



Red Hat Enterprise Linux OpenStack Platform 7

DNS-as-a-Service Guide

Integrate DNS Management with Red Hat Enterprise Linux OpenStack Platform

Red Hat Enterprise Linux OpenStack Platform 7 DNS-as-a-Service Guide

Integrate DNS Management with Red Hat Enterprise Linux OpenStack Platform

OpenStack Team

rhos-docs@redhat.com

Legal Notice

Copyright © 2018 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution-Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux ® is the registered trademark of Linus Torvalds in the United States and other countries.

Java ® is a registered trademark of Oracle and/or its affiliates.

XFS ® is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL ® is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js ® is an official trademark of Joyent. Red Hat Software Collections is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack ® Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

Abstract

A guide for integrating DNS with RHEL OpenStack Platform.

Table of Contents

CHAPTER 1. OVERVIEW OF DNSAAS	3
1.1. TOPICS COVERED IN THIS GUIDE	3
1.2. DNSAAS PREREQUISITES	3
1.3. DNSAAS SERVICES	3
1.4. DNSAAS INTEGRATION WITH COMPUTE AND OPENSTACK NETWORKING	4
CHAPTER 2. MANUAL DNSAAS INSTALLATION	5
CHAPTER 3. INSTALL AND CONFIGURE POWERDNS	10
3.1. INSTALL POWERDNS	10
3.2. CONFIGURE POWERDNS	10
3.3. TROUBLESHOOTING	11
3.4. TEST DNSAAS INTEGRATION WITH POWERDNS	12
3.5. CONFIGURE AUTO-GENERATION OF DNS RECORDS (NOVA FIXED AND NEUTRON FLOATING)	12
3.6. TEST OPENSTACK NETWORKING (NEUTRON) FLOATING IP RECORD CREATION	13
3.7. CLEANUP OPENSTACK NETWORKING AND COMPUTE DNS ENTRIES	13
3.8. CLEANUP THE POWERDNS CONFIGURATION	14
CHAPTER 4. INSTALL AND CONFIGURE BIND9	15
4.1. BASIC BIND INSTALLATION	15
4.2. CONFIGURE BIND	15
4.3. CONFIGURE THE DNSAAS POOL TARGET FOR BIND	16
4.4. TEST BIND	16
4.5. TEST DNSAAS INTEGRATION WITH BIND9	17
4.6. CONFIGURE AUTO-GENERATION OF DNS RECORDS (NOVA FIXED AND NEUTRON FLOATING)	17
4.7. TEST OPENSTACK NETWORKING FLOATING IP RECORD CREATION	18
4.8. CLEANUP OPENSTACK NETWORKING AND COMPUTE DNS ENTRIES	18

CHAPTER 1. OVERVIEW OF DNSAAS

Red Hat Enterprise Linux OpenStack Platform 7 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 PowerDNS and Bind9.



NOTE

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. 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](#).

1.1. TOPICS COVERED IN THIS GUIDE

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

1.2. DNSAAS PREREQUISITES

- A fully functioning OpenStack Networking-based, non-HA OpenStack environment.
- An OpenStack Image Service (*glance*) image loaded, for testing auto-creation.

1.3. 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 (<i>AXFR</i>). Not needed for PowerDNS or BIND backends.
------------------------	---

**NOTE**

The **zone-manager** service is expected to be added in the next major release. It will run periodic tasks on zones to provide a mechanism for identifying lost events.

1.4. 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 servers (PowerDNS or 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.

CHAPTER 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_enterprise_linux_openstack_platform/7/html-single/director_installation_and_usage/#sect-Registering_your_System

1. Install the DNSaaS packages on the *controller* node:

```
# 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
```

2. Create the DNSaaS and Pool Manager databases. Update the **IDENTIFIED BY** 'ComplexAlphanumericPassword' value to suit your environment.

