# Lernbehelf zu „Grundlagen der IT-Sicherheit und Kryptographie”

Johann Wolfgang Taferl

24. April 2007

**Zusammenfassung**

Alle Rechte bleiben bei Ao. Univ.Prof. Dr. Andreas Uhl bzw. der Universität Salzburg. Alle Angaben sind ohne Gewähr und erheben keinen Anspruch auf Vollständigkeit!

## 1 Erklären Sie die Funktionsweise von RSA und die mathematische Grundlage der Entschlüsselung (im einfachen Fall $(m_i, n) = 1$). Wo geht Euler ein?

Das RSA-Verfahren beruht auf der Schwierigkeit der Faktorisierung großer Zahlen. Der Public und der Private Key sind Funktionen eines Paares großer (100 oder 200 Stellen) Primzahlen. Um den Plaintext aus dem Public Key und dem Ciphertext zu ermitteln muß im Wesentlichen das Produkt der beiden Zahlen faktorisiert werden.

Das Verfahren basiert auf folgenden Schritten

1. Um die Schlüssel zu erzeugen, wählen Sie zwei zufällig generierte Primzahlen $p$ und $q$, (möglichst gleich groß).
2. Berechnen Sie das Produkt $n = pq$.
3. Wählen Sie den Verschlüsselungs Key $e$, sodass $e$ und $\Phi(n)$[1] relativ prim sind.
4. Berechnen Sie den Entschlüsselungskey $d$, sodass $ed \equiv 1\ (mod\ \Phi(n))$, oder auch $e$ ist invers zu $d$.

Verschlüsselung: $c_i = m_i^e\ mod\ n$
Entschlüsselung: $m_i = c_i^e\ mod\ n$

**Mathematische Grundlage:**
Euklidsche Verallgemeinerung des kleinen Fermat:

$$ggt(a, n) = 1 \Rightarrow a^{\Phi(n)} \equiv 1\ mod\ n$$

$$a^{\Phi(n)}\ mod\ n = 1$$

Inverses:

$$x = y^{-1}\ mod\ n \Rightarrow xy = 1\ mod\ n$$

[1] $\Phi(n) = (p-1)(q-1)$. Ist $n$ eine Primzahl, so ist $\Phi(n) = n - 1$

wenn $ggt(x, n) = 1 \Rightarrow$

$$\Rightarrow x^{\Phi(n)-1} \; mod \; n = y$$

Aus dem Ver- und Entschlüsseln:

$$m_i = m_i^{e^d} = m_i^{ed} = m_i^{1+k\Phi(n)} = m_i \, m_i^{k\Phi(n)}$$

Durch Euler[2]:

$$m_i^{\Phi(n)} = 1 \; mod \; n \Rightarrow m_i = m_i * 1^k = m_i$$

# 2 Zum Verständnis von RSA: gegeben sind als public key n=15 und e=7. Ver- und entschlüsseln sie die Nachricht m=8 mittels RSA.

Als erstes muß der Private Key $d$ gefunden werden:

$$ed \; = \; 1 \; (mod \; \Phi(n))$$

wobei $\Phi(n) \; = \; (p-1)(q-1)$ bzw. $\Phi(n) \; = \; n-1$ wenn $n$ prim.
Außerdem ist $n \; = p * q$, wobei $p$ und $q$ prim.
$\Rightarrow \; 15 \; = \; 3 * 5$
$\Rightarrow \underline{\Phi(15) \; = (3-1)(5-1) \; = 2 * 4 \; = \; 8}$
$\Rightarrow \; d \; = \; e^{\Phi(n)-1} \; (mod \; 8)$

Nun muß nur noch d berechnet werden:
$d = 7^{8-1} \; mod \; 8 = (7^2)^3 * 7 \; mod \; 8 = 1^3 * 7 \; mod \; 8 = 7 \Rightarrow \underline{d = 7}$

Zum berechnen des Ciphertextes und der Message werden folgende Formeln benötigt:

$$c = m^e \; mod \; n$$

$$m = c^d \; mod \; n$$

Die Message $m = 8$ soll nun ver- und entschlüsselt werden:
$c = 8^7 \; mod \; 15$
$c = (8^2)^3 * 8 \; mod \; 15 = 64^3 * 8 \; mod \; 15$
$c = 4^3 * 8 \; mod \; 15 = 64 * 8 \; mod \; 15$
$c = 4 * 8 \; mod \; 15 = 32 \; mod \; 15$
$\underline{\Rightarrow \; c = 2}$

Aus $c = 2$ soll nun wieder die Message entschlüsselt werden:
$m = 2^7 \; mod \; 15$
$m = 2^4 * 2^3 \; mod \; 15 = 16 * 8 \; mod \; 15$
$m = 1 * 8 \; mod \; 15$
$\underline{\Rightarrow \; m = 8}$

[2]Hier geht „Euler" ein!

