UNIVERSITY OF ZAGREB
FACULTY OF ELECTRICAL ENGINEERING AND COMPUTING

SEMINAR

iproute2 and iptables packet

Neven Miculinic
Mentor:

Zagreb, January 2018.

CONTENTS

1. Introduction

Networking is one of the most important topic in everyday computer use. Almost every meaningful action we do in digital forencis, sooner or later involves networkings. Whether it's simply performing backups, viewing facebook messages, sending emails, or accessing databases and utilizing VPN/SSH. As in any complex system, one surely describing networking, many things may go wrong and multiple attacks are possible. For the computer forensics purpose, this essay describes bacis linux networking primities, from the time network packet enters the machine, reaches local process, and exits the machine.

It describes two tools consisting used most commonly in Linux networking. First is iptables, part of Netfilter project. Netfilter net is a framework providing various kernel hooks within network stack allowing user to modify and alter network packages. IPtables is their most commonly used utility. It shall be decribed in more detail in following chapters.

Iproute she is a collection of userspace utilities for controlling and monitoring various aspects of networking in the Linux kernel, including routing, network interfaces, tunnels, traffic control, and network-related device drivers. In this essay the focus in only on routing, and just a brief introduction and basic/most common commands.

2. iptables

This chapter desvribes packet path through various iptables tables and chains. The rest of the chapter is dedicated for explaining basic ip tables concept, with next chapter showing various application. Visual overview can be seen in figure 2.1.

2.1. Rule

Iptables have various rule, when matches their target is executed. They function as if-then construct. The most common rule 'ifs' are source/destination address, protocol and/or interface. They can be combined with and/or clauses. Furthermore any valid BPF BPF bytecode can be rule 'if'

The most common rule targets are ACCEPT, DROP, REJECT ones which perform packet filtering. In NAT table, common ones are DNAT, SNAT and MASQUERADE which perform IP:port NATing. NAT related targets are explained in section 3.1. Other common rule targets are LOG and jump to another chain.

2.2. Chain

Chain is a list of rules which are matched in order. Rule can be terminal (most of them) or nonterminal (e.g. LOG, ULOG). Upon reaching the terminal rule (e.g. DROP) chain has reached its end. Chain can have it's default policy (e.g. DROP for table filter in INPUT chain). There are two types of chains – system (PREROURING, INPUT, FORWARD, OUTPUT, POSTROUTING) and user defined chains. User defined chains server se tar-

