Übung zur Vorlesung „Sicherheit“
Übung 4

Thomas Agrikola
Thomas.Agrikola@kit.edu

22.06.2017

Sicherheit – Praktische Aufgabe/Wettbewerb

- Wir werden demnächst eine praktische Aufgabe veröffentlichen, in der es darum geht ein aktuell verwendetes Krypto-Verfahren (mit ungünstig gewählten Parametern) zu brechen.
  - für Details siehe Aufgabenblatt
- Die erste korrekte Lösung gewinnt einen kleinen Preis.

Sicherheit – Übungsblatt 4 – Aufgabe 1

ElGamal-Verschlüsselungsverfahren

- Gruppe $\mathbb{Z}_{59}^{\times}$
- Erzeuger $g := 27$ erzeugt Untergruppe $\mathbb{G}$ der Quadrate, nicht ganz $\mathbb{Z}_{59}^{\times}$!

- Gruppenoperation: Multiplikation modulo 59
- $\mathbb{Z}_{59}^{\times}$ ist Einheitengruppe von $\mathbb{Z}_{59}$
  - d.h. alle invertierbaren Elemente
  - hier: $\mathbb{Z}_{59}^{\times} = \mathbb{Z}_{59} \setminus \{0\}$, da 59 prim
- $\mathbb{G} := \{x^2 \bmod 59 \colon x \in \mathbb{Z}_{59}^{\times}\}$ ist UG der Quadrate und hat Ordnung 29

Sicherheit – Übungsblatt 4 – Aufgabe 1

Tricks zum Rechnen:

(i) Finden von handhabbaren Repräsentanten, z.B.:

$$\begin{aligned} 51^2 \bmod 59 &= (59-8)^2 \bmod 59 \\ &= (-8)^2 \bmod 59 \\ &= 64 \bmod 59 \\ &= 5 \bmod 59 \end{aligned}$$

Sicherheit – Übungsblatt 4 – Aufgabe 1

Tricks zum Rechnen:

(ii) Reduzierung des Exponenten modulo der Gruppenordnung, z.B.:

$$\begin{aligned} 3^{60} \bmod 59 &= 3^{60 \bmod 58} \bmod 59 \\ &= 3^2 \bmod 59 \\ &= 9 \bmod 59 \end{aligned}$$

- Hintergrund: Potenzieren mit Gruppenordnung ergibt das neutrale Element.
- Ordnung von $\mathbb{Z}_{59}^{\times}$ ist $\varphi(59) = 58$ (da 59 prim).

Sicherheit – Übungsblatt 4 – Aufgabe 1 (a)

Berechnen Sie zum geheimen Schlüssel $sk := 20$ den öffentlichen Schlüssel pk.

$$
\begin{aligned}
pk &= g^{sk} \bmod N \\
&= 27^{20} \bmod 59 \\
&= (3^3)^{20} \bmod 59 \\
&= 3^{60 \bmod 58} \bmod 59 \\
&= 3^2 \bmod 59 \\
&= 9 \bmod 59
\end{aligned}
$$

Sicherheit – Übungsblatt 4 – Aufgabe 1 (b)

Verschlüsseln Sie $m := 22$ unter dem in (a) berechneten Schlüssel (mit dem Zufall $r := 12$).

- Erinnerung: $\mathrm{Enc}(pk, M) = (g^r, pk^r \cdot M)$

$$
\begin{aligned}
C_1 &= g^r \bmod N &&= 27^{12} \bmod 59 \\
&= 3^{36} \bmod 59 &&\overset{(*)}{=} 3^{29+7 \bmod 29} \bmod 59 \\
&= 3^7 \bmod 59 &&= 3^4 \cdot 3^3 \bmod 59 \\
&= 81 \cdot 3^3 \bmod 59 &&= 22 \cdot 3^3 \bmod 59 \\
&= 66 \cdot 3^2 \bmod 59 &&= 7 \cdot 9 \bmod 59 \\
&= 63 \bmod 59 &&= 4 \bmod 59
\end{aligned}
$$

$(*)$: 3 ist ein Quadrat, denn: $11^2 \bmod 59 = 3 \bmod 59$. Also liegt 3 in $\mathbb{G}$ und $|\mathbb{G}| = 29$.

Sicherheit – Übungsblatt 4 – Aufgabe 1 (b)

Verschlüsseln Sie $m := 22$ unter dem in (a) berechneten Schlüssel (mit dem Zufall $r := 12$).

$$
\begin{aligned}
C_2 &= pk^r \cdot m \bmod N \\
&= 9^{12} \cdot 22 \bmod 59 \\
&= 9^{12} \cdot (59 + 22) \bmod 59 \\
&= 3^{24} \cdot 81 \bmod 59 \\
&= 3^{24} \cdot 3^4 \bmod 59 \\
&= 3^{28} \bmod 59 \\
&= 3^{29-1 \bmod 29} \bmod 59 \quad \text{(3 ist Quadrat!)} \\
&= 3^{-1} \bmod 59 \\
&= 20 \bmod 59 \quad \text{(denn } 3 \cdot 20 \bmod 59 = 1\text{)}
\end{aligned}
$$

Sicherheit – Übungsblatt 4 – Aufgabe 1 (b)

Verschlüsseln Sie $m := 22$ unter dem in (a) berechneten Schlüssel (mit dem Zufall $r := 12$).

Insgesamt ergibt sich:

$$C = (C_1, C_2) = (4, 20)$$

Sicherheit – Übungsblatt 4 – Aufgabe 1 (c)

Entschlüsseln Sie $C := (51, 8)$ unter dem Schlüssel aus (a).

- Erinnerung: $\mathrm{Dec}(sk, (C_1, C_2)) = C_2 \cdot C_1^{-sk}$

Wir berechnen zuerst:

$$\begin{aligned} C_1^{sk} &= 51^{20} \bmod 59 \\ &= (-8)^{20} \bmod 59 \\ &= \left((-2)^3\right)^{20} \bmod 59 \\ &= (-2)^{60} \bmod 59 \\ &= (-2)^2 \bmod 59 \\ &= 4 \bmod 59 \end{aligned}$$

Invertieren ergibt:

$$C_1^{-sk} = 4^{-1} \bmod 59 = 15 \bmod 59$$

(Berechnung durch erw. euklid. Alg. oder „Scharfes Hinsehen“)

Sicherheit – Übungsblatt 4 – Aufgabe 1 (c)

Entschlüsseln Sie $C := (51, 8)$ unter dem Schlüssel aus (a).

$$\begin{aligned} C_2 \cdot C_1^{-sk} &= 8 \cdot 15 \bmod 59 \\ &= 120 \bmod 59 \\ &= 2 \bmod 59 \end{aligned}$$

Sicherheit – Übungsblatt 4 – Aufgabe 1

Fazit:

- ElGamal ist praxistauglich
- (und IND-CPA-sicher unter der Diffie-Hellman Annahme)
- Wichtiger, grundlegender Baustein für komplexere Krypto

Sicherheit – Übungsblatt 4 – Aufgabe 2

Frage: Warum eigentlich UG der Quadrate und nicht ganz $\mathbb{Z}_p^\times$?

Für eine Gruppe $\mathbb{G}$ sei $Q(\mathbb{G}) := \{x^2 : x \in \mathbb{G}\}$.

