RTP - כיצד מזרימים מדיה בזמן אמת

מאת שחף גרגיר ודביר גולן

מבוא

יישומים רבים משתמשים בשיחות והעברת מידע בזמן אמת דרך האינטרנט, כדוגמת WhatsApp, Skype שבהן מבוצעות בעזרת האינטרנט שיחות וידאו ורגילות ועוד (חשוב להדגיש כי אפליקציות אלו הן דוגמאות להמחשה של השימוש בהעברת מידע בזמן אמת ואינן עושות שימוש בפרוטוקולים המתוארים במאמר).

מאמר זה, יעסוק בפרוטוקול Real-time Transport Protocol, או בקיצור RTP, שלרוב פועל מעל UDP בפורטים 16,384 - 32,767[1], חשוב לציין כי קיימים מקרים בהם נעשה שימוש גם ב-TCP, כאשר אמינות המידע חשובה. לדוגמה בהעברת קבצים בזמן אמת.

כמו כן, נסביר אודות הפרוטוקול, האופן בו הוא מתבצע, מבנה חבילות המידע (ה-Headers והערכים המופיעים בו). בנוסף, נציג דוגמאות למימושים שונים של הפרוטוקול ונתאר בקצרה כיצד מצפינים את המידע המועבר וכיצד מבטיחים את מהימנות המידע. יתר על כן, נציג שני רכיבים: Mixers ו-Translators, המשמשים להעברת מידע ב-RTP.

במהלך המאמר ניתקל רבות בפרוטוקול Real-time Transport Control Protocol, RTCP, הקרוב מאוד לפרוטוקול RTP, לכן נסביר גם אודותיו, ונסביר בקצרה על ניתוח RTP באמצעות התוכנה Wireshark.

קיים מאמר נוסף ב-DigitalWhisper בגיליון מספר 60, בשם "הצפנת VoIP למתחילים" המתעסק ב-RTP אך הוא מתמקד בעיקר בהצפנת חבילות המידע, ואנו במאמר זה מרחיבים ומתעסקים בכל הפרוטוקול, כפי שציינו למעלה. העמקה בנושא ההצפנה ניתן לקרוא במאמר לעיל.

[1] נלקח מ-https://www.voip-info.org/rtp-ports

מושגים חשובים

לפני שנתחיל להסביר אודות הפרוטוקול וכל מה שמלווה בכך, נציין מספר מושגים שכדאי להכיר ולהבין לפני קריאת המאמר, ונסביר על כל אחד מהם בקצרה:

- **RTP Payload** - המידע המועבר על ידי RTP בחבילת המידע. למשל, דגימות אודיו או וידאו דחוס.

- **עיקרון קצה לקצה** - (End-to-End Principle) עיקרון עיצובי זה קובע כי, פעולות מסוימות עם מטרות ספציפיות ליישום, אמורות לקרות אצל משתמש הקצה ולא בצומת אמצעי כלשהו ברשת (למשל בנתב וכו'), כל עוד ניתן לממש אותה אצל המשתמש הנ"ל בצורה מלאה ונכונה. ברור כי פחות משתלם להוסיף פונקציות לרשת, ובפרט במקרים בהם משתמשי הקצה צריכים לממש אותן בכל מקרה. עיקרון זה ממומש לרוב על מנת לשמור על רמת סיבוכיות נמוכה של הרשת ואופן הפעולה שלה, וזמן ביצוע "סביר יחסית" לכלל המשתמשים.

- **ריצוד** - באלקטרוניקה, ריצוד הוא סטייה ממחזוריות של אותות, הגורם ל"הפרעה" מסוימת ביישום. ריצוד יכול לגרום לצג התצוגה להבהב, להשפיע על הביצועים של מעבדים במחשבים אישיים, להציג השפעות לא רצויות אחרות באותות שמע, ולגרום לאובדן נתונים המועברים בין התקני רשת, וכדומה.

- **חבילות מידע אבודות (Packet Loss)** - חבילות מידע נתונים אשר "מטיילות" ברשת מחשבים ואינן מגיעות ליעדן (הן לא מסתובבות לעד ברשת האינטרנט בעקבות ה-TTL בפרוטוקול IP, שהוא מספר מוגדר המציין כמות של רכיבים רשתיים בהם חבילת המידע יכולה לעבור. לאחר שה-TTL מגיע ל-0, חבילת המידע "נופלת"). אובדן זה נגרם עקב שגיאות בהעברת הנתונים. הפסד חבילות מידע נמדד לפי אחוז חבילות המידע שאבדו ביחס חבילות מידע שנשלחו. TCP למשל, מגלה אובדן כזה ומבצע שידורים חוזרים של חבילות המידע כדי להבטיח העברת הודעות מהימנות. בהקשר של הפרוטוקול שלנו, Real-Time Transport Protocol, לרוב אין שימוש ב-TCP מכיוון שהמידע צריך לעבור בזמן קצר ככל הניתן, ולא רצוי "לבזבז" זמן על שליחת חבילות המידע הפגומות/האבודות מחדש, לכן RTP פועל מעל UDP. בהמשך נראה כי ניתן לסדר את הבעיות נ"ל באמצעות אלגוריתם לתיקון שגיאות.

- **Error Concealment** - הסתרת שגיאות הינה טכניקה המשמשת בעיבוד אותות שמטרתה למזער את הידרדרות האותות הנגרמת בעקבות נתונים חסרים, כלומר אובדן חבילות מידע. אות הוא בעצם מסר שנשלח ממשדר למקלט במספר חבילות מידע קטנות.

- **RTP Session** - קבוצת משתמשים המעבירה בינה לבין עצמה מידע בעזרת RTP, וכל משתמש מדווח לכל המשתמשים האחרים על המידע שקיבל בעזרת RTCP (החלק האחראי על השליטה והבקרה בפרוטוקול). כל משתתף מגדיר Transport Address עבור כל Session שהוא משתתף בו. לכל משתתף ב-Session מסוים יש מספר מזהה שונה, על מנת להבדיל בין מקורות שונים. כל משתמש יכול להיות חלק מכמה Sessions בו זמנית. לרוב, בתקשורת מולטימדיה כל סוג מידע בתקשורת

מועבר ב-RTP Session נפרד ומשתמש בחבילות מידע RTCP משלו. למשל, בשיחות וידאו, לרוב יפתח Session לשידור הקול ו-Session אחר לשידור הוידאו.

- **Mixer** - היא מערכת שמקבלת חבילות מידע RTP ממקור אחד או יותר, כלומר זרמי מידע, ומאחדת את חבילות מידע אלו לחבילת מידע של RTP אחת, כלומר לזרם מידע אחד, ושולחת אותה. כמו כן, ייתכן שינוי בשיטת קידוד המידע. למשל, קידוד המידע באופן חסכוני לשידור על קווים בעלי קיבולת נמוכה, או איחוד של מספר תמונות של משתתפים בשיחה לכדי תמונה אחת ובכך לחסוך במידע כפול. ה-Mixer מציב מספרי חבילות מידע וזמנים חדשים בכל חבילות מידע ה-RTP החדשות שנוצרות. כלומר, ל"תחנת" Mixer יש SSRC (קיצור של Synchronization source Identifier) משלה המוכנס לשדה המתאים בחבילות מידע ה-RTP שהיא יוצרת (משפט זה יתבהר בהמשך המאמר).

