Aufgabenblatt 4
IT-Security
Angewandte Informatik
WS 2018/19

Lernziele – 4 Punkte

- Zufallszahlen mit Blum-Blum-Shub

Dieses Aufgabenblatt baut auf den drei vorherigen auf. Es wird das BBS-Verfahren, was sehr ähnlich zum RSA-Verfahren ist, realisiert und dazu passend die Langzahlbibliothek erweitert.

Aufgaben

Basierend auf den Routinen der letzten Aufgabenblätter wird nun das Blum-Blum-Shub-Verfahren zur Generierung von Pseudozufallszahlen realisiert. Dazu sind folgende Routinen erforderlich:

- proc initRandomBBS(BigInt s, nat size>0) - initialisiert den Generator mit dem Startwert s und der Bitlänge size von n=p*q.
- func byte randomBBS() - liefert das nächste pseudozufällige Byte

Dazu beachten Sie bitte folgende Hinweise:

- Bei der Erzeugung müssen die mathematischen Bedingungen von RSA und BBS beachtet werden.
- Von jedem erzeugten Wert wird immer das unterste Bit genommen. Nach 8 Aufrufen kann das Byte geliefert werden.
- Wenn mit Threads gearbeitet wird, können alle 8 Bits eines Bytes bei 8 CPUs parallel berechnet werden. Die Threads sollten aber durch einen Thread-Pool verwaltet werden.

Am besten ist es, wenn Sie eine oberflächliche statistische Analyse machen:

- In einem 256er-Array von Ints vermerken Sie die Häufigkeiten der Bytewerte. Diese müssten nach mind. 10.000 Generierungen etwa gleich sein.
- Dann wiederholen Sie dies mit einem 64K-Int-Array, wobei Sie nun die Häufigkeiten von 16 bit-Werten zählen. Diese sollten nach mind. 1 Mio. Generierungen auch etwa gleich sein.

Um diese Tests auszuwerten, schreiben Sie eine Routine, die den Durch-

schnittswert, den kleinsten und den größten Wert berechnet bzw. feststellt. Das ist zwar primitivste Statistik, aber besser als gar nichts.

Wenn Ihnen dies nicht gefällt, dann machen Sie zusätzlich einen Chi-Quadrat-Test (freiwillig).

Zur Realisierung ist noch eine weitere Langzahl-Routine erforderlich:

- func bool bit(BigInt x, int y>=0) – liefert das Bit an der Position y, von 0 angezählt, odd() ist daher bit(x,0)

Bitte beachten Sie folgendes Prinzip: **Es kommt auf Korrektheit und nicht auf Performanz an**.

Links

1. https://de.wikipedia.org/wiki/Blum-Blum-Shub-Generator
2. http://simul.iro.umontreal.ca/testu01/tu01.html

Abnahme

Zur Abnahme des gehören folgende Dateien:

- Testfälle mit Reports – hier die "Statistik",
- Source-Code und
- Projektdateien, z.B. netbeans-project oder make-Dateien etc.

Alle Tests vom Dozenten müssen minimal benutzt werden; es können noch weitere Test durchgeführt werden. Die Vorführung der Lösung besteht in folgenden Ablauf: (1) Source-Code-Begutachtung, (2) Übersetzung und (3) Testlauf.

Wer mit vorgefertigten Langzahl-Bibliotheken arbeitet erhält 2 Punkte Abzug.

Auch diese Aufgabe kann per Email abgegeben werden, dann mit allen Sourcen, Übersetzungsdateien, z.B. make bzw. eclipse/netbeans-Projekte, einschließlich der Tests. Alles muss ohne weiteres auf dem Rechner des Dozenten laufen können (Java, C++, Linux, CentOS 6.10). Programmieren Sie also portabel. Im Falle von Python, Ruby oder Lua bitte noch die notwendige Laufzeitumgebung mitliefern.