Linux Basic and Installation

(Course code LX02)

Student Exercises

ERC 6.0

Authorized

IBM Training

IBM certified course material

Trademarks

IBM® is a registered trademark of International Business Machines Corporation.

The following are trademarks of International Business Machines Corporation in the United States, or other countries, or both:

AIX®
AT®
DB2®
Domino®
Lotus®
Notes®
OS/2®
400®

PS/2® is a trademark or registered trademark of Lenovo in the United States, other countries, or both.

PostScript is either a registered trademark or a trademark of Adobe Systems Incorporated in the United States, and/or other countries.

Microsoft, Windows and Windows NT are trademarks of Microsoft Corporation in the United States, other countries, or both.

Intel and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

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

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

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

October 2008 edition

The information contained in this document has not been submitted to any formal IBM test and is distributed on an “as is” basis without any warranty either express or implied. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer’s ability to evaluate and integrate them into the customer’s operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will result elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

Contents

Trademarks

The reader should recognize that the following terms, which appear in the content of this training document, are official trademarks of IBM or other companies:

IBM® is a registered trademark of International Business Machines Corporation.

The following are trademarks of International Business Machines Corporation in the United States, or other countries, or both:

AIX® AT® DB2®
Domino® Lotus® Notes®
OS/2® 400®

PS/2® is a trademark or registered trademark of Lenovo in the United States, other countries, or both.

PostScript is either a registered trademark or a trademark of Adobe Systems Incorporated in the United States, and/or other countries.

Microsoft, Windows and Windows NT are trademarks of Microsoft Corporation in the United States, other countries, or both.

Intel and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

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

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

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

Exercises Description

The objective of the **Linux Power User** exercises is to let the you become familiar with installing and running Linux on your personal workstation. To achieve this, a variety of real-world exercises are performed, aimed at simulating real-world tasks.

Each exercise unit consists of two parts:

Exercise Instructions — This section contains what it is you are to accomplish. There are no definitive details on how to perform the tasks. You are given the opportunity to work through the exercise given what you learned in the unit presentation, utilizing the unit Student Notebook, your past experience, the online documentation and maybe a little intuition.

Exercise Instructions With Hints — This section is an exact duplicate of the Exercise Instructions section except that in addition, specific details and/or hints are provided to help step you through the exercise. A combination of using the Instructions section along with Instructions With Hints section can make for a rewarding combination providing you with no hints when you don't want them and hints when you need them.

In this last section, multiple ways to accomplish the same task are often provided. Where this has been done, the various methods are separated by an **-OR-**

All exercises and hints apply both to Fedora, RHEL, and SLES equally, unless mentioned.

Exercise 1. Introduction to Linux

Notice: This unit has no exercises. This page is here to ensure that unit numbers and exercise numbers stay synchronized.

Exercise 2. Installing Linux

What this exercise is about

This exercise lets you install Linux.

What you should be able to do

After completing this exercise, you should have experience with:

- Preparing a system for installation
- Partitioning a system
- Installing Linux

Required Materials

- A set of installation CDs or a network capable boot CD for your distribution

Exercise Instructions

Note

The exercises in this course material have been designed for and tested on the following three distributions:

- Fedora Core 7
- Red Hat Enterprise Linux (RHEL) 5.1 Enterprise Client
- SUSE Linux Enterprise Desktop (SLED) 10 SP1

If you are using one of these three distributions, follow the instructions below that apply to your distribution. If you are using another distribution, or another version of one of the three distributions above, then your instructor gives you additional information.

Optional: Depending on the circumstances, your instructor might have to loan you a full set of CDs for each distribution, so that you can perform a CD-based install, or your instructor might have to loan you an installation CD so that you can perform a network-based install.

If you need to perform a network install, your instructor will give you additional information, specifically:

- The install method: NFS (FTP or HTTP is also available)
- The IP address that is to be used for your workstation, if DHCP is not used
- The name or IP address of the install server
- The path to the installation images on the install server

Installing Fedora Core 7 or Red Hat Enterprise Linux 5.1 Desktop

__ 1. Turn on or reboot the computer.

__ 2. Your systems needs to boot to the network instead of from the hard drive or CD. Depending on your hardware, you might need to press F12 or F9 or F1 to force a network boot. You see a screen with a text version of the IBM logo that gives you different installation options. You will simply type an appropriate number and then press Enter.

   - For Fedora 7, type **11** and press **Enter**.
   - For RHEL 5.1 Client, type **13** and press **Enter**.

Note

If your system is not booting to the network, ask your instructor for additional assistance.

__ 3. Choose the language for the installation process, click **OK**.

__ 4. Choose your keyboard model and layout, click **OK**.

__ 5. On the TCP/IP screen, you want to disable **IPv6 Support**, click **OK**.

__ 6. At the initial graphic screen, click **Next**.

__ 7. If you are installing RHEL 5.1, you are asked to input an “Installation Number” -- choose **Skip Installation Number**; then click **OK**.

- Next, click in the pull-down that says **Remove Linux partitions** and choose **Create custom layout;** then click **Next**.
- The Disk Druid screen displays and shows the current layout of your disks. You first need to **Delete** all partitions manually. You can then start adding Linux partitions. Make sure you create three additional partitions:
  - One partition is used as root partition. Its Mount Point should be “/”, the File System Type should be **ext3**, and the size of this partition should be 6 GB (**6000 MB**).
  - Add a boot partition. Its Mount Point should be /**boot**, the File System Type should be **ext3,** and the size of this partition should be **100 MB**.
  - The last partition is used as swap space, which does not have a mount point. The size should be equal to the amount of real memory, with a maximum of **1000 MB**, and the File System Type should be **swap** (the Mount Point shows **<Not Applicable>**)

__ 8. **Let the instructor check your partition configuration before you save it!** After the instructor has checked your partition configuration, click **Next**.

__ 9. The installation program now allows you to configure your boot loader. You can accept all defaults here, then click **Next**.

__ 10. Configure your network adapters. Your instructor should tell you whether to use **DHCP** or will provide you with the IP Address, Netmask, Network and Broadcast addresses, with the Hostname, Gateway and DNS addresses. Enter these values, double-check them, and click **Next**.

__ 11. Now select your **Time Zone** and clear the **UTC** check box, then click **Next**.

__ 12. In the next screen you need to set the root password. For convenience in the class, set the root password to **ibmlnx**: then click **Next**.

__ 13. At the Software screen, select **Customize now**". Click **Next** and add the **KDE Desktop Environment** group. Also, click **Development** and add **Development Tools**; then click **Next**.

__ 14. Note the location of the log file and click **Next**.

__ 15. The installation program now formats the filesystems and installs Linux. This might take anywhere from 5 minutes to an hour, depending on the number of packages to install, and the speed of the computer.

While installing, you can see what is going on in detail by switching to the third virtual terminal with **Ctrl+Alt+F3**. Switch back to the graphical installation screen with **Ctrl+Alt+F7**. Also, take a look at other virtual screens (1 through 6).

__ 16. When your installation is complete -- click **Reboot** to reboot your system.

__ 17. When your Linux system boots for the first time, the Fedora/RHEL Setup Agent is started. Click the **Forward** button.

__ 18. Read the License Agreement, if asked; then select **Yes, I agree** and click **Forward**.

__ 19. The next screen allows you to configure firewall rules. Choose **Disabled** from the **Firewall** list and click **Forward** and then **Yes**. Then **Disable** the SELinux Setting the same way, and click **Forward** and then click **Yes**.

__ 20. RHEL 5.1 will ask you if you want **Kdump** enabled -- you do *not* so click **Forward**.

__ 21. Check the date and time. If the network has an NTP server, configure it here as well. Click **Forward**.

__ 22. Fedora 7 will then show your hardware profile -- Click **Do not send** then **Forward**, then **No**, **do not send**.

__ 23. RHEL 5.1 Client will ask if you want to **Set Up Software Updates**. Click **No> Forward> No thanks> Forward**.

__ 24. Add a personal user account for yourself, with a password you make up yourself, then click **Forward**.

__ 25. Verify that your sound card has been detected and is configured correctly, by playing a test sound. Then click **Yes**, then **Finish**.

__ 26. RHEL 5.1 Client will ask if you want to install any additional CDs, click **No> Finish> OK**.

__ 27. The installer might ask that you reboot your machine at this time -- do so and the installation is complete.

Installing SUSE Linux Enterprise Desktop 10 SP1

__ 28. Turn on or reboot the computer.

__ 29. Your systems needs to boot to the network instead of from the hard drive or CD. Depending on your hardware, you might need to press F12 or F9 or F1 to force a network boot. You see a screen with a text version of the IBM logo that gives you different installation options. You will simply type an appropriate number and then press **Enter**. For **SLED 10**, type **15** and press **Enter**.

Note

If your system is not booting to the network, ask your instructor for additional assistance.

__ 30. Select the language and keyboard map (if asked) for the network configuration process and click **Next**.

__ 31. On the License Agreement screen, choose **Yes, I agree** and click **Next**.

__ 32. If your system has already been installed with Linux, then a window might open stating this. Select **New installation** and click **Next**.

__ 33. Browse through the autodetected installation settings, and make changes if required:

- Make sure your **Time Zone** and **UTC**/**Local** choices are correct.
- Click **Partitioning;** then click **Create Custom Partition Setup**, then **Next**. Now click **Custom Partitioning** and **Next**. Delete all partitions that you see. Make these choices to create three partitions:
    - Click> **Primary**> **OK**. Format as '**ext3**', highlight the number in the **End** box and change it to **6GB**, enter "/" in the **Mount Point** field, then click **OK**.
    - Click> **Primary**> **OK**, Format as '**ext3**', highlight the number in the **End** box and change it to **100MB**, enter "/**boot**" in the **Mount Point** field, then click **OK**.
    - Click> **Primary**> **OK**, Format as '**Swap**', highlight the number in the **End** box and change it to **1GB**,and then click **OK**> click **Finish**.
- Click **Software**, make sure that you add **KDE** and **C/C++ Compiler and Tools** to the default selection of software; then click **Accept**.

Click **Accept** (on any and all pop-up screens) and click **Install**. SLED 10 now installs itself. This takes 5 minutes to an hour, depending on the speed of your computer.

__ 34. Note that SLED 10 might automatically reboot midway through the installation process. This is normal. When the initial boot screen appears, do nothing so that the system boots from hard disk. The installation process should continue automatically.

__ 35. Next, you need to enter the root password. For convenience in class, use **ibmlnx** as the root password and click **Next**.

__ 36. On the **Hostname and Domain Name** screen, select the box next to **Change Hostname via DHCP** check box and then click **Next**.

__ 37. The **Network Configuration** screen allows you to configure your network. Make sure all detected values are okay. If necessary, consult your instructor for IP addresses and such. Then, click the word **Enabled** next to **Firewall** to toggle the firewall setting; then click **Disable IPv6** and click **Next**.

__ 38. Even if you have an Internet connection, click **No, skip the test** for your Internet connection test and click **Next**.

__ 39. Select **Local (/etc/passwd)** as **User Authentication Method** screen. Click **Next**.

__ 40. Add a local user account for yourself, using a secret password. Do *not* select **Automatic Login**. Then click **Next**.

__ 41. SuSEConfig now executes several configuration scripts. This might take several minutes.

__ 42. If you feel like it, read the **Release Notes** for this version. Then click **Next**.

__ 43. Check your Hardware Configuration, (You may change the Graphics Card/Monitor settings if you know what they should be.) Then click **Next,** and then **Finish**.

__ 44. Select the **Clone for Autoyast** check box and log in when the system is ready.

End of exercise

Exercise 3. Using the System

What this exercise is about

The purpose of this exercise is to become familiar with Linux, the command syntax and some basic commands. The exercise also serves to show some multiuser concepts.

What you should be able to do

At the end of the lab, you should be able to:

- Switch between virtual terminals
- Log in to a Linux system and change passwords
- Execute basic commands
- Use the **wall** and **write** commands to communicate with other users
- Use keyboard control keys to control command line output
- Use the mouse to copy and paste commands
- Use the command history
- Lock a Linux system
- Log out of a Linux system

Exercise Instructions

Logging in on a virtual terminal

In this section, you are going to log in to the system using both text and graphical virtual terminals.

__ 1. If the install went correctly then you should now see a graphical login prompt. If this is not the case, ask your instructor to fix this. (You learn how to do this yourself later in the course.)

__ 2. Verify that you indeed have seven different virtual terminals. Cycle through them by pressing Alt+F*n*, where *n* is the terminal number you want to access. Use Ctrl+Alt+F*n* when you are in a graphical terminal.

__ 3. In your first virtual terminal (tty1), log in to the system with your own username, which you also configured when installing the system.

__ 4. In your second virtual terminal (tty2), log in to the system as root. After having logged in, look at the command prompt. Do you notice anything different from the command prompt in the other virtual terminals?

__ 5. In your seventh virtual terminal (tty7), log in to the system with your own username and password.

__ 6. Open a terminal window. Take a look at the command prompt. Does it differ from the command prompt on tty1? Why or why not?

Basic Commands

In this section, we are going to execute some basic commands, in order to familiarize yourself with the command syntax of Linux, and the fact that you are currently on a multiuser, multi-tasking system.

All commands in this section are executed on virtual terminal seven (the graphical login prompt where you are logged in as yourself), using the terminal window you just opened, unless specified otherwise.

