Übung zur Vorlesung „Sicherheit“
Übung 3

Thomas Agrikola
Thomas.Agrikola@kit.edu

01.06.2017

Wdh.: Hashfunktionen

- „Fingerabdruck“ $H_k : \{0,1\}^* \to \{0,1\}^k$
- k ist Sicherheitsparameter
- **Kollisionsresistenz:** Für alle PPT-Algorithmen $\mathcal{A}$ ist

$$\Pr[(X, X') \leftarrow \mathcal{A}(1^k) \colon X \neq X' \wedge H_k(X) = H_k(X')]$$

  vernachlässigbar.

- Eine Kollision „reicht“, um die Kollisionsresistenz zu brechen!
- **Wichtig:** Nachricht für Kollision wird nicht vorgegeben!
  - Es gibt auch andere Definitionen
  - Wir beschränken uns auf die Obige!

Sicherheit – Übungsblatt 3 – Aufgabe 1

- $H, H' : \{0,1\}^* \to \{0,1\}^k$ beliebige kollisionsresistente Hashfunktionen.
- Daraus wird neue Hashfunktion $\hat{H}$ konstruiert.
- Frage: Ist $\hat{H}$ (für bel. koll. res. H, H') kollisionsresistent?

Socrative: Aufgabe 1

`https://b.socrative.com/login/student/`

Room: `SICHERHEIT`

- $H, H' : \{0,1\}^* \to \{0,1\}^k$ beliebige kollisionsresistente Hashfunktionen.
- Daraus wird neue Hashfunktion $\hat{H}$ konstruiert.
- Frage: Ist $\hat{H}$ (für bel. koll. res. H, H') kollisionsresistent?

Sicherheit – Übungsblatt 3 – Aufgabe 1 (a)

$\hat{H}(x) := H(x\|x)$ **ist kollisionsresistent**.

- Sei x, y mit $x \neq y$ eine Kollision.
- Dann gilt:

$$\hat{H}(x) = \hat{H}(y) \Rightarrow H(x\|x) = H(y\|y).$$

- $x \neq y \Rightarrow x\|x \neq y\|y$.
- Also ist $x\|x, y\|y$ Kollision für H.

Formal:

- Konstr. aus PPT $\mathcal{A}$ der Kollisionen für $\hat{H}$ findet...
- ... einen PPT $\mathcal{B}$, der Kollisionen für H findet...
- Widerspruch zur Kollisionsresistenz von H.
- Also: $\hat{H}$ kollisionsresistent.

Sicherheit – Übungsblatt 3 – Aufgabe 1 (b)

$\hat{H}(x) := H(x) \oplus H'(x)$ **ist im Allg. nicht kollisionsresistent**.

- Betrachte $H = H'$.
- Dann gilt für alle x:

$$\hat{H}(x) = H(x) \oplus H(x) = 0^k.$$

- Und somit für alle x, y:

$$\hat{H}(x) = \hat{H}(y).$$

Also: Kollisionsres. hier abhängig von den Hashfunktionen!
(auch bei $H \neq H'$ problematisch, betrachte z.B. $H'(x) = \overline{H(x)}$)

Sicherheit – Übungsblatt 3 – Aufgabe 1 (c)

$\hat{H}(x) := H(H'(x))$ **ist kollisionsresistent**.

- Sei x, y mit $x \neq y$ eine Kollision, also

$$\hat{H}(x) = H(H'(x)) = H(H'(y)) = \hat{H}(y).$$

- Fall 1: $H'(x) = H'(y)$
  - ... ist Kollision für H'.
  - Konstr. Widerspruch zur Kollres. von H'.
- Fall 2: $H'(x) \neq H'(y)$
  - ... dann ist $H'(x), H'(y)$ eine Kollision für H.
  - Konstr. Widerspruch zur Kollres. von H.

Formal: Konstruiere Angreifer...

Sicherheit – Übungsblatt 3 – Aufgabe 1 (d)

$\hat{H}(x) := F(x)\|H(x)$ mit F bel. Fkt. **ist kollisionsresistent**.

- Sei x, y mit $x \neq y$ eine Kollision.
- Dann gilt:

$$\begin{aligned} & \hat{H}(x) = \hat{H}(y) \\ \Rightarrow\ & F(x)\|H(x) = F(y)\|H(y) \\ \Rightarrow\ & H(x) = H(y). \end{aligned}$$

- D.h. x, y ist auch Kollision für H.
- Widerspruch zur Kollisionsresistenz von H.

Sicherheit – Übungsblatt 3 – Aufgabe 1 (e)

$\hat{H}(x) := \begin{cases} x & \text{wenn } |x| = k \\ H(x) & \text{sonst} \end{cases}$ **ist nicht kollisionsresistent**.

Für alle y mit $|y| \neq k$ gilt:

- $\hat{H}(y) = H(y)$
- Aber: $|H(y)| = k$...
- ... also $\hat{H}(H(y)) = H(y)$.
- Außerdem $H(y) \neq y$ (unterschiedliche Länge!)

