Introduction to Computer Networks

Practice #1

Hasidi Netanel

- **What is the Objective of Networking?**

  Enable communication between applications on different computers.

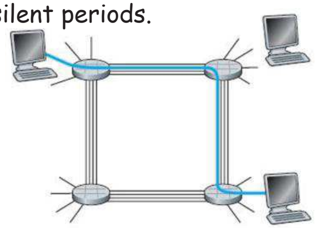
  Two computers are said to be interconnected if they are able to exchange information.

- The Internet is a collection of interconnected networks.
- **End systems** are connected together by a network of communication links.
- **End systems** access the internet thorough ISPs.

2 ways to share:

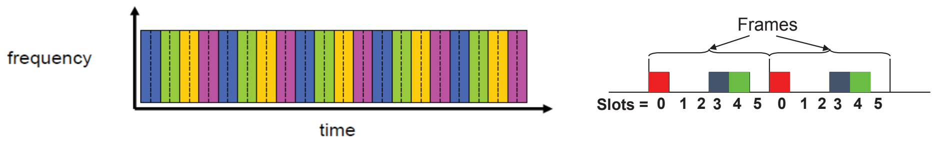
Circuit Switching

- It's the method used by telephone network.
- All resources (e.g. communication links) needed by call dedicated to that call.
- A call has three phases:
  1. Establish circuit from end-to-end ("dialing").
  2. Communicate.
  3. Close circuit ("tear down").
- "Wasteful" – dedicated circuits are idle during silent periods.

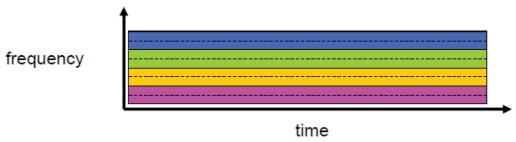


Sharing a circuit:

1)TDM - Time divided into frames and frames divided into slots.



2)FDM – The bandwidth is divided and each connection gets its own band in the frequency spectrum.

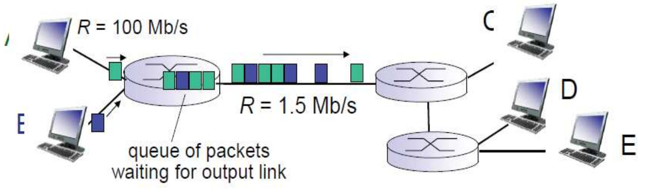


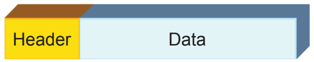
Differences Between Circuit & Packet Switching

Circuit-switching	Packet-Switching
Guaranteed capacity	No guarantees (best effort)
Capacity is wasted if data is bursty	More efficient
Before sending data establishes a path	Send data immediately
All data in a single flow follow one path	Different packets might follow different paths
No reordering; constant delay; no pkt drops	Packets may be reordered, delayed, or dropped

Packet Switching

- Used in the Internet
- Data is sent in **Packets** (header contains control info, e.g., source and destination addresses)





- Per-packet routing
- At each node the entire packet is received, stored, and then forwarded (**store-and-forward**)
- Packets in a flow may not follow the same path. (keep in mind for the rest of the course)
- **No capacity is allocated**

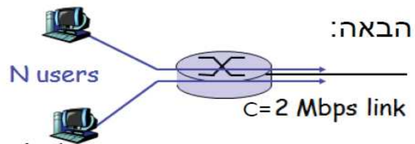
Queuing and loss:
if arrival rate (in bits) to links exceeds transmission rate of link for a period of time:
- packet will queue, **wait** to be transmitted on link.
- packets can be **dropped** (lost) if memory (buffer) fills up.

Q1

- כעת הניחו שמשתמשים ב-Packet switching ו-N=40
- ג. מהי ההסתברות שבדיוק M (מסעיף א') מתוך N משתמשים פעילים?
- ד. מהי ההסתברות שהדרישה עולה על קיבולת הערוץ?
- ה. מה התפוקה הממוצעת של הערוץ?

Q1

- נתונה הרשת הבאה:



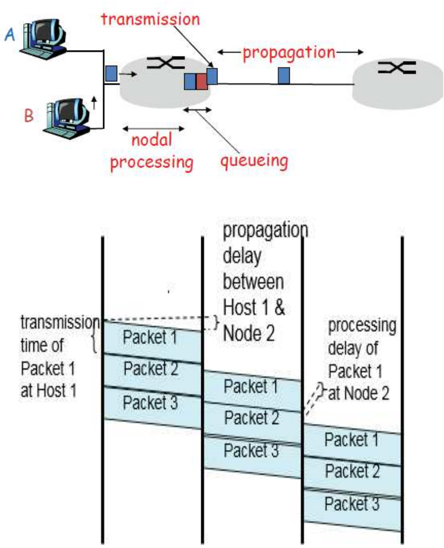
- כל אחד מהמשתמשים משדר בקצב קבוע (בלתי תלוי) של R=0.25Mbps כאשר הוא פעיל, אך כל משתמש פעיל רק 20% מהזמן.
- א. בשימוש ב-Circuit switching, מהו מס' המשתמשים המקסימאלי בהם הרשת יכולה לתמוך בו זמנית?
- ב. מהי התפוקה הממוצעת (קצב שידור בפועל) של הערוץ עבור התשובה שחישבתם ב-א'?

Types of Delay

- **Processing Delay** – the time to examine the packet and determine where to direct it.
- **Queuing Delay-** the time that packet can wait if the traffic is heavy and other packets are also waiting to be transmitted.
- **Transmission Delay-** the amount of time it requires to push all the packet's bits into the link.

depends on the Bandwidth (R) and the length of the packet (L)

$D_{trans} = L/R$



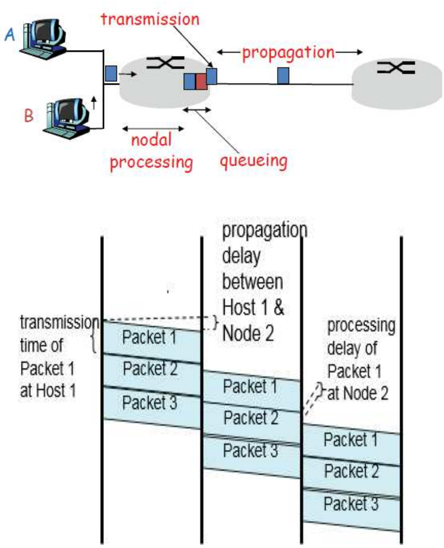
Types of Delay

- **Propagation Delay-** the time since all the bits pushed into the link till all the packet propagate to the end of the link.

depend on the physical medium of the link (fiber optics, twisted pair..) and on its range.

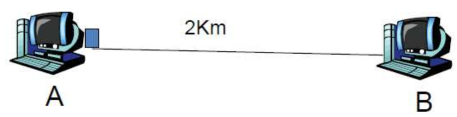
$D_{prop} = d/s$ $s \approx 2*10^8 m/sec$

- $D = D_{proc}+D_{queue}+D_{trans}+D_{prop}$



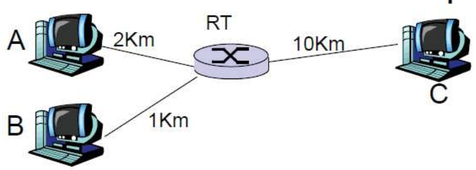
Q2

- הודעה שאורכה 500 בתים משודרת בקצב של 4Mbps על קו שאורכו 2 ק"מ.
- כמה זמן יעבור עד שההודעה תגיע ליעדה?



Q3

- מחשבים A ו-B מתחילים לשדר חבילה בגודל 2KB כל אחד
- R_A=3Mbps, R_B=2.5Mbps, R_{RT}=10Mbps
- זמן העיבוד של RT הוא 0.2ms
- כמה זמן ייקח לכל אחת מהחבילות להגיע ל- C במלואן?



- תרגיל 1: רשת ברוחב פס של 2Mbps המכילה N משתמשים. כל אחד מהמשתמשים משדר בקצב קבוע (בת"ל) של R=256Kbps כאשר הוא פעיל. כל משתמש פעיל רק 20% מהזמן.

א. מהו המס' המקסימלי M של משתמשים בהם ניתן לתמוך בו זמנית?

$$\frac{C}{R} = \frac{2Mbps}{256Kbps/user} = \frac{2048Kbps}{256Kbps/user} = 8\ users$$

ב. מהי התפוקה הממוצעת?

$$20\% * (256Kbps * 8) = 20\% * 2Mbps = 0.4Mbps$$

ג. מה ההסתברות שבדיוק 8 מתוך 40 פעילים כעת?

$$P(X = 8) = \binom{40}{8} * 0.2^8 * 0.8^{32} = 0.156$$

ד. מה ההסתברות שהדרישה עולה על קיבולת הערוץ?

$$P(X > 8) = \sum_{i=9}^{40} \binom{40}{i} * 0.2^i * 0.8^{(40-i)}$$

ה. מהי התפוקה הממוצעת של הערוץ?

$$\sum_{i=0}^{8} \binom{40}{i} * 0.2^i * 0.8^{(40-i)} * R * i + \sum_{i=9}^{40} \binom{40}{i} * 0.2^i * 0.8^{(40-i)} * C = 1.75Mbps$$

- תרגיל 2: הודעה שאורכה 500 בתים משודרת בקצב של 4Mbps על קו שאורכו 2 ק"מ. כמה זמן יעבור עד שההודעה תגיע ליעדה?

א. Tproc = 0, Tqueue = 0

ב. Ttrans = $\frac{L}{R} = \frac{500[B]}{4[Mb/sec]} = \frac{(500*8)[b]}{(4*2^{20})[b/sec]} = \frac{500*8}{4*2^{20}}[sec] = 0.95[ms]$

ג. Tprop = $\frac{d}{s} = \frac{2[Km]}{2*10^8[m/sec]} = \frac{2000[m]}{200000000[m/sec]} = 0.00001[sec] = 0.01[ms]$

ד. $T = T_{trans} + T_{prop} = 0.95[ms] + 0.01[ms] = 0.96[ms]$

- תרגיל 3: מחשבים A ו-B מתחילים לשדר חבילה בגודל 2KB כ"א.

נתון: R_A=3Mbps, R_B=2.5Mbps, R_R=10Mbps

נתון: D_{AR}=2Km, D_{BR}=1Km, D_{RC}=10Km

נתון: D_{Proc}=0.2ms

א. $A \rightarrow R = 0 + 0 + \frac{2KB}{3Mbps} + \frac{2Km}{2*10^8 mps} = \frac{16Kb}{(3*1024)Kb} sec + \frac{1}{10^5} sec = 5.208ms + 0.01ms$

ב. $B \rightarrow R = 0 + 0 + \frac{2KB}{2.5Mbps} + \frac{1Km}{2*10^8 mps} = 6.25ms + 0.005ms = 6.255ms$

ג. $(1) R \rightarrow C = (5.218) + 0.2 + 0 + \frac{16Kb}{10240Kb} sec + \frac{10Km}{2*10^8 mps} =$

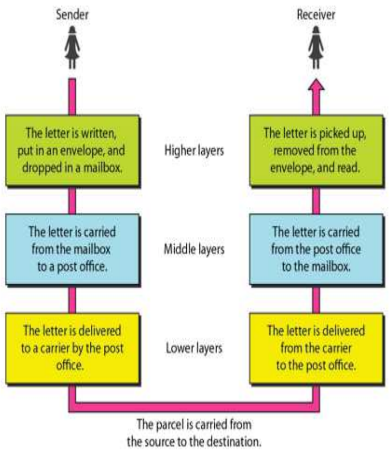
$$= 5.218 + 0.2 + 1.5625 + 0.05 = \mathbf{7.0305ms}$$

ד. $(2) R \rightarrow C = (6.255) + 0.2 + [5.218 + 1.5625 + 0.2 - (6.255 + 0.2)] + 1.5625 + 0.05$

$$= 6.255 + 0.2 + 0.5255 + 1.5625 + 0.2 = \mathbf{8.593ms}$$

Why layering?

- We use the concept of **layers** in our daily life.
- let us consider two friends who communicate through postal mail. The process of sending a letter to a friend would be complex if there were no services available from the post office.
- Using explicit structure allows identification & relationship of complex system's pieces.
- change of implementation of layer's service transparent to rest of system.



Introduction to Computer Networks

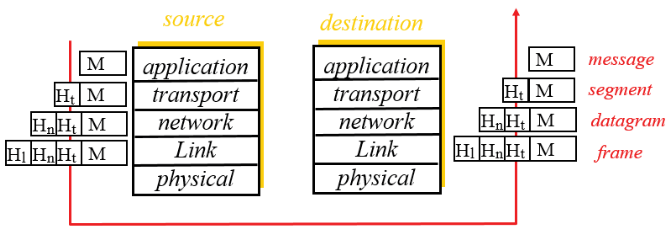
Practice #2

Hasidi Netanel

Protocol Layering

- To reduce their design complexity, most networks are organized as a **stack** of **layers** or **levels**, each one built upon the one below it.
- The purpose of each layer is to offer certain services to the higher layers, shielding those layers from the details of how the offered services are actually implemented.
- **Encapsulation**

  the process of each layer at the sending computer adding its own header information, in the form of meta-data to the actual **payload** (the data from the layer above). Ex: the network layer inserts a header with source and destination IPs.

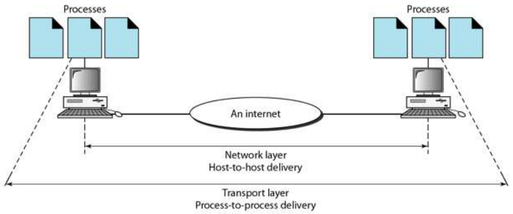


Protocols

- A **protocol** is an agreement between the communicating parties on how communication is to proceed.
- Protocols are set of rules.
- A protocol defines the format and the order of messages exchanged between two or more communicating entities as well as the actions taken on the transmission and/or receipt of a message or other event.
- Questions a protocol **may** answer:
  1. What do you want to do? (Application).
  2. Where are you going? (Addressing).
  3. How do you get there? (Media types).
  4. Did you get there? (Acknowledgments, Error checking).

The Transport Layer

- Responsibilities:
  - provides virtual end-to-end links between peer processes- delivering messages from one process to another.
- Issues:
  - error detection.
  - reliable communication.
  - packets reordering.
  - end-to-end flow control.



The Application Layer

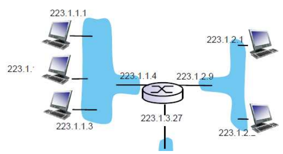
- Responsibility:
  - Enables users to access the network.
  - Providing user interfaces and services.
  - Providing understandable language between two applications on multiple end systems.
- Issues:
  - appropriate selection of “type of service”.
  - Data encryption, conversion and compression.
  - Authentication.

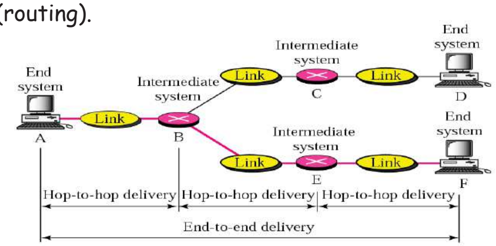
The Data Link Layer

- Responsibility:
  - Connect machines on the same networks.
  - *Framing* (dividing data into chunks) and moving it from one hop (node) to another.
  - Controls how data is placed into the media and is received from the media- who can transmit, when and for how long.
- Issues:
  - *Reliable Delivery*
  - Flow Control
  - Error Detection
  - Error Correction
  - Half/Full Duplex

The Network Layer

- Responsibilities:
  - Pass packets from one local Network to another- from the source host to the destination host.
  - fragmentation & reassembly
  - translation between different network types.
- Issues:
  - path selection between end-systems (routing).
  - Error reporting (status messages).

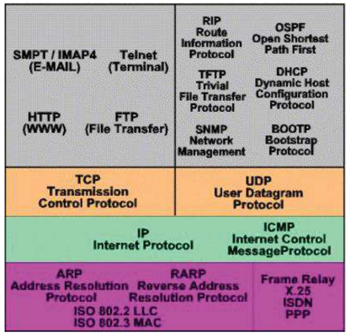




Transfer Units Types

Application Layer	Messages
Transport Layer	Segments
Network Layer	Datagrams
Data Link Layer	Frames

Protocols Examples



The Physical Layer

- Responsibility:
  - Encode binary digits into signals and transmit it over a communication channel.
- Issues:
  - mechanical and electrical interfaces
  - time per bit
  - distances

- An end-system processes up to which layer?
- A router processes up to which layer?
- A link-layer switch processes up to which layer?

Wireshark Example

	Time	Source	Destination	Protocol	Length	Info
4	2.98156300	10.0.0.1	107.22.19.8	TCP	66	27876→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM=1
9	3.15662500	107.22.19.8	10.0.0.1	TCP	66	80→27876 [SYN, ACK] Seq=0 Ack=1 Win=14600 Len=0 MSS=1360 SACK_PERM=1 WS=128
10	3.15676900	10.0.0.1	107.22.19.8	TCP	54	27876→80 [ACK] Seq=1 Ack=1 Win=17680 Len=0
11	3.15712000	10.0.0.1	107.22.19.8	HTTP	347	GET / HTTP/1.1
13	3.32948400	107.22.19.8	10.0.0.1	TCP	54	80→27876 [ACK] Seq=1 Ack=294 Win=15744 Len=0
14	3.33124200	107.22.19.8	10.0.0.1	TCP	1414	[TCP segment of a reassembled PDU]
15	3.33146000	107.22.19.8	10.0.0.1	HTTP	994	HTTP/1.1 200 OK (text/html)
16	3.33151900	10.0.0.1	107.22.19.8	TCP	54	27876→80 [ACK] Seq=294 Ack=2301 Win=17680 Len=0
17	3.33174400	107.22.19.8	10.0.0.1	TCP	54	80→27876 [FIN, ACK] Seq=2301 Ack=294 Win=15744 Len=0
18	3.33174600	10.0.0.1	107.22.19.8	TCP	54	27876→80 [FIN, ACK] Seq=294 Ack=2301 Win=17680 Len=0
19	3.33192500	10.0.0.1	107.22.19.8	TCP	54	27876→80 [ACK] Seq=295 Ack=2302 Win=17680 Len=0

Keep in mind

Data-Link: communication between machines on the same network (two adjacent nodes).

Network: communication between machines on possibly different networks.

Transport: communication between processes (running on machines on possibly different networks).

Today:

- ❑ Network Applications Architectures (Client Server vs. P2P).
- ❑ Http protocol.
- ❑ Web Cache (Proxy Server).

