Red Hat OpenStack Platform 10

DNS-as-a-Service Guide

Integrate DNS Management with Red Hat OpenStack Platform

Last Updated: 2018-05-03

Red Hat OpenStack Platform 10 DNS-as-a-Service Guide

Integrate DNS Management with Red Hat OpenStack Platform

OpenStack Team
rhos-docs@redhat.com

Legal Notice

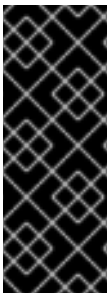
Abstract

A guide for integrating DNS with Red Hat OpenStack Platform.

Table of Contents

CHAPTER 1. OVERVIEW OF DNSAAS

Red Hat OpenStack Platform includes a Technology Preview of DNS-as-a-Service (DNSaaS), also known as Designate. DNSaaS includes a REST API for domain and record management, is multi-tenanted, and integrates with OpenStack Identity Service (*keystone*) for authentication. DNSaaS includes a framework for integration with Compute (*nova*) and OpenStack Networking (*neutron*) notifications, allowing auto-generated DNS records. In addition, DNSaaS includes integration support for Bind9.



IMPORTANT

DNS-as-a-Service (DNSaaS), also known as Designate, is available in this release as a *Technology Preview*, and therefore is not fully supported by Red Hat. If you are interested in running DNSaaS in your production environment, please file a support ticket and mention the bug tracker **BZ#1374002**, so we can gauge the interest for this tool. For more information about Technology Preview features, see https://access.redhat.com/support/offerings/techpreview.

1.1. TOPICS COVERED IN THIS CHAPTER

- Manual DNSaaS installation steps, as DNSaaS is not currently included in Director deployment.
- Managing and configuring DNSaaS from the command line interface.
- Integration with Bind9, including API usage and auto-creation of instance records.

1.1.1. DNSaaS prerequisites

- A fully functioning non-HA OpenStack environment.

1.1.2. DNSaaS services

A deployment of DNSaaS includes the following components:

designate-api	Provides an OpenStack-native REST API.
designate-central	Handles requests and coordinates storage in the mysql database.
designate-mdns	A small MiniDNS server used only to communicate with other DNS servers over standard DNS protocol.
designate-pool-manager	Manages the states of the DNS servers that DNSaaS manages. Ensures the backend DNS servers are in sync with DNSaaS.
designate-sink	An optional service that is used to listen to nova and neutron notification events to trigger automatic record creation/deletion.

designate-agent	Used for DNS servers that cannot accept zone transfers (*AXFR*). Not needed for BIND backends.

1.1.3. DNSaaS integration with Compute and OpenStack Networking

DNSaaS record management begins when the **`designate-sink`** service sends a message to **`designate-central`**, which then triggers the workflow described below:

1. **`designate-sink`** receives an *instance boot/delete* event from Compute, or a *floating IP add/remove* event from OpenStack Networking. These events are sent using the OpenStack message bus.
2. **`designate-sink`** constructs the FQDN of the host from the VM name and the configured domain ID (see below).
3. **`designate-sink`** tells **`designate-central`** to add/delete the record with the given name and IP address.
4. **`designate-central`** adds/deletes the record in the DNSaaS database (shared between **`designate-central`** and **`designate-mdns`**).
5. **`designate-central`** tells **`designate-pool-manager`** to send a **`DNS NOTIFY`** to the backend DNS server (BIND9) for this domain.
6. The backend DNS servers receive the **`DNS NOTIFY`** and send an **`AXFR`** (zone transfer) request to **`designate-mdns`**.
7. **`designate-mdns`** reads the changes from the database and sends them to the backend DNS servers in the **`AXFR`** response.

1.2. MANUAL DNSAAS INSTALLATION

NOTE

Your server must be registered to receive the OpenStack packages. For more information, see https://access.redhat.com/documentation/en-us/red_hat_openstack_platform/10/html-single/director_installation_and_usage/#sect-Registering_your_System

1. Install the DNSaaS and BIND packages on the *controller* node. **NOTE**: You can also an external BIND service; you will need change the variables below accordingly.

```
yum install openstack-designate-api openstack-designate-central
openstack-designate-sink openstack-designate-pool-manager openstack-
designate-mdns openstack-designate-common python-designate python-
designateclient openstack-designate-agent openstack-utils bind bind-
utils
```

2. Configure DNS. It is important to define a specific boolean, otherwise you will get AVCs / Access denied in Designate when creating new zones:

```
setsebool named_write_master_zones 1
```

3. Configure ISC BIND to listen in all IP addresses:

```
sed -i -e "s/listen-on port.*/listen-on port 53 { any; };/"
/etc/named.conf
```

4. Configure **rndc** to bind in all IP addresses, accepts only **rndc-key** key:

```
sed -i '/^options.*/i include "/etc/rndc.key"; controls {      inet
* allow { any; } keys { "rndc-key"; }; };' /etc/named.conf
```

5. Allow queries for local DNS server from all IP addresses:

```
sed -i '/allow-query.*/d' /etc/named.conf
```

6. Configure DNS server to permit new zone creation via **rndc**:

```
sed -i '/^options.*/a      allow-new-zones yes;      allow-query {
any; };' /etc/named.conf
```

