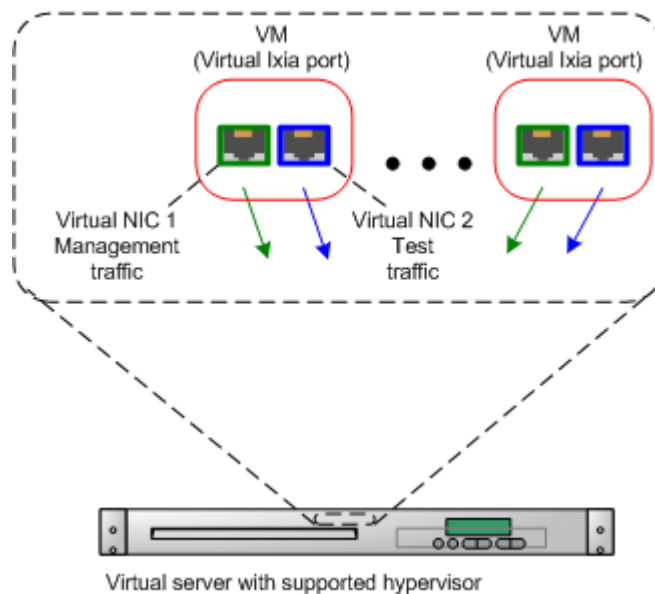


39

IxVM

IxVM is a software-based test platform that enables you to turn standard Linux Ethernet ports into virtual Ixia ports. IxVM can create virtual Ixia ports from the virtual Ethernet ports on a Linux virtual machine (VM), or from the physical Ethernet ports on a physical Linux server.



To configure the traffic generated by the virtual Ixia ports, you use compatible versions of Ixia applications such as IxExplorer, IxNetwork, and IxLoad. When you use these applications, working with a virtual Ixia port is the same as working with a real chassis, with only a few minor differences.

IxVM offers the following benefits:

- Low hardware cost - you can use low-cost Linux servers or dedicated virtualization servers to generate traffic.
- More efficient use of hardware - the same Linux servers used to generate Ixia traffic can also be used for other non-Ixia applications, or the virtual Ixia ports can be hosted on a virtualization server used to host other applications.
- Choice of deployment models - IxVM components are supplied as pre-configured .ova templates or as standalone RPM packages for quick deployment.
- Rapid deployment - Virtual Ixia ports can be instantiated as necessary, used to generate traffic, and then destroyed when no longer needed, releasing system resources for other uses.
- Ease of Use - IxVM-aware Ixia applications are nearly identical to the standard versions, reducing learning time.
- Reduced System Administration - because the IxVM chassis is virtual, it does not have to be housed in a lab or monitored.

In this section:

- [Features](#)
- [Requirements](#)
- [Licensing](#)

Features

IxVM main features include:

- Support for VMware vSphere, KVM or Xen hypervisors.
- Ixia-enhanced kernel OVA template for full functional routing and switching testing.
- Reduced-footprint rpm-based installers for KVM or Xen and bare-metal Linux deployments.
- 32 bit Ixia Kernel self-extracting images for KVM or Xen hypervisors.
- Discovery service that finds virtual Ixia ports and adds them to list of available ports in the test application.
- Deployment of IxVM software upgrades using Ixia Deployment Wizard.
- Jumbo frame generation for high-throughput testing.
- Support for many IxNetwork and IxLoad protocols.

Requirements

IxVM requires the following:

Hardware

IxVM Server: IxVM server requires Windows 7 32-bit.

ESX(i) deployments: IxVM runs on any virtualization server that ESX(i) supports.

KVM, Xen or other deployments: For KVM or Xen and bare-metal deployments, Ixia recommends a high-performance server with CPUs that include virtualization extensions such as Intel-VT or AMD-V. A high-end application server is available from Ixia - contact your Ixia sales representative for more information.

Hypervisor / Host OS

IxVM supports the following hypervisors or host OSes:

- VMware ESXi 5.0 or ESXi 5.1 for vSphere deployments (OVA deployments)
- KVM (QEMU) over CentOS 6.3 64-bit (KVM self-extracting deployments)
- Xen over CentOS 5.6 64-bit (Xen self-extracting deployments)

VM Operating Systems (Guest OS)

IxVM virtual ports can be created on any VM running one of the following operating systems:

- Ixia-enhanced kernel
- RedHat Enterprise Linux 6.3, 32-bit
- CentOS 6.3, 32-bit
- SUSE Linux Enterprise Server 11, 32 bit

Distribution Methods

IxVM is distributed using a variety of methods. The table below lists how IxVM is distributed for supported combinations of hypervisor and guest OS.

Hypervisor or Host OS	Guest OS Distribution Method	
	Ixia Kernel	RedHat/CentOS/SUSE
VMware ESXi 5.0	OVA/RPM	RPM
VMware ESXi 5.1	OVA/RPM	RPM
KVM (QEMU) over CentOS 6.3 64-bit	RPM/Self-extracting image	RPM
Xen over CentOS 5.6 64-bit	RPM/Self-extracting image	RPM

Note: The Ixia Kernel rpm should only be used for upgrading to newer versions of IxVM using the Deployment Wizard.

VM Discovery

Discovery Server discovers virtual Ixia ports, and adds them to the list of available ports in Ixia testing applications. You can download Discovery Server from the IxVM page of Ixia's website.

Update Utility

Deployment Wizard updates the IxVM platform with new versions of the IxVM software. You can download Deployment Wizard from the IxVM page of Ixia's website.

Ixia Application Software

The following applications are supported on the IxVM platform:

- IxNetwork
- IxExplorer
- IxLoad

You can download all three from Ixia's website.

The following tables lists the Ixia applications you can run on each combination of hypervisor and guest OS:

Hypervisor or Host OS	Guest OS
	Ixia Kernel
VMware ESXi 5.0	IxExplorer
	IxNetwork
	IxLoad
VMware ESXi 5.1	IxExplorer
	IxNetwork
	IxLoad
KVM (QEMU) over CentOS 6.3 64-bit	IxExplorer
	IxNetwork
Xen over CentOS 5.6 64-bit	IxExplorer
	IxNetwork

Hypervisor or Host OS	Guest OS
	RedHat/CentOS 6.3 32-bit
VMware ESXi 5.0	IxExplorer
	IxNetwork
VMware ESXi 5.1	IxExplorer
	IxNetwork
KVM (QEMU) over CentOS 6.3 64-bit	IxExplorer
	IxNetwork
Xen over CentOS 5.6 64-bit	IxExplorer
	IxNetwork

Hypervisor or Host OS	Guest OS
	SUSE LINUX ES 11 32-bit
VMware ESXi 5.0	IxExplorer
	IxNetwork
VMware ESXi 5.1	IxExplorer
	IxNetwork
KVM (QEMU) over CentOS 6.3 64-bit	IxExplorer
	IxNetwork
Xen over CentOS 5.6 64-bit	IxExplorer
	IxNetwork
CentOS 5.6 64-bit on bare metal	

About the Red Hat and CentOS Kernels

RedHat Enterprise Linux 6.x and CentOS 6.x are based on the same kernel. RedHat Enterprise Linux requires a subscription, in return for which Red Hat Inc. provides support and other services. CentOS is the Community Enterprise edition, and is free. The same RPM packages can be compiled and installed on both RedHat and CentOS.

Licensing

The following are the licensing requirements of IxVM and its related Ixia components:

Licensed:

- IxVM Server requires a license.
- IxLoad, IxNetwork, and IxNetwork-FT require licenses.

Note: All IxVM-enabled test applications such as IxVM Server, IxExplorer, IxNetwork/IxNetwork-FT and IxLoad use floating licenses, meaning that a specified license server stores a number of licenses that can be re-used alternatively by a number of installed test applications.

Each time a test application starts, one license is used from those available on the licensing server. This allocation process continues as more applications request licenses, until the pool of remaining licenses is depleted.

For each of the test applications, you specify the licensing server in the following locations:

- For IxVM Server, specify the server host on the TOOLS > OPTIONS > GENERAL tab.
- For IxNetwork-FT, specify the server host on the IxExplorer TOOLS > OPTIONS > LICENSE MANAGEMENT tab.
- For IxNetwork, specify the server host on the FILE > TOOLS > LICENSING tab.
- For legacy Network-FT TCL scripts, follow the instructions for setting the license server through IxExplorer / Network-FT. Once you set the Explorer / Network-FT license server, you can run your TCL scripts.
- For IxLoad, specify the server host by clicking OPTIONS > LICENSE SERVER on the toolbar.

If you change the license server location in any of the test applications, you must restart the application for the new location to take effect.

The license server host must have the IxProxy service running on it to traverse any firewalls between the license server host and the hosts running the test applications. IxProxy is supplied with IxVM Server and must be manually started using the SERVICES - ADMINISTRATIVE tool of Windows.

Not Licensed:

- IxExplorer does not require a license.
- Analyzer does not require a license.
- Deployment Wizard does not require a license

Note: The new Subscription Licensing model for IxVM ports, will impose to have installed on the license server at least the same number of licenses as the number of ports from the virtual chassis.

For example, if on the virtual chassis there are 10 ports, then on the license server there should be at least 10 Tier-3 licenses.

The Ixia licensing process is described in the *Ixia Licensing Guide*.

IxVM Deployment Models

There are three general ways that you can deploy IxVM:

- **vSphere:** In a vSphere deployment, you use vSphere to create VMs that are based on the Ixia kernel OVA template or on one of the supported Linux distributions.
- **KVM or Xen:** In a KVM or Xen deployment, you install IxVM disk images on a KVM or Xen server, and the supporting software on either a Windows VM or Windows computer. You can also use the KVM or Xen self-extracting image to create Ixia Kernel VMs.
- **Bare metal:** In a bare metal deployment, you install IxVM RPM packages on a bare metal Linux server, and the supporting software on a Windows computer.

This section describes each method.

In this section:

- [*vSphere Deployment*](#)
- [*KVM Deployment*](#)
- [*XEN Deployment*](#)
- [*XEN Deployment*](#)
- [*Windows components*](#)

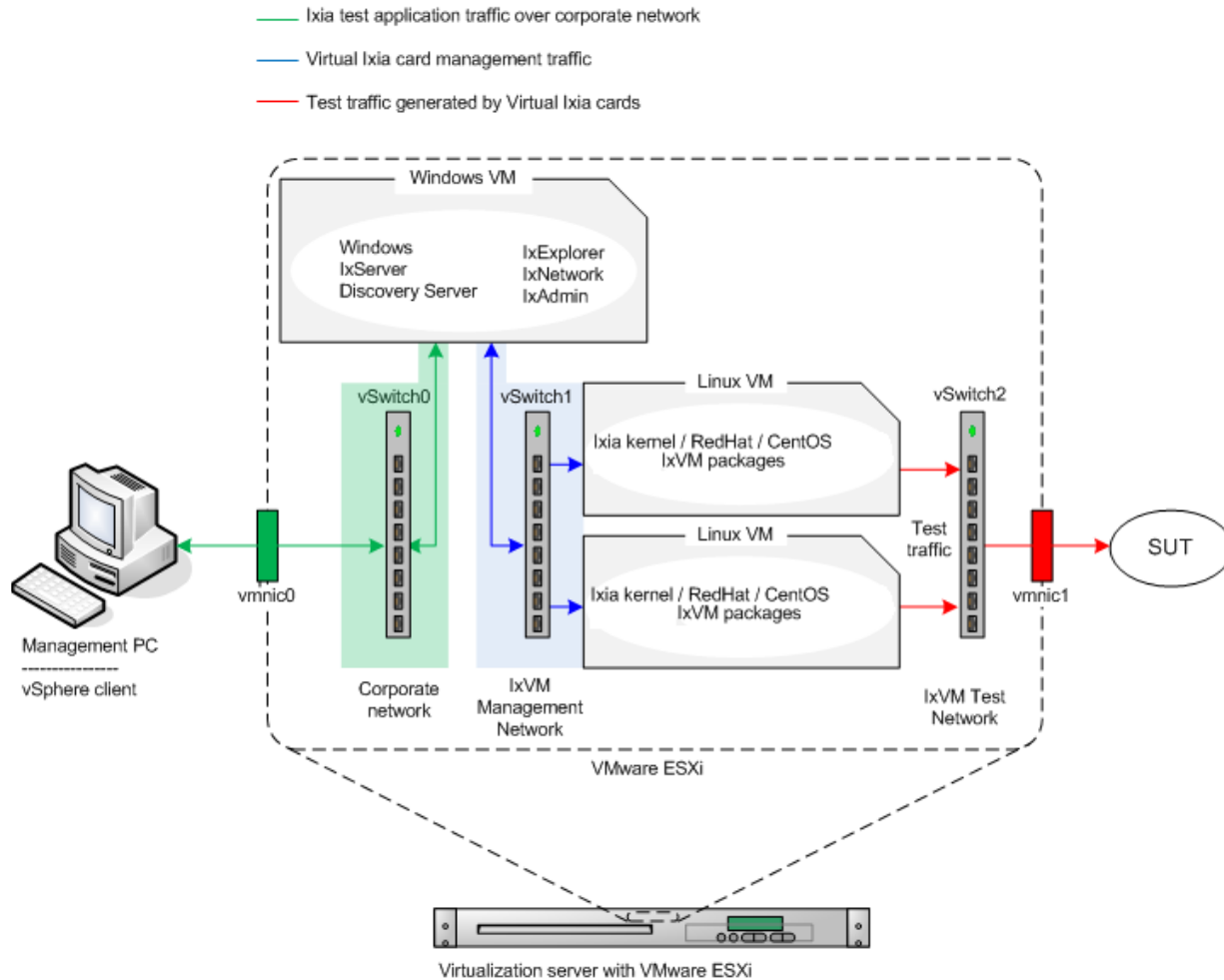
vSphere Deployment

In an vSphere deployment, you use vSphere to create VMs based on the Ixia kernel template file (.ova).

In the figure below, the following components are present:

- **Virtualization server with VMware ESX(i):** The virtualization server is the physical machine that hosts the ESX(i) virtualization operating system.
- **Windows VM:** The Windows VM is a virtual Windows computer hosted on the virtualization server, and functions as the IxVM chassis controller. It hosts the following applications:
 - **IxServer,** which manages traffic between Ixia applications and the virtual Ixia ports.
 - **Discovery Server,** which discovers virtual Ixia ports running on the virtualization server, and adds them to the lists of test ports in Ixia testing applications.
 - **IxNetwork, IxExplorer, and IxLoad,** the Ixia testing applications.
- **Linux VMs:** The Linux VMs are VMs hosted on the virtualization server that are running one of the supported Linux kernels and the IxVM packages that generate the test traffic. Each VM has two NICs: one for management traffic, and one for test traffic.

- Virtual switches: The vSwitches are virtual equivalents of physical switches and perform the same function, routing traffic to and from the virtual Ixia ports. The virtual switches are created using vSphere.
- Windows computer: The Windows computer is used to access and control the VMs on the ESX(i) host using vSphere.



KVM Deployment

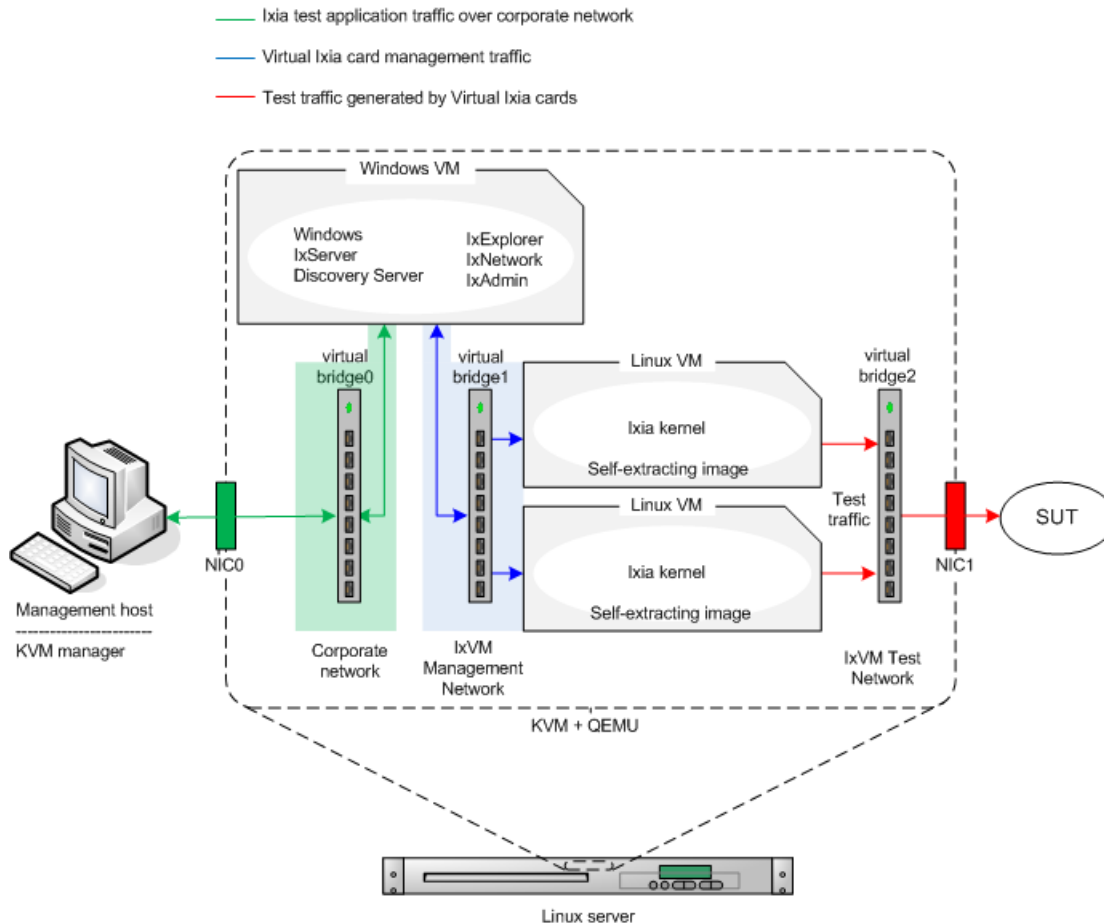
In a KVM deployment, the virtual Ixia ports are created on virtual machines running under the KVM hypervisor on a KVM server. The figure below shows the components used in a KVM deployment.

In the figure below, the following components are present:

- KVM server: The KVM server is the physical machine that hosts the KVM hypervisor.

- **Windows VM:** The Windows VM is a virtual Windows computer hosted on the KVM server, and functions as the IxVM chassis controller. It hosts the following applications:
 - **IxServer,** which manages traffic between Ixia applications and the virtual Ixia ports.
 - **Discovery Server,** which discovers virtual Ixia ports running on the virtualization server, and adds them to the lists of test ports in Ixia testing applications.
 - **IxNetwork and IxExplorer,** the Ixia testing applications.

Alternatively, a physical computer can also be used as the IxVM controller.
- **Linux VMs:** The Linux VMs are VMs hosted on the virtualization server that are running one of the supported Linux kernels and the IxVM packages that generate the test traffic. Each VM has two NICs: one for management traffic, and one for test traffic.
- **Virtual switches:** The virtual switches are virtual equivalents of physical switches and perform the same function, routing traffic to and from the virtual Ixia ports. The virtual switches are created using KVM.
- **Management host:** The management host is used to manage the VMs on the KVM host using a KVM management utility.



XEN Deployment

In a XEN deployment, the virtual Ixia ports are created on virtual machines running under the XEN hypervisor on a XEN server. The figure below shows the components used in a XEN deployment.

In the figure below, the following components are present:

- XEN server: The XEN server is the physical machine that hosts the XEN hypervisor.
- Windows VM: The Windows VM is a virtual Windows computer hosted on the XEN server, and functions as the IxVM chassis controller. It hosts the following applications:
 - IxServer, which manages traffic between Ixia applications and the virtual Ixia ports.
 - Discovery Server, which discovers virtual Ixia ports running on the virtualization server, and adds them to the lists of test ports in Ixia testing applications.
 - IxNetwork and IxExplorer, the Ixia testing applications.

Alternatively, a physical computer can also be used as the IxVM controller.

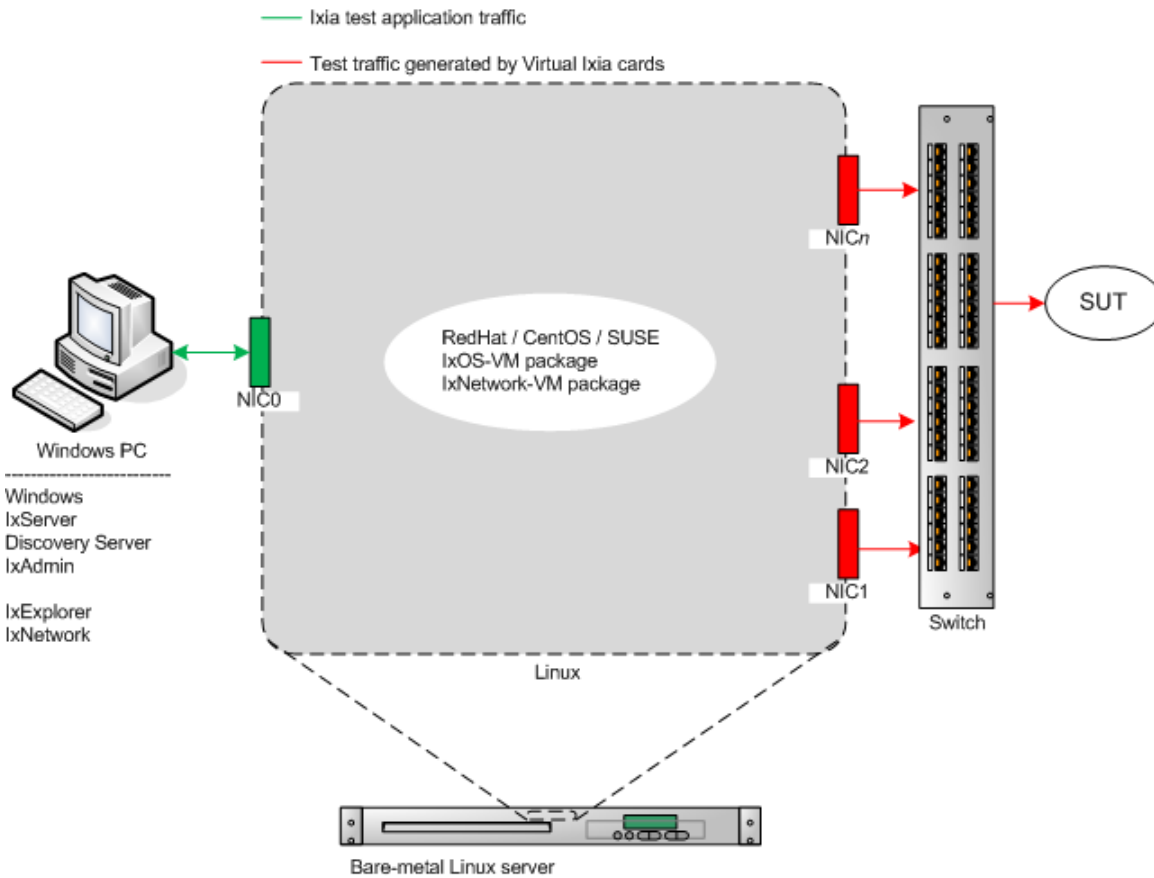
- **Linux VMs:** The Linux VMs are VMs hosted on the virtualization server that are running one of the supported Linux kernels and the IxVM packages that generate the test traffic. Each VM has two NICs: one for management traffic, and one for test traffic.
- **Virtual switches:** The virtual switches are virtual equivalents of physical switches and perform the same function, routing traffic to and from the virtual Ixia ports. The virtual switches are created using XEN.
- **Management host:** The management host is used to manage the VMs on the XEN host using a XEN management utility.

Bare-metal Linux Deployment

Although deploying IxVM as a series of VMs is the most typical scenario for using IxVM, it can also be hosted on a bare-metal Linux server. The figure below shows the components used in a bare-metal deployment of IxVM.

In the figure below, the following components are present:

- **Linux server:** The Linux server runs one of the Linux kernels supported by IxVM, and the supporting IxVM packages that generate the test traffic.
- **Windows computer:** The Windows computer hosts the following applications:
 - **IxServer,** which manages traffic between Ixia applications and the virtual Ixia ports.
 - **Discovery Server,** which discovers virtual Ixia ports running on the virtualization server, and adds them to the lists of test ports in Ixia testing applications.
 - **IxExplorer and IxNetwork,** the Ixia applications used to configure and run tests, and that use the virtual Ixia ports to generate traffic.
- **Switch:** The switch routes the test traffic to and from the SUT.



Windows components

The Windows components required to manage and use IxVM virtual cards can be installed in any of the following places:

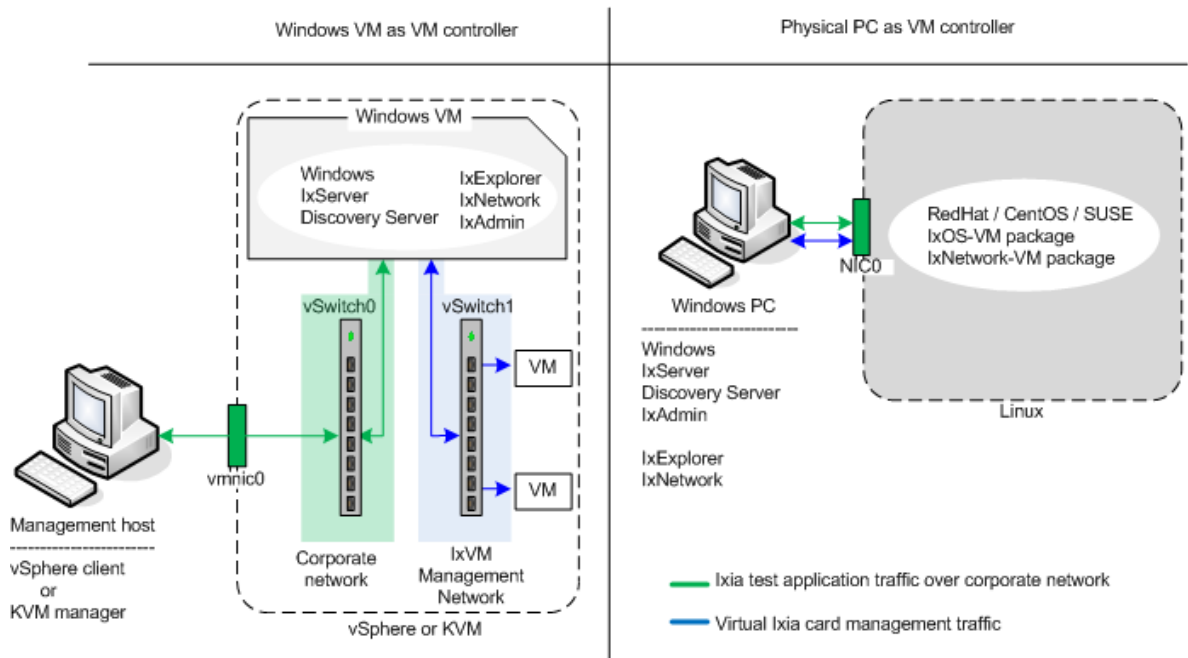
- On a Windows VM
If you have a vSphere, KVM or Xen environment, you can create a Windows VM (or use an existing one) and then install the IxVM components on it.
- On a physical Windows computer
In vSphere, KVM or Xen deployments, you can also use a physical computer as the IxVM controller. If you are deploying on bare metal, a physical computer is the only option for the IxVM controller.

Note: You cannot install the IxVM component on an Ixia chassis.

Installing IxVM Components on a Physical computer or Windows VM

To use a physical computer or Windows VM as the IxVM controller, install the following applications on the computer or Windows VM, making sure to select the options for IxVM support when you install them:

- IxOS
- Discovery Server
- Deployment Wizard
- IxExplorer



TCP/UDP Ports Required

The following ports are needed for communication between Windows VM and Linux VMs:

TCP

- 998 betaftpd_shadow
- 999 inetd
- 1000 ixdiscoveryagent
- 6001 ixServiceManager
- 6665 InterfaceManager
- 9101 ixStatDaemon
- 9102 ixStatDaemon
- 9613 ixDodClient
- 9614 pcpuManagerar
- 10116 ixdiscoveryagent

UDP

- 123 ntpd
- 1000 ixdiscoveryagent
- 10116 ixdiscoveryagent

Windows VM ports

Windows VM ports are as follows:

- **TCP ports:** 1000, 1080, 5285, 6001, 6005, 8021, 9101, 9102, 9613, 17668, 17669, 17670, 17672, 33000, 57843, 57845, 57846, 57847, 61248, 61260, 61272, 61274
- **UDP ports:** 1000, 65031

Installing IxVM

The installation process for IxVM depends on the deployment type you are using:

- **vSphere:** In a vSphere deployment, you can install IxVM by creating VMs in vSphere and basing them on the Ixia kernel OVA template. The OVA template includes all the supporting software required for IxVM pre-installed.
- **KVM or Xen:** In a KVM or Xen deployment, you use self-extracting disk images to create VMs that have all the IxVM packages already installed, or you use RPMs to install the IxVM packages on VMs that you have created yourself.
- **Bare-metal:** In a bare-metal deployment, you use RPM to install the IxVM packages on a bare-metal server that is running one of the supported Linux distros.

Note: On Xen hypervisors, installing IxVM on a VM created without the self-extracting script file is not supported.