- $Q(\mathbb{G})$ ist eine Untergruppe von $\mathbb{G}$.
- In $\mathbb{Z}_p^\times$ kann effizient überprüft werden, ob ein Element y in $Q(\mathbb{Z}_p^\times)$ liegt oder nicht:

$$y^{\frac{p-1}{2}} = 1 \pmod{p} \Leftrightarrow y \in Q(\mathbb{Z}_p^\times)$$

Frage: Wie kann man diese Eigenschaften von $\mathbb{Z}_p^\times$ ausnutzen, um die IND-CPA-Sicherheit von ElGamal anzugreifen?

Sicherheit – Übungsblatt 4 – Aufgabe 2

Beobachtung:

- Für zwei Elemente $x, y \in Q(\mathbb{Z}_p^\times)$ ist auch $x \cdot y \in Q(\mathbb{Z}_p^\times)$ (denn $Q(\mathbb{Z}_p^\times)$ ist eine Gruppe).
- Ist aber $x \in Q(\mathbb{Z}_p^\times)$ und $y \in \mathbb{Z}_p^\times \setminus Q(\mathbb{Z}_p^\times)$, dann ist

$$x \cdot y \in \mathbb{Z}_p^\times \setminus Q(\mathbb{Z}_p^\times),$$

denn:

- Da $x \in Q(\mathbb{Z}_p^\times)$, existiert ein $\overline{x} \in \mathbb{Z}_p^\times$, so dass $\overline{x}^2 = x \bmod p$.
- Angenommen $x \cdot y \in Q(\mathbb{Z}_p^\times)$, dann existiert ein $\overline{z} \in \mathbb{Z}_p^\times$, so dass $\overline{z}^2 = x \cdot y \bmod p$.
- Dann gilt aber

$$\begin{aligned} & \overline{z}^2 = x \cdot y \bmod p \\ \iff & \overline{z}^2 \cdot x^{-1} = y \bmod p \\ \iff & \left(\overline{z} \cdot \overline{x}^{-1}\right)^2 = y \bmod p, \ \ldots \end{aligned}$$

$\ldots$ was ein Widerspruch zu $y \notin Q(\mathbb{Z}_p^\times)$ ist.

Sicherheit – Übungsblatt 4 – Aufgabe 2

Idee: Ein Angreifer $\mathcal{A}$ kann einem Chiffrat "ansehen", ob ein Quadrat oder ein Nicht-Quadrat verschlüsselt wurde.

- Ein ElGamal-Chiffrat hat folgende Form: $(a, b = a^{sk} \cdot m)$ für eine Nachricht $m \in \mathbb{Z}_p^\times$.
- Wenn wir also wissen, dass $a^{sk} \in Q(\mathbb{Z}_p^\times)$ ist, dann ist $b \in Q(\mathbb{Z}_p^\times) \Leftrightarrow m \in Q(\mathbb{Z}_p^\times)$.
- Ob $a^{sk} \in Q(\mathbb{Z}_p^\times)$ ist, können wir so herausfinden:
  - Es gilt: $a^{sk} \in Q(\mathbb{Z}_p^\times)$, falls $a \in Q(\mathbb{Z}_p^\times)$ oder falls $g^{sk} \in Q(\mathbb{Z}_p^\times)$ (denn g ist ein Erzeuger von $\mathbb{Z}_p^\times$ und damit ein Nicht-Quadrat; wenn g^{sk} also ein Quadrat ist, ist auch a^{sk} ein Quadrat).

Sicherheit – Übungsblatt 4 – Aufgabe 2

Der IND-CPA Angreifer $\mathcal{A}$ funktioniert dann wie folgt:

- Zunächst gibt $\mathcal{A}$ als Challenge ein Nicht-Quadrat m_0 und ein Quadrat m_1 aus.
- Als Antwort erhält $\mathcal{A}$ ein Chiffrat $c^* = (a, b)$.
- $\mathcal{A}$ prüft, ob $a^{sk} \in Q(\mathbb{Z}_p^\times)$ ist.
  - Falls nicht, rät $\mathcal{A}$ ein Bit b' zufällig und gibt b' aus.
  - Sonst gibt $\mathcal{A}$ das Bit $b' := \begin{cases} 1 & \text{falls } b \in Q(\mathbb{Z}_p^\times) \\ 0 & \text{sonst} \end{cases}$ aus.

Sicherheit – Übungsblatt 4 – Aufgabe 2

Analyse:

- In $\mathbb{Z}_p^\times$ sind genau die Hälfte der Elemente Quadrate.
- D.h. mit Wkt. $\frac{1}{2}$ ist das zufällige Element ein Quadrat und mit Wkt. $\frac{1}{2}$ ist g^{sk} ein Quadrat.

$$
\begin{aligned}
&\Pr[\mathcal{A} \text{ rät richtig}] \\
=&\Pr[\mathcal{A} \text{ rät richtig} \wedge a^{sk} \in Q(\mathbb{Z}_p^\times)] + \Pr[\mathcal{A} \text{ rät richtig} \wedge a^{sk} \notin Q(\mathbb{Z}_p^\times)] \\
=&1 \cdot \underbrace{\Pr[a^{sk} \in Q(\mathbb{Z}_p^\times)]}_{=\frac{3}{4}} + \frac{1}{2} \cdot \underbrace{\Pr[a^{sk} \notin Q(\mathbb{Z}_p^\times)]}_{=\frac{1}{4}} = \frac{7}{8}.
\end{aligned}
$$

- Damit ist $\mathcal{A}$ um $\frac{3}{8}$ (also nicht-vernachlässigbar) besser als raten.

$\Longrightarrow$ **ElGamal in $\mathbb{Z}_p^\times$ unsicher!**

Sicherheit – Übungsblatt 4 – Aufgabe 2

Exkurs: Wieso sind in $\mathbb{Z}_p^\times$ genau die Hälfte der Elemente Quadrate?

- Der Erzeuger g kann kein Quadrat sein, sonst existiert ein $x \in \mathbb{Z}_p^\times$ mit $x^2 = g$ und damit ist $g^{\frac{p-1}{2}} = (x^2)^{\frac{p-1}{2}} = x^{p-1} = 1 \pmod p$, was ein Widerspruch dazu ist, dass g ein Erzeuger ist.
- die Quadrate in $\mathbb{Z}_p^\times$ sind also genau $g^2, g^4, g^6, \ldots, g^{p-1}$
- und die Nicht-Quadrate in $\mathbb{Z}_p^\times$ sind $g^1, g^3, g^5, \ldots, g^{p-2}$
- (die unteren Elemente sind Nicht-Quadrate, da das Produkt von Quadraten und Nicht-Quadraten Nicht-Quadrate sind, schreibe dazu z.B. $g^3 = g^2 \cdot g$)

Sicherheit – DLog-Algorithmen

- Es ist wichtig beurteilen zu können, in welchen Gruppen das DLog-Problem schwer ist.
- Wir lernen nun Algorithmen kennen, mit denen diskrete Logarithmen bei kleiner Gruppenordnung oder bei ungünstiger Wahl von Parametern mit vertretbarem Aufwand gefunden werden können.

Eingabe: g, h, Gesucht: α, sodass $h = g^{\alpha}$.

Brute force:
Für alle $i \in \mathbb{Z}_{|\mathbb{G}|}$ vergleiche g^i mit h.
Zeitkomplexität: $O(|\mathbb{G}|)$

Sicherheit – Baby-Step-Giant-Step-Algorithmus

Eingabe: g, h, Gesucht: α, sodass $h = g^{\alpha}$.