Somit: $y, H(y)$ Kollision für $\hat{H}$.

Sicherheit – Übungsblatt 3 – Aufgabe 1 (f)

$\hat{H}(x) := H(F(x))$ für eine injektive Fkt. $F : \{0,1\}^* \to \{0,1\}^*$ **ist kollisionsresistent**.

- Sei x, y mit $x \neq y$ eine Kollision.
- Dann gilt:

$$H(F(x)) = H(F(y)).$$

- Da F injektiv, ist

$$F(x) \neq F(y).$$

- D.h. $F(x), F(y)$ ist eine Kollision für H.
- Widerspruch zur Kollisionsresistenz von H.

Sicherheit – Übungsblatt 3 – Aufgabe 2 (a)

RSA-Verschlüsselung: $N = P \cdot Q$ mit den Primzahlen $P = 23$ und $Q = 11$.

(i) Setzen Sie die RSA-Schlüsselgenerierung fort.

(ii) Verschlüsseln Sie die Nachricht $M = 17$.

(iii) Entschlüsseln Sie das Chiffrat aus (ii).

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

Schlüsselgenerierung:
$N = P \cdot Q = 253$, $\varphi(N) = (P-1)(Q-1) = 220$

- Ziehe $e \leftarrow \{3, ..., \varphi(N) - 1\}$ mit $\text{ggT}(e, \varphi(N)) = 1$.
- Zum Beispiel $e = 19$.
- Berechne $d = e^{-1} \bmod \varphi(N)$ mit dem erweiterten euklidischen Algorithmus.

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

Erweiterter euklidischer Algorithmus (EEA) für A, B berechnet $s, t \in \mathbb{Z}$ mit:

$$\mathrm{ggT}(A, B) = s \cdot A + t \cdot B.$$

In unserem Fall:

$$\underbrace{\mathrm{ggT}(\varphi(N), e)}_{=1}\ \mathrm{ggT}(\varphi(N), e) = s \cdot \varphi(N) + t \cdot e$$

$$\Rightarrow \qquad 1 = t \cdot e \bmod \varphi(N)$$

Also setzen wir $d := t$.

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

Jetzt der Algorithmus mit $\varphi(N) = 220, e = 19$:

$$
\begin{aligned}
220 &= 11 \cdot 19 + 11 \\
19 &= 1 \cdot 11 + 8 \\
11 &= 1 \cdot 8 + 3 \\
8 &= 2 \cdot 3 + 2 \\
3 &= 1 \cdot 2 + 1 \\
2 &= 2 \cdot 1 + 0
\end{aligned}
$$

$\Rightarrow \mathrm{ggT}(220, 19) = 1$

$$
\begin{aligned}
1 &= 3 - 1 \cdot 2 \\
&= 3 - (8 - 2 \cdot 3) \\
&= -1 \cdot 8 + 3 \cdot 3 \\
&= -1 \cdot 8 + 3 \cdot (11 - 1 \cdot 8) \\
&= 3 \cdot 11 - 4 \cdot 8 \\
&= 3 \cdot 11 - 4 \cdot (19 - 1 \cdot 11) \\
&= -4 \cdot 19 + 7 \cdot 11 \\
&= -4 \cdot 19 + 7 \cdot (220 - 11 \cdot 19) \\
&= 7 \cdot 220 + \underbrace{(-81)}_{=d} \cdot 19
\end{aligned}
$$

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

Also gilt:

$$\begin{aligned} d &= -81 && \bmod \varphi(N) \\ &= -81 + \varphi(N) && \bmod \varphi(N) \\ &= -81 + 220 && \bmod \varphi(N) \\ &= 139 && \bmod \varphi(N) \end{aligned}$$

Insgesamt:

$$pk = (N, e) = (253, 19)$$
$$sk = (N, d) = (253, 139)$$

Sicherheit – Übungsblatt 3 – Aufgabe 2 (a)(ii),(iii)

Verschlüsseln Sie die Nachricht $M = 17$.
$pk = (N, e) = (253, 19)$

$$C := M^e \bmod N = 17^{19} \bmod 253 = 189.$$

Entschlüsseln Sie das Chiffrat $C = 189$.
$sk = (N, d) = (253, 139)$

$$M := C^d \bmod N = 189^{139} \bmod 253 = 17.$$

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

Rechentrick:

- Sei ϕ die Abbildung $\phi\colon \mathbb{Z}_N \to \mathbb{Z}_P \times \mathbb{Z}_Q$, $x \mapsto (x \bmod P, x \bmod Q)$.
- Chinesischer Restsatz: ϕ ist ein Ring-Isomorphismus.
- Die **Umkehrabbildung** ϕ^{-1} ist wie folgt definiert:

$$\phi^{-1}(a,b) := a \cdot Q \cdot (Q^{-1} \bmod P) + b \cdot P \cdot (P^{-1} \bmod Q)$$

