Telekommunikációs Hálózatok

9. gyakorlat

HÁLÓZATI CÍMFORDÍTÁS

NAT, porttovábbítás, SSH Tunnel, iptables

Hálózati címfordítás (NAT)

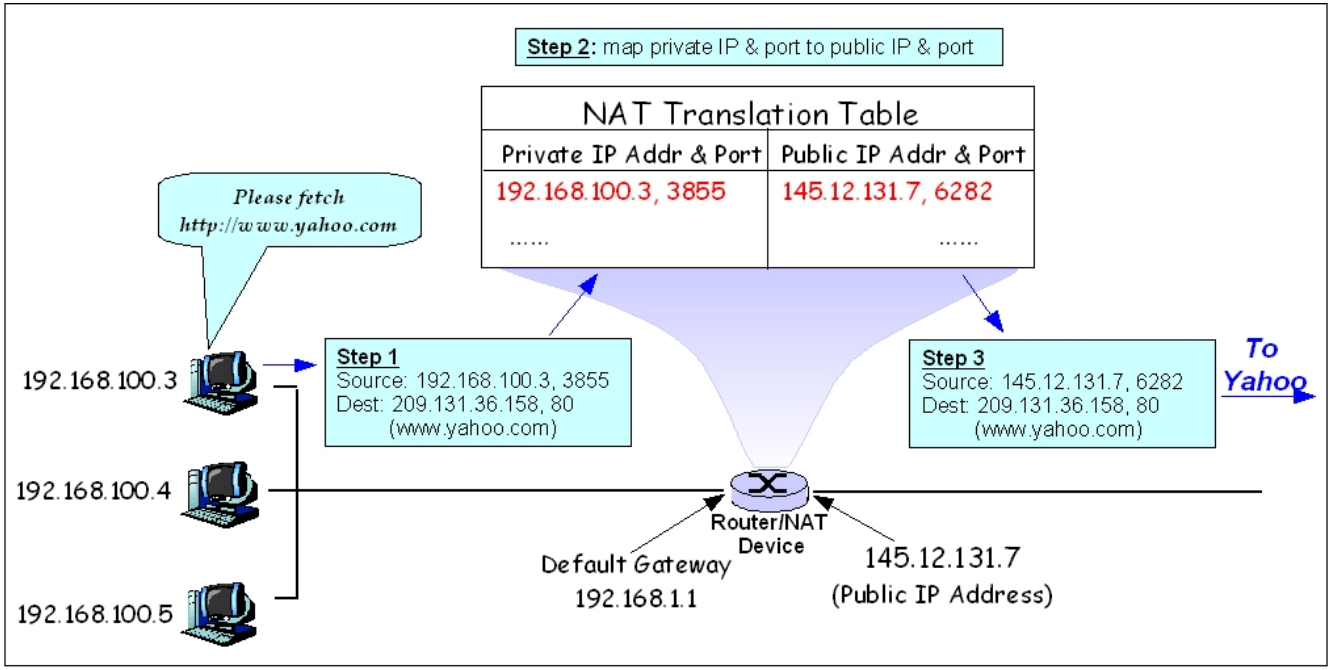
- Gyors javítás az IP címek elfogyásának problémájára.
- Az internet forgalomhoz minden cégnek egy vagy legalábbis kevés IP címet adnak (publikus IP cím(ek))
- A publikus IP cím hozzá van rendelve egy router-hez, a helyi hálózaton (LAN) belül, - amely mögötte van, - minden eszközhöz egy privát IP cím van rendelve
- A privát IP címek csak a LAN-on belül érvényesek (vannak IP cím tartományok erre a célra foglalva)

Hálózati címfordítás (NAT)

- Ha a helyi hálózaton lévő másik géppel akarunk kapcsolatot létesíteni → közvetlenül el tudjuk érni
- Amikor helyi eszközről akarunk egy külső eszközt elérni, mi történik?
- Szükségünk van port mezők használatára, ami TCP-nél vagy UDP-nél van

Hálózati címfordítás (NAT)

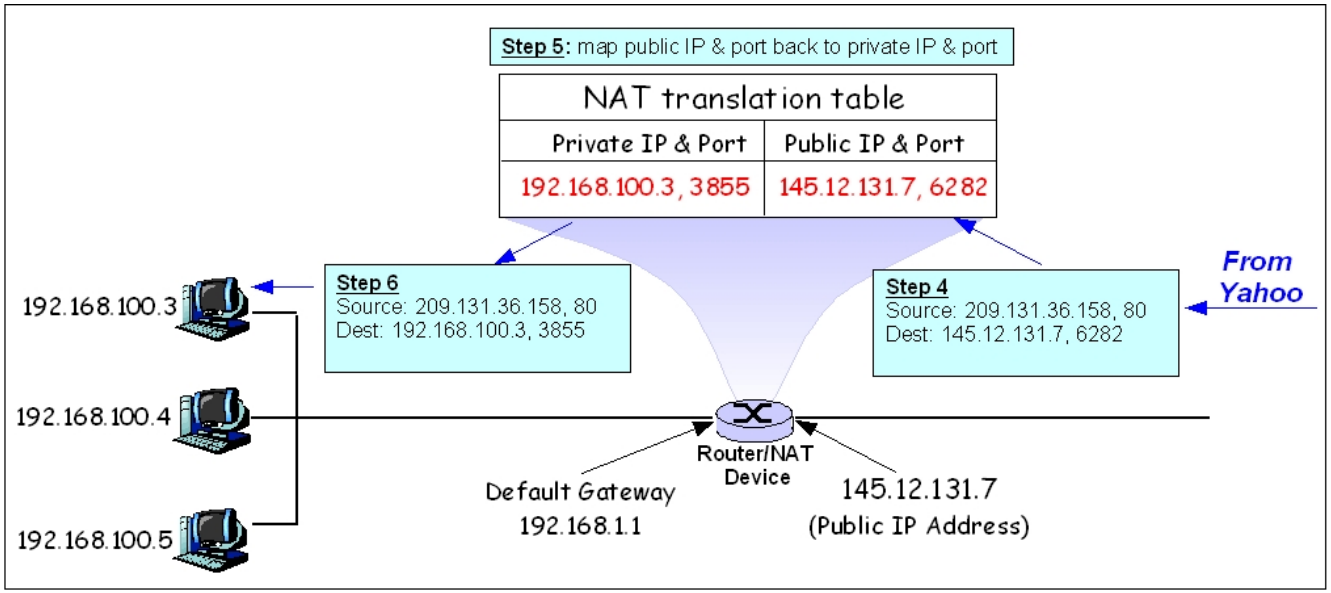
Forrás: https://en.wikibooks.org/wiki/Communication_Networks/NAT_and_PAT_Protocols



- 192.168.100.3 privát IP című gépről HTTP kérés, 3855 porton → Default gateway (192.168.1.1): megnézi a transzlációs tábláját:
  - Ha létezik már a (192.168.100.3, 3855) párhoz (publikus IP cím, port) bejegyzés → lecseréli a küldő forrását arra
  - Ha nincs létrehoz egy új bejegyzést (egyedi lesz!), és azt használja fel a cseréhez

Hálózati címfordítás (NAT)

