

# MidoNet クイック スタート ガイド

## RHEL 7 / Kilo (RDO)

5.0-SNAPSHOT (2016-01-04 14:15 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) .....	7
4. MidoNetのインストール .....	11
NSDBノード .....	11
コントローラノード .....	14
Midolman のインストール .....	15
MidoNetのホストの登録 .....	17
5. ネットワークの初期設定 .....	19
6. BGP アップリンク構成 .....	20
7. 高度な手順 .....	23



---

1

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)

## 第2章 環境の基本構成

### 目次

ネットワークの構成 .....	4
SELinuxの構成 .....	4
レポジトリの構成 .....	4

### ネットワークの構成



#### 重要

すべてのホスト名はDNSまたはローカルで解決できる必要があります。

このガイドでは、OpenStackのドキュメントの下記にある指示に従っていることを前提としています。

[OpenStack Networking \(neutron\)](#)

### SELinuxの構成



#### 重要

このガイドはSELinux（インストール済みの場合）が`permissive`または`disabled`のいずれかのモードであることが前提となります。

モードを変更する場合は、次のコマンドを実行します。

```
# setenforce Permissive
```

SELinuxの構成を恒久的に変更する場合は、`/etc/selinux/config` ファイルを適宜変更してください。

```
SELINUX=permissive
```

### レポジトリの構成

必要なソフトウェアレポジトリを構成して、インストールしたパッケージを更新します。

1. Red Hatベースレポジトリを有効にする

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

2. 追加のRed Hatレポジトリを有効にする

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

3. レポジトリの優先順位付けを有効にする

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









```
[DEFAULT]
...
core_plugin = midonet.neutron.plugin_v2.MidonetPluginV2
```

代わりに、次の手順を実行します。

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

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

```
[MIDONET]
# MidoNet API URL
midonet_uri = http://controller:8181/midonet-api
# MidoNet administrative user in Keystone
username = midonet
password = MIDONET_PASS
# MidoNet administrative user's tenant
project_id = service
```

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

このまま適用します。

OpenStackのインストールガイドとともに、「[Load-Balancer-as-a-Service \(LBaaS\) を構成する](#)」[9]に記載されている、Load-Balancer-as-a-Service (LBaaS) を設定してください。

代わりに、次の手順を実行します。

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

```
# midonet-db-manage upgrade head
```



適用\*しないで\*ください。

- ### 3. ネットワーキング共通のコンポーネントを構成する場合

適用\*しないで\*ください。

- #### 4. モジュラーレイヤー2 (ML2) のプラグインを構成する場合

適用\*しないで\*ください。

- ## 5. Open vSwitch (OVS) サービスを構成する場合

適用\*しないで\*ください。

- ## 6. コンピュートでネットワーキングの使用を構成する場合

このまま適用します。

- ## 7. インストールを終了する場合

適用\*しないで\*ください。

代わりに、次のサービスを再開します。

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

## 第4章 MidoNetのインストール

## 目次

NSDB ノード .....	11
コントローラノード .....	14
Midolman のインストール .....	15
MidoNetのホストの登録 .....	17

## NSDB ノード

## ZooKeeperのインストール

## 1. ZooKeeperパッケージをインストールする

```
# yum install java-1.7.0-openjdk-headless
# yum install zookeeper zkdump nmap-ncat
```

## 2. ZooKeeperを構成する

### a. 共通の構成

/etc/zookeeper/zoo.cfg ファイルを編集して、次のものを含めます。

```
server.1=nsdb1:2888:3888
server.2=nsdb2:2888:3888
server.3=nsdb3:2888:3888
```

データのディレクトリを作成します。

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



## 重要

実稼働展開では、スナップショットのストレージをコミットログとは別のディスクに構成することをお勧めします。このように設定するには、`zoo.cfg` のパラメータ `dataDir` を別のディスクに変更します。

### b. ノード固有の構成

## i. NSDB ノード 1

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

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

## ii.NSDB ノード 2

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

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

## ii NSDB ノード 3

`/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-headless
# 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'
```





