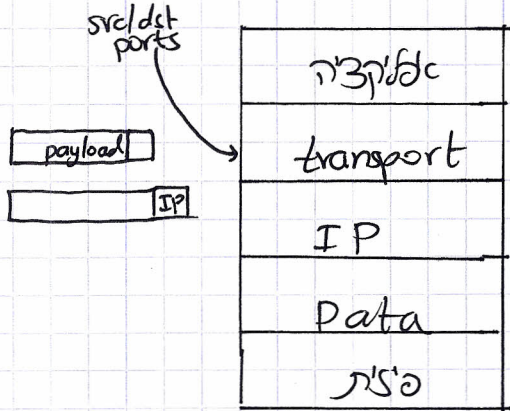


תקשורת - תרבות 4

כמותן יהיו 2 סגורות
1 ותצי סגורות

- transport
 - Sockets
 - Reliability
- נצטרך את

שכבת התעבורה מבטיחה להעביר חבילה מ-Socket השלמה עד לתחילת התקלה,
(שניה אחת ע"י socket).



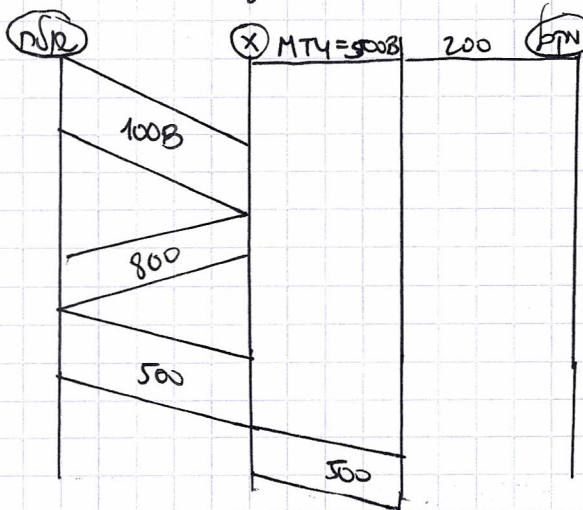
כל שכבה מבטיחה תכונות שונות לשכבה שמעליה.

skype ו-DNS הם דוגמים שרצים מאחורי UDP.

UDP אינו מבטיח כלום, רק מנסה לשלוח datagram מהשלח למקבל. אם יש אובדן חבילה ה-UDP - אין לנו צורך למחן לשלוח.

TCP זורה סמאנטציה, יכול לשנות Path MTU discovery, יכול לשנות MTU מינימלי (דבר) מהשלח למקבל. ע"י התקרת התפל של MF.

ב-TCP הוא שומר state, connect וכן הוא יכול לשנות את ה-discovery הנה.



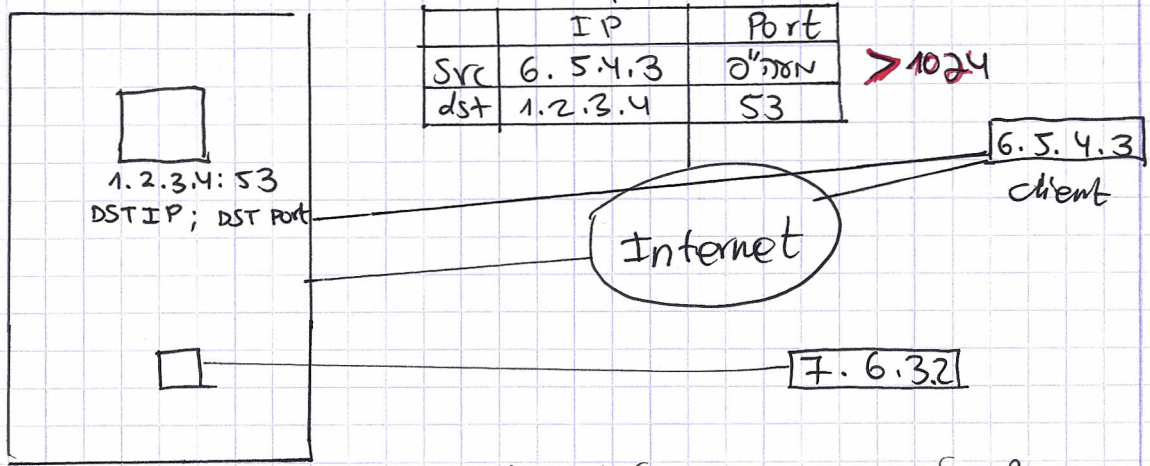
TCP יכול לשנות את החוק שומר דומה. שומר כמה חבילה לא מות לשלוח את MTU discovery. גמ'ז שולחים את התקטימ'ל שצ'ינו עד עכשיו - כך נשלים את Path MTU discovery. אם לא השחצים אומכם ה-MTU הנה אל אנה לא ישלח לי שום דבר (לא חוצרת ICMP). וכך נוצר שינוי לשלוח באופן חנה.

