Red Hat Enterprise Linux 9

Identity Management での外部 Red Hat ユーティリティーの使用

IdM でのサービスと Red Hat 製品の統合

Last Updated: 2024-06-26

Red Hat Enterprise Linux 9 Identity Management での外部 Red Hat ユーティリティーの使用

IdM でのサービスと Red Hat 製品の統合

法律上の通知

概要

管理者は、サービスと Red Hat 製品を Red Hat Identity Management (IdM) ドメインに統合できます。これには、Samba、Ansible、automount などのサービスや、OpenShift Container Platform、OpenStack、Satellite などの製品が含まれます。これにより、IdM ユーザーはこれらのサービスと製品にアクセスできるようになります。

目次

多様性を受け入れるオープンソースの強化

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

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

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

RED HAT ドキュメントへのフィードバック (英語のみ)

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

Jira からのフィードバック送信 (アカウントが必要)

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

第1章 IDM とその他の RED HAT 製品の統合

次のリンクは、IdM と統合される他の Red Hat 製品のドキュメントへのリンクです。IdM ユーザーがサービスにアクセスできるように、これらの製品を設定することができます。

Ansible Automation Platform

Setting up LDAP authentication

OpenShift Container Platform

LDAP アイデンティティープロバイダーの設定

OpenStack Platform

OpenStack Identity (keystone) と Red Hat Identity Manager (IdM) の統合

Satellite

Red Hat Identity Management の使用

Single Sign-On

SSSD および FreeIPA Identity Management の統合

仮想化

Configuring an external LDAP provider

第2章 外部 ID プロバイダーを使用した IDM に対する認証

ユーザーを OAuth 2 デバイス承認フローをサポートする外部アイデンティティープロバイダー (IdP) に関連付けることができます。これらのユーザーが、RHEL 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

2.1. 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。

2.2. 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

2.3. 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 ID プロバイダー (IdP) を Identity Management (IdM) 環境に接続するには、IdM で IdP 参照を作成します。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**

2.4. 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

2.5. 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

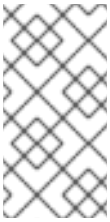
Keycloak と Okta で必要な IdP テンプレートのベース URL

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

2.6. 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
```

2.7. 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
```

2.8. 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。

2.9. 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
```

2.10. IPA IDP-* コマンドの --PROVIDER オプション

次の ID プロバイダー (IdP) は、OAuth 2.0 デバイス認可グラントフローをサポートしています。

- Azure AD を含む Microsoft Identity Platform
- Google

- GitHub
- Red Hat Single Sign-On (SSO) を含む Keycloak
- Okta

ipa idp-add コマンドを使用してこれらの外部 IdP のいずれか 1 つへの参照を作成する場合、**--provider** オプションで IdP タイプを指定できます。これは、以下で説明する追加オプションに拡張されます。

--provider=microsoft

Microsoft Azure IdP では、**--organization** オプションで **ipa idp-add** に指定できる Azure テナント ID に基づくパラメーター化が可能です。live.com IdP のサポートが必要な場合は、**--organization common** オプションを指定します。
--provider=microsoft を選択すると、次のオプションを使用するように拡張されます。**--organization** オプションの値は、表内の文字列 **${ipaidporg}** を置き換えます。

オプション	値
--auth-uri=URI	**https://login.microsoftonline.com/${ipaidporg}/oauth2/v2.0/authorize**
--dev-auth-uri=URI	**https://login.microsoftonline.com/${ipaidporg}/oauth2/v2.0/devicecode**
--token-uri=URI	**https://login.microsoftonline.com/${ipaidporg}/oauth2/v2.0/token**
--userinfo-uri=URI	**https://graph.microsoft.com/oidc/userinfo**
--keys-uri=URI	**https://login.microsoftonline.com/common/discovery/v2.0/keys**
--scope=STR	**openid email**
--idp-user-id=STR	**email**

--provider=google

--provider=google を選択すると、次のオプションを使用するように拡張されます。

オプション	値
--auth-uri=URI	**https://accounts.google.com/o/oauth2/auth**
--dev-auth-uri=URI	**https://oauth2.googleapis.com/device/code**
--token-uri=URI	**https://oauth2.googleapis.com/token**
--userinfo-uri=URI	**https://openidconnect.googleapis.com/v1/userinfo**

オプション	値
--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**
--dev-auth-uri=URI	**https://${ipaidpbaseurl}/realms/${ipaidporg}/protocol/openid-connect/auth/device**

オプション	値
--token-uri=URI	**https://${ipaidpbaseurl}/realms/${ipaidporg}/protocol/openid-connect/token**
--userinfo-uri=URI	**https://${ipaidpbaseurl}/realms/${ipaidporg}/protocol/openid-connect/userinfo**
--scope=STR	**openid email**
--idp-user-id=STR	**email**

--provider=okta

Okta に新しい組織を登録すると、新しいベース URL が関連付けられます。このベース URL は、**ipa idp-add** コマンドの **--base-url** オプションで指定できます。

```
[root@client ~]# ipa idp-add MyOkta --provider okta --base-url dev-12345.okta.com --client-id
<your-client-id>
```

--provider=okta を選択すると、次のオプションを使用するように拡張されます。**--base-url** オプションに指定した値は、表内の文字列 **${ipaidpbaseurl}** を置き換えます。

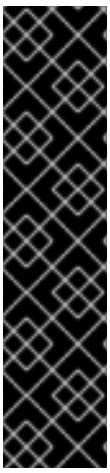
オプション	値
--auth-uri=URI	**https://${ipaidpbaseurl}/oauth2/v1/authorize**
--dev-auth-uri=URI	**https://${ipaidpbaseurl}/oauth2/v1/device/authorize**
--token-uri=URI	**https://${ipaidpbaseurl}/oauth2/v1/token**
--userinfo-uri=URI	**https://${ipaidpbaseurl}/oauth2/v1/userinfo**
--scope=STR	**openid email**
--idp-user-id=STR	**email**

関連情報

- 事前入力された IdP テンプレート

第3章 IDM ドメインメンバーでの SAMBA の設定

Red Hat Identity Management (IdM) ドメインに参加しているホスト上で Samba をセットアップできます。IdM のユーザー、および可能であれば、信頼された Active Directory (AD) ドメインのユーザーは、Samba が提供する共有およびプリンターサービスにアクセスできます。



重要

IdM ドメインメンバーで Samba を使用する機能は、テクノロジープレビュー機能で、特定の制限が含まれています。たとえば、IdM 信頼コントローラーは Active Directory グローバルカタログサービスをサポートしておらず、分散コンピューティング環境/リモートプロシージャコール (DCE/RPC) プロトコルを使用した IdM グループの解決をサポートしていません。結果として、AD ユーザーは、他の IdM クライアントにログインしている場合、IdM クライアントでホストされている Samba 共有とプリンターにのみアクセスできます。Windows マシンにログインしている AD ユーザーは、IdM ドメインメンバーでホストされている Samba 共有にアクセスできません。

IdM ドメインメンバーに Samba をデプロイしているお客様は、ぜひ Red Hat にフィードバックをお寄せください。

AD ドメインのユーザーが Samba によって提供される共有およびプリンターサービスにアクセスする必要がある場合は、AES 暗号化タイプが AD になっていることを確認してください。詳細は、GPO を使用した Active Directory での AES 暗号化タイプの有効化 を参照してください。

前提条件

- ホストは、クライアントとして IdM ドメインに参加している。
- IdM サーバーとクライアントの両方が RHEL 9.0 以降で実行されている必要がある。

3.1. SAMBA をドメインメンバーにインストールするための IDM ドメインの準備

IdM クライアントに Samba を設定する前に、IdM サーバーで **ipa-adtrust-install** ユーティリティーを使用して IdM ドメインを準備する必要があります。

注記

ipa-adtrust-install コマンドを自動的に実行するシステムは、AD 信頼コントローラーになります。ただし、**ipa-adtrust-install** は、IdM サーバーで 1 回のみ実行する必要があります。

前提条件

- IdM サーバーがインストールされている。
- パッケージをインストールし、IdM サービスを再起動するには、root 権限が必要です。

手順

1. 必要なパッケージをインストールします。