Application
Transport
Network
Link
Physical

Hasidi Netanel

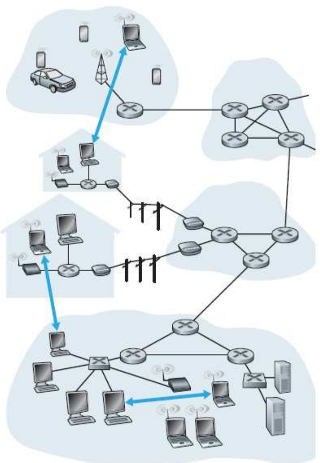
Introduction to Computer Networks

Practice #3

Hasidi Netanel

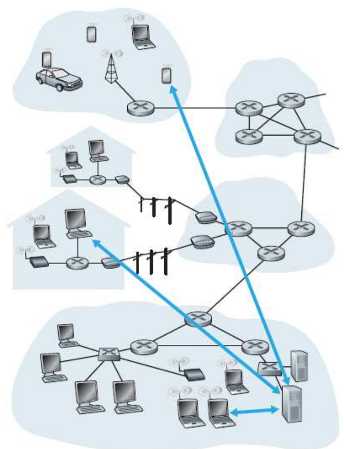
P2P (Peer to Peer)

- There is minimal (or no) reliance on always-on infrastructure servers.
- There is direct communication between pairs of intermittently connected hosts, called peers.
- The peers communicate without passing through a dedicated server.
- P2P architecture is self-scalability- one does not only requesting service but also adds service capacity to the system by distributing files to other peers.



Client-Server

- There is an always-on host called Server, which services requests from many other hosts, called Clients.
- Client hosts can be either sometimes-on or always-on.
- Servers are powerful computers and clients rely on Server for resources such as files, devices and even processing power.



Q1

- ברשת ישנם n משתמשים ושרת אחד. קצב העלאת הנתונים של השרת הוא 20Mbps. קצב ההורדה של חצי מהמשתמשים הוא 5Mbps ושל החצי השני הוא 2.5Mbps. קצב ההעלאה של כל אחד מהמשתמשים הוא 512Kbps.
- מהו הזמן המינימאלי הדרוש להפצת קובץ בגודל 0.5GB עבור ארכיטקטורת C-S ועבור P2P כאשר n=20, n=2000.

- **Distribute Files - *Client-Sever* :**
  - Time Require to the Server to send one copy of the file $\frac{F}{u_s}$
  - Time Require to one Client to download the file $\frac{F}{d_i}$
  - Total time Require to distribute the files

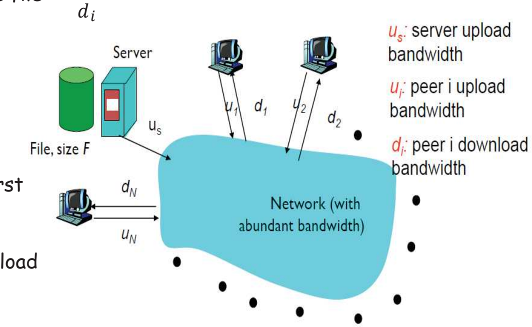
$$\max\left(\frac{N \cdot F}{u_s}, \frac{F}{\min(d_i)}\right)$$

- **Distribute Files - *P2P* :**
  - Time Require to the Server to send the first copy of the file $\frac{F}{u_s}$
  - Time Require to the slowest Client to download the file $\frac{F}{\min(d_i)}$
  - Minimum time distributing the files

$$\frac{N \cdot F}{u_s + \sum u_i}$$

  - Total time Require to distribute the files

$$\max\left(\frac{F}{u_s}, \frac{F}{\min(d_i)}, \frac{N \cdot F}{u_s + \sum u_i}\right)$$

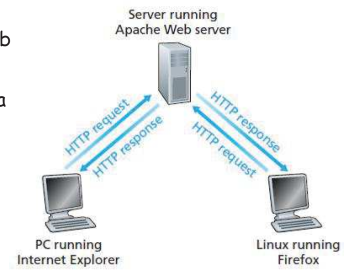


Web Cache (Proxy Server)

- Is a network entity that satisfies *HTTP requests* on the behalf of an origin Web server.
- The Web cache has its own disk storage and keeps copies of recently requested objects in this storage.
- A user's browser can be configured so that all of the user's *HTTP requests* are first directed to the Web cache.
- Once a browser is configured, each browser request for an object is first directed to the Web cache.
- What is good for:
  - It stores common objects near to the client and reduce the network traffic which allow more quick services. Also it reduce the Web traffic in the internet as a whole.

HTTP Protocol

- HTTP defines how Web clients request Web pages from Web servers and how servers transfer Web pages to clients.
- HTTP is implemented in two programs: a client program and a server program. The client program and server program, executing on different end systems, talk to each other by exchanging HTTP messages.
- HTTP uses TCP as its underlying transport protocol.



HTTP Request Message

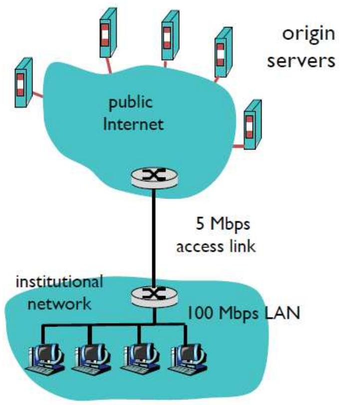
```
GET /somedir/page.html HTTP/1.1
Host: www.someschool.edu
Connection: close
User-agent: Mozilla/5.0
Accept-language: fr
```

HTTP Response Message

```
HTTP/1.1 200 OK
Connection: close
Date: Tue, 09 Aug 2011 15:44:04 GMT
Server: Apache/2.2.3 (CentOS)
Last-Modified: Tue, 09 Aug 2011 15:11:03 GMT
Content-Length: 6821
Content-Type: text/html
```

File Distribution Delays

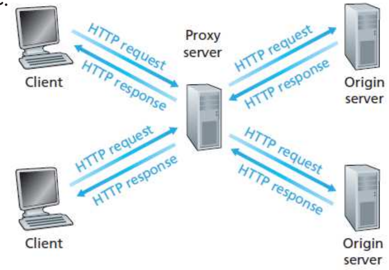
- **Internet Delay** – time to propagate on the link.
- **Access Delay** – the time it takes to transfer files from the internet to the local network.
- **LAN Delay** - the time it takes to transfer files from the local network to the local host.
- Note : all those delays including all the delays we've studied in practice #1.



Web Cache (Proxy Server)

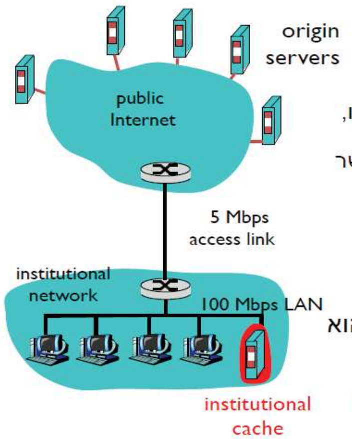
- The Process:

1. The browser establishes a *TCP connection* to the Web cache and sends an *HTTP request* for the object to the Web cache.
2. The Web cache checks to see if it has a copy of the object stored locally. If it does, the Web cache returns the object within an *HTTP response message* to the client browser.
3. If the web cache does not have the object it opens a *TCP connection* to the origin server and sends an *HTTP request* for the object.
4. The origin server sends the object to the web cache which stores a copy on its local storage.
5. The web cache then send the object within an *HTTP response message* to the client browser.



- Conditional GET : when the *HTTP GET request message* includes an *If-Modified-Since* header line . In this way the cache can ask the server the object only if it's has been modified since the specified date.

Q2



- נתונים:
  - גודל קובץ ממוצע: **100**Kb
  - קצב בקשות של קבצים מהשרתים: **15** בשנייה
  - השהיית אינטרנט (התפשטות על הקו, הלוך-חזור): **2** שניות
  - השהיית גישה ו-LAN: ms $\mathbf{X^3/2}$ כאשר X הוא אחוז הניצולת
  - גישה החוצה = 5Mbps
  - גישה LAN= 100Mbps

א. מהו זמן העברת הקבצים?

ב. נניח שרוחב הפס (גישה החוצה) הוא עכשיו 20Mbps, מהו זמן העברת הקבצים?

ג. נניח שקיים שרת פרוקסי בעל hit rate של 70% (רוחב פס 5Mbps), מהו זמן העברת הקבצים?

- תרגיל 1: ברשת ישנם N משתמשים ושרת אחד. קצב העלאת הנתונים של השרת הוא 20Mbps. קצב ההורדה של חצי מהמשתמשים הוא 5Mbps ושל החצי השני הוא 2.5Mbps.
קצב ההעלאה של כל אחד מהמשתמשים הוא 512Kbps.
מהו הזמן המינימאלי הדרוש להפצת קובץ בגודל 0.5GB עבור ארכיטקטורת C-S ועבור P2P כאשר N=20, ו- N=2000 .

א. עבור CS, כאשר N=20:
   - $N * \frac{F}{Us} = 20 * \frac{4096[Mb]}{20[Mb/sec]} = 4096[sec]$
   - $\frac{F}{\min(d_i)} = \frac{4096[Mb]}{2.5[Mb/sec]} = 1638.4[sec]$
   - $\boldsymbol{CS\ time = \ 4096[sec] = 68.26[min]}$

ב. עבור P2P, כאשר N=20:
   - $\frac{F}{U_s} = \frac{4096[Mb]}{20[Mbps]} = 204.8[sec]$
   - $\frac{F}{\min(d_i)} = \frac{4096[Mb]}{2.5[Mbps]} = 1638.4[sec]$
   - $N * \frac{F}{U_s + \sum U_i} = 20 * \frac{4096[Mb]}{20+20*0.5[Mbps]} = 2730.67[sec]$
   - $\boldsymbol{P2P\ time = 2730.67[sec] = 45.51[min]}$

ג. עבור CS, כאשר N=2000:
   - $N * \frac{F}{Us} = 2000 * \frac{4096[Mb]}{20[Mb/sec]} = 409{,}600[sec]$
   - $\frac{F}{\min(d_i)} = \frac{4096[Mb]}{2.5[Mb/sec]} = 1638.4[sec]$
   - $\boldsymbol{CS\ time = \ 409,600[sec] = 4.7[days]}$

ד. עבור P2P, כאשר N=2000:
   - $\frac{F}{U_s} = \frac{4096[Mb]}{20[Mbps]} = 204.8[sec]$
   - $\frac{F}{\min(d_i)} = \frac{4096[Mb]}{2.5[Mbps]} = 1638.4[sec]$
   - $N * \frac{F}{U_s + \sum U_i} = 2000 * \frac{4096[Mb]}{20+2000*0.5[Mbps]} = 8031.37[sec]$
   - $\boldsymbol{P2P\ time = 8031.37[sec] = 2.23[h]}$

- תרגיל 2: גודל קובץ ממוצע 100Kb. קצב בקשות קבצים: 15/sec. השהיית אינטרנט 2 שניות. השהיית גישה ו-LAN: $X^3/2$ כאשר X הוא אחוז הניצולת

א. עבור המצב הנתון (גישה החוצה = 5Mbps)

- $Internet\ Delay = 2[sec]$
- $Access\ Delay = \frac{X^3}{2} = \frac{29.29^3}{2} = 12572.8[ms] = 12.5728[sec]$

  $X = \frac{15*100}{5*1024} = 29.29\%$
- $LAN\ Delay = \frac{Y^3}{2} = \frac{1.46^3}{2} = 1.57[ms] = 0.00157[sec]$

  $Y = \frac{15*100}{100*1024} = 1.46\%$
- $Total\ time = 2 + 12.6728 + 0.00157 = 14.67437[sec]$

ב. עבור המצב שבו הגישה החוצה = *20Mbps*:

- $Internet\ Delay = 2[sec]$
- $Access\ Delay = \frac{X^3}{2} = \frac{7.324^3}{2} = 196.4[ms] = 0.1964[sec]$

  $\boldsymbol{X = \frac{15*100}{20*1024} = 7.324\%}$
- $LAN\ Delay = \frac{Y^3}{2} = \frac{1.46^3}{2} = 1.57[ms] = 0.00157[sec]$ (כמו קודם)
- $Total\ time = 2 + 0.1964 + 0.00157 = 2.19797[sec]$

ג. עבור שרת פרוקסי עם פגיעה של 70%, רוחב פס כמו קודם – *5Mbps*

כל הבקשות יגיעו לפרוקסי, 70% מהן יתקבלו מהשרת, 30% לא יתקבלו והשרת יבקש אותן מהאינטרנט ואז ישלח למחשב המבקש.

- $Internet\ Delay = 2[sec] * 0.3 = 0.6[sec]$
- $Access\ Delay = \frac{X^3}{2} = \frac{8.789^3}{2} = 339.467[ms] = 0.3394[sec]$

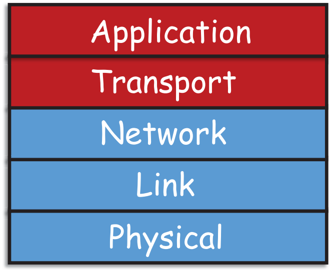
  $X = \frac{15*100*0.3}{5*1024} = 8.789\%$
- $LAN\ Delay = \frac{Y^3}{2} = \frac{1.904^3}{2} = 3.453[ms] = 0.00345[sec]$
- $\boldsymbol{Y = \frac{15*100*0.7+15*100*0.3*2}{100*1024} = 1.904\%}$

  כאשר יש פגיעה – הקובץ עובר מהפרוקסי למחשב (70% מהזמן)

  כאשר אין פגיעה – הקובץ עובר מהנתב לפרוקסי, ועוד פעם מהפרוקסי למחשב
- $Total\ time = 0.6 + 0.3394 + 0.00345 = 0.94285[sec]$

Today:

- Persistent & Non-Persistent Http.
- The Connection between Application & Transport Layers.
- Wireshark.



Introduction to Computer Networks

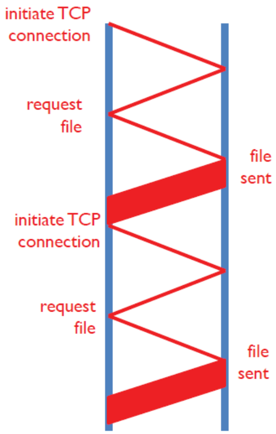
Practice #4

Hasidi Netanel

Persistent & Non-Persistent Http

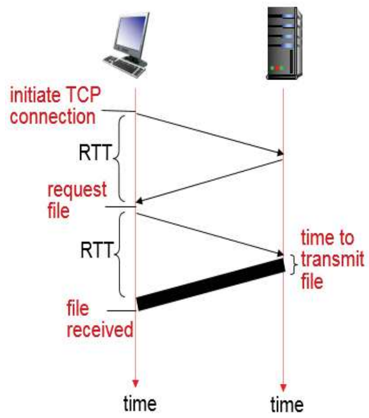
- ***Non Persistent Http***
  - A connection which only one object sent per tcp connection.
  - Downloading multiple objects required multiple connections
  - Initiate tcp connection, transfer object and then close the connection.
  - Connection field = close.
  - Execution Time =

$$(2 \cdot RTT + T_{Trans}) \cdot \#Files$$



Persistent & Non-Persistent Http

- ***RTT (Round Trip Time)***
  - is the time it takes for a small packet to travel from client to server and then back to the client.
  - The RTT includes packet-propagation delays, packet queuing delays in intermediate routers and switches, and packet-processing delays



The Connection between Application & Transport Layers

- The ***Transport Layer*** provide logical communication between app processes running on different hosts.

<u>Services an application may need</u>

- Data integrity – some apps need 100 % reliable transfer, some can tolerate some loss.
- Timing – low delay.
- Throughput.
- Security – encryption.

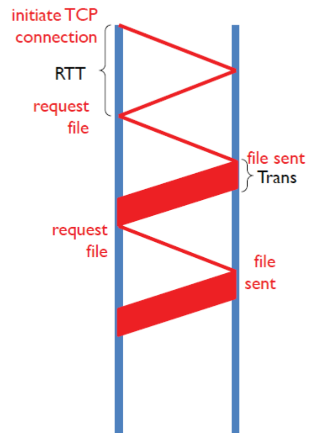
<u>Services an application may need</u>

- Reliable Transfer.
- Flow Control.
- Congestion Control.
- Connection Setup.
- ***Not Provide:***
  - Delay guarantees.
  - Bandwidth guarantees.

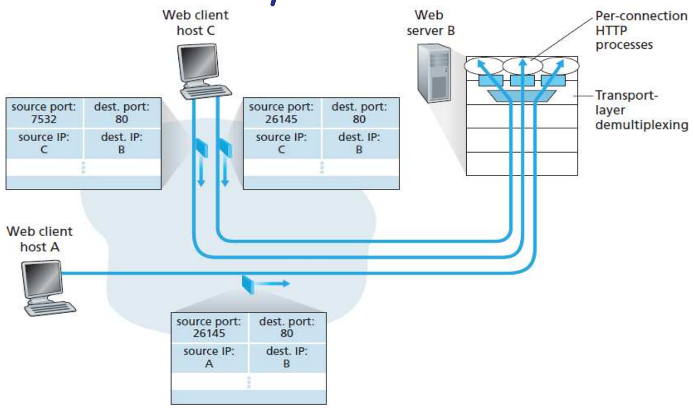
Persistent & Non-Persistent Http

- ***Persistent Http***
  - multiple objects can be sent over single TCP connection between client and server.
  - Initiate tcp connection only at the beginning, transfer objects and then close the connection.
  - Default field in http 1.1 .
  - Connection field = Keep-Alive.
  - Execution Time =

$$\text{RTT} + (RTT + T_{Trans}) \cdot \#Files$$



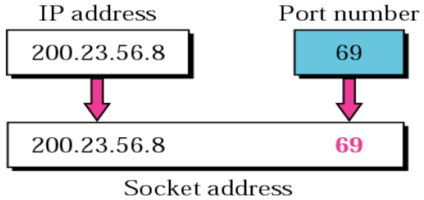
The Connection between Application & Transport Layers



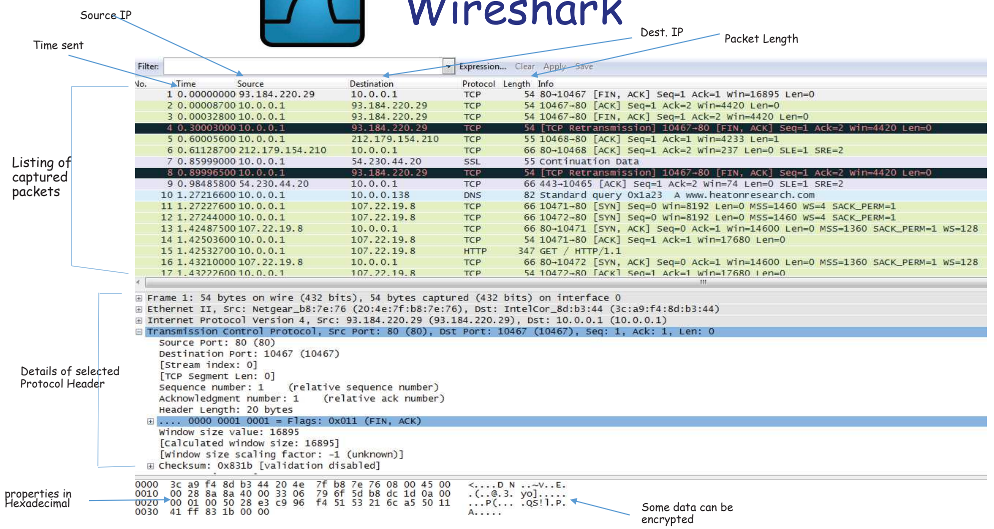
Two clients, using the same destination port to communicate with the same web server application.

