Ministerstvo financií
Slovenskej republiky

EFQM
Recognised for excellence
5 star - 2012

Konfigurácia siete, firewall-u, VPN v OS Linux

RNDr. Jaroslav Janáček, PhD.

KPMG
cutting through complexity™

Obsah

- Základná konfigurácia siete v OS Linux
- Konfigurácia VLAN
- Konfigurácia bridge
- Netfilter – Linux firewall
- Policy routing
- VPN

Základná konfigurácia siete

- parametre sieťových rozhraní (interface)
  - IP adresa, maska, broadcast-ová adresa
- smerovacia tabuľka (routing table)
  - povolenie smerovania
- údaje pre prevod medzi menami a adresami
  - hosts, nsswitch.conf, DNS
- sieťové služby
  - protocols, services, inetd.conf

Konfigurácia sieťového rozhrania

- `ifconfig [meno]`
  - zobrazí základné parametre rozhrania (alebo všetkých aktívnych)
  - `-a` – všetky (t.j. aj neaktívne)
- `ifconfig meno param1 ...`
  - nastaví uvedené parametre

Konfigurácia sieťového rozhrania

- parametre pre ifconfig
  - *adresa* `netmask` *maska* `broadcast` *br_addr*
    - IP adresa, maska, broadcast-ová adresa
    - `ifconfig eth0 10.0.0.1 netmask 255.255.255.0 broadcast 10.0.0.255`
  - *adresa/dlzka*
    - dlzka = maska v tvare „počet bitov“
    - `ifconfig eth0 10.0.0.1/24`

Konfigurácia sieťového rozhrania

- parametre pre ifconfig
  - `up`
    - aktivuje interface
  - `down`
    - deaktivuje interface
  - `hw ether 00:11:22:33:44:55`
    - nastaví L2 (MAC) adresu
  - `mtu 1480`
    - nastaví MTU (max. veľkosť rámca)

Konfigurácia sieťového rozhrania

- parametre pre ifconfig
  - `[-]arp`
    - zapne/vypne (-) používanie ARP pre interface
  - `[-]promisc`
    - zapne/vypne príjem všetkých rámcov (normálne len rámce určené pre moju L2 adresu a L2 broadcast/multicast
  - `pointopoint` *`adresa`*
    - nastaví adresu druhej strany pre point-to-point linku
    - treba špecifikovať masku 255.255.255.255

Konfigurácia sieťového rozhrania

- parametre pre ifconfig
  - `add` *`adresa/dlzka`*
    - pridá IPv6 adresu
  - `del` *`adresa/dlzka`*
    - odstráni IPv6 adresu

Konfigurácia sieťového rozhrania

- vytvorenie aliasu rozhrania
  - na pridanie ďalšej IPv4 adresy
  - `ifconfig` **`eth0:1`** `1.2.3.4 ...`
- odstránenie aliasu
  - `ifconfig` **`eth0:1`** `down`

Konfigurácia sieťového rozhrania

- ifconfig automaticky vytvorí aj záznam v smerovacej tabuľke
  - pre sieť, ak je maska <32 bitov
  - pre 1 počítač v prípade masky 32 bitov a použitia pointopoint

Konfigurácia sieťového rozhrania

- arp
  - manipulácia s ARP cache – tabuľka používaná na získavanie L2 (napr. Ethernet) adresy pre IPv4 adresu
  - `arp [-n]` – vypíše tabuľku
  - `arp -i eth0 -s 1.2.3.4 00:01:02:03:04:05`
    - vytvorí permanentný záznam
  - `arp -i eth0 -d 1.2.3.4`
    - vymaže záznam z tabuľky

Konfigurácia sieťového rozhrania

- ip
  - súčasť balíka iproute2
  - univerzálny nástroj na konfiguráciu
    - sieťových rozhraní: `ip link, ip addr`
    - neighbour cache (ARP aj NDP): `ip neigh`
    - smerovacej tabuľky (IPv4 aj IPv6): `ip route`
    - tunelov: `ip tunnel`
    - ...

Konfigurácia sieťového rozhrania

- ip link
  - konfigurácia L2 parametrov

```
# ip link show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
mode DEFAULT
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
state UP mode DEFAULT qlen 1000
    link/ether 18:03:73:c1:16:89 brd ff:ff:ff:ff:ff:ff
```

Konfigurácia sieťového rozhrania

- `ip link set` *`meno`* `...`
  - `up | down`
    - zapne / vypne rozhranie
  - `arp on | off`
    - zapne / vypne ARP pre zariadenie
  - `promisc on | off`
    - zapne / vypne promiskuitný mód pre zariadenie
  - `mtu 1400`
    - nastaví MTU

Konfigurácia sieťového rozhrania

- `ip link set` *`meno`* `...`
  - `address 00:01:02:03:04:05`
    - nastaví L2 adresu rozhraniu
  - `name` *`novemeno`*
    - premenuje rozhranie

Konfigurácia sieťového rozhrania

- ip addr
  - konfigurácia IPv4/IPv6 adries (L3 parametre)

```
# ip addr list
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
state UP qlen 1000
    link/ether 18:03:73:c1:16:89 brd ff:ff:ff:ff:ff:ff
    inet 158.195.87.39/25 brd 158.195.87.127 scope global eth0
    inet6 fe80::1a03:73ff:fec1:1689/64 scope link
       valid_lft forever preferred_lft forever
```

Konfigurácia sieťového rozhrania

- `ip addr add ... dev` *`meno`*
  - `1.2.3.4/24`
    - IPv4 adresa a dĺžka masky
  - `broadcast 1.2.3.255`
    - broadcastová adresa
  - `broadcast +`
    - vypočíta broadcastovú adresu automaticky
  - `peer 3.3.3.3`
    - nastaví adresu druhej strany pre point-to-point linku

Konfigurácia sieťového rozhrania

```
# ip addr add 10.0.0.1/24 broadcast + dev eth0
# ip addr add 10.0.1.1 peer 10.0.1.2 dev eth0
# ip addr add 10.0.2.1 peer 10.0.2.0/24 dev eth0
# ip addr show dev eth0
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
state UP qlen 1000
    link/ether 18:03:73:c1:16:89 brd ff:ff:ff:ff:ff:ff
    inet 158.195.87.39/25 brd 158.195.87.127 scope global eth0
    inet 10.0.0.1/24 brd 10.0.0.255 scope global eth0
    inet 10.0.1.1 peer 10.0.1.2/32 scope global eth0
    inet 10.0.2.1 peer 10.0.2.0/24 scope global eth0
    inet6 fe80::1a03:73ff:fec1:1689/64 scope link
       valid_lft forever preferred_lft forever


10.0.0.0/24  proto kernel  scope link  src 10.0.0.1
10.0.1.2  proto kernel  scope link  src 10.0.1.1
10.0.2.0/24  proto kernel  scope link  src 10.0.2.1
```

Konfigurácia sieťového rozhrania

- `ip addr del 1.2.3.4/24 dev eth0`
  - odstráni adresu z rozhrania
- `ip addr add fdaa::1/64 dev eth0`
  - príklad ručného pridania IPv6 adresy na rozhranie

Konfigurácia sieťového rozhrania

- ip neigh
  - konfigurácia neighbour cache (ARP, NDP)
  - ip -4 neigh ...
    - IPv4 (ARP)
  - ip -6 neigh ...
    - IPv6 (NDP)
  - `ip neigh add 1.2.3.4 lladdr 00:01:02:03:04:05 dev eth0`
  - `ip neigh del 1.2.3.4 dev eth0`

Konfigurácia sieťového rozhrania

```
# ip neigh add 10.0.0.2 lladdr 00:01:02:03:04:05 dev eth0
# ip neigh add fdaa::2 lladdr 00:01:02:03:04:05 dev eth0
# ip neigh show
fdaa::2 dev eth0 lladdr 00:01:02:03:04:05 PERMANENT
158.195.87.115 dev eth0 lladdr d2:c5:01:00:00:07 STALE
158.195.87.126 dev eth0 lladdr 00:10:dc:ce:a3:0e REACHABLE
10.0.0.2 dev eth0 lladdr 00:01:02:03:04:05 PERMANENT
158.195.87.111 dev eth0 lladdr 00:10:dc:ce:61:60 REACHABLE
```

Perzistentné uloženie konfigurácie

- Debian / Ubuntu / ...
  - /etc/network/interfaces
- RedHat / Fedora / ...
  - /etc/sysconfig/network
    - spoločné parametre
  - /etc/sysconfig/network-scripts/ifcfg-eth0
    - parametre pre konkrétny interface

Konfig. siete - Debian

- /etc/network/interfaces
  - auto
    - medzerami oddelený zoznam interfacov, ktoré majú byť automaticky nakonfigurované
  - allow-hotplug
    - zoznam interfacov, ktoré majú byť automaticky nakonfigurované pri pripojení (vzniku)
  - iface *meno-interface rodina-protokolov metóda parametre*
    - konfigurácia konkrétneho interface-u

Konfig. siete - Debian

- metódy pre rodinu *inet*
  - loopback
    - pre lo
  - dhcp
    - získa konfiguráciu protokolom DHCP
  - static
    - address
    - netmask
    - broadcast
    - gateway

Konfig. siete - Debian

- spoločné parametre
  - pre-up *príkaz*
  - up | post-up *príkaz*
  - down | pre-down *príkaz*
  - post-down *príkaz*
- skripty v /etc/network/if-*param*.d/
  - vykonajú sa po vykonaní priamo uvedeného príkazu

Konfig. siete - Debian

- `ifup meno-interface-u`
  - nakonfiguruje zadaný interface
  - `-a` – všetky „auto“ interface-y
- `ifdown meno-interface-u`
  - odkonfiguruje zadaný interface
  - `-a` – všetky „auto“ interface-y

