Red Hat OpenStack Platform 10

Deploy Fernet on the Overcloud

Deploy Fernet on the Red Hat OpenStack Platform director overcloud

Last Updated: 2019-04-29

Red Hat OpenStack Platform 10 Deploy Fernet on the Overcloud

Deploy Fernet on the Red Hat OpenStack Platform director overcloud

OpenStack Team
rhos-docs@redhat.com

Legal Notice

Abstract

Deploy Fernet on the Red Hat OpenStack Platform director overcloud.

Table of Contents

CHAPTER 1. DEPLOY FERNET ON THE OVERCLOUD

This chapter describes how to configure your Overcloud to use the Fernet token provider.

- **Key Management** - This example uses *keystone-manage* to generate the overcloud Fernet keys on the undercloud. These keys will not actually be used by the Undercloud since it is configured to use the UUID token format by default. If you do configure the undercloud to use the Fernet token format after following the procedure in this document, it will use the same keys as the overcloud (which may not be desirable).
- **Swift Artifacts** - This implementation uses Heat swift artifacts, which puts a copy of the Fernet key directory on every node in your deployment (not just the Controller node). You will need to consider whether this outcome is acceptable for your deployment requirements.

1.1. PREPARE THE FERNET KEYS

This section generates the Fernet keys on the undercloud, and uploads them into swift.

1. On the undercloud node, use **`keystone_manage`** to generate Fernet keys:

```
$ . ~/stackrc
$ sudo keystone-manage fernet_setup --keystone-user keystone --keystone-
group keystone
```

2. Create a tar file containing the Fernet keys:

```
$ sudo tar -zcf keystone-fernet-keys.tar.gz -P /etc/keystone/fernet-keys
```

NOTE

the keys in the controller nodes should not be changed manually. All controller nodes should have the exact same set of Fernet keys, otherwise a token generated by one controller won't be accepted by the others.

3. Upload the Fernet keys as swift artifacts:

```
$ upload-swift-artifacts -f keystone-fernet-keys.tar.gz
```

1.2. CONFIGURE THE OVERCLOUD TO USE FERNET

This section creates a YAML file that configures keystone to use **`fernet`** as the token provider. This setting is then applied to your existing overcloud in a later step.

1. Create a file named **`fernet.yaml`** that contains the required **`token_provider`** setting:

```
parameter_defaults:
  controllerExtraConfig:
    keystone::token_provider: 'fernet'
```

2. Deploy the overcloud, including the *fernet.yaml* file that was created in the previous step. For example:

```
source /home/stack/stackrc
openstack overcloud deploy --templates -e /home/stack/fernet.yaml
```

NOTE

If re-deploying the overcloud in the future, you will need to ensure that you still include *fernet.yaml*, to prevent the token provider from being re-configured to use a different format.

The process may take some time to complete.

1.3. REVIEW THE FERNET DEPLOYMENT

Review the overcloud controller configuration to confirm that the process was successful:

1. Retrieve the IP address of the controller node:

```
$ openstack server list
+--------------------------------------+-------------------------+----
----+---------------------+
| ID                                   | Name                    |
Status | Networks            |
+--------------------------------------+-------------------------+----
----+---------------------+
| 756fbd73-e47b-46e6-959c-e24d7fb71328 | overcloud-controller-0  | ACTIVE
| ctlplane=192.0.2.16 |
| 62b869df-1203-4d58-8e45-fac6cd4cfbee | overcloud-novacompute-0 | ACTIVE
| ctlplane=192.0.2.8  |
+--------------------------------------+-------------------------+----
----+---------------------+
```

2. SSH to the controller:

```
$ ssh heat-admin@192.0.2.16
Last login: Tue Sep  6 00:09:59 2016 from 192.0.2.1
```

3. Retrieve the values of the token driver and provider settings:

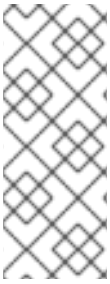
```
$ sudo crudini --get /etc/keystone/keystone.conf token driver
sql
$ sudo crudini --get /etc/keystone/keystone.conf token provider
fernet
```

