Terminal Commands Summary, Part II (Lesson 8)

Linux Distributions

Best suited for basic computing/day-to-day tasks - Ubuntu and Fedora
Best for penetration testers - Kali
Best for software developers - Fedora
Kali and Ubuntu are based on Debian
3 Use Cases for Ubuntu - Cloud building software, build connected sw, run containers at scale
Best for general purpose server - Ubuntu and Fedora
Best for security audit of organizations' networks - Kali
Many possible distributions for a security team

Important Linux Directories

- `/`, the root directory, which contains all other files
  - /etc/ - contains important configuration files, i.e., fstab
  - /etc/passwd - to see users in system
  - /etc/shadow - contains the hashed passwords of every user on the system
  - /etc/sudoers - to determine what users can run administrative tasks
  - /etc/group - contains users and groups
  - /etc/skel - files and folders that are automatically added to a new users home directory when the user is added. Can be modified to customize the new users folder structure
  - /etc/hosts - a local DNS database. It contains the IP <-> Hostname mappings that the server understands on its own. To resolve any IP addresses _not_ contained in this file, the server will have to query a different DNS database.
- `/bin`, - contains executables required for the basic function of the system
  - Executables such as grep and bash are contained here
- `/usr`, - contains user-facing files and applications
  - `/usr/share/man` - contains manpages used by `man`
  - `/usr/bin`- contains nonessential binaries such as `cowsay` and `apt-get` - binaries specific to the user
  - `/usr/sbin`- user programs for system administration such as adduser, chpasswd
- `/var` - contains files whose values change frequently
  - `/var/log` - System logs
  - `/var/tmp` - Temporary files required through reboots
  - `/var/backups`- Contains system backups
- `/tmp` - applications can write "throwaway" data not needed between reboots
  - A common place for malware to cover its tracks as it is cleared on system reboot
- `/home` - contains users' personal data - i.e., /home/freya
  - contains individual users' personal directories and information in, e.g., `/home/<username>/Documents`; `/home/<username>/Downloads`; etc

Commands

Run ‘sudo’ prior to commands that require root privileges. This will work if your account has access for sudo.

cut - remove sections from each line of files

- Suppose you want to use `cut` to chop up the following line in a file: `Jane;Doe;jane@gmail.com;24`. The fields contain her first and last name; email; and age, respectively.
- Which delimiter should you use?

cut -b <filename>

- Which field do you specify if you want to get her first name?

cut -b 1-4 <filename>

- Which field do you specify if you want to get her age?

cut -b 25-26 <filename>

- Which field do you specify if you want to get her last name?

cut -b 6-8 <filename>

- Which field do you specify if you want to get her email?

cut -b 10-23 <filename>

- Which fields do you specify if you want to get her first _and_ last name?

cut -b 1-4,6-8 <filename>

Processes

`supervisord` runs as root. This is a service manager, which starts and restarts services.

ps - lists all processes started by the current user

‘ps aux’ to list all processes

kill - to kill a process - ‘kill <PID>’ - get the PID by listing process with command above

killall <processname> - To kill all processes with a given name

use the `-s` flag to send a specific signal with `kill`

kill -s SIGTERM 107

Installing Packages

A Personal **Package** Archive (**PPA**) is a software repository for uploading source packages to be built and published as an APT repository.

‘apt update’ - To Update PPAs

‘apt install <packages> ‘ - To install new packages (or apt-get install <package_name>)

apt install fortune
apt install cowsay

Use nano to edit ~/.bashrc file so commands are run each time terminal is opened:
nano ~/.bashrc
In nano add the following to ~/.bashrc: fortune | cowsay -f tux

To remove a package:
'apt remove <package>" to remove a package, also run 'apt autoremove' right after to clean up properly

Editing Files
Use nano - 'nano <filename>'

