Authentisierte Verschlüsselung

Ziel: *Vertraulichkeit* und *Authentizität* der Nachricht

Spiel Authentizität $Auth_{\mathcal{A},\Pi}(n)$

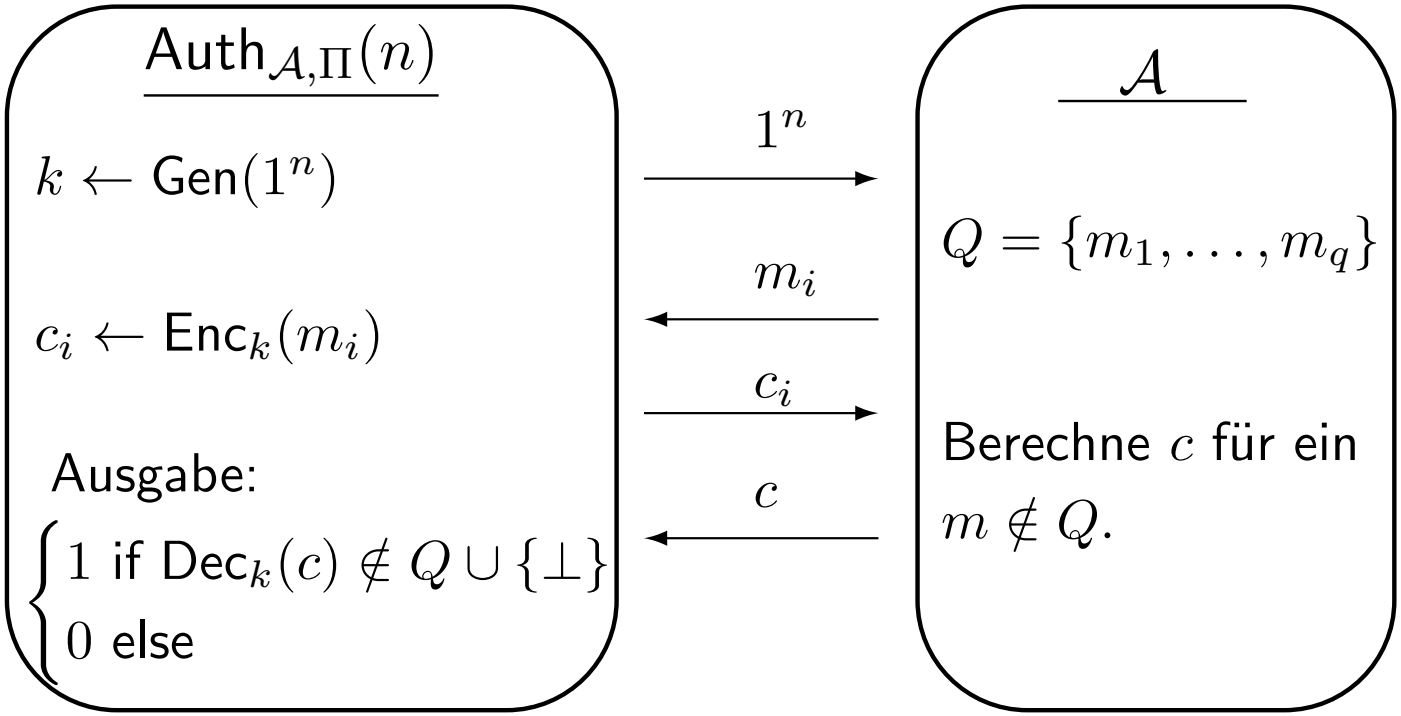
Sei $\mathcal{A}$ ein Angreifer gegen $\Pi = (Gen, Enc, Dec)$.

1. $k \leftarrow Gen(1^n)$
2. $c \leftarrow \mathcal{A}^{Enc_k(\cdot)}(1^n)$, d.h. $\mathcal{A}$ darf $Enc_k(m)$ für beliebige m anfragen.
3. Sei Q die Menge der $Enc_k(\cdot)$-Anfragen und $m := Dec_k(c)$.

$$Auth_{\mathcal{A},\Pi}(n) = \begin{cases} 1 & \text{falls } m \neq \perp \text{ und } m \notin Q \\ 0 & \text{sonst} \end{cases}.$$

Stärkere Definition: Sei Q_c die Menge der Antworten auf $Enc_k(\cdot)$ Anfragen. Definiere $SAuth_{\mathcal{A},\Pi}(n) = 1$ gdw. $m \neq \perp$ und $c \notin Q_c$. ($SAuth_{\mathcal{A},\Pi}$ Spiel einfacher für $\mathcal{A}$ zu gewinnen als $Auth_{\mathcal{A},\Pi}$.)

