XgOS

Remote Boot Guide

Part No.: E52520-01
July 2014

Please
Recycle

Adobe PostScript

Contents

Using This Documentation

This guide explains how to configure remote boot for the Oracle Fabric Interconnect. This document is written for system administrators, authorized service providers, and users who have advanced experience administering servers and networks.

Product Notes

For late-breaking information and known issues about the interconnect, refer to the product notes and release notes at:

`http://www.oracle.com/goto/FABRIC-INTERCONNECT/docs`

Related Documentation

Documentation	Links
All Oracle products	http://www.oracle.com/documentation
Oracle Virtual Networking documentation	http://www.oracle.com/goto/FABRIC-INTERCONNECT/docs
Oracle Fabric Interconnect documentation	http://www.oracle.com/goto/FABRIC-INTERCONNECT/docs
Oracle Solaris 11 OS	http://www.oracle.com/goto/Solaris11/docs

Feedback

Provide feedback about this documentation at:

http://www.oracle.com/goto/docfeedback

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

CHAPTER 1

Remote Booting Overview

This chapter introduces the various ways to implement remote server booting when using Oracle Xsigo Fabric Interconnect to manage your server I/O. It contains the following sections:

Remote Boot Sequence

All computers boot from boot devices, which might include a local SATA disk, a CD-ROM, a USB floppy device, or an Ethernet controller. Xsigo has implemented a ROM BIOS extension for its HCA cards. This extension, called XgBoot, enables you to use Xsigo virtual I/O resources as boot devices for the server. You configure the system BIOS to include XgBoot in the boot order and configure Oracle's Xsigo Fabric Interconnect to make certain virtual resources bootable.

This is a generalized description of the boot process:

1. On power-up, the server's BIOS performs basic hardware initialization and a power-on self-test (POST).
2. BIOS reads the boot sequence, where the XgBoot for the Xsigo HCA should be the first bootable device in the list.
3. The host server establishes a connection to the Fabric Interconnect, where the Fabric Interconnect determines the I/O resource to use and how to set up the communication path from the host server to the boot volume.

4. XgBoot interrogates the boot volume to determine if it is present and bootable. If it is bootable, the Xsigo HCA functions as a hard-disk controller. BIOS begins to treat the Xsigo HCA as the controller for the remote hard disk.

5. The OS loader is installed, which is specific to each OS type:
   - For Linux, the loader is GRand Unified Bootloader (GRUB). This loader resides in the boot sector of a bootable disk. The software responsible for loading the GRUB loader is held in the Option ROM of the HCA and runs in the context of BIOS.
   - For Windows, the loader is NTLoader and is loaded by the same option ROM in the HCA.

   When the loader runs, it typically reads a configuration file from the disk and allows the user to boot the OS in a number of configurations. For GRUB, this file is at `/boot/grub/grub.conf`.

   For Linux, GRUB loads the kernel and the initial RAM disk (`initrd` for Red Hat 4.x and 5.x hosts, and `initramfs` for Red Hat 6.x hosts)) into memory and begin running the kernel. The root file system (`rootfs`) given to the kernel is the `initrd` or `initramfs`. The kernel runs a program in the `initrd` or `initramfs` in the location `/init`. Typically this program is a shell script that loads the kernel modules and mounts the real root file system. See Chapter 2 for more details.

6. The data (Linux kernel or Windows) is sent from the hard disk and loaded into memory. It begins to work and loads all the necessary drivers into the host server.

   Note the boot messages that are displayed as a server connects to a WWN SAN storage device.

```
XgBoot Version 1.3 Built: Fri Dec 14 12:47:53 PST 2007
HCA FW version: 5.2.0
HCA Node Guid: 0x2c90200221d68
Bringing up port 1..
Port 1 bringup successful, LID: 30, SM-LID: 1
Bringing up port 2..
Port 2 bringup FAILED
XSMP session to GUID 0x13970201000201, LID: 1 Successful
IOP connections Successful  GUID: 0x139703010003ca, LID: 6
Number of bootable targets (chassis): 1
WWN: 50:06:01:60:3a:20:1e:83, LUN: 0
```

   You should also look for the message `XgBoot detected PnP BIOS` that indicates the Xsigo option ROM is installed on the HCA and the proper device-failover mechanisms are supported. If there are two Xsigo HCAs in the system with two option ROMs, two `XgBoot detected PnP BIOS` entries appear. Thereafter, the system boots up.

General Configuration Process

Whatever remote booting process you select, you use the following high-level steps:

1. Set up the boot volume.

   You need the boot image available on the network.

2. Configure bootable I/O resources on the Fabric Interconnect.

   For SAN Booting, this is a VHBA configured as bootable. For PXE boot, you use a bootable VNIC. Either resource must be assigned to a Server Profile that is bound to the server that boots remotely.

3. Configure the server.

   HCAs on the server must have compatible firmware and their option ROM must be enabled. See Chapter 8 for instructions on updating firmware.

4. Connect the Server Profile on the Fabric Interconnect to the server.

5. Set the Server Profile to either SAN Boot (`sanboot=`) or iSCSI Bootable (`iscsiboot=`).

Chapters about different kinds of remote booting contain the specific instructions to accomplish this configuration.

Server Profile Delivery to the Server

When a server boots locally, the Fabric Interconnect becomes aware of its presence on one of its InfiniBand links. When a server cannot boot, the Fabric Interconnect becomes aware of something on the link, but has no host name or other information about the server because no communication is possible. Because of this limitation, assigning a Server Profile with the remote booting configuration to a server works differently from the usual assignment.

To assign a Server Profile to a server that boots remotely, you need the GUID of the HCA port over which it boots. You can get this GUID in the following ways:

- When first installing the server, note the HCA GUID ID on the HCA label. Adding one (1) to this number gives you the port GUID for the first HCA port. Adding two (2) to this number gives you the port GUID for the second HCA port.

- After completing the remote-booting configuration, power up the server. Wait until you see the power-up lights both on the server and on the Fabric Interconnect port where the server connects. Then, at an Fabric Interconnect CLI prompt, type the following command:

```
show physical-server
name      guid          descr port               os      version  server-profile
-------------------------------------------------------------------------------
unknown                      2c90200204cbe
```

The number listed under the port heading is the port GUID for the server.

After you have the port GUID, you can use it in assigning the Server Profile, as shown:

```
set server-profile remote-boot-profile connect 2c90200204cbe
set server-profile remote-boot-profile san-boot 2c90200204cbe
set server-profile esx50 iscsi-boot vnic -target-iqn=IQN -lun=iscsi-LUN
```

Types of Remote Booting

The remainder of this book provides specific information and instructions for configuring different types of remote booting.

- Remote Booting for Linux Servers

  See Chapter 3, Chapter 4, and Chapter 5.

- Remote Booting for ESX Servers

  See Chapter 6.

- Remote Booting for Windows Servers

  See Chapter 7.

CHAPTER 2

Xsigo `initrd` or `initramfs`

When you configure a Linux server for remote booting, you provide it with an `initrd` or `initramfs` somewhere on the network. An initial RAM disk (`initrd` or `initramfs`) is a RAM-based file system provided to the kernel at boot time. The `initrd` or `initramfs`'s purpose is to mount the root file system. The format of the `initrd` or `initramfs` is typically a compressed Copy Input Output archive but is dependent on operating system version.

In Red Hat Enterprise Linux 6.0 and later hosts, the `initrd` is actually an `initramfs` (init RAM file system). The `initrd` and initramfs are conceptually and functionally the same. However, you should be aware of the terminology change depending on which version of Linux OS is SAN booted on your host.

This chapter describes the Xsigo `initrd` and explains how to configure its use. It contains the following sections:

`initrd` Functionality

The Xsigo `initrd` is a customized `initrd` for:

- Linux SAN Boot
- PXE boot
- iSCSI Boot

The `initrd` loads the virtual I/O drivers and mounts the root file system using a VHBA or a VNIC.

The Xsigo `initrd` is used to provide diskless operation of a server using a Xsigo VNIC or from a SAN disk using a VHBA. The file is a standard format Linux `initrd` that is supplied by the PXE server or GRand Unified Bootloader (GRUB) along with a Linux kernel. When the kernel boots with the `initrd`, the Xsigo drivers are loaded and the Linux root file system is mounted from a remote SAN disk.

In order to specify which server to mount, the `initrd` reads the kernel command line and looks for Xsigo-specific options. Typically, the Xsigo configuration comes from the Fabric Interconnect. These options are used to specify which VNIC to use as the network interface and where to mount the root file system from.

For SAN Boot the `initrd` can read the kernel command line, or it can use information supplied from the Xsigo Fabric Interconnect.

To see the contents of an `initrd`:

```
zcat xsigo-initrd-kernel-version-i386.img | cpio -t
```

To extract the contents into a directory:

```
cd destdir
zcat xsigo-initrd-kernel-version-i386.img | cpio -uid
```

The Xsigo `initrd` does not include the utilities and other files that you need if you are using multipathing. If you use multipathing, you must add any multipathing requirements to the `initrd` before deploying it. For instructions about modifying the `initrd` for multipathing, see "Modify the initrd for Multipathing With RHEL 5.x Hosts" on page 26.

Also see "SAN Installation Using Linux Installers" on page 17.

`initrd` Scripts

The `initrd` contains a bash script called `init` plus Aikido scripts that perform higher level functions. The Aikido interpreter is supplied as part of the `initrd` in order to run the high-level scripts. The `init` script performs the following duties:

1. Makes device nodes (`/dev/tty`, etc).
2. Loads kernel modules and Xsigo drivers.
3. Waits for the Xsigo drivers to settle.

Settling means to wait for the IB link to come up, to wait for VNICs and VHBAs to appear.

4. Mounts the root file system (`/sbin/init`).

   Information on where to mount the root file system is obtained, one at a time, from the following locations:

   - From kernel arguments (see "Controlling the initrd through Kernel Command-Line Arguments" on page 13)
   - From `/proc/driver/xsvhba/san-info` for SAN and PXE boot

5. If all else fails, drops to a text-based menu system that is used for troubleshooting.

   See "Troubleshooting With bootmenu" on page 7.

Troubleshooting With `bootmenu`

If the root file system cannot be mounted automatically, the `initrd` enters a mode called `bootmenu`. This mode is a CLI-based menu used to troubleshoot problems and try alternatives.

Topics include:

`bootmenu` Main Menu

You can force the boot menu to be run by adding the kernel argument `bootmenu` to command line.

The Main menu looks like this:

```
                        Xsigo Systems Virtual Boot (1.0.0)
-- Boot configuration ----------------------------------------------------------
                                [No boot configuration]

-- Main menu -------------------------------------------------------------------

d. Boot from SAN Disk n. Boot from network  t. Terminal settings
```

```
r. Refresh screen     c. Reset terminal     s. Spawn shell
E. Exit to shell      R. Reboot             P. Power off

Selection:
```

The top part of the screen display allows you to enter and display the current boot configuration. Press a key to invoke a menu option (no return is required). When entering information, TAB moves to the next field. Pressing ENTER moves to the next field and terminates the entry at the last field. You can also use the UP and DOWN arrow keys to move between fields.

SAN Boot Configuration Through `bootmenu`

The `d` option in the Main menu brings up the SAN Boot menu to configure SAN Boot information.

```
                        Xsigo Systems Virtual Boot (1.0.0)
-- Boot configuration ----------------------------------------------------

                              [No boot configuration]

-- SAN Boot menu ---------------------------------------------------------

v. Show VHBAs                    l. Enter LVM configuration
d. Enter SAN disk configuration  b. Boot operating system
r. Refresh screen                q. Return to previous menu

Selection:
```

SAN LVM Information Through `bootmenu`

The `l` option in the SAN Boot menu enables you to enter LVM information.

```
                        Xsigo Systems Virtual Boot (1.0.0)
-- SAN LVM configuration -------------------------------------------------

     Group          VolGroup00_______
     Volume         LogVol00_________
     Mount opts     _________________


-- SAN Boot menu ---------------------------------------------------------

v. Show VHBAs                    l. Enter LVM configuration
```

```
d. Enter SAN disk configuration b. Boot operating system
r. Refresh screen               q. Return to previous menu

Selection:
```

SAN Disk Information Through `bootmenu`

The `d` option in the SAN Boot menu enables you to enter SAN disk information (target and LUN).

```
                        Xsigo Systems Virtual Boot (1.0.0)
-- SAN disk configuration ----------------------------------------------------

     VHBA          vhba1_____________
     Target        3_________________
     WWPN          00:33:13:0a:34:54
     LUN           42________________
     Mount opts    __________________

-- SAN Boot menu -------------------------------------------------------------

v. Show VHBAs                   l. Enter LVM configuration
d. Enter SAN disk configuration b. Boot operating system
r. Refresh screen               q. Return to previous menu
Selection:
```

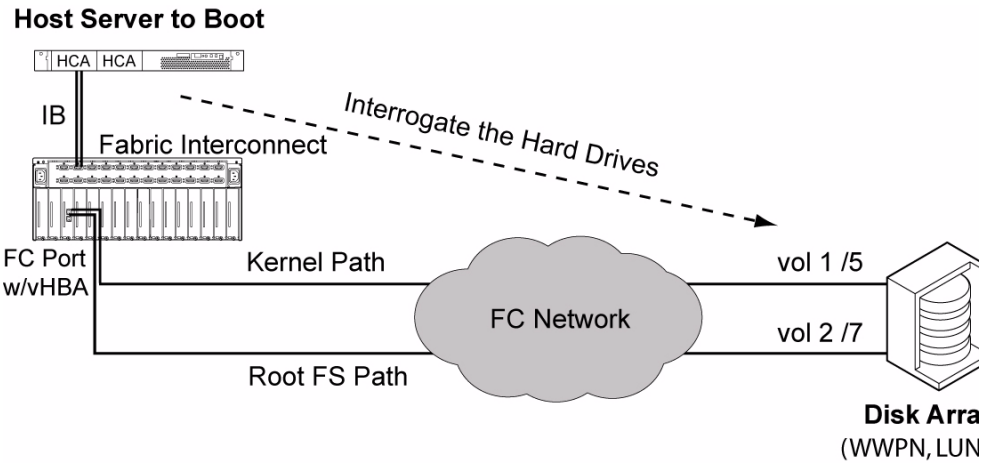
CHAPTER 3

Linux Server SAN Boot

This chapter explains how to set up SAN Boot for a Linux server using virtual HBAs. It contains the following sections:

Linux SAN Boot Topology

SAN Boot allows you to boot a Linux host server from a SAN volume accessed through a VHBA. The remote disk to boot from is identified by a target World Wide Port Name (WWPN) and Logical Unit Number (LUN) on a storage disk array device. SAN Boot allows Linux hosts with an InfiniBand HCA to connect to Oracle's Xsigo Fabric Interconnect, and receive their boot information from a disk connected to the host by a Xsigo VHBA.



`initrd` and Fabric Interconnect Commands

You can send instructions to the `initrd` through the Fabric Interconnect, by using the information model in the Fabric Interconnect (entered via the command-line interface), which communicates with the host's VHBA driver (`/proc/driver/vhba/san-info`). See "Supporting SAN Boot Through the CLI" on page 14.

For example:

```
set server-profile myserver san-boot myvhba 1:2:3:4:5:6:7:8 42
```

If you build a SAN Boot configuration for a Server Profile on an Fabric Interconnect, the VHBA driver creates a file called `/proc/driver/xsvhba/san-info` containing information similar to the kernel arguments. The `initrd` can read this file and determine the root file system location from it.

Controlling the `initrd` through Kernel Command-Line Arguments

Sometimes users have an existing system for SAN Boot and prefer not to use the Xsigo Fabric Interconnect for SAN Boot configuration. Instead, kernel arguments in `grub.conf` are used to specify where devices should be mounted. Multiple devices are supported, where multiple `sanboot=` arguments are included on the command line. The following information overrides the information provided by the VHBA driver.

Settling Wait Times

Some kernel arguments control aspects of the settle facility in the `initrd`. Settling is used to wait for Xsigo's drivers to initialize.

The options are:

```
sanwait[=secs]
```

This option waits for VHBAs for *secs* seconds. A value of 0 means no wait. If no value is specified, then the default of 30 seconds is used.

```
netwait[=secs]
```

This option waits for VNIC for *secs* seconds. A value of 0 means no wait. If no value is specified, then the default is used (30 seconds).

If both of these arguments are not present, then both are deemed to be enabled by default (wait for net and SAN).

When booting from SAN, the default behavior is to wait for both VNICs and VHBAs to settle. Because there might be no VNICs present in the Server Profile, time is wasted by waiting for them. You can specify that the `initrd` waits for only the VHBA by adding only `sanwait` to the kernel command line.

Arguments for Troubleshooting

`bootdebug` – Turns on debugging detail in the `initrd`.

`bootmenu` – Runs the bootmenu instead of automatically booting. See "Troubleshooting With bootmenu" on page 7 for more details.

`emergency` – Runs `bash` after mounting the root file system. Do not run the system `init` scripts.

`init=/bin/bash` – Runs `bash` after mounting the root file system.

`single` –Boots to single-user mode instead of the default run-level configured in `/etc/inittab`.

Supporting SAN Boot Through the CLI

Review the following topics to learn more about SAN Boot.

SAN Boot Roles

SAN Boot operates in three different roles (phases):

- **Load** – Loads the operating system and `initrd` (in the case of Linux) from a SAN disk. The kernel and `initrd` are located and installed into memory.
- **Mount** – Mounts the root file system on a SAN disk. It might or might not be the same disk as the operating system was loaded from. The following mount types are available: logical volume management (LVM) and direct. In the mount role, `initrd` starts and mounts the device file system corresponding to the VHBA specified. The target disk and its root file system inside the OS are mounted.
- **Loadmount** – Both roles are performed, and all files come from the same location. That is, the same target disk contains the boot image and the root file system. This common use case loads the kernel and `initrd` into memory, where `initrd` starts and mounts the device file system corresponding to the VHBA specified.

Syntax for SAN Boot

```
set server-profile name san-boot [vhba|none] wwpn lun
show server-profile server san-boot
```

SAN Boot Parameter Descriptions

`set server-profile` *name* – Identifies a named Server Profile to boot from.

san-boot *vhba*|none – Creates a boot object and associates it with a named VHBA. The SAN Boot object can use only the VHBAs that are available to the Server Profile. You must have a previously created VHBA and scanned for available LUNs.

wwpn – Available target WWPN, as used for a physical hard disk.

lun – Available LUN ID on the target, as used for a logical hard disk.

`none` – Removes the SAN Boot association from a Server Profile.

Note – For more information on LUNs and targets, refer to the XgOS Command-Line Interface User Guide.

`show server-profile` *server* `san-boot` – Displays SAN Boot information for a Server Profile.

In the following examples:

- The server name is `myserver`.
- The VHBA used for SAN Booting is `vhba1`.
- The disk to be mounted is LUN 42 on 1:2:3:4:5:6:7:8

To load using the Fabric Interconnect's information model:

```
set server-profile myserver san-boot vhba1 1:2:3:4:5:6:7:8 42
```

To display the SAN Boot facilities on an Fabric Interconnect:

```
show server-profile myserver san-boot
server    role     vhba   mnt-type  lvm-grp      lvm-vol   dev            mnt-
opts   disks
-------------------------------------------------------------------------------
-
myserver loadmount vhba1                                    1:2:3:4:5:6:7:8(42/
LM)
1 record displayed
```

Acknowledging SAN Boot Issues

SAN Boot is complex. This section presents some issues to keep in mind.

