Red Hat OpenStack Platform 16.0

Identity サービスとの統合

Active Directory または Red Hat Identity Management を外部認証バックエンドとして使用する方法

Last Updated: 2022-09-13

Red Hat OpenStack Platform 16.0 Identity サービスとの統合

Active Directory または Red Hat Identity Management を外部認証バックエンドとして使用する方法

Enter your first name here. Enter your surname here.
Enter your organisation's name here. Enter your organisational division here.
Enter your email address here.

法律上の通知

概要

Active Directory または Red Hat Identity Management を外部認証バックエンドとして使用する方法

目次

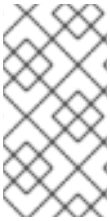
前書き

Identity サービス（コード名 **keystone**）は、Red Hat OpenStack Platform 16.0 の認証と承認の機能を提供します。

本ガイドは、Microsoft Active Directory Domain Service (AD DS)、Red Hat Identity Management (IdM) および LDAP に Identity サービスを統合する方法について説明します。

第1章 ACTIVE DIRECTORY との統合

本章では、Identity サービス (keystone) を Active Directory ドメインサービスに統合する方法について説明します。以下のユースケースでは、Identity サービスが特定の Active Directory ドメインサービス (AD DS) のユーザーを認証しつつ、Identity サービスデータベース内で承認設定および重要なサービスアカウントを保持します。この手順を実行すると、Identity サービスは、AD DS に読み取り専用でアクセスしてユーザーアカウントの認証を行う一方で、認証されたアカウントに割り当てる権限を引き続き管理するようになります。



注記

director を使用している場合には、「*4章director でのドメイン固有の LDAP バックエンドの使用*」を参照してください。以下で参照されている設定ファイルは、Puppet によって管理されているためです。このため、**openstack overcloud deploy** プロセスを実行するたびに、自分で追加したカスタム設定が上書きされる可能性があります。

1.1. 主要な用語

- **認証**: パスワードを使用して、ユーザーが本人であることを検証するプロセス。
- **承認**: 認証されたユーザーに対して、アクセスしようとしているリソースの適切なパーミッションが付与されていることを確認するプロセス。
- **ドメイン**: この用語は、AD DS のドメインとは異なり、ユーザー、グループ、プロジェクトの領域確保のために Identity サービスで設定される追加の名前空間のことを指します。異なる LDAP または AD DS 環境のユーザーを認証する別個のドメインを設定することができます。

1.2. 前提条件

本ガイドのデプロイメント例は、以下を前提としています。

- Active Directory ドメインサービスが設定済みで、稼働していること。
- Red Hat OpenStack Platform が設定済みで、稼働していること。
- DNS 名前解決が完全に機能しており、かつ全ホストが適切に登録されていること。
- AD DS 認証トラフィックが LDAPS で暗号化され、ポート 636 を使用していること。

1.3. 影響評価

以下のステップを実行すると、AD DS ユーザーが OpenStack に対して認証を実行して、リソースにアクセスできるようになります。OpenStack のサービスアカウント (keystone、glance など) および承認管理 (パーミッション、ロール、プロジェクト) は Identity サービスのデータベースに残ります。パーミッションとロールは、Identity サービスの管理ツールを使用して AD DS アカウントに割り当てられます。

1.3.1. 高可用性のオプション

この設定により、単一の Active Directory Domain Controller に依存するようになるため、Identity サービスがその AD Domain Controller に対して認証できない場合には、プロジェクトユーザーが影響を受けることになります。このリスクを管理するオプションは複数あります。たとえば、Identity サービスが 個別の AD Domain Controller ではなく DNS エイリアスやロードバランシングアプライアンスにクエ

リーを実行するように設定することが可能です。また、Domain Controller の 1 つが利用できない場合には、keystone が異なる Domain Controller にクエリーを実行するように設定することもできます。詳細は、「高可用性の設定」 を参照してください。

1.4. 停止の要件

- AD DS バックエンドを追加するには、Identity サービスを再起動する必要があります。
- keystone v3 に切り替えるには、全ノード上の Compute サービスを再起動する必要があります。
- ユーザーは、AD DS でアカウントが作成されるまでは、Dashboard にアクセスできません。ダウンタイムを短縮するには、この変更の前に十分余裕をもって AD DS アカウントのプレステージを行うことを検討してください。

1.5. ファイアウォールの設定

ファイアウォールで AD DS と OpenStack の間のトラフィックをフィルタリングしている場合には、以下のポートを介したアクセスを許可する必要があります。

ソース	送信先	タイプ	ポート
OpenStack コントローラーノード	Active Directory Domain Controller	LDAPS	TCP 636

1.6. ACTIVE DIRECTORY ドメインサービスの設定

本項では、Active Directory の管理者が完了する必要のあるタスクについて説明します。

表1.1 設定手順

タスク	詳細
サービスアカウントの作成	サービスアカウントの名前は、**svc-ldap** など、サービスアカウントの命名規則に従って指定することができます。これは、通常のドメインユーザーアカウントを指定できます。管理者権限は必要ありません。
ユーザーグループの作成	OpenStack へのアクセス権限がユーザーに必要な場合は、このグループに所属する必要があります。ユーザーグループの名前は、**grp-openstack** など、ユーザーグループの命名規則に従って指定することができます。このグループのメンバーが、Dashuboard 内の **プロジェクト** グループのメンバーでもある場合には、そのプロジェクトへのアクセス権が付与されます。
プロジェクトグループの作成	OpenStack の各プロジェクトには、対応する AD グループが必要です。たとえば、**grp-openstack-demo** や **grp-openstack-admin** などがあります。

サービスアカウントの設定	サービスアカウント **svc-ldap** は、**grp-openstack** グループのメンバーである必要があります。
LDAPS 公開鍵のエクスポート	公開鍵 (秘密鍵ではない) は、**DER-encoded x509 .cer** ファイルの形式でエクスポートします。
OpenStack 管理者へのキーの送信	OpenStack の管理者は、このキーを使用して、OpenStack および Active Directory の LDAPS 通信を暗号化します。
AD DS ドメインの NetBIOS 名の取得。	OpenStack 管理者は、Keystone ドメインにこの名前を使用して、複数の環境間で一貫性のあるドメイン名を指定することができます。

たとえば、以下の手順では、Active Directory Domain Controller で実行する PowerShell コマンドが示されています。

1. LDAP ルックアップアカウントを作成します。このアカウントは、Identity サービスが AD DS LDAP サービスにクエリーを実行するのに使用されます。

```
PS C:\> New-ADUser -SamAccountName svc-ldap -Name "svc-ldap" -GivenName LDAP -
Surname Lookups -UserPrincipalName svc-ldap@lab.local  -Enabled $false -
PasswordNeverExpires $true -Path 'OU=labUsers,DC=lab,DC=local'
```

2. このアカウントのパスワードを設定し、有効にします。AD ドメインのパスワードの複雑さの要件を満たすパスワードを指定するように要求されます。

```
PS C:\> Set-ADAccountPassword svc-ldap -PassThru | Enable-ADAccount
```

3. **grp-openstack** という名前の OpenStack ユーザーグループを作成します。

```
PS C:\> NEW-ADGroup -name "grp-openstack" -groupscope Global -path
"OU=labUsers,DC=lab,DC=local"
```

4. プロジェクトグループを作成します。

```
PS C:\> NEW-ADGroup -name "grp-openstack-demo" -groupscope Global -path
"OU=labUsers,DC=lab,DC=local"
PS C:\> NEW-ADGroup -name "grp-openstack-admin" -groupscope Global -path
"OU=labUsers,DC=lab,DC=local"
```

5. **svc-ldap** ユーザーを **grp-openstack** グループに追加します。

```
PS C:\> ADD-ADGroupMember "grp-openstack" -members "svc-ldap"
```

6. AD Domain Controller から **Certificates MMC** を使用して、DER で暗号化された **x509** .cer ファイルとして LDAPS 証明書の (秘密鍵ではなく) 公開鍵 をエクスポートします。このファイルを OpenStack 管理者に送信します。

7. AD DS ドメインの NetBIOS 名の取得。

```
PS C:\> Get-ADDomain | select NetBIOSName
NetBIOSName
-----------
LAB
```

この値を OpenStack 管理者に送信します。

1.7. LDAPS 証明書の設定

注記

LDAP 認証に複数のドメインを使用する場合、**Unable to retrieve authorized projects** または **Peer's Certificate issuer is not recognized** など、さまざまなエラーが発生する可能性があります。これは、keystone が特定ドメインに誤った証明書を使用すると発生する可能性があります。回避策として、すべての LDAPS 公開鍵を単一の **.crt** バンドルにマージし、このファイルを使用するようにすべての keystone ドメインを設定します。

keystone は LDAPS クエリーを使用してユーザーアカウントを検証します。このトラフィックを暗号化するために、keystone は **keystone.conf** で定義されている証明書ファイルを使用します。この手順では、Active Directory から取得した公開鍵を **.crt** 形式に変換して、keystone が参照できる場所にコピーします。

1. OpenStack Identity (keystone) を実行中のノードに、LDAPS 公開鍵をコピーし、**.cer** から **.crt** に変換します。この例では、**addc.lab.local.cer** とうい名前の元の証明書ファイルを使用しています。

```
# openssl x509 -outform der -in addc.lab.local.pem -out addc.lab.local.crt
# cp addc.lab.local.crt /etc/pki/ca-trust/source/anchors
```

注記

オプションとして、**ldapsearch** などの診断のコマンドを実行する必要がある場合には、RHEL の証明書ストアに証明書を追加する必要もあります。

1. **.cer** から **.pem** に変換します。この例では、**addc.lab.local.cer** とうい名前の元の証明書ファイルを使用しています。

```
# openssl x509 -inform der -in addc.lab.local.cer -out addc.lab.local.pem
```

2. OpenStack コントローラーに **.pem** をインストールします。たとえば、Red Hat Enterprise Linux の場合は以下を実行します。

```
# cp addc.lab.local.pem /etc/pki/ca-trust/source/anchors/
# update-ca-trust
```

