Name: Emmanuel Luis B. Cuyugan	Date Performed: 16/01/2024
Course/Section: CPE232/CPE31S1	Date Submitted: 16/01/2024
Instructor: DR. JONATHAN V. TAYLAR	Semester and SY: 2nd 2023-2024

Activity 1: Configure Network using Virtual Machines

1. **Objectives:**

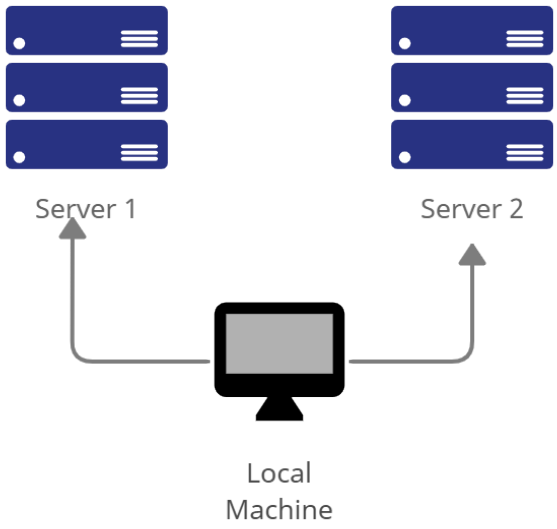
1.1. Create and configure Virtual Machines in Microsoft Azure or VirtualBox

1.2. Set-up a Virtual Network and Test Connectivity of VMs

2. **Discussion:**

Network Topology:

Assume that you have created the following network topology in Virtual Machines, *provide screenshots for each task*. (Note: *it is assumed that you have the prior knowledge of cloning and creating snapshots in a virtual machine*).



Task 1: Do the following on Server 1, Server 2, and Local Machine. In editing the file using nano command, press control + O to write out (save the file). Press enter when asked for the name of the file. Press control + X to end.

1. Change the hostname using the command *sudo nano /etc/hostname*

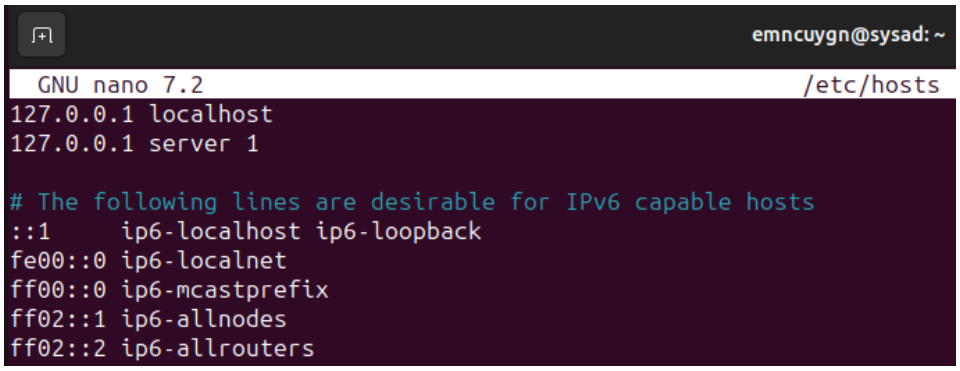
1.1 Use server1 for Server 1

1.2 Use server2 for Server 2

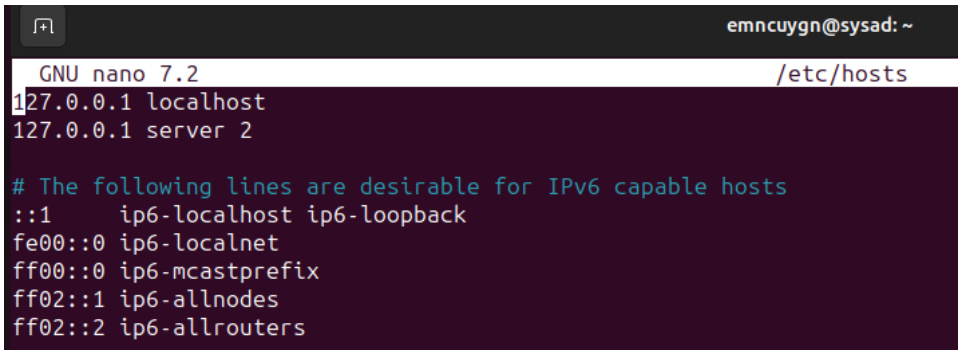
1.3 Use workstation for the Local Machine

