

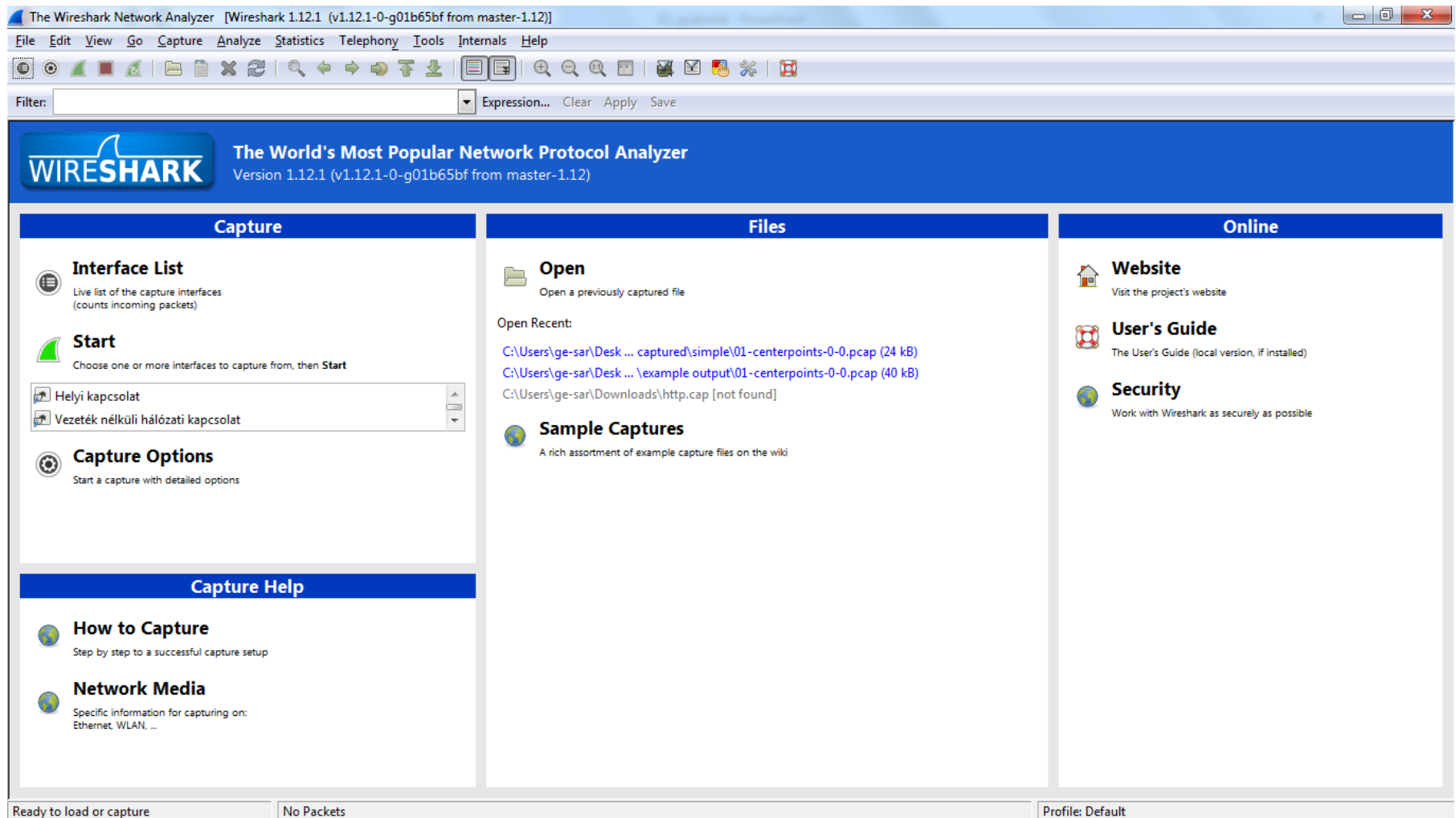
Telekommunikációs Hálózatok

10. gyakorlat

Wireshark

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

Wireshark



Wireshark ablakok

The screenshot displays the Wireshark interface with the following components:

- Filter Bar:** Located at the top, it contains a text input field with the filter `http.out.pcapng` and a search icon.
- Packet List:** A table showing captured packets. The columns are No., Time, Source, Destination, Protocol, Length, and Info. The first four packets are TCP SYN packets from 192.168.1.100 to 173.194.116.143.
- Packet Details:** A hierarchical view of the selected packet (No. 1). It shows the Ethernet II header, Internet Protocol Version 4 header, and Transmission Control Protocol header.
- Packet Bytes:** A hex dump view of the selected packet's raw bytes, showing the Ethernet II frame structure.
- Status Bar:** At the bottom, it displays "Packets: 1453 - Displayed: 1453 (100.0%)" and "Profiles: Default".

