

מטלה 3:  
רשתות תקשורת  
TCP

מגישים:  
328596978  
212767214

### MAKEFILE

מצורף לקוד קובץ מייקפייל שמפעיל את קבצי c בפקודה make של sender ו-receiver כמו כן, ניתן למחוק את כל הקבצים המיותרים שנוצרו עקב הפעלת המייקפייל עם הפקודה make clean.

### receiver.c

בעת הפעלת המייקפייל נפעיל קודם ./receiver ונראה בהדפסה שנוצר הסוקט והוא נמצא בהזנה עבור יצירת קשר עם sender. ברגע שנוצר הקשר receiver מקבל חצי מהקובץ מהsender ואז יש בדיקת האותנטיקציה של 4 ספרות של שני ת"ז המגישים xor (ברגע שהסנדר מאשר את xor אם הוא נכון שולח את החצי השני). אחרי אישור האותנטיקציה יש שינוי בcc algoritem לreno ע"י setsocket (כמו כן במקביל יש את השינוי בsender). לאחר מכן receiver מחכה אם המשתמש מצד sender רוצה לעשות שליחה נוספת של חלקי הקובץ. אם כן בחלק הראשון receiver משנה חזרה cc algoritem לcubic ואז עושה שוב את פעולה כמו בפעם הקודמת. לבסוף אם המשתמש הפסיק את שליחת הקבצים, receiver מפרסם את זמני הקבלה ששמרנו באמצעות רשימה מקושרת בה אחרי כל שליחה של חצי קובץ שמנו בdata את הזמן במיקרו שניות. אחרי הדפסת הזמנים התוכנה גם מדפיסה ממוצע. ולבסוף סוגרת את הסוקט ומדפיסה שהתוכנית נסגרה.

### sender.c

יוצר קשר tcp עם receiver. לאחר מכן, פותח קובץ **הערה** שם הקובץ צריך להיות send.txt (גודל לא משנה) מחלק את הקובץ לשניים ולאחר מכן, שולח את הראשון לreceiver ואז מחכה לתשובה של xor אם יש אישור משנה cc algoritem לreno ושולח את כל החלק השני של הקובץ.  
בterminal:

המשתמש מקבל הודעה האם הוא רוצה לשלוח שוב קובץ מחדש yes/no במידה והתשובה yes sender נכנס ללולאה ושולח מחדש את הקובץ (לפני שליחה של החלק הראשון הוא משנה את cc algoritem חזרה לcubic). במידה והתשובה no התוכנית תשאל אם המשתמש רוצה לצאת (exit ? yes/no) במידה והוא עונה no חוזר לשאלה הקודמת במידה ו yes אז התוכנית נסגרת.

## תצלומים של איבוד פאקטות:

## אחוזי איבוד:

## 0% איבוד פאקטות:

```
niktita@nikita-VirtualBox:~/documents/Test_1/create_server$ make
gcc -Wall -g -c -o Receiver.o Receiver.c
gcc -Wall -g -o Receiver Receiver.o
gcc -Wall -g -o Sender Sender.c
niktita@nikita-VirtualBox:~/documents/Test_1/create_server$ ./Receiver
Server socket created successfully.
Binding successfully.
Listening...
loop: 1
time first half is: 155032
(c) send Authentication XOR to sender
time second half is: 137506
loop: 2
time first half is: 99478
(c) send Authentication XOR to sender
time second half is: 133604
loop: 3
time first half is: 95408
(c) send Authentication XOR to sender
time second half is: 132096
loop: 4
time first half is: 97112
(c) send Authentication XOR to sender
time second half is: 132204
loop: 5
time first half is: 91640
(c) send Authentication XOR to sender
time second half is: 133435

The times (in micro-seconds) of first half are: 155032 99478 95408 97112 91640
The times (in micro-seconds) of second half are: 137506 133604 132096 132204 133435
The average (in micro-seconds) of first half is: 107734
The average (in micro-seconds) of second half is: 133769
***Closing the connection.***
niktita@nikita-VirtualBox:~/documents/Test_1/create_server$
```

```

kali@kali:~/VirtualBox/~/Documents/Test3/create_server$ ./Sender
Server socket created successfully.
Connected to Server.
first half of File sent
second half of File sent
USER DECISION: send the file again ? yes/no --> yes

first half of File sent
second half of File sent
USER DECISION: send the file again ? yes/no --> yes

first half of File sent
second half of File sent
USER DECISION: send the file again ? yes/no --> yes

first half of File sent
second half of File sent
USER DECISION: send the file again ? yes/no --> yes

first half of File sent
second half of File sent
USER DECISION: send the file again ? yes/no --> no
EXIT ? yes/no --> no
n1k17@kali:~/VirtualBox/~/Documents/Test3/create_server$

