פתרון בחינה בקורס מבוא לתקשורת 2002– מועד א'

<u>המרצה</u>: דני ביקסון
<u>תאריך הבחינה</u>: 5.7.02
<u>משך הבחינה</u>: 3 שעות.
<u>חומר עזר</u>: דף נוסחאות אישי בגודל A4. יש לוודא שמופיע שמכם עליו. בסיום הבחינה יש להגישו עם מחברת הבחינה.
<u>הוראות</u>: בבחינה 4 שאלות. יש לענות על שלוש שאלות מתוכן. ערכה של כל שאלה הינו 33 נקודות. ליד כל סעיף מופיע ערכו. במקרה שעניתם על 4 שאלות 3 השאלות הראשונות יבדקו.
בהצלחה! :-)

1. א. (25 נקודות) בשיטת שידור CDMA נתונות התחנות הבאות:

```
A: 0 0 0 1 1 0 1 1
B: 0 0 1 0 1 1 1 0
C: 0 1 0 1 1 1 0 0
D: 0 1 0 0 0 0 1 0
```

- חשב מה יהיה השידור כאשר A משדר 0 , B ו- C משדרים 1, ו- D לא משדר דבר.
- חשב מהו הפיענוח לגבי תחנה A.

<u>תשובה</u>: ראשית נתרגם את היצוג הוקטורי של התחנות ליצוג בי-פולרי:

A: (-1,-1,-1,1,1,-1,1,1)
B: (-1,-1,1,-1,1,1,1,-1)
C: (-1,1,-1,1,1,1,-1,-1)
D: (-1,1,-1,-1,-,1,-1,1,-1)

שידור ההודעה:

S = <u>A</u>+B+C = (-1,1,1,-1,1,3,-1,-3)

כלומר נסכום את היצוג של B ו-C יחד עם המשלים של A.

הפענוח של התשדורת של A:

<S*A>/8 = (-1,1,1,-1,1,3,-1,-3)*(-1,-1,-1,1,1,-1,1,1) = (1-1-1+1-3-1-3)/8 = -8/8 = -1

כלומר A שידר 0, וזה מה שרצינו.

ב. (5 נקודות) הסבר בקצרה מתי אנחנו משתמשים בשיטת שידור CDMA ואיזה בעיה היא באה לפתור.

<u>תשובה</u>: משתמשים בשיטת CDMA בשכבת הגישה לערוץ שהינה תת-שכבה של שכבת ערוץ הנתונים. הבעיה שהיא באה לפתור הינה בעית הגישה לרשת של מספר תחנות שרוצות לשדר בו"ז מבלי שההודעות יפריעו אחת לשניה, כלומר נוכל לפענח מה כל תחנה שדרה מתוך השידור הכללי.
שיטה זאת נמצאת בשימוש ברשתות wireless בהם השידור הוא אנלוגי וניתן לסכום בצורה לינארית מספר אותות ששודרו לאות שיהנו צירוף לינארי של האותות. שלא כמו ברשת LAN בה אם שתי תחנות ישדרו ביחד יווצר collision ולא נדע מה שודר.

ג. (3 נקודות) לגבי סעיף א', האם ניתן לדעת כמה ביטים אנחנו נשדר בפועל עבור כל ביט יחיד שברצוננו לשלוח?

<u>תשובה:</u> עבור כל ביט יחיד שברצוננו לשדר בהקדמה עלינו לשדר מספר ביטים כאורך ה- chip sequence ובדוגמא שלנו 8 ביטים.

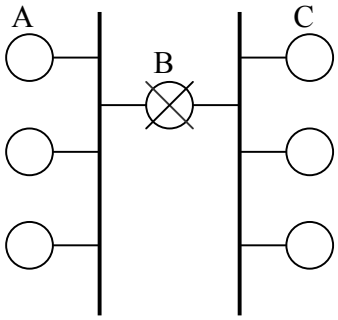
2. א. (5 נקודות) הסבר בקצרה מהו התפקיד של פרוטוקול ARP.

<u>תשובה:</u> תפקיד פרוטוקול ARP הינו למפות בין כתובות IP לכתובות MAC וזאת ע"מ שנוכל להעביר הודעות בתוך LAN כאשר ידועה לנו כתובת IP בלבד ואנו צריכים לתרגמה לכתובת MAC.

ב. (3 נקודות) באיזה שכבה / שכבות במודל השכבות הוא פועל ?

<u>תשובה:</u> הפרוטוקול פועל בשכבת ערוץ הנתונים משום שמשתמש בכתובות MAC וב- broadcast ובכל מקרה לא ניתן להריץ אותו שלא בתוך LAN. אולם הוא מחייב גם היכרות של כתובות IP שהינה כתובת של שכבת הרשת.

