Aufgabe 1: Zentrale Begriffe der Kryptographie

Aufgabe 1.1: Unterschiedliche Chiffren

- Symmetrisches Kryptosystem
  - Anz. d. Schlüssel: 1
  - Verwendung für Ver- und Entschlüsselung
  - Alle involvierten Personen müssen den Schlüssel geheimhalten
- Asymmetrisches Kryptosystem
  - Anz. d. Schlüssel: 2
  - Öffentlicher Schlüssel für die Verschlüsselung
  - Privater Schlüssel für die Entschlüsselung
  - Jede Person muss nur ihren eigenen privaten Schlüssel geheimhalten

Aufgabe 1.2: Hybride Kryptosysteme

a)

Je länger die Nachricht ist, desto uneffizienter ist die Ver- und Entschlüsselung, daher lohnt sich ab einer gewissen Länge der Nachricht der Wechsel auf hybrides Kryptosystem.

b)

Alice erzeugt einen neuen symmetrischen Session-Key k und verschlüsselt mit diesem die Nachricht an Bob. Dann verschlüsselt sie k assymmetrisch mit Bobs public key und schickt die verschlüsselte Nachricht und den verschlüsselten Schlüssel an Bob.

c)

Die Nachricht setzt sich aus den symmetrisch verschlüsselten Nutzdaten (der eigentlichen Nachricht) und dem assymmetrisch verschlüsselten Session-Key zusammen

Aufgabe 2: Parkhaus

Aufgabe 2.1: Funktionsweise

Der dritte Barcode von Rechts ist immer derselbe. Die zweite Zahl des zweiten Barcodes von rechts gibt die Uhrzeit der Ticketausstellung ins Sekunden an. Die erste Zahl des rechten Barcodes gibt das Datum in Tagen ab dem 01.01.1994 an. Die Zweite Zahl ist eine fest gewählte Zahl, die sich je nach Parkdeck P0/P2 um die letze Ziffer (..0/..1) unterscheidet. Die dritte Zahl könnte ein Zufallswert oder Hash sein, der zur Identifikation des Tickets dient Diese Zahl steht umsortiert unten rechts: 1-> 5, 2-> 2, 3-> 6, 4-> 4, 5-> 1, 6-> 3

Beim Befahren des Parkhauses wird der aktuelle Zeitpunkt und eine ID per Barcode auf das Ticket gedruckt. Bei dem Bezahlvorgang wird der Barcode eingelesen und aus der Differenz des aktuellen Zeitpunkts und des Ausgabezeitpunkts der Preis errechnet. Falls sich links noch ein zusätzlicher Barcode befindet (z.B vom Kino oder Geschäft) wird dieser in die Preisberechnung mit einbezogen. Nach dem Bezahlvorgang wird ein weiter Barcode aufgedruckt, der das Ticket als bezahlt verifiziert. 10 Minuten lang wird die ID des Tickets im System abgespeichert um die Schranken bei Einwurf des Tickets mit dieser ID zu öffnen.

Aufgabe 2.2: Sicherheitsanalyse

Der Barcode des Kinos ist immer derselbe, wodurch dieser einfach von einem Angreifer auf ein bestehendes Ticket gedruckt werden kann. Auch der Barcode für die Park-Konditionen ist sehr klein und somit einfach zu analysieren und zu fälschen. Falls die letzten 6 Ziffern kein Hash sind, bzw. nicht aus dem Ausgabezeitpunkt berechnet werden, kann sich der Angreifer ein gültiges Ticket mit zuvor ausgedruckter ID (letzen 6 Ziffern) und selbst definiertem Datum und Zeitpunkt selber drucken.

Angreifermodell:

- Rolle: Benutzer
- Verbreitung: Ticketschalter, Mülleimer (weggeworfene Tickets)
- Verhalten: aktiv, er versucht aktiv eigene Tickets zu drucken, bzw. vorhandene zu verändern um kostenlos die Schranken von innen öffnen zu können
- Rechenkapazität: beschränkt (Gebrauch eines normalen Computers und eines Barcode-druckers)

Aufgabe 2.3: Umsetzung mit kryptographischen Techniken

Um Betrug effektiv zu verhindern, sollten alle Daten, die von der Ticketausgabe auf das Ticket gedruckt werden verschlüsselt sein. Dies kann mit einem symmetrischen Verschlüsselungsverfahren (z.B AES) geschehen, dessen Schlüssel nur der Ticketausgabe und dem Kassenautomaten zugänglich sind. Diese Schlüssel sollten sicher, von außen nicht zugänglich auf den Systemen abgespeichert werden. Die Park-Konditionen anderer Geschäfte sollten bevor sie auf das Ticket

gedruckt werden, asymmetrisch signiert werden (z.B RSA). In diesem Fall sollte der Kassenautomat die öffentlichen Schlüssel aller berechtigter Geschäfte speichern, um die Gültigkeit der Park-Konditionen zu verifizieren. Diese Park-Konditionen sollten vor der Verschlüsselung mit der der ID des Tickets kombiniert werden, damit die signierten Barcodes nicht einfach kopiert werden können. Die Gesamtgröße des Barcodes würde damit jedoch vervielfachen.

