



# Oracle Linux 7: Advanced Administration

Activity Guide – Volume II  
D90758GC10  
Edition 1.0 | September 2015 | D92968

Learn more from Oracle University at [oracle.com/education/](http://oracle.com/education/)

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

#### **Disclaimer**

This document contains proprietary information and is protected by copyright and other intellectual property laws. You may copy and print this document solely for your own use in an Oracle training course. The document may not be modified or altered in any way. Except where your use constitutes "fair use" under copyright law, you may not use, share, download, upload, copy, print, display, perform, reproduce, publish, license, post, transmit, or distribute this document in whole or in part without the express authorization of Oracle.

The information contained in this document is subject to change without notice. If you find any problems in the document, please report them in writing to: Oracle University, 500 Oracle Parkway, Redwood Shores, California 94065 USA. This document is not warranted to be error-free.

#### **Restricted Rights Notice**

If this documentation is delivered to the United States Government or anyone using the documentation on behalf of the United States Government, the following notice is applicable:

#### **U.S. GOVERNMENT RIGHTS**

The U.S. Government's rights to use, modify, reproduce, release, perform, display, or disclose these training materials are restricted by the terms of the applicable Oracle license agreement and/or the applicable U.S. Government contract.

#### **Trademark Notice**

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

#### **Author**

Craig McBride

#### **Technical Contributors and Reviewers**

Avi Miller, Elena Zannoni, Wim Coekaerts, Harald Van Breederode, Joel Goodman, Manish Kapur, Yasar Akthar, Antoinette O'Sullivan, Gavin Bowe, Steve Miller, Herbert Van Den Bergh, Todd Vierling and John Haxby

This book was published using: **Oracle Tutor**

## Table of Contents

---

<b>Practices for Lesson 1: Course Introduction .....</b>	<b>1-1</b>
Course Practice Environment: Security Credentials.....	1-2
Practices for Lesson 1: Overview.....	1-3
Practice 1-1: Exploring the dom0 Environment.....	1-4
Practice 1-2: Starting, Stopping, and Listing VM Guests .....	1-11
Practice 1-3: Exploring the host01 VM .....	1-13
Practice 1-4: Exploring the host02 VM .....	1-17
Practice 1-5: Exploring the host03 VM .....	1-20
Practice 1-6: Logging Off from Your Student PC .....	1-22
<b>Practices for Lesson 2: Network Addressing and Name Services .....</b>	<b>2-1</b>
Practices for Lesson 2: Overview.....	2-2
Practice 2-1: Configuring a DHCP Server.....	2-3
Practice 2-2: Configuring a DHCP Client.....	2-6
Practice 2-3: Viewing and Testing the DNS Configuration.....	2-9
Practice 2-4: Configuring a Caching-Only Nameserver .....	2-16
<b>Practices for Lesson 3: Authentication and Directory Services .....</b>	<b>3-1</b>
Practices for Lesson 3: Overview.....	3-2
Practice 3-1: Configuring an OpenLDAP Server .....	3-3
Practice 3-2: Implementing OpenLDAP Authentication.....	3-21
Practice 3-3: Authenticating from an OpenLDAP Client .....	3-26
<b>Practices for Lesson 4: Pluggable Authentication Modules (PAM) .....</b>	<b>4-1</b>
Practices for Lesson 4: Overview.....	4-2
Practice 4-1: Configuring PAM for a Single Login Session .....	4-3
Practice 4-2: Configuring PAM to Prevent Non-root Login.....	4-8
<b>Practices for Lesson 5: Web and Email Services.....</b>	<b>5-1</b>
Practices for Lesson 5: Overview.....	5-2
Practice 5-1: Configuring the Apache Web Server.....	5-3
<b>Practices for Lesson 6: Installing Oracle Linux 7 by Using Kickstart.....</b>	<b>6-1</b>
Practices for Lesson 6: Overview.....	6-2
Practice 6-1: Performing a Kickstart Installation.....	6-3
Practice 6-2: Using Rescue Mode.....	6-14
<b>Practices for Lesson 7: Samba Services.....</b>	<b>7-1</b>
Practices for Lesson 7: Overview.....	7-2
Practice 7-1: Configuring a Samba Server .....	7-3
Practice 7-2: Accessing Samba Shares from a Client Host .....	7-8
Practice 7-3: Accessing a Linux Samba Share from a Windows System .....	7-12
<b>Practices for Lesson 8: Advanced Software Package Management.....</b>	<b>8-1</b>
Practices for Lesson 8: Overview.....	8-2
Practice 8-1: Exploring the host04 VM .....	8-3
Practice 8-2: Managing Yum Plug-Ins .....	8-9
Practice 8-3: Using Yum Utilities .....	8-16
Practice 8-4: Creating an RPM Package .....	8-22
Practice 8-5: Managing Software Updates with PackageKit .....	8-31
Practice 8-6: Working with Yum History and Yum Cache .....	8-39
<b>Practices for Lesson 9: Advanced Storage Administration.....</b>	<b>9-1</b>

Practices for Lesson 9: Overview.....	9-2
Practice 9-1: Creating and Mounting a File System .....	9-3
Practice 9-2: Implementing Access Control Lists .....	9-6
Practice 9-3: Setting Disk Quotas .....	9-9
Practice 9-4: Encrypting a File System.....	9-13
Practice 9-5: Using kpartx .....	9-16
Practice 9-6: Exploring and Configuring Udev .....	9-20
<b>Practices for Lesson 10: Advanced Networking .....</b>	<b>10-1</b>
Practices for Lesson 10: Overview.....	10-2
Practice 10-1: Configuring Network Bonding by Using the GUI .....	10-3
Practice 10-2: Configuring Network Bonding from the Command Line.....	10-21
Practice 10-3: Working with Bonded Interfaces .....	10-26
Practice 10-4: Configuring 802.1Q VLAN Tagging by Using the GUI.....	10-37
Practice 10-5: Configuring 802.1Q VLAN Tagging from the Command Line .....	10-46
Practice 10-6: Working with VLAN Interfaces .....	10-49
Practice 10-7: Configuring a Site-to-Site VPN .....	10-58
<b>Practices for Lesson 11: OCFS2 and Oracle Clusterware.....</b>	<b>11-1</b>
Practices for Lesson 11: Overview.....	11-2
Practice 11-1: Preparing for an OCFS2 Configuration.....	11-3
Practice 11-2: Verifying that the Required Software Is Installed .....	11-9
Practice 11-3: Configuring the Cluster Layout .....	11-10
Practice 11-4: Configuring and Starting the O2CB Cluster Stack Service .....	11-14
Practice 11-5: Creating an OCFS2 Volume.....	11-17
Practice 11-6: Mounting an OCFS2 Volume.....	11-21
Practice 11-7: Tuning and Debugging OCFS2.....	11-26
<b>Practices for Lesson 12: iSCSI and Multipathing.....</b>	<b>12-1</b>
Practices for Lesson 12: Overview.....	12-2
Practice 12-1: Configuring an iSCSI Server (Target).....	12-3
Practice 12-2: Configuring an iSCSI Client (Initiator).....	12-14
Practice 12-3: Configuring iSCSI Multipathing .....	12-21
<b>Practices for Lesson 13: Control Groups (Cgroups).....</b>	<b>13-1</b>
Practices for Lesson 13: Overview.....	13-2
Practice 13-1: Exploring cgroup Integration Into systemd.....	13-3
Practice 13-2: Exploring cgroup Hierarchies and cgroup Subsystem Parameters .....	13-10
Practice 13-3: Controlling Access to System Resources .....	13-15
<b>Practices for Lesson 14: Virtualization with Linux.....</b>	<b>14-1</b>
Practices for Lesson 14: Overview.....	14-2
Practice 14-1: Preparing the Virtualization Host for KVM .....	14-3
Practice 14-2: Starting the Virtual Machine Manager and Preparing to Create a Virtual Machine.....	14-9
Practice 14-3: Creating a Virtual Machine .....	14-22
Practice 14-4: Managing Your Virtual Machine .....	14-36
<b>Practices for Lesson 15: Linux Containers (LXC) .....</b>	<b>15-1</b>
Practices for Lesson 15: Overview.....	15-2
Practice 15-1: Completing Linux Container Prerequisites.....	15-3
Practice 15-2: Creating an Oracle Linux Container .....	15-7
Practice 15-3: Using lxc Commands .....	15-11
<b>Practices for Lesson 16: Docker .....</b>	<b>16-1</b>
Practices for Lesson 16: Overview.....	16-2

Practice 16-1: Using sftp to Upload Docker Package and Images .....	16-3
Practice 16-2: Installing and Configuring Docker .....	16-5
Practice 16-3: Using Docker Commands.....	16-9
<b>Practice for Lesson 17: Security Enhanced Linux (SELinux) .....</b>	<b>17-1</b>
Practice for Lesson 17: Overview .....	17-2
Practice 17-1: Exploring SELinux.....	17-3
Practice 17-2: Configuring an SELinux Boolean .....	17-11
Practice 17-3: Configuring SELinux Context.....	17-15
<b>Practices for Lesson 18: Core Dump Analysis.....</b>	<b>18-1</b>
Practices for Lesson 18: Overview.....	18-2
Practice 18-1: Configuring Kdump .....	18-3
Practice 18-2: Creating a Core Dump File.....	18-12
Practice 18-3: Preparing Your System to Analyze the vmcore.....	18-14
Practice 18-4: Using the crash Utility.....	18-16
<b>Practices for Lesson 19: Dynamic Tracing with DTrace .....</b>	<b>19-1</b>
Practices for Lesson 19: Overview.....	19-2
Practice 19-1: Using sftp to Upload DTrace Packages.....	19-3
Practice 19-2: Installing the DTrace Packages .....	19-8
Practice 19-3: Using DTrace from the Command Line .....	19-12
Practice 19-4: Creating and Running D Scripts.....	19-20
<b>Appendix - NIS Configuration.....</b>	<b>20-1</b>
Appendix - Overview .....	20-2
Practice A-1: Configuring an NIS Server .....	20-3
Practice A-2: Configuring an NIS Client.....	20-9
Practice A-3: Implementing NIS Authentication .....	20-11
Practice A-4: Testing NIS Authentication.....	20-15
Practice A-5: Auto-Mounting a User Home Directory .....	20-17
Practice A-6: Restoring the Systems to Their Original State.....	20-20
<b>Appendices: Remote Access Options.....</b>	<b>21-1</b>
Appendices: Overview.....	21-2
Appendix A: Using an NX Client to Connect to dom0.....	21-3
Appendix B: Using an NX Player to Connect to dom0.....	21-7
Appendix C: Using VNC (TightVNC) to Connect Directly to VM Guests.....	21-13
Appendix D: Using NoMachine Version 4 to Connect to dom0 .....	21-16



## **Practices for Lesson 11: OCFS2 and Oracle Clusterware**

**Chapter 11**

## Practices for Lesson 11: Overview

---

### Practices Overview

In these practices, you perform the following:

- Prepare for an OCFS2 configuration.
- Install or upgrade the software required for OCFS2.
- Configure the cluster layout.
- Configure and start the O2CB cluster stack service.
- Create an OCFS2 volume.
- Mount an OCFS2 volume.
- Perform OCFS2 tuning and debugging.

## Practice 11-1: Preparing for an OCFS2 Configuration

---

### Overview

In this practice, you perform the following:

- Reconfigure VMs to use shared storage.
- View a private network for VM guests.

### Assumptions

- You are the `root` user on **dom0**.

### Tasks

If you completed Practice 2-1 and Practice 2-2 that used DHCP to assign an IP address to `eth1` on **host01**, go directly to step 2.

Only perform step 1 if `eth1` on **host01** does not have an IP address.

1. If necessary, assign a static IP address to `eth1` on **host01**.

- From **dom0**, use the `ssh` command to log in to **host01**.
  - The `root` password is `oracle`.

```
[dom0]# ssh host01
root@host01's password: oracle
Last login: ...
[root@host01 ~]#
```

- From **host01**, use the `cd` command to change to the `/etc/sysconfig/network-scripts` directory.

```
# cd /etc/sysconfig/network-scripts
```

- Use the `vi` editor to edit the `ifcfg-eth1` file as follows:

- Only the bold lines should need to be edited, but verify that all lines are as shown.

```
# vi ifcfg-eth1
DEVICE=eth1
HWADDR=00:16:3e:00:02:01
TYPE=Ethernet
UUID=... (leave this as is)
ONBOOT=yes
NM_CONTROLLED=yes
BOOTPROTO=none
IPV6INIT=no
IPADDR=192.168.1.200
NETMASK=255.255.255.0
BROADCAST=192.168.1.255
```

- Use the `ifup eth1` command to start the `eth1` network interface.

```
# ifup eth1
Determining if ip address 192.168.1.200 is already in use for
device eth1...
```

- e. Use the `ifconfig eth1` command to verify that `eth1` has an IP address.

```
# ifconfig eth1
eth1      Link encap:Ethernet HWaddr 00:16:3E:00:02:01
          inet addr:192.168.1.200 ...
          inet6 addr: ...
          UP BROADCAST RUNNING ...
...
...
```

- f. Use the `exit` command to log off **host01**.

```
# exit
logout
Connection to host01 closed.
```

2. If necessary, shut down all VMs.

- If you just completed Practice 10, all the VMs are already shut down and you can proceed to step 3.
  - If the VMs are not shut down, perform steps 2a and 2b.
- a. From **dom0**, run the `xm destroy <VM name>` command on all VMs that are currently running.
- The following example runs this command on all three VMs.

```
# xm destroy host01
# xm destroy host02
# xm destroy host03
```

- b. Use the `xm list` command to confirm that all VM domains are shut down.

- When all VMs are shut down, only **dom0** appears in the output of the `xm list` command.

# xm list					
Name	ID	Mem	VCPUs	State	Time (s)
Domain-0	0	2048	2	r-----	281.1

3. Change the “`xvdb`” entries in the VM configuration files to shared storage for all three VMs.

- a. From **dom0**, use the `grep` command to list the `xvdb` entries in the VM guest `vm.cfg` files.

- The quotes in the following example are normal single quotes, not back quotes.

```
[dom0]# grep xvdb /OVS/running_pool/host0[123]/vm.cfg
host01/vm.cfg: 'file:/OVS/running_pool/host01/u01.img,xvdb,w',
host02/vm.cfg: 'file:/OVS/sharedDisk/physDisk1.img,xvdb,w!',
host03/vm.cfg: 'file:/OVS/running_pool/host03/u01.img,xvdb,w',
```

- Note that **host01** and **host03** use `u01.img` in the `/OVS/running_pool/host0[123]/vm.cfg` file for the `xvdb` virtual disk.
- Note that **host02** uses the `/OVS/sharedDisk/physDisk1.img` file for the `xvdb` virtual disk.
- Also note that the **host02** entry uses `w!` to indicate that this is shared storage.

- b. Use the `vi` editor to change the “xvdb” entry in the **host01** `vm.cfg` file to match the “xvdb” entry in the **host02** `vm.cfg` file.

- Make a backup copy of the file beforehand.

```
# cd /OVS/running_pool/host01
# cp vm.cfg vm.cfg.BAK
# vi vm.cfg
...
`file:/OVS/running_pool/host01/u01.img,xvdb,w' ,      (old entry)
`file:/OVS/sharedDisk/physDisk1.img,xvdb,w!' ,        (new entry)
...
```

- c. Use the `vi` editor to change the “xvdb” entry in the **host03** `vm.cfg` file to match the “xvdb” entry in the **host02** `vm.cfg` file.

- Make a backup copy of the file beforehand.

```
# cd /OVS/running_pool/host03
# cp vm.cfg vm.cfg.BAK
# vi vm.cfg
...
`file:/OVS/running_pool/host03/u01.img,xvdb,w' ,      (old entry)
`file:/OVS/sharedDisk/physDisk1.img,xvdb,w!' ,        (new entry)
...
```

- d. Use the `grep` command to list the `xvdb` entries in the VM guest `vm.cfg` files.

```
[dom0]# grep xvdb /OVS/running_pool/host0[123]/vm.cfg
host01/vm.cfg: `file:/OVS/sharedDisk/physDisk1.img,xvdb,w!' ,
host02/vm.cfg: `file:/OVS/sharedDisk/physDisk1.img,xvdb,w!' ,
host03/vm.cfg: `file:/OVS/sharedDisk/physDisk1.img,xvdb,w!' ,
```

- Do not continue until the **host01** and **host03** `xvdb` entries are identical to the **host02** `xvdb` entry.

#### 4. Start all VMs: **host01**, **host02**, and **host03**.

- Start **host03** first because it is configured as a DHCP server.
  - Start **host02** second.
  - Start **host01** last because it obtains an IP address from **host03** for the `eth1` interface.
- a. From the `/OVS/running_pool/host03` directory on **dom0**, run the `xm create vm.cfg` command to start the **host03** VM.

```
# cd /OVS/running_pool/host03
# xm create vm.cfg
Using config file "./vm.cfg".
Started domain host03 (id=#)
```

- b. From the `/OVS/running_pool/host02` directory on **dom0**, run the `xm create vm.cfg` command to start the **host02** VM.

```
# cd /OVS/running_pool/host02
# xm create vm.cfg
```

```
Using config file "./vm.cfg".
Started domain host02 (id=#)
```

- c. From the /OVS/running\_pool/host01 directory on **dom0**, run the `xm create vm.cfg` command to start the **host01** VM.

```
# cd /OVS/running_pool/host01
# xm create vm.cfg
Using config file "./vm.cfg".
Started domain host01 (id=#)
```

- d. Use the `xm list` command to confirm whether all VM domains are running.

Name	ID	Mem	VCPUs	State	Time(s)
Domain-0	0	2048	2	r-----	304.5
host01	22	1536	1	-b-----	18.7
host02	21	1536	1	-b-----	159.0
host03	20	1536	1	-b-----	13.2

- In this example, all VMs are running. The ID and Time(s) values are examples.
- Do not be concerned if the state of the VMs is ‘r’ or ‘b’. Continue to step 5.

5. Log in and view a private network configuration on the VM guests.

- a. From **dom0**, use the `ssh` command to log in to **host01**.

- The root password is **oracle**.

```
[dom0]# ssh host01
root@host01's password: oracle
Last login...
[root@host01 ~]#
```

- b. From **host01**, use the `ip addr` command to display the network configuration.

```
# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue ...
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet addr:127.0.0.1/8 scope host lo
    ...
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc ...
    link/ether 00:16:3e:00:01:01 brd ff:ff:ff:ff:ff:ff
    inet 192.0.2.101/24 brd 192.0.2.255 scope global eth0
    ...
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc ...
    link/ether 00:16:3e:00:02:01 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.200/24 brd 192.168.1.255 scope global eth1
    ...
```

- Note that **eth1** is on the private network (192.168.1).
- The **eth1** interface was configured to use DHCP in “Practices for Lesson 2: Network Addressing and Name Services.”

- In this example, the IP address for eth1 is 192.168.1.200, but because DHCP is used, the IP address for your eth1 interface might be different.
  - This practice assumes the IP address for eth1 on **host01** is 192.168.1.200. Substitute the correct IP address, if different, as necessary in this practice.
- c. From **dom0**, open a new terminal window, **su -** to become the **root** user (password is **oracle**) on **dom0**, and use the **ssh** command to log in to **host02**.
- The root password on **host02** is **oracle**.

```
[dom0] $ su -
Password: oracle
[dom0] # ssh host02
root@host02's password: oracle
Last login...
[root@host02 ~] #
```

- d. From **host02**, use the **ip addr** command to display the network configuration.

```
# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue ...
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet addr:127.0.0.1/8 scope host lo
    ...
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc ...
    link/ether 00:16:3e:00:01:02 brd ff:ff:ff:ff:ff:ff
    inet 192.0.2.102/24 brd 192.0.2.255 scope global eth0
    ...
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc ...
    link/ether 00:16:3e:00:02:02 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.102/24 brd 192.168.1.255 scope global eth1
    ...
```

- Note that eth1 is on the private network (IP address is 192.168.1.102).
- e. From **dom0**, open a third terminal window, **su -** to become the **root** user (password is **oracle**) on **dom0**, and use the **ssh** command to log in to **host03**.
- The root password on **host03** is **oracle**.

```
[dom0] $ su -
Password: oracle
[dom0] # ssh host03
root@host03's password: oracle
Last login...
[root@host03 ~] #
```

- f. From **host03**, use the **ip addr** command to display the network configuration.

```
# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue ...
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
```

```
inet addr:127.0.0.1/8 scope host lo
...
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc ...
    link/ether 00:16:3e:00:01:03 brd ff:ff:ff:ff:ff:ff
    inet 192.0.2.103/24 brd 192.0.2.255 scope global eth0
...
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc ...
    link/ether 00:16:3e:00:02:03 brd ff:ff:ff:ff:ff:ff
    inet 192.0.2.104/24 brd 192.0.2.255 scope global eth1
...
4: eth2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc ...
    link/ether 00:16:3e:00:03:03 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.103/24 brd 192.168.1.255 scope global eth2
...
```

- Note that eth2 is on the private network (IP address is 192.168.1.103).

You also need to disable or modify `firewalld` to allow network traffic on the private network interface. You perform this step in “Practice 11-6: Mounting an OCFS2 Volume.”

## Practice 11-2: Verifying that the Required Software Is Installed

---

### Overview

In this practice, you perform the following:

- Ensure that the OCFS2 package is installed.
- Ensure that the version of the UEK is the same on all VMs.
- Enable kernel settings for O2CB.
- Perform each step on all VMs: **host01**, **host02**, and **host03**.

### Assumptions

You are the `root` user on the **host01**, **host02**, and **host03** VM guests.

### Tasks

1. Install the `ocfs2-tools` package on the VM guests.

Use the `yum install` command to install the `ocfs2-tools` package.

- Answer `y` to “Is this ok”.

```
# yum install ocfs2-tools
...
Is this ok [y/d/N] : y
...
Complete!
```

2. Ensure that the same version of the UEK kernel is installed on the VM guests.

Use the `uname -r` command to determine the UEK kernel version.

- The UEK version is `3.8.13-55.1.6.el7uek` on all VMs.

```
# uname -r
3.8.13-55.1.6.el7uek.x86_64
```

3. Enable kernel settings for O2CB on the VM guests.

- a. Use the `vi` editor to add the following two entries to the end of the `/etc/sysctl.conf` file.

```
# vi /etc/sysctl.conf
...
kernel.panic_on_oops = 1
kernel.panic = 30
```

- b. Use the `sysctl -p` command to cause the changes made to `/etc/sysctl.conf` to take effect immediately.

```
# sysctl -p
...
kernel.panic_on_oops = 1
kernel.panic = 30
```

4. Repeat steps 1, 2, and 3 for the remaining VM guests.

## Practice 11-3: Configuring the Cluster Layout

---

### Overview

In this practice, you perform the following:

- Execute all steps from **host01**.
- Create the cluster layout configuration file on **host01**.
- Copy the cluster layout configuration file from **host01** to **host02** and to **host03**.

### Assumptions

You are the `root` user on the **host01** VM guest.

### Tasks

1. From **host01**, configure the cluster layout as follows:

- Cluster name = `mycluster`
- Number of nodes in the cluster = 3
- Heartbeat mode = local
- First node name = `host01`, IP address = 192.168.1.200
- Second node name = `host02`, IP address = 192.168.1.102
- Third node name = `host03`, IP address = 192.168.1.103

- a. Use the `o2cb` command to add the `mycluster` cluster.

```
# o2cb add-cluster mycluster
```

- b. Use the `o2cb` command to list the `mycluster` information in the cluster layout configuration file, `/etc/ocfs2/cluster.conf`.

- You could also use the `cat` command or any shell command (`less`, `more`) that displays the content of a text file.

```
# o2cb list-cluster mycluster
cluster:
    node_count = 0
    heartbeat_mode = local
    name = mycluster
```

- Note that the default heartbeat mode is local.

- c. Use the `o2cb` command to add the `host01` node to the `mycluster` cluster.

```
# o2cb add-node mycluster host01
```

- d. Use the `o2cb` command to list `mycluster` information.

- The order of your output might be different from the following example. The `cluster` section might be listed before the `node` section. This is not a problem.

```
# o2cb list-cluster mycluster
node:
    number = 0
    name = host01
    ip_address = 192.0.2.101
    ip_port = 7777
    cluster = mycluster

cluster:
    node_count = 1
    heartbeat_mode = local
    name = mycluster
```

- Note that by default, the IP address for the node is obtained from the `/etc/hosts` file.
- This entry needs to be removed and re-added; it is not the address for the private network interface.

- e. Use the `o2cb` command to remove the `host01` node from the `mycluster` cluster.

```
# o2cb remove-node mycluster host01
```

- f. Use the `o2cb` command to add the `host01` node to the `mycluster` cluster.

- The following sample command uses `192.168.1.200` as the IP address.
- On your system, the IP address of `eth1` on **host01** might not be `192.168.1.200`.
- Replace `192.168.1.200`, if necessary with the IP address of `eth1` on **host01**.

```
# o2cb add-node mycluster host01 --ip 192.168.1.200
```

- g. Use the `o2cb` command to add the `host02` node with IP address `192.168.1.102` to the `mycluster` cluster.

```
# o2cb add-node mycluster host02 --ip 192.168.1.102
```

- h. Use the `o2cb` command to add the `host03` node with IP address `192.168.1.103` to the `mycluster` cluster.

```
# o2cb add-node mycluster host03 --ip 192.168.1.103
```

- i. Use the `o2cb` command to list `mycluster` information.

- The order of your output might be different.

```
# o2cb list-cluster mycluster
node:
    number = 0
    name = host01
    ip_address = 192.168.1.200
    ip_port = 7777
    cluster = mycluster
```

```

node:
    number = 1
    name = host02
    ip_address = 192.168.1.102
    ip_port = 7777
    cluster = mycluster

node:
    number = 2
    name = host03
    ip_address = 192.168.1.103
    ip_port = 7777
    cluster = mycluster

cluster:
    node_count = 3
    heartbeat_mode = local
    name = mycluster

```

- j. Use the `o2cb` command to list `mycluster` information with the `--oneline`, argument.

```
# o2cb list-cluster mycluster --oneline
node: 0 host01 192.168.1.200:7777 mycluster
node: 1 host02 192.168.1.102:7777 mycluster
node: 2 host03 192.168.1.103:7777 mycluster
cluster: 3 local mycluster
```

- The cluster layout configuration is complete.
2. Copy the cluster layout configuration file to all nodes in the cluster.
- Use the `scp` command to copy `/etc/ocfs2/cluster.conf` from **host01** to **host02**. The root user password is `oracle`.

```
# scp /etc/ocfs2/cluster.conf host02:/etc/ocfs2/cluster.conf
The authenticity of host 'host02 (192.0.2.102)' can't be
established. RSA key fingerprint is ...
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'host02,192.0.2.102' (RSA) to the
list of known hosts.
root@host02's password: oracle
scp: /etc/ocfs2/cluster.conf: no such file or directory
```

- Note that the copy failed because the `/etc/ocfs2` directory does not exist on the remote host, **host02**.
- If you are logged on to **host02** in another terminal window, use the `mkdir` command to make the `/etc/ocfs2` directory on **host02**. Then repeat the preceding `scp` command to copy the file.

- If you are not logged on to **host02**, perform the following step (2b).
- b. Use the `sftp` command to establish a connection from **host01** to **host02**.
- After connecting to **host02**, make the `/etc/ocfs2` directory on **host02**.
  - After making the directory, use the `put` command to upload the file.
  - After transferring the file, use the `exit` command to close the `sftp` connection.

```
# sftp host02
Connecting to host02...
root@host02's password: oracle
sftp> mkdir /etc/ocfs2
sftp> put /etc/ocfs2/cluster.conf /etc/ocfs2/cluster.conf
Uploading /etc/ocfs2/cluster.conf to /etc/ocfs2/cluster.conf
/etc/ocfs2/cluster.conf          100% ...
sftp> exit
```

3. Repeat step 2 to copy the cluster layout configuration file from **host01** to **host03**.

- Substitute **host03** for **host02** in the commands.
- The cluster layout configuration file now exists on all nodes.

## Practice 11-4: Configuring and Starting the O2CB Cluster Stack Service

---

### Overview

In this practice, you perform the specified step on each VM: **host01**, **host02**, and **host03**.

### Assumptions

You are the `root` user on all VM guests.

### Tasks

1. Configure the cluster stack service.
  - a. Use the `service o2cb` command without any arguments to display usage.
    - Note that you must invoke `/etc/init.d/o2cb` directly, or use the `service` command, to configure the cluster stack. The `systemctl` utility does not support the “configure” command.

```
# service o2cb
Usage: /etc/init.d/o2cb {force-reload|configure|load|unload|
online|offline|force-offline|status|online-status}
```

1. Configure the cluster stack service.
  - b. Use the `service` command to run the `/etc/init.d/o2cb` initialization script to configure the cluster stack.
    - Enter `mycluster` as the cluster to start on boot.
    - Accept the defaults (press `Enter`) for all other queries.

```
# service o2cb configure
Configuring the O2CB driver.
```

This will configure the on-boot properties of the O2CB driver. The following questions will determine whether the driver is loaded on boot. The current values will be shown in brackets ('[]'). Hitting `<ENTER>` without typing an answer will keep that current value. `Ctrl-C` will abort.

```
Cluster stack backing O2CB [o2cb]: ENTER
Cluster to start on boot (Enter "none" to clear) [ocfs2]:
mycluster
Specify heartbeat dead threshold (>=7) [31]: ENTER
Specify network idle timeout in ms (>=5000) [30000]: ENTER
Specify network keepalive delay in ms (>=1000) [2000]: ENTER
Specify network reconnect delay in ms (>=2000) [2000]: ENTER
Writing O2CB configuration: OK
Stat: cannot read file system information for '/dlm': No such
file or directory
```

1. Configure the cluster stack service.
  - c. Use the `service` command to run the `/etc/init.d/o2cb` initialization script to load the modules.

```
# service o2cb load
Loading filesystem "configfs": OK
Loading stack plugin "o2cb": OK
Loading filesystem "ocfs2_dlmfs": OK
Creating directory '/dlm': OK
Mounting ocfs2_dlmfs filesystem at /dlm: OK
```

- d. Repeat step 1b to configure the cluster stack.
- Accept the defaults (press Enter) for all other queries.

```
# service o2cb configure
Configuring the O2CB driver.
```

This will configure the on-boot properties of the O2CB driver. The following questions will determine whether the driver is loaded on boot. The current values will be shown in brackets ('[]'). Hitting <ENTER> without typing an answer will keep that current value. Ctrl-C will abort.

```
Cluster stack backing O2CB [o2cb]: ENTER
Cluster to start on boot (Enter "none" to clear) [mycluster]: ENTER
Specify heartbeat dead threshold (>=7) [31]: ENTER
Specify network idle timeout in ms (>=5000) [30000]: ENTER
Specify network keepalive delay in ms (>=1000) [2000]: ENTER
Specify network reconnect delay in ms (>=2000) [2000]: ENTER
Writing O2CB configuration: OK
Clean userdlm domains: OK
Unmounting ocfs2_dlmfs filesystem: OK
Unloading module "ocfs2_dlmfs": OK
Unloading module "ocfs2_stack_o2cb": OK
Unloading module "configfs": OK
```

- e. Use the `service` command to run the `/etc/init.d/o2cb` initialization script to start the cluster.

```
# service o2cb online mycluster
Loading filesystem "configfs": OK
Loading stack plugin "o2cb": OK
Loading filesystem "ocfs2_dlmfs": OK
Mounting ocfs2_dlmfs filesystem at /dlm: OK
Setting cluster stack "o2cb": OK
Registering O2CB cluster "mycluster": OK
Setting O2CB cluster timeouts : OK
```

- f. Use the `service` command to run the `/etc/init.d/o2cb` initialization script to view the status and settings of the cluster stack.

```
# service o2cb status
Driver for "configfs": Loaded
Filesystem "configfs": Mounted
Stack glue driver: Loaded
Stack plugin "o2cb": Loaded
Driver for "ocfs2_dlmfs": Loaded
Filesystem "ocfs2_dlmfs": Mounted
Checking O2CB cluster mycluster: Online
Heartbeat dead threshold = 31
Network idle timeout: 30000
Network keepalive delay: 2000
Network reconnect delay: 2000
Heartbeat mode: Local
Checking O2CB heartbeat: Not active
Please run 'systemctl enable o2cb.service' to enable o2cb
```

- Note that the cluster status is Online.
- Note that the O2CB heartbeat is not active. The heartbeat becomes active after the volume is mounted.
- Note the last message, “Please run ‘`systemctl enable o2cb.service`’ to enable `o2cb`”.

- g. Use the `systemctl` command to enable the `o2cb` service.

```
# systemctl enable o2cb
ln -s '/usr/lib/systemd/system/o2cb.service'
'/etc/systemd/system/multi-user.target.wants/o2cb.service'
```

2. Repeat step 1 for the remaining VM guests.

- The O2CB cluster stack needs the same configuration on each VM guest.

## Practice 11-5: Creating an OCFS2 Volume

---

### Overview

In this practice, you perform the following:

- Perform all steps from **host01**.
- Create a single partition on the shared disk.
- Create OCFS2 volumes with default settings.
- Create OCFS2 volumes with settings for `mail`, `datafiles`, and `vmstore` usage.

### Assumptions

You are the `root` user on the **host01** VM.

### Tasks

1. From **host01**, create a partition on the shared disk.

- Though it is not required, it is recommended that you create OCFS2 volumes only on partitions because only partitioned volumes can be mounted by label.

Use the `fdisk` command to create a single partition on `/dev/xvdb`. Use the entire disk for the partition.

```
# fdisk /dev/xvdb
...
Command (m for help): n
Partition type:
      p   primary (0 primary, 0 extended, 4 free)
      e   extended
Select (default p): ENTER
Using default response p
Partition number (1-4, default 1): ENTER
First sector (2048-20971519, default 2048): ENTER
Using default value 2048
Last sector, +sectors or +size{K,M,G} (2048-20971519, default 20971519): ENTER
Using default value 20971519
Partition 1 of type Linux and of size 10 GiB is set

Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks
```

2. Create different types of OCFS2 volumes on the partitioned disk, /dev/xvdb1.
- Use the `mkfs.ocfs2` command to create a file system without any options (all defaults).
- ```
# mkfs.ocfs2 /dev/xvdb1
mkfs.ocfs2 1.8.0
Cluster stack: classic o2cb
Label:
Features: sparse extended-slotmap backup-super unwritten
inline-data strict-journal-super xattr indexed-dirs refcount
discontig-bg
Block size: 4096 (12 bits)
Cluster size: 4096 (12 bits)
Volume size: 10736369664 (2621184 clusters) (2621184 blocks)
Cluster groups: 82 (tail covers 8448 clusters, rest cover 32256
clusters)
Extent allocator size: 4194304 (1 groups)
Journal size: 67108864
Node slots: 4
Creating bitmaps: done
Initializing superblock: done
Writing system files: done
Writing superblock: done
Writing backup superblock: 2 block(s)
Formatting Journals: done
Growing extent allocator: done
Formatting slot map: done
Formatting quota files: done
Writing lost_found: done
mkfs.ocfs2 successful
```
- Note the default features: sparse extended-slotmap backup-super unwritten inline-data strict-journal-super xattr indexed-dirs refcount discontig-bg
  - Note the default block size and cluster size: 4 KB
  - Note the default number of node slots: 4
  - Note the default journal size: 67108864
- Use the `mkfs.ocfs2` command to create a file system with the `-T mail` option.
    - Specify this type when you intend to use the file system as a mail server.
    - Mail servers perform many metadata changes to many small files, which require the use of a large journal.

- Answer **y** to “Proceed (y/N).”

```
# mkfs.ocfs2 -T mail /dev/xvdb1
mkfs.ocfs2 1.8.0
Cluster stack: classic o2cb
Overwriting existing ocfs2 partition.
Proceed (y/N) : y
Filesystem Type of mail
...
Journal size: 268435456
...
mkfs.ocfs2 successful
```

- Note the larger journal size: 268435456
  - All other settings are the defaults.
- c. Use the `mkfs.ocfs2` command to create a file system with the `-T datafiles` option.
- Specify this type when you intend to use the file system for database files.
  - These file types use fewer fully allocated large files, with fewer metadata changes, and do not benefit from a large journal.
  - Answer **y** to “Proceed (y/N).”

```
# mkfs.ocfs2 -T datafiles /dev/xvdb1
mkfs.ocfs2 1.8.0
Cluster stack: classic o2cb
Overwriting existing ocfs2 partition.
Proceed (y/N) : y
Filesystem Type of datafiles
...
Cluster size: 131072 (17 bits)
...
Cluster groups: 3 (tail covers 17400 clusters, rest cover 32256
clusters)
Extent allocator size: 8388608 (2 groups)
Journal size: 33554432
...
mkfs.ocfs2 successful
```

- Note the differences from the default file system:
  - Cluster size: 131072 (17 bits). Default is 4096 (12 bits).
  - Cluster groups: 3. Default is 82.
  - Extent allocator size: 8388608 (2 groups). Default is 4194304 (1
 groups).
  - Journal size: 33554432. Default is 67108864.

- d. Use the `mkfs.ocfs2` command to create a file system with the `-T vmstore` option.
- Specify this type when you intend to store virtual machine images.
  - These file types are sparsely allocated large files and require moderate metadata updates.
  - Answer `y` to “Proceed (y/N).”

```
# mkfs.ocfs2 -T vmstore /dev/xvdb1
mkfs.ocfs2 1.8.0
Cluster stack: classic o2cb
Overwriting existing ocfs2 partition.
Proceed (y/N): y
Filesystem Type of vmstore
...
Cluster size: 131072 (17 bits)
...
Cluster groups: 3 (tail covers 17400 clusters, rest cover 32256
clusters)
Extent allocator size: 8388608 (2 groups)
Journal size: 134217728
...
mkfs.ocfs2 successful
```

- Note that the differences from the default file system are similar to the settings for the `datafiles` file system type.
  - The exception is the journal size: 134217728
- e. Use the `mkfs.ocfs2` command to create a file system with the label `myvolume`.
- Answer `y` to “Proceed (y/N).”

```
# mkfs.ocfs2 -L "myvolume" /dev/xvdb1
mkfs.ocfs2 1.8.0
Cluster stack: classic o2cb
Overwriting existing ocfs2 partition.
Proceed (y/N): y
Label: myvolume
...
mkfs.ocfs2 successful
```

- Note that a label is assigned: `myvolume`
- All other settings are defaults.

## Practice 11-6: Mounting an OCFS2 Volume

---

### Overview

In this practice, you perform the following:

- Mount the shared OCFS2 volume on each VM: **host01**, **host02**, and **host03**.
- Disable `iptables` to allow the heartbeat between the nodes.
- Copy files to the shared volume from one host and remove the copied files from another host.
- Create a file on the shared volume from one host and edit the file from another host.

### Assumptions

You are the `root` user on all VM guests.

### Tasks

1. From **host01**, mount the OCFS2 volume.

- a. Use the `mkdir` command to make a mount point, `/u01`, for the OCFS2 volume.

```
[host01]# mkdir /u01
```

- b. Use the `mount` command to mount the OCFS2 volume by label `myvolume` on the `/u01` mount point.

- There is a delay in the clustered mount operation.

```
[host01]# mount -L myvolume /u01
```

- c. Use the `service o2cb status` command to display the status of the O2CB heartbeat.

- Note that the heartbeat is active after the volume is mounted.

```
[host01]# service o2cb status
```

```
...
```

```
Checking O2CB heartbeat: Active
```

- d. Use the `cp` command to copy `/boot/vmlinuz*` files to `/u01`. Use the `ls` command to display the contents of `/u01`.

- You are able to copy files to the OCFS2 volume.

```
[host01]# cp /boot/vmlinuz* /u01
```

```
[host01]# ls /u01
```

```
lost+found
```

```
vmlinuz...
```

2. From **host02**, mount the OCFS2 volume.

- a. Use the `mkdir` command to make a mount point, `/u01`, for the OCFS2 volume.

```
[host02]# mkdir /u01
```

- b. Use the `mount` command to mount the OCFS2 volume by label `myvolume` on the `/u01` mount point.

- Note that the mount fails on **host02** with a “can’t find LABEL” error message.

```
[host02]# mount -L myvolume /u01
mount: can't find LABEL="myvolume"
```

- c. Use the `cat` command to display the `/proc/partitions` file.

- Note that the `xvdb1` partition is not listed.

```
[host02]# cat /proc/partitions
major minor #blocks name
 202      0   20971520 xvda
 202      1    512000 xvda1
 202      2   20458496 xvda2
 202     16   10485760 xvdb
 202     48   20971520 xvdd
 11      0    4193280 sr0
 252      0   18317312 dm-0
 252      1   2097152 dm-1
```

- d. Use the `partprobe` command on `/dev/xvdb` to inform the OS of partition changes.

```
# partprobe /dev/xvdb
```

- e. Use the `cat` command to display the `/proc/partitions` file.

- Note that the `xvdb1` partition is now listed.

```
[host02]# cat /proc/partitions
major minor #blocks name
 202      0   20971520 xvda
 202      1    512000 xvda1
 202      2   20458496 xvda2
 202     16   10485760 xvdb
 202     17   10485760 xvdb1
 202     48   20971520 xvdd
 11      0    4193280 sr0
 252      0   18317312 dm-0
 252      1   2097152 dm-1
```

- f. Use the `mount` command to mount the OCFS2 volume by label `myvolume` on the `/u01` mount point.

- Note that after a short delay, the mount fails on **host02** but with a different error message.

```
[host02]# mount -L myvolume /u01
mount.ocfs2: Transport endpoint is not connected while mounting
/dev/xvdb1 on /u01. Check 'dmesg' for more information on this
error.
```

- g. Use the `dmesg` command to determine the cause of the failed mount.
- A number of `o2net` and `ocfs2` messages appear in the output of `dmesg`.
  - The `o2net` error indicates that the nodes are unable to establish a heartbeat on port 7777.

```
[host02]# dmesg
...
o2net: Connection to node host01 (num 0) at 192.168.1.200:7777
shutdown, state 7
...
```

- h. Use the `systemctl stop firewalld` command on both **host01** and **host02** to disable `firewalld` and allow communication over port 7777.

```
[host01]# systemctl stop firewalld
[host02]# systemctl stop firewalld
```

- i. From **host02**, use the `mount` command to mount the OCFS2 volume by label `myvolume` on the `/u01` mount point.
- The `mount` command succeeds now that communication over port 7777 is allowed.

```
[host02]# mount -L myvolume /u01
```

- j. Use the `service o2cb status` command to display the status of the O2CB heartbeat.
- Note that the heartbeat is active after the volume is mounted.

```
[host02]# service o2cb status
...
Checking O2CB heartbeat: Active
```

- k. Use the `ls` command to display the contents of `/u01`.
- Note that the `vmlinuz*` files, which were copied to the volume from **host01**, are present on the volume mounted on **host02**.

```
[host02]# ls /u01
lost+found
vmlinuz...
```

3. From **host03**, mount the OCFS2 volume.

- a. Use the `mkdir` command to make a mount point, `/u01`, for the OCFS2 volume.

```
[host03]# mkdir /u01
```

- b. Use the `partprobe` command on `/dev/xvdb` to inform the OS of partition changes.

```
# partprobe /dev/xvdb
```

- c. Use the `systemctl stop firewalld` command to disable `firewalld` and allow communication over port 7777.

```
[host03]# systemctl stop firewalld
```

- d. Use the `mount` command to mount the OCFS2 volume by label `myvolume` on the `/u01` mount point.

- The `mount` command succeeds.

```
[host03]# mount -L myvolume /u01
```

- e. Use the `service o2cb status` command to display the status of the O2CB heartbeat.

- Note that the heartbeat is active after the volume is mounted.

```
[host03]# service o2cb status
```

```
...
```

```
Checking O2CB heartbeat: Active
```

- f. Use the `ls` command to display the contents of `/u01`.

- Note that the `vmlinuz*` files, which were copied to the volume from **host01**, are present on the volume mounted on **host03**.

```
[host03]# ls /u01
```

```
lost+found
```

```
vmlinuz...
```

4. Test the write and read properties of the shared OCFS2 volume.

- a. From **host02**, use the `rm` command to remove the `vmlinuz*` files on `/u01`.

- Answer `y` to remove the `vmlinuz*` files on `/u01`.

```
[host02]# rm /u01/vmlinuz*
```

```
rm: remove regular file '/u01/vmlinuz...? y
```

```
...
```

- b. From **host02**, use the `vi` editor to create a file on `/u01` named `host02_test`.

- Enter the following contents in the `host02_test` file:

```
[host02]# vi /u01/host02_test
```

```
This is a test file created from host02.
```

- c. From **host01**, use the `ls` command to list the contents of `/u01`.

```
[host01]# ls /u01
```

```
lost+found
```

```
host02_test
```

- d. From **host01**, use the `vi` editor to edit the `host02_test` file on `/u01`.

- Add the second line as follows:

```
[host01]# vi /u01/host02_test
```

```
This is a test file created from host02.
```

```
This file was edited from host01.
```

- e. From **host02**, use the `cat` command to display the contents of the `/u01/host02_test` file.

```
[host02]# cat /u01/host02_test
```

```
This is a test file created from host02.
```

```
This file was edited from host01.
```

- f. From **host03**, use the `cat` command to display the contents of the `/u01/host02_test` file.
- Note that a file created by one host on a shared volume can be edited from another host.

```
[host03]# cat /u01/host02_test
This is a test file created from host02.
This file was edited from host01.
```

## Practice 11-7: Tuning and Debugging OCFS2

---

### Overview

In this practice, you perform the following:

- Query the file system attributes of an OCFS2 volume.
- Enable and disable file system features.
- Detect and fix errors on the file system.
- List OCFS2 volumes and the nodes mounting the OCFS2 volumes.

### Assumptions

You are the `root` user on all VM guests.

### Tasks

1. Query file system attributes.

From `host01`, use the `tunefs.ocfs2` command to display the following OCFS2 file system attributes for `/dev/xvdb1`. Format the command to include the attribute names. Print each attribute on a new line.

- Block Size (%B)
- Cluster Size (%T)
- Number of Node Slots (%N)
- Volume Label (%V)
- Volume UUID (%U)
- Compatible Features (%M)
- Incompatible Features (%H)
- Read-only Compatible Features (%O)

**Note:** The “\” character at the end of each line is optional. Inserting “\” and pressing ENTER provides a new line for the same command-line command. The new line begins with the “>” character, which does not need to be typed by you. You can eliminate the “\” character and continue typing the entire command on a single line if desired.

```
[host01]# tunefs.ocfs2 -Q "Block Size: %B\n\
> Cluster Size: %T\nNumber of Node Slots: %N\n\
> Volume Label: %V\nVolume UUID: %U\n\
> Compatible Features: %M\nIncompatible Features: %H\n\
> Read-Only Compatible Features: %O\n" /dev/xvdb1
Block Size: 4096
Cluster Size: 4096
Number of Node Slots: 4
Volume Label: myvolume
Volume UUID: ...
Compatible Features: backup-super strict-journal-super
Incompatible Features: sparse extended-slotmap inline-data xattr
indexed-dirs refcount discontig-bg
```

### Read-Only Compatible Features: unwritten

- The file system features are split up into three categories:
  - Compatible** is a feature that the file system does not need to fully understand to safely read/write to the volume. An example of this is the `backup-super` feature. Because the `backup-super` blocks are typically not read or written, an older file system can safely mount a volume with this feature enabled.
  - Incompatible** is a feature that the file system needs to fully understand to read/write to the volume. Most features fall under this category.
  - Read-only Compatible** is a feature that the file system needs to fully understand to write to the volume. Older software can safely read a volume with this feature enabled. An example of this would be user and group quotas and the `unwritten` feature.
- The following table lists the features, feature category, kernel version, and the `ocfs2-tools` version in which each feature became available.

| Features                                                                                                                   | Category     | Kernel Version | Tools Version                |
|----------------------------------------------------------------------------------------------------------------------------|--------------|----------------|------------------------------|
| <code>backup-super</code> – Indicates that the volume has backups of the super block                                       | Compatible   | All            | <code>ocfs2-tools</code> 1.2 |
| <code>strict-journal-super</code> – Indicates that the file system is using version 2 of the JBD super block               | Compatible   | All            | All                          |
| <code>local</code> – Is enabled when a volume is mounted without a cluster stack. It is also referred to as a local mount. | Incompatible | Linux 2.6.20   | <code>ocfs2-tools</code> 1.2 |
| <code>sparse</code> – Allows the file system to be efficient in terms of both performance and space usage                  | Incompatible | Linux 2.6.22   | <code>ocfs2-tools</code> 1.4 |
| <code>inline-data</code> – Allows the file system to store small files and directories in the inode block itself           | Incompatible | Linux 2.6.24   | <code>ocfs2-tools</code> 1.4 |
| <code>extended-slotmap</code> – Is used to map mounted nodes to system file resources                                      | Incompatible | Linux 2.6.27   | <code>ocfs2-tools</code> 1.6 |
| <code>xattr</code> – Is an                                                                                                 | Incompatible | Linux 2.6.29   | <code>ocfs2-tools</code> 1.6 |

|                                                                                                                                                      |                      |              |                 |
|------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------|--------------|-----------------|
| extended attribute.<br>Extended attributes are name:value pairs that can be associated within a file system                                          |                      |              |                 |
| <b>indexed-dirs</b> – Is a file system that creates indexes in directories to improve directory entry lookup performance                             | Incompatible         | Linux 2.6.30 | ocfs2-tools 1.6 |
| <b>metaecc</b> – Enables the file system to compute and validate the checksums for all metadata blocks. It allows for better metadata integrity.     | Incompatible         | Linux 2.6.29 | ocfs2-tools 1.6 |
| <b>refcount</b> – Enables the creation of reference counted (refcount) trees that are required to support reflinks                                   | Incompatible         | Linux 2.6.32 | ocfs2-tools 1.6 |
| <b>discontig-bg</b> – Allows the file system to grow the inode and the extent allocators even when there is no large contiguous free chunk available | Incompatible         | Linux 2.6.35 | ocfs2-tools 1.6 |
| <b>clusterinfo</b> – Enables storing the cluster stack information in the super block                                                                | Incompatible         | Linux 2.6.37 | ocfs2-tools 1.8 |
| <b>unwritten</b> – Allows applications to instantly pre-allocate large extents within a file                                                         | Read-Only Compatible | Linux 2.6.23 | ocfs2-tools 1.4 |

|                                                                                                           |                      |              |                 |
|-----------------------------------------------------------------------------------------------------------|----------------------|--------------|-----------------|
| <b>usrquota</b> – Allows the file system to track the amount of space and number of inodes for each user  | Read-Only Compatible | Linux 2.6.29 | ocfs2-tools 1.6 |
| <b>grpquota</b> – Allows the file system to track the amount of space and number of inodes for each group | Read-Only Compatible | Linux 2.6.29 | ocfs2-tools 1.6 |

- The file system has all the available file system features enabled with the following exceptions: local, metaecc, usrquota, and grpquota.
2. Enable and disable features on the file system.
- Use the `tunefs.ocfs2` command to enable the `metaecc` and `usrquota` features on `/dev/xvdb1`.
- ```
[host01]# tunefs.ocfs2 --fs-features=metaecc,usrquota /dev/xvdb1
tunefs.ocfs2: Trylock failed while opening device "/dev/xvdb1"
```
- Note that the command fails.
  - Use the `umount` command to unmount `/u01`, and then repeat the previous command.
  - There is a slight delay in unmounting the file system.
- ```
[host01]# umount /u01
[host01]# tunefs.ocfs2 --fs-features=metaecc,usrquota /dev/xvdb1
tunefs.ocfs2: Trylock failed while opening device "/dev/xvdb1"
```
- Note that the command still fails.
  - Use the `umount` command to unmount `/u01` on the other nodes—**host02** and **host03**—and then repeat the previous `tunefs.ocfs2` command on **host01**.
- ```
[host02]# umount /u01
[host03]# umount /u01
[host01]# tunefs.ocfs2 --fs-features=metaecc,usrquota /dev/xvdb1
```
- The command succeeds when the volume was unmounted on all nodes.
  - The volume needs to be unmounted across the cluster before enabling or disabling any features.
  - Repeat the `tunefs.ocfs2` command that was executed in step 1.

```
[host01]# tunefs.ocfs2 -Q "Block Size: %B\nCluster Size:
%T\nNumber of Node Slots: %N\nVolume Label: %V\nVolume UUID:
%U\nCompatible Features: %M\nIncompatible Features: %H\nRead-
Only Compatible Features: %O\n" /dev/xvdb1
Block Size: 4096
Cluster Size: 4096
Node Slots: 4
Volume Label: myvolume
Volume UUID: ...
Compatible Features: backup-super strict-journal-super
```

```
Incompatible Features: sparse extended-slotmap inline-data
metaecc xattr indexed-dirs refcount discontig-bg
Read-Only Compatible Features: unwritten usrquota
```

- Note that the new features, **metaecc** and **usrquota**, are enabled (in **bold**).
- e. Use the `tunefs.ocfs2` command to disable the `metaecc` and `usrquota` features on `/dev/xvdb1`.
- Note that an error occurs but the features are removed. This is a known bug.

```
[host01]# tunefs.ocfs2 --fs-features=nometaecc,nousrquota
/dev/xvdb1
tunefs.ocfs2: I/O error on channel while closing device
"/dev/xvdb1"
```

- f. Repeat the `tunefs.ocfs2` command that was executed in step 1.

```
[host01]# tunefs.ocfs2 -Q "Block Size: %B\nCluster Size:
%T\nNumber of Node Slots: %N\nVolume Label: %V\nVolume UUID:
%U\nCompatible Features: %M\nIncompatible Features: %H\nRead-
Only Compatible Features: %O\n" /dev/xvdb1
Block Size: 4096
Cluster Size: 4096
Node Slots: 4
Volume Label: myvolume
Volume UUID: ...
Compatible Features: backup-super strict-journal-super
Incompatible Features: sparse extended-slotmap inline-data xattr
indexed-dirs refcount discontig-bg
Read-Only Compatible Features: unwritten
```

3. Detect any errors on the file system.

Use the `fsck.ocfs2` command to detect and fix errors on `/dev/xvdb1`.

```
[host01]# fsck.ocfs2 -f /dev/xvdb1
fsck.ocfs2 1.8.0
Checking OCFS2 filesystem in /dev/xvdb1:
  Label: myvolume
  UUID: ...
  Number of blocks: 2621184
  Block size: 4096
  Number of clusters: 2621184
  Cluster size: 4096
  Number of slots: 4

/dev/xvdb1 was run with -f, check forced.
Pass 0a: Checking cluster allocation chains
Pass 0b: Checking inode allocation chains
Pass 0c: Checking extent block allocation chains
Pass 1: Checking inodes and blocks.
```

```

Pass 2: Checking directory entries.
Pass 3: Checking directory connectivity.
Pass 4a: checking for orphaned inodes
Pass 4b: Checking inodes link counts.
All passes succeeded.

```

4. List mounted volumes.

- a. Use the `mounted.ocfs2` command to list all OCFS2 volumes.

```
[host01]# mounted.ocfs2 -d
Device      Stack  Cluster   F   UUID                           Label
/dev/xvdb1   o2cb          ...                                myvolume
```

- b. Use the `mounted.ocfs2` command to list the nodes that are currently mounting each OCFS2 volume.

```
[host01]# mounted.ocfs2 -f
Device      Stack  Cluster   F   Nodes
/dev/xvdb1   o2cb          Not mounted
```

- c. Use the `mount` command to mount the OCFS2 volume by label `myvolume` on the `/u01` mount point.

```
[host01]# mount -L myvolume /u01
```

- d. Use the `mounted.ocfs2` command to list the nodes that are currently mounting each OCFS2 volume.

```
[host01]# mounted.ocfs2 -f
Device      Stack  Cluster   F   Nodes
/dev/xvdb1   o2cb          host01
```

- e. From `host02`, use the `mount` command to mount the OCFS2 volume by label `myvolume` on the `/u01` mount point.

```
[host02]# mount -L myvolume /u01
```

- f. Use the `mounted.ocfs2` command, from either `host01` or `host02`, to list the nodes that are currently mounting each OCFS2 volume.

```
# mounted.ocfs2 -f
Device      Stack  Cluster   F   Nodes
/dev/xvdb1   o2cb          host01, host02
```

5. Restore the VMs to their original state.

- a. Use the `umount` command to unmount `/u01` from the `host01` and `host02` VMs.

```
[host01]# umount /u01
[host02]# umount /u01
```

- b. Use the `systemctl` command to stop the `o2cb` service on all VMs.

```
[host01]# systemctl stop o2cb
[host02]# systemctl stop o2cb
[host03]# systemctl stop o2cb
```

- c. Use the `systemctl` command to disable the `o2cb` service from starting at boot on all VMs.

```
[host01]# systemctl disable o2cb
rm '/etc/systemd/system/multi-user.target.wants/o2cb.service'
[host02]# systemctl disable o2cb
rm '/etc/systemd/system/multi-user.target.wants/o2cb.service'
[host03]# systemctl disable o2cb
rm '/etc/systemd/system/multi-user.target.wants/o2cb.service'
```

- d. Use the `systemctl` command to bring down all the VMs.

```
[host01]# systemctl poweroff
...
[host02]# systemctl poweroff
...
[host03]# systemctl poweroff
...
```

- e. From **dom0**, run the `xm list` command to confirm that all VM domains are shut down.
- When all VMs are shut down, only **dom0** appears in the output of the `xm list` command.

<code>[dom0]# xm list</code>					
Name	ID	Mem	VCPUs	State	Time (s)
Domain-0	0	2048	2	r-----	281.1

- f. From the `/OVS/running_pool/host01` directory on **dom0**, use the `cp` command to restore the **host01** `vm.cfg` file from the backup made in Practice 9-1.

```
[dom0]# cd /OVS/running_pool/host01
[dom0]# cp vm.cfg.BAK vm.cfg
cp: overwrite 'vm.cfg'? y
```

- g. From the `/OVS/running_pool/host03` directory on **dom0**, use the `cp` command to restore the **host03** `vm.cfg` file from the backup made in Practice 9-1.

```
[dom0]# cd /OVS/running_pool/host03
[dom0]# cp vm.cfg.BAK vm.cfg
cp: overwrite 'vm.cfg'? y
```

- h. From the `/OVS/running_pool/host01` directory on **dom0**, run the `xm create vm.cfg` command to start the **host01** VM.

```
[dom0]# cd /OVS/running_pool/host01
[dom0]# xm create vm.cfg
Using config file "./vm.cfg".
Started domain host01 (id=#)
```

- i. From the /OVS/running\_pool/host02 directory on **dom0**, run the `xm create vm.cfg` command to start the **host02** VM.

```
[dom0]# cd /OVS/running_pool/host02
[dom0]# xm create vm.cfg
Using config file "./vm.cfg".
Started domain host02 (id=#)
```

- j. From the /OVS/running\_pool/host03 directory on **dom0**, run the `xm create vm.cfg` command to start the **host03** VM.

```
[dom0]# cd /OVS/running_pool/host03
[dom0]# xm create vm.cfg
Using config file "./vm.cfg".
Started domain host03 (id=#)
```

- k. From **dom0**, run the `xm list` command to confirm whether all VM domains are running.

# xm list					
Name	ID	Mem	VCPUs	State	Time(s)
Domain-0	0	2048	2	r-----	304.5
host01	4	1536	1	-b----	18.7
host02	2	1536	1	-b----	159.0
host03	3	1536	1	-b----	13.2

- In this example, all VMs are running. The ID and Time (s) values are examples.



## **Practices for Lesson 12: iSCSI and Multipathing**

**Chapter 12**

## Practices for Lesson 12: Overview

---

### Practices Overview

In these practices, you perform the following:

- Configure **host02** as an iSCSI target
- Configure **host03** as an iSCSI initiator
- Configure iSCSI multipathing on **host03**

## Practice 12-1: Configuring an iSCSI Server (Target)

---

### Overview

In this practice, you:

- Configure **host02** as an iSCSI server
- Install the `targetcli` software package
- Create block backstore storage objects
- Create an iSCSI target
- Add the Logical Unit Numbers (LUNs) to the Target Portal Group (TPG)
- Change the network portal
- Configure Access Control Lists (ACLs)
- Trust port 3260 for the Transmission Control Protocol (TCP) in the `firewalld` configuration

### Assumptions

- You are the `root` user on **dom0**.

### Tasks

1. From **dom0**, use the `ssh` command to log in to **host02**.

- The `root` password is `oracle`.

```
[dom0]# ssh host02
root@host02's password: oracle
Last login: ...
[root@host02 ~]#
```

2. Modify the partition table on **host02**.

- a. Use the `fdisk -l` command to view the partition table. Pipe the output to `grep` and search for the string “/dev”.
- Note that there are two 10.7 GB storage devices: `/dev/xvdb` and `/dev/xvde`.
- You create two block backstore storage objects from `/dev/xvdb` and `/dev/xvde`.

```
# fdisk -l | grep /dev
Disk /dev/xvda: 21.5 GB, 21474836480 bytes, 41943040 sectors
 /dev/xvda1      *     2048    1026047    512000    83    Linux
 /dev/xvda2        1026048   41943040   20458496    8e    Linux LVM
Disk /dev/xvdb: 10.7 GB, 10737418240 bytes, 20971520 sectors
 /dev/xvdb1        2048    20971519    10484736    83    Linux
Disk /dev/xvdd: 21.5 GB, 21474836480 bytes, 41943040 sectors
Disk /dev/xvde: 10.7 GB, 10737418240 bytes, 20971520 sectors
Disk /dev/mapper/ol-root: 18.8 GB, 18798870528 bytes, ...
Disk /dev/mapper/ol-swap: 2147 MB, 2147483648 bytes, ...
```

- b. Use the `fdisk /dev/xvdb` command and delete the `/dev/xvdb1` partition.

```
# fdisk /dev/xvdb
...
Command (m for help): d
Selected partition 1
Partition 1 is deleted

Command (m for help): w
The partition table has been altered!
...
```

- c. Repeat step 2a to view the partition table.

- Note that `/dev/xvdb1` no longer exists.

```
# fdisk -l | grep /dev
Disk /dev/xvda: 21.5 GB, 21474836480 bytes, 41943040 sectors
/dev/xvda1      *     2048    1026047    512000    83    Linux
/dev/xvda2        1026048   41943040   20458496    8e    Linux LVM
Disk /dev/xvdb: 10.7 GB, 10737418240 bytes, 20971520 sectors
Disk /dev/xvdd: 21.5 GB, 21474836480 bytes, 41943040 sectors
Disk /dev/xvde: 10.7 GB, 10737418240 bytes, 20971520 sectors
Disk /dev/mapper/ol-root: 18.8 GB, 18798870528 bytes, ...
Disk /dev/mapper/ol-swap: 2147 MB, 2147483648 bytes, ...
```

3. Install the `targetcli` software package and enable and start the `target` service.

- a. Use the `yum` command to install the `targetcli` software package on **host02**.

- Answer `y` to “Is this ok.”

```
# yum install targetcli
...
Is this ok [y/d/N]: y
...
Complete!
```

- b. Use the `systemctl` command to enable and start the `target` service.

```
# systemctl enable target
ln -s '/usr/lib/systemd/system/target.service'
'/etc/systemd/system/multi-user.target.wants/target.service'
# systemctl start target
```

4. Explore the `targetcli` command-line interface.

- a. Run the `targetcli` command to access the command-line interface.

```
# targetcli
targetcli shell version 2.1.fb37
Copyright 2011-2013 by Datera, Inc and others.
For help on commands, type 'help'.
```

```
/>
```

b. Use the `help` command to view the available commands.

- You can get help on specific commands by entering `help <command>`.

```
/> help
```

#### GENERALITIES

```
=====
```

This is a shell in which you can create, delete and configure configuration objects.

The available commands depend on the current path or target path you want to run a command in: different paths have different sets of available commands, i.e. a path pointing at an iSCSI target will not have the same available commands as, say, a path pointing at a storage object.

The prompt that starts each command line indicates your current path. Alternatively (useful if the prompt displays an abbreviated path to save space), you can run the `pwd` command to display the complete current path.

```
...
```

#### AVAILABLE COMMANDS

```
=====
```

The following commands are available in the current path:

- bookmarks action [bookmark]
- cd [path]
- clearconfig [confirm]
- exit
- get [group] [parameter...]
- help [topic]
- ls [path] [depth]
- pwd
- refresh
- restoreconfig [savefile] [clear\_existing]
- saveconfig [savefile]
- sessions [action] [sid]
- set [group] [parameter=value...]
- status
- version

- c. Use the `get` command to view the available configuration groups.

```
/> get

AVAILABLE CONFIGURATION GROUPS
=====
global
```

- d. Use the `get global` command to view the configuration parameters.

```
/> get global

GLOBAL CONFIG GROUP
=====
auto_add_default_portal=true
-----
If true, adds a portal listening on all IPs to new targets.

auto_add_mapped_luns=true
-----
If true, automatically create node ACLs mapped LUNs after
creating a new target LUN or a new node ACL

auto_cd_after_create=false
-----
If true, changes current path to newly created objects.

auto_enable_tpgt=true
-----
If true, automatically enables TPGTs upon creation.

auto_save_on_exit=true
-----
If true, saves configuration on exit.

...
tree_show_root=true
-----
Whether or not to display tree root.

tree_status_mode=true
-----
Whether or not to display status in tree.
```

- e. Use the `ls` command to view the object hierarchy.

- Note that the initial object hierarchy is empty.

```
/> ls
o- / ..... [...]
o- backstores ..... [...]
| o- block ..... [Storage Objects: 0]
| o- fileio ..... [Storage Objects: 0]
| o- pscsi ..... [Storage Objects: 0]
| o- ramdisk ..... [Storage Objects: 0]
o- iscsi ..... [Targets: 0]
o- loopback ..... [Targets: 0]
```

- f. Use the `help` command to get help on the `create` command.

- Note that the `create` command is not available from the current root-level directory.

```
/> help create
Cannot find help topic create.
```

5. Create block backstore storage objects.

- a. Use the `cd` command to change to the `/backstores/block` directory.

```
/> cd /backstores/block
/backstores/block>
```

- b. Use the `help` command to get help on the `create` command.

- Note that the `create` command is available from the `/backstores/block` directory.

```
/backstores/block> help create

SYNTAX
=====
create name dev [readonly]

DESCRIPTION
===========
Creates a Block Storage object. Dev is the path to the TYPE_DISK
block device to use.
```

- c. Use the `create` command to create the following named block storage objects:

- LUN\_1 on `/dev/xvdb`
- LUN\_2 on `/dev/xvde`

```
/backstores/block> create name=LUN_1 dev=/dev/xvdb
'export_backstore_name_as_model' is set but emulate_model_alias
not supported by kernel.
/backstores/block> create name=LUN_2 dev=/dev/xvde
'export_backstore_name_as_model' is set but emulate_model_alias
not supported by kernel.
```

- d. Use the `ls` command to view the block storage objects.

```
/backstores/block> ls
o- block ..... [Storage Objects: 2]
  o- LUN_1 ..... [/dev/xvdb (10.0GiB) write-thru deactivated]
  o- LUN_2 ..... [/dev/xvde (10.0GiB) write-thru deactivated]
```

6. Create an iSCSI target.

- a. Use the `cd` command to change to the `/iscsi` directory.

```
/backstores/block> cd /iscsi
/iscsi>
```

- b. Use the `help` command to get help on the `create` command.

- Note that the `create` command usage is different when issuing the command from the `/iscsi` directory than it is when issuing from the `/backstores/block` directory.

```
/iscsi> help create

SYNTAX
=====
create [wwn]

DESCRIPTION
===========
Creates a new target. The wwn format depends on the transport(s)
supported by the fabric module. If the wwn is omitted, then a
target will be created using either a randomly generated WWN of
the proper type, or the first unused WWN in the list of possible
WWNs if one is available...
...
```

- c. Use the `create` command to create an IQN (iSCSI Qualified Name) called `iqn.2015-07.com.example.host02` with a target named `tgt1`.

- A Target Portal Group (TPG) is created as a result of this command.

```
/iscsi> create iqn.2015-07.com.example.host02:tgt1
Created target iqn.2015-07.com.example.host02:tgt1.
Created TPG 1.
Global pref auto_add_default_portal=true
Created default portal listening on all IPs (0.0.0.0), port
3260.
```

- d. Use the `ls` command to view the TPG hierarchy.
- Note that the initial TPG hierarchy is empty with exception of a network portal.

```
/iscsi> ls
o- iscsi ..... [Targets: 1]
  o- iqn.2015-07.com.example.host02:tgt1 ..... [TPGs: 1]
    o- tpg1 ..... [no-gen-acls, no-auth]
      o- acls ..... [ACLs: 0]
      o- luns ..... [LUNs: 0]
      o- portals ..... [Portals: 1]
        o- 0.0.0.0:3260 ..... [OK]
```

7. Add the LUNs to the TPG.

- a. Use the `cd` command to change to the `iqn.2015-07.com.example.host02:tgt1/tpg1/luns` directory.
- You can use the TAB key to complete the pathname.
  - For example, type `iqn` then press the TAB key to complete the pathname.

```
/iscsi> cd iqn.2015-07.com.example.host02:tgt1/tpg1/luns
/iscsi/iqn.20...gt1/tpg1/luns>
```

- b. Use the `help` command to get help on the `create` command.
- Note that the `create` command in this directory creates a new LUN in the TPG.

```
/iscsi/iqn.20...gt1/tpg1/luns> help create

SYNTAX
=====
Create storage_object [lun] [add_mapped_luns]

DESCRIPTION
===========
Creates a new LUN in the Target Portal Group, attached to a
storage object. If the lun parameter is omitted, the first
available LUN in the TPG will be used. If present, it must be a
number greater than 0. Alternatively, the syntax lunX where X
is a positive number is also accepted.

...
```

c. Use the `create` command to create two LUNS as follows:

- Create LUN1 from `/backstores/block/LUN_1`.
- Create LUN2 from `/backstores/block/LUN_1`.

```
/iscsi/iqn.20.../luns> create /backstores/block/LUN_1 lun1
Created LUN 1.
/iscsi/iqn.20.../luns> create /backstores/block/LUN_2 lun2
Created LUN 2.
```

- d. Use the `ls` command to view the LUNs.

```
/iscsi/iqn.20.../luns> ls
o- luns ..... [LUNs: 2]
  o- lun1 ..... [block/LUN_1 (/dev/xvdb)]
  o- lun2 ..... [block/LUN_2 (/dev/xvde)]
```

8. Change the portal to the IP address of **host02**.

- Use 192.0.2.102, which is the address on the public network for **host02**.
- a. Use the `cd` command to change to the `portals` directory.
  - The `portals` directory is on the same level as the `luns` directory in the hierarchy.
  - Use the `pwd` command to view the current location in the hierarchy.

```
/iscsi/iqn.20.../luns> cd ../portals
/iscsi/iqn.20.../portals> pwd
/iscsi/iqn.2015-07.com.example.host02:tgt1/tpg1/portals
```

- b. Use the `ls` command to view the current portals.

- Note that the current portal of 0.0.0.0:3260 is created by default by the `/iscsi create` command.

```
/ iscsi/iqn.20.../portals> ls
o- portals ..... [Portals: 1]
o- 0.0.0.0:3260 ..... [OK]
```

- c. Use the `delete` command to delete the current portal.

```
/ iscsi/iqn.20.../portals> delete 0.0.0.0 3260
Deleted network portal 0.0.0.0:3260
```

- d. Use the `ls` command to view the current portals.

```
/ iscsi/iqn.20.../portals> ls
o- portals ..... [Portals: 0]
```

- e. Use the `help` command to get help on the `create` command.

- Note that the `create` command in this directory creates a network portal.

```
/iscsi/iqn.20.../portals> help create

SYNTAX
=====
create [ip_address] [ip_port]

DESCRIPTION
===========
Creates a Network Portal with specified ip_address and
ip_port. If ip_port is omitted, the default port for
the target fabric will be used. If ip_address is omitted,
INADDR_ANY (0.0.0.0) will be used.

...
```

- f. Use the `create` command to specify 192.0.2.102 as the portal IP address.

```
/iscsi/iqn.20.../portals> create 192.0.2.102
Using default IP port 3260
Created network portal 192.0.2.102:3260
```

- g. Use the `ls` command to view the current portals.

```
/ iscsi/iqn.20.../portals> ls
o- portals ..... [Portals: 1]
o- 192.0.2.102:3260 ..... [OK]
```

9. Configure ACLs.

- Create an ACL for each initiator.
- a. Use the `cd` command to change to the `acls` directory.
  - The `acls` directory is on the same level as the `portals` directory in the hierarchy.
  - Use the `pwd` command to view the current location in the hierarchy.

```
/iscsi/iqn.20.../portals> cd ../acls
/iscsi/iqn.20.../acls> pwd
/iscsi/iqn.2015-07.com.example.host02:tgt1/tpg1/acls
```

- b. Use the `help` command to get help on the `create` command.
  - Note that the `create` command in this directory creates a node ACL for the initiator node.

```
/iscsi/iqn.20.../acls> help create

SYNTAX
=====
create wwn [add_mapped_luns]

DESCRIPTION
===========
Creates a Node ACL for the initiator node with the specified
wwn. The node's wwn must match the expected WWN Type of the
target's fabric module.

...
```

- c. Use the `create` command to an ACL with the previously created IQN, `iqn.2015-07.com.example.host02`, for the **host03** initiator node.

```
/iscsi/iqn.20.../acls> create iqn.2015-07.com.example.host02:host03
Created Node ACL for iqn.2015-07.com.example.host02:host03
Created mapped LUN 2.
Created mapped LUN 1.
```

- d. Use the `ls` command to view the ACL.

```
/iscsi/iqn.20.../acls> ls
o- acls ..... [ACLs: 2]
  o- iqn.2015-07.com.example.host02:host03 ..... [Mapped LUNs:2]
    | o- mapped_lun1 ..... [lun1 block/LUN_1 (rw)]
    | o- mapped_lun2 ..... [lun2 block/LUN_2 (rw)]
```

10. View and save the configuration.

- a. Use the `cd` command to change to the root level of the hierarchy.

```
/ iscsi/iqn.20.../acls> cd /
```

- b. Use the `ls` command to view the object hierarchy.

```
/> ls
o- / ..... [...]
  o- backstores ..... [...]
    | o- block ..... [Storage Objects: 2]
      | | o- LUN_1 .... [/dev/xvdb (10.0GiB) write-thru deactivated]
      | | o- LUN_2 .... [/dev/xvde (10.0GiB) write-thru deactivated]
    | o- fileio ..... [Storage Objects: 0]
    | o- pscsi ..... [Storage Objects: 0]
    | o- ramdisk ..... [Storage Objects: 0]
  o- iscsi ..... [Targets: 1]
    | o- iqn.2015-07.com.example.host02:tgt1 ..... [TPGs: 1]
      | o- tpg1 ..... [no-gen-acls, no-auth]
      | o- acls ..... [ACLs: 1]
        | | o- iqn.2015-07.com.example.host02:tgt1:192.0.2.103 ..... [Mapped LUNs:2]
          | | | o- mapped_lun1 ..... [lun1 block/LUN_1 (rw)]
          | | | o- mapped_lun2 ..... [lun2 block/LUN_2 (rw)]
        | o- luns ..... [LUNs: 2]
          | | o- lun1 ..... [block/LUN_1 (/dev/xvdb)]
          | | o- lun2 ..... [block/LUN_2 (/dev/xvde)]
        | o- portals ..... [Portals: 1]
          | o- 192.0.2.102:3260 ..... [OK]
  o- loopback ..... [Targets: 0]
```

- c. Use the `exit` command to quit the `targetcli` utility.

- The configuration is automatically saved to the `/etc/target/saveconfig.json` file.

```
/> exit
Global pref auto_save_on_exit=true
Last 10 configs saved in /etc/target/backup.
Configuration saved to /etc/target/saveconfig.json
```

11. Trust port 3260 for TCP in the firewalld configuration.

- a. Use the firewall-cmd to list the current configuration.

```
# firewall-cmd --list-all
public (default, active)
  interfaces: eth0 eth1
  sources:
  services: dhcpcv6-client ssh
  ports:
  masquerade: no
  forward-ports
  icmp-blocks:
  rich rules:
```

- b. Use the firewall-cmd to open the 3260/tcp port.

- Include the --permanent option.

```
# firewall-cmd --permanent --add-port=3260/tcp
success
```

- c. Use the firewall-cmd to reload the firewalld configuration.

```
# firewall-cmd --reload
success
```

- d. Use the firewall-cmd to list the current configuration.

