Cryptographics Pflichtenheft

18. November 2013

Inhaltsverzeichnis

1 Zielbestimmung

Im Rahmen der PSE-Veranstaltung soll für das Kryptologikum des Instituts für Kryptographie und Sicherheit eine Software (Cryptographics / Anicrypto?) zur Demonstration kryptographischer Verfahren erstellt werden.

Das Programm soll im Laufe der Ausstellung Kryptologikum das Interesse der Besucher wecken, sich mit Verschlüsselung zu befassen und schließlich auch ausgewählte Inhalte der Kryptographie näher bringen

Soll Spaß machen.

1.1 Musskriterien

- Implementierung der Caesar-Chiffre
- Implementierung Vigen‘ere-Chiffre
- Implementierung RSA
- Implementierung Diffie-Hellmann
- Implementierung Hashfunktionen
- Robuste Programmierung
- Intuitive Benutzerführung
- Schnelle Reaktionszeiten
- Zugriff auf Krypto-Verfahren über Zeitleiste:
  - Pop-Up mit kurzem Umriss des Verfahrens
  - Draufklicken um zum Verfahren zu Gelangen
  - Einfärbung grün/gelb/rot nach Komplexität des Verfahrens
- Drei Optionen je Verfahren: Demo, Selsbstversuch, weitere Informationen
- Angeleiter Selbstversuch bei "roten"Verfahren
- Interaktion des Benutzers über den Touchscreen
- einfache, übersichtliche UI, die sich gut auf Touchscreen bedienen lässt
- Robustheit gegen außergewöhnliche Interaktion
- Man soll das Programm nicht schließen können währen der Vorführung
- Praxisbezug bei modernen Kryptoverfahren
- Sammlung wiederverwendbare UI-Element um z.B. die Texteingabe gleich zu behandeln

- Krypto-Algorithmen lassen sich schrittweise ausführen, um den Prozess langsam zu verdeutlichen (bspw. bei Caesar: Buchstabe für Buchstabe bis Codewort generiert wurde)
- der Benutzer soll die Möglichkeit bekommen Dinge Schritt für Schritt in seiner eigenen Geschwindigkeit nachzuvollziehen und ggf. zurückspringen können
- Es darf kein Zurückkommen zum Betriebssystem möglich sein
- Visualisierungen sollten Grafiken und Animationen verwenden um Vorgänge zu verdeutlichen

1.2 Wunschkriterien

- Implementierung Public Key – Infrastruktur
- Implementierung Shamir Secret Sharing
- Implementierung Passwortsicherheit
- Implementierung One-Time-Pad
- Implementierung Blockchiffre (DES, AES)
- Bei Problemen im Selbstversuch hilft der Computer mit Tipps aus
- Modulares Austauschen/Hinzufügen von weiteren kryptografischen Verfahren
- Einschätzung des aktuellen Wissenstands des Benutzers.
- Implementierung elliptischer Kurven
- Literaturhinweise(auch weiterführende Literatur), sofern Interesse besteht.
- Visualisierungen sollten möglichst ansprechend sein, indem sie z.B. Analogien verwenden (ein Datenpaket wird zu einer Postkarte, ein Schlüssel wird zum einem tatsächlichen Schlüssel, ...).
- Wenn über einen Zeitraum x keine Nutzereingaben erfolgen erscheint ein Countdown nach dessen Ablauf das Programm auf den Willkommensbildschirm zurücksetzt
- Visualisierung von Modulo mit einer Uhr
- Visualisierung von Angriffen auf Verfahren
- (Blockchiffre im ECB-Modus vs. CBC-Modus, siehe Bild in
- http://de.wikipedia.org/wiki/Electronic_Code_Book_Mode)
- Brute-Force Angriff auf (symmetrisch/asymmetrisch?) Verschlüsselung

- Enigma?
- Zahlentheorie: modulares Rechnen/ Faktorisierungsproblem/ RSA

1.3 Abgrenzungskriterien

- Sämtliche kryptologischen Verfahren werden nur zu Vorführungszwecken implementiert. Eine sichere Implementierung ist nicht vorgesehen.
- Dies ist ein Vorführprogramm für eine Ausstellung, demnach ist eine andere Verwendung nicht ratsam, da dafür ganz andere Anforderungen gelten, die hier nicht beachtet werden. (umformulieren!)
- keine optimierte und 100% standardkonforme Implementierung von Krypto-Algorithmen, sondern Fokus auf schrittweiser Ausführbarkeit
- keine formale Korrektheit bei Erklärungen sondern Ansatz ohne nötige Vorkenntnisse um breiter Masse zugänglich zu sein

2 Produkteinsatz

2.1 Anwendungsbereiche

Cryptographics soll in erster Linie als Ausstellungsstück für das Kryptologikum des Instituts für Kryptographie und Sicherheit (IKS) am Karlsruher Institut für Technologie (KIT) dienen.

