תיאור מערכת Ping + Ping and Watchdog timer

סקירת מערכת..................................................................................................................2

פונקציונליות המערכת.......................................................................................................2

איך מריצים......................................................................................................................2

תרשים זרימה לחלק של Ping.............................................................................................3

עץ החלטות למצבים של Ping and Watchdog timer ..........................................................4

הסברים על הקוד (ping.c)........................................................................................... 5-10

תמונות הרצה של ping.c +תמונות מwireshark ..........................................................11-14

הסברים על הקוד (new\_ping.c).................................................................................15-20

הסברים על הקוד (watchdog.c).................................................................................20-24

**סקירת מערכת**

החלק הראשון של המערכת מאפשר לבצע פקודת Ping לכתובת IP מסוימת על מנת לבדוק תקינות החיבור של נקודות הקצה הזאת לIP שהכנסנו לו.

בשלב הראשון נפתח RAW סוקט לכתובת IP שהוכנס בהרצה של המערכת , בשלב השני נשלח לכתובת IP דרך פרוטוקול ICMP הודעה מסוג ICMP-REQUEST בכדי לבדוק חיבור (הודעה זו מיצג נקראת "פינג") ובחזרה נקבל הודעת ICMP-ECHO-REPLY המכונה "פונג" שבאמצעות נוכל לנתח ולראות את תקינות החיבור שלנו ליעד מסויים ברשת.

בחלק השני של המערכת יש לנו רכיב שמדמה Watchdog timer שמכיל טיימר ומקבל כול פעם שבוצע פקודת פינג והגיע תשובה חזרה הודעה שבוצע פינג בהצלחה ומאפס את הטיימר שלו,

במידה וטיימר מראה שעבר מעל 10 שניות ולא הגיעה הודעת שבוצע פינג בהצלחה ה Watchdog סוגר את התוכנית ומדפיס הודעה שאין חיבור לכתובת יעד.

**פונקציונליות המערכת**

בחלק הראשון למערכת יש פונקציונליות של בדיקת תקינות החיבור בין מקור מסויים ברשת ליעד מסויים ברשת והצגת הנתונים הרלוונטים מתוך הפאקטה של ICMP כדי לנתח ולראות את הפרטים מהודעת ICMP-ECHO-REPLY שקיבלנו.

בחלק השני של המערכת אנחנו בודקים אם יש חיבור יציב ורציף אם כתובת היעד בכול 10 שניות באמצעות טיימר שסופר כול כמה שניות מגיעה חזרה הודעת ICMP-ECHO-REPLY בwatchdog .

במידה ולא הגיע ל watchdog הודעה שהתקבל פינג מעבר 10 שניות הוא שלח הודעה ללקוח שמצבע את הפינג שלא הגיע הודעת פינג מעל 10 שניות באמצעות הודעת "time out " וסוגר את הסוקט מולו.

ובמידה והכול בסדר ומגיע פינג רציף מתחת ל10 שניות כול פעם המערכת רצה בצורה אינסופית.

**איך מריצים**

המערכת רצה על מערכת הפעלה Linux בלבד.

מצורף למערכת קובץ Makefile שבו כתובים פקודות שמקמפלות את הקבצים ומכינים אותם להרצה.

לכן בכדי להריץ את החלק הראשון יש לבצע את השלבים הבאים :

1. להיכנס לטרמינל דרך הנתיק של התקייה ולהריץ את הפקודה maka all.
2. לרשום את הפקודה ./parta <ip addrees> ואיפה שרשום ip address נוסיף את הכתובת IP שאליה נרצה לבצע פקודת Ping.

בכדי להריץ את החלק השני יש לבצע את השלבים הבאים :

1. להיכנס לטרמינל דרך הנתיק של התקייה ולהריץ את הפקודה maka all.
2. לרשום את הפקודה ./partb <ip addrees> ואיפה שרשום ip address נוסיף את הכתובת IP שאליה נרצה לבצע פקודת Ping.

**תרשים זרימה לחלק הראשון**

קבלה דרך הmain את כתובת הIP של היעד

יצירת RAW סוקט עם היעד ברשת לפי הIP

שליחת הודעת ICMP REQUEST לכתובת יעד בפונקצית ping

חזרה על התהליך הצורה אינסופית

קבלת הודעת ICMP REPLY מכתובת היעד דרך RAW סוקט בפונקצית listener

נעביר את המידע שקיבלנו לפונקצית display שתדפיס לנו המידע שקיבלנו

נבצע בדיקת checksum לאיתור שגיאות בקבלת הפאקטות

**עץ החלטות של החלק השני**

יצירת RAW סוקט עם היעד ברשת לפי הIP

יצירת חיבור TCP סוקט עם הwatchdog

שליחת הודעת ICMP REQUEST לכתובת יעד בפונקצית ping

קבלת הודעת ICMP REPLY מכתובת היעד דרך RAW סוקט בפונקצית listener ונקבל הודעה מwatchdog בהתאם למצב הטיימר

נבצע בדיקת checksum לאיתור שגיאות בקבלת הפאקטות

נשלח הודעה שיש חיבור מוצלח עם היעד לWatchdog

נעביר את המידע שקיבלנו לפונקצית display שתדפיס לנו המידע שקיבלנו

ההודעה הבאה הגיעה בזמן תקין מתחת ל 10 שניות

שולח הודעה ל listener שהזמן עבר

סוגר את הסוקט מול הלקוח ששולח לו הודעות על הפינג

עבר מעל 10 שניות מהודעה להודעה

נריץ אצל ה watchdog טיימר של 10 שניות שסופר אחורה ומתאפס כול פעם ל 10 שניות שמגיעה הודעה שחיבור הצליח מהפינג

**Ping.c**

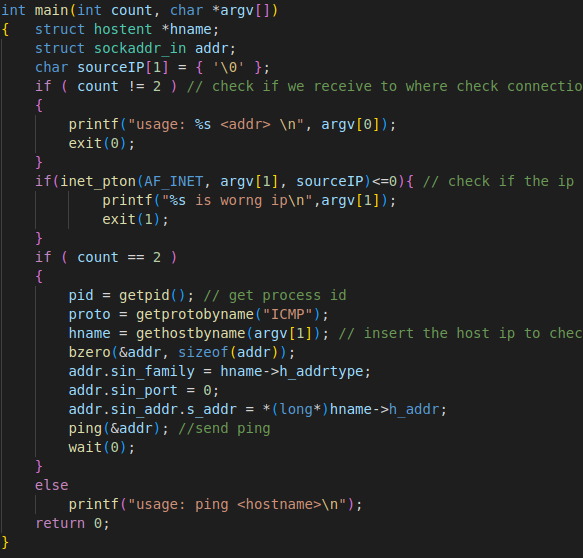
תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטיהקוד מבוסס על קוד שנלמד בתרגול .

