Kryptographie – Prof. Dr. Johannes Blömer, Zusammenfassung von Florian Schoppmann

Zusammenfassung Kryptographie

Notation

Symbol	*Entsprechung*
$\mathcal{B}$	$\{0,1\}^8$

Wenn nicht anders vorausgesetzt: N sei RSA-Modulus
p, q seien Primzahlen
n sei $\in \mathbb{N}$

Zahlentheorie

Definition: p heißt starke Primzahl, falls $p = 2q+1$ gilt.

Satz: $\alpha \in \mathbb{Z}_p^*$ Generator $\iff$ Für jeden Primteiler q von $p-1$ gilt: $\alpha^{(p-1)/q} \neq 1 \bmod p$
Beweis: „$\Longrightarrow$“ klar „$\Longleftarrow$“ Annahme: α ist kein Generator. $\Longrightarrow \alpha^d = 1 \bmod p$ für ein d mit $d|p-1$. $\Longrightarrow d$ ist Vielfaches eines Primteilers q von $p-1$. $\Longrightarrow \alpha^{(p-1)/q} = (\alpha^d)^c = 1 \bmod p$, wobei $c \in \mathbb{N}$.

Satz: Sei α Generator von $\mathbb{Z}_p^*$. Dann gilt: α^i Generator $\iff \mathrm{ggT}(i, p-1) = 1 \bmod p$.
Beweis: „$\Longrightarrow$“ Annahme: $c := \mathrm{ggT}(i, p-1) \neq 1 \bmod p$. $\Longrightarrow (\alpha^i)^{(p-1)/t} = (\alpha^{p-1})^c = 1 \bmod p$ für ein $c \in \mathbb{N}$. $\Longrightarrow \alpha^i$ ist kein Generator.
„$\Longleftarrow$“ Sei $\beta \in \mathbb{Z}_p^*$ beliebig. Es gibt ein k, so dass $\beta = \alpha^k = (\alpha^i)^{k/i \bmod (p-1)} \bmod p$. Nach Voraussetzung ex. $i^{-1} \bmod p-1$.

---

1. Teil

Symmetrische Verschlüsselungsverfahren

Definition: Ein Kryptosystem ist ein 5-Tupel $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ mit:

- $\mathcal{P}$ endl. Menge von Klartexten
- $\mathcal{C}$ endl. Menge von Chiffretexten
- $\mathcal{K}$ endl. Menge von Schlüsseln
- $\mathcal{E}$ (endl.) Menge von Verschlüsselungsfunktionen
- $\mathcal{D}$ (endl.) Menge von Entschlüsselungsfunktionen

und $\forall K \in \mathcal{K} : (\exists e_K \in \mathcal{E},\ e_K : \mathcal{P} \longrightarrow \mathcal{C},\ d_K \in \mathcal{D},\ d_K : \mathcal{C} \longrightarrow \mathcal{P} : (\forall x \in \mathcal{P} : d_K(e_K(x)) = x))$

Definition:
Blockchiffre: Operiert auf Klartextblöcken fester Länge. Ggf. Padding. Typischerweise Verschlüsselungs aller Klartextblöcke mit dem gleichen Schlüssel – unabhängig ihrer Position in der Nachricht.
Stromchiffren:

Anwendungsbeispiele:
Substitutionischiffre (Blockchiffre) Ersetzt jedes Zeichen (jeden Block) des Klartextes durch ein zugeordnetes Chiffretextzeichen, Problem: Häufigkeitsanalyse
Shift-Chiffre (Substitutionschiffre, reduzierter Schlüsselraum) „Verschiebung“ jedes Buchstabens des Alphabets
Affine Chiffre (Substitutionschiffre) $\mathcal{P} = \mathcal{C} = \mathbb{Z}_p$, p Primzahl, $\mathcal{K} = (\mathbb{Z}_p \setminus \{0\}) \times \mathbb{Z}_p$. K hat die Form (a, b), $e_K(x) := ax + b \bmod p$
Hill-Chiffre $\mathcal{P} = \mathcal{C} = \mathbb{Z}_p^m, \mathcal{K} = \mathrm{GL}_m(\mathbb{Z}_p)$. $x \in \mathcal{P}$ hat die Form $(x_1, \ldots, x_m) \in \mathbb{Z}_p^m$. $e_K(x) := K \cdot x$, $d_K(y) = K^{-1} \cdot y$
Vignère-Chiffre $\mathcal{P} = \mathcal{C} = \mathcal{K} = \mathbb{Z}_p^m$, p Primzahl. $x \in \mathcal{P}$ hat die Form $(x_1, \ldots, x_m)$, $K \in \mathcal{K}$ die Form $(k_1, \ldots, k_m)$. $e_K(x) := (x_1 + k_1, \ldots, x_m + k_m) \bmod p$.
Permutations-Chiffre $\mathcal{P} = \mathcal{C} = A^m$, A beliebige endl. Menge. $\mathcal{K}$ Menge aller Permutationen der Zahlen von 1 bis m. Ein Schlüssel k ist eine solche Permutation π. $x \in \mathcal{P}$ hat die Form $(x_1, \ldots, x_m)$. $e_K(x) := (x_{\pi(1)}, \ldots, x_{\pi(m)})$.

Kerckhoff'sches Prinzip: Ein Angreifer kennt die vollständige Spezifikation des eingesetzten Systems. Unterscheide:
Chiffretext-Angriff: (CA) Angreifer kennt einen Chiffretext $e_K(x)$. Gesucht: x und/oder K
Klartext-Angriff: (PA) Angreifer kennt ein Paar $(x, e_K(x))$. Gesucht: K
Angriff mit gewählten Klartexten: (CPA) Angreifer kennt viele Paare $(x_i, e_K(x_i))$, wobei x_i frei gewählt. Gesucht: $x(\neq x_i \forall i)$ und/oder K.
Angriff mit gewählten Chiffretexten: (CCA) Angreifer kennt viele Paare $(y_i, d_K(y_i))$, wobei y_i frei gewählt. Gesucht: K.

Definition:
algorithmische Sicherheit: Ein Angreifer muss mind. N Operationen durchführen
beweisbare Sicherheit: Ein Angreifer muss ein bekannt schweres Problem lösen (z. B. Faktorisieren)

perfekte Sicherheit: Ein Angreifer kann sein Ziel bei Chiffretext-Angriffen nicht erreichen. D. h. in $d_K(x)$ sind keinerlei Informationen über x enthalten, wenn man K nicht kennt. Formal: $\Pr(P = x|C = y) = \Pr(P = x)\ \forall x \in \mathcal{P},\ y \in \mathcal{C}$ ($\Longrightarrow \Pr(C = y|P = x) = \Pr(C = y)$).

