

Homework 3 Dry

סטודנטים מגישים:

אבו מוך ראמי – 324276765

ramiab@campus.technion.ac.il

عياشي محمد – 213014061

muhammeda@campus.technion.ac.il

שאלה 1 - Networking - תקשורת (52 נק')

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

ה��פקיד של ARP היא למצוא את ה MAC Adress של מכונת היעד אם יודעים רק את ה IP Adress שלו, אם התקן היעד אינו ברשות המקומית אז מוחזירים את ה MAC Adress של ה first-hop router .

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

את ה IP Adress או host name שנitin לתרגם אותו לכתובת IP באמצעות DNS) ואת ה Port של השרת .

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

את ה IP Adress ו את ה Port של השרת .

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

כלום, השרת ידע מי שלוח לו אחרי ההתחברות ולא לפני .

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

את ה IP Adress ו את ה Port של הלקוח .

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

ההבדל ביניהם הוא שהפורט (port) של השרת קבוע וידוע מראש כך שהשרת מודין לקשרים הנכנסות לעומת פорт (port) הלקוח שנבחר בצורה אקראית על ידי מערכת הפעלה .

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

פרוטוקול UDP פשוט יותר, מהיר יותר אבל לא מבטיח שכל המידע יגיע באופן תקין ומסודר לעומת פרוטוקול TCP שմבטיח ודואג שכל הפאקטוט מגיעות באופן תקין .
ולכן אפליקציה שאכפת לה מאובדן מידע تعدיף TCP .
ואפליקציה שמעדיפה רק מהירות עדיף UDP .

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

- . לשולח פקודות (frame) מחשבי קצה בתוך אותה רשת (LAN connectivity).
- . לתרגם כתובת IP לכתובת MAC.
- . לתרגם שם השרת לכתובת IP.
- . לתרגם שם השרת לכתובת MAC.
- . לשולח פקודות בין מחשבי קצה ברשתות שונות (WAN).
- . לאפשר תקשורת בין שני תהליים במחשבי קצה ברשתות שונות (WAN).

ג'יומוק:

למען סיפוק פשטות למשתמש ה DNS מתרגם את שם השרת לכתובת IP .

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

- א. יידי 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. יידי של הצפנת המידע + שימוש של מספר קטן של כתובות IP עבור הרבה מכשירים בתוך הרשת.
- ו. יידי של הצפנת המידע + הסתרת זהות הלקוח.

ג'יומוק:

פרוטוקול NAT מספק פתרון לניהול ואופטימיזציה לכתובות IP של כל המכשירים שמחוברים לאותה רשת מקומית.

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

- א. הלוקוח חייב לדעת גם שם של ה-domain של השרת וגם מספר הפורט של השרת.
- ב. שרת חייב לדעת כתובת IP של הלוקוח, אך הלוקוח לא חייב לדעת כתובת IP של השרת.
- ג. שרת חייב לדעת כתובת IP של הלוקוח, וגם הלוקוח חייב לדעת כתובת IP של השרת.
- ד. השרת חייב לדעת גם כתובת IP וגם מספר הפורט של הלוקוח.
- ה. הלוקוח חייב לדעת כתובת שם של ה-domain של השרת. הפורט הינו קבוע לפי סוג ה-application-layer.
- ו. המידע הנוחז תלוי הצד שיוזם את החיבור.

ג'יומוק:

הלוקוח צריך לדעת את name domain שניתן לתרגם אותו לכתובת IP באמצעות DNS ואת ה Port של השרת , לעומת השרת הוא לא חייב לדעת כלום לפני החיבור.

שאלה 2 - סינכרון (8 נק')

לאחר הריב המתווך של נוגה (מוכרת בעיקר על ידי השיר שלו, "חץ קרנלי") ודנה (המוכרת בעיקר על ידי השיר שלו, "צא להמתנה"), עולם הפופ הישראלי החלק לשתי קבוצות, קבוצת נוגה וקבוצת דנה. בין הקבוצות שררה שנאה רבה ולא היו מוכנים לשחות באותו החדר, וכך הוגדר כי כאשר חבר אחד הקבוצה רצה להיכנס לחדר מסוים עליו לצאת לפחות הבא: אם יש חברי קבוצה אחרת בחדר אז אסור לו להיכנס עד שייעזבו (לעומת זאת, מספר חברים מסוימת הקבוצה יכולים לשחות בחדר באותו הזמן).

