

IT-Sicherheit

- Sicherheit vernetzter Systeme -

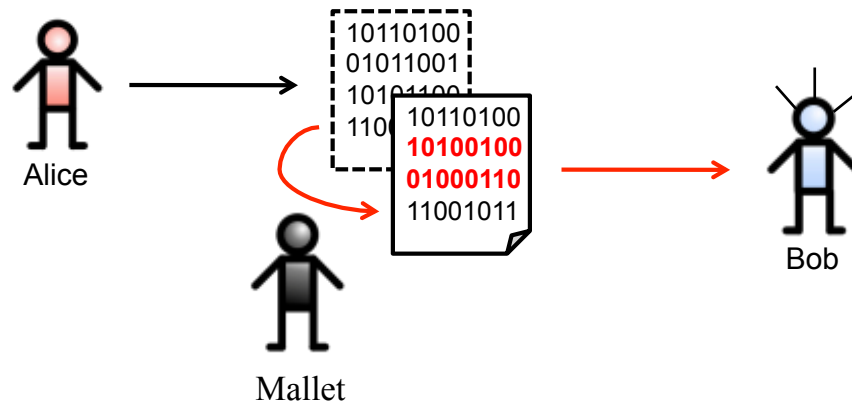
Kapitel 7: Kryptographische Hash-Funktionen

Inhalt

- Definition: Kryptographische Hash-Verfahren
- Angriffe gegen One-Way-Hash-Funktionen
- Konstruktion von Hash-Funktionen
- Algorithmen:
 - MD5
 - SHA-3 (Keccak)

Hash-Funktionen zur Integritätssicherung

- Ziel: Sicherstellen, dass Manipulationen an einer übertragenen Nachricht erkannt werden.



- Beispiel Software-Distribution:

[[downloads]]

Show pagesource Old revisions

Trace: » downloads

Downloads

- The complete Changelog
- You can browse the file archive [here](#).
- For installation information, see README.
- Don't forget to [patch your driver](#) before using the software.
- See [this page](#) to know how to use the software.

Current Sources

This tarball contains the latest Linux sources.

[aircrack-ng-1.1.tar.gz](#)

SHA1: 16eed1a8cf06eb8274ae382150b56589b23adf77

MDS: f7a24ed8fad122c4187d06bfd6f998b4

Current Sources

This tarball contains the latest Linux sources.

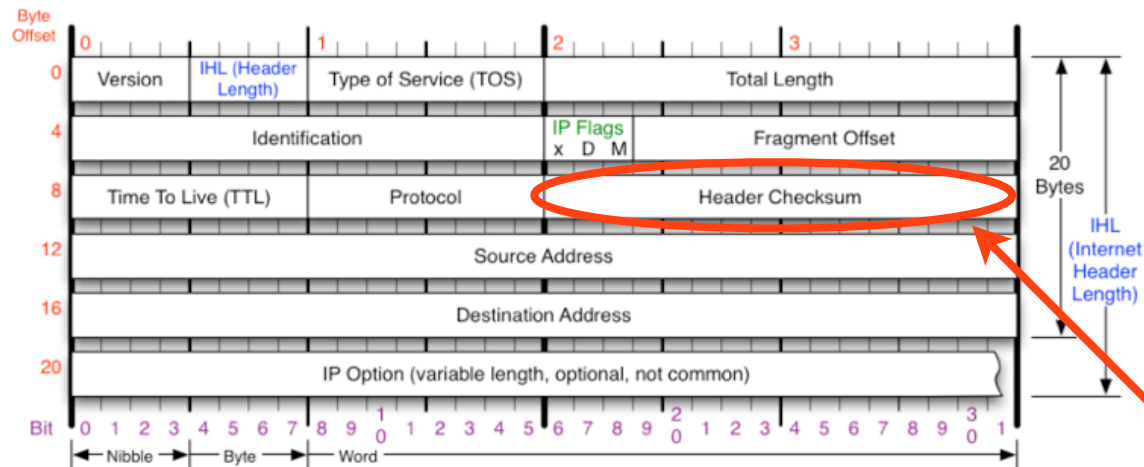
[aircrack-ng-1.1.tar.gz](#)

SHA1: 16eed1a8cf06eb8274ae382150b56589b23adf77

MDS: f7a24ed8fad122c4187d06bfd6f998b4

Herkömmliche vs. kryptographische Hash-Funktionen

- Prüfsummen dienen der Erkennung von (unbeabsichtigten) Übertragungsfehlern, z.B. beim IPv4-Header:



Quelle: nmap.org

Version Version of IP Protocol. 4 and 6 are valid. This diagram represents version 4 structure only.	Protocol IP Protocol ID. Including (but not limited to): 1 ICMP 17 UDP 57 SKIP 2 IGMP 47 GRE 88 EIGRP 6 TCP 50 ESP 89 OSPF 9 IGRP 51 AH 115 L2TP	Fragment Offset Fragment offset from start of IP datagram. Measured in 8 byte (2 words, 64 bits) increments. If IP datagram is fragmented, fragment size (Total Length) must be a multiple of 8 bytes.	IP Flags x D M x 0x80 reserved (evil bit) D 0x40 Do Not Fragment M 0x20 More Fragments follow RFC 791 Please refer to RFC 791 for the complete Internet Protocol (IP) Specification.
Header Length Number of 32-bit words in TCP header, minimum value of 5. Multiply by 4 to get byte count.	Total Length Total length of IP datagram, or IP fragment if fragmented. Measured in Bytes.	Header Checksum Checksum of entire IP header	

16-bit Addition /
Einerkomplement

Kryptographische Hash-Funktionen: Grundlagen

■ Hash-Funktionen

- bilden „Universum“ auf endlichen Bildbereich ab
- sind **nicht** injektiv
- Bildbereich i.d.R. sehr viel kleiner als Universum
- Kollisionen möglich:

$$\exists x, y \in U : x \neq y \quad \wedge \quad h(x) = h(y)$$

■ *Kryptographische* Hash-Funktion H:

- Eingabe: beliebig langes Wort m aus dem Universum U
- Ausgabe: Hashwert H(m) mit fester Länge
- H soll möglichst kollisionsresistent sein

Beispiel

- MD5-Hashwerte sind immer 128 Bits lang
 - egal, wie lange die Eingabe ist

```
829c11ba6dcdfe045dd1e5a77b34c05e 00o-SDK_3.2.1_Linux_x86-64_install-deb_en-US.tar.gz
0f8abee370438e49e7ea0c2287589760 00o-SDK_3.2.1_Linux_x86-64_install-rpm_en-US.tar.gz
35e8406c95c58b0087b9ad964faa13b8 00o-SDK_3.2.1_Linux_x86_install-deb_en-US.tar.gz
ecc8271619ad788203cc61c7d9930522 00o-SDK_3.2.1_Linux_x86_install-rpm_en-US.tar.gz
dddab486fd466bb1fc1a126d75919a3f 00o-SDK_3.2.1_MacOS_x86_install_en-US.dmg
c4eb9f161736ba64933bdc81c274d8bc 00o-SDK_3.2.1_Solaris_Sparc_install_en-US.tar.gz
3e8c88a645a8706e6c1bf7ef103bb993 00o-SDK_3.2.1_Solaris_x86_install_en-US.tar.gz
7a7f4ea9173f9b8d9ae71cdf65c328f0 00o-SDK_3.2.1_Win_x86_install_en-US.exe
```

- Weil es nur 2^{128} verschiedene MD5-Hashwerte gibt, existieren beliebig viele Dateien mit demselben MD5-Hashwert
 - = Kollision
- Zwei sehr ähnliche, aber nicht identische Eingaben sollen nicht denselben MD5-Hashwert haben
 - = Kollisionsresistenz
- Angreifer versucht, die Nachricht m „sinnvoll“ in m' abzuändern, so dass $md5(m) = md5(m')$

Def. Kryptographische Hashfunktion

■ Schwache Hash-Funktion H:

- H besitzt die Eigenschaften einer Einwegfunktion
- Hashwert $H(m) = h$ mit $|h|=k$ (z.B. $k = 128$ Bits) ist bei gegebener Nachricht m einfach zu berechnen
- Bei gegebenem $h = H(m)$ für $m \in A_1^*$ ist es praktisch unmöglich, eine (sinnvolle) m' zu finden mit:

$$m' \neq m, m' \in A_1^* \quad \wedge \quad H(m') = h$$

■ Starke Hash-Funktion H:

- H hat alle Eigenschaften einer schwachen Hash-Funktion
- Es ist zusätzlich praktisch unmöglich, eine Kollision zu finden, d.h. ein Paar verschiedene Eingabewerte m und m' mit:

$$m' \neq m, m, m' \in A_1^* \quad \wedge \quad H(m) = H(m')$$

Birthday Attack auf One-Way-Hash-Funktionen

- Wie viele Personen brauchen Sie, damit mit Wahrscheinlichkeit $P > 0,5$ eine weitere Person mit Ihnen Geburtstag hat?
- Wie viele Personen brauchen Sie, damit mit Wahrscheinlichkeit $P > 0,5$ zwei Personen am selben Tag Geburtstag haben?
- Wie können Sie dieses Wissen für Angriffe gegen Hash-Funktionen nutzen?

Birthday Attack: Vorgehensweise

1. Alice sichert mit einem k Bits langen Hash eine Nachricht M
 2. Mallet erzeugt $2^{k/2}$ Variationen der Nachricht M
- Die Wahrscheinlichkeit für eine Kollision ist größer 0,5.
 - Wie können $2^{k/2}$ Variationen erzeugt werden?
 - Z.B. Einfügen von „Space – Backspace – Space“ Zeichen zwischen Wörtern
 - Wörter durch Synonyme ersetzen
 -

Beispiel für einen Brief mit 2^{37} Variationen

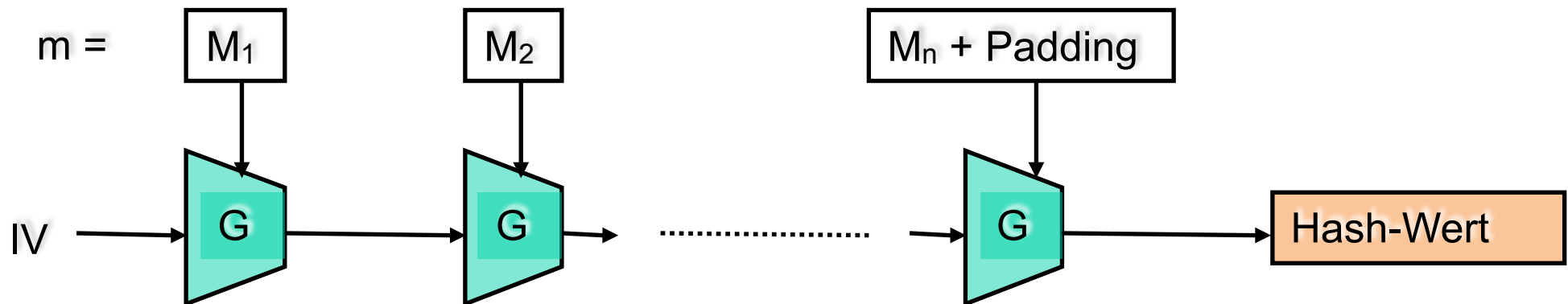
■ [Stal 98]

