Hálózatok építése és üzemeltetése

Hálózati funkciók a gyakorlatban

Hol tartunk?

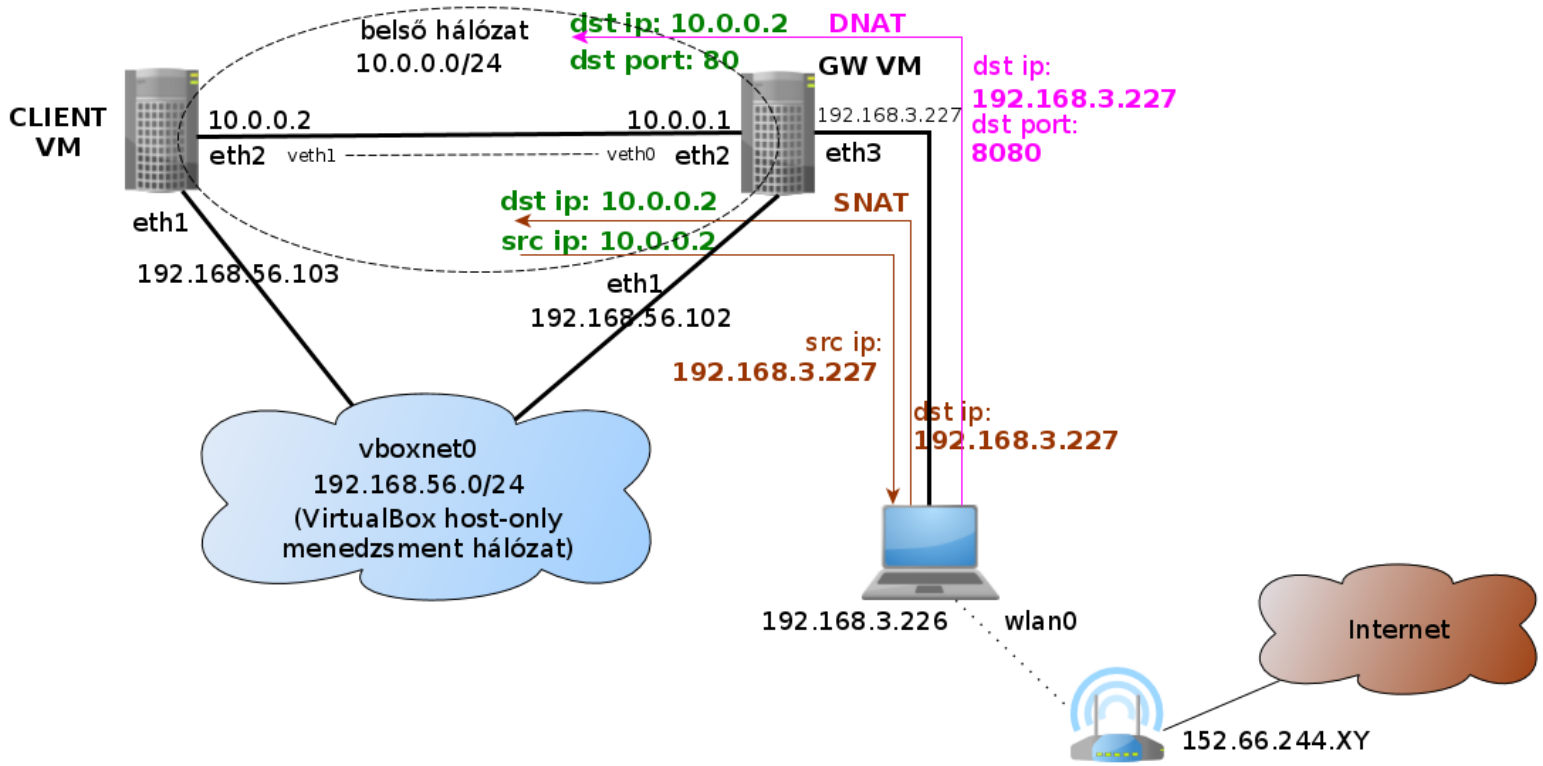
- UNIX/Linux alapok
  - történelem, GNU/Linux rendszerek felépítése, alapvető parancsok, “user mode”, szűrők, Bash alapok…
- Linux hálózatkezelés (és Linux admin alapok)
  - root jogosultság, partíciók, fájlrendszerek, Linux boot folyamata, “service”-ek, csomagok, hálózatkezelés alapok
- Szoftver szerszámok hálózatkezeléshez
  - ifconfig, route, ip (iproute2)
  - ping, traceroute, netstat, tcpdump, wireshark
  - bash, python, scapy