Xsigo Drivers On A SAN-Booted Server

The environment on a SAN Booted server is no different from a standard server with respect to the Xsigo drivers. On a SAN-booted server, the drivers are installed or upgraded by using `rpm -Uvh` *driver-name*.

SAN Boot Time

When you are using SAN boot for your Linux server, the boot process might pause for several minutes at the starting `udev` message. You can reduce the duration of this pause by setting the `udevtimeout` kernel parameter to a lower value.

To reset the `udevtimeout` parameter when using GRUB, modify the entry as follows:

```
// initial entry
kernel /vmlinuz-2.6.18-53.el5 ro root=/dev/VolGroup00/LogVol00 rhgb quiet
```

```
// modified entry
kernel /vmlinuz-2.6.18-53.el5 ro root=/dev/VolGroup00/LogVol00 rhgb quiet
udevtimeout=5
```

Installing Linux on the SAN

You can use any of the following approaches to install the SAN volume:

- "Perform an Installation Over Multipath VHBAs Using XgBoot" on page 23
- "Modify the initrd for Multipathing With RHEL 5.x Hosts" on page 26

SAN Installation Using Linux Installers

The Anaconda installer is supported for Red Hat servers. Xsigo does not modify the installer. Instead Xsigo uses the version supplied by the OS provider and adds in Xsigo's driver disks. A Xsigo driver disk is created for a specific kernel for a specific OS.

The xsigo-boot tar archive contains `.img` files and scripts used to support booting the Xsigo drivers.

TABLE 3-1 Files and Scripts Used for the Xsigo `initrd`

File Name	Purpose
`xsigo-boot-`*`kver-version-arch`*`.tar`	Boot file compiled for a specific kernel version.
`xsigo-initrd-`*`kver-version-arch`*`.img`	A Linux `initrd` with Linux that uses the Xsigo drivers. The `initrd` is used with Red Hat Enterprise Linux distributions earlier that version 6.X
`xsigo-initramfs-`*`kver-version-arch`*`.img`	A Linux `initramfs` with Linux that uses the Xsigo drivers. The `initramfs` is conceptually similar to an `initrd`, but the `initramfs` is used with Red Hat Enterprise Linux distributions equal to or later that version 6.X
`xsigo-rhdd-`*`kver-version-arch`*`.img`	A Red Hat Anaconda installer driver disk.
`xg-insert-dd`	For RHEL5, if you want to avoid using a CD to load the driver disk, you can insert the disk into the Anaconda `initrd` using the script `xg-insert-dd`. It creates a new `initrd` prefixed with the string `xsigo-`. The `xg-insert-dd` script is used on Red Hat Enterprise Linux hosts earlier than RHEL 6 Update 1, which is the first RHEL 6 platform supported. (RHEL 6 Update 0 is not supported.)
`xg-insert-dd-ext2`	For RHEL 4, inserts the driver disk into the Anaconda `initrd`.

You can add the Xsigo driver disk into the installer in the following ways:

- Boot from a CD-ROM.

  Create an ISO image that provides the required drivers to the installer during runtime. This is supported on RHEL 4.x and 5.x and earlier hosts, but is not supported on RHEL 6.X and higher hosts.

- Boot from the `initrd` itself.

  For RHEL 4 and 5 servers, Xsigo provides a script that takes a standard `initrd` (for example the Anaconda `initrd` from the Red Hat installation disk) and then inserts the Xsigo drivers (as a driver disk) into that `initrd`. Thereafter, enter the Xsigo Anaconda `initrd` into the PXE boot system as the `initrd`. As the kernel boots, it loads the Xsigo drivers. For RHEL 6.x server, Xsigo you use the `append` command to add the Xsigo drivers to the `initrd`.

- On the Fabric Interconnect, set a VHBA to load using a specific WWPN and LUN ID.

  The simplest setup is to use the same disk for loading the operating system and mounting the root file system – the loadmount role. For example:

```
set server-profile name san-boot vhba-name wwpn lun-ID
```

The `initrd` loads from the mounted SAN Boot object. Once installed, the SAN disk appears as any other disk in the system.

▼ Perform a CD Installation of 5.0.x Host Drivers on RHEL 5 Hosts

This example illustrates how to perform the SAN installation from CD-ROM using Linux installers. The procedure is comprised of the following general phases:

- create a temporary workspace (a `tmp` directory)
- get the extracted `xsigo-boot` image
- use the `xsigo-insert-dd` script to inject the Xsigo host drivers into the RHEL `initrd`
- create an ISO image of the `initrd` with the Xsigo host drivers in it. Afterwards, you can move the new ISO image to the SAN where it is used to SAN or iSCSI boot a connected host.

1. **Download the corresponding RHEL iso and the `xsigo-boot` tar package to a server.**

2. **Create a directory /tmp/remaster for doing the remaster of the RHEL 5 iso to inject the xsigo modules.**

```
#mkdir /tmp/remaster
```

3. **Create subdirectories in this directory.**

```
#mkdir /tmp/remaster/iso
#mkdir /tmp/remaster/extracted
#mkdir /tmp/remaster/xsigo
```

4. **Change to the directory /tmp/remaster/xsigo and extract the xsigo-boot tar package.**

```
#cd /tmp/remaster/xsigo
#tar -xvf /builds/drivers/5.0.1.LX3D/redhat/xsigo-boot-2.6.18-
238.el5-5.0.1.LX3D-x86_64.tar
```

5. **Mount the ISO to extract the contents.**

```
#mount -o loop /export/isos/testing/rhel-server-5.6-x86_64-
dvd.iso /tmp/remaster/iso
```

6. **Copy the contents to the extracted subdirectory.**

```
# rsync -av /tmp/remaster/iso/ /tmp/remaster/extracted/.
```

7. **Unmount the** `iso` **directory.**

```
# umount /tmp/remaster/iso
```

8. **Change directory to** `isolinux` **in the** `extracted` **folder.**

```
# cd /tmp/remaster/extracted/isolinux
```

9. **Use the** `xg-insert-dd` **script to insert the xsigo modules into the** `initrd.img`**.**

```
#/tmp/remaster/xsigo/xg-insert-dd /tmp/remaster/xsigo/xsigo-rhdd-2.6.18-
238.el5-5.0.1.LX3D-x86_64.img initrd.img
```

This generates a `xsigo-initrd.img` file

10. **Move the original** `initrd.img` **to** `initrd.img.orig`**.**

```
# mv /tmp/remaster/extracted/isolinux/initrd.img
/tmp/remaster/extracted/isolinux/initrd.img.orig
```

11. **Move the** `xsigo-initrd.img` **to** `initrd.img`**.**

```
# mv /tmp/remaster/extracted/isolinux/xsigo-initrd.img
/tmp/remaster/extracted/isolinux/initrd.img
```

12. **Come out of the** `isolinux` **directory.**

```
# cd /tmp/remaster/extracted
```

13. **Now the xsigo modules are loaded in to the** `initrd.img`**, use the** `mkisofs` **to make a** `iso` **from this contents.**

```
#/usr/bin/mkisofs -R -J -T -o /tmp/xsigo-rhel-server-5.6-x86_64-dvd.iso
-b isolinux/isolinux.bin -c isolinux/boot.cat -no-emul-boot
-boot-load-size 4 -boot-info-table -R -m TRANS.TBL .
```

The remastered ISO is available in `/tmp/xsigo-rhel-server-5.6-x86_64-dvd.iso`.

14. **Burn this ISO and use this for Installing on VHBA LUN or VNIC provided iSCSI LUN.**

▼ Perform a RHEL 5 SAN Boot Installation and Configuration through `xg-insert-dd` for PXE Boot

For Red Hat to install the Xsigo virtual I/O, you build a custom `initrd` from the default `initrd` that Red Hat provides. To support creating the custom `initrd`, Xsigo provides `xg-insert-dd` which is a script that injects the Xsigo host drivers into the default `initrd` provided by Red Hat. The `xg-insert-dd` script is required to allow the Red Hat server to recognize the Xsigo devices and attach the correct host drivers. Without the custom `initrd`, the Xsigo devices, including the bootable VNIC and VHBA, are marked as unknown devices, and eventually get marked not bootable and are bypassed while the Red Hat `initrd` is loading. If the bootable VNIC and VHBA are marked not bootable, the server is not able to SAN boot.

To use the `xg-insert-dd` script, you need the following software:

- Xsigo boot drivers (the rhdd image)
- Stock `initrd` (provided by Red Hat on your installation media)
- `xg-insert-dd` script

Copy them all to a temporary working directory to create the custom `initrd`.

The `xg-insert-dd` script has the following syntax:

```
sh xg-insert-dd {-iscsiboot[=boot.tgz]] dd.img anaconda
```

where:

- *dd.img* is the Xsigo boot driver that you want to insert into the default `initrd`
- anaconda is the default Red Hat `initrd`

Note – Disregard the -iscsiboot=boot.tgz option.

When `xg-insert-dd` is run, it opens the default `initrd` for editing, places the Xsigo host drivers inside, then repacks the `initrd` as a custom `initrd` with the xsigo- prefix. When the server uses this custom `initrd`, the Xsigo drivers are loaded at boot time.

1. **On the host server, install an HCA containing Xsigo's SAN Boot option ROM.**

   If the HCA does not have the correct option ROM, see "" on page 103 for instructions about how to install it.

Note – Many SAN Boot-capable servers exist. However only certain BIOSes support a certain number of HCAs with the Xsigo option ROM. Consult Xsigo Support for the hardware compatibility matrix.

---

2. **Download the appropriate xsigo-boot-*kernel*.tar file unique to your distribution and architecture. RHEL5 32-bit is used in this example.**

   Untar `xsigo-boot-`*kernel*`.tar` file and create a PXE Boot image by using the `xsigo-insert-dd` tool and following the prompts presented while running `xsigo-insert-dd`.

   From Linux:

```
tar xvf xsigo-boot-kernel.tar
```

3. **From the Fabric Interconnect, create a Server Profile for the server performing the RHEL5 install to SAN storage.**

   For the physical connection, use the host's GUID.

   You should have the GUID address available. The GUID is commonly displayed on the HCA packaging. The GUID is also displayed during the server's POST, so you can get the GUID during the server's POST.

```
add server-profile pokemon 2c02123a5303
```

   For more about assigning a Server Profile to a server that cannot boot yet, see "Server Profile Delivery to the Server" on page 3.

4. **Add the VHBA to the Server Profile:**

```
add vhba vhba0.pokemon 3/1
```

5. **Confirm that LUNs are visible to the Server Profile by running a rescan and showing target LUNs.**

   At this point you have already configured the storage side (such as, zoning, disk allocation, and so on).

```
set vhba vhba0.pokemon prescan
show vhba vhba0.pokemon targets
```

6. **As an option, remove all physical hard drives from server. This step is not mandatory.**

7. **Set up the Server Profile to do SAN Boot.**

   The `set server-profile` *name* `san-boot` command provides the server with the LUN that the host needs to connect to boot up.

```
set server-profile pokemon san-boot vhba0 50:06:01:60:3A:20:1E:83
```

8. **Log in to your virtual terminal and use the `initrd` found in the TAR ball.**

9. **Install the IB stack first:**

```
rpm -ivh kernel-ib.rpm
```

10. **Then install the Xsigo host driver RPM:**

```
rpm -ivh xsigo-host-drivers.rpm
```

11. **After installation is complete, you copy over the `xsigo-initrd-`*kernel-version*`-i386.img` file into the SAN storage `/mnt/sysinstall/boot` directory and modify the GRUB configuration for this `initrd` file.**

    Then, edit the `initrd` line to read the following:

```
initrd /xsigo-initrd-kernel-version-i386.img
```

12. **After adding the preceding line to the `initrd`, write the changes (`:w`) and quit the editor (`:q`).**

13. **Reboot the server.**

▼ Perform an Installation Over Multipath VHBAs Using XgBoot

This example shows how to SAN install a Red Hat Enterprise Linux on the SAN. For this example, RHEL 6u3 is used.

To complete this procedure, you need the following:

- Two Fabric Interconnects connected to an RHEL server
- A fully configured and functional PXE server and TFTP server
- A fully configured and functional storage target, which at least one LUN available. The LUN must be large enough to contain the RHEL image on it completely.
- A RHEL 6u3 server with an HCA installed and running the latest version of XgBoot. This requirement allows the server to detect the Fabric Interconnect

1. **If you have not already created a Server Profile, do so now.**

   For illustrative purposes, this procedure uses the Server Profile named `sar`.

```
add server-profile sar
```

2. **Repeat the previous step on the second Fabric Interconnect.**

3. **Add a bootable VNIC from one of the Fabric Interconnects:**

```
admin@lanai[xsigo] add vnic pxevnic.sar 2/1 -boot-capable=true
```

   As an option, you can display the VNIC tor

```
admin@lanai[xsigo] show vnic pxevnic.sar
name              state       mac-addr                ipaddr       if        if-
state       ha-state       local-id       type       vlans       qos       flags
-------------------------------------------------------------------------------
pxevnic.sar       up/up       00:13:97:30:E0:1B                    2/1
up                          0                       none       --        -b-----
```

4. **Add two VHBAs (a multipath VHBA) from each Fabric Interconnect and make sure each multipath VHBA points to the same LUN.**

   For example to add the primary VHBA:

```
admin@lanai[xsigo] add vhba vhba1.sar 1/1 -wwn-id=601
```

   As an option, you can display the VHBA to verify that it is correctly created:

```
admin@lanai[xsigo] show vhba vhba1.sar target
vhba                      name                  wwnn
wwpn                                          lun-ids
------------------------------------------------------------------------------
vhba1.sar                                  50:0A:09:80:88:2A:39:9F
50:0A:09:81:88:2A:39:9F                       99
1 record displayed
```

5. **Set the Server Profile to support SAN Boot functionality.**

```
admin@lanai[xsigo] set server-profile sar san-boot vhba1 50:0A:09:81:88:2A:39:9F
99 mount lvm
```

   As an option, you can display the Server Profile to verify that it was correctly created.

```
admin@lanai[xsigo] show server-profile sar san-boot
server          role              vhba          mnt-type          lvm-grp
lvm-vol           dev           mnt-opts           disks
------------------------------------------------------------------------------
sar       loadmount     vhba1      lvm
50:0A:09:81:88:2A:39:9F(99/LM)
1 record displayed
```

6. **On the second Oracle Fabric Interconnect, repeat the previous step to create the secondary VHBA.**

   For example, to create the second VHBA:

```
admin@maui[xsigo] add vhba vhba2.sar 1/1 -wwn-id=601
```

   As an option, you can display the VHBA to verify that it is correctly created.

```
admin@maui[xsigo] show vhba vhba2.sar target
vhba                      name                  wwnn
wwpn                                          lun-ids
```

```
--------------------------------------------------------------------------------
vhba2.sar                                  50:0A:09:80:88:2A:39:9F
50:0A:09:81:88:2A:39:9F                           99
1 record displayed
```

7. **Set the Server Profile to support SAN Boot functionality.**

```
admin@maui[xsigo] set server-profile sar san-boot vhba2 50:0A:09:81:88:2A:39:9F
99 mount lvm

admin@maui[xsigo] show server-profile sar san-boot
server         role             vhba          mnt-type         lvm-grp
lvm-vol          dev          mnt-opts          disks
--------------------------------------------------------------------------------
sar            loadmount        vhba2        lvm
 50:0A:09:81:88:2A:39:9F(99/LM)
1 record displayed
```

8. **Log in to the PXE server and make sure that the label contains the multipath option (mpath) so that the installer looks at the LUNs from the two VHBAs as a Multipath Disk.**

   If needed, edit the label to make it as follows:

```
label RHEL6U3-505LX
kernel vmlinuz-rhel6u3-x86_64
append initrd=initrd-rhel6u3-x86_64.img,xsigo-rhdd-2.6.32-279.el6.x86_64-
5.0.5.LX-x86_64.img  mpath  network
```

9. **Log into the server, and enter the BIOS.**
10. **In BIOS, change the boot sequence so that XgBoot is at the top of the boot devices list.**

    By doing so, you ensure that XgBoot boots through the bootable VNIC.

11. **Boot from the label and continue the installation.**
12. **When selecting the hard disk on which to install, make sure it shows the LUNs as the multipath disk (for example, either** `/dev/mapper/mpath0` **or** `/dev/dm-0`**).**
13. **Continue with the manual installation and before the reboot install the kernel-ib and Xsigo host drivers RPMs.**
14. **Reboot the server, and this time it boots from the Multipath LUN by checking** `multipath -l` **command.**

▼ Modify the `initrd` for Multipathing With RHEL 5.x Hosts

Xsigo provides a multipath-capable RPM file (multipath.rpm). If your Red Hat servers are running in a multipath environment, you install `multipath.rpm`. Previously, you were required to edit the `initrd` to allow multipathing, but this is no longer required. Instead, just make sure to install multipath.rpm. You no longer need to make changes to Device Mapper.

1. **Install `multipath.rpm` before rebooting the server.**
2. **Make sure the Xsigo devices that the server boots from are not in the Blacklist file.**

   Red Hat configurations black list every device by default. As a result, all devices are ignored. Depending on your network, you might need to comment out the entire Blacklist, or edit the Blacklist to create a custom stanza that causes only some boot devices to be ignored. For information about editing or customizing the Blacklist file, consult the documentation that accompanied your multipath software.

Enabling SAN Boot for RHEL 6.1 and Later Hosts

Review and perform the following topics to enable SAN Boot for RHEL 6.1 and later hosts:

Comparison to RHEL 6.0 and Earlier Hosts

The procedure for configuring SAN Boot on a Red Hat Enterprise Linux host is the same as for Red Hat 5.x hosts, with the exception of how the Xsigo host drivers are included in the RHEL 6.1 `initramfs`. (The `initramfs` is conceptually and functionally the same as the `initrd` in RHEL 5.X distributions and earlier.)

With RHEL 6.1 hosts, the process of adding the Xsigo host drivers to make a modified `initrd` has been made much simpler through the use of the `append` command. This command allows you to simply add the Red Hat disk drivers to the 6.1 OS image when the server is PXE or SAN Booting. The `append` command is typed on the PXE Boot Server that booting hosts connect to during their PXE or SAN Boot phase.

The `append` Command (SAN Boot)

The `append` command takes the place of the `xg-insert-dd` tool that you use with Red Hat 5.x hosts. As a result, the `xg-insert-dd` tool is not used with RHEL 6.x hosts, but still is required with RHEL 5.x hosts. The `append` command performs the same function as the `xg-insert-dd` does for RHEL 5, but in a more streamlined way. To inject the Xsigo host drivers into the RHDD, use the `append initrd` option:

```
append initrd=initramfs-string,xsigo-rhdd-image ksdevice=device-name network
```

where:

- *initramfs-string* is the Red Hat OS image and the Xsigo boot driver (RHDD) that you want to insert into the Red Hat OS image separated by a comma only--no blank.
- *device-name* specifies the particular interface that you want the installer (for example, eth2) to use for the kickstart process.

You use the base RHEL 6.1 `initramfs` and the Xsigo RHDD image with the `append` command.

```
append initrd=initrd-rhel6u1-x86_64.img,xsigo-rhdd-2.6.32-131.0.15.el6.x86_64-
3.6.9.LX3-x86_64.img ksdevice=eth2 network
```

Note – Pay close attention to the syntax, especially the comma (,) separator between the RHEL image and the Xsigo RHDD. There should be no blank spaces between the RHEL image and the Xsigo RHDD--only a comma. If spaces exist, or the comma is not present, the command fails and the Xsigo host drivers are not appended to the RHEL image. As a result, the server does not SAN Boot.