7. Create **rndc** initial configuration:

```
rndc-confgen -a
```

8. Permit group **named** to write in **/var/named**:

```
chmod g+w /var/named
```

9. Fix **rndc** key permissions:

```
chgrp named /etc/rndc.key
chmod g+r /etc/rndc.key
```

10. And finally, start the DNS service:

```
systemctl enable named
systemctl start named
```

11. Source your **openstackrc** file, as the following steps interact with OpenStack services.

12. To ease the deployment process, this guide relies on a number of variables; you will need to populate the values accordingly:

```
CONTROLLER_IP_ADDRESS=192.168.2.1
ZONE_NAME=testzone.example.com
INTERNAL_NET_NAME=net_internal
INSTANCES_PROJECT_NAME=myinstancesproject
SERVICES_PROJECT_NAME=service
DESIGNATE_PASSWORD=SecureDesignatePassword

EXTERNAL_DNS_SERVER_IP=$CONTROLLER_IP_ADDRESS
EXTERNAL_DNS_SERVER_FQDN=`hostname`
DESIGNATE_VIP_IP=$CONTROLLER_IP_ADDRESS
```

```
RABBIT_SERVER_IP=$CONTROLLER_IP_ADDRESS
REDIS_SERVER_IP=$CONTROLLER_IP_ADDRESS
MYSQL_SERVER_IP=$CONTROLLER_IP_ADDRESS
KEYSTONE_SERVER_IP=$CONTROLLER_IP_ADDRESS
DESIGNATE_SERVER_1=$CONTROLLER_IP_ADDRESS
```

13. The following variables will also populate the required IDs that will be used during the install process.

```
SERVICES_TENANT_ID=`openstack project show $SERVICES_PROJECT_NAME -f
value -c id`
INSTANCES_TENANT_ID=`openstack project show $INSTANCES_PROJECT_NAME
-f value -c id`
DEFAULT_NAMESERVER_ID=$(uuidgen)
DEFAULT_TARGET_ID=$(uuidgen)
INTERNAL_NET_ID=`openstack network show $INTERNAL_NET_NAME -f value
-c id`
```

14. Create the backend database:

```
mysql -u root << EOF
CREATE DATABASE designate;
GRANT ALL ON designate.* TO 'designate'@'%' IDENTIFIED BY
'$DESIGNATE_PASSWORD';
GRANT ALL ON designate.* TO 'designate'@'localhost' IDENTIFIED BY
'$DESIGNATE_PASSWORD';
CREATE DATABASE designate_pool_manager;
GRANT ALL ON designate_pool_manager.* TO 'designate'@'%' IDENTIFIED
BY '$DESIGNATE_PASSWORD';
GRANT ALL ON designate_pool_manager.* TO 'designate'@'localhost'
IDENTIFIED BY '$DESIGNATE_PASSWORD';
FLUSH PRIVILEGES;
quit
EOF
```

15. Create the DNSaaS service account in keystone:

```
openstack user create designate --password $DESIGNATE_PASSWORD --
email designate@localhost
```

16. Add the DNSaaS account to the **service** project:

```
openstack role add --project $SERVICES_PROJECT_NAME --user designate
admin
```

17. Create the DNSaaS service:

```
openstack service create dns --name designate --description
"Designate DNS Service"
```

18. Create the DNSaaS endpoint:

```
openstack endpoint create --region RegionOne --publicurl
http://$DESIGNATE_VIP_IP:9001 --internalurl
```

```
http://$DESIGNATE_VIP_IP:9001 --adminurl
http://$DESIGNATE_VIP_IP:9001 designate
```

19. Add the keystone token settings to the DNSaaS configuration:

```
crudini --set /etc/designate/designate.conf keystone_authtoken
auth_uri http://$KEYSTONE_SERVER_IP:5000/v2.0
crudini --set /etc/designate/designate.conf keystone_authtoken
identity_uri http://$KEYSTONE_SERVER_IP:35357/
crudini --set /etc/designate/designate.conf keystone_authtoken
admin_tenant_name $SERVICES_PROJECT_NAME
crudini --set /etc/designate/designate.conf keystone_authtoken
project_name $SERVICES_PROJECT_NAME
crudini --set /etc/designate/designate.conf keystone_authtoken
admin_user designate
crudini --set /etc/designate/designate.conf keystone_authtoken
admin_password $DESIGNATE_PASSWORD
```

20. Configure the API extensions for DNSaaS:

```
crudini --set /etc/designate/designate.conf service:api
enabled_extensions_v1 "diagnostics, quotas, reports, sync, touch"
crudini --set /etc/designate/designate.conf service:api
enabled_extensions_v2 "quotas, reports"
```

21. Configure DNSaaS to integrate with the **`Instances`** project:

```
crudini --set /etc/designate/designate.conf service:central
managed_resource_tenant_id $INSTANCES_TENANT_ID
```