__ 7. Change your password. Memorize this password because no one can find out your password if you forget it.

__ 8. Display the system's date.

__ 9. Display the whole calendar for the year 2008.

__ 10. Display the month of January for the year 1999 and 99. Are 1999 and 99 the same?

__ 11. Generate a list of all users present on your system.

__ 12. Display your login name.

__ 13. Display the login information of your own user account, and of root.

__ 14. Clear your screen.

__ 15. Print the text **Out to lunch** on your display.

__ 16. Make sure you are willing to receive messages.

__ 17. Write the message **Out to lunch** to the display of root. Check whether root got the message.

__ 18. Write the message **Out to lunch** to the display of all users. Check whether everybody on your system got the message.

Keyboard and Mouse Tips

__ 19. The bash shell has a command history function. View some of the commands you have entered. Try to alter one of these commands; then run the command again.

__ 20. Your terminal has a buffer that keeps track of the output of your commands. View the output of the previous commands.

__ 21. Bash supports command and filename completion with the TAB character. Try to use this feature, both on commands and on filenames.

__ 22. Both in a text terminal and an emulated terminal in the graphical desktop, try to re-execute commands by scrolling up a little, selecting the command with the left mouse button, and then pasting it onto the same terminal again with the middle mouse button.

Also try this across different text and graphical terminals.

Note

SuSE does not enable **gpm** by default; so your mouse won't work in a text terminal when you are using SuSE.

Using the history

__ 23. Use the history command to view the last 20 commands you typed.

__ 24. Execute one of the commands from the history list.

__ 25. Execute the **echo** command again, this time changing the word *lunch* to *dinner.*

__ 26. Bash also supports searching in the history. Try this feature as well.

Locking terminals

Note

Not all distributions install **vlock** and **xlock** by default. If **vlock** and **xlock** are not installed, then you learn how to do that in Exercise 15 - Basic System Configuration.

__ 27. Lock a virtual terminal. Can you switch to another virtual terminal while this one is locked? Unlock the terminal.

__ 28. Lock the console. Can you switch to another virtual terminal now? Unlock the console.

__ 29. Lock the graphical environment and then unlock it again.

Logging off

__ 30. Log off all users that are logged in at any TTY.

End of exercise

Exercise Instructions with Hints

» All hints are identified with the two greater-than symbols like this one.

» All hints apply to all distributions equally, unless mentioned.

Logging in on a virtual terminal

In this section, you are going to log in to the system using both text and graphical virtual terminals.

__ 1. If the install went correctly, then you should now see a graphical login prompt. If this is not the case, ask your instructor to fix this. (You learn how to do this yourself later in the course.)

__ 2. Verify that you indeed have seven different virtual terminals. Cycle through them by pressing Alt-F*n*, where *n* is the terminal number you want to access. Use Ctrl+Alt+F*n* when you are in a graphical terminal.

» **<Ctrl+Alt+F1>**

» **<Alt+F2>**

» **<Alt+F3>**

» **<Alt+F4>**

» **<Alt+F5>**

» **<Alt+F6>**

» **<Alt+F7>**

__ 3. In your first virtual terminal (tty1), log in to the system with your own username, which you also configured when installing the system.

» **<Ctrl+Alt+F1>**

» Login: **(your username)**

» Password: **(your password)**

__ 4. In your second virtual terminal (tty2), log in to the system as root. After having logged in, look at the command prompt. Do you notice anything different from the command prompt in the other virtual terminals?

» **<Alt+F2>**

» Login: **root**

» Password: **ibmlnx**

__ 5. In your seventh virtual terminal (tty7), log in to the system with your own username and password.

» Login: **(your username)**

» Password: **(your password)**

__ 6. Open a terminal window. Take a look at the command prompt. Does it differ from the command prompt on tty1? Why or why not?

» On a Fedora or Red Hat system, a terminal window can be started from the “Red Hat” button in the upper left hand corner; System Tools; Terminal. You can also drag this icon to your quick launch bar, if you want to.

» On a SuSE system, the terminal icon can be found in the launch bar.

Basic Commands

In this section, you execute some basic commands, to familiarize yourself with the command syntax of Linux. All commands in this section are executed on virtual terminal 7 (the graphical login prompt where you are logged in as yourself), using the terminal window you just opened, unless specified otherwise.

__ 7. Change your password. Memorize this password because no one can find out your password if you forget it.

» $ **passwd**

» Changing password for <username>

» (current) UNIX password: **(your current password)**

» New UNIX password: **(your new password)**

» Retype new UNIX password: **(your new password)**

» passwd: all authentication tokens updated successfully

__ 8. Display the system's date.

» $ **date**

__ 9. Display the whole calendar for the year 2008.

» $ **cal 2008**

__ 10. Display the month of January for the year 1999 and 99. Are 1999 and 99 the same?

» $ **cal 1 1999**

» $ **cal 1 99**

__ 11. Generate a list of all users present on your system.

» $ **who**

- OR -

» $ **finger**

__ 12. Display your login name.

» $ **whoami**

- OR -

» $ **who am i**

__ 13. Display the login information of your own user account, and of root.

» $ **finger <username>**

» $ **finger root**

__ 14. Clear your screen.

» $ **clear**

__ 15. Print the text **Out to lunch** on your display.

» $ **echo Out to lunch**

__ 16. Make sure you are willing to receive messages

» $ **mesg y**

__ 17. Write the message **Out to lunch** to the display of root. Check whether root got the message.

» $ **write root**

» **Out to lunch**

» **<Ctrl-D>**

» **<Ctrl-Alt-F2>**

» **<Alt-F7>**

__ 18. Write the message **Out to lunch** to the display of all users. Check whether everybody on your system got the message.

» $ **wall**

» **Out to lunch**

» **<Ctrl-D>**

» **<Ctrl-Alt-F1>**

» **<Alt-F2>**

» **<Alt-F7>**

Keyboard and Mouse Tips

__ 19. The bash shell has a command history function. View some of the commands you have entered. Try to alter one of these commands, then run the command again.

» **<arrow up>**

» **<arrow down>**

___ 20. Your terminal has a buffer that keeps track of the output of your commands. View the output of the previous commands.

» **<shift PgUp>**

» **<shift PgDn>**

___ 21. Bash supports command and filename completion with the TAB character. Try to use this feature, both on commands and on filenames.

» $ **pass<Tab>**

» $ **cat /etc/pass<Tab>**

___ 22. Both in a text terminal and an emulated terminal in the graphical desktop, try to re-execute commands by scrolling up a little, selecting the command with the left mouse button, and then pasting it onto the same terminal again with the middle mouse button.

Also try this across different text and graphical terminals.

Note

SuSE does not enable **gpm** by default; so your mouse won't work in a text terminal when you are using SuSE.

Using the history

___ 23. Use the history command to view the last 20 commands you typed.

» $ **history 20**

___ 24. Execute one of the commands from the history list.

» $ **!2**

___ 25. Execute the **echo** command again, this time changing the word *lunch* to *dinner*.

» $ **!echo:s/lunch/dinner/**

___ 26. Bash also supports searching in the history. Try this feature as well.

» $ **<Ctrl-R>cle**

Locking terminals

Note

Not all distributions install **vlock** and **xlock** by default. If **vlock** and **xlock** are not installed, then you learn how to do that in Exercise 15 - Basic System Configuration.

__ 27. Lock a virtual terminal. Can you switch to another virtual terminal while this one is locked? Unlock the terminal.

» **<Ctrl-Alt-F1>**

» $ **vlock**

» **<Alt-F2>**

» **<Alt-F1>**

» Type your password or the root password **ibmlnx** to unlock the terminal.

__ 28. Lock the console. Can you switch to another virtual terminal now? Unlock the console.

» $ **vlock -a**

» **<Alt-F2>**

» Type your password or the root password **ibmlnx** to unlock the console.

__ 29. Lock the graphical environment and then unlock it again.

» **<Ctrl-Alt-F7>**

» $ **xlock**

- OR -
Click the padlock icon.
- OR -
Use the Lock Screen function in your **Start** menu.

» Type your password to unlock the graphical environment.

Logging off

__ 30. Log off all users that are logged in at any TTY.

» **<Ctrl-Alt-F1>**

» $ **exit**

» **<Alt-F2>**

» $ **logout**

» **<Alt-F7>**

» Click the **GNOME** or **KDE** button and select **Log out**

End of exercise

Exercise 4. Working with Files and Directories

What this exercise is about

This exercise provides the students with the opportunity to begin working with directories and the files they contain.

What you should be able to do

At the end of the lab, you should be able to:

- Work with directories
- Work with files
- Work with files and directories recursively
- Work with binary files

Exercise Instructions

Working with directories

__ 1. If you are not logged in as yourself at **tty7**, log in now. Make sure a terminal window is open.

__ 2. Check the directory you are placed in. What directory is this? __________

__ 3. Change your current directory to the root directory (/)**.**

__ 4. Verify that you are in the root directory and then execute both a simple and a long listing of the files in that directory.

__ 5. List all files in the current directory and list all files in the current directory and below.

Note

This command provides extensive output. When you have seen enough, end the command with the correct **<Ctrl>** sequence.

__ 6. Return to your home directory and list its contents including hidden files.

__ 7. Create a directory in your home directory called **mydir**. Then, issue the command to view a long listing of your home directory and the ~/**mydir** directory. (Do not show the contents of the directories.) What is the size of each directory? __________

__ 8. Change to the **mydir** directory. Create two zero-length files called **myfile1** and **myfile2**.

__ 9. Issue the command to view a long listing of the contents of the **mydir** directory. What are the sizes of **myfile1** and **myfile2**?_______________

__ 10. Return to your home directory and use the **ls -R** command to view your directory tree.

__ 11. Try to remove the **mydir** directory. Does it work?

__ 12. Go to the mydir directory once more and delete the two files in that directory. Then go back up to your home directory and delete the mydir directory.

Working with files

__ 13. Look at the contents of the /**etc**/**passwd** file. The /**etc**/**passwd** file contains a list of all the users authorized to use the system.

__ 14. Copy the /etc/passwd file to your home directory, and rename it to **usersfile**.

__ 15. Split the usersfile into a number of smaller files, of 200 bytes each.

__ 16. Make a long listing of all files in your home directory.

Working with files and directories recursively

__ 17. Create a directory sub1 and create a directory sub2 in sub1. Do this all with one command.

__ 18. Go to the sub2 directory and create a file called myfile.

__ 19. Go back to your home directory. Then make a copy of the whole sub1 directory tree by the name of tree1. Make a recursive listing of all files and directories in sub1 and tree1.

__ 20. You now have two directory trees, named sub1 and tree1. Move the directory tree tree1 into the sub1 subdirectory.

__ 21. List the contents of your home directory. Make a recursive listing of all files and directories in the sub1 directory.

Working with binary files

__ 22. List the content of the file /bin/ls using **od** or **hexdump**.

__ 23. List all strings in the /bin/ls program.

End of exercise

Exercise Instructions with Hints

Working with directories

__ 1. If you are not logged in as yourself at **tty7**, log in now. Make sure you've got a terminal window open.

» **<Ctrl-Alt-F7>**

» Login: **<username>**

» Password: **<password>**

» Open a terminal window.

__ 2. Check the directory you are placed in. What directory is this? ___________

» $ **pwd**

__ 3. Change your current directory to the root directory (/)**.**

» $ **cd** /

__ 4. Verify that you are in the root directory and then execute both a simple and a long listing of the files in that directory.

» $ **pwd**

» $ **ls**

» $ **ls -l**

__ 5. List all files in the current directory and list all files in the current directory and below.

Note

This command provides extensive output. Once you have seen enough, end the command with the correct **<Ctrl>** sequence.

» $ **ls -a**

» $ **ls -R**

<Ctrl-C>

__ 6. Return to your home directory and list its contents including hidden files.

» $ **cd**

- OR -

$ **cd** ~

» $ **ls -a**

__ 7. Create a directory in your home directory called mydir. Then, issue the command to view a long listing of your home directory and the ~/mydir directory. (Do not show the contents of the directories.) What is the size of each directory? ___________

» $ **mkdir mydir**

» $ **ls -ld .**

» $ **ls -ld mydir**

- OR -

$ **ls -ld . mydir**

__ 8. Change to the **mydir** directory. Create two zero-length files called **myfile1** and **myfile2**.

» $ **cd mydir**

» $ **touch myfile1**

» $ **touch myfile2**

- OR -

$ **touch myfile1 myfile2**

__ 9. Issue the command to view a long listing of the contents of the **mydir** directory. What are the sizes of **myfile1** and **myfile2**?_______________

» $ **ls -l**

__ 10. Return to your home directory and use the **ls -R** command to view your directory tree.

» $ **cd**

» $ **ls -R**

__ 11. Try to remove the **mydir** directory. Does it work?

» $ **rmdir mydir**

__ 12. Go to the mydir directory once more and delete the two files in that directory. Then go back up to your home directory and delete the mydir directory.

» $ **cd mydir**

» $ **rm myfile1 myfile2**

» $ **cd ..**

» $ **rmdir mydir**

Working with files

__ 13. Look at the contents of the **/etc/passwd** file. The **/etc/passwd** file contains a list of all the users authorized to use the system.

» $ **cat /etc/passwd**

- OR -

You can use **more** or **less** in place of the **cat** command.

