digital whisper
Technologic Papers

הכרת SCTP - חלק ראשון

מאת עידו קנר

הקדמה

בעולם התקשורת קיים מודל שבע השכבות של ה-OSI. כאשר מדברים על שכבה מספר אחת, כולם יודעים במה מדובר, אך ככול שעולים במספר השכבות, מכירים פחות ופחות מה קורה. יש כאלו שיודעים לזרוק שמות של פעולות ופרוטוקולים מסוימים, אך לא ניתן להכיר הכל.

השכבה הרביעית במספר, היא שכבה אשר הרבה מכירים, כאשר מרבית האנשים המתעסקים במקצוע יודעים לציין כי ישנם שני פרוטוקולים העובדים בשכבה זו, כאשר הפרוטוקול הראשון הוא TCP שמאוד נפוץ, ונחשב לאמין, והשני הוא UDP, אשר בנוי יותר ל-real-time ונחשב לפחות אמין.

העניין הוא שישנם עוד פרוטוקולים, אשר יודעים לשלב פעולות שונות, ומשום-מה אין הרבה אנשים אשר מכירים אותם, כדוגמת: RSVP, DCCP ואחרים. כל פרוטוקול שכזה בעל תפקיד מאוד מוגדר ברמת התעבורה. למשל RSVP יודע לשמור על משאבים מסוימים לתעבורה, בעוד ש-DCCP מספק גישה של UDP עם חלקים קטנים יותר המגיעים מעולם ה-TCP.

מאמר זה נכתב על מנת ללמד על אחד מפרוטוקולים אללו, אשר קיבל את השם: Stream Control Transmission Protocol או בשמו המקוצר SCTP.

המאמר בנוי בשני חלקים, כאשר החלק הראשון הוא הסבר המאפשר להבין על מה מדובר, והחלק השני מדבר על כיצד הפרוטוקול בעצם בנוי בפועל - כלומר הסבר של מבנה הפקטה עצמה.

מאמר זה אינו מגיע להיות שלם, ויוצא מנקודת הנחה כי הקורא מעולם לא הכיר או לא נכנס לעובי הקורה בנושא הפרוטוקול, אך כן בעל ידע והבנה ברשתות, ולכן מנסה להישאר בכיוון הלימוד של הפרוטוקול ללא הלימוד של עולם הרשתות (TCP/IP). בנוסף לאמור, אין המאמר מנסה ללמד על תתי פרוטוקולים אשר נמצאים בשימוש, והם אינם ייחודיים לפרוטוקול זה.

חלק ראשון

על רגל אחת, וללא עשיית סדר בנושא, עולם הרשתות של TCP/IP הציג לנו רעיון מאוד מעניין - לבנות דברים בגישה שכל חלק יודע להתמחות במשהו אחד. לחלקים האלו, סיפקו את השם מודל שבע השכבות של OSI.

החלק הראשון ברשימה הוא חיבור פיזי, החלק השני ברשימה הוא כרטיס רשת, השלישי היא רשת ה-IP הרביעית זה כיצד המידע יעבור אלינו עד אשר מגיעים לחלק השביעי שהוא המידע שהתוכנה הסופית צריכה לקחת ולממש.

יחסית בהתחלה, נוצרו שני פרוטוקולים אשר מעבירים את המידע, אחד בשם TCP, אשר יודע לנהל את כל נושא התעבורה ברמה שלו, והשני הוא UDP, אשר מעביר את המידע בלי יותר מידי לבדוק דברים, ובכך מאפשר תעבורה מהירה יותר, אך יש איתו לא מעט בעיות, לדוגמא - שאת הבדיקות צריך להיעשות ברמת התוכנה הסופית (שכבה מספר 7), ולא ברמת התעבורה.

אילו בדיקות? למשל האם הסדר הגיע נכון, האם הגיעו כל החלקים, או האם התקשורת הגיע בכלל ליעד. כל פרוטוקול ברמה הגבוהה ביותר, צריך להכיל בתוכו מידע אשר יסייע לנושא, בעוד שב-TCP, המידע הזה נמצא בשכבה מספר 4, הכוללת גם ניהול פעולות congestion - כלומר כמה פעמים לחזור על בקשה עד אשר נכריז עליה שהיא לא הצליחה.

