Programmierpraktikum
Thema: Kryptographie
Sommersemester 2004

2. Übungsblatt

1. (*) Folgender Chiffretext wurde mit dem Substitutionsverfahren verschlüsselt:

   ```
   agz pbmqhslxmwk dstzmw bsz kmz xhgepmxhm xshp, kmw mqw
   ymwxjt xsimw ngwwhm, kmz: qjt cqw mqw clmzimz zgyx.
   tmlhm qxh kmz xhgepmxhm xshp, kmw dmyswk qw kmz vzmqmw
   bmeh xsimw nsww: qjt cqw mqw cmzeqwmz.
   ```

   Welcher Schlüssel wurde benutzt und wie lautet somit der zugehörige Klartext?

2. (Oliver/Philipp/Tim) Im package `crypto.decrypt` ist die Klasse `Analyze` zu vervollständigen. Alle Methoden arbeiten auf der Klassenvariablen `ciphertext`. Die zwei Methoden

   ```
   public void analyzeBigrams(); und
   public void analyzeTrigrams();
   ```

   suchen nach auftretenden Bi- und Trigrammen im gegebenen Chiffretext. Die Prozentzahlen für jedes Bi- sowie Trigramm sollen in den Variablen `bigramChart` bzw. `trigramChart` gespeichert werden. Für die zehn häufigsten Bi- und Trigramme stehen jeweils zwei Arrays in der Klasse zur Verfügung:

   ```
   bigrams   - bigramsFreq        sowie
   trigrams  - trigramsFreq
   ```

   In dem ersten Array wird der explizite String gespeichert, in dem zweiten die dazugehörige Prozentzahl. Alle Prozentzahlen sollen auf zwei Stellen nach dem Komma gerundet werden.

3. (Alex/Oleg/Yves) Für den Kasiski-Test wird die Funktion

   ```
   public void findLetterChains();
   ```

   benötigt. Diese soll die 10 häufigsten Buchstabenketten im Chiffretext `ciphertext` finden und in der Variablen `repeatedStrings` speichern. Dabei sollen die Ketten absteigend ihrer Länge nach geordnet werden. Die Abstände zwischen zwei gleichen Ketten sollen in der Variablen `repeatedStringsDistance` festgehalten werden. Zuletzt soll die Methode

   ```
   public int suggestKeyLength();
   ```

   implementiert werden. Diese rät die Länge des benutzten Schlüsselwortes anhand der gefundenen Abstände der Buchstabenketten (ggT aller Abstände wäre die einfachste Lösung).

4. (David/Pavel) Folgende Änderungen müssen beim GUI durchgeführt werden:

   - Logo anpassen
   - Shortcuts im MessageWindow einbauen (z.B. Ctrl-ENTER für Versenden)
   - Send-Button im MessageWindow auf disabled, falls kein Schlüssel vorhanden
   - Buttons ”Enter Own Key” und ”Set Insecure Key” im KeyFrame entfernen und stattdessen...
   - ...einen Menupunkt ”Send Challenge” einbauen, welcher bei einem Rechtsklick auf einen Buddy erscheinen soll

   Beim Auswählen von ”Send Challenge” soll außerdem ein Popup mit einem editierbaren Textfeld erscheinen. Für das Textfeld sollen dieselben Regeln gelten wie vorher für das Textfeld im MessageWindow, wenn ein unsicherer Schlüssel ausgewählt wurde (u.a. nur Kleinbuchstaben). Eine ComboBox mit den vier Verfahren

   - Caesar
   - Substitution
   - Vigenere

   soll im Popup enthalten sein sowie ein Button zum versenden der Challenge.

5. (David/Pavel) Das Verschicken von ”Challenges” soll implementiert werden. Diese sind von der Form:

```
<challenge>
<id>0</id>
<from>1</from>
<to>3</to>
<text>ejft jtu fjo uftu</text>
<cryptomethod>Caesar</cryptomethod>
<clearlength>0</clearlength>
</challenge>
```

   Nach dem Dechiffrieren muß eine Benachrichtigung an den Urheber der Challenge gesendet werden. Diese Antwort hat folgendes Format:

```
<solution>
<id>0</id>
<from>1</from>
<to>3</to>
<text>dies ist der klartext</text>
<cryptomethod>Caesar</cryptomethod>
<key>21</key>
</solution>
```

Abgabe: bis Freitag, 4.6.2004