

Linux Administration

Firewalls

Xavier Belanger

**This work is licensed under
a Creative Commons Attribution-ShareAlike 4.0 International License.**

<http://creativecommons.org/licenses/by-sa/4.0/>

You are free to:

- **Share** — copy and redistribute the material in any medium or format
- **Adapt** — remix, transform, and build upon the material for any purpose, even commercially.

The licensor cannot revoke these freedoms as long as you follow the license terms.

Under the following terms:

- **Attribution** — You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.
- **ShareAlike** — If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.
- **No additional restrictions** — You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

Why using a firewall?

- The network where your system is hosted may be considered as “hostile”.
- Some applications may not provide an easy way to restrict accesses.
- This would provide an additional layer of protection.
- You may need to use firewall rules to modify the network traffic.

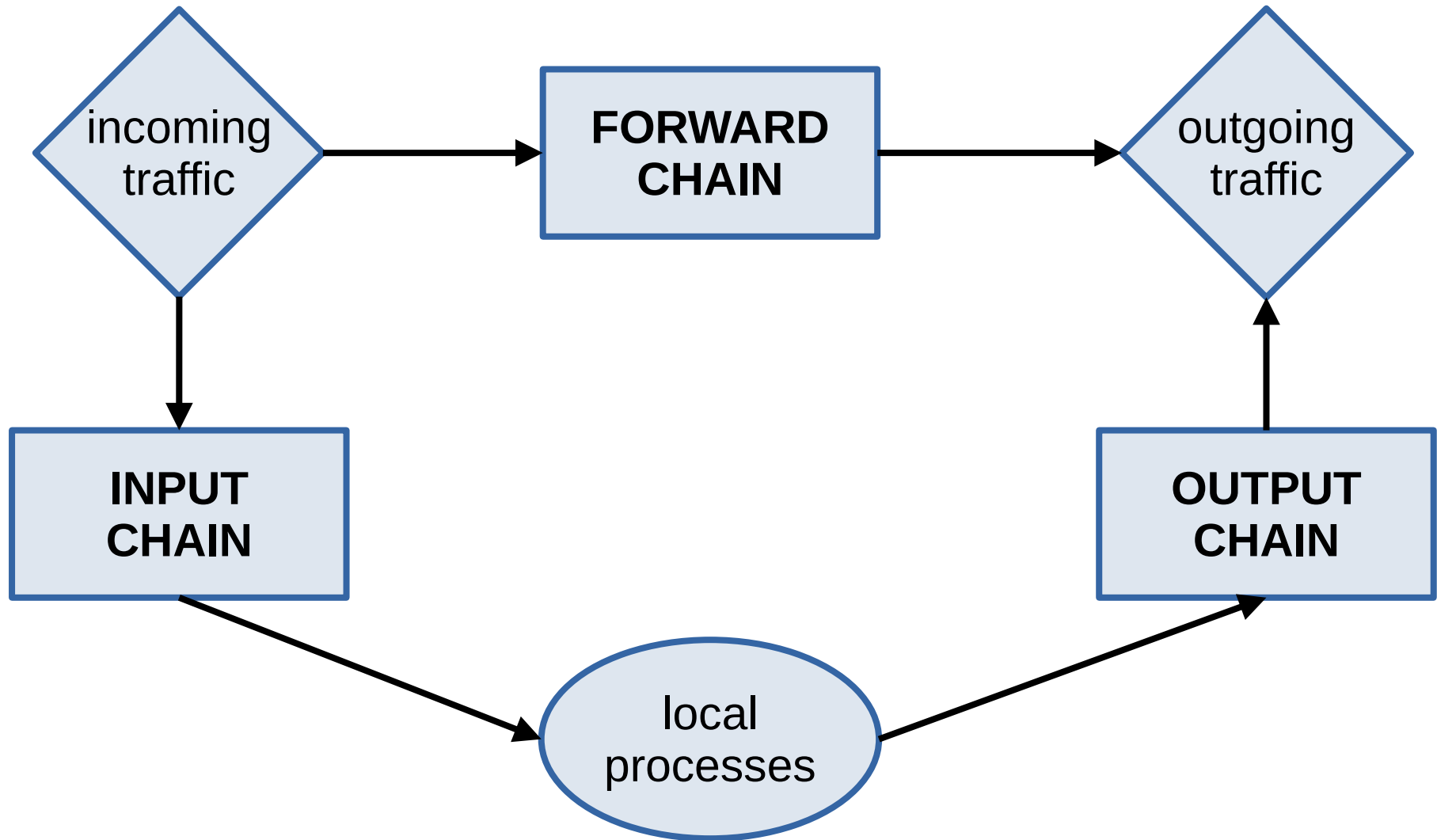
The Netfilter project

- Network filtering and related applications for the Linux system are under the umbrella of the Netfilter project.
- The main tools available to manage firewalls are iptables and nftables.
- <https://netfilter.org/>

iptables

- *iptables* is currently the most common filtering tool used on Linux systems.
- It relies on various network tables; each table contains chains, with a default policy (accept or drop).
- Filtering rules can be added or removed in each chain.
- 'Filter' is the default table, with three chains: input, forward and output.

iptables filter table flow



Defining the default policy

The default policy should be to reject silently all packets ('drop'); only authorized packets will be accepted.

- iptables -F (flush all the chains)
- iptables -X (delete all the chains)
- iptables -P INPUT DROP
- iptables -P FORWARD DROP
- iptables -P OUTPUT DROP

Manipulating rules

- Adding a rule: `iptables -A <options>`
- Inserting a rule: `iptables -I <options>`
- Deleting a rule: `iptables -D <options>`
- Replacing a rule: `iptables -R <options>`
- Listing the rules:
`iptables -L -n -v --line-number`

Rule actions

- A rule should specify a target for a matching packet:
 - ACCEPT: the packet is transferred to the system
 - DROP: the packet is discarded
 - REJECT: the packet is refused and the sender is notified
 - LOG: the packet is logged and sent to the next rule
- Other options are available, for more advanced situations.

Creating your first rule

- A rule usually match one of those criteria:

Source IP Address

Source Port

Destination IP Address

Destination Port

- Check the `/etc/services` file and the ‘`ss`’ command output for port numbers.
- Other criteria are available: network interface, connection status, ...

Rule example

- If you want to allow all HTTP connections to your system coming from the network 192.168.5.0/24:

```
iptables -A INPUT -s 192.168.5.0/24 --dport 80 -j ACCEPT
```

- Incoming connections from other networks, on the same port will be managed with the default policy.

nftables

- This is the successor of iptables, using different tools and syntax, and allowing new types of operations.
- Both iptables and nftables may be available on the same system, but you want to use only one and ignore the other.
- Some of the main differences are:
 - nftables doesn't provide pre-build tables
 - the syntax is different; you can use the *iptables-translate* command to convert iptables scripts
 - a rule can perform multiple actions (blocking and logging for instance)
 - the same rules can be used for both IPv4 and IPv6
 - performances have been improved

nftables syntax

One benefit of nftable is that the syntax is more explicit.

- nft { add | delete | list | flush } table { ipv4 | ipv6 | inet }
table table_name
- nft { add | create | delete | rename | list | flush } chain
table_name chain_name <options>
- nft { add | insert | replace | delete } rule <options>
- nft list ruleset
- nft flush ruleset

Firewall script

- Firewall rules can be set or modified manually, but it is strongly recommended to apply them with a script, during the boot process.
- Depending on your distribution, some scripts or tools may already be provided (firewalld on Red Hat Enterprise Linux, ufw on Ubuntu for instance).
- Test your script when you are not relying on network connectivity!