The following sections describe each method:

- [OVA-based Installation](#)
- [KVM Installation](#)
- [Bare-metal \(RPM\) Installation](#)

OVA-based Installation

The sequence for deploying IxVM on vSphere is as follows:

1. Download and install VMware ESX(i) on your VMware server (if you have not already done so).
2. If you intend to use a VM as the IxVM chassis controller, create a Windows VM in VMware, and install IxServer and the supporting applications on it.
3. Download the Ixia kernel template (.ova) file, and use the template to create the one or more Linux VMs to host the virtual Ixia ports
4. Configure Discovery Server.

The following sections describe each of these steps.

After you have completed the steps, you can begin using the virtual Ixia ports with your Ixia testing application.

Downloading and Installing vSphere

If you have not already installed VMware's ESX(i), you must install it on your server. ESX(i) is available free and has the same user interface as the enterprise versions of vSphere and vCenter Server.

Note: ESX(i) is a Type 1 hypervisor -- a virtualization operating system. It is not a virtualization application that you can run on top of another operating system. If you install ESX(i), it replaces the existing operating system on your server

A video is available showing the ESX(i) installation process: <http://www.youtube.com/watch?v=3UjSd8rbJvQ> (<http://www.youtube.com/watch?v=3UjSd8rbJvQ>). Note the steps at the end of the video for configuring the password and IP address of the server.

1. Access the VMware web site and register for a free account: <https://www.vmware.com/tryvmware/index.php?p=free-esxi&lp=1> (<https://www.vmware.com/tryvmware/index.php?p=free-esxi&lp=1>)
2. Download the VMware ESX(i) ISO image, and then burn it to a CD-ROM. Make sure that you save the ESX(i) license key.
3. Power on your server and enter the BIOS setup utility.
4. Enable all the virtualization features in the BIOS.
5. In the BIOS, make sure that the boot order shows the CD-ROM first and the internal hard disk second.
6. Insert the CD-ROM into your server's CD-ROM drive, save the changes to the BIOS and then reboot the server.
7. Install ESX(i) on your server.
8. On the computer that you will use to access the server, open a web browser and enter the server's IP address in the URL field.
9. Install vSphere client on the computer.

Creating the Source and Destination Networks

Before you create and deploy the VMs, you should already have created the virtual networks that they will use. To create the virtual networks, you use vSphere client. You will need two networks:

On the Windows VM (virtual chassis):

- An access network that enables you to access the virtual chassis. This network connects to your corporate LAN. The default network created for this purpose in vSphere is named "VM Network".
- A card management network to act as "virtual backplane" that connects to the IxVM cards.

Two networks are pre-configured on the Ixia kernel OVA:

- IxVM Management Network, the virtual backplane.
- IxVM Test Network 1, the path for test traffic to the DUT or SUT.

During the process of creating the VMs, you map the source networks configured in the .ova template to the destination networks on your VMware server.

To create the networks, use the following procedure:

1. Open vSphere client, and display the Inventory tab.
2. Click the Configuration tab.
3. In the Hardware pane, click Networking.
4. Click Add Networking.

The Add Networking wizard starts, and displays the Connection Type window.

5. Click Virtual Machine, then click Next.

The Network Access window displays

6. Click Create a Virtual Switch.

The Connection Settings window displays.

7. In the Network Label field type a name, then click Next.

8. Repeat for the additional vSwitches (networks).

Deploying the IxVM Appliances (OVAs)

The IxVM Ixia kernel is available as an Open Virtual Appliance (.ova) file, which is a template for a virtual machine. The .ova template creates a VM with a Linux kernel that has been modified by Ixia for greater performance in some testing scenarios. To use the .ova file, you use vSphere client to create a VM, and specify the .ova file as the template for the VM.

Note: Before you create and deploy the VMs, you must already have created the virtual networks that they will use in vSphere. See [Creating the Source and Destination Networks](#) on page 39-16.

To create a VM based on an IxVM OVA file:

1. Download the .ova file from the Ixia website. Store them in a location where they can be accessed from the ESX(i) server.
2. From the vSphere client menubar, choose FILE | DEPLOY OVF TEMPLATE.

The Source window displays.

3. Choose DEPLOY FROM FILE:, then click BROWSE, and select the .ova file. Click NEXT.

The OVF Template Details window displays.

4. Make a note of the Username and Password shown on the OVF Template Details window. The default usernames and passwords are:

root/ixia123

Click NEXT.

The End User License window displays.

5. Click ACCEPT to accept the license, then click NEXT.

The Name and Location window displays.

6. Enter a unique name for the VM, then click NEXT.

The Datastore window displays.

7. Select the data store where you want your VM files to be stored, then click NEXT.

The Network Mapping window displays. On this window, you map the networks configured in the .ova template to the networks on your virtualization server.

8. Map the template's networks to your VMware networks, then click NEXT.

The Ready to Complete window displays, showing a summary of the deployment options you have selected.

9. Choose one:

- If you are installing using vSphere client with vCenter, the wizard displays windows enabling you to select the IP address allocation method. Choose the method you want to use.
- If you are installing using vSphere client without vCenter, the VM is automatically created with IP address allocation set to DHCP. If you want to use static IP addresses, you must use the VM's console interface to specify the IP addresses of the VM after it boots. See *Configuring static Addresses* (see "Configuring Static Addressing" on page 2).

Review the final deployment details, then click FINISH. Allow the VM several minutes to deploy.

10. Boot the VM.

11. Repeat the deployment procedure for the remaining VMs that you want to create.

Configuring Static Addressing

If you created an IxVM template using vSphere without vCenter, the VM is automatically created with IP address allocation set to DHCP. If you want to use static IP addresses, you must use the VM's console interface to specify the IP addresses of the VM after it boots.

To configure a static address on a VM:

1. In vSphere client, select the VM that you want to configure.
2. Click the CONSOLE tab.

A console session for the VM starts, and prompts you to login.

3. Login to the VM with the user name and password configured on it when it was created.

Note: You can display the user name and password by clicking the Summary tab, then select Annotations, and then click Edit. Scroll through the window until the user name and password displays.

4. Start the vi editor and load the IP address script for the eth0 interface by typing the following command:

```
vi /etc/sysconfig/network-scripts/ifcfg-eth0
```

5. Type I or press the INSERT key to switch to edit mode.
6. Change BOOTPROTO to static.
7. Add a new line that contains the following:

```
IPADDR=<ipaddress>
```

where <ipaddress> is the address you want to assign to VM's eth0 interface.

```
NETMASK=<mask>
```

where <mask> is the netmask to be applied to the IP address.

8. Close and save the file: Press ESC, then type: : W Q ! (colon, w(rite), q(uit), exclamation point).
9. Repeat for the eth1 interface (edit the file ifcfg-eth1).
10. Issue the following commands to bring the interfaces up:

```
ifup eth0
```

```
ifup eth1
```

Configuring Discovery Server

Use the following procedure to configure Discovery Server.

1. Start Discovery Server using the following command:

```
START | ALL PROGRAMS | IXIA | IXIA DISCOVERY SERVER | IXIA  
DISCOVERY SERVER
```

Discovery Server starts, and minimizes to the tray.

2. Double-click the **DISCOVERY SERVER** icon.
3. Click the **BROADCAST DISCOVERY** tab.

The Broadcast Addresses list contains the list of IP addresses configured in the VM. For some installations, this may include the address of the IxVM management port (the first LAN connection), and the second LAN connection. Discovery Server displays the IP addresses for all the network adapters configured on the Windows VM so that you can select the networks that you want to search for IxVM cards.

4. Select the check box for the address of the IxVM card management IP port (typically, 10.0.0.x), and clear the check box for the second LAN connection.

Note: If you restart Discovery Server, the check boxes are reset to their default values.

Optimizing Performance

You may be able to improve performance by applying the following optimizations:

Power Management

In the ESX(i) server BIOS, ensure that Power Management is set to High Performance.

Hyperthreading

In vSphere, click the Configuration tab, then click Processors, and disable hyperthreading. Reboot the ESX(i) server.

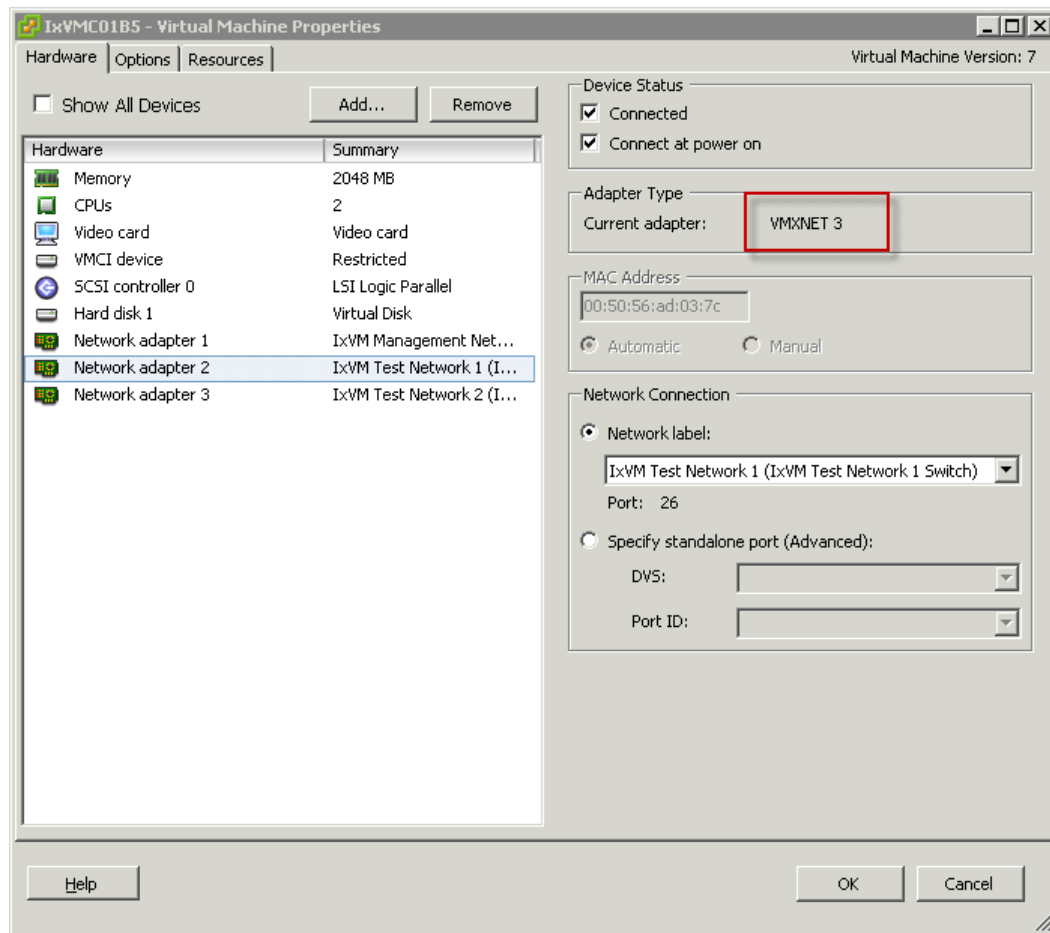
Figure 39-1. The Configuration Tab

The screenshot shows the VMware ESX Configuration console for a local host. The 'Configuration' tab is selected, and the 'Processors' section is expanded. The 'Hyperthreading' setting is highlighted with a red box and is set to 'Disabled (Not Active)*'. Other settings in the 'Processors' section include 'General' (Model: Intel(R) Xeon(R) CPU E5520 @ 2.27GHz, Processor Speed: 2.3 GHz, Processor Sockets: 2, Processor Cores per Socket: 4, Logical Processors: 8) and 'System' (Manufacturer: Dell Inc., Model: PowerEdge R610, BIOS Version: 1.2.6, Release Date: 7/17/2009 12:00:00 AM, Asset Tag: unknown, Service Tag: 905QVH1). A note at the bottom states: '* The host BIOS must enable hyperthreading before the host can activate the feature. Make sure the BIOS is correctly configured prior to rebooting this host.'

NIC Type

In vSphere, click the Hardware tab, and change the test NIC Interface types to VMXNET3.

Figure 39-2. The Hardware Tab



Visit VMware's Networking Blog for more details on how to enable Jumbo Frames: <http://blogs.vmware.com/networking/>

KVM Installation

For KVM deployments, Ixia supplies IxVM in two forms:

- as a script containing a self-extracting disk image (.img) that creates a new VM running the Ixia kernel
To install using the script, follow the procedure below.
- as RPM packages that you can install on existing Linux VMs
To install using the RPMs, follow the same procedure as for a bare metal deployment. For details see [Bare-metal \(RPM\) Installation](#) on page 39-74.

Before Installation

Before you install IxVM:

- Install and configure KVM/Qemu, if you have not already done so.
In KVM, create the bridges (virtual networks) for the VMs you will create. You will need two bridges:
 - one for the card management traffic
 - one for the generated test traffic

Note: For details on the various switches the script accepts, run the script with the help switch before you run it to install IxVM:

```
./VM_IxVM_QemuKVM-<version>.sh -help
```

Installing IxVM on KVM

1. The script requires root-level privileges to run. Login to the KVM host using an account that has root privileges.
2. Copy the VM_IxVM_QemuKVM-<version>.sh script to the KVM host. If you are copying using WinSCP, set the Transfer Settings to Binary mode .
3. Make the script file executable:

```
chmod +x VM_IxVM_QemuKVM-<version>.sh
```

4. Set your path to the location where you want to store the VM (or use -d to specify a different path when you run the script).

5. Run the script.

```
./VM_IxVM_QemuKVM-<version>.sh
```

If you want to create the the VM in a different location, use the -d switch followed by the path:

```
./VM_IxVM_QemuKVM-<version>.sh -d <path>
```

The script starts, and prompts you for a name for VM.

6. Enter a name for the VM, then press ENTER.

The script displays a list of the bridges and virtual networks on the KVM host. You must choose two bridges or virtual networks:

- one for the card management traffic
 - one for the generated test traffic
7. Type the name of the bridge or virtual network you want to use for the management traffic, then press ENTER.
 8. Type the name of the bridge or virtual network you want to use for the test traffic, then press ENTER.

The script unpacks the .img file and creates a VM in the current directory.

9. If you are using Virtual Machine Manager (VMM), restart it before you start the new VM.

The password for the VM's root account is ixia123.

Note: In addition to the .img file, the script also creates an .xml file in

```
/etc/libvirt/qemu/
```

that describes the VM's hardware configuration.

If you delete the VM from VMM, only the .xml file is deleted; you must manually delete the .img file

KVM Tutorial

Introduction

KVM (Kernel-based Virtual Machine) is a full virtualization solution for Linux that takes advantage of CPU based hardware acceleration such as Intel VT or AMD-V. KVM itself exists as a kernel module and can be installed on a Linux system that has the supported set of Intel/AMD processors.

This document will demonstrate one use case of virtualization by leveraging the capabilities of the KVM Linux kernel extensions and open source virtualization software such as Red Hat's Virtual Machine Manager.

Other sources of information

<http://www.linux-kvm.org>

KVM Homepage.

<http://libvirt.org>

libvirt Homepage.

<http://libvirt.org/docs.html>

libvirt Documentation.

Before you begin

Please make sure that you have followed the first part of this guide: "Preparing Ixia Application Controller for KVM based IxVM" and ensure that you have VNC remote connectivity to your server.

Hardware requirements

This document focuses on the Ixia Application controller (also known as "AppServer") and the time of writing, this is a single-processor Intel Xeon (quad core), 24GB of DDR2 RAM, two integrated GbE LAN ports, and one slim DVD-ROM drive.



Software requirements

- Windows 7 ISO or CD/DVD.
- IxOS 6.50 EA + IxVM Server component.
- IxNetwork 7.12 EA.

Time requirements

Start to finish, Windows 7 installation, IxOS and IxNetwork installation can take from an hour to three hours (not including the time to download).

CentOS version requirements

At the time of writing this document, only CentOS 6.3 is fully supported for IxVM – it is also important to note that performing an update (for i.e. `yum update`) will upgrade the version of CentOS to an unsupported version

Note: Running IxVM on a CentOS 5.6 KVM should still work, but there are some limitations that are caused by the version of this hypervisor. Ixia suggests you use a CentOS 6.3 KVM.

-- please do not run such commands when working with Ixia IxVM software.

In addition, make sure you ignore popup messages such as the following!

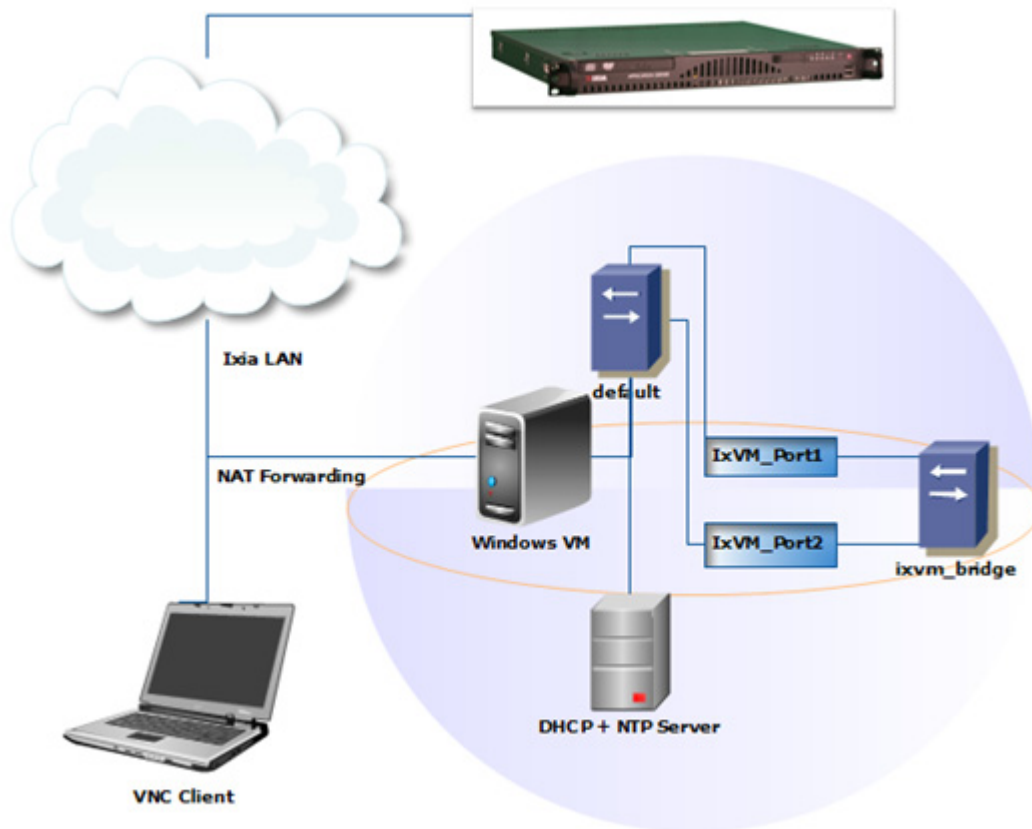
Figure 39-3. The pop-up message



Topology

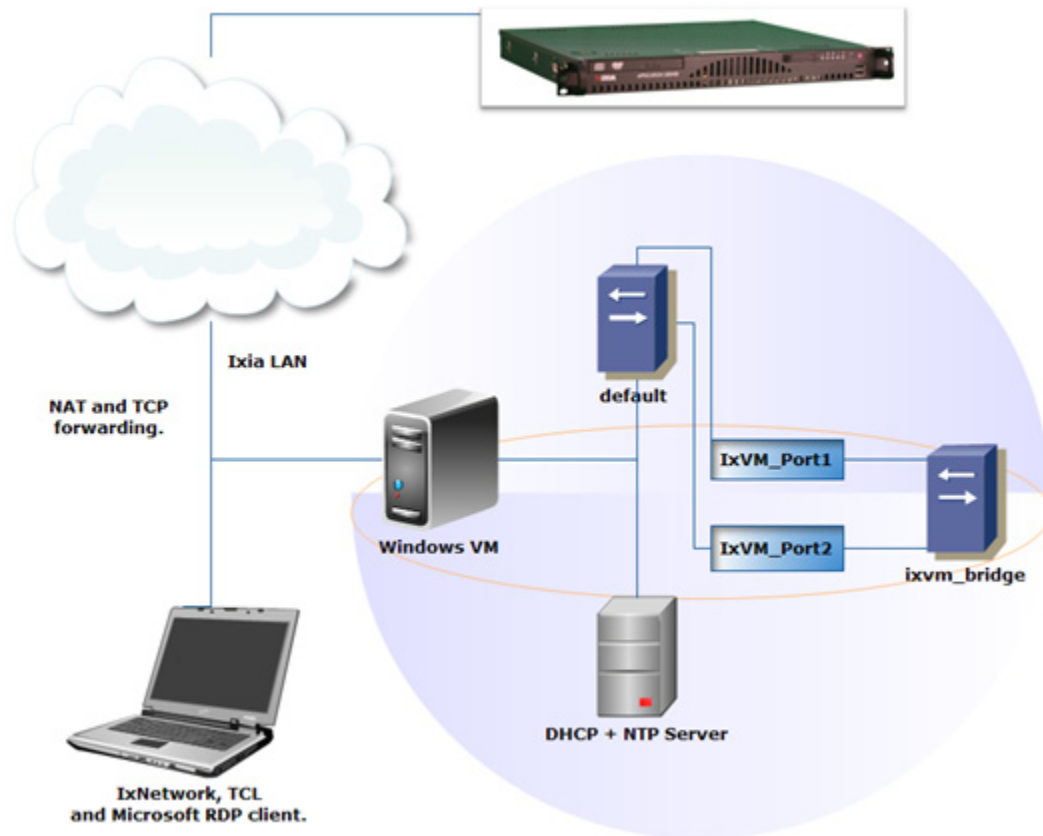
Use case 1 – Back to back test

The sphere in the following topology represents all virtualized components that are hosted by the Ixia application controller including the DHCP and NTP server.



Use case 2 – External connectivity

Building upon the previous use-case, this will focus on enabling port forwarding on the host, which will allow non-virtualized components such as IxTCL wish shell, IxNetwork and the Microsoft Remote Desktop client to remotely connect to the virtualized Windows VM without consuming CPU or memory resources on the host – this would considerably benefit customers that are conservative regarding host CPU/memory consumption by the Windows VM.



Use case 1 – Back to back test

Prerequisites

Ensure that you have the following:

- License for IxVM/IxNetwork.
- Ixia web login to access Ixia’s Downloads & Updates.

Introduction to libvirt daemon

Ensure that the libvirtd is running:[root@localhost ~]# service libvirtd status

libvirtd (pid xxxx) is running...

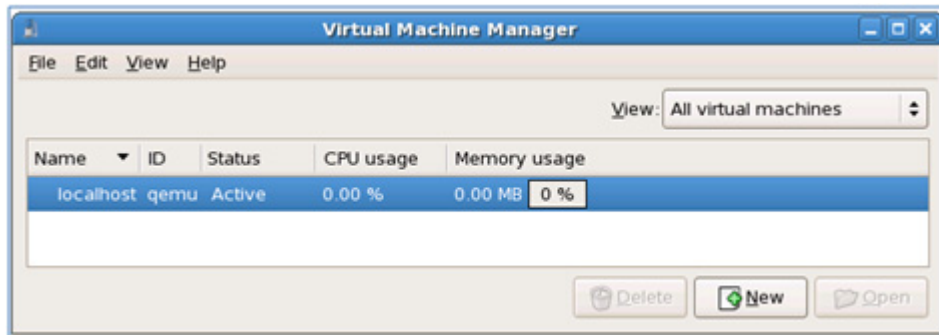
If for some reason, the service/daemon is not running, manually start of the service:

```
[root@localhost ~]# service libvirtd start
Starting libvirtd daemon:[OK]
```

Establish the QEMU connection

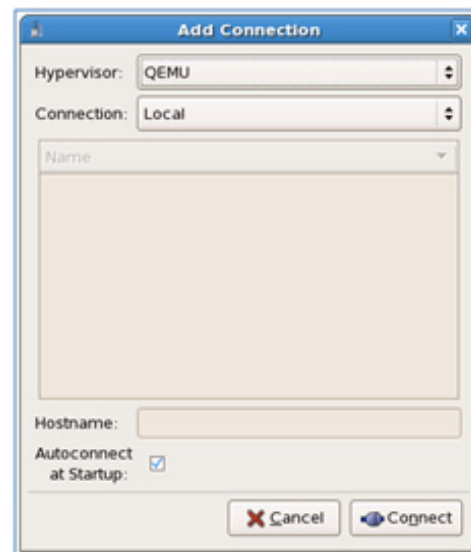
Launch VMM (Virtual Machine Manager) from Applications -> Systems Tools -> Virtual Machine Manager.

Figure 39-4. Virtual Machine Manager screen



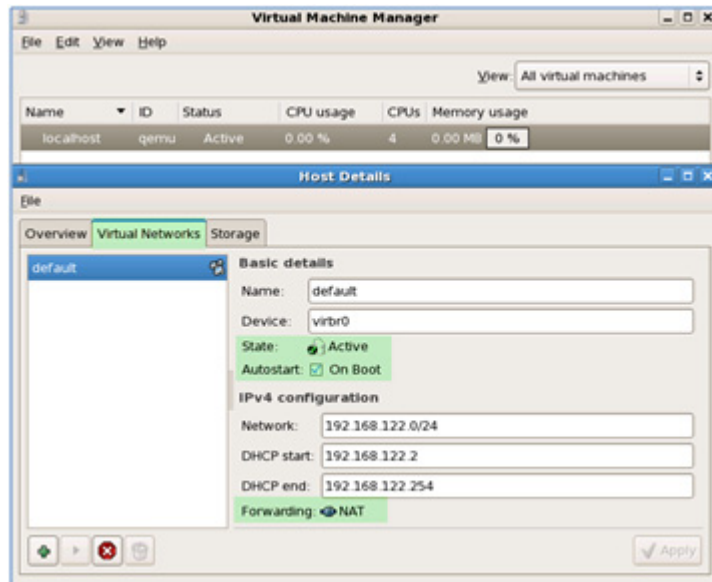
Establish a connection to the QEMU/KVM hypervisor via File -> Add Connection -> Connect:

Figure 39-5. The Add Connection screen



Note: Make sure the hypervisor is set to QEMU and not XEN. The default hypervisor for the GUI is XEN.

Figure 39-6. The Virtual Machine Manager screen



Create Ethernet Bridge for Management network

On the main dialog of Virtual Machine Manager, click on the connection to localhost (ID: qemu) to highlight it and then navigate to: Edit -> Host Details -> Virtual Networks. Click on the default network to highlight it and ensure that it is Active and enabled for “Autostart on Boot”:

VMM has utilized the Linux `brctl` command to setup and maintain an Ethernet bridge. The name of the device/network from `libvirt`'s perspective is `default` and the Linux Ethernet bridge identifier is `virbr0`; to determine the status of this bridge you can simply run: ``ifconfig virbr0`` from the shell. The purpose of the bridge is to simply connect different multiple Ethernet capable virtual devices together just as a real layer-2 switch would operate. In the context of this particular use case, the bridge will serve as the management network that will interconnect the Windows VM and two IxVM ports.

VMM will also provide the DHCP start and end addresses as parameters to a process called `dnsmasq`, which is a lightweight DHCP server (among other capabilities) to provide DHCP addresses to all connected hosts on the default bridge, and for this use case this will translate to the Windows VM and the `eth0` interface of the two IxVM ports.

For more information on `libvirt` networking:

- <http://wiki.libvirt.org/page/Networking>
- http://wiki.libvirt.org/page/Libvirtd_and_dnsmasq

Create Ethernet bridge for test ports

To establish a dedicated and isolated bridge between the IxVM ports, create a new virtual network (click on the button with the + icon):

Figure 39-7. Create a new virtual network screen



Network Name: ixvm_bridge

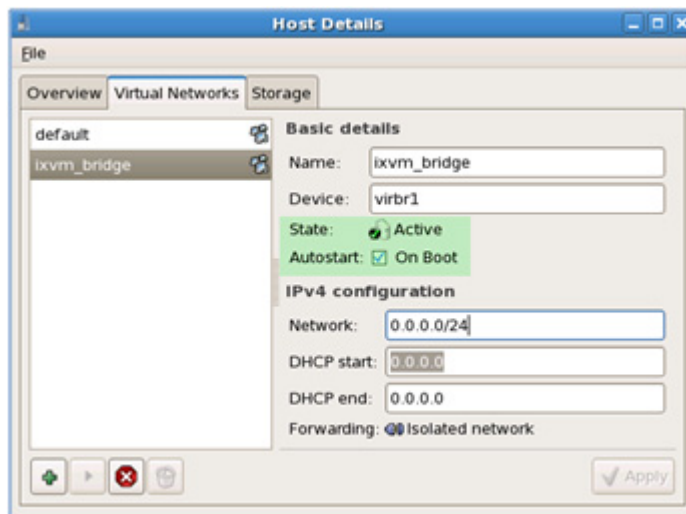
Network: 0.0.0.0/24

DHCP Start/End range: 0.0.0.0

Physical network: Isolated virtual network

Ethernet Bridge: ixvm_bridge should now be running and DHCP (through dnsmasq) should NOT be providing out any addresses since the range is null:

Figure 39-8. Host Details screen



Now, close VMM and create a new shell using Terminal (from Applications -> Accessories).

Note: This step assumes you have a separate file system mounted under /nobackup as defined in the first document. If you need to mount this directory, you must do so via `mkdir /nobackup` && `mount /dev/sda2 /nobackup`.

