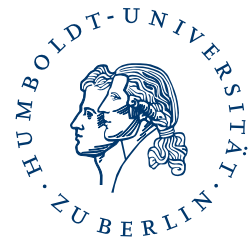
Einführung in die Kryptologie

Johannes Köbler



Institut für Informatik
Humboldt-Universität zu Berlin

SS 2022

Produktchiffren

- Produktchiffren erhält man durch die sequentielle Anwendung mehrerer Verschlüsselungsverfahren
- Sie können extrem schwer zu brechen sein, auch wenn die einzelnen Komponenten leicht zu brechen sind

Definition

- Seien $KS_i = (M_i, C_i, E_i, D_i, K_i, \mathcal{S}_i)$, $i \in \{1, 2\}$, Kryptosysteme mit $C_1 = M_2$
- Dann ist das Produktkryptosystem $KS_1 \times KS_2$ von KS_1 und KS_2 definiert als $(M_1, C_2, E, D, K_1 \times K_2, \mathcal{S})$ mit $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ und

$$E(k_1, k_2; x) = E_2(k_2, E_1(k_1, x)) \text{ sowie } D(k_1, k_2; y) = D_1(k_1, D_2(k_2, y))$$

für alle $x \in M_1$, $y \in C_2$ und $(k_1, k_2) \in K_1 \times K_2$

- Der Schlüsselraum von $KS_1 \times KS_2$ umfasst also alle Schlüsselpaare $(k_1,k_2) \in K_1 \times K_2$, wobei wir voraussetzen, dass die beiden Schlüssel unabhängig gewählt werden (d.h. es gilt $p(k_1, k_2) = p(k_1)p(k_2)$)

Produktchiffren

Beispiel

- Bilden wir das Produkt $KS = KS^* \times KS^+$ der
  - multiplikativen Chiffre $KS^* = (M, C, \mathbb{Z}_m^*, E^*, D^*)$ mit $M = C = \mathcal{A} = \{a_0, \ldots, a_{m-1}\}$ und der
  - additiven Chiffre $KS^+ = (M, C, \mathbb{Z}_m, E^+, D^+)$,

  so gilt für jeden Schlüssel $(k_1, k_2) \in K = \mathbb{Z}_m^* \times \mathbb{Z}_m$:

  $$E(k_1, k_2; x) = E^+(k_2, E^*(k_1, x)) = k_1 x + k_2$$

- Dies bedeutet, dass das Produkt $KS^* \times KS^+ = (M, C, K, E, D)$ der multiplikativen und additiven Chiffre die affine Chiffre ist
- Welche Chiffre erhalten wir, wenn wir das Produkt $KS^+ \times KS^*$ der additiven und der multiplikativen Chiffre bilden?

Beispiel (Fortsetzung)

- Das Produkt $KS^+ \times KS^*$ ist das Kryptosystem $KS' = (M, C, K', E', D')$, in dem für jeden Schlüssel $(k_2, k_1) \in K' = \mathbb{Z}_m \times \mathbb{Z}_m^*$ gilt:

$$E'(k_2, k_1; x) = k_1(x + k_2) = k_1 x + k_1 k_2 = E(k_1, k_1 k_2; x)$$

- Die Abbildung $(k_2, k_1) \mapsto (k_1, k_1 k_2)$ ist also eine Bijektion zwischen den Schlüsselräumen $\mathbb{Z}_m \times \mathbb{Z}_m^*$ und $\mathbb{Z}_m^* \times \mathbb{Z}_m$, die jeden Schlüssel $(k_2, k_1) \in \mathbb{Z}_m \times \mathbb{Z}_m^*$ auf einen Schlüssel $(k_1, k_1 k_2) \in \mathbb{Z}_m^* \times \mathbb{Z}_m$ mit $E_{(k_1, k_1 k_2)} = E'_{(k_2, k_1)}$ abbildet
- Somit können wir auch jeden Schlüsselgenerator $\mathcal{S}$ für $KS^* \times KS^+$ in einen Generator $\mathcal{S}'$ für $KS^+ \times KS^*$ transformieren (und $\mathcal{S}'$ auch wieder zurück in $\mathcal{S}$), so dass $\mathcal{S}'$ in $KS^+ \times KS^*$ jede Chiffrierfunktion mit der gleichen Wahrscheinlichkeit erzeugt wie $\mathcal{S}$ in $KS^* \times KS^+$
- Daher können die beiden Kryptosysteme $KS^* \times KS^+$ und $KS^+ \times KS^*$ als gleich (oder besser äquivalent, siehe Übungen) angesehen werden, d.h. die beiden Kryptosysteme KS^* und KS^+ kommutieren ◁