```
[root@ipaserver ~]# dnf install ipa-server-trust-ad samba-client
```

2. IdM 管理ユーザーとして認証します。

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

3. **ipa-adtrust-install** ユーティリティーを実行します。

```
[root@ipaserver ~]# ipa-adtrust-install
```

統合 DNS サーバーとともに IdM がインストールされていると、DNS サービスレコードが自動的に作成されます。

IdM が統合 DNS サーバーなしで IdM をインストールすると、**ipa-adtrust-install** は、続行する前に DNS に手動で追加する必要があるサービスレコードのリストを出力します。

4. スクリプトにより、**/etc/samba/smb.conf** がすでに存在し、書き換えられることが求められます。

```
WARNING: The smb.conf already exists. Running ipa-adtrust-install will break your existing
Samba configuration.

Do you wish to continue? [no]: yes
```

5. このスクリプトは、従来の Linux クライアントが信頼できるユーザーと連携できるようにする互換性プラグインである **slapi-nis** プラグインを設定するように求めるプロンプトを表示します。

```
Do you want to enable support for trusted domains in Schema Compatibility plugin?
This will allow clients older than SSSD 1.9 and non-Linux clients to work with trusted users.

Enable trusted domains support in slapi-nis? [no]: yes
```

6. SID 生成タスクを実行して、既存ユーザーに SID を作成するように求められます。

```
Do you want to run the ipa-sidgen task? [no]: yes
```

これはリソースを集中的に使用するタスクであるため、ユーザー数が多い場合は別のタイミングで実行できます。

7. **(必要に応じて)** デフォルトでは、Windows Server 2008 以降での動的 RPC ポートの範囲は **49152-65535** として定義されます。ご使用の環境に異なる動的 RPC ポート範囲を定義する必要がある場合は、Samba が異なるポートを使用するように設定し、ファイアウォール設定でそのポートを開くように設定します。以下の例では、ポート範囲を **55000-65000** に設定します。

```
[root@ipaserver ~]# net conf setparm global 'rpc server dynamic port range' 55000-
65000
[root@ipaserver ~]# firewall-cmd --add-port=55000-65000/tcp
[root@ipaserver ~]# firewall-cmd --runtime-to-permanent
```

8. **ipa** サービスを再起動します。

```
[root@ipaserver ~]# ipactl restart
```

9. **smbclient** ユーティリティーを使用して、Samba が IdM からの Kerberos 認証に応答することを確認します。

```
[root@ipaserver ~]# smbclient -L ipaserver.idm.example.com -U user_name --use-kerberos=required
lp_load_ex: changing to config backend registry
    Sharename       Type      Comment
    ---------       ----      -------
    IPC$            IPC       IPC Service (Samba 4.15.2)
...
```

3.2. IDM クライアントでの SAMBA サーバーのインストールおよび設定

IdM ドメインに登録されたクライアントに Samba をインストールして設定できます。

前提条件

- IdM サーバーとクライアントの両方が RHEL 9.0 以降で実行されている必要がある。
- IdM ドメインは、ドメインメンバーに Samba をインストールするための IdM ドメインの準備 の説明に従って準備されます。
- IdM に AD で設定された信頼がある場合は、Kerberos の AES 暗号化タイプを有効にします。たとえば、グループポリシーオブジェクト (GPO) を使用して、AES 暗号化の種類を有効にします。詳細は、GPO を使用した Active Directory での AES 暗号化の有効化 を参照してください。

手順

1. **ipa-client-samba** パッケージをインストールします。

```
[root@idm_client]# dnf install ipa-client-samba
```

2. **ipa-client-samba** ユーティリティーを使用して、クライアントを準備し、初期 Samba 設定を作成します。

```
[root@idm_client]# ipa-client-samba
Searching for IPA server...
IPA server: DNS discovery
Chosen IPA master: idm_server.idm.example.com
SMB principal to be created: cifs/idm_client.idm.example.com@IDM.EXAMPLE.COM
NetBIOS name to be used: IDM_CLIENT
Discovered domains to use:

 Domain name: idm.example.com
NetBIOS name: IDM
        SID: S-1-5-21-525930803-952335037-206501584
   ID range: 212000000 - 212199999

 Domain name: ad.example.com
NetBIOS name: AD
        SID: None
   ID range: 1918400000 - 1918599999

Continue to configure the system with these values? [no]: yes
Samba domain member is configured. Please check configuration at /etc/samba/smb.conf
and start smb and winbind services
```

3. デフォルトでは、**ipa-client-samba** は、ユーザーが接続したときにそのユーザーのホームディレクトリーを動的に共有するために、**/etc/samba/smb.conf** ファイルに **[homes]** セクションが自動的に追加されます。ユーザーがこのサーバーにホームディレクトリーがない場合、または共有したくない場合は、**/etc/samba/smb.conf** から次の行を削除します。

```
[homes]
    read only = no
```

4. ディレクトリーとプリンターを共有します。詳細は、次のセクションを参照してください。
   - POSIX ACL を使用した Samba ファイル共有の設定
   - Windows ACL で共有の設定
   - プリントサーバーとしての Samba の設定
5. ローカルファイアウォールで Samba クライアントに必要なポートを開きます。

```
[root@idm_client]# firewall-cmd --permanent --add-service=samba-client
[root@idm_client]# firewall-cmd --reload
```

6. **smb** サービスおよび **winbind** サービスを有効にして開始します。

```
[root@idm_client]# systemctl enable --now smb winbind
```

検証手順

samba-client パッケージがインストールされている別の IdM ドメインメンバーで次の検証手順を実行します。

- Kerberos 認証を使用して、Samba サーバー上の共有をリスト表示します。

```
$ smbclient -L idm_client.idm.example.com -U user_name --use-kerberos=required
lp_load_ex: changing to config backend registry

    Sharename       Type      Comment
    ---------       ----      -------
    example         Disk
    IPC$            IPC       IPC Service (Samba 4.15.2)
...
```

関連情報

- **ipa-client-samba(1)** の man ページ

3.3. IDM が新しいドメインを信頼する場合は、ID マッピング設定を手動で追加

Samba では、ユーザーがリソースにアクセスする各ドメインの ID マッピング設定が必要です。IdM クライアントで実行している既存の Samba サーバーでは、管理者が Active Directory (AD) ドメインに新しい信頼を追加した後、ID マッピング設定を手動で追加する必要があります。

前提条件

- IdM クライアントで Samba を設定している。その後、IdM に新しい信頼が追加されている。
- Kerberos の暗号化タイプ DES および RC4 は、信頼できる AD ドメインで無効にしている。セキュリティー上の理由から、RHEL 9 はこのような弱い暗号化タイプに対応していません。

手順

1. ホストのキータブを使用して認証します。