Sicherheitsspiel Authentizität



Authentisierte Sicherheit

Definition Authentizität

Ein Verschlüsselungsverfahrens Π liefert *Authentizität* falls für alle ppt Angreifer $\mathcal{A}$ gilt

$$\mathrm{Ws}[\mathit{Auth}_{\mathcal{A},\Pi}(n) = 1] \leq \mathrm{negl}(n).$$

Achtung: CCA-Sicherheit alleine impliziert nicht Authentizität.

Definition Sicherheit eines Verschlüsselungsverfahrens

Ein Verschlüsselungsverfahren Π liefert authentisierte Sicherheit (ist AE-sicher), falls es CCA-sicher ist und Authentizität liefert.

Auf Englisch: authenticated encryption (AE).

Alternative Sicherheitsdefinition

Definition Authentisierte Sicherheit

Verschlüsselungsverfahren $\Pi = (Gen, Enc, Dec)$ ist AE'-sicher, falls für alle PPT Angreifer $\mathcal{A}$ gilt:

$$|\Pr[\mathcal{A}^{Enc_k(\cdot),Dec_k(\cdot)}(1^n) = 1] - Pr[\mathcal{A}^{Enc_k^0(\cdot),Dec_k^0(\cdot)}(1^n) = 1]| = negl(n),$$

mit Wahrscheinlichkeitsraum: $k \leftarrow Gen(1^n)$ und zufällige Münzwürfe von $\mathcal{A}$. Die Orakel sind definiert als

- $Enc_k^0(m) := Enc_k(0)$
- $Dec_k^0(c) := \begin{cases} m & \text{falls } c \text{ Ausgabe von Anfrage } Enc_k^0(m) \text{ war.} \\ \perp & \text{sonst} \end{cases}$.

Man kann zeigen, dass AE' Sicherheit äquivalent zur starken AE Sicherheit ist.

Sicherheit von Encrypt-then-authenticate Π_{cca}

Satz Sicherheit von Π_{cca}

Sei $\Pi' = (Gen', Enc', Dec')$ CPA-sicher und $\Pi_M = (Gen_M, Tag, Vfy)$ ein sicherer MAC mit eindeutigen Tags. Dann ist $\Pi_{cca} = (Gen, Enc, Dec)$ AE-sicher.

Beweis:

- Die CCA-Sicherheit von Π_{cca} wurde bereits gezeigt.
- Sei $\mathcal{A}$ ein Angreifer im Spiel $Auth_{\mathcal{A},\Pi_{cca}}(n)$ mit Erfolgsws $\epsilon(n)$.
- Wir konstruieren daraus einen Angreifer $\mathcal{A}'$ für Π_M.

