Red Hat OpenStack Platform 11
ユーザーおよびアイデンティティー管理ガイド

ユーザーおよび認証メカニズムの管理

OpenStack Team

Red Hat OpenStack Platform 11 ユーザーおよびアイデンティティー管理ガイド

ユーザーおよび認証メカニズムの管理

OpenStack Team
rhos-docs@redhat.com

法律上の通知

概要

ユーザーおよびアイデンティティー管理ガイドでは、Red Hat OpenStack Platform 環境のユーザーロール、クォータ、プロジェクト、プロジェクトセキュリティー、Identity サービスの管理手順を説明します。

目次

前書き

クラウドの管理者は、プロジェクト、ユーザー、ロールを管理することができます。プロジェクトとは、ユーザーの割り当てが可能な、クラウド内の組織単位のことで、テナントまたはアカウントとしても知られています。ユーザーは、1 つまたは複数のプロジェクトのメンバーにすることができ、ロールは、ユーザーが実行できるアクションを定義します。

各 OpenStack デプロイメントには、最低でもプロジェクト、ユーザー、ロールが 1 つずつあり、それらが連携している必要があります。クラウド管理者は、プロジェクトとユーザーの追加、更新、削除、1 つまたは複数のプロジェクトへのユーザーの割り当てを行うことができます。プロジェクトとユーザーは、個別に管理することが可能です。

Keystone Identity サービスでユーザー認証を設定して、サービスおよびエンドポイントへのアクセスを制御することも可能です。Keystone では、トークンベースの認証が提供され、LDAP と Active Directory と統合することができるため、ユーザーとアイデンティティーを外部で管理し、Keystone とユーザーデータを同期できます。

第1章 ユーザー管理

1.1. ユーザー管理

クラウド管理者は、Dashboard でユーザーの追加、変更、削除ができます。ユーザーは、1 つまたは複数のプロジェクトに所属することができます。また、プロジェクトとユーザーは個別に管理することができます。

1.1.1. ユーザーの作成

Dashboard でユーザーを作成するには、以下の手順に従ってください。主要なプロジェクトとロールをユーザーに割り当てることができます。Dashboard で作成したユーザーは、デフォルトでは Keystone のユーザーとなっています。Active Directory ユーザーを統合するには、Red Hat OpenStack Platform の Identity サービスに含まれる LDAP プロバイダーを設定してください。

1. Dashboard に管理ユーザーとしてログインして **アイデンティティー > ユーザー** を選択します。
2. **ユーザーの作成** をクリックします。
3. ユーザーのユーザー名、メールアドレス、仮のパスワードを入力します。
4. **主プロジェクト** のリストからプロジェクトを選択します。
5. **ロール** のリストからロールを選択します (デフォルトは**_member_** です)。
6. **ユーザーの作成** をクリックします。

1.1.2. ユーザーの編集

以下の手順に従って、主プロジェクトなど、ユーザーの詳細を更新します。

1. Dashboard に管理ユーザーとしてログインして **アイデンティティー > ユーザー** を選択します。
2. ユーザーの **アクション** コラムで、**編集** をクリックします。
3. **ユーザーの更新** ウィンドウで、**ユーザー名**、**メール**、**主プロジェクト** を更新できます。
4. **ユーザーの更新** をクリックします。

1.1.3. ユーザーの有効化/無効化

以下の手順に従って、ユーザーを有効化または無効化します。1 度に 1 ユーザーしか無効化または有効化できません。無効化されたユーザーは Dashboard にはログインできず、OpenStack サービスへのアクセスもできません。また、無効化されたユーザーの主プロジェクトもアクティブに設定できません。アクションを元に戻せないユーザーの削除とは異なり、無効化されたユーザーをもう 1 度有効化することができます。また、ユーザーが無効な場合には、Dashboard のユーザーとプロジェクトのアクションを実行するには、ユーザーを有効化する必要があります。

1. Dashboard に管理ユーザーとしてログインして **アイデンティティー > ユーザー** を選択します。

2. **アクション** コラムでドロップダウンリストをクリックし、**ユーザーの有効化** または **ユーザーの無効化** を選択すると、**有効** コラムの値が **`True`** または **`False`** に更新されます。

1.1.4. ユーザーの削除

管理者ユーザーが Dashboard を使用してユーザーを削除するには、以下の手順を実行します。このアクションは、ユーザーの無効化とは異なり、元に戻すことはできません。ユーザーを無効にした場合には、所属するプロジェクトのメンバー一覧から削除されます。ユーザーとプロジェクトのペアに関連付けられたロールはすべて失われます。

1. Dashboard に管理ユーザーとしてログインして **アイデンティティー > ユーザー** を選択します。
2. 削除するユーザーを選択します。
3. **ユーザーの削除** をクリックします。**ユーザーの削除の確認** ウィンドウが表示されます。
4. **ユーザーの削除** をクリックしてアクションを確認します。

第2章 ロールの管理

2.1. ロールの管理

OpenStack はロールベースアクセス制御 (RBAC) のメカニズムを使用して、リソースへのアクセスを管理します。ロールは、ユーザーが実行可能なアクションを定義します。デフォルトでは、テナントにアタッチされるメンバーロールと、管理者以外のユーザーが環境を管理できるようにする管理者ロールという事前定義済みのロールが 2 つあります。パーミッションには抽象レベルがあり、管理者が必要なロールを作成して適切にサービスを設定することができる点に注意してください。

2.1.1. ロールの表示

利用可能な事前定義済みのロールを一覧表示するには、以下のコマンドを使用します。

```
$ openstack role list
+----------------------------------+---------------+
| ID                               | Name          |
+----------------------------------+---------------+
| 4fd37c2c993a4acab8e1b5896afb8687 | SwiftOperator |
| 9fe2ff9ee4384b1894a90878d3e92bab | _member_      |
| a0f19c1381c54770ae068456c4411d82 | ResellerAdmin |
| ae49e2b796ea4820ac51637be27650d8 | admin         |
+----------------------------------+---------------+
```

指定したロールの詳細を取得するには、以下のコマンドを実行します。

```
$ openstack role show admin
```

例

```
$ openstack role show admin
+-----------+----------------------------------+
| Field     | Value                            |
+-----------+----------------------------------+
| domain_id | None                             |
| id        | ae49e2b796ea4820ac51637be27650d8 |
| name      | admin                            |
+-----------+----------------------------------+
```