Create the following directories (if they do not already exist):

```
[root@localhost ~]# cd /nobackup
[root@localhost nobackup]# mkdir libvirt
[root@localhost nobackup]# cd libvirt/
[root@localhost libvirt]# mkdir images
[root@localhost libvirt]# cd images
[root@localhost images]# pwd
/nobackup/libvirt/images
```

Download the latest KVM IxVM image into the above directory and ensure executable permissions for root:

```
[root@localhost images]# chmod u+x VM_IxVM_QemuKVM-1.0.0.68.sh
```

Instantiate IxVMPort1

Run the script to create instance for IxVM_Port1:

```
[root@localhost images]# ./VM_IxVM_QemuKVM-1.0.0.68.sh
Host System: CentOS
```

```

Creating virtual machine
Enter virtual machine(domain) name: IxVM_Port1
  Setting up network configuration
      NETWORKS_LIST
default
ixvm_bridge
Enter management network name: default
Enter test network name: ixvm_bridge
Stopping libvirtd daemon: [ OK ]
Starting libvirtd daemon: [ OK ]
VIRTUAL MACHINE INFO
Machine name: IxVM_Port1
Management NIC: default
Test NIC: ixvm_bridge
#####
# [warn]: libvirt daemon has just been restarted in order
that #
# new configuration to take effect; note that it may be #
# needed to restart Virtual Machine Manager also      #
#####

```

Note: The script has essentially created a new QEMU-KVM instance and connected the new VM's eth0 virtual interface to the default network and eth1 to the ixvm_bridge (created earlier). When the IxVM_Port1 starts, it will attempt to retrieve an IP address from the DHCP server through eth0 interface.

Instantiate IxVMPort2

```

Run the script again for IxVM_Port2:
[root@localhost images]# ./VM_IxVM_QemuKVM-1.0.0.68.sh
Host System: CentOS
Creating virtual machine
Enter virtual machine(domain) name: IxVM_Port2
  Setting up network configuration
      NETWORKS_LIST
default
ixvm_bridge
Enter management network name: default
Enter test network name: ixvm_bridge

```



```

Stopping libvirtd daemon: [ OK ]
Starting libvirtd daemon: [ OK ]
VIRTUAL MACHINE INFO
Machine name: IxVM_Port2
Management NIC: default
Test NIC: ixvm_bridge
#####
# [warn]: libvirt daemon has just been restarted in order
that #
# new configuration to take effect; note that it may be #
# needed to restart Virtual Machine Manager also      #
#####

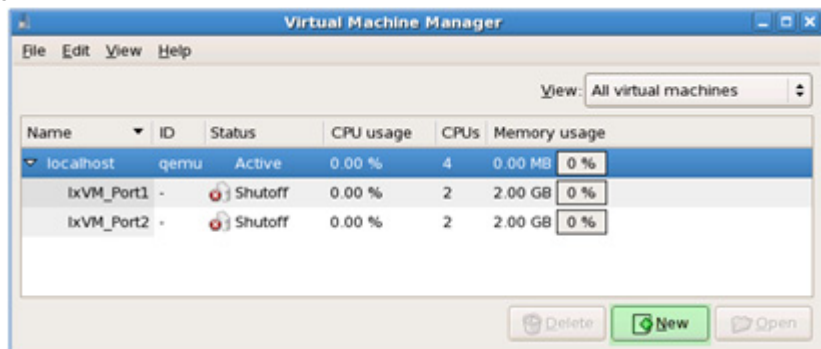
```

Note: This second VM is connected to the same default network and ixvm_bridge. The two VMs will be able to talk to each other and to the Windows VM via the default network; moreover, the two VMs will have a private and isolated connection to each other via the eth1 interfaces and ixvm_bridge.

Create and instantiate IxVM management VM

Go back to the VMM GUI, highlight the “qemu” connection and click on “New” to create a new VM instance:

Figure 39-9. The VM instance screen



Note: The VM status for both instances is in the Shutoff state and while they are in this state they will not consume any host CPU cycles as it will need all available processing power to complete the following task of instantiating a Windows VM as fast as possible.

Name: IxVM

Virtualization method: Fully virtualized (CPU architecture: x86_x64)

Hypervisor: KVM

Installation method: Local installation media.

OS Type: Windows

OS Variant: Microsoft Windows XP (x86_64)

Installation media: <Select ISO or CD-ROM>

Storage: File (disk image)

Location: /nobackup/libvirt/images/IxVM.img

Size: 60000 MB

Allocate: Uncheck (this option will allocate disk space on the fly as needed).

Network: Virtual Network -> Network: default.

Memory

Max memory: 4000 MB

Startup mem.: 1024 MB

Virtual CPUs: 2

Click on “Finish” in the last dialog to create and launch the VM.

Install Windows 7

Once the VM has been successfully instantiated, follow and select the default options of installing Windows 7:

Pre-requisites:

- Ensure that the Windows 7 installation key is available
- Ensure that you have the right drivers available before you start the installation.

Installation Steps:

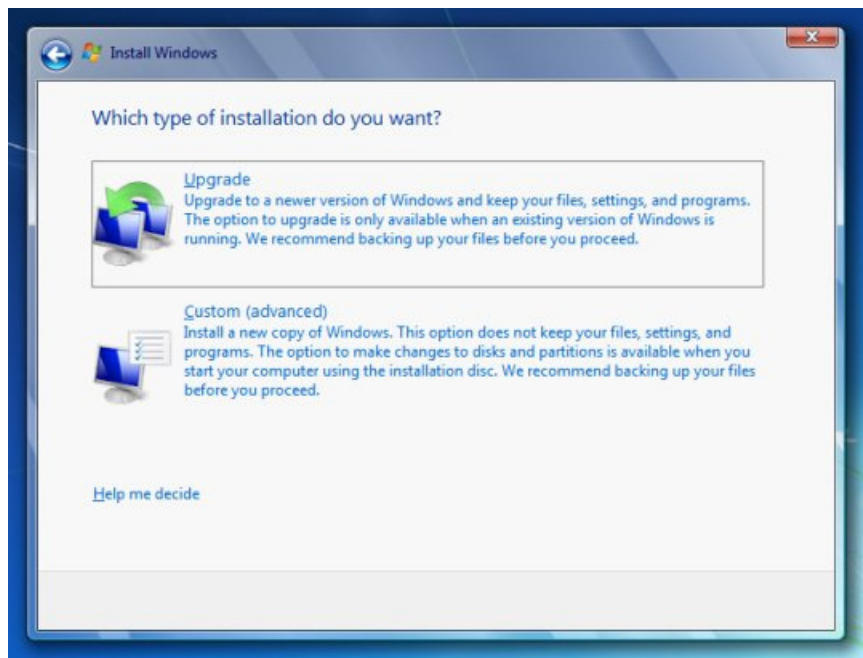
1. First, you need to choose your language and keyboard settings.

Figure 39-10. Windows 7 Language and Keyboard



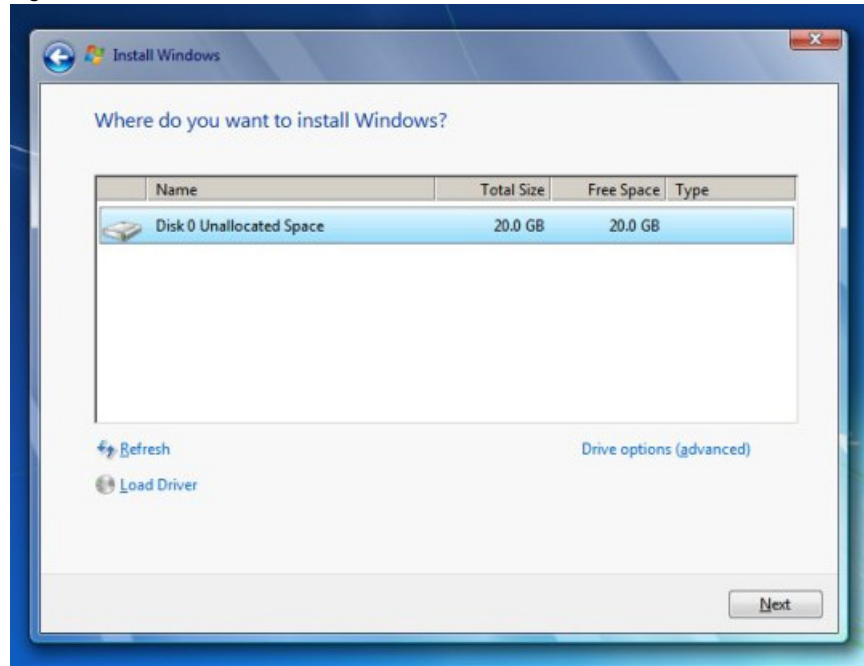
2. The *Install now* window appears. Click *Install now* to proceed.
3. The License Agreement window appears. Select the *I accept the license terms* check box and click *OK*.
4. The Installation Type window appears where you need to select the type of installation you want to do. Select *Custom (advanced)* for a fresh install of Windows 7.

Figure 39-11. Windows 7 Installation Type



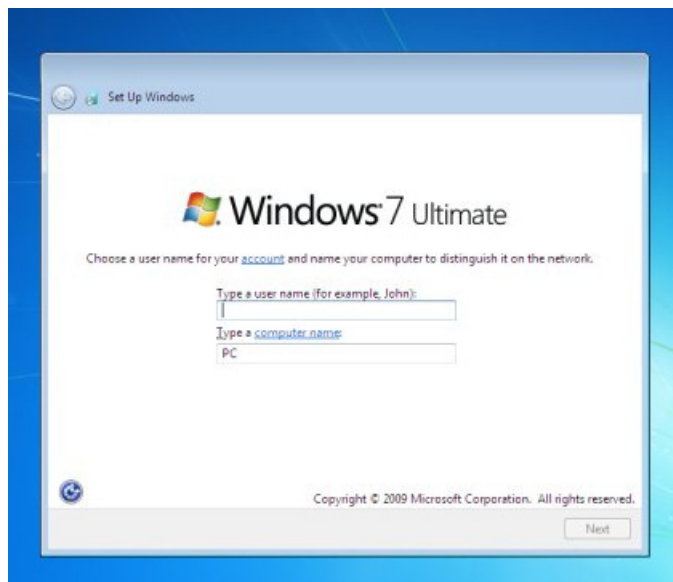
- Next you need to choose the location in your disk where you want to install Windows 7.

Figure 39-12. Windows 7 Installation Location



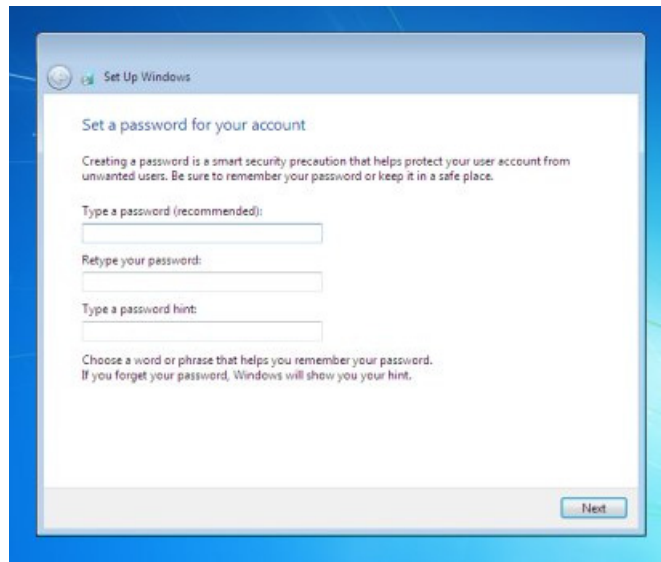
- The installation starts and after sometime you will be asked to provide your user name and computer name.
You can provide the user name as *IxVM User*.

Figure 39-13. Windows 7 User name



- Next you need to provide a password for your account.

Figure 39-14. Windows 7 Password



Note: Leaving the password blank is optional but will provide an advantage as it will allow the Windows VM to login automatically without a blocking login prompt.

8. You then need to configure the updates to *Ask me later*. Next configure your time zone and location.
Windows 7 is now successfully installed on your virtual machine.

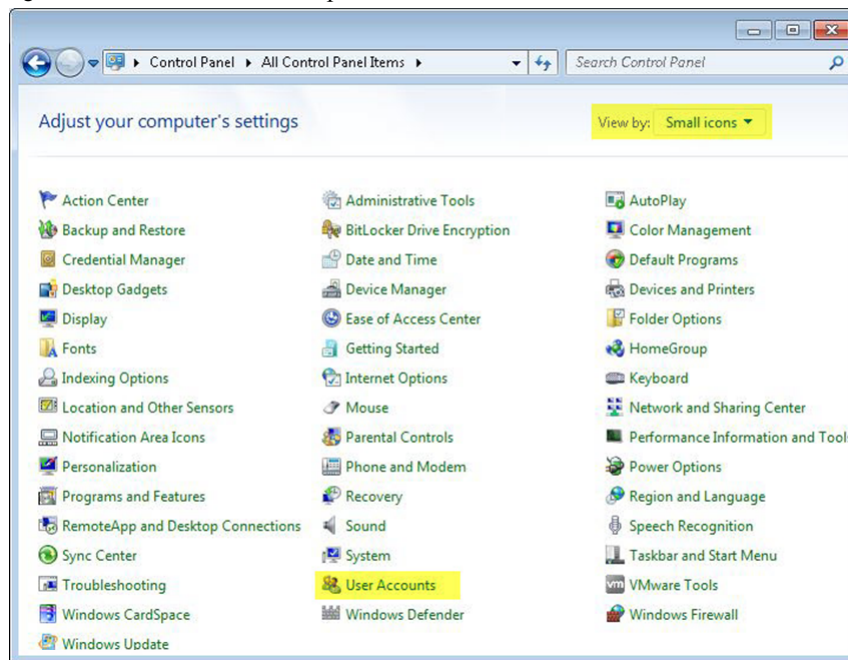
Customize Windows 7

To improve security, Windows 7 has the user account control option that is set by default to ask permission for specific operations (ex: running different executable files). To run IxVM properly, this option has to be disabled:

To change the settings, change the view in Control Panel to "Small icons" and go to "User Accounts".

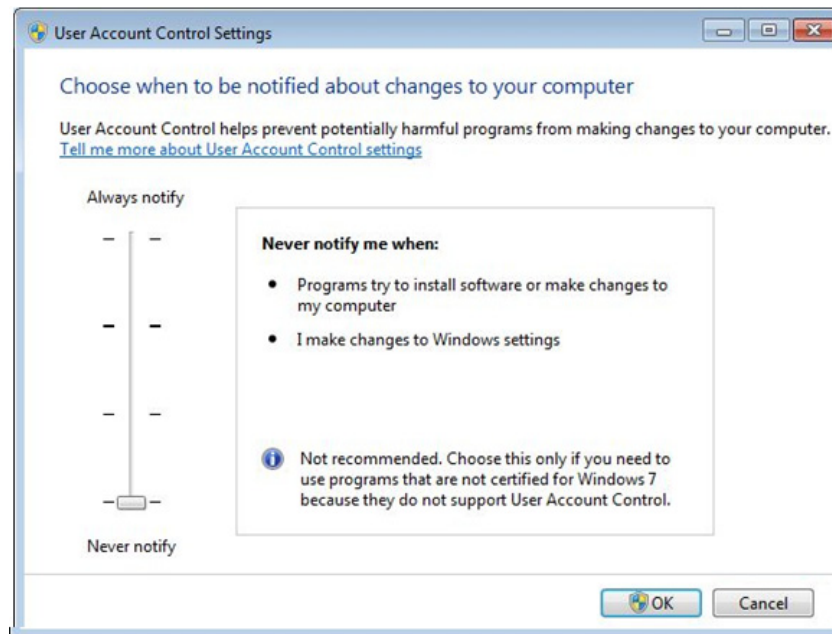
1. Click **Start** and then select **Control Panel** option.
2. Click **User Accounts** option. The user control panel screen shows as follows:

Figure 39-15. The user control panel screen



Go to "Change User Account Control Settings" and drag the dial down to "Never notify". The screen shows the options as follows:

Figure 39-16. The user control panel screen

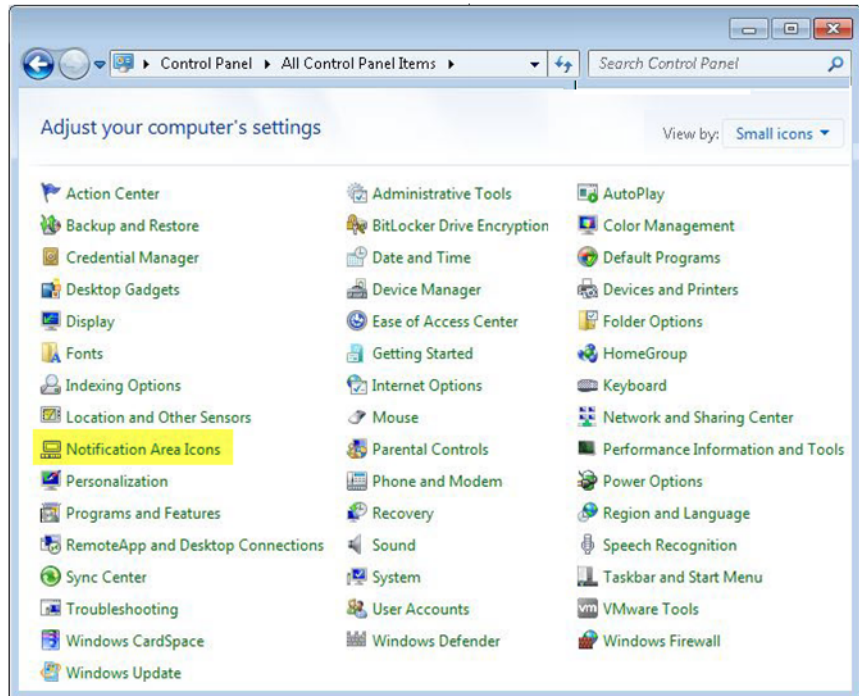


You need to restart Windows for the changes to take place. Please do this before continuing with the rest of the changes.

By default, Windows will display alerts relating to the status of the Firewall, Windows Automatic Updates and Virus Protection. Since none of these services are required, disable the alerts:

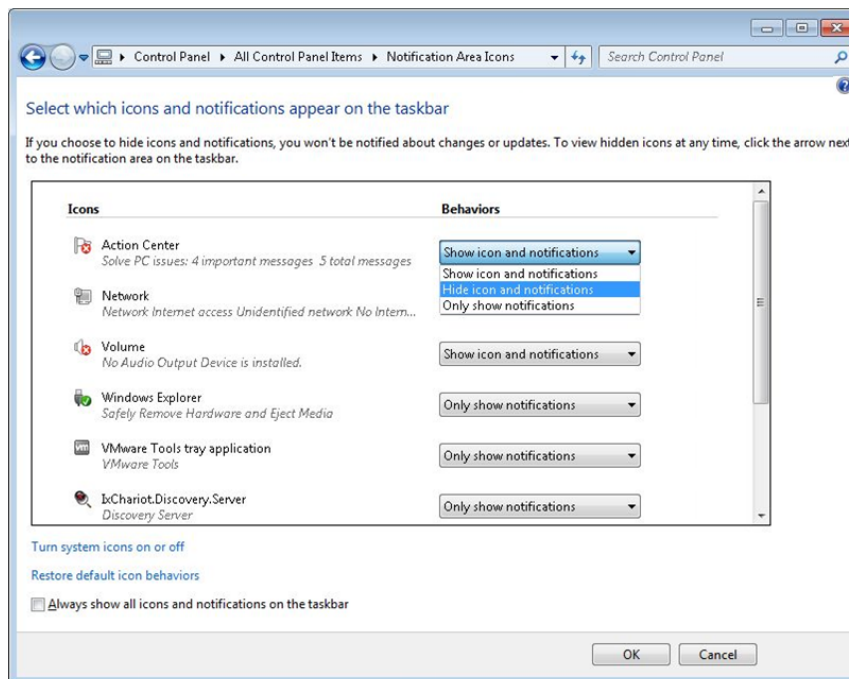
1. Click **Start** and then select **Control Panel** option.
2. Click **Notification Area** icons.

Figure 39-17. The user control panel screen



Now change the Action center behavior to "**Hide icon and notifications**" and press **OK**.

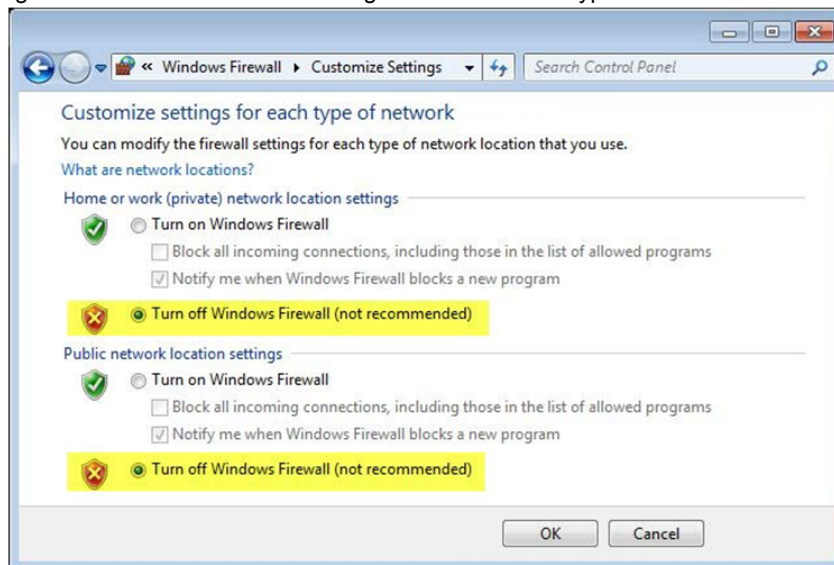
Figure 39-18. The user control panel screen



Ensure the Windows Firewall is disabled (since it will interfere with IxVM management traffic):

1. Click **Start** and then select **Windows Firewall** option.
2. Turn **Windows Firewall** on or off.

Figure 39-19. The Customize settings of each network type screen

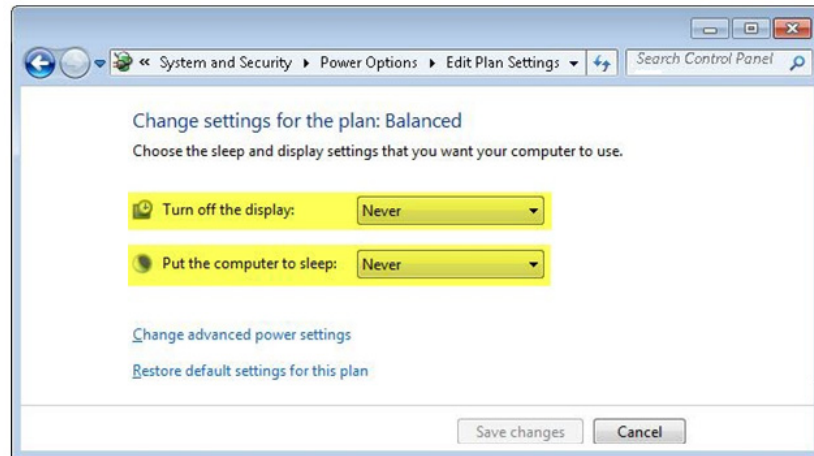


The "Sleep" and "Turn off display" options need to be disabled also:

1. Click **Start** and then select **Control Panel** option.

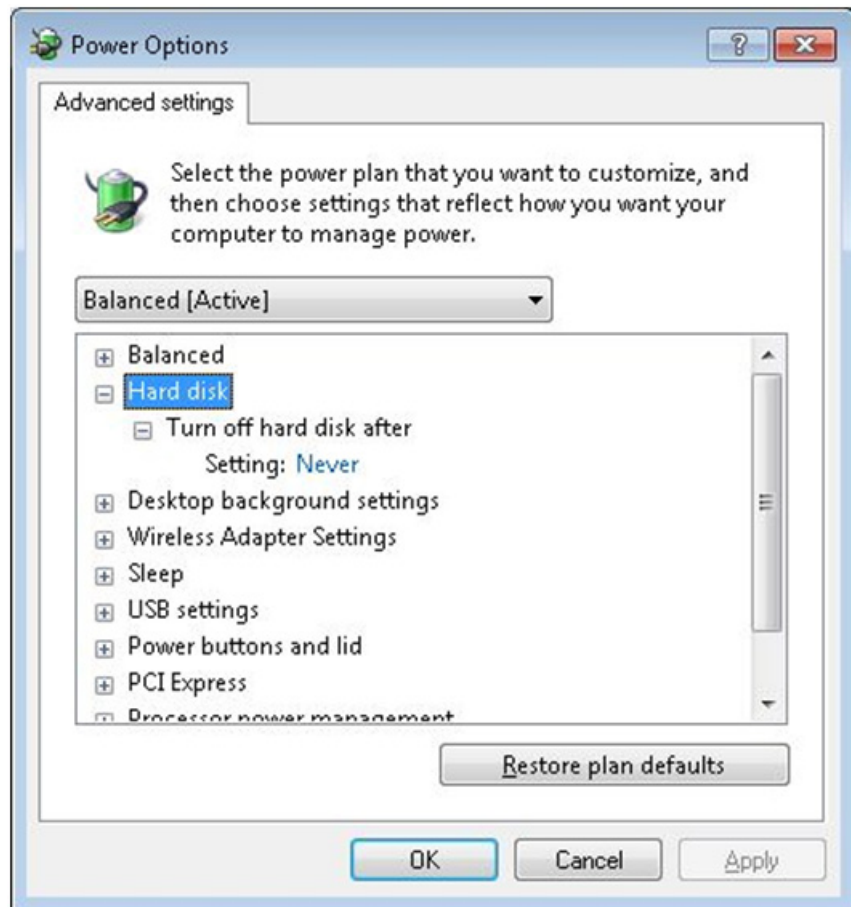
2. Click **Power options** and then select **Change plan settings** option.

Figure 39-20. The Change plan settings screen



In the same window as above, go to "Change advanced power settings" and under **Hard Disk** option Turn off hard disk after Setting enter 0 (zero) to set to Never. When done, click **OK** and **Save** changes to finish with setting up the power options.

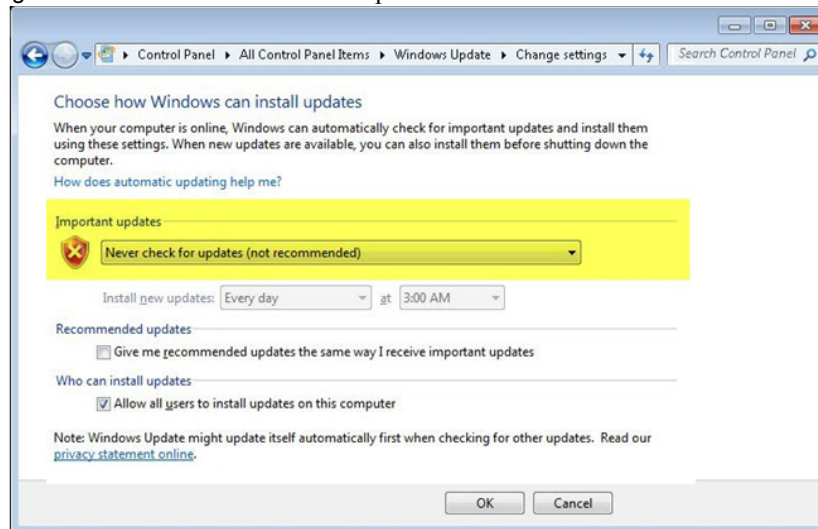
Figure 39-21. The Power Options screen



Next, disable the automatic updates:

1. Click **Start** and then select **Control Panel** option.
2. In the windows updates and in the left side of the screen click on the **Change settings**.
3. Set the Important updates option to **Never check for updates**.

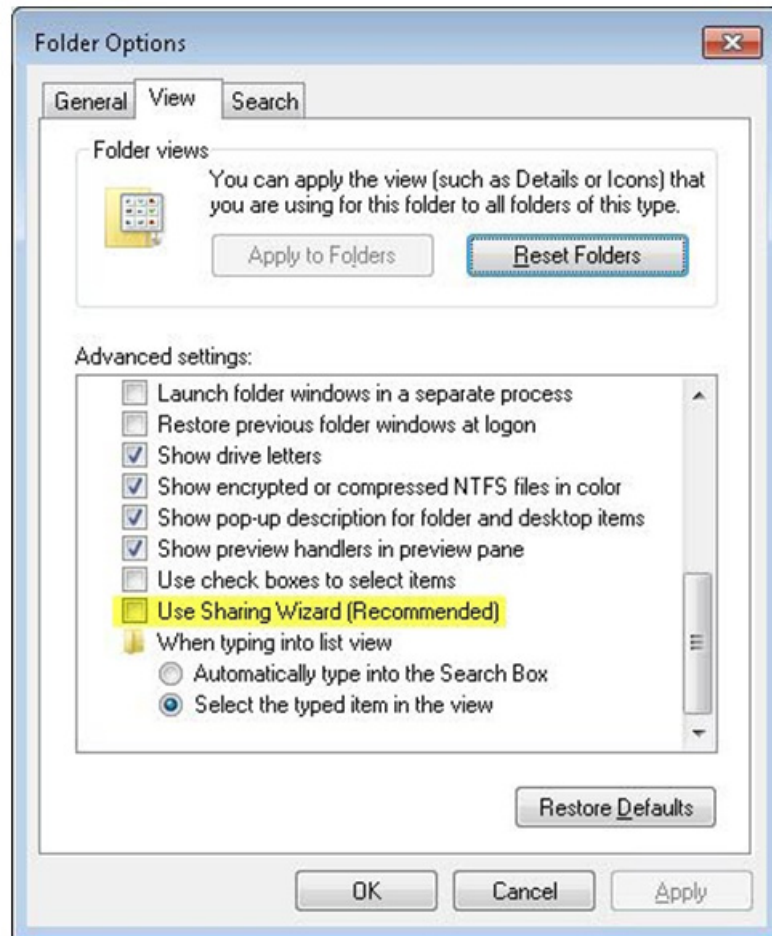
Figure 39-22. The Never check for updates screen



In case you intend to use Deployment Wizard to install or upgrade a virtual chassis on this Windows machine you need to disable Simple file sharing or Sharing wizard:

1. In the Windows Explorer options click Tools.
2. Click **View** tab under **Folder Options**.
3. Scroll down to the bottom where you will find "Use Sharing Wizard".
4. Clear the option.

