administrators
      - Stage User Administrators
      - Group Administrators
```

-

5. ファイルを保存します。

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

```
$ ansible-playbook --vault-password-file=password_file -v -i
~/<MyPlaybooks>/inventory role-member-user-present-copy.yml
```

関連情報

- Ansible Vault を使用したコンテンツの暗号化 を参照してください。
- IdM のロール を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-role** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/iparole** ディレクトリーのサンプルの Playbook を参照してください。

34.7. ANSIBLE を使用して IDM RBAC ロールを設定しないようにする手順

Identity Management (IdM) のロールベースアクセス制御 (RBAC) を管理するシステム管理者は、誤って管理者がユーザーに割り当てることがないように、使用しなくなったロールが削除されていることを確認する必要があります。

以下の手順では、Ansible Playbook を使用してロールが削除されていることを確認する方法を説明します。以下の例では、カスタムの **user_and_host_administrator** ロールが IdM に存在しないことを確認する方法を説明します。

前提条件

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

手順

1. **~/<MyPlaybooks>/** ディレクトリーに移動します。

```
$ cd ~/<MyPlaybooks>/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/role/** にある **role-is-absent.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/role/role-is-absent.yml role-is-absent-copy.yml
```

3. Ansible Playbook ファイル (**role-is-absent-copy.yml**) を開きます。

4. **iparole** タスクセクションに以下の変数を設定して、ファイルを調整します。

   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は、ロールの名前に設定します。
   - **state** 変数は、**absent** に設定されていることを確認します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Playbook to manage IPA role with members.
  hosts: ipaserver
  become: true
  gather_facts: no

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - iparole:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: user_and_host_administrator
      state: absent
```

5. ファイルを保存します。

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

```
$ ansible-playbook --vault-password-file=password_file -v -i ~/<MyPlaybooks>/inventory role-is-absent-copy.yml
```

関連情報

- Ansible Vault を使用したコンテンツの暗号化 を参照してください。
- IdM のロール を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-role** Markdown ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/iparole** ディレクトリーのサンプルの Playbook を参照してください。

34.8. ANSIBLE を使用して、ユーザーグループに IDM RBAC ロールを割り当てる手順

Identity Management (IdM) のロールベースアクセス制御 (RBAC) を管理するシステム管理者は、junior administrators など、特定のユーザーグループにロールを割り当てます。

以下の例では、Ansible Playbook を使用して、同梱の IdM RBAC **helpdesk** ロールを **junior_sysadmins** に割り当てる方法を説明します。

前提条件

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

手順

1. **~/<MyPlaybooks>/** ディレクトリーに移動します。

```
$ cd ~/<MyPlaybooks>/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/role/** にある **role-member-group-present.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/role/role-member-group-present.yml
role-member-group-present-copy.yml
```

3. Ansible Playbook ファイル (**role-member-group-present-copy.yml**) を開きます。
4. **iparole** タスクセクションに以下の変数を設定して、ファイルを調整します。
   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は、割り当てるロールの名前に設定します。
   - **group** 変数はグループ名に設定します。
   - **action** 変数は **member** に設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Playbook to manage IPA role with members.
  hosts: ipaserver
  become: true
  gather_facts: no
```

```
vars_files:
- /home/user_name/MyPlaybooks/secret.yml
tasks:
- iparole:
    ipaadmin_password: "{{ ipaadmin_password }}"
    name: helpdesk
    group: junior_sysadmins
    action: member
```

5. ファイルを保存します。

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

```
$ ansible-playbook --vault-password-file=password_file -v -i
~/<MyPlaybooks>/inventory role-member-group-present-copy.yml
```

関連情報

- Ansible Vault を使用したコンテンツの暗号化 を参照してください。
- IdM のロール を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-role** Markdown ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/iparole** ディレクトリーのサンプルの Playbook を参照してください。

34.9. ANSIBLE を使用して特定のユーザーに IDM RBAC ロールが割り当てられないようにする手順

Identity Management (IdM) のロールベースアクセス制御 (RBAC) を管理するシステム管理者は、会社内で別の役職に異動した後など、特定のユーザーに RBAC ロールが割り当てられないようにすることもできます。

以下の手順では、Ansible Playbook を使用して、 **user_01** および **user_02** という名前のユーザーが **helpdesk** ロールに割り当てられないようにする方法を説明します。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
  - この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。

- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。

手順

1. **~/<MyPlaybooks>/** ディレクトリーに移動します。

```
$ cd ~/<MyPlaybooks>/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/role/** にある **role-member-user-absent.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/role/role-member-user-absent.yml role-member-user-absent-copy.yml
```

3. Ansible Playbook ファイル (**role-member-user-absent-copy.yml**) を開きます。

4. **iparole** タスクセクションに以下の変数を設定して、ファイルを調整します。

   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は、割り当てるロールの名前に設定します。
   - **user** リストをユーザーの名前に設定します。
   - **action** 変数は **member** に設定します。
   - **state** 変数は **absent** に設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Playbook to manage IPA role with members.
  hosts: ipaserver
  become: true
  gather_facts: no

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - iparole:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: helpdesk
      user
      - user_01
      - user_02
      action: member
      state: absent
```

5. ファイルを保存します。

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

```
$ ansible-playbook --vault-password-file=password_file -v -i
~/<MyPlaybooks>/inventory role-member-user-absent-copy.yml
```

関連情報

- Ansible Vault を使用したコンテンツの暗号化 を参照してください。
- IdM のロール を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-role** Markdown ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/iparole** ディレクトリーのサンプルの Playbook を参照してください。

34.10. ANSIBLE を使用してサービスを IDM RBAC ロールに所属させるように設定する手順

Identity Management (IdM) のロールベースアクセス制御 (RBAC) を管理するシステム管理者は、IdM に登録されている特定のサービスが、特定のロールのメンバーになっていることを確認する必要がある場合があります。以下の例では、カスタムの **web_administrator** ロールを使用して **client01.idm.example.com** サーバー上で実行中の **HTTP** サービスを管理できるようにする方法を説明します。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
  - この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。
- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。
- **web_administrator** ロールが IdM に存在する。
- **HTTP/client01.idm.example.com@IDM.EXAMPLE.COM** サービスが IdM に存在する。

手順

1. **~/<MyPlaybooks>/** ディレクトリーに移動します。

```
$ cd ~/<MyPlaybooks>/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/role/** にある **role-member-service-present.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/role/role-member-service-present-absent.yml role-member-service-present-copy.yml
```

3. Ansible Playbook ファイル (**role-member-service-present-copy.yml**) を開きます。

4. **iparole** タスクセクションに以下の変数を設定して、ファイルを調整します。

   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は、割り当てるロールの名前に設定します。
   - **service** リストはサービス名に設定します。
   - **action** 変数は **member** に設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Playbook to manage IPA role with members.
  hosts: ipaserver
  become: true
  gather_facts: no

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - iparole:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: web_administrator
      service:
      - HTTP/client01.idm.example.com
      action: member
```

5. ファイルを保存します。

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

```
$ ansible-playbook --vault-password-file=password_file -v -i ~/<MyPlaybooks>/inventory role-member-service-present-copy.yml
```

関連情報

- Ansible Vault を使用したコンテンツの暗号化 を参照してください。
- IdM のロール を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-role** Markdown ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/iparole** ディレクトリーのサンプルの Playbook を参照してください。

34.11. ANSIBLE を使用してホストを IDM RBAC ロールに所属させるように設定する手順

Identity Management (IdM) でロールベースアクセス制御を管理するシステム管理者は、特定のホストまたはホストグループが特定のロールに関連付けられていることを確認する必要がある場合があります。以下の例では、カスタムの **web_administrator** ロールが **HTTP** サービスを実行している **client01.idm.example.com** の IdM ホストを管理できるようにする方法を説明します。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
  - この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。
- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。
- **web_administrator** ロールが IdM に存在する。
- **client01.idm.example.com** ホストが IdM に存在する。

手順

1. **~/<MyPlaybooks>/** ディレクトリーに移動します。

```
$ cd ~/<MyPlaybooks>/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/role/** にある **role-member-host-present.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/role/role-member-host-present.yml role-member-host-present-copy.yml
```

3. Ansible Playbook ファイル (**role-member-host-present-copy.yml**) を開きます。
4. **iparole** タスクセクションに以下の変数を設定して、ファイルを調整します。
   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は、割り当てるロールの名前に設定します。
   - **host** リストをホストの名前に設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Playbook to manage IPA role with members.
  hosts: ipaserver
  become: true
  gather_facts: no

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - iparole:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: web_administrator
      host:
      - client01.idm.example.com
      action: member
```

5. ファイルを保存します。

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

```
$ ansible-playbook --vault-password-file=password_file -v -i
~/<MyPlaybooks>/inventory role-member-host-present-copy.yml
```

関連情報

- Ansible Vault を使用したコンテンツの暗号化 を参照してください。
- IdM のロール を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-role** Markdown ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/iparole** ディレクトリーのサンプルの Playbook を参照してください。

34.12. ANSIBLE を使用してホストグループを IDM RBAC ロールに所属させるように設定する手順

Identity Management (IdM) でロールベースアクセス制御を管理するシステム管理者は、特定のホストまたはホストグループが特定のロールに関連付けられていることを確認する必要がある場合があります。以下の例では、カスタムの **web_administrator** ロールが **HTTP** サービスを実行している IdM ホストの **web_servers** グループを管理できるようにする方法を説明します。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。

- ~/MyPlaybooks/ ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
- この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。

- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。
- **web_administrator** ロールが IdM に存在する。
- **web_servers** ホストグループが IdM に存在する。

手順

1. **~/<MyPlaybooks>/** ディレクトリーに移動します。

```
$ cd ~/<MyPlaybooks>/
```

2. **/usr/share/doc/ansible-freeipa/playbooks/role/** にある **role-member-hostgroup-present.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/role/role-member-hostgroup-present.yml role-member-hostgroup-present-copy.yml
```

3. Ansible Playbook ファイル (**role-member-hostgroup-present-copy.yml**) を開きます。
4. **iparole** タスクセクションに以下の変数を設定して、ファイルを調整します。
   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は、割り当てるロールの名前に設定します。
   - **hostgroup** リストはホストグループ名に設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Playbook to manage IPA role with members.
  hosts: ipaserver
  become: true
  gather_facts: no

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - iparole:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: web_administrator
      hostgroup:
      - web_servers
      action: member
```

5. ファイルを保存します。

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

```
$ ansible-playbook --vault-password-file=password_file -v -i
~/<MyPlaybooks>/inventory role-member-hostgroup-present-copy.yml
```

関連情報

- Ansible Vault を使用したコンテンツの暗号化 を参照してください。
- IdM のロール を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーの **README-role** Markdown ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/iparole** ディレクトリーのサンプルの Playbook を参照してください。

第35章 ANSIBLE PLAYBOOK を使用した RBAC 権限の管理

ロールベースアクセス制御 (RBAC) は、ロール、権限およびパーミッションで定義する、ポリシーに依存しないアクセス制御メカニズムです。特に大企業では、RBAC を使用すると、責任の領域を個別に設定する階層管理システムを作成できます。

本章では、Ansible Playbook を使用して Identity Management (IdM) で RBAC 権限を管理する以下の操作について説明します。

- Ansible を使用してカスタムの RBAC 特権を存在させる手順
- Ansible を使用してカスタムの IdM RBAC 特権にメンバーパーミッションを存在させる手順
- Ansible を使用して IdM RBAC 特権にパーミッションが含まれないようにする手順
- Ansible を使用してカスタムの IdM RBAC 権限の名前を変更する手順
- Ansible を使用して IdM RBAC 特権が含まれないようにする手順

前提条件

- RBAC の概念と原則 を理解している。

35.1. ANSIBLE を使用してカスタムの IDM RBAC 特権を存在させる手順

Identity Management (IdM) のロールベースアクセス制御 (RBAC) でカスタム権限を完全に機能させるには、ステージごとに進めていく必要があります。

1. パーミッションが割り当てられていない特権を作成します。
2. 選択したパーミッションを特権に追加します。

以下の手順では、後でパーミッションを追加できるように、Ansible Playbook を使用して空の特権を作成する方法を説明します。この例では、ホスト管理に関連するすべての IdM パーミッションを組み合わせられるように **full_host_administration** という名前の特権を作成する方法を説明します。

前提条件

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

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

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

2. **/usr/share/doc/ansible-freeipa/playbooks/privilege/** にある **privilege-present.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/privilege/privilege-present.yml privilege-present-copy.yml
```

3. Ansible Playbook ファイル (**privilege-present-copy.yml**) を開きます。

4. **ipaprivilege** タスクセクションに以下の変数を設定してファイルを調整します。

   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は、新しい特権 **full_host_administration** の名前に設定します。
   - 必要に応じて、**description** 変数を使用して特権を記述します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Privilege present example
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure privilege full_host_administration is present
    ipaprivilege:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: full_host_administration
      description: This privilege combines all IdM permissions related to host administration
```

5. ファイルを保存します。

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

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory privilege-present-copy.yml
```

35.2. ANSIBLE を使用してカスタムの IDM RBAC 特権にメンバーパーミッションを存在させる手順

Identity Management (IdM) のロールベースアクセス制御 (RBAC) でカスタム権限を完全に機能させるには、ステージごとに進めていく必要があります。

1. パーミッションが割り当てられていない特権を作成します。

2. 選択したパーミッションを特権に追加します。

以下の手順では、Ansible Playbook を使用して、前の手順で作成した特権にパーミッションを追加する方法を説明します。この例では、ホスト管理に関連する IdM パーミッションをすべて、**full_host_administration** という名前の特権に追加する方法を説明します。デフォルトでは、パーミッションは **Host Enrollment**、**Host Administrators** および **Host Group Administrator** 特権の間で分散されます。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
  - この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。
- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。
- **full_host_administration** 権限が存在する。Ansible を使用して特権を作成する方法の詳細は、Ansible を使用してカスタムの IdM RBAC 特権を存在させる手順 を参照してください。

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

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

2. **/usr/share/doc/ansible-freeipa/playbooks/privilege/** にある **privilege-member-present.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/privilege/privilege-member-present.yml
privilege-member-present-copy.yml
```

3. Ansible Playbook ファイル (**privilege-member-present-copy.yml**) を開きます。
4. **ipaprivilege** タスクセクションに以下の変数を設定してファイルを調整します。
   - 使用しているユースケースに合わせて、タスクの **名前** を調節します。
   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は、特権の名前に設定します。
   - **permission** は、特権に追加するパーミッションの名前を設定します。
   - **action** 変数が **member** に設定されていることを確認します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Privilege member present example
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure that permissions are present for the "full_host_administration" privilege
    ipaprivilege:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: full_host_administration
      permission:
      - "System: Add krbPrincipalName to a Host"
      - "System: Enroll a Host"
      - "System: Manage Host Certificates"
      - "System: Manage Host Enrollment Password"
      - "System: Manage Host Keytab"
      - "System: Manage Host Principals"
      - "Retrieve Certificates from the CA"
      - "Revoke Certificate"
      - "System: Add Hosts"
      - "System: Add krbPrincipalName to a Host"
      - "System: Enroll a Host"
      - "System: Manage Host Certificates"
      - "System: Manage Host Enrollment Password"
      - "System: Manage Host Keytab"
      - "System: Manage Host Keytab Permissions"
      - "System: Manage Host Principals"
      - "System: Manage Host SSH Public Keys"
      - "System: Manage Service Keytab"
      - "System: Manage Service Keytab Permissions"
      - "System: Modify Hosts"
      - "System: Remove Hosts"
      - "System: Add Hostgroups"
      - "System: Modify Hostgroup Membership"
      - "System: Modify Hostgroups"
      - "System: Remove Hostgroups"
```

5. ファイルを保存します。

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

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory privilege-member-present-copy.yml
```

35.3. ANSIBLE を使用して IDM RBAC 特権にパーミッションが含まれないようにする手順

Identity Management (IdM) のシステム管理者は、IdM のロールベースアクセス制御をカスタマイズできます。

以下の手順では、Ansible Playbook を使用して、特権からパーミッションを削除する方法を説明します。この例では、管理者がセキュリティー上のリスクを考慮するため、デフォルトの **Certificate Administrators** 特権から **Request Certificates ignoring CA ACLs** を削除する方法を説明します。

前提条件

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

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

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

2. **/usr/share/doc/ansible-freeipa/playbooks/privilege/** にある **privilege-member-present.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/privilege/privilege-member-absent.yml
privilege-member-absent-copy.yml
```

3. Ansible Playbook ファイル (**privilege-member-absent-copy.yml**) を開きます。
4. **ipaprivilege** タスクセクションに以下の変数を設定してファイルを調整します。
   - 使用しているユースケースに合わせて、タスクの **名前** を調節します。
   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は、特権の名前に設定します。
   - **permission** のリストは、特権から削除するパーミッションの名前に設定します。
   - **action** 変数が **member** に設定されていることを確認します。
   - **state** 変数は **absent** に設定されていることを確認します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Privilege absent example
  hosts: ipaserver
```

```
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure that the "Request Certificate ignoring CA ACLs" permission is absent from
the "Certificate Administrators" privilege
    ipaprivilege:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: Certificate Administrators
      permission:
      - "Request Certificate ignoring CA ACLs"
      action: member
      state: absent
```

5. ファイルを保存します。

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

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory privilege-member-absent-copy.yml
```

35.4. ANSIBLE を使用してカスタムの IDM RBAC 権限の名前を変更する手順

Identity Management (IdM) のシステム管理者は、IdM のロールベースアクセス制御をカスタマイズできます。

以下の手順では、たとえば、パーミッションの一部を削除したなどの理由から、特権の名前を変更する方法を説明します。パーミッションを削除した結果、特権の名前は正確ではなくなりました。この例では、管理者は **full_host_administration** 特権の名前を **limited_host_administration** に変更します。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
  - この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。
- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。
- **full_host_administration** 権限が存在する。特権の追加方法は、Ansible を使用してカスタムの IdM RBAC 特権を存在させる手順 を参照してください。

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

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

2. **/usr/share/doc/ansible-freeipa/playbooks/privilege/** にある **privilege-present.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/privilege/privilege-present.yml rename-privilege.yml
```

3. Ansible Playbook ファイル (**rename-privilege.yml**) を開いて編集します。

4. **ipaprivilege** タスクセクションに以下の変数を設定してファイルを調整します。

- **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
- **name** 変数は、現在の特権名に設定します。
- **rename** 変数を追加して、特権の新しい名前に設定します。
- **state** 変数を追加し、 **renamed** に設定します。

5. 以下のように、Playbook 自体の名前を変更します。

```
---
- name: Rename a privilege
  hosts: ipaserver
```

6. 以下のように、Playbook のタスクの名前を変更します。

```
[...]
tasks:
- name: Ensure the full_host_administration privilege is renamed to limited_host_administration
  ipaprivilege:
  [...]
```

以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Rename a privilege
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure the full_host_administration privilege is renamed to limited_host_administration
    ipaprivilege:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: full_host_administration
      rename: limited_host_administration
      state: renamed
```

7. ファイルを保存します。

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

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory rename-privilege.yml
```

35.5. ANSIBLE を使用して IDM RBAC 特権が含まれないようにする手順

Identity Management (IdM) のシステム管理者は、IdM のロールベースアクセス制御をカスタマイズできます。以下の手順では、Ansible Playbook を使用して RBAC 特権が削除されていることを確認する方法を説明します。この例では、**CA administrator** 特権が存在しないことを確認する方法を説明します。この手順が終わると、IdM で認証局を管理できるユーザーは **admin** だけになります。

前提条件

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

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

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

2. **/usr/share/doc/ansible-freeipa/playbooks/privilege/** ディレクトリーにある **privilege-absent.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/privilege/privilege-absent.yml privilege-absent-copy.yml
```

3. Ansible Playbook ファイル (**privilege-absent-copy.yml**) を開きます。
4. **ipaprivilege** タスクセクションに以下の変数を設定してファイルを調整します。
   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数は、削除する権限の名前に設定します。
   - **state** 変数が **absent** に設定されていることを確認します。
5. 以下のように、Playbook のタスクの名前を変更します。

```
[...]
tasks:
- name: Ensure privilege "CA administrator" is absent
  ipaprivilege:
  [...]
```

以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Privilege absent example
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure privilege "CA administrator" is absent
    ipaprivilege:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: CA administrator
      state: absent
```

6. ファイルを保存します。

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

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory privilege-absent-copy.yml
```

35.6. 関連情報

- IdM の特権 を参照してください。
- IdM のパーミッション を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーで利用可能な **README-privilege** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/ipaprivilege** ディレクトリーのサンプルの Playbook を参照してください。

第36章 ANSIBLE PLAYBOOK を使用した IDM での RBAC パーミッションの管理

ロールベースアクセス制御 (RBAC) は、ロール、権限およびパーミッションで定義する、ポリシーに依存しないアクセス制御メカニズムです。特に大企業では、RBAC を使用すると、責任の領域を個別に設定する階層管理システムを作成できます。

本章では、Ansible Playbook を使用して Identity Management (IdM) で RBAC パーミッションの管理時に行う、以下の操作について説明します。

- Ansible を使用して RBAC パーミッションを存在させる手順
- Ansible を使用して属性を含めて RBAC パーミッションを追加する手順
- Ansible を使用して RBAC パーミッションをバインドする手順
- Ansible を使用して属性を IdM RBAC パーミッションをメンバーにする手順
- Ansible を使用して属性が IdM RBAC パーミッションのメンバーに含まれないようにする手順
- Ansible を使用して IdM RBAC パーミッションの名前を変更する手順

前提条件

- RBAC の概念と原則 を理解している。

36.1. ANSIBLE を使用して RBAC パーミッションを存在させる手順

Identity Management (IdM) のシステム管理者は、IdM のロールベースアクセス制御 (RBAC) をカスタマイズできます。

以下の手順では、Ansible Playbook を使用して、パーミッションを特権に追加できるように IdM にパーミッションを追加する方法を説明します。この例では、目的とする以下の状態を達成する方法を説明します。

- **MyPermission** パーミッションが存在する。
- **MyPermission** パーミッションだけがホストに適用できる。
- パーミッションを含む特権を付与されたユーザーは、エントリーに対して以下の操作すべてを実行できます。
  - Write
  - Read
  - Search
  - Compare
  - Add
  - Delete

前提条件

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

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

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

2. **/usr/share/doc/ansible-freeipa/playbooks/permission/** ディレクトリーにある **permission-present.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/permission/permission-present.yml
permission-present-copy.yml
```

3. Ansible Playbook ファイル (**permission-present-copy.yml**) を開きます。
4. **ipapermission** タスクセクションに以下の変数を設定して、ファイルを調整します。
   - 使用しているユースケースに合わせて、タスクの **名前** を調節します。
   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数はパーミッションの名前に設定します。
   - **object_type** 変数は **host** に設定します。
   - **right** 変数は **all** に設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Permission present example
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure that the "MyPermission" permission is present
    ipapermission:
      ipaadmin_password: "{{ ipaadmin_password }}"
```

```
name: MyPermission
object_type: host
right: all
```

5. ファイルを保存します。

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

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory permission-present-copy.yml
```

36.2. ANSIBLE を使用して属性を含めて RBAC パーミッションを追加する手順

Identity Management (IdM) のシステム管理者は、IdM のロールベースアクセス制御 (RBAC) をカスタマイズできます。

以下の手順では、Ansible Playbook を使用して、パーミッションを特権に追加できるように IdM にパーミッションを追加する方法を説明します。この例では、目的とする以下の状態を達成する方法を説明します。

- **MyPermission** パーミッションが存在する。
- **MyPermission** パーミッションだけがホストの追加に使用できる。
- パーミッションを含む特権を付与されたユーザーは、ホストのエントリーに対して以下の操作すべてを実行できる。
  - Write
  - Read
  - Search
  - Compare
  - Add
  - Delete
- **MyPermission** パーミッションを含む特権のあるユーザーが作成したホストエントリーに **description** の値を追加できる。

注記

IdM LDAP スキーマでは、パーミッションの作成または変更時に指定できる属性のタイプは制約されません。ただし、**object_type** が **host** の場合に **attrs: car_licence** を指定すると、後でパーミッションを実行して、車のライセンス値をホストに追加使用とすると **ipa: ERROR: attribute "car-license" not allowed** というエラーメッセージが表示されます。

前提条件

- IdM 管理者パスワードを把握している。

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
  - この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。
- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

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

2. **/usr/share/doc/ansible-freeipa/playbooks/permission/** ディレクトリーにある **permission-present.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/permission/permission-present.yml
permission-present-with-attribute.yml
```

3. Ansible Playbook ファイル (**permission-present-with-attribute.yml**) を開きます。
4. **ipapermission** タスクセクションに以下の変数を設定して、ファイルを調整します。
   - 使用しているユースケースに合わせて、タスクの **名前** を調節します。
   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数はパーミッションの名前に設定します。
   - **object_type** 変数は **host** に設定します。
   - **right** 変数は **all** に設定します。
   - **attrs** 変数は **description** に設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Permission present example
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure that the "MyPermission" permission is present with an attribute
    ipapermission:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: MyPermission
```

```
    object_type: host
    right: all
    attrs: description
```

5. ファイルを保存します。

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

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory permission-present-with-attribute.yml
```

関連情報

- RHEL 7 の **Linux Domain Identity, Authentication and Policy Guide**の User and group schema を参照してください。

36.3. ANSIBLE を使用して RBAC パーミッションをバインドする手順

Identity Management (IdM) のシステム管理者は、IdM のロールベースアクセス制御 (RBAC) をカスタマイズできます。

以下の手順では、Ansible Playbook を使用して、パーミッションを特権に追加できないように IdM からパーミッションを削除する方法を説明します。

前提条件

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

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

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

2. **/usr/share/doc/ansible-freeipa/playbooks/permission/** ディレクトリーにある **permission-absent.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/permission/permission-absent.yml
permission-absent-copy.yml
```

3. Ansible Playbook ファイル (**permission-absent-copy.yml**) を開きます。
4. **ipapermission** タスクセクションに以下の変数を設定して、ファイルを調整します。
   - 使用しているユースケースに合わせて、タスクの **名前** を調節します。
   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数はパーミッションの名前に設定します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Permission absent example
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure that the "MyPermission" permission is absent
    ipapermission:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: MyPermission
      state: absent
```

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

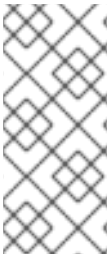
```
$ ansible-playbook --vault-password-file=password_file -v -i inventory permission-
absent-copy.yml
```

36.4. ANSIBLE を使用して属性を IDM RBAC パーミッションをメンバーにする手順

Identity Management (IdM) のシステム管理者は、IdM のロールベースアクセス制御 (RBAC) をカスタマイズできます。

以下の手順では、Ansible Playbook を使用して、属性が IdM の RBAC パーミッションのメンバーであることを確認する方法を説明します。この手順を行うと、パーミッションのあるユーザーは、属性のあるエントリーを作成できます。

この例では、**MyPermission** パーミッションを含む特権を持つユーザーにより作成されたホストエントリーに、**gecos** および **description** の値を設定する方法を説明します。



注記

IdM LDAP スキーマでは、パーミッションの作成または変更時に指定できる属性のタイプは制約されません。ただし、**object_type** が **host** の場合に **attrs: car_licence** を指定すると、後でパーミッションを実行して、車のライセンス値をホストに追加使用とすると **ipa: ERROR: attribute "car-license" not allowed** というエラーメッセージが表示されます。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
  - この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。
- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。
- **MyPermission** パーミッションが存在する。

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

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

2. **/usr/share/doc/ansible-freeipa/playbooks/permission/** ディレクトリーにある **permission-member-present.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/permission/permission-member-present.yml permission-member-present-copy.yml
```

3. Ansible Playbook ファイル (**permission-member-present-copy.yml**) を開きます。
4. **ipapermission** タスクセクションに以下の変数を設定して、ファイルを調整します。
   - 使用しているユースケースに合わせて、タスクの **名前** を調節します。
   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数はパーミッションの名前に設定します。
   - **attrs** リストを **description** および **gecos** 変数に設定します。
   - **action** 変数が **member** に設定されていることを確認します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Permission member present example
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure that the "gecos" and "description" attributes are present in
"MyPermission"
    ipapermission:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: MyPermission
      attrs:
      - description
      - gecos
      action: member
```

5. ファイルを保存します。

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

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory permission-
member-present-copy.yml
```

36.5. ANSIBLE を使用して属性が IDM RBAC パーミッションのメンバーに含まれないようにする手順

Identity Management (IdM) のシステム管理者は、IdM のロールベースアクセス制御 (RBAC) をカスタマイズできます。

以下の手順では、Ansible Playbook を使用して、属性が IdM の RBAC パーミッションに含まれないようにする方法を説明します。この手順を実行すると、パーミッションのあるユーザーが IdM LDAP にエントリーを作成すると、そのエントリーで、値に属性を関連付けて設定できません。

この例では、目的とする以下の状態を達成する方法を説明します。

- **MyPermission** パーミッションが存在する。
- **MyPermission** パーミッションを含む特権のあるユーザーが作成したホストエントリーに **description** 属性を使用できない。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。

- この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。

- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。
- MyPermission パーミッションが存在する。

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

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

2. **/usr/share/doc/ansible-freeipa/playbooks/permission/** ディレクトリーにある **permission-member-absent.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/permission/permission-member-absent.yml permission-member-absent-copy.yml
```

3. Ansible Playbook ファイル (**permission-member-absent-copy.yml**) を開きます。
4. **ipapermission** タスクセクションに以下の変数を設定して、ファイルを調整します。
   - 使用しているユースケースに合わせて、タスクの **名前** を調節します。
   - **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
   - **name** 変数はパーミッションの名前に設定します。
   - **attrs** 変数は **description** に設定します。
   - **action** 変数は **member** に設定します。
   - **state** 変数が **absent** に設定されていることを確認します。

   以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Permission absent example
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure that an attribute is not a member of "MyPermission"
    ipapermission:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: MyPermission
      attrs: description
      action: member
      state: absent
```

5. ファイルを保存します。

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

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory permission-member-absent-copy.yml
```

36.6. ANSIBLE を使用して IDM RBAC パーミッションの名前を変更する手順

Identity Management (IdM) のシステム管理者は、IdM のロールベースアクセス制御をカスタマイズできます。

以下の手順では、Ansible Playbook を使用してパーミッションの名前を変更する方法を説明します。この例では、**MyPermission** の名前を **MyNewPermission** に変更する方法を説明します。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
  - この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。
- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。
- **MyPermission** が IdM に存在する。
- **MyNewPermission** が IdM に存在しない。

手順

1. **~/ MyPlaybooks/** ディレクトリーに移動します。

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

2. **/usr/share/doc/ansible-freeipa/playbooks/permission/** ディレクトリーにある **permission-renamed.yml** ファイルのコピーを作成します。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/permission/permission-renamed.yml permission-renamed-copy.yml
```

3. Ansible Playbook ファイル (**permission-renamed-copy.yml**) を開きます。
4. **ipapermission** タスクセクションに以下の変数を設定して、ファイルを調整します。
   - 使用しているユースケースに合わせて、タスクの **名前** を調節します。

- **ipaadmin_password** 変数は IdM 管理者のパスワードに設定します。
- **name** 変数はパーミッションの名前に設定します。

以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Permission present example
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Rename the "MyPermission" permission
    ipapermission:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: MyPermission
      rename: MyNewPermission
      state: renamed
```

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

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory permission-renamed-copy.yml
```

36.7. 関連情報

- IdM のパーミッション を参照してください。
- IdM の特権 を参照してください。
- **/usr/share/doc/ansible-freeipa/** ディレクトリーで利用可能な **README-permission** ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/ipapermission** ディレクトリーのサンプルの Playbook を参照してください。

第37章 ID ビューを使用した IDM クライアントのユーザー属性値のオーバーライド

IdM LDAP サーバーに保存されているユーザー属性またはグループ属性 (ログイン名、ホームディレクトリー、認証に使用する証明書、**SSH** 鍵など) を Identity Management (IdM) ユーザーがオーバーライドする場合、IdM 管理者は、IdM ID ビューを使用して特定の IdM クライアントの値を再定義できます。たとえば、IdM へのログインに最も一般的に使用される IdM クライアントのユーザーに、別のホームディレクトリーを指定できます。

本章では、クライアントとして IdM に登録されたホストで IdM ユーザーに関連付けられた POSIX 属性値を再定義する方法を説明します。

37.1. ID ビュー

Identity Management (IdM) の ID ビューは、以下の情報を指定する IdM クライアント側のビューです。

- 一元定義された POSIX ユーザーまたはグループ属性の新しい値
- 新しい値が適用されるクライアントホスト。

ID ビューには、1 つ以上の上書きが含まれます。オーバーライドは、一元管理された POSIX 属性値の特定の代替です。

IdM サーバーでは、IdM クライアントに ID ビューのみを定義できます。IdM クライアントにクライアント側の上書きをローカルで設定することはできません。

たとえば、ID ビューを使用して、以下の目的を達成できます。

- 環境ごとに異なる属性値を定義する。たとえば、IdM 管理者または別の IdM ユーザーに、IdM クライアントごとに異なるホームディレクトリーを指定できます。ある IdM クライアントのユーザーのホームディレクトリーとして **/home/encrypted/username** を、別のクライアントでは **/dropbox/username** を設定できます。代わりに、この状況で ID ビューを使用すると便利です。たとえば、クライアントの **/etc/sssd/sssd.conf** ファイルにある **fallback_homedir**、**override_homedir** または他のホームディレクトリー変数を変更すると、すべてのユーザーに影響します。手順例は、IdM クライアントで IdM ユーザーのホームディレクトリーをオーバーライドする ID ビューの追加を参照してください。
- 以前に生成された属性値は、ユーザーの UID の上書きなど、別の値に置き換えます。この機能は、たとえば、IdM ユーザーの UID を 1009 にするなど、通常は LDAP で行うのが困難なシステム全体の変更を実現する場合に便利です。IdM ユーザー UID の生成に使用される IdM ID 範囲は、1000 や 10000 程度の小さい値からは開始されません。IdM ユーザーがすべての IdM クライアントで UID 1009 のローカルユーザーの権限を借用する理由が存在する場合は、ID ビューを使用して、IdM でユーザーが作成したときに生成された IdM ユーザーの UID をオーバーライドできます。

重要

ID ビューは、IdM サーバーではなく、IdM クライアントにのみ適用できます。

関連情報

- Active Directory ユーザーの ID ビューの使用
- SSSD クライアント側のビュー

37.2. ID ビューが SSSD パフォーマンスに対して与える可能性のある悪影響

ID ビューを定義すると、IdM は指定の上書き値を、IdM サーバーの System Security Services Daemon (SSSD) キャッシュに配置します。その後、IdM クライアントで実行している SSSD は、サーバーキャッシュから上書き値を取得します。

特定の最適化と ID ビューを同時に実行できないため、ID ビューを適用すると、SSSD (System Security Services Daemon) のパフォーマンスに悪影響を与える可能性があります。たとえば、ID ビューは、SSSD がサーバー上でグループを検索するプロセスの最適化を防ぎます。

- ID ビューを使用すると、グループ名が上書きされた場合、SSSD は返されたグループメンバー名リストの各メンバーをチェックする必要があります。
- ID ビューを使用しないと、SSSD はグループオブジェクトのメンバー属性からユーザー名だけを収集できます。

SSSD のキャッシュが空の場合や、キャッシュを消去した場合に、この負の影響が現れ、全エントリーが無効になります。

37.3. ID ビューがオーバーライドできる属性

ID ビューは、ユーザーおよびグループ ID のオーバーライドで構成されます。上書きは、新しい POSIX 属性値を定義します。

ユーザー ID およびグループ ID のオーバーライドは、以下の POSIX 属性の新しい値を定義できます。

ユーザー属性

- ログイン名 (**uid**)
- GECOS エントリー (**gecos**)
- UID 番号 (**uidNumber**)
- GID 番号 (**gidNumber**)
- ログインシェル (**loginShell**)
- ホームディレクトリー (**homeDirectory**)
- SSH 公開鍵 (**ipaSshPubkey**)
- 証明書 (**userCertificate**)

グループ属性

- グループ名 (**cn**)
- グループ GID 番号 (**gidNumber**)

37.4. ID ビューのコマンドのヘルプの取得

IdM コマンドラインインターフェイス (CLI) で、Identity Management (IdM) ID ビューに関連するコマンドのヘルプを表示できます。

前提条件

- IdM ユーザーの Kerberos チケットを取得している。

手順

- ID ビューおよびオーバーライドの管理に使用するコマンドをすべて表示するには、次のコマンドを実行します。

```
$ ipa help idviews
ID Views

Manage ID Views

IPA allows to override certain properties of users and groups[...]
[...]
Topic commands:
  idoverridegroup-add          Add a new Group ID override
  idoverridegroup-del          Delete a Group ID override
[...]
```

- 特定のコマンドの詳細なヘルプを表示するには、コマンドに **--help** オプションを追加します。

```
$ ipa idview-add --help
Usage: ipa [global-options] idview-add NAME [options]

Add a new ID View.
Options:
  -h, --help      show this help message and exit
  --desc=STR      Description
[...]
```

37.5. ID ビューを使用して、特定のホストにある IDM ユーザーのログイン名をオーバーライドする

特定の IdM ユーザーに関連付けられた POSIX 属性値をオーバーライドする特定の IdM クライアントの ID ビューを作成するには、次の手順に従います。この手順では、**idm_user** という名前の IdM ユーザーが、**user_1234** ログイン名を使用して **host1** という名前の IdM クライアントにログインできるようにする ID ビューの例を使用します。

前提条件

- IdM 管理者としてログインしている。

手順

1. 新しい ID ビューを作成します。たとえば、**example_for_host1** という名前の ID ビューを作成するには、次のコマンドを実行します。

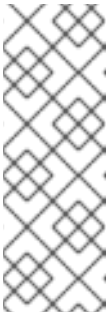
```
$ ipa idview-add example_for_host1
---------------------------
Added ID View "example_for_host1"
---------------------------
  ID View Name: example_for_host1
```

2. ユーザーの上書きを **example_for_host1** ID ビューに追加します。ユーザーログインをオーバーライドするには、以下を実行します。

   - **ipa idoverrideuser-add** コマンドを入力します。
   - ID ビューの名前を追加します。
   - ユーザー名 (アンカーとも呼ばれます) を追加します。
   - **--login** オプションを追加します。

   ```
   $ ipa idoverrideuser-add example_for_host1 idm_user --login=user_1234
   ----------------------------
   Added User ID override "idm_user"
   ----------------------------
     Anchor to override: idm_user
     User login: user_1234
   ```

   利用可能なオプションのリストは、ipa idoverrideuser-add --help を実行します。



注記

ipa idoverrideuser-add --certificate コマンドは、指定された ID ビューのアカウントの既存証明書をすべて置き換えます。別の証明書を追加するには、代わりに **ipa idoverrideuser-add-cert** コマンドを使用します。

```
$ ipa idoverrideuser-add-cert example_for_host1 user --
certificate="MIIEATCC..."
```

3. 必要に応じて、**ipa idoverrideuser-mod** コマンドを使用すると、既存のユーザーの上書きに新しい属性値を指定できます。

4. **example_for_host1** を **host1.idm.example.com** ホストに適用します。

```
$ ipa idview-apply example_for_host1 --hosts=host1.idm.example.com
----------------------------
Applied ID View "example_for_host1"
----------------------------
hosts: host1.idm.example.com
------------------------------------------
Number of hosts the ID View was applied to: 1
------------------------------------------
```

注記

ipa idview-apply コマンドでは、**--hostgroups** オプションも使用できます。このオプションは、ID ビューを、指定のホストグループに所属するホストに適用しますが、ホストグループ自体との関連付けは行いません。代わりに、**--hostgroups** オプションは指定されたホストグループのメンバーを拡張して、**--hosts** オプションを個別にすべて適用します。

つまり、今後ホストがホストグループに追加されても、ID ビューは新しいホストには適用されません。

5. 新しい設定を **host1.idm.example.com** システムに適用するには、次のコマンドを実行します。

   a. root でシステムに対して SSH 接続します。

   ```
   $ ssh root@host1
   Password:
   ```

   b. SSSD キャッシュを削除します。

   ```
   root@host1 ~]# sss_cache -E
   ```

   c. SSSD デーモンを再起動します。

   ```
   root@host1 ~]# systemctl restart sssd
   ```

検証手順

- **user_1234** の認証情報がある場合は、その認証情報を使用して **host1** で IdM にログインできます。

  1. ログイン名 **user_1234** を使用して **host1** に SSH 接続します。

  ```
  [root@r8server ~]# ssh user_1234@host1.idm.example.com
  Password:

  Last login: Sun Jun 21 22:34:25 2020 from 192.168.122.229
  [user_1234@host1 ~]$
  ```

  2. 作業ディレクトリーを表示します。

  ```
  [user_1234@host1 ~]$ pwd
  /home/idm_user/
  ```

- または、**host1** に root 認証情報がある場合は、その認証情報を使用して **idm_user** および **user_1234** の **id** コマンドの出力を確認できます。

```
[root@host1 ~]# id idm_user
uid=779800003(user_1234) gid=779800003(idm_user) groups=779800003(idm_user)
[root@host1 ~]# user_1234
uid=779800003(user_1234) gid=779800003(idm_user) groups=779800003(idm_user)
```

37.6. IDM ID ビューの変更

Identity Management (IdM) の ID ビューは、特定の IdM ユーザーに関連する POSIX 属性値をオーバーライドします。既存の ID ビューを変更するには、次の手順に従ってください。具体的には、**idm_user** という名前のユーザーが IdM クライアントの **host1.idm.example.com** の **/home/idm_user/** ではなく、ユーザーのホームディレクトリーとして **/home/user_1234/** ディレクトリーを使用できるように ID ビューを変更する方法を説明します。

前提条件

- **host1.idm.example.com** への root アクセス権限がある。

- **admin** などの必要な権限を持つユーザーとしてログインしている。
- IdM クライアント **host1** に適用される **idm_user** の ID ビューが設定されている。

手順

1. root で、**idm_user** がユーザーのホームディレクトリーとして **host1.idm.example.com** で使用するディレクトリーを作成します。

```
[root@host1 /]# mkdir /home/user_1234/
```

2. ディレクトリーの所有権を変更します。

```
[root@host1 /]# chown idm_user:idm_user /home/user_1234/
```

