Homework 3 Dry

|  |  |  |
| --- | --- | --- |
| **ei@campus.technion.ac.il** | 209462415 | יניב איכלר |
| **graif.roey@campus.technion.ac.il** | 315111401 | רואי גרייף |

# **שאלה 1 - Networking - תקשורת (52 נק')**

**חלק פתוח:** הסבירו בקצרה (לא יותר מ1-2 שורות!)

**לאחר מסע בחופי ישראל, יוליה כלבת הים הנזירית** שמעה על מסיבת הקיץ של הפקולטה למדעי המחשב, ה-"טאוביץ", והחליטה לקפוץ לביקור. כשהגיעה לבניין טאוב, הבינה יוליה כי לא תמצא פה חוף ים וחיפשה דרכים לצנן את גופה. יוליה החליטה להזמין את קראנץ הפיסטוק שכולם מדברים עליו וניסתה להזמין לעצמה אחד, אך אבוי, יוליה היא כלבת ים ולא עברה עדיין את הקורס מערכות הפעלה ולכן לא יודעת כיצד עובדת רשת האינטרנט, עזרו ליוליה להבין מושגים בסיסיים בעולם הרשתות על מנת שתוכל להזמין לעצמה קראנץ פיסטוק.

**א. (4 נקודות)** הסבירו מה תפקיד של פרוטוקול ARP.

כאשר אנחנו פונים לכתובת IP של LAN אחר, יש בLAN זה ברוב המקרים יותר מdevice אחת המחוברים לרשת הפנימית ולכן יש צורך להבדיל בינהם. עם זאת, עלינו לדעת לאיזה מכשיר עלינו לפנות וזהו תפקיד הARP להחזיר למי שפנה את רשימת הdevices הקיימים ברשת. הARP מחזיר כתובת MAC.

**ב. (4 נקודות)** איזה מידע הלקוח צריך לדעת על השרת לפני ההתחברות?

הלקוח צריך לדעת מהו הIP של השרת, הport שעליו להתחבר.

**ג. (4 נקודות)** איזה מידע הלקוח ידע על השרת אחרי ההתחברות?

שום מידע נוסף. השרת שולח את התגובה שלו והלקוח מקבל רק את התגובה.

**ד. (4 נקודות)** איזה מידע השרת צריך לדעת על הלקוח לפני ההתחברות?

שום מידע נוסף. כי עוד לא בוצע חיבור בינהם.

**ה. (4 נקודות)** איזה מידע השרת ידע על הלקוח אחרי ההתחברות?

השרת ידע את כתובת הIP של הלקוח, את הport של הלקוח ואת הפרוטוקול תקשורת שהוחלט בינהם.

**ו. (6 נקודות)** מה הבדל בין הפורט (port) שבשימוש השרת וזה של הלקוח. אין נבחר כל אחד מהם?

לשרת צריך שיהיה port מוכר על מנת שהclient יוכל לדעת לאן לפנות מבלי להשתמש בפרוטוקול ARP ולכן נהוג להשתמש בפורטים של שרת ידועים כגון פורט מספר 80 לhttp ו443 לhttps. בclient זה המערכת הפעלה שמקצה מתוך הטווח הלא שמור. והport של הclient יכול להיות זמני ואין חשיבות למספר עצמו בניגוד לשרת.

**ז. (6 נקודות)** מה הבדל בין פרוטוקול TCP ו-UDP? הסבירו למה חלק מהאפליקציות מעדיפות TCP וחלק UDP.

TCP- נותן למשתמש ביטחון שהמידע המועבר הוא נכון ובסדר הנכון כלומר אין איבוד מידע. משתמשים בו לרוב בהעברת מידע שבו אסור שיהיה איבוד או שינוי של מידע זה כגון העברת כסף בביט או שליחת מייל.

UDP- יש לUDP יתרון אחד משמעותי על TCP וזהו מהירות. כלומר לUDP מותר לאבד מידע אבל העיקר שהמידע יגיע מהר. נרצה להשתמש בUDP בשיחות זום או במשחקי מחשב online עם הרבה משתמשים שמשחקים יחדיו. החסרונות הם שהזום לא יהיה הכי חד ולפעמים אנחנו נראה קפיצות קטנות שלרוב לא נשים לב אליהם.

**ח. (6 נקודות)** מהו תפקיד פרוטוקול ה- DNS?

א.       לשלוח פקטות  (frame) מחשבי קצה בתוך אותה רשת (LAN connectivity).

ב.       לתרגם כתובת IP לכתובת MAC.

ג. לתרגם שם השרת לכתובת IP.

ד.       לתרגם שם השרת לכתובת MAC.

ה.      לשלוח פקטות בין מחשבי קצה ברשתות שונות (WAN).

ו.         לאפשר תקשורת בין שני תהליכים במחשבי קצה ברשתות שונות (WAN).

