**רשתות תקשורת- חלק "יבש"**

1. **תארו במלים שלכם 5 חסרונות/מגבלות של TCP.**
2. האינטרקציה בין מנגנון בקרת הגודש (CC) לבין המשלוח האמין.  
   המטרה היא לשלוט במספר החבילות בתוך הרשת תוך ביצוע בקרת זרימה על השולח ולהבטיח מסירה אמינה. לכן, כל אובדן של מנות מגביל את התקדמות החלון עד שנצליח לשלוח שוב את המידע שאבד. מספר החבילות שנשלחו אך לא קיבלנו עליהן ACK והן כן התקבלו, אינו משקף בהכרח את מספר החבילות שעדיין נמצאות ברשת. כלומר, אי ההתאמה גורר חוסר יעילות בניהול העומס.   
   באופן ספציפי, כאשר מתרחש אובדן מנות, מנגנון בקרת הגודש של TCP מגיב על ידי הפחתת קצב השידור כדי להקל על העומס ברשת. עם זאת, תגובה זו עלולה להוביל לחוסר יעילות בניהול זרימת הנתונים, במיוחד בתרחישים שבהם מספר החבילות הבלתי מאושרות (מנות שנשלחו אך טרם אושרו כפי שהתקבלו) אינו משקף במדויק את מספר החבילות בפועל שעדיין במעבר בתוך הרשת כפי שכבר הסברנו לעיל.

הקטע ממחיש את האתגר הזה באמצעות דוגמה: גם כאשר אין כרגע מנות במעבר בתוך הרשת עקב אובדן המנות, מנגנון בקרת הגודש של TCP עשוי לעצור את שידור המנות החדשות עד לשידור מחדש של החבילה האבודה ואישורה. הדבר עלול לגרום לעיכוב גדול בהעברת מנות נוספות, למרות שלרשת עשויה להיות יכולת כן להכיל ולהעביר אותן.

אי התאמה זו בין מספר החבילות שאינן מאושרות לבין מספר החבילות בפועל במעבר מהווה אתגר עבור TCP, מכיוון שהיא משפיעה על יעילות העברת הנתונים ויכולה להוביל לביצועים לא אופטימליים.

כמו כן, איור 1 משקף בעיה אצל השולח, שבה השולח נחסם מלשלוח מנות נוספות עד לשחזור החבילה האבודה.

1. "חסימת ראש קווים" (HLB)  
   תופעה המתרחשת ב-TCP בשל הבטחתה של מסירה רציפה של זרמי בתים של נתונים.   
   ב - TCP, אם חבילה אובדת במהלך המעבר, לא ניתן להעביר את החבילות הבאות ליישום המקבל עד שהחבילה האבודה תועבר מחדש, תתקבל ויתקבל עליה ACK. וכן, גם חבילות שכבר התקבלו -> לא יהיה ניתן לבצע בהן אף שימוש. הסיבה לכך היא ש -TCP מבטיח מסירה רציפה של נתונים, כלומר יש להעביר מנות בסדר שנשלחו.

בדוגמה המצורפת, חבילה 2 נמסרה בהצלחה למקלט, אך חבילה 1 אבדה במהלך ההעברה. כתוצאה מכך, לא ניתן להעביר את הנתונים מחבילה 2 לאפליקציה עד שחבילה 1 תשודר מחדש ותתקבל. עיכוב זה במסירת מנות עוקבות עקב אובדן חבילה מוקדמת יותר הוא מה שמהווה "חסימת ראש קו".  
איור 2 משקף בעיה במקלט, שבה הנתונים שכבר התקבלו (במקרה זה, חבילה 2) נחסמים בתוך הליבה של המקלט עד שהחבילה האבודה (חבילה 1) משודרת מחדש ומעובדת כדי לשמור על משלוח רציף.

1. עיכוב בגלל הגדרת החיבור

הבעיה המתוארת כאן היא העיכוב שנגרם מהגדרת חיבור TCP, במיוחד כאשר מעורבים אמצעי אבטחה נוספים כגון TLS (Transport Layer Security).

