If you don't have your own machine to install and play with Xen, I suggest you should create an account on Netlab www.netlab.cc.gatech.edu) by joining the XenHack project. You should explore how to use Netlab on that website. You can create your own experiment and allocate one node on which you can install Xen.

Some useful instructions
http://swiki.cc.gatech.edu:8080/phd/798

1. fdisk /dev/sda --> For creating the desired layout of Debian and swap partitions on the drive
2. mkfs /dev/sda# --> For formatting a file system prior to installation
3. mkswap /dev/sda# --> For setting up a swap partition

Its assumed here that you know how to acquire a login and connect to the machines via ssh as well as telnet. Else you can email me asking for more information.

To power cycle a machine, use project's web interface or *node_reboot* command.

Debian installation

The instructions to install Debian can be found here
http://www.debian.org/releases/stable/i386/apcs04.html.en

Follow the instructions for installing Debian Linux on the Netlab machine and complete the installation on two partitions. The first partition could be used as primary for Dom0 booting and the second one would be used for booting a guest VM eventually

After the basic install is over, you need to modify two files
1. /etc/network/interfaces to add the configured Ethernet device so that it can contact dhcp server and retrieve an IP address for itself
2. /etc/inittab to change the baud rate from 9600 to 115200 in the line that contains /sbin/getty ttyS0

1) install grub on /dev/sda6 (Assuming /dev/sda6 is the partition where primary Debian is)

```
# mount /dev/sda6 /mnt
# mkdir -p /mnt/boot/grub
# cp /boot/grub/stage* /boot/grub/e2fs_stage1_5 /mnt/boot/grub
# umount /mnt
# grub
grub> root (hd0,5)
grub> setup (hd0,5)
grub> quit
```

2) modify lilo conf

add following lines to /dev/sda2:/etc/lilo.conf

other=/dev/sda6
label=FC4Xen

3) run the lilo command.

4) add these lines to grub

```
default=1
timeout=5
#splashimage=(hd0,0)/boot/grub/splash.xpm.gz
#hiddenmenu
serial --unit=0 --speed=115200 --word=8 --parity=no --stop=1
terminal serial
title Debian
        root (hd0,6)
        kernel /boot/vmlinuz-2.6.8-2-686-smp root=/dev/sda7 ro
        console=ttyS0,115200
        initrd /boot/initrd.gz
```

After all this, you can boot and see if the installation works properly.

Grub modifications need to be done only for the primary install. For the secondary Debian installation only the step 4 needs to be done with appropriate kernel image and init files.

Some useful commands:
For creating multiple copies of guest domain Debian installations which might be needed in order to boot more than one guest domains on one machine:

1. Make the DomU partition of proper size (or change the size of some existing installation using resize2fs e.g. 'resize2fs -pF /dev/sda8'
2. Make a file image of the partition using dd e.g. 'dd if=/dev/sda8 of=guest_img bs=4K count=786432' for a 3GB partition using 4K block size
3. Now create more partitions using fdisk and install this image on these partitions e.g. 'dd if=guest_img of=/dev/sda10'. Make sure the partition is bigger than the image.
4. You might have to create different swap partitions and modify the /etc/fstab on the new partition images to point to the right devices

After the second installation of the Debian, the grub.conf will look like

```
default=1
timeout=5
#splashimage=(hd0,0)/boot/grub/splash.xpm.gz
#hiddenmenu
serial --unit=0 --speed=115200 --word=8 --parity=no --stop=1
terminal serial
```

```
title PrimaryDebian
        root (hd0,6)
        kernel /boot/vmlinuz-2.6.8-2-686-smp root=/dev/sda7 ro
        console=ttyS0,115200
        initrd /boot/initrd.gz
title SecondaryDebian
        root (hd0,8)
        kernel /boot/vmlinuz-2.6.8-2-686-smp root=/dev/sda9 ro
        console=ttyS0,115200
        initrd /boot/initrd.gz
```

(assuming that the installations are on /dev/sda7 and /dev/sda9. /dev/sda6 and /dev/sda8 could be the swap partitions for these installations and you can verify that in /etc/fstab)

After installing Xen, there will be an additional entry which would look like:

```
title Xen0
        root (hd0,8)
        Kernel /boot/xen.gz com1=115200,8n1
        Module /boot/vmlinuz-2.6-xen0 root=/dev/sda9 ro
        console=ttyS0,115200
```

Dom 0 will take care of booting the guest domains and so they do not need entries

You can change the boot order to boot Xen 0 by default.