```

[illegible]

(קובץ pcap של וירשארק מצורף בניפרד)

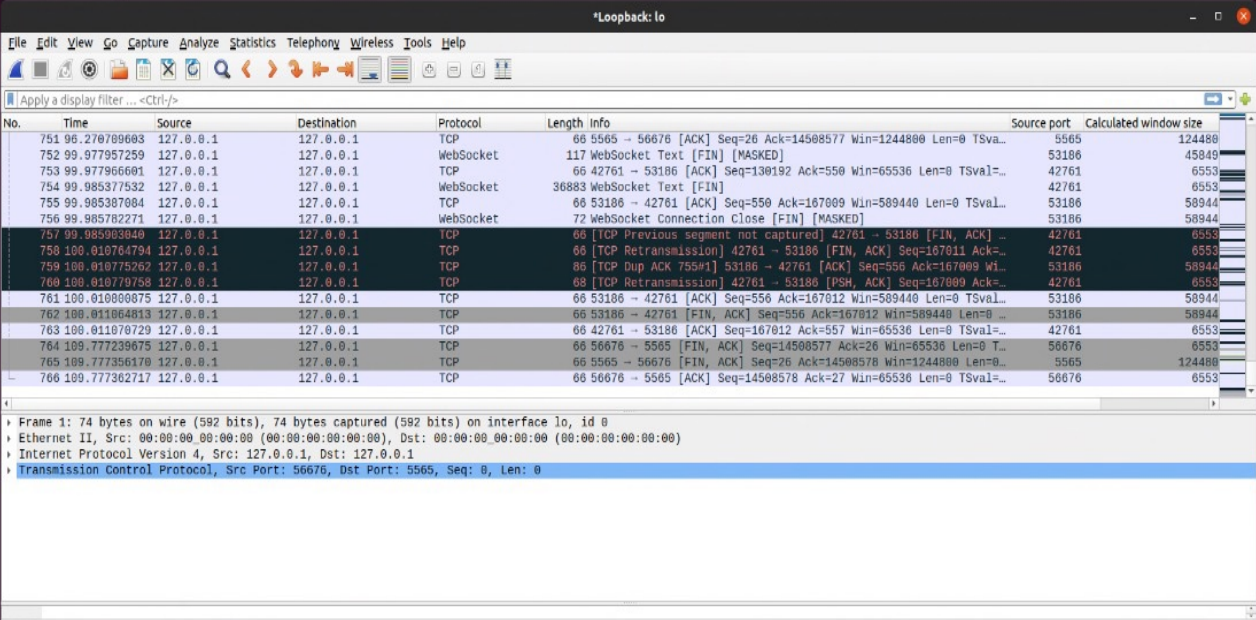
ניתן לראות כי שלחנו את הקובץ 5 פעמים (מצד שמאל sender) כמו כן ב%0 איבוד התוצאה הראה שבממוצע האלגוריתם cubic הראה תוצאה יותר מהירה בהשוואה לreno.

