RHCEv8 Online Class 05062021 10:00pm
RHCSA-sysadmin
- Configuring and Securing SSH
- Analyzing and Storing Logs
- Archiving, Accurate TIME

---

## Configuring and Securing SSH

### How access to host

1-direct access
2-remote access
    2-1-cli
        **2-1-1-ssh**
        2-1-2-telnet
    2-2-gui
        2-2-1-tigerVNC, Xrdp, ....
3-web access
    3-1-cockpit

### ACCESSING THE REMOTE COMMAND LINE WITH SSH

### WHAT IS OPENSSH?

-OpenSSH implements the Secure Shell or SSH protocol in the RHEL systems.
-The SSH protocol enables systems to communicate in an encrypted and secure fashion over an insecure network.

### ssh info

**package:** openssh.x86_64, openssh-clients.x86_64, openssh-server.x86_64
**daemon:** sshd.service
**config file:** /etc/ssh/sshd_config
**port:** 22/tcp
**log:** /var/log/secure
**dir:** ~/.ssh

### Implement OpenSSH on linux

```
# yum install openssh* -y
# systemctl enable sshd.service
# systemctl start sshd.service
# systemctl status sshd.service
# firewall-cmd --permanent --add-port=22/tcp
or
# firewall-cmd --permanent --add-service=ssh
# firewall-cmd --reload
# firewall-cmd --list-all
# netstat -lnptu
tcp        0      0 0.0.0.0:22
tcp6       0      0 :::22
```

### How to config local DNS

over each host:

```
# vim /etc/hosts
<ip> <FQDN> <alias>
172.25.250.10   servera.lab.example.com servera
172.25.250.11   serverb.lab.example.com serverb
:wq!
```

### run ssh command

ssh connection from servera(ssh client) to serverb(ssh server)

```
servera   172.25.250.10      servera.lab.example.com
serverb   172.25.250.11      serverb.lab.example.com
# ssh <remote user>@<remote host name/ip>
# ssh root@172.25.250.11
# ssh -l root 172.25.250.11
# ssh -l root 172.25.250.11 -p <custom port>
# ssh root@172.25.250.11
# ssh root@172.25.250.11 <command>
# ssh -l root 172.25.250.11 <command>
to finish ssh connection
# exit
# press Ctrl+d
```

ssh methods
1-host-key authentication
2-key-based authentication

1-host-key authentication(def.)
public-key + password
from servera to serverb
[root@servera ~]# ls -a ~/
[root@servera ~]# ssh root@172.25.250.11
The authenticity of host '172.25.250.11 (172.25.250.11)' can't be established.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '172.25.250.11' (ECDSA) to the list of known hosts.
root@172.25.250.11's password: ******

when press yes, serverb public-key will transfer to servera stores in ~/.ssh/known_hosts and should type password

over serverb
# w ->shows who is connecting now with details
# tail /var/log/secure

2-key-based authentication (password less)
public-key + private-key
NOTE: it's one-way connection.

steps:
1-create public-key and private-key
2-keep private-key and transfer public-key to customers
3-make password-less connection

servera to serverb
1-create public-key and private-key
# ssh-keygen
Generating public/private rsa key pair.
press Enter
press Enter
press Enter
# ls -a ~/.ssh/
. .. id_rsa id_rsa.pub

2-keep private-key and transfer public-key to customers
[root@servera ~]# ssh-copy-id root@172.25.250.11

NOTE: when transfer pub-key content, this content stores where on serverb
[root@serverb ~]# cat ~/.ssh/authorized_keys

3-make password-less connection
[root@servera ~]# ssh root@172.25.250.11
[root@serverb ~]# exit
[root@servera ~]#

Harden ssh password-less connection with passphrase
[root@servera ~]# ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa): press Enter
Created directory '/root/.ssh'.
Enter passphrase (empty for no passphrase): coss@2021
Enter same passphrase again: coss@2021
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.

[root@servera ~]# ssh-copy-id root@172.25.250.11
[root@servera ~]# ssh root@172.25.250.11
Enter passphrase for key '/root/.ssh/id_rsa': coss@2021
[root@serverb ~]# exit
[root@servera ~]#

**How to save temporary** passphrase in to memory to prevent type in ssh time
**set**
[root@servera ~]# eval $(ssh-agent)
Agent pid 27066
[root@servera ~]# # ssh-add
Enter passphrase for /root/.ssh/id_rsa: **coss@2021**
Identity added: /root/.ssh/id_rsa (root@servera.lab.example.com)
[root@servera ~]# ssh root@172.25.250.11
[root@serverb ~]# exit
[root@servera ~]#

**remove**
[root@servera ~]# # ssh-add -D
All identities removed.
[root@servera ~]# ssh root@172.25.250.11
Enter passphrase for key '/root/.ssh/id_rsa': **coss@2021**
[root@serverb ~]# exit
[root@servera~]#

points
**1-prevent make ssh through root user**
[root@serverb ~]# vim /etc/ssh/sshd_config
PermitRootLogin no
:wq!
# systemctl reload sshd.service
yes->permit
no->doesn't permit
**verify**
# ssh root@172.25.250.11
Enter passphrase for key '/root/.ssh/id_rsa': **coss@2021**
root@172.25.250.11's password: **redhat**
Permission denied, please try again.