```
[root@idm_client]# kinit -k
```

2. **ipa idrange-find** コマンドを使用して、新しいドメインのベース ID と ID 範囲のサイズの両方を表示します。たとえば、次のコマンドは **ad.example.com** ドメインの値を表示します。

```
[root@idm_client]# ipa idrange-find --name="AD.EXAMPLE.COM_id_range" --raw
---------------
1 range matched
---------------
  cn: AD.EXAMPLE.COM_id_range
  ipabaseid: 1918400000
  ipaidrangesize: 200000
  ipabaserid: 0
  ipanttrusteddomainsid: S-1-5-21-968346183-862388825-1738313271
  iparangetype: ipa-ad-trust
----------------------------
Number of entries returned 1
----------------------------
```

次の手順で、**ipabaseid** 属性および **ipaidrangesize** 属性の値が必要です。

3. 使用可能な最高の ID を計算するには、次の式を使用します。

```
maximum_range = ipabaseid + ipaidrangesize - 1
```

前の手順の値を使用すると、**ad.example.com** ドメインで使用可能な最大 ID は **1918599999** (1918400000 + 200000 - 1) です。

4. **/etc/samba/smb.conf** ファイルを編集し、ドメインの ID マッピング設定を **[global]** セクションに追加します。

```
idmap config AD : range = 1918400000 - 1918599999
idmap config AD : backend = sss
```

ipabaseid 属性の値を最小値として指定し、前の手順で計算された値を範囲の最大値として指定します。

5. **smb** サービスおよび **winbind** サービスを再起動します。

```
[root@idm_client]# systemctl restart smb winbind
```

検証手順

- Kerberos 認証を使用して、Samba サーバー上の共有をリスト表示します。

```
$ smbclient -L idm_client.idm.example.com -U user_name --use-kerberos=required
lp_load_ex: changing to config backend registry

    Sharename       Type      Comment
    ---------       ----      -------
    example         Disk
    IPC$            IPC       IPC Service (Samba 4.15.2)
...
```

3.4. 関連情報

- Installing an Identity Management client

第4章 NIS から IDENTITY MANAGEMENT への移行

ネットワーク情報サービス (NIS) サーバーには、ユーザー、グループ、ホスト、netgroups、および自動マウントマップに関する情報を追加できます。システム管理者は、全ユーザー管理操作が IdM サーバーで実行されるように、これらのエントリータイプ、認証、および承認を NIS サーバーから Identity Management (IdM) サーバーに移行できます。NIS から IdM に移行すると、Kerberos などのよりセキュアなプロトコルを利用できます。

4.1. IDM での NIS の有効化

NIS と Identity Management (IdM) サーバー間で通信できるようにするには、IdM サーバーで NIS 互換性オプションを有効にする必要があります。

前提条件

- IdM サーバーの root アクセス権限がある。

手順

1. IdM サーバーで NIS リスナーと互換性プラグインを有効にします。

```
[root@ipaserver ~]# ipa-nis-manage enable
[root@ipaserver ~]# ipa-compat-manage enable
```

2. **必要に応じて**、より厳密なファイアウォール設定を行うには、固定ポートを設定します。たとえば、ポートを未使用のポート **514** に設定するには、次のコマンドを実行します。

```
[root@ipaserver ~]# ldapmodify -x -D 'cn=directory manager' -W
dn: cn=NIS Server,cn=plugins,cn=config
changetype: modify
add: nsslapd-pluginarg0
nsslapd-pluginarg0: 514
```

警告

他のサービスとの競合を回避するため、1024 を超えるポート番号は使用しないようにしてください。

3. ポートマッパーサービスを有効にして起動します。

```
[root@ipaserver ~]# systemctl enable rpcbind.service
[root@ipaserver ~]# systemctl start rpcbind.service
```

4. Directory Server を再起動します。

```
[root@ipaserver ~]# systemctl restart dirsrv.target
```

4.2. NIS から IDM へのユーザーエントリーの移行

NIS の **passwd** マップには、名前、UID、プライマリーグループ、GECOS、シェル、ホームディレクトリーなどのユーザーに関する情報が含まれます。このデータを使用して、NIS ユーザーアカウントを Identity Management (IdM) に移行します。

前提条件

- NIS サーバーの root アクセス権限がある。
- IdM で NIS が有効になっている。
- NIS サーバーが IdM に登録されている。

手順

1. **yp-tools** パッケージをインストールします。

```
[root@nis-server ~]# dnf install yp-tools -y
```

2. NIS サーバーで、以下の内容を含む **/root/nis-users.sh** スクリプトを作成します。

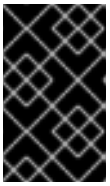
```
#!/bin/sh
# $1 is the NIS domain, $2 is the primary NIS server
ypcat -d $1 -h $2 passwd > /dev/shm/nis-map.passwd 2>&1

IFS=$'\n'
for line in $(cat /dev/shm/nis-map.passwd) ; do
 IFS=' '
 username=$(echo $line | cut -f1 -d:)
 # Not collecting encrypted password because we need cleartext password
 # to create kerberos key
 uid=$(echo $line | cut -f3 -d:)
 gid=$(echo $line | cut -f4 -d:)
 gecos=$(echo $line | cut -f5 -d:)
 homedir=$(echo $line | cut -f6 -d:)
 shell=$(echo $line | cut -f7 -d:)

 # Now create this entry
 echo passw0rd1 | ipa user-add $username --first=NIS --last=USER \
     --password --gidnumber=$gid --uid=$uid --gecos="$gecos" --homedir=$homedir \
     --shell=$shell
 ipa user-show $username
done
```

3. IdM **admin** ユーザーとして認証します。

```
[root@nis-server ~]# kinit admin
```

4. スクリプトを実行します。以下に例を示します。

```
[root@nis-server ~]# sh /root/nis-users.sh nisdomain nis-server.example.com
```



重要

このスクリプトは、名、姓にハードコードされた値を使用し、パスワードを **passw0rd1** に設定します。ユーザーは、次回ログイン時に一時パスワードを変更する必要があります。

4.3. ユーザーグループの NIS から IDM への移行

NIS **グループ** マップには、グループ名、GID、グループメンバーなどのグループ情報が含まれます。このデータを使用して、NIS グループを Identity Management (IdM) に移行します。

前提条件

- NIS サーバーの root アクセス権限がある。
- IdM で NIS が有効になっている。
- NIS サーバーが IdM に登録されている。

手順

1. **yp-tools** パッケージをインストールします。

```
[root@nis-server ~]# dnf install yp-tools -y
```

2. 以下の内容を含めて、NIS サーバーに **/root/nis-groups.sh** スクリプトを作成します。

```
#!/bin/sh
# $1 is the NIS domain, $2 is the primary NIS server
ypcat -d $1 -h $2 group > /dev/shm/nis-map.group 2>&1

