Alternative OS – Lecture 3 – 26.9.2006

Plan:

1. Logging in
2. Navigating the file system
3. Managing directories
4. Understanding directory listings
5. Managing files
6. Editing files
7. Web browsing in text mode
8. Logging off and turning off the system

1. Logging in

Before you login, you have to "bootstrap" the system. That is turn the computer on, and let the operating system prepare it for work.

During start up Unix "pools itself up by its own bootstraps":

a. Kernel is loaded /unix; /vmunix; /vmlinuz
   i. Bootloader is started
   ii. Bootloader loads the kernel itself
   iii. Kernel tests available memory
b. Initialization tasks
c. System programs are loaded and started
d. Login program is loaded

After connecting with a Unix system, a user is prompted for a *login* username, and then a *password.* While user types the password, the input will not be displayed.

A user can later change his password using *passwd* command. The program would ask to type the same password twice in order to avoid typos.

There are some rules concerning passwords:

Don't

use a word (or words) in any language
use a proper name
use information that can be found in your wallet
use information commonly known about you (car license, pet name, etc)
use control characters. Some systems can't handle them

write your password anywhere
ever give your password to *anybody*

Do

use a mixture of character types (alphabetic, numeric, special)
use a mixture of upper case and lower case
use at least 6 characters
choose a password you can remember
change your password often
make sure nobody is looking over your shoulder when you are entering your password

After you finish your session with the system you have to *log out.* Here we list several ways to *exit* a program or a system.

^C	interrupt
^D	leave the system (usually is disabled)
Exit	leave the shell
logout	leave the system

2. Navigating and listing directories

You can choose a directory using *cd* command:

```
% cd /some/place/in/the/tree
```

Use *pwd* command to assert your position. This command prints the current working directory.

```
% pwd
```

One can *list* the working directory content using *ls* command:

```
% ls
```

3. Understanding directory listings

If you give *ls* command with *–F* option, it will list the working directory entries with corresponding *trailing symbols:*

Directories	/
sockets	=

symbolic links @
executables *

Ordinary files don't have a trailing symbol.

4. Managing directories

Please, choose your home directory. It can be done in several ways:

```
% cd /home/student

% cd ~student

% cd ~

% cd
```

5. Path – relative and absolute

A place of a file or a place of a directory is specified by the *path.*
Give me an example of a path!

```
/bin

/home/student/barsik
```

All these paths start at the root, and they are called *absolute paths.* However, there can be a path that doesn't start with *'/'* sign!

Choose your *parent directory* by giving the following command

```
% cd ..
```

Now print your working directory!

```
% pwd
```

Now give the following command

```
% mkdir student/sharik
```

Have you noticed that we did not specify the path starting with the root, the *'/'* symbol? This time we gave the path *relative* to */home* directory. This kind of paths that does not start with *'/'* is called *relative paths.*

So there are two kinds of paths:

Absolute – start from the root of the directories tree
(/home/student/barsik)

Relative – start from the current directory
(student/barsik)

Now choose your home directory again, and delete the directory of your pet (~/barsik).

```
% cd ~
% rmdir barsik/
```

Let us now try to create a directory inside the deleted pet directory. Use a relative path to achieve that.

```
% mkdir barsik/fish
```

No surprise, it does not work! Indeed, the /home/student/barsik directory no longer exists. However, there is a way around!

```
% mkdir –p barsik/fish
```

Now list your home directory.

```
% ls
```

You can see that barsik/ is back! That is the –p option of mkdir command made it create all *intermediate* directories that were needed to create /home/student/barsik/fish. In this case it was /home/student/barsik that was supposed to contain fish/ subdirectory, but didn't yet exist itself. So mkdir command created it, when it was needed.

As you can see, the real power of Unix commands is in their options. However, it is not always possible to remember all the options that a command has. For that reason, a *MANUAL* is at your service!

6. Using 'man' pages

The manual is invoked in two different ways:

Situation One – You know the name of the command, and you want to also know the available options (flags). You call the *man page* for that command (program) like that

% man *command*

Situation Two - You know what the command does, you know the *topic*, but you don't know the name of the command. You call the *man page* for that *topic*.

% man **–k** *topic*

Examples:

% man ls

% man cd

% man –k directory

% man –k terminal

After you have your *man page* (manual for the command you are looking for) displayed, you can use keys *'j'* and *'k'* to scroll the text. The *man page* is usually displayed by the *less* editor. If you press *'h'*, the *help* will be shown. There are few commands that you will often need to know:

j – move forward one line
k – move backward one line
z – move forward one window
w – move backward one window
g – go to the first line of the file
G – go to the last line of the file
q – quit the editor (close *man page*)

7. Some useful shell commands

There are some shell commands that can be fun or need. It is up to you if you ever come to use them.

cal - displays a simple calendar. The ***–y*** flag can be used to display the whole year calendar.

% cal –y

date - shows you current date and time when invoked without arguments. A *superuser* can also set the current date and time using this utility (command). Look up the manual for more information about this command:

% man date

clear – simply clears the screen.

End of the class.

8. **How to use vi:**
   http://www.groovyweb.uklinux.net/?category=linux&page_name=how%20to%20use%20vi
9. **How to use lynx:**
   Arrow keys: Up and Down to move. Right to follow a link; Left to go back.
   H)elp O)ptions P)rint G)o M)ain screen Q)uit /=search [delete]=history list

4. **Halting and rebooting:**
   a. shutdown [-g sec] [-nqr]; shutdown [+min]
   b. halt [-nq] = shutdown –h
   c. reboot [-nq] = shutdown -r