



MidoNet Quick Start Guide

for RHEL 7 / Juno (RDO)

2015.03 (2015-04-27 09:13 UTC)



docs.midonet.org

MidoNet Quick Start Guide for RHEL 7 / Juno (RDO)

2015.03 (2015-04-27 09:13 UTC)

Copyright © 2015 Midokura SARL All rights reserved.

MidoNet is a network virtualization software for Infrastructure-as-a-Service (IaaS) clouds.

It decouples your IaaS cloud from your network hardware, creating an intelligent software abstraction layer between your end hosts and your physical network.

This guide walks through the minimum installation and configuration steps necessary to use MidoNet with OpenStack.



Note

Please consult the [MidoNet Mailing Lists or Chat](#) if you need assistance.

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Table of Contents

Preface	iv
Conventions	iv
1. Architecture	1
Hosts and Services	1
2. Basic Environment Configuration	3
Networking Configuration	3
SELinux Configuration	3
Repository Configuration	3
3. OpenStack Installation	5
Identity Service (Keystone)	5
Compute Services (Nova)	5
Networking Service (Neutron)	6
4. MidoNet Installation	11
NSDB Nodes	11
Controller Node	14
Midolman Installation	15
MidoNet Host Registration	15
5. Further Steps	17

Preface

Conventions

The MidoNet documentation uses several typesetting conventions.

Notices

Notices take these forms:



Note

A handy tip or reminder.



Important

Something you must be aware of before proceeding.



Warning

Critical information about the risk of data loss or security issues.

Command prompts

\$ prompt

Any user, including the root user, can run commands that are prefixed with the \$ prompt.

prompt

The root user must run commands that are prefixed with the # prompt. You can also prefix these commands with the **sudo** command, if available, to run them.

1. Architecture

Table of Contents

Hosts and Services	1
--------------------------	---



Important

This guide assumes the following system architecture, based on [Figure 2.1. Minimal architecture example with OpenStack Networking \(neutron\)—Network layout](#) of the OpenStack Documentation. This architecture consists of three hosts:

- Controller Node (**controller**)
- Gateway Node (**network**)
- Compute Node (**compute1**)

The *MidoNet Network State Database (NSDB)* uses [ZooKeeper](#) and [Cassandra](#) to store network topology and state information. The NSDB components can be installed on separate hosts, but this guide assumes them to be installed on all three nodes (**controller**, **network**, **compute1**).

The *MidoNet Agent (Midolman)* has to be installed on all nodes where traffic enters or leaves the virtual topology, in this guide this are the **network** and **compute1** nodes.

The *Midonet API* can be installed on a separate host, but this guide assumes it to be installed on the **controller** node.

The *Midonet Command Line Interface (CLI)* can be installed on a separate host, but this guide assumes it to be installed on the **controller** node.

The *Midonet Neutron Plugin* replaces the ML2 Plugin and has to be installed on all three nodes (**controller**, **network**, **compute1**).

Hosts and Services

Controller Node (**controller**)

- General
 - Database (MariaDB)
 - Message Broker (RabbitMQ)
- OpenStack
 - Identity Service (Keystone)
 - Image Service (Glance)
 - Compute (Nova)

- Networking (Neutron)
- Dashboard (Horizon)
- MidoNet
 - API
 - CLI
 - Neutron Plugin
 - Network State Database (NSDB)
 - Network Topology (ZooKeeper)
 - Network State Information (Cassandra)

Gateway Node (network)

- OpenStack
 - Networking (Neutron)
 - DHCP Agent
 - Metadata Agent
- MidoNet
 - Agent (Midolman)
 - Neutron Plugin
 - Network State Database (NSDB)
 - Network Topology (ZooKeeper)
 - Network State Information (Cassandra)

Compute Node (compute1)

- OpenStack
 - Compute (Nova)
 - Networking (Neutron)
- MidoNet
 - Agent (Midolman)
 - Neutron Plugin
 - Network State Database (NSDB)
 - Network Topology (ZooKeeper)
 - Network State Information (Cassandra)

2. Basic Environment Configuration

Table of Contents

Networking Configuration	3
SELinux Configuration	3
Repository Configuration	3

Networking Configuration



Important

All hostnames must be resolvable, either via DNS or locally.

SELinux Configuration



Important

This guide assumes that SELinux (if installed) is either in permissive state or disabled.

To change the mode, execute the following command:

```
# setenforce Permissive
```

To permanently change the SELinux configuration, edit the `/etc/selinux/config` file accordingly:

```
SELINUX=permissive
```

Repository Configuration

Configure necessary software repositories and update installed packages.

