# Remote Calculator + Expect + Cron job + Email notice

Long Nguyen Hoang
19578

# What Is Secure Shell (ssh)?

Secure Shell (SSH) is a popular networking protocol that lets us access a remote computer over an insecure network such as the Internet.

In this tutorial, we'll dive into it and explore various aspects of it.

Secure Shell or Secure Socket Shell is a network protocol. It is an application layer protocol that is in the 7th later of the Open Systems Interconnection (OSI) network model. It also refers to the suite of utilities that implements the SSH protocol.

Secure Shell also supports both password and key-based authentication. Password-based authentication let users provide username and password to authenticate to the remote server. A key-based authentication allows users to authenticate through a key-pair. The key pairs are two cryptographically secure keys for authenticating a client to a Secure Shell server.

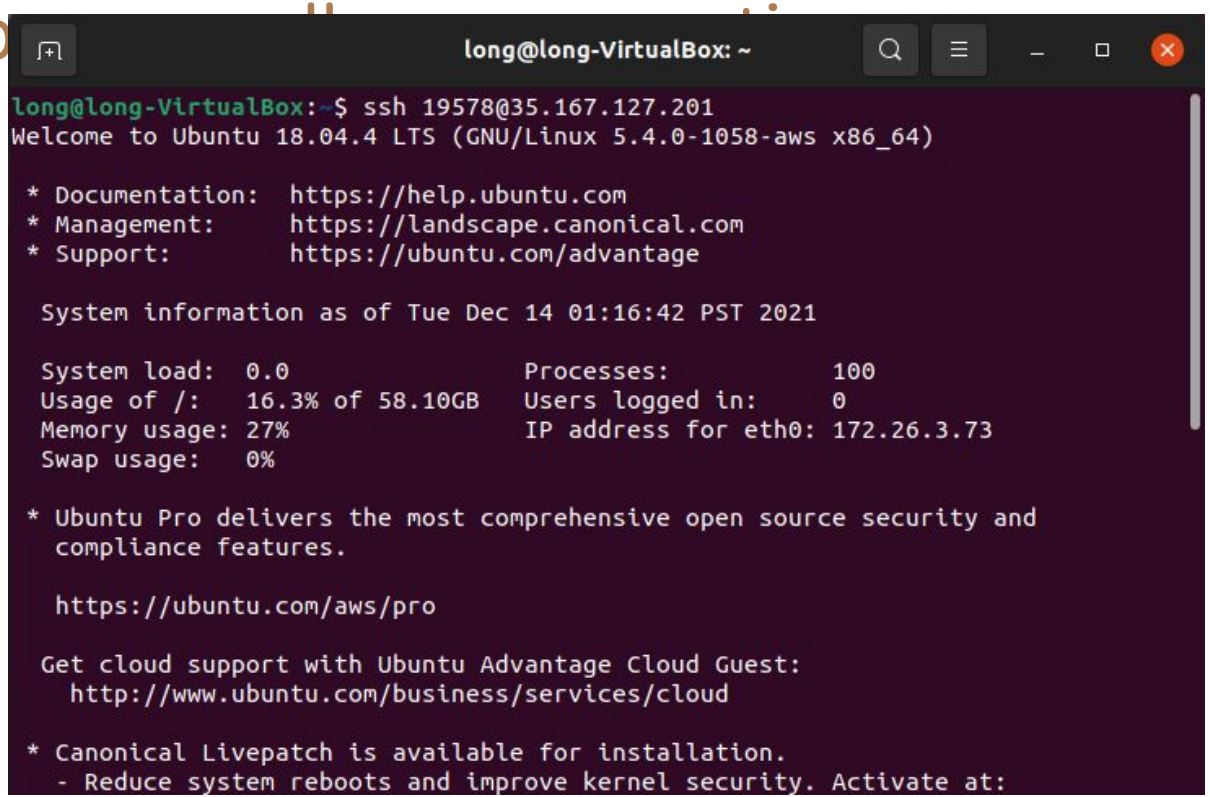
Furthermore, the Secure Shell protocol also encrypts data communication between two computers. It is extensively used to communicate with a remote computer over the Internet.

