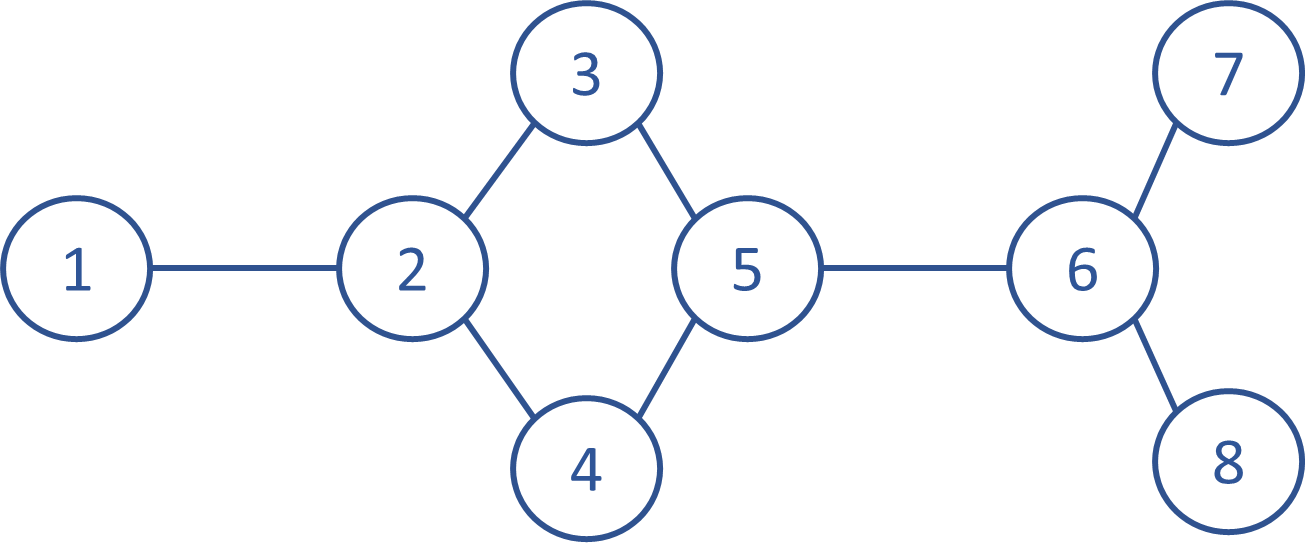
# פיתוח מערכת שליחת הודעת על בסיס רשת עמית-לעמית (Peer-to-Peer)

## הגדרת התרגיל

עליכם לפתח צומת (node) ברשת שמטרתה לאפשר תקשורת בין צמתים שונים. הצמתים לא בהכרח מתקשרים ישרות האחד לשני, והם יכולים לתקשר באמצעות צמתים אחרים.

לדוגמה, נתונה הרשת הבאה המורכבת משמונה צמתים:



לדוגמא צומת מס' 2, מחוברת לשלושה צמתים: 1,3 ו-4.

אם צומת מס' 2 מעוניין לשלוח הודעה לצומת מס' 6 הוא יכול לעשות את זה דרך אחד מהמסלולים הבאים: 2← 3 ←5←6 או 2←4←5←6. בחירת המסלול בו ישתמש הצומת יוגדר בהמשך.

## פקודות משתמש בסיסיות

כל צומת מקבל פקודות מהמשתמש שאותה הצומת קורא באמצעות הפונקציה fgets.

הפקודות אותם משתמש יכול לתת לצומת מתומצתות בטבלה הבאה:

|  |  |  |  |
| --- | --- | --- | --- |
| תיאור | מבנה הפקודה | דוגמה | תיאור הדוגמה |
| קביעת מזהה לצומת | setid,<*id*> | setid,7 | קביעת מזהה הצומת למספר 7 |
| התחברות לצומת חדש (יכול להגיע | connect,<*ip*>:<*port*> | connect,127.0.0.1:1234 | בקשת התחברות לצומת בכתובת 127.0.0.1 בפורט 1234 |
| שליחת הודעה לצומת | send,<*id*>,<*len*>,<*message*> | send,3,11,hello world | שליחת ההודעה “hello world” לצומת מספר 3. 11 מציין שמספר התווים לשליחה הוא 11. |
| הדפס מסלול לצומת | route,<*id*> | route,3 | בקשה להדפיס את המסלול מהצומת הנוכחי לצומת 3 |
| הדפס רשימת צמתים מחוברים | Peers | peers | בקשה להדפיס מי הקבצים המחוברים לצומת הנוכחי |