▼ Perform a RHEL 6.1 SAN Boot Installation and Configuration for PXE Boot

For Red Hat to install the Xsigo virtual I/O, you add the Xsigo modules into the default `initramfs` that Red Hat provides by using the `append initrd` command to append the Xsigo host drivers to the default `initramfs`. The Xsigo host drivers must be added to the `initramfs` to allow the Red Hat server to recognize the Xsigo devices and attach the correct host drivers.

If the Xsigo devices bootable VNIC and VHBA are not in the `initramfs`, the Linux OS temporarily marks them unknown devices, and they eventually are marked not bootable. When the Xsigo VNIC and VHBAs are marked not bootable, they are bypassed while the Red Hat `initramfs` is loading, and the server cannot SAN boot from the Xsigo devices.

To use the `append` command, you need the following software:

- Xsigo boot drivers (the RHDD image)
- Stock `initrd` (provided by Red Hat on your installation media)

1. **On the host server, install an HCA containing Xsigo's SAN Boot option ROM.**

   If the HCA does not have the correct option ROM, see "" on page 103 for instructions about how to install it.

Note – Many SAN-boot-capable servers exist. However only certain BIOSes support a certain number of HCAs with the Xsigo option ROM. Consult Xsigo Support for the hardware compatibility matrix.

---

2. **Download the appropriate xsigo-boot-*kernel*.tar file unique to your distribution and architecture.**

3. **Create the modified `initramfs` with the `append initrd` command.**

4. **From the Fabric Interconnect, create a Server Profile for the server performing the RHEL6 install to SAN storage.**

   For the physical connection, use the host's HCA GUID.

   You can get the server's GUID from the HCA packaging. Or, the GUID is also displayed during the server's power on self-test (POST), so you can watch for it during the server's POST. You should have the server's GUID available for this part of the procedure.

```
add server-profile pokemon 2c02123a5303
```

   For more about assigning a Server Profile to a server that cannot boot yet, see "Server Profile Delivery to the Server" on page 3.

5. **Add a VHBA to the Server Profile:**

```
add vhba vhba0.pokemon 3/1
```

6. **Confirm that LUNs are visible to the Server Profile by running a rescan and showing target LUNs.**

   At this point you have already configured the storage side (such as, zoning, disk allocation, and so on).

```
set vhba vhba0.pokemon rescan
show vhba vhba0.pokemon targets
```

7. **As an option, remove all physical hard drives from server.**

   This step is not mandatory.

8. **Set up the Server Profile to do SAN Boot.**

   The `set server-profile` *name* `san-boot` command provides the server with all the information that it needs to connect to SAN storage upon boot up (such as, the VHBA to use, WWPN and LUN to connect to, volume group, logical volume to use, and so on).

```
set server-profile pokemon san-boot vhba0 50:06:01:60:3A:20:1E:83
0
```

9. **After installation is complete but before rebooting the server, you install the kernel `IB.rpm` file:**

```
rpm -ivh kernel-ib.rpm
```

10. **After the `kernel-ib` rpm is successfully installed, install the Xsigo host driver RPM:**

```
rpm -ivh xsigo-host-drivers.rpm
```

11. **Reboot the server.**

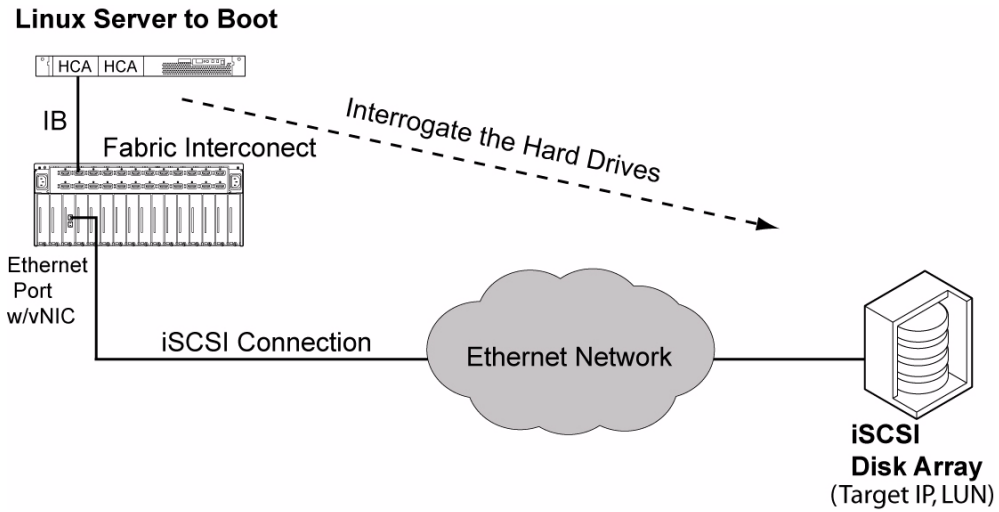
CHAPTER 4

Linux Server iSCSI Boot

This chapter explains how to set up iSCSI booting. It includes the following sections:

Linux iSCSI Boot Topology

Oracle's Xsigo Fabric Interconnect supports booting a Red Hat Linux 6u1 server over a VNIC using an iSCSI connection. The remote disk to boot from is identified by a target iSCSI Qualified Name (IQN) and Logical Unit Number (LUN) on an iSCSI storage disk array.



Understanding iSCSI Booting

Review the following topics to learn more about iSCSI Boot.:

- "iSCSI Boot Configuration Overview" on page 32
- "iSCSI Boot Terminology" on page 33
- "iSCSI Boot Caveats" on page 33

iSCSI Boot Configuration Overview

When configuring iSCSI boot, you perform the same general steps as for any remote booting setup:

1. Configure the host server with drivers and firmware to enable remote booting.
2. Create the bootable Server Profile and VNIC.
3. Create the iSCSI LUN, which is where the iSCSI PXE boot image is installed.
4. Make sure that the PXE Boot server is correctly set up.

   You need access to the default config file. This document assumes that your PXE Boot server is already configured. Details about configuring your PXE Boot server are out of the scope of this document.

iSCSI Boot Terminology

TABLE 4-1 iSCSI Boot Terms

Term	Definition
initiator	The host server that is booting over an iSCSI connection
target	The iSCSI array
IQN	An iSCSI qualified name of an initiator or target
target IP	The IP address of the target filer or array. This is optional if you are using DHCP addressing on the VNIC.

iSCSI Boot Caveats

- Linux hosts can iSCSI Boot if they are running Red Hat Enterprise Linux (RHEL) 6u1 or later unless otherwise documented.
- To get the initiator IQN, the iSCSI boot profile must be created on Oracle's Xsigo Fabric Interconnect and pushed to the server that is iSCSI booted. The initiator IQN is not learned until the Server Profile is created.
- When installing the iSCSI Boot PXE image onto the iSCSI LUN, only one method is supported. See "The append Command (iSCSI Boot)" on page 35. Other methods exist, for example CD installs, but are not currently supported by Xsigo.

I/O Resource Configuration

These topics describe the commands used to configure I/O resources:

Command Syntax

```
set server-profile name iscsi-boot [vnic|none] targetIP

set server-profile name iscsi-boot [vnic|none] targetIP [mount {direct /dev/node |
LABEL=label] | lvm group-name

set server-profile name iscsi-boot [vnic|none] targetIP volume-name}]

show server-profile server iscsi-boot [-detail]-target-iqn=targetIqn
```

Command Parameter Descriptions

`set server-profile` *name* – Identifies a named Server Profile to boot from.

`iscsi-boot [`*vnic*`|none]` – Creates a boot object and associates it with a named VNIC. The iSCSI boot object can use only the VNICs that are available to the Server Profile. You must have a previously created the VNIC.

`none` – Removes the iSCSI boot object from a Server Profile.

targetIP – Specifies the IP address of the storage array or filer. This is not the specific boot volume.

`show server-profile` *server* `iscsi-boot [-detail]` – Displays iSCSI boot information for a Server Profile.

`-target-iqn` – Specifies the iSCSI qualified name of the target array. By default, this IQN is not offered automatically, so you set up your DHCP server to offer the IQN. By default For static addressing, always include this modifier.

Command Optional Modifiers

`-lun` – logical unit number of the boot volume. The default is LUN 0.

`-port` – port number on target filer or array. The default is port 3260.

`-protocol` – transport protocol. Leave this qualifier set to 6, TCP, unless you have a specific reason to change it.

`-target-iqn` – target iSCSI qualified name. This is optional when using DHCP addressing but required when using static addressing. If you use DHCP, when the `initrd` starts it runs the DHCP client to get the target's address. If you use static addressing, you must provide the target IQN on the command line.

`-target-pg` – a standard iSCSI target portal group

Creating the iSCSI Boot PXE Image for RHEL 6.1

With RHEL 6.1 hosts, you will add stanzas to the default config file on the PXE boot server The stanzas will add the Xsigo host drivers. This process occurs through the use of the `append` command. This command allows you to simply add the Xsigo VNIC devices to the RHEL 6.1 OS image when the RHEL 6.1 server is PXE or iSCSI Booting. The `append` command is typed on the PXE Boot Server that booting hosts connect to during their PXE or iSCSI Boot phase, and the command allows you to add the Xsigo-specific information to the default config file.

Review and perform the following topics to create an iSCSI Boot PXE image.

- "The append Command (iSCSI Boot)" on page 35
- "Perform a RHEL 6.1 iSCSI Boot Installation and Configuration for PXE Boot" on page 36

The `append` Command (iSCSI Boot)

The `append` command allows you to place Xsigo devices into the default Red Hat 6.1 or later OS image. This procedure occurs on the PXE server default config file.

Note – The `append` command is the only method currently supported by Xsigo. Other methods, such as CD ROM installs, are not supported.

To inject the Xsigo host drivers into the RHDD, use the `append initrd` option:

```
append initrd=initramfs-string xsigo-rhdd-image ksdevice=device-name network
```

where:

- *initramfs-string* is the Red Hat OS image and the Xsigo boot driver (RHDD) that you want to insert into the Red Hat OS image separated by a comma only--no blank.
- *device-name* specifies the particular interface that you want the installer (for example, eth2) to use for the kickstart process.

You use the base RHEL 6.1 `initramfs` and the Xsigo RHDD image with the `append` command.:

```
append initrd=initrd-rhel6u1-x86_64.img,xsigo-rhdd-2.6.32-131.0.15.el6.x86_64-
3.6.9.LX3-x86_64.img ksdevice=device-name kspath=path-to-kickstart-file network
```

Note – Pay close attention to the syntax, especially the comma (,) separator between the RHEL image and the Xsigo RHDD. There should be no blank spaces between the RHEL image and the Xsigo RHDD--only a comma. If spaces exist, or the comma is not present, the command fails and the Xsigo host drivers are not appended to the RHEL image. As a result, the server will not present iSCSI disks to which you want to install.

▼ Perform a RHEL 6.1 iSCSI Boot Installation and Configuration for PXE Boot

For Red Hat to install the Xsigo virtual I/O, you add the Xsigo modules into the default `initramfs` that Red Hat provides by using the `append initrd` command to append the Xsigo host drivers to the default `initrd`. The Xsigo host drivers must be in the `initrd` to allow the Red Hat server to recognize the Xsigo devices and attach the correct host drivers. Red Hat Enterprise uses the `initrd` for install, but afterward, uses the `initramfs` for subsequent boots.

If the Xsigo bootable VNIC is not in the `initramfs`, the Linux OS temporarily marks it as an unknown device, and it eventually is marked not bootable. When the Xsigo VNIC is marked not bootable, it is bypassed while the Red Hat `initrd` is loading, and the server cannot iSCSI boot from the Xsigo devices.

To use the `append` command, you will need the following software:

- Xsigo boot drivers (the RHDD image)
- Stock `initrd` (provided by Red Hat on your installation media)

Note – To complete this procedure, you need specify the initiator IQN that is allowed to access the LUN where the boot `initramfs` is located. To get the initiator IQN, you must create and deploy the ISCSI boot Server Profile. The initiator IQN is learned only after the iSCSI boot Server Profile is created.

1. On the host server, install an HCA containing Xsigo's iSCSI Boot option ROM.

Note – Many iSCSI-boot-capable servers exist. However only certain BIOSes support certain HCAs with the Xsigo Option ROM. Consult Oracle Support for the hardware compatibility matrix.

2. **On the PXE Boot server, download the appropriate** `xsigo-boot-`*kernel*`.tar` **file unique to your distribution and architecture.**

3. **On the PXE Boot Server, untar** `xsigo-boot-`*kernel*`.tar` **file and copy the** `xsigo-rhdd-`*kernel*`.img` **to the PXE Boot server.**

   From Linux:

```
tar xvf xsigo-boot-kernel.tar
```

4. **On the PXE server, append the Xsigo information to the default config file, as documented in** "The append Command (iSCSI Boot)" on page 35**.**

5. **On the Fabric Interconnect, create a Server Profile for the server performing the RHEL6 install to storage.**

   For the physical connection, use the host's HCA GUID.

   You can get the server's GUID from the HCA packaging. Or, the GUID is also displayed during the server's power on self-test (POST), so you can watch for it during the server's POST. You should have the server's GUID available for this part of the procedure.

```
add server-profile pokemon 2c02123a5303
```

   For more about assigning a Server Profile to a server that cannot boot yet, see "Server Profile Delivery to the Server" on page 3.

6. **Set up the Server Profile to do iSCSI Boot.**

   The `set server-profile` *name* `iscsi-boot` command provides the server with all the information that it needs to connect to storage upon boot up (such as, the VNIC to use for storage, IP address of the storage, and the target IQN).

```
set server-profile name iscsi-boot vnic-to-storage target-ip target-iqn
```

   For example:

```
set server-profile pokemon iscsi-boot eth5 192.168.8.108 -target-iqn=iqn.1992-
08.com.netapp:sn.118047284
```

   After the iSCSI boot Server Profile is created, you need the initiator IQN to complete the boot up sequence.

7. **Display the details for the iSCSI Boot Server Profile, and get the `i-iqn` from the resulting output.**

   Make a note of the `i-iqn`. You use it to allow the iSCSI Boot sequence to complete.

```
show server-profile pokemon iscsi-boot -detail
------------------------------------------------------------------------
server    pokemon
vnic      eth5
target    192.168.8.108
i-iqn     iqn.2006-09.com.xsigo:xg.139701f025
t-iqn     iqn.1992-08.com.netapp:sn.118047284
t-pg
lun       0
port      3260
proto     6
mnt-type  direct
lvm-grp
lvm-vol
dev
------------------------------------------------------------------------
1 record displayed
```

8. **As an option, remove all physical hard drives from server.**

   This step is not mandatory.

9. **After installation is complete but before rebooting the server, you copy the kernel-IB and Xsigo host drivers to `/mnt/sysimage/temp`.**

   However, at this point, you will still be in `chroot` jail, and need to escape.

10. **Escape chroot jail:**

```
chroot /mnt/sysimage
```

11. **Change directory to `/tmp`:**

```
cd /tmp
```

12. **Install the kernel IB RPM:**

```
rpm -ivh kernel-ib.rpm
```

13. **Then install the Xsigo host driver RPM:**

```
rpm -ivh xsigo-host-drivers.rpm
```

14. **Reboot the server.**

    During this reboot, the boot sequence is interrupted and you are prompted for the `i-iqn`. Type it when prompted, then allow the server to boot up.

CHAPTER 5

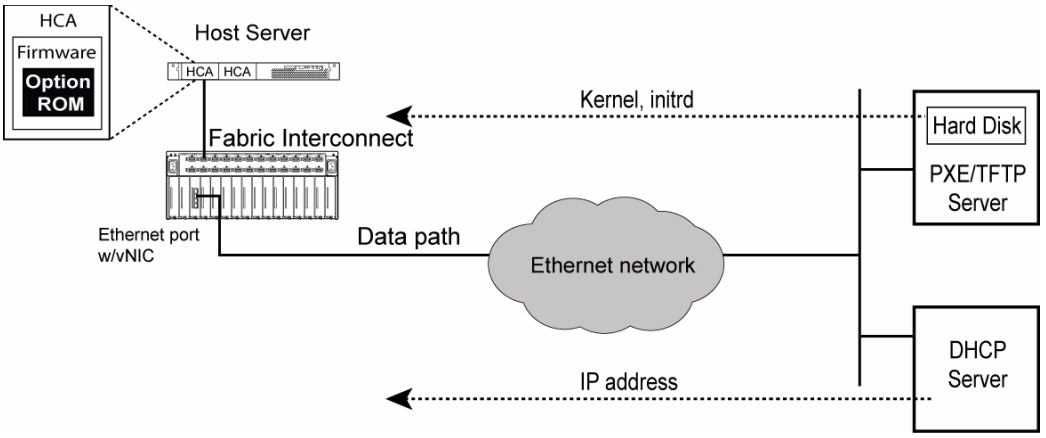
PXE Boot

Preboot Execution Environment (PXE boot) enables a host server to boot up over the network. Using the PXE boot protocol, the host server's HCA option ROM obtains the kernel boot image and initial RAM disk (`initrd`) from the network (not a local disk).

This chapter explains how to configure PXE boot for a VNIC. It contains the following sections:

PXE Boot Topology

During the BIOS boot sequence, the host server's PXE agent (option ROM) scans the network for a PXE image and reads its boot up instructions from a boot file stored on a boot server.



PXE Installation

These topics discuss a PXE installation:

PXE Installation Overview

Xsigo's PXE solution does not modify the Linux installer (Red Hat's Anaconda installer). Instead, Xsigo uses the versions supplied directly by the OS provider and adds in Xsigo's driver disks. A driver disk is an image that provides the installer with the required drivers. In Xsigo's case, the VNIC and VHBA drivers are needed to use them for installation.

To use a driver disk, you have to make it available to the installer during runtime. The typical way to do this is using a CD or floppy. To do this, an image called `xsigo-rhdd-`*`kversion-version-arch`*`.img` is provided. This file is an Anaconda driver disk.

To use it, add the kernel argument `dd` to the kernel on boot up. Or at the boot: prompt, type `linux dd`. Anaconda will prompt you for a driver disk when it runs. The easiest way is to burn the ISO image onto a CD and put it in the CD drive. Anaconda will read the CD and install the Xsigo drivers.

When the drivers are loaded (which might take up to 30 seconds), Anaconda will then detect new VNICs along with any physical Ethernet interfaces present on the server. The VNICs appear with the same name as was given on the Xsigo chassis (vnic0, and so on).

Note – Due to a limitation in the Anaconda installer, the VNICs might appear with the description `Unknown device: 199d:8202` instead of a more legible string.

You modify the stock Anaconda `initrd` by inserting our driver disk into it. This is done using the `xg-insert-dd` script. If you do this, the `Unknown device` error is not present and the VNICs will appear as Xsigo VNICs.

For information about installing the SAN volume, see "SAN Installation Using Linux Installers" on page 17

The Driver Disk and the Anaconda `initrd`

If you want to avoid using a CD to load the driver disk, you can insert the disk into the Anaconda `initrd` using the script `xg-insert-dd`. To do this, you must have a built driver disk and an Anaconda `initrd` (obtained from your Red Hat installation disks). The `xg-insert-dd` script creates a new `initrd` prefixed with the string `xsigo-`. If you use this in place of the default Anaconda `initrd`, the Xsigo drivers are loaded at boot time without the need to specify `dd` on the kernel command line or having to insert a CD.