תחילה הגדרנו את גדול הפאקטה להיות 64 , הגדרנו מבנה בשם packet שמכיל 2 שדות 1. מבנה בשם icmphdr שמכיל את הhader של icmp 2. סטרינג שיכיל את המידע שנרצה לשלוח.

לאחר מכן הצערנו על משתנים גלובליים pid שיכיל את מספר id של התהליך , מבנה protoent שמכיל מידע על פרוטוקולים ,טיימר לחישוב זמני ההגעה של הודעות ה פינג משתנה שאליו נמדוד את זמן ההתחלה של השליחה ומשתנה יגיד מתי קיבלנו את ההודעת הפינג הראשונה.

בנוסך יש את החתימות של הפונקציות שהשתמשנו בהם checksum, display, listener, ping .



בפונקציית הmain הגדרנו מצביע למבה hostent בשם hname שמכיל מידע על hostים ומבנה sockaddr\_in בשם addrשמכילה מידע על כתובת אינטרנט וchar שישמש אותנו לבדיקת תקינות של כתובת הIP של יעד.

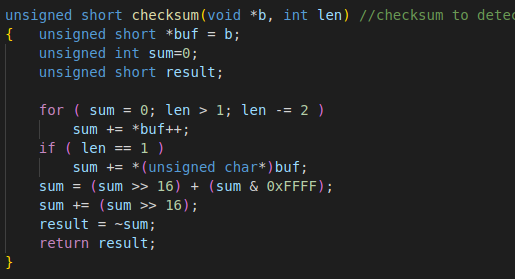
נבדוק שקיבלנו מהמשתמש 2 קלטים שאחד מהם הוא שם הקובץ והשני או הכתובת IP שאליה נבצע את פינג, אם קיבלנו יותר או פחות קלטים נדפיס למשתמש הודעת שגיאה.

לאחר מכן נבדוק שהIP שקיבלנו הוא IP חוקי באמצעות פונקצית inet\_pton שמחזירה ערך שלם , אם חזר 0 או 1- סימן שהייתה שגיאה והIP לא תקין לכן נדפיס הודעה בהתאם.

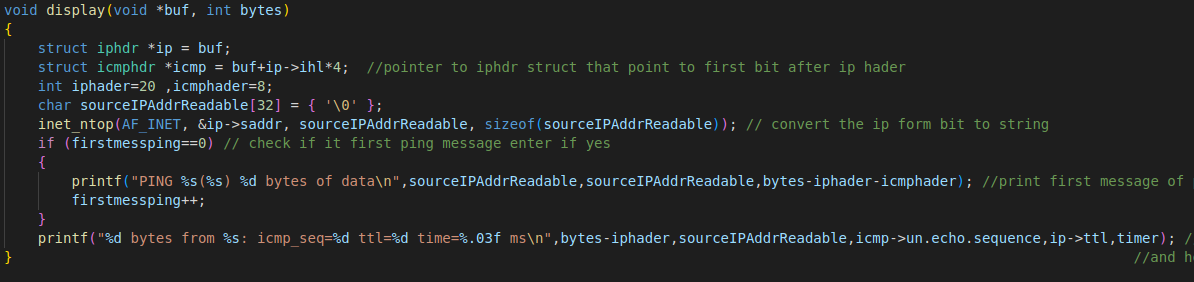
כעת לאחר שבדקו המשתמש הכניס את הכמות הקלטים הנכונה נשמור לתוך משנה pid שהגדרנו כמשתנה גלובליי את ה ID של התהליך שעליו רץ הmain נכניס לporto את שם הפרוטוקול איתו נעבוד (icmp) ול hname את כתובת הIP לביצוע הפינג.

נאפס את המנה addr ונכניס לתוכו את ה הIP שאליה נבצע פינג ,את הפורט וסוג הIP 4V או 6V.

ונקרא לפונקציה ping תשלח לIP שהמשתמש נתן הודעות icmp לבדיקת תקינות החיבור.



פונקצית checksum זהו פונקצייה שבאמצעות נמצא שגיאות וביטים שהגיעו לא כמו שציפינו שיגעו ובכך נהייה בבקרה על המידע שמגיע.



פונקציית display זהו פונקציה שבאמצעות נדפיס את המידע שנרצה להציג למשתמש כחלק מפקודת הפינג שהוא ביקש.

מידע זה מגיע מהודעת ה icmp שמשם נשלוף את המידע.

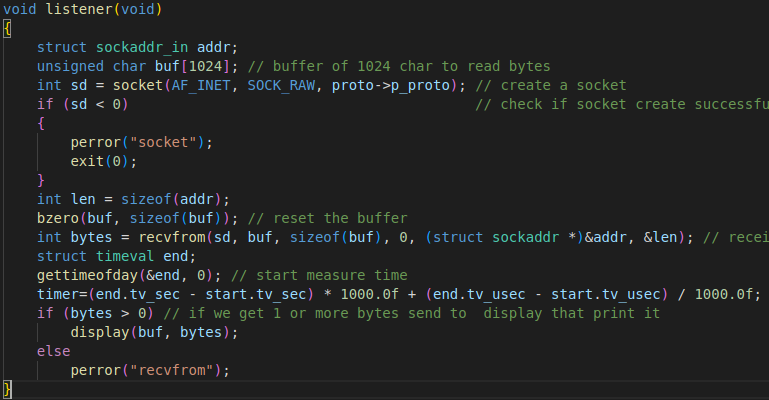
נגדיר 2 מצביעים 1 מסוג iphdr שיצביע על buf שקיבלנו ,ומצביע מסוג icmphdr שיצביע על hader של הודעת ה icmp .

נגדיר 2 משתנים שיכילו את כמות ביטים של הhader של פרוטוקול IP ושל הhader של הicmp.

ניצור סטרינג שיכיל את כתובת הIP של היעד ונכניס את הIP לתוכה באמצעות פונקציית inet\_ntop שממירה את הכתובת מביטים לסטרינג ושם בתוך המשתנה שהגדרנו.

נבדוק שזה ההודעה הראשונה שקיבלנו באמצעות משתנה firstmessping בכדי להדפיס את הפרטים הראשונים על היעד ואז נעלה את משתנה ל1.

לאחר מכן נדפיס את כמות הביטים שקיבלנו ,מספר הפינג ,ה ttl שלו , ומשך הזמן שלקח לתשובה להגיע .