File Permissions
-rwx-rw-r-- : first char is SUID; next 3 chars is owner permission (ie, read/write/execute), next 3 chars is group permission (ie, read/write), last 3 chars is "other" permission (ie, read)
Symbolic permission change: i.e., chmod u=rwx,g=rw,o=r permissions_file
chmod g=r,o= * - to give group read access and remove other permission from files in current directory
Chmod -R g=rw, o= GroupFAQs - for group read/write access for GroupFAQs folder (remove other permission)
Octal permission change: Key numbers are 4(read), 2(write), 1(execute):
i.e., chmod 764 permissions_file : for example under "File Permissions"

Super User
'Sudo' - Stands for 'Super User Do' - prepending commands with this command will invoke system permissions of the root user (also known as super user)

Use 'visudo' to edit the '/etc/sudoers' file to determine what users can run administrative tasks (super user tasks). I.e., to allow teddy the ability to run 'apt' add this to sudoers:
teddy ALL=(ALL:ALL) /usr/bin/apt
/var/log/auth.log: show authentication log messages

Use 'ls l" to list long form of files and directories (i.e., get detailed information)

Users and Groups
'su' - to switch users, i.e., `su jimmy`
'su -p' - to switch login to root user and preserve current environment

'id' - to give the user id, group id, and other permission info for a user, as in 'id randall'
whoami - to print your username

To find out usernames on a system - `ls /home` or `cat /etc/passwd`

‘adduser’ - to add a user. Makes it easy to customize user creation process, set user passwords and create and manage groups and group membership - i.e., ‘adduser ralph’

Can also use ‘useradd,’ but adduser is preferred as it additionally allows the above

`passwd` - to change a user’s password: `passwd <username>`

Primary group: sets the group owner of a file or process when that user creates a file

Secondary group: determines what files they have access to

‘groupadd’ - to add a group - i.e., ‘groupadd developers’

‘usermod’ - to change parameters of a user - i.e., primary group or secondary groups. I.e.,

To change a user’s primary group: `usermod -g <primary group name>`

To change a user’s secondary group: `usermod -aG <secondary group name>`

usermod -g www-data foobar (to change the primary group of foobar to www-data)

usermod -aG hr_administrates bertha (add bertha to secondary group hr_administrates)

‘chmod’ - to change permissions i.e., read, write, execute (change mode - for user, group and “everyone else”). I.e.,

chmod u+rwx,g=rw,o= permissions_file

chmod 760 permissions_file (octal notation)

‘chown’ - stands for ‘change owner’ to change the owner and specify group you are assigning

chown bernard:finance spreadsheet

grep 'group-name-here' /etc/group - to list all members of a group (/etc/group file is the user group file). Can also use ‘members’ command, or ‘lid’ or ‘libuser-lid’ to list user’s groups or group’s users.

‘groups’ command to display group memberships for a user, as in ‘groups vivek’

The /etc/**passwd file** is a text-based database of information about users that may log into the system or other operating system user identities that own running processes.

Linux passwords are **stored** in the /etc/shadow file. They are salted and the algorithm being used depends on the particular distribution and is configurable.

Firewalls

‘ufw’ - uncomplicated firewall. These are used for small networks.

‘iptables ‘ - These are used for enterprise networks

ufw

‘ufw allow from 172.17.0.9’ - to allow a specific host from IP address ‘172.17.0.9’ to access this machine
‘ufw deny 80’ - to deny all traffic using a specific port

enable: starts up the ufw firewall service
disable: stops the firewall service
reload:reloads the firewall rules (after changes are made)
to any: allows access to any port on the host
allow: creates an allow/whitelist rule
deny: creates a deny/blacklist rule
reject: reject rules are like deny rules, except they cause ufw to drop the packet AND send an error message back to the source machine. This is somewhat less secure than a deny rule, because it reveals something about your firewall configuration to the source host.
status numbered: shows a list of firewall rules numbered in order of execution
insert: inserts a rule to a specific location in the list
delete: deletes a specific rule by its number in the list
status: to verify that a firewall is running
default: allows administrators to set default rules, which apply to all packets except for those that have a specific rule set up for them already
logging: tells ufw how much information to log. When logging is enabled, ufw keeps information about every packet it drops. The available levels are off, low, medium and high. Increasing log levels cause ufw to save more information about packets it drops.