Special Considerations for Red Hat 4 Linux

Red Hat 4 is a mature operating system now. It uses an older kernel (2.6.9) and an older Anaconda installer (version 10). This causes us some problems resulting in different behavior when using Anaconda. Here are some special rules that need to be followed in order to use RHEL4:

- The script `xg-insert-dd-ext2` must be used to insert the driver disk into the Anaconda `initrd`.
- You need to add the kernel argument: `dd=path:/xsigo-dd.img`.
- You will see 30 additional Ethernet interfaces numbered sequentially from the last physical port (eth2, eth3...eth31).
- You will have to call your bootable VNIC `eth`*x* where *x* is a number from 2 to 31.

The reasons for these are that the kernel is too old to support proper addition of PCI devices with given names. It automatically creates a full PCI bus and you cannot change the name given to the device.

The Anaconda supplied with rhel4 has a bug that causes it to crash if you supply a `dd.img` in the `initrd` (this should cause the driver disk to be automatically loaded). That is why you have to supply the `dd=path:/xsigo-dd.img` as a kernel argument.

Xsigo HCA Firmware Configuration

To support PXE boot, you must flash the Xsigo HCA firmware with the new PXE boot programming options. Check the XgOS Software Upgrade Guide or Release Notes for the Oracle XgOS release on your Fabric Interconnect to determine the compatible firmware levels. To upgrade firmware and enable the option ROM, see Chapter 8.

Configuring Linux Servers for PXE Boot

Configure a Linux server for PXE Boot by reviewing or performing these topics:

PXE Boot Configuration Overview

The following PXE boot services are supported:

Booting to disk—The kernel and `initrd` come from the network, then the host server mounts modules that enable its local disk to be visible on the network.

A network install—The host server runs an OS installer program over the network and chooses any of the available options.

Configure a VNIC on the Fabric Interconnect to support IP connectivity to the boot server. One or more servers must be reachable for the DHCP and PXE/TFTP services. These services are hosted and running on either the same, or different physical servers or virtual machines.

DHCP and TFTP servers are any freeware or commercially available product. Most Linux server distributions include a freeware DHCP server. For example, a distribution can include Internet Systems Consortium DHCP Server which is freeware available through the ISC website at https://www.isc.org/software/dhcp.

For configuring PXE boot, you configure an instance of a DHCP server and an instance of a TFTP server. In the TFTP server, you will edit the default file (pxelinux.cfg/default) to point to the location where the Linux distribution and a kickstart script. The kickstart script is used for configuring installing options. When the kickstart script runs, the boot process brings up the servers that are connected to the Fabric Interconnect through one or more VNICs.

▼ Determine MAC Addresses for DHCP Requests

This section documents how to retrieve MAC address information from the Fabric Interconnect. You will use this information when you configure the DHCP server.

1. **Determine a range of MAC addresses that are allowed to receive DHCP packets for PXE boot,**

```
admin@iowa[xsigo] show hardware
# Xsigo System Hardware Status
# Model: VP780-CH-SDR
# Serial: 050610240
# Base MAC: 00:13:97:01:80:00
# Base WWN: 50:01:39:70:00:00:80:00
# Locator LED: on
#
# Date: Sun May 11 01:32:18 GMT 2008
# User: admin
...
```

In this example, the `Base MAC` address in the pool of MAC addresses is allowed to receive DHCP requests.

Note – The Fabric Interconnect's MAC address consists of hard-coded bits and variable bits. The MAC Address Mask determines the variable bits of the address. For example, the MAC Address and MAC Address Mask fields in the displayed dialog box are 00:13:97:01:80:00 and 12, respectively. The MAC address mask of 12 indicates that the last 12 bits of the address are variable. As a result, the address are interpreted as
00:13:97:01:8x:xx where x is a configurable part of the MAC address.

---

2. **Configure and start DHCP services.**

   See "Configure DHCP Services" on page 46.

▼ Configure DHCP Services

To configure PXE boot through DHCP, first configure the DHCP server. Configuring the DHCP server is accomplished by configuring DHCP lease variables and network and subnet addresses in the `/etc/dhcpd.conf` file. The minimum parameters to provide DHCP services are:

- default DHCP lease time
- maximum DHCP lease time
- DDNS update style
- subnet address, netmask, and address and netmask ranges that are used by booting servers
- PXE boot Linux kernel
- IP interface address of the VNIC (or physical Ethernet interface) of the TFTP boot server.
- path to the Linux distributions

1. **Edit `/etc/dhcpd.conf` and add the following content:**

   a. Set the DHCP lease and renew parameters.

```
default-lease-time 600;
max-lease-time 7200;
```

   b. Set the addresses assigned to the booting servers.

```
subnet 1.22.0.0 netmask 255.255.0.0{
range 1.22.0.1 1.255.255.254
```

   c. Identify the boot loader used for PXE booting. The boot loader is part of the `syslinux` RPM.

```
filename 'pxelinux.0';
```

   d. Set the VNIC or physical NIC interface address that will host the TFTP boot server. This address is usually the same as the DHCP server host.

```
next-server 1.255.255.249;
```

e. Identify the locations of the Linux ramdisk.

```
option root-path "/tftpboot/";
}
```

2. If the DHCP service is not active, start it:

```
service dhcpd start
```

This step activates the DHCP daemon.

Note – DHCP servers post all error and warning messages to a log file. When you enable DHCP, if errors occur, you can check the log file at /var/log/messages

---

3. If the DHCP services are already on, you can keep them persistent through boot cycles:

```
chkconfig dhcpd on
```

4. Configure the TFTP services.

See "Configure TFTP Services" on page 47.

▼ Configure TFTP Services

Configuring a TFTP server consists of creating a directory for the Linux distributions, editing the `tftp` file and reloading the `xinetd` file.

Note – The following installation and configuration files are required for PXE boot up in a Red Hat Linux environment. Unless otherwise indicated, these files are supplied by you, the customer:
- `pxelinux.0` which is the network bootable file.
- `vmlinuz-2.6.9-42.EL` which is the bootable compressed Linux kernel.
- `initrd-rehl4u4_xsigo-i386.img`, which is the Xsigo ramdisk image file. This file is provided by Xsigo.
- `pxelinux.cfg`, which is the directory that contains a configuration file named `default`.

---

1. Verify that a TFTP RPM is installed on the PXE server:

```
rpm -qa | grep tftp
```

- If a TFTP server is installed, go to Step 3.
- If a TFTP server is not installed, you can download and install it. Go to Step 2.

2. **After downloading the TFTP server RPM, install it.**

```
rpm ivh tftp-server-*.rpm
```

   After the TFTP server is installed, proceed with configuring the TFTP server for PXE booting. Go to Step 3.

3. **Create a directory for the Linux distributions:**

```
mkdir /tftpboot
```

4. **Set ownership of the directory to `nobody`:**

```
chown nobody:nobody /tftpboot
```

   Also, ensure that `/tftpboot` has read-write permissions set for any accounts that are using the directory.

5. **Edit the `/etc/xinetd.d/tftp` file to change the following line to `no`.**

```
disable = no
```

   This step is mandatory.

6. **(Optional) Add the following new lines at the prompt of the `/etc/xinetd.d/tftp` file:**

```
log_type   =FILE /var/log/ftplog
log-on-success+=HOST EXIT DURATION
log-on-failure+=HOST USERID ATTEMPT
```

7. **Load the `xinetd` file onto the server:**

```
service xinetd reload
```

8. **Configure a VNIC to support PXE Booting.**

   See "Configure a PXE Boot VNIC" on page 49.

▼ Configure a PXE Boot VNIC

The next steps in configuring a VNIC to support PXE boot, is to create a Server Profile and populate it with a VNIC.

1. **Locate a physical server on which you can configure a Server Profile:**

```
show fabric-port
-----------------------------------------------------------------
name         alexander
type         hcaPort
descr
port         iowa:ServerPort23
id           2c90200204cbe
state        NA/up
m-key        0
lid          14
sm-lid       1
link-width   4x
link-speed   2_5_Gbps
-----------------------------------------------------------------
```

   In this example, the red text highlights the host name of the physical server. Make a note of this string. You will use it in the next step. If host names are not assigned to the servers, you can use the server's GUID which is indicated in the `id` field in the output of the `show fabric-ports` command.

2. **Add a Server Profile and specify the host name or GUID of the physical connection.**

   For example, to add a Server Profile called `pxeboot` to the physical server `alexander`, you would type the following command:

```
add server-profile pxeboot alexander@ServerPort23
```

3. **Add a VNIC, specify its port and interface number.**

   For example, to add a VNIC called `vn1.pxeboot` to slot 6, port 2:

```
add vnic vn1.pxeboot 6/2
```

4. **On the VNIC, set the following parameters:**
   - set the address type to `dhcp`
   - set `-boot-capable=true`

- configure an IP address and netmask (only if addresses are not assigned by DHCP)

For example, to configure `vn1.pubstest2` for PXE boot through DHCP, and configure IP address, type:

```
set vnic vn1.pubstest2 -addr-type=dhcp -boot-capable=true
```

5. **Reboot the host, and while the host is booting, enter the BIOS setup.**
6. **In the server BIOS, select the HCA as a boot device.**

   Set this HCA either as the first boot device, or to a relatively high position in the server's boot devices list, to speed up the boot process. If other devices are higher in the boot order, they are attempted, but will eventually fail and pass to the HCA.
7. **When the host boots from the `pxelinux.cfg` directory, choose the boot option from those offered.**

   Two common boot options are `-linux` and `-local`, but others might be present:
   - Select `-linux` for performing a PXE boot again.
   - Select `-local` for booting from hard drive.

Configuring an ESXi 4.1 Server for PXE Boot

Configure an VMware ESXi 4.1 server for PXE Boot by peforming these tasks:

- "Create a PXE Boot Server Profile" on page 51
- "Edit the PXE Config File" on page 52
- "Make the New PXE Config Available to Booting Servers" on page 52
- "Load the Image Into the ESXi 4.1 Server" on page 53

Note – Configuring PXE boot for an ESX 4.1 Classic server is a different process. See "Configuring an ESX 4.1 Classic Server for PXE Boot" on page 54.

▼ Create a PXE Boot Server Profile

If you have not already created a PXE Boot Server Profile, you must do so. The PXE Boot Server profile must have only one VNIC that is connected to the PXE server where the ESXi 4.1 boot image will reside.

1. **Create the Server profile.**

   For example, to create the Server Profile `esx41i` for the PXE boot server `gorgon` which is connected through IB port 23 on the Oracle Fabric Interconnect `tuffy`:

```
add server profile esx41i gorgon@tuffy:ServerPort23
```

2. **Create the VNIC for the PXE Boot Server Profile.**

   For example, to create a VNIC named `vnic1` in Server Profile `esx41i` and have the VNIC terminated on Gigabit Ethernet port 1 in slot 2:

```
add vnic vnic1.esx41i 2/1
```

3. **Create a VHBA for the PXE Boot Server Profile.**

   For example, to create a VHBA named `vhba1` in Server Profile `esx41i` and have the VHBA terminated on Fibre Channel port 1 in slot 8:

```
add vhba vhba1.esx41i 8/1
```

4. **Set the VNIC for PXE Booting:**

```
set vhba vhba.esx41i -boot-capable=true
```

5. **If the server has an OS, verify that the Server Profile is configured correctly and in the `up/up` state.**

```
show server-profile esx41i
name     state   descr   connection                    def-gw   vnics   vhbas
-------------------------------------------------------------------------------
esx41i    up/up           gorgon@tuffy:ServerPort23              1       1
1 record displayed
```

   If the server has no OS installed, the Server Profile state is `up/unassigned`, and not `up/up`.

6. **Edit the PXE config file on the PXE boot server.**

   See "Edit the PXE Config File" on page 52

▼ Edit the PXE Config File

On your network's PXE server, you move the modified ISO onto the PXE server, and edit the PXE server's config file to add the ESXi 4.1 bulletins provided with the Xsigo host drivers.

1. **Move the stock ISO image to the PXE server.**

```
cp /tmp/VMware-VMvisor-Installer-4.1.0-171294.x86_64.iso VMware-VMvisor-
Installer-4.1.0-171294.x86_64.iso
sudo mount -o loop VMware-VMvisor-Installer-4.1.0-171294.x86_64.iso iso
```

2. **Using vi or some other common Linux editor, add the following lines to the PXE server's PXE configuration file:**

```
label esx41i-install

kernel esx41i/mboot.c32
append esx41i/vmkboot.gz --- esx41i/vmkernel.gz --- esx41i/sys.vgz ---
esx41i/cim.vgz --- esx41i/ienviron.tgz --- esx41i/image.tgz ---
esx41i/install.tgz --- esx41i/xsigo.tgz
```

Note – The `xsigo.tgz` file is the Xsigo depot file renamed.

3. **Save the changes and quit the editor.**
4. **Make the PXE configuration available to the servers.**

See "Make the New PXE Config Available to Booting Servers" on page 52.

▼ Make the New PXE Config Available to Booting Servers

After all modifications are made and the new `initrd` is on the PXE Server, you flag the newly created `initrd` so that it is used to boot any server that has received the PXE boot image. For example, the steps in this section set the options in the kickstart script so that a booting server can receive required install-time settings.

1. **Edit the kickstart script to include labels that append the `initrd`.**

   For example

```
label esx41i-pxe3 #3.5.0 drivers
        kernel vmlinuz-esx-260247
        append initrd=XG-3.5.0-initrd-260247.img mem=768M askmedia
```

2. **Close the default file, making sure to preserve the changes.**

3. **Load the image into the ESXi 4.1 server.**

   See "Load the Image Into the ESXi 4.1 Server" on page 53.

▼ Load the Image Into the ESXi 4.1 Server

After the PXE Boot VNIC is created in the PXE Boot Server Profile, and the ISO image is modified, and the PXE server's config file points to the correct files to use for PXE booting, you can now load the ISO image into the ESXi 4.1 server, and boot the server.

Note – The following procedure assumes the installation of the PXE boot image through a lights out management solution.

---

1. **From the ESXi 4.1 server, enter the PXE menu and boot from it.**

   The ESXi 4.1 begins loading all pertinent OS files.

   After the OS files have been loaded, the server boots to the ESXi 4.1 native operating system. .

   As part of the boot sequence, you must read the license agreement.

2. **Accept (or Decline) the license agreement.**

   Follow the installer until you are prompted to specify a boot device on the Select a Disk dialog box.

3. **On the Select a Disk dialog box, select the hard disk on the PXE server.**

4. **Press Enter to continue the installer until you see the Confirm Install dialog box.**

   The PXE servers hard disk must be dedicated to the modified ISO. If your PXE server's hard disk already contains data, a dialog box is displayed warning that data will be overwritten.

   You can either overwrite the existing data, or abort the current installation, store the data on a different device, and then resume the installation.

5. **When the correct boot disk is confirmed, the installation runs to completion.**

   The Installation Complete dialog box is displayed.

6. **Emove any installation medium (CD or DVD) in the ESXi 4.1 server and allow the server to reboot.**

   When the server reboots, it progresses through the boot devices until it locates the PXE server's boot device from which it retrieves the boot image.

Configuring an ESX 4.1 Classic Server for PXE Boot

Configure an VMware ESX 4.1 Classic server for PXE Boot by reviewing and performing these topics:

Note – Configuring PXE boot for an ESXi 4.1 Server is a different process. For information about configuring PXE boot for an ESXi 4.1 installation, see "Configuring an ESXi 4.1 Server for PXE Boot" on page 50.

Configuration Prerequisites

To perform this procedure, you will need:

- Root access on the ESX 4.1 server
- The ESX 4.1 Server's OS CD
- The Xsigo ESX 4.1 Classic host driver software

The `make PXE` Tool

To include the Xsigo host drivers in the PXE boot disk, you use a shell script called `xsigo-mkpxe-initrd.sh` (also called `make PXE`) to inject the Xsigo host drivers into the ESX 4.1 `initrd`.

The `make PXE` tool has the following usage help, which is invoked by typing the tool without any of the mandatory qualifiers.

```
xsigo-mkpxe-initrd.sh --initrd VMware-initrd --xgfile Xsigo driver tgz file
[--output filename] [-d]
--initrd VMware initrd    required VMware initrd from VMware ESX Classic CD

--xgfile XG File          required file containing compatible Xsigo drivers

--output                       optional filename for output. Default is XG-INITRD
FILENAME

-d                            debug. Produces a logfile of xsigo-mkpxe-initrd.log
```

By default, when the Xsigo host drivers are successfully injected into the ESX `initrd`, a new `initrd` is created called `XG-initrd.img`.

Typically, you need to use only the `--initrd` and `--xgfile` qualifiers of the `make PXE` tool.

- You can use `--output` to rename the modified `initrd` if needed. This option supports changing the default name of the modified `initrd` from `XG-initrd.img` to something else.
- You can use the `-d` option if you encounter errors while attempting to inject the Xsigo host drivers into the ESX `initrd`, or if you are explicitly requested to do so by Oracle Support.

▼ Extract the `make PXE` Tool

The `make PXE` tool is included as part of the standard Xsigo host drivers bundle for ESX 4.1 Classic.

1. **Log in as `root` user on the ESX 4.1 server.**
2. **Copy the ESX `initrd` off of the ESX 4.1 OS CD.**

   You can put the ESX `initrd` in any directory. For this procedure, assume we will use opt/
3. **Put the Xsigo host drivers on the ESX 4.1 server.**

4. **Locate the** `xsigo-4.1.0.164009.2.2.0.esx41i.tgz` **file and decompress it to a directory.**

   For illustrative purposes, the `opt/` directory is shown, but you can use whatever directory you want as long as the ESX `initrd` and the `xsigo-4.1.0.164009.2.2.0.esx41i.tgz` file reside in the same directory:

```
tar -zxvf xsigo-4.1.0.164009.2.3.0.esx4.tgz opt/
```

5. **When the bundle is extracted, locate the** `xsigo-mkpxe-initrd.sh` **file.**

   This is the tool you will use to inject the Xsigo host drivers into the ESX `initrd`.

6. **Create the modified** `initrd`**.**

   See "Create the Modified initrd" on page 56.

▼ Create the Modified `initrd`

To create the modified `initrd`, you modify the `initrd` on a Linux server. You will need the following:

- to be logged in to the Linux server with root privileges
- execute privileges on whatever directory you will use for injecting the Xsigo host driver

To create the modified `initrd`, you use the `make PXE` tool (`xsigo-mkpxe-initrd.sh`) to inject the Xsigo host drivers into the ESX `initrd`.

1. **Move the** `xsigo-mkpxe-initrd.sh` **into the same directory as the Xsigo host drivers.**

2. **Inject the Xsigo host driver into the ESX** `initrd` **with the** `make PXE` **tool:**

```
sh xsigo-mkpxe-initrd.sh --initrd initrd.img --xgfile xsigo-4.1.0.164009.2.3.0-
7.esx4.tgz

Outputting XG-initrd.img
```

3. **When the modified `initrd` is created, list the contents of the directory (ls).**

   The modified `initrd` (`XG-initrd.img`) should be present.

```
ls
initrd.img xsigo-mkpxe-initrd.sh XG-initrd.img xsigo-4.1.0.164009.2.3.0-
7.esx4.tgz
```

   If the `XG-initrd.img` is not listed, you can use the `make PXE` tool in debug mode (with the `-d` qualifier) and check the log file (`xsigo-mkpxe-initrd.log`) for pertinent error messages. If the log file does not provide useful information, you can contact Oracle Support.