```
mkdir -p /var/run/cassandra
chown cassandra:cassandra /var/run/cassandra
su $CASSANDRA_OWNr -c "$CASSANDRA_PROG -p $pid_file" > $log_file 2>&1
retval=$?
```

[...]

#### 4. Cassandraを有効にして開始する

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

## 5. Cassandra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、Cassandraが適切に動作するか確認します。



## 重要

Cassandraデーモンの実行が失敗する場合、ログで「buffer overflow」のエラーメッセージが出たら、`/etc/hosts` ファイル内の `127.0.0.1` アドレスからホスト名へのマッピング・エントリを設定してみて（`hostname -i` の戻り値が `127.0.0.1` になりますように）、実行エラーの解決が出来るかもしれません。

基本的な検査は、`nodetool status` 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稼働している場合は、最初の列に UN (Up/Normal) と返されます。

```
$ 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
```

## コントローラノード

## MidoNet Clusterのインストール

## 1. MidoNet Clusterパッケージをインストールする

```
# yum install midonet-cluster
```

## 2. mn-conf をセットアップする

/etc/midonet/midonet.conf を編集し、mn-conf を ZooKeeper クラスタにポイントします。

```
[zookeeper]
zookeeper hosts = nsdb1:2181,nsdb2:2181,nsdb3:2181
```

### 3. NSDB へのアクセスを構成する

この手順は 1 回だけ実行してください。1 回実行すれば、MidoNet Cluster と Agent ノードで NSDB へのアクセスがセットアップされます。

次のコマンドを実行して、Zookeeper と Cassandra のサーバーアドレスにクラウド全体の値を設定します。

```
$ cat << EOF | mn-conf set -t default
zookeeper {
    zookeeper_hosts = "nsdb1:2181,nsdb2:2181,nsdb3:2181"
}
cassandra {
    servers = "nsdb1,nsdb2,nsdb3"
}
EOF
```

次のコマンドを実行して、Cassandra レプリケーション係数を設定します。

```
$ echo "cassandra.replication_factor : 3" | mn-conf set -t default
```

#### 4. Keystone へのアクセスを構成する

この手順は 1 回だけ実行してください。1 回実行すれば、MidoNet Cluster ノードで Keystone へのアクセスがセットアップされます。

```
$ cat << EOF | mn-conf set -t default
cluster.auth {
    provider_class = "org.midonet.cluster.auth.keystone.v2_0.KeystoneService"
    admin_role = "admin"
    keystone.tenant_name = "admin"
    keystone.admin_token = "ADMIN_TOKEN"
    keystone.host = controller
    keystone.port = 35357
}
EOF
```

## 5. MidoNet Cluster を起動する

```
# systemctl start midonet-cluster.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

~/midonetcrc ファイルを作成し、修正して次を含めます。

```
[cli]
api_url = http://controller:8181/midonet-api
username = admin
password = ADMIN_PASS
project_id = admin
```

## Midolman のインストール

仮想トポロジへのトラフィックの出入り口となるすべてのノードには、MidoNet Agent (Midolman) をインストールする必要があります。このガイドでは、gateway1、gateway2、compute1 の各ノードがこれに相当します。

## 1. Midolman パッケージをインストールする

```
# yum install java-1.8.0-openjdk-headless
# yum install midolman
```

## 2. mn-conf をセットアップする

/etc/midolman/midolman.conf を編集し、mn-conf を ZooKeeper クラスタにポイントします。

```
[zookeeper]
zookeeper_hosts = nsdb1:2181,nsdb2:2181,nsdb3:2181
```

### 3. リソース使用の設定

リソース使用を設定するために 各エージェントホスト で下記の手順を実行します。



## 重要

本番環境では large （大）テンプレートを強くお勧めします。

a. Midolman リソーステンプレート

Midolmanリソーステンプレートを設定するためには、次のコマンドを実行します。

```
$ mn-conf template-set -h local -t TEMPLATE_NAME
```

TEMPLATE\_NAME を以下のいずれかのテンプレートに置き換えます。

```
agent-compute-large
agent-compute-medium
agent-gateway-large
agent-gateway-medium
default
```

