Infokommunikációs Laboratórium 3. mérés

Wireshark mérés

(TTMER104)

Beugrók mintakérdések (azt mondta nem lesz beugró, de készüljünk fel)

(Hibát tartalmazhat!)

1. Milyen feladatokat kell megoldani a mérési gyakorlaton?

DNS lekérdezés és válasz

webes forgalom elemzése: teljes oldalletöltés vizsgálata, TCP szintű elemzés

VoIP hívás elemzés: hívás vizsgálata, hívás közben átvitt médiafolyam vizsg.

Nagy mennyiségű adatátvitel vizsgálat: hálózati paraméterek, átviteli karakterisztika vizsgálat

2. Ismertess vázlatosan a DNS működését!

A DNS az Interneten használatos hierarchikus névrendszer. Működése:

A DNS egy elosztott adatbázis alapján működik és kliend-szerver modellt használ. A hierarchia csúcsát a root nameserverek szolgálják ki. Ezek alatt állnak az egyes subdomainek authoritatve DNS szerverei. A kliens oldal (DNS resolver) kezdeményezi a lekérdezések sorozatát, amely végül a teljes feloldáshoz vezet. A DNS lekérdezés lehet rekurzív vagy nemrekurzív. Az utóbbi esetben a DNS szerver olyan rekordot ad meg, amelynek a feloldásáért lehet rekurzív vagy nemrekurzív. Az utóbbi esetben a DNS szerver olyan rekordot ad meg, amelynek a feloldásáért maga illetékes. A rekurzív esetben a DNS szerver más DNS-lekérdezések után tudja megválaszolni a kérést.

link: https://smartcomlab.tmit.bme.hu/alpha/meresek/lanuser.htm#dns

3. Adja meg a TCP/IP protokoll stack rétegeit! Mi az egyes rétegek feladata?

TCP/IP rétegek:

- Physical PHY – fizikai réteg
- Data Link Control – Adatkapcsolati réteg
- Network – Hálózati réteg
- Transport – Szállítási réteg
- Application – Alkalmazási réteg

(Gondolom ez, de jó lenne, ha valahol értelmesen le lenne írva….)

OSI layer		TCP/IP layer
7 6 5	Application: Alkalmazási réteg DHCP, DNS, RTP, SIP, SNMP, HTTP	5
4	Transport: Szállítási réteg TCP, UDP	4
3-4	Routing Protocols RIP (Routing Information Protocol) BGP,EGP,GGP (Gateway Protocols) IGRP OSPF	3
3	Network: Hálózati réteg IP, IPX	3
2	Data Link Control DLC: Adatkapcsolati réteg alrétegek: MAC, LLC	2
1	Physical PHY: Fizikai réteg (Ethernet 10/100/1000BASE-T)	1

4. Ismertesse az Ethernet keret szerkezetét! Mekkora lehet a keret maximális mérete?

Maximális keret: 1518 byte

6 DA – Cél MAC címe

6 SA – Forrás Mac Címe

2 PT – Típus hossz

16-1000 C – data

+ PAD

A jelenlegi Interneten az IP általában Ethernet felett fut (pontosabban EthernetII keretekbe csomagoljuk az IP csomagokat), amelynek MTU-ja 1500 byte. Bizonyos eszközgyártók lehetővé teszik ún. Jumbo-frame-ek használatát, amelyek esetén az MTU 9000 byte is lehet, ez azonban csak LAN-okon fordulhat elő, border protokollok (pl. PPPoE) ezt redukálják a szokásos méretre. WLAN-on az MTU értéke 2272 byte.

