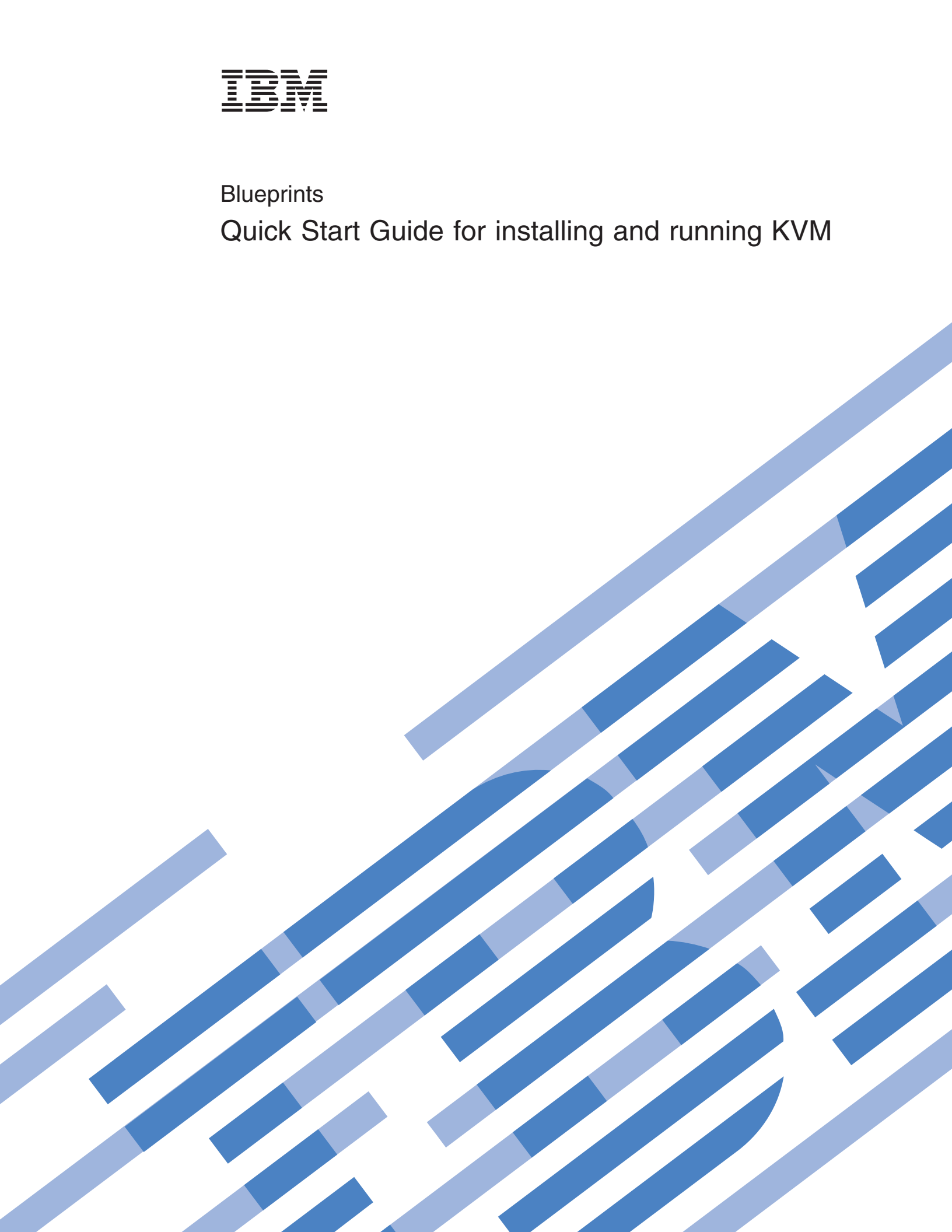

Blueprints

Quick Start Guide for installing and running KVM

Blueprints

Quick Start Guide for installing and running KVM

Note

Before using this information and the product it supports, read the information in "Notices" on page 37.

Third Edition (March 2011)

Contents

Chapter 1. Scope, requirements, and support

This blueprint applies to System x® running Linux. You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

Systems to which this information applies

System x running Linux

Intended audience

This blueprint is intended for Linux system administrators and programmers who want to experiment with using KVM, or have a minimal need for using KVM.

Scope and purpose

This Blueprint provides a quick starting point to begin using a KVM guest, but it assumes that the KVM guests to be created will be using local storage and therefore does not provide information about other storage options. The process of installing an operating system on a KVM guest is also out of the scope of this paper as the process will be the same as installing on any physical machine. If you need to create and manage more than a few KVMs,see *The developer's approach to installing and managing KVMs* at http://publib.boulder.ibm.com/infocenter/lnxinfo/v3r0m0/topic/liaai/kvmadv/kvmadvstart.htm.

Test environment

These instructions were tested on a System x HS21 blade system running the stock Red Hat Enterprise Linux 5.4 kernel.

Hardware requirements

For more information about the hardware requirements, see Chapter 3, "Enabling KVM support on your hardware," on page 5.

Software requirements

The host system must be running Red Hat Enterprise Linux 5.4. For more information about installing Red Hat Enterprise Linux 5.4, see the Red Hat Enterprise Linux 5 Installation Guide at http://www.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/5.4/html/Installation_Guide/index.html.

Other considerations

Throughout this blueprint, instructions are written for KVM guests that use local storage. Note, however, that live migration is only possible if a guest is running on shared storage. Therefore, if you would like to have the capability of live migration in the future, set up shared storage.

Author names

Monza Lui

Kersten Richter

Other contributors

Santwana Samantray

Subrata Modak

IBM® Services

Linux offers flexibility, options, and competitive total cost of ownership with a world class enterprise operating system. Community innovation integrates leading-edge technologies and best practices into Linux.

IBM is a leader in the Linux community with over 600 developers in the IBM Linux Technology Center working on over 100 open source projects in the community. IBM supports Linux on all IBM servers, storage, and middleware, offering the broadest flexibility to match your business needs.

For more information about IBM and Linux, go to ibm.com/linux  (https://www.ibm.com/linux)

IBM Support

Questions and comments regarding this documentation can be posted on the developerWorks® Virtualization Blueprint Community Forum:

http://www.ibm.com/developerworks/forums/forum.jspa?forumID=1272 

The IBM developerWorks discussion forums let you ask questions, share knowledge, ideas, and opinions about technologies and programming techniques with other developerWorks users. Use the forum content at your own risk. While IBM will attempt to provide a timely response to all postings, the use of this developerWorks forum does not guarantee a response to every question that is posted, nor do we validate the answers or the code that are offered.

Chapter 2. KVM overview

Kernel-based Virtual Machine (KVM) is a hardware-assisted, fully virtualized solution for Linux on x86 hardware that contains virtualization extensions (specifically Intel VT or AMD-V). After you install KVM, you can run multiple guests (virtual machines), with each guest running a different operating system image. Each of these virtual machines has private, virtualized hardware, including a network card, storage, memory, and graphics adapter.

For more information about the list of KVM-supported guest operating systems, go to http://www.linux-kvm.org/page/Guest_Support_Status

Related reference:

Chapter 1, “Scope, requirements, and support,” on page 1
This blueprint applies to System x running Linux. You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

Chapter 3. Enabling KVM support on your hardware

The host machines must use either Intel VT or AMD-V chipsets that support hardware-assisted virtualization.

If Linux is already installed on your system, you can also determine if your system processor supports KVM by running the following command:

```
grep -E 'vmx|svm' /proc/cpuinfo
```

If this command returns output, then your system supports KVM. The **vmx** processor feature flag represents Intel VT chipset while the **svm** flag represents AMD-V. Note which KVM flag was returned as it will be useful for loading the correct module later.