# SSH Setup

```
long@long-VirtualBox:~$ ssh-keygen -t dsa -f .ssh/id_dsa
Generating public/private dsa key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in .ssh/id_dsa
Your public key has been saved in .ssh/id_dsa.pub
The key fingerprint is:
SHA256:+lc0MtJ5yb35PDWIzVwvK6xI5XUWkRXnRV+7Y/VceRI long@long-VirtualBox
The key's randomart image is:
+---[DSA 1024]----+
|               E=B|
|               .+B|
|          . o ooo*|
|         . = * .=*|
|        S..=*.=*+|
|       . o o.B=.+|
|      . . .o   =o|
|       o .. o ..o|
|        o... .  .|
+----[SHA256]-----+
```

```
long@long-VirtualBox:~/.ssh$ scp id_dsa.pub 19578@35.167.127.201:~/.ssh/id_dsa.p
ub
The authenticity of host '35.167.127.201 (35.167.127.201)' can't be established.
ECDSA key fingerprint is SHA256:jGUnk2PP6ZhQ8NhJycmvJ/DJqiKQiPAzmKyRoZrLuIw.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '35.167.127.201' (ECDSA) to the list of known hosts.
19578@35.167.127.201's password:
Permission denied, please try again.
19578@35.167.127.201's password:
scp: /home/19578/.ssh/id_dsa.pub: No such file or directory
long@long-VirtualBox:~/.ssh$ scp id_dsa.pub 19578@35.167.127.201:~/.ssh/id_dsa.p
ub
19578@35.167.127.201's password:
id_dsa.pub                                    100%  610    11.6KB/s   00:00
long@long-VirtualBox:~/.ssh$
```

SSH p


long@long-VirtualBox: ~
long@long-VirtualBox:~$ ssh 19578@35.167.127.201
Welcome to Ubuntu 18.04.4 LTS (GNU/Linux 5.4.0-1058-aws x86_64)
* Documentation: https://help.ubuntu.com
* Management: https://landscape.canonical.com
* Support: https://ubuntu.com/advantage
System information as of Tue Dec 14 01:16:42 PST 2021
System load: 0.0 Processes: 100
Usage of /: 16.3% of 58.10GB Users logged in: 0
Memory usage: 27% IP address for eth0: 172.26.3.73
Swap usage: 0%
* Ubuntu Pro delivers the most comprehensive open source security and compliance features.
https://ubuntu.com/aws/pro
Get cloud support with Ubuntu Advantage Cloud Guest:
http://www.ubuntu.com/business/services/cloud
* Canonical Livepatch is available for installation.
- Reduce system reboots and improve kernel security. Activate at:

# Interactive calculator running and code on server

```
19578@CS522:~/cs522$ ./calculate
N1:5
N2:4
Operation:+
Result:9
19578@CS522:~/cs522$
```

```
echo -n "N1:"
read N1

echo -n "N2:"
read N2

echo -n "Operation:"
read Operation

if [ "$Operation" = "+" ]
then
        (( result = N1 + N2))
elif [ "$Operation" = "-" ]
then
        ((result = N1 - N2))
elif [ "$Operation" = "*" ]
then
        ((result = N1 * N2))
elif [ "$Operation" = "/" ]
then
        ((result = N1 / N2))
else
        echo "Error: wrong input $Operation"
        exit 1
fi

echo "Result:$result"
```

