ASA 仮想 の KVM への展開

カーネルベースの仮想マシン（KVM）を実行できる任意のサーバークラスの x86 CPU デバイスに ASA 仮想 を導入できます。

重要 ASA 仮想 の最小メモリ要件は 2GB です。現在の ASA 仮想 が 2GB 未満のメモリで動作している場合、ASA 仮想 マシンのメモリを増やさないと、以前のバージョンから 9.13(1) 以降にアップグレードできません。また、最新バージョンを使用して新しい ASA 仮想 マシンを再導入できます。

注意事項と制約事項

ASA 仮想 の導入に使用される特定のハードウェアは、導入されるインスタンスの数や使用要件によって異なります。作成する各仮想アプライアンスには、ホストマシン上での最小リソース割り当て（メモリ、CPU 数、およびディスク容量）が必要です。

重要 ASA 仮想 は、8GB のディスクストレージサイズで導入されます。ディスク容量のリソース割り当てを変更することはできません。

（注） ASA 仮想 バージョン 9.16.x 以降で、デバイス構成が 16 vCPU および 32GB RAM の ASAv100 から ASAv10 にダウングレードする場合は、デバイス構成を 1 vCPU および 4GB RAM にする必要があります。

ASA 仮想 を導入する前に、次のガイドラインと制限事項を確認します。

KVM での ASA 仮想 のシステム要件

最適なパフォーマンスを確保するために、以下の仕様に準拠していることを確認してください。ASA 仮想 には、次の要件があります。

- ホスト CPU は、仮想化拡張機能を備えたサーバークラスの x86 ベースの Intel または AMD CPU である必要があります。

  たとえば、ASA 仮想 パフォーマンステストラボでは、2.6GHz で動作する Intel® Xeon® CPU E5-2690v4 プロセッサを搭載した Cisco Unified Computing System™（Cisco UCS®）C シリーズ M4 サーバーを最低限使用しています。

推奨される vNIC

最適なパフォーマンスを得るためには、次の vNIC を推奨します。

- PCI パススルーでの i40e：サーバーの物理 NIC を VM に関連付け、DMA（ダイレクトメモリアクセス）を介して NIC と VM の間でパケットデータを転送します。パケットの移動に CPU サイクルは必要ありません。
- i40evf/ixgbe-vf：実質的に上記と同じですが（NIC と VM 間の DMA パケット）、NIC を複数の VM 間で共有できます。SR-IOV は、導入の柔軟性が高いため、一般的に推奨されます。参照先
- virtio：10Gbps の動作をサポートしますが、CPU サイクルも必要な準仮想化ネットワークドライバです。

（注） KVM システムで実行されている ASA 仮想 インスタンスでは、vNIC ドライバ i40e バージョン 2.17.4 を使用する SR-IOV インターフェイスでデータ接続の問題が発生する場合があります。この問題の回避策として、この vNIC バージョンを他のバージョンにアップグレードすることを推奨します。

パフォーマンスの最適化

ASA 仮想 の最高のパフォーマンスを実現するために、VM とホストの両方を調整することができます。詳細については、パフォーマンスの調整 （18 ページ）を参照してください。

- **NUMA**：ゲスト VM の CPU リソースを単一の Non-Uniform Memory Access（NUMA）ノードに分離することで、ASA 仮想 のパフォーマンスを向上できます。詳細については、NUMA のガイドライン （19 ページ）を参照してください。
- **Receive Side Scaling**：ASA 仮想 は Receive Side Scaling（RSS）をサポートしています。これは、ネットワークアダプタによって複数のプロセッサコアにネットワーク受信トラフィックを分散するために使用されるテクノロジーです。詳細については、Receive Side Scaling（RSS）用の複数の RX キュー （22 ページ）を参照してください。
- **VPN の最適化**：ASA 仮想 で VPN パフォーマンスを最適化するための追加の考慮事項については、VPN の最適化 （24 ページ）を参照してください。

クラスタリング

バージョン 9.17 以降、クラスタリングは KVM で展開された ASA 仮想インスタンスでサポートされます。詳細については、「ASA Cluster for the ASAv」を参照してください。

CPU ピニング

KVM 環境で ASA 仮想 を機能させるには、CPU ピニングが必要です。CPU ピンニングの有効化 （18 ページ）を参照してください。

ハイ アベイラビリティ ガイドラインのためのフェールオーバー

フェールオーバー配置の場合は、スタンバイ装置が同じライセンス権限付与を備えていることを確認してください（たとえば、両方の装置が 2Gbps の権限付与であることなど）。

重要 ASA 仮想 を使用して高可用性ペアを作成する場合は、データインターフェイスを各 ASA 仮想 に同じ順序で追加する必要があります。完全に同じインターフェイスが異なる順序で各 ASA 仮想 に追加されると、ASA 仮想 コンソールにエラーが表示されることがあります。また、フェールオーバー機能にも影響が出ることがあります。

Proxmox VE 上の ASA 仮想

Proxmox Virtual Environment（VE）は、KVM 仮想マシンを管理できるオープンソースのサーバー仮想化プラットフォームです。Proxmox VE は、Web ベースの管理インターフェイスも提供します。

Proxmox VE に ASA 仮想 を導入する場合は、エミュレートされたシリアルポートを持つように VM を設定する必要があります。シリアルポートがないと、ブートアッププロセス中に ASA 仮想 がループ状態になります。すべての管理タスクは、Proxmox VE Web ベース管理インターフェイスを使用して実行できます。

（注） Unix シェルまたは Windows Powershell に慣れている上級ユーザー向けに、Proxmox VE は仮想環境のすべてのコンポーネントを管理するコマンド ライン インターフェイスを提供します。このコマンドラインインターフェイスには、インテリジェントなタブ補完機能と UNIX の man ページ形式の完全なドキュメントがあります。

ASA 仮想 を正しく起動するには、VM にシリアルデバイスを設定する必要があります。

1. メイン Management Center の左側のナビゲーションツリーで ASA 仮想 マシンを選択します。
2. 仮想マシンの電源をオフにします。
3. **Hardware** > **Add** > **Network Device** を選択して、シリアルポートを追加します。
4. 仮想マシンの電源をオンにします。
5. Xterm.js を使用して ASA 仮想 マシンにアクセスします。

ゲスト/サーバーで端末をセットアップしてアクティブ化する方法については、Proxmox シリアル端末のページを参照してください。

IPv6 のサポート

KVM で IPv6 をサポートする設定の vNIC を作成するには、IPv6 設定パラメータで構成される XML ファイルをインターフェイスごとに作成する必要があります。**virsh net-create** <<*interface configuration XML file name*>> コマンドを使用してこれらの XML ファイルを実行することにより、IPv6 ネットワークプロトコルを使用する vNIC をインストールできます。

インターフェイスごとに、次の XML ファイルを作成できます。

- 管理インターフェイス：*mgmt-vnic.xml*
- 診断インターフェイス：*diag-vnic.xml*
- 内部インターフェイス：*inside-vnic.xml*
- 外部インターフェイス：*outside-vnic.xml*

例：

IPv6 設定の管理インターフェイス用の XML ファイルを作成する方法。

```
<network>
            <name>mgmt-vnic</name>
            <bridge name='mgmt-vnic' stp='on' delay='0' />
            <ip family='ipv6' address='2001:db8::a111:b220:0:abcd' prefix='96'/>
        </network>
```

同様に、他のインターフェイス用の XML ファイルも作成する必要があります。

次のコマンドを実行して、KVM にインストールされている仮想ネットワークアダプタを確認できます。

```
virsh net-list
        brctl show
```

UEFI およびセキュアブートの制限事項

KVM では、UEFI ファームウェアおよび UEFI セキュアブートはグリーンフィールド（新規）展開でのみサポートされており、展開時に設定する必要があります。

レガシー BIOS モードを使用する既存のブラウンフィールド展開は、影響なしでバージョン 9.24 にアップグレードできます。展開後の UEFI ファームウェアへの切り替えまたは UEFI セキュアブートの有効化はサポートされていません。

アップグレードの制約事項と制限事項

アップグレード復元の制約事項

⚠

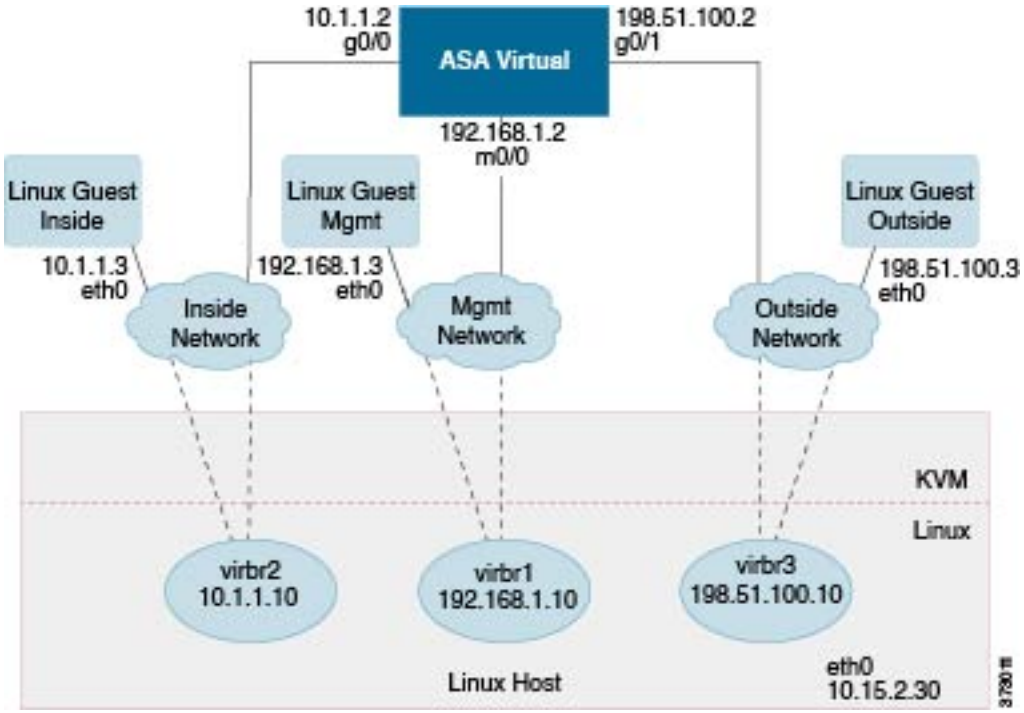
注意 アップグレードの復元はサポートされていません。

アップグレード前に必ずバックアップを作成してください。ASA Virtual 9.24 にアップグレードした後、以前のソフトウェアバージョンにロールバックすることはできません。以前のバージョンに戻すには、Management Center を再展開する必要があります。

概要

次の図は、ASA 仮想 と KVM のネットワークトポロジの例を示します。この章で説明している手順は、このトポロジの例に基づいています。ASA 仮想 は、内部ネットワークと外部ネットワークの間のファイアウォールとして動作します。また、別個の管理ネットワークが設定されます。

図 *1 : KVM* を使用した *ASA* 仮想 の導入例



前提条件

- Cisco.com から ASA 仮想 qcow2 ファイルをダウンロードし、Linux ホストに格納します。

  http://www.cisco.com/go/asa-software

（注） Cisco.com のログインおよびシスコ サービス契約が必要です。

- このマニュアルの導入例では、ユーザーが Ubuntu 18.04 LTS を使用していることを前提としています。Ubuntu 18.04 LTS ホストの最上部に次のパッケージをインストールします。
  - qemu-kvm
  - libvirt bin
  - bridge-utils
  - Virt-Manager
  - virtinst
  - virsh tools
  - genisoimage