  - denn: $\phi^{-1}(a,b) \bmod P =$
$$\left(a \cdot \underbrace{Q \cdot (Q^{-1} \bmod P)}_{=1} + b \cdot \underbrace{P \cdot (P^{-1} \bmod P)}_{=0} \right) \bmod P.$$
  - für Q analog

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

Rechentrick: $189^{139} \bmod 253$

$$
\begin{aligned}
\phi(189^{139}) &= (189^{139} \bmod 23, 189^{139} \bmod 11) \\
&= (189^{139 \bmod 22} \bmod 23, 189^{139 \bmod 10} \bmod 11) \\
&= (5^7 \bmod 23, 2^9 \bmod 11) \\
&= (\underbrace{25}_{=2} \cdot \underbrace{\underbrace{25}_{=2} \cdot \underbrace{25}_{=2}}_{=20=-3} \cdot 5 \bmod 23, \underbrace{\underbrace{16}_{=5} \cdot \underbrace{16}_{=5}}_{=3} \cdot 2 \bmod 11) \\
&= (17 \bmod 23, 6 \bmod 11)
\end{aligned}
$$

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

Rechentrick: $189^{139} \bmod 253$

$$\phi^{-1}(a, b) = a \cdot Q \cdot (Q^{-1} \bmod P) + b \cdot P \cdot (P^{-1} \bmod Q)$$

$$\begin{aligned}
&\phi^{-1}(17 \bmod 23, 6 \bmod 11) \\
&= \underbrace{\underbrace{17 \cdot 11 \cdot (\underbrace{11^{-1} \bmod 23}_{=21=-2})}_{=132} + \underbrace{6 \cdot 23 \cdot (\underbrace{23^{-1} \bmod 11}_{=1})}_{=138}}_{=270} \bmod 253 \\
&= 17 \bmod 253
\end{aligned}$$

Sicherheit – Übungsblatt 3 – Aufgabe 2 (b)

Die Funktion Rab_N ist wie folgt definiert:

$$\begin{aligned}\text{Rab}_N \colon \mathbb{Z}_N &\to \mathbb{Z}_N, \\ x &\mapsto x^2 \bmod N\end{aligned}$$

Wieso ist die Funktion nicht auch so zu invertieren wie bei RSA?

$$\text{ggT}(2, \varphi(N)) = \text{ggT}(2, \underbrace{(P-1)}_{\text{gerade}} \cdot \underbrace{(Q-1)}_{\text{gerade}}) = 2$$

$\Rightarrow$ 2 hat kein multiplikativ Inverses modulu $\varphi(N)$.

Sicherheit – Übungsblatt 3 – Aufgabe 2 (c)

Die Funktion Rab_N ist wie folgt definiert:

$$\begin{aligned}\mathrm{Rab}_N : \mathbb{Z}_N &\to \mathbb{Z}_N, \\ x &\mapsto x^2 \bmod N\end{aligned}$$

- Annahme: es existiert ein effizienter Algorithmus $\mathcal{A}$, der bei Eingabe (N, y), ein x berechnet, sodass $\mathrm{Rab}_N(x) = y$ gilt.
  Vereinfachung: $\mathcal{A}$ ist immer erfolgreich.
- Zeige: Dann gibt es auch einen effizienten Algorithmus $\mathcal{B}$, der RSA-Moduli faktorisiert.

Sicherheit – Übungsblatt 3 – Aufgabe 2 (c)

$\mathcal{B}$ funktioniert wie folgt:

- Bei Eingabe $N = P \cdot Q$ zieht $\mathcal{B}$ eine zufällige Zahl x aus $\mathbb{Z}_N$ und berechnet $y := \text{Rab}_N(x) = x^2 \bmod N$.
- Dann ruft $\mathcal{B}$ den Algorithmus $\mathcal{A}$ mit Eingabe (N, y) auf.
- $\mathcal{A}$ berechnet ein x', sodass $\text{Rab}_N(x') = y = \text{Rab}_N(x)$ gilt.
  - nach dem chinesischen Restsatz hat y die folgenden Quadratwurzeln in $\mathbb{Z}_N \cong \mathbb{Z}_P \times \mathbb{Z}_Q$:

$$(x \bmod P, x \bmod Q), (-x \bmod P, -x \bmod Q)$$
$$(x \bmod P, -x \bmod Q), (-x \bmod P, x \bmod Q)$$

  - (die unteren beiden sind für $\mathcal{B}$ interessant, die oberen beiden kennt $\mathcal{B}$ selbst, nämlich $\pm x \bmod N$)

Sicherheit – Übungsblatt 3 – Aufgabe 2 (c)

- $\mathcal{A}$ findet mit Wahrscheinlichkeit $\frac{1}{2}$ eine Wurzel x', sodass $x' \not\equiv \pm x \bmod N$ (er weiß ja nicht, welches x von $\mathcal{B}$ ursprünglich gewählt wurde)
- es gilt

$$\begin{aligned} & x^2 \equiv x'^2 \bmod N \\ \Leftrightarrow & (x - x') \cdot (x + x') \equiv 0 \bmod N \\ \Leftrightarrow & (x - x') \cdot (x + x') = k \cdot N \text{ für ein } k \in \mathbb{Z}. \end{aligned}$$