## 10% איבוד פאקטות:

```
nikita@nikita-VirtualBox: ~/Documents/Test_1/create_server
nikita@nikita-VirtualBox:~/Documents/Test_1/create_server$ ./Receiver
Server socket created successfully.
Binding successful.
Listening...
loop: 1
time first half is: 165678
(+) send Authentication XOR to sender
time second half is: 93157
loop: 2
time first half is: 98915
(+) send Authentication XOR to sender
time second half is: 87709
loop: 3
time first half is: 113873
(+) send Authentication XOR to sender
time second half is: 90042
loop: 4
time first half is: 310378
(+) send Authentication XOR to sender
time second half is: 89754
loop: 5
time first half is: 112303
(+) send Authentication XOR to sender
time second half is: 100081

The times (in micro-seconds) of first half are: 165678 98915 113873 310378 112303
The times (in micro-seconds) of second half are: 93157 87709 90042 89754 100081
The average (in micro-seconds) of first half is: 160229
The average (in micro-seconds) of second half is: 92148
=====Closing the connection.=====
nikita@nikita-VirtualBox:~/Documents/Test_1/create_server$

nikita@nikita-VirtualBox:~/Documents/Test_1/create_server$ ./Sender
Server socket created successfully.
Connected to Server.
first half of File sent
second half of File sent
USER DECISION: send the file again ? yes/no --> yes
first half of File sent
second half of File sent
USER DECISION: send the file again ? yes/no --> yes
first half of File sent
second half of File sent
USER DECISION: send the file again ? yes/no --> yes
first half of File sent
second half of File sent
USER DECISION: send the file again ? yes/no --> yes
first half of File sent
second half of File sent
USER DECISION: send the file again ? yes/no --> no
Exit ? yes/no --> yes
=====Closing the connection.=====
nikita@nikita-VirtualBox:~/Documents/Test_1/create_server$
```



No.	Time	Source	Destination	Protocol	Length	Info	Source port	Calculated window size
751	96.270709603	127.0.0.1	127.0.0.1	TCP	66	5565 → 56678 [ACK] Seq=26 Ack=14508577 Win=1244800 Len=0 TSval=...	5565	124480
752	99.977957259	127.0.0.1	127.0.0.1	WebSocket	117	WebSocket Text [FIN] [MASKED]	53186	45849
753	99.977966601	127.0.0.1	127.0.0.1	TCP	66	42761 → 53186 [ACK] Seq=130192 Ack=550 Win=65536 Len=0 TSval=...	42761	6553
754	99.985377532	127.0.0.1	127.0.0.1	WebSocket	36883	WebSocket Text [FIN]	42761	6553
755	99.985387804	127.0.0.1	127.0.0.1	TCP	66	53186 → 42761 [ACK] Seq=550 Ack=167009 Win=509440 Len=0 TSval=...	53186	58944
756	99.985782271	127.0.0.1	127.0.0.1	WebSocket	72	WebSocket Connection Close [FIN] [MASKED]	53186	58944
757	99.985903040	127.0.0.1	127.0.0.1	TCP	66	[TCP Previous segment not captured] 42761 → 53186 [FIN, ACK] ...	42761	6553
758	100.010764794	127.0.0.1	127.0.0.1	TCP	66	[TCP Retransmission] 42761 → 53186 [FIN, ACK] Seq=167011 Ack=...	42761	6553
759	100.010775262	127.0.0.1	127.0.0.1	TCP	66	[TCP Dup ACK 755#1] 53186 → 42761 [ACK] Seq=550 Ack=167009 Win=...	53186	58944
760	100.010770759	127.0.0.1	127.0.0.1	TCP	66	[TCP Retransmission] 42761 → 53186 [PSH, ACK] Seq=167009 Ack=...	42761	6553
761	100.010800075	127.0.0.1	127.0.0.1	TCP	66	53186 → 42761 [ACK] Seq=550 Ack=167012 Win=509440 Len=0 TSval=...	53186	58944
762	100.011084813	127.0.0.1	127.0.0.1	TCP	66	53186 → 42761 [FIN, ACK] Seq=550 Ack=167012 Win=509440 Len=0 ...	53186	58944
763	100.011070720	127.0.0.1	127.0.0.1	TCP	66	42761 → 53186 [ACK] Seq=167012 Ack=557 Win=65536 Len=0 TSval=...	42761	6553
764	100.777239675	127.0.0.1	127.0.0.1	TCP	66	56678 → 5565 [FIN, ACK] Seq=14508577 Ack=26 Win=65536 Len=0 TSval=...	56678	6553
765	100.777356170	127.0.0.1	127.0.0.1	TCP	66	5565 → 56678 [FIN, ACK] Seq=26 Ack=14508578 Win=1244800 Len=0 ...	5565	124480
766	100.777362717	127.0.0.1	127.0.0.1	TCP	66	56678 → 5565 [ACK] Seq=14508578 Ack=27 Win=65536 Len=0 TSval=...	56678	6553

```
Frame 1: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface lo, id 0
Ethernet II, Src: 00:00:00:00:00:00 (00:00:00:00:00:00), Dst: 00:00:00:00:00:00 (00:00:00:00:00:00)
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
Transmission Control Protocol, Src Port: 56678, Dst Port: 5565, Seq: 0, Len: 0
```

