Firewall Security Concept in Redhat Linux

================ Firewall Concept ======================
OR

How to Allow and Deny any incoming request from any client machine on Linux servers ?

===

Types of services

1- Internal service

2- External service OR Network based services

===

What is firewalls in linux ?

Firewall basically use to allow or deny any request on server machine.

With the help of firewall we can set OR configure allow and deny rules to access any services in my machine.

===
Types of Services ?

1- Internal services ==> daemon + PID
2- External Services ==> daemon + PID + Port number running [which is to take request from any client machine]

===
Example: ssh , ftp , database, httpd , DNS .... etc

Types of firewall ?

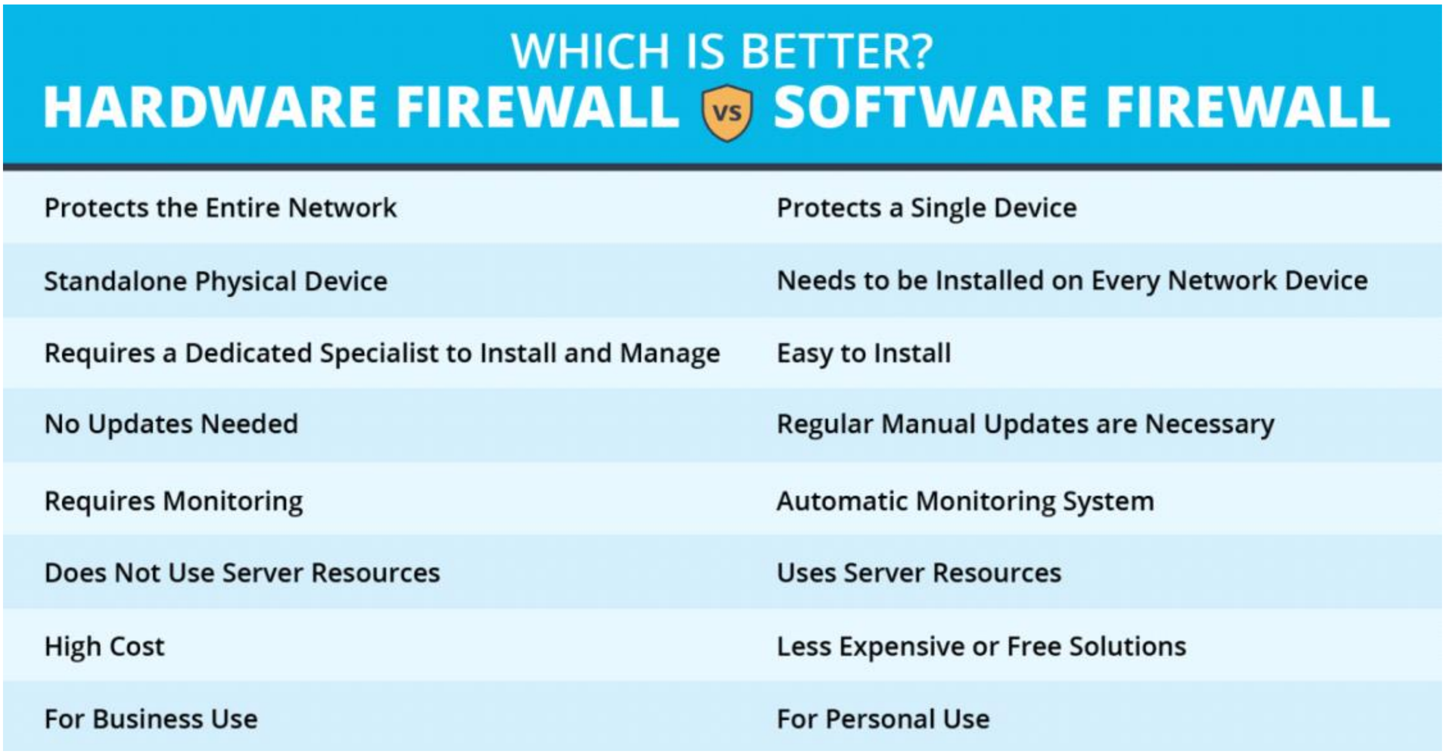
Physical Infra

1- hardware based firewall ==> which is available in market

2- Software based firewall ==> With in the OS # systemctl stop firewalld # systemctl disable firewalld

Cloud infra

3- Third party public cloud provider based firewall.... aws cloud , google cloud, alibaba , azure cloud ...openstack ...etc



WHICH IS BETTER?