Smerovacia tabuľka

- route
- ip route

```
# route -n
Kernel IP routing table
Destination  Gateway          Genmask          Flags Metric Ref UseIface
0.0.0.0      158.195.87.126   0.0.0.0          UG    0      0      0 eth0
158.195.87.0 0.0.0.0          255.255.255.128  U     0      0      0 eth0
# ip route show
default via 158.195.87.126 dev eth0
158.195.87.0/25 dev eth0  proto kernel  scope link  src 158.195.87.39
# ip -6 route show
fe80::/64 dev eth0  proto kernel  metric 256
```

Smerovacia tabuľka

- `route add -host` *`adresa`* `[gw` *`router`*`] [dev` *`rozhranie`*`]`
  - pridá smerovanie na 1 počítač cez gw alebo priamo
- `route add -net` *`adresa`* `netmask` *`maska`* `[gw` *`router`*`] [dev` *`rozhranie`*`]`
  - pridá smerovanie na sieť
- `route del -host | -net ...`
  - odstráni záznam

Smerovacia tabuľka

- `ip route add` *`adresa/dlzka`*
  `[via` *`router`*`] [dev` *`rozhranie`*`]`
  - pridá smerovanie na počítač / sieť cez uvedený router alebo priamo
- `ip route del ...`
  - odstráni smerovanie
  - treba správne uviesť všetky potrebné parametre

Smerovacia tabuľka

- voľba zdrojovej adresy
  - `ip route add ... src` *`zdr_adresa`*
  - nastaví, z akej adresy budú štandardne odchádzať packety, ak budú smerované na základe tohto záznamu
  - užitočné, ak máme na rozhraní viac adries

Smerovacia tabuľka

```
# ip route add 10.0.1.0/24 via 10.0.0.2
# ip route add 10.0.2.0/24 dev eth0
# ip route add fdaa:1::/64 via fdaa::2
# ip route add fdaa:2::/64 dev eth0
# ip route show
default via 158.195.87.126 dev eth0
10.0.0.0/24 dev eth0  proto kernel  scope link  src 10.0.0.1
10.0.1.0/24 via 10.0.0.2 dev eth0
10.0.2.0/24 dev eth0  scope link
158.195.87.0/25 dev eth0  proto kernel  scope link  src 158.195.87.39
# ip -6 route show
fdaa::/64 dev eth0  proto kernel  metric 256
fdaa:1::/64 via fdaa::2 dev eth0  metric 1024
fdaa:2::/64 dev eth0  metric 1024
fe80::/64 dev eth0  proto kernel  metric 256
```

Povolenie smerovania

- IPv4
  - /proc/sys/net/ipv4/ip_forward
    - 1 – smerovanie zapnuté, 0 – vypnuté
- IPv6
  - /proc/sys/net/ipv6/conf/all/forwarding
    - 1 – smerovanie zapnuté, 0 – vypnuté

```
# echo 1 > /proc/sys/net/ipv4/ip_forward
```

Vybrané sieťové parametre IPv4

- /proc/sys/net/ipv4/conf/{all, eth0, ...}/
  - accept_redirects
  - send_redirects
  - proxy_arp
    - automatické proxy ARP pre adresy z inej siete
  - rp_filter
    - 0 – vypnutý
    - 1 – strict – zahodí packet, ak príde z iného rozhrania, než kam by bola smerovaná odpoveď
    - 2 – loose – zahodí packet, ak by nevedel poslať odpoveď

Vybrané sieťové parametre IPv6

- /proc/sys/net/ipv6/conf/{all, eth0, ...}/
  - forwarding
  - accept_ra
    - akceptovať RAdv (0 nie, 1 ak nie je router, 2 áno)
    - potrebné pre autokonfiguráciu
  - disable_ipv6
    - zakazuje IPv6 na rozhraní

Prevod medzi menami a adresami

- /etc/hosts
  - lokálna databáza mien a adries
  - `adresa meno alias ...`

```
127.0.0.1 localhost
10.0.0.1  pocitac.domena.sk   pocitac

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

Prevod medzi menami a adresami

- súbor `/etc/resolv.conf`
  - obsahuje nastavenie DNS resolvera (klienta)
  - `domain lokálna_doména`
  - `search dom1 dom2 ...`
    - domain a search sa navzájom vylučujú
    - určujú domény, v ktorých sa skúšajú hľadať neúplné mená
  - `nameserver IP_adresa`
    - určuje DNS server, ktorého sa resolver bude pýtať
    - max. 3 záznamy

Prevod medzi menami a adresami

- súbor `/etc/nsswitch.conf`
  - určuje, z akých zdrojov sa berú informácie pre mapovanie medzi menami a číslami – aj medzi menami počítačov a IP adresami
  - `hosts: files dns`
    - najprv `/etc/hosts`, potom DNS

Sieťové služby

- v súbore `/etc/services` je priradenie názvov služieb číslam portov a protokolov
  - meno číslo portu/protokol alias
- mená protokolov sa na ich čísla mapujú pomocou súboru `/etc/protocols`
- programy poskytujúce sieťové služby (démony, servery) buď
  - počúvajú na príslušnom porte – čakajú na požiadavky, alebo
  - ich spúšťa "superserver" inetd

Sieťové služby – inetd

- **inetd** sa riadi súborom `/etc/inetd.conf`
  - medzerami oddelené položky
  - meno typ protokol wait/nowait user prog args
    - meno – názov podľa /etc/services
    - typ – **dgram** alebo **stream**
    - protokol – **udp** alebo **tcp**
    - wait = inetd počká na skončenie procesu, kým bude spracovávať ďalšie požiadavky na danom porte
    - nowait = inetd pri ďalšej požiadavke na porte spustí ďalšiu inštanciu programu
    - user – používateľ, s ktorého UID sa má program spustiť

Sieťové služby – inetd

  - prog – cesta k súboru, ktorý sa má spustiť
  - args – argumenty (vrátane nultého – meno programu)
- na spúšťanie serverov z inetd sa často využíva program tcpd, ktorý robí kontrolu prístupu k službám
  - riadi sa súbormi `/etc/hosts.allow` a `/etc/hosts.deny`
  - najprv sa konzultuje hosts.allow, ak sa nenájde zhoda, tak hosts.deny, ak sa nenájde zhoda ani tam, spojenie sa povolí

Sieťové služby – inetd

- štruktúra `/etc/hosts.{allow,deny}`
  - zoznam služieb : zoznam klientov
  - služba:
    - meno[@počítač]
    - ALL
  - klient:
    - ALL
    - počítač
    - KNOWN (známe meno)
    - UNKNOWN (neznáme meno)
    - PARANOID (meno nekorešponduje s adresou)

Sieťové služby – inetd

- počítač:
  - IPv4 adresa 10.0.0.1
  - IPv4 adresa/maska 10.0.0.0/255.255.255.0
  - IPv4 adresa/dĺžka 10.0.0.0/24
  - [IPv6 adresa] [fdaa:1::1]
  - [IPv6 adresa]/dĺžka [fdaa:1::]/64
  - 10.0.
  - meno abc.xyz.sk
  - .xyz.sk
- zoznam
  - pol1, pol2 EXCEPT pol3

Sieťové služby – inetd

```
/etc/hosts.allow:
hello: 10.0.0.0/24, 127.0.0.1, [::1], [fdaa:1::]/64, aaa.xy.sk


/etc/hosts.deny:
hello: ALL
ALL: PARANOID
```

Sieťové služby

- príkaz `netstat`
  - výpis sieťových spojení a otvorených portov, routovacej tabuľky, štatistík, ...
  - najbežnejšie parametre
    - `-n` – IP adresy a čísla portov vypisuje číselne (inak sa snaží konvertovať na meno)
    - `-a` – výpis prebiehajúcich spojení aj otvorených portov čakajúcich na spojenie alebo datagram (inak len prebiehajúce)
    - `-r` – výpis routovacej tabuľky
    - `-p` – výpíše aj číslo príslušného procesu

Konfigurácia VLAN

- umožňuje príjem / vysielanie Ethernet-ových rámcov s VLAN tag-om (IEEE 802.1Q)
- cez jedno fyzické sieťové rozhranie je možné pripojiť viacero VLAN
- každá VLAN je reprezentovaná samostatným virtuálnym sieťovým rozhraním s vlastnou konfiguráciou

Konfigurácia VLAN

- `vconfig add` *`rozhranie vlan_id`*
  - vytvorí nové vlan rozhranie pre VLAN s číslom vlan_id
- `vconfig rem` *`vlan_rozhranie`*
  - odstráni VLAN rozhranie

Konfigurácia VLAN

- `vconfig set_name_type` *typ*
  - nastaví typ pre tvorbu mena VLAN rozhrania
    - VLAN_PLUS_VID vlan0002
    - VLAN_PLUS_VID_NO_PAD vlan2
    - DEV_PLUS_VID eth0.0002
    - DEV_PLUS_VID_NO_PAD eth0.2

Konfigurácia VLAN

```
# vconfig add eth0 10
Added VLAN with VID == 10 to IF -:eth0:-
# ifconfig eth0.10
eth0.10   Link encap:Ethernet  HWaddr 18:03:73:c1:16:89
          BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