Dear Anthony,

{ This letter is } to introduce { you to } { Mr. } Alfred { P. }
{ I am writing } { to you } { -- }
Barton, the { new } { chief } jewellery buyer for { our }
{ newly appointed } { senior } { the }
Northern { European } { area } . He { will take } over { the }
{ Europe } { division } { has taken } { -- }
responsibility for { the } { all } our interests in { watches and jewellery }
{ the whole of } { jewellery and watches }
in the { area } . Please { afford } him { every } help he { may need }
{ region } { give } { all the } { needs }
to { seek out } the most { modern } lines for the { top } end of the
{ find } { up to date } { high }
market. He is { empowered } to receive on our behalf { samples } of the
{ authorized } { specimens }
{ latest } { watch and jewellery } products, { up } to a { limit }
{ newest } { jewellery and watch } { subject } { maximum }
of ten thousand dollars. He will { carry } a signed copy of this { letter }
{ hold } { document }
as proof of identity. An order with his signature, which is { appended }
{ attaches }
{ authorizes } you to charge the cost to this company at the { above }
{ allows } { head office }
address. We { fully } expect that our { level } of orders will increase in
{ -- } { volume }
the { following } year and { trust } that the new appointment will { be }
{ next } { hope } { prove }
{ advantageous } to both our companies.
{ an advantage }

Konstruktion kryptographischer Hash-Funktionen

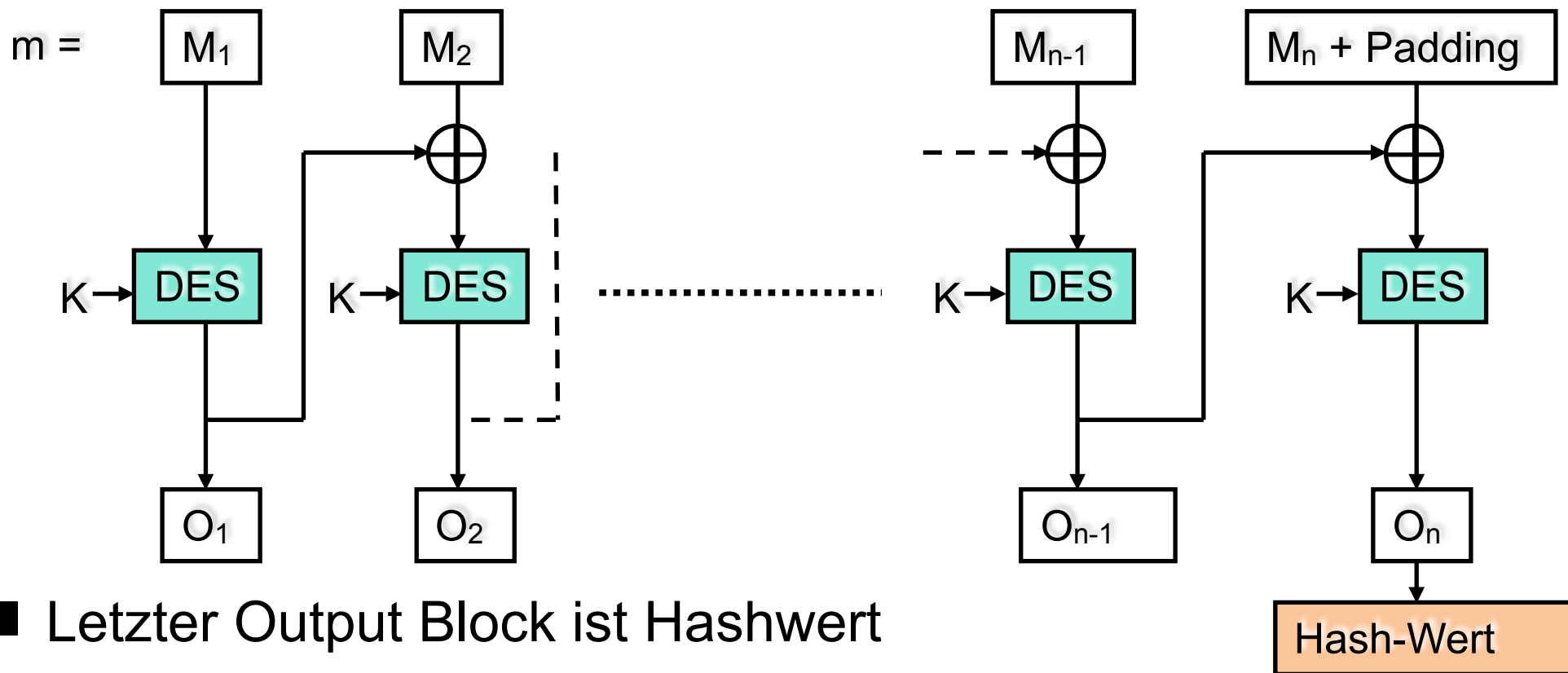
- Folge von Kompressionsfunktionen G
- Nachricht m wird in Blöcke M_i mit fester Länge y zerlegt
- Hash-Verfahren wird mit Initialisierungswert IV vorbelegt



- Letzter Block M_n muss ggf. auf vorgegebene Länge y „aufgefüllt“ werden (Padding)
- Als Kompressionsfunktion G können verwendet werden:
 - Hash-Funktionen auf der Basis symmetrischer Blockchiffren
 - Dedizierte Hash-Funktionen