The Connection between Application & Transport Layers

- ***Port Number*** – A port number is a way to identify a specific process to which an Internet or other network message is to be forwarded.
- ***Socket*** – A network socket is one endpoint in a communication flow between two programs running over a network.
- Socket contains 4 tuples – source IP, Dest. IP, source Port, Dest. Port.

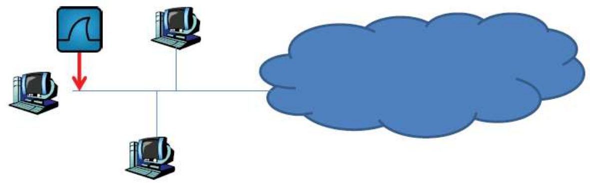


Wireshark



Wireshark

- Wireshark is a free and open source network protocol analyzer that enables users to interactively browse the data traffic on a computer network.
- it is used to analyze the structure of different network protocols and has the ability to demonstrate encapsulation.



Wireshark

9	1.04633400	10.0.0.1	10.0.0.138	DNS	79 Standard query 0xc22d A www.httprecipes.com
10	1.33992000	10.0.0.138	10.0.0.1	DNS	120 Standard query response 0xc22d CNAME httprecipes.com A 107.22.19.8
11	1.34128200	10.0.0.1	107.22.19.8	TCP	66 3695→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM=1
12	1.34191100	10.0.0.1	10.0.0.138	DNS	75 Standard query 0x290e A httprecipes.com
13	1.34320200	10.0.0.138	10.0.0.1	DNS	91 Standard query response 0x290e A 107.22.19.8
14	1.34431000	10.0.0.1	10.0.0.138	DNS	75 Standard query 0xb6be AAAA httprecipes.com
15	1.35129000	107.22.19.8	10.0.0.1	TCP	58 80→3695 [SYN, ACK] Seq=0 Ack=1 Win=27456 Len=0 MSS=1360
16	1.35145100	10.0.0.1	107.22.19.8	TCP	54 3695→80 [ACK] Seq=1 Ack=1 Win=17680 Len=0
17	1.35185600	10.0.0.1	107.22.19.8	HTTP	347 GET / HTTP/1.1

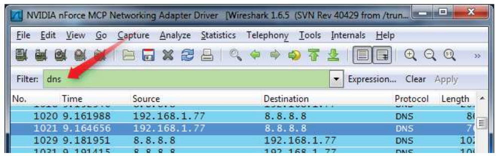
כמה שכבות לוקחות חלק בפאקטות 9,10,12,13,14?

וכמה לוקחות חלק ב 11,15,16?

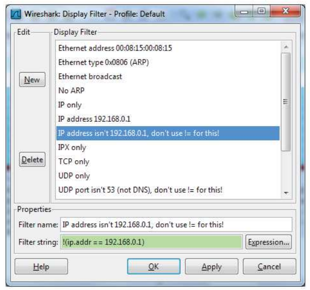
מדוע בקשת הדף היא האחרונה בסדרת חבילות זו?

Wireshark

You can filter packets by their top layer protocol name



You can use the filter expression editor



Wireshark

```
40 3.57023600 107.22.19.8   10.0.0.1      HTTP   210 HTTP/1.1 200 OK  (text/html)
41 3.57058600 10.0.0.1      107.22.19.8   TCP     54 3695→80 [FIN, ACK] Seq=294 Ack=2301 Win=17524 Len=0
42 3.57743300 107.22.19.8   10.0.0.1      TCP     54 80→3695 [FIN, ACK] Seq=2301 Ack=294 Win=123 Len=0
43 3.57751100 10.0.0.1      107.22.19.8   TCP     54 3695→80 [ACK] Seq=295 Ack=2302 Win=17524 Len=0
44 3.60978800 10.0.0.1      107.22.19.8   TCP     66 3696→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM=1
45 3.61056800 10.0.0.1      10.0.0.138    DNS     82 Standard query 0x8604  A www.heatonresearch.com

Hypertext Transfer Protocol
Line-based text data: text/html
  <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">\n
  <HTML>\n
  <HEAD>\n
  \t<TITLE>HTTP Recipes</TITLE>\n
  \t<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">\n
  \t<meta http-equiv="Cache-Control" content="no-cache">\n
  \t<meta http-equiv="Content-Style-Type" content="text/css">\n
  <link rel="alternate" type="application/rss+xml" title="RSS" href="http://www.httprecipes.com/1/12/rss2.xml">\n
  </HEAD>\n
  \n
  <BODY>\n
  \n
  <table border="0"><tr><td>\n
```

האם בקשת דף האינטרנט הצליחה ומה סוג התוכן שאתם מבחינים בו?

Wireshark

```
    Questions: 1
    Answer RRs: 2
    Authority RRs: 0
    Additional RRs: 1
⊟ Queries
  ⊟ www.httprecipes.com: type A, class IN
      Name: www.httprecipes.com
      [Name Length: 19]
      [Label Count: 3]
      Type: A (Host Address) (1)
      Class: IN (0x0001)
⊟ Answers
  ⊞ www.httprecipes.com: type CNAME, class IN, cname httprecipes.com
  ⊞ httprecipes.com: type A, class IN, addr 107.22.19.8
⊟ Additional records
  ⊞ <Root>: type OPT
```

מה התוכן של הודעה זו?
איזה שכבה משרתת סוג תוכן זה?

א. השינוי הוא בכתובת ה IP בתוך תחום הכתובת של חברת, GoDaddy ולכן

כאשר משנים את כתובת IP של השרת יש צורך לעדכן רק בשרתים המהימנים

המחזיקים את הכתובת www.yossi.co.il.

ב. על יוסי לבצע מס' שינויים:

- לעדכן בשרת ה DNS Authorative של GoDaddy שהוא כבר לא שם .

- לעדכן בשרת ה TLDשאחראי על co.il, שכעת יש authorative DNS חדש ,שבו נמצא השרת המהימן האחראי לדומיין שלו.

- לעדכן את השרתים המהימנים של החברה החדשה בכתובת של האתר שלו

Today:

- ARQ Protocols.

Application
Transport
Network
Link
Physical

Introduction to Computer Networks

Practice #5

Hasidi Netanel

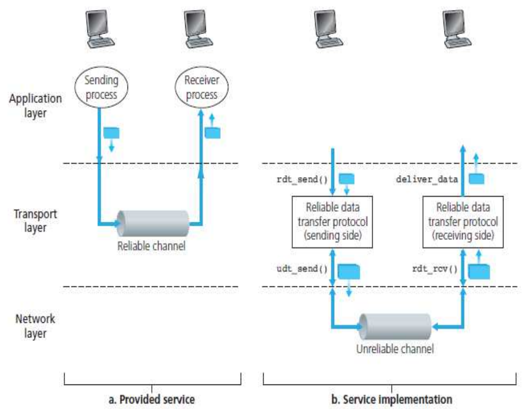
ARQ (Automatic Repeat reQuest) Protocols

- We know now the channel isn't reliable so we need to find a way that allows the receiver to let the sender know what has been received correctly, and what has been received in error and thus requires repeating.
- If the information didn't received properly, it must be send again.
- Reliable data transfer protocols based on this technique (retransmission) knows as *ARQ protocols*.
- We can estimate the protocol based on Utilization,

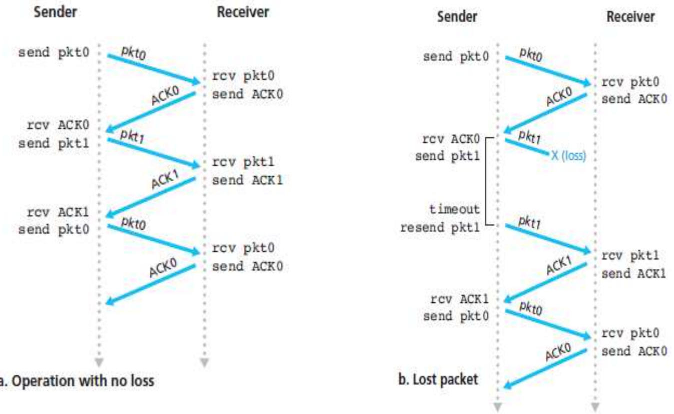
$$U = \frac{transmission\ time\ takes\ without\ loss}{actual\ time\ till\ ack}$$

What is Reliable Data Transfer?

- ***Reliable Channel***
  - With a reliable channel, no transferred data bits are corrupted or lost, and all are delivered in the order in which they were sent.
  - It is the responsibility of a **reliable data transfer protocol** to implement this service abstraction.



Stop & Wait ARQ

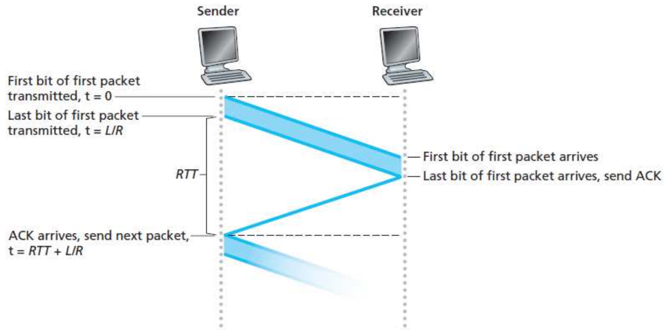


a. Operation with no loss

b. Lost packet

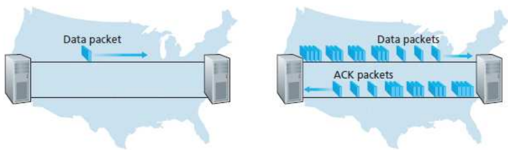
Stop & Wait ARQ

- Send one packet and wait for ACK.
- If the ACK has arrived, send another packet.
- If the ACK has lost, sent the packet again.



ARQ Protocols

- ***Pipelining technique***
  - Rather than operate in a stop-and-wait manner, the sender is allowed to send multiple packets without waiting for acknowledgments.
  - The sender and receiver sides of the protocols may have to buffer more than one packet.



a. A stop-and-wait protocol in operation

b. A pipelined protocol in operation

Stop & Wait ARQ

Without loss

$$U_{sender} = \frac{L/R}{RTT + L/R} = \frac{0.008}{3 + 0.008} = 0.0026$$

$$U_{sender} = \frac{T_{tr}}{T_{RTT} + T_{tr}} = \frac{1}{\frac{T_{RTT} + T_{tr}}{T_{tr}}} = \frac{1}{\frac{T_{RTT}}{T_{tr}} + 1}$$

β מייצג את כמות החבילות שאפשר לשלוח ב-RTT אחד. $\beta = \frac{T_{RTT}}{T_{tr}}$

$$U_{sender} = \frac{1}{\beta + 1}$$

With loss

Pr(loss packet) = p

חבילה מועברת עד שהיא נשלחת בהצלחה

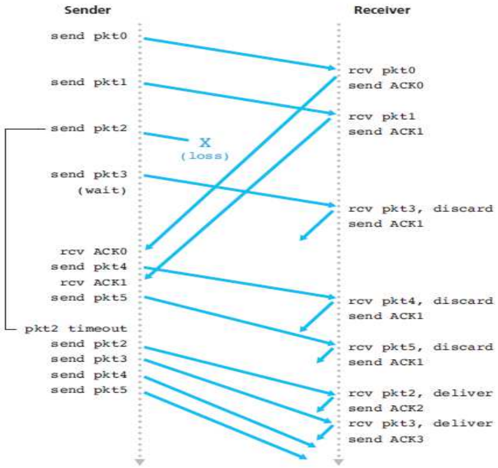
כלומר, התפלגות גיאומטרית להצלחה (1-p)

$$X \sim G(1-p) \qquad E(X) = \frac{1}{1-p}$$

E(X) – כמה פעמים צריך לשלוח בפועל עד שמגיעה חבילה מוצלחת

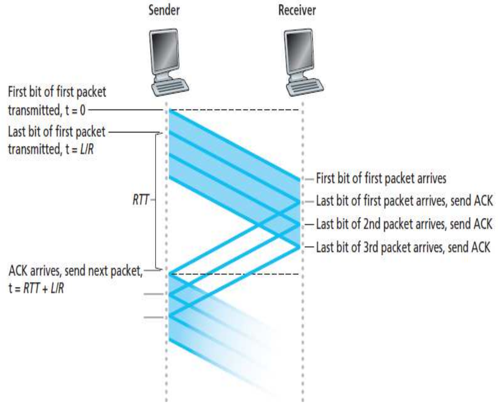
$$U_{sender} = \frac{1}{(\beta + 1) * E(X)} = \frac{1-p}{\beta + 1}$$

GBN (Go Back N) ARQ



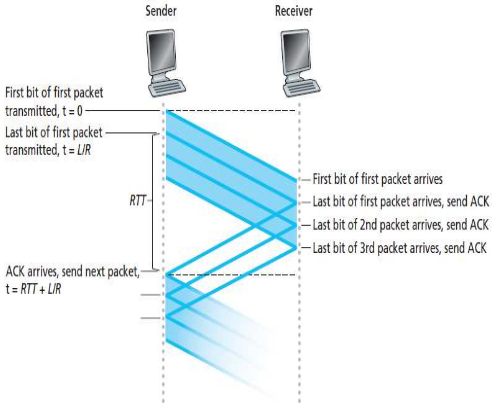
GBN (Go Back N) ARQ

- The sender is allowed to transmit multiple packets (when available) without waiting for an acknowledgment, but is constrained to have no more than some maximum allowable number, *N*, of unacknowledged packets in the pipeline.
- An acknowledgment for a packet with sequence number *n* will be taken to be a **cumulative acknowledgment**, indicating that all packets with a sequence number up to and including *n* have been correctly received at the receiver.
- If a timeout occurs, the sender resends *all* packets that have been previously sent but that have not yet been acknowledged.



SR(Selective Repeat) ARQ

- Retransmit only what is really need !
- The SR receiver will acknowledge a correctly received packet (individually) whether or not it is in order.
- Out-of-order packets are buffered until any missing packets are received, at which point a batch of packets can be delivered in order to the upper layer.



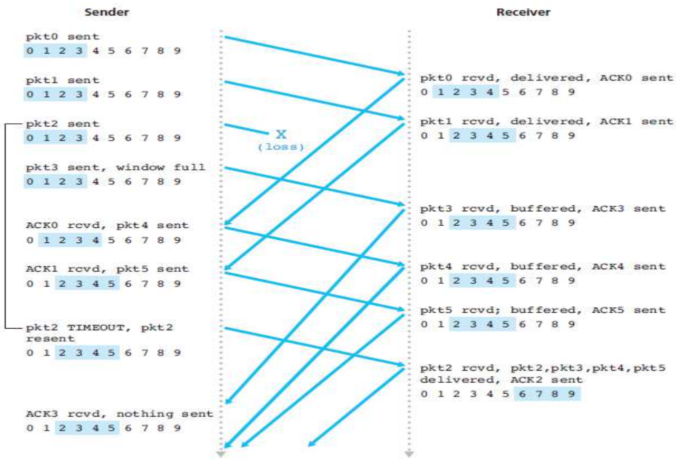
GBN (Go Back N) ARQ

Without loss

$$U_{sender} = \frac{N * L/R}{RTT + L/R} = \frac{3 * 0.008}{3 + 0.008} = 0.0208$$

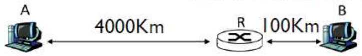
$$U_{sender} = \frac{N}{(\beta + 1)}$$

SR(Selective Repeat) ARQ



Q.

- נתונה רשת בעלת 2 מחשבים וביניהם נתב
- ל-A יש מידע רב שעליו לשלוח ל-B
- הפרוטוקול שעובד בין A ל-R הוא GBN עם חלון בגודל 5
- הפרוטוקול שעובד בין R ל-B הוא S&W
- קצב השליחה של A הוא 400Kbps
- גודל הסגמנט הוא 256B
- הניחו כי אין שגיאות ואיבודים על הקו, אך כן ממתינים ל-ACK, כמו כן הניחו כי גודל חבילות ACK הינו זניח.



- א. חשבו את קצב השידור המינימאלי הדרוש בין R כך שהחוצצים לא יוצפו מעבר לקיבולת שלהם.
- ב. בהינתן קצב השידור שחושב בסעיף א', הניחו שמחשב A שולח הודעה בגודל 100KB לצומת B. כמה זמן ייקח עד שההודעה כולה תגיע ליעד? (הניחו כי גודל הכותרת של ההודעה – ה-Header – הוא זניח)

DNS – Domain Name System

Domain Name System - מערכת לתרגום כתובות URL ל IP . המערכת בנויה משרתים המחזיקים מאגרי מידע בצורה מבוזרת לפי היררכיות בכדי להתמודד בעיקר עם הבעיות הבאות:

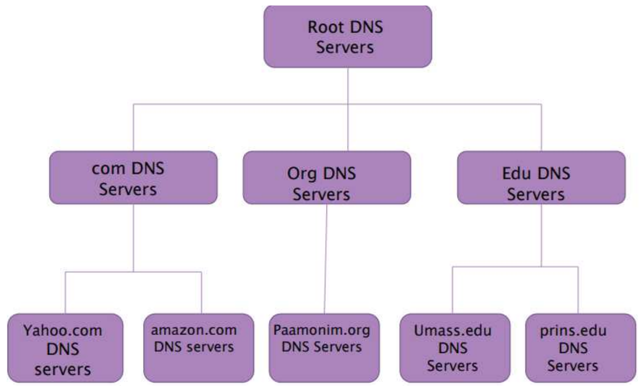
- עומס תעבורה
- מרחק פיזי מתחנות היעד
- תחזוקה שוטפת

DNS Hierarchy

ה DNS בנוי מ 3 היררכיות:

- Root DNS Servers - מחזיקים את הכתובות של שרתי TLD וממירים בין הסיומת של ה URL לבין כתובת ה IP של שרת ה TLDהרלוונטי.
- Top Level Domain – שרתים אלו אחראיים על הסיומות כגון: com , gov, il וכו'.
- Authoritative DNS Servers - כל ארגון, או חברה שרוצה לאפשר למשתמשים להתחבר אליה, צריכה להחזיק שרת הממיר בין הכתובות URLשלה לבין כתובות ה IPשל השרתים שלה (או לחילופין לשכור מקום בחברת אחסון אתרים). לרוב, חברות וארגונים גדולים (כגון אוני', גוגל, וכו') יחזיקו שרתים שלהם, לעומת אנשים פרטיים או חברות קטנות שישתמשו בשירותים של חברות אחסון

DNS Hierarchy



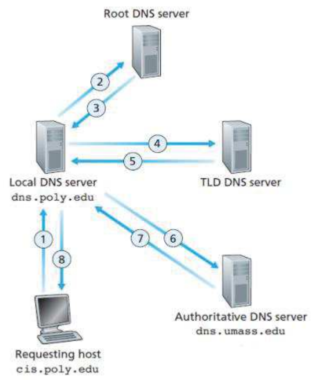
Local DNS Server

- אינו חלק מההיררכיה של ה DNS, עם זאת הוא ממלא תפקיד חשוב בארכיטקטורה של ה DNS.
- כל ISP לדוגמא אוני', חברה גדולה, נטויז'ן, וכו' מתחזקת שרת מקומי.
- כאשר משתמש שולח שאילתת DNS, ה ISP מפנה אותו ראשית לשרת המקומי ומשם השרת המקומי מפנה את השאילתא לתוך היררכיית הDNS .
- שרת מקומי לרוב יהיה באותו LAN של המשתמש או במרחק של ראוטר או שניים ממנו.

