Netfilter, NAT, Routing, Firewalls, NIDS

CSE 548 Spring 2023

jedimaestro@asu.edu

The netfilter.org project

What is the netfilter.org project?

The netfilter project is a community-driven collaborative FOSS project that provides packet filtering software for the Linux 2.4.x and later kernel series. The netfilter project is commonly associated with iptables and its successor nftables.

The netfilter project enables packet filtering, network address [and port] translation (NA[P]T), packet logging, userspace packet queueing and other packet mangling.

The netfilter hooks are a framework inside the Linux kernel that allows kernel modules to register callback functions at different locations of the Linux network stack. The registered callback function is then called back for every packet that traverses the respective hook within the Linux network stack.

iptables is a generic firewalling software that allows you to define rulesets. Each rule within an IP table consists of a number of classifiers (iptables matches) and one connected action (iptables target).

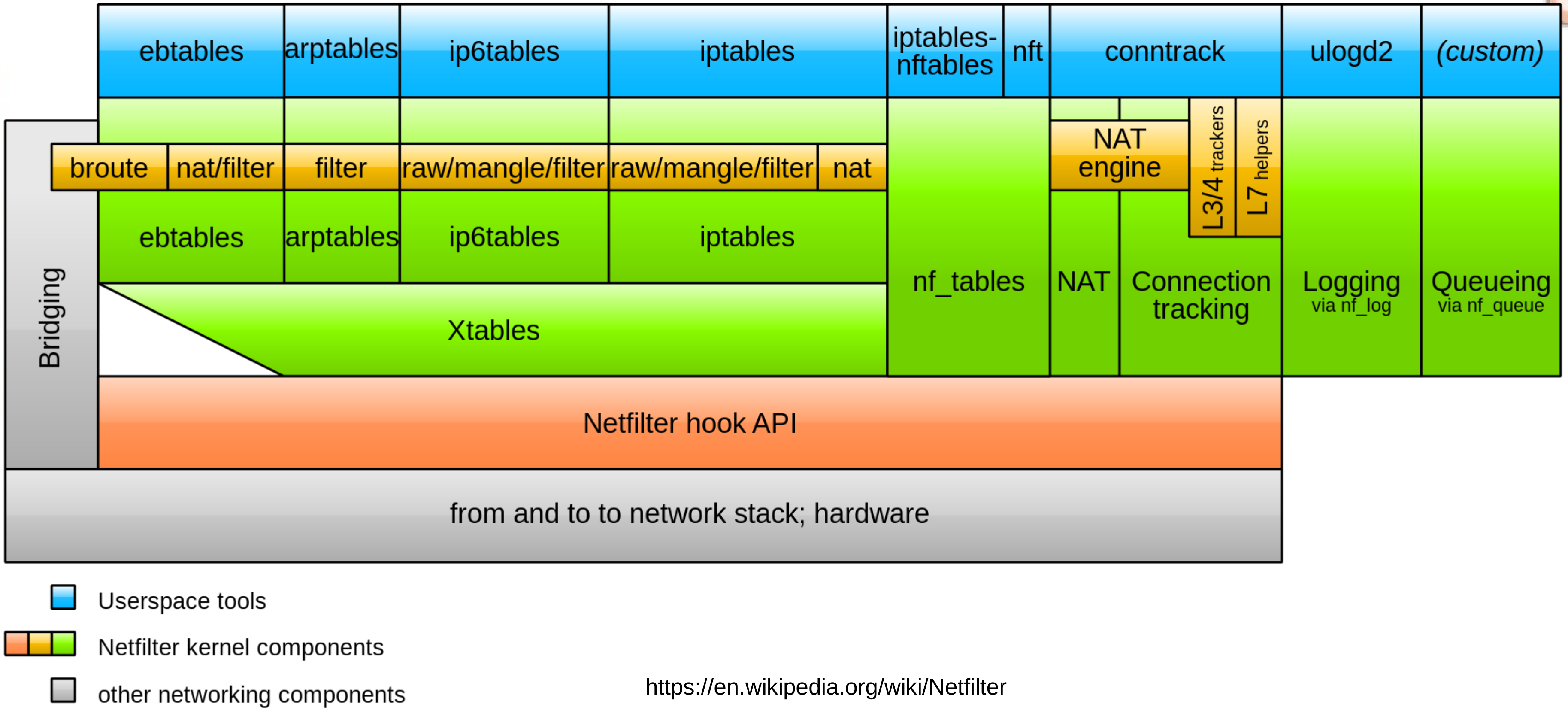
nftables is the successor of iptables, it allows for much more flexible, scalable and performance packet classification. This is where all the fancy new features are developed.

Alternatives to Netfilter

- FreeBSD
  - IPFW, natd, IPFILTER, PF
- Russia’s TSPU
  - Appears to be a custom implementation and not one of the above
- SOCKS proxies in user space, instead of NAT
  - Used by Tor, ShadowSocks, etc.
- Great Cannon? Great Firewall?
  - I don’t know one way or the other (maybe user space stack, or based on Linux, combination of both… who knows?)