לרוב שתי גישות אלו מספיקות, אך, ככול שהזמן עבר, נמצאו עוד מקרי קצה אשר בהם צריכים לשלב תכונות שונות בין TCP לבין UDP, ונוצרו עוד פרוטוקולים חדשים, אשר מנסים לספק מענה שכזה.

קיצורים ומונחי יסוד

כאשר מדברים על SCTP ישנם מספר קיצורים ומונחי יסוד, אשר חלקים גם מופיעים כאן במאמר:

מונח	פירוש
MAC	Message Authentication Code - גישה לבדיקת תקינות המבוססת על פונקציות Hash קריפטוגרפיות (בגישה דומה ל-HMAC - RFC2104 אך לרוב לא זהה) באמצעות מפתח פרטי - סימטרי - כלומר משותף לכל הצדדים הנוגעים בדבר, על מנת לוודא את אמינות המידע בין הצדדים.
RTO	Retransmission Timeout
RTT	Round-Trip Time
SCTP	Stream Control Transmission Protocol
RTTVAR	Round-Trip Time Variation
SRTT	Smoothed RTT
TCB	Transmission Control Block
TLV	Type-Length-Value coding format
TSN	Transmission Sequence Number
ULP	Upper-Layer Protocol
User	כאשר מדברים על משתמש, מדברים על התוכנה (שכבה שבע) - User Application .
Message	הודעה מצד המשתמש המוגשת ל-SCTP על ידי ה-ULP.
Multiplexing	איגוד מספר פעולות שונות תחת אותה קורת גג.
Bundling	פעולה אופציונלית של Multiplexing אשר יכולה להכיל יותר מהודעת המשתמש אחת באותה פקטת SCTP.
Congestion	משתנה האחראי על הגבלת גודל המידע בבתים שניתן לשלוח ליעד מסוים לפני שמקבלים אישור.
Chunk	"חתיכה", יחידה של מידע בתוך פקטת SCTP אשר מורכבת מ"ראש" המתאר את החתיכה והתוכן של החתיכה (Payload).
Cumulative TSN Ack Point	מספר הסופי לחתיכה האחרונה, והגיע עליה אישור.

Inactive destination	יעד אשר מכיל שגיאות, או אינו זמין לקבל הודעות.
Network Byte Order	השימוש ב-Big Endian - שהבתים החשובים (MSB) נמצאים בהתחלה.
Ordered Message	הודעת משתמש אשר נשלחה לפי הסדר, עם התחשבות לכל הודעות העבר של המשתמש שנשלחו.
Outstanding TSN	מספר TSN אשר שייך לחתיכה אשר נשלחה ליעד, אך טרם התקבל עליה אישור.
Path	הנתיב אשר נלקח על ידי פקטת SCTP אשר נשלחה ליעד מסוים לפי כתובת. כאשר, שליחה ליעד אחר, אינה מבטיחה נתיב שונה.
PMTU	נתיב MTU, היכולת להבטיח נתיב לפי גודל ה-MTU.
rwnd	Receiver Window - משתנה של SCTP המחזיק בתוכו את החישוב העדכני ביותר לחלון השליחה של היעד, ונשמר בבתים. זה מאפשר לשולח הבנה של הגודל הקיים אצל היעד המקבל את הבקשה.
Primary Path	הנתיב המרכזי וברירת המחדל ליעד ולמקור ההודעות. ההגדרה מחזיקה בתוכה את כתובת המקור, ולא רק את כתובת היעד, היות והמימוש ירצה לרוב לדעת כיצד לחזור למקור על ידי שליטה בנתיב התעבורה עצמו.

ישנם עוד מונחים, אשר יכנסו בחלק הבא של המאמר.

מהו SCTP?

הפרוטוקול של SCTP הוגדר על ידי IETF בשנת 2000 על מנת לאפשר העברת נתוני Signaling System 7 או SS7 בקיצור. SS7 משתמש בחיבור קווי, ובעל תשתית עצמאית משלו, ואינו קשור לרשת TCP/IP. הפרוטוקול החדש עבור SS7 של IETF קיבל את השם SIGTAN, ובשביל להצליח ולספק לו מענה על שלל המרכיבים שלו, בעצם יצרו את SCTP.