שלב חוצרת ICMP <= 40B ד-headers, לכן ה"כ חוצרת כצלב 70/80B.

UDP יתר יעיל כי אינו שומר state ואינו מורה ל-Ack על כל חבילה.

שכבת האנאליס מבקרת את ה-IP, ומת'ה בקוקה ל-IP כפי לשלוח את ה-Socket

שורת DNS - ממפה בין כתובות IP ל-URL, כ"י התולים לא צריכים טיפוס
 ע"פ כתובת ה-URL. שמו פרוטוקול נשכרת האפליקציה.
 זה פרוטוקול http נוספת אפליקציה.
 UDP (DNS)



DNS רץ ממשל UDP כי הוא צריך להיות מאלו מהיר
 ולכן stateless

ה- port הוא המזהה של ה-Socket

ה-Socket מוגדר ע"י: DST PORT + DEST IP

מהו יוצרת לקחת את ה- IP משכנת ה-IP את ה- port משכנת התעבורה.

יצירת socket ה- python: Client.py

```
import socket
```

```
host = '1.2.3.4'  
port = 53
```

```
s = socket.socket(socket.AF_INET,
```

```
socket.SOCK_STREAM)
```

```
s.bind((host, port))
```

socket Name	Protocol
stream	- TCP
datagram	- UDP

בפתיחת (נצטר) אלו פרוטוקול
 ע"י שימוש למדידת חילוף קצרות אמינה

הבעלים בה לקוח ולמדי:

המפיק השרת הוא מזהה לקבל פניות, לכן זה שונה.

נרשום ב host - IP, ה- host של השרת זה יהיה 0.0.0.0

נקיים לא לשים שם בלי, אלא את ה- local host - כחכם במעלה, אלא את כולו המערכת.

אם נשים את זה hardcoded אז ה- host זה לא ירץ למחשב.

ה- connect עשה ושרת שאין לנתונים, מחכה לקבל נתונים.

כולם מניעים לאלו מקום דו-כיוון, לכן נרצה לראות - fairness.

ב buffer כותבים את מה שקרא.

כיצד? אם קלינט אחד ישלח חמון

כפניות של קלינטים אחרים.

מימש אפשרי לכך הוא ע"י socket - socket מאגין, כך שכל אחד עונה

קלינט הוא יקצה לו socket חדש. כל קלינט יקח מקום מוקצה.

