Linux Routers and Community Networks

Llorenç Cerdà-Alabern
http://personals.ac.upc.edu/llorenc
llorenc@ac.upc.edu

Universitat Politènica de Catalunya,
Barcelona, Spain

Revision: 0e36644 (2015-07-09)

Parts

Part IV

Lab 3: Firewall configuration

Outline

Description

Objectives

- At the heart of Linux firewall configuration there is the `iptables` command.
- Objectives of this lab:
  - Getting familiar with the basic usage of `iptables`.
  - Use OpenWrt firewall configuration web interface.

Part IV

Lab 3: Firewall configuration

Outline

Iptables terminology

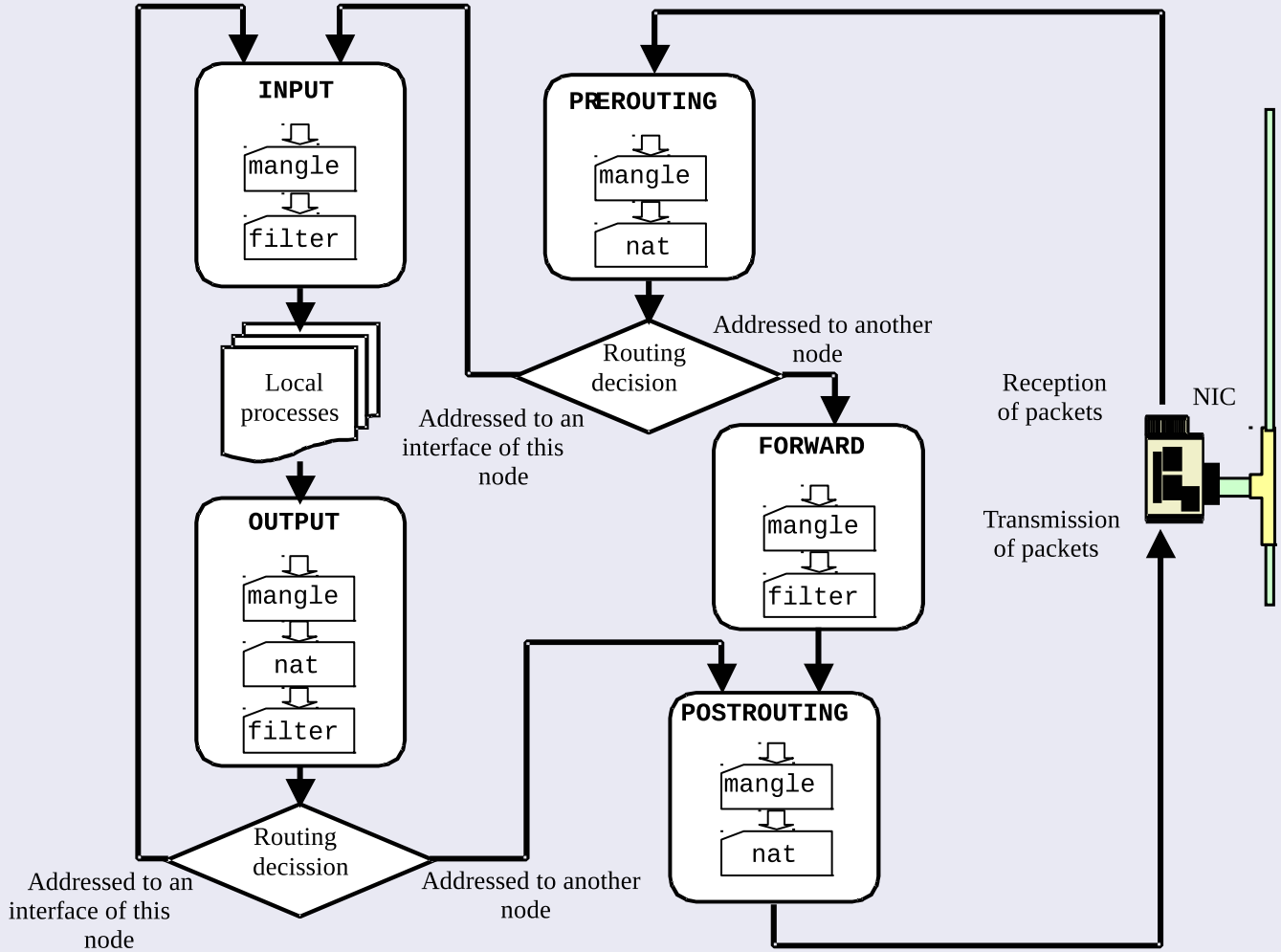
- The iptables command allows you to filter and/or modify some fields of the packets as they move through different stages (or chains) of the IP layer of the Linux machine.
- These built-in chains are `PREROUTING`, `FORWARD`, `POSTROUTING`, `INPUT` and `OUTPUT`.
- When the packet moves through one of these chains, the IP layer consults the tables where the iptables command makes possible to add rules.
- An action, or target, is applied to packets that match the expression specified in a rule. These tables are `mangle`, `filter` and `nat`.