השימושים ב-SS7 הם עבור עולם הטלפוניה. כאשר אחד השימושים העיקריים שלו, הוא שליחה וקבלה של SMS, ושימוש נפוץ נוסף הוא התמודדת עם טלפוניה בין מרכזיות אשר נמצאות בסביבות שונות, תחת אותה תשתית.

כאשר צריכים לממש את SS7 על גבי רשת ה-IP, ישנם הרבה סוגים שונים של בעיות אשר היו צריכים לפתור. הנה רשימה חלקית של הבעיות שהיו צריכים לפתור:

- כיצד להעביר הודעות?
- כיצד להעביר איתות?
- כיצד ניתן לעבוד עם מספר שרתים במקביל?
- כיצד ניתן להבטיח איכות תעבורה?
- כיצד ניתן לדחוס כמה שיותר מידע בבקשה אחת?
- כיצד ניתן להבטיח כי חלק מהמידע ידווח על שהגיע ליעד?
- כיצד להתעלם ממידע אשר אין צורך לדעת האם הוא הגיע?

על מנת לפתור חלק גדול מבעיות אלו, ניתן היה לשלב תעבורה שחלקה הוא TCP וחלקה הוא UDP, אך זוהי גישה מאוד בזבזנית, אשר אינה מאפשרת לדחוס כמה שיותר מידע בבקשה אחת (multiplexing). זו הסיבה שבעצם נוצר SCTP.

הבדל נוסף גדול ומשמעותי של SCTP אל מול TCP הוא בכך שTCP עובד על בסיס stream של מידע, בעוד ש-SCTP מבוסס הודעות (message).

מה ההבדל בין Stream לבין Message?

תעבורה על בסיס הודעות מדברת על כך שהודעות זה משהו שיודע להתחלק במידע לחלקים קטנים, או קבוצות של מידע ובכך מי שמקבל את המידע יודע מתי ההתחלה, אמצע והסוף של ההודעה. כאשר מדובר ב-stream, יש צורך לשלוח רצף של מידע, ובכך הרצף עצמו קובע את ההתחלה, אמצע וסוף של המידע.

למרות שב-TCP ניתן לדעת לפי כל פקטה מה הסדר, אי אפשר לדעת בפרוטוקול את הסדר של המידע, אלא רק את סדר השליחה שלו. כלומר אין מידע לגבי סדר התוכן, אלא רק לגבי מצב השליחה עצמו, בעוד שעל בסיס הודעות, כדוגמת SCTP, ניתן לדעת עוד ברמת הפרוטוקול את סדר המידע עצמו.

הפרוטוקול של SCTP יודע להכיל בתוכו מידע של מדיה למשל, בגישה של real-time, כאשר במידה והמידע לא הגיע, ממשיכים הלאה במקום לנסות ולקבל אותו בכל מחיר. אך בנוסף לכך, הפרוטוקול יודע להעביר הודעות אשר חשוב מאוד שיגיעו כמו שצריך, וחשוב לדעת להתעכב עליהם עד אשר יגיעו כמו שצריך.

אחרי התכנון, במימוש של SCTP ישנן התכונות הבאות:

- תמיכה ב-multihoming.
- יכולת עבודה ב"חתיכות" (chunks) בערוצים שונים, ללא צורך בתמיכת HOL (Head of Line).
- שליטה כיצד להתנהג עם מידע:
  - האם יהיה עליו פידבק.
  - האם לנסות ולשלוח אותו הלאה.
  - האם להתעלם מהמידע.
- הבטחת איכות תעבורה על ידי שימוש באלגוריתמים של Path Selection.
- יכולת להתמודד עם Jumbo Frames, כאשר מדובר ב-MTU גדול במיוחד (מעל 1,500 בתים).
- כלים ומימושים לסיוע באבטחת המידע ומניעת התקפות שונות, כדוגמת Replay attack.

MultiHoming

כאשר מדברים על Multihoming, מדברים על כך שמחפשים שרידות של תעבורה, על ידי שימוש ביותר משרת אחד בשביל להעביר את המידע לנקודה אחת, במידה ואחד מהשרתים אינו מבצע את הפעולה שלו בצורה טובה, או אינו נגיש.