3. ID ビューが現在適用されているホストを含め、ID ビューを表示します。**example_for_host1** という名前の ID ビューを表示するには、次のコマンドを実行します。

```
$ ipa idview-show example_for_host1 --all
  dn: cn=example_for_host1,cn=views,cn=accounts,dc=idm,dc=example,dc=com
  ID View Name: example_for_host1
  User object override: idm_user
  Hosts the view applies to: host1.idm.example.com
  objectclass: ipaIDView, top, nsContainer
```

この出力は、現在 ID ビューが **host1.idm.example.com** に適用されることを示しています。

4. **example_for_host1** ID ビューのユーザーのオーバーライドを変更します。ユーザーのホームディレクトリーをオーバーライドするには、次のコマンドを実行します。
   - **ipa idoverrideuser-add** コマンドを入力します。
   - ID ビューの名前を追加します。
   - ユーザー名 (アンカーとも呼ばれます) を追加します。
   - **--homedir** オプションを追加します。

```
$ ipa idoverrideuser-mod example_for_host1 idm_user --
homedir=/home/user_1234
-----------------------------
Modified a User ID override "idm_user"
-----------------------------
  Anchor to override: idm_user
  User login: user_1234
  Home directory: /home/user_1234/
```

利用可能なオプションのリストは、**ipa idoverrideuser-mod --help** を実行します。

5. 新しい設定を **host1.idm.example.com** システムに適用するには、次のコマンドを実行します。
   a. root でシステムに対して SSH 接続します。

```
$ ssh root@host1
Password:
```

b. SSSD キャッシュを削除します。

```
root@host1 ~]# sss_cache -E
```

c. SSSD デーモンを再起動します。

```
root@host1 ~]# systemctl restart sssd
```

検証手順

1. **idm_user** として **host1** に **SSH** 接続します。

```
[root@r8server ~]# ssh idm_user@host1.idm.example.com
Password:

Last login: Sun Jun 21 22:34:25 2020 from 192.168.122.229
[user_1234@host1 ~]$
```

2. 作業ディレクトリーを出力します。

```
[user_1234@host1 ~]$ pwd
/home/user_1234/
```

関連情報

- デフォルト信頼ビューの変更による AD ユーザーのグローバル属性の定義

37.7. IDM クライアントで IDM ユーザーのホームディレクトリーをオーバーライドする ID ビューの追加

Identity Management (IdM) の ID ビューは、特定の IdM ユーザーに関連する POSIX 属性値をオーバーライドします。この手順では、**host1** という名前の IdM クライアントの **idm_user** に適用される ID ビューを作成し、ユーザーが **/home/idm_user/** ではなく、ユーザーホームディレクトリーとして **/home/user_1234/** ディレクトリーを使用できるようにします。

前提条件

- **host1.idm.example.com** への root アクセス権限がある。
- **admin** などの必要な権限を持つユーザーとしてログインしている。

手順

1. root で、**idm_user** がユーザーのホームディレクトリーとして **host1.idm.example.com** で使用するディレクトリーを作成します。

```
[root@host1 /]# mkdir /home/user_1234/
```

2. ディレクトリーの所有権を変更します。

```
[root@host1 /]# chown idm_user:idm_user /home/user_1234/
```

3. ID ビューを作成します。たとえば、**example_for_host1** という名前の ID ビューを作成するには、次のコマンドを実行します。

```
$ ipa idview-add example_for_host1
---------------------------
Added ID View "example_for_host1"
---------------------------
  ID View Name: example_for_host1
```

4. ユーザーの上書きを **example_for_host1** ID ビューに追加します。ユーザーのホームディレクトリーをオーバーライドするには、次のコマンドを実行します。

   - **ipa idoverrideuser-add** コマンドを入力します。
   - ID ビューの名前を追加します。
   - ユーザー名 (アンカーとも呼ばれます) を追加します。
   - **--homedir** オプションを追加します。

```
$ ipa idoverrideuser-add example_for_host1 idm_user --homedir=/home/user_1234
----------------------------
Added User ID override "idm_user"
----------------------------
  Anchor to override: idm_user
  Home directory: /home/user_1234/
```

5. **example_for_host1** を **host1.idm.example.com** ホストに適用します。

```
$ ipa idview-apply example_for_host1 --hosts=host1.idm.example.com
----------------------------
Applied ID View "example_for_host1"
----------------------------
hosts: host1.idm.example.com
-------------------------------------------
Number of hosts the ID View was applied to: 1
-------------------------------------------
```

注記

ipa idview-apply コマンドでは、**--hostgroups** オプションも使用できます。このオプションは、ID ビューを、指定のホストグループに所属するホストに適用しますが、ホストグループ自体との関連付けは行いません。代わりに、**--hostgroups** オプションは指定されたホストグループのメンバーを拡張して、**--hosts** オプションを個別にすべて適用します。

つまり、今後ホストがホストグループに追加されても、ID ビューは新しいホストには適用されません。

6. 新しい設定を **host1.idm.example.com** システムに適用するには、次のコマンドを実行します。

   a. root でシステムに対して SSH 接続します。

```
$ ssh root@host1
Password:
```

b. SSSD キャッシュを削除します。

```
root@host1 ~]# sss_cache -E
```

c. SSSD デーモンを再起動します。

```
root@host1 ~]# systemctl restart sssd
```

検証手順

1. **idm_user** として **host1** に **SSH** 接続します。

```
[root@r8server ~]# ssh idm_user@host1.idm.example.com
Password:
Activate the web console with: systemctl enable --now cockpit.socket

Last login: Sun Jun 21 22:34:25 2020 from 192.168.122.229
[idm_user@host1 /]$
```

2. 作業ディレクトリーを出力します。

```
[idm_user@host1 /]$ pwd
/home/user_1234/
```

関連情報

- ID ビューを使用して IdM クライアント上の AD ユーザーの Default Trust View の属性をオーバーライドする

37.8. IDM ホストグループへの ID ビューの適用

ipa idview-apply コマンドでは、**--hostgroups** オプションを使用できます。ただし、このオプションは、指定のホストに現在所属するホストに ID ビューを適用する 1 回限りの操作として機能しますが、ホストグループ自体と ID ビューを動的に関連付けることはありません。**--hostgroups** オプションは、指定したホストグループのメンバーを拡張して、**--hosts** オプションを個別にすべて適用します。

新しいホストを後でホストグループに追加する場合は、**--hosts** オプションで **ipa idview-apply** コマンドを使用して、新しいホストに ID ビューを手動で適用する必要があります。

同様に、ホストグループからホストを削除すると、ID ビューは削除後でも、ホストに割り当てられます。削除されたホストから ID ビューの適用を解除するには、**ipa idview-unapply id_view_name --hosts=name_of_the_removed_host** コマンドを実行する必要があります。

次の目標を達成するには、次の手順に従ってください。

1. ホストグループを作成し、そのグループにホストを追加する方法。
2. ID ビューをホストグループに適用する方法。
3. ホストグループに新規ホストを追加し、ID ビューを新しいホストに適用する方法。

前提条件

- ホストグループに適用する ID ビューが IdM に存在することを確認します。たとえば、AD ユーザーの GID をオーバーライドする ID ビューを作成するには、ID ビューを使用して IdM クライアント上の AD ユーザーの Default Trust View の属性をオーバーライドする を参照してください。

手順

1. ホストグループを作成し、そのグループにホストを追加します。

   a. ホストグループを作成します。たとえば、**baltimore** という名前のホストグループを作成するには、次のコマンドを実行します。

   ```
   [root@server ~]# ipa hostgroup-add --desc="Baltimore hosts" baltimore
   ----------------------------
   Added hostgroup "baltimore"
   ----------------------------
   Host-group: baltimore
   Description: Baltimore hosts
   ```

   b. ホストグループにホストを追加します。たとえば、**host102** および **host103** を **baltimore** ホストグループに追加するには、次のコマンドを実行します。

   ```
   [root@server ~]# ipa hostgroup-add-member --hosts={host102,host103} baltimore
   Host-group: baltimore
   Description: Baltimore hosts
   Member hosts: host102.idm.example.com, host103.idm.example.com
   -------------------------
   Number of members added 2
   -------------------------
   ```

2. ホストグループのホストに ID ビューを適用します。たとえば、**example_for_host1** ID ビューを **baltimore** ホストグループに適用するには、次のコマンドを実行します。

   ```
   [root@server ~]# ipa idview-apply --hostgroups=baltimore
   ID View Name: example_for_host1
   ---------------------------------------
   Applied ID View "example_for_host1"
   ---------------------------------------
     hosts: host102.idm.example.com, host103.idm.example.com
   -----------------------------------------------
   Number of hosts the ID View was applied to: 2
   -----------------------------------------------
   ```

3. ホストグループに新規ホストを追加し、ID ビューを新しいホストに適用します。

   a. ホストグループに新規ホストを追加します。たとえば、**somehost.idm.example.com** ホストを **baltimore** ホストグループに追加するには、次のコマンドを実行します。

   ```
   [root@server ~]# ipa hostgroup-add-member --hosts=somehost.idm.example.com
   baltimore
     Host-group: baltimore
     Description: Baltimore hosts
     Member hosts:  host102.idm.example.com,
   ```

```
  host103.idm.example.com,somehost.idm.example.com
  -------------------------
  Number of members added 1
  -------------------------
```

b. 必要に応じて、ID ビュー情報を表示します。たとえば、**example_for_host1** ID ビューの詳細を表示するには、次のコマンドを実行します。

```
[root@server ~]# ipa idview-show example_for_host1 --all
  dn: cn=example_for_host1,cn=views,cn=accounts,dc=idm,dc=example,dc=com
  ID View Name: example_for_host1
[...]
  Hosts the view applies to: host102.idm.example.com, host103.idm.example.com
  objectclass: ipaIDView, top, nsContainer
```

この出力では、ID ビューが **baltimore** ホストグループに新規追加されたホスト **somehost.idm.example.com** に適用されていないことが分かります。

c. ID ビューを新規ホストに適用します。たとえば、ID ビュー **example_for_host1** を **somehost.idm.example.com** に適用するには、次のコマンドを実行します。

```
[root@server ~]# ipa idview-apply --host=somehost.idm.example.com
ID View Name: example_for_host1
-----------------------------------------
Applied ID View "example_for_host1"
-----------------------------------------
  hosts: somehost.idm.example.com
-----------------------------------------------
Number of hosts the ID View was applied to: 1
-----------------------------------------------
```

検証手順

- ID ビュー情報を再度表示します。

```
[root@server ~]# ipa idview-show example_for_host1 --all
  dn: cn=example_for_host1,cn=views,cn=accounts,dc=idm,dc=example,dc=com
  ID View Name: example_for_host1
[...]
  Hosts the view applies to: host102.idm.example.com, host103.idm.example.com,
somehost.idm.example.com
  objectclass: ipaIDView, top, nsContainer
```

この出力は、ID ビューが **baltimore** ホストグループに新規追加されたホスト **somehost.idm.example.com** に適用されていることを示しています。

37.9. ANSIBLE を使用して、特定のホスト上の IDM ユーザーのログイン名とホームディレクトリーをオーバーライドする

この手順では、**idoverrideuser ansible-freeipa** モジュールを使用して、特定の Identity Management (IdM) ユーザーに関連付けられた POSIX 属性値をオーバーライドする特定の IdM クライアント用の ID ビューを作成します。この手順では、**idm_user** という名前の IdM ユーザーがログイン名 **user_1234** を使用して **host1.idm.example.com** という名前の IdM クライアントにログインできるようにする ID

ビューの例を使用します。さらに、この ID ビューは idm_user のホームディレクトリーを変更します。そのため、host1 へのログイン後、ユーザーのホームディレクトリーが **/home/user_1234/** になります。

前提条件

- コントロールノードでは、
  - Ansible バージョン 2.14 以降を使用している。
  - **ansible-freeipa** パッケージがインストールされている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している。RHEL 9.4 以降を使用している。
  - **secret.yml** Ansible vault に **ipaadmin_password** が保存されている。
- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。

手順

1. 次の内容を含む Ansible Playbook ファイル **add-idoverrideuser-with-name-and-homedir.yml** を作成します。

```
---
- name: Playbook to manage idoverrideuser
  hosts: ipaserver
  become: false
  gather_facts: false
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml

  tasks:
  - name: Ensure idview_for_host1 is present
    idview:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: idview_for_host1
  - name: Ensure idview_for_host1 is applied to host1.idm.example.com
    idview:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: idview_for_host1
      host: host1.idm.example.com
      action: member
  - name: Ensure idm_user is present in idview_for_host1 with homedir /home/user_1234
and name user_1234
    ipaidoverrideuser:
      ipaadmin_password: "{{ ipaadmin_password }}"
      idview: idview_for_host1
      anchor: idm_user
      name: user_1234
      homedir: /home/user_1234
```

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

```
$ ansible-playbook --vault-password-file=password_file -v -i
<path_to_inventory_directory>/inventory <path_to_playbooks_directory>/add-
idoverrideuser-with-name-and-homedir.yml
```

3. [オプション] **root** の認証情報を持っている場合は、新しい設定を **host1.idm.example.com** システムにすぐに適用できます。

   a. システムに **root** として SSH 接続します。

   ```
   $ ssh root@host1
   Password:
   ```

   b. SSSD キャッシュを削除します。

   ```
   root@host1 ~]# sss_cache -E
   ```

   c. SSSD デーモンを再起動します。

   ```
   root@host1 ~]# systemctl restart sssd
   ```

検証

1. **idm_user** として **host1** に **SSH** 接続します。

```
[root@r8server ~]# ssh idm_user@host1.idm.example.com
Password:

Last login: Sun Jun 21 22:34:25 2020 from 192.168.122.229
[user_1234@host1 ~]$
```

2. 作業ディレクトリーを出力します。

```
[user_1234@host1 ~]$ pwd
/home/user_1234/
```

関連情報

- **ansible-freeipa** アップストリームドキュメントの idoverrideuser モジュール

37.10. ANSIBLE を使用して、IDM クライアントへの SSH 鍵ログインを有効にする ID ビューを設定する

この手順では、**idoverrideuser ansible-freeipa** モジュールを使用して、IdM ユーザーが特定の SSH 鍵を使用して特定の IdM クライアントにログインできるようにします。この手順では、**idm_user** という名前の IdM ユーザーが SSH 鍵を使用して **host1.idm.example.com** という名前の IdM クライアントにログインできるようにする ID ビューの例を使用します。

注記

この ID ビューは、特定の HBAC ルールを強化するために使用できます。

前提条件

- コントロールノードでは、
  - Ansible バージョン 2.14 以降を使用している。
  - **ansible-freeipa** パッケージがインストールされている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している。RHEL 9.4 以降を使用している。
  - **secret.yml** Ansible vault に **ipaadmin_password** が保存されている。
- **idm_user** の SSH 公開鍵にアクセスできる。
- **idview_for_host1** ID ビューが存在する。
- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。

手順

1. 次の内容を含む Ansible Playbook ファイル **ensure-idoverrideuser-can-login-with-sshkey.yml** を作成します。

```
---
- name: Playbook to manage idoverrideuser
  hosts: ipaserver
  become: false
  gather_facts: false
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml

  tasks:
  - name: Ensure test user idm_user is present in idview idview_for_host1 with sshpubkey
    ipaidoverrideuser:
      ipaadmin_password: "{{ ipaadmin_password }}"
      idview: idview_for_host1
      anchor: idm_user
      sshpubkey:
      - ssh-rsa AAAAB3NzaC1yc2EAAADAQABAAABgQCqmVDpEX5gnSjKuv97Ay ...
  - name: Ensure idview_for_host1 is applied to host1.idm.example.com
    ipaidview:
      ipaadmin_password:  "{{ ipaadmin_password }}"
      name: idview_for_host1
      host: host1.idm.example.com
      action: member
```

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

```
$ ansible-playbook --vault-password-file=password_file -v -i
<path_to_inventory_directory>/inventory <path_to_playbooks_directory>/ensure-
idoverrideuser-can-login-with-sshkey.yml
```

3. [オプション] **root** の認証情報を持っている場合は、新しい設定を **host1.idm.example.com** システムにすぐに適用できます。

   a. システムに **root** として SSH 接続します。

   ```
   $ ssh root@host1
   Password:
   ```

   b. SSSD キャッシュを削除します。

   ```
   root@host1 ~]# sss_cache -E
   ```

   c. SSSD デーモンを再起動します。

   ```
   root@host1 ~]# systemctl restart sssd
   ```

検証

- 公開鍵を使用して **host1** に **SSH** 接続します。

```
[root@r8server ~]# ssh -i ~/.ssh/id_rsa.pub idm_user@host1.idm.example.com

Last login: Sun Jun 21 22:34:25 2023 from 192.168.122.229
[idm_user@host1 ~]$
```

出力により、正常にログインしたことが確認されます。

関連情報

- **ansible-freeipa** アップストリームドキュメントの idoverrideuser モジュール

37.11. ANSIBLE を使用して、IDM クライアントのローカルサウンドカードへのユーザー ID オーバーライドアクセス権を付与する

ansible-freeipa group および **idoverrideuser** モジュールを使用して、Identity Management (IdM) または Active Directory (AD) ユーザーを IdM クライアント上の **audio** ローカルグループのメンバーにすることができます。これにより、IdM または AD ユーザーに、ホスト上のサウンドカードへの特権アクセスが付与されます。この手順で使用する例では、最初の Playbook タスクで **Default Trust View** ID ビューに **aduser@addomain.com** ID オーバーライドを追加します。次の Playbook タスクで、RHEL ホスト上の **audio** ローカルグループの GID に対応する GID 63 の **audio** グループを IdM に作成します。同時に、aduser@addomain.com ID オーバーライドを IdM オーディオグループにメンバーとして追加します。

前提条件

- 手順の最初の部分を実行する対象である IdM クライアントへの **root** アクセス権を持っている。この例では、これは **client.idm.example.com** です。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。

- RHEL 9.4 以降を使用している。
- **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
- この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。

- AD フォレストが IdM と信頼関係にある。この例では、AD ドメインの名前は **addomain.com** であり、ローカルグループ **audio** に存在することを確認する AD ユーザーの完全修飾ドメイン名 (FQDN) は **aduser@addomain.com** です。
- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。

手順

1. **client.idm.example.com** で、**/etc/nsswitch.conf** ファイルに **[SUCCESS=merge]** を追加します。

```
[...]
# Allow initgroups to default to the setting for group.
initgroups: sss [SUCCESS=merge] files
```

2. **audio** ローカルグループの GID を特定します。

```
$ getent group audio
---------------------
audio:x:63
```

3. Ansible コントロールノードで、**aduser@addomain.com** ユーザーオーバーライドを Default Trust View に追加するタスクを含む **add-aduser-to-audio-group.yml** Playbook を作成します。

```
---
- name: Playbook to manage idoverrideuser
  hosts: ipaserver
  become: false

  tasks:
  - name: Add aduser@addomain.com user to the Default Trust View
    ipaidoverrideuser:
      ipaadmin_password: "{{ ipaadmin_password }}"
      idview: "Default Trust View"
      anchor: aduser@addomain.com
```

4. 同じ Playbook 内の別の Playbook タスクを使用して、**GID** 63 を持つグループ **audio** を IdM に追加します。aduser idoverrideuser をグループに追加します。

```
  - name: Add the audio group with the aduser member and GID of 63
    ipagroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: audio
```

```
idoverrideuser:
- aduser@addomain.com
gidnumber: 63
```

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

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory add-aduser-to-audio-group.yml
```

検証

1. AD ユーザーとして IdM クライアントにログインします。

```
$ ssh aduser@addomain.com@client.idm.example.com
```

2. AD ユーザーのグループメンバーシップを確認します。

```
$ id aduser@addomain.com
uid=702801456(aduser@addomain.com) gid=63(audio) groups=63(audio)
```

関連情報

- idoverrideuser および ipagroup に関する **ansible-freeipa** アップストリームドキュメント
- IdM でのローカルグループとリモートグループのグループマージの有効化

37.12. ANSIBLE を使用して、特定の UID を持つ IDM ユーザーが ID ビューに存在することを確認する

自分のコンピューターがあるラボで作業していて、サーバーによってエクスポートされた共有ドライブ内に **/home/** ディレクトリーがある場合、次の 2 つのユーザーが存在する場合があります。

- Identity Management (IdM) に一元的に保存されている、システム全体のユーザー
- 当該システムに保存されている、アカウントがローカルであるユーザー

IdM ユーザーとしてログインしている場合でも、ローカルユーザーとしてログインしている場合でも、ファイルへのフルアクセスが必要な場合は、両方のユーザーに同じ **UID** を付与することでそれが可能になります。

この手順では、**ansible-freeipa idoverrideuser** モジュールを使用して以下を実行します。

- **idview_for_host01** という名前の ID ビューを host01 に適用します。
- idview_for_host01 で、**UID** 20001 を持つ idm_user のユーザー ID オーバーライドが存在することを確認します。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。

- Ansible バージョン 2.14 以降を使用している。
- Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
- RHEL 9.4 以降を使用している。
- **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
- この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。

- **idview_for_host1** ID ビューが存在する。

手順

1. Ansible コントロールノードで、次の内容を含む **ensure-idmuser-and-local-user-have-access-to-same-files.yml** Playbook を作成します。

```
---
- name: Ensure both local user and IdM user have access to same files
  hosts: ipaserver
  become: false
  gather_facts: false

  tasks:
  - name: Ensure idview_for_host1 is applied to host1.idm.example.com
    ipaidview:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: idview_for_host01
      host: host1.idm.example.com
  - name: Ensure idmuser is present in idview_for_host01 with the UID of 20001
    ipaidoverrideuser:
      ipaadmin_password: "{{ ipaadmin_password }}"
      idview: idview_for_host01
      anchor: idm_user
      UID: 20001
```

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

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory ensure-idmuser-and-local-user-have-access-to-same-files.yml
```

関連情報

- **ansible-freeipa** アップストリームドキュメントの idoverrideuser モジュール

37.13. ANSIBLE を使用して、IDM ユーザーが 2 つの証明書を使用して IDM クライアントにログインできるようにする

通常はパスワードを使用して IdM にログインする Identity Management (IdM) ユーザーが、スマートカードのみを使用して特定の IdM クライアントに認証されるようにする場合は、そのクライアントでそのユーザーの証明を要求する ID ビューを作成します。

この手順では、**ansible-freeipa idoverrideuser** モジュールを使用して以下を実行します。

- **idview_for_host01** という名前の ID ビューを host01 に適用します。
- idview_for_host01 で、2 つの証明書を持つ **idm_user** のユーザー ID オーバーライドが存在することを確認します。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - RHEL 9.4 以降を使用している。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
  - この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。
  - この例では、**cert1.b64** および **cert2.b64** 証明書が、Playbook を実行しているディレクトリーと同じディレクトリーにあることを前提としています。
- **idview_for_host01** ID ビューが存在する。

手順

1. Ansible コントロールノードで、次の内容の **ensure-idmuser-present-in-idview-with-certificates.yml** Playbook を作成します。

```
---
- name: Ensure both local user and IdM user have access to same files
  hosts: ipaserver
  become: false
  gather_facts: false

  tasks:
  - name: Ensure idview_for_host1 is applied to host01.idm.example.com
    ipaidview:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: idview_for_host01
      host: host01.idm.example.com

  - name: Ensure an IdM user is present in ID view with two certificates
    ipaidoverrideuser:
      ipaadmin_password: "{{ ipaadmin_password }}"
      idview: idview_for_host01
      anchor: idm_user
```

```
    certificate:
    - "{{ lookup('file', 'cert1.b64', rstrip=False) }}"
    - "{{ lookup('file', 'cert2.b64', rstrip=False) }}"
```

rstrip=False ディレクティブにより、検索されたファイルの末尾から空白が削除されるのを防ぎます。

2. ファイルを保存します。

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

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory ensure-idmuser-present-in-idview-with-certificates.yml
```

関連情報

- **ansible-freeipa** アップストリームドキュメントの idoverrideuser モジュール

37.14. ANSIBLE を使用して、IDM グループに IDM クライアントのサウンドカードへのアクセス権を付与する

ansible-freeipa の **idview** および **idoverridegroup** モジュールを使用して、Identity Management (IdM) または Active Directory (AD) ユーザーを IdM クライアント上の **audio** ローカルグループのメンバーにすることができます。これにより、IdM または AD ユーザーに、ホスト上のサウンドカードへの特権アクセスが付与されます。

この手順で使用する例では、**idview_for_host01** ID ビューに、**GID** 63 を持つ **audio** グループの ID オーバーライドを追加します。この GID は、RHEL ホスト上の **audio** ローカルグループの GID に対応するものです。**idview_for_host01** ID ビューは、**host01.idm.example.com** という名前の IdM クライアントに適用されます。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - RHEL 9.4 以降を使用している。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
  - この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。

手順

1. [オプション] RHEL ホスト上の **audio** ローカルグループの GID を特定します。

```
$ getent group audio
----------
audio:x:63
```

2. Ansible コントロールノードで、次のタスクを含む **give-idm-group-access-to-sound-card-on-idm-client.yml** Playbook を作成します。

```
---
- name: Playbook to give IdM group access to sound card on IdM client
  hosts: ipaserver
  become: false

  tasks:
  - name: Ensure the audio group exists in IdM
    ipagroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: audio

  - name: Ensure idview_for_host01 exists and is applied to host01.idm.example.com
    ipaidview:
      ipaadmin_password:  "{{ ipaadmin_password }}"
      name: idview_for_host01
      host: host01.idm.example.com

  - name: Add an override for the IdM audio group with GID 63 to idview_for_host01
    ipaidoverridegroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      idview: idview_for_host01
      anchor: audio
      GID: 63
```

3. ファイルを保存します。

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

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory give-idm-group-access-to-sound-card-on-idm-client.yml
```

検証

1. IdM クライアントで、IdM 管理者の認証情報を取得します。

```
$ kinit admin
Password:
```

2. テスト IdM ユーザーを作成します。

```
$ ipa user-add testuser --first test --last user --password
User login [tuser]:
Password:
Enter Password again to verify:
----------------------
Added user "tuser"
----------------------
```

3. ユーザーを IdM オーディオグループに追加します。

```
$ ipa group-add-member --tuser audio
```

4. tuser として host01.idm.example.com にログインします。

```
$ ssh tuser@host01.idm.example.com
```

5. ユーザーのグループメンバーシップを確認します。

```
$ id tuser
uid=702801456(tuser) gid=63(audio) groups=63(audio)
```

関連情報

- idoverridegroup、idview、ipagroup に関する **ansible-freeipa** アップストリームドキュメント
- IdM でのローカルグループとリモートグループのグループマージの有効化

37.15. NIS ドメインの IDENTITY MANAGEMENT への移行

NIS ドメインを IdM に移行する際に、ファイルやディレクトリーのパーミッションを変更しないように、ID ビューを使用して既存のホストにホスト固有の UID と GID を設定することができます。

前提条件

- **kinit admin** コマンドを使用して、管理者として自分自身を認証済みである。

手順

1. IdM ドメインにユーザーとグループを追加します。
   a. **ipa user-add** コマンドを使用して、ユーザーを作成します。詳細は、IdM へのユーザーの追加 を参照してください。
   b. **ipa group-add** コマンドを使用して、グループを作成します。詳細は、IdM へのグループの追加 を参照してください。
2. ユーザーの作成中に生成された IDs IdM をオーバーライドします。
   a. **ipa idview-add** コマンドを使用して、新しい ID ビューを作成します。詳細は、ID ビューコマンドのヘルプの取得 を参照してください。
   b. **ipa idoverrideuser-add** および **idoverridegroup-add** をそれぞれ使用して、ユーザーとグループの ID オーバーライドを ID ビューに追加します。
3. **ipa idview-apply** コマンドを使用して、特定のホストに ID ビューを割り当てます。
4. NIS ドメインの使用を停止します。

検証

1. すべてのユーザーとグループが ID ビューに正しく追加されたかどうかを確認するには、**ipa idview-show** コマンドを使用します。

```
$ ipa idview-show example-view
```

```
ID View Name: example-view
User object overrides: example-user1
Group object overrides: example-group
```

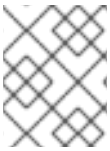
第38章 ACTIVE DIRECTORY ユーザーの ID ビューの使用

IdM-AD 信頼環境において、Active Directory (AD) ユーザーの POSIX 属性に新しい値を指定するために、ID ビューを使用することができます。

デフォルトでは、IdM はすべての AD ユーザーに**デフォルト信頼ビュー** を適用します。個々の IdM クライアントで追加の ID ビューを設定することで、特定のユーザーが受け取る POSIX 属性をさらに調整することができます。

38.1. デフォルト信頼ビューの仕組み

デフォルト信頼ビュー、信頼ベースのセットアップで AD ユーザーとグループに常に適用されるデフォルトの ID ビューです。これは、**ipa-adtrust-install** コマンドを使用して信頼を確立する際に自動的に作成され、削除することはできません。



注記

デフォルト信頼ビューは AD ユーザーおよびグループのオーバーライドのみを受け入れ、IdM ユーザーおよびグループのオーバーライドは受け入れません。

Default Trust View を使用すると、AD ユーザーおよびグループのカスタム POSIX 属性を定義できます。これにより、AD で定義された値を上書きできます。

表38.1 Default Trust View の適用

	AD の値	Default Trust View	結果
Login	ad_user	ad_user	ad_user
UID	111	222	222
GID	111	(値なし)	111

追加の ID ビューを設定して、IdM クライアントのデフォルト信頼ビューをオーバーライドすることもできます。IdM は、デフォルト信頼ビューの上に、ホスト固有の ID ビューからの値を適用します。

- ホスト固有の ID ビューに属性が定義されている場合、IdM はこの ID ビューの値を適用します。
- ホスト固有の ID ビューで属性が定義されていない場合、IdM はデフォルト信頼ビューからの値を適用します。

表38.2 デフォルト信頼ビューの上にホスト固有の ID ビューを適用する

	AD の値	Default Trust View	ホスト固有の ID ビュー	結果
Login	ad_user	ad_user	(値なし)	ad_user
UID	111	222	333	333

	AD の値	Default Trust View	ホスト固有の ID ビュー	結果
GID	111	(値なし)	333	333

注記

ホスト固有の ID ビューのみを適用して、IdM クライアントのデフォルト信頼ビューをオーバーライドできます。IdM サーバーとレプリカは、常にデフォルト信頼ビューの値を適用します。

関連情報

- ID ビューを使用した IdM クライアントのユーザー属性値のオーバーライド

38.2. デフォルト信頼ビューの変更による AD ユーザーのグローバル属性の定義

Active Directory (AD) ユーザーの POSIX 属性を IdM デプロイメント全体を通じてオーバーライドしたい場合は、デフォルト信頼ビューでそのユーザーのエントリーを変更します。この手順では、AD ユーザー **ad_user@ad.example.com** の GID を 732000006 に設定します。

前提条件

- IdM 管理者として認証されている。
- グループが GID とともに存在するか、グループの ID オーバーライドで GID を設定する必要があります。

手順

1. IdM 管理者として、GID 番号を 732000006 に変更する AD ユーザーの ID オーバーライドをデフォルト信頼ビューに作成してください。

```
# ipa idoverrideuser-add 'Default Trust View' ad_user@ad.example.com --gidnumber=732000006
```

2. すべての IdM サーバーとクライアントの SSSD キャッシュから **ad_user@ad.example.com** ユーザーのエントリーをクリアします。これにより、古いデータが削除され、新しいオーバーライド値が適用されるようになります。

```
# sssctl cache-expire -u ad_user@ad.example.com
```

検証

- **ad_user@ad.example.com** ユーザーの情報を取得して、GID が更新された値を反映することを確認します。

```
# id ad_user@ad.example.com
uid=702801456(ad_user@ad.example.com) gid=732000006(ad_admins)
groups=732000006(ad_admins),702800513(domain users@ad.example.com)
```

38.3. ID ビューを使用して IDM クライアント上の AD ユーザーの DEFAULT TRUST VIEW の属性をオーバーライドする

Active Directory (AD) ユーザーのデフォルト信頼ビューから、いくつかの POSIX 属性をオーバーライドすることもできます。たとえば、ある特定の IdM クライアントで AD ユーザーに異なる GID を与える必要がある場合があります。ID ビューを使用して、AD ユーザーのデフォルト信頼ビューの値をオーバーライドし、単一のホストにそれを適用することができます。この手順では、**host1.idm.example.com** IdM クライアントの **ad_user@ad.example.com** AD ユーザーの GID を 732001337 に設定する方法を説明します。

前提条件

- **host1.idm.example.com** IdM クライアントへの root アクセス権限がある。
- 必要な権限を持つユーザー (例:**admin** ユーザー) としてログインしている。

手順

1. ID ビューを作成します。たとえば、**example_for_host1** という名前の ID ビューを作成するには、次のコマンドを実行します。

```
$ ipa idview-add example_for_host1
----------------------------
Added ID View "example_for_host1"
----------------------------
  ID View Name: example_for_host1
```

2. ユーザーの上書きを **example_for_host1** ID ビューに追加します。ユーザーの GID をオーバーライドするには、以下を実行します。
   - **ipa idoverrideuser-add** コマンドを入力します。
   - ID ビューの名前を追加します。
   - ユーザー名 (アンカーとも呼ばれます) を追加します。
   - **--gidnumber=** オプションを追加します。

```
$ ipa idoverrideuser-add example_for_host1 ad_user@ad.example.com --
gidnumber=732001337
----------------------------
Added User ID override "ad_user@ad.example.com"
----------------------------
  Anchor to override: ad_user@ad.example.com
  GID: 732001337
```

3. **example_for_host1** を **host1.idm.example.com** IdM クライアントに適用します。

```
$ ipa idview-apply example_for_host1 --hosts=host1.idm.example.com
----------------------------
Applied ID View "example_for_host1"
----------------------------
hosts: host1.idm.example.com
```

```
----------------------------------------------
Number of hosts the ID View was applied to: 1
----------------------------------------------
```

注記

ipa idview-apply コマンドでは、**--hostgroups** オプションも使用できます。このオプションは、ID ビューを、指定のホストグループに所属するホストに適用しますが、ホストグループ自体との関連付けは行いません。代わりに、**--hostgroups** オプションは指定されたホストグループのメンバーを拡張して、**--hosts** オプションを個別にすべて適用します。

つまり、今後ホストがホストグループに追加されても、ID ビューは新しいホストには適用されません。

4. **host1.idm.example.com** IdM クライアントの SSSD キャッシュから、**ad_user@ad.example.com** ユーザーのエントリーをクリアします。これにより、古いデータが削除され、新しいオーバーライド値が適用されるようになります。

```
[root@host1 ~]# sssctl cache-expire -u ad_user@ad.example.com
```

検証手順

1. ad_user@ad.example.com として、host1 に **SSH** で接続します。

```
[root@r8server ~]# ssh ad_user@ad.example.com@host1.idm.example.com
```

2. **ad_user@ad.example.com** ユーザーの情報を取得して、GID が更新された値を反映することを確認します。

```
[ad_user@ad.example.com@host1 ~]$ id ad_user@ad.example.com
uid=702801456(ad_user@ad.example.com) gid=732001337(admins2)
groups=732001337(admins2),702800513(domain users@ad.example.com)
```

38.4. IDM ホストグループへの ID ビューの適用

ipa idview-apply コマンドでは、**--hostgroups** オプションを使用できます。ただし、このオプションは、指定のホストに現在所属するホストに ID ビューを適用する 1 回限りの操作として機能しますが、ホストグループ自体と ID ビューを動的に関連付けることはありません。**--hostgroups** オプションは、指定したホストグループのメンバーを拡張して、**--hosts** オプションを個別にすべて適用します。

新しいホストを後でホストグループに追加する場合は、**--hosts** オプションで **ipa idview-apply** コマンドを使用して、新しいホストに ID ビューを手動で適用する必要があります。

同様に、ホストグループからホストを削除すると、ID ビューは削除後でも、ホストに割り当てられます。削除されたホストから ID ビューの適用を解除するには、**ipa idview-unapply id_view_name --hosts=name_of_the_removed_host** コマンドを実行する必要があります。

次の目標を達成するには、次の手順に従ってください。

1. ホストグループを作成し、そのグループにホストを追加する方法。
2. ID ビューをホストグループに適用する方法。

3. ホストグループに新規ホストを追加し、ID ビューを新しいホストに適用する方法。

前提条件

- ホストグループに適用する ID ビューが IdM に存在することを確認します。たとえば、AD ユーザーの GID をオーバーライドする ID ビューを作成するには、ID ビューを使用して IdM クライアント上の AD ユーザーの Default Trust View の属性をオーバーライドする を参照してください。

手順

1. ホストグループを作成し、そのグループにホストを追加します。

   a. ホストグループを作成します。たとえば、**baltimore** という名前のホストグループを作成するには、次のコマンドを実行します。

   ```
   [root@server ~]# ipa hostgroup-add --desc="Baltimore hosts" baltimore
   ----------------------------
   Added hostgroup "baltimore"
   ----------------------------
   Host-group: baltimore
   Description: Baltimore hosts
   ```

   b. ホストグループにホストを追加します。たとえば、**host102** および **host103** を **baltimore** ホストグループに追加するには、次のコマンドを実行します。

   ```
   [root@server ~]# ipa hostgroup-add-member --hosts={host102,host103} baltimore
   Host-group: baltimore
   Description: Baltimore hosts
   Member hosts: host102.idm.example.com, host103.idm.example.com
   -------------------------
   Number of members added 2
   -------------------------
   ```

2. ホストグループのホストに ID ビューを適用します。たとえば、**example_for_host1** ID ビューを **baltimore** ホストグループに適用するには、次のコマンドを実行します。

   ```
   [root@server ~]# ipa idview-apply --hostgroups=baltimore
   ID View Name: example_for_host1
   ----------------------------------------
   Applied ID View "example_for_host1"
   ----------------------------------------
     hosts: host102.idm.example.com, host103.idm.example.com
   ------------------------------------------
   Number of hosts the ID View was applied to: 2
   ------------------------------------------
   ```

3. ホストグループに新規ホストを追加し、ID ビューを新しいホストに適用します。

   a. ホストグループに新規ホストを追加します。たとえば、**somehost.idm.example.com** ホストを **baltimore** ホストグループに追加するには、次のコマンドを実行します。

   ```
   [root@server ~]# ipa hostgroup-add-member --hosts=somehost.idm.example.com
   baltimore
     Host-group: baltimore
   ```

```
  Description: Baltimore hosts
  Member hosts:  host102.idm.example.com,
host103.idm.example.com,somehost.idm.example.com
-------------------------
Number of members added 1
-------------------------
```

b. 必要に応じて、ID ビュー情報を表示します。たとえば、**example_for_host1** ID ビューの詳細を表示するには、次のコマンドを実行します。

```
[root@server ~]# ipa idview-show example_for_host1 --all
  dn: cn=example_for_host1,cn=views,cn=accounts,dc=idm,dc=example,dc=com
  ID View Name: example_for_host1
[...]
  Hosts the view applies to: host102.idm.example.com, host103.idm.example.com
  objectclass: ipaIDView, top, nsContainer
```

この出力では、ID ビューが **baltimore** ホストグループに新規追加されたホスト **somehost.idm.example.com** に適用されていないことが分かります。

c. ID ビューを新規ホストに適用します。たとえば、ID ビュー **example_for_host1** を **somehost.idm.example.com** に適用するには、次のコマンドを実行します。

```
[root@server ~]# ipa idview-apply --host=somehost.idm.example.com
ID View Name: example_for_host1
-----------------------------------------
Applied ID View "example_for_host1"
-----------------------------------------
  hosts: somehost.idm.example.com
-------------------------------------------------
Number of hosts the ID View was applied to: 1
-------------------------------------------------
```

検証手順

- ID ビュー情報を再度表示します。

```
[root@server ~]# ipa idview-show example_for_host1 --all
  dn: cn=example_for_host1,cn=views,cn=accounts,dc=idm,dc=example,dc=com
  ID View Name: example_for_host1
[...]
  Hosts the view applies to: host102.idm.example.com, host103.idm.example.com,
somehost.idm.example.com
  objectclass: ipaIDView, top, nsContainer
```

この出力は、ID ビューが **baltimore** ホストグループに新規追加されたホスト **somehost.idm.example.com** に適用されていることを示しています。

第39章 ID 範囲を手動で調整

IdM サーバーは、一意のユーザー ID (UID) とグループ ID (GID) の番号を生成します。異なる ID 範囲を作成し、レプリカに割り当てることで、同じ ID 番号が生成されないようにします。デフォルトでは、このプロセスは自動的に実行されます。ただし、IdM サーバーのインストール時に IdM ID 範囲を手動で調整したり、レプリカの DNA ID 範囲を手動で定義したりできます。

39.1. ID 範囲

ID 番号は **ID 範囲** に分類されます。個別のサーバーとレプリカに別々の数値範囲を指定することで、エントリーに対して発行された ID 番号が別のサーバーまたはレプリカの別のエントリーですでに使用されている可能性がなくなります。

ID 範囲には、以下の 2 つのタイプがあります。

- 最初のサーバーのインストール時に割り当てられる IdM **ID 範囲**。この範囲は作成後に変更できません。ただし、元の IdM ID 範囲に加えて、新しい IdM ID 範囲を作成できます。詳細は、自動 ID 範囲の割り当て および 新しい IdM ID 範囲の追加 を参照してください。

- ユーザーが変更できる**DNA (Distributed Numeric Assignment)** の ID 範囲。既存の IdM ID 範囲内に収まるようにする必要があります。詳細は、DNA ID 範囲の手動割り当て を参照してください。
  レプリカには、**次の** DNA ID 範囲を割り当てることもできます。レプリカは、現在の範囲で ID が不足すると、次の範囲を使用します。レプリカが削除された場合、次の範囲は自動的に割り当てられないため、 手動で割り当てる 必要があります。

範囲は、ドメインのバックエンド 389 Directory Server インスタンスの一部として、DNA プラグインによってサーバーとレプリカとの間で更新され、共有されます。

DNA 範囲の定義は、次の 2 つの属性によって設定されます。

- サーバーの次に使用可能な番号: DNA 範囲の下限値

- 範囲サイズ: DNA 範囲内の ID の数

初期の下限範囲は、プラグインインスタンスの設定時に設定されます。その後、プラグインは下限値を更新します。利用可能な数を範囲に分割すると、サーバーは、互いに重複せずに、継続的に数字を割り当てることができます。