HARDWARE FIREWALL	SOFTWARE FIREWALL
Protects the Entire Network	Protects a Single Device
Standalone Physical Device	Needs to be Installed on Every Network Device
Requires a Dedicated Specialist to Install and Manage	Easy to Install
No Updates Needed	Regular Manual Updates are Necessary
Requires Monitoring	Automatic Monitoring System
Does Not Use Server Resources	Uses Server Resources
High Cost	Less Expensive or Free Solutions
For Business Use	For Personal Use

== /etc/hosts.allow and /etc/hosts.deny

In linux we have os defined firewalls inbuilt in OS....

1- TCP - Wrappers ==> Support till RHEL-7

2- IP-Tables ===> Till rhel-6

by RHEL-7===>new firewalls ==> firewalld [Recommended]

===

3- IPtables replaced with firewalld (new firewall concept in rhel-7 + rhel- 8 + rhel-9)

===

Note: in RHEL-8 we can not controll any traffic using TCP wrappers...
this technology has been removed in rhel-8. but till RHEL-7 we can eaisly use it....

===

Advance Firewall ==> IP-tables OR Firewalld { New product in rhel-7 + rhel-8 + rhel-9 }

default firewall in rhel-7 + 8 + 9 is firewalld but still we can apply the rules using iptables
in rhel-7 + in rhel-8 and in rhel-9. but it is not a recommended practise

we need to install iptables related packages.

===

Note: at a time in a machine we can use single firewalls only either we can use iptables or else we can go with firewalld.

but on same time we can not use both services.

```
=====================================================

[root@node20 ~]# cat  /etc/hosts.deny
cat: /etc/hosts.deny: No such file or directory
[root@node20 ~]#
[root@node20 ~]# cat  /etc/hosts.allow
cat: /etc/hosts.allow: No such file or directory
[root@node20 ~]#
[root@node20 ~]#
[root@node20 ~]#
[root@node20 ~]# rpm -qa  iptables
[root@node20 ~]#
[root@node20 ~]# rpm -qa  firewalld
firewalld-1.0.0-4.el9.noarch
[root@node20 ~]#
[root@node20 ~]# systemctl stop firewalld
[root@node20 ~]# systemctl disable  firewalld
Removed /etc/systemd/system/multi-user.target.wants/firewalld.service.
Removed /etc/systemd/system/dbus-org.fedoraproject.FirewallD1.service.
[root@node20 ~]#

[root@node20 ~]# systemctl start firewalld
[root@node20 ~]# systemctl enable firewalld



[root@node20 ~]#
[root@node20 ~]#
[root@node20 ~]# pstree  | grep crond
        |-crond
[root@node20 ~]#
[root@node20 ~]# pstree  | grep sshd
        |-sshd
[root@node20 ~]#
[root@node20 ~]# pstree  | grep vsftpd
[root@node20 ~]#
[root@node20 ~]#
[root@node20 ~]# pidof crond
1051
[root@node20 ~]# pidof sshd
970
[root@node20 ~]#
[root@node20 ~]# netstat -tunlp |  grep crond
[root@node20 ~]# netstat -tunlp |  grep sshd
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      970/sshd: /usr/sbin
tcp6       0      0 :::22                   :::*                    LISTEN      970/sshd: /usr/sbin
[root@node20 ~]#

[root@node20 ~]# ss  -tunlp |  grep crond
[root@node20 ~]# ss  -tunlp |  grep sshd
tcp   LISTEN 0      128          0.0.0.0:22        0.0.0.0:*    users:(("sshd",pid=970,fd=3))
tcp   LISTEN 0      128             [::]:22           [::]:*    users:(("sshd",pid=970,fd=4))
[root@node20 ~]#

[root@node20 ~]# ss  -tunlp |  grep httpd
[root@node20 ~]#
[root@node20 ~]#
```

```
[root@node20 ~]# pidof httpd
[root@node20 ~]#
[root@node20 ~]#
[root@node20 ~]# rpm -qa httpd
[root@node20 ~]#
```

daemon + port number ===> # netstat OR # ss

```
#  cat  /etc/services ===> we can check the default port number of any linux based services
```