2. Edit the hosts using the command *sudo nano /etc/hosts*. Edit the second line.

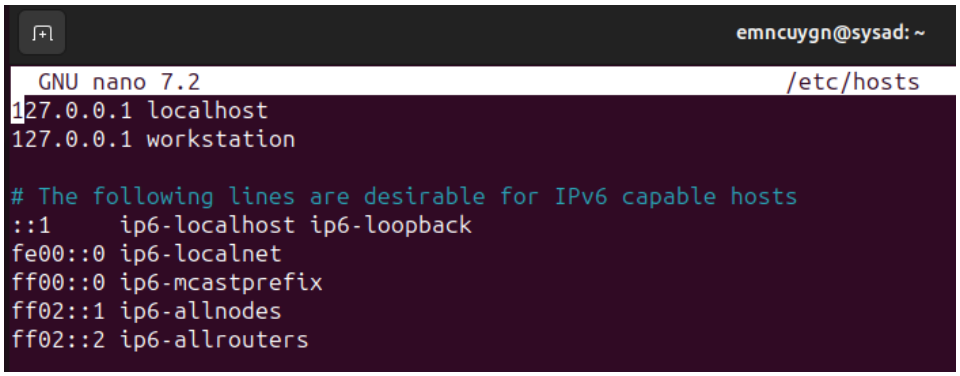
2.1 Type 127.0.0.1 server 1 for Server 1