ניתן לראות כי שלחנו את הקובץ 5 פעמים (מצד ימין sender) כמו כן ב10% איבוד התוצאה הראה שבממוצע האלגוריתם שחזר הראה תוצאה יותר מהירה בהשוואה לcubic.

## 15% איבוד פאקטות:

The image shows a Kali Linux terminal window and a Wireshark network traffic capture. The terminal window is split into two panes. The left pane shows the output of a server script, and the right pane shows the output of a client script. Both scripts are running on a Kali Linux virtual machine.

**Terminal - Left Pane (Server):**

```
nikita@nikita-VirtualBox: ~/Documents/Test_1/create_server
Server socket created successfully.
Binding successful.
Listening...
loop: 1
time first half is: 149017
(*) send Authentication XOR to sender
time second half is: 107841
loop: 2
time first half is: 92443
(*) send Authentication XOR to sender
time second half is: 93916
loop: 3
time first half is: 88814
(*) send Authentication XOR to sender
time second half is: 92732
loop: 4
time first half is: 3625963
(*) send Authentication XOR to sender
time second half is: 88603
loop: 5
time first half is: 123441
(*) send Authentication XOR to sender
time second half is: 91334

The times (in micro-seconds) of first half are: 149017 92443 88814 3625963 123441
The times (in micro-seconds) of second half are: 107841 93916 92732 88603 91334
The average (in micro-seconds) of first half is: 815935
The average (in micro-seconds) of second half is: 94885
=====Closing the connection,=====
nikita@nikita-VirtualBox: ~/Documents/Test_1/create_server$
```

**Terminal - Right Pane (Sender):**

```
nikita@nikita-VirtualBox: ~/Documents/Test_1/create_server$ ./Sender
Server socket created successfully.
Connected to Server.
First half of File sent
second half of File sent
USER DECISION: send the file again ? yes/no --> yes
First half of File sent
second half of File sent
USER DECISION: send the file again ? yes/no --> yes
First half of File sent
second half of File sent
USER DECISION: send the file again ? yes/no --> yes
First half of File sent
second half of File sent
USER DECISION: send the file again ? yes/no --> no
Exit ? yes/no --> yes
=====Closing the connection,=====
nikita@nikita-VirtualBox: ~/Documents/Test_1/create_server$
```

**Wireshark:**

The Wireshark window shows a network traffic capture on interface lo. The packet list shows several TCP and DNS packets. The packet details pane shows the selected packet (Frame 1) and its details.