1.8. IDENTITY サービスの設定

以下のステップでは、Identity サービス (keystone) を AD DS と統合するための準備をします。

注記

director を使用している場合には、以下で参照されている設定ファイルが Puppet によって管理されている点に注意してください。このため、**openstack overcloud deploy** プロセスを実行するたびに、自分で追加したカスタム設定が上書きされる可能性があります。これらの設定を director ベースのデプロイメントに適用するには、「*4章director でのドメイン固有の LDAP バックエンドの使用*」を参照してください。

1.8.1. コントローラーの設定

注記

設定ファイルを更新する場合には、特定の OpenStack サービスはコンテナー内で実行されるようになったことを認識する必要があります。これは、keystone、nova、cinder などのサービスが対象です。そのため、考慮すべき特定の管理プラクティスがいくつかあります。

- 物理ノードのホストオペレーティングシステム上の設定ファイル (例: **/etc/cinder/cinder.conf**) は更新しないでください。コンテナー化されたサービスはこのようなファイルを参照しません。
- コンテナー内で実行されている設定ファイルは更新しないでください。コンテナーを再起動すると変更が失われてしまいます。
  代わりに、コンテナー化されたサービスに変更を加える必要がある場合は、コンテナーの生成に使用される設定ファイルを更新する必要があります。これらのファイルは **/var/lib/config-data/puppet-generated/** 内に保管されています。

  以下に例を示します。
- keystone: **/var/lib/config-data/puppet-generated/keystone/etc/keystone/keystone.conf**
- cinder: **/var/lib/config-data/puppet-generated/cinder/etc/cinder/cinder.conf**
- nova: **/var/lib/config-data/puppet-generated/nova/etc/nova/nova.conf**
  変更内容は、サービスを再起動した後に適用されます。例: **sudo systemctl restart tripleo_keystone**

keystone サービスを実行しているそれぞれの OpenStack ノードで、以下の手順を実行します。

1. SELinux を設定します。

```
# setsebool -P authlogin_nsswitch_use_ldap=on
```

出力には、以下のようなメッセージが含まれている場合がありますが、これは無視できます。

```
Full path required for exclude: net:[4026532245].
```

2. **domains** ディレクトリーを作成します。

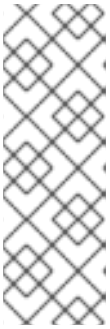
```
# mkdir /var/lib/config-data/puppet-generated/keystone/etc/keystone/domains/
# chown 42425:42425 /var/lib/config-data/puppet-generated/keystone/etc/keystone/domains/
```

3. keystone が複数のバックエンドを使用するように設定します。

注記

dnf install crudini を使用して **crudini** をインストールする必要がある場合があります。

```
# crudini --set /var/lib/config-data/puppet-generated/keystone/etc/keystone/keystone.conf identity domain_specific_drivers_enabled true
# crudini --set /var/lib/config-data/puppet-generated/keystone/etc/keystone/keystone.conf identity domain_config_dir /etc/keystone/domains
# crudini --set /var/lib/config-data/puppet-generated/keystone/etc/keystone/keystone.conf assignment driver sql
```

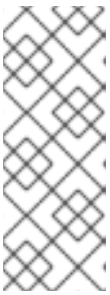


注記

director を使用している場合には、**/var/lib/config-data/puppet-generated/keystone/etc/keystone/keystone.conf** が Puppet によって管理されている点に注意してください。このため、**openstack overcloud deploy** プロセスを実行するたびに、自分で追加したカスタム設定が上書きされる可能性があります。上書きされた場合には、この設定を毎回手動で追加し直す必要がある場合があります。director ベースのデプロイメントについては、「*4章director でのドメイン固有のLDAP バックエンドの使用*」を参照してください。

4. Dashboard で複数のドメインを有効にします。以下の行を **/var/lib/config-data/puppet-generated/horizon/etc/openstack-dashboard/local_settings** に追加します。

```
OPENSTACK_API_VERSIONS = {
    "identity": 3
}
OPENSTACK_KEYSTONE_MULTIDOMAIN_SUPPORT = True
OPENSTACK_KEYSTONE_DEFAULT_DOMAIN = 'Default'
```



注記

director を使用している場合には、**/var/lib/config-data/puppet-generated/horizon/etc/openstack-dashboard/local_settings** が Puppet によって管理されている点に注意してください。このため、**openstack overcloud deploy** プロセスを実行するたびに、自分で追加したカスタム設定が上書きされる可能性があります。上書きされた場合には、この設定を毎回手動で追加し直す必要がある場合があります。

horizon コンテナーを再起動して、設定を適用します。

```
$ sudo systemctl restart tripleo_horizon
```

5. 追加のバックエンドを設定します。
以下の例では、**LAB** は、Identity サービスドメインとして使用する NetBIOS の名前です。

   a. AD DS を統合するための keystone ドメインを作成します。
   上記のステップで取得した NetBIOS 名の値をドメイン名として使用します。このアプローチでは、ログインプロセス中に、一貫したドメイン名をユーザーに提示することができます。たとえば、NetBIOS 名が **LAB** の場合には、以下のコマンドを実行します。

```
$ openstack domain create LAB
```

注記

このコマンドが使用できない場合には、**# source overcloudrc-v3** のコマンドを実行して、コマンドラインセッションでの keystone v3 へのアクセスが有効化されているかどうかを確認します。

b. 設定ファイルを作成します。
AD DS バックエンドを追加するには、**/var/lib/config-data/puppet-generated/keystone/etc/keystone/domains/keystone.LAB.conf** (**LAB** は、前のステップで取得した NetBIOS 名に置き換えます) という名前の新規ファイルに LDAP 設定を入力します。以下の設定例は、実際に使用する AD DS デプロイメントに合わせて編集する必要があります。

```
[ldap]
url                = ldaps://addc.lab.local:636
user               = CN=svc-ldap,OU=labUsers,DC=lab,DC=local
password           = RedactedComplexPassword
suffix             = DC=lab,DC=local
user_tree_dn       = OU=labUsers,DC=lab,DC=local
user_objectclass   = person
user_filter        = (|(memberOf=cn=grp-openstack,OU=labUsers,DC=lab,DC=local)
(memberOf=cn=grp-openstack-admin,OU=labUsers,DC=lab,DC=local)
(memberOf=memberOf=cn=grp-openstack-demo,OU=labUsers,DC=lab,DC=local))
user_id_attribute  = sAMAccountName
user_name_attribute = sAMAccountName
user_mail_attribute = mail
user_pass_attribute =
user_enabled_attribute = userAccountControl
user_enabled_mask  = 2
user_enabled_default = 512
user_attribute_ignore = password,tenant_id,tenants
group_objectclass  = group
group_tree_dn      = OU=labUsers,DC=lab,DC=local
group_filter       = (CN=grp-openstack*)
group_id_attribute = cn
group_name_attribute = name
use_tls            = False
tls_cacertfile     =/etc/pki/ca-trust/source/anchors/anchorsaddc.lab.local.pem

query_scope        = sub
chase_referrals    = false

[identity]
driver = ldap
```

各設定項目について説明します。

設定	説明
url	認証に使用する AD Domain Controller。 LDAPS ポート **636** を使用します。

設定	説明
user	LDAP クエリーに使用する AD アカウントの **識別名**。たとえば、**Get-ADuser svc-ldap | select DistinguishedName** を使用する AD 内の **svc-ldap** アカウントの **識別名** の値を特定することができます。
password	上記で使用した AD アカウントのパスワード (プレーンテキスト形式)。
suffix	AD ドメインの **識別名**。この値は、**Get-ADDomain | select DistinguishedName** を使用して特定することができます。
user_tree_dn	OpenStack アカウントを含む **組織単位** (OU)。
user_objectclass	LDAP ユーザーの種別を定義します。AD には **person** の種別を使用します。
user_filter	Identity サービスに対して提示するユーザーをフィルタリングします。その結果、**grp-openstack** グループのメンバーのみに Identity サービスで定義されているパーミッションを付与することができます。この値には、グループの **完全な識別名** が必要です:**Get-ADGroup grp-openstack | select DistinguishedName**
user_id_attribute	ユーザー ID に使用する AD 値をマッピングします。
user_name_attribute	**names** に使用する AD 値をマッピングします。
user_mail_attribute	ユーザーのメールアドレスに使用する AD 値をマッピングします。
user_pass_attribute	この値は空白のままにします。
user_enabled_attribute	アカウントが有効にされているかどうかを検証する AD の設定。
user_enabled_mask	アカウントが有効化されているかを判断するために確認すべき値を定義します。ブール値が返されない場合に使用します。
user_enabled_default	アカウントが有効化されていることを示す AD 値。

設定	説明
user_attribute_ignore	Identity サービスが無視する必要のあるユーザー属性を定義します。
group_objectclass	**groups** に使用する AD 値をマッピングします。
group_tree_dn	そのユーザーグループを含む **組織単位** (OU)。
group_filter	Identity サービスに提示するグループをフィルタリングします。
group_id_attribute	グループ ID に使用する AD 値をマッピングします。
group_name_attribute	グループ名に使用する AD 値をマッピングします。
use_tls	TLS を使用するかどうかを定義します。STARTTLS ではなく LDAPS で暗号化する場合には、無効にする必要があります。
tls_cacertfile	**.crt** 証明書ファイルへのパスを指定します。
query_scope	**grp-openstack** グループに所属するユーザーを特定する場合に、Identity サービスがネスト化された子 OU 内での検索ができるように設定します。
chase_referrals	**false** に設定します。この設定により **python-ldap** が匿名のアクセスによる全参照を追跡しないようにします。

6. 設定ファイルの所有権を keystone ユーザーに変更します。

```
# chown 42425:42425 /var/lib/config-data/puppet-
generated/keystone/etc/keystone/domains/keystone.LAB.conf
```

7. keystone サービスを再起動して、変更を適用します。

```
# sudo podman exec -it keystone pkill -HUP -f keystone
```

