Die Klausur

Dienstag den 19.2. um 9 Uhr im Felix-Klein-Hörsaal

Für die Klausur ...

... sind die Sicherheits-Definitionen ganz zentral.

Eine Aufgabe könnte sein:

a) Wofür ist "IND-CPA-sicher" die Abkürzung?
b) Wie lautet die formale Definition davon, dass ein kryptographisches System mit privatem Schlüssel IND-CPA-sicher ist?
c) Begründen Sie: Ein deterministisches Verschlüsselungsverfahren ist nicht IND-CPA-sicher.

Aktueller Überblick

§0 Einführende Worte (✓)

§1 Geschichtlicher Überblick (✓)

§2 Zufall (✓)

§3 Perfekte Sicherheit und ihre Grenzen (✓)

§4 Angriffsszenarien (✓)

§5 Der komplexitätstheoretische Ansatz(✓)

§6 Pseudozufallsgeneratoren und Stromchiffren (✓)

§7 Pseudozufallsfunktionen und Blockchiffren

§8 Message Authentication Codes

§9 ...

§7 Pseudozufallsfunktionen und Blockchiffren

(IND-CPA-sichere Verschlüsselungssysteme)

Blockchiffren

Blockchiffren

Praxisorientierte Idee. Eine Blockchiffre verschlüsselt und entschlüsselt (größere) Blöcke.

Eine Stromchiffre verschlüsselt Bit für Bit und hat dabei einen Zustand.

[Aber: Wenn man mit einer Blockchiffre mehr als einen Block verschlüsseln will, muss man einen vernünftigen "Modus" verwenden.]

Blockchiffren

Welche Anforderungen soll man an Blockchiffren stellen?

Wir wollen den üblichen komplexitätstheoretischen Zugang wählen.

Wie immer brauchen wir einen Sicherheitsparameter n.

Wir haben wieder (zufällige) Schlüssel k, Nachrichten / Klartexte m und Chiffriertexte c.

Der Einfachheit halber sollen

Schlüssel, Nachrichten und Chiffriertexte dieselbe Länge haben n,

der Schlüssel uniform zufällig in $\{0,1\}^n$ gewählt werden.

Blockchiffren

Für festen Sicherheitsparameter n definiert nun $\mathcal{E}nc$ eine Funktion

$$\{0,1\}^n \times \{0,1\}^n \longrightarrow \{0,1\}^n \ , \ (k^{(0)}, m^{(0)}) \mapsto \mathcal{E}nc_{k^{(0)}}(m^{(0)}) \ .$$

Für festes n ist das eine Funktion

$$f : \{0,1\}^n \times \{0,1\}^n \longrightarrow \{0,1\}^n \ , \ (k^{(0)}, m^{(0)}) \mapsto f_{k^{(0)}}(m^{(0)}) \ .$$

Das kann man als Funktion in zwei Variablen betrachten

oder als eine Schar von Funktionen:

$$(f_{k^{(0)}})_{k^{(0)} \in \{0,1\}^n} : \{0,1\}^n \longrightarrow \{0,1\}^n \ .$$

Blockchiffren

Für variables n:

$$(f_{k^{(0)}})_{k^{(0)} \in \{0,1\}^*}$$

mit $n = |k|$: $f_{k^{(0)}} : \{0,1\}^n \longrightarrow \{0,1\}^n$.

Wir bezeichnen die Variable mit x (statt mit m).

Für festes n haben wir die Zufallsvariable k mit Werten in $\{0,1\}^n$. Diese definiert eine "zufällige Funktion" f_k.

Also: Für jeden Wert $K^{(0)}$ von k haben wir eine Funktion $f_{k^{(0)}}$, insgesamt eine zufällige Funktion.