ג. (25 נקודות) נתונה הרשת הבאה:



A ו- C הינם מחשבים, A שייך ל- LAN 1
ו- C שייך ל-LAN 2 . B הינו router המחבר בין שני ה- LANS.
A שולח IP packet ל- C כאשר הוא יודע את כתובת IP של C. תאר את קביעת המסלול של ה-IP packet מ-A ועד ל- C תוך כדי שימוש בפרוטוקול ARP בהנחה שטבלאות ה- ARP של כל התחנות ריקות בהתחלה.
רמז: ניתן להשתמש בעובדה ש- A יודע את ה- IP של B ברשת LAN 1 .

<u>תשובה:</u>

א. ראשית A בודק בעזרת network mask של IP_C האם C נמצא ב- LAN 1 .
ב. מכיוון ש-C לא נמצאב- LAN 1, A שולח בקשת ARP ב- broadcast ב- LAN 1 כאשר השאלה היא מהו MAC_B שהינו ה- gateway וזאת ע"פ IP_B שידוע לו.
ג. B מקבל את הבקשה ושולח תשובה ישירות ל- A (לא ע"י broadcast) כאשר התשובה מכילה את MAC_B.
ד. A מקבל את התשובה, מעדכן את MAC_B בטבלת ה- ARP שלו , בונה frame שבתוכה החבילה שאותה רוצה לשלוח ל- C ושולח אותה ל- B.
ה. B מקבל את ה- frame מוציא מתוכה את ה- IP packet המיועד ל- C. בעזרת ה- network mask המופעל על IP_C הוא בודק האם קיים ערך לרשת הנ"ל בטבלת הניתוב שלו. מכיוון שקיים ערך כזה הוא מעביר את החבילה ליציאה שלו המחוברת ל-LAN2.
ו. מכיוון ש- B לא יודע מהי כתובת MAC_C הוא ישלח בקשת ARP בדומהלסעיף ב.
ז. C יענה לו וישלח לו את MAC_C.
ח. סוף סוף, B ישלח את החבילה ל- C.

3. א. (5 נקודות) הסבר מה ההבדל בין Flow control לבין Congestion control.

<u>תשובה:</u> flow control – בקרה של הזרימה (מהירות השליחה בקו) בין שתי תחנות הקצה. (למשל מונע משולח מהיר "להציף" מקבל איטי.) מתבצע ברמת ה- transport במודל השכבות.

congestion control – בקרה של עומס על <u>המסלול</u> בין שתי תחנות הכוללת את תחנות הקצה וכן את כל תחנות הביניים במסלול. (למשל בקרה של עומס לאורך המסלול בנקודה כלשהיא) מתבצע הן ברמת ה- transport וכן ברמת ה- network.

ב. (18 נקודות) הסבר כיצד מתבצע TCP flow control.

<u>תשובה</u>: בפרוטוקול TCP, המקבל מאותת לשולח מהי כמות המידע שהוא מוכן לקבל בכל רגע נתון ע"י שדה בן 16 ביט ב- TCP header הנקרא receiver window size. כאשר לשלוח אסור לשלוח כמות שתעלה על גודל החלון המקבל שפורסם, לפני קבלת ACK על המידע.
המקבל עצמו מקצה buffer הנקרא RcvBuffer בעל גודל נתון וקבוע מראש ע"י מערכת ההפעלה, בו הוא שומר את מידע שהתקבל לפני שהועלה לרמת האפליקציה. עליו הוא מחזיק שני מצביעים:
lastByteRead – זהו מצביע על הבית האחרון שהועלה לרמת האפליקציה, וכן lastByteRcvd זהו מצביע על הבית האחרון שהתקבל מהרשת. במשתנה RcvWinSize הוא מחזיק את כמות המידע אותו הוא מוכן לקבל. החישוב מתבצע בצורה הבאה:

RcvWinSize = RcvBufSize - [lastByteRcvd - lastByteread]

כלומר, כמה מקום פנוי ב- RcvBuffer.
בכל שליחה של TCP packet המקבל שולח את גודל חלון המקבל בשדה המתאים ב- header.
השולח מצידו, מחזיק buffer דומה הנקרא sendBuffer בו הוא מחזיק את המידע שנשלח ועדיין לא נקבל עליו ACK. בצורה דומה ישנם שני מצביעים: lastByteSent מצביע אל הבית האחרון שנשלח. LastByteAcked מצביע על הבית האחרון שעליו קיבלנו אישור. המקום הפנוי ב- SendBuffer מחושב בצורה דומה:

SendWinSize = SendBuffer - [lastByteSend - lastByteAcked]

בכל רגע נתון, גודל חלון השולח חייב להיות קטן מגודל חלון המקבל:
SendWinSize <= RecvWinSize.