שיטות לביצוע שאילתות

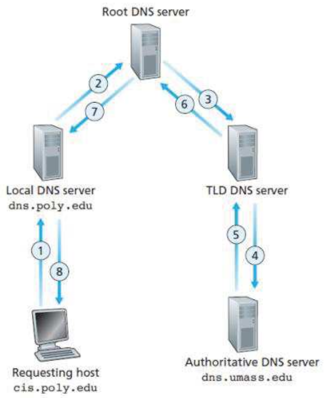
ישנן 2 שיטות לביצוע שאילתת DNS:

- **איטרטיבית**- בשיטה זו, המחשב המבקש, שולח בקשה לאחד משרתי ה Root, לאחר קבלת התשובה (כתובת של שרת TLD) הוא פונה (בעצמו) לשרת ה TLDולאחר קבלת תשובה, הוא פונה (שוב בעצמו) לשרת המהימן (Authoritative)



שיטות לביצוע שאילתות

- **רקורסיבית-** בשיטה זו, המחשב המבקש שולח בקשה אחת לשרת ה, Root.
  שרת הRoot , לא מחזיר תשובה עדיין אלא שולח בעצמו את הבקשה לשרת ה TLDשאינו מחזיר תשובה עדיין אלא שולח את השאילתא לשרת המהימן שמחזיר תשובה לשרת ה TLDשמחזיר תשובה לשרת ה Root שמחזיר תשובה למחשב המבקש.



תרגיל

א. יוסי פתח אתר ורכש שם דומיין לאתר: www.Yossi.co.il יוסי משתמש בשירותי אחסון אתרים של GoDaddy , והאתר שלו מאוחסן בשרת שכתובתו 66.96.147.149. לאחר שנה שבה פעל האתר ללא בעיות החליטו בחברת GoDaddy להעביר את האתר של יוסי לשרת אחר שכתובתו 66.96.147.150.אלו שינויים יש לבצע במערכת ה- DNS כדי שהגולשים באתר יוכלו להמשיך לגלוש לאתר של יוסי ללא בעיות?

ב. יוסי החליט לשנות את חברת האחסון לחברה אחרת בעלת שרתים אחרים. אילו שינויים ידרשו לבצע במערכת ה DNSבמקרה זה?

שאלה 1:

נתונה רשת בעלת 2 מחשבים וביניהם נתב, ל-A יש מידע רב שעליו לשלוח ל-B, הפרוטוקול שעובד בין A ל-R הוא GBN עם חלון בגודל 5. הפרוטוקול שעובד בין R ל-B הוא S&W.

קצב השליחה של A הוא 400Kbps, גודל הסגמנט הוא 256B. הניחו כי אין שגיאות ואיבודים על הקו, אך כן ממתינים ל-ACK, כמו כן הניחו כי גודל חבילות ACK הינו זניח.

א. חשבו את קצב השידור המינימאלי הדרוש בין R כך שהחוצצים לא יוצפו מעבר לקיבולת שלהם.

נוודא שגודל החלון (מבחינת כמות הודעות) גדול מ-5 (N<β+1):

$$\beta = \frac{T_{RTT}}{T_{tr}} = \frac{2*\frac{d}{s}}{\frac{L}{R_A}} = \frac{0.04}{0.005} = 8$$

$$N = 5 < 8 + 1$$

Window Time (A-R): $T_{trans} + T_{RTT} = \frac{L}{R_A} + 2 * \frac{d}{s} = \frac{256*\frac{8}{1024}}{400} + 2 * \frac{4000*1000}{2*10^8} = 0.005 + 0.04 = 0.045\ [sec]$

גודל החלון של 5 הודעות ב- R-B צריך להיות באותו הגודל של A-R כדי שכל ההודעות יספיקו להישלח עד שההודעה הבאה תגיע.

$0.045 = 5(T_{trans} + T_{RTT}) \rightarrow 0.009 = \frac{2}{R} + 2 * \frac{100*1000}{2*10^8} \rightarrow 0.009 = \frac{2}{R} + 0.001 \rightarrow R = \frac{2}{0.008} = \mathbf{250Kbps}$

ב. בהינתן קצב השידור שחושב בסעיף א', הניחו שמחשב A שולח הודעה בגודל 100KB לצומת B. כמה זמן ייקח עד שההודעה כולה תגיע ליעד? (הניחו כי גודל הכותרת של ההודעה – ה-Header – הוא זניח)

תחילה, נחשב כמה חבילות צריכות להישלח:

$$\frac{100KB}{256B} = \frac{100}{0.25} = 400$$

400 ההודעות הללו נשלחות בקבוצות של 5 הודעות בחלון *GBN*. כלומר, יש צורך ב-80 חלונות.

חשוב לציין שיש פעולות שקורא במגביל, ולכן רק נספור את הזמנים שקובע התוצאה.

נסתכל על 79 החלונות הראשונים בנפרד, ועל החלון האחרון (ה-80) באופן פרטני:

Time to send the first 79 windows (A->R): $79 * WindowSize = 79 * 0.045 = 3.555$
After 3.555 seconds from the start, R has received 79 windows worth of segments and A is ready to send the last windows worth. However, now we don't care about the time it takes to send the last ACK but rather at what time the last segment arrives at B completely.

Time to transmit the Last window (A->R->B):

$$T_{trans_A} + \frac{1}{2}T_{RTT_{AR}} + 4\left(T_{trans_R} + T_{RTT_{RB}}\right) + T_{trans_R} + \frac{1}{2}T_{RTT_{RB}} = 0.005 + 0.02 + 4\left(\frac{2}{250} + 0.001\right) + \frac{2}{250} + 0.0005 = 0.0695\ sec$$

הסבר: $T_{trans_A} + \frac{1}{2}T_{RTT_{AR}}$ – הזמן שהסגמנט **הראשון** יצא מ-A ויגיע ל-R (בלי ACK)

(note: once the first segment arrives at R, we know the last 4 will arrive immediately following by the method of GBN. Therefore we continue timing the transference of the 5 segments from R->B)

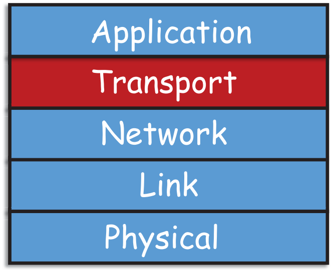
$4\left(T_{trans_R} + T_{RTT_{RB}}\right)$ - הזמן ש-4 הסגמנטים (מתוך 5) יצאו מ-R, יגיעו ל-B ויחזרו עם ACK בהצלחה

$T_{trans_R} + \frac{1}{2}T_{RTT_{RB}}$ - הזמן שהסגמנט החמישי (האחרון) יצא מ-R ויגיע ל-B (**בלי** ACK).
אחרי נקודה זו ההודעה הגיעה במלואה ל-B. עדיין נותר הזמן לשלוח ACK-ים מ-B ל-R ומ-R ל-A, אך זה לא רלוונטי מבחינת השאלה.

Total Time = $\mathbf{3.555 + 0.0695 = 3.6245}\ \boldsymbol{sec}$

Today:

- UDP
- TCP



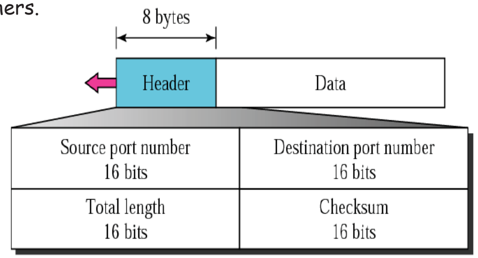
Introduction to Computer Networks

Practice #6

Hasidi Netanel

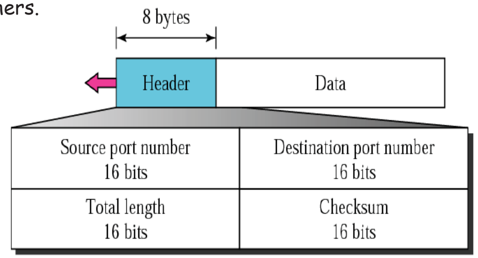
Connectionless Transport: UDP

- Does just about as little as a transport protocol can do
  - Provide a multiplexing/demultiplexing service in order to pass data between network layer and the current application level process (plus some light error detection service).
- Is Connectionless
  - no handshaking between UDP sender, receiver.
  - each UDP segment handled independently of others.
- Is "best effort"
  - The application is almost directly talking with the network layer therefore packet can be lost and deliver out of order to the application.



Connectionless Transport: UDP

- Does just about as little as a transport protocol can do
  - Provide a multiplexing/demultiplexing service in order to pass data between network layer and the current application level process (plus some light error detection service).
- Is Connectionless
  - no handshaking between UDP sender, receiver.
  - each UDP segment handled independently of others.
- Is "best effort"
  - The application is almost directly talking with the network layer therefore packet can be lost and deliver out of order to the application.



UDP Checksum

- Provide error detection.

Sender

0110011001100000
0101010101010101
1000111100001100
1 0100101011000001
1
0100101011000010

Checksum: 1011010100111101

Receiver

0110011001100000
0101010101010111
1000111100001100
1 0100101011000011
1
0100101011000100

checksum → 1011010100111101
1 0000000000000001
1
0000000000000010 → Error Detection !

UDP Checksum

- Provide error detection.

Sender

0110011001100000
0101010101010101
1000111100001100
1 0100101011000001
1
0100101011000010

Checksum: 1011010100111101

Receiver

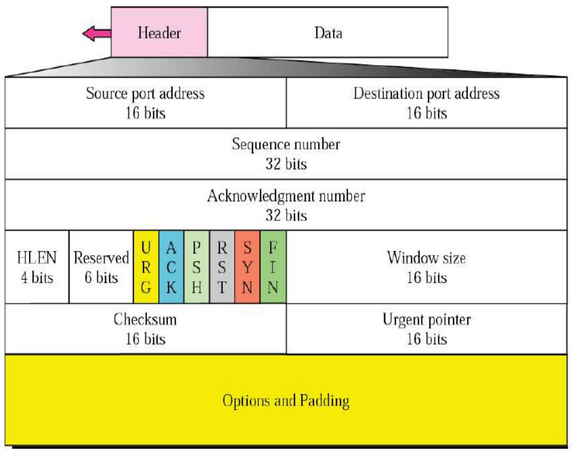
0110011001100000
0101010101010101
1000111100001100
1 0100101011000001
1
0100101011000010

checksum → 1011010100111101
1111111111111111 → No errors

Connection Oriented: TCP

TCP creates reliable service using:

- Pipelined technique.
- Cumulative Acks.
- Single retransmission timer.
- Checksum.
- Retransmissions triggered by:
  - Timeout events.
  - Duplicate Acks.



UDP Checksum

- Provide error detection.

Sender

0110011001100000
0101010101010101
1000111100001100
1 0100101011000001
1
0100101011000010

Checksum: 1011010100111101

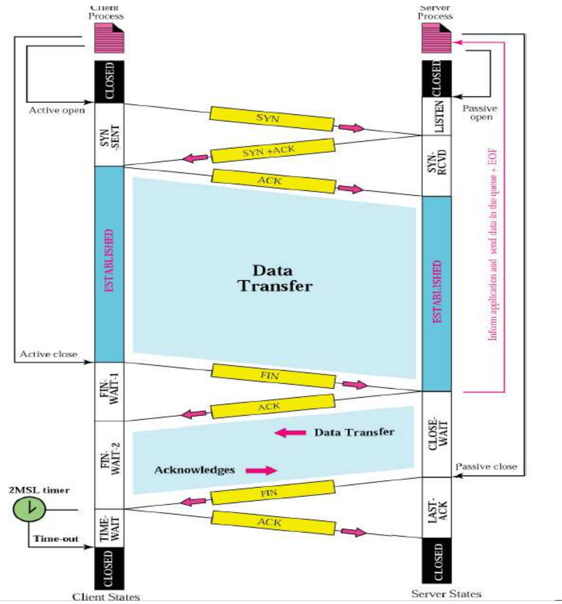
Receiver

0110011001100000
0101010100010101
1000111101001100
1 0100101011000001
1
0100101011000010

checksum → 1011010100111101
1111111111111111 → No error Detection ! But there is an Error!

Connection Oriented: TCP

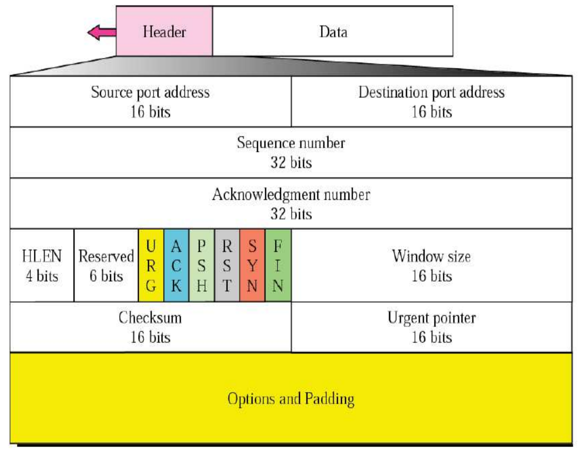
TCP connection establishment and Termination



Connection Oriented: TCP

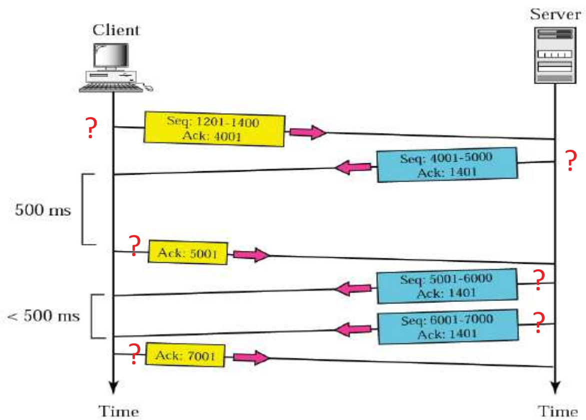
TCP Segment Format

- Typical Header size = 20 B.
- Window size – is used for *flow control*, indicate the number of bytes that a receiver is willing to accept.
- ACK bit – is used if the segment contains an acknowledgment for a segment that has been successfully received.
- SYN bit - Sent in the first packet when initiating a connection.
- RST bit - The flag causes the receiver to reset the connection.
- FIN bit - Used for closing a connection. Both sides need to send a FIN.



TCP ACK Rules

ציין ליד כל אחד מהפאקטות איזה מס' חוק הוא מייצג?

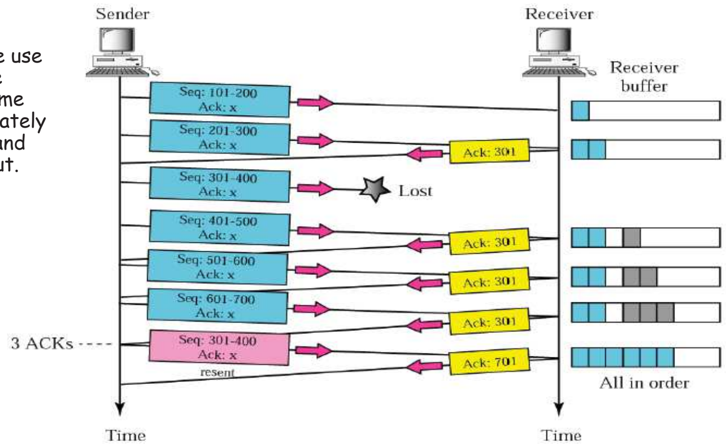


TCP ACK Rules

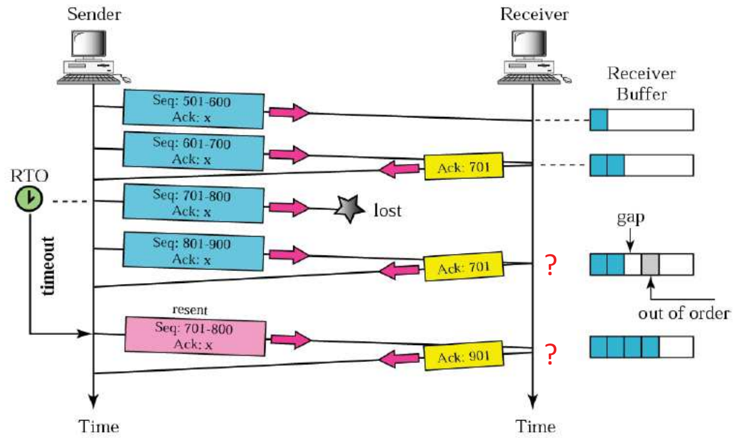
1) If (need to send data)
   Send also an ACK giving next seq. N expected (piggyback). → Reduce traffic.
2) If (no data to send & receive only one in-order ACK)
   Delay Ack until another ack arrives || until period of time has passed. → Prevents extra traffic.
3) If (new in-order segment has arrived & didn't send ack for the packet before)
   Send immediately Ack. (straightforward of 2).
4) If (out-of-order segment received) ← Creates a gap.
   Send immediately ack (its next seq. num expected) → leads to ***Fast Retransmission***.
5) If (missing segment arrives) ← Fills a gap. solve the problem of **lost segment**.
   Send Ack (probably a cumulative ack) → informs sender that missing segment finally received.
6) If (segment arrives many times) ← Solve the problem of **lost ack**.
   Send immediately ack.

TCP ACK Rules

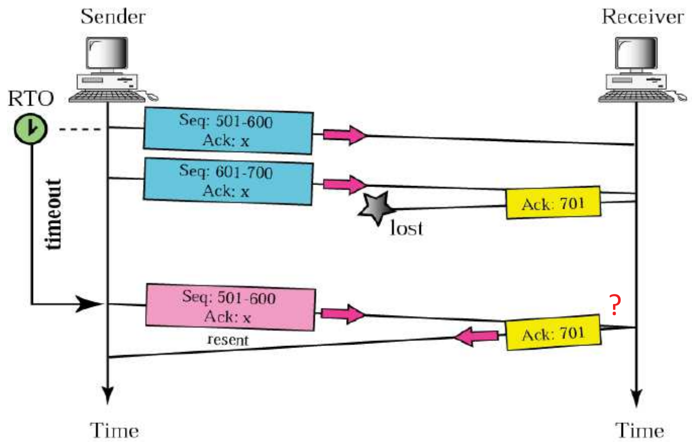
Fast Retransmission- in order to save timeout we use 3 dup. ack rule. When we received 3 ack on the same segment we send immediately the requested segment and don't wait for the timeout.



