תרגיל מעשי - מבוא לתקשורת נתונים תש"ף
שרת פריצת סיסמאות מבוזר (hash cracker)

גירסא 1.1 - תודה להערות של גל, אמיר, חן והוד

מבוא

פונקציית hash היא פונקציה המקבלת קלט באורך לא מוגבל ופולטת פלט באורך קבוע. פונקציית hash קריפטוגרפית היא כזו שקשה "להריץ מהסוף להתחלה" - ממש בגדול, בהינתן פלט של הפונקציה, קשה למצוא את הקלט שמתאים לפונקציות כאלו, ולכן יש פשוט להריץ את הפונקציה שוב ושוב עם קלטים שונים עד שמוצאים את הקלט הנכון:

https://en.wikipedia.org/wiki/Cryptographic_hash_function

יש שימוש רב לפונקציות כאלו בתחום אבטחת המידע, ובפרט בתחום איחסון הסיסמאות. פונקציית hash נפוצה ופופלרית היא הפונקציה sha-1. לרוב הפונקציה מקבלת מחרוזת ומוציאה פלט בן 160 ביטים, שמיוצג ע"י קידוד hex כמחרוזת טקסט בת 40 תווים, הנה כך:

```
yos@orenlab96:~$ echo -n shade|sha1sum
7f5bb03cf507c861269be561971108be8f37d832  -
yos@orenlab96:~$ echo -n viper|sha1sum
a346f3083515cbc8ca18aae24f331dee2d23454b  -
yos@orenlab96:~$ echo -n ?????|sha1sum
9017347a610d1436c1aaf52764e6578e8fc1a083  -
```

(הערה: את המחרוזת השלישית אני לא מגלה לכם, נראה אם תצליחו לנחש לבד 😎)

המטרה שלכם היא לבנות מערכת מבוזרת שתקבל פלט של הפונקציה sha-1 ותחזיר את המחרוזת שגרמה לפלט הזה. כדי שזמן הריצה לא יהיה מטורף נתון לכם שהקלט מורכב אך ורק מהאותיות הקטנות a-z בלי רווחים, ושאורך המחרוזת ידוע לכם מראש.

מבנה מערכת כללי

המערכת מורכבת מלקוח וכמות לא מוגבלת של שרתים. הלקוח מקבל מהמשתמש את הפלט של פונקציית ה-hash, בצורה של 40 תווים בפורמט hex, ואת אורך המחרוזת של הקלט. אח"כ הלקוח מנסה לחפש שרתים. אחרי שמצא כמה שרתים הוא מעביר לכל אחד את הפרמטרים שקיבל מהמשתמש ותחום קלטים שיש לחפש בו (למשל, שרת אחד יחפש קלט בתחום aaa-gzz והשני בתחום haa-zzz). השרתים יעברו על כל הקלטים האפשריים בתחום שקיבלו ויחזירו ללקוח את הקלט שמתאים ל-hash, אם מצאו, או הודעת שגיאה אם לא מצאו. אחרי שהשרתים מסיימים את הסריקה הלקוח ידפיס על המסך את הקלט, או לחילופין ידפיס הודעת שגיאה.

הרצה לדוגמא

1. המשתמש מפעיל לקוח אחד וכמה שרתים
2. הלקוח מדפיס על המסך:

```
Welcome to <your-team-name-here>. Please enter the hash:
```

3. המשתמש מקליד את ה-hash:

```
422ab519eac585ef4ab0769be5c019754f95e8dc
```

4. הלקוח מדפיס על המסך:

```
Please enter the input string length:
```

5. המשתמש מקליד את אורך המחרוזת:

```
6
```

6. הלקוח שולח הודעת DISCOVER
7. שני שרתים עונים להודעה ע"י שליחת הודעת OFFER: שרת 1 ושרת 2.
8. הלקוח שולח לשרת 1 הודעת REQUEST על מחרוזת באורך 6 בתחום בין aaaaaa ל-mzzzzz
9. הלקוח שולח לשרת 2 הודעת REQUEST על מחרוזת באורך 6 בתחום בין naaaaa ל-zzzzzz (שתי ההודעות נשלחות במקביל)
10. שרת 1 לא מצא את ה-hash ושולח ללקוח הודעת NACK
11. שרת 2 מצא את ה-hash ומחזיר ללקוח הודעת ACK
12. הלקוח מדפיס את התשובה למסך:

```
The input string is tashaf
```

צד שרת

השרת מאזין ב-UDP לפורט 3117. כאשר הוא קולט הודעת DISCOVER הוא שולח הודעת OFFER. כאשר הוא קולט הודעת REQUEST הוא מבצע את הסריקה עבור כל המחרוזות בתחום הנתון. אם הוא מצא את ה-hash המבוקש הוא מחזיר ACK, אחרת הוא מחזיר NACK