2.1.2. ロールの作成および割り当て

クラウド管理者は、以下のコマンド一式を使用して Keystone クライアントでロールを作成、管理できます。各 OpenStack のデプロイメントには、最低でもプロジェクト、ユーザー、ロールが 1 つずつ必要で、それぞれ連携されている必要があります。ただし、ユーザーは複数のプロジェクトのメンバーになることができます。複数のプロジェクトにユーザーを割り当てるには、ロールを作成して、ユーザーとプロジェクトのペアにそのロールを割り当てます。Dashboard でユーザーを作成して、主プロジェクトとデフォルトのロールを割り当てることができる点に注意してください。

注記

ユーザー、ロール、プロジェクトの指定には名前または ID を使用することができます。

1. **new-role** という名前のロールを作成します。

```
$ openstack role create [ROLE_NAME]
```

例

```
$ openstack role create new-role
+-----------+----------------------------------+
| Field     | Value                            |
+-----------+----------------------------------+
| domain_id | None                             |
| id        | 880c116b6a55464b99ca8d8d8fe26743 |
| name      | new-role                         |
+-----------+----------------------------------+
```

2. ユーザーをプロジェクトに割り当てるには、ロールをユーザーとプロジェクトのペアに割り当てる必要があります。これには、ユーザー、ロール、プロジェクト名/ID を取得してください。

   a. ユーザーを一覧表示します。

   ```
   $ openstack user list
   ```

   b. ロールを一覧表示します。

   ```
   $ openstack role list
   ```

   c. プロジェクトを一覧表示します。

   ```
   $ openstack project list
   ```

3. ユーザーとプロジェクトのペアにロールを割り当てます。

```
openstack role add --project [PROJECT_NAME] --user [USER_ID]
[ROLE_ID]
```

例

以下の例では、**demo** プロジェクトで **admin** ロールを **admin** ユーザーに割り当てます。

```
$ openstack role add --project demo --user
895e43465b9643b9aa29df0073572bb2
ae49e2b796ea4820ac51637be27650d8
```

4. **admin** ユーザーのロール割り当てを確認します。

```
$ openstack role assignment list --user [USER_ID]  --project
[PROJECT_ID]
```

例

```
$ openstack role assignment list --user
895e43465b9643b9aa29df0073572bb2 --project demo
+----------------------------------+------------------------
---------+-------+----------------------------------+--------
+-----------+
| Role                               | User
| Group | Project                           | Domain | Inherited
|
+----------------------------------+------------------------
---------+-------+----------------------------------+--------
+-----------+
| ae49e2b796ea4820ac51637be27650d8 |
895e43465b9643b9aa29df0073572bb2 |       |
7efbdc8b4ab448b8b5aeb9fa5898ce23 |        | False     |
+----------------------------------+------------------------
---------+-------+----------------------------------+--------
+-----------+
```

2.2. 暗黙的なロールとドメイン固有のロール

2.2.1. 暗黙的なロール

OpenStack では、ユーザーが特定のロールに割り当てられているのを確認して、アクセス制御を適用します。最近までは、そのようなロールはユーザーまたはユーザーがメンバーとなっているグループに明示的に割り当てられていましたが、Identity サービス (keystone) に暗黙的なロールの割り当ての概念が追加されたので、ユーザーが 1 つのロールに明示的に割り当てられている場合には、別のロールにも暗黙的に割り当てられている可能性があります。

2.2.2. 推論規則

暗黙的な割り当てはロールの推論規則で管理されます。推論規則は、**上位が下位を暗黙に示す** 形式で書かれます。たとえば、1 つのルールで **admin** ロールは **_member_** ロールを暗黙的に割り当てるように記述することができます。その結果、プロジェクトの **admin** に割り当てられたユーザーは、**_member_** ロールにも暗黙的に割り当てられます。

暗黙的なロール では、ユーザーのロール割り当ては累積的に処理され、ユーザーは下位のロールを継承することができます。暗黙的なロールは、その結果を指定するために作成される推論規則によって異なります。

2.2.2.1. Keystone の設定

keystone が暗黙的なルールを順守するには、**/etc/keystone/keystone.conf** で **infer_roles** の設定を有効にする必要があります。

```
[token]
infer_roles = true
```

暗黙的なロールは、一連の定義済み推論規則によって統制されます。これらのルールにより、1 つのロールを割り当てることによって、他のロールのメンバーシップをどのように暗黙的に割り当てられることができるかを決定します。「暗黙的なロールの実例」に記載の例を参照してください。

2.2.3. 特定のロールが暗黙的となるのを防ぐ方法

特定のロールがユーザーに暗黙的に割り当てられるのを防ぐことが可能です。たとえば、**/etc/keystone/keystone.conf** でロールの **ListOpt** を追加することができます。

```
[assignment]
prohibited_implied_role = admin
```

この設定は、特定のロールがユーザーに暗黙的に割り当てられるのを常に防ぎます。そのロールに対するアクセス権は、暗黙的ではなく明示的に付与しなければならないようになります。

2.2.3.1. 暗黙的なロールの実例

本項では、ロールを暗黙的に割り当てるための推論規則の作成方法について説明します。このルールは、1 つのロールが別のロールのメンバーシップを暗黙的に継承できるようにする方法を制御します。 以下の手順で使用するルールの例は、**admin** ロールのメンバーに **_member_** のアクセスも付与されるようにします。

2.2.3.1.1. ユーザーへのロールの割り当て:

1. **_member_** ロールを暗黙的に継承するユーザーの ID を取得します。以下に例を示します。

```
$ openstack user show User1
+---------------------+----------------------------------+
| Field               | Value                            |
+---------------------+----------------------------------+
| domain_id           | default                          |
| enabled             | True                             |
| id                  | ce803dd127c9489199c89ce3b68d39b4 |
| name                | User1                            |
| options             | {}                               |
| password_expires_at | None                             |
+---------------------+----------------------------------+
```

2. **demo** プロジェクトの ID を取得します。

```
$ openstack project show demo
+-------------+----------------------------------+
| Field       | Value                            |
+-------------+----------------------------------+
| description | default tenant                   |
| domain_id   | default                          |
| enabled     | True                             |
| id          | 2717ebc905e449b5975449c370edac69 |
| is_domain   | False                            |
| name        | demo                             |
| parent_id   | default                          |
+-------------+----------------------------------+
```

3. **admin** ロールの ID を取得します。

```
$ openstack role show admin
+-----------+----------------------------------+
| Field     | Value                            |
+-----------+----------------------------------+
| domain_id | None                             |
| id        | 9b821b2920544be7a4d8f71fa99fcd35 |
| name      | admin                            |
+-----------+----------------------------------+
```