8. **admin** ユーザーに、ドメインのアクセス権を付与します。

注記

この手順を実行しても、OpenStack admin アカウントには実際の AD DS ドメインに対するパーミッションは付与されない点を念頭に置いてください。この場合には、**ドメイン** という用語は、OpenStack が使用する keystone ドメインのことを指しています。

a. **LAB** ドメインの **ID** を取得します。

```
# openstack domain show LAB
+---------+----------------------------------+
| Field   | Value                            |
+---------+----------------------------------+
| enabled | True                             |
| id      | 6800b0496429431ab1c4efbb3fe810d4 |
| name    | LAB                              |
+---------+----------------------------------+
```

b. **admin** ユーザーの **ID** の値を取得します。

```
# openstack user list --domain default | grep admin
| 3d75388d351846c6a880e53b2508172a | admin      |
```

c. **admin** ロールの **ID** の値を取得します。

```
# openstack role list
+----------------------------------+-----------------+
| ID                               | Name            |
+----------------------------------+-----------------+
| 01d92614cd224a589bdf3b171afc5488 | admin           |
| 034e4620ed3d45969dfe8992af001514 | member          |
| 0aa377a807df4149b0a8c69b9560b106 | ResellerAdmin   |
| 9369f2bf754443f199c6d6b96479b1fa | heat_stack_user |
| cfea5760d9c948e7b362abc1d06e557f | reader          |
| d5cb454559e44b47aaa8821df4e11af1 | swiftoperator   |
| ef3d3f510a474d6c860b4098ad658a29 | service         |
+----------------------------------+-----------------+
```

d. 返されたドメインおよび admin の ID を使用して、keystone **LAB** ドメインの **admin** ロールに **admin** ユーザーを 追加するコマンドを構築します。

```
# openstack role add --domain 6800b0496429431ab1c4efbb3fe810d4 --user
3d75388d351846c6a880e53b2508172a 785c70b150ee4c778fe4de088070b4cf
```

e. コマンドで NetBIOS 名を指定して、AD DS ドメイン内のユーザーー覧を確認します。

注記

リブートまたはサービスの再起動後には、LDAP に対してクエリーを実行できるようになるまでに多少時間がかかる場合があります。

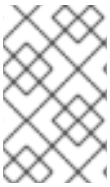
```
# openstack user list --domain LAB
```

f. ローカルの Identity サービスデータベース内のサービスアカウントを確認します。

```
# openstack user list --domain default
```

1.8.2. Active Directory グループのメンバーによるプロジェクトへのアクセスの許可

認証されたユーザーが OpenStack リソースにアクセスするのを許可するための推奨される方法は、特定の Active Directory グループを承認してプロジェクトへのアクセス権を付与することです。これにより、OpenStack の管理者は、プロジェクト内で各ユーザーをロールに割り当てる必要がなくなります。代わりに、Active Directory グループにプロジェクト内のロールが付与されます。その結果、これらの Active Directory グループのメンバーである Active Directory ユーザーは、事前定義済みのプロジェクトにアクセスできるようになります。



注記

個々の Active Directory ユーザーの承認プロセスを手動で管理する方が望ましい場合には、「Active Directory ユーザーによるプロジェクトへのアクセスの許可」を参照してください。

本項は、Active Directory の管理者が以下のステップをすでに完了していることを前提としています。

- Active Directory で **grp-openstack-admin** という名前のグループを作成する。
- Active Directory で **grp-openstack-demo** という名前のグループを作成する。
- 必要に応じて、上記のグループの 1 つに Active Directory ユーザーを追加する。
- Active Directory ユーザーを **grp-openstack** グループに追加する。
- 目的のプロジェクトを作成する。以下の例では、**openstack project create --domain default --description "Demo Project" demo** を使用して作成した **demo** という名前のプロジェクトを使用します。

以下のステップでは、AD グループにロールを割り当てます。これで、グループのメンバーに OpenStack リソースへのアクセス権が付与されます。

1. AD グループの一覧を取得します。

```
# openstack group list --domain LAB
+------------------------------------------------------------------+--------------------+
| ID                                                               | Name               |
+------------------------------------------------------------------+--------------------+
| 185277be62ae17e498a69f98a59b66934fb1d6b7f745f14f5f68953a665b8851 | grp-openstack      |
| a8d17f19f464c4548c18b97e4aa331820f9d3be52654aa8094e698a9182cbb88 | grp-openstack-admin |
| d971bb3bd5e64a454cbd0cc7af4c0773e78d61b5f81321809f8323216938cae8 | grp-openstack-demo  |
+------------------------------------------------------------------+--------------------+
```

2. ロールの一覧を取得します。

```
# openstack role list
+----------------------------------+----------------+
```

```
| ID                               | Name           |
+----------------------------------+----------------+
| 01d92614cd224a589bdf3b171afc5488 | admin          |
| 034e4620ed3d45969dfe8992af001514 | member         |
| 0aa377a807df4149b0a8c69b9560b106 | ResellerAdmin  |
| 9369f2bf754443f199c6d6b96479b1fa | heat_stack_user |
| cfea5760d9c948e7b362abc1d06e557f | reader         |
| d5cb454559e44b47aaa8821df4e11af1 | swiftoperator  |
| ef3d3f510a474d6c860b4098ad658a29 | service        |
+----------------------------------+----------------+
```

3. Active Directory グループに上記のロールを 1 つまたは複数追加して、プロジェクトへのアクセス権を付与します。たとえば、**grp-openstack-demo** グループのユーザーを **demo** プロジェクトの一般ユーザーにするには、そのグループを **member** ロールに追加する必要があります。

```
# openstack role add --project demo --group
d971bb3bd5e64a454cbd0cc7af4c0773e78d61b5f81321809f8323216938cae8  member
```

その結果、**grp-openstack-demo** のメンバーは、AD DS のユーザー名とパスワードを入力してから、ドメインフィールドにも **LAB** と入力すると Dashboard にログインすることができます。

注記

ユーザーに **Error: Unable to retrieve container list.** というエラーメッセージが表示され、コンテナーの管理が可能であることが想定されている場合には、**SwiftOperator** ロールに追加する必要があります。

1.8.3. Active Directory ユーザーによるプロジェクトへのアクセスの許可

grp-openstack AD グループのメンバーである AD DS ユーザーには、Dashboard 内の **プロジェクト** にログインするパーミッションを付与することができます。

1. AD ユーザーの一覧を取得します。

```
# openstack user list --domain LAB
 +------------------------------------------------------------------+----------------+
```

```
| ID                                                                | Name          |
+-------------------------------------------------------------------+---------------+
| 1f24ec1f11aeb90520079c29f70afa060d22e2ce92b2eba7784c841ac418091e | user1         |
| 12c062faddc5f8b065434d9ff6fce03eb9259537c93b411224588686e9a38bf1 | user2         |
| afaf48031eb54c3e44e4cb0353f5b612084033ff70f63c22873d181fdae2e73c | user3         |
| e47fc21dcf0d9716d2663766023e2d8dc15a6d9b01453854a898cabb2396826e | user4
|
+-------------------------------------------------------------------+---------------+
```

2. ロールの一覧を取得します。

```
# openstack role list
+----------------------------------+----------------+
| ID                               | Name           |
+----------------------------------+----------------+
| 01d92614cd224a589bdf3b171afc5488 | admin          |
| 034e4620ed3d45969dfe8992af001514 | member         |
| 0aa377a807df4149b0a8c69b9560b106 | ResellerAdmin  |
| 9369f2bf754443f199c6d6b96479b1fa | heat_stack_user |
| cfea5760d9c948e7b362abc1d06e557f | reader         |
| d5cb454559e44b47aaa8821df4e11af1 | swiftoperator  |
| ef3d3f510a474d6c860b4098ad658a29 | service        |
+----------------------------------+----------------+
```

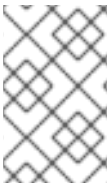
3. 一覧表示されたロールの中から 1 つまたは複数のロールをユーザーに追加して、プロジェクトへのアクセス権を付与します。たとえば、**user1** を **demo** プロジェクトの一般ユーザーにするには、**member** ロールに追加します。

```
# openstack role add --project demo --user
1f24ec1f11aeb90520079c29f70afa060d22e2ce92b2eba7784c841ac418091e member
```

または、**user1** を **demo** プロジェクトの管理ユーザーにするには、**admin** ロールに追加します。

```
# openstack role add --project demo --user
1f24ec1f11aeb90520079c29f70afa060d22e2ce92b2eba7784c841ac418091e admin
```

その結果、**user1** は AD DS のユーザー名とパスワードを入力してから、 **Domain** のフィールドにも **LAB** と入力すると Dashboard にログインすることができます。



注記

ユーザーに **Error: Unable to retrieve container list.** というエラーメッセージが表示され、コンテナーの管理が可能であることが想定されている場合には、**SwiftOperator** ロールに追加する必要があります。

1.9. ドメインタブへのアクセスの許可

admin ユーザーが **Domain** タブを見えるようにするには、そのユーザーを **default** ドメインで **admin** ロールに割り当てる必要があります。

1. **admin** ユーザーの UUID を確認します。

```
$ openstack user list | grep admin
| a6a8adb6356f4a879f079485dad1321b | admin      |
```

2. **default** ドメインの **admin** ロールを **admin** ユーザーに追加します。

```
$ openstack role add --domain default --user a6a8adb6356f4a879f079485dad1321b admin
```

その結果、**admin** ユーザーに **Domain** タブが見えるようになります。

1.10. 新規プロジェクトの作成

上記の統合ステップが完了した後に新規プロジェクトを作成する場合には、**Default** ドメインと、自分で作成した keystone ドメインのどちらに作成するかを決定する必要があります。これは、ワークフローとユーザーアカウントの管理方法を考慮して決定することができます。**Default** ドメインは、サービスアカウントと **admin** プロジェクトの管理に使用する内部ドメインとして考えることができます。また、分離の目的で、AD でバッキングされているユーザーを異なる keystone ドメインで使用することも可能です。

1.11. DASHBOARD へのログインプロセスの変更

Identity サービスに複数のドメインを設定すると、Dashboard のログインページに新しい **ドメイン** フィールドが有効になります。

ユーザーは、ログイン認証情報にマッチするドメインを入力する必要があります。このフィールドには、keystone で提示されるドメインの 1 つを手動で入力する必要があります。利用可能なエントリーを一覧表示するには、**openstack** コマンドを使用します。