==

First Topic :==================================

==

************** TCP - Wrappers ***************

1- IT is an example of software firewalls .

2- with the help of tcp wrappers we can apply only incomming based rules on server machine
   to allow or deny any traffic coming to the server side.

Types of Rules :-   1- Incoming   2- Outgoing   3- forwarding

Note: With the help iptables and firewalld we can apply any types of rules in linux OS.

3- TCP-wrappers always works only on these two files:-   note: syntax are same in both files.

  1-   /etc/hosts.deny   ==> to apply deny rules

  2-   /etc/hosts.allow   ==> to apply Allow rules

==

Syntax:-

daemonname: Clientlist

==
IN RHEL-9

```
[root@localhost ~]# cat /etc/hosts.deny
cat: /etc/hosts.deny: No such file or directory
[root@localhost ~]#
[root@localhost ~]# cat /etc/hosts.allow
cat: /etc/hosts.allow: No such file or directory
[root@localhost ~]#
[root@localhost ~]#
```

==

==

Example-1

How to block any particular machine to access the SSH remote login using TCP- Wrappers.

```
machine-1   ====> IP address  is  ====>    172.25.0.20
machine-2  ====>  IP address  is  ====>    172.25.0.250
```

ON Machine-1

```
# vim  /etc/hosts.deny



sshd:  172.25.0.250  [EX: Client Node IP]

:wq

# systemctl  restart sshd
```

machine-2 ===>   172.25.0.250

```
# ssh root@172.25.0.20   [Server node IP]
```

Note: you should get errors here...remote login will not allowed by server machine [172.25.0.20]

==

note: do this activity only on rhel-7 OS, it will not support in rhel-8 and rhel-9

==

==

Possible examples or rules in TCP - wrappers   OR Syntax

==

```
#  vim  /etc/hosts.deny  or   hosts.allow
```

Examples:-

```
sshd:  172.25.0.250
sshd:  172.25.0.250  172.25.0.251  172.25.0.252
sshd:  172.25.0.0/24
sshd:  .example.com
sshd:  10.0.0.0/8    172.25.0.0/24
sshd:  ALL
vsftpd: 192.168.0.250
sshd,vsftpd: ALL
sshd: ALL
vsftpd: ALL
ALL:  172.25.0.250
ALL:  ALL
sshd: ALL EXCEPT 172.25.0.250
sshd: ALL EXCEPT  172.25.0.0/24
ALL EXCEPT sshd: ALL
```

==

Advance Options:

```
#  vim  /etc/hosts.deny

sshd:  172.25.0.250  :  ALLOW

:wq

#  vim  /etc/hosts.allow

sshd:  172.25.0.250 :  DENY

:wq
```

===

Second Topic

****************   IP-Tables   **

1- IP-Tables is also an example of Software firewalls OR host based firewalls.
2- With the help of IP-Tables we can apply incomming , outgoing and port forwarding based rules.

