אבטחת מערכות מידע

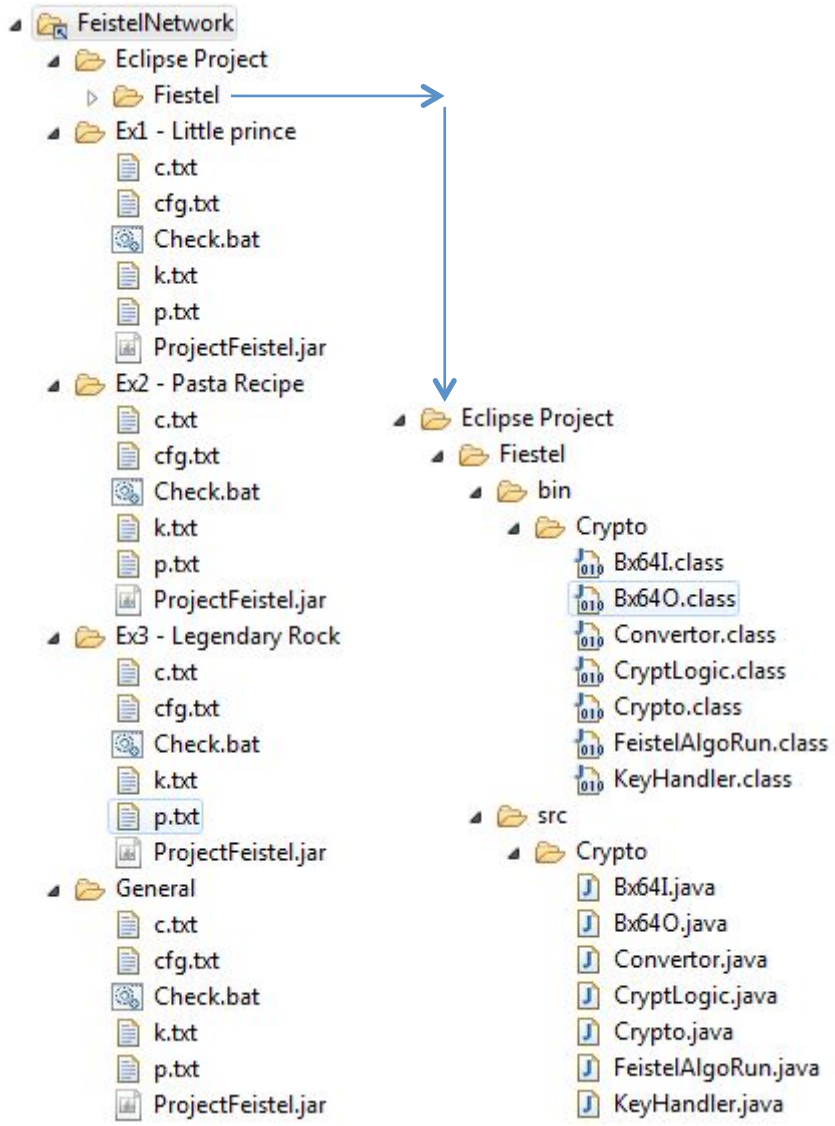
פרויקט Feistel Network

מגישים:

- מתן גידניאן 200849605
- אשר הולדר 322057712

תאריך הגשה: 18.01.2015

הקבצים המצורפים בתרגיל זה:



תיקיות Examples מכילות את 3 הדוגמאות.

תיקיית General מאפשרת הרצה של טקסטים לבחירתך.

תיקיית Eclipse Project מכילה את קבצי המקור, את כל הפרויקט מסביבת העבודה שלנו.
התייחסות מיוחדת לתיקייה זו בהמשך המסמך.

הוראות הפעלה:

1. **הפעלה של הדוגמאות**

ישנם 3 תיקיות Example.
בתוך כל תיקייה יש את כל הקבצים הנדרשים להרצת אותה הדוגמא.
כל שצריך לעשות הוא לשנות את הקובץ cfg.txt בהתאם למה שרוצים לעשות, כלומר לבחור בין: encrypt, decrypt, verify > לעדכן בקובץ ולשמור אותו.
כעת להריץ את הקובץ Check.bat שיפעיל את הJAR בתיקייה בקלות.

2. **הפעלה על קבצים שאינם מהדוגמאות**

ניגשים לתיקייה General, ומוסיפים את הקבצים הרצויים.
p.txt טקסט לפני הצפנה
c.txt טקסט לאחר הצפנה
משנים את הקובץ cfg.txt בהתאם למה שרוצים לעשות, כלומר לבחור בין: encrypt, decrypt, verify > לעדכן בקובץ ולשמור אותו.
כעת להריץ את הקובץ Check.bat שיפעיל את הJAR בתיקייה בקלות.

הערה
בקובץ cfg.txt נא לכתוב באותיות קטנות (lower case).
כמו כן, קובץ זה מוגדר תחילה ל verify.

הדוגמאות שנבחרו:

Example 1 חלק מהספר הנסיך הקטן
Example 2 מתכון מנצח לפסטה! (כי מי לא אוהב פסטה ?)
Example 3 מילים לאחד השירים היותר מפורסמים... LED ZEPPELIN - Stairway To Heaven

על הפרוייקט

טכני

הפרויקט נבנה בסביבת Eclipse
הבנייה בוצעה בסביבת Java 7
בחר לעבוד בעיקרון המודולריזציה על מנת שהקוד יהיה ברור יותר, ויקל על הליך הכתיבה שלו.
הפרויקט מבוסס על DES שכן הוא מימוש של Feistel Network, ביצענו מימוש בהתאם למידע שמצאנו בויקיפדיה בעיקר ומספר מקורות קטנים יותר. Feistel, DES#2, DES#1

מחלקות

- Bx64I, Bx64O
מחלקות אלו עוטפות את InputStream, OutputStream ומספקות דרך קלה לעבוד עם קלט ופלט בבסיס 64 כפי שנדרש בפרויקט.
קיימים אצלן גם אוגרים פנימיים שמשמשים למניפולציה על המידע, ומטודות שמתקבלות ע"י ירושה וממומשות באופן חדש כדי להתאים לצרכינו.

