Red Hat OpenStack Platform 16.2

Integrate OpenStack Identity with external user management services

Use Active Directory or Red Hat Identity Management as an external authentication back end

Last Updated: 2021-10-25

Red Hat OpenStack Platform 16.2 Integrate OpenStack Identity with external user management services

Use Active Directory or Red Hat Identity Management as an external authentication back end

OpenStack Team
rhos-docs@redhat.com

Legal Notice

Abstract

Integrate the OpenStack Identity (keystone) service with Microsoft Active Directory Domain Service (AD DS), Red Hat Identity Management (IdM), and LDAP.

Table of Contents

MAKING OPEN SOURCE MORE INCLUSIVE

Red Hat is committed to replacing problematic language in our code, documentation, and web properties. We are beginning with these four terms: master, slave, blacklist, and whitelist. Because of the enormity of this endeavor, these changes will be implemented gradually over several upcoming releases. For more details, see our CTO Chris Wright's message.

PROVIDING FEEDBACK ON RED HAT DOCUMENTATION

We appreciate your input on our documentation. Tell us how we can make it better.

Using the Direct Documentation Feedback (DDF) function

Use the **Add Feedback** DDF function for direct comments on specific sentences, paragraphs, or code blocks.

1. View the documentation in the *Multi-page HTML* format.
2. Ensure that you see the **Feedback** button in the upper right corner of the document.
3. Highlight the part of text that you want to comment on.
4. Click **Add Feedback**.
5. Complete the **Add Feedback** field with your comments.
6. Optional: Add your email address so that the documentation team can contact you for clarification on your issue.
7. Click **Submit**.

CHAPTER 1. INTEGRATING OPENSTACK IDENTITY (KEYSTONE) WITH ACTIVE DIRECTORY

You can integrate OpenStack Identity (keystone) with Microsoft Active Directory Domain Service (AD DS). Identity Service authenticates certain Active Directory Domain Services (AD DS) users but retains authorization settings and critical service accounts in the Identity Service database. As a result, Identity Service has read-only access to AD DS for user account authentication and continues to manage privileges assigned to authenticated accounts.

By integrating the Identity service with AD DS, you allow AD DS users to authenticate to Red Hat OpenStack Platform (RHOSP) to access resources. RHOSP service accounts, such as the Identity Service and the Image service, and authorization management remain in the Identity Service database. Permissions and roles are assigned to the AD DS accounts using Identity Service management tools.

The process to integrate OpenStack Identity with Active Directory includes the following stages:

1. Configure Active Directory credentials and export the LDAPS certificate
2. Install and configure the LDAPS certificate in OpenStack
3. Configure director to use one or more LDAP backends
4. Configure Controller nodes to access the Active Directory backend
5. Configure Active Directory user or group access to OpenStack projects
6. Verify that the domain and user lists are created correctly
7. Optional: Create credential files for non-admin users.

1.1. CONFIGURING ACTIVE DIRECTORY CREDENTIALS

To configure Active Directory Domain Service (AD DS) to integrate with OpenStack Identity, set up an LDAP account for Identity service to use, create a user group for Red Hat OpenStack users, and export the LDAPS certificate public key to use in the Red Hat OpenStack Platform deployment.

Prerequisites

- Active Directory Domain Services is configured and operational.
- Red Hat OpenStack Platform is configured and operational.
- DNS name resolution is fully functional and all hosts are registered appropriately.
- AD DS authentication traffic is encrypted with LDAPS, using port 636.
- Recommended: Implement AD DS with a high availability or load balancing solution to avoid a single point of failure.

Procedure

Perform these steps on the Active Directory server.

1. Create the LDAP lookup account. This account is used by Identity Service to query the AD DS LDAP service:

```
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. Set a password for this account, and then enable it. You will be prompted to specify a password that complies with your AD domain's complexity requirements:

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

3. Create a group for RHOSP users, called **grp-openstack**. Only members of this group can have permissions assigned in OpenStack Identity.

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

4. Create the Project groups:

```
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. Add the **svc-ldap** user to the **grp-openstack** group:

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

6. From an AD Domain Controller, use a **Certificates MMC** to export your LDAPS certificate's public key (not the private key) as a DER-encoded **x509** .cer file. Send this file to the RHOSP administrators.

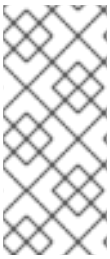
7. Retrieve the NetBIOS name of your AD DS domain.

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

Send this value to the RHOSP administrators.

1.2. INSTALLING THE ACTIVE DIRECTORY LDAPS CERTIFICATE

OpenStack Identity (keystone) uses LDAPS queries to validate user accounts. To encrypt this traffic, keystone uses the certificate file defined by **keystone.conf**. To configure the LDAPS certificate, convert the public key received from Active Directory into the **.crt** format and copy the certificate to a location where keystone will be able to reference it.



NOTE

When using multiple domains for LDAP authentication, you might receive various errors, such as **Unable to retrieve authorized projects**, or **Peer's Certificate issuer is not recognized**. This can arise if keystone uses the incorrect certificate for a certain domain. As a workaround, merge all of the LDAPS public keys into a single **.crt** bundle, and configure all of your keystone domains to use this file.

Prerequisites

- Active Directory credentials are configured.
- LDAPS certificate is exported from Active Directory.

Procedure

1. Copy the LDAPS public key to the node running OpenStack Identity and convert the **.cer** to **.crt**. This example uses a source certificate file named **addc.lab.local.cer**:

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

2. Optional: If you need to run diagnostic commands, such as **ldapsearch**, you also need to add the certificate to the RHEL certificate store:

   a. Convert the **.cer** to **.pem**. This example uses a source certificate file named **addc.lab.local.cer**:

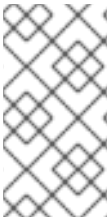
```
# openssl x509 -inform der -in addc.lab.local.cer -out addc.lab.local.pem
```

   b. Install the **.pem** on the Controller node. For example, in Red Hat Enterprise Linux:

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

1.3. CONFIGURING DIRECTOR TO USE DOMAIN-SPECIFIC LDAP BACKENDS

To configure director to use one or more LDAP backends, set the **KeystoneLDAPDomainEnable** flag to **true** in your heat templates, and set up environment files with the information about each LDAP backend. Director then uses a separate LDAP backend for each keystone domain.



NOTE

The default directory for domain configuration files is set to **/etc/keystone/domains/**. You can override this by setting the required path with the **keystone::domain_config_directory** hiera key and adding it as an **ExtraConfig** parameter within an environment file.

Procedure

1. In the heat template for your deployment, set the **KeystoneLDAPDomainEnable** flag to **true**. This configures the **domain_specific_drivers_enabled** option in keystone within the **identity** configuration group.
2. Add a specification of the LDAP backend configuration by setting the **KeystoneLDAPBackendConfigs** parameter in **tripleo-heat-templates**, where you can then specify your required LDAP options.
3. Create a copy of the **keystone_domain_specific_ldap_backend.yaml** environment file:

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

4. Edit the **/home/stack/templates/keystone_domain_specific_ldap_backend.yaml** environment file and set the values to suit your deployment. For example, this parameter create a LDAP configuration for a keystone domain named **testdomain**:

```
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
```

NOTE

The **keystone_domain_specific_ldap_backend.yaml** environment file contains the following deprecated write parameters:

- **user_allow_create**
- **user_allow_update**
- **user_allow_delete**

The values for these parameters have no effect on the deployment, and can be safely removed.

5. Optional: Add more domains to the environment file. For example:

```
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
    ...
```

This results in two domains named **domain1** and **domain2**; each will have a different LDAP domain with its own configuration.

1.4. CONFIGURING OPENSTACK IDENTITY DOMAINS ON CONTROLLER NODES

To configure Controller nodes that run OpenStack Identity (keystone) to integrate with an external user management service, first configure SELinux to use LDAP authentication and create the **domains** directory on the Controller node. Then configure OpenStack Identity to use multiple back ends and the Dashboard to use multiple domains.

NOTE

If you are using director, note that the configuration files referenced in this procedure are managed by Puppet. Therefore, any custom configuration you add might be overwritten whenever you run the **openstack overcloud deploy** command.