Pseudozufallsfunktionen

Zufällige Funktionen

Es sei eine Schar

$$(f_{k^{(0)}})_{k^{(0)} \in \{0,1\}^*}$$

von Funktionen gegeben mit: mit $n = |k^{(0)}|$ ist

$$f_{k^{(0)}} : \{0,1\}^n \longrightarrow \{0,1\}^n \, .$$

Für festen Sicherheitsparameter n und k uniform zufällig auf $\{0,1\}^n$ haben wir die zufällige Funktion f_k.

Diese vergleichen wir mit

“echte zufälligen Funktionen”,

“echt zufälligen Bijektionen” (in der Kryptographie: Substitutionen)

Zufällige Funktionen

Was soll eine “echte zufällige Funktion” sein?

Zufällige Funktionen

Ein "echt zufälliger Bitstring" r der Länge n ist eine Zufallsvariable mit Werten in $\{0,1\}^n$, der uniform zufällig unter allen 2^n solchen Bitstrings ist.

Algorithmische Beschreibung: Es werden zufällige Bits $r_1, \ldots, r_n$ unabhängig gewählt. Dann ist $r = r_1 \ldots r_n$ uniform zufällig gewählt in $\{0,1\}^n$ ("echt zufällig").

Zufällige Funktionen

Eine uniform zufällige Funktion $\{0,1\}^n \longrightarrow \{0,1\}^\ell$ ist eine zufällige Funktion, die uniform zufällig unter allen diesen Funktionen ist.

Algorithmische Beschreibung: Für alle $x \in \{0,1\}^n$ werden uniform und unabhängig Werte $f(x)$ in $\{0,1\}^\ell$ gewählt.

Eine unform zufällige Permutation auf $\{0,1\}^n$ ist eine zufällige Permutation auf $\{0,1\}^n$, die uniform zufällig unter allen Permutationen auf $\{0,1\}^n$ ist.

Algorithmische Beschreibung: Für $x \in \{0,1\}^n$ werden (in irgendeiner Reihenfolge) uniform Werte $f(x) \in \{0,1\}^n$ verschieden von den bisherigen Werten gewählt.

Pseudozufallsfunktionen

Es sei eine Schar

$$(f_{k^{(0)}})_{k^{(0)} \in \{0,1\}^*}$$

von Funktionen gegeben mit: mit $n = |k^{(0)}|$ ist $f_{k^{(0)}} : \{0,1\}^n \longrightarrow \{0,1\}^n$.

Ideen

Wenn f_k (mit k uniform auf $\{0,1\}^n$, n variabel) nicht effizient unterscheidbar ist von einer "echt zufälligen" Funktion: Die Schar heißt Pseudozufallsfunktion.

Wenn f_k nicht effizient unterscheidbar ist von einer "echt zufälligen" Bijektion: Die Schar heißt Pseudozufallspermutation.

Wie immer ist ein Unterscheider $\mathcal{D}$ ein beliebiger Algorithmus.

Spiel für Pseudozufallsfunktionen

Eingabe: Sicherheitsparameter 1^n

1. Unterscheider und Herausforderer erhalten 1^n.
2. Der Herausforder wählt $i \in \{0,1\}$ unform zufällig
   Es wird wie folgt eine Funktion $f : \{0,1\}^n \longrightarrow \{0,1\}^n$ definiert:
   - Wenn $i = 0$: $f : \{0,1\}^n \longrightarrow \{0,1\}^n$ wird uniform zufällig gewählt.
   - Wenn $i = 1$: k in $\{0,1\}^n$ wird uniform zufällig gewählt, $f := f_k$.
3. Der Unterscheider kann jeweils für $x \in \{0,1\}^n$ den Wert $f(x)$ anfordern / erhalten.
4. Der Unterscheider entscheidet sich für $j \in \{0,1\}$.

Der Unterscheider hat gewonnen, wenn $i = j$ ist.

