פקולטה: מדעי הטבע

מחלקה: מדעי המחשב ומתמטיקה

שם הקורס: תקשורת ומחשוב

קוד הקורס: **2-7036110-2**

תאריך בחינה 8/10/15 : סמ' ג מועד א

משך הבחינה: **3 שעות**

שם המרצה: עמית דביר

חומר עזר :**סגור**, מצ"ב חומר עזר כחלק מהבחינה

שימוש במחשבון :**כן** סוג: **רגיל**

פירוט הניקוד לכל שאלה:

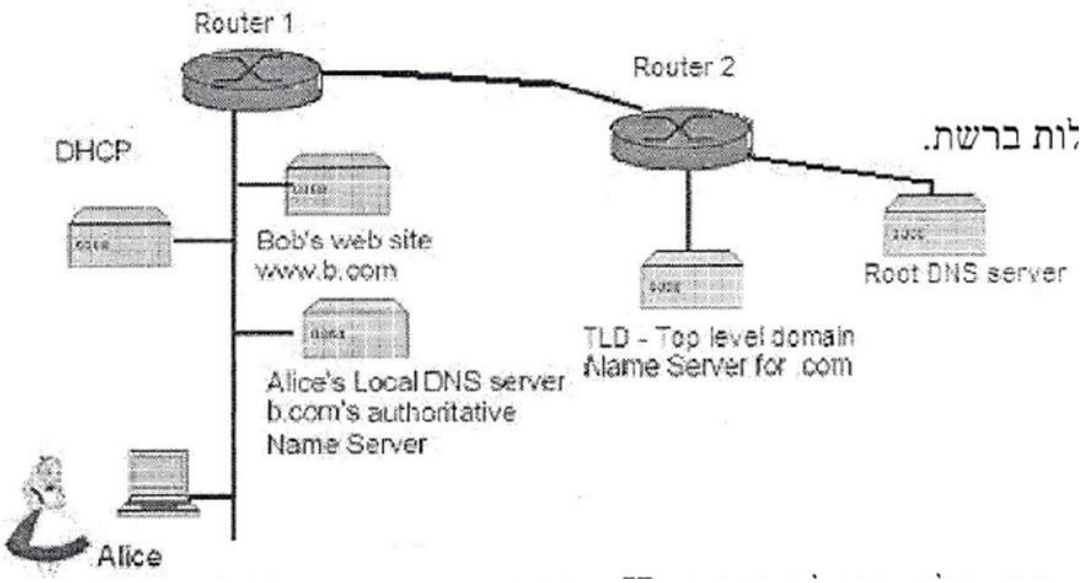
שאלה	ניקוד	מתוך
1	15	
2	15	
3	15	
4	15	
5	10	
6	20	
7	10	
סה"כ	100	

הוראות כלליות: **שימו לב כי בסוף הבחינה יש חומר עזר. הניסוח הוא בלשון זכר**

מתאמי נוחות ומתייחס לכולם!

חלק א (60 נקודות)

שאלה 1 (15 נקודות): אליס



אליס רוצה להגיע לאתר של בוב. תאר את התהליך כולו מהרגע שאליס מתחברת לרשת ועד הגעת כל האתר אליה חזרה. טבלאות הניתוב מעודכנות וטבלאות ה-DNS ריקות. הסבירו מה צריך להיות היחס בין ה-MTU לגודל הדף שאנחנו רוצים להעביר.

Protocol	S. Port	D. Port	S. IP	D. IP	S. MAC	D. MAC	Short Explanation

שאלה 2 (15 נקודות): שכבת האפליקציה

- **(5 נק')** מדוע DHCP עובד מעל UDP? האם יש הגיון או יתרונות להחליף את השימוש ל-TCP? הסבר
- **(5 נק')** מדוע DNS עובד מעל UDP? האם יש הגיון או יתרונות להחליף את השימוש ל-TCP? הסבר
- **(5 נק')** נניח כי לקוח ביקש משרת אובייקט מסויים שגודלו יותר מה-MSS, כיצד זה יראה ברשת ואיפה בדיוק הודעת ה- http response בכל הסיפור?

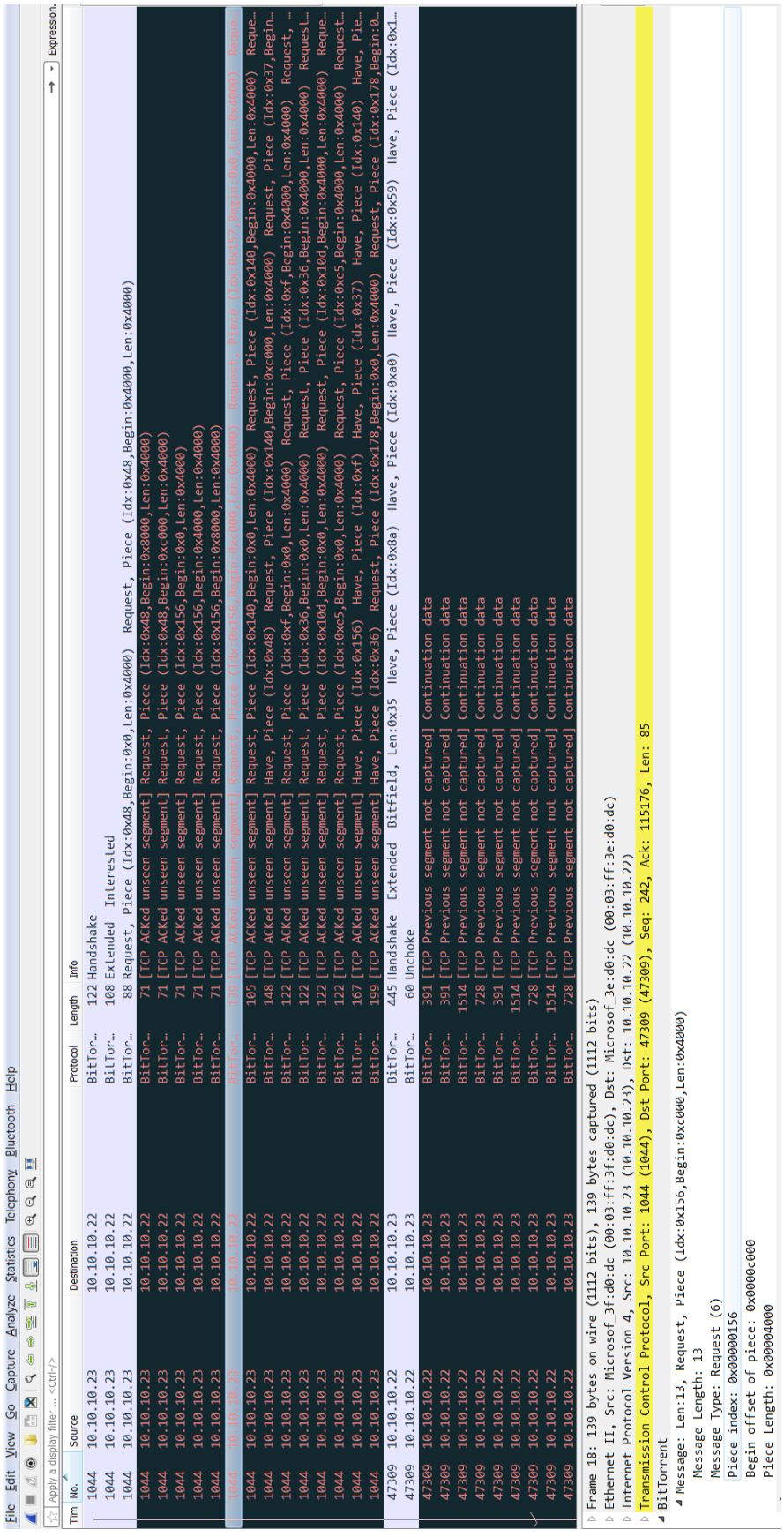
שאלה 3 (15 נקודות): שכבת התעבורה ואפליקציה

- **(3 נק')** הסבירו מדוע יש חשיבות הן ל- congestion control and flow control.
- התבונן על התמונה הבאה

```
C:\>nslookup www.nickphone.com
Server:  dns1.bgu.ac.il
Address:  132.72.140.46

Non-authoritative answer:
Name:    nickphone.com
Address:  132.72.23.184


C:\>nslookup nap.cse.bgu.ac.il
Server:  dns1.bgu.ac.il
Address:  132.72.140.46

Name:    nap.cse.bgu.ac.il
Address:  132.72.23.184
```

  - **(2 נק')** הסבירו כל שורה בתמונה
  - **(2 נק')** הסבירו את משמעות העובדה כי כתובת132.72.140.46 חוזרת על עצמה פעמיים
  - **(2 נק')** הסבירו את משמעות העובדה כי כתובת 132.72.23.184 חוזרת על עצמה פעמיים
- **(6 נק')** אם נפתח במחשב שני דפדפנים וניגש לשני אתרים שונים, כיצד נוכל בעזרת wireshark לזהות למי שייכת חבילה מסויימת (לאיזה/מאיזה אתר)

שאלה 4 (15 נקודות): wireshark, הסבירו את התמונה מה ההודעות ואיזה תהליך מתקיים פה



שאלה 5 (10 נק'), הסבירו את תשובתכם כל סעיף 2.5 נקודות

- בשכבת התעבורה קיים מנגנון בדיקת שגיאות. מדוע אם כך יש צורך לבצע בדיקת שגיאות גם בשכבת הערוץ?
- בטבלת המיתוג של המתג (switch) יש מיפוי לכל המחשבים ברשת. האם באופן תיאורטי יכול A להשתמש במידע שיש במתג על מנת לקבל את כתובת ה MAC - של B במקום לבצע ARP לכל המחשבים ברשת? אם כן הסבירו כיצד ניתן לבצע זאת, אם לא, הסבירו מדוע.
- חברכם לכיתה העלה קובץ לאתר מסויים ואתם רוצים להוריד אותו. כאשר אתם מבקשים את הקובץ אתם מקבלים הודעת שגיאה

  Error 404 - The document that has been requested either no longer exists,
  or has never existed on the server.