פונקצית listener זוהי הפונקציה שמחכה לקבל את התשובות להודעות icmp שנשלחות אל היעד.

תחילה נגדיר מבנה שיכיל את הכתובת הIP של היעד וbuffer בגדול 1024 ביטים שיכיל את המידע שיגיע.

ניצור סוקט מסוג raw סוקט לחיבור עם היעד באמצעות פונקצית sock שמחזירה ערך שלם , אם חזר ערך קטן מ 0 או 0 סימן שהייתה בעיה עם יצירת הסוקט לכן נדפיס הודעת שגיאה.

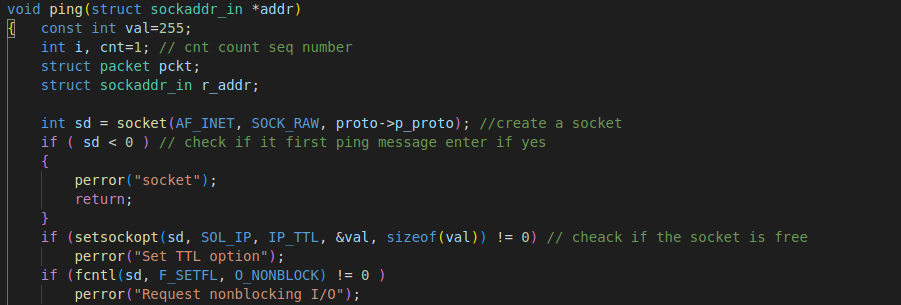
לאחר מכן נגדיר משתנה שיכיל את גודל המבנה addr שמכיל את הכתובת יעד ונאפס את הbuffer .

נקבל את מידע מהסוקט באמצעות הפקודה recvfrom שכותבת את המידע שהתקבל לbuffer , ונמדוד את הזמן כדי לחשב את משך הזמן שעבר באמצעות פקודה gettimeofday.

נבצע החסרה של הזמן סיום מהזמן ההתחלה ונכניס הזמן למשתנה timer .

ולבסוף נבדוק אם התקבל מידע ואם כן נשלח אותו לdisplay להדפסה ואם לא קיבלנו מידע נדפיס הודעת שגיאה בהתאם.

בסיום החלק של פונקצית ה listener נחזור בחזרה לפונקציית ping כדי לשלוח שוב ליעד הודעה icmp לבדיקת תקינות החיבור.

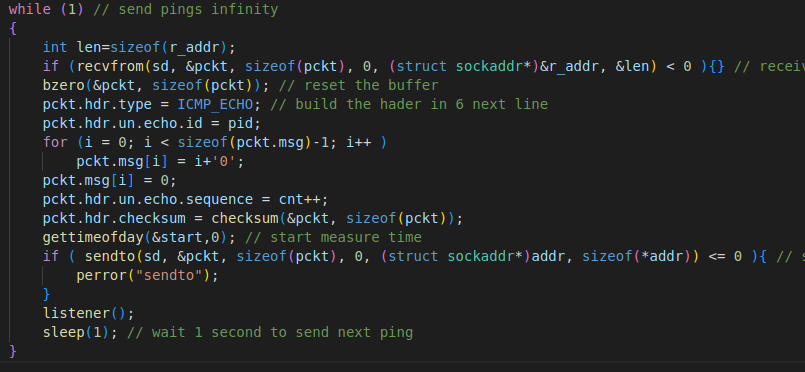


פונקצית ping זוהי הפונקציה ששולח את ההודעות ה icmp ליעד לבדיקת תקינות החיבור.

נגדיר מספר משתנים cnt לספירת מספר ההודעות שנשלחו , פאקטה באמצעות מבנה pckt ,כתובת IP של היעד באמצעות המבנה sockaddr\_in וbuffer בגודל 1024 ביטים לקריאת המידע שהתקבל.

ניצור סוקט לשליחת ההודעת icmp מסוג raw סוקט באמצעות פונקצית sock שמחזירה ערך שלם , אם חזר ערך קטן מ 0 או ש 0 סימן שהייתה בעיה עם יצירת הסוקט לכן נדפיס הודעת שגיאה.

נבדוק שהסוקט פנוי באמצעות הפונקציה setsockop שמחזירה ערך שלם אם חזר ערך שונה מ 0 סימן שהיתה שגיאה ולכן נדפיס הודעה בהתאם.



לאחר מכן נרוץ בלולאה אינסופית כיון שלא נגביל את כמות הפעמים של ביצוע פקודת ה פינג,

נשמור במשנה את גודל המבנה שמכיל את הכותב של היעד, נאפס את הפאקטה באמצעות bzero .

ונכניס נתונים לפאקטה icmp\_echo שזה שלחת פינג pid זה ID של התהליך .

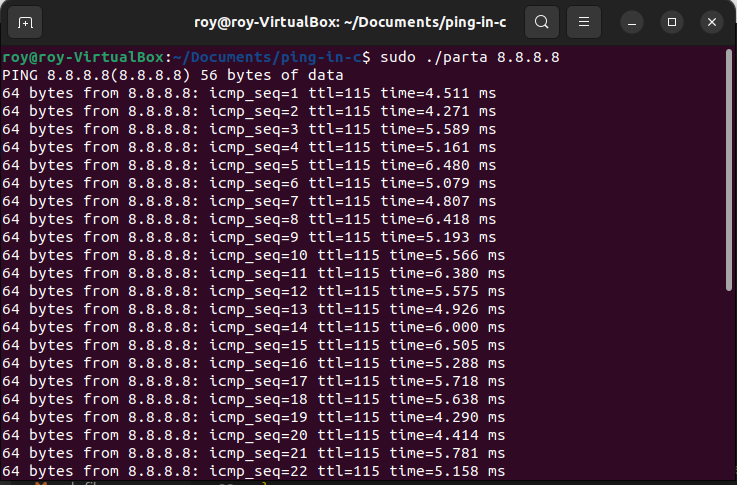
ואז נכניס את הנתונים של הdata של הודעת ה icmp בלולאה ,מכניס את מספר הפינג שנשלח (cnt ) ונאפס את הchecksum .

נשלח את הבקשה לכתובת היעד באמצעות פונקצית sendto שמחזירה ערך שלם אם חזר ערך קטן או שווה ל 0 סימן שהייתה שגיאה לכן נדפיס הודעת שגיאה בהתאם.

לאחר מכן נקרא לפונקצית listener שזוהי הפונקציה שקוראת את המידע שמגיע בחזרה מהיעד ושולחת אותו להדפסה למשתמש.

