

**Fun fact: mininet betűméret növelés: *ctrl+bal* klikk nyomva tart**  
**MAGYAR BILLENTYŰ: setxkbmap hu**

route	Route tábla (-n kapcsolóval nincs címfordítás)
ip route get [szerver ip]	Egy ip-hez vezető interface kiderít
sudo route add default gw [def. gw IP-je]  Sudo ip route add default via [ip] dev [interf]	Default gateway beállítás
Traceroute [host]	Egy hosthoz vezető interfacek útvonala
sudo ip route add 192.168.5.0/24 via 125.5.0.1 dev r1-eth5	Route beállítása. A 192.168.5.0/24 alhálózat a 125.5.0.1 dev r1-eth5 (r1 router) fele menjen
dpctl dump-flows	flyamtabla
3389 (TCP)	VM Rtp csatlakozás
setxkbmap hu	Magyar bill.
Xterm h1	h1 host konzol
h0 arp	h0 host arp táblája

## 1 gyakorlati feladat (4. kiugró)

Az alábbi parancsot kiadva a BME Cloud-ban (Smallville) futtatott HaEpUz VM-en (ügyelve arra, hogy a \$NEPTUN értéket a saját, tényleges neptun-kódodra cseréld le, természetesen \$ nélkül!) indítsd el a saját mininetes hálózatodat. A hálózat indításakor megkapod azon eszközök nevét és IP címét, melyeket a feladatok megoldása során használnod kell majd. A környezetben a [10.0.0.0/8-as](#) hálózat menedzsment hálózatként működik, a feladatok megoldása során ezen a hálózaton keresztül nem irányíthatjuk az adatsík forgalmát! Kiadandó parancs a saját környezet indítására:

```
wget -nv -O- https://sb.tmit.bme.hu/haepuz/tsv1 | sudo sh /dev/stdin $NEPTUN
```

Segítség, ha nem akarjuk mindig a jelszót másolgatni:  
Hozzunk létre a .ssh könyvtárban egy kulcspárt és a publikus kulcsot adjuk hozzá az authorized\_keys fájlhoz. Ez pl. az alábbi parancsokkal tehető meg:

```
cd ~/.ssh
ssh-keygen [3x enter]
cat id_rsa.pub >> authorized_keys
```

### 1. Kéáerdés.

- ✓ Ha a "Host A" gépről pingeljük a "Server" gépet, a "Host A" melyik interfészén történik a kommunikáció? Add meg a kérdéses interfész nevét! 1/1

```
ip route get [szerver ip]
125.0.1.1 via 192.168.4.254 dev h41-eth1 src 192.168.4.41 uid 1000
h41-eth1
```

---

## 2. Kéáérés

- ✓ Add meg az előző kérdés megválaszolásához (a forgalom megfigyeléséhez) használt parancsot! 1/1

```
ip route get [szerver ip]
```

---

## 3. kéáérés

- ✓ A "Host B" gépről nem tudjuk pingelni a "Server" gépet, pedig innen is hasonlóan kéne működni, mint a "Host A" gépről. Milyen parancs volt az, amivel sikerült felderíteni a hibát? (A hiba felderítéséhez persze több parancs használata szükséges, hacsak nem elsőre találjuk el. Itt most arra vagyunk kíváncsiak, amivel meglett a hiba.) 1/1

`route -n` -> JAJAJ nincs default gateway. Ezt be kéne állítani

```
cloud@cloud-40875:~$ route -n
Kernel IP routing table
Destination      Gateway         Genmask         Flags Metric Ref    Use Iface
10.0.0.0          0.0.0.0         255.0.0.0       U        0      0        0 h44-eth0
192.168.4.0       0.0.0.0         255.255.255.0   U        0      0        0 h44-eth1
```

---

#### 4. kéáérés

✓ Add meg az előző hiba javításához használt parancsot/parancsokat! 1/1

Mire kéne állítani a default gw-t????

Amire van állítva A-nak a gw-je arra állítjuk a B-jét is.

```
sudo route add default gw [A host def. gw IP-je]
```

Így már nyugodtan pingelgethetünk. :)

```
sudo route add default gw [A host def. gw IP-je]
```

```
sudo ip route add default via [default gw ip-je] dev  
[interface]
```

---

#### 5. kéáérés

✗ A "Host A" és "Host C" eszközök között milyen interfészekon halad a forgalom? Add meg a parancsot, amivel ezt feltérképezted és add meg a parancs kimenetét! .../1

Host A-ból traceroute-olok a Host C-be

```
traceroute h51 (h51 = C Host)
```

```
r4 (itt megy ki), r1, r5, h51
```

```
cloud@cloud-40875:~$ traceroute h51
traceroute to h51 (192.168.5.51), 30 hops max, 60 byte packets
 1  r4 (192.168.4.254)  5.298 ms  6.828 ms  4.860 ms
 2  r1 (125.4.0.2)    6.541 ms  6.604 ms  6.706 ms
 3  r5 (125.5.0.1)    6.736 ms  6.764 ms  6.793 ms
 4  h51 (192.168.5.51) 19.030 ms 20.584 ms 20.638 ms
```

---

#### 6. Kéáérés

- ✗ A "Host C" egy /24-es alhálózatra csatlakozik. Rajta kívül még több más eszköz is csatlakozik ugyanerre az alhálózatra. A "Host A" gépről próbáld pingelni a "Host C"-nél eggyel nagyobb IP címmel rendelkező eszközt! Melyik IP címről érkezik válasz? Hány interfészen halad át a csomag?

```
ping [Host C + 1]  
125.4.0.2 ről érkezik (r1 router egyik interface-e)
```

```
cloud@cloud-40875:~$ ping 192.168.5.52  
PING 192.168.5.52 (192.168.5.52) 56(84) bytes of data.  
From 125.4.0.2 icmp_seq=1 Destination Net Unreachable  
From 125.4.0.2 icmp_seq=2 Destination Net Unreachable  
From 125.4.0.2 icmp_seq=3 Destination Net Unreachable  
From 125.4.0.2 icmp_seq=4 Destination Net Unreachable
```

```
traceroute 192.168.5.52 (=125.4.0.2)
```

```
cloud@cloud-40875:~$ traceroute 192.168.5.52  
traceroute to 192.168.5.52 (192.168.5.52), 30 hops max, 60 byte packets  
 1  r4 (192.168.4.254)  9.262 ms  9.349 ms  9.366 ms  
 2  r1 (125.4.0.2)  9.376 ms !N  8.981 ms !N *
```