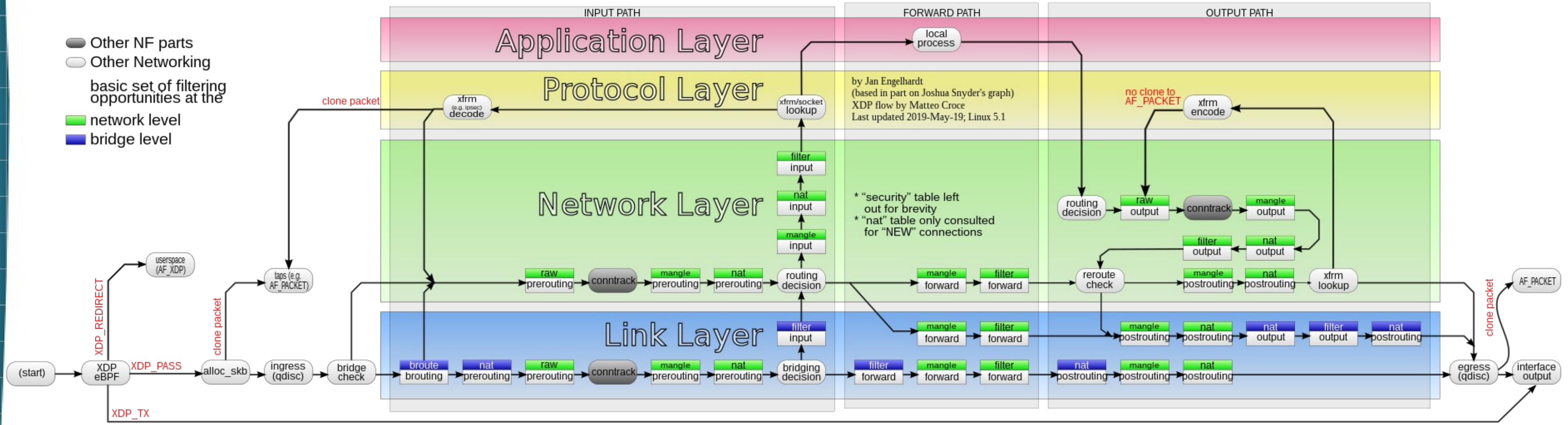
Netfilter components

Jan Engelhardt, last updated 2014-02-28 (initial: 2008-06-17)