ולבסוף באמצעות פנקציית sleep נמתין בשליחה לשליחה שניה אחת כדי לא להציף את היעד בהודעות icmp .

נחזור על תהליך שליחת הפינג ליעד לבדיקת החיבור באופן אינסופי.

**הרצה של ping.c**

בתמונה זו ניתן לראות דוגמא להרצה של התוכנית .

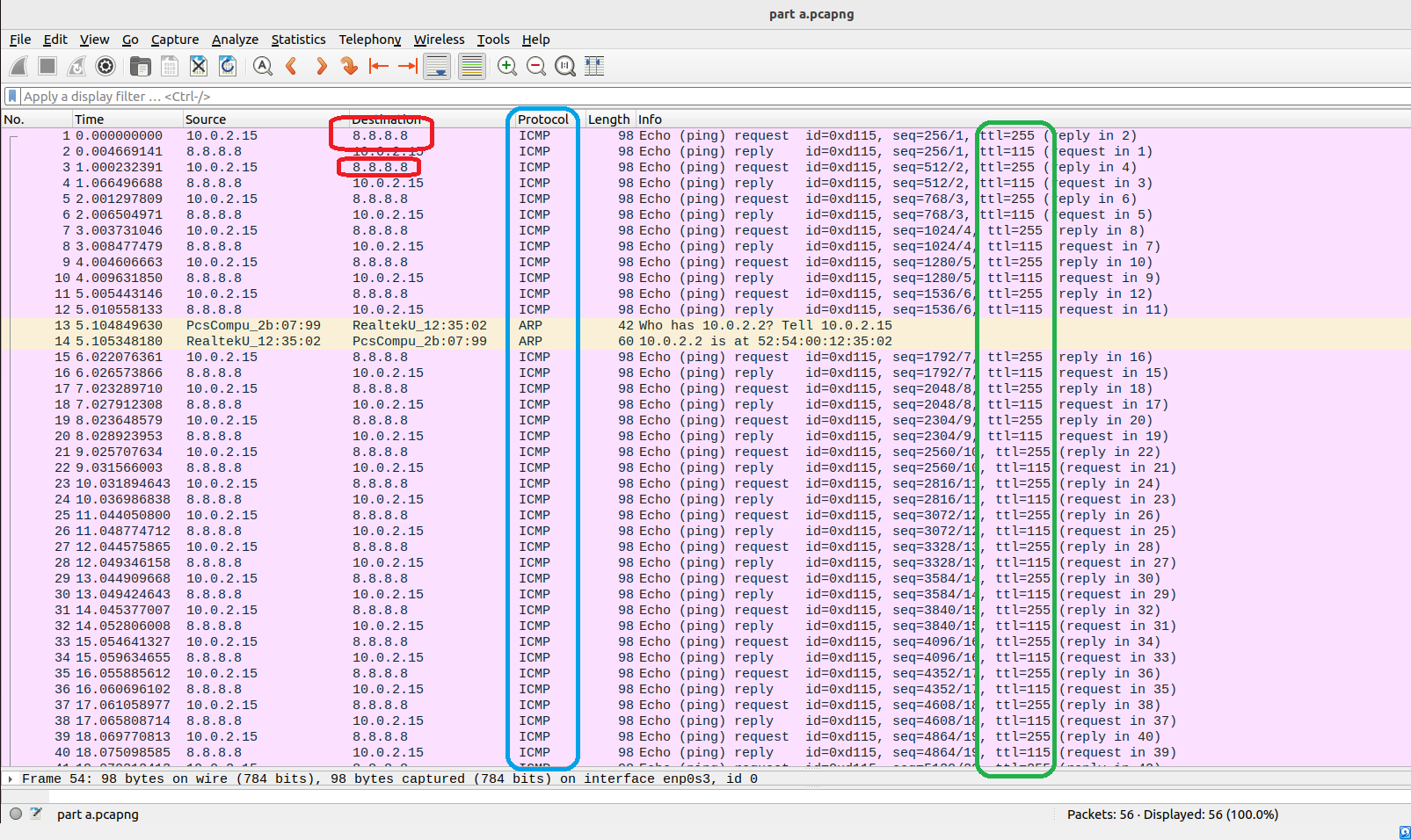
בהרצה הזאת בדקנו חיבור לכתובת IP 8.8.8.8 שזה שרת גוגל , כפי שניתן לראות היה לנו חיבור לשרת הנ"ל וקיבלנו ממנו את כול הפרטים על מנת שנוכל להדפיס את הודעת הפינג כפי שהיא מוצגת למשתמש.

ניתן לראות את כמות הביטים של ה data שהגיעו שחישבנו אותם בכך שהחסרנו מהם את כמות הביטים של ה hader של פרוטוקול הIP (20 ביטים) ו את כמות הביטים של הhader של פרוטוקול הICMP (8 ביטים).

בנוסף מוצג למשתמש את כתובת הIP של היעד שהוא הכניס , את המספר הסידורי של הפינג בסדר עולה , ttl של החבילה שזה כמה רכיבי תקשורת החבילה יכולה לעבור עד שהיא תיזרק.

ולבסוף מוצג למשתמש את הכמות מילישניות שלקח למידע להגיע (מרגע שליחת הבקשה עד לקבלת התשובה מהיעד).

**תמונות מwireshark בעת הרצת התוכנית**

התמונות לקוחות מהקלטה parta .

כפי שניתן לראות בתמונה במסך הראשי של wireshark רואים את כתובת ה IP של היעד שאליו אנחנו מבצעים את הפינג לבידקת קישוריות.

ניתן לראות גם את סוג הפרוטוקול איתו עבדנו שהוא בעצם פרוטוקול ICMP .

ניתן לראות גם שהודעה שנשלחת ליעד היא מסוג replay והודעה שחוזר אלינו מהיעד היא מסוג request ובאות אחת אחרי השנייה לכול replay נקבל את request שלו מיד אחריו.

בנוסך ניתן לראות את הttl של החבילה ששלחנו ליעד ואת הttl של החבילה שהגיע אלינו מהיעד אפשר לראות שהttl של החבילות שהגיעו אלינו הוא 115 .

