קורס רשתות תקשורת וסייבר

תוכן

שיעור 1

הסבר על המושג סייבר

1. סייבר זה לא מה שמדמיינים כהתעסקות של האקרים לביצוע פשעים וכו', זה אחד הביטויים לשימוש בסייבר. סייבר זה למעשה הסביבה הדיגיטלית בה אנו מתקשרים ומעבירים ונחשפים למידע שמגיע אלינו ממקורות שונים. למשל כשרופא מגיע לחולה וכל המדדים שלו מוצגים לו במחשב, והוא יכול לגשת לראות בדיקות שביצע, או רקע רפואי וכו', אפילו שליחת מייל פשוט או גלישה באינטרנט – מאחורי זה יש סביבה ומכלול תהליכים הקשורים ליכולת להעביר מידע ולתקשר בין אנשים, מקומות וכו', וזה בעצם סייבר.

האמצעי הכי נפוץ לקישוריות סייבר זה אינטרנט, אבל יש עוד אמצעים להעברת תקשורת, למשל – בלוטוס, או תשלום ע"י אשראי זה גם חלק מעולם הסייבר.

2. רשת תקשורת מחשבים – מושג המתאר תקשורת בין מחשבים בודדים, שכל אחד מהם stand alone. מטרת הרשת תקשורת מחשבים היא א. העברת נתונים בין מחשבים, למשל שליחת מייל, ב. שיתוף משאבים, למשל לקשר את כל המחשבים למדפסת המשרדית.

הדבר הנפוץ ביותר לשיתוף הוא האחסון, כלומר לשתף מידע הנמצא באחסון במקום מסוים – שעוד אנשים יהיו חשופים אליו.

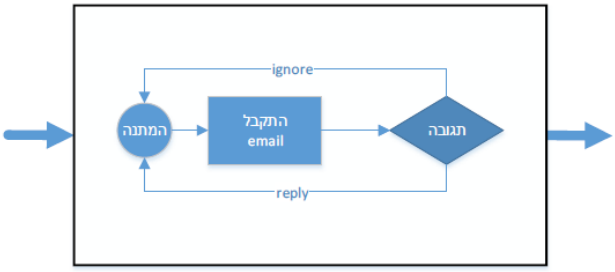
3. סוגי רשתות תקשורת

א. רשת תקשורת מקומית (local area network) LAN – רשת שפרוסה באותו מתחם/ מבנה, ולכולם יש יכולת לגשת לקבצים מסויימים שנמצאים בתיקיית רשת וכד'. דוגמא נוספת שאנו מכירים זה הראוטר שיש לכולנו בבית (בהערת אגב נציין שלקופסה הזו קוראים ראוטר אבל ראוטר זה חלק מסוים וחשוב בקופסה הזו, אבל יש בה עוד המון רכיבים וחלקים שבאופן מתכלל קוראים להכל ראוטר). בין כל המרכיבים שבראוטר יש רכיב שנקרא רכזת, שפורס את הרשת המקומית בבית לצורך שיתוף במשאב מסוים – האינטרנט.

ב. רשת תקשורת מרחבית (wide area network) WAN - אנו מבינים כי לרוב LAN של בניין מסוים באוניברסיטה או של ראוטר בבית שלי, שונה מהLAN של בניין ליד או מהראוטר של השכן שלי. עם זאת – ניתן לחבר מספר LANים יחד ולקבל רשת תקשורת מרחבית. האמצעי לקשר את הרשתות הללו יכול להיות קווי – ע"י חיבור של כבל פיזי (חשמלי/ אופטי) או ע"י חיבור אלחוטי – ע"י גלי רדיו (למשל ויי פיי).

4. מקור השם סייבר הגיע מיוונית מהמילה cybernetic (/קיברנטיקה, מאותו מקור של המילה קברניט – השולט ומנהיג את הספינה/מטוס), מושג שלא התחיל מעולם המחשבים, המתאר גישה/מתודה שבאמצעותה ניתן לשלוט במערכת. כדי שמערכת תהיה מערכת סייבר חייבת להתקיים בה פעולה של משוב, כלומר פעולה המשפיעה על המערכת שכתוצאה ממנה המערכת מייצרת תגובה (לדוגמא עובדים הבאים לקברניט והוא מחליט מה צרך לעשות).

ניתן דוגמא למערכת קיברנטית – מייל. בהתחלה אני במצב המתנה, לא התקבלו שם פידבקים. ואז אני מקבל איזשהו קלט חיצוני, ואז המערכת עוברת למצב של תגובה, וניתן להשיב למייל, ליצור שינוי במערכת ולהגיב, או להתעלם ממנו



ולחזור למצב המתנה. נורברט ב1948 הגדיר את המונח סייבר כך:

" The scientific study of control and communication in the animal and the machine"

הגדרה יותר מאוחרת מ2007 מגדירה את הסייבר כך:

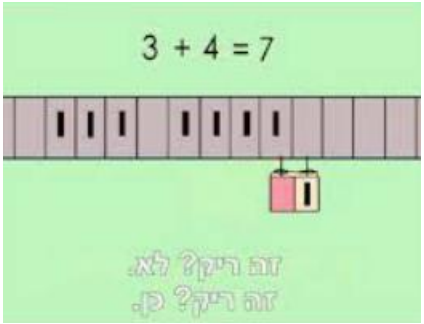
"Cybernetics is the study of systems and processes that interact with themselves and produce themselves from themselves"

כלומר מערכת עם אינטראקציה עצמית (משוב) וע"י פעולות שהיא עושה היא יכולה להשתנות.

<u>סייבר כיום</u>

5. לפי ההגדרה של סייבר, המערכות שליטה הראשונות שהיו ראויות להיקרא מערכות סייבר היו פשוטות ולא משמעותיות בכלל לעומת מה שאנו מכירים כיום, אך החידוש בהן היה בעובדה שלא נדרשה התערבות אנושית מבחוץ. בהמשך השנים הלך והתפתח מושג הAI – artificial intelligence, בינה מלאכותית, שעם התפתחותו הצליח לבצע הרבה דברים שלפני כן לא היה ניתן לחשוב שמחשב יהיה מסוגל לעשות , כגון משחק שחמט, או תרגום שפות – שנדרש "להבין" לפי הקשר ולא בהכרח כל תרגום מילולי הוא נכון.
6. כאמור בהגדרת קיברנטיקה מתבצע משוב, בהגדרה פורמלית למערכת משוב יש 3 מאפיינים – קלט, פלט, ודגימה של הסביבה: output (t) = f (input(t) , sampling (t'<t))
t זה דגימה של הרגע הנוכחי, t' זה דגימה או יותר נכון מה שהמערכת למדה בעקבות כל פרקי הזמן הקטנים מהרגע הנוכחי, ומה הובילה כל החלטה. הוא מסתמך על כל מה שהוא ראה כדי לחזות מה הדבר הנכון לעשות בעתיד – במהלך הבא.
אז המשוואה הזו מתארת לי תהליך של פונקציה/מודל מסוים, המקבלת קלט – למשל רואה מה קורה – בכל זמן t נתון, ועושה לזה דגימה ביחס לכל מה שקדם לזה – ומה ש"יוצא" מהפונקציה זה הפלט שתחזיר לאותו רגע נתון.

<u>מכונת מצבים סופית/אינסופית</u>



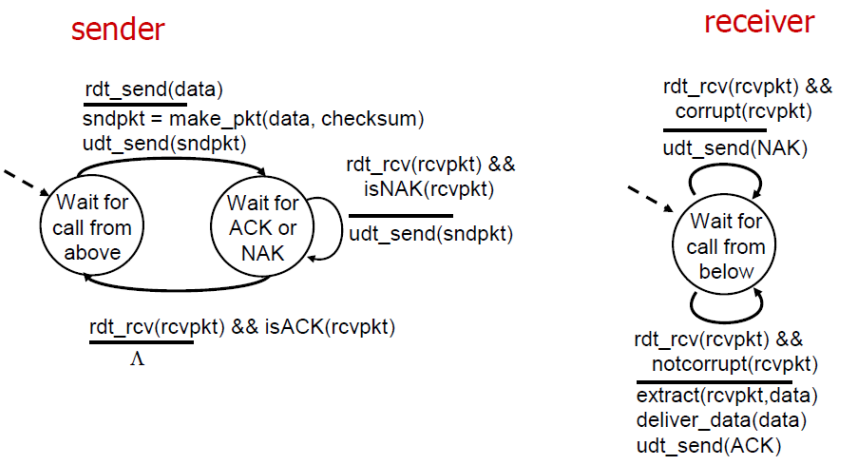
7. אלן טיורינג נחשב לאבי אבות המחשבים, הוא המציא מכונה דמיונית שיש בה סרט אינסופי שעליה יש 'ראש קורא' והמכונה יודעת לבצע פעולות חישוב/העתקה ו"להחליט" האם לכתוב משהו בתא על הסרט או למחוק או לזוז ימינה/שמאלה – לא ניכנס לזה כרגע אבל הקונספט שלו הביא להמצאת המחשבים הראשונים.
8. המחשב שהתפתח אז ומוכר לנו היום הוא מסוג FSM כלומר Final State Machine – מכונה בעל מספר סופי של מצבים. דוגמא למכונה בעלת מספר סופי של מצבים זה רולטה, בעוד שלמשל אם אני אעמוד במרחק 2 מטר מקיר ואזרוק כדור על הקיר וארשום את כל הנקודות שהוא נחת בהם על הרצפה – הנקודות יהיו סביר להניח במרחק של בין 0 ל2 מטר ממני (בהנחה שלא זרקתי חזק) זו לא "מכונת" מצבים סופית, כי יש אינסוף נקודות על הרצפה ביני ובין הקיר גם אם זה רק 2 מטר, זה שיש גבולות גזרה לא אומר שזה סופי. מתג תאורה – הוא מכונת מצבים סופית – יש לו on ו off. מחשב כאמור הוא מכונת מצבים סופית, גם דברים שנראים לנו רציפים ואינסופיים הם בעצם חלוקה של המחשב להמון המון יחידות קטנות שלנו כבני אדם אין יכולת לקלוט אך זה מה שקורה "מאוחרי הקלעים". למשל כשאנחנו רואים קטע וידאו – לנו זה נראה רציף אבל למעשה

אלו אלפי תמונות שמוצגות במהירות מאוד גבוהה (נמדד בfps – frames per second) והמוח משלים את כל התמונות לתנועה רציפה. באותה מידה גם התמונה עצמה נראית "רגיל" אבל למעשה כל תמונה מורכבת מאלפי פיקסלים של שילוב אדום, ירוק, כחול (RGB), כאמור הם אמנם באלפים – אבל זו עדיין כמות סופית.

<u>תקשורת בין משדר למקלט</u>

9. נסביר כעת את אופן הפעולה של תקשורת בין 2 מכונות מצבים סופיות – משדר ומקלט. אך לפני כן הקדמה – לכל אחד מאיתנו יש בתעודת זהות ספרה אחרונה המשמשת ספרת ביקורת. הספרה הזו מחושבת לפי פונקציה מסויימת על כל המספרים בתעודת זהות, כך שהתוצאה היא ספרת הביקורת. הצורך של ספרת הביקורת היא לפעמים שלאור טעות הקלדה הוחלפה ספרה אחת למשל, כאשר נקבל תעודת זהות נוכל פשוט להשתמש בפונקציה לחישוב ספרת ביקורת, ואם התוצאה יוצאת זהה לספרת ביקורת שהוקלדה – סימן שהמספר ככל הנראה (90%) נכון.

הסברנו את זה כדי להבהיר שיש יכולת לבדוק האם קלט שהתקבל הוא תקין או שהיו בו שגיאות מצד המשתמש/ השחתות לאור העברת המידע ברשת. הפעולה הזו של הבדיקה נקראת checksum. נסתכל על פעולת המשדר – בהתחלה יש לו מצב המתנה "wait for call from above", לאחר מכן אם מתרחש העברת נתונים אמינה - reliable - rdt data transfer כלומר מייצא דאטה – אז הוא מכין מנה של שידור שמורכבת מהדאטה עצמה



ומפונקציית הchecksum לבדיקת נכונות הדאטה. לפעולה הזו הוא קורא send pocket ושולח אותה ברשת ועובר למצב של המתנה – לדעת מהמקלט האם הוא קיבל את הקלט.

למקלט יש מצב אחד – מחכה לקריאה מלמעלה – מהמשדר, אם הוא קיבל מנה והיא לא הושחתה (ע"י שימוש בchecksum) אז הוא שולף את הדאטה ומעביר אותה למשתמש או בדרך שבה הוא עובד (למשל ברמקול וכד') ובסוף שולח חזרה אות למשדר שהאות הגיע בצורה תקינה – ACK שזה למעשה קיצור של acknowledge – אישור. בינתיים המשדר נמצא במצב של המתנה – wait for ack or nak – אם הוא מקבל משוב מסוג ack אז הוא חוזר למצב המתנה לשלוח את המנה הבאה. אופציה אחרת זה שהמקלט בדק את הדאטה מול הchecksum וראה שיש אי התאמה – לכן הוא מבצע פעולה של corrupt כלומר לא משתמש במנה ולא מחזיר למשתמש (או לרמקול לפי איך שהוא עובד) כלום, ורק מחזיר למשדר אות של nak – כלומר not acknowledge. מה עושה לו המשדר שקיבל nak? שולח את השדר מחדש- כי יכול להיות שמשהו בהעברת המידע בדרך השתבש.

הערה חשובה – אנו בכל נושא רשתות התקשורת יוצאים מנקודת הנחה שאין לנו בעיות בעיבוד, כלומר שהמערכת לא טעתה בחישוב למשל, והתקלות היחידות שיש הן הפרעות בתקשורת ובהעברת המידע. ניתן להציג זאת כך - אני יודע מה אני רוצה לומר, ולצד השני יש יכולת להבין,

אם הצד השני לא הבין אותי כנראה היו הפרעות – כי דיברתי חלש / כי היה רעש חיצוני / כי דיברתי מהר, אבל המלל עצמו לא שגוי מבחינת היותו מלל אמיתי והגיוני (אצל חלקנו 😊)

10. ההגדרה המדויקת כיום של סייבר לפי מילון אוקספורד - הסביבה הרעיונית שבה מתרחשת תקשורת דרך רשתות מחשבים:

" The notional environment in which communication over computer networks occurs "

שיעור 2

יסודות המחשב

1. המחשב עצמו הוא אולי לא סייבר בעצמו אבל הוא המרכיב המהותי ביותר עליו בנוי עולם הסייבר כפי שאנו מכירים אותו היום – סייבר – שליטה – כחיבור של מחשבים.
נתחיל בשאלה – האם מחשבון הוא מחשב? הוא מבצע פעולות חישוביות, מקבל קלט מבחוץ, מסוגל לבצע דברים שבן אדם לא יכול לעשות במאית שניה וכד'.
ההגדרה המילונית של מחשב:

"A programmable electronic device, designed to accept data, perform prescribed mathematical and logical operations at high speed, and display the results of these operations"

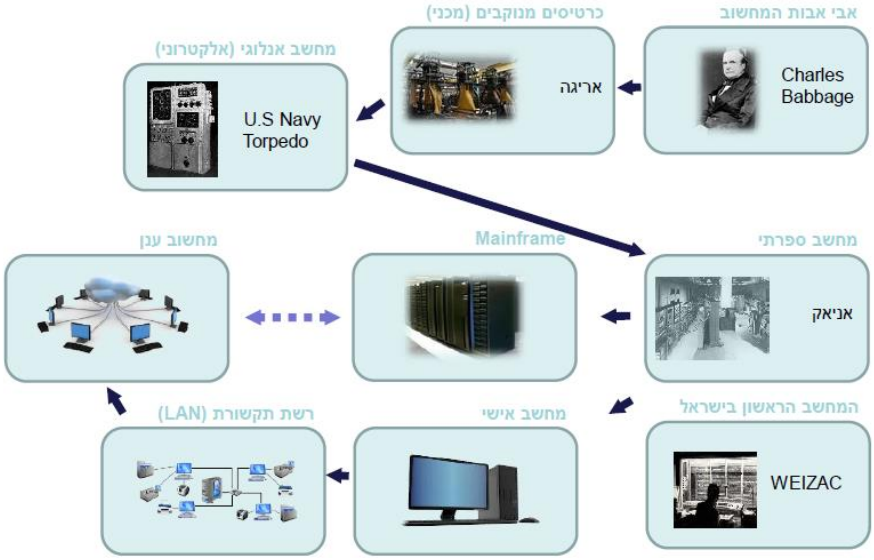
ההגדרות של מחשב אם כן הם:

- מכשיר הניתן לתכנות
- נועד לקבל נתונים
- מסוגל לבצע פעולות מתמטיות והגיוניות שנקבעו לו מראש
- מסוגל לבצע את הפעולות במהירות גבוהה מאוד
- מסוגל להציג את התוצאות והפלט של הפעולות שעשה.

המחשבון אמנם מכיל קודים בתוכו אבל לנו אין יכולת להתערב בחוץ. אם כן נשאל- האם הפלאפון שלנו הוא מחשב? כן, אמנם אנחנו לא יושבים וכותבים לו קוד אבל כאשר אנחנו מורידים אפליקציות למשל אנחנו למעשה מכניסים לו קוד חדש שאנו מעוניינים שהוא יבצע.
תנאים הכרחיים להגדרת מכונה כמחשב:

- מגיבה באופן מוגדר היטב למערכת פקודות
- ביכולתה לבצע תוכנית באופן עצמאי

2. <u>ההיסטוריה של המחשב:</u>



3. בשונה מהמחשב האנלוגי שכבר היה אלקטרוני, אבל עבד בצורה אנלוגית – כלומר רציפה מבחינת זרם החשמל שלה – דבר שהיה לא יעיל בכלל, המחשב הראשון שדומה למחשב שלנו נקרא אניאק. השינוי במחשב זה הוא המעבר מצורה אנלוגית ורציפה לצורה ספרתית – כלומר מערכת המייצגת את המידע שלה ע"י מספרים, ופעולות רבות ומהירות שלאור מהירותן הן נראות כרציפות (כמו הדוגמה עם המסך או הפיקסלים משיעור קודם). אחד המחשבים הראשונים בעולם, והראשון בישראל, פותח במכון ויצמן ב1955 בעלות של רבע מיליון דולר, היה באורך של כמה מטרים ושקל כ17 טון. מיותר לציין שכוח החישוב של המחשב הזה הוא אולי מאית ממה שהטלפון שלנו היום מסוגל לבצע.

לאחר מכן התפתחו 2 ערוצים של שימוש במחשב הספרתי – mainframe – ששימש בעיקר גופים גדולים שרצו שמשתמשים רבים יוכלו לבצע דרכו חישובים ולהתחבר אליו ולמידע שבו, והמחשב האישי – שהיה בעצם מעין הmainframe אלא שבתצורה קטנה יותר, עם כוח חישובי או יכולות נמוכות יותר, אך הוא יועד לשימוש האישי והיה ניתן להתחבר אליו באופן יחידי, כלומר רק אדם אחד יכל להשתמש בו ולעבוד עליו בזמן נתון.

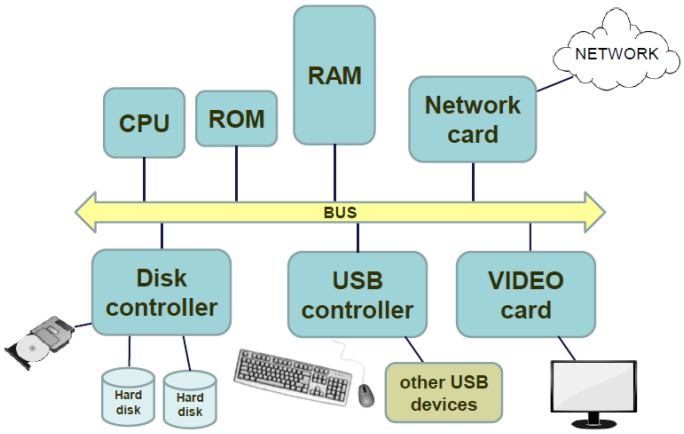
לאחר מכן באה ההתפתחות המבוקשת שעליה נדבר הרבה בקורס – רשת התקשורת. הדבר עלה בעקבות הצורך לשלוט במידע בזמן אמת כאשר יש לנו ריבוי משתמשים. למשל אם למשל במחסן של חנות מקבלים סחורה כרגע ומעדכנים את זה במחשב – לחנות עצמה אין יכולת לדעת מה יש או אין במלאי, אלא היו צריכים ללכת למחשב עצמו ולראות מה שהוזן בו. בעקבות צורך זה נולד מושג הLAN (שלמדנו בשיעור קודם) והיכולת של מספר משתמשים להתחבר לרשת המחשבים.