* לחיצת יד משולשת של TCP: כאשר שתי נקודות קצה רוצות לתקשר באמצעות TCP, עליהן ליצור תחילה חיבור באמצעות תהליך הנקרא לחיצת יד משולשת. זה כולל סדרה של שלוש הודעות שהוחלפו בין הלקוח לשרת כדי לסנכרן מספרי רצף ולקבוע פרמטרים לתקשורת. רק לאחר השלמת לחיצת היד הזו ניתן להעביר נתוני יישום בין שתי נקודות הקצה.  
  למעשה, הלקוח שולח בקשה לשרת, השרת מאשר את הבקשה ושולח ללקוח את האישור, ולבסוף הלקוח שולח לשרת אישור על כך שקיבל את אישורו.
* לחיצת יד TLS: אם רוצים לאבטח את חיבור ה-TCP באמצעות TLS, נדרש חיבור הלוך ושוב נוסף להחלפת אישורי אבטחה. במהלך לחיצת היד של TLS, הלקוח והשרת מנהלים משא ומתן על אלגוריתמי הצפנה, מחליפים מפתחות קריפטוגרפיים ומאמתים אחד את זהותו של זה. תהליך זה מבטיח שהנתונים המועברים דרך חיבור ה-TCP מוצפנים ומאובטחים מפני האזנה או חבלה.

הבעיה עם לחיצות הידיים הללו היא העיכוב שהם מציגים לפני שהעברת הנתונים בפועל יכולה להתחיל. כל מעבר הלוך ושוב מוסיף עיכוב לתקשורת, מה שיכול להיות משמעותי במיוחד בתרחישים שבהם ההשהייה הנמוכה היא חיונית, כמו תקשורת בזמן אמת או יישומים אינטראקטיביים.

1. מגבלות עקב כותרת פרוטוקול קבועה:

הבעיה המודגשת כאן נוגעת למגבלות המוטלות על ידי המבנה הקבוע של כותרת ה-TCP, במיוחד בהקשר של הגדלת מהירויות הרשת.

* שדות כותרת בגודל קבוע: כותרת ה-TCP מורכבת משדות בגודל קבוע, כלומר לכל שדה יש אורך קבוע מראש. אמנם יש שדה אופציונלי בכותרת ה-TCP, אך הוא מוגבל ל-40 בתים לכל היותר. מגבלה זו מציבה אתגרים כאשר יש צורך לכלול פונקציות נוספות בכותרת, כגון אפשרויות אישור סלקטיבי (SACK).
* פונקציות בקרת אריזה: עיצוב פרוטוקול TCP אורז את כל פונקציות הבקרה, כולל הגדרת חיבור, פירוק, איפוס ואישור נתונים, בכותרת TCP של 20 בתים. משמעות הדבר היא שככל שיותר פונקציות מזוהות או נדרשות, יש להכיל אותן בשטח המוגבל של הכותרת.
* ההשפעה של הגדלת מהירויות הרשת: ככל שמהירויות הרשת גדלות עם הזמן, שדות מסוימים בכותרת ה-TCP הופכים לבלתי מתאימים. באופן ספציפי, שדות מספר הרצף והאישור (ACK) הם כל אחד באורך 4 בתים בלבד, בעוד ששדה גודל חלון בקרת הזרימה הוא באורך של 2 בתים בלבד. מגבלה זו הופכת לבעייתית במהירויות רשת גבוהות:

- מספרי רצף ושדות ACK עשויים להתעטף מהר מדי, ולהוביל לבלבול פוטנציאלי ואובדן של שלמות הנתונים.

- הגודל הקטן של חלון בקרת הזרימה מגביל ישירות את התפוקה של חיבור TCP. התפוקה מוגבלת על ידי גודל החלון מוכפל ב-Round-Trip Time (RTT), כלומר גודל חלון קטן יותר מגביל את כמות הנתונים שניתן לשלוח דרך החיבור.

בסך הכל, המבנה הקבוע של כותרת ה-TCP מציג אתגרים בהיענות לדרישות הרשת המתפתחות, במיוחד עם המהירות הגוברת של רשתות מודרניות. טיפול במגבלות אלה כרוך לעתים קרובות במינוף אפשרויות כותרת TCP וחקירה של עיצובי פרוטוקולים חלופיים כדי להבטיח ביצועים וסקלביליות מיטביים בסביבות רשת מהירות.

1. מזהה חיבור וכתובת IP ייחודיים

הבעיה המתוארת כאן נוגעת לאתגרים הקשורים לשמירה על מזהה חיבור ייחודי ב-TCP כאשר כתובות IP משתנות במהלך חיי החיבור.