以下の例では、AD DS アカウントには **LAB** ドメインを指定する必要があります。**admin** のような組み込みの keystone アカウントには、ドメインに **Default** を指定する必要があります。

```
# openstack domain list
+----------------------------------+---------+---------+-----------------------------------------------------------------
-----+
| ID                               | Name    | Enabled | Description                                                     |
+----------------------------------+---------+---------+-----------------------------------------------------------------
-----+
| 6800b0496429431ab1c4efbb3fe810d4 | LAB     | True    |
|
| default                          | Default | True    | Owns users and projects available on Identity API v2. |
+----------------------------------+---------+---------+-----------------------------------------------------------------
-----+
```

1.12. コマンドラインへの変更

特定のコマンドでは、対象のドメインを指定する必要がある場合があります。たとえば、以下のコマンドに **--domain LAB** を追加すると、LAB ドメイン内のユーザー (**grp-openstack** グループのメンバー) が返されます。

```
# openstack user list --domain LAB
```

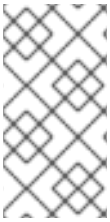
--domain Default を追加すると、組み込みの keystone アカウントが返されます。

```
# openstack user list --domain Default
```

1.13. AD DS 統合のテスト

以下の手順では、Dashboard の機能へのユーザーアクセスをテストして、AD DS の統合を検証します。

1. AD にテストユーザーを作成し、そのユーザーを **grp-openstack** AD DS グループに追加します。
2. テストユーザーを **demo** プロジェクトの **_member_** ロールに追加します。
3. AD テストユーザーの認証情報を使用して Dashboard にログインします。
4. 各タブをクリックし、エラーメッセージなしに正常に表示されているかどうかを確認します。
5. Dashboard を使用してテストインスタンスをビルドします。



注記

上記の手順で問題が発生した場合には、ステップ 3 から 5 までを組み込みの **admin** アカウントで実行してください。正常に実行できた場合には、OpenStack が想定通りに機能していることが実証されるので、問題は AD と Identity の統合設定の間のどこかにあることになります。「トラブルシューティング」 を参照してください。

1.14. 高可用性の設定

keystone v3 が有効化されている場合には、ドメインの設定ファイルに複数の AD Domain Controller をリストして、この設定を高可用性にすることが可能です。

1. 2 番目のサーバーを **url** エントリーに追加します。たとえば、**keystone.LAB.conf** ファイル内の **url** 設定を更新すると、keystone は全クエリートラフィックをリスト内で 1 番目の Domain Controller である **addc.lab.local** に送信します。

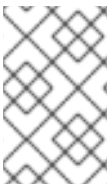
```
url =  ldaps://addc.lab.local,ldaps://addc2.lab.local
```

   addc.lab.local が利用できないためにクエリーが失敗した場合には、Identity サービスはリストに記載されている次のサーバー **addc2.lab.local** にクエリーの送信を試みます。この設定では、ラウンドロビン式にはクエリーが実行されないので、ロードバランスのソリューションとしては考慮できない点に注意してください。

2. **/etc/openldap/ldap.conf** でネットワークのタイムアウトを設定します。

```
NETWORK_TIMEOUT 2
```

また、コントローラーとドメインコントローラーの間でファイアウォールが設定されている場合には、ドメインコントローラーがダイアログを表示せずにコントローラーからのパケットを破棄してしまわないように設定する必要があります。このように設定すると、**python-keystoneclient** が機能停止を適切に検知して、リスト内の次のドメインコントローラーに要求をリダイレクトすることができます。



注記

クエリーがリストの第 2 の LDAP サーバーにリダイレクトされる間に接続の遅延が発生する可能性があります。これは、第 2 のサーバーへの接続を試みるには、第 1 のサーバーがタイムアウトになる必要があるためです。

1.15. 非管理ユーザー用の RC ファイルの作成

非管理ユーザー用の RC ファイルを作成する必要がある場合があります。以下に例を示します。

```
$ cat overcloudrc-v3-user1
# Clear any old environment that may conflict.
for key in $( set | awk '{FS="="}  /^OS_/ {print $1}' ); do unset $key ; done
export OS_USERNAME=user1
export NOVA_VERSION=1.1
export OS_PROJECT_NAME=demo
export OS_PASSWORD=RedactedComplexPassword
export OS_NO_CACHE=True
export COMPUTE_API_VERSION=1.1
export no_proxy=,10.0.0.5,192.168.2.11
export OS_CLOUDNAME=overcloud
export OS_AUTH_URL=https://10.0.0.5:5000/v3
export OS_AUTH_TYPE=password
export PYTHONWARNINGS="ignore:Certificate has no, ignore:A true
SSLContext object is not available"
export OS_IDENTITY_API_VERSION=3
export OS_PROJECT_DOMAIN_NAME=Default
export OS_USER_DOMAIN_NAME=LAB
```

1.16. トラブルシューティング

1.16.1. LDAP 接続のテスト

注記

このコマンドを実行すると、ホストオペレーティングシステム内で必要な証明書が特定されるはずです。詳しい情報は、**「LDAPS 証明書の設定」**の項を参照してください。

Active Directory Domain Controller に対してテストクエリーをリモートで実行するには、**ldapsearch** を使用します。クエリーが成功した場合には、ネットワーク接続が機能しており、AD DS サービスが稼働中であることを確認できます。以下の例では、テストクエリーはサーバー **addc.lab.local** のポート**636** に対して実行されます。

```
# ldapsearch -Z -x -H ldaps://addc.lab.local:636 -D "svc-ldap@lab.local" -W -b
"OU=labUsers,DC=lab,DC=local" -s sub "(cn=*)" cn
```

注記

ldapsearch は、**openldap-clients** パッケージに含まれています。このパッケージは、**# dnf install openldap-clients** のコマンドを実行するとインストールすることができます。

1.16.2. 証明書トラストの設定テスト

ldapsearch のテストの際に **Peer's Certificate issuer is not recognized.** というエラーを受け取った場合には、**TLS_CACERTDIR** パスが正しく設定されていることを確認してください。以下に例を示します。

- **/etc/openldap/ldap.conf**

```
TLS_CACERTDIR /etc/openldap/certs
```

注記

一時的な回避策として、証明書の検証を無効にすることを検討してください。

この設定は、永続的には使用しないでください。

- **/etc/openldap/ldap.conf**

```
TLS_REQCERT allow
```

この値を設定した後に **ldapsearch** クエリーが機能した場合には、証明書トラストが正しく設定されているかどうかを確認する必要があります。

1.16.3. ポートアクセスのテスト

nc を使用して、LDAPS ポート **636** がリモートでアクセス可能であることを確認します。この例では、サーバー **addc.lab.local** に対してプローブを実行します。ctrl-c を押してプロンプトを終了します。

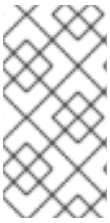
```
# nc -v addc.lab.local 636
Ncat: Version 6.40 ( http://nmap.org/ncat )
Ncat: Connected to 192.168.200.10:636.
^C
```

接続を確立できなかった場合には、ファイアウォールの設定に問題がある可能性があります。

第2章 IDENTITY MANAGEMENT の統合

本章では、Identity サービス (keystone) を Red Hat Identity Management と統合する方法について説明します。

以下のユースケースでは、Identity サービスが特定の Red Hat Identity Management (IdM) のユーザーを認証しつつ、Identity サービスデータベース内で承認設定および重要なサービスアカウントを保持します。
この手順を実行すると、Identity サービスは、IdM に読み取り専用でアクセスしてユーザーアカウントの認証を行う一方で、認証されたアカウントに割り当てる権限を引き続き管理するようになります。



注記

director を使用している場合には、「*4章director でのドメイン固有の LDAP バックエンドの使用*」を参照してください。以下で参照されている設定ファイルは、Puppet によって管理されているためです。このため、**openstack overcloud deploy** プロセスを実行するたびに、自分で追加したカスタム設定が上書きされる可能性があります。

注記

novajoin を使用した追加の統合オプションについては、「*3章novajoin を使用した IdM との統合*」を参照してください。

2.1. 主要な用語

- **認証**: パスワードを使用して、ユーザーが本人であることを検証するプロセス。
- **承認**: 認証されたユーザーに対して、アクセスしようとしているシステムの適切なパーミッションが付与されていることを確認するプロセス。
- **ドメイン**: Identity サービス内で設定する追加のバックエンド。たとえば、Identity サービスは、外部の IdM 環境内のユーザーを認証するように設定することができます。このように設定されたユーザーの集合は、**ドメイン** として考えることができます。

2.2. 前提条件

本ガイドのデプロイメント例は、以下を前提としています。

- Red Hat Identity Management が設定済みで、稼働していること。
- Red Hat OpenStack Platform が設定済みで、稼働していること。
- DNS 名前解決が完全に機能しており、かつ全ホストが適切に登録されていること。

2.3. 影響評価

以下のステップを実行すると、IdM ユーザーが OpenStack に対して認証を実行して、リソースにアクセスできるようになります。OpenStack のサービスアカウント (keystone、glance など) および承認管理 (パーミッションとロール) は Identity サービスのデータベースに残ります。パーミションとロールは、Identity サービスの管理ツールを使用して IdM アカウントに割り当てられます。

2.3.1. 高可用性のオプション

この設定により、単一の IdM サーバーの可用性に依存するようになるため、Identity サービスがその IdM サーバー に対して認証できない場合には、プロジェクトユーザーが影響を受けることになります。このリスクを管理するオプションは複数あります。たとえば、keystone が個別の IdM サーバーではなく DNS エイリアスやロードバランシングアプライアンスにクエリーを実行するように設定することが可能です。また、IdM サーバー の 1 つが利用できない場合には、keystone が異なる IdM サーバーにクエリーを実行するように設定することもできます。詳細は、「高可用性の設定」 を参照してください。

2.4. 停止の要件

- IdM バックエンドを追加するには、Identity サービスを再起動する必要があります。
- **keystone v3** に切り替えるには、全ノード上の Compute サービスを再起動する必要があります。
- ユーザーは、IdM でアカウントが作成されるまでは、Dashboard にアクセスできません。ダウンタイムを短縮するには、この変更の前に十分余裕をもって IdM アカウントのプレステージを行うことを検討してください。