## b. Java Virtual Machine (JVM) リソーステンプレート

JVMリソーステンプレートを設定するためには、デフォルトの `/etc/midolman/midolman-env.sh` ファイルを以下のいずれかに置き換えます。

```
/etc/midolman/midolman-env.sh.compute.large
/etc/midolman/midolman-env.sh.compute.medium
/etc/midolman/midolman-env.sh.gateway.large
/etc/midolman/midolman-env.sh.gateway.medium
```

#### 4. すべてのエージェントで MidoNet Metadata Proxy へのアクセスを構成する

この手順は 1 回だけ実行してください。1 回実行すれば、すべての MidoNet Agent ノードで MidoNet Metadata Proxy がセットアップされます。

次のコマンドを実行して、クラウド全体の値を設定します。

```
$ echo "agent.openstack.metadata.nova_metadata_url : ¥"http://  
/nova_metadata_host:nova_metadata_port¥"" | mn-conf set -t default  
$ echo "agent.openstack.metadata.shared_secret : shared_secret" | mn-conf set -t  
default  
$ echo "agent.openstack.metadata.enabled : true" | mn-conf set -t default
```

nova\_metadata\_host、nova\_metadata\_port、shared\_secret は適切な値に 置きかえてください。これらの値は対応する Nova Metadata API の設定に マッチしている必要があります。

nova\_metadata\_host と nova\_metadata\_port には Nova Metadata API が 要求を受け付けるアドレスを指定します。 shared\_secret には nova.conf の "neutron"



```
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 ネットワークの初期設定



### 重要

OpenStack文書の [Create initial networks](#) 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. 外部ネットワークの作成

外部ネットワークは下記のコマンドで作成します。

```
$ neutron net-create ext-net --router:external
```

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

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

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

T - 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

- T - 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

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

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

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



### 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 set asn 64512
midonet> router router0 add bgp-peer asn 64513 address 198.51.100.1
router0:peer0

midonet> router router0 list bgp-peer
peer peer0 asn 64513 address 198.51.100.1

midonet> router router0 add bgp-peer asn 64514 address 203.0.113.1
router0:peer1

midonet> router router0 list bgp-peer
peer peer0 asn 64513 address 198.51.100.1
peer peer1 asn 64514 address 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 add bgp-network net 192.0.2.0/24
router0:net0
```

```
midonet> router router0 list bgp-network
net net0 net 192.0.2.0/24
```

## 8. 仮想ポートを物理ネットワークインターフェースにバインドする

MidoNet プロバイダルーターの仮想ポートをゲートウェイノードの物理ネットワークインターフェースにバインドします。



## 重要

物理インターフェースの状態が UP になっていて、IP アドレスが割り当てられていないことを確認してください。

a. MidoNet ホストをリストし、ゲートウェイノードを検索します。

```
midonet> host list
host host0 name gateway1 alive true
host host1 name gateway2 alive true
[...]
```

この例のホストは host0 と host1 です。

b. ゲートウェイノードの物理インターフェースをリストします。

```
midonet> host host0 list interface
[...]
iface eth1 host_id host0 status 3 addresses [] mac 01:02:03:04:05:06 mtu 1500 type
Physical endpoint PHYSICAL
[...]

midonet> host host1 list interface
[...]
iface eth1 host_id host0 status 3 addresses [] mac 06:05:04:03:02:01 mtu 1500 type
Physical endpoint PHYSICAL
[...]
```

c. 物理ホストインターフェースを MidoNet プロバイダルーターの仮想ポートにバインドします。

```
midonet> host host0 add binding port router0:port0 interface eth1
host host0 interface eth1 port router0:port0

midonet> host host1 add binding port router0:port1 interface eth1
host host1 interface eth1 port router0:port1
```

d. ステートフルポートグループを構成します。

```
midonet-cli> port-group create name uplink-spg stateful true
pgroup0
```

e. ポートをポートグループに追加します。

```
midonet> port-group pgroup0 add member port router0:port0
port-group pgroup0 port router0:port0

midonet> port-group pgroup0 add member port router0:port1
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7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。



### 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。