22. Add the connection to the backend database:

```
crudini --set /etc/designate/designate.conf storage:sqlalchemy
connection
mysql+pymysql://designate:$DESIGNATE_PASSWORD@$MYSQL_SERVER_IP/desig
nate
crudini --set /etc/designate/designate.conf
pool_manager_cache:sqlalchemy connection
mysql+pymysql://designate:$DESIGNATE_PASSWORD@$MYSQL_SERVER_IP/desig
nate_pool_manager
```

23. And the Messaging endpoint:

```
crudini --set /etc/designate/designate.conf oslo_messaging_rabbit
rabbit_hosts $RABBIT_SERVER_IP:5672
```

24. Populate and prepare the Designate MySQL database:

```
su -s /bin/sh -c "designate-manage database sync" designate
su -s /bin/sh -c "designate-manage pool-manager-cache sync"
designate
```

25. Enable and start *only* the **`central`** and **`api`** designate services:

```
systemctl enable designate-central designate-api
systemctl start designate-central designate-api
```

26. Create the following file in **/etc/designate/pools.yaml**. Remember that you need to change the variables **EXTERNAL_DNS_SERVER_FQDN**, **EXTERNAL_DNS_SERVER_IP** and **DESIGNATE_SERVER_1**. There are provisions for additional DNS servers, if needed:

```
- name: default
  description: Default BIND9 Pool

  attributes:
 external: true
  ns_records:
 - hostname: $EXTERNAL_DNS_SERVER_FQDN.
   priority: 1
#   - hostname: $EXTERNAL_DNS_SERVER_FQDN_2.
#  priority: 1
  nameservers:
 - host: $EXTERNAL_DNS_SERVER_IP
   port: 53
#   - host: $EXTERNAL_DNS_SERVER_IP_2
#  port: 53

  targets:
 - type: bind9
   description: BIND9 Server 1
   masters:
     - host: $DESIGNATE_SERVER_1
       port: 5354
#    - host: $DESIGNATE_SERVER_2
#      port: 5354
#    - host: $DESIGNATE_SERVER_3
#      port: 5354
   options:
     host: $EXTERNAL_DNS_SERVER_IP
     port: 53
     rndc_host: $EXTERNAL_DNS_SERVER_IP
     rndc_port: 953
     rndc_key_file: /etc/designate/rndc.key

#   - type: bind9
#  description: BIND9 Server 2
#  masters:
#    - host: $DESIGNATE_SERVER_1
#      port: 5354
##   - host: $DESIGNATE_SERVER_2
##     port: 5354
##   - host: $DESIGNATE_SERVER_3
##     port: 5354
#  options:
#    host: $EXTERNAL_DNS_SERVER_IP_2
#    port: 53
#    rndc_host: $EXTERNAL_DNS_SERVER_IP_2
#    rndc_port: 953
#    rndc_key_file: /etc/designate/rndc.key
```

27. Copy the **rndc** keyfile to **/etc/designate**:

```
cp -f /etc/rndc.key /etc/designate/rndc.key
```

28. Ensure **designate** owns it:

```
chown designate:designate /etc/designate/rndc.key
```

29. Load the above YAML file into the DNSaaS runtime configuration:

```
su -s /bin/sh -c "designate-manage pool update" designate
```

30. Start the remaining DNSaaS services:

```
systemctl enable designate-pool-manager designate-mdns designate-
sink
systemctl start designate-pool-manager designate-mdns designate-sink
```

31. Create a DNS zone and export the **ZONE_ID** variable after its creation:

```
ZONE_ID=`openstack zone create --email admin@$ZONE_NAME $ZONE_NAME.
-f value -c id`
```

32. Add the UUID of the new zone to the nova and neutron handlers:

```
crudini --set /etc/designate/designate.conf handler:nova_fixed
domain_id $ZONE_ID
crudini --set /etc/designate/designate.conf
handler:neutron_floatingip domain_id $ZONE_ID
```

33. Restart the DNSaaS services:

```
systemctl restart designate-api designate-central designate-mdns
designate-pool-manager designate-sink
```

34. The DNSaaS portion is now fully configured. Next, you will configure neutron integration. Add **dns** to the list of ML2 drivers. For example:

```
crudini --set /etc/neutron/plugins/ml2/ml2_conf.ini ml2
extension_drivers port_security,dns
```

35. If you want your DNS agent (**dnsmasq**) to query DNSaaS (it does not by default):

```
crudini --set /etc/neutron/dhcp_agent.ini DEFAULT
dnsmasq_dns_servers $EXTERNAL_DNS_SERVER_IP
```

36. Enable DNSaaS integration for neutron:

```
crudini --set /etc/neutron/neutron.conf designate url
http://$DESIGNATE_VIP_IP:9001/v2
crudini --set /etc/neutron/neutron.conf designate admin_auth_url
http://$DESIGNATE_VIP_IP:35357/v2.0
```

```
crudini --set /etc/neutron/neutron.conf designate admin_username
designate
crudini --set /etc/neutron/neutron.conf designate admin_password
$DESIGNATE_PASSWORD
crudini --set /etc/neutron/neutron.conf designate admin_tenant_name
$SERVICES_PROJECT_NAME
crudini --set /etc/neutron/neutron.conf designate
allow_reverse_dns_lookup True
crudini --set /etc/neutron/neutron.conf designate
ipv4_ptr_zone_prefix_size 24
crudini --set /etc/neutron/neutron.conf designate
ipv6_ptr_zone_prefix_size 116
crudini --set /etc/neutron/neutron.conf designate insecure true
```