  הסבירו האם הסיבה לכך היא שחברכם כבר לא מחובר לשרת.
- לאחר שהורדתם את הקובץ ממקור לא ידוע (לא לעשות זאת יותר) אז פתאם כל פעם שאתם מבקשים דף אינטרנט מסוים אתם מקבלים פרסומות לפני שאתם מקבלים את האתר. הסבירו כיצד הדבר יכול לקרות.

חלק ב (20 נקודות)

שאלה 6 (20 נקודות): תכנות - משחק בול-פגיעה בין תוכנת לקוח ותוכנת שרת.

תיאור המשחק:

תכנית השרת תבחר מספר כלשהו בן 4 ספרות בצורה אקראית (כל ספרה בין 1-9, בלי חזרות, כלומר – כל ספרה תופיע במספר פעם אחת לכל היותר).
לדוגמה, 1234 הוא מספר חוקי, 1223 ו-0123 אינם חוקיים. וזה המספר אותו המשתמש (תוכנית הלקוח) צריך לנחש:

- אם אחת הספרות במספר המנוחש זהה לספרה במספר המקורי וממוקמת נכון זה בול.
- אם אחת הספרות במספר המנוחש זהה לספרה במספר המקורי ואך ממוקמת במקום שונה זאת פגיעה.

על תוכנית השרת להודיע עבור כל ניחוש כמה בולים היו וכמה פגיעות.

חיבור בין תוכנת השרת ותוכנת הלקוח יימשך עד אשר המשתמש ינחש את המספר במלואו (4 בולים) ואז תשלח דרך סוקט הודעה מתאימה (לדומה, congratulations!) והודעה האומרת כמה ניחושים היו עד הניצחון (כולל הניחוש האחרון המוצלח) או עד לקבלת הודעת סיום המשחק מהמשתמש.

תכנית הלקוח צריכה לנחש מה המספר, כשאר בכל תור התוכנית תבקש מהמשתמש לנחש מספר ותקלוט את אותו ניחוש דרך סוקט.

עליכם לכתוב תכנית-לקוח בשפת Java אשר תפעיל מחשק בול-פגיעה. לאחר ניחוש ראשון, התוכנית בכל תור תשאל את המשתמש האם להמשיך משחק או לסיים. בבחירת סיום המשחק ע"י המשתמש, תוכנית הלקוח שולחת דרך סוקט הודעה מתאימה לשרת ובכך סוקט בצד השרת נסגר וגם נסגר סוקט בצד הלקוח.

ניתן ליישם את המערכת תחת פרוטוקול TCP או UDP.

להלן מחלקות ה-Java לשימושכם:

חבילת java.net: Socket, ServerSocket, DatagramSocket,
DatagramPacket, InetAddress

חבילת java.io: BufferedReader, InputStreamReader, PrintWriter,
DataOutputStream

חלק ג (10 נקודות)

שאלה 6 (10 נקודות): שאלת מחשבה,

- **(5 נק')** מה ההבדל בין עיקרון של SPDY לבין HTTP 1. 0 ?
- **(5 נק')** מהו היתרון בביצועים בין SPDY לבין HTTP 1. 0

והמחישו באמצעות דוגמא של כניסה לאתר (הורדת עמוד ראשי)

ו 3 תמונות מאותו שרת.

מושג	הסבר
Access Point	נקודת גישה לתקשורת Ethernet אלחוטי .נקודות הגישה מחוברות ביניהן בקשר קווי.
Ack	אישור על קבלת הודעה תקינה.
Address Resolution Protocol (ARP)	כדי לשלוח מסגרת לצומת ברשת הפנימית, צריך לדעת מהי הכתובת הפיזית שלה (MAC). על מנת לגלות כתובת זו נשתמש בפרוטוקול ARP. כאשר ידועה לנו כתובת הלוגית (IP) של התחנה נשלח שאילתה ARP ברשת לכל התחנות (broadcast). אם קיימת צמת פעילה ברשת בעלת כתובת ה- IP המבוקשת היא תחזיר הודעת ARP. המידע ייאגר בטבלה מקומית לצורכי מיפוי של IP לכתובת MAC וכמובן שלכל עדכון יהיה TTL משלו.
AIMD	**Additive Increase Multiplicative Decrease** אופן פעולה: בכל סבב (RTT) בו לא התרחש אירוע אובד, חלון העומס מועלה ב-MSS אחד ,כלומר ההוספה לערך נעשית בצורה חיבורית .לעומת זאת ,כאשר מתרחש אירוע אובדן ,חלון אחד ,כלומר ההפחתה מהערך נעשית -MSS העומס מורד למחצית מערכו הקודם ,אך לא מתחת לבצורה כפלית. **המוטיבציה :**אם לא התרחש אירוע אובדן לאחר זמן סבב אחד ,אזי יתכן שהרשת אינה עמוסה ושישנו רוחב פס נוסף שניתן לנצל לשליחת נתונים .לפיכך נעשה ניסיון להעלות במעט את חלון העומס ו"לגשש "האם ישנו רוחב פס נוסף ,כאשר אם נעלה אותו בערך גדול מדי ,אנו עלולים להגיע בבת אחת לערך בו יתרחש אובדן ,ולכן העלייה נעשית בקצב איטי . לעומת זאת ,כאשר מתרחש אירוע אובדן ,אזי אם נוריד את חלון העומס רק במידה קטנה , אנו עלולים עדיין להישאר בערך בו יהיה אובדן ,ועד שנגיע בחזרה לערך בו כבר לא יהיה , עלול לעבור זמן רב ,מה שיביא להשהייה גדולה בזמני ההגעה של החבילות ליעד ובבזבוז רוחב פס בשליחה מחדש של חבילות שאבדו .לפיכך חלון העומס מופחת לערך בו סביר להניח שאם רק בנקודה זו התרחש אובדן וקודם לכן לא היה ,אזי בערך החדש שוב כבר לא יהיה ,ולכן ההפחתה היא בקצב מהיר .מצב העלייה הליניארית נקרא גם "הימנעות מעומס."
AP	**Access Point**
API	**Application Program Interface**
Application Layer	רמת היישום אחראית לתמוך בישומי רשת ,וכוללת הרבה פרוטוקולים ,ביניהם HTTP לתמיכה ב-web, SMTP לתמיכה בדוא"ל ו-FTP לתמיכה בהעברת קבצים
ASCII	**American Standard Code for Information Interchange** קוד תקני בינלאומי ,הפורמט הנפוץ ביותר לקבצי טקסט במחשבים ובאינטרנט .קוד ASCII הבסיסי מאפשר ייצוג של 128 תווים שונים ,כל תו מיוצג ע"י 7 סיביות .לעומת זאת ,בקוד ASCII המורחב כל תו מיוצג ע"י 8 סיביות.
Autonomous Systems	הרשת מחולקת לאוטונומיות ,כאשר כל אוטונומיה יכולה להכיל מספר רשתות ,כך שהיא שהיא מעין "רשת אינטרנט קטנה" (למשל קמפוס בר-אילן יכול להיות אוטונומיה) אם נרצה להעביר הודעות בתוך אוטונומיה אז נשתמש בנתבים מסוימים של אותה אוטונומיה ,ואם נרצה להעביר הודעה לאוטונומיה אחרת ,נשתמש בנתב שיודע להעביר הודעות בין אוטונומיות .קיימים 3 סוגים של אזורים אוטונומיים: **1) STUB** - זהו אזור אוטונומי קטן שמחובר לרשת בחיבור אחד למשל חברה שמחוברת לספק אינטרנט שלה 2) Multihomed AS - דוגמא, חברה שמחוברת לאזורים אוטונומיים אחרים ביותר מחיבור אחד וזה משיקולים שונים של אמינות ועלויות למשל. אך למרות שחברה זאת מחוברת לכמה אזורים אוטונומיים היא אינה מוכנה להעביר הודעות מאזור אחד לשני. **3)** Transit AS - בדרך כלל ספקי אינטרנט זהו אזור אוטונומי שמחבר הרבה אזורים אוטונומיים אחד לשני.