- Note that ports now specifies 3260/tcp.

```
# firewall-cmd --list-all
public (default, active)
  interfaces: eth0 eth1
  sources:
  services: dhcpcv6-client ssh
  ports: 3260/tcp
  masquerade: no
  forward-ports
  icmp-blocks:
  rich rules:
```

## Practice 12-2: Configuring an iSCSI Client (Initiator)

---

### Overview

In this practice, you:

- Configure **host03** as an iSCSI initiator
- Install the `iscsi-initiator-utils` package
- Configure and start the `iscsid` service
- Discover iSCSI targets by using the `SendTargets` discovery method
- Query the Open-iSCSI persistent database
- Observe the settings in the iSCSI initiator configuration file
- Establish a TCP session between the target and the initiator
- Verify the usability of the iSCSI device

### Assumptions

- You are the `root` user on **dom0**.
- The **host03** VM is running.

### Tasks

1. Configure and start the `iscsid` service on **host03**.
  - a. From **dom0**, open a new terminal window if necessary.
    - Use the `su -` command to become the `root` user. The password is `oracle`.

```
[dom0] $ su -
Password: oracle
[dom0] #
```
  - b. From **dom0**, use the `ssh` command to log in to **host03**.
    - The `root` password is `oracle`.

```
[dom0] # ssh host03
root@host03's password: oracle
[root@host03 ~]#
```
  - c. Use the `yum` command to install the `iscsi-initiator-utils` package on **host03**.
    - In this example, the package is already installed.

```
[host03] # yum install iscsi-initiator-utils
...
Package iscsi-initiator-utils-6.2.0.873-29.0.1.el7.x86_64
already installed and latest version
Nothing to do
```
  - d. Use the `vi` editor to edit `/etc/iscsi/initiatorname.iscsi`.
    - Replace `InitiatorName` with the initiator name that you previously configured as ACL on the **host02** target.
    - In Practice 12-1 step 9c, the ACL you configured was `iqn.2015-07.com.example.host02:host03`.

```
# vi /etc/iscsi/initiatorname.iscsi
InitiatorName=iqn.2015-07.com.example.host02:host03
```

- e. Use the `systemctl` command to enable and start the `iscsid` service.

```
# systemctl enable iscsid
ln -s '/usr/lib/systemd/system/iscsid.service'
'/etc/systemd/system/multi-user.target.wants/iscsid.service'
# systemctl start iscsid
```

2. Discover the iSCSI target at the specified portal.

- In Practice 12-1 step 8f, the network portal you configured was 192.0.2.102.
- a. Use the `iscsiadm` command to discover iSCSI targets by using the `SendTargets` discovery method from IP address (portal) 192.0.2.102.
  - This command starts the `iscsi` service if the service is not already active.

```
# iscsiadm -m discovery -t st -p 192.0.2.102
192.0.2.102:3260,1 iqn.2015-07.com.example.host02:tgt1
```

- b. View the `nodes` and `send_targets` tables in the Open-iSCSI persistent database, `/var/lib/iscsi`.

```
# ls /var/lib/iscsi/nodes
iqn.2015-07.com.example.host02:tgt1
# ls /var/lib/iscsi/send_targets
192.0.2.102,3260
```

- c. Use the `iscsiadm` command to query the `send_targets` table in the persistent database.

```
# iscsiadm -m discoverydb -t st -p 192.0.2.102
# BEGIN RECORD 6.2.0-873-28
discovery.startup = manual
discovery.type = sendtargets
discovery.sendtargets.address = 192.0.2.102
discovery.sendtargets.port = 3260
discovery.sendtargets.auth.authmethod = None
discovery.sendtargets.auth.username = <empty>
discovery.sendtargets.auth.password = <empty>
discovery.sendtargets.auth.username_in = <empty>
discovery.sendtargets.auth.password_in = <empty>
discovery.sendtargets.timeo.login_timeout = 15
discovery.sendtargets.use_discoveryd = No
discovery.sendtargets.discoveryd_poll_inval = 30
discovery.sendtargets.reopen_max = 5
discovery.sendtargets.timeo.auth_timeout = 45
discovery.sendtargets.timeo.active_timeout = 30
discovery.sendtargets.iscsi.MaxRecvDataSegmentLength = 32768
# END RECORD
```

- Much of the information in the database is derived from the settings in the iSCSI initiator configuration file, /etc/iscsi/iscsid.conf.
- d. Use the grep command to search for auth.authmethod in /etc/iscsi/iscsid.conf.
- Challenge-Handshake Authentication Protocol (CHAP) allows iSCSI targets and initiators to prove their identity to each other.
  - CHAP prevents clear-text passwords from appearing on the network.

```
# grep auth.authmethod /etc/iscsi/iscsid.conf
# To enable CHAP authentication set node.session.auth.authmethod
#node.session.auth.authmethod = CHAP
# set discovery.sendtargets.auth.authmethod to CHAP. The default
# is None.
#discovery.sendtargets.auth.authmethod = CHAP
```

3. Establish a TCP session between the iSCSI target and the initiator.

- Log in to establish a session.
  - iSCSI target LUNs are not available until a session is established.
- a. Use the iscscliadm command to view active sessions.

```
# iscscliadm -m session
iscscliadm: No active sessions
```

- b. Use the fdisk command to display the current devices in /dev.
- This example shows three virtual disks, /dev/xvda, /dev/xvdb, and /dev/xvdd.
  - This example also shows LVM volumes for the root and swap partitions.

```
# fdisk -l | grep /dev
Disk /dev/xvda: 12.9 GB, 12884901888 bytes, 25165824 sectors
/dev/xvda1      *     2048      1026047      512000    83    Linux
/dev/xvda2        1026048     25165823     12069888    8e    Linux LVM
Disk /dev/xvdb: 10.7 GB, 10737418240 bytes, 20971520 sectors
/dev/xvdb1        2048     2099199     1048576    83    Linux
Disk /dev/xvdd: 10.7 GB, 10737418240 bytes, 20971520 sectors
Disk /dev/mapper/ol-root: 11.1 GB, 11068768256 bytes, ...
Disk /dev/mapper/ol-swap: 1287 MB, 1287651328 bytes, ...
```

- c. Use the iscscliadm command to log in and establish a session.

```
# iscscliadm -m node -l
Logging in to [iface: default, target: iqn.2015-07.com.example.host02:tgt1, portal: 192.0.2.102:3260] (multiple)
Login to [iface: default, target: iqn.2015-07.com.example.host02:tgt1, portal: 192.0.2.102:3260]
successful.
```

- d. Use the iscscliadm command to view active sessions.

```
# iscscliadm -m session
tcp: [1] 192.0.2.102:3260,1 iqn.2015-07.com.example.host02:tgt1
(non-flash)
```

- e. Use the `fdisk -l` command to display the current devices in `/dev`.

- The two LUNs from the iSCSI target are now available as SCSI block devices, `/dev/sda` and `/dev/sdb`.

```
# fdisk -l | grep /dev
Disk /dev/xvda: 12.9 GB, 12884901888 bytes, 25165824 sectors
/dev/xvda1      *    2048     1026047     512000    83    Linux
/dev/xvda2        1026048   25165823   12069888    8e    Linux LVM
Disk /dev/xvdb: 10.7 GB, 10737418240 bytes, 20971520 sectors
/dev/xvdb1        2048    2099199    1048576    83    Linux
Disk /dev/xvdd: 10.7 GB, 10737418240 bytes, 20971520 sectors
Disk /dev/mapper/ol-root: 11.1 GB, 11068768256 bytes, ...
Disk /dev/mapper/ol-swap: 1287 MB, 1287651328 bytes, ...
Disk /dev/sda: 10.7 GB, 10737418240 bytes, 20971520 sectors
Disk /dev/sdb: 10.7 GB, 10737418240 bytes, 20971520 sectors
```

- f. Use the `iscsiadm` command to view active sessions with `<printlevel> 3`.

- Use the `<printlevel>` option to view additional detail on the session.

```
# iscsiadm -m session -P 3
iSCSI Transport Class version 2.0-870
Target: iqn.2015-07.com.example.host02:tgt1 (non-flash)
        Current Portal: 192.0.2.102:3260,1
        Persistent Portal: 192.0.2.102:3260,1
*****
        Interface:
*****
        Iface Name: default
        Iface Transport: tcp
        Iface Initiatorname: iqn.2015-07.com.example...
        Iface IPAddress: 192.0.2.104
        Iface HWAddress: <empty>
        Iface Netdev: <empty>
        SID: 1
        iSCSI Connection State: LOGGED IN
        iSCSI Session State: LOGGED_IN
        Internal iscsid Session State: NO CHANGE
*****
        Timeouts:
*****
        Recovery Timeout: 120
        TARGET Reset Timeout: 30
        LUN Reset Timeout: 30
        Abort Timeout: 15
*****
```

```

CHAP:
*****
username: <empty>
password: *****
username_in: <empty>
password_in: *****
*****
Negotiated iSCSI params:
*****
HeaderDigest: None
DataDigest: None
MaxRecvDataSegmentLength: 262144
MaxXmitDataSegmentLength: 262144
FirstBurstLength: 65536
MaxBurstLength: 262144
ImmediateData: Yes
InitialR2T: Yes
MaxOutstandingR2T: 1
*****
Attached SCSI devices:
*****
Host Number: 2 State: running
scsi12 Channel 00 Id 0 Lun:1
Attached scsi disk sda State: running
scsi12 Channel 00 Id 0 Lun:2
Attached scsi disk sdb State: running

```

4. Verify the usability of the iSCSI device.

- Use the `fdisk` command to create a partition on `/dev/sda` with the following parameters:
  - Primary partition
  - Partition number 1
  - First sector 8192
  - Last sector (size) +1G

```

# fdisk /dev/sda
Welcome to fdisk (util-linux 2.23.2).

Changes will remain in memory only, until you decide to write them. Be careful before using the write command.

Command (m for help): n
Partition type:
      p    primary (0 primary, 0 extended, 4 free)

```

```

      e   extended
Select (default p): ENTER
Using default response: p
Partition number (1-4, default 1): ENTER
First sector (8192-20971519, default 8192): ENTER
Using default value 8192
Last sector, +sectors or +size{K,M,G} (8192-20971519, default
20971519): +1G
Partition 1 of type Linux and of size 1 GiB is set

Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.

```

- b. Use the `mkfs` command to create an ext4 file system on `/dev/sda1`.

```

# mkfs -t ext4 /dev/sda1
...
Writing superblocks and filesystem accounting information: done

```

- c. Use the `mkdir` and `mount` (with the `_netdev` option) commands to mount the file system on `/iscsi_dev`.

- Because it is a network device, be sure to use the `_netdev` option when mounting.

```

# mkdir /iscsi_dev
# mount /dev/sda1 -o _netdev /iscsi_dev

```

- d. Use the `mount` command to display mounted file systems.

```

# mount
...
/dev/sda1 on /iscsi_dev type ext4 (rw,relatime,...,_netdev)

```

- e. Use the `cp` command to copy `/boot/vmlinuz*` files to `/iscsi_dev`. Use the `ls` command to display the contents of `/iscsi_dev`.

```

# cp /boot/vmlinuz* /iscsi_dev
# ls /iscsi_dev
lost+found
vmlinuz-0-rescue...
vmlinuz-3.10.0-229.el7.x86_64
vmlinuz-3.8.13-55.1.6.el7uek.x86_64

```

5. Remove the `/dev/sda1` partition in preparation for the next practice.

- a. Use the `umount` command to unmount `/dev/sda1`.

```
# umount /dev/sda1
```

- b. Use the `fdisk` command to remove the `/dev/sda1` partition.

```
# fdisk /dev/sda
Welcome to fdisk (util-linux 2.23.2).

Changes will remain in memory only, until you decide to write
them. Be careful before using the write command.

Command (m for help): d
Selected partition 1
Partition 1 is deleted

Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks
```

- c. Use the `cat` command to view the `/proc/partitions` file and ensure that the `sda1` partition no longer exists.
- Note that the `sda` device exists but the `sda1` partition is gone.

```
# cat /proc/partitions
major minor # blocks name
 202        0   12582912  xvda
...
    8        0   10485760   sda
    8       16   10485760   sdb
```

## Practice 12-3: Configuring iSCSI Multipathing

---

### Overview

In this practice, you:

- Perform all steps from **host03**
- Install the `device-mapper-multipath` packages
- Configure DM-Multipath for the iSCSI block device
- Verify failover to redundant path is successful
- Remove DM-Multipath configuration
- Remove iSCSI initiator configuration

### Assumptions

You are the `root` user on the **host03** VM.

### Tasks

1. Install and explore the DM-Multipath package.
  - a. Use the `yum` command to install the `device-mapper-multipath` package on **host03**.
    - In this example, the package is already installed.

```
# yum install device-mapper-multipath
...
Package device-mapper-multipath-0.4.9-77.el7.x86_64 already
installed and latest version
Nothing to do
```
  - b. Use the `rpm -ql` command to view the files installed with the `device-mapper-multipath` package.
 

```
# rpm -ql device-mapper-multipath
/etc/multipath
/usr/lib/systemd/system/multipathd.service
/usr/lib/udev/rules.d/11-dm-mpath.rules
/usr/lib/udev/rules.d/62-multipath.rules
/usr/sbin/mpathconf
/usr/sbin/mpathpersist
/usr/sbin/multipath
/usr/sbin/multipathd
/usr/share/doc/device-mapper-multipath-<version>
...
/usr/share/man/man5/multipath.conf.5.gz
/usr/share/man/man8/mpathconf.8.gz
```

```
/usr/share/man/man8/mpathpersist.8.gz
/usr/share/man/man8/multipath.8.gz
/usr/share/man/man8/multipathd.8.gz
```

- c. Use the `ls` command to view the `/usr/share/doc/device-mapper-multipath-<version>` directory.
- The `multipath.conf` file is a basic configuration file that is used as a starting point.
  - You can create the `/etc/multipath.conf` file manually. Otherwise the `/usr/share/.../multipath.conf` file is copied into the `/etc` directory when you enable multipathing.

```
# ls /usr/share/doc/device-mapper-multipath-0.4.9
AUTHOR      COPYING      FAQ      multipath.conf
```

## 2. Enable DM-Multipath.

- a. Use the `mpathconf --help` command to view the usage.

```
# mpathconf --help
Usage: /usr/sbin/mpathconf <command>

Commands:
Enable: --enable
Disable: --disable
Set user_friendly_names (Default y): --user_friendly_names <y|n>
Set find_multipaths (Default y): --find_multipaths <y|n>
Load the dm-multipath modules on enable (Default y): ...
Start/stop/reload multipathd (Default n): --with_multipathd ...
```

- b. Use the `mpathconf` command to enable DM-Multipath.

- Include the `--find_multipaths n` option to override the default of `y`.

```
# mpathconf --enable --find_multipaths n
```

- c. Use the `mpathconf` command without any options to display the status.

```
# mpathconf
multipath is enabled
find_multipaths is disabled
user_friendly_names is enabled
dm_multipath module is loaded
multipathd is not running
```

- d. Use the `lsmod` command to view the `dm_multipath` module.

```
# lsmod | grep dm_multipath
dm_multipath    17280   0
dm_mod          81615   9  dm_multipath,dm_log,dm_mirror
```

- e. Use the `systemctl` command to view the status of the `multipathd` service.

- The service is enabled to start at boot time but is currently inactive.

```
# systemctl status multipathd
multipathd.service - Device-Mapper Multipath Device Controller
    Loaded: loaded (/usr/lib/systemd/system/multipathd...enabled)
    Active: inactive (dead)
    ...
...
```

3. Use the less command to view the /etc/multipath.conf file.

- Press q to exit the less command when you have viewed the file.

```
# less /etc/multipath.conf
...
...
```

- The default file contains all comments with the following exceptions:

```
defaults {
    user_friendly_names yes
    find_multipaths no
}
blacklist {
}
```

- Make no changes to the file at this time.

4. Start the DM-Multipath daemon and view the multipathed devices.

- a. Use the ls -l command to view a long listing of the /dev/mapper directory.

```
# ls -l /dev/mapper
crw-rw---- ... control
lrwxrwxrwx ... ol_root -> ../dm-0
lrwxrwxrwx ... ol_swap -> ../dm-1
```

- b. Use the systemctl command to start the multipathd service.

```
# systemctl start multipathd
```

- c. Use the ls -l command to view a long listing of the /dev/mapper directory. (Only a partial output is shown.)

- Notice the new mpatha and mpathb files in /dev/mapper.

```
# ls -l /dev/mapper
crw-rw---- ... control
lrwxrwxrwx ... mpatha -> ../dm-2
lrwxrwxrwx ... mpathb -> ../dm-3
lrwxrwxrwx ... ol_root -> ../dm-0
lrwxrwxrwx ... ol_swap -> ../dm-1
```

- d. Use the fdisk command to display the devices in /dev.

- Notice that the /dev/mapper/mpatha device now exists.

- Notice that the `/dev/mapper/mpathb` device now exists.

```
# fdisk -l | grep /dev
...
Disk /dev/mapper/mpatha: 10.7 GB, 10737418240 bytes, ... sectors
Disk /dev/mapper/mpathb: 10.7 GB, 10737418240 bytes, ... sectors
```

5. View the multipath topology and the WWIDs of the devices.

- a. Use the `multipath -ll` command to list the maximum multipath topology.

- In the following example, `36001405346939038cc9480caf0dd9a9d` is the WWID of the underlying `/dev/sdb` device.
- In the following example, the `36001405a7c28190541f4d61880050090` is the WWID of the underlying `/dev/sda` device.
- These WWIDs might be different on your system.

```
# multipath -ll
...
mpathb(36001405346939038cc9480caf0dd9a9d) dm-3 LIO-ORG ,IBLOCK
size=10G features='0' hwhandler='0' wp=rw
`-- policy='service-time 0' prio=1 status=active
   `-- 2:0:0:2 sdb 8:16      active ready running
mpatha(36001405a7c28190541f4d61880050090) dm-2 LIO-ORG ,IBLOCK
size=10G features='0' hwhandler='0' wp=rw
`-- policy='service-time 0' prio=1 status=active
   `-- 2:0:0:1 sda 8:0      active ready running
```

- b. Use the `/usr/lib/udev/scsi_id` command to view the WWID of the `/dev/sda` device.

- Use the absolute pathname of the command because the `/usr/lib/udev` directory is not in your `PATH` variable.
- The `--whitelisted` option must be included to generate output.
- Note that the WWID is the same as shown in step 5a for `/dev/sda`.

```
# /usr/lib/udev/scsi_id --whitelisted --replace-whitespace --
device=/dev/sda
36001405a7c28190541f4d61880050090
```

- c. Use the `/usr/lib/udev/scsi_id` command to view the WWID of the `/dev/sdb` device.

- Use the absolute pathname of the command because the `/usr/lib/udev` directory is not in your `PATH` variable.
- The `--whitelisted` option must be included to generate output.
- Note that the WWID is the same as shown in step 5a for `/dev/sdb`.

```
# /usr/lib/udev/scsi_id --whitelisted --replace-whitespace --
device=/dev/sdb
36001405346939038cc9480caf0dd9a9d
```

6. Verify the usability of the multipathed iSCSI device.

- a. Use the `mkfs` command to create an ext4 file system on `/dev/mapper/mpatha`.

```
# mkfs -t ext4 /dev/mapper/mpatha
...
Writing superblocks and filesystem accounting information:done
```

- b. Use the `mount` command (with the `_netdev,errors=continue` option) to mount the file system on `/iscsi_dev`.

- Because it is a network device, be sure to use the `_netdev` option when mounting it.
- Open-iSCSI can fail both paths temporarily while failing over, resulting in the system remounting a file system in read-only mode on error unless you include an `errors=continue` option as a `mount` option.

```
# mount /dev/mapper/mpatha -o _netdev,errors=continue /iscsi_dev
```

- c. Use the `cp` command to copy `/boot/vmlinuz*` files to `/iscsi_dev`. Use the `ls` command to display the contents of `/iscsi_dev`.

```
# cp /boot/vmlinuz* /iscsi_dev
# ls /iscsi_dev
lost+found
vmlinuz-0-rescue...
vmlinuz-3.10.0-229.el7.x86_64
vmlinuz-3.8.13-55.1.6.el7uek.x86_64
```

## 7. Verify DM-Multipath failover.

- a. Use the `iscsiadm -m session -P 3` command to determine the IP address of the initiator for the active session.

- Pipe the output to `grep` and search for the “`IPaddress`” string.
- In this example, the IP address is `192.0.2.104`.
- This is the IP address for the `eth1` interface on **host03**.

```
# iscsiadm -m session -P 3 | grep IPaddress
Iface IPaddress: 192.0.2.104
```

- b. Use the `ifdown` command to bring down the `eth1` interface.

```
# ifdown eth1
Device 'eth1' successfully disconnected.
```

- c. Use the `cp` command to copy `/boot/initramfs*` files to `/iscsi_dev`.

- Use the `ls` command to display the contents of `/iscsi_dev`.

```
# cp /boot/initramfs* /iscsi_dev
# ls /iscsi_dev
initramfs-0-rescue...img
initramfs-3.10.0-229.el7.x86_64.img
initramfs-3.8.13-55.1.6.el7uek.x86_64.img
lost+found
vmlinuz-0-rescue...
vmlinuz-3.10.0-229.el7.x86_64
```

```
vmlinuz-3.8.13-55.1.6.el7uek.x86_64
```

- d. Use the `rm` command to remove the `vmlinuz*` files on `/iscsi_dev`.
- Answer `y` to remove each `vmlinuz*` file.

```
# rm /iscsi_dev/vmlinuz*
rm: remove regular file '/iscsi_dev/vmlinuz...'? y
...
```

- e. Use the `tail` command to view the `/var/log/messages` file.
- Sample output is shown. Your output might be different.

```
# tail /var/log/messages
...
<date> host03 iscsid: Kernel reported iSCSI connection 1:0 error
<date> host03 iscsid: connection1:0 is operational after
recovery (1 attempts)
```

- f. Use the `iscsiadm -m session -P 3` command to determine the IP address of the initiator for the active session.
- Pipe the output to `grep` and search for the “`IPaddress`” string.
  - In this example, the IP address is now `192.0.2.103`.
  - This is the IP address for the `eth0` interface on **host03**.
  - This confirms the failover from `eth1` to `eth0` was successful.

```
# iscsiadm -m session -P 3 | grep IPaddress
Iface IPaddress: 192.0.2.103
```

- g. Use the `ifup` command to bring up the `eth1` interface.

```
# ifup eth1
Connection successfully activated ...
```

## 8. Remove DM-Multipath.

- a. Use `umount` to unmount `/iscsi_dev`.

```
# umount /iscsi_dev
```

- b. Use the `systemctl` command to stop and disable the `multipathd` service.

```
# systemctl stop multipathd
# systemctl disable multipathd
rm '/etc/systemd/system/sysinit.target.wants/multipathd.service'
```

## 9. Remove iSCSI initiator and reboot **host03**.

- a. Use the `iscsiadm` command to log out from all sessions.

```
# iscsiadm -m node -u
Logging out of session [sid: 1, target: iqn.2015-07.com.example.host02:tgt1, portal: 192.0.2.102:3260]
Logout of [sid: 1, target: iqn.2015-07.com.example.host02:tgt1, portal: 192.0.2.102:3260] successful.
```

- b. Use the `systemctl` command to stop and disable the `iscsid` service.

```
# systemctl stop iscsid
Warning: Stopping iscsid, but it can still be active by:
  iscsid.socket
# systemctl disable iscsid
rm '/etc/systemd/system/sysinit.target.wants/iscsid.service'
```

- c. Use the `/bin/rm -r` command to delete the files in the `nodes` directory in the iSCSI database.

```
# /bin/rm -r /var/lib/iscsi/nodes/*
```

- d. Use the `/bin/rm -r` command to delete the files in the `send_targets` directory in the iSCSI database.

```
# /bin/rm -r /var/lib/iscsi/send_targets/*
```

- e. Use the `systemctl reboot` command to reboot **host03**.

```
# systemctl reboot
...
Connection to host03 closed.
```

10. Log off **host02** VM.

Use the `exit` command to log off **host02**.

```
# exit
Logout
Connection to host02 closed.
```



## **Practices for Lesson 13: Control Groups (Cgroups)**

**Chapter 13**

## Practices for Lesson 13: Overview

---

### Practices Overview

In these practices, you perform the following:

- Explore cgroup integration into `systemd`.
- Explore cgroup hierarchies and cgroup subsystem parameters.
- Control access to system resources.

## Practice 13-1: Exploring cgroup Integration Into `systemd`

---

### Overview

In this practice, you do the following:

- Explore the `/sys/fs/cgroup/systemd` directory.
- Explore `systemd` slice units.
- Explore `systemd` scope units.
- Use the `systemd-cgls` command to view the slice and cgroup hierarchy.
- Explore `systemd` service units.

### Assumptions

- You are the root user on **dom0**.

### Tasks

1. From **dom0**, use the `ssh` command to log in to **host03**.

- The root password is **oracle**.

```
[dom0]# ssh host03
root@host03's password: oracle
Last login...
[root@host03 ~]#
```

2. Explore the `/sys/fs/cgroup/systemd` directory.

- The `/sys/fs/cgroup/systemd` directory contains a hierarchy of slice, scope, and service units for the cgroup tree.
- a. Use the `cd` command to change to the `/sys/fs/cgroup/systemd` directory.
  - Use the `ls -l` command to view the contents of the `/sys/fs/cgroup/systemd` directory.
  - Note that there are two subdirectories: `system.slice` and `user.slice`.

```
# cd /sys/fs/cgroup/systemd
# ls -l
-rw-r--r-- ... cgroup.clone_children
--w--w--w- ... cgroup.event_control
-rw-r--r-- ... cgroup.procs
-rw-r--r-- ... notify_on_release
-rw-r--r-- ... release_agent
drwxr-xr-x ... system.slice
-rw-r--r-- ... tasks
drwxr-xr-x ... user.slice
```

- b. Use the `ls -l` command to view the contents of the `/sys/fs/cgroup/systemd/system.slice` directory.
  - The `system.slice` directory contains services and other system processes.
  - Note that there are subdirectories for different `systemd` unit types such as `service`, `socket`, `mount`, and `swap`.

```
# ls -l system.slice
drwxr-xr-x ... abrt-ccpp.service
drwxr-xr-x ... abrtd.service
drwxr-xr-x ... abrt-oops.service
...
drwxr-xr-x ... avahi-daemon.socket
drwxr-xr-x ... boot.mount
...
drwxr-xr-x ... dev-disk-by\...swap
drwxr-xr-x ... dev-dm\x2d1.swap
drwxr-xr-x ... dev-hugepages.mount
...
drwxr-xr-x ... tuned.service
drwxr-xr-x ... upower.service
drwxr-xr-x ... vmtoolsd.service
```

- c. Use the `ls -l` command to view the contents of the `/sys/fs/cgroup/systemd/user.slice` directory.
- The `user.slice` directory is the top-level directory for all user sessions.
  - Note that in this example, subdirectories exist for processes owned by UID 0 (root), UID 42 (gdm), and UID 1000 (oracle user).

```
# ls -l user.slice
-rw-r--r-- ... cgroup.clone_children
--w--w--w- ... cgroup.event_control
-rw-r--r-- ... cgroup.procs
-rw-r--r-- ... notify_on_release
-rw-r--r-- ... tasks
drwxr-xr-x ... user-0.slice
drwxr-xr-x ... user-1000.slice
drwxr-xr-x ... user-42.slice
```

- d. Use the `ls -l` command to view the contents of the `/sys/fs/cgroup/systemd/user.slice/user-1000.slice` directory.
- The `/sys/fs/cgroup/systemd/user.slice` hierarchy contains user processes that run with transient cgroups called scopes.
  - Note the `session-337.scope` subdirectory in this example. The scope directory name might be different on your system.

```
# ls -l user.slice/user-1000.slice
-rw-r--r-- ... cgroup.clone_children
--w--w--w- ... cgroup.event_control
-rw-r--r-- ... cgroup.procs
-rw-r--r-- ... notify_on_release
drwxr-xr-x ... session-337.scope
-rw-r--r-- ... tasks
```

- e. Use the `ls -l` command to view the contents of the `/sys/fs/cgroup/systemd/user.slice/user-1000.slice/session-337.scope` directory.

```
# ls -l user.slice/user-1000.slice/session-337.scope
-rw-r--r-- ... cgroup.clone_children
--w--w--w- ... cgroup.event_control
-rw-r--r-- ... cgroup.procs
-rw-r--r-- ... notify_on_release
-rw-r--r-- ... tasks
```

3. Explore `systemd` slice units.

- Slice units group units that manage system processes, such as service units and scope units, in a hierarchical tree for resource management purposes.
- Slice unit file names have a `.slice` extension. See the `systemd.slice(5)` man page for details.
- a. Use the `systemctl -t slice -a` command to list all loaded slice units on your system.
  - Sample output is shown.
  - Slices do not contain processes, they organize a hierarchy in which scopes and services are placed.
  - The actual processes are contained in scopes or in services.

```
# systemctl -t slice -a
UNIT           LOAD   ACTIVE SUB     DESCRIPTION
-.slice         loaded  active  active  Root Slice
system-getty.slice  loaded  active  active  system-getty.slice
system-lvm2\x2dpvscan.slice  loaded  inactive  dead ...
system-systemd\x2dfsck.slice  loaded  inactive  dead ...
system.slice    loaded  active  active  System Slice
user-0.slice    loaded  active  active  user-0.slice
user-1000.slice  loaded  active  active  user-1000.slice
user-42.slice   loaded  active  active  user-42.slice
user.slice      loaded  active  active  User and Session Slice
...
```

- b. Use the `find` command to list the location and file name of slice units.

- The root slice is denoted as `-.slice`.

```
# find / -name "*.slice"
...
/sys/fs/cgroup/systemd/user.slice
/sys/fs/cgroup/systemd/user.slice/user-1000.slice
/sys/fs/cgroup/systemd/user.slice/user-0.slice
/sys/fs/cgroup/systemd/user.slice/user-42.slice
/sys/fs/cgroup/systemd/system.slice
/sys/fs/cgroup/systemd/system.slice/system-lvm2\x2dpvscan.slice
/sys/fs/cgroup/systemd/system.slice/system-systemd\x2dfsck.slice
```

```
/sys/fs/cgroup/systemd/system.slice/system-getty.slice
/usr/lib/systemd/system/.slice
/usr/lib/systemd/system/user.slice
/usr/lib/systemd/system/system.slice
/usr/lib/systemd/system/machine.slice
```

4. Explore `systemd` scope units.

- Scope units are similar to service units but manage foreign processes instead of starting them as well.
- Scope unit file names have a `.scope` extension. See the `systemd.scope(5)` man page for details.
- a. Use the `systemctl -t scope -a` command to list all loaded scope units on your system.
  - Note the `session-337.scope` unit for the `oracle` user in this example. The scope unit name might be different on your system.

```
# systemctl -t scope -a
UNIT           LOAD   ACTIVE SUB     DESCRIPTION
session-337.scope    loaded  active running Session ... oracle
...
```

- b. Use the `find` command to list the location and file name of scope units.

```
# find / -name "*.scope"
...
/run/systemd/system/session-337.scope
/sys/fs/cgroup/systemd/user.slice/user-1000.slice/session-
337.scope
```

5. Use the `systemd-cgls` command to view the slice and cgroup hierarchy.

- `system.slice` contains services and other system processes.
- `user.slice` contains user processes that run with transient cgroups called scopes.
- Note that services and scopes contain processes and are placed in slices that do not contain processes.
- In this example, processes for the `oracle` user are running in the `session-337.scope` scope under the `/user.slice/user-1000.slice` slice.
  - The `oracle` user has a UID of 1000.
- Also note that the `-.slice` is not shown as it is implicitly identified with the root of the entire tree.

```
# systemd-cgls
Working Directory /sys/fs/cgroup/systemd:
|-1 /usr/lib/systemd/system --switched-root --system ...
|`-user.slice
|  |-user-1000.slice
|    |-session-337.scope
|      |-16799 gdm-session-worker [pam/gdm-password]
|      |-16821 /usr/bin/gnome-keyring-daemon --daemonize -login
```

```

| |     |-16839 gnome-session --session gnome-classic
...
|-system.slice
  |-udisks2.service
    |-17075 /usr/lib/udisks2/udisksd --no-debug
  |-bluetooth.service
    |-17057 /usr/lib/bluetoothd -n
  |-polkit.service
    |-16806 /usr/lib/polkit-1/polkitd --no-debug
...
  |-auditd.service
    |-514 /sbin/auditd -n
    |-533 /sbin/audispd
    |-537 /usr/sbin/seddispatch
...
  |-sshd.service
    |-1237 /usr/sbin/sshd -D
...
  |-abrt.service
    |-541 /usr/sbin/abrt -d -s
...

```

6. Explore `systemd` service units.

- Use the `ls -l` command to view the contents of the `/sys/fs/cgroup/systemd/system.slice/abrt.service` directory.
  - Note the `tasks` file. Each “service” directory has a `tasks` file.
  - The `tasks` file records the process ID (PID) of the process associated with the service, and in turn, with the cgroup and the associated subsystem parameter settings.

```
# ls -l system.slice/abrt.service
-rw-r--r-- ... cgroup.clone_children
--w--w--w- ... cgroup.event_control
-rw-r--r-- ... cgroup.procs
-rw-r--r-- ... notify_on_release
-rw-r--r-- ... tasks
```

- Use the `cat` command to view the contents of the `/sys/fs/cgroup/systemd/system.slice/abrt.service/tasks` file.
  - Sample output of 541 is shown.
- Use the `ps -ef` command to show all processes, pipe the output to `grep`, and search for the string 541.
  - Substitute 541 with whatever number was returned from step 5b.

- Note that 541 in this example is the PID of the `abrt` service.
- Also note that the parent PID (PPID) is 1, which is the PID of `systemd`.

```
# ps -ef | grep 541
root 541 1 ... /usr/sbin/abrt -d -s
...
```

- d. Use the `systemctl` command to view the status of the `abrt` service.

- Note the “CGroup” information in the output of this command.

```
# systemctl status abrt
abrt.service - ABRT Automated Bug Reporting Tool
  Loaded: loaded (/usr/lib/systemd/system/abrt.service; ...)
  Active: active (running) since ...
    Main PID: 541 (abrt)
      CGroup: /system.slice/abrt.service
              |-541 /usr/sbin/abrt -d -s
...

```

- e. Use the `cat` command to view the contents of the `/sys/fs/cgroup/systemd/system.slice/sshd.service/tasks` file.

- Sample output of 1237 is shown.

```
# cat system.slice/sshd.service/tasks
1237
```

- f. Use the `ps -ef` command to show all processes, pipe the output to `grep`, and search for the string 1237.

- Substitute 1237 with whatever number was returned from step 5e.
- Note that 1237 in this example is the PID of the `sshd` service.
- Also note that the PPID is 1, which is the PID of `systemd`.

```
# ps -ef | grep 1237
root 1237 1 ... /usr/sbin/sshd -D
...
```

- g. Use the `systemctl` command to view the status of the `sshd` service.

- Note the “CGroup” information in the output of this command.

```
# systemctl status sshd
...
Main PID: 1237 (sshd)
  CGroup: /system.slice/sshd.service
          |-1237 /usr/sbin/sshd -D
...

```

- h. Use the `cat` command to view the contents of the `/sys/fs/cgroup/systemd/system.slice/crond.service/tasks` file.

- Sample output of 1248 is shown.

```
# cat system.slice/crond.service/tasks  
1248
```

- i. Use the `ps -ef` command to show all processes, pipe the output to `grep`, and search for the string 1248.

- Substitute 1248 with whatever number was returned from step 5h.
- Note that 1248 in this example is the PID of the `crond` service.
- Also note that the PPID is 1, which is the PID of `systemd`.

```
# ps -ef | grep 1248  
root      1248      1  ... /usr/sbin/crond -n  
...
```

- j. Use the `systemctl` command to view the status of the `crond` service.

- Note the “CGroup” information in the output of this command.

```
# systemctl status crond  
...  
Main PID: 1248 (crond)  
CGroup: /system.slice/crond.service  
| -1248 /usr/sbin/crond -n  
...
```

## Practice 13-2: Exploring cgroup Hierarchies and cgroup Subsystem Parameters

---

### Overview

In this practice, you do the following:

- View the mounted cgroup subsystems (resource controllers).
- View the cgroup subsystem parameters.
- Install the `kernel-uek-doc` package.
- View cgroup subsystem documentation.

### Assumptions

You are the root user on **host03**.

### Tasks

1. View the mounted cgroup subsystems (resource controllers).
  - a. Use the `cat` command to view the `/proc/cgroups` file.

```
# cat /proc/cgroups
#subsys_name    hierarchy    num_cgroups    enabled
cpuset          2            1              1
cpu             3            1              1
cpuacct         3            1              1
memory          4            1              1
devices         5            1              1
freezer         6            1              1
net_cls          7            1              1
blkio           8            1              1
perf_event       9            1              1
hugetlb         10           1              1
```

- b. Use the `ls -l` command to view a long listing of the `/sys/fs/cgroup` directory.

```
# ls -l /sys/fs/cgroup
drwxr-xr-x. ... blkio
lrwxrwxrwx. ... cpu -> cpu,cpuacct
lrwxrwxrwx. ... cpuacct -> cpu,cpuacct
drwxr-xr-x. ... cpu,cpuacct
drwxr-xr-x. ... cpuset
drwxr-xr-x. ... devices
drwxr-xr-x. ... freezer
drwxr-xr-x. ... hugetlb
drwxr-xr-x. ... memory
drwxr-xr-x. ... net_cls
```

```
drwxr-xr-x. ... perf_event
drwxr-xr-x. ... systemd
```

- c. Use the `mount` command, pipe the output to `grep`, and search for the “cgroup” string to view the mounted resource controllers.

```
# mount | grep cgroup
tmpfs on /sys/fs/cgroup type tmpfs (rw,nosuid,nodev,noexec,...)
cgroup on /sys/fs/cgroup/systemd type cgroup (rw,nosuid,nodev...)
cgroup on /sys/fs/cgroup/cpuset type cgroup (rw,nosuid,nodev...)
cgroup on /sys/fs/cgroup/cpu,cpuacct type cgroup (rw,nosuid...)
cgroup on /sys/fs/cgroup/memory type cgroup (rw,nosuid,nodev...)
cgroup on /sys/fs/cgroup/devices type cgroup (rw,nosuid,nodev...)
cgroup on /sys/fs/cgroup/freezer type cgroup (rw,nosuid,nodev...)
cgroup on /sys/fs/cgroup/net_cls type cgroup (rw,nosuid,nodev...)
cgroup on /sys/fs/cgroup/blkio type cgroup (rw,nosuid,nodev...)
cgroup on /sys/fs/cgroup/perf_event type cgroup (rw,nosuid...)
cgroup on /sys/fs/cgroup/hugetlb type cgroup (rw,nosuid,nodev...)
```

2. View the cgroup subsystem parameters.

- a. Use the `yum` command to install the `kernel-uek-doc` package.
- This package contains the cgroup subsystem documentation.
  - Answer `y` to “Is this ok”.

```
# yum install kernel-uek-doc
...
Is this ok [y/d/N] : y
...
Complete!
```

- b. Use the `cd` command to change to the `/usr/share/doc/kernel-doc-<version>/Documentation/cgroups` directory.
- Use the `ls` command to view the contents of the directory.

```
# cd /usr/share/doc/kernel-doc-3.8.13/Documentation/cgroups
# ls
00-INDEX cpusets.txt memory.txt
blkio-controller.txt devices.txt net_prio.txt
cgroup_event_listener.c freezer-subsystem.txt resource...
cgroups.txt hugetlb.txt
cpuacct.txt memcg_test.txt
```

- c. Use the `cat` command to view the `00-INDEX` file.

- This file provides a short description of all files in this directory.

```
# cat 00-INDEX
00-INDEX
    - this file
blkio-controller.txt
```

```

        - Description for Block IO Controller, implementation...
cgroups.txt
        - Control Groups definition, implementation details...
cgroup_event_listener.c
        - A user program for cgroup listener.
cpuacct.txt
        - CPU Accounting Controller; account CPU usage for ...
cpusets.txt
        - documents the cpusets feature; assign CPUs and Mem ...
devices.txt
        - Device Whitelist Controller; description, interface...
...

```

- d. Use the `ls` command to view the subsystem parameters for the `blkio` resource controller.

- View the `/sys/fs/cgroup/blkio` directory to see the `blkio` parameters.
- Only a partial listing is shown.

```
# ls /sys/fs/cgroup/blkio
blkio.io_merged    blkio.throttle_io_service_bytes ...
blkio.io_queued   blkio.throttle.io_serviced      ...
...
```

- e. Use the `less` command to view the `blkio-controller.txt` file.

- This file contains a description of the `blkio` subsystem.
- Scroll down to see the `blkio` subsystem parameter descriptions.
- The example command assumes you are in the `/usr/share/doc/kernel-doc-3.8.13/Documentation/cgroups` directory.

```
# less blkio-controller.txt
                                Block IO Controller
=====
Overview
=====
cgroup subsys "blkio" implements the block io controller...
...
- blkio.weight
        - Specifies per cgroup weight. This is default weight...
...
- blkio.weight_device
        - One can specify per cgroup per device rules using ...
...
- blkio.time
        - disk time allocated to cgroup per device in milli...
...
```

```

- blkio.sectors
  - number of sectors transferred to/from disk by the ...
  ...
- blkio.io_service_bypthes
  - Number of bytes transferred to/from the disk by ...
  ...

```

- f. Use the `ls` command to view the subsystem parameters for the `cpuset` resource controller.

- View `/sys/fs/cgroup/cpuset` directory to see the `cpuset` parameters.
- Only a partial listing is shown.

```
# ls /sys/fs/cgroup/cpuset
...
cpuset.cpu_exclusive      cpuset.mems
cpuset.cpus                cpuset.sched_load_balance
.....
```

- g. Use the `less` command to view the `cpusets.txt` file.

- This file contains a description of the `cpuset` subsystem.
- Scroll down to see the `cpuset` subsystem parameter descriptions.
- The example command assumes you are in the `/usr/share/doc/kernel-doc-3.8.13/Documentation/cgroups` directory.

```
# less cpusets.txt
CPUSETS
=====
...
- cpuset.cpus: list of CPUS in that cpuset
- cpuset.mems: list of Memory Nodes in that cpuset
- cpuset.memory_migrate flag: if set, move pages to cpuset...
- cpuset.cpu_exclusive flag: is cpu placement exclusive?
- cpuset.mem_exclusive flag: is memory placement exclusive?
- cpuset.mem_hardwall flag: is memory allocation hardwalled?
...
```

- h. Use the `ls` command to view the subsystem parameters for the `memory` resource controller.

- View the `/sys/fs/cgroup/memory` directory to see the `memory` parameters.
- Only a partial listing is shown.

```
# ls /sys/fs/cgroup/memory
...
memory.limit_in_bytes    memory.usage_in_bytes ...
.....
```

- i. Use the `less` command to view the `memory.txt` file.

- This file contains a description of the `memory` subsystem.

- Scroll down to see the memory subsystem parameter descriptions.
- The example command assumes you are in the /usr/share/doc/kernel-doc-3.8.13/Documentation/cgroups directory.

```
# less memory.txt
Memory Resource Controller

NOTE: The Memory Resource Controller has generically been ...
...
memory.limit_in_bytes      # set/show limit of memory usage
memory.memsw.limit_in_bytes # set/show limit of memory+Swap...
memory.failcnt              # show the number of memory ...
...
```

- j. Continue to view the cgroup subsystem parameters and descriptions as you wish.
- The <subsystem> parameters are stored in the /sys/fs/cgroup/<subsystem> directory.
  - The descriptions of the <subsystem> are stored in the /usr/share/doc/kernel-doc-3.8.13/Documentation/cgroups/<subsystem> directory.

## Practice 13-3: Controlling Access to System Resources

---

### Overview

In this practice, you do the following:

- View the properties of a `systemd` service unit.
- Use the `systemctl set-property` command to change the value of subsystem parameters for a service.
- Define resource control settings for a service in a unit configuration file.

### Assumptions

You are the root user on **host03**.

### Tasks

1. Use the `systemctl show` command to view the properties of a `systemd` service unit.
  - Included in the output are resource control settings.
  - See the `systemd.resource-control(5)` man page for a description resource control unit settings.
  - a. View the properties of the `sshd` service unit.

```
# systemctl show sshd
Id=sshd.service
Names=sshd.service
...
Slice=system.slice
ControlGroup=/system.slice/sshd.service
...
CPUAccounting=no
CPUShares=1024
BlockIOAccounting=no
BlockIOWeight=1000
MemoryAccounting=no
...
```

- b. View the properties of the `sshd` service unit, pipe the output to `grep -i`, and search for the “mem” string.

```
# systemctl show sshd | grep -i mem
LimitMEMLOCK=65536
MemoryAccounting=no
MemoryLimit=18446744073709551615
```

- c. View the properties of the `sshd` service unit, pipe the output to `grep -i`, and search for the “blockio” string.

```
# systemctl show sshd | grep -i blockio
BlockIOAccounting=no
BlockIOWeight=1000
```

- d. Use the `--property=CPUShares` option to view value of the `CPUShares` property only for the `sshd` service.

```
# systemctl show --property=CPUShares sshd
CPUShares=1024
```

2. Use the `systemctl set-property` command to change the value of a `cpu`, `cpuacct` subsystem parameter for the `sshd` service.

- a. Before setting a `cpu`, `cpuacct` subsystem parameter, use the `ls` command to view the contents of the `/sys/fs/cgroup/cpu,cpuacct/system.slice/sshd.service` directory.
- You view this directory again after setting the `cpu`, `cpuacct` subsystem parameter.

```
# ls /sys/fs/cgroup/cpu,cpuacct/system.slice/sshd.service
ls: cannot access
/sys/fs/cgroup/cpu,cpuacct/system.slice/sshd.service: No such
file or directory
```

- Note that before setting any `cpu`, `cpuacct` subsystem parameters, the directory does not exist.
- b. Change the `CPUShares` property from the default of 1024 to 512 and change the `CPUAccounting` property to `true` for the `sshd` service.

- This halves the access the `sshd` service has to CPU time.

```
# systemctl set-property sshd.service CPUShares=512
CPUAccounting=true
```

- c. Run the `ls` command in step 2a again.
- Note that the directory now exists and contains the `cpu`, `cpuacct` subsystem parameters.

```
# ls /sys/fs/cgroup/cpu,cpuacct/system.slice/sshd.service
cgroup.clone_children  cpuacct.usage_percpu  cpu.shares
cgroup.event_control   cpu.cfs_period_us    cpu.stat
cgroup.procs           cpu.cfs_quota_us     notify_on_release
cpuacct.stat           cpu.rt_period_us      tasks
cpuacct.usage          cpu.rt_runtime_us
```

- d. Use the `systemctl show` command to view the value of the `CPUShares` property and the value of the `CPUAccounting` property for the `sshd` service.
- Note that the change is effective immediately.

```
# systemctl show --property=CPUShares --property=CPUAccounting
sshd
CPUAccounting=yes
CPUShares=512
```

- e. Use the `cat` command to view the contents of the `/sys/fs/cgroup/cpu,cpuacct/system.slice/sshd.service/cpu.shares` file.
- Note that the change is also written to the configuration unit file so that it persists across reboot.

- All of the other `cpu`, `cputacct` subsystem parameters in the directory contain default values.

```
# cat
/sys/fs/cgroup/cpu,cputacct/system.slice/sshd.service/cpu.shares
512
```

- f. Use the `cat` command to view the contents of the `/sys/fs/cgroup/cpu,cputacct/system.slice/sshd.service/tasks` file.
- In this example, the output is 1237.
  - This is the PID of the `sshd` service, which can be confirmed by using the `ps` command.
  - In the `ps -ef` command, grep for whatever number was contained in the `tasks` file.

```
# cat /sys/fs/cgroup/cpu,cputacct/system.slice/sshd.service/tasks
1237
# ps -ef | grep 1237
root 1237 1 ... /usr/sbin/sshd -D
```

3. Use the `systemctl set-property` command to change the value of a `memory` subsystem parameter for the `sshd` service.

- a. Before setting a `memory` subsystem parameter, use the `ls` command to view the contents of the `/sys/fs/cgroup/memory/system.slice/sshd.service` directory.
- You view this directory again after setting the `memory` subsystem parameter.

```
# ls /sys/fs/cgroup/memory/system.slice/sshd.service
ls: cannot access
/sys/fs/cgroup/memory/system.slice/sshd.service: No such file or
directory
```

- Note that before setting any `memory` subsystem parameters, the directory does not exist.
- Change the `MemoryLimit` property to `1G` and change the `MemoryAccounting` property to `true` for the `sshd` service.
- This limits the maximum amount of memory the `sshd` service can use to 1 GB.

```
# systemctl set-property sshd MemoryLimit=1G
MemoryAccounting=true
```

- c. Run the `ls` command in step 3a again.
- Note that the directory now exists and contains the `memory` subsystem parameters.

```
# ls /sys/fs/cgroup/memory/system.slice/sshd.service
cgroup.clone_children    memory.max_usage_in_bytes
cgroup.event_control     memory.memsw.failcnt
cgroup.procs              memory.memsw.limit_in_bytes
memory.failcnt            memory.memsw.max_usage_in_bytes
...
memory.limit_in_bytes    tasks
```

- d. Use the `systemctl show` command to view the value of the `MemoryLimit` property and the value of the `MemoryAccounting` property for the `sshd` service.
- Note that the change is effective immediately.

```
# systemctl show --property=MemoryLimit --property=MemoryAccounting sshd
MemoryAccounting=yes
MemoryLimit=1073741824
```

- e. Use the `cat` command to view the contents of the `/sys/fs/cgroup/memory/system.slice/sshd.service/memory.limit_in_bytes` file.
- Note that the change is also written to the configuration unit file so that it persists across reboot.
  - All of the other `memory` subsystem parameters in the directory contain default values.

```
# cat /sys/fs/cgroup/memory/system.slice/sshd.service/memory.limit_in_bytes
1073741824
```

- f. Use the `cat` command to view the contents of the `/sys/fs/cgroup/memory/system.slice/sshd.service/tasks` file.
- In this example, the output is 1237.
  - This is the PID of the `sshd` service.

```
# cat /sys/fs/cgroup/memory/system.slice/sshd.service/tasks
1237
```

4. Define resource control settings for the `crond` service in a unit configuration file.
- In addition to changing resource control settings from the command line using the `systemctl` command, you can define resource control settings in the unit configuration file.
  - a. Use the `cp` command to copy the `crond.service` file from the `/usr/lib/systemd/system` directory to the `/etc/systemd/system` directory.

- Unit configuration files in the `/usr/lib/systemd/system` directory are provided by the installed RPM packages and are not to be edited.
- Unit configuration files in the `/etc/systemd/system` directory take precedence over files in the `/usr/lib/systemd/system` directory.

```
# cp /usr/lib/systemd/system/crond.service /etc/systemd/system/
```

- b. Use the `vi` editor to edit the `/etc/systemd/system/crond.service` file.
- Set the `MemoryLimit` property to 500M.
  - Set the `MemoryAccounting` property to true.
  - Change the resource settings for a service under the `[Service]` heading in the unit configuration file. Changes in the example are in bold font.

```
# vi /etc/systemd/system/crond.service
[Unit]
```

```

Description=Command Scheduler
After=auditd.service systemd-user-sessions.service time-sync...

[Service]
MemoryLimit=500M
MemoryAccounting=true
EnvironmentFile=/etc/sysconfig/crond
ExecStart=/usr/sbin/crond -n $CRONDARGS
ExecReload=/bin/kill -HUP $MAINPID
KillMode=process

[Install]
WantedBy=multi-user.target

```

- c. Use the `systemctl show` command to view the properties of the `crond` service unit, pipe the output to `grep -i`, and search for the “memory” string.

- Note that the properties are set to default values.

```

# systemctl show crond | grep -i memory
MemoryAccounting=no
MemoryLimit=18446744073709551615

```

- d. After editing the unit configuration files, use the `systemctl daemon-reload` command to reload all unit files and re-create the entire dependency tree.

```
# systemctl daemon-reload
```

- e. Use the `systemctl restart` command to restart the `crond` service.

```
# systemctl restart crond
```

- f. Run step 4c again to view the changes.

```

# systemctl show crond | grep -i memory
MemoryAccounting=yes
MemoryLimit=524288000

```

- g. Use the `ls` command to view the contents of the `/sys/fs/cgroup/memory/system.slice/crond.service` directory.

- Note that the directory exists and is populated with the memory subsystem parameters.

```

# ls /sys/fs/cgroup/memory/system.slice/crond.service
cgroup.clone_children    memory.max_usage_in_bytes
cgroup.event_control     memory.memsw.failcnt
cgroup.procs              memory.memsw.limit_in_bytes
memory.failcnt           memory.memsw.max_usage_in_bytes
...
memory.limit_in_bytes   tasks

```

- h. Use the `cat` command to view the contents of the `/sys/fs/cgroup/memory/system.slice/crond.service/tasks` file.

- In this example, the output is 7344.
- This is the PID of the `crond` service, which can be confirmed by using the `ps` command.
- In the `ps -ef` command, grep for whatever number was contained in the `tasks` file.

```
# cat /sys/fs/cgroup/memory/system.slice/crond.service/tasks  
7344  
# ps -ef | grep 7344  
root 7344 1 ... /usr/sbin/crond -n
```

## 5. Log off host03.

Use the `exit` command to log off **host03**.

```
# exit  
logout  
Connection to host03 closed.
```

## **Practices for Lesson 14: Virtualization with Linux**

**Chapter 14**

## Practices for Lesson 14: Overview

---

### Practices Overview

In these practices, you get familiar with the tools to create and manage virtual guests in a KVM environment.

When active, KVM turns a physical machine into a virtualization host.

In the practices for this lesson, you use a Linux host called **host05**, which is not a physical machine in your practice environment; **host05** is a virtual machine running on your lab PC. For this reason, KVM is available in **host05** but is not active. All the tasks in these practices work the same, whether KVM is active or not. The side effect of KVM not being active is that virtual guests deployed from **host05** perform very slowly. This side effect has little impact on the exercises.

During the practices for this lesson, you use the following commands to manage your virtual guests: `virt-manager` and `virsh`. These two commands are part of the `libvirt` toolkit. With the `libvirt` toolkit, you can manage the virtualization capabilities offered by KVM on your virtualization host, **host05**.

## Practice 14-1: Preparing the Virtualization Host for KVM

---

### Overview

In this practice, you prepare your virtualization host, **host05**, to run a virtual guest. The virtualization host runs Oracle Linux 7.1.

### Assumptions

This practice makes the following assumptions about **host05**, your virtualization host:

- The virtualization package groups are already installed:
  - Virtualization Client
  - Virtualization Platform
  - Virtualization Hypervisor
  - Virtualization Tools
- SELinux is enabled, and the firewall is enabled.
- IP forwarding is enabled for NAT networking.

This practice also assumes that you are the `root` user on **dom0**.

### Tasks

1. Shut down all VMs.

- You need to release the memory allocated to the running VMs before starting **host05**.
- a. From **dom0**, run the `xm list` command to display the running VMs.

# xm list						
Name	ID	Mem	VCPUs	State	Time (s)	
Domain-0	0	2048	2	r-----	...	
host01	242	1536	1	-b----	...	
host02	243	1536	1	-b----	...	
host03	244	1536	1	-b----	...	

- In this example, **host01**, **host02**, and **host03**, in addition to **Domain-0**, are running.
- b. Use the `xm destroy <host#>` command to shut down each running VM.
- This example shuts down all three VMs.
- Run the `xm list` command again to verify all three VMs have been shut down.

# xm destroy host01
# xm destroy host02
# xm destroy host03
# xm list
Name ID Mem VCPUs State Time (s)
Domain-0 0 2048 2 r----- ...

- In this example, all three VMs have been shut down.

2. Start the **host05** virtual machine, which acts as your virtualization host for the practices in this lesson.

- a. Use the `cd` command to change to the `/OVS/running_pool/host05` directory.

```
# cd /OVS/running_pool/host05
```

- b. Run the `xm create` command to start the **host05** virtual machine.

```
# xm create vm.cfg
```

3. Log in to **host05**.

- a. Determine the VNC port number for **host05** by running the `xm list -l host05 | grep location` command.

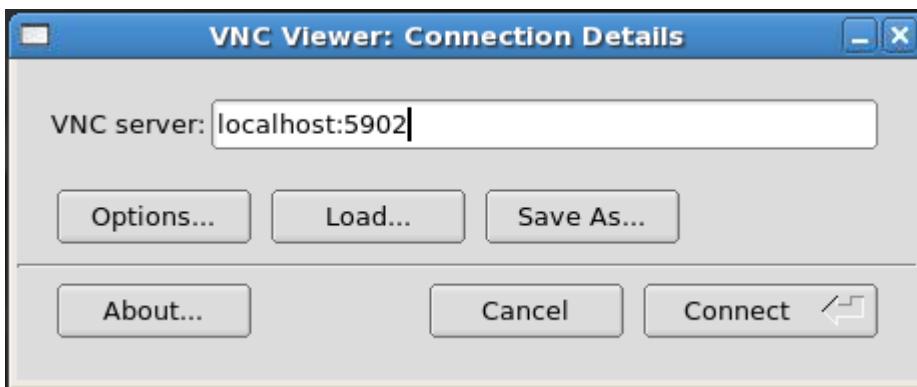
```
# xm list -l host05 | grep location
      (location 0.0.0.0:5902)
      (location 3)
```

- The sample shown indicates that the port number is 5902. Your port number might be different.

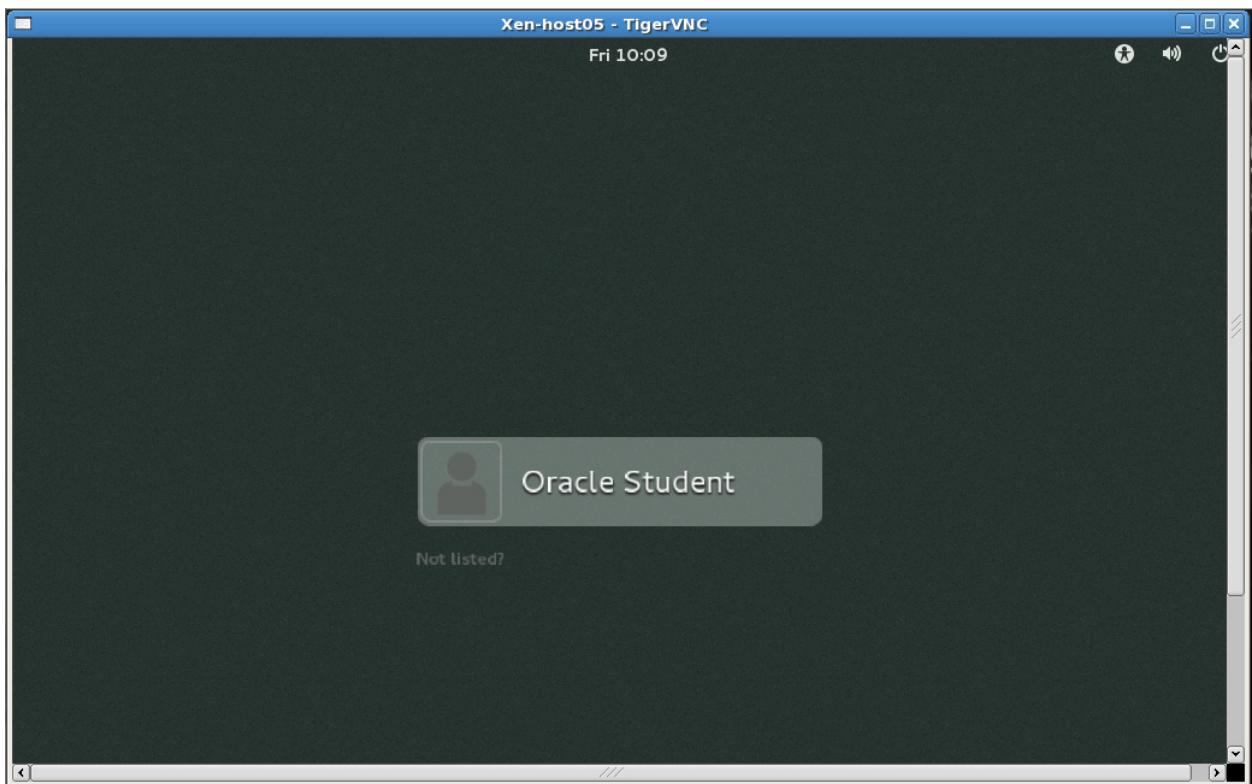
- b. Run the `vncviewer&` command.

```
# vncviewer&
```

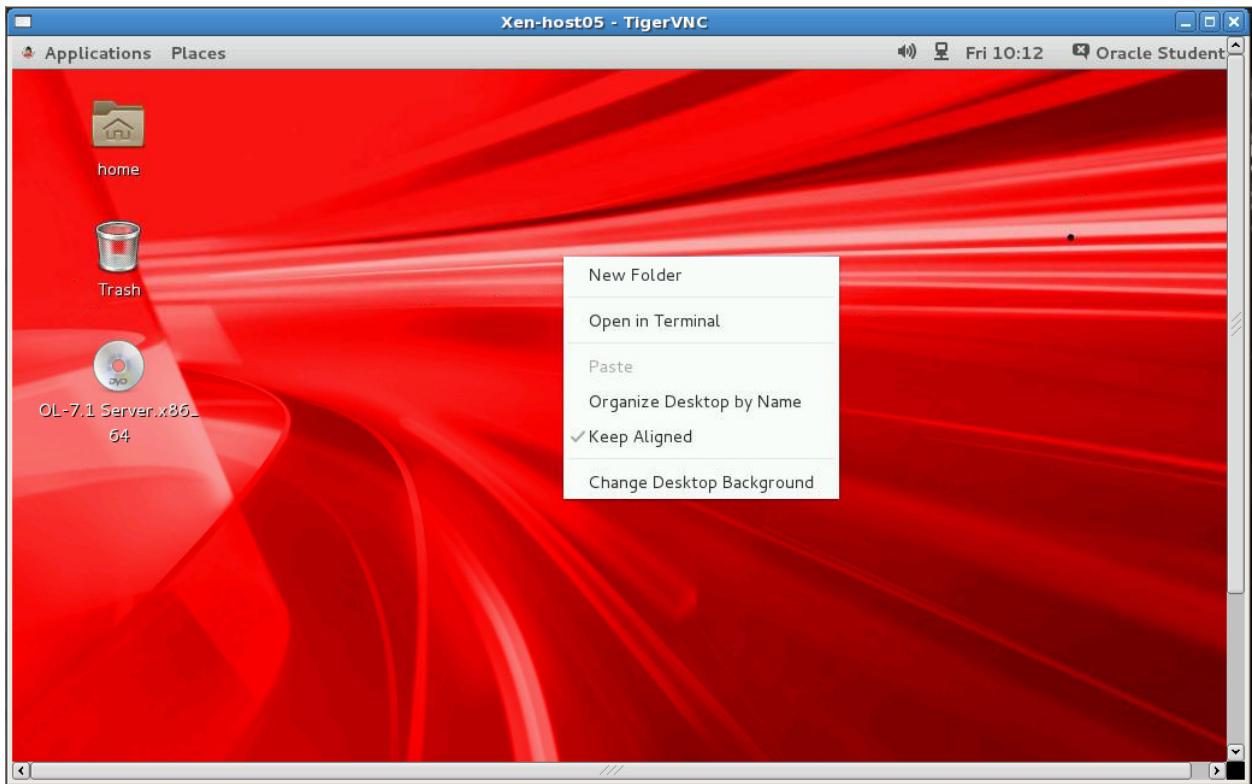
- The VNC Viewer: Connection Details dialog box is displayed.
- Enter `localhost:<port_number>`, substituting the port number displayed from the previous `xm list -l host05 | grep location` command. For example, if the port number is 5902, enter `localhost:5902` and click “Connect.”



- The GNOME login screen appears. You might have to press ENTER to see the screen.



- d. Click “Oracle Student” in the list of users. You are prompted for the password.
- e. Enter `oracle` for the Password and click “Sign In.”
  - The GNOME desktop appears.
- f. Right-click on the desktop to display the pop-up menu.



- g. From the pop-up menu, select “Open in Terminal.”
  - A terminal window appears.
- h. In the terminal window, use the `su -` command to become the `root` user.
  - The `root` password is `oracle`.

```
$ su -
Password: oracle
#
```

4. List and examine the packages that are installed to support KVM operations.
  - a. Use the `rpm -ql` command to list the executable files in the `libvirt-daemon` package.
    - Pipe the output to `grep bin` to display only the commands and utilities.

```
# rpm -ql libvirt-daemon | grep bin
/usr/sbin/libvirtd
/usr/sbin/virtlockd
```

- The `libvirt-daemon` package contains the `libvirtd` daemon, which communicates with the hypervisor on the virtualization host.

- b. Use the `rpm -q1` command to list the executable files in the `libvirt-client` package.

- Pipe the output to `grep bin` to display only the commands and utilities.

```
# rpm -q1 libvirt-client | grep bin
/usr/bin/virsh
/usr/bin/virt-host-validate
/usr/bin/virt-pki-validate
/usr/bin/virt-xml-validate
```

- The `libvirt-client` package contains the `virsh` command, an important command that you use to manage virtual machines and their resources.

- c. Use the `rpm -q1` command to list the executable files in the `virt-manager` package.

- Pipe the output to `grep bin` to display only the commands and utilities.

```
# rpm -q1 virt-manager | grep bin
/usr/bin/virt-manager
```

- The `virt-manager` package contains the `virt-manager` command that launches the Virtual Machine Manager graphical user interface.

- d. Use the `rpm -q1` command to list the executable files in the `virt-install` package.

- Pipe the output to `grep bin` to display only the commands and utilities.

```
# rpm -q1 virt-install | grep bin
/usr/bin/virt-clone
/usr/bin/virt-install
/usr/bin/virt-xml
```

- The `virt-install` package contains the `virt-install` command that you can use to create virtual machines from the command line or from within scripts.

## 5. Examine the status of KVM.

- a. Use the `lsmod` command to list the loaded KVM modules.

```
# lsmod | grep kvm
```

- In this example, the KVM modules are not loaded.

- b. Use the `modprobe` command to load the KVM module.

- Repeat the `lsmod` command to verify that the KVM module is loaded.

```
# modprobe kvm
# lsmod | grep kvm
kvm                  432586  0
```

- The `kvm` module is now loaded.
- The `kvm` module is not enough to use KVM successfully. KVM needs the hardware acceleration provided by the processor on the physical host machine.

- c. Attempt to load the `kvm` module associated with either Intel or AMD:

```
# modprobe kvm-intel
modprobe: ERROR: could not insert 'kvm-intel': Operation not
supported
# modprobe kvm-amd
```

```
modprobe: ERROR: could not insert 'kvm-amd': Operation not
supported
```

- In your practice environment, you cannot load the Intel or AMD KVM module in **host05** because **host05** is not running directly on the physical machine. As a result, you cannot use the KVM acceleration provided by the processor. This is a limitation of your practice environment, not a limitation of KVM.
  - Without KVM acceleration, **host05** uses the QEMU virtual machine emulator exclusively to run virtual guests. All the practices in this lesson work the same, whether the acceleration is present or not. Without the KVM acceleration, the performance of the virtual machine that you deploy in Practice 14-3 is going to be poor but this situation does not affect the outcome of the practice.
  - At your own site, with proper support by your Intel or AMD-based processor, you can take full advantage of the KVM acceleration provided by the hardware extensions in your physical machine.
- d. Use the `virt-host-validate` command to check if the host supports all of the libvirt hypervisor drivers it is aware of.
- This command shows that hardware virtualization is not supported.

```
# virt-host-validate
  QEMU: Checking for hardware virtualization      : WARN (Only
emulated CPUs are available, performance will be significantly
limited)
  QEMU: Checking for device /dev/vhost-net      : PASS
  QEMU: Checking for device /dev/net/tun        : PASS
  LXC: Checking for Linux >= 2.6.26            : PASS
```

6. Ensure that the `libvирtd` daemon is running and starts at boot time.

Use the `systemctl` command to verify that the `libvирtd` service is running.

- In this example, `libvирtd` is loaded and running.

```
# systemctl status libvирtd
libvирtd.service - Virtualization daemon
   Loaded: loaded (/usr/lib/systemd/system/libvирtd.service; ...)
   Active: active (running) since ...
```

7. Ensure that IP forwarding is enabled for NAT networking.

Use the `sysctl` command to view the setting of `net.ipv4.ip_forward`.

```
# sysctl -a | grep net.ipv4.ip_forward
net.ipv4.ip_forward = 1
```

- IP forwarding is enabled for you in **host05**. In other environments, you have to enable IP forwarding because `virt-manager` does not do it automatically for you.
- IP forwarding and NAT are needed for networking of virtual machines connecting to the default virtual network.

## Practice 14-2: Starting the Virtual Machine Manager and Preparing to Create a Virtual Machine

### Overview

In this practice, you start the libvirt graphical user interface by using the `virt-manager` command. The libvirt graphical user interface is called the Virtual Machine Manager.

With the Virtual Machine Manager, you manipulate two types of entities:

- Connections
- Virtual machines deployed using each connection.

A connection is used to connect to a specific hypervisor:

- The connection can be local (the hypervisor is located on the host where the Virtual Machine Manager runs) or remote.
- Supported hypervisors include KVM-QEMU, Linux Containers (LXC), and Xen.

In all the practices for this lesson, you use a connection to a KVM-QEMU type hypervisor.

Before attempting to create virtual machines, ensure that the networking and storage resources are in place. You perform this verification from the Virtual Machine Manager.

### Assumptions

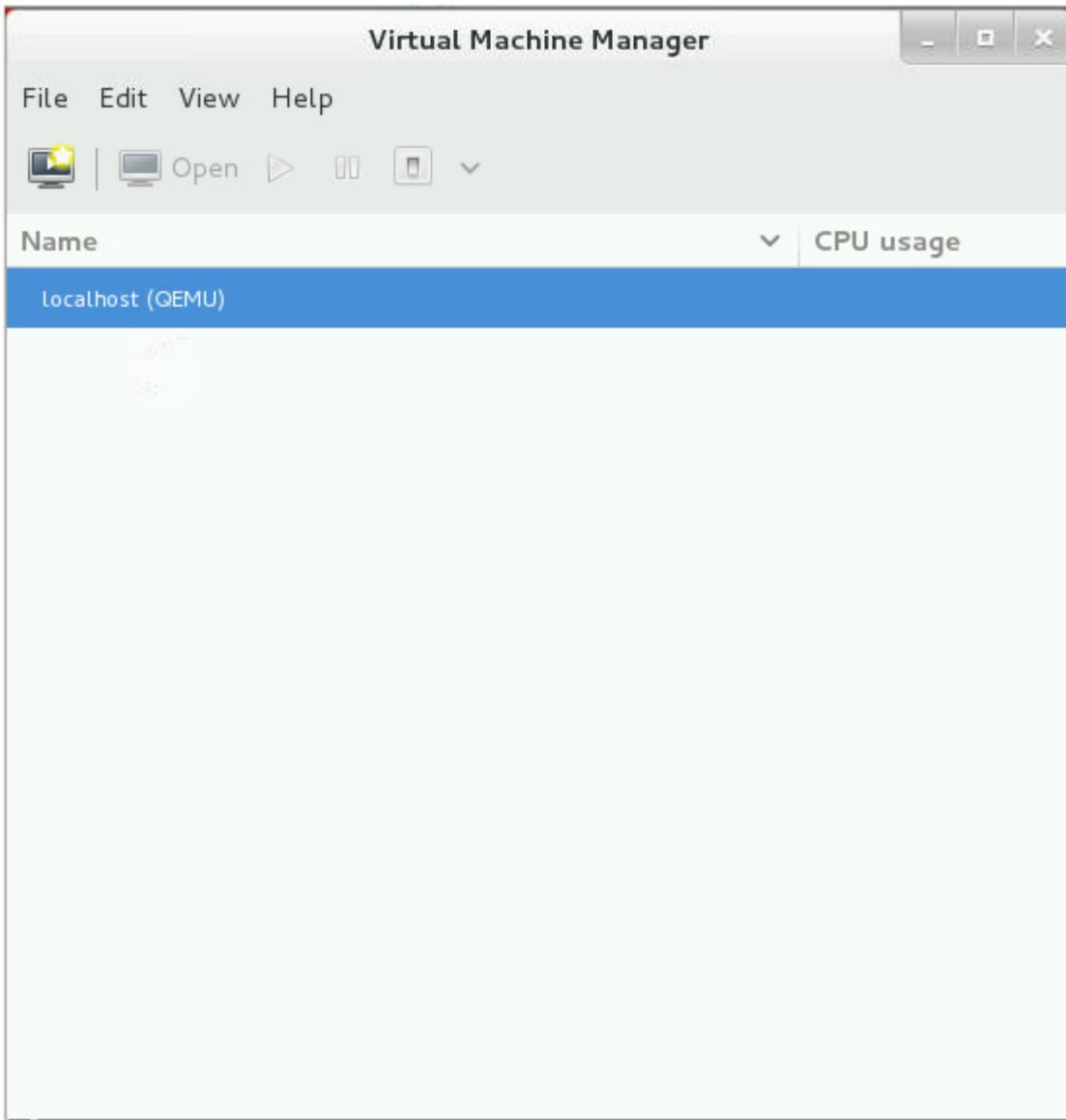
You are the `root` user on **host05**.

### Tasks

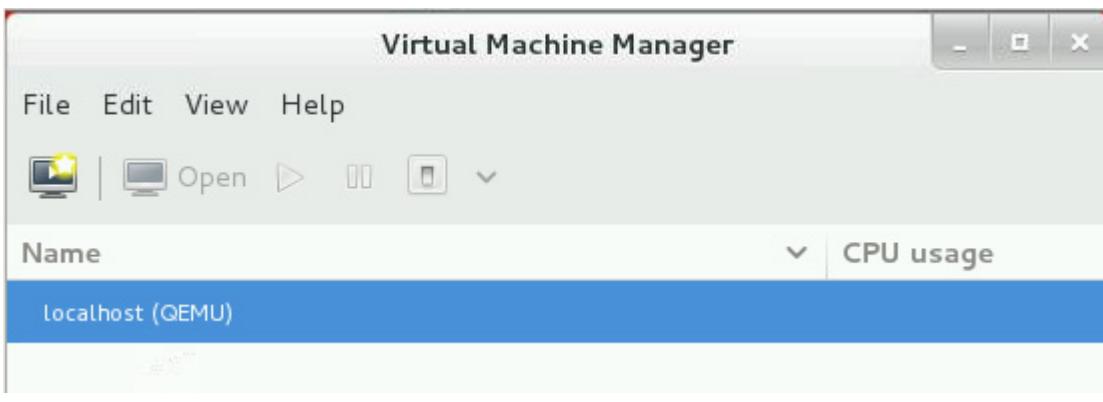
1. Start the Virtual Machine Manager.
  - a. From **host05**, run the `virt-manager` command to launch the Virtual Machine Manager.

```
# virt-manager&
```

- The Virtual Machine Manager main window appears.

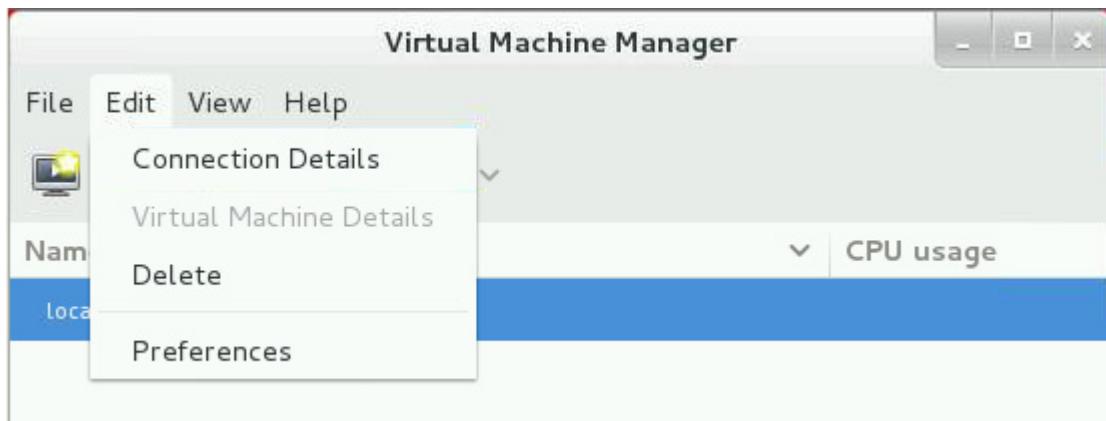


- If you get an error stating that the `libvirt` package is not installed, run the `virt-manager` command again.
2. Examine the connection and connection details.
- Highlight the default connection named `localhost`.

