



OpenStack Neutron ML2 用 Cisco ACI プラグインのインストール

この章の内容は、次のとおりです。

- [前提条件, 1 ページ](#)
- [OpenStack Neutron Modular Layer 2 用 Cisco ACI プラグインをインストールするためのワークフロー, 2 ページ](#)
- [設置場所の準備, 3 ページ](#)
- [サーバの準備, 3 ページ](#)
- [OpenStack 修正のインストール, 5 ページ](#)
- [ホストエージェントおよびサービスエージェントの起動, 11 ページ](#)
- [外部接続の設定, 12 ページ](#)

前提条件

開始するには、次の前提条件を満たしている必要があります。

- Red Hat Linux、OpenStack、および Application Policy Infrastructure Controller (APIC) の十分な知識がある。
- Cisco Application Centric Infrastructure (ACI) ファブリックが機能している。
- サーバまたは仮想マシンで Red Hat または CentOS バージョンを実行している (リポジトリ「サーバ」)。
- コントローラまたはネットワーク ノード、およびコンピューティング ノードが ACI リーフスイッチに接続されている (ネットワーク ノードとコントローラが同じノード上にある場合、本マニュアルで使われる「ネットワーク ノード」はコントローラを指します)。

- OpenStack の RPM ディストリビューション (RDO) または Red Hat 7 以降の Red Hat OpenStack (RHOS) 6 (Juno) で EPEL がイネーブルになっており、可能であればデモ ネットワークがプロビジョニングされていない (EPEL をイネーブルにしてデモ ネットワークをディセーブルにするには、応答ファイルに次の行を追加します)。

```
CONFIG_USE_EPEL=y  
CONFIG_PROVISION_DEMO=n
```

- OpenStack にネットワーク/ルータおよび仮想マシンが作成されていない (OpenStack にある既存のネットワーク/ルータおよび仮想マシンは削除してください)。

OpenStack Neutron Modular Layer 2 用 Cisco ACI プラグインをインストールするためのワークフロー

次のワークフローを使用して、OpenStack Neutron Modular Layer 2 (ML2) 用 Cisco ACI プラグインをインストールします。

- 1 設置場所を準備します。
[設置場所の準備, \(3 ページ\)](#) を参照してください。
- 2 サーバを準備します。
[サーバの準備, \(3 ページ\)](#) を参照してください。
- 3 OpenStack 修正をインストールします。
[OpenStack 修正のインストール, \(5 ページ\)](#) を参照してください。
- 4 OpFlex エージェントと Application Policy Infrastructure Controller (APIC) OVS エージェントをインストールします。
[OpFlex エージェントおよび APIC OVS エージェントのインストール, \(9 ページ\)](#) を参照してください。
- 5 OpFlex を設定します。
[OpFlex の設定, \(9 ページ\)](#) を参照してください。
- 6 ホストエージェントとサービスエージェントを起動します。
[ホストエージェントおよびサービスエージェントの起動, \(11 ページ\)](#) を参照してください。
- 7 次の手順を実行して、コンピューティング ノードをすべて設定します。
 - a [サーバの準備, \(3 ページ\)](#)
 - b [OpFlex エージェントおよび APIC OVS エージェントのインストール, \(9 ページ\)](#)
 - c [OpFlex の設定, \(9 ページ\)](#)
- 8 外部接続を設定します。
[外部接続の設定, \(12 ページ\)](#) を参照してください。

設置場所の準備

YUM リポジトリを作成して、Red Hat OpenStack プラットフォームをインストールする場所を準備する必要があります。

手順

-
- ステップ 1** cisco.com から apic-m12 および opflex RPM をダウンロードします。
- ステップ 2** httpd サービスをインストールして起動します。
- ```
yum install httpd
service httpd start
```
- ステップ 3** RPM を /var/www/html ディレクトリに移動します。
- ```
# mv apic_m12 opflex /var/www/html
```
- ステップ 4** リポジトリを作成します。たとえば、apic_m12 パッケージが apic_m12 ディレクトリにあり、opflex パッケージが opflex ディレクトリにある場合は、次のコマンドを使用します。
- ```
createrepo /var/www/html/opflex
createrepo /var/www/html/apic_m12
```
- 