4. **User1** ユーザーに、**demo** プロジェクトに対する **admin** 権限を付与します。

```
$ openstack role add --user User1 --project demo admin
```

5. **admin** ロールの割り当てを確認します。

```
$ openstack role assignment list --user User1 --project demo --
effective
+----------------------------------+------------------------
---------+-------+----------------------------------+--------
+-----------+
| Role                             | User
| Group | Project                          | Domain | Inherited
|
+----------------------------------+------------------------
---------+-------+----------------------------------+--------
+-----------+
| 9b821b2920544be7a4d8f71fa99fcd35 |
ce803dd127c9489199c89ce3b68d39b4 |       |
2717ebc905e449b5975449c370edac69 |        | False     |
+----------------------------------+------------------------
---------+-------+----------------------------------+--------
+-----------+
```

2.2.3.1.2. 推論規則の作成

admin ロールを **User1** に付与するステップが完了したので、次に以下のステップに従って推論規則を作成します。

1. 最初に User 1 の現在のロールメンバーシップを確認します。

```
$ openstack role assignment list --user User1 --project demo --
effective
+----------------------------------+------------------------
---------+-------+----------------------------------+--------
+-----------+
| Role                             | User
| Group | Project                          | Domain | Inherited
|
+----------------------------------+------------------------
---------+-------+----------------------------------+--------
+-----------+
| 9b821b2920544be7a4d8f71fa99fcd35 |
```

```
ce803dd127c9489199c89ce3b68d39b4 |       |
2717ebc905e449b5975449c370edac69 |        | False     |
+----------------------------------+------------------------
---------+-------+----------------------------------+--------
+-----------+
```

2. ロール ID の一覧を取得します。

```
$ openstack role list
+----------------------------------+---------------+
| ID                               | Name          |
+----------------------------------+---------------+
| 9b821b2920544be7a4d8f71fa99fcd35 | admin         |
| 9fe2ff9ee4384b1894a90878d3e92bab | _member_      |
| ea199fe4293745719c2afd3402ed7b95 | ResellerAdmin |
| fe8eba5dfd1e4f4a854ad20a150d995e | SwiftOperator |
+----------------------------------+---------------+
```

3. 推論規則を作成します。現在このロールは **curl** で作成します。この例では、前のステップで返されたロールの ID を使用します。また、**keystone.conf** の **admin_token** を使用してコマンドを実行します。

```
source overcloudrc
export OS_TOKEN=`grep ^admin_token /etc/keystone/keystone.conf |
awk -F'=' '{print $2}'`
curl -X PUT  -H "X-Auth-Token: $OS_TOKEN" -H "Content-type:
application/json"
$OS_AUTH_URL/roles/9b821b2920544be7a4d8f71fa99fcd35/implies/9fe2f
f9ee4384b1894a90878d3e92bab
```

4. CLI を使用して結果を確認します。この例では、**9fe2ff9ee4384b1894a90878d3e92bab** の ID で示されている **_member_** ロールへの暗黙的なアクセスが User1 に付与されています。

```
source overcloudrc
# openstack role assignment list --user User1 --project demo --
effective
+----------------------------------+------------------------
---------+-------+----------------------------------+--------
+-----------+
| Role                             | User
| Group | Project                          | Domain | Inherited
|
+----------------------------------+------------------------
---------+-------+----------------------------------+--------
+-----------+
| 9b821b2920544be7a4d8f71fa99fcd35 |
ce803dd127c9489199c89ce3b68d39b4 |       |
2717ebc905e449b5975449c370edac69 |        | False     |
| 9fe2ff9ee4384b1894a90878d3e92bab |
ce803dd127c9489199c89ce3b68d39b4 |       |
2717ebc905e449b5975449c370edac69 |        | False     |
+----------------------------------+------------------------
---------+-------+----------------------------------+--------
+-----------+
```

5. **curl** を使用して推論規則を確認します。

```
source overcloudrc
export OS_TOKEN=`grep ^admin_token /etc/keystone/keystone.conf |
awk -F'=' '{print $2}'`
curl -s -H "X-Auth-Token: $OS_TOKEN" $OS_AUTH_URL/role_inferences
| python -mjson.tool
{
    "role_inferences": [
        {
            "implies": [
                {
                    "id": "9fe2ff9ee4384b1894a90878d3e92bab",
                    "links": {
                        "self":
"https://osp.lab.local:5000/v3/roles/9fe2ff9ee4384b1894a90878d3e9
2bab"
                    },
                    "name": "_member_"
                }
            ],
            "prior_role": {
                "id": "9b821b2920544be7a4d8f71fa99fcd35",
                "links": {
                    "self":
"https://osp.lab.local:5000/v3/roles/9b821b2920544be7a4d8f71fa99f
cd35"
                },
                "name": "admin"
            }
        }
    ]
}
```

2.2.4. ドメイン固有のロール

ドメイン固有のロールを使用すると、ロールのルールを定義する際に、より粒度の高い制御が可能となるので、ロールが既存の **prior** ロールのエイリアスとして機能することができます。ドメイン固有のロールを暗黙的に継承するグローバルロールを設定することはできない点に注意してください。このため、1 つのプロジェクト内のユーザーの有効なロールの割り当てを一覧表示しても、ドメイン固有のロールはありません。

ドメイン固有のロールを作成できるのは、keystone ドメインを管理するユーザーです。このユーザーは、OpenStack のデプロイメントの管理者である必要はありません。このため、ドメイン固有のロールの定義は特定のドメインに限定することが可能です。

注記

ドメイン固有のロールは、トークンのスコープには使用できません。これはグローバルロールでのみ行うことができます。

2.2.4.1. ドメイン固有のロールの使用

この例では、ドメイン固有のロールを作成して、その効果を確認する方法を説明します。

1. ドメインを作成します。

```
$ openstack domain create corp01
```

2. ドメインを指定するロールを作成します (このパラメーターは **--domain** とは異なる点に注意してください)。

```
$ openstack role create operators --role-domain domain-corp01
```

第3章 グループの管理

3.1. KEYSTONE グループの管理

3.1.1. コマンドラインの使用

Identity サービス (keystone) グループを使用すると、一定のパーミッションを複数のユーザーアカウントに割り当てることができます。以下の例では、グループを作成して、そのグループにパーミッションを割り当てます。その結果、そのグループに割り当てられているのと同じパーミッションがグループのメンバーに継承されます。