Der Erfolg / Vorteil des Unterscheiders $\mathcal{D}$ ist ("wie immer")

$$2 \cdot (\mathbf{P}[\mathcal{D} \text{ gewinnt}] - \frac{1}{2}) .$$

Pseudozufallsfunktionen

Definition. Eine Pseudozufallsfunktion ist eine Schar von Funktionen $(f_{k^{(0)}})_{k^{(0)} \in \{0,1\}^*}$ mit:

Für $n = |k^{(0)}|$ ist $f_{k^{(0)}} : \{0,1\}^n \longrightarrow \{0,1\}^n$.

Und:

- $(k^{(0)}, x^{(0)}) \mapsto f_{k^{(0)}}(x^{(0)})$ ist effizient berechenbar
- Jeder PPT-Unterscheider im Spiel hat nur vernachlässigbar kleinen Erfolg.

Beachte. Die $f_{k^{(0)}}$'s müssen nicht bijektiv sein.

Exkurs: Algorithmen und Funktionen

Exkurs: Algorithmen und Funktionen

Ein Pseudozufallsgenerator ist ein deterministischer Algorithmus, eine Pseudozufallsfunktion eine Funktion.

Das ist aber nicht der wesentliche Unterschied.

Exkurs: Algorithmen und Funktionen

Funktion versus Algorithmus:

Mengentheoretisch-dogmatische Position:

Eine Funktion $f : X \longrightarrow Y$ ist per Definition eine Relation zwischen X und Y, d.h. eine Teilmenge von $X \times Y$.

Die Funktion f ist per Definition identisch mit ihrem Graphen $\Gamma_f \subseteq X \times Y$. Es soll gelten:

$$\forall x \in X : \exists! y \in Y : x \sim_f y$$

Ein Algorithmus ist dasselbe wie (z.B.) eine Turing-Maschine.

Exkurs: Algorithmen und Funktionen

Aber: Wir betrachten explizit gegebene Funktionen

und wir gehen mit Algorithmen intuitiv um.

Dann:

Funktionen sind stets (explizit) regelbasiert = (explizit) algorithmisch definiert.

Algorithmen definieren Funktionen und sind intuitiv gegeben.

$\Longrightarrow$ In der Anwendung so gut wie kein Unterschied.

[Außer: Zwei Funktionen (im Sinne von Regeln) gelten genau dann als gleich, wenn für Wert Definitionsbereichs die Bilder übereinstimmen.]

Ein einfaches IND-CPA-sicheres Verfahren

Ein IND-CPA-sicheres Verfahren

Es sei eine Funktionsschar $(f_{k^{(0)}})_{k^{(0)}}$ wie zuvor gegeben
$((k^{(0)}, x^{(0)}) \mapsto f_{k^{(0)}}(x^{(0)})$ ist in Polynomzeit berechenbar).

$\mathcal{E}nc$. Eingabe: k, m

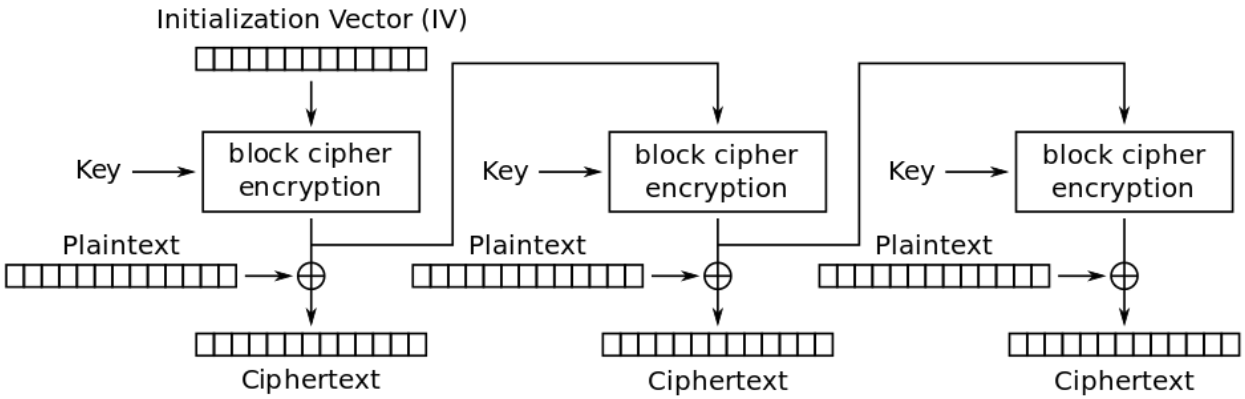
Wähle $r \in \{0,1\}^n$ uniform.