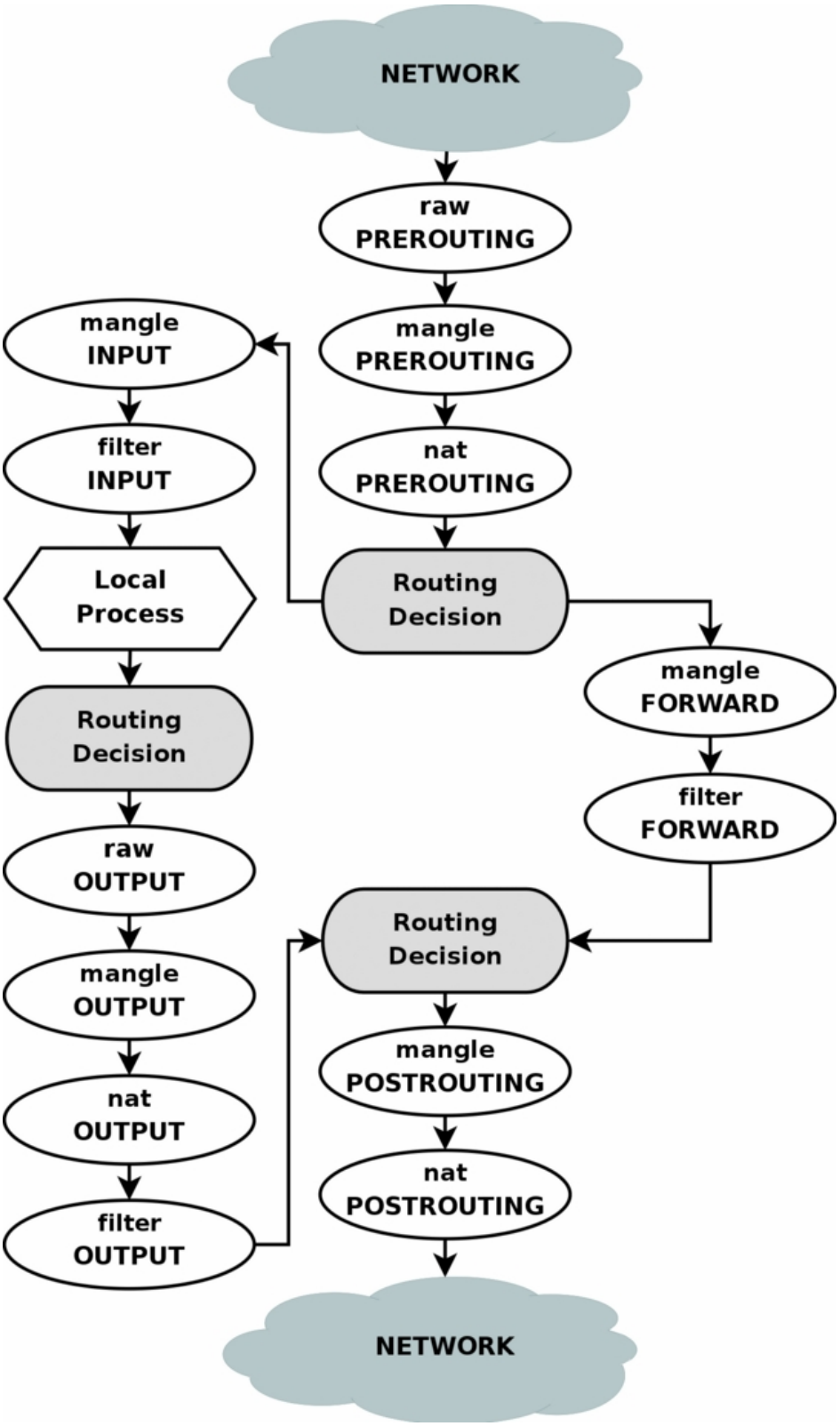

Figure 2.1: Overview of iptables. The lowercase word on top is the table and the upper case word below is the chain. Source Ipt

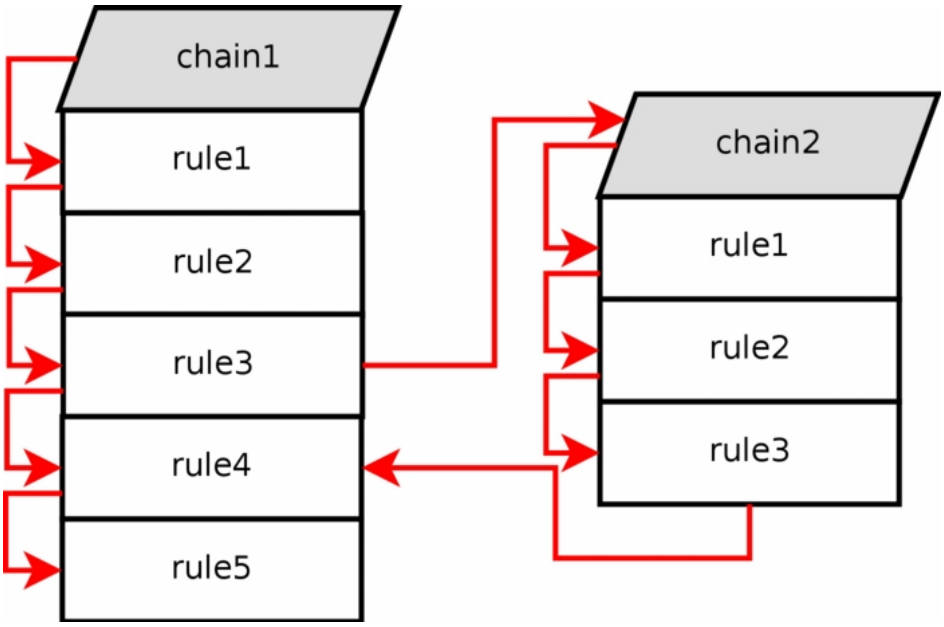

Figure 2.2: Table chain subtraverstion. Source Ipt

get jump within the same table (e.g. jump to user defined chain). They are created with `iptables -N <chain_name> -t <table name [filter default]>`. Chain traversal is depicted in figure 2.2. System chains are:

- PREROURING – Packet arriving in the kernel before any routing
- INPUT – Packet is destined for the local process
- FORWARD – Packet isn't destined for the local process
- OUTPUT – Packet originated from local process
- POSTROUTING – Packet departing from the machine after all routing takes place

Refer to figure 2.1 for their interaction.