Mai téma

- Egyszerű hálózat bekonfigurálása
- egy konkrét példán keresztül
- lépésről-lépésre

A példa hálózatunk

Hálózati elrendezés



Előkészítés

- Virtuális link létrehozása
  - sudo ip link add veth0 type veth peer name veth1
  - sudo ifconfig veth0 up; sudo ifconfig veth1 up
  - virtuális Ethernet pár
    - egyik oldalon BE, másikon KI
- Virtuális gépek konfigurálása (VirtualBox)
  - gateway (GW)
    - eth1: “Host-only Adapter”, menedzsment interfész
      - hoszt gépről tudunk belépni egy belső hálózaton
    - eth2: “Bridged Adapter”, veth0
      - virtuális link bekötése
    - eth3: “Bridged Adapter”, wlan0
      - megkapja a hoszt gép wlan interfészét is bridge módban
  - kliens (CLIENT)
    - eth1: “Host-only Adapter”, menedzsment interfész
    - eth2: “Bridged Adapter”, veth1

Start!

- Virtuális gépek indítása
  - belépés a menedzsment interfészen
    - ssh -Y mininet@192.168.56.102 (GW)
    - ssh -Y mininet@192.168.56.103 (CLIENT)
  - interfészek manuális konfigurálása
    - (NAT interfész leállítása: sudo ifdown eth0)
    - virtuális összeköttetés a VM-ek között
      - sudo ifconfig eth2 up
    - GW: külső kapcsolat beállítása
      - sudo ifconfig eth3 up
      - sudo dhclient –v eth3
    - közben ellenőrizzük a
      - routing táblát (route -n)
      - névfeloldás beállítását (cat /etc/resolv.conf)

Összeköttetés tesztelése

- eth2 interfészek konfigurálása
  - GW: sudo ifconfig eth2 10.0.0.1/24
  - CLIENT: sudo ifconfig eth2 10.0.0.2/24
  - ping?
    - ping 10.0.0.2 <-> ping 10.0.0.1
  - web?
    - lynx 10.0.0.2
  - ssh?
    - ssh 10.0.0.2

Hogyan tovább?

- Belső hálózatról (CLIENT) szeretnénk “netezni”
- Azért minden forgalmat nem szeretnénk engedélyezni
- Manuális konfiguráció???

Hogyan tovább?

- Belső hálózatról (CLIENT) szeretnénk “netezni”
  - NAT
- Azért minden forgalmat nem szeretnénk engedélyezni
- Manuális konfiguráció???

Hogyan tovább?