__ 14. Copy the /etc/passwd file to your home directory, and rename it to **usersfile**.

» $ **cp /etc/passwd ~/usersfile**

» $ **mv passwd usersfile**

__ 15. Split the usersfile into a number of smaller files, of 200 bytes each.

» $ **split -b 200 usersfile usersfile.**

__ 16. Make a long listing of all files in your home directory.

» $ **ls -l ~**

Working with files and directories recursively

__ 17. Create a directory sub1 and create a directory sub2 in sub1. Do this all with one command.

» $ **mkdir -p sub1/sub2**

__ 18. Go to the sub2 directory and create a file called myfile.

» $ **cd sub1/sub2**

» $ **touch myfile**

__ 19. Go back to your home directory. Then make a copy of the whole sub1 directory tree by the name of tree1. Make a recursive listing of all files and directories in sub1 and tree1.

» $ **cd**

» $ **cp -R sub1 tree1**

» $ **ls -l**

» $ **ls -R sub1**

» $ **ls -R tree1**

__ 20. You now have two directory trees, named sub1 and tree1. Move the directory tree tree1 into the sub1 subdirectory.

» $ **mv tree1 sub1**

__ 21. List the contents of your home directory. Make a recursive listing of all files and directories in the sub1 directory.

» $ **ls -l**

» $ **ls -R sub1**

Working with binary files

__ 22. List the content of the file /bin/ls using **od** or **hexdump**.

» $ **od /bin/ls**

- OR -

$ **hexdump /bin/ls**

__ 23. List all strings in the /bin/ls program.

» $ **strings /bin/ls**

End of exercise

Exercise 5. File and Directory Permissions

What this exercise is about

This exercise provides the student the opportunity to work with file and directory permissions.

What you should be able to do

At the end of the lab, you should be able to apply file and directory permissions.

Exercise Instructions

Creating User Accounts

To demonstrate permissions in full, you need to create a few additional users, tux1 and tux2, who both are members of the penguins group. For this, you need to execute a few command that have not been covered in the course, and which normally do not need to be executed by a regular user. They are covered in full in LX03.

__ 1. On tty3, log in as root.

__ 2. Execute the following series of commands:

```
# groupadd penguins
# useradd -m -g penguins -c “Tux the Penguin (1)” tux1
# useradd -m -g penguins -c “Tux the Penguin (2)” tux2
# passwd tux1
New password: penguin1
Retype new password: penguin1
# passwd tux2
New password: penguin2
Retype new password: penguin2
```

__ 3. On tty1, log in as **tux1** with password **penguin1**, and on tty2, log in as **tux2** with password **penguin2**.

File and directory permissions

__ 4. Switch to VT 1, where you are logged in as tux1, and look at the permissions on your home directory.

__ 5. Switch to VT2, where you are logged in as tux2. Try to change to the home directory of tux1, or read the contents of the home directory of tux1. Does this work?

On a Fedora or Red Hat system, both commands fail, because the default permissions on a user’s home directory are set to **rwx------**. On a SuSE system, both command succeed, because the default permissions are set to **rwxr-xr-x**.

__ 6. **Fedora**/**Red Hat only:** Switch to tty1. Change the permissions on the home directory of tux1 so that other users are allowed to read and access it. Then try to access the directory again as tux2. Does this work now?

__ 7. As tux2, try to create and delete files in tux1s home directory. Does this work?

__ 8. Switch once again to tty1. Create a bin directory (**Fedora**/**Red Hat only**) and copy the file /bin/ls in there, renaming it to my_ls in the process.

__ 9. Set the permissions on my_ls to **rw-r-----**, and then try to execute it, both as tux1 and tux2. Does this work? Why not?

__ 10. Now set the permissions to **rwxr-xr-x**, and then try to execute it once more, both as tux1 and tux2. Does this work now?

__ 11. Try to execute my_ls as tux1 and as tux2, and as yourself, but now with permissions **rw-------**, **rw-rw----**, **rwx------**, **rwx--x---** and **rwx--x--x** as well. What permissions are required, at a minimum, for tux1 to execute my_ls? What permissions are required for tux2? What permissions does your own user account require?

End of exercise

Exercise Instructions with Hints

Creating User Accounts

To demonstrate permissions in full, you need to create a few additional users, tux1 and tux2, who both are members of the penguins group. For this, you need to execute a few command that have not been covered in the course, and which normally do not need to be executed by a regular user. They are covered in full in LX03.

__ 1. On tty3, log in as root.

» **<Ctrl+Alt+F3>**

» Login: **root**

» Password: **ibmlnx**

__ 2. Execute the following series of commands:

```
# groupadd penguins
# useradd -m -g penguins -c “Tux the Penguin (1)” tux1
# useradd -m -g penguins -c “Tux the Penguin (2)” tux2
# passwd tux1
New password: penguin1
Retype new password: penguin1
# passwd tux2
New password: penguin2
Retype new password: penguin2
```

__ 3. On tty1, log in as **tux1** with password **penguin1**, and on tty2, log in as **tux2** with password **penguin2**.

» **<Alt-F1>**

» Login: **tux1**

» Password: **penguin1**

» **<Alt-F2>**

» Login: **tux2**

» Password: **penguin2**

File and directory permissions

__ 4. Switch to VT 1, where you are logged in as tux1, and look at the permissions on your home directory.

» **<Alt-F1>**

» $ **ls -ld /home/tux1**

__ 5. Switch to VT2, where you are logged in as tux2. Try to change to the home directory of tux1, or read the contents of the home directory of tux1. Does this work?

» **<Alt+F2>**

» $ **cd /home/tux1**

» $ **ls /home/tux1**

On a Fedora or Red Hat system, both commands fail, because the default permissions on a user's home directory are set to **rwx------**. On a SuSE system, both commands succeed, because the default permissions are set to **rwxr-xr-x**.

__ 6. **Fedora/Red Hat only:** Switch to tty1. Change the permissions on the home directory of tux1 so that other users are allowed to read and access it. Then try to access the directory again as tux2. Does this work now?

» **<Alt+F1>**

» $ **chmod 755 /home/tux1**

- OR -

$ **chmod go+rx /home/tux1**

» **<Alt-F2>**

» $ **cd /home/tux1**

» $ **ls /home/tux1**

__ 7. As tux2, try to create and delete files in tux1s home directory. Does this work?

» $ **touch testfile**

__ 8. Switch once again to tty1. Create a bin directory (**Fedora/Red Hat only**) and copy the file /bin/ls in there, renaming it to my_ls in the process.

» **<Alt+F1>**

» $ **mkdir /home/tux1/bin** (**Fedora/Red Hat only**)

» $ **cp /bin/ls /home/tux1/bin/my_ls**

__ 9. Set the permissions on my_ls to **rw-r-----**, and then try to execute it, both as tux1 and tux2. Does this work? Why not?

» $ **chmod 640 /home/tux1/bin/my_ls**

- OR -

$ **chmod u=rw,g=r,o=/home/tux1/bin/my_ls**

» $ **my_ls**

» **<Alt+F2>**

» $ **/home/tux1/bin/my_ls**

» **<Alt+F1>**

__ 10. Now set the permissions to **rwxr-xr-x**, then try to execute it once more, both as tux1 and tux2. Does this work now?

- » $ **chmod 755 /home/tux1/bin/my_ls**

  - OR -

  $ **chmod u=rwx,go=rx /home/tux1/bin/my_ls**

- » $ **my_ls**
- » **<Alt+F2>**
- » $ **/home/tux1/bin/my_ls**
- » **<Alt+F1>**

__ 11. Try to execute my_ls as tux1 and as tux2, and as yourself, but now with permissions **rw-------**, **rw-rw----**, **rwx------**, **rwx--x---** and **rwx--x--x** as well. What permissions are required, at a minimum, for tux1 to execute my_ls? What permissions are required for tux2? What permissions does your own user account require?

End of exercise

Exercise 6. Linux Documentation

What this exercise is about

The purpose of this exercise is to give the students the opportunity to explore and experiment with the **man** and **info** commands. Students also read the FAQ and HOWTO documentation.

What you should be able to do

At the end of the lab, you should be able to:

- Use the **man** command
- Use the **info** command
- Locate and use other Linux documentation

Exercise Instructions

Man Pages

__ 1. If you are not already logged on, log in as **tux1** at **tty1**.

__ 2. Display the man pages for the **man** command. Read the text that follows to obtain a better understanding of the functionality of the **man** command.

__ 3. Search for the string **PAGER** in the manual page of the man command.

__ 4. Use the <Q> key to end the **man** command.

__ 5. Display the man page of the **ls** command. Move though the manual pages:

- Go to the last page.
- Go to the previous page.
- Go to the first page.

Type these commands while looking at the man page of **ls**.

__ 6. Close the **man** command.

__ 7. Find out which manual pages all deal with **passwd**. Then view each page, giving the correct section number.

Note

If the **man -k** or **apropos** commands do not work, then you need to run the **makewhatis** command as root. Normally, the **makewhatis** command is executed each night automatically, but because your system is freshly installed, this might not have happened yet.

Info command

__ 8. View the info documentation for the **finger** command. Are you actually reading info documentation now?

__ 9. Move through this page by using the Space and Backspace keys.

__ 10. Read the help for the **info** command. Use the L key to go back to the finger information.

__ 11. End the **info** command.

__ 12. Read the info documentation of the **info** command. Use the menu by using the Tab and M keys.

__ 13. **info** has a nice built-in tutorial. If you have spare time during this course, look at the tutorial to see some of the advanced features of **info**.

Other Documentation

__ 14. Make a listing of all directories in the /usr/share/doc directory. Browse some of these directories to see what sort of information is available.

__ 15. If the classroom systems have an Internet connection, then take a look at the http://www.tldp.org Web site. This is the main documentation Web site for Linux.

Note that in some classrooms, some additional configuration of your Web browser might be needed because the classroom is behind a socks or proxy-based firewall. In this case, your instructor gives you additional instructions.

End of exercise

Exercise Instructions with Hints

Man Pages

__ 1. If you are not already logged on, log in as **tux1** at **tty1**.

» **<Ctrl+Alt+F1>**

» Login: **tux1**

» Password: **penguin1** (the password does not appear on the screen)

__ 2. Display the man pages for the **man** command. Read the text that follows to obtain a better understanding of the functionality of the **man** command.

» $ **man man**

__ 3. Search for the string **PAGER** in the manual page of the man command.

» **/PAGER**

__ 4. Use the Q key to end the **man** command.

» **q**

__ 5. Display the man page of the **ls** command. Move though the manual pages:

- Go to the last page.
- Go to the previous page.
- Go to the first page.

Type these commands while looking at the man page of **ls**.

» $ **man ls**

» Go to last page: **G**

» Go to previous page: **b**

» Go to first page: **1G**

__ 6. Close the **man** command.

» **q**

__ 7. Find out which manual pages all deal with **passwd**. Then view each page, giving the correct section number.

Note

If the **man -k** or **apropos** commands do not work, then you need to run the **makewhatis** command as root. Normally, the **makewhatis** command is executed each night automatically, but because your system is freshly installed, this might not have happened yet.

- » $ **man -k passwd** or **apropos passwd**
- » $ **man 1 passwd**
- » $ **man 5 passwd**

Info command

__ 8. View the info documentation for the **finger** command. Are you actually reading info documentation now?

- » $ **info finger**
- » No. Look at the upper left corner of your screen. It says *manpages*, which means that there is no info documentation for finger. If **info** cannot locate the correct info document, it locates and displays its manual page. If there is no manual page, info shows the top node.

__ 9. Move through this page by using the Space and Backspace keys.

- » **<Space>** shows the next page of information
- » **<Backspace>** show the previous page

__ 10. Read the help for the **info** command. Use the **<L>** key to go back to the finger information.

- » To enter help, type **?**
- » To quit the help, type **l**.

__ 11. End the **info** command.

- » **q**

__ 12. Read the info documentation of the **info** command. Use the menu by using the Tab and M keys.

- » **info info**
- » **<Tab>**
- » **m**
- » **<Enter>**
- » **q**

__ 13. **info** has a nice built-in tutorial. If you have spare time during this course, look at the tutorial to see some of the advanced features of **info**.

- » Start the tutorial with the **info** command.
- » $ **info**
- » **q**

Other Documentation

__ 14. Make a listing of all directories in the /usr/share/doc directory. Browse some of these directories to see what sort of information is available.

» $ **cd /usr/share/doc**

» $ **ls**

» Browse some directories and see what documentation is available.

» $ **cd**

__ 15. If the classroom systems have an Internet connection, then take a look at the http://www.tldp.org Web site. This is the main documentation Web site for Linux.

Note that in some classrooms some additional configuration of your Web browser might be needed because the classroom is behind a socks or proxy-based firewall. In this case, your instructor gives you additional instructions.

End of exercise

Exercise 7. Editing Files

What this exercise is about

The purpose of this exercise is to give the students the opportunity to create and edit files using the most common UNIX editor, **vi**, and to try out a number of other editors that might be available.

