**חקירת פרוטוקול וקוד שרת לקוח בתרגיל 2.6 - עידן רטיק י"א3**

1. בפעולת Main של הserver בתוך נמצאת לולאת while אשר מקבלת לקוחות חדשים.

2. מיד לאחר התחברות השרת משגר Thread שהtargret שלו היא פעולת handle\_client והיא מקבלת סוקט, מס thread וכתובתIp+port. לא ניתן לשגר thread לפני ההתחברות כי אז לא תהיה לפעולה handle\_client את הפרמטרים של הלקוח החדש והיא לא תוכל לטפל בו.

3. הפרמטרים ששולח הmain thread אל הthread הם:

* סוקט של הלקוח, כתובת IP+port: הכרחי על מנת להעביר מידע ולקבל מידע מהשרת.
* מס thread: כדי שהשרת יוכל להציג בטרמינל איזה מספר לקוח שהתחבר שלח או קיבל מידע

4. עד ללולאה יוצר רשימה של threads, יוצר אובייקט סוקט של השרת, מקשר אותו לפורט 1233 וכתובת 0.0.0.0. אחכ באמצעות פעולת listen() גורם לסוקט של השרת להיות מוכן להתחברות של עד 20 לקוחות. לבסוף גורם לכך שברגע שסוקט ייסגר יהיה אפשר להתחבר אל הפורט מחדש מיד.

5. השלבים שעוברת בקשה אצל השרת הם:

1. נסיון לקבל 1000 תווים מהלקוח
2. בדיקת המידע שהתקבל (האם תקין לפי הפרוטוקול – אם לא, תגובה בהתאם)
3. 'חילוץ' הבקשה מההודעה
4. 'חילוץ' הפקודה מהבקשה
5. בניית תגובת שרת בהתאם לפקודה (במקרה של פקודת יציאה – ווידוא יציאה ניתוק. במקרה של פקודה לא חוקית – בניית תגובה בהתאם)
6. שליחת תגובת השרת (לפי הפרוטוקול)

6. פעולות הget מחזירות את המידע עצמו שאותו הלקוח שואף לקבל ללא שום תוספות ושינויים

7. לוגיקה עסקית באמצעותה נקבע כיצד מועבר המידע עצמו בין המחשב לבין המשתמש. במקרה הזה – protocol\_build\_reply קובע כיצד ייראה המידע כאשר הוא מועבר מהשרת ללקוח (לפי הפרוטוקול)

8. שדה האורך מתווסף לכל הודעה בפעולת send\_data, אשר מקבלת את ההודעה להעברה לפני שצורף שדה האורך לפניה. זה לא חלק מהbusiness logic מפ\

9. כן, פעולות הget הן פעולות שתפקידן להביא את המידע עצמו בלי שום דבר נוסף, שאר הפעולות תפקידן לדאוג כיצד המידע יועבר (הצד התקשורתי)

10. השרת עושה השוואה בין האורך של ההודעה פחות 9 (כלומר, האורך של הפקודה) לבין המספר שמייצג שדה האורך. אם הם שונים סימן שישנה שגיאה. אם אורך ההודעה כולה פחות מ13 גם אז ההודעה שגויה מפני שאורך המינימום הוא 13 (8 תווים שדה אורך + ‘~’ + 4 תווים הפקודה = 13)

11. השרת מחליט על קוד הודעה חוזר לפי קוד ההודעה שקיבל מהלקוח (ע"פ הפרוטוקול)

12. הודעתexit . אשר תגרום לשרת לבצע פעולות של עיבוד מידע וגם תגרום לשרת לבצע פעולות שקשורות בתקשרות.

13. אין נעילות למשאב all\_to\_die על מנת שכל thread יכול לשנות אותו.

14. All\_to\_die הופך לtrue כאשר יותר ממספר מסויים (כרגע 100000) של לקוחות התחבר אל השרת סה"כ (כלומר לא מספר הלקוחות שמחוברים באותו רגע אלא סהכ מתחילת הפעילות שלו)

15. Exepct הראשון מטפל בexceptions שנוצרו כתוצאה משגיאה בסוקט (לדוגמה recv כאשר הלקוח איננו מחובר עוד)

Except השני מטפל בבעיות כלליות אחרות כגון פנייה לאינדקס שאינו קיים ברשימה וכו'

16. לולאה זו עוברת על כל הthreads הפעילים ובאמצעות פעולת join() "חוסמת" או עוצרת אותם (עד שייסגרו).