4. **Create a PXE boot Server Profile.**

   See "Create a PXE Boot Server Profile" on page 57.

▼ Create a PXE Boot Server Profile

If you have not already created a PXE Boot Server Profile, you must do so. The PXE Boot Server profile must have only one bootable VNIC that is connected to the PXE server where the ESX4 boot image will reside.

1. **Create the Server Profile for the PXE Server.**

   For example, to create the Server Profile `esx4` for the PXE boot server `gorgon` which is connected through IB port 23 on the Fabric Interconnect `tuffy`:

```
add server profile esx4 gorgon@tuffy:ServerPort23
```

2. **Create the VNIC for the PXE Boot Server Profile.**

   For example, to create a VNIC named `vnic1` in Server Profile `esx4` and have the VNIC terminated on port 1 in slot 8:

```
add vnic vnic1.esx4 8/1
```

3. **Set the VNIC for PXE Booting:**

```
set vnic vnic1.esx4 -boot-capable=true
```

4. **If the server has an OS loaded, verify that the Server Profile is configured correctly and in the up/up state.**

```
show server-profile esx4
name     state   descr   connection                    def-gw   vnics   vhbas
-------------------------------------------------------------------------------
esx4     up/up           gorgon@tuffy:ServerPort23              1
1 record displayed
```

If the server does not have an OS loaded, the Server Profile is `up/unassigned`, and not `up/up`.

5. **Edit the PXE config file on the PXE boot server.**

See "Edit the PXE Config File" on page 58.

▼ Edit the PXE Config File

On your network's PXE server, you move the modified `initrd` onto the PXE server, and edit the PXE server's config file to add the ESX 4.1 Classic bulletins provided with the Xsigo host drivers.

1. **Move the modified `initrd` image to the PXE server.**

You can use SCP or any other common file-transfer protocol to get the modified `initrd` onto the PXE server.

2. **Using vi (or some other common Linux editor), add the following lines to the PXE server's PXE config file.**

The following example shows using a kickstart named `vmware/4.1.0-164009` in the method string. If you are using a kickstart script, it will most likely have a different name.

```
label esx41-pxe3 #3.5.0 drivers
        kernel vmlinuz-esx-260247
        append initrd=XG-3.5.0-initrd-260247.img mem=768M askmedia
```

3. **Save the changes and quit the editor.**

4. **Make the PXE configuration available to the servers.**

See "Make the New PXE Config Available to Booting Servers" on page 59.

▼ Make the New PXE Config Available to Booting Servers

After all modifications are made and the new `initrd` is on the PXE Server, you flag the newly created `initrd` so that it is used to boot any server that has received the PXE boot image. For example, the steps in this section set the options in the kickstart script so that a booting server can receive required install-time settings. See "Unattended Installation With Kickstart" on page 59.

1. **Edit the kickstart script to include labels that append the `initrd`.**

   The following example shows using a kickstart named `vmware/4.1.0-164009` in the method string. If you are using a kickstart script, it will most likely have a different name.

```
label esx4-pxe

kernel vmlinuz-esx-164009

append initrd=XG-initrd.img method=http://PXE-server.domain.com/vmware/4.1.0-
164009 mem=1024M
```

Note – You must set the memory allocation to 1024 MB or higher. Do not use a lower memory value, and do not forget to include this parameter.

2. **Close the kickstart script, making sure to preserve the changes.**

Unattended Installation With Kickstart

When using kickstart to set up your SAN Boot environment, refer to the Red Hat Linux Installation Guide at `https://www.redhat.com/docs/manuals/enterprise/` and follow the instructions for preparing a kickstart installation. The information that follows provides the specifics of using kickstart to deploy the `xsigo-initrd`.

Before setting up the kickstart installation, prepare the `xsigo-initrd` and Xsigo host drivers for a standard manual install. If you are using multipathing software, customize your `xsigo-initrd` as described in "Modify the initrd for Multipathing With RHEL 5.x Hosts" on page 26.

When your `xsigo-initrd` is ready, write a script to copy the `initrd` to the SAN volume. For example, if your `xsigo-initrd` is available from an internal web server, your script might include the following:

```
cd /boot
wget http://sample_server.domain.com/linux/xsigo/xsigo-initrd-kernel-version.img
cd /tmp
sed -e 's/initrd-kernel/xsigo-initrd-kernel-version/' /boot/grub/grub.conf
/tmp/grub.conf
mv -vf /tmp/grub.conf /boot/grub/grub.conf
```

In this example, the script gets the Xsigo image from the web server and modifies the GRUB configuration file. When your script is complete, invoke it from the `%post` section of your kickstart script.

CHAPTER 6

ESX Host iSCSI and SAN Boot

This chapter discusses remote booting of the ESX host by both the iSCSI and SAN boot protocols. The procedures to configure iSCSI or SAN boot are very similar, and so are combined. Notations and supplemental text appear where there are differences between iSCSI and SAN. Both VNICs and VHBAs are identified as virtual devices:

- iSCSI boot protocol – utilizes VNIC virtual devices
- SAN boot protocol – utilizes VHBA virtual devices

This chapter contains the following sections:

Configuration Scenarios

Depending on the version of OS and the boot protocol, determine which configuration scenario is correct for you.

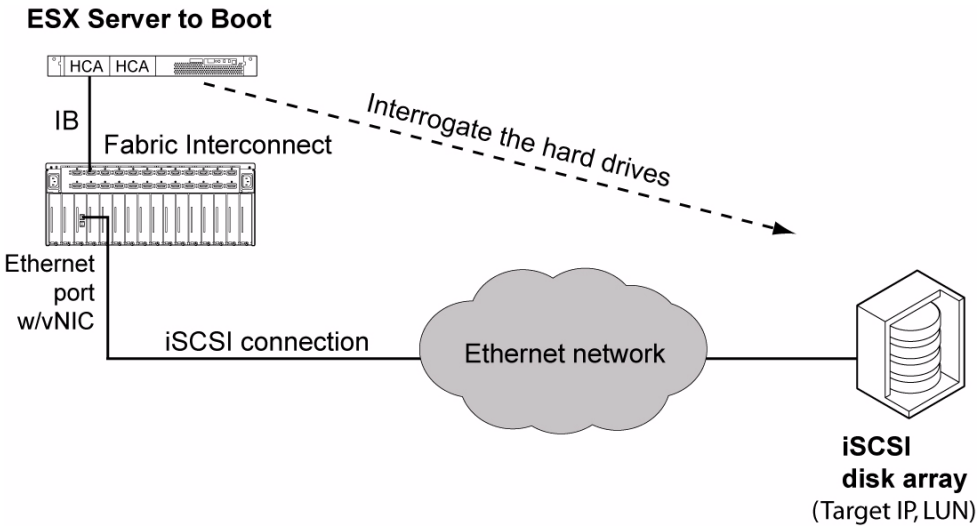
Operating System	Boot Protocol	Links
ESXi 5.1	SAN	• "SAN Boot Topology" on page 63 • "Autodeploying ESXi 5.1 and 5.5 Host Drivers" on page 80 • "Configuring the ESXi 5.x Boot Protocol" on page 64
ESXi 5.0	ISCSI or SAN	• "Visualizing Boot Protocol Topologies" on page 62 • "Configuring the ESXi 5.x Boot Protocol" on page 64
ESXi 4.1	ISCSI or SAN	• "Visualizing Boot Protocol Topologies" on page 62 • "Configuring the ESXi 4.1 Boot Protocol" on page 71
ESX 4.1 Classic	ISCSI or SAN	• "Visualizing Boot Protocol Topologies" on page 62 • "Configuring the ESX 4.1 Classic Boot Protocol" on page 76

Visualizing Boot Protocol Topologies

These topics describe the iSCSI and SAN boot protocol topologies.

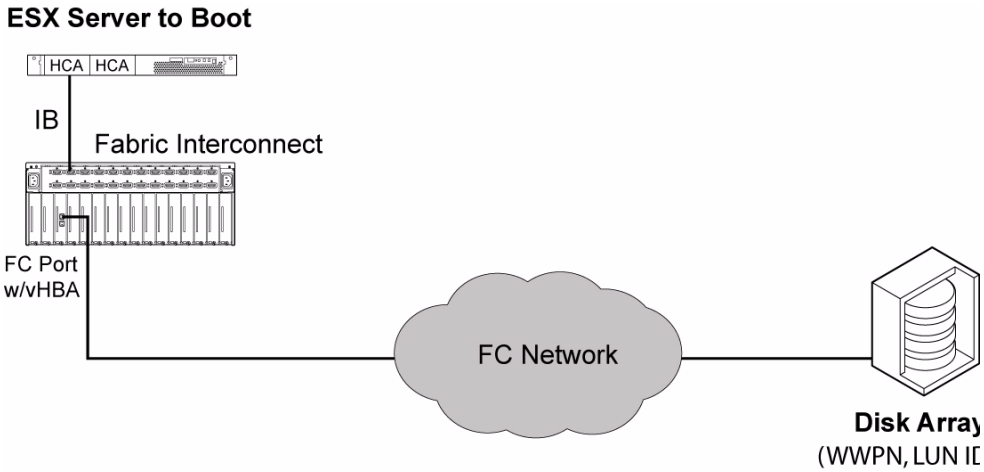
iSCSI Boot Topology

iSCSI Boot enables you to boot a VMware ESX server from a LUN on an iSCSI array accessed through a VNIC. The remote disk to boot from is identified by a target iSCSI Qualified Name (IQN) and Logical Unit Number (LUN) on a storage disk array device.



SAN Boot Topology

SAN Boot enables you to boot a VMware ESX server from a SAN volume accessed through a VHBA. The remote disk to boot from is identified by a target World Wide Port Name (WWPN) and Logical Unit Number (LUN) on a storage disk array device.



Configuring the ESXi 5.x Boot Protocol

Configure protocol boot for the VMware ESXi 5.x servers by completing these topics:

- "Remove the Previous Host Driver" on page 64
- "Creating a Modified ISO Image (ESXi 5.x)" on page 64
- "Create a Boot Protocol Server Profile (ESXi 5.x)" on page 67
- "Set the Server HCA High in the Boot Order" on page 69 (iSCSI only)
- "Load the Image Into the ESXi 5.x Server" on page 69

Note – You need additional utilities (for example, Microsoft PowerShell for iSCSI boot and `mkisofs` and `tar`, `gzip`, etc for SAN boot) for a typical default install.

▼ Remove the Previous Host Driver

Before installing the new ESX host drivers, ensure that you remove any previous version of host driver.

- **Remove the host drivers.**

  This example removes the ESX 4.1 host drivers:

```
esxupdate -b bundle-ID remove
```

Creating a Modified ISO Image (ESXi 5.x)

These topics help you create a modified ISO image.

- "Guidelines for Creating a Modified ISO Image (ESXi 5.x)" on page 64
- "Create a Modified ISO Image (ESXi 5.x)" on page 65

Guidelines for Creating a Modified ISO Image (ESXi 5.x)

To have the VNICs and VHBAs available to the ESXi 5.x OS for booting, you de-package an existing ESXi 5.x bundle, inject the ESXi host drivers into the ESX OS, then repackage the modified ISO.

Before creating a modified ISO image for ESXi 5.x hosts, be aware of the following:

- Creating the custom ISO is accomplished through Microsoft Windows PowerShell — specifically, the VMware vSphere PowerCLI plug-in for PowerShell. The Windows server needs this tool installed. Make sure the Windows server has the correct requirements to run PowerShell and PowerCLI.
- Creating the custom ISO is supported on a Windows host server only. The server requirements are determined by the PowerShell application.
- You use a pre-configured ESXi bundle as a baseline, then inject the necessary component into it:
  - For ESXi 5.0 Update 0 (GA), the ESXi bundle is `VMware-ESXi-5.0.0-469512-depot.zip` and is available from the VMware website.
  - For ESXi 5.0 Update 1, the ESXi bundle is the `VMware-ESXi-5.0.0-623860-depot.zip` file.
- You need full administrative rights on the Windows server where you create the custom ISO.

▼ Create a Modified ISO Image (ESXi 5.x)

In this procedure, the working directory is `\images\New` for the user `adminA`. The procedure uses the VMware 5.0 GA bundle 469512 for illustrative purposes. If your hosts are running ESXi 5.0 Update 1, use the bundle 623860.

1. **Install PowerShell on the Windows server if you have not done so already.**

   Refer to the PowerShell documentation for instructions.

2. **Install the PowerCLI plug-in if you have not done so already.**

   Refer to the PowerShell documentation for instructions.

3. **Download the `VMware-ESXi-5.0.0-469512-depot.zip` file to the Windows server.**

4. **Start PowerCLI.**

5. **In PowerCLI, import the ESXi 5.x bundle and the Xsigo host drivers into PowerCLI:**

```
Add-EsxSoftwareDepot -DepotUrl C:\Users\adminA\Desktop\images\New\VMware-ESXi-
5.0.0-469512-depot.zip
Add-EsxSoftwareDepot -DepotUrl C:\Users\adminA\Desktop\images\New\
xsigo_5.0.1ESX.1-1vmw.500.0.0.406165.zip
```

6. **Specify the file that you want to use when creating the output ISO.**

   The profile determines metadata about the output ISO, such as formatting, compression method, and so on. In this example, the profile is named `ESXi-5.0.0-469512-standard-xsigo`.

```
New-EsxImageProfile -CloneProfile ESXi-5.0.0-469512-standard -name "ESXi-5.0.0-
469512-standard-xsigo"
```

Note – The `-name` string supplied for the profile is enclosed in quotation marks. The quotation marks are required syntax for the profile's name.

7. **Add the IB stack and other dependencies to the depot.**

```
Add-EsxSoftwarePackage -ImageProfile ESXi-5.0.0-469512-standard-xsigo
-SoftwarePackage net-ib-core
Add-EsxSoftwarePackage -ImageProfile ESXi-5.0.0-469512-standard-xsigo
-SoftwarePackage net-mlx4-core
Add-EsxSoftwarePackage -ImageProfile ESXi-5.0.0-469512-standard-xsigo
-SoftwarePackage net-ib-mad
Add-EsxSoftwarePackage -ImageProfile ESXi-5.0.0-469512-standard-xsigo
-SoftwarePackage net-ib-sa
Add-EsxSoftwarePackage -ImageProfile ESXi-5.0.0-469512-standard-xsigo
-SoftwarePackage net-ib-ipoib
Add-EsxSoftwarePackage -ImageProfile ESXi-5.0.0-469512-standard-xsigo
-SoftwarePackage net-mlx4-ib
Add-EsxSoftwarePackage -ImageProfile ESXi-5.0.0-469512-standard-xsigo
-SoftwarePackage net-xscore
Add-EsxSoftwarePackage -ImageProfile ESXi-5.0.0-469512-standard-xsigo
-SoftwarePackage net-xsvnic
Add-EsxSoftwarePackage -ImageProfile ESXi-5.0.0-469512-standard-xsigo
-SoftwarePackage net-xve
Add-EsxSoftwarePackage -ImageProfile ESXi-5.0.0-469512-standard-xsigo
-SoftwarePackage scsi-xsvhba
```

8. **Create single output ISO containing all required files from the depot.**

Note – For completeness, the following example assumes unsigned drivers.

```
Export-EsxImageProfile -ImageProfile ESXi-5.0.0-469512-standard-xsigo -
ExportToIso
-FilePath C:\Users\adminA\Desktop\images\New\VMware-VMvisor-Installer-5.0.0-
469512_Xsigo.x86_64.iso -NoSignatureCheck
```

Note – Oracle makes every effort to release signed, certified host drivers. However, on some occasions, unsigned drivers might be released. If you receive unsigned host drivers, the `Export-EsxImageProfile` command has the `-NoSignatureCheck` option, which bypasses signature checking. Use the `-NoSignatureCheck` option for unsigned drivers only.

9. **Create a boot protocol Server Profile.**

   See "Create a Boot Protocol Server Profile (ESXi 5.x)" on page 67.

▼ Create a Boot Protocol Server Profile (ESXi 5.x)

If you have not already created a boot protocol Server Profile, you must do so. The Server Profile must have at least one virtual device that is connected to the storage where the ESXi 5.x boot image resides.

1. **Create the Server Profile.**

   This example creates the Server Profile `esx50` for `server10`, which is connected through IB port 23 on Oracle Fabric Interface `tuffy`:

```
add server profile esx50 server10@tuffy:ServerPort23
```

2. **Create the virtual device for the boot protocol Server Profile.**

   This example creates a VNIC named `vnic1` in Server Profile `esx50`, and has the VNIC terminated on the fibre channel port 1 in slot 8:

```
add vnic vnic1.esx50 8/1 -boot-capable=true
```

   This example creates a VHBA named `vhba1` in Server Profile `esx50`, and has the VHBA terminated on the fibre channel port 1 in slot 8:

```
add vhba vhba1.esx50 8/1 -boot-capable=true
```

3. **Set the Server Profile for booting and connected to the WWN of the target where the boot image resides.**

   For iSCSI boot and `vnic1`:

```
set server-profile esx50 iscsi-boot vnic1
-target-iqn=iqn.1992-08.com.netapp:sn.11804728455 -lun=203
```

   For SAN boot and `vhba1`:

```
set server-profile esx50 san-boot vhba1 22:00:00:50:CC:20:0E:6E
203
```

4. **Verify that the Server Profile is up and connected.**

   For iSCSI boot:

```
show server-profile esx50 iscsi-boot

server role vNIC mnt-typelvm-grplvm-vol dev mnt-opts disks
------------------------------------------------------------------------------
------------------
esx50 loadmount vnic1  22:00:00:50:CC:20:0E:6E (203/LM)
```

   For SAN boot:

```
show server-profile esx50 san-boot

server role vhba mnt-typelvm-grplvm-vol dev mnt-opts disks
------------------------------------------------------------------------------
------------------
esx50 loadmount vhba1  22:00:00:50:CC:20:0E:6E (203/LM)
```

Tip – Make a note of the size of the LUN on which the boot image resides. You are prompted to select the correct LUN when you load the host drivers onto the ESXi 5.x server, and knowing the LUN's size helps you identify it from the list of connected storage targets.

With the modified ISO ready and the boot protocol Server Profile created, you load the ISO onto the ESXi 5.x server and reboot the server so that the boot protocol virtual device is recognized as the primary boot option for the ESXi 5.x server.

5. **Consider your next step.**

   - If the boot protocol is iSCSI, set the server HCA high in the boot order.

     See "Set the Server HCA High in the Boot Order" on page 69.

- If the boot protocol is SAN, load the image into the server.
  See "Load the Image Into the ESXi 5.x Server" on page 69.

▼ Set the Server HCA High in the Boot Order

Note – This procedure is for the iSCSI boot protocol only. Do not perform this procedure for the SAN boot protocol.

In order to perform an iSCSI Boot, the server uses the Xsigo Option ROM on the HCA. You must set the HCA Option ROM higher in the boot order than the virtual CD to correctly boot the server.

1. **Enter the server's BIOS.**
2. **Select the Option ROM in the boot devices list.**
3. **Move the Option ROM higher than the virtual CD device in the boot order.**
4. **Save and exit the server's BIOS.**

   Setting the server to boot through the HCA Option ROM populates the iSCSI Boot Firmware Table (iBFT) with the disks to install the boot image on.
5. **Load the image into the server.**

   See "Load the Image Into the ESXi 5.x Server" on page 69.

