Operating Systems **CSI3131** ***LabExtra*** *Winter 2011*

Virtual Machines
Introduction to File Systems

Goal

1. Use of a virtual machine (using Virtual PC)
2. Explore the administration of Linux file systems

Running a Virtual Machine

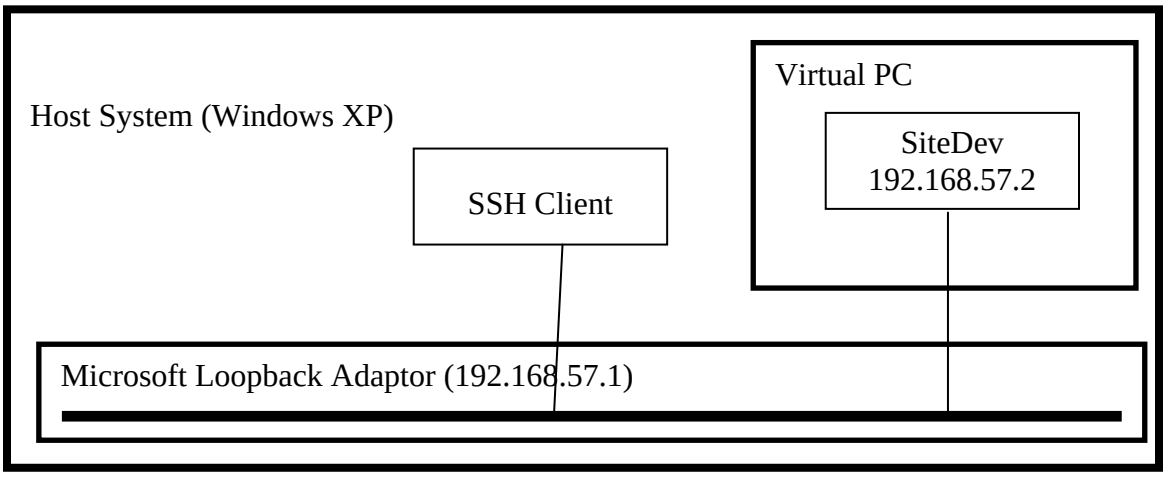
The lab exercise will give you the opportunity to work with a virtual machine in which runs a Linux operating system (system called SiteDev); the virtual machine runs under the host operating system Windows XP. You will NOT be able to use the SITE Linux systems, as you will need the superuser account *root* to complete your work. The environment required to complete the lab contains the following components (the figure below shows how these components are related):

Virtual PC – is Windows software that creates a virtual machine to allow any other operating system to run within the Windows OS.

Linux – is actual Linux OS that was installed in the virtual machine in the same fashion as it would have been installed in real hardware.

Microsoft Loopback Adaptor – is Windows software that provides a virtual LAN within Windows to allow that host system to communication with the virtual system. This is essentially a private LAN and does not allow the Linux to access any external LAN connected to the host system.

SSH Client – is software that essentially plays the role of a terminal (it does provide encryption so that communication with the remote system is secure). SSH connects to the Linux virtual machine just like it connects to the SITE Linux system.



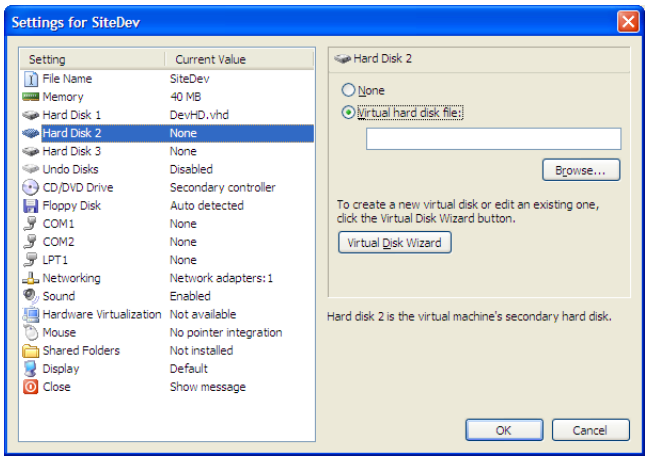
Virtual PC and the Linux virtual machines have been installed in on PCs in SITE 0110 lab and the PCs in SITE lab 2052. This is the environment used for this lab and in preparation for the final lab of the session. The document "CSI3131VPCInstructions.pdf" provides instructions on setting up and running Virtual PC and SiteDev. If interested you may download into your personal system Virtual PC is from http://www.microsoft.com/windows/virtualpc/default.mspx (see Virtual Campus for a previous version that runs under XP) and download the "sitedev" Linux Virtual Machine (including documentation) from Virtual Campus (see the Labs page).