Besuchern der Ausstellung soll das Funktionsprinzip und die Verwendung historischer, sowie aktueller, kryptographischer Verfahren näher gebracht werden. Diese sollen anhand von vereinfachten und beispielhaften Szenarien aus dem Alltag vermittelt werden, mit dem Ziel ein größeres Interesse an der Materie zu wecken.

Cryptograhics soll primär auf dem eeePC im Kryptologikum eingesetzt werden. Portabilität wäre jedoch wünschenswert.

- Ausstellung.

2.2 Zielgruppen

Das Programm richtet sich insbesondere an Kinder, Jugendliche und Erwachsene mit grundlegenden Kenntnissen der Mathematik. Die Zielgruppe ist demnach ein anonymes Publikum, das sich potenziell aus fachfremden Personen zusammensetzt. Deshalb dürfen für die Verwendung von Cryptographics keine Vorkenntnisse in der Kryptographie vorausgesetzt werden.

- Besucher einer kryptographischen Ausstellung mit unterschiedlichem Wissensstand und Interesse.

2.3 Betriebsbedingungen

Das Tool soll im Dauerbetrieb problemlos laufen. Es dürfen keine Ausnahmen auftreten, die den Betrieb des Tools behindern, oder gar abstürzen lassen und auf den Desktop zurückzukehren. Der Betrieb soll für den eeePC optimiert sein.

Optimale Ausführung auf gegebener Hardware (Intel Atom, Touchscreen)

-Einsatz im Rahmen einer Ausstellung.
Kein Abstürzen bei unkontrollierten, „wilden“ Nutzereingaben
System darf nicht manipuliert werden, d.h. keine oder nur geschützte Möglichkeit um Programm zu beenden

reine Offline-Anwendung. Darf keine Internetanbindung benötigen

3 Produktumgebung

Software/Hardware... => Weiß ich grad nicht. XP auf nem eeePc Einsatz auf eeePC wieauchimmer (keine Ahnung wie die Kiste heißt) - Produktumgebung ist ein Windows-PC mit reisistivem Touchscreen und Windows XP

3.1 Software

3.2 Hardware

4 Produktfunktionen

4.1 Funktionale Anforderungen

/FA100/ Darstellung aller Visualisierungen auf einer Zeitleiste

/FA101/ Interaktion mit der Zeitleiste, um eine einzelne Visualiserung auswählen zu können

/FA102/ Darstellung einer Übersichtsseite zur gewählten Visualisierung

/FA103/ Start der gewählten Visualisierung

/FA104/ Abbruch einer gewählten Visualisierung, um zum Startbildschirm zurückzukehren

/FA105/ Darstellung weiterführender Information nach erfolgreichem Beenden einer Visualisierung

/FA100/ Prozess: Diffie-Hellman Schlüsselaustausch

Ziel: Vermittlung des Diffie-Hellman Schlüsselaustauschs mit Farben-Analogie

Vorbedingung: keine

Nachbedingung (Erfolg): Alice und Bob haben sich auf eine gemeinsame, geheime Farbe geeinigt die Eve nicht kennt

Nachbedingung (Fehlschlag): Alice und Bob haben eine gemeinsame, nicht geheime Farbe oder sich auf gar keine Farbe geeinigt

Auslösendes Ereignis: Ausgewählt in der Timeline

Anmerkung: das hier vorgestellte Spiel soll zum einen Demonstriert werden, zum anderen soll es der Nutzer auch selber spielen

Beschreibung:

1. Erkläre das Ziel sich auf ein gemeinsames Geheimnis zu einigen durch Nutzung eines unsicheren Übertragungskanals bei dem jeder Lauschen kann, ohne das diese das Geheimnis erfahren
2. Demonstriere das Prinzip der Einwegfunktion, anhand Farben die man mischen kann, Farben zu mischen ist einfach, zur einer gemischten Farbe herauszufinden welche Farben verwendet wurden hingegen ist schwer
3. Alice und Bob einigen sich auf eine gemeinsame nicht-geheime Farbe A
4. Alice wählt eine geheime Farbe X und mischt sie mit A zur Farbe AX
5. Bob wählt eine geheime Farbe Y und mischt sie mit A zur Farbe AY
6. Alice schickt AX zu Bob u. Bob schickt AY zu Alice
7. Alice mischt AX mit Y zu AXY
8. Bob mischt AY mit X zu AYX
9. Wenn Eve weder X noch Y erfahren hat, kennt Eve nicht AXY