**נימוק**:

בני אדם לא זוכרים את כתובות הIP בעל פה אבל כן זוכרים שמות של אתרים ולכן התפקיד של הDNS הוא לתרגם את שם השרת (“שקל” לזכור) לכתובת הIP (“שקשה” לזכור).

**ט. (8 נקודות)** מהו תפקיד פרוטוקול ה- NAT?

א.       וידוי של הצפנת המידע.

ב. שימוש של מספר קטן של כתובות IP עבור הרבה מכשירים בתוך הרשת.

ג.        הסתרת זהות הלקוח.

ד.       הסתרת זהות השרת.

ה.      וידוי של הצפנת המידע + שימוש של מספר קטן של כתובות IP עבור הרבה מכשירים בתוך הרשת.

ו.         וידוי של הצפנת המידע + הסתרת זהות הלקוח.

**נימוק**:

עקב מחסור עולמי בכתובות IP, לא ניתן לתת לכל מכשיר כתובת IP משלו. הNAT מאפשר לכמה מכשירים על אותה רשת לגשת לאינטרנט עם אותו IP. בנוסף, נותן אבטחה בכך שהשרת לא יודע איזה מכשיר בדיוק פנה אליו.

**י. (6 נקודות)** מה נכון במודל תקשורת שרת/לקוח על מנת ליצור connection (חיבור)?

א.       הלקוח חייב לדעת גם שם של ה-domain של השרת וגם מספר הפורט של השרת.

ב.       שרת חייב לדעת כתובת IP של הלקוח, אך הלקוח לא חייב לדעת כתובת IP של השרת.

ג.        שרת חייב לדעת כתובת IP של הלקוח, וגם הלקוח חייב לדעת כתובת IP של השרת.

ד.       השרת חייב לדעת גם כתובת IP וגם מספר הפורט של הלקוח.

ה.      הלקוח חייב לדעת כתובת שם של ה-domain של השרת. הפורט הינו קבוע לפי סוג ה-application

ו.         המידע הנחוץ תלוי בצד שיוזם את החיבור.

**נימוק**:

שאלה 2 - סינכרון (48 נק')

לאחר הפרידה המתוקשרת של נוגה(מוכרת בעיקר על ידי השיר שלה, "חד קרנל") ומרבי, עולם הפופ הישראלי התחלק לשתי קבוצות, קבוצת נוגה וקבוצת מרבי. בין הקבוצות שררה שנאה רבה ולא היו מוכנים לשהות באותו החדר, ולכן הוגדר כי כאשר חבר אחת הקבוצות רוצה להיכנס לחדר מסויים עליו לציית לכלל הבא:אם יש חברי קבוצה אחרת בחדר אזי אסור לו להיכנס ועליו להמתין עד שיעזבו (לעומת זאת, מספר חברים מאותה הקבוצה יכולים לשהות בחדר באותו הזמן).

סמני נכון / לא נכון (אין צורך להסביר):

1. **(3 נק')** יכולים להיות שני חברים מקבוצות שונות באותו חדר במקביל: **נכון / לא נכון**
2. **(3 נק')** יכולים להיות שני חברים מאותה הקבוצה בחדר במקביל: **נכון / לא נכון**
3. **(3 נק')** חברי קבוצה אחת עלולים להרעיב (כניסת) חברי קבוצה אחרת: **נכון / לא נכון**

בסעיפים הבאים מוצג קוד למימוש כניסה ויציאה של חברים בקבוצות השונות אל ומחדר מסוים, כאשר נתון כי:

* כל חוט מייצג חבר קבוצה כלשהי.
* בכניסה לחדר חבר הקבוצה קורא ל (onArrival(int team, שמקבלת את הקבוצה אליה שייך.
* ביציאה מהחדר חבר הקבוצה קורא ל (onLeave(int team שמקבלת את הקבוצה אליה שייך.
* הערכים 0 ו-1 של team מייצגים את קבוצת מרבי וקבוצת נוגה, בהתאמה.
* (הניחו שאמצעי הסנכרון עברו אתחול תקין והתעלמו מבעיות קומפילציה אם ישנן, שכן מטרת השאלה אינה לבדוק שגיאות אתחול/תחביר).

|  |  |  |  |
| --- | --- | --- | --- |
| 1.  2.  3.  4.  5.  6.  7.  8.  9.  10. | #include <pthread.h>  int members = 0;  mutex\_t global;  void **onLeave**(int team) {  mutex\_lock(&global);  members --;  mutex\_unlock(&global);  } | 11.  12.  13.  14.  15.  16.  17.  18.  19.  20. | void **onArrival**(int team) {  mutex\_lock(&global);  while (members > 0) {  mutex\_unlock(&global);  sleep(10);  mutex\_lock(&global);  }  mutex\_unlock(&global);  members++;  } |

1. **(12 נק')** בהתייחס לקוד הנ״ל, הקיפי אתכל התשובות הנכונות (עשויה להיות יותר מאחת).  
   עבור כל תשובה שהקפת, תארי דוגמת הרצההמובילה לתשובה זו.

