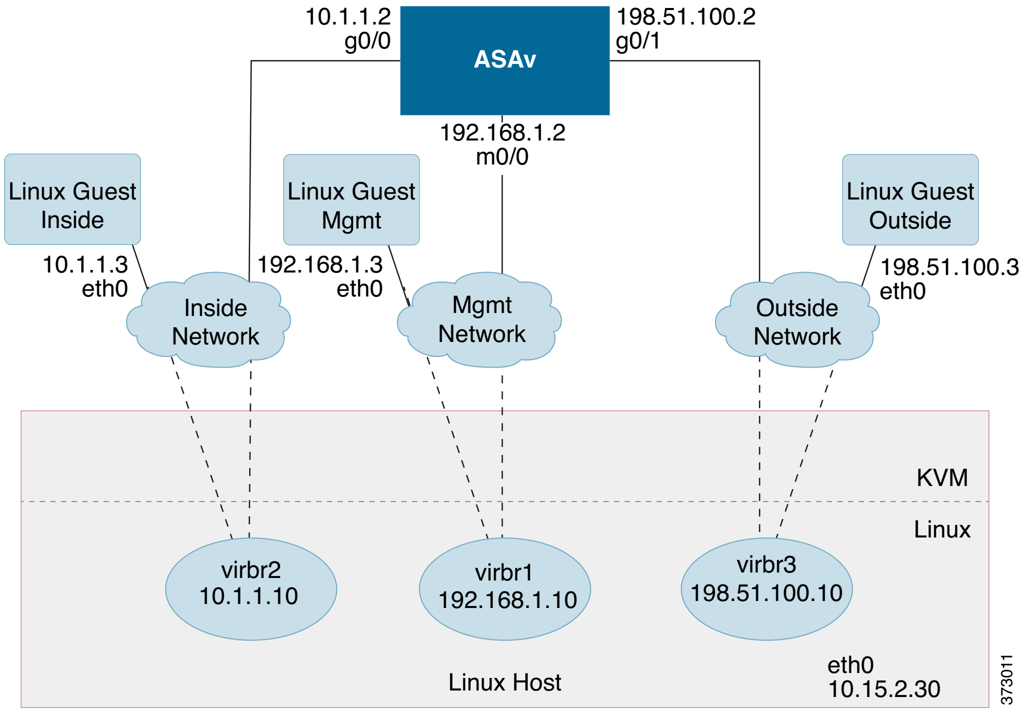
Deploy the ASAv Using KVM

You can deploy the ASAv using the Kernel-based Virtual Machine (KVM).

About ASAv Deployment Using KVM

Figure 1 on page -15 shows a sample network topology with ASAv and KVM. The procedures described in this chapter are based on the sample topology. You requirements will dictate the exact procedures you need. The ASAv acts as the firewall between the inside and outside networks. A separate management network is also configured.

Figure 1 Sample ASAv Deployment Using KVM



Prerequisites for the ASAv and KVM

- Download the ASAv qcow2 file from Cisco.com and put it on your Linux host:

  Note: A Cisco.com login and Cisco service contract are required.

- For the purpose of the sample deployment in this document, we are assuming you are using Ubuntu 14.04 LTS. Install the following packages on top of the Ubuntu 14.04 LTS host:
  - qemu-kvm
  - libvirt-bin
  - bridge-utils
  - virt-manager
  - virtinst
  - virsh tools
  - genisoimage
- Performance is affected by the host and its configuration. You can maximize the throughput of the ASAv on KVM by tuning your host. For generic host-tuning concepts, see Network Function Virtualization Packet Processing Performance of Virtualized Platforms with Linux and Intel Architecture.
- Useful optimizations for Ubuntu 14.04 include the following:
  - macvtap—High performance Linux bridge; you can use macvtap instead of a Linux bridge. Note that you must configure specific settings to use macvtap instead of the Linux bridge.
  - Transparent Huge Pages—Increases memory page size and is on by default in Ubuntu 14.04.
  - Hyperthread disabled—Reduces two vCPUs to one single core.
  - txqueuelength—Increases the default txqueuelength to 4000 packets and reduces drop rate.
  - pinning—Pins qemu and vhost processes to specific CPU cores; under certain conditions, pinning is a significant boost to performance.
- For information on optimizing a RHEL-based distribution, see Red Hat Enterprise Linux6 Virtualization Tuning and Optimization Guide.
- For KVM system requirements, see Cisco ASA Compatibility.

Prepare the Day 0 Configuration File

You can prepare a Day 0 configuration file before you launch the ASAv. This file is a text file that contains the ASAv configuration that will be applied when the ASAv is launched. This initial configuration is placed into a text file named “day0-config” in a working directory you chose, and is manipulated into a day0.iso file that is mounted and read on first boot. At the minimum, the Day 0 configuration file must contain commands that will activate the management interface and set up the SSH server for public key authentication, but it can also contain a complete ASA configuration. The day0.iso file (either your custom day0.iso or the default day0.iso) must be available during first boot.

Note: To automatically license the ASAv during initial deployment, place the Smart Licensing Identity (ID) Token that you downloaded from the Cisco Smart Software Manager in a text file named ‘idtoken’ in the same directory as the Day 0 configuration file.