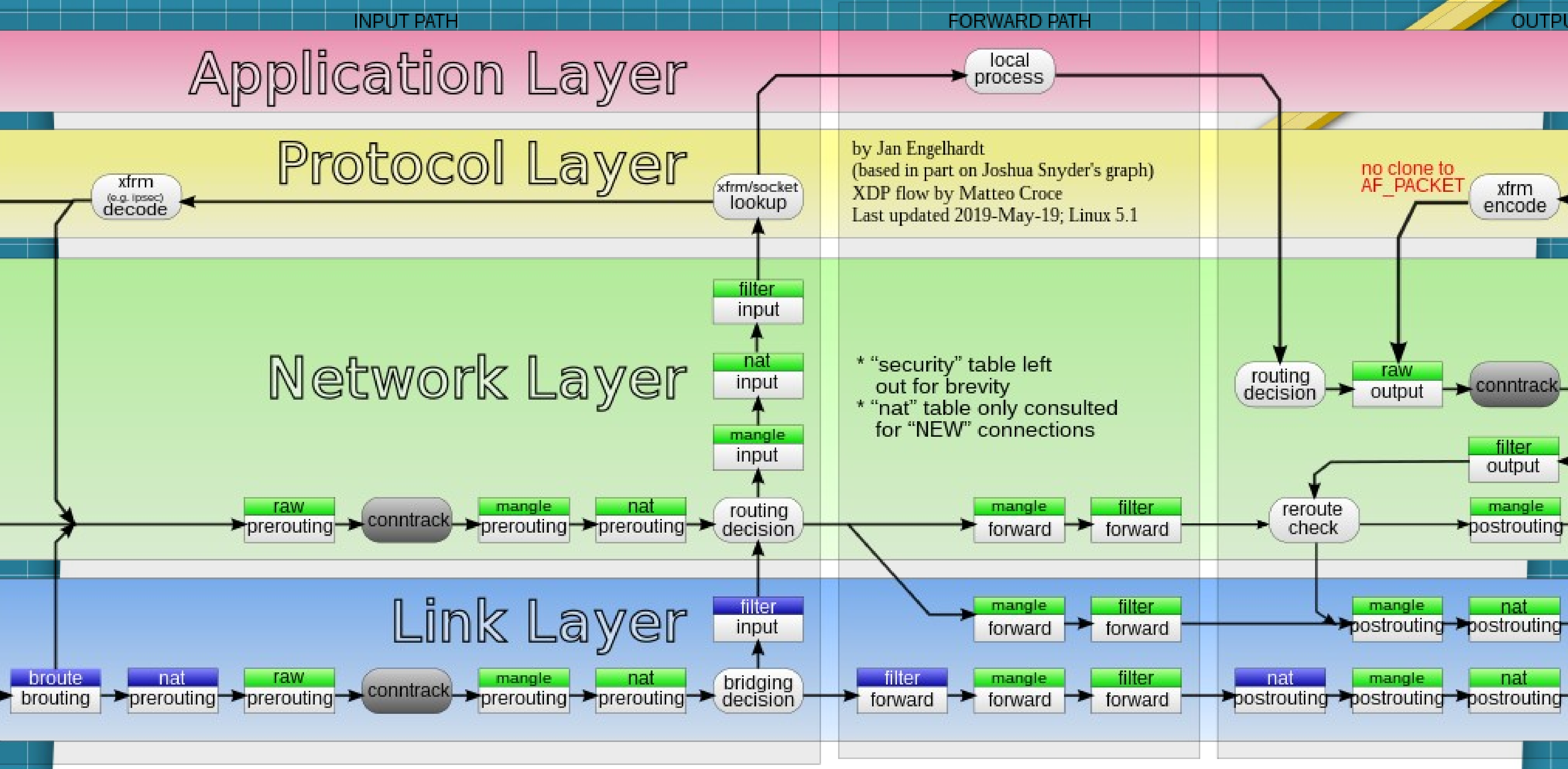


https://en.wikipedia.org/wiki/Netfilter

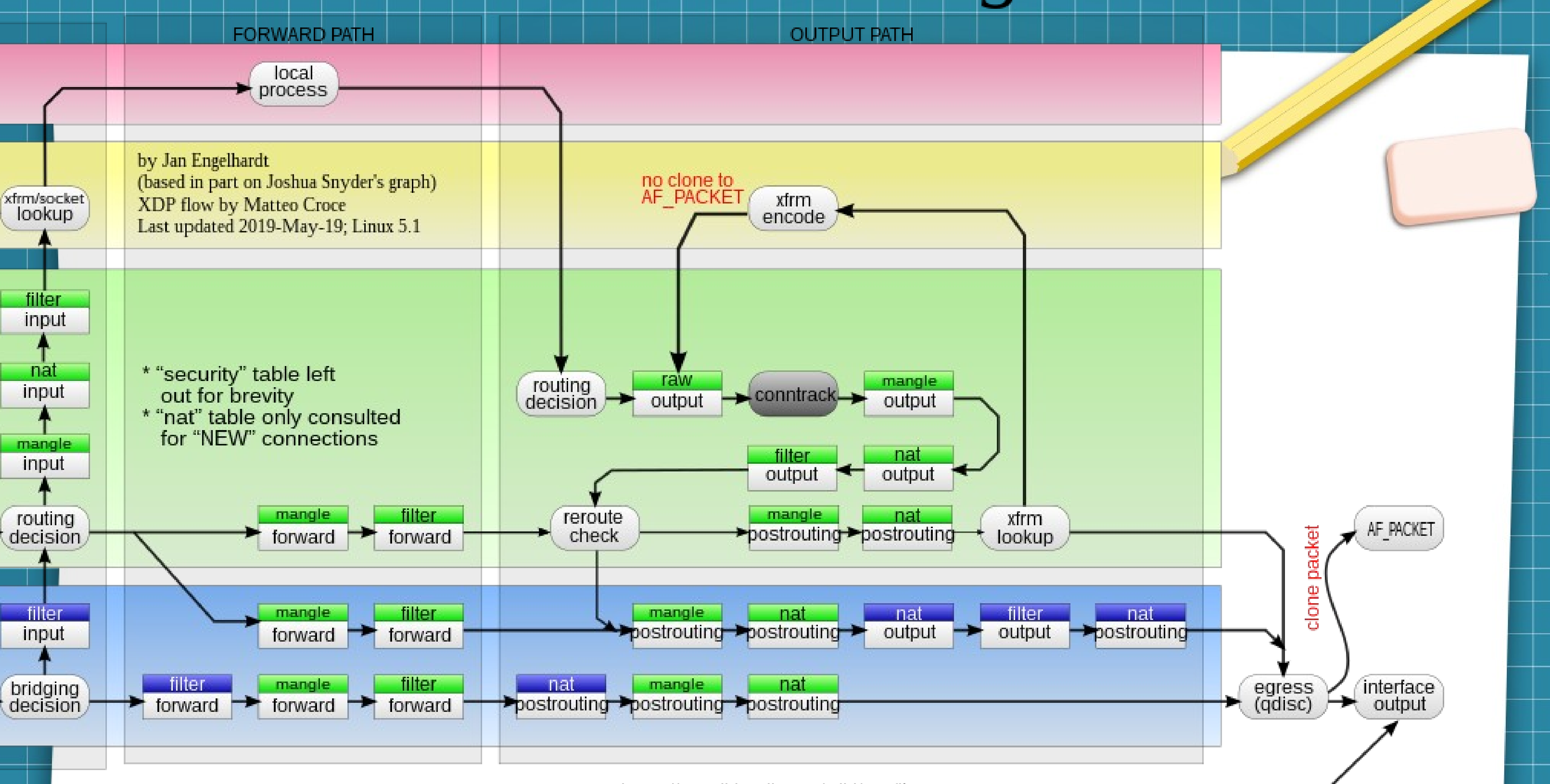
Packet flow in Netfilter and General Networking