link: https://smartcomlab.tmit.bme.hu/alpha/meresek/lantcp.htm#tcpsetup

5. Ismertesse az IP fejléc szerkezetét! Mekkora lehet az IP csomag maximális mérete?

IPv4 keret fejléce

```
bytes (20)   from basic MAC and tagged MAC
   +--------------------------------+
 1 | Verzió (4bit)  - IHL (4bit)    |  45  (5x32 bit) min hossz:5
   +--------------------------------+
 1 | Szolgálattípus                 |  00
   +--------------------------------+
 2 | Total length                   |  05DC = 1500 bytes
   | network order: első byte msb   |  max. length: 65536 bytes
   +--------------------------------+
 2 | Identification                 |  abcd = 43981
   +--------------------------------+
 2 | flag (3 bit) +                 |  00 00
   | fragment offset (13 bit)       |
   +--------------------------------+
 1 | élettartam                     |  00
   +--------------------------------+
 1 | protokoll                      |  04  továbbított (szállítási) protokoll (0x06 - TCP, 0x11 - UDP) kódok: RFC 790
   +--------------------------------+
 2 | IP header checksum             |  0bb5
   +--------------------------------+
 4 | source IP address              |  01010102
   +--------------------------------+
 4 | destination IP address         |  02020202
   +--------------------------------+
 4 | options (3byte) padding (1byte)|
   +--------------------------------+
```

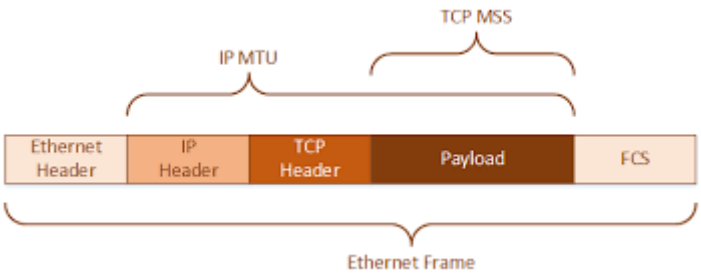
1.Bájt	2. Bájt	3. Bájt	4. Bájt	
Verzió szám (4)	Hossz (5)	Szolgáltatás típus (0)	Csomag hossza	1. Szó
Egyedi azonosító szám		Jelzőbitek és tördelési letolás (fragment offset)		2. Szó
Életben maradási idő	Szállított típus (TCP=6, UDP=17)	Fejléc ellenőrző összeg		3. Szó
Forrás (Feladó) IP címe				4. Szó
Cél (Címzett) IP címe				5. Szó
Opciók			Kitöltés	6. Szó

5 . ábra Az IP fejléc szerkezete

(Egy maximális értéket megadni a felkészülési anyagban luxus….)

6. Mi a különbség az MSS és a MTU között?

MSS olyan, mint a MTU, csak 4. rétegen TCP-vel használva. MSS a maximum méret, amit a csomag elfoglalhat, azután, hogy a csomag maximális méretéből levontuk az IP, TCP és más fejléceket. Vagyis, ha MTU 1500 byte, illetve az IP és TCO fejlécek 20 byte-osak, akkor MSS 1460 byte.



Tehát, MSS függ az MTU-tól, mivel MTU-ra igyekszünk választani a csomagok méretét és ezekből a csomagokból MSS-nyi adatot tudunk kapni, vagyis a maximális átküldött adat/csomag-ot adja.

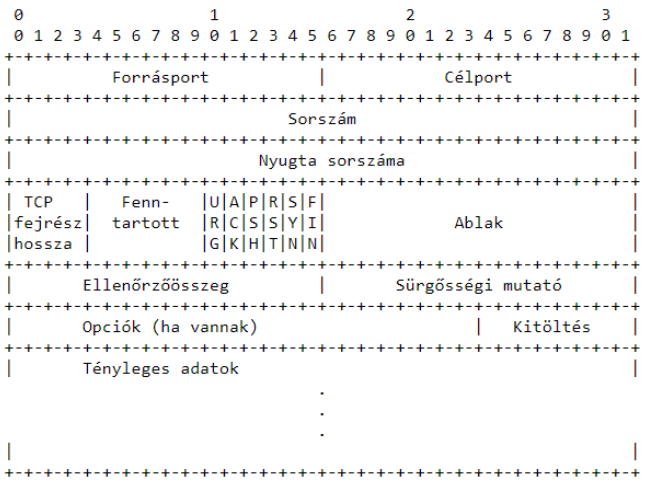
7. Ismertesse a TCP szegmens fejlécnek szerkezetét! Mekkora lehet a TCP szegmens maximális mérete?

A fejléc a fentiek szerint legalább 20 oktetből áll…. DE HOGY EZ MENNYI AZT NE ÍRJÁTOK BELE A JEGYZETBE……..