1. קיימת בעיית נכונות עקב race condition למשאבים משותפים.

שחקן מקבוצה 1 נכנס לחדר ובין שורה 18 ל19 יש החלפת הקשר ואת השחקן מקבוצה 0 נכנס לחדר ושניהם ביחד בחדר.

1. קיימת בעיית DeadLock / Livelock בקוד.

חבר קבוצה 1 נמצא בחדר ולא עוזב לעולם אז חבר קבוצה אחר שירצה להיכנס יתקע בתוך הwhile לנצח ולא יוכל לצאת עד שהראשון יצא. אז members לא יוכל להיות מספר גדול יותר מ1.

1. הקוד משתמש ב-Busy Wait שפוגע בנצילות המעבד.

כשמישהו רוצה להיכנס ויש אנשים בחדר אז הוא עושה while במקום לעשות wait. למרות שיש sleep העובדה שיש while על מספר האנשים בחדר מהווה busy wait.

1. הקוד מפר את כלל הכניסה לחדר (שהוגדר בתחילת השאלה).

לפי הקוד אי אפשר להכניס שני אנשים מאותה קבוצה לחדר.

וגם הקוד מפר את החוק ששני אנשים מקבוצות שונות לא יכולים להיות ביחד באותו חדר.

המימוש של כניסה ויציאה שונה כך שישתמש במשתני תנאי:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18 | int members[2] = {0}; // 2 counters  cond\_t conds[2]; // 2 condition variables  mutex\_t global;  void **onArrival**(int team) {  mutex\_lock(&global);  int other = team? 0 : 1;  while(members[other] > 0)  cond\_wait(&conds[team] , &global);  members [team]++;  mutex\_unlock(&global);  }  void **onLeave**(int team) {  mutex\_lock(&global);  members [team]--;  int other = team? 0 : 1;  cond\_signal(&conds[other]);  mutex\_unlock(&global);  } |

אך עומר (עתודאי במדמ"ח) טען שקוד זה גורם לחוטים להתעורר שלא לצורך ומיד לחזור למצב המתנה.

1. **(7 נק')** הסבירי את טענתו של עומר באמצעות דוגמת ריצה קונקרטית.

**עומר צודק, כי נניח ויש 5 אנשים מאותה קבוצה בקגע שאחד מהם יעזוב הוא ישלח cond\_signal ואז אדם אחר מקבוצה אחרת שמחכה להיכנס יתעורר ויראה שעדיין יש חברי קבוצה אחרים בחדר ויחזור לישון.**

1. **(8 נק')** כיצד ניתן לתקן את הבעיה שהציג עומר בסעיף הקודם?

**צריך לעשות שרק האדם האחרון שיוצא מהחדר ישלח את הסיגנל ולכן צריך להוסיף בשורה בין 15 ל16: if(members[team] == 0)**

עומר ניסה לשפר עוד את יעילות הקוד והחליט להשתמש בשני מנעולים: מנעול ראשון בעבור חברי קבוצה הנכנסים לחדר, ומנעול שני בעבור חברי קבוצה היוצאים מהחדר. להלן המימוש החדש (השינויים בקוד מודגשים):

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20 | int members [2] = {0}; // 2 counters  cond\_t conds[2]; // 2 condition variables  **mutex\_t m\_arrival, m\_leave;** // there are \*2\* locks now  void onArrival(int team){  mutex\_lock**(&m\_arrival);**  int other = team? 0 : 1;  while(members [other] > 0)  cond\_wait(&conds[team] , **&m\_arrival**);  **int tmp = members [team];**  **members [team] = tmp + 1;**  mutex\_unlock**(&m\_arrival);**  }  void onLeave(int team){  mutex\_lock**(&m\_leave);**  **int tmp = members[team];**  **members [team] = tmp – 1;**  int other = team? 0 : 1;  cond\_signal(&conds[other]);  mutex\_unlock**(&m\_leave);**  } |

1. **(12 נק')** בהתייחס לקוד הנ״ל, הקיפי את כל התשובות הנכונות (עשויה להיות יותר מאחת).  
   עבור כל תשובה שהקפת, תארי דוגמת הרצה המובילה לתשובה זו.
2. יתכנו 2 חברים מקבוצות שונות בתוך החדר ביחד, עקב race condition למשאב משותף.
3. יתכן מצב שחבר קבוצה כלשהי לא נכנס לחדר למרות כלל הכניסה שמתיר זאת, עקב race condition למשאב משותף.
4. קיימת בעיית DeadLock / Livelock בקוד.
5. יתכן מצב שחבר קבוצה כלשהי יחכה למרות כלל הכניסה שמתיר זאת, כאשר אין race condition למשאב משותף.