Network Project Week

Xiaolin Wang

June 2019

Contents

1 Caution

- You are **required** to write your report with `Emacs org-mode`
  - Org-mode Tutorials
    - David O'Toole Org tutorial
    - More org-mode tutorials
    - `info org`
  - Emacs中文网

- You have to submit your report as a `tar ball` which have both an `org file` and a generated `html file` included.
  - Report template: `org file`, `html file`
  - A small tutorial about writing lab report: `tutorial.ttyrec`. To view it:

    ```
    ttyplay tutorial.ttyrec
    ```

    Feel free to make your own `ttyrec` file while doing this lab work. For example:

    ```
    ttyrec my-ssh-lab-work.ttyrec
    ttyrec my-http-lab-work.ttyrec
    ttyrec my-email-lab-work.ttyrec
    ttyrec my-ftp-lab-work.ttyrec
    ttyrec my-iptables-lab-work.ttyrec
    ```

  - This file itself is generated from an org file (proj-week.org). You can take it as an example.

- **Deadline:** `2019-06-28 (Fri), 23:59`
  - Submit your report as a `tgz` file here. In your `tgz` file, there must be:
    - an org file
    - an html file
    - optionally one or more `ttyrec` files
  - Late reports will be penalized 20% per day.

- MS-word file will **NOT** be accepted.

- Cheating will result in automatic failure of this work.

2 SSH (25 pts)

2.1 Installation (5 pts)

In our Debian system, openssh-server and openssh-client are installed by default. And the ssh server should have been running. You can check it by

```
nmap localhost
```

The output of the above command should contain the following line:

```
22/tcp    open    ssh
```

And you should be able to connect to your local ssh server by

```
ssh username@localhost
```

NOTE: You should change username to your real user name (should be stud in the lab).

If you cannot find the ssh server nor can you find the ssh command, you should check whether the openssh-server and openssh=client are installed by

```
aptitude search '~i openssh'
```

If you cannot see any outputs, that means you haven't got the necessary packages install. So you have to install them by

```
sudo aptitude install openssh-client openssh-server
```

2.2 Basic usage (5 pts)

```
ssh user@server
```

You've tried connecting your own ssh server in previous section. Now you can try ssh into your neighbor's system.

And you can also try

```
ssh user@server [command]
```

Where command could be any valid shell command, for example

```
ssh user@server ls -l
ssh user@server df
ssh user@server w
ssh user@server free
```

2.3 SSH without password (5 pts)

If you want to login to `cs3.swfu.edu.cn` without being asked for password every time, you can do the following:

1. Generate a new keypair

   ```
   ssk-keygen -t rsa
   ```

2. Copy the keyfile to remote machine (`cs3.swfu.edu.cn`).

   ```
   ssh-copy-id username@cs3.swfu.edu.cn
   ```

3. Login to `cs3` without password prompt

   ```
   ssh username@cs3.swfu.edu.cn
   ```

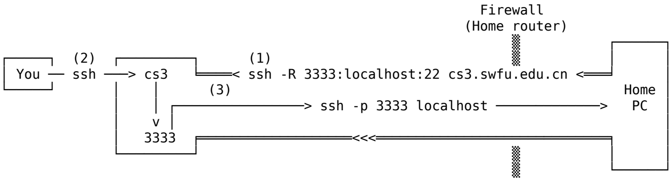
4. **CAUTION!** If you are doing the above steps at a lab PC, now you **must** remove the `key file`, otherwise everybody using this PC can login to your `cs3 account` without a password!

   ```
   rm -rf ~/.ssh
   ```

   This password-less setup should only be used within your own private computer, e.g. your laptop. **DO NOT USE IT AT ANY PUBLIC COMPUTER!!!**

2.4 Port forwarding (5 pts)

2.4.1 Reverse port forwarding



As long as you can login to `cs3`, this setup enables you to access your home PC from anywhere!

1. At your home PC, do

   ```
   ssh -R 3333:localhost:22 cs3user@cs3.swfu.edu.cn
   ```

   This will open up a reverse ssh tunnel to `cs3.swfu.edu.cn`.

2. At `cs3`, do

   ```
   ssh -p 3333 homeuser@localhost
   ```

   Now, a connection is made from `cs3:22` to `your-home-pc:3333`.

3. **Your task:** use `netstat` at both local and remote side to figure out the TCP connections in this setup.

2.4.2 Local port forwarding

```
┌─────────┐
│         │    (1)                       ┌─────┐
│   You   └══> ssh cs3 -L 3333:cs2:80 >══╛ cs3 │
│    │                                   │     │   ┌────────┐
│ (2)│  ┌───> curl -v http://localhost:3333 ───────> cs2:80 │
│    v  │                                │     │   │        │
│   3333  ╒═════════════>>>══════════════╕     │   └────────┘
│         │                              └─────┘
└─────────┘
```

1. At your PC (usually restricted), do

   ```
   ssh user@cs3.swfu.edu.cn -L 3333:cs2.swfu.edu.cn:80
   ```

   Local machine listens on port 3333, and forward traffic to `cs2` on port 80. That means you can open a web browser, and visit `http://localhost:3333`. You should see the same page as `http://cs2.swfu.edu.cn`

2. **Your task:** use `netstat` at both local and remote side to figure out the TCP connections in this setup.