Forrásport + Célport + Sorszám + Nyugta sorszáma + TCP fejrész hossza + Fenntartott + ablak + Ellenőrzőösszeg + Sűrgősségi mutató + Opciók + kitöltés =

16 bit + 16 bit + 32 bit + 32 bit + 60 byte + 16 bit + 6 bit + 6 bit + 16 bit + 8 bit + PAD = 80 byte (??? ez nem biztos, hogy jó)

A TCP szegmens fejlécének szerkezete



8. Soroljon fel 5 eltérő tulajdonságot a TCP és az UDP között!

TCP | UDP

lassú | gyors

megbízható | kevésbé megbízható

20 byte | 8 byte

Elvesztett adatokat újra kéri | nem foglalkozik vele

(Kolléga jóvoltából)

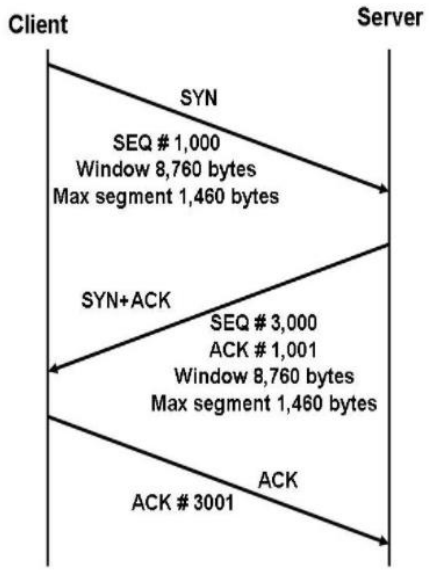
9. Mik azok a portszámok, mire használatosak?

A két végpontot azok IP-száma, míg a rajtuk futó, egymással kommunikálni kívánó alkalmazásokat a portszámok (amelyekkel tulajdonképpen interfészket jelölünk meg) azonosítják. Ez az azonosítás globálisan egyedi. Érdemes megjegyezni, hogy (i) ez az azonosító változhat az átviteli út különböző részein (lásd: NAT), illetve (ii) az azonosító egyes elemei különböző rétegekben használatosak.

link: https://smartcomlab.tmit.bme.hu/alpha/meresek/lantcp.htm#tcpservices

10.Ismertesse a TCP kapcsolatfelépítési folyamatát!

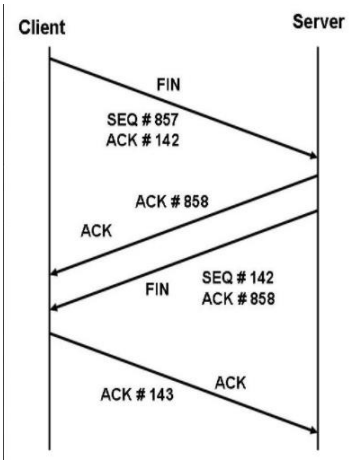
Minthogy a TCP összeköttetés orientált (connection oriented) hálózati rétegbeli protokoll, így mielőtt a TCP felhasználásával adatot szeretnénk átvinni két különböző számítógépen futó alkalmazás között, kapcsolatot kell felépítenünk. A TCP kapcsolatot kezdeményező felet kliensnek (client host) nevezzük, míg a másik felet kiszolgálónak (server host) hívjuk. A TCP kapcsolat létrehozását - amit gyakran hívnak angol terminológiával 3-way-handshake-nek - az alábbi ábra szemléleti:



link: https://smartcomlab.tmit.bme.hu/alpha/meresek/lantcp.htm#tcpsetup

11.Ismertesse a TCP kapcsolatlezárási folyamatát!

A TCP adatfolyam utolsó byte-jának átvitele után az összeköttetést le kell zárnunk. A lezárás a két fél által kölcsönösen elküldött FIN jelzőbites szegmensekből és az arra adott nyugtákból áll, vagyis alapvetően 4 üzenetet használunk, ami nem is meglepő, ha arra gondolunk, hogy a TCP full-duplex összeköttetést nyújt. Egy lehetséges esetet mutat be az alábbi ábra. Érdemes megjegyezni, hogy az állapotgráfban nem azonnal a második nyugta után kerülünk CLOSED állapotba, hanem még ki kell várni egy időzítő lejártát. (Asszem maga az ábra is elég kell legyen….)



link: https://smartcomlab.tmit.bme.hu/alpha/meresek/lantcp.htm#tcpclose

12. Hogyan működik a TCP-ben a folyamatvezérlés? Hogyan számítjuk a küldő ablakát?

(TCP forgalomszabályozás) megakadályozza, hogy egy gyors küldő elárassza a vevőt (folyamvezérlés - flow control)

Az egyidőben nyugtázatlan adatmennyiséget tehát az "Ablak" hirdeti meg (Advertised Window). Ennek változtatásával a célpont egyértelműen a forrás tudtára adhatja, hogy mennyi adatot képes még fogadni, vagyis megakadályozza, hogy egy gyors küldő elárassza a vevőt (folyamvezérlés - flow control).