2.5. ファイアウォールの設定

ファイアウォールが IdM と OpenStack の間のトラフィックをフィルタリングしている場合には、以下のポートを介したアクセスを許可する必要があります。

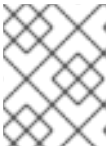
ソース	送信先	タイプ	ポート
OpenStack コントローラーノード	Red Hat Identity Management	LDAPS	TCP 636

2.6. IDM サーバーの設定

IdM サーバーで、以下のコマンドを実行します。

1. LDAP ルックアップアカウントを作成します。このアカウントは、Identity サービスが IdM の LDAP サービスにクエリーを実行するのに使用します。

```
# kinit admin
# ipa user-add
First name: OpenStack
Last name: LDAP
User  [radministrator]: svc-ldap
```



注記

作成が完了したら、このアカウントのパスワード期限の設定を確認してください。

2. **grp-openstack** という名前の OpenStack ユーザーグループを作成します。OpenStack Identity でパーミッションを割り当てることができるのは、このグループのメンバーのみです。

```
# ipa group-add --desc="OpenStack Users" grp-openstack
```

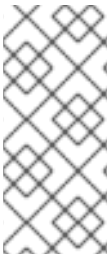
3. **svc-ldap** アカウントのパスワードを設定して、**grp-openstack** グループに追加します。

```
# ipa passwd svc-ldap
# ipa group-add-member --users=svc-ldap grp-openstack
```

4. **svc-ldap** ユーザーとしてログインし、指示に従ってパスワード変更を実施します。

```
# kinit svc-ldap
```

2.7. LDAPS 証明書の設定



注記

LDAP 認証に複数のドメインを使用する場合、**Unable to retrieve authorized projects** または **Peer's Certificate issuer is not recognized** など、さまざまなエラーが発生する可能性があります。これは、keystone が特定ドメインに誤った証明書を使用すると発生する可能性があります。回避策として、すべての LDAPS 公開鍵を単一の **.crt** バンドルにマージし、このファイルを使用するようにすべての keystone ドメインを設定します。

1. IdM の環境で、LDAPS 証明書を見つけます。このファイルの場所は、**/etc/openldap/ldap.conf** で確認することができます。

```
TLS_CACERT /etc/ipa/ca.crt
```

2. keystone サービスを実行している OpenStack ノードにファイルをコピーします。たとえば、以下のコマンドは、**scp** を使用して、ca.crt を **node.lab.local** という名前のノードにコピーします。

```
# scp /etc/ipa/ca.crt root@node.lab.local:/root/
```

3. OpenStack ノードで、.crt を .pem に変換します。

```
# openssl x509 -in ca.crt -out ca.pem -outform PEM
```

4. .crt を証明書のディレクトリーにコピーします。keystone サービスは、この場所を使用して証明書にアクセスします。

```
# cp ca.crt/etc/pki/ca-trust/source/anchors
```

注記

オプションとして、**ldapsearch** などの診断のコマンドを実行する必要がある場合には、RHEL の証明書ストアに証明書を追加する必要もあります。以下に例を示します。

```
# cp ca.pem /etc/pki/ca-trust/source/anchors/
# update-ca-trust
```

2.8. IDENTITY サービスの設定

以下のステップでは、IdM との統合に備えて、Identity サービスの準備を行います。

注記

director を使用している場合には、以下で参照されている設定ファイルが Puppet によって管理されている点に注意してください。このため、**openstack overcloud deploy** プロセスを実行するたびに、自分で追加したカスタム設定が上書きされる可能性があります。これらの設定を director ベースのデプロイメントに適用するには、「*4章director でのドメイン固有の LDAP バックエンドの使用*」を参照してください。

2.8.1. コントローラーの設定

注記

設定ファイルを更新する場合には、特定の OpenStack サービスはコンテナー内で実行されるようになったことを認識する必要があります。これは、keystone、nova、cinder などのサービスが対象です。そのため、考慮すべき特定の管理プラクティスがいくつかあります。

- 物理ノードのホストオペレーティングシステム上の設定ファイル (例: **/etc/cinder/cinder.conf**) は更新しないでください。コンテナー化されたサービスはこのようなファイルを参照しません。
- コンテナー内で実行されている設定ファイルは更新しないでください。コンテナーを再起動すると変更が失われてしまいます。
  代わりに、コンテナー化されたサービスに変更を加える必要がある場合は、コンテナーの生成に使用される設定ファイルを更新する必要があります。これらのファイルは **/var/lib/config-data/puppet-generated/** 内に保管されています。

  以下に例を示します。
- keystone: **/var/lib/config-data/puppet-generated/keystone/etc/keystone/keystone.conf**
- cinder: **/var/lib/config-data/puppet-generated/cinder/etc/cinder/cinder.conf**
- nova: **/var/lib/config-data/puppet-generated/nova/etc/nova/nova.conf**
  変更内容は、コンテナーを再起動した後に適用されます。例: **sudo systemctl restart tripleo_keystone**

keystone サービスを実行しているコントローラーで、以下の手順を実行します。

1. SELinux を設定します。

```
# setsebool -P authlogin_nsswitch_use_ldap=on
```

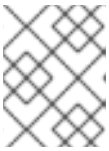
出力には、以下のようなメッセージが含まれている場合がありますが、これは無視できます。

```
Full path required for exclude: net:[4026532245].
```

2. **domains** ディレクトリーを作成します。

```
# mkdir /var/lib/config-data/puppet-generated/keystone/etc/keystone/domains/
# chown 42425:42425 /var/lib/config-data/puppet-generated/keystone/etc/keystone/domains/
```

3. Identity サービスが複数のバックエンドを使用するように設定します。



注記

dnf install crudini を使用して **crudini** をインストールする必要がある場合があります。

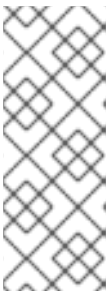
```
# crudini --set /var/lib/config-data/puppet-generated/keystone/etc/keystone/keystone.conf identity domain_specific_drivers_enabled true
# crudini --set /var/lib/config-data/puppet-generated/keystone/etc/keystone/keystone.conf identity domain_config_dir /etc/keystone/domains
# crudini --set /var/lib/config-data/puppet-generated/keystone/etc/keystone/keystone.conf assignment driver sql
```

注記

director を使用している場合には、**/var/lib/config-data/puppet-generated/keystone/etc/keystone/keystone.conf** が Puppet によって管理されている点に注意してください。このため、**openstack overcloud deploy** プロセスを実行するたびに、自分で追加したカスタム設定が上書きされる可能性があります。上書きされた場合には、この設定を毎回手動で追加し直す必要がある場合があります。director ベースのデプロイメントについては、「*4章director でのドメイン固有のLDAP バックエンドの使用*」を参照してください。

4. Dashboard で複数のドメインを有効にします。以下の行を **/var/lib/config-data/puppet-generated/horizon/etc/openstack-dashboard/local_settings** に追加します。

```
OPENSTACK_API_VERSIONS = {
    "identity": 3
}
OPENSTACK_KEYSTONE_MULTIDOMAIN_SUPPORT = True
OPENSTACK_KEYSTONE_DEFAULT_DOMAIN = 'Default'
```



注記

director を使用している場合には、**/var/lib/config-data/puppet-generated/horizon/etc/openstack-dashboard/local_settings** が Puppet によって管理されている点に注意してください。このため、**openstack overcloud deploy** プロセスを実行するたびに、自分で追加したカスタム設定が上書きされる可能性があります。上書きされた場合には、この設定を毎回手動で追加し直す必要がある場合があります。

horizon コンテナーを再起動して、設定を適用します。

```
$ sudo systemctl restart tripleo_horizon
```

5. 追加のバックエンドを設定します。

   a. IdM 統合のための keystone ドメインを作成します。新規 keystone ドメインに使用する名前を決定してから、ドメインを作成する必要があります。たとえば、以下のコマンドは **LAB** という名前の keystone ドメインを作成します。

```
$ openstack domain create LAB
```

注記

このコマンドが使用できない場合には、コマンドラインセッションでの **keystone v3** へのアクセスが有効化されているかどうかを確認してください。

b. 設定ファイルを作成します。
IdM バックエンドを追加するには、**/var/lib/config-data/puppet-generated/keystone/etc/keystone/domains/keystone.LAB.conf** (**LAB** は、前のステップで作成したドメイン名に置き換えます) という名前の新規ファイルに LDAP 設定を入力します。以下の設定例は、実際に使用する IdM デプロイメントに合わせて編集する必要があります。

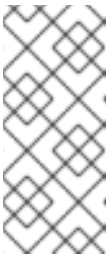
```
[ldap]
url =  ldaps://idm.lab.local
user = uid=svc-ldap,cn=users,cn=accounts,dc=lab,dc=local
user_filter = (memberOf=cn=grp-openstack,cn=groups,cn=accounts,dc=lab,dc=local)
password = RedactedComplexPassword
user_tree_dn = cn=users,cn=accounts,dc=lab,dc=local
user_objectclass = inetUser
user_id_attribute = uid
user_name_attribute = uid
user_mail_attribute = mail
user_pass_attribute =
group_tree_dn              = cn=groups,cn=accounts,dc=lab,dc=local
group_objectclass            = groupOfNames
group_id_attribute          = cn
group_name_attribute       =  cn
group_member_attribute  = member
group_desc_attribute       = description
use_tls                = False
query_scope                = sub
chase_referrals              = false
tls_cacertfile =/etc/pki/ca-trust/source/anchors/anchorsca.crt

[identity]
driver = ldap
```

各設定項目について説明します。

設定	説明
url	認証に使用する IdM サーバー。LDAPS ポート **636** を使用します。
user	LDAP クエリーに使用する IdM 内のアカウント。
password	上記で使用した IdM アカウントのパスワード (プレーンテキスト形式)。