* כתובות IP דינמיות: ב-TCP, השילוב של כתובות IP ומספרי יציאות של שתי נקודות קצה משמש כמזהה החיבור. עם זאת, כתובת ה-IP של כל אחד מארח הקצה בחיבור TCP עשויה להשתנות במהלך חייו של החיבור עקב סיבות שונות. כגון: מולטי-הום מארח (בעל מספר ממשקי רשת), ניידות (למשל, מעבר בין רשתות שונות), או הימצאות מאחורי כתובת רשת מכשיר תרגום (NAT).  
  השפעת שינויי כתובת ה-IP כאשר כתובת ה-IP של אחת מנקודות הקצה משתנה, היא מנתקת את חיבור ה-TCP הקיים מכיוון שמזהה החיבור מבוסס על כתובות ה-IP ומספרי היציאות של נקודות הקצה. כתוצאה מכך, כל הנתונים שהוחלפו עד אותה נקודה בחיבור ייזרקו.
* דרישה ליצירת חיבור מחדש: כדי להתאושש מחיבור ה-TCP הכושל שנגרם כתוצאה משינוי כתובת IP, נדרשת לחיצת יד תלת כיוונית חדשה כדי להגדיר חיבור חדש. זה כרוך בהתחלת תהליך יצירת קשר חדש בין נקודות הקצה, הכולל החלפת מנות סנכרון (SYN) ואישור (ACK) כדי לסנכרן מספרי רצף ולקבוע פרמטרים של תקשורת.
* שיבוש ותקורה: תהליך הקמה מחדש של חיבור TCP עקב שינויים בכתובת IP שמציג הפרעה ותקורה. זה מפריע לזרימת הנתונים בין נקודות הקצה המתקשרות ודורש תעבורת רשת נוספת ותקורה לעיבוד כדי לנהל משא ומתן וליצור את החיבור החדש.

בסך הכל, הבעיה מדגישה את האתגרים הקשורים לשמירה על חיבורי TCP בסביבות רשת דינמיות שבהן כתובות IP עשויות להשתנות. טיפול בבעיה זו כרוך לעתים קרובות ביישום מנגנונים כגון העברת חיבורים, התמדה בהפעלה או פרוטוקולים כמו IP נייד על מנת להבטיח המשכיות תקשורת חלקה ולמזער את ההפרעות הנגרמות כתוצאה משינויים בכתובת IP.

1. **ציינו 5 תפקידים שפרוטוקול תעבורה צריך למלא.**
2. הגדרת מזהה חיבור ומזהה נתונים
3. ניהול קשרי התעבורה

• הגדרה והפסקה של מצב חיבור

• בקרת חילופי מידע בין שני הקצוות: היכולת לקבל גמישות בהגדרת הודעות בקרה חדשות.

• תמיכה בשינויים בכתובת ה-IP המארח (יכולה לנבוע מ-multihoming מארח, ניידות פיזית או סיבות אחרות).