## サーバの準備

ネットワーク ノードとしても機能するコントローラを含む、すべてのサーバについて次の手順を実行します。この手順の設定例では、次の内容を前提とします。

- eth1 はリーフに接続されます。
- Application Policy Infrastructure Controller (APIC) はデフォルトの VLAN 4093 (インフラ VLAN) で設定されます。この番号は、Application Policy Infrastructure Controller に設定されている VLAN の番号で置き換えてください。

### 手順

- 
- ステップ 1** net-tools をインストールし、ifconfig および route コマンドを有効にします。
- ```
# yum install net-tools
```
- ステップ 2** wget をインストールします。
- ```
yum install wget
```
- ステップ 3** /etc/yum.repos.d/opflex.repo コンフィギュレーションファイルを編集して次の行を追加することで、パッケージの YUM リポジトリを有効にします。
- ```
[opflex]
name=opflex repo
```

```

baseurl=http://172.29.172.13/opflex
failovermethod=priority
enabled=1
gpgcheck=0
[ml2-apic]
name=opflex repo
baseurl=http://172.29.172.13/ml2-apic
failovermethod=priority
enabled=1
gpgcheck=0

```

この変更によって opflex および apic_ml2 リポジトリを追加することができます。

ステップ 4 リポジトリが正常に機能することを確認します。

```
# yum makecache
```

エラーがなければ、リポジトリは正常に機能しています。エラーが発生した場合は、先に進む前に問題をトラブルシューティングしてください。

ステップ 5 OpenStack 設定のネットワーク ノードおよびコンピューティング ノードを、ダイレクト モードまたは VPC/ボンド モードでリーフ スイッチに接続します。

ステップ 6 APIC のインフラ VLAN タグ付きサブインターフェイスをサーバで定義します。これはリーフに接続されているインターフェイスのサブインターフェイスです。

ステップ 7 インターフェイス コンフィギュレーション ファイル `/etc/sysconfig/network-scripts/ifcfg-eth1` に次の行を追加することで、メイン インターフェイス (eth1) の MTU を 1600 に設定します。

```
MTU=1600
```

ステップ 8 eth1 インターフェイスにバウンスします。

```
# ifdown eth1
# ifup eth1
```

ステップ 9 次の内容を含む `/etc/sysconfig/network-scripts/ifcfg-eth1.4093` ファイルを作成して、VLAN 4093 の VLAN サブインターフェイスを設定します。

```

PERSISTENT_DHCLIENT=1
DHCPRELEASE=1
DEVICE=eth1.4093
ONBOOT=yes
PEERDNS=yes
NM_CONTROLLED=no
HWADDR=<mac address of eth1, or self-derived; see note below>
TYPE=Ethernet
BOOTPROTO=dhcp
VLAN=yes
ONPARENT=yes
MTU=1600

```

(注) LLDP フレームのスイッチへの送信に LLDP 設定を使用するには、親インターフェイスと同じ MAC アドレスは使用できません。LLDP 転送を有効にした場合に未定義の動作を引き起こす原因となるため、親インターフェイスの MAC アドレスは使用しないでください。

eth1 インターフェイスでは、APIC インフラストラクチャ ネットワークの DHCP アドレスを要求します。サーバが正しい DHCP オプションを提供するためには、このインターフェイスの dhclient 設定が必要です。

ステップ 10 次の内容を含む /etc/dhcp/dhclient-eth1.4093.conf ファイルを作成します。

```
send dhcp-client-identifier 01:<mac-address of .4093 interface>;
request subnet-mask, domain-name, domain-name-servers, host-name;
send host-name <server-host-name>;

option rfc3442-classless-static-routes code 121 = array of unsigned integer 8;
option ms-classless-static-routes code 249 = array of unsigned integer 8;
option wpad code 252 = string;

also request rfc3442-classless-static-routes;
also request ms-classless-static-routes;
also request static-routes;
also request wpad;
also request ntp-servers;
```

ステップ 11 次の内容を含む /etc/sysconfig/network-scripts/route-eth1.4093 ファイルを作成して、マルチキャストルートのサブインターフェイスを設定します。

```
ADDRESS0=224.0.0.0
NETMASK0=240.0.0.0
GATEWAY0=0.0.0.0
METRIC0=1000
```

ステップ 12 サーバで LLDPPD を有効にします。LLDPD パッケージをインストールします。

```
# cd /etc/yum.repos.d/
# wget http://download.opensuse.org/repositories/home:vbernat/RHEL_7/home:vbernat.repo
# yum install lldpd
```