▼ Load the Image Into the ESXi 5.x Server

After you have modified the ISO image, put the image on a network-reachable device, and created a boot protocol Server Profile, follow this procedure to load the ISO image into the ESXi 5.x server, and boot the server. When it boots, you have an option to select the LUN that contains the boot protocol ISO.

Note – This procedure installs the boot protocol image and configures the boot protocol feature through an ILO or DRAC.

---

1. **From the ESXi 5.x server, locate and mount the modified ISO (ESXi-5.0.0-Standard-Xsigo Installer).**

   When the ESXi-5.0.0-standard-xsigo Installer is mounted, files are loaded onto the server.

Note – Be aware that loading files onto the server can take a long time. You must wait for this process to complete.

After the OS files have been loaded, the server boots to the ESXi 5.x native operating system.

2. **Accept (or Decline) the license agreement and then press Enter to continue the installer.**

   Follow the installer until you are prompted to specify a boot disk on the Select a Disk to Install or Upgrade dialog box.

3. **On the Select a Disk to Install or Upgrade dialog box, select the LUN on which the boot protocol virtual device is connected.**

Tip – You can display details about a LUN by pressing F1.

4. **Press Enter to continue the installer until you see the Confirm Install dialog box.**

   If the LUN has already had ESXi 5.x installed, a VMware file system is labeled on the LUN. The installer prompts you to determine how you want to proceed.

5. **On the VMFS Format dialog box, select the option as needed for your installation:**
   - Upgrade ESXi, preserve VMFS datastore.
   - Install ESXi, preserve VMFS datastore.
   - Install ESXi, overwrite VMFS datastore. This is required for a fresh install.

Note – Additional dialog boxes for the keyboard type and language are displayed. Make sure to select the correct option for your ESX installation.

6. **When prompted, log in to the ESXi 5.x server.**

   After you log in, the installation runs to completion. The Confirm Installation dialog box is displayed.

7. **Press F11 to confirm that you want to continue the installation.**

   The OS and Xsigo host drivers are loaded onto the ESXi 5.x Server.

8. **Finalize the installation BY DOING WHAT?.**

   As part of finalizing the OS and host driver installation, you must reboot the server to load the new OS and host drivers into memory.

9. **Press Enter to reboot the server.**

   The server reboots and progresses through the boot devices until it locates the boot protocol virtual device from which it retrieves the boot protocol image.

Note – For the iSCSI boot protocol, during the first boot, the VHBA is attempted by default. When no OS is recognized on the VHBA, the server skips the VHBA and tries the VNIC. When the VNIC is used, the server logs in to the iSCSI array and gets the IQN information it needs. Because the VNIC has an OS available to boot from (on the iSCSI LUN connected to the VNIC), that OS is used and the server boots over the VNIC. This procedure occurs each time the server is iSCSI booted.

Also, when booting from a VNIC, the login message while the VNIC is logging in to the iSCSI array is displayed on screen so rapidly that you might not recognize that the server is booting from VNIC. Also, no IQN or other recognizable iSCSI information is displayed on the screen.

Note – For the SAN boot protocol, a NIC card is attempted first, then the VHBA. The text `VHBA installing` indicates the name of the VHBA configured for SAN Booting. If the name is incorrect, reperform this procedure.

After the boot protocol virtual device is recognized as a boot device, the ESXi 5.x server completes its boot up. VNICs and VHBAs can then be configured in the ESX server.

Configuring the ESXi 4.1 Boot Protocol

Configure protocol boot for the VMware ESXi 4.1 servers by completing these topics

- "Creating a Modified ISO Image (ESXi 4.1)" on page 72
- "Create a Boot Protocol Server Profile (ESXi 4.1)" on page 73
- "Load the Image Into the ESXi 4.1 Server" on page 75

Note – You need additional utilities (for example, `mkisofs` and `tar`, `gzip`, etc) for a typical default install.

Creating a Modified ISO Image (ESXi 4.1)

These topics help you create a modified ISO image.

Guidelines for Creating a Modified ISO Image (ESXi 4.1)

For this procedure, you need both of the following ISO files:

- `VMware-VMVisor-Installer-4.1.0-171294.x86_64.iso` . Oracle does not provide the VMware installation medium, you must obtain it from VMware.
- `xsigo-esx-driver-disk-4.1.0.260247.3.5.0-14.iso`. This bundle contains a shell script (`xsigo-add-drivers-esx41i.sh`) that inserts the host drivers for ESXi 4.1 into the ISO image. You receive this ISO file as part of the host drivers bundle download.

To have the VNICs and VHBAs available to the ESXi 4.1 OS for booting, you de-package an existing ESXi 4.1 bundle, inject the ESXi host drivers into the ESX OS, then repackage the modified ISO. The modified image is then used as the boot image.

To create the modified ISO image, you must:

- Be logged in to any Linux system with `root` privileges
- Have execute privileges on the ISO
- Have execute privileges on whatever directory used to uncompress and recompress the modified ISO.
- Use the `xsigo-add-drivers-esx41i.sh` shell script which has the following syntax:

```
xsigo-add-drivers-esx41i.sh --esxiso --driveriso --extrarpm --output -d
--isolinuxcfg --ksconfig --hca-installer-hooks
```

▼ Create a Modified ISO Image (ESXi 4.1)

1. **Get the VMware installation medium onto a Linux host and place it into a working directory.**

   For example, `opt/`.

2. **Locate the `xsigo-4.1.0.260247.esx41i.tgz` file and uncompress it to the same directory.**

```
tar -zxvf xsigo-4.1.0.260247.esx41i.tgz opt/
```

3. **Use the `xsigo-add-drivers` script and recompress the bundle.**

   This step takes a few minutes, and progress messages are displayed to indicate the individual stages in the overall job.

   When the command completes, the new ISO image is created as `XG-VMware-Visor-Installer`. You use this image to boot the ESXi 4.1 Server.

4. **Create a boot protocol Server Profile.**

   See "Create a Boot Protocol Server Profile (ESXi 4.1)" on page 73.

▼ Create a Boot Protocol Server Profile (ESXi 4.1)

You must create a boot protocol Server Profile, to perform protocol boot. The boot protocol Server Profile must have only one virtual device that is connected to the storage where the ESXi 4.1 boot image resides.

1. **Create the Server Profile.**

   This example creates the Server Profile `esx41i` for `server10`, which is connected through IB port 23 on Oracle Fabric Interface `tuffy`:

```
add server profile esx41i server10@tuffy:ServerPort23
```

2. **Create the virtual device for the boot protocol Server Profile.**

   This example creates a VNIC named `vnic1` in Server Profile `esx41i`, and has the VNIC terminated on the fibre channel port 1 in slot 8

```
add vnic vnic1.esx41i 8/1 -boot-capable=true
```

   This example creates a VHBA named `vhba1` in Server Profile `esx41i`, and has the VHBA terminated on the fibre channel port 1 in slot 8:

```
add vhba vhba1.esx41i 8/1 -boot-capable=true
```

3. **Set the Server Profile for booting and connected to the WWN of the target where the boot image resides:**

   For iSCSI boot and `vnic1`:

```
set server-profile esx41i iscsi-boot vnic1
-target-ign=iqn.1992-08.com.netapp:sn.118047284
-lun=203
```

   For SAN boot and `vhba1`:

```
set server-profile esx41i san-boot vhba1 22:00:00:50:CC:20:0E:6E
203
```

4. **Verify that the Server Profile is up and connected as shown.**

   For iSCSI boot:

```
show server-profile esx41i iscsi-boot

server role vNIC mnt-type lvm-grp lvm-vol dev mnt-opts disks
-------------------------------------------------------------------------------
------------------
esx41i loadmount vnic1 static iqn.1992-08.com.netapp:sn.118047284 (203/LM)
```

   For SAN boot:

```
how server-profile esx41i san-boot

server role vhba mnt-type lvm-grp lvm-vol dev mnt-opts disks
-------------------------------------------------------------------------------
------------------
esx41i loadmount vhba1 static 22:00:00:50:CC:20:0E:6E (203/LM)
```

Note – Make a note of the size of the LUN on which the boot image resides. You are prompted to select the correct LUN when you load the host drivers onto the ESXi 4.1 server, and knowing the LUN's size helps you to select it from the list of connected storage targets.

---

5. **Load the image into the server.**

   See "Load the Image Into the ESXi 4.1 Server" on page 75.

▼ Load the Image Into the ESXi 4.1 Server

After you have modified the ISO image, put the image on a network-reachable device, and created a boot protocol Server Profile, follow this procedure to load the ISO image into the ESXi 4.1 server, and boot the server. When it boots, you have an option to select the LUN that contains the boot protocol ISO.

Note – This procedure installs the boot protocol image and configures the boot protocol feature through an ILO or DRAC.

---

1. **From the ESXi 4.1 server, locate and mount the modified ISO (**`XG-VMware-Visor-Installer`**).**

   When the ISO is mounted, ESXi 4.1 begins loading all pertinent OS files.

Note – Be aware that loading files onto the server can take a long time. You must wait for this process to complete.

---

2. **After the OS files have been loaded, the server boots to the ESXi 4.1 installer.**
3. **Accept (or Decline) the license agreement.**
4. **Press Enter to continue the installer.**

   Follow the installer until you are prompted to specify a boot disk on the Select a Disk dialog box.

5. **On the Select a Disk dialog box, select the LUN on which the boot protocol virtual device is connected.**

Note – Notice that the specific target and LUN IDs are not displayed in the list. Because you made a note of the boot protocol LUN's size, you should be able to determine which LUN to select as the boot disk.

---

6. **Press Enter to continue the installer until you see the Confirm Install dialog box.**

   The boot protocol LUN must be dedicated to the modified ISO. If your boot protocol LUN already contains data, a dialog box is displayed warning that data will be overwritten.

   You can either overwrite the existing data, or abort the current installation, store the data on a different LUN, and then resume the installation.

7. **When the correct boot disk is confirmed, the installation runs to completion.**

   The Installation Complete dialog box is displayed.

8. **Remove any installation medium (CD or DVD) in the ESXi 4.1 server and allow the server to reboot.**

   The server reboots and progresses through the boot devices until it locates the boot protocol virtual device from which it retrieves the boot protocol image.

Note – For the iSCSI boot protocol, during the first boot, the VHBA is attempted by default. When no OS is recognized on the VHBA, the server skips the VHBA and tries the VNIC. When the VNIC is used, the server logs in to the iSCSI array and gets the IQN information it needs. Because the VNIC has an OS available to boot from (on the iSCSI LUN connected to the VNIC), that OS is used and the server boots over the VNIC. This procedure occurs each time the server is iSCSI booted.

When booting from a VNIC, the login message while the VNIC is logging in to the iSCSI array is displayed on screen so rapidly that you might not recognize that the server is booting from a VNIC. Also, no IQN or other recognizable iSCSI information is displayed on the screen.

Note – For the SAN boot protocol, a NIC card is attempted first, then the VHBA. The text `VHBA installing` should indicate the name of the VHBA configured for SAN Booting. If the name is incorrect, reperform this procedure.

After the boot protocol virtual device is recognized as a boot device, the ESXi 4.1 server completes its boot up. VNICs and VHBAs can then be configured in the ESX server.

Configuring the ESX 4.1 Classic Boot Protocol

Configure protocol boot for the VMware ESX 4.1 Classic servers by completing these topics:

ESX 4.1 Classic Boot Protocol Configuration Considerations

Before configuring ESX 4.1 boot protocol, be aware of the following considerations:

- If a VMFS exists on the ESX 4.1 that is configured for boot protocol, that VMFS must be deleted from the ESX Server before configuring the ESX 4.1 boot protocol. The process of configuring boot protocol for ESX 4.1 creates a separate instance of VMFS, and sometimes does not remove any existing VMFS on the ESX server.

  For example, if you are upgrading from an earlier update of ESX 4.1 boot protocol, or if you are re-installing on an ESX 4.1 Server and configuring boot protocol, you must remove any existing VMFS before beginning the ESX 4.1 boot protocol configuration procedure.
- While up to four LUNs have been tested, for simplicity, make only one LUN available to the virtual device in the boot protocol Server Profile.
- Also, be aware that the ESX installer does not allow for reformatting any unknown partitions, so you might need to clear entries from the partition table.

▼ Install and Configure the ESX 4.1 Classic Boot Protocol

Note – This procedure assumes the installation of the boot protocol image through a lights out management solution.

---

1. **Make sure that you have a boot protocol Server Profile configured.**
2. **Make sure that the virtual device present in the boot protocol Server Profile can reach the LUN from which the server is booted.**
3. **Insert the latest ESX 4X install CD and wait for the CD to autorun and display the (top-level) Options menu.**
4. **In the Options menu, use the up and down arrows to select the Install ESX in graphical mode option.**

   This option enables you to set or modify boot-loader arguments.
5. **Press the F2 key to enter edit mode for the Boot Options.**
6. **Backspace over the** `mem=` **argument and overwrite the default kernel memory with the recommended minimum amount of memory.**

   The default is 1024 MB, but you can set the value higher if desired.

7. **Press Enter to resume the installer.**

   When the Install progress bar completes, the Welcome screen is displayed.

8. **Click Next and proceed through the ESX Installer.**

9. **Read and accept the license agreement, and respond to all prompts until the Custom Drivers dialog box is displayed.**

10. **On the Custom Drivers dialog box, in the Load Custom Drivers section, click Yes.**

    A pop-up window alerts you to select the drivers that you want to install.

11. **Click OK to close the pop-up window, then click Add... and browse to the location of the Xsigo host drivers**

    You need to install the driver ISO file to be able to connect to the Xsigo ESX host drivers ISO.

Note – The Xsigo ESX host drivers must be accessible to the ESX Installer for them to be successfully installed. For example, the Xsigo ESX host drivers are added to the Installer directly from the `xsigo.iso` file or from a network shared device.

---

12. **Double click each Xsigo host driver ISO to add them to the Drivers table in the ESX Installer.**

    The required modules are:

    - `ib-basic`
    - `vmware-esx-drivers-net-`*`xxxx`*
    - `vmware-esx-drivers-ulp-`*`xxxx`*

    An additional pop-up window that warns you about loading custom drivers.

13. **Click the I Accept check box, and click I Accept.**

    The Load Drivers pop-up window is displayed.

14. **Click Yes to continue the installation.**

    A progress bar is displayed while the ESX drivers and Xsigo ESX host drivers progress through-installation.

15. **Continue the installer until you see the Network Configuration dialog box.**

16. **Press Alt+F2 to enter the debug shell on the ESX server.**

    This step suspends the ESX Installer so that you can use the Xsigo `install-load` script.

17. **In the debug shell, press Enter to activate the ESX console.**

18. **At the prompt, run the Xsigo `install-load` script:**

```
/tmp/drivers/usr/sbin/esxcfg-xgutil install-load
```

The script loads necessary software modules and verifies that the boot disk is available.

19. **When the script completes, press Alt+F6 to exit the debug shell and return to the ESX Installer.**

20. **On the Specify ESX Datastore dialog box, specify the datastore for virtual machines on the ESX Server.**

Make sure to select:

- Create new datastore
- Create on the same device as ESX
- Alternatively, you can preserve an existing data store by clicking Use existing datastore and specifying the partition. This option requires that the datastore be from the same version of ESX. For example, an ESX Classic 4.1 datastore used for booting an ESX 4.1 Classic server.

21. **Continue the ESX Installer by configuring the standard information required by the ESX 4.1 Server.**

    a. **Specify the IP address method for the ESX Server.**

    b. **Select Standard Setup to boot off of a single LUN or hard drive.**

    c. **Select the storage target that contains the LUN.**

Note – If you are upgrading an existing ESX server (or re-installing) and the target was previously associated with the ESX server, additional pop-up windows that assist you with performing a clean install are displayed.

---

    d. **Select the time zone in which the ESX server is being configured for protocol boot.**

    e. **Specify the NTP server (if applicable) with which the ESX server will synchronize.**

    f. **Set the administrator `root` password.**

22. **When all ESX drivers, Xsigo host drivers, and boot protocol options are specified, verify the intended configuration by reviewing the Summary of Installation dialog box.**

23. **Reconnect to the ESX Installer (or re-insert the physical medium if you are installing from DVD).**

    If you see the pop-up window to reconnect to (or re-insert) the ESX installer DVD, do so.

24. **Complete the installation by reviewing and confirming the remaining dialog boxes.**

25. **Reboot the ESX Server.**

Autodeploying ESXi 5.1 and 5.5 Host Drivers

Through autodeploy, ESXi 5.1 and 5.5 hosts are set up to receive host drivers through a PXE boot server. Autodeployments are useful in large deployments of ESXi 5.1 and 5.5 hosts. Autodeployment is not available for the iSCSI boot protocol, or for ESXi 5.0, ESXi 4.1, or ESX 4.1 servers.

Configuring for autodeployment requires you to perform these tasks:

- "Install the Required Software" on page 80
- "Configure TFTP and vCenter Servers and DHCP" on page 81
- "Create the Boot Image for the ESXi 5.1 or 5.5 Hosts" on page 82
- "Create the Host Profile for Autodeploy" on page 84
- "Autodeploy Through the Host Profile" on page 85

▼ Install the Required Software

1. **Download vCenter 5.1.**
2. **Download the ESXi 5.1 or 5.5 offline bundle.**
3. **Install vCenter Server 5.1.**
4. **Install vSphere Client 5.1.**
5. **Install PowerShell 2.0 on vCenter Server.**
6. **Install PowerCLI 5.1 on vCenter Server.**
7. **Install autodeploy (which is located in the vCenter server ISO).**

8. **Install a TFTP Server on the vCenter Servers.**

   See "Configure TFTP and vCenter Servers and DHCP" on page 81.

▼ Configure TFTP and vCenter Servers and DHCP

After you install all of the required software, follow this procedure to set up the TFTP server, the vCenter server, and the DHCP address allocation.

Note – In this section you are configuring multiple servers so that IP addresses are allocated to host interfaces. Make sure that the DHCP server, TFTP server, and vCenter server are all on the same subnet to avoid connectivity problems.

---

1. **Start the TFTP server by clicking File -> Configure -> Start.**

   When this step is complete, a folder called `c:\TFTP-Root` is created. The folder name might be different depending on which vendor or version of TFTP is used.

Note – If you are running the TFTP Server on Windows, make sure that the firewall is open for TFTP communication.

---

2. **On the vSphere client Home screen, verify that the Auto-deploy icon is present.**
   - If the icon is not present, verify that autodeploy is actually installed on the vSphere server.
   - If autodeploy is not installed, download and install it now.
   - If autodeploy is installed but not running, check that the auto-deploy listener service is running, then re-install.
3. **Click the Auto-deploy icon to display the Auto-deploy screen.**
4. **On the Auto-deploy screen, find the BIOS DHCP location, and copy the file name indicated for later use.**

   An example of the file name is `undionly.kxpe.vmw.hardwired`.
5. **Click download TFTP Boot.zip and download the file to `c:\TFTP-Root` folder.**
6. **Unzip the file in `c:\TFTP-Root`.**
7. **Click the folder to select it, then copy the location of the BIOS DHCP file name.**
8. **Go to the DHCP Server and add the file name and IP address of the TFTP Server.**
9. **Choose Start -> Administrative Tools -> DHCP.**

10. **Expand the IPv4 -> Scope -> Address Pool and edit to update with the IP address range available to you.**

11. **Expand the IPv4 -> Scope Options, then right-click Configure -> Options and set the following options:**

    - number 066 Boot Server Host name with the IP address of the TFTP Server
    - number 067 Bootfile Name with the BIOS DHCP file name (for example, `undionly.kxpe.vmw.hardwired`).