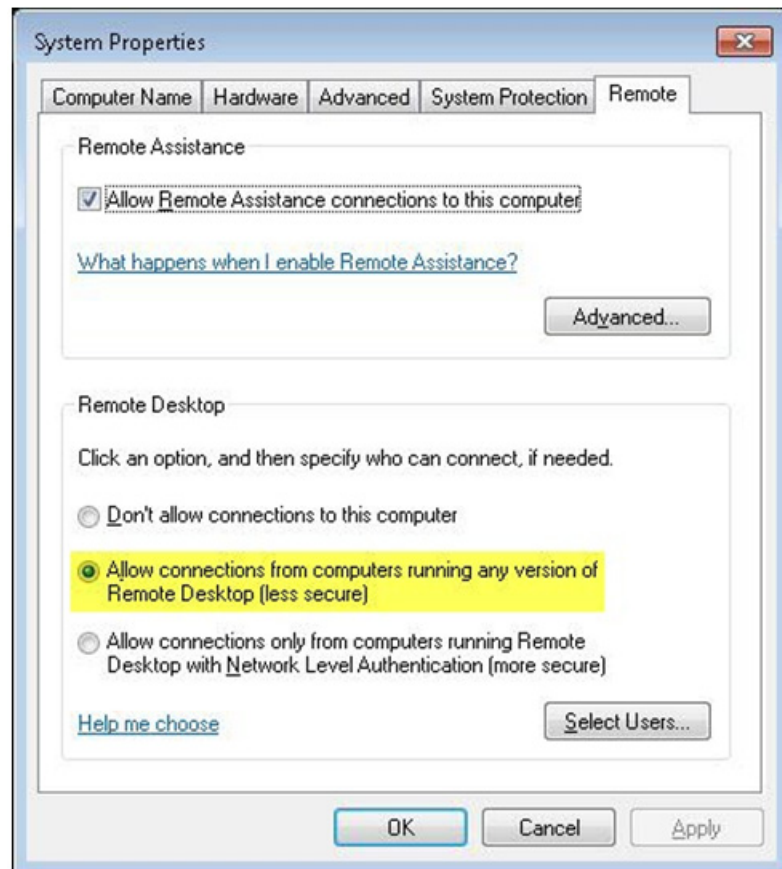
Figure 39-23. The View Tab screen



Finally to login remotely to this Windows 7 machine, you need to enable the remote desktop option:

1. Click start and right click on **My Computer**.
2. In the left side of the Window select **Remote settings**.
3. Select the second option from the radio button options as shown in the below screen shot:

Figure 39-24. The Remote Tab screen

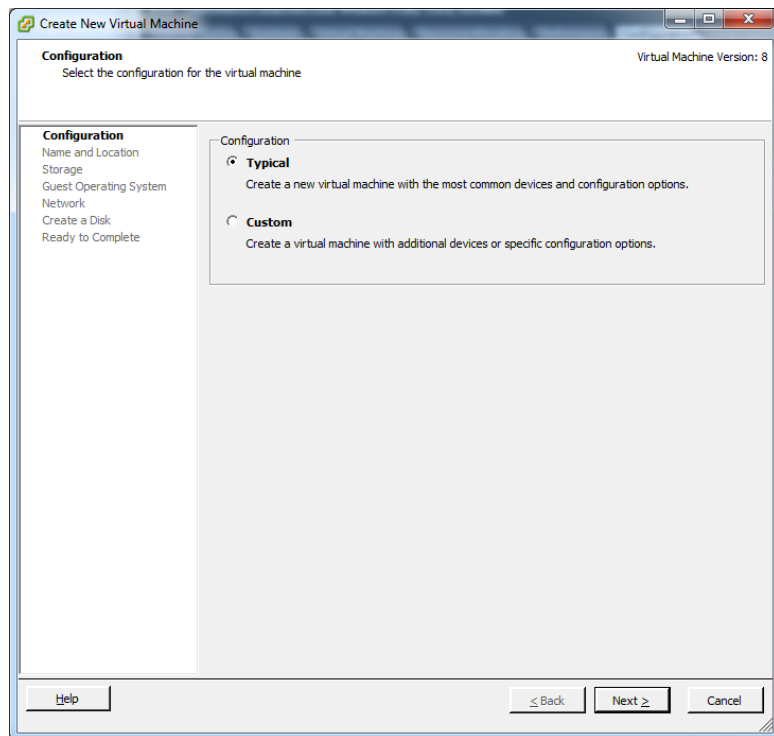


IxVM Live CD Installation

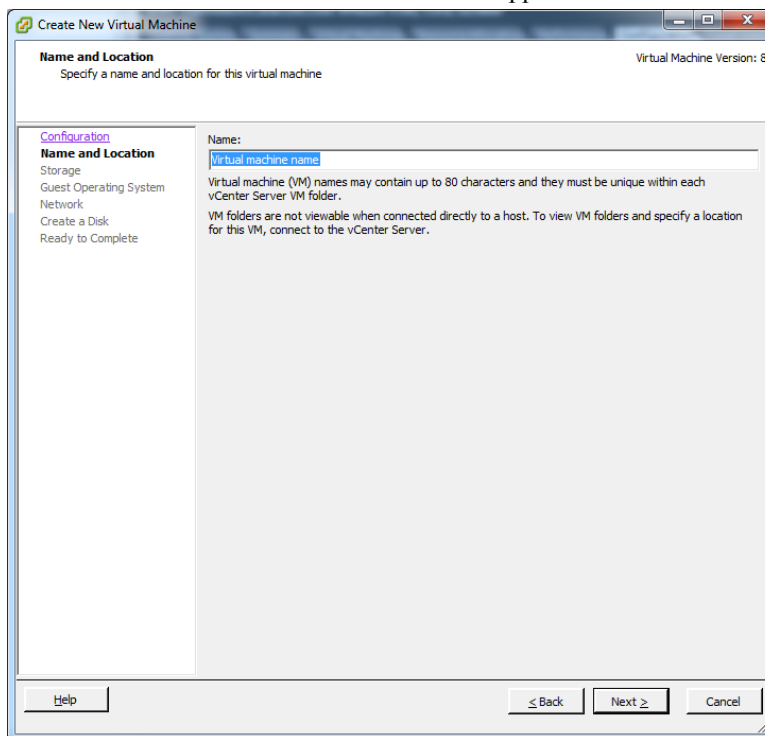
The IxVM liveCD image can be installed in a virtual machine created on any of the supported *Hypervisor / Host OS* on page 39-3.

Here is a step by step guide to deploy a liveCD image on a virtual machine, using VMware 5.0:

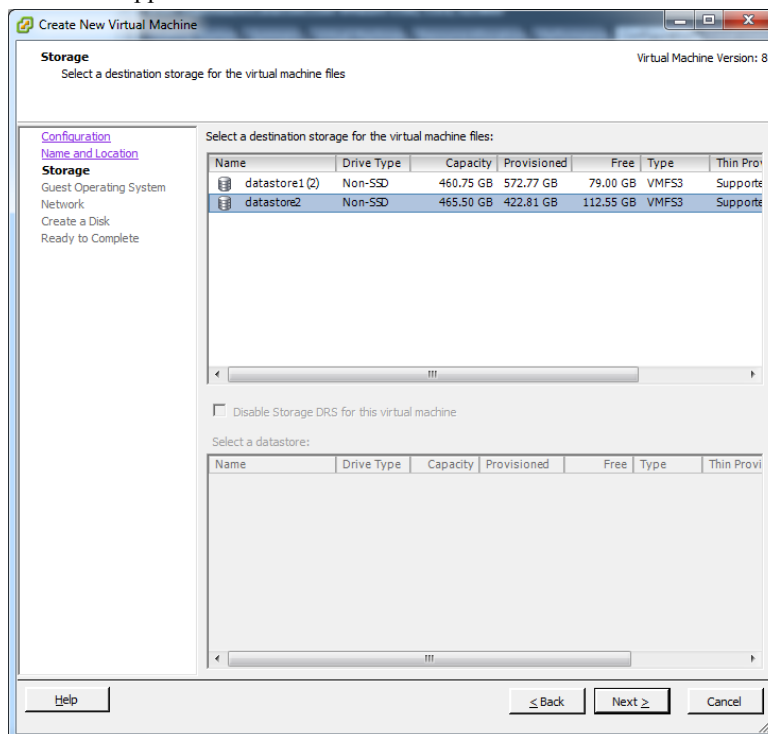
1. Copy the .iso file for the live CD to the hypervisor datastore.
2. On the **File** menu, click **New**, and then click **Virtual Machine** (Ctrl+N hot-key). The **Configuration** window appears as shown below.



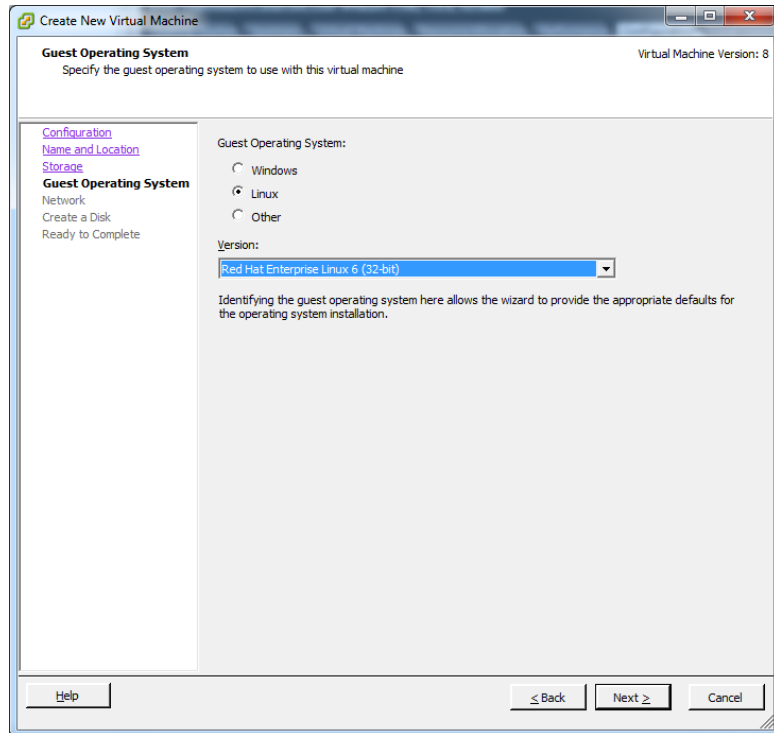
3. Select the type of configuration you want for the virtual machine, and then click **Next**. The **Name and Location** window appears.



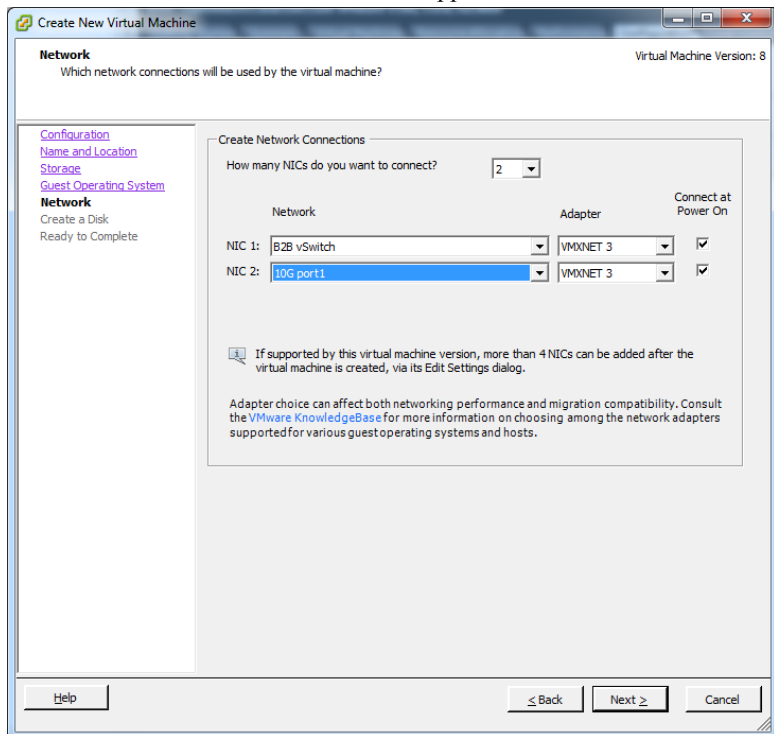
4. Type the name of the new virtual machine, and then click **Next**. The **Storage** window appears.



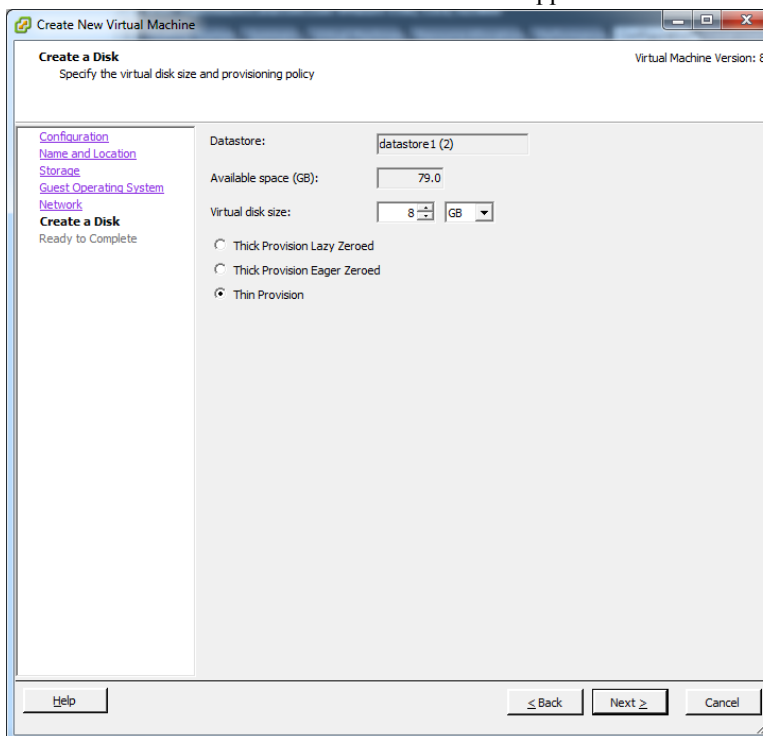
5. Select the destination storage for the virtual machine files, and then click **Next**. The **Guest Operating System** window appears.



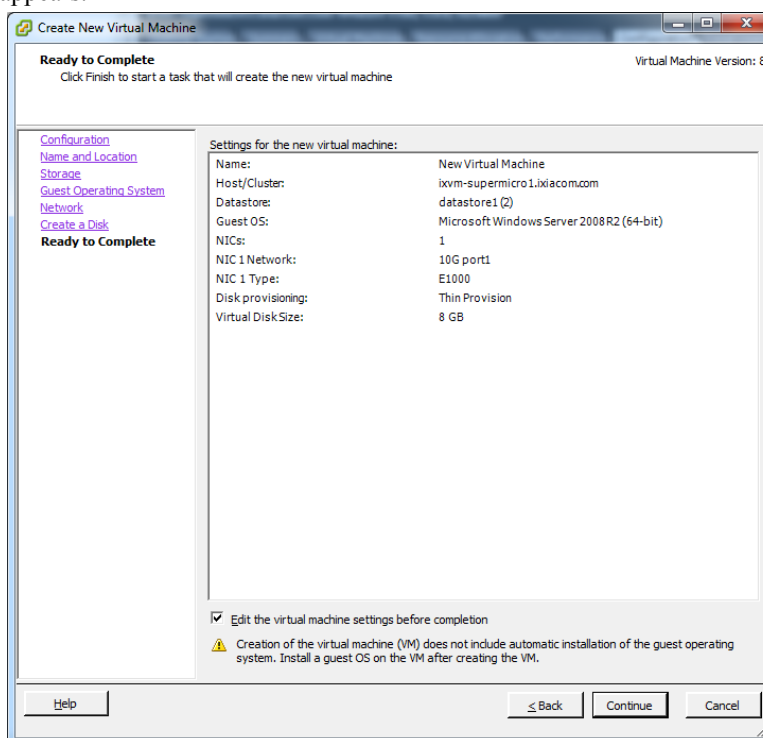
6. Select the **Red Hat Enterprise Linux 6 (32-bit)** guest operating system, and then click **Next**. The **Network** window appears.



- Specify the network connections that will be used with the virtual machine, and then click **Next**. The **Create a Disk** window appears.



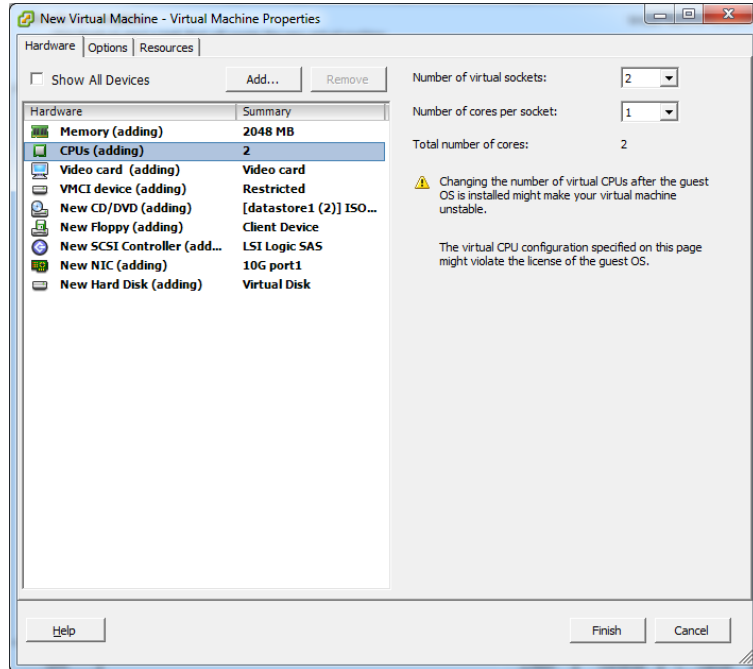
- Create the disk for the virtual machine by specifying the disk size and provisioning policy, and then click **Next**. The **Ready to Complete** window appears.



9. Click **Continue**. The **Virtual Machine Properties** window appears.

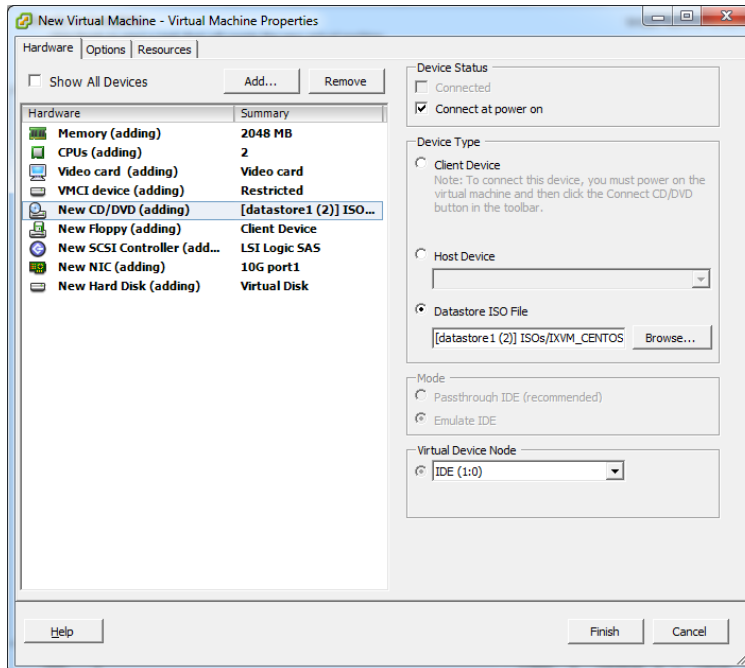
Note: Select the **Edit the virtual machine settings before completion** check box before clicking **Continue**.

10. In the **Virtual Machine Properties** window, select **CPUs** from the list of Hardware on the left. Add 2 CPU cores and 2 GB of memory to the VM.



11. Next select **New CD/DVD** and define the following properties:

- Under **Device Status**, select the **Connect at power on** check box.
- Under **Device Type**, click **Datastore ISO File**.
- Click **Browse**, and then select the live CD .iso file from the local datastore.

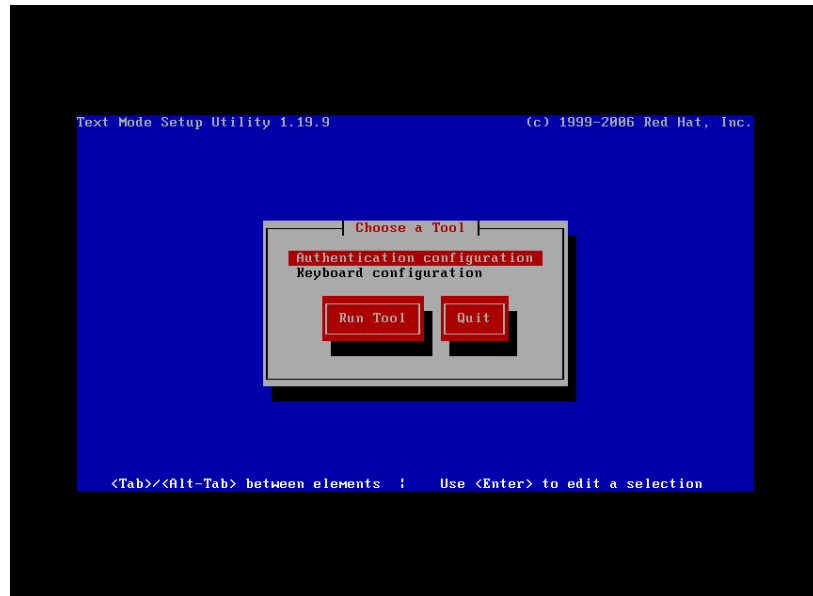


12. Click **Finish** and localize the VM in the hypervisor inventory.

13. Power on the virtual machine. The following boot screen appears.



14. After the booting is done, the setup utility tool appears as shown below.



15. Navigate to the **Quit** button by clicking **Tab**. The CentOS login screen appears, where you have to log in with the root user (no password required).

Note:

- The LiveCD can only be used in the supported virtualized environments (VMware, KVM, ESX).
- The LiveCD cannot be used on physical machines due to the custom Ixia kernel.
- The LiveCD can be used only if there is at most one hard disk configured on the virtual machine (it also works with no hard disk configured).

Install IxOS and IxNetwork

From the Windows VM instance, download and install IxOS 6.60 EA and IxNetwork 7.20 EA Patch1 (respectively). You will be asked to reboot after the IxOS installation and once more after the IxNetwork 7.20 EA Patch1 installation. In addition, please ensure that the appropriate IxVM/IxNetwork licenses are also installed.

Note: When installing IxOS, make sure Client, IxVMServer and Tcl Server are selected to be installed. In the “Custom setup” dialog, you will be given a choice over Demo Server or IxVM Server. Select IxVM Server.

Install and configure NTP server on the host

For this use case, we will be using the Linux host as the NTP server. To do this, the NTP package should be installed:

```
[root@localhost ~]# yum install ntp
[root@localhost ~]# service ntpd start
Starting ntpd:[ OK ]
```

To allow the host to serve NTP requests, modify `/etc/ntp.conf` and append the following line:

```
restrict 192.168.122.0 mask 255.255.255.0 nomodify notrap
```

Note: 192.168.122/24 is the default network for libvirt and since we have used the default network for the management bridge, the entry for `/etc/ntp.conf` should match.

Allow the NTP service to run at boot-time and manually restart the service for the new modifications to take affect:

```
[root@localhost ~]# chkconfig ntpd on
[root@localhost ~]# service ntpd restart
Shutting down ntpd:[ OK ]
Starting ntpd:[ OK ]
```

Configuring DHCP Server

The DHCP server is required for a faster way to deploy IxVM Cards and attach them to Virtual Chassis in an environment that is isolated from any automatic IP address policy. This server provides IP addresses based on default configuration,

and also has an interface that can be used as a tool to customize the DHCP Server.

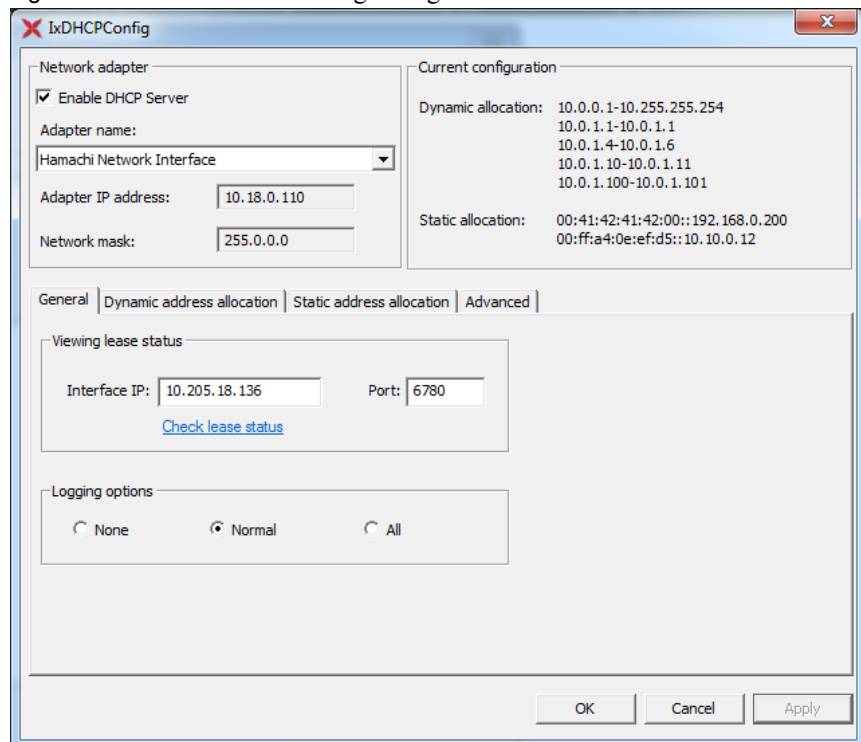
IxVM server needs to include DHCP server for accelerating card deployment in private management networks.

Note: Open DHCP Server used by IxVM server only listens on Static Interfaces that means the IP address is fixed and not obtained from another DHCP Server. Any dynamic interfaces specified are ignored.

You can configure the DHCP server through the DHCP Configuration dialog box that can be found in IxServer **DHCP Configuration** under under **Tools**. This option is unavailable if you have not installed the IxVM component for IxServer. This command starts a standalone application that configures the DHCP server.

The following image shows the IxDHCPConfig dialog box:

Figure 39-25. The IxDHCPConfig dialog box



- **Network adapter:** This section provides details of all available interfaces on the system. Under Logging options, if you click All, server listens to all available interfaces. An interface is considered available if it has a static IP that means the IP address is fixed and not obtained from another DHCP Server. Any dynamic interfaces are ignored.

- **Enable DHCP Server:** This section starts or stops the DHCP service.
- **Adapter Name:** Displays the name of the selected adapter.
- **Adapter IP address and Network Mask:** When you choose a network adapter, the IP and Mask is automatically populated under Adapter IP address and Network mask.
- **Current configuration:** This represents a short description of more important configured information on DHCP Server.
 - **Network Adapter IP:** This is an interface on which the DHCP Server listens after IP request.
 - **Dynamic address allocation:** This shows all IP ranges that can be used for dynamic allocation.
 - **Static address allocation:** This shows all pair MAC :: IP configured for static IP allocation.

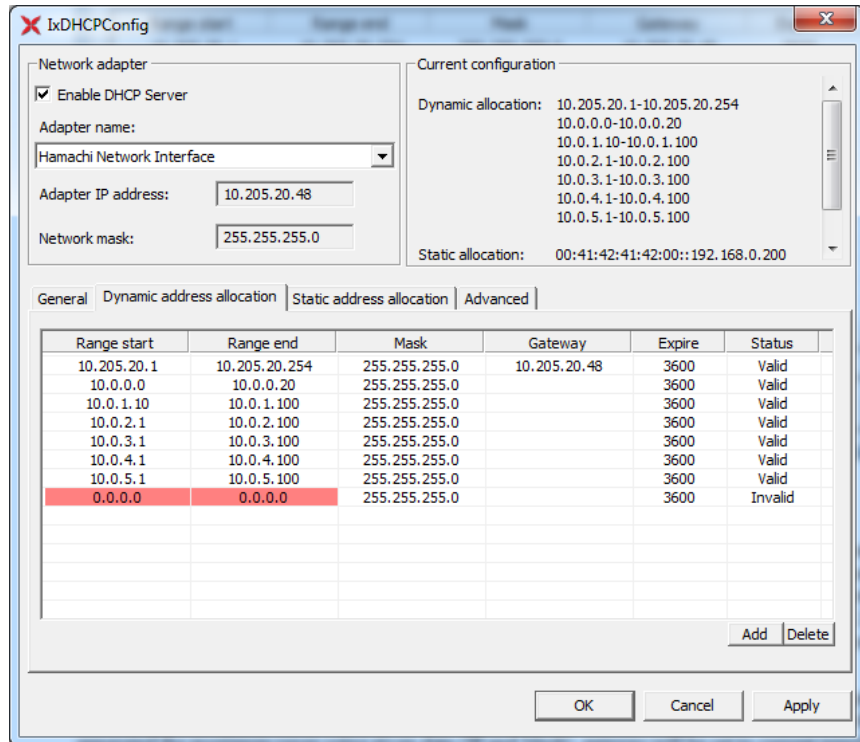
General Tab:

- **Viewing lease status:** This option shows the used IPs offered by DHCP Server. You can configure it on any interface (IP must be valid) to check the IP allocation status on server by using an HTTP request on this IP:Port configuration.
- **Check lease status:** This link shows the lease status that is set up by IP and Port in the default browser.
- **Loggings options:** This option shows the levels of logging offered by DHCP Server. By default, it is set to Normal.