What you should be able to do

At the end of the lab, you should be able to:

- Use **vi** to create and edit files
- List a few other editors that are available on your system

Exercise Instructions

Working with vi

__ 1. If you aren't already logged in as **tux1** at **tty1**, log in now.

__ 2. Ensure that you are in your home directory. Create a file in your home directory named **vitest** using **vi**.

Type the following text and the marine alphabet into the **vitest** file. Adding the alphabet is an easy way to fill a couple of screens of information needed for later use. This is a training session about the usage of the **vi** editor. We need some more lines to learn the most common commands of the editor.

```
a alpha
b bravo
c charlie
…
(the rest of the marine alphabet)
x x-ray
y yankee
z zulu
```

__ 3. Return to command mode. Write and quit the file. Notice that as soon as you press the colon (**:**), it appears below the last line of your input area. When the buffer is empty and the file is closed, you see a message giving the number of lines and characters in the file.

Cursor Movement Keys

__ 4. Open **vitest** file again. Notice that the bottom line of the screen indicates the name of the file and number of characters.

__ 5. Using the H,J, K, and L keys, practice moving the through the file.

__ 6. Use the appropriate **vi** commands to move through the text:

- Move forward one page.
- Move back one page.
- Move the cursor to the first line on the screen.
- Move the cursor to the last line in the file.
- Move the cursor to the first line in the file.
- Move the cursor to line 5 of the file.
- Move the cursor to the end of the line.
- Move the cursor to the beginning of the line.

__ 7. Change the file vitest so that after each letter of the alphabet a common first name is added that starts with that letter. Make sure you use different methods for switching from command mode to insert mode.

__ 8. Practice some more with all the commands that are listed on your cheat sheet.

__ 9. Save the file but do not exit **vi**.

Using set to Customize the Editing Session

__ 10. Turn online numbering and set your tab stop to 4.

Global search and replace

__ 11. Replace all spaces in your file with tabs.

__ 12. Save your file.

Working with other editors

__ 13. Your system has various other text mode and graphical editors available as well. Start some of these to get acquainted with them.

Note

All editors listed in the course material might not be available or installed on your distribution.

End of exercise

Exercise Instructions with Hints

Working with vi

__ 1. If you aren't already logged in as **tux1** at **tty1**, log in now.

» **<Ctrl+Alt+F1>**

» Login: **tux1**

» Password: **penguin1**

__ 2. Ensure that you are in your home directory. Create a file in your home directory named **vitest** using **vi**.

Type the following text and the marine alphabet into the **vitest** file. Adding the alphabet is an easy way to fill a couple of screens of information needed for later use. This is a training session about the usage of the **vi** editor. We need some more lines to learn the most common commands of the editor.

```
a alpha
b bravo
c charlie
...
(the rest of the marine alphabet)
x x-ray
y yankee
z zulu
```

» $ **cd**

» $ **pwd**

» $ **vi vitest**

» First type an **I** to enter input mode. Remember that **vi** starts in command mode.

» Then type the contents of the file.

__ 3. Return to command mode. Write and quit the file. Notice that as soon as you press the colon (**:**), it appears below the last line of your input area. When the buffer is empty and the file is closed, you see a message giving the number of lines and characters in the file.

» Use the Esc key to go from input mode to command mode.

» Saving the file and closing **vi** can be done with one of these commands: **:wq** or **:x** or **ZZ**

Cursor Movement Keys

__ 4. Open **vitest** file again. Notice that the bottom line of the screen indicates the name of the file and number of characters.

» $ **vi vitest**

__ 5. Using the H, J, K, and L keys, practice moving the through the file.

» **j** down one line

» **k** up one line

» **h** left one character

» **l** right one character

__ 6. Use the appropriate **vi** commands to move through the text:

- Move forward one page.
- Move back one page.
- Move the cursor to the first line on the screen.
- Move the cursor to the last line in the file.
- Move the cursor to the first line in the file.
- Move the cursor to line 5 of the file.
- Move the cursor to the end of the line.
- Move the cursor to the beginning of the line.

» **Ctrl+F** - Move forward one page.

» **Ctrl+B** - Move back one page.

» **H** - Move the cursor to the first line on the screen.

» **G** - Move cursor to last line in the file.

» **1G** or **:1 and Enter** - Move cursor to first line in file.

» **5G** or **:5 and Enter** - Move cursor to line 5.

» **$** - Move cursor to end of line.

» **0** (zero) or ***n*** - Move cursor to beginning of line.

__ 7. Change the file vitest so that after each letter of the alphabet a common first name is added that starts with that letter. Make sure you use different methods for switching from command mode to insert mode.

» Use **vi** commands to add the words. Be sure to try the **i, I, a,** and **A** commands. The file should look like this afterwards:

```
a alpha Arnold
b bravo Brad
c charlie Charles
...
```

__ 8. Practice some more with all the commands that are listed on your cheat sheet.

__ 9. Save the file but do not exit **vi**.

» **:w <Enter>**

Using set to Customize the Editing Session

__ 10. Turn online numbering and set your tab stop to 4.

» **:set number**

» **:set tabstop=4**

Global search and replace

__ 11. Replace all spaces in your file with tabs.

» **:%s/ /<Tab>/g**

(where <*Tab*> is the Tab key. This shows up as **^I** when you type it.)

Hint

colon percent s slash space slash <Tab> slash g <Enter>)

__ 12. Save your file.

» **:wq**

- OR -
:x
- OR -
ZZ

Working with other editors

__ 13. Your system has various other text mode and graphical editors available as well. Start some of these to get acquainted with them.

Note

All editors listed in the course material might not be available or installed on your distribution.

» $ **pico vitest**

» $ **mcedit vitest**

» $ **hexedit vitest**

» **<Ctrl-Alt-F7>**

» $ **gedit vitest**

» $ **kedit vitest**

End of exercise

Exercise 8. Shell Basics

What this exercise is about

This exercise provides an opportunity to get to know the basic features of the Linux shell (bash).

What you should be able to do

At the end of the lab, you should be able to:

- Use wildcards for file name expansion
- Redirect standard in, standard out, and standard error
- Use pipes to provide the output of one process as input to another process
- Perform command grouping and line continuation

Exercise Instructions

Wildcards

__ 1. If you are not logged in as **tux1** at **tty1**, log in now.

__ 2. Go to the /**etc** directory and make a list of all files here.

__ 3. Use **ls** with wildcards to list file names:

- That end with **conf**
- That begin with a **d** or **D**
- That contain an **o** in the fifth position
- That contain the word **tab** (in any combination with capitals and lowercase characters)
- That end with a number
- That do not end with a number

(Note that wildcard expansion is done by the shell. If one of the filenames that matches is a directory name, then **ls** by default lists the contents of that directory, instead of the filename itself. To prevent this, use the **-d** option.)

__ 4. What happens if you execute the command **ls -d ?[!y]*[e-g]**? What would the shortest filename be that can match? Execute this command to verify your answer.

__ 5. Return to your home directory.

Redirection

__ 6. Use the **cat** command and redirection to create a file called **junk** containing a few lines of text. When you have typed a few lines, end your input to the **cat** command and return to the shell prompt. Then view the contents of the file you just created.

__ 7. Append more lines to the **junk** file using redirection. Then view the contents of the file **junk** and check if all the lines you saved in this file are there.

Pipes, Tees, and Filters

__ 8. Count the number of files in your current directory. Use a pipe, do not count the files manually.

__ 9. Does **ls > tempfile ; wc -l tempfile ; rm tempfile** do the same thing as the pipe you made in the previous command? Why or why not?

__ 10. Use the **ls** command and save the output in a file called **tempfile2** before you count the files.

__ 11. Use the **sed** command to alter the output of the **ls -l** /**etc**/ command so that it looks like you own all files in /etc. Execute this both with and without the *global* option. What is the difference?

__ 12. Use the **awk** command to display the first and ninth column of the output of the **ls -l /etc**/ command.

__ 13. Use the **tac** command to display the output of the **ls** command in reverse order.

__ 14. Use the **nl** command to number the lines of **tempfile**.

__ 15. Use the **pr** command to format **tempfile** for the printer.

__ 16. Combine all usersfile parts from exercise 4 into one big file, called usersfile5. Check to see if this file is identical to the original usersfile.

Command Grouping

__ 17. On the same command line, display the current system date and all the users that are logged in, together with some explaining comments, and save all this to one file after numbering the lines. Check your output.

Process Environment

__ 18. Display all your variables that are defined in your current process environment. Also display all variables that are currently exported.

__ 19. Create a variable **x** and set its value to **10**. Check the value of the variable. Again, display all your current variables.

__ 20. Create a subshell. Check to see what value variable **x** holds in the subshell. What is the value of **x**? ________ List the subshell's current variables. Do you see a listing for **x**? ________

__ 21. Set the value of **x** to **500** and go back to your parent process. What is the current value of **x**? ________ Why?________

__ 22. Make sure that child processes inherit the variable **x**. Verify this by creating a subshell and checking the value of variable **x**. After this, exit your subshell.

End of exercise

Exercise Instructions With Hints

Wildcards

__ 1. If you are not logged in as **tux1** at **tty1**, log in now.

» **<Ctrl-Alt-F1>**

» Login: **tux1**

» Password: **penguin1** (the password does not appear on the screen)

__ 2. Go to the /**etc** directory and make a list of all files here.

» $ **cd /etc**

» $ **ls**

__ 3. Use **ls** with wildcards to list file names:

- that end with **conf**
- that begin with a **d** or **D**
- that contain an **o** in the fifth position
- that contain the word **tab** (in any combination with capitals and lowercase characters)
- that end with a number
- that do not end with a number

(Note that wildcard expansion is done by the shell. If one of the filenames that matches is a directory name, then **ls** by default lists the contents of that directory, instead of the filename itself. To prevent this, use the **-d** option.)

» $ **ls -d *conf**

» $ **ls -d [dD]***

» $ **ls -d ????o***

» $ **ls -d *[tT][aA][bB]***

» $ **ls -d *[0-9]**

» $ **ls -d *[!0-9]**

__ 4. What happens if you execute the command **ls -d ?[!y]*[e-g]**? What would the shortest filename be that can match? Execute this command to verify your answer.

» $ **ls -d ?[!y]*[e-g]**

__ 5. Return to your home directory.

» $ **cd**

Redirection

__ 6. Use the **cat** command and redirection to create a file called **junk** containing a few lines of text. When you have typed a few lines, end your input to the **cat** command and return to the shell prompt. Then view the contents of the file you just created.

» $ **cat > junk**

» Type some lines of information

» **<Ctrl-D>** (At the beginning of a new line)

» $ **cat junk**

__ 7. Append more lines to the **junk** file using redirection. Then view the contents of the file **junk** and check if all the lines you saved in this file are there.

» $ **cat >> junk**

» Type some lines of information

» **<Ctrl+D>** (At the beginning of a new line)

» $ **cat junk**

Pipes, Tees, and Filters

__ 8. Count the number of files in your current directory. Use a pipe, do not count the files manually.

» $ **ls | wc -l**

__ 9. Does **ls > tempfile ; wc -l tempfile ; rm tempfile** do the same thing as the pipe you made in the previous command? Why or why not?

» Almost, but it counts tempfile too; so it counts one file too many.

» $ **ls > tempfile**

» $ **more tempfile**

__ 10. Use the **ls** command and save the output in a file called **tempfile2** before you count the files.

» $ **ls | tee tempfile2 | wc -l**

__ 11. Use the **sed** command to alter the output of the **ls -l /etc**/ command so that it looks like you own all files in /etc. Execute this both with and without the *global* option. What is the difference?

» $ **ls -l /etc | sed s/root/tux1/**

» $ **ls -l /etc | sed s/root/tux1/g**

__ 12. Use the **awk** command to display the first and ninth column of the output of the **ls -l /etc/** command.

```
» $ ls -l /etc | awk '{print $1 " " $9}'
```

Note

The $9 needs to be $8 in SLES.

__ 13. Use the **tac** command to display the output of the **ls** command in reverse order.

```
» $ ls | tac
```

__ 14. Use the **nl** command to number the lines of **tempfile**.

```
» $ nl tempfile
```

__ 15. Use the **pr** command to format **tempfile** for the printer.

```
» $ pr tempfile
```

__ 16. Combine all usersfile parts from Exercise 4 into one big file, called usersfile5. Check to see if this file is identical to the original usersfile.

```
» $ su - <username>                - where <username> is your username
» $ cat usersfile* > usersfile5
» $ diff usersfile usersfile5
```

Command Grouping

__ 17. On the same command line, display the current system date and all the users that are logged in, together with some explaining comments, and save all this to one file after numbering the lines. Check your output.

```
» $ ( date ; who ) | nl > users
» $ cat users
```

Process Environment

__ 18. Display all your variables that are defined in your current process environment. Also display all variables that are currently exported.

```
» $ set | less
» $ env | less
```

__ 19. Create a variable **x** and set its value to **10**. Check the value of the variable. Again, display all your current variables.

```
» $ x=10
```

» $ **echo $x**

» $ **set | less**

» $ **env | less**

__ 20. Create a subshell. Check to see what value variable **x** holds in the subshell. What is the value of **x**? _______ List the subshell's current variables. Do you see a listing for **x**? _______

» $ **bash**

» $ **echo $x**

» You should see no output, only an empty line.

» $ **set | less**

» You should not see a listing for **x**.

__ 21. Set the value of **x** to **500** and go back to your parent process. What is the current value of **x**? _______ Why?_______

» $ **x=500**

» $ **exit**

» $ **echo $x**

__ 22. Make sure that child processes inherit the variable **x**. Verify this by creating a subshell and checking the value of variable **x**. After this, exit your subshell.