- **Translator** - זוהי מערכת תיווך שמטרתה להעביר חבילות מידע RTP ליעדן מבלי לשנות את שדה ה-SSRC של חבילות המידע. למשל, השינוי בחבילות מידע יכול להיות בפורמט המידע. דוגמה לשימוש ברכיב Translator: נניח שתחנה A שולחת חבילות מידע RTP לתחנה B. חבילות מידע ה-RTP מכילות מידע וידאו בפורמט 2-MPEG (לינק לקריאה נוספת). על מנת לשדר באיכות הנדרשת בפורמט זה דרוש רוחב פס גדול יותר ממה שהקו בין A ל-B מסוגל לספק. נניח שבין A ל-T (שהיא תחנת ה-Translator) ישנו רוחב פס מספיק לשידור וידאו בפורמט 2-MPEG. במקרה זה A יכולה לשדר את המידע ל-T, ו-T תמיר את המידע לפורמט MPEG-4 (לינק לקריאה נוספת), שהוא חסכוני הרבה יותר, ותשלח את המידע ל-B. נשים לב שמאחר ו-T לא שינתה את שדה ה-SSRC בחבילות ה-RTP שקיבלה מ-A, היא בעצם "שקופה" מבחינתה של תחנת B, כלומר B "חושבת" שהמידע הגיע ישירות מתחנה A.

להלן שרטוט המציג את הדוגמה לעיל:

- **End System** - החלק באפליקציה שתפקידו לייצר את המידע שבחבילות מידע ה-RTP, ובכיוון השני לצרוך את המידע בחבילות מידע ה-RTP שהגיעו.

הסבר כללי

פרוטוקול RTP פועל בעיקרון קצה לקצה ומזרים מדיה בזמן אמת. הפרוטוקול מספק אמצעים כדי לפצות על ריצוד וגילוי של חבילות מידע אבודות ולסדר חבילות מידע שהגיעו בסדר שונה משליחתן[2]. פרוטוקול RTP מאפשר העברת מידע למספר יעדים שונים דרך שידורי IP ותומך גם ב-IPv4 וגם ב-IPv6. פרוטוקול RTP נחשב לתקן הראשי והעיקרי להעברת שמע וידאו ברשתות IP, על פי RFC 3550.

תוכנות התומכות בהזרמת מולטימדיה צריכות לבצע שיקולים וחישובים למען רצף ההזרמה למשתמש, לכן, כדי שחבילות מידע המתווספות כל הזמן יגיעו בזמן המדויק למקום הנכון, אחרות עלולות ללכת לאיבוד. אובדן זה עלול ליצור קטיעות למשך שברירי שניות, אך ניתן לתקן זאת באמצעות אלגוריתם אשר נמצא תחת הקטגוריה "Error Concealment" (ראו דוגמה לאלגוריתם כנ"ל בשם LNCS 2849).

RTP ניתן למימוש גם באמצעות פרוטוקול TCP, אך זה לא נפוץ מכיוון שבאפליקציות הבנויות על TCP, יש עדיפות גבוהה של אמינות לעומת זמן, מה שעלול לשבש או לעוות את המידע (כלומר, המידע אשר יוצג למשתמש יהיה לא רציף, ולכן יהיו עיוותים ושיבושים במידע הכולל המועבר), לכן רוב היישומים המשתמשים בפרוטוקול RTP, בנויים על ה-UDP.

RTP פותח על ידי קבוצת הוידאו/שמע של (Internet Engineering Task Force) IETF, ה-RTP ממומש בשיתוף עם פרוטוקולים נוספים כמו ה-H.323[3] ו-RTSP[4]. פרוטוקול RTP עובד בשיתוף פרוטוקול נוסף, RTCP, אשר הוא החלק האחראי על בקרה, שתפקידיו לשלוח פקודות או לספק מידע על המשתתפים בשיחה, ועל השליטה הכללית.

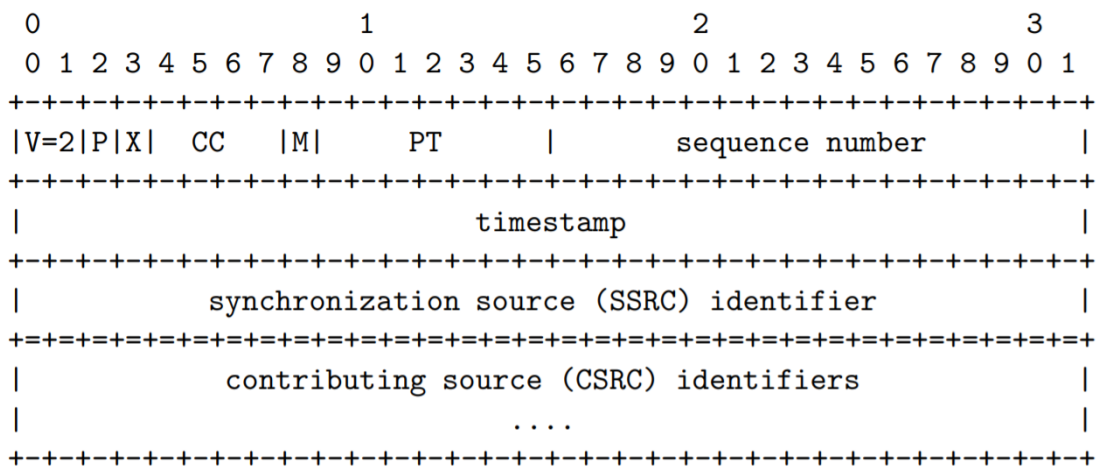
פרוטוקול ה-**RTP** מכיל נתונים בזמן אמת ולכן המידע המסופק על ידי פרוטוקול זה עובר תהליך של החתמה (Time-stamping), בה מצויין הזמן המדויק על מנת ליצור סנכרון מקסימלי. בדומה לחבילות מידע אחרות, גם חבילות מידע ה-RTP הן בעלות מספר רצף (Sequence Numbers), שנועד לזיהוי חבילות מידע אבודות, סידור וארגון מחדש של כל חבילות המידע ביחד כאשר הן מגיעות ליעד. כמו כן, חבילות מידע אלו הן בעלות Payload Format ("פורמט מטענים"), אשר מציין את הפורמט המקודד של הנתונים.

פרוטוקול ה-**RTCP** (קיצור של **RTP Control Protocol**, RFC 3605) אחראי לאיכות השירות, למשוב ולסינכרון בין זרמי המדיה השונים. יתר על כן, רוחב הפס של ה-RTCP מהווה (בדרך כלל) כחמישה אחוזים[5] בפרוטוקול לעומת ה-RTP.

---

[2] דבר חשוב במיוחד בקרב העברות UDP על רשתותIP
[3] לקריאה נוספת: https://tools.ietf.org/pdf/rfc3508.pdf
[4] לקריאה נוספת: https://tools.ietf.org/pdf/rfc7826.pdf
[5] לקוח מ: https://en.wikipedia.org/wiki/RTP_Control_Protocol#Protocol_functions

מבנה חבילות המידע



[השרטוט לקוח מ-https://tools.ietf.org/pdf/rfc3550.pdf]

כפי שניתן לראות, בטבלה הנ"ל המחשה טובה יותר של החלקים בכל חבילת מידע. כעת, נסביר על כל חלק וחלק וכיצד הוא משתלב בתפקיד של הפרוטוקול:

- **Version** - מציין באיזו גרסת RTP מדובר.
- **P (שמקורה במילה Padding - ריפוד)** - אם הביט P דולק, חבילת ה-RTP מכילה מידע נוסף בסוף חבילת המידע שאינו חלק ממנה. הבית האחרון במידע הנוסף, מכיל את מספר הבתים של המידע הנוסף, כולל עצמו. שימוש אפשרי ל-Padding הוא לטובת אלגוריתמים להצפנה שדורשים שגודל המידע יהיה כפולה של קבוע מסוים כלשהו (למשל 128 ביטים).
- **Extension - (הרחבה) X** - אם ערכו של הביט X הוא 1, לחבילת המידע יתווסף בלוק נוסף, בלוק ההרחבה, הסיומת, עליו נדון בהמשך.
- **CC** - או בשמו המלא, CSRC Count, מצביע על גודל המערך שנמצא בשדה ה-CSRC (עליו נדון בהמשך), כלומר, מספר ה-CSRC-ים שיופיעו בהמשך חבילת המידע.
- **M** - או בשמו המלא, Marker, הוא הביט שמסמן לאפליקציה על אירוע משמעותי כלשהו שמציינת חבילת המידע בה הוא דולק. כלומר, הוא דגל המסמן חשיבות על המידע המועבר, אם הוא דולק, פירוש הדבר שלמידע זה יש עדיפות מיוחדת ברמת האפליקציה (בדומה לשדה ה-TOS בחבילת IP). יהיה יותר נוח להבין זאת בעזרת דוגמה, למשל, דגל זה דולק כאשר יש תחילה או חזרה אל דיבור על ידי משתתף בשיחת אינטרנט.
- **PT (Payload Type) - סוג המידע** - שדה זה מיועד לזיהוי הפורמט של המידע/השמע שיועבר בהמשך. למשל, עקב בעיות בתקשורת מחליטה אפליקציה כלשהי ששולחת אודיו לשנות את הקידוד לפורמט חסכוני יותר על חשבון איכות. מקבל המידע יתעלם מחבילות מידע RTP שסוג המידע המופיע בשדה זה לא מוכר לו.

- **Sequence_Number** - מכיל מספר סידורי של חבילת RTP, כלומר מייצג את המיקום היחסי של חבילת המידע ב-Session נתון. המספר עולה ב-1 בכל חבילת מידע שנשלחת, ומאפשרת לשמור על סידור חבילות מידע (זה נכון שגם ב-TCP קיים Sequence Number אך מאחר שלרוב הפרוטוקול רץ על UDP לא ניתן להסתמך על כך וחשוב שתהיה לו בקרה עצמית), סדר ניגון המידע שהתקבל ולזהות חבילות מידע שאבדו. המספר שבו מתחילה הספירה של חבילות המידע נבחר באופן אקראי וכך תורם לסיום והפרדת השיחה באופן מלא, לא יוכל להיווצר בלבול עם שיחה חדשה כי היא תתחיל ממספר שונה לגמרי.

- **Timestamp** - בדומה ל-Sequence Number, הערך הראשוני של שדה זה נבחר באופן אקראי ובהמשך הערך עולה ביחס אליו. הערך שנמצא בחלק זה הוא הזמן בו נוצר הבית הראשון בחבילת המידע (כמובן בתוספת הערך האקראי שנבחר לו מלכתחילה), כך שהזמן צריך להיות מדויק ומסונכרן עם ה"זמן האמיתי", האפליקציה בוחרת זמן מדויק שעובר בין כל יחידה ליחידה. למשל, אפשר להגדיר שכל יחידה בשדה היא ms10 (עשר מילי שניות), ולכן המספר המצוין בשדה הוא הזמן שעבר מתחילת הקשר (חבילת המידע הראשונה) ועד לבית הראשון ב-Sequence.

- באותה האפליקציה, ניתן להגדיר זמן שונה המציין עלייה של השדה באחד, אך, צריך לבדוק שניתן לבסוף לסנכרן את השניים ולתאם ביניהם. ישנו שימוש בשדה זה למרות שקיים כבר שדה דומה, ה-Sequence Number, על מנת לדעת מתי חבילת המידע נשלחה ביחס לתחילת השיחה, ולאו דווקא לדעת את סדר השליחה שלה ביחס לחבילות המידע האחרות.

- **SSRC מקור סנכרון - (Synchronization Source)** - השדה המזהה של יוצר חבילת המידע. מספר זה הוא אחד ויחיד בכל חבילת מידע (כלומר, לא מופיעים מספרים של שני session-ים), השדה נבחר באקראיות ומשתמש באלגוריתם שבודק שאין כפילות מזהים בין משתתפי השיחה. גודל השדה הוא 32 ביטים, ובעזרתו יודעים רכיבי הקצה להבדיל בין session-ים שונים.

- **CSRC (Contributing source - מקור תורם)** - זהו מערך בגודל בין 0 עד 15 ביטים, אשר מכיל רשימות של SSRC שנוצרו. במקרה בו נדרש שילוב של מספר מקורות שמע (למשל בשיחת ועידה), שדה זה יכיל את רשימת מזהי ה-SSRC של כל המקורות (שכמותם בהתאם לשדה ה-CC), וכך יאפשר את שילוב השמע בצורה מסונכרנת.

- **Extension Header**[6] - בפרוטוקול RTP, מאופשרות הרחבות ייעודיות שונות. כמו שכתבנו קודם, כאשר הדגל X דולק, פירוש הדבר שנעשה שימוש בהרחבה כלשהי, ובשדה זה מפורטים כל פרטי ההרחבה (רכיבי קצה שלא מכירים הרחבה מסוימת יתעלמו ממנה). ההרחבה מיועדת להעברת מידע שאיננו תקני ושאפליקציות משתמשות בו לצורך ניסוי/בדיקה, תוך שמירת תאימות עם תקן RTP. בלוק ההרחבה יתווסף מיד אחרי השדה SSRC או CSRC (אם קיים), והוא מכיל 3 שדות: שדה בגודל 16 ביט

[6] לקריאה נוספת: https://tools.ietf.org/pdf/rfc8285.pdf

לשימוש האפליקציה המנסה (למשל, ציון סוגים שונים של הרחבות), שדה Length בגודל 16 ביט שמכיל את גודל שדה ה-Extension (ללא השדה הקודם ושדה ה-Length עצמו), ושדה ה-Extension המכיל את המידע ה"ניסיוני".

- CSRC identifiers – שדה נוסף בפרוטוקול, שאינו חייב להופיע בחבילה.

מימושים לפרוטוקול

בשנים האחרונות השימוש בפרוטוקול זה גדל ומשתלב בנושאים רבים בחיים שלנו, כפי שציינו בהתחלה, יישומים רבים משתמשים בשיחות והעברת מידע בזמן אמת דרך האינטרנט (WhatsApp, Skype ועוד...).

פרוטוקול RTP משומש באופן נרחב במערכות מדיה ובידור, הכוללות Streaming (הזרמת מדיה), כגון טלפוניה, יישומי ועידות וידאו, שירותי טלוויזיה ותוכנות לחץ-כדי-לדבר (push-to-talk) המבוססות על האינטרנט.

נוסף על כך, RTP הינו אחד היסודות הטכניים של Voice over Internet Protocol (ובקיצור, VoIP). זהו שם כללי לטכנולוגיות ולשיטות שונות להעברת דיבור על גבי רשתות IP. טכנולוגיות אלו מהוות בסיס למגוון רחב של יישומים (לא בהכרח RTP, אלא טכנולוגיות מגוונות תחת השם VoIP):

- שיחות דיבור בתוכנות למסרים מיידיים (כדוגמת ICQ ו-Skype).
- שיחות טלפון על גבי האינטרנט באמצעות מכשירי טלפון ייעודיים.
- שיחות ממחשבים אל קווי טלפון "רגילים".
- שיחות בין שני קווי טלפון "רגילים" אשר חברת הטלפון בוחרת להעבירן על גבי רשתות IP.