1. מסירת נתונים אמינה  
   • מזהה נתונים ייחודי שיש להחליף בצורה מהימנה עם הקצה השני.  
   • בקרת זרימת חלון לאספקה אמינה: יכולת להימנע מבעיית חסימת ראש קו.
2. בקרת גודש: צריך לשלוט במספר החבילות בתוך הרשת.
3. אבטחה. ליתר דיוק, אנשים רואים כיום ערוצים מוצפנים כאבטחת רשת, למרות שערוצים מוצפנים TLL מספקים רק סודיות נתונים ולא הצפנה. וכן, כיום, אותנטיות ואמון של צד מרוחק מנוהלים באמצעות רשויות אישורים של צד שלישי (CA).
4. **תארו את אופן פתיחת הקשר "לחיצת ידיים" ב-Quic.   
   כיצד הוא משפר חלק מהחסרונות של TCP שתיארתם בסעיף 1?**פתיחת מערכת יחסים בלחיצת יד ב-QUIC כוללת את השלבים הבאים:
5. לחיצות ידיים משולבות של הובלה וקריפטוגרפיה: QUIC משלב החלפת פרמטרים של תעבורה ופרמטרים קריפטוגרפיים לתהליך בודד של זמן הלוך ושוב (1-RTT). זה כולל ביצוע החלפת מפתחות TLS 1.3 מאומתת והחלפת פרמטרי תעבורה מאומתים בו זמנית.
6. משא ומתן על מזהי חיבור: החבילה הראשונית ב-QUIC משמשת למשא ומתן על מזהי חיבור לחיבור חדש. כל נקודת קצה המאכלסת את השדה, מזהה חיבור מקור עם הערך הנבחר שלה, ולאחר מכן משמישה את נקודת הקצה השנייה כמזהה חיבור היעד עבור מנות עתידיות.
7. אימות כתובת (אופציונלי): עם קבלת חבילה ראשונית, שרת יכול לאמת את כתובת הלקוח על ידי שליחת חבילת "נסה שוב" המכילה אסימון אקראי. הלקוח חוזר על אסימון זה בחבילה ראשונית חדשה כדי להמשיך בתהליך לחיצת היד.
8. הודעות לחיצת יד משובצות TLS 1.3: הודעות לחיצת יד TLS 1.3 מוטמעות בחבילות הראשוניות, ומבססות סוד משותף כדי להגן על הסודיות והאותנטיות של מנות עתידיות ב-1-RTT.
9. אימות מזהי חיבור: מזהי החיבור שנבחרו כלולים בפרמטרי התעבורה QUIC, אשר מאומתים במהלך תהליך לחיצת היד של TLS.
10. תמיכה בנתונים 0-RTT : QUIC מאפשר ללקוח לשלוח נתוני יישומים מוצפנים 0-RTT בחבילה הראשונה שלו אל השרת על ידי שימוש חוזר בפרמטרים שנקבעו מחיבור קודם ומפתח TLS 1.3 משותף מראש (PSK) זהות שהונפקה על ידי השרת. עם זאת, נתוני 0-RTT אלה אינם מוגנים מפני התקפות שידור חוזר.
11. טיפול במקרים של שימוש של T/TCP: על ידי תמיכה בשליחת נתוני 0-RTT, QUIC יכול לטפל במקרים שבהם נדרש TCP Fast Open (T/TCP).

QUIC משפר כמה מהחסרונות של TCP בדרכים הבאות:

* השהיית לחיצת יד מופחתת: על ידי שילוב של העברה ולחיצות יד קריפטוגרפיות לנסיעה אחת הלוך ושוב, QUIC ממזער את ההשהיה הנדרשת להקמת חיבור מאובטח בהשוואה ל-TCP, הדורש לפחות 2 זמני נסיעה הלוך ושוב.
* משא ומתן על מזהי חיבור: QUIC מנהל משא ומתן על מזהי חיבור במהלך לחיצת היד, בדומה ל-TCP, מה שמאפשר הגדרה אמינה של חיבורים חדשים.
* תמיכה בנתוני 0-RTT: QUIC מאפשרת ללקוחות לשלוח נתוני יישומים מוצפנים 0-RTT בחבילה הראשונה לשרת, מה שמשפר את מהירות העברת הנתונים הראשונית בהשוואה ל-TCP.
* אימות כתובת: QUIC מאפשר אופציונלית לשרתים לאמת כתובות של לקוחות במהלך תהליך לחיצת היד, מה שמשפר את האבטחה בהשוואה ל-TCP.

בסך הכל, העיצוב של QUIC שואף לתת מענה למגבלות של TCP על ידי אופטימיזציה של זמן לחיצת היד, שיפור האבטחה ותמיכה בהעברת נתונים יעילה, מה שהופך אותו למתאים לסביבות רשת מודרניות.

1. **תארו בקצרה את מבנה החבילה של QUIC. כיצד הוא משפר חלק מהחסרונות של TCP שתיארתם בסעיף 1?**

מבנה החבילה של QUIC:בניגוד למבנה החבילה ב-TCP בו פורמט כותרות החבילה קבוע, כאן יש שני סוגים של כותרות של מנות. עבור יצירת החיבור נשתמש בפורמט של **כותרת ארוכה** משום שעבור יצירת החיבור עם השרת/"הקצה השני" עלינו לדעת יותר מידע על החבילה. לאחר מכן, נוכל להשתמש בפורמט של **כותרת קצרה** כי כעת החיבור קיים ואין צורך במידע רב על מנת להעביר את המנות (צורת עבודה יעילה יותר לעומת ה-TCP). ראוי לציין שבפורמט הכותרת הקצרה, ניתן להטמיע בכול חבילה פריים אחד או יותר ואף כל מסגרת לא חייבת להיות מאותו הסוג של השנייה כל עוד היא בתוך מגבלת ה-MTU.  
  
