SZÉCHENYI ISTVÁN EGYETEM
GYŐR
TÁVKÖZLÉSI TANSZÉK

Mérési utasítás **ifconfig, ping, WireShark használata**

Az ifconfig parancs

Az `ifconfig` parancs a Linux Ethernet interfészek (hálózati vezérlők) hálózati paramétereinek beállítására, és az aktuális beállítások és állapotok kiíratására szolgál. Paraméterek nélkül használva, csak az éppen aktív hálózati interfészeket sorolja fel, a legfontosabb paraméterekkel.

Például:

```
root@feher4#:ifconfig
eth0    Link encap:Ethernet  HWaddr 00:50:56:ae:00:34
        inet addr:193.224.129.168  Bcast:193.224.129.175  Mask:255.255.255.240
        inet6 addr: 2001:738:2c01:8000:250:56ff:feae:34/64 Scope:Global
        inet6 addr: fe80::250:56ff:feae:34/64 Scope:Link
        UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
        RX packets:17571 errors:0 dropped:0 overruns:0 frame:0
        TX packets:3135 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:1863396 (1.7 MiB)  TX bytes:1104162 (1.0 MiB)

eth1    Link encap:Ethernet  HWaddr 00:50:56:ae:00:35
        inet addr:10.9.0.200  Bcast:10.9.0.255  Mask:255.255.255.0
        inet6 addr: fe80::250:56ff:feae:35/64 Scope:Link
        UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
        RX packets:13415 errors:0 dropped:0 overruns:0 frame:0
        TX packets:7904 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:1741065 (1.6 MiB)  TX bytes:3278014 (3.1 MiB)

lo      Link encap:Local Loopback
        inet addr:127.0.0.1  Mask:255.0.0.0
        inet6 addr: ::1/128 Scope:Host
        UP LOOPBACK RUNNING  MTU:16436  Metric:1
        RX packets:40232 errors:0 dropped:0 overruns:0 frame:0
        TX packets:40232 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:0
        RX bytes:4792781 (4.5 MiB)  TX bytes:4792781 (4.5 MiB)
```

Ha a parancs argumentuma egy interfész, akkor csak a megadott interfészt beállításait jeleníti meg.

Például:

```
root@feher4#:ifconfig eth0
eth0    Link encap:Ethernet  HWaddr 00:50:56:ae:00:34
        inet addr:193.224.129.168  Bcast:193.224.129.175  Mask:255.255.255.240
        inet6 addr: 2001:738:2c01:8000:250:56ff:feae:34/64 Scope:Global
        inet6 addr: fe80::250:56ff:feae:34/64 Scope:Link
        UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
        RX packets:17571 errors:0 dropped:0 overruns:0 frame:0
        TX packets:3135 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:1863396 (1.7 MiB)  TX bytes:1104162 (1.0 MiB)
```

Fontos még a `-a` paraméter is. Segítségével az összes interfész (beleértve a nem aktívakat is) beállításai kiírathatóak.

Az `ifconfig` parancs segítségével lehet beállítani is az egyes interfészeket a következő szintaxis alkalmazásával:

```
ifconfig <interfész neve> <ip cím> netmask <netmaszk> up
```

Amennyiben egy interfész **down** azaz lekapcsolt állapotban van, úgy az up kapcsolóval lehet azt aktívvá tenni. A Linux képes kiszámolni a megadott információk alapján a broadcast cím értékét így azt nem kell megadnunk. Fontos megjegyezni, hogy az `ifconfig` parancs segítségével elvégzett beállítások csak a legközelebbi újraindításig maradnak meg.