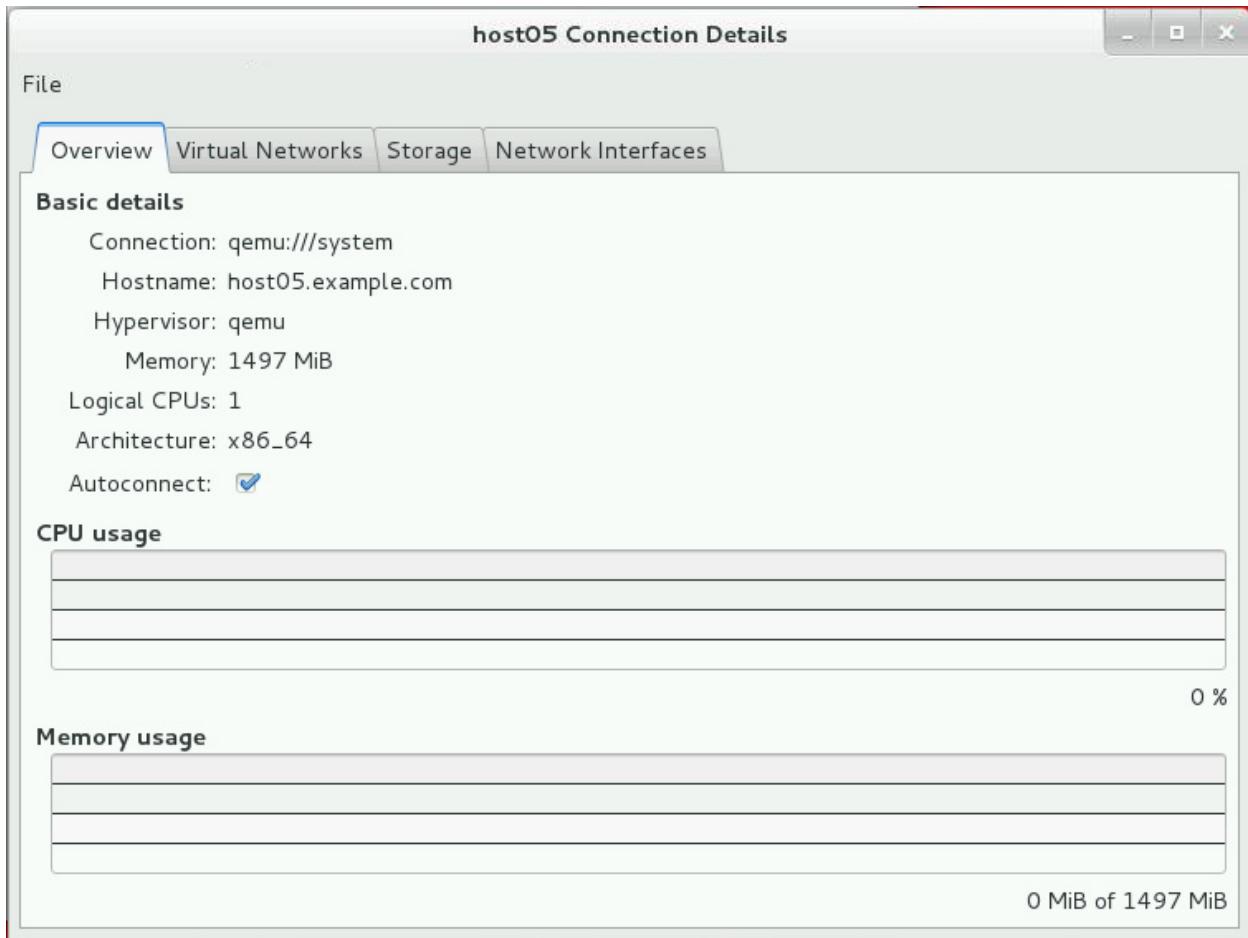


Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

- b. Select “Edit > Connection Details” from the menu bar.

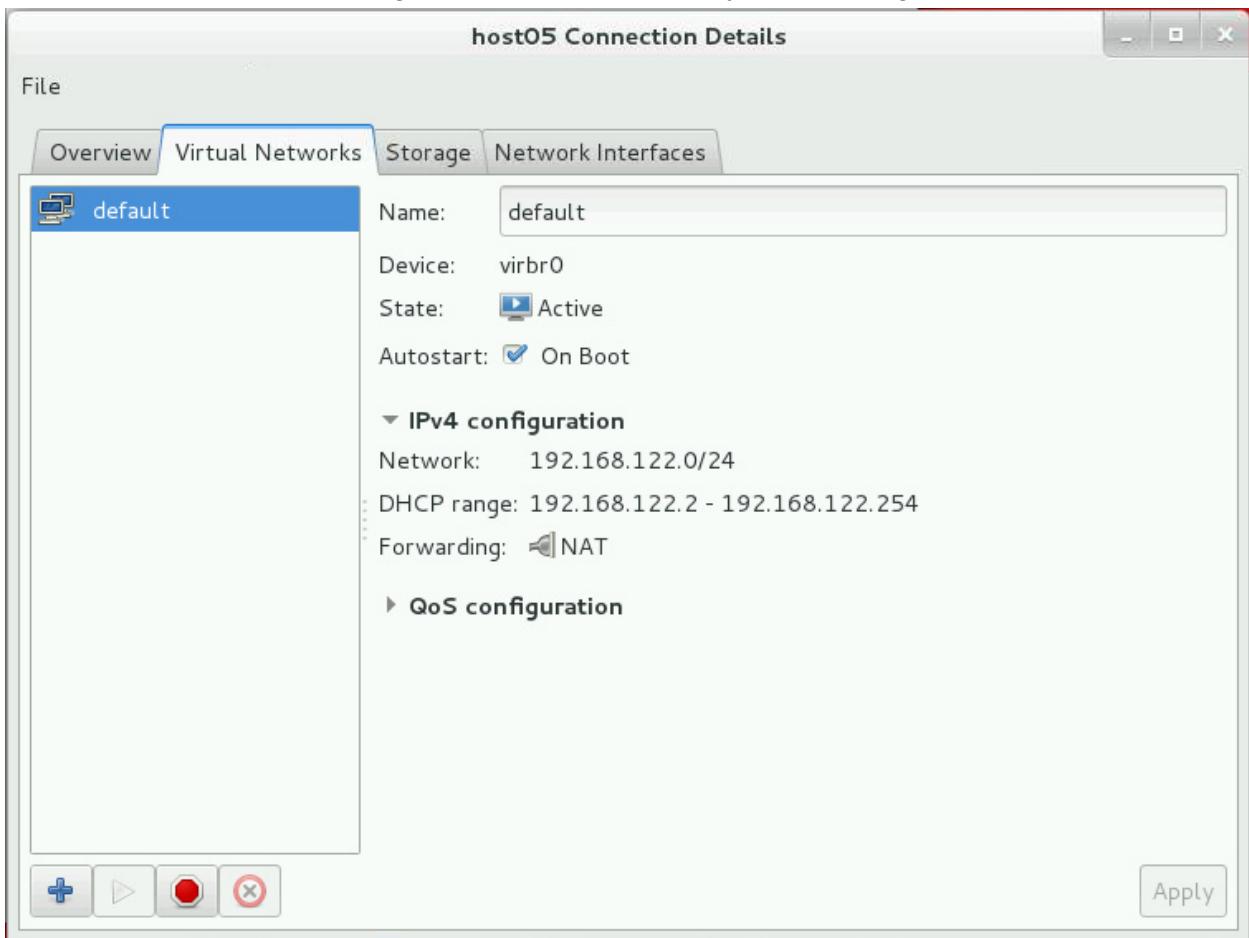


- The information for the Overview tab appears.



- This tab shows the type of hypervisor used, and the memory and virtual CPUs available to create virtual machines.
- Even when KVM is fully implemented, the Hypervisor field shows `qemu`. KVM and QEMU work together to run virtual machines.

3. Examine the “Virtual Networks” details.
  - a. Click the “Virtual Networks” tab to display information about virtual networks.
    - Expand the “IPv4 configuration” section to display the following window.



- There is only one virtual network available. This virtual network is called `default` and is configured automatically on `host05` when you start the Virtual Machine Manager for the first time.
  - You use the `default` virtual network when configuring networking for the virtual machine that you create in the next practice.
  - The default virtual network is associated with network device `virbr0`, which is a bridge. The network bridge acts like a virtual switch to which virtual machines connect.
  - The `virbr0` bridge is not attached to any physical NIC (like `eth0` or `eth1`) on the virtualization host. Instead, NAT and IP forwarding are used to forward packets from the virtual guests to the external network. You can find more information about virtual networking at this site: <http://wiki.libvirt.org/page/VirtualNetworking>.
- b. From a terminal window on `host05`, use the `brctl` command to display the bridge information for `virbr0`.

```
# brctl show
bridge name     bridge id          STP enabled      interfaces
virbr0          8000...           yes              virbr0-nic
```

- c. You can also obtain information for `virbr0` by using the `ip addr` command.

```
# ip addr
...
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc ...
    link encap:Ethernet 00:16:3e:00:01:05 brd ff:ff:ff:ff:ff:ff
    inet addr:192.0.2.105/24 brd 192.0.2.255 scope global eth0
    ...
3: virbr0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc...
    link/ether 52:54:00:d2:e7:1e brd ff:ff:ff:ff:ff:ff
    inet addr:192.168.122.1/24 brd 192.168.122.255 scope glo...
    ...
4: virbr0-nic: <BROADCAST,MULTICAST> mtu 1500 qdisc ...
    link/ether 52:54:00:d2:e7:1e brd ff:ff:ff:ff:ff:ff
```

- d. Use the `ps -ef` command and pipe the output to `grep` to display the `dnsmasq` process.

- The libvirt API uses the `dnsmasq` program to provide each virtual network switch with a range of IP addresses that are provided to guests through DHCP.

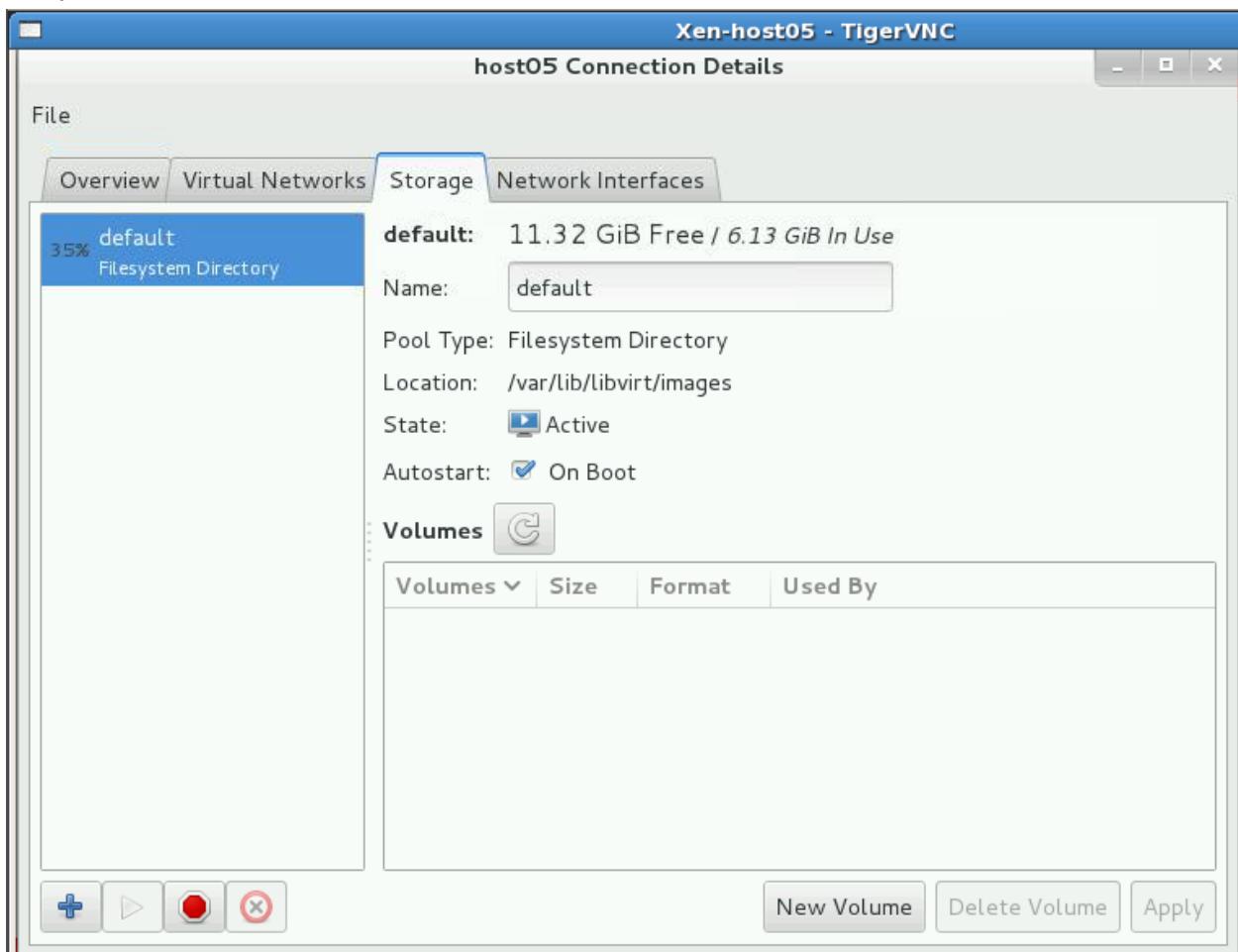
```
# ps -ef | grep dns
nobody ... /sbin/dnsmasq -conf-
file=/var/lib/libvirt/dnsmasq/default.conf ...
...
```

- e. Use the `cat` command to view the `/var/lib/libvirt/dnsmasq/default.conf` file.

- This file specifies the range of IP addresses.

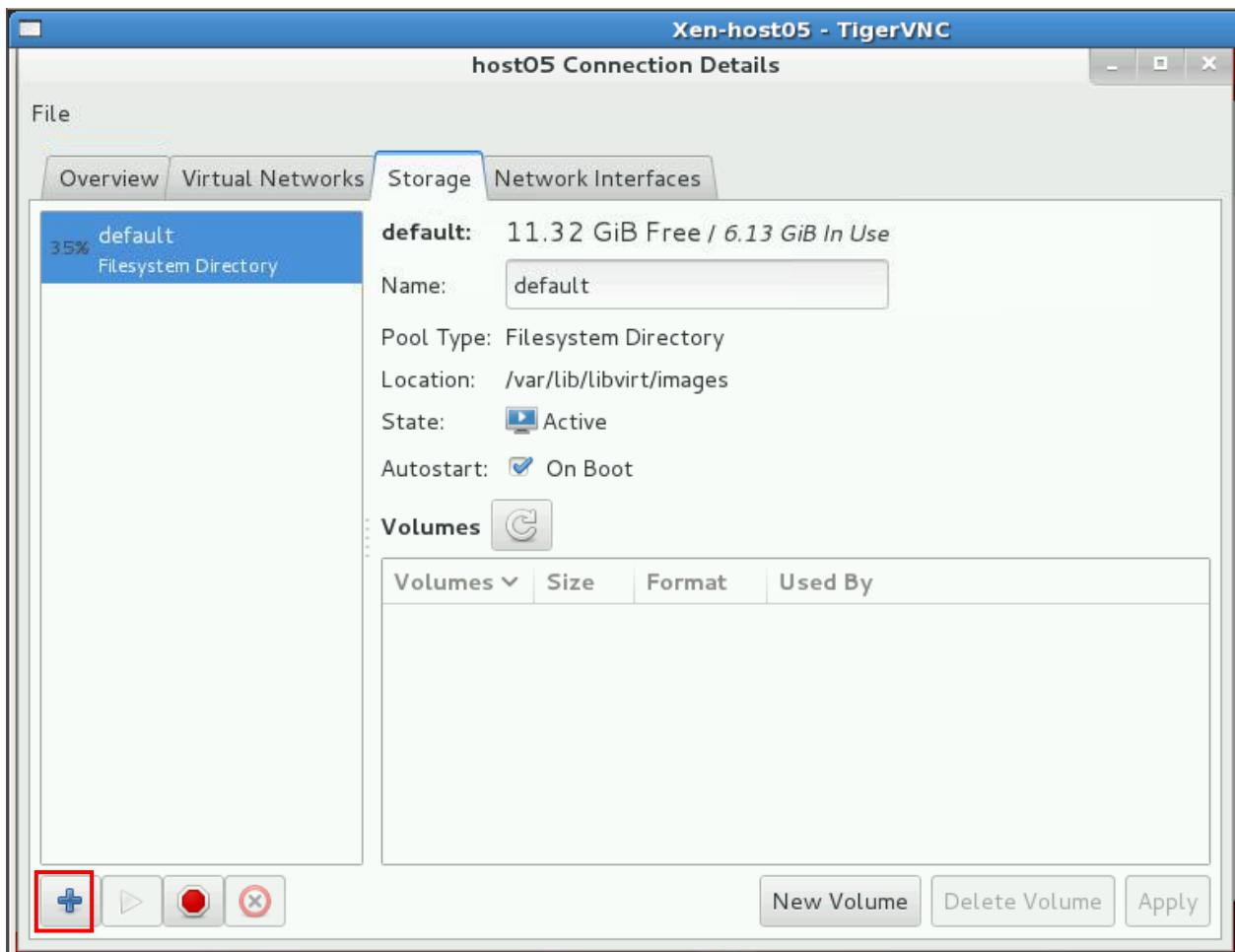
```
# cat /var/lib/libvirt/dnsmasq/default.conf
...
dhcp-range=192.168.122.2,192.168.122.254
...
```

4. Examine the “Storage” details and add a new storage pool.
  - a. Click the “Storage” tab to display information about available storage and volumes in your KVM environment.

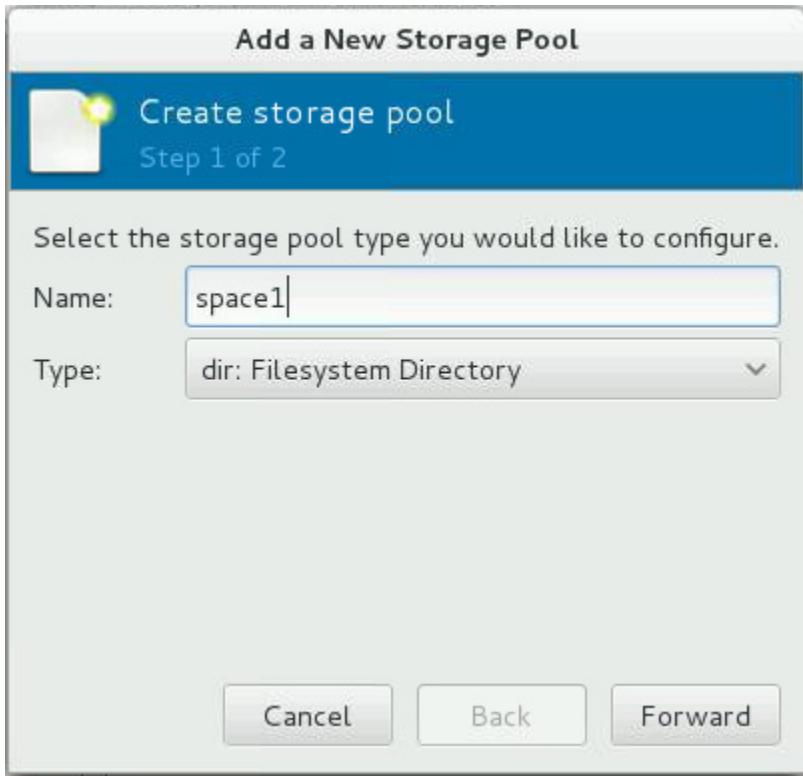


- By default, disk images for virtual machines are created in the /var/lib/libvirt/images directory, which is a storage pool of type file system. You can create additional pools of storage and these pools can reside on NFS shares or iSCSI volumes.
- There are no volumes in the Volumes area in the right pane for the default pool. As you create new virtual machines and assign storage to these virtual machines, new volumes appear in this window.
- When you create a virtual machine in the next practice, you use an already existing disk image for the virtual machine. In this step, you create a new storage pool for the filesystem directory where this existing disk image resides.

- b. Click the “Add Pool” button in the toolbar in the bottom left corner of the window.

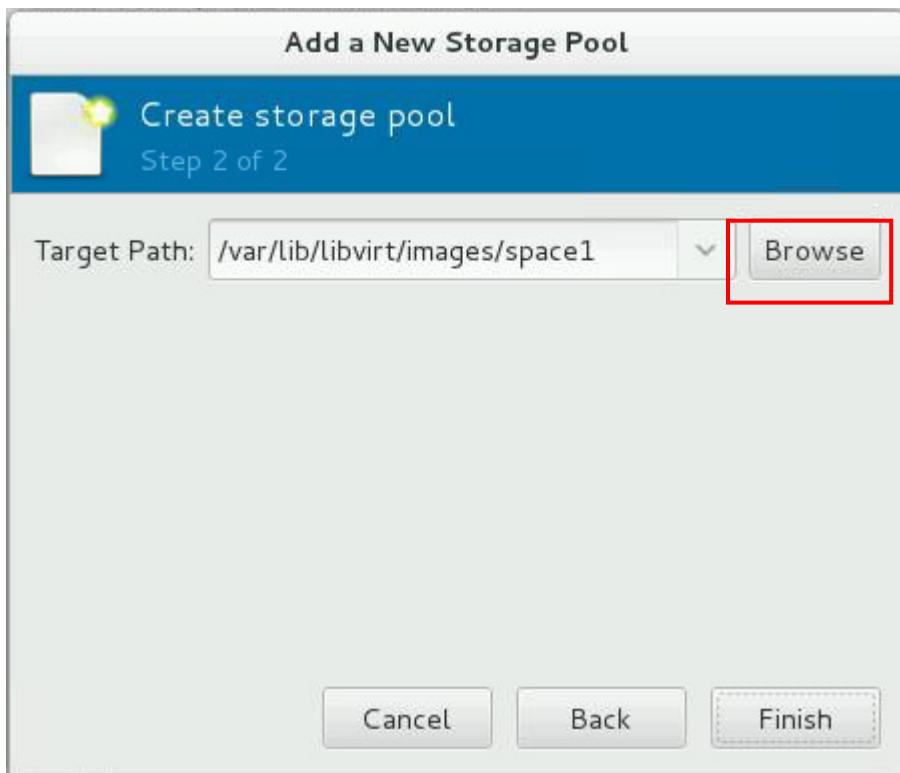


- c. On the “Add a New Storage Pool, Step 1 of 2” window:
- Enter `space1` as the new storage pool name.
  - Accept “dir: Filesystem Directory” as the storage pool type.

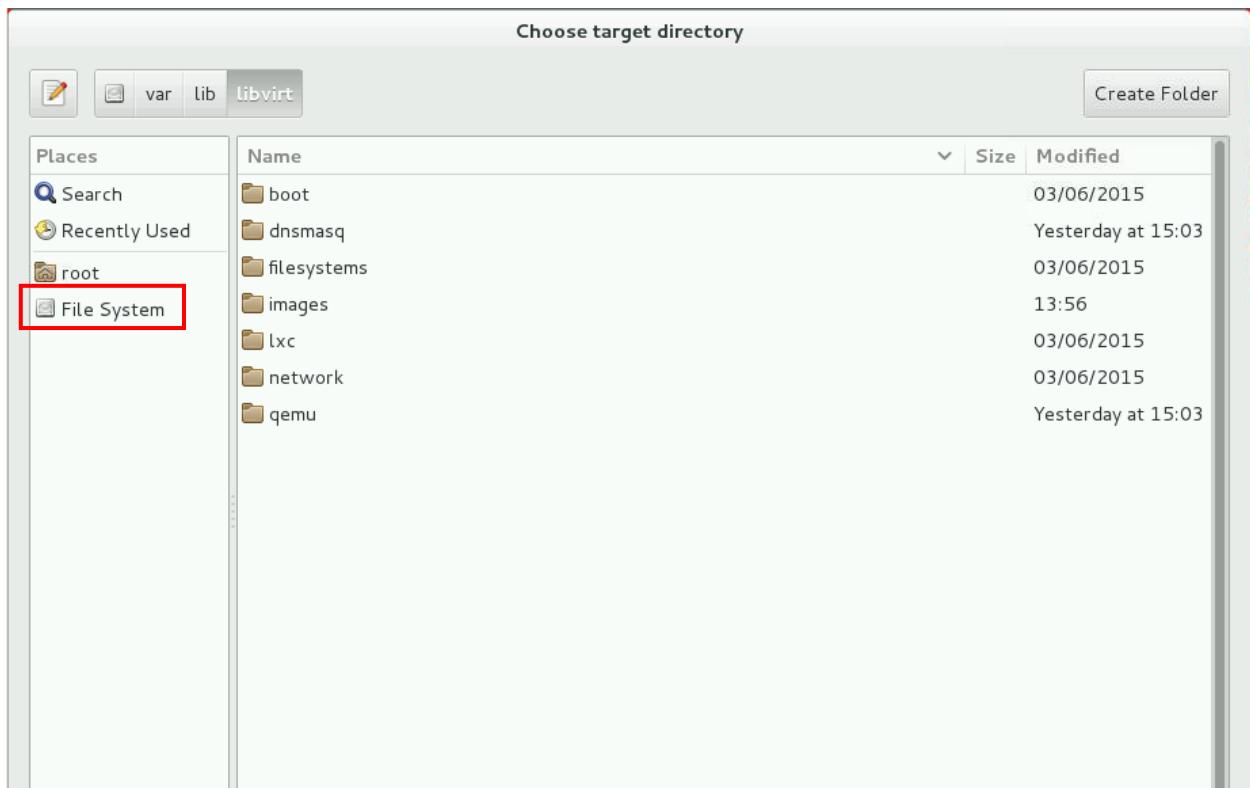


- Click “Forward” to continue.

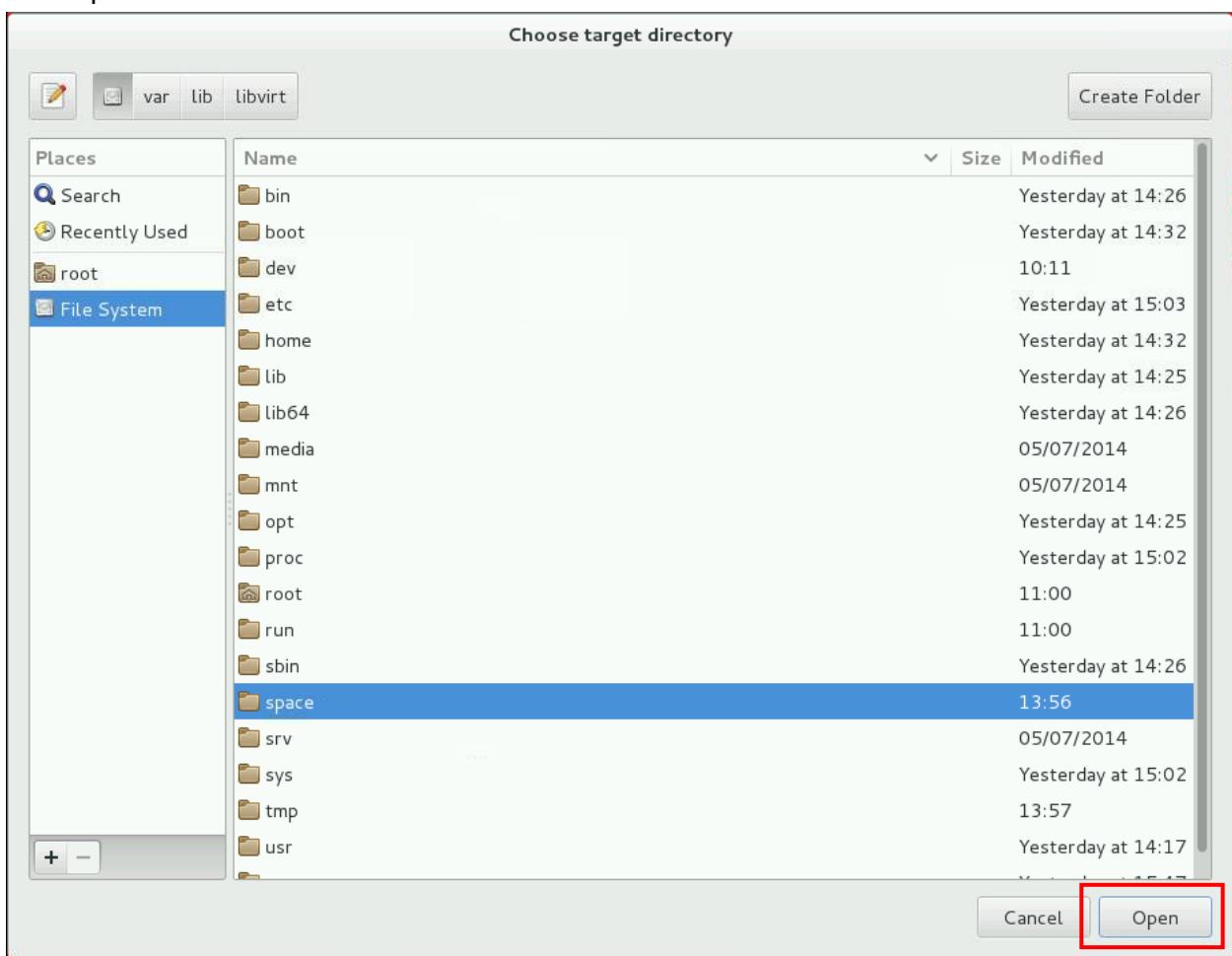
- d. On the “Add a New Storage Pool, Step 2 of 2” window, click the “Browse” button.



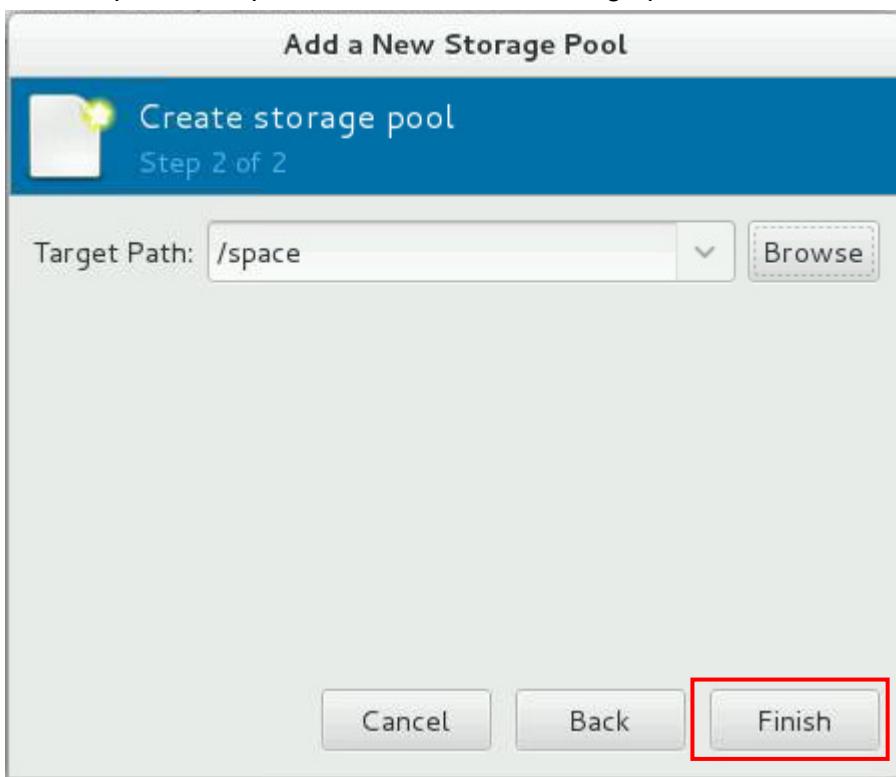
- e. On the “Choose target directory” window, click “File System.”



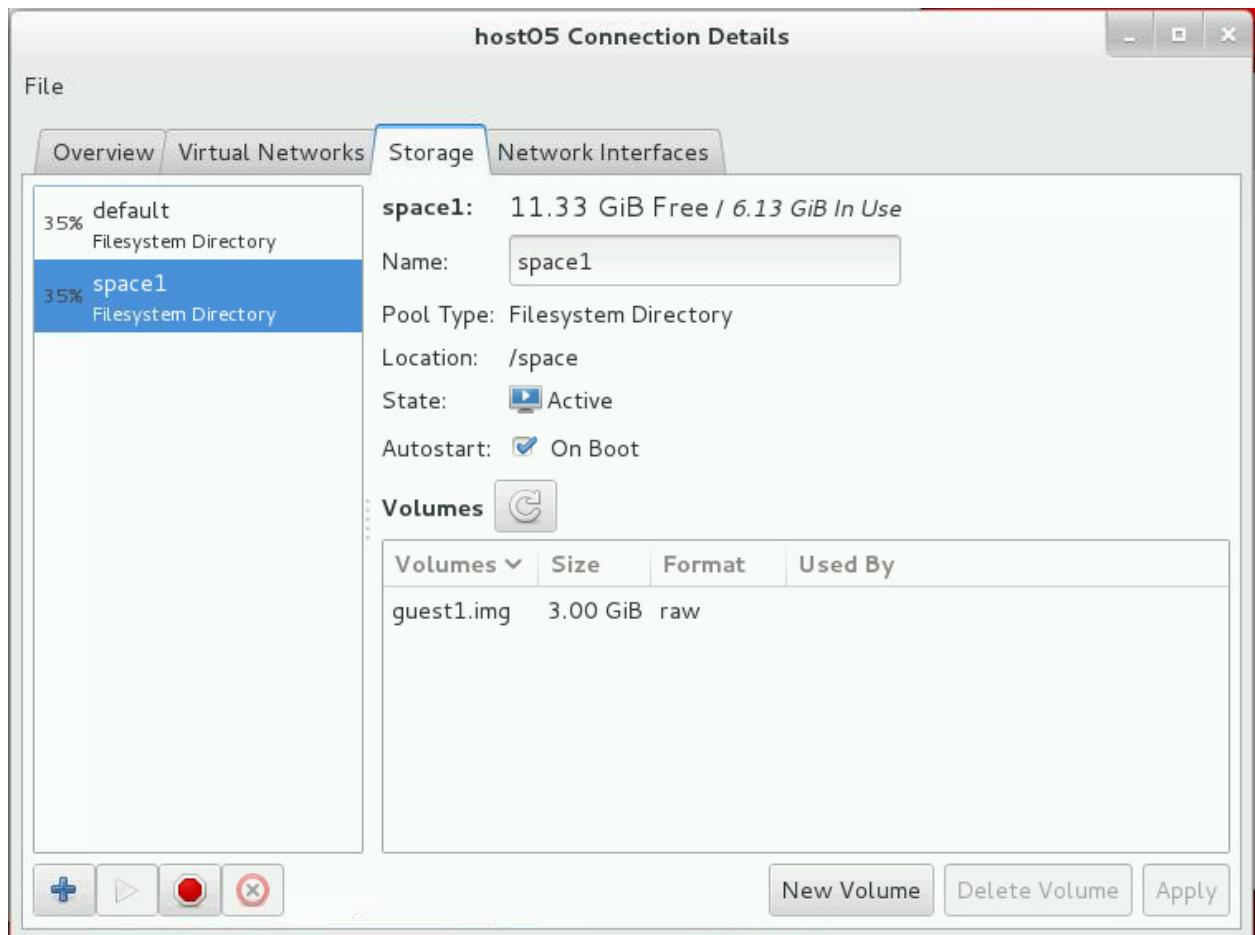
- f. In the window listing the directories and file systems, highlight space and click the "Open" button.



- g. On the “Add a New Storage Pool, Step 2 of 2” window, click the “Finish” button to complete the operation to add a new storage pool.



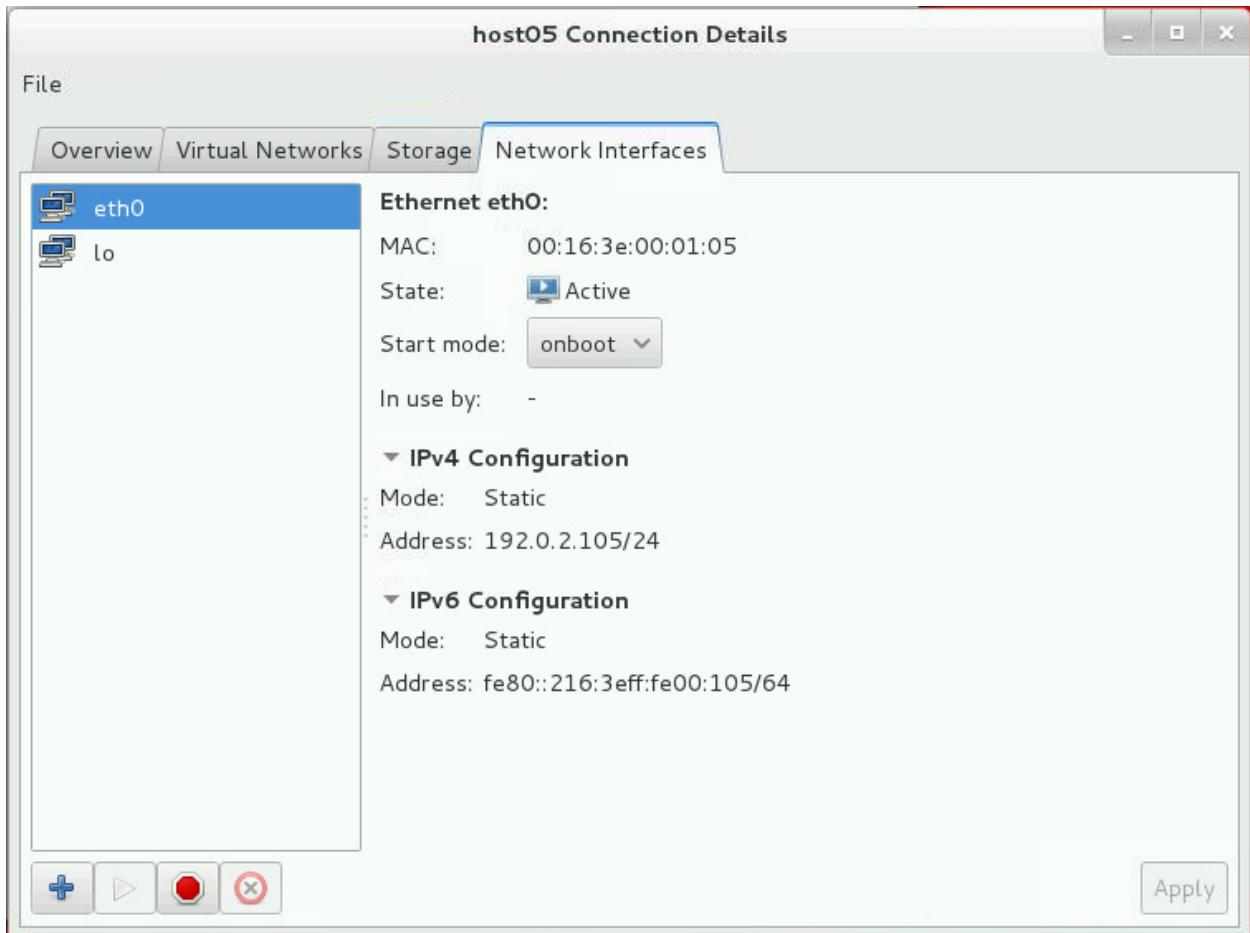
- h. Highlight the space1 storage pool in the left pane to list the existing volumes in the pool.
- Note the existence of the guest1.img volume in the /space file system directory.



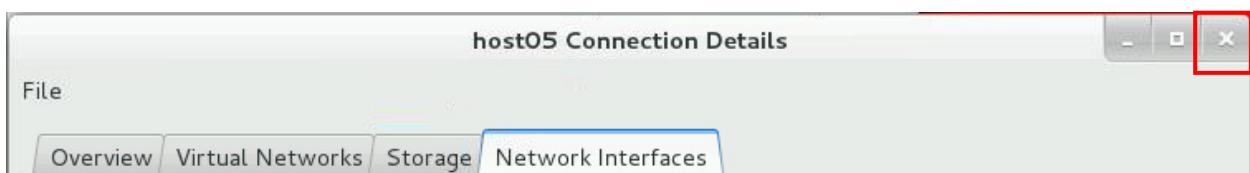
- i. From a terminal window, use the `ls -l` command to view the /space directory.
- The guest1.img contains a minimal install of Oracle Linux 7.1.
  - This image is provided for you to use when creating your new virtual machine in the next practice for this lesson.
  - Creating a new virtual machine from an existing image eliminates the need to perform a new installation of the operating system.

```
# ls -l /space
-rw----- 1 root root 3221225472 <date_time> guest1.img
```

5. Examine the “Network Interfaces” details and close the “host05 Connection Details” window.
- Click the “Network Interfaces” tab to display information about the NICs on **host05**.
    - Expand the “IPv4 Configuration” and “IPv6 Configuration” sections to display the following window.



- There is only one NIC available on **host05**. If other NICs are available, they show up in the left pane, and if a NIC is not configured, it is disabled.
  - You can use available NICs to create new virtual networks, using NAT with IP forwarding, or bridged virtual networks in routed mode as described in the slides for this lesson.
- Close the Virtual Machine Manager “host05 Connection Details” window by clicking the x in the top-right corner of the window.



## Practice 14-3: Creating a Virtual Machine

### Overview

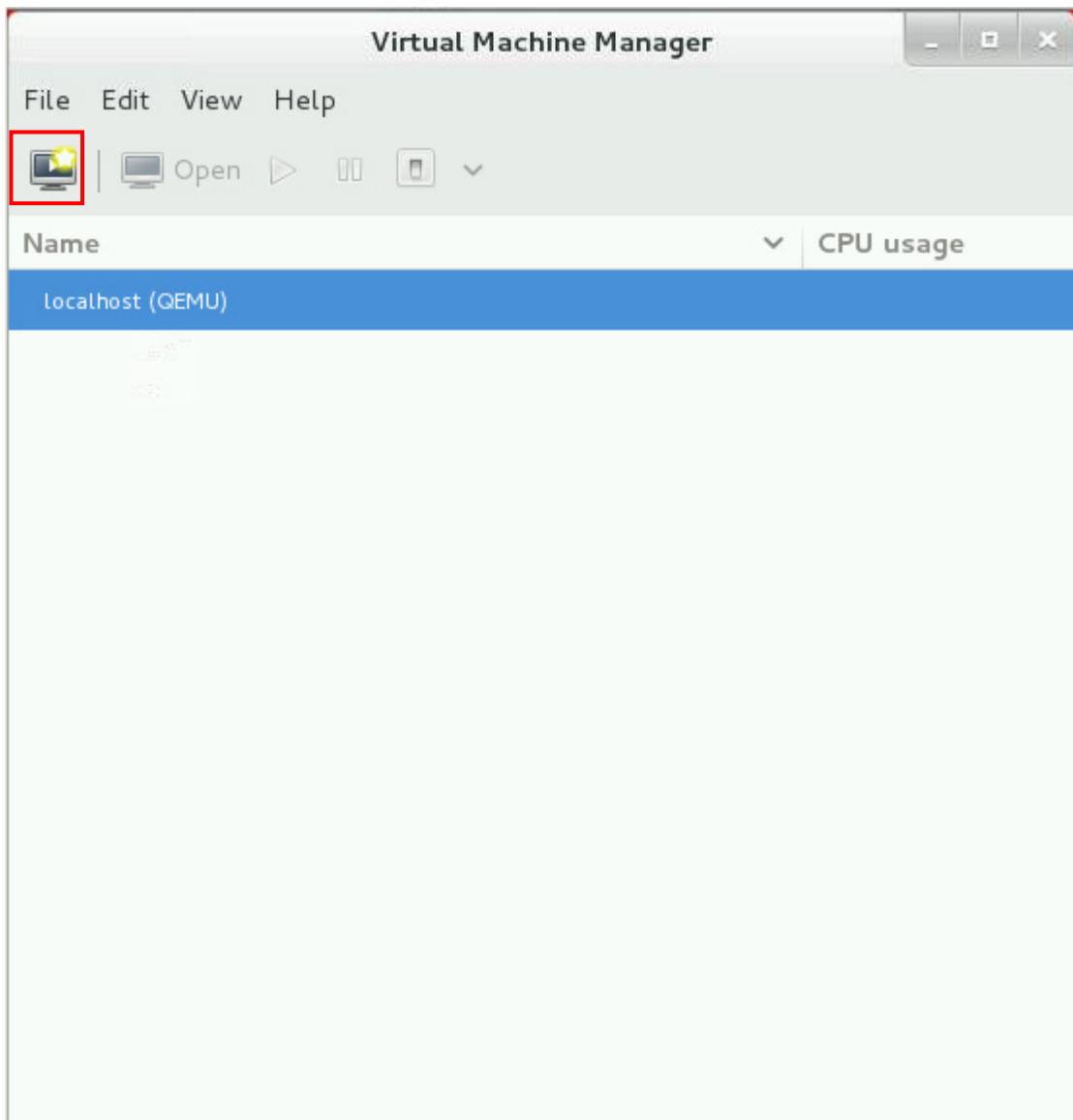
In this practice, you create a new virtual machine and import a disk image for this new virtual machine. The disk image already contains an installed OS, which means that you do not need to perform an OS installation for your new virtual machine.

### Assumptions

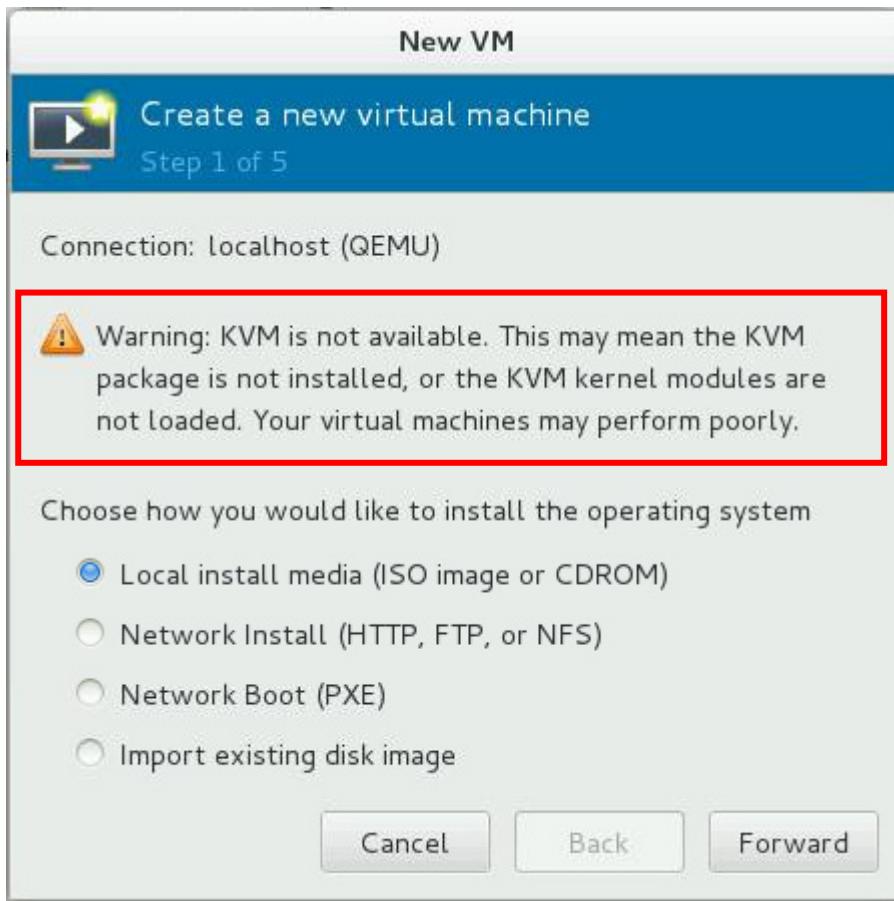
- Virtual Machine Manager is running on **host05**.

### Tasks

- Click the “Create a new virtual machine” icon in the toolbar in the Virtual Machine Manager GUI.

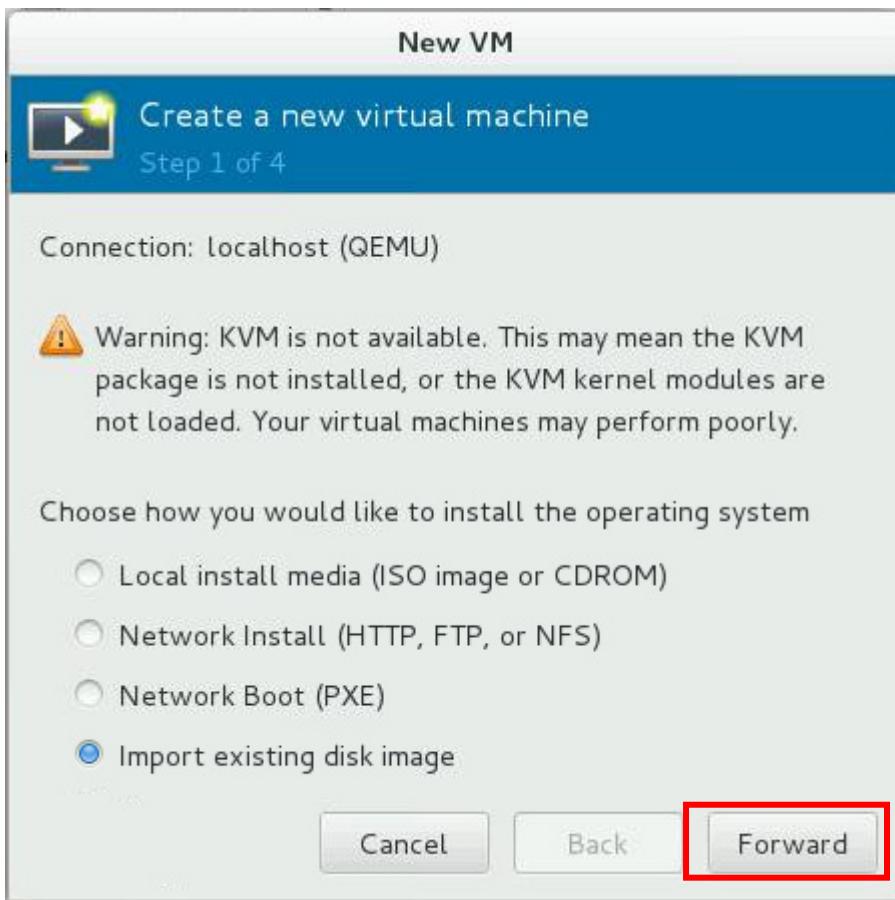


- Notice the warning in the window that appears.

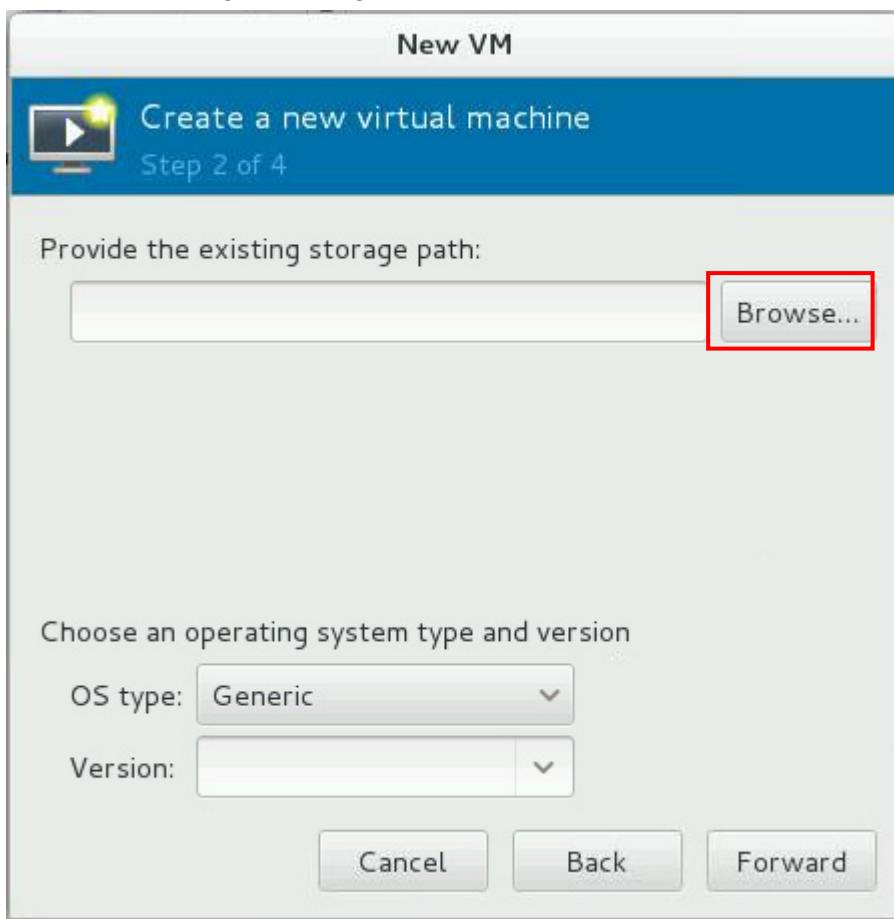


- KVM is not available, as discussed in the Practices Overview. You are still able to create and manage virtual machines, but the virtual machines perform poorly.

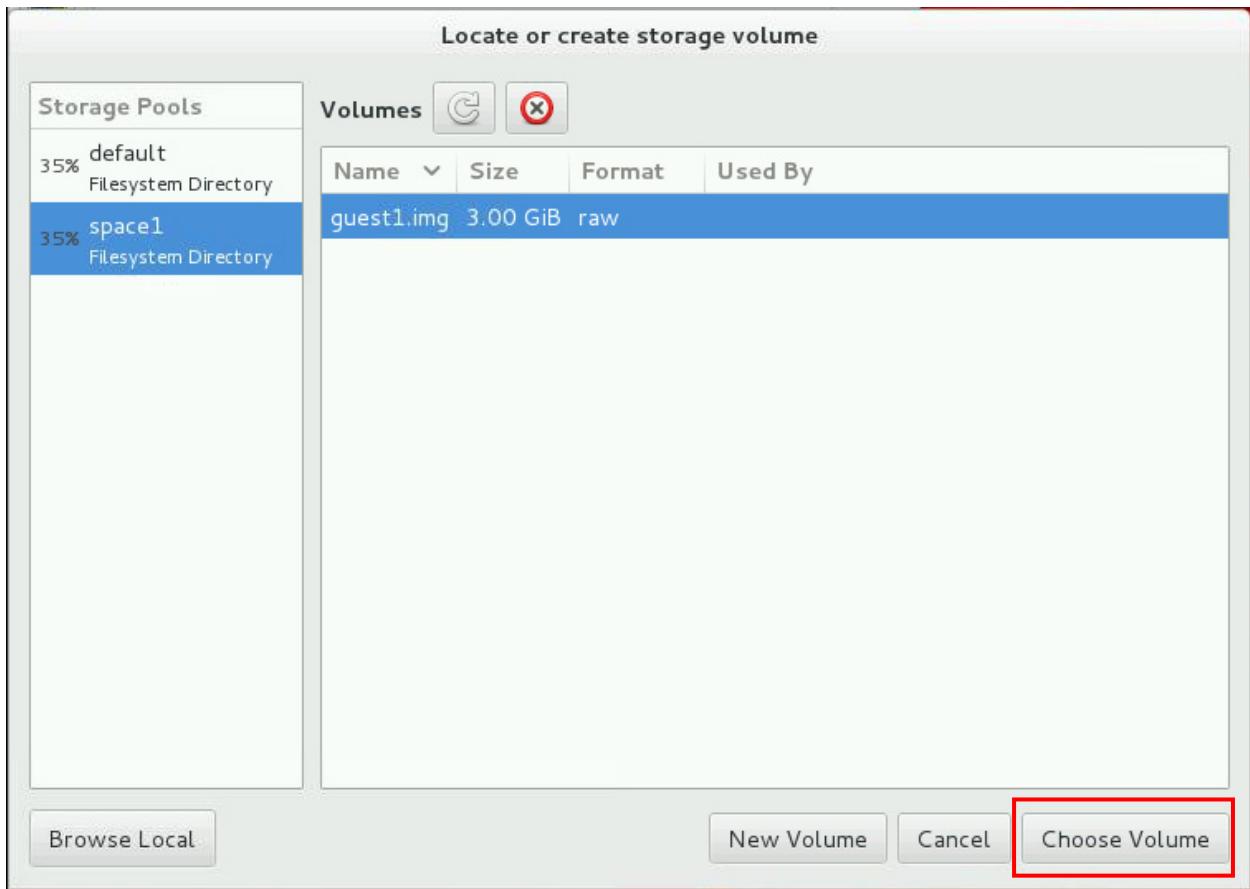
2. In the “Create a new virtual machine, Step 1 of 5” window:
  - a. Click the “Import existing disk image” radio button.
    - “Step 1 of 5” changes to “Step 1 of 4.”
  - b. Click the “Forward” button.



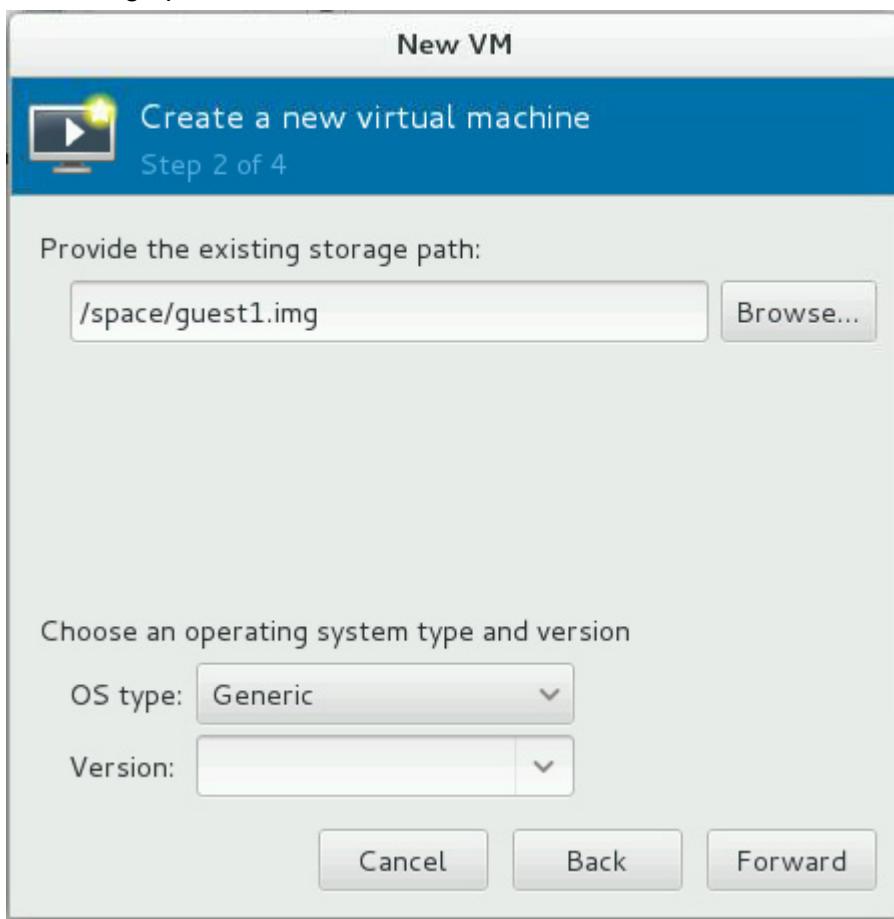
- In the “Create a new virtual machine, Step 2 of 4” window, click the “Browse” button to locate the existing disk image for guest1.



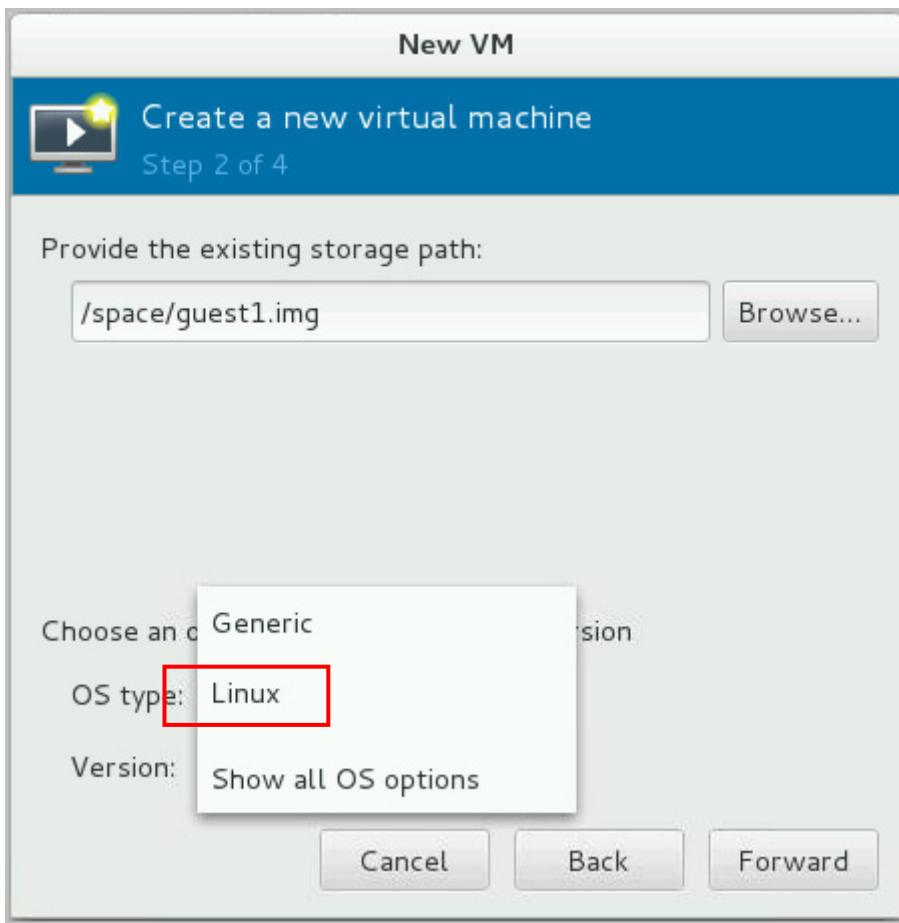
4. In the “Locate or create storage volume” window:
- Click the space1 Storage Pool in the left pane
  - Click the guest1.img Volume in the right pane
  - Click the “Choose Volume” button



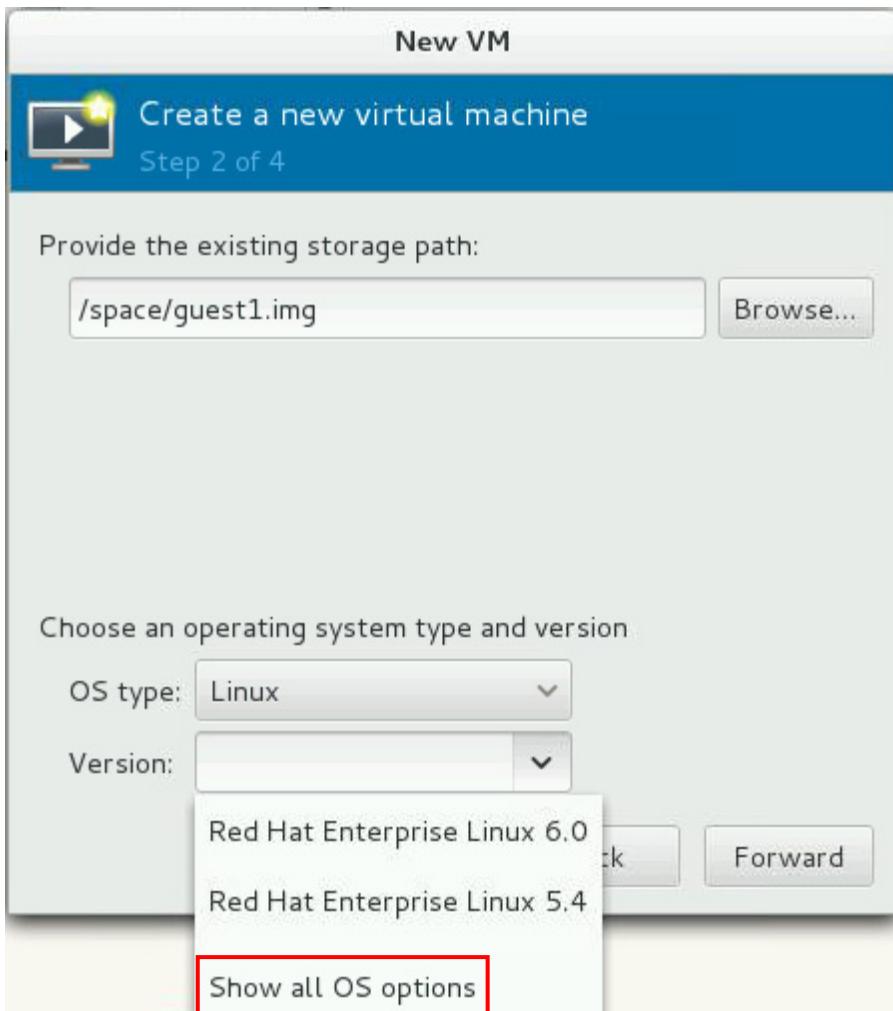
- You are returned to the Step 2 of 4 window, and the selected volume appears in the storage path field.



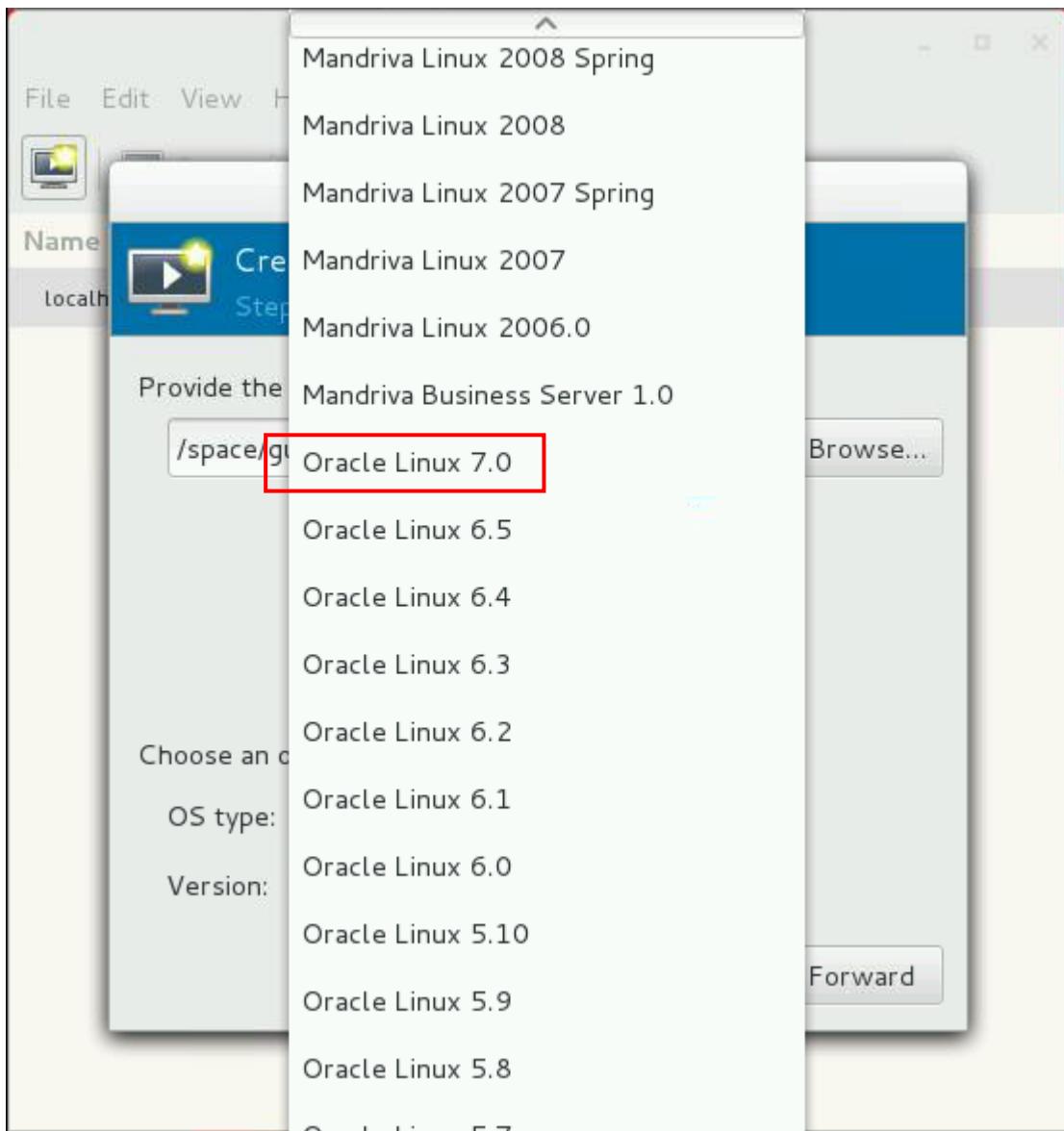
- To choose the appropriate operating system type and version, select “Linux” from the “OS type” drop-down menu.



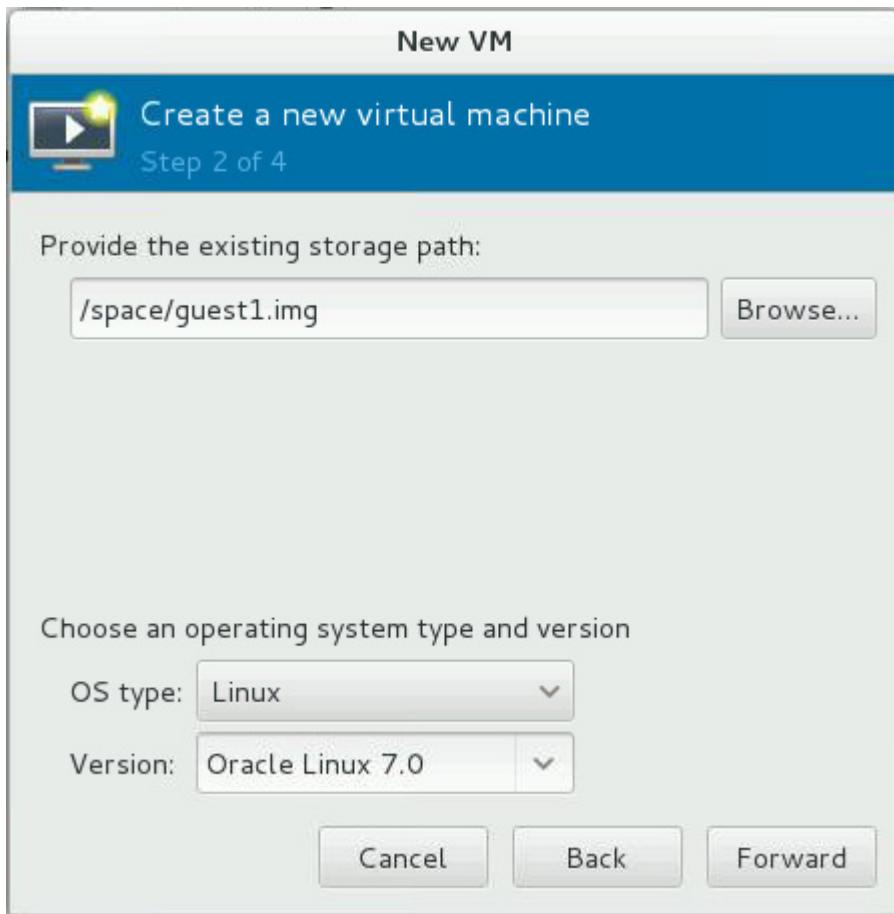
6. Select the OS Version.
  - a. Select the appropriate variant for Linux by clicking “Show all OS options” from the “Version” drop-down menu.



- b. Select “Oracle Linux 7.0” in the “Version” menu.
- Click the drop-down menu to view the version list.
  - Position your mouse pointer on the down arrow key at the bottom of the list to scroll to Oracle Linux 7.0.



- After making your selections, the window appears as shown.

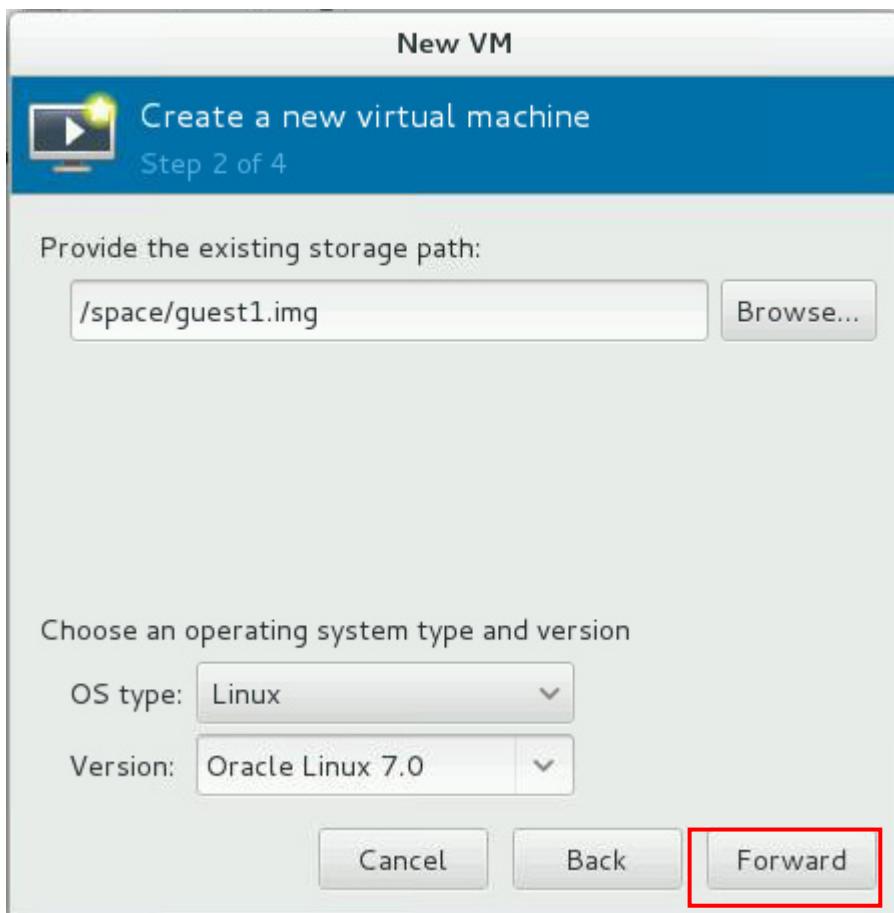


- Run the following command from a terminal window to display the list of available OS versions.

