VE482 Lab 2

Liu Yihao 515370910207

1 Basic shell

- Use the `mkdir`, `touch`, `mv`, `cp`, and `ls` commands to:
  - Create a file named `test`.
    ```
    touch test
    ```
  - Move test to `dir/test.txt`, where `dir` is a new directory.
    ```
    mkdir dir
    mv test dir/test.txt
    ```
  - Copy `dir/test.txt` to `dir/test_copy.txt`.
    ```
    cp dir/test.txt dir/text_copy.txt
    ```
  - List all the files contained in `dir`.
    ```
    ls dir -a
    ```
- Use the `grep` command to:
  - List all the files form `/etc` containing the pattern `127.0.0.1`.
    ```
    grep -r '127.0.0.1' /etc
    ```
  - Only print the lines containing your username and root in the file /etc/passwd (only one grep should be used)
    ```
    grep -rE '(liu|root)' /etc/passwd
    ```
- Use the `find` command to:
  - List all the files from `/etc` that have been accessed less than 24 hours ago.
    ```
    find /etc -atime 1
    ```
  - List all the files from `/etc` whose name contains the pattern “netw”.
    ```
    find /etc -name '*netw*'
    ```
- In the bash man-page read the part related to redirections. Explain the following signs `>`, `>>`, `<<<`, `>&1`, and `2>&1 >`. What is the use of the tee command.

  `>` redirects the standard output into a file.

  `>>` redirects and appends the standard output into a file.

  `<<<` redirects the contents on the right as the standard input of the command on the left.

  `>&1` redirects the standard output into standard output (meaningless).

  `2>&1 >` redirects the standard error into standard output, and redirects the origin standard output into a file.

- Explain the behaviour of the `xargs` command and of the | sign.

  `xargs` is used to build and execute command lines from standard input, by combining multi lines and extra spaces into a line with single spaces.

  The | sign pipes the standard output of the command on the left into the command on the right as the standard input.

- What are the `head` and `tail` commands? How to "live display" a file as new lines are appended?

  `head` and `tail` are used to get the first and last several lines of a file.

  Use the `-f` option of `tail` to "live display" a file as new lines are appended.

- How to monitor the system using `ps`, `top`, `free`, `vmstat`?

  `ps` is used to monitor the processes.

  `top` is used to monitor the CPU and RAM of processes.

  `free` is used to monitor the RAM.

  `vmstat` is used to monitor the RAM, IO and CPU in a period.

- In Minix 3, how to manage softwares (install, remove, update...)?

```
pkgin update       # Update the package repository
pkgin install name # Install a package
pkgin remove name  # Remove a package
pkgin upgrade name # Upgrade a package
pkgin search name  # Search a package
```

- What is the purpose of the commands `ifconfig`, `adduser`, and `passwd`?

  `ifconfig` is used to checks the state of network.

  `adduser` is used to create a new user.

  `passwd` is used to set password for the current user.

2 Working on a remote server

- Setup an SSH server on Minix 3. From Linux (using ssh) or Windows (using Putty) log into Minix 3. Note: the network need to be properly setup on the Virtual Machine (VM).

```
ssh root@192.168.1.101
```

- What is the default SSH port? Change this port for port 2222. Log into Minix 3 using this new SSH server setup.

  The default port is 22.

  On Minix3:

```
vi /etc/ssh/sshd_config
```

  and edit the option "Port".

  On Linux:

```
ssh root@192.168.1.101 -p2222
```

- List and explain the role of each the file in the `$HOME/.ssh` directory. In `$HOME/.ssh/config`, create an entry for Minix 3.

```
ls $HOME/.ssh
```

- Briefly explain how key-only authentication works in SSH. Generate a key-pair on the host system and use it to log into Minix 3 without a password.

  On Minix3:

```
ssh-keygen -t rsa
```

  and copy `$HOME/.ssh/id_rsa` to Linux.

  On Linux:

```
ssh root@192.168.1.101 -p2222 -iid_rsa
```

3 Basic Bash scripting

- What should be the first line of a Bash script?

```
#/bin/bash
```

- What are the main differences between sh, bash, csh, and zsh?
- How to define and access variables?

```
var=1        # define a variable named var and assign it as 1
echo ${var} # echo the defined variable
```

- What is the meaning of `$0, $1,..., $?, $!`?

  `$0` means `argv[0]` in C.

  `$1` means `argv[1]` in C.

  `$?` means the exit status of the last command.

  `$!` means the process id of the last command.

- How to define arrays and access or assign elements?