37. Restart the neutron and nova services:

```
openstack-service restart neutron
openstack-service restart nova
```

38. Configure your neutron network to use DNSaaS:

```
neutron net-update $INTERNAL_NET_ID --dns_domain $ZONE_NAME.
```

1.3. TEST OPENSTACK NETWORKING FLOATING IP RECORD CREATION

1. Check that the zone is correctly configured and in an **`ACTIVE`** state:

```
$ openstack zone list
+--------------------------------------+----------------------+-
--------+------------+--------+--------+
| id                                   | name                 |
type    |     serial | status | action |
+--------------------------------------+----------------------+-
--------+------------+--------+--------+
| 1bae1c6e-06e6-4a5d-8b03-483875478b73 | testzone.example.com. |
PRIMARY | 1523471795 | ACTIVE | NONE   |
+--------------------------------------+----------------------+-
--------+------------+--------+--------+
```

2. Enumerate the existing networks to retrieve the UUIDs. These will be used in the later steps. This example uses the **`internal`** and **`external`** networks:

```
$ openstack network list
+--------------------------------------+----------+-------------
-----------------------+
| ID                                   | Name     | Subnets
|
+--------------------------------------+----------+-------------
-----------------------+
| 0efce5d7-b2ec-4877-b6bb-de339a76c80b | external | d03cdc41-6962-
4d3d-bed1-68b0a5f6b93c |
| c020e6a9-f483-48a9-893d-983ae23d248a | internal | b59684af-d4ea-
403e-8b14-d3821a46d52f |
```

```
+--------------------------------------+----------+-------------
-----------------------+
```

3. Create a instance named **testinstance**, using base image named **web**, flavor **m1.small**, attached to network **internal**, with SSH keypair **keypair-demo**:

```
$ openstack server create --image web --flavor m1.small --nic net-
id=c020e6a9-f483-48a9-893d-983ae23d248a --key-name keypair-demo -f
value -c id  testinstance
14e1d0da-30bd-4adf-927b-8f54932cbe95
```

4. Confirm that your instance enters the **ACTIVE** state before proceeding:

```
$ openstack server list
+--------------------------------------+-------------+--------+-
---------------------+------------+
| ID                                   | Name        | Status |
Networks             | Image Name |
+--------------------------------------+-------------+--------+-
---------------------+------------+
| 14e1d0da-30bd-4adf-927b-8f54932cbe95 | testinstance | ACTIVE |
interna=192.168.10.12 | web        |
+--------------------------------------+-------------+--------+-
---------------------+------------+
```

5. Review the DNSaaS records and confirm that the new instance does not yet have a record:

```
$ openstack recordset list 1bae1c6e-06e6-4a5d-8b03-483875478b73
+--------------------------------------+----------------------+-
-----+-------------------------------------------------------------
------------+--------+--------+
| id                                   | name                 |
type | records
| status | action |
+--------------------------------------+----------------------+-
-----+-------------------------------------------------------------
------------+--------+--------+
| 4092f9f2-4fcb-4097-91bc-c4fa3b0965f9 | testzone.example.com. | NS
| example10-cont1.testzone.example.com.
| ACTIVE | NONE   |
| c08400a1-2bb1-4f6b-ae90-9f1b3a3dec82 | testzone.example.com. | SOA
| example10-cont1.testzone.example.com. admin.testzone.example.com.
|        |        |
|                                      |                      |
| 1523471795 3531 600 86400 3600
| ACTIVE | NONE   |
+--------------------------------------+----------------------+-
-----+-------------------------------------------------------------
------------+--------+--------+
```

6. Create a floating IP address in the **external** network.

```
 ----
$ openstack floating ip create -f value -c floating_ip_address
```

```
0efce5d7-b2ec-4877-b6bb-de339a76c80b
172.25.250.146
```

7. Attach the floating IP to **testinstance**:

```
$ openstack server add floating ip 14e1d0da-30bd-4adf-927b-
8f54932cbe95 172.25.250.146
```

8. Check DNSaaS records. This example shows an RR entry for **testinstance.testzone.example.com**, in a **PENDING** state.

```
$ openstack recordset list 1bae1c6e-06e6-4a5d-8b03-483875478b73
+--------------------------------------+------------------------
-----------+------+------------------------------------------------
---+---------+--------+
| id                                   | name
| type | records                                          | status
| action |
+--------------------------------------+------------------------
-----------+------+------------------------------------------------
---+---------+--------+
| 4092f9f2-4fcb-4097-91bc-c4fa3b0965f9 | testzone.example.com.
| NS   | example10-cont1.testzone.example.com.            | ACTIVE
| NONE   |
| c08400a1-2bb1-4f6b-ae90-9f1b3a3dec82 | testzone.example.com.
| SOA  | example10-cont1.testzone.example.com.            | PENDING |
UPDATE |
|                                      |
|      | admin.testzone.example.com. 1523473397 3531 600 |
|        |
|                                      |
|      | 86400 3600                                       |
|        |
| 058f6862-cf1b-4da3-9d78-b3a890aff32f |
testinstance.testzone.example.com. | A    | 172.25.250.146
| PENDING | CREATE |
+--------------------------------------+------------------------
-----------+------+------------------------------------------------
---+---------+--------+
```

