רשתות תקשורת

מטלה 3

ת"ז 1: 208762971 ת"ז 2: 206751976

הסבר על מימוש ה- Reliable UDP

המימוש יחסית פשוט. ה-RUDP רוכב על-גבי ה-udp הרגיל ומוסיף לו תכונות של חיבור אמין. המימוש כולל:

1. הקמת וסגירת קשר בצורה מסודרת
2. וידוא שכל מה שנשלח התקבל לצד השני
3. וידוא ע"י המקבל שהמידע לא הושחת בדרך

מנגנון 2 פועל בשיטת StopNWait, כלומר אחרי כל שליחת חבילה השולח יחכה לקבלת חבילת אישור – חבילה עם ACK מופעל – ורק אז ימשיך לחבילה הבאה. לכן מספיק ביט אחד לסימון ה-ACK. מנגנון 3 מתבצע ע"י פונקציית ה-checksum שסופקה בקובץ ה-pdf. לאחר קבלת חבילה עם מידע (payload) המקבל יפעיל את הפונקציה על החבילה ויוודא שהתוצאה זהה לשדה ה-checksum בחבילה שהתקבלה. רק אם התשובה חיובית הוא ישלח חבילת ACK.

ה-Header של ה-RUDP הוא:

```
struct _rudp_header {
    unsigned short checksum;//
    unsigned short length;//
    unsigned char flags;
};
```

זיהוי שהמידע תקין ולא הושחת בדרך

גודל המידע (bytes)

דגל (flags)	משמעות
ACK	החבילה האחרונה שנשלחה התקבלה בהצלחה
SIN	חבילה זו מיועדת לפתיחת קשר חדש (תחילת לחיצת הידיים)
FIN	אחד מהצדדים מעוניין לסגור את החיבור
STOF	חבילה זו היא החבילה הראשונה של הקובץ*
ENOF	חבילה זו היא החבילה האחרונה של הקובץ*

*ה-udp socket שנמצא בבסיס ה-RUDP מוגבל בגודל החבילה שהוא מסוגל לשלוח בפעימה יחידה (65507 bytes). לכן כאשר רוצים לשלוח קובץ גדול צריך לפרקו לחלקים קטנים. על-מנת שנוכל למדוד זמנים כחלק מעבודת המחקר ולדעת כמה זמן לוקח לקובץ בגודל 2MB להגיע נצטרך לדעת מתי התת-קובץ הראשון נשלח ומתי התת-קובץ האחרון הגיע, ולכן קיימים הדגלים STOF(StartOfFile) ו-ENOF(End of File).

- **טיפול באובדן חבילות:**

השתמשנו במנגנון timeout לזיהוי אובדן חבילות:

```
struct timeval sec = {sec_tout, 0};
 if (setsockopt(udp_sock, SOL_SOCKET, SO_RCVTIMEO, (char*) &sec, sizeof(sec)) < 0)
```

כאשר *sec_tout* הוא פרמטר (בשניות) שהפונקציה שבונה את ה-RUDP מקבלת מהמשתמש. אם ה-timeout הופעל והשולח לא קיבל חבילת ACK שמידע הועבר בהצלחה הוא יבצע retransmission על החבילה האחרונה שנשלחה.

- **הקמת וסגירת קשר:**

מומש בצורה דומה לפרוטוקול TCP.

הקמת קשר מתבצעת ע״י שליחת חבילה עם דגל SIN מופעל למקבל (השרת). לאחר מכן המקבל שולח חבילה עם דגלי SIN+ACK, ולבסוף השולח שולח חבילת ACK לאותת שאפשר להתחיל לשלוח מידע.

סגירת הקשר מתבצעת באופן דומה ע״י שליחת חבילה עם דגל FIN מופעל למקבל, המקבל שולח בחזרה חבילה עם דגלי FIN+ACK וסוגר את ה-socket אצלו ולבסוף השולח שולח ACK וסוגר את ה-socket אצלו.