TCP ACK Rules



TCP ACK Rules



Today:

- ❑ RTO & RTT Estimation
- ❑ TCP Flow Control
- ❑ TCP Congestion Control

Application
Transport
Network
Link
Physical

Introduction to Computer Networks

Practice #7

Hasidi Netanel

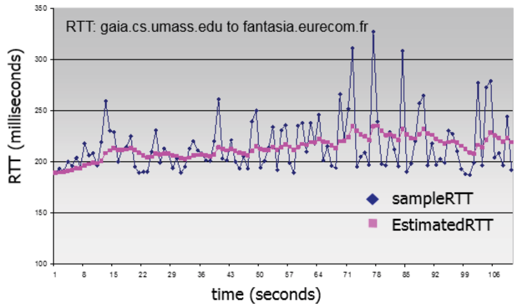
RTO & RTT Estimation

- Setting the timeout value (RTO)
  - Estimated RTT + "safely margin"
  - We measure the deviation of the *Sampled RTT* from the *Estimation RTT*.
  - Based on ***RTT*** -> but RTT **varies**.

```
DevRTT = (1-β)*DevRTT +
              β*|SampleRTT-EstimatedRTT|
              (typically, β = 0.25)
```

```
TimeoutInterval = EstimatedRTT + 4*DevRTT
```

estimated RTT "safety margin"

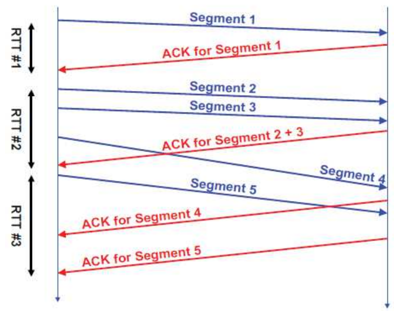


RTO & RTT Estimation

- How to estimate TCP timeout value - RTO (Retransmission time out)?
  - Value too small -> results in unnecessary retransmission.
  - Value too long -> long waiting time, long timing.
  - Based on ***RTT*** -> but RTT **varies**.
- We set RTO as a function of RTT.
- How to estimate RTT ?
  - **SampleRTT** - Each TCP connection measures the time difference between the transmission of a segment and the receipt of the corresponding ACK.

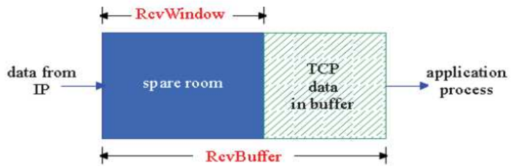
```
EstimatedRTT = (1- α)*EstimatedRTT + α*SampleRTT
```

- Typical value $\alpha = 0.125$
  - We want to consider also the last sample because it is the most updated. Nevertheless the average estimation is more steady.



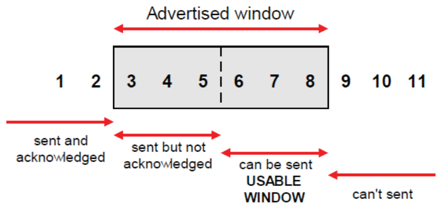
TCP Flow Control

- sender won't overflow receiver's buffer by transmitting too much, too fast.
- receive side of TCP connection has a receive buffer:



TCP "lie" about the real buffer size for a situation where it overflow.

- receiver advertises a window ***rwnd*** (RcvWin) with each acknowledgement.
- Window can be
  - closed (by sender) when data is sent and ack'd.
  - opened (by receiver) when data is read



Q1

לפניך פלט של תוכנה wireshark. חשב זמן RTT של כל סגמנט.

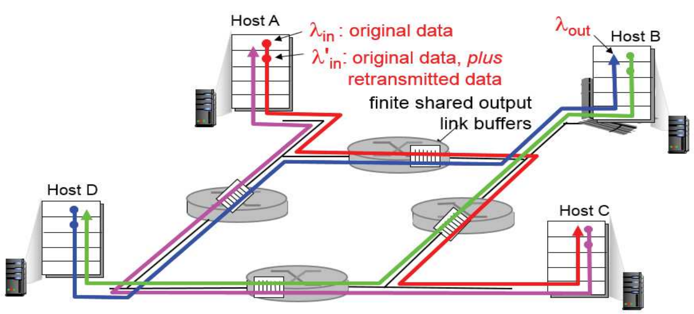
```
 6 1.427574000  10.0.0.2        128.119.245.12  TCP   680 63576 > http [PSH, ACK] Seq=1 Ack=1 Win=16384 Len=626
 7 1.428210000  10.0.0.2        128.119.245.12  TCP  1414 63576 > http [ACK] Seq=627 Ack=1 Win=16384 Len=1360
 8 1.428258000  10.0.0.2        128.119.245.12  TCP  1414 63576 > http [ACK] Seq=1987 Ack=1 Win=16384 Len=1360
 9 1.428304000  10.0.0.2        128.119.245.12  TCP  1414 63576 > http [ACK] Seq=3347 Ack=1 Win=16384 Len=1360
10 1.585183000  128.119.245.12  10.0.0.2        TCP    54 http > 63576 [ACK] Seq=1 Ack=627 Win=7168 Len=0
11 1.585337000  10.0.0.2        128.119.245.12  TCP  1414 63576 > http [ACK] Seq=4707 Ack=1 Win=16384 Len=1360
12 1.58957600   128.119.245.12  10.0.0.2        TCP    54 http > 63576 [ACK] Seq=1 Ack=1987 Win=9856 Len=0
13 1.589714000  10.0.0.2        128.119.245.12  TCP  1414 63576 > http [ACK] Seq=6067 Ack=1 Win=16384 Len=1360
14 1.589771000  10.0.0.2        128.119.245.12  TCP  1414 63576 > http [ACK] Seq=7427 Ack=1 Win=16384 Len=1360
15 1.591828000  128.119.245.12  10.0.0.2        TCP    54 http > 63576 [ACK] Seq=1 Ack=3347 Win=12544 Len=0
16 1.591928000  10.0.0.2        128.119.245.12  TCP  1414 63576 > http [ACK] Seq=8787 Ack=1 Win=16384 Len=1360
17 1.591973000  10.0.0.2        128.119.245.12  TCP  1414 63576 > http [ACK] Seq=10147 Ack=1 Win=16384 Len=1360
18 1.594937000  128.119.245.12  10.0.0.2        TCP    54 http > 63576 [ACK] Seq=1 Ack=4707 Win=15360 Len=0
19 1.59503000   10.0.0.2        128.119.245.12  TCP  1414 63576 > http [ACK] Seq=11507 Ack=1 Win=16384 Len=1360
20 1.595081000  10.0.0.2        128.119.245.12  TCP  1414 63576 > http [ACK] Seq=12867 Ack=1 Win=16384 Len=1360
21 1.744014000  128.119.245.12  10.0.0.2        TCP    54 http > 63576 [ACK] Seq=1 Ack=6067 Win=18048 Len=0
22 1.74414300   10.0.0.2        128.119.245.12  TCP  1414 63576 > http [ACK] Seq=14227 Ack=1 Win=16384 Len=1360
23 1.744191000  10.0.0.2        128.119.245.12  TCP  1414 63576 > http [ACK] Seq=15587 Ack=1 Win=16384 Len=1360
24 1.744228000  10.0.0.2        128.119.245.12  TCP   680 63576 > http [PSH, ACK] Seq=16947 Ack=1 Win=16384 Len=626
25 1.795969000  128.119.245.12  10.0.0.2        TCP    54 http > 63576 [ACK] Seq=1 Ack=7427 Win=20736 Len=0
```

Segment	Send Time	ACK Time	Sample RTT	Estimated RTT
1	1.4275	1.5851	0.1576	0.1576
2	1.4282	1.5895	0.1613	0.1580625
3	1.4285	1.5918	0.1633	0.158717188
4	1.4283	1.5949	0.1666	0.159702539
5	1.5853	1.744	0.1587	0.159577222
6	1.5897	1.7959	0.2062	0.165405069

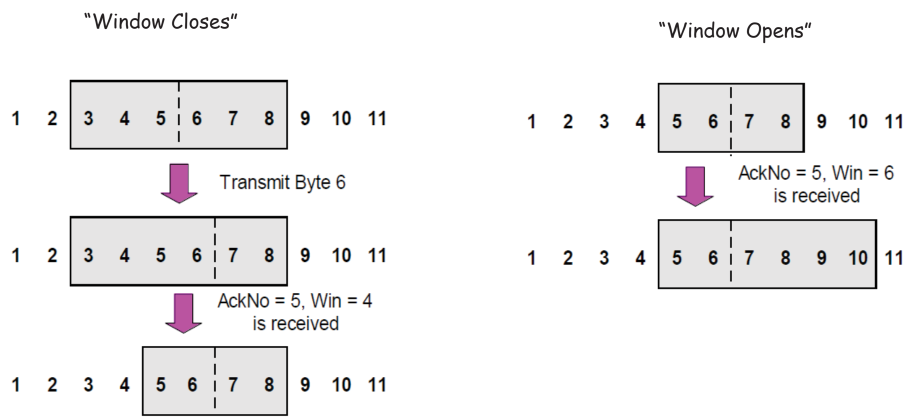
TCP Congestion Control

- Congestion occurs when load on the network greater than capacity of the network, e.g. "too many sources sending too much data too fast for *network* to handle".
- Different from Flow Control the *network* can't handle, not the end point.

Keep in mind! Students used to be confused



TCP Flow Control

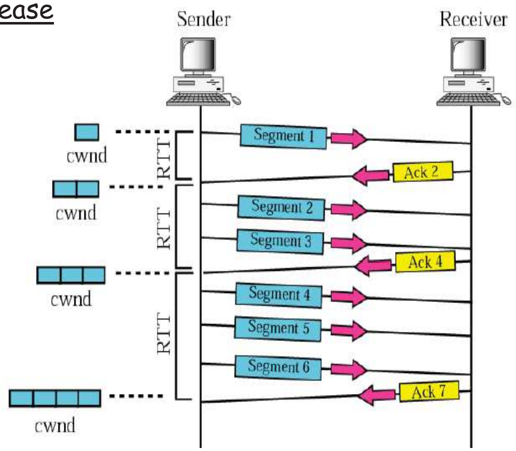


TCP Congestion Control

Congestion Avoidance – Additive increase

- Congestion Avoidance is used if ***cwnd* >= *ssthresh***.
- Increase the value of cwnd by just a single MSS every RTT.

$$cwnd = cwnd + MSS$$



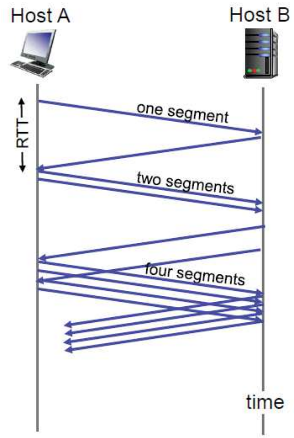
TCP Congestion Control

Slow Start

- Slow start is used if ***cwnd* < *ssthresh***.
- Slow start named because it starts with the congestion window ***cwnd*** = 1 MMS .
- increase rate exponentially. ***cwnd*** is doubled for every RTT received.

$$cwnd = cwnd \cdot 2$$

- initial rate is slow but ramps up exponentially fast



TCP Window Size

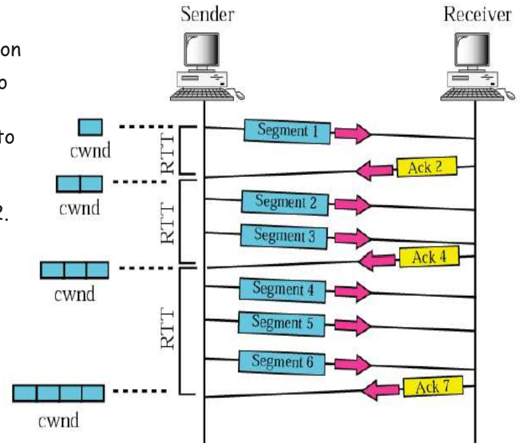
- TCP separates receiver congestion from network congestion, and uses window flow controls for each
  - rwnd: receiver window opened (by receiver) when data is read.
  - cwnd: congestion window
- TCP must not send data with a higher sequence number than the sum of the highest acknowledged sequence number and min(cwnd, rwnd).

Window Size = minimum (rwnd , cwnd)

TCP Congestion Control

Congestion Detection

- We react **differently** to congestion detection
- packet loss detected by a ***timeout*** → go into slow start (cwnd = 1 MMS).
- packet loss detected by ***Dup ACKs*** → go into ***fast recovery*** (cwnd = cwnd /2).
- In **both cases** threshold dropped to half (multiplicative decrease) sstresh = cwnd / 2.
- In ***fast recovery*** we increase the value of cwnd by 1 for every dup ack received.



Q2

מחשב מסוים משדר קובץ בגודל של 1MB על גבי סגמנטי TCP בעלי MSS = 1460 בתים.
כמו כן ידוע כי ערך הסף (Threshold) לחלון ה congestion control הוא 128 סגמנטים בחלון.

א. כמה סגמנטים אמורים להישלח?
ב. נתון כי בחלון 11 קרתה תקלה רגעית באחד הנתבים- תקלה שגרמה לכל הסגמנטים בחלון הזה להיזרק. בסיום החלון השולח הגיע ל Timeout מבלי שקיבל אישור על אף אחד מהסגמנטים הללו. למעט מקרה זה נתון כי כל שאר הסגמנטים שנשלחו הגיעו בהצלחה כמו גם כל האישורים עליהם (ACKs).
מלא את הטבלה הבאה עד לשורה בה כל הסגמנטים הנדרשים עברו בהצלחה.

Window number	#Segments Per Window	Total Segment Sent
1	1	1
....	....	....

$|F| = 1\,MB$, $MSS = 1460\,B$

$$\#Seg = \frac{F}{MSS} = \left\lceil \frac{1 \cdot 1024^2\,B}{1460\,B} \right\rceil = 719 \text{ seg}$$

$SSThresh = 128$ seg

Win Num	Seg per win	Total Seg Sent	
1	1	1	slow start
2	2	3	
3	4	7	
4	8	15	
5	16	31	
6	32	63	
7	64	127 = 64+63	
8	128 ← thresh	255	congestion avoidance
9	129	384	
10	130	514	
Fault (11)	—	—	timeout — congestion Detection: $ssThresh = \frac{cwnd}{2} = \frac{130}{2} = 65$
again 11	1	515	slow start
12	2	517	
13	4	521	
14	8	529	
15	16	545	
16	32	577	
17	64	641	
18	65 ←	706	congestion Avoidance
19	13 [illegible]	719	

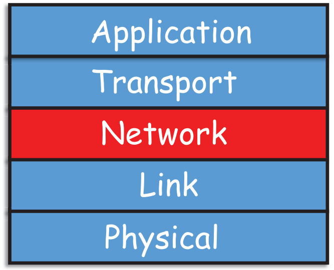
Introduction to Computer Networks

Practice #8

Hasidi Netanel

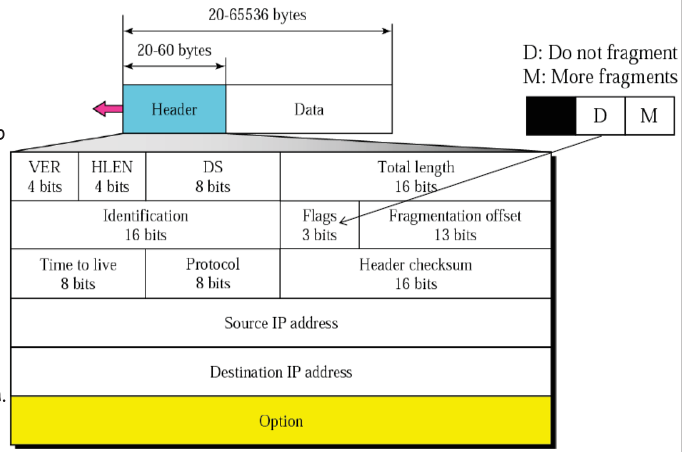
Today:

- ❑ IP Header.
- ❑ IP Fragmentation.
- ❑ ICMP.
- ❑ DHCP.
- ❑ NAT.



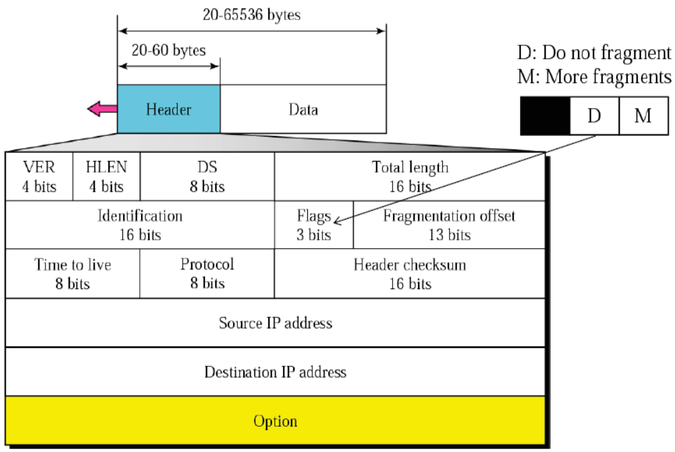
IP Header

- Identification- datagram number (analogues to sequence number in TCP).
- Time to live- number of hops until the destination. If this field gets to zero and didn't have reached destination- we throw the datagram.
- Fragmentation offset- if the datagram was fragmented, its indicated the data offset (multiply by 8).
- Flags- indicate if there are more fragmented datagrams in the current datagram identification.



IP Header

- Protocol- analogous to the role of the port number field in the transport layer segment. used only when an IP datagram reaches its final destination. The value of this field indicates the specific transport-layer protocol to which the data portion of this IP datagram should be passed.

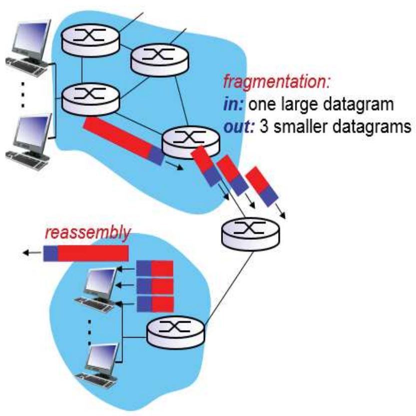


Q1

נתונה datagram בגודל 4096 בתים, אשר מגיעה לערוץ בו ה MTU הינו 1500 בתים.
הניח שמספר הזיהוי של החבילה היא 21.
עבור כל פרגמנט שיווצר רשמו את השדות הבאים ב header
{identification_Number, length, offset, flag}

IP Fragmentation

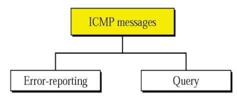
- Not all link-layer protocols can carry network-layer packets of the same size. Therefore the network layer must fragment datagrams before the Link Layer can carry those packets.
- ***MTU*** (Maximum Transfer Unit) -
The maximum amount of **data** that a link-layer frame can carry. For example, Ethernet frames can carry up to 1,500 bytes of data, whereas frames for some wide-area links can carry no more than 576 bytes.
- Fragments are reassembled only at the final destination.