- ASAvU は ASA 9.22 以降サポートされています。
- パフォーマンスはホストとその設定の影響を受けます。ホストを調整することで、KVM での ASA 仮想のスループットを最大化できます。一般的なホスト調整の概念については、『NFV Delivers Packet Processing Performance with Intel』を参照してください。
- Ubuntu 18.04 の便利な最適化には、次のものが含まれます。
  - macvtap：高性能の Linux ブリッジ。Linux ブリッジの代わりに macvtap を使用できます。ただし、Linux ブリッジの代わりに macvtap を使用する場合は、特定の設定を行う必要があります。
  - Transparent Huge Pages：メモリページサイズを増加させます。Ubuntu 18.04 では、デフォルトでオンになっています。

    Hyperthread disabled：2 つの vCPUを 1 つのシングル コアに削減します。
  - txqueuelength：デフォルトの txqueuelength を 4000 パケットに増加させ、ドロップ レートを低減します。
  - pinning：qemu および vhost プロセスを特定のCPU コア にピン接続します。特定の条件下では、ピン接続によってパフォーマンスが大幅に向上します。
- RHEL ベースのディストリビューションの最適化については、『Red Hat Enterprise Linux 7 Virtualization Tuning and Optimization Guide』を参照してください。
- ASA ソフトウェアおよび ASA 仮想 ハイパーバイザの互換性については、Cisco Secure Firewall ASA の互換性 [英語] を参照してください。
- OVMF パッケージを KVM ホストにインストールする必要があります。参考のため、OVMF 設定のサンプルを以下に追加します。

```
$ cat /usr/share/libvirt/firmware/ovmf.json
{
 "description": "UEFI OVMF firmware",
  "interface-types": ["uefi"],
  "mapping": {
    "device": "flash",
    "mode": "readonly",
    "firmware": "/usr/share/OVMF/OVMF_CODE.fd"
  },
  "features": {
    "secure-boot": false
  },
  "targets": [
    {
      "architecture": "x86_64",
      "machines": ["q35", "pc"]
    }
  ]
}

$ cat /usr/share/libvirt/firmware/ovmf-secureboot.json
{
  "description": "UEFI Secure Boot with OVMF",
  "interface-types": ["uefi"],
  "mapping": {
    "device": "flash",
```

```
    "mode": "readonly",
    "firmware": "/usr/share/OVMF/OVMF_CODE.secboot.fd",
    "nvram-template": "/usr/share/OVMF/OVMF_VARS.fd"
  },
  "features": {
    "secure-boot": true
  },
  "targets": [
    {
      "architecture": "x86_64",
      "machines": ["pc", "q35"]
    }
  ]
}
```

第 0 日のコンフィギュレーション ファイルの準備

ASA 仮想 を起動する前に、第 0 日用のコンフィギュレーション ファイルを準備できます。このファイルは、ASA 仮想 の起動時に適用される ASA 仮想 の設定を含むテキストファイルです。この初期設定は、「day0-config」というテキストファイルとして指定の作業ディレクトリに格納され、さらに day0.iso ファイルへと処理されます。この day0.iso ファイルが最初の起動時にマウントされて読み取られます。第 0 日用コンフィギュレーション ファイルには、少なくとも、管理インターフェイスをアクティブ化するコマンドと、公開キー認証用 SSH サーバーを設定するコマンドを含める必要がありますが、すべての ASA 設定を含めることもできます。

day0.iso ファイル（カスタム day0.iso またはデフォルト day0.iso）は、最初の起動中に使用できる必要があります。

- 初期導入時に自動的に ASA 仮想 にライセンスを付与するには、Cisco Smart Software Manager からダウンロードした Smart Licensing Identity（ID）トークンを「idtoken」というテキストファイルに格納し、第 0 日用構成ファイルと同じディレクトリに保存します。
- 仮想 VGA コンソールではなく、ハイパーバイザの**シリアルポート**から ASA 仮想 にアクセスし、設定する場合は、第 0 日用構成ファイルにコンソールシリアルの設定を追加して初回ブート時にシリアルポートを使用する必要があります。
- トランスペアレントモードで ASA 仮想 を導入する場合は、トランスペアレントモードで実行される既知の ASA 構成ファイルを、第 0 日用構成ファイルとして使用する必要があります。これは、ルーテッド ファイアウォールの第 0 日用コンフィギュレーション ファイルには該当しません。

（注） この例では Linux が使用されていますが、Windows の場合にも同様のユーティリティがあります。

手順

ステップ 1 「day0-config」というテキストファイルに ASA 仮想 の CLI 設定を記入します。3 つのインターフェイスの設定とその他の必要な設定を追加します。

最初の行は ASA のバージョンで始める必要があります。day0-config は、有効な ASA 構成である必要があります。day0-config を生成する最適な方法は、既存の ASA または ASA 仮想 から実行コンフィギュレーションの関連部分をコピーする方法です。day0-config 内の行の順序は重要で、既存の **show running-config** コマンド出力の順序と一致している必要があります。

例：

```
ASA Version
!
interface management0/0
ipv6 enable
ipv6 address 2001:db8::a111:b220:0:abcd/96
nameif management
security-level 100
no shut

interface gigabitethernet0/0
ipv6 enable
ipv6 address 2001:db8::a111:b221:0:abcd/96
nameif inside
security-level 100
no shut

interface gigabitethernet1/0
ipv6 enable
ipv6 address 2001:db8::a111:b222:0:abcd/96
nameif outside
security-level 100
no shut

crypto key generate rsa general-keys modulus 4096
ssh ::/0 inside
ssh timeout 60
ssh version 2
aaa authentication ssh console LOCAL

dns domain-lookup management
dns server-group DefaultDNS
name-server 2001:4860:4860::8888
```

ステップ 2 （任意）ASA 仮想 の初期導入時に自動的にライセンスを許諾する場合は、day0-config ファイルに次の情報が含まれていることを確認してください。

- 管理インターフェイスの IP アドレス
- （任意）SSmart Licensing で使用する HTTP プロキシ
- HTTP プロキシ（指定した場合）または tools.cisco.com への接続を有効にする **route** コマンド
- tools.cisco.com を IP アドレスに解決する DNS サーバー

- 要求する ASA 仮想 ライセンスを指定するための Smart Licensing の設定
- （任意）CSSM での ASA 仮想 の検索を容易にするための一意のホスト名

ステップ 3 （任意）Cisco Smart Software Manager によって発行された Smart License ID トークンファイルをコンピュータにダウンロードし、ダウンロードファイルから ID トークンをコピーし、ID トークンのみを含む「idtoken」というテキストファイルを作成します。

ステップ 4 テキスト ファイルを ISO ファイルに変換して仮想CD-ROM を生成します。

例：

```
stack@user-ubuntu:-/KvmAsa$ sudo genisoimage -r -o day0.iso day0-config idtoken
I: input-charset not specified, using utf-8 (detected in locale settings)
Total translation table size: 0
Total rockridge attributes bytes: 252
Total directory bytes: 0
Path table size (byptes): 10
Max brk space used 0
176 extents written (0 MB)
stack@user-ubuntu:-/KvmAsa$
```

この ID トークンによって、Smart Licensing サーバーに ASA 仮想 が自動的に登録されます。

ステップ 5 ステップ 1 から 5 を繰り返し、導入する ASA 仮想 ごとに、適切な IP アドレスを含むデフォルトの構成ファイルを作成します。

仮想ブリッジ XML ファイルの準備

ASA 仮想 ゲストを KVM ホストに接続し、ゲストを相互接続する仮想ネットワークを設定する必要があります。

（注） この手順では、KVM ホストから外部への接続は確立されません。

KVM ホスト上に仮想ブリッジ XML ファイルを準備します。第 0 日のコンフィギュレーション ファイルの準備 （8 ページ） に記載されている仮想ネットワーク トポロジの例では、3 つの仮想ブリッジ ファイル（virbr1.xml、virbr2.xml、virbr3.xml）が必要です（これらの 3 つのファイル名を使用する必要があります。たとえば、virbr0 はすでに存在しているため使用できません）。各ファイルには、仮想ブリッジの設定に必要な情報が含まれています。仮想ブリッジに対して名前と一意の MAC アドレスを指定する必要があります。IP アドレスの指定は任意です。

手順

ステップ 1 3 つの仮想ネットワーク ブリッジ XML ファイルを作成します。次の例では、virbr1.xml、virbr2.xml、および virbr3.xml です。

例：

```
<network>
<name>virbr1</name>
<bridge name='virbr1' stp='on' delay='0' />
<mac address='52:54:00:05:6e:00' />
<ip address='192.168.1.10' netmask='255.255.255.0' />
</network>
```

例：

```
<network>
<name>virbr2</name>
<bridge name='virbr2' stp='on' delay='0' />
<mac address='52:54:00:05:6e:01' />
<ip address='10.1.1.10' netmask='255.255.255.0' />
</network>
```

例：

```
<network>
<name>virbr3</name>
<bridge name='virbr3' stp='on' delay='0' />
<mac address='52:54:00:05:6e:02' />
<ip address='198.51.100.10' netmask='255.255.255.0' />
</network>
```

ステップ 2 以下を含むスクリプトを作成します（この例では、スクリプトに virt_network_setup.sh という名前を付けます）。

```
virsh net-create virbr1.xml
virsh net-create virbr2.xml
virsh net-create virbr3.xml
```

ステップ 3 このスクリプトを実行して、仮想ネットワークを設定します。このスクリプトは、仮想ネットワークを稼働状態にします。ネットワークは、KVM ホストが動作している限り稼動します。

```
stack@user-ubuntu:-/KvmAsa$ virt_network_setup.sh
```

（注）
Linux ホストをリロードする場合は、virt_network_setup.sh スクリプトを再実行する必要があります。スクリプトはリブート後に継続されません。

ステップ 4 仮想ネットワークが作成されたことを確認します。

```
stack@user-ubuntu:-/KvmAsa$ brctl show
bridge name bridge id STP enabled Interfaces
virbr0 8000.0000000000000 yes
virbr1 8000.5254000056eed yes virb1-nic
virbr2 8000.5254000056eee yes virb2-nic
```

```
virbr3 8000.5254000056eec yes virb3-nic
stack@user-ubuntu:-/KvmAsa$
```

ステップ 5 virbr1 ブリッジに割り当てられている IP アドレスを表示します。これは、XML ファイルで割り当てた IP アドレスです。

```
stack@user-ubuntu:-/KvmAsa$ ip address show virbr1
S: virbr1: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN
link/ether 52:54:00:05:6e:00 brd ff:ff:ff:ff:ff:ff
inet 192.168.1.10/24 brd 192.168.1.255 scope global virbr1
valid_lft forever preferred_lft forever
```

ASA 仮想 の導入

導入スクリプトを使用した起動

virt-install ベースの導入スクリプトを使用して ASA 仮想 を起動できます。

手順

ステップ 1 「virt_install_asav.sh」という virt-install スクリプトを作成します。

ASA 仮想 マシンの名前は、この KVM ホスト上の他の全 VM で一意である必要があります。

ASA 仮想 では最大 10 のネットワークがサポートされます。この例では 3 つのネットワークが使用されています。ネットワーク ブリッジの句の順序は重要です。リストの最初の句は常に ASA 仮想 の管理インターフェイス（Management 0/0）、2 番目の句は ASA 仮想 の GigabitEthernet 0/0、3 番目の句は ASA 仮想 の GigabitEthernet 0/1 に該当し、GigabitEthernet 0/8 まで同様に続きます。仮想 NIC は Virtio でなければなりません。