Szűrők definiálására
alkalmas input eszközök

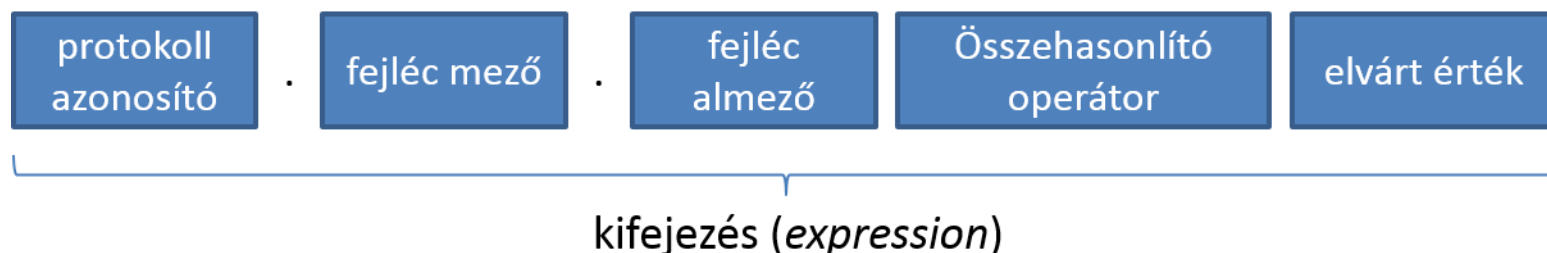
Csomag összefoglaló
nézete

Kiválasztott csomag
hierarchikus nézete

Kiválasztott csomag
bájt-alapú nézete

Szűrés statisztikái

Wireshark szűrők

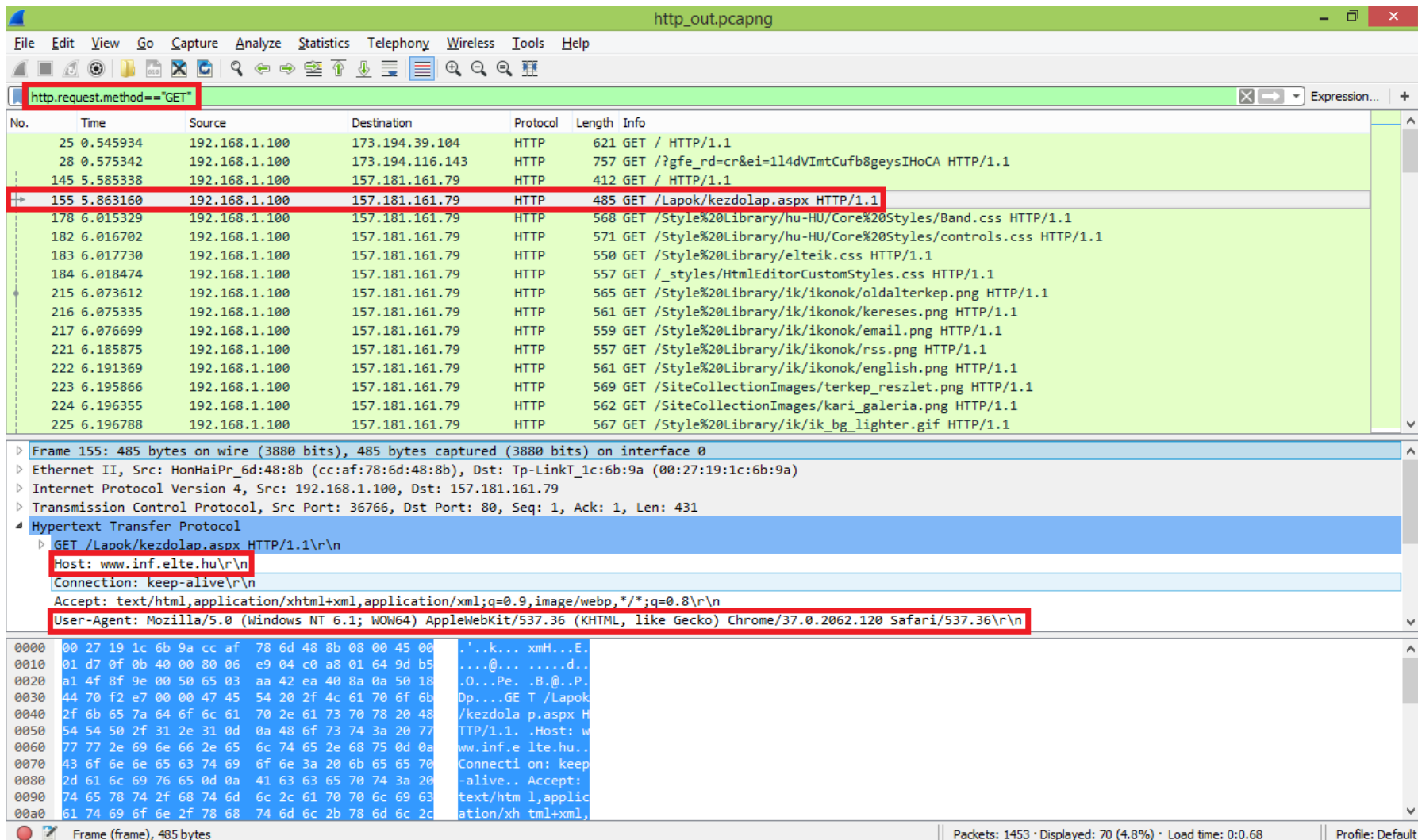


- Operátorok: or, and, xor, not
- protokollok: ip, tcp, http... (teljes listát lásd → Analyze→Display filter expression...)
- Példa: tcp.flags.ack==1 and tcp.dstport==80 (tcp nyugta flag és fogadó port beállítva)

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*.)

Wireshark példa megoldás I.



The image shows a Wireshark capture of an HTTP GET request. The packet list on the left shows a series of HTTP requests from 192.168.1.100 to 157.181.161.79. The selected packet (155) is a GET request for /Lapok/kezdolap.aspx. The packet details on the right show the structure of the HTTP request, including the Host, Connection, Accept, and User-Agent headers. The packet bytes on the bottom show the raw data of the request.

