Disclaimer

Aufgaben aus dieser Vorlage stammen aus der Vorlesung *Kryptographie* und wurden zu Übungszwecken verändert oder anders formuliert! Für die Korrektheit der Lösungen wird keine Gewähr gegeben.

1. Definitionen zu Kryptosystemen: Vervollständige die Definition eines Kryptosystems, sowie die Definition von possibilistischer und informationstheoretischer Sicherheit!

   (a) Ein Kryptosystem ist ein Tupel $S = (X, K, Y, e, d)$, wobei

   Antwort:
   - X nicht leere endliche Menge als Klartext
   - K nicht leere endliche Menge als Schlüssel
   - Y eine Menge als Chiffretexte
   - $e : X \times K \to Y$ Verschlüsselungsfunktion
   - $d : Y \times K \to X$ Entschlüsselungsfunktion

   (b) Dechiffrierbedingung

   Antwort: $\forall x \in X \forall k \in K : d(e(x, k), k) = x$

   (c) Surjektivität

   Antwort: $\forall y \in Y \exists x \in X, k \in K : y = e(x, k)$

   (d) Unter einer Chiffre von S versteht man

   Antwort: die Funktion $e(., k) : X \to Y$, $x \to e(x, k)$ für festes $k \in K$

   (e) Ein Kryptosystem heißt possibilistisch sicher, wenn gilt

   Antwort:
   - $\forall y \in Y \forall x \in X \exists k \in K : e(x, k) = y$
   - Bei possibilistischer Sicherheit und Klartexten und Schlüsseln, die Zeichenreihen über einem Alphabet sind, müssen Schlüssel mindestens so lang sein wieder zu übermittelnde Text
   - in der Verschlüsselungstabelle für e kommen in jeder Spalte alle Chiffretexte vor
   - die Einträge in jeder Zeile der Tabelle für e müssen verschieden sein

   (f) Sei (S, P_k) ein Kryptosystem mit Schlüsselverteilung. Es heißt informationstheoretisch sicher bezüglich Pr_x, wenn gilt

   Antwort:
   - wenn für alle $x \in X, y \in Y$ mit $Pr(y) > 0$ gilt: $Pr(x) = Pr(x|y)$.
   - wenn es bezüglich jeder beliebigen Klartextverteilung Pr_X informationstheoretisch sicher ist
   - (X, K, Y, e, d) ist possibilistisch sicher und $Pr_K(k) = \frac{1}{|K|}$ für alle $k \in K$
   - in der Verschlüsselungstabelle für e in jeder Spalte alle Chiffretexte vorkommen (possibilistische Sicherheit) und dass die Schlüsselverteilung Pr_K uniform ist
   - $\forall x \in X \forall y \in Y : Pr(x, y) = Pr(x)Pr(y)$ (Eintreten von x und Eintreten von y sind unabhängig)
   - $\forall x \in X$ mit $Pr(x) > 0$ und alle $y \in Y$ gilt $Pr(y) = Pr(y|x)$ (andere Formulierung der Unabhängigkeit)
   - Für alle $x, x' \in X$ mit $Pr(x), Pr(x') > 0$ und alle $y \in Y$ gilt $P^x(y) = P^{x'}(y)$.

   (g) Betrachte nun das konkrete Kryptosystem mit Schlüsselverteilung $S[Pr_K] = (X = \{a, b, c\}, K = \{k_1, k_2, k_3, k_4, k_5, k_6, k_7\}, Y = \{A, B, C, D, E\}, e, d, Pr_K)$, wobei e und Pr_K folgender Tabelle zu entnehmen sind und d die Dechiffrierbedingung erfüllt.

k	$Pr_K(k)$	$e(a, k)$	$e(b, k)$	$e(c, k)$
k_1	10%	A	E	B
k_2	15%	E	D	A
k_3	15%	D	A	C
k_4	5%	C	B	A
k_5	20%	B	C	E
k_7	10%	E	C	D

Ist S possibilistisch sicher? Gebe an, wie sich dies anschaulich in der Tabelle ausdrückt oder demonstriere dies anhand eines Gegenbeispiels.

Antwort: Ja S ist possibilistisch sicher. In jeder Spalte kommen alle Chiffretexte vor und in jeder Zeile sind die Einträge verschieden voneinander

(h) Ist $S[Pr_K]$ bezüglich der Gleichverteilung Pr_K auf den Klartexten informationstheoretisch sicher? Gebe an, wie sich dies anschaulich in der Tabelle ausdrückt oder demonstriere dies anhand eines Gegenbeispiels.

Antwort: Ja das Kryptosystem ist informationstheoretisch sicher.
Die Schlüsselverteilung ist nicht uniform. Jeder Schlüssel k hat eine andere Chiffre $x \to e(x,k)$. Die (absoluten) Wahrscheinlichkeiten für die Chiffretexte sind ebenfalls nicht uniform ($P(E)_a = \frac{1}{15} + \frac{1}{10} \approx \frac{1}{16}$, $P(B)_a = 20\%$). Die informationstechnische Sicherheit drückt sich dadurch aus, dass die Chiffretextwahrscheinlichkeiten auch für jeden Klartext (also jede Spalte) separat auftreten.

2. Sicherheit von Block Kryptosystemen

In der Vorlesung wurde folgende Situation als Szenarium 2 eingeführt: „Alice möchte Bob mehrere verschiedene Klartexte vorher bekannter und begrenzter Länge übermitteln. Sie verwendet dafür immer denselben Schlüssel. Eva hört die Chiffretexte mit und kann sich sogar einige Klartexte mit dem verwendeten Schlüssel verschlüsseln lassen."

(a) Nenne ein informationstheoretisch sicheres Block-Kryptosystem, das von Eva in Szenarium 2 leicht gebrochen werden kann.