2.3. Tables

Tables are bread and butter of this package. Each table defines specific hooks in the kernel for various system chains. They are associated with specific chain (that is NAT table in PREROURING and POSTROUTING are different). Each table is composed of multipe chains, what system defined ones, what user defined chains. There are 5 tables:

- raw – applied before any connection tracking takes place
- mangle – Mostly used for quality-of-service (QoS) header bit setting
- filter – packet filtering (DROP, ACCEPT and REJECT tagets)
- nat – NATing packages (DNAT, SNAT, MASQUERADE)

– security – packet marking (SECMARK, CONNSECMARK) for SELinux.

Refer to link **?** for more detail. In the following few subsections

2.4. Extensions

Iptables offers multiple modules you can use. You can view all installed modules by `ls -l /lib/iptables` and iptables will load all required modules dynamically. One of the most common ones is connection tracking.

2.4.1. Conection tracking

If connection tracking is enabled (and can be disabled in raw TABLE with `-j NOTRACK` for rule match) each packet can be in following states:

– NEW – first packet of the connection

– ESTABLISHED – both server and client have sent a package

– RELATED – related connection to an enstablished one. Protocol specific (e.g. there's FTP, IRC, etc. support in the kernel for RELATED connection tracking)

– INVALID – connection state cannot be determined

2.5. Routing tables

3. Example usecases

3.1. NAT

For NAT there are three specific targets related to NATing:

– SNAT – Source Network Address Translation. Exit packets source IP and port are rewriten to supplied source IP address. It is only valid in nat table within POSTROUTING chain. Downside is our source IP address must be known and static (or static range). Examples:

```
# Change source addresses to 1.2.3.4.
iptables -t nat -A POSTROUTING -o eth0 -j SNAT --to 1.2.3.4

# Change source addresses to 1.2.3.4, 1.2.3.5 or 1.2.3.6
iptables -t nat -A POSTROUTING -o eth0 -j SNAT \
    --to 1.2.3.4-1.2.3.6

# Change source addresses to 1.2.3.4, ports 1-1023
iptables -t nat -A POSTROUTING -p tcp -o eth0 -j SNAT \
    --to 1.2.3.4:1-1023
```

– DNAT – Destination Network Address Translation. It changes the package reciepeing, useful for servers behind firewall. It is only valid within nat table and PREROURING and OUTPUT chain. Examples:

```
iptables -t nat -A PREROUTING -p tcp -d 15.45.23.67 --dport 80 \
    -j DNAT --to-destination 192.168.1.1-192.168.1.10
```

– REDIRECT – DNAT but make destination local host. Only the port is changed.

```
iptables -t nat -A PREROURING -p tcp -o eth0 -j REDIRECT \
    --to-ports 1234
```

– MASQUERADE – it's similar to SNAT, but it doesn't require source IP address. It automatically grabs IP address information from sending interface. This is used in dymanically assigned IP connections. Example:

```
iptables -t nat -A POSTROUTING -p TCP -j MASQUERADE
```

3.2. Disabling internet access for specific device at specific time

For example you might have a really smart teen adicted to the internet. And you'd like disabling his internet access at the router level at certain times, while keeping rest functioning. It can be simly done with one iptables command and few extra modules

```
iptables -A PREROURING -m mac --mac-source 00:0F:EA:91:04:08 \
    -m time --timestart 9:00 --timestop 18:00 -j ACCEPT
iptables -A PREROURING -m mac --mac-source 00:0F:EA:91:04:08 \
    -j DROP
```

This is more efficient than IP filtering since you're probably running DHCP on your network dynamically assigning IP addresses. Nevertheless, it's easy for attacker (your teen) to figure out his MAC address is filterer, and to spoof it. Yet, hopefully by the time he figures it out, he'll already be a functional adult.

4. Bibliography

Bpf - the forgotten bytecode. https://blog.cloudflare.com/bpf-the-forgotten-bytecode/. (Accessed on 01/23/2018).

Iptables tutorial 1.2.2. https://www.frozentux.net/iptables-tutorial/iptables-tutorial.html#TRAVERSINGOFTABLES. (Accessed on 01/22/2018).

netfilter/iptables project homepage - the netfilter.org project. http://www.netfilter.org/. (Accessed on 01/23/2018).

shemminger/iproute2: Linux routing utilities. https://github.com/shemminger/iproute2. (Accessed on 01/23/2018).