A 10 years journey in Linux firewalling

Pass the Salt, summer 2018
Lille, France

<pablo@netfilter.org>
Pablo Neira Ayuso

What is Netfilter?

- Not just… iptables

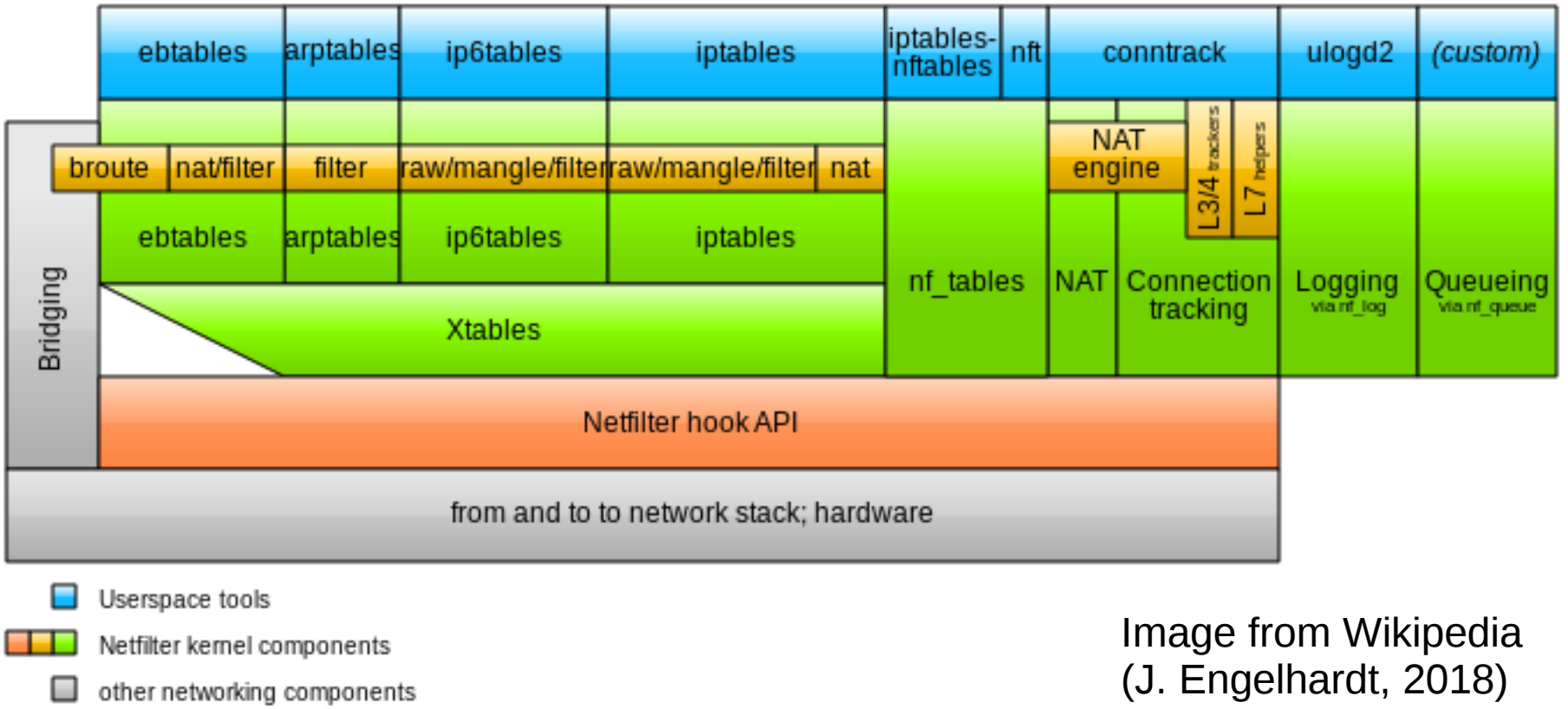


Image from Wikipedia
(J. Engelhardt, 2018)

A few dates…

- Fall 1998: Coding starts
- 1999 merged upstream (kernel 2.4.x)
- 2005: my first contributions
- 2006: Conntrack-tools / ctnetlink
- Coreteam member since 2007
- 2011 maintainership hand-over
- De Facto coreteam head since 2011.
- nftables is merged upstream 2013 (kernel 3.13)
  - nftables 0.9.0 release: June 2018

Why something new?