Antwort: Aus Kenntnis von $x \in \{0,1\}^l$ und $y = e(x,k)$ für ein einziges Paar $(x,k) \in X \times K$ kann Eva den Schlüssel $k = x \oplus_l y$ berechnen. Das gilt für das Cäsar-System, das Vigenère-System und das informationstheoretisch sichere **Vernam**-System.

(b) In der Vorlesung wurde possibilistische Sicherheit für Szenarium 2 definiert. Nenne ein l-Block-Kryptosystem, das diese Definition erfüllt. Die nötige Schlüsselmenge K hat Größe...

Antwort: Ein Kryptosystem $S = (X, K, Y, e, d)$ ist possibilistisch sicher bzgl. Szenarium 2 ,wenn für jedes $1 \leq r \leq |X|$, jede Folge von paarweise verschiedenen Klartexten $x_1, x_2, ..., x_r \in X$, jeden Schlüssel $k \in K$ und jedes $y \in Y \backslash \{e(x_i,k) | 1 \leq i < r\}$ ein Schlüssel $k' \in K$ existiert mit $e(x_i,k) = e(x_i,k')$ für alle $1 \leq i < r$ und $e(x_r,k') = y$.
Die nötige Schlüsselmenge K hat Größe $|\{\pi | \pi : X \to Y \text{ ist injektiv}\}| = \frac{|Y|!}{(|Y|-|X|)!} \geq |X|!$ viele Schlüssel. Mit $X = \{0,1\}^{128}$ gibt es also $\geq 2^{128}!$ viele Schlüssel.

(c) Nenne ein Block-Kryptosystem aus der Vorlesung, das gegenwärtig für Szenarium 2 in der Praxis benutzt wird.

Antwort: Triple-DES, AES

(d) Beschreibe das Konzept eines l-Unterscheiders und das zugehörige Sicherheitsspiel. Definiere den Vorteil eines Unterscheiders.

Antwort:
Unterscheider U: Ein l-Unterscheider ist ein randomisierter Algorithmus $U(F : \{0,1\}^l \to \{0,1\}^l) : \{0,1\}$, dessen Laufzeit bzw. Ressourcenaufwand durch eine Konstante beschränkt ist. Das Argument des l-Unterscheiders ist eine Chiffre F. Diese ist als „Orakel" gegeben, das heißt als Prozedur, die nur ausgewertet werden kann, deren Programmtext U aber nicht kennt. Das Programm U kann F endlich oft aufrufen, um sich Paare zu besorgen. Danach kann U noch weiter rechnen, um zu einer Entscheidung zu kommen. Das von U gelieferte Ergebnis ist ein Bit. Für ein gegebenes Block-Kryptosystem B ist das gewünschte Verfahren: Programm U sollte 1 liefern, wenn F eine Chiffre $e(.,k)$ zu B ist, und 0, wenn $F = \pi$ für eine Permutation $\pi \in P\{0,1\}^l$ ist, die keine B-Chiffre ist.
Spiel G_U^B: Wir definieren ein Spiel, mit dem ein beliebiges Block-Kryptosystem B und ein beliebiger Unterscheider U darauf getestet werden, ob B gegenüber U „anfällig" ist oder nicht. Die Idee ist folgende: Man entscheidet mit einem Münzwurf (Zufallsbit b), ob U für seine Untersuchungen als $F(.)$ eine zufällige Chiffre $e(.,k)$ von B („Realwelt") oder eine zufällige Permutation π von $\{0,1\}^l$ („Idealwelt") erhalten soll. Dann rechnet U mit F als Orakel und gibt dann seine Meinung ab, ob er sich in der Realwelt oder in der Idealwelt befindet. U „gewinnt", wenn diese Meinung zutrifft.
Vorteil:

- der Vorteil von U bzgl. B ist $adv(U,B) := 2(Pr(G_U^B = 1) - \frac{1}{2})$
- Für jeden l-Unterscheider U und jedes l-Block-KS B gilt $-1 \geq adv(U,B) \geq 1$
- Werte $adv(U,B) < 0$ sind uninteressant (Ausgaben können vertauscht werden um positiven Vorteil zu erhalten)

3. Betriebsmodi von Blockchiffren. Gegeben ist das 4 -Block-Kryptosystem $B = (\{0,1\}^4, \{0,1\}^4, \{0,1\}^4, e, d)$, wobei e der Tabelle 1 entnommen werden kann.

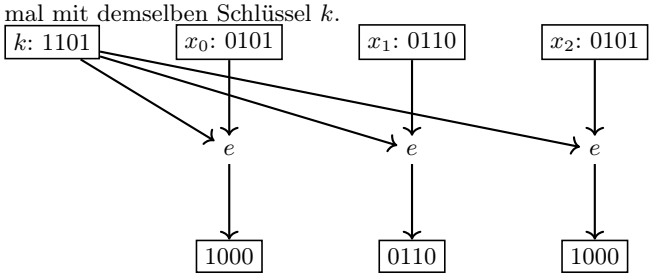
(a) Zeichne die Schaltbilder, sodass Sie die Verschlüsselung des Klartextes $x = 0101\ 0110\ 0101$ mit dem Schlüssel $k = 1101$ in dem Kryptoschema darstellen, das zu B in der jeweiligen Betriebsart gehört.