2 interfacen

---

## 7. Kaerdís

- ✗ Jelentkezz be arra az eszközre, amelyikhez az utolsó azonosított interfész tartozik! (A bejelentkezéshez használható IP cím a hálózat indításakor szintén listázásra került.) Mi okozza a hibát? Add meg a felderítéshez használt egyetlen parancsot, annak kimenetéről a hibát okozó sort, valamint egy egy mondatos magyarázatot arról, hogy ez mit jelent!

Ez az interface az r1 routerhez tartozik.  
ssh 10.1.1.1

Megnézzük a route tábláját  
route -n

```
Host C: ns1: 192.168.5.51  
Server: s1: 125.0.1.1  
Routers:  
router 'r1': ssh 10.1.1.1  
Interfaces:  
125.0.1.254;  
125.2.0.2;  
125.3.0.2;  
125.4.0.2;  
125.5.0.2;
```

```
cloud@cloud-40875:~$ route
Kernel IP routing table
Destination    Gateway         Genmask         Flags Metric Ref    Use Iface
10.0.0.0        0.0.0.0         255.0.0.0       U        0      0        0 r1-eth0
125.0.1.0       0.0.0.0         255.255.255.0   U        0      0        0 r1-eth1
125.2.0.0       0.0.0.0         255.255.255.252 U        0      0        0 r1-eth2
125.3.0.0       0.0.0.0         255.255.255.252 U        0      0        0 r1-eth3
125.4.0.0       0.0.0.0         255.255.255.252 U        0      0        0 r1-eth4
125.5.0.0       0.0.0.0         255.255.255.252 U        0      0        0 r1-eth5
192.168.2.0     r2              255.255.255.0   UG       0      0        0 r1-eth2
192.168.3.0     r3              255.255.255.0   UG       0      0        0 r1-eth3
192.168.4.0     r4              255.255.255.0   UG       0      0        0 r1-eth4
h51             r5              255.255.255.255 UGH      0      0        0 r1-eth5
```

Látjuk h nincs default gateway a 192.168.5.0 hálózathoz

```
sudo ip route add 192.168.5.0/24 via 125.5.0.1 dev r1-eth5
```

Az a hiba, hogy nincs route a cél felé.

Route -n

H51- es sor mert van host csak nincs hozzá teljes subnet

## 8. Köerdícs

✗ Adjál olyan megoldást a problémára, ami a kérdéses alhálózat minden elemére javítja a hibát! Másold be a használt parancsot/parancsokat!

```
sudo ip route add 192.168.5.0/24 via 125.5.0.1 dev r1-eth5 ->
```

ezzel lesz teljes elérhetőség

Látható h milyen szép a végeredmény. Lehet pin gelni.

```
cloud@cloud-40875:~$ route
Kernel IP routing table
Destination    Gateway         Genmask         Flags Metric Ref    Use Iface
10.0.0.0        0.0.0.0         255.0.0.0       U        0      0        0 r1-eth0
125.0.1.0       0.0.0.0         255.255.255.0   U        0      0        0 r1-eth1
125.2.0.0       0.0.0.0         255.255.255.252 U        0      0        0 r1-eth2
125.3.0.0       0.0.0.0         255.255.255.252 U        0      0        0 r1-eth3
125.4.0.0       0.0.0.0         255.255.255.252 U        0      0        0 r1-eth4
125.5.0.0       0.0.0.0         255.255.255.252 U        0      0        0 r1-eth5
192.168.2.0     r2              255.255.255.0   UG       0      0        0 r1-eth2
192.168.3.0     r3              255.255.255.0   UG       0      0        0 r1-eth3
192.168.4.0     r4              255.255.255.0   UG       0      0        0 r1-eth4
192.168.5.0     r5              255.255.255.0   UG       0      0        0 r1-eth5
h51             r5              255.255.255.255 UGH      0      0        0 r1-eth5
```

### 3. Vizsga feladat (6. Gyak kiugró)

Az alábbi parancsot kiadva a BME Cloud-ban (Smallville) futtatott HaEpUz VM-en (ügyelve arra, hogy a \$NEPTUN értéket a saját, tényleges neptun-kódodra cseréld le, természetesen \$ nélkül!) indítsd el a saját környezetet. Egy felugró ablakban elindul egy mininetes hálózatemuláció és egy másik ablakban pedig egy pox controller. Kiadandó parancs a saját környezet indítására:

```
wget -nv -O- https://sb.tmit.bme.hu/haepuz/errorv1 | sh /dev/stdin $NEPTUN
```

#### 1 Kearedes

- ✗ Próbáld ki, hogy a h1 hosztról nem lehet pingelni a.../2  
h10-es hosztot. Azért nem, mert az egyik switch  
egyik folyamtábla-bejegyzésében szándékosan el  
van írva az output port értéke. A feladat  
megkeresni, hogy az elrontott folyamtábla-  
bejegyzéshez milyen cookie érték tartozik. A  
megoldás mezőbe ezt a cookie értéket kell  
hexadecimális formában beírni (pl. 0xa4).

```
h1 ping h10
```