Produktchiffren

Definition

- Ein Kryptosystem $KS = (M, C, K, D, E)$ mit $M = C$ heißt endomorph
- Ein endomorphes Kryptosystem KS heißt idempotent, falls $KS \times KS$ äquivalent zu KS ist (in Zeichen: $KS \times KS = KS$)

Beispiel

- Eine leichte Rechnung zeigt, dass
  - die additive Chiffre,
  - die multiplikative Chiffre und
  - die affine Chiffre

  idempotent sind
- Dies trifft auch auf
  - die Blocktransposition sowie
  - die Vigenère- und Hill-Chiffre

  zu (siehe Übungen)

◁

Produktchiffren

- Will man durch mehrmalige Anwendung (Iteration) derselben Chiffre eine höhere Sicherheit erreichen, so darf diese nicht idempotent sein
- Man kann versuchen, durch sequentielle Ausführung zweier idempotenter Systeme KS_1 und KS_2 ein System $KS = KS_1 \times KS_2$ zu erhalten, das nicht idempotent ist
- Da KS im Fall $KS_1 \times KS_2 = KS_2 \times KS_1$ wegen

$$\begin{aligned} KS \times KS &= (KS_1 \times KS_2) \times (KS_1 \times KS_2) \\ &= KS_1 \times (KS_2 \times KS_1) \times KS_2 \\ &= KS_1 \times (KS_1 \times KS_2) \times KS_2 \\ &= (KS_1 \times KS_1) \times (KS_2 \times KS_2) \\ &= KS_1 \times KS_2 \\ &= KS \end{aligned}$$

idempotent ist, dürfen hierbei KS_1 und KS_2 jedoch nicht kommutieren

Iterierte Blockchiffren

- Im Folgenden betrachten wir Blockchiffren über dem Binäralphabet $A = \{0, 1\}$ und auch der Schlüsselraum wird von der Form $\{0, 1\}^k$ sein
- Die Schlüssellänge bezeichnen wir bis auf weiteres mit k und einzelne Schlüssel eines Kryptosystems mit K
- Eine iterierte Blockchiffre wird durch eine Rundenfunktion (round function) g und einen Key-Schedule Algorithmus f beschrieben
- Ist N die Rundenzahl, so erzeugt f bei Eingabe eines Schlüssels K eine Folge $f(K) = (K^1, \ldots, K^N)$ von N Rundenschlüsseln K^i für g

Iterierte Blockchiffren

- Mit diesen wird ein Klartext $x = w^0$ durch N-malige Anwendung der Rundenfunktion g zu einem Kryptotext $y = w^N$ verschlüsselt:

  $$w^1 := g(K^1, w^0)$$
  $$\vdots$$
  $$w^N := g(K^N, w^{N-1})$$

- Um y wieder zu entschlüsseln, muss die inverse Rundenfunktion g^{-1} mit umgekehrter Rundenschlüsselfolge $K^N, \ldots, K^1$ benutzt werden:

  $$w^{N-1} := g^{-1}(K^N, w^N)$$
  $$\vdots$$
  $$w^0 := g^{-1}(K^1, w^1)$$

- Beispiele für iterierte Chiffren sind der aus 16 Runden bestehende DES-Algorithmus und der AES mit einer variablen Rundenzahl $N \in \{10, 12, 14\}$, die wir später behandeln werden

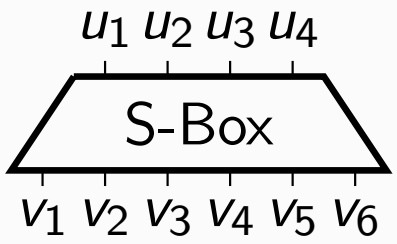
Substitutions-Permutations-Netzwerke

- Als Basisbausteine für die Rundenfunktion von iterierten Blockchiffren eignen sich Substitutionen und Transpositionen besonders gut
- Aus Effizienzgründen sollten die Substitutionen nur eine relativ kleine Blocklänge ℓ haben