Satz: Ein Krypto-System $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ mit Zufallvariablen P und K und $|\mathcal{K}| = |\mathcal{C}| = |\mathcal{P}|$ ist genau dann perfekt sicher, wenn
1. $\forall x \in \mathcal{P}, y \in \mathcal{C} : \exists! k \in \mathcal{K} : y = e_k(x)$
2. K ist gleichverteilt: $\forall k \in \mathcal{K} : \Pr(K = k) = \frac{1}{|\mathcal{K}|}$.

Anwendungsbeispiel: One-time-pads sind perfekt sicher: $n \in \mathbb{N}$, $\mathcal{P} = \mathcal{C} = \mathcal{K} = \{0,1\}^n$, $k = (k_1, \ldots, k_n)$, $x = (x_1, \ldots, x_n)$.
$e_k(x) = (x_1 \oplus k_1, \ldots, x_n \oplus k_n)$, $d_k(x) = (y_1 \oplus k_1, \ldots, y_n \oplus k_n)$

Definition: Iterierte Chiffre: Verschlüsselungsschema bestehend aus: Schlüsselplan $K \longrightarrow (K^{(1)}, \ldots, K^{(\mathrm{Nr})})$, $K^{(i)} \in \{0,1\}^l$, wobei Nr die Rundenanzahl ist, und Rundenfunktion $g : \{0,1\}^n \times \{0,1\}^l \longrightarrow \{0,1\}^n$, wobei $\forall r \in \{0,1\}^l : g_r := g(\cdot, r)$ ist invertierbar.

Definition: Substitutions-Permutationsnetzwerk (SPN): Wird beschrieben durch: Anzahl von Runden: Nr, Schlüsselplan $K = (K^{(1)}, \ldots, K^{(\mathrm{Nr}+1)}) \in (\{0,1\}^{lm})^{(\mathrm{Nr}+1)}$, $\mathcal{P} = \mathcal{C} = (\{0,1\}^l)^m \cong \{0,1\}^{lm}$, bijektive „S-Box“ $\pi_s : \{0,1\}^l \longrightarrow \{0,1\}^l$, Permutation $\pi_p : \{1, \ldots, lm\} \longrightarrow \{1, \ldots, lm\}$.

Lemma: Falls π_s linear ist, gibt es lineare Funktionen g und f, so dass die Verschlüsselung $\mathrm{SPN}_K(x) := e_K(x) = g(x) \oplus f(K) = g(x) \oplus f_1(K^{(1)}) \oplus \ldots \oplus f_{\mathrm{Nr}+1}(K^{(\mathrm{Nr}+1)})$. g und f sind dabei bekannt für einen Angreifer und es gilt: $e_K(0) = f(K)$.

Lineare Kryptanalyse: Annahme: „Große“ Menge von Paaren (X_i, Y_i) von Klartexten, Chiffretexten gegeben.
$u^{(\mathrm{Nr}-1)} = (u_1^{(\mathrm{Nr}-1)}, \ldots, u_{lm}^{(\mathrm{Nr}-1)}) \in \{0,1\}^{lm}$
$K^{(1)}||\ldots||K^{(\mathrm{Nr})} = (k_1, \ldots, k_t) \in \{0,1\}^t$
$x = (x_1, \ldots, x_{lm}) \in \{0,1\}^{lm}$
Es gilt $\forall s \in \{0,1\}^{lm}$ bei geeigneten $I_s \subseteq \{1, \ldots, lm\}$, $J_s \subseteq \{1, \ldots, t\}$:
$u_s^{(\mathrm{Nr}-1)} = \sum_{i \in I_s} x_i \oplus \sum_{j \in J_s} k_j$
$\Longleftrightarrow u_s^{(\mathrm{Nr}-1)} \oplus \sum_{i \in I_s} x_i = \sum_{j \in J_s} k_j$
Algorithmus zur Ermittlung des letzten Rundenschlüssels $K^{(\mathrm{Nr}+1)}$:
Wähle Kandidaten L
Berechne für alle Paare (X_i, Y_i): $\pi_s^{-1}(Y_i \oplus L) =: \bar{u}^{(\mathrm{Nr}-1)}$
Falls für alle Paare und alle s gilt: $\bar{u}^{(\mathrm{Nr}-1)} \oplus \sum_{i \in I_s} x_i = \text{const.}$, dann gebe L aus.

Differentielle Kryptanalyse:

Advanced Encryption Standard (AES)
Schema:

```
State ← x
State ← AddRoundKey( State, K^1 )
for i = 1, ..., Nr − 1 do
    State ← SubByte( State )
    State ← ShiftRow( State )
    State ← MixColumn( State )
    State ← AddRoundKey( State, K^{i+1} )
end for
State ← SubByte( State )
State ← ShiftRow( State )
State ← AddRoundKey( State, K^{Nr+1} )
```

Zustandsmatrix State, wenn Repräsentation im Speicher (= Klartext vor der ersten Operation) $x_0 \ldots x_{15}$: $\begin{pmatrix} x_0 & x_4 & x_8 & x_{12} \\ x_1 & x_5 & x_9 & x_{13} \\ x_2 & x_7 & x_{10} & x_{14} \\ x_3 & x_8 & x_{11} & x_{15} \end{pmatrix}$

SubByte : $\mathbb{F}_{256} \longrightarrow \mathbb{F}_{256}$ (wird angewandt auf alle 16 Bytes eines Blocks),
$\mathrm{SubByte}(w) := A(\mathrm{Inv}(w))$, wobei
$\mathrm{Inv} : \mathbb{F}_{256} \longrightarrow \mathbb{F}_{256}$,
$\mathrm{Inv}(w) := \begin{cases} w^{-1}, & \text{falls } w \neq (00) \\ (00), & \text{sonst} \end{cases}$ und
$A : \mathbb{F}_{256} \longrightarrow \mathbb{F}_{256}$ ist definiert durch Identifikation von $\mathbb{F}_{256}$ mit $\mathbb{F}_2[x]/(m(x))$, wobei $m(x) = x^8 + x^4 + x^3 + x + 1$:
$A(w) := w \cdot t_1 + t_0 \bmod x^8 + 1$
Dabei sind dann $t_0(x) = x^4 + x^3 + x^2 + x + 1$ und $t_1(x) = x^6 + x^5 + x + 1$.

ShiftRow : ${\mathbb{F}_{256}}^{4\times4} \longrightarrow {\mathbb{F}_{256}}^{4\times4}$,
$\mathrm{ShiftRow}(\begin{pmatrix} S_{00} & S_{01} & S_{02} & S_{03} \\ S_{10} & S_{11} & S_{12} & S_{13} \\ S_{20} & S_{21} & S_{22} & S_{23} \\ S_{30} & S_{31} & S_{32} & S_{33} \end{pmatrix}) := \begin{pmatrix} S_{00} & S_{01} & S_{02} & S_{03} \\ S_{11} & S_{12} & S_{13} & S_{14} \\ S_{22} & S_{23} & S_{20} & S_{21} \\ S_{33} & S_{30} & S_{31} & S_{32} \end{pmatrix}$

MixColumn : ${\mathbb{F}_{256}}^4 \longrightarrow {\mathbb{F}_{256}}^4$ (wird angewandt auf alle 4 Spalten),
$\mathrm{MixColumn}(\begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{pmatrix}) := \begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{pmatrix}$.