Amikor a csomagok sorrendben, hibátlanul érkeznek, akkor az ablak mérete tulajdonképpen állandó (vagy csak kissé ingadozik a késleltetett nyugta miatt), ugyanakkor a kezdőpontja folyamatosan emelkedik ("csúszik" - innen az elnevezés).

Erre az önszabályozó jellegre az angol termiológiában a "self clocking" kifejezéssel hivatkoznak.

Ha nem sorrendhelyesen érkeznek a csomagok, akkor a nyugta sorszáma (vagyis az ablak kezdőpontja) változatlan, ugyanakkor a mérete csökken, hiszen a sorrenden kívül érkezett csomagokat eltároljuk (amennyiben az ablakon belülre esnek, egyébként eldobjuk).

Ugyancsak eldobjuk a duplikált csomagokat. Ilyen módon, a csúszó ablakkal ellátott vevőpuffer szolgál a szegmensek sorrendjének helyreállítására. Az adatfolyam elveszett részeinek helye üresen marad az ablakban, a nyugtákkal jelezzük a hiányt és a pótlólag megérkező szegmensekről kumulatív nyugtát küldünk, így lesz az adatátvitel megbízható. A csúszó ablak (Sliding Window) használatát könnyen megérthetjük az alábbi ábrából és képletekből.

küldő ablakának számítása (nem találtam meg):

13. Mire használatos az ún. „Window Scale” TCP opció?

Ablak (Window) - A TCP forgalomszabályozása során használt változó, megmondja, hogy a vevő mennyi adatot képes még fogadni. A lehetséges 65535 byte nem túl nagy, a "Window Scale" opció segítségével növelhető.

link: https://smartcomlab.tmit.bme.hu/alpha/meresek/lantcp.htm#tcpseqnack

14. Mire használatos az ún. „Timestamp” TCP opció?

Timestamp (Időbélyeg) opció használatával lehetőségünk van arra, hogy akár nagy ablakméret mellett is megkülönböztessünk azonos sorszámmal érkező szegmenseket.

Fontos tudnunk, hogy az időbélyeg időegységeket számol, a számlálót a SYN-SYN/ACK üzenetváltás során tudjuk inicializálni és szinkronizálni. Az időalap megválasztásával kacsolatban a szabvány szerint 1 msec és 1 sec közötti értéket kell használni, a gyakorlatban néhányszor 10 és néhányszor 100 msec közötti időalapot szoktak választani. Ez az Időbélyeg opció másik célra való használatára megfelelő, ugyanis arra is lehetőségünk van az időbélyeg visszaküldésével, hogy az RTT-t viszonylag pontosan

mérni tudjuk. A mérés eredményét mind az RTO számítására, mind a torlódás előrejelzésére használhatjuk.

link: https://smartcomlab.tmit.bme.hu/alpha/meresek/lantcp.htm#tcpseqnack

15.Mi a pszeudófejléc és mi a szerepe?

A számításnál egy úgynevezett pszeudófejlécet is figyelembe veszünk, ami sorrendben a forrás és a célpont IP címét, egy csupa 0 byte-ot, a belső protokoll jelzését 8 biten (jelen esetben értéke 6 a TCP miatt) és a TCP szegmens hosszát 16 biten ábrázolva. Ezt a 96 bitet a TCP szegmens elé tesszük, az ellenőrző összeg mezőjét csupa 0-val töltjük fel, majd elvégezzük az összegzést és az eredményt beírjuk az ellenőrző összeg mezőjébe.

link: https://smartcomlab.tmit.bme.hu/alpha/meresek/lantcp.htm#tcpheader

16.Hogyan működik a kumulatív nyugtázás?