注記

`openstack group` サブコマンドには keystone**v3** が必要です。

1. **`grp-Auditors`** というグループを作成します。

```
$ openstack group create grp-Auditors
+-------------+----------------------------------+
| Field       | Value                            |
+-------------+----------------------------------+
| description |                                  |
| domain_id   | default                          |
| id          | 2a4856fc242142a4aa7c02d28edfdfff |
| name        | grp-Auditors                     |
+-------------+----------------------------------+
```

2. keystone グループの一覧を表示します。

```
$ openstack group list --long
+----------------------------------+-------------+-----------
+-------------+
| ID                               | Name        | Domain ID |
Description |
+----------------------------------+-------------+-----------
+-------------+
| 2a4856fc242142a4aa7c02d28edfdfff | grp-Auditors | default   |
|
+----------------------------------+-------------+-----------
+-------------+
```

3. **`_member_`** ロールを使用して **`demo`** プロジェクトにアクセスするための**`grp-Auditors`** グループパーミッションを付与します。

```
$ openstack role add _member_ --group grp-Auditors --project demo
```

4. 既存のユーザー **`user1`** を **`grp-Auditors`** グループに追加します。

```
$ openstack group add user grp-Auditors user1
user1 added to group grp-Auditors
```

5. **user1** が **grp-Auditors** のメンバーであることを確認します。

```
$ openstack group contains user grp-Auditors user1
user1 in group grp-Auditors
```

6. **user1** に割り当てられている有効なパーミッションを確認します。

```
$ openstack role assignment list --effective --user user1
+----------------------------------+------------------------
---------+-------+----------------------------------+--------
+-----------+
| Role                             | User
| Group | Project                          | Domain | Inherited
|
+----------------------------------+------------------------
---------+-------+----------------------------------+--------
+-----------+
| 9fe2ff9ee4384b1894a90878d3e92bab |
3fefe5b4f6c948e6959d1feaef4822f2 |       |
0ce36252e2fb4ea8983bed2a568fa832 |        | False     |
+----------------------------------+------------------------
---------+-------+----------------------------------+--------
+-----------+
```

3.1.2. Dashboard の使用

Dashboard を使用して keystone グループのメンバーシップを管理することができます。グループへのロールパーミッションの割り当てには、上記の例で説明したようにコマンドラインを使用する必要があります。

3.1.2.1. グループの作成

1. Dashboard に管理ユーザーとしてログインして **アイデンティティー > グループ** を選択します。
2. **+グループの作成** をクリックします。
3. グループの名前と説明を入力します。
4. **グループの作成** をクリックします。

3.1.2.2. グループメンバーシップの管理

Dashboard を使用して keystone グループのメンバーシップを管理することができます。

1. Dashboard に管理ユーザーとしてログインして **アイデンティティー > グループ** を選択します。
2. 編集する必要のあるグループの **メンバーの管理** をクリックします。
3. **ユーザーの追加** を使用して、グループにユーザーを追加します。ユーザーを削除する必要がある場合には、そのユーザーのチェックボックスを選択して、**ユーザーの削除** をクリックします。

第4章 クォータ管理

4.1. クォータ管理

クラウド管理者は、プロジェクトのクォータを設定、管理できます。各プロジェクトには、リソースが割り当てられており、プロジェクトユーザーには、これらのリソースを使用するパーミッションが付与されます。これにより、相互のパーミッションやリソースを干渉することなく、複数のプロジェクトが単一のクラウドを使用できます。リソースクォータのセットは、新規テナントの作成時に事前設定されます。クォータには、テナントに割り当て可能な仮想 CPU、インスタンス、RAM、Floating IP の数量が含まれます。クォータは、テナント (またはプロジェクト) と、テナントのユーザーレベルの両方で強制できます。Dashboard を使用して新規/既存のテナントの Compute または Block Storage のクォータを設定または変更できる点に注意してください。Dashboard でのプロジェクトクォータの設定および更新の手順については、「5章 *プロジェクト管理*」を参照してください。

4.1.1. ユーザーのコンピュートクォータの表示

ユーザーに現在設定されているクォータの値を一覧表示するには、以下のコマンドを実行します。

```
$ nova quota-show --user [USER] --tenant [TENANT]
```

例

```
$ nova quota-show --user demoUser --tenant demo
+-----------------------------+-------+
| Quota                       | Limit |
+-----------------------------+-------+
| instances                   | 10    |
| cores                       | 20    |
| ram                         | 51200 |
| floating_ips                | 5     |
| fixed_ips                   | -1    |
| metadata_items              | 128   |
| injected_files              | 5     |
| injected_file_content_bytes | 10240 |
| injected_file_path_bytes    | 255   |
| key_pairs                   | 100   |
| security_groups             | 10    |
| security_group_rules        | 20    |
| server_groups               | 10    |
| server_group_members        | 10    |
+-----------------------------+-------+
```

4.1.2. ユーザーのコンピュートクォータの更新

特定のクォータ値を更新するには、以下のコマンドを実行します。

```
$ nova quota-update --user [USER] --[QUOTA_NAME] [QUOTA_VALUE] [TENANT]
$ nova quota-show --user [USER] --tenant [TENANT]
```

例

```
$ nova quota-update --user demoUser --floating-ips 10 demo
$ nova quota-show --user demoUser --tenant demo
+-----------------------------+-------+
| Quota                       | Limit |
+-----------------------------+-------+
| instances                   | 10    |
| cores                       | 20    |
| ram                         | 51200 |
| floating_ips                | 10    |
| ...                         |       |
+-----------------------------+-------+
```

注記

quota-update コマンドのオプション一覧を表示するには、以下を実行します。

```
$ nova help quota-update
```

4.1.3. ユーザーのオブジェクトストレージクォータの設定

オブジェクトストレージクォータは、以下のカテゴリーに分類できます。

- コンテナークォータ: 合計サイズ (バイト単位) または単一のコンテナーで保存可能なオブジェクト数を制限します。
- アカウントクォータ: Object Storage サービスでユーザーが利用可能な合計サイズ (バイト単位) を制限します。

コンテナークォータまたはアカウントクォータのいずれかを設定するには、Object Storage プロキシーサーバーにおいて、**proxy-server.conf** ファイルの **[pipeline:main]** セクションに **container_quotas** または **account_quotas** (または両方) のパラメーターを追加する必要があります。

```
[pipeline:main]
pipeline = catch_errors [...] tempauth container-quotas \
account-quotas slo dlo proxy-logging proxy-server

[filter:account_quotas]
use = egg:swift#account_quotas

[filter:container_quotas]
use = egg:swift#container_quotas
```