צד לקוח

הלקוח מאזין לקלט מהמקלדת של המשתמש ואוסף את ה-hash ואת אורך המחרוזת. אח"כ הוא שולח הודעת DISCOVER, מחכה לקבל הודעות OFFER במשך שנייה אחת, ואוסף במבנה נתונים את הכתובות של כל השרתים שענו לו. אחרי שנייה הוא מחלק את הקלט לתחומים ושולח לכל שרת הודעת REQUEST עבור חלק ממרחב הקלטים האפשריים. אח"כ הוא מחכה עד שיקבל הודעת ACK אחת לפחות, או לחילופין הודעות NACK מכל השרתים. את הפלט של הריצה הוא מדפיס למסך.

מבנה הודעות

בסה"כ יש מבנה אחד שמשמש לכל ההודעות, ואלו הם שדותיו. הודעות בהן לא עושים שימוש בשדה מסויים עדיין יכללו את השדה, אבל המקלט יידע להתעלם ממנו.

Team Name - exactly 32 chars (256 bits)

מכיל את שם הקבוצה, מרופד ברווחים כך שאורכו יהיה בדיוק 32 תווים. פרס מיוחד יוענק לצוותים עם השם הסייברי ביותר.

Type - 1 char (8 bits), יכול לקבל את הערכים:

- 1=discover
- 2=offer
- 3=request
- 4=acknowledge
- 5=negative acknowledge

הכוונה לערך בינארי ולא ל-string

Hash - 40 chars (320 bits)

מכיל את ה-hash שרוצים לפרוץ בפורמט hex בהודעות 3,4,5

Original Length - 1 char (8 bits)

מכיל את אורך הקלט בהודעות 3,4,5. הכוונה לערך בינארי ולא ל-string.

Original String Start - 1-256 chars

מכיל את המחרוזת שצריך להתחיל לחפש ממנה בהודעה מספר 3

מכיל את מחרוזת הקלט המקורי שנמצאה בהודעה מספר 4

Original String End - 1-256 chars

מכיל את המחרוזת שצריך לחפש עד אליה (ועד בכלל) בהודעה מספר 3

ההודעות תמיד משודרות מעל UDP, כאשר השרת מאזין בפורט 3117 והלקוח בכל פורט פנוי. הודעות discover נשלחות מעל IP broadcast (כתובת 255.255.255.255) וכל שאר ההודעות נשלחות מעל IP Unicast רגיל. ההודעות נשלחות כבלוק של ביטים בלי מפרידים (חובבי ה-python מוזמנים לעיין בפונקציה struct.pack).

רמזים ועצות

1. על כל הטקסט להיות מקודד בקידוד UTF-8
2. שפת התיכנות יכולה להיות python, C# או java, כל עוד הקוד מקורי ואתם יכולים להסביר בו כל שורה.
3. לשם נוחות המימוש ה-hash מיוצג בפורמט hex. הביצועים של הקוד שלכם יהיו טובים בהרבה אם תמירו את המחרוזת לפורמט בינארי של 160 ביט ותבצעו כך את ההשוואות.
4. <u>אין איסור על העתקה</u> (אפילו מומלץ שתחלקו באמצעות הפורום) <u>אך ורק עבור הפונקציות הבאות</u>:
   - פונקציה שמקבלת כקלט מחרוזת UTF-8 ומחזירה את ה-SHA-1 שלה בקידוד hex
   - פונקציה שמקבלת כקלט אורך מחרוזת וכמות שרתים, ומחזירה חלוקה הוגנת לתחומים של כל המחרוזות באורך הנתון
   - פונקציה שמקבלת כקלט תחום מחרוזות, וקוראת לפונקציה אחרת פעם אחת עבור כל מחרוזת בתחום הנתון

5. כדי לקבל ניקוד מלא על התוכנה שלכם:
   - לתמוך ב-threads ולאפשר לכמה משתמשים להתחבר לשרת בו זמנית
   - להראות תאימות - השרת שלכם צריך לעבוד עם לקוחות של תלמידים אחרים ולהיפך
   - בצד השרת - להגביל בזמן את משך הריצה, ולהיכשל אם לוקח יותר מדי זמן. משך ההגבלה צריך להיות פרמטר שקל לשנות, עם ערך default הגיוני, נניח 10 שניות.

- בצד הלקוח - להגביל בזמן את משך הזמן שמחכים לתגובה של השרת, ולנתק אותו אם לוקח יותר מדי זמן. משך ההגבלה צריך להיות פרמטר שקל לשנות, עם ערך default הגיוני, נניח 15 שניות.
- לטפל בצורה טובה במקרי קצה - קלט לא חוקי, ניתוקים באמצע השיחה, וכו