טכנולוגיות ה-VoIP מעבירות דיבור על גבי רשת הנתונים בצורה דיגיטלית. הדיבור האנושי הוא אנלוגי במקורו, ועל מנת להעבירו על גבי רשתות נתונים יש צורך לקודד אותו בצורה דיגיטלית. המקודד (Voice Coder) הוא לב-ליבה של טכנולוגיית ה-VoIP. באמצעות קידוד מתאים של אותות הדיבור (ישנה דרך נוספת לעבד את המידע האנלוגי, בעזרת תדר נייקוויסט, אשר ניתן לקרוא עליו כאן), ניתן להתגבר על תכונות רשת ה-IP[7] ולהצליח להעביר עליה דיבור.

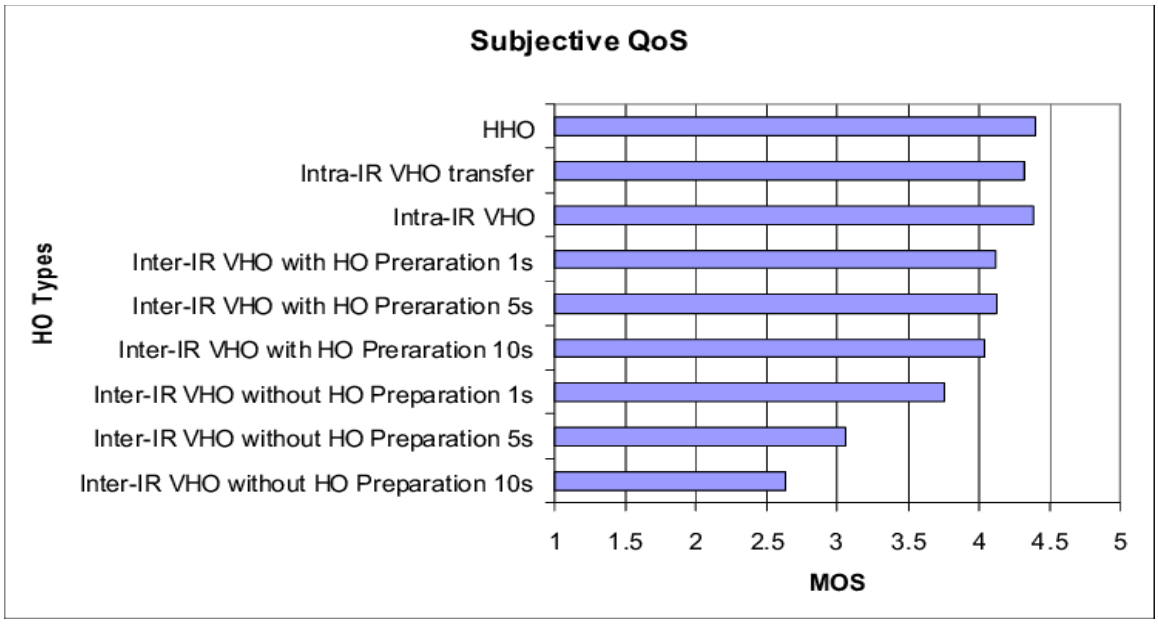
המקודד דוחס את השמע לקצבים נמוכים בהרבה מבטלפוניה רגילה, ובכך חוסך רוחב סרט רב. חלק מרוחב הסרט שנחסך משמש להעברת אותות בקרה להתמודדות עם התכונות הנ"ל (כמו למשל שידור מחדש של חבילות, מספור כל חבילה במספר סידורי, הוספת קודים לתיקון שגיאות ועוד). יתר על כן,

[7] התכונות הבעייתיות של רשת ה-IP: רוחב הסרט הזמין לשימוש מושפע מהעומס הרגעי על הרשת וממספר המשתמשים והיישומים הצורכים את משאבי הרשת בכל רגע ורגע, ולכן גם יכול להשתנות בכל רגע ורגע. בנוסף, המידע ברשתות נתונים לא נשלח בצורה רציפה אלא בחבילות מידע, אשר כל אחת מכילה נתח מסוים מתוך כלל המידע, והחבילה עוברת דרך מסלול שונה ברשת על פי העומס הקיים בכל רגע ורגע בנקודות שונות ברשת

המקודד מעביר רק דיבור אנושי, מסנן צלילים מיותרים ורעשי רקע ונמנע מהעברת השקט שבין המשפטים, המילים והאותיות. העמקה בנושא של טלפוניית IP ניתן למצוא כאן, מאמר מתוך גליון 20.

מושגים חשובים הקשורים ל-VoIP:

- **Jitter** - זהו מצב בו חבילות מידע מסוימות הגיעו לנמען בסדר שונה, או "נאבדו", מה שמוביל לשיחה פגומה ומעורפלת. עקב כך ש-VoIP מסתמך על חיבור האינטרנט, ה-Jitter ייגרם כתוצאה מחיבור אינטרנט לא יציב או תקלות רשת למיניהן.
- **Delay/Latency** - העיכוב בין מתן ההוראה להעברת נתונים לבין העברת הנתונים עצמם. השוואה פשוטה לעולם הגיימינג, עיכוב זה הוא מרווח הזמן בין פעולת השחקן לביצועה. התרחשות עיכוב כזה מכונה "Lag".
- **Mean Opinion Score** - בדומה לשמו, ה-MOS הוא ציון בין 1.0 ל-5.0 שמקבל חיבור בין מכשירים המתבסס על VoIP. באמצעות ניקוד/ציון זה ניתן למדוד את איכות השיחה את הביצועים, את השינויים והבעיות הלא רצויות שעלולות להיגרם. ה-MOS מושפע מחומרה, רוחב פס, Jitter, Latency, איבוד פקטות ועוד...
- **Quality of Service** - איכות השירות, או בקיצור QoS, הוא התיאור או המדידה של הביצועים הכוללים של השירות, ובמיוחד כיצד הביצועים נראים בעיני משתמשי השירות. על מנת למדוד את ה-QoS, ישנם מספר היבטים שלרוב נכללים בחישוב, כמו למשל אובדן חבילות מידע, עיכוב השידור (Latency), חשיבות ותעדוף חבילת המידע, זמינות ו-Jitter.



[לקוח מהאתר ResearchGate]

בטבלה הנ"ל דוגמאות למקודדים נפוצים וציוני ה-MOS שלהם.

לאחר שהסברנו לכם קצת על טכנולוגיית ה-VoIP שקשורה רבות ל-RTP, נראה לכם דוגמה למימוש של הפרוטוקול RTP בשיחת ועידה:

קבוצת עבודה כלשהי נפגשת לדיון, באמצעות שירותי IP Multicast באינטרנט בשביל תקשורת קולית. באמצעות מנגנון הקצאה כלשהו, יושב ראש קבוצת העבודה מקבל כתובת קבוצת שידור (Multicast Group Address) וזוג פורטים. פורט אחד משמש לנתוני שמע, בעוד שהפורט השני משמש עבור חבילות מידע בקרה (מסוג RTCP). כתובת זו ופרטי הפורט מופצים לשאר המשתתפים בדיון (המשתתפים המיועדים). אם רצויה סודיות, הנתונים וחבילות מידע הבקרה עשויים להיות מוצפנים (נפרט על הצפנת RTP בהמשך), ובמקרה זה יש ליצור ולהפיץ גם מפתח הצפנה (הפרטים המדויקים של מנגנוני ההקצאה וההפצה ההלו הם מעבר להיקף הפעילות של RTP. ניתן לקרוא כאן). התוכנה בה משתמש כל משתתף בוועידה, שולחת נתוני שמע בחתיכות קטנות, למשל במשך של ms20 (עשרים אלפיות השנייה). לכל נתח של נתוני שמע קודמת RTP Header.