Iptables Fundamentals

Tables

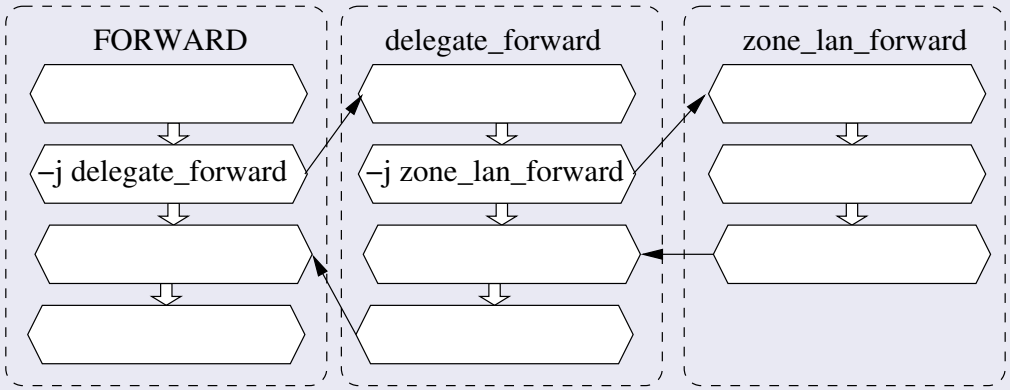
- The `mangle` table allows you to add rules that change some of the fields of the packets, such as TTL or TOS.
- The `nat` table allows you to add rules that change the IP addresses of the packets. The types of rules are: `SNAT` to change the source address and `DNAT` to change the target address.
- The `filter` table allows you to add rules for filtering. The types of rules are: `DROP` to discard a packet and `ACCEPT` to accept it.

Processing of IP datagrams in a Linux router



User specified chains

- User can create user-chains.
- If a packet enters a built-in chain, e.g. `FORWARD` in the filter table, a rule can specify a jump to a user-chain within the same table.



Part IV

Lab 3: Firewall configuration

Outline

Creation of rules

Generic iptables command

```
iptables [-t <table>] <command> <expression> -j <rule type>
```

- If the table is not indicated, the rule is considered to be applied to `filter` table.
- The expression (or match) identifies which are the packets in which the rule has to be applied.
- The type of rule (or target) identifies what has to be done with the packet matching the rule: `DROP`, `ACCEPT`, `SNAT`, `DNAT` or jump to a user-chain.

Creation of rules

Commands

- `-A <chain>`: adds a rule to the table of the chain `<chain>`.
  Example: `iptables -A INPUT ....`
- `-D <chain> <n>`: removes the rule <n> from a table of <chain>.
  Example: `iptables -D INPUT 1`.
- `-I <chain> <n> ...`: inserts a rule in the line that is specified.
  Example: `iptables -I INPUT 1 --dport 80 -j ACCEPT`
- `-L <chain>`: lists the rules of a table of `<chain>`.
  - `-n` not translate numerical addresses into names.
  - `-v` verbose.
  - `--line` show the number of the rule.