2.4.3 References

- SSH Tips
- SSH: More than secure shell
- SSH Tricks
- SSH Port Forwarding
- SSH, The Secure Shell: The Definitive Guide — SSH Port Forwarding

2.5 Pair working with SSH+Tmux (5 pts)

Suppose Alice and Bob are both sitting in our A7 lab. And they're working on a cooperative project. Sometimes they have to edit a file, e.g. `helloworld.c` together. How? Very easy…

2.5.1 Case 1

If both Alice and Bob use the same username (e.g. `stud`) to work together,

1. Bob opens a terminal. At the command prompt, he types:

   ```
   tmux new -s pair
   ```

2. Alice logins to Bob's machine via SSH:

```
ssh stud@bob.ip.address
tmux a -t pair
```

3. Now, they're sharing the same tmux session, and can co-edit their `helloworld.c` in it.

2.5.2 Case 2

If Alice and Bob use different username, for example, they both have accounts in `cs3` server, and want to do co-working there, they can use a shared socket to achieve this.

1. Bob logins to `cs3`, and starts a tmux session with a shared socket.

   ```
   ssh bob@cs3.swfu.edu.cn
   tmux -S /tmp/bob new -s bob
   chmod 777 /tmp/bob
   ```

2. Alice ssh into `cs3`, and attach to Bob's tmux session

   ```
   ssh alice@cs3.swfu.edu.cn
   tmux -S /tmp/bob a -t bob
   ```

2.5.3 More

- `man ssh`
- `man tmux`
- Build a Command Line Remote Pairing Setup
- Remote Pairing With SSH, Tmux, and Vim
- A Simple Pair Programming Setup with SSH and Tmux
- Some people call me ”the remote pairing guy”…
- Googling `ssh tmux pair working`

Now, you are sitting in the lab. Please feel free to work together to get the following tasks done.

3 HTTP (15 pts)

3.1 Install Apache2

```
sudo aptitude install apache2
```

3.2 Play with it

Your tasks Create your own website

- How do I know my web server is running? (`nmap`, `systemctl status apache2`)
- How to configure it? (`/usr/share/doc/apache2/`, `/etc/apache2/`)
- Is my apache2 working well? (`/var/log/apache2/`)
- Where is my homepage? (`/var/www/`)
- How to write a homepage? (`/var/www/index.html`)
- How to give every user a homepage? (`~/public_html/index.html`)

4 Email (15 pts)

4.1 SMTP (8 pts)

4.1.1 Install Exim4

```
sudo aptitude install exim4
```

4.1.2 Play with it

Your tasks

- How do I know my SMTP server is running? (`nmap`, `systemctl status exim4`)
- How to configure it? (`/usr/share/doc/exim4/`, `/etc/exim4/`, `sudo dpkg-reconfigure exim4-config`)
- Is my exim4 working well? (`/var/log/exim4/`)
- How to send/receive emails? (`mail`, `mutt`, `nc server 25`)

4.2 POP3/IMAP4 (7 pts)

4.2.1 Install Dovecot roundcube

```
sudo aptitude install dovecot-imapd dovecot-pop3d roundcube
```

4.2.2 Play with it

Your tasks

- How do I know my POP3/IMAP4 server is running? (`nmap`, `systemctl status dovecot`)
- How to configure it? (`/usr/share/doc/dovecot*/`, `/etc/dovecot/`, `/usr/share/doc/roundcube-core`, `/etc/roundcube`)
- Is my dovecot working well? (`/var/log/mail.*/`)
- How to send/receive emails? (`/usr/share/doc/roundcube-core/`)

5 FTP (15 pts)

5.1 Install vsftpd lftp

```
sudo aptitude install vsftpd lftp
```

5.2 Play with it

Your tasks

- How do I know my FTP server is running? (`nmap`, `systemctl status vsftpd`)
- How to configure it? (`/usr/share/doc/vsftpd/`, `/etc/vsftpd.conf`)
- Is my vsftpd working well? (`/var/log/vsftpd.log`)
- How to transfer files? (`lftp`)

6 IPTables (30 pts)

6.1 Writing a simple rule set

If you try the following commands:

```
$ sudo iptables -P INPUT ACCEPT
$ sudo iptables -F
$ sudo iptables -A INPUT -i lo -j ACCEPT
$ sudo iptables -A INPUT -m state --state ESTABLISHED,RELATED -
j ACCEPT
$ sudo iptables -A INPUT -p tcp --dport 22 -j ACCEPT
$ sudo iptables -P INPUT DROP
$ sudo iptables -P FORWARD DROP
$ sudo iptables -P OUTPUT ACCEPT
$ sudo iptables -L -v
```

You will get the following output:

```
Chain INPUT (policy DROP 0 packets, 0 bytes)
pkts bytes target   prot opt in   out  source     destination
 0     0   ACCEPT   all  --  lo   any  anywhere   anywhere
 0    0  ACCEPT  all -- any any anywhere  anywhere  state RELATED,ESTABLISHED
 0    0  ACCEPT  tcp  -- any any anywhere  anywhere  tcp dpt:ssh
Chain FORWARD (policy DROP 0 packets, 0 bytes)
pkts bytes target    prot opt in    out    source   destination
Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
pkts bytes target    prot opt in    out    source   destination
```

Read the following short tutorial to know why:

- Writing a simple rule set

6.2 Your tasks

1. How to block all connections from your next desk?
2. How to block only SSH connections from your next desk?
3. How to block all other than SSH connections from your next desk?

6.3 References

- Iptables Howto
- The Beginner’ s Guide to iptables, the Linux Firewall
- google `iptables tutorial`