Definition

- Für ein Wort $u = u_1 \cdots u_n \in \{0,1\}^n$ und Indizes $1 \leq i \leq j \leq n$ bezeichne $u[i,j]$ das Teilwort $u_i \cdots u_j$ von u
- Im Fall $n = ml$ bezeichnen wir das Teilwort $u[(i-1)l+1, il]$ auch einfach mit $u_{(i)}$, d.h. es gilt $u = u_{(1)} \cdots u_{(m)}$, wobei $|u_{(i)}| = l$ ist

Substitutions-Permutations-Netzwerke



- Sei $\sigma_S : \{0,1\}^l \to \{0,1\}^{l'}$ eine Substitution, die Binärblöcke u der Länge l in Blöcke $v = \sigma_S(u)$ der Länge l' überführt (auch S-Box S genannt)
- Für $\sigma_S(u)$ schreiben wir auch einfach $S(u)$
- Durch parallele Anwendung von m Kopien der S-Box S erhalten wir die Substitution $\sigma_{mS} : \{0,1\}^{ml} \to \{0,1\}^{ml'}$ mit

$$\sigma_{mS}(u_1 \cdots u_{ml}) = \sigma_S(u_{(1)}) \cdots \sigma_S(u_{(m)})$$

- Auch hier schreiben wir für $\sigma_{mS}(u_1 \cdots u_{ml})$ auch einfach $S(u_1 \cdots u_{ml})$
- Für die Speicherung einer S-Box S mit $\sigma_S : \{0,1\}^l \to \{0,1\}^{l'}$ auf einem Chip werden $l' \cdot 2^l$ Bit Speicherplatz benötigt (im Fall $l = l'$ also $l2^l$ Bit)
- Für $l = l' = 16$ wären dies beispielsweise 2^{20} Bit, was Smartcard-Anwendungen bereits ausschließen würde
- Für eine Transposition P auf $\{0,1\}^\ell$ bezeichnen wir die zugehörige Permutation auf $[\ell]$ mit π_P oder einfach mit π, falls P aus dem Kontext bekannt ist, d.h. $P(u_1 \cdots u_\ell) = u_{\pi(1)} \cdots u_{\pi(\ell)}$

Substitutions-Permutations-Netzwerke

Definition

- Für natürliche Zahlen $m, l \geq 1$ sei $M = C = \{0,1\}^\ell$ mit $\ell = ml$
- Ein Substitutions-Permutations-Netzwerk (SPN) wird durch eine S-Box S, eine Blocktransposition P mit Blocklänge $\ell = ml$ und durch eine Funktion $f : \{0,1\}^k \to \{0,1\}^{\ell(N+1)}$ beschrieben
- Hierbei realisiert die S-Box S eine Permutation σ_S auf $\{0,1\}^l$ und N ist die Rundenzahl des SPN
- Die Funktion f transformiert einen (externen) Schlüssel $K \in \{0,1\}^k$ in ein Key-Schedule $f(K) = (K^1, \ldots, K^{N+1})$ von $N+1$ Rundenschlüsseln
- Unter ihnen wird ein Klartext $x \in \{0,1\}^\ell$ durch folgenden Chiffrieralgorithmus in einen Kryptotext $y = E_{f,S,P}(K,x) \in \{0,1\}^\ell$ überführt:

1 $w^0 := x$
2 **for** $r := 1$ **to** $N-1$ **do**
3 $\quad u^r := w^{r-1} \oplus K^r$
4 $\quad v^r := \sigma_{mS}(u^r)$
5 $\quad w^r := P(v^r)$
6 $u^N := w^{N-1} \oplus K^N$
7 $v^N := \sigma_{mS}(u^N)$
8 $y := v^N \oplus K^{N+1}$

Substitutions-Permutations-Netzwerke

Die Chiffrierfunktion $E_{f,S,P}(K, x)$

- Zu Beginn jeder Runde $r \in \{1, \ldots, N\}$ wird w^{r-1} zuerst einer XOR-Operation mit dem Rundenschlüssel K^r unterworfen (round key mixing) und das Resultat u^r den S-Boxen zugeführt
- Auf die Ausgabe v^r der S-Boxen wird in jeder Runde $r \leq N-1$ die Transposition P angewendet, was die Eingabe w^r für die nächste Runde $r+1$ liefert

```
1  w^0 := x
2  for r := 1 to N − 1 do
3     u^r := w^{r−1} ⊕ K^r
4     v^r := S(u^r)
5     w^r := P(v^r)
6  u^N := w^{N−1} ⊕ K^N
7  v^N := S(u^N)
8  y := v^N ⊕ K^{N+1}
```

