Vorlesung Sicherheit

Dennis Hofheinz

ITI, KIT

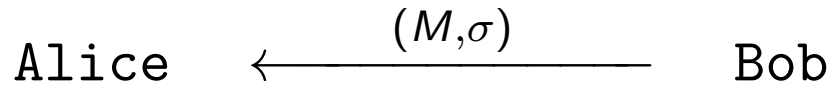
19.05.2014

Überblick

Überblick

Ziel

- Authentifizierte Übermittlung auf unauthentifiziertem Kanal:



- Nachricht M soll vor Veränderungen geschützt werden
- Idee: Sende „Unterschrift“ σ mit Nachricht
- Anforderungen:
  - `Bob` muss σ (aus/für Nachricht M) berechnen können
  - `Alice` muss σ (zusammen mit M) verifizieren können
  - Außenseiter soll kein gültiges σ für neues M erzeugen können

Grundidee MACs

- Annahme: `Alice` und `Bob` besitzen gemeinsames Geheimnis K

$$\texttt{Alice}_K \xleftarrow{(M,\sigma)} \texttt{Bob}_K$$

- Signieren: $\sigma \leftarrow \mathrm{Sig}(K, M)$
- Verifizieren: $\mathrm{Ver}(K, M, \sigma) \in \{0, 1\}$
- Korrektheit: $\mathrm{Ver}(K, M, \sigma) = 1$ für alle K, M und $\sigma \leftarrow \mathrm{Sig}(K, M)$
- Wird „MAC“ (Message Authentication Code) genannt

Überblick

Sicherheitsmodell

- **Diskussion:** Wünschenswerte Sicherheitseigenschaften?

Sicherheitsdefinition

- Schema EUF-CMA-sicher $\Leftrightarrow$ kein PPT-Angreifer $\mathcal{A}$ gewinnt folgendes Spiel nicht-vernachlässigbar oft:
  1. $\mathcal{A}$ erhält Zugriff auf ein $\mathrm{Sig}(K, \cdot)$-Orakel
  2. $\mathcal{A}$ gibt Ausgabe (M^*, σ^*)
  3. $\mathcal{A}$ gewinnt, wenn $\mathrm{Ver}(K, M^*, \sigma^*) = 1$ und M^* „frisch“
- Modelliert passive Angriffe ($\mathcal{A}$ erhält keinen Ver-Zugriff)
  - Für viele Verfahren (z.B. bei eindeutigem σ) äquivalent zu Definition mit Ver-Orakel für $\mathcal{A}$
  - Intuition: wenn $\mathcal{A}$ Ver-Anfrage mit $\mathrm{Ver}(K, M, \sigma) = 1$ und „frischem“ (also nicht schon von Sig erzeugtem) σ generiert, ist das schon eine gefälschte Signatur

Überblick

Hash-Then-Sign-Paradigma

- **Problem:** viele Verfahren signieren nur kurze Bitstrings
- **Lösung:** signiere $\mathrm{H}(M) \in \{0,1\}^k$ anstelle von $M \in \{0,1\}^*$

Theorem (Sicherheit des Hash-Then-Sign-Paradigmas)

Sei $(\mathrm{Sig}, \mathrm{Ver})$ *EUF-CMA-sicher und* H *eine kollisionsresistente Hashfunktion. Dann ist der durch* $\mathrm{Sig}'(K, M) = \mathrm{Sig}(K, \mathrm{H}(M))$, $\mathrm{Ver}'(K, M, \sigma) = \mathrm{Ver}(K, \mathrm{H}(M), \sigma)$ *erklärte MAC EUF-CMA-sicher.*

Beweis.

Beweisstrategie: ein EUF-CMA-Angreifer $\mathcal{A}'$ auf $(\mathrm{Sig}', \mathrm{Ver}')$ muss entweder eine H-Kollision oder eine Signatur σ für einen „frischen“ Hashwert $\mathrm{H}(M)$ finden, um das EUF-CMA-Spiel zu gewinnen. □

Pseudorandom Functions

- Nützlicher theoretischer Baustein: Pseudorandom Functions

Definition (Pseudorandom Function, PRF)