**לכל הפקודות הצומת עונה ack אם הצליח או nack אם נכשל**. אם יש מידע נוסף, הוא יודפס לאחר מכן.

## פרוטוקול התקשורת בין הצמתים

התקשורת בין הצמתים מבוססת על הודעות במבנה ידוע ומוגדר מראש כפי שיוגדר להלן. מבנה ורצף ההודעות מרכיבים את פרוטוקול התקשורת בין הצמתים.

כל צומת מבצע ארבע פעולות בסיסיות מול צמתים אחרים:

1. **התחברות** הפעולה הבסיסית שהצומת עושה הוא להתחבר לפחות לצומת אחד ברשת. צומת יכול להיות מחובר ליותר מצומת אחד.   
   **שימו לב:** כל צומת צריך להיות מסוגל גם לקבל התחברויות מצמתים אחרים.
2. **חיפוש מסלול לצומת**. בכדי לשלוח הודעה לצומת יש להכיר את המסלול אל הצומת. במידה וצומת היעד מחובר ישירות לצומת המקור, ניתן לשלוח ישירות את הודעה, אחרת יש לאתר את המסלול אל הצומת.
3. **ממסור (relay) של הודעה**. כאשר צומת מקור איננו מחובר ישירות לצומת היעד, הוא יכול לבקש מצמתים במסלול לממסר את ההודעה אל צומת היעד.
4. **שליחת הודעה**. שליחת הודעה היא כאשר היא מגיעה לצומת היעד.

הפרוטוקול הוא פרוטוקול בינארי מורכב מהודעת (messages) בגודל קבוע של 512 בתים. מבנה כל הודעה בפרוטוקול מתואר בטבלה הבאה:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| MSG ID  (4 bytes) | Source ID  (4 bytes) | Destination ID  (4 bytes) | # Trailing Msg  (4 bytes) | Function ID  (4 bytes) | Payload  (492 bytes) |

משמעות השדות:

1. Msg ID – מזהה ייחודי להודעה
2. Source ID – מזהה הצומת השולח
3. Destination ID – מזהה הצומת המקבל
4. # Trailing Msg – מספר החלקים שנשארו להודעה הנוכחית
5. Function ID - מזהה הפונקציה
6. Payload

## רשימת הפונקציות הבסיסיות

|  |  |  |  |
| --- | --- | --- | --- |
| מספר הפונקציה | שם הפונקציה | פרמטרים | פונקציית תגובה |
| 1 | Ack | מספר ההודעה עליה נשלח ה-ack (4 בתים ראשונים ב-payload) |  |
| 2 | Nack | מספר ההודעה עליה נשלח ה-nack (4 בתים ראשונים ב-payload) |  |
| 4 | Connect | ללא | ack או nack |
| 8 | Discover | צומת היעד (4 בתים ראשונים ב-payload) | route |
| 16 | Route | 1. מספר הודעת ה-discover המקורית (4 בתים ראשונים ב-payload) 2. אורך התשובה (4 בתים שניים ב-payload). במידה ולא נמצא מסלול אורך התשובה יהיה 0. 3. הצמתים לפי סדר השליחה (4 בתים לכל צומת) |  |
| 32 | Send | 1. אורך ההודעה (4 בתים ראשונים ב-payload של ההודעה הראשונה בלבד) 2. ההודעה עצמה (החל מהבית החמישי) | ack או nack |
| 64 | Relay | 1. מה הצומת הבא במסלול (4 בתים ראשונים ב-payload) 2. מספר ההודעות העוקבות אותם יש לממסר (4 בתים שניים ב-payload) | ack או nack |

## סדר הפעולות בשליחת הודעה

### התחברות

פעולה זו מתבצעת כאשר מתקבלת פקודה מהמשתמש להתחברות לצומת חדש.