1. Enable Red Hat base repository

```
# subscription-manager repos --enable=rhel-7-server-rpms
```

2. Enable additional Red Hat repositories

```
# subscription-manager repos --enable=rhel-7-server-extras-rpms  
# subscription-manager repos --enable=rhel-7-server-optional-rpms
```

3. Enable repository prioritization

```
# yum install yum-plugin-priorities
```

4. Enable EPEL repository

```
# yum install http://dl.fedoraproject.org/pub/epel/7/x86_64/e/epel-  
release-7-5.noarch.rpm
```

5. Enable RDO repository

```
# yum install http://rdo.fedorapeople.org/openstack-juno/rdo-release-juno.rpm
```

6. Enable DataStax repository

Create the `/etc/yum.repos.d/datastax.repo` file and edit it to contain the following:

```
# DataStax (Apache Cassandra)
[datastax]
name = DataStax Repo for Apache Cassandra
baseurl = http://rpm.datastax.com/community
enabled = 1
gpgcheck = 0
gpgkey = https://rpm.datastax.com/rpm/repo_key
```

7. Enable MidoNet repositories

Create the `/etc/yum.repos.d/midonet.repo` file and edit it to contain the following:

```
[midonet]
name=MidoNet
baseurl=http://repo.midonet.org/midonet/v2015.03/RHEL/7/stable/
enabled=1
gpgcheck=1
gpgkey=http://repo.midonet.org/RPM-GPG-KEY-midokura

[midonet-openstack-integration]
name=MidoNet OpenStack Integration
baseurl=http://repo.midonet.org/openstack-juno/RHEL/7/stable/
enabled=1
gpgcheck=1
gpgkey=http://repo.midonet.org/RPM-GPG-KEY-midokura

[midonet-misc]
name=MidoNet 3rd Party Tools and Libraries
baseurl=http://repo.midonet.org/misc/RHEL/7/misc/
enabled=1
gpgcheck=1
gpgkey=http://repo.midonet.org/RPM-GPG-KEY-midokura
```

8. Install available updates

```
# yum clean all
# yum upgrade
```

9. If necessary, reboot the system

```
# reboot
```


3. OpenStack Installation

Table of Contents

Identity Service (Keystone)	5
Compute Services (Nova)	5
Networking Service (Neutron)	6



Important

Follow the [OpenStack Juno Installation Guide for Red Hat Enterprise Linux 7](#), but **note the following differences**.

Identity Service (Keystone)



Important

Follow the OpenStack documentation's [Chapter 3. Add the Identity service](#) instructions, but **note the following additions**.

1. Create MidoNet API Service

As Keystone admin, execute the following command:

```
$ keystone service-create --name midonet --type midonet --description  
"MidoNet API Service"
```

2. Create MidoNet Administrative User

As Keystone admin, execute the following commands:

```
$ keystone user-create --name midonet --pass MIDONET_PASS --tenant  
service  
$ keystone user-role-add --user midonet --role admin --tenant service
```

Compute Services (Nova)



Important

Follow the OpenStack documentation's [Chapter 5. Add the Compute service](#) instructions, but **note the following differences**.

Controller Node



Note

Follow the OpenStack documentation's [Install and configure controller node](#) instructions as is.

Compute Node



Important

Follow the OpenStack documentation's [Install and configure a compute node](#) instructions, but **note the following additions**.

1. Configure libvirt

Edit the `/etc/libvirt/qemu.conf` file to contain the following:

```
user = "root"
group = "root"

cgroup_device_acl = [
    "/dev/null", "/dev/full", "/dev/zero",
    "/dev/random", "/dev/urandom",
    "/dev/ptmx", "/dev/kvm", "/dev/kqemu",
    "/dev/rtc", "/dev/hpet", "/dev/vfio/vfio",
    "/dev/net/tun"
]
```

2. Restart the libvirt service

```
# systemctl restart libvirtd.service
```

3. Install nova-rootwrap network filters

```
# yum install openstack-nova-network
# systemctl disable openstack-nova-network.service
```

4. Restart the Compute service

```
# systemctl restart openstack-nova-compute.service
```

Networking Service (Neutron)



Important

Follow the OpenStack documentation's [Chapter 6. OpenStack Networking \(neutron\)](#) instructions, but **note the following differences**.

Controller Node



Important

Follow the OpenStack documentation's [Install and configure controller node](#) instructions, but **note the following differences**.

1. To configure prerequisites

Apply as is.

2. To install the Networking components

Do **not** apply.

