

Introduction à la sécurité TP 3 - Pare-feu IPTables/Netfilters

Maciej Korczynski & Simon Fernandez

Septembre 2020

Objectifs

- Mise en place d'un système de pare-feu avec Netfilter / IPTables

iptables

Netfilter is a module of Linux kernel that allows to control / modify / filter / follow network connection. It provides firewall functions (connection / sharing / authorization of network traffic).

iptables is a command-line interface to configuring Netfilter.

iptables works internally on 5 tables :

- **filter** -> packet filtering (security)
- **nat** -> packet "NATing" (linux gateway)
- **mangle** -> packet modification (QoS for example)
- **raw** -> run before connection tracking mechanisms, to avoid the computational cost
- **security** -> run after the **filter** table, used to authenticate users

Default table used is **filter**.

Table Filtering

There are 3 internal chains in this table :

- INPUT
- FORWARD
- OUTPUT

There are 3 actions possible from this table (in fact there are others, but not much used) :

- ACCEPT
- DROP
- REJECT

The difference between DROP and REJECT is that REJECT will send a ICMP error code packet to the sender, DROP will be silent.

To set the default action of a chain :

```
iptables -P chain action
```

(for example: `iptables -P INPUT ACCEPT`, to accept all input packets)

Packets go through rules one by one (in the order of the list in the chain).

If a matching rule's action is **ACCEPT/DROP/REJECT** the packet will no longer be processed by further rules. It will be accepted/dropped/rejected accordingly and the processing stops there.

Never forget that rules are applied in the list order ! This can have huge security implications to invert 2 rules.

Some standard command :

- `iptables -L -v` : list (verbosely) all rules in the filter table
- `iptables -F chain` : clear all rules (in given chain if specified or all chains)
- `iptables -A chain rule` : append a rule to the specified chain
- `iptables -D chain rule/number` : delete a rule in the given chain (identified by complete rule or position index)
- `iptables -I chain position rule` : insert a rule in the given position in the given chain

Some rules syntax :

- `-p protocol -j ACTION` (minimalist form)
- First level options (`-something`) often have sub-options (`--somethingelse`) to tune things a bit more.

```
-p tcp --destination-port 80 -j DROP
```

This drops all TCP packets going to port 80

(Complete command would be : `iptables -A INPUT -p tcp --dport 80 -j DROP`)

Practice

Be careful, if you are connecting to the machine with `ssh`, forbidding all traffic will drop the connection and you will be “locked” outside of your remote machine. Add the rules in an adapted order so that your connection is not dropped

- Forbid all connection by default.
- Accept only ping (ICMP protocol), `ssh` and `http` traffic. Accept also people from outside to access to your website.
- Test the difference between actions `REJECT` and `DROP` with the previous rules (with `wireshark` for example)
- Check your configurations with the `nmap` tool from TP2

Useful notion

You can create you own user chain, with

```
iptables -n Name
```

For example, you can create a chain for `tcp` traffic with 80 as destination port, and redirect all traffic into it :

```
iptables -n HTTP
iptables -A INPUT -p tcp --dport 80 -j HTTP
iptables -A HTTP [your rules..]
```

There are some useful actions :

- `-j LOG -log-prefix "my log prefix"`
- `-j RETURN` : exit the current ‘user’ chain and continue processing in the parent chain

Sometimes, you will need some module of `netifilter/iptables`, to active this module you will need to use `modprobe`.

Conntrack

You can follow network connection and add rules according to the “state” of the connection with `conntrack`. `conntrack` proposes 6 states : `INVALID|ESTABLISHED|NEW|RELATED|SNAT|DNAT`. You can also combine states. It can be very useful for some protocol or for some specific configurations.

Practice

- Describe briefly how a FTP connection can be established between a client and a server. List the different packets sent, and the source-destination ports
- Find adapted rules to allow a FTP connection. You will need to active `ip_conntrack_ftp` (or `nf_conntrack_ftp` according to your kernel) with `modprobe`, if it is not already activated.

iptables offers a lot of powerful and advanced features (load balancers, hiding internal servers, ...), but a bad use of it can be dangerous for system safety, so be careful when you build your rules.

Saving the configuration

By default, **iptables** rules are not saved, they will last until the machine is rebooted. In order to make the changes permanent, you have to export your changes to a file, and load the rule file at boot :

- **iptables-save** will output your actual configuration
- **iptables-restore** will load a configuration

In order to apply your configuration at each boot, you will have to setup your system so that it calls **iptables-restore** when booting (with **systemd** for example)