/FA200/ Kryptographische Verfahren auswählen

/FA300/ Funktionen von den jeweiligen Verfahren abhängig

/FA400/ Jederzeit die Möglichkeit für interaktive Hilfe ([?]-Button)

- Funktionen: Willkommensbildschirm, Algo/Visualisierung wählen, Algo bearbeiten, Soforthilfe passend zum Algo falls man nicht weiter weiß, Algo beenden, ggf. Kontrollfragen oder selbstständiges Anwenden des Gelernten am Ende (z.B. ein Wort mit Caesar verschlüsseln)

RSA und Public-Key Verschlüsselung anhand von Pad-Locks als öffentliche Schlüssel und den Schlüssel dazu als privaten, vielleicht mit Komplementärfarben visualisieren ähnlich wie Diffie-Hellman

4.2 Nichtfunktionale Anforderungen

/NA100/ Schnelle Reaktionszeit

/NA200/ Große Robustheit

/NA300/ Intuitive Benutzerführung

- Die Anwendung muss möglichst performant und verzögerungsfrei auf Benutzereingaben reagieren (soweit das die HW zulässt...)

- Die Anwendung muss leicht zu bedienen sein

5 Produktdaten

5.1 Nutzerdaten

- zu speichern sind ggf. Nutzungsstatistiken, um herauszufinden, welche Visualisierung gut ankommt und welche nicht. Dies erlaubt das Ausstellungsstück zielgerichtet weiterzuentwickeln.

6 Benutzerschnittstelle

[Grafiken, Entwürfe]

7 Globale Testfälle

- automatisiertes Testen mithilfe von JUnit vor allem der Krypto-Algorithmen
- ggf. automatisiertes Testen der UI mithilfe von Szenarien und geeignetem Framework Stresstest mit möglichst zufälligen und willkürlichen Eingaben (wie es Kinder eben tun) Usability Tests mit passenden Testpersonen

8 Systemmodelle

8.1 Systemübersicht

Das System verwendet das bekannte MVC-Entwurfsmuster. Weiterhin ist geplant, das System in zwei Schichten zu unterteilen. Von unten nach oben enthält Schicht 1 wiederverendbare und in sich abgeschlossene Bibliotheken. Zum einen handelt es sich hierbei um Code von Dritten (z.B. um QR-Codes zu generieren). Weiterhin ist eine Sammlung wiederverwendbarer Klassen (nachfolgen *CGKit* genannt) geplant. *CGKit* wird vor allem aus UI-Elemente bestehen, die für die Implementierung aller Visualisierungen wertvoll sind.