Planning

If you intend to update any configuration files, you need to be aware that certain OpenStack services now run within containers; this applies to keystone, nova, and cinder, among others. As a result, there are certain administration practices to consider:

- Do not update any configuration file you might find on the physical node's host operating system, for example, **/etc/cinder/cinder.conf**. This is because the containerized service does not reference this file.
- Do not update the configuration file running within the container. This is because any changes are lost once you restart the container.
  Instead, if you need to add any changes to containerized services, you will need to update the configuration file that is used to generate the container. These are stored within **/var/lib/config-data/puppet-generated/**

  For example:

  - 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**

  Any changes will then be applied once you restart the service. For example: **sudo systemctl restart tripleo_keystone**

Procedure

Perform this procedure on each Controller node that runs the OpenStack Identity (keystone) service.

1. Configure SELinux:

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

   The output might include messages similar to this. They can be ignored:

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

2. Create the **domains** directory:

```
# 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. Configure keystone to use multiple back ends:

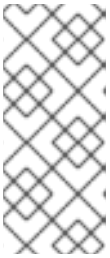
NOTE

You might need to install **crudini** using **dnf install crudini**.

```
# 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
```

4. Enable multiple domains in dashboard. Add these lines to **/var/lib/config-data/puppet-generated/horizon/etc/openstack-dashboard/local_settings**:

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



NOTE

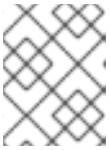
If you are using director, note that **/var/lib/config-data/puppet-generated/horizon/etc/openstack-dashboard/local_settings** is managed by Puppet. Consequently, any custom configuration you add might be overwritten whenever you run the **openstack overcloud deploy** process. As a result, you might need to re-add this configuration manually each time.

Restart the horizon container to apply the settings:

```
$ sudo systemctl restart tripleo_horizon
```

5. Create the keystone domain for the external service integration with the NetBIOS name value retrieved previously as the domain name. This approach allows you to present a consistent domain name to users during the login process. In this example, **LAB** is the NetBIOS name to use as the Identity Service domain.

```
$ openstack domain create LAB
```



NOTE

If this command is not available, check that you have enabled keystone v3 for your command line session by running **# source overcloudrc-v3**.

6. Create the configuration file for the external service that you are integrating:

- Active Directory Domain Service (AD DS): Enter the LDAP settings in a new file called **/var/lib/config-data/puppet-generated/keystone/etc/keystone/domains/keystone.LAB.conf** (where **LAB** is the NetBIOS name retrieved previously). You will need to edit the sample settings below to suit your AD DS deployment:

```
[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
```

Explanation of each setting:

Setting	Description
url	The AD Domain Controller to use for authentication. Uses LDAPS port **636**.
user	The *Distinguished Name* of an AD account to use for LDAP queries. For example, you can locate the *Distinguished Name* value of the *svc-ldap* account in AD using **Get-ADuser svc-ldap | select DistinguishedName**
password	The plaintext password of the AD account used above.

Setting	Description
suffix	The *Distinguished Name* of your AD domain. You can locate this value using **Get-ADDomain | select DistinguishedName**
user_tree_dn	The *Organizational Unit* (OU) that contains the OpenStack accounts.
user_objectclass	Defines the type of LDAP user. For AD, use the **person** type.
user_filter	Filters the users presented to Identity Service. As a result, only members of the **grp-openstack** group can have permissions defined in Identity Service. This value requires the full *Distinguished Name* of the group: **Get-ADGroup grp-openstack | select DistinguishedName**
user_id_attribute	Maps the AD value to use for user IDs.
user_name_attribute	Maps the AD value to use for *names*.
user_mail_attribute	Maps the AD value to use for user email addresses.
user_pass_attribute	Leave this value blank.
user_enabled_attribute	The AD setting that validates whether the account is enabled.
user_enabled_mask	Defines the value to check to determine whether an account is enabled. Used when booleans are not returned.
user_enabled_default	The AD value that indicates that an account is enabled.
user_attribute_ignore	Defines user attributes that Identity Service should disregard.
group_objectclass	Maps the AD value to use for *groups*.
group_tree_dn	The *Organizational Unit* (OU) that contains the user groups.
group_filter	Filters the groups presented to Identity Service.

Setting	Description
group_id_attribute	Maps the AD value to use for group IDs.
group_name_attribute	Maps the AD value to use for group names.
use_tls	Defines whether TLS is to be used. This needs to be disabled if you are encrypting with LDAPS rather than STARTTLS.
tls_cacertfile	Specifies the path to the *.crt* certificate file.
query_scope	Configures Identity Service to also search within nested child OUs, when locating users that are members of the **grp-openstack** group.
chase_referrals	Set to **false**, this setting prevents**python-ldap** from chasing all referrals with anonymous access.

- Red Hat Identity Manager (IdM): Enter the LDAP settings in a new file called **/var/lib/config-data/puppet-generated/keystone/etc/keystone/domains/keystone.LAB.conf** (where **LAB** is the domain name created previously). You will need to edit the sample settings below to suit your IdM deployment:

```
[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
```

Explanation of each setting:

Setting	Description
url	The IdM server to use for authentication. Uses LDAPS port **636**.
user	The account in IdM to use for LDAP queries.
password	The plaintext password of the IdM account used above.
user_filter	Filters the users presented to Identity Service. As a result, only members of the **grp-openstack** group can have permissions defined in Identity Service.
user_tree_dn	The path to the OpenStack accounts in IdM.
user_objectclass	Defines the type of LDAP user. For IdM, use the **inetUser** type.
user_id_attribute	Maps the IdM value to use for user IDs.
user_name_attribute	Maps the IdM value to use for *names*.
user_mail_attribute	Maps the IdM value to use for user email addresses.
user_pass_attribute	Leave this value blank.

NOTE

Integration with an IdM group will only return direct members, and not nested groups. As a result, queries that rely on **LDAP_MATCHING_RULE_IN_CHAIN** or **memberof:1.2.840.113556.1.4.1941:** will not currently work with IdM.

7. Change ownership of the configuration file to the keystone user:

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

8. Restart the keystone service to apply the changes:

```
# sudo systemctl restart tripleo_keystone
```

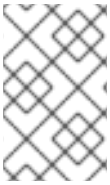
-

Additional resources

- Section 1.5, “Granting the admin user access to the OpenStack Identity domain”
- Section 1.3, “Configuring director to use domain-specific LDAP backends”

1.5. GRANTING THE ADMIN USER ACCESS TO THE OPENSTACK IDENTITY DOMAIN

To allow the **admin** user to access the OpenStack Identity (keystone) domain and see the **Domain** tab, get the ID of the domain and the **admin** user, and then assign the **admin** role to the user in the domain.



NOTE

This does not grant the OpenStack admin account any permissions on the external service domain. In this case, the term *domain* refers to OpenStack's usage of the keystone domain.

Procedure

This procedure uses the **LAB** domain. Replace the domain name with the actual name of the domain that you are configuring.

1. Get the ID of the **LAB** domain:

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

2. Get the ID of the **admin** user from the **default** domain:

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

3. Get the ID of the **admin** role:

```
$ openstack role list
```

The output depends on the external service you are integrating with:

- Active Directory Domain Service (AD DS):

```
+----------------------------------+----------------+
| ID                               | Name           |
+----------------------------------+----------------+
| 01d92614cd224a589bdf3b171afc5488 | admin          |
| 034e4620ed3d45969dfe8992af001514 | member         |
```

```
| 0aa377a807df4149b0a8c69b9560b106 | ResellerAdmin   |
| 9369f2bf754443f199c6d6b96479b1fa | heat_stack_user |
| cfea5760d9c948e7b362abc1d06e557f | reader          |
| d5cb454559e44b47aaa8821df4e11af1 | swiftoperator   |
| ef3d3f510a474d6c860b4098ad658a29 | service         |
+----------------------------------+-----------------+
```

- Red Hat Identity Manager (IdM):

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

4. Use the domain and admin IDs to construct the command that adds the **admin** user to the **admin** role of the keystone **LAB** domain:

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

1.6. GRANTING EXTERNAL GROUPS ACCESS TO RED HAT OPENSTACK PLATFORM PROJECTS

To grant multiple authenticated users access to Red Hat OpenStack Platform (RHOSP) resources, you can authorize certain groups from the external user management service to grant access to RHOSP projects, instead of requiring OpenStack administrators to manually allocate each user to a role in a project. As a result, all members of these groups can access pre-determined projects.

Prerequisites

- Ensure that the external service administrator completed the following steps:
  - Creating a group named **grp-openstack-admin**.
  - Creating a group named **grp-openstack-demo**.
  - Adding your RHOSP users to one of these groups as needed.
  - Adding your users to the **grp-openstack** group.
- Create the OpenStack Identity domain. This procedure uses the **LAB** domain.
- Create or choose a RHOSP project. This procedure uses a project called **demo** that was created with the **openstack project create --domain default --description "Demo Project" demo** command.

Procedure

1. Retrieve a list of user groups from the OpenStack Identity domain:

```
# openstack group list --domain LAB
```

The command output depends on the external user management service that you are integrating with:

- Active Directory Domain Service (AD DS):

```
+------------------------------------------------------------------+--------------------+
| ID                                                               | Name               |
+------------------------------------------------------------------+--------------------+
| 185277be62ae17e498a69f98a59b66934fb1d6b7f745f14f5f68953a665b8851 | grp-
openstack       |
| a8d17f19f464c4548c18b97e4aa331820f9d3be52654aa8094e698a9182cbb88 | grp-
openstack-admin |
| d971bb3bd5e64a454cbd0cc7af4c0773e78d61b5f81321809f8323216938cae8 | grp-
openstack-demo  |
+------------------------------------------------------------------+--------------------+
```

- Red Hat Identity Manager (IdM):

```
+------------------------------------------------------------------+--------------------+
| ID                                                               | Name               |
+------------------------------------------------------------------+--------------------+
| 185277be62ae17e498a69f98a59b66934fb1d6b7f745f14f5f68953a665b8851 | grp-
openstack       |
| a8d17f19f464c4548c18b97e4aa331820f9d3be52654aa8094e698a9182cbb88 | grp-
openstack-admin |
| d971bb3bd5e64a454cbd0cc7af4c0773e78d61b5f81321809f8323216938cae8 | grp-
openstack-demo  |
+------------------------------------------------------------------+--------------------+
```

2. Retrieve a list of roles:

```
# openstack role list
```

The command output depends on the external user management service that you are integrating with:

- Active Directory Domain Service (AD DS):

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

- Red Hat Identity Manager (IdM):

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

3. Grant the user groups access to RHOSP projects by adding them to one or more of these roles. For example, if you want users in the **grp-openstack-demo** group to be general users of the **demo** project, you must add the group to the **member** or **_member_** role, depending on the external service that you are integrating with:

   - Active Directory Domain Service (AD DS):

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

   - Red Hat Identity Manager (IdM):

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

Result

Members of **grp-openstack-demo** can log in to the dashboard by entering their username and password and entering **LAB** in the **Domain** field:

NOTE

If users receive the error **Error: Unable to retrieve container list.**, and expect to be able to manage containers, then they must be added to the **SwiftOperator** role.

Additional resources

- Section 1.7, “Granting external users access to Red Hat OpenStack Platform projects”

1.7. GRANTING EXTERNAL USERS ACCESS TO RED HAT OPENSTACK PLATFORM PROJECTS

To grant specific authenticated users from the **grp-openstack** group access to OpenStack resources, you can grant these users direct access to Red Hat OpenStack Platform (RHOSP) projects. Use this process in cases where you want to grant access to individual users instead of granting access to groups.

Prerequisites

- Ensure that the external service administrator completed the following steps:
  - Adding your RHOSP users to the **grp-openstack** group.
  - Creating the OpenStack Identity domain. This procedure uses the **LAB** domain.
- Create or choose a RHOSP project. This procedure uses a project called **demo** that was created with the **openstack project create --domain default --description "Demo Project" demo** command.

Procedure

1. Retrieve a list of users from the OpenStack Identity domain:

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

2. Retrieve a list of roles:

```
# openstack role list
```

The command output depends on the external user management service that you are integrating with:

- Active Directory Domain Service (AD DS):

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

```
| d5cb454559e44b47aaa8821df4e11af1 | swiftoperator   |
| ef3d3f510a474d6c860b4098ad658a29 | service         |
+----------------------------------+-----------------+
```

- Red Hat Identity Manager (IdM):

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

3. Grant users access to RHOSP projects by adding them to one or more of these roles. For example, if you want **user1** to be a general user of the **demo** project, you add them to the **member** or **_member_** role, depending on the external service that you are integrating with:

   - Active Directory Domain Service (AD DS):

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

   - Red Hat Identity Manager (IdM):

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

4. If you want **user1** to be an administrative user of the **demo** project, add the user to the **admin** role:

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

Result

The **user1** user is able to log in to the dashboard by entering their external username and password and entering **LAB** in the **Domain** field:

NOTE

If users receive the error **Error: Unable to retrieve container list.**, and expect to be able to manage containers, then they must be added to the **SwiftOperator** role.

Additional resources

- Section 1.6, "Granting external groups access to Red Hat OpenStack Platform projects"

1.8. VIEWING THE LIST OF OPENSTACK IDENTITY DOMAINS AND USERS

Use the **openstack domain list** command to list the available entries. Configuring multiple domains in Identity Service enables a new **Domain** field in the dashboard login page. Users are expected to enter the domain that matches their login credentials.

IMPORTANT

After you complete the integration, you need to decide whether to create new projects in the **Default** domain or in newly created keystone domains. You must consider your workflow and how you administer user accounts. If possible, use the **Default** domain as an internal domain to manage service accounts and the **admin** project, and keep your external users in a separate domain.

In this example, external accounts need to specify the **LAB** domain. The built-in keystone accounts, such as **admin**, must specify **Default** as their domain.

Procedure

1. Show the list of domains:

```
# openstack domain list
+----------------------------------+---------+---------+-----------------------------------------------------
--------------+
| ID                               | Name    | Enabled | Description                                       |
```

```
+----------------------------------+---------+---------+-----------------------------------------------------
--------------+
| 6800b0496429431ab1c4efbb3fe810d4 | LAB     | True    |
|
| default                          | Default | True    | Owns users and projects available on Identity API
v2. |
+----------------------------------+---------+---------+-----------------------------------------------------
--------------+
```

2. Show the list of users in a specific domain. This command example specifies the **--domain LAB** and returns users in the LAB domain that are members of the **grp-openstack** group:

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

You can also append **--domain Default** to show the built-in keystone accounts:

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

1.9. CREATING A CREDENTIALS FILE FOR A NON-ADMIN USER

After you configure users and domains for OpenStack Identity, you might need to create a credentials file for a non-admin user.

Procedure

- Create a credentials (RC) file for a non-admin user. This example uses the **user1** user in the file.

```
$ 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
```

1.10. TESTING OPENSTACK IDENTITY INTEGRATION WITH AN EXTERNAL USER MANAGEMENT SERVICE

To test that OpenStack Identity (keystone) successfully integrated with Active Directory Domain Service (AD DS), test user access to dashboard features.

Prerequisites

- Integration with an external user management service, such as Active Directory (AD) or Red Hat Identity Manager (IdM)

Procedure

1. Create a test user in the external user management service, and add the user to the **grp-openstack** group.
2. In Red Hat OpenStack Platform, add the user to the **_member_** role of the **demo** project.
3. Log in to the dashboard with the credentials of the AD test user.
4. Click on each of the tabs to confirm that they are presented successfully without error messages.
5. Use the dashboard to build a test instance.

NOTE

If you experience issues with these steps, log in to the dashboard with the **admin** account and perform the subsequent steps as that user. If the test is successful, it means that OpenStack is still working as expected and that an issue exists somewhere in the integration settings between OpenStack Identity and Active Directory.

Additional resources

- Section 1.11, “Troubleshooting Active Directory integration”

1.11. TROUBLESHOOTING ACTIVE DIRECTORY INTEGRATION

If you encounter errors when using the Active Directory integration with OpenStack Identity, you might need to test the LDAP connection or test the certificate trust configuration. You might also need to check that the LDAPS port is accessible.

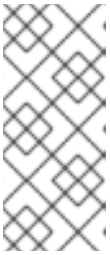
NOTE

Depending on the error type and location, perform only the relevant steps in this procedure.

Procedure

1. Test the LDAP connection by using the **ldapsearch** command to remotely perform test queries against the Active Directory Domain Controller. A successful result here indicates that network connectivity is working, and the AD DS services are up. In this example, a test query is performed against the server **addc.lab.local** on port **636**:

```
# 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
```



NOTE

- **ldapsearch** is a part of the **openldap-clients** package. You can install this using **# dnf install openldap-clients**
- This command expects to find the necessary certificate in your host operating system.

2. If you receive the error **Peer's Certificate issuer is not recognized.** while testing the **ldapsearch** command, confirm that your **TLS_CACERTDIR** path is correctly set. For example:

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

3. As a temporary workaround, consider disabling certificate validation.

IMPORTANT

This setting must not be permanently configured.

In the **/etc/openldap/ldap.conf**, set the **TLS_REQCERT** parameter to **allow**:

```
TLS_REQCERT allow
```

If the **ldapsearch** query works after setting this value, you might need to review whether your certificate trusts are correctly configured.

4. Use the **nc** command to check that LDAPS port **636** is remotely accessible. In this example, a probe is performed against the server **addc.lab.local**. Press **ctrl-c** to exit the prompt.

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

Failure to establish a connection might indicate a firewall configuration issue.

CHAPTER 2. INTEGRATING OPENSTACK IDENTITY (KEYSTONE) WITH RED HAT IDENTITY MANAGER (IDM)

When you integrate OpenStack Identity (keystone) with Red Hat Identity Manager (IdM), OpenStack Identity authenticates certain Red Hat Identity Management (IdM) users but retains authorization settings and critical service accounts in the Identity Service database. As a result, Identity Service has read-only access to IdM for user account authentication, while retaining management over the privileges assigned to authenticated accounts. You can also use **novajoin** to enroll your nodes with IdM.

NOTE

The configuration files for this integration are managed by Puppet. Therefore, any custom configuration that you add might be overwritten the next time you run the **openstack overcloud deploy** command. You can use director to configure LDAP authentication instead of manually editing the configuration files.

Review the following key terms before you plan and configure the IdM integration:

- **Authentication** - The process of using a password to verify that the user is who they claim to be.
- **Authorization** - Validating that authenticated users have proper permissions to the systems they're attempting to access.
- **Domain** - Refers to the additional back ends configured in Identity Service. For example, Identity Service can be configured to authenticate users from external IdM environments. The resulting collection of users can be thought of as a *domain*.

The process to integrate OpenStack Identity with IdM includes the following stages:

1. Enroll the undercloud and overcloud in IdM with novajoin
2. Implement TLS-e on the undercloud and overcloud with Ansible
3. Configure IdM server credentials and export the LDAPS certificate
4. Install and configure the LDAPS certificate in OpenStack
5. Configure director to use one or more LDAP backends
6. Configure Controller nodes to access the IdM backend
7. Configure IdM user or group access to OpenStack projects
8. Verify that the domain and user lists are created correctly
9. Optional: Create credential files for non-admin users

2.1. PLANNING THE RED HAT IDENTITY MANAGER (IDM) INTEGRATION

When you plan your OpenStack Identity integration with Red Hat Identity Manager (IdM), ensure that both services are configured and operational and review the impact of the integration on user management and firewall settings. Red Hat Identity Manager IdM depends on SRV records to do load

balancing. You should not put a load balancer in front of IdM.

Prerequisites

- Red Hat Identity Management is configured and operational.
- Red Hat OpenStack Platform is configured and operational.
- DNS name resolution is fully functional and all hosts are registered appropriately.

Permissions and roles

This integration allows IdM users to authenticate to OpenStack and access resources. OpenStack service accounts (such as keystone and glance), and authorization management (permissions and roles) will remain in the Identity Service database. Permissions and roles are assigned to the IdM accounts using Identity Service management tools.

High availability options

This configuration creates a dependency on the availability of a single IdM server: Project users will be affected if Identity Service is unable to authenticate to the IdM Server. It is not recommended to place a load balancer in front of IdM, however you can configure keystone to query a different IdM server, should one become unavailable.

Outage requirements

- The Identity Service will need to be restarted in order to add the IdM back end.
- Users will be unable to access the dashboard until their accounts have been created in IdM. To reduce downtime, consider pre-staging the IdM accounts well in advance of this change.

Firewall configuration

Communication between IdM and OpenStack consists of the following:

- Authenticating users
- IdM retrieval of the certificate revocation list (CRL) from the controllers every two hours
- Certmonger requests for new certificates upon expiration

NOTE

A periodic certmonger task will continue to request new certificates if the initial request fails.

If firewalls are filtering traffic between IdM and OpenStack, you will need to allow access through the following port:

+

Source	Destination	Type	Port
OpenStack Controller Node	Red Hat Identity Management	LDAPS	TCP 636

2.2. ENROLLING NODES IN RED HAT IDENTITY MANAGER (IDM) WITH NOVAJOIN

Novajoin is the default tool that you use to enroll your nodes with Red Hat Identity Manager (IdM) as part of the deployment process. As a result, you can integrate IdM features with your Red Hat OpenStack Platform deployment, including identities, kerberos credentials, and access controls. You must perform the enrollment process before you proceed with the rest of the IdM integration.

The enrollment process includes the following steps:

1. Adding the undercloud node to the certificate authority (CA)
2. Adding the undercloud node to IdM
3. Optional: Setting the IdM server as the DNS server for the overcloud
4. Preparing the environment files and deploying the overcloud
5. Testing the overcloud enrollment in IdM and in RHOSP
6. Optional: Adding DNS entries for novajoin in IdM

NOTE

IdM enrollment with novajoin is currently only available for the undercloud and overcloud nodes. Novajoin integration for overcloud instances is expected to be supported in a later release.

2.2.1. Adding the undercloud node to the certificate authority

Before you deploy the overcloud, add the undercloud to the certificate authority (CA) by installing the **python3-novajoin** package on the undercloud node and running the **novajoin-ipa-setup** script.

Procedure

1. On the undercloud node, install the **python3-novajoin** package:

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

2. On the undercloud node, run the **novajoin-ipa-setup** script, and adjust the values to suit your deployment:

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

Use the resulting One-Time Password (OTP) to enroll the undercloud.

2.2.2. Adding the undercloud node to Red Hat Identity Manager (IdM)

After you add the undercloud node to the certificate authority (CA), register the undercloud with IdM and configure novajoin. Configure the following settings in the **[DEFAULT]** section of the **undercloud.conf** file.

Procedure

1. Enable the **novajoin** service:

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

2. Set a One-Time Password (OTP) so that you can register the undercloud node with IdM:

```
ipa_otp = <otp>
```

3. Set the overcloud's domain name to be served by neutron's DHCP server:

```
overcloud_domain_name = <domain>
```

4. Set the hostname for the undercloud:

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

5. Set IdM as the nameserver for the undercloud:

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

6. For larger environments, review the novajoin connection timeout values. In the **undercloud.conf** file, add a reference to a new file called **undercloud-timeout.yaml**:

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

   Add the following options to **undercloud-timeout.yaml**. You can specify the timeout value in seconds, for example, **5**:

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

Optional: If you want the local openSSL certificate authority to generate the SSL certificates for the public endpoints in director, set the **generate_service_certificate** parameter to **true**:

+

```
generate_service_certificate = true
```

1. Save the **undercloud.conf** file.

2. Run the undercloud deployment command to apply the changes to your existing undercloud:

```
$ openstack undercloud install
```

2.2.3. Setting Red Hat Identity Manager (IdM) as the DNS server for the overcloud

To enable automatic detection of your IdM environment and easier enrollment, set IdM as your DNS server. This procedure is optional but recommended.

Procedure

1. Connect to your undercloud:

```
$ source ~/stackrc
```

2. Configure the control plane subnet to use IdM as the DNS name server:

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

3. Set the **DnsServers** parameter in an environment file to use your IdM server:

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

   This parameter is usually defined in a custom **network-environment.yaml** file.

2.2.4. Preparing environment files and deploying the overcloud with novajoin enrollment

To deploy the overcloud with IdM integration, you create and edit environment files to configure the overcloud to use the custom domain parameters **CloudDomain** and **CloudName** based on the domains that you define in the overcloud. You then deploy the overcloud with all the environment files and any additional environment files that you need for the deployment.

Procedure

1. Create a copy of the **/usr/share/openstack-tripleo-heat-templates/environments/predictable-placement/custom-domain.yaml** environment file:

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

2. Edit the **/home/stack/templates/custom-domain.yaml** environment file and set the **CloudDomain** and **CloudName*** values to suit your deployment:

```
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. Choose the implementation of TLS appropriate for your environment:

   - Use the **enable-tls.yaml** environment file to protect external endpoints with your custom certificate:

a. Copy **/usr/share/openstack-tripleo-heat-templates/environments/ssl/enable-tls.yaml** to **/home/stack/templates**.

b. Modify the **/home/stack/enable-tls.yaml** environment file to include your custom certificate and key.

c. Include the following environment files in your deployment to protect internal and external endpoints:

- enable-internal-tls.yaml
- tls-every-endpoints-dns.yaml
- custom-domain.yaml
- enable-tls.yaml

```
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
```

- Use the **haproxy-public-tls-certmonger.yaml** environment file to protect external endpoints with an IdM issued certificate. For this implementation, you must create DNS entries for the VIP endpoints used by novajoin:

a. You must create DNS entries for the VIP endpoints used by novajoin. Identify the overcloud networks located in your custom **network-environment.yaml file in `/home/stack/templates**:

```
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
```

b. Create a list of virtual IP addresses for each overcloud network in a heat template, for example, **/home/stack/public_vib.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'}]
```

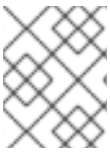
c. Add DNS entries to the IdM for each of the VIPs, and zones as needed:

```
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
```

d. Include the following environment files in your deployment to protect internal and external endpoints:

- enable-internal-tls.yaml
- tls-everywhere-endpoints-dns.yaml
- haproxy-public-tls-certmonger.yaml
- custom-domain.yaml
- public_vib.yaml

```
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 /usr/share/openstack-tripleo-heat-templates/environments/services/haproxy-public-tls-certmonger.yaml \
   -e /home/stack/templates/custom-domain.yaml \
   -e /home/stack/templates/public-vip.yaml
```



NOTE

You cannot use novajoin to implement TLS everywhere (TLS-e) on a pre-existing deployment.

Additional resources

- Section 2.3, “Implementing TLS-e with Ansible”

2.2.5. Testing overcloud enrollment in Red Hat Identity Manager (IdM)

After you complete the undercloud and overcloud enrollment in IdM with novajoin, you can test that the enrollment is successful by searching for an overcloud node in IdM and checking that the host entry includes **Keytab:True**. You can also log in to the overcloud node and confirm that the **sssd** command can query IdM users.

1. Locate an overcloud node in IdM and confirm that the host entry includes **Keytab:True**:

```
$ 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. Log in to the overcloud node and confirm that **sssd** can query IdM users. For example, to query an IdM user named **susan**:

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

2.3. IMPLEMENTING TLS-E WITH ANSIBLE

Red Hat recommends the new ansible-based **tripleo-ipa** method over the default **novajoin** method to configure your undercloud and overcloud with TLS-e. You can use the following procedure to implement TLSe on either a new installation of Red Hat OpenStack Platform, or an existing deployment you wish to configure with TLS-e. You must use this method if you deploy Red Hat OpenStack Platform with TLS-e on pre-provisioned nodes.

NOTE

If you are implementing TLS-e for an existing environment, you are still required to run commands such as the **openstack undercloud install**, and the **openstack overcloud deploy** commands. These are procedures are idempotent and will only adjust your existing deployment configuration to match updated templates and configuration files.

2.3.1. Configuring TLS-e on the undercloud

Prerequisites

Ensure that all configuration steps for the undercloud, such as the creation of the stack user, are complete. For more details, see Director Installation and Usage for more details

Procedure

1. Configure the hosts file
   Set the appropriate search domains and the nameserver on the undercloud in **/etc/resolv.conf**. For example, if the deployment domain is **example.com**, and the domain of the FreeIPA server is **bigcorp.com**, then add the following lines to /etc/resolv.conf:

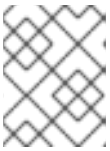
```
search example.com bigcorp.com
nameserver $IDM_SERVER_IP_ADDR
```

2. Install required software:

```
sudo dnf install -y python3-ipalib python3-ipaclient krb5-devel
```

3. Export environmental variables with values specific to your environment.:

```
export IPA_DOMAIN=bigcorp.com
export IPA_REALM=BIGCORP.COM
export IPA_ADMIN_USER=$IPA_USER
export IPA_ADMIN_PASSWORD=$IPA_PASSWORD
export IPA_SERVER_HOSTNAME=ipa.bigcorp.com
export UNDERCLOUD_FQDN=undercloud.example.com
export USER=stack
export CLOUD_DOMAIN=example.com
```



NOTE

The IdM user credentials must be an administrative user that can add new hosts and services.

4. Run the **undercloud-ipa-install.yaml** ansible playbook on the undercloud:

```
ansible-playbook \
--ssh-extra-args "-o StrictHostKeyChecking=no -o UserKnownHostsFile=/dev/null" \
/usr/share/ansible/tripleo-playbooks/undercloud-ipa-install.yaml
```

5. Add the following parameters to undercloud.conf

```
undercloud_nameservers = $IDM_SERVER_IP_ADDR
overcloud_domain_name = example.com
```

6. Deploy the undercloud:

```
openstack undercloud install
```

Verification

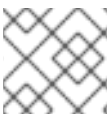
Verify that the undercloud was enrolled correctly by completing the following steps:

1. List the hosts in IdM:

```
$ kinit admin
$ ipa host-find
```

2. Confirm that **/etc/novajoin/krb5.keytab** exists on the undercloud.

```
ls /etc/novajoin/krb5.keytab
```



NOTE

The **novajoin** directory name is for legacy naming purposes only.

2.3.2. Configuring TLS-e on the overcloud

When you deploy the overcloud with TLS everywhere (TLS-e), IP addresses from the Undercloud and Overcloud will automatically be registered with IdM.

NOTE

To disable automatic IP address registration, set the **IDMModifyDNS** heat parameter to false:

```
parameter_defaults:
    ....
    IdMModifyDNS: false
```

1. Before deploying the overcloud, create a YAML file **tls-parameters.yaml** with contents similar to the following. The values you select will be specific for your environment:

   - The **DnsServers** parameter should have a value that reflects the IP address of the IdM server.
   - If the domain of the IdM server is different than the cloud domain, include it in the **DnsSearchDomains** parameter. For example: **DnsSearchDomains: ["example.com", "bigcorp.com"]**
   - If you have preprovisioned nodes, set the value of the **IDMInstallClientPackages** parameter to **true** to install required packages on overcloud nodes.
   - The shown value of the **OS::TripleO::Services::IpaClient** parameter overrides the default setting in the **enable-internal-tls.yaml** file. You must ensure the **tls-parameters.yaml** file follows **enable-internal-tls.yaml** in the **openstack overcloud deploy** command.
   - If you are running a distributed compute node (DCN) architecture with cinder configured as active-active, you must add and set the **EnableEtcdInternalTLS** parameter to **true**.