Sei $\mathrm{PRF} : \{0,1\}^k \times \{0,1\}^k \to \{0,1\}^k$ eine über $k \in \mathbb{N}$ parametrisierte Funktion. PRF heißt Pseudorandom Function (PRF), falls für jeden PPT-Algorithmus $\mathcal{A}$ die Funktion

$$\mathrm{Adv}^{\mathrm{prf}}_{\mathrm{PRF},\mathcal{A}}(k) := \Pr\left[\mathcal{A}^{\mathrm{PRF}(K,\cdot)}(1^k) = 1\right] - \Pr\left[\mathcal{A}^{\mathrm{R}(\cdot)}(1^k) = 1\right]$$

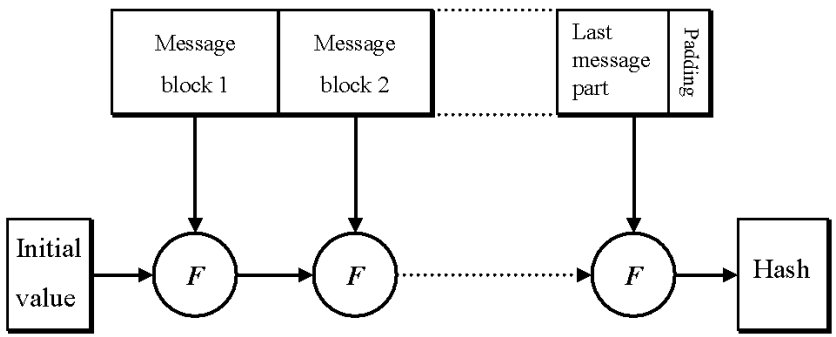
vernachlässigbar ist, wobei $\mathrm{R} : \{0,1\}^k \to \{0,1\}^k$ eine echt zufällige Funktion ist.

Kandidat für eine Pseudorandom Function

- PRF-Kandidat, ausgehend von Hashfunktion H:

$$\mathrm{PRF}(K, X) := \mathrm{H}(K, X)$$

- **Vorsicht:** diese Konstruktion hat ihre Tücken
  - Manchmal (Merkle-Damgård) lässt sich Hashwert „erweitern“:



  - $\mathrm{H}(K, X)$ kann zu $\mathrm{H}(K, X, X')$ erweitert werden
  - Bricht PRF-Eigenschaft für Eingaben unterschiedlicher Länge

Überblick

Pseudorandom Functions und MACs

Theorem (MACs aus PRFs und Hashfunktionen)

Sei $\mathrm{PRF} : \{0,1\}^k \times \{0,1\}^k \to \{0,1\}^k$ *eine PRF und* $\mathrm{H} : \{0,1\}^* \to \{0,1\}^k$ *eine kollisionsresistente Hashfunktion. Dann ist der durch* $\mathrm{Sig}(K, M) = \mathrm{PRF}(K, \mathrm{H}(M))$ *gegebene MAC EUF-CMA-sicher.*

Beweis.

- Gehen wir von einem erfolgreichen EUF-CMA-Angreifer $\mathcal{A}$ aus
- Wir dürfen annehmen, dass $\mathcal{A}$ eine Fälschung (M^*, σ^*) mit „frischem“ (d.h. noch nicht signiertem) M^* ausgibt
- $\mathcal{A}$ kann also als PRF-Unterscheider aufgefasst werden, der mit nicht-vernachlässigbarer Wkt. $\mathrm{PRF}(K, \mathrm{H}(M^*))$ vorhersagt
- Derartige Voraussage nur mit PRF (nicht aber bei R) möglich

□

Überblick

Konkrete Konstruktionen

- Unsicher (für $\mathrm{H} \in \{\text{MD5}, \text{SHA-1}\}$):

$$\mathrm{Sig}(K, M) = \mathrm{H}(K, M)$$

- Besser, gerade hergeleitet:

$$\mathrm{Sig}(K, M) = \mathrm{H}(K, \mathrm{H}(M))$$

- Noch besser, sehr populär: HMAC

$$\mathrm{Sig}(K, M) = \mathrm{H}(K \oplus \mathrm{opad}, \mathrm{H}(K \oplus \mathrm{ipad}, M))$$

($\mathrm{opad}, \mathrm{ipad} \in \{0,1\}^k$ feste Konstanten)

HMAC

- Erinnerung: HMAC

$$\mathrm{Sig}(K, M) = \mathrm{H}(K \oplus \mathrm{opad}, \mathrm{H}(K \oplus \mathrm{ipad}, M))$$