המושג האחרון שנולד מזה, ומוכר לנו היום הרבה יותר הוא הענן. ברשת התקשורת יש לנו משאבים, כגון הדיסק המרכזי שמאחסן את הנתונים, אבל לצורך זה נדרש לחבר גם את מקור האחסון אל הרשת באופן קווי- פיזי. הרעיון של הענן הוא שבזכות תשתית האינטרנט הרחבה שיש לנו כיום אנחנו לא חייבים שמאגר הנתונים יהיה מחובר פיזית אלא ובמקום לשים אותו במקום מסוים ברשת ולחבר אותו לרשת המחשבים, נשים אותו במקום מרוחק – שנקרא לו ענן, שאני לא ידוע איפה הוא נמצא, אבל יש לי אפשרות לגשת אליו, בעזרת האינטרנט. במובן מסוים הענן הינו

גרסה משופרת ומתקדמת יותר של ה mainframe, מאחר והוא בעל דמיון ל mainframe, כאחסון מרכזי אחד, ולא כל מחשב פרטי עם האחסון שלו, שניתן לגשת אליו ממספר מקומות, אלא שזו כאמור גרסה יותר טכנולוגית ומתקדמת. צריך להבין שכל חיבור הוא חיבור פיזי – ע"י כבל/ גלי רדיו/ ועוד, אלא שהמהפכה של האינטרנט היא בעובדה שלא כל גוף צריך להתקין את התשתית מההתחלה אלא להתחבר לרשת הכלל עולמית.

<u>מבנה המחשב</u>

4. הארכיטקטורה של המחשב:



- CPU – ראשי תיבות של central processing unit, יחידת עיבוד מרכזית – מוכר לנו יותר בשם מעבד, מעין המוח של המחשב, ובו מתבצעת הפעולות של המחשב. באופן פשוט ניתן להסביר ששפת התוכנה נטענת לתוך הcpu, והוא יודע לפענח את הקוד. עוצמת המחשב נובעת באופן ישיר מעוצמת הcpu – שיכול להכיל גם מספר cpu בעוצמה שונה, ובעל כמות ליבות שונה. החלוקה למספר ליבות זה בעצם היכולת לבצע כמה דברים במקביל. (דגש חשוב – המעבד לא יודע לקרוא קודים בשפות התכונה השונות, לתוכנה, יש קומפיילר שהוא זה שממיר את הקוד שאנו הכנסנו לשפת מכונה – 1 או 0, כלומר הזרמת חשמל או עצירה מהזרמת חשמל – ואת זה המחשב יודע לקרוא ולתרגם).
הcpu עובד בפעימות, כמו לב, ולא בצורה רציפה. הוא מכיל שעון חיצוני שמשדר לו "פולסים", גל ריבועי, שאומר לו מתי לבצע עוד פעולה. מתאפיין/ נמדד בין השאר במהירות השעון (למשל GHz2.80). אם כן אז למה שלא נגביר לו את מהירות השעון – שניתן לו פקודות לבצע במרווחים יותר קצרים? התשובה היא שcpu עובד מאוד קשה וצורך הרבה אנרגיה, העלאה של הדרישה ממנו תגרום להתחממות שלו, זו הסיבה שלכל מעבד מוצמד מאוורר. תיאורטית אני יכול להאיץ את המעבד יותר – אבל אני עלול לשרוף אותו.
- RAM – ראשי תיבות של random access memory, הוא למעשה הזיכרון או התשתית של הcpu באמצעותו הוא מבצע את כל הפעולות החישוב. כלומר כאשר אנחנו אומרים שאנו נותנים פקודה (טוענים קוד) אל הcpu הקוד הזה למעשה נשמר בram, ואז הcpu שולף ממנו שורה שורה ומבצע אותה, ושומר בו דברים שהוא נדרש "לזכור" להבא לצורך ביצוע פעולות שנדרש לבצע. ברגע שהמחשב סיים את הפעולה שקיבל וביצע ואין לו סיבה יותר לשמור את המידע ששמר בram - הוא "זורק" אותו. (במקרה של הפסקת חשמל – הזיכרון שהיה בram ייעלם גם, לכן קורה שאנו עובדים על קובץ וכשיש הפסקת חשמל

לא ניתן לשחזר את הקובץ שעבדתי עליו – כי נשמר בram כל זמן העבודה שלי כשלא שמרתי אותו בזיכרון המחשב עצמו).
אפשר להקביל את ההבדל בין הזיכרון ram לזיכרון האחסון של המחשב (דיסק קשיח) כהבדל בין ספרים שיש לי בבית וקראתי פעם ואני משתמש מידי פעם, לבין ספרים שאני קורא כל הזמן או שאני בדיוק קורא אותם עכשיו – שאת הספרים הכלליים אניח בספריה ואת אלו שאני משתמש כרגע או בדרך כלל – אניח על השולחן לידי כך שאוכל בקלות להגיע אליהם.
הסיבה שהוא נקרא random access הוא שלא משנה לו איפה בדיוק הוא שומר זיכרון מסוים אלא פשוט במקום אקראי פנוי, ולא בהכרח במקום ספציפי ומסודר ביחס לזיכרון הנוסף שמחזיק, שיש לו בזיכרון שיהיה לו, כך שיהיה לו קל לגשת אליו במהירות. ברגע שנשמור את הקובץ (מה שאנו מכירים – לשמור בתיקייה/ על שולחן העבודה) אנו מעבירים את המידע מהram לזיכרון הקשיח).

- ROM – ראשי תיבות read-only memory –(זיכרון לקריאה בלבד) בשונה מהram שנמחק כאשר לא משתמשים בו או כשהמחשב מכובה למשל, הrom הוא זיכרון שמובנה במחשב, שהוא קבוע במחשב ולא ניתן לשנות אותו אבל הוא נשאר זמין גם אחרי כיבוי המחשב, הוא כולל קטע קוד מאוד בסיסי כדי לאפשר למחשב לחזור חזרה לעבודה כאשר נפעיל את המחשב. מעין starter של המערכת.
- Hard Disk – (או במחשבים ניידים - SSD) אחסון קשיח, הזיכרון של המחשב שאני מעוניין לשמור, כמו תוכנות, קבצים, תמונות וכו'. אם נחזור להקבלה עם הספרים כשהסברנו על הram, הדיסק קשיח הוא מקור אחסון שהגישה אליו היא לא ישירה מהcpu – לכן הזמן שייקח לcpu להשתמש במידע שבתוכו הוא איטי יותר. בנוסף זיכרון קשיח זול יותר (ולכן למשל בזיכרון הקשיח יש לרוב 512 ג'יגה או 1 טרה בייט וכד' אבל בRAM לעומת זאת 8/16 ג'יגה בייט).
לזיכרון הקשיח יש חומר בעל "זיכרון מגנטי" כך שניתן לשחזר את המידע ש"נטבע" עליו, ללא צורך בחשמל. (הגישה למידע כן דורשת חשמל, היא מתבצעת באמצעות ראש קורא, ובעזרתו יהיה ניתן לקרוא את הזיכרון המגנטי שעליו).
במחשבים ניידים הזיכרון של המחשב הוא SSD– ראשי תיבות של Solid-State Drive שבגלל שמחשב נייד בנוי באופן כללי בעוצמת רכיבים נמוכה יותר, נדרש שהזיכרון יהיה חיצוני, כמו הhard disk אבל קרוב יותר כדי שהגישה אליו תהיה מהירה יותר.
זו הסיבה שבמחשבים נייחים אנחנו יכולים לשמור את הקבצים על כונן C – שהוא הSSD, שעליו נשמור את המערכת הפעלה, ואת התוכנות הבסיסיות כמו קבצי office וכד' – וכך הגישה אליהם תהיה מהירה יותר מאם היו על הדיסק הקשיח, וכונן D – שהוא הזיכרון הקשיח, שהוא לרוב יהיה גם יותר גדול מבחינת כמות האחסון שניתן לשמור בו.

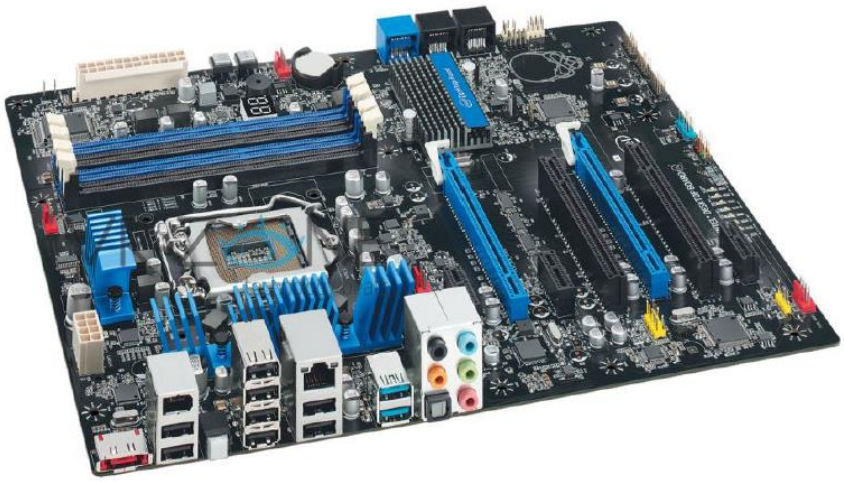
<u>רכיבי i/o:</u>

- USB – ראשי תיבות של universal serial bus (בעברית: אפיק טורי אוניברסלי) אמצעי להוצאת נתונים מחוץ למחשב.
- Video card – מוכר לנו בשם כרטיס מסך, מאפשר להוציא פלט באופן ויזואלי על הצג. (ככל שהכרטיס מסך יותר מתקדם כך הרזולוציה או מגוון הצבעים שהוא יהיה מסוגל

לבטא יהיה גדול יותר והתצוגה תהיה מהירה וחלקה יותר). במחשבי גיימרים שנדרשת רמת תצוגה גבוהה ומהירה מאוד יש כרטיס גרפי, עם מעבד עצמי (ומאוורר כמובן).

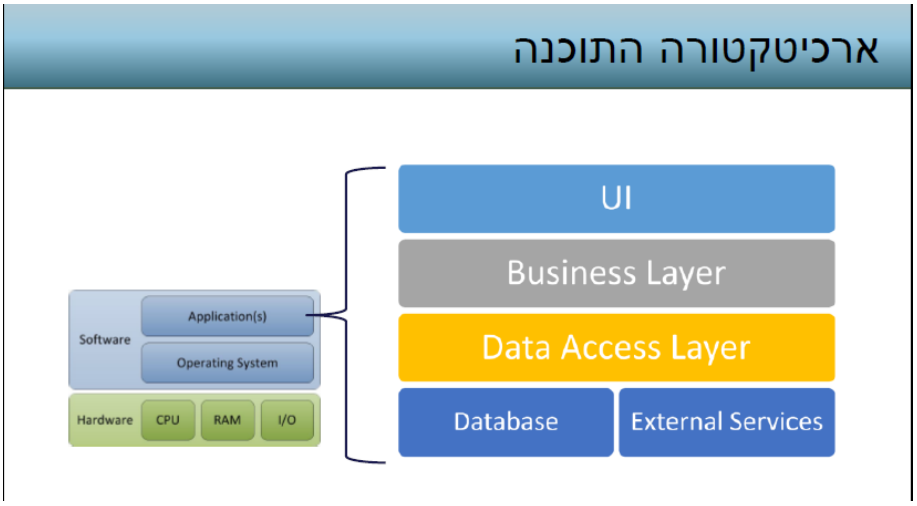
- Network card – כרטיס רשת, המשמש לחיבור לרשתות שדיברנו עליהם קודם, שניתן להתחלק ל2 סוגים -קווי, בחיבור פיזי, ואלחוטי – המאפשר לי להתחבר לוויי פיי ולאינטרנט חיצוני.
- Bus – התשתית המאפשרת את העברת המידע (הביטים – 0/1) של המחשב בין החלקים השונים שלו. נמדד בביטים למשל 32 ביט או 64 ביט – זהו למעשה תיאור של כמות חוטי הנחושת המחברים כל חלק ומבטאים את כמות הביטים שהוא מסוגל להעביר ביחידת זמן/העברה אחת – 32 או 64 ביטים. הכוח של הusb הוא שניתן לחבר מספר חיבורים והוא יודע להתמודד איתם ולהריץ אותם בbus משותף.

5. לוח האם :



<u>ארכיטקטורה כללית של מחשב</u>

6. נעשה zoom-out – כל מחשב/ מכשיר אחר מורכב מחומרה (hardware) ותוכנה (software) החומרה זה הדברים הממשיים – כמו המעבד, הram, הזיכרון קשיח – כלומר רכיבים פיזיים של המחשב, והתוכנה זה הקוד/ מערכות ההפעלה שמובנות על החומרה הנ"ל.



- User interface – ממשק המשתמש, האופן בו התוכנה תוצג למשתמש.
- Business layer – מעין ה"לוגיקה" של המחשב, כלומר מה צורך העבודה שלו, למשל אם נדרש למיין נתונים בטבלה- באיזה סוג מיון להשתמש, או כל תא/ מקש לאן מפנה וכו'.
- Data access layer – מאפשרת גישה לבסיס הנתונים, למשל שליפת שם של אדם לפי תעודת הזהות שלו.
- Database – בסיס הנתונים, המנהל את אגירת המידע, הטבלאות וכו'.
- External services – לא נתמקד בזה כרגע.

שיעור 3

רשת האינטרנט

1. רשת האינטרנט היא בעצם כמות עצומה של 'מכונות' ממוחשבות שכולן מקושרות ביניהן, באופן תיאורטי יכולה להיווצר תקשורת בין כל מכונה לכל מכונה. המכונות השימושיות ביותר לתקשורת זו אלו כמובן המחשב. מספר מושגים הקשורים לאינטרנט ולמבנהו:

א. Host – המונח host בתרגומו עלול להיות מטעה – מארח – בפועל המשמעות שלו זה יחידת קצה (end system), כלומר המכשיר שבסופו של דבר מבצע/ משתמש בעצם התקשורת. למשל הטלפונים/ המחשבים שלנו הם יחידות קצה, מעין עלה בקצה של האינטרנט שממנו ניתן לבצע תקשורת. ראוטר/ אנטנה וכו' אינם host, הם צמתים שמתחברים אליהם כחלק מהרשת.

ב. Communication links – האמצעים הפיזיים שמחברים את התחנות השונות ברשת, (נרחיב עליהם בהמשך):

- חוט חשמל
- סיב אופטי (בשונה מהחוט חשמל שמעביר גלי חשמל הסיב מעביר גלי אור),
- שידור אלחוטי - לוויין, ראוטר (גם תקשורת אלחוטית היא פיזית – באמצעות גלי רדיו וכד').

רוחב פס - כמו שאני יכול לשאול ממה עשוי כביש בו נוסעות מכוניות, אני רוצה לדעת על קצב הנסיעה בכביש, או יותר מדויק – כמה מקסימום מכוניות יכולות לעבור בכביש הזה ביחידת זמן – בהקבלה לכך הנתונים יכולים לעבור ברשת באמצעות ה communication links, וה'קצב' מעבר הזה נמדד ע"י המושג bandwidth – רוחב פס. כשרשת מסויימת מתאפיינת ב'פס רחב' זה לא שהאמצעי תקשורת הפיזי עצמו רחב אלא כמה יחידות מידע מסוגל להעביר ביחידת זמן – נמדד כמות הביט לשנייה.

אם הראוטר אצלנו בבית הוא במהירות של 500 מגה בייט למשל – המשמעות היא שהוא מסוגל להעביר 500מגה (מיליון) ביטים בשנייה.

[הערה – אם נמדוד את מהירות האינטרנט שלנו, פעמים רבות נקבל פחות ממה שקנינו, הסיבה לזה היא שהמהירות הזו היא המהירות הממוצעת לאורך כל שעות היום, גם בשעות לילה בהם אין "עומס" או מפריעים רבים והתקשורת יכולה לעבור הרבה יותר מהר. ה500 מגה שקנינו הוא הממוצע של כל הזמנים האלו. נוכל להבין את זה כמו

בדוגמא עם הכביש – בשעות העומס גם אם בכביש מסוים מותר או ניתן לנסוע 90 ו100 קמ"ש – בשעות העומס הגיוני מאוד שהמהירות שרכבים ייסעו בכביש היא נמוכה יותר].

ג. Packet switches – מיתוג מנות, זהו מושג שעוד נסביר בהמשך אך ניתן להסביר בשלב זה שהמידע לא עובר בצורה רציפה אלא עובר ב"מנות" וחלקים.

<u>פרוטוקול</u>

2. פרוטוקול באופן כללי הוא רשימה של כללים שמטרתם להסדיר תהליך. בתקשורת בין בני אדם יש פרוטוקולים, למשל – הסכמה שאם שני בני אדם מדברים הם לא יכולים לדבר ביחד אלא אחד מדבר והשני מקשיב, אם שניהם ידברו לא בטוח שכל המידע יעבור מאחד לשני – תהיה בעיה בתקשורת. כלל חשוב נוסף – שידברו באותה שפה, אם שני אנשים מדברים בשפות שונות הם לא יבינו אחד את השני.
הסיבה שרשת האינטרנט מצליחה לתקשר ולהעביר מידע מנקודת קצה לנקודת קצה היא שקיימים פרוטוקולים מוסכמים מראש המכתיבים לכולם "איך עובדים", למשל כשאני גולש באינטרנט אני מציית לפרוטוקול שנקרא http – (מופיע לנו בתחילתו של כתובת אינטרנט).
3. הפרוטוקולים באינטרנט מגדירים לנו פורמטים, כלומר מבני נתונים, שגם הכלי השולח וגם המקבל צריכים לדעת להבין. למשל אם אני אנסה לפתוח קובץ pdf בתוכנה שלא מכירה את הפורמט (מבנה הנתונים שממנו בנוי הקובץ) של הקובץ, לא אצליח לפתוח אותו באופן תקין.
הפרוטוקול מגדיר גם סדר פעולות מקובלות, כמו שכשמתחילים שיחה אומרים שלום, וכשנפרדים אומרים להתראות – כאשר רוצים להיכנס לאתר, הפורמט מגדיר איך נראית תחילת שיחה/ התקשרות עם אתר מסוים – כמו שמחייגים בטלפון – צריך שהצד השני יקבל את הקריאה שלי להתחיל איתו שיחה, ואם לא ניתן להתקשר אל הצד השני יופיע לנו למשל שכתובת האינטרנט לא חוקית/ לא זמינה.
הפרוטוקול / אופן הפנייה של מחשב אחד לאחר נקרא TCP connection request, ואם המחשב השני לא דוחה את הפנייה שלו הוא משדר לו חזרה TCP connection response.
4. התקנים והפרוטוקולים של האינטרנט הם קבועים ופתוחים – כלומר כל מי שרוצה ללמוד אותם יכול, אבל גורם פרטי לא יכול פשוט לשנות אותם, אלא הם מוסכמים בינלאומית. חבל אפל לדוגמא, בראשיתה, הוציאה את האייפון עם אפשרות טעינה ע"י מטען ייחודי שאך ורק היא יכולה לשווק והתאים רק למכשיריה – כדי לשמור על הבלעדיות במכירת המטענים. האיחוד האירופי קבע שאסור למכור סמארטפונים שהתקן שלהם הוא לא התקן הסטנדרטי של usb הקבוע ואפל נאלצה ליישר קו עם החוק.
מייקרוסופט בראשית דרכה שראתה אתה התפתחות רשת האינטרנט הגדלה והולכת, גם לא רצתה שמתחרים רבים ישתמשו בפלטפורמה שלה, ולכן הקימה רשת אינטרנט מקבילה – msn ראשי תיבות של Microsoft network , וכל גרסאות הwindows שנמכרו היו בעלי פרוטוקולים חסויים ולא אפשרו לאנשים לכתוב תוכנות או לייצר מוצרים על בסיס הקיים – אבל כמו שאנו מכירים הרצון שלהם לא התאפשר וכך קיבלנו את האינטרנט כמו שאנו מכירים היום.
[דגש חשוב, כאשר ישנה ארכיטקטורה פתוחה/ פרוטוקול פתוח – הכוונה שאופן העבודה והכללים והאמצעים פתוחים לכולם, ולא שכל אחד יכול להיכנס למידע עצמו, ניתן להיכנס לאתר מסוים אבל להיתקל בבקשת סיסמה. דוגמא נוספת – במחשבים צבאיים יש אפשרות לעבוד על תוכנות office של מייקרוסופט, אופן העבודה והכללים באינטרנט הצבאי לא שונה ממקבילתה

האזרחית, אבל הרשת הצבאית גם לא מחוברת לרשת הכללית, וגם היא מכילה שכבות הכנה מסוימות שלא מאפשרות למשתמשים מבחוץ לגשת למידע שברשת שלהם).