- es gilt aber $(x \pm x') \not\equiv 0 \bmod N$ (da $x' \not\equiv \pm x \bmod N$)
  $\Rightarrow x - x'$ und $x + x'$ sind keine Vielfachen von N.
- $x - x'$ und $x + x'$ enthalten je genau einen der Faktoren P und Q
  $\Rightarrow$ finde Faktoren durch z.B. ggT$(x - x', N)$

Sicherheit – Übungsblatt 3 – Aufgabe 3

- $\mathsf{F} : \{0,1\}^{2k} \to \{0,1\}^k$ kollres. Fkt.
- $|M| = L < 2^k$
- Definiere $\mathsf{H}_{\mathsf{MD}}(M)$ wie folgt:
  1. Setze $B := \lceil \frac{L}{k} \rceil$.
  2. Füge Padding hinzu:

$$M \| 0^{k - L \bmod k} = M_1 \| \ldots \| M_B$$

  3. Nun insgesamt B k-Bit-Blöcke $M_1, \ldots, M_B$.
  4. Setze $M_{B+1} := L$, wobei L mit exakt k Bits codiert wird.
  5. Setze $Z_0 := 0^k$.
  6. Für $i = 1, \ldots, B+1$ berechne $Z_i := \mathsf{F}(Z_{i-1} \| M_i)$.
  7. Gib Z_{B+1} aus.

Sicherheit – Übungsblatt 3 – Aufgabe 3

Betrachte $H'_{MD}(M) = Z_B$ (statt Z_{B+1}!).
Ist H'_{MD} kollisionsresistent?

H'_{MD} ist nicht kollisionsresistent!

- Länge der Nachricht fließt nicht mehr in Hashwert ein.
- Sei M Nachricht mit $|M| \bmod k \neq 0$.
  - M muss also gepaddet werden!
- Dann ist z.B. $M' := M \| 0$ eine Kollision für M.
  - M' hat Länge $L + 1$.

Sicherheit – Übungsblatt 3 – Aufgabe 3

Betrachte den Padding Schritt:

- Bei $H'_{MD}(M)$:

$$M \| 0^{k-L \bmod k}$$

- Bei $H'_{MD}(M')$:

$$\begin{aligned} M' \| 0^{k-(L+1) \bmod k} &= (M \| 0) \| 0^{k-(L+1) \bmod k} \\ &= M \| 0^{k-L \bmod k} \end{aligned}$$

Sicherheit – Übungsblatt 3 – Aufgabe 3

- Nach Padding verwenden also beide die selben Blöcke.
- Somit:

$$H'_{MD}(M) = H'_{MD}(M')$$

- Man hat quasi Nuller vom Padding in die Nachricht geschoben.

- Nachricht M mit passender Länge ist leicht zu finden...
- Außerdem gilt natürlich $M \neq M'$.
- Also ist H'_{MD} nicht kollisionsresistent.

Sicherheit – Übungsblatt 3 – Aufgabe 3

Idee: Beschränkung auf Nachrichten M mit $|M| = n \cdot k$, $n \in \mathbb{N}$
Ist H'_{MD} dann kollisionsresistent?

- Antwort: Nein, auch dann nicht!
- Abhängig von F
- Genaueres siehe Musterlösung

Sicherheit – Übungsblatt 3 – Aufgabe 3

Fazit:

- Für die Sicherheit der Merkle-Damgård-Konstruktion ist es essentiell, dass wir die Länge der Eingabe in den Hash einbringen (außer es sind nur Eingaben gleicher fester Länge erlaubt)

Gruppen (Wiederholung)

- Algebraische Struktur, Paar $(\mathbb{G}, *)$
- $\mathbb{G}$ ist eine Menge
  - In unserer VL nur endliche Gruppen interessant
  - D.h. Mengen mit $|\mathbb{G}| \in \mathbb{N}$
- $*$ ist eine zweistellige Verknüpfung von Elementen aus $\mathbb{G}$ nach $\mathbb{G}$, d.h.

$$\forall a, b \in \mathbb{G} : a * b \in \mathbb{G}$$

- Eine Gruppe hat mehrere Eigenschaften:
  - Assoziativität
  - Neutrales Element
  - Inverse Elemente
- Eigenschaften erfassen „Rechenregeln“, die uns eigentlich geläufig sind.