Note: If you want to deploy the ASAv in transparent mode, you must use a known running ASA config file in transparent mode as the Day 0 configuration file. This does not apply to a Day 0 configuration file for a routed firewall.

Note: We are using Linux in this example, but there are similar utilities for Windows.

Procedure

1. Enter the CLI configuration for the ASAv in a text file called “day0-config”. Add interface configurations for the three interfaces and any other configuration you want.

   The fist line should begin with the ASA version. The day0-config should be a valid ASA configuration. The best way to generate the day0-config is to copy the desired parts of a running config from an existing ASA or ASAv. The order of the lines in the day0-config is important and should match the order seen in an existing show run command output.

   Example:

```
ASA Version 9.4.1
!
interface management0/0
   nameif management
   security-level 100
   ip address 192.168.1.2 255.255.255.0
   no shutdown
interface gigabitethernet0/0
   nameif inside
   security-level 100
   ip address 10.1.1.2 255.255.255.0
   no shutdown
interface gigabitethernet0/1
   nameif outside
   security-level 0
   ip address 198.51.100.2 255.255.255.0
   no shutdown
http server enable
http 192.168.1.0 255.255.255.0 management
crypto key generate rsa modulus 1024
username AdminUser password paSSw0rd
ssh 192.168.1.0 255.255.255.0 management
aaa authentication ssh console LOCAL
```

2. (Optional) Download the Smart License identity token file issued by the Cisco Smart Software Manager to your computer.

3. (Optional) Copy the ID token from the download file and put it a text file named ‘idtoken’ that only contains the ID token.

4. (Optional) For automated licensing during initial ASAv deployment, make sure the following information is in the day0-config file:

   – Management interface IP address
   – (Optional) HTTP proxy to use for Smart Licensing
   – A **route** command that enables connectivity to the HTTP proxy (if specified) or to tools.cisco.com
   – A DNS server that resolves tools.cisco.com to an IP address
   – Smart Licensing configuration specifying the ASAv license you are requesting
   – (Optional) A unique host name to make the ASAv easier to find in CSSM

5. Generate the virtual CD-ROM by converting the text file to an ISO file:

```
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$
```

The Identity Token automatically registers the ASAv with the Smart Licensing server.

6. Repeat Steps 1 through 5 to create separate default configuration files with the appropriate IP addresses for each ASAv you want to deploy.

Prepare the Virtual Bridge XML Files

You need to set up virtual networks that connect the ASAv guests to the KVM host and that connect the guests to each other.

Note: This procedure does not establish connectivity to the external world outside the KVM host.

Prepare the virtual bridge XML files on the KVM host. For the sample virtual network topology described in Prepare the Day 0 Configuration File, page 16, you need the following three virtual bridge files: virbr1.xml, virbr2.xml, and virbr3.xml (you must use these three filenames; for example, virbr0 is not allowed because it already exists). Each file has the information needed to set up the virtual bridges. You must give the virtual bridge a name and a unique MAC address. Providing an IP address is optional.

Procedure

1. Create three virtual networks bridge XML files:

   virbr1.xml:

```
<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>
```

   virbr2.xml:

```
<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>
```

   virbr3.xml:

```
<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>
```

2. Create a script that contains the following (in our example, we will name the script virt_network_setup.sh):

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

3. Run this script to setup the virtual network. The script brings the virtual networks up. The networks stay up as long as the KVM host is running.

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

Note: If you reload the Linux host, you must re-run the virt_network_setup.sh script. It does not persist over reboots.

4. Verify that the virtual networks were created:

```
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$
```

5. Display the IP address assigned to the virbr1 bridge. This is the IP address that you assigned in the XML file.

```
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
```

Launch the ASAv

Use a virt-install based deployment script to launch the ASAv.

Procedure

1. Create a virt-install script called "virt_install_asav.sh".

   The name of the ASAv VM must be unique across all other virtual machines (VMs) on this KVM host. The ASAv can support up to 10 networks. This example uses three networks. The order of the network bridge clauses is important. The first one listed is always the management interface of the ASAv (Management 0/0), the second one listed is GigabitEthernet 0/0 of the ASAv, and the third one listed is GigabitEthernet 0/1 of the ASAv, and so on up through GigabitEthernet0/8. The virtual NIC must be Virtio.

```
virt-install \
   --connect=qemu:///system \
   --network network=default,model=virtio \
   --network network=default,model=virtio \
   --network network=default,model=virtio \
   --name=asav \
   --cpu host \
   --arch=x86_64 \
   --machine=pc-1.0 \
   --vcpus=1 \
   --ram=2048 \
   --os-type=linux \
   --os-variant=generic26 \
   --noacpi \
   --virt-type=kvm \
   --import \
   --disk path=/home/kvmperf/Images/desmo.qcow2,format=qcow2,device=disk,bus=ide,cache=none \
   --disk path=/home/kvmperf/asav_day0.iso,format=iso,device=cdrom \
   --console pty,target_type=virtio \
   --serial tcp,host=127.0.0.1:4554,mode=bind,protocol=telnet
```

2. Run the virt_install script:

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

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

A window appears displaying the console of the VM. You can see that the VM is booting. It takes a few minutes for the VM to boot. Once the VM stops booting you can issue CLI commands from the console screen.