FreeBSD/Xen How-to guide

Roger Pau Monné
Citrix Systems R&D,
royger@FreeBSD.org

March 3, 2015

Abstract

This article provides detailed instructions about how to setup a FreeBSD/Xen PVH Dom0 (host). It also describes how to create and manage guests, including detailed instructions about how to install and configure some of them.

1 Required materials

In order to setup a FreeBSD/Xen Dom0 an Intel box that supports EPT and IOMMU is required (AMD systems are not yet supported). Almost all recent Intel Core chips have this features and they have been present on the Xeon series for longer. It is recommended that the box used has serial output in order to debug possible problems with Xen and FreeBSD. Without a serial cable it is almost impossible to diagnose early boot problems.

The hardware used for the demo will be an Intel NUC5i3MYHE:

- Intel Core i3-5010U.
- 8GB of RAM.
- 500GB Hard drive.
- Intel 10/100/1000Mbps Ethernet card.
- Serial output.

The following materials are also needed in order to perform the installation:

- mfsBSD image: used to boot the system and install FreeBSD.
- FreeBSD installation sets.

This materials are available at:

http://people.freebsd.org/~royger/install_sets/

Before installing FreeBSD the mfsBSD image has to be copied to an USB drive using dd:

```
# dd if=/path/to/mfsbsd.img of=/path/of/usb/dev bs=1m
```

An Internet connection will also be needed in order to perform the tutorial.

2 Installing FreeBSD

The bare metal install of FreeBSD will be done using mfsBSD and the zfsinstall script. In the opinion of the author this is one of the fastest and simplest ways to setup a FreeBSD systems with ZFS-on-Root.

The first step is to boot the box using the USB drive that has been flashed with the mfsBSD image. The mfsBSD image provided has been configured to use both the video adapter and the serial console, so installation can be done from either input methods. The first step is to install FreeBSD on the hard drive using zfsinstall:

```
# gpart destroy -F ada0
# zfsinstall -d ada0 -u http://people.freebsd.org/~royger/install_sets/ -s 4g
```

Once the process finishes we would like to perform some modifications to the installed system before rebooting. The first step is enabling serial output, so we need to add the following to /boot/loader.conf:

```
boot_multicons="YES"
boot_serial="YES"
comconsole_speed="115200"
console="comconsole,vidconsole"
boot_verbose="YES"
```

And modify /etc/ttys in order to enable the login prompt on the serial console:

```
ttyu0   "/usr/libexec/getty std.115200" dialup  on  secure
```

Now we are going to configure the network interface in order to have network connectivity and automatically start sshd on boot. We need to add the following to /etc/rc.conf:

```
ifconfig_em0="DHCP"
sshd_enable="YES"
sendmail_enable="NONE"
```

Since this is a test system, and we haven't created any users we are also going to enable ssh root logins by adding the following line to /etc/ssh/sshd_config:

```
PermitRootLogin yes
```

Now we can reboot into the newly installed system in order to install Xen.

3 Installing Xen

Before installing Xen we need to install all the dependencies required to build it. This can be easily done using the `pkg(7)` tool:

```
# pkg install git bash python bcc glib pkgconf yajl gmake pixman \
        perl5 markdown bison gettext gawk gcc47
```

We are going to use the development version of Xen, so we need to fetch the sources using git:

```
# git clone git://xenbits.xen.org/xen.git
```

Now we need to compile and install it:

```
# cd xen
# ./configure HOSTCC=gcc47 CC=gcc47
# gmake -j8 xen HOSTCC=gcc47 CC=gcc47
# cp xen/xen /boot/
# gmake -j8 install-tools HOSTCC=gcc47 CC=gcc47
# mkdir -p /var/run/xen/
```

Once the install process has finished we can move on to configuring the system. The first step is going to be to tell the loader to boot the Xen kernel and FreeBSD, this is done by adding the following to `/boot/loader.conf`:

```
xen_cmdline="dom0_mem=2048M dom0pvh=1 console=com1,vga iommu=debug guest_loglvl=all loglvl=all"
xen_kernel="/boot/xen"

vfs.zfs.arc_max="1G"
if_tap_load="YES"
```

In this example we are giving 2GB to Dom0, and we set a couple of debug options in order to get more verbose output from Xen. The next step is setting up a bridge in order to provide network to the guests. We would like to use the mac address of the physical network interface as the mac address of the bridge, so we also add the following to `/boot/loader.conf`:

```
net.link.bridge.inherit_mac=1
```

Then we need to modify `/etc/rc.conf` to configure the bridge interface:

```
cloned_interfaces="bridge0"
ifconfig_bridge0="addm em0 SYNCDHCP"
ifconfig_em0="up"
```

It is also recommended to remove the limit on the number of wired pages, since the Xen toolstack makes use of them, this is done by modifying `/etc/sysctl.conf`:

```
vm.max_wired=-1
```

And finally we need to enable the xencommons init script in `/etc/rc.conf` and enable the Xen console in `/etc/ttys`:

```
xencommons_enable="YES"
```

```
xc0     "/usr/libexec/getty Pc"         xterm   on  secure
```

4 The Xen toolstack

The default toolstack that comes with Xen is called xl. This is a cli utility that can be used to manage guests running on the system. One of the first commands that we can try is:

```
# xl list
Name                                ID   Mem VCPUs      State       Time(s)
Domain-0                            0   2048     4      r-----      14.5
```

This outputs a list of the guest that are currently running. As we can see we only have one guest that's the FreeBSD Dom0. The list of all supported xl commands can be fetched using `xl help`.

5 Creating guests

This section contains examples about how to setup some common guest types.

