Sicherheit von Lamport Einwegsignaturen

Satz CMA-Sicherheit von Lamport

Unter der Annahme, dass f eine Einwegfunktion ist sind Lamport Einwegsignaturen CMA-sicher für Nachrichtenlänge ℓ polynomiell in n.

Beweis: Sei $\Pi = (Gen, Sign, Vrfy)$ das Lamport Signaturverfahren.

- Sei $\mathcal{A}$ ein Angreifer mit $\epsilon(n) := \mathrm{Ws}[Forge^{einweg}_{\mathcal{A},\Pi}(n) = 1]$.
- Wir konstruieren einen Invertierer $\mathcal{A}'$ für f mittels $\mathcal{A}$.

Algorithmus Invertierer $\mathcal{A}'$

EINGABE: y

1. Wähle $i \in_R \{0,1\}^\ell$ und $b \in_R \{0,1\}$.
2. Berechne $(pk, sk) \leftarrow Gen(1^n)$. Setze $y_{i,b} \leftarrow y$ in pk.
3. $(m, \sigma) \leftarrow \mathcal{A}$. Bei Signaturanfrage für m' antworte mit $\sigma' = (\sigma_{1,m'_1}, \ldots, \sigma_{\ell,m'_\ell})$ falls $m'_i \neq b$. Sonst Abbruch.

AUSGABE: $= \begin{cases} x_i & \text{falls } m_i = b \\ \text{Abbruch} & \text{sonst} \end{cases}$.

Sicherheit von Lamport Einwegsignaturen

Beweis: Fortsetzung

- Sei (m, σ) eine gültige Signatur mit $m_i = b$.
- Dann ist σ_i ein Urbild von y, d.h. $f(\sigma_i) = y$.
- Wahl von y im Invertier-Spiel erfolgt durch $x \in_R D$ und $y := f(x)$.
- D.h. pk ist identisch verteilt zum Lamport Signaturverfahren.

- Benötigen $m'_i \neq b$ und $m_i = b \neq m_i$. Es gilt $\mathrm{Ws}[m'_i \neq b] = \frac{1}{2}$.
- Wegen $m \neq m'$ folgt $\mathrm{Ws}[m'_i \neq m_i] \geq \frac{1}{\ell}$.
- Wir erhalten insgesamt $\mathrm{Ws}[\mathit{Invert}_{\mathcal{A},f}(n) = 1]$

$$\begin{aligned} &= \mathrm{Ws}[\mathit{Forge}^{\mathit{einweg}}_{\mathcal{A},\Pi}(n) \wedge (m'_i \neq b) \wedge (m'_i \neq m_i)] \\ &= \mathrm{Ws}[\mathit{Forge}^{\mathit{einweg}}_{\mathcal{A},\Pi}(n)] \cdot \mathrm{Ws}[(m'_i \neq b)] \cdot \mathrm{Ws}[(m'_i \neq m_i)] \geq \epsilon(n) \cdot \frac{1}{2\ell} \end{aligned}$$

- Aufgrund der Einweg-Eigenschaft von f gilt

$$\mathrm{negl}(n) \geq \mathrm{Ws}[\mathit{Invert}_{\mathcal{A},f}(n) = 1].$$

- Daraus folgt $\epsilon(n) \leq 2\ell \cdot \mathrm{negl}(n)$.
- Dies ist vernachlässigbar für polynomielles ℓ.

Einwegsignaturen für Nachrichten beliebiger Länge

Satz Einwegsignaturen für Nachrichten beliebiger Länge

Unter der Annahme kollisionsresistenter Hashfunktionen existiert ein CMA-sicheres Einwegsignatur-Verfahren für Nachrichten beliebiger Länge.

Beweis:

- Aus der Existenz von kollisionsresistenten Hashfunktionen folgt die Existenz von Einwegfunktionen. (Übung, siehe auch Krypto I)
- Nutzen Einwegfunktion f zur Konstruktion von CMA-sicheren Lamport-Signaturen der Nachrichtenlänge ℓ.
- Nutzen Hash-and-Sign Paradigma für beliebige Nachrichtenlänge. Hier verwenden wir erneut kollisionsresistente Hashfunktionen.