Aufgabe 3: Authentifizierungsprotokolle

Aufgabe 3.1: Verschlüsselte Passwort-Übermittlung

Ein passiver Angreifer kann c empfangen und damit ohne Kenntnis des Benutzernamen und des Passwortes sich gegenüber den Server authentifizieren. Außerdem ist eine Man-in-the-Middle Attacke Möglich bei der ein aktiver Angreifer sich als Server gegenüber dem Benutzer ausgibt und sich als Benutzer gegenüber dem Server.

Aufgabe 3.2: Authentifikationssystem auf Basis indeterministischer symmetrischer Verschlüsselung

Solange die Zufallszahl nicht verifiziert wird, z.B in einer zweiten Nachricht $E_k(r)$ bringt diese keine weitere sicherheit, es wird zwar jedes mal ein anderes c verwendet, der Angreifer kann jedoch jedes dieser c benutzen um sich zu verifizieren.

Aufgabe 3.3: Challenge-Response-Authentifizierung

Es wird verhindert, dass ein passiver Angreifer die Authentifikation mitschneidet und erneut benutzt. Jedoch kann sich ein aktiver Angreifer zwischen Benutzer und Dienstanbieter schalten und somit als Man-in-the-Middle die Kommunikation abfangen.

Aufgabe 3.4: Sichere Challenge-Response-Authentifizierung

Aufgabe 4: "Mensch ärgere Dich nicht"über das Telefon

Aufgabe 4.1: Protokoll

Aufgabe 4.2: Würfeln über Telefon

Aufgabe 5: RSA-Verfahren

Aufgabe 5.1: Grundlagen

- Public (n, e) (RSA-Modul, Verschlüsselungsexponent)

$n = pq$ (etwa gleichgroße unabhängige Primzahlen)

$1 < e < \varphi(n)$ und $ggT(e; \varphi(n)) = 1$

$\varphi(n) = (p-1)(q-1)$

- Private d (Entschlüsselungsexponent)

  $1 < d < \varphi(n)$ und $de \ mod \ \varphi(n) = 1$

  d Multiplikatives Inverses zu e

- Verschlüsselung:

  $c = m^e \ mod \ n$

- Entschlüsselung

  $m = c^d mod \ n$

- Sicherheit beruht auf dem Faktorisierungsproblem
- In der Praxis wird RSA am häufigsten für den Schlüsselaustausch in hybriden Verschlüsselungen (z.B TLS) und für die digitale Signatur verwendet.

Aufgabe 5.2: Anwendung

Verschlüsselungsexponent d ist 3243

Klartext: Fuer die GSS-Klausur sind folgende Themen wichtig: Schutzziele, Angreifermodelle, Rainbow Tables, die (Un-)Sicherheit von Passwoertern und dazugehoerige Angriffe, Zugangs- und Zugriffskontrolle, Biometrische Verfahren, Timing-Attack und Power-Analysis, Grundlagen der Kryptographie, Authentifikationsprotokolle, das RSA-Verfahren und natuerlich alle anderen Inhalte, die wir in der Uebung und der Vorlesung behandelt haben :-)

Aufgabe 5.2.1: decrypt.py

```
import gmpy

e = 67
p = 281
q = 389
n = p * q
phi = (p-1) * (q-1)
d = int(gmpy.invert(e,phi))
c = 103625, 71396, 5872, 102989, 10232, 36843, 71765, 5872, 10232, 14809, 108822, 108822, 6

k = ''

for i in c:
```

```
    k+= chr(pow(i,d)%n)
print(k)
```

Aufgabe 5.3: Sichere Implementierung

Bei einer Chosen-Plaintext-Attack kann der Angreifer sich einen beliebigen Klartext verschlüsseln lassen. Der Angreifer könnte so einmal den kompletten Zeichenvorrat verschlüsseln lassen um so zu jedem Zeichen den zugehörigen Schlüsseltext zu ermitteln. Auf diese Weise kann er jegliche Schlüsseltexte entschlüsseln ohne den privaten Schlüssel zu kennen. Der CBC-Mode kann diesen Angriff verhindert, indem er den Klartext in Blöcke einer bestimmten Größe aufteilt und jeder Block, bevor er verschlüsselt wird mit dem vorherigen verschlüsselten Block xor-verknüpft wird. Der erste Block wird mit einem zufälligen Initial-Wert xor-verknüpft.