Filter: `http.request.method=="GET"`

No.	Time	Source	Destination	Protocol	Length	Info
25	0.545934	192.168.1.100	173.194.39.104	HTTP	621	GET / HTTP/1.1
28	0.575342	192.168.1.100	173.194.116.143	HTTP	757	GET /?gfe_rd=cr&ei=1l4dVImtCufb8geysIHoCa HTTP/1.1
145	5.585338	192.168.1.100	157.181.161.79	HTTP	412	GET / HTTP/1.1
155	5.863160	192.168.1.100	157.181.161.79	HTTP	485	GET /Lapok/kezdolap.aspx HTTP/1.1
178	6.015329	192.168.1.100	157.181.161.79	HTTP	568	GET /Style%20Library/hu-HU/Core%20Styles/Band.css HTTP/1.1
182	6.016702	192.168.1.100	157.181.161.79	HTTP	571	GET /Style%20Library/hu-HU/Core%20Styles/controls.css HTTP/1.1
183	6.017730	192.168.1.100	157.181.161.79	HTTP	550	GET /Style%20Library/elteik.css HTTP/1.1
184	6.018474	192.168.1.100	157.181.161.79	HTTP	557	GET /_styles/HtmlEditorCustomStyles.css HTTP/1.1
215	6.073612	192.168.1.100	157.181.161.79	HTTP	565	GET /Style%20Library/ik/ikonok/oldalalterkep.png HTTP/1.1
216	6.075335	192.168.1.100	157.181.161.79	HTTP	561	GET /Style%20Library/ik/ikonok/kereses.png HTTP/1.1
217	6.076699	192.168.1.100	157.181.161.79	HTTP	559	GET /Style%20Library/ik/ikonok/email.png HTTP/1.1
221	6.185875	192.168.1.100	157.181.161.79	HTTP	557	GET /Style%20Library/ik/ikonok/rss.png HTTP/1.1
222	6.191369	192.168.1.100	157.181.161.79	HTTP	561	GET /Style%20Library/ik/ikonok/english.png HTTP/1.1
223	6.195866	192.168.1.100	157.181.161.79	HTTP	569	GET /SiteCollectionImages/terkep_reszlet.png HTTP/1.1
224	6.196355	192.168.1.100	157.181.161.79	HTTP	562	GET /SiteCollectionImages/kari_galeria.png HTTP/1.1
225	6.196788	192.168.1.100	157.181.161.79	HTTP	567	GET /Style%20Library/ik/ik_bg_lighter.gif HTTP/1.1

Frame 155: 485 bytes on wire (3880 bits), 485 bytes captured (3880 bits) on interface 0

Ethernet II, Src: HonHaiPr_6d:48:8b (cc:af:78:6d:48:8b), Dst: Tp-LinkT_1c:6b:9a (00:27:19:1c:6b:9a)

Internet Protocol Version 4, Src: 192.168.1.100, Dst: 157.181.161.79

Transmission Control Protocol, Src Port: 36766, Dst Port: 80, Seq: 1, Ack: 1, Len: 431

Hypertext Transfer Protocol

GET /Lapok/kezdolap.aspx HTTP/1.1\r\n

Host: www.inf.elte.hu\r\n

Connection: keep-alive\r\n

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8\r\n

User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/37.0.2062.120 Safari/537.36\r\n

0000 00 27 19 1c 6b 9a cc af 78 6d 48 8b 08 00 45 00 .'.k... xmH...E.
0010 01 d7 0f 0b 40 00 80 06 e9 04 c0 a8 01 64 9d b5@... ..d..
0020 a1 4f 8f 9e 00 50 65 03 aa 42 ea 40 8a 0a 50 18 .O...Pe..B@..P..
0030 44 70 f2 e7 00 00 47 45 54 20 2f 4c 61 70 6f 6b Op...GE T /Lapok
0040 2f 6b 65 7a 64 6f 6c 61 70 2e 61 73 70 78 20 48 /kezdola p.aspx H
0050 54 54 50 2f 31 2e 31 0d 0a 48 6f 73 74 3a 20 77 TTP/1.1. .Host: w
0060 77 77 2e 69 6e 66 2e 65 6c 74 65 2e 68 75 0d 0a ww.inf.e lte.hu..
0070 43 6f 6e 6e 65 63 74 69 6f 6e 3a 20 6b 65 65 70 Connecti on: keep
0080 2d 61 6c 69 76 65 0d 0a 41 63 63 65 70 74 3a 20 -alive.. Accept:
0090 74 65 78 74 2f 68 74 6d 6c 2c 61 70 70 6c 69 63 text/htm l,applic
00a0 61 74 69 6f 6e 2f 78 68 74 6d 6c 2b 78 6d 6c 2c ation/xh tml+xml,

Frame (frame), 485 bytes

Packets: 1453 · Displayed: 70 (4.8%) · Load time: 0:0.68

Profile: Default

Wireshark példa megoldás I.

- Milyen oldalakat kértek le a szűrés alapján HTTP GET metódussal? Milyen böngészőt használtak hozzá?
- Szűrés: `http.request.method=="GET"`
- User-agent header-ből lehet következtetni a böngésző típusára: User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/37.0.2062.120 Safari/537.36
- Ehhez segítségünkre lehet ez a link:
http://www.zytrax.com/tech/web/browser_ids.htm

Wireshark példa megoldás II.

http_out.pcapng

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

http.accept == "image/webp,*/*;q=0.8"

No.	Time	Source	Destination	Protocol	Length	Info
215	6.073612	192.168.1.100	157.181.161.79	HTTP	565	GET /Style%20Library/ik/ikonok/oldalterkep.png HTTP/1.1
216	6.075335	192.168.1.100	157.181.161.79	HTTP	561	GET /Style%20Library/ik/ikonok/kereses.png HTTP/1.1
217	6.076699	192.168.1.100	157.181.161.79	HTTP	559	GET /Style%20Library/ik/ikonok/email.png HTTP/1.1
221	6.185875	192.168.1.100	157.181.161.79	HTTP	557	GET /Style%20Library/ik/ikonok/rss.png HTTP/1.1

Frame 215: 565 bytes on wire (4520 bits), 565 bytes captured (4520 bits) on interface 0

Ethernet II, Src: HonHaiPr_6d:48:8b (cc:af:78:6d:48:8b), Dst: Tp-LinkT_1c:6b:9a (00:27:19:1c:6b:9a)