To deny all incoming and outgoing traffic (default rule):

sudo ufw default deny incoming
sudo ufw default deny outgoing

‘sudo ufw status verbose’ - to see running default policies

To enable HTTP and HTTPS traffic:

sudo ufw allow out to any port 53 -- for DNS
sudo ufw allow out to any port 80 -- for HTTP
sudo ufw allow out to any port 443 - for HTTPS

To save as much information as possible about dropped packets, set logging level to high: ‘sudo ufw logging high’
Ufw saves information about dropped packets in the file: /var/log/ufw.log

Firewalls should drop packets by default and allow access or egress if it has been explicitly enabled for a port or protocol.
Drop Incoming: ‘ufw default deny incoming’
Drop Outgoing: ‘ufw default deny outgoing’

Save each machine's configuration in a file called: `/tmp/<machine_name>.policy`.
`ufw status verbose > /tmp/<machine_name>.policy`

IPTables

IP Tables match each packet that crosses the networking interface against a set of rules, then decides an action.

Rules - Conditions under which a server should accept or drop a packet

Chain - A series of rules that a firewall applies to network packet

- Input Chains - determine whether the firewall accepts or drops incoming packets
- Output Chains - determine whether the firewall accepts or drops outbound connections
- Forward Chains - determine whether the firewall will act as a router and forward the packet to its destination

Tables - A collection of chains that serve a related purpose

Example 1:

```
# Clear existing rules
iptables -F
iptables -F -t nat
iptables -X
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP
```

-F flushes a chain; -X deletes a chain
-t specifies a table (nat in example)
-P sets the policy for <chain> to <target>

Example 2:

```
# The subnet this server lives on
SUBNET = 192.168.10.0/24

# State rules
iptables -A INPUT -m state --state INVALID -j LOG --log-prefix "DROP INVALID "
--log-ip-options --log-tcp-options
iptables -A INPUT -m state --state INVALID -j DROP
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
```

```
# ACCEPT rules
iptables -A INPUT -i eth1 -p tcp -s $SUBNET --dport 22 --syn -m state  --state NEW -j ACCEPT
iptables -A INPUT -p icmp --icmp-type echo-request -j ACCEPT
```

Important points:

1.) --log-ip-options --log-tcp-options : tell iptable to log traffic it intercepts
2.) -A : appends a rule to the current list of rules
3.) -m : specifies a match which is an extension module that tests for a specific property
4.) --state flag. Possible values
    a.) INVALID: The incoming packet doesn't have a proper state
    b.) ESTABLISHED: The incoming packet belongs to an open connection
    c.) RELATED: The incoming packet is related to another already ESTABLISHED connection. E.g., `FTP-DATA` channels are _related_ to `FTP` channels
    d.) NEW: The incoming packet is initiating a connection.
5.) -j : specifies a chain to jump to if the packet matches the rule
6.) ACCEPT rule 1: `iptables -A INPUT -i eth1 -p tcp -s $SUBNET --dport 22 --syn -m state --state NEW -j ACCEPT`
    a.) `-A INPUT` says: "Append this rule to the end of the INPUT chain."
    b.) `-i eth1` says: "Apply this rule to packets coming in from the `eth1` network interface."
    c.) `-p tcp` says: "Apply this rule to packets sent over `tcp`."
    d.) `-s $SUBNET` says: "Only accept TCP packets sent from the local subnet..."
    e.) `--dport 22` says: "...connecting to port 22..."
    f.) `--syn` says: "...if they have _only_ the `SYN` flag set."
    g.) `-m state --state NEW` says: "Only accept packets that meet these conditions if they meet the previous conditions and are starting a new connection."
    h.) `-j ACCEPT` says: "If the packet matches the preceding conditions, let it through!"
    i.) In English: "I'm adding a rule to the input chain. The rule is: _Only_ allow TCP packets to connect to port 22 on this server if they are initiating a NEW connection; have only the SYN flag set; and come from within the local subnet."
    j.) Port 22 is for SSH, so we could also say: "I'm adding a rule to the input chain. The rule is: _Only_ allow TCP packets to connect to port 22 on this server if they are initiating a NEW connection; have _only_ the SYN flag set; and come from within the local subnet."
7.) iptables -A INPUT -p icmp --icmp-type echo-request -j ACCEPT
    a.) ICMP is used by the `ping` utility to determine if hosts on a network are reachable
    b.) Accept all ping messages sent to this server that are 'echo requests'