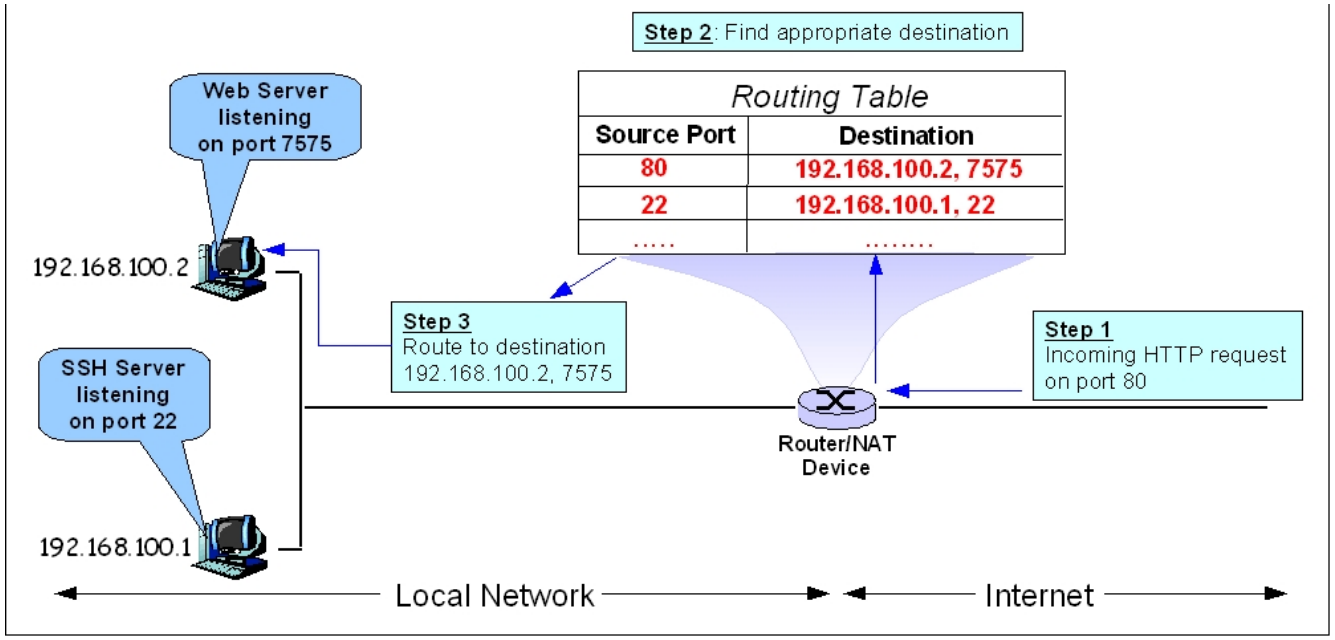
Forrás: https://en.wikibooks.org/wiki/Communication_Networks/NAT_and_PAT_Protocols



- A HTTP válasz a yahoo-tól ugyanúgy a router transzlációs tábláján keresztül megy végbe, csak fordított irányban
- Egy különbség: hiányzó bejegyzés esetén a csomagot eldobja a router

Porttovábbítás (port forwarding)

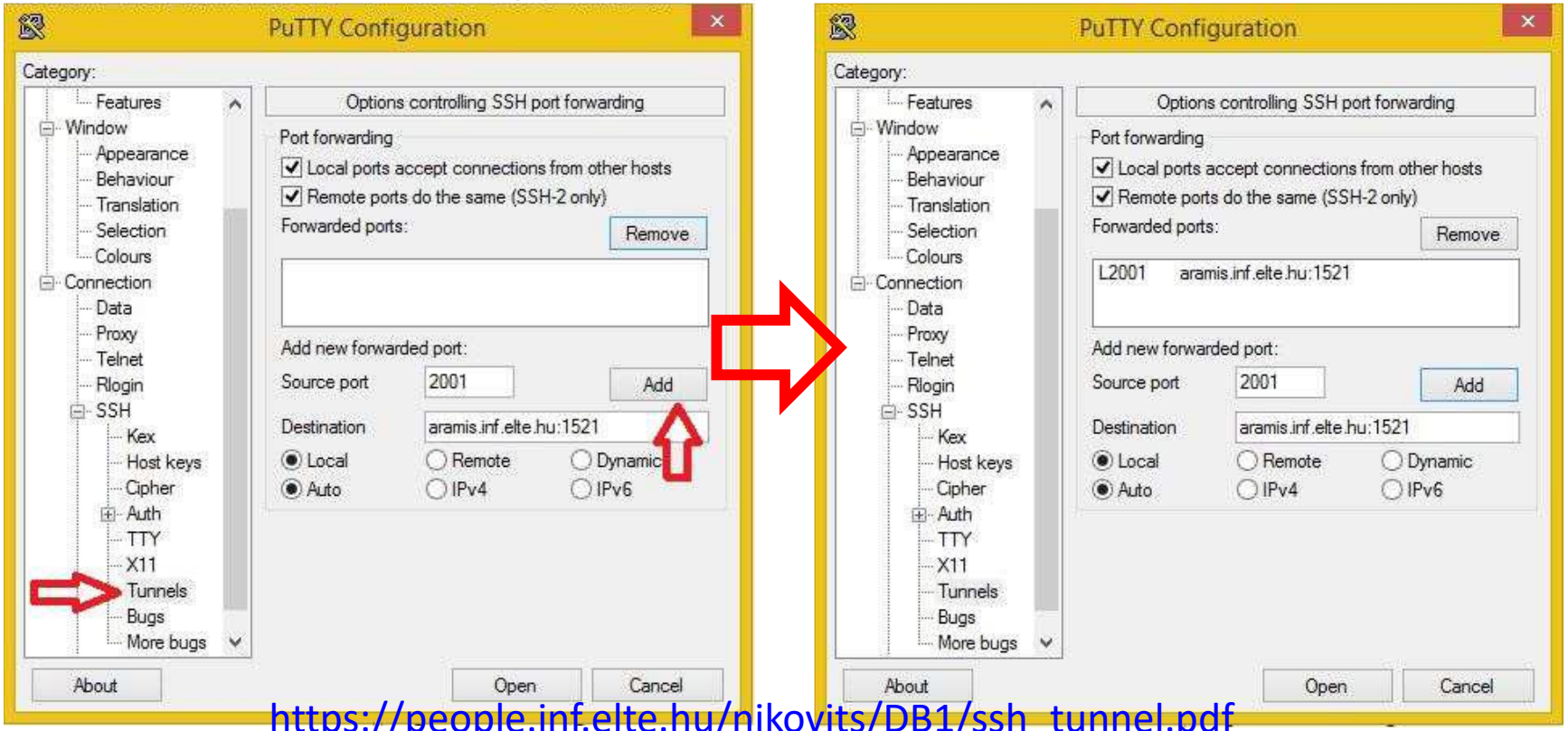
Forrás: https://en.wikibooks.org/wiki/Communication_Networks/NAT_and_PAT_Protocols



- Az előző példánál a címfordítás transzparens volt (csak a router tudott arról, hogy IP konverzió zajlik). Mit lehet tenni, ha pl. egy belső hálózaton lévő HTTP szervert akarunk elérni kívülről?
- **Porttovábbítás** lehetővé teszi adott lokális hálózaton (LAN) lévő privát IP címek külső elérését egy megadott porton keresztül
- Gyakorlatilag ez a *statikus* NAT alkalmazása

SSH Tunnel

- A porttovábbítás egyik tipikus alkalmazása
- Windows (putty) beállítások
  - (Nyitni kell egy ssh kapcsolatot a caesar.elte.hu-ra)