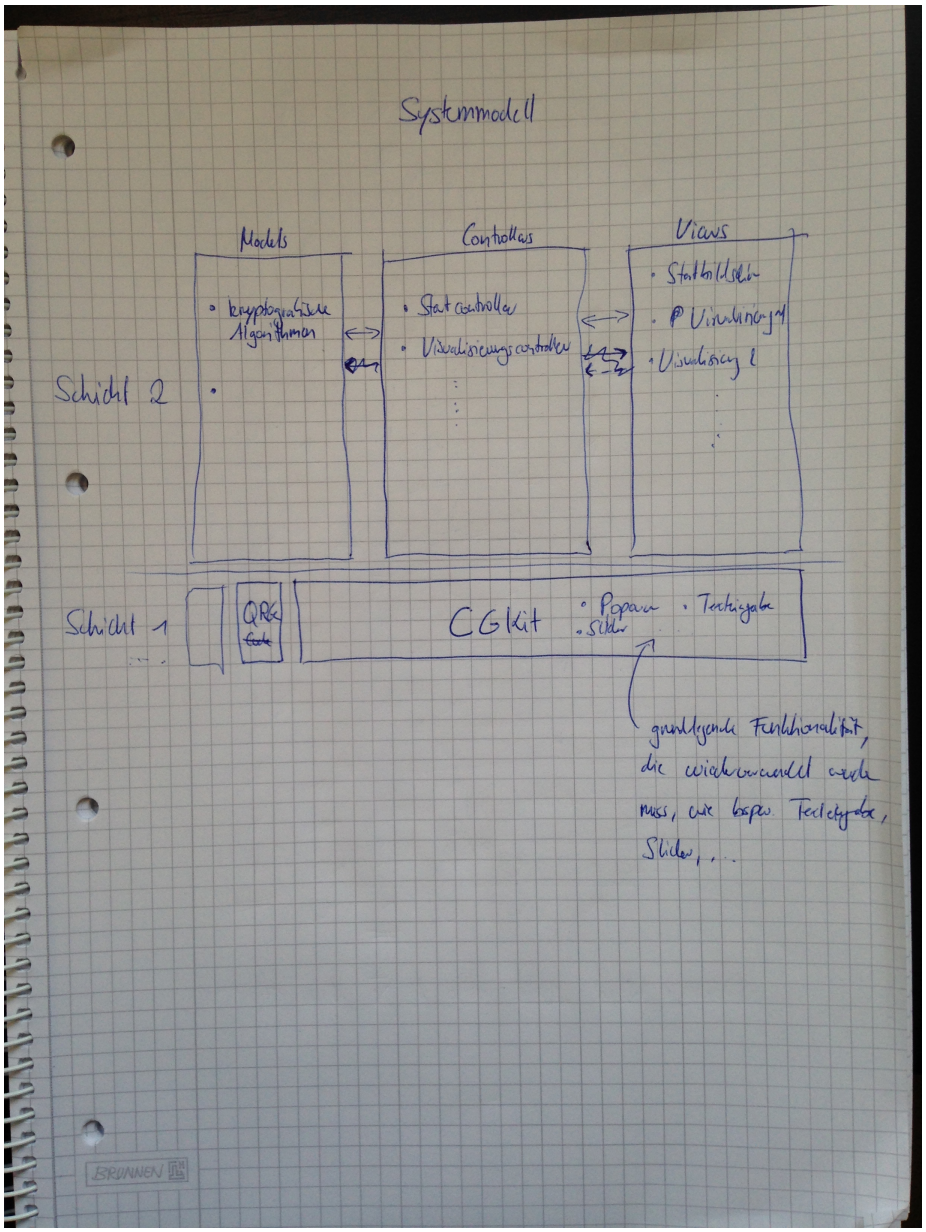

Abbildung 1: Schematische Darstellung des beschriebenen Systems

9 Beispielszenarien

- Beispiel läuft schrittweise ab. Problem wird dargestellt: A möchte B eine Nachricht schicken, die nicht von einer anderen Person als B gelesen wer-

den soll. C versucht die Nachricht zu bekommen. Dabei können A und B nur über ein unsicheres Medium kommunizieren (Analogie z.B. schlüssel per Post verschicken), das von C abgehört wird. itemA und B generieren ihre Schlüsselpaare. Der genaue Schlüssel ist nicht relevant.

- A fragt B nach seinem public key, den B auch bereitwillig mitteilt. C hört den public key von B.
- A verschlüsselt die Nachricht mit dem public key von B und verschickt die Nachricht an B. C hört die verschlüsselte Nachricht ab.
- B entschlüsselt die Nachricht von A mit seinem private key und kann die Nachricht lesen
- C versucht mit der Nachricht irgendetwas anzufangen und versucht sie mit dem abgehörten public key von B zu entschlüsseln. Das scheitert natürlich.

Beispiel einer Visualisierung anhand des Caesar-Algos:

- Es wird erklärt, dass vermutlich Caesar dieses Verfahren verwendet hat um geheime Nachrichten zu verschlüsseln
- Erklärung des Prinzips an einem Beispielsatz. Es werden schrittweise alle gleichen Buchstaben markiert und in ihr Äquivalent umgewandelt (z.B. "Hallo, wie geht's?> zuerst alle H zu K, dann alle a zu d, dann alle l zu o). Hier gibt es die Möglichkeit, schrittweise durchzugehen und noch mal zurück zu gehen. Außerdem kann man wenn man das Prinzip verstanden hat ans Ende springen.
- Als nächstes muss die Person ein Wort selbst verschlüsseln mit Vorschrift (z.B. a -> b).
- Am Ende wird mithilfe eines Diagrammes und der Kenntnis, das E der häufigste Buchstabe im Deutschen ist gezeigt, dass man diese Verschlüsselung leicht umgehen kann

10 Qualitätsbestimmung