設定	説明
user_filter	Identity サービスに対して提示するユーザーをフィルタリングします。その結果、**grp-openstack** グループのメンバーのみに Identity サービスで定義されているパーミッションを付与することができます。
user_tree_dn	IdM 内の OpenStack アカウントへのパス。
user_objectclass	LDAP ユーザーの種別を定義します。IdM には **inetUser** の種別を使用します。
user_id_attribute	ユーザー ID に使用する IdM 値をマッピングします。
user_name_attribute	**names** に使用する IdM 値をマッピングします。
user_mail_attribute	ユーザーのメールアドレスに使用する IdM 値をマッピングします。
user_pass_attribute	この値は空白のままにします。



注記

IdM グループとの統合で返されるのは直接のメンバーだけで、ネスト化されたグループは返されません。したがって、**LDAP_MATCHING_RULE_IN_CHAIN** または **memberof:1.2.840.113556.1.4.1941:** に依存するクエリーは、現在 IdM では機能しません。

6. 設定ファイルの所有者を keystone ユーザーに変更します。

```
# chown 42425:42425 /var/lib/config-data/puppet-generated/keystone/etc/keystone/domains/keystone.LAB.conf
```

7. admin ユーザーにドメインへのアクセス権を付与します。

注記

この手順を実行しても、OpenStack admin アカウントには IdM のパーミッションは付与されません。ここで使われるドメインという用語は、OpenStack が使用する keystone ドメインのことを指しています。

a. **LAB** ドメインの **ID** を取得します。

```
$ openstack domain show LAB
+---------+----------------------------------+
| Field   | Value                            |
+---------+----------------------------------+
| enabled | True                             |
| id      | 6800b0496429431ab1c4efbb3fe810d4 |
| name    | LAB                              |
+---------+----------------------------------+
```

b. **admin** ユーザーの **ID** の値を取得します。

```
$ openstack user list --domain default | grep admin

| 3d75388d351846c6a880e53b2508172a | admin      |
```

c. **admin** ロールの **ID** の値を取得します。

```
# openstack role list
+----------------------------------+---------------+
| ID                               | Name          |
+----------------------------------+---------------+
| 544d48aaffde48f1b3c31a52c35f01f9 | SwiftOperator |
| 6d005d783bf0436e882c55c62457d33d | ResellerAdmin |
| 785c70b150ee4c778fe4de088070b4cf | admin         |
| 9fe2ff9ee4384b1894a90878d3e92bab | _member_      |
+----------------------------------+---------------+
```

d. 返されたドメインおよび admin の ID を使用して、keystone LAB ドメインの **admin** ロールに **admin** ユーザーを追加するコマンドを構築します。

```
$ openstack role add --domain 6800b0496429431ab1c4efbb3fe810d4 --user
3d75388d351846c6a880e53b2508172a 785c70b150ee4c778fe4de088070b4cf
```

8. keystone サービスを再起動して、変更を適用します。

```
$ sudo podman restart keystone
```

9. コマンドで keystone ドメイン名を指定して、IdM ドメイン内のユーザー一覧を確認します。

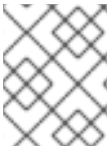
```
$ openstack user list --domain LAB
```

10. ローカルの keystone データベース内のサービスアカウントを確認します。

```
$ openstack user list --domain default
```

2.8.2. IdM グループのメンバーによるプロジェクトへのアクセスの許可

認証されたユーザーが OpenStack リソースにアクセスするのを許可するための推奨される方法は、特定の IdM グループを承認してプロジェクトへのアクセス権を付与することです。これにより、OpenStack の管理者は、プロジェクト内で各ユーザーをロールに割り当てる必要がなくなります。代わりに、IdM グループにプロジェクト内のロールが付与されます。その結果、これらの IdM グループのメンバーである IdM ユーザーは、事前定義済みのプロジェクトにアクセスできるようになります。



注記

個々の IdM ユーザーの承認プロセスを手動で管理する方が望ましい場合には、「IdM ユーザーによるプロジェクトへのアクセスの許可」を参照してください。

本項は、IdM の管理者が以下のステップをすでに完了していることを前提としています。

- IdM で **grp-openstack-admin** という名前のグループを作成する。
- IdM で **grp-openstack-demo** という名前のグループを作成する。
- 必要に応じて、上記のグループの 1 つに IdM ユーザーを追加する。
- IdM ユーザーを **grp-openstack** グループに追加する。
- 目的のプロジェクトを作成する。以下の例では、**openstack project create --domain default --description "Demo Project" demo** を使用して作成した **demo** という名前のプロジェクトを使用します。

以下のステップでは、IdM グループにロールを割り当てます。これで、グループのメンバーに OpenStack リソースへのアクセス権が付与されます。

1. IdM グループの一覧を取得します。

```
$ openstack group list --domain LAB
+------------------------------------------------------------------+--------------------+
| ID                                                               | Name               |
+------------------------------------------------------------------+--------------------+
| 185277be62ae17e498a69f98a59b66934fb1d6b7f745f14f5f68953a665b8851 | grp-
openstack       |
| a8d17f19f464c4548c18b97e4aa331820f9d3be52654aa8094e698a9182cbb88 | grp-
openstack-admin |
| d971bb3bd5e64a454cbd0cc7af4c0773e78d61b5f81321809f8323216938cae8 | grp-
openstack-demo  |
+------------------------------------------------------------------+--------------------+
```

2. ロールの一覧を取得します。

```
$ openstack role list
+----------------------------------+---------------+
| ID                               | Name          |
+----------------------------------+---------------+
| 0969957bce5e4f678ca6cef00e1abf8a | ResellerAdmin |
| 1fcb3c9b50aa46ee8196aaaecc2b76b7 | admin         |
| 9fe2ff9ee4384b1894a90878d3e92bab | _member_      |
| d3570730eb4b4780a7fed97eba197e1b | SwiftOperator |
+----------------------------------+---------------+
```

3. IdM グループに上記のロールを 1 つまたは複数追加して、プロジェクトへのアクセス権を付与します。たとえば、**grp-openstack-demo** グループ内のユーザーを **demo** プロジェクトの一般ユーザーにするには、そのグループを **_member_** ロールに追加する必要があります。

```
$ openstack role add --project demo --group
d971bb3bd5e64a454cbd0cc7af4c0773e78d61b5f81321809f8323216938cae8  _member_
```

その結果、**grp-openstack-demo** のメンバーは、IdM のユーザー名とパスワードを入力してから、ドメインフィールドにも **LAB** と入力すると Dashboard にログインすることができます。

注記

ユーザーに **Error: Unable to retrieve container list.** というエラーメッセージが表示され、コンテナーの管理が可能であることが想定されている場合には、**SwiftOperator** ロールに追加する必要があります。

2.8.3. IdM ユーザーによるプロジェクトへのアクセスの許可

grp-openstack IdM グループのメンバーである IdM ユーザーには、Dashboard 内の **プロジェクト** にログインするパーミッションを付与することができます。

1. IdM ユーザーの一覧を取得します。

```
# openstack user list --domain LAB
 +------------------------------------------------------------------+----------------+
| ID                                                               | Name           |
+------------------------------------------------------------------+----------------+
| 1f24ec1f11aeb90520079c29f70afa060d22e2ce92b2eba7784c841ac418091e | user1          |
| 12c062faddc5f8b065434d9ff6fce03eb9259537c93b411224588686e9a38bf1 | user2          |
| afaf48031eb54c3e44e4cb0353f5b612084033ff70f63c22873d181fdae2e73c | user3          |
| e47fc21dcf0d9716d2663766023e2d8dc15a6d9b01453854a898cabb2396826e | user4
|
+------------------------------------------------------------------+----------------+
```

2. ロールの一覧を取得します。

```
# openstack role list
+----------------------------------+---------------+
| ID                               | Name          |
+----------------------------------+---------------+
| 544d48aaffde48f1b3c31a52c35f01f9 | SwiftOperator |
| 6d005d783bf0436e882c55c62457d33d | ResellerAdmin |
```

```
| 785c70b150ee4c778fe4de088070b4cf | admin        |
| 9fe2ff9ee4384b1894a90878d3e92bab | _member_     |
+----------------------------------+--------------+
```

3. 一覧表示されたロールの中から 1 つまたは複数のロールをユーザーに追加して、プロジェクトへのアクセス権を付与します。たとえば、**user1** を **demo** プロジェクトの一般ユーザーにするには、**member** ロールに追加します。

```
# openstack role add --project demo --user
1f24ec1f11aeb90520079c29f70afa060d22e2ce92b2eba7784c841ac418091e _member_
```

または、**user1** を **demo** プロジェクトの管理ユーザーにするには、**admin** ロールに追加します。

```
# openstack role add --project demo --user
1f24ec1f11aeb90520079c29f70afa060d22e2ce92b2eba7784c841ac418091e admin
```

その結果、**user1** は IdM のユーザー名とパスワードを入力してから、 **Domain** のフィールドにも **LAB** と入力すると Dashboard にログインすることができます。

注記

ユーザーに **Error: Unable to retrieve container list.** というエラーメッセージが表示され、コンテナーの管理が可能であることが想定されている場合には、**SwiftOperator** ロールに追加する必要があります。

2.9. ドメインタブへのアクセスの許可

admin ユーザーが **Domain** タブを見えるようにするには、そのユーザーを **default** ドメインで **admin** ロールに割り当てる必要があります。

1. **admin** ユーザーの UUID を確認します。

```
$ openstack user list | grep admin
| a6a8adb6356f4a879f079485dad1321b | admin      |
```

2. **default** ドメインの **admin** ロールを **admin** ユーザーに追加します。

```
$ openstack role add --domain default --user a6a8adb6356f4a879f079485dad1321b admin
```

その結果、**admin** ユーザーに **Domain** タブが見えるようになります。

2.10. 新規プロジェクトの作成

上記の統合ステップが完了した後に新規プロジェクトを作成する場合には、**Default** ドメインと、自分で作成した keystone ドメインのどちらに作成するかを決定する必要があります。これは、ワークフローとユーザーアカウントの管理方法を考慮して決定することができます。**Default** ドメインは、サービスアカウントと **admin** プロジェクトに使用する内部ドメインとして考えることができるので、AD でバッキングされているユーザーを異なる keystone ドメインに内に配置することは理にかなっています。これは、IdM ユーザーが管理されているのと全く同じ keystone ドメインである必要はありません。また、分離の目的で複数の keystone ドメインを使用することも可能です。