המדיה הפיזית

5. כמו שלמדנו כבר, ביט היא יחידת המידע האלמנטרית עם 2 מצבים – 1 או 0, הזרמה של חשמל(1) או עצירה מהזרמה (0) שאנו מייצגים גם בתור true או false. כמו שהסברנו כדי שהביטים יעברו חייב להיות משהו פיזיקלי שמאפשר את התקשורת. כאשר אני מדבר מה שקורה בעצם זה שמיתרי הקול שלי מרטיטים את האוויר ומייצרים גלי קול על ידי הפרש לחצים באוויר, האוויר הוא מוחשי – ניתן להרגיש אותו כשיש רוח, ולמרות שהרעידות באוויר שאני מייצר הן לא כאלו שניתן להרגיש את הרוח שלהם, בכל זאת באוזן האנושית קיים קרום עור התוף, כך שהרוח שפוגעת בעור התוף גורמת לו לתנודות שהמוח מפרש כדיבור. אם החדר מאוד רועש – יהיה לי קשה מאוד לשמוע כי מגיעים קולות רבים מידי שלמוח קשה לסווג את הקול בפני עצמו ולפרש אותו.

כאשר אני מדבר עם אדם אחר גלי הקול מגיעים גם לאדם העומד מולי, אך הם בפועל פוגעים גם בקירות וברהיטים שבחדר. כך גם תחנת רדיו – שולחת גלי רדיו להמון כיוונים, ואם יש בסביבתה מכשיר בעל יכולת קליטה הוא מפענח את הגלים לכדי מידע.

6. האמצעים להעברת המידע:

א. כבל חשמלי - החיסרון של הכבל החשמלי, שהוא הכבל הנפוץ ביותר כיום, הוא שהוא מאוד רגיש להפרעות סביבתיות, בעיקר להפרעות אלקטרו מגנטיות, שההפרעות האלו מייצרות "רעש" למידע העובר בכבל, ולפגוע בו עד רמה כזו שהן יכולות לשבש אותו לגמרי. (לכן המציאו 2 סוגי פתרונות:

ב. סיב אופטי - הסיב האופטי עשוי מזכוכית ומעביר אור, ולא חשמל, ולכן גם פחות מושפע מרעשים אופייניים של שדה אלקטרו מגנטי, ובכך מאפשר את העברת המידע במלוא רוחב הפס עם מינימום הפרעות בדרך. החיסרון שלו הוא שהוא יקר יותר.

ג. כבל קואקסיאלי - אנו מכירים אותו בעיקר ככבל של טלוויזיה או ראוטרים, המלופף בחוטי חשמל (זוג שזור) המייצרים סביבו שדה אלקטרו מגנטי – המונע מרעשים מבחוץ לפגוע בו, ובנוסף עטוף גם בשריג מתכתי העוטף את החוטים ובפעמים שיחובר לאדמה ישמש כהארקה ויאפשר העברת מידע, במוצר שדורש קבלת מידע ללא שיבושים, ולא למשל להעברת חשמל לתנור, שאין משמעות לרעשי חוץ כי לא מקבל מידע.

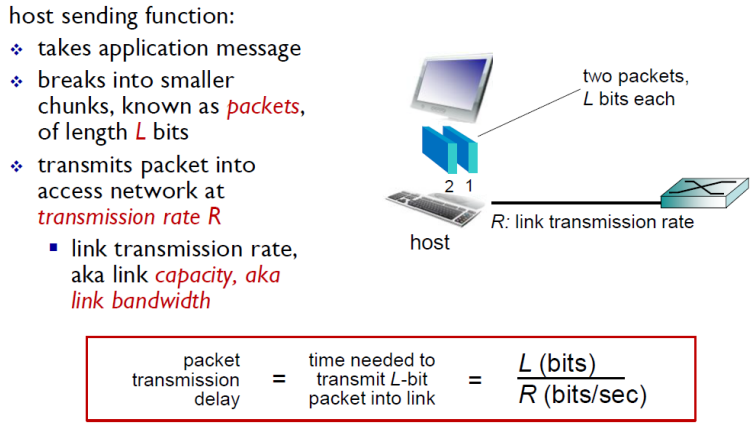
ד. Wifi – טכנולוגיה המוכרת לכולנו מחיי היום יום המאפשרת לפרוש רשת תקשורת מקומית (LAN) בהתבסס על גלי רדיו במקום על חיבור קווי.

ה. לוויין – אחת הסיבות לשימוש בלוויין היא לא רק הצורך להעביר מידע בין מקומות רחוקים אלא נובע מהעובדה שכדור הארץ הוא עגול, וגלי רדיו היוצאים מנקודה אחת, הם יכולים להיות חזקים מאוד כדי להגיע למקומות רחוקות אבל חזקים ככל שיהיו הם לא יוכלו לבצע סיבוב ולהגיע לקצה השני של העולם. היתרון של הלוויין זה גובהו, ובכך הוא מתגבר על הקימור של כדור הארץ ומסוגל לקלוט או לשדר ל2 הנקודות.

(לרוב אנו באופן אישי לא נתחבר ללוויין ישירות חוץ במקרה של שימוש בwaze או gps אחר, אבל גם אז זה לא באופן מלא של תקשורת מחשבים – כי אנו רק קולטים מידע מהלוויין אבל לא משדרים אליו שום דבר).

Packet switching – מיתוג מנות

7. כאשר ישנו שדר מסוים היוצא ממכשיר, הפרוטוקול מחלק את השדר לשברי מקטעים לפי כללי מוגדרים מראש, הוא שולח את המנות אל הרשת באופן כזה שכל מנה נשלחת בנפרד, בדרכים שונות, ולא תלויה בדרך שמנה אחרת עשתה כדי להגיע ליעד. למשל – אם יש לי משלוח מסוים אני יכול לשלוח אותו בשיירת משאיות שיסעו אחת אחרי השנייה עד ליעד, יצאו ויפרקו ביחד אם אחת נעצרת אז כולן נעצרות איתה, אבל בסוף הכל מגיע יחד. אופציה אחרת היא שכל משאית נשלחת לעצמה, וכביכול תבחר את הדרך שלה, אבל היעד הסופי הוא אותו יעד. ההיגיון שכל מנה תבחר את הדרך משלה בכך שזה מאפשר אוטונומיה, כלומר אפשרות לבחור את הדרך האידיאלית בכל מצב, אם 2 משאיות יצאו והכי מהיר היה דרך כביש 2, ומשאית שלישית יצאה אחרי כמה דקות וכבר נהיה שם פקק – ככל הנראה שיש דרך אחרת מהירה יותר שיהיה עדיף לנסוע ממנה.



אחד האתגרים של האינטרנט זה שעל אף היעילות הגדולה יותר באוטונומיה של כל מנה להגיע ליעדה, זה להתמודד עם הבעיה שיכולה לקרות בדרך - שהמשלוחים שיצאו מאוחר יותר יגיעו לפני או שהיו אמורות להגיע קודם לכן. אני לא יכול לקרוא קובץ שהחלק השלישי שלו למשל מופיע בראש הקובץ והראשון שלו, ש"התעכב" לאור רעשי רקע ובעיות בהעברת המידע שלו - מופיע בסוף.

Circuit switching – מיתוג מעגלי

8. המושג שקדם לניתוב מנות הוא ניתוב מעגלי, לא בהכרח מעגל אלא נתיב – שדרכו כל המידע יעבור ברצף אחד אחרי השני. זוהי השיטה הישנה, שהייתה נפוצה בעיקר בתקשורת טלפונית, כך שכאשר אדם רצה להתקשר מישראל לארצות הברית השיחה הייתה עוברת דרך מרכזיות שכל אחת התחברה לאחת אחריה עד שהיה ניתן להעביר את השיחה בחיבור חשמלי ישיר מפה לארצות הברית או לכל יעד אחר.

מיתוג מנות מול מיתוג מעגלי

9. החיסרון שהיה במיתוג מעגלי הוא שאם אירעה תקלה בדרך – למשל אם אחד הקווים נפל – הקשר כולו ינותק, בשונה ממיתוג מנות בו המנה שנתקעה תמצא דרך אחרת להגיע. חיסרון נוסף שיש למיתוג מעגלי הוא שכאשר הוא מוביל תקשורת מנקודה אחת לשנייה – הוא תופס את כל הקווים בדרך במלואם (למשל אם קו יכול להעביר 10 מגה בשנייה, שיחה שצורכת מגה 1 בשנייה תתפוס 10% לכל הקווים בדרך לכל אורך השיחה, בשונה ממיתוג מנות שהמידע לא נשאר בקו וכבר עובר לקו הבא ובכך מאפשר לכמות גדולה הרבה יותר של משתמשים להעביר מידע על אותו הקו.

10. החיסרון שיש במיתוג מנות הוא שרוחב הפס הוא דבר לא קבוע, בניתוב מעגלי אני יודע מראש מה הביצועים של כל מעגל דרכו בחרתי לעבור, בשונה ממיתוג מנות שאין לי יכולת לדעת האם תהיה העברה טובה של המידע ועד כמה. יש פעמים שלא באמת אכפת לנו אם העברת התקשורת מתרחשת מהר או לאט או ליתר דיוק – שלא באמת משנה לי לדעת מראש כמה מהר ונקי המידע יעבור, לדוגמא ווטסאפ/ אסמס/ מייל, אם ההודעה תגיע תוך חצי שניה או תוך 5 שניות או תוך דקה – זה שאני לא יודע מראש מה יהיו הביצועים זה לא מהותי. עם זאת תקשורת לצורכי מדיה למשל סרטון ביוטיוב, אם יש פער של 10 שניות בין התצוגה הוויזואלית לתצוגה הקולית זה פוגע במהות של הפעולה שאני מעוניין לבצע.
מה הפתרון של המדיה? הפתרון שיוסף נתן לפרעה – לאגור אוכל בשנים של השפע כדי שיהיה לשנות הרעב. כלומר "להשתמש" ברוחב הפס כאשר ישנה אפשרות גדולה להעברת מידע ולאגור נתונים להמשך, ואז ניתן להמשיך ולהציג את המידע שנשמר במאגר, ובמקביל לקבל עוד מהמידע, גם אם רוחב הפס קטן יותר. הפעולה הזו נקראת buffering - אחסון זמני, כך מתמודדים עם המצב שהצגנו קודם לכן למשאיות המגיעות מוקדם יותר מאלו שיצאו לפניהן – הם מצטרפות למאגר וממתינות, כדי שכאשר יגיעו המנות המוקדמות הן יוצגו, ויהיה ניתן להציג את המנות המאוחרות יותר ישר בלי צורך לחכות לשדר.

שיעור 4

מעט חזרה על שיעור קודם

1. ראינו בשיעור הקודם את 2 השיטות האופייניות להעברת נתונים – השיטה הישנה, מיתוג מעגלים, בו נוצר לכל תקשורת נתיב קווי ישיר שעליו מבוצעת התקשורת לכל אורך התקשורת, כלומר כל ביט שיעבור – יעבור באותו מסלול בדיוק אחד אחרי השני. השיטה המודרנית יותר זה מיתוג מנות בו המידע מחולק למנות (pockets) וכל מנה היא אוטונומית, מלבד הדרישה היחידה – המקור והיעד, כך שבאותו רוחב פס – התשתית להעברת המידע שעומדת לרשותי – אני יכול להעביר נפח נתונים הרבה יותר גדול. נחזור ונדגיש שבכל הנושא של תקשורת נתונים ה"מאבק" שלי תמיד יהיה על רוחב הפס, ולא על מהירות המעבד או גודל הזיכרון.
עם זאת יש פעמים שבכל זאת נעדיף להשתמש במיתוג מעגלים, וזה כאשר אני רוצה שהשימוש ברוחב הפס יהיה דטרמיניסטי, קבוע וידוע מראש, כך שהוא אמנם לא יהיה יעיל אבל אני יכול לנסות ולמנוע הפרעות (במשל – לסגור את הכביש) ורוחב הפס יהיה מובטח לי. בעוד שבמיתוג מנות אני לעולם לא אצליח "לנבא" את העומס ברגע מסוים ולכן אני לא יודע איך התקשורת תתנהג לאורך העברת המידע. וכמו שהסברנו יש פעמים שזה קריטי לי כאשר יש הפרעות, למשל בשיחת וידאו או בהפעלת סרטון, ודיברנו על אחד הפתרונות – buffering.

הימנעות מאובדן מנות – מידע

2. פתרון אחר זה למשל להקטין את איכות הוידאו בסרטון למשל, וכך עוברים פחות ביטים של הפיקסלים ובכך מפנים את הפס. העובדה שכיום הולכת ומתרחבת העברת מידע בצורה של streaming- הזרמת מידע אונליין - שיחות וידאו, סרטונים, סדרות טלוויזיה וכו' והעומס על מיתוג המנות הולך וגדל ומקשה פעמים רבות על הזרמת המידע אונליין.
עם כל העומס הנ"ל, מערכות התקשורת שואפים לייצר מנגנון מיתוג מנות שיתנהג עד כמה שאפשר כמו מנגנון מיתוג מעגלי circuit like behavior – בכך לנסות להשתמש ביתרונות של כל אחת מהשיטות.

3. נסביר מעט על הbuffering. כאשר מספר אמצעי תקשורת מבצעים תקשורת דרך כלי אחר, למשל מספר מחשבים המחוברים לראוטר (לאו דווקא ראוטר ביתי, אלא כל תחנת העברת מידע, שיכולה להיות ביתית ויכולה להיות אזורית וכו'), העברת המידע נעשית דרכו והמידע לא מאוחסן בו בדרך כלל. אך כאשר ישנו עומס מאוד גדול של נתונים המוזרמים אליו, בעוד שרוחב הפס שהוא יכול לקבל או לשדר קטנים יותר מרוחב הפס הדרוש להעברה חלקה – מידע שהוזרם אליו ולא נשלח עלול להיאבד. ניקח כמשל כוס עם חור קטן בתחתית – אם יוזרמו מים בעוצמה נמוכה – המים לא יגלשו מהכוס החוצה, אך אם עוצמת הזרם תהיה חזקה מאוד המים ימלאו את הכוס יותר מהר ממה שהחור מסוגל להוציא ומים יגלשו החוצה לא כפי שרציתי, ובכך אאבד מים. על מנת למנוע אובדן של מנות התקינו בראוטר זיכרון זמני – buffer – כך שיוכל להוציא את המידע שהתקבל ועדיין לא שודר באופן מבוקר. אבל בכל זאת – אם הbuffer יתמלא גם הוא – יאבדו מנות.
4. מה הפתרון? קל מאוד לומר שכדאי להגדיל את רוחב הפס ע"י חיבור כבל נוסף וכד'. פתרון הבעיה על ידי השקעה של עוד משאבים הוא לפעמים אכן פתרון טוב ונדרש אבל לפעמים זה לא עניין של מה בכך. למשל בנייה של מתקן תקשורת אזורי עם אנטנות חזקות יותר, או העברה של כבל תת ימי באורך מאות קילומטרים בין יבשות, אנו לא נתקין עוד כבל כזה בעלות של מיליוני שקלים כי יש הפרעות בקליטה. פתרון אחר הוא אולי לשדר חזרה דחייה, למשל אם שלחתי יותר מידי קבצים למדפסת, היא יכולה לדחות את השליחה הבאה להדפסה ותשלח את ההודעה שלא ניתן לשלוח להדפסה כעת ויהיה ניתן מאוחר יותר. אבל הדבר מורכב הרבה יותר כאשר מדובר במשתמשים רבים מאוד, ובנוסף השדר המוחזר עצמו של דחיית הבקשה גם הוא דורש נתונים והעברת המסר הזה על אותו פס.
5. חשוב להבין שאובדן של מנות זה לא באג במערכת, אלא הכרח המציאות, ודבר מובנה ברשת השאלה היא מה עושים עם זה והתשובה היא שהפרוטוקול של הרשת – PCP – דואג אוטומטית להשלים את המידע במידה והוא לא התקבל ע"י העברת המנות שוב, כמו שראינו לפני כן כשהסברנו על משדר ומקלט.

<u>זמן תגובה – שיהוי (delay)</u>

6. ברשת האינטרנט יש דיליי, שלא תמיד אנחנו מרגישים אותו בגלל המהירות הגבוהה שלו אבל תמיד קיים דיליי. משך הזמן של הדיליי מורכב ממשך הזמן של 4 חלקים:
   א. Processing, עיבוד - כשמנה נכנסת לראוטר הוא דבר ראשון בודק לאן היא נדרש להגיע.
   ב. Queue, תור – כמו שהסברנו בראוטר יש זיכרון זמני המקבל מידע ומזרים את המידע לפי התור וסדר הגעת המידע. זהו העקרון של תור – FIFO-first in, first out.
   ג. Transmission, הפצה – זהו הדיליי העיקרי, כלומר באיזה קצב ניתן לדחוף את הנתונים אל הפס, גם אם לא יהיה תור – אם רוחב הפס ההוא 100 מגה זאת אומרת שעובר 100 מגה ביטים ביחידת זמן אחת, כך שלא ניתן באותה יחידת זמן להעביר מידע בגודל ג'יגה.
   ד. Propagation, התפשטות – לאחר שמנה עברה דרך הראוטר ונכנסה לקו היא צריכה להגיע ליעד הבא, והדבר זה נטו המרחק שנדרש לעבור שזה בעצם התלות באורך הקו ליעד הבא.

<u>שכבות הפרוטוקול</u>

7. רשת האינטרנט מורכבת ממודל של שכבות, כדי להבין את מודל השכבות ניקח כמשל את האמצעים הנדרשים כדי לטוס לחו"ל.

שלב ראשון הוא קניית כרטיס, שלב שני זה תהליך הקליטה - הגעה לשדה התעופה, צ'ק אין, בידוק ומסירת מזוודות, שלב שלישי זה ללכת אל המטוס עם כרטיס הטיסה ולעלות עליו, שלב רביעי הוא הטיסה עצמה לפי מסלול קבוע, ושלב חמישי זה נחיתה. מפה התהליך קורה בדיוק הפוך – מתבצעת נחיתה, לאחר מכן אני יוצא מהמטוס, לאחר מכן אוסף את המזוודות, מעלה בעיות וכד' אם היו ולבסוף מגיע אל היעד שלי.
ההקבלה הזו היא בעצם החלוקה המהותית לשלבים, הנעשים בסדר כרונולוגי, כך שיש ממשק בין שלב לשלב – ניתן לעבור לשלב הבא כאשר השלב הנוכחי בוצע. דרך נוספת לקרוא לתהליך הזה הוא תהליך מודולרי, כלומר להתייחס לבעיה בחלקים ובכך לפשט ולבזר את הבעיה.

<u>מודל חמשת השכבות</u>

8. מודל השכבות של האינטרנט מורכב למעשה מ5 שכבות:

- Application - יישום
- Transport – תעבורה
- Network – הרשת
- Link – הקשר
- Physical – פיזית

נעבור כעת על כל אחת מהשכבות ונסביר אותה

א. Application – יישום

השכבה העליונה, אמנם היא חלק ממודל הרשת אבל היא עצמה אינה מבצעת תקשורת ברמה הבסיסית שלה. בשכבה זו יש לנו את היישומים שהם הצרכנים של התקשורת, למשל – ווטסאפ, עם כמה שזה מפתיע, זו אפליקציה – יישום שדרכו ניתן להתקשר אל השכבות שמתחתיה שהן מבצעות את התקשורת. האפליקציה זה רק <u>הממשק</u> שמקבלת נתונים ועושה להם המרה למושגים אחרים שאותם ניתן להעביר כמידע. אפליקציה נוספת שכולנו מכירים זה הדפדפן, שכבות אחרות עושות את התקשורת עבורו אבל אפליקציית הדפדפן רק ממירה ממושגי התקשורת למושגי שלנו שנוכל לקלוט ולשגר חזרה פלט לפי הצורך שלנו. הדפדפן למשל, אמון לפרוטוקול שמותאם אליו והוא מוכר לנו מתחילת השורה של כל אתר - http.
יישומים אופייניים לשכבת האפליקציה (application layer):
2 ארכיטקטורות אופייניות לשכבה זו –

- client server – הרוב המכריע של התקשורת באינטרנט מבוססת על ארכיטקטורה של שרת לקוח, דפדפן כרום למשל הוא הלקוח, המשתמש באתר המאוחסן בשרת. אם כן היחסים הם באופן שבו השרת מחזיק את המידע ומעביר אותו והלקוח מציג אותו. גם דיבור בין שני אנשים הוא לא דיבור ישיר אלא מבוסס על אותו מבנה, כי כאשר אני מדבר אני יוצר תנודות באוויר – כך שהאוויר הוא השרת, והאוזן (והמוח) של השומע זה הלקוח שממירים את התנודות למידע שהועבר. הערה על השרת – בהגדרה שלו, מלבד במצב של תקלה, הוא אמור להיות תמיד בעל יכולת התקשרות אליו – always on host.