**תנאי מקדים**: על צומת להכיר את המזהה שלו, לפני שהוא מתחבר לצומת החדש.

רצף הפעולות:

1. פתיחת TCP socket לכתובת (IP, port) שהתקבלו מהמשתמש.
2. במידה והפתיחה נכשלה, יש לשלוח הודעת nack למשתמש
3. שליחת הודעה עם connect עם Source ID מאותחל למזהה הצומת ו-0 כמזהה הצומת השני
4. הצומת השני צריך להחזיר הודעת ack עם מזהה הצומת שלו ב-Source ID, מזהה הצומת הנוכחי ב-Destination ID, ומספר הודעת ה-connect ב-payload
5. במידה ואחד המזהים לא נכון, יש להחזיר הודעת nack למשתמש.
6. במידה וכל המזהים נכונים, יש לשמור את מזהה הצומת, כתובת ה-IP וה-port במערך. יש לוודא שאורך המערך הוא באורך מספר הצמתים המחוברים לצומת הנוכחי. יש להדפיס את מזהה הצומת למשתמש

### חיפוש מסלול

פעולה זו מתבצעת כאשר מתקבלת פקודה מהמשתמש לשלוח הודעה לצומת אחר, אבל לא ידוע מסלול לצומת המבוקש.

**תנאי מקדים**: הצומת מחובר לפחות לצומת אחד נוסף.

אופן הפעולה מבוסס על ביצוע הצפה של הודעת discover ברשת וקבלת המסלול בחזרה.

1. אם צומת היעד הוא צומת מחובר חזור מיד הודעת ack.
2. הצומת יעביר הודעת discover לכל הצמתים השכנים שעוד לא קיבלו ההודעה. אם אין צומת כזה, יש להחזיר nack.
3. לכל צומת שכן, בצע את הפעולות החל משלב 1.
4. אם מכל השכנים התקבלה הודעת nack החזר nack.
5. לכל השכנים שהחזירו route, בחר את המסלול הקצר ביותר. מבין המסלולים הקצרים בחר את המסלול שהסדר הלקסיקוגרפי שלו הוא הנמוך יותר.
6. למסלול שנבחר הוסף את מזהה הצומת הנוכחי בראש הרשימה תוך התאמת כלל השדות ושלח הודעת route למקור הבקשה. יש להחזיר הודעת route על כל אחד מהחיבורים מהם הגיע בקשת discover.

את רשימת המסלולים יש לשמור במערך שאורכו כמספר המסלולים הידועים, שמכיל מצביע לרשימת הצמתים על המסלול.

### ממסור

כאשר יש לשלוח הודעה לצומת יעד, אבל צומת היעד איננו מחובר ישירות לצומת המקור, יש לממסר את ההודעה לצומת היעד. פעולת הממסור היא פקודה לצומת מחוברת להעביר אוסף של הודעות לצומת שהיא מחוברת אליה.

את רצף הצמתים הנדרש ניתן לקבל מבקשת *חיפוש מסלול*. להודעה שצריכה להגיע מצומת לצומת על מסלול באורך צמתים, מצומת , יש לשלוח הודעות relay אחת בתוך השנייה ובסוף הודעת send.

כל צומת שמקבל הודעת relay ממסר את מספר ההודעות **העוקבות** שהוא התבקש לצומת שאליה הוא התבקש לממסר את המידע.

אם צומת מתבקש לממסר הודעות לצומת שאיננו מחובר אליו הוא צריך להוציא הודעת nack.

לדוגמה, לפי הציור לעיל, כדי לשלוח הודעה מצומת 1 לצומת 5, על המסלול צומת 1 תשלח שרשור של שתי הודעות relay ובתוכם הודעת send (ראו להלן).

לאחר שהצומת יקבל ack או nack על ממסור מהצומת הבא הוא יוציא הודעת ack או nack לצומת ששלח אליו את ההודעה בהתאמה.

### שליחה

צומת מקור ששולח הודעה לצומת יעד ישלח הודעת send (לאחר שמומסרה כראוי) בה תוכן ההודעה.

1. במידה וצומת מקבל הודעת send שאיננה מיועדת אליו הוא יוציא חזרה הודעת nack למי ששלח לו את ההודעה.
2. במידה והודעת ה-send מיועדת לצומת הנוכחי, הוא ידפיס את ההודעה למשתמש ויחזיר הודעת ack לצומת ששלח אליו את ההודעה.

# חומרי עזר

לתרגיל מצורף קוד שמאפשר להאזין בו זמנית על מספר socket ועל ה-stdin (מקלדת). הדבר נדרש כדי שניתן יהיה לקבל קלט מהמשתמש (מקלדת) או מכל אחד מהצמתים שמחוברים לצומת הנוכחי. במידה ומתקבל קלט כאמור יש לפעול על פי ההגדרות.