In order to change the configuration parameters (say enabling SMP support in Xen) of the currently compiled Xen kernel, cd into the xen-unstable.hg directory which was made while installing xen using mercurial. Run 'make menuconfig'

After config is done, run 'make KERNELS="linux-2.6-xen0 linux-2.6-xenU" {dist/install}

If you make a distro, run 'make install' with the KERNELS option again. You can run make with the **-j option** to parallelize the process

Installing Xen

For getting Xen on local machine, first you need to install mercurial.

1. Getting Mercurial
http://www.selenic.com/mercurial/release/?M=D

2. Getting Xen

After mercurial is installed, Xen sources can be downloaded on the machine using the *hg* command from the Xen website (where the mercurial repositories are located)
http://xenbits.xensource.com

You should preferably get the xen-unstable.hg for development purposes

In order to replicate the Xen source from web,
#hg xen-unstable.hg

3. Building Xen Source

Installation might keep failing due to missing packages like make, patch, crypto-devel, bzip, ncurses-devel etc. Keep installing these packages until make world succeeds

4. Updating Xen Sources and Recompiling

Whenever you need to update the Xen source tree, cd into the Xen root directory which would be the xen-unstable.hg directory, run the following command to pull the latest sources

#hg pull -u

After this run the *make "KERNEL=…" dist* and *make "KERNEL=…" install* and reboot the machine

(make dist is required if you want to install this xen on some other machine too. Basically make dist puts all the binaries in the dist directory which can be transfered to other machines to install xen there.)

To see details of the Xen version that you are running
#hg tip

Xen Root = the unstable folder where hg downloads code

Scheduler and other Xen related files reside in *Xen-Root/xen (/common for sched code)*

5. Accessing Internet from guest domains

6. Changing config
   Execute the following in the dom0 root e.g.
   /root/vishakha/xen-unstable.hg/linux-2.6.16.33-xen0/

   #Make menuconfig
   #make … dist (from unstable if you want to make a dist else only install)
   #make … install

7. If things go wrong as they sometimes will -
   1. If xm console hangs - check /var/log/xen/xend-debug.log
      If it says some thing like user tool set may have to be built again, run
      #make -C tools clean
      from the xen-unstable folder and then
      #make … install

Xen README

http://www.xensource.com/xen/about.html

What is Xen?
============

Xen is a Virtual Machine Monitor (VMM) originally developed by the Systems Research Group of the University of Cambridge Computer Laboratory, as part of the UK-EPSRC funded XenoServers project. Xen is freely-distributable Open Source software, released under the GNU GPL. Since its initial public release, Xen has grown a large development community, spearheaded by XenSource Inc, a company created by the original Xen development team to build enterprise products around Xen.

The 3.0 release offers excellent performance, hardware support and enterprise-grade features such as x86_32-PAE, x86_64, SMP guests and live relocation of VMs. This install tree contains source for a Linux 2.6 guest; ports to Linux 2.4, NetBSD, FreeBSD and Solaris will follow later (and are already available for previous Xen releases).

This file contains some quick-start instructions to install Xen on your system. For full documentation, see the Xen User Manual. If this is a pre-built release then you can find the manual at: dist/install/usr/share/doc/xen/pdf/user.pdf

If you have a source release, then 'make -C docs' will build the manual at docs/pdf/user.pdf.

Quick-Start Guide - **Source Release *** (This is what we use)
==========

First, there are a number of prerequisites for building a Xen source release. Make sure you have all the following installed, either by visiting the project webpage or installing a pre-built package provided by your Linux distributor:
* GCC (preferably v3.2.x or v3.3.x; older versions are unsupported)
* GNU Make
* GNU Binutils
* Development install of zlib (e.g., zlib-dev)
* Development install of Python v2.3 or later (e.g., python-dev)
* Development install of curses (e.g., libncurses-dev)
* bridge-utils package (/sbin/brctl)
* iproute package (/sbin/ip)
* hotplug or udev

[NB. Unless noted otherwise, all the following steps should be performed with root privileges.]