לכל תחנת קצה (host) יש כתובת רשת (IP address) זוהי כתובת קבועה אליו בלבד. Client לא חייב להיות עם כתובת, אבל כשהוא פונה לserver מסוים הוא חייב שתהיה לו כתובת קבועה שניתן לפנות אליה.
client לעומת הserver, לא חייב להיות מחובר תמיד, כדי לשלוח מייל/ווטסאפ לאדם אחר אני לא צריך שהטלפון/האינטרנט שלו יפעל, התקשורת היא עם השרת של הצד השני, השרת שלי מעביר את המידע לשרת השני לפי הip של היעד, וכאשר הצד השני ידליק את הטלפון (שהוא למעשה משמש כאפליקציה – האמצעי המציג את המידע שהתקבל) אז הוא יראה את מה שהעברתי לו. כל זמן שהclient של הצד השני לא הופעל המידע יחכה בשרת של הצד השני, ואם השרת עצמו ייפול – תתקבל אצלי הודעה חוזרת שלא ניתן לשלוח.
יש מצבים בהם הלקוח עצמו יהיה מעין server, למשל בווטסאפ, שיש הצפנה מקצה לקצה – האפליקציה עצמה לא רק ממירה מידע לתצוגה אלא גם מבצעת פעולה של הצפנה של המידע ושולחת אותו כג'יבריש לשרת, שיעביר לשרת הבא והווטסאפ של מקבל ההודעה יבצע פעולה של ההמרה באופן כזה שמהווה מעין שרת עצמי, או במקרה אחר למשל – כשאנחנו מתחברים לויי פיי כללי בבית קפה וכד' הטלפון מהווה מעין client שהוא גם server ומקבל בעצמו כתובת ip אבל זו לא תהיה כתובת קבועה אלא רגעית עד לסוף השימוש הנוכחי, במקום או רשת אחרת – יקבל ip חדש לצורך תקשורת הנתונים. (הip הזה באותה נקודת זמן תהיה יחידה בלבד ואין באותה נקודת הזמן עוד server עם אותו ip).
לסיכום הנקודה של client server – לקוחות – דורשי המידע – מעולם לא מבצעים את ההתקשרות ביניהם באופן ישיר אלא תמיד באמצעות שרת.

- (peer to peer) P2P – עמית לעמית, זוהי ארכיטקטורה פחות נפוצה, בה כל האמצעים ה"משתתפים" בתקשורת גם מספקים שירותים וגם דורשים שירותים. בארכיטקטורה אין חובה לאמצעים להיות always on host – ביכולת להתחבר אליו. חנויות פיראטיות של שירים לטלפון למשל משתמשות בשיטה זו בפרוטוקול שנקרא torrent, בניגוד לחנויות מקוונות (למשל itunes או google play) בו אם אדם מסוים התחבר לשרת של אדם אחר – הוא יכול להוריד ממנו חלק מהקובץ/מהמידע שהוא מעוניין בו, ויכול להשתמש בו, אבל מעתה גם הוא עצמו נעשה שרת שאחרים יוכלו להתחבר דרכו ולהעתיק את המידע אליהם. מה המוטיבציה שלי אז לשתף את המידע הזה? אם כמה אנשים משתמשים בשרת שלי הם מקטינים לי את רוחב הפס, אלא שבפרוטוקול עצמו מוגדר מעין "עין תחת עין" כך שמי שלא משתף ממידע שנמצא אצלו – הוא מגביל אותו מאוד מבחינת רוחב הפס שלו. נשים לב כי מספר השרתים גדל בהתאם לצריכה והביקוש של מידע מסוים. היתרון של P2P הוא המהירות שניתן להגיע למידע – כי הוא קיים ונגיש בהרבה שרתים ולאו דווקא באחד.

ב. Transport – תעבורה

השכבה האחראית על מיתוג המנות – בה מתבצעת קבלת המידע כקלט מהתוכנה - האפליקציה, <u>חלוקה</u> למנות והעברתה לשכבה שמתחתיה – הרשת, כדי שהיא תעביר את המידע הלאה, או לחילופין קבלת המידע שנשלח אליה מהרשת, בדיקה שכל המנות

הגיעו, סידורן בסדר הנכון והעברתם לאפליקציה, וכן במידת הצורך להחזיר חזרה לרשת את המסר שהמנות הגיעו משובשות ולבקש לשלוח את השדר שוב. הפרוטוקול הידוע לשכבה זו הוא פרוטוקול TCP.

ג. Network – הרשת

השכבה שמקבלת משכבת הtransport את המידע כמנות ואת היעד המבוקש, והיא אחראית לבצע ברשת את פעולת הrouting – ניתוב, לבחור באיזו דרך להעביר את המנות. בשכבת הרשת יש אלגוריתמים שמחליטים עבור כל מנה באיזו נתיב היא תלך. שכבה זו מכירה רק את המקור ואת היעד, האם המנה עברה בצורה טובה או לא – היא לא יודעת, זו תהיה "בעיה" של שכבת הtransport. דגש – שכבה זו רק מחליטה איך להעביר כל מנה בדרך אבל היא לא זו שתבצע את ההעברה בפועל – זו תהיה המשימה של השכבה שמתחתיה.

ד. Link – הקשר

השכבה האחראית על ההעברה בפועל לפי הוראות ההגעה שניתנו לה בשכבת הnetwork. שכה זו מטפלת בקשר בין תחנות שכנות, כלומר בין שתי תחנות שיש ביניהן חיבור פיזי שמחבר באופן ישיר בלי תחנה נוספת באמצע. לאחר מכן לאחר שמידע הועבר מתחנה ראשונה לשנייה השכנה לה – השנייה יכולה להעביר לשלישית וכו'.

ה. Physical – פיזית

בסופו של דבר מעבר לקודים והמערכות המסובכות – חייב להיות משהו פיזי, כמו שהזכרנו, שיעביר את המידע. מידע לא מופיע פתאום במקום אלא אם הועבר באמצעים פיזיים (כבל מתכת – חשמל/ סיב אופטי – אור / גלים אלקטרו מגנטיים – מועברים במרחב)

9. תהליך המעבר במודל:

משתמש באפליקציה יוצר מידע שמומר בשכבת האפליקציה לשדר מסוים שמועבר לtransport, שכבה זו מוסיפה למידע header – כותרת, מידע מסוים שרלוונטי רק לשכבת הtransport, בכך יוצרת מנות. המנה בשכבת הtransport נקראת segment. כעת היא יודעת לאן המנה מיועדת אבל לא יודעת איך להעביר אותה, לכן היא מעבירה את המידע עם הheader לשכבת הnetwork. שכבת הnetwork מוסיפה למידע header משלה שנקרא Hm , ומחליטה באיזה נתיב להעביר כל מנה, ולכן מעבירה לשכבת הlink שמבצעת את ההעברה בפועל (עם הוספה של header משלה שנקרא frame) כך שההעברה תתבצע באמצעות השכבה הפיזית.

לאורך הדרך – בתחנות הביניים - יבואו לידי ביטוי רק 3 השכבות האחרונות, לא תהיה משמעות לשכבת האפליקציה והtransport מכיוון שהמידע לא צריך להיות מעובד למשתמש חיצוני, וגם לא צריך להיות מחולק שוב למנות או מאוחד חזרה למידע אחד.

כל אחת מהשכבות מקבלת את המידע עם כל הheaders ורואה את הheader שלה ויודעת שהמידע הזה ממוען אליה ולכן לא משדרת אותו – כך מבדילה האם המידע הגיע מלמעלה או מלמטה ויודעת כיצד ולמי להעביר.

10. הבהרה חשובה – מודל השכבות זהו לא מודל פיזי אלא תאורטי, לא ניתן לפתוח את המחשב ולראות כל רכיב מבין חמשת השכבות. זהו רעיון שמאפשר להבין לעומק יותר איך עובד המחשב ואת התהליך של העברת המידע באופן מלא.

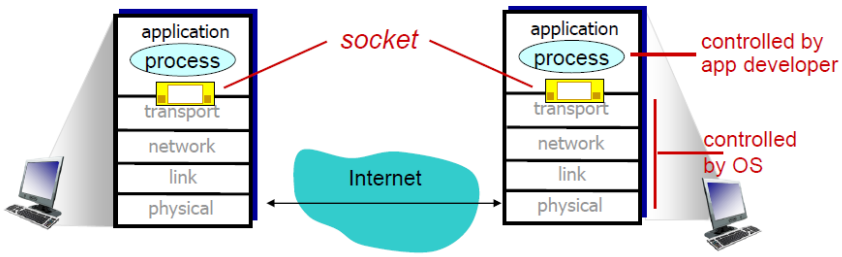
שיעור 5

עוד על שכבת הapplication

1. בשכבת האפליקציה ישנו מושג שחשוב להכיר – process.

Process הוא למעשה תוכנה או יותר מובן - צרכן תקשורת שרץ בתוך מחשב, ובכל מחשב / נקודת קצה (host) יכולים לרוץ מספר תהליכים במקביל. דוגמא לצרכן תקשורת – דפדפן, אבל הוא מכיל מספר תהליכים – למשל כאשר פתוחים אצלנו מספר חלוניות בדפדפן – כל לשונית מייצרת תקשורת עצמאית ומחוברת לשרת ספציפי מסוים – של יוטיוב/ גוגל וכו'. דוגמא נוספת היא המייל – בו יש תהליך של דואר נכנס ותהליך של דואר יוצא. התהליכים למעשה לא תלויים אחד בשני, תוכנה אחת יכולה לייצר מספר processes. ואם דיברנו על ארכיטקטורת client server יש לנו תהליכים שרצים בclient שזה תמיד במשיכה, כלומר לאחר שהלקוח ביקש לצפות בדף אינטרנט מסוים התהליך מתחיל לרוץ, וכאלו שרצים בserver בהם התהליך מתבצע בדחיפה למשל כאשר התקבל מייל.

שכבת האפליקציה כאמור, חייבת להתקשר אל שכבת הtransport, במיוחד כדי לבצע תהליך, לכן ישנו מושג – **Socket** שמשמעותו השער דרכו מתחברת האפליקציה אל שכבת התעבורה, האחראית לחלק למנות את המידע ולסדרו לקראת שילוח ברשת, וכל התקשורת בין 2 השכבות הנ"ל תעבור דרך השער הזה.



2. מושגים נוספים בשכבת האפליקציה:

Data-loss - משמעותו מה מידת הסבלנות שלי או יותר נכון האם אני מוכן לקבל מצב שהמידע משובש. במייל למשל – אני לא מוכן – no loss, כי אני לא רוצה שיהיה מצב שבו רק חלק מהמייל התקבל, ונבנה את האפליקציה כך שכל זמן שכל המידע לא התקבל הtransport ישלח דרישה להעברה מחדש. בסרטון או בחנות אפליקציות למשל אני יותר מוכן לקבל הפרעות בתקשורת כך שניתן למשל לעצור את הסרטון ולחכות שייטען עוד מידע – חלקו הוצג וחלקו עדיין לא הגיע, וכשיהיה חיבור יציב הסרטון ימשיך. לכן אני פחות רגיש לאיבוד מידע loss-tolerant.

Throughput – מהירות החיבור שמרגיש בפועל (כאמור זה שיש לי 100 מגה בבית לא אומר שזה מה שיהיה לי בפועל, בגלל ההפרעות בתקשורת).

Time sensitive- עד כמה אני רגיש לזמני התגובה/ שיהוי, הוא שונה בכך שThroughput נמדד על פני קטע זמן, והוא מודד <u>בממוצע</u> מה ביטים ירדו ביחידת זמן אחת. הTime sensitive אומר האם משנה לי האם היו שינויים במהירות התקשורת – באימייל למשל לא משנה לי אם המידע הועבר בהתחלה מהר ואז לאט או לא העיקר שהגיע, בשונה מסרטון המשודר בלייב שכן אכפת לי.

פרוטוקולים בשימוש שכבת האפליקציה

הפרוטוקולים הבאים פועלים <u>בשכבת הtransport</u> ואליהם מתחברת שכבת האפליקציה. כל אפליקציה יכולה לבחור באיזה פרוטוקול להשתמש בהתאם לצורך.

1. TCP service: - בגדול - "לעבוד נכון", תכונות:

reliable transport - הפרוטוקול מבטיח את העברת המידע באופן תקין גם כאשר הקו לא יציב, למשל ע"י דרישה להעברת המידע פעם נוספת. היתרון – אמינות, המחיר – רוחב הפס.

flow control: - בקרת זרימה, ובכך מונע מצב של איבוד מידע כאשר הsender מציף את הreciever (הדוגמא עם הכוס עם החור בתחתית). למשל כאשר מישהו אחר מכתיב לי משהו ואני כותב אנחנו "עובדים" עם TCP במנגנון של flow control.

congestion control: -בקרת העומס, כאשר הוא שולח מידע, במידה והוא מבחין בעומס ברשת הוא מוריד את קצב השליחה של הנתונים כדי לא להעמיס על הרשת, המחיר – מהירות העברה.

2. UDP service: - בגדול - "לעבוד מהר", תכונות:

unreliable data transfer – במובן מסוים אפשר לומר שהוא בדיוק הפוך מהפרוטוקול הקודם, הוא אחראי על העברת המידע בלי להתחשב האם התקבל או לא התקבל, או האם הרשת עמוסה או לא, ואם אבדו מנות בדרך הן אבדו לנצח. מתי זה אפקטיבי? למשל שידור של משחק כדורגל, למערכת אין סיבה לחזור אחורה ולהראות לך "מה היה לפני 5 שניות שלא הספקת לראות".

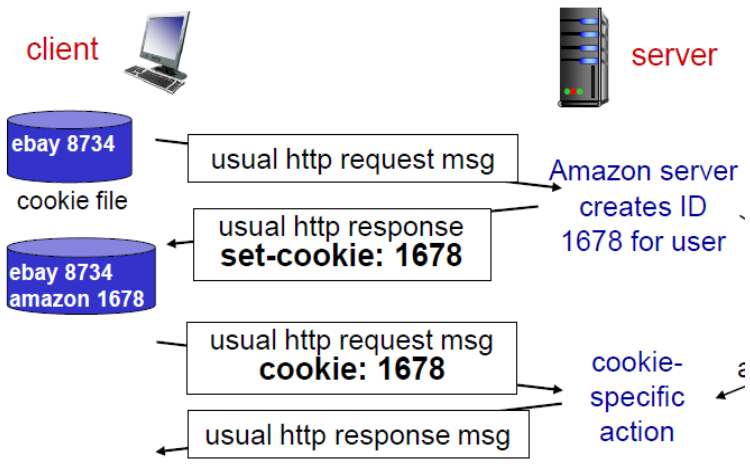
החיסרון – אמינות, היתרון – הרבה יותר מהיר.

3. Http - hypertext transfer protocol

כל דפי האינטרנט כתובים בשפה הנקראת html, המפרטת מה יהיה באתר ואיך יעוצב. מאחורי כל אתר אינטרנט יושב http server, וכאשר אני רוצה לגשת אל אתר מסוים הגישה לאתר (ולשרת) תהיה באמצעות פרוטוקול http היודע לגשת אל הweb-server בו מאוחסן המידע של האתר, ולקרוא את קבצי הhtml ולהציג אותם.

4. Cookies - עוגיות

לhttp יש תוסף מפורסם שנקרא cookies. לאחר שהתחברתי לאתר למשל, הדפדפן למעשה שותל אצלי במחשב מידע מסוים (למשל user id ספציפי לאתר הזה) כך שבפעם הבאה הדפדפן שלי ידע שוב שזה אני. (לכן אם אני מחפש בגוגל מכונת כביסה, הוא יציג לי באתרים אחרים פרסומות של מכונות כביסה). דוגמא נוספת היא סל קניות – על ידי



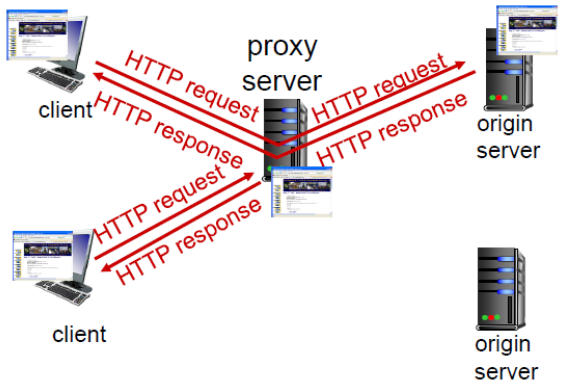
שמירת העוגיות אני יכול להוסיף פרטים לקנייה לעגלה כך שבפעם הבאה שאכנס הפריטים עדיין יהיו מוצגים לי, בלי שאצטרך בהכרח להירשם לאתר. עוד דוגמא - כניסה לאתר הדורש סיסמה לאחר שאני כבר הקלדתי אותה בעבר.

דגש חשוב – עוגיות זו תכונה השייכת פר מחשב, ולא פר אתר משתמש, העוגיות נשמרות ב-hard disk של המחשב עצמו, ואם אעבור למחשב אחר בלי שהתחברתי למשתמש שלי – הוא לא יזכור את הנתונים עליי.

התועלת של עוגיות זה חיסכון בזמן ויעילות, החיסרון – פוגע בפרטיות.

5. Cash – מטמון

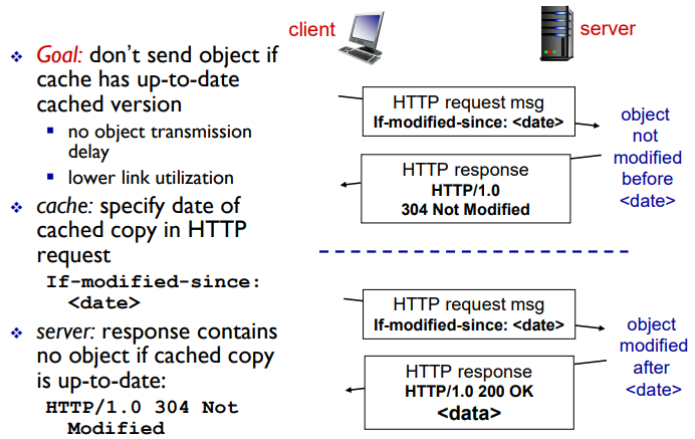
כאשר יש לי מידע מסוים שנמצא בשימוש פעמים רבות ביחס למידע אחר הנמצא באותו מקום, אני מעוניין שהדפים האלו יישארו זמינים יותר כך שהגישה אליהם תהיה מהירה יותר ולא יהיה צריך לטעון את כל המידע באופן מלא כל פעם מחדש. האתר של גוגל, למשל, שנמצא בשימוש פעמים רבות אצל כמעט כולנו, יושב באיזשהו שרת מחוץ לגבולות ישראל, ספק האינטרנט לא צריך כל פעם שמישהו נכנס לאתר של גוגל לקבל את המידע הרחוק כל כך בחו"ל, אלא הוא שומר את הגישה למידע אצלו כי מישהו, שגם משתמש בתשתית שלו, גלש כמה דקות לפני בגוגל. השרת שמאחסן את המטמון נקרא proxy, ובכך חוסך הרבה מרוחב הפס שהיה צריך להשתמש בו אילו לא היה משתמש במטמון.



שיעור 6

המשך על cash

1. אמרנו אם כן ששרת proxy הוא שמאחסן את המטמון - מידע מאתר מסוים הנמצא בשימוש פעמים רבות. איזו בעיה יכולה להיווצר בשימוש בשרת proxy? שהמידע שבו לא יהיה מעודכן אם האתר הראשי עדכן חלק מהמידע בו, שכעת אינו נמצא בשרת הproxy. על מנת לפתור את הבעיה הזו קיימת פקודה מיוחדת בhtml ולמעשה כחלק מפרוטוקול http, בשם conditional get. בפעם הראשונה כשאני נכנס לדף אינטרנט שלא נמצא אצלי במטמון (cash) אני מבצע פקודה בשם get מהhtml, בפעמים הבאות אני יכול לבצע conditional get ולדרוש מהשרת להציג לי את הדף ששמור אצלו, אבל יחד עם הבקשה להצגת האתר ששמור אצלו גם נשלחת בקשה - במידה והאתר המקורי עודכן מאז אותה שעה שאני משכתי את האתר אל שרת הproxy שלי - תחזיר לי את האתר המעודכן. זו הסיבה שפעמים רבות כשאנו מנסים לרענן דף אינטרנט ללא הצלחה זה יכול להיות בגלל בעיה או מידע ש"נתקע" במטמון, ואז הפתרון הוא למחוק את המטמון ולהיכנס לאתר מחדש.