Best Effort	שירות המשתדל להעביר חבילות ללא טעיות ובסדר המתאים ,אך לא מתחייב לכך
Border Gateway Protocol	זהו אלגוריתם Distance Vector כאשר הטבלה שכל נתב שמריץ את הפרוטוקול הזה שולח לשכניו כוללת נתיב שלם ליעד ולא רק השכן הבא. הצורך בכל הנתיב ולא רק במחיר והיעד טמון בשיקולים כלכליים וארגוניים ,כמו אי רצון של ספק אינטרנט אחד לנתב דרך מתחרה שלו וכדו. 'כמו כן ,יש אפשרות למנוע שליחת הודעות דרכי למתחרה שלי על ידי אי פרסום נתיב אליו.
Bridge	עובד בשיטת Store & Forward. המסגרת עוברת רק למי שהיא מיועדת לו. שקוף לתחנות הרשת. אין צורך להתקין את תצורת הרשת מפני שיש לו מנגנון לימוד עצמי. מחלק את הרשת למרחבי התנגשות מקומיים. משתמש בכתובת ה- MAC של היעד והמקור. יתרונות: • פעולה פשוטה ואין צורך בעיבוד גדול . • טבלאות הגשר נלמדות לבד. • מתאים לרשתות קטנות. חסרונות: • התעבורה מוגבלת לעץ הפורש. • לא חוסם broadcast (תקלה יכולה לגרום לעומס)
Broadcast	כתובת Broadcast ידועה כ "הכל אחד" (all ones) או בכתיב של בסיס 16 FF:FF:FF:FF:FF:FF: כלומר 48 ביטי '1'.
CDN	Content Distribution Network רשת לחלוקת מידע .כדי לשפר ביצועים ,פרט ל-Proxy. ספק המידע רוצה עותקים של התוכן שלו בכל העולם כך שכל לקוח יקבל את התוכן מהעותק הקרוב אליו . ה-CDN בנוי משרתי קצה הממוקמים במקומות אסטרטגיים בעולם ומכילים את כל המידע או רק את החלק הרלוונטי לאזור בו הם יושבים. התוכן מתעדכן משרתי המקור .עיקר מטרתם של שרתי הקצה הוא קיצור המרחק בין המשתמשים לשרתים.
Channel Partitioning	חלוקת הערוץ – לחלקים קטנים יותר (לדוגמא: חלוקת תדירויות, חלוקת זמנים וכו), לכל צומת ישנה בלעדיות על כל חלק. יעיל והוגן כאשר יש הרבה תשדורות. בזבזני כאשר ישנן מעט תשדורות.
Checksum	מציאת שגיאות פשוטות בלבד ללא מנגנון לתיקון שגיאות
Circuit Switching	במיתוג מעגלי, משאבים כמו מכלאים ורוחב פס מוקצים מראש ע"מ להבטיח את התקשורת בין מערכות קצה. לעומת זאת, במיתוג מנות אין הקצה של משאבים, והגישה אל המשאבים היא ישירה .האינטרנט היא רשת מיתוג מנות, ולכן, כאשר מערכת קצה אחת רוצה לשלוח מנה למערכת קצה אחרת, אין שום הבטחה מתי המנה הזאת תגיע – האינטרנט עושה את מירב המאמצים שהוא יכול לעשות ברגע נתון בשביל לשלוח מנה, אך יכול להיות מצב של עומס במתגים, ואז המנה סובלת מהשהייה. טלפוניה היא דוגמה לרשת מיתוג מעגלי – הקו שבו משוחחים שני אנשים מוקצה מראש, דבר המאפשר שליחת מידע בין השניים בקצב קבוע. ישנם שתי שיטות למימוש המיתוג המעגלי : FDMA, TDMA כאשר ה-TDM יעיל יותר מ-FDM אבל עדיין יש כאן הרבה בזבוז, מכיוון שכאשר לא משתמשים במשאבים שהוקצו לצורך התקשרות כלשהיא (למשל כאשר שני אנשים מדברים בטלפון ופתאום שותקים), אז אך התקשרות אחרת לא יכולה להשתמש במשאבים האלה.
Client_isn	שייך ל-TCP. הלקוח בוחר מספר אקראי אשר ישמש למספור ההודעות מהשרת אליו. הלקוח שולח מספר זה בהודעה הראשונה שהוא שולח לשרת בעת בקשתו להקים קשר (SYN)
Collision	התנגשות

Conditional GET	מטרתו היא לצמצם את זמן הגישה והיא עושה זאת על ידי כך שאם תאריך העדכון האחרון של הדף על השרת זהה לתאריך העדכון שכבר קיים ב-cache של המשתמש הוא לא יישלח שנית אלא הדף הקיים יוצג. השימוש בשיטה זו נעשה על ידי הוספת שדה ב- header, המבקש לשלוח את הדף רק אם הוא עודכן אחרי תאריך מסוים.
CongWin	חלון העומס – משתנה המשמש להגבלת קצב השליחה. השולח ישתמש בנוסחא: LastByteSent – LastByteAcked < = min { CongWin, RcvWindow }
Cookies	עוגיות. הדפדפן שומר את העוגיות, המכילות פרטים על המשתמש, ובכל פעם שהוא שולח בקשה לשרת, הוא מצרף ל-header את הפרטים השמורים בעוגיה המתאימה לשרת אליו הוא פונה. עוגיה יכולה להמחק באחת משתי דרכים – או שהמשתמש מוחק ידנית את הקובץ בו נשמר המידע, או ש"פג תוקפה
Count To Infinity	בעיה שיכול להווצר בעקבות שימוש בווקטור מרחקים, יכול להווצר מסלול מעגלי לתקופה מסויימת. ניתן למנוע אותו חלקית על ידי שימוש ב-Poisoned Reverse, לא מאפשרים מסלולים מעגליים.
CRC	Cyclic Redundancy Check
Datagram	שירות אותו מקבלים ברשת האינטרנט, זהו שירות של Best Effort, כלומר הוא משתדל להעביר חבילות בלי טעות אבל הוא לא מתחייב לכך. אין הקמת ערוץ שיחה. לחבילות של Datagram יש את כתובת ה- IP של מחשב היעד כאשר חבילות שונות מאותו מקור יכולות להגיע בזמנים שונים ובמסלולים שונים. השיטה טובה להעברת מידע בין מחשבים, שכן לפי תקן זה תחנות הקצה צריכות להיות חכמות ולבצע בעצמן פעולות בקרה והתאוששות משגיאות, בזמן שהרשת עצמה היא מאוד פשוטה, לכן כך עובדת שכבת הרשת של האינטרנט, כיוון שרשת זו היא מאוד הטרוגנית ומורכבת מרשתות רבות בעלות מאפיינים שונים, והביצועים של תחנות קצה ונתבים שונים, שונים אלו מאלו ולכן יש צורך בצורת תקשורת שתלויה כמה שפחות ברשתות המרכיבות אותה – היא מאפשרת לנו יותר גמישות, וכל רשת תומכת כפי יכולתה.
Decentralized Information	כל נתב מכיר רק את הסביבה הקרובה אליו, כלומר רק את מי שמחובר אליו באופן ישיר ועל פי המידע הזה בוחר את המסלול הטוב ביותר דרך הצומת הבא. האלגוריתמים הללו נקראים Distance Vector והם נמצאים בשימוש ברשת האינטרנט. נבדיל בין 2 סוגי אלגוריתמים – סטאטיים ודינאמיים. אלגוריתמים סטאטיים מניחים שאין שינוי בצורת הגרף, כלומר במספר הצמתים ובמספר הקשתות ומשקלן, ואילו אלגוריתמים דינאמיים יכולים לדגום את הרשת במרווחי זמן קבועים, או לבצע שינוי בכל פעם שמתרחש אירוע כמו למשל הוספת קשת.
Demultiplexing	פילוג. המחשב המקבל מעביר את הסגמנטים המתקבלים אל ה-socket הנכון על פי המידע שב-headers
Distance Vectors	בכל פעם שמישהו מקבל מידע הוא מפיץ אותו לשכניו, והדבר נמשך עד שאין שכנים שמחליפים בניהם מידע. באלגוריתמים אלה אין סיגנל עצירה כיוון שמידע יופץ כל זמן שיש להפיצו. אלגוריתמים אלה מבוזרים – כל צומת מתקשר רק עם שכניו, והם א-סינכרוניים – כל צומת שולח הודעות כשהוא מקבל אותם, מבלי לחכות לצמתים אחרים, כך שאפשרי שצומת מסוים יקבל עדכון 1 ואת עדכון 2 עוד לפני שצומת אחר יקבל את עדכון 1. לכל צומת יש טבלת מרחקים שאומרת מה המרחק המצטבר ליעד דרך כל שכן שמחובר ישירות לאותו צומת. רגיש לתקלות בנתבים, כלומר אם נתב מסוים מודיע בטעות אז סביר להניח שרוב הנתבים יעדכנו את הטבלאות שלהם בצורה לא תקינה.
DNS	Domain Name System