סמי 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 מאותו הקבוצה בחדר במקביל: נכון / לא נכון
3. (3 נק') חברי קבוצה אחת עלולים להרעיב (כנית) 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 בלבד.
- בכניסה לחדר חבר הקבוצה קורא ל `(int team)Arrival`, שמקבלת את הקבוצה אליה שייר.
- ביציאה מהחדר חבר הקבוצה קורא ל `(int team)onLeave` שמקבלת את הקבוצה אליה שייר.
- הערך 0 ו-1 של `team` מייצגים את קבוצת דנה וקבוצת נוגה, בהתאם.
- (הניחו שאמצעי הסyncron עברו אתחול תקין והتعلמו מביעות קומפיילציה אם ישן, שכן מטרת השאלה אינה לבדוק שגיואט אתחול/תחבר).

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

1. (12 נק') בהתייחס לקוד הנ"ל, הקופי את כל התשובות הנכונות (עשוייה להיות יותר מאתחת).
 עברו כל תשובה שהקפת, תאריך דוגמת הריצה המובילת לתשובה זו.

- a. קיימת בעיית נוכנות עקב race condition למשאים משותפים.
- b. קיימת בעיית Livelock / DeadLock בקוד.
- c. הקוד משתמש忙-waitBusy שפוגע בניצול המעבד.
- d. הקוד מפרק את כל הכניסה לחדר (שהוגדר בתחילת השאלה).

nymok:

- c – במקורה שיש מעבד יחיד במחשב, אם יש שני חוטים כך שהראשון נכנס לפונקציה OnArrival ומצילח, ואז החט השני נכנס לאותה פונקציה ובפעם הראשונה שנכנס לוולאה ועושה Sleep ומיד קורת הצלפת הקשר לחוט הראשון כשמבוצע OnLeave אז החוט השני יצטרך להמתין את כל עשר השניות בכוח עד שיוכל להמשיך את פונקציית OnArrival ולכן זה יוכל זמן מעבד מיותר ולכן זה לא יכול waiting busy .
- d – נניח כי חוט ראנדון מקבוצה מס' 0 נכנס לפונקציה (0)OnArrival וסימן בהצלחה כלומר נכנס לחדר, ואז חוט שני מקבוצה מס' 0 נכנס גם לפונקציה (0)OnArrival מוביל שייצא החוט הראשון מהחדר, אז הוא נכנס ללולאה נוספת, וישן 10 שניות וחודר ללולאה עוד פעם וכך ואז לא יוכל 2 חברים מקבוצה להיכנס לחדר באותו זמן.

המימוש של כניסה ויציאה שונה בכך שימוש במשתני תנאי:

```

1 int members[2] = {0}; // 2 counters
2 cond_tconds[2]; // 2 condition variables
3 mutex_t global;
4 void onArrival(int team) {
5     mutex_lock(&global);
6     int other = team? 0 : 1;
7     while(members[other] > 0)
8         cond_wait(&conds[team], &global);
9     members [team]++;
10    mutex_unlock(&global);
11 }
12 void onLeave(int team) {
13     mutex_lock(&global);
14     members [team]--;
15     int other = team? 0 : 1;
16     cond_broadcast(&conds[other]);
17     mutex_unlock(&global);
18 }
```

אר נדב (יושב ראש מדמ"ח) טען שקוד זה גורם לחוטים להתעורר שלא לצורך ומיד לחזור למצב המתנה.
1. (7 נק') הסבירו את טענתו של נדב באמצעות דוגמת ריצה קונקרטית.

אם מבצעים (1) 5 פעמים אחת אחרי השנייה, ואז מנסים לבצע (0)OnArrival אז הוא יתקע בתוך ה- while (הוא ייכה לחדר להתרוקן) , בכל פעם שמשיחו מבצע (1)OnLeave(0) הוא ישולח סיגנל להעיר את כל החוטים שביצעו (0)OnArrival שזה במקורה שלנו חוט אחד, אבל החדר עדין יש בו חוטים מקבוצה 1 ולכן הוא יתעורר 5 פעמים (ויחזר לחכות ב 4 מהם) , למרות שהוא צריך להתעורר פעם אחת.