When you have been able to run Virtual PC/Site Dev, shutdown the Linux operating system using the command "*shutdown –h now*"; it will be necessary to be logged in as root (password is "*site*") to execute this command. If you are logged in as "*test*", execute "*su – root*" and give the password "*site*" to move to the root account. Continue with the lab after shutting down *SiteDev* (DO NOT shut down Virtual PC).

Preparing a second Hard Disk

Virtual PC defines hard drives as files (virtual disks). You will find the file "*DevHD.vhd*" in the directory *C:\VirtualMachine\SiteDev*. This file is the Linux OS main virtual drive that contains the OS software. Note in the window below "*Settings for SiteDev*" that *DevHd.vhd* is defined as "*Hard Disk 1*". You will add a second hard drive, "*Hard Drive 2*" and prepare the drive for use with the Linux OS to gain some insight into basic file system concepts (to help you prepare for the course material on file systems, the final lab and assignment). Use the following steps to add a hard disk file to the virtual machine.

1. In the directory "*C:\VirtualMachine\SiteDev*", unzip the file *HardDrive2.vhd* (from the zip file provided *HardDrive2.zip*). This file will serve as a hard drive for creating a number of partitions and various file systems within the partitions. The following steps allow you to add the file as a hard drive to the Linux Virtual machine.
   a. Highlight the virtual machine named *siteDev* and click on *Settings*.
   b. Click on *Hard Drive 2*.
   c. On the right hand side of the window, select *Virtual hard disk file* and used the *Browse* button to add the *HardDrive2.vhd* file as the hard drive (see the illustration to the left).
   d. Close the window by clicking on OK.



2. You can now start the Linux OS by clicking on the *Start* button in the main VirtualPC window.

3. A new terminal window will appear. Linux will boot and you will be prompted for a user name.
4. Start a Secure Shell (SSH), and connect the Linux OS, by connecting to the address 192.168.57.2. Do your work with SSH.
5. Log in as root, and complete the following exercises to manage the hard drive.

Exercises in Managing the Hard Drive.

Examining the Hard Drive Devices

From the point of view of the OS, a hard drive consists of a large array of bytes that are divided into blocks. Each device is represented in the Linux file system by a single file. As root you can manipulate this file.

1. Two drives should exist within your system, one for each hard drive; */dev/hda* (main drive, i.e. *Hard Drive 1*) and */dev/hdb* (*Hard Drive 2*). List these devices with "`ls –l /dev/hda /dev/hdb`" and you should obtain an output similar to:
   ```
   brw-rw----    1 root     disk       3,   0 Jan 30  2003 /dev/hda
   brw-rw----    1 root     disk       3,  64 Jan 30  2003 /dev/hdb
   ```
   Notice that the line starts with a "*b*", this indicates that the device is a block device. Also note that only root has permissions to access this file (i.e. the hard drives directly).
2. The devices can be treated as a single large file. In fact, accessing the contents of the device files allows you to examine how a file system has been setup. For example "`cat /dev/hda | od –c >log`" will read the contents of */dev/hda* and pipe the output to *od*, a utility that interprets the binary data read from the standard input to write it to the standard output in readable form as characters (when possible) (run `man od` for details on *od*). Do not run the command very long (only a few seconds) as you will fill up your hard drive. Type *Cntrl-C* to terminate the command. Examine the contents of the file *log* to see the data read from the hard drive. You should be able to find some text that is stored in some files.

Partitioning the Drive

1. Now that you know were the hard drives are, lets do something with the second hard drive you added to the system. The hard drive reference by */dev/hdb* is empty (or at the very least, contains garbage). The OS needs to create a file systems (organize the contents) into which it can create a directory and store files. We shall study such organizations later in the course.
2. Before creating file systems, it is possible to divide the drive into pieces, called partitions. You can see how */dev/hda* is partitioned by executing "`fdisk /dev/hda`" and then enter "`p`" for print. This prints the partition table for the drive. (Careful to make any alterations to the hard drive partitions since this could lead to making the Linux OS inoperable). You obtain find output similar to:

```
[root@sitedev root]# fdisk /dev/hda
```

```
Command (m for help): p

Disk /dev/hda: 2621 MB, 2621251584 bytes
128 heads, 63 sectors/track, 634 cylinders
Units = cylinders of 8064 * 512 = 4128768 bytes

   Device Boot    Start       End    Blocks   Id  System
/dev/hda1   *         1        25    100768+  83  Linux
/dev/hda2            26       569   2193408   83  Linux
/dev/hda3           570       634    262080   82  Linux swap
```

The output can be interpreted as follows:

- The first three lines give physical information on the hard drive. In particular, the first line gives the total size in terms of MB and bytes. The second line attempts to provide information on the physical makeup of the hard drive – this is often logical and does not represent the actual physical characteristics of the drive (consider that in this case the drive is virtual). The third line gives the number of bytes located in a single cylinder. This is important since the partitions are expressed in terms of cylinders. Note that this hard drives consists of 634 cylinders (numbered 1 to 634).
- Three partitions have been created on the hard drive; each can be referenced using the file names */dev/hda1*, */dev/hda2/*, and */dev/hda3* (under column *Device*).
- The first partition */dev/hda1* occupies cylinders 1 to 25. The asterisk in the Boot column indicates that the Boot files are located in this partition (in particular, Block 0 in this partition is the boot block).

3. Now examine the Hard Drive 2 using the command "`fdisk /dev/hdb`". You should obtain output that resembles:

```
[root@sitedev root]# fdisk /dev/hdb
Device contains neither a valid DOS partition table, nor Sun, SGI or
OSF disklabel
Building a new DOS disklabel. Changes will remain in memory only,
until you decide to write them. After that, of course, the previous
content won't be recoverable.

Warning: invalid flag 0x0000 of partition table 4 will be corrected
by w(rite)

Command (m for help): p

Disk /dev/hdb: 1048 MB, 1048190976 bytes
32 heads, 63 sectors/track, 1015 cylinders
Units = cylinders of 2016 * 512 = 1032192 bytes

   Device Boot    Start       End    Blocks   Id  System

Command (m for help):
[root@sitedev root]#
```

Notice that since the drive does not contain a partition table, so *fdisk* creates by default a DOS disklabel for it (see the initial messages. The utility *fdisk* provides the means for defining other types of disklabel (SUN, BSD, etc). We shall experiment with

partitioning the disk using the DOS disk label. Examine the size of the cylinders – 1032192 bytes. How many cylinders exist on the disk?

4. Consider partitioning the drive into four parts, each with the sizes indicated in the table below. Divide the number of cylinders and complete the following table so that you can create four partitions of the following sizes:

Partition Number	Size	Number of cylinders	Start cylinder	End cylinder
1	500 MB			
2	248 MB			
3	236 MB			
4	64 MB			
Total				

5. Now create the partitions. Here is the interaction with *fdisk* for creating the first partition (input by the administrator are in bold and italics):

```
Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 1
First cylinder (1-1015, default 1):
Using default value 1
Last cylinder or +size or +sizeM or +sizeK (1-1015, default 1015):
485

Command (m for help): p

Disk /dev/hdb: 1048 MB, 1048190976 bytes
32 heads, 63 sectors/track, 1015 cylinders
Units = cylinders of 2016 * 512 = 1032192 bytes

   Device Boot    Start       End    Blocks   Id  System
/dev/hdb1             1       485    488848+  83  Linux
```

Repeat to create the other 3 partitions as primary partitions 2 to 4. When you have created the partitions, display (using p) the partition table and note the device names assigned to each partition (first column). Save your changes and exit using the "w" command as shown below.

```
Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.
[root@sitedev root]#
```

Creating file systems.

Partitioning a drive, simply divides the drive. It does not yet allow the operating system to use them.

1. Examine how partitions from the main hard drive are mounted onto the file system. Execute the command "`df`". This command should provide how the various "mounted" file systems are mounted into the Linux directory tree. It should give you an output similar to:

```
[root@sitedev root]# df
Filesystem           1K-blocks      Used Available Use% Mounted on
/dev/hda2              2158952    904748   1144536  45% /
/dev/hda1                97570      5146     87386   6% /boot
```

   Note that the partitions of the main drive */dev/hda2* and */dev/hda1* are identified in the first column under the heading *Filesystem*. Each of the file systems are attached to a point in the Linux directory – see the column "*Mounted on*". The partition "*/dev/hda2*" is mounted on the root of the directory "*/*" while the "*/dev/hda1*" (that contains the boot files) are mounted at the directory "*/boot*". The other columns give the size of the file systems and how the disk space is used.

2. Ensure that you are in the home directory for root and create four directories to mount the partitions you created in the second hard drive using "`mkdir mnt1 mnt2 mnt3 mnt4`". Then try mounting the first partition onto *mnt1* using the command "`mount /dev/hdb1 mnt1`"

```
[root@sitedev root]# mkdir mnt1 mnt2 mnt3 mnt4
[root@sitedev root]# ls
anaconda-ks.cfg  install.log  install.log.syslog  mnt1  mnt2  mnt3
mnt4
[root@sitedev root]# mount /dev/hdb1 mnt1
mount: you must specify the filesystem type
 [root@sitedev root]#
```

   An error occurs, because the no file system exists on the partition yet and the Linux OS cannot recognize the type of file system in the partition. It is necessary to format the contents of the partition to allow the possibility of creating directories and allocating space to files (such organization of a file system shall be studied later in the course).