Definition:

electronic code book (ECB): $y_i = e_K(x_i)$

cipher block chaining (CBC): $y_0 := 0$
$y_i = e_K(x_i + y_{i-1})$

output feedback mode (OFB): $z_0 = IV$
$z_i = e_K(z_{i-1}),\ y_i = x_i + z_i$

cipher feedback mode (CFB): $y_0 = IV$
$z_i = e_K(y_{i-1}),\ y_i = x_i + z_i$

Definition: Feistel-Chiffren sind iterierte Chiffren, in der jeder Runde eine sogenannte Feistel-Runde ist. Vor jeder Runde wird der Text in eine linke Hälfte und eine rechte eingeteilt. Dann wird auf die rechte Hälfte eine Funktion losgelassen, die Teile des Schlüssels bzw. des sogenannten Rundenschlüssels zusätzlich als Parameter mitbekommt. Das Ergebnis wird mit XOR mit der linken Texthälfte verknüpft. Das Ergebnis hiervon ist dann die rechte Texthälfte für die nächste Runde. Die alte rechte Texthälfte wird die neue linke.
Vorteil: Man ist nicht auf umkehrbare Funktionen f angewiesen.

Schema von DES (Data Encryption Standard):
DES arbeitet auf $\mathcal{P} = \mathcal{C} = \{0,1\}^{64}$. $\mathcal{K} = \{0,1\}^{56}$. Rundenanzahl: 16. (Vom Schlüssel K lin. abh. Rundenschlüssel $K^{(i)} \in \{0,1\}^{48}$ (Permutation einer Auswahl von Bits von K).

Sei S^0 der Klartext, so teilen wir stets $S^{(i-1)}$ in die Hälften $L^{(i-1)}$, R^{i-1}, wobei $L^{i-1}, R^{i-1} \in \{0,1\}^{32}$. Sind K^i die Rundenschlüssel, so berechnen wir zur Verschlüsselung:

1: $S^{i-1} = L^{i-1}R^{i-1}$
2: $L^i = R^{i-1}$
3: $R^i = f(R^{i-1}, K^i) \oplus L^{i-1}$
4: $S^i = L^iR^i$

$f : \{0,1\}^{32} \times \{0,1\}^{48} \longrightarrow \{0,1\}^{32}$
Schema für f:

1: Berechne $(B_1, B_2, \ldots, B_8) = E(R^{i-1}) \oplus K^{(i)}$, $B_i \in \{0,1\}^6$ ($E(R^{i-1})$ erweitert R^{i-1}, indem 16 der 32 Bits von R^{i-1} zweimal geschrieben werden.)
   Es bezeichne b_{ij} fortan das j-te Bit von B_i
2: **for** $i = 1, \ldots, 8$ **do**
3: Setze s_i = dem Eintrag an Zeile $(b_{i1}b_{i6})_2$ und Spalte $(b_{i2}b_{i3}b_{i4}b_{i5})_2$ aus der S-Box $S_i \in (\{0,1\}^4)^{4\times 16}$.
4: **end for**
5: Gebe $s_1 \ldots s_8 \in \{0,1\}^{8\cdot 4}$ zurück.

Asymetrische oder public-key Kryptographie

Für einen Schlüssel $K \in \mathcal{K}$ gilt: $K = (sK, pK)$, wobei sK geheim und pK öffentlich. Zusätzliche Sicherheitsfrage: Kann sK aus pK berechnet werden?

RSA: Kryptosystem $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ mit $\mathcal{P} = \mathcal{C} = \mathbb{Z}_N$, $\mathcal{K} = \{(N, p, q, e, d) | p, q \text{ prim},\ N = p \cdot q,\ e, d \in \mathbb{Z}_N^* \text{ mit } ed = 1 \bmod \Phi(N)\}$
Öffentlich: (e, N), Privat: (p, q, d) (p, q aus d berechenbar!)

Sei im Folgenden $n := \lfloor \log N \rfloor$.

Satz: $m^e \bmod N$ ist in Laufzeit $\mathcal{O}(n^3)$ mit dem Quadrieren-Multiplizieren-Algorithmus berechenbar.
Beweis: Es gilt: $m^e = \prod_{i=0}^{n} m^{e_i 2^i}$, wobei $(e_n \ldots e_0)_2 = e$.

Satz: d kann aus $e, \phi(N)$ mit Aufwand $\mathcal{O}(\log^2 n)$ berechnet werden (und umgekehrt).
Beweis: Euklidischer Algorithmus liefert $d, k \in \mathbb{Z}$ mit $1 = \text{ggT}(e, \Phi(N)) = d \cdot e + k \cdot \Phi(N) \Longrightarrow ed = 1 \bmod \Phi(N)$

Satz: (Chinesischer Rest-): Seien $m_1, \ldots, m_k \in \mathbb{N}$ paarw. teilerfremd, $a_1, \ldots, a_k$ beliebig. Dann hat

$$\left| \begin{array}{c} x = a_1 \bmod m_1 \\ \vdots \\ x = a_n \bmod m_n \end{array} \right|$$

eine eindeutige Lösung modulo $M = \prod_{i=1}^{n} m_i$.
Beweis: Für $k = 0, \ldots, M-1$ gibt es M unterschiedliche Tupel $(k \bmod m_1, \ldots, k \bmod m_n)$. (Andernfalls gäbe es $0 \le a < b < M$, so dass $b - a = c_i \cdot m_i$ für $i = 1, \ldots, n,\ c_i \in \mathbb{N} \Longrightarrow b - a = c \cdot M,\ c \in \mathbb{N}$). Andererseits gibt es genau M unterschiedliche Tupel $(k_1, \ldots, k_n)$ mit $0 \le k_i < m_i$ für $i = 1, \ldots, n$. Folglich muss für jedes $(k_1, \ldots, k_n) \in \mathbb{Z}_{m_1} \times \cdots \times \mathbb{Z}_{m_n}$ gelten: $\exists k \in \mathbb{Z}_M : (k_1, \ldots, k_n) = (k \bmod m_1, \ldots, k \bmod m_n)$.