Net paranccsal megnéztük h a h10 mivel van kapcsolatban → S10

```
mininet> net
h1 h1-eth0:s1-eth1
h2 h2-eth0:s2-eth1
h3 h3-eth0:s3-eth1
h4 h4-eth0:s4-eth1
h5 h5-eth0:s5-eth1
h6 h6-eth0:s6-eth1
h7 h7-eth0:s7-eth1
h8 h8-eth0:s8-eth1
h9 h9-eth0:s9-eth1
h10 h10-eth0:s10-eth1
s1 lo: s1-eth1:h1-eth0 s1-eth2:s2-eth2
s2 lo: s2-eth1:h2-eth0 s2-eth2:s1-eth2 s2-eth3:s3-eth2
s3 lo: s3-eth1:h3-eth0 s3-eth2:s2-eth3 s3-eth3:s4-eth2
s4 lo: s4-eth1:h4-eth0 s4-eth2:s3-eth3 s4-eth3:s5-eth2
s5 lo: s5-eth1:h5-eth0 s5-eth2:s4-eth3 s5-eth3:s6-eth2
s6 lo: s6-eth1:h6-eth0 s6-eth2:s5-eth3 s6-eth3:s7-eth2
s7 lo: s7-eth1:h7-eth0 s7-eth2:s6-eth3 s7-eth3:s8-eth2
s8 lo: s8-eth1:h8-eth0 s8-eth2:s7-eth3 s8-eth3:s9-eth2
s9 lo: s9-eth1:h9-eth0 s9-eth2:s8-eth3 s9-eth3:s10-eth2
s10 lo: s10-eth1:h10-eth0 s10-eth2:s9-eth3
c0
```

`dptcl dump-flows` → lekértük a folyam tábla ábrákat

Látszódik hogy S4 nél már nincs vissza csomag (csak 1x van a 16 csomag (azért 16 mert 16 ping csomagot küldtünk)) Ez azt jelenti h S5 ben van vmi elkúrva.

```
cookie=0x18, duration=1072.983s, table=0, n_packets=0, n_bytes=0, ip,d_l_src=00:00:00:00:00:0a,d_l_dst=00:00:00:00:00:0a,nw_src=10.0.0.10,nw_dst=10.0.0.10 actions=output:"s3-eth1"
*** s4
cookie=0x0, duration=1072.980s, table=0, n_packets=4, n_bytes=168, priority=1,arp actions=FL000
cookie=0x1e, duration=1072.975s, table=0, n_packets=0, n_bytes=0, ip,d_l_src=00:00:00:00:00:01,d_l_dst=00:00:00:00:00:0a,nw_src=10.0.0.10 actions=output:"s4-eth1"
cookie=0x11, duration=1072.975s, table=0, n_packets=0, n_bytes=0, ip,in_port="s4-eth3",d_l_src=00:00:00:00:00:0a,d_l_dst=00:00:00:00:00:0a actions=output:"s4-eth3"
cookie=0xbd, duration=1072.975s, table=0, n_packets=0, n_bytes=0, ip,in_port="s4-eth3",d_l_src=00:00:00:00:00:01 actions=output:"s4-eth3"
cookie=0x5b, duration=1072.975s, table=0, n_packets=16, n_bytes=1568, ip,in_port="s4-eth2",nw_dst=10.0.0.10 actions=output:"s4-eth3"
cookie=0x17, duration=1072.975s, table=0, n_packets=0, n_bytes=0, ip,in_port="s4-eth2",nw_src=10.0.0.10 actions=output:"s4-eth1"
cookie=0xf, duration=1072.975s, table=0, n_packets=0, n_bytes=0, ip,d_l_src=00:00:00:00:00:0a,nw_src=10.0.0.10 actions=output:"s4-eth2"
cookie=0x0, duration=1072.975s, table=0, n_packets=4, n_bytes=168, priority=1,arp actions=FL000
*** s5
cookie=0x53, duration=1072.993s, table=0, n_packets=0, n_bytes=0, ip,d_l_dst=00:00:00:00:00:01,nw_src=10.0.0.1,nw_dst=10.0.0.10 actions=output:"s5-eth1"
cookie=0x27, duration=1072.993s, table=0, n_packets=16, n_bytes=1568, ip,in_port="s5-eth3",d_l_src=00:00:00:00:00:0a,nw_src=10.0.0.10 actions=output:"s5-eth1"
cookie=0x15, duration=1072.993s, table=0, n_packets=16, n_bytes=1568, ip,in_port="s5-eth2",d_l_src=00:00:00:00:00:01,nw_dst=10.0.0.10 actions=output:"s5-eth3"
cookie=0x5e, duration=1072.993s, table=0, n_packets=0, n_bytes=0, ip,in_port="s5-eth3",d_l_src=00:00:00:00:00:01 actions=output:"s5-eth2"
cookie=0x46, duration=1072.993s, table=0, n_packets=0, n_bytes=0, ip,d_l_src=00:00:00:00:00:0a,nw_src=10.0.0.1 actions=output:"s5-eth2"
cookie=0x42, duration=1072.993s, table=0, n_packets=0, n_bytes=0, ip,in_port="s5-eth2",d_l_src=00:00:00:00:00:0a,d_l_dst=00:00:00:00:00:01 actions=output:"s5-eth2"
cookie=0x0, duration=1072.993s, table=0, n_packets=4, n_bytes=168, priority=1,arp actions=FL000
*** s6
cookie=0x2a, duration=1073.068s, table=0, n_packets=0, n_bytes=0, ip,d_l_dst=00:00:00:00:00:01,nw_src=10.0.0.10,nw_dst=10.0.0.10 actions=output:"s6-eth2"
cookie=0x28, duration=1073.068s, table=0, n_packets=0, n_bytes=0, ip,d_l_src=00:00:00:00:00:0a,d_l_dst=00:00:00:00:00:0a actions=output:"s6-eth2"
cookie=0x2b, duration=1073.068s, table=0, n_packets=0, n_bytes=0, ip,d_l_dst=00:00:00:00:00:01,nw_src=10.0.0.1 actions=output:"s6-eth3"
cookie=0x1f, duration=1073.068s, table=0, n_packets=16, n_bytes=1568, ip,d_l_dst=00:00:00:00:00:01,nw_src=10.0.0.10 actions=output:"s6-eth2"
cookie=0xc, duration=1073.068s, table=0, n_packets=16, n_bytes=1568, ip,d_l_dst=00:00:00:00:00:0a,nw_src=10.0.0.1 actions=output:"s6-eth3"
cookie=0x2, duration=1073.068s, table=0, n_packets=0, n_bytes=0, ip,in_port="s6-eth2",d_l_dst=00:00:00:00:00:01,nw_dst=10.0.0.10 actions=output:"s6-eth2"
cookie=0x0, duration=1073.059s, table=0, n_packets=4, n_bytes=168, priority=1,arp actions=FL000
*** s7
cookie=0x3f, duration=1073.061s, table=0, n_packets=16, n_bytes=1568, ip,in_port="s7-eth2",nw_src=10.0.0.1,nw_dst=10.0.0.10 actions=output:"s7-eth3"
cookie=0x20, duration=1073.061s, table=0, n_packets=0, n_bytes=0, ip,nw_src=10.0.0.10,nw_dst=10.0.0.10 actions=output:"s7-eth1"
cookie=0x63, duration=1073.061s, table=0, n_packets=16, n_bytes=1568, ip,d_l_src=00:00:00:00:00:0a,d_l_dst=00:00:00:00:00:01,nw_dst=10.0.0.1 actions=output:"s7-eth3"
cookie=0x47, duration=1073.061s, table=0, n_packets=0, n_bytes=0, ip,in_port="s7-eth3",d_l_src=00:00:00:00:00:0a,nw_dst=10.0.0.10 actions=output:"s7-eth3"
cookie=0x37, duration=1073.061s, table=0, n_packets=0, n_bytes=0, ip,in_port="s7-eth2",d_l_src=00:00:00:00:00:01,nw_dst=10.0.0.1 actions=output:"s7-eth2"
```