# 3 Zum Verständnis von RSA: gegeben sind als public key $n = 15$ und $e = 7$. Sie fangen einen Ciphertext $c = 8$ ab. Ermitteln Sie durch eine Attacke den dazugehörigen Plaintext.

Im Grunde ist dieses Beispiel gleich dem Bsp. 2, jedoch ist nicht die Message gegeben sondern der Ciphertext. Vorgehensweise:

1. $n$ bzw. $\Phi(n)$ berechnen.
2. $d$ aus $ed = 1 \ mod \ \Phi(n)$ berechnen.
3. $m$ aus $m = c^d \ mod \ n$ berechnen.

## 3.1 Erklären Sie die Rolle der Faktorisierung bei Ihrer Attacke.

Durch die Faktorisierung kann $p$ und $q$ ermittlet werden. Dadurch kann $\Phi(n)$ und in weiterer Folge $d = e^{-1} \ mod \ \Phi(n)$ berechnet werden.

# 4 Wie könnte eine Mischung zwischen public key und symmetrischen Verfahren funktionieren (Stichwort: Hybride Verfahren).

Ein Public Key Verfahren kann verwendet werden, um Keys eines symetrischen Verfahrens verschlüsselt zu verteilen:

1. Bob schickt Alice seinen Public Key.
2. Alice erzeugt einen zufälligen Session Key $K$, verschlüsselt diesen mit Bobs Public Key und schickt ihn zu Bob.
3. Bob entschlüsselt Alices Nachricht mit seinem Private Key und erhält den Session Key.

$$D_B(E_B(K)) = K$$

4. Beide ver- und entschlüsseln ihre Kuommunikation mit dem Session Key.

## 4.1 Warum ist so ein Verfahren sinnvoll?

- Public Key Systeme sind langsamer (etwa um den Faktor 1000)
- Sie sind durch Choosen-Plaintext Attacken verwundbar.
  $C = E(P)$, wobei $P$ ein Plaintext aus einer Menge von $n$ möglichen Plaintexten ist. Eine Kryptoanalyse muß nur alle $n$ verschlüsseln und das Resultat mit $C$ vergleichen, um $P$ zu bestimmen.
- Durch den zufälligen Session Key wird die Gefahr diesen zu verlieren reduziert.
  (Es bleibt die Verwundbarkeit des Private Keys)

# 5 Wie funktioniert die Geburtstagsattacke gegen one-way Hash funktionen.

- Alice besitzt zwei Versionen eines Vertrages, wobei eine die echte ist, und die andere zu Ungunsten von Bob verändert worden ist.
- Alice verändert beide geringfügig und berechnet die Hash-Werte der veränderten Dokumente ($2^{32}$ verschiedene Dokumente).
- Alice sucht ein Paar von Hash-Werten, die gleich sind, und rekonstruiert die Dokumente.
- Alice und Bob unterschreiben den Vertrag, der für Bob in Ordnung ist mit einem Protokoll, bei dem er den Hash des Dokumentes unterschreibt.
- Irgendwann ersetzt Alice die Originalversion mit der anderen - equivalent unterschrieben von Bob!!

## 5.1 Worauf beruht sie?

Der Hashwert ist $m$ Bits lang. Um einen vorgegebenen Hashwert zu erzeugen, müssen $2^m$ Nachrichten erzeugt werden. Damit zwei beliebige Nachrichten den selben Wert haben, muss man nur $2^{\frac{m}{2}}$ erzeugen.

## 5.2 Abhilfe?

- Hashlänge von 64 Bit auf 128 Bit erhöhen
- Hash erzeugen und an Nachricht anhängen (Kann beliebeig oft wiederhohlt werdnen)

# 6 Was ist die „meet in the middle attack”

CryptoanalystIn kennt $P_1, C_1, P_2, C_2$, sodaß

$$C_1 = E_{K_2}(E_{K_1}(P_1))$$

$$C_2 = E_{K_2}(E_{K_1}(P_2))$$

Für alle möglichen $K(K_1$ oder $K_2)$ wird $E_K(P_1)$ berechnet und im Speicher abgelegt. Danach wird $D_K(C_1)$ für alle $K$ berechnet und das identische Ergebnis im Speicher gesucht. Wird es gefunden, kann der aktuelle Schlüssel $K_2$ sein und der im Speicher $K_1$.