4. Test the Fernet provider:

```
$ openstack token issue
WARNING: openstackclient.common.utils is deprecated and will be removed
after Jun 2017. Please use osc_lib.utils
+------------+----------------------------------------------------------
------------------------------------------------------------------------
-----------------------------------------------------------+
| Field | Value |
+------------+----------------------------------------------------------
```

```
----------------------------------------------------------------------
----------------------------------------------------------+
| expires | 2016-09-20 05:26:17+00:00 |
| id | gAAAAABX4LppE8vaiFZ992eah2i3edpO1aDFxlKZq6a_RJzxUx56QVKORrmW0-oZK3-
Xuu2wcnpYq_eek2SGLz250eLpZOzxKBR0GsoMfxJU8mEFF8NzfLNcbuS-iz7SV-
N1re3XEywSDG90JcgwjQfXW-8jtCm-n3LL5IaZexAYIw059T_-cd8 |
| project_id | 26156621d0d54fc39bf3adb98e63b63d |
| user_id | 397daf32cadd490a8f3ac23a626ac06c |
+------------+-----------------------------------------------------
----------------------------------------------------------------------
----------------------------------------------------------+
```

The result should include the long Fernet token. This token will still be shorter in length than the PKI token.



NOTE

The keys used to sign tokens are now available in the undercloud's swift. The keys should remain in swift in case you need to deploy a new controller, however, you can delete them using the **`swift`** command, if needed:

```
swift delete overcloud-artifacts keystone-fernet-keys.tar.gz
```

CHAPTER 2. ROTATE THE FERNET KEYS

It is recommended that you rotate your Fernet keys regularly, as a compromised keystone key can allow an attacker to generate their own tokens and subsequently grant themselves access to a project. During the key rotation process, the primary key is relegated to secondary key status, and a new primary key is issued, thereby reducing the value of a compromised primary key. Secondary keys can only be used to decrypt tokens that were created with previous primary keys, and cannot issue new ones.

For more information on Fernet keys, see https://access.redhat.com/documentation/en-us/red_hat_openstack_platform/14/html-single/security_and_hardening_guide/index#fernet_tokens.

Fernet uses three types of keys, which are stored in **`/etc/keystone/fernet-keys`**. The highest-numbered file contains the primary key, which is used to generate new tokens and decrypt existing fernet tokens.

- **`0`** - Contains the staged key, and will always be numbered **`0`**. This key will be promoted to a primary key during the next rotation.
- **`1`** and **`2`** - Contain the secondary keys.
- **`3`** - Contains the primary key. This number will increment each time the keys are rotated, with the highest number always serving as the primary key.

2.1. SET THE MAXIMUM NUMBER OF FERNET TOKENS

The maximum number of active keys can be determined with the following calculation: **`fernet-keys = token-validity(hours) / rotation-time(hours) + 2`**. For example, to enable 24 hours token-validity, rotation every 12 hours will resemble: **`24/12 + 2 = 4`**.

You can configure director to manage the overcloud's Fernet keys by adding a **`fernet_max_active_keys`** setting to the **`controllerExtraConfig`** section of your environment file:

1. To set the maximum active keys for Fernet, add the following to your environment file:

```
parameter_defaults:
(...)
  controllerExtraConfig:
    keystone::fernet_max_active_keys: 4
```

NOTE

This approach has been replaced by a Mistral workflow in Red Hat OpenStack Platform 12 and higher.

2.2. ROTATE THE KEYS

This procedure creates a script that will rotate the Fernet tokens in your deployment, then copy them to your other controller nodes.

1. Review the existing Fernet keys:

```
[heat-admin@overcloud-controller-0 ~]$ sudo ls /etc/keystone/fernet-
keys
0  1  2
```

-

- **0** - Contains the *staged* key, (which becomes the next primary key) and will always be numbered **0**.
- **1** - Contains the *secondary* key.
- **2** - Contains the *primary* key. This number will increment each time the keys are rotated, with the highest number always serving as the primary key.

NOTE

- The maximum number of keys is determined by the **fernet_max_active_keys** property, using **4** by default.
- The keys are propagated across all controllers.