ステップ 13 スイッチへの LLDP フレームの送信に問題があるため、/etc/sysconfig/lldpd ファイルの LLDP_OPTIONS パラメータに "-r" を付加してフレームをブロックします。

```
LLDPD_OPTIONS="-r"
```

何らかの理由で LLDP 転送が必要な場合は、APIC のインフラ VLAN サブインターフェイスが親インターフェイスの MAC アドレスを継承しないことを確認し、この手順は省略してください。

ステップ 14 LLDPPD を有効にして起動します。

```
# service lldpd start
# systemctl enable lldpd
```

OpenStack 修正のインストール

OpenStack コントローラまたはネットワーク ノードに、追加の OpenStack 修正パッケージをインストールする必要があります。パッケージには EPEL リポジトリが必要です。EPEL が有効な状態で OpenStack をインストールしなかった場合は、EPEL を手動で有効にする必要があります。

- eth1 はリーフに接続されます。

- Application Policy Infrastructure Controller (APIC) はデフォルトの VLAN 4093 (インフラ VLAN) で設定されます。この番号は、Application Policy Infrastructure Controller に設定されている VLAN の番号で置き換えてください。

手順

- ステップ 1** 次のパッケージをインストールします。
- ```
yum install python-pip
yum install python-pbr
yum install neutron-opflex-agent apicapi neutron-ml2-driver-apic
```
- ステップ 2** oslo-concurrency との依存関係の欠落に関するエラーが表示された場合は、/etc/yum.repos.d/epel-testing.repo ファイルを修正して [epel-testing] セクションに「enabled=1」を設定することで、epel-testing リポジトリを有効にしてください。
- ```
[epel-testing]
name=Extra Packages for Enterprise Linux 7 - Testing - $basearch
#baseurl=http://download.fedoraproject.org/pub/epel/testing/7/$basearch
mirrorlist=https://mirrors.fedoraproject.org/metalink?repo=testing-epel7&arch=$basearch
failovermethod=priority
enabled=1
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-EPEL-7
```
- (注) epel-testing に独自のリポジトリ ファイルが存在することがあります。この場合、epel-testing を有効にするには、/etc/yum.repos.d/epel-testing.repo ファイルの編集が必要な可能性があります。
- ステップ 3** /etc/neutron/neutron.conf ファイルを編集して service_plugins パラメータを更新します。
- ```
service_plugins = cisco_apic_l3,lbaas,metering
```
- ステップ 4** /etc/neutron/plugins/ml2/ml2\_conf.ini ファイルを編集して、mechanism\_drivers、type\_drivers、および tenant\_network\_type パラメータを更新します。
- ```
mechanism_drivers = cisco_apic_ml2
type_drivers = opflex,local,flat,vlan,gre,vxlan
tenant_network_types = opflex
```
- a) 「network_vlan_ranges」を含むすべての行をコメントアウトします。
- ステップ 5** /etc/neutron/dhcp_agent.ini ファイルを編集して dhcp_driver パラメータを更新します。
- ```
dhcp_driver = apic_ml2.neutron.agent.linux.apic_dhcp.ApicDnsmasq
```
- a) DEFAULT セクションに次のオプションが存在することを確認してください。
- ```
[DEFAULT]
ovs_integration_bridge = br-int
enable_isolated_metadata = True
```
- ステップ 6** DHCP エージェントを再起動します。
- ```
systemctl restart neutron-dhcp-agent.service
```

**ステップ 7** /etc/neutron/plugins/ml2/ml2\_conf\_cisco.ini ファイルを編集して次のパラメータを設定します。

```
[DEFAULT]
apic_system_id = <any string>

[opflex]
networks = '*'

[m12_cisco_apic]
apic_hosts = comma-separated IP address list of APIC controllers
 10.1.2.1 or 10.1.2.1,10.1.2.2,10.1.2.3
apic_username = Username for APIC controller
apic_password = Password for 'apic_username'
apic_domain_name = <string>
use_vmm = True
apic_use_ssl = True
apic_clear_node_profiles = True
enable_aci_routing = True
enable_arp_flooding = True
apic_name_mapping = use_name (If there are multiple OpenStack instances per
 fabric, use use_uuid)
apic_entity_profile = <any string>
apic_vmm_domain = <any string>
apic_app_profile_name = <any string>
apic_function_profile = <any string>

apic_sync_interval = 0
apic_agent_report_interval = 30
apic_agent_poll_interval = 2
apic_provision_infra = True
apic_provision_hostlinks = True (Set to False if the interface profiles, switch
 profiles, policy groups are pre-existing on the APIC)

apic_vpc_pairs = leave blank if not VPC, otherwise
 101:102,103:104 (101,102,103 and 104 are switch ids as
 registered in APIC and pc pair between switch ids 101 and
 102, 103 and 104)

[group_policy]
policy_drivers = implicit_policy,apic

[group_policy_implicit_policy]
default_ip_pool = 192.168.0.0/16
```