Wenn nun versucht wird, $P_2$ mit $K_1$ und $K_2$ zu verschlüsseln und man bekommt $C_2$, so ist ziemlich sicher, daß $K_1$ und $K_2$ gefunden wurden. Wenn nicht, wird einfach weiter gesucht. Ein Problem ist allerdings der Speicherbedarf von $2^n$ Blocks. Bei DES : $2^{56}$ 64-Bit Blöcke. Das sind etwa $10^{17}$ Bytes.

## 6.1 Was sind die Konsequenzen ihres Daseins?

TODO

## 6.2 wie kann die nicht-Gruppenstruktur von DES sonst ausgenutzt werden?

TODO

# 7 Erklären Sie die Grundprinzipien von elliptic curve cryptography.

Beide Seiten A und B des zu sichernden Kommunikationskanals einigen sich öffentlich auf eine gültige elliptische Kurve und einen Punkt P auf dieser Kurve. A beschafft sich nun geheim eine Zufallszahl $a_s$, diese Zahl ist der private Schlüssel von A. Analog beschafft sich B seinen privaten Schlüssel $b_s$.

A berechnet nun seinen öffentlichen Schlüssel $a_p = a_s P$. Analog bestimmt B seinen öffentlichen Schlüssel $b_p = b_s P$. Nach der Theorie der elliptischen Kurven liegen beide öffentliche Schlüssel auf der Kurve.

**Ver- und Entschlüsseln:**

Es gilt nun $a_s b_p = a_s b_s P = b_s a_p$. Damit ist ein Schlüssel gegeben, der nur für A und B einfach zu berechnen ist und ein öffentlich ausgetauschtes Geheimnis darstellt. Das Geheimnis ist zerstört, wenn aus den öffentlichen Punkten $P$ und $a_p$ beziehungsweise $P$ und $b_p$ die Zufallszahlen $a_s$ oder $b_s$ mit vertretbarem Aufwand berechnet werden können.

## 7.1 Was sind die Vor- und Nachteile?

- Der große Vorteil dieser Verfahren liegt darin, daß sehr viele der bisher besprochenen Kryptoalgorithmen in diesem Framework gleich sicher aber mit wesentlich kürzeren Schlüsseln realisiert werden können, d.h. die Modulo Operationen können schneller durchgeführt werden (und sind daher auch nicht so anfällig gegen sog. „timing-attacken").
- Die Parameterwahl ist aber wesentlich aufwendiger als bei klassischen Verfahren und muß auf leistungsfahigen Architekturen durchgeführt werden.

## 7.2 Für welche Anwendungsbereiche sind diese Verfahren besonders interessant?

Die kurzen Schlüssel machen solche Kryptosysteme interessante Kandidaten für mobile environments (z.B. Handyverschlüsselung).

# 8 Erklären Sie anhand eines einfachen Protokolls den Diffie-Hellman Key-exchange Algorithmus.

Benötigt werden $n$ primzahlen und $g$. Wobei $g$ eine Primitivwurzel modulo $n$ ist.

1. Alice wählt eine zufällige große Zahl $x$ und schickt Bob $X = g^x \ mod \ n$.
2. Bob wählt eine zufällige große Zahl $y$ und schickt Alice $Y = g^y \ mod \ n$.
3. Alice berechnet $K = Y^x \ mod \ n$.
4. Bob berechnet $K' = X^y \ mod \ n$.

$K = K' = g^{xy} \ mod \ n$. Niemand kann das berechnen, weil nur $n, g, X, Y$ über den Kanal geschickt wurden. Es müßte ein diskreter Logarithmus berechnet werden, um $x$ oder $y$ zu erhalten. $K$ ist also der geheime Schlüssel, den beide unabhängig berechnet haben.

Wichtig ist, dass $n$ sehr groß gewählt wird und auch $\frac{n-1}{2}$ eine Primzahl ist.

### 8.1 Wählen Sie $n = 5$ und rechnen Sie ein konkretes Beispiel durch.

Als erstes muß eine Primitivwurzel modulo $n$ gefunden werden:

$2^1 = 2 \; mod \; 5$
$2^2 = 4 \; mod \; 5$
$2^3 = 8 \; mod \; 5 = 3 \; mod \; 5$
$2^4 = 16 \; mod \; 5 = 1 \; mod \; 5$

Alle Zahle von 1 bis $(5 - 1)$ sind vorhanden $\Rightarrow$ 2 ist eine Primitivwurzel modulo 5.