Baby-Step-Giant-Step-Algorithmus:

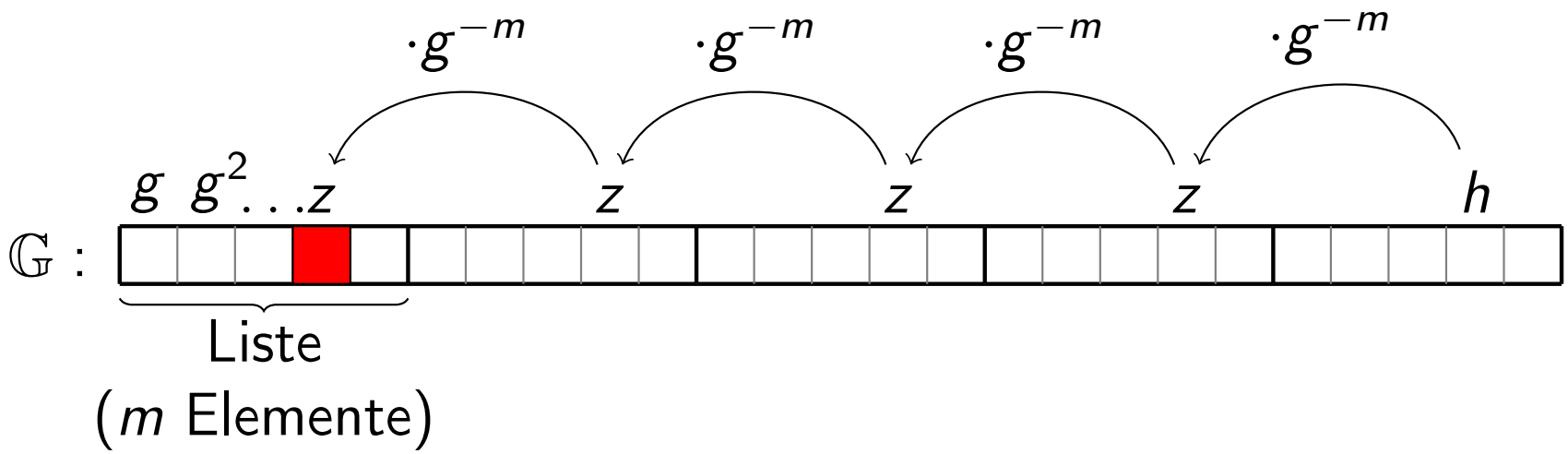
- Sei $m := \sqrt{|\mathbb{G}|}$.
- Berechne Liste mit Einträgen (i, g^i) für $0 \leq i \leq m$.
- Sortiere Liste nach zweiter Komponente.
  $\Rightarrow$ binäre Suche möglich
- Berechne (für $0 \leq j \leq m$) $z := g^{-j \cdot m} \cdot h$ und benutze binäre Suche, um z in der Liste zu finden.
  - Falls z in Liste enthalten: Sei (i, z) der zugehörige Eintrag.
    $\Rightarrow g^i = z = g^{-j \cdot m} \cdot h = g^{-j \cdot m + \alpha}$
    $\Rightarrow \alpha = i + j \cdot m \bmod |\mathbb{G}|$

Zeitkomplexität: $O(m \cdot \log(m))$, Speicherkomplexität: $O(m)$

Sicherheit – Baby-Step-Giant-Step-Algorithmus

Eingabe: g, h, Gesucht: α, sodass $h = g^{\alpha}$.

Baby-Step-Giant-Step-Algorithmus:



(m Elemente)

"Giant-Steps":

berechne $z := g^{-j \cdot m} \cdot h$ und suche nach z in Liste

- Falls z in Liste enthalten
  $\Rightarrow g^i = z = g^{-j \cdot m} \cdot h = g^{-j \cdot m + \alpha}$
  $\Rightarrow \alpha = i + j \cdot m \bmod |\mathbb{G}|$

Sicherheit – Pohlig-Hellman-Algorithmus

Eingabe: g, h, Gesucht: α, sodass $h = g^\alpha$.
Gruppenordnung: $|\mathbb{G}| = p_1^{e_1} \cdots p_r^{e_r}$ zerfällt.

Beobachtungen:

- Chinesischer Restsatz: um $\alpha \pmod{|\mathbb{G}|}$ zu finden, reicht es $\alpha_i := \alpha \pmod{p_i^{e_i}}$ zu berechnen.
- α_i kann (eindeutig) dargestellt werden als

$$\alpha_i = l_0 + l_1 p_i + l_2 p_i^2 + \cdots + l_{e_i-1} p_i^{e_i-1}$$

  mit $0 \leq l_0, \ldots, l_{e_i-1} < p_i$.
- Es reicht also jeweils diese l_j zu finden.

Sicherheit – Pohlig-Hellman-Algorithmus

Erinnerung: $\alpha = \alpha_i = l_0 + l_1 p_i + l_2 p_i^2 + \cdots + l_{e_i-1} p_i^{e_i-1} \bmod p_i^{e_i}$

Berechne l_0:

- $l_0 = \alpha \pmod{p_i}$

$$g^\alpha = h \Rightarrow (g^\alpha)^{\frac{|\mathbb{G}|}{p_i}} = h^{\frac{|\mathbb{G}|}{p_i}} \Leftrightarrow \underbrace{\left(g^{\frac{|\mathbb{G}|}{p_i}}\right)}_{:=g_0}{}^{\alpha} = \underbrace{h^{\frac{|\mathbb{G}|}{p_i}}}_{:=h_0}$$

- Wichtig: g_0 erzeugt eine Gruppe der Ordnung p_i
  $\Rightarrow$ Brute-force/Baby-Step-Giant-Step angewandt auf diese Gleichung liefert $\alpha \pmod{p_i} = l_0$.

Sicherheit – Pohlig-Hellman-Algorithmus

Erinnerung: $\alpha = \alpha_i = l_0 + l_1 p_i + l_2 p_i^2 + \cdots + l_{e_i-1} p_i^{e_i-1} \bmod p_i^{e_i}$

Seien $l_0, \ldots, l_{k-1}$ bekannt, berechne l_k:

$$g^\alpha = h \Rightarrow (g^\alpha)^{\frac{|\mathbb{G}|}{p_i^{k+1}}} = h^{\frac{|\mathbb{G}|}{p_i^{k+1}}} \Leftrightarrow \left(g^{\frac{|\mathbb{G}|}{p_i^{k+1}}}\right)^{\alpha \bmod p_i^{k+1}} = h^{\frac{|\mathbb{G}|}{p_i^{k+1}}}$$

$$\Leftrightarrow \left(g^{\frac{|\mathbb{G}|}{p_i^{k+1}}}\right)^{\sum_{j=0}^{k-1} l_j p_i^j + l_k p_i^k \bmod p_i^{k+1}} = h^{\frac{|\mathbb{G}|}{p_i^{k+1}}}$$

$$\Leftrightarrow \underbrace{\left(g^{\frac{|\mathbb{G}|}{p_i^{k+1}}}\right)^{l_k p_i^k}}_{=\left(g^{\frac{|\mathbb{G}|}{p_i}}\right)^{l_k}} = h^{\frac{|\mathbb{G}|}{p_i^{k+1}}} \cdot \left(g^{\frac{|\mathbb{G}|}{p_i^{k+1}}}\right)^{-\sum_{j=0}^{k-1} l_j p_i^j}$$

$\Rightarrow$ Brute-force/Baby-Step-Giant-Step angewandt auf diese Gleichung liefert l_k.