DES als Kompressionsfunktion

■ DES im Cipher Block Chaining (CBC) Mode

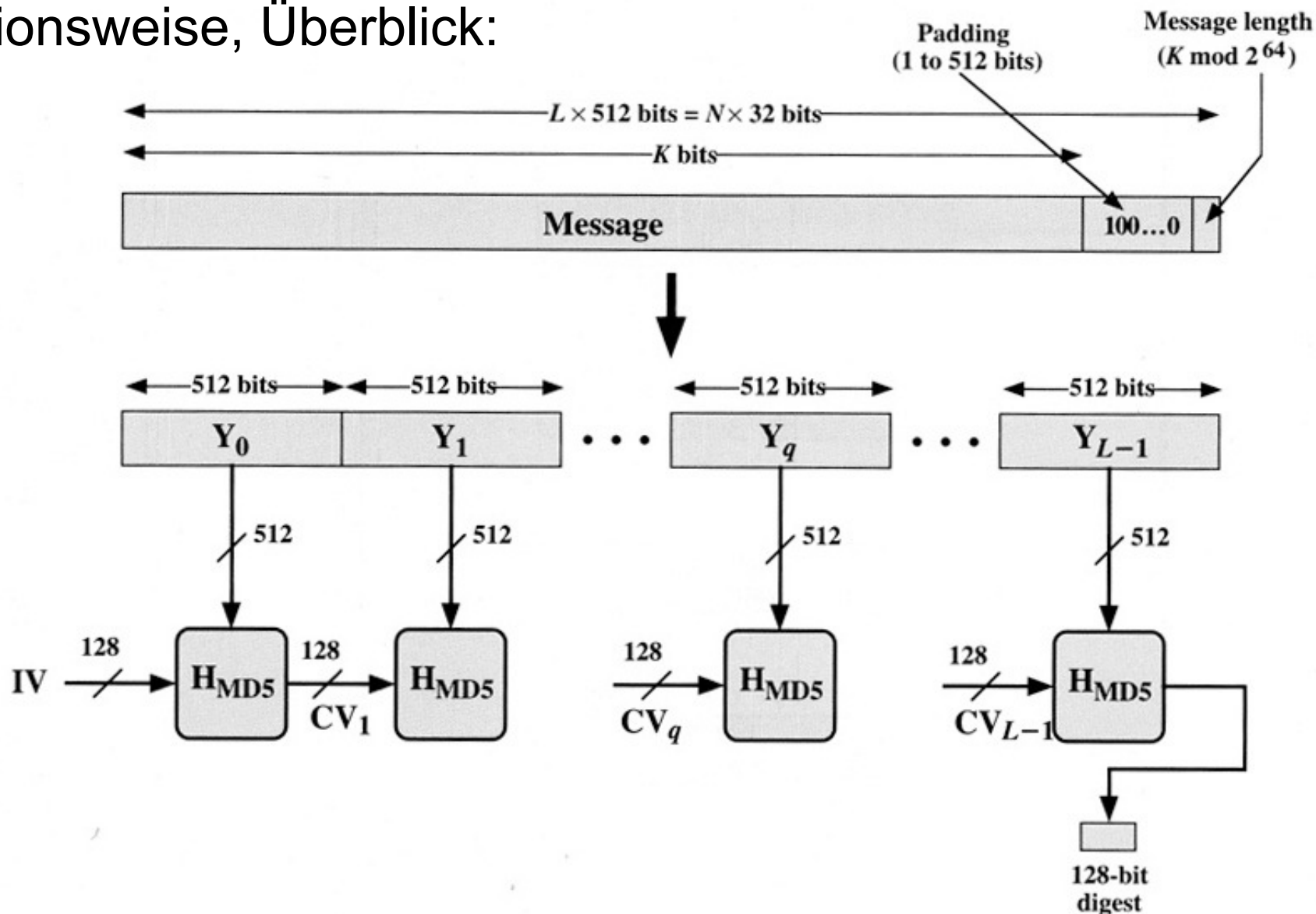


■ Letzter Output Block ist Hashwert

■ Länge des Hashwerts?

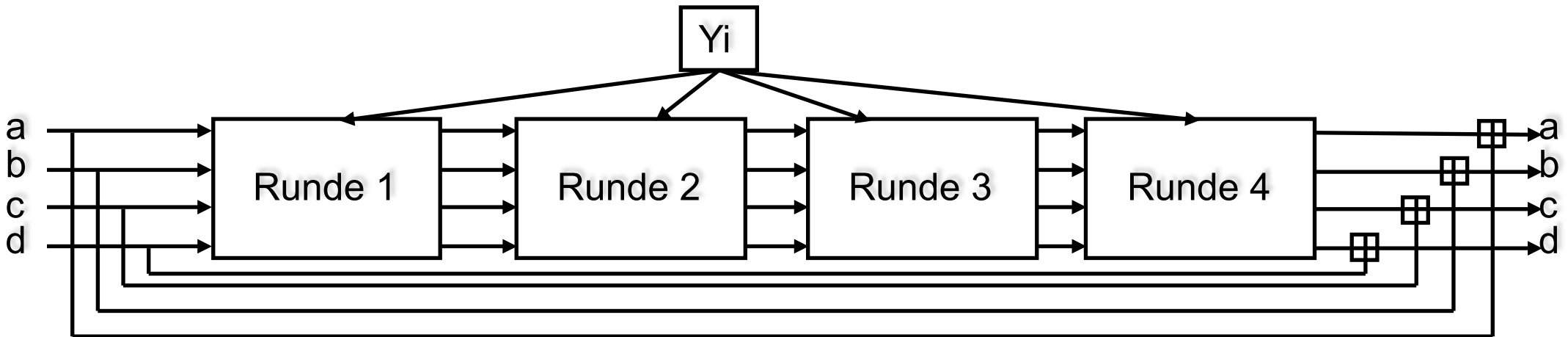
Hash-Funktionen: MD5 (1991)

- Ausgabelänge 128 Bit, arbeitet auf 512-Bit-Eingabeblöcken
- Funktionsweise, Überblick:



MD5: Ablauf

1. Padding Bits der Nachricht hinzufügen
2. Länge der Originalnachricht (mod 2^{64}) anfügen
3. Nachricht in 512-Bit-Blöcke aufteilen
4. Initialisierung von 32-Bit-Variablen:
 $A = 0x01234567$ $C = 0xFEDCBA98$
 $B = 0x89ABCDEF$ $D = 0x76543210$
5. Zuweisung $a=A$, $b=B$, $c=C$, $d=D$
6. Kompressionsfunktion H_{MD5} angewendet auf jeden (Teil-)Block