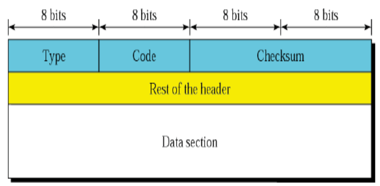
ICMP

Type	Code	Description	
3	0-15	Destination unreachable	Notification that an IP datagram could not be forwarded and was dropped. The code field contains an explanation.
4	0	Source Quench	Notification that an IP datagram has been discarded due to congestion in a router or the dest. host. The source must slow down the sending of datagrams until the congestion is relieved
5	0-3	Redirect	Informs about an alternative route for the datagram and should result in a routing table update. The code field explains the reason for the route change.
11	0, 1	Time exceeded	Sent when the TTL field has reached zero (Code 0) or when there is a timeout for the reassembly of segments (Code 1)
12	0, 1	Parameter problem	Sent when the IP header is invalid (Code 0) or when an IP header option is missing (Code 1)

ICMP



- IP does not provide any clue to where datagrams were get lost.
- ICMP is a helper protocol that supports IP with facility for: Error reporting & Simple queries.
- ICMP does not correct errors, it simply reports them.
- ICMP always reports error messages to the original source.
- Checksum – not consider the IP header.



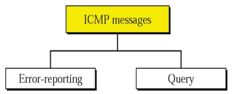
Q2

:Wireshark לפנייך פלט של תוכנה

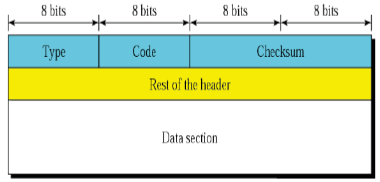
No.	Source	Destination	Protocol	Info
140	192.168.0.4	81.218.31.185	ICMP	Echo (ping) request id=0x0001, seq=4774/42514, ttl=6
141	62.219.189.186	192.168.0.4	ICMP	Time-to-live exceeded (Time to live exceeded in transit)
142	192.168.0.4	81.218.31.185	ICMP	Echo (ping) request id=0x0001, seq=4775/42770, ttl=6
143	62.219.189.186	192.168.0.4	ICMP	Time-to-live exceeded (Time to live exceeded in transit)
144	192.168.0.4	81.218.31.185	ICMP	Echo (ping) request id=0x0001, seq=4776/43026, ttl=6
145	62.219.189.186	192.168.0.4	ICMP	Time-to-live exceeded (Time to live exceeded in transit)
154	192.168.0.4	81.218.31.185	ICMP	Echo (ping) request id=0x0001, seq=4777/43282, ttl=7
155	212.179.16.98	192.168.0.4	ICMP	Time-to-live exceeded (Time to live exceeded in transit)
156	192.168.0.4	81.218.31.185	ICMP	Echo (ping) request id=0x0001, seq=4778/43538, ttl=7
157	212.179.16.98	192.168.0.4	ICMP	Time-to-live exceeded (Time to live exceeded in transit)
158	192.168.0.4	81.218.31.185	ICMP	Echo (ping) request id=0x0001, seq=4779/43794, ttl=7
159	212.179.16.98	192.168.0.4	ICMP	Time-to-live exceeded (Time to live exceeded in transit)
168	192.168.0.4	81.218.31.185	ICMP	Echo (ping) request id=0x0001, seq=4780/44050, ttl=8
169	81.218.31.185	192.168.0.4	ICMP	Echo (ping) reply id=0x0001, seq=4780/44050, ttl=55
170	192.168.0.4	81.218.31.185	ICMP	Echo (ping) request id=0x0001, seq=4781/44306, ttl=8
171	81.218.31.185	192.168.0.4	ICMP	Echo (ping) reply id=0x0001, seq=4781/44306, ttl=55
172	192.168.0.4	81.218.31.185	ICMP	Echo (ping) request id=0x0001, seq=4782/44562, ttl=8
174	81.218.31.185	192.168.0.4	ICMP	Echo (ping) reply id=0x0001, seq=4782/44562, ttl=55

א. מאיזה שכבה התהליך החל ? (אפליקציה, תעבורה, רשת , לא ניתן לדעת).
ב. למי שייכת הכתובת 212.179.16.98 ? (נתב, מחשב\שרת , לא ניתן לדעת).
ג. ב. למי שייכת הכתובת 81.218.31.185 ? (נתב, מחשב\שרת , לא ניתן לדעת).
ד. כמה נתבים יש בדרך עד שההודעה מגיעה ליעדה (אם בכלל)?

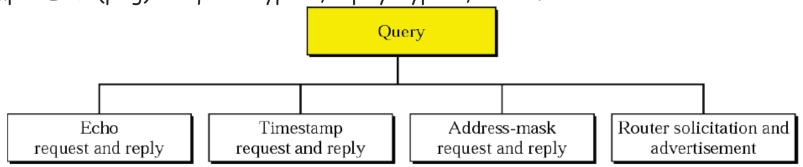
ICMP



- IP does not provide any clue to where datagrams were get lost.
- ICMP is a helper protocol that supports IP with facility for: Error reporting & Simple queries.
- ICMP does not correct errors, it simply reports them.
- ICMP always reports error messages to the original source.
- Checksum – not consider the IP header.

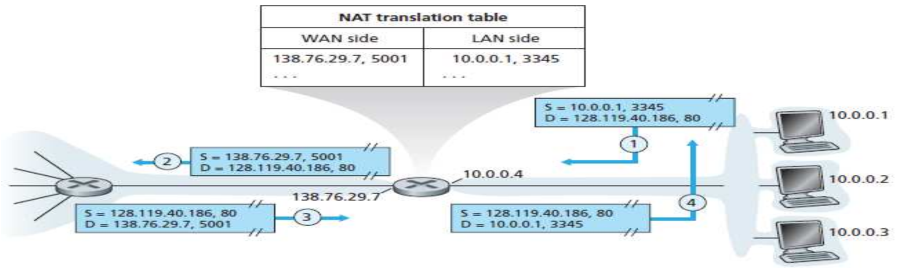


- Example: Timestamp – request: type=13, replay: type=14.
- Example: Echo (ping) – request: type=8, replay: type=0,code=0.



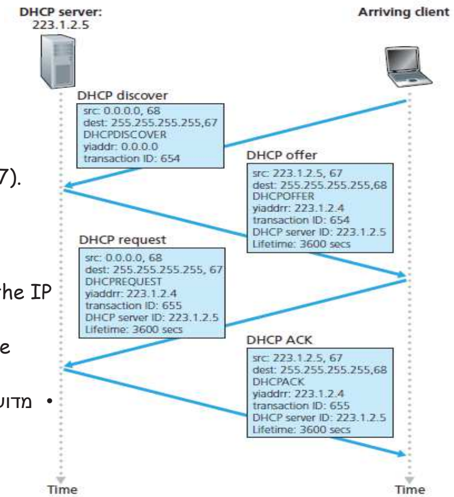
NAT (Network Address Translation)

- A Technique used to translate public network address to private network address.
- Allow us to use one set of IP addresses for internal traffic and a second set of addresses for external traffic.
- NAT serves these purposes:
  - Provide type of firewall – hiding internal IP address.
  - Enable a company to connect multiple machines to one IP address allocated by ISP instead buy a set of addresses.



DHCP

- A protocol that allow host to dynamically obtain its IP address from network server when it joins network.
- *Plug & Play* – refer to the ability of the protocol to connecting a host into a network automatically.
- DHCP Message Types:
  - DHCP Discover. (send within UDP packet to port 67).
  - DHCP Offer. (Broadcast message).
  - DHCP Request. (also Broadcast).
  - DHCP Ack.
- *IP address lease time* – the amount of time for which the IP address will be valid.
- *Transaction ID* – to identify each request (there can be more than one DHCP server within a network).
- מדוע הודעת DHCP Request נשלחת ב broadcast ולא ב unicast?



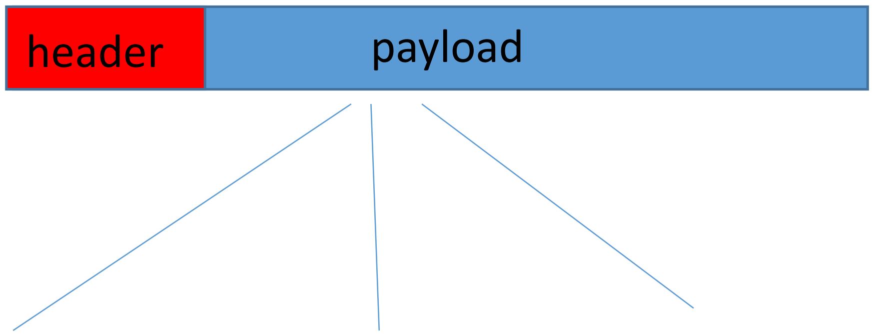
NAT (Network Address Translation)

- Advantages:
  - Need to buy just one IP address from ISP that can serve the whole local network.
  - Can change local addresses without notifying the outside world.
  - Can change ISP without change addresses of internal devices (the opposite from above).
  - Provide security- internal device addresses are not explicitly addressable.
- *Disadvantages:*
  - *Belongs to the network layer but do transport layer work .*
  - *Cant deliver encrypted packets (IPsec).*
  - *What happened if someone from outside wants to initiate the connection (traversal Problem).*

שאלה 1

|Datagram|= 4096 B
MTU = 1500 B
ID = 21
Data = 4096-20= 4076

$$\#fragmentedDatagrams = \frac{4076}{1500-20} = 2.754 \rightarrow 3$$



כמו שנאמר
בתרגול ב
payload נמצא ה
data וההדרים של
השכבות הקודמות

ID:	21	21	21
Size:	1480	1480	4076-2*1480= 116
Offset:	0	1480/8 = 185	185*2= 370
Flag:	1	1	0

שאלה 2

א. התהליך החל משכבת האפליקציה. אלו הקלטות על הפעלה של כלי ה traceroute שהוא סוג של אפליקציה לכל דבר. זה שהוא משתמש ונעזר בהודעות פינג באמצעות פרוטוקול icmp לא אומר שזה מתחיל משכבה 3.

ב. מכתובת זו נשלחת הודעה icmp time to leave שזה אומר שמס' הקפיצות נגמר- מס' הראוטרים שההודעה ברשת יכולה לעבור, כלומר זהו הראוטור האחרון בדרך ליעד. רק נתב יכול לשלוח הודעה כזו.

ג. לא ניתן לדעת את אופי התחנה כיוון שאפשר לשלוח פינג גם לנתב וגם למחשב\שרת.

ד. 7/8 תלוי באופי התחנה האחרונה (מחשב או נתב). מזהים שיש reply עבור פינג שנשלח עם TTL=8.

Introduction to Computer Networks

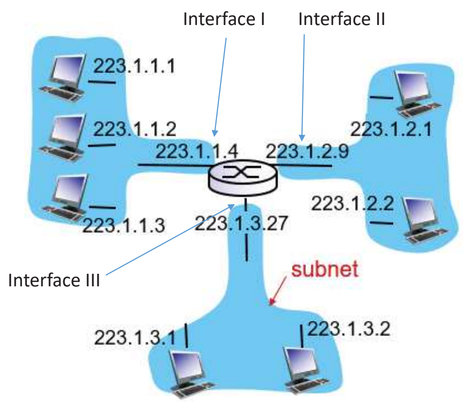
Practice #9

Hasidi Netanel

Today:

- Subnets

Application
Transport
Network
Link
Physical



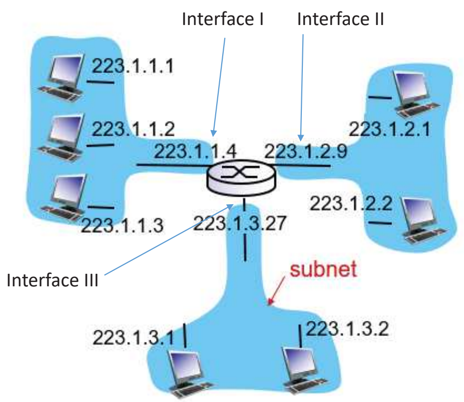
כתובת IP- כתובת בת 4 אוקטטות (8 ביטים בכל אחת) המזהה את הנתב או ה host.

$$223.1.1.1 = \underbrace{11011111}_{223}\ \underbrace{00000001}_{1}\ \underbrace{00000001}_{1}\ \underbrace{00000001}_{1}$$

Interface- היציאה הפיסית ("הרגל") היוצרת את החיבור בין הראוטר ל host. לנתב יכולות להיות כמה "רגליים". לכל רגל כתובת משלה. כל interface מפריד בין ה subnets.

Subnet- קבוצה של מחשבים שחולקים אותו prefix מסוים של כתובת ip. ניתן להעביר מידע בין מחשבים באותו ה subnet ללא מעורבות של נתב.

Classful Routing- יש 5 מחלקות (סוגים) של הקצאת כתובות המוגדרים היטב (A,B,C,D,E), לכל אחת יש את טווח הכתובות שלה.



Classless Routing- מס' הכתובות אזל, נאלצו לעשות שינוי => CIDR- לכל תת רשת יש prefix מסוים זהה. הרישום מתבצע בצורה a.b.c.d /x כאשר x היא "מסכת הרשת" (**subnet mask**) שאומרת כמה ביטים שייכים לאותו רצף.

subnet part | host part

11001000 00010111 00010000 00000000

200.23.16.0/23

טיפ: באמצעות "משקל" הביט משמאל למסיכה (subnet increment number) נוכל לדעת מה כתובת ה subnet הבא ולהקל בחישוב. בדוגמא הזו "הקפיצות" יהיו ב 2.

גודל subnet מקסימלי (מס' כתובות חופשיות)-

2 כתובת הן שמורות- אחת לכתובת ה subnet ואחת לכתובת broadcast.

$$2^{32-x} - 2$$

Q1

א. מה כתובת הרשת של host שכתובתו 150.123.45.80
ומסכת רשת של 255.255.255.192

192→ 2 bits of '1'→ /26

150.123.45.80 → A.B.C.01|010000 →
subnet address: 150.123.45.64 /26

ב. מה מס' המחשב ב subnet? 80-64 = 16

ג. מה מס' המחשבים המקסימלי ברשת?

$$6\ free\ bits\ \rightarrow 2^6 - 2 = 62\ hosts$$

Q2

א. ניתנת לכם כתובת 222.85.64.0 /24
עליכם לחלק את ה subnet לתתי רשתות בנות: 15 hosts , 28 hosts, 10 hosts

S2: 15+2 = 17 → 5 bit needed → A.B.C.001|… /27
S1: 28+2 = 30 → 5 bit needed → A.B.C.000|… /27
S3: 10/2 = 12 → 4 bit needed → A.B.C.0100|.. /28

	S1	S2	S3
Subnet Add.	222.85.64.0 / 27	222.85.64.32 / 27	222.85.64.64/ 28
Subnet Broadcast	222.85.64.31	222.85.64.63	222.85.64.79
Host Ranges	222.85.64.1 - 222.85.64.30	222.85.64.33 -222.85.64.62	222.85.64.65 - 222.85.64.78

ב. מס' כתובות מבוזבזות\לא מנוצלות:

$$(8-2-1)\cdot(2^5-2)+1\cdot(2^4-2)+(2^5-2-28)+(2^5-2-15)+(2^4-2-10)$$

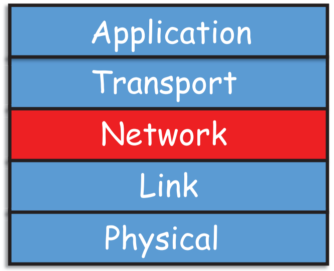
Introduction to Computer Networks

Practice #10

Hasidi Netanel

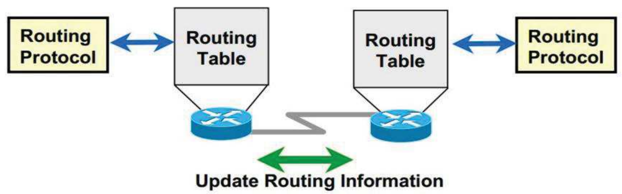
Today:

- ❑ Link State.
- ❑ Distance Vector.



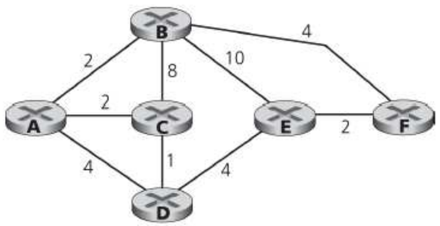
Routing Algorithms

- The purpose of a routing algorithm is to find a "good" path from source router to destination router.
- Need to consider also policy issues.
- A **global routing algorithm** computes the least-cost path between a source and destination using complete, global knowledge about the network (such as link state).
- In a **decentralized routing algorithm**, the calculation of the least-cost path is carried out in an iterative, distributed manner. (such as distance vector)



Link State

- Is a global algorithm – every node has the net topology and link costs known to all nodes.
- Computes least cost paths from one node ('source") to all other nodes.
- Iterative: after k iterations, know least cost path to k destination.



כיצד מחשב האלגוריתם את
המסלול הקל ביותר מקודקוד D
(למשל) לכל שאר הצמתים
ברשת?

Distance Vector

Initiation Stage:

d

N	D	hop
a	∞	-
b	∞	-
c	2	c
f	4	f
g	1	g

c

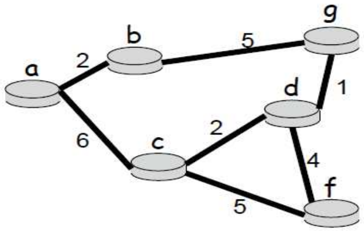
N	D	hop
a	6	a
b	∞	-
d	2	d
f	5	f
g	∞	-

b

N	D	hop
a	2	a
c	∞	-
d	∞	-
f	∞	-
g	5	g

a

N	D	hop
b	2	b
c	6	c
d	∞	-
f	∞	-
g	∞	-



g

N	D	hop
a	∞	-
b	5	b
c	∞	-
d	1	d
f	∞	-

f

N	D	hop
a	∞	-
b	∞	-
c	5	c
d	4	d
g	∞	-

Distance Vector

- Each node maintains a Vector which estimate the distances to all other nodes.
- ***It's Distributed*** - each node receives some information from one or more of its *directly attached* neighbors, performs a calculation, and then distributes the results of its calculation back to its neighbors.
- ***It is iterative*** - the process continues on until no more information is exchanged between neighbors.
- ***Is asynchronous*** in that it does not require all of the nodes to operate in lockstep with each other.
- When a node x receives new DV estimate from neighbor, it updates its own DV using B-F equation:

$$D_x(y) \leftarrow \min_v\{c(x,v) + D_v(y)\}$$

- IF there was a change in the DV, the node send it's new DV to it's neighbors.
- An iteration can also caused by local link cost change.