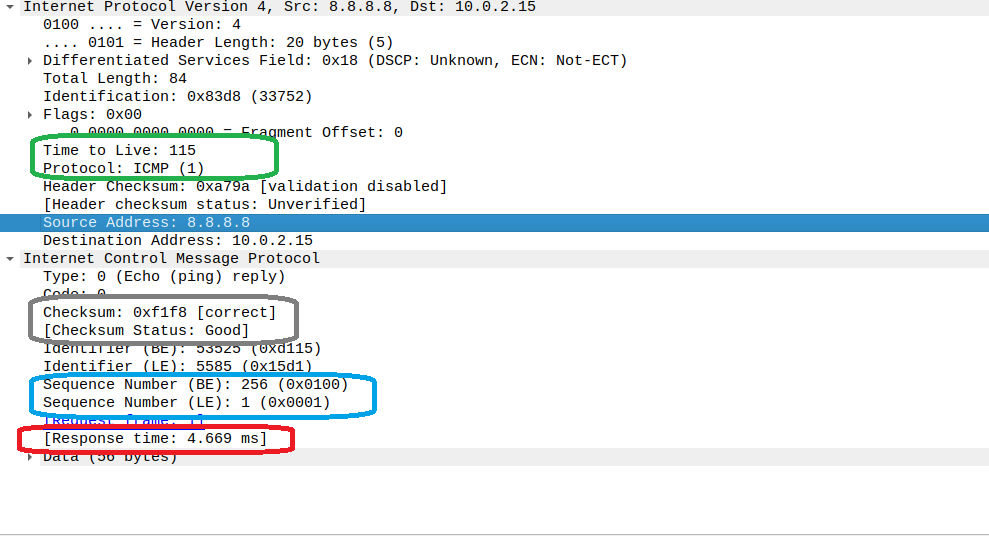
תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטיתמונה של replay ליעד

כפי שניתן לראות במלבן האדום אפשר לראות גדול של הhader של פרוטוקול ה IP בחבילה שנשלחה זה 20 ביטים כמו שציינו קודם.

במלבן הירוק ניתן לראות את כתובת ה IP שלנו ושל היעד .

במלבן הכחול ניתן לראות את סוג הבקשה מתוך הודעת ה ICMP ששלחו ליעד בקשה מסוג ECHO שהמספר שלה הוא 8 והיא בעצם בקשת פינג.

במלבן השחור ניתן לראות שאנחנו עובדים עם checksum לאיתור שגיאות ולכן מופיע שה checksum היה נכון (correct ) ושהסטטוס שלו הוא good .

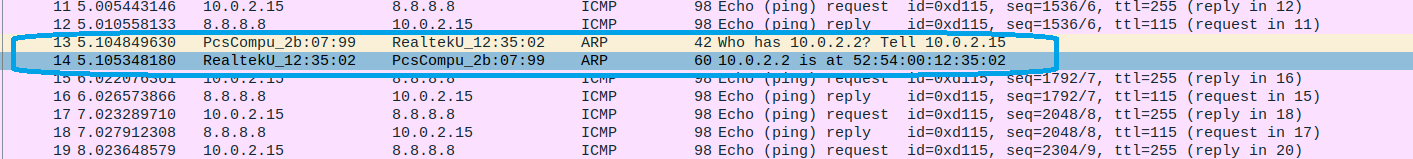
תמונה של request מהיעד

כפי שניתן לראות במלבן הירוק ניתן לראות את ttl של החבילה ואת סוג הפרוטוקול שאיתו שלחנו את החבילה.

במלבן האפור ניתן לראות שוב את checksum לאיתור שגיאות רק בצד השני שכמובן גם פה checksum היה נכון (correct ) ושהסטטוס שלו הוא good .

במלבן הכחול ניתן לראות את המספר הסידורי של הפינג שנשלח sequence number .

במלבן האדום ניתן לראות את המשך הזמן שלקח לחבילה להגיע מגע שנשלחה הבקשה ליעד עד שחזרה התשובה בדיוק של מילישניות.



במלבן הכחול מופעיות 2 חבילות שבפרוטוקול ARP שזה פרוטוקול שמשמש לאיתור כתובת MAC של נקודה ברשת לפי כתובת ה IP שלה.

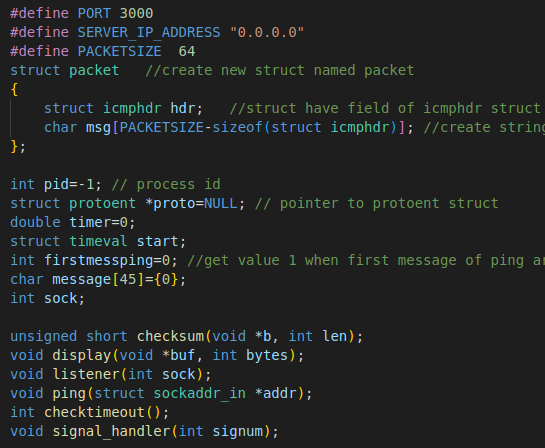
התהליך עובד ככה שמתבצע שידור של  [broadcast frame](https://he.wikipedia.org/wiki/Broadcast)המכיל את כתובת ה IP של התחנה המבוקשת אל כול התחנות באותו טווח הכתובות ברשת על פי הכתובת הפיזית בלבד.

התחנה שתזהה את כתובת ה IP שלה , תשלח בחזרה את כתובת הMAC של חזרה אל תחנת המקור .

לכן כפי שניתן לראות החבילה הראשונה שואלת מי זה 10.0.2.2 ? ובתגובה לזה בחבילה השנייה

ש10.0.2.2 זה 52:54:00:12:35:02 שזה כתובת הMAC של התחנה.

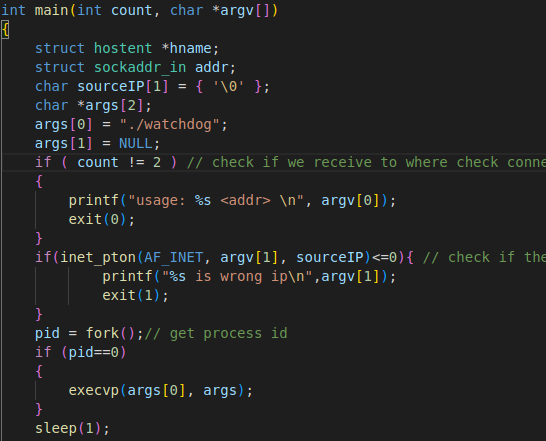
**New\_ping.c**

קוד זה מבוסס על ping.c לכן נציג רק את החלקים שבהם הוספנו חלקים ונסביר אותם.

הגדרנו משתנים חדשים פורט 3000 ו IP של המחשב כדי שנוכל לפתוח איתם סוקט TCP לwatchdog .

הוספנו משתנים גלובליים שהם מערך של תווים שייצג את ההודעה שנשלח אל הwatchdog ומשנה מסוג int שיחזיר את ה socket descriptor אל watchdog .