שימוש נוסף ב-multihoming הוא שימוש בנתיבי רשת באיכות שונה על מנת להבטיח תעבורה טובה. למשל כמות קפיצות (hop) בין שרתים עד אשר המידע מגיע ליעד, מהירות התעבורה בין כל יעד וכיוב'...

בגדול מאוד, הרעיון של Multihoming אומר כי המקור ו/או היעד מכילים יותר משרת אחד, ובמידה והשרת הראשי או הראשון אינם מגיבים, המידע יתקבל לשרת השני. כמובן שניתן שיהיו יותר משני שרתים. ובנוסף, ישנם מצבים בהם אחד מהצדדים אינו תומך בגישה של Multihoming ועדיין הצד שיודע לתמוך, ידע להתמודד עם המצב.

כאשר SCTP מדבר על Multihoming, מדובר למעשה במצב בו מספקים מספר יעדים. כאשר תעבורת השרת המוציא מקבל את את הבקשה ורואה את היעדים השונים, חלקה העליון של הבקשה (ULP) כיעד הראשי להודעה. בברירת המחדל בנושא, תעבורת השרת תמיד תבחר יעד ראשי אחד, אלא אם מוגדר ברמת ה-SCTP יעד ברור.

השרת אשר מוציא את הבקשה תמיד צריך לדעת להעביר את הנתונים לשרת הנבחר, דבר הכולל גם הודעות תעבורה כדוגמת ACK-ים שונים, עליהם ידובר בהמשך. עניין זה כולל גם את היעד, אשר צריך לענות חזרה למקור, כאשר גם הוא יכול במקור להיות בעל תמיכה ב-multihoming.

כאשר יש מספר הודעות ממקורות שונים לבקשה מחולקת, הצד המקבל, צריך לענות לשולח, ולא לשרת הראשי כתגובה להודעות עצמן.

כאשר יש ניסיון שליחה ליעד אשר אינו פעיל, בין אם בשל שגיאות, ובין אם באמצע התקשורת, יהיה צורך להחליף את השרת אשר אינו באוויר כרגע, ותהיה החלפה של השרת על ידי ניסיון לשלוח ליעד חלופי, ואם זה נכשל, יהיה דיווח על השגיאה הלאה.

דיווח השגיאה יתבצע רק כאשר יגיע timeout או חוסר יכולת למצוא יעד פעיל.

חתיכות

ב-SCTP פקטה מורכבת מ"ראש" או הכרזה מוכרת וידועה (common header) ומ"חתיכת" מידע (chunk).

הפרוטוקול יודע להכיל בתוכו מספר חלקים שכאלו, אשר מורכבים מהכרזה מוכרת וידועה ו"חתיכת מידע". הגבול לכמות החתיכות, הוא גודל ה-MTU.

חלק מחתיכות המידע חייבות להגיע שלמות, וחלק לא, בהתאם להרבה מאוד הגדרות בפרוטוקול. המידע אשר חייב להגיע בצורה שלמה, הוא למעשה הודעות שמורות של הפרוטוקול, למשל הודעה על התחלת השליחה (בסגנון לחיצת שלוש הידיים של TCP) בשם INIT, מנגנון של "עוגיות" (אדבר על כך בחלק אבטחת המידע), שליחת heartbeat, שנוי ה-window לטיפול ב-congestion ועוד. המידע הזה חייב להגיע בשלמותו בפעם אחת, ולכן אינו יהיה בחלקים.

גם מידע המגיע בחלקים, יכול להיות מוגדר כ"שלם". זה יקרה כאשר שולחים מידע על פעולות של הבקשה עצמה, כדוגמת פעולת ה-INIT למשל, והתשובה שלה שתהיה INIT ACK.

