

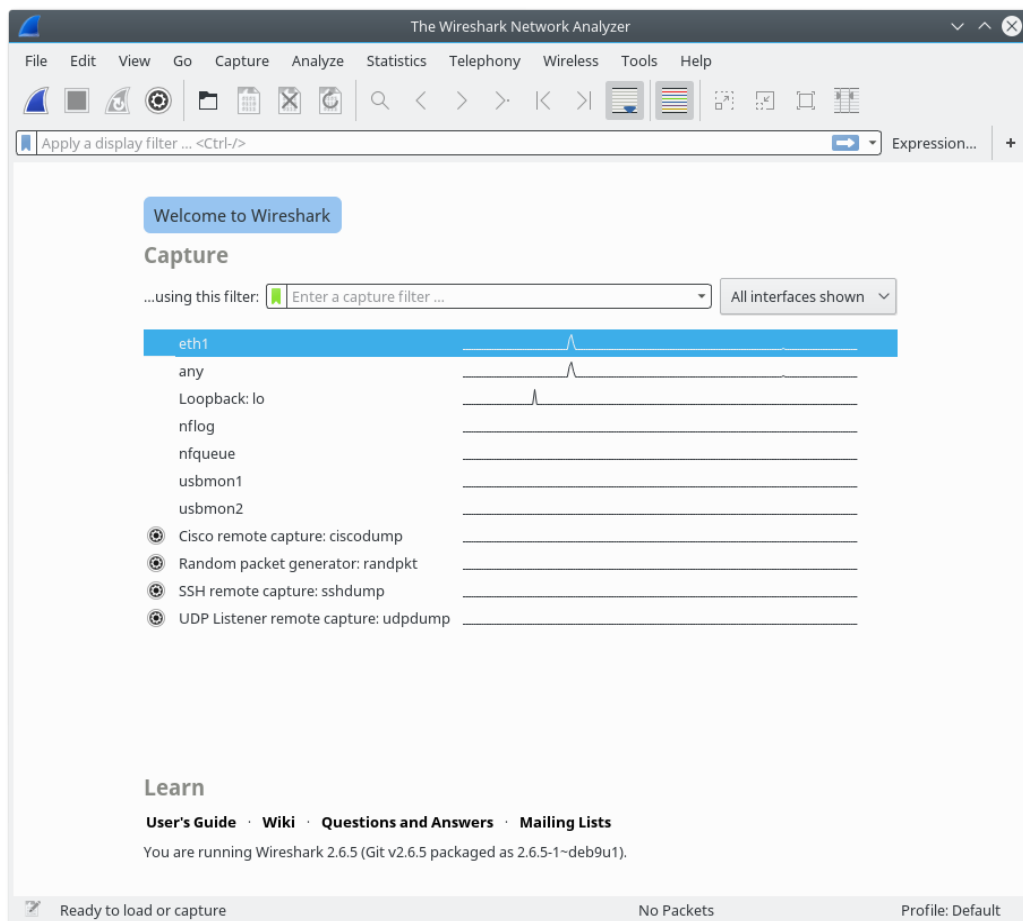


## Mérési utasítás

## Wireshark használata, TCP kapcsolatok analízálása

A Wireshark (korábbi nevén Ethereal) a legfejlettebb hálózati sniffer és analízátor program. 1998 óta fejlesztik, jelenleg a GPL 2 licenz alatt. Nem igen találni ilyen széleskörű szolgáltatásokkal és ismeretekkel rendelkező hálózati analízátor programot. Támogatott operációs rendszerek: Windows, Linux, OS X, Solaris, FreeBSD, NetBSD és még sok egyéb. Grafikus interaktív interfésszel rendelkezik. Az OSI ISO modell 2-7 rétegének minden implementációját tudja analízálni. A program által jelenleg ismert protokollok száma jelenleg több mint 81000!

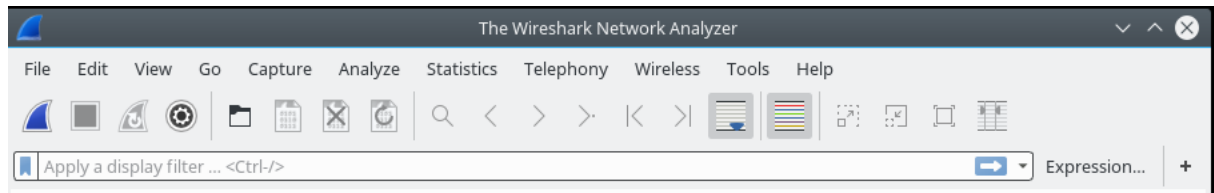
A Wireshark analízátor funkcióit több könyv, illetve elektronikus irodalom írja le több száz oldal terjedelemben, így gyakorlaton csak az alap funkciókkal ismerkedünk meg.