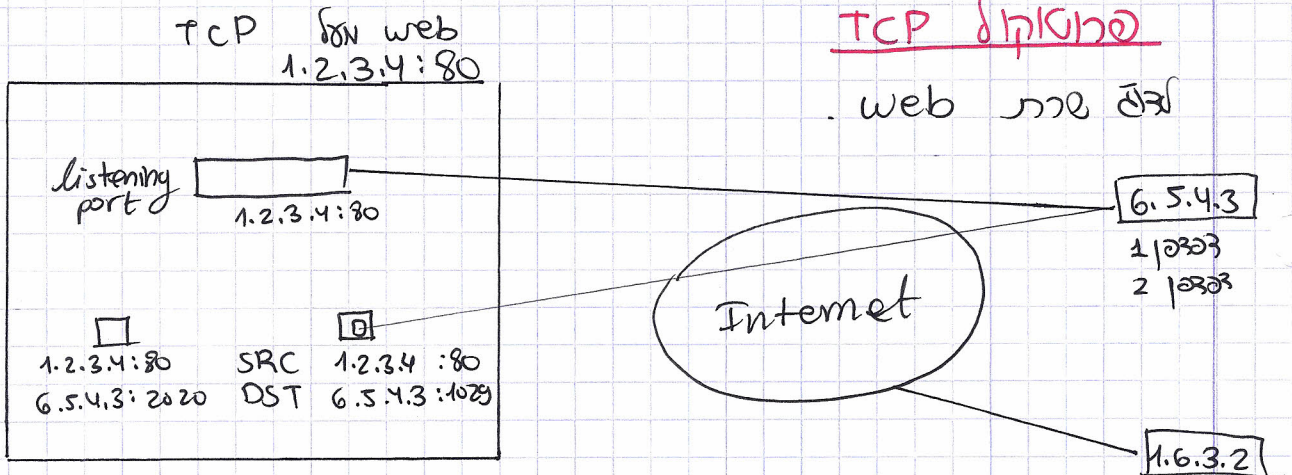
ה קלט ניתן למחוק במעט קלינטים כמעט הכותמים.

הסיני ודנה שעש"נו הוא ברמת האפליקציה, ה- user space.

סה"כ יכולים להיות למקס 1024 - 2¹⁶ לקוחות.

פרוטוקול TCP

לדוגמה שרת web.



הlistening socket הוא יחודי וטו"ו מלווה ע"י רביע"ה.

Socket ה TCP מלווה ע"י רביע"ה, הרביע"ה נמצאת ה-header.

ניתן למחוק בהתבדה יותר קלינטים, לעומת UDP.

למטה port יבני או שניה בחירת listening port, ואם הוא יקבל פורט שגוי.

TCP מבטח הנתונים כי יש buffer (פרד) לכל socket. אך עש"ן צריך להבטיח

הנתונים עם מבחנות עם לקוחות שמעודדים.

עם מנת להבטיח הנתונים צריך להבטיח מקט' חילולים אפשריים ולקוחות.

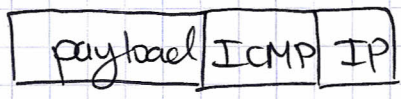
צריך להגדיל את מעט connections במקביל, למשל.

תשרת עצמו מבטל דמקס, אך מוגבלים עם ע"י SRC.



העברת קול מלקוח לשרת.

הנתב מאפשר לפגוש רק הודעות ICMP (שבת IP).



ה-ICMP זה פרוטוקול ששבת ה-IP.

נציין בheader מה type ההודעה.

בפרט: החלפות צריכות להיות תקינות מבחינת הסדרות של ICMP.

מה שמפריד socket מ-UDP/TCP ומה שאם ה-header.

אנו נצטרך להשתמש ב-raw-socket כדי לבנות headers שפגוש, צריך לעקוף את מה שיש. נשדד מופע מסוף header איתנו את הערכים בסדר המדויק. ה-headers צריך להיות את הסדר הנכונים, אחרת כרטיס הסדר לא יעלה את ההודעה אם זה לא תקין.

נצטרך לקרוא את header של ICMP כי הנתב ששם את ה-payload. אנו מקודדים את כפי לתוך את ה-checksum.

ניגן אופרטור דבר מנין שבת לזכור הקצוות. (ניגן לקודד דאוק חתלה מקב 16bit. צריכים לבדוק את ה-MTU אצל הרויזר את הקול. $2^{16} \log$ - זה הביטוי שניגן לקודד.

Reliability

המטרה: לתת לי אסוף ולמקבל שהמידע אכן עבר נכון.
משמע ה-reliability ב-TCP מורכב, ונלמד על דבורה הדומה.

RDT 1.0

data / checksum
כל בקשה?