Feladatok

1. Indítsa el a terminál (konsole) programot, mely ekvivalens azzal a terminállal amiben eddig dolgozott.
   K menü -> Alkalmazások-> Rendszer -> Terminál
2. Írassa ki az összes interfészt, és állapotukat! Hány interfészt lát?
   ```
   ifconfig –a
   ```
3. Állítsa be a fekete gép eth0 interfészének a következő IP címet:192.168.100.<190+gépszám> (netmaszk: 255.255.255.0), majd adja meg az alapértelmezett hálózati átjárót.
   ```
   ifdown eth0
   ifconfig eth0 192.168.100.190+gépszám netmask 255.255.255.0 up
   route add default gw 192.168.100.1
   ```
4. Ellenőrizze, hogy sikerült-e beállítani az IP címet!
   ```
   ifconfig eth0
   ```

A ping parancs

A `ping` parancs a hálózati kapcsolat tesztelésére szolgál. Segítségével ICMP echo üzenetet (lásd tankönyv) lehet küldeni a paraméterben megadott hosztnak. Linux esetében a ping parancs a csomagokat addig küldi a megadott hosztnak, míg leállításra nem kerül. Ezt a <CTRL>+<C> billentyűkombinációval lehet megtenni.

Például:

```
root@cloud:~# ping 10.9.0.1
PING 10.9.0.1 (10.9.0.1) 56(84) bytes of data.
64 bytes from 10.9.0.1: icmp_req=1 ttl=64 time=1.79 ms
64 bytes from 10.9.0.1: icmp_req=2 ttl=64 time=0.292 ms
^C
--- 10.9.0.1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 0.292/1.042/1.793/0.751 ms
```

Lehetőség van megadni, hogy a parancs pontosan hány ICMP echo request üzenetet küldjön a célállomás felé. (Hányszor „pingeljük” meg az adott hosztot.) Ezt a `-c` kapcsoló segítségével lehet megtenni.

Például:

```
root@cloud:~# ping -c 1 10.9.0.1
PING 10.9.0.1 (10.9.0.1) 56(84) bytes of data.
64 bytes from 10.9.0.1: icmp_req=1 ttl=64 time=0.354 ms

--- 10.9.0.1 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.354/0.354/0.354/0.000 ms
```

Feladatok

1. Kérdezze le a `ping` parancs paramétereit, majd tanulmányozza azokat!
   `(ping)`
2. A ping parancs segítségével küldjön pontosan 10 ICMP ehco request üzenetet a 192.168.100.1-es IP címre. Tanulmányozza az eredményeket! Milyen átlagos és maximális válaszidőket tapasztalt?
   `(ping -c 10 192.168.100.1)`

WireShark program alkalmazása

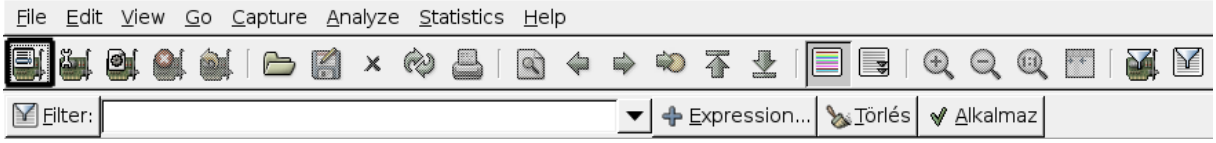
A WireShark (korábbi nevén Ethereal) egy rendkívül fejlett hálózati sniffer és analizátor program. Fejlesztése 1998-óta folyik, jelenleg GPL 2 licensz alatt. Nem igen találni ilyen széleskörű szolgáltatásokkal és ismeretekkel rendelkező hálózati analizátor programot. Támogatott operációs rendszerek: 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 analizálni. A program által jelenleg ismert protokollok száma több mint 81000, de bárki készíthet hozzá újabbakat.

A WireShark analizátor funkcióit több könyv, illetve elektronikus irodalom írja le több száz oldal terjedelemben, így a gyakorlat keretében csak az alap funkciókkal ismerkedünk meg.