Nun muß das Protokoll durchgerechnet werden:

1. Alice wählt eine zufällige große Zahl $x = 3$ und schickt Bob $X = 2^3 \; mod \; 5 = 3$.
2. Bob wählt eine zufällige große Zahl $y = 4$ und schickt Alice $Y = 2^4 \; mod \; 5 = 1$.
3. Alice berechnet $K = Y^x \; mod \; n = 1^3 \; mod \; 5 = 1 = K$.
4. Bob berechnet $K' = X^y \; mod \; n = 3^4 \; mod \; 5 = 81 \; mod \; 5 = 1 = K'$.

# 9 Was ist ein Zero-Knowledge Protokoll und was ist der Unterschied zu z.B. klassischen Authentifizierungsprotokollen?

Ein Zero-Knowledge-Protokoll stellt eine Möglichkeit zur Verfügung, eine Partei von der Gültigkeit einer Aussage zu überzeugen, ohne dabei etwas anderes als die Gültigkeit der Aussage selbst preiszugeben.

Im Gegensatz zu klassischen Authentifizierungsprotokollen muß das Geheimnis in keinster weise preisgegeben werden.

### 9.1 Beschreiben Sie detailliert die Funktionsweise des Feige-Fiat Schamit Identifikations Schemas.

Ein zufälliger Modul $n = pq, \; n \geq 512$ Bits.
$v$ muß ein quadratisches Residuum modulo $n$ sein ($x^2 = v \; mod \; n$ hat eine Lösung und $v^{-1}$ exisitiert)

Keys:

- Peggies Public Key: $v$
- Peggies Private Key: $s$ mit $s \equiv \sqrt{v^{-1}} \; mod \; n$

Protokollablauf:

1. Peggy wählt ein zufälligens $r < n$ und berechnet $x = r^2 \; mod \; n$. Dann schickt sie $x$ an Viktor.
2. Viktor schickt Peggy ein zufälliges Bit $b$.
3. $b = 0$: Peggy schickt r.
   $b = 1$: Peggy schickt $y = rs$.
4. $b = 0$: Victor überprüft, ob $x = r^2 \; mod \; n$ ist. D.h. Peggy kennt $\sqrt{x}$
   $b = 1$: Victor überprüft, ob $x = y^2 v \; mod \; n$ ist. D.h. Peggy kennt $\sqrt{v^{-1}}$

Kennt Peggy $s$ nicht, so kann sie $r$ so wählen, dass die Viktor täuscht, wenn er $b = 0$ oder $b = 1$ schickt, aber nicht beides!. D.h. die Chance ist 50%.

Peggy und Viktor wiederhohlen das Protokoll, bis Vicktor überzeugt ist, dass Peggy $s$ kennt.

#### 9.1.1 Erklären Sie die zero-knowledge Eigenschaft.

Peggy kann beweisen, dass sie $s$ kennt, ohne s preis geben zu müssen!

# 10 Passwörter finden!

## 10.1 Wie funktionert die Dictionary Attack?

Eine Liste von häufigen Passwörtern wird mit der (bekannten) One-Way Funktion verschlüsselt und mit den Einträgen in der Passwortliste verglichen. Diese Attacke ist sehr erfolgreich, da viele Menschen bei der Passwortgenerierung nicht sehr phantasievoll sind.

#### 10.1.1 Wie kann mit Salt Abhilfe geschafft werden?

Bietet eine mögliche Abhilfe. Ein zufälliger String wird einem Eintrag angehängt, ehe die Funktion angewendet wird.

## 10.2 Erkläre die Feldmay/Karn Liste?

Eine Liste mit 732.000 Passwörtern und je 4096 salt values (12 Bit salt). Etwa 30 % aller Passwörter kann mit dieser Liste geknackt werden.

## 10.3 Wie funktionert SKEY?

Dieses System besteht aus einer One-Way Funktion $f$ und einer zufälligen Zahl $R$. $f(R), f(f(R)), ..., f^{100}(R) \Rightarrow x_1, x_2, ..., x_{100}$ Alice merkt sich die Zahlen und der Computer speichert $x_{101}$. Wenn Alice sich einloggen will, gibt sie $x_{100}$ an. Der Computer berechnet $f(x_{100})$ und vergleicht das Ergebnis mit $x_{101}$. Wenn die beiden übereinstimmen, wird Alice akzeptiert. Der Computer ersetzt $x_{100}$ und $x_{101}$. Alice streicht $x_{100}$ von ihrer Liste.

# 11 Erklären Sie Betriebsarten von Blockciphern (ECM, CBC, CFB, OFB) und diskutieren Sie jeweils kurz die Vor- bzw. Nachteile sowie deren Errorpropagation.

