VE482 Lab Report

Lab 2 - Fall 2021

Name: Shengyuan Xu
ID: 518370910200
Email: xushengyuan@sjtu.edu.cn

Table of Contents

- Getting started with Minix 3
- SSH
- Basic git

1 MINIX 3

Softwate Management

- install: `pkgin install package`
- remove `pkgin remove package`
- update: `pkgin update`

Commands

- `ifconfig`: display basic network information, including LAN ip, netmask and so on; configure TCP/IP
- `adduser`: add a new user to the system; created a user folder (and skeleton files) if required
- `passwd`: modify (or initially set) a user's password

2 SSH

Setup

```
pkgin update
pkgin install openssh
vim /usr/pkg/etc/ssh/sshd_config # change Port 22 to Port 2222
passwd # set a password for root
```

Q&A

- the default SSH port is `22`
- files in `$HOME/.ssh` includes:
  - `id_rsa`: private key
  - `id_rsa.pub`: public key
  - `known_hosts`: a list with SSH fingerprints for each machine that the user have ever logged into.
  - `authorized_keys`: a list of public keys. When a user with a certain private key tries to connect with the host, it is compared with existing public keys in this file. If matched, the connection is allowed.
  - key-only authentication: first, a pair of key is generated by the user, including a private key and a public key. Next, the public key is sent to the server, which is stored and marked as authorized. In this way, the server will check the private key of any users that tries to connect with the server via SSH. If the keys are matched, the user can login onto the server.
    - use command `ssh-keygen` to generate `ssh` key pair on windows or wsl
    - copy the public key in `~/.ssh/id_rsa.pub` on Windows or wsl into `/root/.ssh/authorized_keys` on minix3
    - lon into Minix3 with SSH: `ssh root@192.168.164.128 -p 2222`. This time, we don't need to input password

3 Git

help

- `help`: display common Git commands that are used in various situations.

start a working area

- `clone`: Clone a repository into a new directory
- `init`: Create an empty Git repository or reinitialize an existing one

work on the current change

- `add`: Add file contents to the index

examine the history and state

- `diff`: Show changes between commits, commit and working tree, etc
- `log`: Show commit logs

grow, mark and tweak your common history

- `branch`: List, create, or delete branches
- `checkout`: Switch branches or restore working tree files
- `commit`: Record changes to the repository
- `merge`: Join two or more development histories together
- `reset`: Reset current HEAD to the specified state
- `tag`: Create, list, delete or verify a tag object signed with GPG

collaborate

- `push`: Update remote refs along with associated objects
- `pull`: Fetch from and integrate with another repository or a local branch
- `fetch`: Download objects and refs from another repository