39.2. 自動 ID 範囲の割り当て

IdM ID 範囲

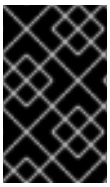
デフォルトでは、IdM サーバーのインストール時に IdM ID 範囲が自動的に割り当てられます。**Ipa-server-install** コマンドは、使用可能な合計 1 万の範囲から、20 万個の ID を無作為に選択して割り当てます。このようにランダムな範囲を選択すると、今後別の 2 つの IdM ドメインを統合する場合に、ID の競合が発生する可能性を大幅に削減できます。

注記

この IdM の ID 範囲は、作成後は修正できません。DNA (Distributed Numeric Assignment) の ID 範囲を手動で調整できるのは、DNA ID 範囲の手動割り当て で説明されているコマンドを使用する場合だけです。インストール時に、IdM ID 範囲に一致する DNA 範囲が自動的に作成されます。

DNA ID 範囲

IdM サーバー 1 台をインストールしている場合は、このサーバーが DNA ID 範囲全体を制御します。新規レプリカをインストールし、レプリカが独自の DNA ID 範囲を要求すると、サーバーの初期 ID 範囲が分割され、サーバーとレプリカの間で分散されます。レプリカは、初期サーバーで使用可能な DNA ID 範囲の残りの半分を受け取ります。次に、サーバーとレプリカは、新規ユーザーまたはグループのエントリーに元の ID 範囲に対応する部分を使用します。また、レプリカが割り当てられた ID 範囲を使い果たしそうになり、ID が 100 未満しか残っていない場合には、レプリカは他の使用可能なサーバーに接続して、新しい DNA ID 範囲を要求します。



重要

レプリカをインストールしても、即座に **ID** 範囲を受け取ることはありません。レプリカは、DNA プラグインの初回使用時 (ユーザーの初回追加時など) に ID 範囲を受け取ります。

レプリカが DNA ID 範囲を要求する前に最初のサーバーが機能しなくなると、レプリカはサーバーに問い合わせて ID 範囲を要求することができません。レプリカに新しいユーザーを追加しようとすると失敗します。このような場合は、無効になったサーバーに割り当てられている ID 範囲を確認 し、ID 範囲を手動でレプリカに割り当てる ことができます。

39.3. サーバーインストール時の IDM ID 範囲の手動割り当て

デフォルトの動作をオーバーライドし、無作為に割り当てる代わりに、IdM ID 範囲を手動で設定できます。

重要

UID の値が 1000 以下の ID 範囲は設定しないでください。1000 以下の値はシステム使用向けに予約されています。また、SSSD サービスは ID の値 0 を処理しないので、0 値が含まれる ID 範囲は設定しないでください。

手順

- **ipa-server-install** では以下の 2 つのオプションを使用することで、サーバーのインストール時に IdM ID 範囲を手動で定義できます。
  - **--idstart**: UID および GID 番号の開始値を指定します。
  - **--idmax**: UID および GID 番号の最大値を指定します。デフォルトでは、この値は **--idstart** の開始値に 199,999 を加えたものになります。

検証手順

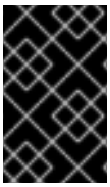
- ID 範囲が正しく割り当てられているかを確認するには、**ipa idrange-find** コマンドを使用して、割り当てられた IdM ID 範囲を表示します。

```
# ipa idrange-find
----------------
1 range matched
----------------
  Range name: IDM.EXAMPLE.COM_id_range
  First Posix ID of the range: 882200000
  Number of IDs in the range: 200000
```

```
  Range type: local domain range
----------------------------
Number of entries returned 1
----------------------------
```

39.4. 新しい IDM ID 範囲の追加

レプリカの ID が不足し、元の IdM ID 範囲を使い果たした場合など、場合によっては、元の IdM 範囲に加え、新しい IdM ID 範囲の作成が必要になる場合があります。



重要

新しい IdM ID 範囲を追加しても、新しい DNA ID 範囲は自動的に作成されません。必要に応じて、レプリカに新しい DNA ID 範囲を手動で割り当てる必要があります。割り当て方法の詳細は、DNA ID 範囲の手動割り当て を参照してください。

手順

1. 新しい IdM ID 範囲を作成するには、**ipa idrange-add** コマンドを使用します。新しい範囲名、範囲の最初の ID 番号、および範囲サイズを指定する必要があります。

```
# ipa idrange-add IDM.EXAMPLE.COM_new_range --base-id=1000000 --range-size=200000
----------------------------------------
Added ID range "IDM.EXAMPLE.COM_new_range"
----------------------------------------
  Range name: IDM.EXAMPLE.COM_new_range
  First Posix ID of the range: 1000000
  Number of IDs in the range: 200000
  Range type: local domain range
```

2. Directory Server を再起動します。

```
# systemctl restart dirsrv@IDM.EXAMPLE.COM.service
```

これにより、新しい範囲の UID を使用してユーザーを作成するときに、セキュリティー識別子 (SID) が割り当てられるようになります。

3. オプション: ID 範囲をすぐに更新します。

   a. System Security Services Daemon (SSSD) キャッシュをクリアします。

```
# sss_cache -E
```

   b. SSSD デーモンを再起動します。

```
# systemctl restart sssd
```

注記

SSSD キャッシュをクリアせずにサービスを再起動すると、SSSD は、IdM サーバーに保存されているドメインリストとその他の設定データを更新するときに、新しい ID 範囲のみを検出します。

検証手順

- **ipa idrange-find** コマンドを使用すると、新しい範囲が正しく設定されているかどうかを確認できます。

```
# ipa idrange-find
----------------
2 ranges matched
----------------
  Range name: IDM.EXAMPLE.COM_id_range
  First Posix ID of the range: 882200000
  Number of IDs in the range: 200000
  Range type: local domain range

  Range name: IDM.EXAMPLE.COM_new_range
  First Posix ID of the range: 1000000
  Number of IDs in the range: 200000
  Range type: local domain range
----------------------------
Number of entries returned 2
----------------------------
```

39.5. IDM ID 範囲におけるセキュリティーおよび相対識別子のロール

Identity Management (IdM) ID 範囲は、いくつかのパラメーターによって定義されます。

- 範囲名
- 範囲の最初の POSIX ID
- 範囲サイズ: 範囲内の ID の数
- 対応する **RID 範囲** の最初の **相対 ID** (RID)
- **セカンダリー RID 範囲** の最初の RID

これらの値は、**ipa idrange-show** コマンドを使用して表示できます。

```
$ ipa idrange-show IDM.EXAMPLE.COM_id_range
  Range name: IDM.EXAMPLE.COM_id_range
  First Posix ID of the range: 196600000
  Number of IDs in the range: 200000
  First RID of the corresponding RID range: 1000
  First RID of the secondary RID range: 1000000
  Range type: local domain range
```

セキュリティー識別子

ローカルドメインの ID 範囲からのデータは、IdM サーバーによって内部的に使用され、一意の **セキュリティー識別子** (SID) が IdM ユーザーおよびグループに割り当てられます。SID は、ユーザーオブジェクトとグループオブジェクトに格納されます。ユーザーの SID は、以下で構成されます。

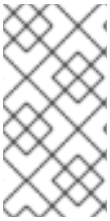
- ドメイン SID
- ドメイン SID に追加された 4 桁の 32 ビット値である、ユーザーの **相対識別子** (RID)

たとえば、ドメイン SID が S-1-5-21-123-456-789 で、このドメインのユーザーの RID が 1008 の場合、ユーザーの SID は SID of S-1-5-21-123-456-789-1008 になります。

相対識別子

RID 自体は、次の方法で計算されます。

ユーザーの POSIX UID から範囲の最初の POSIX ID を引き、対応する RID 範囲の最初の RID を結果に追加します。たとえば、**idmuser** の UID が 196600008、最初の POSIX ID が 196600000、そして最初の RID が 1000 の場合、**idmuser** の RID は 1008 になります。



注記

ユーザーの RID を計算するアルゴリズムは、対応する RID を計算する前に、特定の POSIX ID が割り当てられた ID 範囲内にあるかどうかをチェックします。たとえば、最初の ID が 196600000 で範囲サイズが 200000 の場合、1600000 の POSIX ID は ID 範囲外となるり、アルゴリズムはその RID を計算しません。

セカンダリー相対識別子

IdM では、POSIX UID は POSIX GID と同一にすることができます。これは、196600008 の UID を持つ **idmuser** がすでに存在する場合でも、196600008 の GID を持つ新しい **idmgroup** グループを作成できることを意味します。

ただし、SID で定義できるオブジェクトは、ユーザー **または** グループの 1 つだけです。**idmuser** 用にすでに作成されている S-1-5-21-123-456-789-1008 の SID は、**idmgroup** と共有することはできません。**idmgroup** の代替 SID を生成する必要があります。

IdM は、SID との競合を避けるために、**セカンダリー相対識別子** (セカンダリー RID) を使用します。このセカンダリー RID は、以下で構成されます。

- セカンダリー RID ベース
- 範囲サイズ (デフォルトではベース範囲サイズと同じ)。

上記の例では、セカンダリー RID ベースは 1000000 に設定されています。新しく作成された **idmgroup** の RID を計算するには、ユーザーの POSIX UID から範囲の最初の POSIX ID を引き、結果にセカンダリー RID 範囲の最初の RID を追加します。したがって、**idmgroup** には 1000008 の RID が割り当てられます。その結果、**idmgroup** の SID は S-1-5-21-123-456-789-1000008 になります。

ユーザーまたはグループオブジェクトが以前に手動で設定された POSIX ID で作成されている場合にのみ、IdM はセカンダリー RID を使用して SID を計算します。そうでない場合、自動割り当てにより、同じ ID が 2 回割り当てられることを防ぎます。

関連情報

- Ansible を使用して新規ローカル IdM ID 範囲を追加する方法

39.6. ANSIBLE を使用して新規ローカル IDM ID 範囲を追加する方法

レプリカの ID が不足し、元の IdM ID 範囲を使い果たした場合など、場合によっては、元の IdM 範囲に加え、新しい Identity Management (IdM) ID 範囲の作成が必要になる場合があります。以下の例は、Ansible Playbook を使用して新しい IdM ID 範囲を作成する方法を説明しています。

注記

新しい IdM ID 範囲を追加しても、新しい DNA ID 範囲は自動的に作成されません。必要に応じて、新しい DNA ID 範囲を手動で割り当てる必要があります。割り当て方法の詳細は、DNA ID 範囲の手動割り当て を参照してください。

前提条件

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

手順

1. **~/MyPlaybooks/** ディレクトリーに移動します。

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

2. 次の内容で **idrange-present.yml** playbook を作成します。

```
---
- name: Playbook to manage idrange
  hosts: ipaserver
  become: no

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure local idrange is present
    ipaidrange:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: new_id_range
      base_id: 12000000
      range_size: 200000
      rid_base: 1000000
      secondary_rid_base: 200000000
```

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

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

5. **ipaserver** に **SSH** 接続し、Directory Server を再起動します。

```
# systemctl restart dirsrv@IDM.EXAMPLE.COM.service
```

これにより、新しい範囲の UID を使用してユーザーを作成するときに、セキュリティー識別子 (SID) が割り当てられるようになります。

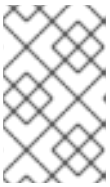
6. オプション: ID 範囲をすぐに更新します。

a. **ipaserver** で、System Security Services Daemon (SSSD) キャッシュをクリアします。

```
# sss_cache -E
```

b. **ipaserver** で SSSD デーモンを再起動します。

```
# systemctl restart sssd
```



注記

SSSD キャッシュをクリアせずにサービスを再起動すると、SSSD は、IdM サーバーに保存されているドメインリストとその他の設定データを更新するときに、新しい ID 範囲のみを検出します。

検証手順

- **ipa idrange-find** コマンドを使用すると、新しい範囲が正しく設定されているかどうかを確認できます。

```
# ipa idrange-find
----------------
2 ranges matched
----------------
  Range name: IDM.EXAMPLE.COM_id_range
  First Posix ID of the range: 882200000
  Number of IDs in the range: 200000
  Range type: local domain range

  Range name: IDM.EXAMPLE.COM_new_id_range
  First Posix ID of the range: 12000000
  Number of IDs in the range: 200000
  Range type: local domain range
----------------------------
Number of entries returned 2
----------------------------
```

関連情報

- IdM ID 範囲におけるセキュリティーおよび相対識別子のロール

39.7. AD への信頼を削除した後の ID 範囲の削除

IdM 環境と Active Directory (AD) 環境間の信頼を削除している場合は、それに関連付けられている ID 範囲を削除することを推奨します。

警告

信頼できるドメインに関連付けられた ID 範囲に割り当てられた ID は、IdM に登録されているシステムのファイルおよびディレクトリーの所有権に引き続き使用される可能性があります。

削除した AD 信頼に対応する ID 範囲を削除すると、AD ユーザーが所有するファイルおよびディレクトリーの所有権を解決できなくなります。

前提条件

- AD 環境への信頼を削除している。

手順

1. 現在使用されている ID 範囲をすべて表示します。

```
[root@server ~]# ipa idrange-find
```

2. 削除した信頼に関連付けられた ID 範囲の名前を識別します。ID 範囲の名前の最初の部分は、信頼の名前 (**AD.EXAMPLE.COM_id_range** など) になります。

3. 範囲を削除します。

```
[root@server ~]# ipa idrange-del AD.EXAMPLE.COM_id_range
```

4. SSSD サービスを再起動して、削除した ID 範囲への参照を削除します。

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

関連情報

- コマンドラインを使用した信頼の削除 を参照してください。
- IdM Web UI を使用した信頼の削除 を参照してください。

39.8. 現在割り当てられている DNA ID 範囲の表示

サーバーで現在アクティブな DNA (Distributed Numeric Assignment) の ID 範囲と、次の DNA 範囲が割り当てられている場合にはその範囲の両方を表示できます。

手順

- トポロジー内のサーバーに設定されている DNA ID 範囲を表示するには、以下のコマンドを使用します。
  - **ipa-replica-manage dnarange-show** は、全サーバー (サーバーを指定した場合は指定されたサーバーでのみ) に設定されている現在の DNA ID 範囲を表示します。以下に例を示します。

```
# ipa-replica-manage dnarange-show
serverA.example.com: 1001-1500
serverB.example.com: 1501-2000
serverC.example.com: No range set

# ipa-replica-manage dnarange-show serverA.example.com
serverA.example.com: 1001-1500
```

  - **ipa-replica-manage dnanextrange-show** は、全サーバーに現在設定されている次の DNA ID 範囲を表示します。サーバーを指定した場合は、指定したサーバー上でのみ表示されます。以下に例を示します。

```
# ipa-replica-manage dnanextrange-show
serverA.example.com: 2001-2500
serverB.example.com: No on-deck range set
serverC.example.com: No on-deck range set

# ipa-replica-manage dnanextrange-show serverA.example.com
serverA.example.com: 2001-2500
```

39.9. 手動による ID 範囲の割り当て

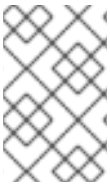
特定の状況では、DNA (Distributed Numeric Assignment) の ID 範囲を手動で割り当てする必要があります。たとえば、以下のような場合です。

- レプリカの ID がなくなり、IdM ID 範囲がすべて使われている。
  レプリカに割り当てられた DNA ID 範囲を使い果たし、IdM 範囲で使用可能な空き ID がなくなったため、ID の追加要求に失敗した場合。

  この状況を解決するには、レプリカに割り当てられた DNA ID 範囲を拡張します。これは、以下の 2 つの方法で実行できます。
  - 別のレプリカに割り当てられる DNA ID 範囲を短くし、新たに利用可能な値を、ID 範囲を使い果たしたレプリカに割り当てます。
  - 新しい IdM ID 範囲を作成し、この作成した IdM 範囲内でレプリカに新しい DNA ID 範囲を設定します。
    新しい IdM ID 範囲を作成する方法は 新しい IdM ID 範囲の追加 を参照してください。
- レプリカが機能しなくなる
  レプリカが停止して削除する必要がある場合、レプリカの DNA ID 範囲は自動的に取得されません。つまり、以前にレプリカに割り当てられていた DNA ID 範囲は使用できなくなります。DNA ID 範囲を復元し、他のレプリカで使用できるようにします。

  これを行うには、その範囲を別のサーバーに手動で割り当てる前に、ID 範囲の値を調べます。また、UID や GID が重複しないように、回復した範囲からの ID の値がユーザーまたはグループに割り当てられていないことを確認します。これは、既存のユーザーおよびグループの UID と GID を調べて実行できます。

DNA ID 範囲の手動割り当て のコマンドを使用して、レプリカに DNA ID 範囲を手動で割り当てることができます。



注記

新しい DNA ID 範囲を割り当てると、サーバーまたはレプリカ上の既存のエントリーの UID は同じになります。現在の DNA ID 範囲を変更しても、IdM は過去に割り当てられた範囲の記録を保持するため、これにより問題が発生することはありません。

39.10. DNA ID 範囲の手動割り当て

場合によっては、機能していないレプリカに割り当てられた DNA ID 範囲を再割り当てするなど、既存レプリカに DNA (Distributed Numeric Assignment) の ID 範囲を手動で割り当てないといけない場合があります。詳細は、手動による ID 範囲の割り当て を参照してください。

DNA ID 範囲を手動で調整する場合は、新たに調整した範囲が IdM ID 範囲に含まれていることを確認してください。これは、**ipa idrange-find** コマンドを使用して確認できます。そうでない場合、コマンドは失敗します。

重要

ID 範囲を重複しないように注意してください。サーバーまたはレプリカに割り当てた ID 範囲のいずれかが重複すると、この 2 つのサーバーにより、異なるエントリーに同じ ID 値を割り当てる可能性があります。

前提条件

- **オプション。**機能していないレプリカから DNA ID 範囲を復元する場合は、最初に 現在割り当てられている DNA ID 範囲の表示 に記載のコマンドを使用して ID 範囲を見つけます。

手順

- 指定のサーバーの現在の DNA ID 範囲を定義するには、**ipa-replica-manage dnarange-set** を使用します。

```
# ipa-replica-manage dnarange-set serverA.example.com 1250-1499
```

- 指定のサーバーの次の DNA ID 範囲を定義するには、**ipa-replica-manage dnanextrange-set** を使用します。

```
# ipa-replica-manage dnanextrange-set serverB.example.com 1500-5000
```

検証手順

- 現在割り当てられている DNA ID 範囲の表示 で説明されているコマンドを使用して、新しい DNA 範囲が正しく設定されているかどうかを確認できます。

第40章 SUBID 範囲の手動管理

コンテナー化環境では、IdM ユーザーが subID 範囲を手動で割り当てる必要がある場合があります。次の手順では、subID 範囲を管理する方法について説明します。

40.1. IDM CLI を使用した SUBID 範囲の生成

Identity Management (IdM) の管理者は、subID 範囲を生成し、それを IdM ユーザーに割り当てることができます。

前提条件

- IdM ユーザーが存在する。
- IdM **admin** Ticket-Granting Ticket (TGT) を取得した。詳細は、kinit による IdM への手動ログイン を参照してください。
- この手順を実行する IdM ホストへの **root** アクセス権がある。

手順

1. [オプション] 既存の subID 範囲を確認します。

```
# ipa subid-find
```

2. subID 範囲が存在しない場合は、次のいずれかの方法を選択します。
   - 1 つの subID 範囲を生成し、1 つの IdM ユーザーに割り当てます。

```
# ipa subid-generate --owner=idmuser

Added subordinate id "359dfcef-6b76-4911-bd37-bb5b66b8c418"

  Unique ID: 359dfcef-6b76-4911-bd37-bb5b66b8c418
  Description: auto-assigned subid
  Owner: idmuser
  SubUID range start: 2147483648
  SubUID range size: 65536
  SubGID range start: 2147483648
  SubGID range size: 65536
```

   - 複数の subID 範囲を生成し、すべての IdM ユーザーに割り当てます。

```
# /usr/libexec/ipa/ipa-subids --all-users

Found 2 user(s) without subordinate ids
  Processing user 'user4' (1/2)
  Processing user 'user5' (2/2)
Updated 2 user(s)
The ipa-subids command was successful
```

3. [オプション] デフォルトで新しい IdM ユーザーに subID 範囲を割り当てます。

```
# ipa config-mod --user-default-subid=True
```

-

検証

- ユーザーに subID 範囲が割り当てられていることを確認します。

```
# ipa subid-find --owner=idmuser

1 subordinate id matched

  Unique ID: 359dfcef-6b76-4911-bd37-bb5b66b8c418
  Owner: idmuser
  SubUID range start: 2147483648
  SubUID range size: 65536
  SubGID range start: 2147483648
  SubGID range size: 65536

Number of entries returned 1
```

40.2. IDM WEBUI インターフェイスを使用した SUBID 範囲の生成

Identity Management (IdM) 管理者は、subID 範囲を生成し、IdM WebUI インターフェイスでユーザーに割り当てることができます。

前提条件

- IdM ユーザーが存在する。
- IdM **admin** Kerberos チケット (TGT) を取得した。詳細は、Web UI で IdM にログイン: Kerberos チケットの使用 を参照してください。
- この手順を実行する IdM ホストへの **root** アクセス権がある。

手順

1. IdM WebUI インターフェイスで、**Subordinate IDs** タブを展開し、**Subordinate IDs** オプションを選択します。
2. **Subordinate IDs** インターフェイスが表示されたら、インターフェイスの右上隅にある **Add** ボタンをクリックします。**Add subid** ウィンドウが表示されます。
3. **Add subid** ウィンドウで、所有者つまり subID 範囲の割り当て先のユーザーを選択します。
4. **Add** ボタンをクリックします。

検証

- **Subordinate IDs** タブの下のテーブルを確認します。新しいレコードがテーブルに表示されます。所有者は、subID 範囲を割り当てたユーザーです。

40.3. IDM CLI を使用した IDM ユーザーの SUBID 情報の表示

Identity Management (IdM) ユーザーは、IdM ユーザーの subID 範囲を検索し、関連情報を表示できます。

前提条件

- IdM クライアントで subID 範囲を設定した。
- IdM ユーザーの Ticket-Granting Ticket (TGT) を取得した。詳細は、kinit による IdM への手動ログイン を参照してください。

手順

- subID 範囲の詳細を表示するには、以下を実行します。
  - 範囲の所有者である Identity Management (IdM) ユーザーの一意の ID ハッシュがわかっている場合は、以下を実行します。

```
$ ipa subid-show 359dfcef-6b76-4911-bd37-bb5b66b8c418

  Unique ID: 359dfcef-6b76-4911-bd37-bb5b66b8c418
  Owner: idmuser
  SubUID range start: 2147483648
  SubUID range size: 65536
  SubGID range start: 2147483648
  SubGID range size: 65536
```

  - その範囲内の特定の subID がわかっている場合は、以下を実行します。

```
$ ipa subid-match --subuid=2147483670

1 subordinate id matched

  Unique ID: 359dfcef-6b76-4911-bd37-bb5b66b8c418
  Owner: uid=idmuser
  SubUID range start: 2147483648
  SubUID range size: 65536
  SubGID range start: 2147483648
  SubGID range size: 65536

Number of entries returned 1
```

40.4. GETSUBID コマンドを使用して SUBID 範囲をリスト表示する

システム管理者は、コマンドラインインターフェイスを使用して、Identity Management (IdM) またはローカルユーザーの subID 範囲をリスト表示できます。

前提条件

- **idmuser** ユーザーが IdM に存在する。
- **shadow-utils-subid** パッケージがインストールされている。
- **/etc/nsswitch.conf** ファイルを編集できる。

手順

1. **/etc/nsswitch.conf** ファイルを開き、**subid** 変数を **sss** 値に設定して、**shadow-utils** ユーティリティーが IdM subID 範囲を使用するように設定します。

```
[...]
subid: sss
```

注記

subid フィールドには 1 つの値のみを指定できます。**subid** フィールドを **sss** ではなく **file** 値に設定するか、値なしに設定すると、**shadow-utils** ユーティリティーが **/etc/subuid** ファイルと **/etc/subgid** ファイルの subID 範囲を使用するように設定されます。

2. IdM ユーザーの subID 範囲をリスト表示します。

```
$ getsubids idmuser
0: idmuser 2147483648 65536
```

最初の値 2147483648 は、subID 範囲の開始位置を示します。2 番目の値 65536 は、範囲のサイズを示します。

第41章 IDM CLI でのホストの管理

本章では、Identity Management (IdM) の ホスト および ホストエントリー と、IdM CLI でホストとホストエントリーを管理する際に以下の操作が実行されます。

- ホストの登録
- IdM ホストエントリーの追加
- IdM ホストエントリーの削除
- ホストの再登録
- ホストの名前変更
- ホストの無効化
- ホストの再有効化

本章には、前提条件、コンテキスト、および操作の結果の 概要表 も含まれています。

41.1. IDM のホスト

Identity Management (IdM) は、以下の ID を管理します。

- ユーザー
- サービス
- ホスト

ホストはマシンを表します。ホストには、IdM LDAP に IdM ID となるエントリーがあります。これは IdM サーバーの 389 Directory Server のインスタンスです。

IdM LDAP のホストエントリーは、その他のホストとドメイン内のサービスとの関係を確立するために使用されます。この関係では、ドメイン内ホストの認可および制御の **委譲** が不可欠な要素です。ホストは、**ホストベースのアクセス制御** (HBAC) ルールで使用できます。

IdM ドメインは、共通の ID 情報、共通ポリシー、および共有サービスを使用して、マシン間で共通性を確立します。ドメインのクライアントとしてのドメイン機能に属するマシンです。これは、ドメインが提供するサービスを使用することを意味します。IdM ドメインは、マシン専用の 3 つの主なサービスを提供します。

- DNS
- Kerberos
- 証明書の管理

IdM のホストは、そのホストで実行しているサービスと密接に接続されています。

- サービスエントリーは、ホストに関連付けられています。
- ホストには、ホストとサービスの両方の Kerberos プリンシパルが格納されます。

41.2. ホスト登録

本セクションでは、ホストを IdM クライアントとして登録し、登録中および登録後に何が起こるかを説明します。本セクションでは、IdM ホストの登録と、IdM ユーザーの登録を比較します。また、本セクションでは、ホストで利用可能な代替タイプの認証も概説します。

ホストの登録は、以下の要素で構成されます。

- IdM LDAP でのホストエントリーの作成: 場合によっては、IdM CLI で **ipa host-add** コマンドを使用するか、同等の IdM Web UI 操作 を使用します。
- ホストで IdM サービス (System Security Services Daemon (SSSD)、Kerberos、certmonger など) を設定し、ホストを IdM ドメインに参加させる。

2 つのアクションは、個別に実行することも一緒に実行することもできます。

個別に実行すると、異なるレベルの特権を持つ 2 人のユーザー間で 2 つのタスクを分割できます。これは、一括デプロイメントに役立ちます。

ipa-client-install コマンドは、2 つのアクションを一緒に実行できます。コマンドは、そのエントリーがまだ存在していない場合は IdM LDAP にホストエントリーを作成し、そのホストに Kerberos サービスと SSSD サービスの両方を設定します。このコマンドにより、ホストが IdM ドメイン内に移動し、接続先の IdM サーバーを識別できるようになります。IdM が管理する DNS ゾーンにホストが属する場合は、**ipa-client-install** でホストに DNS レコードも追加します。コマンドはクライアントで実行する必要があります。

41.3. ホストの登録に必要なユーザー権限

ホスト登録操作では、権限のないユーザーが不要なマシンを IdM ドメインに追加しないように、認証が必要になります。必要な特権は、次のようないくつかの要因によって異なります。

- ホストエントリーが **ipa-client-install** の実行とは別に作成される場合
- ワンタイムパスワード (OTP) が登録に使用される場合

必要に応じて IdM LDAP にホストエントリーを手動で作成するためのユーザー特権

CLI コマンド **ipa host-add** または IdM Web UI を使用して、IdM LDAP にホストエントリーを作成するのに必要なユーザー特権は **ホストの管理者** です。**ホスト管理者** の特権は、**IT スペシャリスト** ロールから取得できます。

クライアントを IdM ドメインに参加させるためのユーザー特権

ホストは、**ipa-client-install** コマンドの実行時に IdM クライアントとして設定されます。**ipa-client-install** コマンドの実行に必要な認証情報のレベルは、以下のような登録シナリオのどれに該当するかによって異なります。

- IdM LDAP のホストエントリーが存在しません。このシナリオでは、管理者の完全な認証情報または **ホスト管理者** ロールが必要です。完全な管理者とは **admins** グループのメンバーです。**ホスト管理者** ロールは、ホストの追加およびホストの登録の特権を提供します。このシナリオの詳細は ユーザー認証情報を使用したクライアントのインストール: 対話的なインストール を参照してください。
- IdM LDAP のホストエントリーが存在します。このシナリオでは、**ipa-client-install** を正常に実行するには、制限された管理者の認証情報が必要です。この場合、制限されている管理者には、**ホストの登録** 特権を提供する **登録管理者** ロールがあります。詳細は ユーザー認証情報を使用したクライアントのインストール: 対話的なインストール を参照してください。
- IdM LDAP にホストエントリーが存在し、完全または限定された管理者により、ホストの OTP

が生成されました。このシナリオでは、正しい OTP を指定して **--password** オプションを指定して **ipa-client-install** コマンドを実行すると、通常のユーザーとして IdM クライアントをインストールできます。詳細は ワンタイムパスワードでクライアントのインストール: 対話的なインストール を参照してください。

登録後、IdM ホストは、IdM リソースにアクセスできるように、新しいセッションをすべて認証します。IdM サーバーがマシンを信頼し、そのマシンにインストールされているクライアントソフトウェアからの IdM 接続を受け入れるには、マシン認証が必要です。クライアントを認証すると、IdM サーバーはそのリクエストに応答できます。

41.4. IDM ホストとユーザーの登録と認証: 比較

IdM のユーザーとホストの間には多くの類似点があります。この類似点には、登録ステージで見られるものと、デプロイメントステージでの認証に関係するものがあります。

- 登録段階 (ユーザーおよびホストの登録):
  - 管理者は、ユーザーまたはホストが実際に IdM に参加する前に、ユーザーとホストの LDAP エントリーを作成できます。コマンドは、ステージユーザーの場合は **ipa stageuser-add** で、ホストの場合は **ipa host-add** です。
  - ホストで **ipa-client-install** コマンドを実行すると、**キーテーブル** (または略してキータブ)、(ある程度ユーザーパスワードに類似する) 対称キーを含むファイルが作成され、ホストが IdM レルムに参加します。同様に、ユーザーはアカウントをアクティブ化するときにパスワードを作成するように求められ、IdM レルムに参加します。
  - ユーザーパスワードは、ユーザーのデフォルトの認証方法ですが、キータブはホストのデフォルトの認証方法です。キータブは、ホストのファイルに保存されます。

表41.1 ユーザーおよびホストの登録

アクション	ユーザー	ホスト
登録前	$ ipa stageuser-add user_name [--password]	$ ipa host-add host_name [--random]
アカウントのアクティブ化	$ ipa stageuser-activate user_name	$ ipa-client install [--password] (ホスト自体で実行する必要があります)

- デプロイメント段階 (ユーザーおよびホストセッションの認証):
  - ユーザーが新しいセッションを開始すると、ユーザーはパスワードを使用して認証を行います。同様に、切り替え時に、ホストがそのキータブファイルを提示して認証を行います。SSSD (System Security Services Daemon) は、このプロセスをバックグラウンドで管理します。
  - 認証が成功すると、ユーザーまたはホストは、Kerberos チケット発行許諾チケット (TGT) を取得します。
  - 次に、TGT を使用して、特定のサービスの特定のチケットを取得します。

表41.2 ユーザーおよびホストセッションの認証

	ユーザー	ホスト
認証のデフォルト手段	**パスワード**	**キータブ**
セッションの開始 (通常のユーザー)	$ **kinit user_name**	**[ホストへの切り替え]**
認証成功の結果	**TGT** は、特定サービスへのアクセスの取得に使用されます。	**TGT** は、特定サービスへのアクセスの取得に使用されます。

TGT およびその他の Kerberos チケットは、サーバーにより定義された Kerberos サービスおよびポリシーの一部として生成されます。Kerberos チケットの最初の付与、Kerberos 認証情報の更新、および Kerberos セッションの破棄もすべて IdM サービスにより自動的に処理されます。

IdM ホストの代替認証オプション

キータブとは別に、IdM は、その他の 2 つのタイプのマシン認証にも対応しています。

- SSH 鍵。ホストの SSH 公開キーが作成され、ホストエントリーにアップロードされます。そこから、SSSD (System Security Services Daemon) は IdM を ID プロバイダーとして使用し、OpenSSH およびその他のサービスと一緒に機能して、IdM の中央にある公開鍵を参照できます。
- 機械の証明書。この場合、マシンは IdM サーバーの認証局により発行され、IdM の Directory Server に保存されている SSL 証明書を使用します。次に、証明書はマシンに送信され、サーバーに対する認証時に提示されます。クライアントでは、証明書は certmonger というサービスにより管理されます。

41.5. ホスト操作

次のセクションでは、ホストの登録と有効化に関連する最も一般的な操作、前提条件、コンテキスト、およびこれらの操作を実行した結果について説明します。

表41.3 ホスト操作パート 1

アクション	アクションの前提条件	コマンド実行が妥当な時期	システム管理者によりアクションの実行方法と実行するコマンド
クライアントの登録	詳細は、**Identity Management のインストール** の Identity Management クライアントをインストールするためのシステムの準備 を参照してください。	ホストが IdM レルムに参加する時	IdM ドメインでマシンをクライアントとして登録する場合は、2 つのプロセスがあります。**ipa host-add** コマンドを実行すると、クライアントにホストエントリーが作成されて (389 Directory Server インスタンスに格納されます) から、クライアントをプロビジョニングするキータブが作成されます。プロセスは、いずれも **ipa-client-install** コマンドで自動的に実行されます。この手順は個別に実行することもできます。これにより、管理者は、クライアントを実際に設定する前にマシンと IdM を準備できます。これにより、一括デプロイメントなど、より柔軟な設定シナリオが可能になります。

アクション	アクションの前提条件	コマンド実行が妥当な時期	システム管理者によりアクションの実行方法と実行するコマンド
クライアントの無効化	ホストに、IdM のエントリーが必要です。ホストにはアクティブなキータブが必要です。	(メンテナンス目的などで) IdM レルムからホストを一時的に削除する時	**ipa host-disable host_name**
クライアントの有効化	ホストに、IdM のエントリーが必要です。	一時的に無効にしたホストが再びアクティブになるようにする時	**ipa-getkeytab**
クライアントの再登録	ホストに IdM へのエントリーが必要です。	元のホストが失われていて、同じホスト名でホストがインストールされている時	**ipa-client-install --keytab** または **ipa-client-install --force-join**
クライアントの登録解除	ホストに、IdM のエントリーが必要です。	IdM レルムからホストを完全に削除する時	**ipa-client-install --uninstall**

表41.4 ホスト操作パート 2

アクション	管理者はコマンドを実行するマシン	アクションの実行時に発生する動作、ホストが IdM で機能する場合の結果、および導入または削除される制限
クライアントの登録	2 ステップでの登録の場合 - **ipa host-add** は、任意の IdM クライアントで実行できます。**ipa-client-install** の次のステップは、クライアント自体で実行する必要があります。	デフォルトでは、SSSD が認証と認可のために IdM サーバーに接続するように設定します。必要に応じて、PAM (Pluggable Authentication Module) と NSS (Name Switching Service) を、Kerberos および LDAP を介して IdM サーバーと連携するように設定することができます。
クライアントの無効化	IdM の任意のマシン (ホスト自体も含む)。	ホストの Kerberos 鍵および SSL 証明書が無効にされており、ホストで実行しているすべてのサービスが無効になります。

アクション	管理者はコマンドを実行するマシン	アクションの実行時に発生する動作、ホストが IdM で機能する場合の結果、および導入または削除される制限
クライアントの有効化	IdM のマシン。無効なホストで実行する場合は、LDAP 認証情報を提供する必要があります。	ホストの Kerberos 鍵と SSL 証明書が再び有効になり、ホストで実行しているすべての IdM サービスが再度有効になります。
クライアントの再登録	再登録するホスト。LDAP 認証情報を提供する必要があります。	ホスト用に新しい Kerberos キーが生成され、以前のキーが置き換えられます。
クライアントの登録解除	登録解除するホスト。	このコマンドは IdM の設定を解除し、マシンを以前の状態に戻そうとします。このプロセスには、IdM サーバーからホストの登録を解除するステップが含まれます。登録解除には、IdM サーバーでプリンシパルキーを無効にすることで設定されます。**/etc/krb5.keytab** (**host/<fqdn>@REALM**) のマシンプリンシパルは、登録解除する IdM サーバーに認証するのに使用されます。このプリンシパルが存在しない場合は登録解除に失敗します。管理者はホストプリンシパル (**ipa host-disable <fqdn>**) を無効にする必要があります。

41.6. IDM LDAP のホストエントリー

Identity Management (IdM) ホストエントリーには、ホストに関する情報とホストに含めることができる属性が含まれています。

LDAP ホストエントリーは、IdM 内のクライアントに関するすべての関連情報が含まれます。

- ホストに関連付けられたサービスエントリー
- ホストおよびサービスプリンシパル
- アクセス制御ルール
- 物理的な場所やオペレーティングシステムなどのマシン情報

注記

IdM Web UI の **Identity → Hosts** タブには、IdM LDAP に保存されている特定のホストに関する情報がすべて表示されないことに注意してください。

ホストエントリー設定プロパティー

ホストエントリーには、ホストに関する情報 (物理的な場所、MAC アドレス、鍵、証明書など、システム設定を除く) を含めることができます。

この情報は、ホストエントリーが手動で作成された場合に、作成時に設定できます。また、この情報のほとんどは、ホストがそのドメインに登録してからホストエントリーに追加できます。

表41.5 ホスト設定プロパティー

UI フィールド	コマンドラインオプション	説明
説明	**--desc**=description	ホストの説明。
局所性	**--locality**=locality	ホストの地理的な場所。
場所	**--location**=location	データセンターのラックなど、ホストの物理的な場所。
プラットフォーム	**--platform**=string	ホストのハードウェアまたはアーキテクチャー。
オペレーティングシステム	**--os**=string	ホストのオペレーティングシステムとバージョン。
MAC アドレス	**--macaddress**=address	ホストの MAC アドレス。これは多値属性です。MAC アドレスは、NIS プラグインにより、ホスト用の NIS の **ethers** マップを作成するために使用されます。
SSH 公開鍵	**--sshpubkey**=string	ホストの完全 SSH 公開鍵。これは複数値の属性であるため、複数の鍵を設定できます。
プリンシパル名 (編集不可)	**--principalname**=principal	ホストの Kerberos プリンシパル名。**-p** に別のプリンシパルを明示的に設定しない限り、クライアントのインストール時にホスト名がデフォルトになります。これはコマンドラインツールを使用して変更できますが、UI で変更することはできません。
ワンタイムパスワードの設定	**--password**=string	このオプションは、一括登録で使用できるホストのパスワードを設定します。
-	**--random**	このオプションは、一括登録で使用されるランダムパスワードを生成します。
-	**--certificate**=string	ホストの証明書ブロブ。
-	**--updatedns**	これにより、IP アドレスが変更した場合に、ホストが DNS エントリーを動的に更新できるかどうかが設定されます。

41.7. IDM CLI で IDM ホストエントリーの追加

コマンドラインインターフェイス (CLI) を使用して Identity Management (IdM) にホストエントリーを追加するには、次の手順に従います。

ホストエントリーは、**host-add** コマンドを使用して作成されます。このコマンドは、ホストエントリーを IdM Directory Server に追加します。CLI で **ipa help host** を入力し、**ipa host** の man ページで、**host-add** で利用可能なオプションの完全リストを取得します。

ホストを IdM に追加する場合は、いくつかのシナリオがあります。

- 最も基本的な方法として、クライアントホスト名を指定してクライアントを Kerberos レルムに追加し、IdM LDAP サーバーにエントリーを作成します。

```
$ ipa host-add client1.example.com
```

- DNS を管理するために IdM サーバーを設定している場合は、**--ip-address** オプションを使用して DNS リソースレコードにホストを追加します。

例41.1 静的 IP アドレスを持つホストエントリーの作成

```
$ ipa host-add --ip-address=192.168.166.31 client1.example.com
```

- 追加するホストに静的 IP アドレスがない場合や、クライアントの設定時に IP アドレスが不明な場合は、**ipa host-add** コマンドで **--force** オプションを使用します。

例41.2 DHCP でホストエントリーの作成

```
$ ipa host-add --force client1.example.com
```

たとえば、ラップトップは IdM クライアントとして事前設定されている場合がありますが、設定時には IP アドレスがありません。**--force** を使用すると、基本的に IdM DNS サービスにプレースホルダーエントリーが作成されます。DNS サービスがレコードを動的に更新すると、ホストの現在の IP アドレスが検出され、DNS レコードが更新されます。

41.8. IDM CLI でホストエントリーの削除

- **host-del** コマンドを使用して、ホストレコードを削除します。IdM ドメインに DNS が統合されている場合は、**--updatedns** オプションを使用して、あらゆる種類のホストに関連するレコードを DNS から削除します。

```
$ ipa host-del --updatedns client1.example.com
```

41.9. IDENTITY MANAGEMENT クライアントの再登録

本セクションでは、Identity Management クライアントを再登録するさまざまな方法を説明します。

41.9.1. IdM におけるクライアントの再登録

再登録の間、クライアントは新しい鍵 (Kerberos および SSH) を生成しますが、LDAP データベースのクライアントのアイデンティティーは変更されません。再登録後、ホストは、IdM サーバーとの接続を失う前と同じ **FQDN** を持つ同じ LDAP オブジェクトに、キーとその他の情報を保持します。

重要

ドメインエントリーがアクティブなクライアントのみを再登録できます。クライアントをアンインストール (**ipa-client-install --uninstall** を使用) した場合や、ホストエントリーを無効 (**ipa host-disable** を使用) にした場合は再登録できません。

クライアントの名前を変更すると、再登録することができません。これは、Identity Management では LDAP にあるクライアントのエントリーのキー属性はクライアントのホスト名 **FQDN** であるためです。クライアントの再登録中はクライアントの LDAP オブジェクトは変更されませんが、クライアントの名前を変更すると、クライアントの鍵とその他の情報は新しい **FQDN** を持つ異なる LDAP オブジェクトに格納されます。そのため、IdM からホストをアンインストールし、ホストのホスト名を変更して、新しい名前で IdM クライアントとしてインストールするのが、クライアントの名前を変更する唯一の方法です。クライアント名を変更する方法は、Identity Management クライアントシステムの名前の変更 を参照してください。