Internet Protocol Version 4, Src: 192.168.1.100, Dst: 157.181.161.79

Transmission Control Protocol, Src Port: 36766, Dst Port: 80, Seq: 432, Ack: 33346, Len: 511

Hypertext Transfer Protocol

GET /Style%20Library/ik/ikonok/oldalterkep.png HTTP/1.1\r\n

Host: www.inf.elte.hu\r\n

Connection: keep-alive\r\n

Accept: image/webp,*/*;q=0.8\r\n

User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/37.0.2062.120 Safari/537.36\r\n

Referer: http://www.inf.elte.hu/Lapok/kezdolap.aspx\r\n

Accept-Encoding: gzip,deflate,sdch\r\n

Accept-Language: hu-HU,hu;q=0.8,en-US;q=0.6,en;q=0.4\r\n

If-None-Match: "{02084E23-E4E5-4B60-A388-F77D8FAD8892},2"\r\n

If-Modified-Since: Wed, 16 Jan 2008 11:58:25 GMT\r\n

0090 3a 20 6b 65 65 70 2d 61 6c 69 76 65 0d 0a 41 63 : keep-a live..Ac

00a0 63 65 70 74 3a 20 69 6d 61 67 65 2f 77 65 62 70 cept: im age/webp

00b0 2c 2a 2f 2a 3b 71 3d 30 2e 38 0d 0a 55 73 65 72 ,*/*;q=0.8..User

00c0 2d 41 67 65 6e 74 3a 20 4d 6f 7a 69 6c 6c 61 2f -Agent: Mozilla/

00d0 35 2e 30 20 28 57 69 6e 64 6f 77 73 20 4e 54 20 5.0 (Win dows NT

00e0 36 2e 31 3b 20 57 4f 57 36 34 29 20 41 70 70 6c 6.1; WOW 64) Appl

00f0 65 57 65 62 4b 69 74 2f 35 33 37 2e 33 36 20 28 eWebKit/ 537.36 (

0100 4b 48 54 4d 4c 2c 20 6c 69 6b 65 20 47 65 63 6b KHTML, l ike Geck

0110 6f 29 20 43 68 72 6f 6d 65 2f 33 37 2e 30 2e 32 o) Chrom e/37.0.2

0120 30 36 32 2e 31 32 30 20 53 61 66 61 72 69 2f 35 062.120 Safari/5

0130 33 37 2e 33 36 0d 0a 52 65 66 65 72 65 72 3a 20 37.36..R eferer:

0140 68 74 74 70 3a 2f 2f 77 77 72 2e 69 6e 66 2e 65 http://w ww.inf.e

0150 6c 74 65 2e 68 75 2f 4c 61 70 6f 6b 2f 6b 65 7a lte.hu/L apok/kez

0160 64 6f 6c 61 70 2e 61 73 70 78 0d 0a 41 63 63 65 dolap.as px..Acce

0170 70 74 2d 45 6e 63 6f 64 69 6e 67 3a 20 67 7a 69 pt-Encod ing: gzi

0180 70 2c 64 65 66 6c 61 74 65 2c 73 64 63 68 0d 0a p,deflat e,sdch..

0190 41 63 63 65 70 74 2d 4c 61 6e 67 75 61 67 65 3a Accept-L anguage:

01a0 20 68 75 2d 48 55 2c 68 75 3b 71 3d 30 2e 38 2c hu-HU,h u;q=0.8,

HTTP Accept (http.accept), 30 bytes

Packets: 1453 Displayed: 26 (1.8%) Load time: 0:0.63 Profile: Default

Wireshark példa megoldás II.

- Hány darab képet érintett a böngészés?
- Szűrés: `http.accept == "image/webp,*/*;q=0.8"`
- Accept header: a kérésre adott válasz tartalmának elfogadható típusa
- WebP: új, veszteséges tömörítést alkalmazó képformátum, amelyet a Google fejlesztett ki a web hálózati forgalmának csökkentésére

Wireshark példa megoldás III.

Wireshark capture of an HTTPS connection. The filter is `tcp.dstport==443`. The packet list shows a SYN, ACK, and TLS handshake. Packet 13 is selected, showing details of the TLS Client Hello and the application data `http_out.pcapng`.

No.	Time	Source	Destination	Protocol	Length	Info
4	0.007666	192.168.1.100	173.194.116.143	TCP	66	36763→443 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM=1
10	0.043654	192.168.1.100	173.194.116.143	TCP	54	36763→443 [ACK] Seq=1 Ack=1 Win=17160 Len=0
13	0.045430	192.168.1.100	173.194.116.143	TLSv1.2	267	Client Hello
17	0.082913	192.168.1.100	173.194.116.143	TCP	54	36763→443 [ACK] Seq=214 Ack=2861 Win=17160 Len=0
19	0.099481	192.168.1.100	173.194.116.143	TLSv1.2	308	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
22	0.133310	192.168.1.100	173.194.116.143	TCP	54	36763→443 [ACK] Seq=468 Ack=4215 Win=15804 Len=0
24	0.328037	192.168.1.100	173.194.116.143	TCP	54	36763→443 [ACK] Seq=468 Ack=4252 Win=15768 Len=0
31	0.656301	192.168.1.100	173.194.116.143	TLSv1.2	103	Application Data
32	0.656739	192.168.1.100	173.194.116.143	TLSv1.2	91	Application Data
33	0.657355	192.168.1.100	173.194.116.143	TLSv1.2	111	Application Data
34	0.659378	192.168.1.100	173.194.116.143	TLSv1.2	693	Application Data
35	0.671176	192.168.1.100	173.194.39.120	TCP	66	36764→443 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM=1
37	0.692679	192.168.1.100	173.194.39.120	TCP	54	36764→443 [ACK] Seq=1 Ack=1 Win=17160 Len=0
39	0.693518	192.168.1.100	173.194.39.120	TLSv1.2	269	Client Hello
43	0.710588	192.168.1.100	173.194.39.120	TCP	54	36764→443 [ACK] Seq=216 Ack=2861 Win=17160 Len=0
45	0.727492	192.168.1.100	173.194.39.120	TLSv1.2	308	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
48	0.745754	192.168.1.100	173.194.39.120	TCP	54	36764→443 [ACK] Seq=470 Ack=4216 Win=15804 Len=0
53	0.794975	192.168.1.100	173.194.116.143	TCP	54	36763→443 [ACK] Seq=1250 Ack=5893 Win=17160 Len=0