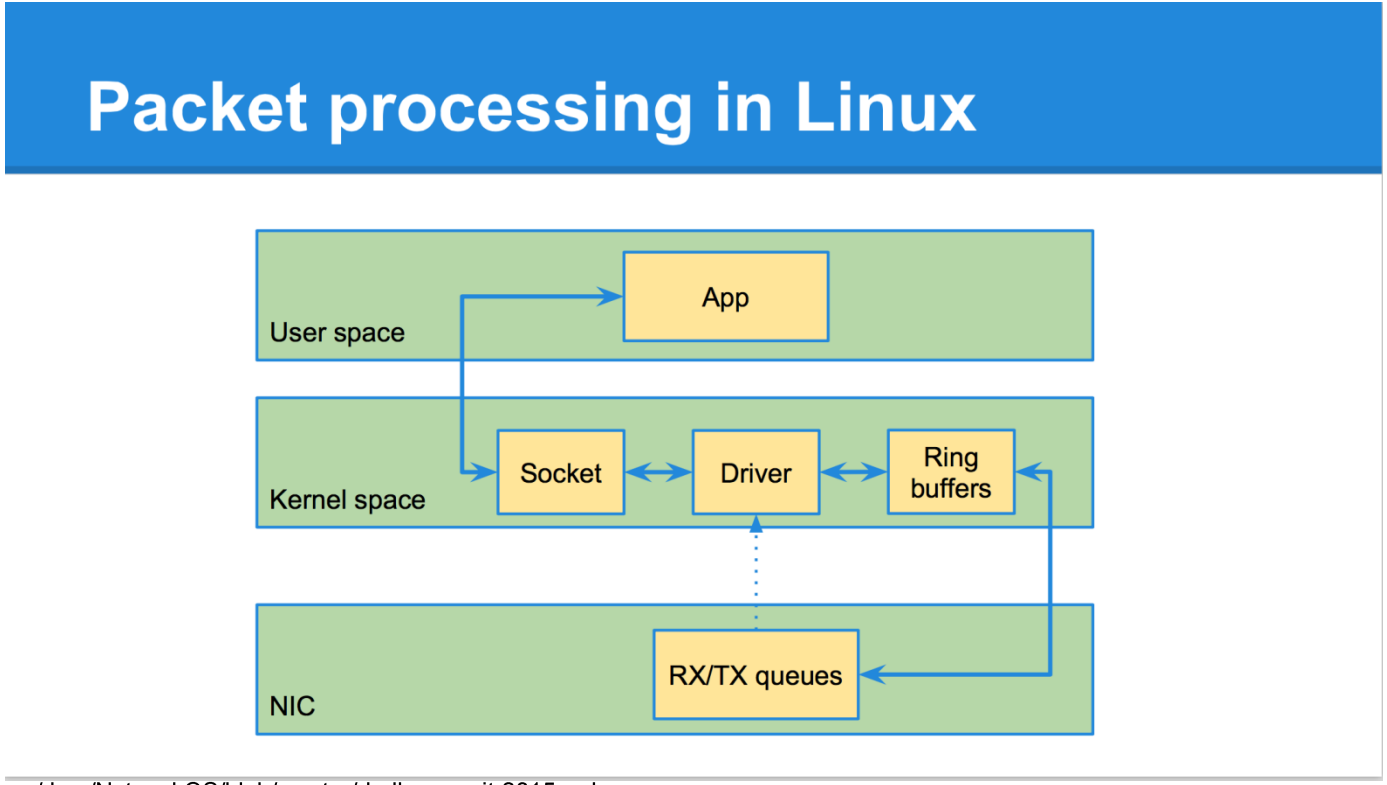
- Belső hálózatról (CLIENT) szeretnénk “netezni”
  - NAT
- Azért minden forgalmat nem szeretnénk engedélyezni
  - Firewall
- Manuális konfiguráció???

Hogyan tovább?

- Belső hálózatról (CLIENT) szeretnénk “netezni”
  - NAT
- Azért minden forgalmat nem szeretnénk engedélyezni
  - Firewall
- Manuális konfiguráció???
  - DHCP, DNS