  Example: `iptables --line -nvL INPUT`.
- `-P <chain>`: specifies the default rule. `ACCEPT` if not specified.
  Example: `iptables -P INPUT DROP`.

Creation of rules

Generic expressions

- `-p <protocol>`: identifies a protocol.
  Example: `iptables -A INPUT -p tcp ...` will add a rule that will be applied for all the TCP packets to the filter table of the `INPUT` chain.
- `-s <@IP src>`: identifies a packet with `<@IP src>`. A mask can be added, and `!` denies the expression.
  Example: `iptables -A INPUT -s ! 192.168.0.0/24 ...` will add a rule to the `filter` table of the `INPUT` chain that will be applied for all the packets with a source `@IP` that doesn't belong to the range `192.168.0.0/24`.
- `-d <@IP destination>`: same for destination address.
- `-i <input-interface>`: input interface. It can be only applied in the chains `INPUT`, `FORWARD` and `PREROUTING`.
  Example, `iptables -A INPUT -i eth0 ...` will add a rule that will be applied for all the packets that will arrive from the `eth0` interface.

Creation of rules

TCP Expressions

Can be only applied for the TCP protocol: `-p tcp`.

- `--sport <src-port>`: TCP packets with `<src-port>`. Range: `initial:final` can be specified. Otherwise, the initial value that will be indicated is 0 and, if not, the final value that will be indicated is 65535.
  Example: `iptables -A INPUT -p tcp --sport 1024: ...` will apply the rule for all the TCP packets with a port number ≥1024.
- `--dport <dst-port>`: same for the destination port.
- `--tcp-flags <flag list> <flags list to 1>`: look at the packets of the `<flag list>` which flags have only a value equal to 1 if they are included in `<flags list to 1>`. Flags can be `SYN`, `FIN`, `ACK`, `RST`, `URG`, `PSH`, `ALL`, `NONE`.
  Example: `iptables -p tcp --tcp-flags SYN,RST,ACK SYN ...` will apply the rule for all the packets with the flag `SYN` enabled and the `RST` and `ACK` to 0.

Creation of rules

UDP Expressions

Can be only applied for the UDP protocol: `-p udp`.

- `--sport <src-port>`: same as stated before for TCP.
- `--dport <dst-port>`: same as stated before for TCP.

ICMP Expressions

Can be only applied for the ICMP protocol: `-p icmp`.

- `--icmp-type <type>`: identifies the `icmp` packets of the type `<type>`. If you execute `iptables -p icmp --help` a list of the possible types will be given.

State Expressions

Regarded as *no implicit*: `-p state`.
States can be: `NEW`, `ESTABLISHED` and `RELATED`.

- `--state <state list>`: identifies `<state list>`.
  Example: `iptables -m state --state RELATED,ESTABLISHED ...` will apply the rule to the packets of the connections in one of these states.

Part IV

Lab 3: Firewall configuration

Outline

The DNAT rule

```
iptables [-t <table>] <command> <expression> -j DNAT
```

- Used with the option `--to-destination`.
- Can be put only in the nat table of the `PREROUTING` and `OUTPUT` chains.
- An address or a range of addresses can be specified (and the connections will be randomly distributed among the specified addresses, in other words, a load-balancing will be done).
- In case of an expression TCP or UDP a port or a range or ports can also be indicated (to distribute the connections among the range of ports).
  Example:

```
~# iptables -t nat -A PREROUTING -p tcp -d 200.10.10.10 --dport 80 \
~#      -j DNAT --to-destination 192.168.10.10-192.168.10.20:80-100
```