**1. Feladat.**

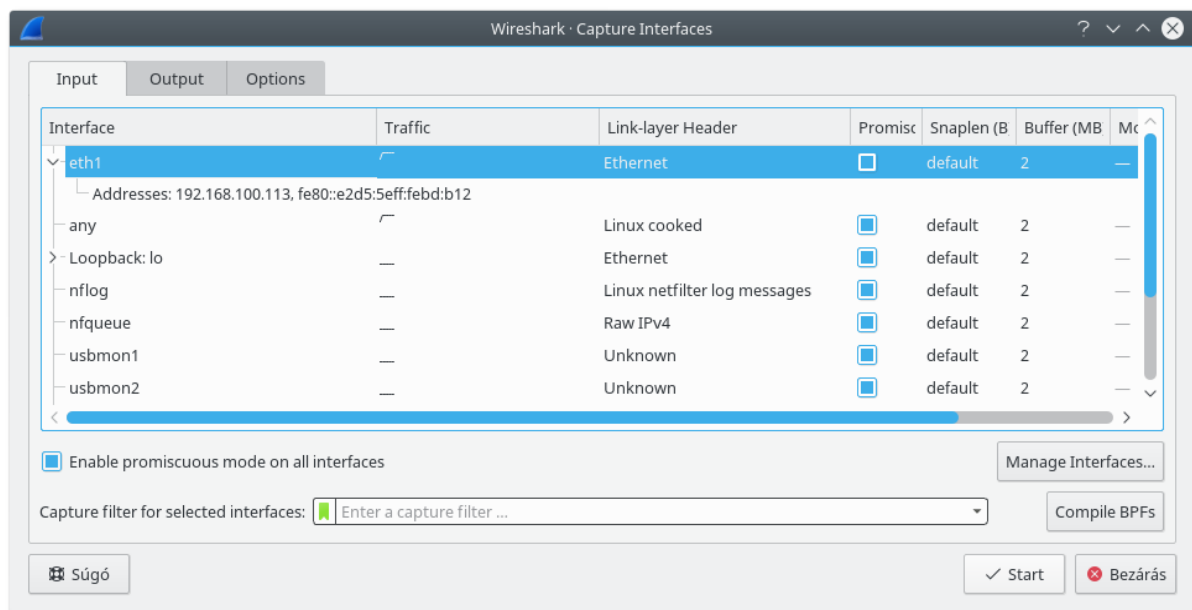
Nyissa meg a wiresharkot! (K-menu -> Alkalmazások ->)

Nézzük át a wireshark kezelőfelületét.



Az első gombbal alapbeállításokkal indíthatunk el csomagelkapást.

Ha egyedi csomagelkapást szeretnénk indítani akkor a beállítás (4. gomb) gombra kattintva előjön a használható interfészek listája valamint itt tudunk output és input beállításokat változtatni.



Legfelül látható, hogy jelen esetben az eth1-es interfészt használjuk. A „Enable promiscuous mode on all interface” kapcsolót mindig hagyjuk bekapcsolva, így ún. monitor módba állítjuk a hálókártyát. Be lehet itt állítani, hogy a Wireshark fájlba mentse el az elkapott csomagokat. Az Options-ban megadhatjuk az analízis leállításának feltételeit is, csomagszám, elkapott csomagok mérete és időkorlát alapján.