- One tool per family in userspace:

  ```
  # iptables -I INPUT -p tcp --dport 80 -j DROP
  # ip6tables -I INPUT -p tcp --dport 80 -j DROP
  # ebtables -I INPUT -p tcp --dport 80 -j drop
  # arptables …
  ```

- nft add rule *ip* filter input tcp dport 80 drop
  - Replace *ip* by *ip6, bridge, arp, inet*

Why something new? (2)

```
#!/usr/sbin/nft

include "another-ruleset.nft"


#
# Allowed NTP servers
#
define ntp_servers = { 84.77.40.132, 176.31.53.99, 81.19.96.148,
138.100.62.8 }

add rule ip foo bar ip saddr $ntp_servers udp dport 123 counter
```

Why something new? (3)

- ip6tables -A INPUT -p icmpv6 \
  --icmpv6-type packet-too-big -j ACCEPT
- ip6tables -A INPUT -p icmpv6 \
  --icmpv6-type neighbour-advertisement -j ACCEPT
- ip6tables -A INPUT -p icmpv6 \
  --icmpv6-type echo-reply -j ACCEPT
- nft add rule ip6 filter input icmpv6 type {\
  packet-too-big,
  time-exceeded, \
  echo-reply }

Why something new? (4)

- How many of you have ever used 'tc'?
  - tc filter add dev eth0 parent 1:0 protocol ip prio 10 u32 \
    match ip protocol 6 0xff \
    match u8 0x10 0xff at nexthdr+13 \
    match u16 0x0000 0xffc0 at 2 \
    police drop
    RTNETLINK answers: Invalid argument
- iptables from prerouting/raw:

  # iptables -I PREROUTING -t raw -p udp –dport 9 -j DROP

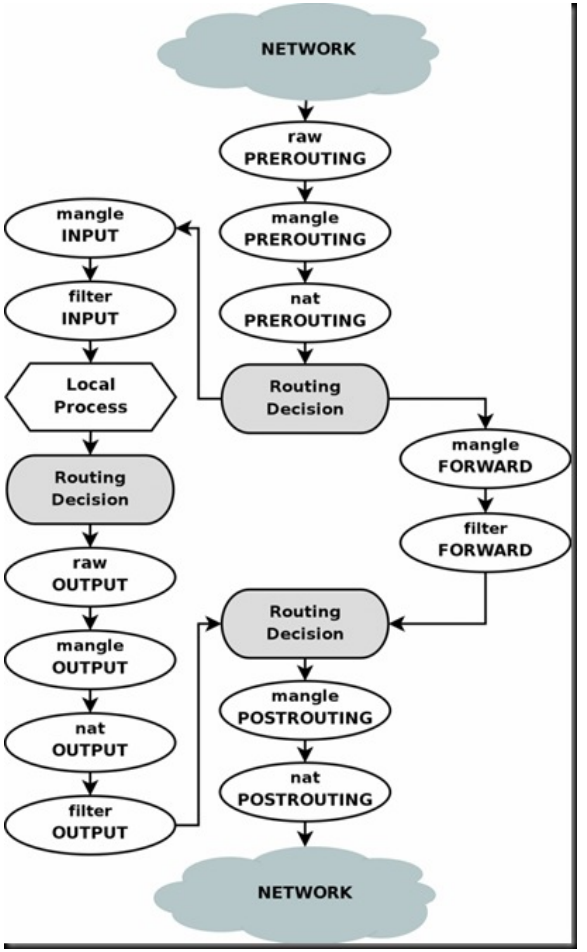
  5999928pps
- nftables from ingress (x2 faster):

  # nft add rule netdev ingress udp dport 9 drop

  12356983pps

  # nft add rule netdev ingress udp dport { 1, 2, …, 384} drop

  11844615pps



Rules

- nft add rule ip foo bar tcp dport != 80
- nft add rule ip foo bar tcp dport 1-1024
- nft add rule ip foo bar meta skuid 1000-1100
- nft add rule ip foo bar ip daddr 192.168.10.0/24
- nft add rule ip foo bar meta mark 0xffffff00/24
- nft add rule ip foo bar ct state new,established
- nft add rule ip foo bar ct mark and 0xffff == 0x123
- nft add rule ip foo bar ct mark set 10
- nft add rule ip foo bar ct mark set meta mark