https://en.wikipedia.org/wiki/Netfilter



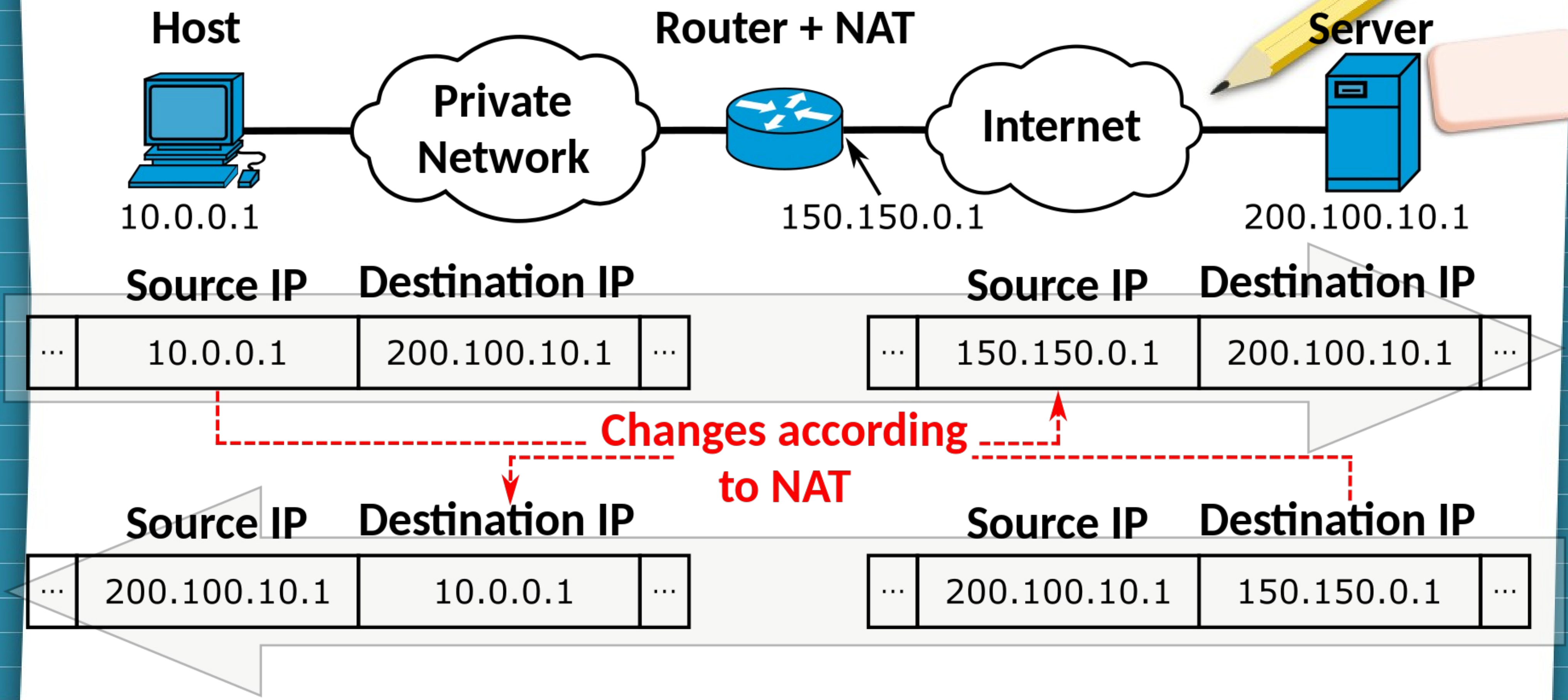
https://en.wikipedia.org/wiki/Netfilter



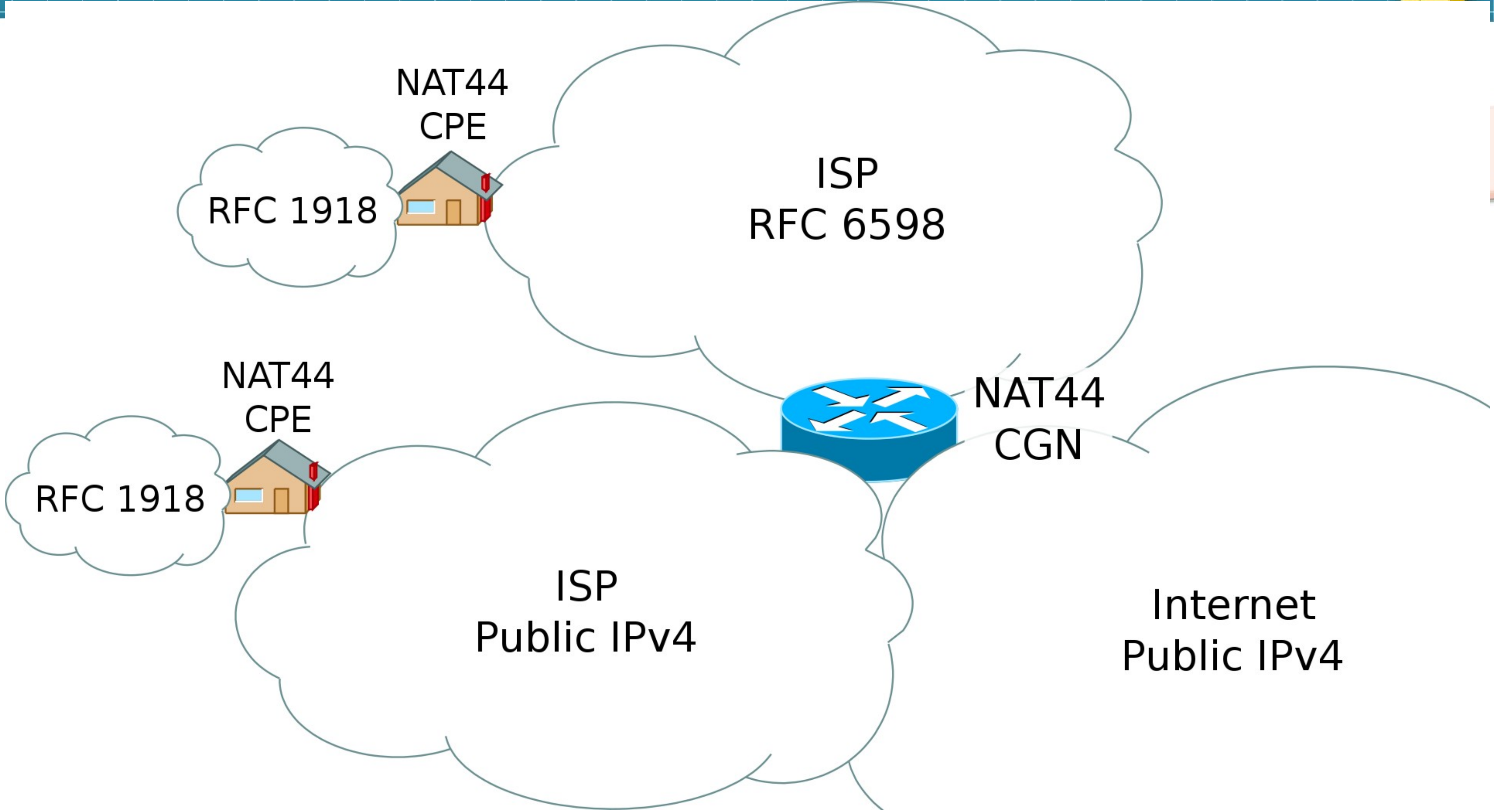
https://en.wikipedia.org/wiki/Netfilter

NAT == Network Address Translation

- Typically between private and public
  - 192.168.0.0/16, 10.0.0.0/8, and 172.16.0.0/12 are private
  - 127.0.0.0/24 is loopback
  - Most of everything else is public (*i.e.*, routable)
- Bogon filtering
  - Internet routers drop packets to/from private IPs (mostly)
- Most of the times you use the Internet you're going through multiple layers of NAT
  - *E.g.*, access points, VPNs, Carrier Grade NAT (CGNAT)