	מטרת שרת ה- DNS היא למפות בין שם השאת לבין ה- IP שלו. ישנם שרתי DNS רבים ברשת מכמה סיבות, כך שאין שרת אחד המחזיק את כל הכתובות אלא ישנה היררכיה של שרתים, הפונים אחד לשני לבקשת המידע. סוגי שרתי DNS: • שרתים מקומיים - הינם שרתי ה-default במחשבי הקצה של המשתמשים ששם כתוב לאן ללכת על מנת לקבל את הכתובת • שרתים סמכותיים – שרתים שיודעים למפות את כל הכתובות במרחב כתובות מסוים. לכל מרחב שמות יש לפחות שני שרתי DNS שהכתובת ידועה להם. כל שינוי של כתובת מצריך שינוי בשני השרתים. • שרתי שמות – לשרתים הללו פונים כאשר לא יודעים מאיפה להשיג את הכתובת. ישנם 13 שרתים כאלו כאשר הם אינם יודעים את המיפוי אלא יודעים לכוון אל השרת הסמכותי. פרוטוקול DNS הינו מצורת שרת-לקוח המשתמש ב-UDP בדרך כלל משום שההודעות קצרות. מנקודת מבטו של הלקוח, ה-DNS הינו קופסא שחורה אליה הוא מזין כתוב URL ומקבל את כתובת ה- IP שלו.
DNS Caching and Records	שרת ה- DNS שומר את הרשומות שלו ב- cache. לכל רשומה יש שדה TTL שאומר עד מתי יש לשמור את הרשומה.
e-Mail	דואר אלקטרוני. לאפליקצית דוא"ל יש 3 מרכיבים עיקריים: 1) תוכנת לקוח – שדרכה הלקוח קורא וכותב הודעות 2) שרת הדוא"ל – דרכו נשלחות ההודעות. השרת מורכב מהרבה תיבות דואר של משתמשים שונים. הוא מחזיק תור של הודעות יוצאות שצריכות להשלח לשרתים אחרים. 3) פרוטוקול הדוא"ל – הפרוטוקול העיקרי בו משתמשים לשליחת דואר אלקטרוני. שליחה – SMTP קבלה – POP3, IMAP, HTTP
FDMA	Frequency-Division Multiplexing הערוץ מקצה תדר לכל התקשרות שמתבצעת בו עד שהיא מסתיימת (יכולות להיות כמה התקשרויות באותו ערוץ).
FIN	שייך ל-TCP. דגל המציין בקשה לסגירת קשר. יוזם הניתוק (בדרך-כלל, הלקוח) שלוח הודעה שבה דגל ה- FIN שווה 1. מקבל ההודעה שולח ACK לאישור ולאחר וכן שולח שוב FIN בעצמו, למקרה שה-ACK ילך לאיבוד. היוזם נשאר עוד בקשר על מנת לקבל אישור של הצד השני וכן את ההודעת ה-FIN (השניה). אם קבל את ה- FIN שולח ACK וסוגר את הקשר
Forwarding	מתפקידי שכבת האינטרנט: העברת החבילות דרך הנתבים
Frame	יחידת המידע העוברת ברמת הערוץ
GET	צורת העברת המידע ע"י HTTP, המידע מועבר בכותרת, לאחר סימן שאלה ואין שימוש בגול ההודעה. זו הצורה הנפוצה ביותר באינטרנט
Global Info	מידע גלובלי כל נתב מכיר את הטופולוגיה של כלל הרשת ובוחר את המסלול הטוב ביותר מהמקור ליעד. האלגוריתמים המתאימים למודל הזה נקראים אלגוריתמי Link State.
Go-Back-N	בפרוטוקול GBN לשולח מותר לשלוח יותר מחבילה אחת בלי לחכות לאישור. קיימת הגבלה למספר הודעות מקסימאלי N ב-Pipeline. הטווח של מספרים סידוריים אפשריים

	לחבילות שנשלחו אבל עדיין לא התקבל עליהן אישור מוגדר כ"חלון" בגודל N. תוך כדי פעולת הפרוטוקול החלון זז קדימה. לכל חבילה יש Timer, וכאשר יש אירוע Timeout לחבילה מסוימת יש שידור מחדש של אותה חבילה. המקבל אין חוצץ ולכן כל החבילות שנשלחו אחרי חבילה זו ישדרו מחדש גם הן. נוצרת בעיה שככל שהערוץ פחות אמין יש יותר שליחות מחודשות של הודעות. העובדה שלמקבל אין חוצץ היא יתרון עבור רכיבים קטנים.
HEAD	צורת העברת מידע ב-HTTP. צורה זו דומה מאד ל-POST, כאשר בתשובה השרת לא מעביר את האובייקט המבוקש אלא רק את ה-header בלבד. נוכח לשימוש בזמן פיתוח ובדיקות
HTTP	HyperText Transfer Protocol פרוטוקול של רמת האפליקציה בו משתמשים ב-web. הוא מדמה מודל של שרת לקוח, כאשר הלקוח הוא הדפדפן אשר מבקש, מקבל ומציג אובייקטים, והשרת הוא שרת ה-web ששולח את האובייקטים ללקוח על פי בקשתו. פרוטוקול זה משתמש תמיד ב-TCP, כאשר מספר ה-port שאליו מתחבר הלקוח הוא 80. מאחר שנתונים אלו קבועים, קל לכתוב אפליקציות חדשות שיקבלו את האובייקטים משרתי ה-web. בתחילה השתמשו ב-HTTP 1.0 וכיום משתמשים ב-HTTP 1.1. לגרסה 1.1 יש תאימות אחורה ולכן תוכנות המסתמכות עליו יכולות להבין גם הודעות מאפליקציות שנכתבו בגרסה הקודמת. בגרסה הראשונה 1.0 לא שמרו על עקביות הקשר (non- persistent connection) אחר כך זיהו שניתן להשתמש באותו קשר למספר בקשות בגרסה 1.1 הפכו מצב זה למצב רגיל (connection persistent). ישנם שני סוגי הודעות ה-HTTP: בקשה (request) ותשובה (response). שני סוגי ההודעות כתובים בפורמט ASCII שהוא פורמט קריא וקל לבדיקה. שלוש צורות עיקריות להעברת המידע: 1) GET 2) POST 3) HEAD
IMAP	Internet Mail Access Protocol
Inter-AS	מחבר בין תת רשתות ע"י נתב. מתמקד לא רק בביצועים אלא גם באפשרות לקבוע מדיניות
Interface	ממשק חיבור בין מחשב קצה או נתב לרשת נקרא ממשק, בדרך כלל לנתבים יהיו שני ממשקים לפחות, כיוון שנתב מחבר לפחות בין שני רשתות (ממשק אחד לכל רשת). לעומת זאת למחשב קצה יהיה ממשק אחד. כתובות IP של נתבים או מחשבי קצה מקושרות לממשק. משמעותו שאם לנתב מסוים יש שני ממשקים יהיו לו שני כתובות IP. שיטה זאת לפעמים עוזרת באיתור תקלות ברשת כאשר אפשר לדעת איזה ממשק יוצר את התקלה
Interior Gateway Protocols	אלגוריתמי ניתוב פנימי בתוך אזור אוטונומי.
Internet Layer	רמת הרשת. מעבירה חבילות ממקור ליעד ברשת האינטרנט, תוך שימוש בצמתים הנקראים נתבים העברת החבילות ברמה זו מתבצעת בין הרשתות השונות המרכיבות את רשת האינטרנט, והנתבים הם אלה שמעבירים הודעות מרשת לרשת עד שמגיעים לרשת היעד, לכן הפרוטוקול צריך להיות מוכר בכל מחשב קצה ובכל נתב. יש שני סוגים של שירותים בסיסיים שניתנים ברמת הרשת : Virtual Circuit,

	Datagram
Intra-AS	תת-רשת המחברת תחנות הקצה. מתמקד בעיקר בביצועים
IP Address	כתובת מספרית בת 32 ביטים (גרסה 4) המזהה כל מחשב שמחובר כרגע לרשת אינטרנט. בכתובת זו משתמשים המחשבים והנתבים. בגרסה 6 ישנו שימוש ב-128 ביטים. מתייחסים אליה ככתובת לוגית.
ISP	Internet Service Provider הגישה לאינטרנט נעשית ע"י ספקי אינטרנט.
LastByteAcked	שייך ל-TCP. מס' הבייט האחרון שאושר שנתקבל ע"י הצד המקבל. מנוהל ע"י השולח
LastByteRcvd	שייך ל-TCP, מס' הבייט האחרון שהגיע משכבת הרשת ונכנס למכלא. מנוהל ע"י המקבל.
LastByteRead	שייך ל-TCP מס' הבייט האחרון שנקרא מהמכלא ע"י היישום של המקבל. גם משתנה זה מנוהל ע"י המקבל
LastByteSent	שייך ל- TCP מס' הבייט האחרון שנשלח לצד המקבל. משתנה זה מנוהל ע"י השולח
Latency	הזמן החולף מרגע שהלקוח יוזם את הקשר ועד שהוא מקבל את האובייקט המבוקש בשלמותו
Link Layer	שכבת הקישור. מעניקה לשכבת הרשת שירות של העברת מנות בין שני מתגים (או ממקור למתג וממתג ליעד). בכל מתג, שכבת הרשת מעבירה את המנה לשכבת הקישור, שאחראית להעביר את המסר למתג הבא ומשם בחזרה לשכבת הרשת. מנה מסויימת יכולה להיות מנוהלת ע"י פרוטוקולי שכבת קישור שונים במתגים שונים. כלומר, שכבת הרשת יכולה לקבל שירות שונה משכבת הקישור במתגים שונים.
Linked State	סוג אלגוריתמים שבעזרתם ניתן לדעת על כל טופולוגית ברשת
Logical Link Control	אחת משתי שכבות בתוך שכבת הערוץ. זוהי שכבה מקשרת לשכבות עליונות יותר,ולכן מספקת את השירותים הבאים: 1) תומכת בפרוטוקולים רבים משכבות גבוהות יותר כגון :IP 2) מספק שירותים אמינים: a. תקשורת אמינה או חצי אמינה b. זיהוי שגיאות
Loss Event	ארוע אובדן של חבילה ברשת. כאשר מתרחש אירוע אובדן, מניח השולח שיש עומס ברשת אשר גרם לכך שהחבילה הגיעה לתור מלא באחד הנתבים בדרך ולפיכך נזרקה ואבדה, ולכן מוריד את קצב השליחה שלו אירוע אובדן מזוהה ע"י התרחשות אחד מהמאורעות הבאים: 1) הופעת Timeout – פרק זמן מסויים מרגע שליחת הסגמנט, אם בחלוף פרק זמן הזה עדיין לא הגיע אישור על הסגמנט, הוא מניח כי סגמנט זה אבד ושולח אותו שנית. 2) הגעת 3 סגמנטים אל השולח בהם שדה מס' האישור זהה, לאחר שכבר הגיע אליו לראשונה סגמנט הנושא את אותו מס' אישור, כלומר 3 אישורים כפולים – כזכור, שדה מס האישור בכל סגמנט TCP מכיל מספר סידורי של הבייט הבא אשר היעד מצפה לקבל. כאשר נעשה כאן שימוש באישורים מצטברים, כלומר אם היעד ישלח למקור אישור המבקש מס' בייט מסויים, זה מעיד על כך שהוא כבר קיבל בשלום את כל הבייטים הקודמים לו, ולכן אם אחד הסגמנטים לא יגיע אליו בשלום, היעד לא יוכל לאשר את כל הסגמנטים שיבואו אחריו, ויאלץ להמשיך לשלוח למקור אישורים המבקשים את הבייטים שהוא עוד לא קיבל. לפיכך, אם בשלב מסויים רואה המקור שמגיעים אליו כל הזמן אישורים המבקשים את אותו מס' בייט, כלומר אין התקדמות במספרים, הוא מניח שאותו סגמנט לא הגיע בשלום ליעד, כלומר אבד ברשת.