Sets and maps

- nft add rule ip foo bar tcp dport { 22, 80, 443 } counter
- nft add set ip foo whitelist { type ipv4_addr \; }
  nft add rule ip foo bar ip daddr @whitelist counter accept
  nft add element ip foo whitelist { \
  192.168.0.1, \
  192.168.0.10 \
  }
- nft add table ip nat
  nft add chain ip nat post { \
  type nat hook postrouting priority 0\; }
  nft add rule ip nat post snat ip saddr map { \
  1.1.1.0/24 : 192.168.3.11 , \
  2.2.2.0/24 : 192.168.3.12 \
  }

Set timeouts

- nft add set ip foo whitelist { \
  type ipv4_addr; \
  timeout 1h; \
  }
- nft add element ip foo whitelist { \
  192.168.2.123, \
  192.168.2.124, \
  }
- nft add set ip foo whitelist { \
  type ipv4_addr; flags timeout; \
  }
- nft add element ip foo whitelist { 192.168.2.123 timeout 10s }
- nft add rule ip foo update @whitelist { ip saddr timeout 30s }

Dictionaries

- nft add chain ip foo tcp-chain
  nft add chain ip foo udp-chain
  nft add chain ip foo icmp-chain

- nft add rule ip foo bar ip protocol vmap { \
  tcp : jump tcp-chain, \
  udp : jump udp-chain, \
  icmp : jump icmp-chain
  }

Contenations

- nft add rule netdev foo bar \
  ether saddr . ip saddr . tcp dport { \
  c0:fe:00:c0:fe:00 . 192.168.1.123 . 80, \
  be:ef:00:be:ef:00 . 192.168.1.120 . 22} \
  counter accept
- nft add rule netdev foo bar ip saddr . tcp dport vmap { \
  192.168.1.123 . 22 : jump whitelist, \
  192.168.1.123 . 80 : jump whitelist, \
  }
- nft add set netdev foo bar { \
  type ether_addr . ipv4_addr \; }
- nft add element netdev foo bar { \
  00:ca:fe:00:be:ef . 192.168.1.123, \
  00:ab:cd:ef:00:12 . 192.168.1.124 \
  }

Comments

- nft add rule ip foo bar \
  ip daddr 8.8.8.8 counter accept\
  comment \“google dns\”
- nft add set ip foo dns-whitelist {\
  type ipv4_addr\;
  }
- nft add element ip foo dns-whitelist { \
  8.8.8.8 comment “google dns”, \
  192.203.230.10 comment “nasa dns”,
  }

Named objects

- Add new named counter
  nft add counter filter http-traffic
- Add new quota
  nft add quota filter http-traffic 25 mbytes
- nft add rule filter output \
  tcp dport https counter name http-traffic
- nft add rule filter output counter name tcp dport map { \
  443 : "https-traffic", \
  80 : "http-traffic", \
  22 : “ssh-traffic”, \
  25 : "smtp-traffic", \

  }

Named objects (2)

- Add map

```
nft add map filter mystats { \
        type ipv4_addr : counter \; }
```

- Reference it from rule

```
nft add rule filter input counter name \
        ip saddr map @mystats
```

- Add new counter objects to map

```
nft add counter filter computer1
nft add counter filter computer2
nft add element filter mystats { \
        192.168.2.3 : "computer1" }
nft add element filter mystats { \
        192.168.2.4 : "computer2" }
```

Flowtable bypass

- Upstream since 4.16.
- Configure flow bypass through **one single rule**:

```
table ip x {
    flowtable f {
        hook ingress priority 0; devices = { eth0, eth1};
    }
    chain y {
        type filter hook forward priority 0;
        ip protocol tcp flow add @f
    }
}
```

- Conntrack entries are owned by the flowtable:
  # cat /proc/net/nf_conntrack
  ipv4 2 tcp 6 src=10.141.10.2 dst=147.75.205.195 sport=36392 dport=443 src=147.75.205.195 dst=192.168.2.195 sport=443 dport=36392 **[OFFLOAD]** mark=0 zone=0 use=2

Flowtable bypass (2)