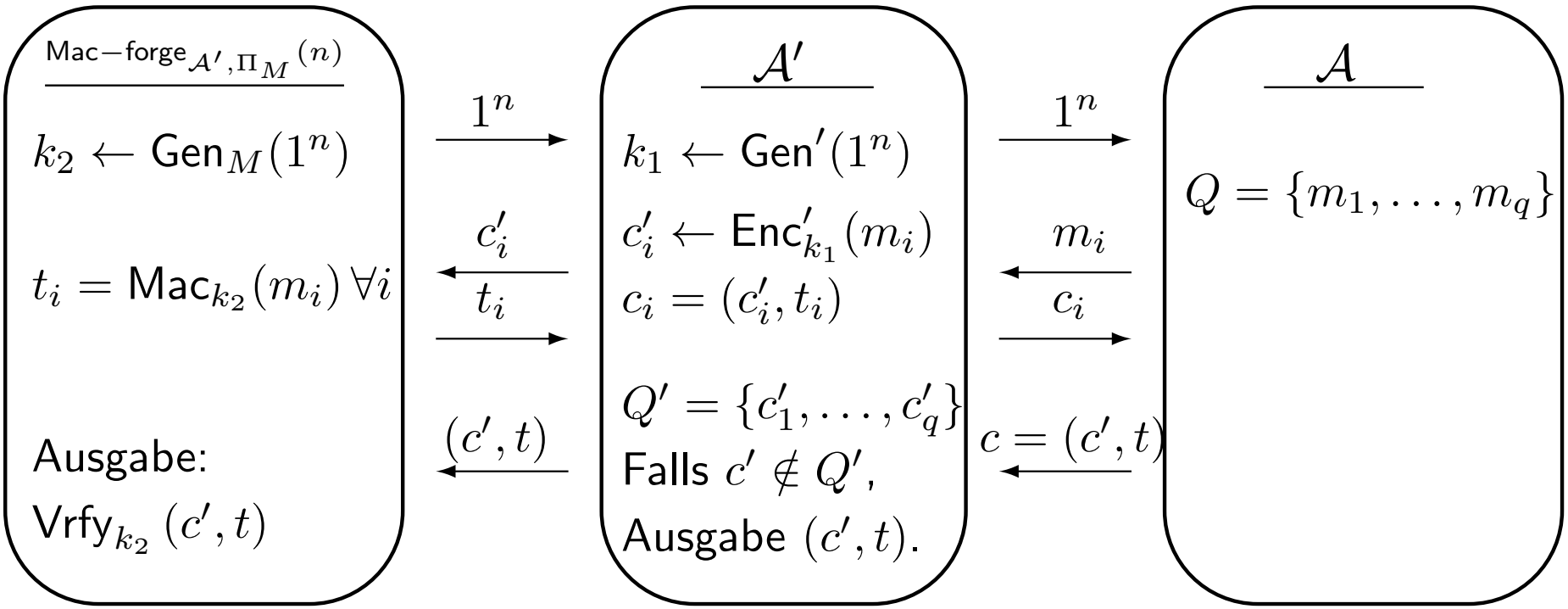
Algorithmus Angreifer $\mathcal{A}'$ für Π_M

EINGABE: 1^n, Orakelzugriff auf $Mac_{k_2}(\cdot)$

1. $k_1 \leftarrow Gen'(1^n)$
2. $c = (c', t') \leftarrow \mathcal{A}^{Enc_{k_1,k_2}(\cdot)}(1^n)$, bei $Enc_{k_1,k_2}(m)$-Anfrage berechne $c' \leftarrow Enc'_{k_1}(m)$, $t = Mac_{k_2}(c')$ und antworte mit $c = (c', t)$.

AUSGABE: $c = (c', t')$

Sicherheit von Encrypt-then-authenticate Π_{cca}



Sicherheit von Encrypt-then-authenticate Π_{cca}

Beweis: Fortsetzung

- Sei $Q = \{m_1, \ldots, m_q\}$ die Menge der $Enc_{k_1,k_2}(\cdot)$-Anfragen.
- Seien c'_i die Verschlüsselungen von m_i für $i = 1, \ldots, q$ in Schritt 2.
- Falls $\mathcal{A}$ Erfolg hat, so gilt $m := Dec_{k_1,k_2}(c) \notin Q$.
- Daraus folgt $c' \notin \{c'_1, \ldots, c'_q\} = Q'$. D.h. t' ist ein Tag für eine nicht an das $Mac_{k_2}(\cdot)$-Orakel angefragte Nachricht c.
- Es folgt

$$\mathrm{Ws}[\text{Mac-forge}_{\mathcal{A}',\Pi_M}(n) = 1] \geq \mathrm{Ws}[Auth_{\mathcal{A},\Pi_{cca}}(n) = 1] = \epsilon(n).$$

- Aus der Sicherheit von Π_M folgt $\epsilon(n) \leq \mathrm{negl}(n)$.

Anmerkung:

- Π_{cca} ist AE-sicher für *jedwede* sichere Instantiierung von Π_E und Π_M.

Die Notwendigkeit zweier Schlüssel k_1, k_2

Faustregel Verwendung verschiedener Schlüssel

Verschiedene Sicherheitsziele sollten durch Wahl verschiedener Schlüssel realisiert werden.

