

## חלק א

הערה: כשאני מסביר לפי מספר מסוים הכוונה לפי מספר ההודעה (ראה חץ כתום)

IP לקוח-192.168.1.21

IP שרת-192.168.1.15

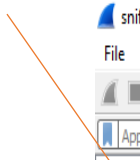
### הקמת החיבור:

1: הלקוח מנסה להתחבר לשרת, על ידי שליחת SYN, הוא מעביר לו את הseq number שאליו אמור להתחיל להיכתב המידע.

2: השרת מחזיר לו בהודעה המכילה שתי מטרות, הראשונה ACK על הודעת הSYN של הלקוח, שבעקבותיה הוא מצרף את הack number שעד אליו קיבל את המידע, שהוא seq+1 שהלקוח שלח.

בנוסף הוא שולח בהודעה SYN ללקוח בכך מודיע שרוצה לקיים אתו חיבור, כמו קודם לכן הוא מצרף seq כדי להגיד מאיפה להתחיל לכתוב את המידע.

3: הלקוח מחזיר הודעת ACK כדי לסמן לשרת שאכן קיבל את הSYN שלו, הוא מצרף אליה number שעד אליו קיבל את המידע, שהוא seq+1 שהשרת שלח.



sniffet.pcapng

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ... <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.1.21	192.168.1.15	TCP	66	55815 → 12345 [SYN] Seq=3655929887 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
2	0.000115	192.168.1.15	192.168.1.21	TCP	66	12345 → 55815 [SYN, ACK] Seq=1853071856 Ack=3655929888 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
3	0.001961	192.168.1.21	192.168.1.15	TCP	60	55815 → 12345 [ACK] Seq=3655929888 Ack=1853071857 Win=262656 Len=0

## מהלך החיבור:

4: הלקוח שולח לשרת את השמות שלנו ומצרף את seq והlen כדי שהשרת ידע מאיפה להתחיל לכתוב את המידע בrecv buffer ואת אורכו הוא גם מעביר את ack number.

5: השרת מחזיר את המידע בהודעה משולבת, הוא גם מודיע שקיבל את המידע ניתן לראות זאת כי ack number גדל בדיוק באורך המידע שהלקוח שלח (0 ביחס להודעה 3), ובנוסף הוא מחזיר ללקוח את מה ששלח לכן הוא מצרף את len, seq שתפקידם כפי שתארתי קודם.

6: הלקוח מחזיר ACK על המידע שהשרת שלח, שוב ניתן לראות זאת כי ack number גדל בדיוק באורך המידע שנשלח.

7: כעת, הלקוח שולח שוב לשרת הפעם את התז שלנו, כמו עם השמות הseq מייצג מאיפה לרשום את המידע בbuffer והlen את גודלו.

8: השרת מחזיר כמובן ack על המידע (ניתן לראות שגדל בכאורך המידע אשר נשלח), בנוסף הוא מחזיר seq, len מתאימים כי הוא מחזיר ללקוח את התז.

3	0.001961	192.168.1.21	192.168.1.15	TCP	60	55815 → 12345	[ACK]	Seq=3655929888	Ack=1853071857	Win=262656	Len=0
4	0.001961	192.168.1.21	192.168.1.15	TCP	81	55815 → 12345	[PSH, ACK]	Seq=3655929888	Ack=1853071857	Win=262656	Len=27
5	0.018627	192.168.1.15	192.168.1.21	TCP	81	12345 → 55815	[PSH, ACK]	Seq=1853071857	Ack=3655929915	Win=65536	Len=27
6	0.074018	192.168.1.21	192.168.1.15	TCP	60	55815 → 12345	[ACK]	Seq=3655929915	Ack=1853071884	Win=262656	Len=0
7	0.080125	192.168.1.21	192.168.1.15	TCP	74	55815 → 12345	[PSH, ACK]	Seq=3655929915	Ack=1853071884	Win=262656	Len=20
8	0.080187	192.168.1.15	192.168.1.21	TCP	74	12345 → 55815	[PSH, ACK]	Seq=1853071884	Ack=3655929935	Win=65536	Len=20

## סיום החיבור:

9: השרת שולח ללקוח שהוא רוצה להתנתק בעזרת הודעת FIN

10: הלקוח שולח לשרת שהוא רוצה להתנתק על ידי הודעת FIN (הוא לא קיבל עדיין את שלו- נסיק זאת כי ack number לא גדל מהודעה 7)

11: השרת מאשר את הודעת הFIN של הלקוח (נסיק זאת כי הack גדל מאז הודעה 9)

12: הלקוח מאשר את הודעת הFIN של השרת ( נראה שהשרת כבר סגר את החיבור כי הוא רצה לסגור וגם קיבל הודעת FIN מהלקוח לכן הוא סגר את החיבור מה שגרם לכך שהודעת הACK של הלקוח על הסגירה לא להגיע)

7	0.080125	192.168.1.21	192.168.1.15	TCP	74	55815 → 12345	[PSH, ACK]	Seq=3655929915	Ack=1853071884	Win=262656	Len=20
8	0.080187	192.168.1.15	192.168.1.21	TCP	74	12345 → 55815	[PSH, ACK]	Seq=1853071884	Ack=3655929935	Win=65536	Len=20
9	0.080261	192.168.1.15	192.168.1.21	TCP	54	12345 → 55815	[FIN, ACK]	Seq=1853071904	Ack=3655929935	Win=65536	Len=0
10	0.080844	192.168.1.21	192.168.1.15	TCP	60	55815 → 12345	[FIN, ACK]	Seq=3655929935	Ack=1853071884	Win=262656	Len=0
11	0.080864	192.168.1.15	192.168.1.21	TCP	54	12345 → 55815	[ACK]	Seq=1853071905	Ack=3655929936	Win=65536	Len=0
12	0.082154	192.168.1.21	192.168.1.15	TCP	60	55815 → 12345	[RST, ACK]	Seq=3655929936	Ack=1853071904	Win=0	Len=0