לקוח

1. נראה שיש הפרדה בין הפרוטוקול לבין ממשק המשתמש.

2. כן

3. Logtcp שונה בין השרת ללקוח בכך ש אצל השרת מתקבל פרמטר נוסף מלבד dir וההודעה שהוא מספר הthread של הלקוח שממנו התקבלה או אליו תשלח ההודעה. אצל הלקוח אין צורך בכך מפני שהוא מתחבר לשרת אחד בלבד.

4. רוב הזמן הלקוח לא יודע שהשרת התנתק מפני שהוא מחליף איתו מידע רק כאשר מתקבלת פקודה על ידי המשתמש. הלקוח מגלה שהשרת התנתק ברגע שהוא שולח פקודה ומתקבלת שגיאה על שהחיבור כבר לא קיים.

שרת ולקוח

1. בקוד יש פעם אחת send ופעם אחת recv. זה מהווה יתרון בגלל שעבור כל הודעה שנשלחת בכל צד, מצופה להתקבל הודעה אחת.

תרשים זרימה קוד לקוח

תרשים זרימה קוד שרת main

תרשים זרימה קוד צד שרת

**חקירת פרוטוקול וקוד שרת לקוח בתרגיל 2.6 - עידן רטיק י"א3**

1. בפעולת Main של הserver בתוך נמצאת לולאת while אשר מקבלת לקוחות חדשים.

2. מיד לאחר התחברות השרת משגר Thread שמבצעאת פעולת handle\_client והיא מקבלת סוקט, מס thread וכתובת Ip+port. לא ניתן לשגר thread לפני ההתחברות כי אז לא תהיה לפעולה handle\_client את הפרמטרים של הלקוח החדש והיא לא תוכל לטפל בו.

3. הפרמטרים ששולח הmain thread אל הthread הם:

* סוקט של הלקוח, כתובת IP+port: הכרחי על מנת להעביר מידע ולקבל מידע מהשרת.
* מס thread: כדי שהשרת יוכל להציג בטרמינל איזה מספר לקוח שהתחבר שלח או קיבל מידע

4. עד ללולאה יוצר רשימה של threads, יוצר אובייקט סוקט של השרת, מקשר אותו לפורט 1233 וכתובת 0.0.0.0. אחכ באמצעות פעולת listen() גורם לסוקט של השרת להיות מוכן להתחברות של עד 20 לקוחות. לבסוף גורם לכך שברגע שסוקט ייסגר יהיה אפשר להתחבר אל הפורט מחדש מיד.

5. השלבים שעוברת בקשה אצל השרת הם:

1. נסיון לקבל 1000 תווים מהלקוח
2. בדיקת המידע שהתקבל (האם תקין לפי הפרוטוקול – אם לא, תגובה בהתאם)
3. 'חילוץ' הבקשה מההודעה
4. 'חילוץ' הפקודה מהבקשה
5. בניית תגובת שרת בהתאם לפקודה (במקרה של פקודת יציאה – ווידוא יציאה ניתוק. במקרה של פקודה לא חוקית – בניית תגובה בהתאם)
6. שליחת תגובת השרת (לפי הפרוטוקול)

6. פעולות הget מחזירות את המידע עצמו שאותו הלקוח שואף לקבל ללא שום תוספות ושינויים

7. לוגיקה עסקית באמצעותה נקבע כיצד מועבר המידע עצמו בין המחשב לבין המשתמש. במקרה הזה – protocol\_build\_reply קובע כיצד ייראה המידע כאשר הוא מועבר מהשרת ללקוח (לפי הפרוטוקול)

8. שדה האורך מתווסף לכל הודעה בפעולת send\_data, אשר מקבלת את ההודעה להעברה לפני שצורף שדה האורך לפניה. זה לא חלק מהbusiness logic מפ\

