IT-Sicherheit

Hash-Funktionen, MACs und Authenticated Encryption

Prof. Dr. Dominik Merli, Prof. Dr. Lothar Braun

Sommersemester 2020

Hochschule Augsburg - Fakultät für Informatik

Hash-Funktionen

Allgemeine Eigenschaften

- Auch ”Streuwert-Funktion”
- Abbildung $h = \textit{hash}(m)$
  - Große/unbegrenzte Eingabemenge
  - Kleine/begrenzte Zielmenge
- Ergebnis h nennt man Hash-Wert (engl. hash value, digest)

Beispiele

- `echo "Max Mustermann erhält Note 1.0!" | sha256sum`
  - `69ab46da0962ed23a1ed973e55ea322b4aa7aae1948a5d93113ed01a90d956dd`
- `echo "Max Mustermann erhält Note 2.0!" | sha256sum`
  - `522a11f5d0928936ad544259afb1894779dce4bdf6be09ad57c592a7bec87a99`
- `echo "Max Mustermann erhält Note 1.0! " | sha256sum`
  - `59ba24cd4d05d5a82a494a895c25448c86eb038124e8e3d79442ca43a42978a9`

Hash-Funktionen haben keinen Schlüssel!

Kryptographische Eigenschaften

- Einweg-Eigenschaft (engl. preimage resistance, one-wayness)
  $\rightarrow$ Gegeben ein Hashwert h ist es praktisch unmöglich eine Nachricht m zu finden, für die gilt $h = hash(m)$

- Schwache Kollisionsresistenz (engl. second preimage resistance)
  $\rightarrow$ Gegeben eine Nachricht m ist es praktisch unmöglich eine Nachricht m' zu finden, für die gilt $h = hash(m) = hash(m')$

- Kollisionsresistenz (engl. collision resistance)
  $\rightarrow$ Es ist praktisch unmöglich zwei Nachrichten m und m' zu finden, für die gilt $h = hash(m) = hash(m')$

Birthday Paradox

- Wie viele Leute müssen auf einer Party sein, so dass mit hoher Wahrscheinlichkeit zwei am gleichen Tag des Jahres Geburtstag haben?
- Triviale Fälle
  - $P(\textit{Kollision bei 1 Person}) = 0$
  - $P(\textit{Kollision bei 366 Personen}) = 1$
- Weitere Überlegungen
  - $P(\textit{mind. eine Kollision}) = 1 - P(\textit{keine Kollision})$
  - $P(\textit{keine Kollision bei 2 Personen}) = (1 - \frac{1}{365})$
  - $P(\textit{keine Kollision bei 3 Personen}) = (1 - \frac{1}{365}) \cdot (1 - \frac{2}{365})$
  - $P(\textit{keine Kollision bei n Personen}) = (1 - \frac{1}{365}) \cdot (1 - \frac{2}{365}) ... (1 - \frac{n-1}{365})$
- Ergebnis
  - $P(\textit{mind. eine Kollision bei 23 Personen}) \approx 50\%$
  - $P(\textit{mind. eine Kollision bei 40 Personen}) \approx 90\%$

Kollisionen bei Hash-Funktionen

- Suchraumgröße für Kollisionen (aus ”Understanding Cryptography” von C. Paar und J. Pelzl)

Kollisions-wahrscheinlichkeit	Hash-Wert Länge				
	128-bit	160-bit	256-bit	384-bit	512-bit
0.5	2^{65}	2^{81}	2^{129}	2^{193}	2^{257}
0.9	2^{67}	2^{82}	2^{130}	2^{194}	2^{258}

- **Fazit:** Sicherheitsniveau von Hash-Funktionen nur ca. Hälfte der Ausgabe-Länge

Anwendungen

- Digitale Signaturen
- Message Authentication Codes (MACs)
- Schlüsselableitung (engl. key derivation)
- Sichere Passwort-Speicherung
- Und vieles mehr ...