https://people.inf.elte.hu/nikovits/DB1/ssh_tunnel.pdf

SSH Tunnel

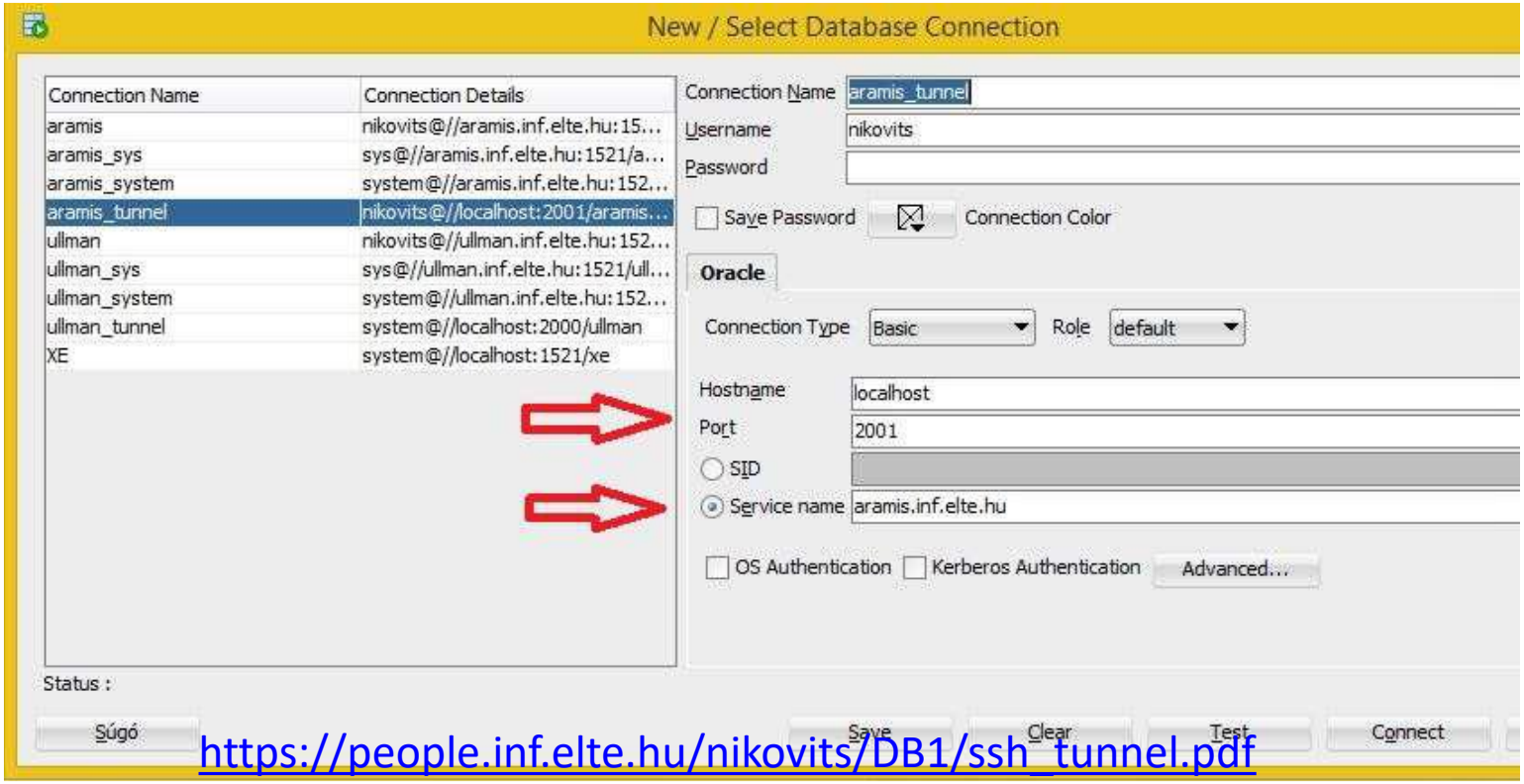
- Linux

```
ssh –L 2001:aramis.inf.elte.hu:1521 user@hostname
```

- ssh –L <localport>:<remote host>:<remote port> <gateway you can ssh in>
  - localport: a localhost ezen porján lesz elérhető a távoli szerver/szolgáltatás
  - remote host:remote port: ide csatlakozik a tunnel végpont, minden, amit a localportra küldünk ide fog továbbítódni és vissza. A gateway-ről elérhetőnek kell lennie!
  - gateway: a gép, amire be tudunk sshval lépni!

SSH Tunnel

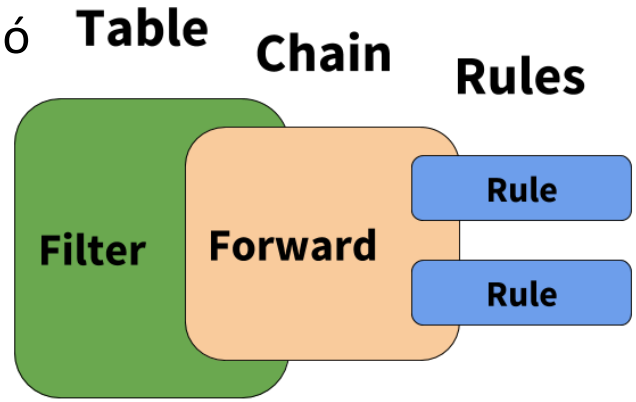
- Használat SqlDeveloper-nél:
  - (ssh kapcsolatnak fenn kell állnia végig az adatbázis kapcsolat ideje alatt)

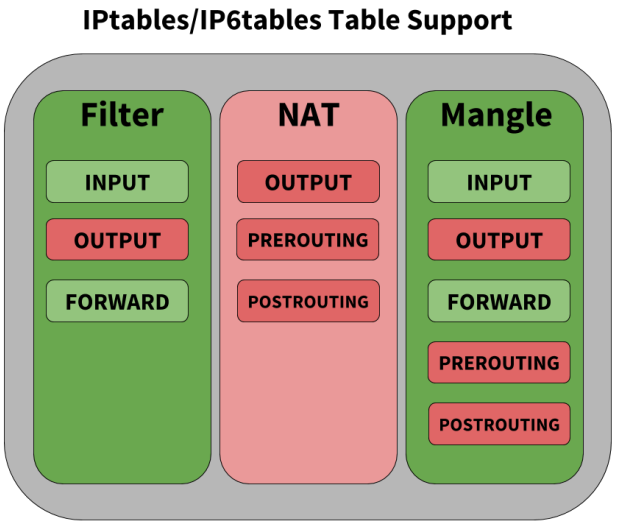


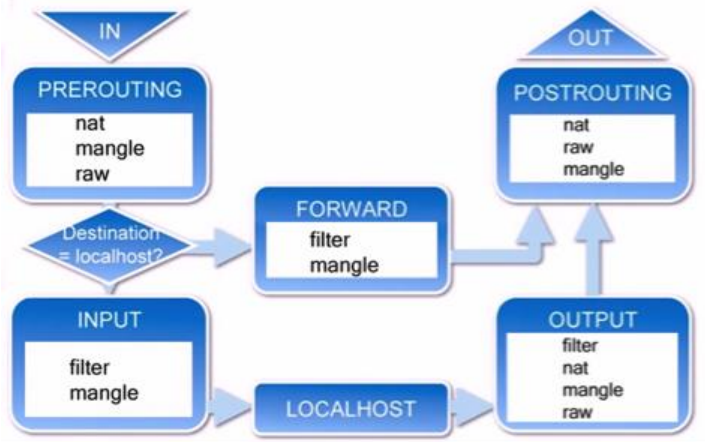
https://people.inf.elte.hu/nikovits/DB1/ssh_tunnel.pdf

iptables

- Az iptables egy Linux alkalmazás, amellyel a felhasználó konfigurálni tud tűzfal funkcionalitást, ill. csomagszűrési/csomagtovábbítási szabályokra, NAT módosítására/lekérdezésére jó







iptables

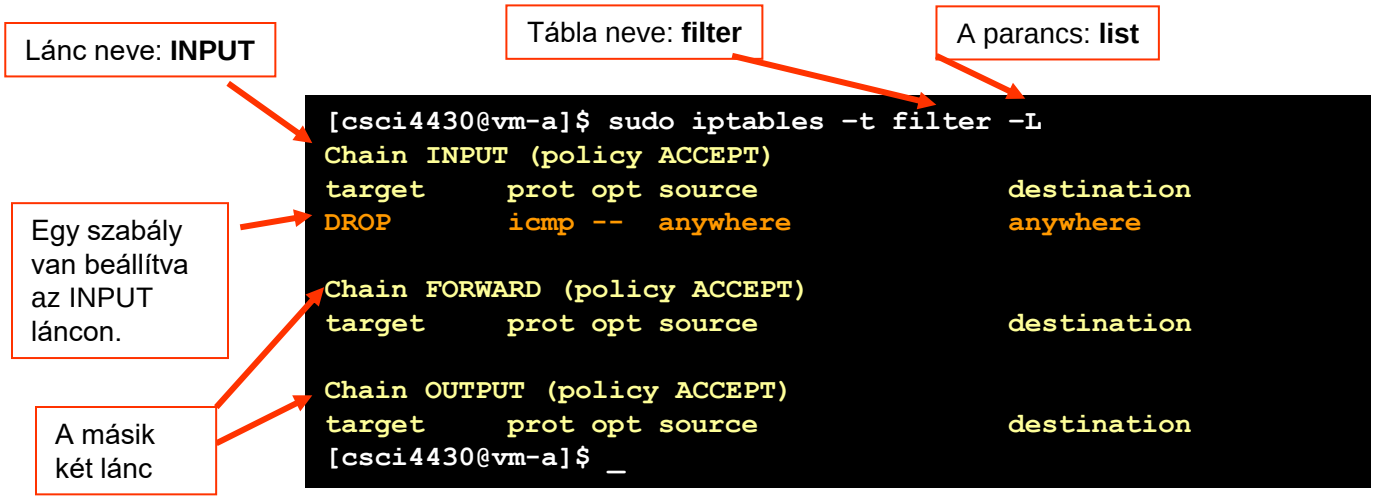
- Alapból három tábla van, amely szabályok halmazait tartalmazza
- A **filter** tábla a csomag szűrésre való
- A **nat** tábla a címfordításra való
- A **mangle** tábla a csomagok speciális célú feldolgozására való (megváltoztatja a csomagok tartalmát)
- Mindegyik táblában szabályok sorozata van, amelyeket láncoknak hívunk

iptables – filter tábla

- Itt három lánc van:
- Az INPUT láncot (az ott megadott szabályok sorozatát) bármely rendszerhez beérkező csomagra használja az alkalmazás
- Az OUTPUT láncot bármely olyan csomagra, amely a rendszerből kilép
- A FORWARD láncot pedig azokra a csomagokra, amelyek továbbítódnak a rendszeren keresztül (tehát ezeket nem a rendszernek szánták)

