# סיכום פרויקט מערכת קריפטוגרפיה

מגישים: איקה בר-מנחם, ניר חמד

## מבוא:

הפרויקט מממש את מערכת ההצפנה של פילייר המאפשר למצביעים להצביע באופן דיסקרטי בין שתי אפשרויות ולחשב את תוצאות ההצבעה ללא חשיפת ההצבעה הפרטית של המצביעים.

מערכת ההצפנה נכתבה בשפת Java בסביבת eclipse. מרבית התיעוד מבוצע בקוד עצמו ובקבצי html המופיעים בפרויקט עצמו.

מסמך זה מהווה תקציר לפרויקט ופורט את היבטי המרכזיים בהיבט של מימוש, יעילות ובטיחות.

## בטיחות סמנטית

חסינות המערכת מתבססת על ההנחה שחישוב מחלקת השאריות הn-ית היא בעיה קשה חישובית. בעיה זו מוכרת כ (Composite Residuosity(CR))

### עמידה כנגד התקפת הודעה נבחרת: המערכת תעמוד כנגד Indistinguishability under chosen-plaintext attack להלן IND-CPA. כלומר עבור שתי הודעות m1, m2 שיוצפנו על ידי המערכת, התוקף לא יוכל להבחין עבור אחת הקריפטוגרמות האם היא הצפנה של m1 או של m2. הסבר: במערכת שלנו, הבחנה בין ההודעות ב IND-CPA שקולה לבעיית CR ולכן קשה.

## תכונות הומאמומרפיות

המערכת מנצלת את התכונה המאומורפית הבאה:  
 במקרה שלנו ההודעות הן 0 או 1 ולכן ניתן לנצל תכונה זו לסכימת כמות האחדות מבין כל ההצבעות שבוצעו.

## כרונולוגיה של המערכת

1. המערכת מבקשת מהמשתמש את מספר הסיביות למפתחות ההצפנה.
2. המערכת מבקשת את מספר המצביעים.
3. המערכת מגרילה מספרים ראשוניים ויוצרת מפתחות ציבוריים ופרטיים.
4. המערכת יוצרת מערך של מצבעים אשר ביצירתם מגרילים אקראית הצבעה (פרטית) של אפס או אחד בהתפלגות שווה.
5. המערכת מגדירה משתנה אשר מונה את מספר המצביעים שהצביעו 1.
6. המערכת עוברת על מערך הבוחרים ומבקשת מהבוחר את הצבעתו (מוצפנת כמובן).
7. ההצבעות מוכפלות (מודולו n^2 , לפי הפרוטוקול) ונסכמות לתוך המשתנה.
8. בסיום התהליך מתבצע פענוח של המשתנה על ידי המפתח הציבורי.
9. הדפסת דוח על גבי המסך לגבי תוצאות ההצבעה וזמני ריצת התכנית.

## הערות לגבי בטיחות המערכת:

1. בטיחות המערכת מתבססת על
2. ההצבעה של הבוחר אנונימית לגמרי והמשתנים מוגדרים כ private.
3. הצבעת הבוחר מודפסת לlogger לצורכי בקרה בלבד. מחיקת שורות הlogger יבטיחו את בטיחות המערכת בהתבסס על נכונות צופן Paillier.

## הערות כלליות לגבי יעילות:

1. כל הפונקציות המתמטיות הן סטטיות, נעשה ניסיון לצמצם יצירת אובייקטים מיותרים.
2. מהירות המערכת מושפעת גם מקריאה לlogger אשר לוקח משאבי מערכת.

## פונקציות אריתמטיות

1. הגרלת מספר אקראי, **generatePrime**  
   **קלט:** ללא, **פלט:** מספר אקראי בגודל מספר הביטים של המערכת מסוג p=4k+3.  
   **סיבוכיות זמן ריצה:** O(numOfBits)  
   **הסבר על מימוש:** הגרלת מספר בן X ביטים, ובדיקה האם הוא ראשוני מהצורה הדרושה.

מתוך משפט על הצפיפות של הראשוניים, וההתפלגות של הראשוניים מהצורה הנ"ל ניתן לומר שנדרשות סדר גודל של X2 הגרלות על מנת לקבל מספר שהוא ראשוני בהסתברות גבוהה.