Sicherheit von MD5

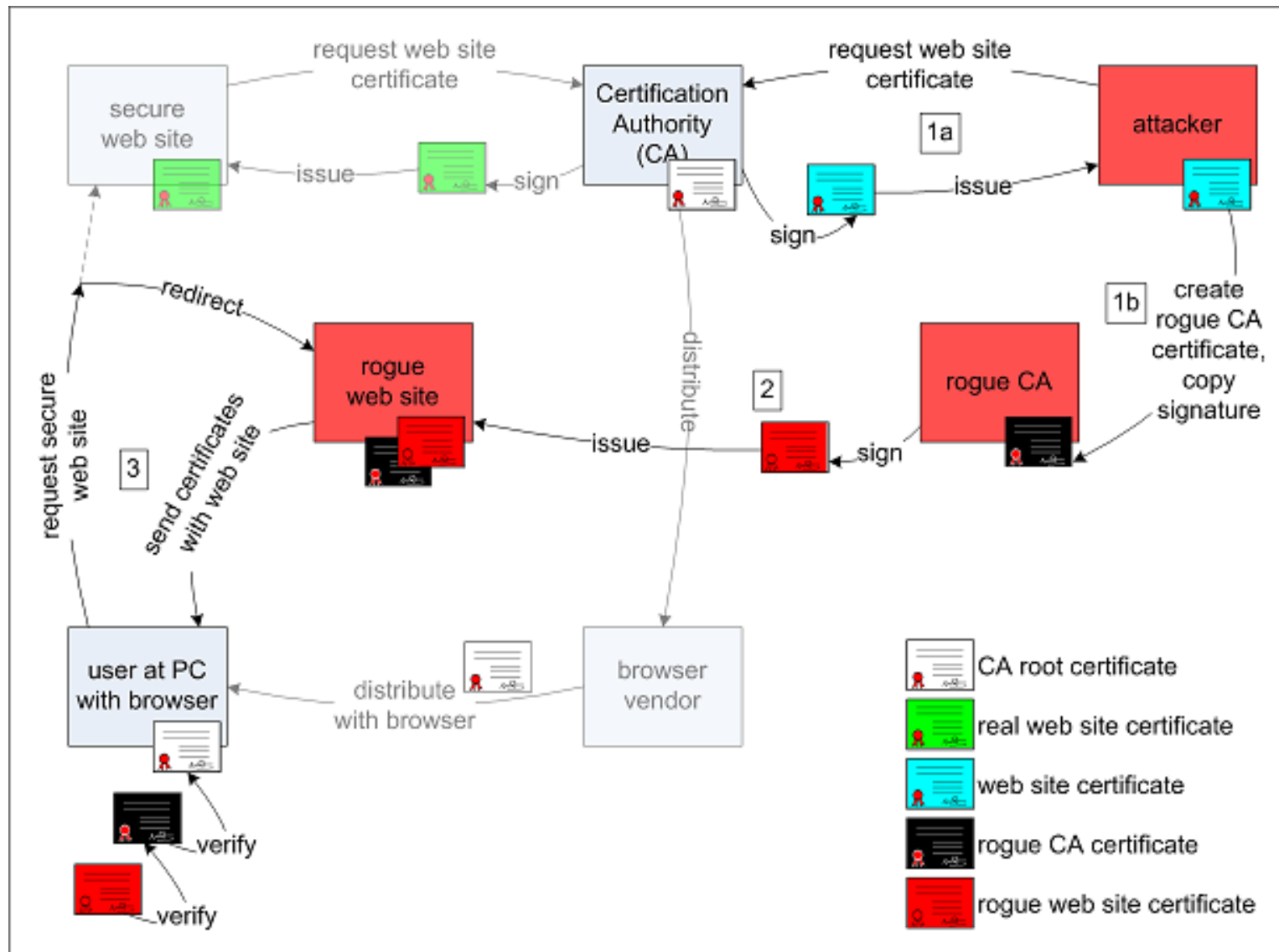
- Differentielle Kryptanalyse auf MD5 mit nur einer Runde [Bers 92]:
 - Für jede der 4 Runden einzeln möglich
 - Angriff auf alle 4 Runden konnte nicht gezeigt werden
- Pseudokollision [BoBo 93]:
 - Zwei verschiedene Variablenbelegungen von a,b,c,d führen für verschiedene Inputblöcke zum gleichen Outputblock
 - Damals schien eine Erweiterung des Ansatzes zu einem allgemeinen Angriff nicht möglich
- Erzeugung einer Kollision in der Kompressionsfunktion [Dobb 96]:
 - Zwei 512 Bit Blöcke produzieren den selben 128 Bit Output
 - Bis dahin gefährlichster bekannter Angriff

Sicherheit von MD5 (Forts.)

- Kollision gefunden [Wang,Feng,Lai,Yu 2004]:
 - $MD5(M,N_i) = MD5(M,N_i')$
 - M und M' zu finden dauert ca. eine Stunde (IBM P690 Cluster)
 - danach N_i und N_i' zu finden 15 Sek. bis 5 Minuten
 - funktioniert mit beliebigen Initialisierungsvektor IV
 - In der Arbeit werden auch Kollisionen für MD4, HAVAL-128 und RIPEMD-128 angegeben
 - Ende des MD5CRK-Projekts (distributed birthday attack)
- Kollision in X.509 Zertifikat gefunden (Kollision in den Schlüsseln) [de Weger 2005]
- Kollision in X.509 Zertifikat mit unterschiedlichen Identitäten [Stevens, Lenstra, de Weger 2006/2007]
- ➔ MD5 (und SHA-1) nicht mehr verwenden!
- ➔ Algorithmen mit längeren Hash-Werten verwenden:
z.B. SHA-256, Whirlpool, SHA-3, o.ä.