פרוטוקול FTP

2. נזכיר כי אנו עדיין מדברים על השכבת האפליקציה, שלא מייצרת תקשורת אלא רק דורשת ומנגישה אותו. פרוטוקול ftp – file transfer protocol, והוא מאפשר העברה של קובץ שלם מצד לצד – מהשרת ללקוח ולפעמים גם מהלקוח לשרת. כאשר אנו למשל מבצעים הורדה של

קובץ מאתר אינטרנט מופעל ברקע ע"י הדפדפן שלנו פרוטוקול FTP והפרוטוקול מעביר את הקובץ משרת האתר אל המחשב שלנו, באותו אופן מתבצעת העלאה.
בעבר כאשר קובץ החל לרדת והייתה תקלה בתקשורת וההעברה נפסקה – ההורדה כולה הייתה מתבטלת, אך הפרוטוקול עבר שדרוגים וכיום אם ארעה תקלה בדרך הוא יודע להמשיך מאותו מקום שהפסיק לקבל בו את הנתונים. תכונה נוספת שיש לפרוטוקול היא אבטחה, שאם קובץ מוגן ע"י סיסמה הוא לא יאפשר להוריד אותו עד להכנסת סיסמה תקינה. בפועל פחות משתמשים ביכולת הזו של הפרוטוקול כי לרוב מיישמים את ההגנה ברמה יותר גבוהה, בין אם זה על הקובץ עצמו, ובין אם הדרישה לסיסמה היא בשלב הכניסה לאתר ולאחר מכן ניתן להפעיל את הFTP על קבצים באתר ללא בעיה.

אימייל

3. באימייל שלנו יש 3 מרכיבים:
   א. User agent – התוכנה, ה"סוכן" הוא למעשה הclient שנמצא אצלנו, למשל מייל של outlook או אם אנו משתמשים בgmail, הלקוח הוא הדפדפן שמציג את השרת של גוגל, או כל תוכנת מייל שנוריד. (מסומן בUA)
   ב. Mail server – שרתי דוא"ל, כל משתמש מייל חייב להיות מקושר לשרת דוא"ל שהוא מספק לו את התשתית, והשרתים הללו מקושרים ביניהם ומאפשרים את העברת המייל מאחד לשני.
   ג. פרוטוקול SMTP – ראשי תיבות של simple mail transfer protocol, ה"חוקים" להעברה של המייל.
4. כתובת המייל מורכבת משטרודל @, כך שמה שמופיע משמאל אליו זה שם המשתמש/המייל, ומימין אליו שם השרת. לא יכולים להיות שני שרתים עם אותו שם, ולא יכולים להיות באותו שרת שתי שמות משתמש עם אותו שם.
   אופן ההעברה תתבצע ע"י שליחת בקשת שליחה ע"י הUA אל הmail server שלי, ואז פרוטוקול SMTP פותח תקשורת TCP עם הmail server של היעד שיתורגם להודעה ע"י הUA שלו. ירידת המייל אל המייל של היעד יבוצע ע"י אחד משלושת הפרוטוקולים הנ"ל:
   א. POP – post office protocol – מאפשר ללקוח להתחבר אל השרת, להזדהות עם יוזר ולהוריד את המייל. לא נותן הורדה מתוחכמת או הגנה מתוחמת, הוא פשוט ובסיסי.
   ב. IMAP – internet mail access protocol – פרוטוקול קצת יותר מתוחכם המבצע את אותה פעולה אלא שמאפשר פעולות נוספות, למשל לנהל ספריות על השרת, ולסווג את המיילים לתיקיות.
   ג. HTTP – המייל נשמר בשרת של האתר ואנו מבקשים גישה אליו, והוא מצייר לנו את תיבת הדואר שלנו השמורה אצלו, למשל gmail, Hotmail, yahoo וכו'. היתרון שלו הוא בכך שהוא לא מחייב אותי להתקין כלום על המחשב שלי או לעדכן אותה מפעם לפעם – הכניסה היא תחת הרשאות אל האתר. החיסרון שלו הוא שהממשק פחות נוח לעבודה.

DNS

5. שירות נוסף ה"ניתן" בשכבת האפליקציה זה DNS, ראשי תיבות של domain name system/server. הכניסה לאתר מסויים מתבצעת ע"י כתובת הURL של האתר, אבל זו לא

כתובת שהמחשב מכיר, והוא מתרגם את זה בסוף לכתובת IP, המורכב מרצפים של רביעיות ספרות מופרדות בנקודה, ועל "תרגום" זה אחראי הDNS.

6. כדי להקים אתר אינטרנט צריך קודם כל להתחבר לשרתים ולהתקין עליהם את הדפים שלי, לאחר מכן אני צריך דומיין, שבאמצעותו יהיה ניתן לבצע תקשורת אל השרת שעליו שמורים הדפים שלי. לכן יש לפנות לרשם הדומיינים, ולרכוש את דומיין ייחודי, ולקשר אותו לIP הספציפי של השרת. לאחר מכן רשם הדומיינים יפיץ אותו בין שרתי הDNS ואז כל מי שירצה יוכל להיכנס אל האתר שלי.
7. שרתי DNS נתונים תחת מתקפות פריצה פעמים רבות כי קל מאוד לפרוץ אותם

א. DDoS- Distributed denial of service, נסביר קודם – המונח DOS זה למעשה מתקפה של מניעת שירות, זו לא מתקפה שנועדה לגנוב מידע אלא להשבית משאבים באינטרנט. המתקפה הזו היא ע"י הצפת הDNS בבקשות רבות מאוד של כתובות אתר. מאחר ולא סביר להניח שמחשב אחד יבצע כל כך הרבה בקשות לאתר מסוים מבצעים DDoS, כלומר את פעולת הDOS בצורה מבוזרת (Distributed) ממספר רב של מחשבים. ובכך מפילים את השרת/האתר – בכך שלא ניתן לגשת לDNS שלו שיתרגם את הכתובת של האתר ובכך לא יהיה ניתן לגשת אליו. אחת הדרכים לעקיפת המתקפה היא ע"י כך שהמחשב זוכר את הכתובת של האתר ולא נדרש לDNS שיתרגם לו את הURL לIP.

ב. מתקפה אחרת שהיא קשה יותר לעקיפה אבל גם קשה יותר לביצוע והיא redirect attack ובה לא משבשים את השרת אלא מצליחים להיכנס "בדרך" בין האתר לבין השרת, ומנתבים את הבקשה של לקוח מסוים מהאתר שרצה אל האתר שאני רוצה, ומעצב אותו בצורה דומה לאתר המבוקש, ובכך יכול "לגנוב" פרטים שהמשתמש יכניס/ יבצע.

8. לסיכום – שכבת האפליקציה מכילה בעיקר שירותים היוצרים אינטראקציה עם המשתמש, לשכבה זו פרוטוקולים אופייניים והיא עושה שימוש בשכבות שמתחתיה כדי לממש את התקשורת.

<u>שכבת הtransport</u>

9. כזכור שכבה זו משמשת ליצירת קשר לוגי בין תחנת המקור לתחנת היעד. נדגיש כי שכבה זו לא מכירה הנתיב עצמו ומה קורה בדרך, זה יהיה התפקיד של השכבה הבאה – שכבת הnetwork. השכבה מחלקת את המידע למנות לקראת העברתם ברשת, המנות בשכבה זו נקראות segments. השכבה מבטיחה אמינות (reliable) של המידע, וכן סדר, כך שהמידע לא יישלח או יתורגם כך שכאשר יורכב מחדש יורכב שלא לפי הסדר המקורי.
התפקיד האחרון של השכבה היא לבצע בקרת עומסים ולא להציף את הרשת במצב בו הרשת מלאה, בין אם להמתין מלשלוח את המידע, ובין אם לקבל פידבק מהצד השני במידה והוא לא מספיק לעמוד בעומס ואז הצד השולח מקטין את מהירות ההעברה. כמו שלמדנו בשכבה זו 2 הפרוטוקולים הבסיסיים הם TCP וUDP.
10. Multiplexing\ Demultiplexing – (ריבוב – מלשון ריבוי, ואי-ריבוב)
זוהי תכונה מאוד חשובה שמבצעת שכבת הtransport. בכל מחשב מופעלים בו זמנית הרבה מאוד צרכני תקשורת – אפליקציות ותוכנות שונות ונפרדות הדורשות כל אחת את המידע שלה בנפרד. קו המידע שמתקבל אצל המחשב שלי הוא אחד ושכבת הtransport יודעת לחלק כל מידע לsocket, מעין שער, של כל אפליקציה. וכן להפך – לקחת מידע מקו אחד ממנה – ולפצל אותו למספר תהליכים החוצה למספר שרתים.

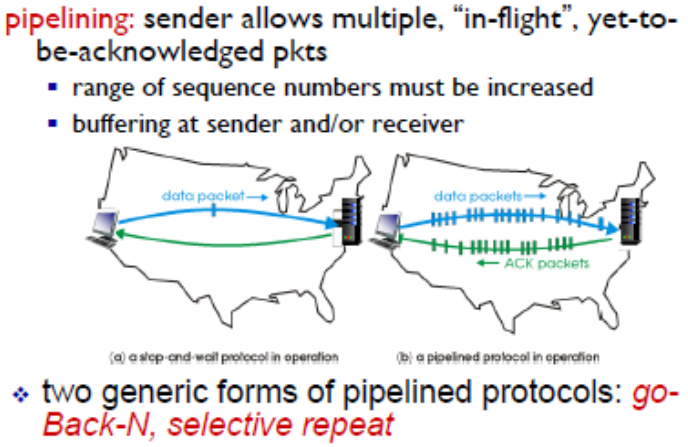
11. לכל אחד מאיתנו יש תעודת זהות שבסופה ישנו מספר המוגדר כספרת ביקורת, שע"י אלגוריתם מסוים יודע לחשב לפי יתר תעודת הזהות מהי ספרת הביקורת, ובכך אם אירעה שגיאה בהקלדה/ בהעברת המידע ספרת הביקורת (סביר להניח – 90%) תשתנה.
באותו אופן על מנת לוודא את אמינות הנתונים המועברים/המתקבלים יש למחשב מספר ביקורת, למשל internet CheckSum, שמייצר מספר ביקורת המוצמד אל המידע המועבר במנה. כאשר המידע מתקבל שכבת הtransport של היעד היודעת מה הchecksum שאמור להתקבל – בודקת האם זה מה שהיא קיבלה ואם הם זהים – מבינה שהמנה אמינה. במידה ואין התאמה – ינסה קודם כל לבצע תיקון – error correction - לבד על פי הchecksum שקיבל. אם עדיין לא הצליח לתקן – ישלח לצד השני בקשה לשליחה חוזרת של המידע.

שיעור 7

תזכורת על מנגנון ack ו nak בשכבת הtransport

1. ניזכר בפעולת המשדר ומקלט שלמדנו בשיעור הראשון (עמוד 5):
"למקלט יש מצב אחד – מחכה לקריאה מלמעלה – מהמשדר, אם הוא קיבל מנה והיא לא הושחתה (ע"י שימוש בchecksum) אז הוא שולף את הדאטה ומעביר אותה למשתמש או בדרך שבה הוא עובד (למשל ברמקול וכד') ובסוף שולח חזרה אות למשדר שהאות הגיע בצורה תקינה – ACK שזה למעשה קיצור של acknowledge – אישור. בינתיים המשדר נמצא במצב של המתנה – wait for ack or nak – אם הוא מקבל משוב מסוג ack אז הוא חוזר למצב המתנה לשלוח את המנה הבאה. אופציה אחרת זה שהמקלט בדק את הדאטה מול הchecksum וראה שיש אי התאמה – לכן הוא מבצע פעולה של corrupt כלומר לא משתמש במנה ולא מחזיר למשתמש (או לרמקול לפי איך שהוא עובד) כלום, ורק מחזיר למשדר אות של nak – כלומר not acknowledge" ולמעשה מבקש שליחה מחדש של המנה.

2. מה הבעיה המהותית בצורת עבודה זו? שאנו לא מנצלים בצורה טובה את הזמן, אני שולח שדר, ומחכה לקבל ack ובזמן הזה אני לא עושה כלום, ואם יוחזר לי nak אשלח שוב אבל הייתי יכול כבר לפני להתחיל לשלוח שוב את המנה מחדש. לצורך זה קיים פרוטוקול pipelined ששולח את המנות מספר פעמים ויודע לבצע בקרה על כך שעל כל המנות קיבל ack והכל הועבר בצורה מלאה.



עוד על פרוטוקול TCP

3. נגענו כבר בפרוטוקול TCP, הפרוטוקול המרכזי של שכבת הtransport (שיתקשר בהמשך לפרוטוקול המרכזי של שכבת הnetwork- IP). נרחיב על העקרונות של פרוטוקול TCP:
   א. Point to point – שולח אחד ומקבל אחד. תקשורת בין 2 נקודות בלבד. הוא לא מסוגל לנהל שיח מרובה משתתפים (דגש – בשכבה זו לא ניתן לבצע זאת, ניתן בשכבת

האפליקציה לפתוח הרבה ערוצי תקשורת שכל אחד מהם הוא TCP מול מספר גורמים ולאחד אותם כך שהמידע יוצג כשילוב של כולם, אך כאמור, לא יהיה ניתן לומר שזה בוצע בשכבת הtransport).

ב. Reliable and in-order – אמינות וסדר (כזכור, אמינות, וזמן ההצגה בצד השני יבואו על חשבון ביצועים)

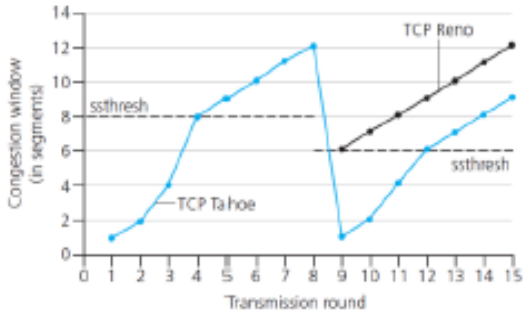
ג. Pipelined – העמסת הרשת במספר מנות בו זמנית כדי להגדיל את הניצול של השימוש ברשת

ד. Full duplex data – תקשורת דו כיוונית, כל צד יכול להיות גם sender וגם receiver, לעומת half duplex, שזו תקשורת חד כיוונית כמו רדיו, שרק הם משדרים אליי ולא אני אליה.

ה. Connection oriented – תקשורת מוכוונת/מונחית. הקמת קשר עם הצד השני, כלומר כדי לבצע את התקשורת נדרש להנחות את השולח עם מי להתקשר. כמו לדוגמא שיחת טלפון שנדרש להקליד מספר טלפון של היעד, או הקלדת כתובת אתר. ביצירת התקשורת נפתח סשן (ערות תקשורת), מתבצעת התקשורת, ונסגר הסשן. (חיוג, שיחה, ניתוק).

ו. Flow controlled – בקרת הצפה, השולח מתחשב ביכולת הקליטה של המקבל ולא יציף אותו במידע, למשל אם שלחתי להדפסה קובץ עם עשרות עמודים, המידע יישמר בbuffer של המדפסת (זיכרון זמני) אלא שהדפסה היא פעולה איטית ואם כמות המידע שנשלח להדפסה וממשיך לזרום לbuffer גדול מקצב יציאת הדפים – אנו עלולים לאבד מידע.

ז. Congestion control – בקרת עומס/גודש, מעבר לדרישה שלא להעמיס על המקבל, חשוב בנוסף לשים לב גם להתחשב ברשת, ולא להעמיס על הרשת כמות מידע שהיא לא תהיה מסוגלת להעביר.

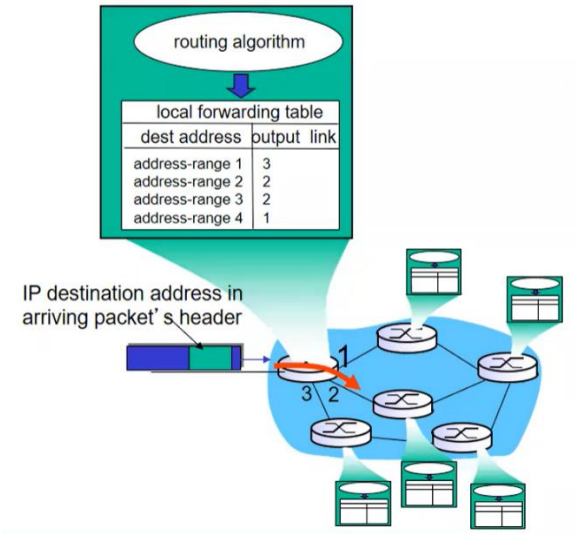


ה"אסטרטגיה" של congestion control זה להאיץ את המידע, המגולם ע"י המושג cwnd - congestion window, כלומר כמה מידע אני מעז לשדר בבת אחת. היתרון בלהאיץ את המידע זה לנצל את מלוא הרשת, אבל להסתכן באיבוד חלק מהמידע. לכן בהתחלה המידע מואץ בצורה אקספוננציאלית ככל שניתן, ואז כשמגיעים לנקודת סף מסוים שבו כנראה יווצר עומס ברשת מאטים ומעבירים את המידע בצורה הדרגתית (לינארית) – כמו שרואים בגרף, האצה מ1 ל4, ואז בסף (threshhold) עוברים לעלייה ב1 כל פעם עד ל"התנגשות" והגעה למצב של עומס ברשת, ואז עצירה מהעברת המידע וחזרה על אותו תהליך אלא שעם threshhold נמוך יותר – חצי מהערך המקסימלי אליו הגיעה. זוהי למעשה הבקרה על מניעת גודש ברשת.

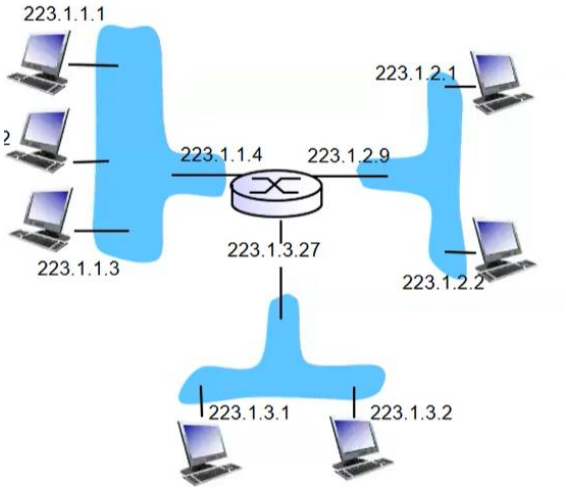
4. לסיכום שכבת הtransport - שכבת התעבורה יוצרת קשר לוגי (ולא פיזי) בין תחנות. השכבה מסוגלת להבטיח העברת מידע באופן אמין – שליחה/פיענוח לפי סדר/נכונות המידע, ולטפל בבעיות עומסים.

שכבת הnetwork

5. כזכור שכבת הnetwork היא מעין מודיעין קווי, שאחראית על החלטת המסלול עבור המנות, דרך אילו תחנות יעברו מתחנת המקור לתחנת היעד. היא לוקחת סגמנט (כך קרויה מנה בשכבת הtransport) ויוצרת datagrams (כך קרויה מנה בשכבת הnetwork). בכל אחת מהצמתים והתחנות שיעבור המידע בדרך אני חייב שיהיה פרוטוקול משכבת הnetwork (כשם שהוייז נדרש בכל צומת לומר לי לאן לפנות) איפה נמצא הפרוטוקול הזה? בראוטר (לכן שמו – מלשון rout, מסלול).
6. פעולות אלמנטריות:
   א. Routing – ניתוב, תכנון מסלול
   ב. Forwarding – קבלת מנה שנכנסה אליו והעברתה אל התחנה המתאימה הבאה
7. הניווט ברשת האינטרנט לא דורש שבכל תחנה (/ צומת) בדרך יתבצע חישוב מחדש לאיזו תחנה נדרש להעביר, אלא מתבסס על טבלאות מיתוג, כלומר קובע מראש נתיבים בתהליך הrouting לפי טווח של כתובות IP, למשל – אם כתובת הIP של היעד הוא בטווח כתובות הIP המסוימת הזו – תעביר אותה דרך output link מספר 3. (כמו מיון של דואר, אם זה למזה"ת תשים בסל הזה, ואת מה שבסל נמיין – אם זה לישראל תשים בסל הזה, ואז נמיין – אם זה לאזור הצפון תשים בסל הזה וכך הלאה עד שיחידת הדואר מקבלת את המכתבים הספציפיים הרלוונטיים אליה, ואז קל יותר לחלק.

