Sichere Programmierung
Projekt 4

David Pierre Sugar
(76050)
Julian Sobott
(76511)

Inhaltsverzeichnis

1 Einleitung

In diesem Praktikum geht es darum Konzepte, die in der Vorlesung *Einführung in die IT-Sicherheit* erklärt wurden, in Java zu implementieren. Hierzu wird die Kryptographie-API von Java verwendet.

2 Aufgabe 1: Kryptographische Hashfunktion

In dieser Aufgabe soll die Erzeugung und Verifikation einer Prüfsumme implementiert werden. Als Hashalgorithmus soll hier `SHA-256` verwendet werden. Für die Aufgabe wurden zwei statische Methoden implementiert. `public static byte[] createChecksum(String message)` und `public static boolean verifyChecksum(byte[] checksum, String message)`. Die erste erzeugt eine Prüfsumme und die zweite verifiziert ob ein String dieselbe Prüfsumme erzeugt.

Fürs Erzeugen wird eine `MessageDigest` Instanz, welche mit `SHA-256` arbeitet, geholt. Mit dieser wird dann, der String in einen entsprechenden Hash Wert umgewandelt.

Fürs Verifizieren, wird auf dieselbe Weise von dem neuen String die Checksum erstellt. Ob die beiden gleich sind, wird mit der statischen Methode `MessageGigest.isEqual()` überprüft.

Zur demosntration der beiden Methoden, wurden in der `main` Methode 2 Strings erstellt. Die Verifikation gibt true zurück, wenn beide Strings gleich sind und false, wenn sie unterschiedlich sind.

3 Aufgabe 2: Symmetrische Verschlüsselung

4 Aufgabe 3 Asymmetrische Verschlüsselung

Diese Aufgabe soll die Nutzung des `RSA` Algorithmus für die asymmetrische Verschlüsselung zeigen. Ein Text soll mit einem public key verschlüsselt werden, und mit dem entsprechendem private key wieder entschlüssselt werden.

Hierzu wird als erstes mithilfe eines `KeyPairGenerator` ein Schlüsselpaar für die Verschlüsselung mit RSA erstellt. Auf die beiden Schlüssel kann dann mithilfe von gettern zugegriffen werden.

Als nächstes wird ein Objekt, welches für die Ver- und Entschlüsselung verantwortlich ist, mithilfe der factory Methode `getInstance` geholt. Dieses kann mehrmal verwendet werden, muss aber vor jeder Verwendung initialisiert werden. Hierbei muss übergeben werden, ob ver- oder entschlüsselt werden soll, und der jeweilige passende key. Mit der `doFinal` Methode, kann dann der ver-/entschlüsselungs Prozess abgeschlossen werden. Hierbei wird immer ein `byte[]` zurückgegebn, welches den Text enthält. Um den verschlüsselten Text schöner auf der Konsole anzuzeigen, wird er noch in Base64 umgewandelt. Außerdem wird das `byte[]` an den Konstruktor der Klasse `String` übergeben, um den Text als zusammenhängenden Text und nicht als Array auszugeben.

5 Aufgabe 4: Digitale Signatur