2.2 Type 127.0.0.1 server 2 for Server 2

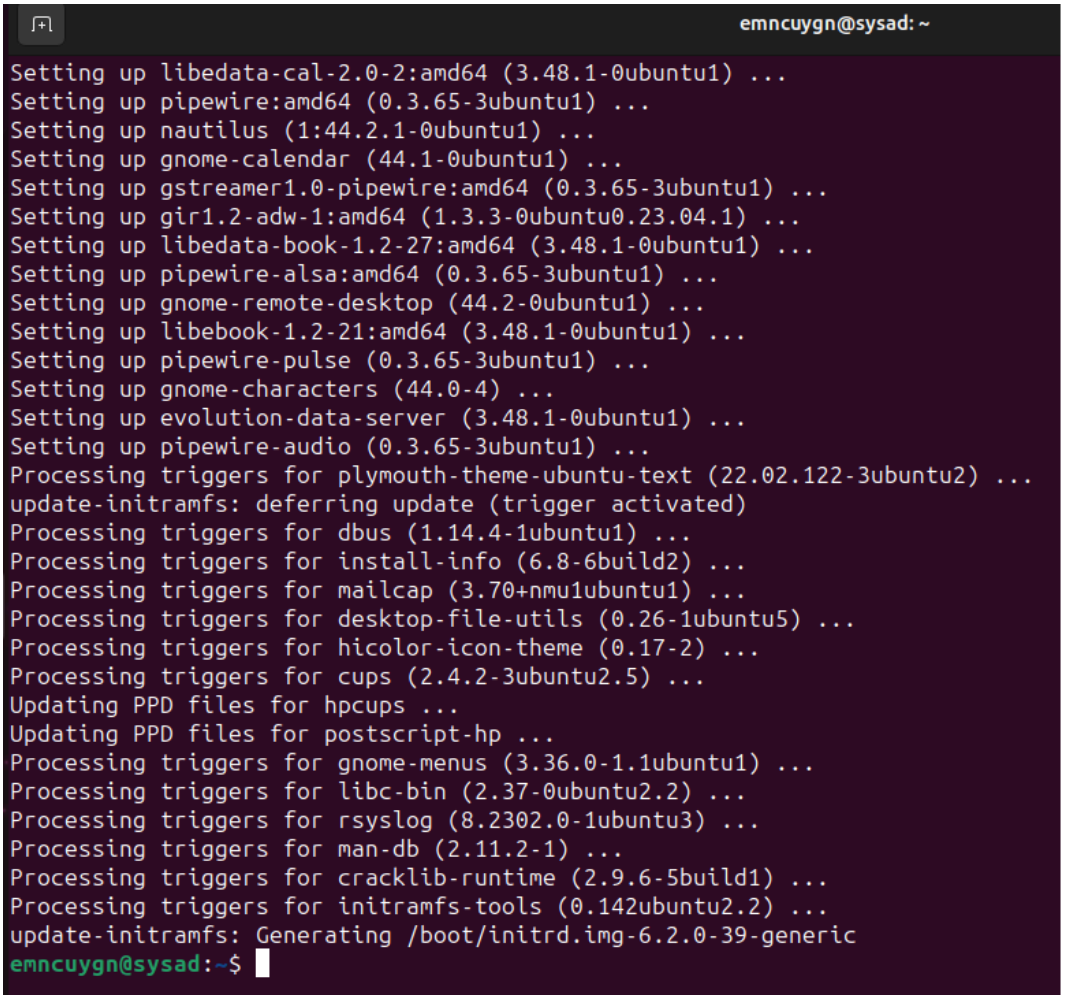


2.3 Type 127.0.0.1 workstation for the Local Machine



Task 2: Configure SSH on Server 1, Server 2, and Local Machine. Do the following:

1. Upgrade the packages by issuing the command *sudo apt update* and *sudo apt upgrade* respectively.



2. Install the SSH server using the command *sudo apt install openssh-server*.

```
emncuygn@sysad: ~
B]
Get:2 http://ph.archive.ubuntu.com/ubuntu lunar-updates/main amd64 openssh-server amd64 1
Get:3 http://ph.archive.ubuntu.com/ubuntu lunar-updates/main amd64 ncurses-term all 6.4-2
Get:4 http://ph.archive.ubuntu.com/ubuntu lunar/main amd64 ssh-import-id all 5.11-0ubuntu
Fetched 751 kB in 3s (270 kB/s)
Preconfiguring packages ...
Selecting previously unselected package openssh-sftp-server.
(Reading database ... 203822 files and directories currently installed.)
Preparing to unpack .../openssh-sftp-server_1%3a9.0p1-1ubuntu8.7_amd64.deb ...
Unpacking openssh-sftp-server (1:9.0p1-1ubuntu8.7) ...
Selecting previously unselected package openssh-server.
Preparing to unpack .../openssh-server_1%3a9.0p1-1ubuntu8.7_amd64.deb ...
Unpacking openssh-server (1:9.0p1-1ubuntu8.7) ...
Selecting previously unselected package ncurses-term.
Preparing to unpack .../ncurses-term_6.4-2ubuntu0.1_all.deb ...
Unpacking ncurses-term (6.4-2ubuntu0.1) ...
Selecting previously unselected package ssh-import-id.
Preparing to unpack .../ssh-import-id_5.11-0ubuntu1_all.deb ...
Unpacking ssh-import-id (5.11-0ubuntu1) ...
Setting up openssh-sftp-server (1:9.0p1-1ubuntu8.7) ...
Setting up openssh-server (1:9.0p1-1ubuntu8.7) ...

Creating config file /etc/ssh/sshd_config with new version
Synchronizing state of ssh.service with SysV service script with /lib/systemd/systemd-sys
Executing: /lib/systemd/systemd-sysv-install disable ssh
Created symlink /etc/systemd/system/sockets.target.wants/ssh.socket → /lib/systemd/system
rescue-ssh.target is a disabled or a static unit, not starting it.
Setting up ssh-import-id (5.11-0ubuntu1) ...
Setting up ncurses-term (6.4-2ubuntu0.1) ...
Processing triggers for man-db (2.11.2-1) ...
Processing triggers for ufw (0.36.1-4.1ubuntu0.1) ...
emncuygn@sysad:~$
```