סוגי הודעות/חתיכות:

קוד	סוג הודעה	הסבר
0	DATA	המידע עצמו (payload)
1	INIT	התחלת שליחה
2	INIT ACK	אישור קבלת השליחה
3	SACK	אישור חלקי בדבר קבלת השליחה
4	HEARTBEAT	בקשת HEARTBEAT
5	HEARTBEAT ACK	אישור על בקשת ה-HEARTBEAT
6	ABORT	בטל פעולות
7	SHUTDOWN	בקשה לסיום מסודר של תשדורת
8	SHUTDOWN ACK	אישור על קבלת SHUTDOWN
9	ERROR	שגיאת פעילות (operation error) - אינה נחשבת למשהו שלא ניתן להמשיך דרכו, בניגוד ל-ABORT, אך יכולה לבוא בשימוש גם של ABORT לשם כך.
10	COOKIE ECHO	התחלה של עוגיות (ארחיב על הנושא בחלק אבטחת המידע)
11	COOKIE ACK	אישור קבלת עוגייה
12	ECNE	שמור עבור הכרזת congestion
13	CWR	שמור עבור הכרזת הפחתת חלון שליחה
14	SHUTDOWN COMPLETE	סיום פעולת הכיבוי
15-62	פנוי	
63	שמור	
64-126	פנוי	
127	שמור	
128-190	פנוי	
191	שמור	
192-254	פנוי	
255	שמור	

כאשר מדובר בהודעות מערכת INIT, INIT ACK ו-SHUTDOWN COMPLETE, החתיכות חייבות להגיע לבד, ואסור להן להגיע עם עוד חתיכות בנוסף.

כאשר המידע (Payload) אינו יכול להיכנס לפקטה בודדת, הפקטה יכולה ליצור מספר חלקים של אותו המידע ובכך ליצור מידע מחולק (fragmented).

ב-SCTP שולחים כאמור את מידע המשתמש בחתיכות, וניתן להגדיר התנהגות של חתיכה מסוימת. למשל ניתן להגיד כי יש להתעלם ממנה, או לשלוח עליה פידבק, ובכך בתוך ה-multiplexing, ניתן להגדיר התנהגות של כל חתיכה. בנוסף, המידע על החתיכה מכיל מספר סידורי עולה (TSN), מה המיקום שלה - למשל התחלה, סוף או בגלל שאין סימון שכזה, יודעים לפי מספר סידורי מה המיקום שלה באמצע.

Path Selection

הבטחת מסלול התעבורה, בניגוד ל-MPLS או DiffSrv, אשר מתבצעים ברמת ה-IP ואף ברמת החיבור, כאן הבטחת המסלול מתבצעת בשכבה גבוהה יותר. המימוש של SCTP מבצע Path MTU Discovery אשר מוגדר ב-rfc 1191 ו-rfc 1981. הרעיון הוא ללכת רק בכיוון שבו ה-MTU הוא בגודל מסוים, ואינו חורג מהגודל הזה.

ההבדלים של SCTP מ-rfc 1191 הם:

1. SCTP תומך ביכולת להחזיק מספר כתובות IP כיעד (multihoming כאמור), וצריך לשמר ניתוב עבור כל כתובת IP בנפרד.
2. השולח צריך לבצע מעקב אחר PMTU המשויך לבקשה עם ה-PMTU הנמוך ביותר עבור כל החיבורים אל היעד. כאשר מחלקים את התעבורה לחלקים שונים (fragmentation), השיוך הזה של ה-PMTU יהיה בשימוש על מנת לחשב כל חלק. גישה זו, תאפשר את היכולת לבצע שליחה מחדש ליעד אחר ללא צורך בביצוע חילוק על בסיס כתובת IP.

עוגיות ואבטחת מידע

לפרוטוקול SCTP ישנם מספר מנגנונים המאפשרים לו לדעת האם המידע שהגיע אמין, האם להתייחס אליו בכלל והאם בכלל היה צריך להגיע אותו מידע. דבר ראשון, עבור אמינות המידע, לכל חתיכה יש checksum אשר פעם היה מבוצע על ידי adler32, אך קיבל שכתוב וב-rfc4960, אשר נכון לכתיבת מאמר זה הוא מימוש הפרוטוקול העדכני, החישוב נעשה על ידי CRC32c.

כאשר יש צורך לעבוד עם תקשורת מוצפנת, RFC 3436 מדבר על המימוש של TLS עם SCTP, ובנוסף יש תמיכה ב-IPSEC עבור הפרוטוקול ב-RFC 3554. בנוסף ישנו hash קריפטוגרפי (MAC) אשר יוצר לרוב

מפתח סימטרי באמצעות יצירת מספר פסודו רנדומלי, כפי שמתואר ב-RFC4086, ומימוש מאוד דומה ל-HMAC המתואר ב-RFC2104. השימוש ב-MAC, מתבצע בשביל למנוע מתקפות DoS על הבקשה, וההגנה נעשת על ידי שימוש ב"עוגיות".