S5-ben az a szar amelyiknek a source ip-je a h10 ipje (h10 -> h1)

(destination h1 vagy source h10)

Ezalapján `0x27`

Az alábbi parancsot kiadva a BME Cloud-ban (Smallville) vagy lokálisan a saját gépen futtatott HaEpUz VM-en (ügyelve arra, hogy a \$NEPTUN értéket a saját, tényleges neptun-kódodra cseréld le, természetesen \$ nélkül!) indítsd el a saját környezetet. Egy felugró ablakban elindul egy mininetes hálózatemuláció és egy másik ablakban pedig egy pox kontroller. Kiadandó parancs a saját környezet indítására:

```
wget -nv -O- https://sb.tmit.bme.hu/haepuz/star | sh /dev/stdin $NEPTUN
```

## 2 Kearedes

✗ A pox kontroller és a mininetes hálózat elindítása .../1  
után a h1 hosztról sikeresen lehet pingelni a h2  
hosztot. Azonban a ping kérésekre nem a h2  
hoszt válaszol, mert a kontroller eltéríti a ping  
forgalmat egy másik hoszthoz, valamint a  
forgalomról egy másolatot is kiküldet a  
kapcsolóval egy nem létező porton. Mi annak a  
hosztnak a neve, ami a h2 felé küldött ping  
kérésekre válaszol (pl: h73)?

h1 ping h2 → nem onnan érkezik a válasz, indítunk egy wiresharkot, és egy  
pinget és a loopback interfészen megvizsgáljuk a ping üzeneteket (openflow\_v1  
szűréssel)



```

--- 10.0.0.2 ping statistics ---
9 packets transmitted, 9 received, 0% packet loss, time 801
rtt min/avg/max/mdev = 57.169/60.543/63.584/2.046 ms
mininet> h1 ping h2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=68.6 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=71.1 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=70.6 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=71.5 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=123 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=99.5 ms
64 bytes from 10.0.0.2: icmp_seq=7 ttl=64 time=56.2 ms
64 bytes from 10.0.0.2: icmp_seq=8 ttl=64 time=57.1 ms
64 bytes from 10.0.0.2: icmp_seq=9 ttl=64 time=57.7 ms
64 bytes from 10.0.0.2: icmp_seq=10 ttl=64 time=60.0 ms
64 bytes from 10.0.0.2: icmp_seq=11 ttl=64 time=71.2 ms
64 bytes from 10.0.0.2: icmp_seq=12 ttl=64 time=71.4 ms
64 bytes from 10.0.0.2: icmp_seq=13 ttl=64 time=74.5 ms
64 bytes from 10.0.0.2: icmp_seq=14 ttl=64 time=81.3 ms
64 bytes from 10.0.0.2: icmp_seq=15 ttl=64 time=80.1 ms
64 bytes from 10.0.0.2: icmp_seq=16 ttl=64 time=80.3 ms
64 bytes from 10.0.0.2: icmp_seq=17 ttl=64 time=80.0 ms
64 bytes from 10.0.0.2: icmp_seq=18 ttl=64 time=79.0 ms
64 bytes from 10.0.0.2: icmp_seq=19 ttl=64 time=78.2 ms
64 bytes from 10.0.0.2: icmp_seq=20 ttl=64 time=77.8 ms
64 bytes from 10.0.0.2: icmp_seq=21 ttl=64 time=77.1 ms
64 bytes from 10.0.0.2: icmp_seq=22 ttl=64 time=75.8 ms
64 bytes from 10.0.0.2: icmp_seq=23 ttl=64 time=74.4 ms
64 bytes from 10.0.0.2: icmp_seq=24 ttl=64 time=75.7 ms
64 bytes from 10.0.0.2: icmp_seq=25 ttl=64 time=74.3 ms
64 bytes from 10.0.0.2: icmp_seq=26 ttl=64 time=73.2 ms
64 bytes from 10.0.0.2: icmp_seq=27 ttl=64 time=72.2 ms
64 bytes from 10.0.0.2: icmp_seq=28 ttl=64 time=71.9 ms
64 bytes from 10.0.0.2: icmp_seq=29 ttl=64 time=71.4 ms
64 bytes from 10.0.0.2: icmp_seq=30 ttl=64 time=58.6 ms
64 bytes from 10.0.0.2: icmp_seq=31 ttl=64 time=56.9 ms
64 bytes from 10.0.0.2: icmp_seq=32 ttl=64 time=56.7 ms
64 bytes from 10.0.0.2: icmp_seq=33 ttl=64 time=56.6 ms
64 bytes from 10.0.0.2: icmp_seq=34 ttl=64 time=55.1 ms
64 bytes from 10.0.0.2: icmp_seq=35 ttl=64 time=58.7 ms
64 bytes from 10.0.0.2: icmp_seq=36 ttl=64 time=57.7 ms
64 bytes from 10.0.0.2: icmp_seq=37 ttl=64 time=56.2 ms