2. Create a script on the undercloud node:

```
$ sudo vi /usr/local/bin/rotate-fernet-tokens.sh
```

3. Add the following contents to the script:

```
#!/bin/bash

source /home/stack/stackrc

controller_name="overcloud-controller-"

tmp_dir=/tmp/fernet_keys

controller0_ip=`openstack server show controller-0 -f value -c
addresses|sed s/ctlplane=//g`
controller1_ip=`openstack server show controller-1 -f value -c
addresses|sed s/ctlplane=//g`
controller2_ip=`openstack server show controller-2 -f value -c
addresses|sed s/ctlplane=//g`

SSH="ssh -o UserKnownHostsFile=/dev/null -o
StrictHostKeyChecking=no"
SCP="scp -o UserKnownHostsFile=/dev/null -o
StrictHostKeyChecking=no"

mkdir -p $tmp_dir
cd $tmp_dir

# rotate fernet tokens on controller-0, tar them up and copy them to
the local node
$SSH heat-admin@${controller0_ip} "sudo keystone-manage
fernet_rotate --keystone-user keystone --keystone-group keystone"
$SSH heat-admin@${controller0_ip} "sudo rm -f /tmp/fernet-keys.tar ;
sudo find /etc/keystone/fernet-keys -maxdepth 1 -type f -execdir sudo
tar -rf /tmp/fernet-keys.tar {} \; ; sudo chown heat-admin.
/tmp/fernet-keys.tar && sudo tar -tf /tmp/fernet-keys.tar"
$SCP heat-admin@${controller0_ip}:/tmp/fernet-keys.tar .
```

```
# copy fernet tokens and display rotated keys on controller-1
$SCP fernet-keys.tar heat-admin@${controller1_ip}:
$SSH heat-admin@${controller1_ip} "sudo find /etc/keystone/fernet-
keys -maxdepth 1 -type f | sudo xargs -I {} rm -f {} ; sudo tar -xf
/home/heat-admin/fernet-keys.tar -C /etc/keystone/fernet-keys ; sudo
chown keystone. /etc/keystone/fernet-keys -R ; sudo ls -al
/etc/keystone/fernet-keys"


# copy fernet tokens and display rotated keys on controller-2
$SCP fernet-keys.tar heat-admin@${controller2_ip}:
$SSH heat-admin@${controller2_ip} "sudo find /etc/keystone/fernet-
keys -maxdepth 1 -type f | sudo xargs -I {} rm -f {} ; sudo tar -xf
/home/heat-admin/fernet-keys.tar -C /etc/keystone/fernet-keys ; sudo
chown keystone. /etc/keystone/fernet-keys -R ; sudo ls -al
/etc/keystone/fernet-keys"


# reload httpd
$SSH heat-admin@${controller0_ip} sudo systemctl reload httpd
$SSH heat-admin@${controller1_ip} sudo systemctl reload httpd
$SSH heat-admin@${controller2_ip} sudo systemctl reload httpd
```

NOTE

This script will restart the **`http`** service on your controllers. This will result in an outage to the horizon and keystone services while the service restarts.

NOTE

This script assumes you have three controller nodes. You will need to update this script if your deployment uses a different number of controllers.

4. Make the script executable:

```
[stack@director ~]$ sudo chmod +x /usr/local/bin/rotate-fernet-
tokens.sh
```

5. Rotate the Fernet keys:

```
$ bash /usr/local/bin/rotate-fernet-tokens.sh
```

6. On a keystone node, review the number of Fernet keys and compare with the previous result:

```
[heat-admin@overcloud-controller-0 ~]$ sudo ls /var/lib/config-
data/puppet-generated/keystone/etc/keystone/fernet-keys
0  1  2  3
```

- **`0`** - Contains the staged key, and will always be numbered **`0`**. This key will be promoted to a primary key during the next rotation.
- **`1`** and **`2`** - Contain the secondary keys.

- **3** - Contains the primary key. This number will increment each time the keys are rotated, with the highest number always serving as the primary key.

7. Consider configuring a cronjob to regularly execute the script from the undercloud. This example runs the script every 12 hours:

```
$ echo "* * * * * stack bash /usr/local/bin/rotate-fernet-tokens.sh"
| sudo tee /etc/cron.d/1rotate-fernet-tokens
```