Instead, install the following packages:

```
# yum install openstack-neutron python-neutron-plugin-midonet
```

3. To configure the Networking server component

Do **not** apply step 'd. Enable the Modular Layer 2 (ML2) plug-in, router service, and overlapping IP addresses'.

Instead, edit the `/etc/neutron/neutron.conf` file and add the following key to the `[DEFAULT]` section:

```
[DEFAULT]
...
core_plugin = midonet.neutron.plugin.MidonetPluginV2
```

4. To configure the Modular Layer 2 (ML2) plug-in

Do **not** apply.

Instead, perform the following steps.

a. Create the directory for the MidoNet plugin:

```
# mkdir /etc/neutron/plugins/midonet
```

b. Create the `/etc/neutron/plugins/midonet/midonet.ini` file and edit it to contain the following:

```
[DATABASE]
sql_connection = mysql://neutron:NEUTRON_DBPASS@controller/neutron

[MIDONET]
# MidoNet API URL
midonet_uri = http://controller:8080/midonet-api
# MidoNet administrative user in Keystone
username = midonet
password = MIDONET_PASS
# MidoNet administrative user's tenant
project_id = service
```

c. Create a symbolic link to direct Neutron to the MidoNet configuration:

```
# ln -s /etc/neutron/plugins/midonet/midonet.ini /etc/neutron/plugin.ini
```

5. To configure Compute to use Networking

Apply as is.

6. To finalize installation

Do **not** apply.

Instead, perform the following steps.

a. Populate the database:

```
# su -s /bin/sh -c "neutron-db-manage --config-file /etc/neutron/
neutron.conf --config-file /etc/neutron/plugins/midonet/midonet.ini
upgrade junos" neutron
```

b. Restart the Compute services:

```
# systemctl restart openstack-nova-api.service openstack-nova-  
scheduler.service openstack-nova-conductor.service
```

- c. Start the Networking service and configure it to start when the system boots:

```
# systemctl enable neutron-server.service  
# systemctl start neutron-server.service
```

Gateway Node



Important

Follow the OpenStack documentation's [Install and configure network node](#) instructions, but **note the following differences**.

1. To configure prerequisites

Apply as is.

2. To install the Networking components

Do not apply.

Instead, install the following package:

```
# yum install openstack-neutron python-neutron-plugin-midonet
```

3. To configure the Networking common components

Do **not** apply step 'd. Enable the Modular Layer 2 (ML2) plug-in, router service, and overlapping IP addresses'.

Instead, edit the `/etc/neutron/neutron.conf` file and add the following key to the `[DEFAULT]` section:

```
[DEFAULT]  
...  
core_plugin = midonet.neutron.plugin.MidonetPluginV2
```

4. To configure the Modular Layer 2 (ML2) plug-in

Do not apply.

5. To configure the Layer-3 (L3) agent

Do not apply.

6. To configure the DHCP agent

Do not apply.

Instead, edit the `/etc/neutron/dhcp_agent.ini` file to contain the following:

```
[DEFAULT]  
interface_driver = neutron.agent.linux.interface.MidonetInterfaceDriver  
dhcp_driver = midonet.neutron.agent.midonet_driver.DhcpNoOpDriver  
use_namespaces = True  
enable_isolated_metadata = True  
  
[MIDONET]
```

```
# MidoNet API URL
midonet_uri = http://controller:8080/midonet-api
# MidoNet administrative user in Keystone
username = midonet
password = MIDONET_PASS
# MidoNet administrative user's tenant
project_id = service
```

7. To configure the metadata agent

Apply as is.

8. To configure the Open vSwitch (OVS) service

Do not apply.

9. To finalize the installation

Do not apply.

Instead, enable and start the following services:

```
# systemctl enable neutron-dhcp-agent.service neutron-metadata-agent.
service neutron-ovs-cleanup.service
# systemctl start neutron-dhcp-agent.service neutron-metadata-agent.
service
```

Compute Node



Important

Follow the OpenStack documentation's [Install and configure compute node](#) instructions, but **note the following differences**.

1. To configure prerequisites

Apply as is.

2. To install the Networking components

Do not apply.

Instead, install the following package:

```
# yum install openstack-neutron
```

3. To configure the Networking common components

Do **not** apply step 'd. Enable the Modular Layer 2 (ML2) plug-in, router service, and overlapping IP addresses'.

4. To configure the Modular Layer 2 (ML2) plug-in

Do not apply.

5. To configure the Open vSwitch (OVS) service

Do not apply.

6. To configure Compute to use Networking

Apply as is.

7. To finalize the installation

Do **not** apply.