Folgerung: Ein solches M, wie im letzten Satz beschrieben, ließe sich mit folgendem Algorithmus finden:

1: **for** $i = 1, \ldots, n$ **do**
2: Setze $M_i := \prod_{j=1,\ j\neq i}^{n} m_j$ {Es folgt: $\text{ggT}(M_i, m_i) = 1$}
3: Setze c_i so, dass $c_iM_i = 1 \bmod m_i$ (mit Euklid. Alg.) {Für $j \neq i$ gilt: $c_iM_i = 0 \bmod m_j$}
4: **end for**

5: Setze $a = \sum_{i=1}^{n} a_i c_i M_i \bmod M$. {Es gilt: $a = a_i c_i M_i = a_i \bmod m_i$}

Miller-Rabin-Test
Eingabe: $N \in \mathbb{N}$

1: Berechne r, t, so dass $N - 1 = 2^r t$, t ungerade
2: Wähle $a \in \mathbb{Z}_N \setminus \{0\}$ zufällig.
3: Setze $b := a^m \bmod N$
4: **if** $b = 1 \bmod N$ **then**
5: Ausgabe „prim“
6: **else**
7: **for** $i = 0, \ldots, r-1$ **do**
8: **if** $b = -1 \bmod N$ **then**
9: Ausgabe „prim“
10: **else**
11: Setze $b := b^2 \bmod N$
12: **end if**
13: **end for**
14: Ausgabe „zusammengesetzt“
15: **end if**

Satz: Sei N prim. Dann gibt Miller-Rabin auch „prim“ zurück.
Beweis: Es gilt: $a^{N-1} = 1 \bmod N \implies a^{2^{r-1}t} = \pm 1 \implies a^{2^{r-1}} = -1 \bmod N$ und Ausgabe „prim“ oder $a^{2^{r-1}} = 1 \bmod N \implies$ Ausgabe „prim“ oder $a^{2^{r-2}} = 1 \implies \ldots \implies$ Ausgabe „prim“ oder $a^t = 1 \bmod N \implies$ Ausgabe „prim“

Satz: Ist N zusammengesetzt, so ist mit Wkt. $\geq \frac{1}{2}$ die Ausgabe des Miller-Rabin-Tests „zusammengesetzt“.
Beweis: Setzte $m := \max\{j \in \mathbb{N} | \exists a \in \mathbb{Z}_N^* \text{ mit } a^{2^j t} = -1 \bmod N\}$. Dann ist $U := \{a \in \mathbb{Z}_N^* | a^{2^m t} = \pm 1 \bmod N\}$ eine Untergruppe von $\mathbb{Z}_N^*$. Es gilt: N zusammengesetzt und Miller-Rabin(a) = „prim“, $a \in \mathbb{Z}_N \implies a \in U$. (Denn wenn Ausgabe „prim“: $a^t = 1$ oder $a^{2^j t} = -1 \bmod N$ mit $j \in \{1, \ldots, m\} \implies a^{2^m t} = \pm 1 \bmod N$). Da U echte Untergruppe von Z_N ist, kann der Test max. für $\frac{N}{2}$ Elemente „prim“ ausgeben. (Man kann sogar zeigen, dass $|U| \leq \frac{\Phi(N)}{2}$).

Satz: (Primzahl-): Es gilt: $\lim_{x \to \infty} \frac{\pi(x) \ln(x)}{x} = 1$

Satz: d aus (e, N) effizient berechenbar $\iff N$ effizient faktorisierbar
Beweis: „$\Longleftarrow$“ klar. „$\Longrightarrow$“ Wir benötigen eine nicht-triviale Wurzel von 1 mod N. Denn: $x \neq \pm 1 \wedge x^2 = 1 \bmod N \implies N | (x-1)(x+1) \implies \mathrm{ggT}(N, x-1) = p$ oder q, da $0 < (x-1), (x+1) < N$.
Bestimmung eines solchen x: Schreibe $ed - 1 = 2^m t$. Überprüfe für ein zufälliges $a \in \mathbb{Z}_N^*$, ob $a^t, a^2 t, a^{2^2 t}, \ldots, a^{2^m t} = 1 \bmod N$. Gilt für ein b, $1 < b \leq m$, dass $a^{2^b t} = 1 \bmod N$ und $a^{2^{b-1} t} \neq \pm 1$, so ist mit $x = a^{2^{b-1} t}$ eine nicht-triviale Wurzel von 1 mod N gefunden.
Erfolgswkt. für zufälliges a ist $\geq \frac{1}{2}$.

Satz: Sei N RSA-Modulus. Kennen wir x, y mit $x \neq \pm y \bmod N$ und $x^2 = y^2 \bmod N$, dann lässt sich N effizient faktorisieren. Ferner gibt es $\Phi(N)$ solcher Paare (x, y).
Beweis: Nach Voraussetzung gilt: $N | (x-y)(x+y) \implies \mathrm{ggT}(N, x-y) = p$ oder q. Mit CRS zeigt man: $\forall y \in \mathbb{Z}_N^* : \exists x \in \mathbb{Z}_N^* : x \neq \pm y \bmod N$ und $x^2 = y^2 \bmod N$
Finden solcher (x, y): Wähle Faktorbasis $\mathcal{B} = \{p_0, \ldots, p_t\}$, d. h. die ersten t Primzahlen. Finde für $i = 1, \ldots, s$: (x_i, z_i) mit $x_i > \sqrt{N}$, $x_i^2 = z_i \bmod N$ und bestimme e_{ij}, so dass $z_i = \prod_{j=0}^{t} p_j^{e_{ij}} \bmod N$. Üblich: $s = t + 10$ und $x_i = i + \lfloor \sqrt{N} \rfloor$ (quadratisches Sieb). Bestimme $f_i \in \{0, 1\}$ so, dass $\forall j = 0, \ldots, t : \sum_{i=1}^{s} f_i e_{ij} = 0 \bmod 2$. Dann folgt: $y^2 := \prod_{i=1}^{s} z_i^{f_i} = \prod_{j=1}^{t} p_j^{\sum_{i=1}^{s} f_i e_{ij}}$ hat nur gerade Exponenten an den Primfaktoren. Folglich kann y durch Halbieren der (bekannten) Exponenten bestimmt werden. Setze $x := \prod_{i=1}^{s} x_i^{f_i}$, dann gilt: $x^2 = y^2$. Je größer s, desto wahrscheinlicher der Erfolg.

Verfahren von Rabin: Schlüssel: $K = (N, p, q)$, nur N öffentlich. $N = pq$, p, q prim mit $p = q = 3 \bmod 4$. $\mathcal{P} = \mathcal{C} = \mathbb{Z}_N^*$.
$e_K(x) = x^2 \bmod N$, nicht injektiv!

$$d_K(y) = \left\{ x \,\middle|\, \begin{array}{l} x = \pm y^{(p+1)/4} \bmod p \\ x = \pm y^{(q+1)/4} \bmod q \end{array} \right\}$$

Denn nach Euler/Fermat: $y^{(p+1)/4} = (x^2)^{(p+1)/4} = x^{(p+1)/2} = x^{1/2} = \pm 1 \bmod p$.

Satz: Rabin-Chiffretext effizient entschlüsselbar $\iff N$ effizient faktorisierbar
Beweis: „$\Longleftarrow$“ klar „$\Longrightarrow$“ Für $x, y \in \mathbb{Z}_N^*$, $y = x^2$ gibt $d_K(y)$ eine Menge mit 4 Elementen. Für 2 Elemente z davon gilt: $x \neq \pm z \bmod N$. Da $N | (x-z)(x+z)$ ergibt $\mathrm{ggT}(N, x-z) = p$ oder q.