בקוד המצורף יש שני פונקציות – אחת להוסיף file descriptor (מזהה של ה-socket) בשם add\_fd\_to\_monitoring. הפונקציה מקבל כקלט את מספר ה-file descriptor ומוסיפה אותו לרשימת ה-file descriptors שאותם היא מנטרת. הקוד מנטר אוטומטית גם קלט שמגיע מהמשתמש

הפונקציה השנייה היא wait\_for\_input הממתינה עד שיהיה קלט באחד ה-file descriptors (רשימת ה-socket שהוזנה באמצעות ב-add\_fd\_to\_monitoring או קלט מהמשתמש). כאשר יש קלט זמין כאמור, הפונקציה wait\_for\_input תחזיר את ה-file descriptor שממנו ניתן לקרוא. לצורך ההבהרה, file descriptor מספר 0 הוא קלט שהגיע מהמשתמש וניתן להשתמש בפונקציות סטנדרטיות לטובת קריאת הקלט מהמשתמש. שימו לב שלאחר קריאת המידע הזמין מומלץ להפעיל שוב את הפונקציה wait\_for\_input **לפני** ביצוע קריאה נוספת מאותו file descriptor. במידה ויש קלט זמין נוסף הפונקציה תחזור מיד.

הקוד המסופק כולל קוד לדוגמה לאופן השימוש בפונקציות.

# הערות כלליות

בדיקת התרגיל תהיה אוטומטית, בחלקה או כולה. לפיכך, אין להוסיף תווים מיותרים, כולל רווחים שורות חדשות וכו'. כמו כן, אין להחסיר תווים. שימו לב שכל שורת קלט\פלט מהמשתמש\אל המשתמש מסתיימת בשורה חדשה (‘\n’) ללא רווחים לפניה.

# דוגמאות

לצורך הפשטות אנו מניחים כי כתובת ה-IP של כל אחד מהצמתים הוא 10.0.0.x כאשר x הוא מספר הצומת. הפורט שמשתמשים בו הוא 1234. לדוגמה, צומת 5 נמצא בכתובת 10.0.0.5 בפורט 1234.

**שימו לב**: כתובות ב-IP והפורטים בבדיקה יתקבלו מהמשתמש ולא יהיו בהכרח כמו בדוגמה.

צבעים בשימוש: רקע ירוק – קלט מהשתמש, רקע צהוב – פלט מהצומת.

## חיבור צומת 3 לצמתים 2, 5

From the user blue

setid,3

ack

connect,10.0.0.5:1234

הודעת connect מצומת 3 לצומת 5:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| MSG ID  (4 bytes) | Source ID  (4 bytes) | Destination ID  (4 bytes) | # Trailing Msg  (4 bytes) | Function ID  (4 bytes) | Payload  (492 bytes) |
| 1 | 3 | 0 | 0 | 4 |  |

הודעת ack מצומת 5 לצומת 3:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| MSG ID  (4 bytes) | Source ID  (4 bytes) | Destination ID  (4 bytes) | # Trailing Msg  (4 bytes) | Function ID  (4 bytes) | Payload  (492 bytes) |
| 10 | 5 | 3 | 0 | 1 | 1 |

Ack

5

connect,10.0.0.2:1234

הודעת connect מצומת 3 לצומת 2:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| MSG ID  (4 bytes) | Source ID  (4 bytes) | Destination ID  (4 bytes) | # Trailing Msg  (4 bytes) | Function ID  (4 bytes) | Payload  (492 bytes) |
| 2 | 3 | 0 | 0 | 4 |  |

הודעת ack מצומת 2 לצומת 3:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| MSG ID  (4 bytes) | Source ID  (4 bytes) | Destination ID  (4 bytes) | # Trailing Msg  (4 bytes) | Function ID  (4 bytes) | Payload  (492 bytes) |
| 10 | 2 | 3 | 0 | 1 | 2 |

Ack

2

## שליחת הודעה מצומת 1 לצומת 2

לפי הדוגמה, צומת 1 צמוד לצומת 2. אנו מניחים שתהליך החיבור בין הצמתים בוצע קודם להרצה הנוכחית.

send,2,12,hello world!