Distance Vector

- *C send its DV to D and F, causing their DV update.*

d

N	D	hop
a	8	c
b	∞	-
c	2	c
f	4	f
g	1	g

c

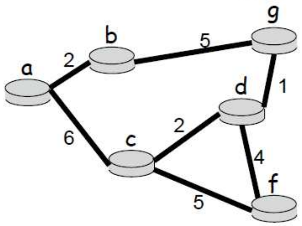
N	D	hop
a	6	a
b	∞	-
d	2	d
f	5	[illegible]
g	∞	-

b

N	D	hop
a	2	a
c	∞	-
d	∞	-
f	∞	-
g	5	g

a

N	D	hop
b	2	b
c	6	c
d	8	c
f	11	c
g	7	b



g

N	D	hop
a	∞	-
b	5	b
c	∞	-
d	1	d
f	∞	-

f

N	D	hop
a	11	c
b	∞	-
c	5	c
d	4	d
g	∞	-

Distance Vector

- *C and B send their DV, causing a DV update of node A.*

d

N	D	hop
a	∞	-
b	∞	-
c	2	c
f	4	f
g	1	g

c

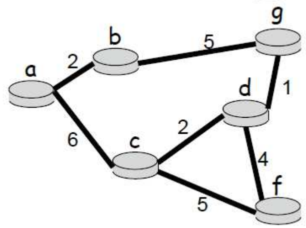
N	D	hop
a	6	a
b	∞	-
d	2	d
f	5	[illegible]
g	∞	-

b

N	D	hop
a	2	a
c	∞	-
d	∞	-
f	∞	-
g	5	g

a

N	D	hop
b	2	b
c	6	c
d	8	c
f	11	c
g	7	b



g

N	D	hop
a	∞	-
b	5	b
c	∞	-
d	1	d
f	∞	-

f

N	D	hop
a	∞	-
b	∞	-
c	5	c
d	4	d
g	∞	-

Distance Vector

- After second iteration.

מדוע באיטרציה השנייה
f "מכיר" את f ?

d

N	D	hop
a	8	c
b	6	g
c	2	c
f	4	f
g	1	g

c

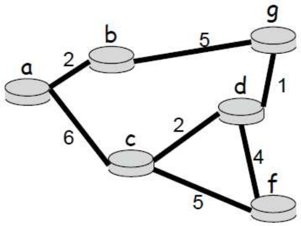
N	D	hop
a	6	a
b	8	a
d	2	d
f	5	f
g	3	d

b

N	D	hop
a	2	a
c	8	a
d	6	g
f	10	g
g	5	g

a

N	D	hop
b	2	b
c	6	c
d	8	c
f	11	c
g	7	b



g

N	D	hop
a	7	b
b	5	b
c	3	d
d	1	d
f	5	d

f

N	D	hop
a	11	c
b	10	d
c	5	c
d	4	d
g	5	d

Distance Vector

- After first iteration.

d

N	D	hop
a	8	c
b	6	g
c	2	[illegible]
f	4	f
g	1	g

c

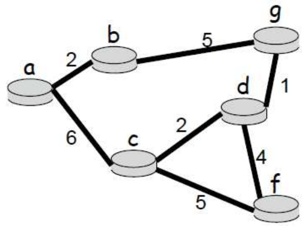
N	D	hop
a	6	a
b	8	a
d	2	d
f	5	[illegible]
g	[illegible]	d

b

N	D	hop
a	2	a
c	8	a
d	6	g
f	∞	-
g	5	g

a

N	D	hop
b	2	b
c	6	c
d	8	c
f	11	c
g	7	b



g

N	D	hop
a	7	b
b	5	b
c	3	d
d	1	d
f	5	d

f

N	D	hop
a	11	c
b	∞	-
c	5	c
d	4	d
g	5	d

Distance Vector

- **GNTF** – *Good News Travels Fast:*

After second iteration.

d

N	D	hop
a	3	c
b	6	g
c	2	c
f	4	f
g	1	g

c

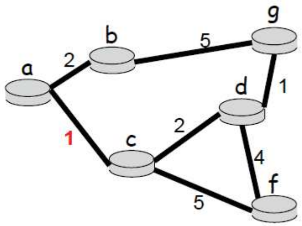
N	D	hop
a	1	a
b	3	a
d	2	d
f	5	f
g	3	d

b

N	D	hop
a	2	a
c	3	a
d	6	g
f	10	g
g	5	g

a

N	D	hop
b	2	b
c	1	c
d	3	c
f	6	c
g	4	c



g

N	D	hop
a	7	b
b	5	b
c	3	d
d	1	d
f	5	d

f

N	D	hop
a	6	c
b	10	d
c	5	c
d	4	d
g	5	d

Distance Vector

- **GNTF** – *Good News Travels Fast:*

We decrease an edge and see that after 2 iterations distances converge.

מי שולח את הוקטור
שלו בעקבות העדכון?

d

N	D	hop
a	8	c
b	6	g
c	2	c
f	4	f
g	1	g

c

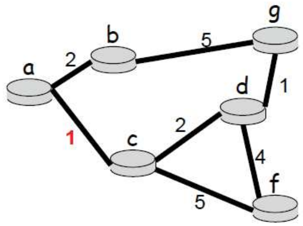
N	D	hop
a	1	a
b	8	a
d	2	d
f	5	f
g	3	d

b

N	D	hop
a	2	a
c	8	a
d	6	g
f	10	g
g	5	g

a

N	D	hop
b	2	b
c	1	c
d	8	c
f	11	c
g	7	b



g

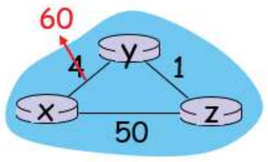
N	D	hop
a	7	b
b	5	b
c	3	d
d	1	d
f	5	d

f

N	D	hop
a	11	c
b	10	d
c	5	c
d	4	d
g	5	d

Distance Vector

- **BNTS**– Bad News Travels Slow.
- Known as Routing Loop Problem.
- ***Show*** that it requires 44 iteration until convergence.



Distance Vector

- ***GNTF*** – Good News Travels Fast:

After second iteration.

d

N	D	hop
a	3	c
b	5	c
c	2	c
f	4	f
g	1	g

c

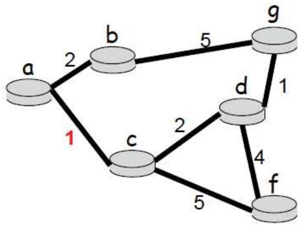
N	D	hop
a	1	a
b	3	a
d	2	d
f	5	f
g	3	d

b

N	D	hop
a	2	a
c	3	a
d	5	a
f	8	a
g	5	g

a

N	D	hop
b	2	b
c	1	c
d	3	c
f	6	c
g	4	c



g

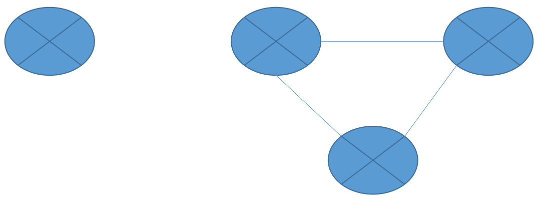
N	D	hop
a	4	d
b	5	b
c	3	d
d	1	d
f	5	d

f

N	D	hop
a	6	c
b	8	c
c	5	c
d	4	d
g	5	d

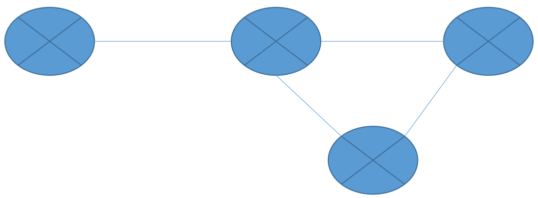
Distance Vector

- **BNTS**– Bad News Travels Slow.
- Can lead to ***Count to infinity problem*** – occurs when a node is removed from the network and the "gateway" to the removed node believes it can still reach it via a neighbor (who unknowingly is using himself as the access route).



Distance Vector

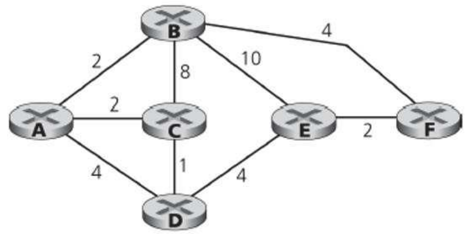
- **BNTS**– Bad News Travels Slow.
- Can lead to ***Count to infinity problem*** – occurs when a node is removed from the network and the "gateway" to the removed node believes it can still reach it via a neighbor (who unknowingly is using himself as the access route).



Distance Vector

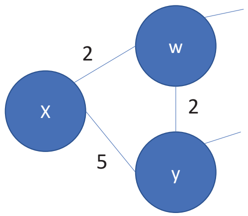
- **BNTS**– Bad News Travels Slow.
- Solutions:
  - **Split Horizon with Poisoned reverse**: If Z routes through Y to get to X: tells Y its (Z's) distance to X is infinite (so Y won't route to X via Z) . *Doesn't always solve the count to infinity problem (why?)*
  - **Defining Infinity** (RIP defines 16 as an infinity) – when reaching to this number we can conclude that the node isn't reachable.

Q1



Step	N'	D(A),P(A)	D(B),P(B)	D(C),P(C)	D(E),P(E)	D(F),P(F)
0	D	4,D	∞	1,D	4,D	∞
1	D,C	3,C	9,C	-	4,D	∞
2	D,C,A	-	5,A	-	4,D	∞
3	D,C,A,E	-	5,A	-	-	6,E
4	D,C,A,E,B	-	-	-	-	6,E
5	D,C,A,E,B,F	-	-	-	-	-

Q2



נתון ש: $D_w(u) = 5$ $D_y(u) = 6$

א. Distance vector של x:

$\boxed{D_x(y) = 4\ \ D_x(w) = 2\ D_x(u) = 7}$

ב. אילו קשתות נשנה כך ש x יעדכן את שכניו על מחיר חדש ל u?

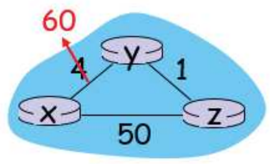
רק קשתות שקשורות ישירות ל x רלוונטיות לכן:

$if\ c(x,y) = \epsilon \leq 1\ \rightarrow D_x(u) = 1 + \epsilon$

הסבר: אם מחיר הקשת גדול מ1 אין מה לעדכן כי המסלול נשאר זהה במחיר 7.
אם מחיר הקשת 1 גם אין מה לעדכן כי המסלול לא משנה רק המחיר.

לגבי קשת c(x,w) כל שינוי במחיר יגרום לעדכן כאשר המחיר המפורסם ינוע בין 5 ל 11.

Q3



ההתכנסות היא אחרי 44 איטרציות:

לפני השינוי :

$\boxed{\begin{array}{l} D_y(x) = 4 \\ D_y(z) = 1 \end{array}}$ $\boxed{\begin{array}{l} D_z(x) = 5 \\ D_z(y) = 1 \end{array}}$

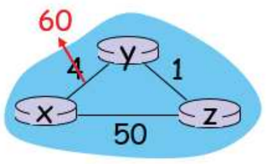
אחרי השינוי:

$$D_y(x) = \min\{c(y,x) + D_x(x), c(y,z) + D_z(x)\} = \min\{60 + 0, 1 + 5\} = 6$$

כלומר כעת ה NH של y כדי להגיע ל x הוא z.
y מעדכן את השכן שלו z וכתוצאה מכך z מבצע עדכון:

$$D_z(x) = \min\{c(z,x) + D_x(x), c(z,y) + D_y(x)\} = \min\{50 + 0, 1 + 6\} = 7$$

Q3



ההתכנסות היא אחרי 44 איטרציות:

Y שוב מקבל את העדכון ובאותו אופן המרחק גדל ל 8 ושוב שולח את זה ל Z שמשנה את המרחק ל 9..
נקבל:

Y: 8 → 10 →… →48 → 50 → 51
Z: 7 → 9 → 11 →… →49 → 50

הערה: בשלב האחרון ש Z מקבל את העדכון על המרחק של 50 הוא מבצע (ה NH שלו הופך להיות x , למרות y כמובן לא יודע את זה)

$$D_z(x) = \min\{c(z,x) + D_x(x), c(z,y) + D_y(x)\} = \min\{50 + 0, 50 + 1\} = 50$$

וכאן זה סיום של "הפינג-פונג" בין y ל z.

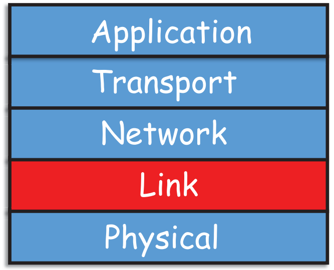
Introduction to Computer Networks

Practice #11

Hasidi Netanel

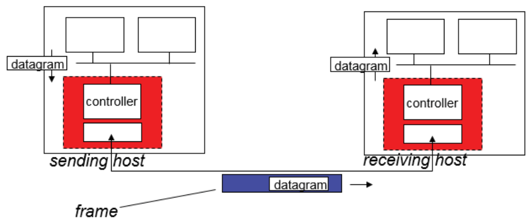
Today:

- ❑ Link Layer Services.
- ❑ Link Layer Addressing.
- ❑ ARP.



The Link Layer

- ***Link*** - communication channel that connect adjacent nodes along communication path .
- ***Data-link layer*** has responsibility of transferring datagram from one node to physically adjacent node over a link.
- Link layer is implemented in a network adapter (***NIC****- Network Interface Card*) as a combination of hardware, software and firmware.



Link Layer Services

- Link Access.
- Reliable Delivery.
- Flow Control.
- Error Detection.
- Error Correction.
- Half/Full Duplex.

(A services that repeat in transport layer, why? We can ask this question in both directions).

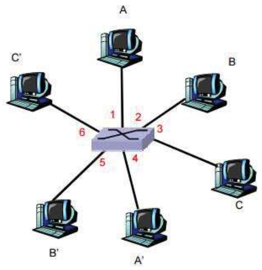
Link Layer Addressing

When switch receives a frame:

1. record link associated with sending host
2. index switch table using MAC dest address
3. if entry found for destination

```
        then {
                if dest on segment from which frame arrived
                        then drop the frame
                else
                        forward the frame on interface indicated
        }
    else flood // forward on all but the interface on which the frame arrived
```

Remember: switches are dumb.



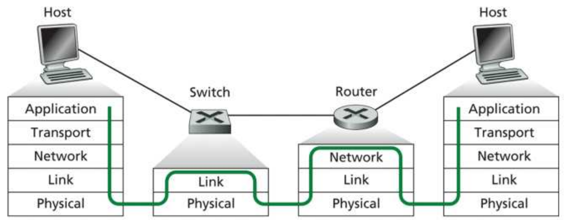
"Forward Table" for switches

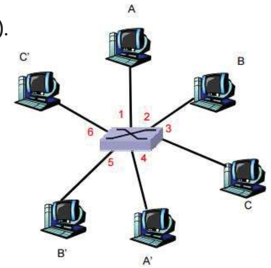
MAC addr	interface	TTL
A	1	60
A'	4	60

Switch table (initially empty)

Link Layer Addressing

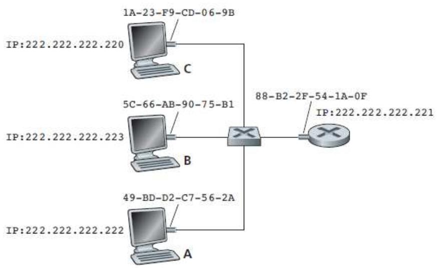
- ***MAC Address*** – 48 bit address, burned in NIC (unique identifier assigned to network interfaces).
  - Portable address. Ex: 1A-2F-BB-76-09-AD
- ***Switch***:
  - keeps a record of the *MAC addresses* of all the devices connected to it.
  - examine incoming frame's *MAC* address, ***selectively*** forward frame to one-or more outgoing links.
  - Transparent - hosts are unaware of presence of switches.
  - One Switch isn't aware of other switches.
  - Plug & play - switches do not need to be configured (self learning).



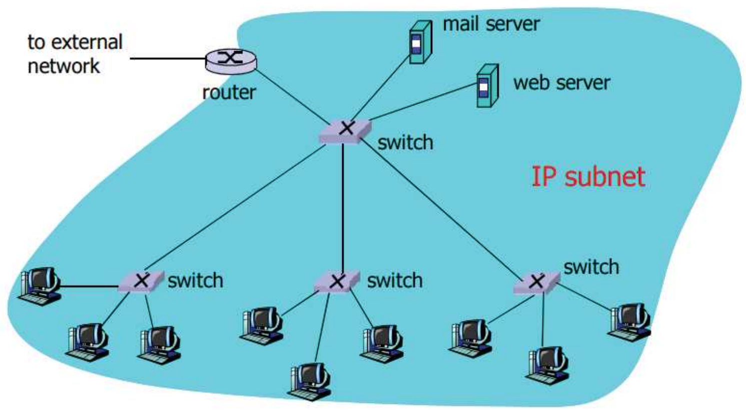


ARP (Address Resolution Protocol)

- Is used to map interface's *MAC* addresses, knowing its *IP* address.
- Remember: it's not the hosts and routers that have link layer address, but rather their adapters (network interfaces), therefore a node can have multiple addresses.
- Each host and Router has ***ARP Table*** which contains the mapping < IP address; MAC address; TTL> .
- **TTL** (Time To Live) – the time the mapping will be forgotten (typically 20 min).
- ARP tables are "plug & play".
- **ARP Query** – ("what is the MAC address of an IP a.b.c.d"?)
  - when node wants to send datagram to another node and it doesn't know it's destination MAC address, it sends *Broadcast* message with MAC address destination of all 'F'.
  - The other node replies with a unicast answer contains it's MAC address.



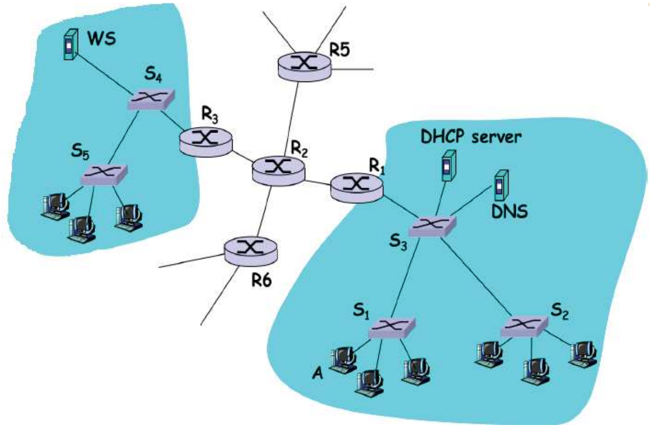
Link Layer Addressing



Q.