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

3. Create the DNSaaS service accounts and endpoint in OpenStack Identity (keystone): This example uses the DNSaaS host IP address **192.168.100.20**. You will likely need to update these steps to suit your environment.

```
# source ~/keystonerc_admin
# keystone user-create --name designate --pass ComplexAlphanumericPassword
--email designate@localhost
# keystone user-role-add --user designate --role admin --service
# keystone service-create --name designate --type dns --description
"Designate DNS Service"
# keystone endpoint-create --service designate --publicurl
"http://192.168.100.20:9001" --adminurl "http://192.168.100.20:9001" --
internalurl "http://192.168.100.20:9001" --region regionOne
```

4. Add firewall rules for DNSaaS:

```
# iptables -I INPUT -p tcp -m multiport --dports 9001 -m comment --comment
"designate incoming" -j ACCEPT
# iptables -I INPUT -p tcp -m multiport --dports 5354 -m comment --comment
```

```
"Designate mdns incoming" -j ACCEPT
```

If hosting DNS locally, ensure that the required rules are open:

```
# iptables -I INPUT -p tcp -m multiport --dports 53 -m comment --comment
"bind/powerdns incoming" -j ACCEPT
# iptables -I INPUT -p udp -m multiport --dports 53 -m comment --comment
"bind/powerdns incoming" -j ACCEPT
# iptables -I INPUT -p tcp -m multiport --dports 953 -m comment --comment
"rndc incoming - bind only" -j ACCEPT
# service iptables save; service iptables restart
```

5. Configure the DNSaaS database connection: Be sure to enter your DNSaaS host IP address correctly in the steps below; replace **ComplexAlphanumericPassword** with the value that aligns with your environment.

```
# openstack-config --set /etc/designate/designate.conf storage:sqlalchemy
connection
mysql://designate:ComplexAlphanumericPassword@192.168.100.20/designate
# openstack-config --set /etc/designate/designate.conf storage:sqlalchemy
max_retries -1
# openstack-config --set /etc/designate/designate.conf
pool_manager_cache:sqlalchemy connection
mysql://designate:ComplexAlphanumericPassword@192.168.100.20/designate_poo
l_manager
# openstack-config --set /etc/designate/designate.conf
pool_manager_cache:sqlalchemy max_retries -1
```

6. Configure authentication to the Identity Service (*keystone*): Make certain that the **admin_password** option aligns with your environment.

```
# openstack-config --set /etc/designate/designate.conf keystone_auth_token
auth_uri http://192.168.100.20:5000/v2.0
# openstack-config --set /etc/designate/designate.conf keystone_auth_token
identity_uri http://192.168.100.20:35357/
# openstack-config --set /etc/designate/designate.conf keystone_auth_token
admin_tenant_name service
# openstack-config --set /etc/designate/designate.conf keystone_auth_token
admin_user designate
# openstack-config --set /etc/designate/designate.conf keystone_auth_token
admin_password ComplexAlphanumericPassword
```

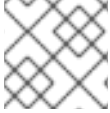
7. Configure the DNSaaS connection to RabbitMQ:

Make certain the **rabbit_userid** and **rabbit_password** options align with your environment.

```
# openstack-config --set /etc/designate/designate.conf
oslo_messaging_rabbit rabbit_hosts 192.168.100.20:5672
# openstack-config --set /etc/designate/designate.conf
oslo_messaging_rabbit rabbit_ha_queues False
# openstack-config --set /etc/designate/designate.conf
oslo_messaging_rabbit rabbit_host 192.168.100.20
# openstack-config --set /etc/designate/designate.conf
oslo_messaging_rabbit rabbit_port 5672
# openstack-config --set /etc/designate/designate.conf
```

```
oslo_messaging_rabbit rabbit_userid amqp_user
# openstack-config --set /etc/designate/designate.conf
oslo_messaging_rabbit rabbit_password ComplexAlphanumericPassword
# openstack-config --set /etc/designate/designate.conf
oslo_messaging_rabbit rabbit_virtual_host /
```