Verify that the KVM-related feature is enabled in the BIOS. Complete the following steps to determine if the HS21 blade test machine running with an Intel Xeon chip has the KVM-related feature enabled:

1. Power off the machine completely.
2. Power on the machine and enter the BIOS by pressing the F1 key during boot.
3. In the BIOS menu, select **Advanced Step → CPU Options**.
4. Confirm that the **Intel Virtualization Technology** option is **Enabled**.
5. Save the setting and exit the BIOS.

Now your hardware is ready for deploying a KVM solution.

Related reference:

Chapter 1, “Scope, requirements, and support,” on page 1
This blueprint applies to System x running Linux. You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

Chapter 4. Installing and configuring KVM-related software

You can install and configure KVM-related software on a host system that already has Red Hat Enterprise Linux 5.4 installed.

If you prefer to install the KVM-related software during the installation of your host system, see the Red Hat Enterprise Linux 5 Virtualization guide at http://www.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/5.4/html/Virtualization_Guide/.

Related reference:

Chapter 1, "Scope, requirements, and support," on page 1
This blueprint applies to System x running Linux. You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

Installing KVM on the host system

After Red Hat Enterprise Linux 5.4 is installed on the host system, you can install KVM-related packages.

Procedure

1. Because KVM is not compatible with Xen, make sure Linux is not running a Xen kernel by running the `uname -a` command. If you see output similar to the following, a Xen kernel is running and must be stopped before continuing with Step 2:

   ```
   2.6.18-164.el5Xen
   ```

2. Install the KVM software using yum:

   ```
   yum install kvm
   ```

3. Install additional virtualization management packages:

   ```
   yum install virt-manager libvirt libvirt-python python-virtinst
   ```

Related reference:

Chapter 1, "Scope, requirements, and support," on page 1
This blueprint applies to System x running Linux. You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

Configuring KVM after installing the packages

After you install the KVM-related packages, you can load the correct KVM modules.

Procedure

1. Insert the KVM module by running the following command:

   ```
   modprobe kvm
   ```

2. Insert the chip-specific KVM module by running one of these commands:
   - For the AMD chip (`svm` flag):

     ```
     modprobe kvm-amd
     ```

   - For Intel chip (`vmx` flag):

     ```
     modprobe kvm-intel
     ```

   You can verify that the modules are inserted and running by running the following command:

```
lsmod|grep kvm

kvm-intel          86248  3
kvm                223264  1 kvm_intel
```

If you need help determining which chip your processor uses, see Chapter 3, "Enabling KVM support on your hardware," on page 5.

3. Start the libvirtd daemon service:

```
/etc/init.d/libvirtd start
Starting libvirtd daemon:                                  [  OK  ]
```

4. Set up libvirtd to start on every reboot:

```
chkconfig libvirtd on
```

5. Verify that the default virtual network is available. This default virtual network comes with an isolated virtual bridge device, virbr0, which is set to the 192.168.122.x subnet by default. The host is assigned the 192.168.122.1 address. You can assign an address to your guest from this subnet by manually setting up your network during or after operating system installation. To verify the availability of the virtual bridge:
   a. Run the **ifconfig virbr0** command.
   b. Verify that the output is similar to the following example:

```
virbr0    Link encap:Ethernet  HWaddr 00:00:00:00:00:00
          inet addr:192.168.122.1  Bcast:192.168.122.255  Mask:255.255.255.0
          inet6 addr: fe80::200:ff:fe00:0/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:57 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 b)  TX bytes:10962 (10.7 KiB)
```

Results

The KVM-related software is now configured and ready to be used.

Related reference:

Chapter 1, "Scope, requirements, and support," on page 1
This blueprint applies to System x running Linux. You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

Chapter 5. Configuring the network

You can use the default network setup, or set up a network bridge in the host. The default network allows only outbound communication from the KVM guests. If the KVM guests need full network access, including communication to and from an external host, set up a Linux bridge in the host.

Before you begin

Verify that the default virtual network is available. This default virtual network comes with an isolated virtual bridge device, virbr0, which is set to the 192.168.122.x subnet by default. The host is assigned the 192.168.122.1 address. To verify the availability of the virtual bridge:

1. Run the `ifconfig virbr0` command.
2. Verify that the output is similar to the following example:

```
virbr0    Link encap:Ethernet  HWaddr 00:00:00:00:00:00
          inet addr:192.168.122.1  Bcast:192.168.122.255  Mask:255.255.255.0
          inet6 addr: fe80::200:ff:fe00:0/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:57 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 b)  TX bytes:10962 (10.7 KiB)
```

Related reference:

Chapter 1, “Scope, requirements, and support,” on page 1
This blueprint applies to System x running Linux. You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

Using the default network setup

About this task

You can use the default network setup if both of the following statements are true:

- You will only access the guest from the KVM host.
- You will access the outside network from the guests.

If you are using the default network setup, continue to Chapter 6, “Creating a KVM guest and preparing to install an operating system,” on page 15.

If you are not using the default network setup, continue to “Setting up a network bridge in the host.”

Setting up a network bridge in the host

Before you begin

Ensure that the network card that you want to use for the bridge is providing the network connection you want for your KVM modules and that the network card is working. This card should be set up to provide the same networking capability that you want your guest KVM to have. The following example shows a card that has already been configured for external access. In this example, **eth0** is the network card used.

```
# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:14:5E:C2:1E:40
          inet addr:10.10.1.152  Bcast:10.10.1.255  Mask:255.255.255.0
```

```
          inet6 addr: fe80::214:5eff:fec2:1e40/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:664 errors:0 dropped:526 overruns:0 frame:0
          TX packets:163 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:69635 (68.0 KiB)  TX bytes:25091 (24.5 KiB)
          Interrupt:74 Memory:da000000-da012800
.................
```

If your network card is not yet set up, create a network script for the card and save it in the **/etc/sysconfig/network-scripts/** directory. For more information about setting up a network card, see the Red Hat Enterprise Linux Deployment Guide (http://docs.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/5/html-single/Deployment_Guide/index.html).

Note: If you are accessing the host machine using the same network card you are configuring for the bridge, any discrepancy might cause you to lose your network connection.

About this task

If the KVM guests need full network access, including communication to and from an external host, set up a Linux bridge in the host. Bridged networking allows you to link two Ethernet network segments using packet forwarding technology. Follow these steps to create a public bridge in the host system.

Restriction: The Linux bridge configuration does not work in a wireless host environment.

Procedure

1. Back up the corresponding network script file at a different location for future reference and for network recovery. Issue the following command to back up the network script for the **ifcfg-eth0** network card to the `/root` directory:

   ```
   # cp /etc/sysconfig/network-scripts/ifcfg-eth0 /root/.
   ```

   Note: Do not copy this file to the same network script directory or any of its subdirectories.
2. Navigate to the `/etc/sysconfig/network-scripts/` directory using the following command:

   ```
   cd /etc/sysconfig/network-scripts/
   ```
3. Create another copy of the network script for defining a Linux bridge associated with the network card to a new file called **/etc/sysconfig/network-scripts/ifcfg-*br0***, where *br0* is the name of the bridge, using the following command:

   ```
   cp ifcfg-eth0 ifcfg-br0
   ```

   The complete content of the Linux bridge's configuration file will be based on what is already in the working script of your network card.
4. Edit the script file to direct packets through the bridge. Your network card most likely is configured with a static IP address (`BOOTPROTO=static`) or is configured to get an IP address from a DHCP server (`BOOTPROTO=dhcp`).
   - If your network card is configured with a static IP address, your original network script file should look similar to the following example:

     ```
     DEVICE=eth0
     BOOTPROTO=static
     HWADDR=00:14:5E:C2:1E:40
     IPADDR=10.10.1.152
     NETMASK=255.255.255.0
     ONBOOT=yes
     ```

     The following table shows the contents of the network configuration scripts for **eth0** and **br0**. Edit your scripts as shown in the following example.