オブジェクトストレージクォータの表示および更新には、以下のコマンドを使用します。プロジェクトに含まれるすべてのユーザーには、そのプロジェクトに指定されているクォータが表示されます。プロジェクトに設定されているオブジェクトストレージのクォータを更新するには、そのプロジェクトの ResellerAdmin のロールが必要です。

アカウントクォータを表示するには、以下のコマンドを実行します。

```
# swift stat

Account: AUTH_b36ed2d326034beba0a9dd1fb19b70f9
```

```
Containers: 0
Objects: 0
Bytes: 0
Meta Quota-Bytes: 214748364800
X-Timestamp: 1351050521.29419
Content-Type: text/plain; charset=utf-8
Accept-Ranges: bytes
```

クォータを更新するには、以下を実行します。

```
# swift post -m quota-bytes:<BYTES>
```

たとえば、アカウントに 5 GB のクォータを指定します。

```
# swift post -m quota-bytes:5368709120
```

クォータの確認をするには **swift stat** コマンドをもう 1 度実行します。

```
# swift stat

Account: AUTH_b36ed2d326034beba0a9dd1fb19b70f9
Containers: 0
Objects: 0
Bytes: 0
Meta Quota-Bytes: 5368709120
X-Timestamp: 1351541410.38328
Content-Type: text/plain; charset=utf-8
Accept-Ranges: bytes
```

第5章 プロジェクト管理

5.1. プロジェクト管理

クラウド管理者は、プロジェクト (テナント) を作成、管理することができます。テナントは、OpenStack ユーザーとリソースの数が割り当てられたプロジェクトです。テナントごとにクォータを設定することができます。これにより、相互のパーミッションやリソースを干渉することなく、複数のプロジェクトが単一のクラウドを使用できるようになります。プロジェクトとテナントという用語はいずれも同じ意味で使用されます。ユーザーは、複数のプロジェクトに割り当てることができます。ユーザーとプロジェクトのペアごとに、ロールを 1 つ割り当てる必要があります。

5.1.1. プロジェクトの作成

プロジェクトの作成、プロジェクトへのメンバーの追加、プロジェクトのリソース制限の設定は、以下の手順を実行します。

1. Dashboard に管理ユーザーとしてログインして **アイデンティティー > プロジェクト** を選択します。
2. **プロジェクトの作成** をクリックします。
3. **プロジェクト情報** タブでプロジェクトの名前と説明を入力します (**有効** のチェックボックスはデフォルトで選択されます)。
4. プロジェクトへのメンバーの追加は、**プロジェクトメンバー** タブの **すべてのユーザー** リストから行います。
5. **クォータ** タブで、プロジェクトのリソースの上限を指定します。
6. **プロジェクトの作成** をクリックします。

5.1.2. プロジェクトの編集

プロジェクトを編集して名前や説明を変更したり、プロジェクトを有効化または一時的に無効化したり、メンバーを更新したりすることができます。

1. Dashboard に管理ユーザーとしてログインして **アイデンティティー > プロジェクト** を選択します。
2. プロジェクトの **アクション** コラムで、下向きの三角をクリックして**プロジェクトの編集** をクリックします。
3. **プロジェクトの編集** ウィンドウでプロジェクトを更新して名前や説明を変更したり、プロジェクトを有効化または一時的に無効化したりすることができます。
4. **プロジェクトメンバー** タブで、必要に応じてメンバーをプロジェクトに追加または削除します。
5. **保存** をクリックします。

注記

有効 のチェックボックスはデフォルトで選択されています。プロジェクトを一時的に無効にするには、**有効** のチェックボックスのチェックマークを外します。無効なプロジェクトを有効にするには、**有効** チェックボックスを選択します。

5.1.3. プロジェクトの削除

1. Dashboard に管理ユーザーとしてログインして **アイデンティティー > プロジェクト** を選択します。
2. 削除するプロジェクトを選択します。
3. **プロジェクトの削除** をクリックします。**プロジェクトの削除の確認** ウィンドウが表示されます。
4. **プロジェクトの削除** をクリックしてアクションを確認します。

プロジェクトが削除され、ユーザーとのペアリングの関連付けは解除されます。

5.1.4. プロジェクトクォータの更新

クォータとは、クラウドリソースを最適化するためにプロジェクトごとに設定可能な操作の制約のことです。クォータを設定して、通知なしにプロジェクトのリソースが使い果たされないようにします。クォータは、プロジェクトレベルとプロジェクトとユーザーレベルの両方で実行できます。

1. Dashboard に管理ユーザーとしてログインして **アイデンティティー > プロジェクト** を選択します。
2. プロジェクトの **アクション** コラムで、下向きの三角をクリックして**クォータの変更** をクリックします。
3. **クォータ** タブで、必要に応じてプロジェクトクォータを変更します。
4. **保存** をクリックします。

5.1.5. 現在のプロジェクトの変更

ユーザーは、メンバーとなっているプロジェクトのみ、現在のプロジェクトとして設定することができます。また、**現在のプロジェクトに設定** オプションを有効にするには、ユーザーが複数のプロジェクトのメンバーである必要があります。現在のプロジェクトとして設定すると、現在のプロジェクトとして指定されたプロジェクトのオブジェクトに、Dashboard からアクセスできるようになります。無効にしたプロジェクトは、有効化しない限り、現在のプロジェクトとして設定できません。

1. Dashboard に管理ユーザーとしてログインして **アイデンティティー > プロジェクト** を選択します。
2. プロジェクトの **アクション** コラムで、下向きの三角をクリックして**現在のプロジェクトに設定** をクリックします。

3. または、管理者権限のないユーザーで、プロジェクトの **アクション** コラムの下向きの三角をクリックして **現在のプロジェクトに設定** をクリックすると、このコラムのデフォルトアクションになります。

5.2. プロジェクトの階層

5.2.1. Identity サービスの階層型マルチテナンシー (HMT)

プロジェクトは、keystone のマルチテナンシーを使用して入れ子にすることができます。マルチテナンシーにより、サブプロジェクトは親プロジェクトのロール割り当てを継承することができます。

5.2.1.1. プロジェクトとサブプロジェクトの作成

階層型マルチテナンシー (HMT) は keystone のドメインとプロジェクトを使用して実装することができます。まず最初に新規ドメインを作成して、そのドメイン内にプロジェクトを作成します。これで、そのプロジェクトにサブプロジェクトを追加できるようになります。また、ユーザーをサブプロジェクトの **admin** ロールに追加すると、そのサブプロジェクトの管理者に昇格することができます。