Forrás: The Chinese University of Hong Kong (CUHK), CSCI4430 Data Communication and Computer Networks
https://millyz.github.io/ta/csci4430_2017/Lab%20on%20iptables.ppt

iptables – filter tábla

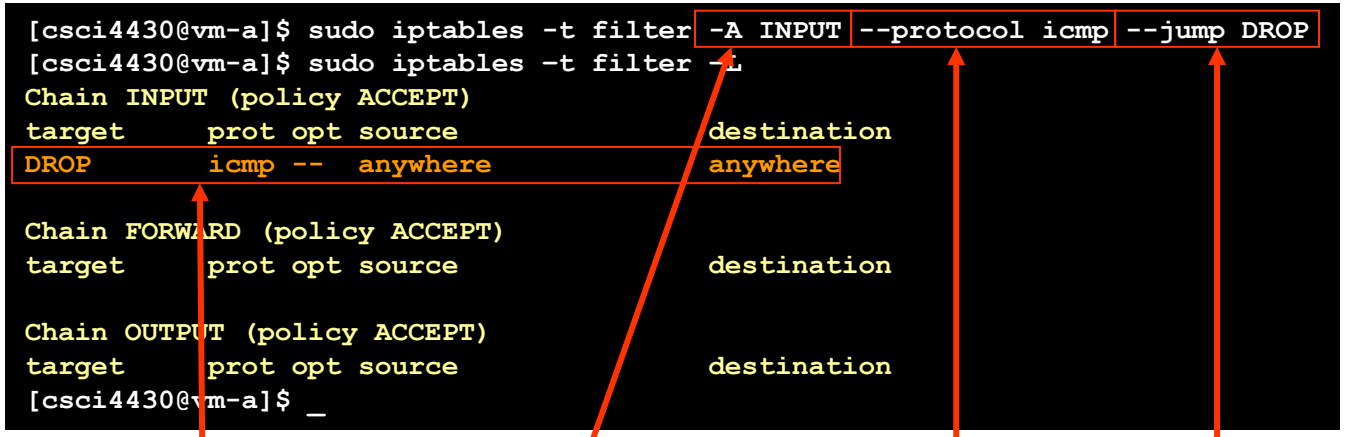


Az INPUT láncban lévő szabály jelentése:

Amikor egy ICMP hasznos teherrel rendelkező csomag áthalad az INPUT-on, DROP-olja ezt a csomagot függetlenül attól, hogy honnan jött, és hova megy.

Forrás: The Chinese University of Hong Kong (CUHK), CSCI4430 Data Communication and Computer Networks
https://millyz.github.io/ta/csci4430_2017/Lab%20on%20iptables.ppt

iptables – filter tábla



Ez a bejegyzés mutatja, hogy egy új szabály sikeresen hozzá lett adva a filter tábla INPUT láncához.

Egy új szabály hozzáadása az INPUT lánchoz.

Az ICMP protokollú csomagok lesznek érdekesek ennél a szabálynál.

Ha egy csomag áthalad az INPUT-on, és egy ICMP csomagról van szó, akkor DROP-olva lesz a csomag.

iptables – nat tábla

- Itt is három lánc van:
- Az OUTPUT lánc itt is van, de kevésbé érdekes
- A PREROUTING lánc még az előtt megváltoztatja a csomagokat mielőtt elérnék az INPUT láncot (pl. porttovábbítást szeretnénk alkalmazni)
- A POSTROUTING lánc pedig azután fogja megváltoztatni a csomagokat miután az OUTPUT láncot elhagyták (pl. a hálózati címfordítás első, egyszerűbb esete)

iptables – nat tábla

- Például szeretnénk a 192.168.1.10 IP címhez és 80-as porthoz jövő csomagot a 192.168.1.20 IP című géphez küldeni a 80-as portjához, akkor az alábbi parancsok (is) kelleni fognak:

```
iptables -t nat -A PREROUTING -p tcp --dport 80 -j DNAT --to-destination 192.168.1.20:80
```

```
iptables -t nat -A POSTROUTING -p tcp -d 192.168.12.20 --dport 80 -j SNAT --to-source 192.168.12.10
```

- (-t kapcsolóval a táblát határozzuk meg, -A PREROUTING : a szabályt a PREROUTING lánc végére szúrja be, -j a csomagcél megadására (SNAT: Source NAT, DNAT: Destination NAT))

iptables

- További példák itt:
- http://linux-training.be/networking/ch14.html
- (a Fájlok között is megvan: Chapter%A014.%A0iptables firewall.pdf)

HÁLÓZATI ELÉRHETŐSÉG/ÚTVONAL

traceroute, ping

Ping a hoszt elérhetőségének ellenőrzésére és a Round Trip Time (RTT) méréséhez

Linuxon

```
lakis@dpdk-pktgen:~$ ping -c 3 berkeley.edu
PING berkeley.edu (35.163.72.93) 56(84) bytes of data.
64 bytes from ec2-35-163-72-93.us-west-2.compute.amazonaws.com (35.163.72.93): icmp_seq=1 ttl=23 time=194 ms
64 bytes from ec2-35-163-72-93.us-west-2.compute.amazonaws.com (35.163.72.93): icmp_seq=2 ttl=23 time=194 ms
64 bytes from ec2-35-163-72-93.us-west-2.compute.amazonaws.com (35.163.72.93): icmp_seq=3 ttl=23 time=193 ms

--- berkeley.edu ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2002ms
rtt min/avg/max/mdev = 193.093/193.937/194.428/0.786 ms
```

Ping a hoszt elérhetőségének ellenőrzésére és a Round Trip Time (RTT) méréséhez

Windowson

```
C:\Users\laki>ping -n 3 berkeley.edu

Pinging berkeley.edu [35.163.72.93] with 32 bytes of data:
Reply from 35.163.72.93: bytes=32 time=200ms TTL=39
Reply from 35.163.72.93: bytes=32 time=201ms TTL=39
Reply from 35.163.72.93: bytes=32 time=200ms TTL=39

Ping statistics for 35.163.72.93:
    Packets: Sent = 3, Received = 3, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 200ms, Maximum = 201ms, Average = 200ms
```

traceroute (linux) – tracert (windows)

Cél a hálózati útvonal meghatározása egy célállomás felé!