クライアント再登録中に行われること

Identity Management は再登録中に以下を行います。

- 元のホスト証明書を破棄する。
- 新規の SSH 鍵を作成する。
- 新規のキータブを生成する。

41.9.2. ユーザー認証情報でクライアントの再登録: 対話的な再登録

許可されたユーザーの認証情報を使用して、Identity Management クライアントを対話的に再登録するには、次の手順に従います。

1. 同じホスト名のクライアントマシンを再作成します。
2. クライアントマシンで **ipa-client-install --force-join** コマンドを実行します。

```
# ipa-client-install --force-join
```

3. スクリプトにより、アイデンティティーがクライアントの再登録に使用されるユーザーの入力が求められます。たとえば、登録管理者 (Enrollment Administrator) ロールを持つ **hostadmin** ユーザーなどが該当します。

```
User authorized to enroll computers: hostadmin
Password for hostadmin@EXAMPLE.COM:
```

関連情報

- **Installing Identity Management** の Installing a client by using user credentials: Interactive installation を参照してください。

41.9.3. クライアントのキータブでクライアントの再登録: 非対話的な再登録

前提条件

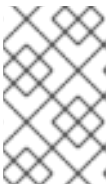
- **/tmp** や **/root** などのディレクトリーに元のクライアントキータブファイルをバックアップします。

手順

以下の手順に従って、クライアントシステムのキータブを使用して、Identity Management (IdM) クライアントを非対話的に再登録します。たとえば、クライアントのキータブを使用した再登録は自動インストールに適しています。

1. 同じホスト名のクライアントマシンを再作成します。
2. バックアップした場所から、再作成したクライアントマシンの **/etc/** ディレクトリーにキータブファイルをコピーします。
3. **ipa-client-install** ユーティリティーを使用してクライアントを再登録し、**--keytab** オプションでキータブの場所を指定します。

```
# ipa-client-install --keytab /etc/krb5.keytab
```



注記

登録を開始するために認証する場合は、**--keytab** オプションで指定するキータブのみが使用されます。再登録中、IdM はクライアントに対して新しいキータブを生成します。

41.9.4. インストール後の Identity Management クライアントのテスト

コマンドラインインターフェイスにより、**ipa-client-install** が正常に実行されたことが通知されますが、独自のテストを行うこともできます。

Identity Management クライアントが、サーバーに定義したユーザーに関する情報を取得できることをテストするには、サーバーに定義したユーザーを解決できることを確認します。たとえば、デフォルトの **admin** ユーザーを確認するには、次のコマンドを実行します。

```
[user@client1 ~]$ id admin
uid=1254400000(admin) gid=1254400000(admins) groups=1254400000(admins)
```

認証が適切に機能することをテストするには、別の IdM ユーザーで **su -** を実行します。

```
[user@client1 ~]$ su - idm_user
Last login: Thu Oct 18 18:39:11 CEST 2018 from 192.168.122.1 on pts/0
[idm_user@client1 ~]$
```

41.10. IDENTITY MANAGEMENT クライアントシステムの名前の変更

ここでは、Identity Management クライアントシステムのホスト名を変更する方法を説明します。

警告

クライアントの名前は手動で変更します。ホスト名の変更が絶対に必要である場合のみ実行してください。

Identity Management クライアントの名前を変更するには、以下を行う必要があります。

1. ホストを準備します。詳細は、名前を変更するための IdM クライアントの準備 を参照してください。
2. ホストから IdM クライアントをアンインストールします。詳細は、Identity Management クライアントのアンインストール を参照してください。
3. ホストの名前を変更します。詳細は、ホストシステムの名前変更 を参照してください。
4. 新しい名前でホストに IdM クライアントをインストールします。詳細は、**Identity Management のインストール** の Identity Management クライアントのインストール を参照してください。
5. IdM クライアントのインストール後にホストを設定します。詳しくはサービスの再追加、証明書の再生成、およびホストグループの再追加 を参照してください。

41.10.1. 名前を変更するための IdM クライアントの準備

現在のクライアントをアンインストールする前に、クライアントの設定を書き留めます。新しいホスト名のマシンを再登録した後にこの設定を適用します

- マシンで実行しているサービスを特定します。
  - **ipa service-find** コマンドを使用して、証明書のあるサービスを特定して出力します。

    ```
    $ ipa service-find old-client-name.example.com
    ```

  - さらに、各ホストには ipa service-find の出力に表示されないデフォルトの **host service** があります。ホストサービスのサービスプリンシパルは **ホストプリンシパル** とも呼ばれ、**host/old-client-name.example.com** になります。
- **ipa service-find old-client-name.example.com** により表示されるすべてのサービスプリンシパルは、**old-client-name.example.com** 上の対応するキータブの場所を決定します。

  ```
  # find / -name "*.keytab"
  ```

  クライアントシステムの各サービスには、**ldap/old-client-name.example.com@EXAMPLE.COM** のように service_name/host_name@REALM の形式を取る Kerberos プリンシパルがあります。
- マシンが所属するすべてのホストグループを特定します。

  ```
  # ipa hostgroup-find old-client-name.example.com
  ```

41.10.2. Identity Management クライアントのアンインストール

クライアントをアンインストールすると、クライアントは Identity Management ドメインから削除され、SSSD (System Security Services Daemon) などのシステムサービスの Identity Management 設定もすべて削除されます。これにより、クライアントシステムの以前の設定が復元します。

手順

1. **ipa-client-install --uninstall** コマンドを実行します。

```
[root@client]# ipa-client-install --uninstall
```

2. クライアントホストの DNS エントリーを、手動でサーバーから削除します。

```
[root@server]# ipa dnsrecord-del
Record name: old-client-client
Zone name: idm.example.com
No option to delete specific record provided.
Delete all? Yes/No (default No): true
-----------------------
Deleted record "old-client-name"
```

3. **/etc/krb5.keytab** 以外のキータブについては、古いプリンシパルを削除します。

```
[root@client ~]# ipa-rmkeytab -k /path/to/keytab -r EXAMPLE.COM
```

4. IdM サーバーで、ホストエントリーを削除します。これにより、すべてのサービスが削除され、そのホストに発行されたすべての証明書が無効になります。

```
[root@server ~]# ipa host-del client.example.com
```

41.10.3. ホストシステムの名前変更

必要に応じてマシンの名前を変更します。以下に例を示します。

```
[root@client]# hostnamectl set-hostname new-client-name.example.com
```

これで、新しいホスト名で、Identity Management クライアントを Identity Management ドメインに再インストールできるようになります。

41.10.4. サービスの再追加、証明書の再生成、およびホストグループの再追加

手順

1. Identity Management (IdM) サーバーで、名前を変更するための IdM クライアントの準備 に定義された各サービスに新しいキータブを追加します。

```
[root@server ~]# ipa service-add service_name/new-client-name
```

2. 名前を変更するための IdM クライアントの準備 で割り当てた証明書のあるサービスに対して証明書を生成します。これには、以下を行います。

- IdM 管理ツールの使用
- **certmonger** ユーティリティーの使用

3. 名前を変更するための IdM クライアントの準備 で特定されたホストグループにクライアントを再追加します。

41.11. ホストエントリーの無効化と再有効化

このセクションでは、Identity Management (IdM) でホストを無効にして再度有効にする方法を説明します。

41.11.1. ホストの無効化

IdM でホストエントリーを無効にするには、この手順を完了します。

ドメインサービス、ホスト、およびユーザーはアクティブなホストにアクセスできます。メンテナンス上の理由などで、アクティブなホストを一時的に削除することが必要になる場合があります。このような状況でホストを削除すると、ホストエントリーと、関連するすべての設定が完全に削除されるため、望ましくありません。代わりに、ホストを無効にするオプションを選択してください。

ホストを無効にすると、ドメインからドメインユーザーを完全に削除せずに、そのドメインにアクセスできなくなります。

手順

- **host-disable** コマンドを使用してホストを無効にします。ホストを無効にすると、ホストで現在アクティブなキータブが強制終了します。以下に例を示します。

```
$ kinit admin
$ ipa host-disable client.example.com
```

ホストを無効にすると、ホストはすべての IdM ユーザー、ホスト、およびサービスが利用できなくなりました。

重要

ホストエントリーを無効にすると、そのホストが無効になるだけではありません。そのホストで設定されているすべてのサービスも無効にします。

41.11.2. ホストの再有効化

無効な IdM ホストを再度有効にするには、次の手順に従います。

ホストを無効にすると、アクティブなキータブが強制的に終了し、設定エントリーを変更せずにホストが IdM ドメインから削除されます。

手順

- ホストを再度有効にするには、以下を追加して、**ipa-getkeytab** コマンドを使用します。
  - キータブを要求する IdM サーバーを指定する **-s** オプション
  - プリンシパル名を指定する **-p** オプション

- キータブを保存するファイルを指定する **-k** オプション

たとえば、**client.example.com** の **server.example.com** から新規ホストキータブを要求し、キータブを **/etc/krb5.keytab** ファイルに保存するには、次のコマンドを実行します。

```
$ ipa-getkeytab -s server.example.com -p host/client.example.com -k /etc/krb5.keytab -D
"cn=directory manager" -w password
```

注記

管理者の認証情報を使用して、**-D "uid=admin,cn=users,cn=accounts,dc=example,dc=com"** を指定することもできます。認証情報は、ホストのキータブの作成を許可されたユーザーに対応することが重要です。

ipa-getkeytab コマンドがアクティブな IdM クライアントまたはサーバーで実行する場合は、ユーザーが、**kinit admin** などを使用して TGT を取得した場合に、LDAP 認証情報 (**-D** および **-w**) を使用せずに実行できます。無効化されたホストでコマンドを直接実行するには、LDAP 認証情報を提供して IdM サーバーに認証します。

第42章 IDM WEB UI でホストエントリーの追加

この章では、Identity Management (IdM) のホストと、IdM Web UI でホストエントリーを追加する操作を説明します。

42.1. IDM のホスト

Identity Management (IdM) は、以下の ID を管理します。

- ユーザー
- サービス
- ホスト

ホストはマシンを表します。ホストには、IdM LDAP に IdM ID となるエントリーがあります。これは IdM サーバーの 389 Directory Server のインスタンスです。

IdM LDAP のホストエントリーは、その他のホストとドメイン内のサービスとの関係を確立するために使用されます。この関係では、ドメイン内ホストの認可および制御の **委譲** が不可欠な要素です。ホストは、**ホストベースのアクセス制御** (HBAC) ルールで使用できます。

IdM ドメインは、共通の ID 情報、共通ポリシー、および共有サービスを使用して、マシン間で共通性を確立します。ドメインのクライアントとしてのドメイン機能に属するマシンです。これは、ドメインが提供するサービスを使用することを意味します。IdM ドメインは、マシン専用の 3 つの主なサービスを提供します。

- DNS
- Kerberos
- 証明書の管理

IdM のホストは、そのホストで実行しているサービスと密接に接続されています。

- サービスエントリーは、ホストに関連付けられています。
- ホストには、ホストとサービスの両方の Kerberos プリンシパルが格納されます。

42.2. ホスト登録

本セクションでは、ホストを IdM クライアントとして登録し、登録中および登録後に何が起こるかを説明します。本セクションでは、IdM ホストの登録と、IdM ユーザーの登録を比較します。また、本セクションでは、ホストで利用可能な代替タイプの認証も概説します。

ホストの登録は、以下の要素で構成されます。

- IdM LDAP でのホストエントリーの作成: 場合によっては、IdM CLI で **ipa host-add** コマンドを使用するか、同等の IdM Web UI 操作 を使用します。
- ホストで IdM サービス (System Security Services Daemon (SSSD)、Kerberos、certmonger など) を設定し、ホストを IdM ドメインに参加させる。

2 つのアクションは、個別に実行することも一緒に実行することもできます。

個別に実行すると、異なるレベルの特権を持つ 2 人のユーザー間で 2 つのタスクを分割できます。これは、一括デプロイメントに役立ちます。

ipa-client-install コマンドは、2 つのアクションを一緒に実行できます。コマンドは、そのエントリーがまだ存在していない場合は IdM LDAP にホストエントリーを作成し、そのホストに Kerberos サービスと SSSD サービスの両方を設定します。このコマンドにより、ホストが IdM ドメイン内に移動し、接続先の IdM サーバーを識別できるようになります。IdM が管理する DNS ゾーンにホストが属する場合は、**ipa-client-install** でホストに DNS レコードも追加します。コマンドはクライアントで実行する必要があります。

42.3. ホストの登録に必要なユーザー権限

ホスト登録操作では、権限のないユーザーが不要なマシンを IdM ドメインに追加しないように、認証が必要になります。必要な特権は、次のようないくつかの要因によって異なります。

- ホストエントリーが **ipa-client-install** の実行とは別に作成される場合
- ワンタイムパスワード (OTP) が登録に使用される場合

必要に応じて IdM LDAP にホストエントリーを手動で作成するためのユーザー特権

CLI コマンド **ipa host-add** または IdM Web UI を使用して、IdM LDAP にホストエントリーを作成するのに必要なユーザー特権は **ホストの管理者** です。**ホスト管理者** の特権は、**IT スペシャリスト** ロールから取得できます。

クライアントを IdM ドメインに参加させるためのユーザー特権

ホストは、**ipa-client-install** コマンドの実行時に IdM クライアントとして設定されます。**ipa-client-install** コマンドの実行に必要な認証情報のレベルは、以下のような登録シナリオのどれに該当するかによって異なります。

- IdM LDAP のホストエントリーが存在しません。このシナリオでは、管理者の完全な認証情報または **ホスト管理者** ロールが必要です。完全な管理者とは **admins** グループのメンバーです。**ホスト管理者** ロールは、ホストの追加およびホストの登録の特権を提供します。このシナリオの詳細は ユーザー認証情報を使用したクライアントのインストール: 対話的なインストール を参照してください。
- IdM LDAP のホストエントリーが存在します。このシナリオでは、**ipa-client-install** を正常に実行するには、制限された管理者の認証情報が必要です。この場合、制限されている管理者には、**ホストの登録** 特権を提供する **登録管理者** ロールがあります。詳細は ユーザー認証情報を使用したクライアントのインストール: 対話的なインストール を参照してください。
- IdM LDAP にホストエントリーが存在し、完全または限定された管理者により、ホストの OTP が生成されました。このシナリオでは、正しい OTP を指定して **--password** オプションを指定して **ipa-client-install** コマンドを実行すると、通常のユーザーとして IdM クライアントをインストールできます。詳細は ワンタイムパスワードでクライアントのインストール: 対話的なインストール を参照してください。

登録後、IdM ホストは、IdM リソースにアクセスできるように、新しいセッションをすべて認証します。IdM サーバーがマシンを信頼し、そのマシンにインストールされているクライアントソフトウェアからの IdM 接続を受け入れるには、マシン認証が必要です。クライアントを認証すると、IdM サーバーはそのリクエストに応答できます。

42.4. IDM ホストとユーザーの登録と認証: 比較

IdM のユーザーとホストの間には多くの類似点があります。この類似点には、登録ステージで見られるものと、デプロイメントステージでの認証に関係するものがあります。

- 登録段階 (ユーザーおよびホストの登録):
  - 管理者は、ユーザーまたはホストが実際に IdM に参加する前に、ユーザーとホストの LDAP エントリーを作成できます。コマンドは、ステージユーザーの場合は **ipa stageuser-add** で、ホストの場合は **ipa host-add** です。
  - ホストで **ipa-client-install** コマンドを実行すると、**キーテーブル** (または略してキータブ)、(ある程度ユーザーパスワードに類似する) 対称キーを含むファイルが作成され、ホストが IdM レルムに参加します。同様に、ユーザーはアカウントをアクティブ化するときにパスワードを作成するように求められ、IdM レルムに参加します。
  - ユーザーパスワードは、ユーザーのデフォルトの認証方法ですが、キータブはホストのデフォルトの認証方法です。キータブは、ホストのファイルに保存されます。

表42.1 ユーザーおよびホストの登録

アクション	ユーザー	ホスト
登録前	$ ipa stageuser-add user_name [--password]	$ ipa host-add host_name [--random]
アカウントのアクティブ化	$ ipa stageuser-activate user_name	$ ipa-client install [--password] (ホスト自体で実行する必要があります)

- デプロイメント段階 (ユーザーおよびホストセッションの認証):
  - ユーザーが新しいセッションを開始すると、ユーザーはパスワードを使用して認証を行います。同様に、切り替え時に、ホストがそのキータブファイルを提示して認証を行います。SSSD (System Security Services Daemon) は、このプロセスをバックグラウンドで管理します。
  - 認証が成功すると、ユーザーまたはホストは、Kerberos チケット発行許諾チケット (TGT) を取得します。
  - 次に、TGT を使用して、特定のサービスの特定のチケットを取得します。

表42.2 ユーザーおよびホストセッションの認証

	ユーザー	ホスト
認証のデフォルト手段	**パスワード**	**キータブ**
セッションの開始 (通常のユーザー)	$ kinit user_name	**[ホストへの切り替え]**
認証成功の結果	TGT は、特定サービスへのアクセスの取得に使用されます。	TGT は、特定サービスへのアクセスの取得に使用されます。

TGT およびその他の Kerberos チケットは、サーバーにより定義された Kerberos サービスおよびポリシーの一部として生成されます。Kerberos チケットの最初の付与、Kerberos 認証情報の更新、および Kerberos セッションの破棄もすべて IdM サービスにより自動的に処理されます。

IdM ホストの代替認証オプション

キータブとは別に、IdM は、その他の 2 つのタイプのマシン認証にも対応しています。

- SSH 鍵。ホストの SSH 公開キーが作成され、ホストエントリーにアップロードされます。そこから、SSSD (System Security Services Daemon) は IdM を ID プロバイダーとして使用し、OpenSSH およびその他のサービスと一緒に機能して、IdM の中央にある公開鍵を参照できます。
- 機械の証明書。この場合、マシンは IdM サーバーの認証局により発行され、IdM の Directory Server に保存されている SSL 証明書を使用します。次に、証明書はマシンに送信され、サーバーに対する認証時に提示されます。クライアントでは、証明書は certmonger というサービスにより管理されます。

42.5. IDM LDAP のホストエントリー

Identity Management (IdM) ホストエントリーには、ホストに関する情報とホストに含めることができる属性が含まれています。

LDAP ホストエントリーは、IdM 内のクライアントに関するすべての関連情報が含まれます。

- ホストに関連付けられたサービスエントリー
- ホストおよびサービスプリンシパル
- アクセス制御ルール
- 物理的な場所やオペレーティングシステムなどのマシン情報

注記

IdM Web UI の **Identity → Hosts** タブには、IdM LDAP に保存されている特定のホストに関する情報がすべて表示されないことに注意してください。

ホストエントリー設定プロパティー

ホストエントリーには、ホストに関する情報 (物理的な場所、MAC アドレス、鍵、証明書など、システム設定を除く) を含めることができます。

この情報は、ホストエントリーが手動で作成された場合に、作成時に設定できます。また、この情報のほとんどは、ホストがそのドメインに登録してからホストエントリーに追加できます。

表42.3 ホスト設定プロパティー

UI フィールド	コマンドラインオプション	説明
説明	**--desc**=description	ホストの説明。
局所性	**--locality**=locality	ホストの地理的な場所。

UI フィールド	コマンドラインオプション	説明
場所	**--location**=location	データセンターのラックなど、ホストの物理的な場所。
プラットフォーム	**--platform**=string	ホストのハードウェアまたはアーキテクチャー。
オペレーティングシステム	**--os**=string	ホストのオペレーティングシステムとバージョン。
MAC アドレス	**--macaddress**=address	ホストの MAC アドレス。これは多値属性です。MAC アドレスは、NIS プラグインにより、ホスト用の NIS の **ethers** マップを作成するために使用されます。
SSH 公開鍵	**--sshpubkey**=string	ホストの完全 SSH 公開鍵。これは複数値の属性であるため、複数の鍵を設定できます。
プリンシパル名 (編集不可)	**--principalname**=principal	ホストの Kerberos プリンシパル名。**-p** に別のプリンシパルを明示的に設定しない限り、クライアントのインストール時にホスト名がデフォルトになります。これはコマンドラインツールを使用して変更できますが、UI で変更することはできません。
ワンタイムパスワードの設定	**--password**=string	このオプションは、一括登録で使用できるホストのパスワードを設定します。
-	**--random**	このオプションは、一括登録で使用されるランダムパスワードを生成します。
-	**--certificate**=string	ホストの証明書ブロブ。
-	**--updatedns**	これにより、IP アドレスが変更した場合に、ホストが DNS エントリーを動的に更新できるかどうかが設定されます。

42.6. WEB UI でのホストエントリーの追加

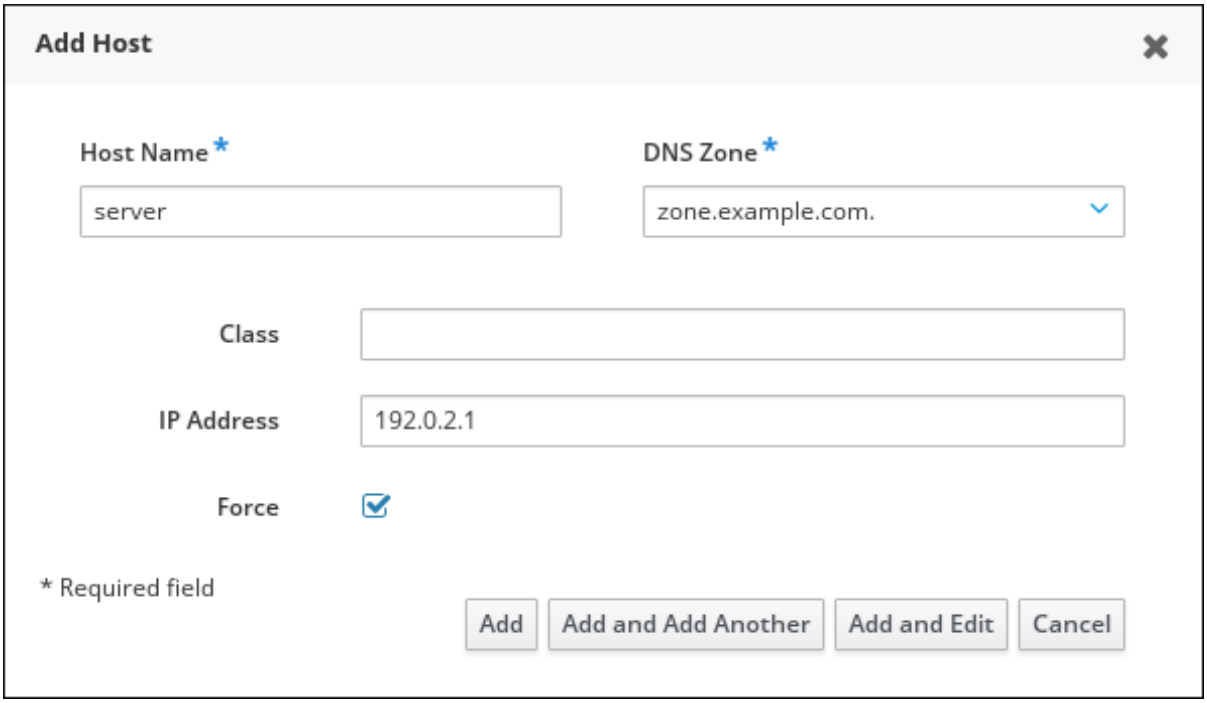
1. **Identity** タブを開き、サブタブの **ホスト** を選択します。

2. ホストリストの上部にある **追加** をクリックします。

図42.1 ホストエントリーの追加

3. マシンの名前を入力し、ドロップダウンリストで、設定済みのゾーンの中からドメインを選択します。ホストに静的 IP アドレスが割り当てられている場合は、ホストエントリーにそのアドレスを追加して、DNS エントリーが完全に作成されるようにします。
Class フィールドには、現時点では特定の目的はありません。

図42.2 ホストウィザードの追加



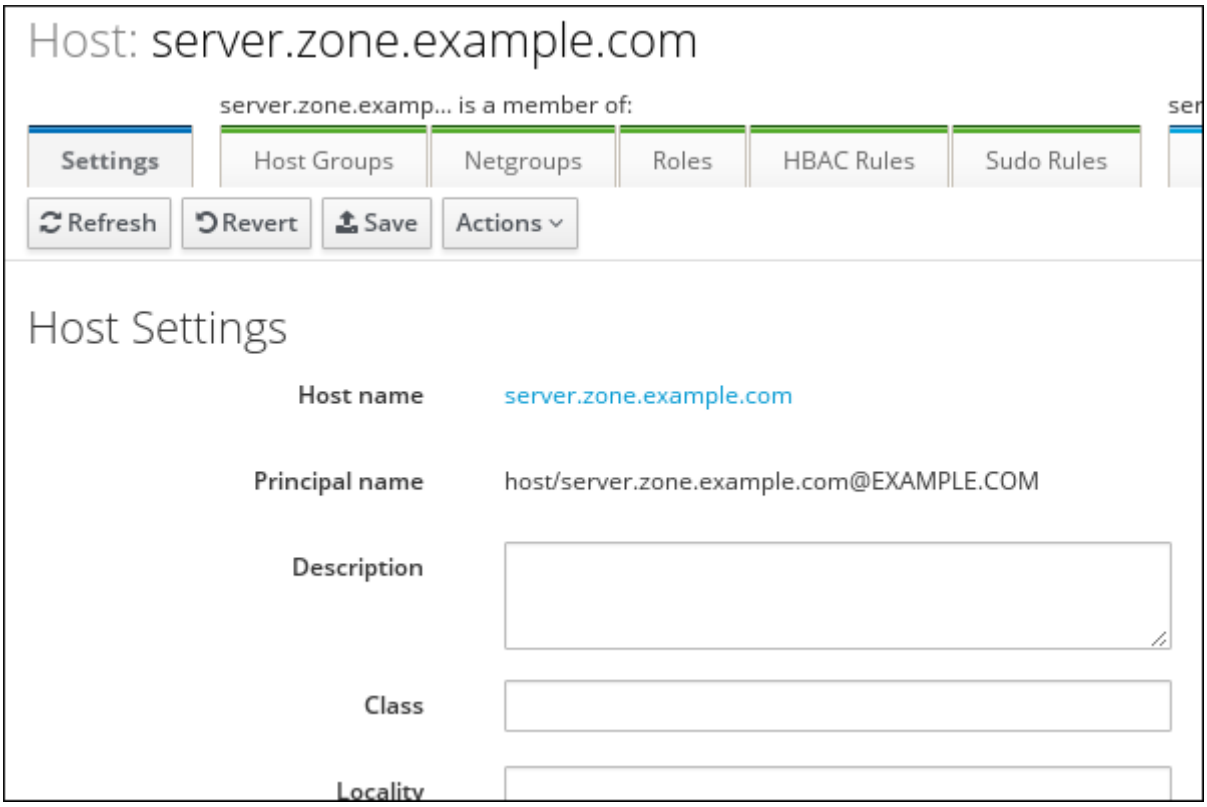
DNS ゾーンは IdM で作成できます。IdM サーバーが DNS サーバーを管理しない場合は、通常のテキストフィールドなど、メニューエリアでゾーンを手動で入力できます。

注記

ホストが DNS 経由で解決できるかどうかの確認を行わないようにするには、**強制** チェックボックスを選択します。

4. **追加および編集** ボタンをクリックして、拡張されたエントリーページに直接選択し、その他の属性情報を入力します。ホストのハードウェアと物理的な場所に関する情報は、ホストエントリーに追加できます。

図42.3 拡張されたエントリーページ



第43章 ANSIBLE PLAYBOOK を使用したホストの管理

Ansible は、システムの設定、ソフトウェアのデプロイ、ローリング更新の実行に使用する自動化ツールです。Ansible には Identity Management (IdM) のサポートが含まれ、Ansible モジュールを使用してホスト管理を自動化できます。

Ansible Playbook を使用してホストおよびホストエントリーを管理する際に、以下のコンセプトに基づき、操作が実行されます。

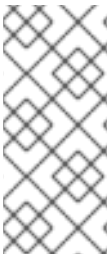
- **FQDN** でのみ定義されている IdM ホストエントリーを存在させる手順
- IP アドレスを使用して IdM ホストエントリーを存在させる手順
- 無作為のパスワードが指定された IdM ホストエントリーを複数存在させる手順
- 複数の IP アドレスが指定された IdM ホストエントリーを存在させる手順
- IdM ホストエントリーがないことを確認する手順

43.1. ANSIBLE PLAYBOOK を使用して FQDN が指定された IDM ホストエントリーを存在させる手順

以下の手順に従って、Ansible Playbook を使用して、Identity Management (IdM) にホストエントリーが存在することを確認します。ホストエントリーは、**完全修飾ドメイン名** (FQDN) によってのみ定義されます。

以下の条件のいずれかが当てはまる場合は、ホストの **FQDN** 名を指定するだけで十分です。

- IdM サーバーが DNS を管理するよう設定されていない。
- ホストに静的 IP アドレスがないか、ホストの設定時に IP アドレスが不明である。**FQDN** だけで定義されたホストを追加すると、基本的に IdM DNS サービスにプレースホルダーエントリーが作成されます。たとえば、ラップトップは IdM クライアントとして事前設定されている場合がありますが、設定時には IP アドレスがありません。DNS サービスがレコードを動的に更新すると、ホストの現在の IP アドレスが検出され、DNS レコードが更新されます。



注記

Ansible ない場合に、**ipa host-add** コマンドを使用すると、ホストエントリーが IdM に作成されます。ホストを IdM に追加すると、IdM でのホストの状態が present になります。Ansible は冪等性に依存しているので、Ansible を使用して IdM にホストを追加するには、ホストの状態を Present (**state: present**) として定義した Playbook を作成する必要があります。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。

- ~/MyPlaybooks/ ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
- この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。

- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。

手順

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

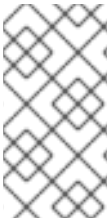
```
[ipaserver]
server.idm.example.com
```

2. IdM に存在させるホストの **FQDN** を使用して Ansible Playbook ファイルを作成します。この手順は、**/usr/share/doc/ansible-freeipa/playbooks/host/add-host.yml** ファイルのサンプルをコピーして変更し、簡素化できます。

```
---
- name: Host present
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Host host01.idm.example.com present
    ipahost:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: host01.idm.example.com
      state: present
      force: true
```

3. Playbook を実行します。

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



注記

以下の手順では、IdM LDAP サーバーにホストエントリーが作成されますが、ホストは IdM Kerberos レルムには登録されません。登録されるようにするには、ホストを IdM クライアントとしてデプロイする必要があります。詳細は、Ansible Playbook を使用した Identity Management クライアントのインストール を参照してください。

検証手順

1. admin として IdM サーバーにログインします。

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

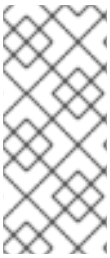
2. **ipa host-show** コマンドを入力し、ホストの名前を指定します。

```
$ ipa host-show host01.idm.example.com
  Host name: host01.idm.example.com
  Principal name: host/host01.idm.example.com@IDM.EXAMPLE.COM
  Principal alias: host/host01.idm.example.com@IDM.EXAMPLE.COM
  Password: False
  Keytab: False
  Managed by: host01.idm.example.com
```

この出力で、**host01.idm.example.com** が IdM に存在することを確認します。

43.2. ANSIBLE PLAYBOOK を使用して DNS 情報など IDM ホストエントリーを存在させる手順

以下の手順に従って、Ansible Playbook を使用して、Identity Management (IdM) にホストエントリーが存在することを確認します。ホストエントリーは、ホストの **完全修飾ドメイン名** (FQDN) および IP アドレスで定義されます。



注記

Ansible ない場合に、**ipa host-add** コマンドを使用すると、ホストエントリーが IdM に作成されます。ホストを IdM に追加すると、IdM でのホストの状態が present になります。Ansible は冪等性に依存しているので、Ansible を使用して IdM にホストを追加するには、ホストの状態を Present (**state: present**) として定義した Playbook を作成する必要があります。

前提条件

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

手順

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

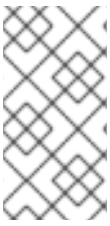
```
[ipaserver]
server.idm.example.com
```

2. IdM に存在させるホストの **完全修飾ドメイン名** (FQDN) で Ansible Playbook ファイルを作成します。また、IdM サーバーが DNS を管理するように設定され、ホストの IP アドレスが分かっている場合は、**ip_address** パラメーターの値を指定します。ホストを DNS リソースレコードに存在させるには、IP アドレスが必要です。この手順は、**/usr/share/doc/ansible-freeipa/playbooks/host/host-present.yml** ファイルのサンプルをコピーして変更し、簡素化できます。また、その他の追加情報を含めることもできます。

```
---
- name: Host present
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure host01.idm.example.com is present
    ipahost:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: host01.idm.example.com
      description: Example host
      ip_address: 192.168.0.123
      locality: Lab
      ns_host_location: Lab
      ns_os_version: CentOS 7
      ns_hardware_platform: Lenovo T61
      mac_address:
      - "08:00:27:E3:B1:2D"
      - "52:54:00:BD:97:1E"
      state: present
```

3. Playbook を実行します。

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



注記

以下の手順では、IdM LDAP サーバーにホストエントリーが作成されますが、ホストは IdM Kerberos レルムには登録されません。登録されるようにするには、ホストを IdM クライアントとしてデプロイする必要があります。詳細は、Ansible Playbook を使用した Identity Management クライアントのインストール を参照してください。

検証手順

1. admin として IdM サーバーにログインします。

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

2. **ipa host-show** コマンドを入力し、ホストの名前を指定します。

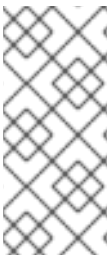
```
$ ipa host-show host01.idm.example.com
  Host name: host01.idm.example.com
  Description: Example host
```

```
Locality: Lab
Location: Lab
Platform: Lenovo T61
Operating system: CentOS 7
Principal name: host/host01.idm.example.com@IDM.EXAMPLE.COM
Principal alias: host/host01.idm.example.com@IDM.EXAMPLE.COM
MAC address: 08:00:27:E3:B1:2D, 52:54:00:BD:97:1E
Password: False
Keytab: False
Managed by: host01.idm.example.com
```

この出力で、**host01.idm.example.com** が IdM に存在することを確認します。

43.3. ANSIBLE PLAYBOOK を使用して無作為のパスワードが指定された IDM ホストエントリーを複数存在させる手順

ipahost モジュールでは、システム管理者は、Ansible タスク 1 つだけを使用して、IdM に複数のホストエントリーが存在するか、存在しないかを確認できます。以下の手順に従って、**fully-qualified domain names** (FQDN) でのみ定義されるホストエントリーを複数存在することを確認します。Ansible Playbook を実行すると、ホストのパスワードが無作為に生成されます。



注記

Ansible ない場合に、**ipa host-add** コマンドを使用すると、ホストエントリーが IdM に作成されます。ホストを IdM に追加すると、IdM でのホストの状態が present になります。Ansible は冪等性に依存しているので、Ansible を使用して IdM にホストを追加するには、ホストの状態を Present (**state: present**) として定義した Playbook を作成する必要があります。

前提条件

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

手順

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

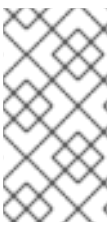
```
[ipaserver]
server.idm.example.com
```

2. IdM に存在させるホストの **完全修飾ドメイン名** (FQDN) で Ansible Playbook ファイルを作成します。IdM にホストがすでに存在している場合でも Ansible Playbook が各ホストに対して無作為にパスワードを生成し、**update_password** が **on_create** に制限されている場合は、**random: true** および **force: true** オプションを追加します。この手順を簡素化するには、**/usr/share/doc/ansible-freeipa/README-host.md** Markdown ファイルからサンプルをコピーして変更できます。

```
---
- name: Ensure hosts with random password
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Hosts host01.idm.example.com and host02.idm.example.com present with random
passwords
    ipahost:
      ipaadmin_password: "{{ ipaadmin_password }}"
      hosts:
      - name: host01.idm.example.com
        random: true
        force: true
      - name: host02.idm.example.com
        random: true
        force: true
    register: ipahost
```

3. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file path_to_playbooks_directory/ensure-hosts-
are-present.yml
[...]
TASK [Hosts host01.idm.example.com and host02.idm.example.com present with random
passwords]
changed: [r8server.idm.example.com] => {"changed": true, "host":
{"host01.idm.example.com": {"randompassword": "0HoIRvjUdH0Ycbf6uYdWTxH"},
"host02.idm.example.com": {"randompassword": "5VdLgrf3wvojmACdHC3uA3s"}}}
```



注記

ランダムなワンタイムパスワード (OTP) を使用して、ホストを IdM クライアントとしてデプロイする場合は、Authorization options for IdM client enrollment using an Ansible playbook または Installing a client by using a one-time password: Interactive installation を参照してください。

検証手順

1. admin として IdM サーバーにログインします。

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

2. **ipa host-show** コマンドを入力し、ホストのいずれかの名前を指定します。

```
$ ipa host-show host01.idm.example.com
  Host name: host01.idm.example.com
  Password: True
  Keytab: False
  Managed by: host01.idm.example.com
```

この出力で、**host01.idm.example.com** が無作為に作成されたパスワードが指定された IdM に存在することを確認します。

43.4. ANSIBLE PLAYBOOK を使用して複数の IP アドレスが指定された IDM ホストエントリーを存在させる手順

以下の手順に従って、Ansible Playbook を使用して Identity Management (IdM) にホストエントリーが存在することを確認します。ホストエントリーは、**完全修飾ドメイン名** (FQDN) と複数の IP アドレスで定義されます。

注記

Ansible **ipahost** モジュールでは、**ipa host** ユーティリティーとは対照的に、ホストの IPv4 および IPv6 アドレスが複数存在させたり、または存在させなかったりできます。**ipa host-mod** コマンドは IP アドレスを処理できません。

前提条件

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

手順

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

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

2. Ansible Playbook ファイルを作成します。**ipahost** 変数の **名前** として、IdM に存在させるホストの **完全修飾ドメイン名** (FQDN) を指定します。**ip_address** 構文を使用して、複数の IPv4 および IPv6 **ip_address** 値をそれぞれ別の行に指定します。この手順

は、**/usr/share/doc/ansible-freeipa/playbooks/host/host-member-ipaddresses-present.yml** ファイルのサンプルをコピーして変更し、簡素化できます。追加情報を含めることもできます。

```
---
- name: Host member IP addresses present
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure host101.example.com IP addresses present
    ipahost:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: host01.idm.example.com
      ip_address:
      - 192.168.0.123
      - fe80::20c:29ff:fe02:a1b3
      - 192.168.0.124
      - fe80::20c:29ff:fe02:a1b4
      force: true
```

3. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file path_to_playbooks_directory/ensure-host-
with-multiple-IP-addreses-is-present.yml
```

注記

この手順では、IdM LDAP サーバーにホストエントリーは作成されますが、ホストは IdM Kerberos レルムに登録されません。登録されるようにするには、ホストを IdM クライアントとしてデプロイする必要があります。詳細は、Ansible Playbook を使用した Identity Management クライアントのインストール を参照してください。

検証手順

1. admin として IdM サーバーにログインします。

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

2. **ipa host-show** コマンドを入力し、ホストの名前を指定します。

```
$ ipa host-show host01.idm.example.com
  Principal name: host/host01.idm.example.com@IDM.EXAMPLE.COM
  Principal alias: host/host01.idm.example.com@IDM.EXAMPLE.COM
  Password: False
  Keytab: False
  Managed by: host01.idm.example.com
```

この出力で、**host01.idm.example.com** が IdM に存在することを確認します。

3. IdM DNS レコードにホストの複数の IP アドレスが存在することを確認するには、**ipa dnsrecord-show** コマンドを入力し、以下の情報を指定します。

   - IdM ドメインの名前
   - ホストの名前

   ```
   $ ipa dnsrecord-show idm.example.com host01
   [...]
     Record name: host01
     A record: 192.168.0.123, 192.168.0.124
     AAAA record: fe80::20c:29ff:fe02:a1b3, fe80::20c:29ff:fe02:a1b4
   ```

   この出力では、Playbook で指定された IPv4 アドレスおよび IPv6 アドレスがすべて **host01.idm.example.com** ホストエントリーに正しく関連付けられていることを確認します。

43.5. ANSIBLE PLAYBOOK を使用して IDM ホストエントリーがないことを確認する手順

以下の手順に従って、Ansible Playbook を使用して Identity Management (IdM) にホストエントリーがないことを確認します。

前提条件

- IdM 管理者の認証情報

手順

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

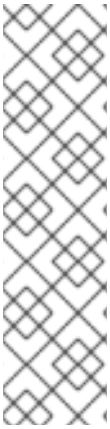
   ```
   [ipaserver]
   server.idm.example.com
   ```

2. IdM に存在させないホストの **完全修飾ドメイン名** (FQDN) を指定して Ansible Playbook ファイルを作成します。IdM ドメインに DNS が統合されている場合は、**updatedns: true** オプションを使用して、あらゆる種類のホストに関連するレコードを DNS から削除します。
   この手順は、**/usr/share/doc/ansible-freeipa/playbooks/host/delete-host.yml** ファイルのサンプルをコピーして変更し、簡素化できます。

   ```
   ---
   - name: Host absent
     hosts: ipaserver

     vars_files:
     - /home/user_name/MyPlaybooks/secret.yml
     tasks:
     - name: Host host01.idm.example.com absent
       ipahost:
         ipaadmin_password: "{{ ipaadmin_password }}"
         name: host01.idm.example.com
         updatedns: true
         state: absent
   ```

3. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file path_to_playbooks_directory/ensure-host-
absent.yml
```



注記

この手順の結果は以下のようになります。

- IdM Kerberos レルムにホストが存在していない。
- IdM LDAP サーバーにホストエントリーが存在しない。

SSSD (System Security Services Daemon) などのシステムサービスの特定の IdM 設定をクライアントホスト自体から削除するには、クライアントで **ipa-client-install --uninstall** コマンドを実行する必要があります。詳細は、IdM クライアントのアンインストール を参照してください。

検証手順

1. admin として **ipaserver** にログインします。

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

2. **host01.idm.example.com** に関する情報を表示します。