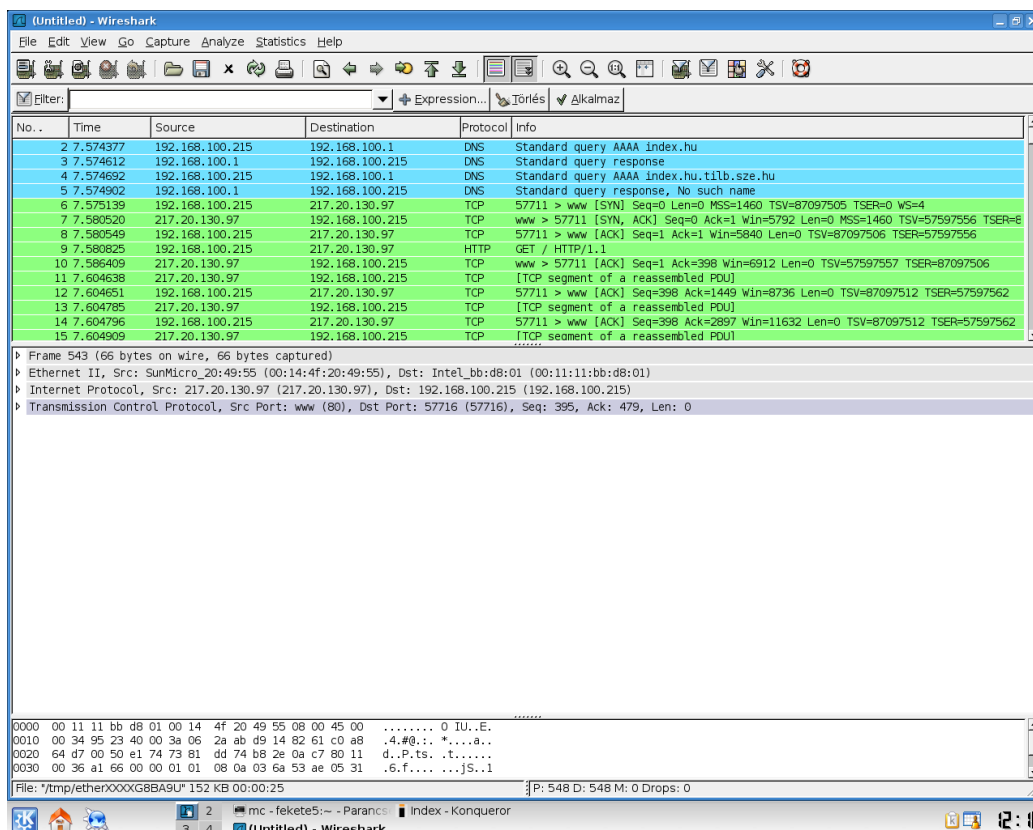
A Display options menüben lehet a csomagelkapás közbeni információkat beállítani. Automatikus „real-time” kijelzés, valamint ennek függvényében a képernyő görgetése, és az elkapott csomagok számának kijelzése.



Az utolsó részben lehet a névfeloldás lehetőségeinek beállítása, vagyis nem IP címeket kell ez esetben keresnünk, hanem az ezekhez hozzárendelt szimbolikus neveket, valamint a MAC-ben az első 3 byte helyett a gyártó neve.

## 2. feladat.

Indítsunk egy csomagelkapást az **eth0**-án, úgy hogy a leállítás feltétele legyen 1 perc, valamint a képernyő automatikusan gördüljön a csomagokkal. (amennyiben a Wireshark megkérdezi, nem kell menteni az előző listát.) Majd a böngészőt elindítva kérje le az **dev.tilb.sze.hu** honlapot.



A Wireshark az elkapott csomagok sorszámát, a forrás és cél IP-t, a protokoll nevét valamint a csomag részletét jeleníti meg első látásra. Alul látható, hogy a Wireshark a különböző protokollokat sorrendbe helyezi. Először a csomag méretét adja meg, majd az Ethernet opciókat. Itt található a forrás és cél MAC cím. Alant az IP protokoll adatai láthatóak mint a forrás és cél IP. Majd végezetül a TCP tulajdonságokat nézhetjük meg. Mint például a forrás és cél port, valamint a különböző TCP bitek értékét (SYN, ACK, FIN stb.).

Jól megfigyelhető a képen, hogy először a mi gépünk lekéri a DNS bejegyzést a névkiszolgálótól, majd megkezdí IP cím alapján a dev.tilb.sze.hu kezdőlapját letölteni.

A hálózatokon sokszor rengeteg „szemét” csomag kering, mint például feszítőfa, illetve más egyéb routing protokoll. Ha ezeket figyelmen kívül szeretnénk hagyni, a csomagszűrőkhöz kell nyúlnunk.