3. Verify if the SSH service has started by issuing the following commands:
   3.1 *sudo service ssh start*
   3.2 *sudo systemctl status ssh*

```
emncuygn@sysad:~$ sudo service ssh start
emncuygn@sysad:~$ sudo systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
     Loaded: loaded (/lib/systemd/system/ssh.service; disabled; preset: enabled)
    Drop-In: /etc/systemd/system/ssh.service.d
             └─00-socket.conf
     Active: active (running) since Tue 2024-01-16 18:31:14 PST; 4s ago
TriggeredBy: ● ssh.socket
       Docs: man:sshd(8)
             man:sshd_config(5)
    Process: 27106 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
   Main PID: 27107 (sshd)
      Tasks: 1 (limit: 4599)
     Memory: 1.7M
        CPU: 13ms
     CGroup: /system.slice/ssh.service
             └─27107 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

Jan 16 18:31:14 sysad systemd[1]: Starting ssh.service - OpenBSD Secure Shell server...
Jan 16 18:31:14 sysad sshd[27107]: Server listening on :: port 22.
Jan 16 18:31:14 sysad systemd[1]: Started ssh.service - OpenBSD Secure Shell server.
```

4. Configure the firewall to all port 22 by issuing the following commands:
   4.1 *sudo ufw allow ssh*
   4.2 *sudo ufw enable*
   4.3 *sudo ufw status*

```
emncuygn@sysad:~$ sudo ufw allow ssh
Rules updated
Rules updated (v6)
emncuygn@sysad:~$ sudo ufw enable
Firewall is active and enabled on system startup
emncuygn@sysad:~$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
22/tcp                     ALLOW       Anywhere
22/tcp (v6)                ALLOW       Anywhere (v6)
```

Task 3: Verify network settings on Server 1, Server 2, and Local Machine. On each device, do the following:

1. Record the ip address of Server 1, Server 2, and Local Machine. Issue the command *ifconfig* and check network settings. Note that the ip addresses of all the machines are in this network 192.168.56.XX.
   1.1 Workstation IP address: 192.168.56.**105**
   1.2 Server 1 IP address: 192.168.56.**106**
   1.3 Server 2 IP address: 192.168.56.**104**
2. Make sure that they can ping each other.
   2.1 Connectivity test for Local Machine 1 to Server 1: ☐ Successful ☐ Not Successful

```
emncuygn@workstation:~$ ping 192.168.56.106
PING 192.168.56.106 (192.168.56.106) 56(84) bytes of data.
64 bytes from 192.168.56.106: icmp_seq=1 ttl=64 time=0.954 ms
64 bytes from 192.168.56.106: icmp_seq=2 ttl=64 time=0.825 ms
64 bytes from 192.168.56.106: icmp_seq=3 ttl=64 time=2.81 ms
64 bytes from 192.168.56.106: icmp_seq=4 ttl=64 time=0.503 ms
64 bytes from 192.168.56.106: icmp_seq=5 ttl=64 time=0.594 ms
64 bytes from 192.168.56.106: icmp_seq=6 ttl=64 time=0.869 ms
64 bytes from 192.168.56.106: icmp_seq=7 ttl=64 time=9.52 ms
64 bytes from 192.168.56.106: icmp_seq=8 ttl=64 time=0.558 ms
64 bytes from 192.168.56.106: icmp_seq=9 ttl=64 time=3.10 ms
64 bytes from 192.168.56.106: icmp_seq=10 ttl=64 time=2.21 ms
```