# vconfig rem eth0.10
Removed VLAN -:eth0.10:-
```

Konfigurácia VLAN

- v Debian / Ubuntu
  - rozpozná meno VLAN rozhrania v /etc/network/interfaces
  - automaticky vytvorí príslušné VLAN rozhranie
  - pri menách typu vlan... potrebuje informáciu o fyzickom rozhraní
    - `vlan-raw-device eth0`

Bridge

- umožňuje vytvoriť softvérový „switch“
- často sa využíva napr. pri virtualizácii
  - na pripojenie virtuálnych počítačov k fyzickej sieti
- podporuje aj STP
  - je možné využiť aj na pripojenie do redundantnej sieťovej infraštruktúry pomocou viacnásobných fyzických liniek

Bridge

- `brctl show`
  - ukáže konfiguráciu bridge-ov
- `brctl showmacs`
  - vypíše tabuľku L2 adries na jednotlivých portoch
- `brctl addbr` *`meno`*
  - vytvorí nový bridge
- `brctl delbr` *`meno`*
  - zruší bridge

Bridge

- `brctl addif` *`meno_br meno_rozhr`*
  - pridá rozhranie ako port do bridge-u
- `brctl delif` *`meno_br meno_rozhr`*
  - odstráni port bridge-u
- `brctl stp` *`meno_br`* `on | off`
  - zapne / vypne STP
- `brctl showstp` *`meno_br`*
  - vypíše parametre a stav STP

Bridge

```
# ifconfig eth0.10 up
# ifconfig eth1 up
# brctl addbr br0
# brctl addif br0 eth0.10
# brctl addif br0 eth1
# brctl stp br0 on
# brctl show
bridge name  bridge id           STP enabled  interfaces
br0          8000.180373c11689   yes          eth0.10
                                              eth1
# ifconfig br0 10.0.0.5 netmask 255.255.255.0 broadcast 10.0.0.255


# brctl delif br0 eth1
# brctl delif br0 eth0.10
# brctl delbr br0
```

Bridge v Debian / Ubuntu

/etc/network/interfaces

```
auto br0
iface br0 inet static
        address 192.168.1.2
        network 192.168.1.0
        netmask 255.255.255.0
        broadcast 192.168.1.255
        bridge_ports eth0 eth1
```

Netfilter – Linux firewall

- úlohy firewall-u na koncovom počítači
  - filtrovať prichádzajúcu komunikáciu
  - filtrovať odchádzajúcu komunikáciu
- úlohy firewall-u na router-i
  - filtrovať prechádzajúcu komunikáciu
  - NAT (preklad adries)

Netfilter – Linux firewall

- stateless firewall
  - každý packet posudzuje samostatne
    - na základe údajov v hlavičkách
- stateful firewall
  - udržiava prehľad o „spojeniach“
  - packet posudzuje v súvislosti s predchádzajúcimi
  - umožňuje implementovať komplexnejšiu politiku
  - je náročnejší na pamäť a procesorový čas
    - je náchylný na DoS

Netfilter – Linux firewall

- súčasť jadra (kernelu) systému
  - subsystém **netfilter**
- stateful firewall
  - implementuje connection tracking
    - TCP, UDP
    - pomocné moduly pre niektoré „problematické“ aplikačné protokoly (napr. FTP)
- aj stateless firewall
  - bez využitia connection tracking-u

Netfilter – Linux firewall

- Connection tracking – klasifikácia packetu
  - NEW
    - packet je prvým packetom „spojenia“
  - ESTABLISHED
    - packet je súčasťou už existujúceho „spojenia“
  - RELATED
    - packet je prvým packetom očakávaného „spojenia“
  - UNTRACKED
    - úmyselne nesledovaný
  - INVALID

Netfilter – Linux firewall

- Connection tracking – tabuľka spojení
  - IP adresa zdroja a cieľa
  - L4 protokol
  - bližšie informácie z L4
    - pre TCP a UDP čísla portov
  - timeout
  - stav

Netfilter – Linux firewall

- Connection tracking
  - ak sa údaje z packetu v tabuľke nenachádzajú, vytvorí sa záznam o novom „spojení“
  - ak sa nachádzajú, packet je súčasťou existujúceho alebo očakávaného „spojenia“, aktualizuje sa timeout a stav
  - po vypršaní timeout-u sa záznam zruší

Netfilter – Linux firewall

- Connection tracking – default timeout
  - TCP
    - od 10s do 5 dní
    - podľa stavu TCP spojenia
  - UDP
    - 30s (počiatočný)
    - 180s (ak ide o „dlhšiu“ komunikáciu)
  - ICMP: 30s
  - generický: 10m

Netfilter – Linux firewall

/proc/sys/net/netfilter/

```
nf_conntrack_generic_timeout: 600
nf_conntrack_icmp_timeout: 30
nf_conntrack_tcp_timeout_close: 10
nf_conntrack_tcp_timeout_close_wait: 60
nf_conntrack_tcp_timeout_established: 432000
nf_conntrack_tcp_timeout_fin_wait: 120
nf_conntrack_tcp_timeout_last_ack: 30
nf_conntrack_tcp_timeout_max_retrans: 300
nf_conntrack_tcp_timeout_syn_recv: 60
nf_conntrack_tcp_timeout_syn_sent: 120
nf_conntrack_tcp_timeout_time_wait: 120
nf_conntrack_tcp_timeout_unacknowledged: 300
nf_conntrack_udp_timeout: 30
nf_conntrack_udp_timeout_stream: 180
```

Netfilter – Linux firewall

```
# conntrack -L
icmp     1 26 src=192.168.9.11 dst=1.1.1.1 type=8 code=0 id=6689
[UNREPLIED] src=1.1.1.1 dst=192.168.9.11 type=0 code=0 id=6689 mark=0
tcp      6 431715 ESTABLISHED src=192.168.9.11 dst=158.195.87.39
sport=59189 dport=22 src=158.195.87.39 dst=192.168.9.11 sport=22
dport=59189 [ASSURED] mark=0
udp      17 23 src=192.168.9.11 dst=1.1.1.1 sport=36040 dport=3333
[UNREPLIED] src=1.1.1.1 dst=192.168.9.11 sport=3333 dport=36040 mark=0
conntrack v0.9.14 (conntrack-tools): 3 flow entries have been shown.