8. Initial DNSaaS configuration:



NOTE

Prior to the *Kilo* release, Compute used `rpc_notifier` below.

```
# openstack-config --set /etc/designate/designate.conf DEFAULT
notification_driver nova.openstack.common.notifier.rpc_notifier
# openstack-config --set /etc/designate/designate.conf DEFAULT
notification_driver messaging
# openstack-config --set /etc/designate/designate.conf DEFAULT
notification_topics notifications_designate
# openstack-config --set /etc/designate/designate.conf service:api
api_host 0.0.0.0
# openstack-config --set /etc/designate/designate.conf service:api
api_port 9001
# openstack-config --set /etc/designate/designate.conf service:api
auth_strategy keystone
# openstack-config --set /etc/designate/designate.conf service:api
enable_api_v1 True
# openstack-config --set /etc/designate/designate.conf service:api
enabled_extensions_v1 "diagnostics, quotas, reports, sync, touch"
# openstack-config --set /etc/designate/designate.conf service:api
enable_api_v2 True
# openstack-config --set /etc/designate/designate.conf service:api
enabled_extensions_v2 "quotas, reports"
```

9. Configure the pool manager:



NOTE

At present, you will not yet configure a pool target as you have not selected a backend. That occurs later in this procedure.

The `pool_id` is hardcoded, so use the **UUID** shown below:

```
# pool_id=794ccc2c-d751-44fe-b57f-8894c9f5c842
# nameserver_id=$(uuidgen)
# target_id=$(uuidgen)
# openstack-config --set /etc/designate/designate.conf
service:pool_manager pool_id $pool_id
# openstack-config --set /etc/designate/designate.conf pool:$pool_id
nameservers $nameserver_id
# openstack-config --set /etc/designate/designate.conf pool:$pool_id
targets $target_id
# openstack-config --set /etc/designate/designate.conf
pool_nameserver:$nameserver_id port 53
# openstack-config --set /etc/designate/designate.conf
```

```
pool_nameserver:$nameserver_id host 192.168.100.20
```

10. Configure the DNSaaS Sink:



NOTE

For now, you will not configure the domain used by sink (as it does not exist yet).

```
# openstack-config --set /etc/designate/designate.conf service:sink
enabled_notification_handlers "nova_fixed, neutron_floatingip"
# openstack-config --set /etc/designate/designate.conf handler:nova_fixed
notification_topics notifications_designate
# openstack-config --set /etc/designate/designate.conf handler:nova_fixed
control_exchange nova
# openstack-config --set /etc/designate/designate.conf handler:nova_fixed
format "%(display_name)s.%(domain)s"
# openstack-config --set /etc/designate/designate.conf
handler:neutron_floatingip notification_topics notifications_designate
# openstack-config --set /etc/designate/designate.conf
handler:neutron_floatingip control_exchange neutron
# openstack-config --set /etc/designate/designate.conf
handler:neutron_floatingip format "%(octet0)s-%(octet1)s-%(octet2)s-%
(octet3)s.%(domain)s"
```

11. Configure Compute and OpenStack Networking to send notifications



NOTE

Ceilometer's agent also listens and consumes notifications. Create a specific **Designate** notifications queue (as shown below) so they don't conflict.

OpenStack Compute in the *Kilo* release uses **messaging** as its notification driver; previously it was `nova.openstack.common.notifier.rpc_notifier`

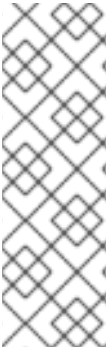
```
# openstack-config --set /etc/nova/nova.conf DEFAULT notification_topics
notifications,notifications_designate
# openstack-config --set /etc/nova/nova.conf DEFAULT
notify_on_state_change vm_and_task_state
# openstack-config --set /etc/nova/nova.conf DEFAULT
instance_usage_audit_period hour
# openstack-config --set /etc/nova/nova.conf DEFAULT instance_usage_audit
true
# openstack-config --set /etc/neutron/neutron.conf DEFAULT
notification_driver neutron.openstack.common.notifier.rpc_notifier
# openstack-config --set /etc/neutron/neutron.conf DEFAULT
notification_topics notifications,notifications_designate
# openstack-service restart nova
# openstack-service restart neutron
```