Tabelle 1: Verschlüsselungsfunktion e

e	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0000	1110	0100	0001	0101	0111	1001	0110	1000	0010	1111	1011	0000	1100	1010	0011	1101
0001	0010	0110	1100	1101	0001	1011	1111	1000	0100	0000	0101	0111	1110	0011	1001	1010
0010	1000	1001	0010	0111	0011	1101	0101	1111	1110	0001	1011	0100	1010	0000	1100	0110
0011	0110	1010	0011	1101	0010	1000	0001	0101	1110	1100	1111	1001	0100	0000	1011	0111
0100	0100	0010	1001	1000	0111	0011	1100	0110	1011	1110	1111	0101	1010	0001	0000	1101
0101	1001	0101	1010	0100	0010	1011	1000	1100	0111	1110	0001	0000	1101	0011	1111	0110
0110	1101	0001	1100	0010	0000	1000	0011	0111	0110	1111	1110	1001	1010	0101	0100	1011
0111	1100	1101	0010	1111	0110	1001	0111	0001	1000	1110	0011	0000	0101	1011	1010	0100
1000	1001	1011	1101	0000	0101	0111	1100	1111	0001	1110	0110	0011	1010	0010	0100	1000
1001	1011	0001	0011	1000	1100	0010	1111	0000	0100	1010	0110	1110	0101	0111	1101	1001
1010	1011	0010	0101	1000	1001	0011	0001	1110	0000	1100	1010	0111	1101	1111	0100	0110
1011	1110	1100	0111	1101	1011	1111	0101	0110	1000	1010	1001	0011	0100	0010	0000	0001
1100	1001	0000	0010	1101	0100	0001	1111	1000	1011	1100	1110	1010	0101	0011	0110	0111
1101	1001	0100	1101	1010	0001	1000	0110	0010	1110	1111	1011	1100	0111	0011	0000	0101
1110	1011	0111	0101	1101	1010	0001	0100	1000	1001	1110	1111	1100	0011	0010	0110	0000
1111	1010	1101	1110	1001	0001	0100	0010	0110	1100	1000	0000	0101	1111	1011	0011	0111

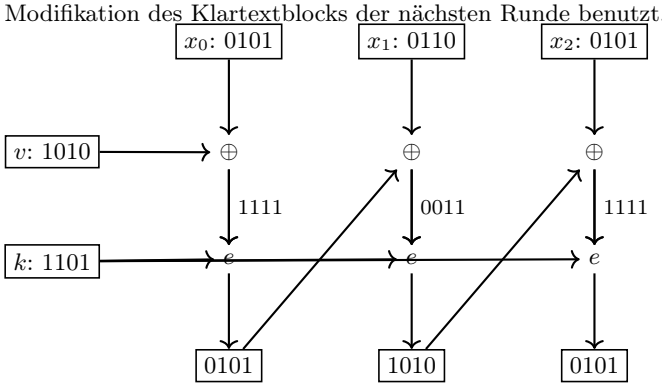
i. Benutze die ECB-Betriebsart (Electronic Code Book)!

Antwort: Ein Schlüssel ist ein Schlüssel k von B. Man verschlüsselt einfach die einzelnen Blöcke von x mit B, jedes mal mit demselben Schlüssel k.



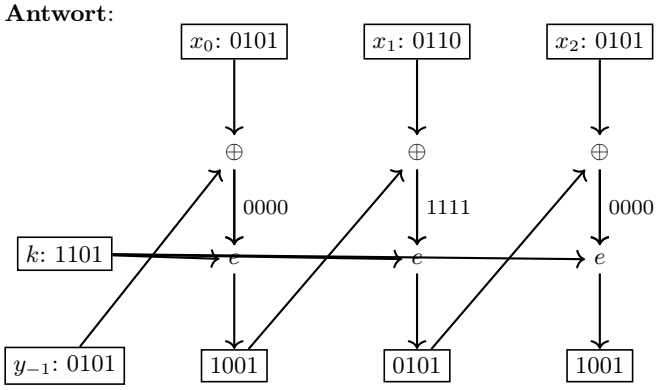
ii. Benutze die CBC-Betriebsart (Cipher Block Chaining)! Gehe davon aus, dass $v = 1010$ als Initialisierungsvektor Teil des Schlüssels ist.

Antwort: Blöcke in Runden $i = 0, 1, ..., m-1$ nacheinander verschlüsselt und das Ergebnis einer Runde wird zur Modifikation des Klartextblocks der nächsten Runde benutzt.



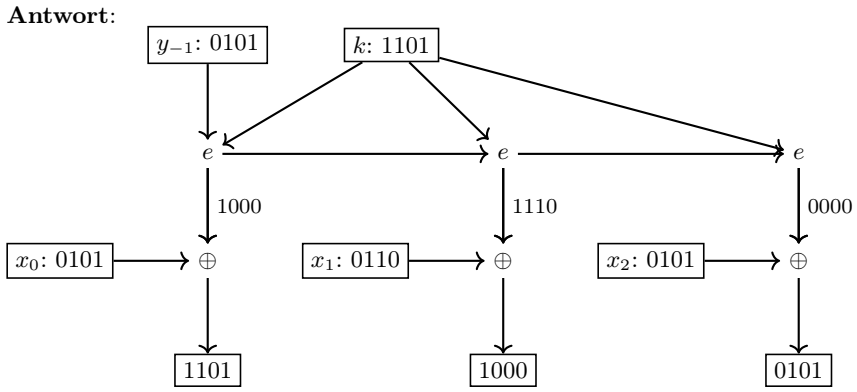
iii. Benutze die R-CBC-Betriebsart (Randomized Cipher Block Chaining)! Gehe davon aus, dass $y_{-1} = 0101$ als Initialisierungsvektor zufällig gewählt wurde.

Antwort:



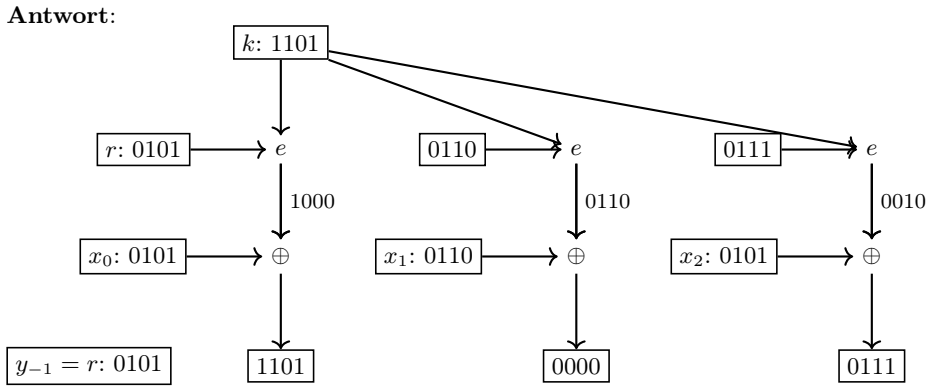
iv. Benutze die OFB-Betriebsart (Output FeedBack)! Gehe davon aus, dass $y_{-1} = 0101$ als Initialisierungsvektor zufällig gewählt wurde.