No.	Time	Source	Destination	Protocol	Length	Info	Source port	Calculated window size
761	99.693405958	127.0.0.1	127.0.0.1	TCP	66	5565 → 56682 [ACK] Seq=26 Ack=14432860 Win=2783184 Len=0 TSva...	5565	278318
762	99.693410718	127.0.0.1	127.0.0.1	TCP	10296	[TCP Previous segment not captured] 56682 → 5565 [PSH, ACK] S...	56682	6553
763	99.693412091	127.0.0.1	127.0.0.1	TCP	78	[TCP Dup ACK 761#1] 5565 → 56682 [ACK] Seq=26 Ack=14432860 W...	5565	278318
764	99.693419028	127.0.0.1	127.0.0.1	TCP	65549	[TCP Out-Of-Order] 56682 → 5565 [ACK] Seq=14432860 Ack=26 Win...	56682	6553
765	99.693423997	127.0.0.1	127.0.0.1	TCP	66	5565 → 56682 [ACK] Seq=26 Ack=14508573 Win=2745216 Len=0 TSva...	5565	274521
766	103.533536671	127.0.0.1	127.0.0.53	DNS	84	Standard query response 0x3b1a A cdn.fwupd.org OPT	48000	
767	103.533628665	127.0.0.1	127.0.0.1	DNS	141	Standard query response 0x3b1a A cdn.fwupd.org CNAME p2.share...	53	
768	105.308911343	127.0.0.1	127.0.0.1	TCP	70	56682 → 5565 [PSH, ACK] Seq=14508573 Ack=26 Win=65536 Len=4 T...	56682	6553
769	105.308917469	127.0.0.1	127.0.0.1	TCP	66	5565 → 56682 [ACK] Seq=26 Ack=14508577 Win=3145088 Len=0 TSva...	5565	314508
770	106.565657642	127.0.0.1	127.0.0.53	DNS	84	Standard query 0x3b1a A cdn.fwupd.org OPT	42221	
771	106.565751311	127.0.0.1	127.0.0.1	DNS	141	Standard query response 0x3b1a A cdn.fwupd.org CNAME p2.share...	53	
772	122.305149202	127.0.0.1	127.0.0.1	TCP	66	56682 → 5565 [FIN, ACK] Seq=14508577 Ack=26 Win=65536 Len=0 T...	56682	6553
773	122.305217794	127.0.0.1	127.0.0.1	TCP	66	5565 → 56682 [FIN, ACK] Seq=26 Ack=14508578 Win=3145728 Len=0...	5565	314572
774	122.511061305	127.0.0.1	127.0.0.1	TCP	66	[TCP Retransmission] 5565 → 56682 [FIN, ACK] Seq=26 Ack=14508...	5565	314572
775	122.718025826	127.0.0.1	127.0.0.1	TCP	66	[TCP Retransmission] 5565 → 56682 [FIN, ACK] Seq=26 Ack=14508...	5565	314572
776	122.718036739	127.0.0.1	127.0.0.1	TCP	66	56682 → 5565 [ACK] Seq=14508578 Ack=27 Win=65536 Len=0 TSva...	56682	6553

Frame 1: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface lo, id 0  
Ethernet II, Src: 00:00:00:00:00:00 (00:00:00:00:00:00), Dst: 00:00:00:00:00:00 (00:00:00:00:00:00)  
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1  
Transmission Control Protocol, Src Port: 56682, Dst Port: 5565, Seq: 0, Len: 0

ניתן לראות כי שלחנו את הקובץ 5 פעמים (מצד ימין sender) כמו כן ב-15% איבוד התוצאה הראה שבממוצע האלגוריתם סובט הראה תוצאה יותר מהירה בהשוואה לסר.



## 20% איבוד פאקטות:

```
nikita@nikita-VirtualBox: ~/Documents/Test_1/create_server
nikita@nikita-VirtualBox:~/Documents/Test_1/create_server$ ./Receiver
Server socket created successfully.
Binding successfull.
Listening...
Loop: 1
time first half is: 383786
(+) send Authentication XOR to sender
time second half is: 107022
Loop: 2
time first half is: 94926
(+) send Authentication XOR to sender
time second half is: 97143
Loop: 3
time first half is: 101694
(+) send Authentication XOR to sender
time second half is: 522261
Loop: 4
time first half is: 313397
(+) send Authentication XOR to sender
time second half is: 97646
Loop: 5
time first half is: 10336
(+) send Authentication XOR to sender
time second half is: 112872

The times (in micro-seconds) of first half are: 383786 94926 101694 313397 10336
The times (in micro-seconds) of second half are: 107022 97143 522261 97646 112872
The average (in micro-seconds) of first half is: 180827
The average (in micro-seconds) of second half is: 187388
=====Closing the connection.=====
nikita@nikita-VirtualBox:~/Documents/Test_1/create_server$

nikita@nikita-VirtualBox:~/Documents/Test_1/create_server$ ./Sender
Server socket created successfully.
Connected to Server.
first half of File sent
second half of File sent
USER DECISION: send the file again ? yes/no --> yes
first half of File sent
second half of File sent
USER DECISION: send the file again ? yes/no --> yes
first half of File sent
second half of File sent
USER DECISION: send the file again ? yes/no --> yes
first half of File sent
second half of File sent
USER DECISION: send the file again ? yes/no --> yes
first half of File sent
second half of File sent
USER DECISION: send the file again ? yes/no --> no
Exit ? yes/no --> yes
=====closing the connection.=====
nikita@nikita-VirtualBox:~/Documents/Test_1/create_server$
```