Table 1. Bridging network files comparison

/etc/sysconfig/network-scripts/ifcfg-eth0	etc/sysconfig/network-scripts/ifcfg-br0
DEVICE=eth0 TYPE=Ethernet HWADDR=00:14:5E:C2:1E:40 ONBOOT=yes NM_CONTROLLED=no BRIDGE=br0	DEVICE=br0 TYPE=Bridge NM_CONTROLLED=no BOOTPROTO=static IPADDR=10.10.1.152 NETMASK=255.255.255.0 ONBOOT=yes

In the left column is the network script file for network card (**eth0**). The pre-existing information about this network card stays the same, but three items are added:

TYPE The device type.

NM_CONTROLLED=no
Specifies that the card is not controlled by the Network Manager. In order for the bridge to work, only one device can be controlled by the Network Manager.

BRIDGE=br0
Associates this card with the bridge.

In the right column is the network script for the bridge (**br0**). The following changes are reflected:

DEVICE The device name.

TYPE The device type. `Bridge` is case-sensitive and must be added exactly as represented here with an upper case `'B'` and lower case `'ridge'`.

NM_CONTROLLED=no
Specifies that the bridge is not controlled by the Network Manager. In order for the bridge to work, only one device can be controlled by the Network Manager.

The other settings are retained from the network card configuration file.

Note: There should not be a hardware address in this file. These values set up the bridge to behave like the network card: the **ifcfg-br0** file acting as an extension of the **ifcfg-eth0** file where the `BRIDGE=br0` is pointing to the **ifcfg-br0** file.

- If your network card is configured with a dynamic IP address, your original network script file should look similar to the following example:

```
DEVICE=eth0
BOOTPROTO=dhcp
HWADDR=00:14:5E:C2:1E:40
ONBOOT=yes
```

The following table shows the contents of the configuration scripts for **eth0** and **br0**. Edit your scripts as shown in the following example.

Table 2. Bridging network files comparison

/etc/sysconfig/network-scripts/ifcfg-eth0	/etc/sysconfig/network-scripts/ifcfg-br0
DEVICE=eth0 TYPE=Ethernet HWADDR=00:14:5E:C2:1E:40 ONBOOT=yes NM_CONTROLLED=no BRIDGE=br0	DEVICE=br0 TYPE=Bridge NM_CONTROLLED=no BOOTPROTO=dhcp ONBOOT=yes

In the left column is the network script file for network card (**eth0**), which is the same as the example for the static IP address scenario. The pre-existing information about this network card stays the same, but three items are added:

TYPE Specifies the device type.

NM_CONTROLLED=no
Specifies that the card is not controlled by the Network Manager. In order for the bridge to work, only one device can be controlled by the Network Manager.

BRIDGE=br0
Associates this card with the bridge.

In the right column is the network script for the bridge (**br0**). The following changes are reflected:

DEVICE The device name.

TYPE The device type. `Bridge` is case-sensitive and must be added exactly as represented here with an upper case `'B'` and lower case `'ridge'`.

NM_CONTROLLED=no
Specifies that the bridge is not controlled by the Network Manager. In order for the bridge to work, only one device can be controlled by the Network Manager.

Note: There should not be a hardware address in this file. These values set up the bridge to behave like the network card: the **ifcfg-br0** file acting as an extension of the **ifcfg-eth0** file where the `BRIDGE=br0` is pointing to the **ifcfg-br0** file.

5. Restart the network to verify that the configuration works.

   If you configured the network incorrectly, the network connection might drop and you might lose access to your machine. If that happens, check the scripts, and then restart the network by running the following command:

   ```
   # service network restart
   ```

6. Disable Netfilter processing in the bridged traffic by appending the following lines to the `/etc/sysctl.conf` file:

   ```
   net.bridge.bridge-nf-call-ip6tables = 0
   net.bridge.bridge-nf-call-iptables = 0
   net.bridge.bridge-nf-call-arptables = 0
   ```

   Note: For more information about why disabling Netfilter processing is a good security measure, see the "Network isolation options" section of *Securing KVM guests and the host system* at http://publib.boulder.ibm.com/infocenter/lnxinfo/v3r0m0/topic/liaai/kvmsec/kvmsecstart.htm.

7. Reload the kernel parameters with the **sysctl** command:

   ```
   # sysctl -p
   net.ipv4.ip_forward = 0
   ...
   net.bridge.bridge-nf-call-ip6tables = 0
   net.bridge.bridge-nf-call-iptables = 0
   net.bridge.bridge-nf-call-arptables = 0
   ```

8. Verify that your network behaves the same way it did before you made the bridging changes, with one exception: the `ifconfig` command returns different output. The following example shows the first two entries of **ifconfig** in the test environment. Note that the bridge, **br0**, now acts for **eth0**:

   ```
   br0       Link encap:Ethernet  HWaddr 00:14:5E:C2:1E:40
             inet addr:10.10.1.152  Bcast:10.10.1.255  Mask:255.255.255.0
             inet6 addr: fe80::214:5eff:fec2:1e40/64 Scope:Link
             UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
             RX packets:125 errors:0 dropped:0 overruns:0 frame:0
             TX packets:81 errors:0 dropped:0 overruns:0 carrier:0
             collisions:0 txqueuelen:0
             RX bytes:16078 (15.7 KiB)  TX bytes:18542 (18.1 KiB)
   eth0      Link encap:Ethernet  HWaddr 00:14:5E:C2:1E:40
             inet6 addr: fe80::214:5eff:fec2:1e40/64 Scope:Link
             UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
             RX packets:206 errors:0 dropped:0 overruns:0 frame:0
   ```

```
          TX packets:58 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:27308 (26.6 KiB)  TX bytes:13881 (13.5 KiB)
          Interrupt:74 Memory:da000000-da012800
```

You can also see this bridge by running the following command:

```
brctl show
bridge name     bridge id               STP enabled     interfaces
virbr0          8000.000000000000       yes
br0             8000.000e0cb30550       no              eth0
```

Results

Your Linux bridge is configured and ready to use.

Chapter 6. Creating a KVM guest and preparing to install an operating system

After you install the KVM packages and optionally set up a Linux bridge on the host system for the guests to use, you can create a KVM guest and install an operating system on it.

For a complete listing of supported operating systems, see http://www.linux-kvm.org/page/Guest_Support_Status.

The following instructions set up SUSE Linux Enterprise Server 11 and Red Hat Enterprise Linux 5.4, respectively, as guest operating systems. You can adapt the instructions to set up any other supported operating system.

There are several different methods of creating a guest on a KVM host machine. The example instructions use two of those methods:

- virt-manager: a GUI tool
- virt-install: a command line tool.

If you prefer a more detailed approach to creating and managing KVMs, see The developer's approach to installing and managing KVMs.

Related reference:

Chapter 1, “Scope, requirements, and support,” on page 1
This blueprint applies to System x running Linux. You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

Creating a KVM guest using virt-manager

The **virt-manager** tool is a GUI tool used to create and manage KVM guests.