注記

keystone の使用する HMT の構造は、現在 Dashboard では表示されません。

以下に例を示します。

1. corp という名前の keystone ドメインを新規作成します。

```
$ openstack domain create corp
+-------------+----------------------------------+
| Field       | Value                            |
+-------------+----------------------------------+
| description |                                  |
| enabled     | True                             |
| id          | 69436408fdcb44ab9e111691f8e9216d |
| name        | corp                             |
+-------------+----------------------------------+
```

2.corp ドメイン内に 親プロジェクト (**private-cloud**) を作成します。

```
$ openstack project create private-cloud --domain corp
+-------------+----------------------------------+
| Field       | Value                            |
+-------------+----------------------------------+
| description |                                  |
| domain_id   | 69436408fdcb44ab9e111691f8e9216d |
| enabled     | True                             |
| id          | c50d5cf4fe2e4929b98af5abdec3fd64 |
| is_domain   | False                            |
| name        | private-cloud                    |
| parent_id   | 69436408fdcb44ab9e111691f8e9216d |
+-------------+----------------------------------+
```

3. private-cloud の親プロジェクト内で corp ドメインも指定して サブプロジェクト (**dev**) を作成します。

```
$ openstack project create dev --parent private-cloud --domain corp
+-------------+----------------------------------+
| Field       | Value                            |
+-------------+----------------------------------+
| description |                                  |
| domain_id   | 69436408fdcb44ab9e111691f8e9216d |
| enabled     | True                             |
| id          | 11fccd8369824baa9fc87cf01023fd87 |
| is_domain   | False                            |
| name        | dev                              |
| parent_id   | c50d5cf4fe2e4929b98af5abdec3fd64 |
+-------------+----------------------------------+
```

4. qa という名前のサブプロジェクトをもう 1 つ作成します。

```
$ openstack project create qa --parent private-cloud --domain corp
+-------------+----------------------------------+
| Field       | Value                            |
+-------------+----------------------------------+
| description |                                  |
| domain_id   | 69436408fdcb44ab9e111691f8e9216d |
| enabled     | True                             |
| id          | b4f1d6f59ddf413fa040f062a0234871 |
| is_domain   | False                            |
| name        | qa                               |
| parent_id   | c50d5cf4fe2e4929b98af5abdec3fd64 |
+-------------+----------------------------------+
```

注記

Identity API を使用してプロジェクトの階層を確認することができます。詳しくは、https://developer.openstack.org/api-ref/identity/v3/index.html?expanded=show-project-details-detail を参照してください。

5.2.1.2. ユーザーへのアクセス権の付与

デフォルトでは、新規作成したプロジェクトにはロールは割り当てられません。ロールのパーミッションを親プロジェクトに割り当てる時には、**--inherited** フラグを指定して、サブプロジェクトが親プロジェクトからパーミッションを継承するように指定することができます。たとえば、親プロジェクトに対する **admin** ロールのアクセス権のあるユーザーには、サブプロジェクトへの **admin** アクセス権も付与されます。

1. プロジェクトに割り当てられている既存のパーミッションを確認します。

```
$ openstack role assignment list --project private-cloud
```

2. 既存のロールを確認します。

```
$ openstack role list
+----------------------------------+--------------+
```

```
| ID                                | Name          |
+-----------------------------------+---------------+
| 3a5137e4b620489791df1152ac013bfa | ResellerAdmin |
| 9fe2ff9ee4384b1894a90878d3e92bab | _member_      |
| cf4f87df933b455f957cf03b6d3784d2 | admin         |
| eef5cea6ff9549aa98cccc208c370d80 | SwiftOperator |
+-----------------------------------+---------------+
```

3. **private-cloud** プロジェクトに対する **user1** のアクセス権をユーザーアカウントに付与します。

```
$ openstack role add --user user1 --user-domain corp --project private-
cloud _member_
```

--inherited フラグを指定して上記のコマンドを再度実行すると、**user1** には、ロールの割り当てから継承された **private-cloud** サブプロジェクトへのアクセス権も付与されます。

```
$ openstack role add --user user1 --user-domain corp --project private-
cloud _member_ --inherited
```

4. パーミッションの更新の結果を確認します。

```
$ openstack role assignment list --effective --user user1 --user-domain
corp
+----------------------------------+------------------------------
---+-------+----------------------------------+--------+-----------
+
| Role                             | User
| Group | Project                          | Domain | Inherited |
+----------------------------------+------------------------------
---+-------+----------------------------------+--------+-----------
+
| 9fe2ff9ee4384b1894a90878d3e92bab | 10b5b34df21d485ca044433818d134be |
| c50d5cf4fe2e4929b98af5abdec3fd64 |        | False     |
| 9fe2ff9ee4384b1894a90878d3e92bab | 10b5b34df21d485ca044433818d134be |
| 11fccd8369824baa9fc87cf01023fd87 |        | True      |
| 9fe2ff9ee4384b1894a90878d3e92bab | 10b5b34df21d485ca044433818d134be |
| b4f1d6f59ddf413fa040f062a0234871 |        | True      |
+----------------------------------+------------------------------
---+-------+----------------------------------+--------+-----------
+
```

この結果では、**user1** が **qa** および **dev** プロジェクトへのアクセス権を継承していることが確認できます。また、親プロジェクトに **--inherited** フラグが適用されたので、**user1** は後ほど作成されるサブプロジェクトにはすべてアクセス権が自動的に付与されます。

5.2.2. アクセスの削除

明示的に割り当てられたパーミッションと継承されたパーミッションは別々に削除する必要があります。以下に例を示します。

1. 明示的に割り当てられたロールからユーザーを削除します。

```
$ openstack role remove --user user1 --project private-cloud _member_
```

2. 変更の結果を確認します。継承されたパーミッションがまだ存在している点に注意してください。

```
$ openstack role assignment list --effective --user user1 --user-domain
corp
+----------------------------------+----------------------------------
---+-------+----------------------------------+--------+-----------
+
| Role                             | User
| Group | Project                          | Domain | Inherited |
+----------------------------------+----------------------------------
---+-------+----------------------------------+--------+-----------
+
| 9fe2ff9ee4384b1894a90878d3e92bab | 10b5b34df21d485ca044433818d134be |
| 11fccd8369824baa9fc87cf01023fd87 |         | True      |
| 9fe2ff9ee4384b1894a90878d3e92bab | 10b5b34df21d485ca044433818d134be |
| b4f1d6f59ddf413fa040f062a0234871 |         | True      |
+----------------------------------+----------------------------------
---+-------+----------------------------------+--------+-----------
+
```