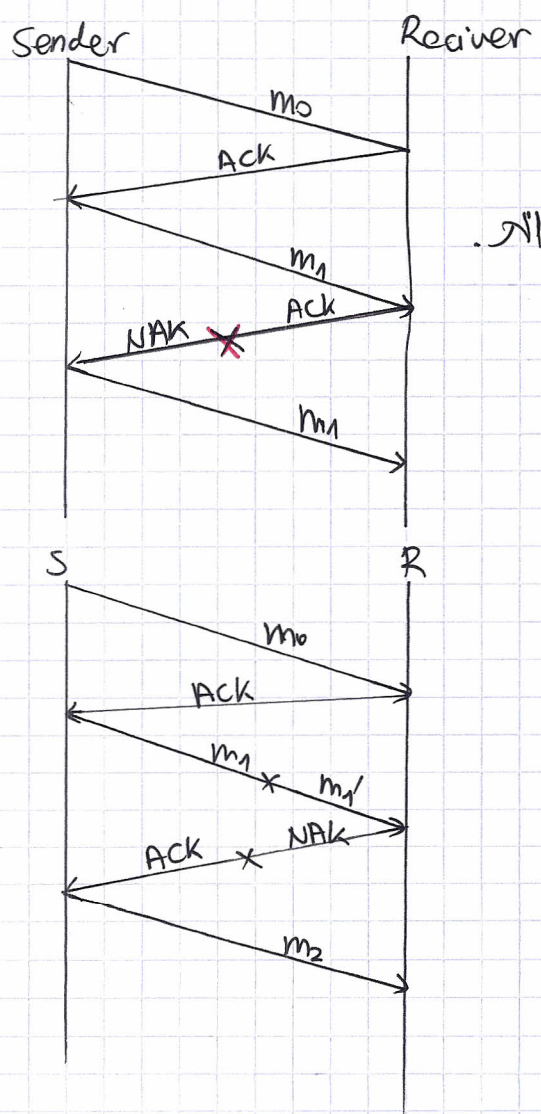
מניח שמקור אמיני: אין אובדן חבילות.
- אין שיבושים.

RDT 2.0

מניח שיש שיבושים.

שלח - מחזרים checksum Retransmission (= כולל את תפילות מחדש).
קבל - ACK/NAK

ברחיש: ACK שתקבל ומחזר NAK (מסמן ש"גיש אכזה).



אין צורך לעבור שיתונים כאלו.
הסתאוקולו אלו יוצר למה מחדש את שילוח.

ברחיש: NAK שתקבל ומחזר ACK.

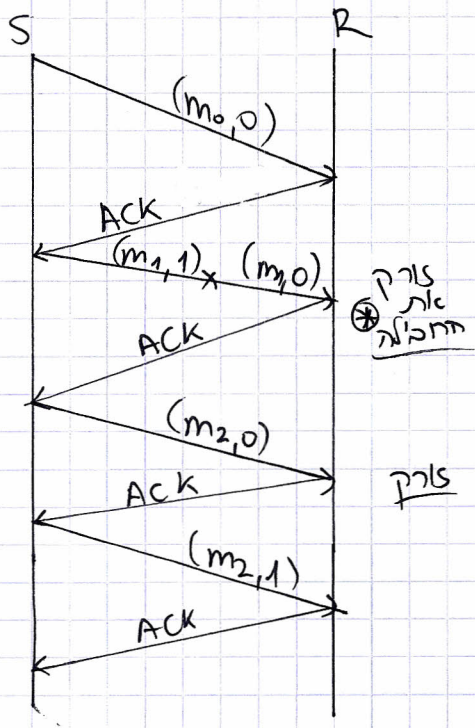
כתוצאה ששיבוש יש אובדן.

כל צד
מחזיר
את התפילות
החדש.

