Secure SHell (SSH)

Dr Peadar Grant

September 20, 2022

1 Shell

The shell is the program we normally interact with in a command-line interface. Examples: PowerShell, Bash, Korn Shell, C Shell. We will return to more details of shell interfaces later on.

1.1 Key concepts

Prompt shows when shell is waiting for input.

Current working directory where commands will read and write files relative to.

Path: list of folders searched for matching command name

1.2 Features

Shells normally provide:

History: list of previous commands recalled (usually the up arrow key).

Redirection using *operators*

1. Standard input to a file.
2. Standard input from a file.
3. Piping the standard output of one command to the standard input of another.

Scripting a sequence of commands to be written.

Variables to capture and recall information.

Control constructs including conditionals, loops, possibly exceptions.

Different shells vary as to what extent they implement these features.

1.3 Terminal

The shell itself is normally accessed by means of a terminal. This is the program we visually see like the PowerShell Application or XTerm in Linux.

2 Secure Shell

SSH is a way to for one computer to connect to another's command-line interface in a secure fashion. It is widely used both in cloud-based and non-cloud environments for remote access.

SSH clients are included in most common operating systems. You can also get SSH client apps for iOS and Android.

An SSH client connects to an SSH server. The SSH server normally makes the command-line interface of the OS available (e.g. bash, powershell):

- All modern UNIX/Linux operating systems come with SSH servers as standard.
- Windows 10 onwards and Windows Server now have SSH servers included but need some configuration to get working.

SSH is relatively easy to get started with — the complexity often comes later when features like key-based authentication, multi-factor authentication, port forwarding and other extras are employed.

2.1 SSH client

Most operating systems use the OpenSSH client, named `ssh`, that is available on the command-line. To connect to a remote machine, we simply supply its name or IP and the username to connect as:

```
# connect via IP
ssh peadar@192.168.0.1

# connect via name
ssh peadar@compute-server.dkit.ie

# connect using same username as on client
ssh 192.168.0.1
ssh compute-server.dkit.ie}
```

3 Key-based authentication

SSH key pairs are an alternative to a username/password. They consist of:

Private key kept on the client and securely stored.

Public key on the server(s) you want to log in to. (The public key can be freely shared around, even put up in public.)

3.1 Creating key pair (windows 10, mac, linux)

Key pairs are created on your own local client computer. Key pairs only need to be generated once. If you already have a key pair created, you can skip on ahead to ???.

To create a 4096-bit RSA key pair, in Powershell/Bash type:

```
ssh-keygen -t rsa -b 4096
```

You can optionally use a passphrase to encrypt the key pair or leave it blank for easier usage. The key pair is then stored in two files in your home directory (same for Mac, Linux, Windows). You can find them by changing into the `.ssh` directory and listing the contents of it:

```
cd .ssh
dir
```

From the directory listing:

```
    Directory: C:\Users\peadar\.ssh


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
-a----       16/10/2020     15:19           3243 id_rsa
-a----       16/10/2020     15:19            749 id_rsa.pub
-a----       16/10/2020     15:32            176 known_hosts
```

The public key is stored in `id_rsa.pub`. The private key is stored in `id_rsa`.

You can of course copy these files to/from a memory stick or online storage. Remember though that if your private key is compromised, anybody can use it.

3.2 Connecting over SSH

In PowerShell/Bash we can use the SSH command to connect to the SSH server on a remote host. This will then present us with a new shell on the remote computer (Bash for Linux/UNIX, PowerShell for Windows). By default, SSH will try all private keys so we don't need to specify which.

```
ssh student@$publicIp
```

The IP addresses of the servers will be given during the lab. The first time you connect to a host you'll get a warning:

```
The authenticity of host '54.78.220.233 (54.78.220.233)' can't be established.
ECDSA key fingerprint is SHA256:8omkD5RLibZNgJJ/B7MAnL7IbEcrmCmIWFdQXbjJf60.
Are you sure you want to continue connecting (yes/no)?
```

Just type `yes` here. Your local SSH client is just confirming it hasn't seen this machine before. If a different key fingerprint shows for the same IP you'll get a warning, which means a host has been changed for another. If you see something like the following then you're connected:

```
       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
2 package(s) needed for security, out of 13 available
Run "sudo yum update" to apply all updates.
[student@ip-10-0-1-80 ~]$
```