- Vorteil (gegenüber $\mathrm{Sig}(K, M) = \mathrm{H}(K, \mathrm{H}(M))$):
  - Zusätzliche Parametrisierung von innerer H-Auswertung erschwert konkrete Angriffe (heuristisches Argument)
  - H-Kollisionen führen noch nicht notwendig zu Sig-Bruch
- Theoretisch und praktisch gut untersucht, de-facto-Standard

Überblick

Zusammenfassung

- MACs stellen Nachrichtenintegrität/-authentizität sicher
- Standard-Sicherheitsdefinition EUF-CMA
- Hash-Then-Sign-Paradigma
- Hashfunktionen als PRFs als MACs
  - Aber: Merkle-Damgård hat hier seine Tücken
- Aktueller Standard: HMAC

Aktuelle Forschung

- Aktuelle Debatte: (Un-)Sinn verschiedener Angreifermodelle
  - Wenn wir über alle (effizienten) Angreifer quantifizieren...
  - ...kennt ein Angreifer auch eine Hashkollision...
  - ...und kann damit z.B. Hash-Then-Sign-Verfahren brechen
  - Deshalb sollte Angreifer für jedes k funktionieren
  - Welchen Sinn haben dann noch „konkrete“ Sicherheitsbeweise?

Überblick

Überblick

Motivation

- Authentifizierte Übermittlung auf unauthentifiziertem Kanal:

$$\texttt{Alice} \quad \xleftarrow{(M,\sigma)} \quad \texttt{Bob}$$

- MACs: `Alice` und `Bob` besitzen gemeinsames Geheimnis K

$$\texttt{Alice}_K \quad \xleftarrow{(M,\sigma)} \quad \texttt{Bob}_K$$

- **Probleme:** Schlüsselverteilung, viele Schlüssel nötig

Motivation

- Alternative: digitale Signaturschemata

$$\texttt{Alice}_{pk} \quad \xleftarrow{(M,\sigma)} \quad \texttt{Bob}_{sk}$$

- $(pk, sk) \leftarrow \mathrm{Gen}(1^k)$ wie bei Public-Key-Verschlüsselung
  - pk öffentlicher Schlüssel
  - sk geheimer Schlüssel
- Signieren: $\sigma \leftarrow \mathrm{Sig}(sk, M)$
- Verifizieren: $\mathrm{Ver}(pk, M, \sigma) \in \{0, 1\}$
- Korrektheit wie bei MACs: $\mathrm{Ver}(pk, M, \sigma) \stackrel{!}{=} 1$ für alle $(pk, sk) \leftarrow \mathrm{Gen}(1^k)$, alle M und alle $\sigma = \mathrm{Sig}(sk, M)$

Überblick

Sicherheitsdefinition

- Schema EUF-CMA-sicher $\Leftrightarrow$ kein PPT-Angreifer $\mathcal{A}$ gewinnt folgendes Spiel nicht-vernachlässigbar oft:
  1. $\mathcal{A}$ erhält pk und Zugriff auf ein $\mathrm{Sig}(sk, \cdot)$-Orakel
  2. $\mathcal{A}$ gibt Ausgabe (M^*, σ^*)
  3. $\mathcal{A}$ gewinnt, wenn $\mathrm{Ver}(pk, M^*, \sigma^*) = 1$ und M^* „frisch“
- Standardbegriff für digitale Signaturen

Überblick

RSA als Signaturschema

- Erinnerung RSA-Verschlüsselung:

$$pk = (N, e) \qquad sk = (N, d)$$
$$\mathrm{Enc}(pk, M) = M^e \bmod N$$
$$\mathrm{Dec}(sk, C) = C^d \bmod N$$

- (**Warnung/Erinnerung:** in dieser Form nicht sicher!)
- Betrachte RSA als Signaturschema:

$$\mathrm{Sig}(sk, M) = M^d \bmod N$$
$$\mathrm{Ver}(pk, M, \sigma) = 1 \quad :\Longleftrightarrow \quad M = \sigma^e \bmod N$$

Eigenschaften von RSA

- RSA als Signaturschema:

$$\mathrm{Sig}(sk, M) = M^d \bmod N$$
$$\mathrm{Ver}(pk, M, \sigma) = 1 \quad :\Longleftrightarrow \quad M = \sigma^e \bmod N$$