- Convertor
מחלקה המשמשת לקריאה וכתיבה מבסיס64 ואל בסיס64
2 המחלקות Bx64I, Bx46O משתמשות בה כדי לעבור לבסיס64 וחזרה ממנו.

- CryptLogic
אחת מ2 המחלקות המשמעותיות בתרגיל זה. מספקת את הניהול של ההצפנה והפיענוח, בעוד שהיא מבצעת תיאום מול Bx64I, Bx64O להפעלת האלגוריתם הצפנה.
כמו כן היא אחראית על ולידציה של הקלט שהיא מקבלת.

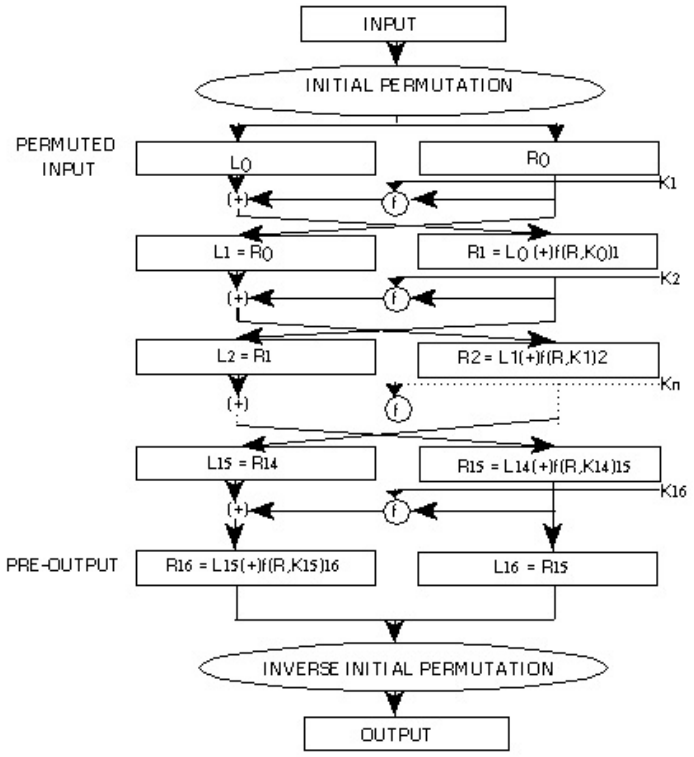
- FeistelAlgoRun
המחלקה השנייה המשמעותית בתרגיל זה. כאן קיימות המתודות שבאמת מבצעות הצפנה/פיענוח, היא עושה זאת עבור בלוק בודד בכל פעם שקוראים לה.

- KeyHandler
משמש על מנת לקרוא ולנהל את המפתח, כמו כן מוודא שהעבודה איתו היא כצפוי לדרישות שאר המחלקות.

- Crypto
מחלקה לקריאת הקונפיגורציה והפעלה של המתודה הרלוונטית כפי שנבחרה.
מוודאת שכל הקבצים הנדרשים קיימים ופולטת שגיאה בהתאם.

הסבר כללי

הפרויקט מבוסס על DES, הצפנה מוכרת שמשתמשת במבנה של Feistel Network.
בחרנו ב DES מכיוון שהוא אלגוריתם קיים ומהווה ניסיון אמיתי בכתיבת הצפנה והבנה של אלגוריתם מסוג זה.
בדרישות הפרויקט קיבלנו מספר ערכים קבועים לשימוש בעת המימוש, שמרנו עליהם כפי שהתבקשנו.
האלגוריתם מחלק את הקבצים השונים (טקסט, הצפנה, מפתח) לחלקים שונים, כלומר ל"בלוקים" (כל חלק בתורו), מבצע על כל בלוק הליך הצפנה.
בהליך ההצפנה, כל בלוק עובר פרמוטציה ראשונית ולאחר מכן את התהליך של פונקציית ההצפנה (שמבוצע בFeistelAlgoRun)



מימין תרשים המציג את התהליך:
לאחר ביצוע הפרמוטציה, בכל איטרציה, הבלוק הימני הנוכחי יבור ללא שינוי ויהפוך לבלוק השמאלי של האיטרציה הבאה.
בעוד הבלוק השמאלי יעבור את פונקציית ההצפנה שמקבלת חלק מהמפתח, את הבלוק הימני. לאחר ביצוע הערבול יבוצע XOR עם הבלוק השמאלי הנוכחי, ולאחר מכן כל המידע הזה יעבור להיות הבלוק הימני של האיטרציה הבאה.
בסוף התהליך נבצע פרמוטציה נוספת שהיא ההפך מהפרמוטציה הראשונית שביצענו.
ניתן לראות בקלות כי מדובר בהליך CBC של עבודה בחלוקת כל המידע לבלוקים.

יעילות

המימוש שלנו יכל להיות יעיל יותר אם היינו עושים הרצה מקבילים של ההצפנה והפענוח של הבלוקים, אך בחרנו לא לעשות זאת מכיוון שזה דורש הרבה מאוד עבודה ונכנס לתחום של safety thread שהופך את כל התרגיל למסובך אפילו יותר.

לאורך הפרוייקט העדפנו לעבוד לרוב עם משתנים מסוג int ו long כדי לפשט את העבודה ואת ההבנה במקום להתעסק עם מערכים של byte וכדומה.

תודה רבה, בדיקה נעימה !