**ECM** Electronic Codebook Mode

Ein Block Plaintext wird in einen Block Ciphertext überführt.

**Vorteile** :

- Die Reihenfolge muß bei der Ver- bzw. Entschlüsselung nicht eingehalten werden. Dies eignet sich daher besonders gut für Verwendung in Datenbanken bzw. für die Parallelisierung.
- Fehler im Code betreffen nur einen spezifischen Block. Wenn Blockgrenzen kontrolliert werden, ist auch ein Fehler von Bits kein Problem.

**Nachteile** :

- Der Vorteil der einzelnen Blöcke macht diesen Modus anfälliger für diverse Attacken. Es kann ein Codebook aller Plaintext- Ciphertext Paare erstellt werden. Große Gefahr ergibt sich daher vor allem am Anfang und am Ende von Nachrichten, da sich hier bei bestimmten Texttypen oft wiederholende Stereotype befinden, z.B. Begrüßung.
- Block Replay: Mallory könnte die verschlüsselten Nachrichten verändern, ohne daß es bemerkt wird.

**CBC** Cipher Block Chaining Mode
Ein zusätzlicher Feedback Mechansimus wird eingefügt. Die Ergebnisse der Verschlüsselung des vorherigen Blocks werden zur Verschlüsselung des aktuellen Blocks verwendet. Jeder Ciphertext Block hängt nicht nur vom entsprechenden Plaintext Block ab, sondern auch von allen vorherigen Plaintext Blöcken.

Auf den Plaintext Block wird mit dem vorhergehenden Ciphertext Block ein XOR angewendet, ehe er verschlüsselt wird.

**Nachteile** Zwei identische Texte werden gleich verschlüsselt. Wenn sie sich ab einem bestimmten Punkt unterscheiden, unterscheiden sich auch die beiden Ciphertexte an der genau gleichen Stelle.

**Errorpropagation** Ein Fehler im Plaintext betrifft zwar den gesamten Ciphertext, beim Entschlüsseln bleibt aber trotzdem nur der ursprüngliche Fehler zurück.
Ciphertext Fehler: Ein Bit zerstört den gesamten Block und ein Bit des nächsten Plaintextes an der Fehlerstelle. Der zweite Block nach dem fehlerhaften Bit ist allerdings wieder korrekt („self recovering").

**CFB** Cipher Feedback Mode
Im CBC Mode muß gewartet werden, bis der gesamte Block vorhanden ist. Bei dieser Variante genügt es, wenn $n$ Bits zur Verf ügung stehen.

Ein Block Algorithmus im CFB Mode arbeitet mit einer Queue die genau Blockgröße hat. Zu Beginn wird die Queue mit einem Initialisierungsvektor gefüllt. Die Queue wird verschlüsselt und auf die $n$-Bit am linken Rand der Queue wird mit dem Plaintext ein XOR angewendet und an das rechte Ende der Queue gestellt. Alle anderen Bits der Queue werden nach links verschoben. Dann kommt der nächste $n$-Bit Block an die Reihe.

Die Entschlüsselung ist genau umgekehrt, aber beide im Verschl üsselungsmodus.

Der Initialisierungsvektor muß bei verschiedenen Anwendungen immer verschieden sein!

**Errorpropagation** Analog zum CBC Mode betrifft ein Plaintextfehler den gesamten Ciphertext. Nach der Entschlüsselung ist der Fehler aber wieder auf die Ausgangsstelle reduziert.
Ciphertext Fehler: Ein Fehler im Ciphertext verursacht einen Fehler im Plaintext. Die Fehlerstelle wird im Shift-Register aber langsam über äußeren Rand geschoben. Nach dem „Hinauswurf" des Fehlers, erholt sich das System wieder.

**OFB** Output Feedback Mode
Das Verfahren ist dem CFB sehr ähnlich und verwendet ebenfalls Feedback. Allerdings werden hier $n$-Bits des verschlüsselten Shift Registers wieder im Shift Register verwendet. Die Entschlüsselung erfolgt wieder genau umgekehrt zur Verschlüsselung. Hier ist der Feedback u. a. von Plain- und Ciphertext, wird auch als „internal feedback" bezeichnet.

**Vorteile** Der gesamte Verschlüsselungsvorgang kann geschehen, bevor der Plaintext vorhanden ist, da nur noch ein XOR angewendet wird.

**Errorpropagation** Hier gibt es keine Error Extension. Ein Bit-Fehler im Ciphertext bewirkt einen Bit-Fehler im Plaintext.

# 12 Welcher Zahl im Restklassensystem Modulo 11 entspricht der Ausdruck $\frac{-5}{4}$ (*mod* 11)? Erklären Sie die Berechnungsschritte.