Sicherheit – Rechenbeispiel

Rechenbeispiel: $\mathbb{Z}_{181}^{\times}$, $g := 2$ ist ein Erzeuger.
Gegeben $2^{\alpha} = 92 \bmod 181$.
$|\mathbb{G}| = 180 = 2^2 \cdot 3^2 \cdot 5$

Berechne $\alpha \bmod 2^2 = l_0 + l_1 \cdot 2 \bmod 4$**:**

$$\left(2^{\frac{|\mathbb{G}|}{2}}\right)^{l_0} = 92^{\frac{|\mathbb{G}|}{2}} \bmod 181$$

$$\Leftrightarrow \left(2^{90}\right)^{l_0} = 92^{90} \bmod 181$$

$$\Leftrightarrow 180^{l_0} = 180 \bmod 181$$

$$\Rightarrow l_0 = 1 \bmod 2$$

$$\left(2^{\frac{|\mathbb{G}|}{2}}\right)^{l_1} = (92)^{\frac{|\mathbb{G}|}{2^2}} \left(2^{\frac{|\mathbb{G}|}{2^2}}\right)^{-\sum_{j=0}^{0} l_j 2^j} \bmod 181$$

$$\Leftrightarrow 180^{l_1} = 19 \cdot (162)^{-1} \bmod 181$$

$$\Leftrightarrow 180^{l_1} = 19 \cdot 19 = 180 \bmod 181$$

$$\Rightarrow l_1 = 1 \bmod 2$$

$$\Rightarrow \alpha = 3 \bmod 2^2$$

Sicherheit – Rechenbeispiel

Rechenbeispiel: $\mathbb{Z}_{181}^{\times}$, $g := 2$ ist ein Erzeuger.
Gegeben $2^{\alpha} = 92 \bmod 181$.
$|\mathbb{G}| = 180 = 2^2 \cdot 3^2 \cdot 5$

Berechne $\alpha \bmod 3^2 = l_0 + l_1 \cdot 3 \bmod 9$**:**

$$\left(2^{\frac{|\mathbb{G}|}{3}}\right)^{l_0} = 92^{\frac{|\mathbb{G}|}{3}} \bmod 181$$

$$\Leftrightarrow \left(2^{60}\right)^{l_0} = 92^{60} \bmod 181$$

$$\Leftrightarrow 48^{l_0} = 48 \bmod 181$$

$$\Rightarrow l_0 = 1 \bmod 3$$

$$\left(2^{\frac{|\mathbb{G}|}{3}}\right)^{l_1} = (92)^{\frac{|\mathbb{G}|}{3^2}} \left(2^{\frac{|\mathbb{G}|}{3^2}}\right)^{-\sum_{j=0}^{0} l_j 2^j} \bmod 181$$

$$\Leftrightarrow 48^{l_1} = 43 \cdot (43)^{-1} \bmod 181$$

$$\Leftrightarrow 48^{l_1} = 1 \bmod 181$$

$$\Rightarrow l_1 = 0 \bmod 3$$

$$\Rightarrow \alpha = 1 \bmod 3^2$$

Sicherheit – Rechenbeispiel

Rechenbeispiel: $\mathbb{Z}_{181}^{\times}$, $g := 2$ ist ein Erzeuger.
Gegeben $2^{\alpha} = 92 \bmod 181$.
$|\mathbb{G}| = 180 = 2^2 \cdot 3^2 \cdot 5$

Berechne $\alpha \bmod 5 = l_0 \bmod 5$:

$$\left(2^{\frac{|\mathbb{G}|}{5}}\right)^{l_0} = 92^{\frac{|\mathbb{G}|}{5}} \bmod 181$$

$$\Leftrightarrow \left(2^{36}\right)^{l_0} = 92^{36} \bmod 181$$

$$\Leftrightarrow 59^{l_0} = 1 \bmod 181$$

$$\Rightarrow l_0 = 0 \bmod 5$$

$\Rightarrow \alpha = 0 \bmod 5$

Sicherheit – Rechenbeispiel

Rechenbeispiel: $\mathbb{Z}_{181}^{\times}$, $g := 2$ ist ein Erzeuger.
Gegeben $2^{\alpha} = 92 \bmod 181$.
$|\mathbb{G}| = 180 = 2^2 \cdot 3^2 \cdot 5$

Chinesischer Restsatz:

$$
\begin{aligned}
\alpha =& 3 \cdot 9 \cdot (9^{-1} \bmod 4) \cdot 5 \cdot (5^{-1} \bmod 4) \\
&+ 1 \cdot 4 \cdot (4^{-1} \bmod 9) \cdot 5 \cdot (5^{-1} \bmod 9) \\
&+ 0 \cdot 4 \cdot (4^{-1} \bmod 5) \cdot 9 \cdot (9^{-1} \bmod 5) \bmod 180 \\
=& 135 + 100 + 0 \bmod 180 \\
=& 55 \bmod 180
\end{aligned}
$$

Sicherheit – Pohlig-Hellman-Algorithmus

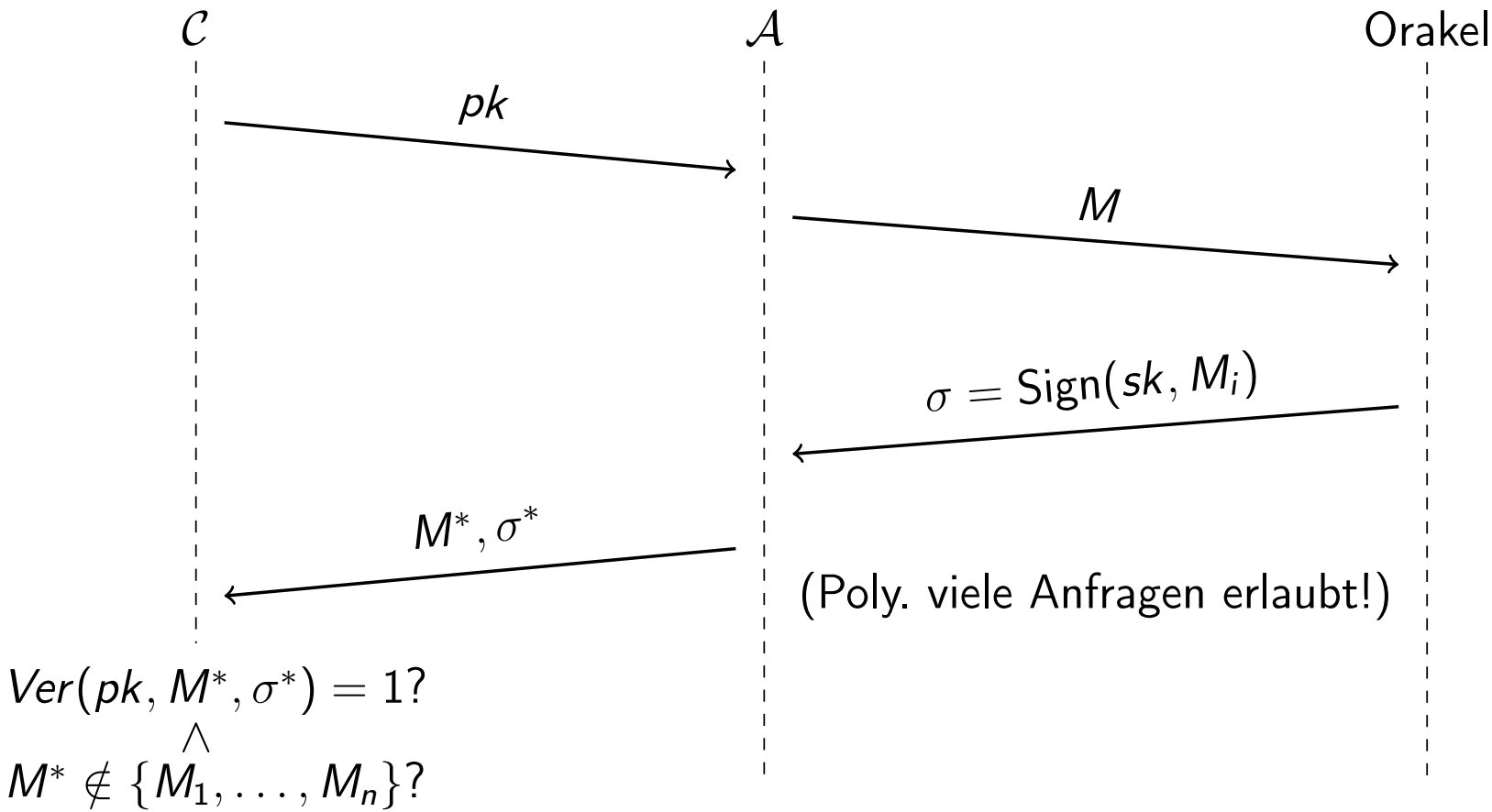
Zeitkomplexität: $O(\sum_{i=1}^{r} e_i \cdot (\log |\mathbb{G}| + \sqrt{p_i}))$.
$\Rightarrow$ zerfällt $|\mathbb{G}|$ in kleine Primfaktoren, ist der Algorithmus **sehr praktikabel...**