» $ **export x**

» $ **env | less**

» $ **bash**

» $ **echo $x**

» $ **exit**

End of exercise

Exercise 9. Working with Processes

What this exercise is about

This exercise familiarizes the students with process manipulation and process control.

What you should be able to do

At the end of the lab, you should be able to:

- Monitor processes
- Change and understand the process environment
- Control jobs
- Terminate processes

Exercise Instructions

Listing Processes

__ 1. Log in at **tty1** as **tux1**.

__ 2. Check the PID of your log in environment and then create a subshell by entering **bash**. What is the process ID of the subshell? Is it different from your login process?

____

____

__ 3. Enter the command **ls -R** / **>outfile 2>/dev/null &** and then show the processes that you are running in the system. Which processes are running?

Note

This command is explained in full in the next units.

____

____

__ 4. While the **ls** command is still running, run the **pstree** command. (It might be necessary to restart the **ls** command.)

__ 5. Log in as **tux2** on **tty2** and run **vi tux2_file**.

__ 6. Go back to **tty1** and show all the processes in your system. If necessary, look in the man pages and info to find the correct options to show all processes running in your system.

Look for your own processes as well as the processes of **tux2**.

__ 7. Again run the **ls -R** / **>outfile 2>/dev/null &** command and then **exit** your current process. List the processes you are running. What happens to processes if you kill their parent process?

____

Job Control

__ 8. Using **vi** or another editor, create the file named **myclock** in your bin directory with the following contents:

```
while true
do
date
sleep 10
```

```
done
```

Make the script executable.

__ 9. Run the script **myclock**. Run this script in the foreground.

__ 10. Suspend the job you just started.

__ 11. List all the jobs that you are running on the system and restart the above job in the background.

__ 12. List all users that are logged in. Bring the job back to the foreground, wait until you get a timestamp, and then exit the job.

Terminating a Process

__ 13. Execute the **myclock** script again, this time in the background. **Hint**: Take note of the PID.

__ 14. List all your processes and kill the **sleep** process. What happened?

__ 15. Now stop the shell script **myclock**.

End of exercise

Exercise Instructions with Hints

Listing Processes

__ 1. Log in at **tty1** as **tux1**.

» **<Ctrl+Alt+F1>**

» Login: **tux1**

» Password: **penguin1** (The password does not appear on screen)

__ 2. Check the PID of your log in environment and then create a subshell by entering **bash**. What is the process ID of the subshell? Is it different from your login process?

____

____

» $ **echo $$**

» $ **bash**

» $ **echo $$**

» Yes, all processes in your system have a unique process ID (PID). So the PID of your login shell and your subshell have to be different. If they are equal you really have a problem ;-).

» $ **exit**

__ 3. Enter the command **ls -R** / **>outfile 2>/dev/null &** and then show the processes that you are running in the system. Which processes are running?

Note

This command is explained in full in the next units.

____

____

» $ **ls -R** / **> outfile 2>/dev/null &**

» $ **ps**

- OR -

$ **ps -ef** (for more information about your processes)

__ 4. While the **ls** command is still running, run the **pstree** command. (It might be necessary to restart the **ls** command.)

» $ **pstree**

__ 5. Log in as **tux2** on **tty2** and run **vi tux2_file**.

» **<Alt+F2>**

» Login: **tux2**

» Password: **penguin2** (the password does not appear on the screen)

» $ **vi tux2_file**

__ 6. Go back to **tty1** and show all the processes in your system. If necessary, look in the man pages and info to find the correct options to show all processes running in your system.

Look for your own processes as well as the processes of **tux2**.

» **<Alt+F1>**

» $ **ps -ef | less**

__ 7. Again run the **ls -R** / **>outfile 2>/dev/null &** command and then **exit** your current process. List the processes you are running. What happens to processes if you kill their parent process?

____

» $ **ls -R** / **>outfile 2>/dev/null &**

» $ **exit**

» Login: **tux1**

» Password: **penguin1**

» $ **ps -ef**

» $ **pstree**

» If the parent process dies, the child processes are transferred to a new parent process, **init**.

Job Control

__ 8. Using **vi** or another editor, create the file named **myclock** in your bin directory with the following contents:

```
while true
do
date
sleep 10
done
```

Make the script executable.

» $ **cd** ~/**bin**

» $ **vi myclock**

» $ **chmod +x myclock**

__ 9. Run the script **myclock**. Run this script in the foreground.

» $ **myclock**

__ 10. Suspend the job you just started.

» **<Ctrl+Z>**

__ 11. List all the jobs that you are running on the system and restart the above job in the background.

» $ **jobs**

» $ **bg %1**

__ 12. List all users that are logged in. Bring the job back to the foreground, wait until you get a timestamp, and then exit the job.

» $ **who**

» $ **fg %1**

» **<Ctrl+C>**

Terminating a Process

__ 13. Execute the **myclock** script again, this time in the background. **Hint**: Take note of the PID.

» $ **myclock &**

__ 14. List all your processes and kill the **sleep** process. What happened?

» $ **ps**

» $ **kill <PID>**

where <*PID*> is the process ID of the sleep command.

» You received a new timestamp immediately after the **kill** command ran.

» **myclock** is a shell script that displays a timestamp every 10 seconds. When you kill the **sleep** process, there is no process to wait for. The script continues and shows you another timestamp.

» Killing processes started from a shell script does not kill the shell script itself.

__ 15. Now stop the shell script **myclock**.

» $ **kill <PID>**

where <*PID*> is the process ID of the process that runs the **myclock** script

Hint

Look for a second instance of bash.

- or -

» $ **kill %<JobNo>**

where <*JobNo*> is the Job Number ID of the process that runs the **myclock** script.

Hint

Look for it using the **jobs** command.

End of exercise

Exercise 10.Linux Utilities

What this exercise is about

The purpose of this exercise is to become familiar with some of the many helpful tools available with Linux.

What you should be able to do

At the end of the lab, you should be able to:

- Search for files that meet specific criteria
- List specific columns of a file
- Search text files for lines that match a pattern
- Sort lines in a file
- Display the first or last few lines of a file
- Find out where executables are located
- Compress files and decompress them

Exercise Instructions

Working with find and locate

__ 1. Log in as **tux1** at **tty1**, if you aren't already.

__ 2. Find and display all files and directories in your home directory.

__ 3. Find all files in your system that begin with the string **abc** and have **ls -l** automatically executed on each file name found. Discard all errors.

__ 4. Repeat the previous command but interactively prompt the user to display the long listing on each file. Do not discard errors, because stderr is used to display the prompt.

__ 5. Find all files starting from /**usr** that are owned by the user **root**.

__ 6. Modify the last command to count the number of files on the whole system owned by **root**. Now alter the command so that you don't get error messages on your screen.

__ 7. Find all directories in your system and save this list in the file **all.directories**. The error message can be sent to the bit bucket. Execute this command in the background.

__ 8. **Fedora**/**Red Hat only:** Use the locate command to locate all files that match the string “passwd”.

Note

SuSE does not install the locate command by default. You learn how to do this in Exercise 15 - Basic System Configuration.

Working with cut

__ 9. Display the contents of the /**etc**/**passwd** file.

__ 10. Only show the user name and the home directory of the users listed in /**etc**/**passwd**.

__ 11. Show the name and the members of all groups listed in /**etc**/**group**.

__ 12. List only the type, size, and name of files in the current directory.

Working with grep

__ 13. Find all lines in the /**etc**/**passwd** that begin with the letter **s**.

__ 14. Repeat the search in the previous instruction, but this time display only the number of lines that contain the pattern.

__ 15. Find all processes running on the system, owned by user **tux1** or **tux2**.

Working with sort

__ 16. Display the contents of the **/etc/passwd** file in alphabetical order.

__ 17. Display the contents of **/etc/passwd** again, but now sorted on the home directory field.

Working with head and tail

__ 18. Display the first 10 lines of the **/etc/passwd** file.

__ 19. Display the last 6 lines of the **/etc/passwd** file.

__ 20. The tail command is also handy for stripping out header information from the output of a command. First, list the processes currently running on your system. Notice the headings. Next, display the processes running on your system excluding the header information.

Working with type, which, and whereis

__ 21. Find out where the **passwd** command is stored. Locate the manual pages and source code for this command.

Working with gzip, gunzip, and zcat

__ 22. Create a big file named **big** in your home directory, for instance by capturing the output of the ls -lR / command. What is the size of big?

__ 23. Make the file twice as large.

__ 24. Note the size of **big**.________
Compress the **big** file. What is the new size of the file and what is its new name?
____

__ 25. Look at the contents of the **big.gz** file.

__ 26. Restore the old **big** file. What is the size of **big** and what is its name? _________

End of exercise

Exercise Instructions with Hints

Working with find and locate

__ 1. Log in as **tux1** at **tty1**, if you aren't already.

» **<Alt+F1>**

» Login: **tux1**

» Password: **penguin1**(the password does not appear on the screen)

__ 2. Find and display all files and directories in your home directory.

» $ **find $HOME**

- OR -
$ **cd ; find .**

__ 3. Find all files in your system that begin with the string **abc** and have **ls -l** automatically executed on each file name found. Discard all errors.

» **touch abcdef**

» $ **find** / **-name 'abc*' -exec ls -l {}** \; **2>/dev/null**

__ 4. Repeat the previous command but interactively prompt the user to display the long listing on each file. Do not discard errors, since stderr is used to display the prompt.

» $ **find** / **-name 'abc*' -ok ls -l {}** \;

__ 5. Find all files starting from /**usr** that are owned by the user **root**.

» $ **find /usr -user root**

__ 6. Modify the last command to count the number of files on the whole system owned by **root**. Now alter the command so that you don't get error messages on your screen.

» $ **find** / **-user root | wc -l**

» $ **find** / **-user root 2>/dev/null | wc -l**

__ 7. Find all directories in your system and save this list in the file **all.directories**. The error message can be sent to the bit bucket. Execute this command in the background.

» $ **find** / **-type d >all.directories 2>/dev/null &**

__ 8. **Fedora/Red Hat only:** Use the locate command to locate all files that match the string "passwd".

Note

SuSE does not install the locate command by default. You learn how to do this in Exercise 15 - Basic System Configuration.

```
» $ locate passwd
```

Working with cut

__ 9. Display the contents of the **/etc/passwd** file.

```
» $ cat /etc/passwd
```

__ 10. Only show the user name and the home directory of the users listed in **/etc/passwd**.

```
» $ cut -f1,6 -d: /etc/passwd
```

__ 11. Show the name and the members of all groups listed in **/etc/group**.

```
» $ cut -f1,4 -d: /etc/group
```

__ 12. List only the type, size, and name of files in the current directory.

```
» $ ls -l | cut -c1,31-42,49-
```

Note

Your column numbers might need to be adjusted a little.

Working with grep

__ 13. Find all lines in the **/etc/passwd** that begin with the letter **s**.

```
» $ grep ^s /etc/passwd
```

__ 14. Repeat the search in the previous instruction, but this time display only the number of lines that contain the pattern.

```
» $ grep -c ^s /etc/passwd
```

__ 15. Find all processes running on the system, owned by user **tux1** or **tux2**.

```
» $ ps aux | egrep "tux1|tux2"
```

Working with sort

__ 16. Display the contents of the **/etc/passwd** file in alphabetical order.

```
» $ sort /etc/passwd
```

__ 17. Display the contents of **/etc/passwd** again, but now sorted on the home directory field.

» $ **sort -t: -k5 /etc/passwd**

Working with head and tail

__ 18. Display the first 10 lines of the **/etc/passwd** file.

» $ **head /etc/passwd**

__ 19. Display the last 6 lines of the **/etc/passwd** file.

» $ **tail -6 /etc/passwd**

__ 20. The **tail** command is also handy for stripping out header information from the output of a command. First, list the processes currently running on your system. Notice the headings. Next, display the processes running on your system excluding the header information.

» $ **ps**

» $ **ps | tail -2**

Working with type, which, and whereis

__ 21. Find out where the **passwd** command is stored. Locate the manual pages and source code for this command.

» $ **type passwd**

- OR -

$ **which passwd**

- OR -

$ **whereis passwd**

Working with gzip, gunzip, and zcat

__ 22. Create a big file named **big** in your home directory, for instance by capturing the output of the ls -lR / command. What is the size of big?

» $ **ls -lR / > big 2>&1**

» $ **ls -l big**

__ 23. Make the file twice as large.

» $ **cp big big2**

» $ **cat big2 >> big**

» $ **rm big2**

__ 24. Note the size of **big.**________

Compress the **big** file. What is the new size of the file and what is its new name?

____

» $ **ls -l big**

» $ **gzip big**

» $ **ls -l big***

» The new name is **big.gz**.

__ 25. Look at the contents of the **big.gz** file.

» $ **zcat big**

__ 26. Restore the old **big** file. What is the size of **big** and what is its name? _________

» $ **gunzip big**

» $ **ls -l big***

» The name is **big** again.

End of exercise

Exercise 11. Shell Scripting

What this exercise is about

After you have been using Linux for a while, you find certain characteristics of your environment that you would like to customize along with some tasks that you execute regularly that you would like to automate.

This exercise introduces you to some of the more common constructs used to help you write shell scripts to customize and automate your computing environment.