1. האם ראשוני, **isPrime**  
   **קלט:** מספר חיובי מסוג BigInteger, **פלט:** true אם המספר ראשוני אחרת false  
   **סיבוכיות זמן ריצה:**לכל היותר gcd יחיד ושלוש העלאות בחזקה מודלול לכן O(numOfBits)  
   מומשה הפרוצדורה שנלמדה בכתה כמבחן ראשוניות.
2. **הערה:** העלאה בחזקה מודולו m, **powerMod**  
   **קלט:** a- המספר אותו מעלים בחזקה, n- החזקה, m מודולו (כל המשתנים הם BigInteger)  
   **פלט:** a^n mod m  
   **סיבוכיות זמן ריצה:**O(log (exponent))

**הערה:** המימוש הנאיבי אינו עומד בזמני ריצה טובים, כאן נדרשה שיטה יעילה בזמן ריצה ובזיכרון

1. מחלק משותף מקמסימלי **gcd**  
   **קלט:** שני מספרים חיוביים מסוג BigInteger **פלט:** המחלק המשותף המקסימלי של שני המספרים, גם כן מסוג BigInteger.  
   **סיבוכיות זמן ריצה:** O(numOfBits^3)  
   **הערות:** מימוש אלגוריתם אוקלידיס על ידי רקורסית זנב.
2. חישוב הופכי בשדה g\*\_n **calculateInverse  
   קלט:** number – המספר לו אנו רוצים הופכי, n – המודולו. **פלט:** ההופכי של number מודולו n.  
   **סיבוכיות זמן ריצה:** O(numOfBits^3)  
   **הערות:** שימוש באלגוריתם אוקלידיס המורחב.
3. הגרלת מספר מודולו n **randomZStar**  
   **קלט:** מספר חיובי n. **פלט:** מספר אקראי מ Z\*\_n  
   **סיבוכיות זמן ריצה:** O(numOfBits^3)  
   **הערות:** המימוש מתבסס על הגרלה של סיבית סיבית (כString) כך שהמספר המוגרל קטן שווה ל n. לאחר מכן מתבצעת בדיקה שאכן המספר המוגרל שייך לקבוצה. במידה ולא התהליך יבוצע מחדש. אחרת יוחזר כפלט.

## השוואת זמני ריצה מול מימוש של פונקציות מערכת

|  |  |  |  |
| --- | --- | --- | --- |
| גודל המפתח | כמות מצביעים | זמן ריצה מימוש שלנו | זמן ריצה מימוש עם פונקציות |
| 256 | 10 | 0.524 שניות | 9.661 שניות |
| 1024 | 10 | 35.887 שניות | 9.96 |
| 1024 | 100 | 158.669 | 69.058 |
|  |  |  |  |

## יצור המפתחות

1. המפתח הפרטי:
   1. משתנים:
   2. הפונקציה L מתוארת על ידי L(u) = (u-1)/n
   3. המפתח הפרטי מיוצג על ידי האובייקט privateKey. הבנאי מקבל כקלט את lambda, mu, n ומשכפל אותם בבנאי. כיוון שהמשתנים פרטיים הם אינם נגישים מרגע היצירה.
   4. למפתח הפרטי יש שיטה ציבורית המקבלת כקלט קריפטוגרמה ומחזירה את ההודעה ע"י אלגוריתם פלייר ובשימוש הפונקציות המתמטיות ב MyPaillier
2. המפתח הציבורי:
   1. המפתח הפרטי מיוצג על ידי אובייקט publicKey. הבנאי מקבל כקלט את n, g ומשכפל אותם בבנאי.
   2. לאובייקט שיטה encode אשר מקבלת כקלט הודעה ומחזירה את ההודעה מוצפנת על ידי המפתח הפרטי.
3. הפעלת הפונקציה generateKey מחזיר מערך בגודל שתיים של מפתח פרטי ומפתח ציבורי.

## ביבליוגרפיה

1. ויקיפדיה <http://en.wikipedia.org/wiki/Paillier_cryptosystem>
2. ויקיפדיה <http://en.wikipedia.org/wiki/IND-CPA>
3. Jonathan Katz and Yehuda Lindell. Introduction to Modern Cryptography. Chapman & Hall/Crcyptography and Network Security Series, pages 408-417
4. Pascal Paillier, Public-Key Cryptosystems Based on Composite Degree Residuosity classes, EuroCrypt' 99, vol 1592 of Lecture Notes in Computer Science, pp. 223-238, Springer-Verlag, 1999.