    At the completion of this section, you should be able to boot your host and can verify that the interface is getting an IP address through the DHCP server.

12. **Create the boot image for the server.**

    See "Create the Boot Image for the ESXi 5.1 or 5.5 Hosts" on page 82.

▼ Create the Boot Image for the ESXi 5.1 or 5.5 Hosts

After the IP address is assigned to the interface in the auto-deployment server, autodeploy fails due to an error that no ESXi image is associated. You need to create a boot image with the host drivers and either the ESXi 5.1 ISO or ESXi 5.5 ISO present by using the PowerShell CLI.

1. **Open PowerCLI and connect to your vCenter Server.**

2. **Type the following commands:**

    - For ESXi 5.1:

```
Add-EsxSoftwareDepot -DepotUrl c:\VMware-ESX-5.1.0-799733-depot.zip
Add-EsxSoftwareDepot http://vcenter-server-ipaddr/vSphere-HA-depot
Add-EsxSoftwareDepot -DepotUrl c:\xsigo-hostdriver.zip
New-EsxImageProfile -CloneProfile "ESXi-5.1.0-799733-standard" -name
"ESXiStatelessImage-Xsigo"
Add-EsxSoftwarePackage -ImageProfile "ESXiStatelessImage-Xsigo" -
SoftwarePackage vmware-fdm
Add-EsxSoftwarePackage -ImageProfile "ESXiStatelessImage-Xsigo" -
SoftwarePackage net-ib-core
Add-EsxSoftwarePackage -ImageProfile "ESXiStatelessImage-Xsigo" -
SoftwarePackage net-ib-mad
Add-EsxSoftwarePackage -ImageProfile "ESXiStatelessImage-Xsigo" -
SoftwarePackage net-ib-sa
```

- For ESXi 5.5:

```
Add-EsxSoftwareDepot -DepotUrl c:\VMware-ESX-5.5.0-1331820-depot.zip
Add-EsxSoftwareDepot http://vcenter-server-ipaddr/vSphere-HA-depot
Add-EsxSoftwareDepot -DepotUrl c:\xsigo-hostdriver.zip
New-EsxImageProfile -CloneProfile "ESXi-5.5.0-1331820-standard" -name
"ESXiStatelessImage-Xsigo"
Add-EsxSoftwarePackage -ImageProfile "ESXiStatelessImage-Xsigo" -
SoftwarePackage net-ib-core
Add-EsxSoftwarePackage -ImageProfile "ESXiStatelessImage-Xsigo" -
SoftwarePackage net-ib-mad
Add-EsxSoftwarePackage -ImageProfile "ESXiStatelessImage-Xsigo" -
SoftwarePackage net-ib-sa
Add-EsxSoftwarePackage -ImageProfile "ESXiStatelessImage-Xsigo" -
SoftwarePackage net-ib-cm
```

3. **Type these command for either ESXi 5.1 or ESXi 5.5.**

```
Add-EsxSoftwarePackage -ImageProfile "ESXiStatelessImage-Xsigo" -
SoftwarePackage net-mlx4-core
Add-EsxSoftwarePackage -ImageProfile "ESXiStatelessImage-Xsigo" -
SoftwarePackage net-mlx4-ib
Add-EsxSoftwarePackage -ImageProfile "ESXiStatelessImage-Xsigo" -
SoftwarePackage net-ib-ipoib
Add-EsxSoftwarePackage -ImageProfile "ESXiStatelessImage-Xsigo" -
SoftwarePackage net-xscore
Add-EsxSoftwarePackage -ImageProfile "ESXiStatelessImage-Xsigo" -
SoftwarePackage net-xsvnic
Add-EsxSoftwarePackage -ImageProfile "ESXiStatelessImage-Xsigo" -
SoftwarePackage net-xve
Add-EsxSoftwarePackage -ImageProfile "ESXiStatelessImage-Xsigo" -
SoftwarePackage scsi-xsvhba
New-DeployRule -Name "FirstBoot" -Item "ESXiStatelessImage-Xsigo" -AllHosts
Add-deployRule -DeployRule "FirstBoot"
Esxport-ESXimageProfile -ImageProfile "ESXiStatelessImage-Xsigo"
-EsxportToBundle -FilePath c:\ESXiStatelessImage-Xsigo.zip
```

At the completion of this procedure, you have a bootable ISO with host drivers included.

You can try doing autodeploy as the rule is created.

```
  Export-ESXimageProfile -ImageProfile "ESXiStatelessImage-Xsigo"
-ExportToIso -FilePath c:\ESXiStatelessImage-Xsigo.iso
```

4. **Create the host profile.**

See "Create the Host Profile for Autodeploy" on page 84.

▼ Create the Host Profile for Autodeploy

A Host Profile is required to support autodeployment. After the boot image is created, perform the following procedure:

1. **Boot the auto-deployed host to verify that autodeployment occurs successfully.**
2. **Open the vSphere client and log in to the vCenter Server.**
3. **In the Server list, find the auto-deployed host.**
4. **Create a cluster and assign it a name—for example, DEVCLUSTER.**
5. **Add the auto-deployed host to the cluster.**
6. **Right-click on the auto-deployed host, and click Enter Maintenance Mode.**
7. **Right-click on the host, and click Host Profile -> Create Profile From Host.**
8. **Type a host name.**
9. **Expand the menu bar by clicking View -> Management -> Host Profiles.**
10. **Right click the new host profile you just created, then select Edit Profile.**
11. **Edit the host profile as needed—for example, specify network interface, login details, and so on).**
12. **Right-click the host, then select Host Profile -> Manage Profile.**
13. **Select the profile you just created.**
14. **Expand the menu bar view -> Management -> Host Profiles -> select the host profile created.**

    The auto-deployed host should be listed in the Hosts and Clusters option.
15. **In the Hosts and Clusters option, right-click on the host and select Update answer file.**
16. **Change any settings (if required).**
17. **In the Hosts and Clusters option, right-click on the host and select Check Profile Compliance.**
18. **In the Hosts and Clusters option, right click on the host and select Apply Profile.**
19. **Start PowerCLI and update the auto-deploy rule with the host profile created:**

```
New-DeployRule –name “ProductionBoot-Xsigo” –Item “ESXiStatelessImage-Xsigo”,
host profile created , cluster created in vCenter Server
-Pattern “vendor=xsigo”
```

20. **Type the following commands:**

```
Add-Deployrule -DeployRule "ProductionBoot-Xsigo"
Remove-DeployRule -DeployRule FirstBoot -delete
```

21. **Create the Server Profile for the SAN boot protocol.**

    See "Create a Boot Protocol Server Profile (ESXi 5.x)" on page 67.

22. **Autodeploy through the host profile.**

    See "Autodeploy Through the Host Profile" on page 85.

▼ Autodeploy Through the Host Profile

After creating the Server Profile, follow this procedure:

1. **Reboot the auto-deploying server.**

   It is deployed with new rule created ProductionBoot-Xsigo.

2. **Confirm that the network interface is similar to the one saved in the host profile.**

3. **You can perform further testing by creating a VNIC on the auto-deploying server and creating a vSwitch.**

Note – While creating the VNIC, you should follow the naming convention of: vmnic*x*, where *x* is a number.

---

4. **Create the Host Profile.**

   See "Create the Host Profile for Autodeploy" on page 84.

5. **Save the profile and apply it on the auto-deploying host.**

   When the auto-deploying host reboots, it should retain the configuration from host profile.

Injecting ESXi 5.1 Host Drivers Manually

These topics describe how to manually inject host drivers into an ESXi 5.1 bundle.

Procedure Overview

The procedure to PXE Boot or SAN Boot an ESXi 5.1 host is the same as for an ESXi 4.1 host, however you do not use the `remaster-iso` script. Instead, you must manually inject the Oracle host drivers into the native ESXi 5.1 OS to have Oracle VNICs and VHBAs available to the ESXi OS. The procedure,"Inject the Oracle Host Drivers into the ESXi 5.1 Bundle" on page 86, documents how to inject the host drivers for a freshly created ESXi 5.1 server. Afterwhich, you can use the procedures "Create a Boot Protocol Server Profile (ESXi 5.x)" on page 67 and "Load the Image Into the ESXi 4.1 Server" on page 75 to configure ESXi 5.1 hosts for SAN Boot.

Manual Injection Guidelines

Be aware of the following:

- Creating the custom ISO is accomplished through Microsoft Windows PowerShell—and specifically the VMware vSphere PowerCLI plug-in for PowerShell. The Windows server needs this tool installed.
- Creating the custom ISO is supported on a Windows host server only. The server requirements are determined by the PowerShell application.
- You use a pre-configured ESXi bundle as a baseline, and then inject the Oracle host drivers into it. The ESXi bundle is `VMware-ESXi-5.1.0-799733-depot.zip` and is available from the VMware website.
- You need full administrative rights on the Windows server on which you create the custom ISO.

▼ Inject the Oracle Host Drivers into the ESXi 5.1 Bundle

This procedure uses an example working directory of `\images\New` for the user `adminA`.

1. **Install PowerShell on the Windows server if you have not done so already.**
2. **Install the PowerCLI plug-in if you have not done so already.**
3. **Download the `VMware-ESXi-5.1.0-799733-depot.zip` file to the Windows server.**

4. **Download the current Oracle host drivers.**

5. **Start PowerCLI.**

6. **In PowerCLI, import the ESXi 5.1 bundle and the Oracle host drivers into PowerCLI:**

```
Add-EsxSoftwareDepot -DepotUrl C:\Users\adminA\Desktop\images\New\VMware-ESXi-
5.1.0-799733-depot.zip
Add-EsxSoftwareDepot -DepotUrl C:\Users\adminA\Desktop\images\New\
xsigo_5.3.1.ESX.1-1vmw.500.0.0.472560.zip
```

7. **Type these commands to specify the profile that you want to use when creating the output ISO.**

   The profile determines metadata about the output ISO, such as formatting, compression method, and so on.

```
Add-EsxSoftwarePackage -ImageProfile profile name -SoftwarePackage net-ib-core
Add-EsxSoftwarePackage -ImageProfile profile name -SoftwarePackage net-mlx4-core
Add-EsxSoftwarePackage -ImageProfile profile name -SoftwarePackage net-ib-mad
Add-EsxSoftwarePackage -ImageProfile profile name -SoftwarePackage net-ib-sa
Add-EsxSoftwarePackage -ImageProfile profile name -SoftwarePackage net-ib-ipoib
Add-EsxSoftwarePackage -ImageProfile profile name -SoftwarePackage net-mlx4-ib
Add-EsxSoftwarePackage -ImageProfile profile name -SoftwarePackage net-xscore
Add-EsxSoftwarePackage -ImageProfile profile name -SoftwarePackage net-xsvnic
Add-EsxSoftwarePackage -ImageProfile profile name -SoftwarePackage net-xve
Add-EsxSoftwarePackage -ImageProfile profile name -SoftwarePackage scsi-xsvhba
```

8. **Create a single output ISO containing all required files from the depot.**

Note – For completeness, the following example assumes unsigned drivers.

```
Export-EsxImageProfile -ImageProfile ESXi-5.1.0-799733-standard-xsigo -
ExportToIso
-FilePath C:\Users\adminA\Desktop\images\New\VMware-VMvisor-Installer-5.1.0-
799733_Xsigo.x86_64.iso -NoSignatureCheck
```

Note – Oracle makes every effort to release signed, certified host drivers. However, on some occasions, unsigned drivers might be released. If you receive unsigned host drivers, the `Export-EsxImageProfile` command has the `-NoSignatureCheck` option, which bypasses signature checking. Use the `-NoSignatureCheck` option for unsigned drivers only.

---

9. **Create a boot protocol Server Profile.**

   See "Create a Boot Protocol Server Profile (ESXi 5.x)" on page 67.

10. **Load the image into the server.**

    See "Load the Image Into the ESXi 4.1 Server" on page 75.

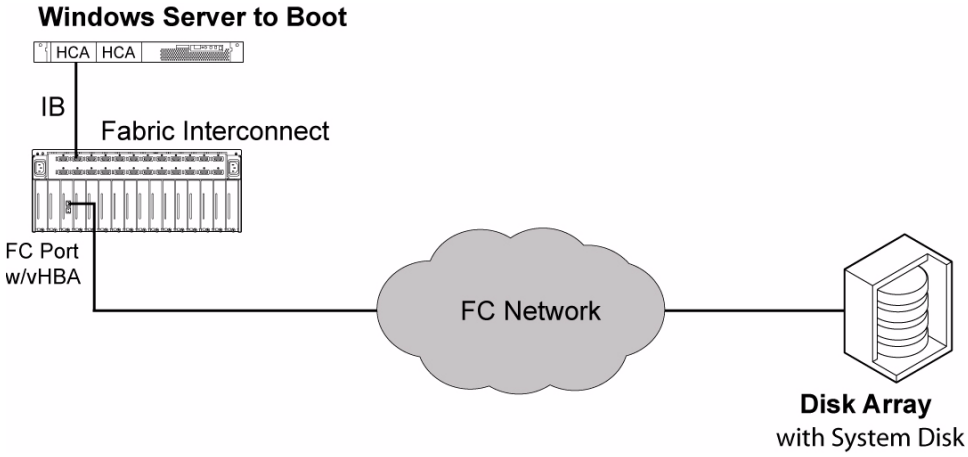
CHAPTER 7

Windows Server SAN Boot

This chapter explains how to set up SAN Boot for a Windows server. It contains the following sections:

Windows SAN Boot Topology

SAN Boot allows you to boot a supported Windows OS from a SAN volume accessed over a VHBA.



Understanding Windows SAN Boot

Windows SAN Boot Conditions

When configuring a Windows server to boot over an Fabric Interconnect VHBA, remember the following requirements:

- Each server requires read/write access to its own dedicated boot volume. Servers cannot share boot volumes.
- You place a fully configured operating system image, including Xsigo host drivers, on the SAN volume. Windows servers do not use the Xsigo `initrd` for SAN Booting.

Requirements for SAN Boot Installation

Before you begin either SAN-boot configuration process, you need:

- CDs or installation files for the supported Windows OS version
- CD or an unzipped archive of the Xsigo host driver software
- Windows Preinstallation Environment (WinPE) for a preliminary boot of the host server

Use the WinPE 3.0 included with the Windows 7 Automated Installation Kit 3.0. For information about creating the WinPE disk, see "Working With the WinPE Disc" on page 90.

If you create a custom WinPE disk for your installations, you must include the Xsigo host drivers so that you can see the VNICs and VHBAs. To include the host drivers, follow the instructions in "Creating the WinPE RAM CD" on page 92.

Working With the WinPE Disc

Review and perform these topics to create a SAN bootable WinPE RAM CD.

Win PE Overview

With Windows SAN Boot, you can use either WinPE disc or Windows Deployment Service (WDS). For illustrative purposes, this procedure documents the WinPE disc method.

WinPE is a scaled-down version of a Windows operating system that provides enough functionality to allow the Windows host server to boot, so that Xsigo host drivers are installed. For more information about WinPE and a WinPE disc, see Microsoft's website. As an alternative, Xsigo has created a sample PE disc with drivers included. You can download the sample WinPE disc by contacting Oracle Support.

When you create the WinPE disc, you load the Xsigo host drivers in PE so that you can access the virtual resources (VNICs and VHBAs).

Note – If you flash the HCA firmware and Option ROM in the Windows Preboot environment (PE), be aware that you must use the 32-bit PE. The reason for this requirement is that the tools to burn the PE image are 32-bit only. If a 64-bit PE system is used you cannot burn the PE image because a 64-bit PE has no 32-bit subsystem where the tools (which are 32-bit) will run.

The overall process of creating the WinPE disc has the following phases:

- Obtaining the WAIK 3.0 package from Microsoft. This package is freeware, so you can download it and install it.
- Obtaining the `loaddriver.bat` file.
- Creating the WinPE RAM CD with Xsigo drivers.

For Windows SAN Boot, WinPE 3.0 is recommended.

▼ Get the Windows 7 Automated Installation Kit

The Windows 7 Automated Installation Kit (WAIK) enables the creation of a bootable WinPE disc. WAIK is freeware, so you can download it from Microsoft:

`http://www.microsoft.com/download/en/details.aspx?id=5753`

1. Download WAIK from Microsoft's website.

2. **Install it on any supported Windows system having a CD/DVD ROM burner installed. This system becomes the Technician PC and writes the PE image onto the CD.**

3. **After installing WAIK, read Microsoft's "Getting Started Guide" for WAIK, as well as any readme file(s) that accompanied WAIK for information about system requirements, known issues, and so on.**

Creating the WinPE RAM CD

These topics describe how to create a bootable WinPE CD with Windows 7 Automated Installation Kit (WAIK).

- "Prerequisites to Creating the WinPE RAM CD" on page 92
- "Create the WinPE RAM CD" on page 93
- "Add Scripting Capability to the WinPE Disc" on page 95

Note – By default, the WinPE environment does not allow running visual basic scripts. If you foresee the need to be able to run scripts in WinPE (for example, to update the Option ROM on the HCA), you add scripting capability to the WinPE CD. For information about adding scripting capabilities, see "Add Scripting Capability to the WinPE Disc" on page 95.

Prerequisites to Creating the WinPE RAM CD

To perform the procedure in this section, you must have the following:

- WAIK, Windows Automated Installation Kit for Windows 7. If you do not have this package, get it now. See "Get the Windows 7 Automated Installation Kit" on page 91.
- CDBurn or DVDBurn (or other compatible burning utility), which allows burning a DVD from an ISO image.
- Technician PC, which is a Windows-compatible workstation or PC with a CD/DVD ROM Drive (RW capable) with Windows AIK installed on it.
- A blank CD or DVD, which is used to create the bootable WinPE RAM CD.
- The `loaddrivers.bat` file, which must be downloaded from the Xsigo FTP site. You will need a valid user name and password to access the file.

▼ Create the WinPE RAM CD

When the listed requirements have been met, you can create a bootable disc by following this procedure:

1. **Type the following command from the WinPE tools command prompt:**

```
copype.cmd x86 c:\winpe_x86
```

2. **Open with Windows AIK command prompt to perform the following steps.**
3. **Change the directory to** `c:\winpe_x86\iso\sources`**.**
4. **Copy the provided** `winpe.wim` **to** `boot.wim`**:**

```
copy C:\Program Files\Windows AIK\Tools\PETools\x86\winpe.wim c:\winpe_x86\iso\
sources\boot.wim
```

5. **Make the** `mount` **directory:**

```
md c:\winpe_x86\iso\sources\mount
```

6. **Mount and edit the** `boot.wim` **file:**

```
dism /mount-wim /wimfile:c:\winpe_x86\iso\sources\boot.wim /index:1
/mountdir:mount
```

7. **Copy the utilities needed to perform the required tasks in WinPE:**

   For imagex.exe:

```
copy C:\Program Files\Windows AIK\Tools\x86\imagex.exe C:\winpe_x86\iso\
sources\mount\windows
copy C:\Program Files\Windows AIK\Tools|petools\x86\bootsect.exe C:\winpe_x86\
iso\sources\mount\windows\
```

   You use `imagex.exe` as a tool when the server is booted into WinPE. You use `dism.exe` to build the PE disc.

