Sicherheit von Merkle Signaturen

Algorithmus Angreifer $\mathcal{A}$ für die Einwegsignatur

EINGABE: pk, Zugriff auf eine Anfrage an Orakel $Sign_{sk}(\cdot)$

1. Berechne $(pk^{(i)}, sk^{(i)}) \leftarrow Gen(1^n)$ für $i = 1, \ldots, \ell$.
   Wähle $i' \in_R [\ell]$. Ersetze $pk^{(i')}$ durch pk. $j \leftarrow 2$
2. $(m, \sigma') \leftarrow \mathcal{A}'(pk^{(1)})$. Signaturanfragen für m: For $i \leftarrow 1$ to $n-1$
   - Falls $pk_{m|_i 0}, pk_{m|_i 1}$ undefiniert, $(pk_{m|_i 0}, pk_{m|_i 1}) \leftarrow (pk^{(j)}, pk^{(j+1)})$.
     $j \leftarrow j + 2$
   - Berechne $\sigma_{m|_i}$, σ_m und σ analog zu Merkle-Signaturen.
     Falls $sk = sk^{(i')}$ benötigt, verwende das Signierorakel $Sign_{sk}(\cdot)$.
3. Sei $\sigma' = ((pk'_{m|_i 0} || pk'_{m|_i 1}, \sigma'_{m|_i})_{i=0}^{n-1}, \sigma'_m)$
   - **Fall 1:** $\exists 0 \leq i < n$ mit $(pk'_{m|_i 0} || pk'_{m|_i 1}) \neq (pk_{m|_i 0} || pk_{m|_i 1})$.
     Sei i minimal. Dann gilt $pk'_{m|_i} = pk_{m|_i} = pk^{(k)}$ für ein $k \in [\ell]$.
     Falls $k = i'$, Ausgabe $(pk'_{m|_i 0} || pk'_{m|_i 1}, \sigma'_{m|_j})$.
   - **Fall 2:** Es gilt $pk'_m = pk_m = pk^{(k)}$ für ein $k \in [\ell]$.
     Falls $k = i'$, Ausgabe (m, σ'_m).

Sicherheit von Merkle Signaturen

Beweis: Fortsetzung

- Verteilung der Nachrichten für $\mathcal{A}'$ ist identisch zum Forge-Spiel.
- D.h. $\mathcal{A}$ liefert eine gültige Signatur (m', σ') mit Ws $\epsilon(n)$.
- Sowohl für Fall 1 als auch für Fall 2 gilt $\mathrm{Ws}[k = i'] = \frac{1}{\ell}$.
- Wir nehmen im folgenden an, dass $k = i'$.
- **Fall 1:** $\exists$ neuer Public-Key in Geschwisterknotenpaar.
- $\mathcal{A}$ stellte eventuell Orakelanfrage für Nachricht $(pk_{m|_i 0}||pk_{m|_i 1})$.
- Wegen $(pk'_{m|_i 0}||pk'_{m|_i 1}) \neq (pk_{m|_i 0}||pk_{m|_i 1})$ ist $\sigma'_{m|_i}$ bezüglich pk eine gültige Signatur für eine neue Nachricht.
- **Fall 2:** pk'_m existiert bereits.
- $\mathcal{A}'$ kann nicht Orakelanfrage m gestellt haben, da er m ausgibt.
- Damit ist σ'_m eine gültige neue Signatur für m bezüglich pk.
- **Insgesamt:** $\mathrm{negl}(n) \geq \mathrm{Ws}[Forge^{einweg}_{\mathcal{A},\Pi}(n) = 1] = \frac{\epsilon(n)}{\ell}$.
- Da ℓ polynomiell ist, folgt $\epsilon(n) \leq \mathrm{negl}(n)$.

Existenz CMA-sicherer Signatur

Korollar Signatursatz

Falls kollisionsresistente Hashfunktionen existieren, so existiert ein CMA-sicheres Signaturverfahren.

Beweis:

- Unter der Annahme kollisionsresistenter Hashfunktionen existieren CMA-sichere Einwegsignaturen.
- Aus CMA-sicheren Einwegsignaturen können CMA-sichere Signaturen konstruiert werden.

Anmerkung:

- Man kann sogar zeigen, dass ein CMA-sicheres Signaturverfahren existiert unter der Annahme der Existenz von Einwegfunktionen.

Digital Signature Standard – Schnorr Signaturen

Systemparameter:

- Primzahlen p, q, wobei q Bitlänge n besitzt und $q|p-1$, $q^2 \nmid p-1$.
- Generator g einer Untergruppe von $\mathbb{Z}_p^*$ mit Ordnung q.