Verfahren von ElGamal: Schlüssel: $K = (p, \alpha, a, \beta)$, p, α, β öffentlich, α Generator von $\mathbb{Z}_p^*$. $\mathcal{P} = \mathbb{Z}_p^*$, $\mathcal{C} = (\mathbb{Z}_p^*)^2$.
$e_K(x) = (a^k \bmod p, x\beta^k \bmod p)$ für ein zufälliges $k \in \mathbb{Z}_{p-1}$.
$d_K(y) = \delta\gamma^{-a} \bmod p$ für $y = (\gamma, \delta)$. Denn: $\delta\gamma^{-a} = x\alpha^{ak}a^{-ka} = x \bmod p$

Der Algorithmus von Shanks

Eingabe: $\alpha \in G$, Ordnung n von α, $\beta \in \langle a \rangle$.
Ausgabe: $\text{dlog}_\alpha \beta$.

1: Setze $m := \lceil \sqrt{n} \rceil$
2: **for** $i := 0, \ldots, m-1$ **do**
3: Füge (i, α^{im}) an Liste L_1 an
4: Füge $(i, \beta\alpha^{-i})$ an Liste L_2 an
5: **end for**
6: Finde in L_1, L_2 Elemente $(i, \alpha^{im}), (j, \beta\alpha^{-j})$ mit identischer zweiter Koordinate. Gebe $\text{dlog}_\alpha \beta = im + j$ aus.

Der Algorithmus von Pohlig-Hellman

Ergebnis: Für ElGamal sollten Gruppen benutzt werden, deren Gruppenordnung mindestens einen großen Primteiler hat.

Definition: Seien eine zyklische Gruppe G, ein Generator α sowie $\beta = \alpha^b$ und $\gamma = \alpha^c$ für unbekannte b, c gegeben. Aus diesen Informationen α^{bc} zu berechnen, ist das Diffie-Hellman-Problem.

Satz: Brechen von ElGamal kann auf Diffie-Hellman reduziert werden und umgekehrt.

---

2. Teil

Digitale Unterschriften

Definition: Ein Unterschriftenverfahren besteht aus einem 5-Tupel $(\mathcal{P}, \mathcal{A}, \mathcal{K}, \mathcal{S}, \mathcal{V})$, wobei:

- $\mathcal{P}$ endl. Menge von Klartexten
- $\mathcal{A}$ endl. Menge von Unterschriften
- $\mathcal{K}$ endl. Menge von Schlüsseln
- $\mathcal{S}$ (endl.) Menge von Signierfunktionen
- $\mathcal{V}$ (endl.) Menge von Verifikationsfunktionen, zu festem Schlüssel K haben sie Signatur $\mathcal{P} \times \mathcal{A} \longrightarrow \{0, 1\}$

Sicherheit:

Vollständiges Brechen: Geheimen Schlüssel sk aus öffentlichem Schlüssel pk bestimmen (oder sig_K aus ver_K bestimmen

Existentielle Fälschung: Angreifer kann eine beliebige, jedoch nicht selbst gewählte Nachricht unterschreiben

Selektive Fälschung: Angreifer kann eine selbst gewählte Nachricht unterschreiben

Angriffe:

Angriff mit Klartexten:

Angriff mit adaptiv gewählten Klartexten: Eve wählt sukzessive Klartexte x_i und erhält dazu passende Unterschriften y_i

Anwendungsbeispiel: RSA-Unterschriften: $N = pq$, $\mathcal{K} = \{(N, e, d) | ed = 1 \bmod \phi(N)\}$. Schlüssel: $K = (N, e, d)$, nur d geheim.
$\text{sig}_K(x) = x^d \bmod N$
$\text{ver}_K(x, y) = 1 :\Longleftrightarrow y^e = x \bmod N$
Problem: existenzielle Fälschung möglich: Wähle (x, y) mit $x = y^e \bmod N$.
Sogar selektive Fälschung möglich bei adaptiv gewählten Klartexten.

ElGamal-Unterschriften: $\mathcal{K} = \{(p, \alpha, a, \beta) | \alpha$ Generator von $\mathbb{Z}p^*, \beta = \alpha^a \bmod p\}$. Schlüssel $K = (p, \alpha, a, \beta)$, nur a geheim. $\mathcal{P} = \mathbb{Z}p^*$, $\mathcal{A} = \mathbb{Z}p^* \times \mathbb{Z}_{p-1}$
$\text{sig}_K(x) = (\gamma, \delta)$ mit $\gamma = \alpha^k \bmod p$ und $\delta := (x - a\gamma)k^{-1} \bmod p - 1$ für zufälliges $k \in \mathbb{Z}^*_{p-1}$
$\text{ver}_K(x, (\gamma, \delta)) = 1 :\Longleftrightarrow \gamma^\delta \beta^\gamma = \alpha^x \bmod p$, denn: $\gamma^\delta \beta^\gamma = \alpha^{k(x-a\gamma)k^{-1}} \alpha^{a\gamma} = \alpha^x \bmod p$
Existenzielle Fälschungen: Wähle $(x, (\gamma, \delta))$ mit $\gamma = \alpha^i \beta^j \bmod p$, $\delta = -\gamma j \bmod p - 1$ und $x = -\gamma ij \bmod p - 1$ für beliebige $i, j \in \mathbb{Z}^*_{p-1}$. Denn: $\gamma^\delta \beta^\gamma = \alpha^{-i\gamma j^{-1}} \alpha^{-aj\gamma j^{-1}} \alpha^{a\gamma} = \alpha^x \bmod p$

Hashfunktionen

Definition: Eine Familie von Hashfunktionen ist ein Tupel $(\mathcal{X}, \mathcal{Y}, \mathcal{K}, \mathcal{H})$, wobei:

- $\mathcal{X}$ Menge von Klartexten
- $\mathcal{Y}$ endl. Menge von Hashwerten
- $\mathcal{K}$ endl. Menge Schlüsseln
- $\mathcal{H}$ (endl.) Menge Hashfunktionen, zu festen $K \in \mathcal{K}$ haben sie Signatur $\mathcal{X} \longrightarrow \mathcal{Y}$

Eine Hashfunktionen h hat die Eigenschaft, dass die folgenden Probleme nicht effizient lösbar sind:

Urbildproblem: Zu $y \in \mathcal{Y}$ ein $x \in \mathcal{X}$ zu finden mit $h(x) = y$

2. Urbildproblem: Zu $x \in \mathcal{X}$ ein $x' \neq x$ zu finden mit $h(x) = h(x')$

Kollisionsproblem: $x, x', x \neq x'$ zu finden mit $h(x) = h(x')$

Idealisierte Hashfunktion: Zufallsorakel

Definition: Ein Algorithmus der entweder korrekte oder gar keine Lösung liefert, heißt Las-Vergas-Alogrithmus.

