Cisco IOS XRv 9000 ルータを KVM 環境にインストールする

KVM ハイパーバイザに Cisco IOS XRv 9000 ルータをインストールするには、以下のファイルタイプが必要です。

- .qcow2：KVM OpenStack 環境でソフトウェア イメージを起動するために使用されます。qcow2 ディスク イメージには、プレインストールされている Cisco IOS XRv 9000 ルータのインスタンスがあります。
- .iso と .qcow2：Virsh アプリケーションを使用して Cisco IOS XRv 9000 ルータ VM を手動で作成するために使用されます。また、Virsh コマンドを使用して KVM 環境で Cisco IOS XRv 9000 ルータを起動するために使用されるサンプル XML 設定が含まれる virsh.xml ファイルも必要です。

KVM のインストール要件

Cisco IOS XRv 9000 ルータは、Kernel Virtual Machine（KVM）を使用する Ubuntu および Red Hat のディストリビューションでサポートされます。KVM に Cisco IOS XRv 9000 ルータをインストールするには、.iso ファイルまたは .qcow2 イメージのどちらかを使用した VM の作成とインストールが必要になります。VM の起動には、KVM コマンドライン、Virsh または OpenStack を使用します。

KVM のインストール要件については、Cisco IOS XRv 9000 ルータの最新のリリース ノートを参照してください。

リリース ノートのリンクについては、表 1を参照してください。

（注）

- リリース 6.0 移行では、45 GB の最小 VM ハード ディスク サイズがサポートされています。
- **du** コマンドを使用すると、仮想ディスク イメージによって使用される実際のディスク容量を確認できます。
- PCIe パススルー インタフェースを VM で機能させるには、grub 設定でオプションの **intel_iommu=on** コマンドを設定する必要があります。
- 組み込みデータプレーンを機能させるには、CPU フラグを VM に渡す必要があります。また、このフラグに **sse4_2** フラグが含まれている必要があります。

OpenStack での KVM サポートの詳細については、「OpenStack での KVM サポート」の項を参照してください。

KVM コマンドラインを使用した Cisco IOS XRv 9000 ルータのインストール

この手順は、Cisco IOS XRv 9000 ルータ用の VM を手動で作成するための一般的なガイドラインです。実行する必要のある正確な手順は、KVM の環境と設定の特性に応じて異なることがあります。詳細については、Red Hat Linux または Ubuntu のドキュメンテーションを参照してください。

この手順では、3 つのトラフィック インターフェイスと 3 つの必須インターフェイス（1 つは XR 管理用、2 つは予約済み）のある ISO ブート設定を作成する方法を説明します。

1. `/usr/bin/kvm \`

   KVM を起動します。

2. `-smbios type=1,manufacturer="cisco",product="Cisco IOS XRv 9000",uuid=97fc351b-431d-4cf2-9c01-43c283faf2a3 \`

   VM インスタンスのユニバーサル固有 ID（UUID）を設定します。UUID は、VM インスタンスを特定するライセンシングの一部として使用されます。

3. `-cpu host \`

   ホスト CPU フラグをゲストに渡します。

4. ```
   -drive file=/home/username/bnbMay13/workdir-username/disk1.raw,if=virtio,media=disk,index=1 \
   -drive file=/home/username/bnbMay13/workdir-username/xrv9k-fullk9-x.iso.baked,media=cdrom,index=2
    \
   ```

   ハードディスクを空にし、ISO を起動します。
   ```

5. `-m 16384 \`

   メモリを作成します。

6. `-smp cores=4,threads=1,sockets=1 \`

   4 つの仮想 CPU と 1 つのソケットを作成します。

7. `-enable-kvm \`

   ハードウェア アクセラレーションを有効化します。

8. `-display none \`

   コンソール アクセスのためにシリアル ポートを使用する場合、VGA コンソールをエミュレートします。このステップは推奨されています。

9. `-rtc base=utc \`

   リアルタイム クロック（RTC）を設定します。

10.
```
-netdev tap,id=host1,ifname=usernameLx1,script=no,downscript=no \
-netdev tap,id=host2,ifname=usernameLx2,script=no,downscript=no \
-netdev tap,id=host3,ifname=usernameLx3,script=no,downscript=no \
-device virtio-net-pci,romfile=,netdev=host1,id=host1,mac=52:46:84:57:A0:DA \
-device virtio-net-pci,romfile=,netdev=host2,id=host2,mac=52:46:C4:F4:36:0F \
-device virtio-net-pci,romfile=,netdev=host3,id=host3,mac=52:46:A5:C0:D0:C5 \
```