About this task

Follow these steps to set up a KVM guest using **virt-manager**. This example installs a SLES 11 guest.

Procedure

1. Start your X11 environment by logging in to your system using the **ssh -X** command.
2. Start **virt-manager** by running the **virt-manager** command.
   This command opens the Virtual Machine Manger window.
3. Select the first row, which represents the host domain, and click **New** to create a new guest. This action opens the Virtual Machine Creation window. Click **Forward** to start entering information about your guest.
4. In the Virtual Machine Name window, enter a name for your guest. In this example, the virtual machine is named *MY_VM1_SLES11*. Click **Forward**.
5. In the Virtualization Method window, select the virtualization method that you want to use.
   - Select `Fully virtualized` (paravirtualization is not supported in KVM).
   - Select the processor architecture you want to simulate. This example uses `x86_64`.

     Note: If you are running KVM on an i686 machine, you cannot simulate the x86_64 architecture.
   - Select `kvm` for Hypervisor.

Click **Forward**.

6. In the Installation Method window, select your installation method.
   - Select your installation media. This example uses an installation source that is available through an NFS mount so `Network Install tree (HTTP, FTP, or NFS)` is selected as the installation media
   - Select your OS type. This example is installing a Linux OS so `Linux` is selected as the OS type
   - Select your OS variant. This example is installing SLES so `Suse Linux Enterprise Server` is selected as the OS variant

   Click **Forward**.
7. Depending on which installation media option you chose, you may need to select an installation source. If you are using the Network PXE boot option, skip this step and go the next step to assign storage.

   If you are using the ISO or CD media, or the network media installation methods, specify the location of your installation media. In this example, the location of the NFS install tree (`nfs://xyz.com/nfs_installdir_sles11`) is specified in the Installation media URL field. Click **Forward**.
8. In the Storage window, assign an existing **Block device** or select to create an **.img File**. If you select **File**, specify its size. Make sure that you allocate enough disk space for your operating system by consulting the operating system's documentation for the minimum amount of disk space needed. In this example, an existing LVM partition, **/dev/mapper/MY_VG1-MY_LVM1**, from a local disk is assigned. Click **Forward**.
9. In the Network window, select **Virtual network** or **Shared physical device**.

   If you have not set up a Linux bridge, choose the default **Virtual network**.

   In this example, **Shared physical device** is used and `br0` (bridge to **eth0**) is specified. This bridge was configured earlier in the section Chapter 5, "Configuring the network," on page 9. Click **Forward**.
10. In the Memory and CPU Allocation window, select the maximum and startup memory, and the number of virtual CPUs that the system should have. In this example, the guest is given `2000 MB` Max memory, `1200 MB` of Startup memory and `2` Virtual CPUs.

    Note: Note that you can dynamically update the two resources while the guest is still running if the new assignment is smaller or equal to the maximum amount allocated.

    Click **Forward**.
11. In the Summary window, verify the settings for creating the guest. When you are satisfied, click **Finish**.

What to do next

After you click **Finish**, the operating system installation dialog opens. Install the operating system as you would on any other system. Remember to use *br0* as your network device. For more information about installing your guest, see "Installing the guest operating system" on page 18.

Related reference:

Chapter 1, "Scope, requirements, and support," on page 1
This blueprint applies to System x running Linux. You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

Creating a KVM guest using virt-install

You can create a KVM guest with the **`virt-install`** command-line tool. Two examples are shown in this section to demonstrate how this tool can be used to create KVMs.

This topic provides information about creating KVM guests, and specific examples for SLES and Red Hat Enterprise Linux. For more details about the **virt-install** command-line tool, see the **virt-install** man page.

Example: Creating a KVM guest with SLES 11

The following command code example creates a KVM guest and opens a SLES 11 installation screen in graphical mode, after the SLES 11 installation source is found. This command requires the **virt-viewer** tool, one of the additional virtualization management packages you can install. It also requires that you log in to the server using `ssh -x` to access the X Window System environment.

```
# virt-install \
       --name kvm1 \
       --ram 500 \
       --disk path=/var/lib/libvirt/images/kvm1.img,size=5 \
       --network network:default \
       --accelerate \
       --vnc \
       -c /tmp/SLES11-x86_64-DVD.iso
```

This command example creates a KVM guest named kvm1 that has 500 MB of memory allocated to it. A local 5-GB file, `kvm1.img`, is created as the storage for the guest in the default directory for KVM guest images, `/var/lib/libvirt/images/`. The default virtual network is used. During installation, KVM kernel acceleration capabilities are used, if they are available. A VNC server is made available for connection, and the installation source is an ISO in `/tmp/SLES11-x86_64-DVD.iso`.

Example: Creating a KVM guest with Red Hat Enterprise Linux 5.4

The following command code example creates a KVM guest and opens a Red Hat Enterprise Linux 5.4 installation screen in text mode, after the Red Hat Enterprise Linux 5.4 installation source is found. This command requires the **virt-viewer** tool, one of the additional virtualization management packages installed as described in "Installing KVM on the host system" on page 7. This command does not require an X Window System environment.

```
# virt-install \
       --name kvm2 \
       --vcpus 2 \
       --ram 1000 \
       --disk path=/dev/mapper/VolGroup00-LogVol03  \
       --network bridge:br0 \
       --arch i686 \
       -l nfs://10.1.1.212/nfsexport/rhel5.4-server-i386-is/
```

This command example creates a KVM guest named kvm2. Two virtual processors are assigned to this guest, and 1 GB of memory is allocated. An existing LVM partition, `/dev/mapper/VolGroup00-LogVol03`, is used for storage. The example assumes that a Linux bridge, **br0,** was set up earlier as described in Chapter 5, "Configuring the network," on page 9. The guest is of i686 architecture. An NFS mount point of host IP 10.1.1.212 and directory `/nfsexport/rhel5.4-server-i386-is/` is specified as the installation source.

Because the **--vnc** option is not specified, no VNC server is made available for connection, and the installation screen is provided in text mode only. Also, the KVM kernel acceleration capabilities are not used even if they are available because the **--acelerate** option is not used.

KVM guest definition file when using virt-install

Each time a new KVM guest is created using the **virt-install** tool, an XML file is created in the `/etc/libvirt/qemu/` directory. The XML file is the definition file for the guest and is named *guest_name*`.xml`.

The convenience of using the **virt-install** tool to create a KVM guest is that you do not have to write the XML file to define it. The trade-off is that you have less freedom in customizing your KVM guest. If you want more flexibility in creating your KVM, see *The developer's approach to installing and managing KVMs* at http://publib.boulder.ibm.com/infocenter/lnxinfo/v3r0m0/topic/liaai/kvmadv/kvmadvstart.htm

Related reference:

Chapter 1, "Scope, requirements, and support," on page 1
This blueprint applies to System x running Linux. You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

Installing the guest operating system

After your guest is created, you can install the operating system for the guest.

If you used **virt-manager**, an installation window should now be open. However, you can use **virt-viewer** to continue the installation or recover the installation window if it is closed.

To install **virt-viewer**, run the following command:

```
# yum install virt-viewer
```

To start using **virt-viewer**, enter the following command:

```
# virt-viewer <guest name>
```

After your installation window opens, you can start installing your KVM guest's operating system as you would on any other physical machine. If you set up the Linux bridge, remember to select it when you set up the network for your guest. If you are using the default virtual network, choosing to use DHCP gives the guest a 192.168.122.x address. If you have a preference as to which 192.168.122.x address the guest is assigned, choose manual configuration and use any unused address from this subnet. Specify your subnet mask as 255.255.255.0, and your gateway as 192.168.122.1, the host's address.