12. Manually verify the `notification_driver` in `nova.conf`:

**NOTE**

Due to the possibility of multiple **notification_drivers** in *nova.conf*, the *openstack-config* command might cause problems. Check in the **DEFAULT** section to ensure you have these two entries:

```
notification_driver=ceilometer.compute.nova_notifier
notification_driver=messaging
```

**NOTE**

If using a separate Compute node, it will need the following settings in *nova.conf*:

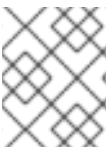
```
notification_driver
=nova.openstack.common.notifier.rabbit_notifier,ceilometer.comp
ute.nova_notifier
notification_driver =messaging
notification_topics=notifications,notifications_designate
```

13. Sync the DNSaaS and Pool Manager cache:

```
# designate-manage database sync
# designate-manage pool-manager-cache sync
```

14. Enable and start the DNSaaS services:

```
# systemctl enable designate-central
# systemctl enable designate-api
# systemctl enable designate-mdns
# systemctl enable designate-pool-manager
# systemctl start designate-central
# systemctl start designate-api
# systemctl start designate-mdns
# systemctl start designate-pool-manager
```

**NOTE**

At this point you have not created a DNS target for your pool, so don't expect a functioning DNSaaS deployment yet.

CHAPTER 3. INSTALL AND CONFIGURE POWERDNS

These steps install PowerDNS, and then configure integration with DNSaaS.

3.1. INSTALL POWERDNS

1. Install but disable the EPEL repository:

```
# cd /root
# curl -O https://dl.fedoraproject.org/pub/epel/7/x86_64/e/epel-release-7-5.noarch.rpm
# yum -y install epel-release-7-5.noarch.rpm
# yum-config-manager --disable epel
```

2. Install the PowerDNS and MySQL backends:

```
# yum -y --enablerepo=epel install pdns pdns-backend-mysql bind-utils
```

3. Create the PowerDNS database. Update the **IDENTIFIED BY** 'ComplexAlphanumericPassword' value to suit your environment.

```
# mysql -u root << EOF
CREATE DATABASE designate_pdns character set = 'utf8';
GRANT ALL ON designate_pdns.* TO 'designate'@'%' IDENTIFIED BY
'ComplexAlphanumericPassword';
GRANT ALL ON designate_pdns.* TO 'designate'@'localhost' IDENTIFIED BY
'ComplexAlphanumericPassword';
FLUSH PRIVILEGES;
quit
EOF
```

3.2. CONFIGURE POWERDNS

1. Update the **local-address** and **mysql-password** values in the `/etc/pdns/pdns.conf` file to suit your environment:

```
# General Config
#setgid=pdns
#setuid=pdns
config-dir=/etc/pdns
socket-dir=/var/run
guardian=yes
daemon=no
disable-axfr=no
local-address=192.168.100.20
local-port=53
master=no
slave=yes
cache-ttl=0
query-cache-ttl=0
negquery-cache-ttl=0
# Launch gmysql backend
launch=gmysql
```

```
# gmysql parameters
gmysql-host=localhost
gmysql-user=designate
gmysql-password=ComplexAlphanumericPassword
gmysql-dbname=designate_pdns
gmysql-dnssec=yes
logging-facility=0
```

2. Configure the DNSaaS pool target for PowerDNS:



NOTE

Remember to update your database password as needed. See the portion containing the string: **ComplexAlphanumericPassword**

```
# openstack-config --set /etc/designate/designate.conf
pool_target:$target_id type powerdns
# openstack-config --set /etc/designate/designate.conf
pool_target:$target_id options "connection:
mysql://designate:ComplexAlphanumericPassword@192.168.100.20/designate_pdn
s?charset=utf8"
# openstack-config --set /etc/designate/designate.conf
pool_target:$target_id masters 192.168.100.20:5354
```