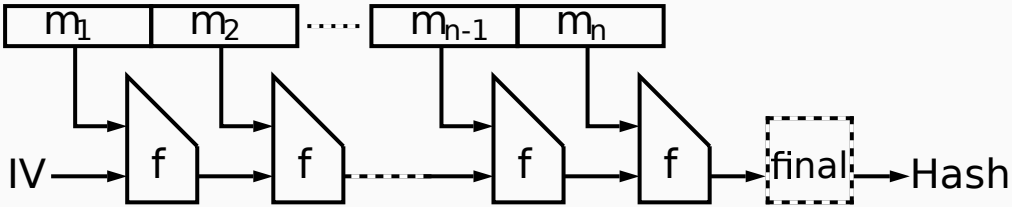
Geschichte

- 1970er Jahre: Erste kryptographische Hash-Funktionen
- 1990er Jahre: MD4, MD5, ...
- 1995: SHA-1 Publikation (Verbesserung von SHA-0)
- 2002: SHA-2 standardisiert
- 2015: SHA-3 Standard publiziert

- Gute geschichtliche Zusammenfassung:
  Bart Preneel: "The First 30 Years of Cryptographic Hash Functions and the NIST SHA-3 Competition", Springer, CT-RSA 2010.

Merkle-Damgard-Konstruktion

- Aufteilen und Padden der Nachricht → gleich lange Blöcke
- Kompressionsfunktion f verarbeitet vorheriges Ergebnis und neuen Block



Megatherium (Wikipedia), CC0

SHA-2

- Familie von Hash-Funktionen, benannt nach Output-Länge

- SHA-256
  - 256-bit Output
  - Arbeitet auf 32-bit Wörtern
  - Benötigt 64 Runden

- SHA-512
  - 512-bit Output
  - Arbeitet auf 64-bit Wörtern
  - Benötigt 80 Runden

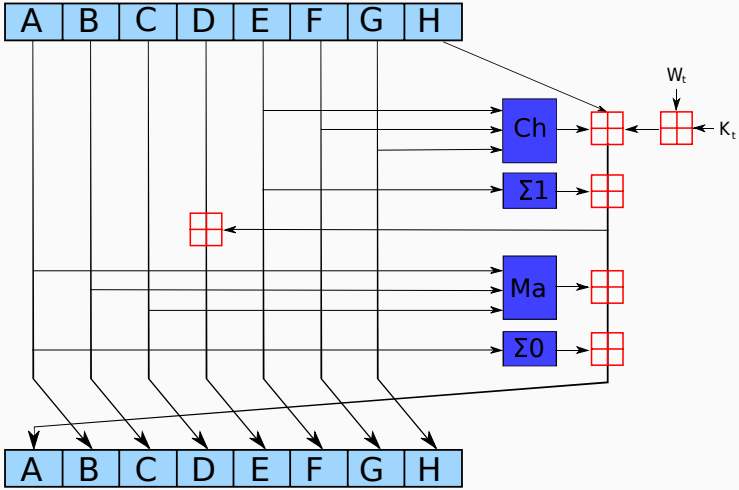
- Teilt Nachricht in 32-bit Wörter auf
- Verarbeitet 64 Wörter mit jeweils 32-bit in einem Durchlauf
- Nutzt acht 32-bit Wörter als Arbeitsvariablen
- Ablauf
  1) Vorbereitung der Nachrichten-Blöcke
  2) Initialisierung der acht Arbeitsvariablen
  3) 64-fache Anwendung der Kompressionsfunktion
  4) Generierung des finalen/zwischenzeitlichen Hash-Werts

SHA-256 Initialisierungskonstanten

- Entsprechen den ersten 32 Bits der Nachkommastellen der Quadratwurzeln der ersten acht Primzahlen

- $H_0^{(0)} = \texttt{6a09e667}$
- $H_1^{(0)} = \texttt{bb67ae85}$
- $H_2^{(0)} = \texttt{3c6ef372}$
- $H_3^{(0)} = \texttt{a54ff53a}$
- $H_4^{(0)} = \texttt{510e527f}$
- $H_5^{(0)} = \texttt{9b05688c}$
- $H_6^{(0)} = \texttt{1f83d9ab}$
- $H_7^{(0)} = \texttt{5be0cd19}$