Antwort:



v. Benutze die R-CTR-Betriebsart (Randomized CounTeR)! Gehe davon aus, dass der Zähler zufällig mit dem Wert $r = 0101$ initialisiert wurde.

Antwort:



(b) Sei S das Kryptoschema, das aus B in der **ECB-Betriebsart** entsteht. Gebe einen Angreifer an, der die Chiffretexte zweier selbstgewählter Klartexte ohne Kenntnis des Schlüssels unterscheiden kann. Eine informelle Beschreibung der Finder- und Raterkomponente des Angreifers ist ausreichend.

Antwort: Ein Block $x \in \{0,1\}^l$ wird immer gleich verschlüsselt. Eva kann also ganz leicht nicht-triviale Informationen aus dem Chiffretext erhalten. Zum Beispiel kann sie sofort sehen, ob der Klartext die Form $x = x_1 x_1$, mit $x_1 \in \{0,1\}^l$, hat oder nicht.

(c) Sei S das Kryptoschema, das aus B in der **CBC-Betriebsart** entsteht. Gebe einen Angreifer an, der die Chiffretexte zweier selbstgewählter Klartexte ohne Kenntnis des Schlüssels unterscheiden kann. Eine informelle Beschreibung der Finder- und Raterkomponente des Angreifers ist ausreichend.

Antwort: Wird zweimal der Klartext x verschlüsselt, so geschieht dies immer durch denselben Chiffretext $y = E(x, (k, v))$. Dies ist eine Folge der Eigenschaft von CBC, deterministisch zu sein.

4. Zahlentheoretische Algorithmen

(a) Auf Eingabe $x, y \in \mathbb{N}$ liefert der Euklidische Algorithmus eine ganze Zahlen d mit ...

Antwort:

1. $a, b : integer; a \leftarrow |x|; b \leftarrow |y|;$
2. *while* $b > 0$ *repeat*
   (a) $(a, b) \leftarrow (b, a \ mod \ b)$; // simultane Zuweisung
3. return a

Der Euklidische Algorithmus liefert eine ganze Zahl d, die der größte gemeinsame Teiler von x und y ist.

(b) Auf Eingabe $x, y \in \mathbb{N}$ liefert der erweiterte Euklidische Algorithmus (EEA) drei ganze Zahlen d, s, t. Welche Eigenschaften erfüllen diese?

Antwort:

1. Für die Ausgabe (d, s, t) gilt $d = ggT(x, y) = s * x + t * y$.

2. Die Anzahl der Schleifendurchläufe ist dieselbe wie beim gewöhnlichen Euklidischen Algorithmus
3. Die Anzahl von Ziffernoperationen ist $O((log\ x)(log\ y))$

Algorithmus:

1. $a, b, sa, ta, sb, tb, q : integer$;
2. $a \leftarrow x; b \leftarrow y$;
3. $sa \leftarrow 1; ta \leftarrow 0; sb \leftarrow 0; tb \leftarrow 1$;
4. while $b > 0$ repeat
   (a) $q \leftarrow a\ div\ b$;
   (b) $(a, b) \leftarrow (b, a - q * b)$;
   (c) $(sa, ta, sb, tb) \leftarrow (sb, tb, sa - q * sb, ta - q * tb)$;
5. return(a, sa, ta)

(c) Für $x = 15$ und $y = 9$ liefert der EEA die Zahlen d=... ,s=... ,t=...

Antwort:
$ggT(x, y) = d = x * s + y * t$
$\downarrow$: $y_i \rightarrow x_{i+1}$, $r_i \rightarrow y_{i+1}$
$\uparrow$: $s_i = t_{i+1}$, $t_i = s_{i+1} - q_i * t_{i+1}$

i	x	y	q (Teiler)	r(est)	s	t	NR $\downarrow$	NR $\uparrow$
1	15	9	1	6	-1	$1 - 1 * (-1) = 2$	$15 - 9 * 1 = 6$	$15 * -1 + 9 * 2 = 3$
2	9	6	1	3	1	$0 - 1 * 1 = -1$	$9 - 6 * 1 = 3$	$9 * 1 + 6 * (-1) = 3$
3	6	3	2	0	0	1	$6 - 3 * 2 = 0$	$6 * 0 + 3 * 1 = 3$

$\Rightarrow d = 3, s = -1, t = 2$

(d) Wenn er auf zwei Zahlen mit je n Bits angewendet wird, führt der erweiterte Euklidische Algorithmus $O(...)$ Bitoperationen aus.

Antwort: $O((log\ x)(log\ y))$

(e) Seien a und N teilerfremde natürliche Zahlen. Wie kann man eine ganze Zahl b ermitteln, die die Gleichung $a * b\ mod\ N = 1$ erfüllt?

Antwort:
$(a * b)\ mod\ N = (a\ mod\ N * b\ mod\ N) mod\ N = 1$

(f) Ergänze den Algorithmenrumpf der Funktion $modexp(x, y, N)$ zur rekursiven Berechnung von $x^y\ mod\ n$ mithilfe der schnellen modularen Exponentiation: Funktion modexp(x,y,N) if y=0 then ... if y=1 then ... z← ... // rekursiver Aufruf if ... then z← ... return z
Dieser Algorithmus führt $O(...)$ modulare Multiplikationen aus.