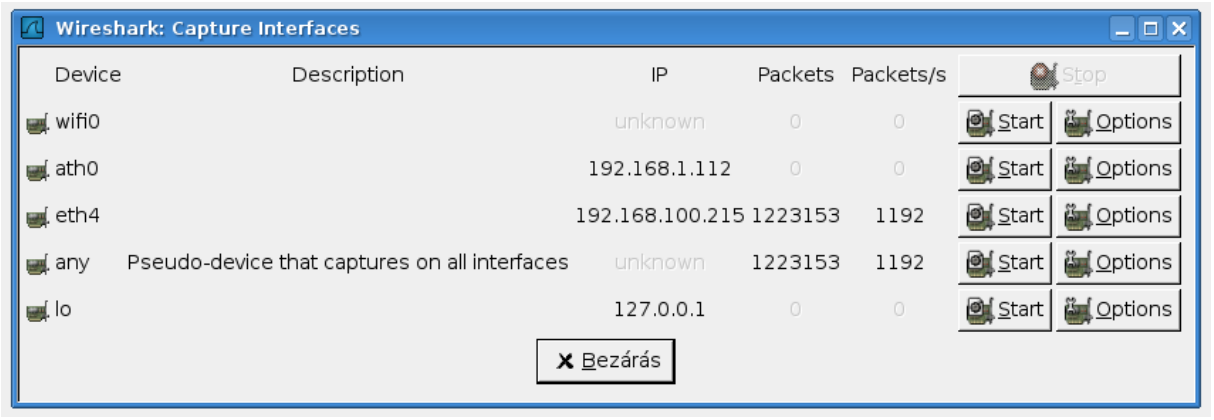
Feladatok

1. Amennyiben nincs telepítve a számítógépre, telepítse a WireSharkot az `apt-get install wireshark` parancs kiadásával!

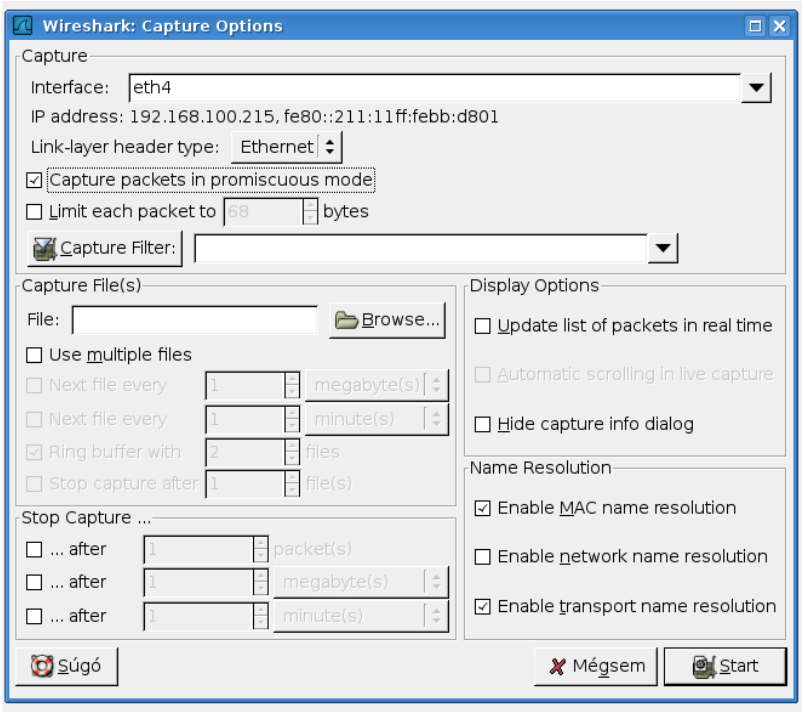
2. Indítsa el a programot, majd nézze át annak kezelőfelületét!



Az első gombbal hívhatja elő a WireShark által elérhető és használható hálózati interfészeket.



Ezen az ábrán láthatóak a „sniffelhető” interfészek, IP címeikkel, és az eddig továbbított csomagok számával. A második gombbal állíthatja be az analizálás tulajdonságait.



Legfelül látható, hogy jelen esetben az eth4 interfészt használjuk. A „Capture packet in promiscuous mode” kapcsoló ún. monitor módba állítja a hálózati kártyát, melyben olyan csomagokat is el tudunk kapni, melyek nem a mi számítógépünknek szólnak. Ezt a kapcsolót mindig kapcsoljuk be a mérések során. Be lehet állítani azt is, hogy a WireShark fájlba mentse el az elkapott csomagokat. Megadhatja az analizálás leállásának 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 is itt állítható be.