Linuxon

```
lakis@dpdk-pktgen:~$ traceroute berkeley.edu
traceroute to berkeley.edu (35.163.72.93), 30 hops max, 60 byte packets
 1  192.168.0.192 (192.168.0.192)  0.292 ms  0.344 ms  0.390 ms
 2  ikoktatok-gate.inf.elte.hu (157.181.167.254)  1.251 ms  1.250 ms  1.265 ms
 3  taurus.centaur-taurus.elte.hu (157.181.126.134)  5.180 ms  5.267 ms  5.325 ms
 4  fw1.firewall.elte.hu (157.181.141.145)  1.271 ms  1.358 ms  1.299 ms
 5  taurus.fw1.fw.backbone.elte.hu (192.153.18.146)  5.626 ms  5.356 ms  5.395 ms
 6  rtr.hbone-elte.elte.hu (157.181.141.9)  2.229 ms  1.245 ms  1.749 ms
 7  tg0-0-0-14.rtr2.vh.hbone.hu (195.111.100.47)  2.377 ms  2.415 ms  2.407 ms
 8  be1.rtr1.vh.hbone.hu (195.111.96.56)  1.945 ms  1.642 ms  1.877 ms
 9  bpt-b4-link..net (80.239.195.56)  1.626 ms  1.581 ms  1.097 ms
10  win-bb2-link.tetelialia.net (62.115.143.116)  196.574 ms win-bb2-link.telia.net (213.155.137.38)  196.993 ms win-bb2-
link.telia.net (213.155.135.222)  180.071 ms
11  ffm-bb4-link.telia.net (62.115.133.79)  199.425 ms  199.232 ms *
12  * * *
13  prs-bb3-link.telia.net (62.115.137.114)  180.494 ms  179.986 ms *
14  sjo-b21-link.telia.net (62.115.119.229)  197.252 ms  197.249 ms  197.264 ms
15  * a100row-ic-300117-sjo-b21.c.telia.net (213.248.87.118)  196.555 ms *
16  nyk-bb4-link.telia.net (62.115.142.222)  180.081 ms 54.240.242.148 (54.240.242.148)  200.986 ms 54.240.242.88
(54.240.242.88)  201.877 ms
17  54.240.242.161 (54.240.242.161)  200.935 ms * *
18  * * *
19  * * *
```

traceroute (linux) – tracert (windows)

Cél a hálózati útvonal meghatározása egy célállomás felé!

Windowson

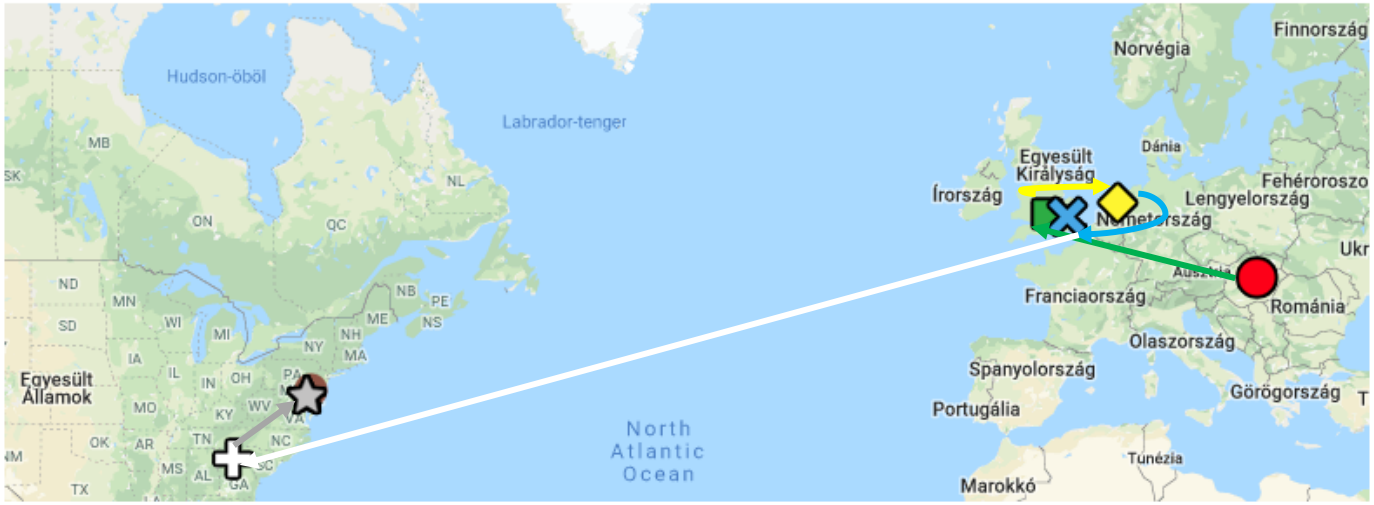
```
tracert -h 50 jhu.edu
Tracing route to jhu.edu [128.220.192.230]
over a maximum of 50 hops:

  1    <1 ms    <1 ms    <1 ms  ikoktatok-gate.inf.elte.hu [157.181.167.254]
  2    <1 ms    <1 ms    <1 ms  leo.leo-centaur.elte.hu [157.181.126.66]
  3    <1 ms    <1 ms    <1 ms  taurus.taurus-leo.elte.hu [157.181.126.45]
  4    <1 ms    <1 ms    <1 ms  fw1.firewall.elte.hu [157.181.141.145]
  5    <1 ms     1 ms    <1 ms  taurus.fw1.fw.backbone.elte.hu [192.153.18.146]
  6     1 ms    <1 ms     1 ms  rtr.hbone-elte.backbone.elte.hu [157.181.141.9]
  7     2 ms     1 ms     2 ms  tg0-0-0-14.rtr2.vh.hbone.hu [195.111.100.47]
  8     1 ms     1 ms     3 ms  be1.rtr1.vh.hbone.hu [195.111.96.56]
  9     1 ms     1 ms     1 ms  hungarnet.mx1.bud.hu.geant.net [62.40.124.101]
 10    14 ms    19 ms    18 ms  62.40.98.47
 11    21 ms    20 ms    21 ms  ae7.mx1.ams.nl.geant.net [62.40.98.186]
 12    28 ms    28 ms    28 ms  ae9-mx1.lon.uk.geant.net [62.40.98.129]
 13   103 ms   103 ms   104 ms  internet2-gw.mx1.lon.uk.geant.net [62.40.124.45]
 14   104 ms   104 ms   104 ms  ae-0.4079.rtsw2.ashb.net.internet2.edu [162.252.70.137]
 15   103 ms   103 ms   104 ms  ae-2.4079.rtsw.ashb.net.internet2.edu [162.252.70.74]
 16   105 ms   104 ms   104 ms  et-11-3-0-1275.clpk-core.maxgigapop.net [206.196.177.2]
 17   107 ms   106 ms   106 ms  hopkins-i2-rtr.maxgigapop.net [206.196.177.70]
 18   106 ms   106 ms   106 ms  128.220.255.73
 19     *        *        *     Request timed out.
 20     *        *        *     Request timed out.
 21     *        *        *     Request timed out.
 22     *        *        *     Request timed out.
 23   106 ms   106 ms   106 ms  boxmigration.jh.edu [128.220.192.230]

Trace complete.
```

traceroute (linux) – tracert (windows)

Cél a hálózati útvonal meghatározása egy célállomás felé!



Legend	
●	hungarnet.mx1.bud.hu.geant.net [62.40.124.101]
■	62.40.98.47
◆	ae7.mx1.ams.nl.geant.net [62.40.98.186]
✖	ae9-mx1.lon.uk.geant.net [62.40.98.129]
✚	ae-0.4079.rtsw2.ashb.net.internet2.edu [162.252.70.137]
★	et-11-3-0-1275.clpk-core.maxgigapop.net [206.196.177.2]
●	boxmigration.jh.edu [128.220.192.230]

www.iplocation.net

+

www.copypastemap.com

PYTHON

subprocess

Subprocess hívások és shell parancsok

Ha nem érdekes az output:

```
import subprocess
subprocess.call(['df', '-h']) # új verziókban run(…)
```

Ha érdekes az output:

```
import subprocess
p = subprocess.Popen(["echo", "hello world"], stdout=subprocess.PIPE)

print(p.communicate()) # eredménye egy tuple (stdout, stderr)

# ('hello world', None)
```

Néha a shell=True argumentum is kell, meg kell nézni a doksit!!!