https://en.wikipedia.org/wiki/Network_address_translation#/media/File:NAT_Concept-en.svg



https://en.wikipedia.org/wiki/Carrier-grade_NAT#/media/File:CGN_IPv4.svg

Routing

- Also called “Layer 3” or “Network Layer”
- Internet routers build routing tables and apply them
  - Using, *e.g.*, BGP
- Hosts also have a lot of routing logic
  - Firewalls, munging, load balancing, multihoming, VPN, *etc.*

Normal Linux...

```
jedi@mariposa:~$ sudo ip route show
default via 192.168.69.250 dev enx7298ee2fab68 proto dhcp metric 100
169.254.0.0/16 dev enx7298ee2fab68 scope link metric 1000
192.168.69.0/24 dev enx7298ee2fab68 proto kernel scope link src 192.168.69.58 me
tric 100
```

```
jedi@mariposa:~$ ip rule show
0:      from all lookup local
32766:  from all lookup main
32767:  from all lookup default
jedi@mariposa:~$ ip route show table local
local 127.0.0.0/8 dev lo proto kernel scope host src 127.0.0.1
local 127.0.0.1 dev lo proto kernel scope host src 127.0.0.1
broadcast 127.255.255.255 dev lo proto kernel scope link src 127.0.0.1
local 192.168.69.58 dev enx7298ee2fab68 proto kernel scope host src 192.168.69.5
8
broadcast 192.168.69.255 dev enx7298ee2fab68 proto kernel scope link src 192.168
.69.58
```

```
ali-imran@aliimran:~$ sudo iptables -L
[sudo] password for ali-imran:
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
ali-imran@aliimran:~$ sudo iptables -t nat -A POSTROUTING -j MASQUERADE
ali-imran@aliimran:~$ sudo iptables -t nat -L
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination

Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

Chain POSTROUTING (policy ACCEPT)
target     prot opt source               destination
MASQUERADE  all  --  anywhere             anywhere
ali-imran@aliimran:~$ sudo sh -c "iptables-save > /etc/iptables/rules.v4"
ali-imran@aliimran:~$
```

https://linuxhint.com/configure-nat-on-ubuntu/

Normal Linux with a VPN...

```
jedi@mariposa:~$ sudo ip route
default via 172.18.11.149 dev tun0
82.102.24.216 via 192.168.69.250 dev enx7298ee2fab68
169.254.0.0/16 dev enx7298ee2fab68 scope link metric 1000
172.18.11.1 via 172.18.11.149 dev tun0
172.18.11.149 dev tun0 proto kernel scope link src 172.18.11.150
192.168.69.0/24 dev enx7298ee2fab68 proto kernel scope link src 192.168.69.58 me
tric 100
```

Android...

11:42 LTE

```
:/ # ip route
100.78.150.116/30 dev rmnet_data2 proto kernel scope lin
k src 100.78.150.117
```

```
:/ # ip rule list
0:      from all lookup local
10000:  from all fwmark 0xc0000/0xd0000 lookup legacy_sy
stem
11000:  from all iif lo oif dummy0 uidrange 0-0 lookup d
ummy0
11000:  from all iif lo oif rmnet_data0 uidrange 0-0 loo
kup rmnet_data0
11000:  from all iif lo oif rmnet_data1 uidrange 0-0 loo
kup rmnet_data1
11000:  from all iif lo oif rmnet_data2 uidrange 0-0 loo
kup rmnet_data2
16000:  from all fwmark 0x10063/0x1ffff iif lo lookup lo
cal_network
16000:  from all fwmark 0xd0001/0xdffff iif lo lookup rm
net_data0
16000:  from all fwmark 0xd0065/0xdffff iif lo lookup rm
net_data1
16000:  from all fwmark 0x10064/0x1ffff iif lo lookup rm
net_data2
17000:  from all iif lo oif dummy0 lookup dummy0
17000:  from all fwmark 0xc0000/0xc0000 iif lo oif rmnet
_data0 lookup rmnet_data0
17000:  from all fwmark 0xc0000/0xc0000 iif lo oif rmnet
_data1 lookup rmnet_data1
17000:  from all iif lo oif rmnet_data2 lookup rmnet_dat
a2
18000:  from all fwmark 0x0/0x10000 lookup legacy_system

19000:  from all fwmark 0x0/0x10000 lookup legacy_networ
k
20000:  from all fwmark 0x0/0x10000 lookup local_network

23000:  from all fwmark 0x64/0x1ffff iif lo lookup rmnet
_data2
31000:  from all fwmark 0x0/0xffff iif lo lookup rmnet_d
ata2
32000:  from all unreachable
```