What you should be able to do

At the end of the lab, you should be able to:

- List common constructs used in writing shell scripts
- Create and execute simple shell scripts

Introduction

You need no programming experience to perform this exercise. Refer to the unit in the Student Notebook for help with the syntax of constructs when creating the shell scripts in this exercise.

Exercise Instructions

Working with Positional Parameters

__ 1. If you are not logged in as **tux1** at **tty1**, log in now.

__ 2. Create a shell script named **parameters** that echoes the five parameters that follow using predefined special variables set by the shell to fill in the blanks. Execute the script using the positional parameters 10 100 1000.

Conditional Execution

__ 3. Using conditional execution, create a shell script named **checkfile** that checks to see if the file named **parameters** exists in your directory. If it exists, use a command to show the contents of the file. Execute the script.

__ 4. Modify the **checkfile** script and change the name of the file from **parameters** to **noname** (check to ensure that you do *not* have a file by this name in your current directory). Also, using conditional execution, if the **cat** command was *not* successful, display the error message, “The file was not found.” Execute the script.

__ 5. Modify the **checkfile** script to accept a single parameter from the command line as input to the **ls** and **cat** commands. Execute the script twice, once using the file named **parameters** and again using the file named **noname**.

__ 6. Execute the **checkfile** script again, but this time with no parameters. What happens? Modify the script so that this does not appear again.

Loops

__ 7. Using the **for** loop, modify the **checkfile** script to accept multiple files as input from the command line instead of just one. If the files are found, display all of them. If the files are not found, display an error message showing all file names that were not found. Look in your directory and note a few valid file names that you can use as input. Execute the script using valid and invalid file names.

__ 8. Now do the same thing, but use a **while** loop in combination with the **shift** command.

Arithmetic

__ 9. From the command line, display the results of multiplying 5 times 6.

__ 10. Now create a shell script named **math** to multiply any two numbers when entered as input from the command line. Execute the script multiplying 5 times 6. Experiment with any other two numbers.

Integration Exercise

__ 11. Use the knowledge you gained in this course to write a script that accepts a directory name as a parameter and calculate the total size of the files in this directory.

End of exercise

Exercise Instructions with Hints

Working with Positional Parameters

__ 1. If you are not logged in as **tux1** at **tty1**, log in now.

» **<Ctrl+Alt+F1>**

» Login: **tux1**

» Password: **penguin1**

__ 2. Create a shell script named **parameters** that echoes the five parameters that follow using predefined special variables set by the shell to fill in the blanks. Execute the script using the positional parameters 10 100 1000.

» $ **vi parameters**

```
echo The name of this shell script is $0.
echo The first parameter passed is number $1.
echo The second parameter passed is number $2.
echo The third parameter passed is number $3.
echo Altogether there were $# parameters passed.
```

» $ **chmod +x parameters**

» $ **./parameters 10 100 1000**

Conditional Execution

__ 3. Using conditional execution, create a shell script named **checkfile** that checks to see if the file named **parameters** exists in your directory. If it exists, use a command to show the contents of the file. Execute the script.

» $ **vi checkfile**

```
[ -f parameters ] && cat parameters
```

» $ **chmod +x checkfile**

» $ **./checkfile**

__ 4. Modify the **checkfile** script and change the name of the file from **parameters** to **noname** (check to ensure that you do *not* have a file by this name in your current directory). Also, using conditional execution, if the **cat** command was *not* successful, display the error message, “The file was not found.” Execute the script.

» $ **vi checkfile**

```
[ -f noname ] && cat noname || echo "The file 'noname' was not
found"
```

» $ **./checkfile**

__ 5. Modify the **checkfile** script to accept a single parameter from the command line as input to the **ls** and **cat** commands. Execute the script twice, once using the file named **parameters** and again using the file named **noname**.

» $ **vi checkfile**

```
[ -f $1 ] && cat $1 || echo $1 was not found
```

» $ **./checkfile parameters**

» $ **./checkfile noname**

__ 6. Execute the **checkfile** script again, but this time with no parameters. What happens? Modify the script so that this does not appear again.

» $ **./checkfile**

» $ **vi checkfile**

```
[ -f "$1" ] && cat "$1" || echo "$1 was not found"
```

» $ **./checkfile**

Loops

__ 7. Using the **for** loop, modify the **checkfile** script to accept multiple files as input from the command line instead of just one. If the files are found, display all of them. If the files are not found, display an error message showing all file names that were not found. Look in your directory and note a few valid file names that you can use as input. Execute the script using valid and invalid file names.

» $ **vi checkfile**

```
for x in $*
do
    [ -f "$x" ] && cat "$x" || echo "$x was not found"
done
```

» $ **ls**

» $ ./**checkfile filename filename filename**

(where *filename* is replaced by valid and invalid file names from your directory.)

__ 8. Now do the same thing, but use a **while** loop in combination with the **shift** command.

» $ **vi checkfile**

```
while [ ! -z "$1" ]
do
  [ -f "$1" ] && cat "$1" || echo "$1 was not found"
  shift
done
```

» $ ./**checkfile filename filename filename**

Arithmetic

___ 9. From the command line, display the results of multiplying 5 times 6.

» $ **echo $((5 * 6))**

___ 10. Now create a shell script named **math** to multiply any two numbers when entered as input from the command line. Execute the script multiplying 5 times 6. Experiment with any other two numbers.

» $ **vi math**

```
echo $(( $1 * $2 ))
```

» $ **chmod +x math**

» $ **./math 5 6**

Integration Exercise

___ 11. Use the knowledge you gained in this course to write a script that accepts a directory name as a parameter and calculates the total size of the files in this directory.

Note

The column numbers might need to be adjusted a little.

» $ **vi sum**

```
if [ -d "$1" ]
then
  sum=0
  for i in $(ls -l "$1" | cut -c32-42)
  do
     sum=`expr "$sum" + "$i"`
  done
  echo "The total size of files in $1 is $sum."
fi
```

» $ **chmod +x sum**

» $ **./sum /tmp**

End of exercise

Exercise 12. The Linux GUI

What this exercise is about

This exercise provides students an opportunity to get acquainted with the two main Linux desktop environments: KDE and GNOME.

What you should be able to do

At the end of this exercise, students should be able to:

- Start X
- Work with GNOME
- Work with KDE
- List and compare various applications within GNOME and KDE

Exercise Instructions

Starting the GUI

__ 1. Log in as **root** on **tty4**.

__ 2. Edit the /etc/inittab file and make sure the default runlevel is 3.

__ 3. Reboot your system. Does the graphical environment get started?

__ 4. Log in as **tux1** on **tty1** and start X with the **startx** command.

__ 5. End your X environment; then log out and log in as root.

__ 6. Edit the /etc/inittab file again and set the default runlevel to 5. Then reboot the system again. Did the graphical environment start?

Working with GNOME and KDE

__ 7. Log in to the graphical environment using your own name.

__ 8. Both the GNOME and KDE project have delivered various applications, such as word processors, file managers, text editors, and so forth, as standard part of the codebase. These applications are typically direct competitors of the corresponding applications on a Microsoft Windows platform.

A default Linux installation installs a lot of these applications, and you can download more from the GNOME and KDE Web sites.

Browse around in both the GNOME and KDE desktop environments and try to identify the name of the application that fulfills a certain function. (You can retrieve the name of the application by opening a terminal window and executing the **ps** command.) Some names have already been filled in as an example.

To switch between KDE and GNOME, use your display managers (login prompt) menu.

Function	**GNOME**	**KDE**
Window manager	sawfish, metacity	kwin
File manager	nautilus	konqueror
Text editor(s)		
Internet dialer		
Email client		
Web browser		
CD Player		

MP3 Player		
Sound mixer		
Word processor		
Spreadsheet		
Presentation package		
Photo/bitmap editor		
Vector oriented graphics editor		
Clipboard		

__ 9. In both desktop environments, explore the themes capabilities. After setting a theme in KDE, start a GNOME application, and vice versa. Does this work?

__ 10. In KDE, try to start a GNOME application and vice versa. Does this work? Try to cut and paste between KDE and GNOME applications. Does this work?

End of exercise

Exercise Instructions with Hints

Starting the GUI

__ 1. Log in as **root** on **tty4**.

» **<Ctrl+Alt+F4>**

» Login: **root**

» Password: **ibmlnx**

__ 2. Edit the /etc/inittab file and make sure the default runlevel is 3.

» # **vi /etc/inittab**

» Make sure the initdefault line looks like this:

```
id:3:initdefault:
```

__ 3. Reboot your system. Does the graphical environment get started?

» # **reboot**

__ 4. Log in as **tux1** on **tty1** and start X with the **startx** command.

» Login: **tux1**

» Password: **penguin1**

» $ **startx**

__ 5. End your X environment, then log out and log in as **root**.

» **<Ctrl+Alt+Backspace>**

» $ **logout**

» Login: **root**

» Password: **ibmlnx**

__ 6. Edit the /etc/inittab file again and set the default runlevel to 5. Then reboot the system again. Did the graphical environment start?

» # **vi /etc/inittab**

» Make sure the initdefault line looks like this:

```
id:5:initdefault:
```

» # **reboot**

Working with GNOME and KDE

__ 7. Log in to the graphical environment using your own name.

__ 8. Both the GNOME and KDE project have delivered various applications, such as word processors, file managers, text editors, and so forth, as standard part of the codebase. These applications are typically direct competitors of the corresponding applications on a Microsoft Windows platform.

A default Linux installation installs a lot of these applications, and you can download more from the GNOME and KDE Web sites.

Browse around in both the GNOME and KDE desktop environments and try to identify the name of the application that fulfills a certain function. (You can retrieve the name of the application by opening a terminal window and executing the **ps** command.) Some names have already been filled in as an example.

To switch between KDE and GNOME, use your display managers (login prompt) menu.

Function	GNOME	KDE
Window manager	sawfish, metacity	kwin
File manager	nautilus	konqueror
Text editor(s)		
Internet dialer		
Email client		
Web browser		
CD Player		
MP3 Player		
Sound mixer		
Word processor		
Spreadsheet		
Presentation package		
Photo/bitmap editor		
Vector oriented graphics editor		
Clipboard		

__ 9. In both desktop environments, explore the themes capabilities. After setting a theme in KDE, start a GNOME application, and vice versa. Does this work?

__ 10. In KDE, try to start a GNOME application and vice versa. Does this work? Try to cut and paste between KDE and GNOME applications. Does this work?

End of exercise

Exercise 13. Customizing the User Environment

What this exercise is about

When users log in, they generally prefer their environment to be customized to meet their specific needs. In this exercise, the students customize their environment with some very useful functions that are invoked every time they log in.

What you should be able to do

At the end of the lab, you should be able to:

- Customize the **.bash_profile** and **.bashrc** files
- Set alias definitions
- Alter umask values

Exercise Instructions

Customizing the shell environment

__ 1. If you are not logged in, log in as **tux1** at **tty1**.

__ 2. Change the appropriate file to change your environment each time you log in. Make sure that you have the following functions when you log in:

- Change the primary prompt to show you the complete path of the current directory.
- Display a message stating your login name and the date you logged in.
- Define an alias **num** that shows you how many users are logged in at that moment.
- Set the variable **cheese** to **gouda**.

__ 3. Log out and log in again. Check if the functions you defined in Step 1 are activated.

- Does your prompt show complete path of the current directory? _________
- Did your message display? ___________
- Can you use the **num** command? __________
- Is the variable **cheese** set to **gouda**? _________

__ 4. If all the questions above are answered with yes, continue with Step 5; else, try Steps 2 and 3 again to fix the problems.

__ 5. Start a subshell and answer the following questions.

- Does your prompt show the complete path of the current directory? ______
- Did your message display? ___________
- Can you use the **num** command? __________
- Can you use the command history with **vi**? __________
- Is the variable **cheese** set to **gouda**? _________

__ 6. If the settings are also available in subshells, continue with Step 9; otherwise, continue with Step 7.

__ 7. Most settings, with the exception of system variables, apply only to the current environment and are not passed to subshells (child processes). There is a configuration file in your system that makes settings available in subprocesses too. Which file is this? __________

__ 8. Edit the **.bash_profile** and **.bashrc** files so that the correct settings are in the correct configuration file. What settings should be in **.bash_profile** and what settings should be in **.bashrc**?

____

____

__ 9. Log out and log in again and see if your settings are set in your login environment. Also check if the settings are set in a subshell.

__ 10. In the previous steps, you altered configuration files and then logged out and in to activate the new settings. How can you activate settings in an altered customization file without logging out and in again?

__ 11. If you are not in your login shell, return there now.

__ 12. Remove the **num** alias from your environment without editing the **.bashrc** or **.bash_profile** file. Then display the list of aliases currently set and try to execute the **num** alias.

__ 13. Add the **num** alias to your environment and check if **num** is there again.

Customizing the X environment

__ 14. Switch to virtual terminal 7 and log in using your own name. Open a few applications, change some themes and log out. While logging out, select the *save session* check box. Then log in again. Do your applications and settings appear again?

End of exercise

Exercise Instructions with Hints

Customizing the shell environment

__ 1. If you are not logged in, log in as **tux1** at **tty1**.

» **<Ctrl+Alt+F1>**

» Login: **tux1**

» Password: **penguin1**

__ 2. Change the appropriate file to change your environment each time you log in. Make sure that you have the following functions when you log in:

- Change the primary prompt to show you the complete path of the current directory.
- Display a message stating your login name and the date you logged in.
- Define an alias **num** that shows you how many users are logged in at that moment.
- Set the variable **cheese** to **gouda**.

