

# MidoNet クイック スタート ガイド

## RHEL 7 / Juno (OSP)

2015.06-SNAPSHOT (2015-10-16 12:02 UTC)

DRAFT



[docs.midonet.org](https://docs.midonet.org)



## 目次

はじめに .....	iv
表記規則 .....	iv
1. アーキテクチャ .....	1
ホストとサービス .....	2
2. 環境の基本構成 .....	4
ネットワークの構成 .....	4
SELinuxの構成 .....	4
レポジトリの構成 .....	4
3. OpenStackのインストール .....	6
アイデンティティサービス (Keystone) .....	6
コンピュートサービス (Nova) .....	6
ネットワーキングサービス (Neutron) .....	10
4. MidoNetのインストール .....	15
NSDBノード .....	15
コントローラノード .....	18
Midolman のインストール .....	20
MidoNetのホストの登録 .....	21
5. BGP アップリンク構成 .....	23
6. 高度な手順 .....	27





T - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT

T - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT

T - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT

- T - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT

T - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT

- T - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT

T - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT

- T - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT

## ゲートウェイノード (gateway1, gateway2)

- BGP Daemon (Quagga)
- MidoNet
  - エージェント (Midolman)





T - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT

T - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT

T - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT

T - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT

T - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT

T - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT

T - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT

T - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT

## 目次



## アイデンティティサービス (Keystone)



## コンピュータサービス (Nova)



## コントローラノード



---

6

#### 2. 8.3.2. Compute Serviceの認証を構成する

このまま適用します。

### 3. 8.3.3. Compute Serviceパッケージをインストールする

そのまま適用\*しないで\*ください。

代わりに、次のパッケージのみインストールします。

```
# yum install openstack-nova-api openstack-nova-conductor openstack-nova-scheduler  
python-cinderclient
```



注記

代わりに、openstack-nova-compute パッケージがコンピュータノードにインストールされます。

#### 4. 8.3.4. Compute ServiceのSSLの使用を構成する

このまま適用します。

#### 5.8.3.5. Compute Serviceを構成する

このまま適用します。

#### 6. 8.3.6. Compute Service Databaseを追加する

このまま適用します。

#### 7.8.3.7. コンピュータサービスを開始する

### a. 1. Message Bus Serviceの開始

適用 しないで ください。 コンピュートノードでのみ必要です。

## b. 2. Libvirt Serviceの開始

適用\*しないで\*ください。 コンピュートノードでのみ必要です。

### c. 3. API Serviceの開始

このまま適用します。

#### d. 4. スケジューラの開始

このまま適用します。

e. 5. コンダクタの開始

このまま適用します。

### f. 6. Compute Serviceの開始

適用\*しないで\*ください。 コンピュートノードでのみ必要です。

## 1. 追加の変更

a. メタデータプロキシを構成する

```
[neutron]
...
service_metadata_proxy = true
metadata_proxy_shared secret = METADATA_SECRET
```



Compute APIサービスを再開します。

```
# systemctl restart openstack-nova-api
```

## 重要

[glance]

```
...
host = controller
```

#### 6. 8.3.6. Compute Service Databaseを追加する

適用\*しないで\*ください。 コントローラ ノードでは実行済みです。

#### 7. 8. 3. 7. コンピュータサービスを開始する

### a. 1. Message Bus Serviceの開始

このまま適用します。

## b. 2. Libvirtd Serviceの開始

このまま適用します。

### c. 3. API Serviceの開始

適用\*しないで\*ください。 コントローラノードでのみ必要です。

#### d. 4. スケジューラの開始

適用\*しないで\*ください。 コントローラノードでのみ必要です。

e. 5. コンダクタの開始

適用\*しないで\*ください。 コントローラノードでのみ必要です。

#### f. 6. コンピュータサービスの開始

このまま適用します。

8. 代わりに、次の手順を実行します

### a. libvirtを構成する

/etc/libvirt/qemu.conf ファイルを修正して次を含めます。

```
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"  
]
```

b. libvirtdサービスを再開する

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

### c. nova-rootwrap ネットワークフィルタをインストールする

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

d. Computeサービスを再開する

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



代わりに、次のパッケージをインストールします。

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

4. 7.3.4. ファイアウォールを構成してOpenStackNetworkingのトラフィックを許可する

このまま適用します。

## コントローラノード



### 重要

Red Hatの文書の [7.4. Configure the Networking Service](#) の指示に従います。ただし、次の相違点に注意してください。