ג. (10 נקודות) הסבר כיצד השולח בפרוטוקול TCP מחשב מהי כמות המידע שמותר לו לשלוח ברגע נתון, בהנחה שברצונו לשלוח כמות גדולה של מידע ככל הנדרש. (רמז: צריך לפרט מספר אילוצים החלים על השולח).

<u>תשובה</u>: קיימים שני אילוצים עיקריים:
א. flow control. כפי שתואר בסעיף הקודם.
ב. congestion control. ישנו משתנה נוסף שהינו גודל חלון העומס (congestion windows size) שנסמנו congWin. בכל רגע נתון חייב להתקיים כי:

SendWinSize <= min{congWin, RcvWinSize}

4. א. (10 נקודות) מהו ההבדל בפרוטוקול HTTP בין persistent connection ל- non-persistent connection?

<u>תשובה</u>: כולם ענו נכון ולכן לא מצאתי צורך לפרט את הפיתרון.

ב. (15 נקודות) הסבר מהו cookie מדוע צריך אותו ואיך משתמשים בו.

<u>תשובה</u>: מכיוון ששרת HTTP הינו stateless, כלומר אינו שומר מידע או מצב לגבי תקשורת המשתמש (וזאת ע"פ ההגדרה שלו) לא ניתן לאחסן בו מידע על המשתמש. מכיוון שהתעורר הצורך לשמור מידע על המשתמש (למשל כאשר רוצים לזהותו) אנו חייבים לשמור את המידע בצד הלקוח. צורה כזאת של שמירה, כאשר השרת שומר מידע אצל הלקוח נקראת cookie.

המימוש: פרוטוקול HTTP מגדיר שני שדות המשמשים לכך: כאשר שרת HTTP מעוניין לשמור מידע על הלקוח הוא יצרף את השדה ב- HTTP reply שנקרא set-cookie. בתוכו יהיה התוכן שהוא מעוניין לשמור. דפדפן ה- web כאשר הוא נתקל בשדה הזה, יוצר קובץ בתיקיה המיועדת לכך ובו

הוא מאחסן את המידע. בפעם הבאה שהלקוח יגלוש לאתר אשר שמר cookie, הדפדפן אחראי לקרוא את הקובץ המתאים לאתר המסוים הנ״ל, ולשלוח את תוכנו בתוך השדה cookie ב- HTTP request. כך השרת יכול לקבל מידע על הלקוח.

מכיוון שלא תמיד נרצה שלשרתים יהיה מידע עלינו המשתמשים, ניתן לשנות את הגדרות הדפדפן ע״מ לחסום את השימוש באופציה הנ״ל. חסרון נוסף הינו העובדה שאם משתמש יעבור בין מחשבים המידע עליו לא עובר איתו, או במקרה שיש משתמשים שונים על אותו מחשב.

ג. (8 נקודות) הסברן כיצד אנו מבצעים HTTP caching בצד ה- client ובצד ה- server.

<u>תשובה:</u>

בצד ה- client אנו משתמשים ב- conditional GET. הלקוח שומר אצלו ב- cache מקומי את הדפים האחרונים שנצפו. באחריות השרת לשלוח HTTP reply המכילות שדה בשם lastUpdate שבעבור כל דף אומר מתי הדף שונה בפעם האחרונה. עבור כל דך שנשמר ב- cache אצל הלקוח, הדפדפן זוכר את תאריך העדכון האחרון של הדף. כאשר הדפדפן מבקש דף מהשרת , במידה ודף זה קיים ב- cache, הוא שולח בקשת GET , אשר הוא מצרף אליה את תאריך הדף האחרון שיש ברשותו. זאת נעשה בעזרת שדה If-modified-since ב- HTTP request header. כאשר השרת מקבל את הבקשה, הוא בודק האם הדף שברשותו עודכן אחרי התאריך שציין הלקוח. במידה ולא, יחזיר לו תשובה האומרת שהדף לא עודכן, והמשתמש יקבל את הדף מתוך ה- cache המקומי. במידה והדף עודן השרת ישלח את הדף מחדש, והלקוח יעדכן אותו מחדש ב- cache.

בצד השרת, אנו מוסיפים שרת HTTP הנקרא proxy-server. כל תקשורת ה- HTTP של הלקוח תעשה <u>אך ורק</u> דרכו. השרת מכיל cache של כל הדפים האחרונים שנצפו ע״י הלקוחות. במקרה שהדף המבוקש נמצא ב- cache של השרת, הוא יחזיר אותו מייד ללקוח. אחרת, ה- proxy-server יפנה לשרת המתאים המכיל את הדף, יקבל ממנו את התשובה, ישמור אותה ב- cache ויחזיר את הדף ללקוח.