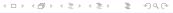


## nftables

Lukas Wais

CODERS.BAY, Linz Austria





- 1 Introduction
  - Definition
- 2 IPtables
  - Recap
- 3 nftables
  - Definition
  - nftables vs. IPtables
  - Examples





# What are IPtables?



### What are IPtables?

- Command line tool for configuring firewall rules
  - often combined with a frontend eg. kentfilter
- IPtables is able to inspect, modify or drop network packets
- The tables consist of **chains** which contain **rules** that are processed in the defined order
- Rules are conditions that have to be true
- All incoming packets are being processed by the very same rules
- 5 standard tables (raw, filter, NAT, mangle, security)







Definition

### What are nftables?

#### nftables

nftables is a framework for filtering packages. It was primarily created to replace the old IPtables, which had a number of performance and scalability issues.

- Merged into 3.13 Kernel (2014)
- combines all tools of the IPtables framework (iptables, ip6tables, arptables, ...) in a single tool







nftables vs. IPtables

# Advantages of nftables

- Easier to use maintain
- Better and easier syntax
- lacktriangle Compatibility layer o you can use old IPtables syntax even if filtering is internally done with nftables
- It does the same as IPtables, but with a different architecture







nftables vs. IPtables

# Advantages of nftables

- More efficient, especially for IPv6
  - Extra IP6tables is not necessary anymore; implemented within the nft-set per default
  - Same for arptables and ebtables
  - Less complexity and less code duplication
- Not only easier to write, also more efficient implemented in the Kernel





Below you will find rule which drops all packets to the destination 192.168.0.110.

nft add rule ip filter output ip daddr 192.168.0.110 drop

old IPtables rule \_\_\_\_\_\_iptables -A OUTPUT -d 1.2.3.4 -j DROP





Another example for the creation of a ruleset that allows packets to use different ports and allows different icmpv6 types:

```
nft ruleset

nft add rule ip6 filter input tcp dport {telnet, http, https} accept

nft add rule ip6 filter input icmpv6 type { nd-neighbor-solicit, echo-request, nd-router-advert, nd-neighbor-advert } accept
```





```
_____ Same ruleset with IPtables ____
```

ip6tables -A INPUT -p tcp -m multiport --dports 23,80,443 -j ACCEPT

ip6tables -A INPUT -p icmpv6 --icmpv6-type neighbor-solicitation -j ACCEPT

ip6tables -A INPUT -p icmpv6 --icmpv6-type echo-request -j ACCEPT

ip6tables -A INPUT -p icmpv6 --icmpv6-type router-advertisement -j ACCEPT

ip6tables -A INPUT -p icmpv6 --icmpv6-type neighbor-advertisement -j ACCEPT





### Redirect all ports to one port.

```
____nftables.conf _____
flush ruleset
table ip nat {
   chain prerouting {
       type nat hook prerouting priority 0;
        tcp dport != 22 redirect to 22
    chain postrouting {
       type nat hook postrouting priority 0;
```



Save and run config nft -f nftables.conf



# Questions?