# cat /proc/net/ip_conntrack
icmp     1 23 src=192.168.9.11 dst=1.1.1.1 type=8 code=0 id=6689
[UNREPLIED] src=1.1.1.1 dst=192.168.9.11 type=0 code=0 id=6689 mark=0
tcp      6 431712 ESTABLISHED src=192.168.9.11 dst=158.195.87.39
sport=59189 dport=22 src=158.195.87.39 dst=192.168.9.11 sport=22
dport=59189 [ASSURED] mark=0
udp      17 21 src=192.168.9.11 dst=1.1.1.1 sport=36040 dport=3333
[UNREPLIED] src=1.1.1.1 dst=192.168.9.11 sport=3333 dport=36040 mark=0
```

Netfilter – Linux firewall

- Počas spracovania packetu sú na viacerých miestach volané funkcie subsystému netfilter, ktoré môžu
  - zastaviť spracovanie packetu
  - povoliť pokračovanie spracovania packetu
  - zmeniť niektoré atribúty packetu

Netfilter – Linux firewall

- činnosť riadená pravidlami v niekoľkých tabuľkách
  - filter – filtrovanie packetov
  - nat – pravidlá pre NAT
  - mangle – úprava packetov, značkovanie
  - raw – vylúčenie z connection tracking-u
  - security – bezpečnostné značkovanie (SELinux)

Netfilter – Linux firewall

- každá tabuľka obsahuje *reťaze (chains)*
  - reťaz obsahuje postupnosť pravidiel
  - reťaze je možné pridávať
    - členenie, sprehľadnenie pravidiel
    - dosiahnutie komplikovanejšieho súboru pravidiel
  - fixné reťaze majú nastaviteľnú default akciu
    - pre prípad, že akcia nie je zmenená niektorým pravidlom

Netfilter – Linux firewall

- filter
  - INPUT
    - packety určené pre tento počítač – prichádzajpce
  - OUTPUT
    - packety vytvorené týmto počítačom – odchádzajúce
  - FORWARD
    - packety prechádzajúce týmto počítačom – smerované

Netfilter – Linux firewall

- nat
  - PREROUTING
    - prichádzajúce packety pred smerovaním
    - umožňuje meniť cieľ (DNAT)
  - OUTPUT
    - vytvorené packety pred smerovaním
  - POSTROUTING
    - odchádzajúce packety po smerovaní
    - umožňuje meniť zdroj (SNAT)

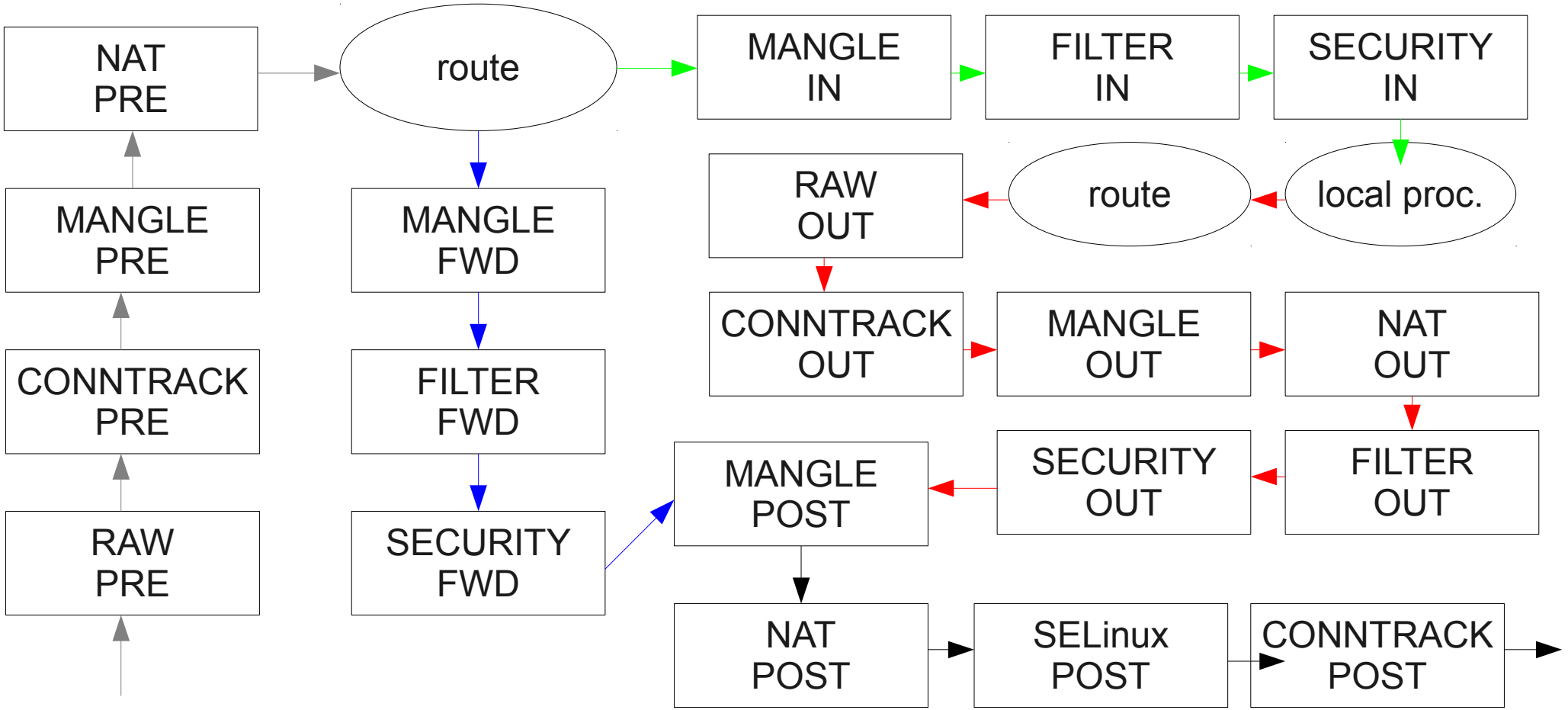
Netfilter – Linux firewall

- mangle
  - PREROUTING
  - OUTPUT
  - INPUT
  - FORWARD
  - POSTROUTING

Netfilter – Linux firewall

- raw
  - PREROUTING
  - OUTPUT
- security
  - INPUT
  - OUTPUT
  - FORWARD

Netfilter – Linux firewall



Netfilter – Linux firewall

- Pravidlo
  - test
    - môže pozostávať z viacerých podmienok, ak sú všetky splnené, pravidlo sa aplikuje
  - cieľ (akcia)
    - ACCEPT – pokračovať v spracovaní
    - DROP – zahodiť packet
    - meno reťaze – pokračovať pravidlami inej reťaze
    - RETURN – pokračovať v predošlej reťazi

Netfilter – Linux firewall

- prechádzanie reťaze
  - zisti, či test pravidla sedí
    - ak nie, pokračuj ďalším pravidlom
    - ak áno, použi cieľ
  - na konci fixnej reťaze
    - použi default akciu
  - na konci inej reťaze
    - pokračuj ďalším pravidlom do predošlej reťaze

Netfilter – Linux firewall

- `iptables [-t tab] príkaz`
  - tab – tabuľka (default: filter)
  - príkaz
    - `-A reťaz pravidlo` – pridá na koniec reťaze
    - `-D reťaz pravidlo` – vymaže pravidlo
    - `-D reťaz poradie` – vymaže pravidlo
    - `-I reťaz [poradie] pravidlo` – vloží pravidlo (def. 1)
    - `-R reťaz poradie pravidlo` – nahradí pravidlo novým

Netfilter – Linux firewall

- `-L [reťaz] [voľby]` – vypíše pravidlá
- `-F [reťaz]` – vymaže všetky pravidlá
- `-Z [reťaz [poradie]]` – vynuluje počítadlá pre pravidlo
- `-N reťaz` – vytvorí novú reťaz
- `-X reťaz` – zruší reťaz
- `-P reťaz akcia` – nastaví default akciu reťazi (policy)

– voľby

- `-n` – neprevádzať čísla (adresy, porty) na mená
- `-v` – zobraziť detaily
- `-x` – zobraziť presné hodnoty
- `--line-numbers` – zobraziť čísla riadkov (pravidiel)

Netfilter – Linux firewall

- pravidlo: *testy -j cieľ*
- test
  - [!] -p *protokol* – L4 protokol (napr. tcp, udp, icmp)
  - [!] -s *adresa/maska* – zdrojová adresa packetu
  - [!] -d *adresa/maska* – cieľová adresa packetu
  - [!] -i *rozhranie*[+] - vstupné sieťové rozhranie
  - [!] -o *rozhranie*[+] - výstupné sieťové rozhranie
  - [!] -f – iný ako prvý fragment packetu
  - -m *modul parametre* – rozširujúci modul s testom

Netfilter – Linux firewall

- modul tcp
  - `[!] --sport port[:port]` – zdrojový port
  - `[!] --dport port[:port]` – cieľový port
  - `[!] --tcp-flags mask val` – TCP príznaky
    - (SYN, ACK, FIN, RST, URG, PSH, ALL, NONE)
  - `[!] --syn` – nastavený SYN príznak (bez ACK, FIN, RST) – prvý packet TCP spojenia

Netfilter – Linux firewall

- modul udp
  - `[!] --sport port[:port]` – zdrojový port
  - `[!] --dport port[:port]` – cieľový port
- modul icmp
  - `[!] --icmp-type type/code`
  - `[!] --icmp-type typename`

Netfilter – Linux firewall

- modul multiport
  - umožňuje uviesť viac portov a rozsahov portov
  - pre tcp a udp
  - `[!] --sports` *`port,port,port:port,...`* - zdrojové porty
  - `[!] --dports` *`port,port,port:port,...`* - cieľové porty
  - `[!] --ports` *`port,port,port:port,...`* - nejaké porty

Netfilter – Linux firewall

- modul state
  - `[!] --state` *`stav`* – connection tracking klasifikácia
    - NEW
    - ESTABLISHED
    - RELATED
    - UNTRACKED
    - INVALID
    - môže ich byť aj viac oddelených čiarkou

Netfilter – Linux firewall

- modul conntrack
  - `[!] --ctstate` *`stav`* – connection tracking
  - novšia náhrada modulu state
  - umožňuje aj ďalšie jemnejšie kontrolovanie stavu

Netfilter – Linux firewall

- NAT
  - source-NAT (SNAT)
    - prepisovanie zdrojovej IP adresy a portu
    - umožňuje komunikovať zo siete so súkromnými adresami do Internet-u
  - destination-NAT (DNAT)
    - prepisovanie cieľovej IP adresy a portu
    - umožňuje prístup z Internet-u na služby vo vnútornej sieti so súkromnými adresami

Netfilter – Linux firewall

- SNAT
  - prvý packet spojenia (smerom von)
    - použije sa reťaz POSTROUTING v tabuľke nat na nájdenie pravidla pre prepis zdrojovej adresy
    - prepíše sa zdrojová adresa a port
    - údaje sa uložia do conntrack tabuľky
  - ďalší packet smerom von
    - zdrojová adresa sa prepíše na hodnoty z conntrack tabuľky
  - packet smerom dnu
    - cieľová adresa sa prepíše inverzne podľa conntrack tabuľky

Netfilter – Linux firewall

- DNAT
  - prvý packet spojenia (smerom dnu)
    - použije sa reťaz PREROUTING alebo OUTPUT tabuľky nat na nájdenie pravidla pre prepis cieľovej adresy
    - prepíše sa cieľová adresa a port
    - údaje sa uložia do conntrack tabuľky
  - ďalší packet spojenia smerom dnu
    - cieľová adresa a port sa prepíšu podľa conntrack tabuľky
  - packet smerom von
    - zdrojová adresa a port sa prepíšu podľa conntrack tabuľky

Netfilter – Linux firewall

- cieľ SNAT
  - `--to-source` *`ipaddr`*`[-`*`ipaddr`*`][:`*`port`*`[-`*`port`*`]]`
    - rozsah IP adries a prípadne portov, na ktoré sa môže prepísať
- cieľ MASQUERADE
  - `--to-ports` *`port`*`[-`*`port`*`]`
    - ako SNAT na vlastnú IP adresu + vymazanie spojení pri down
- cieľ DNAT
  - `--to-destination` *`ipaddr`*`[-`*`ipaddr`*`][:`*`port`*`[-`*`port`*`]]`
    - rozsah IP adries a prípadne portov, na ktoré sa môže prepísať

Netfilter – Linux firewall

- problémy s niektorými aplikačnými protokolmi
  - ak používajú IP adresy a/alebo porty na aplikačnej vrstve
    - napr. FTP
  - potrebujú pomocný modul pre netfilter
    - analýza L7 protokolu a generovanie očakávaných spojení
    - modifikácia údajov L7 protokolu podľa NAT informácií v conntrack tabuľke
  - alebo musia na aplikačnej vrstve detegovať NAT a prospôsobiť sa (napr. SIP)

Netfilter – Linux firewall

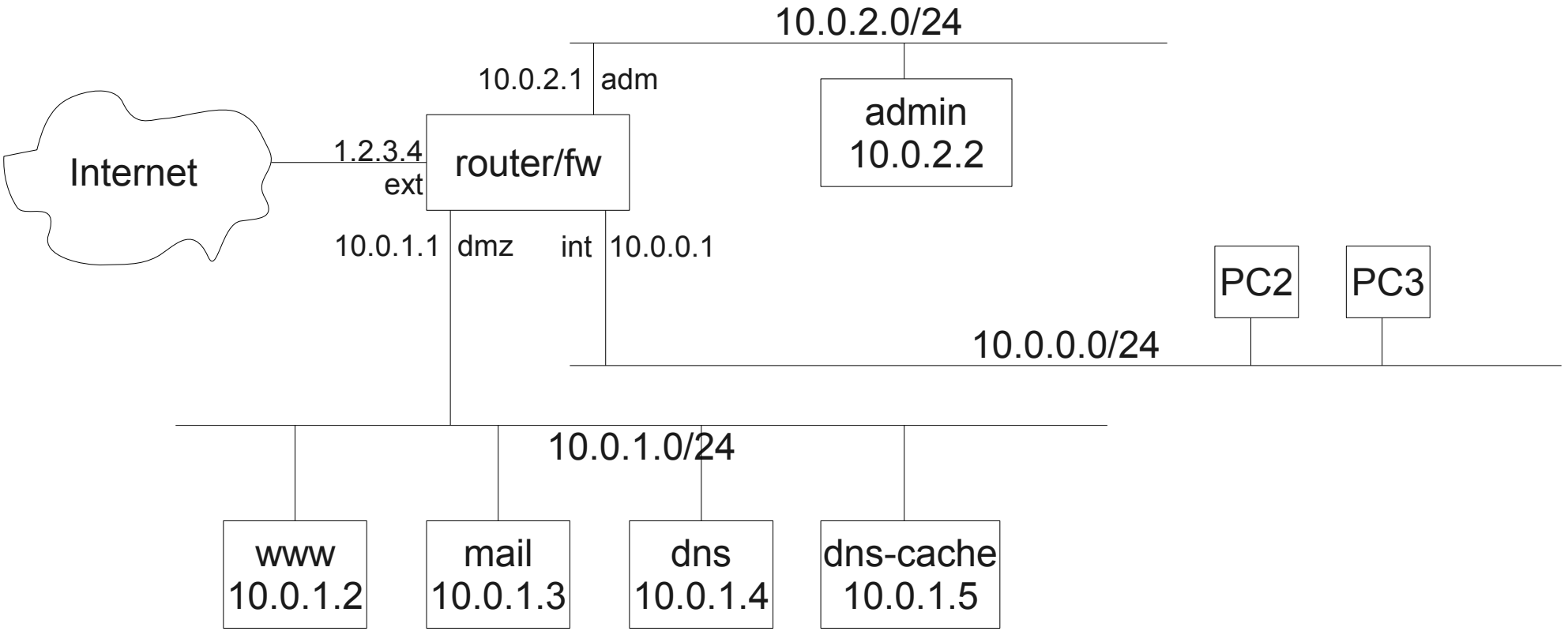
- cieľ LOG
  - odošle informáciu o packete do log-u (a pokračuje ďalej)
  - `--log-level` *`level`* – priorita logu
  - `--log-prefix` *`prefix`* – textový prefix správy
  - `--log-uid` – do správy pridá aj uid vlastníka socket-u, z ktorého bol packet odoslaný
  - `--log-tcp-sequence`
  - `--log-tcp-options`
  - `--log-ip-options`

Netfilter – Linux firewall

Ochrana koncového počítača