נתונה הרשת הבאה, ונתון כי צומת A רוצה לפתוח התקשרות TCP עם WS (TCP 3 handshake).

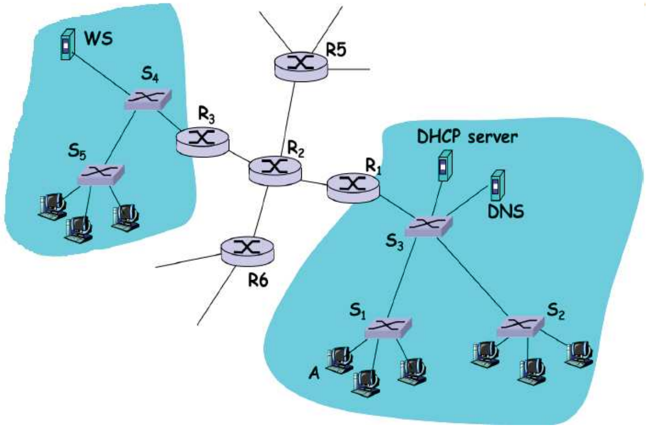
- כמה מסגרות ישודרו במהלך התהליך?
- הנחות לסעיף א':
  - אין איבודים של חבילות.
  - זו התעבורה היחידה ברשת.
  - מחשב A יודע את כתובת ה IP של WS.
  - טבלאות ה ARP מלאות.
  - טבלאות המיתוג מלאות.
  - טבלאות הניתוב מלאות.



Q.

נתונה הרשת הבאה, ונתון כי צומת A רוצה לפתוח התקשרות TCP עם WS (TCP 3 handshake).

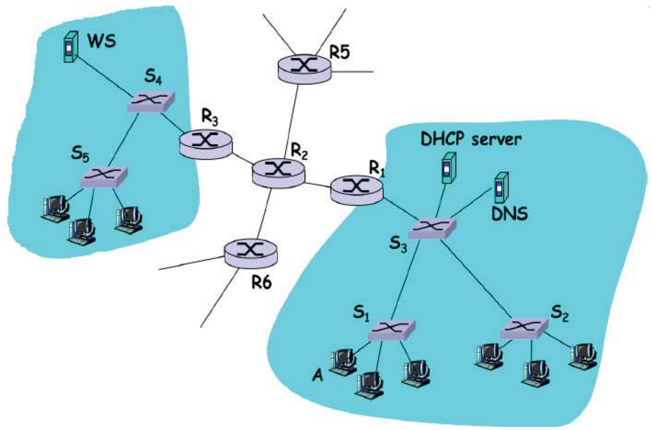
- כמה מסגרות ישודרו במהלך התהליך?
- הנחות לסעיף ב':
  - אין איבודים של חבילות.
  - זו התעבורה היחידה ברשת.
  - מחשב A יודע את כתובת ה IP של WS.
  - טבלאות ה ARP מלאות.
  - טבלאות המיתוג ריקות.
  - טבלאות הניתוב מלאות.



Q.

נתונה הרשת הבאה, ונתון כי צומת A רוצה לפתוח התקשרות TCP עם WS (TCP 3 handshake).

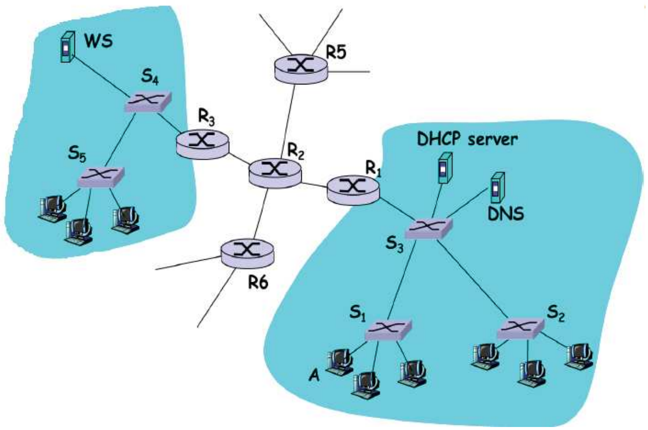
- כמה מסגרות ישודרו במהלך התהליך?
- הנחות לסעיף ג':
  - אין איבודים של חבילות.
  - זו התעבורה היחידה ברשת.
  - מחשב A יודע את כתובת ה IP של WS.
  - טבלאות ה ARP ריקות.
  - טבלאות המיתוג ריקות.
  - טבלאות הניתוב מלאות.



Q.

נתונה הרשת הבאה, ונתון כי צומת A רוצה לפתוח התקשרות TCP עם WS (TCP 3 handshake).

- כמה מסגרות ישודרו במהלך התהליך?
- הנחות לסעיף ד':
  - אין איבודים של חבילות.
  - זו התעבורה היחידה ברשת.
  - מחשב A לא יודע את כתובת ה IP של WS.
  - טבלאות ה ARP ריקות.
  - טבלאות המיתוג ריקות.
  - טבלאות הניתוב מלאות.
  - מחשב A יודע את ה URL של WS.
  - מחשב A יודע את ה IP של ה DNS.



הבהרה להתחלה: הודעה שעוברת מ-A ל-B דרך S1 נספרת כשידור של 2 מסגרות (הראשונה מ-A ל-S1 והשניה מ-S1 ל-B)

א. עם טבלאות מיתוג מלאות וטבלאות ARP מלאות

שלב	תהליך	כמות הודעות
1	A שולח הודעת SYN ל-WS. ההודעה עוברת דרך S1 (1), ממשיכה ל-S3 (1), ממשיכה ל-R1 (1), ל-R2 (1), ל-R3 (1), ל-S4 (1) ול-WS (1)	7
2	B מחזיר ל-A הודעת SYNACK (עוברת את אותו המסלול כמו קודם)	7
3	A שולח ל-B הודעת ACK	7
	סה"כ	**21**

ב. עם טבלאות מיתוג **ריקות** וטבלאות ARP מלאות

שלב	תהליך	כמות הודעות
1	A שולח הודעת SYN ל-B. ההודעה מגיעה עד S1 [A מתווסף לטבלת מיתוג]	1
	S1 עושה Flood להודעה (שולח לכולם חוץ מ-A, אך כולל S3)	3
	S3 עושה Flood להודעה	4
	S2 עושה Flood להודעה	3
	R1 קיבל את ההודעה (מ-S3) ומעביר אותה ל-R2, משם ל-R3 ומשם ל-S4 [R3 מתווסף לטבלת המיתוג של S4]	3
	S4 עושה Flood להודעה (2), S5 עושה Flood להודעה (3)	5
2	B מחזיר הודעת SYNACK (מכיוון ש-R3 ו-A בטבלאות המיתוג השונות, ההודעה נשלחת ישירות כמו קודם) [B מתווסף לטבלת מיתוג]	7
3	A שולח הודעת ACK ל-B. כמו קודם כי עכשיו טבלאות המיתוג מלאות במידע הרלוונטי.	7
	סה"כ	**33**

ג. עם טבלאות מיתוג **ריקות** וטבלאות ARP **ריקות**

שלב	תהליך	כמות הודעות
1	A רוצה לשלוח הודעת SYN ל-B, אך הוא לא יודע את ה-MAC של R. לכן הוא שולח ARP ב-Broadcast. כל תהליך ה-Broadcast מתרחש כמו קודם עם 1+3+4+3 הודעות.	11
	R1 מחזיר תשובה ישירה ל-ARP. מכיוון שטבלאות המיתוג כבר מכירות את A, הן יעבירו אליו ישירות.	3
	כעת A יודע את ה-MAC של R. הוא שולח את ה-SYN שמיועד ל-B. ההודעה מגיעה עד R1.	3
	R1 לא יודע את כתובת ה-MAC של R2 (!!) שולח ARP (הודעה אחת בלבד), מקבל תשובה (עוד הודעה) וכעת הוא יודע את ה-MAC, ויכול להעביר אליו את הודעת ה-SYN המקורית	3
	כנ"ל R2 עם R3	3
	טבלת הARP של R3 ריקה, ולכן הוא ישלח ARP ב-Broadcast כדי למצוא את ה-MAC של WS (הודעה אחת מ-R3 ל-S4, 2 הודעות מ-S4 ו-3 הודעות מ-S5)	6
	WS עונה על ה-ARP ישירות	2
	כעת R3 מכיר את ה-MAC של WS והוא יכול להעביר אליו את ה-SYN	2
2	החזרת תשובת SYNACK מתבצעת ישירות, מכיוון שבתהליכים שקרו קודם טבלאות ה-ARP והמיתוג התמלאו במידע הרלוונטי	7
3	שליחת ACK מתבצעת ישירות	7
	סה"כ	**47**

ד.

שלב	תהליך	כמות הודעות
0	(מציאת כתובת IP) A רוצה לשלוח שאילתה ל-DNS, אך הוא לא יודע את ה-MAC של DNS. שליחת ARP ב-Broadcast	11
	שליחת תשובה מ-DNS ל-A לשאילתת ARP	3
	שליחת שאילתת DNS מ-A ל-DNS	3
	שליחת תשובה מ-DNS ל-A לשאילתת DNS	3
1+2+3	כל התהליך כמו סעיף ג'	47
	סה"כ	**67**

Introduction to Computer Networks

Practice #12

Hasidi Netanel

Today:

- MAC Protocols.

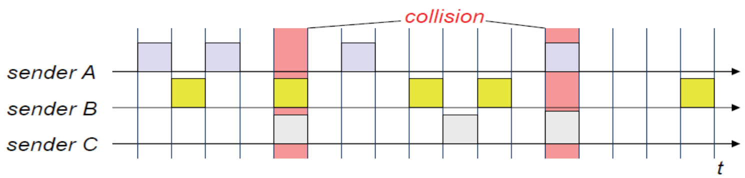
Application
Transport
Network
Link
Physical

MAC (Multiple Access Protocols)

- nodes are capable of transmitting frames at the same time.
- When this happens, all of the nodes receive multiple frames at the same time, which cause the frames to collide.
- Collide messages can't be interpreted correctly at the receivers.
- Therefore all frames involved in the collision are lost and the channel is wasted.
- MAC responsibility is to ensure useful work of the channel, which done through coordination of the transmissions of the active nodes.
- Human analogy:
  - "Give everyone a chance to speak."
  - "Don't speak until you are spoken to."
  - "Don't monopolize the conversation."
  - "Don't interrupt when someone is speaking."
  - "Don't fall asleep when someone is talking."

Random Access Protocols

- A transmitting node always transmits at full rate.
- When there is a collision, each node involved in the collision retransmits (not necessarily right away) its frame until its frame gets through without collision.
- **Slotted ALOHA**
- Time divided into slots size L/R sec (assume all frames consist exactly L bits).
- Evert node can transmit at each slot but only at the beginning of the slot.
- If no collision → node can send new frame in the next slot.
- If collision → node retransmits the frame in the next slot (with prob. P).



CSMA (Carrier Sense Multiple Access)

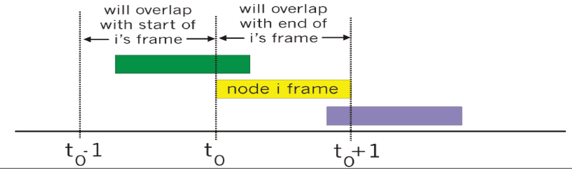
- Human analogy: "don't interrupt others ! ".
- Before transmitting, we check if the channel is empty.
- Even if we can't sense transmission it doesn't mean there is no transmission !
  - Consider A starts transmit at time t0.
  - Consider B starts transmit at time t0 + Tprop – Epsilon.
  - Only at time t0 + Tprop, B can sense the transmission of A.
- Distance & propagation delay play role in the collision probability.

1. Wait until there is data to send.
2. Sense the channel:
   A. If the channel is busy, wait random time and go back to 2.
   B. If the channel is clear → start transmitting.
      a. If there wasn't collision → back to 1.
      b. If there was collision → draw random time and go back to 2.

Random Access Protocols

- A transmitting node always transmits at full rate.
- When there is a collision, each node involved in the collision retransmits (not necessarily right away) its frame <u>until</u> its frame gets through without collision.
- **Pure ALOHA**
- Unslotted, no synchronization, everyone can transmit whenever it wants.
- Collision probability increase.
- If no collision → node can send new frame in the next slot.
- If collision → node retransmits the frame immediately (with prob. P) or wait time = 1 frame transmission time (with prob. 1-P) and so on.

  - frame sent at t_0 collides with other frames sent in $[t_0-1, t_0+1]$

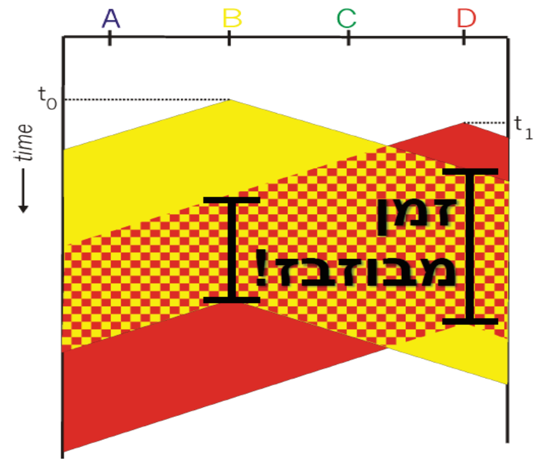


CSMA/CD (Collision Detection)

- If we know that there was a collision we don't need to continue the transmission (packet in any case goes to "garbage").
- Someone else talking → stop talking !

1. Wait until there is data to send.
2. Sense the channel:
   A. If the channel is busy, wait random time and go back to 2.
   B. If the channel is clear → start transmitting.
      a. If there wasn't collision → back to 1.
      b. If there was collision → Stop transmission.
         Send ***Jam Signal (48 bit)***.
         Wait random time and go back to 2.

CSMA (Carrier Sense Multiple Access)



Q.1

נתונים שלושה צמתים בערוץ בו משתמשים
בפרוטוקול slotted ALOHA. בהנחה וכל
הצמתים רוצים לשדר בכל עת ו p היא
ההסתברות שצומת תשדר בכל slot שהוא:

א. מה ההסתברות שצומת כלשהי תוכל לשדר
בהצלחה פאקט?
ב. מה ה efficiency המקסימלי במקרה כזה?

Binary Exponential Backoff

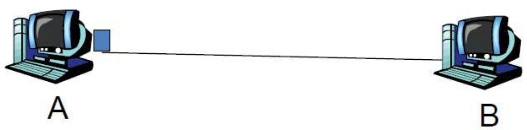
- What is the time each node need to wait until trying to transmit the frame again?
- A node chooses the value of K at a random from the set {0,1,2,…,2^n -1}, where ***n*** is the number of the collision.
- ***Wait Time*** = K * T , where T is the time it requires to send 512 bits.
- So for n = 1 we get {0,1}

  n = 2 we get {0,1,2,3}

  n = 3 we get {0,1,2,3,4,5,6,7} and so on.
- The limit is n = 10.
- In this technique we get less chance to collision again when there were a lot of collisions before (we draw a number from bigger set as ***n*** grows !).

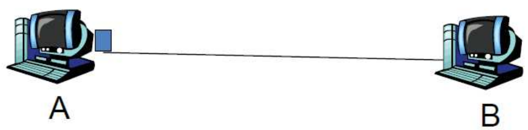
Q.2

- בהנחה שהתרחשה התנגשות, עבור N=2 למחשב A, כמה זמן ביטים יעברו (מההתחלה) לכל היותר עד שיוכל מחשב A להאזין לפני שידור.



Q.2

- נתונים 2 מחשבים A ו-B המשדרים זה לזה בשיטת CSMA/CD. המחשבים במרחק 4300 מטר זה מזה ומשדרים ע"ג תשתית Ethernet של 10Mbps. גודל הפריים הוא 512 ביטים.
- A התחיל לשדר בזמן 0 ולפני שסיים חבילה, B גם מתחיל לשדר. האם A יצליח לסיים את התשדורת לפני שהוא יחזה בהתנגשות?



תרגיל 12

Q1

א) הסתב' שצומת תשדר ב slot היא p
הסתב' שצומת לא תשדר ב slot היא 1-p.

עבור כל' שצומת יצליח לשדר ב slot, ה-2 האחרות חייבות לא לשדר באותו slot

$p(1-p)\cdot(1-p)$

ובגלל שיש 3 צמתים $3p(1-p)^2$.

ב) כדי למקסם את הביטוי נגזור אותו

$$\Rightarrow \frac{d}{dp}\left(3p(1-p)^2\right) = 3\frac{d}{dp}\left(p(1-p)^2\right) =$$

$$= 3\left[p\cdot 2(1-p)\cdot(-1) + (1-p)^2\right] = 3\left[2p^2-2p+1-2p+p^2\right]$$

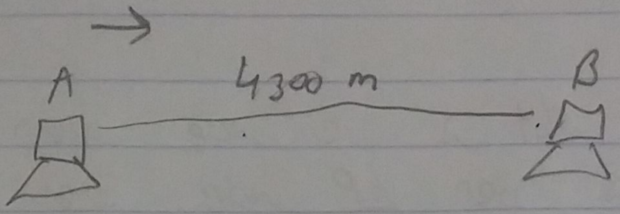
$$= 3\left[3p^2-4p+1\right]$$

$$\Rightarrow = 0 \Rightarrow \quad 2p\cancel{(1-p)} = (1-p)^{\cancel{2}}$$

$$3p = 1 \Rightarrow p = 1/3 \,//$$

$$\Rightarrow \text{max Efficiency} = 3\cdot\frac{1}{3}\left(1-\frac{1}{3}\right)^2 = \boxed{4/9}$$

$\boxed{Q_2}$



$R = 10$ Mbps

$|F| = 512$ bit

א) $T_{prop} = d/s = \frac{4300\,m}{2\cdot 10^8\,m/s} = 0.0215\,ms$

$L = $ # Bits Transmitted in T_{prop}

$\Rightarrow L = R \cdot T_{prop} = 10\cdot 2^{20} \cdot \frac{0.0215}{1000} = \lceil 225.44 \rceil = 226$ bits

אחרי 226 bits B יזהה את ההתנגשות עם A ולכן
סוף השידור הוא יוכל לשדר אחרי 225 ביטים
של A שידר (לפני ש-B משדר הוא בודק את הקו ומוודא שהוא פנוי).

כלומר A יזהה התנגשות אחרי $451 = 225+226$
ביטים. $451 < 512$ ולכן A לא יסיים
לשדר.

ב) $n=2$. אחרי 451 ביטים A יזהה התנגשות.
A שולח אות jam signal באורך 48 bit.
כלומר אחרי $451+48 = 499$ bits הוא יהיה בשקט.

אז הוא ימתין לפי המתנה: נבחר $k=3$ מהטווח $\{0,1,2,3\}$

לכן הוא ימתין זמן $k \cdot 512 = 1536$ bits

סה"כ זמן המתנה = זמן שקט של 2035 bits.