例：

セキュアブート設定の場合：特定のパラメータを指定して *virt-install* コマンドを使用します。

```
virt-install \
  --connect=qemu:///system \
  --network bridge:br1556,model=virtio\
  --network bridge:br-in,model=virtio\
  --network bridge:br-out,model=virtio\
  --name=<prefix>-vm-asav \
  --cpu host \
  --arch=x86_64 \
  --vcpus=8 \
  --ram=16384 \
  --os-type=linux \
  --os-variant=generic \
  --virt-type=kvm \
  --import \
  --watchdog i6300esb,action=reset \
  --disk path=<path to qcow2 file>,format=qcow2,device=disk,bus=virtio,cache=none \
```

```
--disk path=<path to day0.iso file>,format=iso,device=cdrom,bus=sata \
--console pty,target_type=serial \
--serial tcp,host=127.0.0.1:<port>,mode=bind,protocol=telnet \
--boot firmware=efi,loader_secure=yes \
--machine q35 \
--features smm.state=on \
--force
```

（注）
セキュアブートの場合、マシンタイプがq35でSMM（システム管理モード）がオンになっている必要があります。

VM 管理：

（注）
VM を削除するには、次のコマンドを使用します。

```
virsh undefine <vm-name> --nvram
```

（注）
Cisco Secure Firewall ASA バージョン 9.22 以降では、ASAvU ライセンスを使用して、32 コア（上記の例では **vcpus** パラメータ）と 65536 MB（64 GB）の RAM（上記の例では **ram** パラメータ）を入力し、レート制限を削除できます。ASAvU ライセンスの詳細については、『Licensing for the ASA Virtual』を参照してください。

ステップ 2 virt_install スクリプトを実行します。

例：

```
stack@user-ubuntu:-/KvmAsa$ ./virt_install_asav.sh

Starting install...
Creating domain...
```

ウィンドウが開き、VMのコンソールが表示されます。VMが起動中であることを確認できます。VMが起動するまでに数分かかります。VM が起動したら、コンソール画面から CLI コマンドを実行できます。

グラフィカル ユーザー インターフェイスを使用した起動

GUI を使用して KVM 仮想マシンを管理するためのオープンソースオプションがいくつかあります。以下の手順では、virt-manager（Virtual Machine Manager とも呼ばれる）を使用して ASA Virtual を起動します。virt-managerは、ゲスト仮想マシンを作成および管理するためのグラフィカルツールです。

（注） KVM は、さまざまな種類の CPU をエミュレートできます。VM の場合、通常はホストシステムの CPU に厳密に一致するプロセッサタイプを選択する必要があります。これにより、ホストの CPU 機能（CPU フラグとも呼ばれます）が VM で使用できるようになります。CPU タイプを**ホスト**に設定する必要があります。その場合、VMはホストシステムとまったく同じCPUフラグを持ちます。

手順

ステップ 1 virt-manager を起動します（**[アプリケーション（Applications）]** > **[システムツール（System Tools）]** > **[仮想マシンマネージャ（Virtual Machine Manager）]**）。

ハイパーバイザの選択、およびルート パスワードの入力を求められる可能性があります。

ステップ 2 左上隅のボタンをクリックし、[VMの新規作成（New VM）] ウィザードを開きます。

ステップ 3 仮想マシンの詳細を入力します。

a) オペレーティング システムの場合、[既存のディスクイメージをインポート（Import existing disk image）] を選択します。

この方法でディスク イメージ（事前にインストールされた、ブート可能なオペレーティング システムを含んでいるもの）をインポートできます。

b) [次へ（Forward）] をクリックして続行します。

ステップ 4 ディスク イメージをロードします。

a) [参照...（Browse...）] をクリックしてイメージ ファイルを選択します。
b) [OSタイプ（OS type）] には [汎用（Generic）] を選択します。
c) [次へ（Forward）] をクリックして続行します。

ステップ 5 メモリおよび CPU オプションを設定します。

a) ASA Virtual プラットフォームサイズに対応する**メモリ（RAM）**パラメータを設定します。
b) ASA Virtual プラットフォームサイズに対応する **CPU** パラメータを設定します。
c) [次へ（Forward）] をクリックして続行します。

ステップ 6 [インストール前に設定をカスタマイズする（Customize configuration before install）] チェックボックスをオンにして、[名前（Name）] を指定してから [完了（Finish）] をクリックします。

この操作を行うと、別のウィザードが開き、仮想マシンのハードウェア設定を追加、削除、設定することができます。

ステップ 7 CPU 構成を次のように変更します。

左側のパネルから [プロセッサ（Processor）] を選択し、**[設定（Configuration）]** > **[ホストCPU構成のコピー（Copy host CPU configuration）]** を選択します。

これによって、物理ホストの CPU モデルと設定が VM に適用されます。

ステップ 8 仮想ディスクを設定します。

a) 左側のパネルから [ディスク1（Disk 1）] を選択します。
b) [詳細オプション（Advanced Options）] をクリックします。
c) [ディスクバス（Disk bus）] を [Virtio] に設定します。
d) [ストレージ形式（Storage format）] を [qcow2] に設定します。

ステップ 9 シリアル コンソールを設定します。

a) 左側のパネルから [コンソール（Console）] を選択します。

b) [削除（Remove）] を選択してデフォルト コンソールを削除します。
c) [ハードウェアを追加（Add Hardware）] をクリックしてシリアル デバイスを追加します。
d) [デバイスタイプ（Device Type）] で、[TCP net console (tcp)] を選択します。
e) [モード（Mode）] で、[サーバーモード（バインド）（Server mode (bind)）] を選択します。
f) [ホスト（Host）] には「**0.0.0.0**」と入力し、IP アドレスと一意の**ポート**番号を入力します。
g) [Telnetを使用（Use Telnet）] ボックスをオンにします。
h) デバイス パラメータを設定します。

ステップ 10 KVMゲストがハングまたはクラッシュしたときに何らかのアクションが自動でトリガーされるようウォッチドッグ デバイスを設定します。

a) [ハードウェアを追加（Add Hardware）] をクリックしてウォッチドック デバイスを追加します。
b) [モデル（Model）] で、[デフォルト（default）] を選択します。
c) [アクション（Action）] で、[ゲストを強制的にリセット（Forcefully reset the guest）] を選択します。

ステップ 11 ネットワーク インターフェイスを設定します。

[ハードウェアの追加（Add Hardware）] をクリックしてインターフェイスを追加し、**macvtap** を選択するか、共有デバイス名を指定します（ブリッジ名を使用）。

vnic0：管理インターフェイス（必須）

vnic1：診断インターフェイス（必須）

vnic2：外部インターフェイス（必須）

vnic3：内部インターフェイス（必須）

vnic4 ～ 10：データインターフェイス（オプション）

重要
vnic0、vnic1、および vnic3 は、必ず同じサブネットにマップするようにしてください。

ステップ 12 第 0 日のコンフィギュレーション ファイルを使用して展開する場合、ISO の仮想 CD-ROM を作成します。

a) [ハードウェアを追加（Add Hardware）] をクリックします。
b) [ストレージ（Storage）] を選択します。
c) [管理対象またはその他既存のストレージを選択（Select managed or other existing storage）] をクリックし、ISO ファイルの場所を参照します。
d) [デバイスタイプ（Device type）] で、[IDE CDROM] を選択します。

ステップ 13 仮想マシンのハードウェアを設定した後、[適用（Apply）] をクリックします。

ステップ 14 virt-manager の [インストールの開始（Begin installation）] をクリックして、指定したハードウェア設定で仮想マシンを作成します。

（注）
virt-manager で ASA Virtual を起動すると、デフォルトでグラフィカル（SPICE）コンソールが開きます。システムによっては、このコンソールはフリーズしているように見えたり、起動中に一部の出力だけが表示されたりすることがあります。ただし、デバイスはバックグラウンドで正常に起動を続けています。

完全なコンソール出力を表示するには、次に移動します。

[表示（View）] → [コンソール（Consoles）] → [コンソール（Console）] または [シリアル（Serial）]

TCP シリアルコンソールが設定されている場合は、代わりに telnet を使用してコンソールにアクセスします。出力は virt-manager に表示されません。

ホットプラグ インターフェイス プロビジョニング

ASA 仮想 を停止して再起動することなく、インターフェイスを動的に追加および削除できます。ASA 仮想 マシンに新しいインターフェイスを追加する場合、ASA 仮想 はそのインターフェイスを通常のインターフェイスとして検出してプロビジョニングできる必要があります。同様に、ホットプラグ プロビジョニングによって既存のインターフェイスを削除する場合、ASA 仮想 はそのインターフェイスを削除して、関連付けられたすべてのリソースを解放する必要があります。

注意事項と制約事項

インターフェイスのマッピングと番号付け

- ホットプラグインターフェイスを追加する場合、そのインターフェイス番号は、現在の最後のインターフェイス番号に 1 を加えた数になります。
- ホットプラグインターフェイスを削除すると、それが最後の番号のインターフェイスである場合を除き、インターフェイス番号にギャップが生じます。
- インターフェイス番号にギャップがあると、次にホットプラグプロビジョニングされるインターフェイスはそのギャップを埋める番号を使用します。

フェールオーバー

- ホットプラグ インターフェイスをフェールオーバーリンクとして使用する場合、リンクは、ASA 仮想のフェールオーバーペアとして指定されている両方のユニットでプロビジョニングする必要があります。
  - まずハイパーバイザのアクティブ ASA 仮想 にホットプラグ インターフェイスを追加してから、ハイパーバイザのスタンバイ ASA 仮想 にホットプラグ インターフェイスを追加します。
  - アクティブ ASA 仮想 に新たに追加したフェールオーバー インターフェイスを設定します。設定はスタンバイ装置に同期されます。
  - プライマリ ユニットのフェールオーバーを有効にします。

- フェールオーバーリンクを削除する場合、最初にアクティブな ASA 仮想 でフェールオーバー設定を削除します。
  - ハイパーバイザのアクティブな ASA 仮想 からフェールオーバー インターフェイスを削除します。
  - 次に、ハイパーバイザのスタンバイ ASA 仮想 から対応するインターフェイスをすぐに削除します。

制限事項と制約事項

- ホットプラグ インターフェイス プロビジョニングは Virtio 仮想 NIC に限定されます。
- サポートされるインターフェイスの最大数は 10 です。10 を超える数のインターフェイスを追加しようとすると、エラーメッセージが表示されます。
- インターフェイス カード（media_ethernet/port/id/10）を開くことはできません。
- ホットプラグ インターフェイス プロビジョニングでは ACPI が必要です。virt-install スクリプトには `--noacpi` フラグを含めないでください。
- Vector Packet Processing（VPP）が有効になっている場合、KVM 上のアクティブな ASA Virtual に対するホットプラグ インターフェイス プロビジョニング（インターフェイスの追加または削除）はサポートされません。これは、VPP でインターフェイスの変更を通知できないためです。