ה-RTP Header מציין איזה סוג קידוד אודיו (כגון PCM, ADPCM או LPC) נכלל בכל חבילת מידע כך שהשולחים יכולים לשנות את הקידוד במהלך הועידה, לדוגמה, על מנת להתאים משתתף חדש שמחובר דרך קישור ברוחב-פס נמוך או להגיב לאינדיקציות של עומס ברשת. האינטרנט, כמו חבילות מידע רשת אחרות, מאבד מדי פעם חבילות מידע, משחזר אותן ומעכב אותן על פי כמויות זמן משתנות. בשביל להתמודד עם ליקויים אלו, ה-RTP Header מכיל מידע ומספר רצף (Sequence Number) המאפשר לקולטי חבילות המידע לשחזר את התזמון המיוצר על ידי המקור, כך שבדוגמה זו, נתחי שמע מושמעים ברציפות כל ms20 (שחזור תזמון זה מבוצע בנפרד עבור של מקור של חבילות מידע RTP בוועידה). קולט חבילת המידע יכול גם להשתמש במספר הרצף על מנת להעריך כמה חבילות מידע "הולכות לאיבוד".

מאז שחברי קבוצת העבודה הצטרפו ועזבו את הוועידה במשך הדיון, כדאי לדעת מי ממשתף בה בכל רגע נתון וכמה טוב הם מקבלים את נתוני השמע. לשם כך, כל מופע של יישום שמע בוועידה מקבל דוח קבלה בתוספת השם של המשתמש שלו על פורט ה-RTCP.

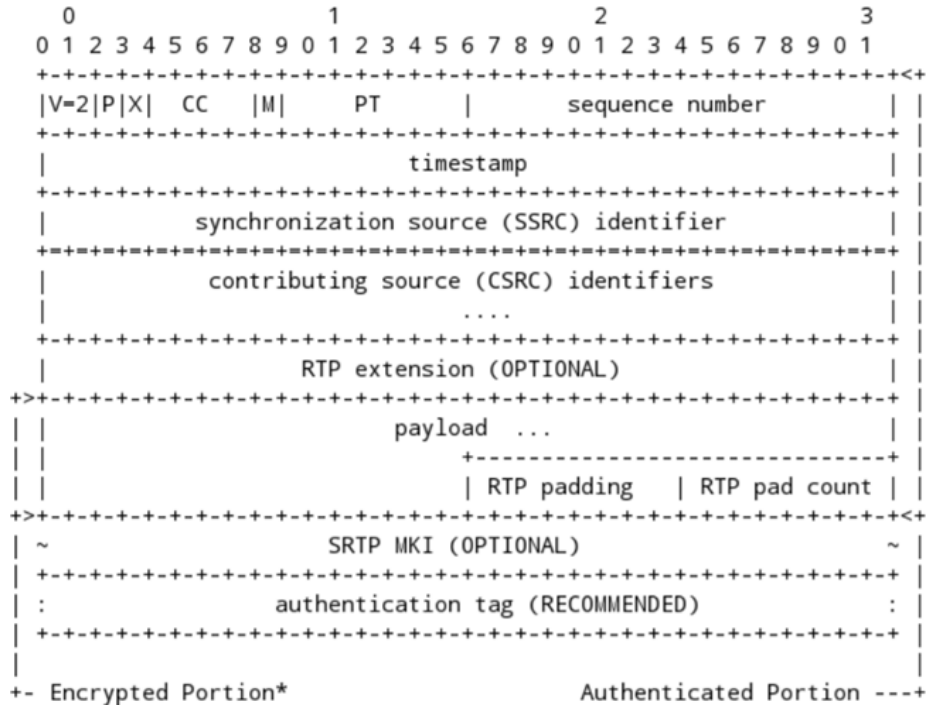
דוח הקבלה מציין עד כמה טוב הדובר הנוכחי מושמע ומתקבל על ידי המופע הספציפי. בנוסף לשם המשתמש, מידע מזהה אחר עשוי גם להיכלל בכפוף לבקרת מגבלות רוחב הפס. המופע שולח חבילת מידע מסוג RTCP BYE (שהיא הודעת סיום תקשורת של המשתמש הנשלחת על ידיו כאשר הוא עוזב את ה-Session) כאשר הוא עוזב את הוועידה.

הצפנה

ישנן שיטות מגוונות להצפין את המידע המועבר בעזרת RTP, בפרק זה נציג חלק מהן.

SRTP

אנו נציג בפניכם כעת את פרוטוקול SRTP (ראשית תיבות של Secure Real-Time Protocol), שהיא שיטה נפוצה מאוד להצפנת המידע המועבר ב-RTP. הרעיון הוא לבצע הגנה לשידור המדיה במצבים של Unicast (כלומר שידור לצד אחד), או Multicast (כלומר שידור למספר גורמים ידועים במקביל, כדוגמת שיחת וועידה). הרעיון של SRTP הינו הצפנה בשני חלקים; חלק אחד הוא הצפנה של חבילת המידע עצמה, והחלק השני הוא הצפנה של חלקי הזרימה. הצפנת חבילת המידע עצמה היא סוג של "Block Cipher" (ובעברית: צופן בלוקים), אשר למעשה לוקחת את כל חבילת המידע, מחלקת לבלוקים, עושה Padding ומצפינה אותה. הצפנת חלקי זרימה, הוא הצפנת המידע (ה-payload) עצמו.



מבנה חבילת המידע מתואר בתרשים כאן למעלה (הלקוח מן RFC 3711), וניתן לראות כי הצפנת ה-block מתאורת בצד ימין למטה, והצפנת המידע (ה-payload) עצמו מתוארת בצד שמאל למטה.

בנוסף ל-SRTP, גם RTCP מכיל פרוטוקול הצפנה ששמו הוא SRTCP. הבעיה היא שלא חובה לבחור בשימוש של SRTP ושל SRTCP. כלומר, גם אם אחד מהפרוטוקולים הללו נכנס להצפנה, זה לא אומר שגם הפרוטוקול השני מוצפן. הפרוטוקול אפילו מאפשר להשתמש בהצפנה בשם "null cipher", שהיא למעשה רק מתנהגת כאילו ישנה הצפנה, אך המידע אינו באמת מוצפן. כאשר רוצים מידע מוצפן כמו שצריך, חייבים לקחת בחשבון את כל הנקודות האלו ולדאוג להן לפני שניתן לדעת שיש הצפנה בפועל של המדיה.

zRTP

זוהי גישת הצפנה מבוססת על עקרון Diffie-Hellman (פרוטוקול דיפי-הלמן) - פרוטוקול המאפשר לשני משתתפים שלא נפגשו מעולם ואינם חולקים ביניהם סוד משותף כלשהו מראש, להעביר אחד לשני מעל גבי ערוץ פתוח (שאינו מאובטח) סוד כלשהו כך שאיש מלבדם אינו יודע מהו (לקוח מתוך ויקיפדיה).

בהצפנה זו יש החלפת מפתחות הצפנה ושימוש ב-SRTP להצפנה עצמה. האות z מייצגת את אחד המתכננים העיקריים של הפרוטוקול בשם פיל צימרמן (Phil Zimmermann, שבנוסף יצר מיזם לפיתוח תוכנה המאפשר ביצוע הצפנה מקצה לקצה של מסרים בצורה קלה ופשוטה, שבעברית נקרא פרטיות טובה למדי, ובאנגלית Pretty Good Privacy - PGP. לקריאה נוספת:
https://en.wikipedia.org/wiki/Pretty_Good_Privacy)

zRTP עובד רק במצב של אחד לאחד, כלומר Unicast, ואינו מתוכנן לעבוד עם מצב של שיחות בין קבוצות. אך המימוש חייב לתמוך ברעיון של Multiplexing - רעיון בו ניתן להעביר יותר ממידע בודד באותו הצינור בדיוק, אותו רעיון של "קו טלפון" מסורתי, אשר כבל אחד יכול "להחזיק" מספר שיחות במקביל.