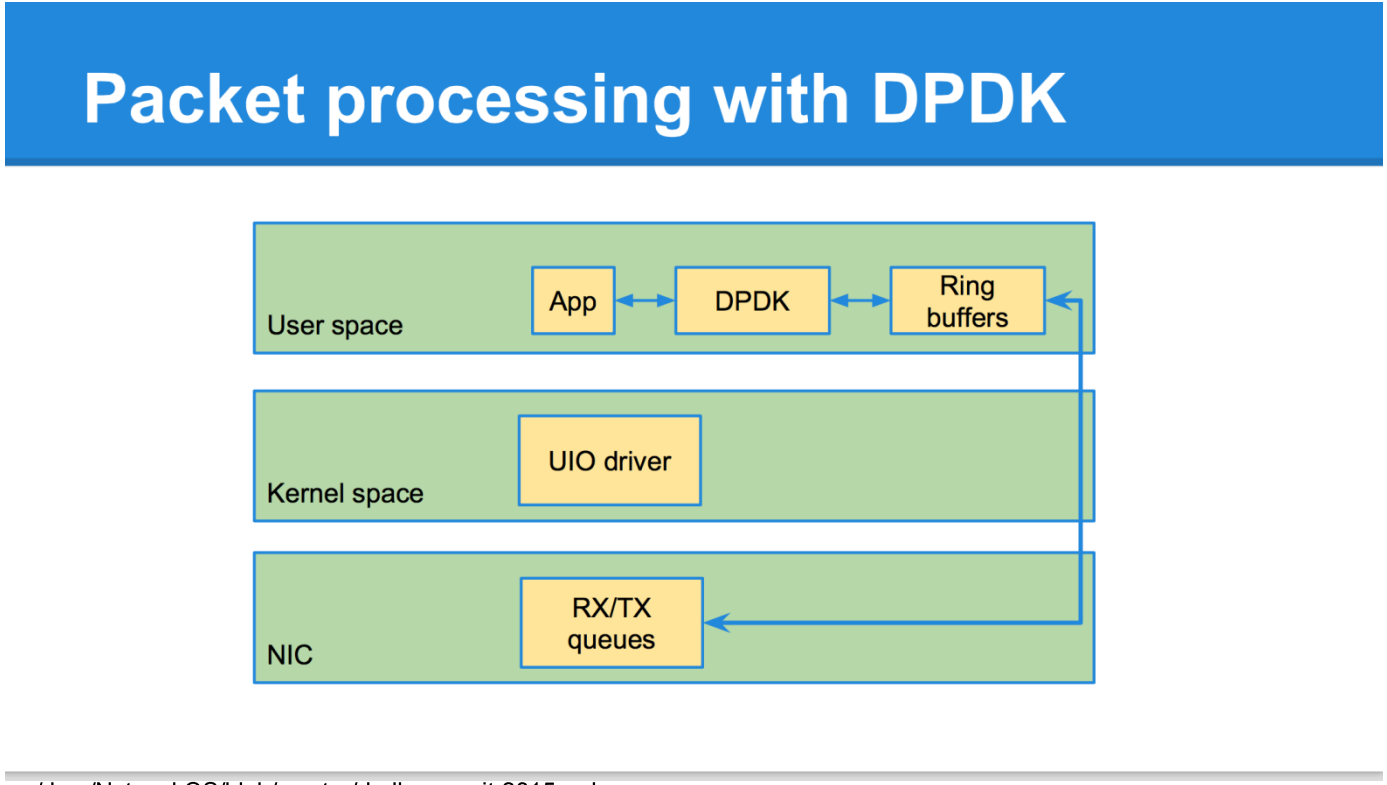
Egy csomag útja a Linux rendszerben

NIC-kernel space-user space



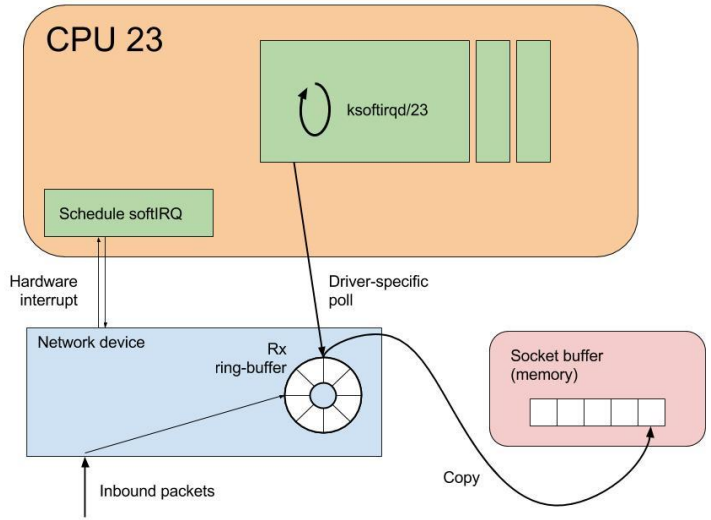
forrás: https://github.com/rkuo/NetworkOS/blob/master/dpdk-summit-2015.md

(Érdekesség)



forrás: https://github.com/rkuo/NetworkOS/blob/master/dpdk-summit-2015.md

NAPI – New API (2.6-os kerneltől)