3. Create a file system on */dev/hdb1* using the command "`mkfs.ext2 /dev/hdb1`". The command will create on /dev/hd1 partition a filesystem of type *ext2* (this type is an extension of the original Linux file system - *minix*). The output of the command should produce something like:

```
[root@sitedev root]# mkfs.ext2 /dev/hdb1
mke2fs 1.32 (09-Nov-2002)
Filesystem label=
OS type: Linux
Block size=1024 (log=0)
Fragment size=1024 (log=0)
122400 inodes, 488848 blocks
24442 blocks (5.00%) reserved for the super user
First data block=1
60 block groups
8192 blocks per group, 8192 fragments per group
2040 inodes per group
Superblock backups stored on blocks:
        8193, 24577, 40961, 57345, 73729, 204801, 221185, 401409

Writing inode tables: done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 23 mounts or
180 days, whichever comes first.  Use tune2fs -c or -i to override.
```

4. Now repeat the *mount* command to mount the file system on *mnt1* ; then execute the command *df*. You should obtain output something like:

```
[root@sitedev root]# mount /dev/hdb1 mnt1
[root@sitedev root]# df
Filesystem           1K-blocks      Used Available Use% Mounted on
/dev/hda2              2158952    904716   1144568  45% /
/dev/hda1                97570      5146     87386   6% /boot
/dev/hdb1               473400        13    448945   1% /root/mnt1
```

   Notice that the size of the file system is lower that the assigned space of 500 MB when the partition was created. This gives an idea of how much of the partition was used to format the file system.

5. Now create other files systems (of different types) and mount them onto the Linux diretory using the commands shown:

```
[root@sitedev root]# mkfs.ext3 /dev/hdb2
mke2fs 1.32 (09-Nov-2002)
Filesystem label=
OS type: Linux
Block size=1024 (log=0)
Fragment size=1024 (log=0)
60480 inodes, 241920 blocks
12096 blocks (5.00%) reserved for the super user
First data block=1
30 block groups
```

```
8192 blocks per group, 8192 fragments per group
2016 inodes per group
Superblock backups stored on blocks:
        8193, 24577, 40961, 57345, 73729, 204801, 221185

Writing inode tables: done
Creating journal (4096 blocks): done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 39 mounts or
180 days, whichever comes first.  Use tune2fs -c or -i to override.
[root@sitedev root]# mkfs.jfs /dev/hdb3
mkfs.vfat 2.8 (28 Feb 2001)
[root@sitedev root]# mkfs.vfat /dev/hdb4
mkfs.msdos 2.8 (28 Feb 2001)

[root@sitedev root]# mount /dev/hdb2 mnt2
[root@sitedev root]# mount /dev/hdb3 mnt3
[root@sitedev root]# mount /dev/hdb4 mnt4
[root@sitedev root]# df
[root@sitedev root]#
Filesystem           1K-blocks      Used Available Use% Mounted on
/dev/hda2              2158952    904776   1144508  45% /
/dev/hda1                97570      5146     87386   6% /boot
/dev/hdb1               473400        13    448945   1% /root/mnt1
/dev/hdb2               234283      4127    218060   2% /root/mnt2
/dev/hdb3               228588       160    228428   1% /root/mnt3
/dev/hdb4                62356         0     62356   0% /root/mnt4
[root@sitedev root]#
```

6. Adding the –*T* argument in the *df* command as shown below provides an additional column that shows the type of the file system in the partition.

```
[root@sitedev mnt4]# df -T
Filesystem    Type   1K-blocks      Used Available Use% Mounted on
/dev/hda2     ext2     2158952    904776   1144508  45% /
/dev/hda1     ext2       97570      5146     87386   6% /boot
/dev/hdb1     ext2      473400        13    448945   1% /root/mnt1
/dev/hdb2     ext3      234283      4127    218060   2% /root/mnt2
/dev/hdb3      jfs      228588       160    228428   1% /root/mnt3
/dev/hdb4     vfat       62356         0     62356   0% /root/mnt4
[root@sitedev root]#
```

7. For more information on the different file systems consult the following links and man pages:
   JFS: http://oss.software.ibm.com/jfs/
   EXT2/EXT3 : man fs, http://e2fsprogs.sourceforge.net/ext2intro.html
   VFAT : man mkfs.vfat, http://en.wikipedia.org/wiki/File_Allocation_Table#Long_File_Names_.28VFAT.2C_LFNs.29