ネットワーク インターフェイスのホットプラグ

KVM ハイパーバイザのインターフェイスを追加および削除するには、virsh コマンド ラインを使用します。

手順

ステップ 1 virsh コマンド ラインのセッションを開きます。

例：

```
[root@asav-kvmterm ~]# virsh
Welcome to virsh, the virtualization interactive terminal.
Type: 'help' for help with commands
'quit' to quit
```

ステップ 2 インターフェイスを追加するには、**attach-interface** コマンドを使用します。

attach-interface{ **--domain** *domain* **--type** *type* **--source** *source* **--model** *model* **--mac** *mac* **--live**}

--domain には、短整数、名前、または完全 UUID を指定できます。--type パラメータは、物理的なネットワーク デバイスを示す *network*、またはデバイスへのブリッジを示す *bridge* のどちらかを指定できます。--source パラメータは、接続のタイプを示します。--model パラメータは、仮想 NIC のタイプを示します。

--mac パラメータは、ネットワーク インターフェイスの MAC アドレスを指定します。--live パラメータは、コマンドが実行しているドメインに影響を与えることを示します。

（注）
使用可能なオプションの詳細については、virsh の公式ドキュメントを参照してください。

例：

```
virsh # attach-interface --domain asav-network --type bridge --source br_hpi --model virtio --mac 52:55:04:4b:59:2f --live
```

（注）
ASA 仮想 でインターフェイス コンフィギュレーション モードを使用して、トラフィックの送受信インターフェイスを設定して有効化します。詳細については、『Cisco ASA シリーズ CLI コンフィギュレーション ガイド（一般的な操作）』の「*Basic Interface Configuration*」の章を参照してください。

ステップ 3 インターフェイスを削除するには、**detach-interface** コマンドを使用します。
detach-interface{ **--domain** *domain* **--type** *type* **--mac** *mac* **--live**}

（注）
使用可能なオプションの詳細については、*virsh* の公式ドキュメントを参照してください。

例：

```
virsh # detach-interface --domain asav-network --type bridge --mac 52:55:04:4b:59:2f --live
```

パフォーマンスの調整

KVM 構成でのパフォーマンスの向上

KVM ホストの設定を変更することによって、KVM 環境内の ASA 仮想 のパフォーマンスを向上させることができます。これらの設定は、ホスト サーバー上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。

CPU ピンニングの有効化

ASA 仮想 では、KVM 環境での ASA 仮想 のパフォーマンスを向上させるために KVM CPU アフィニティオプションを使用する必要があります。プロセッサアフィニティ（CPU ピニング）により、プロセスまたスレッドと中央処理装置（CPU）や幅広い CPU 間のバインドとバインド解除が可能になり、任意の CPU ではなく、指定された CPU でのみプロセスまたはスレッドが実行されるようになります。

ピン接続されていないインスタンスでピン接続されているインスタンスのリソース要件が使用されないようにするために、CPU ピニングを使用しないインスタンスとは別のホストに CPU ピニングを使用するインスタンスを展開するようにホスト集約を設定します。

注目 NUMA トポロジを持たないインスタンスと同じホストに NUMA トポロジを持つインスタンスを展開しないでください。

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。

手順

ステップ 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。

例：

```
virsh nodeinfo
```

ステップ 2 使用可能な vCPU の数を確認します。

例：

```
virsh capabilities
```

ステップ 3 vCPU をプロセッサ コアのセットにピンニングします。

例：

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

virsh vcpupin コマンドは、ASA 仮想 上の vCPU ごとに実行する必要があります。次の例は、vCPU が 4 個の ASA 仮想 構成を使用し、ホストに 8 個のコアが搭載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 asav 0 2
virsh vcpupin asav 1 3
virsh vcpupin asav 2 4
virsh vcpupin asav 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 サーバーの 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サーバーを使用している場合は、複数のソケットにまたがる CPU ピンニングを設定しないでください。

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。

NUMA のガイドライン

Non-uniform Memory Access（NUMA）は、マルチプロセッサシステムのプロセッサに対するメインメモリモジュールの配置について記述する共有メモリアーキテクチャです。プロセッサが

自身のノード（リモートメモリ）内に存在しないメモリにアクセスする場合は、ローカルメモリにアクセスする場合よりも低速の速度で、NUMA 接続を介してデータを転送する必要があります。

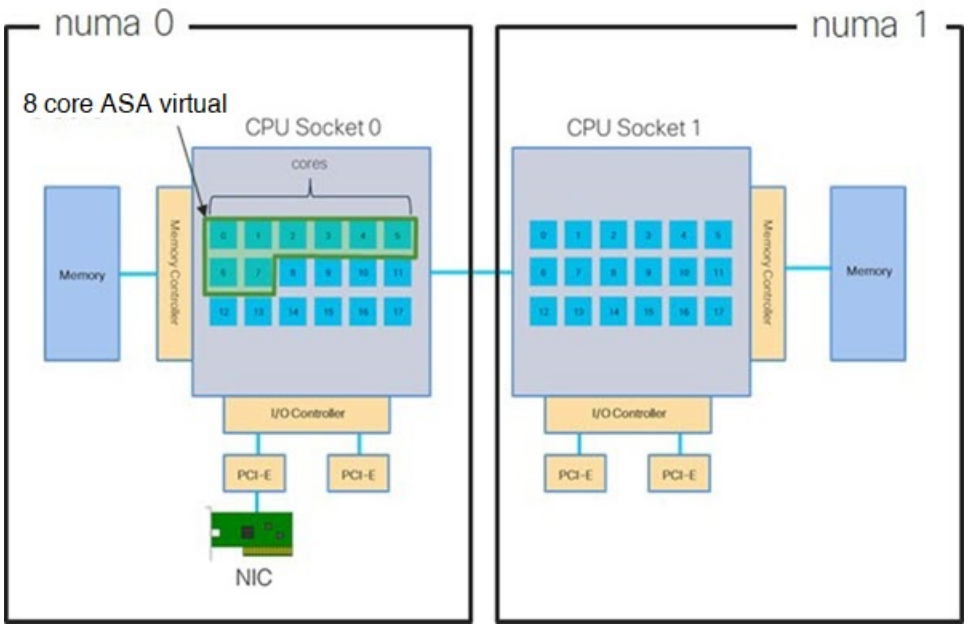
X86サーバーアーキテクチャは、複数のソケットおよびソケット内の複数のコアで構成されています。各 CPU ソケットとそのメモリおよび I/O が、NUMA ノードと呼ばれます。メモリからパケットを効率的に読み取るには、ゲストアプリケーションおよび関連付けられている周辺機器（NIC など）が同じノード内に存在する必要があります。

最適な ASA 仮想 パフォーマンスを実現するには：

- ASA 仮想 マシンは、1つの NUMA ノード上で実行する必要があります。1つの ASA 仮想 が2つのソケットで実行されるように導入されている場合、パフォーマンスは大幅に低下します。
- 8 コア ASA 仮想（図 2 : 8 コア ASA 仮想 NUMA アーキテクチャの例 （20 ページ））では、ホストCPUの各ソケットが、それぞれ8個以上のコアを備えている必要があります。サーバー上で実行されている他の VM についても考慮する必要があります。
- 16 コア ASA 仮想（図 3 : 16 コア ASA 仮想 NUMA アーキテクチャの例 （21 ページ））では、ホスト CPU 上の各ソケットが、それぞれ 16 個以上のコアを備えている必要があります。サーバー上で実行されている他の VM についても考慮する必要があります。
- NIC は、ASA 仮想 マシンと同じ NUMA ノード上にある必要があります。

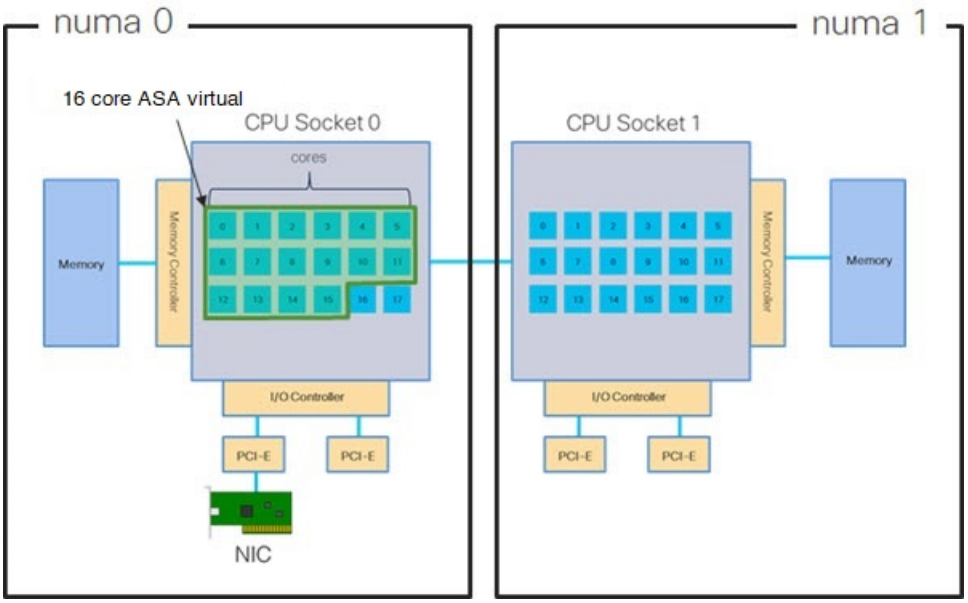
次の図は、2 つの CPU ソケットがあり、各 CPU に 18 個のコアが搭載されているサーバーを示しています。8 コア ASA 仮想 では、ホスト CPU の各ソケットに最低 8 個のコアが必要です。

図 *2 : 8* コア *ASA* 仮想 *NUMA* アーキテクチャの例



次の図は、2 つの CPU ソケットがあり、各 CPU に 18 個のコアが搭載されているサーバーを示しています。16 コア ASA 仮想 では、ホスト CPU の各ソケットに最低 16 個のコアが必要です。

図 *3 : 16* コア *ASA* 仮想 *NUMA* アーキテクチャの例



NUMA の最適化

理想的には、ASA 仮想 マシンは、NIC が動作しているノードと同じ NUMA ノード上で実行する必要があります。手順は次のとおりです。

1. 「lstopo」を使用して NIC がオンになっているノードを判別し、ノードの図を表示します。NIC を見つけて、どのノードが接続されているかをメモします。
2. KVM ホストで、`virsh list` を使用して ASA 仮想 を検出します。
3. `virsh edit <VM Number>` を使用して VM を編集します。
4. 選択したノードに ASA 仮想 を配置します。次の例では、18 コアノードを想定しています。

   ノード 0 への配置：

   ```
   <vcpu placement='static' cpuset='0-17'>16</vcpu>
   <numatune>
       <memory mode='strict' nodeset='0'/>
   </numatune>
   ```

   ノード 1 への配置：

   ```
   <vcpu placement='static' cpuset='18-35'>16</vcpu>
   <numatune>
       <memory mode='strict' nodeset='1'/>
   </numatune>
   ```

5. .xml の変更を保存し、ASA 仮想 マシンの電源を再投入します。
6. VM が目的のノードで実行されていることを確認するには、`ps aux | grep <name of your ASAv VM>` を実行して、プロセス ID を取得します。
7. `sudo numastat -c <ASAv VM Process ID>` を実行して、ASA 仮想 マシンが適切に配置されているか確認します。