```
$ ipa host-show host01.idm.example.com
ipa: ERROR: host01.idm.example.com: host not found
```

この出力では、ホストが IdM に存在しないことを確認します。

43.6. 関連情報

- **/usr/share/doc/ansible-freeipa/README-host.md** Markdown ファイルを参照してください。
- **/usr/share/doc/ansible-freeipa/playbooks/host** ディレクトリーにある追加の Playbook を表示します。

第44章 IDM CLI を使用したホストグループの管理

次の操作を使用して、コマンドラインインターフェイス (CLI) でホストグループとそのメンバーを管理する方法について詳しく説明します。

- ホストグループおよびそのメンバーの表示
- ホストグループの作成
- ホストグループの削除
- ホストグループメンバーの追加
- ホストグループメンバーの削除
- ホストグループメンバーマネージャーの追加
- ホストグループメンバーマネージャーの削除

44.1. IDM のホストグループ

IdM ホストグループを使用すると、重要な管理タスク (特にアクセス制御) を一元管理できます。

ホストグループの定義

ホストグループは、一般的なアクセス制御ルールやその他の特性を持つ IdM ホストセットが含まれるエンティティーです。たとえば、企業の部門、物理的な場所、またはアクセス制御要件に基づいてホストグループを定義できます。

IdM のホストグループには以下が含まれます。

- IdM サーバーおよびクライアント
- その他の IdM ホストグループ

デフォルトで作成されたホストグループ

デフォルトでは、IdM サーバーは、全 IdM サーバーホストのホストグループ **ipaservers** を作成します。

直接および間接のグループメンバー

IdM のグループ属性は、直接メンバーと間接メンバーの両方に適用されます。ホストグループ B がホストグループ A のメンバーである場合、ホストグループ B のすべてのユーザーはホストグループ A の間接メンバーと見なされます。

44.2. CLI での IDM ホストグループの表示

コマンドラインインターフェイス (CLI) を使用して IdM ホストグループを表示するには、次の手順に従います。

前提条件

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

手順

1. **ipa hostgroup-find** コマンドを使用して、すべてのホストグループを検索します。

```
$ ipa hostgroup-find
------------------
1 hostgroup matched
------------------
  Host-group: ipaservers
  Description: IPA server hosts
----------------------------
Number of entries returned 1
----------------------------
```

ホストグループのすべての属性を表示するには、**--all** オプションを追加します。以下に例を示します。

```
$ ipa hostgroup-find --all
------------------
1 hostgroup matched
------------------
  dn: cn=ipaservers,cn=hostgroups,cn=accounts,dc=idm,dc=local
  Host-group: ipaservers
  Description: IPA server hosts
  Member hosts: xxx.xxx.xxx.xxx
  ipauniqueid: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
  objectclass: top, groupOfNames, nestedGroup, ipaobject, ipahostgroup
----------------------------
Number of entries returned 1
----------------------------
```

44.3. CLI を使用した IDM ホストグループの作成

コマンドラインインターフェイス (CLI) を使用して IdM ホストグループを作成するには、次の手順に従います。

前提条件

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

手順

1. **ipa hostgroup-add** コマンドを使用してホストグループを追加します。
たとえば、**group_name** という名前の IdM ホストグループを作成して説明を追加するには、次のコマンドを実行します。

```
$ ipa hostgroup-add --desc 'My new host group' group_name
---------------------
Added hostgroup "group_name"
---------------------
```

```
  Host-group: group_name
  Description: My new host group
----------------------
```

44.4. CLI での IDM ホストグループの削除

コマンドラインインターフェイス (CLI) を使用して IdM ホストグループを削除するには、次の手順に従います。

前提条件

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

手順

1. **ipa hostgroup-del** コマンドを使用してホストグループを削除します。
たとえば、**group_name** という名前の IdM ホストグループを削除するには、次のコマンドを実行します。

```
$ ipa hostgroup-del group_name
--------------------------
Deleted hostgroup "group_name"
--------------------------
```

注記

グループを削除しても、IdM からグループメンバーは削除されません。

44.5. CLI での IDM ホストグループメンバーの追加

コマンド 1 つで、ホストとホストグループを IdM ホストグループのメンバーとして追加できます。

前提条件

- IdM、またはユーザー管理者ロールを管理する管理者権限
- 有効な Kerberos チケット。詳細は、Using kinit to log in to IdM manually を参照してください。
- **オプション。ipa hostgroup-find** コマンドを使用して、ホストおよびホストグループを検索します。

手順

1. ホストグループにメンバーを追加するには、**ipa hostgroup-add-member** を使用して、関連する情報を指定します。以下のオプションを使用して、追加するメンバーのタイプを指定できます。
    - **--hosts** オプションを使用して、ホストを IdM ホストグループに追加します。
    たとえば、**example_member** という名前のホストを **group_name** という名前のグループに追加するには、次のコマンドを実行します。

```
$ ipa hostgroup-add-member group_name --hosts example_member
Host-group: group_name
Description: My host group
Member hosts: example_member
-------------------------
Number of members added 1
-------------------------
```

- **--hostgroups** オプションを使用して、IdM ホストグループにホストグループを追加します。
  たとえば、**nested_group** という名前のホストグループを **group_name** という名前のグループに追加するには、次のコマンドを実行します。

```
$ ipa hostgroup-add-member group_name --hostgroups nested_group
Host-group: group_name
Description: My host group
Member host-groups: nested_group
-------------------------
Number of members added 1
-------------------------
```

- 以下の構文を使用すると、単一のコマンドで複数のホストと複数のホストグループを IdM ホストグループに追加できます。

```
$ ipa hostgroup-add-member group_name --hosts={host1,host2} --hostgroups=
{group1,group2}
```

重要

ホストグループを別のホストグループのメンバーとして追加する場合は、再帰グループを作成しないでください。たとえば、グループ A がグループ B のメンバーである場合は、グループ B をグループ A のメンバーとして追加しないでください。再帰的なグループにより予期しない動作が発生する可能性があります。

44.6. CLI での IDM ホストグループメンバーの削除

コマンド 1 つで IdM ホストグループからホストとホストグループを削除できます。

前提条件

- IdM、またはユーザー管理者ロールを管理する管理者権限
- 有効な Kerberos チケット。詳細は、Using kinit to log in to IdM manually を参照してください。
- **オプション。ipa hostgroup-find** コマンドを使用して、削除するメンバーがグループに含まれていることを確認します。

手順

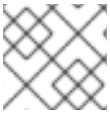
1. ホストグループメンバーを削除するには、**ipa hostgroup-remove-member** コマンドを使用して、関連する情報を指定します。以下のオプションを使用して、削除するメンバーのタイプを指定できます。

- **--hosts** オプションを使用して、IdM ホストグループからホストを削除します。
  たとえば、**example_member** という名前のホストを **group_name** という名前のグループから削除するには、次のコマンドを実行します。

```
$ ipa hostgroup-remove-member group_name --hosts example_member
Host-group: group_name
Description: My host group
-------------------------
Number of members removed 1
-------------------------
```

- **--hostgroups** オプションを使用して、IdM ホストグループからホストグループを削除します。
  たとえば、**nested_group** という名前のグループから **group_name** という名前のホストグループを削除するには、次のコマンドを実行します。

```
$ ipa hostgroup-remove-member group_name --hostgroups example_member
Host-group: group_name
Description: My host group
-------------------------
Number of members removed 1
-------------------------
```



注記

グループを削除しても、IdM からグループメンバーは削除されません。

- 以下の構文を使用すると、単一のコマンドで IdM ホストグループから複数のホストとホストグループを削除できます。

```
$ ipa hostgroup-remove-member group_name --hosts={host1,host2} --hostgroups={group1,group2}
```

44.7. CLI を使用した IDM ホストグループメンバーマネージャーの追加

コマンド 1 つで、ホストとホストグループを IdM ホストグループのメンバーとして追加できます。メンバーマネージャーは、ホストまたはホストグループを IdM ホストグループに追加できますが、ホストグループの属性は変更できません。

前提条件

- IdM、またはユーザー管理者ロールを管理する管理者権限
- 有効な Kerberos チケット。詳細は、Using kinit to log in to IdM manually を参照してください。
- メンバーマネージャーとして追加するホストまたはホストグループの名前と、管理するホストグループ名が必要です。

手順

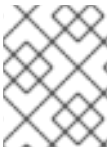
1. **オプション。ipa hostgroup-find** コマンドを使用して、ホストおよびホストグループを検索します。

2. ホストグループにメンバーマネージャーを追加するには、**ipa hostgroup-add-member-manager** を使用します。
たとえば、 **example_member** という名前のユーザーを、**group_name** という名前のグループにメンバーマネージャーとして追加するには、次のコマンドを実行します。

```
$ ipa hostgroup-add-member-manager group_name --user example_member
Host-group: group_name
Member hosts: server.idm.example.com
Member host-groups: project_admins
Member of netgroups: group_name
Membership managed by users: example_member
-------------------------
Number of members added 1
-------------------------
```

3. **--groups** オプションを使用して、1つ以上のホストグループをメンバーマネージャーとして IdM ホストグループに追加します。
たとえば、**admin_group** という名前のホストグループを、**group_name** という名前のグループにメンバーマネージャーとして追加するには、次のコマンドを実行します。

```
$ ipa hostgroup-add-member-manager group_name --groups admin_group
Host-group: group_name
Member hosts: server.idm.example.com
Member host-groups: project_admins
Member of netgroups: group_name
Membership managed by groups: admin_group
Membership managed by users: example_member
-------------------------
Number of members added 1
-------------------------
```



注記

ホストグループにメンバーマネージャーを追加してから、更新が Identity Management 環境のすべてのクライアントに広がるまでに時間がかかる場合があります。

検証手順

- **ipa group-show** コマンドを使用して、ホストユーザーおよびグループがメンバーマネージャーとして追加されたことを確認します。

```
$ ipa hostgroup-show group_name
Host-group: group_name
Member hosts: server.idm.example.com
Member host-groups: project_admins
Membership managed by groups: admin_group
Membership managed by users: example_member
```

関連情報

- 詳細は、**ipa hostgroup-add-member-manager --help** を参照してください。
- 詳細は、**ipa hostgroup-show --help** を参照してください。

44.8. CLI での IDM ホストグループメンバーマネージャーの削除

1 つのコマンドで IdM ホストグループのメンバーマネージャーからホストとホストグループを削除できます。メンバーマネージャーは、IdM ホストグループからホストグループのメンバーマネージャーを削除できますが、ホストグループの属性は変更できません。

前提条件

- IdM、またはユーザー管理者ロールを管理する管理者権限
- 有効な Kerberos チケット。詳細は、Using kinit to log in to IdM manually を参照してください。
- 削除する既存のメンバーマネージャーのホストグループ名と、管理するホストグループ名が必要です。

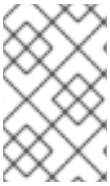
手順

1. **オプション。ipa hostgroup-find** コマンドを使用して、ホストおよびホストグループを検索します。
2. ホストグループからメンバーマネージャーを削除するには、**ipa hostgroup-remove-member-manager** コマンドを使用します。
   たとえば、 **example_member** という名前のユーザーを、**group_name** という名前のグループのメンバーマネージャーから削除するには、次のコマンドを実行します。

```
$ ipa hostgroup-remove-member-manager group_name --user example_member
Host-group: group_name
Member hosts: server.idm.example.com
Member host-groups: project_admins
Member of netgroups: group_name
Membership managed by groups: nested_group
---------------------------
Number of members removed 1
---------------------------
```

3. **--groups** オプションを使用して、IdM ホストグループのメンバーマネージャーからホストグループを 1 つまたは複数削除します。
   たとえば、**group_name** という名前のグループのメンバーマネージャーから **nested_group** という名前のホストグループを削除するには、次のコマンドを実行します。

```
$ ipa hostgroup-remove-member-manager group_name --groups nested_group
Host-group: group_name
Member hosts: server.idm.example.com
Member host-groups: project_admins
Member of netgroups: group_name
---------------------------
Number of members removed 1
---------------------------
```



注記

ホストグループからメンバーマネージャーを削除してから、更新が Identity Management 環境のすべてのクライアントに広がるまでに時間がかかる場合があります。

検証手順

- **ipa group-show** コマンドを使用して、メンバーマネージャーからホストユーザーとホストグループが削除されていることを確認します。

```
$ ipa hostgroup-show group_name
Host-group: group_name
Member hosts: server.idm.example.com
Member host-groups: project_admins
```

関連情報

- 詳細は、**ipa hostgroup-remove-member-manager --help** を参照してください。
- 詳細は、**ipa hostgroup-show --help** を参照してください。

第45章 IDM WEB UI を使用したホストグループの管理

次の操作を使用して、Web インターフェイス (Web UI) でホストグループとそのメンバーを管理する方法について詳しく説明します。

- ホストグループおよびそのメンバーの表示
- ホストグループの作成
- ホストグループの削除
- ホストグループメンバーの追加
- ホストグループメンバーの削除
- ホストグループメンバーマネージャーの追加
- ホストグループメンバーマネージャーの削除

45.1. IDM のホストグループ

IdM ホストグループを使用すると、重要な管理タスク (特にアクセス制御) を一元管理できます。

ホストグループの定義

ホストグループは、一般的なアクセス制御ルールやその他の特性を持つ IdM ホストセットが含まれるエンティティーです。たとえば、企業の部門、物理的な場所、またはアクセス制御要件に基づいてホストグループを定義できます。

IdM のホストグループには以下が含まれます。

- IdM サーバーおよびクライアント
- その他の IdM ホストグループ

デフォルトで作成されたホストグループ

デフォルトでは、IdM サーバーは、全 IdM サーバーホストのホストグループ **ipaservers** を作成します。

直接および間接のグループメンバー

IdM のグループ属性は、直接メンバーと間接メンバーの両方に適用されます。ホストグループ B がホストグループ A のメンバーである場合、ホストグループ B のすべてのユーザーはホストグループ A の間接メンバーと見なされます。

45.2. IDM WEB UI でのホストグループの表示

Web インターフェイス (Web UI) を使用して IdM ホストグループを表示するには、次の手順に従います。

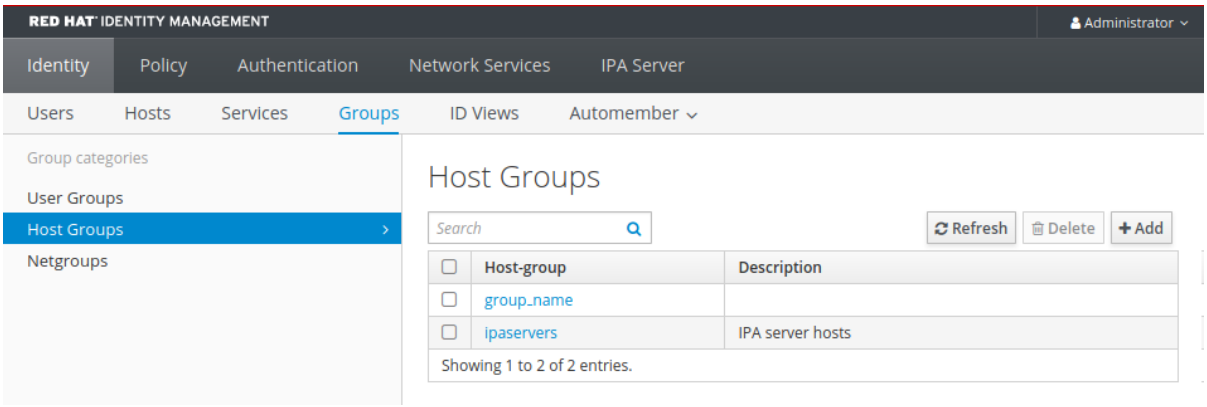
前提条件

- IdM、またはユーザー管理者ロールを管理する管理者権限

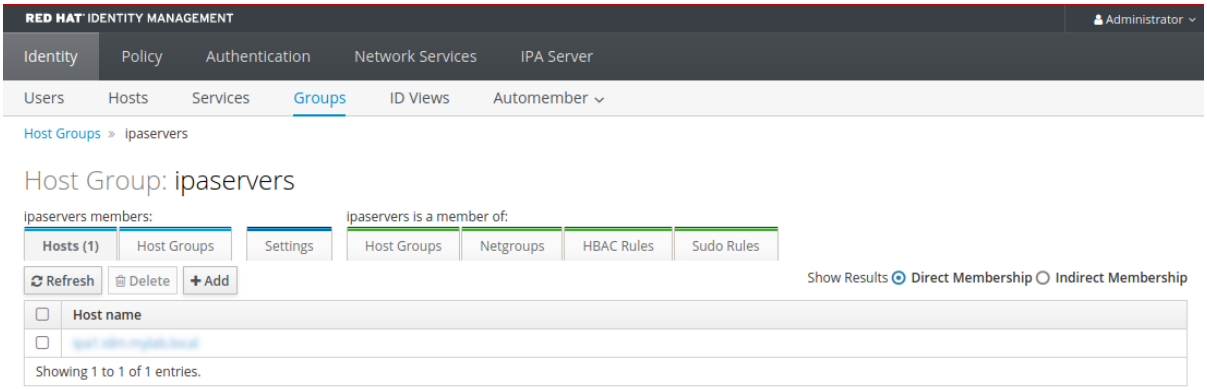
- IdM Web UI にログインしている。詳細は、Web ブラウザーでの IdM Web UI へのアクセス を参照してください。

手順

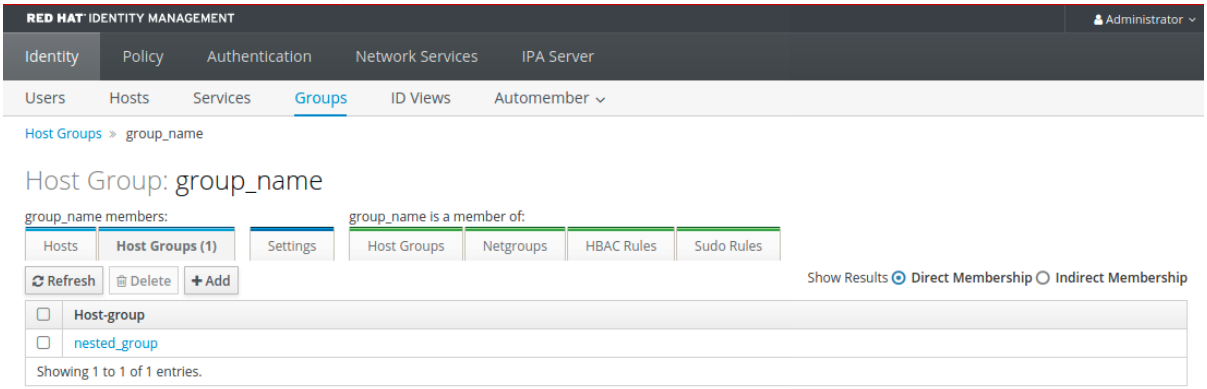
1. **Identity → Groups** をクリックし、**Host Groups** タブを選択します。
   - このページには、既存のホストグループとその説明が記載されています。
   - 特定のホストグループを検索できます。



2. リストのグループをクリックして、このグループに所属するホストを表示します。結果は、直接または間接メンバーに限定できます。



3. **ホストグループ** タブを選択して、このグループに所属するホストグループを表示します (ネスト化されたホストグループ)。結果は、直接または間接メンバーに限定できます。



45.3. IDM WEB UI でのホストグループの作成

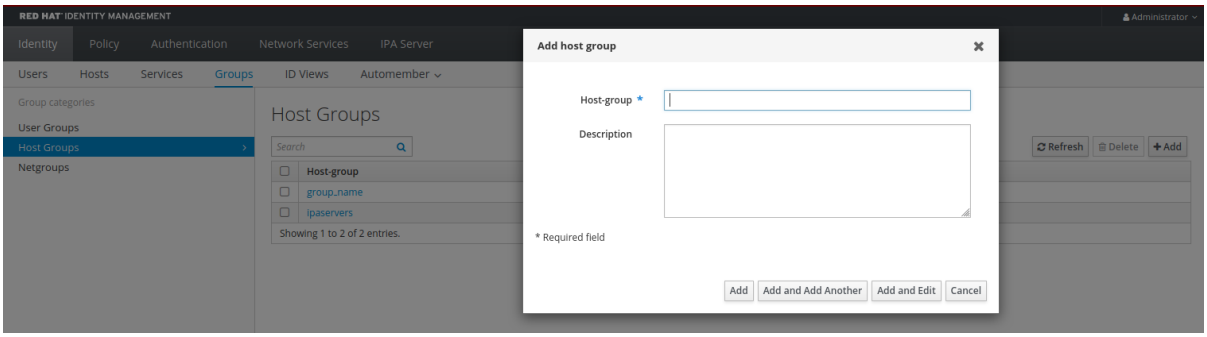
Web インターフェイス (Web UI) を使用して IdM ホストグループを作成するには、次の手順に従います。

前提条件

- IdM、またはユーザー管理者ロールを管理する管理者権限
- IdM Web UI にログインしている。詳細は、Web ブラウザーでの IdM Web UI へのアクセス を参照してください。

手順

1. **Identity → Groups** をクリックし、**Host Groups** タブを選択します。
2. **Add** をクリックします。**Add host group** ダイアログが表示されます。
3. Group: name (必須) および description (オプション) についての情報を指定します。
4. **Add** をクリックして確定します。



45.4. IDM WEB UI でのホストグループの削除

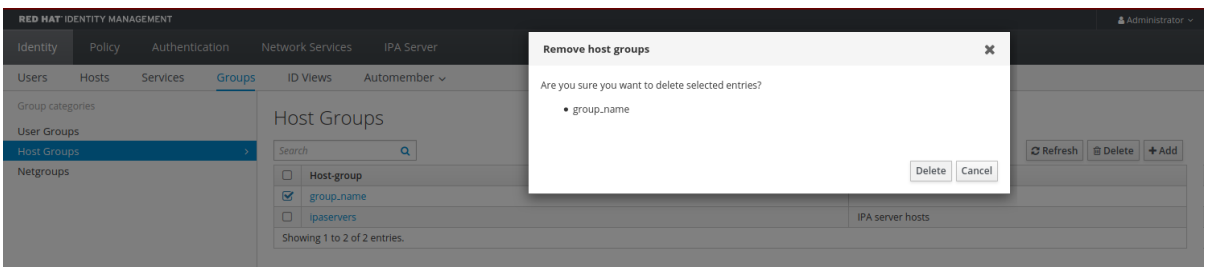
Web インターフェイス (Web UI) を使用して IdM ホストグループを削除するには、次の手順に従います。

前提条件

- IdM、またはユーザー管理者ロールを管理する管理者権限
- IdM Web UI にログインしている。詳細は、Web ブラウザーでの IdM Web UI へのアクセス を参照してください。

手順

1. **Identity → Groups** をクリックし、**Host Groups** タブを選択します。
2. 削除する IdM ホストグループを選択し、**削除** をクリックします。確認ダイアログが表示されます。
3. **Delete** をクリックして確定します。



注記

ホストグループを削除しても、IdM からグループメンバーは削除されません。

45.5. IDM WEB UI でのホストグループメンバーの追加

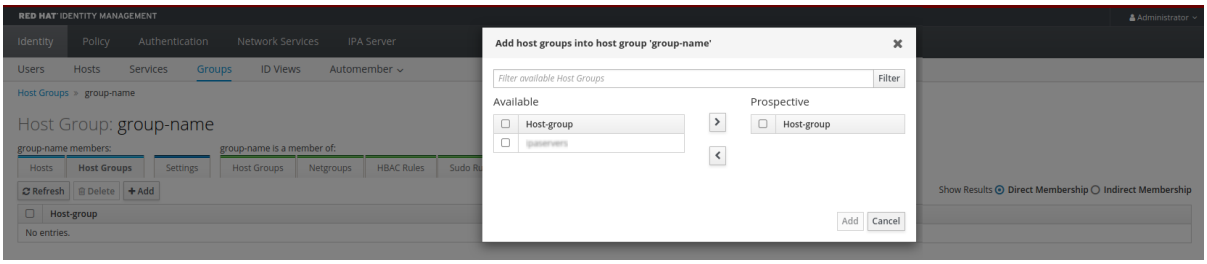
Web インターフェイス (Web UI) を使用して IdM にホストグループメンバーを追加するには、次の手順に従います。

前提条件

- IdM、またはユーザー管理者ロールを管理する管理者権限
- IdM Web UI にログインしている。詳細は、Web ブラウザーでの IdM Web UI へのアクセス を参照してください。

手順

1. **Identity → Groups** をクリックし、**Host Groups** タブを選択します。
2. メンバーを追加するグループの名前をクリックします。
3. 追加するメンバーのタイプに応じて、**ホスト** または **ホストグループ** をクリックします。対応するダイアログが表示されます。
4. 追加するホストまたはホストグループを選択し、› ボタンをクリックして **Prospective** コラムに移動します。
5. **Add** をクリックして確定します。



45.6. IDM WEB UI でのホストグループメンバーの削除

Web インターフェイス (Web UI) を使用して IdM のホストグループメンバーを削除するには、次の手順に従います。

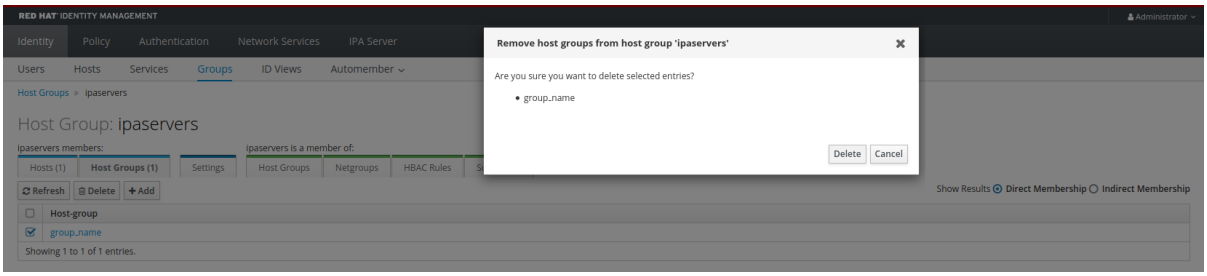
前提条件

- IdM、またはユーザー管理者ロールを管理する管理者権限
- IdM Web UI にログインしている。詳細は、Web ブラウザーでの IdM Web UI へのアクセス を参照してください。

手順

1. **Identity → Groups** をクリックし、**Host Groups** タブを選択します。
2. メンバーを削除するグループの名前をクリックします。

3. 削除するメンバーのタイプに応じて、**ホスト** または **ホストグループ** をクリックします。
4. 削除するメンバーの横にあるチェックボックスを選択します。
5. 削除をクリックします。確認ダイアログが表示されます。



6. Delete をクリックして確定します。選択したメンバーが削除されます。

45.7. WEB UI を使用した IDM ホストグループメンバーマネージャーの追加

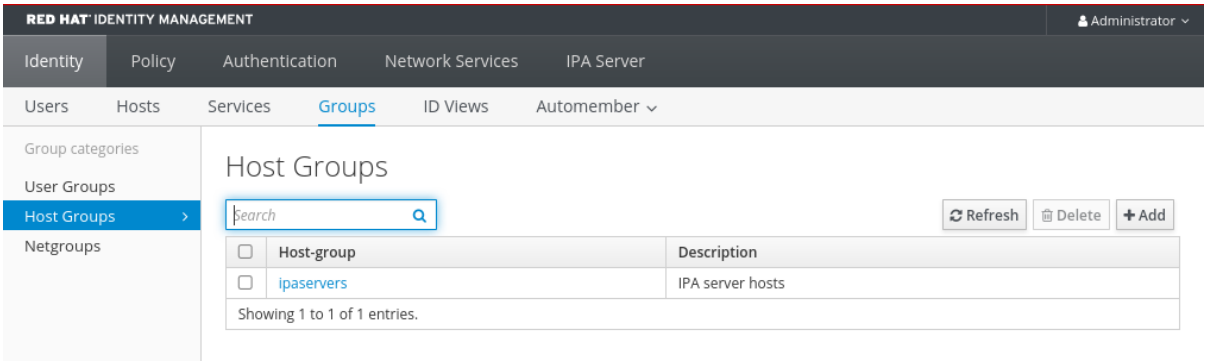
Web インターフェイス (Web UI) を使用して IdM でユーザーまたはユーザーグループをホストグループメンバーマネージャーとして追加するには、次の手順に従います。メンバーマネージャーは、ホストグループのメンバーマネージャーを IdM ホストグループに追加できますが、ホストグループの属性は変更できません。

前提条件

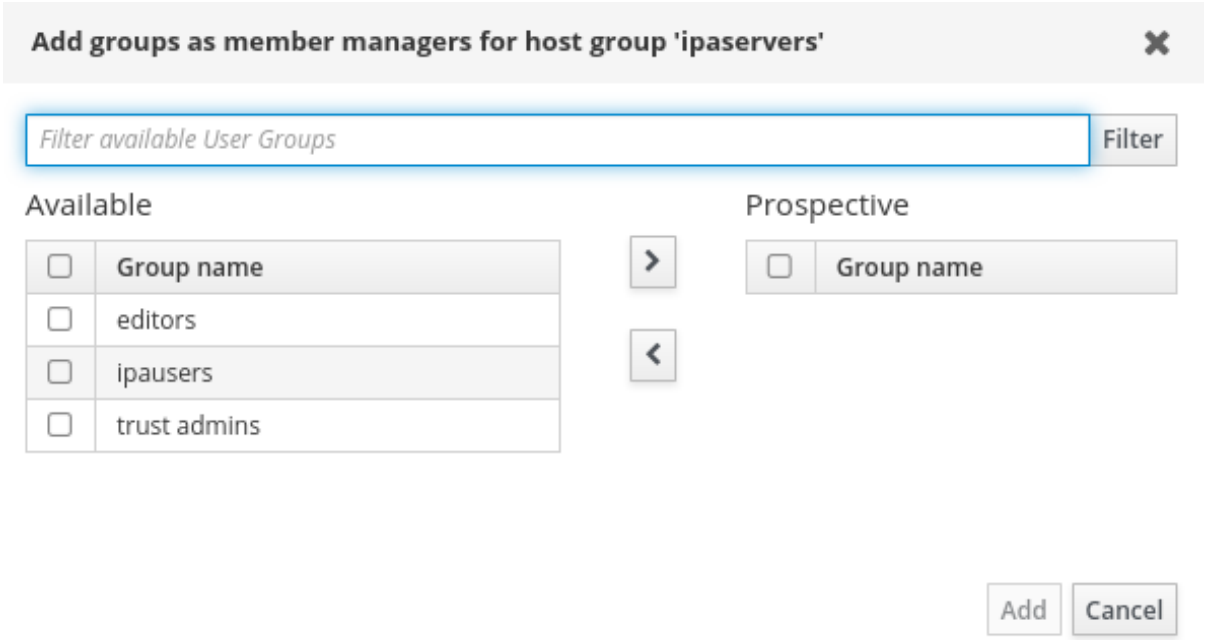
- IdM、またはユーザー管理者ロールを管理する管理者権限
- IdM Web UI にログインしている。詳細は、Web ブラウザーでの IdM Web UI へのアクセス を参照してください。
- メンバーマネージャーとして追加するホストグループ名と、管理するホストグループ名が必要です。

手順

1. **Identity → Groups** をクリックし、**Host Groups** タブを選択します。

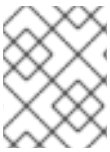


2. メンバーマネージャーを追加するグループの名前をクリックします。
3. 追加するメンバーマネージャーのタイプに合わせて、メンバーマネージャータブの **User Groups** または **Users** をクリックします。対応するダイアログが表示されます。
4. **Add** をクリックします。



5. 追加するユーザーまたはユーザーグループを選択し、> ボタンをクリックして **Prospective** コラムに移動します。

6. **Add** をクリックして確定します。

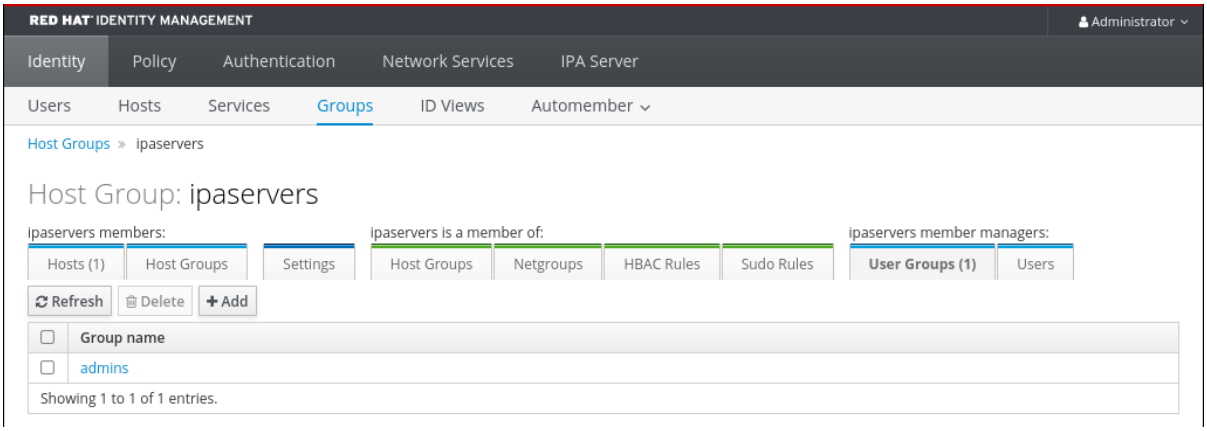


注記

ホストグループにメンバーマネージャーを追加してから、更新が Identity Management 環境のすべてのクライアントに広がるまでに時間がかかる場合があります。

検証手順

- ホストグループダイアログで、ユーザーグループまたはユーザーがグループまたはユーザーのメンバーマネージャーのリストに追加されていることを確認します。



45.8. WEB UI を使用した IDM ホストグループメンバーマネージャーの削除

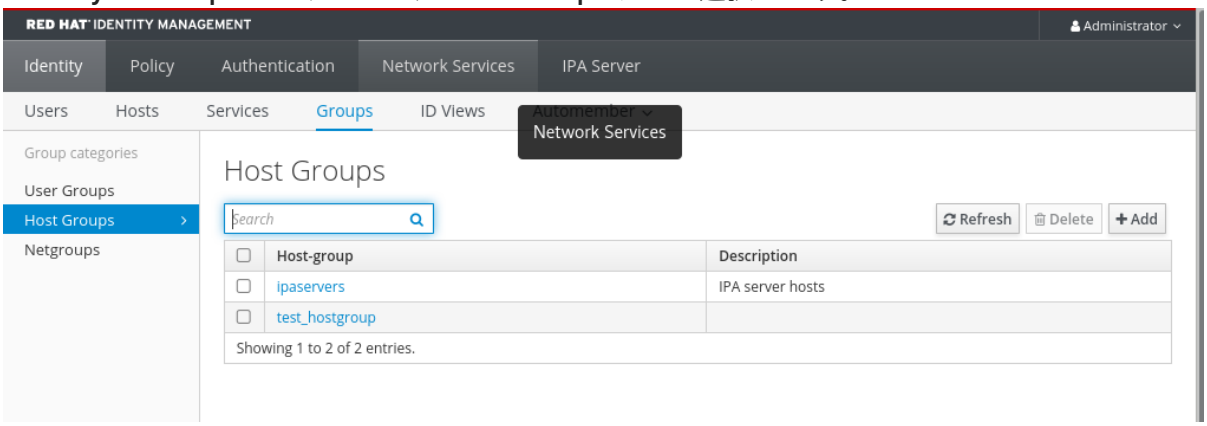
Web インターフェイス (Web UI) を使用して、IdM のホストグループメンバーマネージャーからユーザーまたはユーザーグループを削除するには、次の手順に従います。メンバーマネージャーは、IdM ホストグループからホストグループのメンバーマネージャーを削除できますが、ホストグループの属性は変更できません。

前提条件

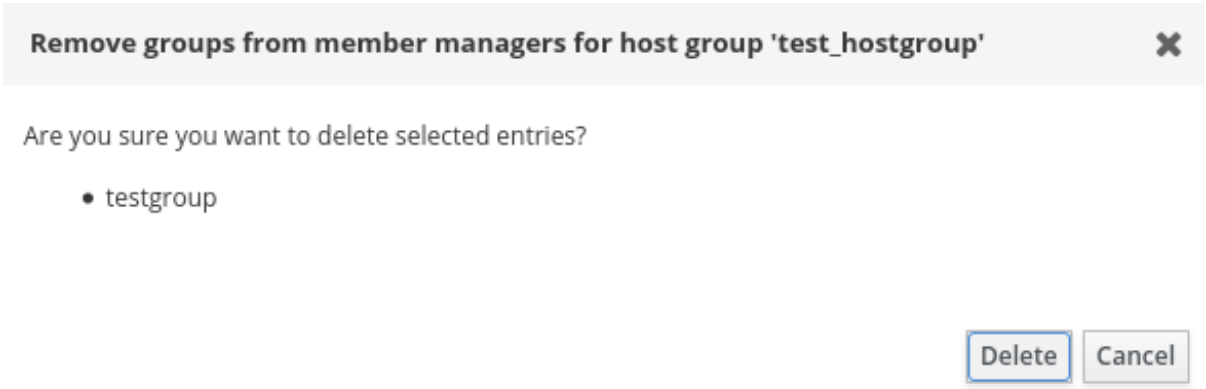
- IdM、またはユーザー管理者ロールを管理する管理者権限
- IdM Web UI にログインしている。詳細は、Web ブラウザーでの IdM Web UI へのアクセス を参照してください。
- 削除する既存のメンバーマネージャーのホストグループ名と、管理するホストグループ名が必要です。

手順

1. **Identity → Groups** をクリックし、**Host Groups** タブを選択します。



2. メンバーマネージャーを削除するグループの名前をクリックします。
3. 削除するメンバーマネージャーのタイプに合わせて、メンバーマネージャータブの **User Groups** または **Users** をクリックします。対応するダイアログが表示されます。
4. 削除するユーザーまたはユーザーグループを選択し、**Delete** をクリックします。
5. **Delete** をクリックして確定します。

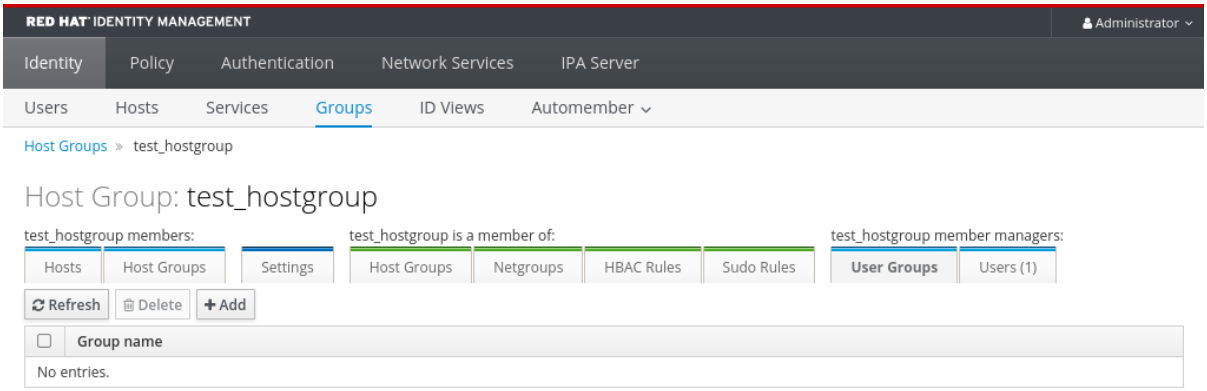


注記

ホストグループからメンバーマネージャーを削除してから、更新が Identity Management 環境のすべてのクライアントに広がるまでに時間がかかる場合があります。

検証手順

- ホストグループダイアログで、ユーザーグループまたはユーザーがグループまたはユーザーのメンバーマネージャーのリストから削除されていることを確認します。


RED HAT IDENTITY MANAGEMENT
Administrator
Identity
Policy
Authentication
Network Services
IPA Server
Users
Hosts
Services
Groups
ID Views
Automember
Host Groups » test_hostgroup
Host Group: test_hostgroup
test_hostgroup members:
Hosts
Host Groups
Settings
test_hostgroup is a member of:
Host Groups
Netgroups
HBAC Rules
Sudo Rules
test_hostgroup member managers:
User Groups
Users (1)
Refresh
Delete
Add
Group name
No entries.

第46章 ANSIBLE PLAYBOOK を使用したホストグループの管理

Identity Management (IdM) のホストグループ と、Ansible を使用して Identity Management (IdM) のホストグループに関係する操作を実行する方法について詳しく説明します。

- IdM のホストグループ
- IdM ホストグループを存在させる手順
- IdM ホストグループにホストを存在させる手順
- IdM ホストグループのネスト化
- IdM ホストグループにメンバーマネージャーを存在させる手順
- IdM ホストグループにホストを存在させないようにする方法
- ネスト化されたホストグループを IdM ホストグループに存在させないようにする方法
- IdM ホストグループにメンバーマネージャーを存在させないようにする方法

46.1. IDM のホストグループ

IdM ホストグループを使用すると、重要な管理タスク (特にアクセス制御) を一元管理できます。

ホストグループの定義

ホストグループは、一般的なアクセス制御ルールやその他の特性を持つ IdM ホストセットが含まれるエンティティーです。たとえば、企業の部門、物理的な場所、またはアクセス制御要件に基づいてホストグループを定義できます。

IdM のホストグループには以下が含まれます。

- IdM サーバーおよびクライアント
- その他の IdM ホストグループ

デフォルトで作成されたホストグループ

デフォルトでは、IdM サーバーは、全 IdM サーバーホストのホストグループ **ipaservers** を作成します。

直接および間接のグループメンバー

IdM のグループ属性は、直接メンバーと間接メンバーの両方に適用されます。ホストグループ B がホストグループ A のメンバーである場合、ホストグループ B のすべてのユーザーはホストグループ A の間接メンバーと見なされます。

46.2. ANSIBLE PLAYBOOK を使用して IDM ホストグループを存在させる手順

以下の手順に従って、Ansible Playbook を使用して、Identity Management (IdM) にホストグループが存在することを確認します。

注記

Ansible を使用しない場合には、**ipa hostgroup-add** コマンドでホストグループエントリーを IdM に作成します。ホストグループを IdM に追加すると、IdM でのホストグループの状態が present になります。Ansible は冪等性に依存しているので、Ansible を使用して IdM にホストグループを追加するには、ホストの状態を Present (**state: present**) として定義した Playbook を作成する必要があります。

前提条件

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

手順

1. **inventory.file** などのインベントリーファイル を作成し、そのファイルで、ターゲットに設定する IdM サーバーのリストと合わせて **ipaserver** を定義します。

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

2. 必要なホストグループ情報を使用して Ansible Playbook ファイルを作成します。たとえば、**databases** という名前のホストグループを存在させるには、**- ipahostgroup** タスクで **name: databases** を指定します。この手順は、**/usr/share/doc/ansible-freeipa/playbooks/user/ensure-hostgroup-is-present.yml** ファイルのサンプルをコピーして変更し、簡素化できます。

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

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  # Ensure host-group databases is present
  - ipahostgroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: databases
      state: present
```