   3 つの NIC を作成します。最初のものは XR 管理インターフェイスにマッピングされ、2 番目と 3 番目は予約済みです。

11.
```
-netdev tap,id=data1,ifname=usernameXr1,script=no,downscript=no \
-netdev tap,id=data2,ifname=usernameXr2,script=no,downscript=no \
-netdev tap,id=data3,ifname=usernameXr3,script=no,downscript=no \
-device e1000,romfile=,netdev=data1,id=data1,mac=52:46:87:18:62:DF \
-device e1000,romfile=,netdev=data2,id=data2,mac=52:46:32:02:90:6F \
-device e1000,romfile=,netdev=data3,id=data3,mac=52:46:34:93:52:1F \
```

   4 番目から 11 番目の NIC は、トラフィック ポートにマッピングされます。最低 1 つのトラフィック インターフェイスが推奨されています。

12.
```
-serial telnet:0.0.0.0:10621,nowait,server \
-serial telnet:0.0.0.0:14713,nowait,server \
-serial telnet:0.0.0.0:18090,nowait,server \
-serial telnet:0.0.0.0:17181,nowait,server \
```

   コンソールにアクセスするための 4 つのシリアル ポートを作成します。最初のシリアル ポートは XR コンソールにマッピングされます。詳細については、「コンソール マッピング」の項を参照してください。最低 1 つのシリアル ポートが推奨されています。

   シリアル コンソール アクセスの設定の詳細については、QEMU を使用した KVM のシリアル コンソール アクセスの設定の項を参照してください。

13. `-boot once=d &`

   ISO を 1 回のみ起動します。

例：

```
/usr/bin/kvm \
      -smbios type=1,manufacturer="cisco",product="Cisco IOS XRv
9000",uuid=97fc351b-431d-4cf2-9c01-43c283faf2a3 \
      -cpu host \
      -drive file=/home/username/bnbMay13/workdir-username/disk1.raw,if=virtio,media=disk,index=1
\
     -drive file=/home/username/bnbMay13/workdir-username/xrv9k-fullk9-x.iso.baked,media=cdrom,index=2
 \
      -m 16384 \
      -smp cores=4,threads=1,sockets=1 \
      -enable-kvm \
      -daemonize \
      -display none \
      -rtc base=utc \
      -name IOS-XRv-9000:username \
      -runas username \
      -netdev tap,id=host1,ifname=usernameLx1,script=no,downscript=no \
      -netdev tap,id=host2,ifname=usernameLx2,script=no,downscript=no \
      -netdev tap,id=host3,ifname=usernameLx3,script=no,downscript=no \
      -device virtio-net-pci,romfile=,netdev=host1,id=host1,mac=52:46:84:57:A0:DA \
      -device virtio-net-pci,romfile=,netdev=host2,id=host2,mac=52:46:C4:F4:36:0F \
      -device virtio-net-pci,romfile=,netdev=host3,id=host3,mac=52:46:A5:C0:D0:C5 \
      -netdev tap,id=data1,ifname=usernameXr1,script=no,downscript=no \
      -netdev tap,id=data2,ifname=usernameXr2,script=no,downscript=no \
      -netdev tap,id=data3,ifname=usernameXr3,script=no,downscript=no \
      -device e1000,romfile=,netdev=data1,id=data1,mac=52:46:87:18:62:DF \
      -device e1000,romfile=,netdev=data2,id=data2,mac=52:46:32:02:90:6F \
      -device e1000,romfile=,netdev=data3,id=data3,mac=52:46:34:93:52:1F \
      -monitor telnet:0.0.0.0:11063,server,nowait \
      -serial telnet:0.0.0.0:10621,nowait,server \
      -serial telnet:0.0.0.0:14713,nowait,server \
      -serial telnet:0.0.0.0:18090,nowait,server \
      -serial telnet:0.0.0.0:17181,nowait,server \
      -boot once=d &
```