Ausgabe $(r, f_k(r) \oplus m)$.

$\mathcal{D}ec$. Eingabe $c = (r, a)$

Ausgabe $f_k(r) \oplus a = m$.

Output Feedback Mode (OFB)



Output Feedback (OFB) mode encryption

Quelle. Wikipedia, Benutzer WhiteTimberwolf

Ein IND-CPA-sicheres Verfahren

Satz. Wenn die Funktionsschar eine Pseudozufallsfunktion definiert, ist das beschriebene Verfahren IND-CPA-sicher.

Beweisskizze

1. Beweis im Zufallsorakelmodell.
2. Vergleich des Erfolgs eines Angreifers im Zufallsorakelmodell und "in echt" ("Standardmodell").

Das Spiel zum CPA

Gegeben: $\mathcal{G}en, \mathcal{E}nc, \mathcal{D}ec$.

Eingabe: Sicherheitsparameter 1^n.

1. Angreifer und Herausforderer erhalten 1^n.
2. Der Herausforderer erzeugt einen Schlüssel $k := \mathcal{G}en(1^n)$.
3. Der Angreifer kann beliebige Nachrichten m mit dem Schlüssel k verschlüsseln lassen (d.h. $\mathcal{E}nc_k(m)$ berechnen lassen).
4. Der Angreifer wählt zwei verschiedene Nachrichten m_1, m_2 gleicher Länge. Er schickt diese an den Herausforderer.
5. Der Herausforderer wählt $i \in \{1, 2\}$ uniform. Er berechnet $c := \mathcal{E}nc_k(m_i)$ und schickt c an den Angreifer.
6. Der Angreifer kann beliebige Nachrichten mit dem Schlüssel k verschlüsseln lassen.
7. Der Angreifer entscheidet sich für $j \in \{0, 1\}$.

Der Angreifer hat gewonnen, wenn $i = j$ ist.

Das Spiel zum CPA

Eingabe: Sicherheitsparameter 1^n.

1. Angreifer und Herausforderer erhalten 1^n.
2. Der Herausforderer erzeugt einen Schlüssel k (uniform in $\{0,1\}^n$).
3. Der Angreifer kann zu $m \in \{0,1\}^n$ beliebig $(r, f_k(r) \oplus m)$ anfordern (r in $\{0,1\}^n$ uniform).
4. Der Angreifer wählt zwei verschiedene Nachrichten m_1, m_2. Er schickt diese an den Herausforderer.
5. Der Herausforderer wählt $i \in \{1,2\}$ uniform. Er wählt r^* in $\{0,1\}^n$ uniform, berechnet $c := (r, f_k(r) \oplus m_i)$ und schickt c an den Angreifer.
6. Der Angreifer kann zu $m \in \{0,1\}^n$ beliebig $(r, f_k(r) \oplus m)$ anfordern (r in $\{0,1\}^n$ uniform).
7. Der Angreifer entscheidet sich für $j \in \{0,1\}$.

Der Angreifer hat gewonnen, wenn $i = j$ ist.

CPA-Sicherheit im Zufallsorakelmodell

Das Verfahren ist CPA-sicher im Zufallsorakelmodell.

Begründung

Es sei $u : \{0,1\}^n \longrightarrow \{0,1\}^n$ uniform zufällig.

Es wird dieses Spiel betrachtet:

Spiel im Zufallsorakelmodell

Eingabe: Sicherheitsparameter 1^n.