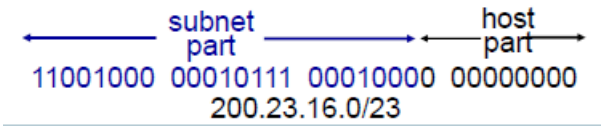


8. כתובת IP היא רשימה באורך קבוע של 32 ביט (אוסף של 32 סיביות שיכולות לקבל 1 או 0) ועם זה מורכבים כל הכתובות בעולם. אנו מכירים את כתובת הIP כרצף של רביעיות מספרים מופרדים בנקודה, זה לצורך היכולת שלנו "לקרוא" את הכתובת הספציפית אבל היא מצביעה על רצף ה0/1 שיושב מאחור. 32 הביטים מחולקים ל4 קבוצות של 8 ביטים, ממירים כל קבוצה כזו של 8 ביטים מספירה בינארית לספירה עשרונית.



Subnet – תת רשת

9. כאשר אנו הולכים לבית קפה/ אוניברסיטה וכו' שיש רשת וויי פיי פתוחה למשתמשים – אני מקבל כתובת IP חדשה, כי פתחתי ערוץ חדש למעבר הנתונים ממני ואליי, לכן יש דבר שנקרא subnet- תת רשת, לכל המשתמשים תחת אותה רשת בנוסף לכתובת הIP שאותה קובע ספק האינטרנט שלנו לכל



יחידת קצה (מחשב/ ראוטר/ שרת) יתווסף גם subnet-part שיהיה זהה לכולם פלוס host-part שזה מעין מונה פנימי של הראוטר ליחידות הקצה שלו (יכול להכיל עד 254 יחידות קצה כך שכל אחד יקבל ספירה פנימית משלו – 256, כלומר 2^8, פחות 2 כתובות שלא ניתן להשתמש בהן). בתמונה אנו יכולים לראות את כתובת הip, 23 אומר כמות הביטים התפוסים בsubnet. אם נשלח לsubnet ובhost-part נרשום 00000000 נתכוון לsubnet עצמו, ואם נרשום 11111111 נשלח לכל היחידות המחוברות לsubnet.

10. איך מתבצעת קבלת הIP לכל host?

ישנו מנגנון שנקרא DHCP – dynamic host configuration protocol (פרוטוקול תצורת מארח דינמי), כאשר מתחבר אליו מכשיר אחד – הוא מקבל את הכתובת הראשונה הפנויה, והשני יקבל את הפנויה הבאה וכך הלאה. הדינאמיות המדוברת באה לידי ביטוי בכך שהכתובות האלו יכולות להשתנות, אם אני יצאתי וחזרתי, או שהייתה הפסקת חשמל והתחברתי מחדש – אני אקבל כתובת חדשה.

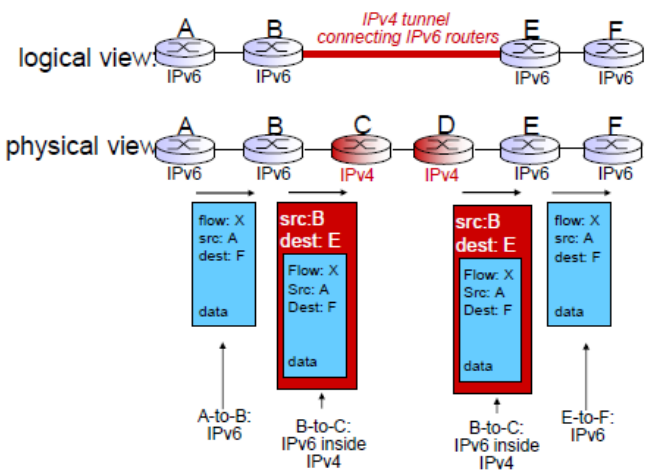
עם זאת ישנה בעיה עם DHCP וזה אם אני רוצה להתחבר אל מכשיר מרחוק, למשל לדוד חכם שאצלי בבית, אם כתובת הIP משתנה כל הזמן, מסיבות כאלו ואחרות – אני לא אצליח להתחבר אל הדוד להדליק אותו. יש כמה דרכים להתמודד עם זה, אחת הדרכים נקראת fix iprs, שזה מבין כל הכתובות שיש לי, להקצות למכשיר מסוים מספר ספציפי קבוע שתמיד ינותב אליו, אבל כמובן החיסרון בכתובת IP קבועה שהיא נתונה לתקלות / פריצות.

מנגנון NAT

11. מנגנון NAT- network address translation, הוא מנגנון המאפשר למספר מכשירים המחוברים לראוטר הביתי למשל, לצאת דרך כתובת IP אחת – של הראוטר. המנגנון שנמצא בראוטר שומר בטבלה את כל הקשרים שהיו לו – כל המכשירים שהתחברו אליו, ומתרגם החוצה כאשר קצר יוצא ממנו, וכשיקבל חזרה יידע לאיזה מבין המכשירים הספציפיים הוא נדרש להחזיר את המידע.

IPv4 / IPv6

12. סוג כתובת הIP המקובל נקרא IPv4, אבל עם התקדמות הטכנולוגיה חל גידול עצום בכתובות הIP הנדרשות, מעבר לגידול בכמות האנשים המשתמשים, המחשבים, הטלפונים וכו' נוספו גם שקעים חכמים, מכוניות עם מחשב שמסוגל לשדר וכו', עד למצב שנהיה עמוס וקושי להסתדר רק עם מרחב שפורס כתובת של 32 ביט. ולכן עלתה גרסה IPv6, המכיל כתובות IP בגודל 128 ביט. אבל כעת נדרשת היכולת להמיר את כל



הכתובות של ה32 ביט ל128 אחרת לא יהיה ניתן לבצע את המעבר. לכן נעשה שימוש בtunneling – ממירים את הכתובת של IPv6 הופכים אותה לדאטה, ואורזים אותה כמנה בפורמט של IPv4, ומעבירים את דרך ראוטרים של 32 עד לתחנת היעד שממירה חזרה לפורמט של 128.

13. המעבר של העולם לכתובות של 128 הוא איטי כי הרבה חברות לא יוצאות מגדרן "להירתם לבעיה העולמית" שכתובות הIP 32 ביט הולכים ומתמעטים, ולא מעוניינים להשקיע כספים במעבר הזה, אבל בסופו של דבר הtunneling מסייע בתיווך בין ה4 ל6 בעידן הזמני שיש את שתיהן, אבל הוא לא פותר את מצוקת הכתובות, אבל בסופו של דבר כולם יצטרכו לעבור לשיטה של 128 ביט.
14. לסיכום מהות שכבת הnetwork- שכבת הרשת אחראית לניתוב המידע, מעין הוייז של רשת האינטרנט. בשכבה זו ממומשת מערכת הכתובות מבוססת IP .

שיעור 8

שכבת הLink

1. בשכבת הקשר מנה נקראת frame, והיא למעשה מכילה את השכבות שמגיעות מעליה, ולמעשה מהרשת – datagram. אם בשכבת הרשת הוחלט לכל מנה הנתיב בה תעבור – שכבת הקשר מקבלת למעשה אוסף של תתי משימות, והיא זו שמנהלת את הקשר בין כל שני ראוטרים שכנים עד למעבר כל הפריימים מהיעד לכתובת.
2. שירותי שכבת הlink :

א. Flow control – התאמת הקצב בין 2 התחנות, למשל אם צד אחד מסוגל לשדר הרבה יותר מידע ממה שהצד השני מסוגל לקלוט. למשל – מחשב ומדפסת, נדרש להאט את קצב ההעברה למדפסת כדי להתחשב ביכולות שלה. לכן יש פרוטוקול בשכבת הקשר בה המדפסת יכולה לשדר למחשב להאט את הקצב, או ליתר דיוק כל פעם שמוכנה לקלוט מבקשת לשדר. אופציה אחרת שעלולה לקרות גם אם בפוטנציאל שני המכשירים מסוגלים לעבוד באותו קצב אבל אחד מהם "מוצף" ומקבל מידע מעוד הרבה מקורות אחרים.

ב. Error detection – זיהוי שגיאות במהלך ההעברה. (למשל כמו הדוגמה עם ספרת הביקורת בתעודת זהות).

ג. Error correction - ניסיון לתקן את השגיאות בהעברה. אנו עושים את הפעולה הזו כל הזמן כשמישהו מדבר אלינו, כך שאם הוא התבלבל באות/ מילה או שלא שמענו חלק בגלל רעש אנחנו מסוגלים להבין את המשמעות ולתקן את המידע בראש ונבין למה הכוונה.

ד. Half duplex and Full duplex – השכבה תומכת באפשרות גם של תקשורת דו כיוונית, או חד כיוונית. כל צד יכול להיות גם sender וגם receiver, או מצד שני שצד מסוים יהיה שולח בלבד, שזו תקשורת חד כיוונית כמו רדיו, שרק הם משדרים אליי ולא אני אליה.
השכבה מאפשרת לשני הראוטרים-התחנות-בדרך להעביר מידע בצורה חד או דו סטרית.

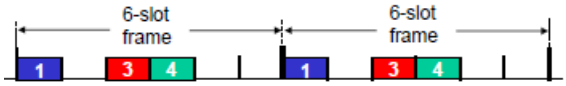
3. המקום העיקרי בו מתרחשים כל הפעולות הללו הוא בכרטיס שנמצא בכל מחשב ובכל תחנה חכמה – כרטיס רשת – או בשמו הרשמי NIC- network interface card, ומאפשר את הממשק בין המכשיר לרשת התקשורת, בין אם היא תהיה תקשורת קווית או אלחוטית.
כרטיס הרשת האלחוטי אם הוא קיים -הוא זה שמאפשר את החיבור לwifi.

4. רשת הLAN הינה הרעיון של חיבור ישיר של מספר מכשירים לראוטר/תחנה אחת, היישום המרכזי ביותר כיום של LAN נקרא אטרנט (לשים לב לא להתבלבל עם אינטרנט) שזהו למעשה התקן/ הפרוטוקול שעל פיו מתנהלת התקשורת ברשת. בתוך הראוטר יש רכיב שנקרא switch או מרכזייה (האופן שניתן להתחבר למשל לswitch זה ע"י למשל חיבור של כבל רשת לאחד מבין ארבעת החורים לכבל רשת שנמצאים בראוטר), הפורסת ומנהלת את רשת הLAN לכל המחשבים המחוברים אליה, למשל את כל המחשבים בבית.
(האטרנט הנוכחי הוא מסוג 802.11 – זהו תקן החיבור של רשת מקומית, כך שאם נראה זאת במפרט מחשב נבין שיש לו אפשרות להתחבר בכבל רשת לראוטר).

multiplexing

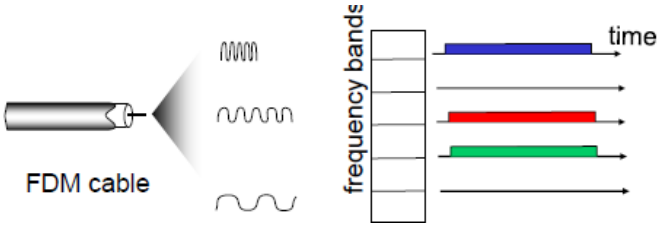
5. פונקציה מאוד חשובה שנמצאת בשכבת הקשר זה multiplexing. כאשר עוברת תקשורת בין 2 תחנות מרכזיות נדרש לווסת את ההעברה, ויסות זה נעשה ע"י הmultiplexing שהוא למעשה העיקרון של העברה של משתמשים רבים על אותו קו. זה נעשה ב2 שיטות:

א. TDMA - time division multiple access- זוהי גישה המאפשרת למשתמשים רבים לשדר, בו זמנית היא מגדירה לכל משתמש slot – תא זמן מסוים שבו יש לו את האפשרות לשדר בכל מחזור זמן אחד.



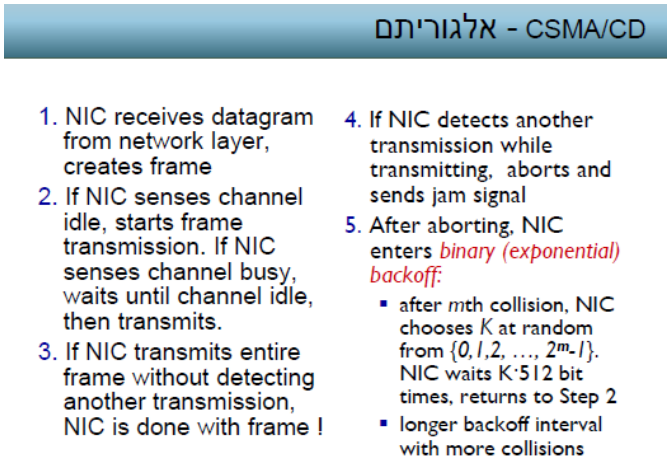
היתרון בו שניתן לשדר מספר משתמשים בו זמנית, החיסרון הוא שרוחב הפס הוא קבוע, ויש הבדל משמעותי אם אני מחלק אותו ל6 משתמשים או ל100 משתמשים שמשדרים על אותה תשתית, כי ככל שיש יותר משתמשים רוחב הפס האפקטיבי יירד. כלומר אני משדר ב1/500 של הזמן, לכן בפועל אני מקבל מהירות נמוכה יותר של האינטרנט בבית ממה ש"הובטח", כי המחיר שקניתי זה המהירות הממוצעת לאורך כל היממה, כולל באמצע הלילה שאין שותפים לקו.

ב. FDMA - frequency division multiple access – שיטה דומה לשיטה הקודמת רק שבמקום לחלק לכל משתמש מרווח זמן – מחלקים את הערוץ לתתי תדרים וכל משתמש מקבל תדר שונה. אבל גם פה כאשר יש יותר משתמשים אז בכל תדר ניתן להעביר במהירות נמוכה יותר. (שיטה זו מקובלת ברדיו למשל, שכל תחנה מקבלת תדר משלה).



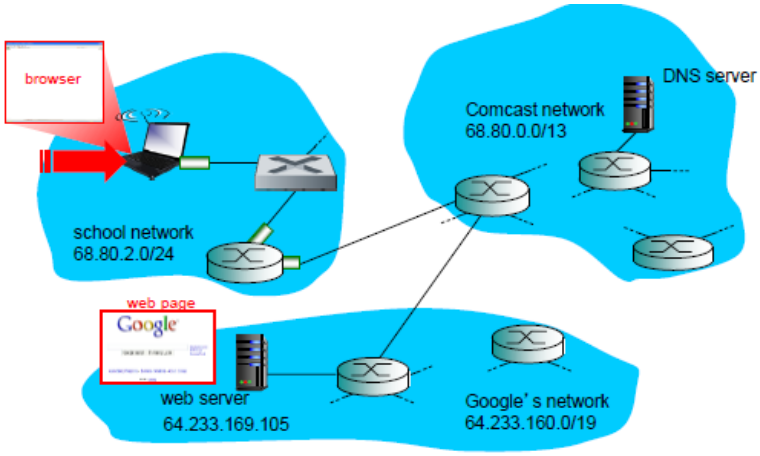
אלגוריתם CSMA\CD

6. ראינו כי שכבת הlink נועדה להעביר את המידע בפועל כמו גם לאפשר למספר תחנות לשוחח בתוך מתווך משותף, אך כאמור עלולות לצוץ מספר בעיות. לכך נועד הפרוטוקול העיקרי של שכבת הlink: CSMA/CD, ראשי תיבות של Carrier Sensing Multiple Access with Collision Detection - בעל יכולת לחוש גישה מרובת משתמשים עם זיהוי התנגשות. כדי להסביר את הרעיון שלו נסתכל על איך אנו מתקשרים – לפי ה"פרוטוקול", החוקים והנימוסים של השיח בין אנשים, לפני שאני מתחיל לדבר אני קודם מקשיב, ואז כשאני רואה שאף אחד אחר לא מדבר אני מדבר וכך יודע שישמעו ויבינו אותי. אם יצא מצב ועוד מישהו זיהה את השקט



וניסה להתחיל לדבר – קרתה התנגשות (Collision), ומיד שנינו נזהה זאת (Detection) ונפסיק בדיבור, כדי לא להיכנס אחד בדברי השני, אחד מאיתנו אולי יכריז "שקט רגע", כדי לנסות לסדר את שוב את השיח, וכל אחד יחכה פרק זמן קצר אחר (רנדומלי) עד שאחד מאיתנו יחזור וידבר והשני יזהה את זה ויחכה לתורו. באותו אופן מתבצע כאשר ישנה התנגשות של מספר משתמשים על אותו מתווך. כמובן שמהירויות התגובה והזמנים שבהם זה קורה הם קצרים מאוד, אבל אלו עדיין מהירויות סופיות, לכן י כולות להיווצר Collisions, וזה לא נמנע.

7. אם כן השלמנו את כל דרך המעבר של התקשורת מהמחשב האישי שלי, כאשר אני יושב בכיתה באוניברסיטה, למשל לדף של גוגל. מהמחשב שלי אני מתחבר לרשת LAN המקומית של הבניין ע"י פרוטוקול DHCP, שהוא מחובר לרשת WAN של האוניברסיטה, היא מחוברת לתחנות כך שבסוף מתחברת לשרת DNS ששומר עותק של הדף כדי לא לדרוש כל פעם את העדכון מגוגל, וכמו שלמדנו כאשר ישנם שינויים – מתחבר שוב לשרת המרוחק של גוגל ומקבל ממנו את המידע המעודכן על האתר.



שיעור 9

מתקפות סייבר

1. זהו סוג כלשהו של פעולה התקפית כנגד מרכיבי מחשוב. רבים מחשיבים את הזירה הקיברנטית בתור זירה נוספת במלחמה מעבר ליבשה, הים והאוויר. על מדינת ישראל מתבצעות מידי יום מעל 3000 (!) התקפות סייבר שלרוב נחמסות ע"י גופים האמונים על הגנת סייבר.

לרוב מסווגים את ה"סיבות" לכך שמישהו מבצע התקפת סייבר ל:

א. Fear factor – הפחדה

ב. Spectacular factor – נזק ממשי, למשל נעילת מחשב ודרישת כופר עבור הפתיחה

ג. Vulnerability factor – פגיעות, הקלות שניתן לפגוע בנתקף

בגדול ניתן לומר כי ישנם 2 סוגים של התקפות - תוכנות שמטרתן לגרום נזק:

א. Semantic attacks – לא נתמקד בה כרגע

ב. Synthetic attacks – נדרש להבדיל בין 3 סוגים של התקפות סינתטיות:

1) וירוס – מעין טפיל, שנטפל לתוכנות אחרות שהן כן לגיטימיות והופך להיות חלק מהן או רץ בסביבה שלהם. אם זה קורה הקובץ שלנו למעשה נגוע בוירוס, התוכנה לא עושה ב100% את מה שהיא נדרשה לו, ובמקביל מריצה פעולות נוספות כמו גניבת מידע/ ביצוע פעולות.

2) תולעת – "חיה עצמית", תוכנה שהיא עצמה כזו שנועדה לחבל, למשל הורדה מהאינטרנט תוכנה מסויימת מאתר לא אמין כך שבפועל זו תוכנה זדונית בעצמה שנועדה לתקוף את המחשב שלי.

2. התגוננות:

א. תוכנת אנטי-וירוס, שמותקנת באופן עצמאי לרוב. האנטי וירוס מתחבר לשרת של האינטרנט ומקבלת ממנו את המידע של המידע שמגיע למחשב ומנסה לנתר גורמים זרים. לאנטי וירוס יש חסרון מסוים והוא שהוא למעשה קו ההגנה האחרון, כלומר אם גורם זר מנסה לפרוץ אליי סימן שהוא עבר את כל ההגנות המקדימות של ישראל ושל ספק האינטרנט וכו', וכעת נשאר להגן ע"י תוכנה שנמצאת כבר בתוך המחשב.

ב. Firewall – חומת אש, לרוב לא יהיה במחשבים פרטיים אלא בעיקר בארגונים, שיציבו תוכנתfirewall כאמצעי שנמצא בין השרת של הרשת הפנימית של הארגון לבין הרשת החיצונית. יש במחשב פרטי פונקציה של firewall אבל זה לא באופן מובהק, כי זו בכל זאת תוכנה שיושבת כבר במחשב עצמו ולא בנקודה מעט רחוקה יותר.