ip rule list

iptables -L

```
:/ # iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

bw_INPUT   all  --  anywhere             anywhere

fw_INPUT   all  --  anywhere             anywhere


Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

oem_fwd    all  --  anywhere             anywhere

fw_FORWARD  all  --  anywhere             anywhere

bw_FORWARD  all  --  anywhere             anywhere

tetherctrl_FORWARD  all  --  anywhere             anywhe
re

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

nm_qti_filter_ssdp_dropper  all  --  anywhere
  anywhere
oem_out    all  --  anywhere             anywhere

fw_OUTPUT  all  --  anywhere             anywhere

st_OUTPUT  all  --  anywhere             anywhere

bw_OUTPUT  all  --  anywhere             anywhere


Chain bw_FORWARD (1 references)
target     prot opt source               destination

bw_costly_rmnet_data2  all  --  anywhere             any
where
bw_costly_rmnet_data2  all  --  anywhere             any
where

Chain bw_INPUT (1 references)
target     prot opt source               destination

bw_global_alert  all  --  anywhere             anywhere

bw_costly_rmnet_data2  all  --  anywhere             any
where
RETURN     esp  --  anywhere             anywhere

RETURN     all  --  anywhere             anywhere
      mark match 0x100000/0x100000
MARK       all  --  anywhere             anywhere
```

```
Chain fw_FORWARD (1 references)
target     prot opt source               destination


Chain fw_INPUT (1 references)
target     prot opt source               destination


Chain fw_OUTPUT (1 references)
target     prot opt source               destination


Chain nm_mdmprxy_doze_mode_skip (0 references)
target     prot opt source               destination


Chain nm_mdmprxy_iface_pkt_fwder (0 references)
target     prot opt source               destination


Chain nm_qti_filter_ssdp_dropper (1 references)
target     prot opt source               destination

DROP       udp  --  anywhere             anywhere
      udp dpt:1900
DROP       udp  --  anywhere             anywhere
      udp dpt:1900

Chain oem_fwd (1 references)
target     prot opt source               destination


Chain oem_out (1 references)
target     prot opt source               destination


Chain st_OUTPUT (1 references)
target     prot opt source               destination


Chain st_clear_caught (2 references)
target     prot opt source               destination


Chain st_clear_detect (0 references)
target     prot opt source               destination

REJECT     all  --  anywhere             anywhere
      connmark match  0x2000000/0x2000000 reject-with ic
mp-port-unreachable
RETURN     all  --  anywhere             anywhere
      connmark match  0x1000000/0x1000000
CONNMARK   tcp  --  anywhere             anywhere
      u32 "0x0>>0x16&0x3c@0xc>>0x1a&0x3c@0x0&0xffff0000=
```