Bsp: Unsicheres Encrypt-then-authenticate durch einen Schlüssel

- Wir verwenden denselben Schlüssel k für Π' und Π_M.
- Sei F eine starke Pseudozufallspermutation auf n Bits.
- D.h. F^{-1} ist ebenfalls eine starke Pseudozufallspermutation.
- Wir konstruieren ein CPA-sicheres Π' mittels
$$Enc'_k(m) = F_k(m||r) \text{ für } m \in \{0,1\}^{\frac{n}{2}}, r \in_R \{0,1\}^{\frac{n}{2}}.$$
- Wir konstruieren einen sicheren MAC Π_M mittels
$$Mac_k(c') = F_k^{-1}(c') \text{ für } c' \in \{0,1\}^n.$$
- Encrypt-then-authenticate liefert
$$c = (c', t) = (F_k(m||r), F_k^{-1}(F_k(m||r))) = (c', m||r).$$
- D.h. c gibt die Nachricht m preis.

Encrypt-and-authenticate kann unsicher sein

Encrypt-and-authenticate:

$$c = Enc_{k_1,k_2}(m) := (c', t) = (Enc'_{k_1}(m), Mac_{k_2}(m)).$$

- D.h. der Tag wird für die Nachricht m berechnet, nicht für c'.
- Sei Π' CPA-sicher und $\Pi_M = (Gen_M, Mac, Vrfy)$ ein sicherer Mac.
- Dann ist Π'_M mit $Mac'_{k_2}(m) = (m, Mac_{k_2}(m))$ ebenfalls sicher, denn gültige Tags (m, t) für Π'_M liefern gültige Tags t für Π_M.
- Instantiierung von Encrypt-and-authenticate mit Π', Π'_M liefert
$$c = (c', (m, Mac_{k_2}(m))).$$
- D.h. c gibt die Nachricht m preis, obwohl Π' und Π'_M sicher sind.

Authenticate-then-encrypt kann unsicher sein

Authenticate-then-encrypt: $c = Enc(m) := Enc'_{k_1}(m||Mac_{k_2}(m))$

- Kodieren Nachricht $m \in \{0,1\}^*$ vor Verschlüsselung:
  Jede 0 wird als 00 oder 11, jede 1 als 01 oder 10.
- Dekodierung in Zweierblöcken: 00 oder 11 zu 0, 01 oder 10 zu 1.
- Wir verschlüsseln $Enc'_k(m) = CMEnc(\text{Kodierung}(m))$, wobei $CMEnc$ die CPA-sichere Verschlüsselung im Counter-Modus ist:
  $CMenc(m_1 \dots m_\ell) = (ctr, m_1 \oplus r_1, \dots, m_\ell \oplus r_\ell)$ mit $r_i = F_k(ctr + i \bmod 2^n)$.

Algorithmus CCA-Angreifer

EINGABE:
$c = Enc_{k_1,k_2}(m) = CMEnc_{k_1}(\text{Kodierung}(m||Mac_{k_2}(m))) = (ctr, c_1, c_2)$,
$Dec_{k_1,k_2}(\cdot)$-Orakel

Für alle Zweierblöcke in c, die eine Kodierung von m enthalten:

1. Berechne $\hat{c} = (ctr, c_1 \oplus 1^n, c_2)$ durch Flippen der Bits in in c_1.
2. Frage Dec_{k_1,k_2} Orakel nach $m = Dec_{k_1,k_2}(\hat{c})$.

AUSGABE: m

Authenticate-then-encrypt kann unsicher sein

- Sei $c = (ctr, c_1, c_2)$ mit $c_1 = Kodierung(m) \oplus r_1$, $c_2 = Kodierung(Mac_{k_2}(m)) \oplus r_2$
- Dann ist $\hat{c} = (ctr, Kodierung(m) \oplus r_1 \oplus 1^n, c_2) = (ctr, Kodierung(m) \oplus r_1, c_2)$
- Flippen der Bits wechselt zwischen den zwei Kodierungen von m.
- $t = Mac_{k_2}(m)$ bleibt gültig, da nur die Kodierung von m geändert wird, nicht m selbst. Mac_{k_2} wird nicht auf $\mathrm{Kodierung}(m)$ angewendet.

Anmerkungen:

- Bsp. zeigt, dass Authenticate-then-encrypt i. allg. nicht sicher ist.
- Das SSL (Secure Sockets Layer) Protokoll im Internet verwendet eine sichere Variante von Authenticate-then-encrypt.