KVM での NUMA 調整の使用に関する詳細については、RedHat のドキュメント『9.3. libvirt NUMA Tuning』を参照してください。

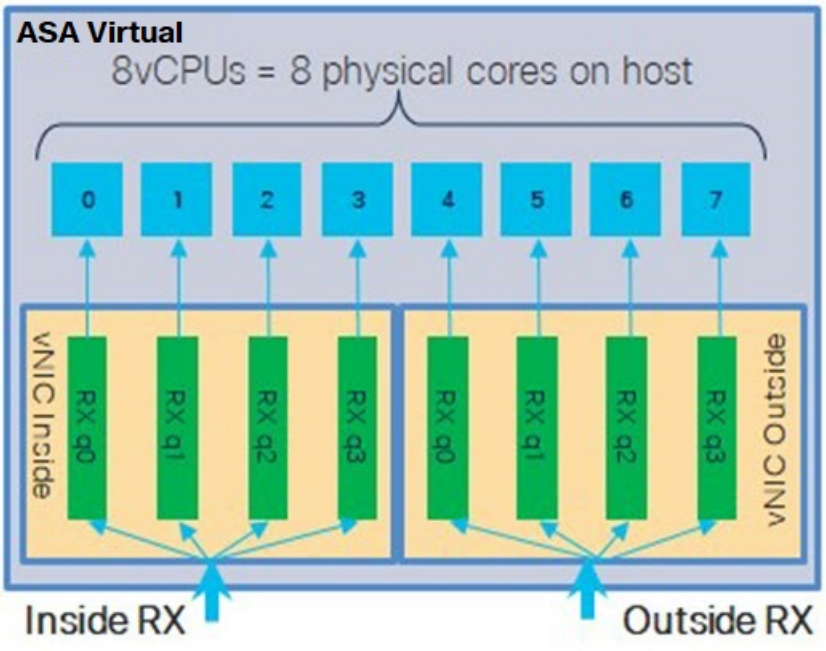
Receive Side Scaling（RSS）用の複数の RX キュー

ASA 仮想 は、複数のプロセッサコアにネットワーク受信トラフィックを分散するためにネットワークアダプタによって使用されるテクノロジーである Receive Side Scaling（RSS）をサポートしています。最大スループットを実現するには、各 vCPU（コア）に独自の NIC RX キューが設定されている必要があります。一般的な RA VPN 展開では、1つの内部/外部ペアのインターフェイスを使用する場合があることに注意してください。

重要 複数の RX キューを使用するには、ASA 仮想 バージョン 9.13(1) 以降が必要です。KVM の場合、*libvirt* のバージョンは 1.0.6 以降である必要があります。

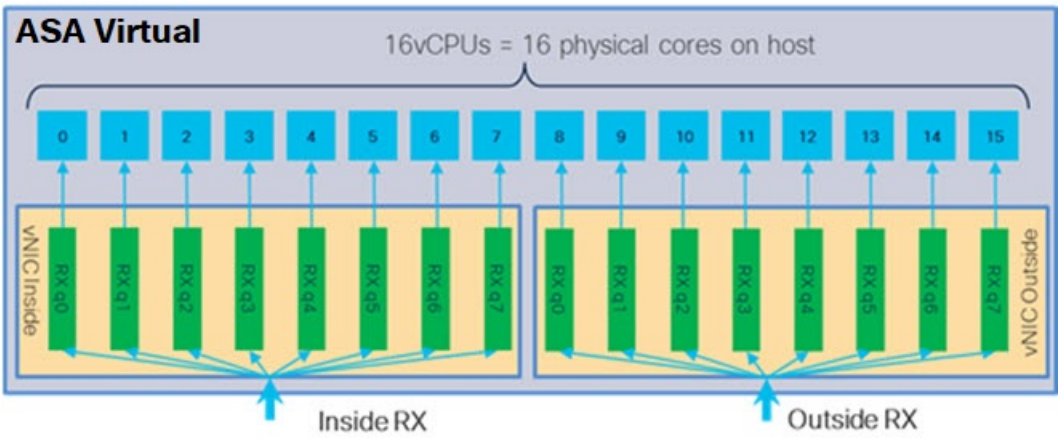
内部/外部ペアのインターフェイスを持つ 8 コア VM の場合、図 4 : 8 コア ASA 仮想 RSS RX キュー （22 ページ）に示すように、各インターフェイスには 4 つの RX キューがあります。

図 *4 : 8* コア *ASA* 仮想 *RSS RX* キュー



内部/外部ペアのインターフェイスを持つ 16 コア VM の場合、図 5 : 16 コア ASA 仮想 RSS RX キュー （23 ページ）に示すように、各インターフェイスには 8 つの RX キューがあり ます。

図 *5 : 16* コア *ASA* 仮想 *RSS RX* キュー



次の表に、KVM 用の ASA 仮想 の vNIC およびサポートされている RX キューの数を示します。サポートされている vNIC の説明については、推奨される vNIC （2 ページ）を参照してください。

表 *1 : KVM* で推奨される *NIC/vNIC*

NIC カード	vNIC ドライバ	ドライバテクノロジー	RX キューの数	パフォーマンス
x710	i40e	PCI パススルー	8（最大）	x710 の PCI パススルーおよび SR-IOV モードは、最適なパフォーマンスを提供します。通常、仮想展開では、複数の VM 間で NIC を共有できるため、SR-IOV が推奨されます。
	i40evf	SR-IOV	8	
x520	ixgbe	PCI パススルー	6	x520 NIC は、x710 よりも 10 ～ 30% パフォーマンスが低くなります。X520 の PCI パススルーおよび SR-IOV モードは、同様のパフォーマンスを提供します。通常、仮想展開では、複数の VM 間で NIC を共有できるため、SR-IOV が推奨されます。
	ixgbe-vf	SR-IOV	2	

NIC カード	vNIC ドライバ	ドライバテクノロジー	RX キューの数	パフォーマンス
該当なし	virtio	準仮想化	8（最大）	ASAv100 には推奨されません。 その他の展開については、KVM での Virtio のマルチキューサポートの有効化 （24 ページ） を参照してください。

KVM での Virtio のマルチキューサポートの有効化

次の例は、libvirt xml を編集するために、Virtio NIC RX キューの数を 4 に設定する方法を示しています。

```
<interface type='bridge'>
 <mac address='52:54:00:43:6e:3f'/>
 <source bridge='clients'/>
 <model type='virtio'/>
 <driver name='vhost' queues='4'/>
 <address type='pci' domain='0x0000' bus='0x00' slot='0x04' function='0x0'/>
</interface>
```

重要 複数の RX キューをサポートするには、*libvirt* のバージョンが 1.0.6 以降である必要があります。

VPN の最適化

ASA 仮想 で VPN パフォーマンスを最適化するための追加の考慮事項は、次のとおりです。

- IPSec のスループットは DTLS よりも高くなります。
- GCM 暗号には、CBC の約 2 倍のスループットがあります。

SR-IOV インターフェイスのプロビジョニング

SR-IOV を使用すれば、複数の VM でホスト内部の 1 台の PCIe ネットワーク アダプタを共有することができます。SR-IOV は次の機能を定義しています。

- 物理機能（PF）：PF は、SR-IOV 機能を含むフル PCIe 機能です。これらは、ホスト サーバー上の通常のスタティック NIC として表示されます。
- 仮想機能（VF）：VF は、データ転送を支援する軽量 PCIe 機能です。VF は、PF から抽出され、PF を介して管理されます。

VF は、仮想化されたオペレーティング システム フレームワーク内の ASA 仮想 マシンに最大 10 Gbps の接続を提供できます。このセクションでは、KVM 環境で VF を設定する方法について説明します。ASA 仮想 上の SR-IOV サポートについては、ASA 仮想 と SR-IOV インターフェイスのプロビジョニングを参照してください。

ASAv5 および ASAv10 で最適なパフォーマンスを得るため、VMXNET3 ドライバを強く推奨します。さらに、SR-IOV インターフェイスを組み合わせて（インターフェイスが混在した状態で）使用すると、特により多くの CPU コアとリソースを割り当てることで、ASA 仮想 とのネットワークパフォーマンスが向上します。

SR-IOV インターフェイスのプロビジョニングに関する要件

SR-IOV をサポートする物理 NIC がある場合、SR-IOV 対応 VF または仮想 NIC（vNIC）を ASA 仮想 インスタンスにアタッチできます。SR-IOV は、BIOS だけでなく、ハードウェア上で実行しているオペレーティング システム インスタンスまたはハイパーバイザでのサポートも必要です。KVM 環境で実行中の ASA 仮想 用の SR-IOV インターフェイスのプロビジョニングに関する一般的なガイドラインのリストを以下に示します。

- ホスト サーバーには SR-IOV 対応物理 NIC が必要です。SR-IOV インターフェイスに関するガイドラインと制限事項を参照してください。
- ホスト サーバーの BIOS で仮想化が有効になっている必要があります。詳細については、ベンダーのマニュアルを参照してください。
- ホスト サーバーの BIOS で IOMMU グローバル サポートが SR-IOV に対して有効になっている必要があります。詳細については、ハードウェア ベンダーのマニュアルを参照してください。
- SR-IOV インターフェイスを使用する KVM 上の ASA 仮想 では、インターフェイスタイプの混在がサポートされています。管理インターフェイスには SR-IOV または VMXNET3 を使用し、データインターフェイスには SR-IOV を使用することができます。

KVM ホスト BIOS とホスト OS の変更

このセクションでは、KVM システム上の SR-IOV インターフェイスのプロビジョニングに関するさまざまなセットアップ手順と設定手順を示します。このセクション内の情報は、Intel Ethernet Server Adapter X520 - DA2 を使用した Cisco UCS C シリーズ サーバー上の Ubuntu 14.04 を使用して、特定のラボ環境内のデバイスから作成されたものです。

始める前に

- SR-IOV 互換ネットワーク インターフェイス カード（NIC）が取り付けられていることを確認します。
- Intel 仮想化テクノロジー（VT-x）機能と VT-d 機能が有効になっていることを確認します。

（注） システム メーカーによっては、これらの拡張機能がデフォルトで無効になっている場合があります。システムごとに BIOS 設定にアクセスして変更する方法が異なるため、ベンダーのマニュアルでプロセスを確認することをお勧めします。

- オペレーティング システムのインストール中に、Linux KVM モジュール、ライブラリ、ユーザ ツール、およびユーティリティのすべてがインストールされていることを確認します。前提条件 （6 ページ）を参照してください。
- 物理インターフェイスが稼働状態であることを確認します。ifconfig <*ethname*> を使用して確認します。

手順

ステップ 1 "root" ユーザー アカウントとパスワードを使用してシステムにログインします。

ステップ 2 Intel VT-d が有効になっていることを確認します。

例：

```
kvmuser@kvm-host:/$ dmesg | grep -e DMAR -e IOMMU
[    0.000000] ACPI: DMAR 0x000000006F9A4C68 000140 (v01 Cisco0 CiscoUCS 00000001 INTL 20091013)
[    0.000000] DMAR: IOMMU enabled
```

最後の行は、VT-d が有効になっていることを示しています。

ステップ 3 */etc/default/grub* 設定ファイル内の GRUB_CMDLINE_LINUX エントリに *intel_iommu=on* パラメータを付加することによって、カーネル内の Intel VT-d をアクティブにします。

例：

```
# vi /etc/default/grub
...
GRUB_CMDLINE_LINUX="nofb splash=quiet console=tty0 ... intel_iommu=on"
...
```