**show message to user before login**
[root@serverb ~]# vim /etc/issue
Welcome to ServerB
:wq!
# vim /etc/ssh/sshd_config
Banner /etc/issue
:wq!
# systemctl reload sshd.service
**verify**
# ssh root@172.25.250.11
# ssh root@172.25.250.11
\S
Kernel \r on an \m

Welcome to ServerB
Enter passphrase for key '/root/.ssh/id_rsa': **coss@2021**
[root@serverb ~]# exit

**show message to user after login**
[root@serverb ~]# vim /etc/motd
You are watching this message after login to sevrerb
if you are not authorized, please logout now!
:wq!
[root@serverb ~]# systemctl reload sshd.service
**verify**
[root@servera ~]# ssh root@172.25.250.11
Welcome to ServerB ->Banner message
Enter passphrase for key '/root/.ssh/id_rsa': **coss@2021**
You are watching this message after login to serverb ->motd message
if you are not authorized, please logout now! ->motd message **Message Of The Day**
[root@serverb ~]# exit

## Analyzing and Storing Logs

what is LOG?
log is type of report

**log on linux**

**syslog** protocol responsible to control logs on linux
1-rsyslog
2-journal log

**monitor logs**

1-offline
# tail /var/log/messages

2-online
# tail -f /var/log/secure

**read log file**

Jun 5 10:22:08 servera sshd[14182]: Server listening on 0.0.0.0 port 22.
Jun 5 10:22:08 ->timestamp
servera ->hostname
sshd[14182] ->process/pid
Server listening on 0.0.0.0 port 22 ->event

**log location on linux**

# ll /var/log/

| | |
|---|---|
| anaconda | ->log of linux installer |
| boot.log | ->OS boot procedure log |
| wtmp | ->log of success login |
| btmp | ->log of failed login |

# last ->log of success login
wtmp begins
# lastb ->log of failed login
btmp begins
# aureport
# aureport --success -l
# aureport --failed -l
# aureport --help

| | |
|---|---|
| chrony | ->ntp log |
| cron | ->scheduler log |
| firewalld | ->firewalld log |
| message | ->all services info levels log |
| secure | ->ssh log |
| tuned | ->tuned log |

**1-rsyslog**

**rsyslog info**

**package:** rsyslog.x86_64
**daemon:** rsyslog.service
**config file:** /etc/rsyslog.conf
**port:** 514 tcp/udp

**implement rsyslog**

# yum install rsyslog* -y
# yum update rsyslog* -y
# systemctl enable rsyslog.service
# systemctl start rsyslog.service
# systemctl status rsyslog.service

# vim /etc/rsyslog.conf
#### MODULES #### ->coifing host as rsyslog server
#### RULES #### ->manage rsyslog
# ### sample forwarding rule ### ->coifing host as rsyslog client

#### RULES ####  ->manage rsyslog
*.*
facility.priority
facility->services
priority->log level
0 emerg
1 alert
2 crit
3 err
4 warning
5 notice
6 info
7 debug

**ex:**
*.alert ->all services in alert level
cron.info ->cron daemon in info level
secure.* ->ssh service with all levels

**Implement rsyslog on linux**
servera rsyslog server 172.25.250.10
serverb rsyslog client 172.25.250.11
**implement**
servera
# vim /etc/rsyslog.conf
#### RULES ####
19 module(load="imudp") # needs to be done just once
20 input(type="imudp" port="514")
24 module(load="imtcp") # needs to be done just once
25 input(type="imtcp" port="514")
:wq!
# systemctl restart rsyslog.service
# firewall-cmd --permanent --add-port=514/tcp
# firewall-cmd --permanent --add-port=514/udp
# firewall-cmd --reload
# firewall-cmd --list-all
# netstat -lnptu
# tail -f /var/log/messages
Jun 5 12:22:06 serverb root[2652]: i'm from serverb
Jun 5 12:22:06 serverb root[2652]: i'm from serverb

serverb
# yum list rsyslog
# systemctl status rsyslog.service
# vim /etc/rsyslog.conf
# ### sample forwarding rule ###
*.* @172.25.250.10 ->upd protocols
*.* @@172.25.250.10 ->tcp protocols
:wq!
# systemctl restart rsyslog.service
# logger "i'm from serverb" ->when do something, host will send copy to rsyslog server too