（注） 上記の例でdisk1.rawというラベルのディスクは、**qemu-img**で作成できます。**qemu-img**は、仮想ハードディスクの形式を変換するユーティリティです。上記の例では、生ディスク形式ではなく、qcow2ディスク形式を使用することができます。

プレインストールされているqcow2イメージも使用できます。その場合、cdromパラメータは削除されます。

Virsh を使用した KVM での Cisco IOS XRv 9000 ルータのインストール

この手順は、Cisco IOS XRv 9000 用の VM を手動で作成するための一般的なガイドラインです。実行する必要のある;正確な手順は、KVM の環境と設定の特性に応じて異なることがあります。詳細については、Red Hat Linux、Ubuntu、および Virsh のドキュメンテーションを参照してください。

始める前に

VM メモリのバルーニングはサポートされていないため、Virsh ファイルの中で memory と currentmemory の単位値（下記参照）は同じでなければなりません。

```
<memory unit='MB'>XXX</memory>
<currentMemory unit='MB'>XXX</currentMemory>
```

ステップ 1 Cisco.com から .iso または .qcow2 イメージ、およびサンプル Virsh XML ファイルをダウンロードします。

ステップ 2 XML ファイルを編集して .iso または .qcow2 イメージを指すようにし、インターフェイス ソースを編集して目的の接続を指定するようにします。

```
<!-- CDROM HDB Disk -->
     <disk type='file' device='cdrom'>
      <driver name='qemu' type='raw'/>
      <source file='/home/<username>/xrv9k-fullk9-x.iso'/>
      <target dev='hdc' bus='ide'/>
      <alias name='ide-cdrom'/>
     </disk>
```

ステップ 3 qcow2 イメージを使用する場合は、CDROM セクションをコメント アウトします。

ステップ 4 iso イメージを使用する場合は、次のようにします。

1. 空の qcow2 ディスクを作成します。

   ```
   qemu-img create -f qcow2 xrv9000.qcow2 45G
   ```

   （注） サポートされる最小の VM ハードディスク サイズは、リリース 6.0 の場合 45 GB、リリース 5.4 の場合は 55 GB です。

2. XML の HDA Disk セクションを編集して、作成した空の qcow2 ディスクを指すようにします。

```
<!-- HDA Disk -->
    <disk type='file' device='disk'>
     <driver name='qemu' type='qcow2'/>
     <source file='/home/<username>/xrv9000.qcow2'/>
     <target dev='vda' bus='virtio'/>
     <alias name='virtio-disk0'/>
    </disk>
```

ステップ 5 Virsh ドメインを作成します。

```
virsh create xrv9k-fullk9-x.virsh.xml
```

ステップ 6 ステップ 5 で作成された Virsh ドメインを検証します。

```
virsh list xrv9k-fullk9-x.virsh.xml
Id    Name                           State
----------------------------------------------------
 149   IOS-XRv-9000_username_virsh        running
```

ステップ 7 以下のような Virsh コマンドを使用して、VM を管理します。

- **reboot**

```
virsh reboot IOS-XRv-9000_USER1_virsh
```

- **shutdown**

```
virsh shutdown IOS-XRv-9000_USER1_virsh
```

- **destroy**

```
virsh destroy IOS-XRv-9000_USER1_virsh
```

シリアル コンソール アクセスの設定の詳細については、Virsh を使用した KVM のシリアル コンソール アクセスの設定の項を参照してください。

詳細については、Virsh コマンド リファレンスのドキュメンテーションを参照してください。

Cisco IOS XRv 9000 ルータの KVM インスタンスを OpenStack に作成する

この手順では、Cisco IOS XRv 9000 ルータを RHEL バージョン 7 以降と OpenStack バージョン 5 以降で使用する方法について説明します。この手順には OpenStack コマンドライン インターフェイスを使用します。この説明は、読者が OpenStack のコマンドと手順に精通していることを前提としています。詳細については、OpenStack のドキュメンテーションを参照してください。