» $ **vi .bash_profile ## or .profile**

```
PS1='$PWD $ '
echo User $LOGNAME logged in at $(date)
alias num="who | wc -l"
cheese=gouda
```

__ 3. Log out and log in again. Check if the functions you defined in Step 1 are activated.

- Does your prompt show complete path of the current directory? _________
- Did your message display? ___________
- Can you use the **num** command? __________
- Is the variable **cheese** set to **gouda**? _________

» $ **exit**

» Login: **tux1**

» Password: **penguin1**

» $ **num**

» $ **echo $cheese**

__ 4. If all the questions above are answered with yes, continue with Step 5; else, try Steps 2 and 3 again to fix the problems.

__ 5. Start a subshell and answer the following questions.

- Does your prompt show the complete path of the current directory? ______
- Did your message display? ____________
- Can you use the **num** command? ___________
- Can you use the command history with **vi**? ___________
- Is the variable **cheese** set to **gouda**? __________

» $ **bash**

» $ **num**

» $ **echo $cheese**

__ 6. If the settings are also available in subshells, continue with Step 9; otherwise, continue with Step 7.

__ 7. Most settings, with the exception of system variables, apply only to the current environment and are not passed to subshells (child processes). There is a configuration file in your system that makes settings available in subprocesses too. Which file is this? ___________

__ 8. Edit the **.bash_profile** and **.bashrc** files so that the correct settings are in the correct configuration file. What settings should be in **.bash_profile** and what settings should be in **.bashrc**?

____

____

» $ **vi .bash_profile ## or .profile**

```
PS1='$PWD $ '
cheese=gouda
export PS1 cheese (and any other variable already exported)
echo User $LOGNAME logged in at $(date)
```

» $ **vi .bashrc**

```
alias num="who | wc -l"
```

__ 9. Log out and log in again and see if your settings are set in your login environment. Also check if the settings are set in a subshell.

» $ **exit**

» Login: **tux1**

» Password: **penguin1**

» $ **num**

» $ **echo $cheese**

» $ **bash**

» $ **num**

» $ **echo $cheese**

__ 10. In the previous steps, you altered configuration files and then logged out and in to activate the new settings. How can you activate settings in an altered customization file without logging out and in again?

» $ **. .bash_profile ## or .profile**

» $ **. .bashrc**

__ 11. If you are not in your login shell, return there now.

» $ **ps**

» $ **exit**

__ 12. Remove the **num** alias from your environment without editing the **.bashrc** or **.bash_profile** file. Then display the list of aliases currently set and try to execute the **num** alias.

» $ **unalias num**

» $ **alias**

» $ **num**

__ 13. Add the **num** alias to your environment and check if **num** is there again.

» $ **. .bashrc**

» $ **num**

Customizing the X environment

__ 14. Switch to virtual terminal 7 and log in using your own name. Open a few applications, change some themes and log out. While logging out, select the *save session* check box. Then log in again. Do your applications and settings appear again?

End of exercise

Exercise 14. Basic System Configuration

What this exercise is about

This exercise provides students an opportunity to become familiar with basic system configuration that might be needed on a workstation.

What you should be able to do

At the end of this exercise, students should be able to:

- Install and deinstall RPMs
- Configure a printer
- Configure a sound card
- Configure the network interface

Required Materials

- A set of installation CDs for your distribution or access to the NFS installation media directories on the classroom server

Exercise Instructions

The RPM Package Manager

__ 1. Log in as root in your graphical environment. Open a terminal window.

__ 2. Make a list of all packages that are installed on the system.

__ 3. List the information of the bash package.

__ 4. List all files in the bash package.

__ 5. List all the package files that are available on the distribution CD-ROMs or Network Install Server.

__ 6. Remember the **vlock** command that we tried to use in Exercise 3? We could not do that exercise because **vlock** was not installed. Now that you know how to install an RPM, install the **vlock** RPM, and try to perform that particular exercise once more.

Before you install the **vlock** RPM, list the information of the RPM, and list all files in the RPM.

```
vlock is in /export/fedo5 or /export/rhel4 or /export/sles10.
```

__ 7. Verify that the application vlock is indeed installed by performing the exercises from Exercise 3.

__ 8. Uninstall vlock and verify that it indeed is no longer available.

Configuring a printer (Optional)

If a printer is available in your classroom (either locally attached to your system or remotely via the network), then your instructor will provide you with the information about this printer. If no printer is available, then skip this exercise.

__ 9. Use your browser to configure your printer.

__ 10. Print the /etc/passwd file.

Configuring a Sound Card (Optional)

__ 11. Use the sound card configuration tool that came with your distribution and configure your sound card. Then try to play some audio.

Configuring your network (Optional)

In most classrooms, it is not possible to alter the network configuration because this might lead to network problems, which might also affect other classes that are currently running. If it is safe to play with network settings, your instructor will give you additional exercises to perform.

__ 12. Browse the files where the network configuration for your system is stored.

__ 13. If the classroom uses DHCP to configure your network card, then take a look at the current configuration with the **ifconfig** and **route** commands.

__ 14. Ask your instructor for permission to modify the current network settings. This is a safety issue, because a wrong network configuration might lead to problems for other students -- even students in other classrooms! If you obtained permission, start the configuration tool that is appropriate for your distribution and configure static networking, using the IP address, netmask, and default gateway that you saw in the previous exercise.

End of exercise

Exercise Instructions with Hints

The RPM Package Manager

___ 1. Log in as root in your graphical environment. Open a terminal window.

___ 2. Make a list of all packages that are installed on the system.

» # **rpm -q -a**

___ 3. List the information of the bash package.

» # **rpm -q -i bash**

___ 4. List all files in the bash package.

» # **rpm -q -l bash**

___ 5. List all the package files that are available on the distribution CD-ROMs or Network Install Server.

» **# mkdir /media/install**

» If you did a CD-based install:

# **mount /media/cdrom**

» If you did a network (NFS) -based install:

# **mount <server>:<dir> /media/install**

Note

<server> is **10.0.0.1**

and **<dir>** is **/export/fedo7** or **/export/rhel51c** or **/export/sled10**

» # **cd /media/install**

» # **find . -name "*.rpm"**

» # **cd**

» # **umount /media/install**

» If you did a CD-based install, do this for all CDs.

___ 6. Remember the **vlock** command that we tried to use in Exercise 3? We could not do that exercise because **vlock** was not installed. Now that you know how to install an RPM, install the **vlock** RPM, and try to perform that particular exercise once more.

Before you install the **vlock** RPM, list the information of the RPM, and list all files in the RPM.

The vlock RPM is in /export/rhel51c or /export/sles10. However, if you are using Fedora 7, you will not find the vlock RPM file in /export/fedo7. You will have to access the vlock RPM from the instructor’s /export/files directory. Ask for help.

» # **rpm -q -i vlock**

You should get an error: package vlock is not installed.

» If you did a CD-based install:

# **mount /media/cdrom**

» If you did a network (NFS)-based install:

# **mount <server>:<dir> /media/install**

» fedora# **cd /media/install/Fedora/**
redhat# **cd /media/install/RedHat/Client**
suse# **cd /media/install/suse/i586**

» # **rpm -qip vlock-*version*.rpm**

» # **rpm -qlp vlock-*version*.rpm**

» # **rpm -ivh vlock-*version*.rpm**

» # **rpm -qi vlock**

» # **rpm -ql vlock**

» # **cd**

» # **umount /media/install**

__ 7. Verify that the application vlock is indeed installed by performing the exercises from Exercise 3.

» **<Ctrl+Alt+F1>**

» $ **vlock**

» $ **vlock -a**

__ 8. Uninstall vlock and verify that it indeed is no longer available.

» **<Alt+F7>**

» # **rpm -e vlock**

» **<Ctrl+Alt+F1>**

» $ **vlock**

Configuring a printer (Optional)

If a printer is available in your classroom (either locally attached to your system or remotely via the network), then your instructor will provide you with the information about this printer. If no printer is available, then skip this exercise.

__ 9. Use your browser to configure your printer.

» Start your browser and use **http://localhost:631** as URL.

- or - **yast2 printer**

- or - **system-config-printer**

- or - **kprinter**

__ 10. Print the /etc/passwd file.

» # **lpr /etc/passwd**

Configuring a Sound Card (Optional)

__ 11. Use the sound card configuration tool that came with your distribution and configure your sound card. Then try to play some audio.

» fedora/redhat# **system-config-soundcard**

suse# **yast2 sound**

Configuring your network (Optional)

In most classrooms, it is not possible to alter the network configuration since this might lead to network problems which may also affect other classes that are currently running. If it is safe to play with network settings, your instructor gives you additional exercises to perform.

__ 12. Browse the files where the network configuration for your system is stored.

» fedora/redhat# **less /etc/sysconfig/network-scripts/ifcfg-***

suse# **more /etc/sysconfig/network/ifcfg-***

__ 13. If the classroom uses DHCP to configure your network card, then take a look at the current configuration with the **ifconfig** and **route** commands.

» $ **ifconfig**

» $ **route**

__ 14. Ask your instructor for permission to modify the current network settings. This is a safety issue, because a wrong network configuration might lead to problems for other students -- even students in other classrooms! If you obtained permission, start the configuration tool that is appropriate for your distribution and configure static networking, using the IP address, netmask, and default gateway that you saw in the previous exercise.

» fedora/redhat# **system-config-network**

» suse# **yast2 lan**

End of exercise

Exercise 15. Integrating Linux in a Windows Environment

What this exercise is about

This exercise provides students an opportunity to become familiar with the different options when integrating Linux in a Windows environment.

What you should be able to do

At the end of this exercise, students should be able to:

- Access Windows filesystems
- Access Windows servers

Notice: Because VMWare and Win4Lin require commercial licenses, they cannot be demonstrated in this class.

WINE requires extensive configuration and is, therefore, not included in the exercises.

Because Bochs is typically not included in a distribution, but needs to be installed from source, it cannot be demonstrated either.

Required Materials

- The NetBIOS name of a Windows server, and the name and password of a user account/home directory on that server

Exercise Instructions

Accessing Windows filesystems

Note

At the time of this writing, the Linux kernel could only mount an NTFS filesystem read-only. Read-write support was under development, but far from reliable. As a result of this, some distribution manufacturers decided not to include NTFS in the precompiled distribution kernel at all. So if your partition type is NTFS, you might not be able to mount it without a kernel recompile. Kernel compiles are beyond the scope of this course.

At the time of this writing, Fedora and Red Hat do not include NTFS support in their distribution, but SuSE does.

__ 1. Make a list of all partitions that exist on your system with the **fdisk -l /dev/hda** (IDE) or **fdisk -l /dev/sda** (SCSI) command.

__ 2. List all filesystems that are currently mounted with the **mount** command. Compare this list with the output of the previous command. This should give you a list of Windows filesystems that are not mounted yet.

__ 3. Create mountpoints under /mnt for all Windows filesystems that you want to mount. Then, mount these filesystems manually, using the **mount** command. Verify that the filesystem was indeed mounted and list the contents of the filesystem.

__ 4. Add a line to the /etc/fstab file so that this filesystem is mounted automatically when the system boots, and reboot the system to verify that this worked.

__ 5. Ask your instructor for a blank floppy disk. Format this disk using the **mformat** command. Try to access the floppy disk, both using the mtools and by mounting it.

WINE (Optional)

Note

You can perform this exercise only if:

- You have a Microsoft Office CD, and a valid license for this product.
- You have a valid Codeweavers CrossOver Office license (a 30-day trial license can be requested on the Web site).

  We are using CrossOver Office because we're not just *running* Microsoft Office, but we're *installing* it as well. CrossOver Office makes this process extremely easy, although it can be done without CrossOver Office.

Because of licensing issues, one or both of the conditions described previously might not have been satisfied in class. In that case, you cannot perform this exercise.

__ 6. Go to the Web page and download the CrossOver Office installation shell script as instructed in the e-mail in which your license (either trial or commercial) was delivered to you.

__ 7. As root, start the CrossOver Office installation shell script.

__ 8. As a regular user, start the CrossOver Setup Program (**Start> CrossOver> Office Setup**). Work through the menus to install Microsoft Office.

__ 9. Start one of the Microsoft Office applications using the **Start> Windows Applications> Programs** menu path.

Win4Lin (Optional)

Note

You can perform this exercise only if:

- You have a valid Win4Lin license.
- You have a Microsoft Windows 95,98, or ME installation CD, and a valid license key.
- You have a fairly fast Internet connection: The Win4Lin installer downloads Win4Lin over the Internet during installation, and this might be a download of 20 MB, depending on the distribution you use, and the options you choose.

Because of licensing issues, one or both of the first two conditions described previously might not have been satisfied in class. In that case, you cannot perform this exercise.

__ 10. Check to see whether your instructor has already downloaded the Win4Lin installer, and what the license key is. Copy over or download the Win4Lin installer to /root.

__ 11. Unpack the Win4Lin installer.

__ 12. Start the Win4Lin installer.

__ 13. Work through the screens of the Win4Lin installer. Provide the license information when asked for, and reboot your system when a Win4Lin capable kernel has been installed.