```

The image shows a Wireshark packet capture on interface 'lo'. The packet list displays a series of ICMP Echo (ping) requests and replies. The packet details pane for packet 11 shows the structure of an ICMP Echo (ping) request, including the type (0), code (0), checksum (0x0721), identifier (20491), and sequence number (1). The packet length is 84 bytes.

**/usr/sbin/tcpdump -i h2-eth0**

És ha xterm h24 en a tcpdump-al újra elkapod a ping csomagokat (újra indítasz pinget) akkor látszik h itt megérkeznek tehát a kérdésre a válasz a **h24**

### 3 Kearedes

✗ Mi annak a nem létező portnak a száma (pl: 211), ahova a controller a másolatot küldeti a kapcsolóval?

The image shows a Wireshark packet capture on interface 'lo'. The packet list displays a series of ICMP Echo (ping) requests and replies. The packet details pane for packet 13 shows the structure of an ICMP Echo (ping) request, including the type (0), code (0), checksum (0x0721), identifier (20491), and sequence number (1). The packet length is 84 bytes.

Második out csomag ahol nem 24 az outport (119 > 50, 50 hoszt van összesen)

**119**

---

4 Kearedes

✗ Hány darab folyambejegyzés található a kapcsoló 0/1  
folyam táblájában?

Fő mininet kózzolban: **dpctl dump-flows**

```
mininet> dpctl dump-flows
*** s1 -----
cookie=0x0, duration=1417.264s, table=0, n_packets=0, n_bytes=0, priority=40000,arp actions=FLOOD
cookie=0x0, duration=1417.265s, table=0, n_packets=24, n_bytes=2352, actions=CONTROLLER:65535
```

2 db

---

## Balancer feladat

Az alábbi parancsot kiadva a BME Cloud-ban (Smallville) futtatott HaEpUz VM-en (ügyelve arra, hogy a \$NEPTUN értéket a saját, tényleges neptun-kódodra cseréld le, természetesen \$ nélkül!) indítsd el a saját környezetet. Egy felugró ablakban elindul egy mininetes hálózatemuláció és egy másik ablakban pedig egy pox kontroller. A rendszerben egy speciális terheléelosztó működik. Kiadandó parancs a saját környezet indítására:

```
wget -nv -O- https://sb.tmit.bme.hu/haepuz/balancer | sh /dev/stdin $NEPTUN
```

- ✓ A hálózat elindítása után hány darab folyambejegyzés található összesen 1/1 a kapcsolók folyamatblájában (mindet kapcsoló minden bejegyzését összeadva)? (1 pont)

10



dpctl dump-flows

```
mininet> dpctl dump-flows
*** s1
cookie=0x0, duration=16.374s, table=0, n_packets=0, n_bytes=0, priority=20000,icmp,nw_dst=10.0.0.10 actions=output:"s1-eth2"
cookie=0x0, duration=16.373s, table=0, n_packets=0, n_bytes=0, priority=20000,udp,nw_dst=10.0.0.10 actions=output:"s1-eth2"
cookie=0x0, duration=16.373s, table=0, n_packets=0, n_bytes=0, priority=20000,tcp,nw_dst=10.0.0.10,tp_dst=22 actions=output:"s1-eth3"
cookie=0x0, duration=16.373s, table=0, n_packets=0, n_bytes=0, priority=20000,tcp,nw_dst=10.0.0.10,tp_dst=80 actions=output:"s1-eth2"
cookie=0x0, duration=16.374s, table=0, n_packets=0, n_bytes=0, priority=1,d1_dst=00:00:00:00:00:01 actions=output:"s1-eth1"
*** s2
cookie=0x0, duration=16.381s, table=0, n_packets=0, n_bytes=0, priority=20000,icmp,nw_dst=10.0.0.10 actions=output:"s2-eth5"
cookie=0x0, duration=16.381s, table=0, n_packets=0, n_bytes=0, priority=20000,udp,nw_dst=10.0.0.10 actions=output:"s2-eth4",output:"s2-eth5"
cookie=0x0, duration=16.382s, table=0, n_packets=0, n_bytes=0, priority=1,d1_dst=00:00:00:00:00:01 actions=output:"s2-eth6"
*** s3
cookie=0x0, duration=16.391s, table=0, n_packets=0, n_bytes=0, priority=20000,tcp,nw_dst=10.0.0.10,tp_dst=22 actions=output:"s3-eth4"
cookie=0x0, duration=16.392s, table=0, n_packets=0, n_bytes=0, priority=1,d1_dst=00:00:00:00:00:01 actions=output:"s3-eth6"
```

10 db

- ✓ A h0 hosztról a 10.0.0.10 címre indított ping hatására hány darab ARP request - response üzenetváltás történik? 1/1 (1 pont)

0 (mivel ismeri)



Helyes válasz

0

h0 arp

```
mininet> h0 arp
Address          HWtype  HWaddress      Flags Mask    Iface
10.0.0.10        ether   00:00:00:00:00:0a  CM           h0-eth0
mininet>
```