Einfache k-weg Signaturen

k-weg Signaturen

- Definiere mittels k-maliger Anwendung von $Gen_{\text{Lamport}}(1^n)$

$$pk = (pk_1, \ldots, pk_k) \text{ und } sk = (sk_1, \ldots, sk_k).$$

- Unterzeichnen die i-te Nachricht m mittels sk_i als (m, σ).
- Man bezeichnet i auch als *Zustand* im Signaturverfahren.
- (m, σ) gilt als gültig, falls (m, σ) für ein pk_i gültig ist.

Nachteile:

- Anzahl k muss bei Schlüsselgenerierung feststehen.
- Länge von pk und sk hängen beide von k ab.

Idee:

- Konstruiere neues Schlüsselpaar nur bei Bedarf.
- Validiere (pk_{i+1}, sk_{i+1}) mittels (pk_i, sk_i).

Zwei Signaturen mit einem öffentlichen Schlüssel

- Sei $\Pi = (Gen, Sign, Vrfy)$ ein Einwegsignaturverfahren.
- Konstruieren $\Pi' = (Gen', Sign', Vrfy')$ für zwei Signaturen.

Gen':

- Erzeuge $(pk_1, sk_1) \leftarrow Gen(1^n)$.

Sign' und Vrfy' von m_1:

- Erzeuge $(pk_2, sk_2) \leftarrow Gen(1^n)$. Berechne $\sigma_1' \leftarrow Sign_{sk_1}(m_1||pk_2)$.
- Ausgabe der Signatur $\sigma_1 = (pk_2, \sigma_1')$. Merke (m_1, σ_1, sk_2).
- Verifikation von $(m_1, \sigma_1) = (m_1, pk_2, Sign_{sk_1}(m_1||pk_2))$:

$$\text{Überprüfe } Vrfy_{pk_1}(m_1||pk_2, \sigma_1') \stackrel{?}{=} 1.$$

Sign' und Vrfy' von m_2:

- Erzeuge $(pk_3, sk_3) \leftarrow Gen(1^n)$. Berechne $\sigma_2' \leftarrow Sign_{sk_2}(m_2||pk_3)$.
- Ausgabe $\sigma_2 = (m_1, \sigma_1, pk_3, \sigma_2')$. Merke (m_2, σ_2, sk_3)
- Verifikation von $(m_2, \sigma_2) = (m_2, m_1, pk_2, \sigma_1', pk_3, \sigma_2')$:

$$\text{Überprüfe } Vrfy_{pk_1}(m_1||pk_2, \sigma_1') \stackrel{?}{=} 1 \textbf{ und } Vrfy_{pk_2}(m_2||pk_3, \sigma_2') \stackrel{?}{=} 1.$$

Beliebige Anzahl von Signaturen

Algorithmus Signaturketten

Sei $\Pi = (Gen, Sign, Vrfy)$ ein Einwegsignaturverfahren.

1. **Gen':** $(pk_1, sk_1) \leftarrow Gen(1^n)$
2. **Sign':** Signieren der Nachricht m_i.
   - Verwende gemerkten Zustand $(m_{i-1}, \sigma_{i-1}, sk_i)$.
   - $(pk_{i+1}, sk_{i+1}) \leftarrow Gen(1^n)$. Berechne $\sigma_i' = Sign_{sk_i}(m_i || pk_{i+1})$.
   - Ausgabe $\sigma_i = (m_{i-1}, \sigma_{i-1}, pk_{i+1}, \sigma_i')$. Merke $(m_i, \sigma_i, sk_{i+1})$.
3. **Vrfy':** Verifikation von $(m_i, \sigma_i) \overset{Sortieren}{=} (m_j, pk_{j+1}, \sigma_j')_{j=1}^{i}$:

   Überprüfe $Vrfy_{pk_j}(m_j || pk_{j+1}, \sigma_j') \overset{?}{=} 1$ für $j = 1, \ldots, i$.

Vorteil: Ein öffentlicher Schlüssel pk_1, beliebige Signaturanzahl.

Nachteile:

- Signaturlänge, Zustandsgröße und Verifikationszeit sind linear in der Anzahl der signierten Dokumente.
- Signaturen geben alle zuvor signierten Dokumente preis.

Merkle-Baum

Idee: Konstruktion von Merkle-Bäumen

- Ersetze Signaturkette durch Baum (sogenannter Merkle-Baum).
- Verwenden Baum der Tiefe n für Nachrichten der Länge n.
- Die Wurzel erhält Label ϵ.
- Die Kinder eines Knotens mit Label w erhalten Label $w0$ und $w1$.
- Blätter besitzen Nachrichten-Label $m \in \{0,1\}^n$.
- Knoten mit Label w speichern Schlüsselpaar pk_w, sk_w.
- Wurzelschlüssel pk_ϵ ist der öffentliche Schlüssel.