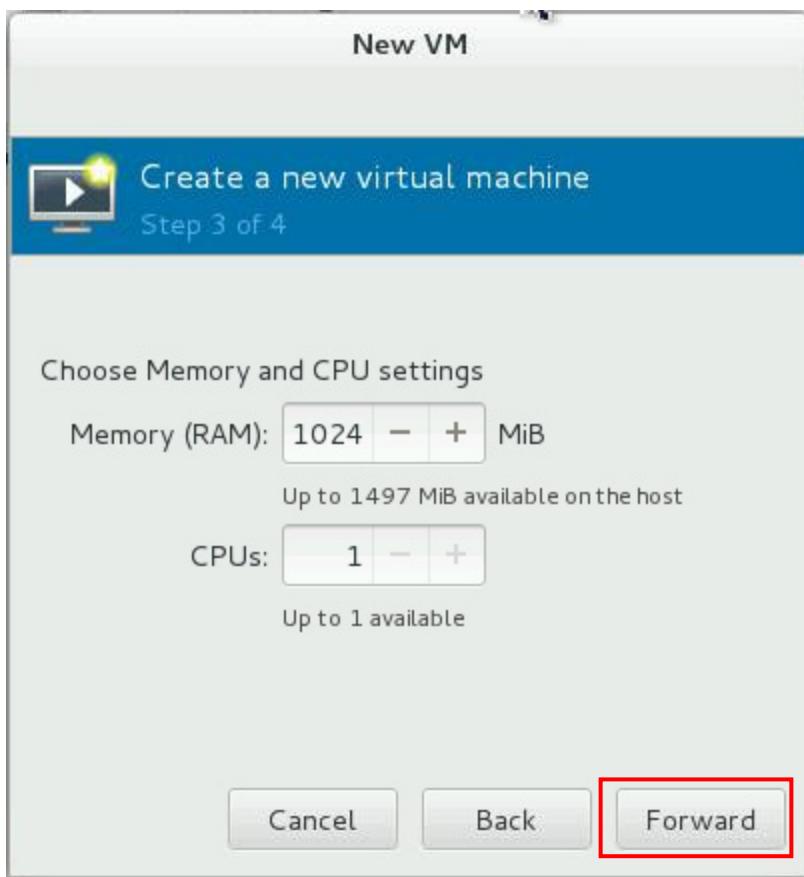
```
# osinfo-query os | grep Oracle
ol5.0          | Oracle Linux 5.0
| 5.0          | http://linux.oracle.com/5.0
ol5.1          | Oracle Linux 5.1
| 5.1          | http://linux.oracle.com/5.1
ol5.10         | Oracle Linux 5.10
| 5.10         | http://linux.oracle.com/5.10
ol5.2          | Oracle Linux 5.2
| 5.2          | http://linux.oracle.com/5.2
ol5.3          | Oracle Linux 5.3
| 5.3          | http://linux.oracle.com/5.3
ol5.4          | Oracle Linux 5.4
| 5.4          | http://linux.oracle.com/5.4
ol5.5          | Oracle Linux 5.5
| 5.5          | http://linux.oracle.com/5.5
ol5.6          | Oracle Linux 5.6
| 5.6          | http://linux.oracle.com/5.6
ol5.7          | Oracle Linux 5.7
| 5.7          | http://linux.oracle.com/5.7
```

ol5.8	Oracle Linux 5.8
5.8	http://linux.oracle.com/5.8
ol5.9	Oracle Linux 5.9
5.9	http://linux.oracle.com/5.9
ol6.0	Oracle Linux 6.0
6.0	http://linux.oracle.com/6.0
ol6.1	Oracle Linux 6.1
6.1	http://linux.oracle.com/6.1
ol6.2	Oracle Linux 6.2
6.2	http://linux.oracle.com/6.2
ol6.3	Oracle Linux 6.3
6.3	http://linux.oracle.com/6.3
ol6.4	Oracle Linux 6.4
6.4	http://linux.oracle.com/6.4
ol6.5	Oracle Linux 6.5
6.5	http://linux.oracle.com/6.5
ol7.0	Oracle Linux 7.0
7.0	http://linux.oracle.com/7.0

- d. After selecting the “OS type” as “Linux” and the “Version” as “Oracle Linux 7.0,” click the “Forward” button.

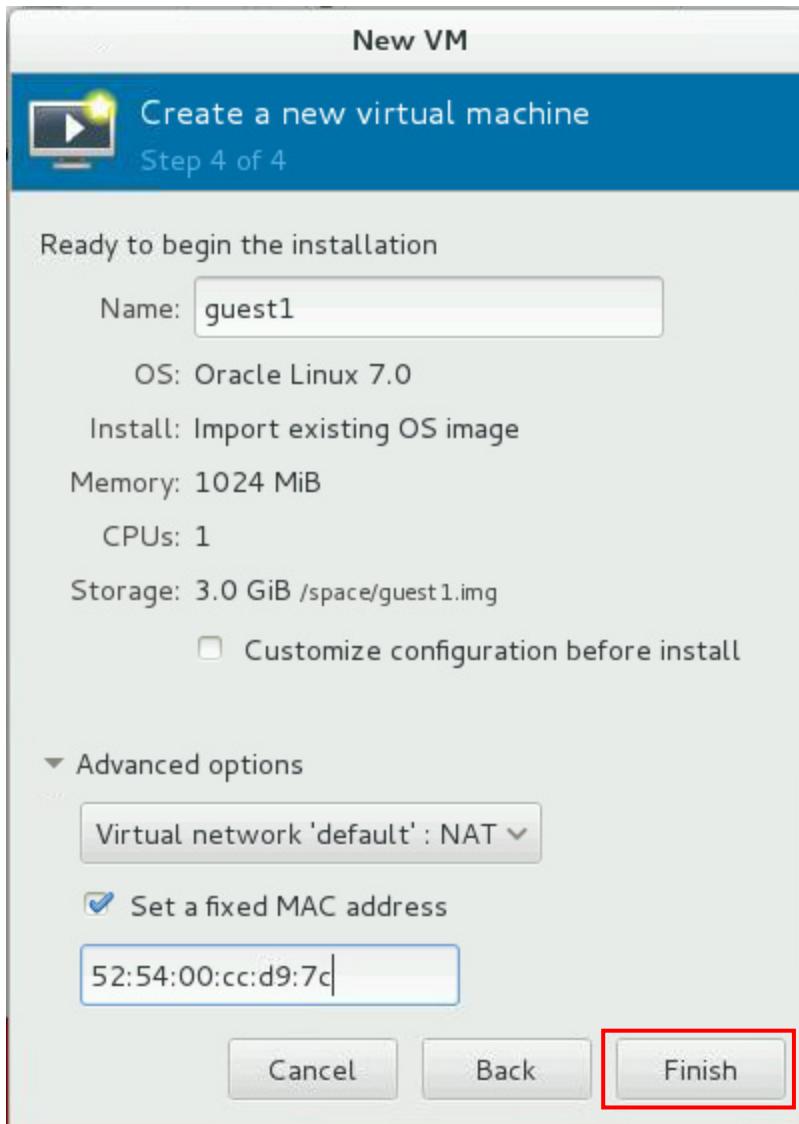


7. In the “Create a new virtual machine, Step 3 of 4” window, accept the default Memory size and CPUs number for the new virtual machine.



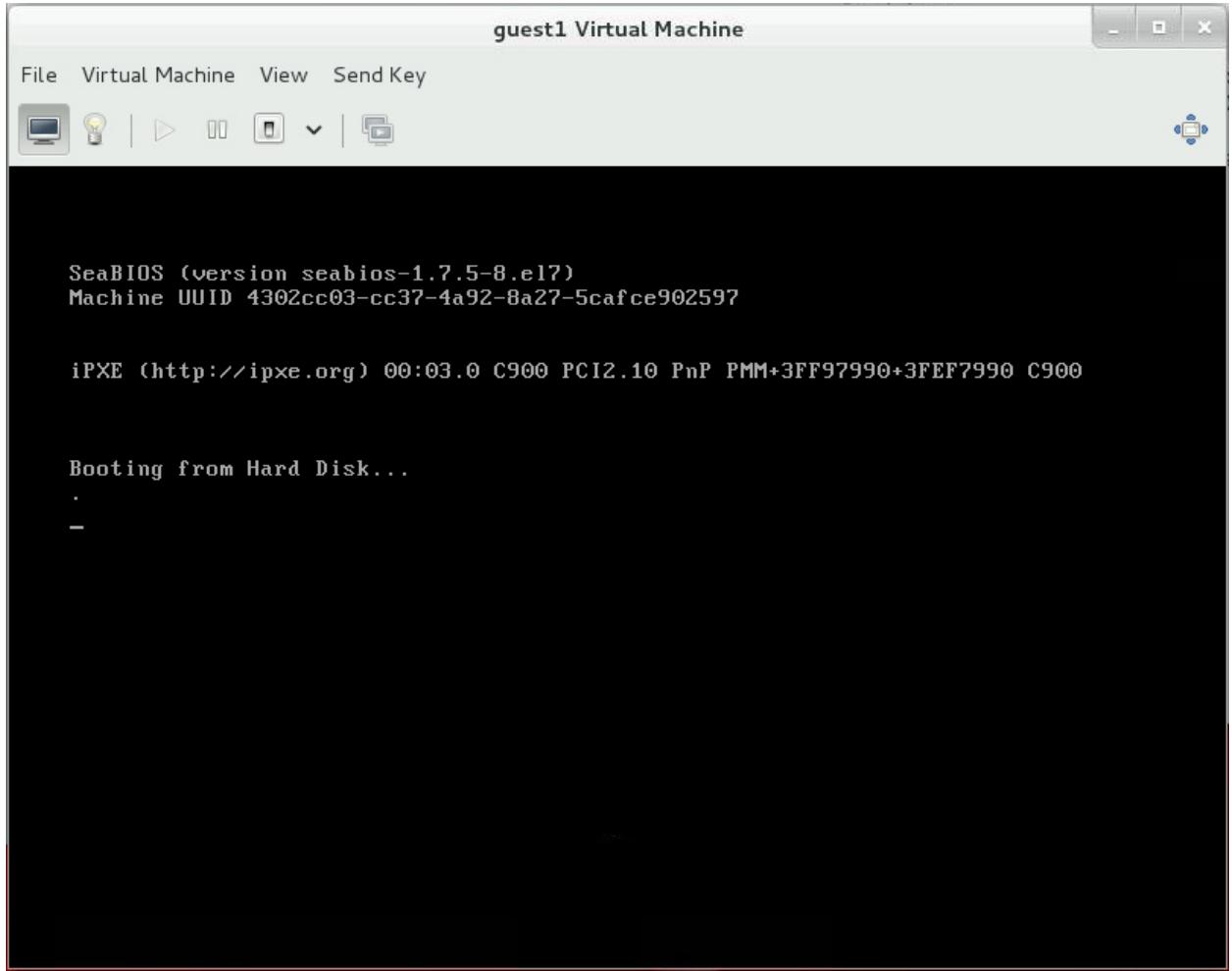
- Click “Forward” to continue.

8. In the “Create a new virtual machine, Step 4 of 4” window:
- Change the Name to guest1.
  - Click the expand button to display “Advanced options.”
  - Change the MAC address to 52:24:00:cc:d9:7c.
    - This is the MAC address of the Ethernet interface of the disk image from which you are creating a virtual machine.
    - Use this MAC address when you create the guest, otherwise the guest will not acquire an IP address through NAT.



- Click “Finish” to complete the creation of your virtual machine.
- The virtual machine is created and starts to boot.

- You are automatically connected to the virtual machine console.



- In your practice environment, your virtual machine is not taking advantage of the acceleration provided by KVM. This limitation of your practice environment is discussed in the Practices Overview section. Without the KVM acceleration, the boot process is very slow.

In the next practice, you manipulate your virtual machine while it is still booting.

- If you no longer have control of your cursor because it is used by the virtual machine, press the Ctrl + Alt key combination to regain control of the cursor.

## Practice 14-4: Managing Your Virtual Machine

### Overview

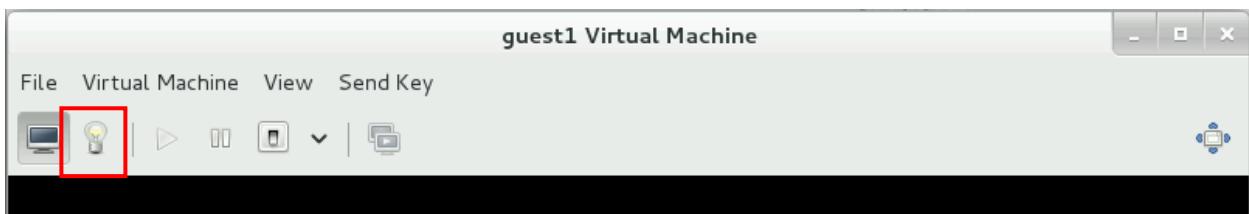
In this practice, you perform various operations on your virtual machine.

### Assumptions

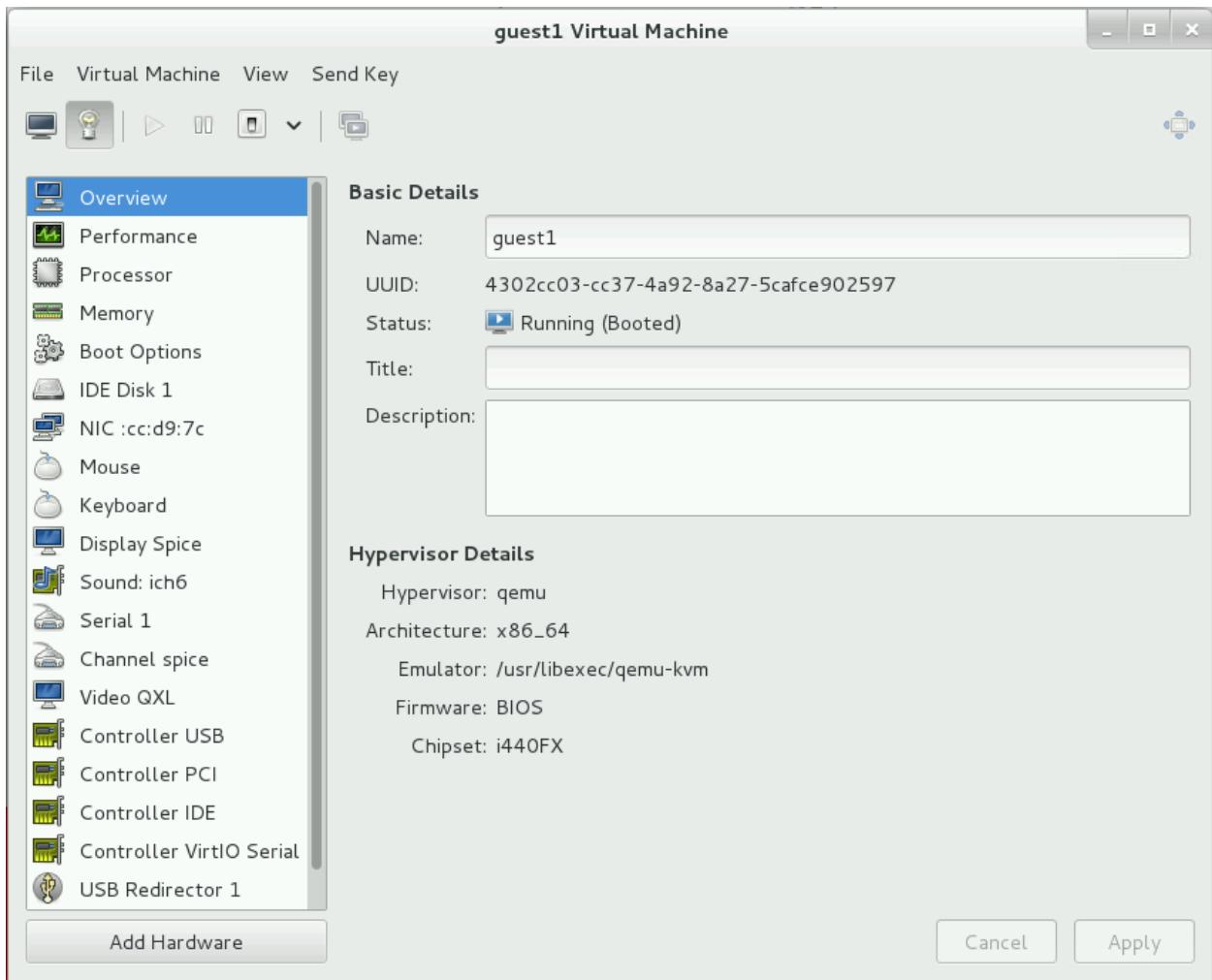
This practice assumes that you have the virtual machine console window opened, as shown in the last step of the previous practice.

### Tasks

1. Using the Virtual Machine Manager, display information about your guest1 virtual machine.
  - a. In the console window of your virtual machine, click the information icon to show the virtual hardware details.

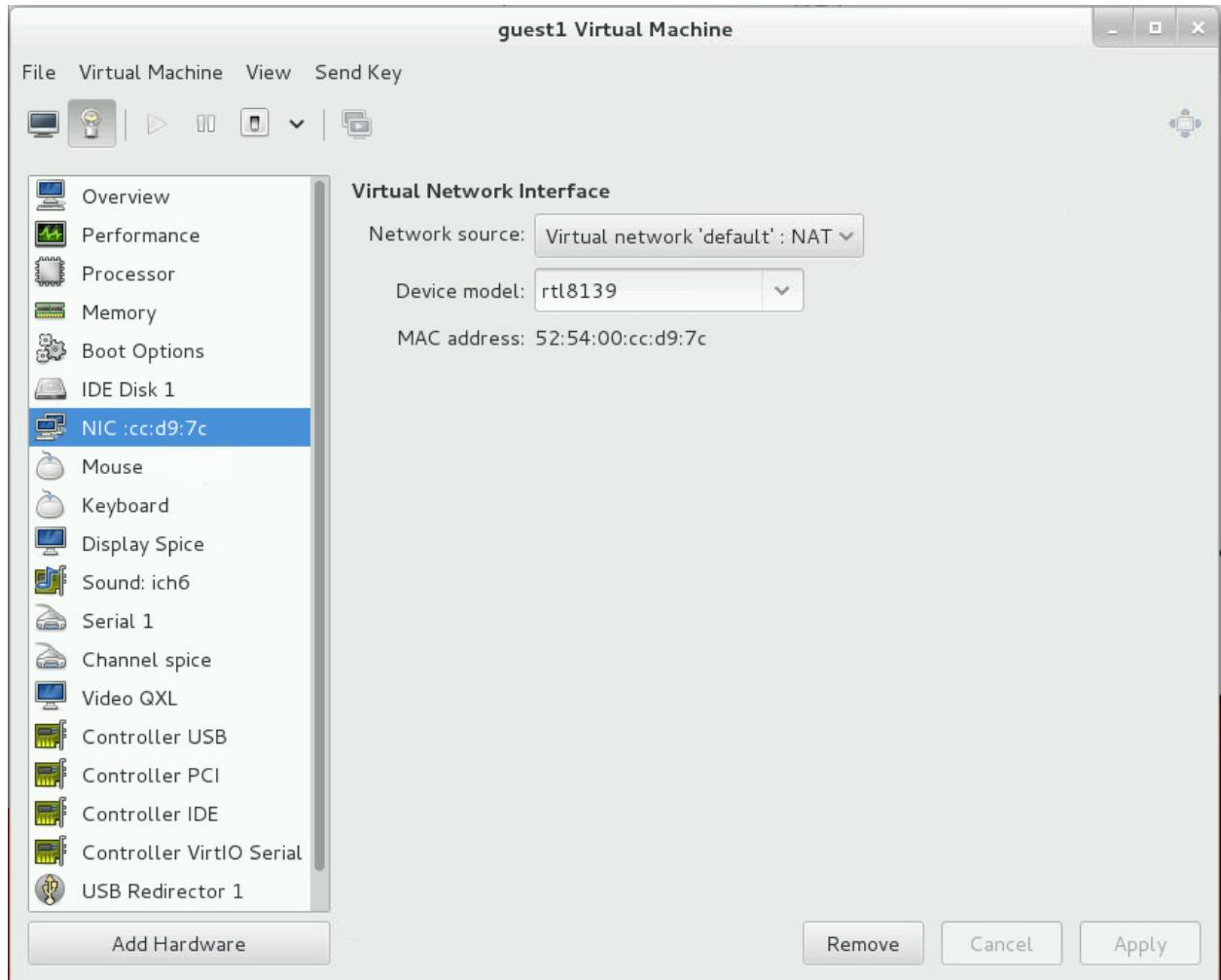


- The Overview perspective appears.



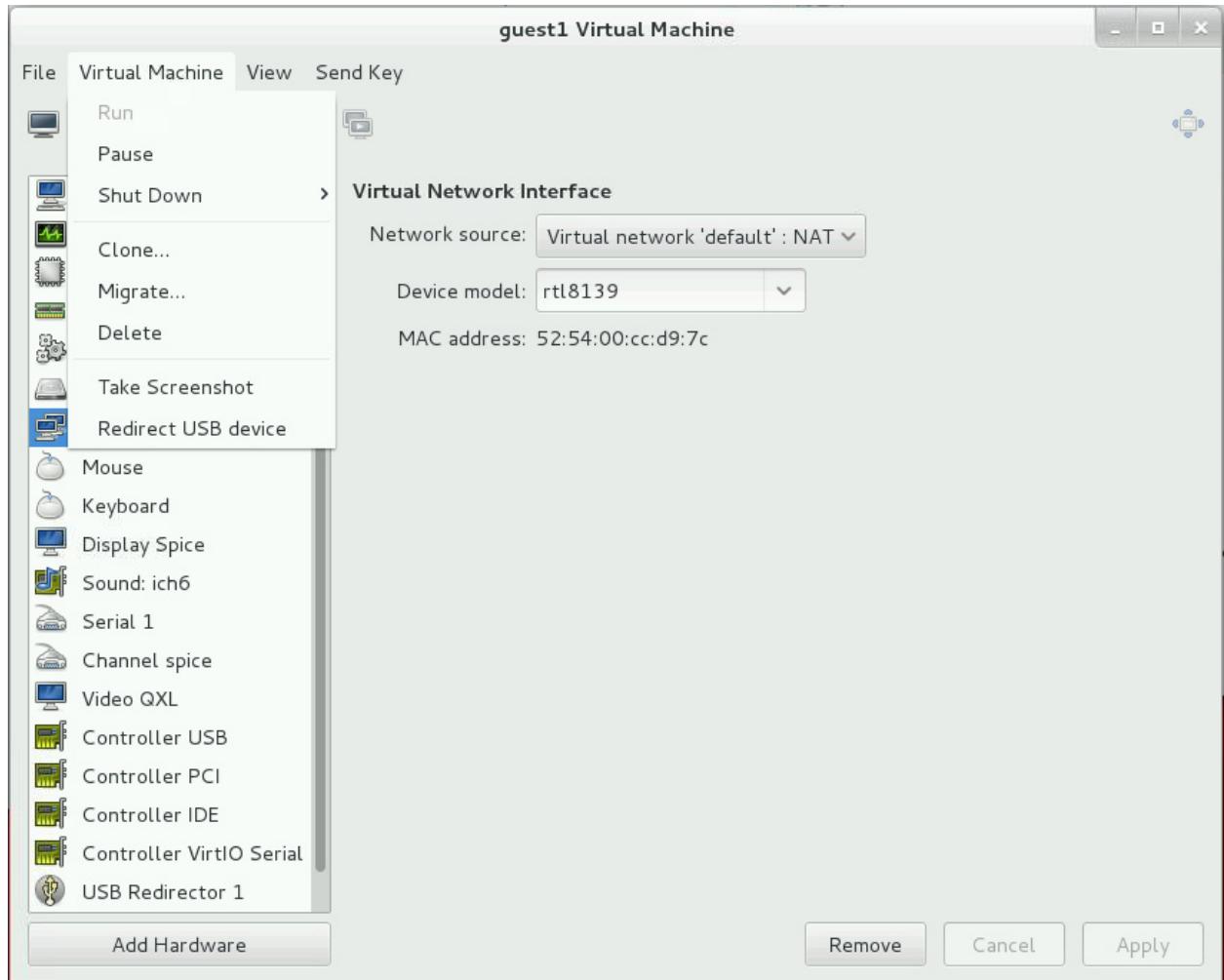
- Note the following items:
  - The UUID located in the Basic Details section. The UUID is a unique identification number that is assigned automatically to your virtual machine.
  - The status of the virtual machine: Running (Booted)
  - The hypervisor, which shows as `qemu`. If your virtualization host supports virtualization hardware assist, the hypervisor shows as `kvm`.

- b. Click the NIC perspective in the left pane.

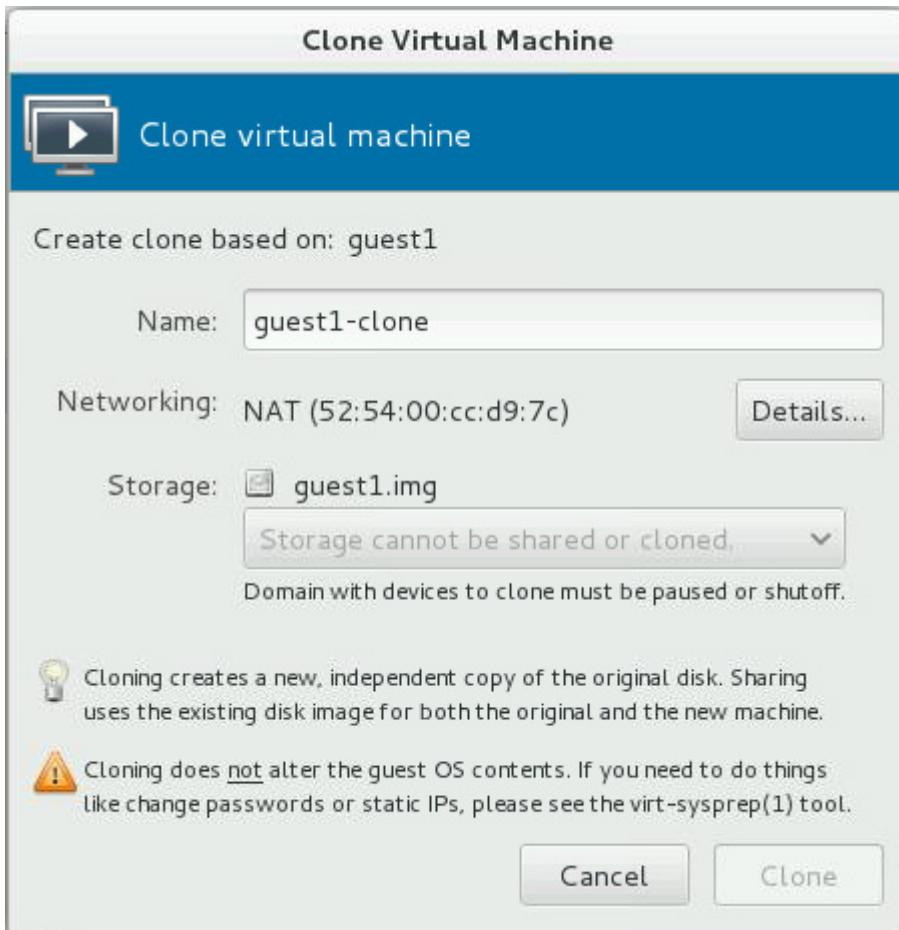


- The single NIC is connected to the virtual network called default.
  - The Device Model indicates rtl8139. You change this setting later in this practice.
- c. Click other perspectives and examine the information available from these perspectives.
- d. Leave the Virtual Machine Details window open.
2. Clone the virtual machine guest1.
- The cloning operation creates a new virtual machine and assigns it a new MAC address and a new UUID. The clone has a virtual disk that is a copy of the virtual disk owned by the source virtual machine used for the cloning operation.
  - Before cloning a virtual machine, you install all the applications and perform the necessary configuration in the operating system of the virtual machine. After customizing your virtual machine, you remove machine-specific information from the virtual machine so that clones of the virtual machine acquire their own identity. You can find more information about this process at this location:  
[https://access.redhat.com/documentation/en-US/Red\\_Hat\\_Enterprise\\_Virtualization/3.5/html/Administration\\_Guide/sect-Sealing\\_Virtual\\_Machines\\_in\\_Preparation\\_for\\_Deployment\\_as\\_Templates.html](https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Virtualization/3.5/html/Administration_Guide/sect-Sealing_Virtual_Machines_in_Preparation_for_Deployment_as_Templates.html)

- In this task, you clone the `guest1` virtual machine from the Virtual Machine Manager. You can also use the `virt-clone` command to create a clone from an existing virtual machine.
  - a. From the Virtual Machine Details window, either showing the console or virtual machine information, select “Virtual Machine > Clone” from the menu bar.

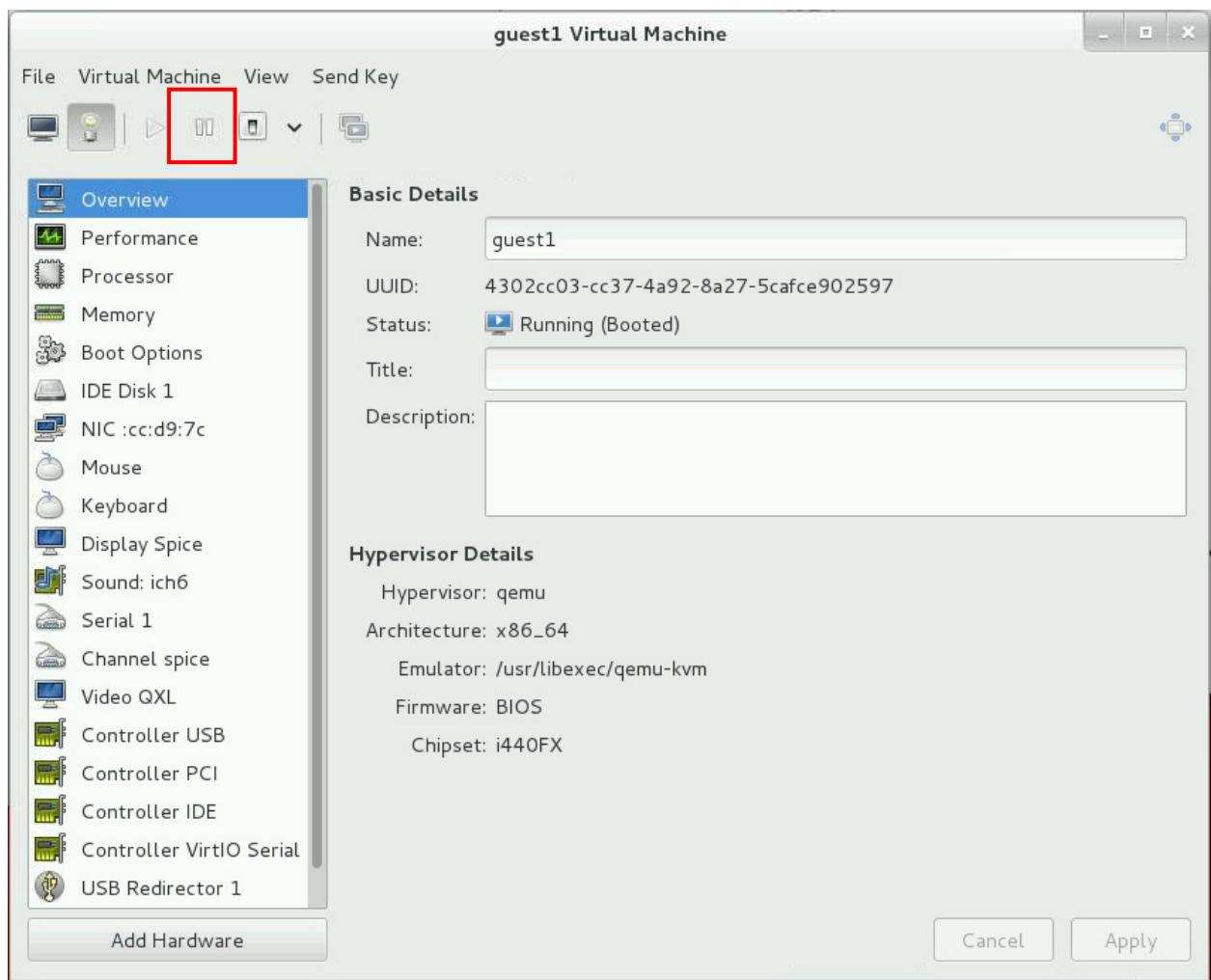


- The Clone Virtual Machine window appears.

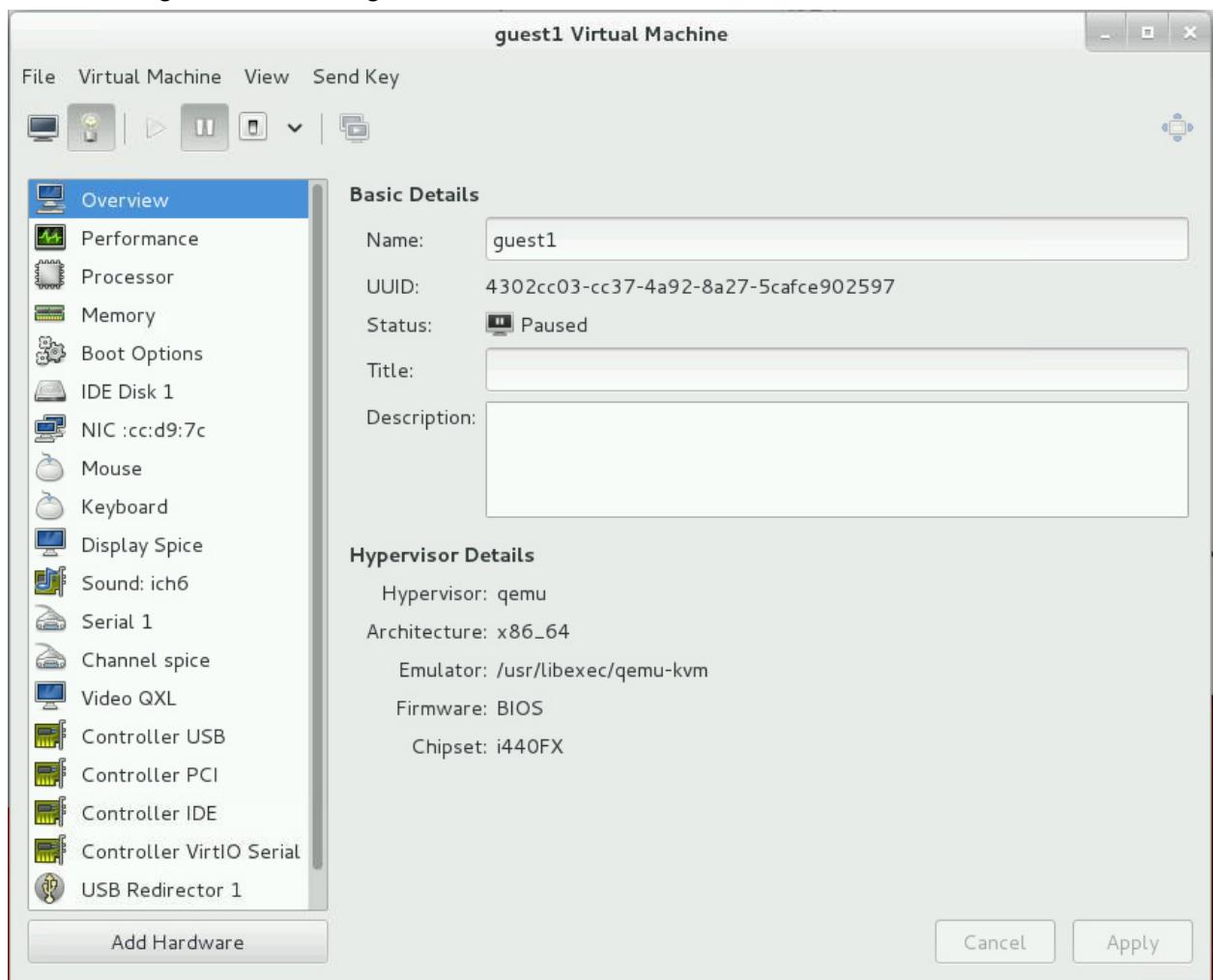


- You cannot clone the virtual machine, because it is in a running state. You must either pause it or shut it down to clone it.
  - To capture a valid state when cloning a virtual machine, it is preferable to pause the virtual machine when it is idle. Because your virtual machine is still booting, the clone created with the following operation might not boot properly.
  - The Red Hat Enterprise Virtualization solution offers hot cloning, which means that you can clone a running virtual machine.
- b. Click the Cancel button on the Clone Virtual Machine window.

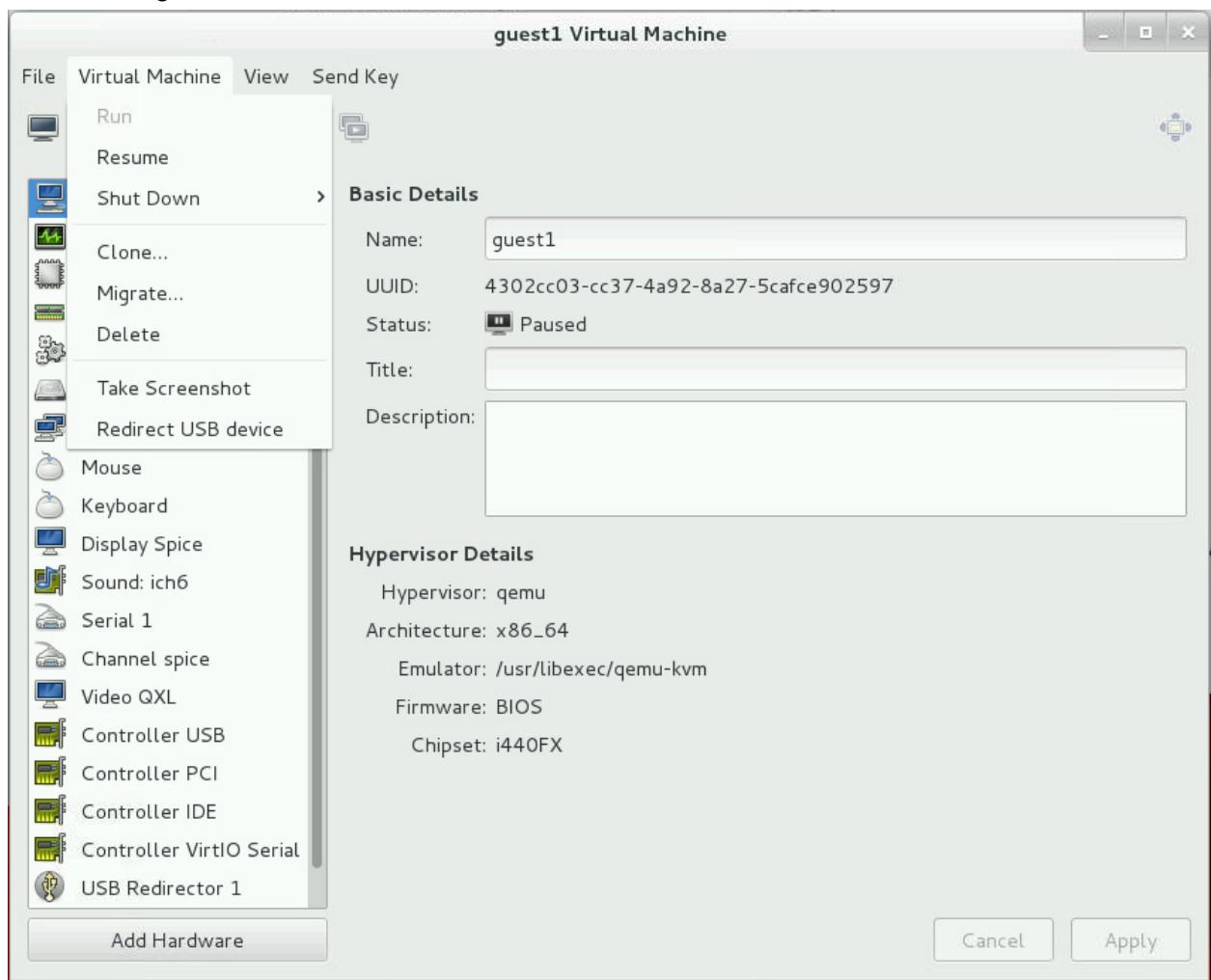
- c. From the Virtual Machine Details window, either showing the console or virtual machine information, click the Pause button on the toolbar.



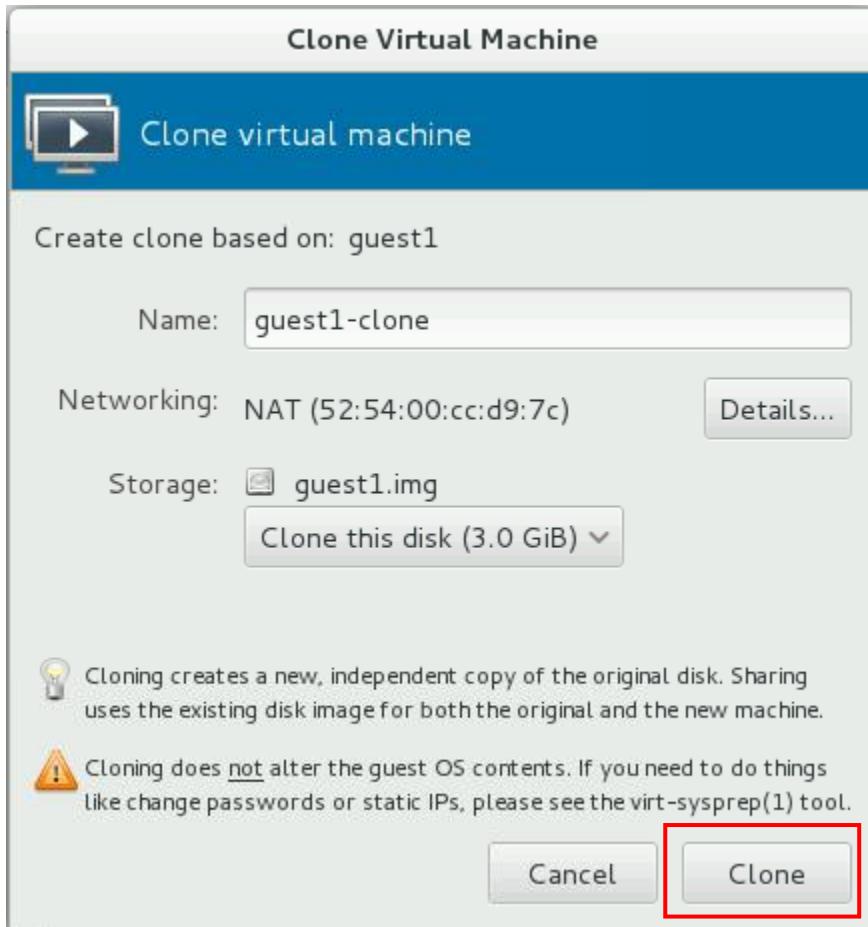
- Click the Overview perspective and note that the status of the virtual machine changes from Running to Paused.



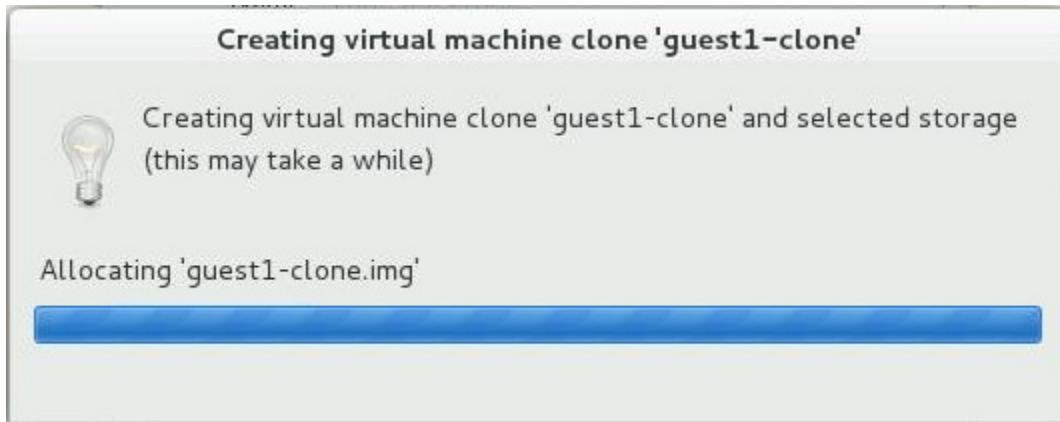
- d. From the Virtual Machine Details window, attempt to clone the virtual machine again by selecting “Virtual Machine > Clone” from the menu bar.



- e. Click the Clone button in the Clone Virtual Machine window.



- The cloning operation starts and takes a few seconds to complete.

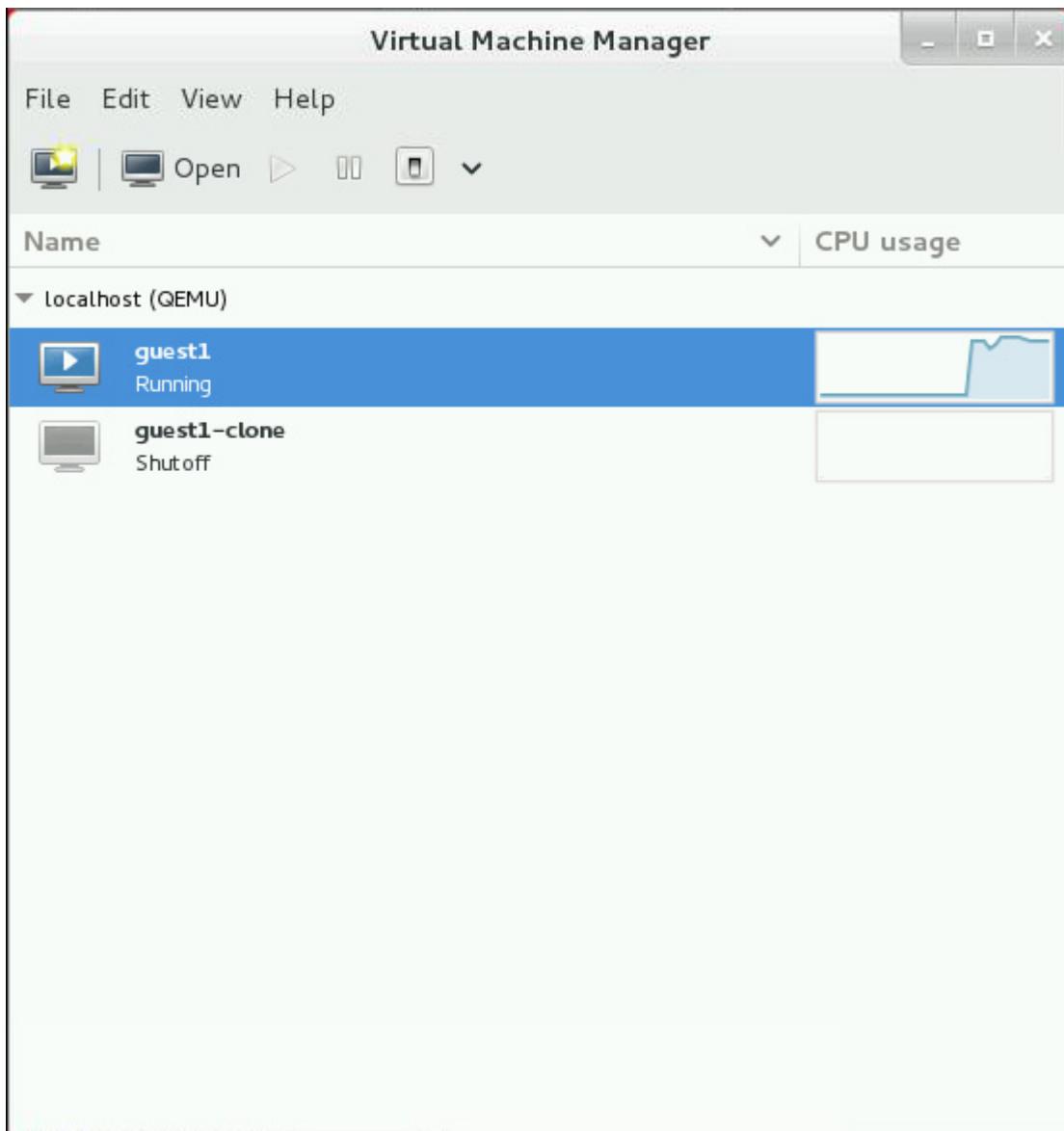


- f. After the cloning completes, select “File > View Manager” from the menu bar to return to the main Virtual Machine Manager window.
- The window displays the new clone.
  - The new clone, called guest1-clone, is independent from guest1, the virtual machine used to create the clone. You can change the parameters and options for the clone without affecting the virtual machine used for the cloning operation.
- g. Resume operation for the guest1 virtual machine

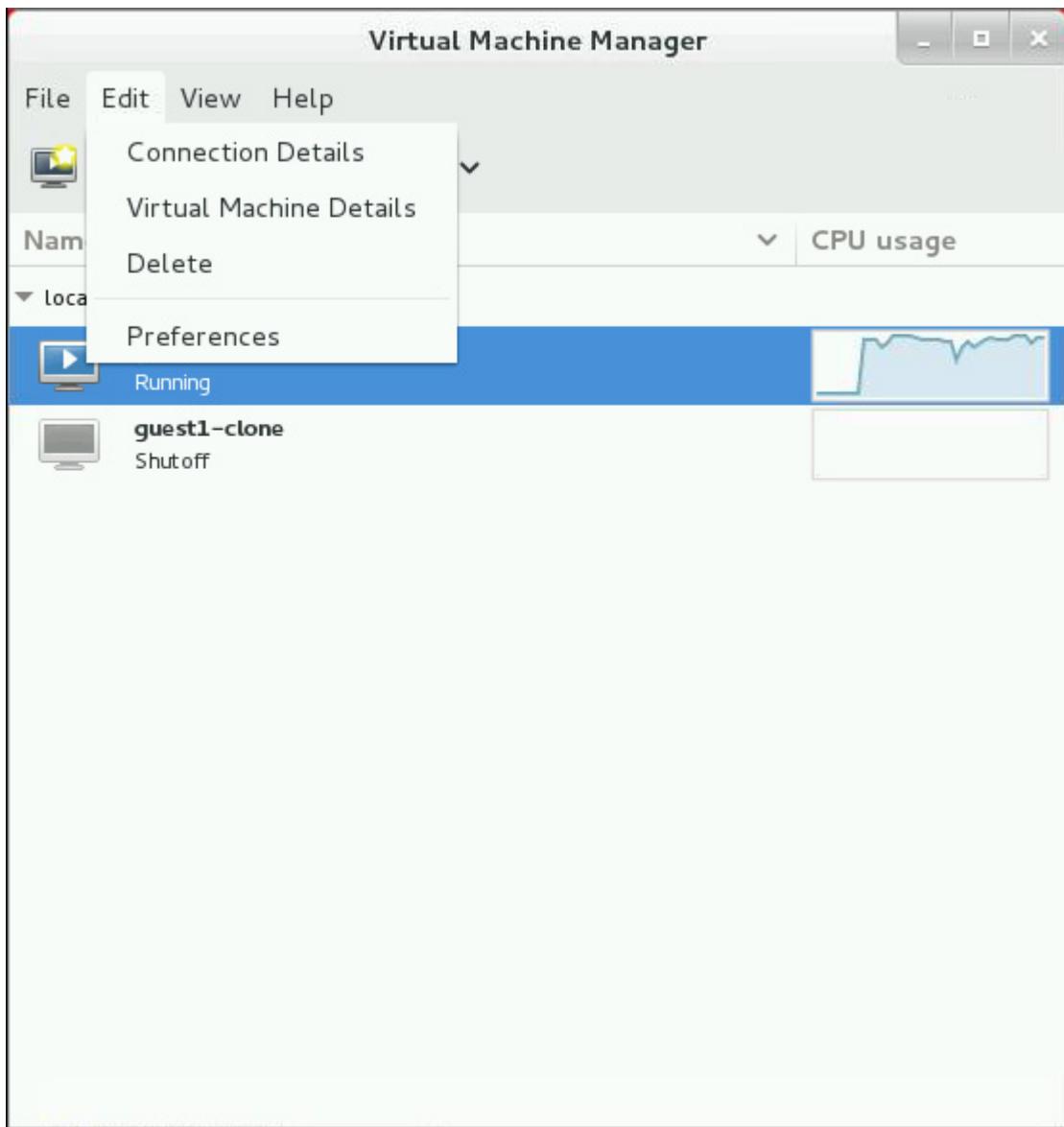
- From a terminal window on **host05**, execute the following command: `virsh resume guest1`.

```
# virsh resume guest1
Domain guest1 resumed
```

- The `guest1` virtual machine resumes its booting.

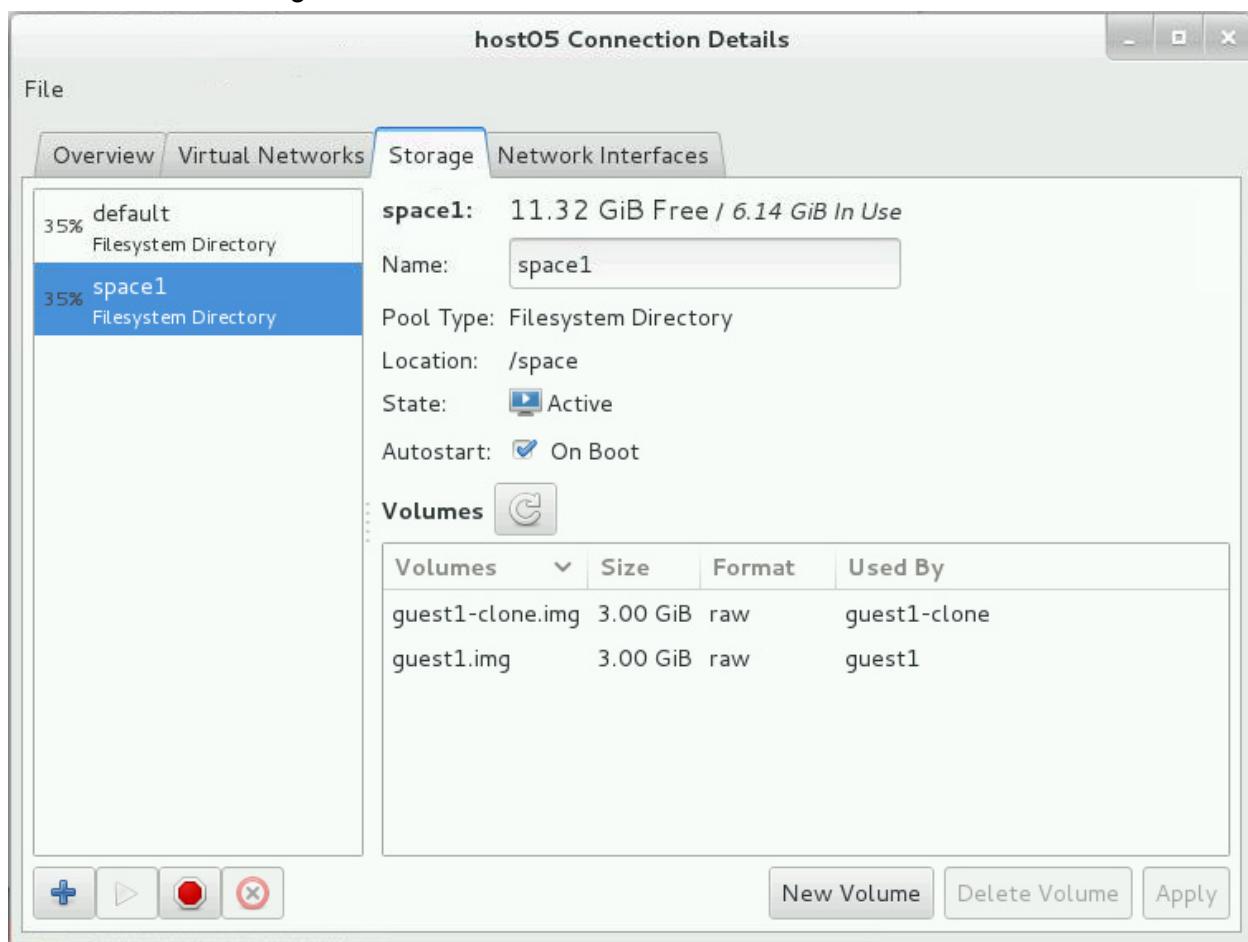


- i. Select “Edit > Connection Details” in the menu bar.



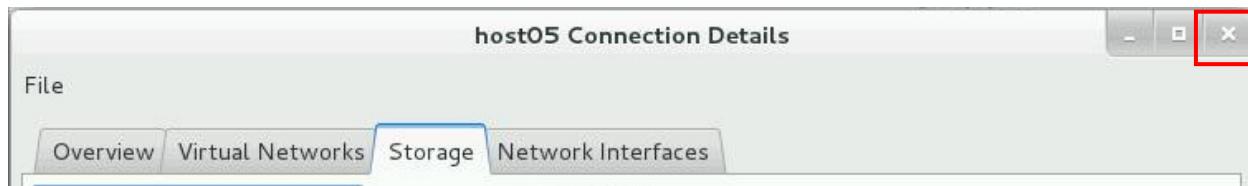
- j. In the **host05** Connection Details window:

- Click the Storage tab.



- The virtual disk for the clone appears in the Volumes pane.

- k. Click the X in the top right corner to close the host05 Connection Details window.



3. Add paravirtualization for the virtual machine.

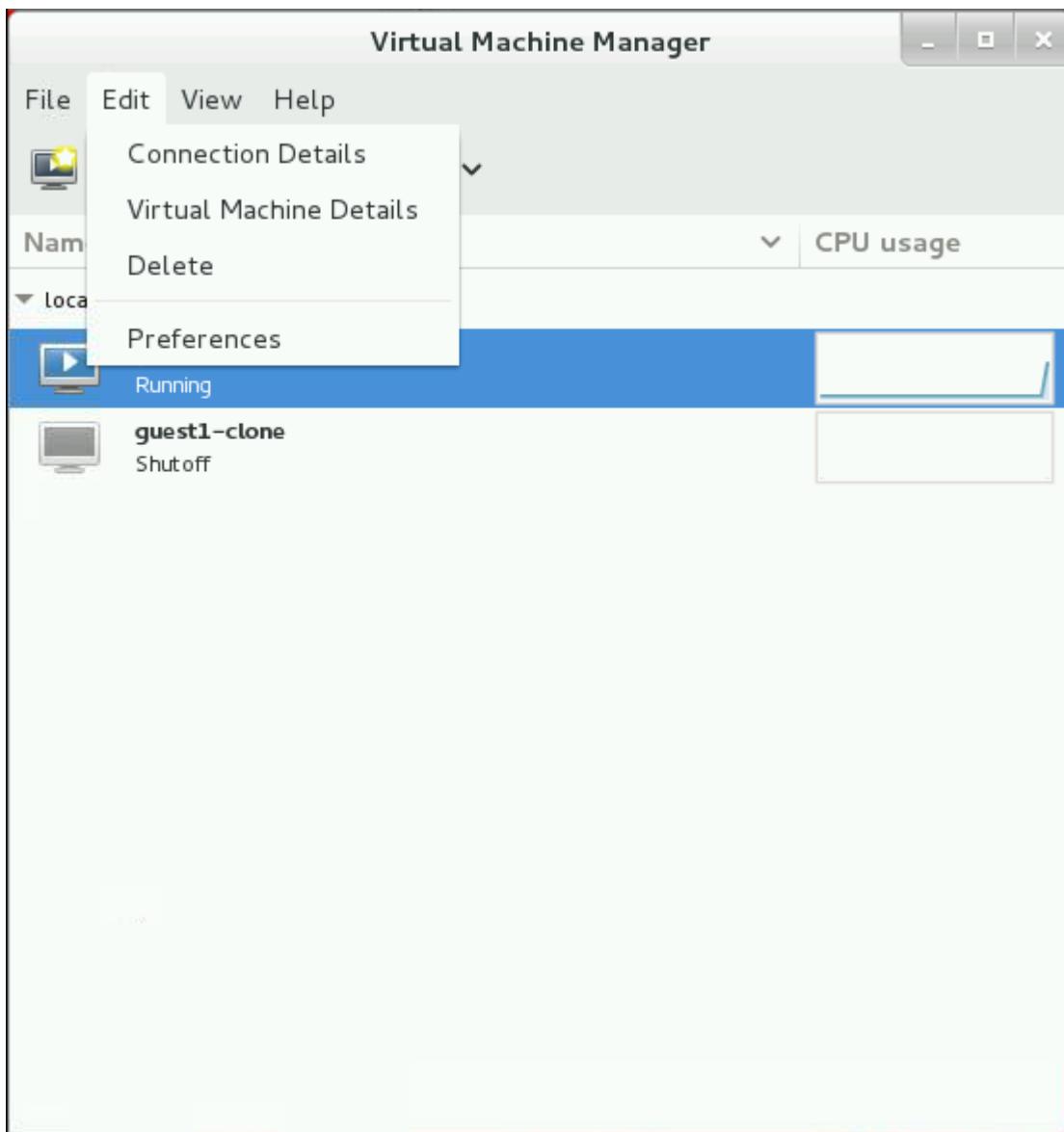
- In this task, you add paravirtualization support for the virtual NIC in the `guest1` virtual machine. A paravirtual driver is virtualization aware and cooperates with the hypervisor for improved performance.
- Dump the virtual machine configuration file by using the `virsh dumpxml` command.

```
# virsh dumpxml guest1
<domain type='qemu' id='1'>
...
<interface type='network'>
    <mac address='52:54:00:cc:d9:7c' />
    <source network='default' bridge='virbr0' />
```

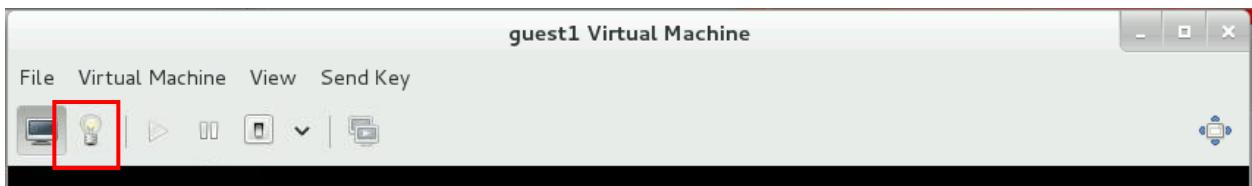
```

<target dev='vnet0' />
<model type='rtl8139' />
<alias name='net0' />
<address type='pci' domain='0x0000' bus='0x00'
slot='0x03' function='0x0' />
</interface>
...
</domain>
```

- The configuration file for the guest1 virtual machine is in XML format.
  - Note the network interface element of type network shown in the previous display. The NIC operations are fully emulated.
- b. In the Virtual Machine Manager main window, highlight guest1. Select “Edit > Virtual Machine Details” in the menu bar.

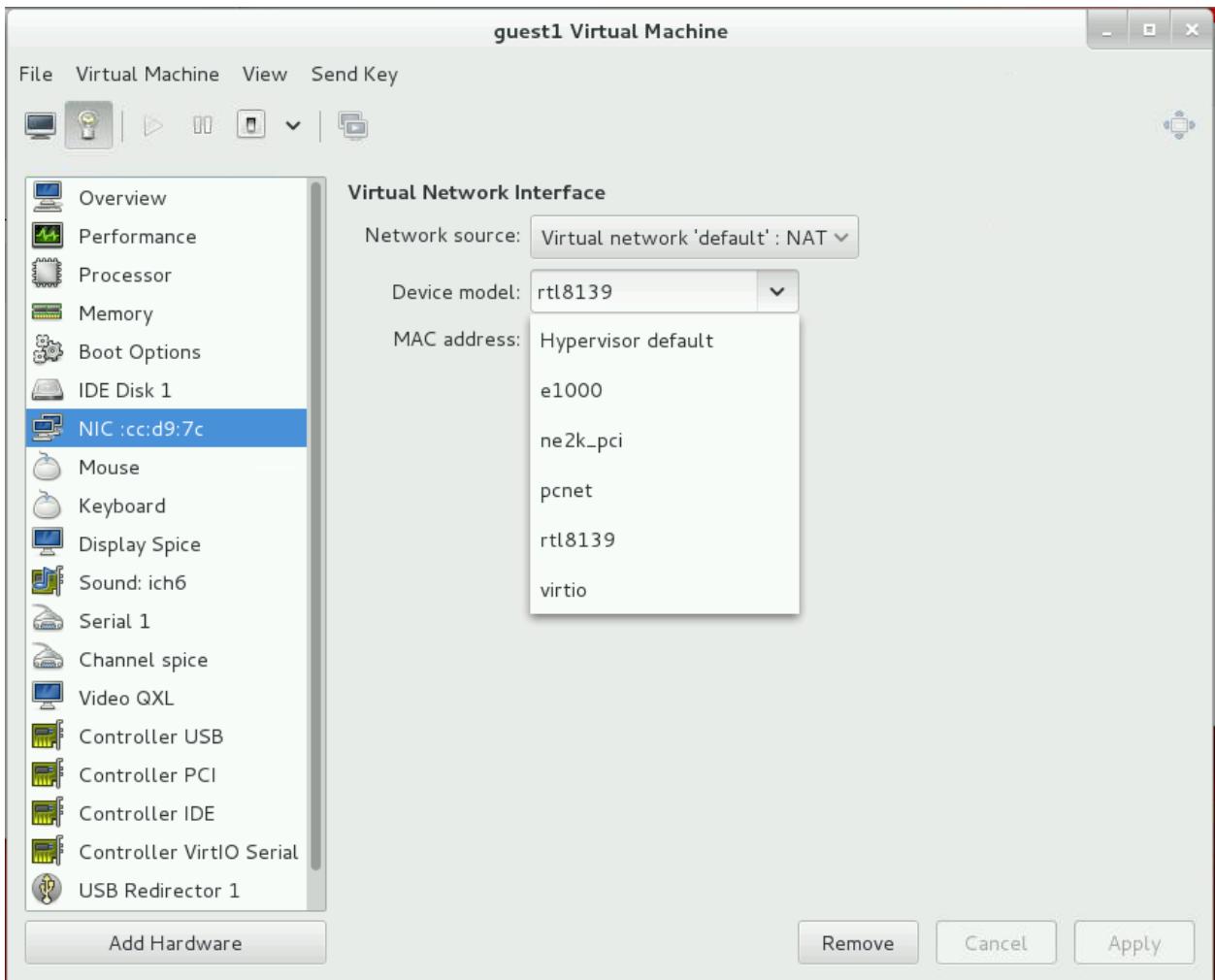


- c. If you get the console view, click the information icon in the toolbar.

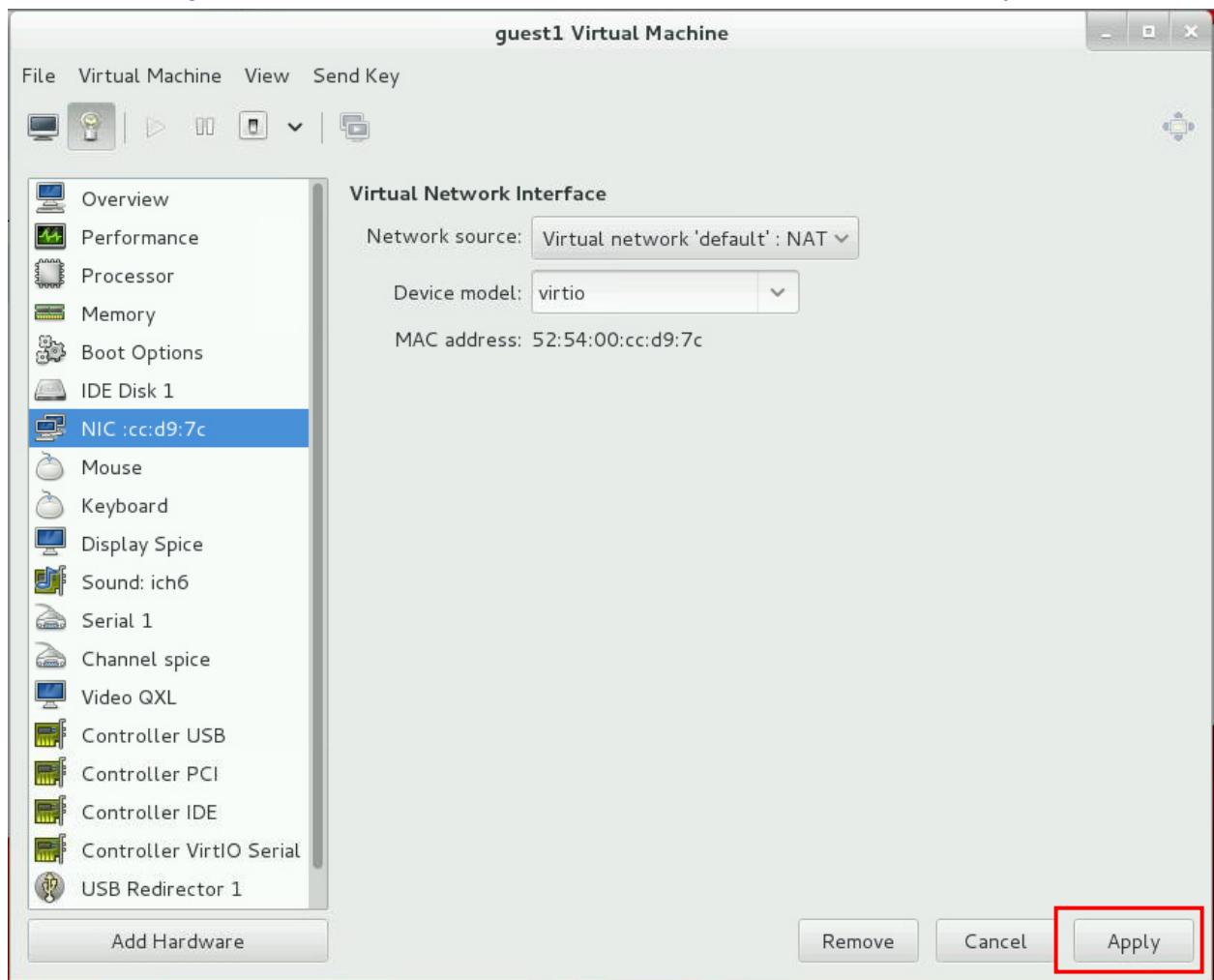


d. In the guest1 Virtual Machine window, perform the following steps:

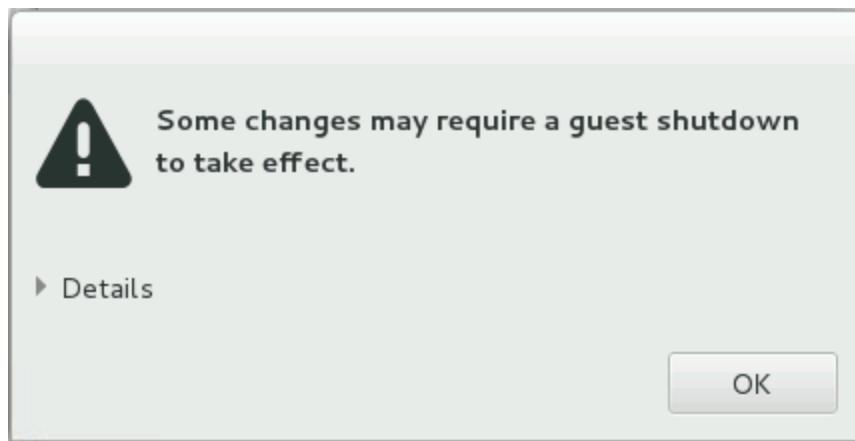
1. Highlight the NIC in the left pane.
2. Click the spinbox for the Device Model.
3. Change the Device Model from Hypervisor default to virtio.



- e. Click the Apply button to confirm your change.
- You might have to maximize the Virtual Machine window to see the Apply button.



- The following window appears. Click OK.



- f. In your terminal window on **host05**, dump the virtual machine configuration file again with the `virsh dumpxml` command.
- The change to the device model for the network interface has not yet taken effect.

```
# virsh dumpxml guest1
<domain type='qemu' id='1'>
...
<interface type='network'>
    <mac address='52:54:00:cc:d9:7c' />
    <source network='default' bridge='virbr0' />
    <target dev='vnet0' />
    <model type='rtl8139' />
    <alias name='net0' />
    <address type='pci' domain='0x0000' bus='0x00'
        slot='0x03' function='0x0' />
</interface>
...
</domain>
```

- g. Use the `cd` command to change to the default location for all virtual machine XM configuration files, `/etc/libvirt/qemu`.
- Use the `ls -l` command to list the contents of the directory.

```
# cd /etc/libvirt/qemu
# ls -l
-rw-----  ...  guest1-clone.xml
-rw-----  ...  guest1.xml
drwx-----  ...  networks
```

- h. Use the `cat` command to display the contents of the `guest1.xml` file.
- Note that this file contains the change to the model type for the `guest1` network interface.

```
# cat guest1.xml
...
<interface type='network'>
    <mac address='52:54:00:cc:d9:7c' />
    <source network='default' />
    <model type='virtio' />
    <address type='pci' domain='0x0000' bus='0x00'
        slot='0x03' function='0x0' />
</interface>
...
</domain>
```

- You can also enable paravirtualization for the disk driver.
- i. In your terminal window on **host05**, use the `virsh list` command to display active domains.

```
# virsh list
  Id  Name           State
  ---
  1   guest1        running
```

- The `virsh` command refers to virtual machines as domains. You can find more information about the life cycle of domains at this location: [http://wiki.libvirt.org/page/VM\\_lifecycle](http://wiki.libvirt.org/page/VM_lifecycle).
- j. Use the `virsh list --all` command to display all domains, regardless of their state.

```
# virsh list --all
  Id  Name           State
  ---
  -   guest1        running
  -   guest1-clone  shut off
```

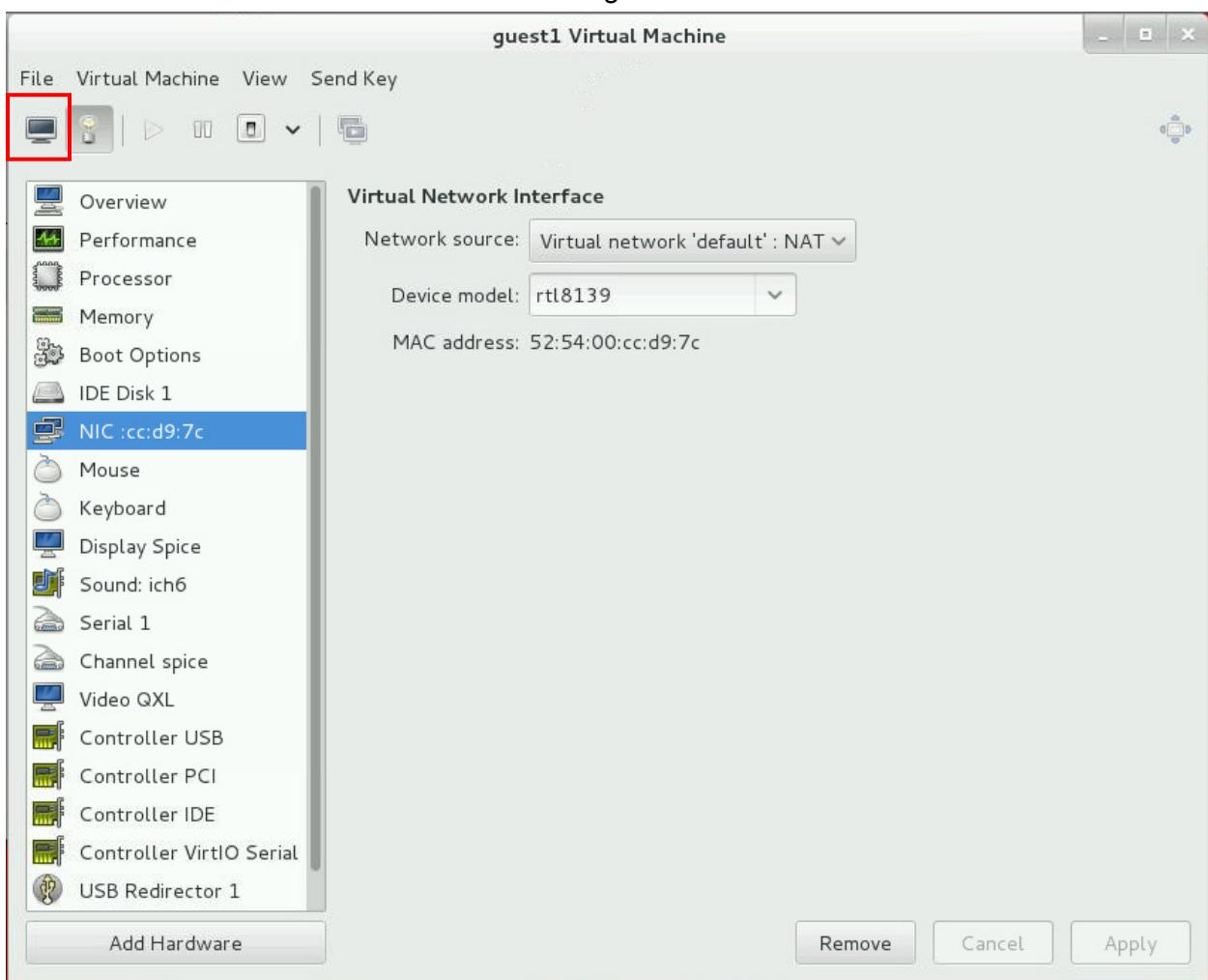
**Note:** As mentioned earlier, in your practice environment your virtual machine is not taking advantage of the acceleration provided by KVM. Without the KVM acceleration, the boot process is very slow.

If the boot process has not completed, you can read through the remaining tasks in this practice, or you can come back to this practice later when the boot process has completed.