✗ A h0 hosztról a 10.0.0.10 címre indított ping-re (ICMP echo request) melyik hoszt fog válaszolni (pl: h25)? (1 pont)

1/0

h5, h7, h8

✗

Szerintem h6

Ha megnézzük a folyamtáblát akkor ki lehet olvasni, hogy merre ment az icmp és az s1 és s3 swicheken halad át. Az s3 swichen a két bejegyzés közül annak néztem meg az output-ját amelyiknek a dest ip a 10.0.0.10. Ez az output a h6-eth1-el volt összekötve, amely a h6 interfésze. Szóval szerintem ezért ő válaszol. Tesztként megnéztem tcpdump-pal a h6-ot, és megjelentek a request/reply párok. De ez csak tipp, lehet kibanagy hülyeség...

h0 ping 10.0.0.10

```
mininet> h0 ping 10.0.0.10
PING 10.0.0.10 (10.0.0.10) 56(84) bytes of data.
64 bytes from 10.0.0.10: icmp_seq=1 ttl=64 time=62.7 ms
```

```

Address      HWtype  HWaddress  Flags Mask  Iface
10.0.0.10    ether    00:00:00:00:00:0a  CM          h0-eth0
mininet> net
h0 h0-eth0:s1-eth1
h1 h1-eth0:s2-eth1
h2 h2-eth0:s2-eth2
h3 h3-eth0:s2-eth3
h4 h4-eth0:s2-eth4
h5 h5-eth0:s2-eth5
h6 h6-eth0:s3-eth1
h7 h7-eth0:s3-eth2
h8 h8-eth0:s3-eth3
h9 h9-eth0:s3-eth4
h10 h10-eth0:s3-eth5
s1 lo: s1-eth1:h0-eth0 s1-eth2:s2-eth6 s1-eth3:s3-eth6
s2 lo: s2-eth1:h1-eth0 s2-eth2:h2-eth0 s2-eth3:h3-eth0 s2-eth4:h4-eth0 s2-eth5:h5-eth0 s2-eth6:s1-eth2 s2-eth7:s3-eth7
s3 lo: s3-eth1:h6-eth0 s3-eth2:h7-eth0 s3-eth3:h8-eth0 s3-eth4:h9-eth0 s3-eth5:h10-eth0 s3-eth6:s1-eth3 s3-eth7:s2-eth7
c0
mininet> h0 ping 10.0.0.10
PING 10.0.0.10 (10.0.0.10) 56(84) bytes of data.
64 bytes from 10.0.0.10: icmp_seq=1 ttl=64 time=62.7 ms
64 bytes from 10.0.0.10: icmp_seq=2 ttl=64 time=60.6 ms
64 bytes from 10.0.0.10: icmp_seq=3 ttl=64 time=60.6 ms
64 bytes from 10.0.0.10: icmp_seq=4 ttl=64 time=60.7 ms
64 bytes from 10.0.0.10: icmp_seq=5 ttl=64 time=60.8 ms
64 bytes from 10.0.0.10: icmp_seq=6 ttl=64 time=60.5 ms
64 bytes from 10.0.0.10: icmp_seq=7 ttl=64 time=60.6 ms
64 bytes from 10.0.0.10: icmp_seq=8 ttl=64 time=60.7 ms
^C
--- 10.0.0.10 ping statistics ---
8 packets transmitted, 8 received, 0% packet loss, time 7013ms
rtt min/avg/max/mdev = 60.505/60.906/62.701/0.682 ms
mininet> ifconfig h6
*** Unknown command: ifconfig h6
mininet> dpctl dump-flows
*** s1
cookie=0x0, duration=671.307s, table=0, n_packets=8, n_bytes=784, priority=20000,icmp,nw_dst=10.0.0.10 actions=output:"s1-eth2"
cookie=0x0, duration=671.307s, table=0, n_packets=0, n_bytes=0, priority=20000,udp,nw_dst=10.0.0.10 actions=output:"s1-eth2"
cookie=0x0, duration=671.306s, table=0, n_packets=0, n_bytes=0, priority=20000,tcp,nw_dst=10.0.0.10,tp_dst=22 actions=output:"s1-eth3"
cookie=0x0, duration=671.306s, table=0, n_packets=0, n_bytes=0, priority=20000,tcp,nw_dst=10.0.0.10,tp_dst=80 actions=output:"s1-eth2"
cookie=0x0, duration=671.307s, table=0, n_packets=8, n_bytes=784, priority=1,dl_dst=00:00:00:00:00:01 actions=output:"s1-eth1"
*** s2
cookie=0x0, duration=671.317s, table=0, n_packets=8, n_bytes=784, priority=20000,icmp,nw_dst=10.0.0.10 actions=output:"s2-eth5"
cookie=0x0, duration=671.316s, table=0, n_packets=0, n_bytes=0, priority=20000,udp,nw_dst=10.0.0.10 actions=output:"s2-eth4",output:"s2-eth5"
cookie=0x0, duration=671.318s, table=0, n_packets=8, n_bytes=784, priority=1,dl_dst=00:00:00:00:00:01 actions=output:"s2-eth6"
*** s3
cookie=0x0, duration=671.332s, table=0, n_packets=0, n_bytes=0, priority=20000,tcp,nw_dst=10.0.0.10,tp_dst=22 actions=output:"s3-eth4"
cookie=0x0, duration=671.333s, table=0, n_packets=0, n_bytes=0, priority=1,dl_dst=00:00:00:00:00:01 actions=output:"s3-eth6"
mininet>

```

(NAGYON időigényes verzió: minegyikre nyitsz egy terminált sudo tcpdumpra és nézed hova érkezik a ping)

dpctl dump-flows

**h5**

✓ A h0 hosztról a 10.0.0.10 címre indított UDP forgalmat melyik hoszt fogadja? (2 pont)