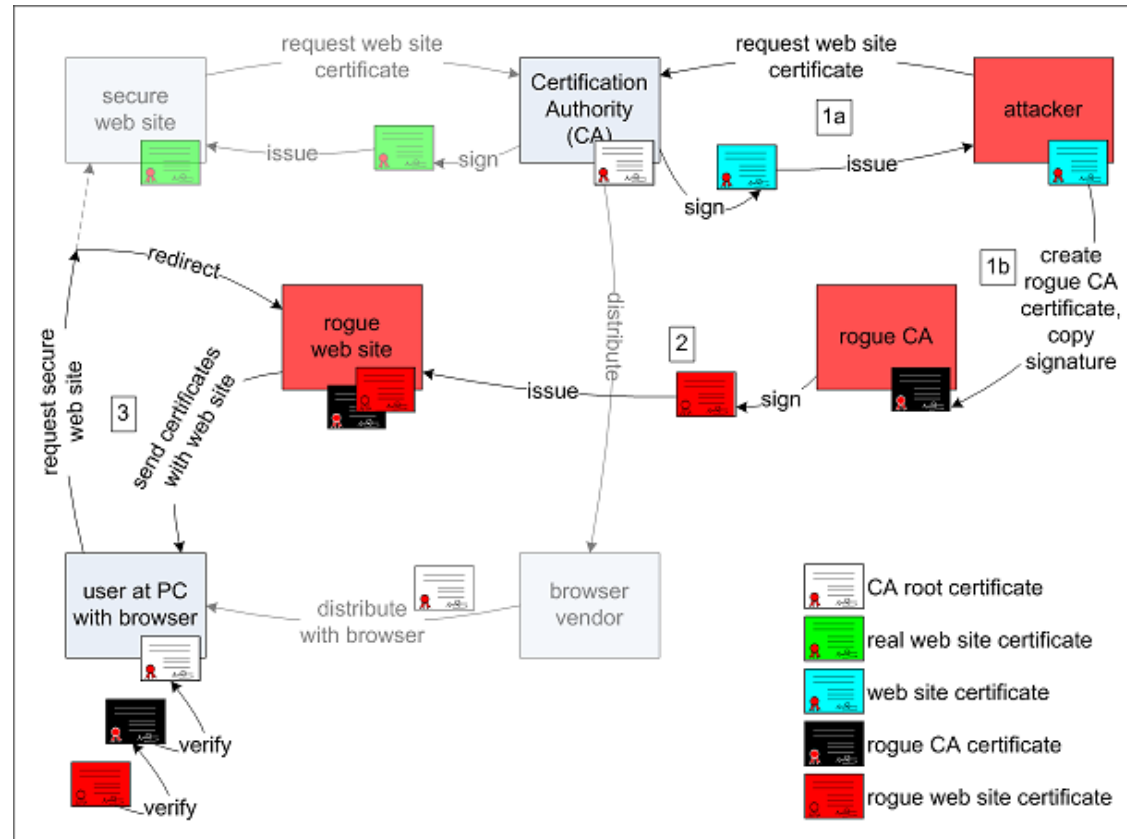
Sicherheit von MD5 (Forts.)

- Bislang umfangreichster, praktisch relevanter Angriff [SSALMOW08]: <http://www.win.tue.nl/hashclash/rogue-ca/>



Sicherheit von MD5 (Forts.)

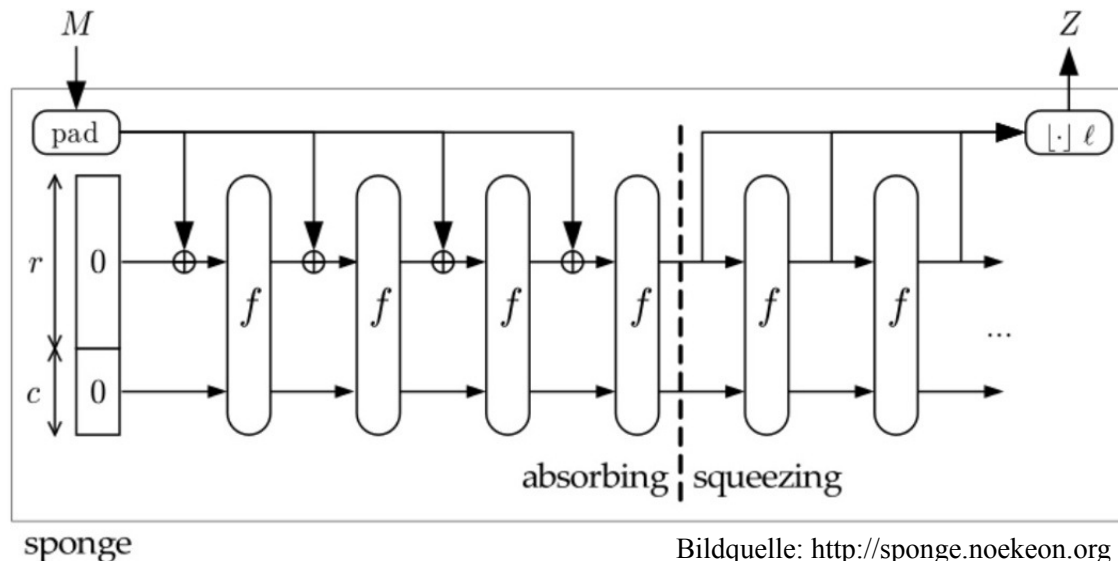
- Bislang umfangreichster, praktisch relevanter Angriff [SSALMOW08]: <http://www.win.tue.nl/hashclash/rogue-ca/>



- Alle Browser, die RapidSSL-Zertifikaten vertrauten, vertrauten auch den mit dem „rogue CA certificate“ ausgestellten Zertifikaten
- Man-in-the-Middle Angriffe: Browser kann bei SSL-Zertifikaten, die MD5-Hashsummen verwenden, die Server-Authentizität nicht mehr zuverlässig prüfen

SHA-3

- 10/2012 vom NIST als Nachfolger von SHA-2 standardisiert
- 2007: Wettbewerb ähnlich zu AES-Standardisierung:
 - motiviert durch erfolgreiche Angriffe auf MD5 und SHA-1
 - 64 Einreichungen, 14 Algorithmen in engerer Auswahl, 5 Finalisten
 - Gewinner: Keccak von Bertoni, Daemen, Peeters und van Assche
- Innovativer Ansatz: Sponge-Funktion