- Régebbi kernel verziókban
  - hardware interrupt alapú működés
  - minden beérkező csomagra IRQ!
  - nem hatékony megoldás
- NAPI
  - poll mode
    - periodikus ellenőrzés
    - egyszerre sok csomag betöltése
  - hw IRQ -> softIRQ ütemezése adott CPU-n
  - feldolgozó processz: ksoftirqd/<cpu-id>
    - megosztható feladatok a core-ok között
  - NIC driver poll függvényét hívja
  - másolás a socket bufferbe
  - ezután jön a teljes network stack

forrás: https://www.lmax.com/blog/staff-blogs/2016/05/06/navigating-linux-kernel-network-stack-receive-path/

Csomagfeldolgozás

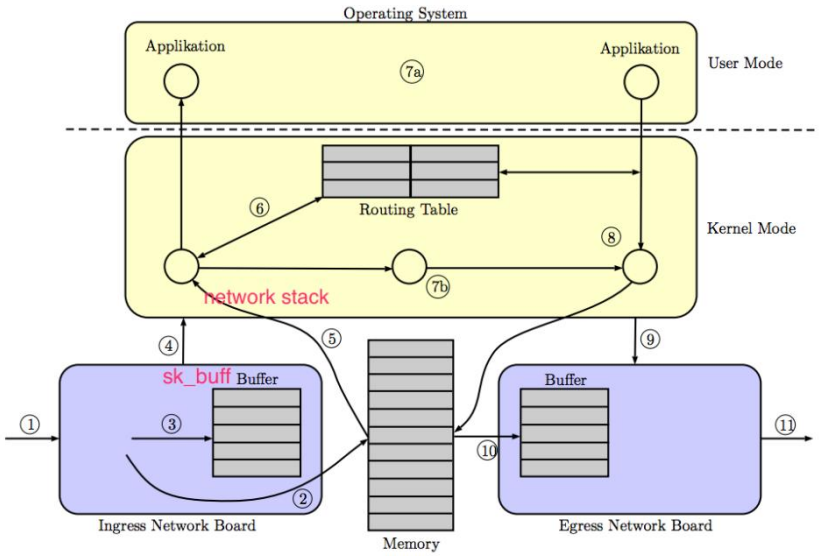


Figure 1: Abstract model of the packet processing steps in Unix-based software routers

- (1) csomag érkezik a hálókártyára
- sk_buff struktúra
  - socket kernel buffer
  - ahogy a kernel reprezentál egy csomagot
- NIC driver ring buffer (3)
  - pointerek az sk_buff struktúrákra
- (2) NIC “be-DMA-zza” a memóriába
- (4) hw IRQ
  - jelzés, hogy van csomag
  - softIRQ ütemezése adott CPU-n
  - NIC hozzáadása a poll_list listához
- (5) ksoftirqd/x:
  - softIRQ handler
  - eszköz “pollozása”
  - minden csomagra a network stack végrehajtása

forrás: https://github.com/rkuo/NetworkOS/blob/master/dpdk-summit-2015.md

Csomagfeldolgozás

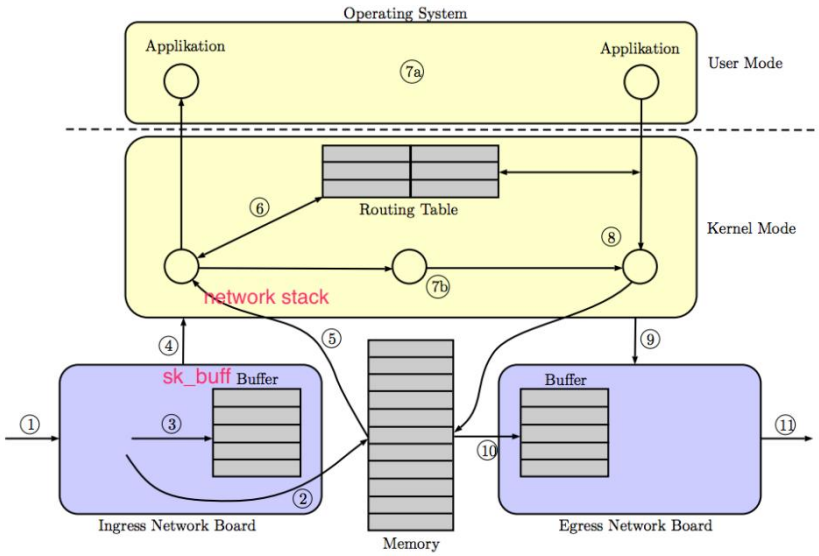


Figure 1: Abstract model of the packet processing steps in Unix-based software routers

