University of Applied Sciences and Arts Northwestern Switzerland
School of Engineering

APSI Lab 1

2. November 2013

put abstract here

Inhaltsverzeichnis

1 Aufgabenstellung

Ihre Aufgabe besteht darin, sogenannte Kollisionen im Hash-Verfahren zu suchen, d.h. nderungen im Originaltext, die den gleichen Hashwert liefern: $h(m_{orig}) = h(m_{fake})$. Wie Sie vielleicht bereits bemerkt haben, handelt es sich um eine praktische Anwendung des bekannten Geburtstagsparadoxons, das Sie in der Mathematik bzw. in der Kryptologie kennengelernt haben.

2 Softwareaufbau

Die Aufgabe wurde mit zwei Klassen implementiert. Die Klasse S̈implified Hashb̈einhaltet die beschriebene Hashfunktion und in der Klasse C̈ollisionGeneratorẅerden die verschiedenen Kombinationen der Mails generiert und nach einer Hashkollision berprft.

2.1 Hashfunktion

Die Hashfunktion wurde nach der Spezifikation der Aufgabenstellung implementiert. Jedoch hat der DES Cipher eine Output-Lnge von 128 Bit, wie man die 128 Bit verkrzen soll auf 64 Bit ist nicht spezifiziert. Deshalb wird der DES-Output in zwei 64-Bit-Blcke aufgeteilt und mit XOR auf einen 64-Bit-Block eingestampft.

3 Variationserzeugung

Fr diese Aufgabe haben wir 2^{32} verschiedene Kombinationsmglichkeiten pro Mail. Diese Kombinationen haben wir in einem Integer codiert. Jedes Bit reprsentiert einen Index eines Platzhalters. Zum Beispiel: Das zweite Bit steht auf 0, dann wird das Wort ”vom Herzenëingesetzt.

3.1 Datenstrukturen

Alle Platzhaltertexte haben wir in einer Hashmap mit P̈latzhalterïndex als Schlssel und einem Arraylist der grsse 2 fr die Texte.

4 Kollisionsdetektion

4.1 Strategie

Wir knnen zwischen zwei Strateien unterscheiden. Entweder, wir generieren alle Original und Fake Variationen linear (beginnend bei 0), oder wir benutzen einen Random-Generator fr die Original und die Fake Strings.

4.2 Datenstrukturen