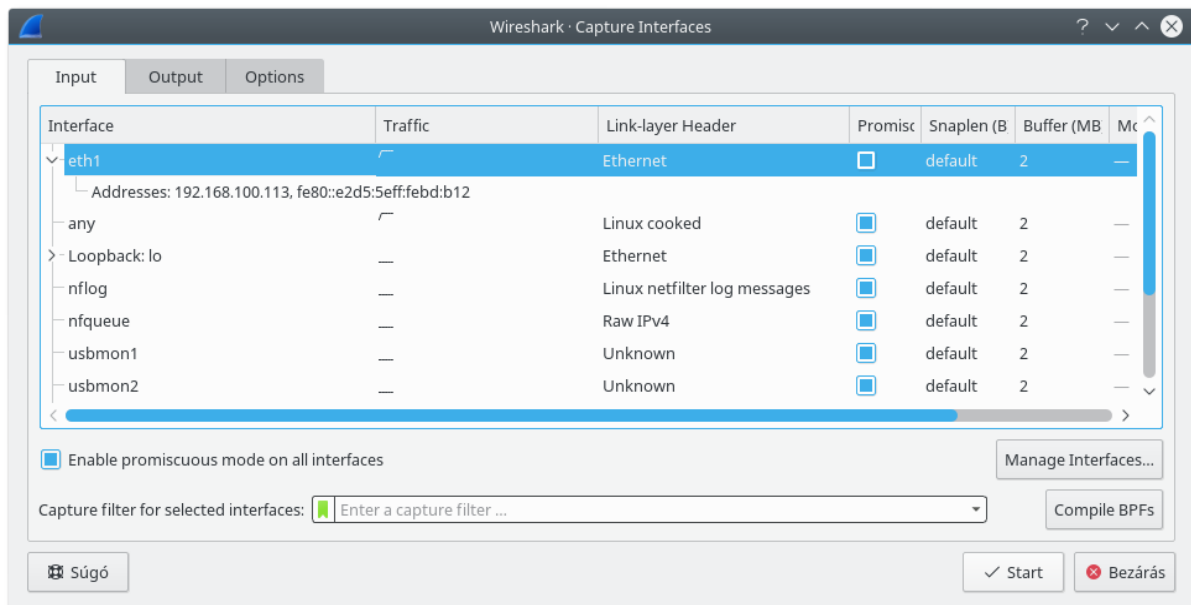
Csomagszűrők két helyen alkalmazhatók:



1. csomagelkapásnál
2. megjelenítésnél

Ha csomagok elkapásánál használunk szűrőt, akkor csak a szűrési feltételeknek megfelelő csomagokat fogja a Wireshark eltárolni. Az eltárolt csomagok közül pedig megjelenítési szűrővel választhatjuk ki, hogy melyek jelenjenek meg a képernyőn. A két fajta szűrő szintaxisa sajnos különböző!

A csomagelkapási beállításokon (4. gomb) belül lehet csomagszűrőket alkalmazni.