1. Als erstes hohle ich den Ausdruck in den positiven Bereich:

$$\frac{-5}{4} \ (mod\ 11) = -5 \cdot \frac{1}{4} \ (mod\ 11) = 6 \cdot 4^{-1} \ (mod\ 11)$$

2. Danach muß das Inverse von 4 (*mod* 11) berechnet werden:

$$3 \cdot 4 = 12 \ (mod\ 11) = 1 \ (mod\ 11) \Rightarrow 3 = 4^{-1} \ (mod\ 11)$$

3. Zum Schluß wird alles zusammengerechnet:

$$6 \cdot 3 \ (mod\ 11) = 18 \ (mod\ 11) = 7 \ (mod\ 11)$$

# 13 Wie können symmetrische Block-cipher verwendet werden um one-way Hash functions daraus bauen zu können?

Dieser Algorithmus ist ganz offensichtlich ein CBC oder CFB mit fixem Schlüssel und Initialisierungsvektor. Der letzte Ciphertext Block ist der Hash Value. Dies ist aber nicht genug. Besser ist es, wenn die Nachricht als Schlüssel und der vorige Hash als Input verwendet wird. Somit entspricht die Blocklänge des Ciphers der Länge des Hashs.

## 13.1 Erklären Sie das anhand der Verwendung von AES für so ein System (was ist Key, was ist Input, was ist der Hashwert, jeweilige Länge der Komponenten, ...).

TODO

## 13.2 Erklären Sie das anhand des Tandem Davics-Mayer Algorithmus.

TODO

# 14 Welches Problem wird bei SET durch sog. Duale Signaturen gelöst und wie funktioniert das?

Ein wichtiges Detail von SET ist die sogenannte „duale Signatur" Zwei Nachrichten werden verbunden die für verschiedene Empfänger bestimmt sind. In diesem Fall geht es um die Bestellungsinformation die an den Händler geht und die Bezahlungsinformation die an die Bank geht sie müssen verbunden sein um bei Streifällen die Aktion rekonstruieren zu können (diese Bezahlung ist für diese Ware und nicht für eine andere), die Bank und der Händler sollen aber andererseits die jeweils anderen Informationen nicht lesen können.

Das wird wie folgt realisiert: der Kunde berechnet einen Hash der OI und PI, fügt diese zusammen und berechnet vom Ergebnis wieder einen Hash. Das Ergebnis wird mit dem privaten Signatur-Key signiert. Hat der Händler nun diese duale Signatur DS, die OI, und den Hash der PI kann er $H(PIMD \| H(OI))$ und $D_{KU_C}(DS)$ berechnen. Sind diese Ausdrucke gleich, hat der Handler die Unterschrift verifiziert. Die Bank kann das anaolg machen mit vertauschten Rollen von PI und OI.

### 14.1 Ablauf von set?

1. Kunde erofffnet ein kreditkartenfähiges Konto bei einer SET unterstützenden Bank
2. Kunde erhält ein digitales Zertifikat (bestätigt den public key und das Ablaufdatum)
3. Händler haben auch Zertifikate (X.509v3 digitale Zertifikate) - für jede Karte die er akzeptiert für zwei Schlüssel: zum Signieren und zum Key Exchange.
4. Der Kunde tätigt eine Bestellung
5. Zusätzlich zum Bestellformular schickt der Händler eine Kopie seiner Zertifikats, daß er ein echter Händler ist.
6. Der Kunde schickt die Bestellung und Zahlungsinformation an den Händler zusammen mit seinem Zertifikat.
7. Der Händler verlangt Zahlungsautorisierung (das Payment Gateway muß bestätigen, daß die Kreditkarte o.k. ist)
8. Der Händler bestätigt die Bestellung
9. Die Ware oder Dienstleistung wird vom Händler bereitgestellt
10. Der Händler verlangt Bezahlung (vom payment Gateway).

## 15 Erklären Sie welche Bereiche der IP Pakete im Tunnel Mode und Transport Mode beim ESP Protokoll von IP Secure authentifiziert bzw. verschlüsselt werden. Was ist Tunnel und Transport Mode?

Die beiden Modi Transport und Tunnel unterscheiden sich bezüglich der Anwendungsszenarien: während der Transport Mode v.a. für sichere End-to-End Kommunikation verwendet wird, den Header unverändert lässt und die IP Payload schützt, wird beim Tunnel Mode das gesamte IP Paket geschützt und mit einem neuen IP Haeder versehen und für das Tunnelling zwischen Routern verwendet.