Types of rules

The SNAT rule

```
iptables [-t <table>] <command> <expression> -j SNAT
```

- Used with the option `--to-source`.
- Can be put only in the nat table of the `POSTROUTING` chain.
- An address or a range of addresses can be specified (and the connections will be randomly distributed among the specified addresses, in other words, a load-balancing will be done).
- In case of an expression TCP or UDP a port or a range or ports can also be indicated (to distribute the connections among the range of ports).
  Example:

```
~# iptables -t nat -A POSTROUTING -p tcp -o ppp0 \
~#  -j SNAT --to-source 200.10.10.10-200.10.10.20:1024-32000
```

Types of rules

The Masquerade rule

```
iptables [-t <table>] <command> <expression> -j MASQUERADE
```

- used as the SNAT target, but it does not require any `--to-source`: the address of an interface is applied to all outgoing packets.
  Example:

```
~# iptables -t nat -A POSTROUTING -p tcp -o ppp0 -j MASQUERADE --to-ports 1024-20000
```

Part IV

Lab 3: Firewall configuration

Outline

Quick edition

Saving iptables configuration to file

- The command:

```
~# iptables-save > cc
```

  dumps all the configuration command that iptables is executing and readdress them to the file `cc`.
- You can edit this file and replace the new configuration of iptables with the command:

```
~# iptables-restore cc
```

Part IV

Lab 3: Firewall configuration

Outline

Lab setup

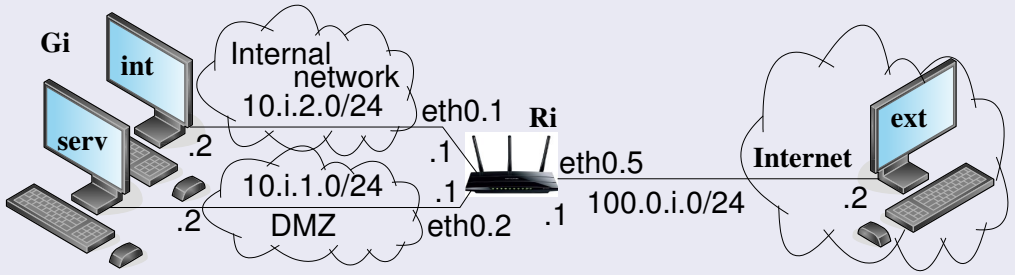
Install Apache server

1. Install Apache server in the PCs. It will be used as web server.

```
~# apt-get update
~# apt-get install apache2
```

Lab setup

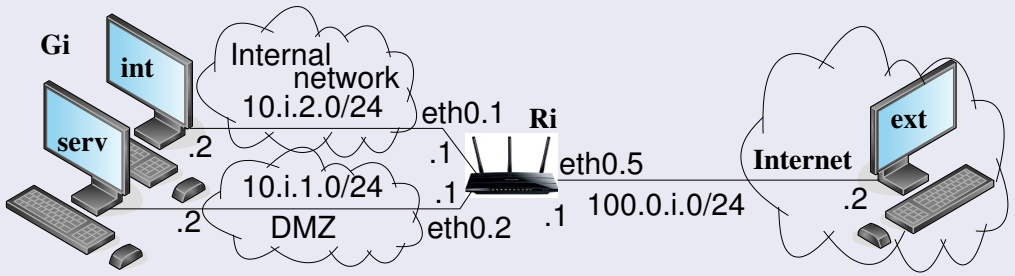
Configuration of the network



1. Each group i, $i = 1, \cdots 10$ will configure the network of the figure.
2. Disable the default OpenWrt firewall configuration.
3. The PC `ext` represents a host in the Internet. `ext` must be configured with no default route such that it can only reach the public address of the router `Ri`.
4. The PC `serv` represents a server in the DMZ.
5. The PC `int` represents a host in the internal network.
6. Configure the network DMZ and the Internal network. Verify that the router `Ri`, `serv` and `int` can reach each other, but `ext` cannot reach `serv` and `int`.