כמו כן, הפרוטוקול מנסה לבדוק האם הצד השני מסוגל לענות בפרוטוקול, ובמידה וכן, מצפין את המידע, ובמידה ולא, המידע עובר בצורה לא מוצפנת. גישה זו קיבלה את השם "הצפנה מזדמנת" (Opportunistic Encryption).

אחת הדרכים ש-zRTP (אגב, פיל גם יצר את התוכנה Zfone, שהיא מאפשרת להשתמש בשיחות קוליות על גבי האינטרנט בעזרת VoIP, והיא גם משתמשת בהצפנה אסימטרית) מבצע זאת, היא על ידי ביצוע סוג של ping, אשר כל מי שתומך ב-zRTP חייב לענות עליו. במידה ואין מענה, הוא יודע כי לא ניתן "להיכנס" למצב הצפנה.

למרות השימוש ב-Hellman-Diffie, אין צורך בהכרזה מוקדמת על מפתח ההצפנה בפרוטוקול, היות והמפתחות נוצרים בזמן ריצה בין שני הצדדים. גישה זו מאפשרת להגן מפני התקפת MiTM (התקפת אדם בתווך) במידה והמחשב שקיבל את הבקשה הראשונה אינו שייך לתוקף, ובכך קשה מאוד "לזהם" את ההצפנה כאשר זו התרחשה בין שני צדדים.

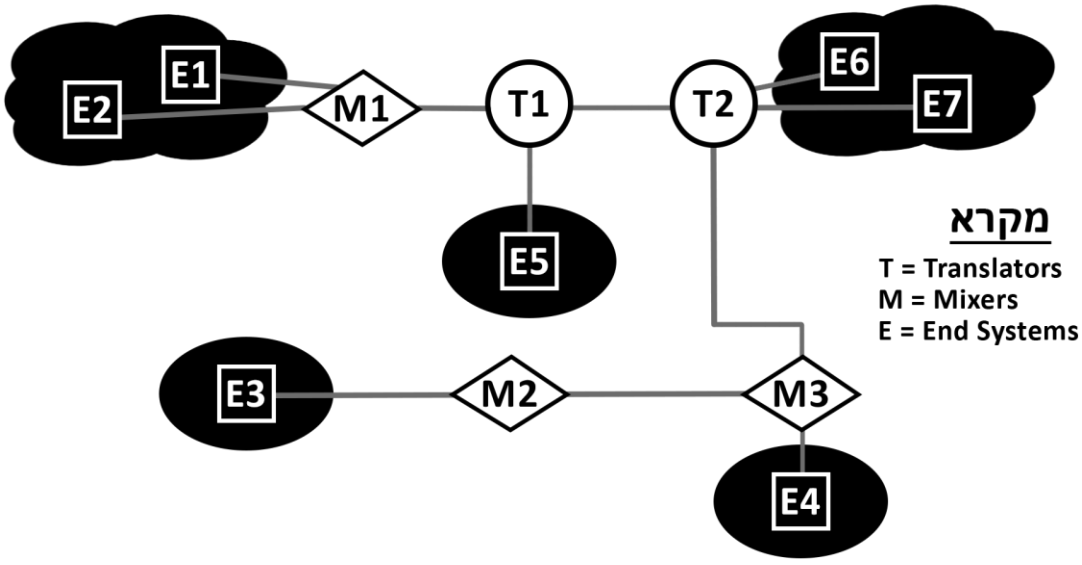
בקשות zRTP מכילות פקודות שונות ולמעשה מכילות תת פרוטוקול תקשורת, כאשר כל תת פקודה שכזו מתועדת ב-RFC ומכילה מבנה משל עצמה. לכן הפרוטוקול נחשב ל"בזבזני" יותר, אך מצד שני מספק יכולות רבות יותר, והגנה טובה יותר, והצפנת המדיה עצמה עדיין מתבצע עם SRTP, אשר zRTP מתערב בו.

להמשך קריאה ולהעמקה נוספת בנושא ההצפנה ל-RTP בפרט ול-VoIP בכלל ניתן לקרוא כאן

רכיבים אלו משמשים להעברה של מידע ב-RTP, עם שינויים אפשריים בו. הם מקשרים בין "ענים" שונים ברמת ה-RTP. למשל, "ענן" יוגדר כקבוצת משתמשים המשתמשת באותה ארכיטקטורת רשת ברמות ה-Network ומעלה, באותה כתובת רשת ובאותו פורט. לדוגמה: IP, UDP, Multicast Address, Port Number או IP, UDP, 2 unicast addresses, 2 Port Number.

ישנם מספר שימושים אפשריים ב-Mixers וב-Translators, אשר כוללים הסרת או הוספת הצפנה, שינוי הקידוד של המידע ושינוי כתובת היעד מ-Multicast למספר כתובות Unicast. ההבדל העיקרי בין שני הרכיבים הוא שה-Translator מעביר זרמי מידע של מקורות בנפרד, ללא איחוד ביניהם, ואילו ה-Mixer מאחד מספר זרמי מידע לכדי זרם מידע אחד.

כעת, נציג בפניכם דוגמה למבנה אפשרי של רשת RTP:



[השרטוט נלקח מ: http://math.haifa.ac.il/oran/adv/RTP.pdf]

ה-Mixer וה-Translator הינם רכיבים שמעברים חבילות מידע RTP. כדי למנוע לולאות בהעברה, כל "ענן" צריך להיות שונה מהשני בלפחות אחד מן הפרמטרים שמגדירים ענן.

כמו כן, Mixers ו-Transaltors לא יחברו במקביל, אלא אם כן יש חלוקה ביניהם של אילו SSRC כל אחד מעביר.

כל ה-SSRC-ים של מקורות באותו RTP Session שחבילות המידע שלהם מועברים על ידי Mixers/Transalotrs צריכים להיות שונים האחד מהשני בכל ה"ענים".

Translator - תחנת Translator מיועדת להעברת חבילות מידע RTP ללא שינוי שדה ה-SSRC שלהן. לפיכך, ה-Translator בעצם שקוף מבחינת הנמענים של חבילות המידע.

לרוב, משמש ה-Translator כאמצעי לשינוי סוג קידוד המידע בחבילות מידע RTP או למטרת איחוד בלוקים של Payload ממספר חבילות מידע שהגיעו מאותו שולח. במקרים אלו, חייב ה-Translator לעדכן את שדות ה-Timestamp וה-Sequence Number של חבילות המידע שעוברות דרכו, כך שיתאימו לחבילות מידע החדשות שייצר. במקרה ששונה הקידוד, יעדכן ה-Translator את שדה ה-Payload Type.

שימוש נפוץ נוסף של Translator הוא תחנת פענוח/הצפנה של מידע מוצפן/לא מוצפן שעובר דרכו.

Mixer - תחנת Mixer מקבלת חבילות מידע מידע ממקור אחד או יותר ומאחדת את זרמי המידע השונים לכדי זרם מידע אחד.