Elveszett csomagok újraküldése - a célcsomópont kumulatív nyugtát küld, a nem nyugtázott adatokat újraküldi a forrás.

Az adatfolyam elveszett részeinek helye üresen marad az ablakban, a nyugtákkal jelezzük a hiányt és a pótlólag megérkező szegmensekről kumulatív nyugtát küldünk, így lesz az adatátvitel megbízható.

link: https://smartcomlab.tmit.bme.hu/alpha/meresek/lantcp.htm#tcpflowctrl

17.Hogyan működik a szelektív nyugtázás?

Lényeg, amit kiszedtem az angol szövegből:

Előfordulhat, hogy abnormális, de kártalan események túl terhelik a hibát naplózó fájlokat, amit elkerülhetünk, ha „körkörösen” naplózunk, vagy csak bizonyos ismert hibákat naplózunk, vagyis szelektíven naplózunk/nyugtázunk.

Angol szöveg:

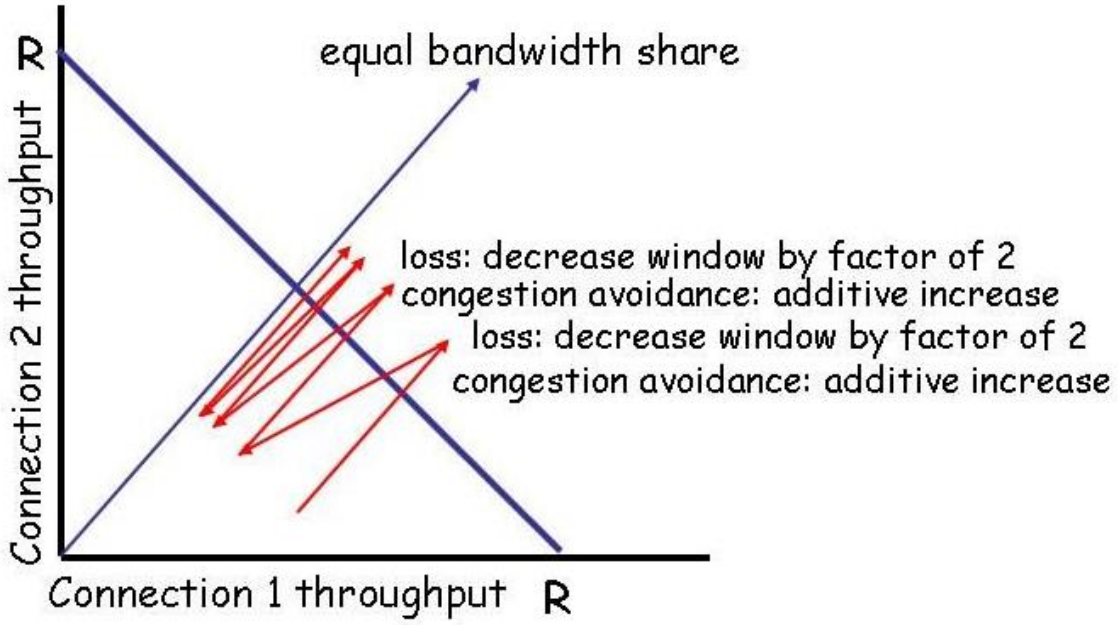
There is a tendency for abnormal but harmless protocol events to overflow error logging files; this can be avoided by using a "circular" log, or by enabling logging only while diagnosing a known failure. It may be useful to filter and count duplicate successive messages. One strategy that seems to work well is: (1) always count abnormalities and make such counts accessible through the management protocol (see [INTRO:1]); and (2) allow the logging of a great variety of events to be selectively enabled. For example, it might useful to be able to "log everything" or to "log everything for host X".

link: https://datatracker.ietf.org/doc/html/rfc1122 (13-14 oldal)

18.Mit jelent az Additive Increase/Multiplicative Decrease? Rajzolja le két versengő TCP folyamra!

Ha egy szűk linken több TCP folyam is osztozik, akkor elvárhatjuk, hogy azok a kezdeti paramétereiktől függetlenül igazságosan osztozhassanak az erőforrásokon. Ezt a torlódás elkerülésnél használatos Additive Increase/Multiplicative Decrease sebességszabályozás biztosítja, amint ez az alábbi ábráról megérthető.

link: https://smartcomlab.tmit.bme.hu/alpha/meresek/lantcp.htm#tcpfairness



19.Mit jelent a TCP esetén a „slow start” (lassú indulás)?