Gruppen: Assoziativität

Assoziativität: Für alle $a, b, c \in \mathbb{G}$ gilt

$$(a * b) * c = a * (b * c).$$

Vergleiche mit Addition ganzer Zahlen, z.B.:

$$(1 + 2) + 3 = 1 + (2 + 3).$$

Gruppen: Neutrales Element

Neutrales Element: Es existiert $e \in \mathbb{G}$, sodass für alle $g \in \mathbb{G}$ gilt

$$e * g = g.$$

Vergleiche z.B. (bei ganzen Zahlen) mit

- Addition mit Null, $x + 0 = x$
- Multiplikation mit Eins, $x \cdot 1 = x$

Gruppen: Inverse Elemente

Inverse Elemente: Für jedes $g \in \mathbb{G}$ existiert ein $g^{-1} \in \mathbb{G}$, sodass

$$g * g^{-1} = e.$$

Vergleiche mit Addition ganzer Zahlen, zu $a \in \mathbb{Z}$ ist $-a \in \mathbb{Z}$ das inverse Element:

$$a + (-a) = 0.$$

Bezüglich Multiplikation gibt es keine inversen Elemente.
„Teilen“ geht nicht, z.B. $1/2 \notin \mathbb{Z}$.

Gruppen: Beispiel

$(\mathbb{Z}, +)$, $(\mathbb{Z}_N, +)$ und $(\mathbb{Z}_P \setminus \{0\}, \cdot)$ sind Gruppen.

Anmerkung: Wir „schreiben“ Gruppen meist multiplikativ.

Zyklische Gruppen (1)

Sei $(\mathbb{G}, \cdot)$ Gruppe.

- In VL besonders wichtig: *Zyklische* Gruppen
- Es existiert mindestens ein Element $g \in \mathbb{G}$, das $\mathbb{G}$ erzeugt.
- D.h. für jedes $a \in \mathbb{G}$ existiert ein $n \in \mathbb{Z}$, sodass $g^n = a$, wobei

$$
\begin{aligned}
n > 0: &\qquad g^n = \underbrace{g \cdot g \cdot \ldots \cdot g}_{n\text{-mal}} \\
n = 0: &\qquad g^0 = e \text{ (neutrales Element)} \\
n < 0: &\qquad g^n = \underbrace{g^{-1} \cdot g^{-1} \cdot \ldots \cdot g^{-1}}_{|n|\text{-mal}}.
\end{aligned}
$$

- g heißt Erzeuger von $\mathbb{G}$

Gruppen: Anmerkungen

- „Sicherheit“ ist keine Algebra-Vorlesung...
- ... wir müssen aber einige Kenntnisse aus „Lineare Algebra“ voraussetzen.

- Wichtige Begriffe/Sätze, die man kennen sollte:
  - Ordnung einer Gruppe/eines Gruppenelements
  - Satz von Lagrange
  - Kleiner Fermat
  - ...
- Wir arbeiten (in den meisten Fällen) eher abstrakt mit Gruppen!
- Für Krypto interessante Gruppen: $(\mathbb{Z}_P \setminus \{0\}, \cdot)$, elliptische Kurven, Gitter

Sicherheit – Übungsblatt 3 – Aufgabe 4

Doktor Meta Aufgabe (Story siehe Blatt)

Sei $\mathbb{G}$ zyklische Gruppe, Ordnung p prim, g Erzeuger

- x zufälliger Wert aus $\mathbb{Z}_p$, $h := g^x$
- Nachrichten sind Tupel aus $\mathbb{Z}_p \times \mathbb{Z}_p$
- Hashfunktion H:

$$H(M) = H((M_1, M_2)) = g^{M_1} h^{M_2}.$$

- g, h öffentlich bekannt, x geheim (nur Doktor Meta bekannt).

Sicherheit – Übungsblatt 3 – Aufgabe 4

Aufgaben:

(a) Zeigen Sie, dass H kollisionsresistent ist.

(b) Warum kann Doktor Meta trotzdem Kollisionen berechnen?

(c) Warum kann jeder Kollisionen berechnen, sobald eine Kollision öffentlich bekannt wird?

Sicherheit – Übungsblatt 3 – Aufgabe 4 (a)

Zeige: H kollisionsresistent.

Annahme: Diskreter Logarithmus in $\mathbb{G}$ schwierig, d.h. für alle PPT $\mathcal{A}$ ist

$$\Pr[\mathcal{A}(g, g^x) = x \mid x \leftarrow \mathbb{Z}_p]$$

vernachlässigbar.

- Reduktion auf DLog
  - Zeige: H nicht kollres. $\Rightarrow$ DLog einfach.
  - Widerspruch zur DLog-Annahme $\Rightarrow$ H kollres.

Sicherheit – Übungsblatt 3 – Aufgabe 4 (a)

Also: Angenommen H nicht kollisionsresistent.

- D.h. es existiert PPT-Algo. $\mathcal{A}$, der mit nicht-vernachlässigbarer Wkt. eine Kollision findet.

Konstruiere aus $\mathcal{A}$ einen PPT-Algo. $\mathcal{B}$, der:

- als Eingabe g, g^x erhält
- und dann $\mathcal{A}$ verwendet, um x zu berechnen.