Playbook で **state: present** が指定されていると、IdM にホストグループがない場合のホストグループの追加要求という意味です。

3. Playbook を実行します。

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

検証手順

1. admin として **ipaserver** にログインします。

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

2. admin の Kerberos チケットを要求します。

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

3. IdM に存在させるホストグループに関する情報を表示します。

```
$ ipa hostgroup-show databases
  Host-group: databases
```

データベース ホストグループが IdM に存在します。

46.3. ANSIBLE PLAYBOOK を使用して IDM ホストグループにホストを存在させる手順

以下の手順に従って、Ansible Playbook を使用して、Identity Management (IdM) のホストグループにホストが存在することを確認します。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
  - この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。
- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。
- Ansible Playbook で参照するホストが IdM に存在する。詳細は Ansible Playbook を使用した IdM ホストエントリーの存在の確認 を参照してください。

- Ansible Playbook ファイルから参照するホストグループが IdM に追加されている。詳細は、Ansible Playbook を使用して IdM ホストグループを存在させる手順 を参照してください。

手順

1. **inventory.file** などのインベントリーファイル を作成し、そのファイルで、ターゲットに設定する IdM サーバーのリストと合わせて **ipaserver** を定義します。

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

2. 必要なホスト情報を使用して Ansible Playbook ファイルを作成します。**ipahostgroup** 変数の **name** パラメーターを使用してホストグループの名前を指定します。**ipahostgroup** 変数の **host** パラメーターを使用してホストの名前を指定します。この手順は、**/usr/share/doc/ansible-freeipa/playbooks/hostgroup/ensure-hosts-and-hostgroups-are-present-in-hostgroup.yml** ファイルのサンプルをコピーして変更し、簡素化できます。

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

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  # Ensure host-group databases is present
  - ipahostgroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: databases
      host:
      - db.idm.example.com
      action: member
```

この Playbook は、**db.idm.example.com** ホストを **データベース** ホストグループに追加します。**action: member** の行は、Playbook の実行時に **databases** グループ自体の追加を試行しないことを示します。代わりに、**db.idm.example.com** の **databases** への追加を試行するだけです。

3. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file path_to_playbooks_directory/ensure-hosts-
or-hostgroups-are-present-in-hostgroup.yml
```

検証手順

1. admin として **ipaserver** にログインします。

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

2. admin の Kerberos チケットを要求します。

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

3. ホストグループに関する情報を表示して、どのホストが存在するかを確認します。

```
$ ipa hostgroup-show databases
  Host-group: databases
  Member hosts: db.idm.example.com
```

db.idm.example.com ホストは、**database** ホストグループのメンバーとして存在します。

46.4. ANSIBLE PLAYBOOK を使用した IDM ホストグループのネスト化

以下の手順に従って、Ansible Playbook を使用して、Identity Management (IdM) ホストグループにネスト化されたホストグループが存在することを確認します。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
  - この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。
- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。
- Ansible Playbook ファイルから参照するホストグループが IdM に存在する。詳細は、Ansible Playbook を使用して IdM ホストグループを存在させる手順 を参照してください。

手順

1. **inventory.file** などのインベントリーファイル を作成し、そのファイルで、ターゲットに設定する IdM サーバーのリストと合わせて **ipaserver** を定義します。

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

2. 必要なホストグループ情報を使用して Ansible Playbook ファイルを作成します。ネストされたホストグループ **A** が、Ansible Playbook のホストグループ **B** に存在することを確認するには、**- ipahostgroup** 変数で **name** 変数を使用して、ホストグループ **B** の名前を指定します。**hostgroup** 変数でネスト化されたホストグループ **A** の名前を指定します。この手順は、**/usr/share/doc/ansible-freeipa/playbooks/hostgroup/ensure-hosts-and-hostgroups-are-present-in-hostgroup.yml** ファイルのサンプルをコピーして変更し、簡素化できます。

```
---
```

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

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  # Ensure hosts and hostgroups are present in existing databases hostgroup
  - ipahostgroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: databases
      hostgroup:
      - mysql-server
      - oracle-server
      action: member
```

この Ansible Playbook は、**database** ホストグループに **myqsl-server** および **oracle-server** ホストグループが存在することを確認します。**action: member** 行は、Playbook が実行されると、**databases** グループ自体を IdM に追加しようとはしません。

3. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file path_to_playbooks_directory/ensure-hosts-
or-hostgroups-are-present-in-hostgroup.yml
```

検証手順

1. admin として **ipaserver** にログインします。

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

2. admin の Kerberos チケットを要求します。

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

3. ネスト化されたホストグループが含まれるホストグループに関する情報を表示します。

```
$ ipa hostgroup-show databases
  Host-group: databases
  Member hosts: db.idm.example.com
  Member host-groups: mysql-server, oracle-server
```

mysql-server および **oracle-server** ホストグループは、**databases** ホストグループに存在します。

46.5. ANSIBLE PLAYBOOK を使用して IDM ホストグループにメンバーマネージャーを存在させる手順

以下の手順では、Ansible Playbook を使用して、IdM ホストおよびホストグループにメンバーマネージャーを存在させる方法を説明します。

前提条件

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
  - この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。
- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。
- メンバーマネージャーとして追加するホストまたはホストグループの名前と、管理するホストグループ名が必要です。

手順

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

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

2. 必要なホストおよびホストグループメンバー管理情報を使用して Ansible Playbook ファイルを作成します。

```
---

- name: Playbook to handle host group membership management
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure member manager user example_member is present for group_name
      ipahostgroup:
        ipaadmin_password: "{{ ipaadmin_password }}"
        name: group_name
        membermanager_user: example_member

  - name: Ensure member manager group project_admins is present for group_name
      ipahostgroup:
        ipaadmin_password: "{{ ipaadmin_password }}"
        name: group_name
        membermanager_group: project_admins
```

3. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file path_to_playbooks_directory/add-member-
managers-host-groups.yml
```

検証手順

ipa group-show コマンドを使用して group_name グループのメンバーマネージャーとして example_member と project_admins が含まれていることを確認できます。

1. 管理者として **ipaserver** にログインします。

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

2. **testhostgroup** に関する情報を表示します。

```
ipaserver]$ ipa hostgroup-show group_name
  Host-group: group_name
  Member hosts: server.idm.example.com
  Member host-groups: testhostgroup2
  Membership managed by groups: project_admins
  Membership managed by users: example_member
```

関連情報

- **ipa hostgroup-add-member-manager --help** を参照してください。
- **ipa** の man ページを参照してください。

46.6. ANSIBLE PLAYBOOK を使用して IDM ホストグループにホストを存在させないようにする方法

以下の手順に従って、Ansible Playbook を使用して、Identity Management (IdM) のホストグループにホストがないことを確認します。

前提条件

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

- Ansible Playbook で参照するホストが IdM に存在する。詳細は Ansible Playbook を使用した IdM ホストエントリーの存在の確認 を参照してください。
- Ansible Playbook ファイルから参照するホストグループが IdM に存在する。詳細は、Ansible Playbook を使用して IdM ホストグループを存在させる手順 を参照してください。

手順

1. **inventory.file** などのインベントリーファイル を作成し、そのファイルで、ターゲットに設定する IdM サーバーのリストと合わせて **ipaserver** を定義します。

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

2. 必要なホストおよびホストグループ情報を使用して Ansible Playbook ファイルを作成します。**ipahostgroup** 変数の **name** パラメーターを使用してホストグループの名前を指定します。**ipahostgroup** 変数の **host** パラメーターを使用して、ホストグループに、存在させないようにするホストの名前を指定します。この手順は、**/usr/share/doc/ansible-freeipa/playbooks/hostgroup/ensure-hosts-and-hostgroups-are-absent-in-hostgroup.yml** ファイルのサンプルをコピーして変更し、簡素化できます。

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

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  # Ensure host-group databases is absent
  - ipahostgroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: databases
      host:
      - db.idm.example.com
      action: member
      state: absent
```

この Playbook では、**databases** ホストグループに **db.idm.example.com** ホストを存在させないようにできます。**action: member** の行で、Playbook の実行時に **databases** グループ自体の削除を試行しないように指定します。

3. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file path_to_playbooks_directory/ensure-hosts-
or-hostgroups-are-absent-in-hostgroup.yml
```

検証手順

1. admin として **ipaserver** にログインします。

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

2. admin の Kerberos チケットを要求します。

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

3. ホストグループと、そのホストグループに含まれるホストに関する情報を表示します。

```
$ ipa hostgroup-show databases
  Host-group: databases
  Member host-groups: mysql-server, oracle-server
```

db.idm.example.com ホストは **データベース** ホストグループに存在していません。

46.7. ANSIBLE PLAYBOOK を使用して IDM ホストグループに、ネスト化されたホストグループを存在させないようにする方法

以下の手順に従って、Ansible Playbook を使用して、Identity Management (IdM) の外部ホストグループからネスト化されたホストグループがないことを確認します。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
  - この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。
- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。
- Ansible Playbook ファイルから参照するホストグループが IdM に存在する。詳細は、Ansible Playbook を使用して IdM ホストグループを存在させる手順 を参照してください。

手順

1. **inventory.file** などのインベントリーファイル を作成し、そのファイルで、ターゲットに設定する IdM サーバーのリストと合わせて **ipaserver** を定義します。

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

2. 必要なホストグループ情報を使用して Ansible Playbook ファイルを作成します。**-ipahostgroup** 変数の **name** 変数を使用して、外部ホストグループの名前を指定します。**hostgroup** 変数でネスト化されたホストグループの名前を指定します。この手順は、**/usr/share/doc/ansible-freeipa/playbooks/hostgroup/ensure-hosts-and-hostgroups-are-absent-in-hostgroup.yml** ファイルのサンプルをコピーして変更し、簡素化できます。

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

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  # Ensure hosts and hostgroups are absent in existing databases hostgroup
  - ipahostgroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: databases
      hostgroup:
      - mysql-server
      - oracle-server
      action: member
      state: absent
```

この Playbook は、**mysql-server** および **oracle-server** ホストグループが **databases** ホストグループにないことを確認します。**action: member** 行は、Playbook の実行時に、**databases** グループ自体の IdM からの削除は試行されないようにします。

3. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file path_to_playbooks_directory/ensure-hosts-
or-hostgroups-are-absent-in-hostgroup.yml
```

検証手順

1. admin として **ipaserver** にログインします。

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

2. admin の Kerberos チケットを要求します。

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

3. ネスト化されたホストグループを存在させないホストグループに関する情報を表示します。

```
$ ipa hostgroup-show databases
  Host-group: databases
```

この出力では、**mysql-server** および **oracle-server** のネスト化されたホストグループが、外部 **databases** のホストグループにないことを確認します。

46.8. ANSIBLE PLAYBOOK を使用して IDM ホストグループを存在させないようにする方法

以下の手順に従って、Ansible Playbook を使用して Identity Management (IdM) にホストグループがないことを確認します。

注記

Ansible を使用しない場合には、**ipa hostgroup-del** コマンドでホストグループエントリーを IdM から削除します。IdM からホストグループを削除すると、IdM にホストグループが存在しない状態になります。Ansible は冪等性に依存しているので、Ansible を使用して IdM からホストグループを削除するには、ホストの状態を Absent (**state: absent**) として定義した Playbook を作成する必要があります。

前提条件

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

手順

1. **inventory.file** などのインベントリーファイル を作成し、そのファイルで、ターゲットに設定する IdM サーバーのリストと合わせて **ipaserver** を定義します。

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

2. 必要なホストグループ情報を使用して Ansible Playbook ファイルを作成します。この手順は、**/usr/share/doc/ansible-freeipa/playbooks/user/ensure-hostgroup-is-absent.yml** ファイルのサンプルをコピーして変更し、簡素化できます。

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

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - Ensure host-group databases is absent
    ipahostgroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: databases
      state: absent
```

この Playbook では、IdM から **databases** ホストグループを存在させないようにします。**state: absent** は、IdM からホストグループが削除されていない限り、ホストグループの削除要求を意味します。

3. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file path_to_playbooks_directory/ensure-
hostgroup-is-absent.yml
```

検証手順

1. admin として **ipaserver** にログインします。

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

2. admin の Kerberos チケットを要求します。

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

3. 存在させないようにするホストグループの情報を表示します。

```
$ ipa hostgroup-show databases
ipa: ERROR: databases: host group not found
```

databases ホストグループが IdM に存在しません。

46.9. ANSIBLE PLAYBOOK を使用して IDM ホストグループからホストを存在させないようにする方法

以下の手順では、Ansible Playbook を使用して、IdM ホストおよびホストグループにメンバーマネージャーを存在させないようにする方法を説明します。

前提条件

- IdM 管理者パスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
  - この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。
- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。
- メンバーマネージャーから削除するユーザーまたはユーザーグループの名前と、管理するホストグループの名前が必要です。

手順

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

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

2. 必要なホストおよびホストグループメンバー管理情報を使用して Ansible Playbook ファイルを作成します。

```
---

- name: Playbook to handle host group membership management
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure member manager host and host group members are absent for
group_name
    ipahostgroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: group_name
      membermanager_user: example_member
      membermanager_group: project_admins
      action: member
      state: absent
```

3. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
path_to_inventory_directory/inventory.file path_to_playbooks_directory/ensure-
member-managers-host-groups-are-absent.yml
```

検証手順

ipa group-show コマンドを使用して、**group_name** グループに **example_member** または **project_admins** がメンバーマネージャーとして含まれているかどうかを確認できます。

1. 管理者として **ipaserver** にログインします。

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

2. **testhostgroup** に関する情報を表示します。

```
ipaserver]$ ipa hostgroup-show group_name
  Host-group: group_name
  Member hosts: server.idm.example.com
  Member host-groups: testhostgroup2
```

関連情報

- **ipa hostgroup-add-member-manager --help** を参照してください。
- **ipa** の man ページを参照してください。

第47章 ホストベースのアクセス制御ルールの設定

ホストベースのアクセス制御 (HBAC) ルールを使用して、Identity Management (IdM) ドメインのアクセス制御を管理できます。HBAC ルールは、どのユーザーまたはユーザーグループが、どのサービスまたはサービスグループ内のサービスを使用して、指定されたホストまたはホストグループにアクセスできるかを定義します。たとえば、HBAC ルールを使用して次の目的を達成できます。

- 指定のユーザーグループのメンバーだけがドメイン内の特定のシステムにアクセスできるように制限する。
- ドメイン内のシステムにアクセスするために特定のサービスのみを使用できます。

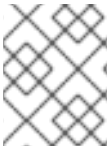
デフォルトでは、IdM は **allow_all** という名前のデフォルトの HBAC ルールで設定されます。この設定では、IdM ドメイン全体の関連するサービスをどれでも使用して、すべてのユーザーがすべてのホストに普遍的にアクセスできます。

デフォルトの **allow_all** ルールを独自の HBAC ルールセットに置き換えることで、さまざまなホストへのアクセスを微調整できます。個別のユーザー、ホスト、またはサービスではなく、ユーザーグループ、ホストグループ、またはサービスグループに HBAC ルールを適用して、アクセス制御管理を集中化および簡素化できます。

47.1. WEBUI を使用した IDM ドメインでの HBAC ルールの設定

ホストベースのアクセス制御用にドメインを設定するには、次の手順を実行します。

1. IdM WebUI で HBAC ルールを作成 します。
2. 新しい HBAC ルールをテスト します。
3. デフォルトの **allow_all** HBAC ルールを無効化 します。



注記

カスタム HBAC ルールを作成する前に **allow_all** ルールを無効にしないでください。無効にすると、どのユーザーもホストにアクセスできなくなります。

47.1.1. IdM WebUI での HBAC ルールの作成

IdM WebUI を使用してホストベースのアクセス制御用にドメインを設定するには、以下の手順に従います。この例では、任意のサービスを使用してドメイン内のすべてのシステムにアクセスする権限を、1 人のユーザー **sysadmin** に付与する方法を説明します。

注記

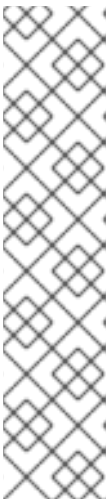
IdM は、ユーザーのプライマリーグループを、IdM グループオブジェクトへのリンクの代わりに、**gidNumber** 属性の数値として保存します。このため、HBAC ルールで参照できるのはユーザーの補助グループだけで、プライマリーグループは参照できません。

前提条件

- ユーザー **sysadmin** が IdM に存在する。

手順

1. **Policy** > **Host-Based Access Control** > **HBAC Rules**を選択します。
2. **Add** をクリックして、新規ルールの追加を開始します。
3. ルールの名前を入力し、**Add and Edit** をクリックして HBAC ルール設定ページを開きます。
4. **Who** エリアで、**Specified Users and Groups** を選択します。次に、**Add** をクリックしてユーザーまたはグループを追加します。
5. **Available** ユーザーのリストから **sysadmin** ユーザーを選択し、**>** をクリックして **Prospective** ユーザーのリストに移動し、**Add** をクリックします。
6. **Accessing** エリアで **Any Host** を選択して、HBAC ルールをすべてのホストに適用します。
7. **Via Service** エリアで **Any Service** を選択して、HBAC ルールをすべてのサービスに適用します。



注記

主なサービスとサービスグループのみが、デフォルトで HBAC ルール用に設定されます。

- 現在利用可能なサービスのリストを表示するには、**Policy** > **Host-Based Access Control** > **HBAC Services** を選択します。
- 現在利用可能なサービスグループのリストを表示するには、**Policy** > **Host-Based Access Control** > **HBAC Service Groups** を選択します。

さらにサービスとサービスグループを追加するには、カスタム HBAC サービス用の HBAC サービスエントリーの追加 および HBAC サービスグループの追加 を参照してください。

8. **HBAC ルール** 設定ページで行った変更を保存するには、ページの上部にある **Save** をクリックします。

47.1.2. IdM WebUI での HBAC ルールのテスト

IdM では、シミュレートシナリオを使用して、さまざまな状況で HBAC 設定をテストできます。シミュレートしたテストを実行することで、実稼働環境に HBAC ルールをデプロイする前に、設定ミスやセキュリティーリスクを見つけることができます。

重要

実稼働環境で使用する前に、カスタム HBAC ルールを常にテストしてください。

IdM では、信頼された Active Directory (AD) ユーザーに対する HBAC ルールの影響については検証されない点に注意してください。IdM LDAP ディレクトリーには AD データが保存されないため、IdM は、HBAC シナリオをシミュレートするときに AD ユーザーのグループメンバーシップを解決できません。

手順

1. **Policy** > **Host-Based Access Control** > **HBAC Test**を選択します。

2. Who ウィンドウで、テスト実行時に使用するアイデンティティーを持つユーザーを指定し、**Next** をクリックします。

3. Accessing ウィンドウで、ユーザーがアクセスするホストを指定し、**Next** をクリックします。

4. Via Service ウィンドウで、ユーザーが使用するサービスを指定し、**Next** をクリックします。

5. Rules ウィンドウで、テストする HBAC ルールを選択し、**Next** をクリックします。ルールを選択しない場合は、すべてのルールがテストされます。
Include Enabled を選択すると、ステータスが **Enabled** であるすべてのルールでテストを実行します。Include Disabled を選択すると、ステータスが **Disabled** であるすべてのルールでテストを実行します。HBAC ルールのステータスを表示および変更するには、**Policy > Host-Based Access Control > HBAC Rules** を選択します。

重要

複数のルールでテストを実行した場合、選択したルールの少なくとも 1 つがアクセスを許可していれば成功となります。

6. Run Test ウィンドウで、**Run Test** をクリックします。

7. テスト結果を確認します。

   - ACCESS DENIED と表示された場合、テストでユーザーのアクセスが許可されていないことを示しています。
   - ACCESS GRANTED と表示された場合、ユーザーはホストに正常にアクセスできることを示しています。

   デフォルトでは、IdM はテスト結果を表示する際に、テストされている HBAC ルールをすべてリスト表示します。

   - Matched を選択すると、正常なアクセスを許可したルールが表示されます。
   - Unmatched を選択すると、アクセスを阻止したルールが表示されます。

47.1.3. IdM WebUI での HBAC ルールの無効化

HBAC ルールを無効にすることはできますが、ルールは非アクティブ化されるだけで、削除されません。HBAC ルールを無効にした場合は、後で再度有効にできます。

注記

カスタム HBAC ルールを初めて設定する場合は、HBAC ルールを無効にすると便利です。新しい設定がデフォルトの **low_all** HBAC ルールで上書きされないようにするには、**low_all** を無効にする必要があります。

手順

1. **Policy > Host-Based Access Control > HBAC Rules**を選択します。

2. 無効にする HBAC ルールを選択します。

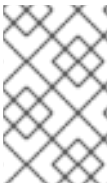
3. **Disable** をクリックします。

4. **OK** をクリックして、選択した HBAC ルールを無効にすることを確認します。

47.2. CLI を使用した IDM ドメインでの HBAC ルールの設定

ホストベースのアクセス制御用にドメインを設定するには、次の手順を実行します。

1. IdM CLI で HBAC ルールを作成 します。
2. 新しい HBAC ルールをテスト します。
3. デフォルトの **allow_all** HBAC ルールを無効化 します。

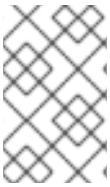


注記

カスタム HBAC ルールを作成する前に、**allow_all** ルールを無効にしないでください。カスタムルールを作成する前にこれを無効にすると、すべてのユーザーのすべてのホストへのアクセスが拒否されます。

47.2.1. IdM CLI での HBAC ルールの作成

IdM CLI を使用してホストベースのアクセス制御用にドメインを設定するには、以下の手順に従います。この例では、任意のサービスを使用してドメイン内のすべてのシステムにアクセスする権限を、1 人のユーザー **sysadmin** に付与する方法を説明します。



注記

IdM は、ユーザーのプライマリーグループを、IdM グループオブジェクトへのリンクの代わりに、**gidNumber** 属性の数値として保存します。このため、HBAC ルールで参照できるのはユーザーの補助グループだけで、プライマリーグループは参照できません。

前提条件

- ユーザー **sysadmin** が IdM に存在する。

手順

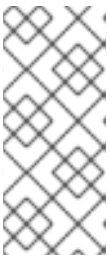
1. **ipa hbacrule-add** コマンドを使用して、ルールを追加します。

```
$ ipa hbacrule-add
Rule name: rule_name
----------------------------
Added HBAC rule "rule_name"
----------------------------
  Rule name: rule_name
  Enabled: TRUE
```

2. **sysadmin** ユーザーのみに HBAC ルールを適用するには、 **ipa hbacrule-add-user** コマンドを使用します。

```
$ ipa hbacrule-add-user --users=sysadmin
Rule name: rule_name
  Rule name: rule_name
  Enabled: True
  Users: sysadmin
```

```
-------------------------
Number of members added 1
-------------------------
```

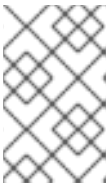


注記

すべてのユーザーに HBAC ルールを適用するには、**ipa hbacrule-mod** コマンドを使用して、all ユーザーカテゴリー **--usercat=all** を指定します。HBAC ルールが個々のユーザーまたはグループに関連付けられていると、**ipa hbacrule-mod --usercat=all** は失敗します。この場合は、**ipa hbacrule-remove-user** コマンドを使用して、ユーザーとグループを削除します。

3. ターゲットホストを指定します。すべてのホストに HBAC ルールを適用するには、**ipa hbacrule-mod** コマンドを使用して、all ホストカテゴリーを指定します。

```
$ ipa hbacrule-mod rule_name --hostcat=all
------------------------------
Modified HBAC rule "rule_name"
------------------------------
  Rule name: rule_name
  Host category: all
  Enabled: TRUE
  Users: sysadmin
```

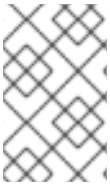


注記

HBAC ルールが個々のホストまたはグループに関連付けられていると、**ipa hbacrule-mod --hostcat=all** は失敗します。この場合は、**ipa hbacrule-remove-host** コマンドを使用して、ホストとグループを削除します。

4. ターゲットの HBAC サービスを指定します。すべてのサービスに HBAC ルールを適用するには、**ipa hbacrule-mod** コマンドを使用して、all サービスカテゴリーを指定します。

```
$ ipa hbacrule-mod rule_name --servicecat=all
------------------------------
Modified HBAC rule "rule_name"
------------------------------
  Rule name: rule_name
  Host category: all
  Service category: all
  Enabled: True
  Users: sysadmin
```



注記

HBAC ルールが個々のサービスまたはグループに関連付けられていると、**ipa hbacrule-mod --servicecat=all** は失敗します。この場合は、**ipa hbacrule-remove-service** コマンドを使用して、サービスとグループを削除します。

検証

- HBAC ルールが正しく追加されたことを確認します。

a. **ipa hbacrule-find** コマンドを使用して、HBAC ルールが IdM に存在することを確認します。

b. **ipa hbacrule-show** コマンドを使用して、HBAC ルールのプロパティーを確認します。

関連情報

- 詳細は、ipa hbacrule-add --help を参照してください。
- カスタム HBAC サービス用の HBAC サービスエントリーの追加 を参照してください。
- HBAC サービスグループの追加 を参照してください。

47.2.2. IdM CLI での HBAC ルールのテスト

IdM では、シミュレートシナリオを使用して、さまざまな状況で HBAC 設定をテストできます。シミュレートしたテストを実行することで、実稼働環境に HBAC ルールをデプロイする前に、設定ミスやセキュリティーリスクを見つけることができます。

実稼働環境で使用する前に、カスタム HBAC ルールを常にテストしてください。

IdM では、信頼された Active Directory (AD) ユーザーに対する HBAC ルールの影響については検証されない点に注意してください。IdM LDAP ディレクトリーには AD データが保存されないため、IdM は、HBAC シナリオをシミュレートするときに AD ユーザーのグループメンバーシップを解決できません。

手順

1. **ipa hbactest** コマンドを使用して、HBAC ルールをテストします。1 つの HBAC ルールをテストする方法と、複数の HBAC ルールをテストする方法があります。

   - 1 つの HBAC ルールをテストするには、以下を実行します。

     ```
     $ ipa hbactest --user=sysadmin --host=server.idm.example.com --service=sudo --
     rules=rule_name
     ----------------------
     Access granted: True
     ----------------------
       Matched rules: rule_name
     ```

   - 複数の HBAC ルールをテストするには、以下を実行します。

     a. **sysadmin** にすべてのホストで **ssh** の使用のみを許可する 2 番目のルールを追加します。

     ```
     $ ipa hbacrule-add --hostcat=all rule2_name
     $ ipa hbacrule-add-user --users sysadmin rule2_name
     $ ipa hbacrule-add-service --hbacsvcs=sshd rule2_name
       Rule name: rule2_name
       Host category: all
       Enabled: True
       Users: admin
       HBAC Services: sshd
     -------------------------
     Number of members added 1
     -------------------------
     ```

b. 次のコマンドを実行して、複数の HBAC ルールをテストします。

```
$ ipa hbactest --user=sysadmin --host=server.idm.example.com --service=sudo --
rules=rule_name --rules=rule2_name
--------------------
Access granted: True
--------------------
  Matched rules: rule_name
  Not matched rules: rule2_name
```

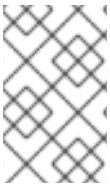
出力において、**Matched rules** には正常なアクセスを許可したルールがリスト表示され、**Not matched rules** にはアクセスを妨げたルールがリスト表示されます。**--rules** オプションを指定しない場合は、すべてのルールが適用されることに注意してください。**--rules** を使用すると、各ルールを個別にテストするのに役立ちます。

関連情報

- 詳細は、**ipa hbactest --help** を参照してください。

47.2.3. IdM CLI での HBAC ルールの無効化

HBAC ルールを無効にすることはできますが、ルールは非アクティブ化されるだけで、削除されません。HBAC ルールを無効にした場合は、後で再度有効にできます。



注記

カスタム HBAC ルールを初めて設定する場合は、HBAC ルールを無効にすると便利です。新しい設定がデフォルトの **low_all** HBAC ルールで上書きされないようにするには、**low_all** を無効にする必要があります。

手順

- **ipa hbacrule-disable** コマンドを使用します。たとえば、**allow_all** ルールを無効にするには、次のコマンドを実行します。

```
$ ipa hbacrule-disable allow_all
-----------------------------
Disabled HBAC rule "allow_all"
-----------------------------
```

関連情報

- 詳細は、**ipa hbacrule-disable --help** を参照してください。

47.3. カスタム HBAC サービス用の HBAC サービスエントリーの追加

主なサービスとサービスグループは、デフォルトで HBAC ルール用に設定されますが、その他のプラグ可能な認証モジュール (PAM) サービスを HBAC サービスとして設定することもできます。これにより、HBAC ルールでカスタム PAM サービスを定義できるようになります。これらの PAM サービスファイルは、RHEL システムの **etc/pam.d** ディレクトリーにあります。

注記

サービスの HBAC サービスとしての追加と、ドメインへのサービスの追加は同じではありません。サービスをドメインに追加すると、ドメイン内の他のリソースでそのサービスを使用できるようにはなりますが、HBAC ルールで使用できるようにはなりません。

47.3.1. IdM WebUI でのカスタム HBAC サービス用の HBAC サービスエントリーの追加

カスタム HBAC サービスエントリーを追加するには、以下で説明する手順に従います。

手順

1. **Policy > Host-Based Access Control > HBAC Services**を選択します。
2. **Add** をクリックして HBAC サービスエントリーを追加します。
3. サービスの名前を入力し、**Add** をクリックします。

47.3.2. IdM CLI でのカスタム HBAC サービスの HBAC サービスエントリーの追加

カスタム HBAC サービスエントリーを追加するには、以下で説明する手順に従います。

手順

- **ipa hbacsvc-add** コマンドを使用します。たとえば、**tftp** サービスのエントリーを追加するには、次のコマンドを実行します。

```
$ ipa hbacsvc-add tftp
-------------------------
Added HBAC service "tftp"
-------------------------
  Service name: tftp
```

関連情報

- 詳細は、**ipa hbacsvc-add --help** を参照してください。

47.4. HBAC サービスグループの追加

HBAC サービスグループにより、HBAC ルールの管理を簡素化できます。たとえば、個々のサービスを HBAC ルールに追加する代わりに、サービスグループ全体を追加できます。

47.4.1. IdM WebUI での HBAC サービスグループの追加

IdM WebUI で HBAC サービスグループを追加するには、以下に概説する手順に従います。

手順

1. **Policy > Host-Based Access Control > HBAC Service Groups**を選択します。
2. **Add** をクリックして HBAC サービスグループを追加します。
3. サービスグループの名前を入力し、**Edit** をクリックします。

4. サービスグループ設定ページで **追加** を選択し、HBAC サービスをグループのメンバーとして追加します。

47.4.2. IdM CLI での HBAC サービスグループの追加

IdM CLI で HBAC サービスグループを追加するには、以下に概説する手順に従います。

手順

1. ターミナルで **ipa hbacsvcgroup-add** コマンドを使用して、HBAC サービスグループを追加します。たとえば、**login** という名前のグループを追加するには、次のコマンドを実行します。

```
$ ipa hbacsvcgroup-add
Service group name: login
------------------------------
Added HBAC service group "login"
------------------------------
  Service group name: login
```

2. **ipa hbacsvcgroup-add-member** コマンドを使用して、HBAC サービスをグループのメンバーとして追加します。たとえば、**sshd** サービスを **login** グループに追加するには、次のコマンドを実行します。

```
$ ipa hbacsvcgroup-add-member
Service group name: login
[member HBAC service]: sshd
  Service group name: login
  Member HBAC service: sshd
------------------------
Number of members added 1
------------------------
```

関連情報

- 詳細は、**ipa hbacsvcgroup-add --help** を参照してください。
- 詳細は、**ipa hbacsvcgroup-add-member --help** を参照してください。

第48章 ANSIBLE PLAYBOOK を使用して IDM にホストベースのアクセス制御ルールを存在させる手順

Ansible は、システムの設定、ソフトウェアのデプロイ、ローリング更新の実行に使用する自動化ツールです。これには、Identity Management (IdM) のサポートが含まれます。

Identity Management (IdM) ホストベースのアクセスポリシーと、Ansible を使用してそれを定義する方法を説明します。

48.1. IDM のホストベースのアクセス制御ルール

ホストベースのアクセス制御 (HBAC) ルールは、サービスグループ内のサービスを使用して、どのユーザーまたはグループがどのホストまたはホストグループにアクセスできるかを定義します。システム管理者は、HBAC ルールを使用して以下の目的を達成できます。

- 指定のユーザーグループのメンバーだけがドメイン内の特定のシステムにアクセスできるように制限する。
- ドメイン内のシステムにアクセスする時に特定のサービスだけを使用できるようにする。

デフォルトでは、IdM は **allow_all** という名前のデフォルトの HBAC ルールで設定されます。この設定では、どのユーザーでも関連のあるサービスをどれでも使用して IdM ドメイン全体にあるすべてのホストに普遍的にアクセスできます。

デフォルトの **allow_all** ルールを独自の HBAC ルールセットに置き換えることで、さまざまなホストへのアクセスを微調整できます。個別のユーザー、ホスト、またはサービスではなく、ユーザーグループ、ホストグループ、またはサービスグループに HBAC ルールを適用して、アクセス制御管理を集中化および簡素化できます。

48.2. ANSIBLE PLAYBOOK を使用して IDM に HBAC ルールを存在させる手順

以下の手順に従って、Ansible Playbook を使用して Identity Management (IdM) にホストベースのアクセス制御 (HBAC) ルールが存在することを確認します。

前提条件

- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
  - この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。
- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。

- HBAC ルールに使用するユーザーとユーザーグループが IdM に存在する。詳細は、Ansible Playbook を使用したユーザーアカウントの管理 および Ansible Playbook を使用した IdM グループおよびグループメンバーの存在の確保 を参照してください。
- HBAC ルールを適用するホストおよびホストグループが IdM に存在する。詳細は、Ansible Playbook を使用したホストの管理 および Ansible Playbook を使用したホストグループの管理 を参照してください。

手順

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

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

2. Ansible Playbook ファイルを作成して、存在させる HBAC ポリシーを定義します。この手順は、**/usr/share/doc/ansible-freeipa/playbooks/hbacrule/ensure-hbacrule-allhosts-present.yml** ファイルのサンプルをコピーして変更し、簡素化できます。

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

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  # Ensure idm_user can access client.idm.example.com via the sshd service
  - ipahbacrule:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: login
      user: idm_user
      host: client.idm.example.com
      hbacsvc:
      - sshd
      state: present
```

3. Playbook を実行します。

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

検証手順

1. 管理者として IdM Web UI にログインします。
2. **Policy → Host-Based-Access-Control → HBAC Test** の順に選択します。
3. **Who** タブで idm_user を選択します。
4. **Accessing** タブで **client.idm.example.com** を選択します。
5. **Via service** タブで **sshd** を選択します。

6. **Rules** タブで **login** を選択します。

7. **Run test** タブで **Run test** ボタンをクリックします。ACCESS GRANTED が表示されると、HBAC ルールが正常に実装されています。

関連情報

- **/usr/share/doc/ansible-freeipa** ディレクトリーの **README-hbacsvc.md** ファイル、**README-hbacsvcgroup.md** ファイル、および **README-hbacrule.md** ファイルを参照してください。

- **/usr/share/doc/ansible-freeipa/playbooks** ディレクトリーのサブディレクトリーにある Playbook を参照してください。

第49章 ユーザーとホストの SSH 公開鍵の管理

SSH (Secure Shell) は、クライアント/サーバーアーキテクチャーを使用して 2 つのシステム間で安全な通信を提供するプロトコルです。SSH を使用すると、ユーザーがサーバーホストシステムにリモートでログインできるようになるほか、あるホストマシンが別のマシンにアクセスできるようになります。

49.1. SSH 鍵の形式

IdM では、以下の 2 つの SSH 鍵形式を使用できます。

- OpenSSH-style key
- Raw RFC 4253-style key

IdM は、IdM の LDAP サーバーに保存する前に、RFC 4253 形式の鍵を OpenSSH スタイルの鍵に自動的に変換することに注意してください。

IdM サーバーは、アップロードされたキーブロブから、RSA または DSA キーといったキーのタイプを識別できます。~/**.ssh/known_hosts** などの鍵ファイルでは、鍵のエントリーは、サーバーのホスト名と IP アドレス、鍵タイプ、および鍵自体によって識別されます。以下に例を示します。

```
host.example.com,1.2.3.4 ssh-rsa AAA...ZZZ==
```

これは、要素が **type key== comment** の順序で含まれるユーザーの公開鍵エントリーとは異なります。

```
"ssh-rsa ABCD1234...== ipaclient.example.com"
```

id_rsa.pub などの鍵ファイルは、鍵タイプ、鍵、追加のコメントまたは識別子の 3 つの部分で構成されます。鍵を IdM にアップロードする場合は、3 つの鍵の部分すべてをアップロードすることも、鍵のみをアップロードすることもできます。鍵をアップロードした場合、IdM は、アップロードした鍵から RSA や DSA などの鍵タイプを自動的に識別します。

~/**.ssh/known_hosts** ファイルのホスト公開鍵のエントリーを使用する場合は、ユーザーの鍵の形式 **key== comment** と一致するように順序を変更する必要があります。

```
ssh-rsa AAA...ZZZ== host.example.com,1.2.3.4
```

IdM は、公開鍵の内容から鍵タイプを自動的に決定できます。個々の鍵を簡単に識別できるようにするためのコメントはオプションです。必須要素は、公開鍵ブロブだけです。

IdM は、次の OpenSSH スタイルのファイルに保存されている公開鍵を使用します。

- ホストの公開鍵は **known_hosts** ファイルにあります。
- ユーザーの公開鍵は **authorized_keys** ファイルにあります。

関連情報

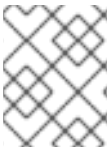
- RFC 4716 を参照してください。
- RFC 4253 を参照してください。

49.2. IDM と OPENSSH

IdM サーバーまたはクライアントのインストール時に、インストールスクリプトの一部として以下が行われます。

- OpenSSH サーバーとクライアントが IdM クライアントマシン上に設定されます。
- SSSD が、ユーザーおよびホストの SSH 鍵をキャッシュに保存および取得するように設定されます。これにより、IdM は SSH 鍵の汎用および集中化されたリポジトリーとして機能できます。

クライアントインストール時に SSH サービスを有効にすると、SSH サービスの初回起動時に RSA 鍵が作成されます。



注記

ipa-client-install インストールスクリプトを実行してマシンを IdM クライアントとして追加すると、クライアントは 2 つの SSH 鍵 (RSA と DSA) を使用して作成されます。

インストールの一環として、以下を設定できます。

- **--ssh-trust-dns** オプションを使用して、鍵のフィンガープリントが保存されている IdM DNS レコードを自動的に信頼するように OpenSSH を設定する。
- OpenSSH を無効にし、インストールスクリプトが **--no-sshd** オプションを使用して OpenSSH サーバーを設定しないようにする。
- **--no-dns-sshfp** オプションを使用して、ホストが独自の DNS エントリーを含む DNS SSHFP レコードを作成できないようにする。

インストール時にサーバーまたはクライアントを設定しない場合は、後で SSSD を手動で設定できます。SSSD を手動で設定する方法については、OpenSSH サービスのキャッシュを提供するように SSSD を設定 を参照してください。SSSD による SSH 鍵のキャッシュには、ローカルマシンでの管理権限が必要です。

49.3. SSH 鍵の生成

SSH 鍵は、OpenSSH の **ssh-keygen** ユーティリティーを使用して生成できます。

手順

1. RSA SSH 鍵を生成するには、次のコマンドを実行します。

```
$ ssh-keygen -t rsa -C user@example.com
Generating public/private rsa key pair.
```

ホスト鍵を生成する場合は、user@example.com を必要なホスト名 (**server.example.com,1.2.3.4** など) に置き換えてください。

2. 鍵を保存するファイルを指定するか、Enter キーを押して表示されたデフォルトの場所をそのまま使用します。

```
Enter file in which to save the key (/home/user/.ssh/id_rsa):
```

ホスト鍵を生成する場合は、既存の鍵を上書きしないように、ユーザーの ~/**.ssh**/ ディレクトリーとは異なる場所に鍵を保存してください。たとえば、/**home**/**user**/**.ssh**/**host_keys** です。

3. 秘密鍵のパスフレーズを指定するか、Enter キーを押してパスフレーズを空白のままにします。

```
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/user/.ssh/id_rsa.
Your public key has been saved in /home/user/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:ONxjcMX7hJ5zly8F8ID9fpbqcuxQK+ylVLKDMsJPxGA user4@example.com
The key's randomart image is:
+---[RSA 3072]----+
|         ..o     |
|         .o +    |
|     E. .  o =   |
|    ..o=  o . +  |
|      +oS. = + o.|
|    . .o .* B =.+|
|     o + . X.+.= |
|      + o o.*+. .|
|       .   o=o . |
+----[SHA256]-----+
```

この SSH 鍵をアップロードするには、表示されたファイルに保存されている公開鍵文字列を使用します。

49.4. ホストの SSH 公開鍵の管理

OpenSSH は公開鍵を使用してホストを認証します。あるマシンが別のマシンにアクセスを試みてキーのペアを提示します。ホストの初回認証時には、ターゲットマシンの管理者は、この要求を手動で認証する必要があります。次に、マシンはホストの公開鍵を **known_hosts** ファイルに保存します。リモートのマシンがターゲットマシンにアクセスを再度試みると、ターゲットマシンは **known_hosts** ファイルをチェックして、認証済みホストに自動的にアクセスを許可します。

49.4.1. IdM Web UI を使用したホストの SSH 鍵のアップロード

Identity Management を使用すると、SSH 公開鍵をホストエントリーにアップロードできます。OpenSSH は公開鍵を使用してホストを認証します。

前提条件

- IdM Web UI、またはユーザー管理者ロールを管理する管理者権限

手順

1. ホストの鍵は ~/**.ssh**/**known_hosts** ファイルから取得できます。以下に例を示します。