הודעת send מצומת 1 לצומת 2:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| MSG ID  (4 bytes) | Source ID  (4 bytes) | Destination ID  (4 bytes) | # Trailing Msg  (4 bytes) | Function ID  (4 bytes) | Payload  (492 bytes) | |
| 221 | 1 | 2 | 0 | 32 | 4B | 12B |
| 12 | Hello world! |

הודעת ack מצומת 2 לצומת 1:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| MSG ID  (4 bytes) | Source ID  (4 bytes) | Destination ID  (4 bytes) | # Trailing Msg  (4 bytes) | Function ID  (4 bytes) | Payload  (492 bytes) |
| 12 | 2 | 1 | 0 | 1 | 221 |

ack

## שליחת הודעה מצומת 2 לצומת 5

ההנחה היא כי כל הצמתים חוברו קודם, מוצג כאן רק הודעות שצומת 1 מעורב בהם.

send,5,12,hello world!

הודעת discover מצומת 2 לצומת 1:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| MSG ID  (4 bytes) | Source ID  (4 bytes) | Destination ID  (4 bytes) | # Trailing Msg  (4 bytes) | Function ID  (4 bytes) | Payload  (492 bytes) |
| 3 | 2 | 1 | 0 | 8 | 5 |

הודעת discover מצומת 2 לצומת 3:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| MSG ID  (4 bytes) | Source ID  (4 bytes) | Destination ID  (4 bytes) | # Trailing Msg  (4 bytes) | Function ID  (4 bytes) | Payload  (492 bytes) |
| 4 | 2 | 3 | 0 | 8 | 5 |

הודעת discover מצומת 2 לצומת 4:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| MSG ID  (4 bytes) | Source ID  (4 bytes) | Destination ID  (4 bytes) | # Trailing Msg  (4 bytes) | Function ID  (4 bytes) | Payload  (492 bytes) |
| 5 | 2 | 4 | 0 | 8 | 5 |

הודעת nack מצומת 1 לצומת 2:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| MSG ID  (4 bytes) | Source ID  (4 bytes) | Destination ID  (4 bytes) | # Trailing Msg  (4 bytes) | Function ID  (4 bytes) | Payload  (492 bytes) |
| 3 | 1 | 2 | 0 | 2 | 3 |

הודעת route מצומת 4 לצומת 2:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| MSG ID  (4 bytes) | Source ID  (4 bytes) | Destination ID  (4 bytes) | # Trailing Msg  (4 bytes) | Function ID  (4 bytes) | Payload  (492 bytes) | | | |
| 100 | 4 | 2 | 0 | 16 | 4B | 4B | 4B | 4B |
| 5 | 2 | 4 | 5 |

הודעת route מצומת 3 לצומת 2:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| MSG ID  (4 bytes) | Source ID  (4 bytes) | Destination ID  (4 bytes) | # Trailing Msg  (4 bytes) | Function ID  (4 bytes) | Payload  (492 bytes) | | | |
| 321 | 3 | 2 | 0 | 16 | 4B | 4B | 4B | 4B |
| 4 | 2 | 3 | 5 |

הודעת relay ו- send )משורשרות( מצומת 2 לצומת 5 דרך צומת 3:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| MSG ID  (4 bytes) | Source ID  (4 bytes) | Destination ID  (4 bytes) | # Trailing Msg  (4 bytes) | Function ID  (4 bytes) | Payload  (492 bytes) | |
| 431 | 2 | 3 | 1 | 64 | 4B | 4B |
| 5 | 1 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| MSG ID  (4 bytes) | Source ID  (4 bytes) | Destination ID  (4 bytes) | # Trailing Msg  (4 bytes) | Function ID  (4 bytes) | Payload  (492 bytes) | |
| 432 | 2 | 5 | 0 | 32 | 4B | 12B |
| 12 | Hello world! |

הודעת ack מצומת 3 לצומת 2:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| MSG ID  (4 bytes) | Source ID  (4 bytes) | Destination ID  (4 bytes) | # Trailing Msg  (4 bytes) | Function ID  (4 bytes) | Payload  (492 bytes) |
| 322 | 3 | 2 | 0 | 1 | 431 |

ack

route,5

ack

2->3->5

peers

ack

1,3,4

## שאלות נפוצות

1. איך ניתן לקבל הודעות ולשלוח הודעות במקביל? כלומר, במידה ואני שולח הודעה לצומת ובאותו זמן מקבל הודעה מצומת, איך אני אמור לתפקד?

