Sicherheit von Π_{MAC2}

Satz Sicherheit von Π_{MAC2}

Sei Π' sicher. Dann ist Π_{MAC2} ebenfalls sicher.

Beweis:

- Sei $\mathcal{A}$ ein Angreifer für Π_{MAC2} mit Erfolgsws $\epsilon(n)$.
- Wir konstruieren einen Angreifer $\mathcal{A}'$ für Π'.

Algorithmus Angreifer $\mathcal{A}'$

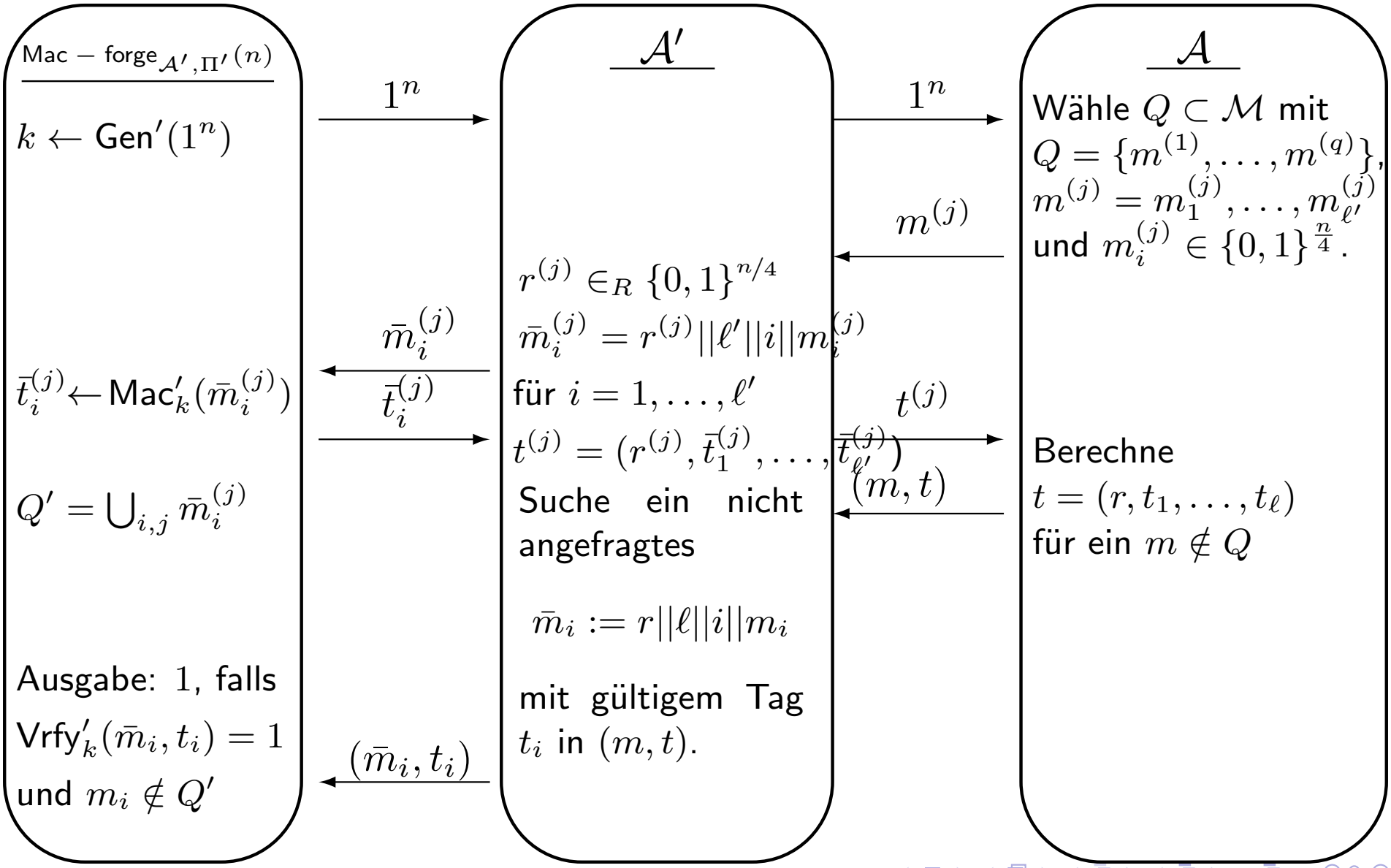
EINGABE: 1^n, Orakel $Mac'_k(\cdot)$.