3. 継承されたパーミッションを削除します。

```
$ openstack role remove --user user1 --project private-cloud _member_ -
-inherited
```

4. 変更の結果を確認します。継承されたパーミッションが削除され、出力が空になりました。

```
$ openstack role assignment list --effective --user user1 --user-domain
corp
```

5.2.3. ネストされたクォータ

現時点では、**ネストされたクォータ** はまだサポートされていません。そのため、クォータはプロジェクトとサブプロジェクトで別々に管理する必要があります。

5.2.4. Reseller の概要

Reseller プロジェクトでは、複数のドメインを階層化することを目標としています。このようなドメインでは、1 つのサブドメインは、完全に有効化された 1 つのクラウドを表現し、最終的にはクラウドの部分的な再販を考慮することができます。この開発の作業は複数の段階に分かれています。第 1 段階については以下に説明します。

5.2.4.1. Reseller の第 1 段階

Reseller (第 1 段階) は、「Identity サービスの階層型マルチテナンシー (HMT)」に記載の階層型マルチテナンシー (HMT) の延長です。従来 keystone ドメインは、データベースバックエンド内に独自のテーブルを備えた、ユーザーとプロジェクトを保管するためのコンテナーとすることを目的としていました。その結果、ドメインは独自のテーブルには保管されなくなり、プロジェクトのテーブルにマージされました。

- ドメインは、プロジェクトの種別の一つとなり、**`is_domain`** フラグで区別されます。

- ドメインは、プロジェクト階層の最上位のプロジェクトを表します。ドメインは、プロジェクト階層のルートです。
- **projects** サブパスを使用してドメインの作成と取得をするように API が更新されました。
  - 新規ドメインを作成するには、**is_domain** フラグを true に指定してプロジェクトを作成します。
  - ドメインであるプロジェクトを一覧表示します。**is_domain** クエリーパラメーターを含むプロジェクトを取得します。

注記

第 1 段階では、ドメインの階層を作成することはできないので、サブドメインはまだ利用できません。また、これによりトークンのスコープは変わらず、keystone 以外のプロジェクトに必要な階層のサポートは実装されません。

5.3. プロジェクトのセキュリティー管理

セキュリティーグループとは、プロジェクトのインスタンスに割り当て可能な IP フィルターのルールセットで、インスタンスへのネットワークのアクセス権限を定義します。セキュリティーグループはプロジェクト別になっており、プロジェクトメンバーは自分のセキュリティーグループのデフォルトルールを編集して新規ルールセットを追加することができます。

プロジェクトにはすべて default セキュリティーグループが存在し、他にセキュリティーグループが定義されていないインスタンスに対して適用されます。このセキュリティーグループは、デフォルト値を変更しない限り、インスタンスへの受信トラフィックをすべて拒否し、送信トラフィックのみを許可します。

5.3.1. セキュリティーグループの作成

1. Dashboard で **プロジェクト > コンピュート > アクセスとセキュリティー**を選択します。
2. **セキュリティーグループ** タブで、**セキュリティーグループの作成** をクリックします。
3. セキュリティーグループに名前と説明を指定して、**セキュリティーグループの作成** をクリックします。

5.3.2. セキュリティーグループのルールの追加

デフォルトでは、新しいグループには、送信アクセスのルールのみが指定されます。他のアクセスを指定するには、新しいルールを追加する必要があります。

1. Dashboard で **プロジェクト > コンピュート > アクセスとセキュリティー**を選択します。
2. **セキュリティーグループ** タブで、編集するセキュリティーグループの**ルールの管理** をクリックします。
3. 新規ルールを追加するには、 **ルールの追加** をクリックします。
4. ルールの値を指定して、**追加** をクリックします。

   以下のルールのフィールドは必須です。

ルール

ルールタイプ。ルールテンプレート (例: **SSH**) を指定する場合には 、そのフィールドは自動的に入力されます。

- TCP: 一般的には、システム間のデータの交換や、エンドユーザーの通信に使用されます。
- UDP: 一般的には、システム間のデータ交換に (特にアプリケーションレベルで) 使用されます。
- ICMP: 一般的には、ルーターなどのネットワークデバイスがエラーや監視メッセージを送信するのに使用されます。

方向

受信 (インバウンド) または送信 (アウトバウンド)

開放するポート

TCP または UDP ルールでは、開放する **ポート** または **ポート範囲** (単一のポートまたはポートの範囲) を入力します。

- ポート範囲では、**ポート番号 (下限)** と **ポート番号 (上限)** にポートの値を入力します。
- 単一のポートの場合は **ポート** フィールドにポートの値を入力します。

タイプ

ICMP ルールのタイプ。**-1:255** の範囲で指定する必要があります。

コード

ICMP ルールのコード。**-1:255** の範囲で指定する必要があります。

接続相手

このルールが適用されるトラフィックの接続元

- CIDR (Classless Inter-Domain Routing): 指定のブロック内の IP へのアクセスを制限する IP アドレスブロック。接続相手フィールドに CIDR を入力します。
- セキュリティーグループ: グループ内のインスタンスが他のグループインスタンスにアクセスできるようにするソースのセキュリティーグループ

5.3.3. セキュリティーグループルールの削除

1. Dashboard で **プロジェクト > コンピュート > アクセスとセキュリティー**を選択します。
2. **セキュリティーグループ** タブで、セキュリティーグループの**ルールの管理** をクリックします。
3. セキュリティーグループルールを選択し、**イメージの削除** ボタンをクリックします。
4. 再度、**ルールの削除** をクリックします。

注記

削除の操作は元に戻すことはできません。

5.3.4. セキュリティーグループの削除

1. Dashboard で **プロジェクト > コンピュート > アクセスとセキュリティー**を選択します。
2. **セキュリティーグループ** タブで、グループを選択して、**セキュリティーグループの削除** をクリックします。
3. **セキュリティーグループの削除** をクリックします。

注記

削除の操作は元に戻すことはできません。

第6章 アイデンティティー管理

6.1. セキュアな LDAP 通信

Identity サービス (Keystone) が LDAP サーバーに対して認証を行うか、LDAP サーバーから識別情報を取得するように設定した場合に、CA 証明書を使用して Identity サービスの LDAP 通信をセキュリティー保護することができます。

本項では、Active Directory からの CA 証明書の取得、CA 証明書ファイルの Privacy Enhanced Mail (PEM) ファイル形式への変換、Identity サービスのセキュアな LDAP 通信設定の 3 つの方法について説明します。それぞれの方法での手順は、CA 信頼が設定された場所および方法に応じて実行するようにしてください。