Antwort:
function $modexp(x, y, m)$

- if $y = 0$ then return 1
- if $y = 1$ then return x
- $z \leftarrow modexp((x * x) mod\ m, \lfloor y/2 \rfloor, m)$; // rekursiver Aufruf
- if y ist ungerade then $z \leftarrow (z * x) mod\ m$
- return z

In jeder Rekursionsstufe ist eine oder sind zwei Multiplikationen modulo m auszuführen, was $O((log\ m)^2)$ Ziffernoperationen erfordert.

(g) Für $m \geq 2$ ist die Menge $\mathbb{Z}_m^*$ definiert durch $\mathbb{Z}_m^* :=$ .... $\mathbb{Z}_m^*$ mit der ... als Operation ist eine ... Gruppe. $\varphi(m) :=$ ....
Drücke $\varphi(m)$ als Funktion von m und seinen Primfaktoren aus: $\varphi(m) =$ ... $* \prod_{...}$ ....
Gebe die folgenden Werte an: $\varphi(2) =$ ..., $\varphi(3) =$ ..., $\varphi(4) =$ ..., $\varphi(5) =$ ..., $\varphi(8) =$ ..., $\varphi(10) =$ ..., $\varphi(12) =$ ..., $\varphi(55) =$ ..., $\varphi(64) =$ ....

Antwort:

(h) Vervollständige den Chinesischen Restsatz: Wenn m und n ... Zahlen sind, dann ist die Abbildung $\Phi :$ ... $\rightarrow$ ..., $x \rightarrow$ ..., ...

Antwort: Der „Chinesische Restsatz" besagt im Wesentlichen, dass für teilerfremde Zahlen m und n die Strukturen $\mathbb{Z}_m \times \mathbb{Z}_n$ (mit komponentenweisen Operationen) und $\mathbb{Z}_{mn}$ isomorph sind.
m und n seien teilerfremd. Dann ist die Abbildung $\Phi : \mathbb{Z}_{mn} \ni x \to (x \ mod \ m, x \ mod \ n) \in \mathbb{Z}_m \times \mathbb{Z}_n$ bijektiv. Weiterhin: Wenn $\Phi(x) = (x_1, x_2)$ und $\Phi(y) = (y_1, y_2)$, dann gilt:

1. $\Phi(x +_{mn} y) = (x_1 +_m y_1, x_2 +_n y_2)$
2. $\Phi(x *_{mn} y) = (x_1 *_m y_1, x_2 *_n y_2)$
3. $\Phi(1) = (1, 1)$

(Dabei bezeichnen $+_j$ und $*_j$ die Addition und die Multiplikation modulo j.)

(i) Vervollständige den kleinen Satz von Fermat: Wenn p ... ist und a in ... liegt, dann gilt: ...

Antwort: Wenn p eine Primzahl ist und $a \in \mathbb{Z}_p^*$ liegt, dann gilt $a^{p-1} \ mod \ p = 1$

(j) Vervollständige den Satz von Euler: Für $m \geq 2$ und x mit ... gilt ... .

Antwort: Für $m \geq 2$ und x mit $ggT(m, x) = 1$ gilt $x\varphi(m) \ mod \ m = 1$

5. Primzahltests und Primzahlerzeugung

(a) Definiere den Begriff „a ist ein F-Lügner" (für N): N ist ... und es gilt ... .

Antwort: Sei $N \geq 3$ ungerade und zusammengesetzt.
Eine Zahl $a \in \{1, ..., N-1\}$ heißt **F-Zeuge** für N, wenn $a^{N-1} mod \ N \neq 1$ gilt.
Eine Zahl $a \in \{1, ..., N-1\}$ heißt **F-Lügner** für N, wenn $a^{N-1} mod \ N = 1$ gilt.
Die Menge der F-Lügner nennen wir L_N^F.

(b) Definiere: N heißt Carmichael-Zahl, wenn ...

Antwort: Eine ungerade zusammengesetzte Zahl N heißt eine Carmichael-Zahl, wenn für alle $a \in \mathbb{Z}_N^*$ die Gleichung $a^{N-1} \ mod \ N = 1$ gilt.

(c) Formuliere den Fermat-Test für eine gegebene ungerade Zahl $N \geq 5$: Wähle... und berechne $c = ...$ Wenn $c = ...$ ist, ist die Ausgabe ...., sonst ist sie ... .

Antwort: Nutze den Fermat-Test, um „Zeugen" dafür anzugeben, dass eine Zahl N zusammengesetzt ist: Wenn wir eine Zahl a mit $1 \leq a < N$ finden, für die $a^{N-1} mod \ N \neq 1$ gilt, dann ist N definitiv keine Primzahl.
Für eine gegebene ungerade Zahl $N \geq 5$: Wähle $a < 5$ und berechne $c = a^{N-1} \ mod \ N$. Wenn $c \neq 1$ ist, ist die Ausgabe N ist kine Primzahl, sonst ist sie eine Primzahl.

(d) Definiere: $b \in \{1, ..., N-1\}$ heißt nichttriviale Quadratwurzel der 1 modulo N, wenn...

Antwort: Eine Zahl $b \in \{2, ..., N-2\}$ mit $b^2 \ mod \ N = 1$ heißt eine nicht triviale Quadratwurzel der 1 modulo N. Bei Primzahlen gibt es solche Zahlen nicht.

(e) Wenn man eine nichttriviale Quadratwurzel b der 1 modulo N gefunden hat, weiß man sicher, dass N.... ist.

Antwort: Wenn es eine nichttriviale Quadratwurzel der 1 modulo N gibt, dann ist N zusammengesetzt.

(f) Definiere den Begriff { qqa ist ein MR-Lügner} (für N): Suche ungerades u und $k \geq 1$ mit...=... . Bilde die Folge $b_0 = ..., b_1 = ..., ..., b_k = ...$ a heißt dann ein MR-Lügner (für N), falls ...