1. (8 נק') כיצד ניתן לתקן את הבעה שהציג נדב בסעיף הקוד?

```

if(members[team] == 0){
    cond_broadcast(&conds[other]);
}
```

במקום שורה 16 בקוד הנ"ל נוסיף את הקוד משמאלו. לפני שמעירם את כל החוטים הממתינים אנו בודקים אם התפינה החדר מחברי הקבוצה האחראית ואז לא גורמים לחוטים להתעורר ללא צורך מיד לחזור למצב המתנה, כי אפשר לנסח לחדר רק אחראי שכל חברי הקבוצה האחראית עזבו ורק אז מעריכים את כל החוטים הממתינים (מכניסים לחדר) מוסיפים את מספרם ומשחררים המניעול.

נדב ניסה לשפר עוד את יעילות הקוד והחליט להשתמש בשני מנעולים: מנעל ראשון בעבר חביי קבוצה המכנים לחדר, ומגעול שני בעבר חביי קבוצה היוצאים מהחדר. להלן המימוש החדש (השינויים בקוד מודגשים):

```

1 int members [2] = {0};           // 2 counters
2 cond_tconds[2];                // 2 condition variables
3 mutex_t m_arrival, m_leave;   // there are *2* locks now
4 void onArrival(int team){
5     mutex_lock(&m_arrival);
6     int other = team? 0 : 1;
7     while(members [other] > 0)
8         cond_wait(&conds[team] , &m_arrival);
9     int tmp = members [team];
10    members [team] = tmp + 1;
11    mutex_unlock(&m_arrival);
12 }
13 void onLeave(int team){
14     mutex_lock(&m_leave);
15     int tmp = members[team];
16     members [team] = tmp - 1;
17     int other = team? 0 : 1;
18     cond_broadcast(&conds[other]);
19     mutex_unlock(&m_leave);
20 }
```

1. (12 נק') בהתייחס לקוד הנ"ל, הקify את כל התשובות הנכונות (עשוייה להיות יותר מ אחת).
 עברו כל תשובה שהקפת, תארו דוגמת הרצה המובילה לתשובה זו.

- a. יתכנו 2 חברים מקבוצות שונות בתוך החדר ביחד, עקב race condition למשאב משותף.
- b. יתכן מצב שחבר קבוצה כלשהו לא נכנס לחדר למשך כל הכניסה שמתיר זאת, עקב race condition למשאב משותף.
- c. קיימת בעיית Livelock / DeadLock בקוד.
- d. סיגנלים עולמים ללכנת לאיבוד.

רימוק:

תרחיש בעיתו ל- a:

מבצעים (1) onArrival(1) וمستיממת בהצלחה.
 מבצעים (1) onLeave(1), ומתבצעת החלפת הקשר אחריו שורה 15 (ערך tmp הוא 1).
 מבצעים (1) onArrival(1), ומסיים ווארך members[1] וmembers[0] יתעדכו להיות 2.
 מבצעים (1) onLeave(1), מאייפה שהפסיקה, ומעדכנת את ערך members[1] ל-0.
 מבצעים (0) onArrival(0) ומכניסה אותו כי members[1] = 0, למורות שיש חבר קבוצה 1 בחדר.

תרחיש בעיתו ל-ב:

מבצעים 2 פעמים `onArrival(1)` שעובדות כמצופה.
מבצעים `onLeave(1)` ומתבצעת החלפת הקשר אחריו שורה 15 (ערך tmp הוא 2)
מבצעים `onLeave(1)` ומתבצעת החלפת הקשר אחריו שורה 15 (ערך tmp הוא 2)
ואז ממשיכים `onArrival(0)` מאיפה שעיצרנו שלא יכנס לעולם כי members[1] ערכו 2, למראות שאין חבר קבוצה 1
בזהר, כתוצאה מה-race condition בעדכן ה-members[1] ב-`onLeave(0)`.

תרחיש בעיתו ל-ב:

מבצעים `onArrival(1)` ומסיימים בהצלחה.
מבצעים `onArrival(0)` ומתבצעת החלפת הקשר אחריו שורה 7 (אחרי בדיקת תנאי ה-while).
מבצעים `onLeave(1)` ומסתיהם בהצלחה.
חוורים `onArrival(0)` מאיפה שעיצרנו ונמשיך להכotta לסיגנל למראות שהוא כבר נשלח ונעלם.