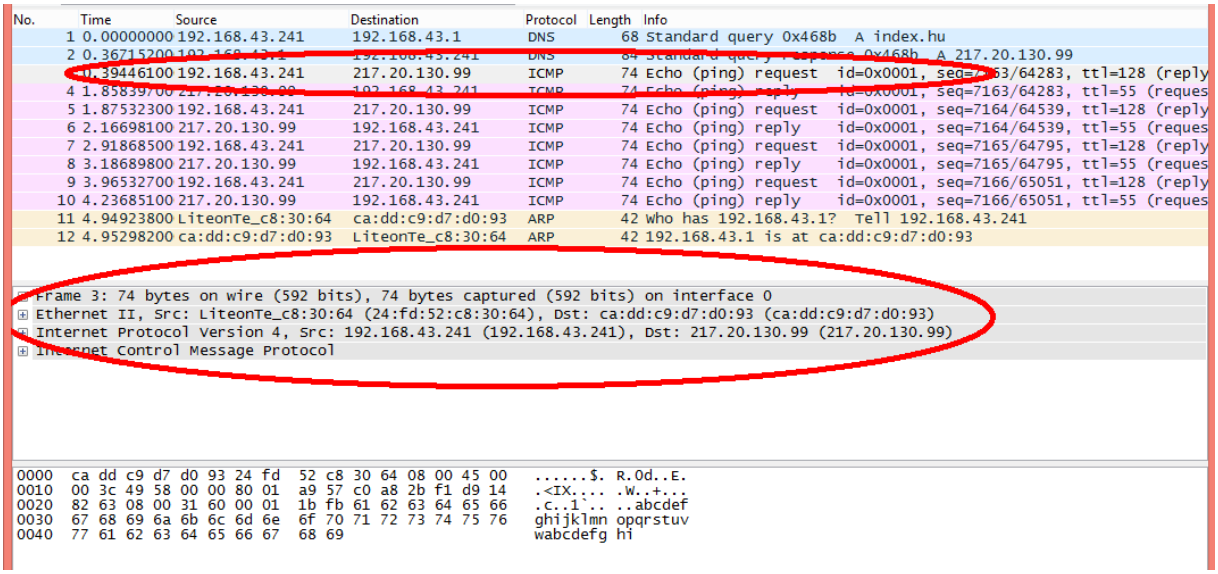
Az utolsó részben lehet beállítani a névfeloldás működését, ekkor nem IP címeket lehet látni, hanem az ezekhez hozzárendelt szimbolikus neveket, valamint a MAC címben az első 3 byte helyett a gyártó neve jelenik meg.

A következő két gomb a csomag elkapás indítása, illetve leállítása.

3. ICMP vizsgálata

Indítson egy csomagelkapást az eth0 interfészen! (Ha a program megkérdezné, úgy nincs szükség az előző adatok mentésére.) Miközben a WireShark az adatokat gyűjti, adjon ki egy `ping -c 4 index.hu` parancsot! Miután a `ping` parancs végzett, állítsa le a WireShark programban a csomagok elkapását! Elemezze ki az eredményeket!

A WireShark az elkapott keretek sorszámát, a forrás és cél IP címet, a protokoll nevét valamint a csomag részletét jeleníti meg első látásra. A program felső ablakában keresse meg az első ICMP Echo request csomagot, és válassza ki azt! Ekkor a középső ablakban megjelenik a kiválasztott csomaghoz tartozó keret, ahol részletesebben is elemezheti azt. Vegye észre, hogy az egyes hálózati rétegek jól megfigyelhetőek ebben a középső ablakban! A legelső sor a keret paramétereit tartalmazza, majd alatta az Ethernet keret, az arra épülő IP, legvégül pedig az ICMP.