Ziel: Zertifiziere Pfad von Wurzel zu m mittels Signaturen.

- Signieren Public-Keys auf Pfad inklusive der Nachbarknoten.

Signieren und Verifizieren von $m = 001$

Signieren von $m = 001$

- Pfad von Wurzel zu m: $\epsilon, 0, 00, 001$ mit Nachbarknoten $1, 01, 000$.
- Signiere $(\mathbf{pk_0}||pk_1)$ mittels sk_ϵ. Sei dies σ_ϵ.
- Signiere $(\mathbf{pk_{00}}||pk_{01})$ mittels sk_0. Sei dies σ_0.
- Signiere $(pk_{000}||\mathbf{pk_{001}})$ mittels sk_{00}. Sei dies σ_{00}.
- Signiere m mittels sk_{001}. Sei dies σ_{001}.
- Signatur ist $\sigma = (pk_0||pk_1, \sigma_\epsilon, pk_{00}||pk_{01}, \sigma_0, pk_{000}||pk_{001}, \sigma_{00}, \sigma_{001})$

Verifizieren von (m, σ):

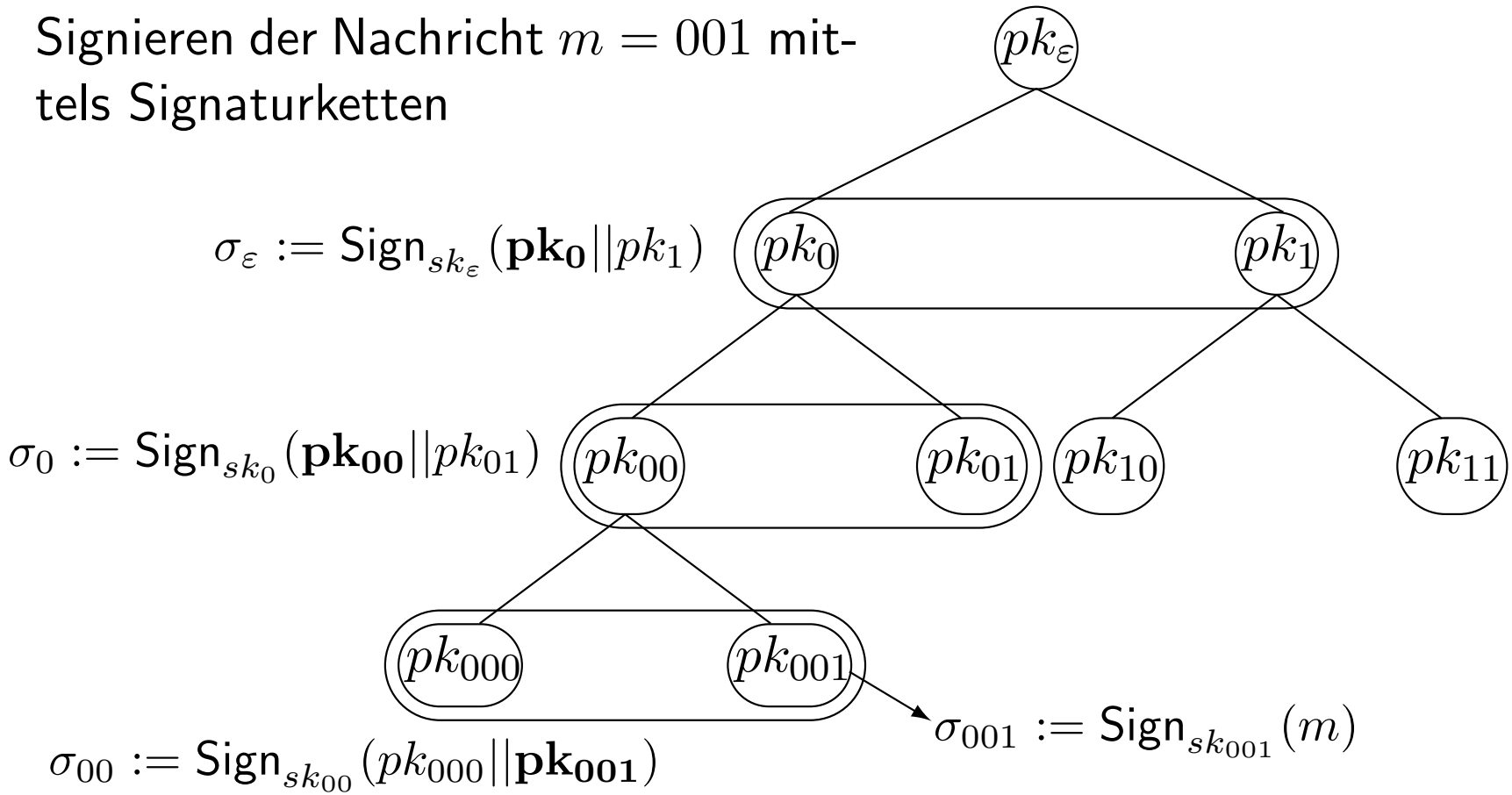
- Verifiziere $(pk_0||pk_1, \sigma_\epsilon)$ mittels pk_ϵ.
- Verifiziere $(pk_{00}||pk_{01}, \sigma_0)$ mittels pk_0.
- Verifiziere $(pk_{000}||pk_{001}, \sigma_{00})$ mittels pk_{00}.
- Verifiziere $(001, \sigma_{001})$ mittels pk_{001}.