   ```
   parameter_defaults:
     DnsSearchDomains: ["example.com"]
     DnsServers: ["192.168.1.13"]
     CloudDomain: example.com
     CloudName: overcloud.example.com
     CloudNameInternal: overcloud.internalapi.example.com
     CloudNameStorage: overcloud.storage.example.com
     CloudNameStorageManagement: overcloud.storagemgmt.example.com
     CloudNameCtlplane: overcloud.ctlplane.example.com
     IdMServer: freeipa-0.redhat.local
     IdMDomain: redhat.local
     IdMInstallClientPackages: False

   resource_registry:
       OS::TripleO::Services::IpaClient: /usr/share/openstack-tripleo-heat-
   templates/deployment/ipa/ipaservices-baremetal-ansible.yaml
   ```

2. Deploy the overcloud. You will need to include the tls-parameters.yaml in the deployment command:

   ```
   DEFAULT_TEMPLATES=/usr/share/openstack-tripleo-heat-templates/
   CUSTOM_TEMPLATES=/home/stack/templates

   openstack overcloud deploy \
   -e ${DEFAULT_TEMPLATES}/environments/ssl/tls-everywhere-endpoints-dns.yaml \
   -e ${DEFAULT_TEMPLATES}/environments/services/haproxy-public-tls-certmonger.yaml \
   ```

```
-e ${DEFAULT_TEMPLATES}/environments/ssl/enable-internal-tls.yaml \
-e ${CUSTOM_TEMPLATES}/tls-parameters.yaml \
...
```

3. Confirm each endpoint is using HTTPS by querying keystone for a list of endpoints:

```
openstack overcloud endpoint list
```

2.4. ENCRYPTING MEMCACHED TRAFFIC UNDER TLS EVERYWHERE (TLS-E)

As a technology preview, you can now encrypt memcached traffic with TLS-e. This feature works with both novajoin and tripleo-ipa:

1. Create an environment file called **memcached.yaml** with the following contents to add TLS support for memcached. Replace the values under the **memcached_node_ips** parameter with the InternalAPI hostnames or IP addresses specific to your environment:

```
parameter_defaults:
  MemcachedTLS: true
  MemcachedPort: 11212
  ExtraConfig:
    memcached_port: 11212
    memcached_authtoken_port: 11211
    memcached_node_ips: "%{alias('memcached_node_names')}"
    nova::cache::backend: "dogpile.cache.pymemcache"
    heat::cache::backend: "dogpile.cache.pymemcache"
    ceilometer::cache_backend: "dogpile.cache.pymemcache"
```

2. Include the **memcached.yaml** environment file in the overcloud deployment process:

```
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 /usr/share/openstack-tripleo-heat-templates/environments/services/haproxy-public-tls-certmonger.yaml \
-e /home/stack/memcached.yaml
...
```

Additional Resources

- For more information about deploying TLSe with novajoin, see Enrolling nodes in Red Hat Identity Manager (IdM) with novajoin
- For more information about deploying TLSe with tripleo-ipa, see Implementing TLS-e with Ansible.

This feature is available in this release as a *Technology Preview*, and therefore is not fully supported by Red Hat. It should only be used for testing, and should not be deployed in a production environment. For more information about Technology Preview features, see Scope of Coverage Details.

2.5. CONFIGURING RED HAT IDENTITY MANAGER (IDM) SERVER CREDENTIALS

To configure the Red Hat Identity Manager (IdM) to integrate with OpenStack Identity, set up an LDAP account for Identity service to use, create a user group for Red Hat OpenStack users, and set up the password for the lookup account.

Prerequisites

- Red Hat Identity Manager (IdM) is configured and operational.
- Red Hat OpenStack Platform (RHOSP) is configured and operational.
- DNS name resolution is fully functional and all hosts are registered appropriately.
- IdM authentication traffic is encrypted with LDAPS, using port 636.
- Recommended: Implement IdM with a high availability or load balancing solution to avoid a single point of failure.

Procedure

Perform this procedure on the IdM server.

1. Create the LDAP lookup account to use in OpenStack Identity Service to query the IdM LDAP service:

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

NOTE

Review the password expiration settings of this account, once created.

2. Create a group for RHOSP users, called **grp-openstack**. Only members of this group can have permissions assigned in OpenStack Identity.

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

3. Set the **svc-ldap** account password and add it to the **grp-openstack** group:

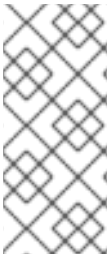
```
# ipa passwd svc-ldap
# ipa group-add-member --users=svc-ldap grp-openstack
```

4. Login as **svc-ldap** user and change the password when prompted:

```
# kinit svc-ldap
```

2.6. INSTALLING THE RED HAT IDENTITY MANAGER (IDM) LDAPS CERTIFICATE

OpenStack Identity (keystone) uses LDAPS queries to validate user accounts. To encrypt this traffic, keystone uses the certificate file defined by **keystone.conf**. To install the LDAPS certificate, copy the certificate from the Red Hat Identity Manager (IdM) server to a location where keystone will be able to reference it, and convert the certificate from **.crt** to **.pem** format.



NOTE

When using multiple domains for LDAP authentication, you might receive various errors, such as **Unable to retrieve authorized projects**, or **Peer's Certificate issuer is not recognized**. This can arise if keystone uses the incorrect certificate for a certain domain. As a workaround, merge all of the LDAPS public keys into a single **.crt** bundle, and configure all of your keystone domains to use this file.

Prerequisites

- IdM server credentials are configured.

Procedure

1. In your IdM environment, locate the LDAPS certificate. This file can be located using **/etc/openldap/ldap.conf**:

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

2. Copy the file to the Controller node that runs the keystone service. For example, the **scp** command copies the **ca.crt** file to the node **node.lab.local**:

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

3. Copy the **ca.crt** file to the certificate directory. This is the location that the keystone service will use to access the certificate:

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

4. Optional: If you need to run diagnostic commands, such as **ldapsearch**, you also need to add the certificate to the RHEL certificate store:

   a. 3. On the Controller node, convert the **.crt** to **.pem** format:

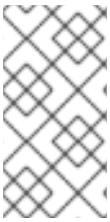
```
# openssl x509 -in ca.crt -out ca.pem -outform PEM
```

   b. Install the **.pem** on the Controller node. For example, in Red Hat Enterprise Linux:

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

2.7. CONFIGURING DIRECTOR TO USE DOMAIN-SPECIFIC LDAP BACKENDS

To configure director to use one or more LDAP backends, set the **KeystoneLDAPDomainEnable** flag to **true** in your heat templates, and set up environment files with the information about each LDAP backend. Director then uses a separate LDAP backend for each keystone domain.



NOTE

The default directory for domain configuration files is set to **/etc/keystone/domains/**. You can override this by setting the required path with the **keystone::domain_config_directory** hiera key and adding it as an **ExtraConfig** parameter within an environment file.

Procedure

1. In the heat template for your deployment, set the **KeystoneLDAPDomainEnable** flag to **true**. This configures the **domain_specific_drivers_enabled** option in keystone within the **identity** configuration group.

2. Add a specification of the LDAP backend configuration by setting the **KeystoneLDAPBackendConfigs** parameter in **tripleo-heat-templates**, where you can then specify your required LDAP options.

3. Create a copy of the **keystone_domain_specific_ldap_backend.yaml** environment file:

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

4. Edit the **/home/stack/templates/keystone_domain_specific_ldap_backend.yaml** environment file and set the values to suit your deployment. For example, this parameter create a LDAP configuration for a keystone domain named **testdomain**:

```
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
```

NOTE

The **keystone_domain_specific_ldap_backend.yaml** environment file contains the following deprecated write parameters:

- **user_allow_create**
- **user_allow_update**
- **user_allow_delete**

The values for these parameters have no effect on the deployment, and can be safely removed.

5. Optional: Add more domains to the environment file. For example:

```
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
    ...
```

This results in two domains named **domain1** and **domain2**; each will have a different LDAP domain with its own configuration.

2.8. CONFIGURING OPENSTACK IDENTITY DOMAINS ON CONTROLLER NODES

To configure Controller nodes that run OpenStack Identity (keystone) to integrate with an external user management service, first configure SELinux to use LDAP authentication and create the **domains** directory on the Controller node. Then configure OpenStack Identity to use multiple back ends and the Dashboard to use multiple domains.

NOTE

If you are using director, note that the configuration files referenced in this procedure are managed by Puppet. Therefore, any custom configuration you add might be overwritten whenever you run the **openstack overcloud deploy** command.

Planning

If you intend to update any configuration files, you need to be aware that certain OpenStack services now run within containers; this applies to keystone, nova, and cinder, among others. As a result, there are certain administration practices to consider:

- Do not update any configuration file you might find on the physical node's host operating system, for example, **/etc/cinder/cinder.conf**. This is because the containerized service does not reference this file.

- Do not update the configuration file running within the container. This is because any changes are lost once you restart the container.
  Instead, if you need to add any changes to containerized services, you will need to update the configuration file that is used to generate the container. These are stored within **/var/lib/config-data/puppet-generated/**

  For example:

  - 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**

  Any changes will then be applied once you restart the service. For example: **sudo systemctl restart tripleo_keystone**

Procedure

Perform this procedure on each Controller node that runs the OpenStack Identity (keystone) service.