6.1.1. Active Directory から CA 証明書を取得する方法

以下のコードは、Active Directory に対してクエリーを実行して CA 証明書を取得する方法の例を示しています。CA_NAME は証明書の名前に置き換え (mmc.exe で確認可能)、その他のパラメーターは実際の設定に応じて変更することができます。

```
CA_NAME="WIN2012DOM-WIN2012-CA"
AD_SUFFIX="dc=win2012dom,dc=com"
LDAPURL="ldap://win2012.win2012dom.com"
ADMIN_DN="cn=Administrator,cn=Users,$AD_SUFFIX"
ADMINPASSWORD="MyPassword"

CA_CERT_DN="cn=latexmath:[$CA_NAME,cn=certification
authorities,cn=public key
services,cn=services,cn=configuration,$]AD_SUFFIX"

TMP_CACERT=/tmp/cacert.`date +'%Y%m%d%H%M%S'`.$$.pem

ldapsearch -xLLL -H
latexmath:[$LDAPURL -D `echo \"$]ADMIN_DN"`-W -s base -b`echo
"$CA_CERT_DN"` objectclass=* cACertificate
```

6.1.2. CA 証明書を PEM ファイル形式に変換する方法

/path/cacert.pem という名前のファイルを作成し、以下の例に示したように、Active Directory から CA 証明書を取得するための LDAP クエリーの内容をヘッダーとフッターの間に追加します。

```
-----BEGIN CERTIFICATE-----
MIIDbzCCAlegAwIBAgIQQD14hh1Yz7tPFLXCkKUOszANB... -----END
CERTIFICATE-----
```

トラブルシューティングを行う場合には、以下のクエリーを実行して LDAP が稼働しているかをチェックし、PEN 証明書ファイルが正しく作成されたことを確認してください。

```
LDAPTLS_CACERT=/path/cacert.pem ldapsearch -xLLL -ZZ -H $LDAPURL -s
base -b "" "objectclass=*" currenttime
```

このクエリーによって、以下のような結果が返されるはずです。

```
dn: currentTime:
20141022050611.0Z
```

CA 証明書が Web サーバーでホストされていた場合には、以下のコマンドを実行して CA 証明書を取得することができます。

例

- $HOST=redhat.com
- $PORT=443

```
# echo Q | openssl s_client -connect $HOST:$PORT | sed -n -e
'/BEGIN CERTIFICATE/,/END CERTIFICATE/ p'
```

6.1.3. Identity サービスのセキュアな LDAP 通信を設定する方法

6.1.3.1. 方法 1

CA 信頼が PEM ファイルを使用して LDAP レベルで設定されている場合は、この方法を使用してください。CA 証明書ファイルの場所は手動で指定します。以下の手順では、Identity サービスのみでなく、OpenLDAP ライブラリーを使用する全アプリケーションの LDAP 通信がセキュリティー保護されます。

1. CA 証明書チェーンが含まれているファイルを PEM 形式で **/etc/openldap/certs** ディレクトリーにコピーします。
2. **/etc/openldap/ldap.conf** を編集して以下のディレクティブを追加します。[CA_FILE] は CA 証明書ファイルの場所と名前に置き換えます。

   ```
   TLS_CACERT /etc/openldap/certs/[CA_FILE]
   ```

3. **httpd** サービスを再起動します。

   ```
   # systemctl restart httpd.service
   ```

6.1.3.2. 方法 2

CA 信頼が Network Security Services (NSS) データベースを介して LDAP ライブラリーレベルで設定されている場合は、この方法を使用してください。**certutil** コマンドを使用して、OpenLDAP ライブラリーが使用する NSS 証明書データベースに CA 証明書をインポートして信頼します。以下の手順では、Identity サービスのみでなく、OpenLDAP ライブラリーを使用する全アプリケーションの LDAP 通信がセキュリティー保護されます。

1. 1. 証明書をインポートして信頼します。[CA_FILE] は CA 証明書ファイルの場所と名前に置き換えます。

   ```
   # certutil -d /etc/openldap/certs -A -n "My CA" -t CT,, -a -i
   [CA_FILE]
   # certutil -d /etc/openldap/certs -A -n "My CA" -t CT,, -a -i
   [CA_FILE]
   ```

2. CA 証明書が正しくインポートされていることを確認します。

```
# certutil -d /etc/openldap/certs -L
```

CA 証明書がリストされ、信頼の属性が **CT,,** に設定されます。

3. **httpd** サービスを再起動します。

```
# systemctl restart httpd.service
```

6.1.3.3. 方法 3

CA 信頼が PEM ファイルを使用して Keystone レベルで設定されている場合は、この方法を使用してください。Identity サービスと LDAP サーバー間の通信をセキュリティー保護する最後のメソッドは、Identity サービスに TLS を設定する方法です。

ただし、上記の 2 つのメソッドとは異なり、このメソッドでは、Identity サービスの LDAP 通信のみがセキュリティー保護され、OpenLDAP ライブラリーを使用する他のアプリケーションの LDAP 通信はセキュリティー保護されません。

以下の手順では、**openstack-config** コマンドを使用して **/etc/keystone/keystone.conf** ファイル内の値を編集します。

1. TLS を有効化します。

```
# openstack-config --set /etc/keystone/keystone.conf ldap use_tls
True
```

2. 証明書の場所を指定します。[CA_FILE] は CA 証明書ファイルの名前に置き換えます。

```
# openstack-config --set /etc/keystone/keystone.conf ldap
tls_cacertfile [CA_FILE]
```

3. LDAP サーバーから受信した TLS セッションに対して実行するクライアント証明書チェックを指定します。[CERT_BEHAVIOR] は以下にあげる動作のいずれか 1 つに置き換えてください。

demand

LDAP サーバーにより証明書が常に要求されます。証明書が提供されなかった場合、または提供された証明書が既存の認証局ファイルに対して検証できなかった場合には、セッションは終了します。

allow

LDAP サーバーにより証明書が常に要求されます。証明書が提供されなくてもセッションは通常どおりに続行されます。証明書が提供されたが、既存の認証局ファイルに対して検証できなかった場合には、その証明書は無視され、セッションは通常通りに続行します。

never

証明書は一切要求されません。

```
# openstack-config --set /etc/keystone/keystone.conf ldap
tls_req_cert [CERT_BEHAVIOR]
```

4. **httpd** サービスを再起動します。

```
# systemctl restart httpd.service
```