Medium Access Control	ברשתות מקומיות יש צורך בתיאום הגישה למדיה המשותפת. שידור בצורה לא מסונכרנת בין מחשבי הקצה הנמצאים באותה רשת מקומית יצור הפרעות, לכן נרצה פרוטוקול שיתאם בין השידורים.
MSS	Maximum Segment Size גודל סגמנט מקסימלי בביטים
MTU	Maximum Transfer Unit size גודל מקסימאלי של חבילת IP. במקרה וגודל החבילה גדול יותר, פרוטוקול IP יפרק את החבילה לכמות החבילות הנדרשת על מנת שיוכל להעביר את כל ההודעה. פירוק חבילה לחלקים יכול להיות בכל אחד מהנתבים אך הרכבה תתבצע במחשב יעד בלבד זה כיוון שפעולת הרכבה היא פעולה מסובכת יחסית ופעולות אלו אנו מעוניינים לחסוך מהנתבים.
Multihomed AS	דוגמא חברה שמחוברת לאזורים אוטונומיים אחרים ביותר מחיבור אחד וזה משיקולים שונים של אמינות ועלויות למשל. אך למרות שחברה זאת מחוברת לכמה אזורים אוטונומיים היא אינה מוכנה להעביר הודעות מאזור אחד לשני.
Multiplexing	ריבוב. המחשב השולח מוסיף למידע header
Nak/NACK	אישור על קבלת הודעה פגומה.
Network Address Translation	פרוטוקול IP v4 מוגבל בכמות כתובות. כדי להרחיב את מרחב הכתובות, ניתן להשתמש ב-NAT. כל מחשב קצה ברשת יהיה בעל כתובת IP פנימית ובעזרתה יוכל להתקשר עם שאר המחשבים שברשת המקומית. בעת בקשת מידע מחוץ לרשת המקומיתף ישמור הנתב המקשר לרשת האינטרנט את הכתובת המקומית וה-port של המבקש, ובמקומם ישלח הודעה עם כתובת IPאמיתית ואליה מצור מספר port אשר לפיו יקשר הנתב בין התגובה שתחזור לבין מחשב הקצה שביקש אותה.
Open Shortest Path First	פרוטוקול זה מריץ Link State Routing Algorithm. משמעותו שכל נתב ברשת (הפנימית) יודע את טבלת המרחקים של כל הרשת.
Packet Loss	איבוד מנה מתרחש כאשר אותו תור של מנות מלא, ומנה שמגיעה למתג לא יכולה להיכנס לתור. או שהמנה שהגיע תלך לאיבוד, או שאחת המנות שנמצאות בתור תלך לאיבוד, וזו שהגיעה תתפוס את מקומה. מקרה של הגעה ליעד בצורה לא תקינה יכול גם להחשב כאיבוד מנה
Packet Switching	מיתוג מנות כאשר מקור רוצה לשלוח מסר, הוא מחלק אותו לחלקים שנקראים מנות. המנות האלה מועברות ברשת ע"י נתבים. רוב הנתבים משתמשים בשיטת Store and Forward כדי להעביר מנות: מתג לא שולח הלאה שום ביט של מנה שהוא מקבל עד שהיא לא מגיעה אליו בשלמותה. המנגנון הזה גורם להשהייה בנתבים. השהיה נוספת בשיטת מיתוג מנות נובעת כתוצאה מהמתנה בתור: לכל נתב יש תור של מנות שהוא רוצה לשלוח, שמחכות שהערוץ יתפנה לשליחתן. ההשהיה הזו תלויה ברמת העומס הקיימת ברשת.
Persistent with pipelining	כל הבקשות נשלחות על אותו קשר אחת אחרי השנייה ללא המתנה לתשובה.
Persistent without pipelining	כל הבקשות נשלחות על אותו קשר, בקשה לא תשלח לפני שהתשובה לבקשה הקודמת לא התקבלה במלואה.
Physical Layer	השכה הפיזית אחראית להעביר את הביטים הספציפיים של המנה ממתג למתג. הפרוטוקולים בשכבה הזאת

	תלויים במדיה הפיסית שבה הביטים משודרים.
Pipelining	מאפשר שליחת מספר הודעות אחת אחרי השניה (לפעמים מתייחסים לזה במקביל)
POP3	Post Office Protocol, פרוטוקול להורדת דואר
Port	שער שמזהה את האפליקציה באופן ייחודי על אותו המחשב
POST	צורת העברת מידע ע"י HTTP. המידע מעובר בגוף ההודעה
Proxy Server (web)	שומר ברשת באיזה שהיא נקודת אמצע דפים שכבר התקבלו בעבר, על מנת שאפשר יהיה להראות אותם מהר יותר בפעם הבאה. זהו מעין שרת אמצע שנמצא בשימוש על ידי ארגונים שרוצים לחסוך בתעבורה. חסרון של שיטה זו היא שהדפים עלולים להיות לא מעודכנים. יתרון השיטה היא שהיא חוסכת זמן ומקטינה את עומס תעבורת המידע ברשת. הדפדפן צריך להכיל קונפיגורציה כזו שתפנה את הבקשות שלו ל-proxy (אחרת הוא צריך להרים ולבצע DPI לכל ההודעות), אם אין ל-proxy את הדף הוא יפנה לשרת המקורי. חשוב לציין שבתהליך ה-proxy מתפקד הן כשרת (מול הלקוח) והן כלקוח (מול שרת ה-web)
RcvBuffer	שייך ל-TCP, גודל המכלא הקבלה. משתנה זה מנוהל ע"י הצד המקבל וערכו ידוע רק לו
RcvWindow	שייך ל-TCP, חלון קבלה – גודל המקום הפנוי במכלא. מנוהל ע"י המקבל
Router	התקן ברמת הרשת, עובד בשיטת Store and Forward, משתמש בכתובת IP על מנת לקחת החלטות ניתוב. מפריד בין תתי רשתות. יתרונות: 1) סוגים שונים של טופולוגיות נתמכות 2) לא מעביר Broadcast 3) מתאים לרשתות גדולות חסרונות: 1) דורש התקנה ומשאבים 2) מבצע פעולות עיבוד מורכבות
Routing Information Protocol	מפעיל distance vector algorithm כאשר מרחק מוגדר ככמות הנתבים בדרך אל היעד. כאשר המספר המקסימלי הוא 15 נתבים. כל נתב שמריץ את הפרוטוקול מפרסם לשכניו את טבלת המרחקים שלו כל 30 שניות. כאשר כל פרסום הוא של עד 25 יעדים שאליהם יש לי נתיבים הכי קצרים. טיפול של RIP בכישלונות: אם אחרי 6 סיבובים של פרסום או 180 שניות נתב לא מקבל משכן מסוים את הפרסום(טבלה) שלו הוא מכריז ששכן זה נפל ובהתאם מעדכן את הטבלה שלו ושולח אותה לשכניו
Round Trip Time	הזמן שלוקח לשלוח חבילה מהלקוח לשרת ובחזרה
Selective Repeat	הפרוטוקול מונע שליחה מחדש מיותרת בגלל שהשולח שולח מחדש רק את החבילות שאבדו או הגיעו משובשות. לשם כך המקבל צריך לשלוח ACKלכל חבילה שהגיעה תקינה בנפרד. נדרש כאן חלון בגודל N על מנת להגביל את מספר החבילות הבלתי מאושרות ב-pipeline. שלא כמו GBN הצד המקבל שולח ACK לחבילה תקינה שהתקבלה גם אם לא הגיעה בסדר הנכון, דבק זה מחייב חוצץ אצל המקבל. גודל החלון חייב להיות קטן או שווה לחצי הגודל של טווח המספרים הסידורים.
Server_isn	שייך ל-TCP, השרת בוחר מספר אקראי אשר ישמש למספור ההודעות מהלקוח אליו. השרת שולח מספר בהודעה הראשונה ששולח ללקוח (SYN+ACK) בעת הקמת קשר
Slow Start	התחלה איטית אופן הפעולה: בתחילת החיבור, עבור כל זמן סבב בו לא התרחש אירוע אובדן, חלון העומס