手順を完了すると、3 つの Neutron ネットワークに接続された 3 つのデータ プレーン インターフェイスのある単一の Cisco IOS XRv 9000 ルータが動作することになります。

（注） トランク インターフェイスが必要な場合は、Neutron ML2 コア プラグインを Nexus 1kv 機能プラグインと併用してください。次の手順では、インターフェイスのアクセス（非タギング）タイプのみ作成します。

始める前に

次のものが必要です。

- Neutron ML2 コア プラグインを備えた OpenStack 5、6 または 7。機能プラグインとしての Open vSwitch、および Tenants ネットワーク タイプとしての VLAN。
- Cisco IOS XRv 9000 ルータ ISO イメージ（VGA タイプ）。

ステップ 1 イメージの準備

OpenStack に Cisco IOS XRv 9000 ルータを導入するには、次のものが必要です。

- Cisco IOS XRv 9000 ISO イメージ（VGA タイプ）
- 45 GB の空白 Cinder ボリューム（仮想ディスク）

（注） サポートされる最小の VM ハードディスク サイズは、リリース 6.0 の場合 45 GB、リリース 5.4 の場合は 55 GB です。

ISO イメージを起動すると、Cisco XRv 9000 ルータが 2 番目のディスク（45 GB ボリュームの空白ディスク）にインストールされます。後で、作成された Cinder ボリュームからルータを起動できます。必要に応じて、ISO イメージの代わりに、プレインストールされていた Cisco XRv 9000 qcow2 ディスクをダウンロードすることもできます。

1. Cisco IOS XRv 9000 ISO イメージ（VGA タイプ）は、OpenStack の glance にインポートする必要があります。イメージを glance にインポートするには、次のコマンドラインを使用します。

```
glance image-create --name xrv9k-fullk9_vga-x --disk-format iso --container bare \
--file xrv9k-fullk9_vga-x.iso
```

2. OpenStack glance のイメージを確認します。

```
 glance image-list
+--------------------------------------+-----------------------------------------+-----------+--------------+----------+--------+
| ID                                   | Name                                    | Disk Format
 | Container Format | Size       | Status |
+--------------------------------------+-----------------------------------------+-----------+--------------+----------+--------+
| 71b44355-32a8-45e7-abfd-f52593f2dc1a | csr1000v-universalk9.03.15.00.S.155-2.S | qcow2
 | bare             | 1335427072 | active |
| e779245a-9491-4bee-bc85-a73e9394b981 | xrv9k-fullk9_vga-x                      | iso
 | bare             | 775370752  | active |
| 3b3ade31-fae6-4354-9902-fb77452a65ab | xrvr-initial-config                     | iso
 | bare             | 358400     | active |
+--------------------------------------+-----------------------------------------+-----------+--------------+----------+--------+
```

ステップ 2 Cinder ボリュームの準備

次のコマンドラインを使用して、Cinder ボリュームを作成します。

1. Cisco IOS XRv 9000 ルータのディスク イメージの Cinder ボリュームを作成し、ブート可能にします。

```
cinder create --display-name xrv9k-disk 45
cinder set-bootable volume-id True
```

cinder list コマンドを実行すると、ルータの ISO イメージのボリューム ID が表示されます。

2. Cinder ボリュームを確認します。

```
cinder list
+--------------------------------------+--------+--------------+------+-------------+----------+-------------------------------------+
|                  ID                  | Status |    Display Name    | Size | Volume Type |
Bootable |             Attached to              |
+--------------------------------------+--------+--------------+------+-------------+----------+-------------------------------------+
|
```

```
| cbef86dd-9819-4daa-81b2-4b905b287974 | Available |    xrv9k-disk     |  45  |     None     |
  true    | 5262e1fe-37f5-4535-bf76-aab26cb86366 |
+--------------------------------------+-----------+-------------------+------+--------------+----------+--------------------------------------+
```

