Filter by IP Address

1. Match by source IP address

Match: -s IP, --source IP
Example: *iptables -A INPUT -s 100.0.0.0/16 -j DROP*

2. Match by destination IP address

Match: -d IP, --destination IP
Example: *iptables -A FORWARD -d 80.0.0.1 -j DROP*

Filter by IP range and address type

1. Match by IP range

Match: -m iprange --src-range ip_start-ip_end
-m iprange --dst-range ip_start-ip_end
Example: *iptables -A INPUT* -m iprange --src-range 10.0.0.10-10.0.0.33 -p tcp --dport 25 *-j DROP*

2. Match by address type

Match: -m addrtype --src-type UNICAST,MULTICAST,BROADCAST
-m addrtype --dst-type UNICAST,MULTICAST,BROADCAST
Example: *iptables -A OUTPUT -m addrtype --dst-type MULTICAST -j DROP*

Filter by TCP or UDP port

1. Match a single port

Match: -p tcp --dport port, -p udp --sport port
Example: *iptables -A INPUT -p tcp --dport 22 -j DROP*

2. Match multiple ports

Match: -m multiport --sports | --dports port1,port2,...
Example:
iptables -A OUTPUT -p tcp -m multiport --dports 80,443 -j ACCEPT

Filter by TCP or UDP port

More examples:

1. Block all incoming traffic to port 80
2. Permit ssh incoming traffic only from a specific ip address
3. Block port 25 and 80 using '*-m multiport*'

Filter by interface

1. Match by incoming interface

Match: -i incoming_interface
Example: *iptables -A INPUT -i wlan0 -j ACCEPT*

2. Match by outgoing interface

Match: -o outgoing_interface
Example: *iptables -A OUTPUT -o enp0s3 ACCEPT*

Note: It is good practice to permit traffic on loopback interface (lo)

Filter by MAC Address

It is possible to filter traffic only by source mac address.

Match: -m mac --mac-source source_mac_address

Example:
iptables -A INPUT -i wlan0 -m mac --mac-source 08:00:27:55:6f:20 -j DROP

Filter by MAC Address

More examples:

1. Drop packets from a specific mac address

2. Permit only a list of mac address through our firewall (NAT Router)

Match by TCP flags

Match:

--syn: match if the syn flag in set

--tcp-flags mask comp. Match when the TCP flags are set as specified.

The first argument mask is the flags which we should examine, written as a comma-separated list, and the second argument comp is a comma-separated list of flags which must be set.

Match by TCP flags

TCP Flags are:

SYN - synchronize

ACK - acknowledgement

FIN - finalize

RST - reset

URG - urgent

PSH - push

ALL

NONE

Match by TCP flags

More examples:

1. Block incoming tcp connections to port 22
2. Permit incoming connections to port 22 only from a specific IP

Filter by date & time

-m time *option*

Time match options:

--datestart time Start and stop time, to be given in ISO 8601

--datestop time (YYYY[-MM[-DD[Thh[:mm[:ss]]]]])

--timestart time Start and stop daytime (hh:mm[:ss])

--timestop time (between 00:00:00 and 23:59:59)

--monthdays value List of days on which to match, separated by comma

[!] --weekdays value List of weekdays on which to match, sep. by comma

--kerneltz Work with the kernel timezone instead of UTC

Filter by date & time (cont)

Note: By default it uses UTC and not system time

--kerneltz makes netfilter use system time instead of UTC time (!!!!)

Example:

1. Permit incoming ssh traffic only between 8am and 6pm on weekdays
2. Allow access to a specific web site only after working hours (> 6pm). This machine is the Router.

Quota match

-m quota --quota *bytes*

Example:

iptables -A OUTPUT -d 80.0.0.1 -p tcp --sport 80 -m quota --quota 1000000000 -j ACCEPT

iptables -A OUTPUT -d 80.0.0.1 -p tcp --sport 80 -j DROP

Limit match

-m limit *option*

There are 2 options:

--limit *value*, where value is the maximum matches pe time-unit (default second)

--limit-burst *value*, where value is maximum burst (matches) before the above limit "kicks in" (default 5)

Examples:

1. iptables -A FORWARD -m limit --limit 1/minute -p udp --dport 53 -j LOG
2. iptables -A INPUT -p tcp --syn -m limit --limit 1/s --limit-burst 7 -j ACCEPT

Connlimit match

-m connlimit *option*

Connlimit match options:

--connlimit-upto n : match if the number of existing connections is less than n

--connlimit-above n match if the number of existing connections is more than n

Example:

iptables -A INPUT -p tcp --dport 25 --syn -m connlimit --connlimit-above 5 -j REJECT --reject-with tcp-rst

Connection tracking

Connection tracking = stateful firewall

Packet states:

1. NEW - 1st packet from a connection
2. ESTABLISHED - packets that are part of an existing connection
3. RELATED - packets that are requesting a new connection and are already part of an existing connection (Ex: FTP)
4. INVALID - packets that are not part of any existing connection

Connection tracking can be used even if the protocol itself is stateless (Ex: UDP).

Connection tracking

Connection tracking = stateful firewall

- Connection tracking = ability to maintain **state information** about connections
- Stateful firewalls are **more secure** than stateless firewalls
- Stateful firewalls decide to accept or to drop packets **based on the relations** these packets are with other packets
- Netfilter is a stateful firewall

Connection tracking (cont)

-m state --state *state*, where state is a comma separated values of packet states

Example:

iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

iptables -P INPUT DROP

Recent match

It creates a **dynamic database** of blacklisted source IP addresses.

Example:

```
iptables -A FORWARD -m recent --name badguy --update --seconds 60 -j DROP
iptables -A FORWARD -p tcp -i eth0 --dport 8080 -m recent --name badguy --set -j DROP
```

The list with blacklisted IP addresses is found at: `/proc/net/xt_recent/LIST_NAME`

Recent match

Recent match options:

--update: checks if the source IP address belongs to the list and updates the **"last seen time"**.

It created a temporary banning or a **quiet time**.

--rcheck: checks if the source IP address belongs to the list and DOESN'T update "last seen time".

Any packet from that source IP address will be dropped for an absolute time given by the number of specified seconds.

--seconds: used with -**-update** or **--rcheck.** Matches the packet only if the source IP address is in the list and the last seen time is valid.

--set: adds the source IP address to the list.

--name: creates a list in which the source IP address will be added and checked.

Filter by country (Geoip)

- An average email server rejects between 30-90% of emails as spam.
- We can filter traffic **based on the country** the packet is coming from or destined to.
- iptables uses xt_geoip which consists of an extension named xtable_addon and GeoIP database to perform country specific filtering.
- xt_geoip **doesn't belong** the standard netfilter/iptables framework. It should be installed.
- The country is specified using the ISO 3166 code: https://en.wikipedia.org/wiki/ISO_3166-1

Filter by country (Geoip) - installation & config

1. Install xtables-addons and dependencies

Ubuntu: sudo apt-get update && sudo apt-get install xtables-addons-common libtext-csv-xs-perl pkg-config iptables-dev

CentOS: yum update && yum install gcc-c++ make automake kernel-devel-`uname -r` wget unzip iptables-devel perl-Text-CSV_XS

2. Download GeoIP list (from MaxMind) as a CSV file

From a temp directory (ex: /tmp) run: sudo /usr/lib/xtables-addons/xt_geoip_dl

3. Create a folder for converted files and import them into xtables

sudo mkdir /usr/share/xt_geoip

sudo /usr/lib/xtables-addons/xt_geoip_build -D /usr/share/xt_geoip *.csv

4. Test it!

iptables -m geoip --help