# Expect script

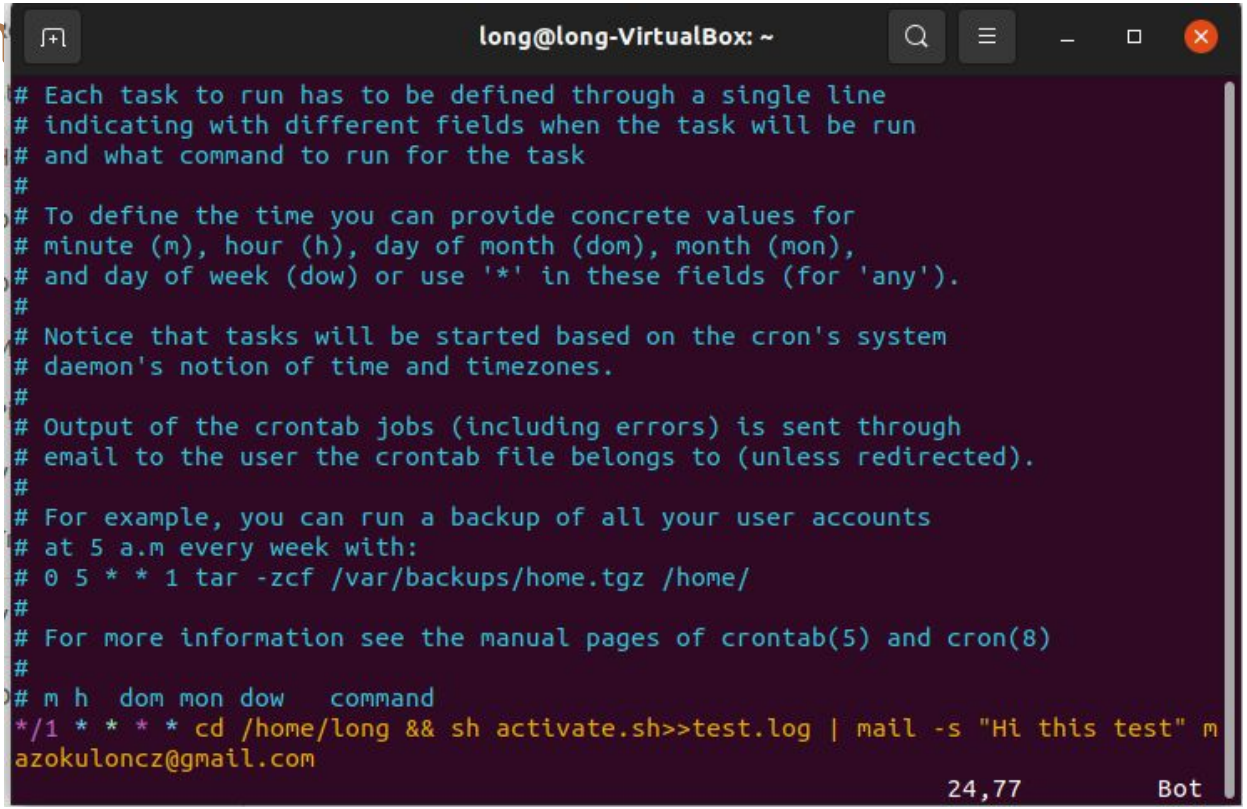
```
1 #!/usr/bin/expect --
2
3 set timeout -1
4 set N1 [lindex $argv 0]
5 set N2 [lindex $argv 1]
6 set Operation [lindex $argv 2]
7 spawn ssh 19578@35.167.127.201 /home/19578/cs522/calculate
8 expect "N1:" { send "$N1\r" }
9 expect "N2:" { send "$N2\r" }
10 expect "Operation:" { send -- "$Operation\r" }
11 expect " " { send "exit\r" }
12
```

# Shell script to automatically execute Expect

```
1 N1=`shuf -i 1-10 -n 1`
2 N2=`shuf -i 1-10 -n 1`
3
4 ans=`shuf -i 1-4 -n 1`
5 if [ "$ans" -eq 1 ]
6 then
7         Operation="+"
8 elif [ "$ans" -eq 2 ]
9 then
10        Operation="-"
11 elif [ "$ans" -eq 3 ]
12 then
13        Operation="*"
14 else
15        Operation="/"
16 fi
17 ./exp "$N1" "$N2" "$Operation"
```

```
long@long-VirtualBox:~$ ./activate.sh
spawn ssh 19578@35.167.127.201 /home/19578/cs522/calculate
N1:3
N2:2
Operation:*
Result:6
long@long-VirtualBox:~$
```

Cron


long@long-VirtualBox: ~
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
*/1 * * * * cd /home/long && sh activate.sh>>test.log | mail -s "Hi this test" m
azokuloncz@gmail.com
24,77
Bot