You need to eventually shut down **host05** so that you can start the required VMs for the remaining practices. The remaining practices require the following VMs:

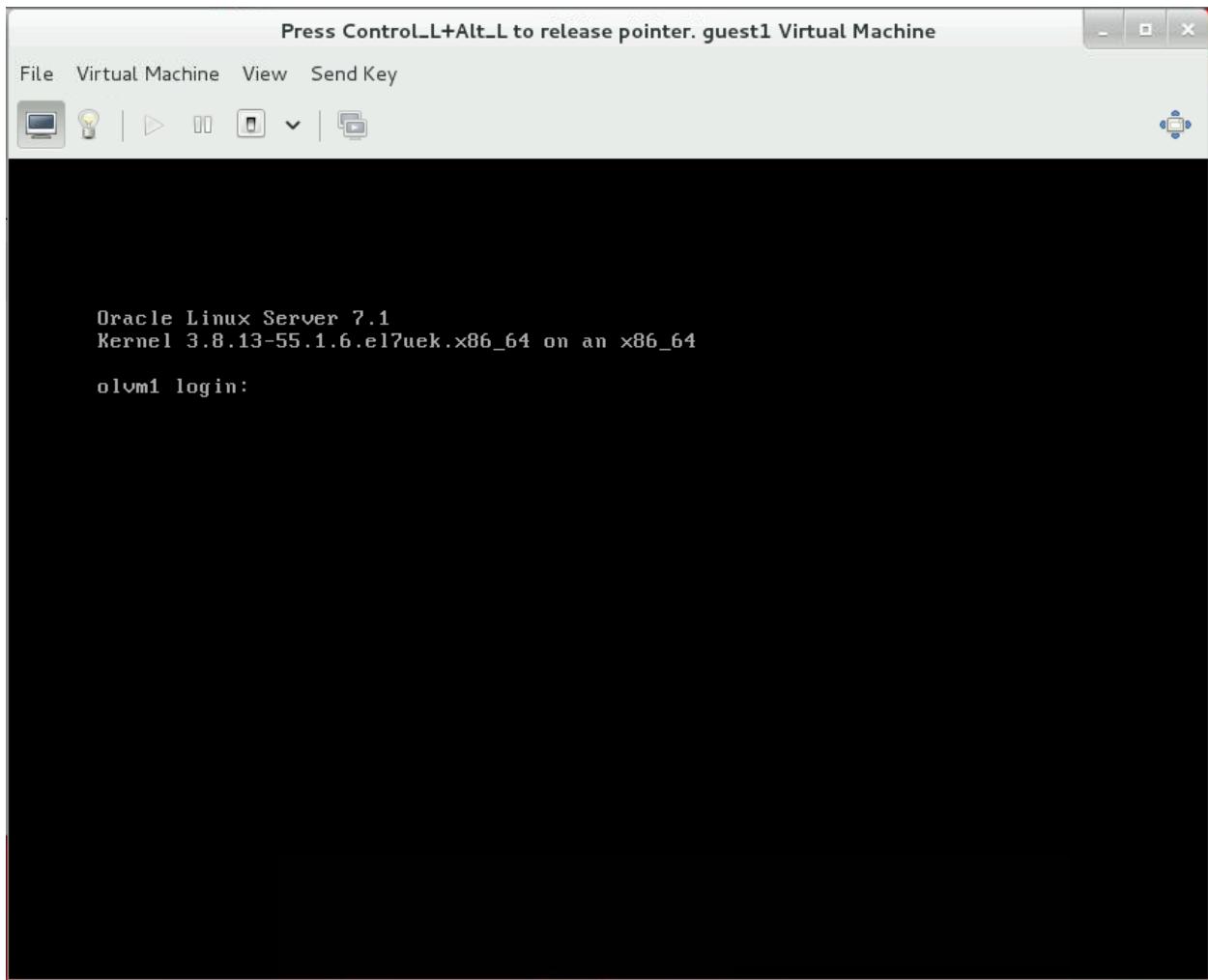
- Practice 15 – Linux Container* uses **host02**
- Practice 16 – Docker* uses **host02**
- Practice 17 – SELinux* uses **host01** and **host03**
  - Practice 17 also uses **host05** again
- Practice 18 – Core Dump Analysis* uses **host03**
- Practice 19 – DTrace* uses **host03**

4. Log in to the **guest1** KVM virtual machine.
  - a. Click the console tool from the Virtual Manager window toolbar.



- b. The console window appears.
  - Press the ENTER key to display the login prompt.
  - Note the login prompt says, "olvml login:".
  - You created the virtual machine by importing an existing OS image and the host name of the existing OS image is **olvml**.

- The name of the virtual machine in KVM does not relate to the host name in the virtual machine. This is also true with Oracle VM Server for x86. You can make them match but that is not a requirement.



c. Log in as the root user.

- The root password is oracle.

```
guest1 login: root
Password: oracle
Last login: ...
[root@guest1 ~]#
```

d. Use the df -h command to list the mounted partitions.

```
# df -h
Filesystem      Size  Used  Avail   Use%  Mounted on
/dev/sda2          ...        /
...
/dev/sda1          ...      /boot
```

- e. Use the `ip addr` command to display the network interface configuration.
- If the `ens3` interface does not have an IP address, then the boot process is not completely finished.
  - Wait a few seconds then run the `ip addr` command again to see the IP address.

```
# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state ...
...
2: ens3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc ...
    Link/ether 52:54:00:cc:d9:7c brd ff:ff:ff:ff:ff:ff
    inet addr:192.168.122.254/24 brd 192.168.122.255 scope ...
...
...
```

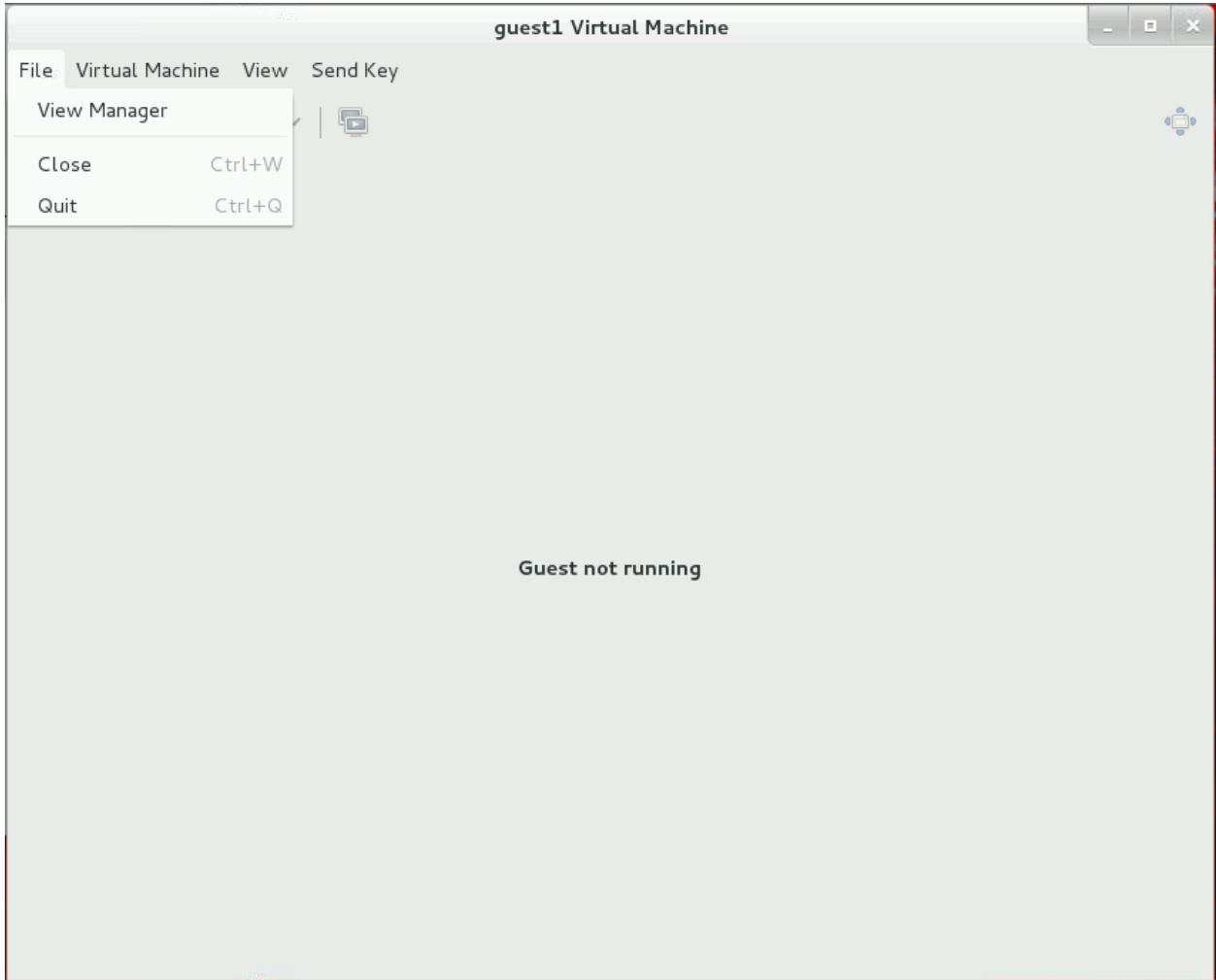
- f. Use the `grep` command to search for the `no_timer_check` string in the `/boot/grub2/grub.cfg` file.

```
# grep no_timer_check /boot/grub2/grub.cfg
...
linux16 /vmlinuz-3.8.13-55.1.6.el7uek.x86_64 ...
crashkernel=auto no_timer_check
...
```

- The `no_timer_check` kernel parameter disables the Linux kernel timer. The kernel timer does not work properly when running KVM guests with qemu emulation. If the guest knows it is running under KVM, then the `no_timer_check` is added automatically at boot time. When running the guest with qemu emulation, you might have to add the `no_timer_check` to the kernel line of your guest during the boot process. The existing OS image you imported to create this virtual machine contained the `no_timer_check` kernel parameter.
- g. Run any other commands you like to explore the configuration.
- h. Use the `systemctl poweroff` command to shut down `guest1`.

```
# systemctl poweroff
...
```

- The window appears as follows after the system powers off.



- Close the Virtual Machine window by selecting “File > Close” from the menu bar.
- Close the Virtual Machine Manager by selecting “File > Quit.”
- Run the `systemctl poweroff` command to shut down **host05**.

```
# systemctl poweroff
```

- From **dom0**, start the **host01**, **host02**, and **host03** VMs.
  - Use the `cd` command to change to the `/OVS/running_pool/host01` directory.
    - Use the `xm create vm.cfg` command to start the **host01** VM.

```
# cd /OVS/running_pool/host01
# xm create vm.cfg
Using config file "./vm.cfg".
Started domain host01 (id=...)
```

- b. Use the `cd` command to change to the `/OVS/running_pool/host02` directory.
- Use the `xm create vm.cfg` command to start the **host02** VM.

```
# cd /OVS/running_pool/host02
# xm create vm.cfg
Using config file "./vm.cfg".
Started domain host02 (id=...)
```

- c. Use the `cd` command to change to the `/OVS/running_pool/host03` directory.
- Use the `xm create vm.cfg` command to start the **host03** VM.

```
# cd /OVS/running_pool/host03
# xm create vm.cfg
Using config file "./vm.cfg".
Started domain host03 (id=...)
```

- d. Run the `xm list` command to verify that all three VMs are running.

# xm list					
Name	ID	Mem	VCPUs	State	Time (s)
Domain-0	0	2048	2	r-----	...
host01	242	1536	1	-b----	...
host02	243	1536	1	-b----	...
host03	244	1536	1	-b----	...

- In this example, **host01**, **host02**, and **host03**, in addition to **Domain-0**, are running as expected.
- The ID values and the State values might differ from your values.



## **Practices for Lesson 15: Linux Containers (LXC)**

**Chapter 15**

## Practices for Lesson 15: Overview

---

### Practices Overview

In these practices, you do the following:

- Prepare your system to create a Linux Container.
- Create a Linux Container from a template.
- Explore container configuration.
- Use `lxc` commands.

## Practice 15-1: Completing Linux Container Prerequisites

---

### Overview

In this practice, you perform the following:

- Review Yum repository configuration on **dom0**.
- Install the Linux Container packages.
- Ensure that the `libvirtd` service is running.
- View information about your kernel configuration.
- Create and mount a Btrfs subvolume.

### Assumptions

- You are the `root` user on **dom0**.

### Tasks

1. Review Yum repository configuration on **dom0**.

- In Practice 15-2, you use the `lxc-create` command to create an Oracle Linux Container from the `lxc-oracle` template.
- The `lxc-oracle` template allows you to get the Oracle Linux software packages from a local Yum repository.
- Run the following commands to review the local Yum repository configuration on **dom0**.

- a. Use the `cd` command to change to the `/var/www/html/repo/` directory.

```
# cd /var/www/html/repo
```

- b. Use the `ls -l` command to view the contents of the directory.

- Note that there is one directory and one regular file in this directory.

```
# ls -l
drwxr-xr-x ... OracleLinux
-rw-r--r-- ... public-yum-ol7.repo
```

- c. Use the `cat` command to view the contents of the `public-yum-ol7.repo` file.

- Note the “`baseurl`” setting in this file.

```
# cat public-yum-ol7.repo
[ol7_u1_base]
name=Oracle Linux 7.1 installation media copy ()
baseurl=http://192.0.2.1/repo/OracleLinux/OL7/1/x86_64/
enabled=1
```

- d. Use the `ls` command to view the contents of the `OracleLinux` directory.

- Note that the RPM packages are available in the `OracleLinux/OL7/1/x86_64/Packages/` directory.

```
# ls OracleLinux/OL7/1/x86_64/
addons    images      RELEASE-NOTES-U1-en          RPM-GPG-KEY-oracle
EFI       isolinux    RELEASE-NOTES-U1-en.html    TRANS.TBL
EULA     LiveOS      repodata
```

GPL	Packages	RPM-GPG-KEY
-----	----------	-------------

2. From **dom0**, use the `ssh` command to log in to **host02**.

- The root password is `oracle`.

```
[dom0]# ssh host02
root@host02's password: oracle
[root@host02 ~]#
```

3. Install the required packages.

a. Run the `yum install` command to install the `lxc` package.

- There are a number of dependency packages that are also installed including `libvirt` and `lxc-libs`.
- Answer "y" to "Is this ok."

```
# yum install lxc
...
Dependencies Resolved
=====
Package ... =====
=====
Installing:
lxc ...
Installing for dependencies:
libvirt ...
libvirt-daemon-config-network ...
libvirt-daemon-config-nwfilter ...
libvirt-daemon-driver-lxc ...
lxc-libs ...

Transaction Summary
=====
Install 1 Package (+5 Dependent packages)

Total download size: 1.3 M
Installed size: 2.7 M
Is this ok [y/d/N]: y
...
Complete!
```

- b. Use the `systemctl` command to check the status of the `libvирtd` service.
- In this example, the service is enabled and running.

```
# systemctl status libvirtd
libvirtd.service - Virtualization daemon
    Loaded: loaded (/usr/lib/systemd/system/libvirtd... (enabled)
              Active: active (running) since ...
...
...
```

4. Run the `lxc-checkconfig` command to display information about your kernel configuration.
- All namespaces, control groups, and miscellaneous kernel functionalities are enabled with the exception of user namespaces. Support for user namespaces is included with kernel version 3.13.

```
# lxc-checkconfig
Kernel configuration not found at /proc/config.gz; search...
Kernel configuration found at /boot/config-3.8.13-55.1.6...
--- Namespaces ---
Namespaces: enabled
Utsname namespaces: enabled
Ipc namespaces: enabled
Pid namespaces: enabled
User namespaces: missing
Network namespaces: enabled
Multiple /dev/pts instances: enabled

--- Control groups ---
Cgroup: enabled
Cgroup clone_children flag: enabled
Cgroup device: enabled
Cgroup sched: enabled
Cgroup cpu account: enabled
Cgroup memory controller: enabled
Cgroup cpuset: enabled

--- Misc ---
Veth pair device: enabled
Macvlan: enabled
Vlan: enabled
File capabilities: enabled

Note : Before booting a new kernel, you can check its
configuration
usage : CONFIG=/path/to/config /usr/bin/lxc-checkconfig
```

5. Set up the Btrfs file system for the Linux Containers.
  - a. Use the `mkfs.btrfs` command to create a Btrfs file system on `/dev/xvdd`.
    - Ensure that you create the file system on `/dev/xvdd`, not `/dev/xvdb`.

```
# mkfs.btrfs -f -L container /dev/xvdd
...
```
  - b. Use the `vi` editor to add the following entry in `/etc/fstab` to mount `/dev/xvdd` on `/container`.
    - The `/container/` directory is created when installing the `lxc` package.

```
# vi /etc/fstab
...
/dev/xvdd  /container  btrfs  defaults  0  0
```
  - c. Use the `mount -a` command to mount file systems in `/etc/fstab`.

```
# mount -a
```

- d. Use the `df` command to view the mounted file systems.
  - Note that `/dev/xvdd` is mounted on `/container`.

```
# df -h
Filesystem  Size  Used  Avail  Use%  Mounted on
...
/dev/xvdd    20G   512K   18G    1%   /container
```

## Practice 15-2: Creating an Oracle Linux Container

---

### Overview

In this practice, you perform the following:

- Create an Oracle Linux Container by using the `lxc-oracle` template.
- Explore the container configuration.

### Assumptions

- You are the `root` user on **host02**.

### Tasks

1. Create an Oracle Linux Container.
  - a. Use the `lxc-create` command to display options supported by the `lxc-oracle` template.
    - Note that you can use the `-u` option and reference a local Yum mirror, which is what you do in step 1b.

```
# lxc-create -t oracle -h

...
-a|--arch=<arch>           architecture (ie. i386, x86_64)
-R|--release=<release>       release to download for the new ...
--rootfs=<path>             rootfs path
-r|--rpms=<rpm name>        additional rpms to install into ...
-u|--url=<url>              replace yum repo url (ie Oracle ...
                            use package repository (ie file:///...
                            arch and release must also be ...
                            copy/clone rootfs at path instead ...
                            only patch the rootfs at path for ...
                            -h|--help

Release is of the format "major.minor", for example "5.8",
"6.3", or "6.latest"
This template supports Oracle Linux releases 4.6 - 7.0
```

- b. Use the `lxc-create` command to create an Oracle Linux Container named `ol-test` from the `lxc-oracle` template. Use the `-u http://192.0.2.1/repo` template option to get packages from the local Yum mirror on **dom0** (192.0.2.1).
  - This command takes a few minutes to complete.
  - Notice that the `lxc-oracle` template creates an `oracle` user with the password `oracle`.
  - The `lxc-oracle` template sets the `root` password to `root`.

- The lxc-oracle template uses the virbr0 bridge set up by libvирtd.

```
# lxc-create -n ol-test -B btrfs -t oracle -- -u
http://192.0.2.1/repo

...
Host is OracleServer 7.1
Create configuration file /container/ol-test/config
Yum installing release 7.1 for x86_64
...
Complete!
Rebuilding rpm database
Patching container rootfs /container/ol-test/rootfs for ...
Configuring container for Oracle Linux 7.1
Added container user:oracle password:oracle
Added container user:root password:root
Container : /container/ol-test/rootfs
Config     : /container/ol-test/config
Network    : eth0 () on virbr0
```

## 2. Explore the container configuration.

- Use the ls command to display the contents of /container.

```
# ls /container
ol-test
```

- Notice that the system object directory with the name of the container, ol-test, exists in /container.
- Use the cd command to change to the /container/ol-test directory, and then use the ls command to display the contents of the directory.

```
# cd /container/ol-test
# ls
config  rootfs
```

- The config file contains the configuration settings for the container, which were obtained from the template script.
- Use the cat command to view the config file.
  - The lxc.arch key specifies the architecture for the container. Without any template options, the architecture is the same as the host system, x86\_64.
  - The lxc.utsname key specifies the host name for the container.
  - Containers can have their own network interface (the lxc-oracle template defaults to eth bridge), or share with the host system (lxc.network.type is not configured; this is not recommended), or have only loopback (lxc.network.type = empty, which is the most secure).
  - The lxc.network.type key is set to veth, which is a peer network device with one side assigned to the container and the other side attached to a bridge specified by lxc.network.link.

- The `lxc.rootfs` key specifies the root file system, `/container/ol-test/rootfs`, for the container. When not specified, the container shares its root file system with the host system.
- Note that `lxc.include` is set to `/usr/share/lxc/config/oracle.common.conf`.

```
# cat config
# Template used to create this container: /usr/.../lxc-oracle
# Parameters passed to the template: -u http://192.0.2.1/repo
# For additional config options, please look at
lxc.container.conf(5)

lxc.network.type = veth
lxc.network.flags = up
lxc.network.link = virbr0
lxc.rootfs = /container/ol-test/rootfs

# Common configuration
lxc.include = /usr/share/lxc/config/oracle.common.conf

# Container configuration for Oracle Linux 7.0
lxc.arch = x86_64
lxc.utsname = ol-test
lxc.cap.drop = sys_resource
lxc.autodev = 1
lxc.kmsg = 0

# Networking
lxc.network.name = eth0
lxc.network.mtu = 1500
lxc.network.hwaddr = fe:b7:ed:07:08:a7
```

- d. Use the `less` command to view the `/usr/share/lxc/config/oracle.common.conf` file.

```
# less /usr/share/lxc/config/oracle.common.conf
# Console settings
lxc.devttydir = lxc
lxc.tty = 4
lxc.pts = 1024

# Mount entries
lxc.mount.auto - cgroup:mixed proc:mixed sys:ro

# Ensure hostname is changed on clone
```

```
lxc.hook.clone = /usr/share/lxc/hooks/clonehostname

# Capabilities
...
lxc.cap.drop = mac_admin mac_override
lxc.cap.drop = sys_module sys_nice sys_pacct
lxc.cap.drop = sys_rawio sys_time

# Control Group devices; all denied except those whitelisted
lxc.cgroup.devices.deny = a
# Allow any mknod (but not reading/writing the node)
lxc.cgroup.devices.allow = c *:* m
...
```

- e. Use the `ls` command to view the contents of the `rootfs` directory.

```
# ls rootfs
bin dev home lib64 mnt proc run selinux sys usr
boot etc lib media opt root sbin srv tmp var
```

- f. Use the `btrfs subvolume list` command to view the subvolumes in the `/container` Btrfs file system.

- Sample output is shown. The ID and gen numbers might be different.

```
# btrfs sub list /container
ID 257 gen 12 top level 5 path ol-test/rootfs
```

## Practice 15-3: Using lxc Commands

---

### Overview

In this practice, you perform the following:

- Use the `lxc-start` command to start the container.
- Log in to the container and explore the configuration.
- Use the `lxc-stop` command to stop the container.
- Use the `lxc-info` command to display the state of the container.
- Use the `lxc-start` command to start the container as a daemon.
- Use the `lxc-console` command to launch a console for the container.
- Use the `lxc-freeze` and `lxc-unfreeze` commands to freeze and unfreeze the container's processes, respectively.
- Use the `lxc-cgroup` command to view and modify container cgroup subsystem parameters.

### Assumptions

- You are the `root` user on `host02`.

### Tasks

1. Verify that a bridge is set up on the host.
  - a. Use the `ip addr` command to view the network interface configuration.
    - Sample output is shown. Output on your system might be different.

```
# ip addr
...
6: virbr0 <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc ...
    link/ether 52:54:00:97:92:a5 brd ff:ff:ff:ff:ff:ff
    inet 192.168.122.1/24 brd 192.168.122.255 scope global ...
        7. virbr0-nic <BROADCAST,MULTICAST> mtu 1500 qdisc pfifo...
            link/ether 52:54:00:97:92:a5 brd ff:ff:ff:ff:ff:ff
```
  - b. If you do not see `virbr0` configuration information, use the `systemctl` command to restart the `libvirtd` service.
 

```
# systemctl restart libvirtd
```

    - Repeat step 1a to ensure that `virbr0` configuration information exists.
2. Start a Linux Container.
  - a. Use the `lxc-start` command to start the `ol-test` container.
    - Log in as the `root` user. Password is `root`.
    - When you start an Oracle Linux container without the daemon option, `-d`, you get a login prompt in your current terminal.

- This login prompt is the “console.” The `lxc-oracle` template, by default, also sets up four additional terminals that you can log in on with `lxc-console`.

```
# lxc-start -n ol-test
systemd 208 running in system mode. (+PAM +LIBWRAP +AUDIT ...)
Detected virtualization 'lxc'.

Welcome to Oracle Linux Server 7.1!

Set hostname to <ol-test>.
[ OK ] Reached target Remote File Systems.
[ OK ] Created slice Root Slice.
...
Oracle Linux Server release 7.1
Kernel 3.8.13-55.1.6.el7uek.x86_64 on an x86_64

ol-test login: root
Password: root
[root@ol-test ~] #
```

### 3. Explore the Linux Container.

- Use the `ps` command to view the processes within the container.
  - Sample output is displayed.
  - When you use the `lxc-start` command to start a container, by default, the copy of `/sbin/init` in the container is started to spawn other processes in the container’s process space.
  - Any device access or system calls are handled by the kernel running on the host.
  - Notice that the `/sbin/init` process within the container has a PID of 1.
  - Notice that four terminals are available.

```
# ps -ef
UID      PID      PPID    ...      CMD
root      1        0    ...      /sbin/init
root     11        1    ...      /usr/lib/systemd/systemd-journald
root     25        1    ...      /usr/sbin/rsyslogd -n
root     27        1    ...      /usr/lib/systemd/systemd-logind
dbus     29        1    ...      /bin/dbus-daemon --system --address...
root     30        1    ...      /sbin/agetty --noclear tty1
root     33        1    ...      /sbin/agetty --noclear tty2
root     37        1    ...      /sbin/agetty --noclear tty3
root     38        1    ...      /sbin/agetty --noclear tty4
root     40        1    ...      login -- root
root    209        1    ...      /sbin/dhclient -H ol-test -1 -q ...
root    209        1    ...      /usr/sbin/sshd -D
root    257       40    ...      -bash
```

```
root      274      257 ... ps -ef
```

- b. Use the `mount` command to display the mounted file systems within the container.

```
# mount
/dev/xvdd on / type btrfs (rw,relatime,seclabel,ssd,space_cache)
devtmpfs on /dev type devtmpfs (rw,nosuid,seclabel,size=...
proc on /proc type proc (rw,nosuid,nodev,noexec,relatime)
proc on /proc/sys type proc (ro,nosuid,nodev,noexec,relatime)
proc on /proc/sysrq-trigger type proc (ro,nosuid,nodev,noexec...
sysfs on /sys type sysfs (ro,relatime,seclabel)
cgroup root on /sys/fs/cgroup type tmpfs (rw,nosuid,nodev,...
...
devpts on /dev/lxc/console type devpts (rw,nosuid,noexec,...
devpts on /dev/lxc/tty1 type devpts (rw,nosuid,noexec,...
...
```

- c. Use the `ip addr` command to display the network interfaces for the container.

- Sample output is shown; yours might be different.

```
# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state ...
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
...
8: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc ...
    link/ether fe:32:8b:96:b6:ad brd ff:ff:ff:ff:ff:ff
    inet 192.168.122.210/24 brd 192.168.122.255 scope global...
...
```

- d. Use the `cat` command to view the `eth0` configuration file, `/etc/sysconfig/network-scripts/ifcfg-eth0`.

- Notice that the default configuration uses DHCP.

```
# cat /etc/sysconfig/network-scripts/ifcfg-eth0
DEVICE=eth0
BOOTPROTO=dhcp
ONBOOT=yes
HOSTNAME=ol-test
DHCP_HOSTNAME=ol-test
NM_CONTROLLED=no
TYPE=Ethernet
```

- e. Use the `cat` command to view the SELinux configuration file, `/etc/selinux/config`.
- Notice that SELinux is disabled because some things in containers do not yet work with SELinux enabled.

```
# cat /etc/selinux/config
SELINUX=disabled
```

4. Log out of the Linux Container.

- a. Use the `exit` command to log off the container.
- Notice that you are still presented with a login prompt.
  - The only way to get back this terminal is by stopping the container, either by issuing a `shutdown` command while logged in, or by running `lxc-stop` from another terminal.

```
# exit
logout

Oracle Linux Server release 7.1
Kernel 3.8.13-55.1.6.el7uek.x86_64 on an x86_64

ol-test login:
```

- b. Open a new terminal window from **dom0**. Become the `root` user by running the `su -` command. The `root` password on **dom0** is `oracle`.

```
[dom0] $ su -
Password: oracle
```

- c. From **dom0**, use the `ssh` command to connect to **host02**. The `root` password on **host02** is `oracle`.

```
[dom0] # ssh host02
root@host02's password: oracle
```

- d. Use the `lxc-info` command to display information for the `ol-test` container.

- Sample output is displayed.
- Notice that the container is **RUNNING**.

```
# lxc-info -n ol-test
Name:          ol-test
State:         RUNNING
PID:           3379
IP:            192.168.122.210
CPU use:       1.39 seconds
BlkIO use:     1.69 MiB
Memory use:    26.87 MiB
KMem use:      0 bytes
Link:          vethESKLMV
TX bytes:     1.75 KiB
```

```
RX bytes:      38.53 KiB
Total bytes:  40.28 KiB
```

- e. Run the `lxc-stop` command.

- Notice that the other terminal window now has a **host02** prompt.

```
# lxc-stop -n ol-test
```

- f. Use the `lxc-info` command to display information for the `ol-test` container.

- Notice that the container is stopped.

```
# lxc-info -n ol-test
Name:          ol-test
State:         STOPPED
```

5. Start a Linux Container as a daemon.

- a. Use the `lxc-start` command with the `-d` option to start the `ol-test` container as a daemon.

- Notice that you are not presented with a login prompt.

```
# lxc-start -n ol-test -d
```

- b. Use the `lxc-info` command to display the state of the `ol-test` container.

- Sample output is displayed.
- In this example, the PID is 4354. Yours might be different.

```
# lxc-info -n ol-test
Name:          ol-test
State:         RUNNING
PID:           4354
IP:            192.168.122.210
CPU use:       0.99 seconds
BlkIO use:     1.06 MiB
Memory use:   25.51 MiB
KMem use:      0 bytes
Link:          vethWIXE06
TX bytes:     776 bytes
RX bytes:     991 bytes
Total bytes:  1.73 KiB
```

- c. Use the `ps` command to view the processes pertaining to the running container. Pipe the output to `grep` and search for the PID from the previous step.

- Notice that a number of `/sbin/agetty` processes are running.
- In this example, the parent (PPID) of the `/sbin/init` process (PID 4354) is 4350. Yours might be different.

```
# ps -ef | grep 4354
root  4354  4350    ... /sbin/init
root  4372  4354    ... /usr/lib/systemd/systemd-journald
root  4392  4354    ... /usr/sbin/rsyslogd -n
root  4393  4354    ... /usr/lib/systemd/systemd-logind
```

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

```

dbus    4395  4354  ... /bin/dbus-daemon --system --address...
root    4398  4354  ... /sbin/agetty --noclear tty1
root    4399  4354  ... /sbin/agetty --noclear tty2
root    4401  4354  ... /sbin/agetty --noclear tty3
root    4402  4354  ... /sbin/agetty --noclear tty4
root    4403  4354  ... /sbin/agetty --noclear --keep-baud ...
root    4598  4354  ... /sbin/dhclient -H ol-test -1 -q ...
root    4652  4354  ... /usr/sbin/sshd -D
...

```

- d. Use the `ps` command to view the command associated with the PPID.

- Notice that the `lxc-start` command is associated with this PID.

```

# ps -ef | grep 4350
root    4350      1  ... lxc-start -n ol-test -d
root    4354    4350  ... /sbin/init
...

```

6. Log in to and out of the Linux Container.

- a. Use the `lxc-console` command to launch a console for the container.

- Log in as the `root` user. Password is `root`.
- Use of `lxc-console` requires that the container be running a `/sbin/agetty` process.
- You can also use `ssh` to log in to the container.
- To log in using `ssh`, SELinux must be disabled on the host system and the container must be running the SSH daemon.

```

# lxc-console -n ol-test

Connected to tty 1
Type <Ctrl+a q> to exit the console, ...

Oracle Linux Server release 7.1
Kernel 3.8.13-55.1.6.el7uek.x86_64 on an x86_64

ol-test login: root
Password: root
[root@ol-test ~]#

```

- b. Press `Ctrl + a` followed by `q` to exit the `lxc-console` session.
- Run the `hostname` command before and after pressing the control sequence. Notice the host name before and after exiting the `lxc-console` session.
  - Do not hold all three keys down at once.
  - Press `Ctrl + a`, then release these keys, and then press `q` to exit.

```
[root@ol-test ~]# hostname
ol-test
# CTRL+a q
[root@host02 ~]# hostname
host02.example.com
```

7. Freeze and unfreeze a Linux Container.

- a. Use the `lxc-info` command to display the state of the `ol-test` container.
- Notice that the state of the container is `RUNNING`.

```
# lxc-info -n ol-test
Name:          ol-test
State:         RUNNING
PID:           4354
IP:            192.168.122.210
CPU use:       1.08 seconds
BlkIO use:     1.06 MiB
Memory use:   26.53 MiB
KMem use:      0 bytes
Link:          vethWIXE06
TX bytes:     936 bytes
RX bytes:     33.99 KiB
Total bytes:  34.91 KiB
```

- b. Use the `lxc-freeze` command to freeze all of the container's processes.

```
# lxc-freeze -n ol-test
```

- c. Use the `lxc-info` command to display the state of the `ol-test` container.
- Notice that the state of the container is `FROZEN`.

```
# lxc-info -n ol-test
Name:          ol-test
State:         FROZEN
PID:           4354
...
```

- d. Use the `lxc-console` command to launch a console for the container.
- You cannot log in while a container is in a FROZEN state.
  - Press `Ctrl + a`, then release these keys, and then press `q` to exit.

```
# lxc-console -n ol-test
```

Type `<Ctrl+a q>` to exit the console, ...

- e. Use the `lxc-unfreeze` command to unfreeze all of the container's processes.

```
# lxc-unfreeze -n ol-test
```

- f. Use the `lxc-info` command to display the state of the `ol-test` container.

- Notice that the state of the container is RUNNING.
- Now you can successfully log in by using the `lxc-console` command.

```
# lxc-info -n ol-test
```

Name:	ol-test
State:	RUNNING
PID:	4354
...	

## 8. View and modify the container resources.

- a. Use the `lxc-cgroup` command to view the devices (`devices.list`) that the container can use.

```
# lxc-cgroup -n ol-test devices.list
c *:* m
b *:* m
c 1:3 rwm
c 1:5 rwm
c 1:7 rwm
c 5:0 rwm
c 1:8 rwm
c 1:9 rwm
c 136:* rwm
c 5:2 rwm
```

- b. Use the `grep` command to display the lines in the `/container/ol-test/config` file that contain the string "devices."

- In this example, no output is displayed.

```
# grep devices /container/ol-test/config
```

- c. Use the `grep` command to display the lines in the `/container/ol-test/config` file that contain the string "include."

```
# grep include /container/ol-test/config
```

```
lxc.include = /usr/share/lxc/config/oracle.common.conf
```

- d. Use the `grep` command to display the lines in the `/usr/share/lxc/config/oracle.common.conf` file that contain the string "devices."
- Notice that these entries correspond with the output of the `lxc-cgroup` command.

```
# grep devices /usr/share/lxc/config/oracle.common.conf
lxc.cgroup.devices.deny = a
lxc.cgroup.devices.allow = c *:* m
lxc.cgroup.devices.allow = b *:* m
lxc.cgroup.devices.allow = c 1:3 rwm
lxc.cgroup.devices.allow = c 1:5 rwm
lxc.cgroup.devices.allow = c 1:7 rwm
lxc.cgroup.devices.allow = c 5:0 rwm
lxc.cgroup.devices.allow = c 1:8 rwm
lxc.cgroup.devices.allow = c 1:9 rwm
lxc.cgroup.devices.allow = c 136:* rwm
lxc.cgroup.devices.allow = c 5:2 rwm
```

- e. Use the `lxc-cgroup` command to view the relative share of CPU time available (`cpu.shares`) to the tasks in the container.

```
# lxc-cgroup -n ol-test cpu.shares
1024
```

- f. Use the `cat` command to view the `cpu.shares` subsystem parameter for the host.

- Note that `cpu.shares` is set to 1024 for the host.

```
# cat /sys/fs/cgroup/cpu,cpuacct/cpu.shares
1024
```

- g. Use the `cat` command to view the `cpu.shares` subsystem parameter for the container.

- Note that `cpu.shares` is also set to 1024 for the container.

```
# cat /sys/fs/cgroup/cpu,cpuacct/lxc/ol-test/cpu.shares
1024
```

- h. Use the `lxc-cgroup` command to change the `cpu.shares` to 750 for the `ol-test` container.

```
# lxc-cgroup -n ol-test cpu.shares 750
```

- i. Use the `cat` command to view the `cpu.shares` subsystem parameter for both the host and the container.

- Note that the value is still 1024 for the host but is now 750 for the container.

```
# cat /sys/fs/cgroup/cpu,cpuacct/cpu.shares
1024
# cat /sys/fs/cgroup/cpu,cpuacct/lxc/ol-test/cpu.shares
750
```

- j. Use the `vi` editor to add an entry to the `/container/ol-test/config` file, which permanently sets `cpu.shares` to 500 for the `ol-test` container.

```
# vi /container/ol-test/config  
...  
lxc.cgroup.cpu.shares = 500
```

- k. Use the `lxc-stop` command to stop the `ol-test` container.

- You need to use `lxc-stop` and `lxc-start` on the container to re-read the container's configuration file.

```
# lxc-stop -n ol-test
```

- l. Use the `lxc-start` command with the `-d` option to start the `ol-test` container as a daemon.

```
# lxc-start -n ol-test -d
```

- m. Use the `cat` command to view the `cpu.shares` subsystem parameter for the container.

- Notice that the `cpu.shares` setting is now persistent.

```
# cat /sys/fs/cgroup/cpu,cpuacct/lxc/ol-test/cpu.shares  
500
```

9. Log off the **host02** VM.

```
# exit  
logout  
Connection to host02 closed.
```

## **Practices for Lesson 16: Docker**

**Chapter 16**

## Practices for Lesson 16: Overview

---

### Practices Overview

In these practices, you do the following:

- Upload the Docker package and Docker images from **dom0** to **host02**.
- Install the Docker package on **host02**.
- Enable and start the `docker` service.
- Display information about Docker configuration and version.
- Load Docker images into the local Docker repository on **host02**.
- Use the `docker` command-line interface.

## Practice 16-1: Using sftp to Upload Docker Package and Images

---

### Overview

In this practice, you:

- Use sftp to upload the docker package from the **dom0** to the **host02** VM. Normally, you obtain this package from the Oracle Linux 7 addons channel on ULN or Public Yum.
- Use sftp to upload the \*.tar files from the **dom0** to the **host02** VM. Normally, you obtain Docker images from Docker Hub.

### Assumptions

- You are the root user on the **host02** VM.
- You are the root user on **dom0**.

### Tasks

1. Use the sftp to transfer the docker package from **dom0** to **host03**.
  - a. From **dom0**, use the cd command to change to the /OVS/seed\_pool/sfws directory.

```
[dom0]# cd /OVS/seed_pool/sfws
```

  - b. Use the ls command to view the directory for the docker\* package.

```
[dom0]# ls docker*
docker-1.6.1-1.0.1.el7.x86_64.rpm
```

  - c. Use the ls command to view the directory for the \*.tar files.
    - The three TAR files are Docker images that have been downloaded from Docker Hub for use in this practice.

```
[dom0]# ls *.tar
fedora.tar    oraclelinux.tar    ubuntu.tar
```

- d. Use the sftp command to connect to **host02** as root. Password is oracle.

```
[dom0]# sftp host02
root@host02's password: oracle
sftp>
```

- e. Use the put command to copy the docker\* packages to **host02**.

```
sftp> put docker*
Uploading docker-1.6.1-1.0.1.el7.x86_64.rpm to /root/...
...
```

- f. Use the `put` command to copy the `*.tar` files to **host02**.

```
sftp> put *.tar  
Uploading fedora.tar to /root/fedora.tar  
...  
Uploading oraclelinux.tar to /root/oraclelinux.tar  
...  
Uploading ubuntu.tar to /root/ubuntu.tar  
...
```

- g. Use the `quit` command to exit sftp.

```
sftp> quit
```

## Practice 16-2: Installing and Configuring Docker

---

### Overview

In this practice, you perform the following:

- Install the Docker package on **host02**.
- Enable and start the `docker` service.
- Display information about Docker configuration and version.
- Load Docker images into the local Docker repository on **host02**.

### Assumptions

- You are the `root` user on **dom0**.

### Tasks

1. Install the `docker` package on **host02**.

- a. From **dom0**, use the `ssh` command to log on to **host02** as the `root` user.
  - Password is `oracle`.

```
# ssh host02
root@host02's password: oracle
```

- b. Use the `rpm` command to install the `docker` package.

```
# rpm -Uvh docker-1.6.1-1.0.1.el7.x86_64.rpm
...
```

2. Use the `systemctl` command to enable and start the `docker` server and display status information about the `docker` service.

- a. Enable the `docker` server.

```
# systemctl enable docker
ln -s '/usr/lib/systemd/system/docker.service'
'/etc/systemd/system/multi-user.target.wants/docker.service'
```

- b. Start the `docker` server.

- It takes a few seconds for the `docker` service to start.

```
# systemctl start docker
```

- c. View status information about the `docker` service.

```
# systemctl status docker
docker.service - Docker Application Container Engine
   Loaded: loaded (/usr/lib/systemd/system/docker.service; ...)
   Active: active (running) since ...
     Docs: http://docs.docker.com
   Main PID: ... (docker)
      CGroup: /system.slice/docker.service
              └─... /usr/bin/docker -d --selinux-enabled
...
...
```

3. Use docker commands to display information about the configuration and version.
- a. Use the docker version command to show the version information.

```
# docker version
Client version: 1.6.1
Client API version: 1.18
Go version (client): go1.3.3
Git commit (client): a8a31ef/1.6.1
OS/Arch (client): linux/amd64
Server version: 1.6.1
Server API version: 1.18
Go version (server): go1.3.3
Git commit (server): a8a31ef/1.6.1
OS/Arch (server): linux/amd64
```

- b. Use the docker info command to display systemwide information about the configuration.
- Note that there are currently 0 containers and 0 images.
  - Note the default Storage Driver is “devicemapper”.

```
# docker info
Containers: 0
Images: 0
Storage Driver: devicemapper
  Pool Name: docker-252:0-37294159-pool
  Pool Blocksize: 65.54 kB
  Backing Filesystem: xfs
  Data file: /dev/loop0
  Metadata file: /dev/loop1
  Data Space Used: 307.2 MB
  Data Space Total: 107.4 GB
  Data Space Available: 15.04 GB
  Metadata Space Used: 729.1 kB
  Metadata Space Total: 2.147 GB
  Metadata Space Available: 2.147 GB
  Udev Sync Supported: true
  Data loop file: /var/lib/docker/devicemapper/.../data
  Metadata loop file: /var/lib/docker/devicemapper/.../metadata
  Library Version: 1.02.93-RHEL7 (2015-01-28)
Execution Driver: native-0.2
Kernel Version: 3.8.13-55.1.6.el7uek.x86_64
Operating System: Oracle Linux Server 7.1
CPUs: 1
Total Memory: 1.462 GiB
Name: host02.example.com
```

ID: FT47:K7G4:KFEE:3MIA:DRAK:2ETS:K7SG:XZTC:YMRY:ZCWO:U53Q:2KXU

4. Load Docker images.

- If you have access to Docker Hub, you can use the `docker pull` command to download an image or a repository from the Docker Hub Registry to your local system.
- The classroom systems do not have access to Docker Hub so you load images from the TAR files you copied from **dom0** in Practice 16-1.
  - a. Use the `docker images` command to list images stored in the local Docker repository.
    - Note that there are no images.

```
# docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	VIRTUAL SIZE
------------	-----	----------	---------	--------------

- b. Use the `docker load` command to load the images from the `oraclelinux.tar` file you copied from **dom0** in Practice 16-1.
  - This command takes a few seconds to complete.
- c. Use the `docker images` command to list images stored in the local Docker repository.
  - Note that there are now two images:
    - `oraclelinux:6.6`
    - `oraclelinux:latest`
  - These images were downloaded from Docker Hub from a system with access to Docker Hub. The TAR file was created by using the following command:
    - `docker save oraclelinux > oraclelinux.tar`
  - The TAR file was then staged on **dom0** for use in this practice.

```
# docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	VIRTUAL SIZE
oraclelinux	6.6	88c24509b051	...	157.7 MB
oraclelinux	latest	8a2b759d9dd8	...	189.5 MB

- d. Use the `docker load` command to load the images from the following TAR files you copied from **dom0** in Practice 16-1.
  - `fedora.tar`
  - `ubuntu.tar`
  - These commands take a few seconds to complete.

```
# docker load --input fedora.tar
# docker load --input ubuntu.tar
```

- e. Use the docker images command to list images stored in the local Docker repository.
- Note that there are now two more images: fedora and ubuntu.

# docker images					
REPOSITORY	TAG	IMAGE ID	CREATED	VIRTUAL SIZE	
oraclelinux	6.6	88c24509b051	...	157.7 MB	
ubuntu	latest	d2a0ecffe6fa	...	188.3 MB	
oraclelinux	latest	8a2b759d9dd8	...	189.5 MB	
fedora	latest	ded7cd95e059	...	186.5 MB	

## Practice 16-3: Using Docker Commands

---

### Overview

In this practice, you perform the following:

- Create a Docker container.
- Start and stop a Docker container.
- Display information for a Docker container.
- Create a Docker container for different Linux distributions.
- Execute a command in a running Docker container.
- Create a Docker image from a Docker container.

### Assumptions

- You are the `root` user on **host02**.

### Tasks

1. Run the `docker` command with no arguments to view the usage.

- Note that there are options and commands.

```
# docker
Usage: docker [OPTIONS] COMMANDS [arg...]

A self-sufficient runtime for linux containers.

Options:
  --api-enable-cors=          Set CORS headers in the ...
  -bm, --bridge=""            Attach containers to a ...
  --bip=""                     Specify network bridge IP
  -D, --debug=false           Enable debug mode
  -d, --daemon=false          Enable daemon mode
  --default-ulimit=[]         Set default ulimits for ...
  --dns=[]                     DNS server to use
  --dns-search=[]              DNS search domains to use
  -e, --exec-driver=native    Exec driver to use
  ...
  -v, --version=false          Print version information ...

Commands:
  attach      Attach to a running container
  build       Build an image from a Dockerfile
  commit      Create a new image from a container's changes
  cp          Copy files/folders from a container's file...
  create      Create a new container
  diff        Inspect changes on a container's filesystem
```

```

events      Get real time events from the server
...
wait        Block until a container stops, then print its ...
Run 'docker COMMAND --help' for more information on a command.

```

2. Create a Docker container.

- Use the `docker ps` command to list running containers.
  - Include the `-a` option to list all containers.
  - Note that there are no containers.

```

# docker ps
CONTAINER ID    IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
# docker ps -a
CONTAINER ID    IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES

```

- Use the `docker images` command to list images stored in the local Docker repository.

```

# docker images
REPOSITORY      TAG          IMAGE ID      CREATED       VIRTUAL SIZE
oraclelinux     6.6          88c24509b051   ...          157.7 MB
ubuntu          latest        d2a0ecffe6fa   ...          188.3 MB
oraclelinux     latest        8a2b759d9dd8   ...          189.5 MB
fedora          latest        ded7cd95e059   ...          186.5 MB

```

- Use the `docker run` command to run a command inside a container.
  - This command creates a container.
  - Create the container from the `oraclelinux` image.
  - Run the `/bin/echo "Hello"` command inside the container.

```

# docker run oraclelinux /bin/echo "Hello"
Hello

```

- Use the `docker ps` command to list running containers.
  - Include the `-a` option to list all containers.
  - Note that there is now one container, which was created from the `docker run` command in step 2c.
  - Note that the CONTAINER ID is automatically created if you do not specify. The CONTAINER ID in the sample output is different on your system.
  - Note that the IMAGE used is `oraclelinux:latest`.

- Note that the container NAME is automatically created. The NAME, `sad_past` in the sample output, is different on your system.

```
# docker ps
CONTAINER ID   IMAGE      COMMAND    CREATED     STATUS      PORTS     NAMES
# docker ps -a
CONTAINER ID   IMAGE          COMMAND           ...      NAMES
ada63a78ed68   oraclelinux:latest "bin/echo Hello" ...   sad_past
```

e. Use the `docker create` command to create a new container.

- This command creates a container similar to the `docker run` command.
  - The `docker run` command creates a container and runs a command immediately inside the container.
  - The `docker create` command creates a container that you can start at a later date.
- Include the following options to the `docker create` command:
  - `--name guest`: Assigns a specific name, `guest`, to the container.
  - `-i`: Specifies to keep STDIN open even if not attached. The default is false.
  - `-t`: Allocates a pseudo-TTY. The default is false.
- Create the container from the `oraclelinux` image.
- In the sample output, the CONTAINER ID of `d5675e0d2b56` displays. Your CONTAINER ID is different.

```
# docker create -i -t --name guest oraclelinux
d5675e0d2b56...
```

f. Use the `docker ps` command to list running containers.

- Include the `-a` option to list all containers.
- Note that there are now two containers.
- Note that the COMMAND for the `guest` container is `/bin/bash`.

```
# docker ps
CONTAINER ID   IMAGE      COMMAND    CREATED     STATUS      PORTS     NAMES
# docker ps -a
CONTAINER ID   IMAGE          COMMAND           ...      NAMES
d5675e0d2b56   oraclelinux:latest "bin/bash"       ...   guest
ada63a78ed68   oraclelinux:latest "bin/echo Hello" ...   sad_past
```

3. Start and stop a Docker container.

- a. Use the `docker start` command to start the `guest` container.
- Note that the container starts but you do not get a shell prompt for the container.

```
# docker start guest
guest
```

- b. Use the `docker ps` command to list running containers.

- Note that the guest container is now running.

```
# docker ps
CONTAINER ID   IMAGE          COMMAND                  ...   NAMES
d5675e0d2b56   oraclelinux:latest "/bin/bash"        ...   guest
```

- c. Use the `docker stop` command to stop the guest container. Use the `docker ps` command to list running containers.

- Note that the guest container is not running.

```
# docker stop guest
guest
# docker ps
CONTAINER ID   IMAGE      COMMAND   CREATED   STATUS    PORTS   NAMES
```

- d. Use the `docker start` command to start the guest container.

- Include the following options to the `docker start` command:
  - `-a`: Attaches the container's STDOUT and STDERR and forward all signals to the process. The default is false.
  - `-i`: Attaches the container's STDIN. The default is false.
- Note that you now have a shell prompt for the guest container.

```
# docker start -a -i guest
[root@d5675e0d2b56 /]#
```

- e. Run the following Linux commands from within the container.

- `hostname`
- `ls`
- `ps`
- `df -h`
- `cat /etc/oracle-release`
- `ip addr`

```
# hostname
d5675e0d2b56
# ls
bin  boot  dev  etc  home  lib  lib64  media  mnt  opt  proc ...
# ps
  PID  TTY      TIME CMD
    1  ?  00:00:00 bash
   20  ?  00:00:00 ps
# df -h
Filesystem           Size  Used  Avail  Use%  Mounted on
/dev/mapper/docker-252:0-...
...
# cat /etc/oracle-release
```

```
Oracle Linux Server release 7.1
# ip addr
1: lo: ...
2: eth0: ...
...
```

- f. Use the `exit` command to log off the container.

```
# exit
[root@host02 ~]# hostname
host02.example.com
```

- g. Use the `docker ps` command to list running containers.

- Note that, after using the `exit` command, the guest container is not running.

# docker ps	CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
-------------	--------------	-------	---------	---------	--------	-------	-------

- h. Use the `docker start` command to start the guest container.

```
# docker start guest
guest
```

4. Display information for a Docker container.

- a. Use the `docker logs` command to look inside the guest container and view the logs.

- Note that the logs contain the commands issued from within the container.

```
# docker logs guest
...
```

- b. Use the `docker inspect` command to view the configuration information for the guest container.

```
# docker inspect guest
[{
    "AppArmorProfile": "",
    "Args": [],
    "Config": {
        "AttachStderr": true,
        "AttachStdin": true,
        "AttachStdout": true,
        "Cmd": [
            "/bin/bash"
        ],
        "CpuShares": 0,
        "Cpuset": "",
        "Domainname": "",
        "Entrypoint": null,
        "Env": null,
        "ExposedPorts": null,
```

```

    "Hostname": "d5675e0d2b56",
    "Image": "oraclelinux",
    "Labels": {},
    "MacAddress": "",
    "Memory": 0,
    "MemorySwap": 0,
    "NetworkDisabled": false,
    ...
},
"Created": "date...",
"Driver": "devicemapper",
"ExecDriver": "native-0.2",
"HostConfig": {
    "Binds": null,
    "CapAdd": null,
    "CapDrop": null,
    "CgroupParent": "",
    ...
},
"HostnamePath": "/var/lib/docker/containers/d5675e0d2b56...
...

```

- c. Use the `docker inspect` command to display the IP address of the guest container.

- Include the `-f {{ .NetworkSettings.IPAddress }}` option.
- Sample output is displayed. Your IP address might be different.
- Note that the IP address is different from the **host02** IP address.

```
# docker inspect -f {{.NetworkSettings.IPAddress}} guest
172.17.0.11
```

- d. View the man page for the `docker-create` command.

- Use the `/--net` command to search the man page for `--net`.
- Use the `n` key to search for the second occurrence of `--net`.
- Note that the `--net="host"` option to the `docker create` command specifies to use the same network stack as **host02**.
- After viewing the man page, use the `q` key to exit.

```
# man docker-create
...
--net
n
--net="bridge"
Set the Network mode for the container
'bridge': creates a new network stack for the container ...
```

```

'none': no networking for this container
'container': reuses another container network stack
'host': use the host network stack inside the container...
...
q

```

- e. Use the `docker inspect` command to display the network mode of the guest container.
- Include the `-f {{.HostConfig.NetworkMode}}` option.
  - Note that the network mode is set to “bridge”. This creates a new network stack, and obtains an IP address for the container that is different from the **host02** IP address.

```
# docker inspect -f {{.HostConfig.NetworkMode}} guest
bridge
```

5. Create a Docker container for different Linux distributions.
- Use the `cat` command to view the operating system version running on **host02**.

- The **host02** VM is running Oracle Linux 7.1.

```
# cat /etc/os-release
NAME="Oracle Linux Server"
VERSION="7.1"
...
```

- Use the `docker run` command with the `-t` and `-i` options to run an interactive container.
- Create the container from the `fedora` image.
  - Run the `/bin/bash` command inside the container.

```
# docker run -t -i fedora /bin/bash
```

- Use the `cat` command to view the operating system version running on the container.
- The container is running Fedora 22.

```
# cat /etc/os-release
NAME="Fedora"
VERSION="22 (Twenty Two)"
...
```

- Use the `exit` command to log off the container.
- Run the `hostname` command to confirm you have logged off the container.

```
# exit
[root@host02 ~]# hostname
host02.example.com
```

- Use the `docker run` command with the `-t` and `-i` options to run an interactive container.
- Create the container from the `ubuntu` image.
  - Run the `/bin/bash` command inside the container.

```
# docker run -t -i ubuntu /bin/bash
```

- f. Use the `cat` command to view the operating system version running on the container.
- The container is running Ubuntu 14.04.2.

```
# cat /etc/os-release
NAME="Ubuntu"
VERSION="14.04.2 LTS, Trusty Tahr"
...
...
```

- g. Use the `exit` command to log off the container.

- Run the `hostname` command to confirm you have logged off the container.

```
# exit
[root@host02 ~]# hostname
host02.example.com
```

6. Execute a command in a running Docker container.

- a. Use the `docker create` command to create a new container.

- Create the container from the `oraclelinux:6.6` image.
- Name the container `ol6.6`.
- Include the `-i` and `-t` options.

```
# docker create -i -t --name ol6.6 oraclelinux:6.6
...
```

- b. Use the `docker start` command to start the `ol6.6` container.

```
# docker start ol6.6
ol6.6
```

- c. Use the `docker exec` command to start the `sshd` service on the `ol6.6` container.

- Within the container, run the following commands:
  - `service sshd status`
  - `service sshd start`
  - `service sshd status`
  - `chkconfig sshd on`
  - `chkconfig sshd --list`

```
# docker exec ol6.6 service sshd status
openssh-daemon is stopped
# docker exec ol6.6 service sshd start
Generating SSH2 RSA host key: [ OK ]
Generating SSH1 RSA host key: [ OK ]
Generating SSH2 DSA host key: [ OK ]
Starting sshd: [ OK ]
# docker exec ol6.6 service sshd status
openssh-daemon (pid ...) is running...
# docker exec ol6.6 chkconfig sshd on
# docker exec ol6.6 chkconfig sshd --list
```

sshd	0 : off	1 : off	2 : on	3 : on	4 : on	5 : on	6 : off
------	---------	---------	--------	--------	--------	--------	---------

7. Create a Docker image from a Docker container.

- Use the `docker commit` command to save the changes to the ol6.6 container to a new image.
  - Use the `-m` option to provide a message describing the changes.
  - Use the `-a` option to provide author information.
  - Provide an image name and a tag of `mymod/sshd:v1`.

```
# docker commit -m="started sshd on OL6.6" -a="My Name" ol6.6  
mymod/sshd:v1  
...
```

- Use the `docker images` command to list images stored in the local Docker repository.
  - Note that the `mymod/sshd:v1` image now exists.

```
# docker images  
REPOSITORY      TAG      IMAGE ID      CREATED          VIRTU...  
mymod/sshd      v1       c71373b840d5    ...      157.7 MB  
oraclelinux     6.6      88c24509b051    ...      157.7 MB  
ubuntu          latest    d2a0ecffe6fa    ...      188.3 MB  
oraclelinux     latest    8a2b759d9dd8    ...      189.5 MB  
fedora          latest    ded7cd95e059    ...      186.5 MB
```

8. Log off the **host02** VM.

```
# exit  
logout  
Connection to host02 closed.
```



## **Practice for Lesson 17: Security Enhanced Linux (SELinux)**

**Chapter 17**

## Practice for Lesson 17: Overview

---

### Practices Overview

In these practices, you:

- Perform SELinux configuration from the GUI and the command line
- Change the value of an SELinux Boolean to allow `samba` to share user home directories
- Change the SELinux context file type to enable a file system to be used for storing KVM virtual machine disk images when SELinux is in `Enforcing` mode

## Practice 17-1: Exploring SELinux

---

### Overview

In this practice, you explore SELinux files and directories, execute SELinux commands, install the SELinux Administration GUI package, and explore the GUI.

### Assumptions

- You are the `root` user on `dom0`.

### Tasks

1. From `dom0`, connect to the `host03` guest by using `vncviewer`.
  - Refer to Practice 3-1: Configuring an OpenLDAP Server, if necessary, for the steps to connect using `vncviewer`.
  - From the GNOME desktop, open a terminal window.
  - Use the `su -` command to become the `root` user on `host03`.
2. As the `root` user on `host03`, explore the SELinux configuration.
  - a. Use the `cat` command to view the `/etc/selinux/config` file.
    - Note that SELinux security policy is set to `enforcing` mode.
    - Note that the `targeted` policy is in use, meaning that only targeted network daemons are protected.

```
# cat /etc/selinux/config
...
SELINUX=enforcing
...
SELINUXTYPE=targeted
```

- b. Run the `sestatus` command.
  - Note that this output confirms the settings in the `/etc/selinux/config` file.

```
# sestatus
SELINUX status:           enabled
...
Loaded policy name:       targeted
Current mode:             enforcing
Mode from config file:   enforcing
...
```

- c. Run the `getenforce` command to display the current mode.
  - Note that this output also confirms the output from the `sestatus` command.
  - If the current mode is `Permissive`, use the `setenforce 1` command to change the mode to `Enforcing`.
  - Note that running `setenforce 0` would set the mode to `Permissive`.

- Using the `setenforce` command to change the mode is not persistent.

```
# getenforce
Enforcing
```

3. Perform SELinux package management.

- a. Use the `rpm` command to list the installed `selinux` packages.

```
# rpm -qa |grep selinux
selinux-policy-targeted...
libselinux-python...
libselinux-...
libselinux-utils...
selinux-policy-...
```

- b. Some SELinux tools are provided by other packages. Use the `rpm` command to list the `policycore` packages.

- Note that the `policycoreutils-gui` package, which provides the SELinux Administration GUI, is not installed by default.

```
# rpm -qa |grep policycore
policycoreutils-python...
policycoreutils-...
```

- c. Use the `yum` command to install the `policycoreutils-gui` package.

- This command also installs some dependency packages.
- Answer `y` when prompted “Is this ok.”

```
# yum install policycoreutils-gui
...
Transaction Summary
=====
Install 1 Package (+3 Dependent packages

Total download size: 5.6 M
Installed size 27 M
Is this ok [y/d/N] : y
...
Complete!
```

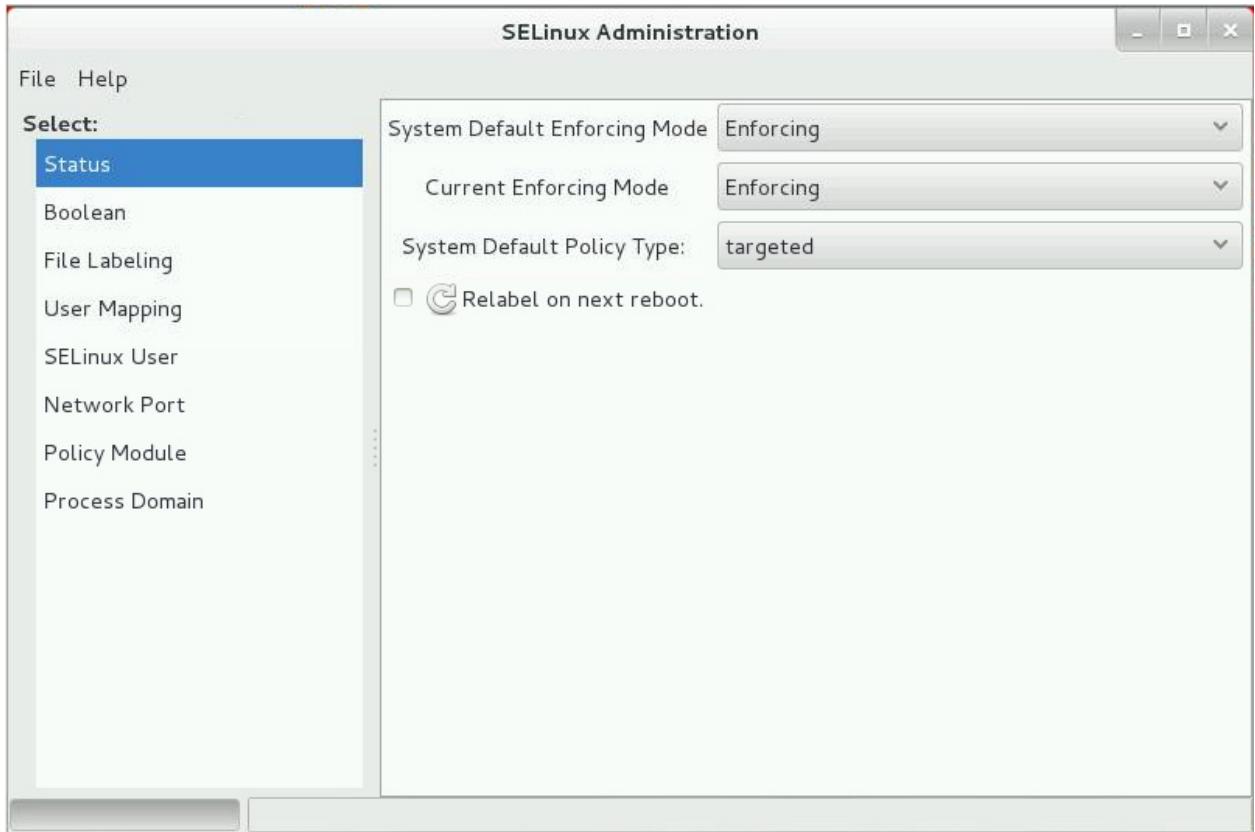
4. Use the SELinux Administration GUI.

- a. Use the `system-config-selinux` command to display the SELinux Administration GUI.

```
# system-config-selinux
```

- The GUI appears with the Status view selected by default.

- Note that policy type is targeted and policy mode is Enforcing.



- Display the "System Default Enforcing Mode" options by clicking the drop-down list.
  - Note the three options:
    - Disabled: No security policy is loaded in the kernel.
    - Permissive: A diagnostic state. Security policy rules are not enforced but messages are logged.
    - Enforcing: Access is denied to users and programs unless permitted by SELinux security policy rules.



- Display the "System Default Policy Type" options by clicking the drop-down list.
  - Note that targeted is the only option.
  - The other possible policy type option is Multi-Level Security (MLS).
  - MLS requires that the `selinux-policy-mls` package be installed (it is not installed by default).



- Select Boolean in the left pane of the GUI.
  - Booleans allow parts of SELinux policy to be changed.

- The Active check box indicates whether the Boolean is on or off.

**SELinux Administration**

File Help

Select:

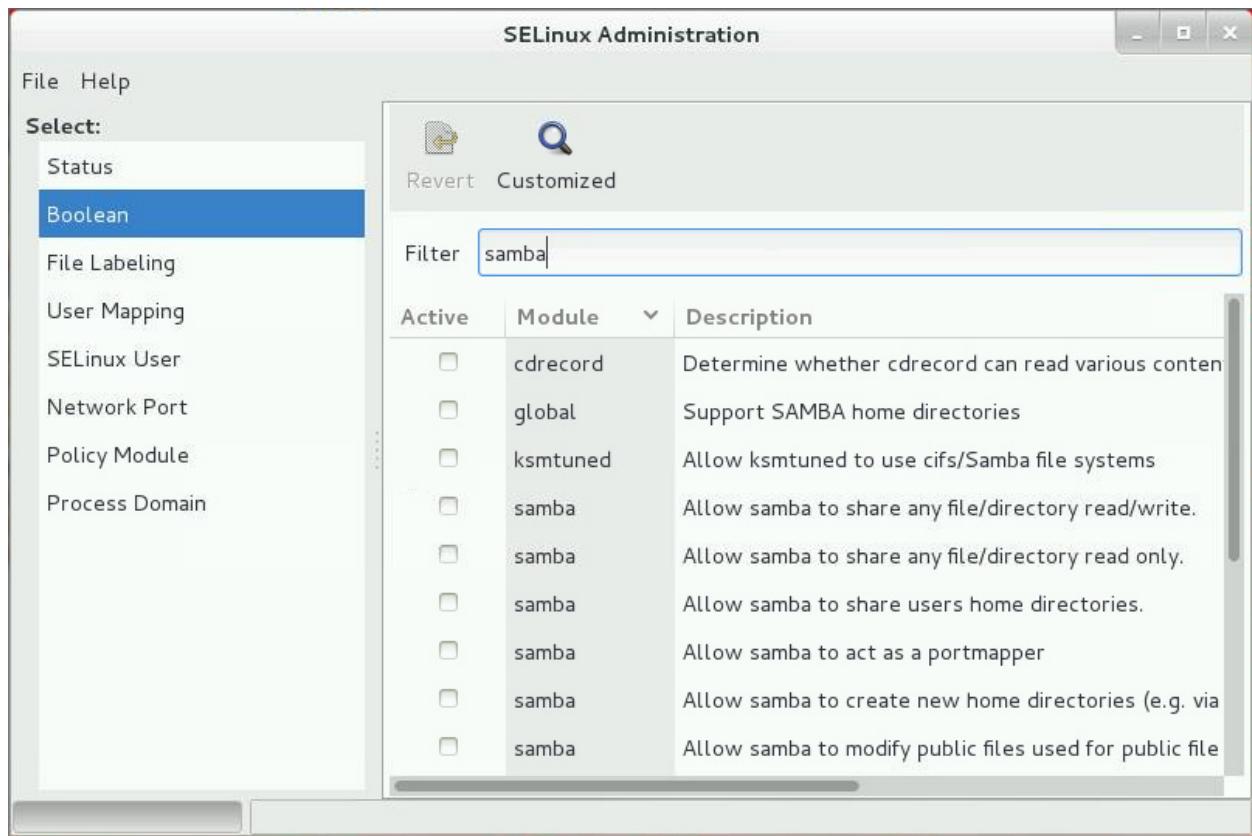
- Status
- Boolean**
- File Labeling
- User Mapping
- SELinux User
- Network Port
- Policy Module
- Process Domain

Revert Customized

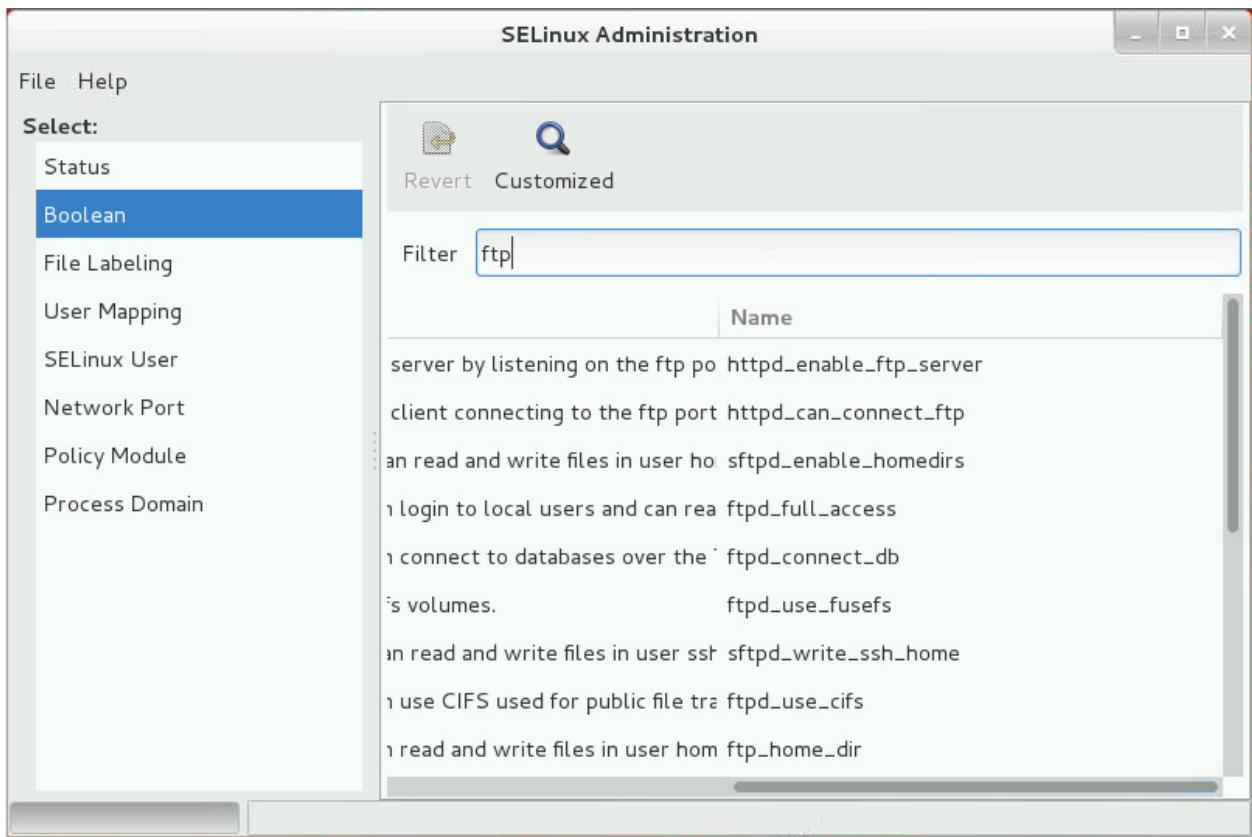
Filter

Active	Module	Description
<input checked="" type="checkbox"/>	abrt	Determine whether abrt-handle-upload can modify pu
<input type="checkbox"/>	abrt	Determine whether ABRT can run in the abrt_handle_
<input type="checkbox"/>	abrt	Allow ABRT to modify public files used for public file t
<input type="checkbox"/>	antivirus	Determine whether can antivirus programs use JIT cor
<input type="checkbox"/>	antivirus	Allow antivirus programs to read non security files on
<input type="checkbox"/>	apache	Allow httpd to access cifs file systems
<input type="checkbox"/>	apache	Allow Apache to modify public files used for public file
<input type="checkbox"/>	apache	Dontaudit Apache to search dirs.
<input type="checkbox"/>	apache	Allow Apache to query NS records

- e. Enter samba as a filter and press Enter to display the following screen.
- Note that each item contains “samba” in the long description.



- f. Scroll to the right to display the Boolean name.
- Note that each Boolean is referenced by a short name.
  - The short name can be used as an argument to several SELinux commands.



- g. Close the SELinux Administration GUI by selecting “File > Quit” from the menu bar.
5. Manage SELinux Booleans.
- Use the `semanage` command to list the Booleans.
    - Note that the Boolean name is listed in the first column, the current setting (on|off) is given in the second column, the default setting (on|off) is given in the third column, and the long description is listed in the last column.

```
# semanage boolean -l
SELINUX Boolean      State Default Description
ftp_home_dir        (off ,  off)  Determine whether ftp can read...
smartmon_3ware       (off ,  off)  Determine whether smartmon ...
...
cron_can_reliable   (off ,  off)  Allow system cron jobs to ...
sftpd_anon_write    (off ,  off)  Determine whether sftpd can ...
```
  - Use the `getsebool` command to view Booleans and their status.
    - Use the `-a` option to display all Booleans and their status.

- Note that this command does not include the long description.

```
# getsebool -a
abrt_anon_write --> off
abrt_handle_event --> off
abrt_upload_watch_anon_write --> on
antivirus_can_scan_system --> off
...
zoneminder_anon_write --> off
zoneminder_run_sudo --> off
```

- c. Include a Boolean name (for example, `use_nfs_home_dirs`) as an argument to display a specific Boolean status.

- **Note:** The `getsebool` command accepts multiple Boolean names as arguments.

```
# getsebool use_nfs_home_dirs
use_nfs_home_dirs --> off
```

- d. You can also view Booleans in the `/sys/fs/selinuxBOOLEANS` directory. Use the `cat` command to view the status of both `use_nfs_home_dirs` and `abrt_upload_watch_anon_write` Booleans.

- A value of 1 indicates that the Boolean is `on`, while 0 indicates `off`.
- The first number indicates the current value of the Boolean.
- The second number represents the pending value of the Boolean.

```
# cat /sys/fs/selinuxBOOLEANS/use_nfs_home_dirs
0 0
# cat /sys/fs/selinuxBOOLEANS/abrt_upload_watch_anon_write
1 1
```

- e. Use the `echo` command to change the pending value of the `use_nfs_home_dirs` Boolean to `on`.

```
# echo 1 > /sys/fs/selinuxBOOLEANS/use_nfs_home_dirs
# cat /sys/fs/selinuxBOOLEANS/use_nfs_home_dirs
0 1
```

- f. Use the `echo` command to commit all pending values for all Booleans.

- The `/sys/fs/selinux/commit_pending_booleans` file is the interface for committing the pending values of all Booleans as the current values.
- Commit pending values by writing a 1 to this file.

```
# echo 1 > /sys/fs/selinux/commit_pending_booleans
```

- g. Use the `cat` command to view the

`/sys/fs/selinuxBOOLEANS/use_nfs_home_dirs` file and use the `getsebool` command to view the `use_nfs_home_dirs` Boolean to confirm the commit.

```
# cat /sys/fs/selinuxBOOLEANS/use_nfs_home_dirs
1 1
# getsebool use_nfs_home_dirs
use_nfs_home_dirs --> on
```

- h. Use the `setsebool` command to change the value of the `use_nfs_home_dirs` Boolean. Confirm the change.

```
# setsebool use_nfs_home_dirs off
# getsebool use_nfs_home_dirs
use_nfs_home_dirs --> off
# cat /sys/fs/selinuxBOOLEANS/use_nfs_home_dirs
0 0
```

- The `setsebool -P` command causes the change to persist across a reboot.

## Practice 17-2: Configuring an SELinux Boolean

---

### Overview

In this practice, you change the value of an SELinux Boolean to allow `samba` to share user home directories.

### Assumptions

- You completed Practices for Lesson 7: Samba Services
- This practice is performed on **host01** and **host03**.
- You are currently the `root` user on **host03**.
- You are instructed to connect to **host01** from **dom0** by using `ssh`.
- The prompts include either **host01** or **host03** to indicate which system to enter the command from.