עוגיות ב-SCTP הן עולם שלם, אשר נלקח במקור מ-RFC2522 - המוכר גם בשם Photuris Session-Key Managment Protocol, אשר מאפשר לנהל session בצורה מאובטחת ברמת תעבורה ורמת ה-IP.

מהי עוגיית Photuris Session-Key?

בקצרה (יחסית) - עוגייה יוצרת session קצר מועד מבוסס מפתחות בין שני צדדים, ללא החלפת המפתחות בניהם ברשת האינטרנט. המפתחות האלו מחליפים בצורה ישירה את הצורך בסמאות למשל, אשר לרוב מוגדרות בשביל לנהל תעבורה מאובטחת. הפרוטוקול משתמש באותם מפתחות פרטיים (shared secret) שהוגדו בעבר לצורך זיהוי הצדדים. העוגייה יוצרת סוג של הגנה מפני הצפת בקשות מכתובות IP אשר אינן מוכרות, או פורטים שנפתחים ב-UDP, על ידי כל שכל צד מעביר "עוגייה" לצד השני.

לאחר שליחת העוגייה, רשימה של מוסכמות על הדיאלוג מוחזרת לשולח, על מנת לחשב את המפתח המשותף. ערך כלשהו מוחלט על ידי הצדדים, ונוצר מפתח פרטי לתקשורת בין הצדדים. כל צד מעביר לצד השני ערך מספרי כלשהו. הערכים האלו הם בשימוש על מנת לחשב את המפתח המשותף בניהם. הצד המקבל של הערך, נשאר stateless עד אשר המפתח המשותף נוצר.

בנוסף, ישנם עוד תכונות משותפות העוברות בין הצדדים, על מנת לסייע ביצירת שיח בעל תכונות מאובטחות. לאחר החלפת המידע, מערכת החלפת המידע מזהה את הצדדים ומוודאת את אמינות המידע אשר נשלחו בפעימה הראשונה והשנייה.

בנוסף, המפתח הפרטי מאפשר ליצור מפתחות נפרדים עבור ה-session לכל כיוון, אשר אחראים על הזדהות "רגילה" ומקובלת גם ללא השימוש בעוגיות.

עוגיית STCP

על מנת להתחיל ולשלוח מידע, SCTP חייב ליצור פעולה בשם INIT, אשר מודיעה על התחלת פעולה. רק כאשר פעולה זו מסתיימת, נוצרת הכרה בין הצדדים. צד המשתמש חייב בכל קצה להשתמש ב- Associate Primitive על מנת לאתחל את השיוך לקצה השני של SCTP. כאשר השיוך מסתיים, נפתח חיבור זרימה unidirectional להעברת המידע של כל הצדדים. פעולות אלו דורשות הגנה, היות והן רגישות מאוד. ההגנה מתבצעת על ידי שימוש בעוגיות.

תהליך האתחול מכיל בתוכו את הצעדים הבאים (נקודה A מנסה להגיע לנקודה Z ונקודה Z צריכה לאשר את הבקשה):

1. נקודה A שולחת חתיכה של INIT אל עבר נקודה Z. בתוך ה-INIT, A חייב לספק טאג הזדהות (verification tag), שנקרא TAG_A לצורך ההדגמה, בתוך המיקום המתאים לטאגים (עליהם אדבר בחלק הבא). הטאג חייב להיות מספר רנדומאלי בין 1 ל-4294967295. כאשר השליחה של ה-INIT התבצעה, נקודה A מתחילה טיימר לטאג, ויוצרת מצב של COOKIE-WAIT.

2. נקודה Z צריכה להגיב מיד עם פעולה מסוג INIT ACK בתוך חתיכה הנשלחת חזרה. כתובת החזרה של INIT ACK חייבת להיות כתובת המקור בפעולת השליחה שאליה ה-ACK INIT מגיב. בתגובה, בנוסף למילוי הפרמטרים השונים, Z חייב להכיל את Tag_A, אך גם ליצור Tag_Z משל עצמו בנקודת האיתחול. בנוסף, Z חייב ליצור ולשלוח ביחד עם ה-INIT ACK עוד עוגייה. כאשר Z שולח את ה- INIT ACK עם כל הפרמטרים הנדרשים, אסור ל-Z ליצור משאבים כלשהם או להחזיק איזשהו מצב עבור השיוך שבוצע, אחרת Z פגיע להתקפות.