Slow start

Ebben a fázisban a torlódási ablak minden nyugta után egy MSS-sel növekszik, ami a gyakorlatban a cwnd megduplázódását jelenti RTT-ként. (A késleltetett nyugtázás ezt a növekedési ütemet természetesen lassítja.) A fázis addig tart, amíg a küldési ablak az SSthresh értékét el nem éri (természetesen lehetséges, hogy sosem történik ez meg, hiszen küldés sebességét a vevő is tudja korlátozni, a küldési ablak az AdvWindow és a cwnd minimuma) vagy csomagvesztést nem tapasztalunk. Az újonnan felépülő TCP összeköttetések esetén a cwnd értékét 1 szegmensre, míg az ssthresh értékét 65535 byte-ra szoktuk állítani. Ha a küldési ablak eléri az ssthresh értékét, akkor a torlódás elkerülés fázisába lépünk át. Ha csomagvesztést tapasztalunk 3 duplikált nyugta alapján, akkor a gyors újraküldési fázisba lépünk át az aktuális küldési ablak felét tekintve új ssthresh-nak és az a cwnd értékét az ssthresh-ra állítjuk (Multiplicative Decrease), ha az újraküldési időzítő lejár, akkor a lassú indulás fázisát kezdjük újra, úgy hogy az ssthresh értéke az aktuális küldési ablak fele lesz, míg a cwnd értékét 1 MSS-re állítjuk.

link: https://smartcomlab.tmit.bme.hu/alpha/meresek/lantcp.htm#tcpcongestionctrl

20.Mit jelent a TCP esetén a „congestion avoidence” (torlódás elkerülés)?

A torlódás elkerülés fázist azért használjuk, hogy a hálózaton elérhető legnagyobb sebességet minél kisebb ingadozásokkal keressük meg. A lassú indulás exponenciális növekedésével szemben itt RTT-

ként lineárisan nő a cwnd. Amikor ebbe a fázisba lépünk, az ssthresh értéke a fele annak a küldési ablakénak, amelyben a torlódást észleltük, míg a cwnd értékét az ssthresh-ra állítjuk. Ezek után minden újonnan érkező ACk-ra a cwnd értékét MSS*MSS/cwnd-vel növeljük. Ez praktikusan RTT-nként 1 MSS-sel növeli a cwnd-t (Additive Increase). Ha újabb csomagvesztés történik és ezt a duplikált nyugták alapján észleljük, akkor a gyors újraküldési fázisba lépünk át az aktuális küldési ablak felét tekintve új ssthresh-nak és az a cwnd értékét az ssthresh-ra állítjuk (Multiplicative Decrease), ha az újraküldési időzítő lejár, akkor a lassú indulás fázisát kezdjük újra, úgy hogy az ssthresh értéke az aktuális küldési ablak fele lesz, míg a cwnd értékét 1 MSS-re állítjuk.

link: https://smartcomlab.tmit.bme.hu/alpha/meresek/lantcp.htm#tcpcongestionctrl

21. Milyen lesz a TCP összeköttetés hosszútávú képe, ha a torlódás korlátozza az átvitelt? Rajzoljon! Hogyan határozná meg a várható átlagsebességet (throughput)?

A TCP átlagsebességére vonatkozólag is egy egyszerű becslést tehetünk: 0.75*cwnd/RTT. Természetesen a becslés akkor igaz, ha a torlódási ablak korlátozza az átvitelt és az átviendő byte-folyam elég hosszú ahhoz, hogy a lassú indulás fázis hossza elhanyagolható legyen a torlódás elkerülés fáziséhoz képest. A fűrészfog görbe nem csak a sebességet jellemzi, hanem alakja megfeleltethető a cwnd méretének, illetve a vevőpufferben található adatmennyiségnek is.

link: https://smartcomlab.tmit.bme.hu/alpha/meresek/lantcp.htm#tcpfairness

22. Mire használatos az RTP protokoll? Mit nyújt a felhasználó számára?

Real Time Transport protokoll:

Az RTP csomagok digitalizált hang és kép valós idejű továbbítására szolgálnak az Interneten keresztül.