Definition: Ein Las Vegas-Algorithmus heißt ein (ϵ, k)-Algorithmus, wenn er mit Wkt. $\geq \epsilon$ eine Lösung liefert und k Hashwerte erfragt.

Satz: Erfolgswahrscheinlichkeiten von Las Vegas-Algorithmen, die jeweils k Hashwerte einer idealen Hashfunktion erfragen:
Urbildproblem: $1 - (1 - \frac{1}{M})^k$

2. Urbildproblem: $1 - \prod_{i=1}^{k-1}(1 - \frac{i}{|Y|}) \geq 1 - e^{-\frac{(k-1)^2}{2|Y|}}$

Iterierte Hashfunktionen: Benötigt compress : $\{0,1\}^{m+t} \longrightarrow \{0,1\}^m$. Beispiele hierfür: $\text{compress}(z\|y) = e(y,z) \oplus z$ oder, falls $m = t$, $\text{compress}(z\|y) = e(z,y) \oplus y$, (jeweils für Verschlüsselungsfunktion $e : \{0,1\}^t \times \{0,1\}^m \longrightarrow \{0,1\}^m$)
Eingabe: $x \in \{0,1\}^*$
Ausgabe: $z_r \in \{0,1\}^m$

1: Expandiere x zu $y \in \{0,1\}^*$ mit $|y| = 0 \bmod t$
2: Setze $y_1\|y_2\|\cdots\|y_r = y$ mit $y_1, \ldots, y_r \in \{0,1\}^t$
3: Wähle $z_0 \in \{0,1\}^m$ $\{z_0 = IV\}$
4: **for** $i = 1, \ldots, r$ **do**
5: $\quad z_i := \text{compress}(z_{i-1}\|y_i$
6: **end for**
7: Gebe z_r aus

Weiteres Verfahren: Merkle-Damgård-Konstruktion
Konkrekte Hashfunktion: SHA-1 (Secure Hash Algorithm)

Message Authentication Codes

Ein MAC ist eine Hashfamilie $(\mathcal{X}, \mathcal{Y}, \mathcal{K}, \mathcal{H})$, so dass $(x, y) \in \mathcal{X} \times \mathcal{Y}$ gültig für Schlüssel $K \in \mathcal{K}$ ist, wenn $y = h_K(x)$. Ferner ists es ohne kenntnis von K schwer, ein gültiges Paar (x, y) zu konstruieren, selbst wenn gültige Paare (x_i, y_i) bekannt sind $(x_i \neq x)$.

Konstruktionssmöglichkeiten:

- Geheimen Schlüssel als Initialvektor in iterierter Hashfunktion
- Mit iterierte Hashfunktion $x\|K$ hashen
- Aus Verschlüsselungsverfahren (z. B. Cipher-Block-Chaining MACs)

Digitale Unterschriften mit Hashing

Schnorr-Unterschriften: p, q mit $q|p-1$, $\alpha \in \mathbb{Z}_p^*$, so dass $\text{ord}(\alpha) = q$ (Intention: kleinere Untegruppe $\langle\alpha\rangle$, jedoch erhöhte Sicherheit, da bekannte Algorithmen für den diskreten Logarithmus nur auf ganz $\mathbb{Z}_p^*$ anwendbar sind), kollisionsresistente Hashfunktion $h : \{0,1\}^* \longrightarrow \mathbb{Z}_q$
$\mathcal{P} = \{0,1\}^*$, $\mathcal{A} = (\mathbb{Z}_q)^2$, $\mathcal{K} = \{(p, \alpha, a, \beta) | a \in \mathbb{Z}_q$ und $\beta = \alpha^a \bmod p\}$, nur a geheim
$\text{sig}_K(x) := (h(x\|\alpha^k), k + a\gamma \bmod q)$ für zufälliges $k \in \mathbb{Z}_q^*$
$\text{ver}_K(x, (\gamma, \delta)) = 1 :\Longleftrightarrow h(x\|\alpha^\delta\beta^{-\gamma}) = \gamma$, denn: $h(x\|\alpha^\delta\beta^{-\gamma}) = h(x\|\alpha^{k+a\gamma}\alpha^{-a\gamma}) = \gamma$

Sicherheit bei Schnorr: Selektive Fälschung: $\text{dlog}_\alpha(\cdot)$ gebraucht, Existenzielle Fälschung: Urbildproblem bei h ist zu lösen

Digital Signature Algorithm (DSA, DSS): p, q mit $q|p-1$, $phatL$ Bits, $L = 0 \bmod 64$, $512 \leq L \leq 1024$, q hat 160 Bits.
$\mathcal{P} = \{0,1\}^*$, $\mathcal{A} = \mathbb{Z}_q \times \mathbb{Z}_q^*$, $\alpha \in \mathbb{Z}_p^*$ mit $\text{ord}(\alpha) = q$, $\mathcal{K} = \{(p, q, \alpha, a, \beta) | \alpha \in \mathbb{Z}_q^*, \beta = \alpha^a \bmod p\}$, nur a geheim
$\text{sig}_K(x) = ((\alpha^k \bmod p) \bmod q, \text{SHA}-1(x) + a\gamma)k^{-1} \bmod q)$ für zufälliges $k \in \{1, \ldots, q-1\}$

$\text{ver}_K(x, (\gamma, \delta)) = 1$
$:\Longleftrightarrow (\alpha^{\text{SHA}-1(x)\delta^{-1}}\beta^{\gamma\delta^{-1}} \bmod p) = \gamma \bmod q.$

Denn: $(\alpha^{\text{SHA}-1(x)\delta^{-1}}\beta^{\gamma\delta^{-1}} \bmod q)$
$= (\alpha^{\delta^{-1}\text{SHA}-1(x)+a\gamma} \bmod p)$
$= (\alpha^k \bmod p) = \gamma \bmod q$

Satz: (Zur Wahl von p) Sei $q \in \mathbb{N}$, $r \in \mathbb{Z}_q^*$ und $\pi(q, r, x) := |\{p | p\text{prim und } p \leq x \text{ und } p = lq + r,\ l \in \mathbb{N}\}|$. Dann gilt: $\pi(q, r, x) \sim \frac{x}{\ln(x)\phi(q)}$

Satz: (Zur Wahl von α) Sei $g \in \mathbb{Z}_p^*$ zufällig. Dann gilt: $\alpha := g^{(p-1)/q} \neq 1 \bmod p \Longrightarrow \text{ord}(\alpha) = q$
Beweis: $\alpha^q = 1 \bmod p \Longrightarrow q$ ist ein Vielfaches von $\text{ord}(\alpha) \Longrightarrow \text{ord}(\alpha) = q$, denn q ist prim und $\text{ord}(\alpha) \neq 1$ nach Vor.
Bemerkungen: $(\alpha \neq 1 \bmod p \Longleftarrow g$ Generator von $\mathbb{Z}_p^*) \Longrightarrow \Pr[\alpha \neq 1 \bmod p] \geq \frac{\phi(p-1)}{p-1}$

Definition: Eine Funktion $f : D \longrightarrow W$ heißt Einwegfunktion, wenn f Urbild-resistent ist.