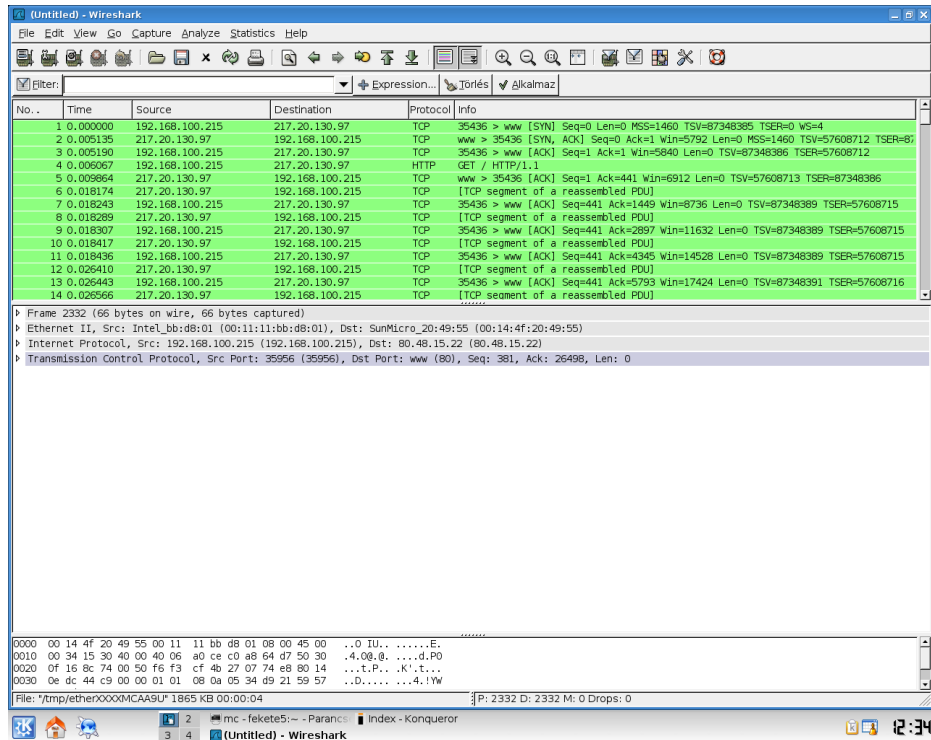


A csomagszűrési beállításokon belül több előre definiált szűrő áll rendelkezésünkre. Ezeket a kis zöld zászlóra kattintva érhetjük el.

Meg lehet adni protokollszűrést, IP cím szűrést, forrás és célport szűrést.

### 3. feladat

Hajtsuk végre az előző feladatot, úgy hogy most filterként beállítjuk, hogy csak a 80-as portot érintő kommunikációt vizsgáljuk. (*port 80*).

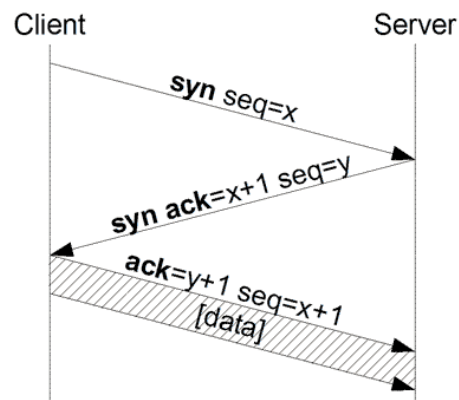


Most csak a 80-as portot érintő kommunikációt jelenítjük meg.

### 4. feladat

Hajtsuk végre az előző feladatot úgy, hogy a csomagelkapás leállításának feltétele 3 csomag elkapása legyen. Ezzel az előző feladatból csak a „three way handshake” vagyis a 3 utas kézfogást kaptuk meg.

Ez a TCP protokoll kapcsolat felépítési fázisa.





| No. | Time        | Source          | Destination     | Protocol | Lengt | Info                             |
|-----|-------------|-----------------|-----------------|----------|-------|----------------------------------|
| 1   | 0.000000000 | 192.168.100.113 | 193.224.130.179 | TCP      | 74    | 56362 → 80 [SYN] Seq=0 Win=29206 |
| 2   | 0.000819919 | 193.224.130.179 | 192.168.100.113 | TCP      | 74    | 80 → 56362 [SYN, ACK] Seq=0 Ack= |
| 3   | 0.000843543 | 192.168.100.113 | 193.224.130.179 | TCP      | 66    | 56362 → 80 [ACK] Seq=1 Ack=1 Win |

A csomagokat „kibontva” látható, hogy a 3 utas kézfogás egy TCP SYN bittel kezdődik egy sequence number=0-val, majd a szerver visszaküldi a TCP SYN,ACK bitekkel egy sequence number=0 és Acknowledge number=1-el, majd ismét válaszolunk egy TCP ACK bittel, ahol mind a sequence number mind az acknowledge number 1-re van állítva.

Természetesen ezek csak jelen helyzetben ilyen értékűek a könnyebb megértés érdekében.

Ezzel létrejött a TCP kapcsolat.

Figyeljük meg a TCP Options mezőjét azokban a csomagokban melyben a SYN bit be van állítva.

Ezekben a csomagokban a két kommunikáló fél engedélyezi egymás között a SACK opciót.

Amennyiben csak egy csomag veszik el nem tudjuk közölni a küldővel, hogy csak azt az egyet küldje újra hiszem az ACK egy előre meghatározott pontig nyugtáz.