```
iptables -P INPUT DROP
iptables -P OUTPUT ACCEPT
iptables -P FORWARD DROP
iptables -F

iptables -A INPUT -i lo -j ACCEPT
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

iptables -A INPUT -p icmp --icmp-type echo-request -j ACCEPT
iptables -A INPUT -p tcp --dport 22 -s 192.168.0.0/24 -j ACCEPT

iptables -A OUTPUT -o lo -j ACCEPT
iptables -A OUTPUT -m state –state ESTABLISHED,RELATED -j ACCEPT
iptables -A OUTPUT -p tcp -m multiport --dports 80,443,53 -j ACCEPT
iptables -A OUTPUT -p udp --dport 53 -d 192.168.0.1 -j ACCEPT
iptables -A OUTPUT -j LOG --log-prefix "podozrivy packet: "
```

Netfilter – Linux firewall



Netfilter – Linux firewall

```
iptables -F
iptables -X
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP

iptables -A INPUT -i lo -j ACCEPT
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -A INPUT -i adm -p tcp --dport 22 -s 10.0.2.0/24 -j ACCEPT
iptables -A OUTPUT -o lo -j ACCEPT
iptables -A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -A OUTPUT -p udp -d 10.0.1.5 --dport 53 -j ACCEPT

iptables -A FORWARD -s 10.0.0.0/24 ! -i int -j DROP
iptables -A FORWARD -s 10.0.1.0/24 ! -i dmz -j DROP
iptables -A FORWARD -s 10.0.2.0/24 ! -i adm -j DROP
iptables -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT
```

Netfilter – Linux firewall

```
iptables -N www
iptables -N mail
iptables -N dns
iptables -N dnscache
iptables -N acclocal
iptables -N pcout
iptables -N dmzout
iptables -N admout

iptables -A FORWARD -d 10.0.1.2 -j www
iptables -A FORWARD -d 10.0.1.3 -j mail
iptables -A FORWARD -d 10.0.1.4 -j dns
iptables -A FORWARD -d 10.0.1.5 -j dnscache
iptables -A FORWARD -s 10.0.2.0/24 -d 10.0.1.0/24 -j ACCEPT

iptables -A FORWARD -s 10.0.0.0/24 -o ext -j pcout
iptables -A FORWARD -s 10.0.1.0/24 -o ext -j dmzout
iptables -A FORWARD -s 10.0.2.0/24 -o ext -j admout
```

Netfilter – Linux firewall

```
iptables -A www -p tcp -m multiport --dports 80,443 -j ACCEPT

iptables -A mail -p tcp -m multiport --dports 25,110,143,995,993 -j
ACCEPT

iptables -A dns -p udp --dport 53 -j ACCEPT
iptables -A dns -p tcp --dport 53 -j ACCEPT

iptables -A dnscache -p udp --dport 53 -j acclocal
iptables -A dnscache -p tcp --dport 53 -j acclocal

iptables -A acclocal -s 10.0.0.0/24 -j ACCEPT
iptables -A acclocal -s 10.0.2.0/24 -j ACCEPT
```

Netfilter – Linux firewall

```
iptables -N commonout
iptables -A commonout -p tcp --dport 80 -j ACCEPT
# lepšie by bolo špecifikovať presnejšie adresy, odkiaľ sa budú
# preberať aktualizácie OS, aplikácií, ...

iptables -A dmzout -j commonout
iptables -A dmzout -s 10.0.1.5 -p udp --dport 53 -j ACCEPT
iptables -A dmzout -s 10.0.1.5 -p tcp --dport 53 -j ACCEPT
iptables -A dmzout -s 10.0.1.3 -p tcp --dport 25 -j ACCEPT

iptables -A admout -j commonout
iptables -A admout -p udp --dport 53 -j ACCEPT
iptables -A admout -p tcp --dport 53 -j ACCEPT

iptables -A pcout -j commonout
iptables -A pcout -p tcp -m multiport --dports 80,443 -j ACCEPT
```

Netfilter – Linux firewall

```
iptables -t nat -F
iptables -t nat -A POSTROUTING -s 10.0.0.0/24 -o ext -j SNAT
 --to-source 1.2.3.4
iptables -t nat -A POSTROUTING -s 10.0.1.0/24 -o ext -j SNAT
 --to-source 1.2.3.4
iptables -t nat -A POSTROUTING -s 10.0.2.0/24 -o ext -j SNAT
 --to-source 1.2.3.4

iptables -t nat -A PREROUTING -i ext -p tcp -m multiport --dports
80,443 -j DNAT --to-destination 10.0.1.2

iptables -t nat -A PREROUTING -i ext -p tcp -m multiport --dports
25,110,143,993,995 -j DNAT --to-destination 10.0.1.3

iptables -t nat -A PREROUTING -i ext -p udp --dport 53 -j DNAT
 --to-destination 10.0.1.4
iptables -t nat -A PREROUTING -i ext -p tcp --dport 53 -j DNAT
 --to-destination 10.0.1.4
```

Netfilter – Linux firewall

- modul length
  - `[!] --length` *`dĺžka`*`[:`*`dĺžka`*`]` – ak je dĺžka packetu v uvedenom rozsahu
- modul mac
  - `[!] --mac-source` *`L2adresa`* – zdrojová L2 (MAC) adresa
- modul ttl – testuje TTL packetu
  - `[!] --ttl-eq` *`ttl`*
  - `--ttl-gt` *`ttl`*
  - `--ttl-lt` *`ttl`*

Netfilter – Linux firewall

- modul hashlimit
  - umožňuje obmedziť počet packetov za jednotku času za skupinu zdrojových/cieľových adries/portov
  - `--hashlimit-name` *meno* – definuje meno pre limit
  - `--hashlimit-upto` *počet*`/second|minute|hour|day`
  - `--hashlimit-above` *počet*`/second|minute|hour|day`
  - `--hashlimit-burst` *počet* – max. počiatočný počet
  - `--hashlimit-mode {srcip|srcport|dstip|dstport},...`
    - podľa čoho sa budú tvoriť skupiny
  - `--hashlimit-srcmask` *dĺžka*
  - `--hashlimit-dstmask` *dĺžka*
    - aká veľká časť adresy sa zohľadní

Netfilter – Linux firewall

```
# iptables -A INPUT -p icmp --icmp-type echo-request -m hashlimit
--hashlimit-name icmp --hashlimit-above 30/minute --hashlimit-burst
3 --hashlimit-mode srcip -j DROP