ג. טכנולוגיות שונות, למשל הצפנה בהעברת המידע

ד. רשויות הגנה – לא סומכים רק על האדם הפרטי, שיידע להתגונן אלא קיימים גופים כמו למשל מערך הסייבר הלאומי וכד' המנסים לאתר פוסטים/ אתרים/ מתקפות ברשת.

ה. חינוך – לא משנה כמה תוכנות חכמות והגנות נשים - אם אדם מתנהג בצורה לא אחראית בסופו של דבר מתקפות סייבר יפגעו בו. למשל הרבה פעמים מקבלים הודעות פישינג, למשל "החבילה שלך הגיעה לדואר, נא מלא פרטי אשראי לקבלת החבילה", זו אמנם לא מתקפה "חכמה" ומורכבת, אבל אם אדם ישר ממלא פרטים ולא שואל את עצמו אם בכלל הזמין חבילה ולמה נדרש לשלם – סביר להניח שהוא מסר כרגע את פרטי האשראי שלו לגורם עוין.

<u>סייבר בעידן מחשוב ענן</u>

3. מחשוב ענן מספק לי משאבי מחשוב על פי דרישה. דוגמא למחשוב ענן - במקום לשמור קבצים ותוכנות על המחשב שלי, אני שומר אותו בענן כמו גוגל דרייב/ dropbox/ iCloud וכו' – שהוא למעשה שרת מרוחק, שאני מושך/דוחף את המידע אליו וממנו, כך שגם אם הגיעו למחשב שלי – הקבצים לא בהכרח יפגעו כי הם לא על המחשב שלי.

מחשוב ענן הוא לא רק לצורך זיכרון, אלא גם משאבי עיבוד, CPU, אם אני נדרש למשל לביצוע פעולה מורכבת מאוד שעלולה לקחת למחשב שלי הרבה מאוד זמן, כי אין לי מה לעשות בהמשך עם כוח עיבוד כל כך גדול. השיטה הזו נקראת on demand, כלומר קבלת שירותים לפי דרישה.

הבהרה - במחשוב ענן לא מספקים לנו דאטה on demand אלא משאבים on demand.

4. הרעיון של מחשוב ענן זה במקום לעשות את הדברים לבד – להזמין את זה כשירות. למשל במקום להכין את הפיצה, להזמין אותה. לכן כל דבר במחשוב ענן יסומן בAAS – as a service, למשל דיסק אחסון יסומן כSTaas- storage as a service.
5. יתרונות מחשוב ענן

א. נגישות –ACCESSIBILIT , אי תלות במיקום יחידת הקצה– אם אני רוצה לראות סרט, אני לא חייב להיות ליד המחשב שבו הוא שמור, אלא יכול ע"י מכשירים אחרים להגיע אליו וכו'.

ב. חיסכון - בעלויות רכישת ותחזוקה של ציוד חומרה / חיסכון בעלויות תוכנה/ בעלויות כוח אדם – במקום שכל מחשב לעצמו יוריד תוכנה/קובץ וישתמש בזיכרון שלו וכו' – כולם משתמשים באותו קובץ מרוחק. במצב זה מתקבל מצב של thin client כלומר שהמחשב עצמו משמש בעיקר להצגה של הנתונים, ולא בשמירה.

ג. אמינות RELIABILITY

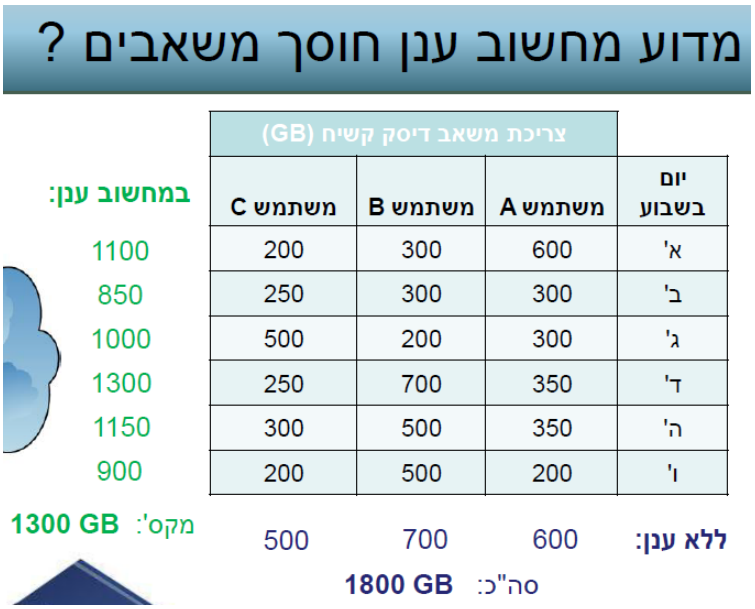
ד. התמודדות עם "פיקים" (Peak Load Capacity) – מאפשר לי להשתמש בכוח עיבוד/ זיכרון גבוה לתקופה מסויימת כפי שאני נדרש אליו בזמן 'פיק' מסוים.

ה. אין צורך במנגנונים לרציפות עסקית (BCP) – אם השרת של אתר מסוים למשל יקרוס כל חצי שעה – המשתמשים יפנו לאתרים אחרים, וזה יגרום להפסד כספים. לא נדרש שכל חברה תבנה לה מערכת שתעזור לה להתמודד עם נפילות כאלו כי הענן יספק את התמיכה הזו.

ו. אין צורך במנגנוני התאוששות מאסון (DRP) – באותו אופן, ארגונים פרטיים לא צריכים להקים כל אחד לעצמו מערכות במידה ויהיו נפילות של התשתיות.

ז. יכולת גידול (scalability) – אם נדרשים עוד מעבדים/זיכרון ניתן לצרפם לתשתיות הענן.

ח. מחשוב ירוק – אם מספר אנשים משתמשים באותה תשתית – נניח שנבנתה תשתית עבור הממוצע שימוש של כולם, יכולים להיות ימים שבהם משתמש אחד יצרוך יותר מכפי שבדרך כלל, סביר להניח שהוא היוכל "להשתמש" בחלק מהמשאבים שמשתמש אחר לא צרך באותו יום, כלומר אם המשתמש השני השתמש בפחות מבדרך כלל.



6. חסרונות מחשוב ענן

א. תלות בספק – לספק של הענן יש גישה למידע, שיש בדבר סיכון מסוים.

ב. אבטחת מידע – למשל כמו המקרה שהתגלה לפני מספר שנים שפייסבוק מכרו חלק מהמידע שהיה אצלה על משתמשים לצורך מניפולציות פוליטיות וכו'.

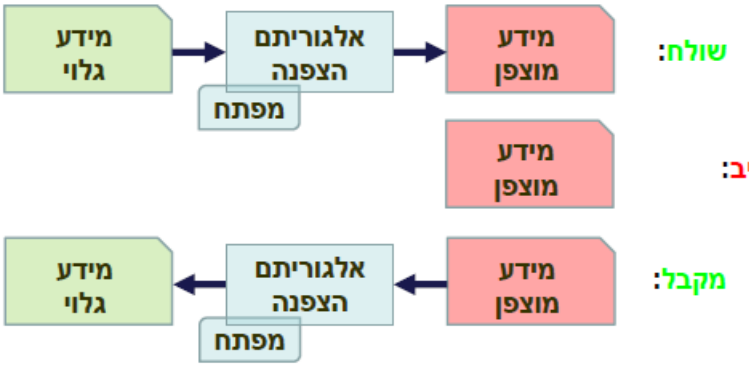
ג. פרטיות

ד. רישוי – לא ברור מבחינה משפטית מי הבעלים של המידע

שיעור 10

קריפטוגרפיה – תורת ההצפנה

1. קריפטוגרפיה (כתב סתרים - קריפטו=נסתר, גרפיה -כתיבה), או תורת ההצפנה היא ענף במתמטיקה ומדעי המחשב לעיסוק ומחקר בשיטות אבטחת מידע ותקשורת נתונים במרחב בו אנו יודעים שעלולים "להאזין" לנו, למעשה אי אפשר לקיים את מרחב הסייבר בצורה משמעותית בלי להשתמש בקריפטוגרפיה.
2. מודל ההצפנה עובד בצורה פשוטה – אני לוקח את המידע הגלוי, מבצע עליו מניפולציות ע"י אלגוריתם הצפנה ומקבל מידע מוצפן. אויב שיראה את ההודעה לאחר השינויים לא יוכל לפענח אותה, אבל הצד המקבל, שיודע לפי איזה רציונל/ אלגוריתם שיניתי את המידע יוכל להחזיר חזרה את הכתב המוצפן לצורתו המקורית.



הצפנה סימטרית וא-סימטרית

1. בהצפנה סימטרית המתפתח בו משתמשים לנעילה ולפתיחה של המידע, כלומר להצפנה ע"י השולח ופיענוח ע"י המקבל – הוא אותו מפתח. למשל אני מעביר הודעה עם מספרים – אני מכפיל כל מספר ב3, גורם זר לא ידוע את גורם ההכפלה הזה, והמקבל כשיקבל את המידע פשוט יחלק ב3 ויקבל את המידע.

   החסרונות של הצפנה סימטרית:

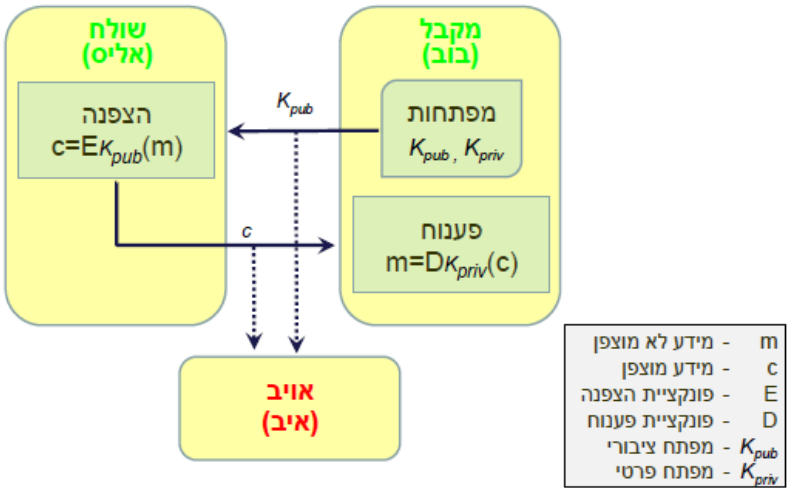
   א. נדרש לתאם עם המקבל מהו המפתח, וזה לא תמיד מתאפשר. אם אני בזום למשל רוצה לתאם עם מישהו שכאשר אני אומר מילה מסויימת אני מתכוון למשהו אחר – בפעמים הבאות זה אולי יצליח אבל כרגע, בשלב התיאום – אנשים כנראה ישמעו אותי. באמת אין לבעיה פתרון, מלבד להיפגש פיזית/ להעביר דף פיזי שרק המקבל יראה, ולאחר מכן נוכל להמשיך לנהל את השיח בצורה מוצפנת.

   פרדוקס ההצפנה הסימטרית היא שאם קיימת דרך בטוחה להעביר את השפה/המפתח להצפנה – אין צורך בהצפנה. אם אין דרך כזו – איך יוכלו להעביר מסרים מוצפנים ביניהם?

   ב. מחייבת החלפה תדירה של המפתחות.

   ג. ההצפנה לא מספקת אימות שהמקבל הוא אכן המקבל. כי אם מישהו עלה על המפתח – כל אחד יכול לבצע את הפיענוח.
2. בהצפנה א-סימטרית, ישנם 2 מפתחות להצפנה. מפתח ציבורי (להצפנה) ומפתח פרטי (לפענוח). המפתח הציבורי מופץ לכל דורש/ לקבוצה מוגדרת, המפתח הפרטי מצוי רק בידי המקבל, קיים קשר מתמטי (חד-חד ערכי) שקשה מאוד לאיתור, ולכן כל אחד יכול להצפין אבל רק בעל המפתח הפרטי יכול לפענח. איך מתבצעת ההצפנה? ניקח משל שבו אני רוצה לקבל יהלום ממישהו, אני שולח אליו קופסה יחד עם מנעול פתוח, הוא שם את היהלום בקופסה וסוגר אותה עם המנעול

שלי, אבל רק לי יש את המפתח כך שכשיגיע אליי אוכל לפתוח אותו. באותו אופן – אני מפרסם את המפתח הציבורי שלי, האדם שרוצה לשלוח אליי מידע יצפין את המידע בעזרת המפתח הציבורי שלי וישלח אליי, ואני אתרגם עם המפתח הפרטי שלי.



3. כדי להבין מעט את הרעיון ניקח פעולה מתמטית למשל שצד אחד מאוד קל לעשות אבל צד שני לא, למשל – חזקה ושורש. מאוד קל לקחת מספר ולעשות לו חזקה, אבל למצוא שורש באיזו רמה של המספר נותן לי את המספר הרצוי זו פעולה קשה יותר. ההצפנה תיעשה עם מודלים מתמטיים הרבה יותר מורכבים, כך שפתרון המפתח הפרטי יהיה כמעט בלתי אפשרי.

RSA

4. הקונספט של ההצפנה הוצג לראשונה בסוף המאה ה18, המודל הראשון שפורסם היה פרוטוקול דיפי-הלמן, אך אחריו הגיע אלגוריתם RSA שהוא ע"י ריבסט, שמיר ואדלמן, שהוא פרוטוקול פשוט אבל חזק מאוד, וכיום הוא אלגוריתם ההצפנה הנפוץ ביותר.
5. הקדמה מתמטית –
   א. מספר ראשוני הוא מספר המתחלק רק בעצמו וב1, מספר פריק מתחלק בעוד גורמים מלבד עצמו ו1, למשל 12=2*2*3, 135240=2*2*2*3*5*7*23. לפי המשפט היסודי של האריתמטיקה כל מספר טבעי יכול להיכתב כמכפלה אחת ויחידה של מספרים ראשוניים. לא ניתן לבחור מספרים ראשוניים אחרים שיתנו את אותו מספר.
   ב. 2 מספרים נחשבים "זרים" אם אין להם שום גורם מחלק משותף, למשל 18 ו12 לא זרים, כי לשניהם יש את 2 וגם את 3 כראשוניים המחלקים אותם. 7 ו20 למשל הם זרים.
   ג. האות היוונית φ - פי (fi) מסמלת את כמות המספרים שקטנים ממספר מסוים וזרים לו.
   למשל $\varphi 8 = 4$, המספרים הזרים וקטנים ממנו הם 1,3,5,7. φ של n כאשר n מספר ראשוני יהיה תמיד המספר פחות 1, כי ראשוני מתחלק רק בעצמו וב1, לכן כל מי שקטן ממנו גם זר לו.
   ד. מודולו, בקיצור mod, מסומן ב%– הוא שארית לאחר החלוקה. למשל 8%5 זה 3, 17%4 זה 1.
6. האלגוריתם –
   א. נבחר 2 מספרים ראשוניים <u>גדולים מאוד</u>, השונים זה מזה – p,q.
   ב. נקבע את המפתח הציבורי להיות מכפלת 2 המספרים, כלומר n = Kpublic1 = p*q
   ה φ של p*q הוא למעשה φ (n)=(p-1)*(q-1), כי כאמור שניהם ראשוניים לכן הפי שלהם הוא הם פחות 1. נשים לב כי (p-1)*(q-1) הוא מספר זוגי.
   ג. נבחר מספר אי זוגי e בתור מפתח ציבורי שני Kpublic2 כך שהוא שלם חיובי המקיים:
      a. 1< Kpublic2< Kpublic1

b. (p-1)*(q-1) ו- Kpublic2 זרים.

ד. נחשב את d בתור המפתח הפרטי Kprivate ע"י מציאת מספר המקיים את המשוואה הבאה:

(Kprivate* Kpublic2) mod [(p-1)(q-1)] = 1

7. קיבלנו 2 מפתחות ציבוריים ומפתח פרטי. כעת אני יכול לשדר את הציבוריים ולפענח חזרה.

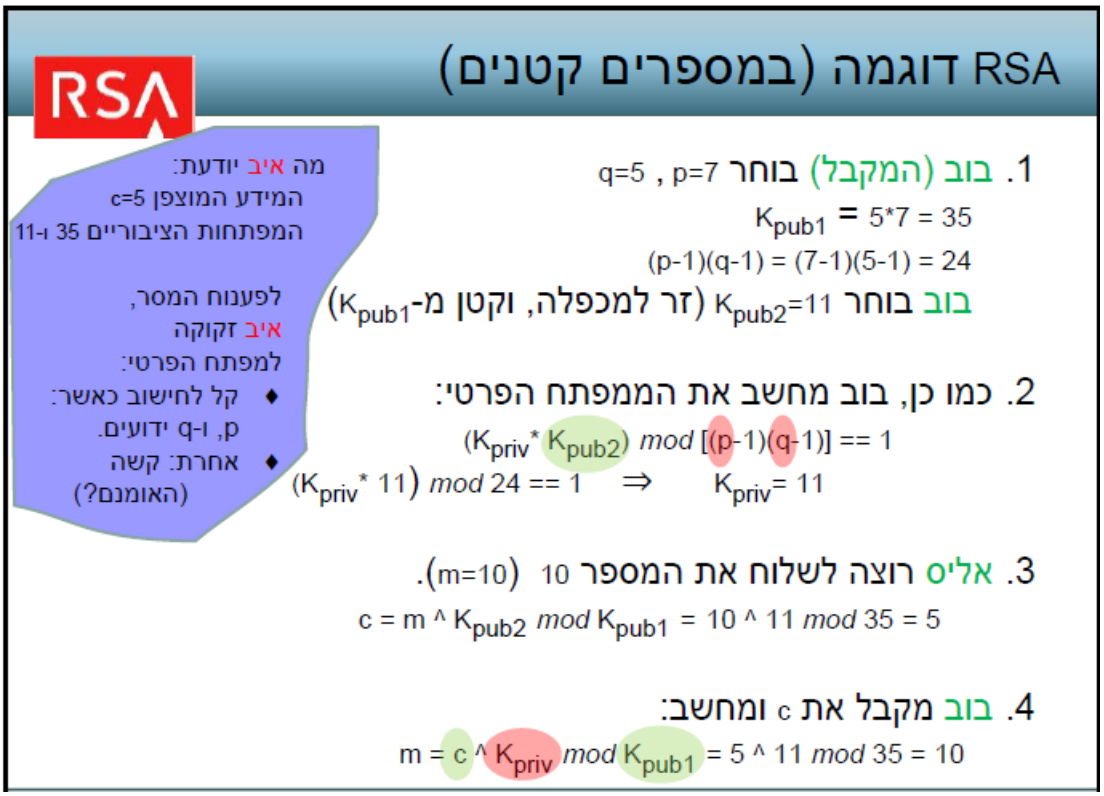
c יסמן את המידע המוצפן ו m זו ההודעה עצמה שאני מעוניין להעביר:

$C = m^{Kpublic2} \bmod Kpublic1$

והפיענוח – כלומר לקבל חזרה את m יתבצע ע"י:

$m = C^{Kprivate} \bmod Kpublic1$

8. דוגמה במספרים קטנים:



כאשר המספרים קטנים – אם 2 המפתחות הציבוריים ידועים – קל מאוד למצוא את את המפתח הפרטי, אבל מקובל שגודל מפתח מינימלי להצפנה הוא 1024 ביט, כלומר מספר ראשוני עם 350 ספרות.

הערה חשובה, הצפנה כזו ע"י מפתח בגודל 1024 ביט מקביל בהצפנה סימטרית למפתח בגודל 80 ביט (!) בלבד, אבל כאמור להצפנה סימטרית יש את החסירון שלא ניתן להיפגש פיזית עם כל אחד שאני מעוניין לבצע איתו תקשורת ו"לסגור" מראש על אופן ההצפנה.

9. דרך מסויימת לחסוך את השימוש הכל כך רחב בתקשורת א-סימטרית היא בפעם הראשונה לבצע תקשורת א-סימטרית ולשלוח בצורה מוצפנת את אופן התקשורת הסימטרית, למשל שכל מילה שאשלח תהיה כתובה באתב"ש (א=ת, ב=ש, ג=ר וכו') וכעת ניתן להמשיך בתקשורת סימטרית.

10. חסרונות של הצפנה א-סימטרית:

א. צריכת משאבי מחשב גבוהה (גורר איטיות)

ב. מפתח ההצפנה ארוך מאוד

ג. אין סכמת הצפנה שהוכיחו מתמטית כבטוחה ובלתי פריצה.