- Am Ende der letzten Runde $r = N$ wird nicht die Transposition P angewandt, sondern der Rundenschlüssel K^{N+1} auf v^N addiert
- Dies wird whitening genannt und bewirkt, dass auch für den letzten Chiffrierschritt der Schlüssel benötigt und somit der Gegner an einer partiellen Entschlüsselung des Kryptotexts gehindert wird
- Zudem wird dadurch eine (legale) Entschlüsselung nach fast demselben Verfahren ermöglicht (siehe Übungen)

Substitutions-Permutations-Netzwerke

- Die S-Boxen sorgen dafür, dass jedes einzelne Bit des Kryptotextes y von mehreren Bits des Klartextes und des Schlüssels abhängt
- Wichtig ist hierbei, dass die Abhängigkeit möglichst komplex ist (also z.B. nicht linear)
- Dadurch werden Angriffe erschwert, die versuchen, Rückschlüsse vom Kryptotext auf den Schlüssel oder Klartext zu ziehen
  (Shannon nannte diese Eigenschaft Konfusion)
- Die Transpositionen über den gesamten Block sorgen dafür, dass sich eine Änderung von einzelnen Bits des Klartextes oder des Schlüssels potentiell auf jedes Kryptotextbit auswirken kann
  (von Shannon als Diffusion bezeichnet)
- Idealerweise wird hierdurch erreicht, dass sich bei Änderung eines einzelnen Eingabe- oder Schlüsselbits jedes Ausgabebit mit Wahrscheinlichkeit $1/2$ ändert

Substitutions-Permutations-Netzwerke

Beispiel

- Wir betrachten ein SPN SP mit Parametern $l = m = N = 4$ und $k = 32$
- Für f wählen wir die Funktion $f(K) = (K^1, \ldots, K^5)$ mit $K^r = K[4(r-1)+1, 4(r-1)+16]$
- Weiter seien $\sigma_S : \{0,1\}^4 \to \{0,1\}^4$ und $\sigma_P : \{1,\ldots,16\} \to \{1,\ldots,16\}$ die folgenden Permutationen:

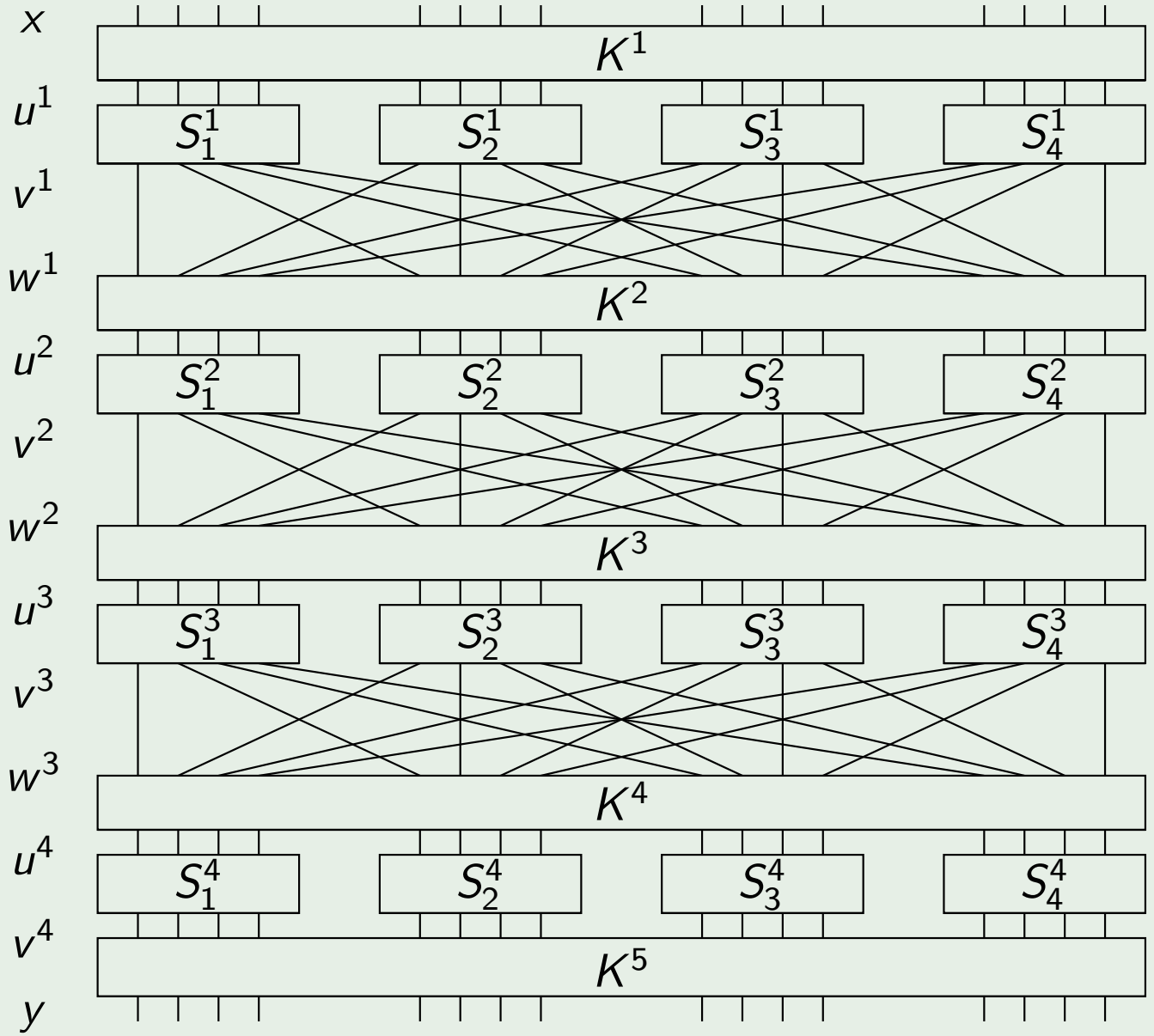
z	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$\sigma_S(z)$	E	4	D	1	2	F	B	8	3	A	6	C	5	9	0	7

wobei die Argumente und Werte von σ_S hexadezimal dargestellt sind, und

i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
$\sigma_P(i)$	1	5	9	13	2	6	10	14	3	7	11	15	4	8	12	16

Substitutions-Permutations-Netzwerke

Beispiel (Fortsetzung)



Substitutions-Permutations-Netzwerke

Beispiel (Schluss)

- Beispielsweise liefert f für den Schlüssel $K = 0011\,1010\,1001\,0100\,1101\,0110\,0011\,1111$ die Rundenschlüssel $f(K) = (K^1, \ldots, K^5)$ mit

$$K^1 = 0011\,1010\,1001\,0100 \qquad K^2 = 1010\,1001\,0100\,1101$$
$$K^3 = 1001\,0100\,1101\,0110 \qquad K^4 = 0100\,1101\,0110\,0011$$
$$K^5 = 1101\,0110\,0011\,1111$$

unter denen der Klartext $x = 0010\,0110\,1011\,0111$ die folgenden Chiffrierschritte durchläuft:

$$\begin{aligned}
x &= 0010\,0110\,1011\,0111 = w^0 \\
w^0 \oplus K^1 &= 0001\,1100\,0010\,0011 = u^1 \\
S(u^1) &= 0100\,0101\,1101\,0001 = v^1 \\
P(v^1) &= 0010\,1110\,0000\,0111 = w^1 \\
&\vdots \\
P(v^3) &= 1110\,0100\,0110\,1110 = w^3 \\
w^3 \oplus K^4 &= 1010\,1001\,0000\,1101 = u^4 \\
S(u^4) &= 0110\,1010\,1110\,1001 = v^4 \\
u^4 \oplus K^5 &= 1011\,1100\,1101\,0110 = y
\end{aligned}$$

◁

Lineare Approximationen

- Sei $f : \{0,1\}^l \to \{0,1\}^{l'}$ eine boolesche Funktion
- Wählen wir die Eingabe $\mathcal{U} = \mathcal{U}_1 \cdots \mathcal{U}_l$ zufällig unter Gleichverteilung, so gilt für die zugehörige Ausgabe $\mathcal{V} = f(\mathcal{U}) = \mathcal{V}_1 \cdots \mathcal{V}_{l'}$ sowie für alle $u \in \{0,1\}^l$ und $v \in \{0,1\}^{l'}$:

$$\Pr[\mathcal{V} = v \mid \mathcal{U} = u] = \begin{cases} 1 & f(u) = v \\ 0 & \text{sonst} \end{cases}$$

