Week 1 - POSIX systems

Sockets

Sockets provide a programming interface for network communications.
They allow applications to **send and receive data through network protocols**

- such as TCP (Transmission Control Protocol) and UDP (User Datagram Protocol).

In the context of POSIX, sockets are designed to be **portable across different operating systems**, ensuring that applications written for one POSIX-compliant system can work on another without significant changes.

- POSIX stands for **Portable Operating System Interface**

POSIX systems

（Portable Operating System Interface）

- it refers to a family of **standards** specified by the IEEE(Institute of Electrical and Electronics Engineers) for **maintaining compatibility between operating systems**.
  不同操作系统间的兼容性
- defines the **application programming interface (API)**, along with **command line shells** and **utility interfaces**, for software compatibility with variants of Unix and other operating systems.
- a set of standards that helps maintain consistency in the behavior of **operating systems**, ensuring that programs written for one POSIX-compliant system can be easily ported to another POSIX-compliant system without major changes.

SSH

SSH, or Secure Shell, is a network protocol that **provides administrators with a secure way to access a remote computer**.

- commonly used to *log into* servers
- execute commands *remotely*
- *move files* from one machine to another over the internet
- provides strong *authentication* and secure communications over insecure channels, like the internet.

Key features of SSH include:

1. **Encryption**: SSH encrypts data sent over the network, protecting it from eavesdropping(窃听), interception(截取), and man-in-the-middle attacks.

2. **Authentication**: It ensures that the connection is made to the intended server and verifies the identity of the user accessing the server.
3. **Integrity**: SSH ensures that the data is not tampered(篡改) with during transmission.
4. **Tunneling and Port Forwarding**: It can be used to tunnel other network protocols securely, enabling secure use of otherwise insecure network applications.
5. **SFTP and SCP**: SSH is used as the basis for secure file transfer protocols like SFTP (Secure File Transfer Protocol) and SCP (Secure Copy Protocol).

SSH is widely used by system administrators and IT professionals for **managing systems and applications remotely**, allowing for secure system maintenance and troubleshooting.

```
SSH client -- encrypted connection --> log into SSH server --> execute commands / administer systems / transfer files / manage the systems remotely
```

```
Last login: Fri Jan 19 13:17:54 on ttys008
(base) ishtar@eduroam-80-201 ~ % ssh localhost
ssh: connect to host localhost port 22: Connection refused
(base) ishtar@eduroam-80-201 ~ % ssh bx23489@seis.bris.ac.uk
The authenticity of host 'seis.bris.ac.uk (137.222.0.126)' can't be established.
RSA key fingerprint is SHA256:9UB1OXr9MA+o1Y05/oUmjApUfHNaI4x78zTpI39se10.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'seis.bris.ac.uk' (RSA) to the list of known hosts.
bx23489@seis.bris.ac.uk's password:
Creating home directory for bx23489.
-bash: warning: setlocale: LC_CTYPE: cannot change locale (UTF-8): No such file or directory
[bx23489@seis-shell ~]$ uname -a
Linux seis-shell 3.10.0-1160.105.1.el7.x86_64 #1 SMP Thu Dec 7 15:39:45 UTC 2023 x86_64 x86_64 x86_64 GNU/Linux
[bx23489@seis-shell ~]$ ssh rd-mvb-linuxlab.bristol.ac.uk
The authenticity of host 'rd-mvb-linuxlab.bristol.ac.uk (137.222.3.13)' can't be established.
ED25519 key fingerprint is SHA256:kJHtwnmD9fumG5o+mto9O6bX9PpapuaqtatQiUMV1s8.
ED25519 key fingerprint is MD5:03:61:18:ae:4f:64:8e:fc:37:e3:e8:8e:44:fa:46:13.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'rd-mvb-linuxlab.bristol.ac.uk,137.222.3.13' (ED25519) to the list of known hosts.
bx23489@rd-mvb-linuxlab.bristol.ac.uk's password:
  ********************************************************************************
  *                                                                              *
  *   IT075745.WKS.BRIS.AC.UK                                                    *
  *                                                                              *
  *   Queries about this system to: servicedesk@bris.ac.uk                       *
  *                                                                              *
  *                                                                              *
  *   This server provides the following services:                               *
  *     - (Documentation at https://www.bristol.ac.uk/it-eng)                    *
  *     - Additional software via the "modules" command                          *
  *     - Home space (backed up to tape) from homes.fen                          *
  *     - SSH access (restricted to particular users/groups)                     *
  *     - Scratch space by request (not backed up!) on /scratch/<username>       *
  *     - mosh access (restricted to particular users/groups)                    *
  *                                                                              *
  ********************************************************************************

  Vagrant configuration will be stored in /tmp/run.vagrant/bx23489.DQ1Ke1Yt
  and VirtualBox configuration in /tmp/run.virtualbox/bx23489.Puzahc50/.config

[bx23489@it075745 ~]$ whoami
bx23489
[bx23489@it075745 ~]$ uname -a
Linux it075745.wks.bris.ac.uk 4.18.0-513.11.1.el8_9.x86_64 #1 SMP Wed Jan 10 22:58:54 UTC 2024 x86_64 x86_64 x86_64 GNU/Linux
[bx23489@it075745 ~]$ exit
logout
Connection to rd-mvb-linuxlab.bristol.ac.uk closed.
[bx23489@seis-shell ~]$ exit
logout
Connection to seis.bris.ac.uk closed.
(base) ishtar@eduroam-80-201 ~ %
```

Key Pair

The relationship between the **public key** and **private key** is such that data encrypted with the public key can only be decrypted with the private key.

key-based login

e.g.