（注）
AMD プロセッサを使用している場合は、代わりに、*amd_iommu=on* をブート パラメータに付加します。

ステップ 4 iommu の変更を有効にするためにサーバーをリブートします。

例：

```
> shutdown -r now
```

ステップ 5 次の形式を使用して *sysfs* インターフェイス経由で *sriov_numvfs* パラメータに適切な値を書き込むことによって、VF を作成します。

```
#echo n > /sys/class/net/device name/device/sriov_numvfs
```

サーバーの電源を入れ直すたびに必要な数の VF が作成されるようにするには、*/etc/rc.d/* ディレクトリに配置されている *rc.local* ファイルに上記コマンドを付加します。Linux OS は、ブートプロセスの最後で *rc.local* スクリプトを実行します。

たとえば、ポートあたり 1 つの VF を作成するケースを以下に示します。お使いのセットアップではインターフェイスが異なる可能性があります。

例：

```
echo '1' > /sys/class/net/eth4/device/sriov_numvfs
echo '1' > /sys/class/net/eth5/device/sriov_numvfs
echo '1' > /sys/class/net/eth6/device/sriov_numvfs
echo '1' > /sys/class/net/eth7/device/sriov_numvfs
```

ステップ 6 サーバーをリブートします。

例：

```
> shutdown -r now
```

ステップ 7 *lspci* を使用して、VF が作成されたことを確認します。

例：

```
 > lspci | grep -i "Virtual Function"
kvmuser@kvm-racetrack:~$ lspci | grep -i "Virtual Function"
0a:10.0 Ethernet controller: Intel Corporation 82599 Ethernet Controller Virtual Function (rev 01)
0a:10.1 Ethernet controller: Intel Corporation 82599 Ethernet Controller Virtual Function (rev 01)
0a:10.2 Ethernet controller: Intel Corporation 82599 Ethernet Controller Virtual Function (rev 01)
0a:10.3 Ethernet controller: Intel Corporation 82599 Ethernet Controller Virtual Function (rev 01)
```

（注）
ifconfig コマンドを使用して、新しいインターフェイスを表示します。

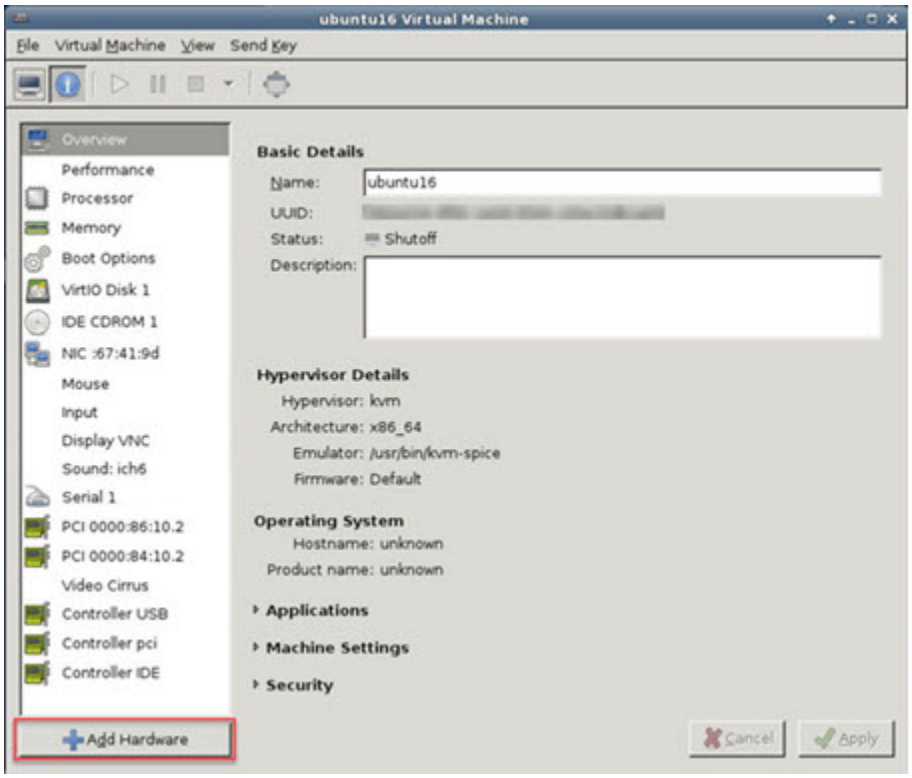
ASA 仮想 への PCI デバイスの割り当て

VF を作成したら、PCI デバイスを追加するのと同様に、VF を ASA 仮想 に追加できます。次の例では、グラフィカル **virt-manager** ツールを使用して、イーサネット VF コントローラを ASA 仮想 に追加する方法について説明します。

手順

ステップ 1 ASA 仮想 を開いて、[Add Hardware] ボタンをクリックし、新しいデバイスを仮想マシンに追加します。

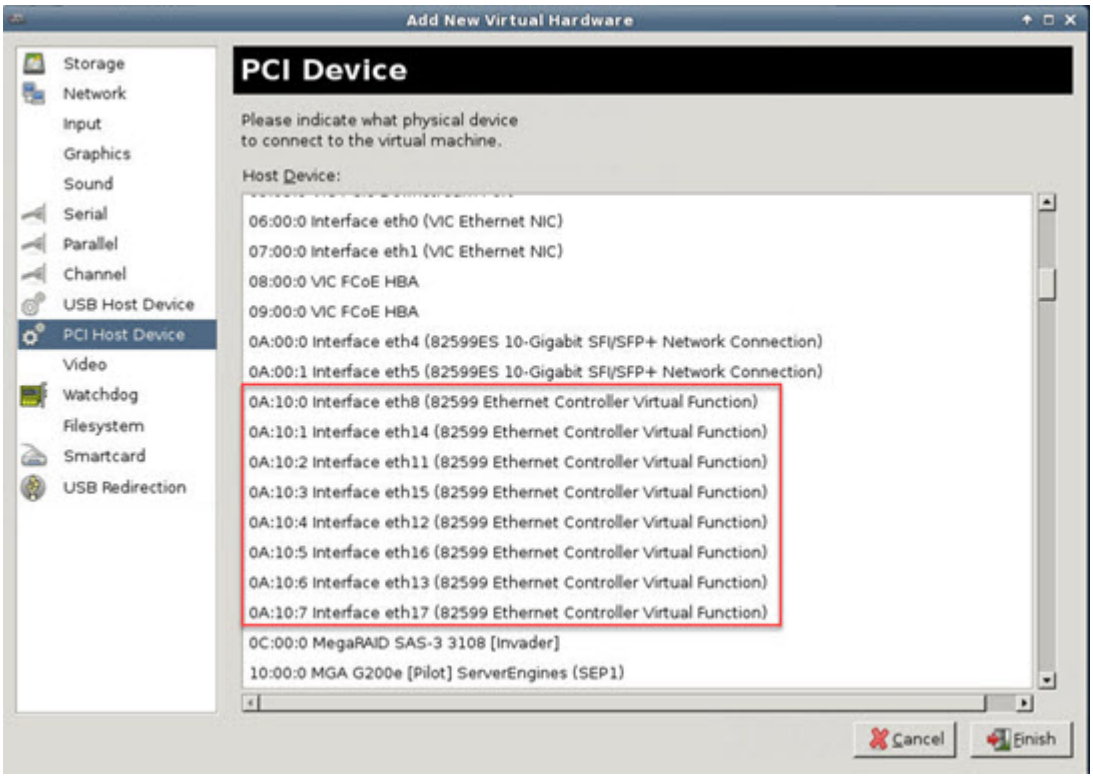
図 *6 :* ハードウェアの追加



ステップ 2 左ペインの [Hardware] リストで [PCI Host Device] をクリックします。

VF を含む PCI デバイスのリストが中央ペインに表示されます。

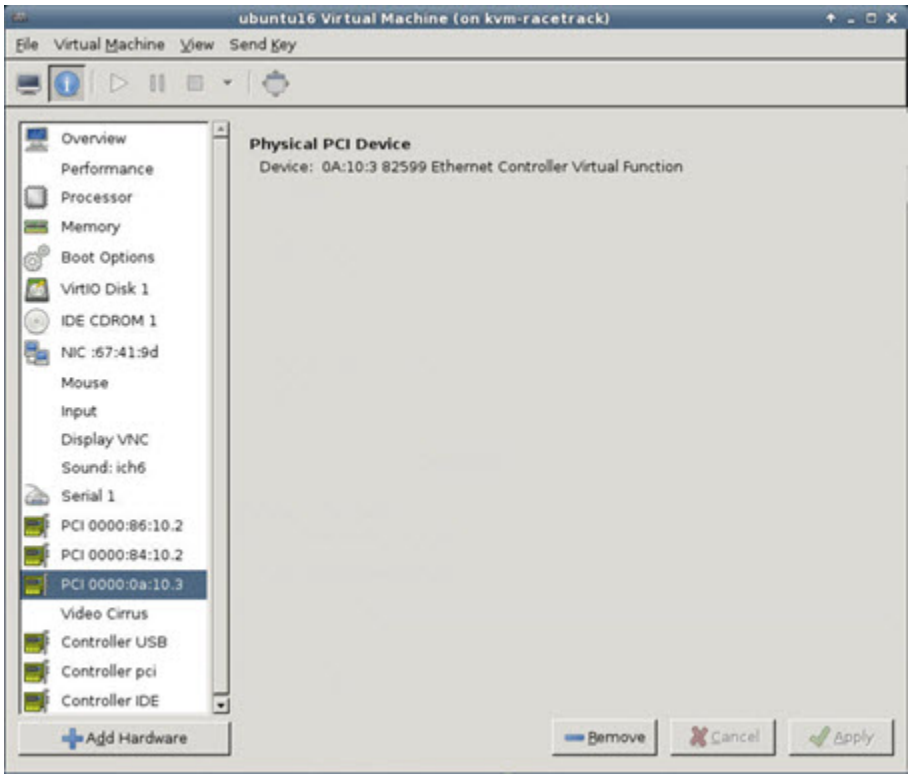
図 *7*:仮想機能のリスト



ステップ 3 使用可能な仮想機能のいずれかを選択して、[Finish] をクリックします。

PCI デバイスがハードウェア リストに表示されます。デバイスの記述が Ethernet Controller Virtual Function になっていることに注意してください。

図 *8 :* 追加された仮想機能



次のタスク

- ASA 仮想 コマンド ラインから、**show interface** コマンドを使用して、新しく設定したインターフェイスを確認します。
- ASA 仮想 でインターフェイス コンフィギュレーション モードを使用して、トラフィックの送受信インターフェイスを設定して有効化します。詳細については、『Cisco Secure Firewall ASA シリーズ CLI コンフィギュレーション ガイド（一般的な操作）』の「*Basic Interface Configuration*」の章を参照してください。

CPU 使用率とレポート

CPU使用率レポートには、指定された時間内に使用されたCPUの割合の要約が表示されます。通常、コアはピーク時以外には合計 CPU 容量の約 30 ～ 40% で動作し、ピーク時は約 60 ～ 70% の容量で動作します。

重要 9.13(1) 以降では、サポートされているすべての ASA Virtual vCPU/メモリ構成ですべての ASA Virtual ライセンスを使用できるようになり、ASA Virtual を使用しているお客様は、さまざまな VM リソースフットプリントで実行できます。