SHA-2 Kompressionsfunktion



- Wettbewerb lief von 2006 bis 2015
- Sieger: **Keccak** Familie von Hash-Funktionen
  - Basiert auf einer ”Sponge Construction”
  - State: 5x5 64-bit Felder, 1600 Bits
  - Output: 224, 256, 384 oder 512 Bits
  - Nur 24 Runden nötig
- Hat auch andere kryptographische Verfahren beeinflusst

QUIZ: Hash Funktionen (twbk.de)

Welches Schutzziel kann mit einer Hash-Funktion verfolgt werden?

A) Leider keines ...

B) Integrität

C) Vertraulichkeit

QUIZ: Hash Funktionen (twbk.de)

Wie viele Bits eines Hashwerts sollten sich ändern, wenn man ein Bit in den Eingangsdaten ändert?

A) Ca. 33 %!

B) Die Hälfte!

C) Möglichst viele!

Message Authentication Codes

Zielsetzung

- Authentizität und Integrität von Nachrichten schützen
- Prüfsumme wird aus Schlüssel und Nachricht berechnet
- Sender und Empfänger benötigen selben Schlüssel

HMAC

- MAC basierend auf Hash-Funktion, z.B. HMAC-SHA256
- Ausgabelänge ist gleich wie bei der Hash-Funktion, z.B. 256-bit
- Sicherheit hängt von der Sicherheit der Hash-Funktion ab
- Definiert in RFC 2104:

$$ipad = \{0x36, ..., 0x36\}$$
$$opad = \{0x5C, ..., 0x5C\}$$

$$HMAC = hash(K \oplus opad | hash(K \oplus ipad | M))$$

CMAC

- MAC basierend auf Block-Chiffre, z.B. AES-CMAC
- Basiert auf CBC-MAC (Chiffre im CBC Modus)
- Ausgabelänge ist Blockgröße der Chiffre, z.B. 128-bit
- Spezifiziert in NIST SP 800-38B und RFC 4493 (für AES)

Authenticated Encryption (AE)

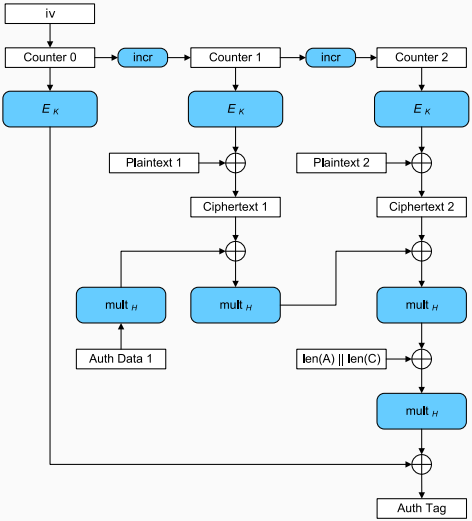
Zielsetzung

- Chiffren, die Verschlüsselung und MAC kombinieren
- Verarbeitung ergibt Ciphertext und Authentication Tag
- Kann durch Betriebsmodus erreich werden
  - z.B. CCM, GCM, CWC, EAX, IAPM, OCB, ...
- Authenticated Encryption with Associated Data (AEAD)
  - Integritäts- und Authentizitäts-Schutz für zusätzliche unverschlüsselte Daten mit Bezug

AES-GCM

- AES im Galois/Counter Mode (GCM)
- Standardisiert in NIST SP800-38D
- Beliebt wegen Effizienz und Performance
- IV muss einzigartig sein

Verschlüsselung im GCM Modus



NIST (Wikipedia), Public Domain

CAESAR Wettbewerb

- Wettbewerb zur Suche von Alternativen zu AES-GCM
- Lief von 2013 bis 2019
- Finales Algorithmen-Portfolio
  - Lightweight applications (resource constrained environments)
    → Ascon, ACORN
  - High-performance applications
    → AEGIS-128, OCB
  - Defense in depth
    → Deoxys-II, COLM

Gibt es noch Fragen?