Dynamic Address Allocation tab:

The following image shows the **Dynamic Address Allocation** tab:

Figure 39-26. The Dynamic Address Allocation tab

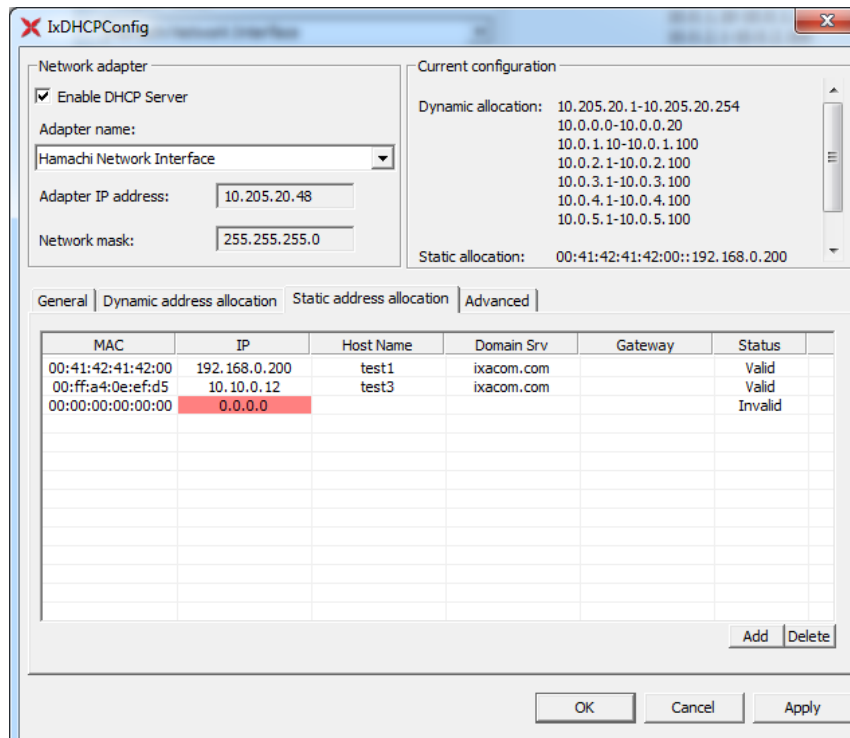


- **Dynamic Address allocation:** This tab has a list of dynamic assigned IPs. Each line represent one range, and each range can have its **Mask**, **Gateway**, and **Expire Time** configured individually.
- **Auto range:** If no ranges exist in the configuration, the application adds one auto generated range when you select the interface, on which the DHCP Server listens. This auto range is generated from selected interface IP and MAC, and the **Gateway** is set to the current interface, the **Expire** time is set to 3600 seconds, and the **Mask** value is taken from the selected interface.
- **Default range:** If you add a new interface, this is autocompleted with a default range. The default range is marked as invalid, and turned as valid when you enter valid data.
- **Range Start –Range End:** The range start checks each item separately to assure that it is a valid IP. If you change an item to invalid in a valid range then that particular item change its background to red and the range becomes invalid. If validation of IP is passed then, start IP must be smaller than end IP, and if it does not pass then both items change their its background to red.
- **Mask and Gateway:** If valid IP address is not showing in these columns then background changed to red.
- **Status:** The status shows current status of a range, and has the following values:
 - Auto: The range is automatically added and is considered as a valid range
 - Invalid: This range is not valid and is not in use.
 - Valid: This specifies a valid range and is included in the configuration file.

Static Address Allocation tab:

The following image shows the **Static Address Allocation** tab:

Figure 39-27. The **Static Address Allocation** tab



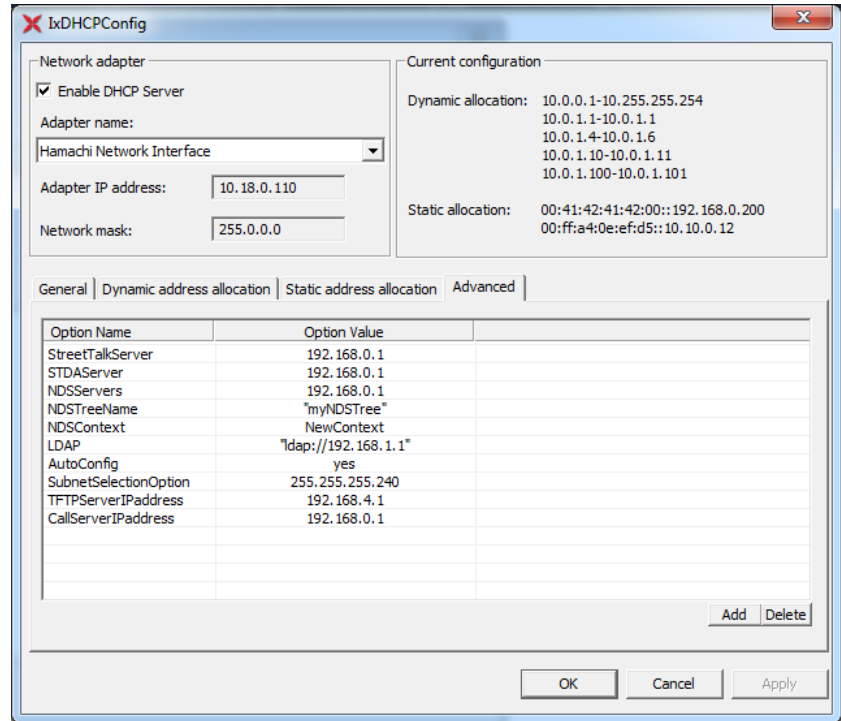
- **Static Address allocation:** This field shows the list of mandatory pair options **MAC** and **IP**. This can be completed with other options as **Host Name**, **Domain Server** and **Gateway**. Each line represents one MAC:IP associations and they must be unique as **Host Name**.
- **Default values:** For static host default values are just MAC and IP these are also mandatory fields.

Advanced Tab:

In its default state this tab is blank and needs to be configured with care as the changes here have a global effect. More detailed information on all available commands refer to [List of DHCP Options](#).

The following image shows the **Advanced** tab:

Figure 39-28. The **Advanced** tab



DHCP Server customization in Deployment Wizard

Deployment Wizard shows DHCP Server customization on both Install and Upgrade flows.

The available options are to Enable or Disable DHCP Server on IxServer.

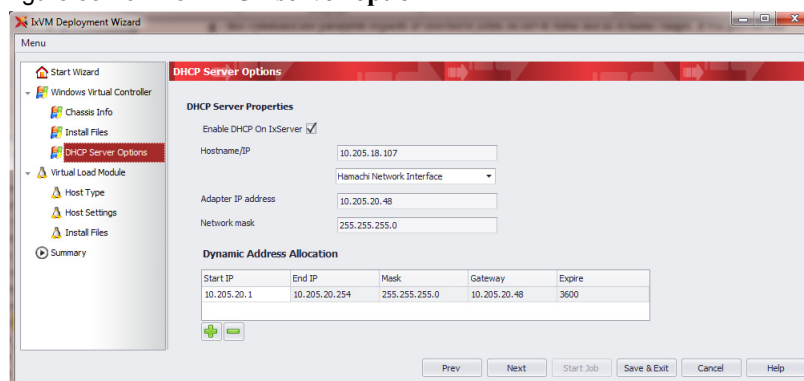
Deployment Wizard shows the same DHCP Server properties as in IxServer - Dynamic Address Allocation tab.

Deployment Wizard provides the following options:

- The list with Network Adapters properties from the host machine.
- Support for updating IxDHCPServer.ini file with properties received from Deployment Wizard.

The following image shows the **DHCP server options** dialog box under **IxVM Deployment wizard**:

Figure 39-29. The DHCP server option



IxOS Topology Transfer

The IxServer.ixs file contains all the information about card and port map, chassis type, card types, stream configurations, and port properties. Using this feature, the IxServer.ixs file gets copied from an older IxOS installation folder to the newly installed IxOS folder. You can copy this file by using various methods as follows:

- Use Deployment Wizard to select the required topology from the available IxOS versions to transfer
- Use the IxOSTransferTopology importer application that is automatically run at the end of any new IxOS installation to:
 - Copy virtual chassis configuration from the latest version
 - Copy virtual chassis configuration from the last run version
 - Copy virtual chassis configuration from a specific IxOS version
 - Copy virtual chassis configuration from a manually selected folder

Use Deployment Wizard to select the required topology from the available IxOS versions to transfer

This option is available when using Deployment Wizard to upgrade an already existing chassis.

In the **Install Files** page under **Windows Virtual Controller**, all the IxOS versions that are detected on the chassis are shown in the right side pane.

You can select any of the available versions to have its IxServer.ixs file copied to the installation folder of the new IxOS version.

Note: All the available versions are shown, which include the uninstalled versions as well. If a version does not have the .ixs file available, it does not appear in the pane.

Use the IxOSTransferTopology importer application that is automatically run at the end of any new IxOS installation

- Copy virtual chassis configuration from the latest version

In this option, the topology configuration file is copied from the latest installed IxOS version. All installed IxOS versions are sorted and the latest version is picked up.

Note: If the application cannot find the latest installed version, this option is unavailable.

- Copy virtual chassis configuration from the last run version

In this option, the topology configuration file is copied from the most recently run IxOS version. The latest version is taken from the registry.

Note: If the application cannot find the last run version or the last run version does not have a valid .ixs file, this option is unavailable.

- Copy virtual chassis configuration from a specific IxOS version

When you select this option, the **Select version** page appears. This page contains all the available versions, from which you can select one version (only the valid versions appear).

Click **Next** again. The configuration file is imported from the selected version. After the import configuration is done, the **Finish** page appears.

- Copy virtual chassis configuration from a manually selected folder

You need to manually identify the source folder, from where you want to copy the file. This can be a backup folder for older IxOS versions, but you need to have only one .ixs file in each sub-folder.

Select the folder from where the topology needs to be copied. You can copy the topology either from installed or uninstalled versions that have a valid .ixs configuration file.

- Clean install

If you select this option, the installer skips the topology transfer step. No topology file is transferred and a new empty topology file is created at the first IxServer start.

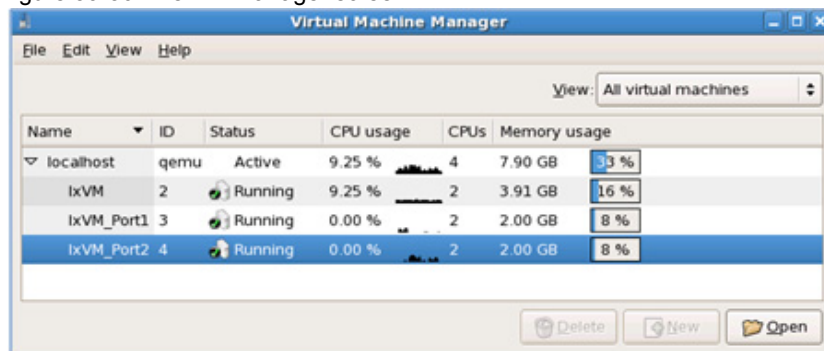
Note: If you want to manually run the IxOSTransferTopology tool between various IxOS installations, you can find it at the following location:

C:\Program Files\Ixia\IxAdmin\bin\IxOSTransferTopology.exe

Instantiate IxVM test ports

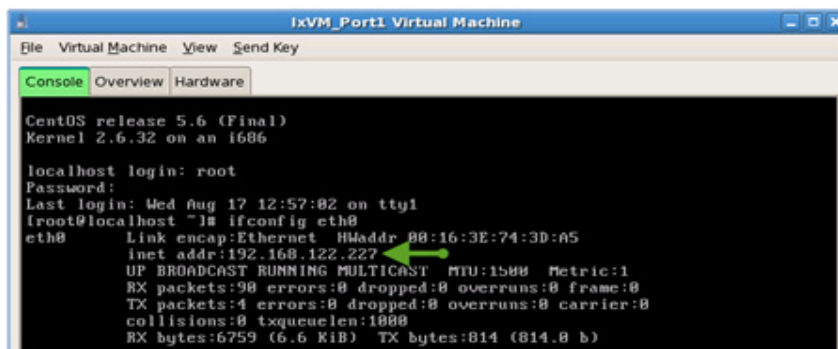
Now that the Windows VM has been successfully established, it's time to bring up the IxVM ports. In the Virtual Machine Manager main GUI, right click on IxVM_Port1 and click on Run (do the same for IxVM_Port2). This will take about 2-5 minutes for the ports to boot up, initialize and obtain an IP address from DHCP.

Figure 39-30. The VM Manager screen



Double-click on IxVM_Port1 (for console); login as root and make a note of the IP address for eth0 (do the same for IxVM_Port2):

Figure 39-31. The console window



Note: At the time of writing, the password for root is ixia123.

Go back into VMM, double-click on the Windows VM and navigate to IxServer -> Tools -> Options. Once you are in the IxServer Options menu, set the NTP Master Server to 192.168.122.1 (the host):

Figure 39-32. The IxServer screen

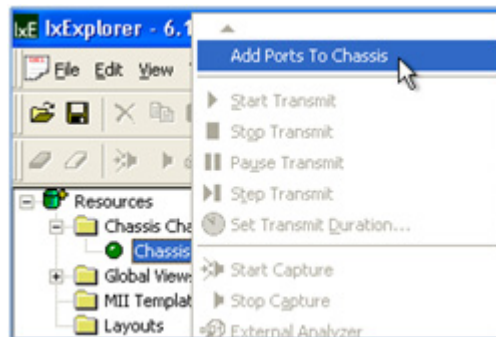


For the changes to take affect; restart IxServer, launch IxExplorer (by connecting to the localhost). Stop the firewall:

```
[root@localhost ~]# /etc/init.d/iptables stop
Flushing firewall rules: [ OK ]
Setting chains to policy ACCEPT: filter [ OK ]
Unloading iptables modules: [ OK ]
[root@localhost ~]#
```

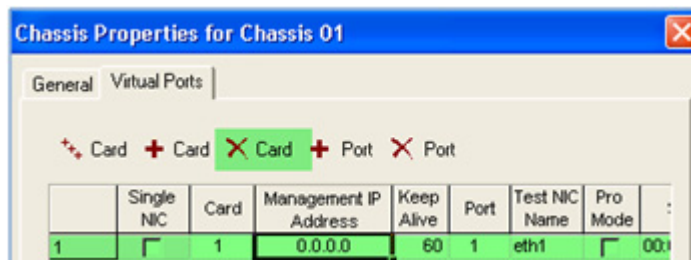
Right click on the chassis (should be in a green state) and click on “Add Ports to Chassis”:

Figure 39-33. The screen showing Add Ports to Chassis option



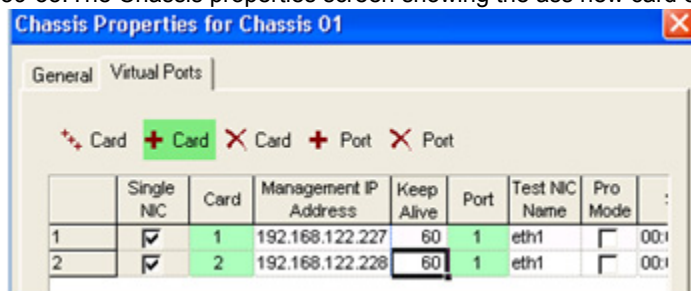
The initial virtual ports dialog contains a default IxVM card/port; delete this by highlighting it and clicking on the following button:

Figure 39-34. The Chassis properties screen



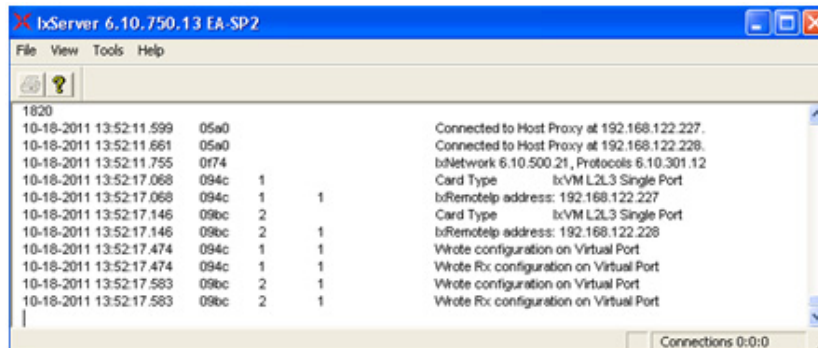
Then add your new cards, apply and OK:

Figure 39-35. The Chassis properties screen showing the add new card option



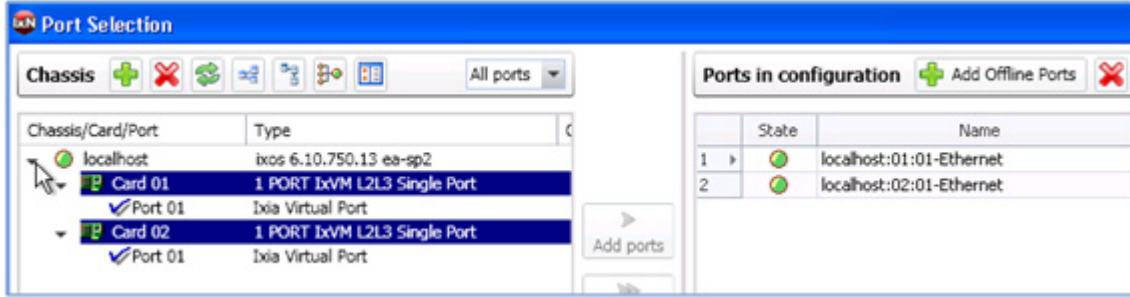
A successful IxVM connection should appear as the following:

Figure 39-36. The successful IxVM connection screen



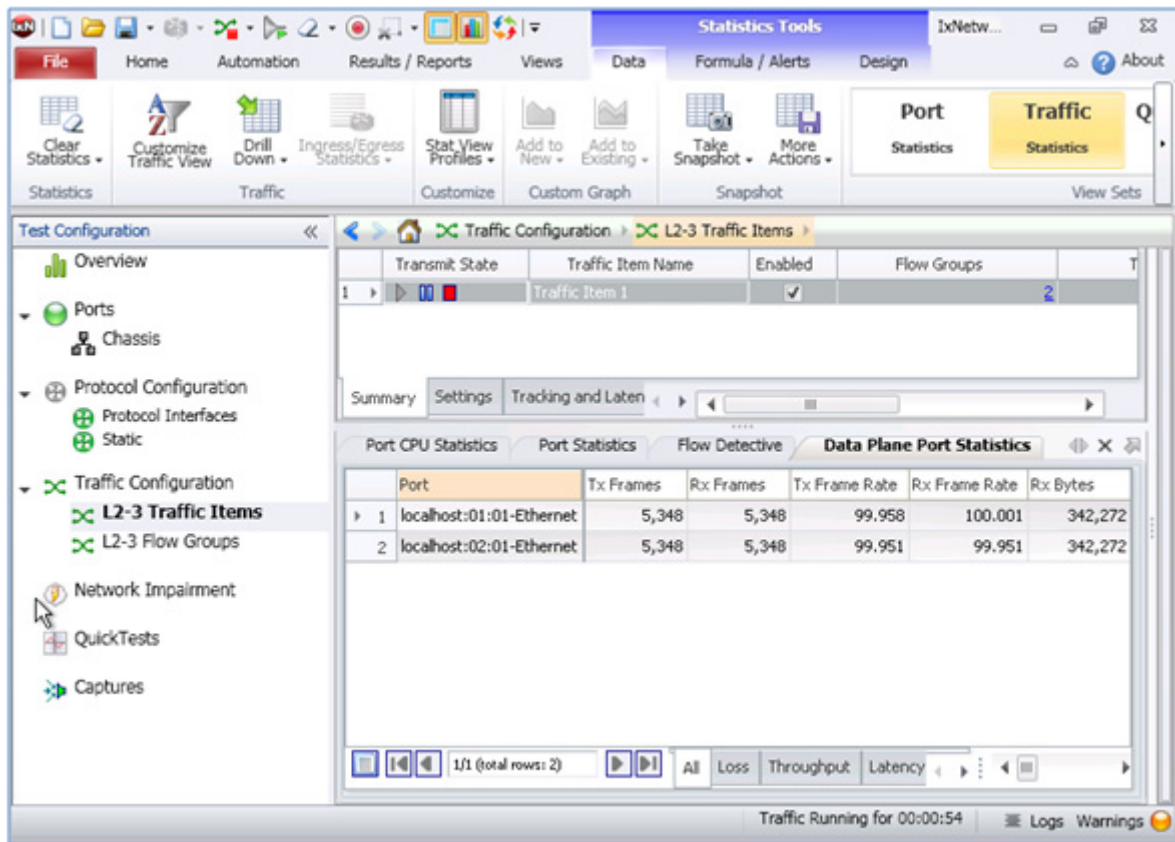
Launch IxNetwork 7.0, use the localhost as the chassis, and add both IxVM ports.

Figure 39-37. The Port Selection screen



Create a back-to-back configuration by adding IPv4/IPv6 protocol interfaces on each port and send some test traffic through at low-rates (for e.g. 100 pps) to verify traffic:

Figure 39-38. The data plane port statistics screen



To look at the raw counters (vnet2 maps to -> IxVM_Port1:eth1 and vnet4 -> IxVM_Port2:et1),

```
[root@localhost ~]# ifconfig vnet2
vnet2      Link encap:Ethernet  HWaddr FE:16:3E:58:F0:42
UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
```

```

RX packets:1561980 errors:0 dropped:0 overruns:0 frame:0
TX packets:1558030 errors:0 dropped:0 overruns:635040
carrier:0

collisions:0 txqueuelen:500

RX bytes:93718728 (89.3 MiB) TX bytes:92541680 (88.2
MiB)

[root@localhost ~]# ifconfig vnet4
vnet4      Link encap:Ethernet  HWaddr FE:16:3E:58:74:0E
UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
RX packets:1882045 errors:0 dropped:0 overruns:0 frame:0
TX packets:1352794 errors:0 dropped:0 overruns:513842
carrier:0

collisions:0 txqueuelen:500

RX bytes:112922556 (107.6 MiB) TX bytes:80227696 (76.5
MiB)

[root@localhost ~]#

```

Use case 2 – External connectivity

Prerequisites

Ensure that you have the following:

- Use-case 1 successfully working and IxVM ports established (back-to-back).
- IxNetwork client installed on your Windows laptop (or an external AppServer).

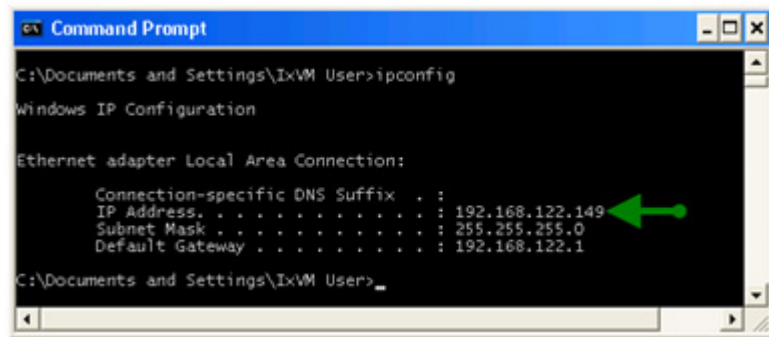
Introduction to xinet.d

In the open source Linux world, xinetd, the eXtended InterNET Daemon is a process-daemon that provides services such as access control, logging, cron services, and port forwarding. Since this service is a fast, secure and efficient means of forwarding traffic from the Windows VM to the external client, xinetd will suffice.

Discover the Windows VM IP

From the VMM GUI, navigate to the Windows VM console, enter the command prompt (Start -> Run -> cmd) and make a note of the IPv4 address that it is assigned (in the following case, the address is 192.168.122.149):

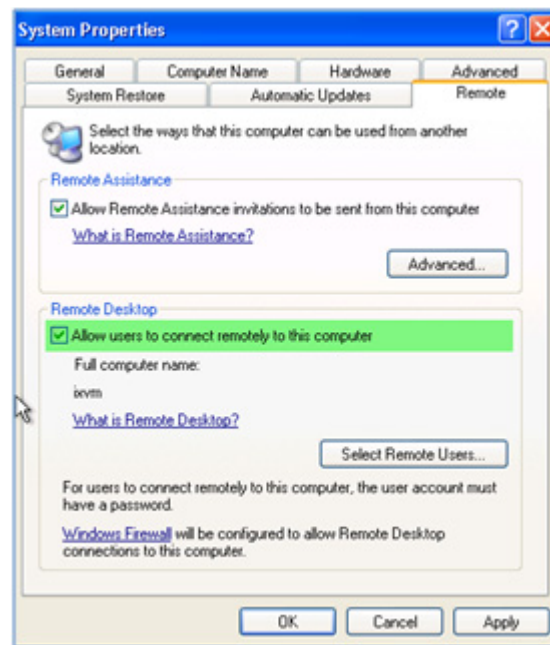
Figure 39-39. The command prompt window



Enable remote desktop for IxVM User

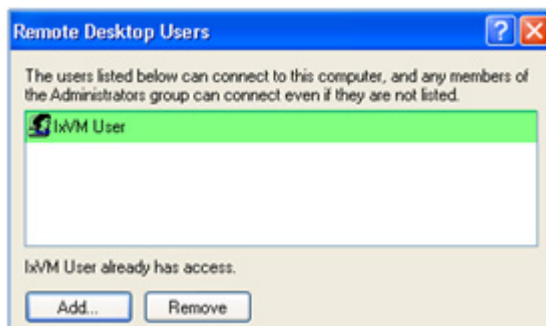
Goto System Properties (from Start -> Run -> right click on “My Computer”), navigate to the “Remote” tab and enable “Allow users to connect remotely to this computer”.

Figure 39-40. The Remote Tab screen



Click on “Select Remote Users...” and add IxVM User:

Figure 39-41. The Remote Desktop Users screen



Install xinetd on the host

```
[root@localhost init.d]# yum install xinetd
```

...

Installed:

```
xinetd.x86_64 2:2.3.14-13.el5
```

Configure common Ixia TCP ports as xinetd services

On the host, edit the `/etc/services` file and at the end of file (under “# Local services”) add the following three entries:

```
ixostcl4555/tcp# IxOS TCL server.
ixnetclient6809/tcp# IxNetwork server.
ixnettc18009/tcp# IxNetwork TCL server.
```

Create port-forwarding definition file for xinetd

Create a file called `ixia` in the `/etc/xinetd.d` directory with the following contents (making a note of `192.168.122.149` as the destination for all services):

```
service ixostcl
{
    flags = REUSE
    socket_type = stream
    wait = no
    user = root
    server = /usr/bin/nc
    server_args = 192.168.122.149 4555
    log_on_failure += USERID
}

service ixnetclient
{
    flags = REUSE
    socket_type = stream
```

```
wait = no
user = root
server = /usr/bin/nc
server_args = 192.168.122.149 6809
log_on_failure += USERID
}
service ixnetctl
{
    flags = REUSE
    socket_type = stream
    wait = no
    user = root
    server = /usr/bin/nc
    server_args = 192.168.122.149 8009
    log_on_failure += USERID
}
service ms-wbt-server
{
    flags = REUSE
    socket_type = stream
    wait = no
    user = root
    server = /usr/bin/nc
    server_args = 192.168.122.149 3389
    log_on_failure += USERID
}
```

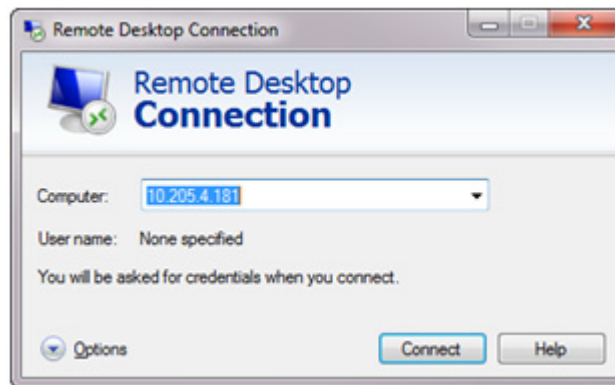
Save and exit the editor, ensure executable permissions for the ixia file, allow xinetd to be automatically restarted on boot-up and manually start the xinetd service:

```
[root@localhost init.d]# chmod +x ixia
[root@localhost init.d]# chkconfig xinetd on
[root@localhost init.d]# service xinetd start
Starting xinetd: [ OK ]
```

Test Microsoft Remote Desktop connection

From your laptop client, connect to your server, for example:

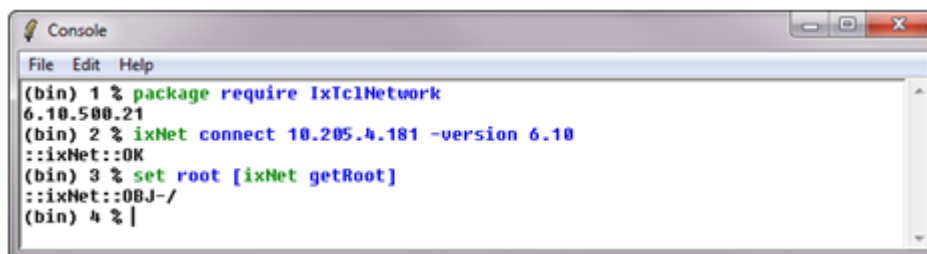
Figure 39-42. Remote Desktop connection screen



Test IxNetwork TCL server

From your laptop client, connect to your server, for example:

Figure 39-43. The console window



```
Console
File Edit Help
(bin) 1 % package require IxTclNetwork
6.10.500.21
(bin) 2 % ixNet connect 10.205.4.181 -version 6.10
::ixNet::OK
(bin) 3 % set root [ixNet getRoot]
::ixNet::OBJ- /
(bin) 4 % |
```

Xen Server Hypervisor Support

The Xen Server installation is very similar to the KVM one.