2/2

h7, h8



Szerintem ez is meg van. Annyi, hogy ugye itt más eredmények jönnek ki a neptun kód miatt, mert nekem pl a fenti folyamatáblában nem ugyanazok az outputok vannak. Szóval szerintem itt is annyi az egész hogy a folyamatáblából ki kell olvasni, hoogy a h0 az s1-el van összekötve, az s1-nél van egy olyan rekord, hogy udp a dest

10.0.0.10 fele, ez át van irányítva az s1-eth2-es interfészre (ez most a fenti folyamatábra alapján),

```
mininet> dpctl dump-flows
*** s1 ***
cookie=0x0, duration=1316.800s, table=0, n_packets=32, n_bytes=3136, priority=20000,icmp,nw_dst=10.0.0.10 actions=output:"s1-eth2"
cookie=0x0, duration=1316.800s, table=0, n_packets=0, n_bytes=0, priority=20000,udp,nw_dst=10.0.0.10 actions=output:"s1-eth2"
cookie=0x0, duration=1316.799s, table=0, n_packets=0, n_bytes=0, priority=20000,tcp,nw_dst=10.0.0.10,tp_dst=22 actions=output:"s1-eth3"
cookie=0x0, duration=1316.799s, table=0, n_packets=0, n_bytes=0, priority=20000,tcp,nw_dst=10.0.0.10,tp_dst=80 actions=output:"s1-eth2"
cookie=0x0, duration=1316.800s, table=0, n_packets=32, n_bytes=3136, priority=1,d1_dst=00:00:00:00:00:01 actions=output:"s1-eth1"
*** s2 ***
cookie=0x0, duration=1316.811s, table=0, n_packets=32, n_bytes=3136, priority=20000,icmp,nw_dst=10.0.0.10 actions=output:"s2-eth5"
cookie=0x0, duration=1316.810s, table=0, n_packets=0, n_bytes=0, priority=20000,udp,nw_dst=10.0.0.10 actions=output:"s2-eth4",output:"s2-eth5"
cookie=0x0, duration=1316.812s, table=0, n_packets=32, n_bytes=3136, priority=1,d1_dst=00:00:00:00:00:01 actions=output:"s2-eth6"
*** s3 ***
cookie=0x0, duration=1316.826s, table=0, n_packets=0, n_bytes=0, priority=20000,tcp,nw_dst=10.0.0.10,tp_dst=22 actions=output:"s3-eth4"
cookie=0x0, duration=1316.827s, table=0, n_packets=0, n_bytes=0, priority=1,d1_dst=00:00:00:00:00:01 actions=output:"s3-eth6"
mininet> net
h0 h0-eth0:s1-eth1
h1 h1-eth0:s2-eth1
h2 h2-eth0:s2-eth2
h3 h3-eth0:s2-eth3
h4 h4-eth0:s2-eth4
h5 h5-eth0:s2-eth5
h6 h6-eth0:s3-eth1
h7 h7-eth0:s3-eth2
h8 h8-eth0:s3-eth3
h9 h9-eth0:s3-eth4
h10 h10-eth0:s3-eth5
s1 lo: s1-eth1:h0-eth0 s1-eth2:s2-eth6 s1-eth3:s3-eth6
s2 lo: s2-eth1:h1-eth0 s2-eth2:h2-eth0 s2-eth3:h3-eth0 s2-eth4:h4-eth0 s2-eth5:h5-eth0 s2-eth6:s1-eth2 s2-eth7:s3-eth7
s3 lo: s3-eth1:h6-eth0 s3-eth2:h7-eth0 s3-eth3:h8-eth0 s3-eth4:h9-eth0 s3-eth5:h10-eth0 s3-eth6:s1-eth3 s3-eth7:s2-eth7
c0
```

h4, h5

## OpenFlow (2020 elővizsga)

Az alábbi parancsot kiadva (ügyelve arra, hogy a \$NEPTUN értéket a saját, tényleges neptun-kódodra cseréld le) egy felugró ablakban elindul egy mininetes hálózatemuláció és egy másik ablakban pedig egy pox kontroller. Kiadandó parancs a saját környezet indítására: `wget -nv -O- https://sb.tmit.bme.hu/haepuz/openflow | sudo sh /dev/stdin $NEPTUN`

✓ Ha a második bejegyzésre sosem érkezik illeszkedő forgalom, akkor az indítás után hány másodperccel törlődik a bejegyzés? (1 pont) 1/1

Csak egy bejegyzés található a folyamatlában.



```
mininet> dpctl dump-flows
*** s1 ***
cookie=0x0, duration=12.468s, table=0, n_packets=0, n_bytes=0, idle_timeout=8017, hard_timeout=9017, udp,tp_src=101 actions=CONTROLLER:65535
cookie=0x0, duration=12.468s, table=0, n_packets=0, n_bytes=0, idle_timeout=8017, hard_timeout=9017, udp,tp_src=102 actions=CONTROLLER:65535
cookie=0x0, duration=12.468s, table=0, n_packets=0, n_bytes=0, idle_timeout=8017, hard_timeout=9017, udp,tp_src=103 actions=CONTROLLER:65535
cookie=0x0, duration=12.467s, table=0, n_packets=0, n_bytes=0, idle_timeout=8017, hard_timeout=9017, udp,tp_src=104 actions=CONTROLLER:65535
cookie=0x0, duration=12.467s, table=0, n_packets=0, n_bytes=0, idle_timeout=8017, hard_timeout=9017, udp,tp_src=105 actions=CONTROLLER:65535
cookie=0x0, duration=12.467s, table=0, n_packets=0, n_bytes=0, idle_timeout=8017, hard_timeout=9017, udp,tp_src=106 actions=CONTROLLER:65535
cookie=0x0, duration=12.467s, table=0, n_packets=0, n_bytes=0, idle_timeout=8017, hard_timeout=9017, udp,tp_src=107 actions=CONTROLLER:65535
```

8017 (idle-timeout)