## 16 Was ist die „man in the middle attack”?

Zur Erinnerung eine kurze Wiederholung:

1. Alice hat Bobs Public Key vom KDC.
2. Alice generiert den Session Key, verschl üsselt ihn mit Bobs Public Key und schickt ihn an Bob.
3. Bob entschlüsselt Aices Nachricht mit seinem Private Key.
4. Beide verwenden den Schlüssel zur Kommunikation.

Dabei gibt es eine bekannte Attacke Man in the Middle Attack:

1. Alice schickt Bob ihren Public Key. Mallory fängt den Key ab und schickt Bob seinen eigenen Public Key.
2. Bob schickt Alice seinen Public key. Wie vorher fängt Mallory den Key ab und schickt Alice seinen eigenen Public Key.
3. Wenn Alice an Bob eine Nachricht schickt, fängt Mallory diese ab, entschlüsselt sie, liest sie, verschl üsselt mit Bobs Key und schickt das Resultat an Bob.
4. Wenn Bob an Alice etwas schickt, handelt Mallory analog.

### 16.1 Wie kann eine Abhilfe aussehen?

Interlock Protokoll:

1. Alice schickt Bob ihren Public Key.
2. Bob schickt Alice seinen Public Key.
3. Alice verschlüsselt ihre Nachrict mit Bobs Key und schickt die halbe Nachricht an Bob.
4. Bob verschlüsselt seine Nachricht mit Alices Key und schickt die Hälfte an Alice.
5. Alice schickt die zweite Hälfte.
6. Bob setzt die beiden Hälften zusammen und entschlüsselt sie. Dann schickt er seine zweite Hälte.
7. Alice setzt die beiden Hälften zusammen und schlüsselt Bobs Nachricht.

Dieses Verfahren verhindert die „Man in the Middle" Attacke, da die Hälfte einer Nachricht ohne die andere Hälfte nicht entschlusselt werden kann.

## 17 Wie könnte eine Mischung zwischen public key und symmetrischen Verfahren funktionieren (Stichwort hybride Verfahren)?

In den meisten praktischen Einsatzbereichen wird Public Key Kryptographie verwendet, um Keys eines symmetrischen Verfahrens verschl üsselt zu verteilen:

1. Bob schickt Alice seinen Public Key.
2. Alice erzeugt einen zufälligen Session Key $K$, verschlüsselt diesen mit Bobs Public Key und schickt in zu Bob.
$$E_B(K)$$
3. Bob entschlüsselt Alices Nachricht mit seinem Private Key und erhält den Session Key.
$$D_B E_B(K) = K$$
4. Beide ver- und entschlusseln ihre Kommunikation mit dem Session Key.

### 17.1 Warum ist so ein Verfahren sinnvoll?

Bei diesem Verfahren wird also ein Session Key erzeugt und nach der Verwendung wieder zerstört. Das verringert die Gefahr, ihn zu verlieren. Es bleibt natürlich die Verwundbarkeit des Private Key.

Ausserdem sind Public Key Systeme sind KEIN Ersatz für symmetrische Algorithmen:

- Public Key Systeme sind langsamer (etwa um den Faktor 1000).
- Sie sind durch Chosen-Plaintext Attacken verwundbar.

# 18 Atacken gegen RSA

## 18.1 „common modulus attack" Wie funktioniert sie? Wie kann Abhilfe geschaffen werden?

Einige mögliche RSA Implementierung gibt jedem das gleiche $n$ aber verschiedene $e$ und $d$. Wird die gleiche Nachricht mit zwei relativ primen Exponenten verschlüsselt, kann der Plaintext ohne Private Key entschlüsselt werden.
$c_1 = m^{e_1} \ mod \ n$
$c_2 = m^{e_2} \ mod \ n$
Der Feind kennt $n, e_1, e_2, c_1, c_2$. $(e_1, e_2) = 1 \Rightarrow \exists r$ und $s$, sodass $re_1 + se_2 = 1, r < 0$. Dann ist nur noch das folgende zu berechnen: $c_1^{-1}$ und $(c_1^{-1})^{-r} \cdot c_2^s = m^{e_1 r} \cdot m^{e_2 s} = m \ mod \ n$

## 18.2 Chosen Ciphertext Attacke

TODO

# 19 Erklären Sie die Funktionsweise (und deren mathematische Grundlage) von Digitalen Signaturen mittels El Gamal Algorithmus.

Die Sicherheit dieses Verfahrens besteht in der Schwierigkeit diskrete Logarithmen in endlichen Körpern zu berechnen. Dieses Verfahren wurde ursprünglich für digitale Unterschriften verwendet.