Hardware requirements

This document focuses on the Ixia Application controller (also known as “AppServer”) and the time of writing, this is a single-processor Intel Xeon (quad core), 24GB of DDR2 RAM, two integrated GbE LAN ports, and one slim VDROM drive.

After installing CentOS 5.6 x64 on the physical machine, install the xen kernel and depending packages (yum install xen virt-manager kernel-xen) and set xend to start at boot (chkconfig xend on). After this, reboot the machine. To check that the host has booted into the xen kernel, use the command „uname -a | grep xen”. If the command returns an output, then the Xen hypervisor can be used.

Xen already provides a bridge that can be used for the management interface (xenbr) and by connecting the virtual machines to this bridge, you should have access to your local management network and DHCP server.

Create an IxVM machine

Run the Xen script to create a virtual machine with all the IxVM components installed already:

```
[root@localhost images]# ./VM_IxVM_XEN-2.0.0.228.sh
Host System: CentOS
Creating virtual machine
Enter virtual machine(domain) name: IxVM_Port1
Setting up network configuration
NETWORKS_LIST
xenbr0
default
ixvm_bridge
Enter management network name: xenbr0
Enter test network name: ixvm_bridge
Stopping libvirtd daemon: [ OK ]
Starting libvirtd daemon: [ OK ]
VIRTUAL MACHINE INFO
Machine name: IxVM_Port1
Management NIC: xenbr0
Test NIC: ixvm_bridge
```

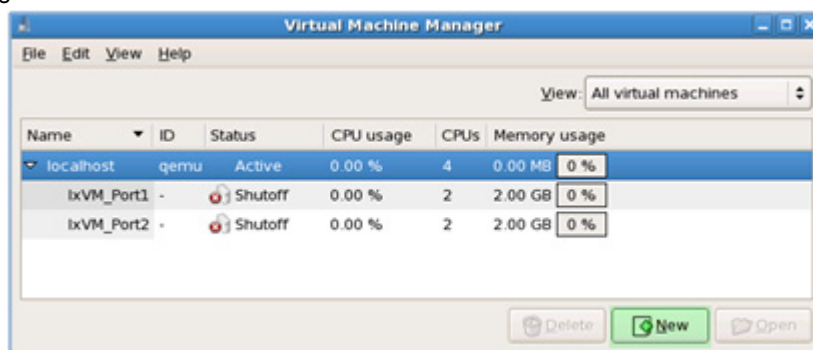
```
#####
# [warn]: libvirt daemon has just been restarted in order
that #
# new configuration to take effect; note that it may be #
# needed to restart Virtual Machine Manager also #
#####
```

Note: Before powering on the virtual machine, make sure that SELinux is not enabled on the host operating system. Edit the `/etc/selinux/conf` file and check that the `SELINUX=` option is set to „disabled”. Reboot the host machine after this change.

Create and instantiate IxVM management VM

Go back to the VMM GUI, highlight the **qemu** connection and click on “New” to create a new VM instance:

Figure 39-44. The VM instance screen



Note: The VM status for both instances is in the Shutoff state and while they are in this state they will not consume any host CPU cycles as it will need all available processing power to complete the following task of instantiating a Windows VM as fast as possible.

Name: IxVM

Virtualization method: Fully virtualized (CPU architecture: x86_x64)

Hypervisor: KVM

Installation method: Local installation media.

OS Type: Windows

OS Variant: Microsoft Windows XP (x86_64)

Installation media: <Select ISO or CD-ROM>

Storage: File (disk image)

Location:/nobackup/libvirt/images/IxVM.img

Size:60000 MB

Allocate:Uncheck (this option will allocate disk space on the fly as needed).

Network:Virtual Network -> Network: default.

Memory

Max memory:4000 MB

Startup mem.:1024 MB

Virtual CPUs:2

Click on “Finish” in the last dialog to create and launch the VM.

***For easy deployment of large setups on the open source Xen Server, Discovery Server, and Chassis Builder, the Deployment Wizard includes support for the Xen Hypervisor.

Discovery Server supports discovering Virtual Load Modules from XenServer by using the following option:

- Manual Discovery
- New plugin for KVM and Xen discovery

Note: For the appliances to be discovered by using the Xen and KVM discovery plugin, the virtual machines need to be created by using the Ixia self-extracting file for the respective hypervisors.

Chassis Builder supports managing appliances from Xen Server when rebuilding the chassis topology. It also offers support for restarting the Virtual Load Modules created on a Xen Server.

Deployment Wizard supports deploying or updating Virtual Load Modules on Xen Server. It has the following functionalities:

- Connects to Xen Hypervisor
- Retrieves datastores
- Retrieves vBridges
- Adds multiple interfaces
- Deploys Virtual Load Modules by using a .sh script
- Upgrades Virtual Load Modules that run on Xen Hypervisors

Xen support in Chassis Builder

When machines are discovered with discovery server and shown in chassis builder, the new type Xen is shown on the type column as shown in the following image:

Figure 39-45. The Type column showing the Xen details

State	Type	NIC	Driver	MAC	IP Address
Physical Name: 10.205.15.212					
Appliance: 10.205.15.212(Xen) IxNetwork:6.10.0.119 IxVM:6.10.0.845					
Availa...	Xen	eth0	virtio_net	00:16:36:39:C7:26	10.205.15.212
Availa...	Xen	eth1	virtio_net	00:16:36:7B:9C:10	
Availa...	Xen	eth2	virtio_net	54:52:00:4B:2F:FA	

In addition, when a new Xen virtual machine is added in the chassis topology, the type of the VM can be seen in the following VM information:

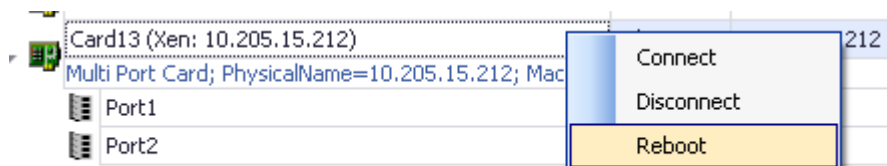
Figure 39-46. The chassis topology details

Name	Interface	IP Address	Owr
127.0.0.1			
Card3		10.205.18.123	
Card8		10.205.18.142	
Card9		10.205.18.115	
Card10		10.205.18.129	
Card12		10.205.16.59	
Card13 (Xen: 10.205.15.212)	eth0	10.205.15.212	
Multi Port Card; PhysicalName=10.205.15.212; Mac=00:16:36:39:C7:26			
Port1	eth1		
Port2	eth2		

We can also restart the Xen virtual machine in a similar way as Qemu, KVM, and VmWare types of machine are restarted.

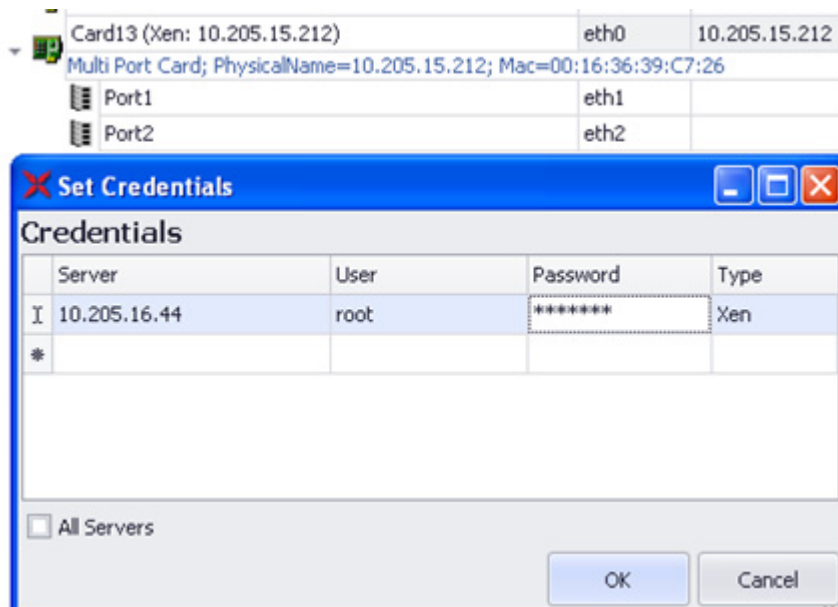
Right-click the machine name, and then click **Reboot** from the shortcut menu. The following image shows the Reboot option:

Figure 39-47. The Reboot option



Set the credentials for the host of the virtual machine, as shown in the following image:

Figure 39-48. The Set Credentials window



The Restart process gets started.

Bare-metal (RPM) Installation

The sequence for a bare-metal installation of IxVM with RPMs is as follows:

1. Download one of the supported Linux OSes, and install it on your bare metal server.
2. Download the IxVM RPMs, and install them on the bare metal server.
3. Install IxServer, Discovery Server, and IxAdmin on a Windows computer.

The following sections describe each of these steps.

After you have completed the steps, you can begin using the virtual Ixia ports with your Ixia testing application.

Installing Linux

Download one of the supported Linux distributions, and install it on your bare metal server. See Requirements (see "Requirements" on page 2) or the release notes for the list of supported Linux distributions.

For whichever distribution you choose, you must use the default (unpatched) kernel.

Installing IxVM Server, IxExplorer, and IxNetwork

For a bare-metal Linux deployment, you install IxServer, IxExplorer, IxNetwork, Discovery Server, and Deployment Wizard on a 32-bit Windows 7 computer that can access the bare-metal server.

- IxServer and IxExplorer are packaged with IxOS and are offered as options during IxOS installation.
- IxNetwork, Discovery Server, and IxAdmin have their own installers, and you install them as you would if you were going to use them with a physical Ixia chassis.

All components can be downloaded from Ixia's website.

To install IxVM Server, IxExplorer, IxNetwork, Discovery Server, and IxAdmin:

1. Download the IxVM IxOS installer, and install IxOS on the Windows computer. Make sure that you select the following two components:
 - IxServer
 - Client
2. Download the IxNetwork installer, and install IxNetwork on the Windows computer. Select all the components, including the one marked IxVM Server (an IxNetwork-specific component required for IxVM, not to be confused with the IxOS IxVM Server component).
3. Download and install Ixia Discovery Server on the Windows computer.
4. Download and install IxAdmin Client on the Windows computer.

Installing the IxVM RPM Packages

Three IxVM components must be installed on the bare metal server:

- IxOS-VM
- IxNetwork-VM
- IxAdminAgent-VM

All three components are supplied as RPM packages compiled for the supported Linux distributions.

To install the IxVM RPM packages:

1. Log on to the bare-metal server under an account with admin privileges.
2. Download the IxVM RPM packages appropriate for the Linux distribution installed on the bare metal server.
3. Type the following to install the IxOS-VM package:

```
rpm -i <ixvm>.rpm
```

where <ixvm> is the name of the IxOS-VM RPM package.
4. Type the following to install the IxNetwork-VM package:

```
rpm -i <ixnetwork>.rpm
```

where <ixnetwork> is the name of the IxNetwork-VM RPM package.
5. Type the following to install the IxAdminAgent-VM package:

```
rpm -i <ixadminagent>.rpm
```

where <ixadminagent> is the name of the IxAdminAgent-VM RPM package.
6. Start the software agent using either of the following methods:
 - Reboot the machine
 - Enter: `/etc/init.d/ixvm start`
7. Enter `ps -e` to verify that `bin/InterfaceManager` is running.
8. After installation, if you need to find the build numbers of the RPMs that are installed, you can use the following commands:
 - IxOS-VM: `rpm -qa ixvm`
 - IxNetwork-VM: `rpm -qa ixnetwork_ixvm`
 - IxAdminAgent-VM: `rpm -qa ixadminagent`

Note: The Ixia Kernel rpm install is only for advanced users and should not be used in Bare-Metal scenarios.

Configuring NTP

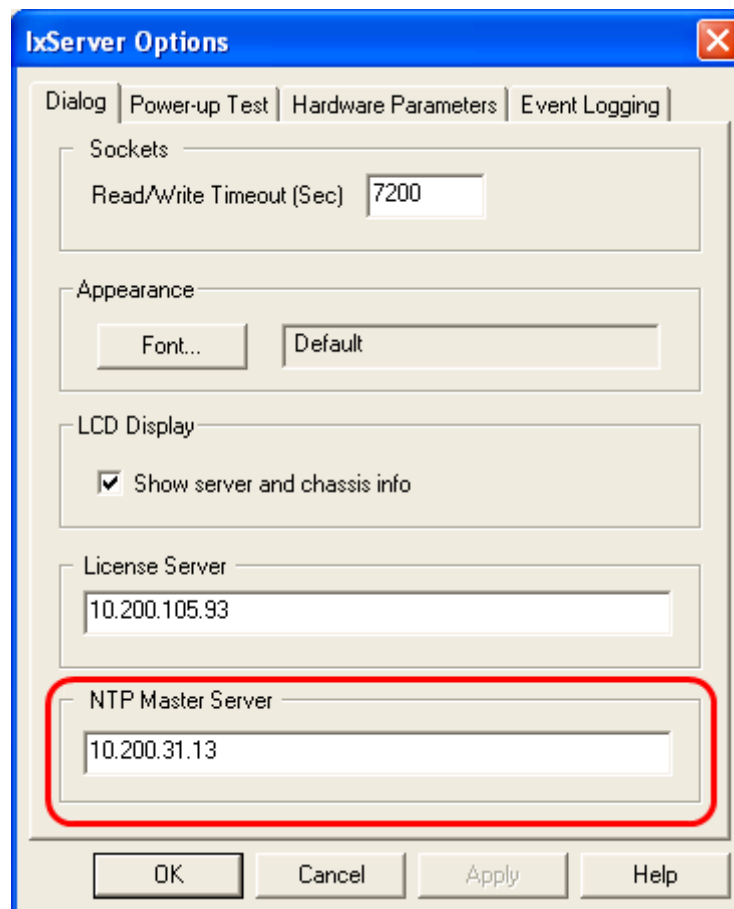
If you are using IxVM ports from different instances of IxVM Server, or you are using a combination of IxVM ports and Ixia hardware ports, you must configure a common NTP time source so that traffic can be synchronized among the ports.

The NTP time source must be configured in IxVM server (for IxVM ports) and in IxOS server (for hardware ports).

To configure an NTP time source:

1. Open IxVM Server or IxServer.
2. On the menu bar, click **TOOLS | OPTIONS | DIALOG**.
3. In the **NTP MASTER SERVER** field, specify the NTP server.
4. Click **OK**.

Figure 39-49. The Dialog tab under IxServer Options



NTP server options for IxVM cards

The IxVM-enabled version of IxServer supports a number of options for using an NTP time server. To configure NTP for IxVM:

1. Display the copy of IxServer running on the IxVM Windows controller host.
2. Click `TOOLS | OPTIONS | DIALOG`.
3. In the NTP Master Server field, enter one of the following:
 - IP address of an NTP server that is reachable by the IxVM cards
 - Hostname of an NTP server (the host name must be resolvable by DNS and reachable by the IxVM cards)
 - 0, to disable the IxVM cards from sourcing NTP through Ixia applications. If you use this option, you must supply a time source to the IxVM cards by some other means.

Single-port vs. Multi-port Cards

There are two types of IxVM virtual load modules: single-port and multi-port.

- Single-port load modules have one interface (virtual port) for generating test traffic.
- Multi-port load modules have multiple interfaces for generating test traffic.

All IxVM virtual load modules require one interface for management traffic, and at least one interface to generate test traffic.

- Single-port modules are virtual appliances operating in a mode that supports the management interface and one test traffic interface. On a single-port card, all the resources are dedicated to a single test port, which can yield higher per-port performance than on a multi-port card (because there is only one test port). On a single-port card, the eth0 interface is the card management interface, and eth1 is the single test traffic interface. With a single-port card, the test traffic and the emulated routing topology traverse a single virtual network.
- Multi-port modules are virtual appliances operating in a mode that supports the management interface and one or more test traffic interfaces. On a multi-port card, the resources are distributed across multiple test ports. On a multi-port card, eth0 interface is the card management interface (same as a single-port card), and eth1 through ethN are the multiple test interfaces. With a multi-port card, the test traffic and the emulated routing topology may traverse multiple virtual networks.

Some test traffic and routing protocols are only supported on single-port cards, while others are supported on either type.

One virtual chassis can control up to 32 virtual cards.

Note: The IxVM test ports can also be TAP interfaces created on the Virtual Machines.

Converting Single-port cards into Multi-port cards

You can convert a single-port card into a multi-port card, or add ports to an existing multi-port card. There are three tasks required for this process:

1. For each test port that you want to add, create an additional test network.
2. Add the additional test ports to the VM card.
3. In the test application (IxExplorer, IxNetwork, or IxLoad), add or discover the ports added to the card.

Step 1. Create the Additional Test Networks

If you are adding ports to a multi-port card, each port should have its own network in vSphere. Use the procedure below to create an additional test network.

To create an additional test network in vSphere:

1. Login to vSphere client.
2. Select the ESX(i) host.
3. Click the CONFIGURATION tab.
4. In the Hardware area, click NETWORKING.
5. Click ADD NETWORKING (upper right).

The Add Network Wizard displays, with the Connection Type set to Virtual Machine.

6. Click NEXT.

The Network Access pane displays.

7. Select CREATE A VIRTUAL SWITCH, then click NEXT.
8. In the NETWORK LABEL field, enter a label for the additional test network, then click NEXT, then click FINISH.

Step 2. Add Ports to the Card

To add ports to an IxVM card, use the procedure below.

To add ports to an IxVM card:

1. Login to vSphere client.
2. Select the VM you want to add ports to.
3. SHUT DOWN or POWER OFF the VM.
4. Select the VM, and then click EDIT VIRTUAL MACHINE on the Getting Started tab.

The Virtual Machine Properties window displays.

5. On the Hardware tab, click ADD.

The Add Hardware wizard displays, with the Device Type pane selected.

6. Select Ethernet Adapter, then click NEXT.

The Network Connection pane displays.

7. In the Adapter Type field, select VMXNET3.
8. In the Network Label field, select the destination test network, then click NEXT, then click FINISH.
9. Repeat steps 4-8 for any additional ports you want to add.
10. Click OK to close the window.
11. Power on the VM.

Step 3. (IxExplorer): Adding a Multi-port Card

In IxExplorer, after adding ports to a card, you must manually add (or re-add) a card to the card list.

To manually add a multi-port card to an IxVM chassis:

1. In vSphere client, select the chassis, click CONSOLE, and login to Windows.
2. Start IxExplorer.
3. Right-click the chassis, and then select PROPERTIES.
4. Select VIRTUAL PORTS.
5. If the card you added ports to is already in the card list, select the card, and remove it.
6. Click the MULTI-ADD CARD (the +++Card) button.
IxExplorer adds the card as a multi-port card (the SINGLE-NIC checkbox is not checked).
7. Select the card, then click ADD PORT. Repeat for each additional port you are want to add.
8. Click OK.

In the chassis/card/port list, the card should now display multiple ports.

Step 4. (IxNetwork): Discovering a Multi- port Card

In IxNetwork, after adding ports to a card, you use Discovery Server to automatically add the card and its ports to the chassis/card/port list.

To discover a card in IxNetwork:

1. After you have added ethernet adapters to the VM, you must rebuild the chassis in IxNetwork, in order for Discovery Server to discover them.
2. In vSphere client, select the chassis, click CONSOLE, and login to Windows.
3. Start IxNetwork.
4. Select TEST CONFIGURATION.
5. Select PORT MANAGER.
6. Click ADD PORTS.
7. If the chassis is in the chassis list, select it. If the chassis is not in the list, add it.

The Add Virtual Chassis window displays.

8. Make sure the PERFORM AUTOMATIC DISCOVERY is checked, then click AUTOMATIC.

IxNetwork triggers Discovery Server to discover IxVM cards. After the discovery process is complete, the card should now display multiple ports in the chassis/card/port list.

Sample Configuration

This section describes the steps in creating a sample IxVM configuration. You can use the steps in this section as a guide in creating your own configuration.

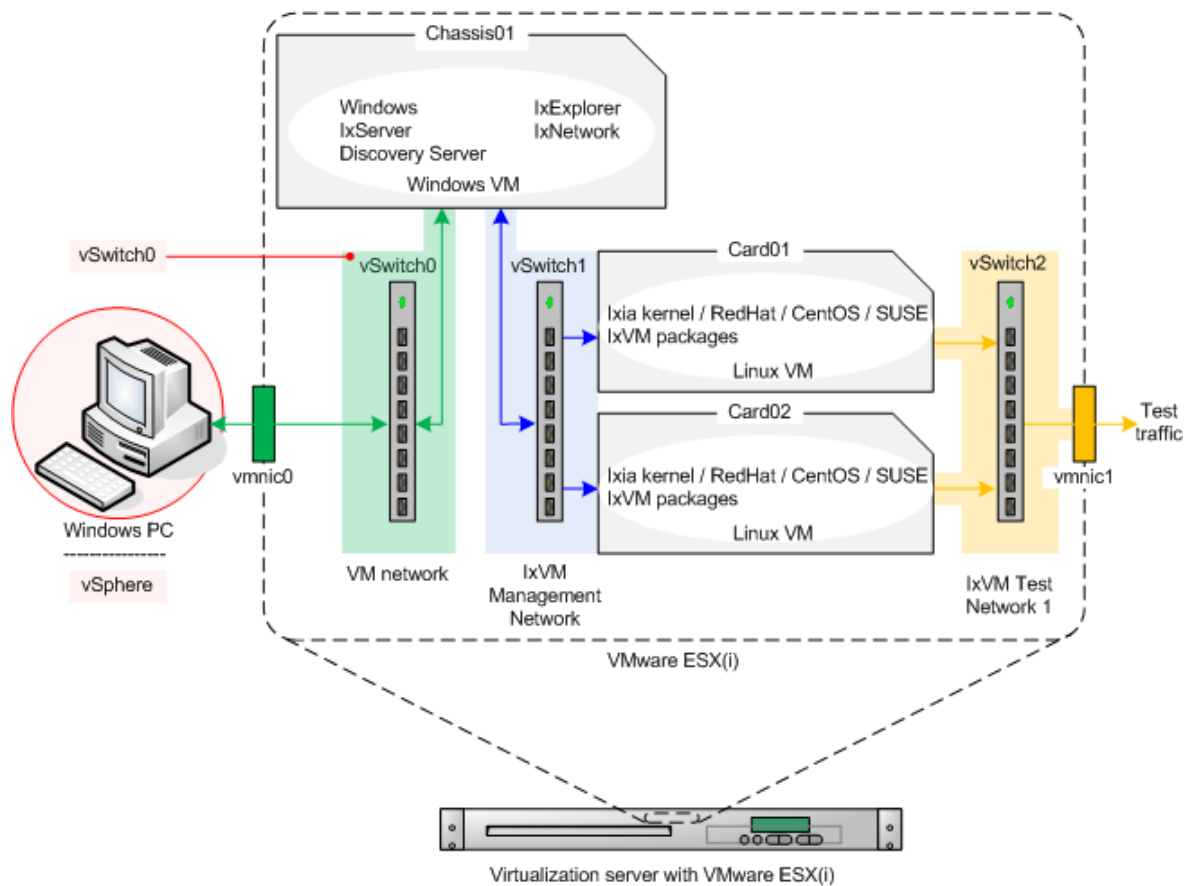
Step 1. Install vSphere

If you have not already installed vSphere, follow the procedure for installing vSphere.

For details see [Downloading and Installing vSphere](#) on page 39-15.

Configuration Details

By default, VMware creates one virtual switch, vSwitch0, which is used to access the ESX(i) server over the corporate LAN. VMware applies the name VM Network to the network served by vSwitch0. In the IxVM Windows VM in this example, this network is labeled Corporate Network.



Step 1. Configure the Source and Destination Networks

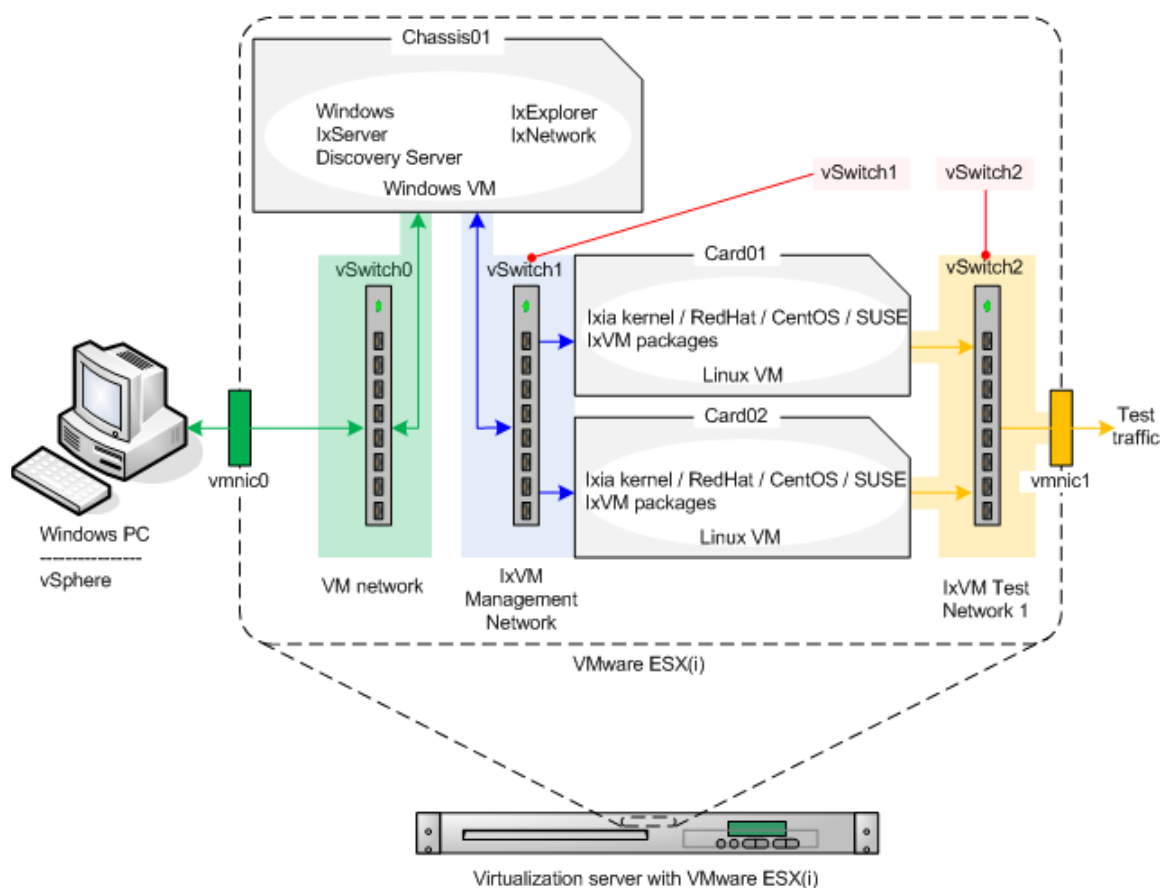
In vSphere, create two virtual switches (vSwitches): one to serve the network that carries the IxVM card management traffic, and one for the network that carries the test traffic. In vSphere, these are named source and destination networks.

For details see [Creating the Source and Destination Networks](#) on page 39-16.