9. כן, פעולות הget הן פעולות שתפקידן להביא את המידע עצמו בלי שום דבר נוסף, שאר הפעולות תפקידן לדאוג כיצד המידע יועבר (הצד התקשורתי)

10. השרת עושה השוואה בין האורך של ההודעה פחות 9 (כלומר, האורך של הפקודה) לבין המספר שמייצג שדה האורך. אם הם שונים סימן שישנה שגיאה. אם אורך ההודעה כולה פחות מ13 גם אז ההודעה שגויה מפני שאורך המינימום הוא 13 (8 תווים שדה אורך + ‘~’ + 4 תווים הפקודה = 13)

11. השרת מחליט על קוד הודעה חוזר לפי קוד ההודעה שקיבל מהלקוח (ע"פ הפרוטוקול)

12. הודעתexit . אשר תגרום לשרת לבצע פעולות של עיבוד מידע וגם תגרום לשרת לבצע פעולות שקשורות בתקשרות.

13. אין נעילות למשאב all\_to\_die על מנת שכל thread יכול לשנות אותו.

14. All\_to\_die הופך לtrue כאשר יותר ממספר מסויים (כרגע 100000) של לקוחות התחבר אל השרת סה"כ (כלומר לא מספר הלקוחות שמחוברים באותו רגע אלא סהכ מתחילת הפעילות שלו)

15. Exepct הראשון מטפל בexceptions שנוצרו כתוצאה משגיאה בסוקט (לדוגמה recv כאשר הלקוח איננו מחובר עוד)

Except השני מטפל בבעיות כלליות אחרות כגון פנייה לאינדקס שאינו קיים ברשימה וכו'

16. לולאה זו עוברת על כל הthreads הפעילים ובאמצעות פעולת join() "חוסמת" או עוצרת אותם (עד שייסגרו).

לקוח

1. נראה שיש הפרדה בין הפרוטוקול לבין ממשק המשתמש.

2. כן

3. Logtcp שונה בין השרת ללקוח בכך ש אצל השרת מתקבל פרמטר נוסף מלבד dir וההודעה שהוא מספר הthread של הלקוח שממנו התקבלה או אליו תשלח ההודעה. אצל הלקוח אין צורך בכך מפני שהוא מתחבר לשרת אחד בלבד.

4. רוב הזמן הלקוח לא יודע שהשרת התנתק מפני שהוא מחליף איתו מידע רק כאשר מתקבלת פקודה על ידי המשתמש. הלקוח מגלה שהשרת התנתק ברגע שהוא שולח פקודה ומתקבלת שגיאה על שהחיבור כבר לא קיים.

שרת ולקוח

1. בקוד יש פעם אחת send ופעם אחת recv. זה מהווה יתרון בגלל שעבור כל הודעה שנשלחת בכל צד, מצופה להתקבל הודעה אחת.

תרשים זרימה קוד לקוח

תרשים זרימה קוד שרת main

תרשים זרימה קוד צד שרת

**חקירת פרוטוקול וקוד שרת לקוח בתרגיל 2.6 - אופיר הופמן י"א3**

1. בפעולת Main של הserver בתוך נמצאת לולאת while אשר מקבלת לקוחות חדשים.

2. מיד לאחר התחברות השרת משגר Thread שהtargret שלו היא פעולת handle\_client והיא מקבלת סוקט, מס thread וכתובתIp+port. לא ניתן לשגר thread לפני ההתחברות כי אז לא תהיה לפעולה handle\_client את הפרמטרים של הלקוח החדש והיא לא תוכל לטפל בו.

3. הפרמטרים ששולח הmain thread אל הthread הם:

* סוקט של הלקוח, כתובת IP+port: הכרחי על מנת להעביר מידע ולקבל מידע מהשרת.
* מס thread: כדי שהשרת יוכל להציג בטרמינל איזה מספר לקוח שהתחבר שלח או קיבל מידע

4. עד ללולאה יוצר רשימה של threads, יוצר אובייקט סוקט של השרת, מקשר אותו לפורט 1233 וכתובת 0.0.0.0. אחכ באמצעות פעולת listen() גורם לסוקט של השרת להיות מוכן להתחברות של עד 20 לקוחות. לבסוף גורם לכך שברגע שסוקט ייסגר יהיה אפשר להתחבר אל הפורט מחדש מיד.

