3. Übungsblatt
Kryptographie und Mediensicherheit (SoSe 2016)

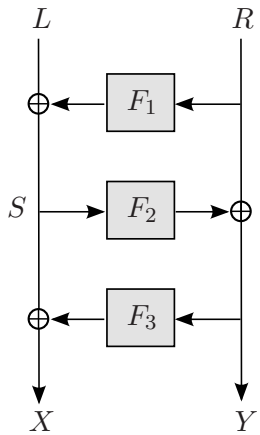
Bauhaus-Universität Weimar, Professur für Mediensicherheit
Prof. Dr. Stefan Lucks, Jakob Wenzel
URL: `http://www.uni-weimar.de/de/medien/professuren/mediensicherheit/teaching`
Abgabe: Bis zum 23.05.2015, 13:30 Uhr zu Beginn der Übung oder per E-Mail an `jakob.wenzel(at)uni-weimar.de`.
Lösungen sind bevorzugt in LaTeX zu verfassen. Ein Template finden Sie auf der Übungsseite der Veranstaltung.

Aufgabe 1 – Luby-Rackoff-/Feistelchiffren (6 Punkte)
Gegeben seien die folgenden Luby-Rackoff-Konstruktionen P3 mit Funktionen $F_1, F_2, F_3 : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$. Es seien K_1, K_2 und K_3 drei unabhängige, zufällig gleichverteilt gewählte und geheime Schlüssel. Beschreiben Sie für jede der untenstehenden Konstruktionen einen möglichst effizienten Chosen-Plaintext-Angreifer und geben Sie seinen Vorteil an.

a) Es gilt: $F_i(K_i, M) = K_i \oplus M$, für alle $1 \leq i \leq 3$.

b) Es gibt ein bekanntes Paar von Eingaben $M \neq M'$, für das gilt: $F_1(K, M) = F_1(K, M')$. F_2 und F_3 sind Zufallsfunktionen.

c) F_1 besitzt die Komplement-Eigenschaft $F_1(M) = \overline{F_1(\overline{M})}$, für alle Eingaben $M \in \{0,1\}^n$. $\overline{M}$ bezeichnet das bitweise Inverse von M. F_2 und F_3 sind Zufallsfunktionen.



Aufgabe 2 – Advanced Encryption Standard (4 Punkte)
Sie haben in der Vorlesung den Advanced Encryption Standard mit den vier Rundenfunktionen `AddRoundKey, SubBytes, MixColumns,` und `ShiftRows` kenenngelernt. Erläutern Sie, inwiefern die Sicherheit des AES beeinflusst wird (inklusive der Beschreibung eines Angreifers und seines Vorteils), wenn die folgenden Fälle eintreten:

a) Die `AddRoundKey`-Operation wird entfernt.

b) Die `SubBytes`-Operation wird entfernt.

c) Die `MixColumns`-Operation wird entfernt.

d) Die `ShiftRows`-Operation wird entfernt.

Aufgabe 3 – Doppelte Verschlüsselung (3 Punkte)
Da die Schlüssellänge des DES zu gering ist, schlägt Alice die Verwendung des 2DES mit zwei unabhängigen 56-bit Schlüsseln K_1 und K_2 vor (siehe Folie 105 der Vorlesung). Beschreiben Sie einen Angriff auf 2DES, der mit Hilfe zweier abgehörter Klartext-Chiffretext-Paare und signifikant weniger als 2^{112} DES-Operationen den korrekten Schlüssel findet. Geben Sie zudem den benötigten Zeit- und Speicheraufwand an.

Aufgabe 4 – DES (4 Punkte)
Zeigen oder widerlegen Sie: Für alle Klartexte $M \in \{0,1\}^{64}$ und alle Schlüssel $K \in \{0,1\}^{56}$ gilt: $\overline{DES_K(X)} = DES_{\overline{K}}(\overline{X})$. Sie brauchen die Initiale und Finale Permutation nicht zu betrachten.