### Tasks

1. In Practice 7-1: Configuring a Samba Server, you configured **host03** as a Samba server. Perform the following tasks to review the configuration on **host03**.
  - a. Use the `systemctl` command to ensure that the `smb` service is enabled and active.
    - In this example, the `smb` service is enabled and active.

```
[host03]# systemctl status smb
smb.service - Samba SMB Daemon
  Loaded: loaded (/usr/lib/systemd/system/smb.service; enabled)
  Active: active (running) since ...
    Main PID: ... (smbd)
      Status: "smbd: ready to serve connections..."
     CGroup: /system.slice/smb.service
...

```

- b. Use the `firewall-cmd` command to ensure that the `samba` service is trusted for the active zone.
  - In this example, the `samba` service is trusted along with other services.

```
[host03]# firewall-cmd --list-services
dhcpcv6-client  ldap  samba  ssh
```

- c. View the contents of the `/etc/passwd` file to ensure that `user01` exists.
  - In this example, `user01` exists.

```
[host03]# cat /etc/passwd
...
user01:x:1008:1009::/home/user01:/bin/bash
```

2. As the `root` user on **dom0**, connect to the **host01** guest by using `ssh`.
  - a. If necessary, open a terminal window on **dom0** and use the `su -` command to become the `root` user.
    - The `root` password is `oracle`.

```
[dom0] $ su -
Password: oracle
#
```

- b. As the **root** user on **dom0**, use the **ssh** command to login to **host01**.

- The password is **oracle**.

```
[dom0] # ssh host01
root@host01's password: oracle
Last login: ...
[root@host01 ~] #
```

- c. Use the **yum** command to ensure that the **samba-client** package is installed on **host01**.

- You were instructed to install this package in Practice 7-2: Accessing Samba Shares from a Client Host.
- In this example, the package is already installed.

```
[host01] # yum install samba-client
OL7.1Dom0
Package samba-client-4.1.12-21.el7_1.x86_64 already installed
and latest version
Nothing to do
```

3. From **host01**, access the Samba shares on **host03** as user **user01**.

- a. Use the **smbclient** command to access the home directory for user **user01** on **host03**.

- The Samba password for **user01** is **MyOracle1**.

```
[host01] # smbclient //host03/user01 -U user01
Enter user01's password: MyOracle1
Domain= [GROUPA] OS= [Unix] Server= [Samba 4.1.12]
smb: \>
```

- b. Use the **ls** command to list the files in the home directory for user **user01**.

- The command fails because SELinux is in “Enforcing” mode.

```
smb: \> ls
NT_STATUS_ACCESS_DENIED listing \*
smb: \>
```

- c. Use the **exit** command to exit the **smb** session.

```
smb: \> exit
```

4. Enable SELinux Boolean on **host03**.

- In Practice 7-2: Accessing Samba Shares from a Client Host, you set SELinux to “Permissive” mode on **host03** to allow Samba users access to their home directories.
- In this task, you leave SELinux in “Enforcing” mode and set a Boolean to allow Samba users access to their home directories.
- a. Use the `getsebool -a` command to list all the SELinux Booleans and their statuses. Pipe the output to `grep` and search for `samba`.
- Note that the `samba_enable_home_dirs` Boolean is set to `off`.

```
[host03]# getsebool -a | grep samba
samba_create_home_dirs --> off
samba_domain_controller --> off
samba_enable_home_dirs --> off
...
```

- b. Use the `semanage boolean -l` command to list the SELinux Booleans and their long descriptions. Pipe the output to `grep` and search for `samba_enable_home_dirs`.
- Based on the long description, this Boolean appears to be the reason you cannot list the files in the Samba user’s home directory on **host03**.

```
[host03]# semanage boolean -l | grep samba_enable_home_dirs
samba_enable_home_dirs (off , off) Allow samba to share users
home directories.
```

- c. Use the `setsebool` command to enable the `samba_enable_home_dirs` Boolean.
  - Use the `-P` option to make the change persistent across reboots. There is a slight delay when using the `-P` option.
  - Use the `getsebool` command to confirm that the Boolean is on.

```
[host03]# setsebool -P samba_enable_home_dirs on
[host03]# getsebool samba_enable_home_dirs
samba_enable_home_dirs --> on
```

5. From **host01**, access the Samba shares on **host03** as user `user01`.

- a. On **host01**, re-issue the `smbclient` command to access the home directory for user `user01` on **host03**.
  - The Samba password for `user01` is `MyOracle1`.

```
[host01]# smbclient //host03/user01 -U user01
Enter user01's password: MyOracle1
Domain= [GROUPA] OS= [Unix] Server= [Samba 4.1.12]
smb: \>
```

- b. Use the `ls` command to list the files in the home directory for user `user01`.
  - Because of the change in the `samba_enable_home_dirs` boolean, you can now list and access the files in `user01`’s home directory.

```
smb: \> ls
.
D
...
```

```
.. D ...
.mozilla DH ...
.bash_logout H ...
.bash_profile H ...
.bashrc H ...
...
smb: \>
```

- c. Use the `exit` command to exit the `smb` session.

```
smb: \> exit
```

6. Log off **host01** and **host03**.

- a. From **host01**, enter `exit` to log off.

```
[host01]# exit
logout
Connection to host01 closed.
[dom0]#
```

- b. From **host03**, click Oracle Student on the notification area to display the drop-down menu. Click Log Out from the menu.
- Click Log Out when prompted.
  - Click the x in the upper-right corner of the GNOME login window to close the window.

## Practice 17-3: Configuring SELinux Context

---

### Overview

In this practice, you change the SELinux context file type to enable a file system to be used for storing KVM virtual machine disk images when SELinux is in enforcing mode.

### Assumptions

- You are the `root` user on `dom0`.

### Tasks

1. Shut down the `host01`, `host02`, and `host03` VMs.

- From `dom0`, run the `xm destroy hostN` command to shut down the three VMs.

```
# xm destroy host01
# xm destroy host02
# xm destroy host03
```

- Run the `xm list` command to display the running VMs.

- In this example, only **Domain-0** is running.

Name	ID	Mem	VCPUs	State	Time (s)
Domain-0	0	2048	2	r-----	...

2. Start the `host05` virtual machine.

- Before starting `host05`, you create a second disk and update the `vm.cfg` file.

  - From `dom0`, use the `cd` command to change to the `/OVS/running_pool/host05` directory.

```
# cd /OVS/running_pool/host05
```

- Use the `dd` command to create a 10 GB utility disk image.

- This command takes a few seconds to complete.

```
# dd if=/dev/zero of=u01.img bs=1M count=10240
10240+0 records in
10240+0 records out
10737418240 bytes (11 GB) copied...
```

- Use the `vi` editor to add the following **bold** line to the `vm.cfg` file.

- This line allows the `host05` VM to recognize the new `u01.img` disk image.

```
# vi vm.cfg
...
disk = [ 'file:/OVS/running_pool/host05/system.img,hda,w',
          'file:/OVS/running_pool/host05/u01.img,hdb,w',
          'file:/OVS/seed_pool/OracleLinux-R7-U1-Server-x86_64-
dvd.iso,hdc:cdrom,r' ]
...
```

- d. Run the `xm create` command to start the **host05** virtual machine.

```
# xm create vm.cfg
Started domain host05 (id=...
```

3. Log in to **host05**.

Use the `ssh` command to log in to 192.0.2.105. The **host05** VM is not resolved on **dom0** so you must use the IP address rather than the host name.

- The root password is `oracle` (all lowercase).
- If you get “No route to host,” wait a few seconds until **host05** boots.
- Enter the `hostname` command to verify you connected to **host05**.

```
# ssh 192.0.2.105
The authenticity of host '192.0.2.105 (192.0.2.105)' can't be
established. RSA key fingerprint is ...
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.0.2.105' (RSA) to the list of
known hosts.
root@192.0.2.105's password: oracle
[root@192.0.2.105 ~]# hostname
host05.example.com
```

4. Configure **host05** to access the local Yum repository on **dom0**.

- a. From **host05**, use the `cd` command to change to the `/etc/yum.repos.d` directory.

```
# cd /etc/yum.repos.d
```

- b. Use the `vi` command to create the `vm.repo` file as follows.

- Note that the `baseurl` references the local Yum repository on **dom0** (192.0.2.1).

```
# vi vm.repo
[OL7.1Dom0]
Name="Oracle Linux 7.1 Dom0 Repo"
baseurl=http://192.0.2.1/repo/OracleLinux/OL7/1/x86_64
enabled=1
gpgkey=http://192.0.2.1/repo/OracleLinux/OL7/1/x86_64/RPM-GPG-
KEY-oracle
gpgcheck=1
```

- c. Use the `vi` command to edit the `public-yum-ol7.repo` file and disable all repositories.

- Set `enabled=0` for all repositories.

```
# vi public-yum-ol7.repo
[ol7_latest]
...
enabled=0
...
[ol7_UEKR3]
...
```

```
enabled=0
...
...
```

- d. Use the `grep` command to view enabled repositories in both files.
- Only the `vm.repo` file contains an enabled repository (`enabled=1`).
  - If this is not the case, repeat task 4 to ensure that only the `vm.repo` is enabled.

```
# grep enabled *
public-yum-ol7.repo:enabled=0
...
vm.repo:enabled=1
```

- e. Run the `yum clean all` command to clean the Yum cache.

```
# yum clean all
Loaded plugins: ...
Cleaning repos: ...
Cleaning up Everything
```

5. Prepare the new disk for use.

- a. Use the `fdisk` command to partition the `/dev/xvdb` device.
- Create a single, primary partition for the entire disk.
  - Press **ENTER** when prompted for the first cylinder and last cylinder.

```
# fdisk /dev/xvdb
...
Command (m for help): n
Partition type
      P   primary (0 primary, 0 extended, 4 free)
      e   extended
Select (default p): ENTER
Using default response p
Partition number (1-4, default 1): ENTER
First sector (2048-20971519, default 2048): ENTER
Using default value 2048
Last sector, +sector or +size{K,M,G}... ENTER
Using default value 20971519
Partition 1 of type Linux and of size 10 GiB is set

Command (m for help): w
The partition table has been altered!
...
Syncing disks.
```

- b. Use the `pvcreate` command to create a physical volume on `/dev/xvdb1`.

```
# pvcreate /dev/xvdb1
Physical volume "/dev/xvdb1" successfully created
```

- c. Use the `vgcreate` command to create a volume group, named `myvolg`, on `/dev/xvdb1`.

```
# vgcreate myvolg /dev/xvdb1
Volume group "myvolg" successfully created
```

- d. Use the `lvcreate` command to create a 5 GB logical volume group, named `myvol`, from the `myvolg` volume group.

```
# lvcreate -L 5G -n myvol myvolg
Logical Volume "myvol" created
```

- e. Use the `mkfs.ext4` command to create an ext4 file system on the `myvol` logical volume.

```
# mkfs.ext4 /dev/mapper/myvolg-myvol
...
Writing superblocks and filesystem accounting information: done
```

- f. Use the `mkdir` command to create a new directory, `/virtstore`, for mounting the new logical volume.

```
# mkdir /virtstore
```

- g. Use the `mount` command to mount the logical volume on `/virtstore`.

```
# mount /dev/mapper/myvolg-myvol /virtstore
```

6. Change the SELinux context file type on `/virtstore`.

- a. Use the `ls -Z` command to view context information for the `/var/lib/libvirt/images` directory.

- The `/var/lib/libvirt/images` directory is the default directory for KVM virtual machine disk images.
- Note that this directory has the `virt_image_t` label applied to it.

```
# ls -dZ /var/lib/libvirt/images
drwx... root root system_u:object_r:virt_image_t:s0 /var/...
```

- b. Use the `ls -Z` command to view context information for the `/virtstore` directory.

- Note that this directory has the `unlabeled_t` label applied to it.

```
# ls -dZ /virtstore
drwx... root root system_u:object_r:unlabeled_t:s0 /virtstore/
```

- c. Use the `semanage` command to set the SELinux file context type to `virt_image_t` for the `/virtstore` directory.

```
# semanage fcontext -a -t virt_image_t "/virtstore(/.*)?"
```

- d. Use the `cat` command to view the context information for the targeted SELinux policy.

```
# cat /etc/selinux/targeted/contexts/files/file_contexts.local
...
/virtstore(/.*)?    system_u:object_r:virt_image_t:s0
```

- e. Use the `ls -dZ` command to view context information for the `/virtstore` directory.

- Note that this directory still has the `unlabeled_t` label applied to it.

```
# ls -dZ /virtstore
drwx...  root  root  system_u:object_r:unlabeled_t:s0  /virtstore/
```

- f. Use the `restorecon` command to change the type on `/virtstore` as defined in the `/etc/selinux/targeted/contexts/files/file_contexts.local` file.

- The `-R` option recursively changes file and directory file labels.
- The `-v` option shows the changes in file labels.

```
# restorecon -R -v /virtstore
restorecon reset /virtstore context system_u:object_r...
...
```

- g. Use the `ls -dZ` command to view context information for the `/virtstore` directory.

- Note that the directory now has the `virt_image_t` label applied to it.
- The `/virtstore` file system is now prepared to store KVM virtual machine disk images when SELinux is in enforcing mode.

```
# ls -dZ /virtstore
drwx...  root  root  system_u:object_r:virt_image_t:s0  /virtstore/
```

## 7. Clean up.

- In this task, you shut down **host05** and restart the **host03** VM.

- a. Run the `systemctl poweroff` command to shut down **host05**.

```
[host05]# systemctl poweroff
Connection to 192.0.2.105 closed by remote host.
Connection to 192.0.2.105 closed.
```

- b. From **dom0**, use the `cd` command to change to the `/OVS/running_pool/host03` directory.

```
[dom0]# cd /OVS/running_pool/host03
```

- c. Run the `xm create vm.cfg` command to start the **host03** VM.

```
# xm create vm.cfg
Using config file "./vm.cfg".
Started domain host03 (id=...)
```

- d. Run the `xm list` command to display the running VMs.
- In this example, the **host03** VM is running.
  - It is not necessary to start the **host01** and **host02** VMs because these VMs are not used in the remaining practices.

```
# xm list
```

Name	ID	Mem	VCPUs	State	Time (s)
Domain-0	0	2048	2	r-----	...
host03		...			

## **Practices for Lesson 18: Core Dump Analysis**

**Chapter 18**

## Practices for Lesson 18: Overview

---

### Practices Overview

In these practices, you:

- Configure Kdump
- Create a core dump file
- Install the kernel-debuginfo RPMs
- Use the `crash` utility to analyze the state of a `vmcore`
- Use the `crash` utility to analyze the state of a running system

## Practice 18-1: Configuring Kdump

---

### Overview

In this practice, you:

- Configure Kdump by using the Kernel Dump Configuration GUI
- View the Kdump configuration, which is updated as you apply changes in the GUI

### Assumptions

- You are the `root` user on `dom0`.
- The Kdump mechanism is not supported on your Xen domU guest systems.
- However, you proceed with the practice as if Kdump is supported.

### Tasks

- Connect to **host03** by using `vncviewer`. If necessary, refer to Practice 3-1: Configuring an OpenLDAP Server for instructions.
  - From the GNOME login window on **host03**, log in as the Oracle Student user. The password is `oracle`.
  - From the GNOME desktop on **host03**, open a terminal window.
  - Use the `su` – command to become the `root` user. The password is `oracle`.
- View the current Kdump configuration.
  - As the `root` user on **host03**, use the `rpm` command to verify that the `kexec-tools` package is installed.
    - In this example, the package is installed.

```
# rpm -q kexec-tools
kexec-tools-2.0.7-19.0.1.el7.x86_64
```
  - Use the `systemctl` command to view the status of the `kdump` service.
    - This confirms that the kernel crash dump mechanism is disabled.

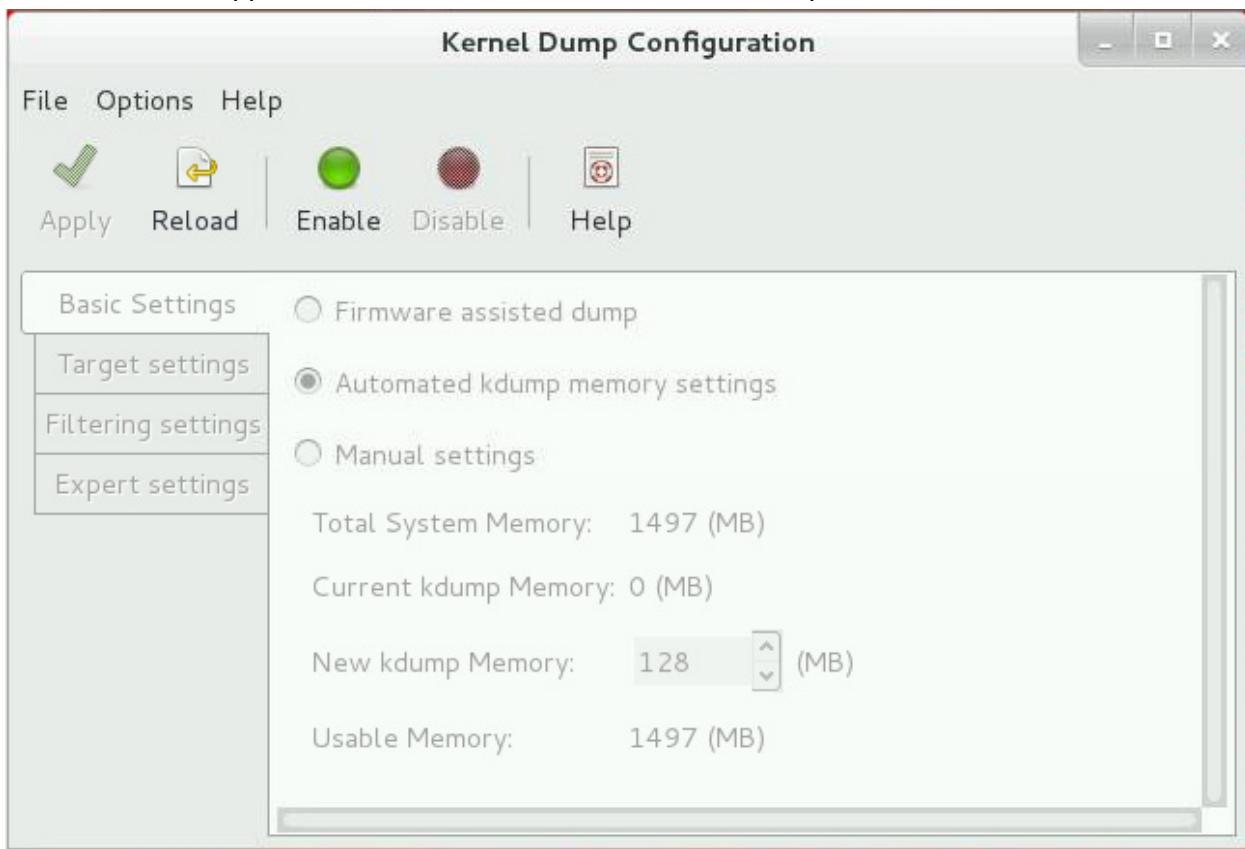
```
# systemctl status kdump
kdump.service - Crash recovery kernel arming
   Loaded: loaded (/usr/lib/systemd/. . . /kdump.service; disabled)
   Active: inactive (dead)
```
  - Use the `grep` command to search for the string “crash” in the `/boot/grub2/grub.cfg` file.
    - No output confirms that memory has not been reserved for the Kdump kernel.

```
# grep crash /boot/grub2/grub.cfg
```

3. Configure Kdump by using the Kernel Dump Configuration GUI.
  - a. Use the `yum` command to install the `system-config-kdump` software package.
    - Answer `y` to “Is this ok.”

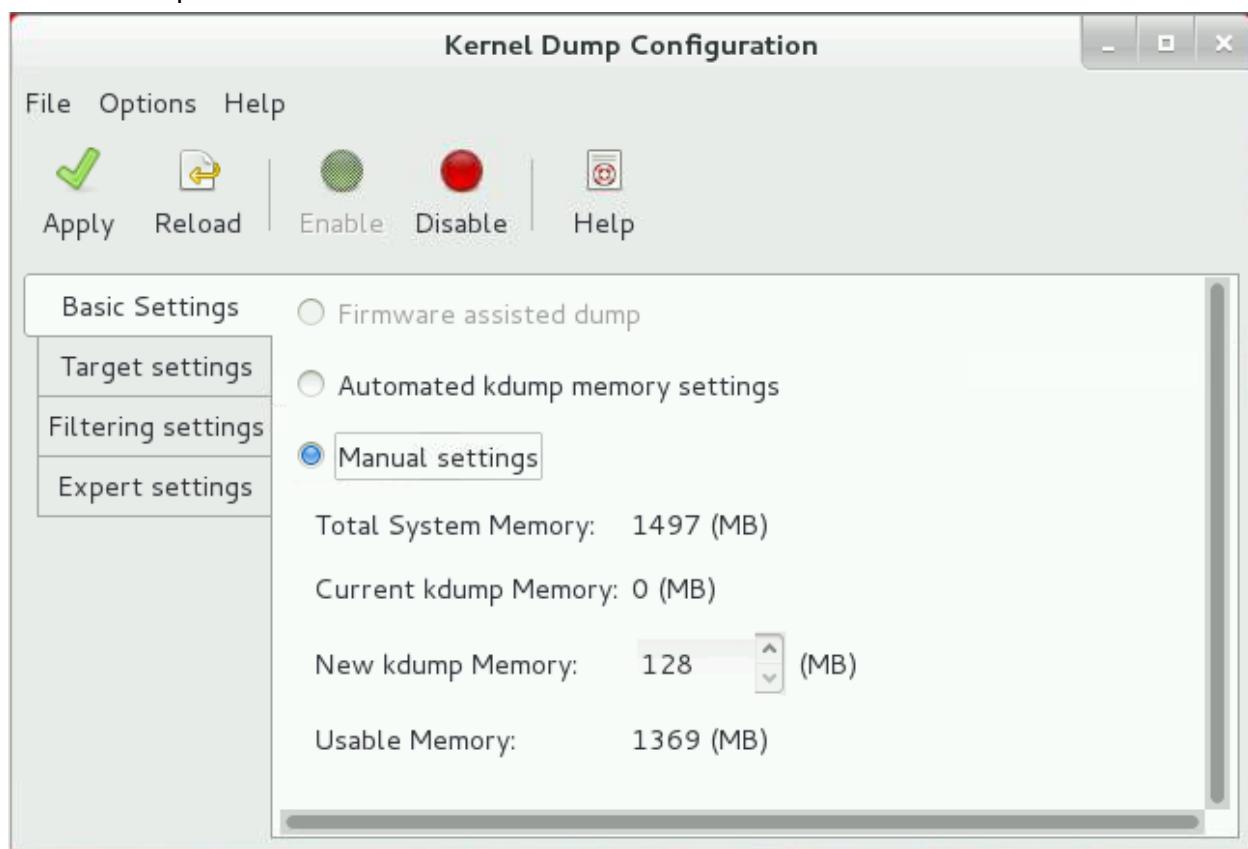
```
# yum install system-config-kdump
...
Is this ok [y/d/N] : y
...
Complete!
```

- b. Use the `system-config-kdump` command to display the Kernel Dump Configuration GUI.
  - The GUI appears and shows that the kernel crash dump mechanism is disabled.

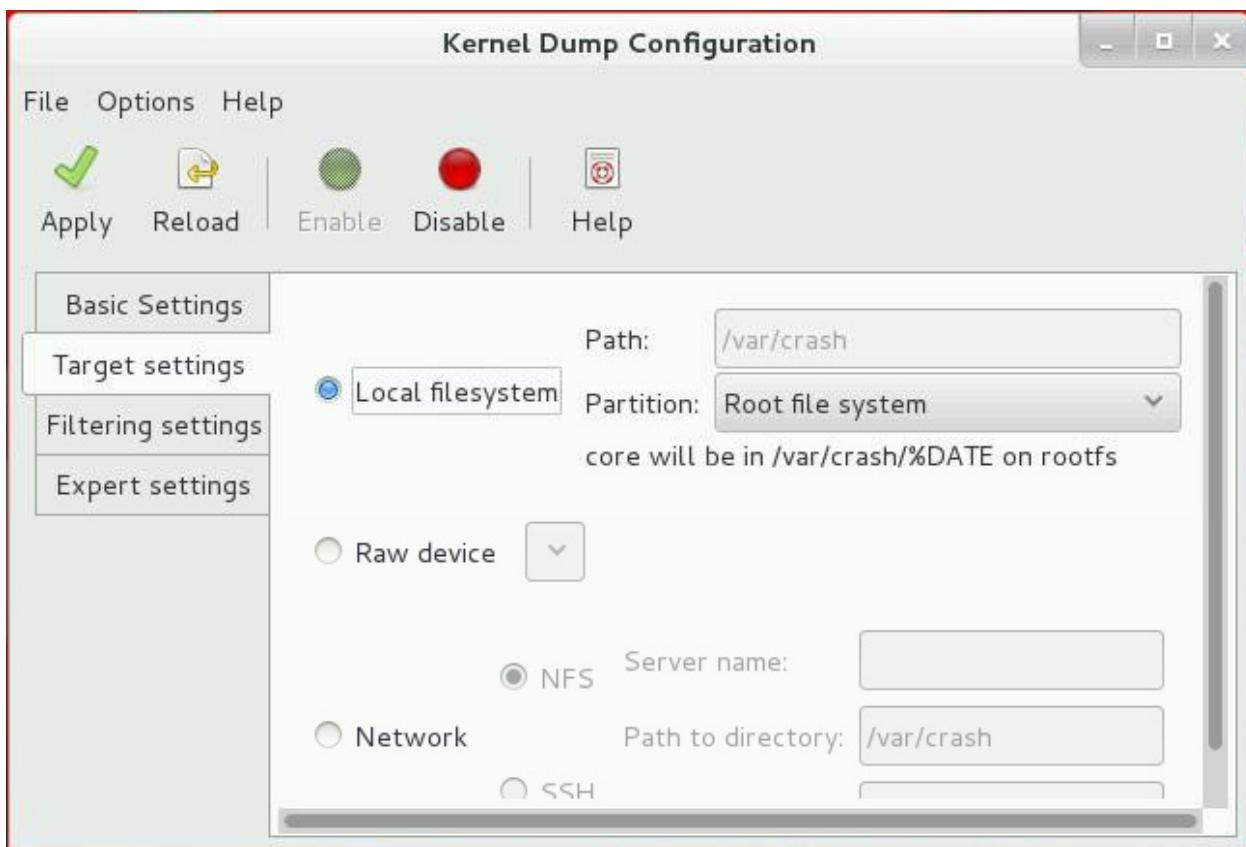


- c. From the Kernel Dump Configuration GUI, click the Enable button.
    - This configures the `kdump` service to start at boot time.
  - d. Click the Manual settings radio button.
    - If necessary, set the New kdump Memory to 128 MB.
    - This amount of memory is represented by the `crashkernel` option and is appended to the kernel line in the GRUB configuration file, `/boot/grub2/grub.cfg`, as follows:
- `linux16 /vmlinuz-3.8.13-68.3.3.el7uek ... crashkernel=128M`

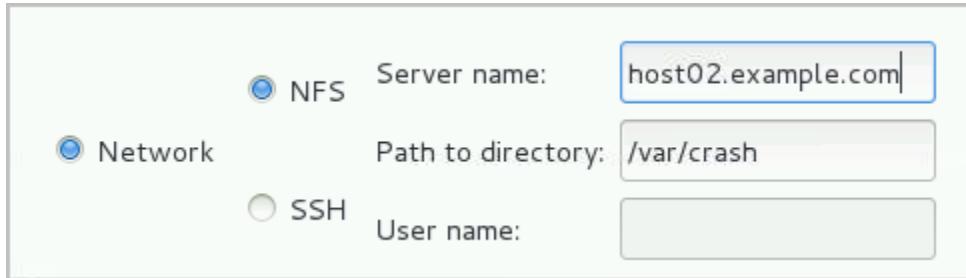
- You must reboot if the amount of memory reserved for Kdump is changed or if Kdump is enabled from a disabled state.



- Click the Target settings tab in the GUI.
  - This page is used to specify the target location for the `vmcore` dump.
  - Notice that the default target location is the `/var/crash` directory on the local file system.
  - This setting is represented as follows in the `/etc/kdump.conf` configuration file:  
path /var/crash

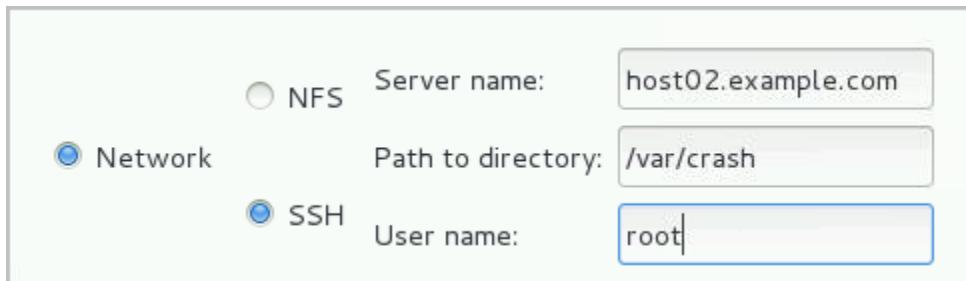


- f. Click the Network and NFS buttons and enter host02.example.com for Server name.



- These settings are represented as follows in the /etc/kdump.conf configuration file:
- ```
path /var/crash
nfs host02.example.com
```

- g. Click the Network and SSH buttons and enter root as the User name.



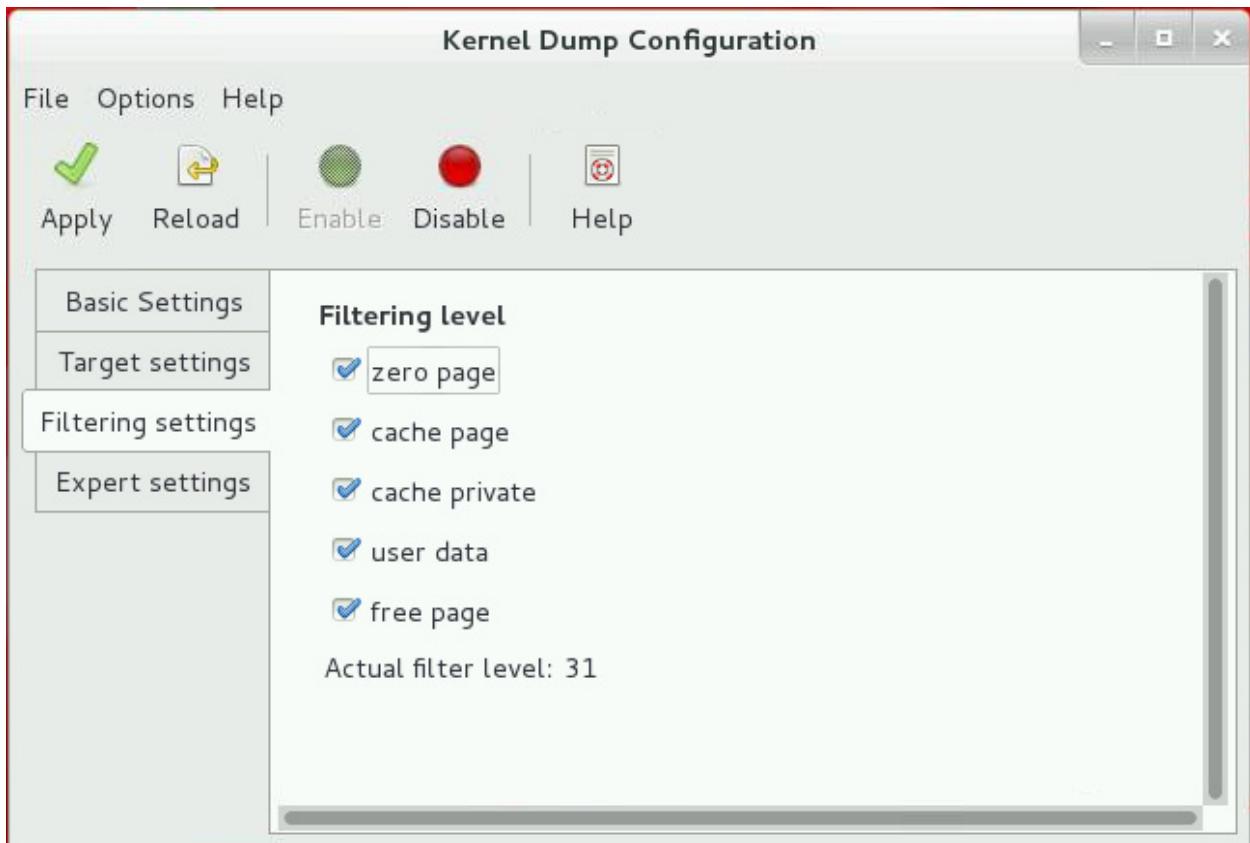
- These settings are represented as follows in the /etc/kdump.conf configuration file:

```
path /var/crash
ssh root@host02.example.com
```

h. Click the Filtering settings tab in the GUI.

- By default, all page types are checked meaning all page types are excluded.
- The default settings are represented as follows in the /etc/kdump.conf configuration file:

```
core_collector makedumpfile -p --message-level 1 -d 31
```



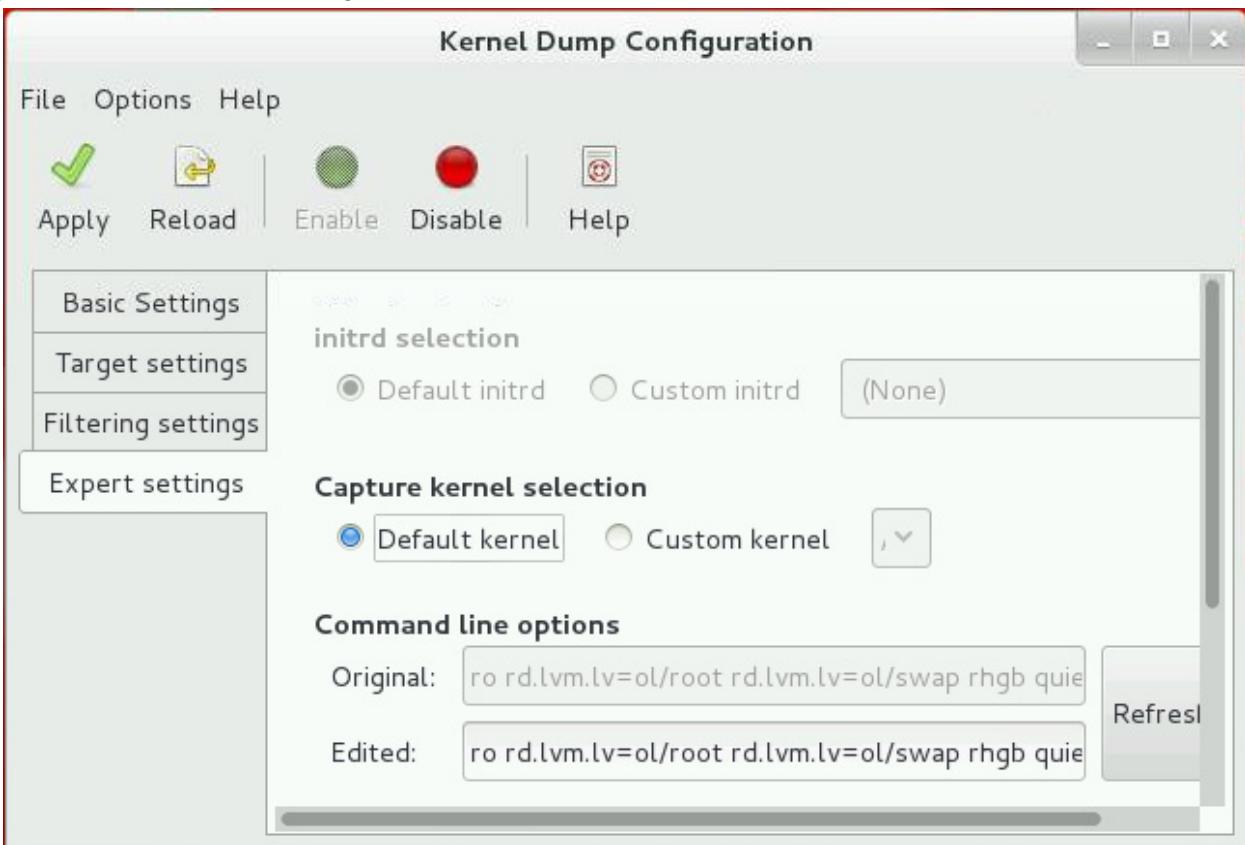
i. Deselect cache page, cache private, and user data.

- These settings exclude only zero page and free page information. This setting is represented as follows in the /etc/kdump.conf configuration file:

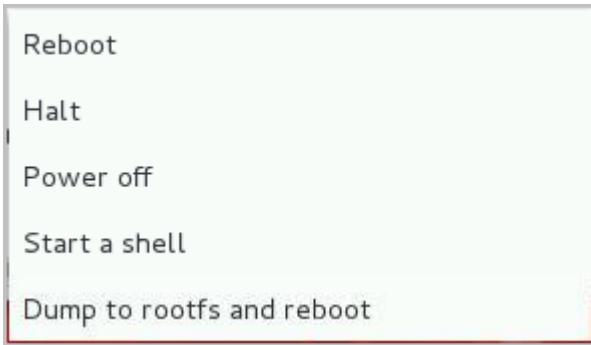
```
core_collector makedumpfile -p -message-level 1 -d 17
```



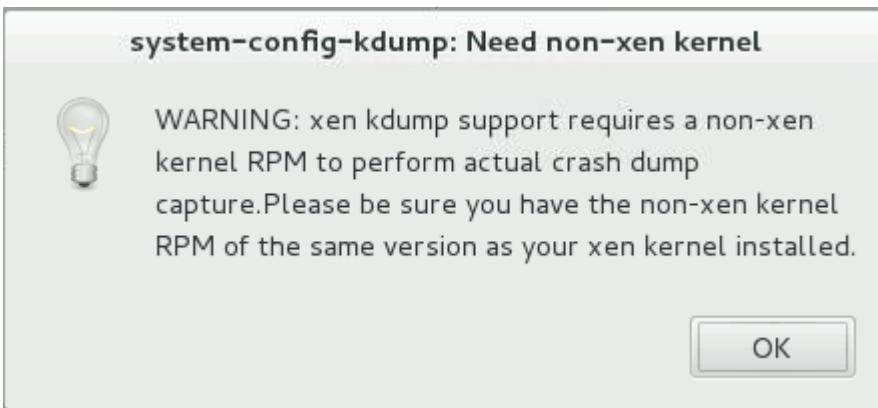
- j. Click the Expert settings tab in the GUI.



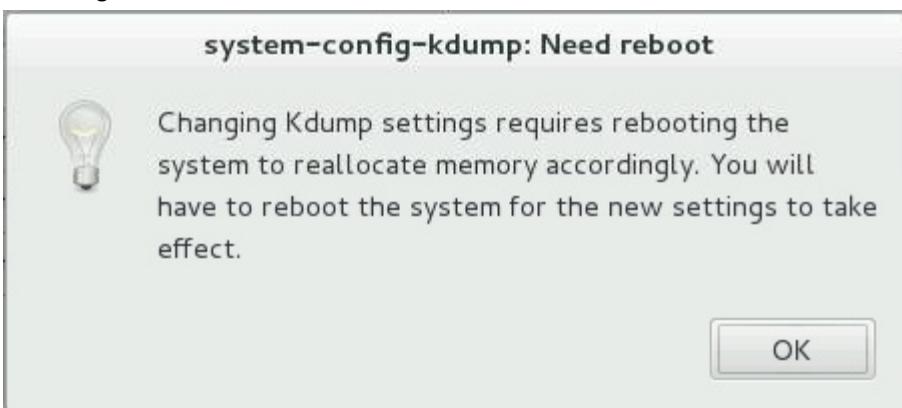
- k. Scroll down and click the Action if dumping fails drop-down menu to display the actions that the system can take if dumping to the intended target fails.



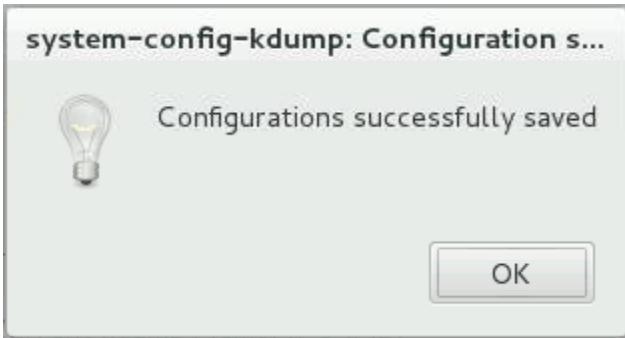
- I. Click the Halt option.
  - This selection is represented as follows in the `/etc/kdump.conf` configuration file:  
`default halt`
- m. Click Apply.
  - The following message appears. The Kdump mechanism is not supported on Xen domU guest systems.



- n. Click OK.
  - The following message appears if you change memory settings and if you enable Kdump from a disabled state.
  - Again, this message does not apply, because Kdump is not supported on the VM guests.



- o. Click OK.
  - The following message appears. The configuration is saved, but the service does not start on VM guests.



- p. Click OK, and then select File > Quit from the Kernel Dump Configuration menu bar.
4. View the configuration changes made by using the Kdump GUI.

- a. Use the `systemctl` command to view the status of the `kdump` service.
- By clicking Enable on the Kdump GUI, the `kdump` service is configured to start at boot time.

```
# systemctl status kdump
kdump.service - Crash recovery kernel arming
    Loaded: loaded (/usr/lib/systemd/. . . /kdump.service; enabled)
    Active: inactive (dead)
```

- b. Use the `grep` command to search for the string "crash" in the `/boot/grub2/grub.cfg` file.
- Note that 128 M of memory has been reserved for the Kdump kernel.

```
# grep crash /boot/grub2/grub.cfg
linux16 /vmlinuz-3.8.13-68.3.3.el7uek . . . crashkernel=128M
```

- c. Use the `tail -4 /etc/kdump.conf` command to display the last four lines in the Kdump configuration file.
- Notice that the settings made from the GUI were written to the Kdump configuration file.

```
# tail -4 /etc/kdump.conf
ssh root@host02.example.com
path /var/crash
core_collector makedumpfile -l --message-level 1 -d 17
default halt
```

- d. Use the `makedumpfile --help | less` command to view usage for the `makedumpfile` command.
- The `makedumpfile` command is designated as `core_collector` in the `/etc/kdump.conf` file.

- Pipe the output of the `makedumpfile --help` command to `less` to view the help pages one screen at a time.

```
# makedumpfile --help | less
...
```

- Scroll down and note the following:
  - The `-l` option compresses the dump data using `lz0`.
  - The default dump level, `-d 31`, excludes zero page, cache with and without private, user data, and free page, and only captures the essential information.
  - The default `--message-level` of 1 prints a progress indicator. A message level of 31 displays progress indicator, common messages, error messages, debug messages, and report messages.
- Enter `q` to quit the `makedumpfile --help | less` command.

5. View the memory statistics.

- Use the `free -m > ~/before` command to write memory statistics to a file.
  - This task saves current memory usage to a file.
  - The contents of this file are compared to memory usage after configuring Kdump to reserve 128 MB of system memory for the capture kernel.

```
# free -m > ~/before
```

- Use the `systemctl reboot` command to reboot **host03**.

```
# systemctl reboot
...
```

- From **dom0**, use the `ssh` command to connect to **host03**.
  - The root password is `oracle`.
  - You need to wait a few seconds for the reboot to complete.

```
[dom0]# ssh host03
root@host03's password: oracle
```

- From **host03**, compare the `free -m` output to the contents of `~/before`.

```
# cat ~/before
      total ...
Mem:       1497 ...
...
# free -m
      total ...
Mem:       1369 ...
...
```

- Notice that there is 128 MB less total memory ( $1497 - 1369 = 128$ ) after reserving this amount for the capture kernel.

## Practice 18-2: Creating a Core Dump File

---

### Overview

In this practice, you perform the following:

- Because Kdump is not supported on your Xen domU guest systems, use the `xm dump-core` command to create a `vmcore` file for **host03**.
- Use the `sftp` command to upload the `vmcore` file from **dom0** to **host03**.

### Assumptions

- You are the `root` user on **host03**.

### Tasks

- If necessary, open a new terminal window from **dom0**.

From this window, use the `su -` command to become the `root` user on **dom0**.

- The root password is `oracle`.

```
[dom0]$ su -
Password: oracle
```

- List domain IDs.

From **dom0**, use the `xm list` command to print information about domains. The output shown is a sample and might not represent your system exactly.

- In this example, there is one guest running, **host03**.
- The domain-id for **host03** is 59. This might differ on your system.

| # xm list |    |      |       |        |          |
|-----------|----|------|-------|--------|----------|
| Name      | ID | Mem  | VCPUs | State  | Time (s) |
| Domain-0  | 0  | 2048 | 2     | r----- | 281.1    |
| host03    | 59 | 1536 | 1     | -b---- | 13.2     |

- Dump **host03**'s virtual memory.

- From **dom0**, use the `xm dump-core` command to dump **host03**'s virtual memory.
- Use the domain-id for **host03** that was obtained from the previous task.
  - The example command uses domain-id 59.
  - This command takes a few seconds to complete.

```
# xm dump-core 59
Dumping core of domain: 59 ...
```

- Use the `cd` command to change to the `/var/xen/dump/host03` directory on **dom0**. Use the `ls` command to display the contents of the directory.

- The `vmcore` file name is only a sample. Your `vmcore` file name is different.

```
# cd /var/xen/dump/host03
# ls
2015-0721-2154.29-host03.59.core
```

4. Upload the vmcore file from **dom0** to **host03**.
  - a. Use the `sftp` command to connect to **host03**.
    - The root password is `oracle`.

```
[dom0]# sftp host03
Connecting to host03
root@host03's password: oracle
sftp>
```

- b. Use the `put *` command to upload the vmcore file in the **dom0** directory to **host03**.

```
sftp> put *
Uploading 2015-0721-2154.29-host03.59.core to /root/2015...
sftp>
```

- c. Use the `quit` command to exit `sftp`.

```
sftp> quit
```

## Practice 18-3: Preparing Your System to Analyze the vmcore

---

### Overview

In this practice, you perform the following:

- Determine the kernel version and the architecture of the vmcore.
- Transfer the `kernel-uek-debuginfo` packages from **dom0** to **host03**.
- Install the `kernel-uek-debuginfo` packages on **host03**.
- Ensure that you have the latest version of `crash` installed.

### Assumptions

- You are logged in as `root` on **dom0**.
- You are logged in as `root` on **host03**.

### Tasks

1. Determine the kernel version and architecture of the vmcore.
  - a. If you have access to the system that produced the vmcore, use the `uname -r` command to determine the kernel version and architecture.
 

```
[host03]# uname -r
3.8.13-55.1.6.el7uek.x86_64
```

    - In this example, the kernel version is `3.8.13-55.1.6.el7uek`.
    - The architecture is `x86_64`.
  - b. If you have access only to the vmcore file, use the `strings <vmcore>` command.
    - In this example, the vmcore file name is `2015-0721-2154.29-host03.59.core`.
    - This file is located in the `root` user's home directory on **host03**.
    - Pipe the output to `grep -i` and search for the string "boot\_image".
    - In this example, the kernel version is `3.8.13-55.1.6.el7uek`.
    - The architecture is `x86_64`.

```
[host03]# strings 2015-0721-2154.29-host03.59.core | grep -i
boot_image
BOOT_IMAGE=/vmlinuz-3.8.13-55.1.6.el7uek.x86_64 ...
```
2. Use the `sftp` command to transfer the `kernel-debuginfo` packages from **dom0** to **host03**.
  - a. Normally you obtain the `kernel-debuginfo` packages for Oracle Linux 7 from <https://oss.oracle.com/ol7/debuginfo>
  - The two required `kernel-debuginfo` RPMs have been downloaded to the `/OVS/seed_pool/debug` directory on **dom0**.
  - a. From **dom0**, use the `cd` command to change to the `/OVS/seed_pool/debug` directory.

```
[dom0]# cd /OVS/seed_pool/debug
```

- b. Use the `ls` command to view the contents of the directory.

```
[dom0]# ls
kernel-uek-debuginfo-3.8.13-55.1.6.el7uek.x86_64.rpm
kernel-uek-debuginfo-common-3.8.13-55.1.6.el7uek.x86_64.rpm
```

- c. Use the `sftp` command to connect to **host03**.

- The root password is `oracle`.

```
[dom0]# sftp host03
Connecting to host03...
root@host03's password: oracle
sftp>
```

- d. Use the `mput *` command to upload all files in the **dom0** directory to **host03**.

```
sftp> mput *
Uploading kernel-uek-debuginfo-3.8.13-55.1.6.el7uek.x86_64.rpm
to /root/...
...
sftp>
```

- e. Use the `quit` command to exit `sftp`.

```
sftp> quit
```

3. From **host03**, install the `kernel-debuginfo` RPMs.

- a. Use the `cd` command to ensure that you are in the `root` user's home directory. Use the `ls` command to display the `kernel-debuginfo` RPMs.

```
[host03]# cd
# ls *debuginfo*
kernel-uek-debuginfo-3.8.13-55.1.6.el7uek.x86_64.rpm
kernel-uek-debuginfo-common-3.8.13-55.1.6.el7uek.x86_64.rpm
```

- b. Use the `rpm -Uvh` command to install each of the `kernel-debuginfo` RPMs.

```
# rpm -Uvh kernel-uek-debuginfo-common-3.8.13-
55.1.6.el7uek.x86_64.rpm
...
# rpm -Uvh kernel-uek-debuginfo-3.8.13-55.1.6.el7uek.x86_64.rpm
...
```

4. From **host03**, use the `yum update` command to ensure that you have the latest version of `crash` installed.

- In this example, the latest version is already installed.

```
# yum update crash
...
No packages marked for update
```

## Practice 18-4: Using the crash Utility

---

### Overview

In this practice, you run the following:

- The `crash` utility on a `vmcore`
- Various `crash` commands to analyze the state of the `vmcore`
- The `crash` utility on a running system
- Various `crash` commands to analyze the state of the running system

### Assumptions

- You are logged in as `root` on `host03`.

### Tasks

1. Determine the location of the `vmlinu`x file in the `kernel-uek-debuginfo` RPM.
  - c. Use the `rpm -q1` command to list the files in an RPM package. Pipe the output to the `grep` command to search for `vmlinu`x.
  - This file is provided as a command-line argument to the `crash` command.

```
# rpm -q1 kernel-uek-debuginfo | grep vmlinu
/usr/lib/debug/lib/modules/3.8.13-55.1.6.el7uek.x86_64/vmlinu
```

2. Use the `ls` command to list the `vmcore` file name. Ensure that you are in the `root` user's home directory.
  - The file name for your `vmcore` is different.

```
# cd
# ls *core
2015-0721-2154.29-host03.59.core
```

3. Use the `crash` command to initialize the core dump analysis session.
  - In this example, the core dump file name is `2015-0721-2154.29-host03.59.core`. Substitute the correct `vmcore` file name as determined in the previous task.
  - Sample output is displayed:

```
# crash 2015-0721-2154.29-host03.59.core
/usr/lib/debug/lib/modules/3.8.13-55.1.6.el7uek.x86_64/vmlinu
...
KERNEL: /usr/lib/debug/lib/modules/3.8.13-
55.1.6.el7uek.x86_64/vmlinu
DUMPFILE: 2015-0721-2154.29-host03.59.core
CPUS: 1
DATE: Thu Feb 20 ...
UPTIME: 11:19:18
LOAD AVERAGE: 0.00, 0.01, 0.05
TASKS: 215
NODENAME: host03.example.com
```

```

RELEASE: 3.8.13-55.1.6.el7uek.x86_64
VERSION: #2 SMP Thu Feb 13 ...
MACHINE: x86_64 (2992 Mhz)
MEMORY: 1.5 GB
PANIC: ""
PID: 0
COMMAND: "swapper"
TASK: ffffffff818ae420 [THREAD_INFO: ffffffff8189a000]
CPU: 0
STATE: TASK_RUNNING (ACTIVE)
WARNING: panic task not found
crash>

```

4. From the `crash>` prompt, use the following `crash` commands to analyze the Linux crash data. Use `help <command>` for each `crash` command to display usage, options, and examples.
  - No sample output is provided in this task because output varies.
  - a. `? -` Display the `crash` commands.
  - b. `bt -` Display kernel stack backtrace.
  - c. `bt -l -` Display the source code line numbers.
  - d. `dev -` Display character and block device data.
  - e. `dev -d -` Display disk I/O statistics.
  - f. `kmem -i -` Display general memory usage information.
  - g. `log -` Display the kernel message buffer.
  - This is the same data displayed with the `dmesg` command.
  - h. `mach -` Display machine-specific data.
  - i. `mach -c -` Display the `cpuinfo` structure.
  - j. `mod -` Display the currently installed kernel modules.
  - k. `mount -` Display information about currently mounted file systems.
  - l. `net -` Display network-related data.
  - m. `ps -` Display process status information.
  - n. `ps -u -` Display only user tasks.
  - o. `history -` Display a history of executed commands.
    - There is no help available for the `history` command.
  - p. `quit -` Exit the `crash` utility.
5. Use the `crash` command with no arguments to analyze the state of the running system.
  - Only a partial sample output is shown.
  - Notice that the `DUMPFILE` is different and the `COMMAND` is “`crash`”.

```

# crash
...
crash: trying /proc/kcore as an alternative to /dev/mem

```

```
KERNEL: /usr/lib/debug/lib/modules/3.8.13-
55.1.6.el7uek.x86_64/vmlinu
DUMPFILE: /proc/kcore
CPUS: 1
...
PID: 8250
COMMAND: "crash"
...
```

6. From the `crash>` prompt, use the following `crash` commands to analyze the running Linux system. Use `help <command>` for each `crash` command to display usage, options, and examples.
  - No sample output is provided in this task because the output varies.
  - a. `files` – Display information about open files.
  - b. `foreach files -R /dev/pts/1` – Search for references to `/dev/pts/1`.
    - If none found, replace `/dev/pts/1` with `/dev/pts/0`.
  - c. `fuser /dev/pts/1` – Display tasks using `/dev/pts/1`.
    - If none found, replace `/dev/pts/1` with `/dev/pts/0`.
  - d. `sig` – Display a task's signal-handling data.
  - e. `sys` – Display system information.
    - This is the same information displayed during `crash` initialization.
  - f. `sys -c` – Display the system call table entries.
  - g. `task` – Display the contents of `task_struct`.
    - Each `task_struct` data structure describes a process or task in the system.
  - h. `vm` – Display a task's virtual memory data.
  - i. `quit` – Exit the `crash` utility.

## **Practices for Lesson 19: Dynamic Tracing with DTrace**

**Chapter 19**

## Practices for Lesson 19: Overview

---

### Practices Overview

In these practices, you perform the following:

- Upload DTrace Packages from **dom0** to **host03**.
- Install and enable DTrace.
- Use DTrace from the command line.
- Use DTrace D scripts.

## Practice 19-1: Using sftp to Upload DTrace Packages

### Overview

In this practice, you perform the following:

- Use sftp to upload the DTrace packages from **dom0** to the **host03** VM.
- These packages were downloaded to **dom0** from two different ULN channels.
- The first ULN channel, **ol7\_x86\_64\_UEKR3**, is highlighted as follows:

| Name                                                                                         | Label                        | Description                                                                                               | Packages |
|----------------------------------------------------------------------------------------------|------------------------------|-----------------------------------------------------------------------------------------------------------|----------|
| <a href="#">Oracle Linux 7 Latest Optional Packages (x86_64)</a>                             | ol7_x86_64_optional_latest   | All optional packages released for Oracle Linux 7 (x86_64) including the latest errata packages. (x86_64) | 4202     |
| <a href="#">Oracle Linux 7 Latest (x86_64)</a>                                               | ol7_x86_64_latest            | All packages released for Oracle Linux 7 (x86_64) including the latest errata packages. (x86_64)          | 4393     |
| <a href="#">Oracle Linux 7 Update 1 installation media copy(x86_64)</a>                      | ol7_x86_64_u1_base           | All packages released for Oracle Linux 7 Update 1 (x86_64). No errata included                            | 4384     |
| <a href="#">Oracle Linux 7 Update 1 Patch (x86_64)</a>                                       | ol7_x86_64_u1_patch          | Updated packages published after release of Oracle Linux 7 Update 1 (x86_64)                              | 371      |
| <a href="#">Oracle Linux 7 GA installation media copy (x86_64)</a>                           | ol7_x86_64_u0_base           | All packages released for Oracle Linux 7 GA (x86_64). No errata included                                  | 4315     |
| <a href="#">Oracle Linux 7 GA Patch (x86_64)</a>                                             | ol7_x86_64_u0_patch          | Updated packages published after release of Oracle Linux 7 GA (x86_64)                                    | 583      |
| <a href="#">MySQL 5.6 for Oracle Linux 7 (x86_64)</a>                                        | ol7_x86_64_MySQL56_community | Latest MySQL 5.6 packages for Oracle Linux 7 (x86_64).                                                    | 15       |
| <a href="#">MySQL 5.5 for Oracle Linux 7 (x86_64)</a>                                        | ol7_x86_64_MySQL55_community | Latest MySQL 5.5 packages for Oracle Linux 7 (x86_64).                                                    | 15       |
| <a href="#">Oracle Linux 7 Dtrace Userspace Tools (x86_64) - Latest</a>                      | ol7_x86_64_Dtrace_userspace  | The latest Dtrace userspace tools for Oracle Linux 7 (x86_64).                                            | 2        |
| <a href="#">Oracle Linux 7 Addons (x86_64)</a>                                               | ol7_x86_64_addons            | Oracle Linux 7 Addons (x86_64).                                                                           | 59       |
| <a href="#">Unbreakable Enterprise Kernel Release 3 for Oracle Linux 7 (x86_64) - Latest</a> | ol7_x86_64_UEKR3             | Latest packages for Unbreakable Enterprise Kernel Release 3 for Oracle Linux 7 (x86_64)                   | 35       |
| <a href="#">OFED supporting tool packages for</a>                                            |                              | Latest OpenFabrics Enterprise Distribution (OFED) supporting                                              |          |

- The 35 packages from the ULN channel o17\_x86\_64\_UEKR3 are shown as follows:

| Unbreakable Linux Network (ULN): Channel Packages - Mozilla Firefox                                                                                                                                                                 |                                                                           |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------|
| File                                                                                                                                                                                                                                | Edit                                                                      |
| Unbreakable Linux Network (ULN): Chan...                                                                                                                                                                                            | +                                                                         |
| <a href="https://linux.oracle.com/pls/apex/f?p=101:5:1476790072934::NO:RP:P5_CHANNEL_ID,P4_CHANNEL_ID,CURRENT_CHANI">https://linux.oracle.com/pls/apex/f?p=101:5:1476790072934::NO:RP:P5_CHANNEL_ID,P4_CHANNEL_ID,CURRENT_CHANI</a> | Google                                                                    |
| dtrace-modules-3.8.13-44.1.4.el7uek-0.4.3-4.el7.x86_64                                                                                                                                                                              | dtrace module                                                             |
| dtrace-modules-3.8.13-44.1.5.el7uek-0.4.3-4.el7.x86_64                                                                                                                                                                              | dtrace module                                                             |
| dtrace-modules-3.8.13-44.el7uek-0.4.3-4.el7.x86_64                                                                                                                                                                                  | dtrace module                                                             |
| dtrace-modules-3.8.13-55.1.1.el7uek-0.4.3-4.el7.x86_64                                                                                                                                                                              | dtrace module                                                             |
| dtrace-modules-3.8.13-55.1.2.el7uek-0.4.3-4.el7.x86_64                                                                                                                                                                              | dtrace module                                                             |
| dtrace-modules-3.8.13-55.1.5.el7uek-0.4.3-4.el7.x86_64                                                                                                                                                                              | dtrace module                                                             |
| dtrace-modules-3.8.13-55.1.6.el7uek-0.4.3-4.el7.x86_64                                                                                                                                                                              | -                                                                         |
| dtrace-modules-3.8.13-55.1.8.el7uek-0.4.3-4.el7.x86_64                                                                                                                                                                              | dtrace module                                                             |
| dtrace-modules-3.8.13-55.2.1.el7uek-0.4.3-4.el7.x86_64                                                                                                                                                                              | dtrace module                                                             |
| dtrace-modules-3.8.13-55.el7uek-0.4.3-4.el7.x86_64                                                                                                                                                                                  | dtrace module                                                             |
| dtrace-modules-3.8.13-68.1.2.el7uek-0.4.3-4.el7.x86_64                                                                                                                                                                              | dtrace module                                                             |
| dtrace-modules-3.8.13-68.1.3.el7uek-0.4.3-4.el7.x86_64                                                                                                                                                                              | dtrace module                                                             |
| dtrace-modules-3.8.13-68.2.2.el7uek-0.4.3-4.el7.x86_64                                                                                                                                                                              | dtrace module                                                             |
| dtrace-modules-3.8.13-68.3.1.el7uek-0.4.3-4.el7.x86_64                                                                                                                                                                              | dtrace module                                                             |
| dtrace-modules-3.8.13-68.3.2.el7uek-0.4.3-4.el7.x86_64                                                                                                                                                                              | dtrace module                                                             |
| dtrace-modules-3.8.13-68.3.3.el7uek-0.4.3-4.el7.x86_64                                                                                                                                                                              | dtrace module                                                             |
| dtrace-modules-3.8.13-68.3.4.el7uek-0.4.3-4.el7.x86_64                                                                                                                                                                              | dtrace module                                                             |
| dtrace-modules-3.8.13-68.el7uek-0.4.3-4.el7.x86_64                                                                                                                                                                                  | dtrace module                                                             |
| dtrace-modules-headers-0.4.3-4.el7.x86_64                                                                                                                                                                                           | -                                                                         |
| dtrace-modules-provider-headers-0.4.3-4.el7.x86_64                                                                                                                                                                                  | -                                                                         |
| kernel-uek-3.8.13-68.3.4.el7uek.x86_64                                                                                                                                                                                              | The Linux kernel                                                          |
| kernel-uek-debug-3.8.13-68.3.4.el7uek.x86_64                                                                                                                                                                                        | The Linux kernel compiled with extra debugging enabled                    |
| kernel-uek-debug-devel-3.8.13-68.3.4.el7uek.x86_64                                                                                                                                                                                  | Development package for building kernel modules to match the debug kernel |
| kernel-uek-devel-3.8.13-68.3.4.el7uek.x86_64                                                                                                                                                                                        | Development package for building kernel modules to match the kernel       |
| kernel-uek-doc-3.8.13-68.3.4.el7uek.noarch                                                                                                                                                                                          | Various documentation bits found in the kernel source                     |
| kernel-uek-firmware-3.8.13-68.3.4.el7uek.noarch                                                                                                                                                                                     | Firmware files used by the Linux kernel                                   |
| libdtrace-ctf-0.4.1-1.el7.x86_64                                                                                                                                                                                                    | -                                                                         |

- The first highlighted package from this channel, dtrace-modules-3.8.13-55.1.6.el7uek-0.4.3-4.el7.x86\_64.rpm, has been downloaded to **dom0**.
- The second highlighted package from this channel, dtrace-modules-headers-0.4.3-4.el7.x86\_64.rpm, is a required dependency package, which has also been downloaded to **dom0**.

- The second ULN channel, `ol7_x86_64_Dtrace_userspace`, is highlighted as follows:

| Name                                                                                         | Label                        | Description                                                                                               | Packages |
|----------------------------------------------------------------------------------------------|------------------------------|-----------------------------------------------------------------------------------------------------------|----------|
| <a href="#">Oracle Linux 7 Latest Optional Packages (x86_64)</a>                             | ol7_x86_64_optional_latest   | All optional packages released for Oracle Linux 7 (x86_64) including the latest errata packages. (x86_64) | 4202     |
| <a href="#">Oracle Linux 7 Latest (x86_64)</a>                                               | ol7_x86_64_latest            | All packages released for Oracle Linux 7 (x86_64) including the latest errata packages. (x86_64)          | 4393     |
| <a href="#">Oracle Linux 7 Update 1 installation media copy (x86_64)</a>                     | ol7_x86_64_u1_base           | All packages released for Oracle Linux 7 Update 1 (x86_64). No errata included                            | 4384     |
| <a href="#">Oracle Linux 7 Update 1 Patch (x86_64)</a>                                       | ol7_x86_64_u1_patch          | Updated packages published after release of Oracle Linux 7 Update 1 (x86_64)                              | 371      |
| <a href="#">Oracle Linux 7 GA installation media copy (x86_64)</a>                           | ol7_x86_64_u0_base           | All packages released for Oracle Linux 7 GA (x86_64). No errata included                                  | 4315     |
| <a href="#">Oracle Linux 7 GA Patch (x86_64)</a>                                             | ol7_x86_64_u0_patch          | Updated packages published after release of Oracle Linux 7 GA (x86_64)                                    | 583      |
| <a href="#">MySQL 5.6 for Oracle Linux 7 (x86_64)</a>                                        | ol7_x86_64_MySQL56_community | Latest MySQL 5.6 packages for Oracle Linux 7 (x86_64).                                                    | 15       |
| <a href="#">MySQL 5.5 for Oracle Linux 7 (x86_64)</a>                                        | ol7_x86_64_MySQL55_community | Latest MySQL 5.5 packages for Oracle Linux 7 (x86_64).                                                    | 15       |
| <a href="#">Oracle Linux 7 Dtrace Userspace Tools (x86_64) - Latest</a>                      | ol7_x86_64_Dtrace_userspace  | The latest Dtrace userspace tools for Oracle Linux 7 (x86_64).                                            | 2        |
| <a href="#">Oracle Linux 7 Addons (x86_64)</a>                                               | ol7_x86_64_addons            | Oracle Linux 7 Addons (x86_64).                                                                           | 59       |
| <a href="#">Unbreakable Enterprise Kernel Release 3 for Oracle Linux 7 (x86_64) - Latest</a> | ol7_x86_64_UEKR3             | Latest packages for Unbreakable Enterprise Kernel Release 3 for Oracle Linux 7 (x86_64)                   | 35       |
| <a href="#">OFED supporting tool packages for</a>                                            |                              | Latest OpenFabrics Enterprise Distribution (OFED) supporting                                              |          |

- The two packages from the ULN channel `o17_x86_64_Dtrace_userspace` are shown as follows:

| Package                                               | Description                 |
|-------------------------------------------------------|-----------------------------|
| <a href="#">dtrace-utils-0.4.5-2.el7.x86_64</a>       | DTrace user interface.      |
| <a href="#">dtrace-utils-devel-0.4.5-2.el7.x86_64</a> | DTrace development headers. |

- The highlighted package from this channel, `dtrace-utils-0.4.5-2.el7.x86_64.rpm`, has been downloaded to **dom0**.

## Assumptions

- You are logged in as `root` on **dom0**.
- You are logged in as `root` on **host03**.
- The DTrace packages from the ULN have been downloaded to the `root` user's home directory on **dom0**.

## Tasks

- Use the `sftp` command to transfer the DTrace packages from **dom0** to **host03**.
  - From **dom0**, use the `cd` command to change to the `/OVS/seed_pool/dtrace_rpms` directory.

```
[dom0] # cd /OVS/seed_pool/dtrace_rpms
```

- b. Use the `ls` command to view the contents of the directory.

```
[dom0]# ls
dtrace-modules-3.8.13-55.1.6.el7uek-0.4.3-4.el7.x86_64.rpm
dtrace-modules-headers-0.4.3-4.el7.x86_64.rpm
dtrace-utils-0.4.5-2.el7.x86_64.rpm
```

- c. Use the `sftp` command to connect to **host03** as `root`.

- The password is `oracle`.

```
[dom0]# sftp host03
Connecting to host03...
root@host03's password: oracle
sftp>
```

- d. Use the `mput *` command to upload all files on **dom0** to **host03**.

```
sftp> mput *
Uploading dtrace-modules-3.8.13-55.1.6.el7uek-0.4.3-
4.el7.x86_64.rpm to /root/...
Uploading dtrace-modules-headers-0.4.3-4.el7.x86_64.rpm to
/root/...
Uploading dtrace-utils-0.4.5-2.el7.x86_64.rpm to /root/...
...
sftp>
```

- e. Use the `quit` command to exit `sftp`.

```
sftp> quit
```

## Practice 19-2: Installing the DTrace Packages

---

### Overview

In this practice, you perform the following:

- Install the DTrace packages.
- Load the DTrace modules into the kernel.
- View the DTrace release notes.

### Assumptions

- You are the `root` user on **host03**.

### Tasks

1. Install the DTrace packages and dependency packages.
  - a. As the `root` user on **host03**, use the `ls` command to view the contents of the `root` user's home directory.
    - Use the `cd` command to ensure that you are in the root user's home directory.
    - The directory contains the DTrace packages along with some files that were created during the OS installation and other files that were used in previous practices.

```
# cd
# ls
...
dtrace-modules-3.8.13-55.1.6.el7uek-0.4.3-4.el7.x86_64.rpm
dtrace-modules-headers-0.4.3-4.el7.x86_64.rpm
dtrace-utils-0.4.5-2.el7.x86_64.rpm
...
```

- b. Use the `rpm -Uvh` command to install each of the DTrace packages.
  - If your system was registered with the following ULN channels, you could use the `yum` command to update to the UEK R3 and to install the DTrace packages.
    - `ol7_x86_64_latest`
    - `ol7_x86_64_UEKR3`
    - `ol7_x86_64_Dtrace_userspace`

```
# rpm -Uvh dtrace-modules-headers-0.4.3-4.el7.x86_64.rpm
...
# rpm -Uvh dtrace-utils-0.4.5-2.el7.x86_64.rpm
...
# rpm -Uvh dtrace-modules-3.8.13-55.1.6.el7uek-0.4.3-
4.el7.x86_64.rpm
...
```

2. Load the DTrace modules into the running kernel.
  - a. Use the `cd` command to change to the `/lib/modules/3.8.13-55.1.6.el7uek.x86_64/kernel/drivers/dtrace` directory.
 

```
# cd /lib/modules/3.8.13-55.1.6.el7uek.x86_64/kernel/drivers/dtrace
```
  - b. Use the `ls -l` command to view the contents of the directory.
 

```
# ls -l
-rw-r--r--. ... dt_perf.ko
-rw-r--r--. ... dtrace.ko
-rw-r--r--. ... dt_test.ko
-rw-r--r--. ... fasttrap.ko
-rw-r--r--. ... profile.ko
-rw-r--r--. ... sdt.ko
-rw-r--r--. ... systrace.ko
```
  - c. Use the `modprobe` command to load the necessary modules.
    - Do not load the `dt_*` modules; they are for development purposes only.
    - It is not necessary to load the `dtrace` module. Loading any of the other modules automatically loads the `dtrace` module.
    - The following set of commands shows that the `dtrace` module is loaded by loading the `fasttrap` module.
    - The last `lsmod` command shows that the `dtrace` module is used by the `systrace`, `sdt`, `profile`, and `fasttrap` modules.

```
# modprobe fasttrap
# lsmod | grep dtrace
dtrace    135025  1  fasttrap
ctf        941   1  dtrace
# modprobe profile
# modprobe sdt
# modprobe systrace
# lsmod | grep systrace
systrace  4166   0
dtrace    135025  4  sdt,fasttrap,systrace,profile
```

3. View the DTrace release notes.
  - a. Use the `cd` command to change to the `/usr/share/doc/dtrace-0.4.5` directory.
 

```
# cd /usr/share/doc/dtrace-0.4.5
```
  - b. Use the `ls` command to display the contents of the directory.
 

```
# ls
INCOMPATIBILITIES  NEWS  README  showUSDT
```

- c. Use the `less` command to view the `INCOMPATIBILITIES` file in the `/usr/share/doc/dtrace-0.4.5` directory.

```
# less INCOMPATIBILITIES
```

This file documents known incompatibilities between Linux and Solaris dtrace, together with the difficulty of overcoming them, and the likelihood that they will be overcome.

Missing providers

-----

Difficulty: Medium

Likelihood: High

A number of providers are missing, including pid, fbt, and net.

...

- d. Use the `less` command to view the `NEWS` file in the `/usr/share/doc/dtrace-0.4.5` directory.

- Note that the DTrace modules are now automatically loaded.

```
# less NEWS
```

0.4.5

-----

New features:

- Provider modules are now automatically loaded from `/etc/dtrace-modules` when DTrace initializes for the first time, at the same time as `dtrace.ko`. (Providers that do not come from the `dtrace-modules` package are not automatically 'yum install'ed.)

Bugfixes:

- Fix intermittent dtrace crash on failure of initial grabs or creations of processes (via `dtrace -p` or `-c`, or via `ustack()`, `usym()`, `uaddr()`, or `umod()`).

...

- e. Use the `less` command to view the `README` file in the `/usr/share/doc/dtrace-0.4.5` directory.

```
# less README
```

Linux DTrace v0.4 for x86\_64 Oracle Unbreakable Enterprise Kernel 3.8

...

- f. Use the `cd` command to change to the `/usr/share/doc/dtrace-modules-3.8.13-55.1.6.el7uek` directory.

```
# cd /usr/share/doc/dtrace-modules-3.8.13-55.1.6.el7uek
```

- g. Use the `ls` command to display the contents of the directory.

```
# ls  
NEWS
```

- h. Use the `less` command to view the `NEWS` file.

- Notice that this file contains new features, changes, and bug fixes for each release.

```
# less NEWS  
DTrace Kernel Modules News  
=====  
  
Release 0.4.3 (May 1st, 2014)  
-----  
Kernel release: 3.8.13-33.el6uek  
  
New features:  
  
- Timer based profile-* probes (profile provider). These probes use the omni-present cyclic support in the UEK3 kernel (3.8.13-32 and later) to fire probes at a specific frequency/interval on every active CPU.  
  
Changes:  
  
- The pid and ppid variables were being reported based on ...  
...
```

## Practice 19-3: Using DTrace from the Command Line

---

### Overview

In this practice, you perform the following:

- List DTrace probes for selected providers.
- Enable DTrace probes.
- Enable probes and specify actions.
- Execute specific `dtrace` commands and observe the output.

### Assumptions

- You are the root on **host03**.

### Tasks

1. List DTrace probes.
  - a. Use the `dtrace -l` command to list DTrace probes for all providers.

```
# dtrace -l
ID PROVIDER MODULE FUNCTION NAME
1 dtrace
2 dtrace
3 dtrace
4 syscall vmlinux read entry
5 syscall vmlinux read return
...
646 profile tick-1000
647 profile tick-5000
```

- b. Use the `dtrace -l -P dtrace` command to list the probes for the `dtrace` provider.

```
# dtrace -l -P dtrace
ID PROVIDER MODULE FUNCTION NAME
1 dtrace
2 dtrace
3 dtrace
```

- c. Use the `dtrace -l -P syscall` command to list the probes for the `syscall` provider.

```
# dtrace -l -P syscall
ID PROVIDER MODULE FUNCTION NAME
4 syscall vmlinux read entry
5 syscall vmlinux read return
6 syscall vmlinux write entry
7 syscall vmlinux write return
8 syscall vmlinux open entry
...
602 syscall vmlinux waitfd entry
```

```
603 syscall    vmlinux      waitfd return
```

- d. Use the `dtrace -l -P sched` command to list the probes for the `sched` provider.

```
# dtrace -l -P sched
ID PROVIDER MODULE FUNCTION NAME
604 sched vmlinux __schedule remain-cpu
605 sched vmlinux __schedule sleep
606 sched vmlinux __schedule preempt
607 sched vmlinux __schedule off-cpu
612 sched vmlinux dequeue_task dequeue
623 sched vmlinux enqueue_task enqueue
624 sched vmlinux finish_task_switch on-cpu
629 sched vmlinux set_user_nice change-pri
631 sched vmlinux sys_sched_yield surrender
632 sched vmlinux try_to_wake_up wakeup
633 sched vmlinux update_process_times tick
634 sched vmlinux yield_to surrender
```

- e. Use the `dtrace -l -P sdt` command to list the probes for the `sdt` provider.
- The SDT probes that are defined for Oracle Linux kernel are likely to change over time.

```
# dtrace -l -P sdt
ID PROVIDER MODULE FUNCTION NAME
622 sdt vmlinux dtrace_sdt_perf measure
627 sdt vmlinux psinfo_cleaner test
```

- f. Use the `dtrace -l -P io` command to list the probes for the `io` provider.

```
# dtrace -l -P io
ID PROVIDER MODULE FUNCTION NAME
610 io vmlinux __wait_on_buffer wait-done
611 io vmlinux __wait_on_buffer wait-start
612 io vmlinux submit_bh start
623 io vmlinux end_bio_bh_io_sync done
```

- g. Use the `dtrace -l -P proc` command to list the probes for the `proc` provider.

```
# dtrace -l -P proc
ID PROVIDER MODULE FUNCTION NAME
608 proc vmlinux __send_signal signal-discard
609 proc vmlinux __send_signal signal-send
614 proc vmlinux do_execve_common.isra.24 exec-success
615 proc vmlinux do_execve_common.isra.24 exec
616 proc vmlinux do_execve_common.isra.24 exec-failure
617 proc vmlinux do_exit exit
618 proc vmlinux do_exit lwp-exit
619 proc vmlinux do_fork create
```

```

620      proc  vmlinux           do_fork lwp-create
621      proc  vmlinux           do_sigtimedwait signal-clear
626      proc  vmlinux get_signal_to_deliver signal-handle
628      proc  vmlinux           schedule_tail lwp-start
629      proc  vmlinux           schedule_tail start

```

- h. Use the `dtrace -l -P profile` command to list the probes for the `profile` provider.

| # dtrace -l -P profile | ID  | PROVIDER | MODULE | FUNCTION NAME |
|------------------------|-----|----------|--------|---------------|
|                        | 635 | profile  |        | profile-97    |
|                        | 636 | profile  |        | profile-199   |
|                        | 637 | profile  |        | profile-499   |
|                        | 638 | profile  |        | profile-997   |
|                        | 639 | profile  |        | profile-1999  |
|                        | 640 | profile  |        | profile-4001  |
|                        | 641 | profile  |        | profile-4999  |
|                        | 642 | profile  |        | tick-1        |
|                        | 643 | profile  |        | tick-10       |
|                        | 644 | profile  |        | tick-100      |
|                        | 645 | profile  |        | tick-500      |
|                        | 646 | profile  |        | tick-1000     |
|                        | 647 | profile  |        | tick-5000     |

2. Open a second window from **dom0** and log in to **host03** as the `root` user.

From the second window, use the `su -` command to become the `root` user on **dom0** and then `ssh` to **host03**.