2.2 Connectivity test for Local Machine 1 to Server 2: ☐ Successful ☐ Not Successful

```
emncuygn@workstation:~$ ping 192.168.56.104
PING 192.168.56.104 (192.168.56.104) 56(84) bytes of data.
64 bytes from 192.168.56.104: icmp_seq=1 ttl=64 time=1.07 ms
64 bytes from 192.168.56.104: icmp_seq=2 ttl=64 time=0.963 ms
64 bytes from 192.168.56.104: icmp_seq=3 ttl=64 time=0.422 ms
64 bytes from 192.168.56.104: icmp_seq=4 ttl=64 time=0.392 ms
64 bytes from 192.168.56.104: icmp_seq=5 ttl=64 time=0.669 ms
64 bytes from 192.168.56.104: icmp_seq=6 ttl=64 time=1.29 ms
64 bytes from 192.168.56.104: icmp_seq=7 ttl=64 time=3.16 ms
64 bytes from 192.168.56.104: icmp_seq=8 ttl=64 time=0.968 ms
64 bytes from 192.168.56.104: icmp_seq=9 ttl=64 time=0.709 ms
64 bytes from 192.168.56.104: icmp_seq=10 ttl=64 time=1.47 ms
```

2.3 Connectivity test for Server 1 to Server 2: ☐ Successful ☐ Not Successful

```
emncuygn@server1:~$ ping 192.168.56.104
PING 192.168.56.104 (192.168.56.104) 56(84) bytes of data.
64 bytes from 192.168.56.104: icmp_seq=1 ttl=64 time=1.27 ms
64 bytes from 192.168.56.104: icmp_seq=2 ttl=64 time=0.712 ms
64 bytes from 192.168.56.104: icmp_seq=3 ttl=64 time=0.362 ms
64 bytes from 192.168.56.104: icmp_seq=4 ttl=64 time=1.06 ms
64 bytes from 192.168.56.104: icmp_seq=5 ttl=64 time=0.385 ms
64 bytes from 192.168.56.104: icmp_seq=6 ttl=64 time=0.837 ms
64 bytes from 192.168.56.104: icmp_seq=7 ttl=64 time=0.416 ms
64 bytes from 192.168.56.104: icmp_seq=8 ttl=64 time=0.387 ms
64 bytes from 192.168.56.104: icmp_seq=9 ttl=64 time=0.692 ms
64 bytes from 192.168.56.104: icmp_seq=10 ttl=64 time=0.691 ms
```

Task 4: Verify SSH connectivity on Server 1, Server 2, and Local Machine.

1. On the Local Machine, issue the following commands:

1.1 ssh username@ip_address_server1 for example, *ssh jvtaylar@192.168.56.120*

```
emncuygn@workstation:~$ ssh emncuygn@192.168.56.106
The authenticity of host '192.168.56.106 (192.168.56.106)' can't be established.
ED25519 key fingerprint is SHA256:CLU0Y/tCV4D1Wxip0r2fDT12/N9G8AEA4AhUsKG7g4k.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.56.106' (ED25519) to the list of known hosts.
emncuygn@192.168.56.106's password:
```

1.2 Enter the password for server 1 when prompted

```
emncuygn@workstation:~$ ssh emncuygn@192.168.56.106
The authenticity of host '192.168.56.106 (192.168.56.106)' can't be established.
ED25519 key fingerprint is SHA256:CLU0Y/tCV4D1Wxip0r2fDT12/N9G8AEA4AhUsKG7g4k.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.56.106' (ED25519) to the list of known hosts.
emncuygn@192.168.56.106's password:
Welcome to Ubuntu 23.04 (GNU/Linux 6.2.0-39-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

0 updates can be applied immediately.


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.
```

1.3 Verify that you are in server 1. The user should be in this format user@server1. For example, *jvtaylar@server1*

```
emncuygn@server1:~$
```

2. Logout of Server 1 by issuing the command *control + D.*

```
emncuygn@server1:~$
logout
Connection to 192.168.56.106 closed.
```