3. Sync your database: This creates the table structure for PowerDNS in its database.

```
# designate-manage powerdns sync $target_id
```

4. Restart DNSaaS to apply the pool changes:

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

4. Start and enable the PowerDNS service:

```
# systemctl enable pdns
# systemctl start pdns
```

5. Test PowerDNS:

```
# netstat -tap | grep pdns
# netstat -tulpn | grep 53
# dig @192.168.100.20
```

3.3. TROUBLESHOOTING

You can review the DNSaaS logs for any error messages:

```
# cd /var/log/designate
```

```
# tail api.log
# tail central.log
# tail mdns.log
# tail pool-manager.log
# tail sink.log
```

3.4. TEST DNSAAS INTEGRATION WITH POWERDNS

1. Create your DNS server record (don't forget the `.` at the end of the `--name` option)

```
# designate server-list
# designate server-create --name $(hostname).
```

2. Create a domain (don't forget the `.` at the end of the `--name` option)

```
# designate domain-list
# designate domain-create --name example.com. --email root@example.com
```

3. Create a record and test lookup (don't forget the `.` at the end of the `--name` option)

```
# DOMAINID=$(designate domain-list | grep example.com | awk '{print $2}')
# designate record-create --name server1.example.com. --type A --data
1.2.3.4 $DOMAINID
# dig +short -p 53 @192.168.100.20 server1.example.com A
```

3.5. CONFIGURE AUTO-GENERATION OF DNS RECORDS (NOVA FIXED AND NEUTRON FLOATING)

The DNSaaS Sink listens to Compute and OpenStack Networking notifications, and will take action based on them. You completed the majority of the configuration in the previous steps, however, you need to specify which domain DNSaaS should use for the auto-generation of DNS entries.

1. Modify the DNSaaS configuration for the example domain:

```
# openstack-config --set /etc/designate/designate.conf handler:nova_fixed
domain_id $DOMAINID
# openstack-config --set /etc/designate/designate.conf
handler:neutron_floatingip domain_id $DOMAINID
# systemctl restart designate-api
# systemctl restart designate-central
# systemctl restart designate-mdns
# systemctl restart designate-pool-manager
# systemctl restart designate-sink
```

2. Test Compute (*nova*) record creation:

Here you will follow a normal **nova boot** procedure. This step can be performed from the OpenStack Dashboard if preferred. Also, any Image Service (*glance*) image should work.

This step assumes you have already validated that your OpenStack environment works without DNSaaS. You must already be able to boot instances, assign floating IP addresses, and have functional networking.


```
# glance image-list
# neutron net-list
# nova boot testserver --flavor m1.tiny --image cirros-0.3.4-x86_64 --key-
name yourkey --security-groups default --nic net-id=<Private Net ID>
```

3. Review the Sink log: Make certain your instance has moved into an **active** status (using **nova list**; **nova console-log testserver**). Once up, you should see a **create_record** entry if it has picked up the notification correctly.

```
# tail /var/log/designate/sink.log
```

Check in PDNS:

```
# dig +short @192.168.100.20 testserver.example.com
```

If this test doesn't work as expected, you can also check directly in the database:

```
# mysql
use designate_pdns;
show tables;
select * from records;
quit;
```

3.6. TEST OPENSTACK NETWORKING (NEUTRON) FLOATING IP RECORD CREATION

Run the command below to test the creation of a floating IP record (replace **pubnet1** with a name appropriate for your environment):

```
# FLOATINGIP=$(neutron floatingip-create pubnet1 | grep
floating_ip_address | awk '{print $4}')
# nova add-floating-ip testserver $FLOATINGIP
# DNSRESULT=$(echo $FLOATINGIP | sed 's/\./-/g').example.com
# dig +short @192.168.100.20 $DNSRESULT
```

You should see a **create_record** event in the log file:

```
# tail /var/log/designate/sink.log
```



NOTE

It is not currently possible to create floating IP addresses by hostname rather than octets. At present, you can instead create a **CNAME** against this manually. In a future update it is expected that you will be able to create by hostname.