$ ping 158.195.87.39
PING 158.195.87.39 (158.195.87.39) 56(84) bytes of data.
64 bytes from 158.195.87.39: icmp_req=1 ttl=62 time=13.1 ms
64 bytes from 158.195.87.39: icmp_req=2 ttl=62 time=14.3 ms
64 bytes from 158.195.87.39: icmp_req=3 ttl=62 time=14.8 ms
64 bytes from 158.195.87.39: icmp_req=4 ttl=62 time=13.9 ms
64 bytes from 158.195.87.39: icmp_req=5 ttl=62 time=17.9 ms
64 bytes from 158.195.87.39: icmp_req=7 ttl=62 time=13.8 ms
64 bytes from 158.195.87.39: icmp_req=9 ttl=62 time=12.1 ms
64 bytes from 158.195.87.39: icmp_req=11 ttl=62 time=12.3 ms
64 bytes from 158.195.87.39: icmp_req=13 ttl=62 time=14.7 ms
64 bytes from 158.195.87.39: icmp_req=15 ttl=62 time=15.1 ms

# cat /proc/net/ipt_hashlimit/icmp
59 158.195.87.244:0->0.0.0.0:0 23424 192000 64000
```

Netfilter – Linux firewall

- modul recent
  - umožňuje vytvárať zoznamy IP adries a následne voči nim iné IP adresy porovnávať
    - napr. na detekciu skenovania siete a odfiltrovanie útočníka
      - ak pošle packet na honeypot, tak mu zahodíme všetky packety na reálne služby
    - na port-knocking
      - pred povolením prístupu napr. na ssh musí administrátor poslať niečo na iný port/adresu

Netfilter – Linux firewall

- modul recent
  - `--name meno` – meno pre zoznam
  - `--set` – pridá zdrojovú IP adresu do zoznamu
  - `--rdest` – namiesto zdrojovej použije cieľovú adr.
  - `[!] --rcheck` – overí, či je adresa v zozname
  - `[!] --update` – ako rcheck a aktualizuje zoznam
  - `[!] --remove` – ako rcheck a odstráni zo zoznamu
  - `--seconds sec` – obmedzí sa na posledných sec sekúnd
  - `--hitcount hits` – overí, či je z adresy zaznamenaných aspoň hits packetov
  - `--rttl` – overí, či TTL packetu zodpovedá TTL prvého zaznamenaného
  - `--reap` – vymaže záznamy o packetoch staršie ako čas v --seconds

Netfilter – Linux firewall

```
# iptables -A INPUT -p udp --dport 3333 -m recent --name knock3333
--set
# iptables -A INPUT -p tcp --dport 22 -m recent --name knock3333
--rcheck --seconds 30 -j ACCEPT


# cat /proc/net/xt_recent/knock3333
src=158.195.87.244 ttl: 62 last_seen: 89813806 oldest_pkt: 4
89671926, 89679310, 89810567, 89813806
```

Netfilter – Linux firewall

- cieľ NOTRACK
  - zabráni vytvoreniu záznamu v conntrack tabuľke a klasifikácii packetu
  - len v raw tabuľke
- cieľ REDIRECT
  - prepíše cieľovú adresu na 127.0.0.1
  - `--to-ports` *`port`*`[-`*`port`*`]` – prepíše cieľový port na niektorý z uvedeného rozsahu (inak nemení)

Netfilter – Linux firewall

- cieľ REJECT
  - zahodí packet (ako DROP), ale pošle naspäť ICMP
  - `--reject-with` *typ*
    - icmp-net-unreachable
    - icmp-host-unreachable
    - icmp-port-unreachable (default)
    - icmp-proto-unreachable
    - icmp-net-prohibited
    - icmp-host-prohibited
    - icmp-admin-prohibited

Netfilter – Linux firewall

- cieľ MARK
  - umožňuje označovať packety 32-bit číslom
  - `--set-mark` *`hodn`*`[/`*`maska`*`]` – x=(x&~m)|h
  - `--set-xmark` *`hodn`*`[/`*`maska`*`]` – x=(x&~m)^h
  - `--and-mark` *`hodn`*`, --or-mark` *`hodn`*`, --xor-mark` *`hodn`*
- modul mark
  - `[!] --mark hodn[/maska]` – zistí, či značka packetu zodpovedá ((x & mask) == hodn)

Netfilter – Linux firewall

- cieľ TTL
  - modifikuje TTL packetu
  - `--ttl-set` *`hodnota`*
  - `--ttl-dec` *`hodnota`*
  - `--ttl-inc` *`hodnota`*
- cieľ TEE
  - vytvorí a pošle kópiu packetu
  - `--gateway` *`IPadresa`* – kam poslať – musí byť „sused“

Netfilter – Linux firewall

- cieľ TCPMSS
  - prípadne zníži hodnotu MSS v TCP SYN packete
  - `--set-mss hodnota` – zníži MSS ak je viac
  - `--clamp-mss-to-pmtu` – automaticky určí hranicu ako PMTU – 40B

Netfilter – Linux firewall

- cieľ SECMARK
  - umožňuje packetu priradiť SELinux kontext
  - v tabuľke security
  - potom je možné v SELinux politike určovať práva na prijatie/odoslanie packetu procesom
  - `--selctx` *`kontext`*

Netfilter – Linux firewall

```
admin@debian:~$ ssh 192.168.122.1
ssh: connect to host 192.168.122.1 port 22: Connection refused


root@debian:~# iptables -t security -A INPUT -p tcp --sport 22 -j
SECMARK --selctx system_u:object_r:ssh_client_packet_t:s0
root@debian:~# iptables -t security -A OUTPUT -p tcp --dport 22 -j
SECMARK --selctx system_u:object_r:ssh_client_packet_t:s0


admin@debian:~$ ssh 192.168.122.1 -l jerry
jerry@192.168.122.1's password:
Linux jerry 3.2.0-4-686-pae #1 SMP Debian 3.2.63-2+deb7u1 i686
...
```

Netfilter – Linux firewall

- ip6tables
  - ako iptables, ale pre IPv6
  - tabuľky raw, mangle, filter, security
    - nemá nat (zatiaľ)
  - rovnaké príkazy
  - rovnaké (+-) testy a ciele
    - adresy sú IPv6

Netfilter – Linux firewall

- `iptables-save [-t table]`
  - vypíše celú tabuľku (alebo všetky) na výstup
- `iptables-restore [-T table]`
  - prečíta tabuľku/y zo vstupu a vloží do kernelu
- `ip6tables-save [-t table]`
  - vypíše celú tabuľku (alebo všetky) na výstup
- `ip6tables-restore [-T table]`
  - prečíta tabuľku/y zo vstupu a vloží do kernelu

Netfilter & bridge

- aj pakety preposlané bridge-om (IPv4, IPv6) sú spracované netfilter-om
  - umožňuje kontrolovať aj tok medzi jednotlivými portami bridge-u (filter/FORWARD)
  - modul physdev
    - `[!] --physdev-in rozhranie`
    - `[!] --physdev-out rozhranie`

Policy routing

- štandardne sa smerovanie robí len na základe cieľovej adresy
- policy routing umožňuje mať viac smerovaích tabuliek a zvoliť jednu z nich na základe niektorých parametrov
  - najrobustnejšie podľa značky packetu, ktorá sa nastavuje pomocou cieľa MARK

Policy routing

- routovacie tabuľky
  - local – lokálne adresy, broadcasty
  - main – hlavná routovacia tabuľka
  - default – prázdna
  - cache – obsahuje aktuálne používané záznamy pre konkrétne ciele, vytvára sa automaticky
  - číslo – vlastná (1-252)
- `ip route ... table` *`tabuľka`*

Policy routing

- pravidlá
  - priorita – určuje poradie, v akom sa aplikujú
  - selektor – určuje podmienku, za ktorej sa aplikuje
    - `fwmark` *`hodnota`*
  - akcia – zvolí smerovaciu tabuľku
    - `table main`
    - `table 100`

Policy routing

- `ip rule add|del|show selektor akcia`

```
# ip rule show
0:  from all lookup local
32766: from all lookup main
32767: from all lookup default
```

pridanie pravidla s priotitou 100: ak značka = 100, použi tabuľku 100

```
# ip rule add pref 100 fwmark 100 table 100
```

```
# ip rule show
0:  from all lookup local
100:   from all fwmark 0x64 lookup 100
32766: from all lookup main
32767: from all lookup default
```

odstránenie pravidla s prioritou 100

```
# ip rule del pref 100
```

Policy routing

```
# ip rule add pref 100 fwmark 100 table 100
# iptables -t mangle -A PREROUTING -s 192.168.0.0/27 -j MARK
          -set-mark 100
# ip route add default via 10.0.0.10 table 100
# ip route flush cache
```

IPv6 tunelovanie

- využitie 6to4 na získanie prístupu do IPv6
  - k IPv4 adrese aa.bb.cc.dd je pridelená IPv6 sieť s adresou 2002:aabb:ccdd::/48
  - IPv6 packety sa tunelujú cez IPv4 Internet
    - do iných sietí typu 6to4 prostredníctvom gateway-a s príslušnou IPv4 adresou
    - do IPv6 Internetu cez 192.88.99.1 (anycast)
  - v Linuxe pomocou tunelu typu `sit`

IPv6 tunelovanie

```
# ip tunnel add tun6to4 mode sit local 158.195.87.39 remote any
# ip link set tun6to4 up
# ip addr add 2002:9ec3:5727::1/16 dev tun6to4
# ip route add 2000::/3 via ::192.88.99.1 dev tun6to4
# ip -6 route show
::/96 via :: dev tun6to4  metric 256
2002::/16 dev tun6to4  proto kernel  metric 256
2000::/3 via ::192.88.99.1 dev tun6to4  metric 1024
fe80::/64 dev eth0  proto kernel  metric 256
fe80::/64 dev vnet0  proto kernel  metric 256
fe80::/64 dev dummy0  proto kernel  metric 256
fe80::/64 dev tun6to4  proto kernel  metric 256