Fazit:

- $\mathbb{Z}_p^\times$ eignet sich im Allgemeinen also nicht, denn die Gruppenordnung $p-1$ zerfällt bei ungünstiger Wahl von p in kleine Faktoren.
- Wähle für p deshalb sogenannte *safe primes*.
  - Das sind Primzahlen, die die Form $p = 2q + 1$ haben, für eine Primzahl q.
  - Die Gruppenordnung von $\mathbb{Z}_p^\times$ ist dann $p - 1 = 2q$.
  - Verwende als Gruppe dann die Untergruppe der Quadrate, diese hat die prime Ordnung q.

Wdh.: EUF-CMA

- Herausforderer $\mathcal{C}$ führt $(pk, sk) \leftarrow \mathsf{Gen}(1^k)$ aus.
- $\mathcal{C}$ stellt $\mathsf{Sign}(sk, \cdot)$-Orakel für $\mathcal{A}$ bereit.



Sicherheit – Übungsblatt 4 – Aufgabe 3

Es sei $\Sigma = (\mathrm{Gen}, \mathrm{Sign}, \mathrm{Ver})$ ein EUF-CMA-sicheres Signaturverfahren.

Wir konstruieren daraus zwei neue Verfahren.

Sicherheit – Übungsblatt 4 – Aufgabe 3 (1) (a)

Betrachte Σ' mit:

- $\mathsf{Gen}'(1^k) := \mathsf{Gen}(1^k)$,
- $\mathsf{Sign}'(sk, M) := (\mathsf{Sign}(sk, M \oplus r), r) = (\sigma_1, \sigma_2)$ $(r \leftarrow \{0,1\}^{|M|})$,
- $\mathsf{Ver}'(pk, M, \sigma) := \mathsf{Ver}(pk, M \oplus \sigma_2, \sigma_1)$.

(a) Korrektheit:

$$\begin{aligned}\mathsf{Ver}'(pk, M, \sigma) &= \mathsf{Ver}(pk, M \oplus r, \sigma_1) \\ &= \mathsf{Ver}(pk, M \oplus r, \mathsf{Sign}(sk, M \oplus r)) \\ &= 1. \quad \checkmark\end{aligned}$$

(folgt aus Korrektheit von Σ)

Sicherheit – Übungsblatt 4 – Aufgabe 3 (1) (b)

Betrachte Σ' mit:

- $\mathsf{Gen}'(1^k) := \mathsf{Gen}(1^k)$,
- $\mathsf{Sign}'(sk, M) := (\mathsf{Sign}(sk, M \oplus r), r) = (\sigma_1, \sigma_2)$ $(r \leftarrow \{0,1\}^{|M|})$,
- $\mathsf{Ver}'(pk, M, \sigma) := \mathsf{Ver}(pk, M \oplus \sigma_2, \sigma_1)$.

(b) Zeigen Sie: Σ' ist nicht EUF-CMA-sicher.

- $\mathcal{A}$ schickt bel. M_1 und erhält Signatur $\sigma_1 = (\sigma_1', r_1)$.
- ... wählt bel. $M^* \neq M_1$.
- ... setzt $r^* = M^* \oplus M_1 \oplus r_1$.
- ... setzt $\sigma^* = (\sigma_1', r^*)$.

Sicherheit – Übungsblatt 4 – Aufgabe 3 (1) (b)

$\sigma^* = (\sigma_1', r^*) = (\sigma_1, M^* \oplus M_1 \oplus r_1)$ ist gültige Fälschung für M^*, denn:

$$\begin{aligned}
\mathsf{Ver}'(pk, M^*, \sigma^*) &= \mathsf{Ver}'(pk, M^*, (\sigma_1', r^*)) \\
&= \mathsf{Ver}(pk, M^* \oplus r^*, \sigma_1') \\
&= \mathsf{Ver}(pk, M^* \oplus (M^* \oplus M_1 \oplus r_1), \sigma_1') \\
&= \mathsf{Ver}(pk, M_1 \oplus r_1, \sigma_1') \\
&= \mathsf{Ver}'(pk, M_1, (\sigma_1', r_1)) \\
&= \mathsf{Ver}'(pk, M_1, \sigma_1) \\
&= 1.
\end{aligned}$$

$\mathcal{A}$ hat polynomielle Laufzeit, Erfolgswkt. ist 1
$\Rightarrow \Sigma'$ nicht EUF-CMA-sicher.

Sicherheit – Übungsblatt 4 – Aufgabe 3 (1)

Anmerkung:

- Auch deterministische Signaturen können EUF-CMA-sicher sein!
- Indeterminismus also nicht „notwendig“...
- ... anders als bei Verschlüsselungsverfahren!