ASA Virtual の vCPU 使用率

ASA Virtual の vCPU 使用率には、データパス、制御ポイント、および外部プロセスで使用されている vCPU の量が表示されます。

vSphere で報告される vCPU の使用率には、ASA Virtual の使用率に加えて、次のものが含まれます。

- ASA Virtual アイドル時間
- ASA Virtual マシンに使用された %SYS オーバーヘッド
- vSwitch、vNIC および pNIC の間を移動するパケットのオーバーヘッド。このオーバーヘッドは非常に大きくなる場合があります。

CPU 使用率の例

CPU 使用率の統計情報を表示するには、**show cpu usage** コマンドを使用します。

例

```
Ciscoasa#show cpu usage

CPU □□□□5□□□ 1%□1 □□□ 2%□5 □□□ 1%
```

報告された vCPU の使用率が大幅に異なる例を次に示します。

- ASA Virtual レポート：40%
- DP：35%
- 外部プロセス：5%
- ASA（ASA Virtual レポート）：40%
- ASA アイドル ポーリング：10%
- オーバーヘッド：45%

オーバーヘッドは、ハイパーバイザ機能の実行、および vSwitch を使用した NIC と vNIC の間のパケット転送に使用されています。

KVM CPU 使用率レポート

値は、

```
virsh cpu-stats domain --total start count
```

コマンドを実行すると、指定されたゲスト仮想マシンのCPU統計情報が表示されます。デフォルトでは、すべてのCPUの統計と合計が表示されます。`--total`オプションを指定すると、合計統計のみ表示されます。`--count`オプションを指定すると、count 個の CPU の統計のみ表示されます。

OProfile、topなどのツールを実行すると、ハイパーバイザとVMの両方のCPU使用率を含む、特定の KVM VM の合計 CPU 使用率が表示されます。同様に、Xen VMM に固有の XenMon などのツールの場合、Xenハイパーバイザ、つまりDom 0の合計CPU使用率が表示されますが、VM ごとのハイパーバイザ使用率には分割されません。

これらのツールとは別に、OpenNebula などのクラウド コンピューティング フレームワークには、VM によって使用される仮想 CPU の割合の大まかな情報のみを提供する特定のツールが存在します。

ASA Virtual と KVM のグラフ

ASA Virtual と KVM の間には CPU % の数値に違いがあります。

- KVM グラフの数値は ASA Virtual の数値よりも常に大きくなります。
- KVMではこの値は「%CPU usage」と呼ばれ、ASA Virtualではこの値は「%CPU utilization」と呼ばれます。

用語「%CPU utilization」と「%CPU usage」は別のものを意味しています。

- CPU utilization は、物理 CPU の統計情報を提供します。
- CPU usage は CPU のハイパースレッディングに基づいた論理 CPU の統計情報を提供します。しかし、1 つの vCPU のみが使用されるため、ハイパースレッディングは動作しません。

KVM では「%CPU usage」は次のように計算されます。

アクティブに使用された仮想 CPU の量。使用可能な CPU の合計に対する割合として指定されます。

この計算は、ホストから見た CPU 使用率であり、ゲスト オペレーティング システムから見た CPU 使用率ではありません。また、これは仮想マシンで使用可能なすべての仮想 CPU の平均 CPU 使用率になります。

たとえば、1 個の仮想 CPU を搭載した 1 つの仮想マシンが、4 個の物理 CPU を搭載した 1 台のホストで実行されており、その CPU 使用率が 100% の場合、仮想マシンは、1 個の物理 CPU をすべて使用しています。仮想 CPU の使用率は、「MHz 単位の使用率 / 仮想 CPU の数 x コア周波数」として計算されます。

KVM での DPU を使用した IPSec トラフィックの拡張およびオフロード

KVM で実行されているデータ処理単位（DPU）での Internet Protocol Security（IPSec）トラフィックの拡張およびオフロードにより、暗号化を多用するパケット処理がホスト CPU から専用 DPU ハードウェアに移行されます。この機能は、パフォーマンスを向上させ、CPU オーバーヘッドを削減し、電力効率を高めるために、最新のデータセンターに導入されています。

大規模フローのオフロード

データセンターのサポートされているデバイス上で ASA Virtual を展開する場合は、超高速パスにオフロードするトラフィックを識別できます。オフロードによって、大容量ファイルの転送など、データ集約型アプリケーションのパフォーマンスを向上させることができます。この機能は、リリース 10.0.0 以降、KVM 上の ASA Virtual 展開でサポートされています。

オフロードされる前に、ASA Virtual は、接続の確立時にアクセスルールやインスペクションなどの通常のセキュリティ処理を最初に適用します。ASA Virtual のセッションも切断されます。一旦接続が確立されると、オフロードの対象であれば、さらなる処理が ASA Virtual ではなくネットワーク インターフェイス カード（NIC）で行われます。

オフロードされたフローは、基本的な TCP フラグとオプションのチェック、設定した場合にはチェックサムの確認などの、制限されたステートフルインスペクションを受信し続けます。システムは必要に応じてさらなる処理のためにファイアウォール システムへのパケットを選択的に増やすことができます。

オフロードが可能なフローを識別するには、フローオフロードサービスを適用するサービスポリシールールを作成します。フローは、パケット内の以下のフィールドと一致する場合にオフロードされます。

- IPv4 送信元および宛先アドレス。
- TCP ポートおよび UDP ポート。
- 標準または 802.1Q タグ付きイーサネット フレームのみ。
- トランスペアレントモードのみ 。インターフェイスを 2 つだけ含むブリッジ グループのマルチキャスト フロー。

IPsec フローのオフロード

IPsec フローのオフロードを使用するように、サポートするデバイスモデルを設定できます。

オフロード操作は、特に、入力の事前復号および復号処理と出力の事前暗号化および暗号化処理に関連しています。システムソフトウェアは、セキュリティポリシーを適用するための内部フローを処理します。

IPsec フローオフロードは、デバイスの仮想トンネルインターフェイス（VTI）ループバックインターフェイスが有効になっている場合にも使用されます。

クラスタ分散型サイト間 VPN モードの非対称フローの場合、IPsec フローオフロードにより、フローオーナーは、クラスタ制御リンクを介して転送された IPsec トラフィックを復号できます。この機能は設定可能ではありません。IPsec フローオフロードを有効にすると常に使用できます。

KVM での ASA Virtual への DOCA のインストールと設定

前提条件

ハードウェア要件：

- UCS-KVM のみ：システムが UCS-KVM 用に設定されていることを確認します。
- UCS-M7：このモデルには DPU 用の統合電源装置が付属しています。

（注） BF3 カードは 幅 x16 の PCI スロットに配置する必要があります。

BF3 カードの PCI スロット幅を特定し、そのカードが幅 x16 スロットに配置されていることを確認するには、次の手順を使用できます。

PCI スロット幅を確認する手順：

ステップ 1：`lspci` コマンドを使用します。

ホストで次のコマンドを使用して、PCI スロットのリンクステータスを確認します。

```
sudo lspci -s -xxxvvv | grep -i sta
```

出力で *LnkSta* フィールドを探します。このフィールドは、PCIe スロットの速度と幅を示します。たとえば、`LnkSta: Speed 8GT/s, Width x16` のように表示されます。

この表示により、カードが幅 x16 で実行されていることを確認できます。

ステップ 2：カードを別のスロットにスワップします。

BF3 カードが幅 x16 スロットに配置されていない場合は、BF3 カードを別の PCIe スロットに移動し、`lspci` コマンドを再度使用して、幅が x16 に変更されるかどうかを確認することを推奨します。

ソフトウェア要件：

- DOCA バージョン：2.9 以降。
- カーネルバージョン：6.8 以降。
- ASA Virtual バージョン：9.24.1.1 以降。

ファームウェアの要件：

BF3 ファームウェア：バージョン 32.41.1300。

これはコマンド `flint -d /dev/mst/mt41692_pciconf0 query`を使用して確認できます。

互換性の問題を回避するため、ファームウェアが指定されているバージョンに更新されていることを確認してください。

UCS サーバーへの DOCA のインストール

手順

ステップ 1 次のコマンドを使用して、Ubuntu 22.04 を実行している UCS サーバーに NVIDIA DOCA をインストールします。

```
wget
https://www.mellanox.com/downloads/DOCA/DOCA_v2.9.1/host/doca-host_2.9.1-018000-24.10-ubuntu2204_amd64.deb
sudo dpkg -i doca-host_2.9.1-018000-24.10-ubuntu2204_amd64.deb
sudo apt-get update
sudo apt-get -y install doca-all
mst start
mlxfwmanager -query wget
https://www.mellanox.com/downloads/firmware/fw-BlueField-3-rel-32_43_1014-900-9D3B6-00CV-A_Ax-NVME-20.4.1-UEFI-21.4.13-UEFI-22.4.14-UEFI-14.36.16-FlexBoot-3.7.500.signed.bin.zip
mkdir doca29
mv
fw-BlueField-3-rel-32_43_1014-900-9D3B6-00CV-A_Ax-NVME-20.4.1-UEFI-21.4.13-UEFI-22.4.14-UEFI-14.36.16-FlexBoot-3.7.500.signed.bin.zip
 doca29
cd doca29
unzip
fw-BlueField-3-rel-32_43_1014-900-9D3B6-00CV-A_Ax-NVME-20.4.1-UEFI-21.4.13-UEFI-22.4.14-UEFI-14.36.16-FlexBoot-3.7.500.signed.bin.zip
flint -d /dev/mst/mt41692_pciconf0 -i
fw-BlueField-3-rel-32_43_1014-900-9D3B6-00CV-A_Ax-NVME-20.4.1-UEFI-21.4.13-UEFI-22.4.14-UEFI-14.36.16-FlexBoot-3.7.500.signed.bin
 b
```

ステップ 2 ホストの電源を再投入します。

大規模なページの設定

大規模なページを設定するには、UCS サーバーで次のコマンドを使用します。

```
echo 2048 > /sys/devices/system/node/node0/hugepages/hugepages-2048kB/nr_hugepages
echo 2048 > /sys/devices/system/node/node1/hugepages/hugepages-2048kB/nr_hugepages
```

仮想機能の作成

Mellanox 側の設定：

UCS サーバーで `lshw -c network -businfo` コマンドを使用してホストに存在する PCI インターフェイスを確認し、BF3 の PCI インターフェイスを選択して、次のコマンドを使用してモードを変更し、VF を作成します。

```
devlink dev eswitch show pci/0000:2a:00.0
devlink dev eswitch show pci/0000:2a:00.1
echo 0 > /sys/bus/pci/devices/0000:2a:00.0/sriov_numvfs
Cisco Confiden□□al
```

```
echo 0 > /sys/bus/pci/devices/0000:2a:00.1/sriov_numvfs
/opt/mellanox/iproute2/sbin/devlink dev eswitch set pci/0000:2a:00.0 mode switchdev
/opt/mellanox/iproute2/sbin/devlink dev eswitch set pci/0000:2a:00.1 mode switchdev
echo 16 > /sys/bus/pci/devices/0000:2a:00.0/sriov_numvfs
echo 16 > /sys/bus/pci/devices/0000:2a:00.1/sriov_numvfs
```

OVS ブリッジを作成し、VF レプレゼンタをブリッジに追加する

次の OVS 設定は、KVM ホストで行う必要があります。

次に、設定手順の例を示します。

- OVS ブリッジを作成してポートを追加します。

```
ovs-vsctl list-br
ovs-vsctl add-br ovs-1
ovs-vsctl add-port ovs-1 ens1f0v4
ovs-vsctl add-port ovs-1 ens1f0np0
ovs-vsctl add-br ovs-2
ovs-vsctl add-port ovs-2 ens1f1v4
ovs-vsctl add-port ovs-2 ens1f1np1
```