Zwei Phasen:
absorbing/squeezing

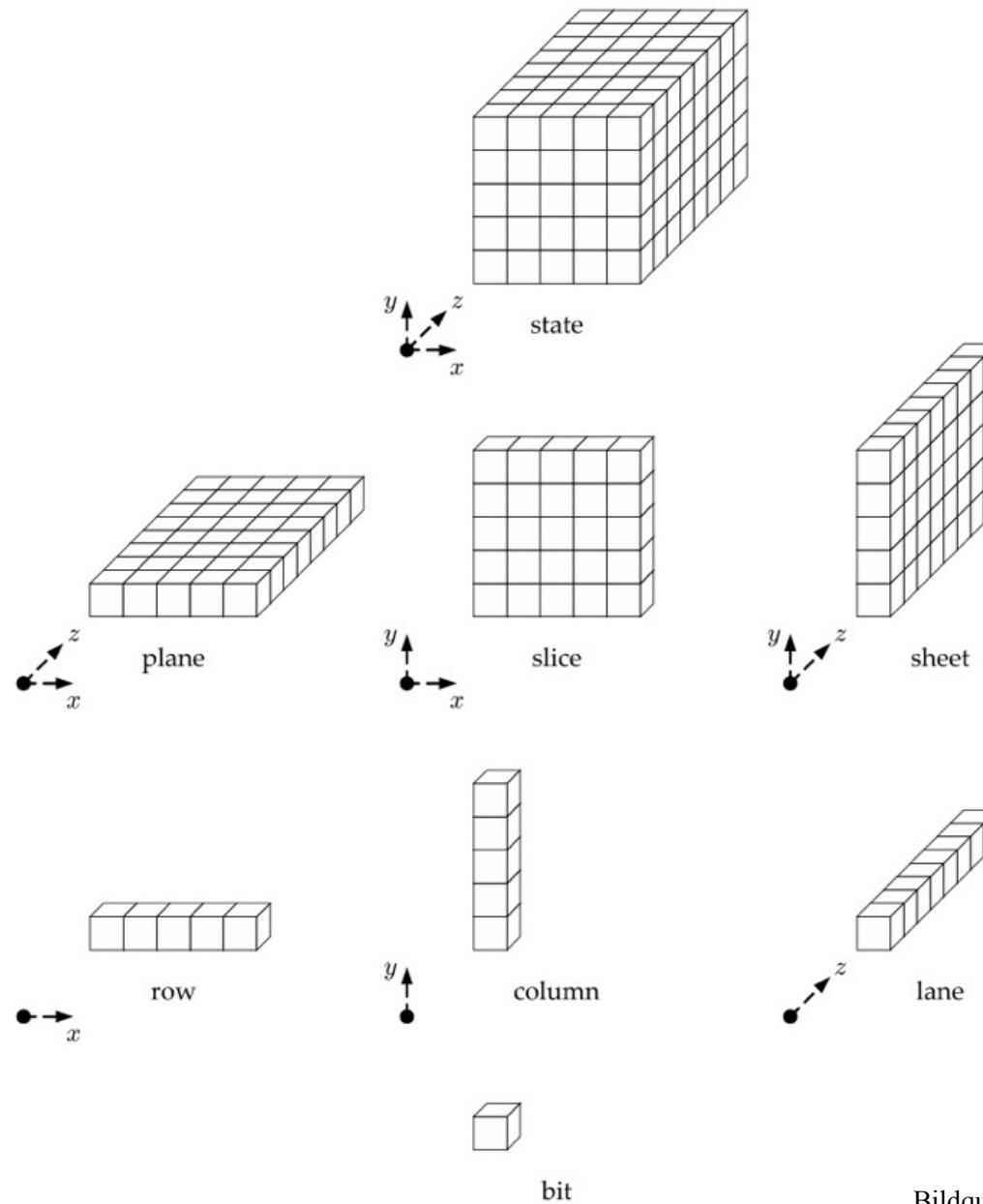
Variable Output-Länge

Keccak: Parametrisierung und Keccak-f

- Als SHA-3 standardisierte Varianten umfassen u.a.
 - SHA3-256: $r=1152$, $c=448$, Ausgabe abgeschnitten nach 256 Bits
 - SHA3-512: $r=576$, $c=1024$, Ausgabe abgeschnitten nach 512 Bits
- $f[b]$ Keccak Permutationsfunktion; Breite der Permutation
 $b = c + r = 25 \cdot 2^1$
- Funktion f betrachtet State als dreidimensionales Array von $GF[2]$
 $a[5][5][w]$ mit $w = 2^1$, $b = c + r = 25 \cdot 2^1$
Beispiel SHA3-256: $b = 1152 + 448 = 1600$,
d.h. $l = 6$, $w = 64$
- Jede Anwendung von f besteht aus n_r Runden:

$$n_r = 12 + 2 \cdot 1, \text{ d.h. für SHA3-256: } n_r = 24$$

Keccak: Terminologie zum State



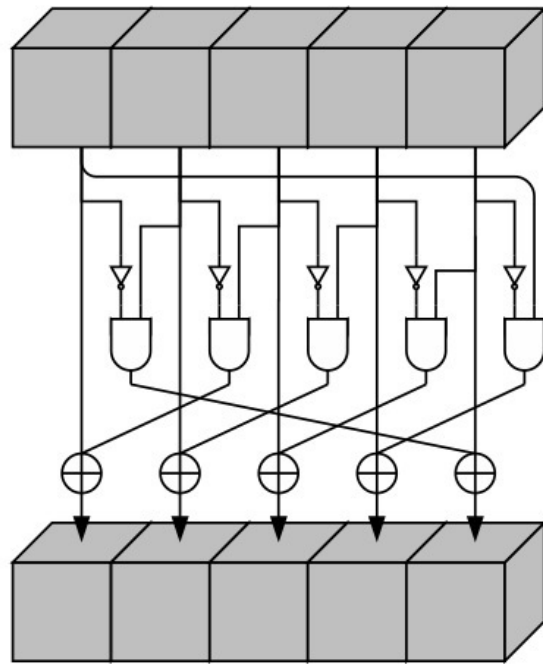
Bildquelle: <http://keccak.noekeon.org/Keccak-reference-3.0.pdf>