8. **Log in to the Xsigo FTP site, and download the** `loaddrivers.bat` **file to the following directory on the local server:**

```
C:\winpe_x86\iso\sources\mount\
```

   For the `loaddrivers.exe` file to load properly, Xsigo's device query tool `xginstdev32.exe` needs to be copied to the root of the PE image:

9. **Copy Xsigo device diagnostic tool to the mounted image:**

```
copy c:\xsigos-3.0.0-whql\xsigo\xginstdev32.exe C:\winpe_x86\iso\sources\mount\
```

10. **Create a temporary folder to contain the WinPE driver package for your environment.**

    For example, a `PEdrivers` folder.

```
md C:\winpe_x86\iso\sources\mount\PEdrivers
```

11. **Copy the WinPE driver package appropriate for your environment into the root of the mounted image:**

```
copy c:\xsigos-3.0.0-whql\xsigo\pedrivers\san-install\x86\*.* C:\winpe_x86\iso\
sources\mount\PEdrivers\
```

12. **As an option, you can add scripting capability, which allows scripts to be executed in the WinPE environment.**

    If you are adding scripting capability to the WinPE disc, do so before burning the WinPE image onto the physical medium (CD or DVD). See "Add Scripting Capability to the WinPE Disc" on page 95.

13. **Unmount the mounted WIM image:**

```
dism /unmount-wim /mountdir:C:\winpe_x86\iso\sources\mount
/commit
```

14. **Create the ISO for the bootable disc:**

```
oscdimg -n -bC:\winpe_x86\etfsboot.com C:\winpe_x86\iso C:\winpe_x86\
winpe_x86.iso
```

15. **Burn the ISO file to a DVD or CD-ROM:**

    - DVD:

```
dvdburn d: C:\winpe_x86\winpe_x86.iso
```

    - CD-ROM:

```
cdburn d: C:\winpe_x86\winpe_x86.iso
```

16. **Boot into WinPE and execute the Xsigo-provided `Loaddrivers.bat` file from the root of X:.**

17. **When the server is booted into the WinPE image, the root of X: should contain the following:**
    - `Xginstdev32.exe`
    - `loaddrivers.bat`
    - `PEdrivers` folder containing the Xsigo WinPE host drivers.

▼ Add Scripting Capability to the WinPE Disc

In some situations, you will need to run visual basic (`.vbs`) scripts in the Windows pre-boot environment. For example, you might need to run the Xsigo Firmware Update script to update the option ROM on an HCA in your Windows server. By default, WinPE does not support .vbs functionality, so if you want to be able to execute a script, you add scripting capability to the WinPE disc that you are creating.

Adding scripting capability is not mandatory for a standard WinPE disc. If you do not explicitly add this functionality, you cannot run scripts from the WinPE disc. In a Windows SAN Boot environment, this is a serious drawback due to not being able to update Option ROM firmware.

Note – If you are adding scripting capability to the WinPE disc, do so before burning the WinPE image onto the physical medium (CD or DVD).

Assuming you have completed the steps in the previous section up through Step 11, you can add scripting capability by following this procedure:

1. **Mount the `boot.wim` image if not already mounted:**

```
dism /mount-wim /wimfile:c:\winpe_x86\iso\sources\boot.wim
/index:1 /mountdir:c:\winpe_x86\iso\sources\mount
```

Note – If you type the command and mount an already mounted `boot.wim`, an error message is displayed. The error can be ignored and it does not affect the procedure.

```
dism /image:c:\winpe_x86\iso\sources\mount /Add-package
/PackagePath:"C:\Program Files\Windows AIK\Tools\PETools\x86\WinPE_FPs\winpe-
scripting.cab"
dism /image:c:\winpe_x86|iso\sources\mount /Add-package
/PackagePath:"C:Program Files\WindowsAIK\Tools\PETools\x86\WinPE_FPs\winpe-
wmi.cab"
```

2. **Return to Step 13 to complete creation of the bootable disc.**

SAN Booting Windows Hosts Without PE

SAN Boot is supported through multiple ways on different Windows OSes. This section documents how to SAN Boot a server by using Shadow Copy. Shadow Copy is part of the Volume Shadow Copy Service (VSS) which is included in the Windows Server 2008 or 2012 OS. With Shadow Copy, you can take a snapshot of the data on a specific local volume on the server. After the volume is captured, you can place it on the appropriate SAN LUN as needed.

Review and perform these topics to enable SAN Boot for a Windows 2008 or 2012 host without a pre-boot environment:

- "Procedure Overview (Windows 2008 or 2012 Without PE)" on page 96
- "SAN Boot Considerations (Windows 2008 or 2012)" on page 97
- "Configure SAN Boot With Shadow Copy" on page 97

Procedure Overview (Windows 2008 or 2012 Without PE)

To configure the server for SAN Booting, you create some commands and scripts and enter the correct content. To create these files, you will use any standard text editor (for example, Notepad). These files are created as part of the procedure, and the file content is given at the appropriate point of the procedure. You will also need to call different files that are embedded in Windows.

The procedure has the following general work flow:

1. Create the correct batch files and scripts on the server.

   The following user-configured files are created:

   - `fixbcd.cmd`
   - `fixregistry.cmd`
   - `fixbootsector_W2k8.cmd`
   - `removescript.script`

   To support these files, the following standard Windows files are required. Make sure that they are available on the server:

- `Bootsect.exe`
- `Imagex.exe`
- `bcdboot.exe`
- `diskshadow.exe`
- `reg.exe`

2. Copy the boot image onto the SAN LUN.
3. Create a SAN Bootable Server Profile that connects the server to the SAN LUN where the boot image exists.
4. Edit the server's BIOS to set the Xsigo VHBA as the boot device.

   By doing so, the server will retrieve the OS from the SAN LUN and boot to runtime.

SAN Boot Considerations (Windows 2008 or 2012)

Following this procedure provides support for SAN Booting a Windows Server 2008 R2 or 2012 R2 host. However, after completing this procedure, be aware of the following considerations:

- The server will no longer be able to boot to either safe mode or the recovery console.
- The SAN LUN must be use either master boot record (MBR) or GUID Partition Table (GPT) partitioning. Dynamic disks are not supported with Xsigo host drivers.
- If you are SAN Booting multiple servers, you perform this procedure once for each of the servers. Also, each server must have its own SAN LUN and its own bootable VHBA connected to it. You cannot connect the server to the same SAN LUN and have all servers boot from a common boot image on a central SAN LUN.

▼ Configure SAN Boot With Shadow Copy

This procedure documents a way to convert an existing server that boots off of its local drive to a Xsigo SAN-Booted server. After the server is converted, the server's local hard drive is not used to boot the server to its OS. In fact, the drive must be completely disabled. No local disks should be available as boot devices when the conversion is complete.

To convert an existing server that locally boots from hard disk to a SAN Boot system, you will perform various parts of this procedure on the server and Oracle's Xsigo Fabric Interconnect.

Note – This procedure assumes that one partition is used on one disk.

If multiple partitions are created on the disk, capturing the image is more complex. You will have to modify the tools listed below to capture both the small boot partition where the BootMGR and Boot Configuration Database (BCD) exist as well as capturing the partition where the OS is installed.

For this procedure, assume that the SAN LUN is `h:`.

Note – This procedure is written for Windows 2008, but is still valid for Windows 2012. However, you might need to change filenames to be respective to the OS type for certain steps.

---

1. **On the server, install the OS to a local disk in the server.**

   Do not attempt to install the OS to the SAN LUN.

2. **Install the Xsigo host drivers and reboot as required.**

   If you need more information, see the Windows Host Software chapter of the *Fabric Interconnect Hardware and Host Drivers Installation Guide*. If needed, update the HCA firmware and Option ROM to the correct version for your hardware type.

3. **On the Fabric Interconnect, add a Server for the physical server you plan to SAN Boot.**

```
add server-profile webapps1
```

4. **Add a VHBA with one LUN to your Server Profile.**

   The LUN should be the appropriate size to contain the OS and applications you intend to install.

```
add vhba vh1.webapps1 8/1
```

5. **Bind the Server Profile to the server by using the** `set server-profile` *name* `connect` *hostname@director-name*`:`*ServerPort* **command.**

```
set server-profile webapps1 connect webserver1@Director1:ServerPort18
```

Note – Because the Xsigo host drivers are already installed (see Step 2), the host is identifiable by host name, as shown in the example. If you need to find the port on which the server is connected, you can type the `show fabric-port` command, and scroll to find the connection for the relevant host.

6. **On the server, using Disk Manager, bring the newly added disk online.**
7. **Using Disk Manager, format the disk and mark it Active.**
8. **Using a text editor, create the backupscript.cmd file.**
9. **In the `backupscript.cmd` file, type the following content:**

```
diskshadow.exe /s backupscript.script
```

When this script runs, it captures the local disk and places a copy onto the LUN.

10. **Create the `backupscript.script` file with the following content.**

```
SET CONTEXT PERSISTENT NOWRITERS
SET VERBOSE ON
BEGIN BACKUP
ADD VOLUME C: ALIAS systemVolumeShadow
CREATE
EXPOSE %systemVolumeShadow% p:
EXEC w2k8_clone_diskshadowImage.cmd
END BACKUP
```

11. **Create the `w2k8_clone_diskShadowImage.cmd` file with the following content.**

```
imagex.exe /CAPTURE /boot P: c:\SanBoot.wim "Windows Server 2008 R2"
format.com h: /q /V:Sanboot /y
bootsect.exe /nt60 h: /force
imagex.exe /APPLY c:\SanBoot.wim 1 H: /VERIFY
diskshadow.exe /s removescript.script
call FixBCD.cmd
call FixBootSector-2k8.cmd
call fixregistry.cmd
```

Note – For Windows 2012 R2, use the filename `w2k12_clone_diskShadowImage.cmd`.

12. **Delete the existing BCD on the SAN LUN:**

```
delete h:\Boot\BCD
```

13. **Create a new BCD on the SAN LUN:**

```
bcdboot.exe c:\Windows  /s h:
```

This executable creates a new BCD on the LUN and forces the new configuration information into the new BCD.

14. **Create the** `FixBCD.cmd` **file with the following content.**

```
bcdedit.exe -store h:\boot\bcd /set {bootmgr} device boot
bcdedit.exe -store h:\boot\bcd /set {default} device boot
bcdedit.exe -store h:\boot\bcd /set {default} osdevice boot
```

When this executable is run, it updates the boot database with the new boot partition information.

15. **Create the** `fixbootsector_W2k8.cmd` **file with the following content.**

```
bootsect.exe /nt60 h: /force
```

When this executable is called, it places the boot sector on the new LUN.

16. **Create the** `fixregistery.cmd` **file with the following content.**

```
reg.exe load hklm\sanboot h:\windows\system32\config\system
reg.exe delete hklm\sanboot\mounteddevices /va /f
reg.exe unload hklm\sanboot
```

When this executable is called, it removes any previously assigned drive letters. If the old drive letters are not removed, the system will boot to the H: drive. If the local disk controller had more than just the boot info and the OS, any additional drive letters will have to be reassigned once the server boots to SAN.

17. **Create the** `removescript.script` **file with the following content.**

```
delete shadows exposed p:
```

18. **On the Fabric Interconnect, make the Server Profile SAN Bootable, by setting the san-boot flag for the Server Profile.**

    Use the `set server profile` *name* `san-boot` *VHBA WWPN LUN ID* command.

```
set server-profile webapps1 san-boot vh1.webapps1 11:22:33:44:55:66:77:88 1
```

19. **Reboot the server and interrupt the POST to enter the BIOS configuration utility.**

20. **Remove the local disk from the boot devices list by disabling the on-board storage device.**

    If disabling the on-board storage is not preferable, you can remove the physical hard drive(s) from the server, but be aware that some array controllers behave unpredictably if the drives are removed.

21. **Set the HCA high enough in the boot devices list order to enable the Xsigo Option ROM to complete before another device is considered for booting the server.**

22. **Save the settings and exit the BIOS.**

    The next time the server boots, it will use the bootable VHBA.

23. **If you are converting multiple servers from local boot to SAN Boot, repeat this procedure as needed for each server.**

Note – The `sysprep.exe` tool is not supported for this method of SAN Booting Windows Server 2008 R2 or 2012 R2 hosts.

CHAPTER 8

Firmware and Option ROM Levels

Whenever you configure a server to boot remotely, you must ensure that the HCA firmware is at the correct version and that the option ROM is installed.

Verifying Firmware and Option ROM

This section provides instructions for these tasks:

- "Verify HCA Firmware and Option ROM for Linux Hosts" on page 103
- "Verify HCA Firmware and Option ROM for Windows Hosts" on page 106

▼ Verify HCA Firmware and Option ROM for Linux Hosts

1. **Log in into the host server as root.**

2. **Unpack the Xsigo HCA firmware package on the server.**

```
rpm -ivh xsigo-hca-firmware_number.i386.rpm
```

Note – Replace xsigo-hca-firmware_2.6.6.i386.rpm with the xsigo firmware for your server. Supported host drivers for each operating system are listed in the release notes.

This step unpacks the `xg_config` tool, which you can use to update the HCA firmware and Option ROM.

3. **Check the firmware and option ROM level:**

a. Log in as `root` of the host server.

b. View the firmware and option ROM levels.

```
/opt/xsigo/bin/xg_config
################################################################################
# Main menu
################################################################################

Selected card:
   Node GUID          : '0002:c902:0020:4934'
   Board ID           : 'MT_0150000001'
   CA type            : 'MT25208'
   Firmware version   : '5.3.0'
   Hardware version   : 'a0'
   Option ROM version : 'XgBoot Version 2.2.0'
```

Oracle's XgOS supports the following minimum HCA firmware levels for Linux hosts:

- InfiniHost Single-Port HCA: 1.3.0 or higher
- InfiniHost Dual-Port HCA: 5.3.0 or higher
- ConnectX Dual-Port HCA: 2.8.0 or higher
- ConnectX-2 Single and Dual-Port HCA: Firmware version 2.8.0 or higher

If your firmware and XgBoot versions are as shown above, you can skip Step 4.

4. On your Linux host server, upgrade the HCA firmware and the option ROM if necessary.

a. If you haven't already done so, log in as `root` on the host server.

b. Upgrade the Xsigo HCA firmware package on the server.

```
rpm -Uvh xsigo-hca-firmware_number.i386.rpm
```

Note – Replace `xsigo-hca-firmware_number.i386.rpm` with the Xsigo host driver for your server. Supported host drivers for each operating system are listed in the release notes.

---

c. Upgrade the firmware and option ROM.

```
/opt/xsigo/bin/xg_config
################################################################################
# Main menu
################################################################################
```

```
Selected card:
   Node GUID          : '0002:c902:0020:4934'
   Board ID           : 'MT_0150000001'
   CA type            : 'MT25208'
   Firmware version   : '5.3.0'
   Hardware version   : 'a0'
   Option ROM version : 'XgBoot Version 2.2.11'

1) Flash HCA Firmware
2) Flash HCA Firmware + Option ROM
3) Flash Option ROM
4) Change selected card
0) Quit
Select option>
```

d. If you are using SAN Boot or might decide to in the future, select option 2.

Otherwise, select option 1. In the following screen output example, option 2 was selected:

```
########################################################################
# Flash HCA Firmware + Option ROM Menu
########################################################################

Selected card:
   Node GUID          : '0002:c902:0020:4934'
   Board ID           : 'MT_0150000001'
   CA type            : 'MT25208'
   Firmware version   : '5.3.0'
   Hardware version   : 'a0'
   Option ROM version : 'XgBoot Version 2.2.11'

1) 5.3.0 (XgBoot Version 1.5)
2) 5.1.400 (XgBoot Version 1.5)
0) Return to previous menu
Select firmware to use>
********************************************************
```

e. Select the most recent firmware (the one displayed first).

You must reboot for the firmware upgrade to take effect. However, you can wait to reboot until you have upgraded the host drivers.

Note – For other servers that were not used for remastering the ISO, you can just boot once from the remastered ISO which is used as a golden master image to boot any number of Citrix Xen 5.6 FP1 servers.

▼ Verify HCA Firmware and Option ROM for Windows Hosts

Oracle's XgOS supports the following minimum HCA firmware levels for Windows hosts:

- Single Port HCA: 1.2.0
- Dual Port HCA: 5.3.0
- Connect-X: 2.6.0

When the `Xg_FWUpdate.vbs` script runs, it first checks the current HCA Device ID and firmware level and determines if an update is required.

1. **Start a command prompt by following Start->Run....**
2. **Change directory (cd) to `%programfiles%\Xsigo Systems\Support\FirmwareUpdate`, which is the directory where the HCA firmware update script is located.**

```
cd %programfiles%\Xsigo Systems\Support\FirmwareUpdate
```

3. **From the prompt, run the script:**

```
cscript Xg_FWUpdate.vbs

Microsoft (R) Windows Script Host Version 5.6
Copyright (C) Microsoft Corporation 1996-2001. All rights reserved.

########################################################################
# (C) 2011 XSIGO SYSTEMS Inc. All rights reserved. This material may not be
# reproduced, displayed, modified or distributed without the express prior
# written permission of the copyright holder.
########################################################################

########################################################################

# Main Menu
################################################################################

Selected HCA Card Number: 0
HCA Device ID : mt25218_pciconf0
Image Type : failsafe
I.S. Version : 1
Device ID :
Chip Revision : a0
GUID Descr : node port1 port2 sys image
GUIDs : 0002c9020021f1f0 0002c9020021f1f1 0002c9020021f1f2 0002c9020021
```

```
f1f3
BOARD ID : mt_0370110001
VSD :
PSID : mt_0370110001
FW Version :
HCA mlx FW Ver : 5.1.400
1) Flash HCA Firmware
2) Change selected card
0) Quit
Select option>
```

4. When prompted, type `1` to enter the Flash HCA Firmware Menu.

```
Select option> 1

###############################################################################
# Flash HCA Firmware Menu
###############################################################################
Selected HCA Card Number: 0
HCA Device ID : mt25218_pciconf0
Image Type : failsafe
I.S. Version : 1
Device ID :
Chip Revision : a0
GUID Descr : node             port1           port2            sys image
GUIDs : 0002c902002410a0 0002c902002410a1 0002c902002410a2 0002c902002410a3
BOARD ID : mt_0370110001
VSD :
PSID : mt_0370110001
FW Version :
HCA mlx FW Ver : 5.1.400

1) 5.3.0
2) 5.1.400
0) Return to previous menu

Select Firmware to Burn>
```

5. **When prompted, select the firmware version that you want to burn onto the HCA in the Windows server.**

   Do not attempt to abort the firmware upgrade process after it has started. The following example shows updating the HCA with firmware version 5.3.0.

```
Select Firmware to Burn> 1
Upgrading HCA firmware 5.1.400 to 5.3.0
This Will Flash HCA with Firmware file .\Image\fw-25218-5_2_0-mhea28-xtc_a1-
a2.bin
Please do not interrupt the burn process or reboot the machine...
Wait till burn completes ...
.....................................................................
---------------------------------------------------------------------
The firmware on one or more of the HCAs has been upgraded.
It is recommended to reboot the machine in order for changes to take effect.
---------------------------------------------------------------------
Press Enter key to continue
```

6. **Press Enter to exit the update script.**

7. **Whenever you run the script and burn firmware on one or more HCAs, shut down the Windows server and then start it to bring the HCAs up.**

   If the HCAs have been updated, this cold boot is required bring them online with the new firmware.

Index