1. 7.4.1. OpenStack Networking Serviceの認証を構成する

このまま適用します。

2. 7.4.2. Networking ServiceのためにRabbitMQ Message Brokerを設定する

このまま適用します。

3. 7.4.3. OpenStack Networking Service Plug-inを設定する

適用\*しないで\*ください。代わりに、次の手順を実行します。

- a. /etc/neutron/neutron.conf ファイルを変更して、次のキーを [DEFAULT] セクションに追加します。

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



### 注記

構成ファイルの行の開始にスペースを残さないでください（これはすべての構成ファイルに適用されます）。

- b. MidoNetプラグインのディレクトリを作成します。

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

/etc/neutron/plugins/midonet/midonet.ini ファイルを作成し、修正して次を含めます。

+

```
[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 = services
```

a. NeutronをMidoNetの構成に転送するためのシンボリックリンクを作成します。

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

#### 4. 7. 4. 4. VXLANおよびGREトンネル

適用\*しないで\*ください。

#### 5. 7. 4. 5. Open vSwitchのトンネルを構成する

適用\*しないで\*ください。

#### 6.7.4.6. OpenStack NetworkingのSQLデータベース接続を構成する

適用\*しないで\*ください。

#### 7. \*7.4.7. OpenStack Networking Databaseを作成する

適用\*しないで\*ください。

代わりに、次のデータベースを作成します。

```
$ mysql -u root -p
CREATE DATABASE neutron;
GRANT ALL PRIVILEGES ON neutron.* TO 'neutron'@'localhost' IDENTIFIED BY
'NEUTRON_DBPASS';
GRANT ALL PRIVILEGES ON neutron.* TO 'neutron'@'%' IDENTIFIED BY 'NEUTRON_DBPASS';
FLUSH PRIVILEGES;
quit
```

Afterwards, run the `neutron-db-manage` command:

```
# neutron-db-manage ¥
--config-file /usr/share/neutron/neutron-dist.conf ¥
--config-file /etc/neutron/neutron.conf ¥
--config-file /etc/neutron/plugin.ini ¥
upgrade head
```

#### 8. 7.4.8. OpenStack Networking Serviceを開始する

このまま適用します。

## DHCP Agent



注記

Since MidoNet does not have the concept of a Network Node like with the default OpenStack networking plugin, the DHCP Agent is going to be installed on the Controller Node.



## 重要

Follow the Red Hat documentation's [7.5. Configure the DHCP Agent](#) instructions, but note the following differences and additions.

## 1. Configuring Authentication

Apply as is.





Do not apply.

## 1. Additional changes

Edit the `/etc/neutron/metadata_agent.ini` file to contain the following:

```
[DEFAULT]
[...]
nova_metadata_ip = controller
metadata_proxy_shared_secret = METADATA_SECRET
```



注記

Use the same METADATA\_SECRET as in the [Nova configuration](#).

### Restart the Metadata Agent:

```
# systemctl restart neutron-metadata-agent.service
```

T - DRAFT

`/var/lib/zookeeper/data/myid` ファイルを作成し、ホストのIDを含めます。

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

### 3. Java Symlinkを作成する

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

#### 4. ZooKeeperを有効にして開始する

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

## 5. ZooKeeperの動作を確認する

すべてのノードのインストールが完了したら、ZooKeeperが適切に動作するか確認します。

基本的な検査は、`ruok` (Are you ok?) コマンドを実行して行えます。サーバーがエラーのない状態で実行している場合は、最初の列に ``imok`` (I am ok) と返されます。

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

詳細情報が必要な場合は、`stat`コマンドを使用すると、パフォーマンスと接続しているクライアントの統計が一覧表示されます。

```
$ 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のインストール

## 1. Cassandraパッケージをインストールする

```
# yum install java-1.7.0-openjdk
# yum install dsc20
```

## 2. Cassandraを構成する

a. 共通の構成

/etc/cassandra/conf/cassandra.yaml ファイルを編集して、次のものを含めます。

```
# The name of the cluster.
cluster name: 'midonet'
```





```
<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>nsdb1:2181, nsdb2:2181, nsdb3:2181</param-value>
</context-param>
```

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

### 3. Tomcatパッケージをインストールする

```
# yum install tomcat
```

#### 4. TomcatHTTPヘッダの最大サイズを構成する

/etc/tomcat/server.xml ファイルを変更して、HTTPコネクタの最大サイズを調整します。

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

## 5. MidoNet APIのコンテキストを構成する

/etc/tomcat/Catalina/localhost/midonet-api.xml ファイルを作成し、修正して次を含めます。

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

## 6. Tomcatを開始する

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

## MidoNet CLIのインストール

## 1. MidoNet CLIパッケージをインストールする

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

## 2. MidoNet CLIを構成する

~/midonetrc ファイルを作成し、修正して次を含めます。

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