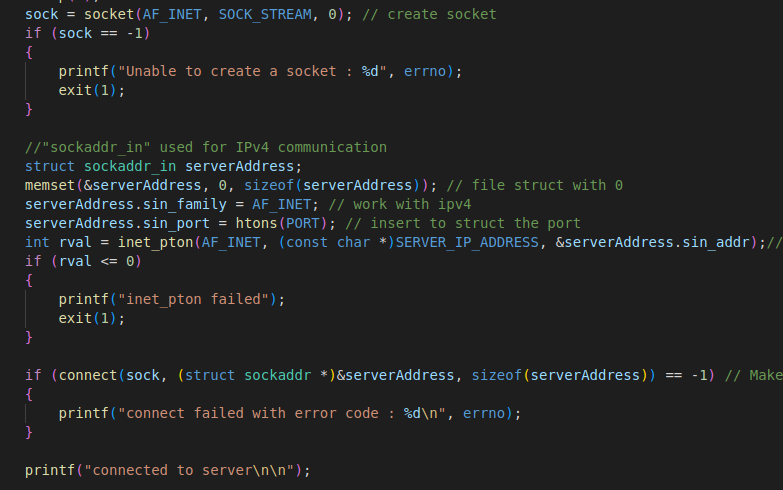
והוספנו עוד 2 חתימות לפונקציות שרשמנו checktimeout שבודקת האם הגיע הודעת time out מ watchdog ועוד חתימה לפונקציה signal\_handler שאם עבר 10 שניות מאז ההודעה האחרונה ששלנו ל watchdog מגיעים לפונקציה הזאת והיא קוראת לchecktimeout לבדיקה אם הגיעה הודעת timeout .



בפונקציית הmain הוספנו מצביע למערך של 2 charים שמכיל את הקובץ הרצה של התוכנית watchdog.

לאחר מכן ניצור תהליך חדש שהוא תהליך ילד באמצעות הפונקציה fork שמחזירה ערך שלם , אם חזר 0 סימן שיצירת התהליך הסתימה בהצלחה לכן נריץ את watchdog באמצעות הפונקציה execvp שתריץ לנו את watchdog בתהליך נוסף.

ולבסוף נמתין שנייה אחת כדי להמשיך את התוכנית כדי שניתן ל watchdog לרוץ ולהיות מוכן לאישור החיבור של הסוקט של התוכנית .new\_ping .



יצרנו סוקט באמצעות הפקודה socket שמקבלת כפרמטים את סוג הIP איתו אנחנו עובדים וסוג הפרוטוקול (אצלנו TCP) , הפונקציה מחזירה ערך שלם לכן לאחר מכן בדקנו אם הפונקציה החזירה 1- אם כן יצירת הסוקט לא הצליחה הדפסנו הודעת שגיאה ויצאנו מהתוכנית.

הגדרנו מבנה בשם sockaddr\_in שהוא מבנה שמכיל מידע של כתובת IP פורט עליו עובדים וסוג הIP שאיתנו אנחנו עובדים גירסא 4 או 6.

תחילה איפסנו את המבנה באמצעות memset באפסים.

ואז הגרנו לו שאנחנו עובדים עם IP V4 והכנסנו לו את הפורט עליו אנחנו עובדים שאותו ההמרנו לביטים באמצעות htons.

ולבסוף המרנו את הכתובת לייצוג בינארי באמצעות פונקציית inet\_pton שמקבלת כפרמטרים סוג הIP עליו עובדים מצביע לכתובת ה IP ואת המקום במבנה שבוא מאחסנים את הכתובת לאחר ההמרה.

פונקציה inet\_pton ערך שלם 1- אם יש שגיאה 0 אם ההמרה לא עברה בהצלחה או מספר גדול מ 0 אם ההמרה בוצע בהצלחה לכן הוספנו תנאי שבודק שהערך המוחזר גדול מ 0 ואם לא מציג הודעת שגיאה ויוצא מהתוכנית.

הקמנו חיבור עם הwatchdog באמצעות הפונקציה connect שמקבלת כפרמטים socket descriptor שקיבלנו מפונקציה socket , מצביע למבנה שהגדרנו קודם וגודל המבנה שהגדרנו בביטים.

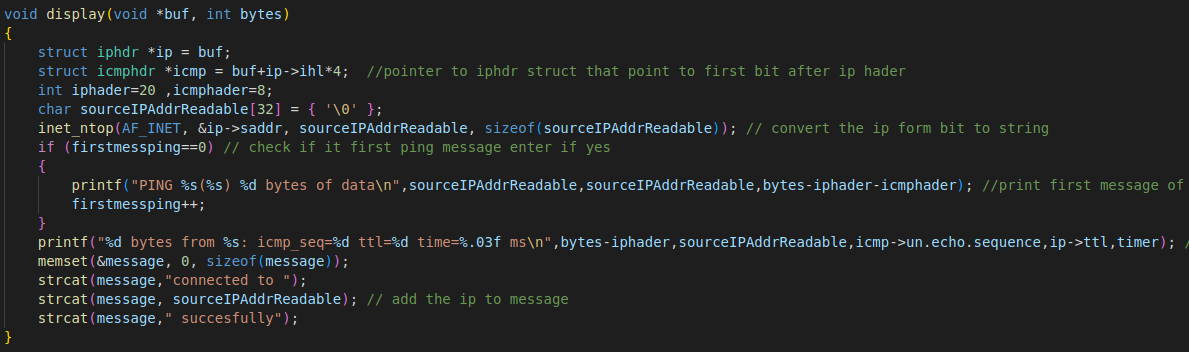
הפונקציה מחזירה 1- אם החיבור לא הוקדם לכן הוספנו בדיקה שהערך החזרה הוא לא 1- ואם הוא כן הפונקציה מדפיסה שגיאה.

תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטי

איפסנו את הsignal באמצעות הפונקציה signal\_handler שכתבנו שהיא מגדירה ל טיימר מה לעשות לאחר שעובר 10 שניות שלא קיבלנו הודעת פינג מהיעד.

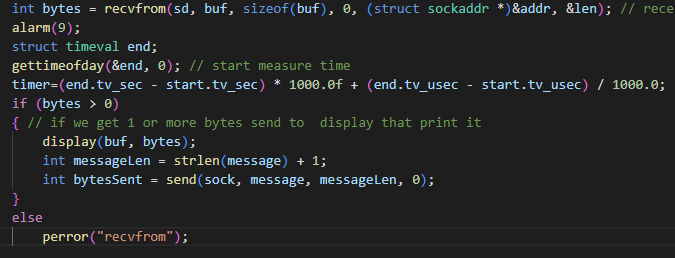
ולבסוף נכנס לתנאי שכתבנו כמו בקוד ping.c ונפעיל את הפונקציה ping ואחריה נוסיף את הפקודה wait שבאמצעות נוודא שהתוכנית הראשית לא נסגרת לפני שהתוכנית watchdog נסגרת.