שיפור חלק מהחסרונות של TCP שתיארנו לעיל בא כאן לידי ביטוי באופן הבא:

1. מבנה החבילה יעיל יותר (כפי שפירטנו).
2. ניהול חבילות משופר לאחר שינוי IP: כאשר מתבצעת החלפת IP, בפרוטוקול TCP נצטרך לבצע לחיצת יד משולשת מחדש. לעומת זאת, ב-QUIC, אילו כבר נוצר בעבר קשר עם אותו IP הנ"ל, לא יהיה צורך בלחיצת יד מחודשת משום שהוא "זוכר" את אותו IP. כך נחסך זמן ושני הצדדים יכולים להתחיל ולשלוח מנות אחד לשני מבלי לבצע שוב לחיצת יד.
3. ניהול התנהגות התקשורת וזרימת הנתונים ברשת: TCP משתמש במנגנוני קביעת חיבור (connection-oriented) המקבילים לתהליכים כמו התחברות, ובדיקת שלמות הנתונים. הוא מקיים בקרת זרימה (flow control) ועומס (TCP CC) באמצעות מנגנונים כגון חלוקת הנתונים לחבילות קטנות יותר, שליחת אישורים (ACKs) חזרה לשולח, וניהול חלוקת הנתונים בצורה שתאפשר לרשת לספוג את התעבורה כמו ה-TCP CC.  
   QUIC לעומת זאת, משתמשת במנגנוני בקרת זרימה שונים מאלה של TCP. למשל, QUIC משתמש בחזרות אוטומטיות, צנזורת אכיפת קצב (congestion enforcement), וניהול זרימת הנתונים בדרך שמאפשרת התאמה גמישה ומהירות גבוהה למצבי הרשת השונים.
4. פורמט מסגרת ACK: ל-QUIC קיימת יכולת לתמוך בעד 256 בלוקים של ACK ב- מסגרת ACK אחת, בהשוואה ל-3 טווחי SACK של TCP עקב מגבלת גודל השדה של TCP. זה מאפשר לבצע ACK למנות שהתקבלו שוב ושוב במספר מסגרות ACK, מה שמוביל לגמישות גבוהה יותר נגד שליחה מחדש של מנות ואובדן. כאשר חבילת QUIC מועברת ל-ACKed, זה מציין שכל הפריימים הנישאים בחבילה זו התקבלו.
5. ריבוי זרמים לעומת זרם יחיד: QUIC משתמשת בריבוי זרמים בתוך שכבת התעבורה בניגוד ל-TCP שאינה עושה זאת ומשתמשת רק בזרם אחד בכול פעם. ניתן למעשה לומר, שכל אחד מן הזרמים ב-QUIC "שווה" לזרם יחיד של TCPsec- המשמש להזמנה של מסירת מסגרות נתונים וזיהוי אובדן ושידור מחדש לאספקת נתונים אמינה.

כמו כן, QUIC משתמש במסגרות STREAM להעברת נתוני אפליקציה וניתן לארוז מספר פריימים מזרמים שונים בחבילת QUIC אחת (מה שלא אפשרי ב-TCP).

1. חסימת ראש קו: משום ש-QUIC משתמשת בריבוי זרמים עצמאיים, היא נמנעת מבעיית חסימת ראש קו הנגרמת כתוצאה מהמתנה לשחזור מנות אבודות. ברגע שקיים אובדן, לא ייחסמו שאר הזרמים ושם המנות יוכלו לעבור באופן תקין ורגיל.
2. Datagram לא אמינה: כיום קיימת דרישה גבוהה להעברת מנות באופן לא אמין על מנת לשדר נתונים בזמן אמת באופן מהיר ככל הניתן. QUIC מאפשרת שיפור בלחיצת יד אמינה ומאומתת ולאחריה מסירת מידע מאובטחת אך לא מהימנה וכן מהירה. בניגוד ל-TCP בו המידע מועבר באופן אמין אך איטי ולא יעיל במקרים בהם הצורך למהירות גבוה.
3. **מה QUIC עושה כאשר חבילות מגיעות באיחור או לא מגיעות כלל?**

כאשר מדברים על אובדן של חבילות ב - QUIC, הפרוטוקול משתמש במספר מנות כדי לזהות את האובדן ולטפל בו. כל מנה ב - QUIC מכילה מספר מסגרות, וה-IP עובר זיהוי של אובדן על סמך מנות אלו.