**ステップ 8** コンピューティングノードが接続されたスイッチごとに、[apic\_switch:<switch id>] という名前のセクションを作成し、このセクションにコンピューティングノードおよび対応するスイッチインターフェイスをリストします。

VPC を使用していない場合は、次の形式を使用してください。

```
<compute_fqdn>=<switch port>
```

例：

```
[apic_switch:201]
fab102-compute-1.noironetworks.com=1/5
fab102-compute-2.noironetworks.com=1/6
```

VPC を使用している場合は、次の形式を使用してください。

```
<compute_fqdn>=vpc-<mod>-<port>/<vpc_policy_group_name>
```

次の例は、以下のように設定されます。

- compute-1 と compute-2 は、VPC を使用してファブリックに接続されます。
- compute-1 には、スイッチ 101 のポート 1/17 とスイッチ 102 のポート 1/17 にメンバーが接続されている bond0 があります。VPC のファブリック上のポリシー グループ名は「bundle-101-1-17-and-102-1-17」です。
- compute-2 には、スイッチ 103 のポート 1/18 とスイッチ 104 のポート 1/18 にメンバーが接続されている bond0 があります。VPC のファブリック上のポリシー グループ名は「bundle-102-1-18-and-102-1-18」です。

/etc/neutron/plugins/ml2/ml2\_conf\_cisco.ini ファイルのエントリは次のとおりです。

```
[apic_switch:101]
compute-1.noiro.lab=vpc-1-17/bundle-101-1-17-and-102-1-17
[apic_switch:102]
compute-1.noiro.lab=vpc-1-17/bundle-101-1-17-and-102-1-17
[apic_switch:103]
compute-2.noiro.lab=vpc-1-18/bundle-101-1-18-and-102-1-18
[apic_switch:104]
compute-2.noiro.lab=vpc-1-18/bundle-101-1-18-and-102-1-18
```

**ステップ 9** /usr/lib/systemd/system/neutron-server.service ファイルを編集して ExecStart 行に「--config-file /etc/neutron/plugins/ml2/ml2\_conf\_cisco\_apic.ini」を追加することで、/etc/neutron/plugins/ml2/ml2\_conf\_cisco.ini ファイルを読み込む neutron-server プロセスを設定します。追加された行は次のようになります

```
ExecStart=/usr/bin/neutron-server
--config-file /usr/share/neutron/neutron-dist.conf
--config-file /etc/neutron/neutron.conf
--config-file /etc/neutron/plugin.ini
--config-file /etc/neutron/plugins/ml2/ml2_conf_cisco.ini
--log-file /var/log/neutron/server.log
```

**ステップ 10** サービス コンフィギュレーションをリロードします。

```
systemctl daemon-reload
```

**ステップ 11** ネットワーク ノード（ネットワーク コントローラがない場合はコントローラ ノード）で neutron-server サービスを再起動します。

```
service neutron-server restart
```

## OpFlex エージェントおよび APIC OVS エージェントのインストール

ネットワークノード、またはコントローラノード（ネットワークコントローラがない場合）で、OpFlex エージェントと Application Policy Infrastructure Controller (APIC) OVS エージェントをインストールします。

### 手順

**ステップ 1** OpFlex エージェントと APIC OVS エージェントをインストールします。

```
yum install neutron-opflex-agent
yum install agent-ovs
```

**ステップ 2** /etc/neutron/plugins/openvswitch/ovs\_neutron\_plugin.ini ファイルを編集して次の行を追加します。

```
[ovs]
enable_tunneling = False
integration_bridge = br-int
local_ip = <ip address of openstack controller>
```

a) 次のオプションをコメントアウトまたは削除します。

- tunnel\_bridge
- local\_ip
- vxlan\_udp\_port
- tunnel\_types

## OpFlex の設定

ネットワークノード、またはコントローラノード（ネットワークコントローラがない場合）で、OpFlex エージェントをインストールした後に設定を行う必要があります。

### 手順

**ステップ 1** デフォルトの Open vSwitch エージェントをディセーブルにします。

```
systemctl disable neutron-openvswitch-agent
service neutron-openvswitch-agent stop
```

