GSS Blatt 6

Aufgabe 1

2)

Asymmetrisch: Für jede Person aus einer Menge von Personen gibt es jeweils 2 Schlüssel: Einen öffentlichen Schlüssel, den andere User bekommen, um Nachrichten an die Person zu schicken und einen privaten Schlüssel, mit dem die Person Nachrichten entschlüsseln kann. D.h. in einer Gruppe aus n Personen brauchen wir für jede Personen 2 Schlüssel, also 2n.

Symmetrisch: Da wir den gleichen Schlüssel für Ver- und Entschlüsselung benutzen, brauchen wir für jedes mögliche Paar in der Gruppe von n Personen einen Schlüssel. Die Anzahl der Paare aus einer Gruppe ergibt sich durch n*(n-1)/2. Also braucht man für n Personen n*(n-1)/2 Schlüssel.

3)

- Unter welchen Umständen wird Alice dazu auf ein sog. hybrides Kryptosystem zurückgreifen?

  Hybride Kryptosysteme eignen sich besonders gut für die Übertragung von großen Dokumenten, da man die Dokumente mit der Geschwindigkeit einer symmetrischen Verschlüsselung verschlüsselt, aber keinen sicheren Weg für den symmetrischen Schlüssel benötigt, da dieser Schlüssel durch die asymmetrische Verschlüsselung sicher verschickt werden kann.

- Wie geht Alice im Detail vor, wenn sie ein hybrides Kryptosystem einsetzt?

  Der Absender Alice (A) produziert ein noch unverschlüsseltes Dokument, das verschlüsselt übertragen werden soll. Angenommen, das zu verschlüsselnde Dokument sei sehr groß - in jenem Fall wird das Dokument symmetrisch verschlüsselt. Dazu wird ein symmetrischer Session-Key generiert, mit welchem das Dokument im Anschluss verschlüsselt wird. Der Session-Key wird anschließend mit Hilfe des öffentlichen Schlüssels von Bob (B) asymmetrisch verschlüsselt. Das verschlüsselte Dokument sowie der verschlüsselte Session-Key werden an Bob (B) übermittelt.

- Wie sieht die übertragene Nachricht in diesem Fall aus?

  Den Nutzdaten der Nachricht geht der symmetrische Schlüssel von Bob voraus, welchen er vorher asymmetrisch verschlüsselt hat, nachdem er mit diesem Schlüssel die tatäschliche Nachricht verschlüsselt hat.

Aufgabe 2

2)

Die Vergünstigungen werden anscheinend einfach auf den Parkschein gestempelt, ohne irgendetwas an den lesbaren Zahlen auf der unteren Hälfte des Parkscheins zu verändern. Außerdem scheinen die Barcodes immer gleich zu sein (die 32 kommt 3 mal über einem Barcode vor und in allen Fällen sind die dazugehörigen 32er-Barcodes identisch).
D.h. es sollte möglich sein, einen zusätzlichen Barcode auf einen Schein zu stempeln, ohne tatsächlich dafür zu bezahlen.
Rolle: Benutzer
Vorbereitung: Systemfehler des Parkhauses (systemweit)
Verhalten: aktiv (Druck eigener Tickets/Veränderung vorhandener Tickets)
Rechenkapazität: Keine herkömmliche Kapazität (möglicherweise Drucker). Ansonsten so lange, wie der Angreifer braucht, um das Muster in den Barcodes zu finden

3)

Da die Barcodes von Kino und vom Einzelhändler nicht einzigartig sind, müssen wir die restlichen Barcodes bei der Verschlüsselung mitbenutzen. Man könnte zum Beispiel während des Drucks des linken Barcodes gleichzeitig aus der jetzt vorliegenden, finalen Barcode-Kombination eine Zahl generieren. Diese Zahl wäre wegen der rechten Barcodes praktisch einzigartig. Nun könnte man diese Zahl symmetrisch verschlüsseln und die verschlüsselte Zahl auf das Ticket drucken. Beim verlassen des Parkhauses wird diese Verschlüsselung wieder aufgehoben und die Kombi-Zahl wird daraufhin mit dem vorliegenden Barcode des Tickets verglichen. Dadurch wäre es Betrügern nicht mehr möglich, einfach einen linken Barcode für freies Parken hinzuzufügen.

Aufgabe 3

2)

Das reverse Engineering von E_k wird erschwert, aber solange r nicht serverseitig generiert und sicher übertragen wird, ändert sich sonst an den Angriffen nichts.

3)

Durch die Challenge des Dienstanbieters erhält man einen Schutz vor Angreifern, die das Passwort auf der Leitung mithören könnten. Allerdings gibt es andere Angriffe, die immer noch möglich sind:

- Known-Plaintext-Angriff
- Wörterbuchangriff
- Chosen-Plaintext-Angriff

Aufgabe 5

2)

Python Script:

```
Import gmpy
e = 47
p = 271
q = 379
n = q * p
phi = (p-1) * (q-1)
d = int(gmpy.invert(e, phi))
c = [14979, 20999, 9653, 14027,.....]

k=" "

for i in c:
    k+= chr(pow(i,d)%n)
print(k)
```

Klartext der Nachricht:

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