**2-journal log**
-journal tightly works with systemd, systemd tightly works with kernel then journal tightly works with kernel and shows kernel logs
-but journal log, its temporary mean after each reboot journal log file will clear and again will generate and start store
-journal location and file:
/run/log/journal/55e2a69db34d4f76b8a0115088861896/system.journal

# journalctl
# journalctl -xn
# journalctl -n 4
# journalctl -p err
# journalctl -p info
# journalctl -f
# journalctl --since today
# journalctl --since "-1 hour"

**How to make** journal permanent?
steps:
# vim /etc/systemd/journald.conf
14 [Journal]
15 Storage=**persistent**
:wq!
# systemctl restart systemd-journald
# ls /run/log/

# ls /var/log/
journal
# du -h /var/log/journal/55e2a69db34d4f76b8a0115088861896/system.journal
8.1M /var/log/journal/55e2a69db34d4f76b8a0115088861896/system.journal

Storage=**persistent** ->make it persistent
**volatile** ->pure temporary
**auto** ->it depends on location
https://www.rsyslog.com/

MAINTAINING ACCURATE TIME
SETTING LOCAL CLOCKS AND TIME ZONES
-Correct synchronized system time is critical for log file analysis across multiple systems.
-The Network Time Protocol (NTP) is a standard way for machines to provide and obtain correct time information on the Internet.
# timedatectl
# timedatectl status
# timedatectl set-ntp <ntp-server ip>
# timedatectl list-timezones
# timedatectl list-timezones
# timedatectl set-timezone <new timezone>

CONFIGURING AND MONITORING CHRONYD
The chronyd service keeps the usually-inaccurate local hardware clock (RTC) on track by synchronizing it to the configured NTP servers.

**chrony info**
**package:** chrony.x86_64
**daemon:** chronyd.service
**config file:** /etc/chrony.conf
**port:** 123udp

**implement chrony on linux**
# yum install chrony* -y
# yum update chrony* -y
# systemctl enable chronyd.service
# systemctl start chronyd.service
# systemctl status chronyd.service
# timedatectl
System clock synchronized: yes
NTP service: active

**How to sync host with chrony server**
https://www.pool.ntp.org/zone/in
and find out your region ntp server list
# vim /etc/chrony.conf
hash unwanted lines about unrelated ntp servers addresse
server <ntp server name/ip> iburst
server 0.in.pool.ntp.org iburst
server 1.in.pool.ntp.org iburst
server 2.in.pool.ntp.org iburst
server 3.in.pool.ntp.org iburst
:wq!
# systemctl restart chronyd.service
# chronyc sources -v

burst ->when ntp server available 24*7
iburst ->when ntp server wont available 24*7 ->aggressive

## Archiving in linux

### MANAGING COMPRESSED TAR ARCHIVES

### THE tar COMMAND

-Archiving and compressing files are useful when creating backups and transferring data across a network.

-One of the oldest and most common commands for creating and working with backup archives is the **tar** command.

**-**tar**->T**ape **AR**chive

```
# tar     c          creates
          x          extract
          t          watch without extract
          v          verbosity
          f          file
          r          appends
          -C         extract in different location
          --delete   deletes
```

**-create**

```
# tar option *.tar <source>
# tar cfv etc.tar /etc/
```

**-watch** without extract

```
# tar tfv etc.tar
```

**-extract**

```
# tar xfv etc.tar
```

**-extract** in other location

```
# tar xfv etc.tar -C /var/tmp/

# mkdir testdir
# touch testdir/file{1..5}
# tar cfv testdir.tar testdir/
# tar tfv testdir.tar
```

**-extract** specific file/dir

```
# tar xvf testdir.tar testdir/file3
```

**-append** file/dir to existing tarbal file

```
# touch coss.txt
# tar rvf testdir.tar coss.txt
# tar tfv testdir.tar
```

**-delete** file/dir from tarbal file

```
# tar --delete -f testdir.tar testdir/file1
# tar tfv testdir.tar
```

**TAR** compression methods

1-gzip
2-bzip2
3-xzip

**1-gzip**

option: z
extension: .tar.gz

```
-create   ->        # tar cfvz etc.tar.gz /etc/
-extract  ->        # tar xvfz etc.tar.gz
```

**1-bzip2**

option: j
extension: .tar.bz2

```
-create   ->        # tar cfvj etc.tar.bz2 /etc/
-extract  ->        # tar xvfj etc.tar.bz2
```

**1-xzip**

option: J
extension: .tar.xz

```
-create   ->        # tar cfvJ etc.tar.xz /etc/
-extract  ->        # tar xvfJ etc.tar.xz
```

compare

```
# du -h etc.tar              ->22M
# du -h etc.tar.gz           ->5.2M
# du -h etc.tar.bz2          ->3.7M
# du -h etc.tar.xz           ->3.2M
```