- Nebenbemerkung: Determinismus bzw. Eindeutigkeit von Signaturen kann sogar vorteilhaft sein
  - Sicherheitsbegriff **s**EUF-CMA „automatisch“ erfüllt, wenn EUF-CMA-sicher

Sicherheit – Übungsblatt 4 – Aufgabe 3 (2) (a)

Betrachte Σ^* mit:

- $\mathsf{Gen}^*(1^k) := \mathsf{Gen}(1^k)$,
- $\mathsf{Sign}^*(sk, M) := \mathsf{Sign}(sk, \overline{M})$ ($\overline{M}$ bitw. Inverse von M),
- $\mathsf{Ver}^*(pk, M, \sigma) = \mathsf{Ver}(pk, \overline{M}, \sigma)$

(a) Korrektheit:

$$\begin{aligned}\mathsf{Ver}^*(pk, M, \sigma) &= \mathsf{Ver}(pk, \overline{M}, \sigma) \\ &= \mathsf{Ver}(pk, \overline{M}, \mathsf{Sign}(sk, \overline{M})) \\ &= 1. \quad \checkmark\end{aligned}$$

(folgt aus Korrektheit von Σ)

Sicherheit – Übungsblatt 4 – Aufgabe 3 (2) (b)

Betrachte Σ^* mit:

- $\mathsf{Gen}^*(1^k) := \mathsf{Gen}(1^k)$,
- $\mathsf{Sign}^*(sk, M) := \mathsf{Sign}(sk, \overline{M})$ ($\overline{M}$ bitw. Inverse von M),
- $\mathsf{Ver}^*(pk, M, \sigma) = \mathsf{Ver}(pk, \overline{M}, \sigma)$

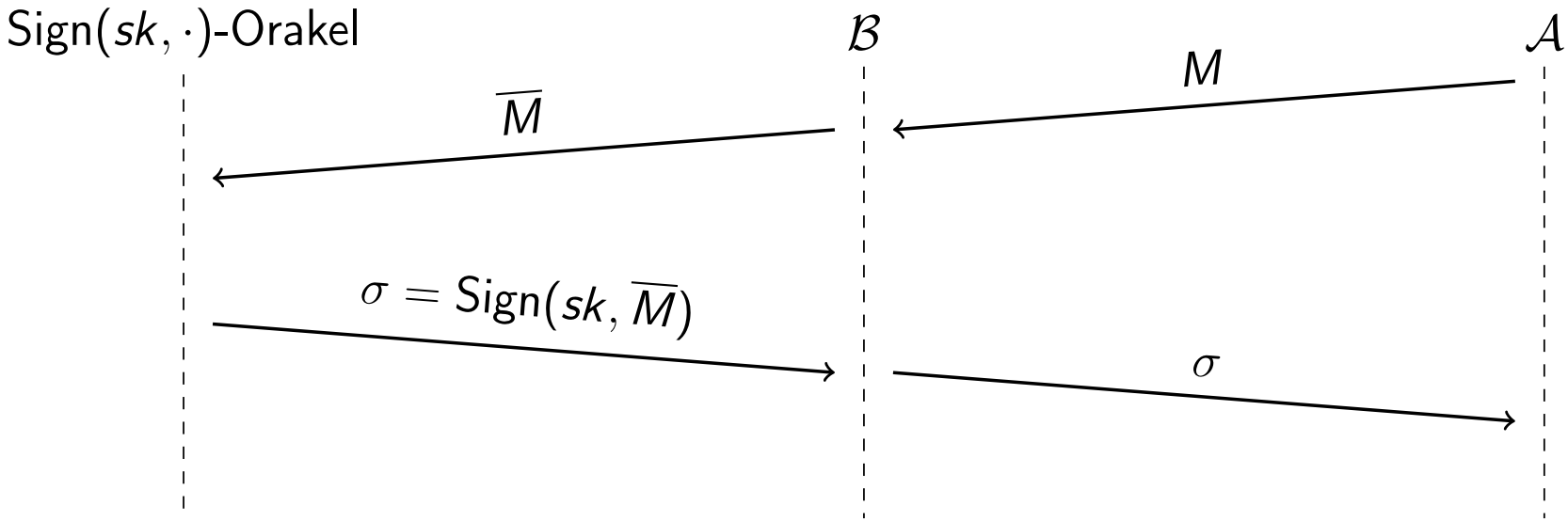
(b) Zeigen Sie: Σ^* ist EUF-CMA-sicher.
Widerspruchsbeweis/Reduktion:

- Annahme: Σ^* nicht EUF-CMA-sicher.
- D.h. es existiert PPT $\mathcal{A}$ mit nicht-vernachl. Erfolgswkt.
- Konstruiere Angreifer $\mathcal{B}$ gegen Σ mit nicht-vernachl. Erfolgswkt.
- Widerspruch zur EUF-CMA-Sicherheit von Σ.
- D.h. Σ^* ist EUF-CMA-sicher.

Sicherheit – Übungsblatt 4 – Aufgabe 3 (2) (b)

Simulation des Orakels:

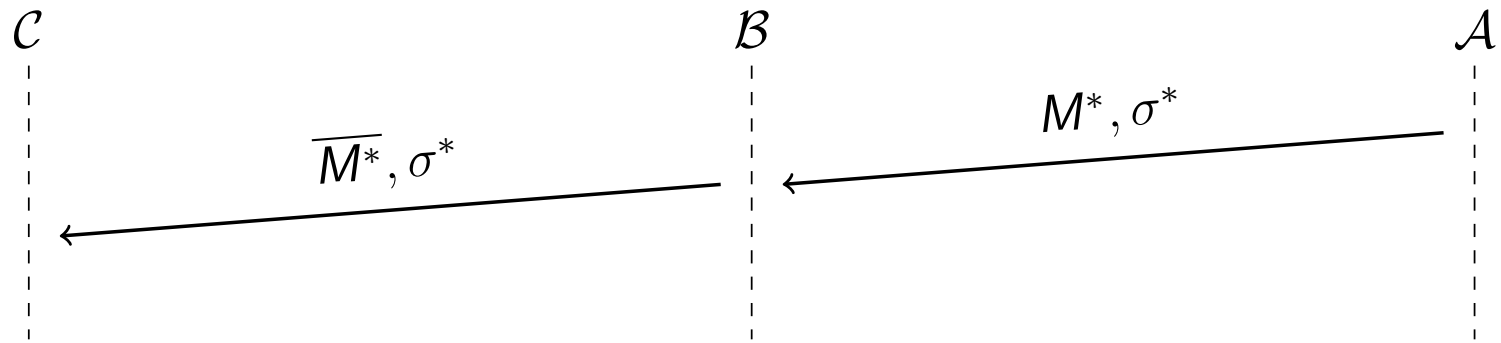
- $\mathcal{B}$ hat Zugriff auf Σ-Sign-Orakel.
- $\mathcal{A}$ braucht Zugriff auf Σ^*-Sign*-Orakel.
- $\mathcal{B}$ muss Σ^*-Sign*-Orakel simulieren.
- Zur Erinnerung: $\mathsf{Sign}^*(sk, M) := \mathsf{Sign}(sk, \overline{M})$



Sicherheit – Übungsblatt 4 – Aufgabe 3 (2) (b)

Verwendung der Fälschung:

- $\mathcal{A}$ schickt Fälschung M^*, σ^*.



Ist M^*, σ^* eine gültige Fälschung für Σ^* so gilt:

- $1 = \mathrm{Ver}^*(pk, M^*, \sigma^*) = \mathrm{Ver}(pk, \overline{M^*}, \sigma^*)$
- Wenn M^* nie von $\mathcal{A}$ ans Orakel geschickt, so schickt $\mathcal{B}$ nie $\overline{M^*}$ an sein Orakel...
- ... d.h. $\overline{M^*}$ ist „frisch“ (Wichtig!)

Sicherheit – Übungsblatt 4 – Aufgabe 3 (2) (b)

Beweisende:

- $\mathcal{B}$ simuliert das EUF-CMA-Spiel für $\mathcal{A}$ perfekt.
- Gibt $\mathcal{A}$ eine gültige Fälschung aus, so auch $\mathcal{B}$.
- $\mathcal{B}$ gewinnt $\Leftrightarrow$ $\mathcal{A}$ gewinnt.
- Also: Pr[$\mathcal{B}$ gewinnt] nicht vernachlässigbar.