פונקצית הdisplay לפונקציה זאת הוספנו את החלק האחרון שבו הכנסו למערך שמכיל את ההודעה שתשלח ל watchdog את המידע שנרצה להעביר.

תחילה נאפס את המערך , ונוסיף אליו את ההודעה הבאה "connected to " וממשתנה sourceIPaddreadable נקבל את הIP של כתובת היעד ונוסיף אותה גם להודעה ולבסוף נוסיף גם את המילה "successfully".

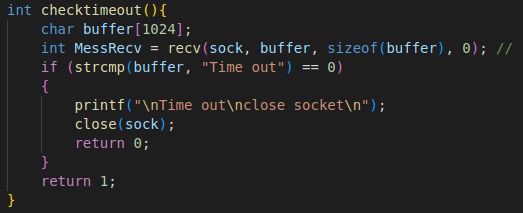
לכן נשלח לwatchdog את ההודעה הבאה " connected to (ip to ping ) successfully ".



בפונקציה listener הוספנו את החלק הבא : alarm(9) שהוא סופר 9 שניות אחורה אם הגיעה ל0 הוא אוטומטית הולך לפונקציה singnal\_handler ועושה מה שיש בתוכה.

בגלל שפונקציית listener רצה שוב ושוב אז השעון שרץ האחורה יתאפס כול פעם שוב ושוב ל 9 שניות עד שבאמת תהיה בעיה לא תגיע הודעת פינג מהיעד לבדיקה.

ובנוסך בבדיקה האם קיבלנו באמת מידע מהיעד נוסיף גם את החלק של השליחה של ההודעה ל watchdog באמצעות כך נגדיר משתנה שיכיל את גודל ההודעה ונשלח את ההודעה באמצעות פונקציית send שמקבל את socket descriptor שהגדרנו בתחילת התוכנית כמשתנה גלובלי , את ההודעה עצמה שיש שלוח ואת גודל ההודעה.

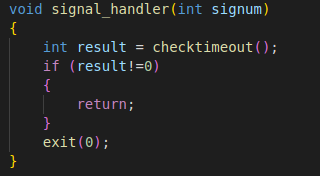


פונקציית checktimeout זהי הפונקציה שתבדוק האם קיבלנו הודעת timeout מwatchdog .

לכן תחילה נגדיר buffer בגדול 1024 ביטים .

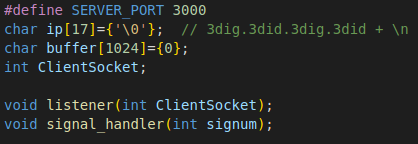
נקבל מהwatchdog את ההודעה שהוא שולח באמצעות הפונקציה recv שמקבל את ה socket descriptor שהגדרנו בתחילת התוכנית כמשתנה גלובלי, את ה buffer שאליו נכתוב את ההודעה ואת גודל הbuffer .

לאחר מכן נבדוק האם באמת קיבלנו את ההודעה של timeout אם כן נדפיס שקיבלו את ההודעת ה timeout ונסגור את הסוקט מול ה watchdog ונחזיר 0 אם לא קיבלנו את הודעת ה timeout נחזיר 1.



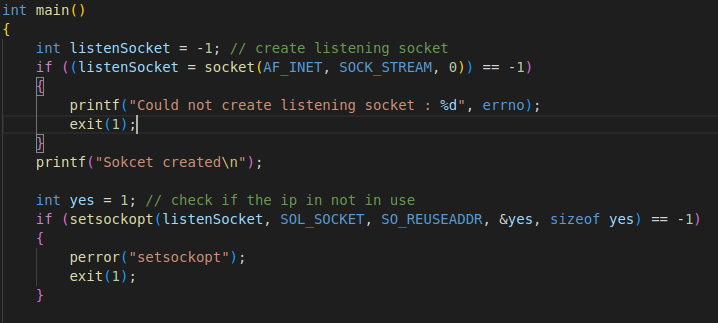
פונקציית הsingnal\_handler זוהי הפונקציה שמגיעים אלי אחרי שהטיימר הגיעה ל 0.

לכן אם הטיימר הגיעה ל 0 סימן שלא קיבלנו תשובה להודעת ה ICMP ששלחנו ליעד אז נקרא לפונקציה checktimeout שתבדוק האם הידע timeout מ watchdog ונבדוק האם הגיעה הודעת timeout אם לא הגיעה נחזור חזרה ואם כן הגיעה נסגור את התוכנית.

**Watchdog.c**

תחילה הגדרנו את קבוע את המשתנה שמחזיק את הפורט שאיתו נעבוד (3000).

ואז הגדרנו מספר משתנים גלובליים שהם מערך בשם IP שמחזיק את הכתובת שעליה מבצעים את הפינג , buffer בגודל 1024 ביטים שאליו נכתוב את המידע שהnew\_ping שולח ו socket descriptor שאליו יתחבר ה new\_ping באמצעות TCP סוקט.



ניצור את הסוקט באמצעות הפקודה socket שמקבלת כפרמטים את סוג הIP איתו אנחנו עובדים וסוג הפרוטוקול (אצלנו TCP) , הפונקציה מחזירה ערך שלם לכן לאחר מכן בדקנו אם הפונקציה החזירה 1- אם כן יצירת הסוקט לא הצליחה הדפסנו הודעת שגיאה ויצאנו מהתוכנית.

אם הצליחה ליצור סוקט נדפיס הודעה בסוקט נוצר.

לאחר מכן נבדוק שהIP לא בשימוש באמצעות פונקציית setsocktop הפונקציה מחזירה ערך שלם אם הוחזר 1- זה אומר שעדיין יש ביטים של סוקט שעדיין מחוברים לכן נדפיס הודעת שגיאה ונצא מהתוכנית.

תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטי

הגדרנו מבנה בשם sockaddr\_in שהוא מבנה שמכיל מידע של כתובת IP פורט עליו עובדים וסוג הIP שאיתנו אנחנו עובדים גירסא 4 או 6.