1. Beantworte $Mac_k(m_1^{(j)} \dots m_{\ell'}^{(j)})$-Anfragen von $\mathcal{A}$ wie folgt: Wähle $r^{(j)} \in_R \{0,1\}^{\frac{n}{4}}$ und berechne $\bar{t}_i^{(j)} = Mac'_k(r^{(j)}||\ell'||i||m_i^{(j)})$ für $i = 1, \dots, \ell'$.
2. $(m,t) = (m_1 \dots m_\ell, r, t_1 \dots t_\ell) \leftarrow \mathcal{A}^{Mac_k(\cdot)}(1^n)$.

AUSGABE: Nicht-angefragtes $\bar{m}_i = r||\ell||i||m_i$ mit gültigem Tag t_i, falls ein

solches in $(\bar{m}_i, t_i)$ existiert.

Sicherheit von Π_{MAC2}



Sicherheit von Π_{MAC2}

Wir definieren die folgenden Ereignisse

- *Forge*: Ein Block $\bar{m}_i = r||\ell||i||m_i$ in (m,t) wurde nicht an $\text{Mac}'_k(\cdot)$ angefragt, aber $\text{Vrfy}'_k(\bar{m}_i, t_i) = 1$.
- *Repeat*: Bei 2 MAC-Anfragen wird dasselbe $r^{(i)} = r^{(j)}$ verwendet.

Aufgrund der Sicherheit von Π' gilt

$$
\begin{aligned}
\text{negl}(n) &\geq \text{Ws}[\text{Mac-forge}_{\mathcal{A}',\Pi'}(n) = 1] \\
&= \text{Ws}[\text{Mac-forge}_{\mathcal{A},\Pi_{MAC2}}(n) = 1 \wedge \textit{Forge}] \\
&= \epsilon(n) - \text{Ws}[\text{Mac-forge}_{\mathcal{A},\Pi_{MAC2}}(n) = 1 \wedge \overline{\textit{Forge}}] \\
&= \epsilon(n) - \underbrace{\text{Ws}[\text{Mac-forge}_{\mathcal{A},\Pi_{MAC2}}(n) = 1 \wedge \overline{\textit{Forge}} \wedge \overline{\textit{Repeat}}]}_{\epsilon_1} \\
&\quad - \underbrace{\text{Ws}[\text{Mac-forge}_{\mathcal{A},\Pi_{MAC2}}(n) = 1 \wedge \overline{\textit{Forge}} \wedge \textit{Repeat}]}_{\leq \text{Ws}[\textit{Repeat}]}
\end{aligned}
$$

Zeigen nun $\epsilon_1 = 0$ und $\text{Ws}[\textit{Repeat}] \leq \text{negl}(n)$. Damit $\epsilon(n) \leq \text{negl}(n)$.

Sicherheit von Π_{MAC2}

zu zeigen: $\mathrm{Ws}[\mathit{Repeat}] \leq \mathrm{negl}(n)$

- Sei $q(n)$ die Anzahl der MAC-Anfragen von $\mathcal{A}$ an $\mathcal{A}'$.
- Bei der i-ten MAC-Anfrage wähle $\mathcal{A}'$ den Identifikator r_i.
- *Repeat* tritt ein, falls $r^{(i)} = r^{(j)}$ für ein $i \neq j$. Sei dies Ereignis $E_{i,j}$.
- Nach Geburtstagsparadoxon gilt:

$$\begin{aligned} \mathrm{Ws}[\mathit{Repeat}] = \mathrm{Ws}\Big[\bigcup_{1\leq i<j\leq q(n)} E_{i,j}\Big] &\leq \sum_{1\leq i<j\leq q(n)} \mathrm{Ws}[E_{i,j}] \\ &\leq \frac{q(n)^2}{2^{\frac{n}{4}}} = \mathrm{negl}(n). \end{aligned}$$