Configuration Details

Name the networks as follows:

- vSwitch1: IxVM Management Network
- vSwitch2: IxVM Test Network 1



Step 3. Deploy a Windows VM

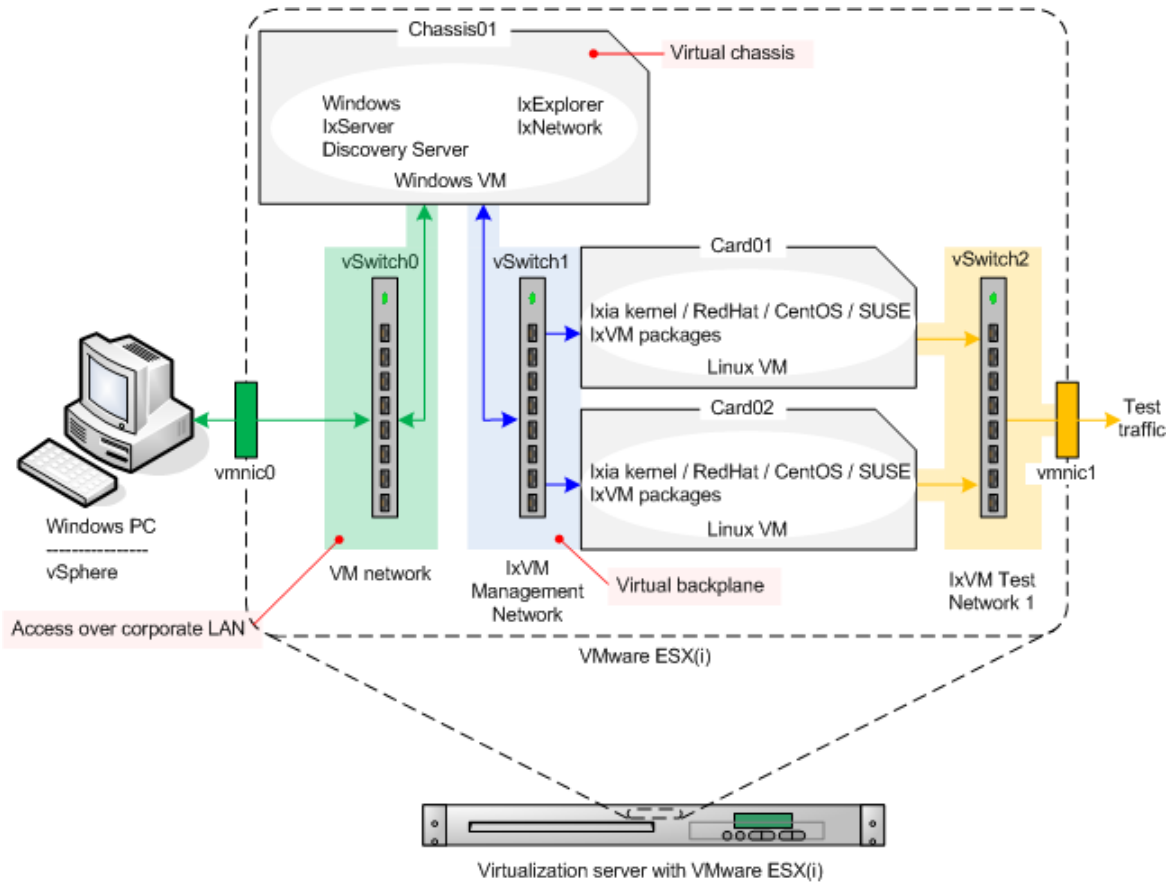
In vSphere, create a Windows VM with two NICs. Refer to the IxOS release notes for the list of Windows versions that IxVM supports.

Configuration Details

- Name the VM Chassis01.

- Create two networks, named and mapped as follows:

vSphere Network	Function
VM Network (default name)	Access to the virtual chassis from the corporate LAN
IxVM Management Network	IxVM card virtual backplane



Step 4. Deploy the Linux OVA Template

Deploy two Linux VMs based on the Ixia kernel OVA.

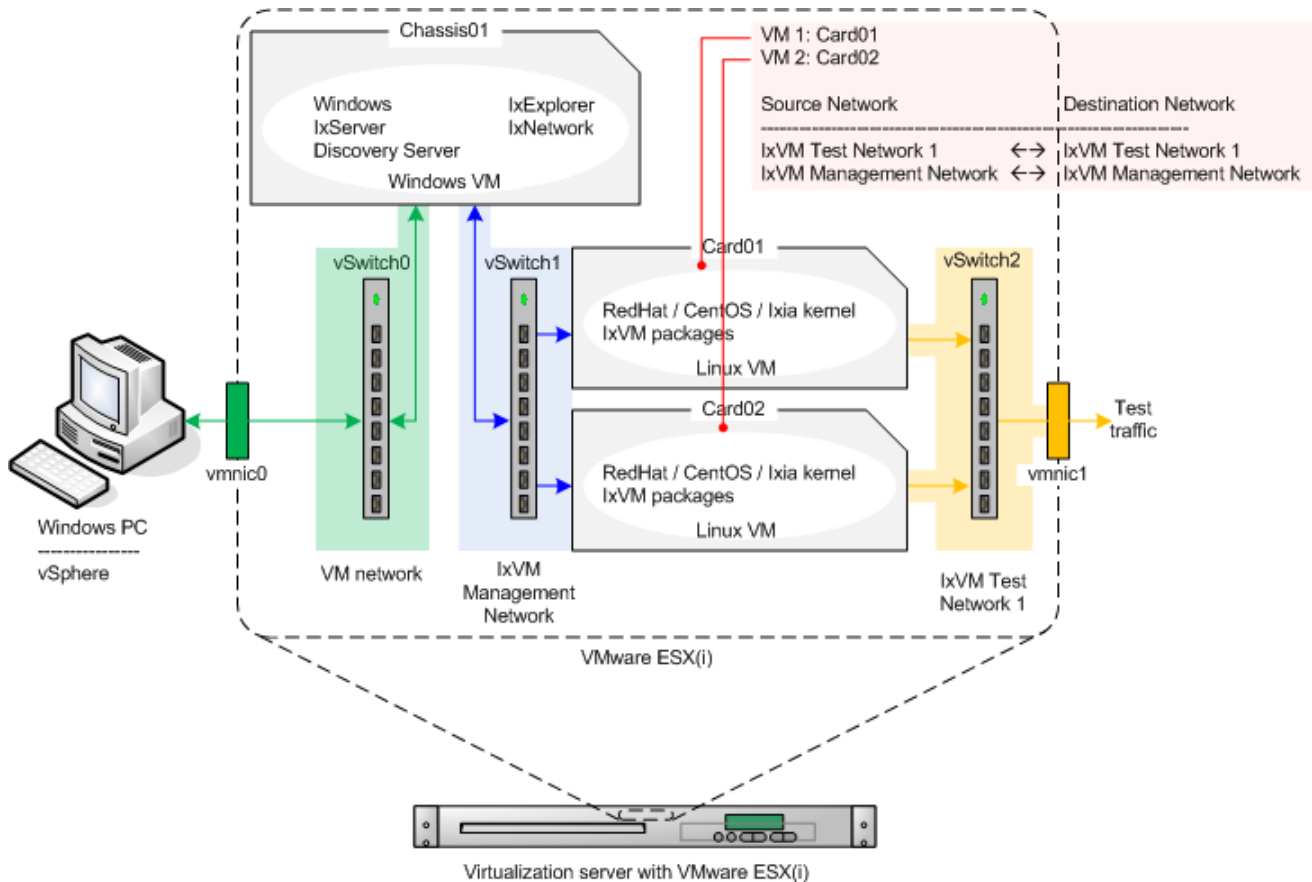
For details see [Deploying the IxVM Appliances \(OVAs\)](#) on page 39-17.

Configuration Details

- Name the first VM Card01
- Name the second VM Card02

- Map the networks as follows:

Source Network	Destination Network
IxVM Test Network 1	IxVM Test Network 1
IxVM Management Network	IxVM Management Network



Step 5. Configure the IxVM card (Linux OVA) Addresses

If you configured the IxVM cards (Card01, Card02, Linux OVAs) to use DHCP addressing, skip this step.

If you configured the IxVM cards to use static addressing, configure their addresses.

For details see [Configuring Static Addressing](#) on page 39-18.

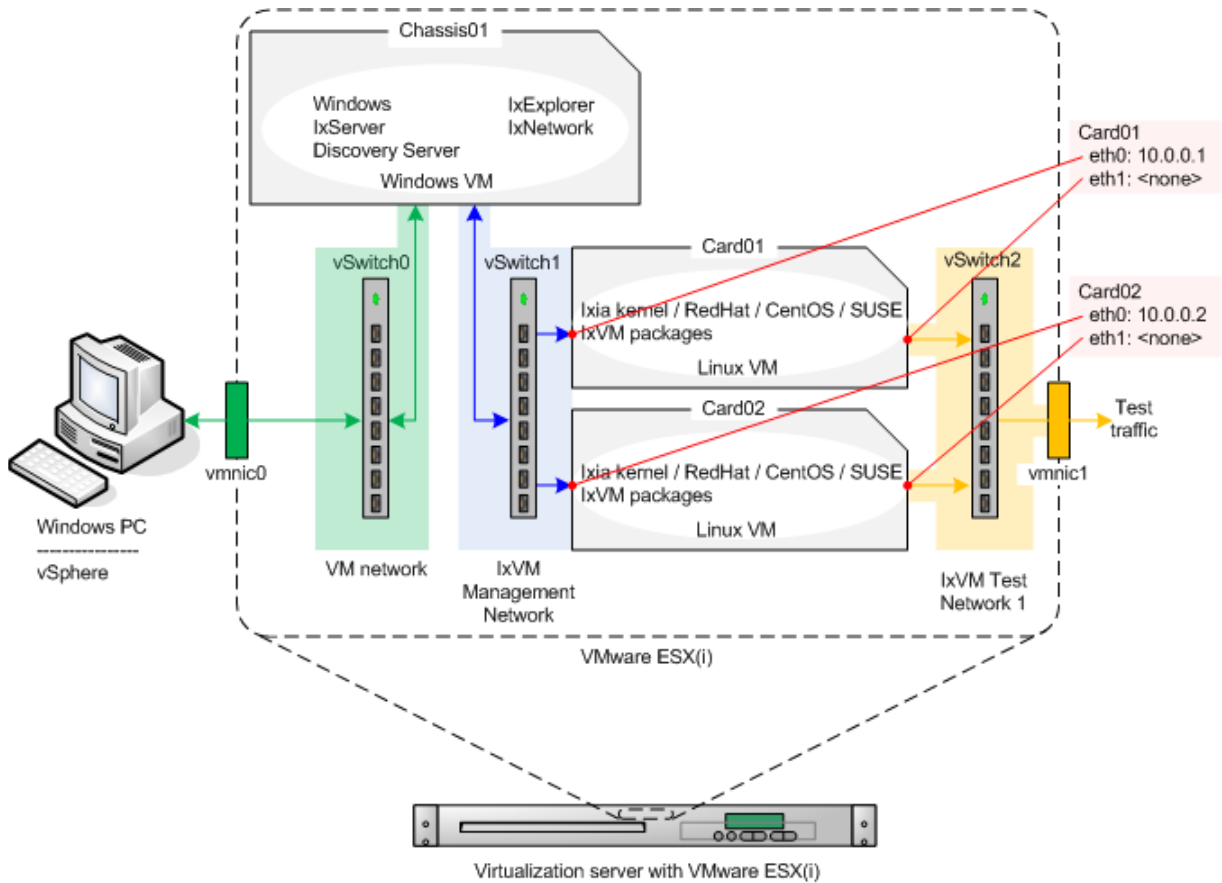
Configuration Details

- Card01 addresses:

eth0: 10.0.0.1
eth1: <no address>

- Card02 addresses:
eth0: 10.0.0.2
eth1: <no address>

Use the console to test connectivity by pinging card02 from card01 (or vice-versa).



Step 6. Configure the IxVM Chassis Addresses

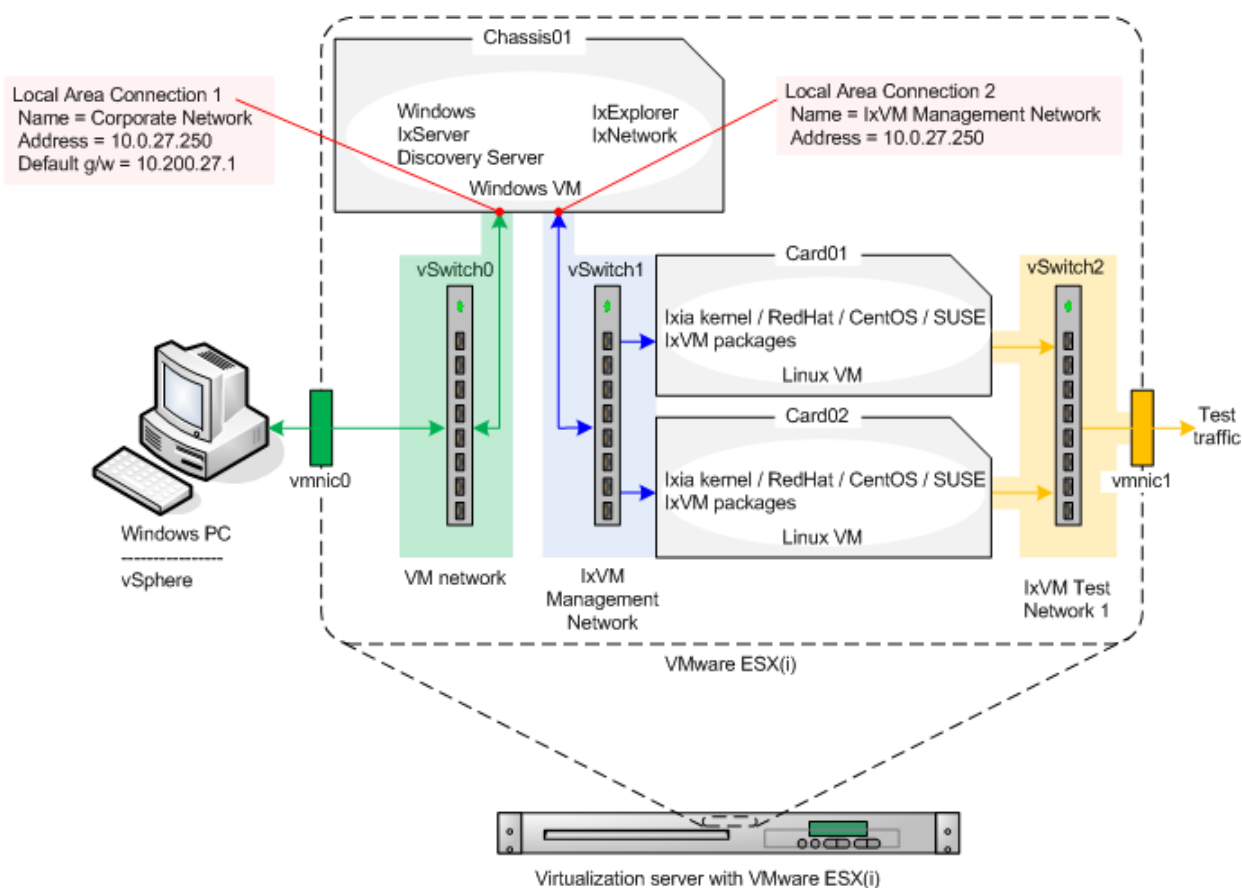
Configure the IxVM chassis controller (the Windows VM), and start IxVM and the supporting services on it.

1. Login to the Windows VM, and go through the procedure to license Windows.
2. Download and install the following components on the IxVM chassis controller.
 - IxVM Server
 - Discovery Server

3. Rename the two local area connections as follows:
 - Local Area Connection 1: VM network
 - Local Area Connection 2: IxVM Management Network
4. If you configured the IxVM chassis controller to use static addressing, configure its addresses as follows:

VM network: 10.200.27.250/24 (Default gateway: 10.200.27.1)

IxVM Management Network = 10.0.0.250/24 (Default gateway: none)



Step 7. Discover the IxVM Cards

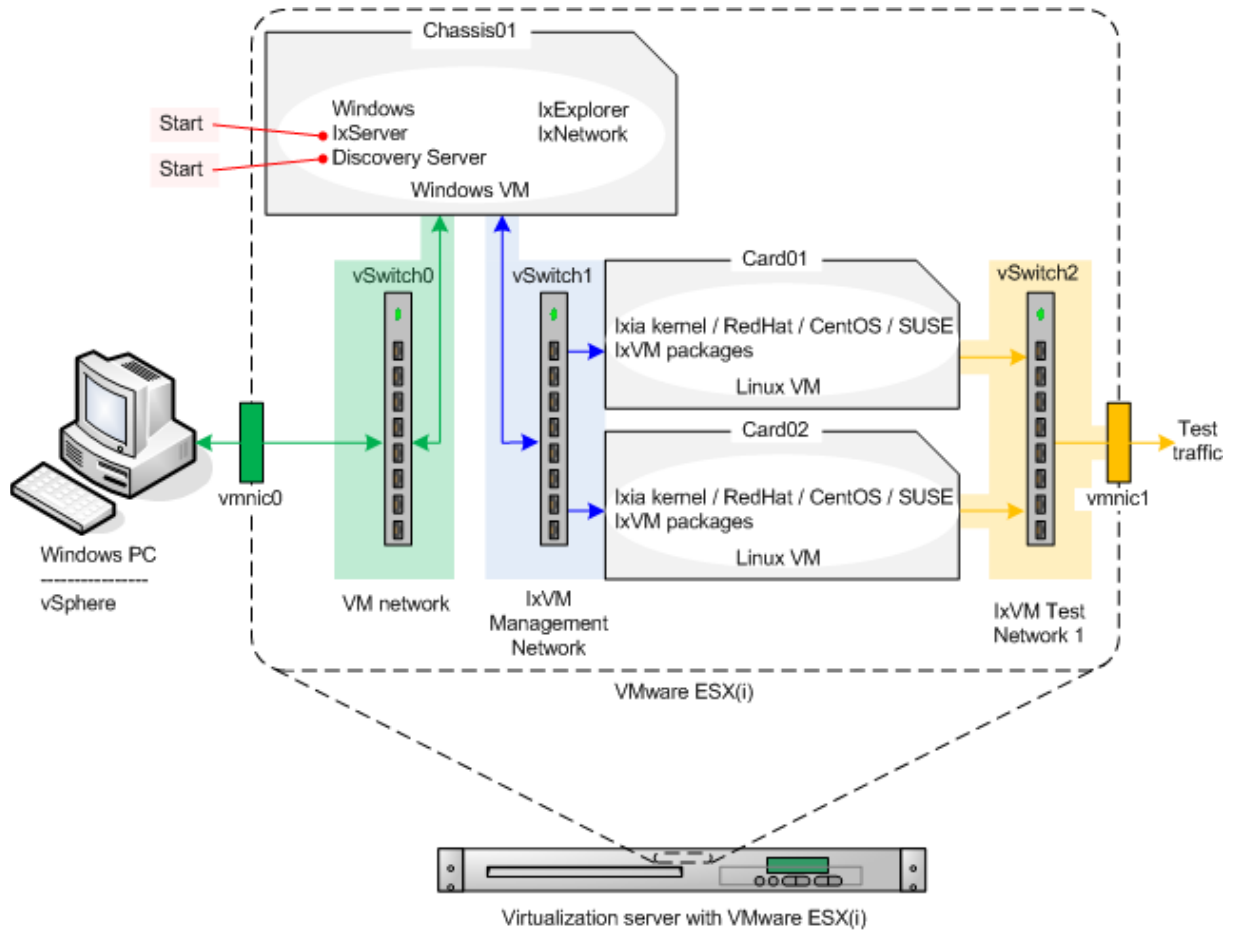
Start the Windows services, and discover the IxVM cards:

1. Start IxServer.
2. Start Discovery Server.
3. In Discovery Server, uncheck CORPORATE NETWORK (10.200.x.x, there are no VMs on this network).
4. Click SERVER | START BROADCAST to start Broadcast Discovery.

5. Click the AUTODISCOVERY tab. This tab should indicate that two endpoints (the IxVM cards) have been discovered.
6. Close the Discovery Server window.

The Discovery Server minimizes and continues to run.

The IxVM cards are ready for use. Start IxExplorer, IxNetwork, or IxLoad and add them to your list of ports.



What to do Next

After you have installed IxVM, you can begin using it. There are two things you need to do to use IxVM with an Ixia testing application:

- Confirm that IxVM supports the protocols that you want to use.
- Find the IxVM virtual load modules running on your test network, and add them to your test configuration.

Supported Protocols

To confirm that IxVM supports the protocols you want to use in a test, check the list of supported protocols in the the test application's user guide.

Finding and Adding IxVM Load Modules

To find and add IxVM load modules, use the Discovery Server. Directions for using Discovery Server are included in the test application's user guide.

Deployment Wizard

As IxVM continues to be adopted by more users, software based test assets will keep on multiplying. The user will take considerable time to deploy large scale systems manually. This could be error prone when port counts reach thousands.

The IxVM Deployment Wizard allows for mass deployment of IxVM software test assets into large scale test environments. It also automates the upgrade of an existing mass deployment to a new version of the software. This facilitates the deployment and upgrade of Windows Virtual Controllers and Virtual Load Modules. You can perform the following actions by using Deployment Wizard:

- Upgrade Virtual Chassis
- Install New Virtual Chassis

Upgrade Virtual Chassis

You can upgrade an existing virtual chassis using the Deployment Wizard. To upgrade an existing chassis, do the following:

1. Open the Deployment Wizard.
2. In the Start Wizard, click Upgrade Virtual Chassis.
3. Click Next. The Chassis Info window appears.
4. In the Hostname/IPbox, type the IP address of the virtual chassis.

Note: You can select the IP address by clicking Previously used. The IP addresses of the chassis that were previously used appears. You can select the IP address of the chassis that you want to upgrade.

Click **Next**. The Source Files dialog box appears.

Click to select the path of each Ixia software installed on the Windows Virtual Controller computer.

Select each Ixia software you want to uninstall.

You can select the following source files:

- IxServer file
- Discovery Server file
- IxNetwork file

Upgrade the load modules by performing the following actions:

Upgrade Appliances

The Upgrade Appliances window shows a list of virtual load modules that you want to update. The following table describes the properties of the virtual load modules:

Field/Control	Description
State	If the virtual load module is available, the check box is selected. In this case, the virtual load module is not assigned to another virtual chassis.
Type	Shows the type of appliance, for example, PC, Qemu, or Vmware.
Appliance	Shows the name of the virtual load module that you want to update.
IP Address	Shows the IP address of the virtual load module.
Info	Shows the list of installed Ixia software on the virtual load module.
NICs No	Shows the number of virtual interfaces that are defined for each virtual load module.

Select Source Files

The Select Source Files window allows you to select the source files to upgrade an existing VM quest OS or IxNetwork to the new version of Ixia software.

1. Click to select the source files needed to upgrade the installed Ixia software (.rpm, .zip).

You can select the following source files:

- IxOS package or .rpm file

- IxNetwork package or .rpm file

Install New Virtual Chassis

You can install a new virtual chassis using the Deployment Wizard. To install a new chassis, do the following:

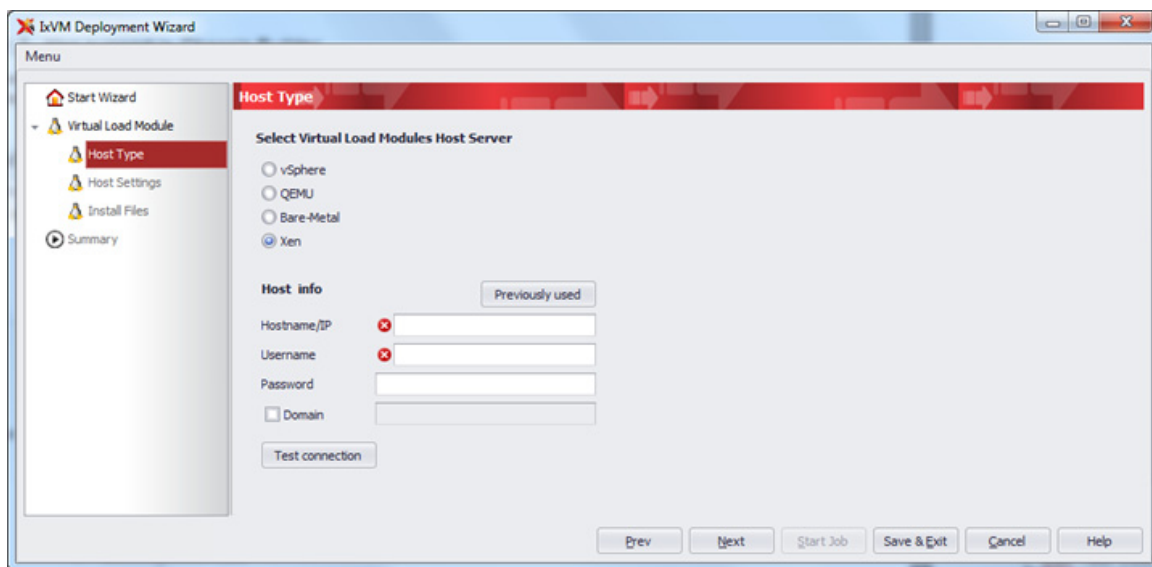
- Configure Windows Virtual Controller Settings
- Configure Virtual Load Module Settings

Xen support in Deployment Wizard

Deployment Wizard offers support for deploying Virtual Load Modules on a Xen host.

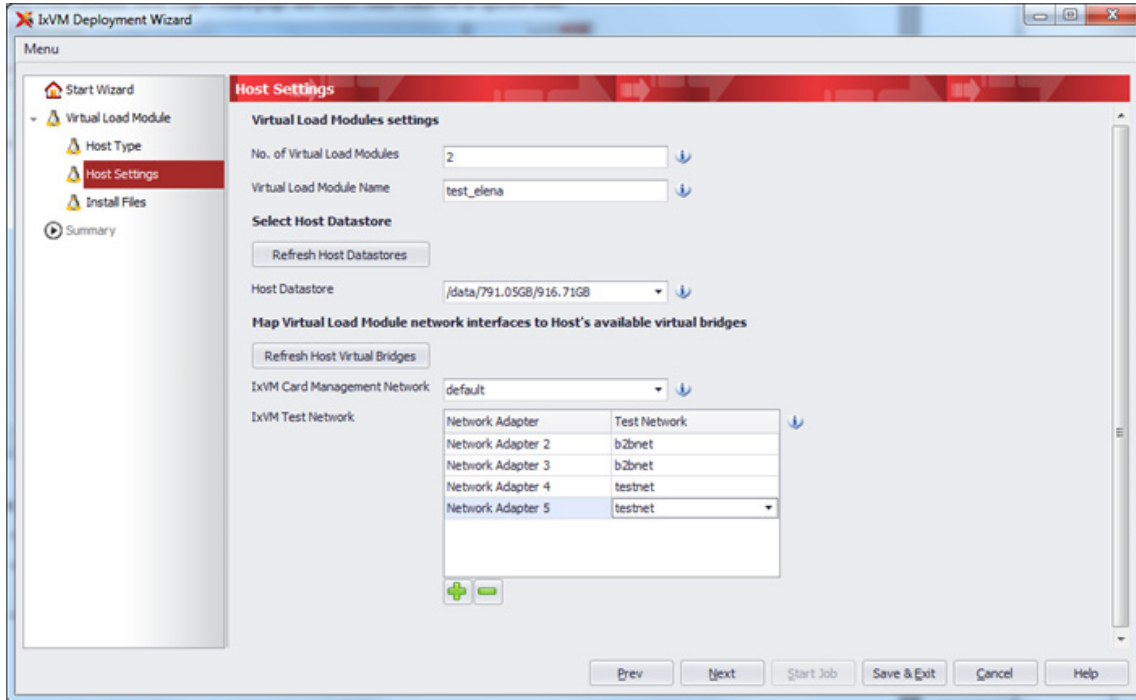
The following image shows the **Host Type** in Deployment Wizard:

Figure 39-50. The Host Type in Deployment Wizard



After testing Xen Host connection with the given credentials, on the next page the Host Settings window shows as follows:

Figure 39-51. The Host Settings window

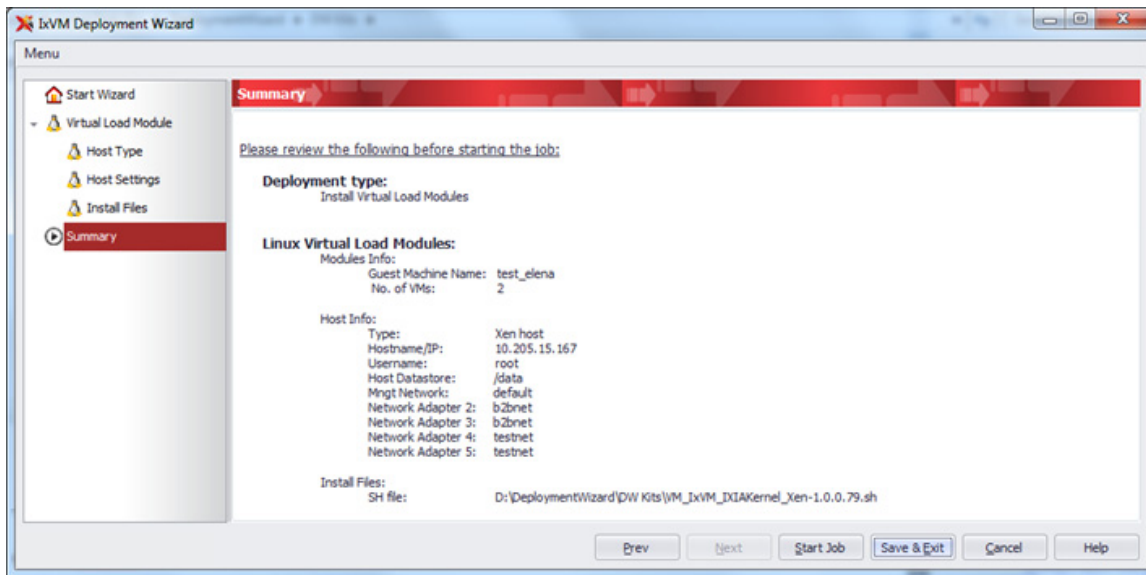


1. Type the **No. of Virtual Load Modules** to be created.
2. Type the **Virtual Load Module Name**.
3. Select the **Host Datastore**.
4. Select the virtual bridges for IxVM Card Management Network and IxVM Test Network.

The self-extracting image (.sh file) used for Virtual Load Module deployed on Xen is different from the Qemu/KVM .sh file (the .sh file name contains the hypervisor name).

The summary page shows the options selected by you on the Install new Virtual Load Module workflow as shown in the following figure:

Figure 39-52. The Virtual Load Module workflow



To deploy a virtual load module manually on a Xen hypervisor you can use the following example:

```
[root@10-205-15-55 OVA]# chmod +x VM_IxVM_XEN-2.0.0.218.sh
```

```
[root@10-205-15-55 OVA]# ./VM_IxVM_XEN-2.0.0.218.sh
```

Configure Windows Virtual Controller Settings

You can configure the settings for Windows Virtual Controller. To configure the settings, do the following:

1. Open the Deployment Wizard.
2. In the **Start Wizard** window, click **Install New Virtual Chassis and Virtual Load Modules**.
3. Click **Next**. The **Chassis Info** window appears.
4. Enter the host details for the existing Windows Virtual Machine or Windows computer. For more information, see [Configure Host Details](#).
5. Click **Next**. The **Source Files** window appears.
6. Click to select the source files needed to install a Virtual Controller on an existing OS.