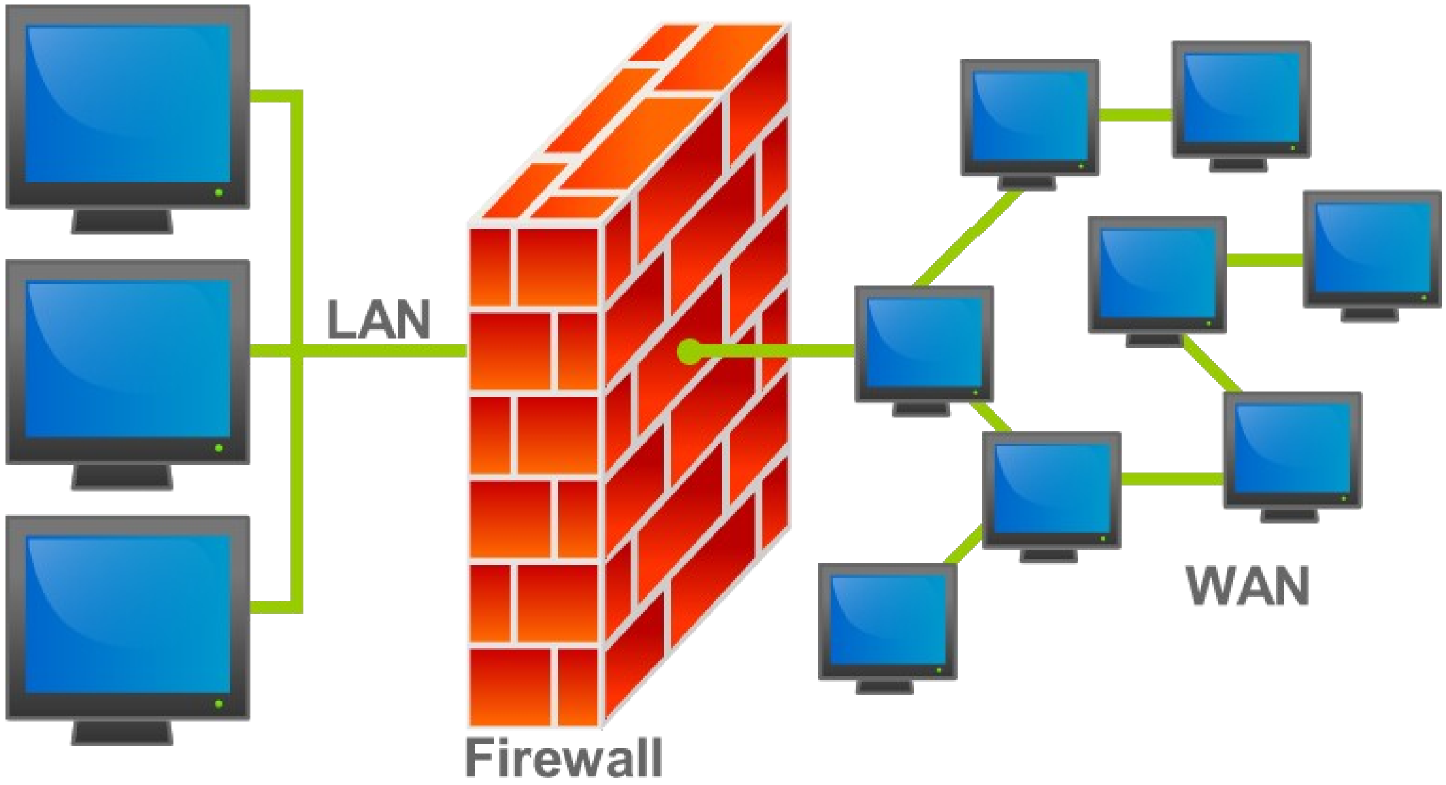
```
Chain bw_OUTPUT (1 references)
target     prot opt source               destination

bw_global_alert  all  --  anywhere             anywhere

bw_costly_rmnet_data2  all  --  anywhere             any
where

Chain bw_costly_rmnet_data2 (4 references)
target     prot opt source               destination

bw_penalty_box  all  --  anywhere             anywhere

REJECT     all  --  anywhere             anywhere
      ! quota rmnet_data2: 9223372036854775807 bytes  re
ject-with icmp-port-unreachable

Chain bw_costly_shared (0 references)
target     prot opt source               destination

bw_penalty_box  all  --  anywhere             anywhere


Chain bw_data_saver (1 references)
target     prot opt source               destination

RETURN     all  --  anywhere             anywhere


Chain bw_global_alert (2 references)
target     prot opt source               destination

           all  --  anywhere             anywhere
      ! quota globalAlert: 2097152 bytes

Chain bw_happy_box (1 references)
target     prot opt source               destination

RETURN     all  --  anywhere             anywhere
     match bpf pinned /sys/fs/bpf/netd_shared/prog_netd_
skfilter_allowlist_xtbpf
bw_data_saver  all  --  anywhere             anywhere


Chain bw_penalty_box (2 references)
target     prot opt source               destination

REJECT     all  --  anywhere             anywhere
     match bpf pinned /sys/fs/bpf/netd_shared/prog_netd_
skfilter_denylist_xtbpf reject-with icmp-port-unreachabl
e
bw_happy_box  all  --  anywhere             anywhere
```

```
Chain st_clear_caught (2 references)
target     prot opt source               destination


Chain st_clear_detect (0 references)
target     prot opt source               destination

REJECT     all  --  anywhere             anywhere
      connmark match  0x2000000/0x2000000 reject-with ic
mp-port-unreachable
RETURN     all  --  anywhere             anywhere
      connmark match  0x1000000/0x1000000
CONNMARK   tcp  --  anywhere             anywhere
      u32 "0x0>>0x16&0x3c@0xc>>0x1a&0x3c@0x0&0xffff0000=
0x16030000&&0x0>>0x16&0x3c@0xc>>0x1a&0x3c@0x4&0xff0000=0
x10000" CONNMARK or 0x1000000
CONNMARK   udp  --  anywhere             anywhere
      u32 "0x0>>0x16&0x3c@0x8&0xffff0000=0x16fe0000&&0x0
>>0x16&0x3c@0x14&0xff0000=0x10000" CONNMARK or 0x1000000
RETURN     all  --  anywhere             anywhere
      connmark match  0x1000000/0x1000000
st_clear_caught  tcp  --  anywhere             anywhere
            state ESTABLISHED u32 "0x0>>0x16&0x3c@0xc>>0
x1a&0x3c@0x0&0x0=0x0"
st_clear_caught  udp  --  anywhere             anywhere


Chain st_penalty_log (0 references)
target     prot opt source               destination

CONNMARK   all  --  anywhere             anywhere
      CONNMARK or 0x1000000
NFLOG      all  --  anywhere             anywhere


Chain st_penalty_reject (0 references)
target     prot opt source               destination

CONNMARK   all  --  anywhere             anywhere
      CONNMARK or 0x2000000
NFLOG      all  --  anywhere             anywhere

REJECT     all  --  anywhere             anywhere
      reject-with icmp-port-unreachable

Chain tetherctrl_FORWARD (1 references)
target     prot opt source               destination

DROP       all  --  anywhere             anywhere


Chain tetherctrl_counters (0 references)
target     prot opt source               destination

:/ #
```

```
Chain nm_qti_filter_ssdp_dropper (1 references)
target     prot opt source               destination

DROP       udp  --  anywhere             anywhere
      udp dpt:1900
DROP       udp  --  anywhere             anywhere
      udp dpt:1900
```

```
REJECT     all  --  anywhere             anywhere
      connmark match  0x2000000/0x2000000 reject-with ic
mp-port-unreachable
RETURN     all  --  anywhere             anywhere
      connmark match  0x1000000/0x1000000
CONNMARK   tcp  --  anywhere             anywhere
      u32 "0x0>>0x16&0x3c@0xc>>0x1a&0x3c@0x0&0xffff0000=
0x16030000&&0x0>>0x16&0x3c@0xc>>0x1a&0x3c@0x4&0xff0000=0
x10000" CONNMARK or 0x1000000
CONNMARK   udp  --  anywhere             anywhere
      u32 "0x0>>0x16&0x3c@0x8&0xffff0000=0x16fe0000&&0x0
>>0x16&0x3c@0x14&0xff0000=0x10000" CONNMARK or 0x1000000
RETURN     all  --  anywhere             anywhere
      connmark match  0x1000000/0x1000000
st_clear_caught  tcp  --  anywhere             anywhere
            state ESTABLISHED u32 "0x0>>0x16&0x3c@0xc>>0
x1a&0x3c@0x0&0x0=0x0"
st_clear_caught  udp  --  anywhere             anywhere
```

Firewall



https://en.wikipedia.org/wiki/Firewall_(computing)#/media/File:Firewall.png

Stateful *vs.* stateless

- Stateful has to refer to conntrack (or something like it)

```
ACCEPT     all  --  anywhere             192.168.122.0/24     ctstate RELATED,ESTABLISH
ED
ACCEPT     all  --  192.168.122.0/24     anywhere
```