1. Angreifer und Herausforderer erhalten 1^n.
2. Der Herausforderer erzeugt einen Schlüssel k (uniform in $\{0,1\}^n$).
3. Der Angreifer kann zu $m \in \{0,1\}^n$ beliebig $(r, u(r) \oplus m)$ anfordern (r in $\{0,1\}^n$ uniform).
4. Der Angreifer wählt zwei verschiedene Nachrichten m_1, m_2. Er schickt diese an den Herausforderer.
5. Der Herausforderer wählt $i \in \{1,2\}$ uniform. Er wählt r^* in $\{0,1\}^n$ uniform, berechnet $c := (r, u(r) \oplus m_i)$ und schickt c an den Angreifer.
6. Der Angreifer kann zu $m \in \{0,1\}^n$ beliebig $(r, u(r) \oplus m)$ anfordern (r in $\{0,1\}^n$ uniform).
7. Der Angreifer entscheidet sich für $j \in \{0,1\}$.

Der Angreifer hat gewonnen, wenn $i = j$ ist.

CPA-Sicherheit im Zufallsorakelmodell

Ein Angreifer kann beliebige Nachrichten $m \in \{0,1\}^n$ verschlüsseln lassen.

Er erhält: $c = (r, u(r) \oplus m)$.

Das ist "genauso gut" wie $(r, u(r))$ zu erhalten.

Man kann das Spiel so umformulieren:

Wir gehen vom IND-EAV-Spiel aus.

Zusätzlich kann der Angreifer vom Herausforderer stets Tupel $(r, u(r))$ erhalten (r uniform zufällig).

Neues Spiel

Eingabe: Sicherheitsparameter 1^n.

1. Angreifer und Herausforderer erhalten 1^n.
2. Der Herausforderer erzeugt einen Schlüssel k (uniform in $\{0,1\}^n$).
3. Der Angreifer kann $(r, u(r))$ anfordern (r in $\{0,1\}^n$ uniform).
4. Der Angreifer wählt zwei verschiedene Nachrichten m_1, m_2. Er schickt diese an den Herausforderer.
5. Der Herausforderer wählt $i \in \{1,2\}$ uniform. Er wählt r^* in $\{0,1\}^n$ uniform und berechnet $c := (r^*, u(r^*) \oplus m_i)$ und schickt c an den Angreifer.
6. Der Angreifer kann $(r, u(r))$ anfordern (r in $\{0,1\}^n$ uniform).
7. Der Angreifer entscheidet sich für $j \in \{0,1\}$.

Der Angreifer hat gewonnen, wenn $i = j$ ist.

CPA-Sicherheit im Zufallsorakelmodell

Also: Der Angreifer kann $(r, u(r))$ erhalten, wenn er will.

- r ist uniform zufällig.
- $u(r)$ ist auch uniform zufällig und unabhängig von r, außer r wurde schon mal gewählt.

Nur wenn einmal $r = r^*$ ist, bringt das was.

Ansonsten könnte er r und $u(r)$ auch selbst erzeugen.

CPA-Sicherheit im Zufallsorakelmodell

Die Wahrscheinlichkeit, dass r^* getroffen wird, ist $\leq \frac{t}{2^n}$.

Der Erfolg ist $\leq \frac{t}{2^n}$.

Für t polynomiell in n ist der Erfolg vernachlässigbar klein.

Vergleich

Vergleich eines Angreifers im Zufallsorakelmodell und “in echt” (“Standardmodell”):

Idee. Wir haben ein Spiel (CPA-Spiel) in zwei Varianten (mit f_k und mit u).

Wir machen aus dem Spiel einen Unterscheider $\mathcal{D}$ für die Funktionsschar $(f_{k^{(0)}})_{k^{(0)}}$.