בעבור כל חבילת ACK שמתקבלת, כל הפריימים הנישאים בחבילה זו נחשבים שהתקבלו. עם זאת, המסגרות הנישאות בחבילה נחשבות לאיבוד אם חבילה זו אינה מאושרת כאשר אושרה חבילה שנשלחה מאוחר יותר, וכאשר קיים סף מסוים.

QUIC משתמש בשני סוגים של סיפים כדי לקבוע אם חבילה שנשלחה קודם אבדה:   
א. מבוסס על מספר מנות: מספר הרצף של החבילה ש"באוויר" קטן מהחבילה המאושרת במספר מסוים.

ב. מבוסס זמן: החבילה ש"באוויר" נשלחה לפחות מספר פעמים במקסימום של ה-RTT המשוער של הרשת הנוכחית וה-RTT האחרון שנדגם לפני החבילה המאושרת.

סיפים אלה מספקים "תקופת חסד" מסוימת, כלומר, תקופת זמן קצוב, לסידור מחדש של מנות ומונעים שידורים חוזרים מיותרים. זה גם נועד למנוע ירידה בביצועים הנגרמת על ידי בקר הגודש בעת זיהוי אובדן מנות.

**איך בפועל זה עובד?**על מנת לזהות את האובדן, QUIC יאתחל טיימר לתקופת זמן הקצוב לבדיקה (נקרא Probe Timeout - PTO), אותה תקופת חסד הנ"ל. בכול פעם שנשלחת חבילה שדורשת אישור, יישלח עימה ה – RTT המשוער smoothed\_rtt, השינוי במדגם RTT rttvar, והזמן המרבי שמהקבל עשוי לחכות בשליחת האישור.  
ברגע שהטיימר פג, השולח ישלח חבילה חדשה הדורשת ACK.  
לאחר זיהוי האובדן, המסגרות האבודות מוכנסות לחבילות יוצאות חדשות (שלהן יוקצו מספרי מנות חדשים).   
כמו כן, עם זיהוי ושחזור אובדן, QUIC תומך במשלוח מהימן של זרם בתים מסודר בדומה לפונקציונליות שמספק RFC.

1. **תארו את בקרת העומס (control congestion) של QUIC.**

נשים לב ש - QUIC מבצע הפרדה בין ניהול העומס לניהול האמינות. הוא משתמש במספרי חבילה כדי לנהל את העומס, ובשטח זיהוי של מסגרות הנתונים עבור ניהול האמינות.

כמו כן, QUIC משתמש בתוכניות המבוססות על חלון להגבלת המספר המקסימלי של בתים (זאת בדומה לדרך ניהול העומסים של TCP עליה למדנו במהלך הסמסטר). עם זאת, אינו משתמש באחד ספציפי, אלא, הוא מספק אותות גנריים לניהול עומסים, והשולח בעל יכולת חופשית ליישם את המנגנון שלו לניהול עומסים.

באופן כללי, QUIC לא יפחית את חלון העומס אלא אם יתגלה עומס עמיד. זאת כדי למנוע הפחתה ללא צורך.  
\* עומס עמיד: (Persistent Congestion) הוא מצב בו הרשת נמצאת במצב של עומס גבוה ותקוע לתקופה ממושכת. במצב זה, הרשת לא מצליחה לספק את הביצועים או לפתור בעיות של תקשורת רשת בצורה סבירה, והתקשורת נתקעת או נפסקת לחלוטין.   
ב – QUIC , מצב זה יקרה כאשר שתי חבילות שדורשות אישור נחשבות לאבודות ואף אחת מהן לא מקבלת ACK.

על מנת לזהות מצב זה, יש צורך בשני דברים עיקריים: דוגמת RTT לפני שנשלחו החבילות ושההבדל בין זמן השליחה שלהן יחרוג מהתקופה העמידה בעמידת העומס, סטיית ה-RTT (rttvar) והזמן המרבי שהמקבל עשוי להשהות בשליחת האישור.

בנוסף, השולח יקבע את שעת השליחה שלו כדי להפחית את הסיכויים ליצירת עומס לטווח הקצר על ידי הבטחה שהמרווח בין שליחת החבילות לא יעבור על גבול אשר מוגדר מראש ומתבסס על ה-RTT הממוצע (smoothed\_rtt), גודל חלון העומס, וגודל החבילה.