Sample iptables commands (see `man iptables` for more info)”

INPUT Chain

```
Iptables -A FORWARD -p tcp --syn --dport 80
-m state --state NEW -j ACCEPT

Iptables -A FORWARD -p tcp --syn --dport 443
-m state --state NEW -j ACCEPT

Iptables -A FORWARD -p tcp --dport 4321 -i
eth0 -s $INT_NET -j ACCEPT
```

INPUT Chain

```
Iptables -A OUTPUT -p tcp --dport 21 --syn
-m state --state NEW -j ACCEPT

Iptables -A OUTPUT -p tcp --dport 22 --syn
-m state --state NEW -j ACCEPT

Iptables -A OUTPUT -p tcp --dport 25 --syn
-m state --state NEW -j ACCEPT
```

- What command do you use to delete all iptables rules?
    `iptables --flush`, or `iptables -F`
- Which command would you use to set the INPUT and OUTPUT default chains to DROP?
    - `iptables --policy INPUT DROP`, or `iptables -PT INPUT DROP`; and `iptables -P OUTPUT DROP`

iptables -A INPUT -i eth0 -p tcp --dport 80 -m state --state NEW,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -o eth0 -p tcp --sport 80 -m state --state ESTABLISHED -j ACCEPT

- The first rule allows incoming traffic to port 80. The second rule allows outgoing traffic from port 80.
- These services enable HTTP traffic.
- If the first rule were implemented without the second, the server could listen to HTTP traffic, but not respond. If the latter were implemented without the former, the server could theoretically send HTTP responses, but wouldn't be able to "hear" HTTP requests.

iptables -A INPUT -i eth0 -p tcp -s 192.168.100.0/24 --dport 3306 -m state --state NEW,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -o eth0 -p tcp --sport 3306 -m state --state ESTABLISHED -j ACCEPT

- Only devices on the subnet `192.168.100.0/24` can connect to this machine's port 3306.
- MySQL runs on port 3306 by default.
- Since databases are so sensitive, access to them should be allowed by as few devices as possible. In this case, the DB is probably either a database on a private subnet which does not need to be exposed to the public Internet; or, it backs a public-facing HTTP server on the local subnet.

- What is the `nat` table?
- What is the PREROUTING chain?
- What is DNAT?
- What is the relationship between `--dport 422` and `192.168.102.37:22`?

iptables -t nat -A PREROUTING -p tcp -d 192.168.102.37 --dport 422 -j DNAT --to 192.168.102.37:22

-The `nat` table sets up Network Address Translation (NAT) rules, which is important for devices being used as routers.
- The PREROUTING chain contains rules that translate the source IP address of outgoing packets to match that of the routing device, _not_ the subnet IP address of the originating machine. This is called "source NAT", or SNAT, and is used to allow a device behind a router to
- DNAT stands for "destination NAT". This is the process of changing the destination IP address of a packet, but _not_ the source. This is intended to allow hosts _outside_ the router's subnet to communicate with devices on that subnet.
- This rule forwards connections to `192.168.102:37:422` to `192.168.102:37:22`. This means SSH clients can connect to `192.168.102:37`'s SSH server through either port 422 or 22