IFS=$'\n'
for line in $(cat /dev/shm/nis-map.group); do
 IFS=' '
 groupname=$(echo $line | cut -f1 -d:)
 # Not collecting encrypted password because we need cleartext password
 # to create kerberos key
 gid=$(echo $line | cut -f3 -d:)
 members=$(echo $line | cut -f4 -d:)

 # Now create this entry
 ipa group-add $groupname --desc=NIS_GROUP_$groupname --gid=$gid
 if [ -n "$members" ]; then
  ipa group-add-member $groupname --users={$members}
 fi
 ipa group-show $groupname
done
```

3. IdM **admin** ユーザーとして認証します。

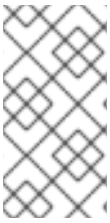
```
[root@nis-server ~]# kinit admin
```

4. スクリプトを実行します。以下に例を示します。

```
[root@nis-server ~]# sh /root/nis-groups.sh nisdomain nis-server.example.com
```

4.4. ホストエントリーの NIS から IDM への移行

NIS **ホスト** マップには、ホスト名や IP アドレスなどのホストに関する情報が含まれます。このデータを使用して、NIS ホストエントリーを Identity Management (IdM) に移行します。



注記

IdM でホストグループを作成すると、対応するシャドウの NIS グループが自動的に作成されます。これらのシャドウ NIS グループに **ipa netgroup-*** コマンドを使用しないでください。**ipa netgroup-*** コマンドは、**netgroup-add** コマンドで作成されたネイティブの netgroups の管理にだけ使用します。

前提条件

- NIS サーバーの root アクセス権限がある。
- IdM で NIS が有効になっている。
- NIS サーバーが IdM に登録されている。

手順

1. **yp-tools** パッケージをインストールします。

```
[root@nis-server ~]# dnf install yp-tools -y
```

2. 以下の内容を含めて、NIS サーバーに **/root/nis-hosts.sh** スクリプトを作成します。

```
#!/bin/sh
# $1 is the NIS domain, $2 is the primary NIS server
ypcat -d $1 -h $2 hosts | egrep -v "localhost|127.0.0.1" > /dev/shm/nis-map.hosts 2>&1

IFS=$'\n'
for line in $(cat /dev/shm/nis-map.hosts); do
 IFS=' '
 ipaddress=$(echo $line | awk '{print $1}')
 hostname=$(echo $line | awk '{print $2}')
 primary=$(ipa env xmlrpc_uri | tr -d '[:space:]' | cut -f3 -d: | cut -f3 -d/)
 domain=$(ipa env domain | tr -d '[:space:]' | cut -f2 -d:)
 if [ $(echo $hostname | grep "\." |wc -l) -eq 0 ] ; then
  hostname=$(echo $hostname.$domain)
 fi
 zone=$(echo $hostname | cut -f2- -d.)
 if [ $(ipa dnszone-show $zone 2>/dev/null | wc -l) -eq 0 ] ; then
  ipa dnszone-add --name-server=$primary --admin-email=root.$primary
 fi
 ptrzone=$(echo $ipaddress | awk -F. '{print $3 "." $2 "." $1 ".in-addr.arpa."}')
 if [ $(ipa dnszone-show $ptrzone 2>/dev/null | wc -l) -eq 0 ] ; then
  ipa dnszone-add  $ptrzone --name-server=$primary --admin-email=root.$primary
 fi
 # Now create this entry
```

```
 ipa host-add $hostname --ip-address=$ipaddress
 ipa host-show $hostname
done
```

3. IdM **admin** ユーザーとして認証します。

```
[root@nis-server ~]# kinit admin
```

4. スクリプトを実行します。以下に例を示します。

```
[root@nis-server ~]# sh /root/nis-hosts.sh nisdomain nis-server.example.com
```

注記

このスクリプトでは、エイリアスなどの特別なホスト設定は移行されません。

4.5. NETGROUP エントリーの NIS から IDM への移行

NIS **netgroup** マップには、netgroup に関する情報が含まれます。このデータを使用して、NIS netgroup を Identity Management (IdM) に移行します。

前提条件

- NIS サーバーの root アクセス権限がある。
- IdM で NIS が有効になっている。
- NIS サーバーが IdM に登録されている。

手順

1. **yp-tools** パッケージをインストールします。

```
[root@nis-server ~]# dnf install yp-tools -y
```

2. 以下の内容を含めて NIS サーバーに **/root/nis-netgroups.sh** スクリプトを作成します。

```
#!/bin/sh
# $1 is the NIS domain, $2 is the primary NIS server
ypcat -k -d $1 -h $2 netgroup > /dev/shm/nis-map.netgroup 2>&1

IFS=$'\n'
for line in $(cat /dev/shm/nis-map.netgroup); do
 IFS=' '
 netgroupname=$(echo $line | awk '{print $1}')
 triples=$(echo $line | sed "s/^$netgroupname //")
 echo "ipa netgroup-add $netgroupname --desc=NIS_NG_$netgroupname"
 if [ $(echo $line | grep "(," | wc -l) -gt 0 ]; then
  echo "ipa netgroup-mod $netgroupname --hostcat=all"
 fi
 if [ $(echo $line | grep ",," | wc -l) -gt 0 ]; then
  echo "ipa netgroup-mod $netgroupname --usercat=all"
 fi
```

```
 for triple in $triples; do
  triple=$(echo $triple | sed -e 's/-//g' -e 's/(//' -e 's/)//')
  if [ $(echo $triple | grep ",.*," | wc -l) -gt 0 ]; then
   hostname=$(echo $triple | cut -f1 -d,)
   username=$(echo $triple | cut -f2 -d,)
   domain=$(echo $triple | cut -f3 -d,)
   hosts=""; users=""; doms="";
   [ -n "$hostname" ] && hosts="--hosts=$hostname"
   [ -n "$username" ] && users="--users=$username"
   [ -n "$domain"   ] && doms="--nisdomain=$domain"
   echo "ipa netgroup-add-member $netgroup $hosts $users $doms"
  else
   netgroup=$triple
   echo "ipa netgroup-add $netgroup --desc=<NIS_NG>_$netgroup"
  fi
 done
done
```

3. IdM **admin** ユーザーとして認証します。

```
[root@nis-server ~]# kinit admin
```

4. スクリプトを実行します。以下に例を示します。

```
[root@nis-server ~]# sh /root/nis-netgroups.sh nisdomain nis-server.example.com
```

4.6. NIS から IDM への自動マウントマップの移行

自動マウントマップは、場所 (親エントリー)、関連のキー、およびマップを定義する入れ子および相互関連のエントリーです。NIS 自動マウントマップを Identity Management (IdM) に移行するには、以下を実行します。

前提条件

- NIS サーバーの root アクセス権限がある。
- IdM で NIS が有効になっている。
- NIS サーバーが IdM に登録されている。

手順

1. **yp-tools** パッケージをインストールします。

```
[root@nis-server ~]# dnf install yp-tools -y
```

2. 以下の内容を含めて、NIS サーバーに **/root/nis-automounts.sh** スクリプトを作成します。