9. After a few seconds wait, you can expect the newly-created entry to change to **ACTIVE**:

```
$ openstack recordset list 1bae1c6e-06e6-4a5d-8b03-483875478b73
+--------------------------------------+------------------------
-----------+------+------------------------------------------------
---+--------+--------+
| id                                   | name
| type | records                                          | status
| action |
+--------------------------------------+------------------------
-----------+------+------------------------------------------------
---+--------+--------+
| 4092f9f2-4fcb-4097-91bc-c4fa3b0965f9 | testzone.example.com.
```

```
| NS   | example10-cont1.testzone.example.com.           | ACTIVE |
NONE   |
| c08400a1-2bb1-4f6b-ae90-9f1b3a3dec82 | testzone.example.com.
| SOA  | example10-cont1.testzone.example.com.           | ACTIVE |
NONE   |
|                                       |
|      | admin.testzone.example.com. 1523473397 3531 600 |        |
|
|                                       |
|      | 86400 3600                                      |
|        |
| 058f6862-cf1b-4da3-9d78-b3a890aff32f |
testinstance.testzone.example.com. | A    | 172.25.250.146
| ACTIVE | NONE   |
+--------------------------------------+-------------------------
-----------+------+----------------------------------------------
---+--------+--------+
```

10. Use the DNSaaS Designate API to create a manual record. This example creates **web.testzone.example.com** as an alias to **testinstance.testzone.example.com**:

```
$ openstack recordset create --records
testinstance.testzone.example.com. --type CNAME  1bae1c6e-06e6-4a5d-
8b03-483875478b73 web
+-------------+--------------------------------------+
| Field       | Value                                |
+-------------+--------------------------------------+
| action      | CREATE                               |
| created_at  | 2018-04-11T19:05:23.000000           |
| description | None                                 |
| id          | 3f99a737-c1a4-4137-a4a5-26f934e60dfa |
| name        | web.testzone.example.com.            |
| project_id  | 6b34751621f449499569dfb077f5a7ed     |
| records     | testinstance.testzone.example.com.   |
| status      | PENDING                              |
| ttl         | None                                 |
| type        | CNAME                                |
| updated_at  | None                                 |
| version     | 1                                    |
| zone_id     | 1bae1c6e-06e6-4a5d-8b03-483875478b73 |
| zone_name   | testzone.example.com.                |
+-------------+--------------------------------------+
```

11. Check the DNSaaS configuration. It should now contain a record for **web.testzone.example.com**:

```
[root@example10-cont1 ~(keystone_admin)]# openstack recordset list
1bae1c6e-06e6-4a5d-8b03-483875478b73
+--------------------------------------+-------------------------
-----------+-------+----------------------------------------------
----+---------+--------+
| id                                   | name
| type  | records                                      | status
| action |
+--------------------------------------+-------------------------
```

```
-----------+-------+-----------------------------------------
----+---------+--------+
| 4092f9f2-4fcb-4097-91bc-c4fa3b0965f9 | testzone.example.com.
| NS    | example10-cont1.testzone.example.com.           | ACTIVE
| NONE   |
| c08400a1-2bb1-4f6b-ae90-9f1b3a3dec82 | testzone.example.com.
| SOA   | example10-cont1.testzone.example.com.           | PENDING
| UPDATE |
|                                       |
|       | admin.testzone.example.com. 1523473523 3531 600 |
|        |
|                                       |
|       | 86400 3600                                      |
|        |
| 058f6862-cf1b-4da3-9d78-b3a890aff32f |
testinstance.testzone.example.com. | A     | 172.25.250.146
| ACTIVE  | NONE   |
| 3f99a737-c1a4-4137-a4a5-26f934e60dfa | web.testzone.example.com.
| CNAME | testinstance.testzone.example.com.              | PENDING
| CREATE |
+--------------------------------------+-----------------------
------+-------+-------------------------------------------------
--------------+--------+
```

12. Run a DNS test, pointing to **`localhost`** as the DNS server, as this is where the DNSaaS service actually runs. This example attempts to resolve **`web.testzone.example.com`**:

```
$ host web.testzone.example.com. localhost
Using domain server:
Name: localhost
Address: ::1#53
Aliases:

web.testzone.example.com is an alias for
testinstance.testzone.example.com.
testinstance.testzone.example.com has address 172.25.250.146
```

13. Check the reverse DNS configuration:

```
$ host 172.25.250.146 localhost
Using domain server:
Name: localhost
Address: ::1#53
Aliases:

146.250.25.172.in-addr.arpa domain name pointer
testinstance.testzone.example.com.
```

For more information, refer to the OpenStack Designate API V2 client documentation.

CHAPTER 2. INSTALLING DNSAAS FOR HIGH AVAILABILITY

This chapter describes how to install DNSaaS (designate) in a high availability configuration. In this configuration the DNSaaS service is installed on a primary node, and it configuration is replicated to secondary nodes. The high availability service is performed by **redis**, allowing a secondary node to take over in the event of a failure on the primary node. Note that **memcached** is not supported as the High Availability back-end.