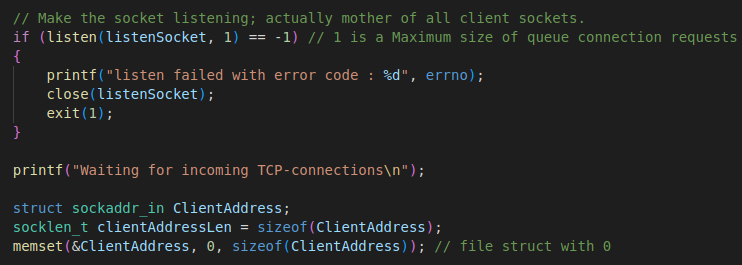
תחילה איפסנו את המבנה באמצעות memset באפסים.

ואז הגרנו לו שאנחנו עובדים עם IP V4 והכנסנו לו את הפורט עליו אנחנו עובדים שאותו ההמרנו לביטים באמצעות htons.

נקשר את הכתובת והפורט עם הסוקט באמצעות הפונקציה bind שמקבלת כפרמטים את הסוקט, את המבנה שהגדרנו קודם ואת הגודל שלו.

הפונקציה מחזירה ערך שלם , אם הוחזר 1- זה אומר שהיה שגיאה בקישור בין הכתובת והפורט לסוקט אם הוחזר ערך אחר הקישור נוצר בהצלחה.

לכן נבדוק אם הוחזר ערך 1- אם כן נציג שגיאה נסגור את הסוקט ונצא מהתוכנית.



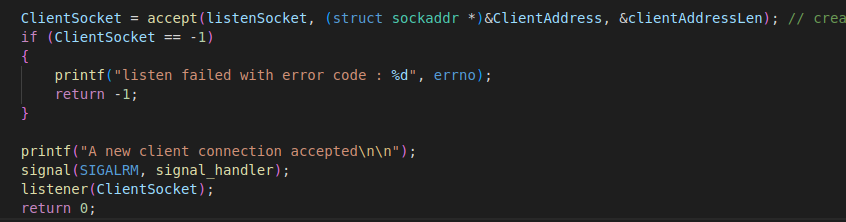
נגדיר את הסוקט שלנו שיכול להאזין ללקוח אחד בו זמנית באמצעות פונקציית listen.

הפונקציה מחזיר ערך שלם אם מחזירה 1- זה אומר שהייתה שגיאה אם ערך אחר זה אומר ש ההגדרה בוצע בהצלחה , לכן נבדוק שלא הוחזר לנו 1- אם כן נדפיס הודעת שגיאה נסגור את הסוקט ונצא מהתוכנית.

בחלק הבאה נדפיס הודעה שאנחנו מחכים לחיבור.

נגדיר עוד מבנה שמכיל את הIP והפורט של הnew\_ping , ונגדיר משתנה שמכיל את הגדול של המבנה הזה.

נאפס את המבנה באמצעות memset באפסים.



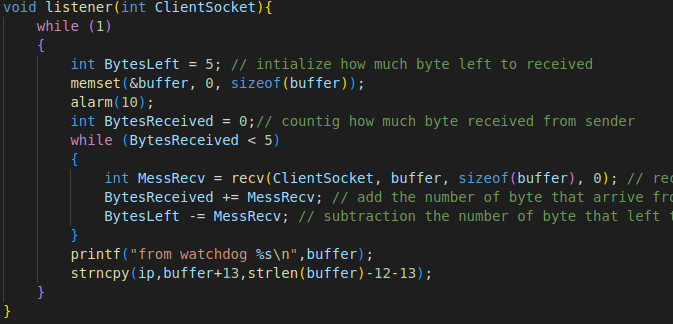
נוציא בקשה לחיבור מתוך תור הבקשות באמצעות הפונקציה accept שמקבל כפרמטר את ה socket descriptor שהגדרנו בתחילת התוכנית כמשתנה גלובלי, מצביע למבנה שמכיל מידע על הכתובת והפורט של הSender ואת גודל המבנה בביטים.

הפונקציה מחזיר ערך שלם אם הוחזר 1- סימן שהייתה שגיאה אחרת החיבור בוצע בהצלחה , לכן נבדוק שלא הוחזר 1- אם כן נדפיס הודעת שגיאה ונצא מהתוכנית.

אם החיבור נוצר בהצלחה נדפיס הודעה של הבוצע התחברות של לקוח חדש.

לאחר מכן הגדיר את הsignal באמצעות הפונקציה signal\_handler שכתבנו שהיא מגדירה ל טיימר מה לעשות לאחר שעובר 10 שניות שלא קיבלנו הודעת מnew\_ping שביצע פינג ליעד.

ונשלח את socket descriptor לפונקציה listener מאזינה לסוקט וקוראת את המידע שנשלח.

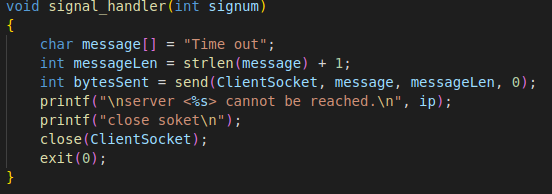


פונקציית listener זוהי הפונקציה שמאזינה לסוקט ומקבלת את המידע שנשלח.

לכן נגדיר לולאה אינסופית כיון שלא ידוע מספר הפעמים שלח לנו הודעה , נגדיר משנה שיכל את כמות הביטים שנשארו לקבל ונאפס את הbuffer ונפעיל שעון שסופר אחורה למשך 10 שניות שנאפס אותו כול פעם שנכנס ללולאה ואם הוא מגיע ל 0 הוא אוטומטית קורא לפונקציה signal\_handler .

נגדיר את כמות הביטים שהתקבלו ונרוץ בלולאה עד שנקבל את כול הביטים של ההודעה.

בתוך הלולאה נקבל את המידע באמצעות recv שהוא קוראת את המידע מתוך socket descriptor וכותבת את המידע ל buffer ונחסיר מכמות הביטים שנשארו את כמות הביטים שקיבלנו ונוסיף למשתנה כמות הביטים שהגיעו את הביטים שקראנו.

לבסוף נדפיס את תוכן של הbuffer , ונעתיק מתוכו את הIP של היעד למשתנה IP שהגדרנו בתחילת התוכנית כמשתנה גלובלי.

פונקציית signal\_handler זוהי הפונקציה שמגדירה מה לבצע שהטיימר מגיע ל 0.

לכן תחילה נכניס למערך את ההודעה שנרצה לשלוח ונשמור את גדול ההודעה במשתנה נוסף.

נשלח את ההודעה באמצעות הפקודה send דרך ה socket descriptor לnew\_ping .

נדפיס למשתמש הודעה שאין חיבור לIP שהוא נתן ביחד עם ה IP עצמו ,נסגור את הסוקט , ונצא מהתוכנית.