- csomagellenőrzés (integrity-verification, cheksum)
  - ha nem OK, eldobja
- **firewall szabályok**
  - routing előtt és után
  - (jön részletesen)
- (6) routing alrendszer
  - ha ide érkezett a csomag
    - tovább a transzport rétegnek
    - (7a) socket API-n keresztül az alkalmazásnak
    - sk_buff -> user space másolás
  - ha nem ide érkezett a csomag
    - (7b) forwarding
    - routing algoritmus, routing tábla alapján

forrás: https://github.com/rkuo/NetworkOS/blob/master/dpdk-summit-2015.md

Csomagfeldolgozás

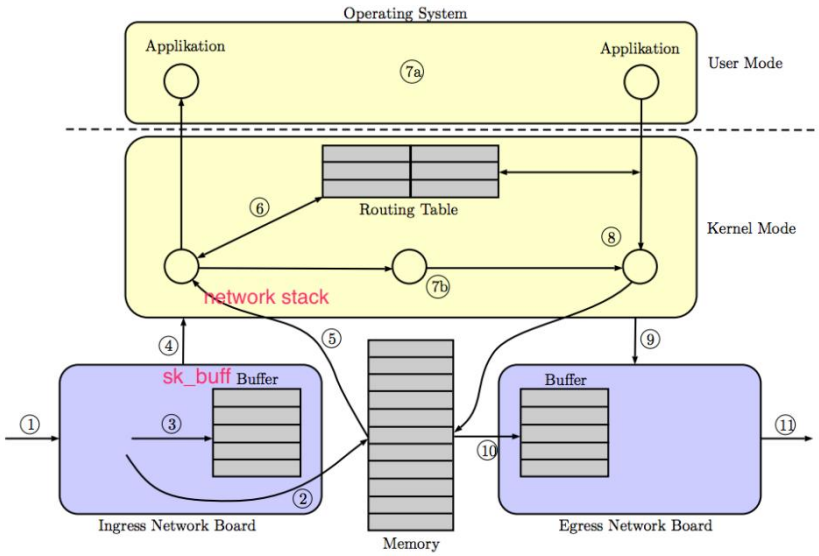
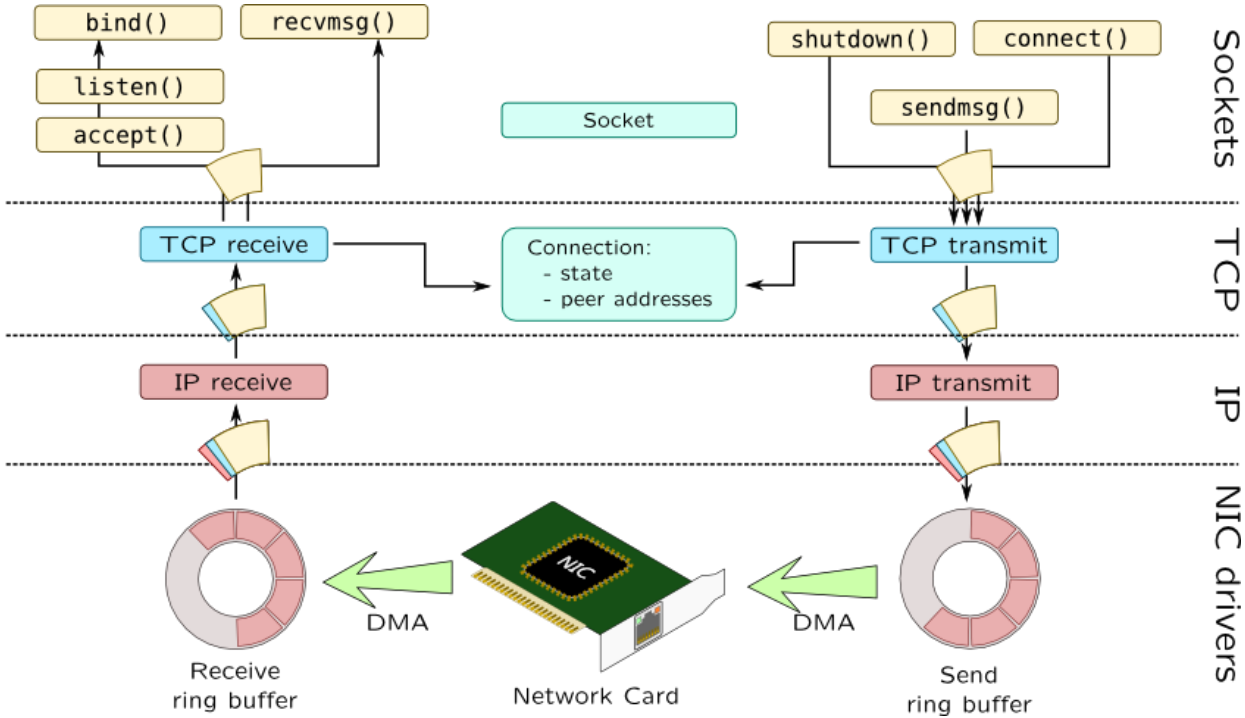