- ブリッジからポートを削除します（必要な場合）。

```
ovs-vsctl del-port ovs-1 ens1f0v4
```

- ハードウェアオフロードを有効にします。

  ovs-vsctl set Open_vSwitch . other_config:hw-offload=true

VF での IPSec の有効化

手順

ステップ 1 暗号/IPsec 用の PCI を特定します。

ホストで次のコマンドを使用して、暗号および IPSec を有効にする PCI を確認します。

```
/opt/mellanox/iproute2/sbin/devlink port show pci/0000:3d:01.0| more
```

ステップ 2 IPsec を有効にします。

IPsec を有効にするには、次のコマンドを使用します。

```
echo 0000:3d:01.5 > /sys/bus/pci/drivers/mlx5_core/unbind
echo none > /sys/class/net/$PF/compat/devlink/encap
/opt/mellanox/iproute2/sbin/devlink dev eswitch set pci/0000:3d:00.0 mode switchdev
/opt/mellanox/iproute2/sbin/devlink port function set pci/0000:3d:00.0/11 ipsec_packet enable
/opt/mellanox/iproute2/sbin/devlink port function set pci/0000:3d:00.0/11 ipsec_crypto enable
echo 0000:3d:01.5 > /sys/bus/pci/drivers/mlx5_core/bind
```

VF レプレゼンタに対するキューサイズの設定

手順

ステップ 1 キューサイズを確認します。

インターフェイスの現在のキューサイズを表示するには、ethtool -l ens2f0np0 コマンドを使用します。

例：`root@FF3-248: /home/admin1# ethtool -l ens2f0np0`

ens2f0np0 のチャンネルパラメータ：

```
Pre-set maximums:
RX: n/a
TX: n/a
Other: n/a
Combined: 63
Current settings:
RX: n/a
TX: n/a
Other: n/a
Combined: 1
```

ステップ 2 キューサイズを大きくします。

キューサイズを大きくするには、次のコマンドを使用します。

```
ethtool -L ens2f0np0 combined 63
```

KVM での ASA Virtual の展開

KVM で ASA Virtual を展開する場合は、「*KVM* での *ASA Virtual* の展開」の章を参照してください。

（注） UCS を再起動するたびに、以下に示す手順と DOCA のインストール手順を除いて、すべての設定を再適用する必要があります。

```
mlxconfig -d /dev/mst/mt41692_pciconf0.1 s INTERNAL_CPU_MODEL=1
INTERNAL_CPU_PAGE_SUPPLIER=1 INTERNAL_CPU_ESWITCH_MANAGER=1 INTERNAL_CPU_IB_VPORT0=1
INTERNAL_CPU_OFFLOAD_ENGINE=1

mlxconfig -d /dev/mst/mt41692_pciconf0 s INTERNAL_CPU_MODEL=1 INTERNAL_CPU_PAGE_SUPPLIER=1
INTERNAL_CPU_ESWITCH_MANAGER=1 INTERNAL_CPU_IB_VPORT0=1 INTERNAL_CPU_OFFLOAD_ENGINE=1
```

（注） この機能は Ubuntu OS – バージョン 22.04 LTS で検証済みです。

フロー オフロードの設定

フローオフロードを設定するには、サービスをイネーブルにしてから、オフロードする対象トラフィックを識別するサービス ポリシーを作成する必要があります。

手順

ステップ 1 次のコマンドを使用して、フローオフロードサービスを有効にします。

```
flow-offload enable
```

フローオフロードはデフォルトで有効になっています。

リロードが必要な場合、ヒットレスなモード変更を行うには、クラスタまたはフェールオーバーペアに関して特に考慮すべき事柄があります。

- クラスタリング：最初に制御ノードでコマンドを入力しますが、制御ノードをすぐにリブートしないでください。

  代わりに、まずクラスタの各ノードをリブートしてから、制御ノードに戻ってリブートします。次に、制御ノードでオフロードサービスポリシーを設定します。

- フェールオーバー：最初にアクティブユニット上でコマンドを入力しますが、アクティブユニットをすぐにリブートしないでください。代わりに、スタンバイ ユニットをリブートしてから、アクティブユニットをリブートします。次に、アクティブ ユニット上でオフロード サービス ポリシーを設定します。

  後でリロードする場合は、ctrl+c を入力してリロードをキャンセルしてください。

例：

```
ciscoasa(config)# flow-offload enable
INFO: DPU offload is enabled.
The new mode will take effect after the reboot.
Proceed with reload?
```

ステップ 2 オフロードする対象のトラフィックを識別するサービス ポリシー ルールを作成します。

フロー オフロードの対象となるトラフィックを識別する L3/L4 クラス マップを作成します。アクセス リストまたはポートによる照合は最も一般的なオプションです。

```
class-map name
match parameter
```

インターフェイス、ポリシーマップ、およびクラスマップの設定例：

```
interface TenGigabitEthernet0/0
nameif inside
security-level 100
ip address 20.0.0.1 255.255.255.0
!
interface TenGigabitEthernet0/1
nameif outside
security-level 0
ip address 10.1.2.5 255.255.255.0
access-list offload extended permit tcp 20.0.0.0 255.0.0.0 10.1.2.0 255.255.255.0 eq www
```

```
access-list offload extended permit tcp 10.1.2.0 255.255.255.0 20.0.0.0 255.0.0.0 eq www
access-list offload extended permit udp 20.0.0.0 255.0.0.0 10.1.2.0 255.255.255.0 eq 100
access-list offload extended permit udp 10.1.2.0 255.255.255.0 20.0.0.0 255.0.0.0 eq 100
access-group offload global
class-map flow_offload
match access-list offload
policy-map offload_policy
class flow_offload
set connection advanced-options flow-offload
set connection random-sequence-number disable
police output 100000000000 2560000000 conform-action transmit exceed-action transmit
police input 100000000000 2560000000 conform-action transmit exceed-action transmit
service-policy offload_policy interface inside
service-policy offload_policy interface outside
```

例：

```
hostname(config)# service-policy offload_policy interface outside
```

global キーワードはポリシー マップをすべてのインターフェイスに適用し、*interface* はポリシーを 1 つのインターフェイスに適用します。グローバル ポリシーは 1 つしか適用できません。インターフェイスのグローバル ポリシーは、そのインターフェイスにサービス ポリシーを適用することで上書きできます。各インターフェイスには、ポリシー マップを 1 つだけ適用できます。

例：

次に、10.1.1.0 255.255.255.224 サブネットからのすべての TCP トラフィックをオフロード対象として分類し、ポリシーを外部インターフェイスにアタッチする例を示します。

```
hostname(config)# access-list offload permit tcp 10.1.1.0 255.255.255.224 any
hostname(config)# class-map flow_offload
hostname(config-cmap)# match access-list offload
hostname(config)# policy-map offload_policy
hostname(config-pmap)# class flow_offload
hostname(config-pmap-c)# set connection advanced-options flow-offload
hostname(config)# service-policy offload_policy interface outside
```

IPsec フローオフロードの設定

IPsec フローオフロードはデフォルトで有効になっています。ただし、出力最適化はデフォルトでは有効になっていないため、この機能が必要な場合は設定する必要があります。

始める前に

IPsec フロー オフロードはグローバルに構成されます。選択したトラフィック フローに対して設定することはできません。

この機能を無効にするには、これらのコマンドの *no* 形式を使用します。

現在の設定状態を表示するには、`show flow-offload ipsec info` コマンドを使用します。

手順

ステップ 1 次のコマンドを使用して、IPsec フローオフロードを有効にします。

```
flow-offload-ipsec
```

ステップ 2 出力最適化を有効にすることで、データパスを最適化して、単一トンネルフローのパフォーマンスを向上させます。

```
flow-offload-ipsec egress-optimization
```

出力最適化の構成は、フロー オフロードとは別です。ただし、出力最適化を有効にしても、IPsec フローオフロードも有効にしないかぎり無意味です。出力最適化はデフォルトでは有効になっていません。

フロー オフロードの制限事項

次のタイプのフローはオフロードできません。

- IPv6 アドレッシングなど、IPv4 アドレッシングを使用しないフロー。
- TCP および UDP 以外のプロトコルに対するフロー。
- 検査が必要なフロー。FTP など場合によっては、コントロール チャネルはオフロードできませんがセカンダリ データ チャネルはオフロードできます。
- デバイスで終端する TLS VPN 接続。
- ルーテッド モードのマルチキャスト フロー。
- 3 つ以上のインターフェイスがあるブリッジ グループに対するトランスペアレント モードのマルチキャスト フロー。
- TCP インターセプト フロー。
- TCP ステートバイパスフロー。同じトラフィックにフローオフロードと TCP ステートバイパスを設定することはできません。
- AAA カットスループロキシフロー。
- Vpath、VXLAN 関連のフロー。
- セキュリティ グループでタグ付けされたフロー。
- クラスタで非対称フローが発生した場合に備えて、別のクラスタノードから転送されるリバースフロー。
- クラスタ内の一元化されたフロー（フローのオーナーが制御ユニットでない場合）。

その他の制限事項：

- フローオフロードとデッド接続検出（DCD）は互換性がありません。オフロードできる接続に DCD を設定しないでください。
- フローオフロード条件に一致する複数のフローがキューイングされて、ハードウェア上の同じ場所に同時にオフロードされる場合、最初のフローのみがオフロードされます。他のフローは通常どおりに処理されます。これをコリジョン（衝突）といいます。

オフロードを無効にする条件

フローがオフロードされた後、フロー内のパケットは次の条件を満たす場合に ASA Virtual に返され、さらに処理されます。

- タイムスタンプ以外の TCP オプションが含まれている。
- フラグメント化されている。
- これらは等コスト マルチパス（ECMP）ルーティングの対象であり、入力パケットは 1 つのインターフェイスから別のインターフェイスに移動する。

IPsec フローオフロードに関する制限事項

次の IPsec フローはオフロードできません。

- IKEv1 トンネル。IKEv2 トンネルのみがオフロードされます。IKEv2 は、より強力な暗号をサポートしています。
- ボリュームベースのキー再生成が設定されているフロー。
- 圧縮が設定されているフロー。
- トランスポートモードのフロー。トンネルモードのフローのみがオフロードされます。
- AH 形式。ESP/NAT-T 形式のみがサポートされます。
- ポストフラグメンテーションが設定されているフロー。
- アンチリプレイ ウィンドウ サイズが 64 ビットではなく、アンチリプレイが無効になっていないフロー。
- ファイアウォールフィルタが有効になっているフロー。
- マルチコンテキストモード。

翻訳について
このドキュメントは、米国シスコ発行ドキュメントの参考和訳です。リンク情報につきましては、日本語版掲載時点で、英語版にアップデートがあり、リンク先のページが移動/変更されている場合がありますことをご了承ください。あくまでも参考和訳となりますので、正式な内容については米国サイトのドキュメントを参照ください。

翻訳について
このドキュメントは、米国シスコ発行ドキュメントの参考和訳です。リンク情報につきましては、日本語版掲載時点で、英語版にアップデートがあり、リンク先のページが移動/変更されている場合がありますことをご了承ください。あくまでも参考和訳となりますので、正式な内容については米国サイトのドキュメントを参照ください。