Sicherheit – Übungsblatt 3 – Aufgabe 4 (a)

$\mathcal{B}$ erhält g, g^x als Eingabe.

- Er setzt $h := g^x$ und gibt g, h an $\mathcal{A}$ (als Hashfkt.) weiter
- $\mathcal{A}$ gibt Kollision $M \neq M^*$ aus (mit nicht-vernach. Wkt.).
- $\mathcal{B}$ berechnet daraus x wie folgt:

$$
\begin{aligned}
H(M) = H(M^*) \quad &\Leftrightarrow \quad g^{M_1} h^{M_2} = g^{M_1^*} h^{M_2^*} \\
&\Leftrightarrow \quad g^{M_1 + xM_2} = g^{M_1^* + xM_2^*} \\
&\Leftrightarrow \quad M_1 + xM_2 = M_1^* + xM_2^* \mod p \\
&\Leftrightarrow \quad (M_2 - M_2^*)x = M_1^* - M_1 \mod p \\
&\Leftrightarrow \quad x = \frac{M_1^* - M_1}{M_2 - M_2^*} \mod p.
\end{aligned}
$$

$\uparrow$

$M_2 \neq M_2^* \bmod p$, sonst auch $M_1 = M_1^* \bmod p$
und damit $M = M^*$ (Widerspruch)

Sicherheit – Übungsblatt 3 – Aufgabe 4 (a)

Somit gilt:

$$\Pr[\mathcal{B}(g, g^x) = x \mid x \leftarrow \mathbb{Z}_p] = \Pr[\mathcal{A} \text{ berechnet Kollision}].$$

- $\Pr[\mathcal{A}$ berechnet Kollision$]$ nach Annahme nicht-vernachl.
- $\Rightarrow$ Widerspruch zu DLog-Annahme!
- Also: $\mathcal{A}$ kann nicht existieren und somit ist H kollisionsresistent.

Sicherheit – Übungsblatt 3 – Aufgabe 4 (b)

Warum kann Doktor Meta trotzdem Kollisionen berechnen?

- Doktor Meta kennt x!
- Er wählt zufällig M_1, M_2, M_1^* (mit $M_1 \neq M_1^*$) ...
- ... und löst die Gleichung (nach M_2^*)

$$
\begin{aligned}
g^{M_1} h^{M_2} = g^{M_1^*} h^{M_2^*} \quad &\Leftrightarrow \quad g^{M_1 + xM_2} = g^{M_1^* + xM_2^*} \\
&\Leftrightarrow M_1 + xM_2 = M_1^* + xM_2^* && \mod p \\
&\Leftrightarrow \qquad M_2^* = \frac{M_1 - M_1^*}{x} + M_2 && \mod p
\end{aligned}
$$

- $M = (M_1, M_2), M^* = (M_1^*, M_2^*)$ ist Kollision.
- ($M \neq M^*$, da $M_1 \neq M_1^*$.)

Sicherheit – Übungsblatt 3 – Aufgabe 4 (c)

Warum kann jeder Kollisionen berechnen, sobald eine Kollision öffentlich bekannt wird?

- Wenn eine Kollision bekannt ist...
- ... kann man wie in (a) x berechnen.

- Wenn man x kennt...
- ... kann man wie in (b) beliebige Kollisionen berechnen.

Socrative: Wdh. & Digitale Signaturen

`https://b.socrative.com/login/student/`

Room: `SICHERHEIT`

- App um Quizze durchzuführen
- Zugang durch Browser oder App
- Als Quizteilnehmer kein Account notwendig.