Figure 1: Abstract model of the packet processing steps in Unix-based software routers

- (8) csomag küldése
  - lokális alkalmazás generálta
  - vagy továbbítandó csomag
- next hop mac címét ki kell találni
  - ARP
- (9) NIC driver csomagküldés függvénye
  - (10) csomag leíró betöltése a ring bufferbe
    - sk_buff pointer
  - (11) csomag küldésre készen
- ha sikeres küldés
  - interrupt a CPU-nak
  - sk_buff felszabadítható

forrás: https://github.com/rkuo/NetworkOS/blob/master/dpdk-summit-2015.md

Mindez felülről



forrás: http://myaut.github.io/dtrace-stap-book/kernel/net.html

NAT

iptables

Network Address Translation

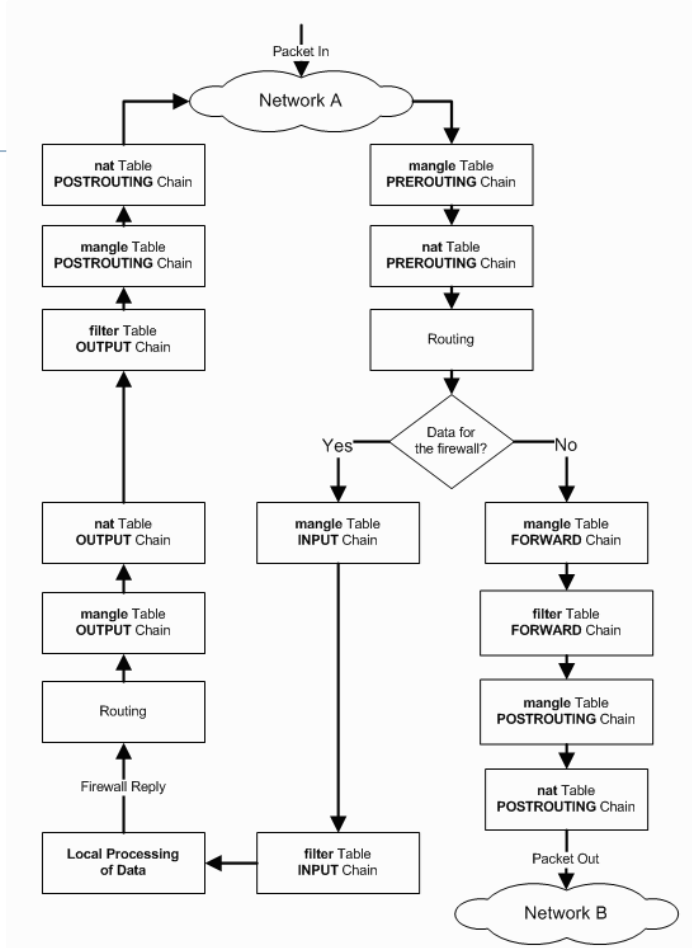
- NAT
  - olyan router, ami megváltoztatja a forrás vagy/és cél IP címet egy csomagban
  - leggyakrabban privát IP alhálózatot kapcsol a publikus internethez
- PAT (Port Address Translation)
  - a forrás vagy/és cél TCP/UDP port számot módosítja
  - általában beleértjük a NAT-ba
- SNAT (Source NAT)
  - forrás címet cserél a (kimenő) csomagokon egy fix címre
- Masquerading
  - forrás címet cserél a kimenő csomagokon dinamikus címre
- DNAT (Destination NAT)
  - cél címet cserél
- Port forwarding
  - DNAT, amikor külső hálózatról engedünk forgalmat a privát alhálózatba
  - külvilág számára látható ip:port számot kell a belső tartományra fordítani