Lab setup

SNAT



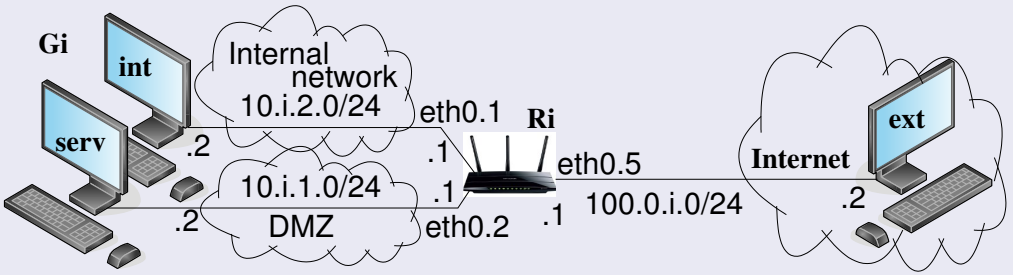
1. Execute the following command in `Ri`:

```
Ri:~# iptables -t nat -A POSTROUTING -o eth0.5 -j SNAT --to-source 100.0.i.1
```

2. Check that now `serv` and `int` can reach `ext` but not vice-versa, why? Check by executing `tcpdump` in `eth0.1`, `eth0.2` and `eth0.5` of `Ri` how the firewall change the IP addresses.

Lab setup

DNAT

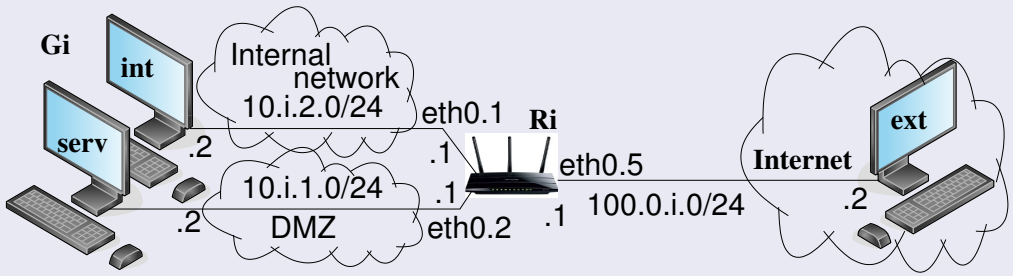


1. Execute the following command in `Ri`:

```
Ri:~# iptables -t nat -A PREROUTING -p tcp -i eth0.5 -d 100.0.i.1 --dport ssh \
Ri:~#    -j DNAT --to-destination 10.i.1.2
```

2. Check that now `ext` can connect to the ssh server of `serv` (for example, by executing `ssh root@100.0.i.1`), but if it pings to `serv` it doesn't reply, why?
3. Check that `ext` has not access to any other service of the `serv` (for example, web).
4. Which is the rule that should be executed in order to get to ping from `ext` to `serv`?, and to get to connect to the web server of `serv`? Try the rules.

Default rules

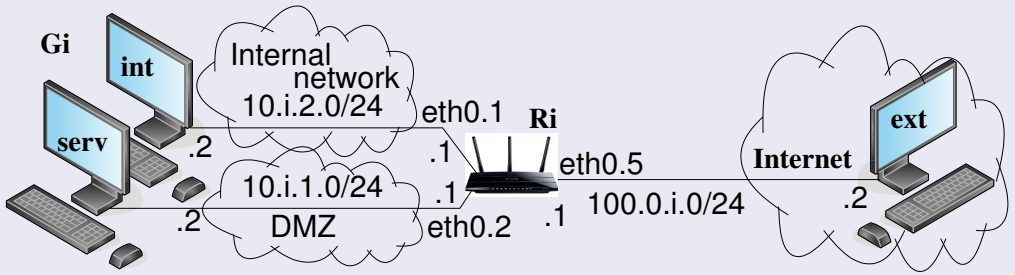


1. Execute the commands in `Ri` (do not delete the previous ones):

```
Ri:~# iptables -P INPUT DROP
Ri:~# iptables -P OUTPUT DROP
Ri:~# iptables -P FORWARD DROP
```