5.1 Debian PV guest

In order to setup a pure Linux PV guest we are going to use Debian. Debian already provides a kernel and initramfs that can be used to setup a PV guest, and a config file that can be used with Xen. First we need to fetch all those parts:

```
# fetch http://ftp.nl.debian.org/debian/dists/wheezy/main/installer-amd64/ \
        current/images/netboot/xen/initrd.gz
# fetch http://ftp.nl.debian.org/debian/dists/wheezy/main/installer-amd64/ \
        current/images/netboot/xen/vmlinuz
# fetch http://ftp.nl.debian.org/debian/dists/wheezy/main/installer-amd64/ \
        current/images/netboot/xen/debian.cfg
```

We are also going to create a ZVOL in order to provide a hard drive to the guest:

```
# zfs create -V 20g tank/debian
```

And finally we need to edit the config file `debian.cfg` in order to set the correct paths:

```
#====================================================================
# AT INSTALLATION TIME
#====================================================================

kernel = "vmlinuz"
ramdisk = "initrd.gz"

#====================================================================
# TO BOOT INSTALLED SYSTEM
#
# Comment all of the above installation options and uncomment the
# below instead
#====================================================================

#bootloader="pygrub"
```

```
16
17 #================================================================================
18 # STANDARD OPTIONS
19 #================================================================================
20 #
21 # The following options are common to both installation time and normal booting.
22 #
23 # Only a subset of the available options are included below.
24 # See /usr/share/doc/xen-utils-common/examples for full examples.
25
26 #--------------------------------------------------------------------------------
27
28 # Initial memory allocation (in megabytes) for the new domain.
29 memory = 1024
30
31 # A name for your domain. All domains must have different names.
32 name = "debian"
33
34 # Number of Virtual CPUS to use, default is 1
35 vcpus = 2
36
37 #--------------------------------------------------------------------------------
38 # Define network interfaces.
39
40 vif = ['bridge=bridge0']
41
42 #--------------------------------------------------------------------------------
43 # Define disks
44
45 disk = ['phy:/dev/zvol/tank/debian,xvda,w']
```

This guest has been configured to use 2 vCPUs and 1GB of RAM. The virtual network card will be added to the `bridge0` automatically by the Xen toolstack. Now we can create the guest and proceed with the installation:

```
1 # xl create -c debian.cfg
```

Once the install process has finished we will need to tweak the guest config file so it boots from the hard drive. This will require changing the top of the config file so it looks like:

```
1 #================================================================================
2 # TO BOOT INSTALLED SYSTEM
3 #
4 # Comment all of the above installation options and uncomment the
5 # below instead
6 #================================================================================
7
8 bootloader="pygrub"
9 [...]
```

Now we can boot into the installed system:

```
1 # xl create -c debian.cfg
```

5.2 FreeBSD PVHVM guest

We can setup a FreeBSD guest using two different methods, we can either use the pre-build VM images, or we can perform a normal install using the ISOs. In this example we are going to use the ISOs so the install process resembles a bare metal FreeBSD install. The first step consists in downloading the install disk and creating a ZVOL to use as disk:

```
# fetch ftp://ftp.freebsd.org/pub/FreeBSD/releases/ISO-IMAGES/10.1/ \
        FreeBSD-10.1-RELEASE-amd64-bootonly.iso
# zfs create -V 20g tank/freebsd
```

Then we need to create the guest configuration file:

```
# This configures an HVM rather than PV guest
builder = "hvm"

# Guest name
name = "freebsd"

# Initial memory allocation (MB)
memory = 1024

# Number of VCPUS
vcpus = 2

# Network devices
vif = [ 'bridge=bridge0' ]

# Disk Devices
disk =  [
        '/dev/zvol/tank/freebsd,raw,hda,rw',
        '/root/freebsd/FreeBSD-10.1-RELEASE-amd64-bootonly.iso,raw,hdc:cdrom,r'
        ]

vnc = 1
vnclisten = "0.0.0.0"
serial = "pty"
```

Now we can create the guest:

```
# xl create freebsd.cfg
```

And attach to the vnc console in order to perform the install:

```
# vncviewer <host>
```

Once the install has finished we can remove the ISO image from the guest configuration file and boot into it. We are going to configure the guest to use the serial console so we can get the boot output and a login prompt. In order to do so we need to modify `/boot/loader.conf` and `/etc/ttys`:

```
boot_multicons="YES"
boot_serial="YES"
comconsole_speed="115200"
console="comconsole,vidconsole"
```

```
ttyu0   "/usr/libexec/getty std.115200" dialup  on  secure
```

Now we can reboot the guest and see how it boots from the serial console using the xl toolstack:

```
# xl shutdown -w freebsd
# xl create -c freebsd.cfg
```

5.3 Windows HVM guest

Installing a Windows guest is quite similar to FreeBSD. In this case the install image is not provided, for this example we are going to use a Windows XP SP3 image. The first step is creating the disk for the guest:

```
# zfs create -V 20g tank/windows
```

And the config file:

```
# This configures an HVM rather than PV guest
builder = "hvm"

# Guest name
name = "windows"

# Initial memory allocation (MB)
memory = 1024

# Number of VCPUS
vcpus = 2

# Network devices
vif = [ 'bridge=bridge0' ]

# Disk Devices
disk =  [
        '/dev/zvol/tank/windows,raw,hda,rw',
        '/root/windows/winxpsp3.iso,raw,hdc:cdrom,r'
        ]

vnc = 1
vnclisten = "0.0.0.0"
serial = "pty"
usbdevice = "tablet"
```

Now we can create the guest:

```
# xl create windows.cfg
```

And as we did to perform the FreeBSD install we need to attach to the vnc console:

```
# vncviewer <host>
```

The image that we are using in this case is automated, so we don't need to perform any manual steps during installation.