3.7. CLEANUP OPENSTACK NETWORKING AND COMPUTE DNS ENTRIES

1. Remove the floating IP address created previously:

```
# nova remove-floating-ip testserver $FLOATINGIP
```

You should see a **delete_record** event in the log:

```
# tail /var/log/designate/sink.log
```

And the record should be removed.

2. Remove the *testserver* created previously:

```
# designate record-list $DOMAINID  
# nova delete testserver
```

You should see another **delete_record** event in the log:

```
# tail /var/log/designate/sink.log
```

If DNSaaS Sink manages a record it may not be updated manually. There is an option **designate --edit-managed record-delete <Domain ID> <Record ID>** which was created in *Liberty* and backported to the *Kilo* release. However it is not currently available in the generally available release of Red Hat Enterprise Linux OpenStack Platform 7. At present, you will need to manually remove it from your backend DNS and DNSaaS database in the event of any issues.

3.8. CLEANUP THE POWERDNS CONFIGURATION

You will not modify the DNSaaS configuration as you will overwrite it when configuring your Bind target.

```
# SERV1ID=$(designate record-list $DOMAINID | grep server1.example.com |  
awk '{print $2}')
```

```
# designate record-delete $DOMAINID $SERV1ID  
# designate domain-delete $DOMAINID  
# systemctl disable pdns  
# systemctl stop pdns
```

CHAPTER 4. INSTALL AND CONFIGURE BIND9

These steps install Bind9, and then configure integration with DNSaaS.

4.1. BASIC BIND INSTALLATION

1. Installed the *BIND* packages:

```
# yum install bind bind-utils
```

2. Configure *named* to listen for incoming connections:

```
# cp /etc/named.conf /etc/named.conf.orig
# sed -i -e "s/listen-on port.*/listen-on port 53 { 127.0.0.1;
192.168.100.20; };" /etc/named.conf
```

4.2. CONFIGURE BIND

1. Write to */etc/rndc.key*:

```
# rndc-confgen -a
```

2. Add the following before **options**

```
# sed -i '/^options.*/i \
include "/etc/rndc.key"; \
controls { \
    inet 127.0.0.1 allow { localhost; } keys { "rndc-key"; }; \
};' /etc/named.conf
```

3. Remove a few existing options you will rewrite later:

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

4. Add the following after **options**:

```
# sed -i '/^options.*/a \
    allow-new-zones yes; \
    allow-query { any; }; \
    recursion no;' /etc/named.conf
```

5. Create the *rndc* configuration. For the Compute node, the *rndc* configuration must point to the DNS server. For example:

```
# cat << EOF > /etc/rndc.conf
include "/etc/rndc.key";
options {
    default-key "rndc-key";
    default-server 192.168.100.20;
    default-port 953;
};
```

```
EOF
```

6. Review the *named* configuration:

```
# named-checkconf /etc/named.conf
```

7. Correct the file permissions:

```
# setsebool -P named_write_master_zones on
# chmod g+w /var/named
# chown named:named /etc/rndc.conf
# chown named:named /etc/rndc.key
# chmod 600 /etc/rndc.key
```

8. Enable and start the *named* service:

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

9. Validate *named* and *rndc*:

```
# dig @localhost localhost
# rndc status
```

4.3. CONFIGURE THE DNSaaS POOL TARGET FOR BIND

1. Overwrite the previous PowerDNS pool target configuration:

```
# openstack-config --set /etc/designate/designate.conf
pool_target:$target_id type bind9
# openstack-config --set /etc/designate/designate.conf
pool_target:$target_id options "rndc_host: 192.168.100.20, rndc_port: 953,
rndc_config_file: /etc/rndc.conf, rndc_key_file: /etc/rndc.key"
# openstack-config --set /etc/designate/designate.conf
pool_target:$target_id masters 192.168.100.20:5354
```