```
server.example.com,1.2.3.4 ssh-rsa
AAAAB3NzaC1yc2EAAAABIwAAAQEApvjBvSFSkTU0WQW4eOweeo0DZZ08F9Ud21xlLy6F
OhzwpXFGIyxvXZ52+siHBHbbqGL5+14N7UvElruyslIHx9LYUR/pPKSMXCGyboLy5aTNl5OQ5
EHwrhVnFDIKXkvp45945R7SKYCUtRumm0Iw6wq0XD4o+ILeVbV3wmcB1bXs36ZvC/M6riefn
9PcJmh6vNCvIsbMY6S+FhkWUTTiOXJjUDYRLlwM273FfWhzHK+SSQXeBp/zIn1gFvJhSZMR
i9HZpDoqxLbBB9QIdIw6U4MIjNmKsSI/ASpkFm2GuQ7ZK9KuMItY2AoCuIRmRAdF8iYNHBT
XNfFurGogXwRDjQ==
```

ホスト鍵を生成することもできます。SSH 鍵の生成 を参照してください。

2. 公開鍵をキーファイルからコピーします。完全な鍵エントリーは、**hostname,IP type key==** の形式です。**key==** のみが必要ですが、エントリー全体を保存することもできます。エントリーの全要素を使用するには、エントリーを再編成して、順番が **type key== [host name,IP]** になるように設定します。

```
cat /home/user/.ssh/host_keys.pub
ssh-rsa AAAAB3NzaC1yc2E...tJG1PK2Mq++wQ== server.example.com,1.2.3.4
```

3. IdM Web UI にログインします。
4. **Identity > Hosts** タブに移動します。
5. 編集するホスト名をクリックします。
6. **Host Settings** セクションで、SSH 公開鍵の **Add** ボタンをクリックします。
7. ホストの公開鍵を **SSH public key** フィールドに貼り付けます。
8. **Set** をクリックします。
9. IdM Web UI ウィンドウの上部にある **Save** をクリックします。

検証

- **Hosts Settings** セクションで、鍵が **SSH public keys** の下にリストされていることを確認します。

49.4.2. IdM CLI を使用したホストの SSH 鍵のアップロード

Identity Management を使用すると、SSH 公開鍵をホストエントリーにアップロードできます。OpenSSH は公開鍵を使用してホストを認証します。ホスト SSH 鍵は、**host-add** を使用してホストを作成するときか、エントリーを後で修正するときに、IdM のホストエントリーに追加されます。

インストールスクリプトで SSH サービスが明示的に無効にされなければ、**ipa-client-install** コマンドで RSA と DSA ホスト鍵が作成されます。

前提条件

- IdM、またはユーザー管理者ロールを管理する管理者権限

手順

1. **--sshpubkey** オプションを指定して **host-mod** コマンドを実行し、base64 にエンコードされた公開鍵をホストエントリーにアップロードします。
ホスト鍵を追加すると、ホストの DNS Secure Shell フィンガープリント (SSHFP) レコードが変更されるため、**--updatedns** オプションを使用してホストの DNS エントリーを更新します。以下に例を示します。

```
$ ipa host-mod --sshpubkey="ssh-rsa RjlzYQo==" --updatedns host1.example.com
```

実際のキーは通常、等号 (=) で終わりますが、より長いです。

2. 複数の鍵をアップロードするには、複数の --sshpubkey コマンドラインパラメーターを入力します。

```
--sshpubkey="RjlzYQo==" --sshpubkey="ZEt0TAo=="
```

注記

ホストには複数の公開鍵を指定できます。

3. ホスト鍵をアップロードしたら、OpenSSH サービスのキャッシュを提供するように SSSD を設定 で説明するように、Identity Management を ID ドメインの 1 つとして使用するよう SSSD を設定し、OpenSSH がホスト鍵管理に SSSD ツールを使用するよう設定します。

検証

- **ipa host-show** コマンドを実行して、SSH 公開鍵が指定されたホストに関連付けられていることを確認します。

```
$ ipa host-show client.ipa.test
...
SSH public key fingerprint:
SHA256:qGaqTZM60YPFTngFX0PtNPCKbIuudwf1D2LqmDeOcuA
                        client@IPA.TEST (ssh-rsa)
...
```

49.4.3. IdM Web UI を使用したホストの SSH 鍵の削除

ホスト鍵は、期限切れになるか、有効でなくなったら削除できます。IdM Web UI を使用して個々のホスト鍵を削除するには、以下の手順に従ってください。

前提条件

- IdM Web UI、またはホスト管理者ロールを管理する管理者権限

手順

1. IdM Web UI にログインします。
2. **Identity > Hosts** タブに移動します。
3. 編集するホスト名をクリックします。
4. **Host Settings** セクションで、削除する SSH 公開鍵の横にある **Delete** をクリックします。
5. ページ上部にある **Save** をクリックします。

検証

- **Host Settings** セクションで、鍵が **SSH public keys** の下にリストされていないことを確認します。

49.4.4. IdM CLI を使用したホストの SSH 鍵の削除

ホスト鍵は、期限切れになるか、有効でなくなったら削除できます。IdM CLI を使用して個々のホスト鍵を削除するには、以下の手順に従います。

前提条件

- IdM CLI、またはホスト管理者ロールを管理する管理者権限

手順

- ホストアカウントに割り当てられたすべての SSH 鍵を削除するには、鍵を指定せずに **--sshpubkey** オプションを **ipa host-mod** コマンドに追加します。

```
$ kinit admin
$ ipa host-mod --sshpubkey= --updatedns host1.example.com
```

  --updatedns オプションを使用してホストの DNS エントリーを更新することを推奨します。

アップロードされた鍵にタイプが含まれていない場合、IdM は鍵から自動的に鍵タイプを決定します。

検証

- **ipa host-show** コマンドを実行して、SSH 公開鍵が指定されたホストに関連付けられていないことを確認します。

```
ipa host-show client.ipa.test
  Host name: client.ipa.test
  Platform: x86_64
  Operating system: 4.18.0-240.el8.x86_64
  Principal name: host/client.ipa.test@IPA.TEST
  Principal alias: host/client.ipa.test@IPA.TEST
  Password: False
  Member of host-groups: ipaservers
  Roles: helpdesk
  Member of netgroups: test
  Member of Sudo rule: test2
  Member of HBAC rule: test
  Keytab: True
  Managed by: client.ipa.test, server.ipa.test
  Users allowed to retrieve keytab: user1, user2, user3
```

49.5. ユーザーの SSH 公開鍵の管理

Identity Management を使用すると、SSH 公開鍵をユーザーエントリーにアップロードできます。対応する SSH 鍵にアクセスできるユーザーは、SSH を使用して Kerberos 認証情報を使用せずに IdM マシン にログインすることができます。SSH 秘密鍵ファイルが利用できない場合でも、ユーザーは Kerberos 認証情報を提供して認証できることに注意してください。

49.5.1. IdM Web UI を使用したユーザーの SSH 鍵のアップロード

Identity Management を使用すると、SSH 公開鍵をユーザーエントリーにアップロードできます。対応する SSH 鍵にアクセスできるユーザーは、SSH を使用して Kerberos 認証情報を使用せずに IdM マシン にログインすることができます。

前提条件

- IdM Web UI、またはユーザー管理者ロールを管理する管理者権限

手順

1. IdM Web UI にログインします。
2. **Identity > Users** タブに移動します。
3. 編集するユーザー名をクリックします。
4. **Account Settings** セクションで、SSH 公開鍵の **Add** ボタンをクリックします。
5. Base 64 でエンコードされた公開鍵文字列を **SSH public key** フィールドに貼り付けます。
6. **Set** をクリックします。
7. IdM Web UI ウィンドウの上部にある **Save** をクリックします。

検証

- **Accounts Settings** セクションで、鍵が **SSH public keys** の下にリストされていることを確認します。

49.5.2. IdM CLI を使用したユーザーの SSH 鍵のアップロード

Identity Management を使用すると、SSH 公開鍵をユーザーエントリーにアップロードできます。対応する SSH 鍵にアクセスできるユーザーは、SSH を使用して Kerberos 認証情報を使用せずに IdM マシン にログインすることができます。

前提条件

- IdM CLI、またはユーザー管理者ロールを管理する管理者権限

手順

1. **--sshpubkey** オプションを指定して **ipa user-mod** コマンドを実行し、base64 にエンコードされた公開鍵をユーザーエントリーにアップロードします。

```
$ ipa user-mod user --sshpubkey="ssh-rsa AAAAB3Nza...SNc5dv== client.example.com"
```

この例では、鍵タイプ、鍵、およびホスト名識別子をユーザーエントリーにアップロードしていることに注意してください。

2. 複数の鍵をアップロードするには、**--sshpubkey** を複数回使用します。たとえば、SSH 鍵を 2 つアップロードするには、次のコマンドを実行します。

```
--sshpubkey="AAAAB3Nza...SNc5dv==" --sshpubkey="RjlzYQo...ZEt0TAo="
```

3. 鍵文字列を手動で貼り付ける代わりに、コマンドリダイレクトを使用して鍵を含むファイルを指定するには、次のコマンドを使用します。

```
ipa user-mod user --sshpubkey="$(cat ~/.ssh/id_rsa.pub)" --sshpubkey="$(cat
~/.ssh/id_rsa2.pub)"
```

-

検証

- **ipa user-show** コマンドを実行して、SSH 公開鍵が指定されたユーザーに関連付けられていることを確認します。

```
$ ipa user-show user
User login: user
  First name: user
  Last name: user
  Home directory: /home/user
  Login shell: /bin/sh
  Principal name: user@IPA.TEST
  Principal alias: user@IPA.TEST
  Email address: user@ipa.test
  UID: 1118800019
  GID: 1118800019
  SSH public key fingerprint:
SHA256:qGaqTZM60YPFTngFX0PtNPCKbIuudwf1D2LqmDeOcuA
                        user@IPA.TEST (ssh-rsa)
  Account disabled: False
  Password: False
  Member of groups: ipausers
  Subordinate ids: 3167b7cc-8497-4ff2-ab4b-6fcb3cb1b047
  Kerberos keys available: False
```

49.5.3. IdM Web UI を使用したユーザーの SSH 鍵の削除

IdM Web UI でユーザープロファイルから SSH 鍵を削除するには、次の手順に従います。

前提条件

- IdM Web UI、またはユーザー管理者ロールを管理する管理者権限

手順

1. IdM Web UI にログインします。
2. **Identity > Users** タブに移動します。
3. 編集するユーザー名をクリックします。
4. **Account Settings** セクションの **SSH public key** で、削除する鍵の横にある **Delete** をクリックします。
5. ページ上部にある **Save** をクリックします。

検証

- **Account Settings** セクションで、鍵が **SSH public keys** の下にリストされていないことを確認します。

49.5.4. IdM CLI を使用したユーザーの SSH 鍵の削除

IdM CLI を使用してユーザープロファイルから SSH 鍵を削除するには、次の手順に従います。

前提条件

- IdM CLI、またはユーザー管理者ロールを管理する管理者権限

手順

1. ユーザーアカウントに割り当てられたすべての SSH 鍵を削除するには、鍵を指定せずに **--sshpubkey** オプションを **ipa user-mod** コマンドに追加します。

```
$ ipa user-mod user --sshpubkey=
```

2. 特定の SSH 鍵のみを削除するには、**--sshpubkey** オプションを使用して、削除する鍵を省略し、保持する鍵を指定します。

検証

- **ipa user-show** コマンドを実行して、SSH 公開鍵が指定されたユーザーに関連付けられていないことを確認します。

```
$ ipa user-show user
User login: user
  First name: user
  Last name: user
  Home directory: /home/user
  Login shell: /bin/sh
  Principal name: user@IPA.TEST
  Principal alias: user@IPA.TEST
  Email address: user@ipa.test
  UID: 1118800019
  GID: 1118800019
  Account disabled: False
  Password: False
  Member of groups: ipausers
  Subordinate ids: 3167b7cc-8497-4ff2-ab4b-6fcb3cb1b047
  Kerberos keys available: False
```

第50章 ドメイン解決順序を設定して AD ユーザーの短縮名を解決する手順

デフォルトでは、**user_name@domain.com** または **domain.com\user_name** の形式で完全修飾名を指定して、Active Directory (AD) 環境からユーザーおよびグループを解決し、認証する必要があります。以下のセクションでは、AD ユーザーおよびグループの短縮名を解決するように IdM サーバーおよびクライアントを設定する方法を説明します。

- ドメイン解決順序の仕組み
- IdM サーバーでのグローバルドメイン解決順序設定
- IdM サーバーの ID ビューのドメイン解決順序設定
- Ansible を使用してドメイン解決順序を持つ ID ビューを作成する
- IdM クライアントの SSSD でのドメイン解決順序設定

50.1. ドメイン解決順序の仕組み

Red Hat では、Active Directory (AD) の信頼を使用する Identity Management (IdM) 環境では、完全修飾名を指定してユーザーおよびグループを解決し、認証することを推奨します。以下に例を示します。

- **idm.example.com** ドメインからの IdM ユーザーの場合は **<idm_username>@idm.example.com**
- **ad.example.com** ドメインからの AD ユーザーの場合は **<ad_username>@ad.example.com**

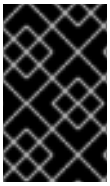
デフォルトでは、**ad_username** などの **短縮名** 形式を使用してユーザーまたはグループの検索を実行すると、IdM は IdM ドメインのみを検索する場合には、AD ユーザーまたはグループの検索に失敗します。短縮名を使用して AD ユーザーまたはグループを解決するには、**ドメイン解決順序** オプションを設定して、IdM が複数のドメインを検索する順番を変更します。

IdM データベースまたは個々のクライアントの SSSD 設定で、ドメイン解決の順序を一元的に設定できます。IdM は、以下の優先順位でドメイン解決の順序を評価します。

- ローカルの **/etc/sssd/sssd.conf** 設定。
- ID ビューの設定。
- グローバル IdM 設定。

備考

- ホストの SSSD 設定に **default_domain_suffix** オプションが含まれ、このオプションを指定せずにドメインへの要求を行う場合は、完全修飾ユーザー名を使用する必要があります。
- **ドメイン解決順序** オプションを使用して **compat** ツリーをクエリーすると、複数のユーザー ID (UID) が返される可能性があります。これで問題がある場合には、Pagure バグレポート Inconsistent compat user objects for AD users when domain resolution order is set を参照してください。



重要

IdM クライアントまたは IdM サーバーで **full_name_format** SSSD オプションは使用しないでください。このオプションにデフォルト値以外の値を使用すると、ユーザー名が表示される方法が変更され、IdM 環境での検索が中断される可能性があります。

関連情報

- Active Directory Trust for Legacy Linux Clients .

50.2. IDM サーバーでのグローバルドメイン解決順序設定

この手順では、IdM ドメイン内の全クライアントにドメイン解決の順序を設定します。この例では、ドメインの解決順序を設定して、以下の順番でユーザーとグループを検索します。

1. Active Directory (AD) root ドメイン **ad.example.com**
2. AD 子ドメイン **subdomain1.ad.example.com**
3. IdM ドメイン **idm.example.com**

前提条件

- AD 環境で信頼を設定している。

手順

- **ipa config-mod --domain-resolution-order** コマンドを使用して、希望の順序で検索するドメインをリスト表示します。コロン (**:**) でドメインを区切ります。

```
[user@server ~]$ ipa config-mod --domain-resolution-
order='ad.example.com:subdomain1.ad.example.com:idm.example.com'
Maximum username length: 32
Home directory base: /home
...
  Domain Resolution Order:
ad.example.com:subdomain1.ad.example.com:idm.example.com
...
```

検証手順

- 短縮名だけを使用して、**ad.example.com** ドメインからユーザー情報を取得できることを確認します。

```
[root@client ~]# id <ad_username>
uid=1916901102(ad_username) gid=1916900513(domain users)
groups=1916900513(domain users)
```

50.3. IDM サーバーの ID ビューのドメイン解決順序設定

この手順では、IdM サーバーおよびクライアントの特定のセットに適用できるように、ID ビューのドメイン解決の順序を設定します。この例では、IdM ホスト **client1.idm.example.com** に **ADsubdomain1_first** という名前の ID ビューを作成して、ドメインの解決順序を設定し、以下の順番

でユーザーとグループを検索します。

1. Active Directory (AD) 子ドメイン **subdomain1.ad.example.com**
2. AD root ドメイン **ad.example.com**
3. IdM ドメイン **idm.example.com**

注記

ID ビューで設定したドメイン解決の順序は、グローバルなドメイン解決順序より優先されますが、SSSD 設定でローカルに設定されたドメイン解決順序をオーバーライドすることはありません。

前提条件

- AD 環境で信頼を設定している。

手順

1. **--domain-resolution-order** オプションを指定して ID ビューを作成します。

```
[user@server ~]$ ipa idview-add ADsubdomain1_first --desc "ID view for resolving AD subdomain1 first on client1.idm.example.com" --domain-resolution-order subdomain1.ad.example.com:ad.example.com:idm.example.com
---------------------------------
Added ID View "ADsubdomain1_first"
---------------------------------
ID View Name: ADsubdomain1_first
Description: ID view for resolving AD subdomain1 first on client1.idm.example.com
Domain Resolution Order: subdomain1.ad.example.com:ad.example.com:idm.example.com
```

2. ID ビューを IdM ホストに適用します。

```
[user@server ~]$ ipa idview-apply ADsubdomain1_first --hosts client1.idm.example.com
---------------------------------
Applied ID View "ADsubdomain1_first"
---------------------------------
  hosts: client1.idm.example.com
-------------------------------------------
Number of hosts the ID View was applied to: 1
-------------------------------------------
```

検証手順

- ID ビューの詳細を表示します。

```
[user@server ~]$ ipa idview-show ADsubdomain1_first --show-hosts
  ID View Name: ADsubdomain1_first
  Description: ID view for resolving AD subdomain1 first on client1.idm.example.com
```

```
  Hosts the view applies to: client1.idm.example.com
  Domain resolution order:
subdomain1.ad.example.com:ad.example.com:idm.example.com
```

- 短縮名だけを使用して、**subdomain1.ad.example.com** ドメインからユーザー情報を取得できることを確認します。

```
[root@client1 ~]# id <user_from_subdomain1>
uid=1916901106(user_from_subdomain1) gid=1916900513(domain users)
groups=1916900513(domain users)
```

50.4. ANSIBLE を使用してドメイン解決順序を持つ ID ビューを作成する

ansible-freeipa idview モジュールを使用すると、Identity Management (IdM) デプロイメントの ID ビューを追加、変更、削除できます。たとえば、短縮名表記を有効にするために、ドメイン解決順序を持つ ID ビューを作成できます。

短縮名表記では、**aduser05@ad.example.com** などの Active Directory (AD) の完全なユーザー名が、短縮ログイン (この場合は **aduser05**) に置き換えられます。そのため、**SSH** を使用して IdM クライアントにログインする場合、**aduser05** は **ssh** aduser05@ad.example.com@client.idm.example.com ではなく **ssh** aduser05@client.idm.example.com と入力できます。**id** などの他のコマンドでも同じように入力できます。

この手順では、Ansible を使用して以下を実行します。

- 短縮名の修飾に使用する、コロン区切りのドメインの文字列を定義します。この例では、文字列は **ad.example.com:idm.example.com** です。
- 文字列で識別される最初のドメインでユーザー名をまず検索するように SSSD に指示する ID ビューを作成します。この例では、**ad.example.com** です。
- ID ビューを特定のホストに適用します。この例では、これは **testhost.idm.example.com** です。

注記

IdM クライアントに適用できるのは、1 つの ID ビューだけです。新しい ID ビューを適用すると、該当する場合、以前の ID ビューが自動的に削除されます。

前提条件

- コントロールノードでは、
  - Ansible バージョン 2.14 以降を使用している。
  - **ansible-freeipa** パッケージがインストールされている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している。
  - RHEL 9.4 以降を使用している。
  - **secret.yml** Ansible vault に **ipaadmin_password** が保存されている。
- **testhost.idm.example.com** が IdM クライアントである。

- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。

手順

1. **~/MyPlaybooks/** ディレクトリーに移動し、次の内容を含む Ansible Playbook ファイル **add-id-view-with-domain-resolution-order.yml** を作成します。

```
---
- name: Playbook to add idview and apply it to an IdM client
  hosts: ipaserver
  vars_files:
  - /home/<user_name>/MyPlaybooks/secret.yml
  become: false
  gather_facts: false

  tasks:
  - name: Add idview and apply it to testhost.idm.example.com
    ipaidview:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: test_idview
      host: testhost.idm.example.com
      domain_resolution_order: "ad.example.com:ipa.example.com"
```

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

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory add-id-view-with-domain-resolution-order.yml
```

検証

1. **testhost.idm.example.com** に SSH で接続します。
2. 短縮名だけを使用して、**ad.example.com** ドメインからユーザー情報を取得できることを確認します。

```
[root@testhost ~]# id aduser05
uid=1916901102(aduser05) gid=1916900513(domain users) groups=1916900513(domain users)
```

関連情報

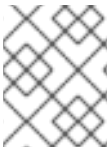
- **ansible-freeipa** アップストリームドキュメントの idview モジュール

50.5. IDM クライアントの SSSD でのドメイン解決順序設定

この手順では、IdM クライアントの SSSD 設定でドメイン解決の順序を設定します。この例では、IdM ホスト **client2.idm.example.com** が、以下の順番でユーザーとグループを検索するように設定します。

1. Active Directory (AD) 子ドメイン **subdomain1.ad.example.com**
2. AD root ドメイン **ad.example.com**

3. IdM ドメイン **idm.example.com**



注記

ローカルの SSSD 設定のドメイン解決順序は、グローバルおよび ID ビュードメインの解決順序よりも優先されます。

前提条件

- AD 環境で信頼を設定している。

手順

1. テキストエディターで **/etc/sssd/sssd.conf** ファイルを開きます。
2. このファイルの **[sssd]** セクションに **domain_resolution_order** オプションを設定します。

```
domain_resolution_order = subdomain1.ad.example.com, ad.example.com,
idm.example.com
```

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

```
[root@client2 ~]# systemctl restart sssd
```

検証手順

- 短縮名だけを使用して、**subdomain1.ad.example.com** ドメインからユーザー情報を取得できることを確認します。

```
[root@client2 ~]# id <user_from_subdomain1>
uid=1916901106(user_from_subdomain1) gid=1916900513(domain users)
groups=1916900513(domain users)
```

50.6. 関連情報

- ID ビューを使用した IdM クライアントのユーザー属性値のオーバーライド

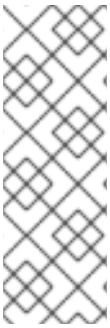
第51章 IDM での AD ユーザープリンシパル名を使用した認証の有効化

51.1. IDM で信頼される AD フォレストのユーザープリンシパル名

Identity Management (IdM) 管理者は、AD ユーザーが別の **ユーザープリンシパル名** (UPN) を使用して IdM ドメインのリソースにアクセスできるようにできます。UPN は、AD ユーザーが認証に使用する別のユーザーログインで、形式は **user_name@KERBEROS-REALM** です。AD フォレストでは、追加の Kerberos エイリアスと UPN 接尾辞の両方を設定することができるため、AD 管理者は **user_name** と **KERBEROS-REALM** の両方に別の値を設定できます。

たとえば、ある会社が **AD.EXAMPLE.COM** Kerberos レルムを使用する場合に、ユーザーのデフォルトの UPN は **user@ad.example.com** です。**user@example.com** などのメールアドレスを使用してユーザーがログインできるようにするには、AD で **EXAMPLE.COM** を別の UPN として設定できます。または、最近企業で合併が行われ、ログインの名前空間が統一して提供されるようにする場合に、UPN (**エンタープライズ UPN とも呼ばれる**) は特に便利です。

UPN 接尾辞は、AD フォレストルートで定義された場合に IdM にだけ表示されます。AD 管理者は、**Active Directory Domain and Trust** ユーティリティーまたは **PowerShell** コマンドラインツールで UPN を定義できます。



注記

Red Hat は、ユーザーへの UPN 接尾辞の設定には、**Active Directory Domain and Trust** ユーティリティーなどのエラー検証を実行するツールを使用することを推奨します。

Active Directory ではこのような操作は検証されないため、**ldapmodify** コマンドを使用してユーザーの **userPrincipalName** 属性を設定するなど、低レベルの変更で UPN を設定することを推奨します。

AD 側で新しい UPN を定義したら、IdM サーバーで **ipa trust-fetch-domains** コマンドを実行して、更新された UPN を取得します。AD UPN が IdM で最新であることを確認する手順 を参照してください。

IdM は、**cn=trusted_domain_name,cn=ad,cn=trusts,dc=idm,dc=example,dc=com** サブツリーの **ipaNTAdditionalSuffixes** 複数値の属性にドメインの UPN 接尾辞を保存します。

関連情報

- AD フォレストルートでの UPN 接尾辞設定の実装方法
- AD ユーザーエントリーを手動で修正して UPN 接尾辞の検証を省略する方法
- 信頼コントローラーおよび信頼エージェント

51.2. AD UPN が IDM で最新であることを確認する手順

信頼される Active Directory (AD) フォレストで、ユーザープリンシパル名 (UPN) 接尾辞を追加または削除してから、IdM サーバーで信頼されるフォレストの情報を更新します。

前提条件

- IdM 管理者の認証情報

手順

- **ipa trust-fetch-domains** コマンドを入力します。空白のような出力も想定範囲である点に注意してください。

```
[root@ipaserver ~]# ipa trust-fetch-domains
Realm-Name: ad.example.com
------------------------------
No new trust domains were found
------------------------------
---------------------------
Number of entries returned 0
---------------------------
```

検証手順

- **ipa trust-show** コマンドを入力し、サーバーが新しい UPN をフェッチしていることを確認します。プロンプトが表示されたら、AD レルムの名前を指定します。

```
[root@ipaserver ~]# ipa trust-show
Realm-Name: ad.example.com
  Realm-Name: ad.example.com
  Domain NetBIOS name: AD
  Domain Security Identifier: S-1-5-21-796215754-1239681026-23416912
  Trust direction: One-way trust
  Trust type: Active Directory domain
  UPN suffixes: example.com
```

この出力では、**example.com** UPN 接尾辞が **ad.example.com** レルムエントリーに含まれることが分かります。

51.3. AD UPN 認証問題のトラブルシューティングデータの収集

Active Directory (AD) 環境および IdM 環境からユーザープリンシパル名 (UPN) 設定に関するトラブルシューティングデータを収集するには、次の手順に従います。AD ユーザーが別の UPN を使用してログインできない場合は、こでの情報を使用してトラブルシューティング作業を絞り込むことができます。

前提条件

- AD ドメインコントローラーから情報を取得できるように IdM 信頼コントローラーまたは信頼エージェントにログインしておく。
- 以下の設定ファイルを変更し、IdM サービスを再起動できるように **root** 権限がある。

手順

1. テキストエディターで **/usr/share/ipa/smb.conf.empty** 設定ファイルを開きます。
2. 以下の内容をファイルに追加します。

```
[global]
log level = 10
```

3. **/usr/share/ipa/smb.conf.empty** ファイルを保存して閉じます。

4. テキストエディターで **/etc/ipa/server.conf** 設定ファイルを開きます。このファイルがない場合は作成します。
5. 以下の内容をファイルに追加します。

```
[global]
debug = True
```

6. **/etc/ipa/server.conf** ファイルを保存して終了します。
7. Apache Web サーバーのサービスを再起動して、設定の変更を適用します。

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

8. AD ドメインから信頼情報を取得します。

```
[root@server ~]# ipa trust-fetch-domains <ad.example.com>
```

9. 以下のログファイルでデバッグの出力とトラブルシューティング情報を確認します。
   - **/var/log/httpd/error_log**
   - **/var/log/samba/log.***

関連情報

- Using rpcclient to gather troubleshooting data for AD UPN authentication issues を参照してください。

第52章 IDM を管理する AD ユーザーの有効化

52.1. AD ユーザーの ID のオーバーライド

AD (Active Directory) ユーザーおよびグループの、POSIX 環境の Identity Management (IdM) リソースへのアクセスを一元管理するには、IdM グループのメンバーとして AD ユーザーの ID ユーザーオーバーライドを追加します。

ID オーバーライドは、特定の Active Directory ユーザーまたはグループのプロパティーが特定の ID ビュー (この場合は **Default Trust View**) 内でどのように見えるかを記述するレコードです。この機能により、IdM LDAP サーバーは、IdM グループのアクセス制御ルールを AD ユーザーに適用できます。

AD ユーザーは、IdM UI のセルフサービス機能 (SSH 鍵のアップロードや個人データの変更など) を使用できます。AD 管理者は、アカウントおよびパスワードを 2 つ使用しなくても、IdM を完全に管理できるようになります。

注記

IdM の一部の機能は、AD ユーザーには現在利用できません。たとえば、IdM の **admins** グループに所属する AD ユーザーが、IdM ユーザーのパスワードを設定することはできません。

重要

IdM の **sudo** ルールに AD ユーザーの ID オーバーライドを使用 **しない** でください。AD ユーザーの ID オーバーライドは、AD ユーザー自体ではなく、AD ユーザーの POSIX 属性のみを表します。

関連情報

- Active Directory ユーザーの ID ビューの使用

52.2. IDM を管理する AD ユーザーを有効にする ID オーバーライドの使用

AD ユーザーの ID オーバーライドを作成して使用し、そのユーザーに IdM ユーザーと同じ権限を付与するには、次の手順に従います。この手順は、信頼コントローラーまたは信頼エージェントとして設定した IdM サーバーで行います。

前提条件

- 作業用の IdM 環境が設定されている。詳細は、Identity Management のインストール を参照してください。
- IdM 環境と AD との間の信頼関係が設定されて動作している。

手順

1. IdM 管理者として、**Default Trust View** で AD ユーザーの ID オーバーライドを作成します。たとえば、ユーザー **ad_user@ad.example.com** の ID オーバーライドを作成するには、次のコマンドを実行します。

```
# kinit admin
# ipa idoverrideuser-add 'default trust view' ad_user@ad.example.com
```

2. **Default Trust View** から IdM グループのメンバーとして ID オーバーライドを追加します。これは、Active Directory と対話するため、POSIX 以外のグループである必要があります。
対象のグループが IdM ロールのメンバーである場合、ID オーバーライドによって表される AD ユーザーは、IdM API (コマンドラインインターフェイス、IdM の Web UI など) の使用時に、ロールにより付与されたすべての権限を取得します。

   たとえば、**ad_user@ad.example.com** ユーザーの ID オーバーライドを IdM **admins** グループに追加するには、次のコマンドを実行します。

   ```
   # ipa group-add-member admins --idoverrideusers=ad_user@ad.example.com
   ```

3. または、**User Administrator** ロールなどのロールに ID オーバーライドを追加することもできます。

   ```
   # ipa role-add-member 'User Administrator' --
   idoverrideusers=ad_user@ad.example.com
   ```

関連情報

- Active Directory ユーザーの ID ビューの使用

52.3. ANSIBLE を使用して AD ユーザーが IDM を管理できるようにする手順

Ansible Playbook を使用してユーザー ID オーバーライドが Identity Management (IdM) グループに存在することを確認するには、次の手順に従います。ユーザー ID オーバーライドは、Active Directory (AD) とのトラストを確立した後に、デフォルトの Trust View で作成した AD ユーザーのオーバーライドです。Playbook を実行すると、AD 管理者などの AD ユーザーは、2 つの異なるアカウントとパスワードを持たなくても IdM を完全に管理できるようになります。

前提条件

- IdM **admin** のパスワードを把握している。
- AD とのトラストをインストール している。
- IdM に AD ユーザーのユーザー ID オーバーライドがすでに存在する。存在しない場合は、**ipa idoverrideuser-add 'default trust view' ad_user@ad.example.com** コマンドで作成します。
- ユーザー ID オーバーライドを追加しようとしているグループが、IdM にすでに存在する 。
- IdM 4.8.7 バージョン以降の IdM を使用している。サーバーにインストールされている IdM のバージョンを表示するには、**ipa --version** を実行します。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。

- この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。

- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。

手順

1. **~/MyPlaybooks/** ディレクトリーに移動します。

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

2. 次の内容で **add-useridoverride-to-group.yml** playbook を作成します。

```
---
- name: Playbook to ensure presence of users in a group
  hosts: ipaserver


  - name: Ensure the ad_user@ad.example.com user ID override is a member of the admins
group:
    ipagroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: admins
      idoverrideuser:
      - ad_user@ad.example.com
```

上記の例では、以下のようになります。

- Secret123 は IdM **admin** パスワードです。

- **admins** は、**ad_user@ad.example.com** ID オーバーライドを追加する IdM POSIX グループの名前です。このグループのメンバーには、完全な管理者権限があります。

- **ad_user@ad.example.com** は、AD 管理者のユーザー ID オーバーライドです。ユーザーは、信頼が確立された AD ドメインに保存されます。

3. ファイルを保存します。

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

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory add-
useridoverride-to-group.yml
```

関連情報

- AD ユーザーの ID のオーバーライド

- /usr/share/doc/ansible-freeipa/README-group.md

- /usr/share/doc/ansible-freeipa/playbooks/user

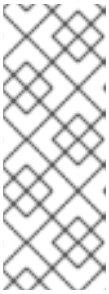
- Using ID views in Active Directory environments

52.4. AD ユーザーが IDM CLI で正しいコマンドを実行できることの確認

Active Directory (AD) ユーザーが Identity Management (IdM) コマンドラインインターフェイス (CLI) にログインし、自分のロールに適したコマンドを実行できることを確認します。

1. IdM 管理者の、現在の Kerberos チケットを破棄します。

```
# kdestroy -A
```



注記

MIT Kerberos の GSSAPI 実装が優先的にターゲットサービスの領域 (この場合は IdM レルム) から認証情報を選択するため、Kerberos チケットの破棄が必要です。これは、認証情報のキャッシュコレクションを意味します。つまり、タイプが **KCM:**、**KEYRING:**、または **DIR:** の認証情報キャッシュが使用されている場合、AD ユーザーの認証情報の代わりに、以前に取得した **admin** またはその他の IdM プリンシパルの認証情報が、IdM API にアクセスするために使用されます。

2. ID オーバーライドが作成された AD ユーザーの Kerberos 認証情報を入手します。

```
# kinit ad_user@AD.EXAMPLE.COM
Password for ad_user@AD.EXAMPLE.COM:
```

3. AD ユーザーの ID オーバーライド使用する、IdM グループのメンバーシップから生じるパーミッションが、そのグループ内の任意の IdM ユーザーと同じものであることをテストします。AD ユーザーの ID オーバーライドが **admins** グループに追加されている場合、AD ユーザーは、たとえば IdM にグループを作成できます。

```
# ipa group-add some-new-group
---------------------------
Added group "some-new-group"
---------------------------
  Group name: some-new-group
  GID: 1997000011
```

52.5. ANSIBLE を使用して AD ユーザーが IDM を管理できるようにする

ansible-freeipa の **idoverrideuser** および **group** モジュールを使用して、信頼済み AD ドメインの Active Directory (AD) ユーザーのユーザー ID オーバーライドを作成し、そのユーザーに IdM ユーザーと同じ権限を付与することができます。この手順で使用する例では、最初の Playbook タスクで **Default Trust View** ID ビューに **administrator@addomain.com** ID オーバーライドを追加します。次の Playbook タスクで、administrator@addomain.com ID オーバーライドを IdM **admins** グループにメンバーとして追加します。その結果、AD 管理者が 2 つの異なるアカウントとパスワードを使用しなくても IdM を管理できるようになります。

前提条件

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

- RHEL 9.4 以降を使用している。
- **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
- この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。

- AD フォレストが IdM と信頼関係にある。この例では、AD ドメインの名前は **addomain.com** であり、AD 管理者の完全修飾ドメイン名 (FQDN) は **administrator@addomain.com** です。
- インベントリーファイル内の **ipaserver** ホストが、信頼コントローラーまたは信頼エージェントとして設定されている。
- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。

手順

1. Ansible コントロールノードで、**administrator@addomain.com** ユーザーオーバーライドを Default Trust View に追加するタスクを含む **enable-ad-admin-to-administer-idm.yml** Playbook を作成します。

```
---
- name: Enable AD administrator to act as a FreeIPA admin
  hosts: ipaserver
  become: false
  gather_facts: false

  tasks:
  - name: Ensure idoverride for administrator@addomain.com in 'default trust view'
    ipaidoverrideuser:
      ipaadmin_password: "{{ ipaadmin_password }}"
      idview: "Default Trust View"
      anchor: administrator@addomain.com
```

2. 同じ Playbook 内の別の Playbook タスクを使用して、AD 管理者ユーザー ID オーバーライドを **admins** グループに追加します。

```
  - name: Add the AD administrator as a member of admins
    ipagroup:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: admins
      idoverrideuser:
      - administrator@addomain.com
```

3. ファイルを保存します。

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

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory enable-ad-admin-to-administer-idm.yml
```

検証

1. AD 管理者として IdM クライアントにログインします。

```
$ ssh administrator@addomain.com@client.idm.example.com
```

2. 有効な Ticket-Granting Ticket (TGT) を取得したことを確認します。

```
$ klist
Ticket cache: KCM:325600500:99540
Default principal: Administrator@ADDOMAIN.COM
Valid starting Expires Service principal
02/04/2024 11:54:16 02/04/2024 21:54:16 krbtgt/ADDOMAIN.COM@ADDOMAIN.COM
renew until 02/05/2024 11:54:16
```

3. IdM の **admin** 権限を確認します。

```
$ ipa user-add testuser --first=test --last=user
----------------------
Added user "tuser"
----------------------
  User login: tuser
  First name: test
  Last name: user
  Full name: test user
[...]
```

関連情報

- idoverrideuser および ipagroup に関する **ansible-freeipa** アップストリームドキュメント
- IdM を管理する AD ユーザーの有効化

第53章 外部 ID プロバイダーを使用した IDM に対する認証

OAuth 2 デバイス認可フローをサポートする外部アイデンティティープロバイダー (IdP) にユーザーを関連付けることができます。これらのユーザーが、RHEL 9.1 以降で利用可能な SSSD バージョンで認証すると、外部 IdP で認証と承認を実行した後に Kerberos チケットを使用した RHEL Identity Management (IdM) Single Sign-On 機能を受け取ります。

主な変更には以下のものがあります。

- **ipa idp-*** コマンドによる外部 IdP への参照の追加、変更、および削除
- **ipa user-mod --user-auth-type=idp** コマンドを使用したユーザーの IdP 認証の有効化

53.1. IDM を外部 IDP に接続する利点

管理者は、クラウドサービスプロバイダーなどの外部 ID ソースに保存されているユーザーが、Identity Management (IdM) 環境に追加された RHEL システムにアクセスできるようにすることができます。そのために、これらのユーザーの Kerberos チケットを発行する認証および認可プロセスをその外部エンティティーに委任できます。

この機能を使用して IdM の機能を拡張し、外部 ID プロバイダー (IdP) に保存されているユーザーが IdM によって管理される Linux システムにアクセスできるようにすることができます。

53.2. IDM が外部 IDP を介してログインを組み込む方法

SSSD 2.7.0 には、**idp** Kerberos 事前認証方法を実装する **sssd-idp** パッケージが含まれています。この認証方法は、OAuth 2.0 Device Authorization Grant フローに従って、認可の判断を外部 IdP に委任します。

1. IdM クライアントユーザーは、コマンドラインで **kinit** ユーティリティーを使用して Kerberos TGT の取得を試行するなどして、OAuth 2.0 デバイス認可付与フローを開始します。
2. 特別なコードと Web サイトのリンクが認可サーバーから IdM KDC バックエンドに送信されます。
3. IdM クライアントは、リンクとコードをユーザーに表示します。この例では、IdM クライアントはコマンドラインにリンクとコードを出力します。
4. ユーザーは、別のホストや携帯電話などのブラウザーで Web サイトのリンクを開きます。
    a. ユーザーは特別なコードを入力します。
    b. 必要に応じて、ユーザーは OAuth 2.0 ベースの IdP にログインします。
    c. ユーザーは、クライアントによる情報へのアクセスを許可するよう求められます。
5. ユーザーは、元のデバイスのプロンプトでアクセスを確認します。この例では、ユーザーはコマンドラインで **Enter** キーを押します。
6. IdM KDC バックエンドは、ユーザー情報にアクセスするために OAuth 2.0 認可サーバーをポーリングします。

サポート対象:

- Pluggable Authentication Module (PAM) ライブラリーの呼び出しを可能にする **keyboard-interactive** 認証方法を有効にして、SSH 経由でリモートからログインする場合。
- **logind** サービスを介してコンソールでローカルにログインする場合。
- **kinit** ユーティリティーを使用して Kerberos TGT (Ticket-granting ticket) を取得する場合。

現在のサポート対象外:

- IdM WebUI に直接ログインする場合。IdM WebUI にログインするには、最初に Kerberos チケットを取得する必要があります。
- Cockpit WebUI に直接ログインする場合。Cockpit WebUI にログインするには、最初に Kerberos チケットを取得する必要があります。

関連情報

- Authentication against external Identity Providers
- RFC 8628: OAuth 2.0 Device Authorization Grant

53.3. 外部アイデンティティープロバイダーへの参照の作成

外部アイデンティティープロバイダー (IdP) を Identity Management (IdM) 環境に接続するには、IdM で IdP 参照を作成します。この手順では、Keycloak テンプレートに基づいて IdP への **my-keycloak-idp** という参照を作成します。その他の参照テンプレートについては、IdM のさまざまな外部 IdP 参照の例 を参照してください。

前提条件

- IdM を OAuth アプリケーションとして外部 IdP に登録し、クライアント ID を取得している。
- IdM 管理者アカウントとして認証可能である。
- IdM サーバーで RHEL 9.1 以降を使用している。
- IdM サーバーで SSSD 2.7.0 以降を使用している。