הצפנת טקסט

11. המחשב לא יודע לעבד אותיות, אלא רק מספרים, לכן כדי להצפין ניתן להשתמש בטבלת ASCII (הסקי) שבה כל אות – גדולה או קטנה, או סימנים כמו נקודה, פסיק, סימן שאלה וכו' מומרים למספר מסוים ומועברים כמספר ומתורגמים חזרה. (זו לא הצפנה, זו פשוט טרנספורמציה מתו למספר שיכול לעבור ברשת). הפורמט העכשווי לאור ריבוי השפות והפונטים והתווים השונים נקרא Unicode.

שיעור 11

לוחמת סייבר - Cyberwarfare

1. עולם מתקפות הסייבר כיום הינו מכלול עצום של גורמים, חלקם לגיטימיים יותר וחלקן פחות, החל ממתקפות של מדינות אויבות, או גופים או ארגונים פוליטיים כנגד מתחרים שלהם, או בין חברות מסחריות המבצעות ריגול תעשייתי וכד' על מנת ליצור נזק בתקשורת של הצד השני או לצורך גניבת מידע/ שימוש במידע של הצד השני על מנת להכשיל את צעדיו או לפעול בצורה טובה יותר. בהתאם למטרות.
2. הגורמים המשתתפים בלוחמת סייבר:

א. המרחב הקיברנטי:

a. רשת הinternat – רשת האינטרנט המוכרת לנו.

b. רשת הintranet – רשת עצמאית, הפועלת בטכנולוגיה זהה לחלוטין לרשת האינטרנט – כולל 5 השכבות, TCP, IP, וכו' אלא שהיא מנותקת מרשת האינטרנט הרגילה של העולם החיצון אלא רשת פנים ארגונית. הדוגמה הטובה ביותר לכך היא הרשת הצבאית.

c. Darknet – רשת היושבת על האינטרנט עצמו, וחלק ממנה, כך שאם אינטראנט היא רשת שמבודדת החוצה – הדארקנט מבודדת לוגית, כלומר יש לה למשל אוסף של משתמשים מורשים, שפעילותם נשמרת ע"י מערכות הגנה וכו'. לרוב הפעילות בדארקנט היא לא לגיטימית ולצורכי דברים לא חוקיים, אבל לא רק.

ב. יצרני חומרה ומערכות הפעלה – כל חברות המחשבים או יצרניות מערכות ההפעלה, למשל windows של מיקרוסופט מספקת למחשב שלנו פתרונות הגנה בסיסיות על המחשב שלנו. (לכן גם אנו מקבלים מידי פעם הודעות על עדכוני גרסה/ עדכוני אבטחה שנועדו להגן עלינו.

ג. ארגוני סייבר – למשל מערך הסייבר הלאומי בישראל, שזה איחוד של מספר גופים שמטרתם היא לספק הגנת סייבר עבור התקשורת הנכנסת והיוצאת מישראל ולנתר גורמים עוינים. או חברות פרטיות כמו checkpoint וכד'.

ד. ממשלות – למשל מתקפות של ישראל כנגד איראן על מנת לשבש את תהליכי הגרעין שלהם, או נסיונות של גורמים עוינים נגדנו.

ה. מחוקקים ורגולטורים – החיים שלנו נמצאים בסייבר, ברשת, ולכן עלולים להגיע לכל מקום ולכן נדרשת חקיקה מה מותר ומה אסור לחברות לעשות, כדוגמת האזנה לטלפונים ועוד דברים רבים שנועדו לשמור על המרחב הקיברנטי בטוח.

3. תקיפה כוללת לרוב את השלבים הבאים: איסוף מידע, הפצה, פגיעה, מחיקת עקבות. אם ניקח לדוגמה מתקפת פישינג – הודעה זדונית במסווה תמים, למשל הודעה כאילו מהדואר שדורשת להכניס פרטי אשראי לצורך קבלת חבילה. איסוף המידע הוא איסוף מספרי טלפון של אנשים דרך אתרים וכו' (לכן אנו נדרשים לשים לב איפה אנחנו מכניסים פרטים שלנו, גם כאלו שהם לא

פרטים סודיים בהכרח), הפצה של ההודעה לכל המספרים שנאספו, פגיעה ע"י שימוש בפרטי כרטיס האשראי שהוכנס ע"י אחד מהנמענים, ולבסוף לנסות למחוק את היכולת של משתמש בהמשך לזהות מאיזה כתובת IP התחברתי לאתר הבנק וביצעתי את הפעולות.
פעולת התקיפה יכולה להתבצע נגד כל אחת מ5 השכבות, אם זה בקליטת שדר הWIFI שזהו השכבה הפיזית, או לשתול וירוס שזו שכבת האפליקציה.
הערה – רובן המכריע של התקיפות אינן זוכות לחשיפה ציבורית. למשל בנק שבוצעה נגדו מתקפת סייבר, בין אם הצליח לבלום אותה ובין אם לא, ואפילו אם יש לו את הכתובת של הגורם הפוגע, כנראה יימנעו מללכת למשטרה להגיש תלונה כי העובדה שניסו לפרוץ אליהם עלול להוביל לבהלה בציבור ולעזיבה של אנשים את שירותיהם.

<u>סוגי תקיפות</u>

4. man in the middle – MITM

זוהי תקיפה קלאסית הקשה מאוד לביצוע, אך אם היא מצליחה יש לה פגיעה מאוד משמעותית שהאסטרטגיה שלה היא 'להיכנס' בטווח תקשורת הזה בין 2 גורמים השולחים תקשורת ביניהם כך שהתקשורת בהכרח תעבור דרכם. תקיפה זאת מתבצעת במקרי הצפנה א-סימטרית.
נניח שבוב ואליס רוצים להיפגש לצורך מטרה סודית, איב רוצה לבצע תקיפה ולמנוע את המפגש ולחטוף את בוב, והיא תבצע זאת באופן הבא:

א. איב מתחזה לבוב, ואליס שולחת (לאיב) בקשה להתקשרות ע"י בקשת המפתח הציבורי של בוב. איב מעבירה לבוב בקשה שישלח את המפתח שלו.
ב. כאשר בוב שולח לאיב (שחושב שהיא אליס) את המפתח שלו, איב מחזירה לאליס את המפתח הציבורי **<u>שלה.</u>**
ג. אליס מחזירה לבוב (לדעתה) "פגוש אותי בתחנת האוטובוס", התוכן המוצפן מתורגם אצל איב ע"י המפתח הפרטי שלה, שכן אליס השתמשה במפתח הציבורי של איב.
ד. איב מחזירה לבוב (שחושב שהיא אליס) ע"י המפתח הציבורי שלו שקיבלה ממנו בהתחלה - "פגוש אותי מאחורי הקניון"
ה. בוב חושב שקיבל את ההודעה מאליס ולכן מגיע אל מאחורי הקניון ושם איב חוטפת אותו.

5. דרכי התגוננות מMITM

א. Authentication – אוטנטיזציה, אימות
כגון שימוש בCA - Certificate Authority (רשות האישורים) – גורם באמצע המודע למפתחות הציבוריים של שני הצדדים ומאשר את המפתחות הציבוריים שנשלחים.
ב. Tamper detection – זיהוי חבלה
כגון איתור השהייה בזמני התגובה, שהרי אם ניקח את דוגמת התקיפה של איב, התהליך הזה של המעבר באמצע דרך איב גורר השהייה מסויימת במהירות התקשורת. המערכת יכולה לחשוד שיש גורם זר ש"התיישב" באמצע.
ג. Forensic analysis – ניתוח משפטי (הרחבת אופקים - פורנזי=פלילי)
כגון בחינת כתובות IP, בדיקה שאכן כתובת הIP שנמסרה לנו שאליה צריך לשלוח – היא אכן הכתובת של היעד.

6. denial of service – DOS

סוג תקיפות הקל בהרבה יותר לביצוע, אך גם מזיק הרבה פחות, והוא עוסק במניעת שירות, וכבר דיברנו עליו בשיעור 6 כאשר דיברנו על שרת הDNS, שהוא שרת (או למעשה רשת של שרתים) האחראים לתרגם את כתובת אתר אינטרנט (url) לכתובת ip ובכך לבצע את התקשורת. המתקפה הזו היא ע"י הצפת הDNS בבקשות רבות מאוד של כתובות אתר. מאחר ולא סביר להניח שמחשב אחד יבצע כל כך הרבה בקשות לאתר מסוים מבצעים DDoS, כלומר את פעולת הDOS בצורה מבוזרת (Distributed) ממספר רב של מחשבים. ובכך מפילים את השרת/האתר שנדרש לבצע כמות גדולה כזו של בקשות תרגום מurl לip – ובכך הDNS לא יצליח לתת למשתמשים בו את השירות המבוקש.

חשוב להבהיר, DDOS לא בהכרח יבוצע ע"י מחשבים שכולם של הארגון התוקף, יכול להיות שהוא ישתיל וירוס בהמון מחשבים לגיטימיים של משתמשים, ובזמן קבוע ומוגדר מראש יפעיל את כולם מרחוק בתקיפה של הDNS.

7. סוגי מתקפות בDOS:

א. Ping – פרוטוקול ברשת שנועד לבדוק את זמן התגובה בתקשורת לאתר מסוים, המחשב למעשה שולח לשרת של האתר "פינג" והוא עונה לי "פונג", ואז הוא בודק כמה זמן עבר עד שקיבל חזרה תגובה. אם אנחנו מציפים שרת של אתר מסוים בהמון בקשות פינג אנחנו למעשה "מסיתים" לזמן מסוים את הקשב שלו מהפעולות הרגילות שלו על מנת "לענות" לנו.

ב. SYN flooding

ג. Jamming attack

ד. DHCP spoofing

ה. DNS Dos - דיברנו

ו. Packet drop attack – מתקפה קשה יותר בה אני משתלט על חלק מהמנות שנשלחות לאותו שרת ומשמיד אותם באיזושהי דרך, ואז הנמען לא מקבל את המידע המלא.

הנזק שמתקפות DOS גורם הוא מינורי יחסית, בעיקר שיבוש או עיכוב מפעולה של הצד השני, ולרוב גם מאוד קל להתגונן מפני התקפות אלו.

8. MAC spoofing - זיוף מק

Mac הוא כתובת מקומית של מחשב ספציפי כאשר הוא מתחבר לרשת אינטרנט, כתובת הmac - (medie access control) מקודדת בכרטיס הרשת של המחשב בשלב ייצורו, אין 2 מכשירים עם אותה כתובת mac.

מתקפת MAC spoofing היא מתקפה ע"י התחזות למחשב אחר ע"י שימוש בmac שלו, למשל כניסה לרשת של האוניברסיטה ושינוי כתובת המייל שלי כך שהשרת יחשוב שהמייל שלי הוא מייל פנים ארגוני.

מתקפה זו איננה עבירה על החוק אך עשויה לרמוז על כוונות זדון, כגון במקרה ארון הלל שוורץ שהתחזה לכתובת מייל של האוניברסיטה שלו והוריד עשרות מאמרים ומחקרים שהיו בתשלום למנויים ופרסם אותם לכל מי שירצה, כי חשב שלא הגיוני ששירות זה יהיה בתשלום.

9. IP address spoofing

באותו אופן כמו mac spoofing אלא שבמקום התחזות לmac ותקיפה ע"י השכבה השנייה, זוהי תקיפה ע"י התחזות של IP כלומר תקיפה ע"י השכבה השלישית. זוהי התחזות מאוד נפוצה. ההתחזות למשתמש אחר, יכולה להיעשות למשל ע"י עקיפת מנגנון authentication. בפעמים רבות זו שיטה לא יעילה כי לא מספיק שעברת את האימות של הip, כי פעמים רבות תידרש גם לשם משתמש וסיסימה וזה שהתחברת דרך IP אחר לא מסייע לכניסה, אבל במתקפת DOS למשל זה כן יעיל, אם אני לא רוצה שינטרו למשל את כתובת הIP שלי.

10. XSS- cross site scripting (סקריפטים בין אתרים)

רעיון קצת מורכב יותר, ובו אני מתערב בקוד של אתר מסוים ומשתיל בו קוד שהאתר לא ישים לב לשינוי מהותי בו ובכך להשתמש במידע באתר. XSS מנצל פרצות אבטחה באתר המאפשרות בכך החדרת קוד זדוני.

11. הנדסה חברתית

התקיפה הנפוצה ביותר ברשת האינטרנט מול משתמשים פרטיים בעיקר, מאחר וארגונים גדולים/ ממשלתיים ישימו לב/ ינטרו את הניסיון בקלות. התקיפה היא ע"י ניסיון פריצה למערכות מחשב תוך ניצול תכונות פסיכולוגיות – לגרום לאדם להאמין במה שאומרים לו ובכך לבצע פעולות כנגד האינטרס שלו, לדוגמא מתקפת פישינג של הכנסת פרטי אשראי על מנת לקבל את החבילה. המשתמש לא מעוניין שלא לקבל את החבילה שלו ואם נדרש ממנו "רק" להכניס פרטי אשראי הוא יעשה זאת. או "התגלתה אצלכם פרצת אבטחה בfacebook, כנסו להחליף סיסמה על מנת לתקן אותה". או שיחת טלפון מנציג של חברה מסויימת שמציע לנו מכשיר מסוים בהנחה משמעותית אם נקנה עכשיו ונמסור פרטי אשראי.

<u>הגנת סייבר</u>

12. חתימה דיגיטלית

ראינו מקודם על הגנה באמצעות CA, שנועדה לאמת את הזהות של המשתמש. דוגמה נוספת היא חתימה דיגיטלית, כמו שיש לכל אדם את החתימה שלו שמעידה על זה שהוא זה האדם ששלח את המכתב – שחתום עליו, חתימה דיגיטלית מעידה על אותו דבר. היא למעשה פונקציה שמורכבת מהמידע עצמו, המייצרת רצף אותיות ומספרים (למעשה מספר בינארי המוצג בצורה אקסהדצימלית) ונשלחת בנפרד מהמידע עצמו, כך שאם בוצע השינוי אפילו הקל ביותר – הוא יגרום לשינוי הרצף. רוב ההגנה היא באמצעי זה.

דרך יישום נפוצה של חתימה דיגיטלית:

א. מפיקים מספר מייצג מהמסמך באמצעות פונקציית גיבוב Hash
ב. חותמים על המספר באמצעות המפתח הפרטי
ג. מקבל המסמך מאמת באמצעות חישוב הערך מפונקציית הגיבוב, פענוח המספר שהתקבל יהיה ע"י המפתח הציבורי שברשותו והשוואה ביניהם.

(הסבר – לכל אחד יש את המפתח הציבורי שלי, כך שאם אני שולח מידע המוצפן עם המפתח הפרטי שלי – כל אחד יכול לתרגם עם הציבורי הנמצא אצלו ולראות שזה אכן אני).

החתימה הדיגיטלית מעבר לאימות השולח גם מספקת כאמור אינדיקציה על שלמות המסמך – ולא בוצעו בו החסרות או הוספות של מידע

13. Public Key Infrastructure - PKI

תשתית כוללת לטיפול בכל הקשור למפתחות ציבוריים, כולל ניפוק של תעודה דיגיטלית לגוף הרוצה שיהיה לו מפתח ציבורי וכולם ידעו שאכן זהו המפתח הציבורי שלו. הבדיקה הזו מול גופים המספקים PKI מתבצעת פעמים רבות אך היא מתבצעת מאחורי הקלעים ואנו לא בהכרח מודעים לכך. ניפוק התעודות הדיגיטליות מבוצע ע"י רשות אמון CA.

ארגונים גדולים וממשלות עשויות להשתמש ברשויות אמון משלהן, לציבור הרחב זמינות רשויות שונות המופעלות ע"י חברות פרטיות כגון: Comodo, Symantec, Verizon

14. Web of Trust

PKI היא גישה ריכוזית, שיש גוף אחד שהכל עובר דרכו, בגישה זו של web of trust אין גורם אחד מרכז, אלא הגופים מפרסמים את המפתח שלהם וגופים אחרים שהם חברים ברשת הזו צריכים לאשר אותו על מנת להצטרף למאגר. (כמו ועדת קבלה לקיבוץ). גישה זו נפוצה ביותר בתחום המטבעות הדיגיטלית – ביטקוין וכד', ובו הכסף לא נתון ברגולציה של הממשלה אלא האחריות אליו מבוזרת בין כל המשתמשים.

15. HTTPS

כבר דיברנו לפני כן על פרוטוקול http – כל דפי האינטרנט כתובים בשפה הנקראת html, המפרטת מה יהיה באתר ואיך יעוצב. מאחורי כל אתר אינטרנט יושב http server, וכאשר אני רוצה לגשת אל אתר מסוים הגישה לאתר (ולשרת) תהיה באמצעות פרוטוקול http היודע לגשת אל הweb-server בו מאוחסן המידע של האתר, ולקרוא את קבצי הhtml ולהציג אותם.

אם כן http זהו קיצור של hypertext transfer protocol וhttps זה קיצור של hypertext transfer protocol secure, שנראה שהוא מופיע בתחילת כתובת האתר לרוב כשניכנס לאתרים כמו בנקים וכד'.

פרוטוקול https מוודא ע"י מנגנון PKI כי המפתח הציבורי שקיבל מאתר הבנק למשל הוא המפתח הציבורי הנכון, מוודא שהמידע עובר בצורה מוצפנת ע"י שימוש בפרוטוקול SSL/ TLS

16. Network Intrusion Detection System - NIDS (מערכת זיהוי חדירת רשת)

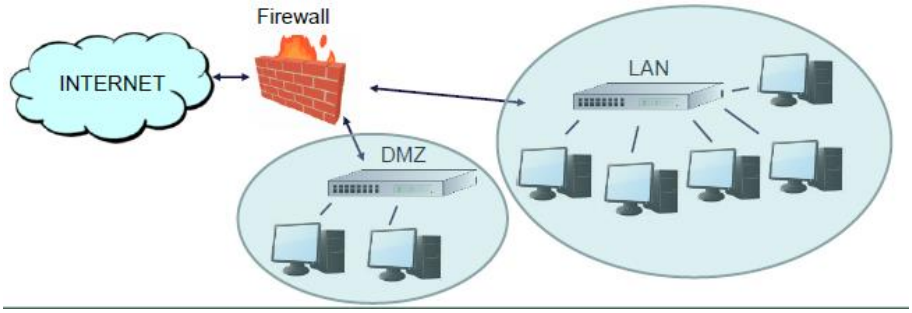
המערכת הזו יושבת על תשתית של גוף מסוים, למשל של האוניברסיטה וכד' וע"י חוקים מוגדרים מראש מנטרת את הרשת ומאתרת הפרה של החוקים. למשל אם בנסיון התחברות הוכנסה סיסמה 1001 ואז 1002, 1003 וכו' או מישהו ש3 פעמים טעה בהקשת הסיסמה– זוהי התנהגות שמעידה על כך שכנראה יש מישהו שמנסה לפרוץ לחשבון.

17. Honeypot

שתילה של נקודות תורפה מלאכותיות שכאשר תוקף ינסה לפרוץ הוא יגלה כי קל מאוד לפרוץ משם – וינסה להיכנס מאותה נקודה ושם ייחסם וכו', בין אם למטרת אבטחה בלבד, ובין אם למטרת תפיסת עבריינים. כאשר יש מספר מלכודות דבש נכנה זאת honey net.

18. Firewall / אנטי וירוס

חומת אש, למעשה גורם הנמצא בין הרשת של גוף מסוים לבין הרשת החיצונית, כך שנועדה להוות חומה כך שכאשר תוקף ינסה לפרוץ – הוא ייתקל קודם בחומה ויידרש לעבור דרכה,



בכך למעש מרחיקים את התוקף ומקשים על החדירה או חוסמת את התקשורת. מאחר וכאשר מישהו חדר ההגנה שנשארה היא בתוך המחשב, אנו מעוניינים שאם מנסה להיכנס - החסימה או "המלחמה" לא תתבצע "בשטח" המחשב אלא במקום חיצוני. מעין שומר בכניסה שמוודא מה נכנס ומה יוצא.

19. דגש אחרון - אמצעים טכנולוגיים לבדם לא יעילים, נדרשת גם התנהגות אחראית של המשתמש, ולוודא חינוך לטכנולוגיה ולהימנע למשל מ: פתיחת URL מתוך הודעה/ הורדה והתקנת תוכנה ממקור לא ידוע/ נדיבות במסירת מידע רגיש/ גלישה לאתרים המוגדרים כאיום.