- PKE$\rightarrow$Sig-Konversion nicht allgemein
  - Allgemeiner lassen sich Nachrichten nicht unbedingt „zuerst ent-, dann wieder verschlüsseln“ (Datentypproblem)
  - Außerdem muss/sollte Enc nicht deterministisch sein
- Zahlreiche Sicherheitsprobleme (nachfolgend)
- Aber: RSA-Signaturen können „repariert“ werden

Probleme von RSA-Signaturen

$$\mathrm{Sig}(sk, M) = M^d \bmod N$$
$$\mathrm{Ver}(pk, M, \sigma) = 1 \quad :\Longleftrightarrow \quad M = \sigma^e \bmod N$$

- Problem: unsinnige Nachrichten können signiert werden
  1. Wähle *zuerst* Signatur $\sigma \in \mathbb{Z}_N$ beliebig
  2. Setze dann $M := \sigma^e \bmod N$
  3. Damit ist σ gültige RSA-Signatur für M
- Bricht EUF-CMA-Sicherheit, (künstliche) problematische Anwendungen denkbar

Probleme von RSA-Signaturen

$$\mathrm{Sig}(sk, M) = M^d \bmod N$$
$$\mathrm{Ver}(pk, M, \sigma) = 1 \quad :\Longleftrightarrow \quad M = \sigma^e \bmod N$$

- Weiteres Problem: Homomorphie von RSA
  1. Angenommen, $\sigma_i = M_i^d \bmod N$ bekannt (für einige i)
  2. Setze dann $\sigma := \prod_i \sigma_i = \prod_i M_i^d = (\prod_i M_i)^d \bmod N$
  3. Damit ist σ gültige RSA-Signatur für $M := \prod_i M_i$
- Neue Signaturen lassen sich aus bekannten berechnen
- Bricht auch EUF-CMA-Sicherheit, (leicht weniger künstliche) problematische Anwendungen denkbar

Lösungsvorschläge?

- **Diskussion:** Wie könnte man RSA-Signaturen „reparieren“?

RSA-PSS

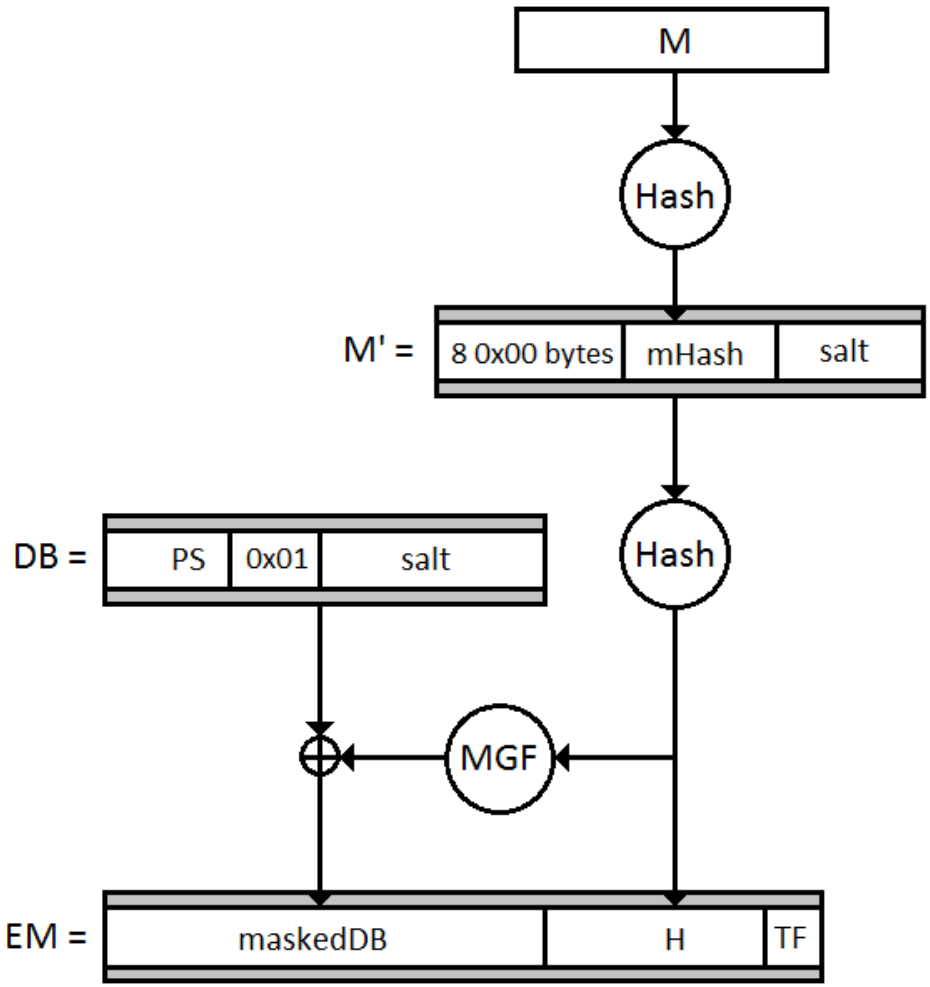
- (RSA-)PSS: „Probabilistic Signature Scheme“
- Idee von RSA-PSS: Vorverarbeitung (Padding) der Nachricht:

$$\mathrm{Sig}(sk, M) = (\mathrm{pad}(M))^d \bmod N$$

$$\mathrm{Ver}(pk, M, \sigma) = 1 \quad :\Longleftrightarrow \quad \sigma^e \bmod N \quad \text{gültiges } \mathrm{pad}(M)$$

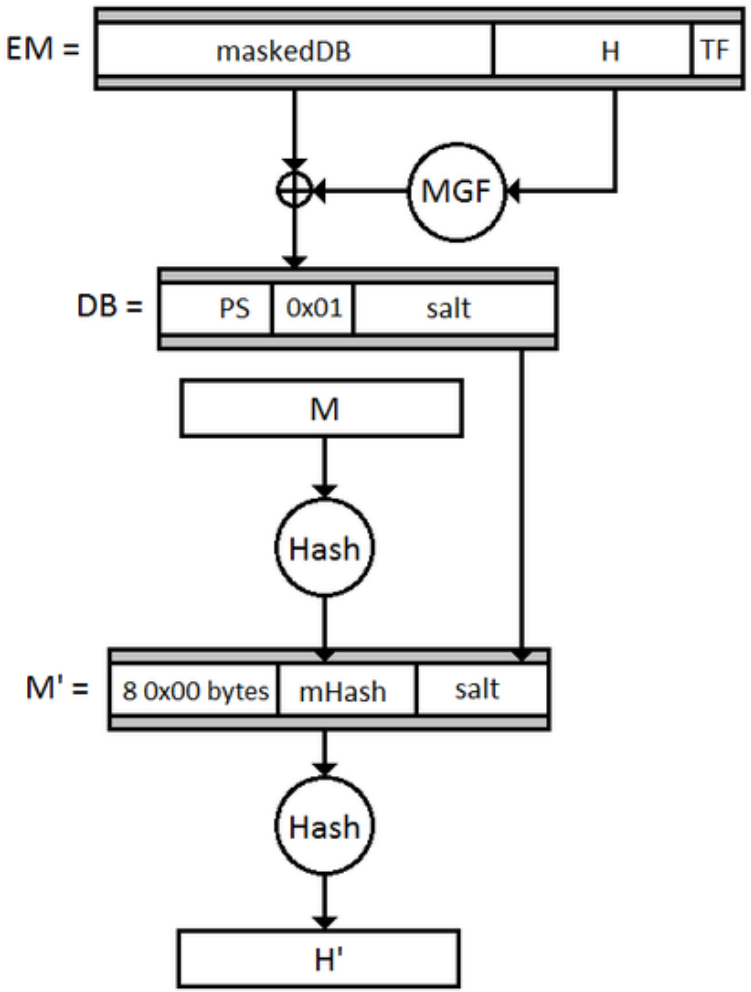
RSA-PSS

- Padding einer Nachricht: $\mathrm{pad}(M)$ (Quelle: Wikipedia)



RSA-PSS

- Verifikation einer gepaddeten Nachricht: (Quelle: Wikipedia)



Sicherheit von RSA-PSS

- RSA-PSS heuristisch (mit idealen H, MGF) EUF-CMA-sicher, sofern RSA-Funktion schwer zu invertieren
  - Jeder EUF-CMA-Angreifer *muss* RSA-Funktion invertieren
- RSA-PSS (wie RSA-OAEP) Teil von PKCS#1
- Bester bekannter Angriff: N faktorisieren (Zahlkörpersieb)
- Parameterwahl wie bei RSA-OAEP (somit $\log_2(N) \approx 2048$)
- Festes, kleines e (z.B. $e = 3$) möglich $\Rightarrow$ effiziente Verifikation