Sicherheit von Π_{MAC2}

zu zeigen: $\mathrm{Ws}[\text{Mac-forge}_{\mathcal{A},\Pi_{MAC2}}(n) = 1 \wedge \overline{Forge} \wedge \overline{Repeat}] = 0$

- **Idee:** $\text{Mac-forge}_{\mathcal{A},\Pi_{MAC2}}(n) = 1$ und $\overline{Repeat}$ impliziert *Forge*.
- Sei $(m, t) = (m_1 \ldots m_\ell, r, t_1 \ldots t_\ell)$ die Ausgabe von $\mathcal{A}$.

Fall 1: Identifikator r unterscheidet sich von allen $r^{(j)}$.

- Dann ist (z.B.) $r||\ell||1||m_1$ nicht-angefragt mit gültigem Tag t_1.

Fall 2: $r = r^{(j)}$ für genau ein $j \in [q(n)]$.

- Sei $m^{(j)} = m_1^{(j)} \ldots m_{\ell'}^{(j)}$ die von $\mathcal{A}$ angefragte Nachricht.
- Fall $\ell \neq \ell'$: Dann ist $r||\ell||1||m_1$ nicht-angefragt mit gültigem Tag t_1.
- Fall $\ell = \ell'$: Wegen $m \notin Q$ gilt $m \neq m^{(j)}$.
- D.h. es existiert ein i, so dass $m_i \neq m_i^{(j)}$.
- Damit wurde $r||\ell||i||m_i$ nicht angefragt. Tag t_i ist dafür gültig.

Fall 3: $r = r^{(j)}$ für mehrere $j \in [q(n)]$.

- Fall ist ausgeschlossen, da wegen $\overline{Repeat}$ gilt $r^{(i)} \neq r^{(j)}$ für alle $i \neq j$.

Sicherer MAC für Nachrichten beliebiger Länge

Korollar Sicherer MAC für Nachrichten beliebiger Länge

Sei F eine Pseudozufallsfunktion. Dann ist Π_{MAC2} für $\Pi' = \Pi_{MAC}$ sicher.

Nachteile:

- Für $m \in (\{0,1\}^{\frac{n}{4}})^{\ell}$ sind ℓ Anwendungen von F_k erforderlich.
- Der MAC-Tag $t = (r, t_1 \ldots t_\ell)$ besitzt Länge $(\ell + 1)n$ Bits.

CBC-MAC für Nachrichten fester Länger

Algorithmus CBC-MAC für Nachrichten fester Länger

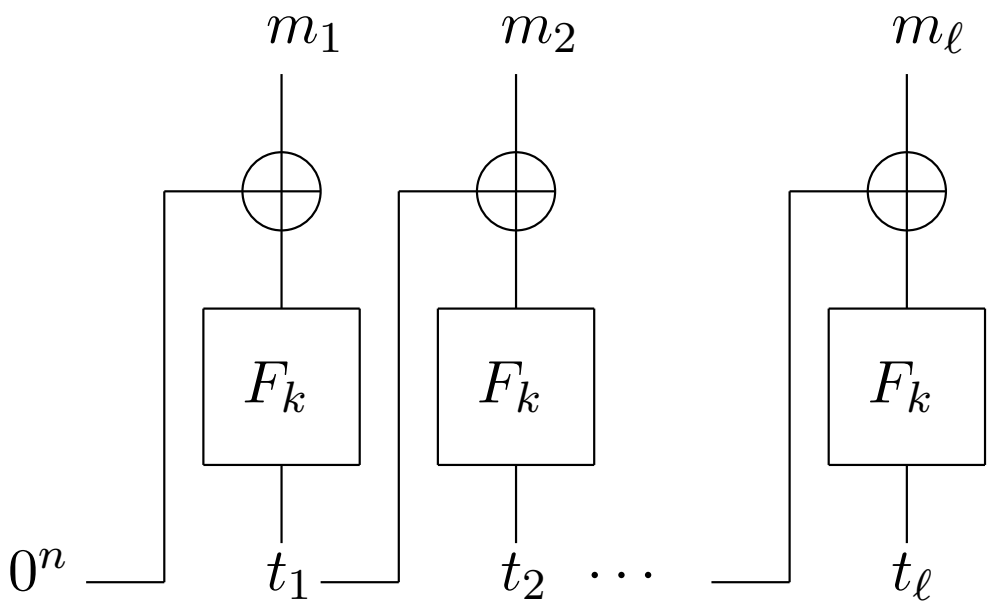
Sei F eine Pseudozufallsfunktion und $m \in (\{0,1\}^n)^\ell$ für festes ℓ.