1. Configure SELinux:

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

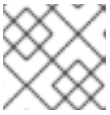
   The output might include messages similar to this. They can be ignored:

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

2. Create the **domains** directory:

```
# 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. Configure keystone to use multiple back ends:



NOTE

You might need to install **crudini** using **dnf install crudini**.

```
# 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
```

4. Enable multiple domains in dashboard. Add these lines to **/var/lib/config-data/puppet-generated/horizon/etc/openstack-dashboard/local_settings**:

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

NOTE

If you are using director, note that **/var/lib/config-data/puppet-generated/horizon/etc/openstack-dashboard/local_settings** is managed by Puppet. Consequently, any custom configuration you add might be overwritten whenever you run the **openstack overcloud deploy** process. As a result, you might need to re-add this configuration manually each time.

   Restart the horizon container to apply the settings:

```
$ sudo systemctl restart tripleo_horizon
```

5. Create the keystone domain for the external service integration with the NetBIOS name value

retrieved previously as the domain name. This approach allows you to present a consistent domain name to users during the login process. In this example, **LAB** is the NetBIOS name to use as the Identity Service domain.

```
$ openstack domain create LAB
```

NOTE

If this command is not available, check that you have enabled keystone v3 for your command line session by running **# source overcloudrc-v3**.

6. Create the configuration file for the external service that you are integrating:

   - Active Directory Domain Service (AD DS): Enter the LDAP settings in a new file called **/var/lib/config-data/puppet-generated/keystone/etc/keystone/domains/keystone.LAB.conf** (where **LAB** is the NetBIOS name retrieved previously). You will need to edit the sample settings below to suit your AD DS deployment:

```
[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
```

     Explanation of each setting:

Setting	Description
url	The AD Domain Controller to use for authentication. Uses LDAPS port **636**.
user	The *Distinguished Name* of an AD account to use for LDAP queries. For example, you can locate the *Distinguished Name* value of the *svc-ldap* account in AD using **Get-ADuser svc-ldap | select DistinguishedName**
password	The plaintext password of the AD account used above.
suffix	The *Distinguished Name* of your AD domain. You can locate this value using **Get-ADDomain | select DistinguishedName**
user_tree_dn	The *Organizational Unit* (OU) that contains the OpenStack accounts.
user_objectclass	Defines the type of LDAP user. For AD, use the **person** type.
user_filter	Filters the users presented to Identity Service. As a result, only members of the **grp-openstack** group can have permissions defined in Identity Service. This value requires the full *Distinguished Name* of the group: **Get-ADGroup grp-openstack | select DistinguishedName**
user_id_attribute	Maps the AD value to use for user IDs.
user_name_attribute	Maps the AD value to use for *names*.
user_mail_attribute	Maps the AD value to use for user email addresses.
user_pass_attribute	Leave this value blank.
user_enabled_attribute	The AD setting that validates whether the account is enabled.
user_enabled_mask	Defines the value to check to determine whether an account is enabled. Used when booleans are not returned.
user_enabled_default	The AD value that indicates that an account is enabled.

Setting	Description
user_attribute_ignore	Defines user attributes that Identity Service should disregard.
group_objectclass	Maps the AD value to use for *groups*.
group_tree_dn	The *Organizational Unit* (OU) that contains the user groups.
group_filter	Filters the groups presented to Identity Service.
group_id_attribute	Maps the AD value to use for group IDs.
group_name_attribute	Maps the AD value to use for group names.
use_tls	Defines whether TLS is to be used. This needs to be disabled if you are encrypting with LDAPS rather than STARTTLS.
tls_cacertfile	Specifies the path to the *.crt* certificate file.
query_scope	Configures Identity Service to also search within nested child OUs, when locating users that are members of the **grp-openstack** group.
chase_referrals	Set to **false**, this setting prevents**python-ldap** from chasing all referrals with anonymous access.

- Red Hat Identity Manager (IdM): Enter the LDAP settings in a new file called **/var/lib/config-data/puppet-generated/keystone/etc/keystone/domains/keystone.LAB.conf** (where **LAB** is the domain name created previously). You will need to edit the sample settings below to suit your IdM deployment:

```
[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
```

Explanation of each setting:

Setting	Description
url	The IdM server to use for authentication. Uses LDAPS port **636**.
user	The account in IdM to use for LDAP queries.
password	The plaintext password of the IdM account used above.
user_filter	Filters the users presented to Identity Service. As a result, only members of the **grp-openstack** group can have permissions defined in Identity Service.
user_tree_dn	The path to the OpenStack accounts in IdM.
user_objectclass	Defines the type of LDAP user. For IdM, use the **inetUser** type.
user_id_attribute	Maps the IdM value to use for user IDs.
user_name_attribute	Maps the IdM value to use for *names*.
user_mail_attribute	Maps the IdM value to use for user email addresses.
user_pass_attribute	Leave this value blank.

NOTE

Integration with an IdM group will only return direct members, and not nested groups. As a result, queries that rely on **LDAP_MATCHING_RULE_IN_CHAIN** or **memberof:1.2.840.113556.1.4.1941:** will not currently work with IdM.

7. Change ownership of the configuration file to the keystone user:

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

8. Restart the keystone service to apply the changes:

```
# sudo systemctl restart tripleo_keystone
```

Additional resources

- Section 2.9, "Granting the admin user access to the OpenStack Identity domain"
- Section 2.7, "Configuring director to use domain-specific LDAP backends"

2.9. GRANTING THE ADMIN USER ACCESS TO THE OPENSTACK IDENTITY DOMAIN

To allow the **admin** user to access the OpenStack Identity (keystone) domain and see the **Domain** tab, get the ID of the domain and the **admin** user, and then assign the **admin** role to the user in the domain.

NOTE

This does not grant the OpenStack admin account any permissions on the external service domain. In this case, the term *domain* refers to OpenStack's usage of the keystone domain.

Procedure

This procedure uses the **LAB** domain. Replace the domain name with the actual name of the domain that you are configuring.

1. Get the ID of the **LAB** domain:

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

2. Get the ID of the **admin** user from the **default** domain:

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

3. Get the ID of the **admin** role:

```
$ openstack role list
```

The output depends on the external service you are integrating with:

- Active Directory Domain Service (AD DS):

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

- Red Hat Identity Manager (IdM):

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

4. Use the domain and admin IDs to construct the command that adds the **admin** user to the **admin** role of the keystone **LAB** domain:

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

2.10. GRANTING EXTERNAL GROUPS ACCESS TO RED HAT OPENSTACK PLATFORM PROJECTS

To grant multiple authenticated users access to Red Hat OpenStack Platform (RHOSP) resources, you can authorize certain groups from the external user management service to grant access to RHOSP projects, instead of requiring OpenStack administrators to manually allocate each user to a role in a project. As a result, all members of these groups can access pre-determined projects.

Prerequisites

- Ensure that the external service administrator completed the following steps:
  - Creating a group named **grp-openstack-admin**.
  - Creating a group named **grp-openstack-demo**.
  - Adding your RHOSP users to one of these groups as needed.
  - Adding your users to the **grp-openstack** group.
- Create the OpenStack Identity domain. This procedure uses the **LAB** domain.

- Create or choose a RHOSP project. This procedure uses a project called **demo** that was created with the **openstack project create --domain default --description "Demo Project" demo** command.

Procedure

1. Retrieve a list of user groups from the OpenStack Identity domain:

```
# openstack group list --domain LAB
```

The command output depends on the external user management service that you are integrating with:

- Active Directory Domain Service (AD DS):

```
+------------------------------------------------------------------+--------------------+
| ID                                                               | Name               |
+------------------------------------------------------------------+--------------------+
| 185277be62ae17e498a69f98a59b66934fb1d6b7f745f14f5f68953a665b8851 | grp-openstack      |
| a8d17f19f464c4548c18b97e4aa331820f9d3be52654aa8094e698a9182cbb88 | grp-openstack-admin |
| d971bb3bd5e64a454cbd0cc7af4c0773e78d61b5f81321809f8323216938cae8 | grp-openstack-demo  |
+------------------------------------------------------------------+--------------------+
```

- Red Hat Identity Manager (IdM):