- The `root` password is `oracle` on **dom0** and on **host03**.

```

[dom0]$ su -
Password: oracle
# ssh host03
root@host03's password: oracle

```

3. Enable DTrace probes.

- Enable probes by specifying them without the `-l` (list) option.
- Use the `dtrace -P dtrace` command to enable all probes for the `dtrace` provider.
  - Notice that the `dtrace:::BEGIN` probe is enabled and displayed.
  - Press `Ctrl + C` to return to the prompt.
  - The `dtrace:::END` probe was not displayed until after you pressed `Ctrl + C`.

- No action was specified; therefore, the default action was taken, which only indicates that the probe fired.

```
# dtrace -P dtrace
dtrace: description 'dtrace' matched 3 probes
CPU      ID      FUNCTION:NAME
0          1          :BEGIN
^C
0          2          :END
```

- Use the `dtrace -n dtrace:::BEGIN` command to enable only the `dtrace:::BEGIN` probe for the `dtrace` provider.
  - The `-n` option specifies to enable the probe by name.
  - Press Ctrl + C to return to the prompt.
  - Notice that only one probe matched—the probe that was specified by name.

```
# dtrace -n dtrace:::BEGIN
dtrace: description 'dtrace:::BEGIN' matched 1 probe
CPU      ID      FUNCTION:NAME
0          1          :BEGIN
^C
```

- Use the `dtrace -n syscall::open:` command to enable all `open` function probes for the `syscall` provider.
  - In the second terminal window on `host03`, enter the `ls` command. This command causes probes to fire as output from the `dtrace` command.
  - After viewing the output of the `dtrace` command, press Ctrl + C to return to the prompt.
  - Notice that two probes matched—the entry to and return from the `open` system call.

```
# dtrace -n syscall::open:
dtrace: description 'syscall::open:' matched 2 probes
CPU      ID      FUNCTION:NAME
0          8          open:entry
0          9          open:return
0          8          open:entry
0          9          open:return
...
^C
```

- Enable probes and specify actions.
  - Use the `dtrace -n 'dtrace:::BEGIN { trace("hello, world"); }'` command to enable the `dtrace:::BEGIN` probe with the action, `trace("hello, world")`.
    - Notice that the action is surrounded by `{ }`.
    - Notice that the entire string, beginning with the probe to be enabled and the associated action to take, is surrounded by `' '`.

- Notice the “hello, world” output, which is the result of the specified `trace()` action.
- Press Ctrl + C to return to the prompt.

```
# dtrace -n 'dtrace:::BEGIN {trace("hello, world");}'
dtrace: description 'dtrace:::BEGIN' matched 1 probe
CPU          ID          FUNCTION:NAME
0            1           :BEGIN  hello, world
^C
```

- b. Add a second action, `exit(0)`, to the previous command.

- Notice that you did not need to press Ctrl + C to return to the prompt.
- The action `exit(0)` caused the command to exit.
- Notice that each action is terminated with a semicolon (;) character.

```
# dtrace -n 'dtrace:::BEGIN {trace("hello, world");exit(0);}'
dtrace: description 'dtrace:::BEGIN' matched 1 probe
CPU          ID          FUNCTION:NAME
0            1           :BEGIN  hello, world
```

- c. Add the `-q` (quiet) option to the previous command.

- Notice that the quiet option suppresses the default output of displaying the CPU where the probe fired, the unique probe ID, the function where the probe fired (if any), and the probe name.

```
# dtrace -q -n 'dtrace:::BEGIN {trace("hello, world");exit(0);}'
hello, world
```

## 5. Execute specific `dtrace` commands.

- No sample output is provided for the following `dtrace` commands.
- Enter each of the examples from the command line and observe the output.

### a. Trace the time of entry for each system call.

- Press Ctrl + C to exit the `dtrace` command.

```
# dtrace -n 'syscall:::entry {trace(timestamp)}'
...
^C
```

### b. Display commands that are executing on your system.

- Do not immediately press Ctrl + C to exit the `dtrace` command.
- You run a command in the second window to generate a `dtrace` output.

```
# dtrace -n 'proc:::exec {trace(stringof(arg0));}'
...
```

- From the second window on **host03**, run the `man ls` command and observe the output from the `dtrace` command in the first window.

```
# man ls
...
```

- After viewing the output from the `dtrace` command in the first window, press Ctrl + C to kill the `dtrace` command.

```
# dtrace -n 'proc:::exec {trace(stringof(arg0));}'
...
^C
```

- Enter q to terminate the `man ls` command in the second window.

```
# man ls
...
q
```

- Display new processes with time stamps and arguments.

- Do not immediately press Ctrl + C to exit the `dtrace` command.
- You run a command in the second window to generate a `dtrace` output.

```
# dtrace -qn 'proc:::exec-success {printf("%d %s\n", timestamp,
curpsinfo->pr_psargs);}'
...
```

- From the second window on **host03**, run the `man ls` command and observe the output from the `dtrace` command in the first window.

```
# man ls
...
```

- After viewing the output from the `dtrace` command in the first window, press Ctrl + C to kill the `dtrace` command.

```
# dtrace -qn 'proc:::exec-success {printf("%d %s\n", timestamp,
curpsinfo->pr_psargs);}'
...
^C
```

- Enter q to terminate the `man ls` command in the second window.

```
# man ls
...
q
```

- Display files opened by the process.

- Do not immediately press Ctrl + C to exit the `dtrace` command.
- You run a command in the second window to generate a `dtrace` output.

```
# dtrace -qn 'syscall::open*:entry {printf("%s %s\n",
execname,copyinstr(arg0));}
...
```

- From the second window on **host03**, run the `man ls` command and observe the output from the `dtrace` command in the first window.

```
# man ls
...
```

- After viewing the output from the `dtrace` command in the first window, press Ctrl + C to kill the `dtrace` command.

```
# dtrace -qn 'syscall::open*:entry {printf("%s %s\n",
execname,copyinstr(arg0));}'
...
^C
```

- Enter q to terminate the `man ls` command in the second window.

```
# man ls
...
q
```

- Display the number of system calls by system call using aggregations.

- When using aggregations (@), no output is displayed until after you press Ctrl + C.
- Do not immediately press Ctrl + C to exit the `dtrace` command.
- You run a command in the second window to generate a `dtrace` output.

```
# dtrace -qn 'syscall:::entry {@num[probefunc] = count();}'
```

- From the second window on **host03**, run the `man ls` command and observe the output from the `dtrace` command in the first window.

```
# man ls
...
```

- Press Ctrl + C to kill the `dtrace` command and display the aggregated totals.

```
# dtrace -qn 'syscall:::entry {@num[probefunc] = count();}'
^C
...
```

- Enter q to terminate the `man ls` command in the second window.

```
# man ls
...
q
```

- Count the number of `write()` system calls and the number of `read()` system calls invoked by processes until you press Ctrl + C.

- Do not immediately press Ctrl + C to exit the `dtrace` command.
- You run a command in the second window to generate a `dtrace` output.

```
# dtrace -qn 'syscall::write:entry,syscall::read:entry
{@[strjoin(probefunc,"() calls")] = count();}'
```

- From the second window on **host03**, run the `man ls` command and observe the output from the `dtrace` command in the first window.

```
# man ls
...
```

- Press Ctrl + C to kill the dtrace command and display the aggregated totals.

```
# dtrace -qn 'syscall::write:entry,syscall::read:entry
{@[strjoin(probefunc,"() calls")] = count();}'
^C
...

```

- Enter q to terminate the man ls command in the second window.

```
# man ls
...
q
```

- g. Aggregate the number of `read()` system calls but use a predicate to exclude the `read()` system calls initiated by the `dtrace` process.

- Do not immediately press Ctrl + C to exit the `dtrace` command.
- You run a command in the second window to generate a `dtrace` output.
- The predicate in this example is `/execname != "dtrace"/`.
- Predicates are always enclosed in `//`.

```
# dtrace -qn 'syscall::read:entry /execname != "dtrace"/ {
@reads [execname, fds[arg0].fi.pathname] = count();}'
```

- From the second window on **host03**, run the `man ls` command and observe the output from the `dtrace` command in the first window.

```
# man ls
...
```

- Press Ctrl + C to kill the `dtrace` command and display the aggregated totals.

```
# dtrace -qn 'syscall::read:entry /execname != "dtrace"/ {
@reads [execname, fds[arg0].fi.pathname] = count();}'
^C
...
```

- Enter q to terminate the `man ls` command in the second window.

```
# man ls
...
q
```

## Practice 19-4: Creating and Running D Scripts

---

### Overview

In this practice, you create and run D scripts.

### Assumptions

- You are the root on **host03**.

### Tasks

1. Create and run a D script.

a. Use the `vi` editor to create the following `rdbufsize.d` file.

- This shows quantized results for the buffer sizes used in `read()` syscalls.
- It gives a statistical breakdown of the sizes passed in the `read()` syscall.
- This can be useful to see what buffer sizes are commonly used.

```
# vi rdbufsize.d
syscall::read:entry
{
    @["read"] = quantize(arg2);
}
```

b. Use the `dtrace -s` command to execute the `rdbufsize.d` D script.

```
# dtrace -s rdbufsize.d
^C
...
```

2. Create and run a D script.

a. Use the `vi` editor to create the following `syscall.d` file:

- This script counts system calls over a 10-second period.

```
# vi syscall.d
syscall:::
{
    @[probefunc] = count();
}
tick-1s
/i++ >= 10/
{
    exit(0);
}
```

b. Use the `dtrace -s` command to execute the `syscall.d` D script.

- This script takes 10 seconds to complete. Do not press Ctrl + C.

```
# dtrace -s syscall.d
...
```

3. Create and run a D script.

a. Use the vi editor to create the following `rtime.d` file.

- This script records the times that processes spend invoking `read()` calls.

```
# vi rtime.d
syscall::read:entry
{
    self->t = timestamp;
}
syscall::read:return
/self->t != 0/
{
    printf("%s spent %d nsecs in read()\n", execname, timestamp -
self->t);
    self->t = 0;
}
```

b. Use the `dtrace -s` command to execute the `rtime.d` D script.

- Press Ctrl + C to exit the script after a few lines of output.

```
# dtrace -s rtime.d
...
^C
```

4. Create and run a D script.

a. Use the vi editor to create the following `io1.d` file.

- This script displays the details of block I/O events.

```
# vi io1.d
io:::done,
io:::start,
io:::wait-done,
io:::wait-start
{
    printf("%8s %10s: %d %16s (%s size %d @ sect %d)\n",
           args[1] ->dev_statname,
           probename,
           timestamp & 1000000000,
           execname,
           args[0] ->b_flags & B_READ ? "R" :
           args[0] ->b_flags & B_WRITE ? "W" : "?",
           args[0] ->b_bcount,
           args[0] ->b_blkno);
}
```

- b. Use the `dtrace -s` command to execute the `iol.d` D script.
- Press Ctrl + C to exit the script.

```
# dtrace -s iol.d
...
^C
```

5. Log off from **host03**.
- Use the `exit` command to log off from **host03**.

```
# exit
logout
Connection to host03 closed.
```

- Repeat step 5a to log off from **host03** from the second terminal window.

## **Appendix - NIS Configuration**

### **Chapter 20**

## Appendix - Overview

---

### Appendix Overview

In these practices, you configure:

- An NIS server
- The NIS client and implement NIS authentication
- Auto-mounting of a user home directory

## Practice A-1: Configuring an NIS Server

---

### Overview

In this practice, you:

- Configure an NIS server in preparation to implement NIS authentication
- Install the NIS service package (`ypserv`), start the service, and configure the service to automatically start at boot time
- Explore the various NIS files and directories, and create NIS maps

### Assumptions

You are the `root` user on `dom0`.

### Tasks

- Log in to **host03** by using `vncviewer`.

- From **dom0**, determine the VNC port number for **host03** by running the `xm list -l host03 | grep location` command.

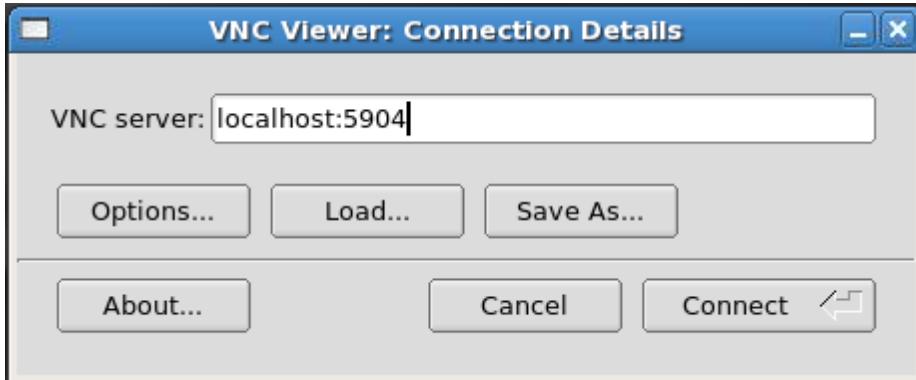
```
[dom0]# xm list -l host03 | grep location
                                (location 0.0.0.0:5904)
                                (location 3)
```

- The sample shown indicates that the port number is 5904. This might not be true in your case.

- From **dom0**, run the `vncviewer&` command.

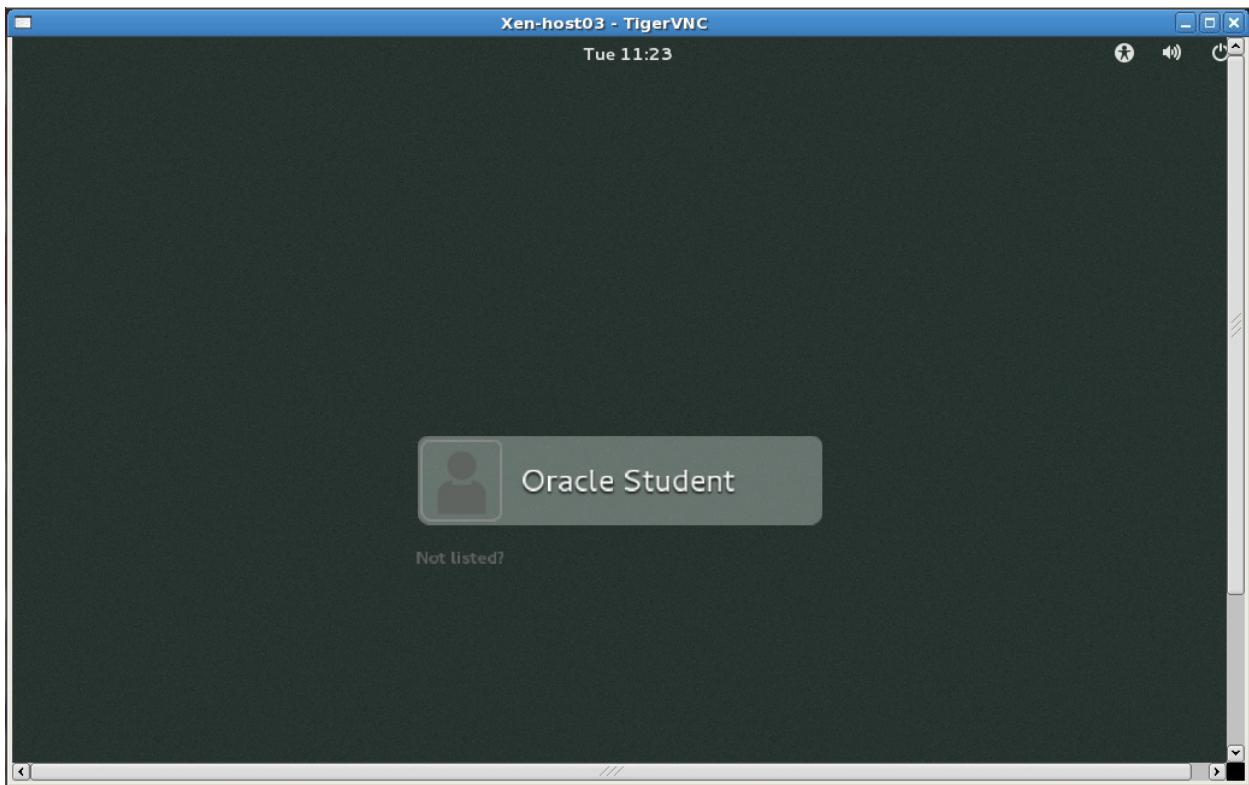
```
[dom0]# vncviewer&
```

- The “VNC Viewer: Connection Details” dialog box appears as shown:



- Enter `localhost:<port_number>`, substituting the port number displayed from the previous `xm list -l host03 | grep location` command.
- For example, if the port number is 5904, enter `localhost:5904` and click “Connect.”

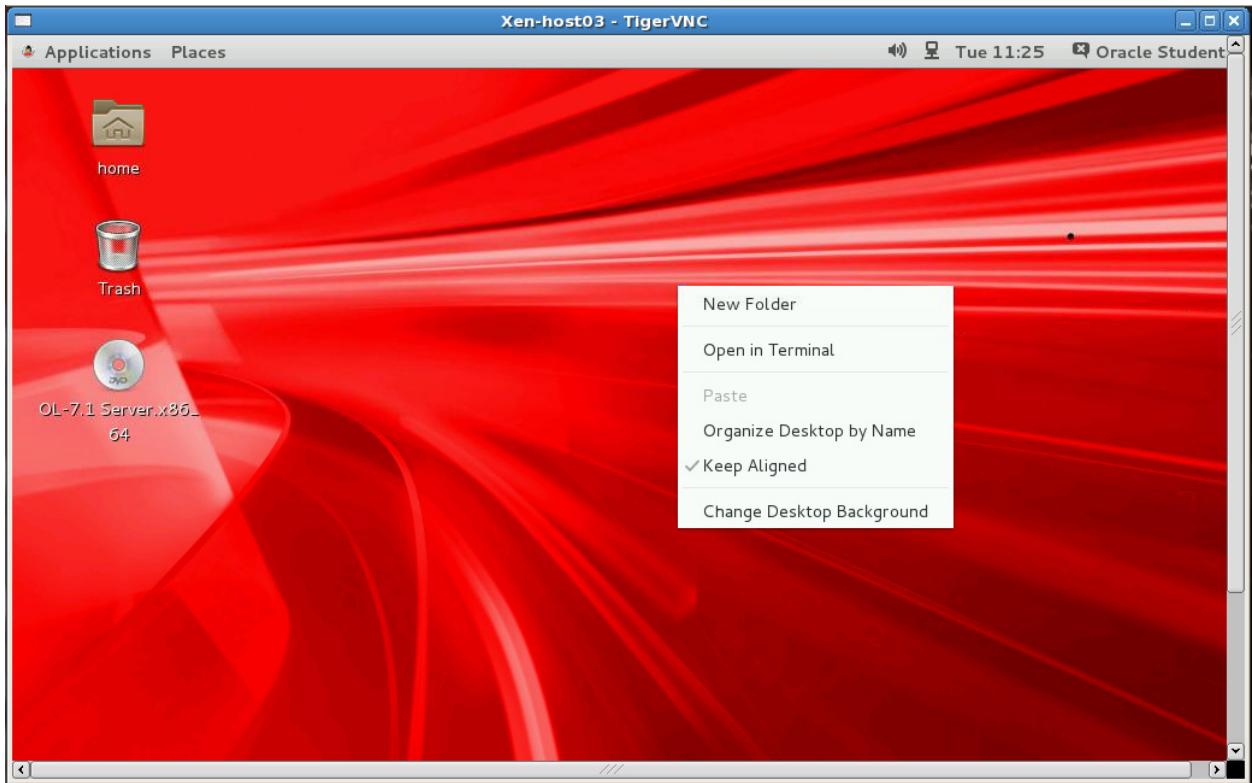
- The GNOME login screen appears:



- d. Click "Oracle Student" in the list of users. You are prompted for the password.
- e. Enter `oracle` for the Password and click "Sign In."

- The GNOME desktop appears.

- f. Right-click the desktop to display the pop-up menu:



- g. From the pop-up menu, select “Open in Terminal.”
- A terminal window appears.
- h. In the terminal window, use the `su -` command to become the `root` user.
- The `root` password is `oracle`.

```
$ su -
Password: oracle
#
```

2. Configure **host03** as an NIS server.

- a. Use the `yum` command to install the `ypserv` package on **host03**.
- Answer `y` when prompted “Is this ok”.

```
# yum install ypserv
...
Transaction Summary
=====
Install 1 Package (+1 Dependent package)

Total download size: 606 k
Installed size: 1.6 M
```

```
Is this ok [y/d/N] : y
...
Complete!
```

- b. Use the vi editor to edit the /etc/sysconfig/network file and add the following line:

```
# vi /etc/sysconfig/network
NISDOMAIN=nis.example.com
```

- c. Use the vi editor to add the following access rule to the end of the /etc/ypserv.conf file.
- This rule allows access to the NIS server from any system on the 192.0.2 subnet.

```
# vi /etc/ypserv.conf
192.0.2.1/24 : * : * : none
```

- d. Use the vi editor to create the /var/yp/securenets file with the following content.
- These entries enhance system security by allowing access only from **localhost** (127.0.0.1) and from systems with IP addresses starting with 192.0.2.

```
# vi /var/yp/securenets
255.255.255.255      127.0.0.1
255.255.255.0        192.0.2.0
```

- e. View the /var/yp/Makefile file (using cat, less, or vi) and locate the following block of lines.
- The “all:” target specifies which NIS maps to create.
  - Do not make any changes to this file at this time.

```
# cat /var/yp/Makefile
...
all: passwd group hosts rpc services netid protocols mail \
# netgrp shadow publickey networks ethers bootparams printcap \
# amd.home auto.master auto.home auto.local passwd.adjunct \
# timezone locale netmasks
```

- f. Use the systemctl command to enable the following NIS services to start at boot time.
- ypserv
  - ypxfrd
  - yppasswdd

```
# systemctl enable ypserv
ln -s '/usr/lib/systemd/system/ypserv.service'
'/etc/systemd/system/multi-user.target.wants/ypserv.service'
# systemctl enable ypxfrd
ln -s '/usr/lib/systemd/system/ypxfrd.service'
'/etc/systemd/system/multi-user.target.wants/ypxfrd.service'
# systemctl enable yppasswdd
```

```
ln -s '/usr/lib/systemd/system/yppasswdd.service'
'/etc/systemd/system/multi-user.target.wants/yppasswdd.service'
```

- g. Use the `systemctl` command to start the three services.

```
# systemctl start ypserv
# systemctl start ypxfrd
# systemctl start yppasswdd
```

- h. Use the `cd` command to change to the `/var/yp` directory. Use the `ls` command to view the contents.

- Notice that there are only two entries in this directory at this time.

```
# cd /var/yp
# ls
Makefile  securenets
```

- i. The `ypinit` command is not in your search path. Use the `find` command to find the location of the `ypinit` command.

- In this example, the `ypinit` command is located in the `/usr/lib64/yp` directory.

```
# which ypinit
/usr/bin/which: no ypinit in (...)

# find / -name ypinit
/usr/lib64/yp/ypinit
```

- j. Include the absolute path name and run `ypinit` with the `-m` option. Do not add any additional hosts; press `Ctrl + D` to complete the list.

- Answer `y` when prompted “Is this correct”.

```
# /usr/lib64/yp/ypinit -m
```

At this point, we have to construct a list of the hosts which will run NIS servers. `host03.example.com` is in the list of NIS server hosts. Please continue to add the names for the other hosts, one per line. When you are done with the list, type a `<control D>`.

Next host to add: `host03.example.com`

Next host to add: `<CTRL-D>`

The current list of NIS servers looks like this:

`host03.example.com`

Is this correct? [y/n: y] **ENTER**

We need a few minutes to build the databases...

Building `/var/yp/nis.example.com/ypservers...`

Running `/var/yp/Makefile...`

Updating `passwd.byname...`

Updating `passwd.byuid...`

Updating `group.byname...`

Updating `group.bygid...`

...

`host03.example.com` has been set up as a NIS master server.

```
Now you can run ypinit -s host-3.example.com on all slave ser...
```

3. View the results of the NIS server configuration.

- Use the `ls` command to view the contents of the `/var/yp` directory.
  - The current directory is `/var/yp`. (You changed to this directory in step 2h.)
  - Note the addition of the `nis.example.com` directory and the `ypservers` file.

```
# ls
Makefile  nis.example.com  securenets  ypservers
```

- Use the `cat` command to display the contents of the `ypservers` file.

- Note that this file contains the host name of the NIS server.

```
# cat ypservers
host03.example.com
```

- Use the `ls` command to view the contents of the `nis.example.com` directory.

- Note that this directory contains the NIS maps.

```
# ls nis.example.com
group.byid  mailaliases  protocols.bynames  servicesbyname
...
```

4. Use the `systemctl` utility to stop the `firewalld` service on **host03**.

- You could create a firewall rule to allow NIS traffic.
- For the purposes of this exercise, stop the `firewalld` service.

```
# systemctl stop firewalld
```

## Practice A-2: Configuring an NIS Client

---

### Overview

In this practice, you configure an NIS client in preparation for implementing NIS authentication. You verify that the NIS client packages are installed, configure the NIS client, and start the NIS client service.

### Assumptions

- This practice is performed on **host01**.
- There is one command in step 5 that might be necessary to run on **host03**.

### Tasks

1. Log in to the **host01** VM guest from **dom0**.
  - a. If necessary, open a second terminal window on **dom0**.
  - b. From the second terminal window on **dom0**, use the `su -` command to become the root user.
    - The root password is `oracle`.

```
$ su -
Password: oracle
#
```

- c. As the root user on **dom0**, use the `ssh` command to log in to **host01**.
  - The root password is `oracle` (all lowercase).

```
[dom0]# ssh host01
root@host01's password: oracle
Last login: ...
[host01]#
```

2. On **host01**, install the required NIS client packages.
  - a. Use the `yum` command to list the available “`yp`” packages.
    - The example pipes the output to `grep` and searches for package names that begin with “`yp`”.
    - This sample output lists three packages.

```
# yum list available | grep "^\^yp"
yp-tools.x86_64    ...
ypbind.x86_64      ...
ypserv.x86_64      ...
```

- b. Use the `yum` command to install the `yp-tools` package and the `ypbind` package.
  - You do not need to install the `ypserv` package on NIS client systems.
  - Answer `y` when prompted “Is this ok”.

```
# yum install yp-tools ypbind
...
Transaction Summary
=====
```

```
Install 2 Packages (+2 Dependent packages)

Total download size: 279 k
Installed size: 570 k
Is this ok [y/d/N]: y
...
Complete!
```

3. Configure **host01** as an NIS client.

- a. Use the `vi` editor to edit the `/etc/sysconfig/network` file. Add the following entry to set the NIS domain name (`NISDOMAIN`) to `nis.example.com`.

```
# vi /etc/sysconfig/network
NISDOMAIN=nis.example.com
```

- Alternatively, you could use the `nisdomainname` command to set the NIS domain name to `nis.example.com`.
- Setting the `NISDOMAIN` from the command line is not persistent across a reboot.

```
# nisdomainname nis.example.com
```

- b. Use the `vi` editor to edit `/etc/yp.conf` and specify the NIS domain and NIS server.
- Use **host03** IP address, `192.0.2.103`, in this file and not the host name.
  - Using the host name might result in a “Host name lookup failure” when starting the `ypbind` service on the NIS client.

```
# vi /etc/yp.conf
domain nis.example.com server 192.0.2.103
```

- c. Use the `systemctl` command to start the `ypbind` service on **host01**.

```
# systemctl start ypbind
```

- d. Run the `ypwhich` command to display the NIS server name.

```
# ypwhich
host03.example.com
```

## Practice A-3: Implementing NIS Authentication

### Overview

In this practice, you use the Authentication Configuration Tool to implement NIS authentication.

### Assumptions

- This practice is performed exclusively on **host03**.
- Ensure that you are using `vncviewer` to connect to **host03** and not using `ssh`.
- You are the `root` user on **host03**.

### Tasks

1. Install the `authconfig-gtk` software package.
  - This package provides the `system-config-authentication` utility.

```
# yum install authconfig-gtk
...
Transaction Summary
=====
Install 1 Package

Total download size: 105 k
Installed size: 247 k
Is this ok [y/d/N] : y
...
Complete!
```

2. Open the Authentication Configuration Tool by running the system-config-authentication command.

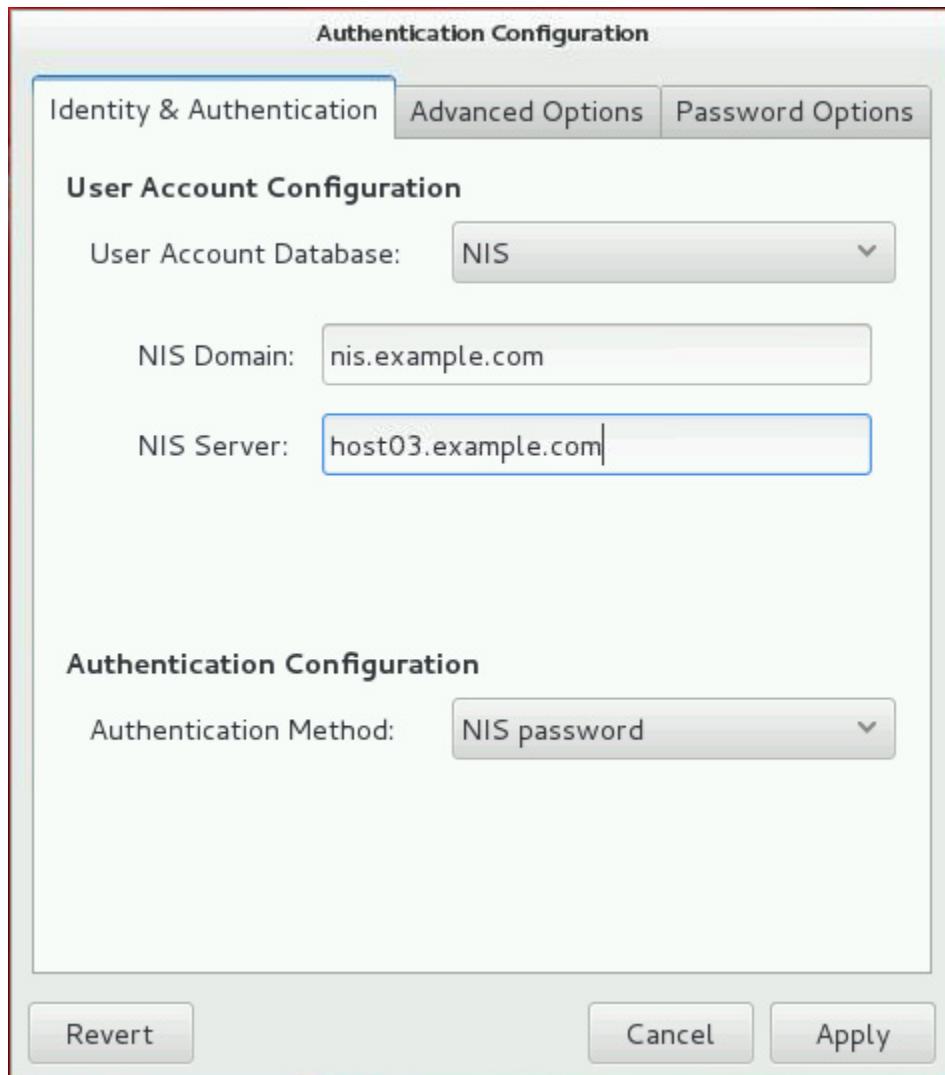
```
# system-config-authentication
```

- The GUI appears:



3. Make the following changes:
  - a. Select NIS from the User Account Database drop-down list.
  - b. Ensure that the NIS Domain is nis.example.com.
  - c. Enter host03.example.com as the NIS Server.

- d. Leave the Authentication Method as NIS password.
- Ensure that your screen looks like the following:



- Alternatively, you could implement NIS authentication from the command line by using the following command:

```
# authconfig --enablenis --nisdomain nis.example.com --nisserver
host03.example.com --update
```

- e. Click "Apply" to save your changes.
- After clicking "Apply," the Authentication Configuration Tool closes.

4. Run the authconfig --test command to view the authentication settings.
  - You can pipe the output to less if desired.

```
# authconfig --test
caching is disabled
...
nss_nis is enabled
  NIS server = "host03.example.com"
  NIS domain = "nis.example.com"
...
```

## Practice A-4: Testing NIS Authentication

---

### Overview

In this practice, you test NIS authentication by creating a new user on the NIS server and logging in as the new user from a remote system.

### Assumptions

- This practice is performed on **host01** and **host03** VMs.
- You are logged in as the `root` user on each system.
- The prompts include either **host01** or **host03** to indicate which system to enter the command from.

### Tasks

1. On **host03**, use the `useradd` and `passwd` commands to create a new user, `nis_user`. Assign a password of `password`. Ignore the “BAD PASSWORD” warning.

```
[host03]# useradd nis_user
[host03]# passwd nis_user
Changing password for user nis_user.
New password: password
BAD PASSWORD: The password is shorter than 9 characters
Retype new password: password
passwd: all authentication tokens updated successfully.
```

2. On **host03**, update the NIS maps by running `ypinit -m` (include the absolute path name). Do not add any additional hosts; press **Ctrl + D** to complete the list.
  - Answer `y` when prompted “Is this correct”.

```
[host03]# /usr/lib64/yp/ypinit -m
At this point, we have to construct a list of the hosts which
will run NIS servers. host03.example.com is in the list of NIS
server hosts. Please continue to add the names for the other
hosts, one per line. When you are done with the list, type a
<control D>.

Next host to add: host03.example.com
Next host to add: <CTRL-D>
The current list of NIS servers looks like this:
host03.example.com
Is this correct? [y/n: y] ENTER
...
```

3. From **host01**, use the `vi` editor to edit the `/etc/nsswitch.conf` file. Change the following entries to query NIS maps first before querying local files.

```
[host01]# vi /etc/nsswitch.conf
passwd:    files sss                      (old entry)
shadow:    files sss                      (old entry)
group:    files sss                      (old entry)
passwd:    nis  files sss                  (new entry)
shadow:    nis  files sss                  (new entry)
group:    nis  files sss                  (new entry)
```

4. From **host01**, log in as `nis_user` to test NIS authentication.

- a. Use the `logout` command to log out as the `root` user.

```
[host01]# logout
Connection to host01 closed.
```

- b. From **dom0**, log in as `nis_user`. Password is `password`.

```
[dom0]# ssh nis_user@host01
nis_user@host01's password: password
Could not chdir to home directory /home/nis_user: No such file
or directory
[host01]$ hostname
host01.example.com
[host01]$ whoami
nis_user
[host01]$ pwd
/
```

- Notice that you were able to authenticate from the NIS server and log in from **host01**.
- However, the `nis_user` home directory is on the NIS server, **host03**.
- In the next practice, you export the `nis_user` home directory as an NFS share and auto-mount the home directory upon `nis_user` remote login.

## Practice A-5: Auto-Mounting a User Home Directory

---

### Overview

In this practice, you export a user home directory as an NFS share, and you configure automounter on a client system to auto-mount the remote home directory on login.

### Assumptions

- This practice is performed on **host01** and on **host03** VMs.
- The prompts include either **host01** or **host03** to indicate which system to enter the command from.

### Tasks

1. Log in to **host01** as the root user.

- a. From **host01**, use the `logout` command to log out as `nis_user`.

```
[host01]# logout
Connection to host01 closed.
```

- b. From **dom0**, log in as root on **host01**. Password is `oracle`.

```
[dom0]# ssh host01
root@host01's password: oracle
Last login: ...
[root@host01 ~]#
```

2. Configure automounter on **host01**.

- Install the `nfs-utils` on **host01** to auto-mount an NFS share on a remote system.
- a. On **host01**, use the `yum` command to install the `nfs-utils` package.
- Answer `y` when prompted “Is this ok”.

```
[host01]# yum install nfs-utils
...
Transaction Summary
=====
Install 1 Package (+15 Dependent package)

Total download size: 1.3 M
Installed size: 3.9 M
Is this ok [y/d/N]: y
...
Complete!
```

- b. On **host01**, use the `yum` command to install the `autofs` package.
- Answer `y` when prompted “Is this ok”.

```
[host01]# yum install autofs
...
Transaction Summary
=====
Install 1 Package (+1 Dependent package)

Total download size: 818 k
Installed size: 5.1 M
Is this ok [y/d/N]: y
...
Complete!
```

- c. Use the `vi` editor to edit `/etc/auto.master` and add the following entry to the beginning of the file:

```
[host01]# vi /etc/auto.master
/home    /etc/auto.home
```

- d. Use the `vi` editor to create the `/etc/auto.home` file with the following entry:

```
[host01]# vi /etc/auto.home
nis_user -fstype=nfs    host03:/home/nis_user
```

- e. Use the `systemctl` utility to enable and start the `autofs` service.

```
[host01]# systemctl enable autofs
ln -s '/usr/lib/systemd/system/autofs.service'
'/etc/systemd/system/multi-user.target.wants/autofs.service'
[host01]# systemctl start autofs
```

### 3. Configure NFS on **host03**.

- a. From **host03**, use the `vi` editor to edit `/etc/exports` and add the following entry:

```
[host03]# vi /etc/exports
/home/nis_user * (rw)
```

- b. Use the `systemctl` utility to enable and start the `nfs-server` service.

```
[host03]# systemctl enable nfs-server
ln -s '/usr/lib/systemd/system/nfs-server.service'
'/etc/systemd/system/multi-user.target.wants/nfs-server.service'
[host03]# systemctl start nfs-server
```

- c. Use the `showmount` command to display the exported file systems.

- Notice that the `nis_user` home directory is now exported.

```
[host03]# showmount -e
Export list for host03.example.com:
/home/nis_user *
```

4. From **host01**, log in as `nis_user` to test auto-mounter.

- a. Use the `logout` command to log out as root user.

```
[host01]# logout
Connection to host01 closed.
```

- b. Log in as `nis_user`. Password is `password`.

```
[dom0]# ssh nis_user@host01
nis_user@host01's password: password
[nis_user@host01]$ hostname
host01.example.com
[nis_user@host01]$ whoami
nis_user
[nis_user@host01]$ pwd
/home/nis_user
```

- Notice that you were able to authenticate from the NIS server and log in, and auto-mount the `nis_user` home directory.

- c. Use the `df` command to display the mounted file systems.

- Notice that the `host03:/home/nis_user` file system is mounted on local file system `/home/nis_user`.

```
[nis_user@host01 ~]$ df -h
Filesystem      Size  Used  Avail   Use%  Mounted on
...
host03:/home/nis_user
      11G  3.2G  7.1G   31%  /home/nis_user
```

- d. Use the `vi` editor to create a file and confirm read-write permissions.

- Create any file name in the `nis_user` home directory with any content.
- Save the file to confirm write permission to the file system.

```
[nis_user@host01 ~] $ vi test_file
Insert some text
Save the file by pressing <Esc> then :wq
```

## Practice A-6: Restoring the Systems to Their Original State

---

### Overview

In this practice, you unconfigure NIS, NFS, and automounter and restore **host01** and **host03** to their original state.

### Assumptions

- This practice is performed on **host01** and **host03** VMs.
- The prompts include either **host01** or **host03** to indicate which system to enter the command from.

### Tasks

#### 1. Disable NIS on **host01**.

- a. Use the `logout` command to log out as `nis_user`.

```
[nis_user@host01]# logout
Connection to host01 closed.
```

- b. From **dom0**, log in as root on **host01**. Password is `oracle`.

```
[dom0]# ssh host01
root@host01's password: oracle
Last login: ...
[root@host01 ~]#
```

- c. Use the `vi` editor to edit `/etc/sysconfig/network` and remove the following entry that sets the NIS domain name (`NISDOMAIN`) to `nis.example.com`.

```
[host01]# vi /etc/sysconfig/network
NISDOMAIN=nis.example.com
```

- d. Use the `vi` editor to edit `/etc/yp.conf` and remove the following entry that specifies the NIS domain and NIS server.

```
[host01]# vi /etc/yp.conf
domain nis.example.com server 192.0.2.103
```

- e. Use the `systemctl` utility to stop and disable the `ypbind` service.

```
[host01]# systemctl stop ypbind
[host01]# systemctl disable ypbind
rm '/etc/systemd/system/multi-user.target.wants/ypbind.service'
```

- f. Use the `vi` editor to edit the `/etc/nsswitch.conf` file.

- Change the following entries to not query NIS maps.

```
[host01]# vi /etc/nsswitch.conf
passwd:  nis files sss                      # delete nis
shadow:  nis files sss                      # delete nis
group:   nis files sss                      # delete nis
```

2. Disable automounter on **host01**.
- Use the `vi` editor to edit `/etc/auto.master` and remove the following entry that references home directories.

```
[host01]# vi /etc/auto.master
/home   /etc/auto.home
```

- Use the `rm` command to remove the `/etc/auto.home` file.

```
[host01]# rm /etc/auto.home
rm: remove regular file '/etc/auto.home'? y
```

- Use the `systemctl` utility to stop and disable the `autofs` service.

```
[host01]# systemctl stop autofs
[host01]# systemctl disable autofs
rm '/etc/systemd/system/multi-user.target.wants/autofs.service'
```

- Use the `logout` command to log out of **host01**.

```
[host01]# logout
Connection to host01 closed.
```

3. Disable NFS on **host03**.

- From **host03**, use the `systemctl` utility to stop and disable the `nfs-server` service.

```
[host03]# systemctl stop nfs-server
[host03]# systemctl disable nfs-server
rm '/etc/systemd/system/multi-user.target.wants/nfs-
server.service'
```

- Use the `vi` editor to edit `/etc/exports` and remove the following entry:

```
[host03]# vi /etc/exports
/home/nis_user *(rw)
```

4. Disable NIS on **host03**.

- From **host03**, use the `systemctl` utility to stop and disable the `ypserv` service.

```
[host03]# systemctl stop ypserv
[host03]# systemctl disable ypserv
rm '/etc/systemd/system/multi-user.target.wants/ypserv.service'
```

- From **host03**, use the `systemctl` utility to stop and disable the `ypxfrd` service.

```
[host03]# systemctl stop ypxfrd
[host03]# systemctl disable ypxfrd
rm '/etc/systemd/system/multi-user.target.wants/pxfrd.service'
```

- From **host03**, use the `systemctl` utility to stop and disable the `yppasswdd` service.

```
[host03]# systemctl stop yppasswdd
[host03]# systemctl disable yppasswdd
rm '/etc/systemd/system/multi-
user.target.wants/yppasswdd.service'
```

- d. Use the `vi` editor to edit the `/etc/sysconfig/network` file and remove the following line.

```
[host03]# vi /etc/sysconfig/network  
NISDOMAIN=nis.example.com
```

- e. Use the `vi` editor to remove the following access rule in the `/etc/ypserv.conf` file.

```
[host03]# vi /etc/ypserv.conf  
192.0.2.1/24 : * : * : none
```

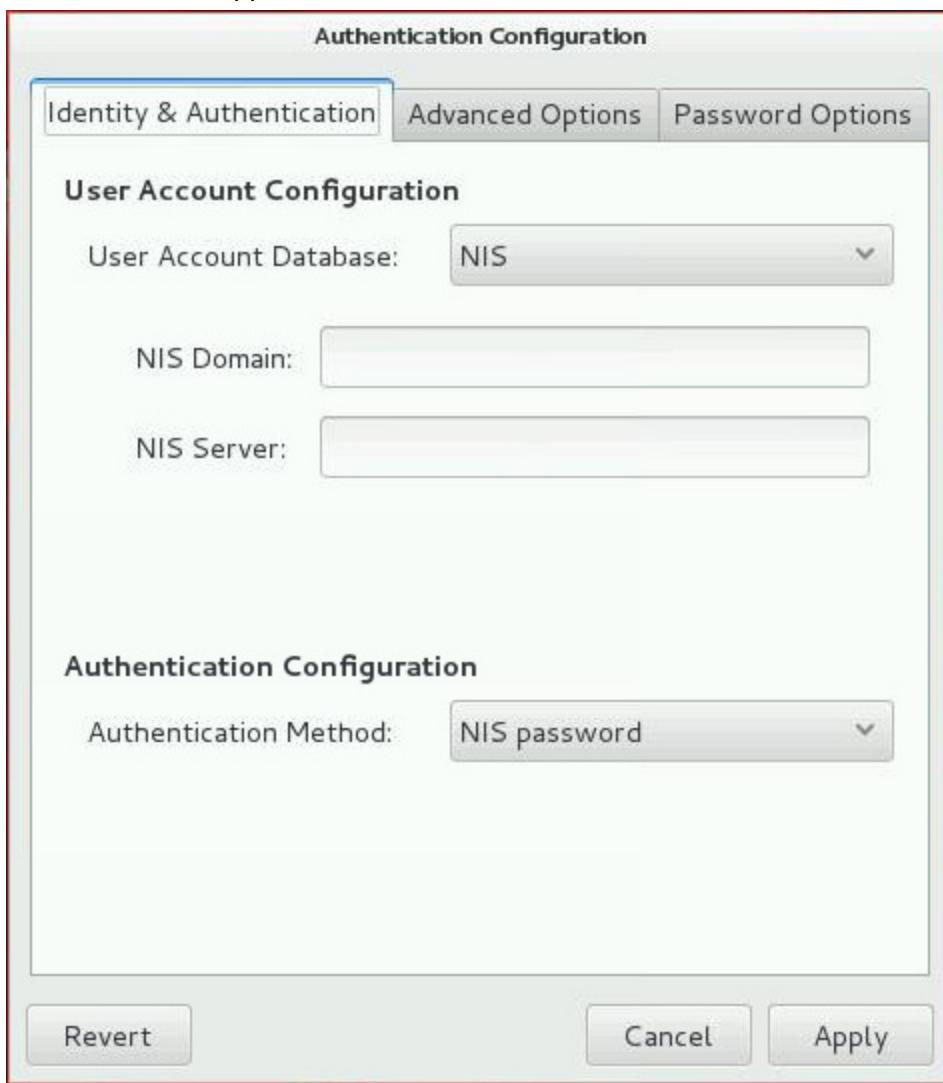
- f. Use the `rm` command to delete the `/var/yp/securenets` file.

```
[host03]# rm /var/yp/securenets  
rm: remove regular file '/var/yp/securenets'? y
```

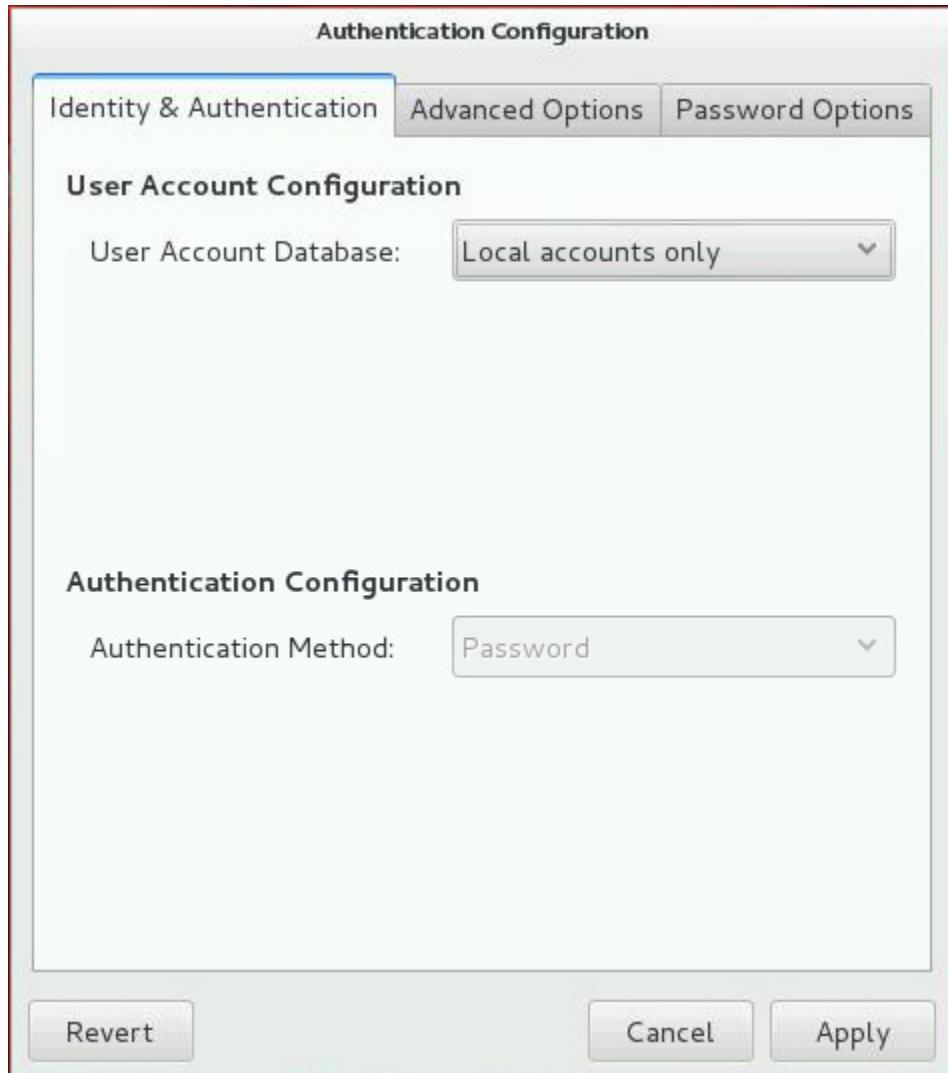
5. On **host03**, open the Authentication Configuration GUI and change the User Account Database selection.
  - a. Run the `system-config-authentication` command to display the GUI.

```
[host03]# system-config-authentication
```

- The GUI appears:



- b. Select Local accounts only from the User Account Database drop-down list.
- Ensure that your screen looks like the following:



- c. Click "Apply" to save your changes.
- After clicking "Apply," the Authentication Configuration Tool closes.
- d. Run the authconfig --test command to view the authentication settings.
- You can pipe the output to less if desired.

```
# authconfig --test
caching is disabled
...
nss_nis is disabled
  NIS server = ""
  NIS domain = ""
...
```

6. Use the systemctl utility to start the firewalld service on host03.

```
[host03]# systemctl start firewalld
```

# **Appendices: Remote Access Options**

**Chapter 21**

## Appendices: Overview

---

### Appendices Overview

The four appendixes show the various options for accessing your student PC remotely:

- Appendix A: Using an NX Client to Connect to **dom0**
- Appendix B: Using an NX Player to Connect to **dom0**
- Appendix C: Using VNC (TightVNC) to Connect Directly to VM Guests
- Appendix D: Using NoMachine Version 4 to Connect to **dom0**

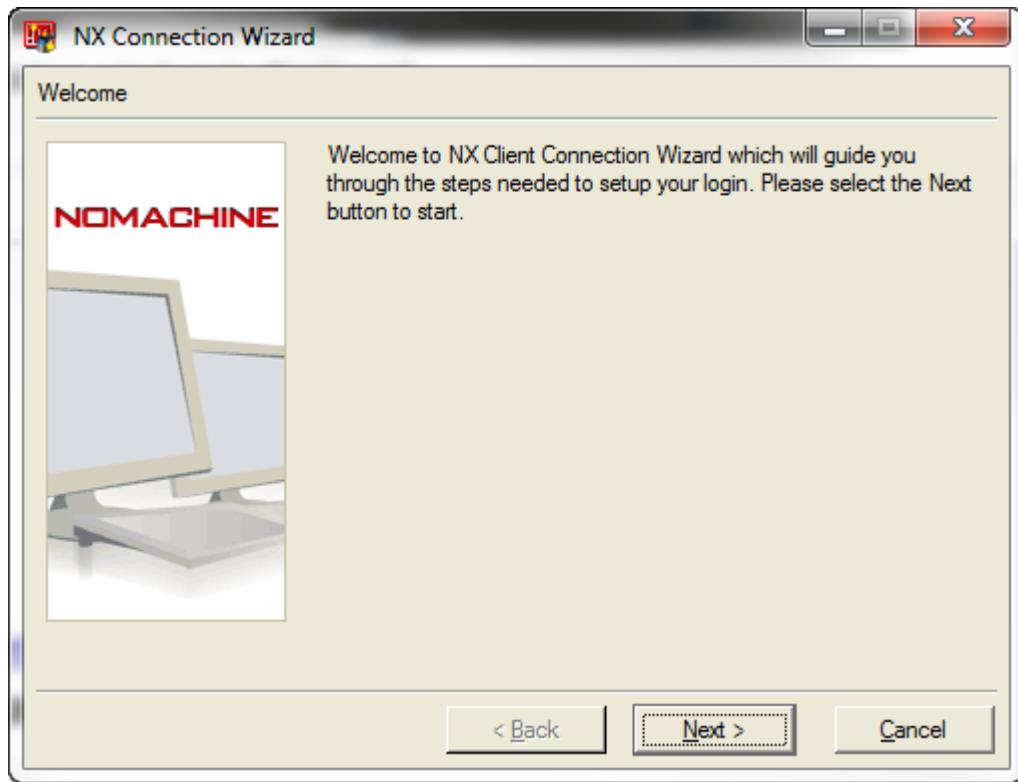
## Appendix A: Using an NX Client to Connect to dom0

### Overview

This appendix discusses accessing your student PC (**dom0**) remotely by using NX Client. The NX Client in this appendix is NX Client for Windows, Version 3.5.0-9.

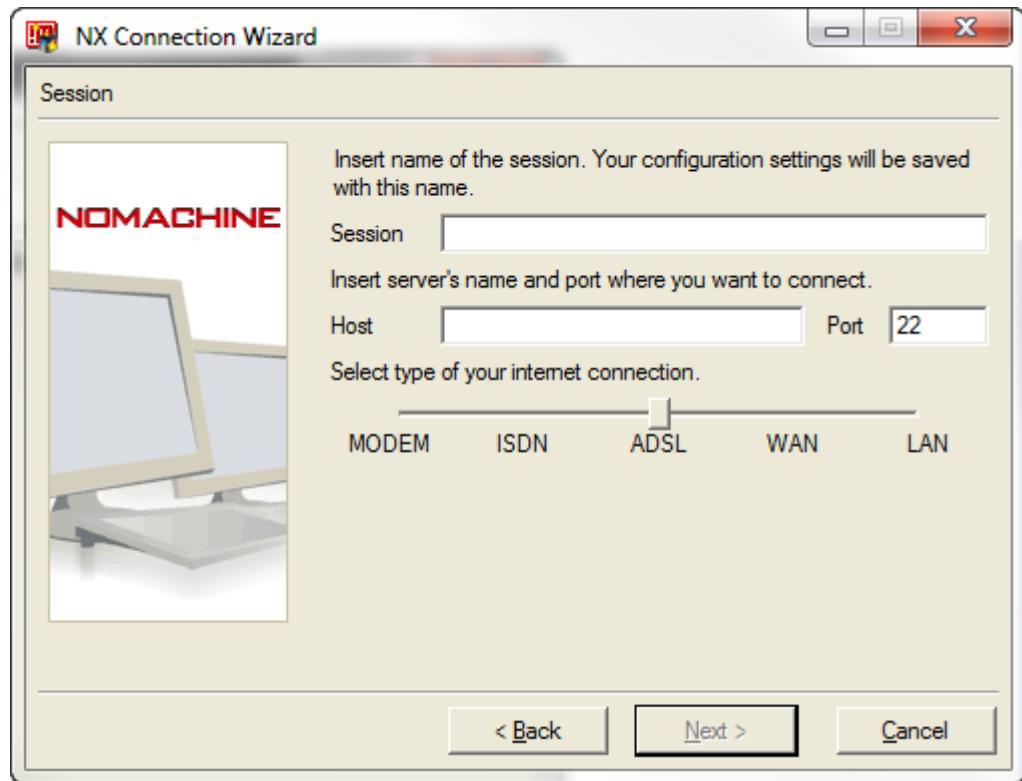
### Steps

1. Install **NX Client** (if necessary) from <http://www.nomachine.com/download.php>.
2. Run **NX Client**. (For example, select **NX Client for Windows** from the Windows Start menu.)
  - An NX Connection Wizard steps you through creating the initial session.
  - The following Welcome window appears.



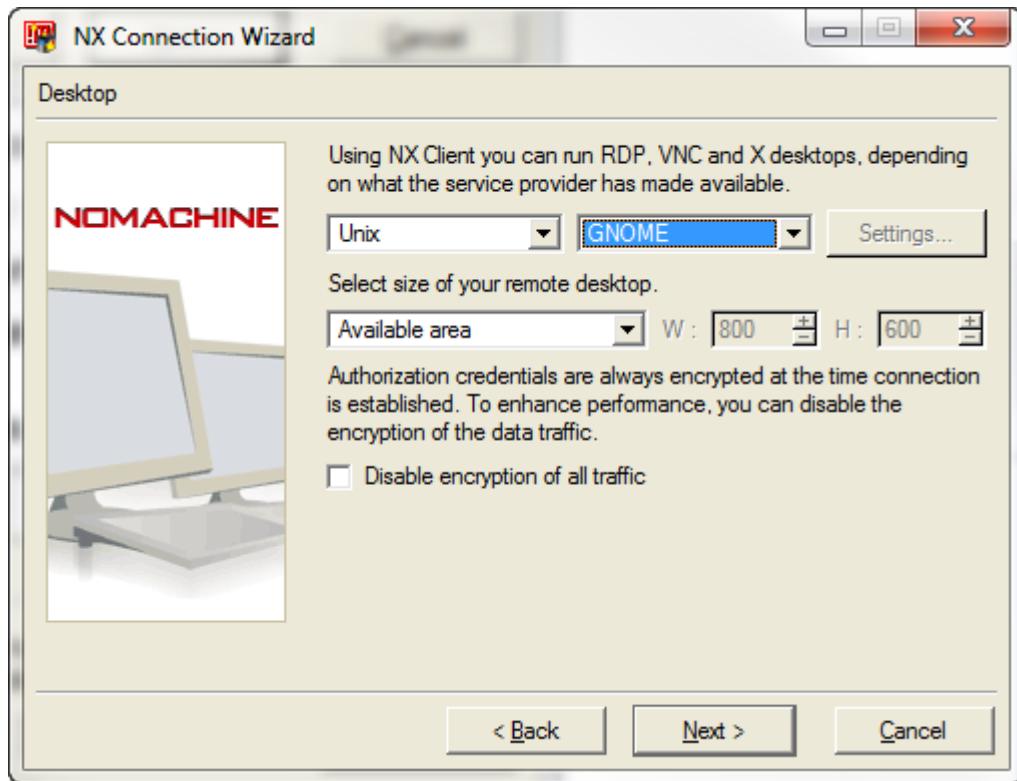
- a. Click **Next**.

- The following Session window appears.

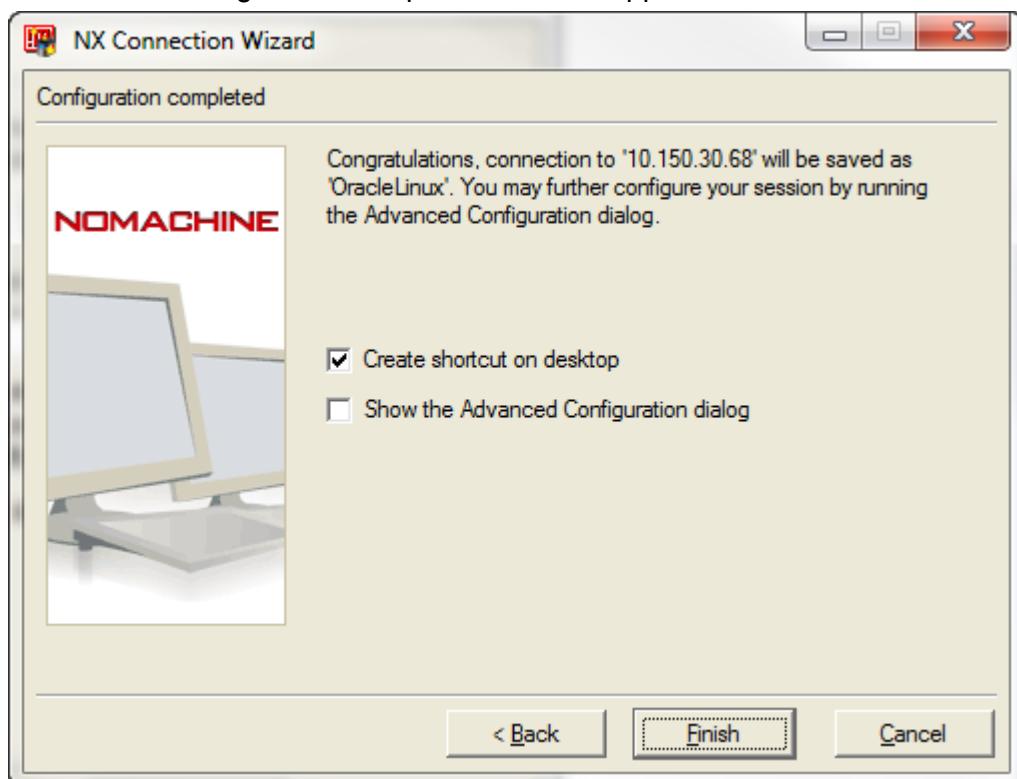


- b. Enter anything you like for **Session** (for example, OracleLinux).
- c. Enter the IP address (provided by your instructor) for **Host**.
- d. Accept the remaining defaults and click **Next**.
- e. The Desktop window appears. Change KDE to **GNOME** by selecting from the drop-down list.

- Your window must look like the following:

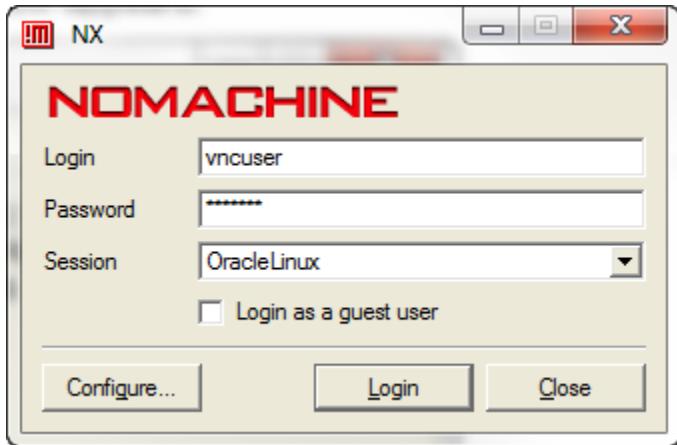


- f. Accept all other defaults and click **Next**.
- The “Configuration completed” window appears.



- g. Click **Finish**.

- The NX Login window appears.



- h. For **Login**, enter vncuser.
- i. For **Password**, enter vnctech.
- j. Your **Session** defaults to the session that you just created. In this example, the **Session** is OracleLinux. Your session name may be different.
- k. Click **Login**.
  - The **dom0** GNOME virtual desktop window appears.
  - Future connections will bypass the configuration wizard and will only open the NX Login window.

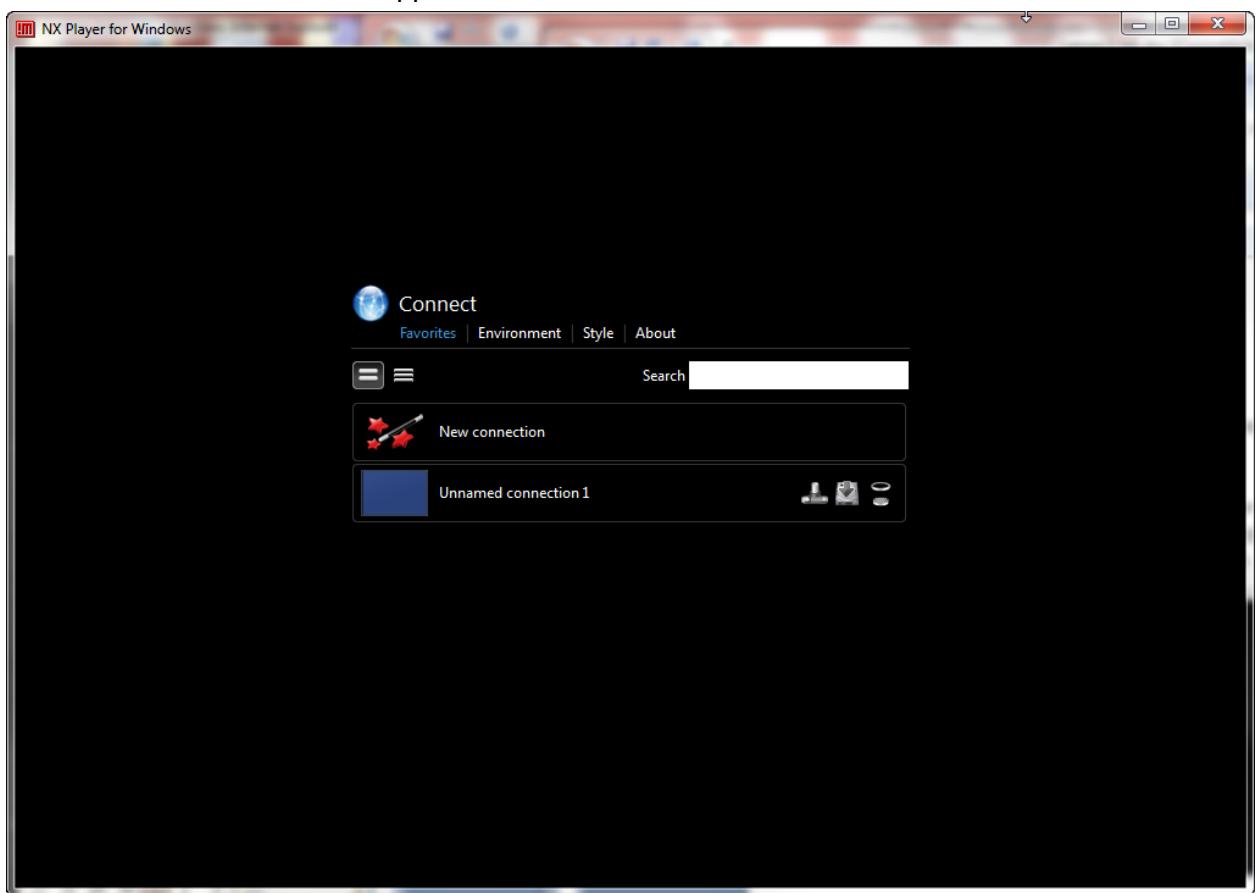
## Appendix B: Using an NX Player to Connect to dom0

### Overview

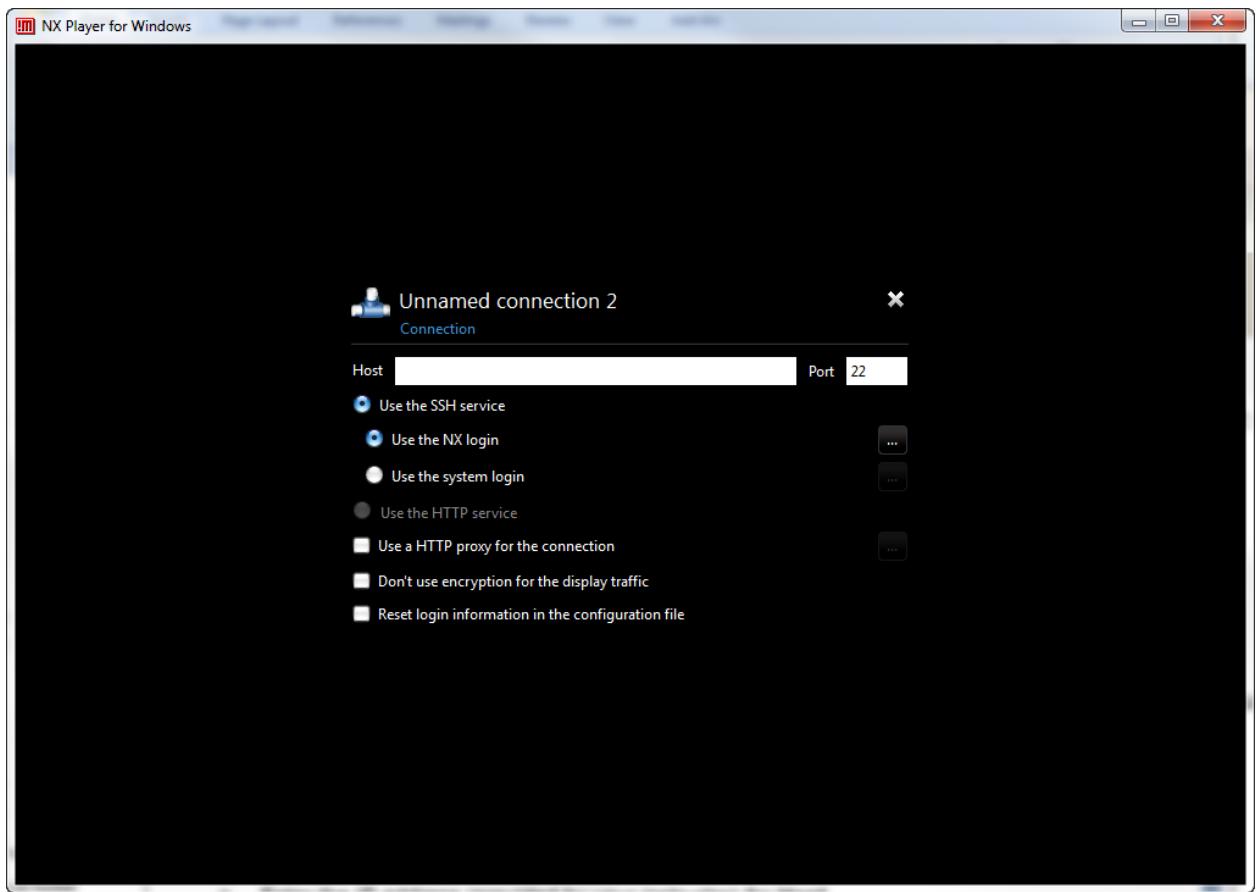
This appendix discusses accessing your student PC (**dom0**) remotely using NX Player. The NX Player in this appendix is NX Player for Windows, Preview 5, version 4.0.132.

### Steps

1. Install **NX Player** (if necessary) from <http://www.nomachine.com/download.php>.
2. Run **NX Player**. (For example, select **NX Player for Windows** from the Windows Start menu.)
  - a. Ensure that the **Favorites** tab is selected.
    - The Connect window appears.

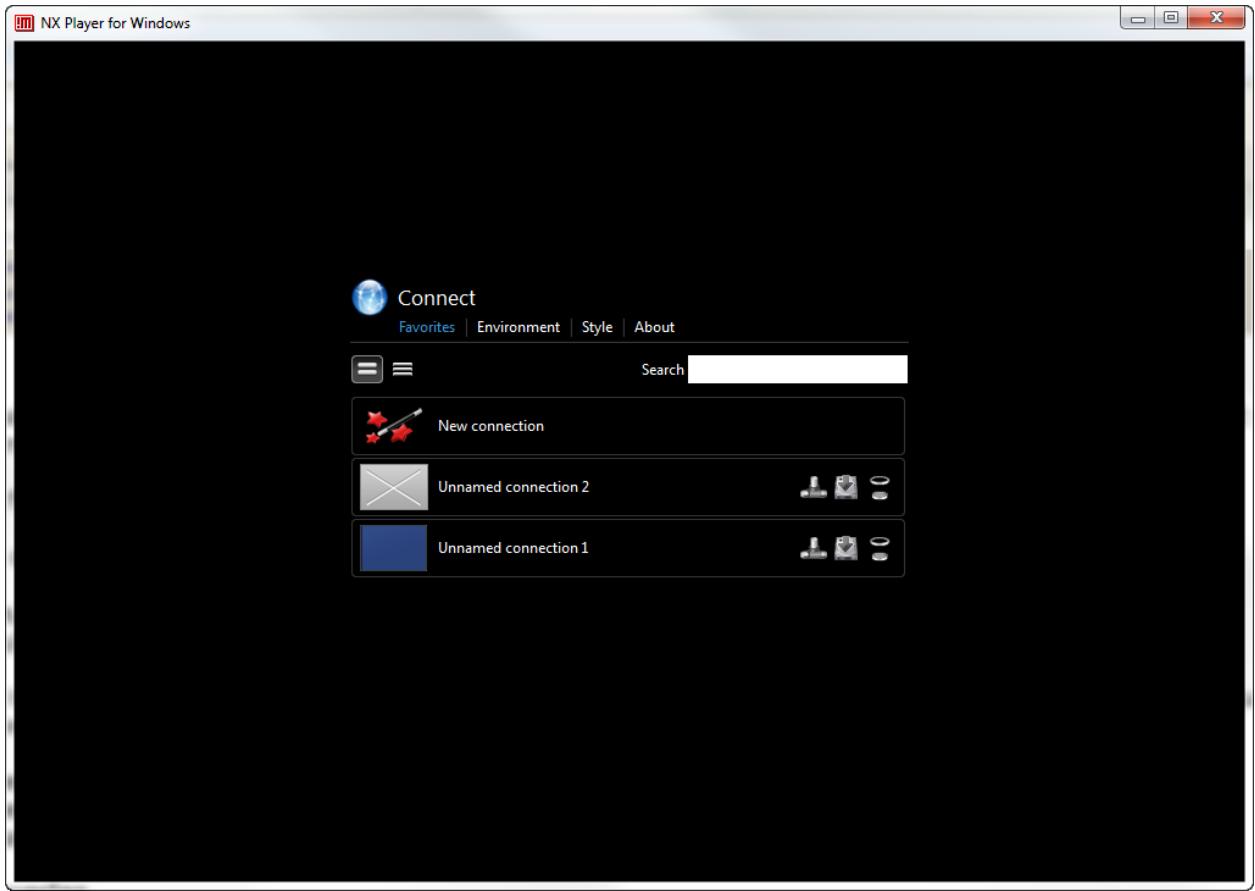


- b. Click **New connection** to display the following window:



- c. Enter the IP address (provided by your instructor) for **Host**.  
d. Accept the defaults:  
1) Port 22  
2) Use the SSH service  
3) Use the NX login  
e. Note the connection name. In this example, it is **Unnamed connection 2**. Yours is most likely **Unnamed connect 1**.

f. Press **Enter**. The following window appears:

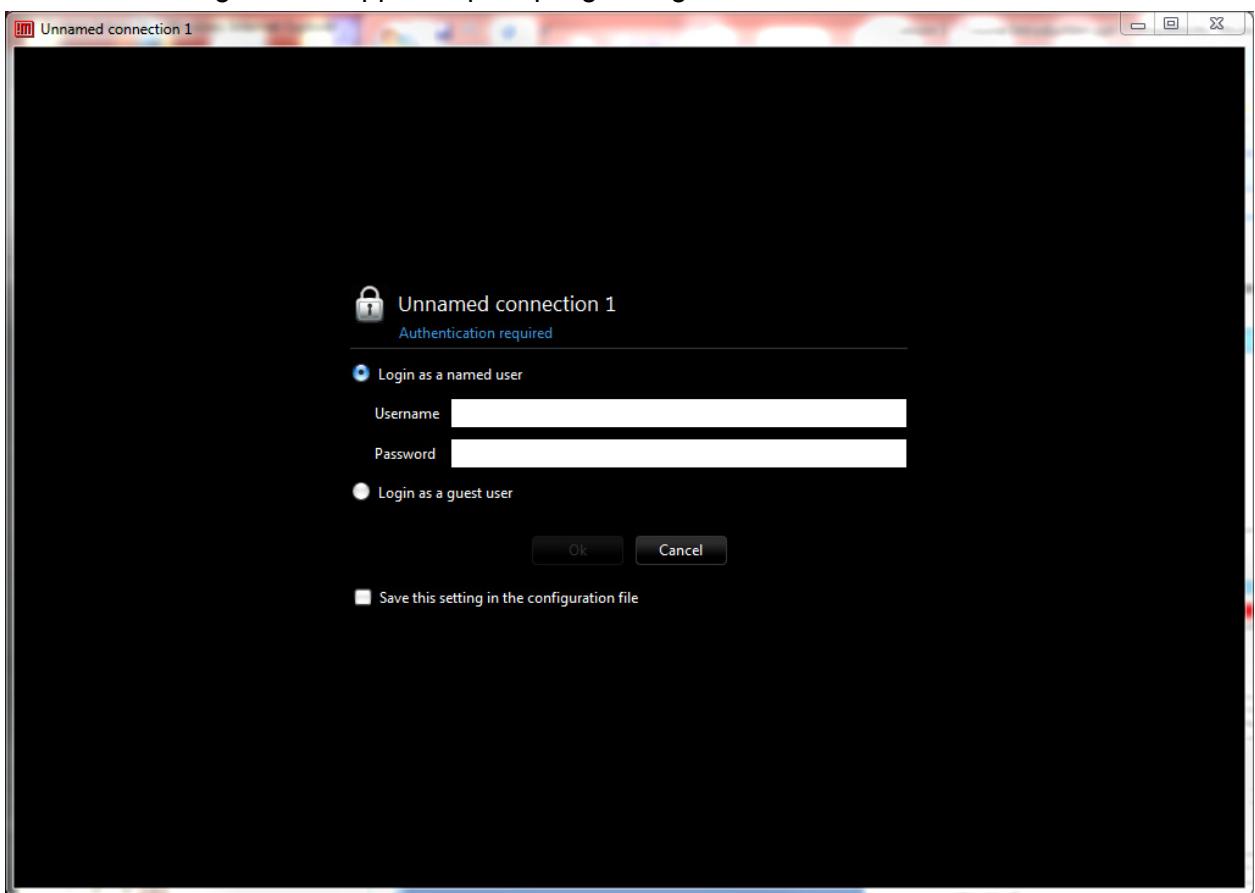


g. Click the connection that you just created (**Unnamed connection 1**, for example).