名前が xrv9k-disk で、**Status-Available** と **Bootable-True** に設定された Cinder ボリュームが表示されるはずです。

ステップ 3 Nova Flavor の設定

この仮想ハードウェア テンプレートは、OpenStack では**フレーバー**と呼ばれます。**nova flavor-create** コマンドを使用すると、新しいフレーバーを作成できます。nova のフレーバーは、RAM サイズ、ディスク、コア数に関する情報を、VM の新しいインスタンスが開始されるときに Nova のスケジューラ プロセスに渡すために使用されます。

Cisco IOS XRv 9000 ルータには、16 GB の RAM、45 GB のハード ディスク、および 4 個の vCPU が必要です。

```
nova flavor-create xrv9k-flavor auto 16384 45 4
```

xrv9k-flavor はフレーバーの名前です。

ステップ 4 Configuring network

Cisco IOS XRv 9000 ルータには、最低 4 つのネットワーク インターフェイスが必要です。最初の NIC または vNIC は XR 管理インターフェイスにマッピングされます。2 番目と 3 番目の NIC は予約済みで、残りの NIC はトラフィック インターフェイスにマッピングされます。インターフェイス マッピングの詳細については、「ルータ ネットワーク インターフェイスの VM ネットワーク インターフェイス カードへのマッピング」の項を参照してください。

ネットワーク設定のサンプルでは、6 つの neutron ネットワークと 6 つの neutron サブネットワークを設定し、VM インスタンスへのパラメータとして 6 つのネットワーク ID を渡すことができます。これは、3 つのトラフィック インターフェイスに相当します。**neutron net-create <*neutron-network-name*>** を使用して、Neutron にネットワークを作成します。

次に例を示します。

```
neutron net-create management-xr
neutron net-create management-other
neutron net-create management-host
neutron net-create datalink-1
neutron net-create datalink-2
neutron net-create datalink-3
```

上記の例で、最初の 3 つのコマンドはコントロール プレーン ネットワークを作成し、最後の 3 つのコマンドはデータ プレーン ネットワークを作成しています。

次に、対応するネットワーク名を使用して、Neutron にサブネットワークを作成します。**neutron subnet-create <*neutron-network-name*> <*IP-subnet*> --name <*neutron-network-name*>** コマンドを使用します。一貫性を確保するために、neutron とサブネットワークに同じ名前を指定します。

次に例を示します。

```
neutron subnet-create management-xr  10.50.70.0/26 --name management-xr
```

```
neutron subnet-create management-other  10.50.70.64/26 --name management-other
neutron subnet-create management-host  10.50.70.128/26 --name management-host
neutron subnet-create datalink-1  10.57.11.0/24 --name datalink-1
neutron subnet-create datalink-2  10.57.12.0/24 --name datalink-2
neutron subnet-create datalink-3  10.57.13.0/24 --name datalink-3
```

ステップ 5 neutron ルータに、management-xr サブネットと management-host サブネットを指定します。

次のコマンドを使用して、neutron ルータに management-xr サブネットと management-host サブネットを指定します。

```
neutron router-interface-add <Neutron router name> <subnet id of management-xr>
neutron router-interface-add <Neutron router name> <subnet id of management-host>
```

（注）
- DHCP の問題を回避するために、Cisco IOS XRv 9000 ルータを起動する前にこの手順を実行する必要があります。
- **neutron router-list** コマンドを実行すると、Neutron ルータ名の一覧が表示されます。
- **neutron subnet-list** コマンドを実行すると、Neutron サブネットの一覧が表示されます。