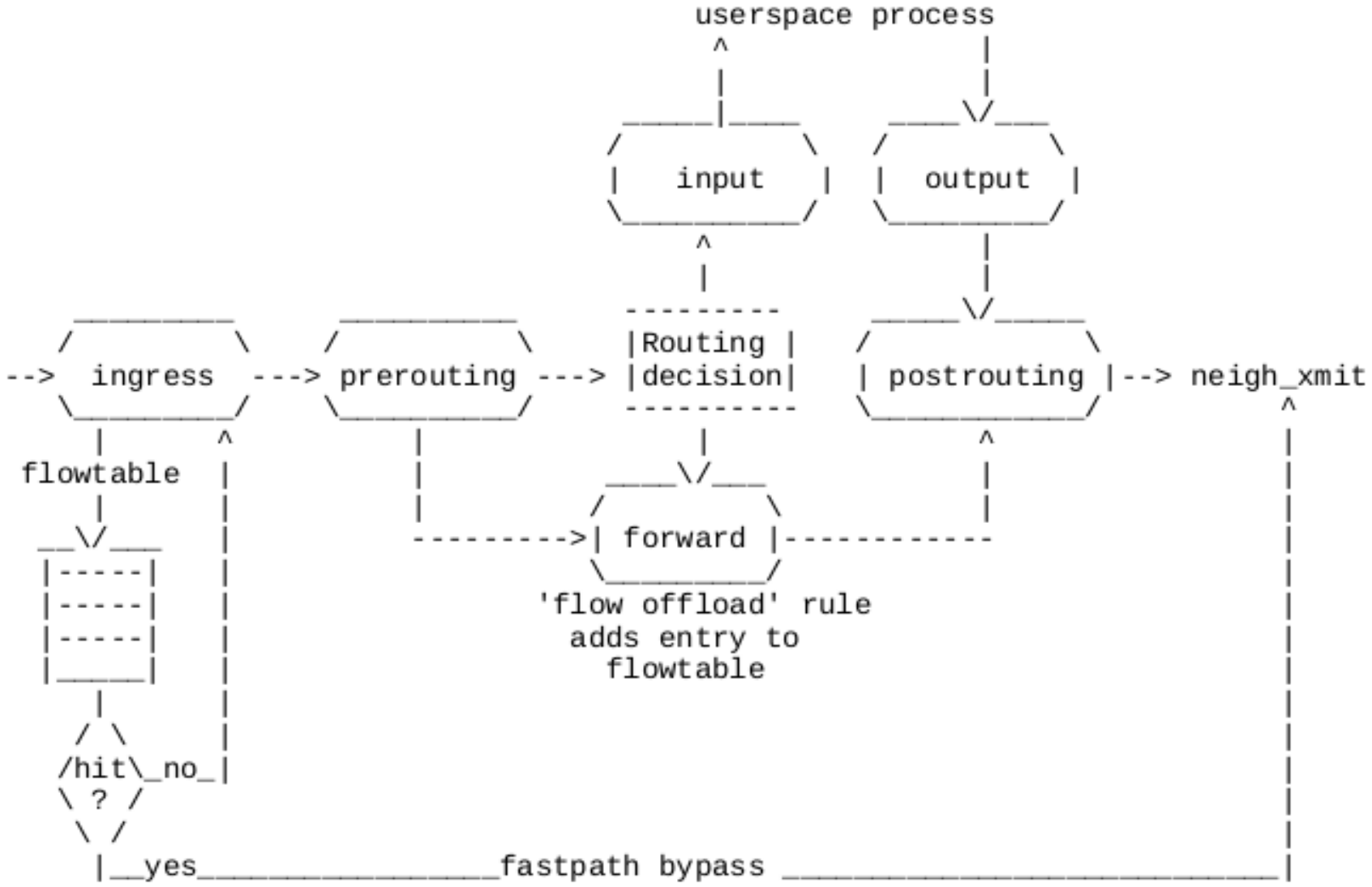


Fig.1 Netfilter hooks and flowtable interactions

Flowtable bypass (3)

- For each packet, extract tuple and perform look up at the flowtable.
  - Miss: Let the packet follow the classic forwarding path.
  - Hit:
    - Attach route from flowtable entry (… flowtable is acting as a cache).
    - If packet is over MTU, pass it up to classic forwarding path.
    - NAT mangling, if any.
    - Decrement TTL.
    - Send packet via neigh_xmit(...).
- Garbage collector:
  - Expire flows if we see no more packets after N seconds.

Flowtable bypass (4)

- Flow offload forward PoC in software is ~2.75 faster in software:
    - pktgen_bench_xmit_mode_netif_receive.sh to dummy device to exercise the forwarding path
        - One single CPU
        - Smallest packet size (worst case)
- Performance numbers:
    - Classic forwarding path (baseline): 1848888pps
    - Flow offload forwarding: 5155382pps

Flowtable bypass (5)

- Hardware offload infrastructure (~200 LOC) available.
- Not yet upstream, waiting for a driver.
- User enables explicitly “offload” flag to enable hardware offload.