IMPORTANT

DNS-as-a-Service (DNSaaS), also known as Designate, is available in this release as a *Technology Preview*, and therefore is not fully supported by Red Hat. If you are interested in running DNSaaS in your production environment, please file a support ticket and mention the bug tracker **BZ#1374002**, so we can gauge the interest for this tool. For more information about Technology Preview features, see https://access.redhat.com/support/offerings/techpreview.

2.1. INSTALL THE DNS SERVICE

This section describes how to install the required DNS service. This can be a standalone server or co-located on an OpenStack controller:

```
#!/bin/bash

yum install -y bind bind-utils

sed -i -e "s/listen-on port.*/listen-on port 53 { any; };/"
/etc/named.conf
sed -i '/^options.*/i include "/etc/rndc.key"; controls {         inet *
allow { any; } keys { "rndc-key"; }; };' /etc/named.conf
sed -i '/allow-query.*/d' /etc/named.conf
sed -i '/^options.*/a         allow-new-zones yes;         allow-query {
any; };' /etc/named.conf

rndc-confgen -a

chmod g+w /var/named
setsebool named_write_master_zones 1

systemctl enable named
systemctl start named
```

2.2. CONFIGURE DNSAAS ON THE PRIMARY NODE

This section describes how to install and configure DNSaaS. Perform these steps on the primary (**master**) node:

1. Install the DNSaaS packages.

```
yum install -y openstack-designate-api openstack-designate-central
openstack-designate-sink openstack-designate-pool-manager openstack-
designate-mdns openstack-designate-common python-designate python-
```

```
designateclient openstack-designate-agent openstack-utils bind bind-
utils python-redis
```

2. Disable the **named** service:

```
systemctl disable named
```

3. Source your **openstackrc** file, as the following steps interact with OpenStack services.

4. To ease the deployment process, this guide relies on a number of variables; you will need to populate the values accordingly:

```
CONTROLLER_IP_ADDRESS=192.168.2.1
ZONE_NAME=testzone.example.com
INTERNAL_NET_NAME=net_internal
INSTANCES_PROJECT_NAME=myinstancesproject
SERVICES_PROJECT_NAME=service
DESIGNATE_PASSWORD=SecureDesignatePassword
EXTERNAL_DNS_SERVER_IP=$CONTROLLER_IP_ADDRESS
EXTERNAL_DNS_SERVER_FQDN=`hostname`
DESIGNATE_VIP_IP=$CONTROLLER_IP_ADDRESS
RABBIT_SERVER_IP=$CONTROLLER_IP_ADDRESS
REDIS_SERVER_IP=$CONTROLLER_IP_ADDRESS
MYSQL_SERVER_IP=$CONTROLLER_IP_ADDRESS
KEYSTONE_SERVER_IP=$CONTROLLER_IP_ADDRESS
DESIGNATE_SERVER_1=$CONTROLLER_IP_ADDRESS

SERVICES_TENANT_ID=`openstack project show $SERVICES_PROJECT_NAME -f
value -c id`
INSTANCES_TENANT_ID=`openstack project show $INSTANCES_PROJECT_NAME
-f value -c id`

DEFAULT_NAMESERVER_ID=$(uuidgen)
DEFAULT_TARGET_ID=$(uuidgen)
INTERNAL_NET_ID=`openstack network show $INTERNAL_NET_NAME -f value
-c id`
```

5. Configure **redis-sentinel**:

   a. Ensure the **/etc/redis.conf** file contains a **bind** clause pointing to the external IP address.

   b. Edit **/etc/redis-sentinel.conf** and change the **localhost** IP address to the Primary Controller public IP address. Remember to do this on each participating controller, and specify the same IP address in every **redis-sentinel** node.

```
sed -i "s/sentinel monitor mymaster 127.0.0.1 6379 2/sentinel
monitor mymaster $REDIS_SERVER_IP 6379 2/g" /etc/redis-
sentinel.conf
```

6. Enable and start the **redis** and **redis-sentinel** services:

```
# systemctl enable redis redis-sentinel
# systemctl start redis redis-sentinel
```

7. Copy **/etc/redis-sentinel.conf** to the other OpenStack controllers that run **redis** and repeat step 3.

8. Export the **redis-sentinel** cluster name:

```
REDIS_SENTINEL_NAME=`grep -v \\# /etc/redis-sentinel.conf | grep
"sentinel monitor" | awk '{print $3}'`
```

9. Create the backend database:

```
mysql -u root << EOF
CREATE DATABASE designate;
GRANT ALL ON designate.* TO 'designate'@'%' IDENTIFIED BY
'$DESIGNATE_PASSWORD';
GRANT ALL ON designate.* TO 'designate'@'localhost' IDENTIFIED BY
'$DESIGNATE_PASSWORD';
CREATE DATABASE designate_pool_manager;
GRANT ALL ON designate_pool_manager.* TO 'designate'@'%' IDENTIFIED
BY '$DESIGNATE_PASSWORD';
GRANT ALL ON designate_pool_manager.* TO 'designate'@'localhost'
IDENTIFIED BY '$DESIGNATE_PASSWORD';
FLUSH PRIVILEGES;
quit
EOF
```