Digitale Signaturen

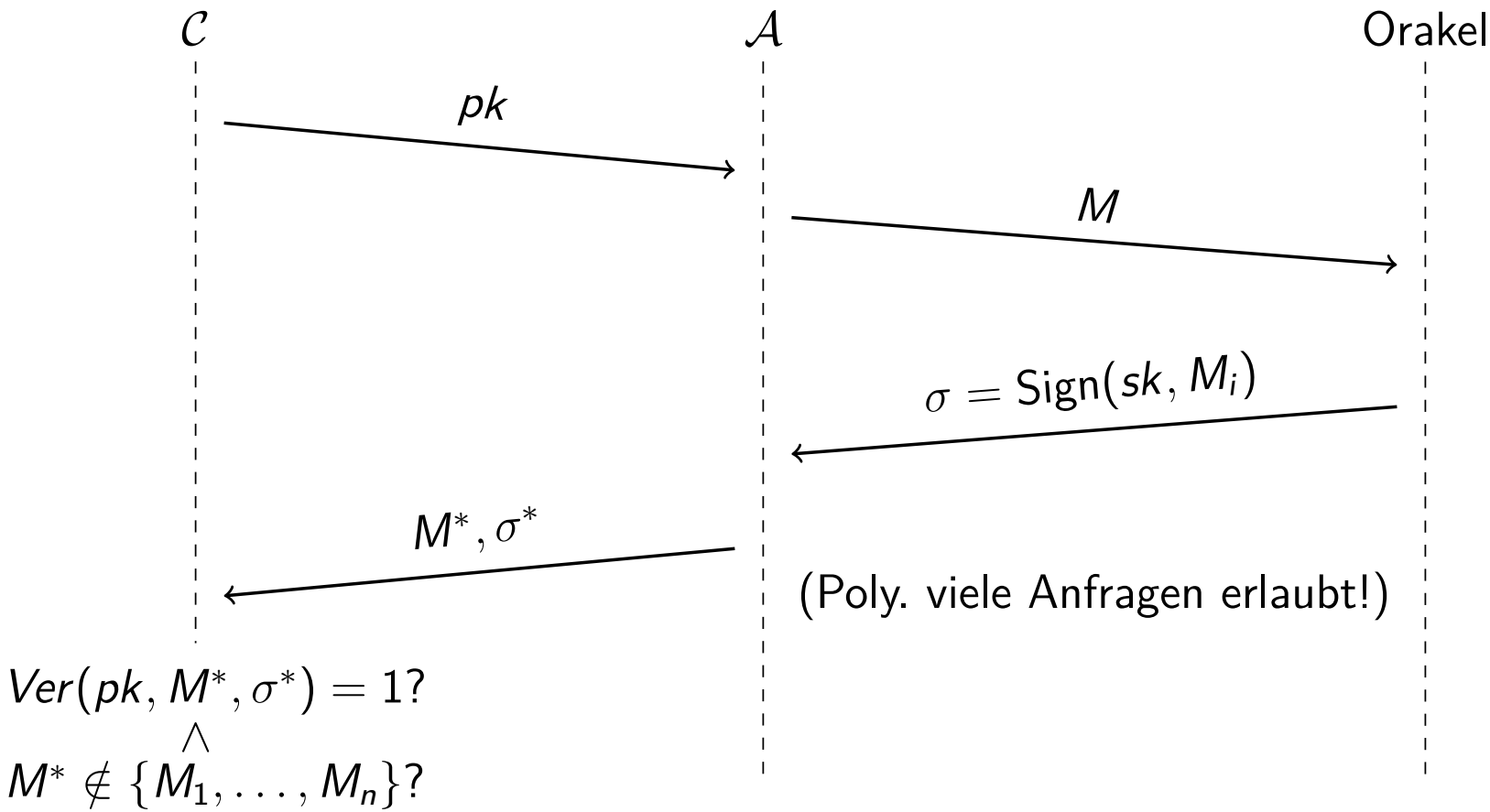
- Gen(1^k): erzeugt Schlüsselpaar pk, sk
- Sign(sk, M): berechnet Signatur σ für M.
- Ver(pk, M, σ): Überprüft σ für M, gibt 0 oder 1 aus.

- Korrektheit: $\text{Ver}(pk, M, \sigma) = 1$ für alle $(pk, sk) \leftarrow \text{Gen}(1^k)$, alle M und alle $\sigma = \text{Sign}(sk, M)$.

- Ziel eines Angreifers: Signaturen fälschen.
- Sicherheitsbegriff: EUF-CMA.

Signaturen – EUF-CMA: Ablauf

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



EUF-CMA: Beispiel

Sei $\Sigma = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ EUF-CMA-sicher,
d.h. für alle PPT-Angreifer $\mathcal{A}$ ist

$$\Pr[\mathcal{A} \text{ gewinnt im EUF-CMA-Spiel}]$$

vernachlässigbar.

Betrachte Σ' mit:

- $\mathsf{Gen}'(1^k) := \mathsf{Gen}(1^k)$
- $\mathsf{Sign}'(sk, M) := \mathit{Sign}(sk, M \| M)$
- $\mathsf{Ver}'(pk, M, \sigma) := \mathsf{Ver}(pk, M \| M, \sigma)$

Zeige: Σ' ist EUF-CMA-sicher (da Σ EUF-CMA-sicher).