5. השלבים שעוברת בקשה אצל השרת הם:

1. נסיון לקבל 1000 תווים מהלקוח
2. בדיקת המידע שהתקבל (האם תקין לפי הפרוטוקול – אם לא, תגובה בהתאם)
3. 'חילוץ' הבקשה מההודעה
4. 'חילוץ' הפקודה מהבקשה
5. בניית תגובת שרת בהתאם לפקודה (במקרה של פקודת יציאה – ווידוא יציאה ניתוק. במקרה של פקודה לא חוקית – בניית תגובה בהתאם)
6. שליחת תגובת השרת (לפי הפרוטוקול)

6. פעולות הget מחזירות את המידע עצמו שאותו הלקוח שואף לקבל ללא שום תוספות ושינויים

7. לוגיקה עסקית באמצעותה נקבע כיצד מועבר המידע עצמו בין המחשב לבין המשתמש. במקרה הזה – protocol\_build\_reply קובע כיצד ייראה המידע כאשר הוא מועבר מהשרת ללקוח (לפי הפרוטוקול)

8. שדה האורך מתווסף לכל הודעה בפעולת send\_data, אשר מקבלת את ההודעה להעברה לפני שצורף שדה האורך לפניה. זה לא חלק מהbusiness logic מפ\

9. כן, פעולות הget הן פעולות שתפקידן להביא את המידע עצמו בלי שום דבר נוסף, שאר הפעולות תפקידן לדאוג כיצד המידע יועבר (הצד התקשורתי)

10. השרת עושה השוואה בין האורך של ההודעה פחות 9 (כלומר, האורך של הפקודה) לבין המספר שמייצג שדה האורך. אם הם שונים סימן שישנה שגיאה. אם אורך ההודעה כולה פחות מ13 גם אז ההודעה שגויה מפני שאורך המינימום הוא 13 (8 תווים שדה אורך + ‘~’ + 4 תווים הפקודה = 13)

11. השרת מחליט על קוד הודעה חוזר לפי קוד ההודעה שקיבל מהלקוח (ע"פ הפרוטוקול)

12. הודעתexit . אשר תגרום לשרת לבצע פעולות של עיבוד מידע וגם תגרום לשרת לבצע פעולות שקשורות בתקשרות.

13. אין נעילות למשאב all\_to\_die על מנת שכל thread יכול לשנות אותו.

14. All\_to\_die הופך לtrue כאשר יותר ממספר מסויים (כרגע 100000) של לקוחות התחבר אל השרת סה"כ (כלומר לא מספר הלקוחות שמחוברים באותו רגע אלא סהכ מתחילת הפעילות שלו)

15. Exepct הראשון מטפל בexceptions שנוצרו כתוצאה משגיאה בסוקט (לדוגמה recv כאשר הלקוח איננו מחובר עוד)

Except השני מטפל בבעיות כלליות אחרות כגון פנייה לאינדקס שאינו קיים ברשימה וכו'

16. לולאה זו עוברת על כל הthreads הפעילים ובאמצעות פעולת join() "חוסמת" או עוצרת אותם (עד שייסגרו).

לקוח

1. נראה שיש הפרדה בין הפרוטוקול לבין ממשק המשתמש.

2. כן

3. Logtcp שונה בין השרת ללקוח בכך ש אצל השרת מתקבל פרמטר נוסף מלבד dir וההודעה שהוא מספר הthread של הלקוח שממנו התקבלה או אליו תשלח ההודעה. אצל הלקוח אין צורך בכך מפני שהוא מתחבר לשרת אחד בלבד.

4. רוב הזמן הלקוח לא יודע שהשרת התנתק מפני שהוא מחליף איתו מידע רק כאשר מתקבלת פקודה על ידי המשתמש. הלקוח מגלה שהשרת התנתק ברגע שהוא שולח פקודה ומתקבלת שגיאה על שהחיבור כבר לא קיים.

שרת ולקוח

1. בקוד יש פעם אחת send ופעם אחת recv. זה מהווה יתרון בגלל שעבור כל הודעה שנשלחת בכל צד, מצופה להתקבל הודעה אחת.