✓ Ha az első bejegyzésre folyamatosan érkezik illeszkedő forgalom, akkor az indítás után hány másodperccel törlődik a bejegyzés? (1 pont) 1/1

9016



```
mininet> dpctl dump-flows
** s1 **
cookie=0x0, duration=12.468s, table=0, n_packets=0, n_bytes=0, idle_timeout=8017, hard_timeout=9017, udp,tp_src=101 actions=CONTROLLER:65535
cookie=0x0, duration=12.468s, table=0, n_packets=0, n_bytes=0, idle_timeout=8017, hard_timeout=9017, udp,tp_src=102 actions=CONTROLLER:65535
cookie=0x0, duration=12.468s, table=0, n_packets=0, n_bytes=0, idle_timeout=8017, hard_timeout=9017, udp,tp_src=103 actions=CONTROLLER:65535
```

9017 (hard- timeout)

✓ Hány darab folyambejegyzés található a kapcsoló folyamatáblájában? (1 pont) 1/1

1



```
mininet> dpctl dump-flows
*** s1 -----
cookie=0x0, duration=12.468s, table=0, n_packets=0, n_bytes=0, idle_timeout=8017, hard_timeout=9017, udp,tp_src=101 actions=CONTROLLER:65535
cookie=0x0, duration=12.468s, table=0, n_packets=0, n_bytes=0, idle_timeout=8017, hard_timeout=9017, udp,tp_src=102 actions=CONTROLLER:65535
cookie=0x0, duration=12.468s, table=0, n_packets=0, n_bytes=0, idle_timeout=8017, hard_timeout=9017, udp,tp_src=103 actions=CONTROLLER:65535
cookie=0x0, duration=12.467s, table=0, n_packets=0, n_bytes=0, idle_timeout=8017, hard_timeout=9017, udp,tp_src=104 actions=CONTROLLER:65535
cookie=0x0, duration=12.467s, table=0, n_packets=0, n_bytes=0, idle_timeout=8017, hard_timeout=9017, udp,tp_src=105 actions=CONTROLLER:65535
cookie=0x0, duration=12.467s, table=0, n_packets=0, n_bytes=0, idle_timeout=8017, hard_timeout=9017, udp,tp_src=106 actions=CONTROLLER:65535
cookie=0x0, duration=12.467s, table=0, n_packets=0, n_bytes=0, idle_timeout=8017, hard_timeout=9017, udp,tp_src=107 actions=CONTROLLER:65535
```

Ahány sor van. képen:7



## 2. Gyakorlati feladat (saját HaEpUz VM)

6 pont

Az alábbi parancsot kiadva (ügyelve arra, hogy a \$NEPTUN értéket a saját, tényleges neptun-kódodra cseréld le, természetesen \$ nélkül!) egy felugró ablakban elindul egy mininetes hálózatemuláció és egy másik ablakban pedig egy pox kontroller. Kiadandó parancs a saját környezet indítására: `wget -nv -O- https://sb.tmit.bme.hu/haepuz/star2 | sudo sh /dev/stdin $NEPTUN`

A két rossz hoszt közül mi a nagyobb sorszámú hoszt neve? (1 pont)

Bónusz feladat kizárólag IMSc pontokért (max 15 IMSc pont): Azt szeretnénk elérni, hogy a h1 hosztról a 10.0.0.10 címre küldött http lekérdezést (80-as port) a h9-es web szerver szolgálja ki, de az ssh kapcsolat (22-es port) a h10-es géppel menjen. Ehhez milyen parancsok kiadása szükséges? (segítség: pl. ovs-ofctl parancsok használhatók)

Ha nem szeretnénk felesleges parancsokat kiadni, milyen parancs kiadásával kezdjük a hiba elhárítását a kisebb sorszámú hoszton? (2 pont)

```
arp
ip addr del
ip link set dev
ip route
ip addr add
route del default gw
dhclient -v
sysctl -w net.ipv4.ip_forward=1
```

Ezt nem tudjuk!!

✓ Az alábbi OpenFlow folyambejegyzés...

1/1

```
cookie=0x0, duration=20s, table=0, n_packets=400, n_bytes=200000,  
idle_timeout=20, hard_timeout=30, idle_age=15, priority=65535,  
tcp, in port=1, vlan tci=0x0000, dl_src=00:00:00:00:00:01, dl_dst=00:00:00:00:00:02,  
nw_src=10.0.0.1, nw_dst=10.0.0.2, nw_tos=0, tp_src=1111, tp_dst=2222  
actions=output:2
```

- ☒ által továbbított csomagok átlagos hossza kisebb, mint 1000 byte ✓

$$n\_bytes=200000 / n\_packets=400 = 500 < 1000 \text{ byte}$$

- ☐ 15s múlva még aktív lesz, ha csak egyetlen illeszkedő csomag érkezik pont 8s múlva

$$duration=20s, + 15 \text{ sec} = 35 \text{ sec} > hard\_timeout=30, \quad \times$$

$$idle\_age=15, + 8 \text{ sec} = 23 \text{ sec} > idle\_timeout=20, \quad \times$$

- ☐ átlagosan kevesebb, mint 20 kbps forgalmat továbbított

- ☒ 8s múlva még aktív lesz, ha csak egyetlen illeszkedő csomag érkezik pont 4s múlva ✓

$$duration=20s, + 8 \text{ sec} = 28 \text{ sec} < hard\_timeout=30, \quad \checkmark$$

$$idle\_age=15, + 4 \text{ sec} = 19 \text{ sec} < idle\_timeout=20, \quad \checkmark$$

**Átlagosan kevesebb, mint 20 kbps forgalmat továbbított:**

20s (duration) alatt 200000 byte-ot (n\_bytes) továbbított. + byte -> bit -> kbit átváltás

n\_bytes    durat.    byte>bit>kbit

$$200000 / 20 * 8 / 1000 = 80 \text{ kbps} \quad (80 > 20)$$