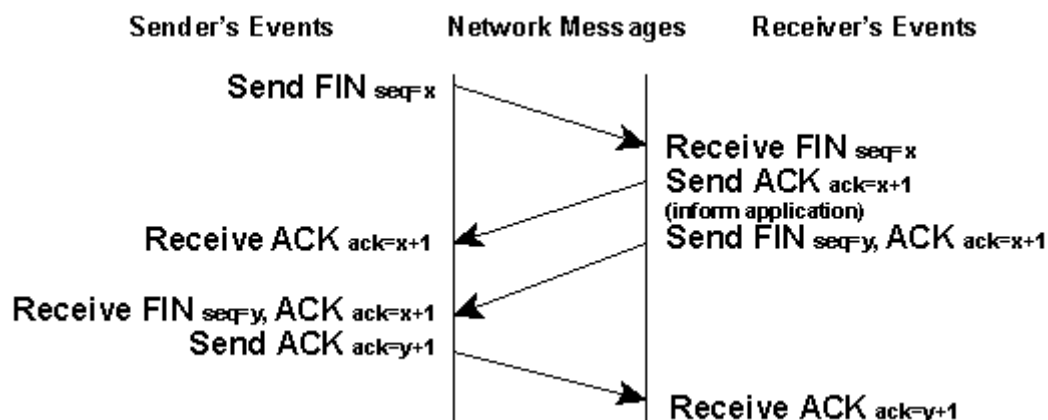
Erre való az SACK, mely segítségével megadhatjuk, hogy melyek azok a csomagok melyek megjöttek a ténylegesen várt nyugta pontjáig, és melyik/melyek azok melyek nem érkeztek meg.

### 5. feladat

Hajtsuk végre az előző feladatot úgy, hogy vegyük ki a csomagelkapás leállítási feltételt, és most is a <http://dev.tilb.sze.hu> lapot kérjük le.

| No. | Time        | Source          | Destination     | Protocol | Length | Info                             |
|-----|-------------|-----------------|-----------------|----------|--------|----------------------------------|
| 1   | 0.000000000 | 192.168.100.113 | 193.224.130.179 | TCP      | 74     | 56364 → 80 [SYN] Seq=0 Win=2920  |
| 2   | 0.000915082 | 193.224.130.179 | 192.168.100.113 | TCP      | 74     | 80 → 56364 [SYN, ACK] Seq=0 Ack= |
| 3   | 0.000925878 | 192.168.100.113 | 193.224.130.179 | TCP      | 66     | 56364 → 80 [ACK] Seq=1 Ack=1 Win |
| 4   | 0.000972532 | 192.168.100.113 | 193.224.130.179 | HTTP     | 520    | GET / HTTP/1.1                   |
| 5   | 0.001698960 | 193.224.130.179 | 192.168.100.113 | TCP      | 66     | 80 → 56364 [ACK] Seq=1 Ack=455 V |
| 6   | 0.002564159 | 193.224.130.179 | 192.168.100.113 | HTTP     | 276    | HTTP/1.1 304 Not Modified        |
| 7   | 0.002569734 | 192.168.100.113 | 193.224.130.179 | TCP      | 66     | 56364 → 80 [ACK] Seq=455 Ack=212 |
| 8   | 0.005361244 | 193.224.130.179 | 192.168.100.113 | TCP      | 66     | 80 → 56364 [FIN, ACK] Seq=211 Ac |
| 9   | 0.005493808 | 192.168.100.113 | 193.224.130.179 | TCP      | 66     | 56364 → 80 [FIN, ACK] Seq=455 Ac |
| 10  | 0.006258644 | 193.224.130.179 | 192.168.100.113 | TCP      | 66     | 80 → 56364 [ACK] Seq=212 Ack=456 |
| 11  | 8.733318945 | 192.168.100.113 | 93.184.220.29   | TCP      | 66     | 42130 → 80 [ACK] Seq=1 Ack=1 Win |
| 12  | 8.750698391 | 93.184.220.29   | 192.168.100.113 | TCP      | 66     | [TCP Acked unseen segment] 80 →  |

Itt az utolsó négy csomagban megfigyelhető a 4 utas kézfogás, mely a TCP kapcsolat lebontását jelenti. Először a szerver küld egy FIN bitet amelyre mi ACK bittel válaszolunk. Majd mi is küldünk egy FIN bitet, amelyre a szerver válaszol ACK-al.



Mivel a hálózaton (feltehetőleg) semmilyen torlódásra utaló jelet nem tapasztaltunk és a TCP protokoll sem, így a TCP FLAGS opciók között a CWR vagyis z a bit mely jelzi, hogy torlódás miatt csökkentettük az átküldhető adatok mennyiségét nem aktív.