Notation:

- Sei $m \in \{0,1\}^n$. Wir definieren $m|_i := m_1 \dots m_i$ für $i = 0, \dots, n$.
- D.h. $m|_i$ ist der i-Zeichen Präfix von m und $m|_i = \epsilon$.

Signaturpfad der Nachricht $m = 001$

Signieren der Nachricht $m = 001$ mittels Signaturketten



Merkle Signaturen

Algorithmus Merkle Signatur

Sei $\Pi' = (Gen', Vrfy', Sign')$ ein Einwegsignaturverfahren.

1. **Gen:** $(pk_\epsilon, sk_\epsilon) \leftarrow Gen'(1^n)$
2. **Sign:** Für Nachricht $m \in \{0,1\}^n$: FOR $i := 0$ to $n-1$
   - Falls $pk_{m|_i 0}, pk_{m|_i 1}$ noch nicht erzeugt, erzeuge und speichere $(pk_{m|_i 0}, sk_{m|_i 0}) \leftarrow Gen'(1^n)$, $(pk_{m|_i 1}, sk_{m|_i 1}) \leftarrow Gen'(1^n)$.
   - Erzeuge $\sigma_{m|_i} \leftarrow Sign'_{sk_{m|_i}}(pk_{m|_i 0}, pk_{m|_i 1})$.

   Berechne $\sigma_m \leftarrow Sign'_{sk_m}(m)$

   Ausgabe von $\sigma = ((pk_{m|_i 0} || pk_{m|_i 1}, \sigma_{m|_i})_{i=0}^{n-1}, \sigma_m)$.
3. **Vrfy':** Für (m, σ) überprüfe

   $Vrfy'_{pk_{m|_i}}(pk_{m|_i 0} || pk_{m|_i 1}, \sigma_{m|_i}) \stackrel{?}{=} 1$ für $i = 0, \ldots, n-1$ und

   $Vrfy'_{pk_m}(m, \sigma_m) \stackrel{?}{=} 1$.

Eigenschaften von Merkle Signaturen

Eigenschaften:

- Erlaubt das Signieren aller möglichen 2^n Nachrichten.
- Schlüsselpaare werden nur bei Bedarf erzeugt.

Vorteile: gegenüber Signaturketten

- Signaturlänge/Verifikationszeit sind linear in der Nachrichtenlänge aber unabhängig von der Anzahl Signaturen.
- Keine Preisgabe der zuvor signierten Nachrichten.

Satz Sicherheit von Merkle Signaturen

Sei Π' eine CMA-sichere Einwegsignatur. Dann sind Merkle Signaturen Π ein CMA-sicheres Signaturverfahren.

Beweis:

- Sei $\mathcal{A}$ ein Angreifer mit $\epsilon(n) := \mathrm{Ws}[Forge_{A,\Pi}(n) = 1]$.
- $\mathcal{A}'$ frage höchstens q Signaturen an. Setze $\ell := 2nq + 1$ als obere Schranke für die Anzahl der benötigten Schlüssel in Π.
- Wir konstruieren mittels $\mathcal{A}$ einen Angreifer $\mathcal{A}'$ für Π'.