	איננו מועלה ב-MSS אחד אלא מוכפל פי 2 מערכו הקודם. תהליך זה נמשך עד אשר מתרחש אירוע האובדן הראשון, ואז הוא מורד למחצית מערכו ומכאן מתחיל לעלות כל הזמן בשיטת AIMD. המוטיבציה: כאשר נוצר חיבור TCP בין שני צדדים, גודלו של חלון העומס נקבע לערך התחלתי של MSS בודד אחד ומשם מתחיל לעלות בקצב לינארי (במידה ולא משתמשים בהתחלה איטית). לעיתים עלול להווצר מצב בו רוחב הפס האפשרי גדול, ולכן אם נעלה את קצב השליחה בצורה כזאת, עלול לעבור זמן רב עד שנגיע לערך הגורם לניצולת טובה של רוחב הפס, מה שיגרום לבזבוז בתחילת החיבור. לפיכך בהתחלה מכפילים כל פעם את גודלו של חלון העומס פי 2 וכך קצב השליחה עולה בצורה אקספוננציאלית, עד אשר מתרחש אירוע האובדן הראשון, ואז הוא מורד למחצית מערכו, כלומר חוזר לערך האחרון שלו בו לא היה אובדן. בשלב הזה אנו יודעים שאנו כבר קרובים הרבה יותר לקצב השליחה המקסימאלי האפשרי ולכן מכאן נמשיך בצורה לינארית.
Simple Mail Transfer Protocol (SMTP)	הפרוטוקול העיקרי בו משתמשים לשליחת דואר אלקטרוני, משתמש ב-port מספר 25. מבוסס TCP ומשמש עבור שליחת הודעות בין תוכנת הלקוח לשרת ובין שרת אחד לשני. כל ההודעות והתשובות נשלחות בצורה ASCII, כאשר היתרון בפורמט זה שהוא נוח לבדיקה אך חסרונו הרב הוא נפח ההודעה הגדול.
Socket API	ממשק לתקשורת בין האפליקציה והפורטוקול. בממשק יש לקבוע את סוג התעבורה, אופן ההתקשרות, זיהוי היעד הנעשה על ידי ה- IP של המחשב והפורט.
Statistical Multiplexing	המנות שנמצאות בתור של המתג לא נשלחות דרך הערוץ בסדר קבוע אלא באופן אקראי.
Stop & Wait	לאחר שליחת הודעה עוצר וממתין עד לקבלת אישור על ההודעה. רק לאחר קבלת האישור ממשיך לשלוח את הודעה הבאה.
STUB	זהו אזור אוטונומי קטן שמחובר לרשת בחיבור אחד למשל חברה שמחוברת לספק אינטרנט שלה
Switch	עובד בדרך כלל בשיטת Store & Forward, אך ישנם שעובדים ב-ut-Through Switching המסגרת עוברת רק למי שהיא מיודעת לו. נעשה שימוש ב-CSMA/CD. שקוף למשתתפים. צורך להתקין את תצורת הרשת מפני שיש לו מנגנון לימוד עצמי. מחלק את הרשת למר התנגשות מקומיים. משתמש בכתובת ה-MAC. מחבר, בדרך כלל, מספר רב של מקטעים. מאפשר מעבר של כל המסגרות שמגיעות בו זמנ בתנאי שלכל אחת מקטע יעד שונה. נקרא גם: Multiple Port Bridge.
SYN	שייך ל-TCP. בהודעה דגל המציין את תחילת ההתקשרות. הלקוח שולח הודעה כשדגל זה שווה ל-1 ונוסף אליה גם ה-clinet_isn. אם השרת יכול לקבל את הקשר, השרת ישלח בחזרה הודעת synack המאשרת את קבלת הקשר.
SYNACK	שייך ל-TCP, אישור של הרשת על הודעת SYN. בהודעה מצרף השרת את ה-server_isn וגם דגל SYN שווה 1.
Transmission Control Protocol (TCP)	פרוטוקול תעבורה אמין עם בקרת זרימה ובקרת עומס. מאפשר שירות connection oriented עם מנגנון פתיחת קשר כאשר שירות התעבורה נמצא בין קצוות הקשר (לדוגמא שרת/לקוח)
Time-Division Multiplexing (TDMA)	הזמן מחולק למסגרות בעלות משך זמן קבוע, וכל מסגרת מחולקת למס' קבוע של חריצי זמן. כאשר הרשת רוצה לבצע התקשרות בערוץ מסויים, היא מקצה חריץ זמן אחד בכל מסגרת לאותה התקשרות. כלומר, כל התקשרות שמתבצעת בערוץ משתמשת בכל רוחב הערוץ בצורה מחזורי ע"פ חריצי זמן שמוקצים לה.

Threshold	בהקשר TCP. גודל החלון בו יסתיים מצב ההתחלה האיטית ויתחיל מצב ההימנעות מעומס.
Time To Live (TTL)	מגדיר אורך חיים של נתון בטבלה/חבילה ברשת.
Timeout	זמן הנמדד הרגע שליחת ההודעה ועד הקביעה שכנראה ההודעה הלכה לאיבוד. זמן זה צריך להיות לפחות כאורכו של RTT אחד עם מקדם ביטחון.
Transit AS	בדרך כלל ספקי אינטרנט זהו אזור אוטונומי שמחבר הרבה אזורים אוטונומיים אחד לשני.
Transport Layer	רמת התעבורה. מעניקה לשכבת הישום שירות של העברת הודעות בין שני הצדדים המתקשרים של הישום.
User Datagram Protocol (UDP)	פרוטוקול תעבורה לא אמין חסר קשר. יתרונות: חסכון התקורה ביצירת החיבור, פרוטוקול פשוט. חסרונות: אין בקרת דחיסה ועומס
Uniform Resource Locator (URL)	מאפיין משאבים אחיד. למעשה זהו שילוב של המידע המוכל בכתובת IP, מידע המתאר מחשב מסוים, ומיקום המשאב במבנה הקבצים של אותו מחשב.
Virtual Circuit	הוא שירות של רשתות שונות אשר בונה נתיב מהמקור ליעד.
אמינות	בחירת שירותי התעבורה: בשירות זה כאשר מגיע מידע שגוי הדבר יזוהה ויישלח מחדש.
בקרת זרימה Flow control	אחד ממנגנוני האימות של TCP. השולח לא ישלח יותר ממה שהיעד יכול לקבל ולטפל.
בקרת עומס Congestion Control	אחד ממנגנוני האימות של TCP. התאמת קצב השידור לעומסים ברשת
זמן שידור Transmission time	זמן שלוקח לדחוף את המידע לתוך הקו. גודל הקובץ חלקי קצב שידור
ניצולת הערוץ	היחס בין זמן השידור בפועל לבין הזמן שעבר מתחילת שליחת הודעה ראשונה עד קבלת האישור עליה.

HTTP GET

GET /index.html HTTP/1.1\r\n

Host: www-net.cs.umass.edu\r\n

User-Agent: Firefox/3.6.10\r\n

Accept: text/html,application/xhtml+xml\r\n

Accept-Language: en-us,en;q=0.5\r\n

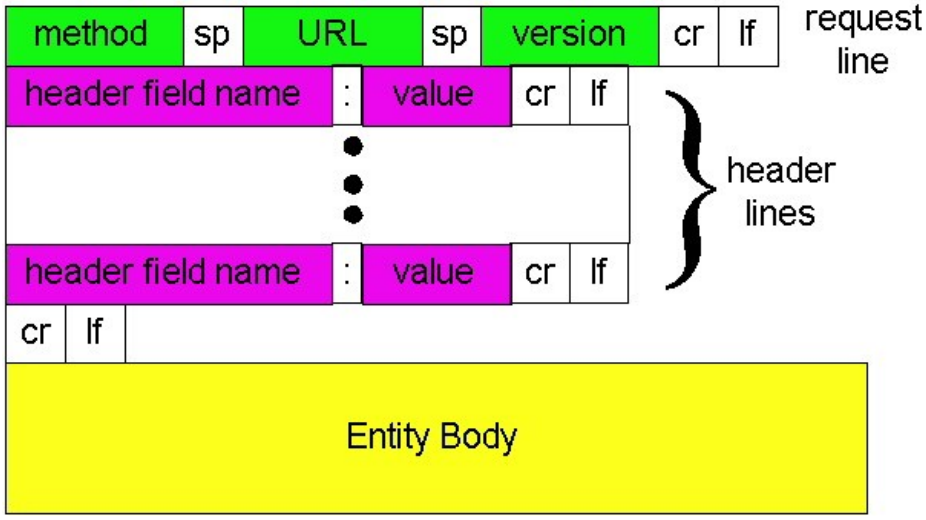
Accept-Encoding: gzip,deflate\r\n

Accept-Charset: ISO-8859-1,utf-8;q=0.7\r\n

Keep-Alive: 115\r\n

Connection: keep-alive\r\n

\r\n



HTTP Response

HTTP/1.1 200 OK\r\n

Date: Sun, 26 Sep 2010 20:09:20 GMT\r\n

Server: Apache/2.0.52 (CentOS)\r\n

Last-Modified: Tue, 30 Oct 2007 17:00:02 GMT\r\n

ETag: "17dc6-a5c-bf716880"\r\n

Accept-Ranges: bytes\r\n

Content-Length: 2652\r\n

Keep-Alive: timeout=10, max=100\r\n

Connection: Keep-Alive\r\n

Content-Type: text/html; charset=ISO-8859-1\r\n

\r\n

data data data data data ...