Algorithmus Digital Signature Standard

1. **Gen:** $(p, q, g, H) \leftarrow Gen(1^n)$ mit Hashfunktion $H : \{0,1\}^* \to \mathbb{Z}_q$.
   Wähle $x \in_R \mathbb{Z}_q$, berechne $y := g^x \bmod p$.
   Setze $pk = (p, q, g, H, y)$, $sk = (p, q, g, H, x)$.
2. **Sign:** Für $m \in \{0,1\}^*$, wähle $k \in_R \mathbb{Z}_q^*$ und berechne
   $$r := (g^k \bmod p) \bmod q \quad \text{und} \quad s := (H(m) + xr) \cdot k^{-1} \bmod q.$$
   Signatur $\sigma = (r, s)$.
3. **Vrfy:** Für $(m, \sigma) = (m, r, s)$ überprüfe
   $$r \stackrel{?}{=} (g^{H(m) \cdot s^{-1} \bmod q} y^{r \cdot s^{-1} \bmod q} \bmod p) \bmod q.$$

Eigenschaften des DSS

Korrektheit:

$$\begin{aligned} g^{H(m)\cdot s^{-1}} y^{r\cdot s^{-1}} &= g^{H(m)(H(m)+xr)^{-1}k} g^{xr(H(m)+xr)^{-1}k} \bmod p \\ &= g^{(H(m)+xr)\cdot(H(m)+xr)^{-1}k} = g^k \bmod p. \end{aligned}$$

Parameterwahl:

- Bitlänge von p: 1024, Bitlänge n von q: 160.
- Die Signaturlänge von $(r,s) \in \mathbb{Z}_q^2$ ist damit nur 320 Bit.
- Dlog in $\mathbb{Z}_p^*$: subexponentieller Index-Calculus Algorithmus
- Dlog in $\langle g \rangle$: Pollard-Rho mit Komplexität $\mathcal{O}(2^{\frac{n}{2}})$.

Sicherheit:

- Keine größeren Schwächen bekannt.
- Aber: DSS besitzt **keinen** Sicherheitsbeweis.

Viele Public-Keys mittels eines Public Keys

Zertifierung

- Zertifizierungsstelle CA (Certificate Authority) veröffentlicht pk_{CA}.
- CA zertifiziert Schlüssel pk_A eines Nutzers Alice mit Zertifikat

$$cert_{CA \to A} \leftarrow Sign_{sk_{CA}}(pk_A).$$

- Alice kann $(pk_A, cert_{CA \to A})$ über unsicheren Kanal verschicken.
- CMA-Sicherheit des Signaturverfahrens verhindert erfolgreiches Fälschen eines Zertifikat für einen anderen Schlüssel pk'_A.
- D.h. mit nur einem öffentlichen Schlüssel kann eine CA beliebig viele weitere öffentliche Schlüssel zertifizieren.
- Liefert sogenannte Public-Key Infrastruktur.

Übersicht Krypto I

Abkürzungen:

- PRNG = Pseudozufallsgenerator
- PRF = Pseudozufallsfunktion.

Funktionalität	**Sicherh.**	**Konstrukt**	**Annahme**
One-Time Pad Verschlüsselung	perfekt	$m \oplus k$	keine
Stromchiffre	KPA	$m \oplus G(k)$	PRNG
Blockchiffre (CBC, OFB, CTR)	CPA	$(r, m \oplus F_k(r))$	PRF
MAC	unfälsch-bar	$F_k(m)$	PRF
Encrypt-then-Authenticate	CCA	$(c,t) = (Enc_{k_1}(m), Mac_{k_2}(c))$	PRF

Übersicht Krypto II - Verschlüsselung

Abkürzung: TD-OWP = Trapdoor-Einwegpermutation

Funktionalität	**Konstrukt**	**Annahme**
Schlüsselaustausch	sicher	Unterscheidungsannahme
Diffie-Hellman	g^{xy}	*DDH:* $g^{xy} \leftrightarrow g^z$
PK Verschlüsselung	CPA	TD-OWP
RSA	$(r^e, hc(r) \oplus m)$	*RSA-Annahme*
Rabin	$(x^2, lsb(x) \oplus m)$	*Faktorisierungsannahme*
PK Verschlüsselung	CPA (ROM)	TD-OWP
ROM-RSA	$(r^e, H(r) \oplus m)$	*RSA-Annahme*
PK Verschlüsselung	CCA (ROM)	TD-OWP + PRF
ROM-RSA2	$(r^e, Enc'_{H(r)}(m))$	*RSA-Annahme*
PK Verschlüsselung	CPA	Unterscheidungsannahme
ElGamal	$(g^y, g^{xy} \cdot m)$	*DDH:* $g^{xy} \leftrightarrow g^z$
Goldwasser-Micali	$z^m \cdot x^2$	*QR:* $QR_N \leftrightarrow QNR_N^{+1}$
Paillier	$(1+N)^m \cdot r^N$	*DCR:* $r^N \bmod N^2 \leftrightarrow r$