2.10.1. Dashboard へのログインプロセスの変更

Identity サービスに複数のドメインを設定すると、Dashboard のログインページに新しい **ドメイン** フィールドが有効になります。
ユーザーは、ログイン認証情報にマッチするドメインを入力する必要があります。このフィールドには、keystone で提示されるドメインの 1 つを手動で入力する必要があります。利用可能なエントリーを一覧表示するには、**openstack** コマンドを使用します。

以下の例では、IdM アカウントは **LAB** ドメインを指定する必要があります。**admin** のような組み込みの keystone アカウントには、ドメインに **Default** を指定する必要があります。

```
$ openstack domain list
+----------------------------------+---------+---------+----------------------------------------------------------------
-----+
| ID                               | Name    | Enabled | Description                                                    |
+----------------------------------+---------+---------+----------------------------------------------------------------
-----+
| 6800b0496429431ab1c4efbb3fe810d4 | LAB     | True    |
|
| default                          | Default | True    | Owns users and projects available on Identity API v2. |
+----------------------------------+---------+---------+----------------------------------------------------------------
-----+
```

2.10.2. コマンドラインへの変更

特定のコマンドでは、対象のドメインを指定する必要がある場合があります。たとえば、以下のコマンドに **--domain LAB** を追加すると、LAB ドメイン内のユーザー (**grp-openstack** グループのメンバー) が返されます。

```
$ openstack user list --domain LAB
```

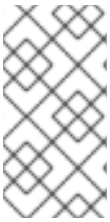
--domain Default を追加すると、組み込みの keystone アカウントが返されます。

```
$ openstack user list --domain Default
```

2.10.3. IdM 統合のテスト

以下の手順では、Dashboard の機能へのユーザーアクセスをテストして、IdM の統合を検証します。

1. IdM にテストユーザーを作成し、そのユーザーを **grp-openstack** IdM グループに追加します。
2. テストユーザーを **demo** プロジェクトの **_member_** ロールに追加します。
3. IdM テストユーザーの認証情報を使用して Dashboard にログインします。
4. 各タブをクリックし、エラーメッセージなしに正常に表示されているかどうかを確認します。
5. Dashboard を使用してテストインスタンスをビルドします。



注記

上記の手順で問題が発生した場合には、ステップ 3 から 5 までを組み込みの **admin** アカウントで実行してください。正常に実行できた場合には、OpenStack が想定通りに機能していることが実証されるので、問題は IdM と Identity の統合設定の間のどこかにあることになります。「トラブルシューティング」 を参照してください。

2.11. 高可用性の設定

keystone v3 が有効化されている場合には、ドメインの設定ファイルに複数の IdM サーバーをリストして、この設定を高可用性にすることが可能です。

1. 2 番目のサーバーを **url** エントリーに追加します。たとえば、**keystone.LAB.conf** ファイル内の **url** 設定を更新すると、Identity サービスは全クエリートラフィックをリスト内で 1 番目の IdM サーバーである **idm.lab.local** に送信します。

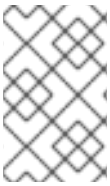
```
url =  ldaps://idm.lab.local,ldaps://idm2.lab.local
```

idm.lab.local が利用できないためにクエリーが失敗した場合には、Identity サービスはリストに記載されている次のサーバー **idm2.lab.local** にクエリーの送信を試みます。この設定では、ラウンドロビン式にはクエリーが実行されないので、ロードバランスのソリューションとしては考慮できない点に注意してください。

1. **/etc/openldap/ldap.conf** でネットワークのタイムアウトを設定します。

```
NETWORK_TIMEOUT 2
```

また、コントローラーと IdM サーバーの間でファイアウォールが設定されている場合には、IdM サーバーがダイアログを表示せずにコントローラーからのパケットを破棄してしまわないように設定する必要があります。このように設定すると、**python-keystoneclient** が機能停止を適切に検知して、リスト内の次の IdM サーバーに要求をリダイレクトすることができます。



注記

クエリーがリストの第 2 の IdM サーバーにリダイレクトされる間に接続の遅延が発生する可能性があります。これは、第 2 のサーバーへの接続を試みるには、第 1 のサーバーがタイムアウトになる必要があるためです。

2.12. 非管理ユーザー用の RC ファイルの作成

非管理ユーザー用の RC ファイルを作成する必要がある場合があります。以下に例を示します。

```
$ cat overcloudrc-v3-user1
# Clear any old environment that may conflict.
for key in $( set | awk '{FS="="}  /^OS_/ {print $1}' ); do unset $key ; done
export OS_USERNAME=user1
export NOVA_VERSION=1.1
export OS_PROJECT_NAME=demo
export OS_PASSWORD=RedactedComplexPassword
export OS_NO_CACHE=True
export COMPUTE_API_VERSION=1.1
export no_proxy=,10.0.0.5,192.168.2.11
export OS_CLOUDNAME=overcloud
export OS_AUTH_URL=https://10.0.0.5:5000/v3
export OS_AUTH_TYPE=password
export PYTHONWARNINGS="ignore:Certificate has no, ignore:A true
SSLContext object is not available"
export OS_IDENTITY_API_VERSION=3
export OS_PROJECT_DOMAIN_NAME=Default
export OS_USER_DOMAIN_NAME=LAB
```

2.13. トラブルシューティング

2.13.1. LDAP 接続のテスト

IdM サーバーに対してテストクエリーをリモートで実行するには、**ldapsearch** を使用します。クエリーが成功した場合には、ネットワーク接続が機能しており、IdM サービスが稼働中であることを確認できます。以下の例では、テストクエリーはサーバー**idm.lab.local** のポート 636 に対して実行されます。

```
# ldapsearch -D "cn=directory manager" -H ldaps://idm.lab.local:636 -b "dc=lab,dc=local" -s sub "
(objectclass=*)" -w RedactedComplexPassword
```

注記

ldapsearch は、**openldap-clients** パッケージに含まれています。このパッケージは、**# dnf install openldap-clients** のコマンドを実行するとインストールすることができます。

2.13.2. ポートアクセスのテスト

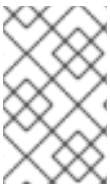
nc を使用して、LDAPS ポート (636) がリモートでアクセス可能であることを確認します。この例では、サーバー **idm.lab.local** に対してプローブを実行します。ctrl-c を押してプロンプトを終了します。

```
# nc -v idm.lab.local 636
Ncat: Version 6.40 ( http://nmap.org/ncat )
Ncat: Connected to 192.168.200.10:636.
^C
```

接続を確立できなかった場合には、ファイアウォールの設定に問題がある可能性があります。

第3章 NOVAJOIN を使用した IDM との統合

novajoin により、使用するノードをデプロイメントプロセスの一部として Red Hat Identity Manager (IdM) に登録できます。その結果、OpenStack デプロイメントで、ID、kerberos 認証情報、アクセス制御などの IdM の機能を統合することができます。



注記

現在、novajoin を使用した IdM の登録は、アンダークラウドとオーバークラウドのノードのみで利用可能です。オーバークラウドインスタンス向けの novajoin の統合は、今後のリリースでサポートされる見込みです。

3.1. アンダークラウドでの NOVAJOIN のインストールと設定

3.1.1. CA へのアンダークラウドの追加

オーバークラウドをデプロイする前には、アンダークラウドを認証局 (CA) に追加する必要があります。

1. アンダークラウドノードで、**python3-novajoin** パッケージをインストールします。

```
$ sudo dnf install python3-novajoin
```

2. アンダークラウドノードで **novajoin-ipa-setup** スクリプトを実行します。値はデプロイメントに応じて調整します。

```
$ sudo /usr/libexec/novajoin-ipa-setup \
    --principal admin \
    --password <IdM admin password> \
    --server <IdM server hostname> \
    --realm <overcloud cloud domain (in upper case)> \
    --domain <overcloud cloud domain> \
    --hostname <undercloud hostname> \
    --precreate
```

以下の項では、ここで設定されたワンタイムパスワード (OTP) を使用してアンダークラウドを登録します。

3.1.2. IdM へのアンダークラウドの追加

以下の手順では、アンダークラウドを IdM に登録して novajoin を設定します。**undercloud.conf** で以下の設定を行います (**[DEFAULT]** セクション内)。

1. novajoin サービスは、デフォルトで無効にされます。有効にするには、以下のように設定します。

```
[DEFAULT]
enable_novajoin = true
```

2. アンダークラウドノードをIdM に登録するためのワンタイムパスワード (OTP) を設定する必要があります。

```
ipa_otp = <otp>
```

3. neutron の DHCP サーバーにより提供されるように、オーバークラウドのドメイン名を設定します。

```
overcloud_domain_name = <domain>
```

4. アンダークラウドに適切なホスト名を設定します。

```
undercloud_hostname = <undercloud FQDN>
```

5. アンダークラウドのネームサーバーとして IdM を設定します。

```
undercloud_nameservers = <IdM IP>
```

6. より大きな環境の場合には、novajoin の接続タイムアウト値を確認する必要があります。**undercloud.conf** で、**undercloud-timeout.yaml** という名前の新規ファイルへの参照を追加します。

```
hieradata_override = /home/stack/undercloud-timeout.yaml
```

undercloud-timeout.yaml に以下のオプションを追加します。タイムアウト値は秒単位で指定することができます (例: **5**)。

```
nova::api::vendordata_dynamic_connect_timeout: <timeout value>
nova::api::vendordata_dynamic_read_timeout: <timeout value>
```

7. **undercloud.conf** ファイルを保存します。

8. アンダークラウドのデプロイコマンドを実行して、既存のアンダークラウドに変更を適用します。

```
$ openstack undercloud install
```

3.2. オーバークラウドでの NOVAJOIN のインストールと設定

以下の項では、オーバークラウドノードを IdM で登録する方法について説明します。

3.2.1. オーバークラウド DNS の設定

IdM 環境を自動検出して、登録をより簡単にするには、IdM を DNS サーバーとして使用することを検討してください。

1. アンダークラウドに接続します。

```
$ source ~/stackrc
```

2. DNS ネームサーバーとして IdM を使用するようにコントロールプレーンサブネットを設定します。

```
$ openstack subnet set ctlplane-subnet --dns-nameserver  <idm_server_address>
```