Zwei Primzahlen $p$ sowie $q \in N, x \in N, q, x$ beliebig und $< p$.

$$y = q^x \ mod \ p$$

$y, q, p$ sind der Public Key, wobei $q$ und $p$ von mehreren Anwednern verwendet werden können. $x$ ist der Private Key.

Wähle $K$ mit $(K, p-1) = 1$ und berechne $a = q^K \ mod \ p$.
Erw. Euklidsche Algorithmus: $M = (xa, Kb) \ mod \ (p-1)$, wobei $b$ gesucht ist.
Die Unterschrift ist das Paar $a$ und $b$. $K$ muß geheim bleiben.
Um die Unterschrift zu verifizieren, muß folgendes berechnet werden:

$$y^a a^b \ mod \ p = q^M \ mod \ p$$

wobei $y^a a^b \ mod \ p = y^{q^k} a^b = y^{q^k} a^{Kb} = q^{xq^k} a^{Kb} = q^{xa} a^{Kb}$.

Jede Unterschrift benötigt ein neues $K$ und muß zufällig gewählt werden.
Für die Verschlüsselung muß ein $K$ mit $(K, p-1) = 1$ gewählt werden.
$a = q^K \ mod \ p$
$b = y^K M \ mod \ p$
$a$ und $b$ sind der Ciphertext (mit doppelter Länge).

Für die Entschlüsselung muß

$$M = \frac{b}{a} \ mod \ p$$

berechnet werden.

# 20 Beschreiben Sie ein Verfahren und eine Anwendung von Secret Splitting für 2 und mehr Personen.

Secret Splitting bedeutet, daß eine Nachricht in mehrere Teile aufgeteilt wird. Jede einzelne Teil ist für sich alleine bedeutungslos. Alle gemeinsam ergeben erst die Nachricht.

Das Verfahren ist einfach. Trent splittet $M$ zwischen Alice und Bob:

1. Trent generiert einen zufälligen String $R$ gleich lang wie $M$
2. Trent wendet ein XOR auf $M$ und $R$ an,$\rightarrow S$:

$$M \oplus R = S$$

3. Trent schickt $R$ an Alice und $S$ an Bob.

Das ganze Verfahren ist eigentlich eine Verschl usselung mit einem One-Time Pad , wo Alice das Pad besitzt und Bob den Ciphertext.

1. Trent generiert drei Strings $R, S, T$.
2. Berechnen:

$$M \oplus R \oplus S \oplus T = U$$

3. $\cdots$

## 20.1 Was ist der Unterschied zu Secret Sharing?

$(m, n)$-threshold Verfahren: ein einfaches Beispiel solcher Verfahren: Die Nachricht wird in $m$ Stücke (shadows) geteilt, sodaß mit $n$ Teilen die Nachricht rekonstruiert werden kann.

# 21 Erklären Sie die wesentlichen Bestandteile eines Kerberos Realms und den Ablauf einer Applikationsanforderung und Durchführung.

- Zum Authentifizieren existiert ein Autentifizierungs Server, der User und Server gegenseitig authentifiziert. Der Authentifizierungsserver (AS) kennt die Passwörter aller User und speichert diese in einer zentralen Datenbank. Zusätzlich teilt der AS einen geheimen Key mit jedem Server im Netzwerk. Um von einem Client aus einen Request an einen Server schicken zu können, muß vom AS ein sogennantes Ticket geholt werden.
- Dazu existiert ein „Ticket Granting Server" (TGS), der für jeden Service ein Ticket pro Session ausstellt, wenn sich der user beim AS authentifiziert hat. Das Ticket granting Ticket wird vom Client Modul während der gesamten Session aufbewahrt und benutzt wenn immer ein neuer Service benötigt wird. Diese Tickets für einen bestimmten Service werden ebenfalls die ganze Session aufbewahrt. Hier wird nun kein Passwort mehr übertragen, denn der AS verschlüsselt das Ticket mit einem aus dem Passwort abgeleiteten Key, der User muß nur auf seinem Client zur Entschlüsselung des keys sein Paassswort eingeben.
- Zusätzlich wird vom AS an TGS und User ein session Key verteilt, mit dem der user immer seine Zugehörigkeit zu seinem Ticket beweisen kann. Auch die Authentifizierung von Servern wird über einen Key geregelt.

## 21.1 Was sind Nachteile von Kerberos?

Probleme bei Kerberos sind

- Schlechte Skalierbarkeit
- Kerberos ist eine Komplettlösung, das Einbinden in andere Applikationen ist aufwendig
- Der TGS muß immer verfügbar sein