1. **Gen:** Wähle $k \in_R \{0,1\}^n$.
2. **Mac:** Für $k \in \{0,1\}^n$ und $m = m_1 \dots m_\ell \in (\{0,1\}^n)^\ell$ setze $t_0 = 0^n$,
$$t_i := F_k(t_{i-1} \oplus m_i) \text{ für } i = 1, \dots, \ell.$$
Ausgabe des Tags $t := t_\ell$.
3. **Vrfy:** Für $k \in \{0,1\}^n$ und $(m,t) \in (\{0,1\}^n)^\ell \times \{0,1\}^n$,
$$\text{Ausgabe} = \begin{cases} 1 & \text{falls } Mac_k(m) = t \\ 0 & \text{sonst} \end{cases}.$$

Anmerkungen:

- Für den CBC-MAC kann Sicherheit für festes ℓ gezeigt werden.
- Tags besitzen nur Länge n, unabhängig von ℓ.
- Konstruktion ist *unsicher* für variables ℓ (Übung).

CBC Mac



Vergleich mit CBC Modus bei Verschlüsselung

Rolle des Initialisierungsvektors

- Benötigen zur Sicherheit beim CBC Modus $IV \in_R \{0,1\}^n$.
- Der CBC-MAC verwendet einen festen Wert $IV = t_0 = 0^n$.
- Verwendet man ein zufälliges $t_0 \in_R \{0,1\}^n$ und gibt (t_0, t_ℓ) als Tag aus, so ist dies eine unsichere MAC-Konstruktion! (Übung)

Ausgabe

- CBC Modus: Ausgabe aller c_i, um entschlüsseln zu können.
- Beim CBC-MAC wird nur t_ℓ ausgegeben. Die Werte $t_1, \ldots, t_{\ell-1}$ kann der Verifizierer selbst bestimmen, die MAC-Länge ist nur n.
- Werden $t_1, \ldots, t_{\ell-1}$ ausgegeben, so ist der MAC unsicher! (Übung)

Man beachte:
Scheinbar harmlose Änderungen können drastische Effekte haben.

Sichere CBC-MACs für Nachrichten variabler Länge

Erzeugen längenabhängiger Schlüssel:

- Berechne $k_\ell := F_k(\ell)$ als Schlüssel für Nachrichten der Länge ℓ.
- Verwende k_ℓ als Schlüssel in CBC, d.h. $t_i := F_{k_\ell}(t_{i-1} \oplus m_i)$.
- Wir erhalten einen eigenen Schlüssel k_ℓ für jede feste Länge ℓ.

Anhängen der Länge:

- Verwende $t_0 := F_k(\ell)$ als Initialisierungsvektor.
- Äquivalent zum Berechnen des Tags von $\ell || m_1 \dots m_\ell$ mit $t_0 = 0^n$.
- Man beachte: Berechnen des Tags $m_1 \dots m_\ell || \ell$ ist unsicher.

Verwenden zweier Schlüssel:

- Wähle $k_1, k_2 \in_R \{0,1\}^n$. Berechne den Tag $t = F_{k_2}(Mac_{k_1}(m))$.
- Alternativ: Wähle $k_1 := F_k(1)$ und $k_2 := F_k(2)$.
- Vorteil: Mac-Berechnung möglich, ohne ℓ a priori zu kennen.