Unterscheider

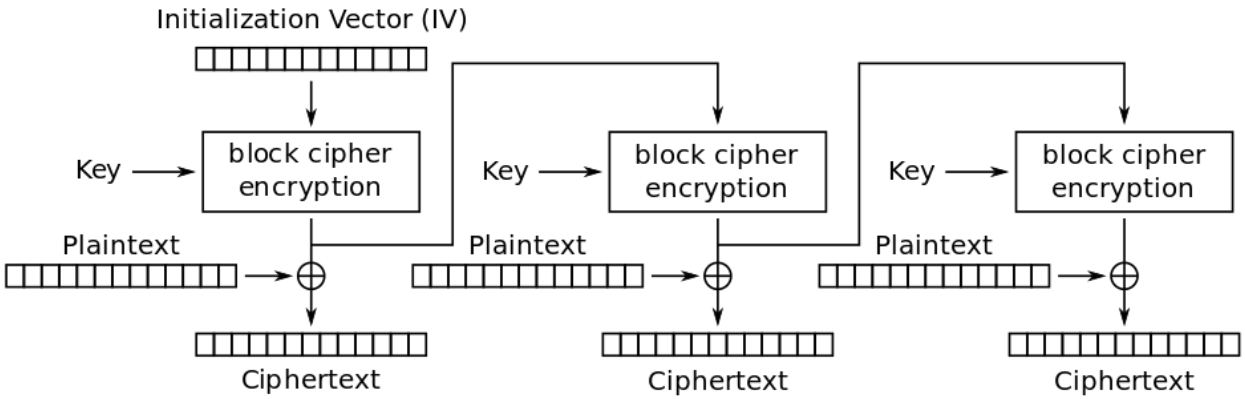
Eingabe: Sicherheitsparameter 1^n

1. Wähle $i \in \{0,1\}$ unform zufällig.
   Es wird wie folgt eine Funktion $f : \{0,1\}^n \longrightarrow \{0,1\}^n$ definiert:
   - Wenn $i = 0$: $f = u$.
   - Wenn $i = 1$: k in $\{0,1\}^n$ wird uniform zufällig gewählt, $f := f_k$.
2. Führe das CPA-Spiel durch mit dem Verfahren, das auf der Funktion f beruht (s.o.) und Angreifer $\mathcal{A}$.
3. Wenn $\mathcal{A}$ gewonnen hat: Gib 1 = "nicht zufällig" aus, wenn $\mathcal{A}$ verloren hat, gib 0 = "zufällig" aus.

$$\begin{aligned}
&\text{Erfolg von } \mathcal{D} = 2 \cdot (\mathbf{P}[\mathcal{D} \text{ gewinnt}] - \tfrac{1}{2}) \\
= \quad &\mathbf{P}[\mathcal{D} \text{ gewinnt} \mid i = 1] + \mathbf{P}[\mathcal{D} \text{ gewinnt} \mid i = 0] - 1 \\
= \quad &\mathbf{P}[\mathcal{D} \text{ gewinnt} \mid i = 1] + \mathbf{P}[\mathcal{D} \text{ verliert} \mid i = 0] \\
= \quad &\mathbf{P}[\mathcal{A} \text{ gewinnt im echten Spiel}] - \mathbf{P}[\mathcal{A} \text{ gewinnt im ZOM}]
\end{aligned}$$

Output Feedback (OFB) Mode

Output Feedback Mode (OFB)