To regain control of the mouse when you are using **virt-viewer** or **virt-manager**, press **Alt+Ctrl**.

Be sure to allow an SSH connection during the installation if you are planning to access your guest through SSH..

For more information about installing Red Hat Enterprise Linux 5, SLES 11, and Windows XP, see the following appropriate document:

- *Red Hat Enterprise Linux 5 Installation Guide* at http://www.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/5.4/html/Installation_Guide/index.html
- *SLES 11 Installation Quick Start* at http://www.novell.com/documentation/sles11/.
- Windows Operating system at http://windows.microsoft.com/en-US/windows/help/install-reinstall-uninstall.

After the guest operating system installation

You can continue to connect to your guest using **virt-viewer**. If you allowed the SSH connection during the guest operating system installation, you can use SSH to connect to your guest. Use either the virtual DHCP assigned address (if you did not set up Linux bridge) or the physical address (if you set up a Linux bridge on the host).

Related reference:

Chapter 1, “Scope, requirements, and support,” on page 1
This blueprint applies to System x running Linux. You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

Chapter 7. Quick Start Guide for installing and running KVM

Kernel-based Virtual Machine (KVM) is a Linux kernel virtualization hypervisor that can host different guest operating systems. This blueprint shows you how to create a basic KVM to use as a sandbox or a one-time-only production environment. This blueprint also demonstrates how to install KVM on Red Hat Enterprise Linux 5.4 host systems, set up a Linux bridge for guests' network, and create a KVM guest and start installation of a guest operating system. Key tools and technologies discussed in this demonstration include KVM, virt-viewer, virt-manager, virt-install, Linux bridge, and virtualization.

Scope, requirements, and support

This blueprint applies to System x running Linux. You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

Systems to which this information applies

System x running Linux

Intended audience

This blueprint is intended for Linux system administrators and programmers who want to experiment with using KVM, or have a minimal need for using KVM.

Scope and purpose

This Blueprint provides a quick starting point to begin using a KVM guest, but it assumes that the KVM guests to be created will be using local storage and therefore does not provide information about other storage options. The process of installing an operating system on a KVM guest is also out of the scope of this paper as the process will be the same as installing on any physical machine. If you need to create and manage more than a few KVMs,see *The developer's approach to installing and managing KVMs* at http://publib.boulder.ibm.com/infocenter/lnxinfo/v3r0m0/topic/liaai/kvmadv/kvmadvstart.htm.

Test environment

These instructions were tested on a System x HS21 blade system running the stock Red Hat Enterprise Linux 5.4 kernel.

Hardware requirements

For more information about the hardware requirements, see Chapter 3, "Enabling KVM support on your hardware," on page 5.

Software requirements

The host system must be running Red Hat Enterprise Linux 5.4. For more information about installing Red Hat Enterprise Linux 5.4, see the Red Hat Enterprise Linux 5 Installation Guide at http://www.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/5.4/html/Installation_Guide/index.html.

Other considerations

Throughout this blueprint, instructions are written for KVM guests that use local storage. Note, however, that live migration is only possible if a guest is running on shared storage. Therefore, if you would like to have the capability of live migration in the future, set up shared storage.

Author names

Monza Lui

Kersten Richter

Other contributors

Santwana Samantray

Subrata Modak

IBM Services

Linux offers flexibility, options, and competitive total cost of ownership with a world class enterprise operating system. Community innovation integrates leading-edge technologies and best practices into Linux.

IBM is a leader in the Linux community with over 600 developers in the IBM Linux Technology Center working on over 100 open source projects in the community. IBM supports Linux on all IBM servers, storage, and middleware, offering the broadest flexibility to match your business needs.

For more information about IBM and Linux, go to ibm.com/linux  (https://www.ibm.com/linux)

IBM Support

Questions and comments regarding this documentation can be posted on the developerWorks Virtualization Blueprint Community Forum:

http://www.ibm.com/developerworks/forums/forum.jspa?forumID=1272 

The IBM developerWorks discussion forums let you ask questions, share knowledge, ideas, and opinions about technologies and programming techniques with other developerWorks users. Use the forum content at your own risk. While IBM will attempt to provide a timely response to all postings, the use of this developerWorks forum does not guarantee a response to every question that is posted, nor do we validate the answers or the code that are offered.

KVM overview

Kernel-based Virtual Machine (KVM) is a hardware-assisted, fully virtualized solution for Linux on x86 hardware that contains virtualization extensions (specifically Intel VT or AMD-V). After you install KVM, you can run multiple guests (virtual machines), with each guest running a different operating system image. Each of these virtual machines has private, virtualized hardware, including a network card, storage, memory, and graphics adapter.

For more information about the list of KVM-supported guest operating systems, go to http://www.linux-kvm.org/page/Guest_Support_Status

Related reference:

Chapter 1, “Scope, requirements, and support,” on page 1
This blueprint applies to System x running Linux. You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

Enabling KVM support on your hardware

The host machines must use either Intel VT or AMD-V chipsets that support hardware-assisted virtualization.

If Linux is already installed on your system, you can also determine if your system processor supports KVM by running the following command:

```
grep -E 'vmx|svm' /proc/cpuinfo
```

If this command returns output, then your system supports KVM. The **vmx** processor feature flag represents Intel VT chipset while the **svm** flag represents AMD-V. Note which KVM flag was returned as it will be useful for loading the correct module later.

Verify that the KVM-related feature is enabled in the BIOS. Complete the following steps to determine if the HS21 blade test machine running with an Intel Xeon chip has the KVM-related feature enabled:

1. Power off the machine completely.
2. Power on the machine and enter the BIOS by pressing the F1 key during boot.
3. In the BIOS menu, select **Advanced Step → CPU Options**.
4. Confirm that the **Intel Virtualization Technology** option is **Enabled**.
5. Save the setting and exit the BIOS.

Now your hardware is ready for deploying a KVM solution.

Related reference:

Chapter 1, “Scope, requirements, and support,” on page 1
This blueprint applies to System x running Linux. You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

Installing and configuring KVM-related software

You can install and configure KVM-related software on a host system that already has Red Hat Enterprise Linux 5.4 installed.

If you prefer to install the KVM-related software during the installation of your host system, see the Red Hat Enterprise Linux 5 Virtualization guide at http://www.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/5.4/html/Virtualization_Guide/.

Related reference:

Chapter 1, “Scope, requirements, and support,” on page 1
This blueprint applies to System x running Linux. You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

Installing KVM on the host system

After Red Hat Enterprise Linux 5.4 is installed on the host system, you can install KVM-related packages.

Procedure

1. Because KVM is not compatible with Xen, make sure Linux is not running a Xen kernel by running the `uname -a` command. If you see output similar to the following, a Xen kernel is running and must be stopped before continuing with Step 2:

   ```
   2.6.18-164.el5Xen
   ```

2. Install the KVM software using yum:

   ```
   yum install kvm
   ```

3. Install additional virtualization management packages:

   ```
   yum install virt-manager libvirt libvirt-python python-virtinst
   ```

Related reference:

Chapter 1, “Scope, requirements, and support,” on page 1
This blueprint applies to System x running Linux. You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

Configuring KVM after installing the packages

After you install the KVM-related packages, you can load the correct KVM modules.