\*Loopback: lo

No.	Time	Source	Destination	Protocol	Length	Info	Source port	Calculated w
454	93.927407991	127.0.0.1	127.0.0.1	TCP	66	5565 → 45384 [ACK] Seq=26 Ack=13647066 Win=2357808 Len=0 TSva...	5565	
455	93.927503248	127.0.0.1	127.0.0.1	TCP	65549	[TCP Previous segment not captured] 45384 → 5565 [ACK] Seq=13...	45384	
456	93.927509005	127.0.0.1	127.0.0.1	TCP	86	[TCP Window Update] 5565 → 45384 [ACK] Seq=26 Ack=13647066 Wi...	5565	
457	93.927518871	127.0.0.1	127.0.0.1	TCP	65549	[TCP Retransmission] 45384 → 5565 [ACK] Seq=13778032 Ack=26 W...	45384	
458	93.927525169	127.0.0.1	127.0.0.1	TCP	78	[TCP Window Update] 5565 → 45384 [ACK] Seq=26 Ack=13647066 Wi...	5565	
459	93.927535384	127.0.0.1	127.0.0.1	TCP	65549	[TCP Retransmission] 45384 → 5565 [PSH, ACK] Seq=13647066 Ack...	45384	
460	93.927540469	127.0.0.1	127.0.0.1	TCP	66	5565 → 45384 [ACK] Seq=26 Ack=13908998 Win=2781184 Len=0 TSva...	5565	
461	94.373928008	127.0.0.1	127.0.0.1	TCP	65549	45384 → 5565 [PSH, ACK] Seq=13908998 Ack=26 Win=65536 Len=654...	45384	
462	99.697625961	127.0.0.1	127.0.0.1	TCP	65549	[TCP Retransmission] 45384 → 5565 [PSH, ACK] Seq=13908998 Ack...	45384	
463	99.697646929	127.0.0.1	127.0.0.1	TCP	78	5565 → 45384 [ACK] Seq=26 Ack=13974481 Win=2912128 Len=0 TSva...	5565	
464	99.697658068	127.0.0.1	127.0.0.1	TCP	65549	[TCP Previous segment not captured] 45384 → 5565 [ACK] Seq=14...	45384	
465	99.697670323	127.0.0.1	127.0.0.1	TCP	65549	45384 → 5565 [PSH, ACK] Seq=14170930 Ack=26 Win=65536 Len=654...	45384	
466	99.697682718	127.0.0.1	127.0.0.1	TCP	78	[TCP Window Update] 5565 → 45384 [ACK] Seq=26 Ack=13974481 Wi...	5565	
467	99.697691951	127.0.0.1	127.0.0.1	TCP	65549	[TCP Retransmission] 45384 → 5565 [ACK] Seq=13974481 Ack=26 W...	45384	
468	99.697698465	127.0.0.1	127.0.0.1	TCP	65549	[TCP Retransmission] 45384 → 5565 [ACK] Seq=14030904 Ack=26 W...	45384	
469	99.697714549	127.0.0.1	127.0.0.1	TCP	66	5565 → 45384 [ACK] Seq=26 Ack=14236413 Win=3012864 Len=0 TSva...	5565	
470	99.697722583	127.0.0.1	127.0.0.1	TCP	65549	45384 → 5565 [ACK] Seq=14236413 Ack=26 Win=65536 Len=65483 TS...	45384	

Frame 1: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface lo, id 0  
Ethernet II, Src: 00:00:00:00:00:00 (00:00:00:00:00:00), Dst: 00:00:00:00:00:00 (00:00:00:00:00:00)  
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1  
Transmission Control Protocol, Src Port: 45384, Dst Port: 5565, Seq: 0, Len: 0

ניתן לראות כי שלחנו את הקובץ 5 פעמים (מצד ימין sender) כמו כן ב20% איבוד התוצאה הראה שבממוצע האלגוריתם אובדן הראה תוצאה יותר מהירה בהשוואה לrenov.