```
#!/bin/sh
# $1 is for the automount entry in ipa

ipa automountlocation-add $1
```

```
# $2 is the NIS domain, $3 is the primary NIS server, $4 is the map name

ypcat -k -d $2 -h $3 $4 > /dev/shm/nis-map.$4 2>&1

ipa automountmap-add $1 $4

basedn=$(ipa env basedn | tr -d '[:space:]' | cut -f2 -d:)
cat > /tmp/amap.ldif <<EOF
dn: nis-domain=$2+nis-map=$4,cn=NIS Server,cn=plugins,cn=config
objectClass: extensibleObject
nis-domain: $2
nis-map: $4
nis-base: automountmapname=$4,cn=$1,cn=automount,$basedn
nis-filter: (objectclass=\*)
nis-key-format: %{automountKey}
nis-value-format: %{automountInformation}
EOF
ldapadd -x -h $3 -D "cn=Directory Manager" -W -f /tmp/amap.ldif

IFS=$'\n'
for line in $(cat /dev/shm/nis-map.$4); do
 IFS=" "
 key=$(echo "$line" | awk '{print $1}')
 info=$(echo "$line" | sed -e "s^$key[ \t]*")
 ipa automountkey-add nis $4 --key="$key" --info="$info"
done
```

注記

このスクリプトでは、NIS 自動マウント情報のエクスポート、自動マウントの場所と関連マップの LDAP データ交換形式 (LDIF) の生成、IdM Directory Server への LDIF ファイルのインポートが行われます。

3. IdM **admin** ユーザーとして認証します。

```
[root@nis-server ~]# kinit admin
```

4. スクリプトを実行します。以下に例を示します。

```
[root@nis-server ~]# sh /root/nis-automounts.sh location nisdomain
    nis-server.example.com map_name
```

第5章 IDM で自動マウントの使用

自動マウントは、複数のシステムにわたってディレクトリーを管理、整理、およびアクセスする方法です。Automount は、ディレクトリーへのアクセスが要求されるたびに、そのディレクトリーを自動的にマウントします。これは、ドメイン内のクライアント上のディレクトリーを簡単に共有できるため、Identity Management (IdM) ドメイン内でうまく機能します。

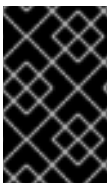
この例では、以下のシナリオを使用します。

- **nfs-server.idm.example.com** は、ネットワークファイルシステム (NFS) サーバーの完全修飾ドメイン名 (FQDN) です。
- 便宜上、**nfs-server.idm.example.com** は、**raleigh** 自動マウントの場所のマップを提供する IdM クライアントとします。

注記

自動マウントの場所は、NFS マップの一意のセットです。たとえば、クライアントが高速接続の恩恵を受けられるように、これらの NFS マップがすべて同じ地理的地域にあることが理想的ですが、これは必須ではありません。

- NFS サーバーは、**/exports/project** ディレクトリーを読み取り/書き込みとしてエクスポートします。
- **developers** グループに属する IdM ユーザーは、**raleigh**の自動マウントの場所を使用する IdM クライアントであれば、**/devel/project/** として、エクスポートされたディレクトリーのコンテンツにアクセスできます。
- **idm-client.idm.example.com** は、**raleigh** の自動マウントの場所を使用する IdM クライアントです。



重要

NFS サーバーの代わりに Samba サーバーを使用して IdM クライアントに共有を提供する場合は、以下の How do I configure kerberized CIFS mounts with Autofs in an IPA environment? を参照してください。を参照してください。

5.1. IDM の AUTOFS と AUTOMOUNT

autofs サービスは、アクセス時にディレクトリーをマウントするように **automount** デーモンに指示することにより、必要に応じてディレクトリーのマウントを自動化します。また、しばらく操作を行わないと、**autofs** は、**automount** に自動マウントされたディレクトリーのマウントを解除するように指示します。静的マウントとは異なり、オンデマンドマウントはシステムリソースを節約します。

マップの自動マウント

autofs を使用するシステムでは、**automount** 設定は複数のファイルに保存されます。プライマリー **automount** 設定ファイルは **/etc/auto.master** です。これには、システムの **automount** マウントポイントのマスターマッピングと、その関連リソースが含まれます。このマッピングは **自動マウントマップ** として知られています。
/etc/auto.master 設定ファイルには、**マスターマップ** が含まれます。他のマップへの参照を含めることができます。このマップは、直接または間接のいずれかになります。ダイレクトマップではマウントポイントに絶対パス名を使用し、間接マップでは相対パス名を使用します。

IdM の自動マウント設定

automount は通常、ローカルの **/etc/auto.master** と関連ファイルからマップデータを取得しますが、他のソースからマップデータを取得することもできます。一般的なソースの 1 つが LDAP サーバーです。Identity Management (Id M) のコンテキストでは、これは 389 Directory Server です。**autofs** を使用するシステムが IdM ドメインのクライアントである場合、**automount** 設定はローカル設定ファイルに保存されません。代わりに、マップ、場所、キーなどの **autofs** 設定は、LDAP エントリーとして IdM ディレクトリーに保存されます。たとえば、**idm.example.com** IdM ドメインの場合、デフォルトの **マスターマップ** は以下のように保存されます。

```
dn:
automountmapname=auto.master,cn=default,cn=automount,dc=idm,dc=example,dc=com
objectClass: automountMap
objectClass: top
automountMapName: auto.master
```

関連情報

- オンデマンドでのファイルシステムのマウント

5.2. RED HAT IDENTITY MANAGEMENT ドメインで KERBEROS を使用する NFS サーバーをセットアップする

Red Hat Identity Management (IdM) を使用すると、NFS サーバーを IdM ドメインに参加させることができます。これにより、ユーザーとグループを一元管理し、認証、整合性保護、トラフィック暗号化に Kerberos を使用できるようになります。

前提条件

- NFS サーバーが Red Hat Identity Management (IdM) ドメインに 登録 されている。
- NFS サーバーが実行および設定されている。

手順

1. IdM 管理者として Kerberos チケットを取得します。

```
# kinit admin
```

2. **nfs/<FQDN>** サービスプリンシパルを作成します。

```
# ipa service-add nfs/nfs_server.idm.example.com
```

3. IdM から **nfs** サービスプリンシパルを取得し、**/etc/krb5.keytab** ファイルに保存します。

```
# ipa-getkeytab -s idm_server.idm.example.com -p nfs/nfs_server.idm.example.com -k
/etc/krb5.keytab
```

4. オプション: **/etc/krb5.keytab** ファイル内のプリンシパルを表示します。

```
# klist -k /etc/krb5.keytab
Keytab name: FILE:/etc/krb5.keytab
KVNO Principal
---- --------------------------------------------------------------------------
```

```
1 nfs/nfs_server.idm.example.com@IDM.EXAMPLE.COM
1 nfs/nfs_server.idm.example.com@IDM.EXAMPLE.COM
1 nfs/nfs_server.idm.example.com@IDM.EXAMPLE.COM
1 nfs/nfs_server.idm.example.com@IDM.EXAMPLE.COM
7 host/nfs_server.idm.example.com@IDM.EXAMPLE.COM
7 host/nfs_server.idm.example.com@IDM.EXAMPLE.COM
7 host/nfs_server.idm.example.com@IDM.EXAMPLE.COM
7 host/nfs_server.idm.example.com@IDM.EXAMPLE.COM
```

デフォルトでは、ホストを IdM ドメインに参加させると、IdM クライアントがホストプリンシパルを **/etc/krb5.keytab** ファイルに追加します。ホストプリンシパルがない場合は、**ipa-getkeytab -s idm_server.idm.example.com -p host/nfs_server.idm.example.com -k /etc/krb5.keytab** コマンドを使用して追加します。

5. **ipa-client-automount** ユーティリティーを使用して、IdM ID のマッピングを設定します。