Instead, restart the following service:

```
# systemctl restart openstack-nova-compute.service
```

4. MidoNet Installation

Table of Contents

NSDB Nodes	11
Controller Node	14
Midolman Installation	15
MidoNet Host Registration	15

NSDB Nodes

ZooKeeper Installation

1. Install ZooKeeper packages

```
# yum install zookeeper zkdump nmap-ncat
```

2. Configure ZooKeeper

a. Common Configuration

Edit the `/etc/zookeeper/zoo.cfg` file to contain the following:

```
server.1=controller:2888:3888
server.2=network:2888:3888
server.3=compute1:2888:3888
```

Create data directory:

```
# mkdir /var/lib/zookeeper/data
# chown zookeeper:zookeeper /var/lib/zookeeper/data
```

b. Node-specific Configuration

i. Controller Node

Create the `/var/lib/zookeeper/myid` file and edit it to contain the host's ID:

```
# echo 1 > /var/lib/zookeeper/data/myid
```

ii. Gateway Node

Create the `/var/lib/zookeeper/myid` file and edit it to contain the host's ID:

```
# echo 2 > /var/lib/zookeeper/data/myid
```

iii. Compute Node

Create the `/var/lib/zookeeper/myid` file and edit it to contain the host's ID:

```
# echo 3 > /var/lib/zookeeper/data/myid
```

3. Create Java Symlink

```
# mkdir -p /usr/java/default/bin/  
# ln -s /usr/lib/jvm/jre-1.7.0-openjdk/bin/java /usr/java/default/bin/  
java
```

4. Enable and start ZooKeeper

```
# systemctl enable zookeeper.service  
# systemctl start zookeeper.service
```

5. Verify ZooKeeper Operation

After installation of all nodes has been completed, verify that ZooKeeper is operating properly.

A basic check can be done by executing the `ruok` (Are you ok?) command on all nodes. This will reply with `imok` (I am ok.) if the server is running in a non-error state:

```
$ echo ruok | nc 127.0.0.1 2181  
imok
```

More detailed information can be requested with the `stat` command, which lists statistics about performance and connected clients:

```
$ echo stat | nc 127.0.0.1 2181  
Zookeeper version: 3.4.5--1, built on 06/10/2013 17:26 GMT  
Clients:  
/127.0.0.1:34768[0](queued=0,recved=1,sent=0)  
/192.0.2.1:49703[1](queued=0,recved=1053,sent=1053)  
  
Latency min/avg/max: 0/4/255  
Received: 1055  
Sent: 1054  
Connections: 2  
Outstanding: 0  
Zxid: 0x260000013d  
Mode: follower  
Node count: 3647
```

Cassandra Installation

1. Install Cassandra packages

```
# yum install dsc20-2.0.10-1  
# echo "exclude=dsc20 cassandra20" >> /etc/yum.conf
```

2. Configure Cassandra

a. Common Configuration

Edit the `/etc/cassandra/conf/cassandra.yaml` file to contain the following:

```
# The name of the cluster.  
cluster_name: 'midonet'  
  
...  
  
# Addresses of hosts that are deemed contact points.  
seed_provider:  
  - class_name: org.apache.cassandra.locator.SimpleSeedProvider  
    parameters:
```



```
- seeds: "controller,network,compute1"
```

b. Node-specific Configuration

i. Controller Node

Edit the `/etc/cassandra/conf/cassandra.yaml` file to contain the following:

```
# Address to bind to and tell other Cassandra nodes to connect to.
listen_address: controller

...

# The address to bind the Thrift RPC service.
rpc_address: controller
```

ii. Gateway Node

Edit the `/etc/cassandra/conf/cassandra.yaml` file to contain the following:

```
# Address to bind to and tell other Cassandra nodes to connect to.
listen_address: network

...

# The address to bind the Thrift RPC service.
rpc_address: network
```

iii. Compute Node

Edit the `/etc/cassandra/conf/cassandra.yaml` file to contain the following:

```
# Address to bind to and tell other Cassandra nodes to connect to.
listen_address: compute1

...

# The address to bind the Thrift RPC service.
rpc_address: compute1
```

3. Enable and start Cassandra

```
# systemctl enable cassandra.service
# systemctl start cassandra.service
```

4. Verify Cassandra Operation

After installation of all nodes has been completed, verify that Cassandra is operating properly.