Keccak-f: Runden

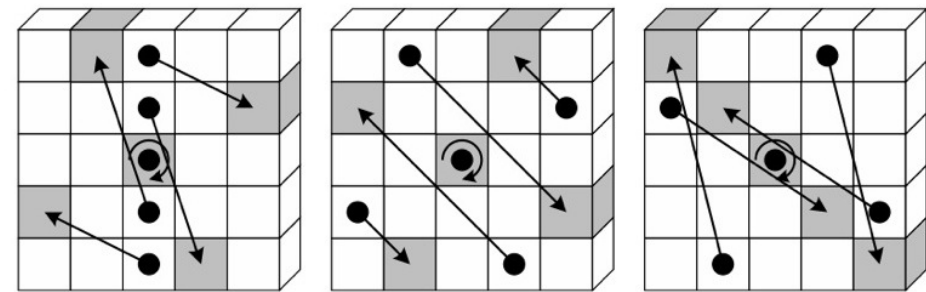
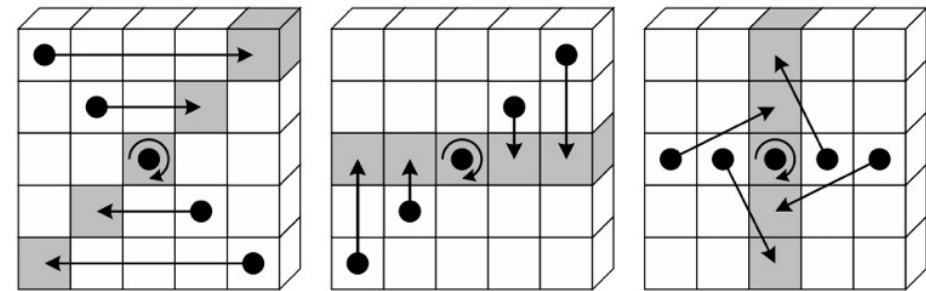
■ Jede Runde besteht aus fünf Schritten:

□ $R = \iota \circ \chi \circ \pi \circ \rho \circ \theta$,

- Addition von Rundenkonstanten
- Nichtlinearität
- Erhöhung der Diffusion in allen drei Dimensionen



χ



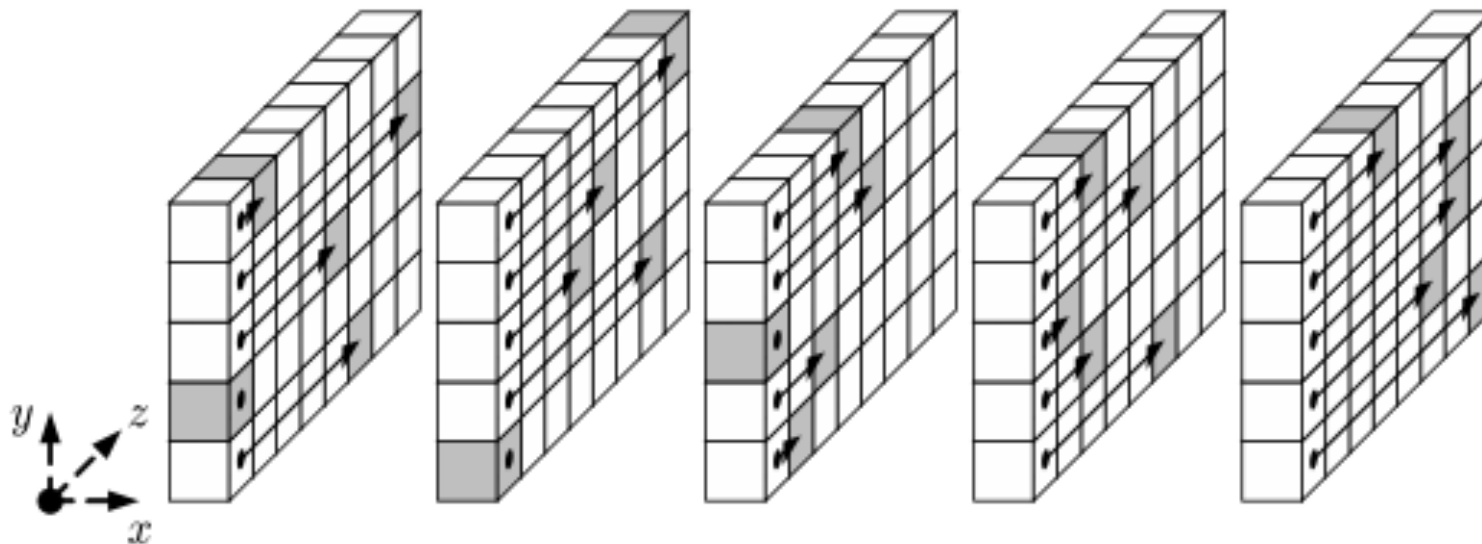
π

Keccak-f: Runden

■ Jede Runde besteht aus fünf Schritten:

□ $R = \iota \circ \chi \circ \pi \circ \rho \circ \theta$,

- Addition von Rundenkonstanten
- Nichtlinearität
- Erhöhung der Diffusion in allen drei Dimensionen



ρ

Keccak: Bewertung

- **Innovativer Ansatz:**
 - Vermeidet Probleme klassischer Merkle-Damgard-Konstrukte wie MD5;
 - ist entsprechend aber noch weniger von Kryptanalytikern untersucht.
 - Komplementär zu SHA-2 verwendbar.
- **Variable Output-Länge**
 - ermöglicht flexible Anpassung an jeweiligen Bedarf
 - Gute Eignung als PRNG für Stream Ciphers
- **Effiziente Implementierung in Hard- und Software möglich**
- **Konservative Sicherheitsreserve durch große Rundenzahl**