Frame 13: 267 bytes on wire (2136 bits), 267 bytes captured (2136 bits) on interface 0

- Ethernet II, Src: HonHaiPr_6d:48:8b (cc:af:78:6d:48:8b), Dst: Tp-LinkT_1c:6b:9a (00:27:19:1c:6b:9a)
- Internet Protocol Version 4, Src: 192.168.1.100, Dst: 173.194.116.143
- Transmission Control Protocol, Src Port: 36763, Dst Port: 443, Seq: 1, Ack: 1, Len: 213
- Secure Sockets Layer

0030 10 c2 4c d5 00 00 16 03 01 00 d0 01 00 00 cc 03 ..L... ..
0040 03 79 3e 23 0a c0 08 98 16 d7 3a 0f 4c 50 0b 3d .y>#... ..LP.=
0050 35 01 f1 22 2f 12 50 7a ad 5c 7b 13 2d a7 5c cd 5.."/.Pz .\{.-.\.
0060 ea 00 00 28 cc 14 cc 13 c0 2b c0 2f 00 9e c0 0a ...((... .+./....
0070 c0 09 c0 13 c0 14 c0 07 c0 11 00 33 00 32 00 393.2.9
0080 00 9c 00 2f 00 35 00 0a 00 05 00 04 01 00 00 7b .../.5.{
0090 00 00 00 12 00 10 00 00 0d 77 77 77 2e 67 6f 6fwww.goo
00a0 67 6c 65 2e 68 75 ff 01 00 01 00 00 0a 00 08 00 gle.hu... ..
00b0 06 00 17 00 18 00 19 00 0b 00 02 01 00 00 23 00#.

Secure Sockets Layer (ssl), 213 bytes

Packets: 1453 · Displayed: 428 (29.5%) · Load time: 0:0.67 · Profile: Default

Wireshark példa megoldás III.

- Volt-e olyan kérés, amely titkosított kommunikációt takar?
- Szűrés: `tcp.dstport==443`
- Transport Layer Security (TLS) titkosító protokoll feletti HTTP kommunikációra utal a 443-as port
- Elektronikus levelezéshez, banki szolgáltatásokhoz stb. elengedhetetlen
- Nélküle le lehet hallgatni a kommunikációt, lásd a következő diát (sample3.pcapng felhasználásával)

Wireshark – „leleplezés”

The image shows the Wireshark network traffic analysis interface. The top menu bar includes File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Wireless, Tools, and Help. Below the menu is a toolbar with various icons for packet capture and analysis. A display filter bar shows "Apply a display filter ... <Ctrl-/>". The packet list pane shows three packets:

No.	Time	Source	Destination	Protocol	Length	Info
12	4.106306	192.168.2.101	84.2.36.197	HTTP	131	POST /pages/user/login.jsp?method=Login HTTP/1.1 (application/x-www-form-urlencoded)
13	4.121234	84.2.36.197	192.168.2.101	TCP	60	80→65533 [ACK] Seq=1 Ack=1393 Win=7890 Len=0
14	6.160352	84.2.36.197	192.168.2.101	HTTP	594	HTTP/1.1 302 Moved Temporarily

The packet details pane for packet 14 shows the following structure:

- Internet Protocol Version 4, Src: 192.168.2.101, Dst: 84.2.36.197
- Transmission Control Protocol, Src Port: 65533, Dst Port: 80, Seq: 1316, Ack: 1, Len: 77
- [2 Reassembled TCP Segments (1392 bytes): #11(1315), #12(77)]
- Hypertext Transfer Protocol
 - POST /pages/user/login.jsp?method=Login HTTP/1.1\r\n
 - Host: **iiww.hu\r\n**
 - Connection: keep-alive\r\n
 - Referer: http://iiww.hu/pages/user/login.jsp\r\n
 - Content-Length: 77\r\n
 - Cache-Control: max-age=0\r\n
 - Origin: http://iiww.hu\r\n
 - Content-Type: application/x-www-form-urlencoded\r\n
 - Accept: application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5\r\n
 - User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.0; en-US) AppleWebKit/534.3 (KHTML, like Gecko) Chrome/6.0.472.59 Safari/534.3\r\n
 - Accept-Encoding: gzip,deflate,sdch\r\n
 - Accept-Language: hu-HU,hu;q=0.8,en-US;q=0.6,en;q=0.4\r\n
 - Accept-Charset: ISO-8859-2,utf-8;q=0.7,*;q=0.3\r\n
 - [truncated]Cookie: ajaxable=1; forgetEmail=0; email=bGFRaXNAaW5mLmVsdGUuaHU\$; httpslogin=0; __utma=114476831.2067882217.1284887869.1284887869.1284887869.1; __utmc=114476831;\r\n
 - [Full request URI: http://iiww.hu/pages/user/login.jsp?method=Login]
 - [HTTP request 1/1]
 - [Response in frame: 14]
 - File Data: 77 bytes
- HTML Form URL Encoded: application/x-www-form-urlencoded
 - Form item: "email" = "kulimasz@perek.hu"
 - Form item: "password" = "kistraktor53"