10. Create the DNSaaS service account in keystone:

```
openstack user create designate --password $DESIGNATE_PASSWORD --
email designate@localhost
```

11. Add the DNSaaS account to the **service** project:

```
openstack role add --project $SERVICES_TENANT_ID --user designate
admin
```

12. Create the DNSaaS service:

```
openstack service create dns --name designate --description
"Designate DNS Service"
```

13. Create the DNSaaS endpoint:

```
openstack endpoint create --region RegionOne --publicurl
http://$DESIGNATE_VIP_IP:9001 --internalurl
http://$DESIGNATE_VIP_IP:9001 --adminurl
http://$DESIGNATE_VIP_IP:9001 designate
```

14. Add the keystone token settings to the DNSaaS configuration:

```
crudini --set /etc/designate/designate.conf keystone_authtoken
auth_uri http://$KEYSTONE_SERVER_IP:5000/v2.0
crudini --set /etc/designate/designate.conf keystone_authtoken
identity_uri http://$KEYSTONE_SERVER_IP:35357/
```

```
crudini --set /etc/designate/designate.conf keystone_authtoken
admin_tenant_name $SERVICES_PROJECT_NAME
crudini --set /etc/designate/designate.conf keystone_authtoken
project_name $SERVICES_PROJECT_NAME
crudini --set /etc/designate/designate.conf keystone_authtoken
admin_user designate
crudini --set /etc/designate/designate.conf keystone_authtoken
admin_password $DESIGNATE_PASSWORD
```

15. Configure the API extensions for DNSaaS:

```
crudini --set /etc/designate/designate.conf service:api
enabled_extensions_v1 "diagnostics, quotas, reports, sync, touch"
crudini --set /etc/designate/designate.conf service:api
enabled_extensions_v2 "quotas, reports"
```

16. Configure DNSaaS to integrate with the **`Instances`** project:

```
crudini --set /etc/designate/designate.conf service:central
managed_resource_tenant_id $INSTANCES_TENANT_ID
```

17. Add the connection to the backend database:

```
crudini --set /etc/designate/designate.conf storage:sqlalchemy
connection
mysql+pymysql://designate:$DESIGNATE_PASSWORD@$MYSQL_SERVER_IP/desig
nate
crudini --set /etc/designate/designate.conf
pool_manager_cache:sqlalchemy connection
mysql+pymysql://designate:$DESIGNATE_PASSWORD@$MYSQL_SERVER_IP/desig
nate_pool_manager
```

18. Add the Messaging endpoint:

```
crudini --set /etc/designate/designate.conf oslo_messaging_rabbit
rabbit_hosts $RABBIT_SERVER_IP:5672
```

19. Add the **`redis-sentinel`** connection:

```
crudini --set /etc/designate/designate.conf coordination backend_url
redis://$REDIS_SERVER_IP:26379?sentinel=$REDIS_SENTINEL_NAME
```

20. Populate and prepare the Designate MySQL database:

```
su -s /bin/sh -c "designate-manage database sync" designate
su -s /bin/sh -c "designate-manage pool-manager-cache sync"
designate
```

21. Enable and start *only* the **`central`** and **`api`** designate services:

```
systemctl enable designate-central designate-api
systemctl start designate-central designate-api
```

22. Create the following file as **/etc/designate/pools.yaml**. Remember that you need to change the variables **EXTERNAL_DNS_SERVER_FQDN**, **EXTERNAL_DNS_SERVER_IP** and **DESIGNATE_SERVER_1**. There are provisions for additional DNS servers, if needed:

```
- name: default
  description: Default BIND9 Pool

  attributes:
    external: true
  ns_records:
    - hostname: $EXTERNAL_DNS_SERVER_FQDN.
      priority: 1
  nameservers:
    - host: $EXTERNAL_DNS_SERVER_IP
      port: 53

  targets:
    - type: bind9
      description: BIND9 Server 1
      masters:
        - host: $DESIGNATE_SERVER_1
          port: 5354
        - host: $DESIGNATE_SERVER_2
          port: 5354
        - host: $DESIGNATE_SERVER_3
          port: 5354
      options:
        host: $EXTERNAL_DNS_SERVER_IP
        port: 53
        rndc_host: $EXTERNAL_DNS_SERVER_IP
        rndc_port: 953
        rndc_key_file: /etc/designate/rndc.key
```

23. Copy **/etc/rndc.key** to **/etc/designate/rndc.key**. Remember to set permissions accordingly:

```
chown designate:designate /etc/designate/rndc.key
```

24. Load the above YAML file into the DNSaaS runtime configuration:

```
su -s /bin/sh -c "designate-manage pool update" designate
```

25. Start the remaining DNSaaS services:

```
systemctl enable designate-pool-manager designate-mdns designate-
sink
systemctl start designate-pool-manager designate-mdns designate-sink
```

NOTE: Do not close your SSH session, as you will need the populated variables in the following sections.