Antwort: Sei $N \geq 3$ ungerade und zusammengesetzt. Wir schreiben $N - 1 = u * 2^k$, für u ungerade, $k \geq 1$. Eine Zahl $a, 1 \leq a < N$, heißt ein MR-Zeuge für N, wenn $b_0 = 1$ oder in der Folge $b_0, ..., b_{k-1}$ zu a kommt $N-1$ vor nicht gilt, d. h. $a^u \not\equiv 1$ und $a^{u*2^i} \not\equiv N-1 (mod \ N)$ für alle i mit $0 \leq i < k$ (Fälle 3 und 4). Eine Zahl $a, 1 \leq a < N$, heißt ein MR-Lügner für N, wenn $b_0 = 1$ oder in der Folge $b_0, ..., b_{k-1}$ zu a kommt $N-1$ vor gilt, $a^u \equiv 1$ oder $a^{u*2^i} \equiv N-1 (mod \ N)$ für ein i mit $0 \leq i < k$ (Fälle 1 und 2). Die Menge der MR-Lügner nennen wir L_N^{MR}.

(g) Ergänze den Algorithmus von Miller/Rabin (Eingabe $N \geq 5$): Funktion Miller-Rabin-Primzahltest(N) Bestimme ... u und $k \geq 1$ so, dass ... Wähle ... $b \leftarrow$... if $b \in \{...\}$ then ... for j from 1 to $k-1$ do $b \leftarrow$ ... if $b = ...$ then ... if $b = ...$ then ... return

Antwort: Der Miller-Rabin-Primzahltest

- Bestimme u ungerade und $k \geq 1$ mit $N - 1 = u * 2^k$
- wähle zufällig ein a aus $\{1, ..., N-1\}$
- $b \leftarrow a^u \ mod \ N$ // mit schnellem Potenzieren
- if $b \in \{1, N-1\}$ then return 0
- for j from 1 to $k-1$ do //„wiederhole (k-1)-mal"
- $b \leftarrow b^2 \ mod \ N$
- if $b = N - 1$ then return 0
- if $b = 1$ then return 1
- return 1

(h) Was kann man über das Ein-/Ausgabeverhalten des Miller-Rabin-Algorithmus auf Eingabe $N \geq 5$ (ungerade) sagen? N zusammengesetzt $\Rightarrow$ ..., N Primzahl $\Rightarrow$...

Antwort: Wenn N zusammengesetzt $\Rightarrow$ gibt es MR-Zeugen.
Wenn N eine Primzahl ist, gibt der MR-Test 0 aus.

(i) Wie kann man vorgehen, um aus dem Miller-Rabin-Test einen Primzahltest zu erhalten, dessen Fehlerwahrscheinlichkeit höchstens $1/4^l$ beträgt?

Antwort: Tatsächlich ist die Fehlerwahrscheinlichkeit durch $1/4^l$ beschränkt, für (von l abhängig) genügend großen. Dies kann man aber nur durch fortgeschrittene zahlentheoretische Untersuchungen über die erwartete Wahrscheinlichkeit, dass eine zufällige ungerade zusammengesetzte Zahl den l-fach iterierten MiRa-Test übersteht, beweisen.

(j) Formuliere den Primzahlsatz:

Antwort: Primzahlsatz: $lim_{x \to \infty} \frac{\pi(x)}{x \backslash ln \ x} = 1$.
Mit $\pi(x)$ bezeichnen wir die Anzahl der Primzahlen, die nicht größer als x sind.

(k) Nach der Ungleichung von Finsler gibt es $\Omega(...)$ Primzahlen im Intervall $[m, 2m)$. Entsprechend muss man für $\mu \in \mathbb{N}$ erwartet nur $O(...)$ Zahlen zufällig aus $[2^{\mu-1}, 2^\mu)$ ziehen, um mindestens eine μ-Bit Primzahl zu erhalten.

Antwort: Ungleichung von Finsler: Für jede ganze Zahl $m \geq 2$ liegen im Intervall $(m, 2m]$ mindestens $m/(3 \ ln(2m))$ Primzahlen: $\pi(2m) - \pi(m) \geq \frac{m}{3 \ ln(2m)}$.
$\Rightarrow \pi(2m) - \pi(m) = O(m/log \ m)$

(l) Zu gegebenem μ soll eine (zufällige) Primzahl im Intervall $[2^{\mu-1}, 2^\mu)$ gefunden werden. Wie geht man vor? wiederhole: ... bis Ergebnis ... erscheint. Wie lässt sich die erwartete Anzahl von Bitoperationen für das Finden einer solchen Primzahl abschätzen? $O(...)$.

Antwort:

6. Das RSA-System

(a) Schlüsselerzeugung: Wähle .... und berechne $N = ...$ sowie $\varphi(N) = ...$ Der öffentliche Schlüssel von Bob ist (N, e), wobei e die Bedingung ... erfüllt. Der geheime Schlüssel von Bob ist (N, d), mit ... . d lässt sich mit folgendem Algorithmus berechnen: ...

Antwort:
Die Schlüssellänge ist festzulegen (etwa $l = 1024, 2048$ oder 4096 Bits). Danach werden zwei (zufällige, verschiedene) Primzahlen p und q bestimmt, deren Bitlänge die Hälfte der Schlüssellänge ist. Nun wird das Produkt $N = pq$ berechnet. Die Zahl N hat l oder $l-1$ Bits. Weiter wird $\varphi(N) = (p-1)(q-1)$ berechnet. Es wird eine Zahl $e \in \{3, ..., \varphi(N) - 1\}$ mit $ggT(e, \varphi(N)) = 1$ gewählt. Dann wird das multiplikative Inverse $d < \varphi(N) modulo \ \varphi(N)$ von e bestimmt, so dass also $ed \ mod \ \varphi(N) = 1$ gilt. (Man beachte, dass nur ungerade e in Frage kommen, weil $\varphi(N)$ gerade ist. Man weiß, dass die Anzahl der geeigneten Werte e mindestens $\frac{\varphi(N)}{log(\varphi(N))}$ ist, so dass die erwartete Anzahl von Versuchen $O(log(\varphi(N))) = O(logN)$ ist.)
Der erwartete Berechnungsaufwand für die Schlüsselerzeugung ist $O((log \ N)^4) = O(l^4)$, weil dies die Kosten der Primzahlerzeugung sind.
Bob erhält aus seiner Rechnung N, e und d. Aus diesen wird das Schlüsselpaar $(k, \hat{k})$ gebildet:

- Der öffentliche Schlüssel k ist das Paar (N, e). Dieser wird bekanntgegeben.
- Der geheime Schlüssel $\hat{k}$ ist (N, d). (Natürlich ist nur der Teil d wirklich geheim.)