כאשר תחנת Mixer יוצרת חבילת מידע של RTP חדשה, היא דואגת לסנכרון נכון של המידע ממנו מורכבת חבילת המידע. היא עושה זאת בהסתמך על שדות ה-Timestamp וה-Sequence Number של חבילות המידע מהן מורכבת חבילת המידע החדשה.

מאחר ותחנת Mixer שולחת חבילת מידע עם מזהה SSRC משלה, היא מוסיפה לחבילות מידע ה-RTP שהיא יוצר שדה CSRC שמכיל את ערכי ה-SSRC-ים של כל המקורות מהם מורכבת חבילת המידע החדשה.

RTCP - הסבר כללי

מנגנון ה-RTCP מבוסס על שליחה מחזורית, אבל בפרקי זמן העשויים להשתנות, של חבילות מידע RTCP בין כל המשתתפים ב-Session, ובשימוש של אותו מנגנון הפצה כמו של חבילות מידע המידע (חבילות מידע RTP). למשל, שימוש באותו עץ הפצה של multicast.

RTCP מבצע ארבעה תפקידים עיקריים:

התפקיד העיקרי הוא לספק משוב בנוגע לאיכות הפצת המידע. המשוב יכול לספק מידע שיהיה יעיל לקביעת מאפייני הקידוד של המידע, ולבדיקה האם מנגנון הפצת המידע תקין. למשל, אם יש הרבה אובדנים, יש להשתמש בקידוד פחות חסכוני אך רגיש פחות לשגיאות. אם עץ הפצה ב-Multicast אינו מגיע לכל היעדים, ניתן יהיה לגלות זאת.

בנוסף לכך, שליחת משוב אל כל המשתתפים מאפשרת למקבל מסוים לאבחן האם בעיה היא לוקאלית או גלובלית (כלומר, האם מקבלים נוספים סובלים מאותה בעיה).

בתור שימוש נוסף, יכולים ספקי שירותי תקשורת (שאינם מעורבים ב-Session של לקוחותיהם) לנצל את הודעות ה-RTCP כדי לזהות בעיות בתשתית הרשת שלהם.

תפקיד נוסף הינו מנגנון RTCP המספק מזהה נוסף, קבוע, לכל RTP Source שנקרא המאפיין הקנוני (CNAME - שהוא מזהה ברמת התעבורה עבור נקודות קצה של RTP, אשר אמור להישאר ללא שינוי גם כאשר מתגלה התנגשות כלשהי, כך שיהיה ניתן לזהות נקודות קצה של RTP באופן ייחודי. ראו גם: רשומת CNAME בפרוטוקול DNS). מאחר ומזהה ה-SSRC נבחר באקראיות, תיתכן (גם אם בסבירות נמוכה) כפילות שתאלץ שינוי של המספר המזהה. המאפיין הקנוני מאפשר למקבל המידע להמשיך ולעקוב אחר המידע שנשלח על ידי שולח שערך ה-SSRC שלו השתנה.

כמו כן, מאפשר לשלב סוגי מידע שונים מאותו שולח המגיעים ב-RTP Sessions שונים, למשל אודיו ווידאו.
כמו כן, מאחר וכל משתתף שולח הודעות RTCP לכל שאר המשתתפים, יכול כל משתתף לדעת כמה משתתפים יש ב-Session ובעזרת מידע זה להתאים את הקצב שבו הוא שולח חבילות RTCP.

יתר על כן, ה-RTCP הינו שירות אופציונאלי: להעביר מידע בקרה מינימלי עבור השיחה, למשל השם של משתמש,לצורך הצגתו על המסך של המשתמשים האחרים. יש לכך יתרון בשיחות שבהן משתתפים יכולים להצטרף ולעזוב שיחות ללא הרשמה מוקדמת.

סוגי חבילות מידע RTCP[8]

חבילות מידע RTCP מחולקות למספר סוגים:

- **Sender Report (ובקיצור, SR):** דיווח מידע הקשור לשליחת מידע, ומידע סטטיסטי הקשור לקבלה של מידע. הודעות אלו נשלחות על ידי משתתפים המשמשים כשולחי מידע פעילים.
- **Receiver Report (ובקיצור, RR):** אלו הודעות המספקות סטטיסטיקה על קבלת מידע ממשתתפים שאינם שולחי מידע פעילים.
- **SDES (ראשי תיבות של Source Descriptions):** אלו הודעות הצגה של מקורות, זיהויים והעברת ה-CNAME שלהם.
- **BYE:** הודעות סיום תקשורת של משתמש. נשלחת על ידי משתמש שעוזב את ה-Session.
- **APP:** הודעות לשימוש ספציפי של אפליקציה שמשתמשת בפרוטוקול RTP.

חבילות מידע RTCP נשלחות בקבוצה של לפחות 2 חבילות מידע או יותר שיקראו "חבילות מידע מאוחדות". חבילת מידע מאוחדת מוכנסת לתוך אותה חבילת המידע ברמה נמוכה יותר, למשל לחבילת מידע של UDP. חבילת מידע מאוחדת מתאפיינת ב:

1. חבילת מידע של RTCP הראשונה בחבילת המידע המאוחדת חייבת להיות חבילת מידע למטרת דיווח מסוג SR או RR. אם עדיין לא נשלח או התקבל מידע בתחנה ששולחת את חבילת המידע, תישלח חבילת מידע של RR ריקה.
2. בכל משלוח של חבילת מידע מאוחדת תיכלל חבילת מידע של SDES המכילה את מזהה ה-CNAME של השולח שלה. זאת כדי שהמקבל ידע לשלב מהר מידע שהוא מקבל ב-RTP Sessions שונים מאותו משתמש.
3. במידה ומספר המקורות שעליהם נשלחות סטטיסטיקות של קבלת מידע עולה על 31, ישלח המידע הנוסף בחבילת המידע של RTCP הבאה בחבילת המידע המאוחדת.
4. חבילות מידע BYE או APP יופיעו אחרונות.

[8] להרחבה ניתן לקרוא ב-RFC 3550

ניתוח הפרוטוקול RTP באמצעות Wireshark

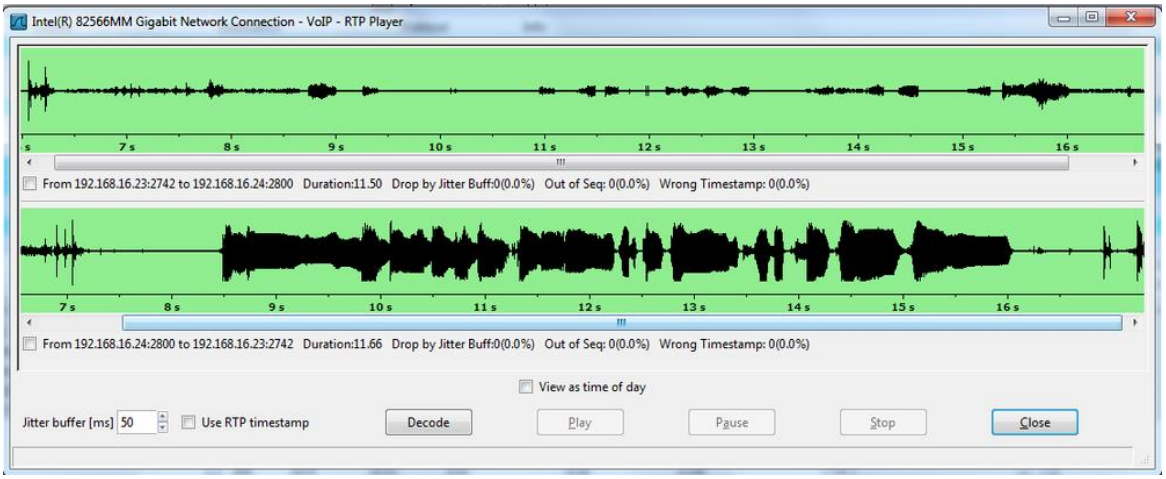
ראשית, Wireshark הינה תוכנת ניטור תעבורת תקשורת ברשת המקומית, אליה מחובר כרטיס הרשת המחובר למחשב (אמנם ניתוח תעבורת הרשת יכול להתבצע גם על פני קובץ PCAP ממחשב אחר). תוכנה זו משמשת בין היתר לאבחון בעיות, ניתוח תעבורת רשת, כלי עזר בתהליך הפיתוח של פרוטוקולי תוכנה ותקשורת, וכמובן למידה.