1. Download and untar the source tarball file. This will be a file named xen-unstable-src.tgz, or xen-$version-src.tgz. You can also pull the current version from the SCMS that is being used (Bitkeeper, scheduled to change shortly).

tar xzf xen-unstable-src.tgz

Assuming you are using the unstable tree, this will untar into xen-unstable. The rest of the instructions use the unstable tree as an example, substitute the version for unstable.

2. cd to xen-unstable (or whatever you sensibly rename it to). The Linux, netbsd and freebsd kernel source trees are in the $os-$version-xen-sparse directories.

On Linux:

3. For the very first build, or if you want to destroy existing .configs and build trees, perform the following steps:

make world
make install

This will create and install onto the local machine. It will build the xen binary (xen.gz), and a linux kernel and modules that can be used in both dom0 and an unprivileged guest kernel (vmlinuz-2.6.x-xen), the tools and the documentation.

You can override the destination for make install by setting DESTDIR to some value.

The make command line defaults to building the kernel vmlinuz-2.6.x-xen. You can override this default by specifying KERNELS=kernelname. For example, you can make two kernels - linux-2.6-xen0 and linux-2.6-xenU - which are smaller builds containing only selected modules, intended primarily for developers that don't like to wait for a full -xen kernel to build. The -xenU kernel is particularly small, as it does not contain any physical device drivers, and hence is only useful for guest domains.

To make these two kernels, simply specify

KERNELS="linux-2.6-xen0 linux-2.6-xenU"

in the make command line.

So the actual make that you issue will look like
make **KERNELS="linux-2.6-xen0 linux-2.6-xenU"** world
make **KERNELS="linux-2.6-xen0 linux-2.6-xenU"** install

If you want to build an x86_32 PAE capable xen and kernel to work on machines with >= 4GB of memory, use XEN_TARGET_X86_PAE=y on the make command line.

4. To rebuild an existing tree without modifying the config:
 # make dist

This will build and install xen, kernels, tools, and docs into the local dist/ directory.

You can override the destination for make install by setting DISTDIR
to some value.

make install and make dist differ in that make install does the right things for your local machine (installing the appropriate version of hotplug or udev scripts, for example), but make dist includes all versions of those scripts, so that you can copy the dist directory to another machine and install from that distribution.

5. To rebuild a kernel with a modified config:

make linux-2.6-xen-config CONFIGMODE=menuconfig (or xconfig)
make linux-2.6-xen-build
make linux-2.6-xen-install

Depending on your config, you may need to use 'mkinitrd' to create
an initial ram disk, just like a native system e.g.
depmod 2.6.16-xen
mkinitrd -v -f --with=aacraid --with=sd_mod --with=scsi_mod initrd-2.6.16-xen.img 2.6.16-xen

Quick-Start Guide - **Pre-Built Binary** Release
==========

[NB. Unless noted otherwise, all the following steps should be
performed with root privileges.]

1. Install the binary distribution onto your filesystem:

sh ./install.sh

Among other things, this will install Xen and Xen-ready Linux
kernel files in /boot, kernel modules and Python packages in /lib,
and various control tools in standard 'bin' directories.

2. Configure your bootloader to boot Xen and an initial Linux virtual
machine. Note that Xen currently only works with GRUB and pxelinux
derived boot loaders: less common alternatives such as LILO are
not supported. You can most likely find your GRUB menu file at
/boot/grub/menu.lst: edit this file to include an entry like the
following:

title Xen 3.0 / XenLinux 2.6
kernel /boot/xen-3.0.gz console=vga
module /boot/vmlinuz-2.6-xen root=<root-dev> ro console=tty0
module /boot/initrd-2.6-xen.img

NB: Not all kernel configs need an initial ram disk (initrd), but
if you do specify one you'll need to use the 'module' grub directive
rather than 'initrd'.

The linux command line takes all the usual options, such as

root=<root-dev> to specify your usual root partition (e.g.,
/dev/hda1).

The Xen command line takes a number of optional arguments described in the manual. The most common is 'dom0_mem=xxxM' which sets the amount of memory to allocate for use by your initial virtual machine (known as domain 0). Note that Xen itself reserves about 32MB memory for internal use, which is not available for allocation to virtual machines.

3. Reboot your system and select the "Xen 3.0 / XenLinux 2.6" menu option. After booting Xen, Linux will start and your initialisation scripts should execute in the usual way.