EUF-CMA: Vorgehen

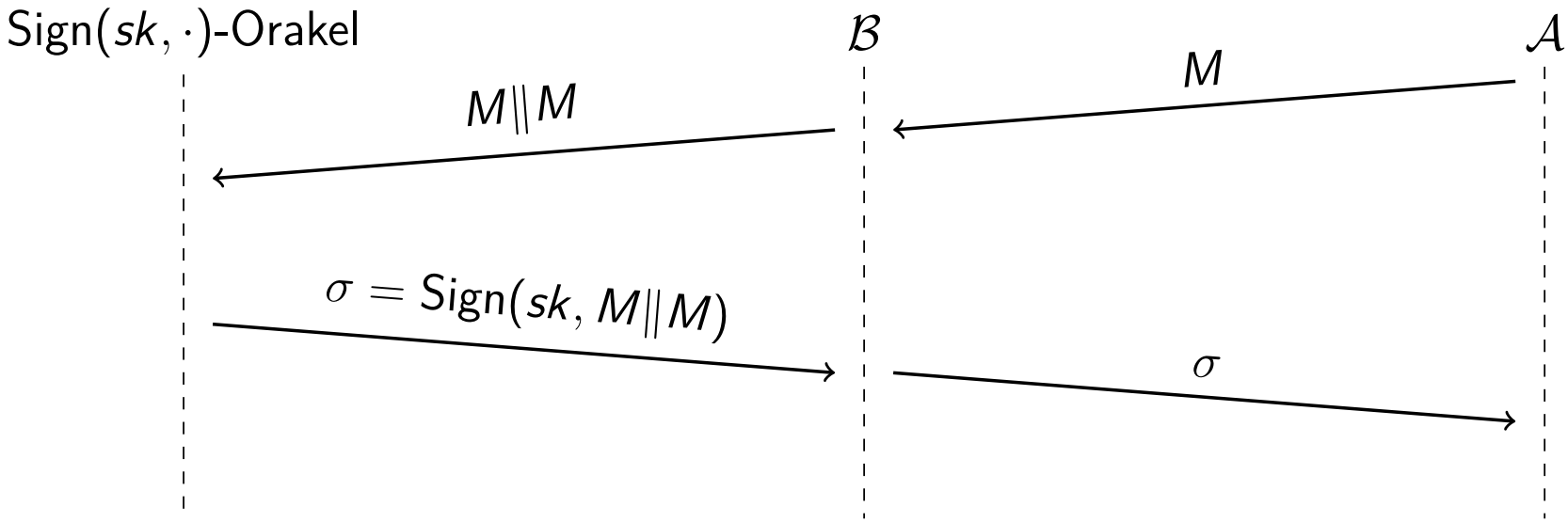
- Wie immer: Widerspruchsbeweis/Reduktion
- Annahme: Σ' nicht EUF-CMA-sicher.
- Konst. aus Angreifer $\mathcal{A}$ für Σ' Angreifer $\mathcal{B}$ für Σ.
- Widerspruch zur Sicherheit von Σ

Wieder zwei Schritte:

- Simulation des EUF-CMA-Spiels (insbes. des Orakels) für $\mathcal{A}$ mit Σ'.
- Verwendung der Fälschung von $\mathcal{A}$ für Σ' um Fälschung für Σ zu finden.

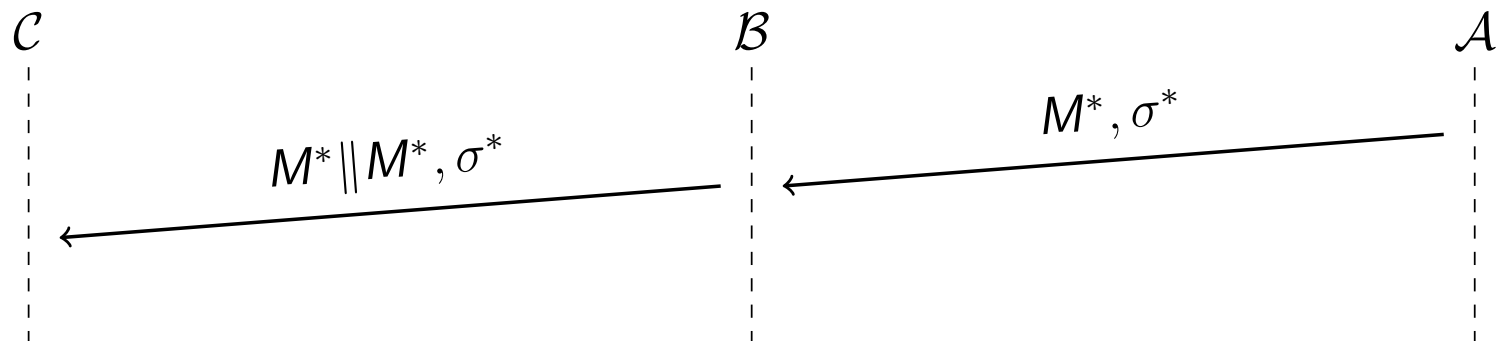
Simulation des Orakels

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



Verwendung der Fälschung

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



Ist $M^*, \sigma*$ eine gültige Fälschung für Σ' so gilt:

- $M^* \| M^*$, $\sigma*$ ist gültige Fälschung für Σ, denn $\mathrm{Ver}(pk, M^* \| M^*, \sigma*) = \mathrm{Ver}'(pk, M^*, \sigma^*) = 1$.
- Wenn M^* nie von $\mathcal{A}$ ans Orakel geschickt, so schickt $\mathcal{B}$ nie $M^* \| M^*$ an sein Orakel.

Beweisende

- $\mathcal{B}$ simuliert das EUF-CMA-Spiel für $\mathcal{A}$ perfekt.
- $\mathcal{A}$ gewinnt $\Rightarrow$ $\mathcal{B}$ gewinnt.
- Also: Pr[$\mathcal{B}$ gewinnt] nicht vernachlässigbar.

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