Sample SMTP interaction

S: 220 hamburger.edu

C: HELO crepes.fr

S: 250 Hello crepes.fr, pleased to meet you

C: MAIL FROM: <alice@crepes.fr>

S: 250 alice@crepes.fr... Sender ok

C: RCPT TO: <bob@hamburger.edu>

S: 250 bob@hamburger.edu ... Recipient ok

C: DATA

S: 354 Enter mail, end with "." on a line by itself

C: Do you like ketchup?

C: How about pickles?

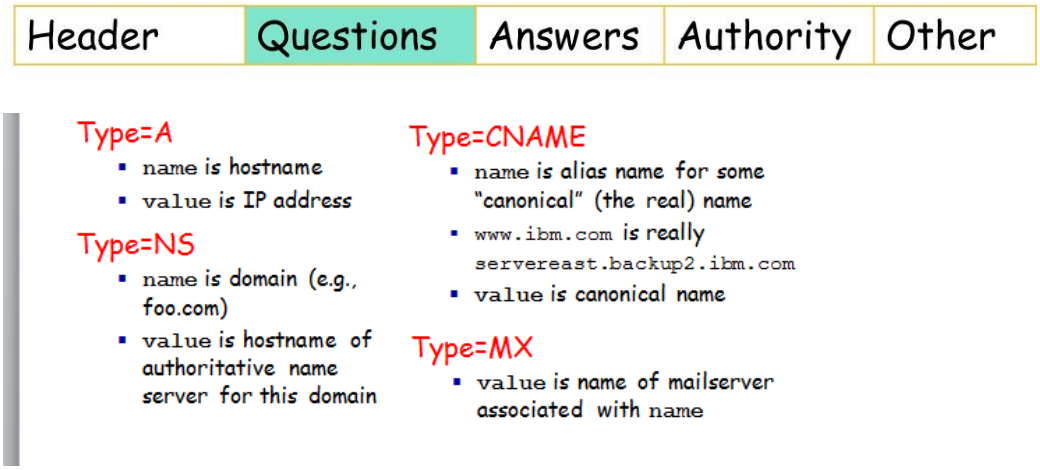
C: .