# ping6 -n www.google.sk
PING www.google.sk(2a00:1450:4001:807::101f) 56 data bytes
64 bytes from 2a00:1450:4001:807::101f: icmp_seq=1 ttl=59 time=25.6 ms
64 bytes from 2a00:1450:4001:807::101f: icmp_seq=2 ttl=59 time=42.0 ms
```

IPv6 tunelovanie

```
# tcpdump -nlp -i eth0
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
21:58:00.035953 IP 158.195.87.39.48184 > 158.195.18.163.53: 31764+ AAAA?
www.google.sk. (31)

21:58:00.036826 IP 158.195.18.163.53 > 158.195.87.39.48184: 31764 1/4/4 AAAA
2a00:1450:4001:807::101f (205)

21:58:00.037099 IP 158.195.87.39 > 192.88.99.1: IP6 2002:9ec3:5727::1 >
2a00:1450:4001:807::101f: ICMP6, echo request, seq 1, length 64

21:58:00.059438 IP 192.88.99.1 > 158.195.87.39: IP6 2a00:1450:4001:807::101f >
2002:9ec3:5727::1: ICMP6, echo reply, seq 1, length 64

21:58:01.038605 IP 158.195.87.39 > 192.88.99.1: IP6 2002:9ec3:5727::1 >
2a00:1450:4001:807::101f: ICMP6, echo request, seq 2, length 64

21:58:01.060369 IP 192.88.99.1 > 158.195.87.39: IP6 2a00:1450:4001:807::101f >
2002:9ec3:5727::1: ICMP6, echo reply, seq 2, length 64
```

IPv6 router

- démon `radvd`
  - zabezpečuje posielanie Router Advertisement správ
  - pomocou nich sa automaticky nakonfigurujú klienti
  - konfigurácia v `/etc/radvd.conf`

```
interface eth0 {
  AdvSendAdvert on;
  prefix 2002:9ec3:5727::/64 {
    AdvOnLink on;
    AdvAutonomous on;
  };
};
```

IPv6 router

```
# ip addr add 2002:9ec3:5727::1/64 dev eth0
# ip -6 addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qlen 1000
    inet6 2002:9ec3:5727::1/64 scope global
       valid_lft forever preferred_lft forever
    inet6 fe80::1a03:73ff:fec1:1689/64 scope link
       valid_lft forever preferred_lft forever
96: tun6to4: <NOARP,UP,LOWER_UP> mtu 1480
    inet6 2002:9ec3:5727::1/16 scope global
       valid_lft forever preferred_lft forever
    inet6 ::158.195.87.39/128 scope global
       valid_lft forever preferred_lft forever
```

IPv6 router

```
# ip -6 route show
::/96 via :: dev tun6to4  metric 256
2002:9ec3:5727::/64 dev eth0  proto kernel  metric 256
2002::/16 dev tun6to4  proto kernel  metric 256
2000::/3 via ::192.88.99.1 dev tun6to4  metric 1024
fe80::/64 dev eth0  proto kernel  metric 256
fe80::/64 dev vnet0  proto kernel  metric 256
fe80::/64 dev dummy0  proto kernel  metric 256
fe80::/64 dev tun6to4  proto kernel  metric 256

# service radvd start
Starting radvd: radvd.
```

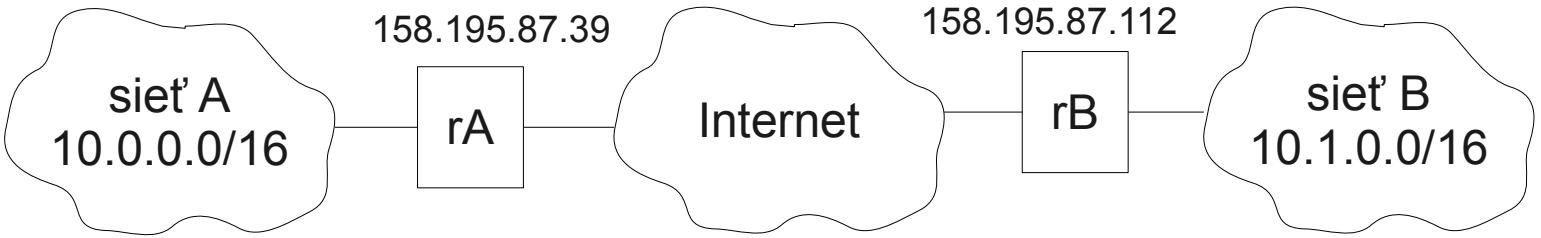
IPv6 klient

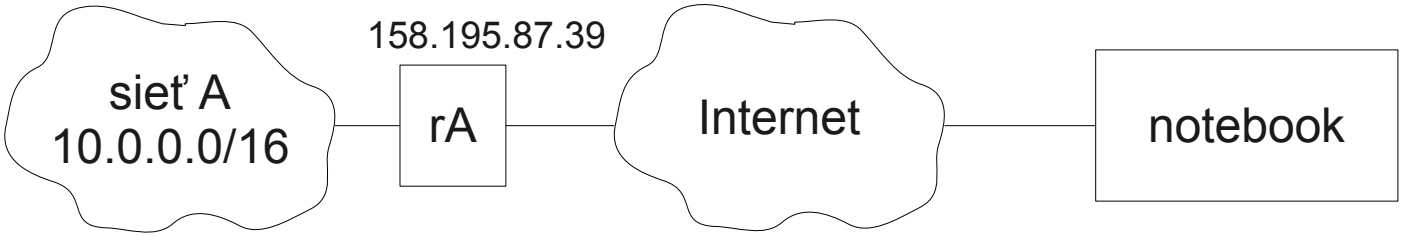
```
# ip -6 addr show dev eth0
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qlen 1000
    inet6 2002:9ec3:5727:0:21c:25ff:fe97:1166/64 scope global dynamic
       valid_lft 86286sec preferred_lft 14286sec
    inet6 fe80::21c:25ff:fe97:1166/64 scope link
       valid_lft forever preferred_lft forever
# ip -6 route show
2002:9ec3:5727::/64 dev eth0  proto kernel  metric 256  expires 86289sec
fe80::/64 dev eth0  proto kernel  metric 256
fe80::/64 dev wlan0  proto kernel  metric 256
fe80::/64 dev dummy1  proto kernel  metric 256
default via fe80::1a03:73ff:fec1:1689 dev eth0  proto kernel  metric 1024
expires 1683sec

# ping6 -n www.google.sk
PING www.google.sk(2a00:1450:4001:807::100f) 56 data bytes
64 bytes from 2a00:1450:4001:807::100f: icmp_seq=1 ttl=58 time=21.6 ms
64 bytes from 2a00:1450:4001:807::100f: icmp_seq=2 ttl=58 time=21.2 ms
```

VPN v OS Linux

- úloha
  - vytvoriť kryptograficky chránený komunikačný kanál, cez ktorý je možné posielať sieťovú komunikáciu
    - dôvernosť, integrita, autentifikácia koncov
- použitie
  - prepojenie sietí cez nedôveryhodnú sieť (Internet)
  - pripojenie počítača (road warrior) zvonku do internej siete

VPN v OS Linux





VPN v OS Linux

- príklad riešení
  - OpenVPN
    - multiplatformové open source riešenie
    - viruálne sieťové rozhranie (tun, tap)
    - výber paketov riešený smerovaním
  - strongSwan
    - IPSec (IKEv1, IKEv2)
    - využíva podporu IPSec v kerneli
    - výber paketov riešený IPSec politikou v kerneli

OpenVPN

- komunikačný kanál
  - UDP (preferovaný)
  - TCP
  - TCP cez HTTPS proxy (CONNECT)
- manažment kľúčov
  - staticky (manuálne)
  - dynamicky (po vzore TLS)
- autentifikácia
  - PKI
  - meno + heslo (pomocou skriptu)

OpenVPN

- konfigurácia typicky v `/etc/openvpn/`*`meno`*`.conf`
- `/etc/default/openvpn`
  - `AUTOSTART="..."` - zoznam VPN, ktoré sa majú automaticky spustiť pri štarte služby openvpn
- zapnutie / vypnutie konkrétnej VPN
  - `service openvpn start meno`
  - `service openvpn stop meno`

OpenVPN

```
#konfiguracia pre rA

proto udp
port 1194
dev tun
ifconfig 192.168.200.1 192.168.200.2
route 10.1.0.0 255.255.0.0
ping 15
ping-restart 60

tls-server
ca ca.pem
dh dh1024.pem
cert siteA-cert.pem
key siteA-key.pem
ns-cert-type client
```

```
#konfiguracia pre rB

remote 158.195.87.39
proto udp
port 1194
dev tun
ifconfig 192.168.200.2 192.168.200.1
route 10.0.0.0 255.255.0.0
ping 15
ping-restart 60

tls-client
ca ca.pem
cert siteB-cert.pem
key siteB-key.pem
ns-cert-type server
```

OpenVPN

- poznámky k certifikátom
  - dôležité je, aby bol v certifikátoch uvedený správny typ (alebo žiadny)
  - certifikát koreňovej CA aj všetkých medzistupňov musí byť v súbore „ca“, v súbore „cert“ len vlastný
  - ak by sa použili iné koreňové CA, budú potrebné všetky (aby vedel overiť aj partnera)

OpenVPN

```
# service openvpn start siteBA
Starting virtual private network daemon: siteBA.
# ip addr show dev tun0
121: tun0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 1500
qdisc pfifo_fast state UNKNOWN qlen 100
    link/none
    inet 192.168.200.2 peer 192.168.200.1/32 scope global tun0
# ip route show dev tun0
10.0.0.0/16 via 192.168.200.1
192.168.200.1  proto kernel  scope link  src 192.168.200.2
# ping 192.168.200.1
PING 192.168.200.1 (192.168.200.1) 56(84) bytes of data.
64 bytes from 192.168.200.1: icmp_req=1 ttl=64 time=0.488 ms
64 bytes from 192.168.200.1: icmp_req=2 ttl=64 time=0.551 ms
```

OpenVPN