You can select the following source files:

- IxServer file
- Discovery Server file
- IxNetwork file
- Configure Host Details

The following table describes the host details that you need to enter for an existing Windows Virtual Machine or Windows computer:

Field/Control	Description
Host Info	Click Previously used to select the IP address. The IP addresses of the chassis that were previously installed appears. You can select the IP address of the chassis that you want to install.
IP/Name	Type the IP address of the existing Windows computer where you want to install the virtual chassis.
User Name	Type the name of the existing Windows computer.
Password	Type the password of the existing Windows computer.
Domain	Select the check box next to the Domain box to make it available. You can enter the domain name of the existing Windows computer.
Test Connection	Click Test Connection to validate the input credentials.

Configure Virtual Load Module Settings

You can configure the settings for the characteristics of the Virtual Load Module you create. The characteristics include hypervisor type, virtual machine name, number of virtual machines, datastore and virtual network interfaces assignments. To configure the settings, do the following:

1. Open the Deployment Wizard.
2. Configure the Windows Virtual Controller settings. For more information, see [Configure Windows Virtual Controller Settings](#).
3. Click **Host Type**. The **Host Type** window appears.
4. Enter the host details for the Virtual Load Module. For more information, see [Configure Host Details](#).
5. Click **Next**. The **Host Settings** window appears.
6. Configure the host settings. For more information, see [Configure Host Settings](#).
7. Click **Next**. The **Source Files** window appears.
8. Click to select the source files needed to install a new guest OS (.ova, .sh).
9. Click **Start** to run the upgrade or installation process.

Configure Host Details

The following table describes the host details for the Virtual Load module:

Field/Control	Description
Select Virtual Load Modules Host Server	Click the relevant host OS information: Options include the following: <ul style="list-style-type: none"> • Sphere • QEMU • Bare-Metal
Find Previous	Click to find the information of hosts that you have used previously. Selecting a host from the list automatically shows the following host information: <ul style="list-style-type: none"> • IP/Name • User Name • Password
IP/Name	Type the IP address of the host server.
User Name	Click to find the host network.
Password	Type the password of the host server.
Domain	Select the check box to include the domain to which the host server belongs.
Test Connection	Click Test Connection to validate the input credentials.

Configure Host Settings

The following table describes the host settings that you need to configure for the Virtual Load Module:

Field/Control	Description
Guest Machine Name	Enter the desired name for the guest machines based on which it will automatically increment for new installation selections. Note: The name of the guest machine will be followed by an incremental number from 1 to the Number of Guests per Host.
Number of Guests per Host	Enter the numbers of guests you need to create on the host. The maximum number of guests per host is 32.
Find Host Datastores	Click to show the available datastores for the selected hypervisor.
Host Datastore	Click the list to select the data store from the ones available on the host.

Field/Control	Description
Find Host Networks	Click to show the available virtual switches/bridges for the selected hypervisor.
IxVM Card Management Network	Click to select the Card Management Network information.
IxVM Test Network	Click to select the test network information for both Linux and Bare metal sources.

Jobs Manager

Jobs Manager is a component of IxVM's Deployment Wizard and it shows a multi-level progress log and thermometers indicating relative progress as the virtual chassis are built or upgraded.

Jobs Manager is a Windows application that appears in the notification area. It gets started on the host on startup and is used for the following activities:

- To obtain feedback on the progress and the status of jobs that are running and the previously run jobs.
- To monitor the IxVM JobInstaller service status and to start the service.
- To show the list of current jobs with their statuses and to allow you to retry, continue, or abort a job.

The Jobs Manager notification area contains the following icons as the status of the IxVM JobInstaller service or the status of the current job that is being processed:

Figure 39-53. The Jobs Manager icons in the notification area



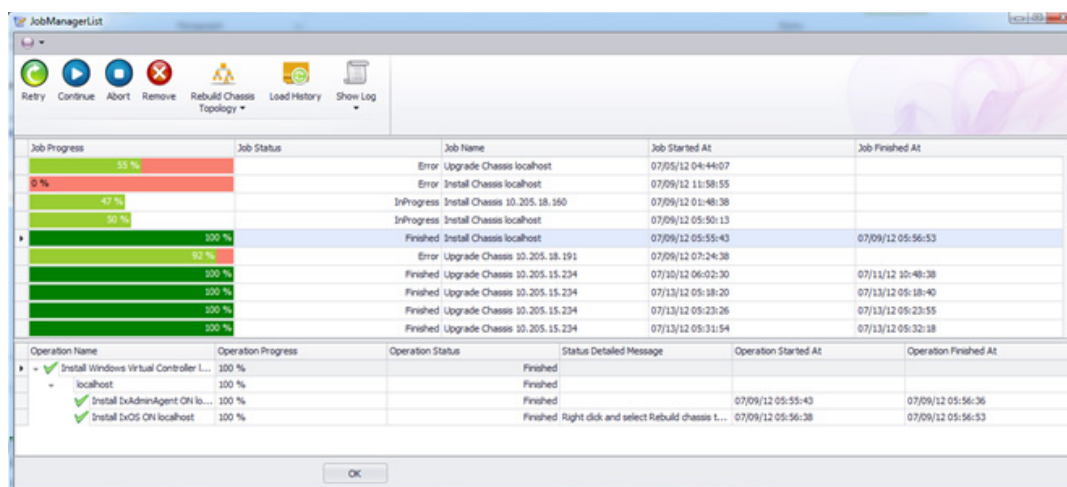
In addition to this, a notification balloon appears for the following most important operations that take place while using Jobs Manager:

- A new job was added to the job queue
- An in progress job returned an error
- Waiting for manual intervention
- A job has finished
- Chassis rebuild has finished

The Continue, Retry, Abort, Remove, and Rebuild chassis commands are available or unavailable depending on the selected job's current status. For example, you can only rebuild the chassis if the jobs have finished without errors. The Show log command refers to the currently selected job and creates a .csv file that shows the message exchange that took place between IxVM's JobInstaller service and the computer that is upgraded or built.

The following image shows the current progress of jobs:

Figure 39-54. The The Jobs Manager List Window



Troubleshooting and Tips

Following are some of the troubleshooting tips:

- **IxNetwork fails to connect to Virtual Chassis (Retrying...)**

Symptoms: IxServer is running and VM is IP reachable.

Attempts to restart Hardware Manager still do not resolve the issue.

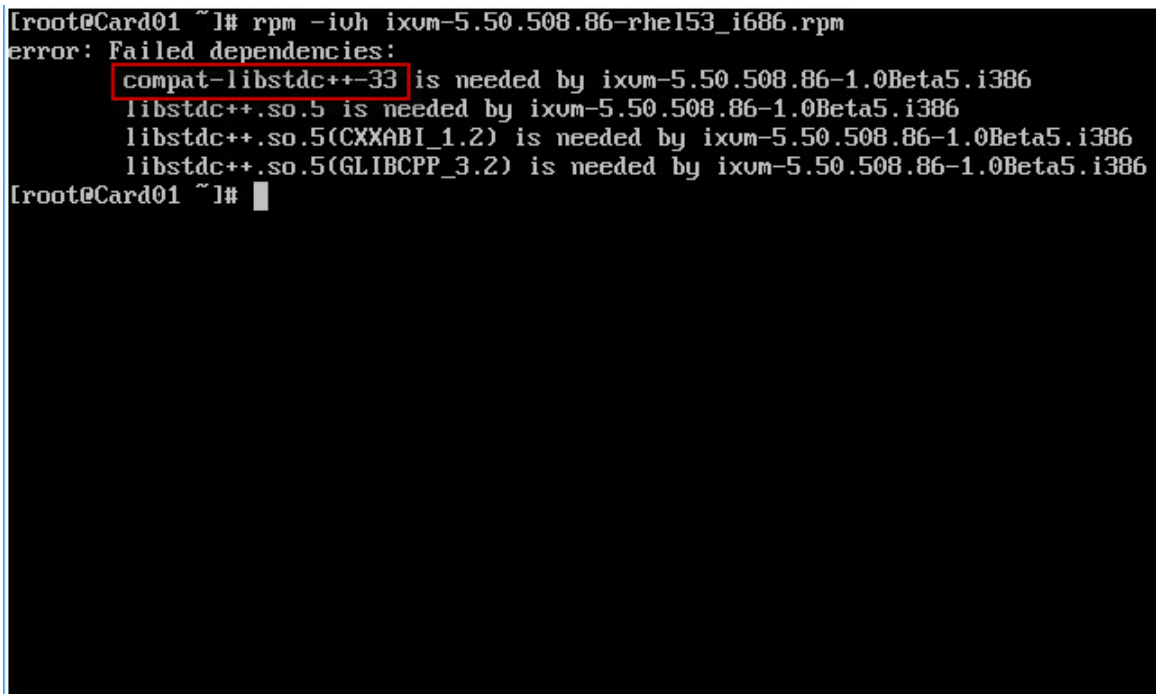
Verify that the IP Address present in the file C:\Program Files\Ixia\IxOS\

- **IxVM RPM fails to install on Bare-Metal RHEL or CentOS machine**

Symptoms: rpm -ivh command fails with error messages indicating that dependencies are missing.

Before you install the IxVM RPMs, download and install the compat-libstdc++-33-* RPM that matches your version of Linux.

Figure 39-55. The console window



```
[root@Card01 ~]# rpm -ivh ixum-5.50.508.86-rhel53_i686.rpm
error: Failed dependencies:
  compat-libstdc++-33 is needed by ixum-5.50.508.86-1.0Beta5.i386
  libstdc++.so.5 is needed by ixum-5.50.508.86-1.0Beta5.i386
  libstdc++.so.5(CXXABI_1.2) is needed by ixum-5.50.508.86-1.0Beta5.i386
  libstdc++.so.5(GLIBC_3.2) is needed by ixum-5.50.508.86-1.0Beta5.i386
[root@Card01 ~]#
```

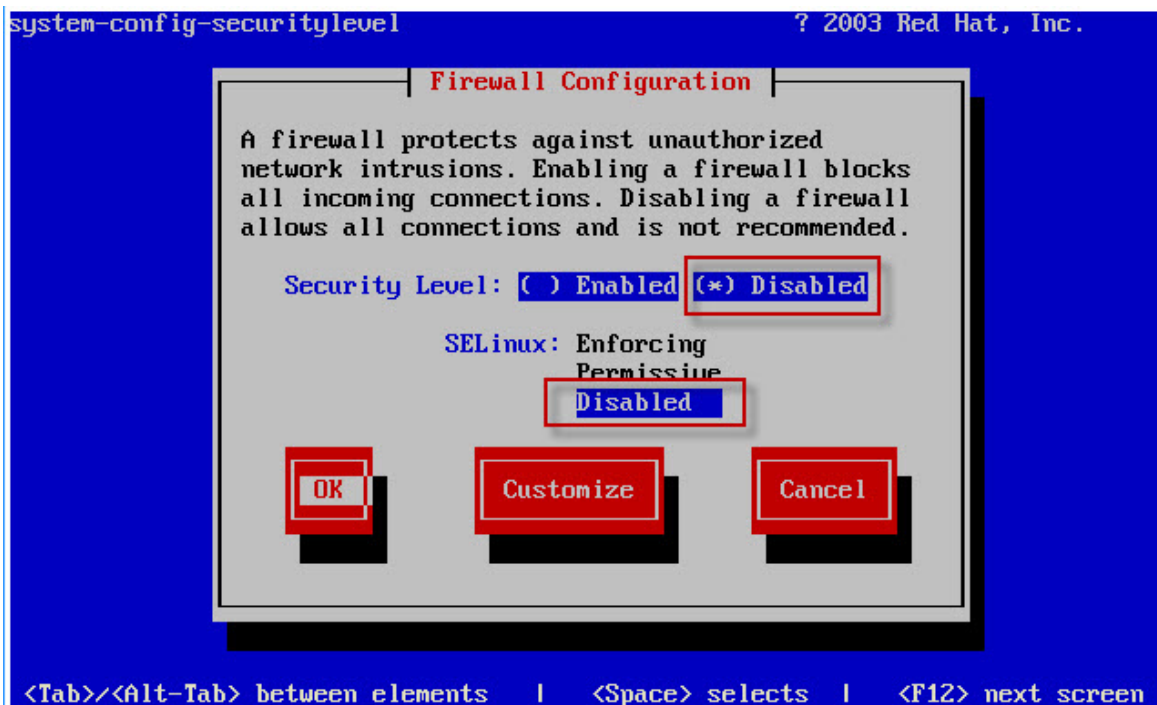
- **Management network connectivity issues in a Bare-Metal Linux Installation**

Try using the following settings:

Firewall Configuration -> Security Level, try using Disabled

Firewall Configuration -> SELinux, try using Disabled

Figure 39-56. The Firewall Configuration screen



- **Statistics De-synchronization and Clock Synchronization**

Symptoms: Stats in the IxNetwork stat views are intermittently vanishing and re-appearing.

VM-VM scenarios

Ensure that all of your VMs are synchronized to the same NTP server, by calling "ntpq -p" on the linux console, and ensure that the server listed is the name/IP address of the system running IxServer VM.

If your VMs which are connected to one instance of IxServer are not synchronized to the same time source, then you may have a third party application updating your local NTP time. See below as to how to customize your NTP Server configuration.

In order to customize the NTP Server being used to synchronize each VM, you can specify the NTP Server used to synchronize all VMs connected to an instance of IxServer with the "NTPServerLocation" registry string, located in the "HKEY_LOCAL_MACHINE\Software\Ix Communications\IxServer\Debug" registry key.

Set to the IP Address of a valid NTP Server.

Set to "0" to prevent IxVM from managing any synchronization with NTP (if you would like to manage it for each VM on your own).

In case you have multiple instances of IxServer VM managing all of your VMs, set the value on all machines running IxServer VM to the same IP address.

VM-HW scenarios

In order to synchronize IxVM cards between VMs and Ixia HW, the following steps are needed.

The NTPServerLocation registry key (see 10.1.2.2) needs to be set to an external NTP server based off of UTC time (see pool.ntp.org for more information) and the VMs need external internet connectivity to be able to communicate with those IP addresses.

The hardware-based ports need to be synchronized to use an AFD1 GPS server as their time source.

- **One Way Latency Measurement**

Symptoms: User sees negative min/max/average latency in the IxNetwork traffic flow views, and IxExplorer's latency view.

When running traffic between multiple different VM cards, IxVM is using software to synchronize the VM cards with each other, which does not give enough precision and leads to one VM card having a local time ahead of the other VM card. Inter Arrival time does not have this issue.

- **Cannot run Control Plane Traffic over MPLS**

In IxNetwork, one of the the default settings for MPLS (LDP) is to run Control Plane traffic over MPLS. The RedHat/CentOS kernel does not support this; the Ixia kernel does.

Tips

- IxVM virtual ports cannot be attached to multiple IxVM Servers. You should disconnect the virtual port from one server, reboot the virtual port's host OS and reconnect to the port from the other server.
- Latency measurement in software is low-precision. You should expect to see typical accuracy of tens of milliseconds, and will occasionally see negative latency values.
- Latency statistics are only available from one IxExplorer or TCL client per port. Two IxExplorer users (or an IxExplorer user and a TCL client session) cannot access latency stats simultaneously for the same virtual port.
- The IxVM Server must be on the same network as the virtual ports – no firewalls or NAT devices should be placed between the IxVM Server and virtual ports.
- Firewalls and SELinux policies should be disabled on the virtual port OS.Active iptables firewall on host.
- In order for some protocols like DHCP, PPP, L2TP, that require unique identification based on the MAC address, the ports should be added to the chassis with promiscuous mode enabled.

Active iptables firewall on host

IxServer and IxVM communicates through TCP or UDP ports, and if there is a firewall on the host computer, you need to add some additional rules to allow traffic from IxServer.

If the Linux VM communicates with the host through a bridge, the iptables firewall should not affect the communication between the Windows VM and Linux VM.

If the traffic between the Windows VM and Linux VM is blocked, following are some examples of how to add new rules so that IxVM works properly.

The examples show how to allow the traffic for the udp port 123, which is required for NTP.

```
iptables -A INPUT -p udp --dport 123 -j ACCEPT
```

When there is no more need to run IxVM, you can delete this rule by typing the following iptables command:

```
iptables -D INPUT -p udp --dport 123 -j ACCEPT
```

The iptables rules order is important because the first rule has precedence. If one iptable rule matches the incoming traffic, the rest of them are ignored. Hence, the iptable rule that accepts the traffic should be added before the iptable rule that drops or rejects the traffic.

For viewing the iptable filtering rules, use the following command:

```
iptables -nvL.
```

The following output represents an example output of the iptables `-nvL` command:

Figure 39-57. The example output of the iptables `-nvL` command

<i>pkts</i>	<i>bytes</i>	<i>target</i>	<i>prot</i>	<i>opt</i>	<i>in</i>	<i>out</i>	<i>source</i>	<i>destination</i>	
0	0	ACCEPT	udp	--	eth0	*	0.0.0.0/0	0.0.0.0/0	udp dpt:67
0	0	ACCEPT	tcp	--	eth0	*	0.0.0.0/0	0.0.0.0/0	tcp dpt:67
10	999	REJECT	all	--	*	*	0.0.0.0/0	0.0.0.0/0	reject-with icmp-port-unreachable

This output shows that there are two ports and protocols that are accepted by the iptable firewall and the last rule shows that everything else is rejected.

If you want to add another iptable rule to accept additional traffic, you need to add it before the REJECT rule.

The following command adds the rule to accept the NTP traffic before the reject rule.

```
iptables -I INPUT 1 -p udp --dport 123 -j ACCEPT
```

The preceding command adds the iptable rule on the first position. You can specify the position of the iptable rule by modifying the parameter.

```
iptables -I INPUT x ... (x represents the rule number)
```

To delete such a rule, the rule number must be specified:

```
iptables -D INPUT x (x-represents the rule number)
```

You should add the same rules for all the required ports. All the ports that are needed for communication between Windows VM and Linux VM are listed in the 'TCP/UDP Ports Required' section.

In addition, the firewall should allow the outgoing traffic from the Linux VMs. One way for doing this is to allow all the traffic that comes from the Linux VMs.

```
iptables -I OUTPUT 1 -p tcp --src a.b.c.d -j ACCEPT,  
where a.b.c.d is the Linux VM ip.
```

Note that this ip should be internet-routable.

TclAPI Support

This section describes the IxOS TclAPI commands and statistics that you can use with an IxVM virtual load module.

In this section:

[*IxOS Tcl API Commands*](#)

[*IxOS Statistics*](#)

IxOS Tcl API Commands

You can use the following IxOS TclAPI commands with IxVM virtual load modules:

- arp
- arpServer
- autoDetectInstrumentation
- byte2IpAddr
- card
- chassis
- chassisChain
- cleanUp
- clearAllMyOwnership
- dectohex
- errorMsg
- filter
- filterPalette
- hextodec
- host2addr
- ip
- ipAddressTable
- ixCheckLinkState
- ixCheckOwnership
- ixCheckTransmitDone
- ixClearArpTable
- ixClearOwnership
- ixClearPacketGroups
- ixClearPerStreamStats

- ixClearPortPacketGroups
- ixClearScheduledTransmitTime
- ixClearStats
- ixClearTimeStamp
- ixCollectStats
- ixConnectToChassis
- ixConnectToTclServer
- ixDisconnectFromChassis
- ixDisconnectTclServer
- ixEnableArpResponse
- ixErrorInfo
- ixGetChassisID
- ixGlobalSetDefault
- ixInitialize
- ixLogin
- ixLogout
- ixPortClearOwnership
- ixPortTakeOwnership
- ixPuts
- ixRequestStats
- ixSetAdvancedStreamSchedulerMode
- ixSetPacketStreamMode
- ixSetPortPacketGroupMode
- ixSetPortPacketStreamMode
- ixSetScheduledTransmitTime
- ixSource
- ixStartPacketGroups
- ixStartPortPacketGroups
- ixStartPortTransmit
- ixStartTransmit
- ixStopPacketGroups
- ixStopPortPacketGroups
- ixStopPortTransmit
- ixStopTransmit
- ixTakeOwnership

- ixTransmitArpRequest
- ixWriteConfigToHardware
- ixWritePortsToHardware
- logMsg
- logOff
- logOn
- map
- mpexpr
- packetGroup
- port
- portGroup
- protocol
- showCmd
- stat
- stream
- streamTransmitStats
- tableUdf
- tableUdfColumn
- tcp
- udf (udf set 2/3 not supported)
- user
- version
- vlan

IxOS Statistics

You can use the following IxOS statistics IxVM virtual load modules:

Note: Some statistics may only work if the protocol or feature they measure is enabled. For example, `bgpTotalSessions` only returns a valid value if BGP is enabled. This is the same behavior as for physical load module.

- asynchronousFramesSent
- bfdAutoConfiguredSessionsUp
- bfdRoutersConfigured
- bfdRoutersRunning
- bfdSessionFlap

- bfdSessionsAutoConfigured
- bfdSessionsConfigured
- bfdSessionsUp
- bgpSessionFlap
- bgpTotalSessions
- bgpTotalSessionsEstablished
- bitsReceived
- bitsReceivedSStream
- bitsSent
- bitsSentSStream
- bytesReceived
- bytesReceivedSStream
- bytesSent
- bytesSentSStream
- captureFilter
- captureTemperature
- cfmBridgesConfigured
- cfmBridgesRunning
- cfmMasConfigured
- cfmMasRunning
- cfmMepsConfigured
- cfmMepsRunning
- cfmRemoteMepsLearned
- cfmSessionFlap
- cfmTrunksConfigured
- cfmTrunksRunning
- droppedFrames
- egressDroppedFrames
- eigrpNeighborDeleted
- eigrpNeighborsLearned
- eigrpRoutersConfigured
- eigrpRoutersRunning
- enableArpStats
- enableBfdStats
- enableBgpStats

- enableCfmStats
- enableEigrpStats
- enableIcmpStats
- enableIshStats
- enableLdpStats
- enableMldStats
- enableMplsTpStats
- enableNeighborSolicitStats
- enableOamStats
- enableOspfStats
- enableOspfV3Stats
- enablePimsmStats
- enableProtocolServerStats
- enableRsvpStats
- enableStpStats
- framesReceived
- framesReceivedSStream
- framesSent
- framesSentSStream
- isisIpV4GroupRecordsLearned
- isisIpV6GroupRecordsLearned
- isisL1DBSize
- isisL2DBSize
- isisMacGroupRecordsLearned
- isisNeighborsL1
- isisNeighborsL2
- isisRBridgesLearned
- isisSessionFlapL1
- isisSessionFlapL2
- isisSessionsConfiguredL1
- isisSessionsConfiguredL2
- isisSessionsUpL1
- isisSessionsUpL2
- ldpBasicSessionsUp
- ldpSessionFlap

- ldpSessionsConfigured
- ldpSessionsUp
- link
- mode
- portCpuBytesReceived
- portCpuFramesReceived
- portCPUFramesSent
- portCpuStatus
- protocolServerRx
- protocolServerTx
- protocolServerVlanDroppedFrames
- rxArpReply
- rxArpRequest
- rxNeighborAdvertisements
- rxNeighborSolicits
- rxPingReply
- rxPingRequest
- scheduledFramesSent
- scheduledTransmitTime
- sequenceErrors
- sequenceFrames
- streamTrigger1
- streamTrigger2
- transmitDuration
- transmitState
- txArpReply
- txArpRequest
- txNeighborAdvertisements
- txNeighborSolicits
- txPingReply
- txPingRequest
- userDefinedStat1
- userDefinedStat2
- vlanTaggedFramesRx

List of DHCP
Options

These are option names used in Open DHCP Server. These are based on IANA names less spaces and dashes. Please refer to <http://www.iana.org/assignments/bootp-dhcp-parameters/bootp-dhcp-parameters.xml> for more info. You can also use options not listed here using tag names directly.

Tag	Option Name in Open DHCP Server	IANA Name	Meaning
1	SubnetMask	Subnet Mask	Subnet Mask Value
2	TimeOffset	Time Offset	Time Offset in Seconds from UTC (note: deprecated by 100 and 101)
3	Router	Router	N/4 Router addresses
4	TimeServer	Time Server	N/4 Timeserver addresses
5	NameServer	Name Server	N/4 IEN-116 Server addresses
6	DomainServer	Domain Server	N/4 DHCP Server addresses
7	LogServer	Log Server	N/4 Logging Server addresses
8	QuotesServer	Quotes Server	N/4 Quotes Server addresses
9	LPRServer	LPR Server	N/4 Printer Server addresses
10	ImpressServer	Impress Server	N/4 Impress Server addresses
11	RLPServer	RLP Server	N/4 RLP Server addresses
12	Hostname	Hostname	Hostname string
13	BootFileSize	Boot File Size	Size of boot file in 512 byte chunks
14	MeritDumpFile	Merit Dump File	Client to dump and name the file to dump it to
15	DomainName	Domain Name	The DNS domain name of the client
16	SwapServer	Swap Server	Swap Server address
17	RootPath	Root Path	Path name for root disk
18	ExtensionFile	Extension File	Path name for more BOOTP info
19	ForwardOn/Off	Forward On/Off	Enable/Disable IP Forwarding
20	SrcRteOn/Off	SrcRte On/Off	Enable/Disable Source Routing
21	PolicyFilter	Policy Filter	Routing Policy Filters
22	MaxDGAssembly	Max DG Assembly	Max Datagram Reassembly Size

Tag	Option Name in Open DHCP Server	IANA Name	Meaning
23	DefaultIPTTL	Default IPTTL	Default IP Time to Live
24	MTUTimeout	MTU Timeout	Path MTU Aging Timeout
25	MTUPlateau	MTU Plateau	Path MTU Plateau Table
26	MTUInterface	MTU Interface	Interface MTU Size
27	MTUSubnet	MTU Subnet	All Subnets are Local
28	BroadcastAddress	Broadcast Address	Broadcast Address
29	MaskDiscovery	Mask Discovery	Perform Mask Discovery
30	MaskSupplier	Mask Supplier	Provide Mask to Others
31	RouterDiscovery	Router Discovery	Perform Router Discovery
32	RouterRequest	Router Request	Router Solicitation Address
33	StaticRoute	Static Route	Static Routing Table
34	Trailers	Trailers	Trailer Encapsulation
35	ARPTIMEOUT	ARP Timeout	ARP Cache Timeout
36	Ethernet	Ethernet	Ethernet Encapsulation
37	DefaultTCPTTL	Default TCPTTL	Default TCP Time to Live
38	KeepaliveTime	Keepalive Time	TCP Keepalive Interval
39	KeepaliveData	Keepalive Data	TCP Keepalive Garbage
40	NISDomain	NIS Domain	NIS Domain Name
41	NISServers	NIS Servers	NIS Server Addresses
42	NTPServers	NTP Servers	NTP Server Addresses
44	NETBIOSNameSrv	NETBIOS Name Srv	NETBIOS Name Servers
45	NETBIOSDistSrv	NETBIOS Dist Srv	NETBIOS Datagram Distribution
46	NETBIOSNodeType	NETBIOS Node Type	NETBIOS Node Type
47	NETBIOSScope	NET BIOS Scope	NETBIOS Scope

Tag	Option Name in Open DHCP Server	IANA Name	Meaning
48	XWindowFont	XWindow Font	X Window Font Server
49	XWindowManager	XWindow Manager	X Window Display Manager
51	AddressTime	Address Time	IP Address Lease Time
58	RenewalTime	Renewal Time	DHCP Renewal (T1) Time
59	RebindingTime	Rebinding Time	DHCP Rebinding (T2) Time
62	NetWare/IPDomain	NetWare/IP Domain	NetWare/IP Domain Name
63	NetWare/IPOption	NetWare/IP Option	NetWare/IP sub Options
64	NIS-Domain-Name	NIS-Domain-Name	NIS+ v3 Client Domain Name
65	NIS-Server-Addr	NIS-Server-Addr	NIS+ v3 Server Addresses
66	TFTPServerName	Server Name	TFTP Server Name
67	BootFileOption	BootFile-Option	Boot File Name
68	HomeAgentAddrs	Home-Agent-Addrs	Home Agent Addresses
69	SMTPServer	SMTP-Server	Simple Mail Server Addresses
70	POP3Server	POP3-Server	Post Office Server Addresses
71	NNTPServer	NNTP-Server	Network News Server Addresses
72	WWWServer	WWW-Server	WWW Server Addresses
73	FingerServer	Finger-Server	Finger Server Addresses
74	IRCServer	IRC-Server	Chat Server Addresses
75	StreetTalkServer	StreetTalk-Server	StreetTalk Server Addresses
76	STDAServer	STDA-Server	ST Directory Assist. Addresses
78	DirectoryAgent	Directory Agent	directory agent information
79	ServiceScope	Service Scope	service location agent scope
83	iSNS	iSNS	Internet Storage Name Service
85	NDSServers	NDS Servers	Novell Directory Services
86	NDSTreeName	NDS Tree Name	Novell Directory Services

Tag	Option Name in Open DHCP Server	IANA Name	Meaning
87	NDSContext	NDS Context	Novell Directory Services
95	LDAP	LDAP	Lightweight Directory Access Protocol
100	PCode	PCode	IEEE 1003.1 TZ String
101	TCode	TCode	Reference to the TZ Database
112	NetinfoAddress	Netinfo Address	NetInfo Parent Server Address
113	NetinfoTag	Netinfo Tag	NetInfo Parent Server Tag
114	URL	URL	URL