```
+------------------------------------------------------------------+--------------------+
| ID                                                               | Name               |
+------------------------------------------------------------------+--------------------+
| 185277be62ae17e498a69f98a59b66934fb1d6b7f745f14f5f68953a665b8851 | grp-openstack      |
| a8d17f19f464c4548c18b97e4aa331820f9d3be52654aa8094e698a9182cbb88 | grp-openstack-admin |
| d971bb3bd5e64a454cbd0cc7af4c0773e78d61b5f81321809f8323216938cae8 | grp-openstack-demo  |
+------------------------------------------------------------------+--------------------+
```

2. Retrieve a list of roles:

```
# openstack role list
```

The command output depends on the external user management service that you are integrating with:

- Active Directory Domain Service (AD DS):

```
+----------------------------------+----------------+
| ID                               | Name           |
+----------------------------------+----------------+
| 01d92614cd224a589bdf3b171afc5488 | admin          |
| 034e4620ed3d45969dfe8992af001514 | member         |
| 0aa377a807df4149b0a8c69b9560b106 | ResellerAdmin  |
| 9369f2bf754443f199c6d6b96479b1fa | heat_stack_user |
```

```
| cfea5760d9c948e7b362abc1d06e557f | reader          |
| d5cb454559e44b47aaa8821df4e11af1 | swiftoperator   |
| ef3d3f510a474d6c860b4098ad658a29 | service         |
+----------------------------------+-----------------+
```

- Red Hat Identity Manager (IdM):

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

3. Grant the user groups access to RHOSP projects by adding them to one or more of these roles. For example, if you want users in the **grp-openstack-demo** group to be general users of the **demo** project, you must add the group to the **member** or **_member_** role, depending on the external service that you are integrating with:

   - Active Directory Domain Service (AD DS):

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

   - Red Hat Identity Manager (IdM):

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

Result

Members of **grp-openstack-demo** can log in to the dashboard by entering their username and password and entering **LAB** in the **Domain** field:

NOTE

If users receive the error **Error: Unable to retrieve container list.**, and expect to be able to manage containers, then they must be added to the **SwiftOperator** role.

Additional resources

- Section 2.11, "Granting external users access to Red Hat OpenStack Platform projects"

2.11. GRANTING EXTERNAL USERS ACCESS TO RED HAT OPENSTACK PLATFORM PROJECTS

To grant specific authenticated users from the **grp-openstack** group access to OpenStack resources, you can grant these users direct access to Red Hat OpenStack Platform (RHOSP) projects. Use this process in cases where you want to grant access to individual users instead of granting access to groups.

Prerequisites

- Ensure that the external service administrator completed the following steps:
  - Adding your RHOSP users to the **grp-openstack** group.
  - Creating the OpenStack Identity domain. This procedure uses the **LAB** domain.
- Create or choose a RHOSP project. This procedure uses a project called **demo** that was created with the **openstack project create --domain default --description "Demo Project" demo** command.

Procedure

1. Retrieve a list of users from the OpenStack Identity domain:

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

2. Retrieve a list of roles:

```
# openstack role list
```

The command output depends on the external user management service that you are integrating with:

- Active Directory Domain Service (AD DS):

```
+----------------------------------+----------------+
| ID                               | Name           |
```

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

- Red Hat Identity Manager (IdM):

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

3. Grant users access to RHOSP projects by adding them to one or more of these roles. For example, if you want **user1** to be a general user of the **demo** project, you add them to the **member** or **_member_** role, depending on the external service that you are integrating with:

   - Active Directory Domain Service (AD DS):

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

   - Red Hat Identity Manager (IdM):

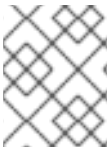
   ```
   # openstack role add --project demo --user
   1f24ec1f11aeb90520079c29f70afa060d22e2ce92b2eba7784c841ac418091e _member_
   ```

4. If you want **user1** to be an administrative user of the **demo** project, add the user to the **admin** role:

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

Result

The **user1** user is able to log in to the dashboard by entering their external username and password and entering **LAB** in the **Domain** field:



NOTE

If users receive the error **Error: Unable to retrieve container list.**, and expect to be able to manage containers, then they must be added to the **SwiftOperator** role.

Additional resources

- Section 2.10, "Granting external groups access to Red Hat OpenStack Platform projects"

2.12. VIEWING THE LIST OF OPENSTACK IDENTITY DOMAINS AND USERS

Use the **openstack domain list** command to list the available entries. Configuring multiple domains in Identity Service enables a new **Domain** field in the dashboard login page. Users are expected to enter the domain that matches their login credentials.

IMPORTANT

After you complete the integration, you need to decide whether to create new projects in the **Default** domain or in newly created keystone domains. You must consider your workflow and how you administer user accounts. If possible, use the **Default** domain as an internal domain to manage service accounts and the **admin** project, and keep your external users in a separate domain.

In this example, external accounts need to specify the **LAB** domain. The built-in keystone accounts, such as **admin**, must specify **Default** as their domain.

Procedure

1. Show the list of domains:

```
# openstack domain list
+----------------------------------+---------+---------+----------------------------------------------------------
--------------+
| ID                               | Name    | Enabled | Description                                              |
```

```
+----------------------------------+---------+---------+-----------------------------------------------------
--------------+
| 6800b0496429431ab1c4efbb3fe810d4 | LAB     | True    |
|
| default                          | Default | True    | Owns users and projects available on Identity API
v2. |
+----------------------------------+---------+---------+-----------------------------------------------------
--------------+
```

2. Show the list of users in a specific domain. This command example specifies the **--domain LAB** and returns users in the LAB domain that are members of the **grp-openstack** group:

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

You can also append **--domain Default** to show the built-in keystone accounts:

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

2.13. CREATING A CREDENTIALS FILE FOR A NON-ADMIN USER

After you configure users and domains for OpenStack Identity, you might need to create a credentials file for a non-admin user.

Procedure

- Create a credentials (RC) file for a non-admin user. This example uses the **user1** user in the file.

```
$ 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
```

2.14. TESTING OPENSTACK IDENTITY INTEGRATION WITH AN EXTERNAL USER MANAGEMENT SERVICE

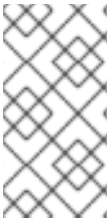
To test that OpenStack Identity (keystone) successfully integrated with Active Directory Domain Service (AD DS), test user access to dashboard features.

Prerequisites

- Integration with an external user management service, such as Active Directory (AD) or Red Hat Identity Manager (IdM)

Procedure

1. Create a test user in the external user management service, and add the user to the **grp-openstack** group.
2. In Red Hat OpenStack Platform, add the user to the **_member_** role of the **demo** project.
3. Log in to the dashboard with the credentials of the AD test user.
4. Click on each of the tabs to confirm that they are presented successfully without error messages.
5. Use the dashboard to build a test instance.



NOTE

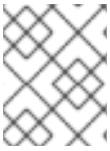
If you experience issues with these steps, log in to the dashboard with the **admin** account and perform the subsequent steps as that user. If the test is successful, it means that OpenStack is still working as expected and that an issue exists somewhere in the integration settings between OpenStack Identity and Active Directory.

Additional resources

- Section 1.11, “Troubleshooting Active Directory integration”

2.15. TROUBLESHOOTING RED HAT IDENTITY MANAGER (IDM) INTEGRATION

If you encounter errors when using the Red Hat Identity Manager (IdM) integration with OpenStack Identity, you might need to test the LDAP connection or test the certificate trust configuration. You might also need to check that the LDAPS port is accessible.



NOTE

Depending on the error type and location, perform only the relevant steps in this procedure.

Procedure

1. Test the LDAP connection by using the **ldapsearch** command to remotely perform test queries against the IdM server. A successful result here indicates that network connectivity is working, and the IdM services are up. In this example, a test query is performed against the server **idm.lab.local** on port **636**:

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

NOTE

ldapsearch is a part of the **openldap-clients** package. You can install this using **# dnf install openldap-clients**.

2. Use the **nc** command to check that LDAPS port **636** is remotely accessible. In this example, a probe is performed against the server **idm.lab.local**. Press **ctrl-c** to exit the prompt.

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

Failure to establish a connection could indicate a firewall configuration issue.