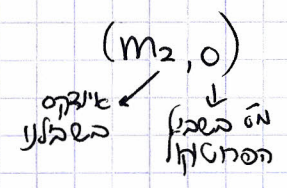
RPT 2.1

שלמה - SN (= sequence number) מחזיקת מידע.
נקבה - מוסר checksum & ACK/NAK

המקרה ש-SN שמתקבל, משמע ש'ה' הפגיאלה לא תקין.
ניתוח: משמע ה-checksum לא נשמר ש'בוש'ים.



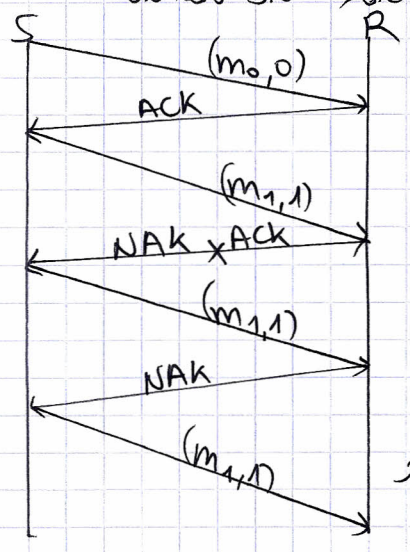
ממס' SN שמתקבל



הפחטוקול לא שלמה את החזרה הזאה.
38 שיהיו מקבל אשור & החזרה אחריו.
* את (שלמה NAK הפחטוקול יתנס
- loop א'נסו'.

שולח ACK וקבאר ה-checksum שוב.
ה-checksum לא תקין את ה-SN.

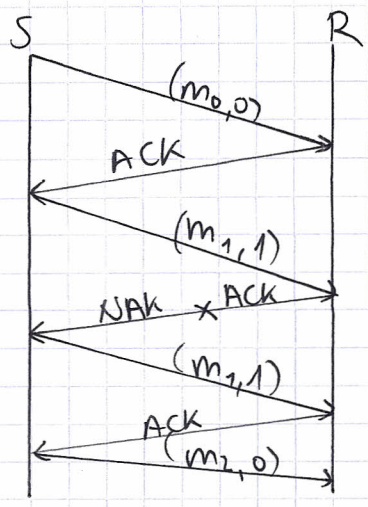
ש'ה' א' RDT כך שהמקבל ישלח NAK אם א'ם SN שובש.



קיבאר את
SN ה'ה'
נכנס ל-loop
א'נסו'!

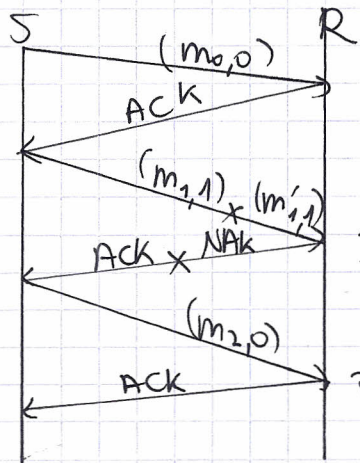
מקרה: ACK עברתי ונאק

גודל פאקטור צריך להיות



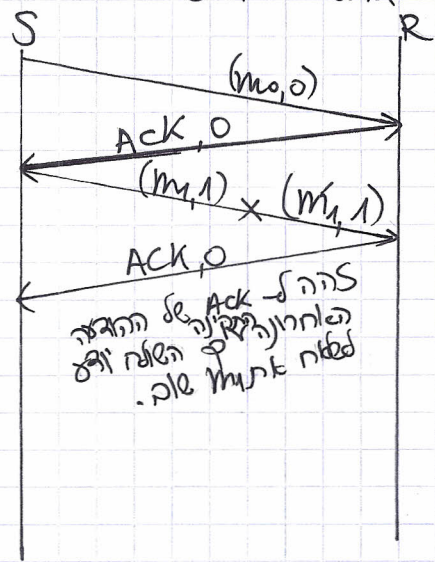
מקרה: NAK עברתי ונאק

פוליס - check if sum of bits
פוליס - נבדק 1-8



RDT2.2

כאן נאק נקודתו אי' אק אפילו נאק נאק או סני - אק



הינה פשוט וקל לראות

רדט 2.2 - 3.0