- Wegen $\Pr[\mathcal{U} = u] = 2^{-l}$ folgt

$$\Pr[\mathcal{V} = v, \mathcal{U} = u] = \begin{cases} 2^{-l} & f(u) = v, \\ 0 & \text{sonst} \end{cases}$$

- Die Funktion f ist linear, wenn eine binäre $(l \times l')$-Matrix A mit $f(u) = uA$ existiert
- In diesem Fall ist jedes Ausgabebit v_j in der Form $v_j = u_{i_1} \oplus \cdots \oplus u_{i_k}$ mit $1 \leq i_1 < \cdots < i_k \leq l$ darstellbar, d.h. $\Pr[\mathcal{V}_j = \mathcal{U}_{i_1} \oplus \cdots \oplus \mathcal{U}_{i_k}] = 1$

Lineare Approximationen

- Für eine lineare Kryptoanalyse benötigen wir Gleichungen der Form

$$\mathcal{V}_{j_1} \oplus \cdots \oplus \mathcal{V}_{j_{k'}} = \mathcal{U}_{i_1} \oplus \cdots \oplus \mathcal{U}_{i_k} \oplus c$$

  mit $1 \leq i_1 < \cdots < i_k \leq l$, $1 \leq j_1 < \cdots < j_{k'} \leq l'$ und $c \in \{0,1\}$, die mit möglichst großer Wahrscheinlichkeit gelten
- Definieren wir für $u, a \in \{0,1\}^l$ und $v, b \in \{0,1\}^{l'}$ die Teilsummen

$$u_a = \bigoplus_{i=1}^{l} a_i u_i \quad \text{und} \quad v_b = \bigoplus_{i=1}^{l'} b_i v_i,$$

  so suchen wir also nach Werten für a, b und c, für die das Ereignis $\mathcal{V}_b = \mathcal{U}_a \oplus c$ (oder $\mathcal{U}_a \oplus \mathcal{V}_b = c$) mit großer Wahrscheinlichkeit eintritt
- In diesem Fall lässt sich nämlich der Wert von $\mathcal{V}_b$ bei Kenntnis von $\mathcal{U}_a$ entsprechend gut vorhersagen
- Wegen $\Pr[\mathcal{U}_a \oplus \mathcal{V}_b = c] = 1 - \Pr[\mathcal{U}_a \oplus \mathcal{V}_b = c \oplus 1]$ kommt es nur darauf an, wie stark die Wahrscheinlichkeit $\Pr[\mathcal{U}_a \oplus \mathcal{V}_b = 0]$ von $1/2$ abweicht
- $\mathcal{V}_b$ lässt sich also in Abhängigkeit von $\mathcal{U}_a$ um so besser vorhersagen, je größer der Absolutbetrag $|\Pr[\mathcal{U}_a \oplus \mathcal{V}_b = 0] - 1/2|$ ist

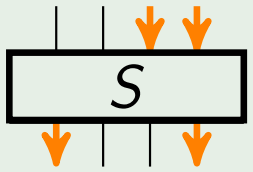
Güte einer linearen Approximation an eine S-Box

Definition

- Der Bias einer Zufallsvariablen $\mathcal{X}$ mit Wertebereich $W(\mathcal{X}) = \{0, 1\}$ ist definiert als $\varepsilon(\mathcal{X}) = \Pr[\mathcal{X} = 0] - 1/2$
- Sei $f : \{0,1\}^l \to \{0,1\}^{l'}$ eine boolesche Funktion
- Eine lineare Approximation an f wird durch ein Paar $(a, b) \in \{0,1\}^l \times \{0,1\}^{l'}$ beschrieben
- Die Güte der durch (a, b) beschriebenen Approximation an f ist der doppelte Absolutbetrag $\gamma_f(a, b) = 2|\varepsilon(\mathcal{U}_a \oplus \mathcal{V}_b)|$ des Bias-Wertes von $\mathcal{U}_a \oplus \mathcal{V}_b$, wobei $\mathcal{U}$ auf $\{0,1\}^l$ gleichverteilt und $\mathcal{V} = f(\mathcal{U})$ ist