A basic check can be done by executing the `nodetool status` command. This will reply with `UN` (Up / Normal) in the first column if the servers are running in a non-error state:

```
$ nodetool -host 127.0.0.1 status
[...]
Status=Up/Down
|/ State=Normal/Leaving/Joining/Moving
-- Address      Load          Tokens    Owns    Host ID
   Rack
```

```
UN 192.0.2.1 123.45 KB 256 33.3%  
11111111-2222-3333-4444-555555555555 rack1  
UN 192.0.2.2 234.56 KB 256 33.3%  
22222222-3333-4444-5555-666666666666 rack1  
UN 192.0.2.3 345.67 KB 256 33.4%  
33333333-4444-5555-6666-777777777777 rack1
```

Controller Node

MidoNet API Installation

1. Install MidoNet API package

```
# yum install midonet-api
```

2. Configure MidoNet API

Edit the `/usr/share/midonet-api/WEB-INF/web.xml` file to contain the following:

```
<context-param>  
  <param-name>rest_api-base_uri</param-name>  
  <param-value>http://controller:8080/midonet-api</param-value>  
</context-param>
```

```
<context-param>  
  <param-name>keystone-service_host</param-name>  
  <param-value>controller</param-value>  
</context-param>
```

```
<context-param>  
  <param-name>keystone-admin_token</param-name>  
  <param-value>ADMIN_TOKEN</param-value>  
</context-param>
```

```
<context-param>  
  <param-name>zookeeper-zookeeper_hosts</param-name>  
  <param-value>controller:2181,network:2181,compute1:2181</param-  
value>  
</context-param>
```

```
<context-param>  
  <param-name>midobrain-properties_file</param-name>  
  <param-value>/var/lib/tomcat/webapps/host_uuid.properties</param-  
value>  
</context-param>
```

3. Install Tomcat package

```
# yum install tomcat
```

4. Configure Tomcat's Maximum HTTP Header Size

Edit the `/etc/tomcat7/server.xml` file and adjust the maximum header size for the HTTP connector:

```
<Connector port="8080" protocol="HTTP/1.1"  
  connectionTimeout="20000"  
  URIEncoding="UTF-8"  
  redirectPort="8443"  
  maxHttpHeaderSize="65536" />
```

5. Configure MidoNet API context

Create the `/etc/tomcat/Catalina/localhost/midonet-api.xml` file and edit it to contain the following:

```
<Context
  path="/midonet-api"
  docBase="/usr/share/midonet-api"
  antiResourceLocking="false"
  privileged="true"
/>
```

6. Start Tomcat

```
# systemctl enable tomcat.service
# systemctl start tomcat.service
```

MidoNet CLI Installation

1. Install MidoNet CLI package

```
# yum install python-midonetclient
```

2. Configure MidoNet CLI

Create the `~/.midonetrc` file and edit it to contain the following:

```
[cli]
api_url = http://controller:8080/midonet-api
username = admin
password = ADMIN_PASS
project_id = admin
```

Midolman Installation

The Midolman agent shall be installed on all network and compute nodes.

1. Install Midolman package

```
# yum install midolman
```

2. Configure NSDB

Edit the `/etc/midolman/midolman.conf` file to contain the following:

```
[zookeeper]
zookeeper_hosts = controller:2181,network:2181,compute1:2181
...

[cassandra]
servers = controller,network,compute1
replication_factor = 3
cluster = midonet
```

3. Start Midolman

```
# systemctl start midolman.service
```

MidoNet Host Registration

1. Launch MidoNet CLI

```
$ midonet-cli  
midonet>
```

2. Create tunnel zone

MidoNet supports the Virtual Extensible LAN (VXLAN) and Generic Routing Encapsulation (GRE) protocols to communicate to other hosts within a tunnel zone.

To use the VXLAN protocol, create the tunnel zone with type 'vxlan':

```
midonet> tunnel-zone create name tz type vxlan  
tzone0
```

To use the GRE protocol, create the tunnel zone with type 'gre':

```
midonet> tunnel-zone create name tz type gre  
tzone0
```

3. Add hosts to tunnel zone

```
midonet> list tunnel-zone  
tzone tzone0 name tz type vxlan  
  
midonet> list host  
host host0 name network alive true  
host host1 name compute1 alive true  
  
midonet> tunnel-zone tzone0 add member host host0  
address ip_address_of_host0  
zone tzone0 host host0 address ip_address_of_host0  
  
midonet> tunnel-zone tzone0 add member host host1  
address ip_address_of_host1  
zone tzone0 host host1 address ip_address_of_host1
```

5. Further Steps

MidoNet installation and integration into OpenStack is completed.

You can now continue with the creation of initial networks in Neutron.



Note

Consult the **Operation Guide** for further instructions on operating MidoNet.