$\Rightarrow$ Widerspruch zur EUF-CMA-Sicherheit von Σ.
$\Rightarrow$ Σ^* muss ebenfalls EUF-CMA-sicher sein.

Sicherheit – Übungsblatt 4 – Aufgabe 4

- Authentifizierter Kanal zwischen A(lice) und B(ob)
- Vereinfachung: Nur Nachrichten von A nach B
- Realisierung mit MAC
- MAC $=$ (Gen, Sign, Ver) EUF-CMA-sicher

Unterschied MAC/Digitale Signaturen

- MACs haben nur einen (geheimen) Schlüssel
- Angreifer hat bei EUF-CMA Zugriff auf Sign-Orakel,
- ... aber nicht auf Ver-Orakel!

Sicherheit – Übungsblatt 4 – Aufgabe 4

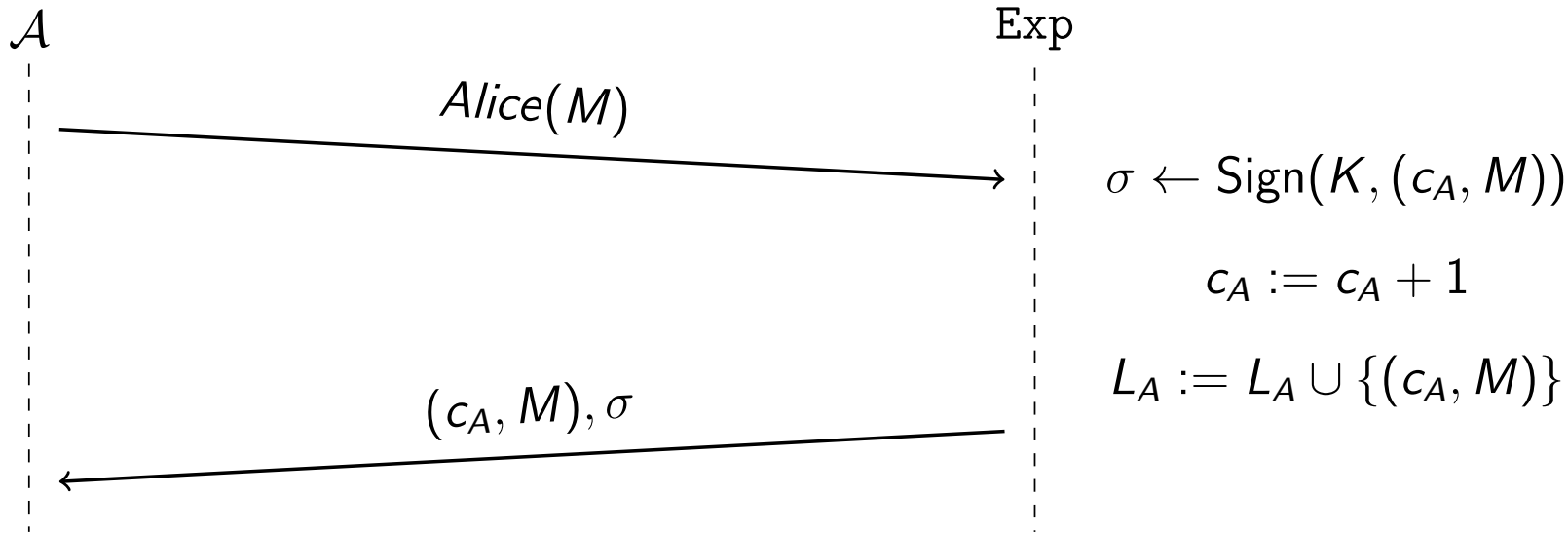
Sicherheitsdefinition von authentifiziertem Kanal:
Setup:

- Exp bereitet Schlüssel etc. vor.

Sicherheit – Übungsblatt 4 – Aufgabe 4

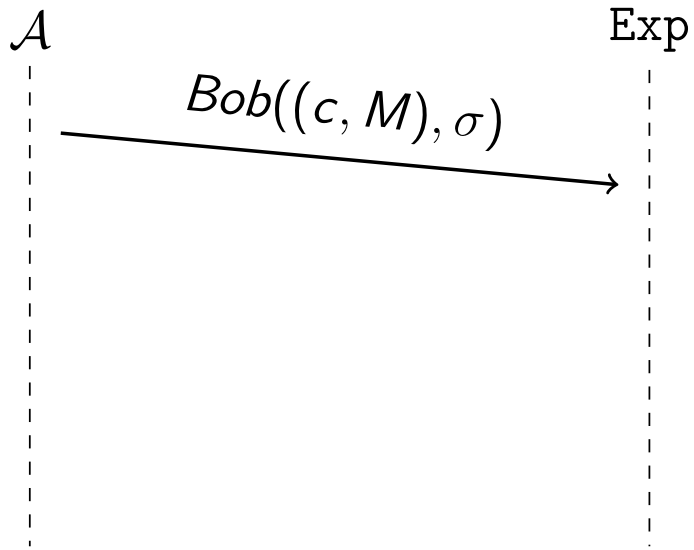
Sicherheitsdefinition von authentifiziertem Kanal:
Ablauf:

- $\mathcal{A}$ darf Nachrichten von A an B schicken: $Alice(M)$



Sicherheit – Übungsblatt 4 – Aufgabe 4

- $\mathcal{A}$ kann Bob ein Zähler/Nachrichten-Paar und eine Signatur empfangen lassen: $Bob((c, M), \sigma)$



wenn $\mathrm{Ver}(K, (c, M), \sigma) = 1$ und $c = c_B$, dann:

$$L_B := L_B \cup \{(c, M)\}$$

$$c_B := c_B + 1$$

sonst: gib 0 aus ($\mathcal{A}$ verliert!)

Sicherheit – Übungsblatt 4 – Aufgabe 4

Sicherheitsdefinition von authentifiziertem Kanal:
Ende:

- Exp überprüft, ob $L_B \subseteq L_A$
- wenn ja:
  - gebe 0 aus ($\mathcal{A}$ verliert)
- sonst:
  - gebe 1 aus ($\mathcal{A}$ gewinnt)
- Wkt., dass $\mathcal{A}$ gewinnt, soll vernachlässigbar sein.

Sicherheit – Übungsblatt 4 – Aufgabe 4

Sicherheitsdefinition von authentifiziertem Kanal:
Ende – Intuition:

- Ziel von $\mathcal{A}$: an Bob mind. eine Nachricht schicken, die
- ... Bob akzeptiert, ...
- ... die aber nicht von Alice geschickt wurde!
- $L_B \subseteq L_A$ heißt, dass Bob nur Nachrichten akzeptiert hat, die von Alice kamen!

Sicherheit – Übungsblatt 4 – Aufgabe 4

Zeige: Existiert ein PPT-Angreifer $\mathcal{A}$, der

- das Sicherheitsspiel gegen den auth. Kanal gewinnt
- (mit nicht-vernachlässigbarer Wahrscheinlichkeit!),

dann

- existiert ein PPT-Angreifer $\mathcal{B}$, der...
- dass EUF-CMA-Spiel gegen MAC gewinnt
- (mit nicht-vernachlässigbarer Wahrscheinlichkeit!).