A sorok elején lévő + jel segítségével az egyes szinteket is alaposabban kielemezheti! (Miközben a program középső részében kijelöl valamit, az a program alsó részében is kiemelten jelenik meg.)

Milyen hosszú egy `ping` parancshoz tartozó keret?

Keresse meg, a vizsgált csomag feladójának fizikai címét! Melyik szinten fogja ezt keresni? Milyen címzési módot talált?

Keresse meg, hogy mi adja meg, hogy IP datagram van a vizsgált Frame-ben! Milyen értéket talált? Hány byte ez az érték?

Keresse meg, hogy milyen forrás IP címről, milyen IP célcímre ment az IP datagram. Melyik szinten fogja ezt keresni? Hány byte hosszú egy cím? Milyen TTL értéket talált?

Keresse meg, hogy mi adja meg, hogy ICMP protokoll van a vizsgált IP datagramban! Milyen értéket talált? Hány byte ez az érték?

Keresse meg, hogy mi adja meg, hogy az ICMP-ben Echo Request van! Milyen értéket talált? Hány byte ez az érték? Melyik szinten fogja ezt keresni?

Hány byte a payload az elfogott ICMP kérésben? Milyen értékűek ezek a byte-ok? Hol találta meg ezt a payloadot?

Most hasonlóan vizsgálja meg az ICMP Echo Replay-t is!

4. UDP vizsgálata

Vizsgálja meg az előző feladatban végrehajtott csomagelkapásban még az ICMP üzenetek előtt elkapott DNS Standard Query üzenetet! Milyen forrás fizikai címről ment a kérés? Milyen forrás IP címről, milyen cél IP címre ment a kérés?

Milyen szállítási szintű protokollt használt az DNS? Milyen érték azonosítja az UDP-t?

Hány byte egy UDP checksum? Hol találta meg?

Milyen forrásportról ment a kérés milyen célportra? Melyik szinten, hol találta meg az értékeket? Mi a DNS kérések portszáma?

Valahol megtalálta a kért nevet? (index.hu) Hol? Melyik szinten?

5. TCP vizsgálata és szűrők

Indítson egy csomagelkapást az eth0-án, úgy, hogy a leállítás feltétele legyen 1 perc, valamint a képernyő automatikusan gördüljön a csomagokkal. (Nem kell menteni az előző listát.) Ezután a böngészőt elindítva kérje le az *index.hu* honlapot.

A TCP protokollt a következő gyakorlaton fogjuk részletesen megismerni, most csak az első TCP szegmens tartalmát vizsgáljuk meg.

Az TCP adategységet szállító IP datagram mely mezőjének milyen értékéből derül ki, hogy a TCP protokoll adategysége utazik fölötte?

A TCP protokoll fejrészében keresse meg a cél port számát!

A hálózatokon sokszor rengeteg „minket nem érdeklő” forgalom van. Ha ezeket figyelmen kívül szeretnénk hagyni, a csomagszűrőkhöz kell nyúlnunk.

SZÉCHENYI ISTVÁN EGYETEM
GYŐR
TÁVKÖZLÉSI TANSZÉK

Csomagszűrők két helyen alkalmazhatók:

1. csomagelkapásnál
2. megjelenítésnél

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

A csomagelkapási beállításokon (2. gomb) belül lehet csomagszűrőket alkalmazni. (A csomagszűrési beállításokon belül több előre definiált szűrő áll rendelkezésünkre. Meg lehet adni protokollszűrést, IP cím szűrést, forrás és célport szűrést. Bővebben majd a következő gyakorlaton foglalkozunk vele.)

Most csak a megjelenítésnél használható szűrőket (Display Filter) ismerjük meg. A korábban megismert gombsor alatt a „Filter:” mezőben adjuk meg a következőt: tcp.port == 80

Vegyük észre, hogy gépelés közben a mező háttere zöldre vált, amikor az addig begépelt szöveg szintaktikailag helyes!

Érvényesítsük a szűrőt az Apply feliratra való kattintással.

Mit tapasztalunk?