2. Restart DNSaaS to apply your pool changes:

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

4.4. TEST BIND

1. Perform the diagnostic commands below:

```
# netstat -tap | grep named
# netstat -tulpn | grep 53
# dig @192.168.100.20
```

2. Check the DNSaaS Logs for errors. Ignore errors in Sink for now, as you have not modified its configuration.

```
# cd /var/log/designate
# tail api.log
# tail central.log
# tail mdns.log
# tail pool-manager.log
# tail sink.log
```

4.5. TEST DNSAAS INTEGRATION WITH BIND9

1. Create an entry for your server:

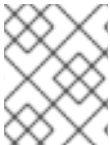
```
# designate server-create --name $(hostname).
```

2. Verify your DNS server record was previously created:

```
# designate server-list
```

3. Create a domain (don't forget the `.` at the end of the `--name` option)

```
# designate domain-list
# designate domain-create --name example.com. --email root@example.com
# DOMAINID=$(designate domain-list | grep example.com | awk '{print $2}')
```



NOTE

When creating a domain from designate against BIND, it is basically running a command similar to this:

```
# rndc -s 192.168.122.41 -p 953 -c /etc/rndc.conf -k /etc/rndc.key addzone
example.com '{ type slave; masters { 192.168.122.41 port 5354; }; file
"slave.example.com.ff532e15-55a9-4966-8f1e-b3eddb2891ba"; };
```

4. Create a record and test lookup (don't forget the `.` at the end of the `--name` option)

```
# designate record-create --name server1.example.com. --type A --data
1.2.3.4 $DOMAINID
# dig +short -p 53 @192.168.100.20 server1.example.com A
```

4.6. CONFIGURE AUTO-GENERATION OF DNS RECORDS (NOVA FIXED AND NEUTRON FLOATING)

1. Modify the DNSaaS configuration for the example domain. This will overwrite the PowerDNS configuration:

```
# openstack-config --set /etc/designate/designate.conf handler:nova_fixed
domain_id $DOMAINID
# openstack-config --set /etc/designate/designate.conf
```

```
handler:neutron_floatingip domain_id $DOMAINID
# systemctl restart designate-api
# systemctl restart designate-central
# systemctl restart designate-mdns
# systemctl restart designate-pool-manager
# systemctl restart designate-sink
```

2. Test OpenStack Compute (nova) record creation:

This follows the same procedure as previously done with PowerDNS:

```
# glance image-list
# neutron net-list
# nova boot testserver --flavor m1.tiny --image cirros-0.3.4-x86_64 --key-
name yourkey --security-groups default --nic net-id=<Private Net ID>
```

3. Check the Sink log:

Once the instance is up, you should see a **create_record** entry, if it has picked up the notification correctly:

```
# tail /var/log/designate/sink.log
```

Check in BIND

```
# dig +short @192.168.100.20 testserver.example.com
```

If this doesn't work, you can also check the files in **/var/named**.

4.7. TEST OPENSTACK NETWORKING FLOATING IP RECORD CREATION

1. Perform the diagnostic commands below (replace **pubnet1** with a name appropriate for your environment):

```
# FLOATINGIP=$(neutron floatingip-create pubnet1 | grep
floating_ip_address | awk '{print $4}')
# nova add-floating-ip testserver $FLOATINGIP
# DNSRESULT=$(echo $FLOATINGIP | sed 's/\./-/g').example.com
# dig +short @192.168.100.20 $DNSRESULT
```

2. You should see a **create_record** event in the log file:

```
# tail /var/log/designate/sink.log
```

4.8. CLEANUP OPENSTACK NETWORKING AND COMPUTE DNS ENTRIES

1. Remove the test floating IP created previously:

```
# nova remove-floating-ip testserver $FLOATINGIP
```

2. You should see a **delete_record** event in the log file:

```
# tail /var/log/designate/sink.log
```

And the record should now be removed.

3. Remove the *testserver* created previously:

```
# designate record-list $DOMAINID  
# nova delete testserver
```

You should see another **delete_record** entry in the log file:

```
# tail /var/log/designate/sink.log
```