Aufgabe 5 – Entropie (Programmieraufgabe) (4 Punkte)
Schreiben Sie ein Programm in Python, dass die Min- und die Shannon-Entropie der Verteilung von Anfangs-Buchstaben eines Textes berechnet. Ihr Programm soll...

1. den Pfad zu einer Textdatei entgegennehmen (Kommandozeilenparameter),
2. den Inhalt der Textdatei laden und in einzelne Wörter trennen,
3. die Anfangsbuchstaben der Wörter zählen,
4. und die Min- und Shannon-Entropie der Anfangsbuchstaben berechnen.

Als Trennzeichen zwischen Wörtern müssen Sie lediglich ein einzelnes Leerzeichen berücksichtigen. Desweiteren können Sie Sonderzeichen und Zahlen ignorieren und nur Groß- und Kleinbuchstaben betrachten.

Beispielaufruf: (für "The Adventures of Sherlock Holmes" von Arthur Conan Doyle)

```
$ python textentropy.py 1661
Downloading http://www.gutenberg.org/cache/epub/1661/pg1661.txt
A 550
B 373
C 240
...
x 0
y 2006
z 4
Min entropy: 2.88 bit/char
Shannon entropy: 4.51 bit/char
```

Bonus (+1 Punkt): Das Gutenberg-Projekt (`http://www.gutenberg.org`) stellt freie Literatur online zur Verfügung. Anstelle eines Pfads zu einer lokalen Datei soll Ihr Programm die ID eines Buchs entgegennehmen, die UTF8-Textdatei des Buchs von der Gutenberg-Website herunterladen und die heruntergeladene Textdatei als Ausgangstext verwenden. *(Hinweis: Das Modul `urllib` könnte hier hilfreich sein.)*

Aufgabe 6 – (Bonus) Password-Hashing (Programmieraufgabe) (+4 Punkte)
Schreiben Sie ein Programm in Python, welches zu gegebenen Hashwerten die dazugehörigen Passwörter rekonstruiert. Ihr Programm soll...

1. die Hashwerte aus einer Textdatei entgegennehmen (ein Hash pro Zeile, Textdatei als Kommandozeilenparameter),
2. alle **ausschließlich aus Kleinbuchstaben** bestehenden Passwörter mit einer **maximalen Länge von sechs Zeichen** ausprobieren bis alle Passwörter zu den gegebenen Hashwerten gefunden wurden,
3. und am Ende die Passwörter zusammen mit ihren Hashwerten ausgeben.
4. Zusätzlich soll ihr Programm noch die für jeden Hashwerte aufgewandte Zeit angeben.

Die folgenden Hashwerte wurden mithilfe der Einwegfunktion SHA1 errechnet.

Hashwert 0: `a47b5cc8f06168f0ec3832a99894834e1d27f744`

Hashwert 1: `03b1d98b479be6c736b0f649b8fc1197c91fe83f`

Hashwert 2: `f95331f8f170067da0c42e9576138625388407e7`

Hashwert 3: `89543ee17a62627c0f220c3901b7b7f0d4c49413`

Hashwert 4: `7068c63f083322f8a7860175f7c0e2873f530580`

Hashwert 5: `6a3a620aa94bd388ed4d35a6f3845a38e0798aa3`

(Hinweis: Sie müssen die Funktion SHA1 nicht selbst implementieren, sondern können eine vorhandene Implementierung nutzen. Zum Beispiel bietet Ihnen das Modul `hashlib` eine Implementierung von SHA1 an.)

```
import hashlib
print(hashlib.sha1(‘‘hashmich’’).hexdigest())
```

Schicken Sie die Lösungen für Aufgabe 5 und 6 als Anhang einer E-Mail

`textentropy_<MatrNr>.py` (Aufgabe 5) oder `crackpwds_<MatrNr>.py` (Aufgabe 6)

an `jakob.wenzel(at)uni-weimar.de` mit dem Betreff `[Krypto SS16] Beleg 3` bis zum 23. Mai 2016, 13:30 Uhr. Es reicht dabei eine Matrikelnummer aus der Gruppe anzugeben. Die vollständigen Namen und Matrikelnummern sollen als