Hasznos segédletek:
https://docs.python.org/3/library/subprocess.html
https://www.pythonforbeginners.com/os/subprocess-for-system-administrators

subprocess – PIPE kezelés

Elvárt kimenet: dmesg | grep hda

```
from subprocess import PIPE, Popen

p1 = Popen(["dmesg"], stdout=PIPE)
p2 = Popen(["grep", "hda"], stdin=p1.stdout, stdout=PIPE)

p1.stdout.close()  # Allow p1 to receive a SIGPIPE if p2 exits.

output = p2.communicate()[0]
```

subprocess – várakozás a process végére

A process állapotának lekérdezése: poll
(Linux-nál a pingnél „-n” helyett „-c” van)

```
from subprocess import PIPE, Popen
import time

p1 = Popen(["ping", '-n', '20', 'berkeley.edu'], stdout=PIPE)

while p1.poll()==None:
       print(" még fut " )
       time.sleep(1)
```

A process végének megvárása: wait – a communicate is megvárja a végét…

```
p1 = Popen(["ping", '-n', '20', 'berkeley.edu'], stdout=PIPE)

p1.wait() # várakozás a végére
```

Feladat 1

- Készítsünk egy egyszerű python alkalmazást, ahol
  - a ***subprocess*** használatával **50 db.** ping csomagot küldjünk az alábbi weboldalakra:

```
google.com
facebook.com
jhu.edu
```

  - az eredményből vegyük ki az átlagos RTT értékeket
  - az átlagokat írjuk ki a kimenetre weboldalanként, pl.:

```
google.com 11ms
facebook.com 14ms
jhu.edu 168.651ms
```

HÁLÓZATI FORGALOM

tcpdump, wireshark

Hálózati forgalom felvétele/elemzése

tcpdump (Linux):

forgalom figyelő eszköz, a hálózati interfészről jövő csomagokat tudja olvasni

```
lakis@dpdk-switch:~$ sudo tcpdump -i enp8s0
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp8s0, link-type EN10MB (Ethernet), capture size 262144 bytes
09:15:26.376139 IP 192.168.0.102.ssh > oktnb35.inf.elte.hu.55015: Flags [P.], seq 4154664816:4154665024, ack 289117644, win 384, length 208
09:15:26.376403 IP 192.168.0.102.43549 > 192.168.0.192.domain: 52681+ PTR? 35.167.181.157.in-addr.arpa. (45)
09:15:26.376994 IP 192.168.0.192.domain > 192.168.0.102.43549: 52681* 1/0/0 PTR oktnb35.inf.elte.hu. (78)
09:15:26.377100 IP 192.168.0.102.57511 > 192.168.0.192.domain: 64457+ PTR? 102.0.168.192.in-addr.arpa. (44)
09:15:26.377645 IP 192.168.0.192.domain > 192.168.0.102.57511: 64457 NXDomain 0/1/0 (79)
09:15:26.377723 IP 192.168.0.102.49012 > 192.168.0.192.domain: 6981+ PTR? 192.0.168.192.in-addr.arpa. (44)
09:15:26.377851 IP 192.168.0.102.ssh > oktnb35.inf.elte.hu.55015: Flags [P.], seq 208:400, ack 1, win 384, length 192
09:15:26.378180 IP 192.168.0.192.domain > 192.168.0.102.49012: 6981 NXDomain 0/1/0 (79)
09:15:26.378267 IP 192.168.0.102.ssh > oktnb35.inf.elte.hu.55015: Flags [P.], seq 400:976, ack 1, win 384, length 576
09:15:26.378291 IP 192.168.0.102.ssh > oktnb35.inf.elte.hu.55015: Flags [P.], seq 976:1248, ack 1, win 384, length 272
09:15:26.378340 IP 192.168.0.102.ssh > oktnb35.inf.elte.hu.55015: Flags [P.], seq 1248:1600, ack 1, win 384, length 352
09:15:26.378387 IP 192.168.0.102.ssh > oktnb35.inf.elte.hu.55015: Flags [P.], seq 1600:1776, ack 1, win 384, length 176
09:15:26.378440 IP 192.168.0.102.ssh > oktnb35.inf.elte.hu.55015: Flags [P.], seq 1776:1952, ack 1, win 384, length 176
09:15:26.378489 IP 192.168.0.102.ssh > oktnb35.inf.elte.hu.55015: Flags [P.], seq 1952:2128, ack 1, win 384, length 176
09:15:26.378538 IP 192.168.0.102.ssh > oktnb35.inf.elte.hu.55015: Flags [P.], seq 2128:2304, ack 1, win 384, length 176
09:15:26.378587 IP 192.168.0.102.ssh > oktnb35.inf.elte.hu.55015: Flags [P.], seq 2304:2480, ack 1, win 384, length 176
09:15:26.378636 IP 192.168.0.102.ssh > oktnb35.inf.elte.hu.55015: Flags [P.], seq 2480:2656, ack 1, win 384, length 176
09:15:26.378685 IP 192.168.0.102.ssh > oktnb35.inf.elte.hu.55015: Flags [P.], seq 2656:2832, ack 1, win 384, length 176
09:15:26.378734 IP 192.168.0.102.ssh > oktnb35.inf.elte.hu.55015: Flags [P.], seq 2832:3008, ack 1, win 384, length 176
09:15:26.378783 IP 192.168.0.102.ssh > oktnb35.inf.elte.hu.55015: Flags [P.], seq 3008:3184, ack 1, win 384, length 176
09:15:26.378832 IP 192.168.0.102.ssh > oktnb35.inf.elte.hu.55015: Flags [P.], seq 3184:3360, ack 1, win 384, length 176
```

Hálózati forgalom felvétele/elemzése

tcpdump – protokoll filter

```
lakis@dpdk-switch:~$ sudo tcpdump -i enp8s0 icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp8s0, link-type EN10MB (Ethernet), capture size 262144 bytes
09:16:49.470737 IP dpdk-pktgen > 192.168.0.102: ICMP echo request, id 5668, seq 1, length 64
09:16:49.470766 IP 192.168.0.102 > dpdk-pktgen: ICMP echo reply, id 5668, seq 1, length 64
09:16:50.471818 IP dpdk-pktgen > 192.168.0.102: ICMP echo request, id 5668, seq 2, length 64
09:16:50.471834 IP 192.168.0.102 > dpdk-pktgen: ICMP echo reply, id 5668, seq 2, length 64
09:16:51.471716 IP dpdk-pktgen > 192.168.0.102: ICMP echo request, id 5668, seq 3, length 64
09:16:51.471732 IP 192.168.0.102 > dpdk-pktgen: ICMP echo reply, id 5668, seq 3, length 64
09:16:52.471713 IP dpdk-pktgen > 192.168.0.102: ICMP echo request, id 5668, seq 4, length 64
09:16:52.471729 IP 192.168.0.102 > dpdk-pktgen: ICMP echo reply, id 5668, seq 4, length 64
09:16:53.471720 IP dpdk-pktgen > 192.168.0.102: ICMP echo request, id 5668, seq 5, length 64
09:16:53.471736 IP 192.168.0.102 > dpdk-pktgen: ICMP echo reply, id 5668, seq 5, length 64
```

Hálózati forgalom felvétele/elemzése

tcpdump – további filterek

```
lakis@dpdk-switch:~$ sudo tcpdump -i enp8s0 host 192.168.0.101 and port 1111
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp8s0, link-type EN10MB (Ethernet), capture size 262144 bytes
09:20:23.289035 IP dpdk-pktgen.48524 > 192.168.0.102.1111: Flags [S], seq 1544265047, win 29200, options [mss 1460,sackOK,TS val 409718781 ecr 0,nop,wscale 7],
length 0
09:20:23.289067 IP 192.168.0.102.1111 > dpdk-pktgen.48524: Flags [R.], seq 0, ack 1544265048, win 0, length 0
```