2. Check that none of the PCs is now reachable. Why?

Forwarding rules



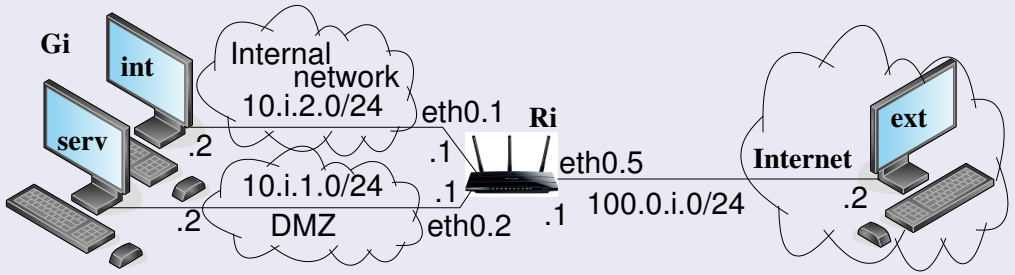
1. Execute the commands in `Ri`:

```
Ri:~# iptables -A FORWARD -i eth0.2 -o eth0.5 -j ACCEPT
Ri:~# iptables -A FORWARD -i eth0.5 -o eth0.2 -m state \
Ri:~#     --state ESTABLISHED,RELATED -j ACCEPT
```

2. Check that now `serv` can ping to `ext` but not vice-versa. Why?
3. Check that now `serv` can access to any of the services of `ext` (for example, try ssh) but not vice-versa. Why?

Lab setup

Forwarding rules

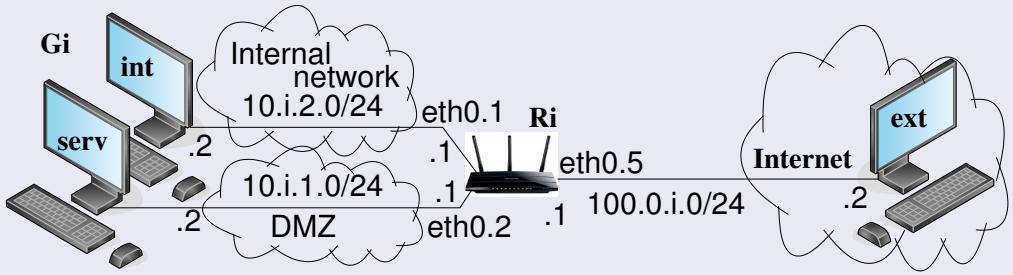


1. Execute the command in `Ri`:

```
Ri:~# iptables -A FORWARD -p tcp -i eth0.5 -o eth0.2 -d 10.i.1.2 -j ACCEPT
```

2. Check that now that `ext` can connect to the ssh server of `serv` but `int` cannot. Why?
3. Check that `ext` only can access to the ssh server of `serv` (check for example that `ext` cannot access the web server of `serv`), why?
4. Configure the firewall in order to get `ext` to ping `serv`.

Forwarding rules



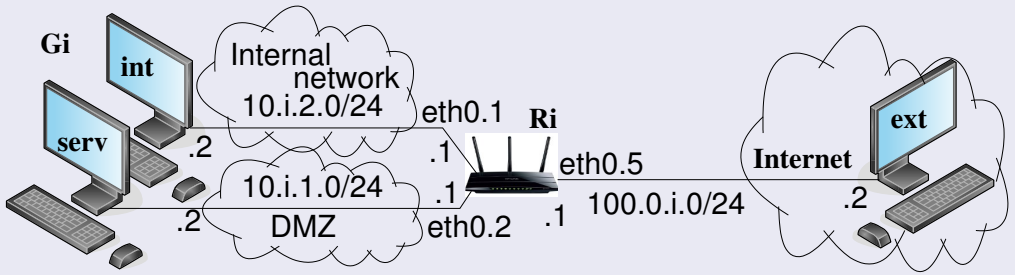
1. Execute the commands in `Ri`:

```
Ri:~# iptables -A FORWARD -i eth0.1 -o eth0.2 -j ACCEPT
Ri:~# iptables -A FORWARD -i eth0.2 -o eth0.1 -j ACCEPT
```