ステップ 6 Cisco IOS XRv 9000 ルータを起動します。

次のコマンド ラインを実行して、ルータを起動します。

```
nova boot
 --flavor xrv9k-flovor \
 --nic net-id={Control plane network ID of management-xr} \
 --nic net-id={Control plane network ID of management-other} \
 --nic net-id={Control plane network ID of management-host} \
 --nic net-id={Data plane network ID of datalink-1} \
 --nic net-id={Data plane network ID of datalink-2} \
 --nic net-id={Data plane network ID of datalink-3} \
 --block-device id={glance ID of Cisco IOS XRv 9000 router's CD volume available in Step 1},\
 --source=image,dest=volume,bus=ide,device=/dev/hdc,size=1,type=cdrom,bootindex=1 \
 --block-device source=volume,id={cinder ID of Cisco IOS XRv 9000 Router's disk volume available in
 Step 1}, dest=volume,size=50,bootindex=0 xrv9k-1
nova list (your instance should be in Active state)
nova get-vnc-console xrv9k-1 novnc
```

nova get-vnc-console コマンドは、Cisco IOS XRv 9000 ルータ コンソールへのアクセスに使用される URL を返します。

（注） **config-drive** を実行すると、VM を初期設定で起動できます。これにより、VM の電源をオンにしたときに、config-drive に設定されているサービスが実行されます。この Cisco IOS XRv 9000 ルータは VM 上で動作しているので、ルータはプレーン テキスト ファイルを使用した有効な XR 設定を受け入れます。CLI からコマンドが入力された場合と同様、起動プロセス中にこのテキスト ファイルはコマンド パーサーによって解析されます。

config-drive のサポートが必要な場合は、**nova boot** コマンドに次の行を追加することにより、XR の初期設定を格納したプレーン テキスト ファイルを渡すことができます。

```
config-drive true user-data <path>/iosxr_config.txt file
/iosxr_config.txt=<path>/iosxr_config.txt
```

シスコの仮想アプライアンス設定（CVAC）の詳細については、「CVAC：ブートストラップ構成のサポート」の項を参照してください。

KVM 構成でのパフォーマンスの向上

KVM 環境内の Cisco IOS XRv 9000 ルータは、KVM ホストの設定を変更することでパフォーマンスの向上が可能です。これらの設定は、ルータの Cisco IOS XR の構成時の設定とは無関係です。このオプションは、Red Hat Enterprise Linux 7.0 KVM で使用できます。

CPU ピニングを有効にすると、KVM 構成でのパフォーマンスを向上できます。

（注） Cisco IOS XR ソフトウェア リリース 5.4 は、KVM の VirtIO インターフェイスで 1518 バイトを超えるジャンボ パケットをサポートしていません。1518 バイトを超えるパケットは破棄されます。

CPU ピニングの有効化

KVM 環境で Cisco IOS XRv 9000 ルータのパフォーマンスを向上させるため、KVM CPU アフィニティ オプションを使用して特定のプロセッサに VM を割り当てることができます。このオプションを使用する場合は、KVM ホストで CPU ピンニングを構成します。

CPU ピンニングを構成するには、次の手順を実行します。

1. KVM ホスト環境で、ピンニングに使用できる vCPU の数を調べるために、ホストのトポロジを確認します。

   ```
   virsh nodeinfo
   ```

2. 使用可能な vCPU の数を確認します。

   ```
   virsh capabilities
   ```

3. vCPU をプロセッサ コアのセットにピンニングします。

   ```
   virsh vcpupin <vm-name> <vcpu-number> <host-core-number>
   ```

virsh vcpupin コマンドは、Cisco IOS XRv 9000 ルータの vCPU ごとに実行する必要があります。次の例は、ホストに 8 個のコアがあり Cisco IOS XRv 9000 ルータが 4 個の vCPU で構成されている場合に必要になる KVM コマンドを示しています。

```
virsh vcpupin xrv9000 0 2
virsh vcpupin xrv9000 1 3
virsh vcpupin xrv9000 2 4
virsh vcpupin xrv9000 3 5
```

ホストのコア番号は、0 ～ 7 のどの番号でもかまいません。詳細については、KVM のドキュメンテーションを参照してください。

（注） CPU ピンニングを構成する場合は、ホスト サーバの CPU トポロジを慎重に検討してください。複数のコアで構成された Cisco IOS XRv 9000 ルータを使用している場合は、複数のソケットにまたがる CPU ピンニングを構成しないでください。

KVM 構成でのパフォーマンスの向上には、専用のシステム リソースが必要になるという短所もあります。