===

```
Package  :    iptables
Daemon  :    iptables
File:              /etc/sysconfig/iptables    <=== Rules are store here
Command :    iptables  <options>
```

===

```
# service  iptables save
```

===
===

/etc/hosts.allow or /etc/hosts.deny ==> TCP wrapper Examples

IP Tables ==> # iptables <options-1> <option-2> .... so on

===

```
# iptables -------rules-configuration------        -j    ACCEPT
                                                   -j    REJECT
                                                   -j    DROP
                                                   -j    REDIRECT
```

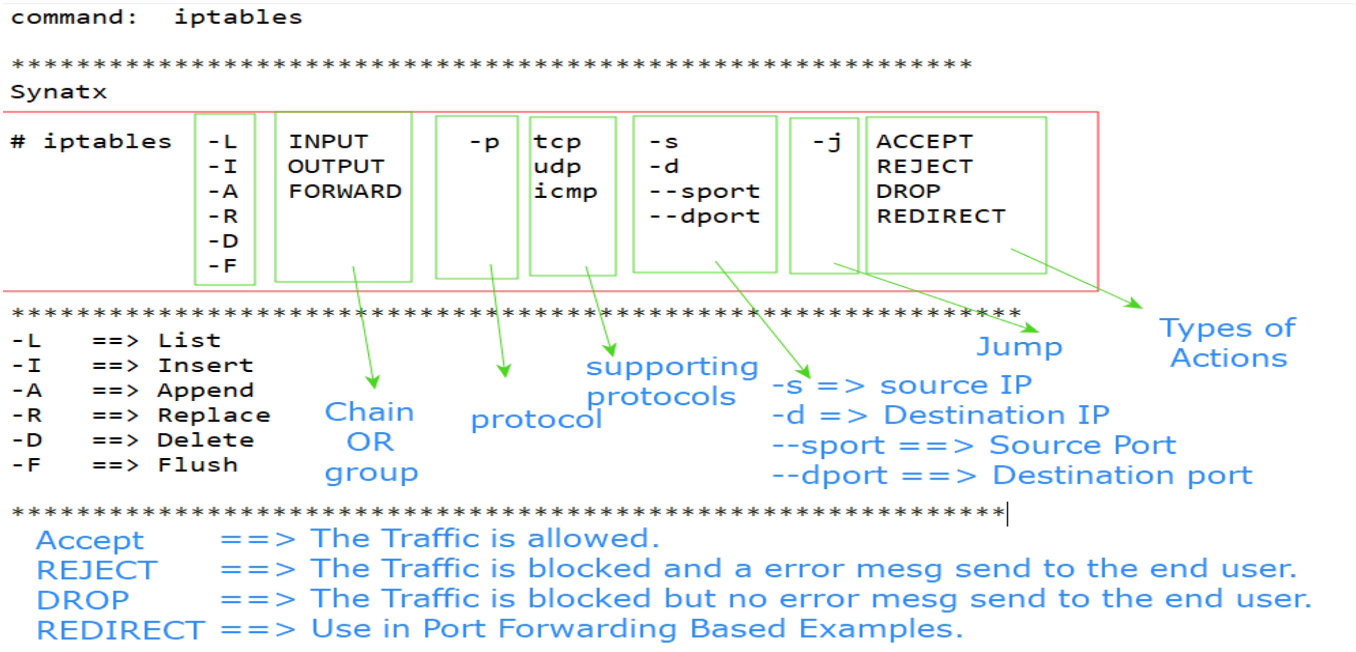
Types of Actions:-

ACCEPT   ==> Traffic is allowed.

REJECT   ==> Traffic is block and error mesg send to END user.

DROP   ==> Traffic is block but no error mesg send to END user.

REDIRECT ==> It always use in Port Forwarding Based Examples.



```
command:  iptables

************************************************************
Synatx

# iptables   -L   INPUT      -p   tcp    -s        -j   ACCEPT
             -I   OUTPUT          udp    -d             REJECT
             -A   FORWARD         icmp   --sport        DROP
             -R                          --dport        REDIRECT
             -D
             -F
***************************************************************
-L    ==> List
-I    ==> Insert
-A    ==> Append
-R    ==> Replace
-D    ==> Delete
-F    ==> Flush
**************************************************************
```

-s ==> source IP
-d ==> Destination IP
--sport ==> Source Port
--dport ==> Destination port

Accept ==> The Traffic is allowed.
REJECT ==> The Traffic is blocked and a error mesg send to the end user.
DROP ==> The Traffic is blocked but no error mesg send to the end user.
REDIRECT ==> Use in Port Forwarding Based Examples.

==

==