3. כאשר A מקבל את ה-INIT ACK מ-Z, הנקודה חייבת לעצור את הטיימר T1-init ויצא ממצב של COOKIE-WAIT. לאחר מכן, A ישלח את מצב העוגיה שהוא קיבל ב-INIT ACK בפעולה של COOKIE ECHO, ויפעיל טיימר חדש ואז יכנס למצב של COOKIE-ECHOED. החתיכה של COOKIE-ECHO יכולה להיות משולבת עם עוד חתיכות נוספות של מידע, אבל היא חייבת להיות החתיכה הראשונה בסדרה, עד אשר יתקבל COOKIE ACK על ידי השולח, כאשר מידע נוסף לא ישלח עד אז על ידי היעד.

4. בקבלת המידע של COOKIE-ECHO, צד Z יענה עם COOKIE ACK לאחר בנייה של TCB ולעבור למצב של ESTABLISHED. COOKIE ACK יכולה להגיע כחלק מחתיכות המחכות לשליחה ו/או חתיכה מסוג SACK, אבל COOKIE ACK חייבת להיות החתיכה הראשונה בפקטה. המימוש יכול לבחור אם הוא יבצע שליחה בעת קבלת ההודעה של COOKIE ECHO תקני או מאוחר יותר.

5. בעת קבלת COOKIE ACK, צד A יעביר את המצב מ-COOKIE-ECHOED למצב של ESTABLISHED, צד A גם יעצור את הטיימר T1-Cookie בנוסף. צד A יבחר גם אם לעדכן את צד ה-ULP על הצלחת החיבור, באמצעות הודעת Up.

חשוב להדגיש כי החתיכות של INIT ו-INIT ACK אינן מורשות להגיע ביחד עם עוד חתיכות וחייבות להיות בפקטה משל עצמן.

תרשים הבא מספק הצגה של פעולת ה-INIT:

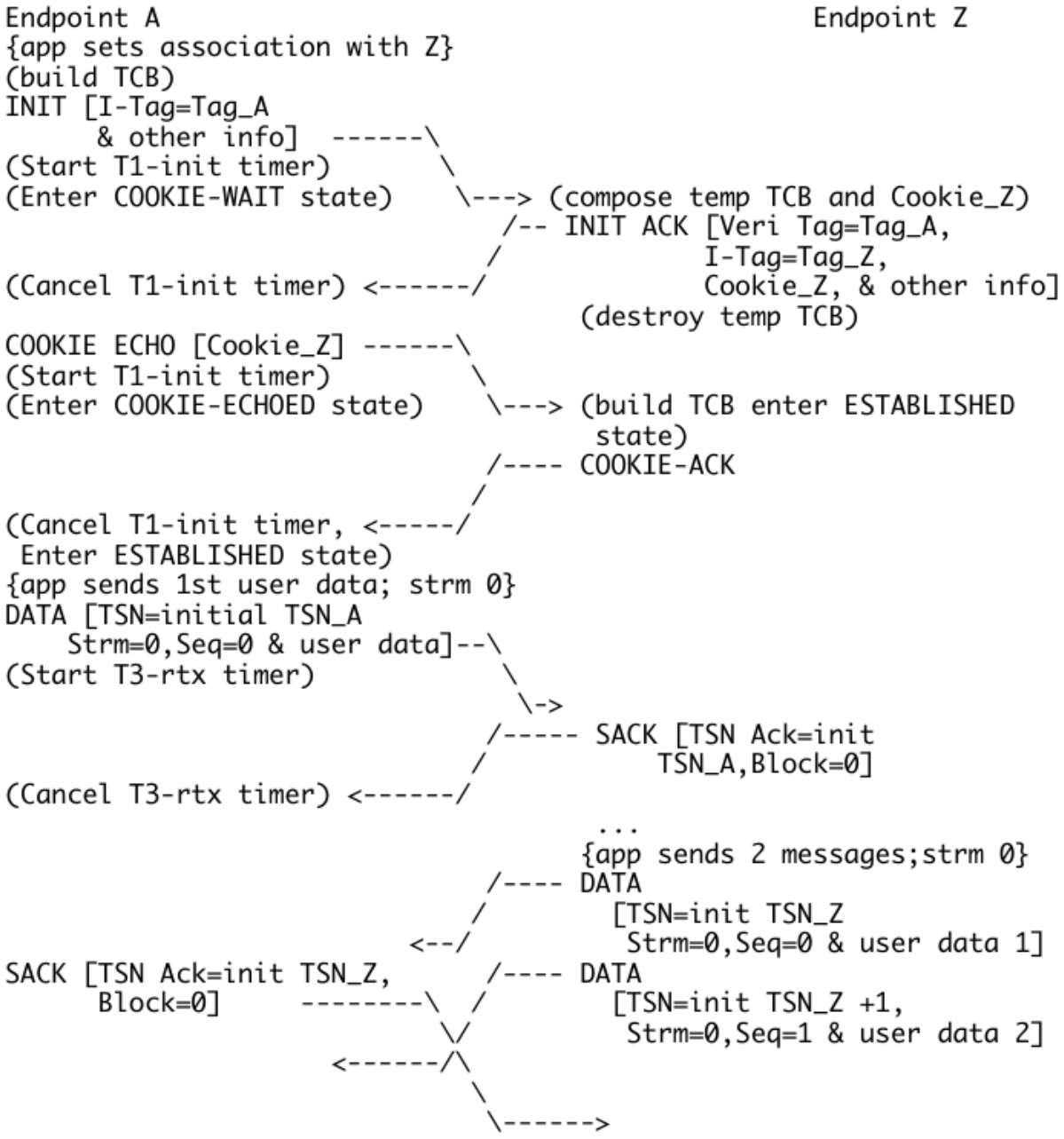

Figure 4: INITIATION Example

[נלקח מ-rfc4960]

סיכום

פרוטוקול SCTP הוא אחד הפרוטוקולים המעניינים ביותר לתעבורה הקיימים כיום לדעתי, ומאפשר לבצע המון פעולות שונות ומשונות. בחלק זה, נגעתי כאן רק בקצה המזלג ברעיון, אשר מספק הבנה לגבי יכולות הפרוטוקול, אך פחות על מימושו והתמודדות עם נושאים שונים.

בזכות כך שהפרוטוקול הוא יחסית פרוטוקול חדש, בזמן תכנונו נלמד הרבה מאוד מבעיות שונות הקיימות ברשת האינטרנט, והתכנון מנסה לספק לבעיות אלו מענה באמצעות כלים שונים. למשל לספק יכולת של High Availability ו-DR באמצעות Multihoming, היכולת לקבוע נתיב תעבורה באמצעות Path MTU Discovery ועד לפתרון עבור בעיות אבטחה כדוגמת DoS ו-Replay Attack באמצעות עוגיות.

אך החלק החשוב ביותר הוא אספקת היכולת לבצע Multiplexing למידע, ובכך לשלוח כמה שיותר מידע של פעולות שונות בין שרתים שונים אשר מצפים להודעה. יכולת זו, יכולה להגיע כאמצעי זרימה, וכאמצעי של הודעה.

בחלק הבא אדבר יותר על המימוש, אך גם הוא יהיה קצר מאוד, ולא יחליף לימוד מקיף יותר של הפרוטוקול.

ביבליוגרפיה

- Stream Control Transmission Protocol (RFC4960)
- Path MTU Discovery - IPv4 (RFC1191)
- Path MTU Discovery for IP version 6 (RFC1981)
- HMAC: Keyed-Hashing for Message Authentication (RFC2104)
- Transport Layer Security over Stream Control Transmission Protocol (RFC3436)
- On the Use of Stream Control Transmission Protocol (SCTP) with IPsec (RFC3554)
- Security Attacks Found Against the Stream Control Transmission Protocol (SCTP) Current Countermeasures (RFC5062)
- Randomness Requirements for Security (RFC4086)
- Photuris: Session-Key Management Protocol (RFC2522)
- Stream Control Transmission Protocol