Output Feedback (OFB) mode encryption

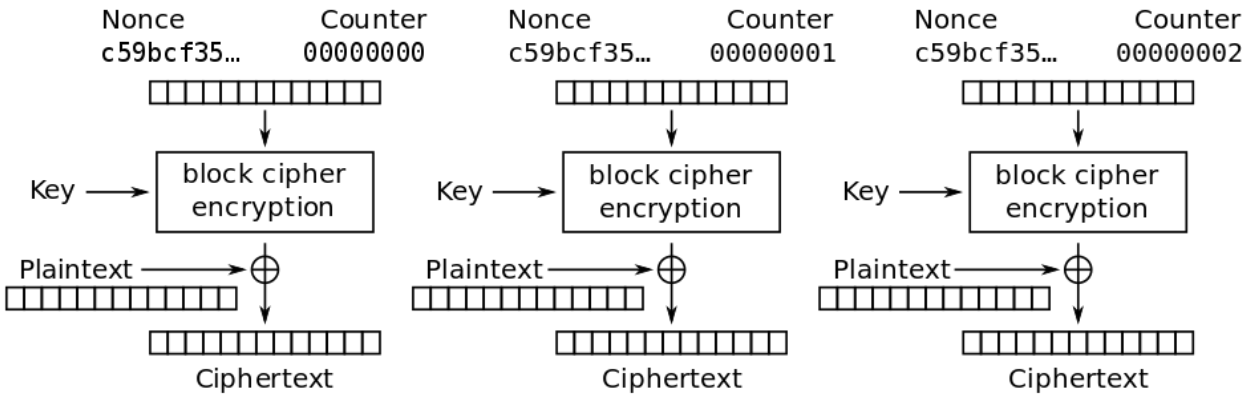
Quelle. Wikipedia, Benutzer WhiteTimberwolf

Output Feedback (OFB) Mode

Satz. Für eine pseudozufällige Funktion ist der Output Feedback Mode (OFB-Modus) IND-CPA-sicher.

Couter (CTR) Mode

Counter (CTR) Mode



Counter (CTR) mode encryption

Quelle. Wikipedia, Benutzer WhiteTimberwolf

Couter (CTR) Mode

Satz. Für eine Pseudozufallspermuation ist der Counter Mode (CTR-Modus) IND-CPA-sicher.

Pseudozufallspermutationen und der CBC-Modus

Pseudozufallspermutationen

Satz. Eine uniforme Zufallspermutation ist nicht effizient unterscheidbar von einer uniformen Zufallsfunktion.

Genauer: Jeder PPT-Unterscheider hat vernachlässigbaren Erfolg.

Beweisidee

Der beste Algorithmus ist:

Anfragen stellen.

Wenn "Doppelte", dann Ausgabe "Pseudozufallsfunktion",

ansonsten Ausgabe "Pseudozufallspermutation".

Die Wahrscheinlichkeit für "Doppelte" ist vernachlässigbar.

Pseudozufallspermutationen

Satz. Eine uniforme Zufallspermutation ist nicht effizient unterscheidbar von einer uniformen Zufallsfunktion (mit gleicher Ein- und Ausgabelänge).

Außerdem per Definition:

Eine uniforme Zufallsfunktion ist nicht effizient unterscheidbar von einer Pseudozufallsfunktion.

Damit:

Eine Pseudozufallspermutation ist nicht effizient unterscheidbar von einer uniformen Zufallsfunktion.

Damit per Definition:

Satz. Eine Pseudozufallspermutation ist auch eine Pseudozufallsfunktion.

Aufgabe 2

Wir betrachten das folgende Verschlüsselungsverfahren aus der Vorlesung für eine Pseudozufallspermutation $(f_{k^{(0)}})_{k^{(0)}}$:

- $\mathcal{G}en$. Unter Eingabe 1^n, gib $k \in \{0,1\}^n$ uniform aus.
- $\mathcal{E}nc$. Unter Eingabe eines Schlüssels k und der Nachricht m (mit $|k| = |m|$), gib

  $$c := f_k(m)$$

  aus.
- $\mathcal{D}ec$. Unter Eingabe eines Schlüssels k und eines Chiffriertexts c (mit $|k| = |c|$), gib

  $$m := f_k^{-1}(c)$$

  aus.

Ist dieses Verfahren stets IND-EAV-sicher?

Starke Pseudozufallspermutationen

Für den Begriff der starken Pseudozufallspermutationen wird dem Angreifer im Spiel auch die Möglichkeit gegeben Werte von, f_k^{-1} zu erhalten.