- Stateless does not need to refer to conntrack

```
ACCEPT     all  --  8.8.8.8              anywhere
```

conntrack

```
marek@mrnew:~$
marek@mrnew:~$ sudo conntrack -L
tcp      6 431995 ESTABLISHED src=192.168.8.144 dst=85.10.202.207 sport=53370 dport=443 src=8
tcp      6 96 TIME_WAIT src=192.168.8.144 dst=216.58.201.174 sport=46610 dport=443 src=216.58
tcp      6 431996 ESTABLISHED src=192.168.8.144 dst=216.58.201.142 sport=56838 dport=443 src=
tcp      6 431993 ESTABLISHED src=192.168.8.144 dst=216.58.201.131 sport=58700 dport=443 src=
tcp      6 431977 ESTABLISHED src=192.168.8.144 dst=216.58.211.37 sport=36436 dport=443 src=2
tcp      6 431968 ESTABLISHED src=192.168.8.144 dst=216.58.201.138 sport=50936 dport=443 src=
tcp      6 431997 ESTABLISHED src=192.168.8.144 dst=64.233.184.189 sport=39562 dport=443 src=
tcp      6 431979 ESTABLISHED src=192.168.8.144 dst=172.217.168.174 sport=51622 dport=443 src
tcp      6 263 ESTABLISHED src=192.168.8.144 dst=216.58.211.37 sport=36428 dport=443 src=216.
tcp      6 431991 ESTABLISHED src=192.168.8.144 dst=172.217.168.174 sport=51570 dport=443 src
tcp      6 431996 ESTABLISHED src=192.168.8.144 dst=147.135.78.157 sport=39234 dport=443 src=
tcp      6 273 ESTABLISHED src=192.168.8.144 dst=172.217.17.10 sport=46478 dport=443 src=172.
tcp      6 431996 ESTABLISHED src=192.168.8.144 dst=216.58.201.131 sport=59140 dport=443 src=
tcp      6 431993 ESTABLISHED src=192.168.8.144 dst=52.44.211.134 sport=42430 dport=443 src=5
tcp      6 291 ESTABLISHED src=192.168.8.144 dst=172.217.16.238 sport=52550 dport=443 src=172
tcp      6 299 ESTABLISHED src=192.168.8.144 dst=74.125.140.189 sport=43698 dport=443 src=74.
tcp      6 263 ESTABLISHED src=192.168.8.144 dst=74.125.140.188 sport=43592 dport=5228 src=74
conntrack v1.4.4 (conntrack-tools): 17 flow entries have been shown.
```

https://blog.cloudflare.com/conntrack-tales-one-thousand-and-one-flows/

Network Intrusion Detection System

- NIDS
  - Can be in-path or on-path
  - Can be passive or active
    - Log a report, inject RSTs, drop, ...
  - Anomoly-based *vs.* rule-based
  - Sometimes the line between firewall and NIDS is not clear
    - Typically firewalls operate in layers 3 and 4 and are in-path, typically NIDS operates in layers 3 through 7 and are on-path

Snort rule examples from

https://cyvatar.ai/write-configure-snort-rules/ ...

Case 1: Securing Email Server With Snort Rules:

alert tcp 192.168.1.0/24 any -> 131.171.127.1 25 (content: “hacking”; msg: ”malicious packet”; sid:2000001;)

Case 2: Detecting TCP SYN Floods

Alert tcp any any -> 192.168.10.5 443 (msg: “TCP SYN flood”; flags:!A; flow: stateless; detection_filter: track by_dst, count 70, seconds 10; sid:2000003;)

Case 3: Securing your Network against Conficker A Worm

alert tcp any any -> any 445 (msg: “conficker.a shellcode”; content: “|e8 ff ff ff ff c1|^|8d|N|10 80|1|c4|Af|81|9EPu|f5 ae c6 9d a0|O|85 ea|O|84 c8|O|84 d8|O|c4|O|9c cc|IrX|c4 c4 c4|,|ed c4 c4 c4 94|&<O8|92|\;|d3|WG|02 c3|,|dc c4 c4 c4 f7 16 96 96|O|08 a2 03 c5 bc ea 95|\;|b3 c0 96 96 95 92 96|\;|f3|\;|24|i| 95 92|QO|8f f8|O|88 cf bc c7 0f f7|2I|d0|w|c7 95 e4|O|d6 c7 17 f7 04 05 04 c3 f6 c6 86|D|fe c4 b1|1|ff 01 b0 c2 82 ff b5 dc b6 1b|O|95 e0 c7 1 cb|s|d0 b6|O|85 d8 c7 07|O|c0|T|c7 07 9a 9d 07 a4|fN|b2 e2|Dh|0c b1 b6 a8 a9 ab aa c4|]|e7 99 1d ac b0 b0 b4 fe eb eb|”; sid: 2000002; rev: 1;)

Case 4: Alerts of Buffer Overflow in BIND

alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:”FTP wuftp bad file completion attempt [“;flow:to_server, established; content:”|?|”; content:”[“; distance:1; reference:bugtraq,3581; reference:bugtraq,3707; reference:cve,2001-0550; reference:cve,2001-0886; classtype:misc-attack; sid:1377; rev:14;)

Firewalls and Deep Packet Inspection (DPI)

- Dual use technology
  - Network access controls
  - Security monitoring and response
  - Load balancing
  - NAT
  - VPNs
  - Surveillance (“userid=*, longlat=*”)
  - Censorship (“falun”)
  - Throttling (“twitter.com”)
  - Targeted attacks (“HTTP GET cbjs.baidu.com/js/o.js”)