תשובות לחלק 3

1. הפעלנו את איבוד הפקטות באחוזים הרצויים, והשווינו את הזמנים והמהירויות של שליחת 5 קבצים בעזרת אלגוריתם reno לעומת אלגוריתם cubic. מצורף תמונות מסך:

Packet-loss percentage	cubic	reno
0	Run #0 Data: Time = 0.8400ms, Speed: 2380.95MB/s Run #1 Data: Time = 0.8880ms, Speed: 2252.25MB/s Run #2 Data: Time = 0.8940ms, Speed: 2237.14MB/s Run #3 Data: Time = 1.0290ms, Speed: 1943.63MB/s Run #4 Data: Time = 0.7980ms, Speed: 2506.27MB/s Average time: 0.8898 ms Average speed: 2264.05 MB/s	Run #0 Data: Time = 0.9220ms, Speed: 2169.20MB/s Run #1 Data: Time = 0.8820ms, Speed: 2267.57MB/s Run #2 Data: Time = 0.5610ms, Speed: 3565.06MB/s Run #3 Data: Time = 0.9570ms, Speed: 2089.86MB/s Run #4 Data: Time = 0.9530ms, Speed: 2098.64MB/s Average time: 0.8550 ms Average speed: 2438.07 MB/s
2	Run #0 Data: Time = 1.0650ms, Speed: 1877.93MB/s Run #1 Data: Time = 0.9880ms, Speed: 2024.29MB/s Run #2 Data: Time = 0.8290ms, Speed: 2412.55MB/s Run #3 Data: Time = 0.9580ms, Speed: 2087.68MB/s Run #4 Data: Time = 0.9140ms, Speed: 2188.18MB/s Average time: 0.9508 ms Average speed: 2118.13 MB/s	Run #0 Data: Time = 0.9700ms, Speed: 2061.86MB/s Run #1 Data: Time = 0.7990ms, Speed: 2503.13MB/s Run #2 Data: Time = 0.5790ms, Speed: 3454.23MB/s Run #3 Data: Time = 0.8710ms, Speed: 2296.21MB/s Run #4 Data: Time = 0.8140ms, Speed: 2457.00MB/s Average time: 0.8066 ms Average speed: 2554.49 MB/s
5	Run #0 Data: Time = 3.0930ms, Speed: 646.62MB/s Run #1 Data: Time = 0.9960ms, Speed: 2008.03MB/s Run #2 Data: Time = 0.9200ms, Speed: 2173.91MB/s Run #3 Data: Time = 0.8940ms, Speed: 2237.14MB/s Run #4 Data: Time = 0.9200ms, Speed: 2173.91MB/s Average time: 1.3646 ms Average speed: 1847.92 MB/s	Run #0 Data: Time = 1.8300ms, Speed: 1092.90MB/s Run #1 Data: Time = 0.7800ms, Speed: 2564.10MB/s Run #2 Data: Time = 0.8900ms, Speed: 2247.19MB/s Run #3 Data: Time = 0.8490ms, Speed: 2355.71MB/s Run #4 Data: Time = 0.9190ms, Speed: 2176.28MB/s Average time: 1.0536 ms Average speed: 2087.24 MB/s
10	Run #0 Data: Time = 2.9290ms, Speed: 682.83MB/s Run #1 Data: Time = 1.4820ms, Speed: 1349.79MB/s Run #2 Data: Time = 1.9830ms, Speed: 1008.57MB/s Run #3 Data: Time = 2.3970ms, Speed: 834.38MB/s Run #4 Data: Time = 1.7820ms, Speed: 1122.33MB/s Average time: 2.1146 ms Average speed: 999.58 MB/s	Run #0 Data: Time = 2.3950ms, Speed: 835.12MB/s Run #1 Data: Time = 1.4270ms, Speed: 1401.54MB/s Run #2 Data: Time = 2.9100ms, Speed: 687.29MB/s Run #3 Data: Time = 2.8100ms, Speed: 711.74MB/s Run #4 Data: Time = 0.9620ms, Speed: 2079.00MB/s Average time: 2.1008 ms Average speed: 1142.94 MB/s

ניתן לראות שסך הכל כאשר השתמשנו באלגוריתם reno לקח <u>פחות</u> זמן לקובץ להישלח והמהירות הייתה יותר טובה מאשר השימוש באלגוריתם cubic.

כאמור, במצב של איבוד פקטות גבוה (10%) אלגוריתם reno נתן מהירות יותר טובה לשליחת הקובץ.