Hálózati forgalom felvétele/elemzése

tcpdump – további filterek

```
lakis@dpdk-switch:~$ sudo tcpdump -vvv -A -i enp8s0 host 192.168.0.101 and port 1111
tcpdump: listening on enp8s0, link-type EN10MB (Ethernet), capture size 262144 bytes
09:27:26.361105 IP (tos 0x10, ttl 64, id 14532, offset 0, flags [DF], proto TCP (6), length 60)
    dpdk-pktgen.48546 > 192.168.0.102.1111: Flags [S], cksum 0xelle (correct), seq 3578222049, win 29200, options [mss 1460,sackOK,TS val 409824549 ecr 0,nop,ws
cale 7], length 0
E..<8.@.@......e...f...W.GU.......r............
.mm%........
09:27:26.361137 IP (tos 0x0, ttl 64, id 0, offset 0, flags [DF], proto TCP (6), length 60)
    192.168.0.102.1111 > dpdk-pktgen.48546: Flags [S.], cksum 0x824a (incorrect -> 0xdda8), seq 1341274724, ack 3578222050, win 28960, options [mss 1460,sackOK,
TS val 835209270 ecr 409824549,nop,wscale 7], length 0
E..<..@.@......f...e.W..O.:d.GU...q .J.........
1.H6.mm%....
09:27:26.361250 IP (tos 0x10, ttl 64, id 14533, offset 0, flags [DF], proto TCP (6), length 52)
    dpdk-pktgen.48546 > 192.168.0.102.1111: Flags [.], cksum 0x7cb0 (correct), seq 1, ack 1, win 229, options [nop,nop,TS val 409824549 ecr 835209270], length 0
E..48.@.@......e...f...W.GU.O.:e....|......
.mm%1.H6
09:27:31.152091 IP (tos 0x10, ttl 64, id 14534, offset 0, flags [DF], proto TCP (6), length 59)
    dpdk-pktgen.48546 > 192.168.0.102.1111: Flags [P.], cksum 0x4a14 (correct), seq 1:8, ack 1, win 229, options [nop,nop,TS val 409825747 ecr 835209270], lengt
h 7
E..;8.@.@......e...f...W.GU.O.:e....J......
.mq.1.H6Hello

09:27:31.152109 IP (tos 0x0, ttl 64, id 29267, offset 0, flags [DF], proto TCP (6), length 52)
    192.168.0.102.1111 > dpdk-pktgen.48546: Flags [.], cksum 0x8242 (incorrect -> 0x734f), seq 1, ack 8, win 227, options [nop,nop,TS val 835210468 ecr 40982574
7], length 0
E..4rS@.@.FU...f...e.W..O.:e.GU......B.....
1.L..mq.
09:27:42.531278 IP (tos 0x0, ttl 64, id 29268, offset 0, flags [DF], proto TCP (6), length 55)
    192.168.0.102.1111 > dpdk-pktgen.48546: Flags [P.], cksum 0x8245 (incorrect -> 0x15be), seq 1:4, ack 8, win 227, options [nop,nop,TS val 835213313 ecr 40982
5747], length 3
E..7rT@.@.FQ...f...e.W..O.:e.GU......E.....
1.X..mq.Hi

09:27:42.531425 IP (tos 0x10, ttl 64, id 14535, offset 0, flags [DF], proto TCP (6), length 52)
    dpdk-pktgen.48546 > 192.168.0.102.1111: Flags [.], cksum 0x5d10 (correct), seq 8, ack 4, win 229, options [nop,nop,TS val 409828592 ecr 835213313], length 0
E..48.@.@......e...f...W.GU.O.:h....]......
.m|.1.X.
09:27:50.984854 IP (tos 0x10, ttl 64, id 14536, offset 0, flags [DF], proto TCP (6), length 67)
    dpdk-pktgen.48546 > 192.168.0.102.1111: Flags [P.], cksum 0xf203 (correct), seq 8:23, ack 4, win 229, options [nop,nop,TS val 409830705 ecr 835213313], leng
th 15
E..C8.@.@......e...f...W.GU.O.:h...........
.m.11.X.Hogy vagyunk?

09:27:50.984872 IP (tos 0x0, ttl 64, id 29269, offset 0, flags [DF], proto TCP (6), length 52)
    192.168.0.102.1111 > dpdk-pktgen.48546: Flags [.], cksum 0x8242 (incorrect -> 0x4c81), seq 4, ack 23, win 227, options [nop,nop,TS val 835215426 ecr 4098307
05], length 0
```

Hálózati forgalom felvétele/elemzése

tcpdump – mentés pcap fájlba és fájlból elemzés

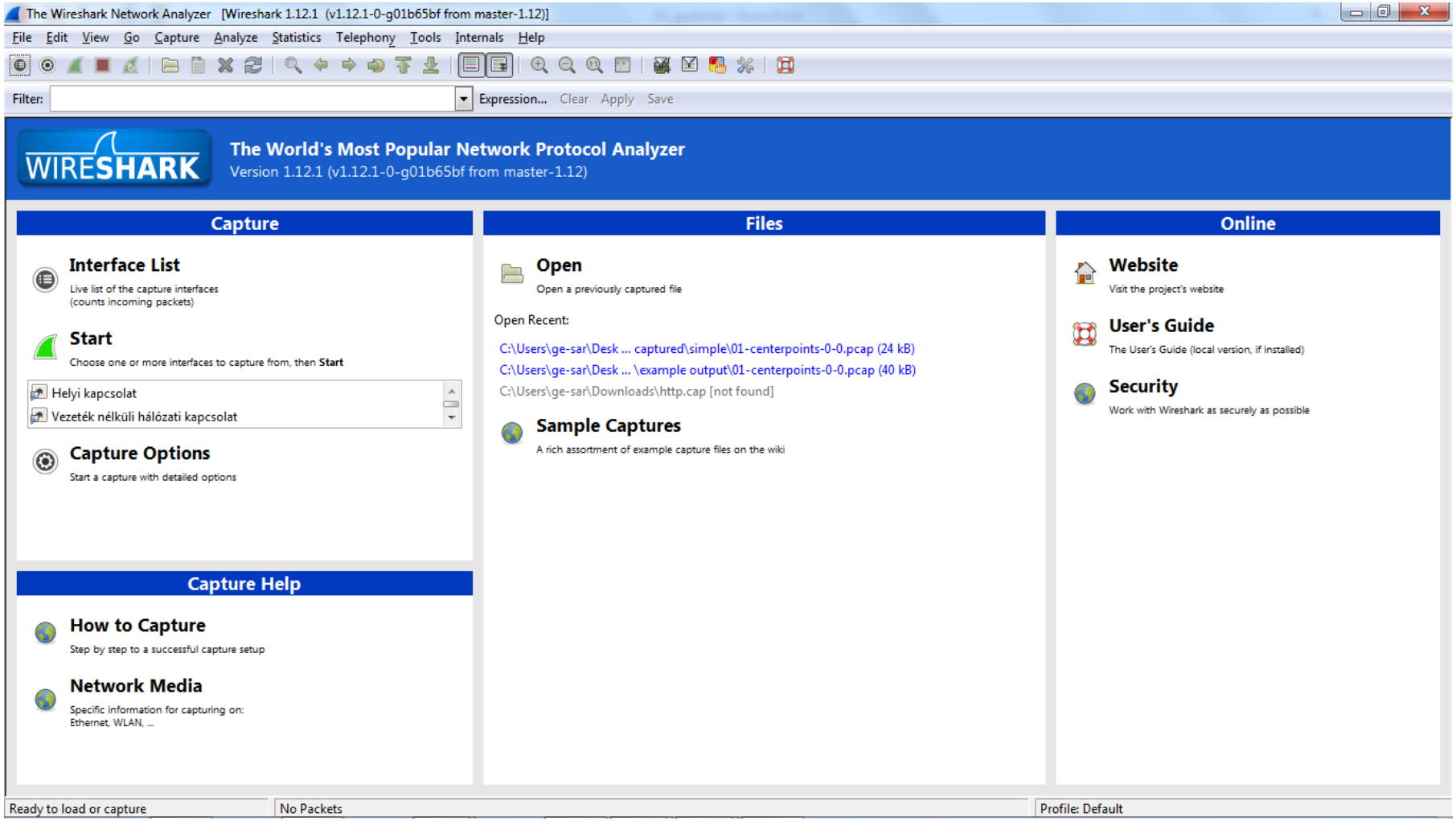
```
lakis@dpdk-switch:~$ sudo tcpdump -w test.pcap -i enp8s0
tcpdump: listening on enp8s0, link-type EN10MB (Ethernet), capture size 262144 bytes
^C4 packets captured
6 packets received by filter
0 packets dropped by kernel
lakis@dpdk-switch:~$ tcpdump -r test.pcap
reading from file test.pcap, link-type EN10MB (Ethernet)
09:31:32.000164 IP 192.168.0.102.ssh > oktnb35.inf.elte.hu.55015: Flags [P.], seq 4154857792:4154857936, ack 289145644, win 384, length 144
09:31:32.060031 IP oktnb35.inf.elte.hu.55015 > 192.168.0.102.ssh: Flags [.], ack 144, win 3542, length 0
09:31:34.354029 IP 192.168.0.192.48309 > 255.255.255.255.7437: UDP, length 173
09:31:37.377992 IP 192.168.0.192.48309 > 255.255.255.255.7437: UDP, length 173
```