```
# ipa-client-automount
Searching for IPA server...
IPA server: DNS discovery
Location: default
Continue to configure the system with these values? [no]: yes
Configured /etc/idmapd.conf
Restarting sssd, waiting for it to become available.
Started autofs
```

6. **/etc/exports** ファイルを更新し、クライアントオプションに Kerberos セキュリティー方式を追加します。以下に例を示します。

```
/nfs/projects/      192.0.2.0/24(rw,sec=krb5i)
```

クライアントが複数のセキュリティー方式を選択できるようにするには、それらをコロンで区切って指定します。

```
/nfs/projects/      192.0.2.0/24(rw,sec=krb5:krb5i:krb5p)
```

7. エクスポートされたファイルシステムを再ロードします。

```
# exportfs -r
```

5.3. IDM CLI を使用した IDM での自動マウントの場所とマップの設定

場所はマップのセットで、すべて **auto.master** に保存されます。1 つの場所に複数のマップを保存できます。また、場所には複数のマップを保存できます。場所のエントリーは、マップエントリーのコンテナーとしてのみ機能します。それ自体は、自動マウント設定ではありません。

Identity Management (IdM) のシステム管理者は、IdM で自動マウントの場所とマップを設定できます。これにより、指定した場所の IdM ユーザーが、ホストの特定のマウントポイントに移動して、NFS サーバーがエクスポートした共有にアクセスできるようになります。エクスポートされた NFS サーバーディレクトリーとマウントポイントの両方が、マップで指定されます。この例では、**raleigh** の場所と、IdM クライアントの**/devel/** マウントポイントにある **nfs-server.idm.example.com:/exports/project** 共有を読み取り/書き込みディレクトリーとしてマウントするマップを設定する方法を説明します。

前提条件

- IdM に登録されているホストに IdM 管理者としてログインしている。

手順

1. **raleigh** の自動マウントの場所を作成します。

```
$ ipa automountlocation-add raleigh
---------------------------------
Added automount location "raleigh"
---------------------------------
  Location: raleigh
```

2. **raleigh**の場所に、**auto.devel** 自動マウントマップを作成します。

```
$ ipa automountmap-add raleigh auto.devel
--------------------------------
Added automount map "auto.devel"
--------------------------------
  Map: auto.devel
```

3. **exports/**共有のキーとマウント情報を追加します。

   a. **auto.devel** マップのキーとマウント情報を追加します。

```
$ ipa automountkey-add raleigh auto.devel --key='*' --info='-sec=krb5p,vers=4 nfs-server.idm.example.com:/exports/&'
----------------------
Added automount key "*"
----------------------
  Key: *
  Mount information: -sec=krb5p,vers=4 nfs-server.idm.example.com:/exports/&
```

   b. **auto.master** マップのキーとマウント情報を追加します。

```
$ ipa automountkey-add raleigh auto.master --key=/devel --info=auto.devel
---------------------------
Added automount key "/devel"
---------------------------
  Key: /devel
  Mount information: auto.devel
```

5.4. IDM クライアントでの自動マウントの設定

Identity Management (IdM) システム管理者は、IdM クライアントに自動マウントサービスを設定して、クライアントが追加された場所に設定した NFS 共有に、ユーザーがクライアントにログインしたときに IdM ユーザーが自動的にアクセスできるようにすることができます。この例では、**raleigh** の場所で利用可能な自動マウントサービスを使用するように IdM クライアントを設定する方法を説明します。

前提条件

- IdM クライアントへの **root** アクセス権限がある。

- IdM 管理者としてログインしている。
- 自動マウントの場所が存在します。サンプルの場所は **raleigh** です。

手順

1. IdM クライアントで、**ipa-client-automount** コマンドを入力して場所を指定します。**-U** オプションを使用して、スクリプトを無人で実行します。

```
# ipa-client-automount --location raleigh -U
```

2. autofs サービスを停止し、SSSD キャッシュをクリアし、autofs サービスを開始して新しい設定をロードします。

```
# systemctl stop autofs ; sss_cache -E ; systemctl start autofs
```

5.5. IDM クライアントで、IDM ユーザーが NFS 共有にアクセスできることの確認

Identity Management (IdM) システム管理者は、特定のグループのメンバーである IdM ユーザーが、特定の IdM クライアントにログインしたときに NFS 共有にアクセスできるかどうかをテストできます。

この例では、以下のシナリオがテストされています。

- **developers** グループに属する **idm_user** という名前の IdM ユーザーは、 **raleigh** automount ロケーションにある IdM クライアントである **idm-client.idm.example.com** に自動マウントされた **/devel/project** ディレクトリー内のファイルの内容を読み書きできます。

前提条件

- IdM ホスト上に Kerberos を使用して NFS サーバーをセットアップした 。
- IdM で自動マウントのロケーション、マップ、マウントポイントを設定 し、そこで IdM ユーザーが NFS共有にアクセスできるように設定している。
- IdM クライアントに自動マウントを設定 している。

手順

1. IdM ユーザーが **read-write** ディレクトリーにアクセスできることを確認します。
   a. IdM ユーザーとして IdM クライアントに接続します。

```
$ ssh idm_user@idm-client.idm.example.com
Password:
```

   b. IdM ユーザーの Ticket Granting Ticket (TGT) を取得します。

```
$ kinit idm_user
```

   c. [オプション] Id M ユーザーのグループメンバーシップを表示します。

```
$ ipa user-show idm_user
```

```
User login: idm_user
[...]
Member of groups: developers, ipausers
```

d. **/devel/project** ディレクトリーに移動します。

```
$ cd /devel/project
```

e. ディレクトリーの内容をリスト表示します。

```
$ ls
rw_file
```

f. ディレクトリーのファイルに行を追加し、**write** パーミッションをテストします。

```
$ echo "idm_user can write into the file" > rw_file
```

g. [オプション] ファイルの更新された内容を表示します。