The packet bytes pane shows the raw data in hexadecimal and ASCII:

```
0020 24 c5 ff fd 00 50 e6 41 31 0f 9a 6f 83 0a 50 18 $...P.A 1..o..P.
0030 fa f0 7c ef 00 00 65 6d 61 69 6c 3d 6b 75 6c 69 ..|...em ail=kuli
0040 6d 61 73 7a 25 34 30 70 65 72 65 63 2e 68 75 26 masz%40p er ec.hu&
0050 70 61 73 73 77 6f 72 64 3d 6b 69 73 74 72 61 6b password =kistrak
0060 74 6f 72 35 33 26 68 74 74 70 73 6c 6f 67 69 6e tor53&ht tpslogin
```

The status bar at the bottom shows "sample3" and "Packets: 381 · Displayed: 381 (100.0%) · Load time: 0:0.16 · Profile: Default".

MININET

Előfeltétel: *Mininet beállítás.pdf* diasoron végigmenni!

Mininet – következő indítás (ha már egyszer be lett állítva)

- A VM indításakor be kell lépni és az alábbi parancsot kiadni:

```
mininet> sudo dhclient
```

- Utána ellenőrizni, hogy milyen privát IP címet kapott:

- (Nagy valószínűség szerint ez ugyanaz, mint korábban, de ha mégsem, akkor sajnos MobaXterm-nél a session-nél módosítani kell a "Remote host"-nál ugyanarra az IP címre.)

```
mininet> ifconfig
```

- Ezután ki lehet "exit" paranccsal lépni a VM-ből, de fontos, hogy nem szabad lezárni a gépet, hanem a MobaXterm-nél el kell indítani a megfelelő session-t. Ezután az alábbiakat kell kiadni:

```
mininet> xauth list
```

- Ha itt a "networksELTE/unix:10..." és "networksELTE:10..." soroknál (vagy "networksELTE/unix:11..." és "networksELTE:11..." soroknál stb.) nem egyezik az alfanumerikus karaktersorozat, akkor újra ki kell adni az "xauth add..." parancsot.

Mininet

- Belépés után listázzuk az alábbi könyvtárt:

```
networks@networksELTE:~$ ls mininetScriptek/ComputerNetworks/L2-switching/
```

- test1 topológia két fájlból áll:
- test1.mn: meg lehet jeleníteni a miniedit segítségével
- test1.py: egyből elindítja a hálózat emulátort

Mininet

- Indítsuk el a miniedit-et:

```
networks@networksELTE:~$ python mininet/examples/miniedit.py&
```

- a *File* menüben meg tudjuk nyitni a .mn kiterjesztésű fájlokat
- Nyissuk meg a test1.mn fájlt
- A *File* menüben az „Export Level 2 Script”-tel lehet létrehozni python szkriptet

Mininet

- Nézzük meg a `test1.py`-t:

```
networks@networksELTE:~/mininetScriptek/ComputerNetworks/L2-switching$ vi test1.py
```

- Egy `LinuxBridge`-et definiálunk, amellyel futtatni tudjuk a feszítőfa algoritmust (Spanning Tree Protocol, STP) hurkok kezelésére
- Hozzáadunk hosztokat is, privát IP címekkel
- Végül összekötjük ezeket a topológia alapján
- A `h1` és `s1` kapcsolat sávszélessége: 10 Mbps (alapból elvileg nem limitált, a `TCLink` osztály azért kell, hogy limitálni tudjuk)

- Indítsuk el:

```
$ sudo python test1.py  
mininet>
```

Mininet

- Elérhető csomópontok:

```
mininet> nodes
```

- Az s1 switchről infót kaphatunk
 - (brctl: ethernet bridge adminisztráció)

```
mininet> sh brctl show
```

- Látszik, hogy nincs engedélyezve az STP
- A h1 h2 hostokon elindíthatunk egy-egy terminált:

```
mininet> xterm h1 h2
```

Mininet

- Itt lekérhetőek az interface adatok, érdemes a mac címet megnézni!

```
# ifconfig
```

- Írassuk ki az ARP tábla aktuális tartalmát:

```
# arp
```

- Az s1 switch forwarding tábláját lekérdezhetjük a mininet konzolban:

```
mininet> sh brctl showmacs s1
```

Mininet

- Derítsük ki, hogy melyik interfésze van s1-nek a h2-vel összekötve (mininet konzol):

```
# mininet> links  
h2-eth0<->s1-eth1 (OK OK)  
...
```

- Figyeljük a forgalmat az „s1-eth1” interfészen!
mininet konzolba írva:

```
mininet> s1 tcpdump -n -i s1-eth1
```

Mininet

- Pingetés xterm ablakból: h2 termináljából: (a h1 h2 nevek itt nem használhatók!)

```
# ping 10.0.0.1
```

- Írassuk ki az ARP tábla aktuális tartalmát:

```
# arp
```

Mininet

- Közben látjuk a mininet konzolban, hogy mentek ARP üzenetek
- Pingetés mininet konzolból, pl.:

```
mininet> h1 ping h2
```