3. Do the same for Server 2.

```
emncuygn@workstation:~$ ssh emncuygn@192.168.56.104
The authenticity of host '192.168.56.104 (192.168.56.104)' can't be established.
ED25519 key fingerprint is SHA256:CLU0Y/tCV4D1Wxip0r2fDT12/N9G8AEA4AhUsKG7g4k.
This host key is known by the following other names/addresses:
    ~/.ssh/known_hosts:1: [hashed name]
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.56.104' (ED25519) to the list of known hosts.
emncuygn@192.168.56.104's password:
Permission denied, please try again.
emncuygn@192.168.56.104's password:
Welcome to Ubuntu 23.04 (GNU/Linux 6.2.0-39-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

0 updates can be applied immediately.


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

emncuygn@server2:~$
logout
Connection to 192.168.56.104 closed.
```

4. Edit the hosts of the Local Machine by issuing the command *sudo nano /etc/hosts.* Below all texts type the following:

4.1 IP_address server 1 (provide the ip address of server 1 followed by the hostname)

4.2 IP_address server 2 (provide the ip address of server 2 followed by the hostname)

```
emncuygn@workstation: ~
  GNU nano 7.2                                  /etc/hosts
127.0.0.1 localhost
127.0.0.1 workstation

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters

192.168.56.106 server1
192.168.56.104 server2
```

4.3 Save the file and exit.

5. On the local machine, verify that you can do the SSH command but this time, use the hostname instead of typing the IP address of the servers. For example, try to do *ssh jvtaylar@server1*. Enter the password when prompted. Verify that you have entered Server 1. Do the same for Server 2.

```
emncuygn@workstation:~$ ssh emncuygn@server1
The authenticity of host 'server1 (192.168.56.106)' can't be established.
ED25519 key fingerprint is SHA256:CLU0Y/tCV4D1Wxip0r2fDT12/N9G8AEA4AhUsKG7g4k.
This host key is known by the following other names/addresses:
    ~/.ssh/known_hosts:1: [hashed name]
    ~/.ssh/known_hosts:4: [hashed name]
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'server1' (ED25519) to the list of known hosts.
emncuygn@server1's password:
Welcome to Ubuntu 23.04 (GNU/Linux 6.2.0-39-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

0 updates can be applied immediately.

New release '23.10' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Tue Jan 16 18:58:49 2024 from 192.168.56.105
emncuygn@server1:~$
```

```
emncuygn@server1:~$
logout
Connection to server1 closed.
emncuygn@workstation:~$ ssh emncuygn@server2
The authenticity of host 'server2 (192.168.56.104)' can't be established.
ED25519 key fingerprint is SHA256:CLU0Y/tCV4D1Wxip0r2fDT12/N9G8AEA4AhUsKG7g4k.
This host key is known by the following other names/addresses:
    ~/.ssh/known_hosts:1: [hashed name]
    ~/.ssh/known_hosts:4: [hashed name]
    ~/.ssh/known_hosts:5: [hashed name]
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'server2' (ED25519) to the list of known hosts.
emncuygn@server2's password:
Welcome to Ubuntu 23.04 (GNU/Linux 6.2.0-39-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

0 updates can be applied immediately.

New release '23.10' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Tue Jan 16 19:03:44 2024 from 192.168.56.105
emncuygn@server2:~$
```

Reflections:

Answer the following:

1. How are we able to use the hostname instead of IP address in SSH commands?
    - **We are able to use the hostname instead of the IP address in SSH commands because the network itself with its nodes are configured to be connected to one another. Then after that, when knowing the IP addresses of the different nodes, we pinged each nodes to each other to know if it will work. Once confirmed, editing /etc/hosts file with sudo nano**

and adding the IP address and the hostname of both servers to the /etc/hosts of the main node, we can execute the command with just the hostname instead of the IP address.

2. How secured is SSH?

   - **SSH is a cryptographic network protocol for operating network services securely over an unsecured network. It uses various encryption algorithms to protect the data in transit. SSH is widely used and proven secure for everyday use. However, security also depends on verifying the identity of the server you are connecting to, by checking its fingerprint the first time you connect.**