```
$ cat rw_file
this is a read-write file
idm_user can write into the file
```

出力は、**idm_user** がファイルに書き込めることを確認します。

第6章 ANSIBLE を使用して IDM ユーザーの NFS 共有を自動マウントする

自動マウントは、複数のシステムにわたってディレクトリーを管理、整理、およびアクセスする方法です。Automount は、ディレクトリーへのアクセスが要求されるたびに、そのディレクトリーを自動的にマウントします。これは、ドメイン内のクライアント上のディレクトリーを簡単に共有できるため、Identity Management (IdM) ドメイン内でうまく機能します。

Ansible を使用して、IdM ロケーションの IdM クライアントにログインしている IdM ユーザーに対して NFS 共有が自動的にマウントされるように設定できます。

この章の例では、次のシナリオを使用します。

- **nfs-server.idm.example.com** は、ネットワークファイルシステム (NFS) サーバーの完全修飾ドメイン名 (FQDN) です。
- **nfs-server.idm.example.com** は、**raleigh** の自動マウントの場所にある IdM クライアントです。
- NFS サーバーは、**/exports/project** ディレクトリーを読み取り/書き込みとしてエクスポートします。
- **開発者** グループに属する IdM ユーザーは、NFS サーバーと同じ **raleigh**の自動マウントの場所にある IdM クライアントであれば、**/devel/project/** としてエクスポートしたディレクトリーのコンテンツにアクセスできます。
- **idm-client.idm.example.com** は、**raleigh** の自動マウントの場所にある IdM クライアントです。

重要

NFS サーバーの代わりに Samba サーバーを使用して IdM クライアントに共有を提供する場合は、以下の How do I configure kerberized CIFS mounts with Autofs in an IPA environment? を参照してください。を参照してください。

この章には次のセクションが含まれています。

1. IdM の Autofs と automount
2. IdM で Kerberos を使用する NFS サーバーをセットアップする
3. Ansible を使用した IdM での自動マウントの場所、マップ、およびキーの設定
4. Ansible を使用して IdM ユーザーを NFS 共有を所有するグループに追加する
5. IdM クライアントでの自動マウントの設定
6. IdM クライアントで、IdM ユーザーが NFS 共有にアクセスできることの確認

6.1. IDM の AUTOFS と AUTOMOUNT

autofs サービスは、アクセス時にディレクトリーをマウントするように **automount** デーモンに指示することにより、必要に応じてディレクトリーのマウントを自動化します。また、しばらく操作を行わな

いと、**autofs** は、**automount** に自動マウントされたディレクトリーのマウントを解除するように指示します。静的マウントとは異なり、オンデマンドマウントはシステムリソースを節約します。

マップの自動マウント

autofs を使用するシステムでは、**automount** 設定は複数のファイルに保存されます。プライマリー **automount** 設定ファイルは **/etc/auto.master** です。これには、システムの **automount** マウントポイントのマスターマッピングと、その関連リソースが含まれます。このマッピングは **自動マウントマップ** として知られています。
/etc/auto.master 設定ファイルには、**マスターマップ** が含まれます。他のマップへの参照を含めることができます。このマップは、直接または間接のいずれかになります。ダイレクトマップではマウントポイントに絶対パス名を使用し、間接マップでは相対パス名を使用します。

IdM の自動マウント設定

automount は通常、ローカルの **/etc/auto.master** と関連ファイルからマップデータを取得しますが、他のソースからマップデータを取得することもできます。一般的なソースの 1 つが LDAP サーバーです。Identity Management (Id M) のコンテキストでは、これは 389 Directory Server です。
autofs を使用するシステムが IdM ドメインのクライアントである場合、**automount** 設定はローカル設定ファイルに保存されません。代わりに、マップ、場所、キーなどの **autofs** 設定は、LDAP エントリーとして IdM ディレクトリーに保存されます。たとえば、**idm.example.com** IdM ドメインの場合、デフォルトの **マスターマップ** は以下のように保存されます。

```
dn:
automountmapname=auto.master,cn=default,cn=automount,dc=idm,dc=example,dc=com
objectClass: automountMap
objectClass: top
automountMapName: auto.master
```

関連情報

- オンデマンドでのファイルシステムのマウント

6.2. RED HAT IDENTITY MANAGEMENT ドメインで KERBEROS を使用する NFS サーバーをセットアップする

Red Hat Identity Management (IdM) を使用すると、NFS サーバーを IdM ドメインに参加させることができます。これにより、ユーザーとグループを一元管理し、認証、整合性保護、トラフィック暗号化に Kerberos を使用できるようになります。

前提条件

- NFS サーバーが Red Hat Identity Management (IdM) ドメインに 登録 されている。
- NFS サーバーが実行および設定されている。

手順

1. IdM 管理者として Kerberos チケットを取得します。

```
# kinit admin
```

2. **nfs/<FQDN>** サービスプリンシパルを作成します。

```
# ipa service-add nfs/nfs_server.idm.example.com
```

3. IdM から **nfs** サービスプリンシパルを取得し、**/etc/krb5.keytab** ファイルに保存します。

```
# ipa-getkeytab -s idm_server.idm.example.com -p nfs/nfs_server.idm.example.com -k
/etc/krb5.keytab
```

4. オプション: **/etc/krb5.keytab** ファイル内のプリンシパルを表示します。

```
# klist -k /etc/krb5.keytab
Keytab name: FILE:/etc/krb5.keytab
KVNO Principal
---- --------------------------------------------------------------------------
   1 nfs/nfs_server.idm.example.com@IDM.EXAMPLE.COM
   1 nfs/nfs_server.idm.example.com@IDM.EXAMPLE.COM
   1 nfs/nfs_server.idm.example.com@IDM.EXAMPLE.COM
   1 nfs/nfs_server.idm.example.com@IDM.EXAMPLE.COM
   7 host/nfs_server.idm.example.com@IDM.EXAMPLE.COM
   7 host/nfs_server.idm.example.com@IDM.EXAMPLE.COM
   7 host/nfs_server.idm.example.com@IDM.EXAMPLE.COM
   7 host/nfs_server.idm.example.com@IDM.EXAMPLE.COM
```

デフォルトでは、ホストを IdM ドメインに参加させると、IdM クライアントがホストプリンシパルを **/etc/krb5.keytab** ファイルに追加します。ホストプリンシパルがない場合は、**ipa-getkeytab -s idm_server.idm.example.com -p host/nfs_server.idm.example.com -k /etc/krb5.keytab** コマンドを使用して追加します。

5. **ipa-client-automount** ユーティリティーを使用して、IdM ID のマッピングを設定します。

```
#  ipa-client-automount
Searching for IPA server...
IPA server: DNS discovery
Location: default
Continue to configure the system with these values? [no]: yes
Configured /etc/idmapd.conf
Restarting sssd, waiting for it to become available.
Started autofs
```

6. **/etc/exports** ファイルを更新し、クライアントオプションに Kerberos セキュリティー方式を追加します。以下に例を示します。

```
/nfs/projects/       192.0.2.0/24(rw,sec=krb5i)
```

クライアントが複数のセキュリティー方式を選択できるようにするには、それらをコロンで区切って指定します。

```
/nfs/projects/       192.0.2.0/24(rw,sec=krb5:krb5i:krb5p)
```

7. エクスポートされたファイルシステムを再ロードします。

```
# exportfs -r
```

6.3. ANSIBLE を使用した IDM での自動マウントの場所、マップ、およびキーの設定

Identity Management (IdM) のシステム管理者は、IdM で自動マウントの場所とマップを設定できます。これにより、指定した場所の IdM ユーザーが、ホストの特定のマウントポイントに移動して、NFS サーバーがエクスポートした共有にアクセスできるようになります。エクスポートされた NFS サーバーディレクトリーとマウントポイントの両方が、マップで指定されます。LDAP 用語では、ロケーションはそのようなマップエントリーのコンテナーです。

この例では、Ansible を使用して、**raleigh** の場所と、IdM クライアント上の **/devel/project** マウントポイントに **nfs-server.idm.example.com:/exports/project** 共有を読み取り/書き込みディレクトリーとしてマウントするマップを設定する方法を説明します。

前提条件

- IdM **admin** のパスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - **~/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. Ansible コントロールノードで、**~/MyPlaybooks/** ディレクトリーに移動します。

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

2. **/usr/share/doc/ansible-freeipa/playbooks/automount/** ディレクトリーにある **automount-location-present.yml** Ansible Playbook ファイルをコピーします。

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/automount/automount-location-present.yml automount-location-map-and-key-present.yml
```