- Kilépés:

```
mininet> exit
```

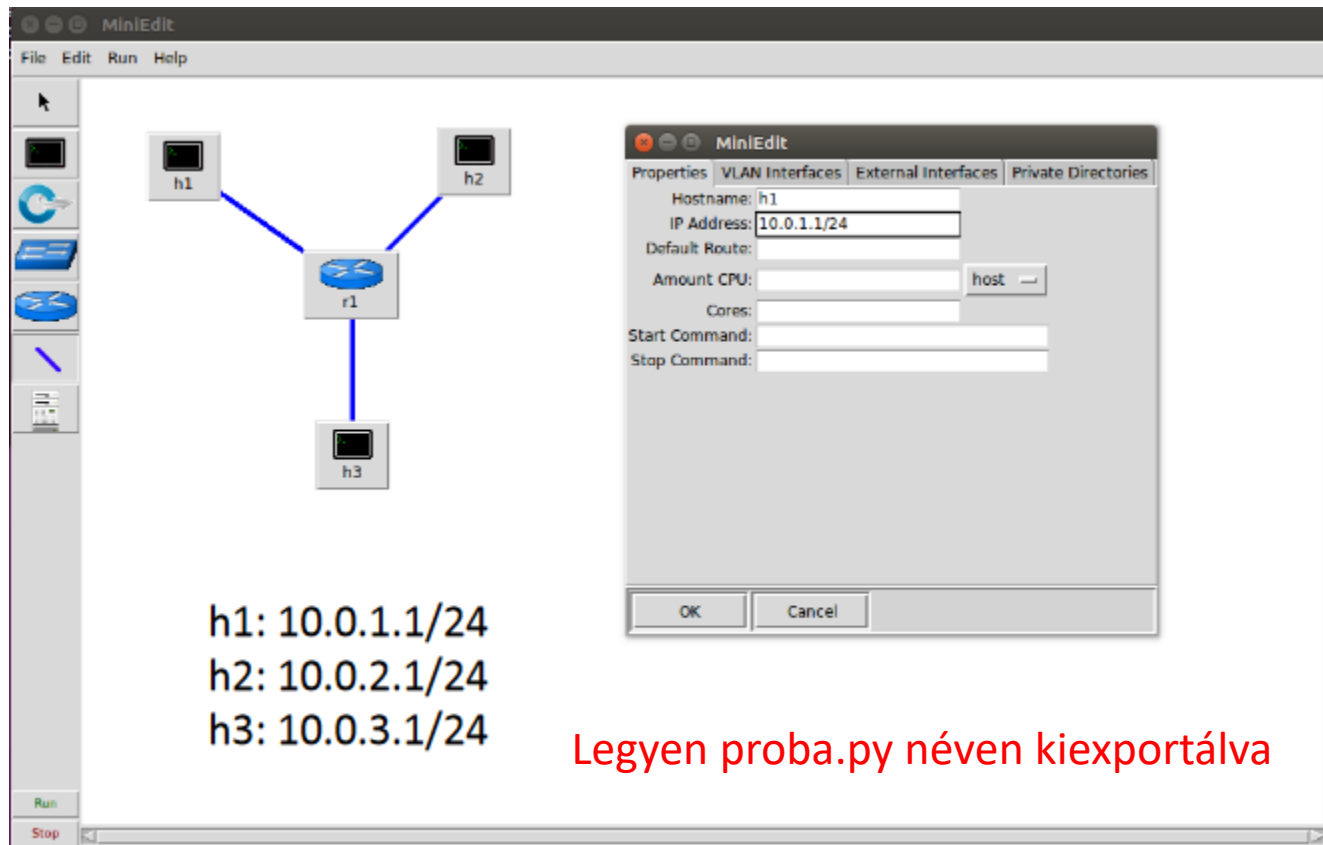
Mininet

- (Átváltás arra a könyvtárra, ahol a miniedit van:)

```
$ cd /home/networks/mininet/examples/  
python miniedit.py&
```


Mininet

- A következő példában létrehozunk miniedit-tel egy kis hálózatot:



Mininet

- Indítsuk el:

```
cd /home/networks/mininetScriptek/ComputerNetworks/L2-switching  
networks@networksELTE:~/mininetScriptek/ComputerNetworks/L2-switching$ sudo python proba.py  
mininet>
```

- A h1 h2 h3 hosztokon és a routeren elindíthatunk egy-egy terminált:

```
mininet> xterm h1 h2 h3 r1
```

Mininet

- A h1 termináljában próbáljuk ki a ping-et a h2 hoszthoz:

```
# ping 10.0.2.1  
connect: Network is unreachable
```

- Router interfész beállítása:

```
mininet> net  
r1 r1-eth0:h1-eth0 r1-eth1:h2-eth0 r1-eth2:h3-eth0  
h3 h3-eth0:r1-eth2  
h1 h1-eth0:r1-eth0  
h2 h2-eth0:r1-eth1
```

- Az r1 termináljában adjunk IP címeket az r1-eth0, r1-eth1, r1-eth2 interfészeknek:

```
# ip addr add 10.0.1.254/24 dev r1-eth0  
# ip addr add 10.0.2.254/24 dev r1-eth1  
# ip addr add 10.0.3.254/24 dev r1-eth2
```

Mininet

- A h2 termináljában az alapértelmezett útvonalat adjuk meg a 10.0.2.254 lokális átjárón keresztül, amelyet az h2-eth0 eszközön lehet elérni:

```
# ip route add default via 10.0.2.254 dev h2-eth0
```

Mininet

- A h3 termináljában az alapértelmezett útvonalat adjuk meg a 10.0.3.254 lokális átjárón keresztül, amelyet az h3-eth0 eszközön lehet elérni:

```
# ip route add default via 10.0.3.254 dev h3-eth0
```

Mininet

- A h1 termináljában az alapértelmezett útvonalat adjuk meg a 10.0.1.254 lokális átjárón keresztül, amelyet az h1-eth0 eszközön lehet elérni:

```
# ip route add default via 10.0.1.254 dev h1-eth0
```

- Ezután nézzük meg az IP routing táblát:

```
# route -n
```

- Most már működni fog a ping:

```
# ping 10.0.2.1
```

Mininet

- A h2 terminálját nyissuk meg!
- iptables szabályok kiírása:

```
# iptables-save
```

- vagy

```
# iptables -L
```

- Ping tiltás szabály felvétele a INPUT lánc elejére:

```
# iptables -I INPUT -p icmp --icmp-type echo-request -j DROP
```