```
zone tzone0 host host1 address ip_address_of_host1

midonet> tunnel-zone tzone0 add member host host2 address ip_address_of_host2
zone tzone0 host host2 address ip_address_of_host2

midonet> tunnel-zone tzone0 add member host host3 address ip_address_of_host3
zone tzone0 host host3 address ip_address_of_host3
```

## 第5章 BGP アップリンク構成

MidoNet では、外部接続にボーダーゲートウェイプロトコル (BGP) を利用します。

BGP にはスケーラビリティと冗長性があるため、実稼動環境では BGP を使用することを強くお勧めします。

デモ環境や POC 環境では、代わりに静的ルーティングを使用できます。詳しくは、操作ガイドを参照してください。

こちらの手順では、次のサンプル環境を想定しています。

- ・ フローティング IP ネットワーク 1 個
  - ・ 192.0.2.0/24
- ・ MidoNet ゲートウェイノード 2 個
  - ・ gateway1、bgp1 に eth1 で接続
  - ・ gateway2、bgp2 に eth1 で接続
- ・ リモート BGP ピア 2 個
  - ・ bgp1、198.51.100.1、AS 64513
  - ・ bgp2、203.0.113.1、AS 64513
- ・ 対応する MidoNet BGP ピア
  - ・ 198.51.100.2、AS 64512
  - ・ 203.0.113.2、AS 64512

次の手順に従って、GBP アップリンクを構成してください。

## 1. Keystone admin テナント ID を特定する

keystone コマンドを使用して、Keystone admin テナント ID を特定します。

\$ keystone tenant-list		
id	name	enabled
12345678901234567890123456789012	admin	True

## 2. MidoNet CLI を起動し、MidoNet プロバイダルーターを検索する

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

MidoNet プロバイダルーターはテナントと関連付けられていないため、最初にアクティブテナントをクリア (cleart) する必要があります。

```
midonet-cli> cclear

midonet-cli> router list
router router0 name MidoNet Provider Router state up
router router1 name Tenant Router state up infiltrer chain0 outfilter chain1
```

この例の場合、MidoNet プロバイダルーターは router0 です。

### 3. admin テナントをロードする

構成をさらに続ける前に、admin テナントを設定 (sett) する必要があります。上記の Keystone から取得した ID を使用してください。

```
midonet-cli> sett 12345678901234567890123456789012
tenant_id: 12345678901234567890123456789012
```

#### 4. BGP セッション用の仮想ポートを作成する

リモート BGP ピアごとに、BGP 通信に使用するポートを MidoNet プロバイダルーター上に作成します。

```
midonet> router router0 add port address 198.51.100.2 net 198.51.100.0/30
router0:port0

midonet> router router0 add port address 203.0.113.2 net 203.0.113.0/30
router0:port1

midonet> router router0 port list
port port0 device router0 state up mac ac:ca:ba:11:11:11 address 198.51.100.2 net
198.51.100.0/30
port port1 device router0 state up mac ac:ca:ba:22:22:22 address 203.0.113.1 net
203.0.113.0/30
[...]
```

この例で作成されたポートは、port0 と port1 です。

## 5. 仮想ポートで BGP を構成する

```
midonet> router router0 port port0 add bgp local-AS 64512 peer-AS 64513
peer 198.51.100.1
router0:port0:bgp0

midonet> router router0 port port0 list bgp
bgp bgp0 local-AS 64512 peer-AS 64513 peer 198.51.100.1

midonet> router router0 port port1 add bgp local-AS 64512 peer-AS 64513
peer 203.0.113.1
router0:port1:bgp0

midonet> router router0 port port1 list bgp
bgp bgp0 local-AS 64512 peer-AS 64513 peer 203.0.113.1
```

## 6. Add routes to the remote BGP peers

In order to be able to establish connections to the remote BGP peers, corresponding routes have to be added.

```
midonet> router router0 route add src 0.0.0.0/0 dst 198.51.100.0/30 port router0:port0
type normal
router0:route0

midonet> router router0 route add src 0.0.0.0/0 dst 203.0.113.0/30 port router0:port1
type normal
router0:route1
```

## 7. BGPルートをアドバタイズする

ホストされている仮想マシンが外部接続できるようにするため、フローティング IP ネットワークを BGP ピアにアドバタイズする必要があります。

```
midonet> router router0 port port0 bgp bgp0 add route net 192.0.2.0/24
router0:port0:bgp0:ad-route0
```



```
port-group pgroup0 port router0:port1
```

```
midonet> port-group pgroup0 list member
port-group pgroup0 port router0:port0
port-group pgroup0 port router0:port1
```

## 第6章 高度な手順

OpenStackへのMidoNetのインストールと設定が完了しました。

これでNeutronの最初のネットワークの構築を続行できます。



注記

Midonetの運用に関する詳細については、「MidoNet 運用 ガイド」を参照してください。