iptables

- először “NAT”-olásra fogjuk használni
- általános célú csomagszűrő
- Linux alatt a csomagszűrés a kernel része (1.1 verziótól)
  - vagy teljesen bele van fordítva
  - vagy modulként tölthető be
- Rövid története
  - első változat
    - BSD UNIX ipfw programjára épült (Alan Cox portolta 1994 végén)
  - Linux 2.0
    - ipfwadm parancs kontrollálta a csomagszűrést
  - Linux 2.2
    - nagymértékben újraírták a kódot
    - új kezelőoldali alkalmazás: ipchains
  - Linux 2.4
    - iptables és a hozzá kapcsolódó kernelrészek újraírása

iptables

- Alapelemek: szabályok, láncok, táblák
  - láncokban tárolt szabályokon “megy” végig a csomag
  - ha illeszkedés van, target végrehajtása
- Három beépített tábla
  - filter: csomagok szűrése, szortírozása
  - **nat: címfordítási feladatok**
  - mangle: általános csomagmódosítások
- Hivatkozás
  - -t nat
  - -t filter (ez a default)
  - -t mangle
- Például
  - nat tábla listázása:
  - sudo iptables -t nat -nvL



SNAT konfigurálása (GW)

- Első lépés: forwarding engedélyezése
  - alapból nem tudjuk routerként használni a gépünket
    - `cat /proc/sys/net/ipv4/ip_forward`
  - engedélyezés
    - `sudo sysctl -w net.ipv4.ip_forward=1`
- címfordítás beállítása
  - `iptables -t nat`
  - `-A POSTROUTING  (append, új szabály hozzáfűzése)`
  - `-s 10.0.0.0/24  (ha ez a source IP)`
  - `-o eth3         (ha ez az output interfész)`
  - `-j SNAT         (akkor source IP fordítása)`
  - `--to-source 192.168.3.227    (erre a címre)`

SNAT konfigurálása (CLIENT)

- default gateway beállítása
  - `sudo route add default gw 10.0.0.1 dev eth2`
- tesztelés
  - ping 8.8.8.8
  - ping index.hu ???

DNAT konfigurálása (GW)

- Adott portokon tegyük elérhetővé kívülről a belső gép
  - web szerverét (8080)
  - ssh szerverét (2222)
- címfordítás beállítása
  - `iptables -t nat`
  - `-A PREROUTING` (append, új szabály hozzáfűzése)
  - `-d 192.168.3.227` (ha ez a destination IP)
  - `-p tcp` (ha TCP protokoll)
  - `--dport 8080` (és 8080-as TCP destination port)
  - `-j DNAT` (akkor destination IP:port fordítása)
  - `--to-destination 10.0.0.2:80` (a belső web szerverre)
- hasonlóan
  - `iptables -t nat -A PREROUTING -d 192.168.3.227 -p tcp --dport 2222 -j DNAT --to-destination 10.0.0.2:22`

DNAT tesztelése

- Hoszt gépről
  - web browser
    - http://192.168.3.227
    - http://192.168.3.227:8080
  - ssh
    - ssh mininet@192.168.3.227
    - ssh mininet@192.168.3.227 –p 2222