Mininet

- Ping tiltás szabály hozzáfűzése az OUTPUT lánc végére:

```
# iptables -A OUTPUT -p icmp --icmp-type echo-request -j DROP
```

- Próba:

```
# ping 10.0.1.1
```

- Ping tiltás szabály törlése:

```
# iptables -D OUTPUT -p icmp --icmp-type echo-request -j DROP
```


Mininet

- iptables port forwarding
- A h3 terminálját nyissuk meg!
- h3 hoszton inditsunk el egy ssh daemon-t

```
# /usr/sbin/sshd
```

- Az r1 terminálját nyissuk meg!
- Állítsuk be a r1-es routeren a forwarding szabályt:

```
# iptables -t nat -A PREROUTING -i r1-eth0 -p tcp -d 10.0.2.1 --dport 2222 -j DNAT \  
--to-destination 10.0.3.1:22
```

- SSH-zunkbe h1-ről a h3-ra a port forwardinggal:

```
# ssh -p 2222 networks@10.0.2.1
```

Mininet

- Indítsuk el a miniedit-et:

```
$ cd /home/networks/mininet/examples/  
python miniedit.py&
```

- Nyissuk meg a `sw-topo.mn` fájlt

- Hurkot tartalmaz!

- Indítsuk el:

```
networks@networksELTE:~/mininetScriptek/ComputerNetworks/L2-switching$ sudo python sw-topo.py  
mininet>
```

Mininet

- Nézzük meg a switcheket a mininet konzolban:

```
mininet> sh brctl show
```

- STP mindenhol ki van kapcsolva!
- h1 és h2 szomszédok

```
mininet> h1 ping h2
```

- Azt tapasztaljuk, hogy nagy a késés és csak néhány csomag megy át
- h1 és h4 távol vannak egymástól

```
mininet> h1 ping h4
```

- Csak sikertelen próbálkozás lesz, semmi se megy át

Mininet

- tcpdump-pal érdekes jelenség látható:

```
mininet> sh tcpdump -n -i any
```

- Multicast üzenetek próbálják a hálózatot felderíteni
- Konklúzió: hurok van a hálózatban, nem igazán működik semmi
- Kilépés:

```
mininet> exit
```

Mininet

- Indítsuk el újra --stp kapcsolóval:

```
networks@networksELTE:~/mininetScriptek/ComputerNetworks/L2-switching$ sudo python sw-topo.py --stp  
mininet>
```

- bridge állapot:

```
mininet> sh brctl show
```

- STP információ az s2 switchhez:

```
mininet> sh brctl showstp s2
```

- Nézzük meg mit ír ki: ki a designated root, ki a designated bridge, mely portok blokkoltak (a körök kiszűrésére)?

Mininet

```
root@networks: /home/networks/ComputerNetworks/L2-switching
*** Starting CLI:
mininet> sh brctl show
bridge name      bridge id        STP enabled  interfaces
s2                8000.32c7c790adac  yes         s2-eth1
s2                8000.32c7c790adac  yes         s2-eth2
s2                8000.32c7c790adac  yes         s2-eth3
s3                8000.369e11b8a7b3  yes         s2-eth4
s3                8000.369e11b8a7b3  yes         s3-eth1
s3                8000.369e11b8a7b3  yes         s3-eth2
s4                8000.4a9490f7e79c  yes         s3-eth3
s4                8000.4a9490f7e79c  yes         s4-eth1
s4                8000.4a9490f7e79c  yes         s4-eth2
s5                8000.2e073f193228  yes         s4-eth3
s5                8000.2e073f193228  yes         s5-eth1
s5                8000.2e073f193228  yes         s5-eth2
s6                8000.1ea24d709a2f  yes         s5-eth3
s6                8000.1ea24d709a2f  yes         s6-eth1
s7                8000.2a410c04c349  yes         s6-eth2
s7                8000.2a410c04c349  yes         s7-eth1
s7                8000.2a410c04c349  yes         s7-eth2
s7                8000.2a410c04c349  yes         s7-eth3

mininet> sh brctl showstp s2
s2
bridge id        8000.32c7c790adac
designated root   8000.1ea24d709a2f
root port        2
max age          20.00
hello time        2.00
forward delay     15.00
ageing time       300.00
hello timer       0.00
topology change timer 0.00
flags

s2-eth1 (1)
port id          8001
designated root   8000.1ea24d709a2f
designated bridge 8000.32c7c790adac
designated port    8001
designated cost    4
flags

state            forwarding
path cost        2
message age timer 0.00
forward delay timer 0.00
hold timer       0.38
```

MiniEdit

File Edit Run Help

Run

Stop

Mininet

- Működik-e most a hálózat???

```
mininet> h1 ping h2
```

```
mininet> h1 ping h4
```

- és megy minden... érdemes még a tcpdumpot is futtatni:

```
mininet> s2 tcpdump -n -i any
```

- látjuk, ahogy az STP üzenetek mennek a szomszédok között.

Házi feladat

mininet beadandó

- **A feladat mindenkinek egyedi, emiatt le kell tölteni a megfelelő topológia fájlt, az alábbi linken keresztül!**
- Feladat leírása és topológiafájl letöltés:

<https://ggombos.web.elte.hu/halobeadando/6-mininet/>

Házi feladat

mininet beadandó

- **Leadás:** A programot **zip** formátumban kell leadni! A .zip fájlban **EGY** darab **input.txt** fájl legyen! A fájlban parancsok szerepeljenek olyan formában ahogy a mininet console-n meg lehet adni.
- Példa parancsok:
h1 ping 10.0.0.2
r3 ifconfig
- A TMS tesztelni fogja a beadott házifeladatot!
- Beadási határidő: **TMS rendszerben**

VÉGE
KÖSZÖNÖM A FIGYELMET!