Das Lamport-Verfahren als Beweis der Existenz beweisbar sicherer Verfahren: $f : D \longrightarrow W$ Einwegfunktion, $\mathcal{P} = \{0,1\}^n$, $\mathcal{A} = D^n$, $\mathcal{K} = D^{2n} \times W^{2n}$
Schlüssel $K =$
$(y_{10}, y_{11}, \ldots, y_{n0}, y_{n1}, z_{10}, z_{11}, \ldots, z_{n0}, z_{n1})$, wobei $(z_{10}, \ldots, z_{n1})$ öffentlich und $z_{ij} = f(y_{ij})$

Sei $x = (x_1, \ldots, x_n)$:
$\text{sig}_K(x) = (y_{1x_1}, \ldots, y_{nx_n})$
$\text{ver}_K(x, (a_1, \ldots, a_n)) = 1 :\Longleftrightarrow f(a_i) = z_{ix_i} \forall i$

Definition: Ein Algorithmus ist ein deterministischer Fälscher, wenn er für identi-

sche Eingaben (hier: öffentliche Schlüssel) stets die gleiche Ausgabe liefert. Die Erfolgswahrscheinlichkeit eines deterministischen Fälschers ist der Anteil der Eingaben für die eine (korrekte) Fälschung ausgegeben wird.

Definition: Eine Funktion $f : D \longrightarrow W$ heißt Trapdoor-Funktion, falls es eine „kurze Beschreibung“ von f^{-1} gibt, so dass mit dieser Beschreibung f effizient invertiert werden kann.

Full Domain Hash $\mathcal{F}$ Familie von Trapdoor-Funktionen, $G : \{0,1\}^* \longrightarrow W$ Hashfunktion, die sich wie eine zufällige Funktion verhält. $\mathcal{P} = \{0,1\}^*$, $\mathcal{A} = D, \mathcal{K} = \{(f, f^{-1}) | f \in \mathcal{F}\}$
$\text{sig}_K(x) = f^{-1}(G(x))$
$\text{ver}_K(x, y) = 1 :\Longleftrightarrow f(y) = G(x)$
Folglich sind existenzelle Fälschungen nicht einmal bei Angriffen mit gewählten Klartexten möglich

Unwiderrufbare Unterschriften

Chaum van Antwerpen: $p = 2q + 1$, $\alpha \in \mathbb{Z}_p^*$ mit $\text{ord}(\alpha) = q$, $a \in \mathbb{Z}_{p-1}$, $\beta = \alpha^a \bmod p$. Schlüssel $K = (p, q, \alpha, a, \beta)$, nur a geheim, $\mathcal{P} = \mathcal{A} = \langle\alpha\rangle$
$\text{sig}_K(x) = x^a \bmod p$
Verifikationsprotokoll, Eingabe (x, y):

1: B wählt $1 \le e_1, e_2 \le q - 1$ zufällig
   B sendet $c = y^{e_1}\beta^{e_2} \bmod p$
2: A sendet $d = c^{a^{-1}} \bmod p$
3: B akzeptiert y als Unterschrift, falls $d = x^{e_1}\alpha^{e_2} \bmod p$

Denn: $d = c^{a^{-1}} = y^{e_1 a^{-1}}\alpha^{a e_2 a^{-1}} = x^{e_1}\alpha^{e_2}$ (wenn Protokoll eingehalten)

Satz: Unabhängig vom Verhalten von A in 2. hat A bei $y \neq x^a \bmod p$ höchstens eine Wahrscheinlichkeit von $\frac{1}{q-1}$, B zur Akzeptanz zu verleiten.

Widerrufprotokoll (soll akzeptieren, wenn $y \neq x^a$, d. h. A nicht unterschrieben hat)

1: B wählt $e_1, e_2 \in \mathbb{Z}_q^*$ zufällig
   B sendet $c = y^{e_1}\beta^{e_2} \bmod p$
2: A sendet $d = c^{a^{-1}} \bmod p$
3: B lehnt ab, falls $d = x^{e_1}\alpha^{e_2} \bmod p$
4: B wählt $f_1, f_2 \in \mathbb{Z}_q^*$ zufällig
   B sendet $C = y^{f_1}\beta^{f_2} \bmod p$
5: A sendet $D = C^{a^{-1}} \bmod p$
6: B lehnt ab, falls $D = x^{f_1}\alpha^{f_2} \bmod p$, akzeptiert, falls $\underbrace{(d\alpha^{-e_2})^{f_1}}_{=x^{e_1 f_1}} = \underbrace{(D\alpha^{-f_2})^{e_1}}_{=x^{f_1 e_1}} \bmod p$

Identifikationsprotokolle

Identifikation durch symmetrische Verschlüsselungsverfahren: K gemeinsamer Schlüssel, e_K entsprechende Verschlüsselungsfunktion

1: B wählt r_B zufällig und sendet $e_K(r_B)$
2: A berechnet r_B, wählt r_A zufällig und sendet $e_K(r_A \| r_B)$
3: B überprüft, ob r_B Teil des Klartextes

Gegenseitige Identifikation:

1: B wählt r_B zufällig und sendet r_B
2: A wählt r_A zufällig und sendet $e_K(r_A \| r_B)$
3: B sender $e_K(r_B \| r_A)$

Identifikation mit asymmetrischen Verfahren: A hat öffentlichen Schlüssel pk und identifiziert sich bei B

1: B wählt r zufällig und sendet $e(pk, r)$, Hashwert$h(r)$
2: A antwortet mit r' (überprüft vorher, ob Hashwert $h(r)$ korrekt)
3: B überprüft, ob $r = r'$

Fiat-Shamir-Protokoll: Trusted Authority TA, A, B

1: TA wählt p, q prim. Setzt $N = pq$
2: A wählt $s \in \mathbb{Z}_N^*$ und hinterlegt $v = s^2 \bmod n$ bei TA
3: A wählt $r \in \mathbb{Z}_N^*$ zufällig, sendet $x = r^2 \bmod N$
4: B wählt $b \in \{0, 1\}$ zufällig, sendet b
5: A sendet $y = rs^b \bmod N$
6: B akzeptiert, wenn $y^2 = xv^b \bmod N$, wobei B von der TA das v erhält

Bemerkungen: Die Möglichkeit $b = 0$ ist wichtig: Weiß A (bzw. ein Betrüger), dass $b = 1$, so könnte er $x = r^2 v^{-1}$ und $y = r$ wählen.

Satz: Fiat-Shamir ist ein Zero-Knowledge-Protkoll, d. h. B lernt aus einem Protokoll $(x_i, b_i, y_i)_{i=0}^s$ mit $y_i^2 = x_i v^{b_i} \bmod N$ nichts Neues (denn: die b_i und y_i sind zufällig unabhängig gleichverteilt, durch entsprechende Wahl der x_i könnte sich B ein Protokoll selbst erzeugen).