Pcap fájl visszajátszására is van lehetőség: tcpreplay

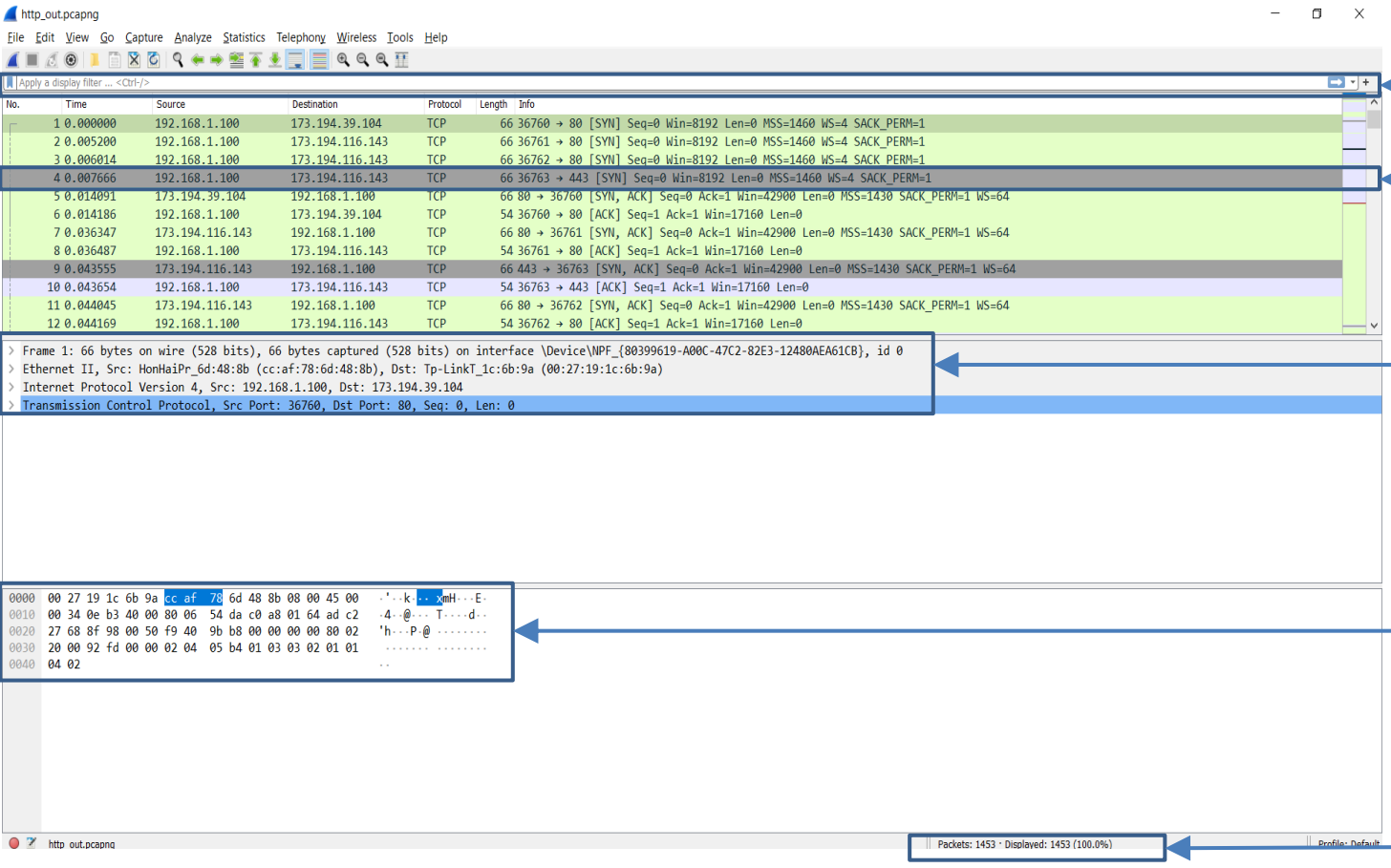
Wireshark

- Forgalomelemző eszköz: korábban rögzített adatok elemzésére szolgál
- Windows-on és Linux-on is elérhető
- www.wireshark.org

Wireshark


The Wireshark Network Analyzer [Wireshark 1.12.1 (v1.12.1-0-g01b65bf from master-1.12)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter: Expression... Clear Apply Save
WIRESHARK
The World's Most Popular Network Protocol Analyzer
Version 1.12.1 (v1.12.1-0-g01b65bf from master-1.12)
Capture
Interface List
Live list of the capture interfaces
(counts incoming packets)
Start
Choose one or more interfaces to capture from, then Start
Helyi kapcsolat
Vezeték nélküli hálózati kapcsolat
Capture Options
Start a capture with detailed options
Capture Help
How to Capture
Step by step to a successful capture setup
Network Media
Specific information for capturing on:
Ethernet, WLAN, ...
Files
Open
Open a previously captured file
Open Recent:
C:\Users\ge-sar\Desk ... captured\simple\01-centerpoints-0-0.pcap (24 kB)
C:\Users\ge-sar\Desk ... \example output\01-centerpoints-0-0.pcap (40 kB)
C:\Users\ge-sar\Downloads\http.cap [not found]
Sample Captures
A rich assortment of example capture files on the wiki
Online
Website
Visit the project's website
User's Guide
The User's Guide (local version, if installed)
Security
Work with Wireshark as securely as possible
Ready to load or capture
No Packets
Profile: Default

Wireshark ablakok



Szűrők definiálására alkalmas input eszközök

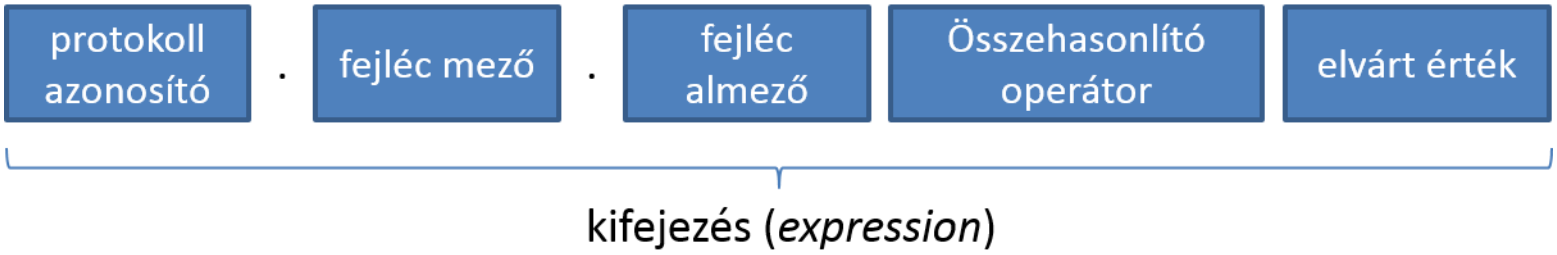
Csomag összefoglaló nézete

Kiválasztott csomag hierarchikus nézete

Kiválasztott csomag bájt-alapú nézete

Szűrés statisztikái

Wireshark szűrők



- Operátorok: or, and, xor, not
- protokollok: ip, tcp, http… (teljes listát lásd → Analyze→Display filter expression…)
- Példa: tcp.flags.ack==1 and tcp.dstport==80 (tcp nyugta flag és fogadó port beállítva)

Wireshark példa

- A **http_out.pcapng** állomány felhasználásával válaszoljuk meg az alábbi kérdéseket:

1. Milyen oldalakat kértek le a szűrés alapján HTTP GET metódussal? Milyen böngészőt használtak hozzá?
2. Hány darab képet érintett a böngészés? (Segítség: *webp.*)
3. Volt-e olyan kérés, amely titkosított kommunikációt takar? (Segítség: *SSL/TLS.*)

VÉGE
KÖSZÖNÖM A FIGYELMET!