Spiel für starke Pseudozufallspermutationen

Eingabe: Sicherheitsparameter 1^n

1. Unterscheider und Herausforderer erhalten 1^n.
2. Der Herausforder wählt $i \in \{1, 2\}$ unform zufällig
   Es wird wie folgt eine Funktion $f : \{0,1\}^n \longrightarrow \{0,1\}^n$ definiert:
   - Wenn $i = 0$: $f : \{0,1\}^n \longrightarrow \{0,1\}^n$ wird als uniform zufällige Pseudozufallspermuation gewählt.
   - Wenn $i = 1$: k in $\{0,1\}^n$ wird uniform zufällig gewählt, $f := f_k$.

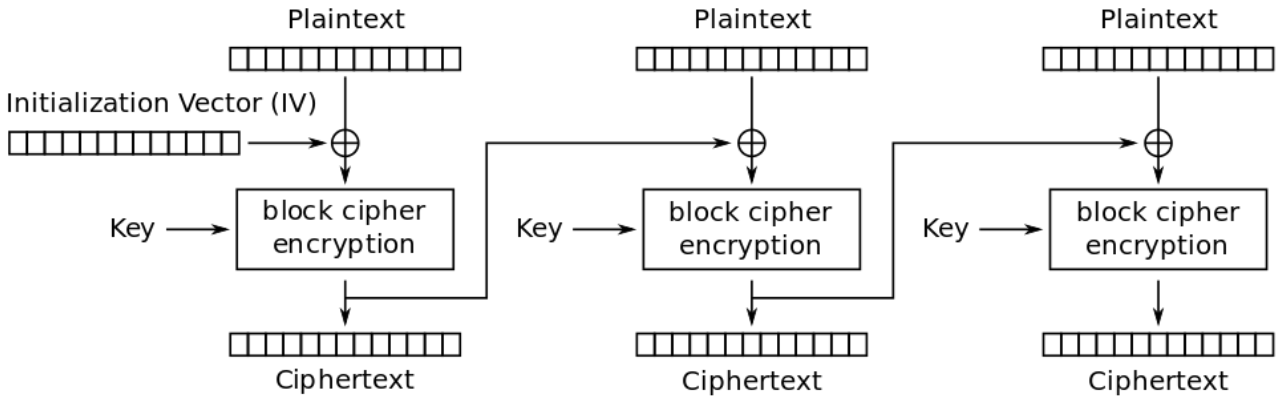
   Der Unterscheider kann für $x \in \{0,1\}^n$ den Wert $f(x)$ anfordern / erhalten und für $y \in \{0,1\}^n$ den Wert $f^{-1}(y)$ anfordern / erhalten.
3. Der Unterscheider entscheidet sich für $j \in \{0,1\}$.

Der Unterscheider hat gewonnen, wenn $i = j$ ist.

Der Erfolg / Vorteil von $\mathcal{D}$ ist ("wie immer")

$$2 \cdot (\mathbf{P}[\mathcal{D} \text{ gewinnt}] - \frac{1}{2}) .$$

Cipher Block Chaining Mode (CBC)



Cipher Block Chaining (CBC) mode encryption

Quelle. Wikipedia, Benutzer WhiteTimberwolf

Der CBC-Modus

Das Verfahren für einen Block:

$\mathcal{E}nc$. Eingabe: k, m

Wähle $r \in \{0,1\}^n$ uniform.

Ausgabe $c = (r, f_k(r \oplus m))$.

$\mathcal{D}ec$. Eingabe $c = (r, a)$

Ausgabe $f_k^{-1}(a) \oplus r = m$.

Satz. Wenn eine Pseudozufallspermutation gegeben ist, ist das Verfahren IND-CPA-sicher.

Blockchiffren

Resultate

Satz

Für eine Pseudozufallsfunktion sind der Output Feedback Mode (OFB-Mode) und der Couter Mode (CTR-Mode) IND-CPA-sicher.

Für eine Pseudozufallspermutation ist der Cipher Block Chaining Mode (CBC-Mode) IND-CPA-sicher.