S: 250 Message accepted for delivery

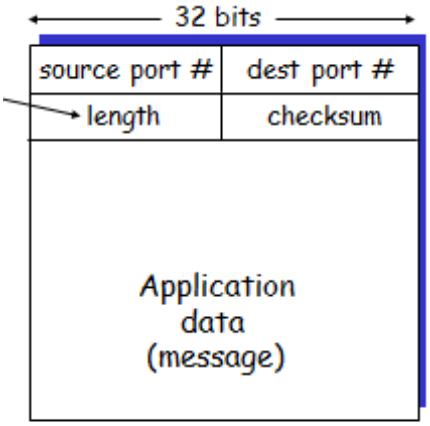
C: QUIT

S: 221 hamburger.edu closing connection

DNS Message

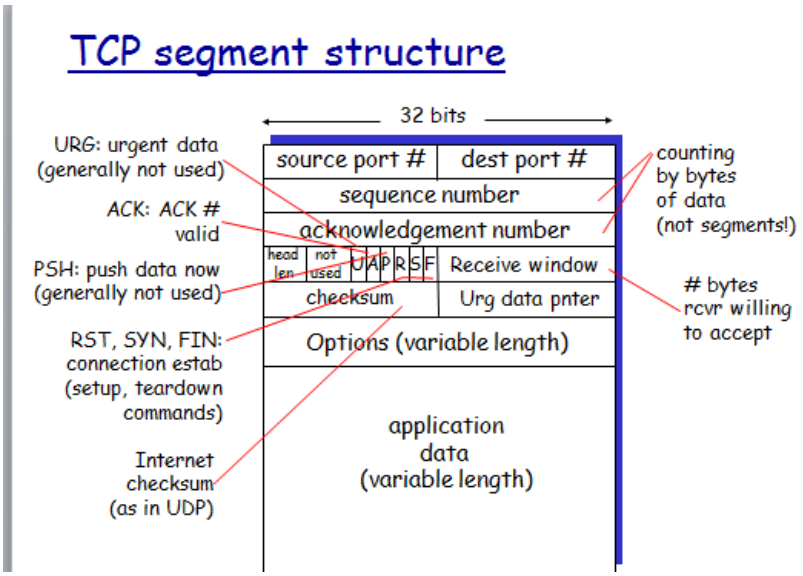


UDP

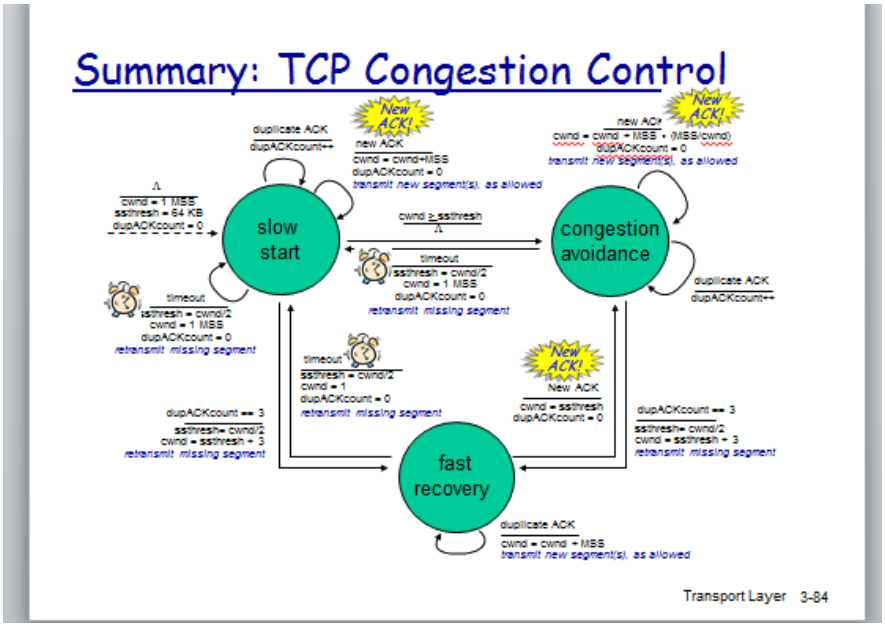


UDP segment format

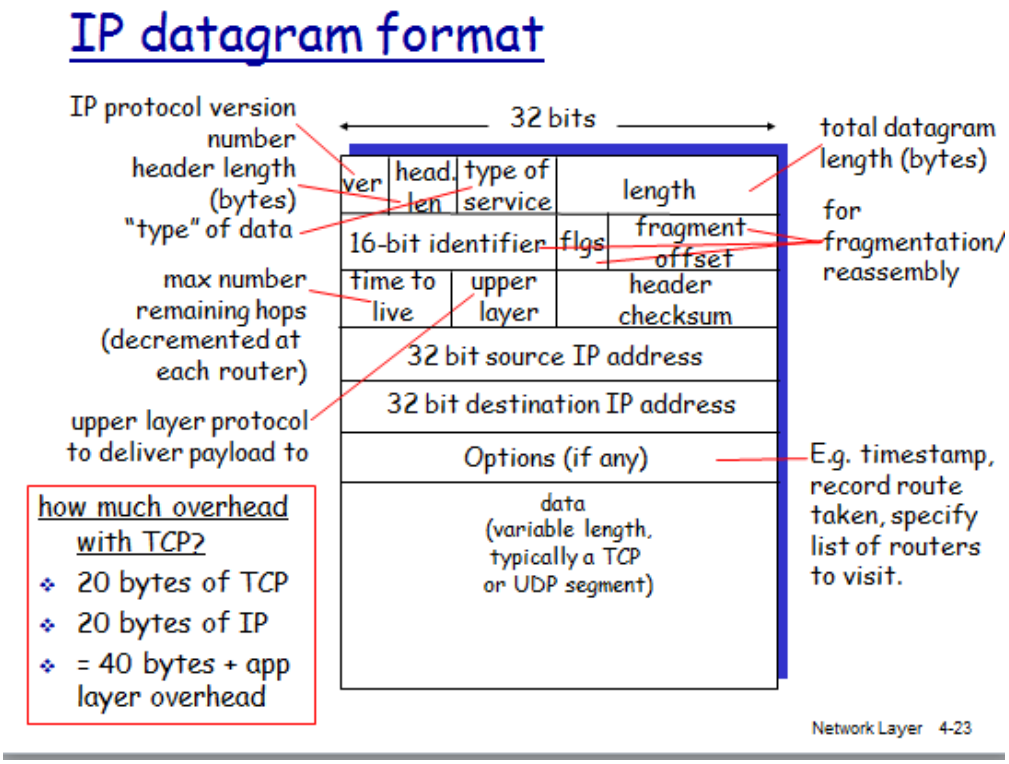
TCP



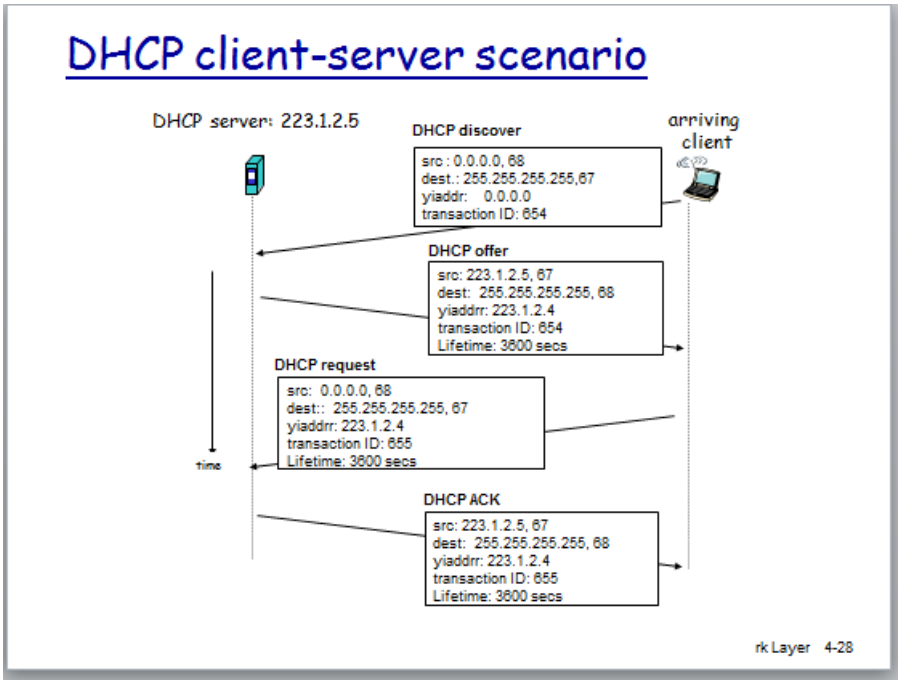
TCP – Congestion Control



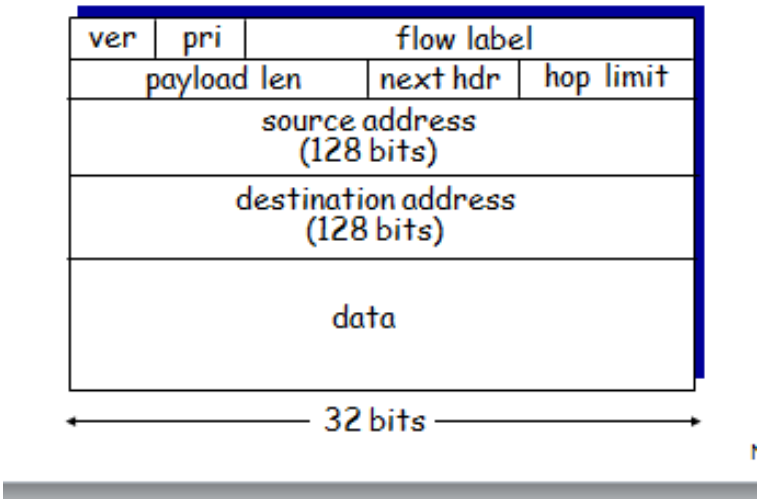
IP


IP datagram format
IP protocol version number
header length (bytes)
"type" of data
max number remaining hops (decremented at each router)
upper layer protocol to deliver payload to
32 bits
ver
head. len
type of service
length
16-bit identifier
flgs
fragment offset
time to live
upper layer
header checksum
32 bit source IP address
32 bit destination IP address
Options (if any)
data (variable length, typically a TCP or UDP segment)
total datagram length (bytes)
for fragmentation/ reassembly
E.g. timestamp, record route taken, specify list of routers to visit.
how much overhead with TCP?
20 bytes of TCP
20 bytes of IP
= 40 bytes + app layer overhead
Network Layer 4-23

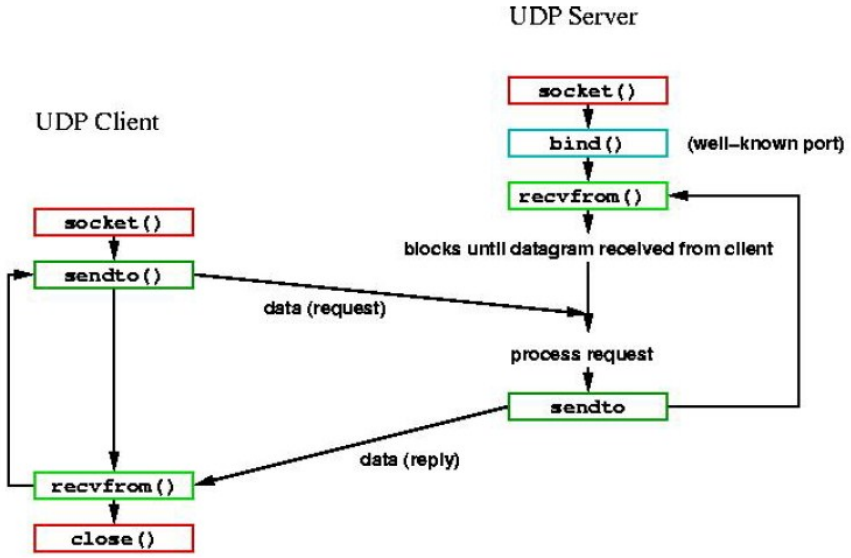
DHCP


DHCP client-server scenario
DHCP server: 223.1.2.5
arriving client
DHCP discover
src : 0.0.0.0, 68
dest.: 255.255.255.255,67
yiaddr: 0.0.0.0
transaction ID: 654
DHCP offer
src: 223.1.2.5, 67
dest: 255.255.255.255, 68
yiaddrr: 223.1.2.4
transaction ID: 654
Lifetime: 3600 secs
DHCP request
src: 0.0.0.0, 68
dest:: 255.255.255.255, 67
yiaddrr: 223.1.2.4
transaction ID: 655
Lifetime: 3600 secs
time
DHCP ACK
src: 223.1.2.5, 67
dest: 255.255.255.255, 68
yiaddrr: 223.1.2.4
transaction ID: 655
Lifetime: 3600 secs
rk Layer 4-28

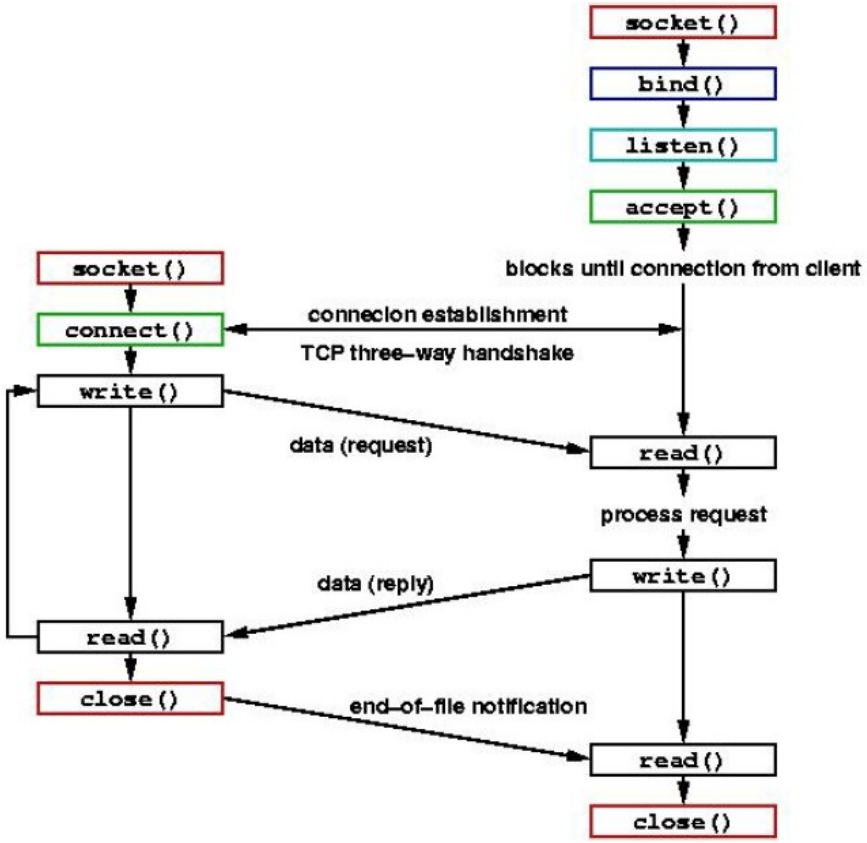
IP v6



Socket UDP



Socket TCP


socket()
bind()
listen()
accept()
blocks until connection from client
socket()
connect()
connecion establishment
TCP three-way handshake
write()
data (request)
read()
process request
write()
data (reply)
read()
close()
end-of-file notification
read()
close()