2.3. ADD SECONDARY NODES

You can add secondary nodes that will participate in the redis cluster. Perform these steps on the secondary nodes:

1. Install the DNSaaS packages.

```
yum install -y openstack-designate-api openstack-designate-central
openstack-designate-sink openstack-designate-pool-manager openstack-
designate-mdns openstack-designate-common python-designate python-
designateclient openstack-designate-agent openstack-utils bind bind-
utils python-redis
```

2. Disable the **named** service:

```
systemctl disable named
```

3. Configure Redis Sentinel:

   a. Esure that in **/etc/redis.conf** the **bind** clause points to the **this** controller external IP address.

   b. Copy the **redis-sentinel** configuration from your master node. Leave the IP address unchanged:

```
scp designate-1:/etc/redis-sentinel.conf /etc
```

4. Enable and start the **redis** and **redis-sentinel** services:

```
# systemctl enable redis redis-sentinel
# systemctl start redis redis-sentinel
```

5. Repeat steps 1 to 3 for each controller that is running **redis**.

6. Test the **redis-sentinel** functionality:

```
# redis-cli -h <PRIMARY CONTROLLER IP ADDRESS> -p 26379
192.168.122.10:26379> sentinel master mymaster
 1) "name"
 2) "mymaster"
 3) "ip"
 4) "192.168.122.10"
 5) "port"
 6) "6379"
 7) "runid"
 8) "1865a0b3b237d20954a4e5fae14c6c7c932b0cf5"
 9) "flags"
10) "master"
11) "link-pending-commands"
12) "0"
13) "link-refcount"
14) "1"
15) "last-ping-sent"
16) "0"
17) "last-ok-ping-reply"
18) "459"
```

```
19) "last-ping-reply"
20) "459"
21) "down-after-milliseconds"
22) "30000"
23) "info-refresh"
24) "7024"
25) "role-reported"
26) "master"
27) "role-reported-time"
28) "509706"
29) "config-epoch"
30) "0"
31) "num-slaves"
32) "1"
33) "num-other-sentinels"
34) "0"
35) "quorum"
36) "2"
37) "failover-timeout"
38) "180000"
39) "parallel-syncs"
40) "1"
192.168.122.10:26379>
```

7. Copy your DNSaaS configuration from your master node:

```
scp designate-1:/etc/designate/* /etc/designate
```

8. Start the only needed services. **NOTE:** Do not start the **pool** agent in the standby nodes.

```
systemctl enable designate-api designate-central designate-mdns
designate-sink
systemctl start designate-api designate-central designate-mdns
designate-sink
```

2.4. CONFIGURE NEUTRON INTEGRATION

Perform this procedure on the primary node.

1. Create the DNS zone:

```
ZONE_ID=`openstack zone create --email admin@$ZONE_NAME $ZONE_NAME.
-f value -c id`
crudini --set /etc/designate/designate.conf handler:nova_fixed
domain_id $ZONE_ID
crudini --set /etc/designate/designate.conf
handler:neutron_floatingip domain_id $ZONE_ID
```

2. Copy the configuration to the remaining designate cluster members. For example:

```
scp /etc/designate/* designate-2:/etc/designate
scp /etc/designate/* designate-3:/etc/designate
```

3. For the primary node only: restart the designate services:

```
for i in api central mdns pool-manager sink ; do
    systemctl restart designate-$i
done
```

4. On the remaining nodes, restart the designate services:

```
for i in api central mdns sink ; do
    systemctl restart designate-$i
done
```

5. On the primary node, configure the neutron integration:

```
crudini --set /etc/neutron/plugins/ml2/ml2_conf.ini ml2
extension_drivers port_security,dns

crudini --set /etc/neutron/neutron.conf DEFAULT dns_domain
$ZONE_NAME.
crudini --set /etc/neutron/neutron.conf DEFAULT external_dns_driver
designate

crudini --set /etc/neutron/neutron.conf designate url
http://$DESIGNATE_VIP_IP:9001/v2
crudini --set /etc/neutron/neutron.conf designate admin_auth_url
http://$DESIGNATE_VIP_IP:35357/v2.0
crudini --set /etc/neutron/neutron.conf designate admin_username
designate
crudini --set /etc/neutron/neutron.conf designate admin_password
$DESIGNATE_PASSWORD
crudini --set /etc/neutron/neutron.conf designate admin_tenant_name
$SERVICES_PROJECT_NAME
crudini --set /etc/neutron/neutron.conf designate
allow_reverse_dns_lookup True
crudini --set /etc/neutron/neutron.conf designate
ipv4_ptr_zone_prefix_size 24
crudini --set /etc/neutron/neutron.conf designate
ipv6_ptr_zone_prefix_size 116
crudini --set /etc/neutron/neutron.conf designate insecure true
```

6. Copy the **/etc/neutron/plugins/ml2/ml2_conf.ini** and **/etc/neutron/neutron.conf** configuration to the other participating controllers.

7. Once the files have finished copying, restart the **neutron** service:

```
openstack-service restart neutron
```

8. Make the **neutron** service aware that all instances within the internal network are now part of the DNS domain managed by designate:

```
neutron net-update $INTERNAL_NET_ID  --dns_domain $ZONE_NAME.
```