**ステップ 2** シスコの修正 Open vSwitch パッケージをインストールします。

```
yum install openvswitch-gbp
```

**ステップ 3** インストールした Open vSwitch パッケージが、前に存在したカーネルバージョンと異なる場合は、サーバをリブートします。

**ステップ 4** Open vSwitch モジュールをリロードします。

```
rmmmod openvswitch
modprobe openvswitch
```

**ステップ 5** OpFlex エージェントのコンフィギュレーションファイルを設定します。OpFlex エージェントは、/etc/opflex-agent-ovs/opflex-agent-ovs.conf ファイルから設定を読み取ります。設定は JSON 形式です。次のサンプルファイルで、<apic\_system\_id>を以前の ini ファイルの設定値に変更し、<hostname of the system>を適切なホスト名に変更します。インフラ VLAN と VLAN インターフェイスおよびブリッジの名前が正しいことを確認します。

```
{
 "log": {
 "level": "debug2"
 },

 "opflex": {
 "domain": "comp/prov-OpenStack/ctrlr-[<apic_system_id>]-<apic_system_id>/sw-InsiemeLSOid",

 "name": "<hostname of this system>",
 "peers": [
 {"hostname": "10.0.0.30", "port": "8009"}
],
 "ssl": {
 "mode": "enabled",
 "ca-store": "/etc/ssl/certs/"
 }
 },

 "endpoint-sources": {
 "filesystem": ["/var/lib/opflex-agent-ovs/endpoints"]
 },

 "renderers": {
 "stitched-mode": {
 "ovs-bridge-name": "br-int",

 "encap": {
 "vxlan": {
 "encap-iface": "br-int_vxlan0",
 "uplink-iface": "eth1.4093",
 "uplink-vlan": 4093,
 "remote-ip": "10.0.0.32",
 "remote-port": 8472
 }
 }
 },
 "forwarding": {
 "virtual-router": {
 "enabled": true,
 "mac": "00:22:bd:f8:19:ff",
 "ipv6": {
 "router-advertisement": "false"
 }
 }
 }
 }
}
```



## 外部接続の設定

コンピューティング ノードの設定後に、外部接続を設定する必要があります。

### 手順

- ステップ 1** ネクストホップ IP アドレスが 1.101.2.1 の場合は、Cisco Application Centric Infrastructure (ACI) L3Out ルータの IP アドレスを 1.101.2.2/30、ホストの SNAT プールを 2.101.2.1/24 に設定して、コントローラの設定にこれらの IP アドレスが含まれていることを確認します。

```
[apic_external_network:Datacenter-Out]
router_id=1.0.0.2
switch=101
port=1/2
```

```
encap=vlan-1011
gateway_ip=1.101.2.1
cidr_exposed=1.101.2.2/30
host_pool_cidr = 2.101.2.1/24
```

- a) 既存のレイヤ3アウトと既存の外部エンドポイントグループを代わりに使用する場合は、コントローラの設定に既存の情報が含まれていることを確認します。

```
[apic_external_network:Datacenter-Out]
preexisting=True
external_epg=Datacenter-Out-Epg
host_pool_cidr = 2.101.2.1/24
```

この例では、「Datacenter-Out-Epg」が既存の外部エンドポイントグループです。

- ステップ 2** すべてのコンピューティング ノードおよびネットワーク ノードで、SNAT を必要とする外部セグメントに「opflex\_external\_segment」という名前の新しいセクションを作成して次の行を追加し、SNAT アドレス割り当て/ゲートウェイが設定に含まれていることを確認します。

```
ip_gateway = <same as host_pool_cidr on the controller>
ip_address_range <start,end range from host_pool_cidr on the controller for the IP addresses
to be used for SNAT from this host>
```

例：

```
[opflex_external_segment:Datacenter-Out]
ip_gateway = 2.101.2.1/24
ip_address_range = 2.101.2.2
```