

Useful Links

- Redhat's NTable's documentation: https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8/html/configuring_and_managing_networking/getting-started-with-nftables_configuring-and-managing-networking
- Quick reference NTables in 10 minutes: https://wiki.nftables.org/wiki-nftables/index.php/Quick_reference-nftables_in_10_minutes
- NTables types: https://wiki.nftables.org/wiki-nftables/index.php/Data_types
- Golang nflog bindings <https://github.com/florianl/go-nflog>
- Golang userspace queuing bindings <https://github.com/AkihiroSuda/go-netfilter-queue>
- Rust nflog bindings <https://docs.rs/nflog/latest/nflog/>
- Rust userspace queuing bindings <https://docs.rs/nfqueue/latest/nfqueue/>

```
table inet firewall { # handle 32
    set imds_authorized { # handle 9
        type uid
        flags interval
        elements = { 1001 }
    }

    map named_map { # handle 10
        type inet_service : ipv4_addr
        elements = { 81 : 192.168.1.102, 8080 : 192.168.1.103 }
    }

    map named_vmap { # handle 11
        type ipv4_addr : verdict
        elements = { 192.168.0.10 : drop, 192.168.0.11 : accept }
    }

    set dns_rate_meter { # handle 12
        type ipv4_addr
        size 64
        flags dynamic,timeout
    }

    set concat_type { # handle 35
        type ipv4_addr . inet_service
        size 64
        flags dynamic,timeout
    }

    chain input { # handle 1
        type filter hook input priority filter; policy drop;
        iif "enp0s1" counter packets 8606 bytes 4263212 # handle 34
        iifname "wlp*" counter packets 0 bytes 0 # handle 33
        iif "enp0s1" counter packets 8624 bytes 4265356 # handle 32
        udp dport 53 jump incoming_dns # handle 16
        meta pkttype multicast log prefix "multicast" group 1 # handle 17
        meta pkttype broadcast log prefix "broadcast " log group 1 # handle 18
        ip protocol vmap { icmp : jump icmp-chain, tcp : jump tcp-chain, udp : jump udp-chain } # handle 19
        ct state established accept # handle 20
        ip saddr vmap @named_vmap # handle 21
        meta nfproto { ipv4, ipv6 } tcp dport 22 accept # handle 22
    }

    chain output { # handle 2
        type filter hook output priority filter; policy accept;
        ip daddr 169.254.0.0/16 meta skuid @imds_authorized log prefix "imds-authorized" group 2 counter packets 0 bytes 0 accept # handle 23
        ip daddr 169.254.0.0/16 log prefix "imds-unauthorized" group 3 counter packets 375 bytes 31500 drop # handle 24
        tcp dport 80 queue num 2 # handle 40
    }

    chain prerouting { # handle 3
        type nat hook prerouting priority filter; policy accept;
        dnath ip to tcp dport map { 80 : 192.168.1.100, 8888 : 192.168.1.101 } # handle 25
    }

    chain postrouting { # handle 4
        type nat hook postrouting priority filter; policy accept;
        snath ip to tcp dport map @named_map # handle 26
    }

    chain icmp-chain { # handle 5
        counter packets 0 bytes 0 # handle 27
    }

    chain tcp-chain { # handle 6
        counter packets 9009 bytes 4285590 # handle 28
    }

    chain udp-chain { # handle 7
        counter packets 74 bytes 10994 # handle 29
    }

    chain incoming_dns { # handle 8
        udp dport 53 add @dns_rate_meter { ip saddr timeout 1m } counter packets 0 bytes 0 accept # handle 30
        counter packets 0 bytes 0 log prefix "dns-rate-meter-breach" group 4 counter packets 0 bytes 0 drop # handle 31
    }
}

table inet flow_table { # handle 33
    flowtable f_t { # handle 6
        hook ingress priority filter
        devices = { dummy1, dummy0 }
    }

    chain forward { # handle 1
        type filter hook forward priority filter; policy accept;
        ip protocol { tcp, udp } flow add @f_t # handle 7
        ip6 nexthdr { tcp, udp } flow add @f_t # handle 8
        ct state established,related counter packets 0 bytes 0 return # handle 9
        ip protocol { tcp, udp } return # handle 10
        ip6 nexthdr { tcp, udp } return # handle 11
    }
}
```