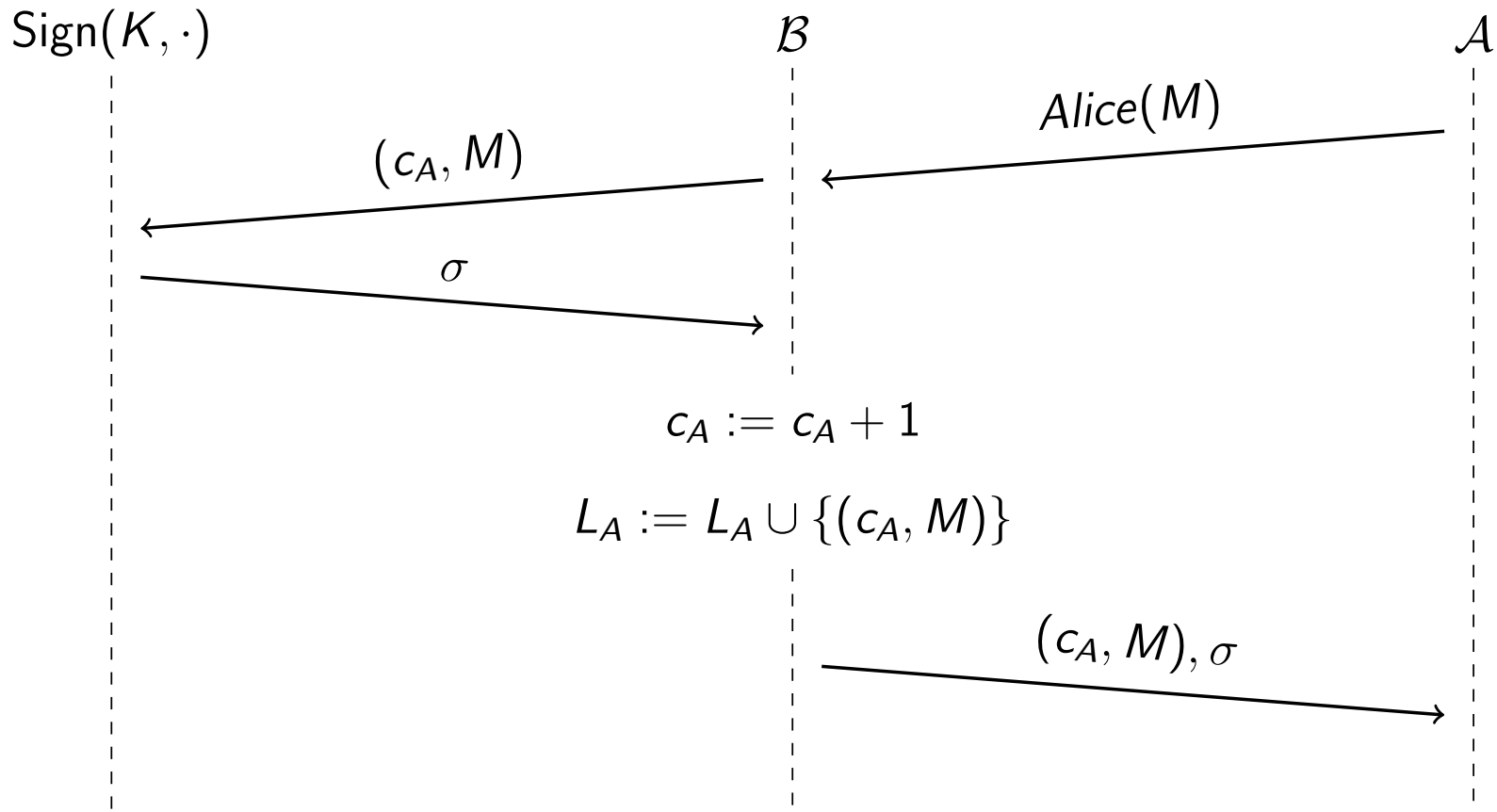
(d.h. MAC EUF-CMA-sicher $\Rightarrow$ Kanal sicher nach dieser Def.)

Sicherheit – Übungsblatt 4 – Aufgabe 4

- $\mathcal{B}$ verwendet $\mathcal{A}$.
- $\mathcal{B}$ spielt das EUF-CMA-Spiel mit MAC
- $\mathcal{B}$ hat Zugriff auf ein Sign$(K, \cdot)$-Orakel
- $\mathcal{B}$ kennt K nicht!
- $\mathcal{B}$ initialisiert den Kanal und berechnet:
  - $c_A := 0, c_B := 0$
  - $L_A := \emptyset$ (L_B wird zur Sim. nicht gebraucht)
- $\mathcal{B}$ startet $\mathcal{A}$

Sicherheit – Übungsblatt 4 – Aufgabe 4

Beantwortung von $Alice(M)$:



Anfragen werden richtig beantwortet!

Sicherheit – Übungsblatt 4 – Aufgabe 4

Beantwortung von $Bob((c_A, M), \sigma)$:

- Schwierig, da $\mathcal{B}$ kein Ver-Orakel hat...
- $\mathcal{B}$ kann Gültigkeit von σ nicht überprüfen
- Entscheidung nur anhand von „$(c, M) \stackrel{?}{\in} L_A$“

Sicherheit – Übungsblatt 4 – Aufgabe 4

$(c, M) \in L_A$:

- $\mathcal{B}$ prüft, ob $c = c_B$.
- Wenn ja: $c_B := c_B + 1$
- Sonst: Abbruch

σ gültig:

- Alles okay!

σ nicht gültig:

- $\mathcal{A}$ hätte nun eigentlich verloren...
- Simulation läuft trotzdem weiter
- Also: Simulation falsch!
- Es können jetzt keine Aussagen über $\mathcal{A}$ mehr getroffen werden
  $\Rightarrow$ Wir müssen davon ausgehen, dass $\mathcal{A}$ nun keine für $\mathcal{B}$ nützlichen Informationen mehr liefert...

Sicherheit – Übungsblatt 4 – Aufgabe 4

$(c, M) \notin L_A$:

- Gebe $(c, M), \sigma$ als Fälschung aus.

σ gültig:

- Fälschung gefunden
- Da $(c, M) \notin L_A$ ist Nachricht „frisch“.
- Gewinnt $\mathcal{A}$, so tritt dieser Fall ein!

σ nicht gültig:

- Dann zwar keine gültige Fälschung...
- ... aber Spiel würde dann eh abgebrochen
- ($\mathcal{A}$ hat „verloren“)

Sicherheit – Übungsblatt 4 – Aufgabe 4

- Gewinnt $\mathcal{A}$, so gibt $\mathcal{B}$ eine gültige Fälschung aus!
- Laufzeit von $\mathcal{B} \approx$ Laufzeit von $\mathcal{A}$
- Erfolgswkt. von $\mathcal{B}$ = Erfolgswkt. von $\mathcal{A}$
- Also: Aus $\mathcal{A}$ einen erfolgreichen MAC-Angreifer $\mathcal{B}$ konstruiert

Sicherheit – Übungsblatt 4 – Aufgabe 4

Welchen Zweck erfüllen die Zähler c_A und c_B?

- Verhinderung von Replay-Angriffen
- Replay-Angriff:
  - Aktiver Angriff
  - Nachrichten abhören...
  - ... und später einfach nochmal senden
- Problematisch z.B. bei
  - Fahrzeugelektronik
  - Türschloss
  - ...

Sicherheit – Übungsblatt 4 – Aufgabe 4

Fazit:

- Auth. Kanal realisierbar durch MACs & Zähler
- Ähnliche Implementierung bei TLS