Procedure

1. Insert the KVM module by running the following command:

   ```
   modprobe kvm
   ```

2. Insert the chip-specific KVM module by running one of these commands:
   - For the AMD chip (`svm` flag):

     ```
     modprobe kvm-amd
     ```

   - For Intel chip (`vmx` flag):

     ```
     modprobe kvm-intel
     ```

   You can verify that the modules are inserted and running by running the following command:

   ```
   lsmod|grep kvm

   kvm-intel          86248  3
   kvm                 223264  1 kvm_intel
   ```

   If you need help determining which chip your processor uses, see Chapter 3, “Enabling KVM support on your hardware,” on page 5.

3. Start the libvirtd daemon service:

   ```
   /etc/init.d/libvirtd start
   Starting libvirtd daemon:                                  [  OK  ]
   ```

4. Set up libvirtd to start on every reboot:

   ```
   chkconfig libvirtd on
   ```

5. Verify that the default virtual network is available. This default virtual network comes with an isolated virtual bridge device, virbr0, which is set to the 192.168.122.x subnet by default. The host is assigned the 192.168.122.1 address. You can assign an address to your guest from this subnet by manually setting up your network during or after operating system installation. To verify the availability of the virtual bridge:
   a. Run the **`ifconfig virbr0`** command.
   b. Verify that the output is similar to the following example:

      ```
      virbr0    Link encap:Ethernet  HWaddr 00:00:00:00:00:00
                inet addr:192.168.122.1  Bcast:192.168.122.255  Mask:255.255.255.0
                inet6 addr: fe80::200:ff:fe00:0/64 Scope:Link
                UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
                RX packets:0 errors:0 dropped:0 overruns:0 frame:0
      ```

```
TX packets:57 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:0
RX bytes:0 (0.0 b)  TX bytes:10962 (10.7 KiB)
```

Results

The KVM-related software is now configured and ready to be used.

Related reference:

Chapter 1, “Scope, requirements, and support,” on page 1
This blueprint applies to System x running Linux. You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

Configuring the network

You can use the default network setup, or set up a network bridge in the host. The default network allows only outbound communication from the KVM guests. If the KVM guests need full network access, including communication to and from an external host, set up a Linux bridge in the host.

Before you begin

Verify that the default virtual network is available. This default virtual network comes with an isolated virtual bridge device, virbr0, which is set to the 192.168.122.x subnet by default. The host is assigned the 192.168.122.1 address. To verify the availability of the virtual bridge:

1. Run the `ifconfig virbr0` command.
2. Verify that the output is similar to the following example:

```
virbr0    Link encap:Ethernet  HWaddr 00:00:00:00:00:00
          inet addr:192.168.122.1  Bcast:192.168.122.255  Mask:255.255.255.0
          inet6 addr: fe80::200:ff:fe00:0/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:57 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 b)  TX bytes:10962 (10.7 KiB)
```

Related reference:

Chapter 1, “Scope, requirements, and support,” on page 1
This blueprint applies to System x running Linux. You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

Using the default network setup

About this task

You can use the default network setup if both of the following statements are true:

- You will only access the guest from the KVM host.
- You will access the outside network from the guests.

If you are using the default network setup, continue to Chapter 6, “Creating a KVM guest and preparing to install an operating system,” on page 15.

If you are not using the default network setup, continue to “Setting up a network bridge in the host” on page 9.

Setting up a network bridge in the host

Before you begin

Ensure that the network card that you want to use for the bridge is providing the network connection you want for your KVM modules and that the network card is working. This card should be set up to provide the same networking capability that you want your guest KVM to have. The following example shows a card that has already been configured for external access. In this example, **eth0** is the network card used.

```
# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:14:5E:C2:1E:40
          inet addr:10.10.1.152  Bcast:10.10.1.255  Mask:255.255.255.0
          inet6 addr: fe80::214:5eff:fec2:1e40/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:664 errors:0 dropped:526 overruns:0 frame:0
          TX packets:163 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:69635 (68.0 KiB)  TX bytes:25091 (24.5 KiB)
          Interrupt:74 Memory:da000000-da012800
.................
```

If your network card is not yet set up, create a network script for the card and save it in the **/etc/sysconfig/network-scripts/** directory. For more information about setting up a network card, see the Red Hat Enterprise Linux Deployment Guide (http://docs.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/5/html-single/Deployment_Guide/index.html).

Note: If you are accessing the host machine using the same network card you are configuring for the bridge, any discrepancy might cause you to lose your network connection.

About this task

If the KVM guests need full network access, including communication to and from an external host, set up a Linux bridge in the host. Bridged networking allows you to link two Ethernet network segments using packet forwarding technology. Follow these steps to create a public bridge in the host system.

Restriction: The Linux bridge configuration does not work in a wireless host environment.

Procedure

1. Back up the corresponding network script file at a different location for future reference and for network recovery. Issue the following command to back up the network script for the **ifcfg-eth0** network card to the `/root` directory:

   ```
   # cp /etc/sysconfig/network-scripts/ifcfg-eth0 /root/.
   ```

   Note: Do not copy this file to the same network script directory or any of its subdirectories.
2. Navigate to the `/etc/sysconfig/network-scripts/` directory using the following command:

   ```
   cd /etc/sysconfig/network-scripts/
   ```
3. Create another copy of the network script for defining a Linux bridge associated with the network card to a new file called **/etc/sysconfig/network-scripts/ifcfg-*br0***, where *br0* is the name of the bridge, using the following command:

   ```
   cp ifcfg-eth0 ifcfg-br0
   ```

   The complete content of the Linux bridge's configuration file will be based on what is already in the working script of your network card.
4. Edit the script file to direct packets through the bridge. Your network card most likely is configured with a static IP address (`BOOTPROTO=static`) or is configured to get an IP address from a DHCP server (`BOOTPROTO=dhcp`).

- If your network card is configured with a static IP address, your original network script file should look similar to the following example:

  ```
  DEVICE=eth0
  BOOTPROTO=static
  HWADDR=00:14:5E:C2:1E:40
  IPADDR=10.10.1.152
  NETMASK=255.255.255.0
  ONBOOT=yes
  ```

  The following table shows the contents of the network configuration scripts for **eth0** and **br0**. Edit your scripts as shown in the following example.

Table 3. Bridging network files comparison

/etc/sysconfig/network-scripts/ifcfg-eth0	**etc/sysconfig/network-scripts/ifcfg-br0**
DEVICE=eth0 TYPE=Ethernet HWADDR=00:14:5E:C2:1E:40 ONBOOT=yes NM_CONTROLLED=no BRIDGE=br0	DEVICE=br0 TYPE=Bridge NM_CONTROLLED=no BOOTPROTO=static IPADDR=10.10.1.152 NETMASK=255.255.255.0 ONBOOT=yes

  In the left column is the network script file for network card (**eth0**). The pre-existing information about this network card stays the same, but three items are added:

  TYPE The device type.

  NM_CONTROLLED=no
  Specifies that the card is not controlled by the Network Manager. In order for the bridge to work, only one device can be controlled by the Network Manager.

  BRIDGE=br0
  Associates this card with the bridge.

  In the right column is the network script for the bridge (**br0**). The following changes are reflected:

  DEVICE The device name.

  TYPE The device type. `Bridge` is case-sensitive and must be added exactly as represented here with an upper case `'B'` and lower case `'ridge'`.

  NM_CONTROLLED=no
  Specifies that the bridge is not controlled by the Network Manager. In order for the bridge to work, only one device can be controlled by the Network Manager.

  The other settings are retained from the network card configuration file.

  Note: There should not be a hardware address in this file. These values set up the bridge to behave like the network card: the **ifcfg-br0** file acting as an extension of the **ifcfg-eth0** file where the `BRIDGE=br0` is pointing to the **ifcfg-br0** file.