```
# tcpdump -nlp -i tun0
tcpdump: verbose output suppressed, use -v or -vv for full protocol
decode
listening on tun0, link-type RAW (Raw IP), capture size 65535 bytes
18:51:12.501403 IP 192.168.200.2 > 192.168.200.1: ICMP echo
request, id 2563, seq 1, length 64
18:51:12.501871 IP 192.168.200.1 > 192.168.200.2: ICMP echo reply,
id 2563, seq 1, length 64
18:51:13.500416 IP 192.168.200.2 > 192.168.200.1: ICMP echo
request, id 2563, seq 2, length 64
18:51:13.500934 IP 192.168.200.1 > 192.168.200.2: ICMP echo reply,
id 2563, seq 2, length 64
```

OpenVPN

```
# tcpdump -nlp -i eth0
tcpdump: verbose output suppressed, use -v or -vv for full protocol
decode
listening on eth0, link-type EN10MB (Ethernet), capture size 65535
bytes
18:53:51.925542 IP 158.195.87.112.1194 > 158.195.87.39.1194: UDP,
length 125
18:53:51.925866 IP 158.195.87.39.1194 > 158.195.87.112.1194: UDP,
length 125
18:53:52.924368 IP 158.195.87.112.1194 > 158.195.87.39.1194: UDP,
length 125
18:53:52.924645 IP 158.195.87.39.1194 > 158.195.87.112.1194: UDP,
length 125
```

OpenVPN

- klient-server konfigurácia
  - jeden server môže obsluhovať viac klientov
  - server môže prideľovať klientom adresy
    - staticky (podľa konfigurácie pre konkrétnych klientov)
    - dynamicky
  - komunikácia medzi klientami môže byť povolená
    - `client-to-client`

OpenVPN

```
#konfiguracia pre rA ako server

proto udp
port 1194
dev tap
server 10.0.200.0 255.255.255.0
push "route 10.0.0.0 255.255.0.0"

#client-to-client

keepalive 15 60

ca ca.pem
dh dh1024.pem
cert siteA-cert.pem
key siteA-key.pem
ns-cert-type client
```

```
#konfiguracia pre klienta

remote 158.195.87.39
proto udp
port 1194
dev tap

client

ca ca.pem
cert siteB-cert.pem
key siteB-key.pem
ns-cert-type server
```

OpenVPN

```
# service openvpn start client
Starting virtual private network daemon: client.
# ip addr show dev tap0
125: tap0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
pfifo_fast state UNKNOWN qlen 100
    link/ether 2e:d7:b2:8b:41:06 brd ff:ff:ff:ff:ff:ff
    inet 10.0.200.2/24 brd 10.0.200.255 scope global tap0
# ip route show dev tap0
10.0.200.0/24  proto kernel  scope link  src 10.0.200.2
10.0.0.0/16 via 10.0.200.1
# ping 10.0.200.1
PING 10.0.200.1 (10.0.200.1) 56(84) bytes of data.
64 bytes from 10.0.200.1: icmp_req=1 ttl=64 time=0.939 ms
64 bytes from 10.0.200.1: icmp_req=2 ttl=64 time=0.627 ms
64 bytes from 10.0.200.1: icmp_req=3 ttl=64 time=0.636 ms
```

OpenVPN

- statická konfigurácia parametrov pre klienta
  - pomocou `client-config-dir` sa zvolí konfiguračný adresár
  - vytvoria sa súbory s menom ako CN v klientskom certifikáte
    - z týchto sa načítajú konfiguračné príkazy špecifické pre konkrétneho klienta

OpenVPN

```
#konfiguracia pre rA ako server

proto udp
port 1194
dev tap

server 10.0.200.0 255.255.255.0 nopool
push "route 10.0.0.0 255.255.0.0"

keepalive 15 60

ca ca.pem
dh dh1024.pem
cert siteA-cert.pem
key siteA-key.pem
ns-cert-type client

client-config-dir clients
```

Súbor `clients/siteB`

```
ifconfig-push 10.0.200.5 255.255.255.0
```

strongSwan

- komunikačný kanál
  - UDP pre IKE (port 500)
  - ESP, AH pre dáta
    - UDP enkapsulácia pre NAT-traversal (port 4500)
- manažment kľúčov
  - IKE (v1, v2)
- autentifikácia
  - PSK (preshared key)
  - PKI
  - EAP

strongSwan

- pozostáva z 2 procesov
  - pluto – IKEv1
  - charon – IKEv2
- konfigurácia
  - /etc/ipsec.conf
  - /etc/ipsec.secrets
  - /etc/ipsec.d/certs – certifikáty
  - /etc/ipsec.d/cacerts – certifikáty CA
  - /etc/ipsec.d/private – kľúče

strongSwan

- spustenie služby
  - `service ipsec start`
- po zmene konfigurácie
  - `ipsec update`
- spustenie VPN
  - `ipsec up meno`
- ukončenie VPN
  - `ipsec down meno`
- zobrazenie stavu
  - `ipsec status, ipsec statusall`

strongSwan

```
# konfigurácia pre rA

config setup
    nat_traversal=yes
    charonstart=yes
    plutostart=no

conn siteAB
  authby=pubkey
  auto=add
  keyexchange=ikev2
  left=158.195.87.39
  leftcert=siteA-cert.pem
  right=158.195.87.112
  rightid="C=SK,
L=Bratislava,O=Comenius
University,  OU=Department of
Computer Science, CN=siteB"
  leftsubnet=10.0.0.0/16
  rightsubnet=10.1.0.0/16
```

```
# konfigurácia pre rB

config setup
    nat_traversal=yes
    charonstart=yes
    plutostart=no

conn siteAB
  authby=pubkey
  auto=add
  keyexchange=ikev2
  left=158.195.87.39
  leftid="C=SK, L=Bratislava,
O=Comenius University,
OU=Department of Computer
Science, CN=siteA"
  right=158.195.87.112
  rightcert=siteB-cert.pem
  leftsubnet=10.0.0.0/16
  rightsubnet=10.1.0.0/16
```

strongSwan

```
# ipsec.secrets pre rA

: RSA siteA-key.pem
```

```
# ipsec.secrets pre rB

: RSA siteB-key.pem
```

```
# ipsec up siteAB
# ipsec status
Security Associations:
      siteAB[5]: ESTABLISHED 8 seconds ago, 158.195.87.112[C=SK,
L=Bratislava, O=Comenius University, OU=Department of Computer
Science, CN=siteB]...158.195.87.39[C=SK, L=Bratislava, O=Comenius
University, OU=Department of Computer Science, CN=siteA]
      siteAB{5}:  INSTALLED, TUNNEL, ESP SPIs: c78fb8ed_i
c24489f2_o
      siteAB{5}:   10.1.0.0/16 === 10.0.0.0/16
```

strongSwan

```
# ip rule show
0:  from all lookup local
220:   from all lookup 220
32766: from all lookup main
32767: from all lookup default

# ip route show table 220
10.0.0.0/16 via 158.195.87.39 dev eth0  proto static  src 10.1.0.1

# ping 10.0.0.1
PING 10.0.0.1 (10.0.0.1) 56(84) bytes of data.
64 bytes from 10.0.0.1: icmp_req=1 ttl=64 time=0.375 ms
64 bytes from 10.0.0.1: icmp_req=2 ttl=64 time=0.414 ms
```

strongSwan

```
# konfigurácia pre rA ako server

...

conn rw
  authby=pubkey
  auto=add
  keyexchange=ikev2
  left=158.195.87.39
  leftcert=siteA-cert.pem
  right=%any
  leftsubnet=10.0.0.0/16
  rightsourceip=10.0.200.0/24
```

```
# konfigurácie pre klienta

...

conn vpn
  authby=pubkey
  auto=add
  keyexchange=ikev2
  left=158.195.87.39
  leftid="C=SK, L=Bratislava,
O=Comenius University,
OU=Department of Computer
Science, CN=siteA"
  leftsubnet=10.0.0.0/16
  right=%defaultroute
  rightcert=siteB-cert.pem
  rightsourceip=%config
```

strongSwan

```
# ipsec up vpn
…
installing new virtual IP 10.0.200.1

# ipsec status
Security Associations:
        vpn[4]: ESTABLISHED 3 minutes ago, 158.195.87.112[C=SK,
L=Bratislava, O=Comenius University, OU=Department of Computer
Science, CN=siteB]...158.195.87.39[C=SK, L=Bratislava, O=Comenius
University, OU=Department of Computer Science, CN=siteA]
        vpn{4}:  INSTALLED, TUNNEL, ESP SPIs: c9111b97_i c21a11cf_o
        vpn{4}:   10.0.200.1/32 === 10.0.0.0/16
```

strongSwan

```
# ip addr show eth0
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
state UP qlen 1000
    link/ether 00:1c:25:97:11:66 brd ff:ff:ff:ff:ff:ff
    inet 158.195.87.112/25 brd 158.195.87.127 scope global eth0
    inet 10.0.200.1/32 scope global eth0

# ip route show table 220
10.0.0.0/16 via 158.195.87.39 dev eth0  proto static  src 10.0.200.1
# ping 10.0.0.1
PING 10.0.0.1 (10.0.0.1) 56(84) bytes of data.
64 bytes from 10.0.0.1: icmp_req=1 ttl=64 time=0.379 ms
64 bytes from 10.0.0.1: icmp_req=2 ttl=64 time=0.422 ms
```