3. **automount-location-map-and-key-present.yml** ファイルを編集用に開きます。
4. **ipaautomountlocation** タスクセクションで次の変数を設定して、ファイルを調整します。
   - **ipaadmin_password** 変数は IdM **admin** のパスワードに設定します。
   - **name** 変数を **raleigh** に設定します。
   - **state** 変数は **present** に設定されていることを確認します。
     以下は、今回の例で使用するように変更した Ansible Playbook ファイルです。

```
---
- name: Automount location present example
  hosts: ipaserver
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Ensure automount location is present
    ipaautomountlocation:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: raleigh
      state: present
```

5. **automount-location-map-and-key-present.yml** ファイルの編集を続けます。

   a. **tasks** セクションで、自動マウントマップの存在を確認するタスクを追加します。

```
[...]
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
[...]
  - name: ensure map named auto.devel in location raleigh is created
    ipaautomountmap:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: auto.devel
      location: raleigh
      state: present
```

   b. 別のタスクを追加して、マウントポイントと NFS サーバー情報をマップに追加します。

```
[...]
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
[...]
  - name: ensure automount key /devel/project is present
    ipaautomountkey:
      ipaadmin_password: "{{ ipaadmin_password }}"
      location: raleigh
      mapname: auto.devel
      key: /devel/project
      info: nfs-server.idm.example.com:/exports/project
      state: present
```

   c. **auto.devel** が **auto.master** に接続されていることを確認する別のタスクを追加します。

```
[...]
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
[...]
- name: Ensure auto.devel is connected in auto.master:
  ipaautomountkey:
    ipaadmin_password: "{{ ipaadmin_password }}"
    location: raleigh
```

```
mapname: auto.map
key: /devel
info: auto.devel
state: present
```

6. ファイルを保存します。

7. Ansible Playbook を実行し、Playbook とインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory automount-location-map-and-key-present.yml
```

6.4. ANSIBLE を使用した NFS 共有を所有するグループへの IDM ユーザーの追加

Identity Management (IdM) システム管理者は、Ansible を使用して、NFS 共有にアクセスできるユーザーのグループを作成し、IdM ユーザーをこのグループに追加できます。

この例では、Ansible Playbook を使用して **idm_user** アカウントが **developers** グループに属していることを確認し、**idm_user** が **/exports/project** NFS 共有にアクセスできるようにする方法について説明します。

前提条件

- **nfs-server.idm.example.com** NFS サーバーへの **root** アクセス権があり、これは **raleigh** automount の場所にある IdM クライアントです。
- IdM **admin** のパスワードを把握している。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - **~/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。
  - **~/MyPlaybooks/** で、Ansible を使用して IdM で automount の場所、マップ、およびキーを設定 するからのタスクがすでに含まれている **automount-location-map-and-key-present.yml** ファイルを作成しました。

手順

1. Ansible コントロールノードで、**~/MyPlaybooks/** ディレクトリーに移動します。

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

2. **automount-location-map-and-key-present.yml** ファイルを編集用に開きます。

3. **tasks** セクションで、IdM **developers** グループが存在し、**idm_user** がこのグループに追加されていることを確認するタスクを追加します。

```
[...]
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
[...]
  - ipagroup:
     ipaadmin_password: "{{ ipaadmin_password }}"
      name: developers
      user:
      - idm_user
      state: present
```

4. ファイルを保存します。

5. Ansible Playbook を実行し、Playbook とインベントリーファイルを指定します。

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory automount-location-map-and-key-present.yml
```

6. NFS サーバーで、**/exports/project** ディレクトリーのグループ所有権を **developers** に変更して、グループ内のすべての IdM ユーザーがディレクトリーにアクセスできるようにします。

```
# chgrp developers /exports/project
```

6.5. IDM クライアントでの自動マウントの設定

Identity Management (IdM) システム管理者は、IdM クライアントに自動マウントサービスを設定して、クライアントが追加された場所に設定した NFS 共有に、ユーザーがクライアントにログインしたときに IdM ユーザーが自動的にアクセスできるようにすることができます。この例では、**raleigh** の場所で利用可能な自動マウントサービスを使用するように IdM クライアントを設定する方法を説明します。

前提条件

- IdM クライアントへの **root** アクセス権限がある。
- IdM 管理者としてログインしている。
- 自動マウントの場所が存在します。サンプルの場所は **raleigh** です。

手順

1. IdM クライアントで、**ipa-client-automount** コマンドを入力して場所を指定します。**-U** オプションを使用して、スクリプトを無人で実行します。

```
# ipa-client-automount --location raleigh -U
```

2. autofs サービスを停止し、SSSD キャッシュをクリアし、autofs サービスを開始して新しい設定をロードします。

```
# systemctl stop autofs ; sss_cache -E ; systemctl start autofs
```

6.6. IDM クライアントで、IDM ユーザーが NFS 共有にアクセスできることの確認

Identity Management (IdM) システム管理者は、特定のグループのメンバーである IdM ユーザーが、特定の IdM クライアントにログインしたときに NFS 共有にアクセスできるかどうかをテストできます。

この例では、以下のシナリオがテストされています。

- **developers** グループに属する **idm_user** という名前の IdM ユーザーは、 **raleigh** automount ロケーションにある IdM クライアントである **idm-client.idm.example.com** に自動マウントされた **/devel/project** ディレクトリー内のファイルの内容を読み書きできます。

前提条件

- IdM ホスト上に Kerberos を使用して NFS サーバーをセットアップした 。
- IdM で自動マウントのロケーション、マップ、マウントポイントを設定 し、そこで IdM ユーザーが NFS共有にアクセスできるように設定している。
- Ansible を使用して、NFS 共有を所有する開発者グループに IdM ユーザーを追加しました 。
- IdM クライアントに自動マウントを設定 している。

手順

1. IdM ユーザーが **read-write** ディレクトリーにアクセスできることを確認します。
   a. IdM ユーザーとして IdM クライアントに接続します。

   ```
   $ ssh idm_user@idm-client.idm.example.com
   Password:
   ```

   b. IdM ユーザーの Ticket Granting Ticket (TGT) を取得します。

   ```
   $ kinit idm_user
   ```

   c. [オプション] Id M ユーザーのグループメンバーシップを表示します。

   ```
   $ ipa user-show idm_user
     User login: idm_user
     [...]
     Member of groups: developers, ipausers
   ```

   d. **/devel/project** ディレクトリーに移動します。

   ```
   $ cd /devel/project
   ```

   e. ディレクトリーの内容をリスト表示します。

   ```
   $ ls
   rw_file
   ```

f. ディレクトリーのファイルに行を追加し、**write** パーミッションをテストします。

```
$ echo "idm_user can write into the file" > rw_file
```

g. [オプション] ファイルの更新された内容を表示します。

```
$ cat rw_file
this is a read-write file
idm_user can write into the file
```

出力は、**idm_user** がファイルに書き込めることを確認します。