**תשובה:** ישנן כמה אפשרויות ליצר סוקטים שיתפקדו במקביל. אנו מציעים לכם להשתמש בקוד המצורף. ישנן שתי פונקציות שעליכם להשתמש על מנת להצליח למקבל את השימוש בסוקטים:

1. add\_fd\_to\_monitoring
2. wait\_for\_input

הפקודה הראשונה מקבלת כקלט file descriptor או במקרה שלכם סוקט, ומוסיפה אותו לרשימת הFDs שניתן למקבל. הפקודה השניה מכניסה את המחשב למצב בו הוא יכול למקבל בין סוקטים, וכאשר מתקבל קלט מאחד מהם ניתן לבצעו.

במקרה שלנו, הקוד כרגע ממקבל בין סוקטים כאשר בצורה דיפולטיבית הוא מקבל מידע גם משורת הפקודה(command line). פקודות אלו ישמשו אתכם בבניית השלד של הקוד, כך שתוכלו לקבל הודעות/בקשות חיבור מצמתים שונים ולבצעם, ובמקביל לבצע פעולות בצומת שלכם. לדוגמא, אם צומת 3 מנסה להתחבר לצומת 4, ובמקביל צומת 1 מנסה להתחבר לצומת 3-סוקט א' של צומת 3 יאזין לפקודות מהעולם, וסוקט ב' יאזין לפקודות הפנימיות של צומת 3. כך, פקודת החיבור לצומת 4 תופעל בסוקט ב', ובסוקט א' יופעל תתקבל הבקשה מצומת 1.

1. איך ניתן להריץ את הדוגמה?

**תשובה:** ניתן להריץ את הפקודה:

gcc example.c select.c

בשורת הפקודה שלכם. בדוגמא כאן, המחשב מאזין לשורת הפקודה, ולשני סוקטים בפורטים 5000 ו5001. כל קלט שתשלחו לאחד מהסוקטים/דרך שורת הפקודה, יודפס על המסך. האזנה זו תסתיים לאחר עשרה קלטים.

1. האם צריך להבין את הקוד המצורף?

**תשובה:** אין חובה להבין את הקוד המצורף אלא לדעת להשתמש בפקודות לעיל. כמו כן, אם אתם רוצים למקבל את הפקודות מהcommand line לבין פקודות מצמתים אחרים ע"י קוד אחר, אתם מוזמנים.

1. איך ניתן להטמיע את הקוד שנתתם בתרגיל שלנו?

**תשובה:** קבצי הselect השונים נותנים לכם את האפשרות להשתמש בשתי הפקודות הנ"ל על מנת למקבל את השימוש בסוקטים והקלט מהמשתמש. כאשר תכתבו את הקוד שלכם, החליפו את example.c מדוגמת הרצת ה-gcc לעיל בשם קובץ הקוד שלכם, וכך תוכלו להשתמש בפקודות לעיל.

1. מהו השימוש בtrailing msg#?

**תשובה:** השימוש הוא במקרה ומשרשרים הודעות, כלומר-שולחים מספר הודעות על מנת להעביר הודעה מצומת לצומת שאינו מחובר אליו ישירות, כמו בדוגמה בה צומת 2 מעביר לצומת 5 הודעה דרך 3. הרעיון הוא שיש להעביר הודעה לצומת המחבר, אחרת ההודעה לא תגיע ליעדה. כלומר, צריך להעביר הודעה בה 2 מעביר ל3 את המידע כהודעה ראשונה, וכהודעה משורשרת את ההודעה בה 2 מעביר ל5 הודעה. בנוסף, במידה ושולחים הודעה שהיא ארוכה מהמקום הזמין ב-payload, המשך ההודעה יופיע במנה הבאה.

1. איך מייצרים msg ID?

**תשובה:** יש לייצר מספר בצורה שאיננה חוזרת על עצמה בגבולות גודל המשתנה(4 בתים), ניתן להשתמש במזהה אקראי (random), סידרתי או אחר, כל זמן שאיננו חוזר על עצמו.

1. האם ההודעות משורשרות, או נמצאות אחת בתוך השניה?

**תשובה:** הכוונה היא שהן משורשרות, כלומר-הודעות עוקבות שנשלחות יחד.