Schnorr-/[Okamoto-]Identifikation: TA wählt p, q mit $q | p - 1$ und $\alpha_1[, \alpha_2] \in \mathbb{Z}_p^*$ mit $\text{ord}(\alpha_1)[= \text{ord}(\alpha_2)] = q$, A wählt $a_1[, a_2] \in \mathbb{Z}_q^*$ zufällig und setzt $\beta = \alpha^{a_1}[\cdot\alpha^{a_2}] \bmod p$. Über die TA sind fortan $p, q, \alpha_1[, \alpha_2]$ erhältlich, Identität von A und öffentlicher Schlüssel (I_A, β) werden ferner von der TA signiert, so dass A selbst seinen öffentlichen Schlüssel verteilen kann.

Protokoll mit B:

1: A wählt $k_1[, k_2] \in \mathbb{Z}_q^*$ zufällig, setzt $\gamma = \alpha_1^{k_1}[\cdot\alpha_2^{k_2}] \bmod p$ und sendet $(I_A, \beta), cert_A, \gamma$
2: B überprüft die Gültigkeit von $(I_A, \beta), cert_A$. Ferner wählt B ein $r \in \mathbb{Z}_q^*$ und sendet r
3: A sendet $y_1 = k_1 + ra_1 \bmod q$ [und $y_2 = k_2 + ra_2 \bmod q$]
4: B akzeptiert, falls $\alpha_1^{y_1}[\cdot\alpha_2^{y_2}] = \gamma\beta^r \bmod p$

Denn: $\alpha_1^{y_1}[\cdot\alpha_2^{y_2}] = \alpha_1^{k_1}\alpha_1^{a_1 r}[\cdot\alpha_2^{k_2}\alpha_2^{a_2 r}] = \gamma\beta^r \bmod p$

Satz: Kennt C ein $\gamma \in \langle\alpha\rangle$ und r_1, r_2 mit $r_1 \neq r_2$, so dass C sich jeweils erfolgreich als A ausgeben könnte, dann kann C
a) bei Schnoor den geheimen Schlüssel a von A und
b) bei Okamoto aus γ, r_1, r_2 zwei Werte e_1, e_2 mit $\beta = \alpha_1^{e_1}\alpha_2^{e_2} \bmod p$ effizient berechnen].

Satz: Kenn C Werte γ, r_1, r_2 wie im vorherigen Satz, dann können C und A gemeinsam $\log_{\alpha_1}\alpha_2$ mit Wahrscheinlichkeit $1 - \frac{1}{1-q}$ effizient berechnen.

Semantische Sicherheit

Ein Kryptosystem heißt semantisch sicher, wenn man aus einem Chiffretext keinerlei Informationen über den Klartext herleiten kann.
Bemerkungen: RSA ist nicht semantisch sicher, da das Jacobi-Symbol für den Chiffretext $\left(\frac{y}{N}\right)$ aufgrund seiner Multiplikativität (und da e ungerade) identisch ist mit dem Jacobi-Symbol für den Klartext $\left(\frac{x}{N}\right)$.

Definition: Gilt $y = x^e \bmod N$ für $x, y \in \mathbb{Z}_N$, so definieren wir:
$\text{half}(y, e, N) = 0 :\Longleftrightarrow 0 \leq x < \frac{N}{2}$, sonst 1 und
$\text{parity}(y, e, N) = 0 :\Longleftrightarrow x$ ist gerade. Für festes N, e schreiben wir nur $\text{half}(y)$ und $\text{parity}(y)$.

Satz: $\text{half}(y) = \text{parity}(2^e y)$

Satz: Lässt sich $\text{half}(y, e, N)$ effizient berechnen, so lässt sich folgender Algorithmus konstruieren, der die RSA-Funktion invertiert:

1: $low := 0, high := N$
2: **for** $i := 1, \ldots, \lceil \log N \rceil$ **do**
3: $\quad mid := (low + high)/2$
4: $\quad$ **if** $\text{half}(y, e, N) = 0$ **then**
5: $\quad\quad high := mid$
6: $\quad$ **else**
7: $\quad\quad low := mid$
8: $\quad$ **end if**
9: $\quad y := 2^e y \bmod N$ $\{2^e y = (2x)^e\}$
10: **end for**
11: Gebe eindeutig bestimmtes Element aus $[low, high] \cap \mathbb{Z}$ aus

Unterscheider U: Kryptosystem $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$.

1: U wählt $x_0, x_1 \in \mathcal{P}$ und sendet diese
2: A wählt ein zufälliges $b \in \{0, 1\}$ und sendet $c = e_K(x_b)$, wobei K der Schlüssel von A
3: U antwortet mit $b' \in \{0, 1\}$

U hat Erfolg, falls $b' = b$. Dabei hat er Vorteil ϵ, falls $\Pr(b' = b) \geq \frac{1}{2} + \epsilon$.

Definition: Ein Kryptosystem heißt ϵ-semantisch sicher, falls kein effizienter Unterscheider Vorteil $\geq \epsilon$ hat.

Ein Kryptosystem heißt ϵ-semantisch sicher bei Angriffen mit gewählten Chiffretexten, falls kein Angreifer E bei folgendem Protokoll Vorteil $\geq \epsilon$ hat:

1: E wählt Chiffretexte $c_1, \ldots, c_l$ und lässt sich diese entschlüsseln
2: E wählt 2 Klartexte x_0, x_1 und sendet diese
3: A wählt ein zufälliges $b \in \{0, 1\}$ und sendet $c = e_K(x_b)$, wobei K der Schlüssel von A
4: E wählt Chiffretexte $c_{l+1}, \ldots, c_n \neq c$ und lässt sich diese entschlüsseln
5: E sendet $b' \in \{0, 1\}$

E hat Erfolg, falls $b' = b$. Dabei hat er Vorteil ϵ, falls $\Pr(b' = b) \geq \frac{1}{2} + \epsilon$.

Optimal Asymmetric Encryption Padding (OAEP):
Trapdoor-Permutation $f : \{0,1\}^k \longrightarrow \{0,1\}^k$, Generatorfunktion $G : \{0,1\}^{k_0} \longrightarrow \{0,1\}^{m+k_1}$ und Hashfunktion $H : \{0,1\}^{m+k_1} \longrightarrow \{0,1\}^{k_0}$. Es gilt: $k = m + k_0 + k_1$.
$e_{G,H}(x) = f(x0^{k_1} \oplus G(r) \| r \oplus H(x0^{k_1} \oplus G(r)))$
$d_{G,H}(y)$:

1: $y_1 \| y_2 := f^{-1}(y)$ $(y_1 \in \{0,1\}^{m+k_1},\ y_2 \in \{0,1\}^{k_0}$
2: $r := H(y_1) \oplus y_2$
3: $s := y_1 \oplus G(r)$
4: Wenn $s = x0^{k_1}$, Ausgabe x