(b) Verschlüsseln von $x \in ... : y = ...$

Antwort: $x \in X = [N] : y = E(x, (N, e)) := x^e \ mod \ N$
(Zu berechnen mit schneller Exponentiation, Rechenzeit $O((log\ N)^3) = O(l^3)$)

(c) Entschlüsseln von $y \in ... : z = ...$

Antwort: $y \in Y : z = D(y, (N, d)) := y^d \ mod \ N$
(Zu berechnen mit schneller Exponentiation, Rechenzeit $O((log\ N)^3) = O(l^3)$)

(d) Formuliere die zentrale Korrektheitsaussage des RSA-Systems: ...=x, für alle zulässigen Klartextblöcke x.

Antwort: Korrektheit/Dechiffrierbedingung von RSA: Wenn $ed \ mod \ \varphi(N) = 1$ gilt, dann haben wir $x^{ed} \ mod \ N = x$, für alle $x \in [N]$.

(e) Beschreibe eine Strategie für RSA-basierte Systeme, mit der verhindert werden kann, dass zwei identische Klartextblöcke bei Verwendung desselben Schlüsselpaars gleich verschlüsselt werden.

Antwort: Verwendung von Prä- und Postblöcken, die vor und nach dem zu verschlüsselnden Textblock angehängt werden mit randomisiertem (Zufallsbits) oder zeitlich abhängigem (Zeitstempel) Inhalt.
Es ist Empfohlen, beim Arbeiten mit RSA den Klartext x durch das Anhängen eines nicht ganz kurzen Zufallsstrings zu randomisieren. Wenn dieser angehängte Zufallsstring die gleiche Länge wie x hat, ist der Chiffretext genauso lang wie bei ElGamal.

7. Das Rabin-Kryptosystem

(a) Komponenten des Rabin-Kryptosystems: Zwei große Primzahlen p und q mit ... . Der öffentliche Schlüssel ist $N =\ldots$, der private Schlüssel von Bob ist ... .

Antwort: Wähle zwei verschiedene zufällige große Primzahlen p und q mit $p \equiv q \equiv 3(mod\ 4)$, also Primzahlen, die um 1 kleiner als ein Vielfaches von 4 sind. Berechne $N = pq$. Der öffentliche Schlüssel ist $k = N$; der geheime Schlüssel ist $\hat{k} = (p, q)$.

(b) Verschlüsselung: Alice möchte einen Block $x \in \ldots$ an Bob schicken. Sie berechnet $y =\ldots$ und sendet y an Bob.

Antwort: Verschlüsselung eines Blocks, der eine Zahl $x < N$ ist: $y := x^2 \ mod \ N$.

(c) Entschlüsselung: Wenn Bob das Chiffrat y erhält, berechnet er $z_1, ...z_4$. Wie hängen diese Zahlen mit y zusammen? ... Mit welchen Formeln und welcher Methode berechnet Bob diese vier Zahlen? modulo p:... modulo q:... Kombination der Teilergebnisse, um (zum Beispiel) z_1 zu erhalten: ... Was ist der maximale Rechenaufwand? $O(...)$ Bitoperationen.

Antwort: Entschlüsselung eines Chiffretextes $y < N$: Wir müssen Quadratwurzeln von y modulo N berechnen, das sind Zahlen b mit $b^2 mod\ N = y$. Wir kennen die Faktoren p und q. Berechne Quadratwurzeln getrennt modulo p und modulo $q : r := y^{(p+1)/4} mod\ p$ und $s := y^{(q+1)/4} mod\ q$. Weil $r^2 \ mod\ p = ((x^2 \ mod\ N)^{(p+1)/4})^2 \ mod\ p = x^{p+1} \ mod\ p = (x^p * x) mod\ p = x^2 \ mod\ p$, gilt $r^2 - x^2 \equiv (r - x)(r + x) \equiv 0(mod\ p)$. Das heißt, dass entweder $r \equiv x(mod\ p)$ oder $p - r \equiv x(mod\ p)$ gilt. Genauso sieht man, dass $s \equiv x(mod\ q)$ oder $q - s \equiv x(mod\ q)$ gilt. Mit der konstruktiven Variante des chinesischen Restsatzes können wir nun vier Zahlen $z_1, ..., z_4 \in [N]$ berechnen, die die folgenden Kongruenzen erfüllen

- $z_1 \equiv r(mod\ p)$ und $z_1 \equiv s(mod\ q)$
- $z_2 \equiv r(mod\ p)$ und $z_2 \equiv q - s(mod\ q)$
- $z_3 \equiv p - r(mod\ p)$ und $z_3 \equiv s(mod\ q)$
- $z_4 \equiv p - r(mod\ p)$ und $z_4 \equiv q - s(mod\ q)$

Wegen der obigen Überlegung ist $x \in \{z_1, ..., z_4\}$. Wir wählen eine dieser vier Möglichkeiten.
Für die Verschlüsselung muss nur eine Quadrierung modulo N durchgeführt werden; sie kostet nur Zeit $O((log\ N)^2)$. Die Entschlüsselung erfordert eine Exponentiation modulo p und eine modulo q und mehrere Anwendungen des erweiterten Euklidischen Algorithmus - insgesamt Zeit $O((log\ N)^3)$.