3. IdM サーバーを使用するように環境ファイルの **DnsServers** パラメーターを設定します。

```
parameter_defaults:
  DnsServers: ["<idm_server_address>"]
```

このパラメーターは、通常カスタムの **network-environment.yaml** ファイルで定義されます。

3.2.2. novajoin を使用するためのオーバークラウドの設定

1. IdM 統合を有効化するには、**/usr/share/openstack-tripleo-heat-templates/environments/predictable-placement/custom-domain.yaml** 環境ファイルのコピーを作成します。

```
$ cp /usr/share/openstack-tripleo-heat-templates/environments/predictable-
placement/custom-domain.yaml \
  /home/stack/templates/custom-domain.yaml
```

2. **/home/stack/templates/custom-domain.yaml** 環境ファイルを編集して、デプロイメントに適した **CloudDomain** と **CloudName*** の値を設定します。以下に例を示します。

```
parameter_defaults:
  CloudDomain: lab.local
  CloudName: overcloud.lab.local
  CloudNameInternal: overcloud.internalapi.lab.local
  CloudNameStorage: overcloud.storage.lab.local
  CloudNameStorageManagement: overcloud.storagemgmt.lab.local
  CloudNameCtlplane: overcloud.ctlplane.lab.local
```

3. オーバークラウドのデプロイプロセスで以下の環境ファイルを追加します。

   - **/usr/share/openstack-tripleo-heat-templates/environments/ssl/enable-internal-tls.yaml**
   - **/usr/share/openstack-tripleo-heat-templates/environments/ssl/tls-everywhere-endpoints-dns.yaml**
   - **/home/stack/templates/custom-domain.yaml**
     以下に例を示します。

```
openstack overcloud deploy \
  --templates \
    -e /usr/share/openstack-tripleo-heat-templates/environments/ssl/enable-internal-
tls.yaml \
    -e /usr/share/openstack-tripleo-heat-templates/environments/ssl/tls-everywhere-
endpoints-dns.yaml \
    -e /home/stack/templates/custom-domain.yaml \
```

     その結果、デプロイされるオーバークラウドノードは自動的に IdM で登録されるようになります。

4. これで設定されるのは、内部エンドポイント向けの TLS のみです。外部エンドポイントには、**/usr/share/openstack-tripleo-heat-templates/environments/ssl/enable-tls.yaml** 環境ファイル (カスタムの証明書とキーを追加するように編集する必要あり) で TLS を追加する通常の方法を使用することができます。そのため、**openstack deploy** コマンドは以下のようになります。

```
openstack overcloud deploy \
```

```
  --templates \
  -e /usr/share/openstack-tripleo-heat-templates/environments/ssl/enable-internal-tls.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/ssl/tls-everywhere-endpoints-dns.yaml \
  -e /home/stack/templates/custom-domain.yaml \
  -e /home/stack/templates/enable-tls.yaml
```

5. また、IdM を使用して公開証明書を発行することもできます。その場合には、**/usr/share/openstack-tripleo-heat-templates/environments/services/haproxy-public-tls-certmonger.yaml** 環境ファイルを使用する必要があります。以下に例を示します。

```
openstack overcloud deploy \
  --templates \
   -e /usr/share/openstack-tripleo-heat-templates/environments/ssl/enable-internal-tls.yaml \
   -e /usr/share/openstack-tripleo-heat-templates/environments/ssl/tls-everywhere-endpoints-dns.yaml \
   -e /home/stack/templates/custom-domain.yaml \
   -e /usr/share/openstack-tripleo-heat-templates/environments/services/haproxy-public-tls-certmonger.yaml
```

注記

novajoin を使用して、既存のデプロイメントに TLS everywhere (TLS-e) を実装することができなくなりました。上記の手順は、Red Hat OpenStack Platform の初期インストール時にのみ使用してください。

3.3. IDM におけるノードの検証

1. IdM でオーバークラウドノードを特定し、ホストのエントリーに **Keytab:True** が含まれていることを確認します。

```
$ ipa host-show overcloud-node-01
  Host name: overcloud-node-01.lab.local
  Principal name: host/overcloud-node-01.lab.local@LAB.LOCAL
  Principal alias: host/overcloud-node-01.lab.local@LAB.LOCAL
  SSH public key fingerprint: <snip>
  Password: False
  Keytab: True
  Managed by: overcloud-node-01.lab.local
```

2. そのノードに SSH 接続し、**sssd** が IdM ユーザーをクエリーできることを確認します。たとえば、**susan** という名前の IdM ユーザーをクエリーするには、以下のコマンドを実行します。

```
$ getent passwd susan
uid=1108400007(susan) gid=1108400007(bob) groups=1108400007(susan)
```

3.4. NOVAJOIN 用の DNS エントリーの設定

haproxy-public-tls-certmonger.yaml テンプレートを使用してエンドポイントの公開証明書を発行する場合には、Novajoin が使用する VIP エンドポイントの DNS エントリーを手動で作成する必要があります。

1. オーバークラウドのネットワークを識別します。通常は、**/home/stack/virt/network/network-environment.yaml** で特定することができます。

```
parameter_defaults:
  ControlPlaneDefaultRoute: 192.168.24.1
  ExternalAllocationPools:
  - end: 10.0.0.149
    start: 10.0.0.101
  InternalApiAllocationPools:
  - end: 172.17.1.149
    start: 172.17.1.10
  StorageAllocationPools:
  - end: 172.17.3.149
    start: 172.17.3.10
  StorageMgmtAllocationPools:
  - end: 172.17.4.149
    start: 172.17.4.10
```

2. 各オーバークラウドネットワークの仮想 IP アドレス (VIP) の一覧を作成します 例: /home/stack/virt/public_vip.yaml

```
parameter_defaults:
  ControlFixedIPs: [{'ip_address':'192.168.24.101'}]
  PublicVirtualFixedIPs: [{'ip_address':'10.0.0.101'}]
  InternalApiVirtualFixedIPs: [{'ip_address':'172.17.1.101'}]
  StorageVirtualFixedIPs: [{'ip_address':'172.17.3.101'}]
  StorageMgmtVirtualFixedIPs: [{'ip_address':'172.17.4.101'}]
  RedisVirtualFixedIPs: [{'ip_address':'172.17.1.102'}]
```

3. それぞれの VIP について、DNS エントリーを IdM に追加します。新たなゾーンも作成する必要があります。IdM 用の DNS レコードおよびゾーン作成の例を、以下に示します。

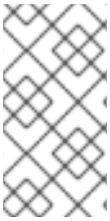
```
ipa dnsrecord-add lab.local overcloud --a-rec 10.0.0.101
ipa dnszone-add ctlplane.lab.local
ipa dnsrecord-add ctlplane.lab.local overcloud --a-rec 192.168.24.101
ipa dnszone-add internalapi.lab.local
ipa dnsrecord-add internalapi.lab.local overcloud --a-rec 172.17.1.101
ipa dnszone-add storage.lab.local
ipa dnsrecord-add storage.lab.local overcloud --a-rec 172.17.3.101
ipa dnszone-add storagemgmt.lab.local
ipa dnsrecord-add storagemgmt.lab.local overcloud --a-rec 172.17.4.101
```

第4章 DIRECTOR でのドメイン固有の LDAP バックエンドの使用

Red Hat OpenStack Platform director では、keystone が 1 つまたは複数の LDAP バックエンドを使用するように設定することができます。この方法では、keystone ドメインごとに別々の LDAP バックエンドが作成されます。

4.1. 設定オプションの指定

Red Hat OpenStack Platform director を使用したデプロイメントの場合には、Heat テンプレートで **KeystoneLDAPDomainEnable** フラグを **true** に設定する必要があります。その結果、keystone で **domain_specific_drivers_enabled** オプションが設定されます (**identity** 設定グループ内)。



注記

ドメイン設定ファイルのデフォルトのディレクトリーは **/etc/keystone/domains/** に設定されています。**keystone::domain_config_directory** hiera キーを使用し、環境ファイル内に **ExtraConfig** パラメーターを追加して、必要なパスを設定することによって上書きすることができます。

LDAP バックエンドの設定の仕様を追加する必要もあります。これは、**tripleo-heat-templates** の **KeystoneLDAPBackendConfigs** パラメーターを使用して設定します。これで、必要な LDAP オプションを指定することができるようになります。

4.2. DIRECTOR デプロイメントの設定

1. **keystone_domain_specific_ldap_backend.yaml** 環境ファイルのコピーを作成します。

```
$ cp /usr/share/openstack-tripleo-heat-
templates/environments/services/keystone_domain_specific_ldap_backend.yaml
/home/stack/templates/
```

2. **/home/stack/templates/keystone_domain_specific_ldap_backend.yaml** 環境ファイルを編集して、デプロイメントに適した値を設定します。たとえば、以下のエントリーは、**testdomain** という名前の keystone ドメイン向けの LDAP 設定を作成します。

```
parameter_defaults:
  KeystoneLDAPDomainEnable: true
  KeystoneLDAPBackendConfigs:
    testdomain:
      url: ldaps://192.0.2.250
      user: cn=openstack,ou=Users,dc=director,dc=example,dc=com
      password: RedactedComplexPassword
      suffix: dc=director,dc=example,dc=com
      user_tree_dn: ou=Users,dc=director,dc=example,dc=com
      user_filter: "(memberOf=cn=OSuser,ou=Groups,dc=director,dc=example,dc=com)"
      user_objectclass: person
      user_id_attribute: cn
```

3. また、複数のドメインを指定するように環境ファイルを設定することも可能です。以下に例を示します。

```
KeystoneLDAPBackendConfigs:
  domain1:
```

```
    url: ldaps://domain1.example.com
    user: cn=openstack,ou=Users,dc=director,dc=example,dc=com
    password: RedactedComplexPassword
    ...
  domain2:
    url: ldaps://domain2.example.com
    user: cn=openstack,ou=Users,dc=director,dc=example,dc=com
    password: RedactedComplexPassword
    ...
```

これにより、**domain1** と **domain2** という名前の 2 つのドメインが指定され、各ドメインには、異なる LDAP ドメインが独自の設定で適用されます。