Q18. Alice is trying to set up key-based login on a service she currently accesses via using `ssh` and entering her password. Here are listings of her `.ssh` directory on her local machine:

```
-rw------- 1 alice alice 389 Jan 19 10:56 authorized_keys
-rw------- 1 alice alice 395 Feb 21 13:03 id_rsa
-rw-r--r-- 1 alice alice 395 Feb 21 13:03 id_rsa.pub
-rw-r--r-- 1 alice alice 225 Feb 21 13:01 known_hosts
```

And on the server:

```
-rw-r--r-- 1 alice alice 395 Feb 21 13:21 id_rsa.pub
-rw-r--r-- 1 alice alice 225 Jan 19 11:02 known_hosts
```

There's a problem evident with her current setup. Identify which file's presence, abscence or visible details indicates the problem.

A. The problem lies with 'authorized_keys' on the local machine.

B. The problem lies with 'authorized_keys' on the server.

C. The problem lies with 'known_hosts' on the local machine.

D. The problem lies with 'id_rsa.pub' on the server.

File	`known_hosts`	`authorized_keys`	`id_rsa`	id_rsa.pub
Location	client	server	client / private key	server / public key
Content	**keep a record of the SSH hosts** the user has connected to and verified	to **store the public keys** that are allowed to log in	**securely** stored on the **client**'s machine and is never shared or transmitted.	The server **needs the public key to be added to the authorized_keys file** for the user, not the id_rsa.pub file itself.

File permission

Question 5 2 out of 2 points

Given a file that shows up as follows in ls -l:

-rw-rw-r-- 1 Breda staff 1024 Jan 1 10:01 logfile

Suppose that group users contains Alice, Breda and Carole; group tech contains Alice and Breda; and group staff contains Breda and Carole.

What are the access rights for the three users on this file?

- Alice: **[a]**
- Breda: **[b]**
- Carole: **[c]**

In all cases, "execute" refers to running the program as ./logfile.

The first character `-` indicates that this is a regular file (not a directory, link, or other special types).

The next three characters `rw-` show that the **owner** of the file (which is "**Breda**" in this case) has read (`r`) and write (`w`) permissions on the file, but not execute (`x`) permissions.

The following three characters `rw-` indicate that **members of the file's group** (which is "**staff**") have read and write permissions as well.

The final three characters `r--` show that all **other users** have only read permission on the file.

- **Alice**: Although Alice is a member of both the "users" and "tech" groups, she is not a member of the "staff" group. The file belongs to the "staff" group, so she falls under the "other users" category. This means Alice has only read (`r`) access to the file.
- **Breda**: Breda is the owner of the file and also a member of the "staff" group. The owner permissions take precedence here, so Breda has both read and write (`rw`) access to the file.

- **Carole**: Carole is not the owner but is a member of the "staff" group. Therefore, Carole has the group permissions, which are read and write (`rw`) access.

first through seis to access the lab machine and login

access to file system
allow address different lab machine to log in

ssh

```
ssh rd-mvb-linuxlab.bristol.as.uk
```

vagrant
create another machine to log in (Virtual machine)

this means the shell can interpret certain characters in your commands and translate these into arguments.

Pipe

e.g. `echo "words in a string" | sed 's/in //' | head -1`

`| sed 's/in //'`
- The pipe `|` **takes the output of the previous command** (`echo "words in a string"`) and **uses it as input for the next command**
- `sed 's/in //'` uses the `sed` stream editor to perform a substitution on its input. The syntax `s/in //` tells `sed` to look for the first instance of the pattern " `in` " (note the space after "in") in each line of input and replace it with nothing (effectively deleting it)
- Week 3 Shell Scripting & Build Tools #sed

`| head -1`
- `head -1` takes the **first line of its input and outputs it**.
Since the input only has one line ("words a string"), it simply outputs that line.

Command Substitution

```
command2 $(command1)
  # capture the output of command1 to be used by command2
```

The shell executes the command inside the parentheses(`$(command`) in a sub-shell.
The **output** of the command, with any trailing newlines removed, is then **inserted into the original command line** in place of the substitution expression. This allows you to use command output as filenames, arguments, or even parts of other commands dynamically.

Shell Redirection

Redirect stdout

`> /dev/null` and `1> /dev/null` both discard **standard output**
`> /dev/null`
- This redirects the standard output (stdout) of a command to `/dev/null` , effectively discarding it.
- `/dev/null` is **a special file that discards all data written to it** (think of it as a data black hole)

`1>`

This is another way of specifying redirection of the standard output. The number `1` represents stdout.

`1> /dev/null` would do the same as `> /dev/null`, redirecting stdout to `/dev/null`

Redirect stderr

`2>`

- This redirects the standard error (stderr) to a specified location.
- The number `2` represents stderr.
- So, `2> /dev/null` redirects all **standard error** to `/dev/null`, discarding them

Combine

- When used in a full command like `command > /dev/null 2>&1`, it means "redirect stderr to the same place as stdout, which has been redirected to `/dev/null`."
  `2>&1` is used to **redirect stderr (2) to wherever stdout (1) is currently directed**
- Essentially, it merges both standard output and error streams into one (stdout) and then discards them if stdout is directed to `/dev/null`.

Common Linux Shell Commands

Package Manager

Install

```
sudo apt install <...>
```

`sudo` (superuser do) allows you to run a command as root, also known as the administrator or superuser. use sudo for system administration instead of logging in as root directly

- with `#` instead of `$` as prompt to warn you that you are working as root.

`apt` is the Debian package manager.

`install PACKAGE` adds a package, which means download and install it and all its dependencies

Update and Upgrade

```
sudo apt update
```

fetches the new package list from the repository. This way, apt can tell you if any packages have been updated to new versions since you last checked.

```
sudo apt upgrade
```

upgrades every package that you already have installed to the latest version in your local package list