לפניכם תמונה של הסנפת חבילת מידע של RTP:

```
Internet Protocol Version 4, Src: 192.168.16.23 (192.168.16.23), Dst: 192.168.16.24
User Datagram Protocol, Src Port: tsb2 (2742), Dst Port: acc-raid (2800)
Real-Time Transport Protocol
⊞ [Stream setup by H245 (frame 597)]
  10.. .... = Version: RFC 1889 Version (2)
  ..0. .... = Padding: False
  ...0 .... = Extension: False
  .... 0000 = Contributing source identifiers count: 0
  0... .... = Marker: False
  Payload type: ITU-T G.711 PCMU (0)
  Sequence number: 11639
  [Extended sequence number: 77175]
  Timestamp: 998248329
  Synchronization Source identifier: 0x196d27c5 (426584005)
  Payload: cec4e14b60cb61f8684a70febfcd5f51494d70c1cdde3f4a...
```

[התמונה של ניתוח חבילת המידע נלקחה מכאן]

ה-Header המוצג הנ"ל מציג את הגרסה, האם קיים padding, האם קיימת הרחבה, מזהי מקור של חבילת המידע, האם דגל ה-Marker דולק, סוג המידע, מספר הרצף, חותמת זמן, סנכרון מזהה מקור, ולבסוף המידע עצמו, בהתאמה. הסברים על כל חלקי חבילת המידע הנ"ל מופיעים קודם לכן, תחת כותרת המשנה "מבנה חבילות המידע". כמו כן, התוכנה Wireshark מכילה בתוכה פיצ'ר מובנה הנקרא "VoIP calls" אשר מטרתו היא לפענח מידע של RTP לפורמט אודיו שניתן לניגון על ידי המשתמש.



[התמונה של השמע המוצג למשתמש נלקחה מכאן]

להסברים נוספים על התוכנה והכלי Wireshark ניתן לקרוא במאמר "עבודה עם Wireshark - חלק א'" שפורסם בגליון 75 במגזין זה.

סיכום

במאמר זה הכרנו מושגים רבים הקשורים לעולם התקשורת בכלל, ולפרוטוקול RTP בפרט, למדנו את בסיס הפרוטוקול ונגענו בנושאים רבים הקשורים לו. ראינו את מבנה חבילת המידע ולשם מה נחוץ כל חלק וחלק בו.

יתר על כן, ראינו את המימושים של הפרוטוקול כיום וכיצד השימוש בו גדל בשנים האחרונות, נגענו קצת בטכנולוגיות ה-VoIP ואפילו הראנו דוגמה לשיחת וועידה באמצעות RTP. כמו כן, דיברנו על שתי שיטות עיקריות להצפנת מידע העובר דרך פרוטוקול RTP, אחד הוא SRTP שהוא הנפוץ ביותר והשני הוא zRTP.

ראינו רכיבים המשמשים להעברת חבילות מידע של RTP: ה-Mixer וה-Translator, הבנו את ההבדל ביניהם ובנוסף לכך ראינו בעזרת תרשים דוגמה למבנה אפשרי של רשת RTP. הבנו מהו פרוטוקול RTCP, ראינו כמה שהוא חשוב ולמדנו מספר תפקידים עיקריים אותם הוא מבצע. ראינו 5 סוגים של חבילות מידע, והבנו את המושג "חבילות מידע מאוחדות" ואת המאפיינים שלו.

קצת עלינו

- **דביר גולן** - בן 15, גר בראשון לציון, לומד בתוכנית אודיסאה באוניברסיטת תל אביב במסלול סייבר בשנה א'. אוהב ללמוד כיצד תהליכים שונים במדעי המחשב קורים "מאחורי הקלעים", בין אם זה בחומרה או בתוכנה.
- **שחף גרגיר** - בן 15, גר בצורן, לומד בתוכנית אודיסאה באוניברסיטת תל אביב במסלול סייבר בשנה א'. אוהב לחקור דברים ולהבין כיצד הם עובדים באמת, לעומק, מה שלרוב לא מקבלים בבית ספר.

בחרנו לעסוק בפרוטוקול RTP עקב כך שהשימוש שלו כיום גדל והוא מתקדם בקצב רב בשנים האחרונות, היינו סקרנים לחקור עליו ועל מה שהוא עושה מאחורי הקלעים. המשימה הייתה לא פשוטה, ובמהלכה למדנו המון, החל מחיפוש מידע, הצגתו בצורה ידידותית ומעניינית, חלוקה וסיכום כאשר יש המון חומר, והרחבה ומחקר מעמיק כאשר יש מעט חומר.

המאמר נכתב תוך ליווי מצידו של שלומי בוטנרו כחלק מהפעילות והפרוייקטים בתכנית אודיסיאה

אנו שמחים שניתנה לנו האפשרות לכתוב את המאמר, דבר שהיה מאתגר ותרם לנו רבות.

קישור לאתר תכנית אודיסאה:

https://www.madaney.net/site/programs/odyssey

ריכוז חומרים

הסבר כללי על הפרוטוקול ועל האופן בו הוא מתבצע, מבנה חבילות המידע והמידע המועבר דרך הפרוטוקול:

https://www.rfc-editor.org/info/rfc4585

הסבר כללי על הפרוטוקול ועל האופן בו הוא מתבצע, מבנה חבילות המידע והמידע המועבר דרך הפרוטוקול:

https://www.rfc-editor.org/info/rfc3550

תיאור קצר והצפנה:

https://www.digitalwhisper.co.il/files/Zines/0x3C/DW60-2-VoIP.pdf

הצפנת העברת המידע, בקרה על המידע:

https://www.rfc-editor.org/info/rfc3711

מבנה חבילת המידע וקצת הסבר על מימושים:

https://en.wikipedia.org/wiki/Real-time_Transport_Protocol

הסבר וקצת מימושים (לא בהכרח מהימן)

https://www.youtube.com/watch?v=CttYdLpGeGA&index=3&list=PLOOa-TqFvRv_hyC1pxjHI86rhYMsQItks&t=0s

שימושים (לא בהכרח מהימן)

https://www.youtube.com/watch?v=Ll68ndJZW4g&list=PLOOa-TqFvRv_hyC1pxjHI86rhYMsQItks&index=17&t=0s

מבנה חבילת המידע:

https://www.youtube.com/watch?v=bHwTcoq7n7E&index=1&list=PLOOa-TqFvRv_hyC1pxjHI86rhYMsQItks

יתרונות וחסרונות:

https://answers.yahoo.com/question/index?qid=20090129044549AAbZQbI&guccounter=1

https://www.cs.columbia.edu/~hgs/rtp/faq.htmlb

יתוח תעבורת הפרוטוקול באמצעות Wireshark:

https://www.oreilly.com/library/view/packet-guide-to/9781449339661/ch04.html

הסבר כללי:

http://math.haifa.ac.il/oran/adv/RTP.pdf