- The Login window appears.

3. Log in.

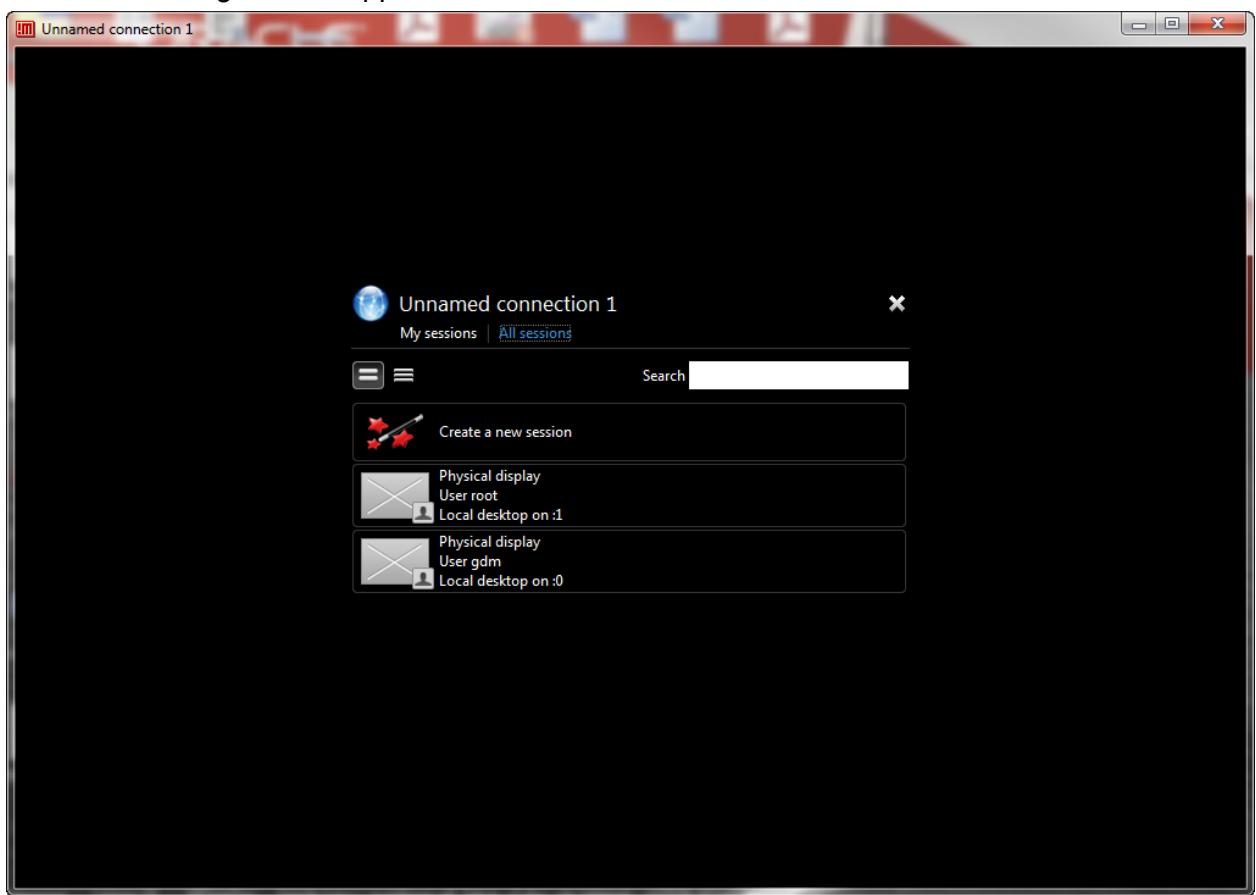
- The following window appears, prompting for login authentication.



- Ensure that **Login as a named user** is selected.
- For **Username**, enter vncuser.
- For **Password**, enter vnctech.
- Click **Ok**.

4. Create a new session.

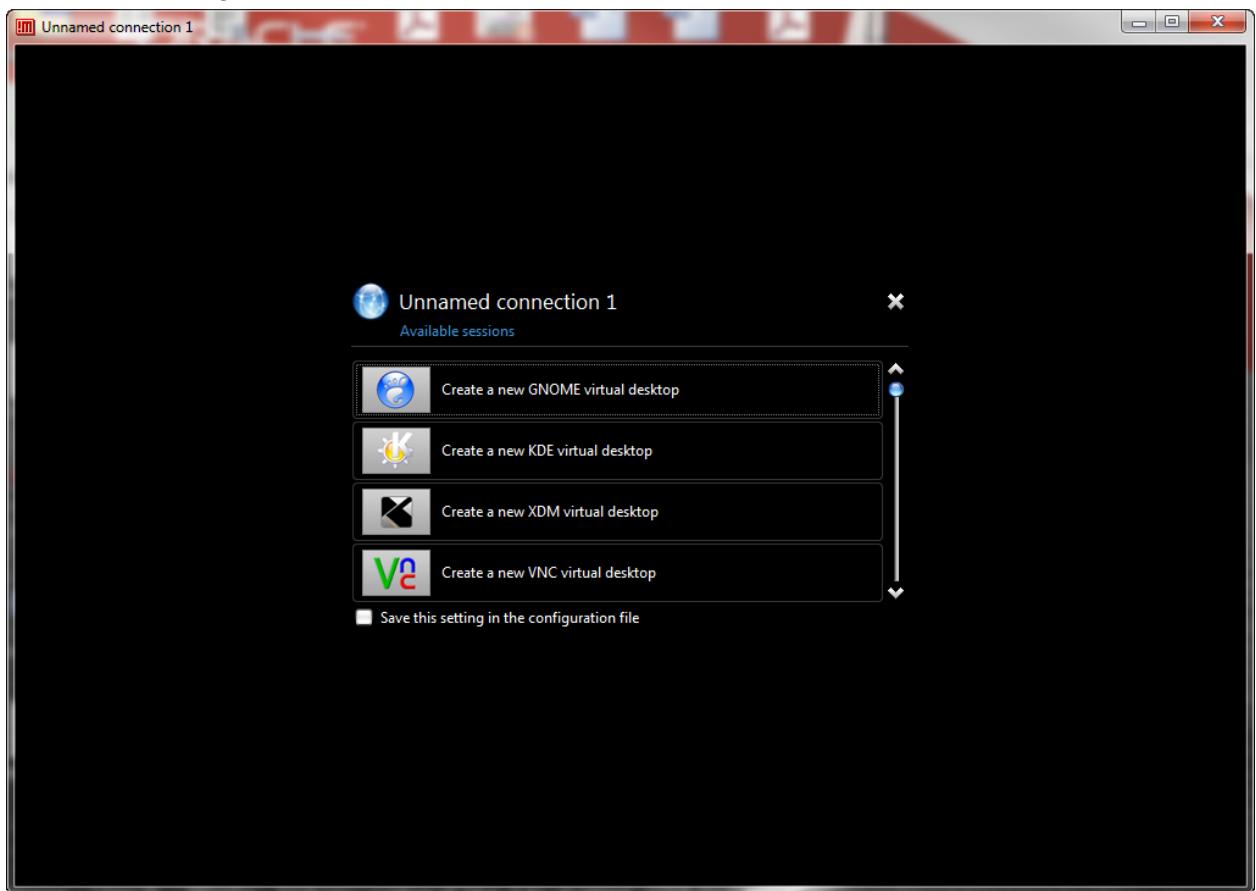
- The following window appears:



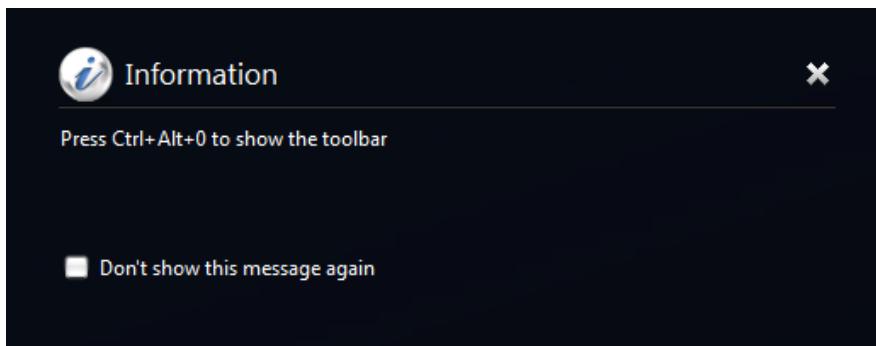
- a. Click "Create a new session."

5. Create a new GNOME virtual desktop.

- The following window appears:



- Click Create a new GNOME virtual desktop.
- Click the **X** in the **Information** message box to close the box.



- The **dom0** GNOME virtual desktop window appears.

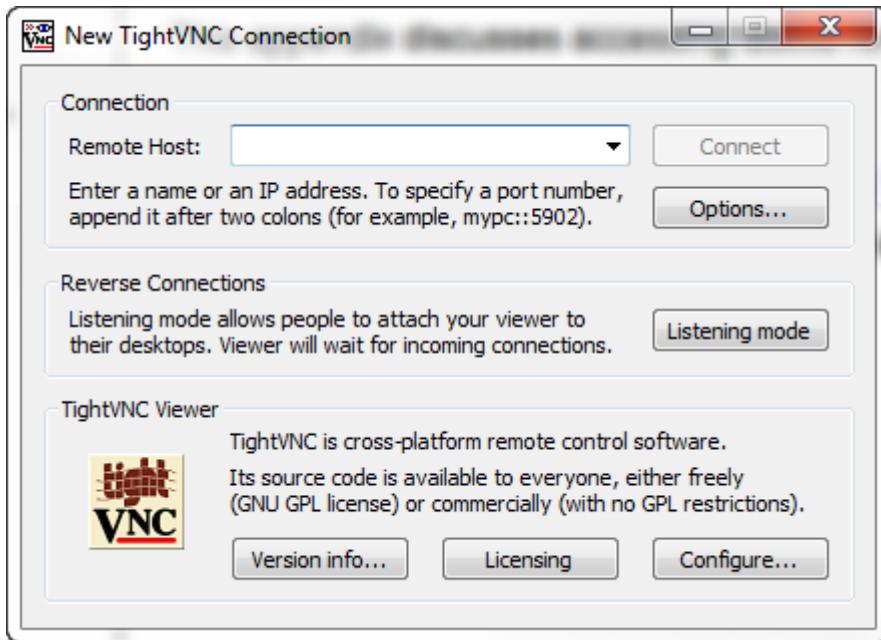
## Appendix C: Using VNC (TightVNC) to Connect Directly to VM Guests

### Overview

This appendix discusses accessing the VM guest systems that directly uses VNC (TightVNC). It is not recommended to connect to **dom0** or to the **host03** VM by using VNC. Both **dom0** and **host03** have the GNOME user interface, which causes various problems when connecting using VNC.

### Steps

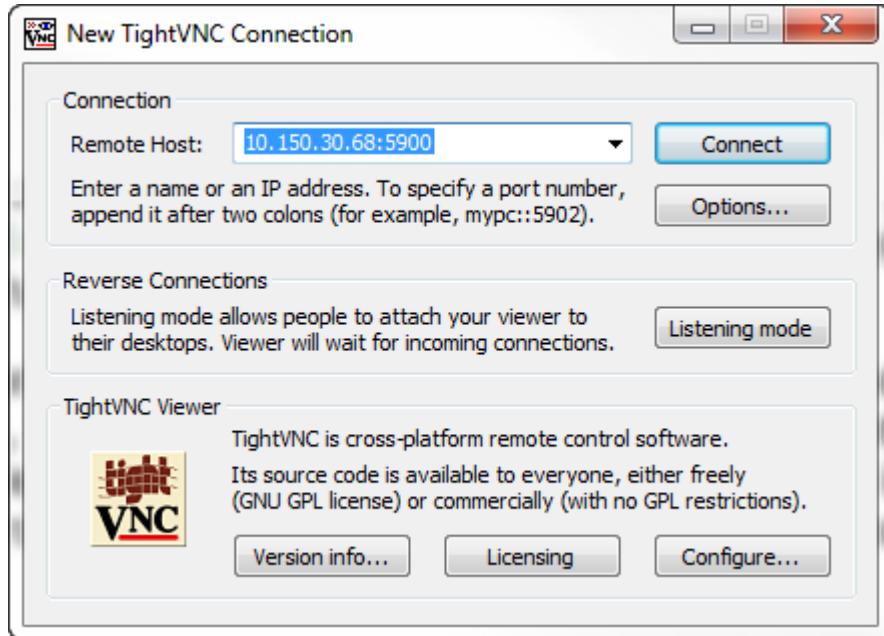
1. Install **tightvnc** (if necessary) from <http://www.tightvnc.com/>.
2. Run **TightVNC Viewer**. (For example, select **TightVNC Viewer** from the Windows Start menu.)
  - The following New TightVNC Connection window appears:



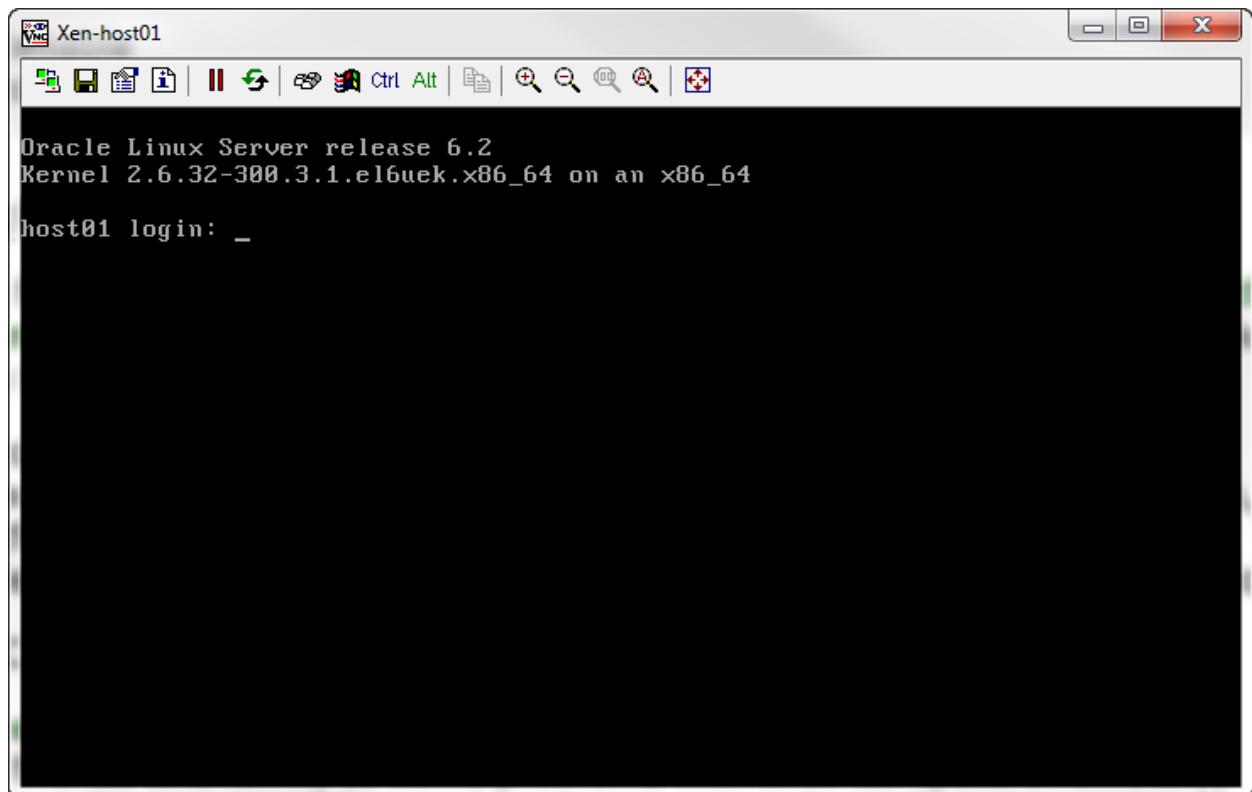
3. Connect directly to your **host01** virtual machine.
  - The following assumptions are made:
    - The **host01** VM was created first (has a port number of 5900).
    - The **host02** VM was created next (has a port number of 5902).
    - The **host03** VM was created last (has a port number of 5903).
  - The output of the following commands (from **dom0** as **root**) indicates that this assumption is true.

```
# xm list -l host01 | grep location
        (location 0.0.0.0:5900)
# xm list -l host02 | grep location
        (location 0.0.0.0:5902)
# xm list -l host03 | grep location
        (location 0.0.0.0:5903)
```

- Enter the IP address (provided by your instructor), followed by the port number to connect directly to a VM guest.
- a. To connect directly to the **host01** VM, enter the following.
  - In this example, the IP address of your student PC is 10.150.30.68. Your IP address is different.



- b. Click **Connect**.
- A terminal window appears.



- c. Log in as `root` with the password `oracle` (leading zero, not the letter "O").

- d. Enter the `hostname` command to confirm that you are logged in to **host01**.

```
# hostname  
host01.example.com
```

- e. Log off by entering the `exit` command.  
f. Close the VNC window by clicking the **X** in the top-right corner of the window.

## Appendix D: Using NoMachine Version 4 to Connect to dom0

### Overview

This appendix describes the procedure to access your lab machine remotely by using the NoMachine client for Windows. This procedure assumes that you have downloaded and installed the NoMachine client from the following location:

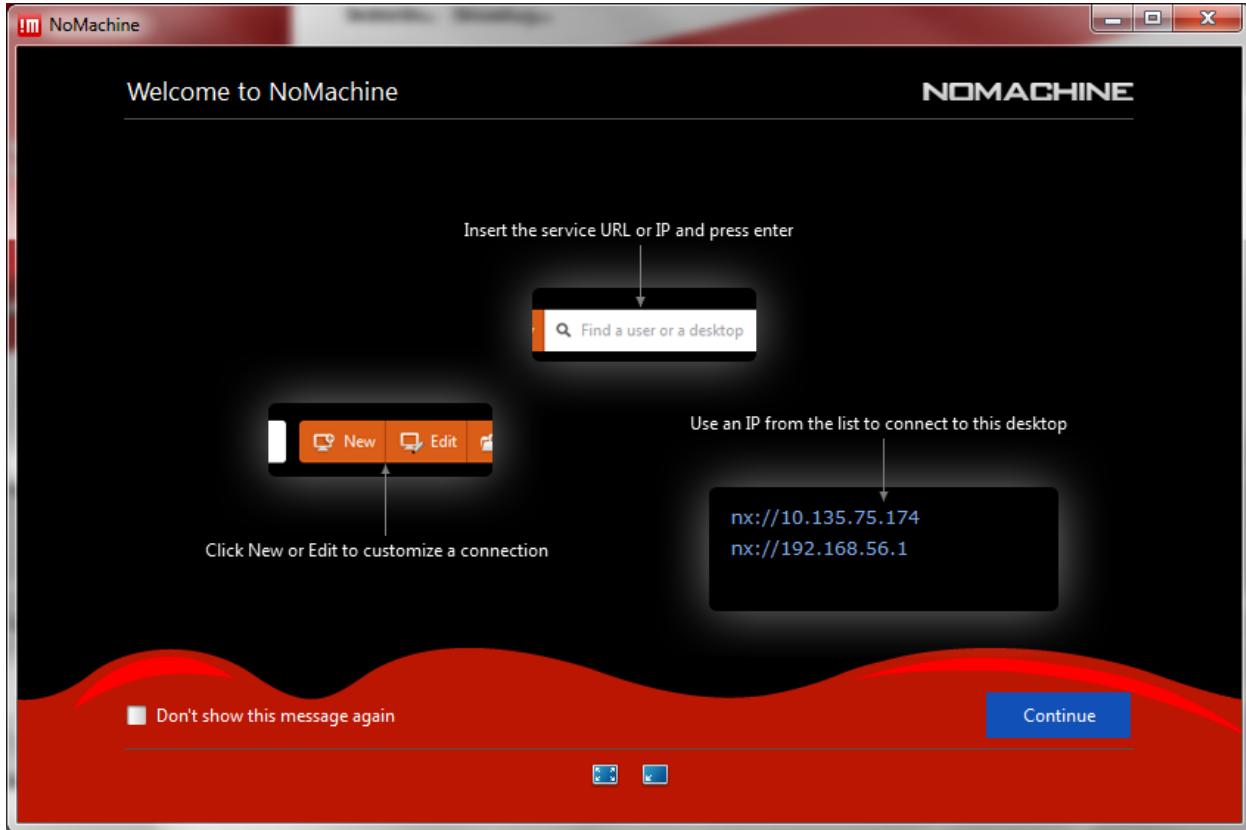
<http://www.nomachine.com/download.php>.

**Note:** If you are accessing your lab environment remotely, you have received instructions on how to access your lab machine. The following steps summarize the configuration and connection tasks when using the NoMachine client for Windows 7.

### Tasks

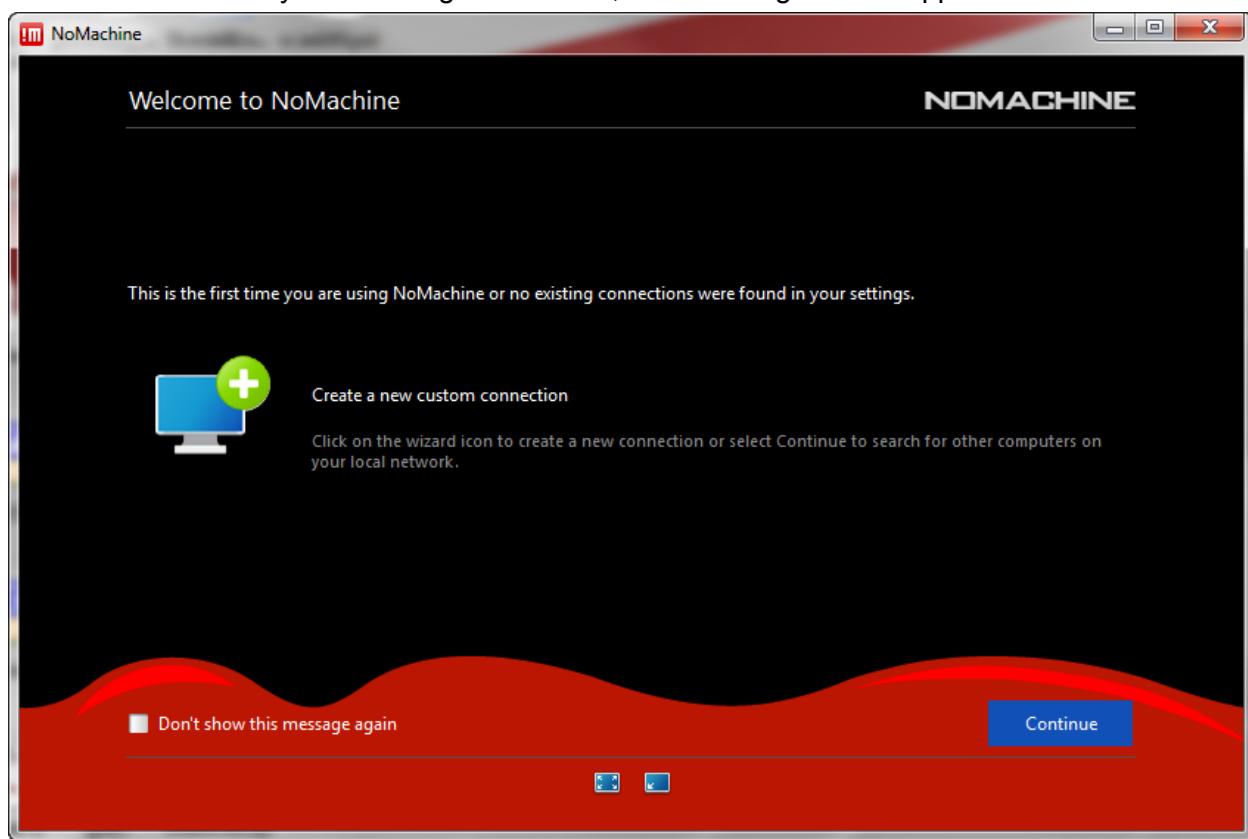
1. Create a session to your assigned lab machine by using the NoMachine Connection Wizard.
  - a. Select the NoMachine program from the Windows Start menu.

The “Welcome to NoMachine” window appears.

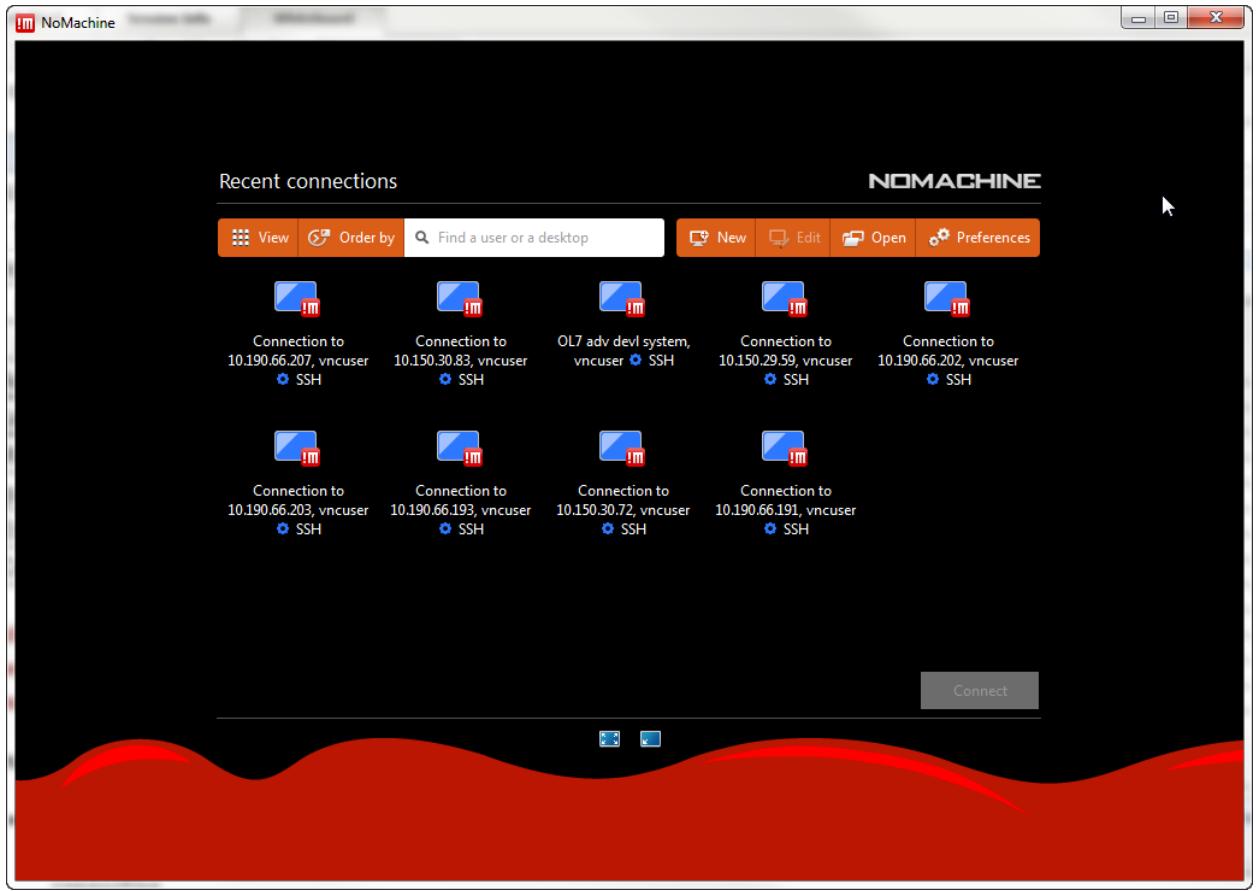


b. Click Continue.

If this is the first time you are using NoMachine, the following window appears:



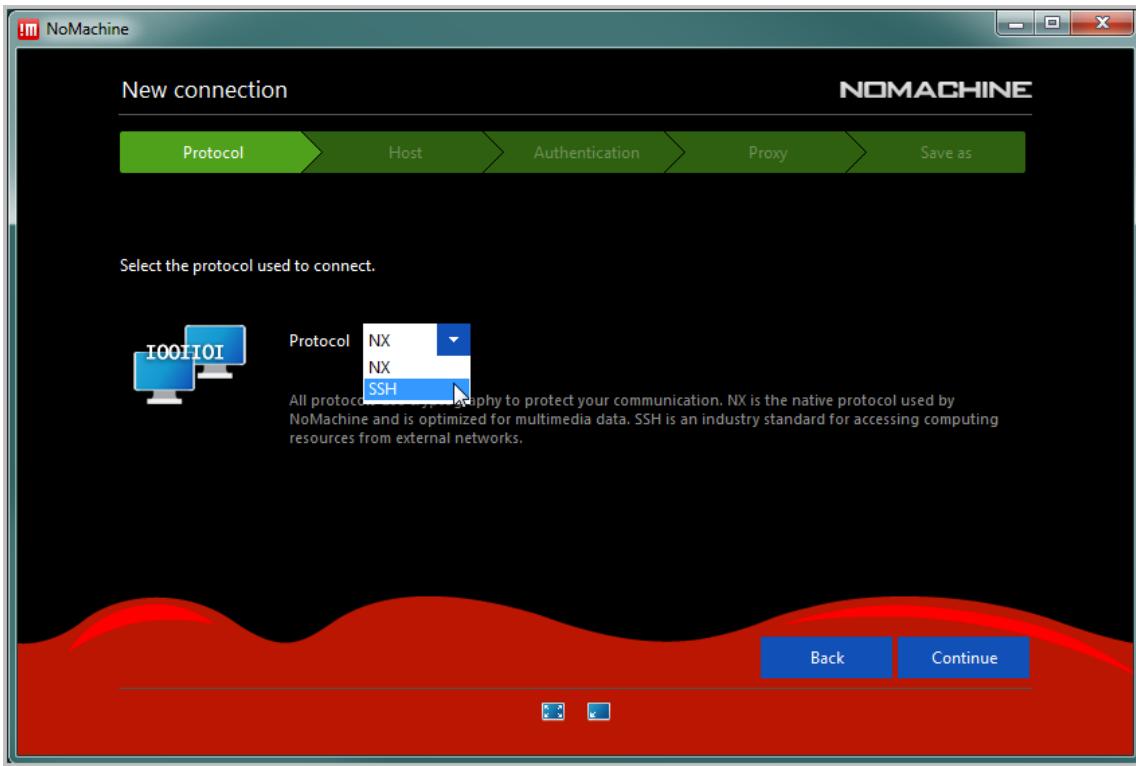
Otherwise, the “Recent connections” window appears.



- c. Click “New” to add a new connection.

The Protocol window of the New Connection Wizard appears.

- d. Select SSH from the Protocol drop-down list.

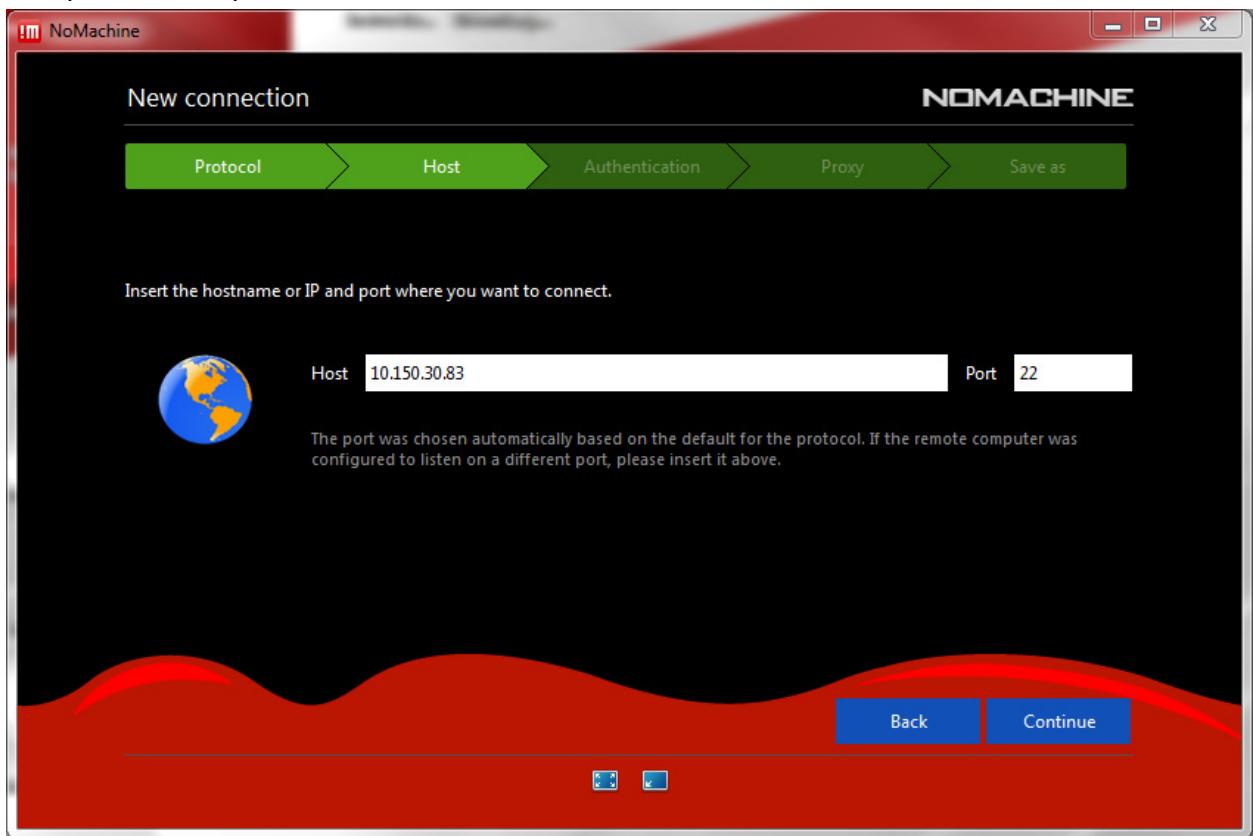


- e. Click the Continue button.

The Host window appears.

- f. In the Host field, enter the IP address that was provided to you by your instructor.

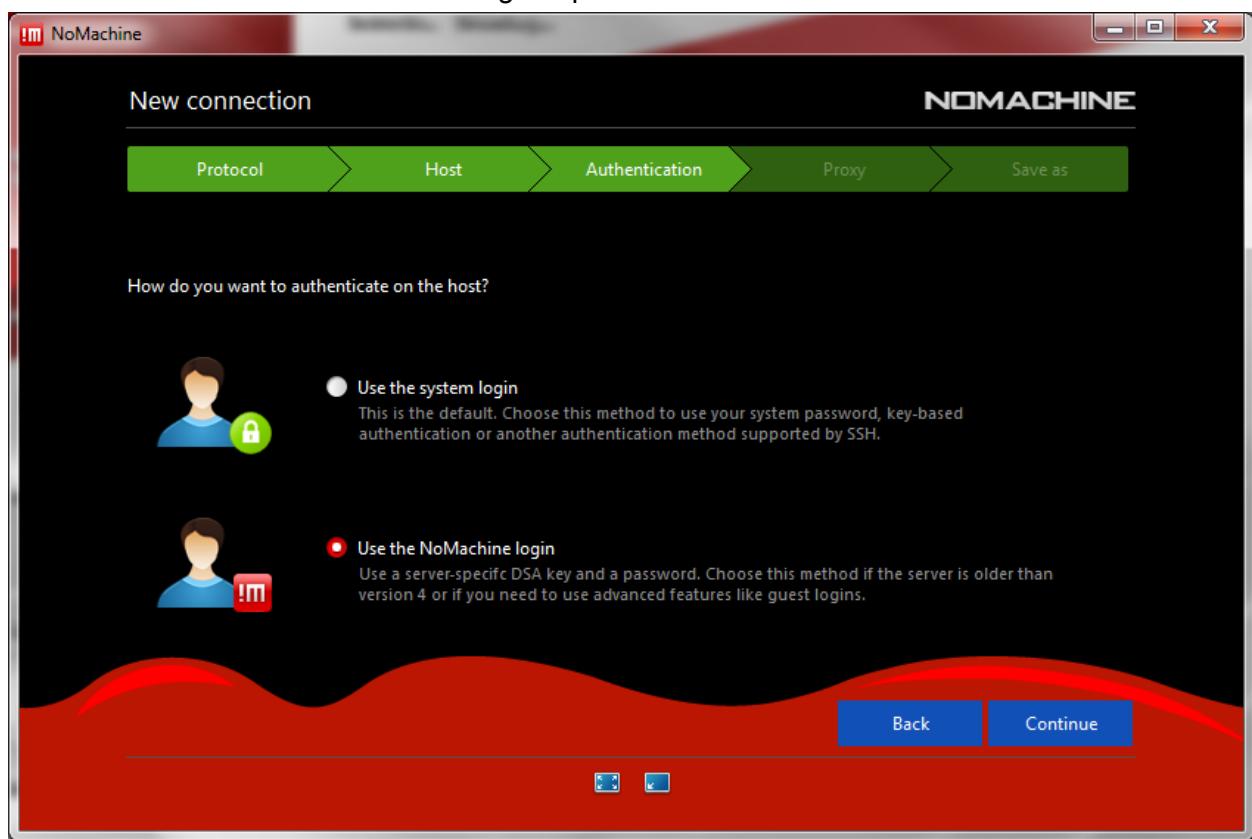
In this example, the 10.150.30.83 IP address is used.  
Accept 22 for the port number.



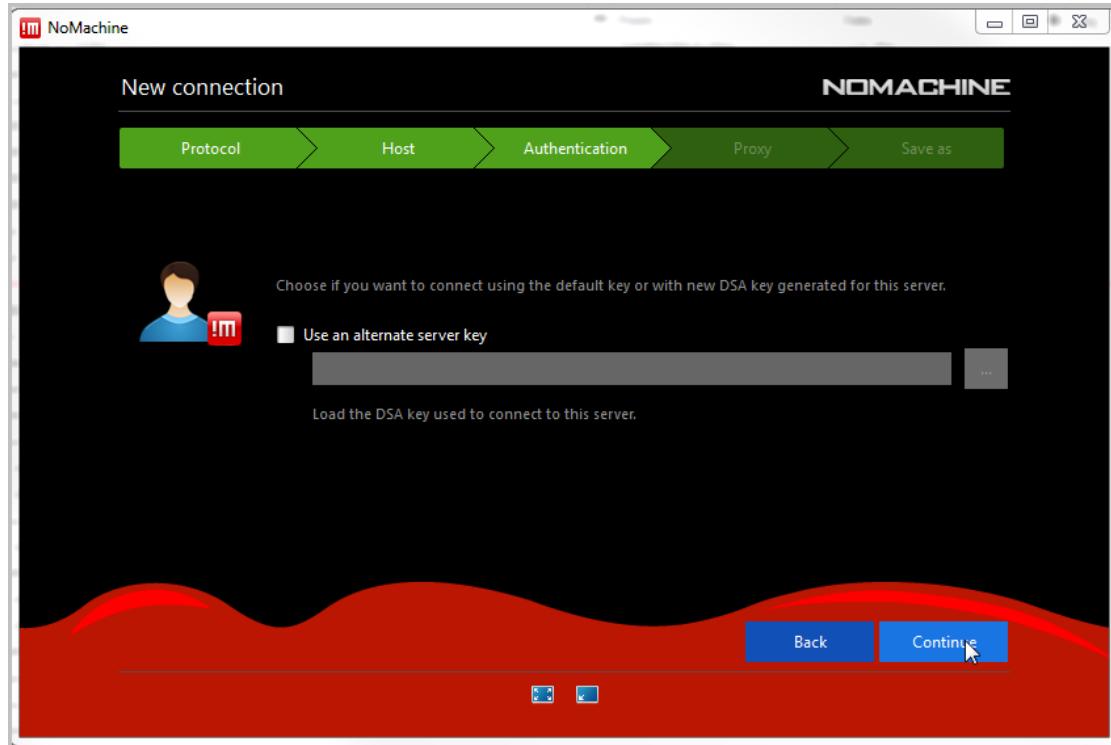
g. Click Continue.

The Authentication window appears.

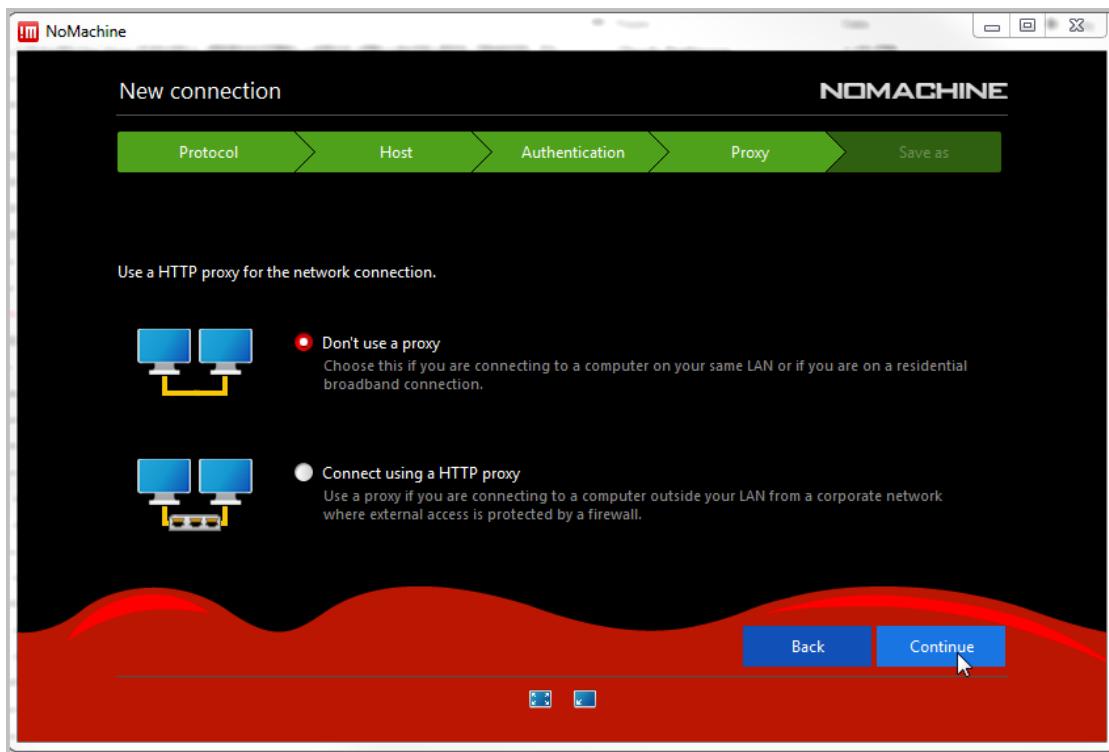
- h. Select the “Use the NoMachine login” option.



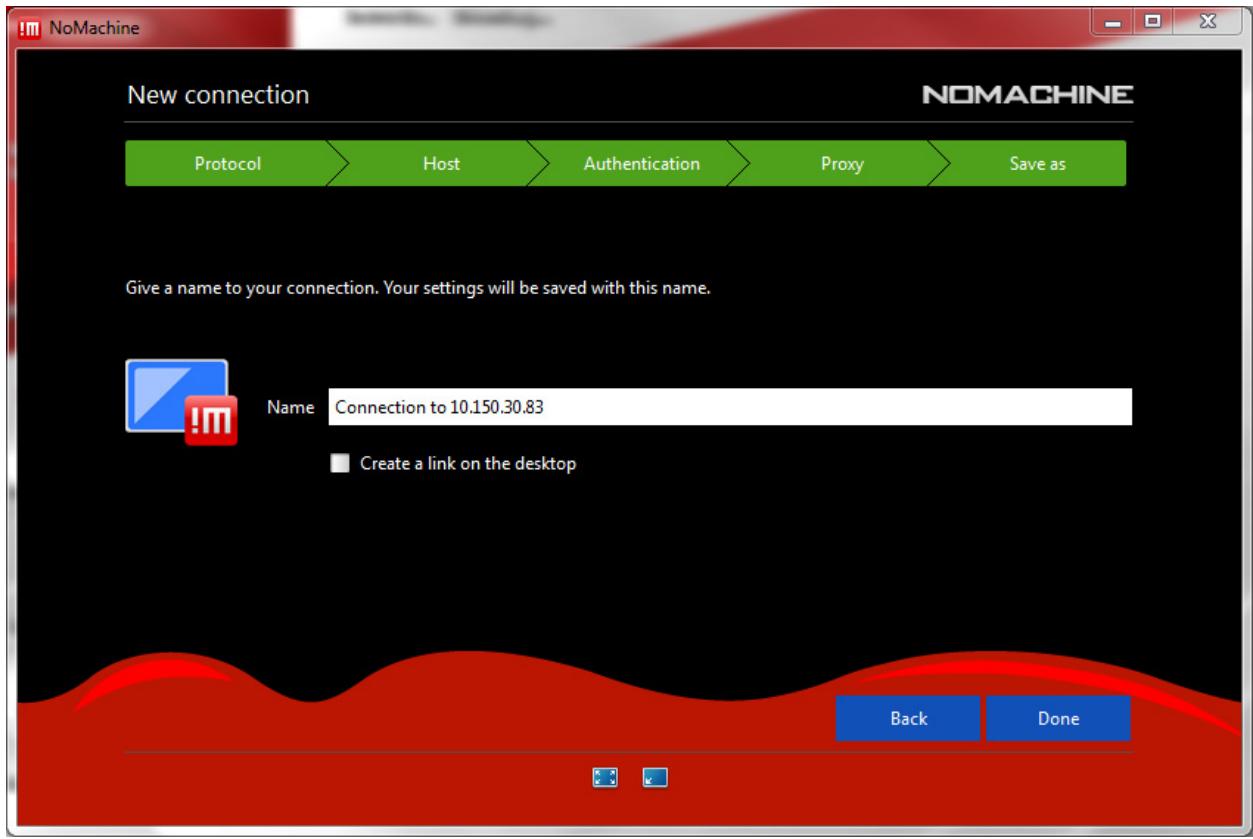
- i. Click Continue.  
j. Leave the “Use an alternate server key” check box deselected and click Continue.



- k. In the Proxy window, leave the “Don’t use a proxy” option selected and click Continue.

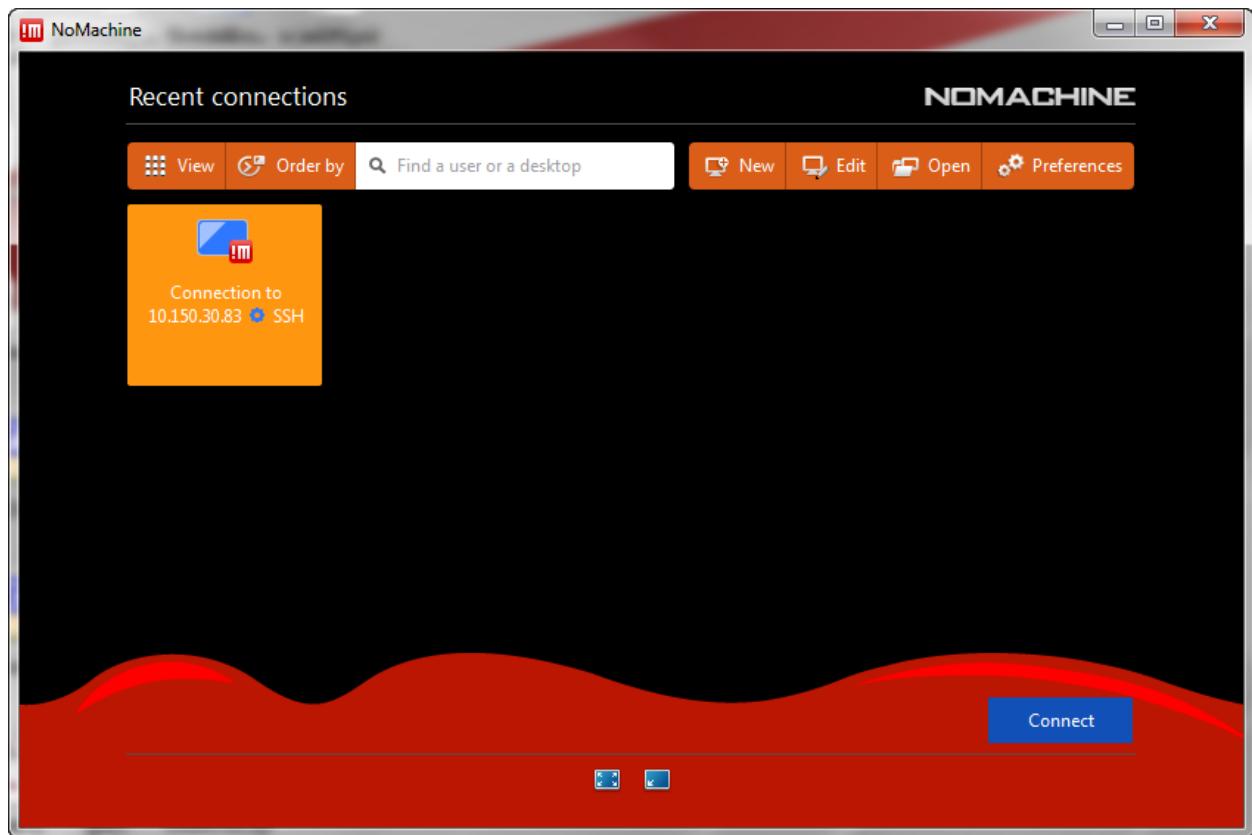


- l. In the “Save as” window, enter a name for your connection or accept the proposed name.

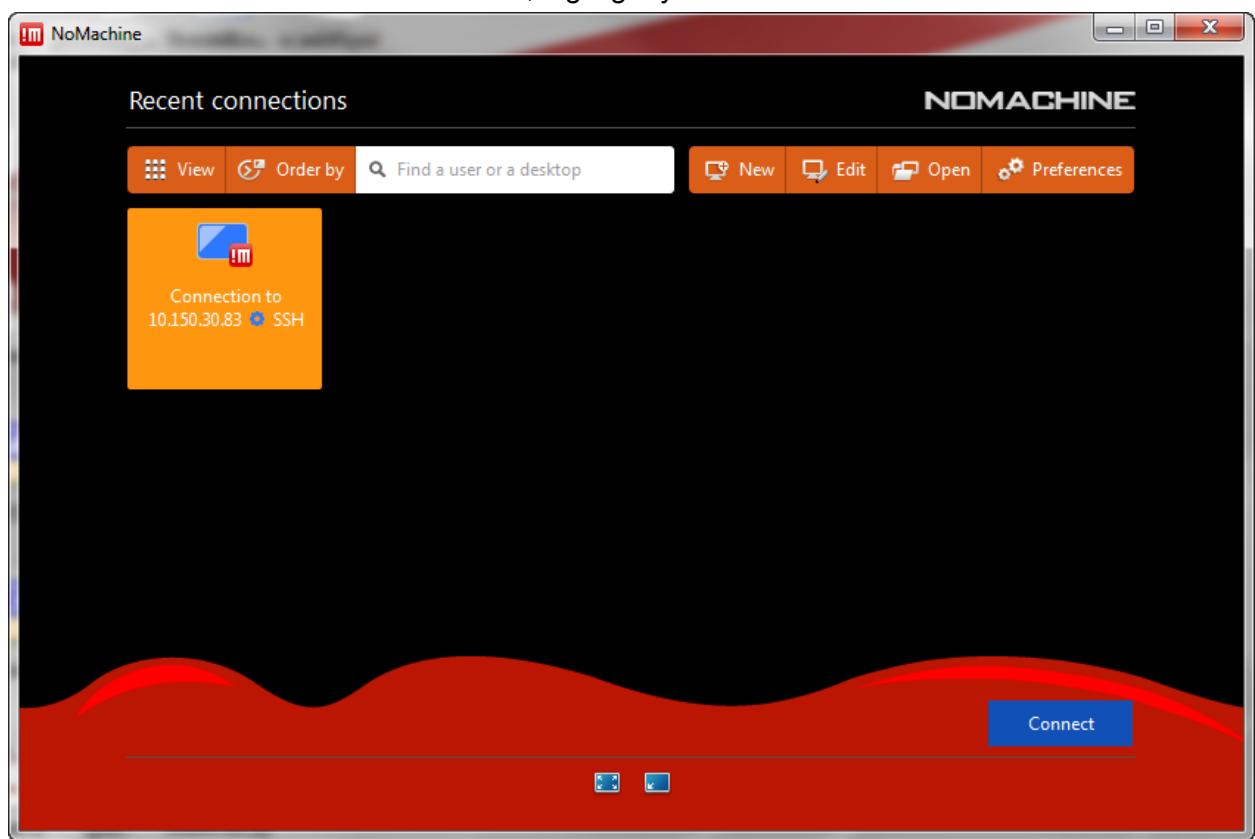


m. Click Done.

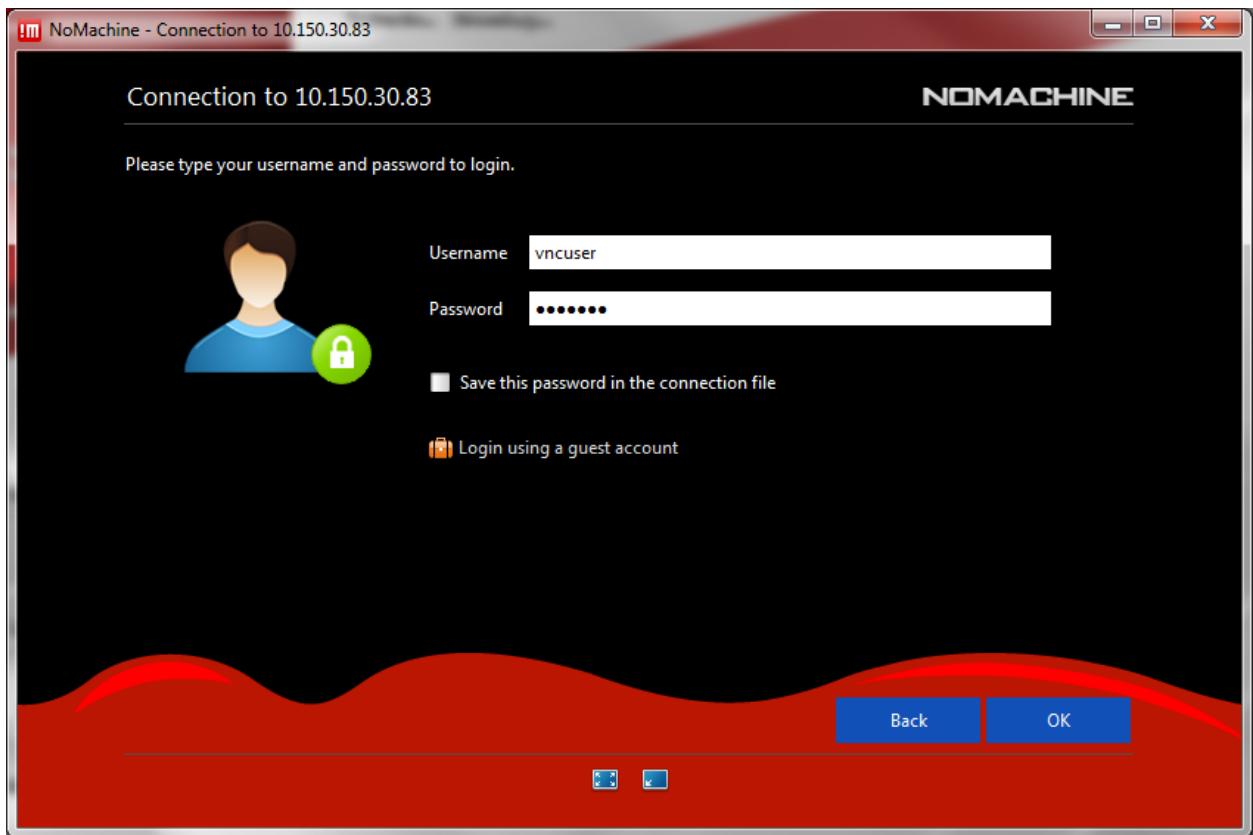
Your new connection appears in the Recent connections window.



2. Access your lab machine by using your newly created connection.
  - a. In the “Recent connections” window, highlight your new connection and click Connect.



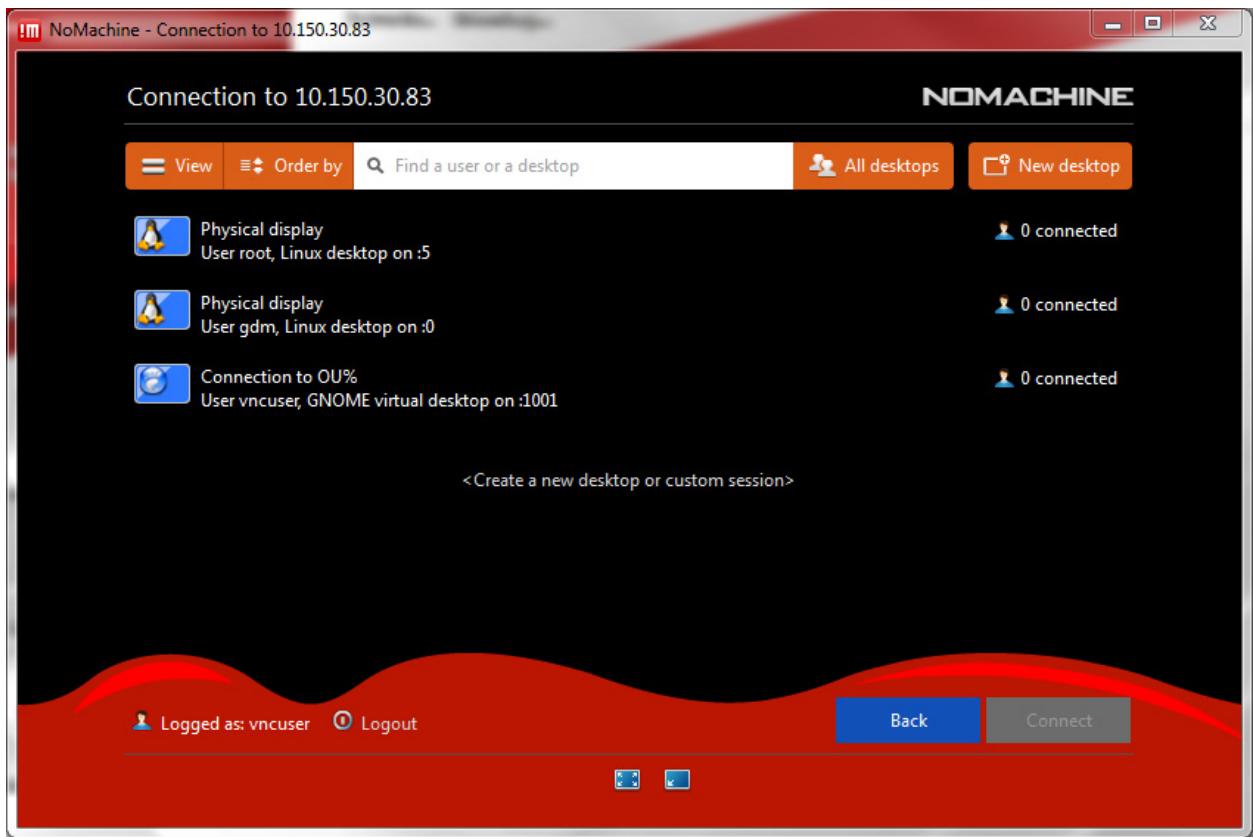
- b. On your new connection login screen, enter the login credentials provided by your instructor.



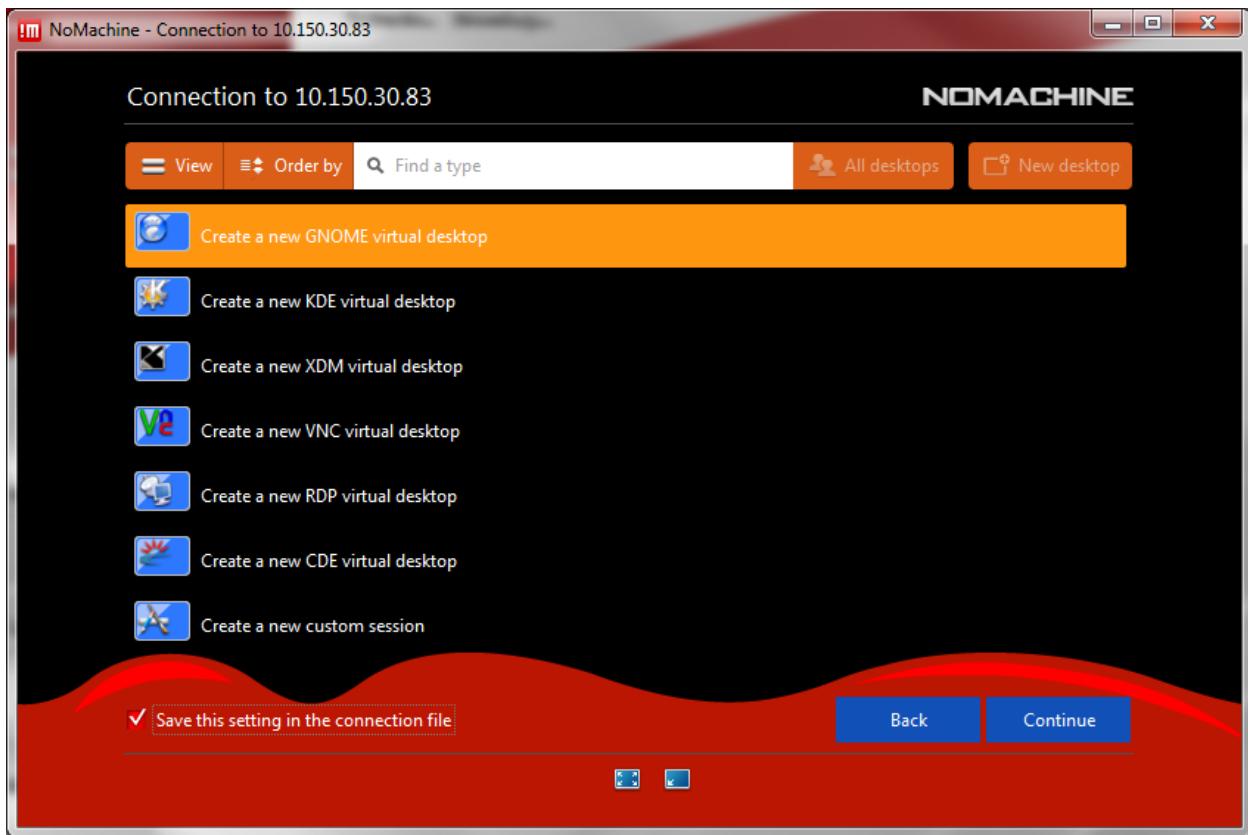
Optionally, you can select the “Save this password in the configuration file” check box.

- c. Click OK.

- d. In the next window, click the “New desktop” session link located in the top-right area of the screen.



- e. Select “Create a new GNOME virtual desktop” and select the “Save this setting in the configuration file” check box.

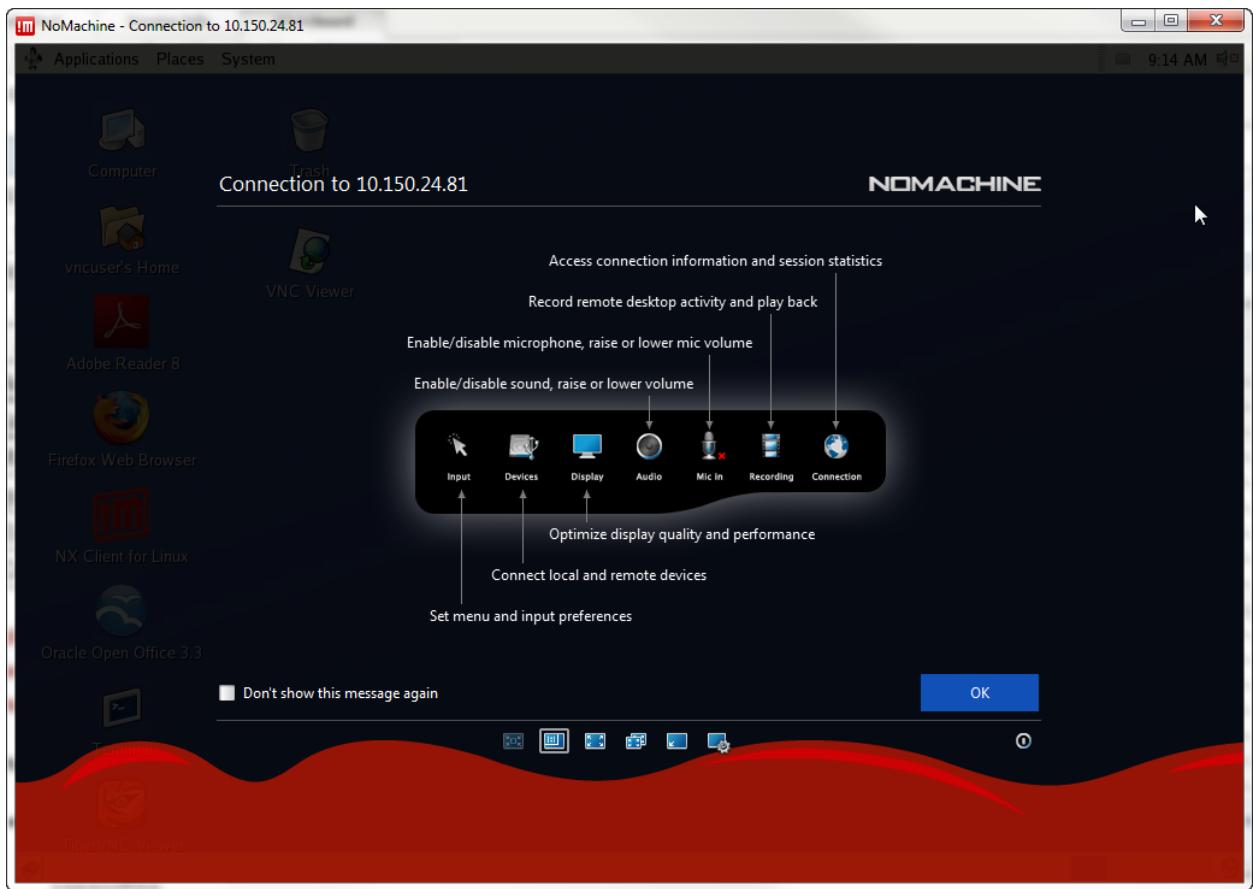


- f. Click Continue.

- g. Click the second window icon (highlighted in red). This allows you to resize your window.

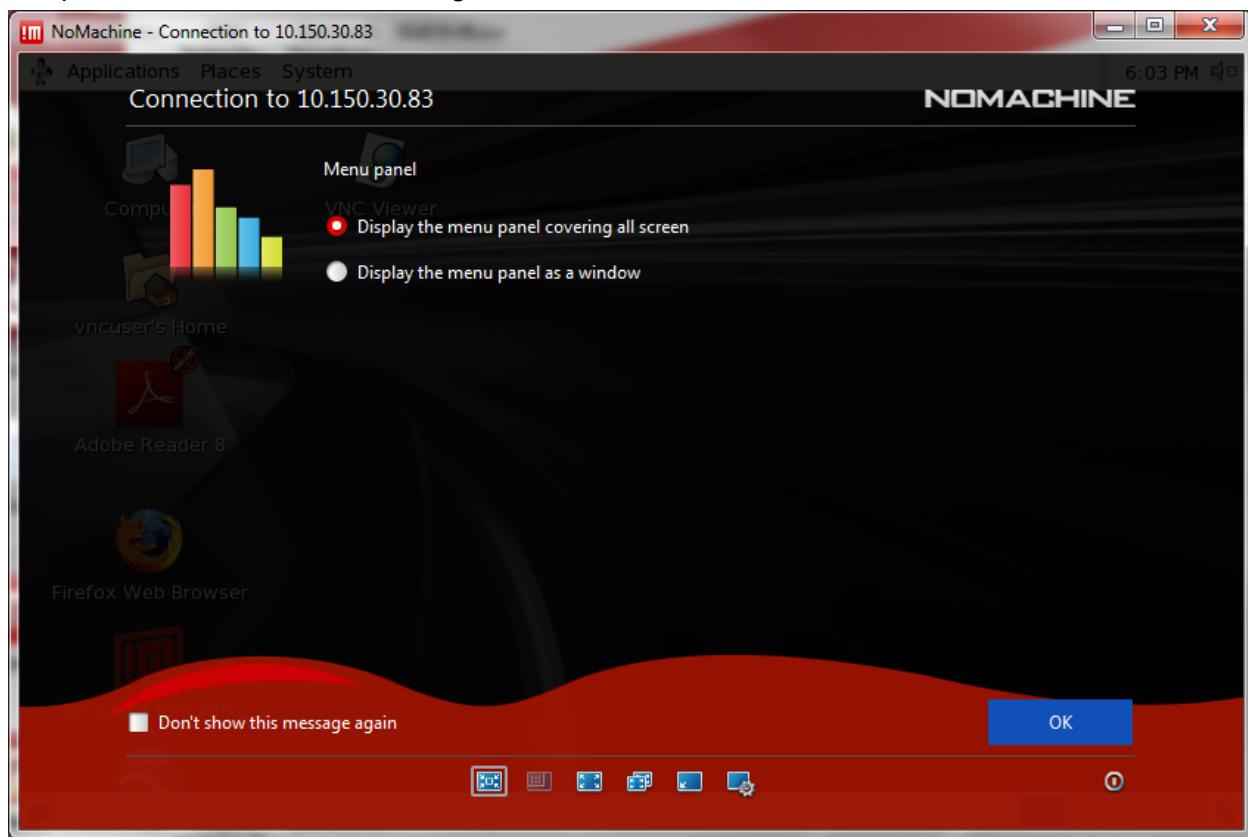


- h. Optionally, you can select the “Don’t show this message again” check box on both screens. Click OK.

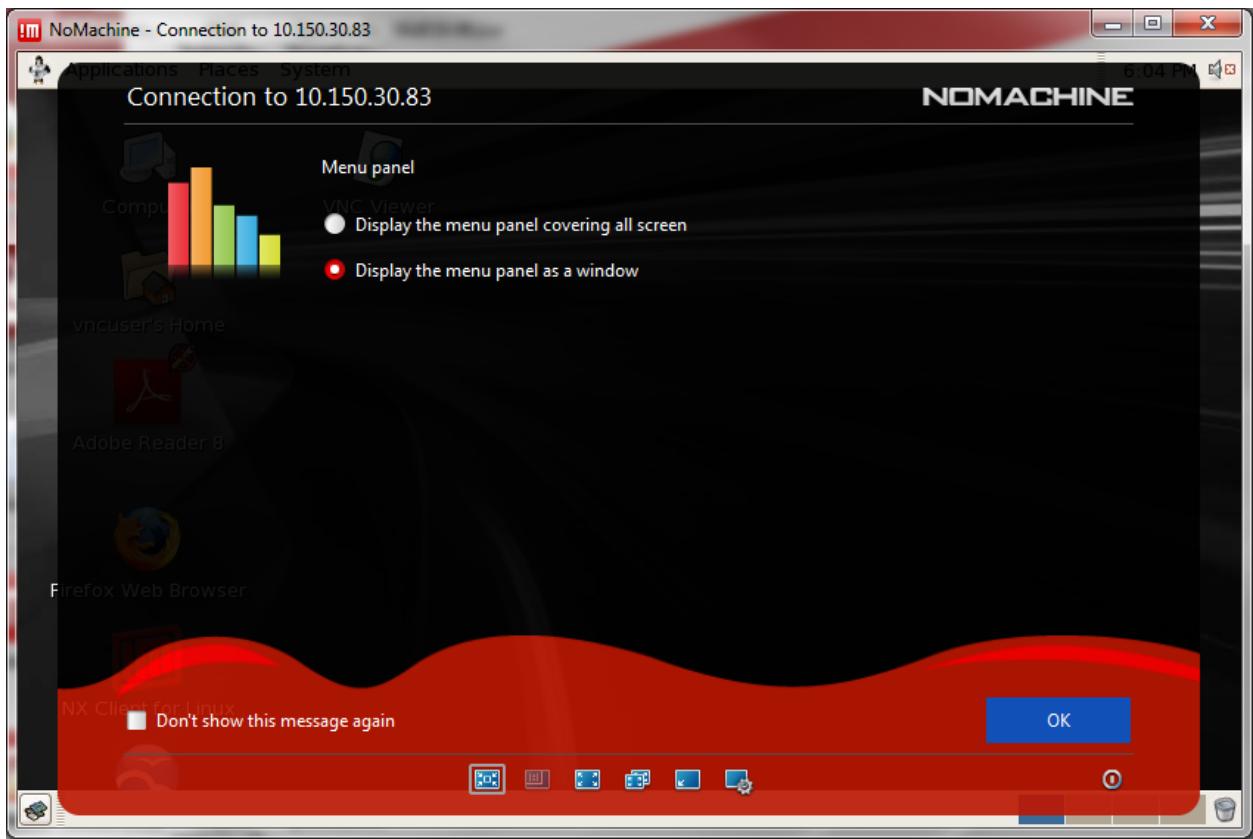


- i. Click OK to dismiss the next window that provides tips for navigation.

- j. The Menu panel options window appears. You can accept the default for the Menu panel as shown in the following screenshot:



- k. Alternatively, you can select the “Display the menu panel as a window” option to display the Menu panel as a centered window.



- I. Click OK.

You are now connected to your lab machine.