Beispiel

- Wir betrachten die durch das Paar $(0011, 1001)$ beschriebene lineare Approximation $\mathcal{A} = \mathcal{U}_3 \oplus \mathcal{U}_4 \oplus \mathcal{V}_1 \oplus \mathcal{V}_4$ an die S-Box S aus vorigem Beispiel



- Dann nimmt die Zufallsvariable $(\mathcal{U}_1, \ldots, \mathcal{U}_4, \mathcal{V}_1, \ldots, \mathcal{V}_4, \mathcal{A})$ die 16 Werte in folgender Tabelle jeweils mit Wahrscheinlichkeit $2^{-4} = 1/16$ an:

u_1	u_2	u_3	u_4	v_1	v_2	v_3	v_4	$u_3 \oplus u_4 \oplus v_1 \oplus v_4$
0	0	0	0	1	1	1	0	1
0	0	0	1	0	1	0	0	1
0	0	1	0	1	1	0	1	1
0	0	1	1	0	0	0	1	1
0	1	0	0	0	0	1	0	0
0	1	0	1	1	1	1	1	1
0	1	1	0	1	0	1	1	1
0	1	1	1	1	0	0	0	1
1	0	0	0	0	0	1	1	1
1	0	0	1	1	0	1	0	0
1	0	1	0	0	1	1	0	1
1	0	1	1	1	1	0	0	1
1	1	0	0	0	1	0	1	1
1	1	0	1	1	0	0	1	1
1	1	1	0	0	0	0	0	1
1	1	1	1	0	1	1	1	1

Beispiel (Fortsetzung)

- Um $\varepsilon(\mathcal{U}_a \oplus \mathcal{V}_b)$ zu berechnen, genügt es, die Anzahl $L(a,b)$ der Eingaben $u \in \{0,1\}^l$ zu bestimmen, für die $f(u)_b = u_a$ ist
- Dann gilt $\Pr[\mathcal{U}_a \oplus \mathcal{V}_b = 0] = \Pr[\mathcal{U}_a = \mathcal{V}_b] = L(a,b)/16$ und somit

$$\varepsilon(\mathcal{U}_a \oplus \mathcal{V}_b) = L(a,b)/16 - 1/2 = (L(a,b) - 8)/16$$

sowie

$$\gamma(a,b) = |2\varepsilon(\mathcal{U}_a \oplus \mathcal{V}_b)|$$

- Für $a = 0011$ und $b = 1001$ gibt es z.B. $L(0011, 1001) = 2$ Eingaben u mit $u_{0011} = S(u)_{1001}$ ($u = 0100$ und $u = 1001$), d.h. es gilt

$$\varepsilon(\mathcal{U}_{0011} \oplus \mathcal{V}_{1001}) = (L(3,9) - 8)/16 = (2 - 8)/16 = -3/8$$

- Die Güte von $\mathcal{A}$ ist also $\gamma(0011, 1001) = |2\varepsilon(\mathcal{U}_{0011} \oplus \mathcal{V}_{1001})| = 3/4$
- Die Tabelle auf der nächsten Folie zeigt die Anzahlen $L(a,b)$ für alle Werte von a und b (in Hexadezimaldarstellung)

Lineare Approximationen

Beispiel (Fortsetzung)

	b															
a	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	16	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8
1	8	8	6	6	8	8	6	14	10	10	8	8	10	10	8	8
2	8	8	6	6	8	8	6	6	8	8	10	10	8	8	2	10
3	8	8	8	8	8	8	8	8	10	2	6	6	10	10	6	6
4	8	10	8	6	6	4	6	8	8	6	8	10	10	4	10	8
								⋮								
B	8	12	8	4	12	8	12	8	8	8	8	8	8	8	8	8
								⋮								
F	8	6	4	6	6	8	10	8	8	6	12	6	6	8	10	8

Unser nächstes Ziel ist, geeignete lineare Approximationen für bestimmte S-Boxen im SPN zu finden, die sich zu einer linearen Approximation der Abbildung $x \mapsto u^4$ zusammenschalten lassen (siehe nächste Folie)

Substitutions-Permutations-Netzwerke

Beispiel (Schluss)