__ 14. Log in as root and start the Win4Lin installer again.

__ 15. When the system-wide installation has finished, log out and log in as yourself. Then start the Win4Lin installer again.

__ 16. Work through the screens again. Eventually, a new windows open in which the Windows installer runs. This should be familiar: Enter the Windows license key when required.

__ 17. When the Windows install is finished, play with it. Also, shut down Windows and try to start it again. From a command prompt, this is done with the **win** command. In certain distributions and desktop environments, a Win4Lin entry is also added to your Start menu.

VMWare (Optional)

Note

You can perform this exercise only if:

- You have a valid VMWare Workstation license, or obtained a demo (30-day) license
- You have a Microsoft Windows installation CD, or a preinstalled Microsoft Windows VMWare image available, and a valid license for Microsoft Windows.

Because licensing issues, one or both of the conditions described previously might not have been satisfied in class. In that case, you cannot perform this exercise.

__ 18. Check with your instructor if the VMWare license, the VMWare RPM, and the Windows CD or images are available, and what their location is.

__ 19. Download and install the VMWare RPM.

__ 20. Run the VMWare configuration script and configure VMWare for your situation. Usually, the defaults are okay.

__ 21. If you have a VMWare license file, then create the directory .vmware in your home directory, and copy the license file into this directory. Alternatively, start **vmware** and enter the license information in the Help; Enter Serial Number window.

__ 22. If your instructor has a preinstalled Windows VMWare image available, then unpack this image. Start VMWare, open the image configuration file (*.vmx), and start the virtual machine.

__ 23. If your instructor has a Windows CD available, then start VMWare, and create a new virtual machine with the New Virtual Machine Wizard. Insert the CD and start the virtual machine.

Accessing Windows servers

Your instructor configures a Windows server or a Samba server so that you can access this using the Samba client software, which is part of Linux. The instructor will provide you with the following information about this server:

- Netbios name
- Share name
- User name
- Password

__ 24. Use the **smbclient** program to retrieve information from the Windows or Samba server. Then use it to access the share *ftp-style*. Upload and download the /etc/passwd file to test if things are working.

__ 25. Create a mount point for this Windows share called /mnt/share.

__ 26. Mount the share on this mountpoint using the **smbmount** command. Verify that the mount succeeded.

__ 27. Edit the /etc/fstab file and add an entry for this share. Then reboot the system and verify that the share was mounted after the reboot.

OpenOffice

__ 28. Start OpenOffice and try to create and save various types of documents: text documents, presentations, spreadsheets, and so forth. Try to save them in Microsoft-compatible formats.

__ 29. Start an Internet browser and use Google to search for and obtain various documents in Microsoft formats (search, for instance, for test.doc, test.ppt and test.xls). See if you can open, modify, and save these files.

End of exercise

Exercise Instructions with Hints

Accessing Windows filesystems

Note

At the time of this writing, the Linux kernel could only mount an NTFS filesystem read-only. Read-write support was under development, but far from reliable. As a result of this, some distribution manufacturers decided not to include NTFS in the precompiled distribution kernel at all. So if your partition type is NTFS, you might not be able to mount it without a kernel recompile. Kernel compiles are beyond the scope of this course.

At the time of this writing, Fedora and Red Hat do not include NTFS support in their distribution, but SuSE does.

__ 1. Make a list of all partitions that exist on your system with the **fdisk -l /dev/hda** (IDE) or **fdisk -l /dev/sda** (SCSI) command. Remember which one you have; the rest of the exercise refers to /**dev/hda.** Use the correct designation.

» # **fdisk -l /dev/hda**

- OR -

# **fdisk -l /dev/sda**

__ 2. List all filesystems that are currently mounted with the **mount** command. Compare this list with the output of the previous command. This should give you a list of Windows filesystems that are not mounted yet.

» # **mount**

__ 3. Create mountpoints under /mnt for all Windows filesystems that you want to mount. Then, mount these filesystems manually, using the **mount** command. Verify that the filesystem was indeed mounted and list the contents of the filesystem.

» # **mkdir /mnt/windows**

» # **mount -t ntfs /dev/hda1 /mnt/windows**

» # **mount**

» # **ls -l /mnt/windows**

__ 4. Add a line to the /etc/fstab file so that this filesystem is mounted automatically when the system boots, and reboot the system to verify that this worked.

» # **vi /etc/fstab**

» Add the following line:

```
/dev/hda1    /mnt/windows ntfs   defaults    0 0
```

» # **reboot**

» After reboot, log in as root.

» # **mount**

__ 5. Ask your instructor for a blank floppy disk. Format this disk using the **mformat** command. Try to access the floppy disk, both using the mtools and by mounting it.

Note

Your PC might not have a floppy drive. In that case, use a CD-ROM or USB key instead. This might require help from the instructor to configure the /etc/mtools.conf file. Feel free to try it yourself, but ask for help if you need it.

» # **mformat a: ### CAUTION: FLOPPY ONLY!!!**

» # **mdir a:**

» # **mcopy /etc/passwd a:**

» # **mdir a:**

» # **mount /media/floppy**

» # **ls /media/floppy**

» # **cat /media/floppy**

» # **umount /media/floppy**

WINE (Optional)

Note

You can perform this exercise only if:

- You have a Microsoft Office CD, and a valid license for this product.
- You have a valid Codeweavers CrossOver Office license (a 30-day trial license can be requested on the Web site).

  We are using CrossOver Office because we're not just *running* Microsoft Office, but we're *installing* it as well. CrossOver Office makes this process extremely easy although it can be done without CrossOver Office.

Because of licensing issues, one or both of the conditions described previously might not have been satisfied in class. In that case, you cannot perform this exercise.

__ 6. Go to the Web page and download the CrossOver Office installation shell script as instructed in the e-mail in which your license (either trial or commercial) was delivered to you.

__ 7. As root, start the CrossOver Office installation shell script.

```
» # ./install-crossover-office-version.sh
```

__ 8. As a regular user, start the CrossOver Setup Program (**Start> CrossOver> Office Setup**). Work through the menus to install Microsoft Office.

__ 9. Start one of the Microsoft Office applications using the **Start> Windows Applications> Programs** menu path.

Win4Lin (Optional)

Note

You can perform this exercise only if:

- You have a valid Win4Lin license
- You have a Microsoft Windows 95,98, or ME installation CD, and a valid license key.
- You have a fairly fast Internet connection: The Win4Lin installer downloads Win4Lin over the Internet during installation, and this might be a download of 20 MB, depending on the distribution you use, and the options you choose.

Because of licensing issues, one or both of the first two conditions described previously might not have been satisfied in class. In that case, you cannot perform this exercise.

__ 10. Check to see whether your instructor already downloaded the Win4Lin installer, and what the license key is. Copy over or download the Win4Lin installer to /root.

__ 11. Unpack the Win4Lin installer.

» # **tar -zxvf netraverse_installer5.tgz**

__ 12. Start the Win4Lin installer.

» # **cd netraverse_installer**

» # **./win4lin-install**

__ 13. Work through the screens of the Win4Lin installer. Provide the license information when asked for, and reboot your system when a Win4Lin capable kernel has been installed.

__ 14. Log in as root and start the Win4Lin installer again.

» # **cd netraverse_installer**

» # **./win4lin-install**

__ 15. When the system-wide installation has finished, log out and log in as yourself. Then start the Win4Lin installer again.

» $ **win4lin-install**

__ 16. Work through the screens again. Eventually, a new window opens in which the Windows installer runs. This should be familiar: Enter the Windows license key when required.

__ 17. When the Windows install is finished, play with it. Also, shut down Windows and try to start it again. From a command prompt, this is done with the **win** command. In certain distributions and desktop environments, a Win4Lin entry is also added to your Start menu.

VMWare (Optional)

Note

You can perform this exercise only if:

- You have a valid VMWare Workstation license, or obtained a demo (30-day) license
- You have an Microsoft Windows installation CD, or a preinstalled Microsoft Windows VMWare image available, and a valid license for Microsoft Windows.

Because of licensing issues, one or both of the conditions described previously might not have been satisfied in class. In that case, you cannot perform this exercise.

__ 18. Check with your instructor if the VMWare license, the VMWare RPM and the Windows CD or images are available, and what their location is.

__ 19. Download and install the VMWare RPM.

» # **rpm -ivh VMware-Workstation-*version*.i386.rpm**

__ 20. Run the VMWare configuration script and configure VMWare for your situation. Usually, the defaults are okay.

» # **vmware-config.pl**

__ 21. If you have a VMWare license file, then create the directory .vmware in your home directory, and copy the license file into this directory. Alternatively, start **vmware** and enter the license information in the Help; Enter Serial Number window.

» # **cd**

» # **mkdir .vmware**

» # **cp /somewhere/license.ws.*version* .vmware/**

__ 22. If your instructor has a preinstalled Windows VMWare image available, then unpack this image. Start VMWare, open the image configuration file (*.vmx), and start the virtual machine.

__ 23. If your instructor has a Windows CD available, then start VMWare, and create a new virtual machine with the New Virtual Machine Wizard. Insert the CD and start the virtual machine.

Accessing Windows servers

Your instructor configures a Windows server or a Samba server so that you can access this using the Samba client software, which is part of Linux. He or she provides you with the following information about this server:

- Netbios name   Share name
- User name   Password

__ 24. Use the **smbclient** program to retrieve information from the Windows or Samba server. Then use it to access the share *ftp-style*. Upload and download the /etc/passwd file to test if things are working.

» # **smbclient -L *winserver* -N**

» # **smbclient -L *winserver* -U *username***

» # **smbclient -L *winserver* -U *username%password***

» # **smbclient** //***winserver*/*share* -U *username%password***

» smb> **put /etc/passwd passwd**

» smb> **get passwd my_passwd**

» smb> **quit**

__ 25. Create a mount point for this Windows share called /mnt/share.

» # **mkdir /mnt/share**

__ 26. Mount the share on this mountpoint using the **smbmount** command. Verify that the mount succeeded.

» # **smbmount //*winserver*/*share* /mnt/share -o username=*username%password***

» # **mount**

» # **ls /mnt/share**

__ 27. Edit the /etc/fstab file and add an entry for this share. Then reboot the system and verify that the share was mounted after the reboot.

» # **vi /etc/fstab**

» Add the following line:

```
//winserver/share /mnt/share smbfs defaults,username=username%password 0 0
```

» # **reboot**

» After reboot, log in as root.

» # **mount**

OpenOffice

__ 28. Start OpenOffice and try to create and save various types of documents: text documents, presentations, spreadsheets, and so forth. Try to save them in Microsoft-compatible formats.

__ 29. Start an Internet browser and use Google to search for and obtain various documents in Microsoft formats (search, for instance, for test.doc, test.ppt, and test.xls). See if you can open, modify, and save these files.

End of exercise

Exercise 16. End-of-course Challenge Exercise

What this exercise is about

This (optional) exercise provides an opportunity to apply all concepts and techniques learned in this course in a single, real-life scenario.

What you should be able to do

At the end of this exercise, students should be able to apply all concepts and techniques learned in this course in a real-life scenario.

Exercise Instructions

You have decided to start using Linux on your home PC, and you have convinced the other members of your family (who also use this PC) to give it a try as well. They are not convinced that they want to get rid of the current MS-Windows installation though.

Together with your family members, you have created a list of requirements regarding the Linux installation:

1. The PC should have a dual-boot installation, where a boot menu allows you to boot the current, Microsoft Windows OS, and the new Linux installation. The default OS should be Linux.
2. The partitioning scheme should include the current Windows partition, a /boot partition, a / partition, a swap partition, and a data partition. The data partition should be a FAT filesystem, so that it is accessible from Windows as D:-drive, and should be mounted under Linux as /mnt/data. The mount permissions of /mnt/data should be set so that everybody can access all files. (Note that a FAT filesystem does not support permissions; so you have to work with mount options to achieve this. Consult the manual page of mount for the correct options.)
3. When Linux boots, it should come up with a graphical login prompt. Because of disk space considerations, install KDE only, not GNOME.
4. Each member of your family should have an individual user account. Create these user accounts, and set the password identical to the username. Your family members can change their passwords later. Obviously, create a user account for yourself also.
5. All unnecessary services should be switched off. For all services that are running, go to your distributions Web site and download and install all available updates.
6. The relevant parts of the documentation that came with your distribution should be copied to disk, in a suitable location.
7. The household printer (if available) should be configured.
8. The sound card (if available) should be configured.
9. The network adapter (if available) should be configured with a dynamic (DHCP) IP address.
10. Your youngest daughter is two years old and has problems with her hand-eye coordination. When she logs in (with help), she should find a desktop full of simple applications that help her train her hand-eye coordination, such as xeyes, xbill, and SameGnome.
11. To maintain the system, you need two shell scripts, which should be executable by root only:
    - A script that checks all filesystems (including the Windows and the data filesystem), and warns you if the utilization of one of them gets above 90%

- A script that prints the amount of data in each user’s home directory, sorted by disk usage

Implement all requirements listed previously as well as possible, using your student guide and any documentation you can find as a reference. A few requirements are not fully covered in the course. In this case, use the manual pages and other documentation to find out the correct commands and options.

This exercise has no hints, but you can consult your instructor if you have problems fulfilling a requirement.

End of exercise