(d) Formuliere die zentrale Sicherheitsaussage des Rabin-Kryptosystems: ...

Antwort: Welche der oberen Möglichkeiten die richtige (ausgewählte) ist, hängt vom Zufall ab, der die Auswahl von x steuert. Jede der 4 Quadratwurzeln von y hat dieselbe Wahrscheinlichkeit 1/4, als x gewählt worden zu sein.

1. Fall: $x = z$, Misserfolg.
2. Fall: $0 < |x - z| < N$ und $x - z$ durch p teilbar, woraus $ggT(x - z, N) = p$ folgt: Erfolg!
3. Fall: $0 < |x - z| < N$ und durch q teilbar, woraus $ggT(x - z, N) = q$ folgt: Erfolg!

4. Fall: $x + z = N$, also $x - z \equiv 2x(mod\ N)$. Weil $2x$ teilerfremd zu N ist, ergibt sich $ggT(x - z, N) = 1$, Misserfolg.

Eva muss also nur $ggT(x - z, N)$ berechnen! Damit gelingt es ihr mit Wahrscheinlichkeit $1/2$, die Faktoren von N zu ermitteln. Durch l-fache Wiederholung desselben Experiments lässt sich die Erfolgswahrscheinlichkeit auf $1 - \frac{1}{2^l}$ erhöhen.

8. Diskreter Logarithmus und das ElGamal-Kryptosystem

Gegeben sei eine zyklische Gruppe $(G, \circ, e)$ der Ordnung (Kardinalität) N mit erzeugendem Element g.

(a) Definiere die Exponentiation mit Basis g und den Logarithmus zur Basis g jeweils mit Definitions- und Wertebereich. exp_g:... →..., ...→... log_g:... →..., ...→... Für die Berechnung der Exponentiation werden $O(...)$ Gruppenoperationen benötigt.

Antwort:

(b) Um die Schlüssel festzulegen, wählt Bob zufällig eine geheime Zahl $b \in$ .... Der öffentliche Schlüssel ist ... mit $B =$...

Antwort: Es wird eine zyklische Gruppe $(G, \circ, e)$ mit einem erzeugenden Element g benötigt, sowie $N = |G|$, so dass das zugehörige DH-Problem schwer zu lösen ist. Ein Element b wird zufällig aus $\{2, ..., |G| - 2\}$ gewählt, und es wird mittels schneller Exponentiation $B = g^b$ berechnet. Der öffentliche Schlüssel ist $k_{pub} = (G, g, B)$, der geheime Schlüssel ist b bzw. $k_{priv} = (G, g, b)$.

(c) Verschlüsselung von Klartextblock $x \in$... mit öffentlichem Schlüssel: ...

Antwort: Wir nehmen an, dass die Menge der möglichen Botschaften (Binärstrings) eine Teilmenge von G ist. Um eine Botschaft $x \in G$ zu verschlüsseln, wählt Alice eine Zufallszahl a aus $\{2, ..., |G| - 2\}$ und berechnet $A = g^a$. Weiter berechnet sie $y := B^a \circ x$. Der Chiffretext ist (A, y).

(d) Entschlüsselung von Chiffretext ... ∈ ... mithilfe von b: ...

Antwort: Bob kennt die Gruppe G und g, sowie A und y (von Alice) sowie seinen geheimen Schlüssel b. Er berechnet $A^b = (g^a)^b = k$. Dann berechnet er das Gruppenelement $z = k^{-1} \circ y$, mit Hilfe der effizienten Invertierung und Gruppenoperation in G.

(e) Gebe das Diffie-Hellman-Problem (DH-Problem) an: Zu Input ..., ... finde ... .

Antwort: Die Idee dabei ist, dass $k = g^{ab}$ ist, wo nur Alice a kennt und nur Bob b. Über den öffentlichen Kanal laufen die Gruppenelemente g^a und g^b. Eva hat also das Problem, aus g^a und g^b den Wert g^{ab} zu berechnen.
Zu Input $k = g^{ab}$, wobei nur Alice a kennt und nur Bob b kennt, finde g^a und g^b.

(f) Zur Sicherheit des ElGamal-Kryptosystems lässt sich feststellen: Eve kann alle bzgl. G und g verschlüsselten Nachrichten effizient entschlüsseln genau dann wenn ...

Antwort:

1. Eva kann alle mit dem ElGamal-Verfahren bzgl. G und g verschlüsselten Nachrichten effizient entschlüsseln, also aus B, A und y die Nachricht x berechnen, die zum Chiffretext (A, y) geführt hat.
2. Eva kann das DH-Problem für G lösen.

Wenn Eva diskrete Logarithmen bezüglich G und g berechnen kann, gelten natürlich 1. und 2. Wir beweisen die Äquivalenz.

- „1.⇒2.": Eva hat $B = g^b$ und $A = g^a$ vorliegen und möchte $k = g^{ab}$ bestimmen. Sie wendet ihr Entschlüsselungsverfahren auf B, A und $y = 1$ an. Es ergibt sich ein Wert x mit $g^{ab} \circ x = k \circ x = y = 1$. Es gilt also $x = k^{-1}$, und Eva kann k durch Invertierung von x in G berechnen.
- „2.⇒1.": Eva hat $B = g^b$,$A = g^a$,$y = g^{ab} \circ x$ vorliegen. Weil sie das DH-Problem lösen kann, kann sie $k = g^{ab}$ berechnen und damit natürlich $x = k^{-1} \circ y$ bestimmen.

(g) Wieso verwendet man in der Praxis lieber Systeme, die auf elliptischen Kurven basieren, als solche, die auf diskreten Logarithmen beruhen?

Antwort:

- wesentlich kleinere Schlüsselmenge bei deutlich höherer Komplexität
- Bisher kein Verfahren zum „zurück-rechnen" vom Öffentlichen zum Privaten Schlüssel bekannt
- nur durch Brute Force möglich zu knacken
- geringer Aufwand bei hoher Sicherheit