2. Now `serv` and `int` can see each other, but they donÃ¢Â€Â™t reach `Ri`. Why?

Lab setup

Forwarding rules



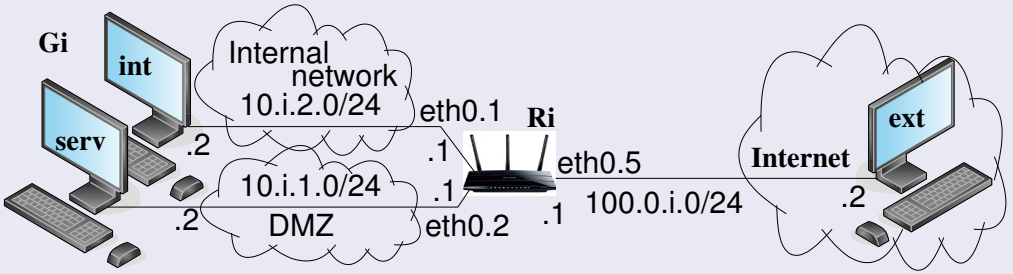
1. Execute the commands in `Ri`:

```
Ri:~# iptables -A INPUT -p icmp -i eth0.1 -d 10.i.2.1 -j ACCEPT
Ri:~# iptables -A OUTPUT -p icmp -o eth0.1 -s 10.i.2.1 -m state \
Ri:~#    --state ESTABLISHED,RELATED -j ACCEPT
```

2. Now `int` can ping to `Ri`, but cannot have access to any of the services of `Ri`. Why? Execute the commands that are necessary to get `serv` to have the same restrictions when accessing to `Ri` than those that `int` has.

Lab setup

Firewall design



Remove the previous configuration of the firewall and design a configuration that meets the following:

1. `int` is the only host which can connect to the ssh server of `Ri`. `int` has not to be able to connect to any other service of `Ri`.
2. `int` is able to connect to the ssh server of `ext`, but neither any other of the services of `ext` nor to any other host of the Internet. `ext` must not be able to start a connection to `int`, nor even to do a ping.
3. `ext` is able to connect to the web server of `serv`, and ping `serv`, but neither any other of the services of `serv` nor to any other host of the internal network.

Part IV

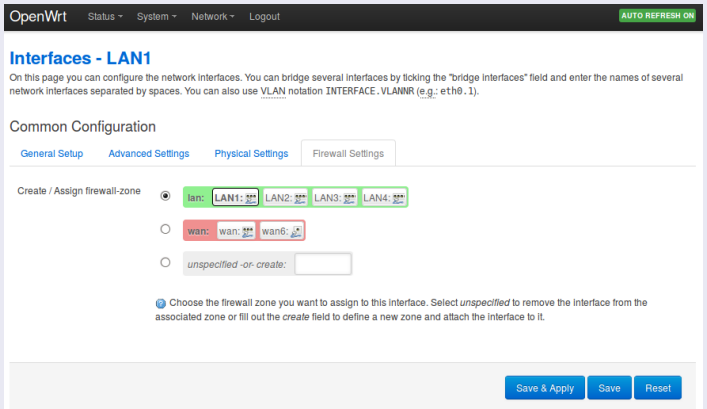
Lab 3: Firewall configuration

Outline

Firewall configuration using the web interface

1. Using the WI, assign `LAN1`, `LAN2`, `LAN3` and `LAN4` to the lan firewall-zone:

   `Network -> Interfaces -> Edit`



2. Start the OpenWrt firewall (`System -> Startup -> firewall -> Enable -> Start`) and use `iptables-save` to observe the configuration.
3. Use the OpenWrt firewall WI (`Network -> Firewall`) to setup some options (e.g. Masquerading), and check the iptables configuration changes with `iptables-save`.