- If your network card is configured with a dynamic IP address, your original network script file should look similar to the following example:

  ```
  DEVICE=eth0
  BOOTPROTO=dhcp
  HWADDR=00:14:5E:C2:1E:40
  ONBOOT=yes
  ```

  The following table shows the contents of the configuration scripts for **eth0** and **br0**. Edit your scripts as shown in the following example.

Table 4. Bridging network files comparison

/etc/sysconfig/network-scripts/ifcfg-eth0	/etc/sysconfig/network-scripts/ifcfg-br0
DEVICE=eth0 TYPE=Ethernet HWADDR=00:14:5E:C2:1E:40 ONBOOT=yes NM_CONTROLLED=no BRIDGE=br0	DEVICE=br0 TYPE=Bridge NM_CONTROLLED=no BOOTPROTO=dhcp ONBOOT=yes

In the left column is the network script file for network card (**eth0**), which is the same as the example for the static IP address scenario. The pre-existing information about this network card stays the same, but three items are added:

TYPE Specifies the device type.

NM_CONTROLLED=no
Specifies that the card is not controlled by the Network Manager. In order for the bridge to work, only one device can be controlled by the Network Manager.

BRIDGE=br0
Associates this card with the bridge.

In the right column is the network script for the bridge (**br0**). The following changes are reflected:

DEVICE The device name.

TYPE The device type. `Bridge` is case-sensitive and must be added exactly as represented here with an upper case `'B'` and lower case `'ridge'`.

NM_CONTROLLED=no
Specifies that the bridge is not controlled by the Network Manager. In order for the bridge to work, only one device can be controlled by the Network Manager.

Note: There should not be a hardware address in this file. These values set up the bridge to behave like the network card: the **ifcfg-br0** file acting as an extension of the **ifcfg-eth0** file where the `BRIDGE=br0` is pointing to the **ifcfg-br0** file.

5. Restart the network to verify that the configuration works.

   If you configured the network incorrectly, the network connection might drop and you might lose access to your machine. If that happens, check the scripts, and then restart the network by running the following command:

   ```
   # service network restart
   ```

6. Disable Netfilter processing in the bridged traffic by appending the following lines to the `/etc/sysctl.conf` file:

   ```
   net.bridge.bridge-nf-call-ip6tables = 0
   net.bridge.bridge-nf-call-iptables = 0
   net.bridge.bridge-nf-call-arptables = 0
   ```

   Note: For more information about why disabling Netfilter processing is a good security measure, see the "Network isolation options" section of *Securing KVM guests and the host system* at http://publib.boulder.ibm.com/infocenter/lnxinfo/v3r0m0/topic/liaai/kvmsec/kvmsecstart.htm.

7. Reload the kernel parameters with the **sysctl** command:

   ```
   # sysctl -p
   net.ipv4.ip_forward = 0
   ...
   net.bridge.bridge-nf-call-ip6tables = 0
   net.bridge.bridge-nf-call-iptables = 0
   net.bridge.bridge-nf-call-arptables = 0
   ```

8. Verify that your network behaves the same way it did before you made the bridging changes, with one exception: the `ifconfig` command returns different output. The following example shows the first two entries of **ifconfig** in the test environment. Note that the bridge, **br0**, now acts for **eth0**:

```
br0       Link encap:Ethernet  HWaddr 00:14:5E:C2:1E:40
          inet addr:10.10.1.152  Bcast:10.10.1.255  Mask:255.255.255.0
          inet6 addr: fe80::214:5eff:fec2:1e40/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:125 errors:0 dropped:0 overruns:0 frame:0
          TX packets:81 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:16078 (15.7 KiB)  TX bytes:18542 (18.1 KiB)
eth0      Link encap:Ethernet  HWaddr 00:14:5E:C2:1E:40
          inet6 addr: fe80::214:5eff:fec2:1e40/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:206 errors:0 dropped:0 overruns:0 frame:0
          TX packets:58 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:27308 (26.6 KiB)  TX bytes:13881 (13.5 KiB)
          Interrupt:74 Memory:da000000-da012800
```

You can also see this bridge by running the following command:

```
brctl show
bridge name     bridge id               STP enabled     interfaces
virbr0          8000.000000000000       yes
br0             8000.000e0cb30550       no              eth0
```

Results

Your Linux bridge is configured and ready to use.

Creating a KVM guest and preparing to install an operating system

After you install the KVM packages and optionally set up a Linux bridge on the host system for the guests to use, you can create a KVM guest and install an operating system on it.

For a complete listing of supported operating systems, see http://www.linux-kvm.org/page/Guest_Support_Status.

The following instructions set up SUSE Linux Enterprise Server 11 and Red Hat Enterprise Linux 5.4, respectively, as guest operating systems. You can adapt the instructions to set up any other supported operating system.

There are several different methods of creating a guest on a KVM host machine. The example instructions use two of those methods:

- virt-manager: a GUI tool
- virt-install: a command line tool.

If you prefer a more detailed approach to creating and managing KVMs, see The developer's approach to installing and managing KVMs.

Related reference:

Chapter 1, “Scope, requirements, and support,” on page 1
This blueprint applies to System x running Linux. You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

Creating a KVM guest using virt-manager

The **virt-manager** tool is a GUI tool used to create and manage KVM guests.

About this task

Follow these steps to set up a KVM guest using **virt-manager**. This example installs a SLES 11 guest.

Procedure

1. Start your X11 environment by logging in to your system using the **ssh -X** command.
2. Start **virt-manager** by running the **virt-manager** command.

   This command opens the Virtual Machine Manger window.
3. Select the first row, which represents the host domain, and click **New** to create a new guest. This action opens the Virtual Machine Creation window. Click **Forward** to start entering information about your guest.
4. In the Virtual Machine Name window, enter a name for your guest. In this example, the virtual machine is named *MY_VM1_SLES11*. Click **Forward**.
5. In the Virtualization Method window, select the virtualization method that you want to use.
   - Select `Fully virtualized` (paravirtualization is not supported in KVM).
   - Select the processor architecture you want to simulate. This example uses `x86_64`.

     Note: If you are running KVM on an i686 machine, you cannot simulate the x86_64 architecture.
   - Select `kvm` for Hypervisor.

   Click **Forward**.
6. In the Installation Method window, select your installation method.
   - Select your installation media. This example uses an installation source that is available through an NFS mount so `Network Install tree (HTTP, FTP, or NFS)` is selected as the installation media
   - Select your OS type. This example is installing a Linux OS so `Linux` is selected as the OS type
   - Select your OS variant. This example is installing SLES so `Suse Linux Enterprise Server` is selected as the OS variant

   Click **Forward**.
7. Depending on which installation media option you chose, you may need to select an installation source. If you are using the Network PXE boot option, skip this step and go the next step to assign storage.

   If you are using the ISO or CD media, or the network media installation methods, specify the location of your installation media. In this example, the location of the NFS install tree (`nfs://xyz.com/nfs_installdir_sles11`) is specified in the Installation media URL field. Click **Forward**.
8. In the Storage window, assign an existing **Block device** or select to create an **.img File**. If you select **File**, specify its size. Make sure that you allocate enough disk space for your operating system by consulting the operating system's documentation for the minimum amount of disk space needed. In this example, an existing LVM partition, **/dev/mapper/MY_VG1-MY_LVM1**, from a local disk is assigned. Click **Forward**.
9. In the Network window, select **Virtual network** or **Shared physical device**.

   If you have not set up a Linux bridge, choose the default **Virtual network**.