2. מצורף צילום מסך לביצועי הRUDP:

Packet-loss percentage	
0	File received! File size: 2MB. Time for receiveing: 7.793645ms. Average speed: 256.619335MB/s File received! File size: 2MB. Time for receiveing: 11.680737ms. Average speed: 171.222073MB/s File received! File size: 2MB. Time for receiveing: 13.465848ms. Average speed: 148.523881MB/s File received! File size: 2MB. Time for receiveing: 11.262176ms. Average speed: 177.585575MB/s File received! File size: 2MB. Time for receiveing: 10.242959ms. Average speed: 195.256078MB/s
2	File received! File size: 2MB. Time for receiveing: 4055.125910ms. Average speed: 0.493203MB/s File received! File size: 2MB. Time for receiveing: 4056.413256ms. Average speed: 0.493046MB/s File received! File size: 2MB. Time for receiveing: 11.753094ms. Average speed: 170.167957MB/s File received! File size: 2MB. Time for receiveing: 4050.179871ms. Average speed: 0.493805MB/s File received! File size: 2MB. Time for receiveing: 5058.243091ms. Average speed: 0.395394MB/s
5	File received! File size: 2MB. Time for receiveing: 10097.726543ms. Average speed: 0.198064MB/s File received! File size: 2MB. Time for receiveing: 2023.200096ms. Average speed: 0.988533MB/s File received! File size: 2MB. Time for receiveing: 4037.917640ms. Average speed: 0.495305MB/s File received! File size: 2MB. Time for receiveing: 6053.913920ms. Average speed: 0.330365MB/s File received! File size: 2MB. Time for receiveing: 6046.092243ms. Average speed: 0.330792MB/s
10	File received! File size: 2MB. Time for receiveing: 12092.975631ms. Average speed: 0.165385MB/s File received! File size: 2MB. Time for receiveing: 12101.354310ms. Average speed: 0.165271MB/s File received! File size: 2MB. Time for receiveing: 6054.775695ms. Average speed: 0.330318MB/s File received! File size: 2MB. Time for receiveing: 25195.134538ms. Average speed: 0.079380MB/s File received! File size: 2MB. Time for receiveing: 18136.787861ms. Average speed: 0.110273MB/s

ביצועי הRUDP הרבה פחות טובים מביצועי הTCP, אפילו במצב שבTCP יש איבוד פקטות גבוה וב-RUDP יש 0 איבוד פקטות.

3. לפי הנתונים שלנו תמיד אעדיף להשתמש ב-TCP על פני המימוש שלנו של RUDP.

תשובות לשאלות בעברית

1. **בקשר ארוך על גבי רשת לא אמינה עם RTT גדול** השינוי עשוי להועיל במידה המירבית.
נימוק: בקשר ארוך אנו נידרש לשלוח הרבה חבילות לכן הגיוני להגדיל את חלון השליחה. ומכיוון שהרשת אינה אמינה אנו יודעים שהרבה חבילות ילכו לאיבוד לכן נרצה להגדיל את חלון השליחה בצורה אגרסיבית יותר. ולבסוף, כיוון שה-RTT גדול ישנה השהייה בין שליחת החבילה וקבלת מענה מהצד השני, לכן נרצה לפצות על כך בהגדלת חלון השליחה.
2. ??
3. $X = 1$

לפי מה שלמדנו בהרצאה מתקיים:

$$d_{nodal} = d_{proc} + d_{queue} + d_{trans} + d_{prop}$$

d_{trans}: transmission delay:
- L: packet length (bits)
- R: link *transmission rate (bps)*
- $d_{trans} = L/R$

d_{prop}: propagation delay:
- d: length of physical link
- s: propagation speed (~2×10^8 m/sec)
- $d_{prop} = d/s$

d_{trans} and d_{prop} *very* different

* Check out the online interactive exercises: http://gaia.cs.umass.edu/kurose_ross

Introduction: 1-48

לפי הנתונים בשאלה d_{proc} ו-d_{queue} זניחים. נחשב את שאר המשתנים:

$$d_{trans} + d_{prop} = \frac{L}{R} + \frac{d}{s} = \frac{1000}{8 \cdot 10^9} + \frac{1000}{2 \cdot 10^8} = \frac{1}{8 \cdot 10^6} + \frac{1}{2 \cdot 10^5} = \frac{41}{8 \cdot 10^6} \approx 0.000005125$$