手順

1. IdM サーバーで IdM 管理者として認証します。

```
[root@server ~]# kinit admin
```

2. Keycloak テンプレートに基づいて IdP への **my-keycloak-idp** という参照を作成します。**--base-url** オプションは、Keycloak サーバーへの URL を **server-name.$DOMAIN:$PORT/prefix** という形式で指定します。

```
[root@server ~]# ipa idp-add my-keycloak-idp \
                 --provider keycloak --organization main \
                 --base-url keycloak.idm.example.com:8443/auth \
                 --client-id id13778
----------------------------------------------
Added Identity Provider reference "my-keycloak-idp"
----------------------------------------------
```

```
  Identity Provider reference name: my-keycloak-idp
  Authorization URI:
https://keycloak.idm.example.com:8443/auth/realms/main/protocol/openid-connect/auth
  Device authorization URI:
https://keycloak.idm.example.com:8443/auth/realms/main/protocol/openid-
connect/auth/device
  Token URI: https://keycloak.idm.example.com:8443/auth/realms/main/protocol/openid-
connect/token
  User info URI: https://keycloak.idm.example.com:8443/auth/realms/main/protocol/openid-
connect/userinfo
  Client identifier: ipa_oidc_client
  Scope: openid email
  External IdP user identifier attribute: email
```

検証

- **ipa idp-show** コマンドの出力に、作成した IdP 参照が表示されていることを確認します。

```
[root@server ~]# ipa idp-show my-keycloak-idp
```

関連情報

- IdM のさまざまな外部 IdP 参照の例
- IdM で外部アイデンティティープロバイダーを管理するための ipa idp-* コマンドのオプション
- ipa idp-* コマンドの --provider オプション
- **ipa help idp-add**

53.4. IDM のさまざまな外部 IDP 参照の例

次の表に、IdM のさまざまな IdP 参照を作成するための **ipa idp-add** コマンドの例を示します。

アイデンティティープロバイダー	重要なオプション	コマンド例
Microsoft Identity Platform、Azure AD	**--provider microsoft --organization**	# ipa idp-add **my-azure-idp** \ **--provider microsoft** \ --organization **main** \ --client-id **<azure_client_id>**
Google	**--provider google**	# ipa idp-add **my-google-idp** \ **--provider google** \ --client-id **<google_client_id>**

アイデンティティープロバイダー	重要なオプション	コマンド例
GitHub	**--provider github**	# ipa idp-add my-github-idp \ **--provider github** \ --client-id **<github_client_id>**
Keycloak、Red Hat Single Sign-On	**--provider keycloak** **--organization** **--base-url**	# ipa idp-add my-keycloak-idp \ **--provider keycloak** \ --organization main \ --base-url **keycloak.idm.example.com:8443/auth** \ --client-id **<keycloak_client_id>**  **注記** Keycloak 17 以降の Quarkus バージョンでは、URI の **/auth/** 部分が削除されています。デプロイメントで Keycloak の非 Quarkus ディストリビューションを使用する場合は、**--base-url** オプションに **/auth/** を含めます。
Okta	**--provider okta**	# ipa idp-add my-okta-idp \ **--provider okta** --base-url **dev-12345.okta.com** \ --client-id **<okta_client_id>**

関連情報

- 外部アイデンティティープロバイダーへの参照の作成
- IdM で外部アイデンティティープロバイダーを管理するための ipa idp-* コマンドのオプション
- ipa idp-* コマンドの --provider オプション

53.5. IDM で外部アイデンティティープロバイダーを管理するための IPA IDP-* コマンドのオプション

次の例は、さまざまな IdP テンプレートに基づいて外部 IdP への参照を設定する方法を示しています。次のオプションを使用して設定を指定します。

--provider

既知の ID プロバイダーのいずれかの定義済みテンプレート

--client-id

アプリケーション登録時に IdP によって発行された OAuth 2.0 クライアント識別子。アプリケーションの登録手順は IdP ごとに異なるため、詳細については各 IdP のドキュメントを参照してください。外部 IdP が Red Hat Single Sign-On (SSO) の場合は、OpenID Connect クライアントの作成 を参照してください。

--base-url

Keycloak と Okta で必要な IdP テンプレートのベース URL

organization

Microsoft Azure で必要な IdP からのドメインまたは組織 ID

--secret

(オプション) 機密 OAuth 2.0 クライアントからのシークレットを要求するように、外部 IdP を設定した場合は、このオプションを使用します。IdP 参照を作成するときにこのオプションを使用すると、シークレットを対話的に求めるプロンプトが表示されます。クライアントシークレットをパスワードとして保護します。

注記

RHEL 9.1 の SSSD は、クライアントシークレットを使用しない非機密 OAuth 2.0 クライアントのみをサポートします。機密クライアントからのクライアントシークレットを必要とする外部 IdP を使用する場合は、RHEL 9.2 以降で SSSD を使用する必要があります。

関連情報

- 外部アイデンティティープロバイダーへの参照の作成
- IdM のさまざまな外部 IdP 参照の例
- ipa idp-* コマンドの --provider オプション

53.6. 外部 IDP への参照の管理

外部 ID プロバイダー (IdP) への参照を作成したら、その参照を検索、表示、変更、および削除できます。この例では、**keycloak-server1** という名前の外部 IdP への参照を管理する方法を示します。

前提条件

- IdM 管理者アカウントとして認証可能である。
- IdM サーバーで RHEL 9.1 以降を使用している。
- IdM サーバーで SSSD 2.7.0 以降を使用している。
- IdM で外部 IdP への参照を作成した。外部 ID プロバイダーへの参照の作成 を参照してください。

手順

1. IdM サーバーで IdM 管理者として認証します。
   ■

```
[root@server ~]# kinit admin
```

2. IdP 参照を管理します。

- エントリーに文字列 **keycloak** が含まれる IdP 参照を見つけるには、以下を実行します。

```
[root@server ~]# ipa idp-find keycloak
```

- **my-keycloak-idp** という名前の IdP 参照を表示するには、以下を実行します。

```
[root@server ~]# ipa idp-show my-keycloak-idp
```

- IdP 参照を変更するには、**ipa idp-mod** コマンドを使用します。たとえば、**my-keycloak-idp** という名前の IdP 参照のシークレットを変更するには、**--secret** オプションを指定してシークレットの入力を求めます。

```
[root@server ~]# ipa idp-mod my-keycloak-idp --secret
```

- **my-keycloak-idp** という名前の IdP 参照を削除するには、以下を実行します。

```
[root@server ~]# ipa idp-del my-keycloak-idp
```

53.7. 外部 IDP 経由での IDM ユーザーの認証を有効にする方法

外部 ID プロバイダー (IdP) 経由で IdM ユーザーを認証できるようにするには、以前に作成した外部 IdP 参照をユーザーアカウントに関連付けます。この例では、外部 IdP 参照 **keycloak-server1** をユーザー **idm-user-with-external-idp** に関連付けます。

前提条件

- IdM クライアントと IdM サーバーで RHEL 9.1 以降を使用している。
- IdM クライアントと IdM サーバーで SSSD 2.7.0 以降を使用している。
- IdM で外部 IdP への参照を作成した。外部 ID プロバイダーへの参照の作成 を参照してください。

手順

- IdM ユーザーエントリーを変更して、IdP 参照をユーザーアカウントに関連付けます。

```
[root@server ~]# ipa user-mod idm-user-with-external-idp \
                 --idp my-keycloak-idp \
                 --idp-user-id idm-user-with-external-idp@idm.example.com \
                 --user-auth-type=idp
--------------------------------
Modified user "idm-user-with-external-idp"
--------------------------------
  User login: idm-user-with-external-idp
  First name: Test
  Last name: User1
  Home directory: /home/idm-user-with-external-idp
  Login shell: /bin/sh
```

```
Principal name: idm-user-with-external-idp@idm.example.com
Principal alias: idm-user-with-external-idp@idm.example.com
Email address: idm-user-with-external-idp@idm.example.com
UID: 35000003
GID: 35000003
User authentication types: idp
External IdP configuration: keycloak
External IdP user identifier: idm-user-with-external-idp@idm.example.com
Account disabled: False
Password: False
Member of groups: ipausers
Kerberos keys available: False
```

検証

- そのユーザーの **ipa user-show** コマンド出力に IdP への参照が表示されることを確認します。

```
[root@server ~]# ipa user-show idm-user-with-external-idp
  User login: idm-user-with-external-idp
  First name: Test
  Last name: User1
  Home directory: /home/idm-user-with-external-idp
  Login shell: /bin/sh
  Principal name: idm-user-with-external-idp@idm.example.com
  Principal alias: idm-user-with-external-idp@idm.example.com
  Email address: idm-user-with-external-idp@idm.example.com
  ID: 35000003
  GID: 35000003
  User authentication types: idp
  External IdP configuration: keycloak
  External IdP user identifier: idm-user-with-external-idp@idm.example.com
  Account disabled: False
  Password: False
  Member of groups: ipausers
  Kerberos keys available: False
```

53.8. 外部 IDP ユーザーとして IDM TICKET-GRANTING TICKET を取得する

Identity Management (IdM) ユーザーの認証を外部アイデンティティープロバイダー (IdP) に委譲している場合、IdM ユーザーは外部 IdP に対して認証することで Kerberos Ticket-Granting Ticket (TGT) を要求できます。

この手順では、以下を実行します。

1. 匿名の Kerberos チケットを取得してローカルに保存します。
2. **-T** オプションを指定した **kinit** を使用して **idm-user-with-external-idp** ユーザーの TGT を要求し、Flexible Authentication via Secure Tunneling (FAST) チャネルを有効にして、Kerberos クライアントと Kerberos Distribution Center (KDC) 間のセキュアな接続を提供します。

前提条件

- IdM クライアントと IdM サーバーが RHEL 9.1 以降を使用している。

- IdM クライアントと IdM サーバーが SSSD 2.7.0 以降を使用している。
- IdM で外部 IdP への参照を作成した。外部 ID プロバイダーへの参照の作成 を参照してください。
- 外部 IdP 参照をユーザーアカウントに関連付けている。外部 IdP 経由での IdM ユーザーの認証を有効にする方法 を参照してください。
- 最初にログインするユーザーに、ローカルファイルシステム内のディレクトリーに対する書き込み権限がある。

手順

1. 匿名 PKINIT を使用して Kerberos チケットを取得し、それを **./fast.ccache** という名前のファイルに保存します。

```
$ kinit -n -c ./fast.ccache
```

2. [オプション] 取得したチケットを表示します。

```
$ *klist -c fast.ccache *
Ticket cache: FILE:fast.ccache
Default principal: WELLKNOWN/ANONYMOUS@WELLKNOWN:ANONYMOUS

Valid starting       Expires              Service principal
03/03/2024 13:36:37  03/04/2024 13:14:28
krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
```

3. **-T** オプションを使用して FAST 通信チャネルを有効にし、IdM ユーザーとして認証を開始します。

```
[root@client ~]# kinit -T ./fast.ccache idm-user-with-external-idp
Authenticate at https://oauth2.idp.com:8443/auth/realms/master/device?user_code=YHMQ-
XKTL and press ENTER.:
```

4. ブラウザーで、コマンド出力に提供される Web サイトでユーザーとして認証します。
5. コマンドラインで **Enter** キーを押して、認証プロセスを終了します。

検証

- Kerberos チケット情報を表示し、**config: pa_type** の行が外部 IdP による事前認証の **152** を示していることを確認します。

```
[root@client ~]# klist -C
Ticket cache: KCM:0:58420
Default principal: idm-user-with-external-idp@IDM.EXAMPLE.COM

Valid starting     Expires            Service principal
05/09/22 07:48:23  05/10/22 07:03:07  krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
config: fast_avail(krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM) = yes
08/17/2022 20:22:45  08/18/2022 20:22:43
krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
config: pa_type(krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM) = 152
```

pa_type = 152 は、外部 IdP 認証を示します。

53.9. 外部 IDP ユーザーとして SSH 経由で IDM クライアントにログインする

外部 ID プロバイダー (IdP) ユーザーとして SSH 経由で IdM クライアントにログインするには、コマンドラインでログインプロセスを開始します。プロンプトが表示されたら、IdP に関連付けられた Web サイトで認証プロセスを実行し、Identity Management (IdM) クライアントでプロセスを終了します。

前提条件

- IdM クライアントと IdM サーバーで RHEL 9.1 以降を使用している。
- IdM クライアントと IdM サーバーで SSSD 2.7.0 以降を使用している。
- IdM で外部 IdP への参照を作成した。外部 ID プロバイダーへの参照の作成 を参照してください。
- 外部 IdP 参照をユーザーアカウントに関連付けている。外部 IdP 経由での IdM ユーザーの認証を有効にする方法 を参照してください。

手順

1. SSH 経由で IdM クライアントへのログインを試みます。

```
[user@client ~]$ ssh idm-user-with-external-idp@client.idm.example.com
(idm-user-with-external-idp@client.idm.example.com) Authenticate at
https://oauth2.idp.com:8443/auth/realms/main/device?user_code=XYFL-ROYR and press
ENTER.
```

2. ブラウザーで、コマンド出力に提供される Web サイトでユーザーとして認証します。
3. コマンドラインで **Enter** キーを押して、認証プロセスを終了します。

検証

- Kerberos チケット情報を表示し、**config: pa_type** の行が外部 IdP による事前認証の **152** を示していることを確認します。

```
[idm-user-with-external-idp@client ~]$ klist -C
Ticket cache: KCM:0:58420
Default principal: idm-user-with-external-idp@IDM.EXAMPLE.COM

Valid starting     Expires            Service principal
05/09/22 07:48:23  05/10/22 07:03:07  krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
config: fast_avail(krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM) = yes
08/17/2022 20:22:45  08/18/2022 20:22:43
krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
config: pa_type(krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM) = 152
```

53.10. IPA IDP-* コマンドの --PROVIDER オプション

次の ID プロバイダー (IdP) は、OAuth 2.0 デバイス認可グラントフローをサポートしています。

- Azure AD を含む Microsoft Identity Platform
- Google
- GitHub
- Red Hat Single Sign-On (SSO) を含む Keycloak
- Okta

ipa idp-add コマンドを使用してこれらの外部 IdP のいずれか 1 つへの参照を作成する場合、**--provider** オプションで IdP タイプを指定できます。これは、以下で説明する追加オプションに拡張されます。

--provider=microsoft

Microsoft Azure IdP では、**--organization** オプションで **ipa idp-add** に指定できる Azure テナント ID に基づくパラメーター化が可能です。live.com IdP のサポートが必要な場合は、**--organization common** オプションを指定します。
--provider=microsoft を選択すると、次のオプションを使用するように拡張されます。**--organization** オプションの値は、表内の文字列 **${ipaidporg}** を置き換えます。

オプション	値
--auth-uri=URI	**https://login.microsoftonline.com/${ipaidporg}/oauth2/v2.0/authorize**
--dev-auth-uri=URI	**https://login.microsoftonline.com/${ipaidporg}/oauth2/v2.0/devicecode**
--token-uri=URI	**https://login.microsoftonline.com/${ipaidporg}/oauth2/v2.0/token**
--userinfo-uri=URI	**https://graph.microsoft.com/oidc/userinfo**
--keys-uri=URI	**https://login.microsoftonline.com/common/discovery/v2.0/keys**
--scope=STR	**openid email**
--idp-user-id=STR	**email**

--provider=google

--provider=google を選択すると、次のオプションを使用するように拡張されます。

オプション	値
--auth-uri=URI	**https://accounts.google.com/o/oauth2/auth**
--dev-auth-uri=URI	**https://oauth2.googleapis.com/device/code**
--token-uri=URI	**https://oauth2.googleapis.com/token**

オプション	値
--userinfo-uri=URI	**https://openidconnect.googleapis.com/v1/userinfo**
--keys-uri=URI	**https://www.googleapis.com/oauth2/v3/certs**
--scope=STR	**openid email**
--idp-user-id=STR	**email**

--provider=github

--provider=github を選択すると、次のオプションを使用するように拡張されます。

オプション	値
--auth-uri=URI	**https://github.com/login/oauth/authorize**
--dev-auth-uri=URI	**https://github.com/login/device/code**
--token-uri=URI	**https://github.com/login/oauth/access_token**
--userinfo-uri=URI	**https://openidconnect.googleapis.com/v1/userinfo**
--keys-uri=URI	**https://api.github.com/user**
--scope=STR	**user**
--idp-user-id=STR	**login**

--provider=keycloak

Keycloak を使用すると、複数のレルムまたは組織を定義できます。多くの場合、これはカスタムデプロイメントの一部であるため、ベース URL とレルム ID の両方が必要です。これらは、**ipa idp-add** コマンドの **--base-url** および **--organization** オプションで指定できます。

```
[root@client ~]# ipa idp-add MySSO --provider keycloak \
    --org main --base-url keycloak.domain.com:8443/auth \
    --client-id <your-client-id>
```

--provider=keycloak を選択すると、次のオプションを使用するように拡張されます。**--base-url** オプションで指定した値は、表内の文字列 **${ipaidpbaseurl}** を置き換え、指定した値は **--organization `option replaces the string `${ipaidporg}** となります。

オプション	値
--auth-uri=URI	**https://${ipaidpbaseurl}/realms/${ipaidporg}/protocol/openid-connect/auth**

オプション	値
--dev-auth-uri=URI	**https://${ipaidpbaseurl}/realms/${ipaidporg}/protocol/openid-connect/auth/device**
--token-uri=URI	**https://${ipaidpbaseurl}/realms/${ipaidporg}/protocol/openid-connect/token**
--userinfo-uri=URI	**https://${ipaidpbaseurl}/realms/${ipaidporg}/protocol/openid-connect/userinfo**
--scope=STR	**openid email**
--idp-user-id=STR	**email**

--provider=okta

Okta に新しい組織を登録すると、新しいベース URL が関連付けられます。このベース URL は、**ipa idp-add** コマンドの **--base-url** オプションで指定できます。

```
[root@client ~]# ipa idp-add MyOkta --provider okta --base-url dev-12345.okta.com --client-id
<your-client-id>
```

--provider=okta を選択すると、次のオプションを使用するように拡張されます。**--base-url** オプションに指定した値は、表内の文字列 **${ipaidpbaseurl}** を置き換えます。

オプション	値
--auth-uri=URI	**https://${ipaidpbaseurl}/oauth2/v1/authorize**
--dev-auth-uri=URI	**https://${ipaidpbaseurl}/oauth2/v1/device/authorize**
--token-uri=URI	**https://${ipaidpbaseurl}/oauth2/v1/token**
--userinfo-uri=URI	**https://${ipaidpbaseurl}/oauth2/v1/userinfo**
--scope=STR	**openid email**
--idp-user-id=STR	**email**

関連情報

- 事前入力された IdP テンプレート

第54章 ANSIBLE を使用して IDM ユーザーの認証を外部アイデンティティープロバイダーに委譲する

idp ansible-freeipa モジュールを使用して、OAuth 2 デバイス認証フローをサポートする外部アイデンティティープロバイダー (IdP) にユーザーを関連付けることができます。IdP 参照と関連付けられた IdP ユーザーの ID が存在する場合は、それらを使用して、**useransible-freeipa** モジュールにより IdM ユーザーの IdP 認証を有効にすることができます。

その後、これらのユーザーが RHEL 9.1 以降で利用可能な SSSD バージョンで認証すると、外部 IdP で認証と認可が実行された後、Kerberos チケットを使用した RHEL Identity Management (IdM) シングルサインオン機能がユーザーに提供されます。

54.1. IDM を外部 IDP に接続する利点

管理者は、クラウドサービスプロバイダーなどの外部 ID ソースに保存されているユーザーが、Identity Management (IdM) 環境に追加された RHEL システムにアクセスできるようにすることができます。そのために、これらのユーザーの Kerberos チケットを発行する認証および認可プロセスをその外部エンティティーに委任できます。

この機能を使用して IdM の機能を拡張し、外部 ID プロバイダー (IdP) に保存されているユーザーが IdM によって管理される Linux システムにアクセスできるようにすることができます。

54.2. IDM が外部 IDP を介してログインを組み込む方法

SSSD 2.7.0 には、**idp** Kerberos 事前認証方法を実装する **sssd-idp** パッケージが含まれています。この認証方法は、OAuth 2.0 Device Authorization Grant フローに従って、認可の判断を外部 IdP に委任します。

1. IdM クライアントユーザーは、コマンドラインで **kinit** ユーティリティーを使用して Kerberos TGT の取得を試行するなどして、OAuth 2.0 デバイス認可付与フローを開始します。
2. 特別なコードと Web サイトのリンクが認可サーバーから IdM KDC バックエンドに送信されます。
3. IdM クライアントは、リンクとコードをユーザーに表示します。この例では、IdM クライアントはコマンドラインにリンクとコードを出力します。
4. ユーザーは、別のホストや携帯電話などのブラウザーで Web サイトのリンクを開きます。
   a. ユーザーは特別なコードを入力します。
   b. 必要に応じて、ユーザーは OAuth 2.0 ベースの IdP にログインします。
   c. ユーザーは、クライアントによる情報へのアクセスを許可するよう求められます。
5. ユーザーは、元のデバイスのプロンプトでアクセスを確認します。この例では、ユーザーはコマンドラインで **Enter** キーを押します。
6. IdM KDC バックエンドは、ユーザー情報にアクセスするために OAuth 2.0 認可サーバーをポーリングします。

サポート対象:

- Pluggable Authentication Module (PAM) ライブラリーの呼び出しを可能にする **keyboard-interactive** 認証方法を有効にして、SSH 経由でリモートからログインする場合。

- **logind** サービスを介してコンソールでローカルにログインする場合。
- **kinit** ユーティリティーを使用して Kerberos TGT (Ticket-granting ticket) を取得する場合。

現在のサポート対象外:

- IdM WebUI に直接ログインする場合。IdM WebUI にログインするには、最初に Kerberos チケットを取得する必要があります。
- Cockpit WebUI に直接ログインする場合。Cockpit WebUI にログインするには、最初に Kerberos チケットを取得する必要があります。

関連情報

- Authentication against external Identity Providers
- RFC 8628: OAuth 2.0 Device Authorization Grant

54.3. ANSIBLE を使用して外部アイデンティティープロバイダーへの参照を作成する

外部アイデンティティープロバイダー (IdP) を Identity Management (IdM) 環境に接続するには、IdM で IdP 参照を作成します。この手順では、**idp ansible-freeipa** モジュールを使用して **github** 外部 IdP への参照を設定します。

前提条件

- IdM を OAuth アプリケーションとして外部 IdP に登録し、IdM ユーザーが IdM への認証に使用するデバイス上でクライアント ID とクライアントシークレットを生成した。この例では、以下を前提としています。
  - **my_github_account_name** が github ユーザーであり、そのアカウントを IdM ユーザーが IdM への認証に使用する。
  - **client ID** が **2efe1acffe9e8ab869f4** である。
  - **client secret** が **656a5228abc5f9545c85fa626aecbf69312d398c** である。
- IdM サーバーで RHEL 9.1 以降を使用している。
- IdM サーバーで SSSD 2.7.0 以降を使用している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - RHEL 9.4 以降を使用している。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
  - この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。

手順

1. Ansible コントロールノードで、**configure-external-idp-reference.yml** Playbook を作成します。

```
---
- name: Configure external IdP
  hosts: ipaserver
  become: false
  gather_facts: false

  tasks:
  - name: Ensure a reference to github external provider is available
    ipaidp:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: github_idp
      provider: github
      client_ID: 2efe1acffe9e8ab869f4
      secret: 656a5228abc5f9545c85fa626aecbf69312d398c
      idp_user_id: my_github_account_name
```

2. ファイルを保存します。

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

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory configure-external-idp-reference.yml
```

検証

- IdM クライアントで、**ipa idp-show** コマンドの出力に、作成した IdP 参照が表示されることを確認します。

```
[idmuser@idmclient ~]$ ipa idp-show github_idp
```

次のステップ

- Ansible を使用して IdM ユーザーが外部 IdP 経由で認証できるようにする

関連情報

- idp **ansible-freeipa** アップストリームドキュメント

54.4. ANSIBLE を使用して IDM ユーザーが外部 IDP 経由で認証できるようにする

user ansible-freeipa モジュールを使用すると、Identity Management (IdM) ユーザーが外部アイデンティティープロバイダー (IdP) 経由で認証できるようになります。これを行うには、以前に作成した外部 IdP 参照を IdM ユーザーアカウントに関連付けます。この手順では、Ansible を使用して、**github_idp** という名前の外部 IdP 参照を **idm-user-with-external-idp** という名前の IdM ユーザーに関連付けます。この手順の結果、ユーザーが **my_github_account_name** github アイデンティティーを使用して、**idm-user-with-external-idp** として IdM に認証できるようになります。

前提条件

- IdM クライアントと IdM サーバーで RHEL 9.1 以降を使用している。
- IdM クライアントと IdM サーバーで SSSD 2.7.0 以降を使用している。
- IdM で外部 IdP への参照を作成した。Ansible を使用して外部アイデンティティープロバイダーへの参照を作成する を参照してください。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - RHEL 9.4 以降を使用している。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成している (この例の場合)。
  - この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。

手順

1. Ansible コントロールノードで、**enable-user-to-authenticate-via-external-idp.yml** Playbook を作成します。

```
---
- name: Ensure an IdM user uses an external IdP to authenticate to IdM
  hosts: ipaserver
  become: false
  gather_facts: false

  tasks:
  - name: Retrieve Github user ID
    ansible.builtin.uri:
      url: “https://api.github.com/users/my_github_account_name”
      method: GET
      headers:
        Accept: “application/vnd.github.v3+json”
    register: user_data

  - name: Ensure IdM user exists with an external IdP authentication
    ipauser:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: idm-user-with-external-idp
      first: Example
      last: User
      userauthtype: idp
      idp: github_idp
      idp_user_id: my_github_account_name
```

2. ファイルを保存します。

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

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory enable-user-to-authenticate-via-external-idp.yml
```

検証

- IdM クライアントにログインし、**idm-user-with-external-idp** ユーザーの **ipa user-show** コマンドの出力に IdP への参照が表示されることを確認します。

```
$ ipa user-show idm-user-with-external-idp
User login: idm-user-with-external-idp
First name: Example
Last name: User
Home directory: /home/idm-user-with-external-idp
Login shell: /bin/sh
Principal name: idm-user-with-external-idp@idm.example.com
Principal alias: idm-user-with-external-idp@idm.example.com
Email address: idm-user-with-external-idp@idm.example.com
ID: 35000003
GID: 35000003
User authentication types: idp
External IdP configuration: github
External IdP user identifier: idm-user-with-external-idp@idm.example.com
Account disabled: False
Password: False
Member of groups: ipausers
Kerberos keys available: False
```

関連情報

- idp **ansible-freeipa** アップストリームドキュメント

54.5. 外部 IDP ユーザーとして IDM TICKET-GRANTING TICKET を取得する

Identity Management (IdM) ユーザーの認証を外部アイデンティティープロバイダー (IdP) に委譲している場合、IdM ユーザーは外部 IdP に対して認証することで Kerberos Ticket-Granting Ticket (TGT) を要求できます。

この手順では、以下を実行します。

1. 匿名の Kerberos チケットを取得してローカルに保存します。
2. **-T** オプションを指定した **kinit** を使用して **idm-user-with-external-idp** ユーザーの TGT を要求し、Flexible Authentication via Secure Tunneling (FAST) チャネルを有効にして、Kerberos クライアントと Kerberos Distribution Center (KDC) 間のセキュアな接続を提供します。

前提条件

- IdM クライアントと IdM サーバーが RHEL 9.1 以降を使用している。
- IdM クライアントと IdM サーバーが SSSD 2.7.0 以降を使用している。

- IdM で外部 IdP への参照を作成した。Ansible を使用して外部アイデンティティープロバイダーへの参照を作成する を参照してください。
- 外部 IdP 参照をユーザーアカウントに関連付けている。Ansible を使用して IdM ユーザーが外部 IdP 経由で認証できるようにする を参照してください。
- 最初にログインするユーザーに、ローカルファイルシステム内のディレクトリーに対する書き込み権限がある。

手順

1. 匿名 PKINIT を使用して Kerberos チケットを取得し、それを **./fast.ccache** という名前のファイルに保存します。

```
$ kinit -n -c ./fast.ccache
```

2. [オプション] 取得したチケットを表示します。

```
$ *klist -c fast.ccache *
Ticket cache: FILE:fast.ccache
Default principal: WELLKNOWN/ANONYMOUS@WELLKNOWN:ANONYMOUS

Valid starting       Expires              Service principal
03/03/2024 13:36:37  03/04/2024 13:14:28
krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
```

3. **-T** オプションを使用して FAST 通信チャネルを有効にし、IdM ユーザーとして認証を開始します。

```
[root@client ~]# kinit -T ./fast.ccache idm-user-with-external-idp
Authenticate at https://oauth2.idp.com:8443/auth/realms/master/device?user_code=YHMQ-XKTL and press ENTER.:
```

4. ブラウザーで、コマンド出力に提供される Web サイトでユーザーとして認証します。
5. コマンドラインで **Enter** キーを押して、認証プロセスを終了します。

検証

- Kerberos チケット情報を表示し、**config: pa_type** の行が外部 IdP による事前認証の **152** を示していることを確認します。

```
[root@client ~]# klist -C
Ticket cache: KCM:0:58420
Default principal: idm-user-with-external-idp@IDM.EXAMPLE.COM

Valid starting     Expires            Service principal
05/09/22 07:48:23  05/10/22 07:03:07  krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
config: fast_avail(krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM) = yes
08/17/2022 20:22:45  08/18/2022 20:22:43
krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
config: pa_type(krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM) = 152
```

pa_type = 152 は、外部 IdP 認証を示します。

54.6. 外部 IDP ユーザーとして SSH 経由で IDM クライアントにログインする

外部 ID プロバイダー (IdP) ユーザーとして SSH 経由で IdM クライアントにログインするには、コマンドラインでログインプロセスを開始します。プロンプトが表示されたら、IdP に関連付けられた Web サイトで認証プロセスを実行し、Identity Management (IdM) クライアントでプロセスを終了します。

前提条件

- IdM クライアントと IdM サーバーで RHEL 9.1 以降を使用している。
- IdM クライアントと IdM サーバーで SSSD 2.7.0 以降を使用している。
- IdM で外部 IdP への参照を作成した。Ansible を使用して外部アイデンティティープロバイダーへの参照を作成する を参照してください。
- 外部 IdP 参照をユーザーアカウントに関連付けている。Ansible を使用して IdM ユーザーが外部 IdP 経由で認証できるようにする を参照してください。

手順

1. SSH 経由で IdM クライアントへのログインを試みます。

```
[user@client ~]$ ssh idm-user-with-external-idp@client.idm.example.com
(idm-user-with-external-idp@client.idm.example.com) Authenticate at
https://oauth2.idp.com:8443/auth/realms/main/device?user_code=XYFL-ROYR and press
ENTER.
```

2. ブラウザーで、コマンド出力に提供される Web サイトでユーザーとして認証します。
3. コマンドラインで **Enter** キーを押して、認証プロセスを終了します。

検証

- Kerberos チケット情報を表示し、**config: pa_type** の行が外部 IdP による事前認証の **152** を示していることを確認します。

```
[idm-user-with-external-idp@client ~]$ klist -C
Ticket cache: KCM:0:58420
Default principal: idm-user-with-external-idp@IDM.EXAMPLE.COM

Valid starting     Expires            Service principal
05/09/22 07:48:23  05/10/22 07:03:07  krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
config: fast_avail(krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM) = yes
08/17/2022 20:22:45  08/18/2022 20:22:43
krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
config: pa_type(krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM) = 152
```

54.7. IPAIDP ANSIBLE モジュールの PROVIDER オプション

次の ID プロバイダー (IdP) は、OAuth 2.0 デバイス認可グラントフローをサポートしています。

- Azure AD を含む Microsoft Identity Platform

- Google
- GitHub
- Red Hat Single Sign-On (SSO) を含む Keycloak
- Okta

idp ansible-freeipa モジュールを使用してこれらの外部 IdP の 1 つへの参照を作成する場合、**ipaidp ansible-freeipa** Playbook タスクの **provider** オプションで IdP のタイプを指定できます。指定すると、以下で説明する追加オプションをさらに指定できます。

provider: microsoft

Microsoft Azure IdP を使用すると、Azure テナント ID に基づくパラメーター設定を行うことができます。Azure テナント ID は、**organization** オプションで指定できます。live.com IdP のサポートが必要な場合は、オプション **organization common** を指定します。
provider: microsoft を選択すると、次のオプションを使用するように拡張されます。表内の文字列 **${ipaidporg}** は、**organization** オプションの値に置き換えます。

オプション	値
auth_uri: URI	**https://login.microsoftonline.com/${ipaidporg}/oauth2/v2.0/authorize**
dev_auth_uri: URI	**https://login.microsoftonline.com/${ipaidporg}/oauth2/v2.0/devicecode**
token_uri: URI	**https://login.microsoftonline.com/${ipaidporg}/oauth2/v2.0/token**
userinfo_uri: URI	**https://graph.microsoft.com/oidc/userinfo**
keys_uri: URI	**https://login.microsoftonline.com/common/discovery/v2.0/keys**
scope: STR	**openid email**
idp_user_id: STR	**email**

provider: google

provider: google を選択すると、次のオプションを使用するように拡張されます。

オプション	値
auth_uri: URI	**https://accounts.google.com/o/oauth2/auth**
dev_auth_uri: URI	**https://oauth2.googleapis.com/device/code**
token_uri: URI	**https://oauth2.googleapis.com/token**

オプション	値
userinfo_uri: URI	**https://openidconnect.googleapis.com/v1/userinfo**
keys_uri: URI	**https://www.googleapis.com/oauth2/v3/certs**
scope: STR	**openid email**
idp_user_id: STR	**email**

provider: github

provider: github を選択すると、次のオプションを使用するように拡張されます。

オプション	値
auth_uri: URI	**https://github.com/login/oauth/authorize**
dev_auth_uri: URI	**https://github.com/login/device/code**
token_uri: URI	**https://github.com/login/oauth/access_token**
userinfo_uri: URI	**https://openidconnect.googleapis.com/v1/userinfo**
keys_uri: URI	**https://api.github.com/user**
scope: STR	**user**
idp_user_id: STR	**login**

provider: keycloak

Keycloak を使用すると、複数のレルムまたは組織を定義できます。多くの場合、Keycloak はカスタムデプロイメントの一部であるため、ベース URL とレルム ID の両方が必要です。これらは、**ipaidp** Playbook タスクの **base_url** および **organization** オプションで指定できます。

```
---
- name: Playbook to manage IPA idp
  hosts: ipaserver
  become: false

  tasks:
  - name: Ensure keycloak idp my-keycloak-idp is present using provider
    ipaidp:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: my-keycloak-idp
      provider: keycloak
      organization: main
      base_url: keycloak.domain.com:8443/auth
      client_id: my-keycloak-client-id
```

–

provider: keycloak を選択すると、次のオプションを使用するように拡張されます。**base_url** オプションで指定した値により表内の文字列 **${ipaidpbaseurl}** が置き換えられ、**organization** オプションで指定した値により文字列 ${ipaidporg} が置き換えられます。

オプション	値
auth_uri: URI	**https://${ipaidpbaseurl}/realms/${ipaidporg}/protocol/openid-connect/auth**
dev_auth_uri: URI	**https://${ipaidpbaseurl}/realms/${ipaidporg}/protocol/openid-connect/auth/device**
token_uri: URI	**https://${ipaidpbaseurl}/realms/${ipaidporg}/protocol/openid-connect/token**
userinfo_uri: URI	**https://${ipaidpbaseurl}/realms/${ipaidporg}/protocol/openid-connect/userinfo**
scope: STR	**openid email**
idp_user_id: STR	**email**

provider: okta

Okta に新しい組織を登録すると、新しいベース URL が関連付けられます。このベース URL は、**ipaidp** Playbook タスクの **base_url** オプションで指定できます。

```
---
- name: Playbook to manage IPA idp
  hosts: ipaserver
  become: false

  tasks:
  - name: Ensure okta idp my-okta-idp is present using provider
    ipaidp:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: my-okta-idp
      provider: okta
      base_url: dev-12345.okta.com
      client_id: my-okta-client-id
```

provider: okta を選択すると、次のオプションを使用するように拡張されます。**base_url** オプションに指定した値により、テーブル内の文字列 **${ipaidpbaseurl}** が置き換えられます。

オプション	値
auth_uri: URI	**https://${ipaidpbaseurl}/oauth2/v1/authorize**
dev_auth_uri: URI	**https://${ipaidpbaseurl}/oauth2/v1/device/authorize**

オプション	値
token_uri: URI	https://${ipaidpbaseurl}/oauth2/v1/token
userinfo_uri: URI	https://${ipaidpbaseurl}/oauth2/v1/userinfo
scope: STR	openid email
idp_user_id: STR	email

関連情報

- 事前入力された IdP テンプレート

第55章 IDM でのリソースベースの制約付き委任の使用

リソースベースの制約付き委任 (RBCD) を使用して、サービスへのアクセスを委任できます。RBCD を使用すると、リソースレベルで委任をきめ細かく制御でき、認証情報が委任されるサービスの所有者がアクセスを設定できます。これは、たとえば、Identity Management (IdM) と Active Directory (AD) 間の統合に役立ちます。

2019 年以降、Microsoft AD では、ターゲットサービスとプロキシーサービスの両方が異なるフォレストに属している場合に、RBCD の使用が強制されます。

55.1. 関連情報

- IdM での制約付き委任の使用

55.2. IDENTITY MANAGEMENT におけるリソースベースの制約付き委任

リソースベースの制約付き委任 (RBCD) は、次のような複数の点で一般的な制約付き委任とは異なります。

- 粒度: RBCD では、委任はリソースレベルで指定されます。
- アクセス許可の責任: RBCD では、アクセスは Kerberos 管理者ではなくサービス所有者によって制御されます。

一般的な制約付き委任では、Service for User to Proxy (**S4U2proxy**) 拡張機能が、ユーザーに代わって別のサービスのサービスチケットを取得します。2 番目のサービスは通常、ユーザーの承認コンテキストの下で、最初のサービスに代わって作業を実行するプロキシーです。制約付き委任を使用することで、ユーザーが Ticket Granting Ticket (TGT) を完全に委任する必要がなくなります。

Identity Management (IdM) は従来、Kerberos **S4U2proxy** 機能を使用して、Web サーバーフレームワークがユーザーの代わりに LDAP サービスチケットを取得することを可能にするものです。

IdM が Active Directory (AD) と統合されると、IdM フレームワークは制約付き委任も使用して、IdM 側と Active Directory 側の両方の SMB および DCE RPC エンドポイントを含むさまざまなサービスに対してユーザーに代わって動作します。

IdM ドメイン内のアプリケーションがユーザーに代わって別のサービスに対して動作する必要がある場合は、委任権限が必要です。一般的な制約付き委任では、ドメイン管理者が、最初のサービスが次のサービスにユーザー認証情報を委任できるようにするルールを明示的に作成する必要があります。RBCD を使用すると、認証情報が委任されるサービスの所有者が委任権限を作成できます。

IdM-AD 統合で、両方のサービスが同じ IdM ドメインの一部である場合は、IdM 側で RBCD 権限を付与できます。

重要

現在、RBCD ルールで設定できるのは、IdM ドメイン内のサービスのみです。ターゲットサービスが AD ドメインの一部である場合、パーミッションは AD 側でのみ付与できます。AD ドメインコントローラーは IdM サービス情報を解決してルールを作成することができないため、この機能は現在サポートされていません。

55.3. RBCD を使用してサービスへのアクセスを委任する

RBCD を使用してサービスへのアクセスを委任するには、サービスが実行しているホストにルールを追

加する必要があります。この手順例では、Kerberos サービス **HTTP/client.example.test** を使用して、Web アプリケーションのユーザー認証情報をファイルサーバー **nfs/client.example.test** に委任する方法を説明します。ホストは常にそれ自体で実行しているサービスを管理するため、これを **client.example.test** ホストで実行できます。

前提条件

- **client.example.test** ホストの **/etc/krb5.keytab** ファイルにアクセスできる。
- **nfs/client.example.test** サービスのキータブが存在します。
- **HTTP/client.example.test** の keytab **/path/to/web-service.keytab** が存在します。

手順

1. **client.example.test** ホストで、Kerberos チケットを取得します。

```
# kinit -k
```

2. RBCD ACL を定義します。

```
# ipa service-add-delegation nfs/client.example.test HTTP/client.example.test

-----------------------------------------------------
Added new resource delegation to the service principal
"nfs/client.example.test@EXAMPLE.TEST"
-----------------------------------------------------
  Principal name: nfs/client.example.test@EXAMPLE.TEST
  Delegation principal: HTTP/client.example.test@EXAMPLE.TEST
```

検証

委任が正しく設定されていることを確認するには、**HTTP** サービスを介してログインし、**NFS** サービスへのプロトコル移行を実行する **testuser** ユーザーをシミュレートできます。

1. NFS サービスを表示して、委任ルールが存在することを確認します。

```
# ipa service-show nfs/client.example.test

  Principal name: nfs/client.example.test@EXAMPLE.TEST
  Principal alias: nfs/client.example.test@EXAMPLE.TEST
  Delegation principal: HTTP/client.example.test@EXAMPLE.TEST
  Keytab: True
  Managed by: client.example.test
```

2. HTTP サービスプリンシパルの Kerberos チケットを取得します。

```
# kinit -kt http.keytab HTTP/client.example.test
```

3. チケット付与チケットが存在することを確認します。

```
# klist -f
Ticket cache: KCM:0:99799
Default principal: HTTP/client.example.test@EXAMPLE.TEST
```

```
Valid starting       Expires              Service principal
10/13/2023 14:39:23  10/14/2023 14:05:07  krbtgt/EXAMPLE.TEST@EXAMPLE.TEST
 Flags: FIA
```

4. **testuser** に代わってプロトコル移行を実行します。

```
# kvno -U testuser -P nfs/client.example.test
nfs/client.example.test@EXAMPLE.TEST: kvno = 1
```

5. **testuser** に代わってプロトコル移行中に取得したチケットが存在することを確認します。

```
# klist -f
Ticket cache: KCM:0:99799
Default principal: HTTP/client.example.test@EXAMPLE.TEST

Valid starting       Expires              Service principal
10/13/2023 14:39:38  10/14/2023 14:05:07  HTTP/client.example.test@EXAMPLE.TEST
 for client testuser@EXAMPLE.TEST, Flags: FAT
10/13/2023 14:39:23  10/14/2023 14:05:07  krbtgt/EXAMPLE.TEST@EXAMPLE.TEST
 Flags: FIA
10/13/2023 14:39:38  10/14/2023 14:05:07  nfs/client.example.test@EXAMPLE.TEST
 for client testuser@EXAMPLE.TEST, Flags: FAT
```