   In this example, **Shared physical device** is used and `br0` (bridge to **eth0**) is specified. This bridge was configured earlier in the section Chapter 5, “Configuring the network,” on page 9. Click **Forward**.
10. In the Memory and CPU Allocation window, select the maximum and startup memory, and the number of virtual CPUs that the system should have. In this example, the guest is given `2000 MB` Max memory, `1200 MB` of Startup memory and `2` Virtual CPUs.

    Note: Note that you can dynamically update the two resources while the guest is still running if the new assignment is smaller or equal to the maximum amount allocated.

Click **Forward**.

11. In the Summary window, verify the settings for creating the guest. When you are satisfied, click **Finish**.

What to do next

After you click **Finish**, the operating system installation dialog opens. Install the operating system as you would on any other system. Remember to use *br0* as your network device. For more information about installing your guest, see "Installing the guest operating system" on page 18.

Related reference:

Chapter 1, "Scope, requirements, and support," on page 1
This blueprint applies to System x running Linux. You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

Creating a KVM guest using virt-install

You can create a KVM guest with the **virt-install** command-line tool. Two examples are shown in this section to demonstrate how this tool can be used to create KVMs.

This topic provides information about creating KVM guests, and specific examples for SLES and Red Hat Enterprise Linux. For more details about the **virt-install** command-line tool, see the **virt-install** man page.

Example: Creating a KVM guest with SLES 11

The following command code example creates a KVM guest and opens a SLES 11 installation screen in graphical mode, after the SLES 11 installation source is found. This command requires the **virt-viewer** tool, one of the additional virtualization management packages you can install. It also requires that you log in to the server using `ssh -x` to access the X Window System environment.

```
# virt-install \
       --name kvm1 \
       --ram 500 \
       --disk path=/var/lib/libvirt/images/kvm1.img,size=5 \
       --network network:default \
       --accelerate \
       --vnc \
       -c /tmp/SLES11-x86_64-DVD.iso
```

This command example creates a KVM guest named kvm1 that has 500 MB of memory allocated to it. A local 5-GB file, `kvm1.img`, is created as the storage for the guest in the default directory for KVM guest images, `/var/lib/libvirt/images/`. The default virtual network is used. During installation, KVM kernel acceleration capabilities are used, if they are available. A VNC server is made available for connection, and the installation source is an ISO in `/tmp/SLES11-x86_64-DVD.iso`.

Example: Creating a KVM guest with Red Hat Enterprise Linux 5.4

The following command code example creates a KVM guest and opens a Red Hat Enterprise Linux 5.4 installation screen in text mode, after the Red Hat Enterprise Linux 5.4 installation source is found. This command requires the **virt-viewer** tool, one of the additional virtualization management packages installed as described in "Installing KVM on the host system" on page 7. This command does not require an X Window System environment.

```
# virt-install \
       --name kvm2 \
       --vcpus 2 \
       --ram 1000 \
       --disk path=/dev/mapper/VolGroup00-LogVol03  \
```

```
--network bridge:br0 \
--arch i686 \
-l nfs://10.1.1.212/nfsexport/rhel5.4-server-i386-is/
```

This command example creates a KVM guest named kvm2. Two virtual processors are assigned to this guest, and 1 GB of memory is allocated. An existing LVM partition, `/dev/mapper/VolGroup00-LogVol03`, is used for storage. The example assumes that a Linux bridge, **br0,** was set up earlier as described in Chapter 5, "Configuring the network," on page 9. The guest is of i686 architecture. An NFS mount point of host IP 10.1.1.212 and directory `/nfsexport/rhel5.4-server-i386-is/` is specified as the installation source.

Because the **--vnc** option is not specified, no VNC server is made available for connection, and the installation screen is provided in text mode only. Also, the KVM kernel acceleration capabilities are not used even if they are available because the **--acelerate** option is not used.

KVM guest definition file when using virt-install

Each time a new KVM guest is created using the **virt-install** tool, an XML file is created in the `/etc/libvirt/qemu/` directory. The XML file is the definition file for the guest and is named *guest_name*`.xml`.

The convenience of using the **virt-install** tool to create a KVM guest is that you do not have to write the XML file to define it. The trade-off is that you have less freedom in customizing your KVM guest. If you want more flexibility in creating your KVM, see *The developer's approach to installing and managing KVMs* at http://publib.boulder.ibm.com/infocenter/lnxinfo/v3r0m0/topic/liaai/kvmadv/kvmadvstart.htm

Related reference:

Chapter 1, "Scope, requirements, and support," on page 1
This blueprint applies to System x running Linux. You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

Installing the guest operating system

After your guest is created, you can install the operating system for the guest.

If you used **virt-manager**, an installation window should now be open. However, you can use **virt-viewer** to continue the installation or recover the installation window if it is closed.

To install **virt-viewer**, run the following command:

```
# yum install virt-viewer
```

To start using **virt-viewer**, enter the following command:

```
# virt-viewer <guest name>
```

After your installation window opens, you can start installing your KVM guest's operating system as you would on any other physical machine. If you set up the Linux bridge, remember to select it when you set up the network for your guest. If you are using the default virtual network, choosing to use DHCP gives the guest a 192.168.122.x address. If you have a preference as to which 192.168.122.x address the guest is assigned, choose manual configuration and use any unused address from this subnet. Specify your subnet mask as 255.255.255.0, and your gateway as 192.168.122.1, the host's address.

To regain control of the mouse when you are using **virt-viewer** or **virt-manager**, press **Alt+Ctrl**.

Be sure to allow an SSH connection during the installation if you are planning to access your guest through SSH..

For more information about installing Red Hat Enterprise Linux 5, SLES 11, and Windows XP, see the following appropriate document:

- *Red Hat Enterprise Linux 5 Installation Guide* at http://www.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/5.4/html/Installation_Guide/index.html
- *SLES 11 Installation Quick Start* at http://www.novell.com/documentation/sles11/.
- Windows Operating system at http://windows.microsoft.com/en-US/windows/help/install-reinstall-uninstall.

After the guest operating system installation

You can continue to connect to your guest using **virt-viewer**. If you allowed the SSH connection during the guest operating system installation, you can use SSH to connect to your guest. Use either the virtual DHCP assigned address (if you did not set up Linux bridge) or the physical address (if you set up a Linux bridge on the host).

Related reference:

Chapter 1, “Scope, requirements, and support,” on page 1
This blueprint applies to System x running Linux. You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

Related information

You can find additional information about the processes and tools described in these procedures.

- Kernel Based Virtual Machine
  http://www.linux-kvm.org/page/Main_Page
- Red Hat Enterprise Linux 5 Virtualization guide
  http://www.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/5.4/html/Virtualization_Guide/
- KVM Guest Support Status
  http://www.linux-kvm.org/page/Guest_Support_Status
- KVM Guest Support Status
  http://www.redhat.com/docs/manuals/enterprise/RHEL-5-manual/Deployment_Guide-en-US/s2-networkscripts-interfaces-eth0.html.
- Red Hat Enterprise Linux 5.4 Installation guide
  http://www.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/5.4/html/Installation_Guide/index.html
- developerWorks Virtualization Blueprint Community Forum 
  http://www.ibm.com/developerworks/forums/forum.jspa?forumID=1272

Related reference:

Chapter 1, “Scope, requirements, and support,” on page 1
This blueprint applies to System x running Linux. You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Dept. LRAS/Bldg. 903
11501 Burnet Road
Austin, TX 78758-3400
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

Trademarks

IBM, the IBM logo, and ibm.com® are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® and ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at Copyright and trademark information at www.ibm.com/legal/copytrade.shtml

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Java™ and all Java-based trademarks and logos are registered trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.

Printed in USA