

IT-Sicherheit

- Sicherheit vernetzter Systeme -

Kapitel 7: Kryptographische Hash-Funktionen

Einschub: Klausur

■ Termin:

- Do.
- 23.02.2012
- 14:00 Uhr

■ Ort:

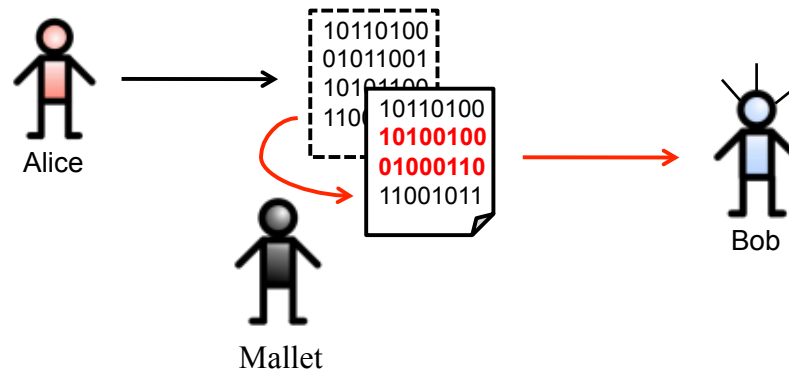
- Hörsaal: A240

Inhalt

- Def.: Kryptographische Hash-Verfahren
- Angriffe gegen One-Way-Hash-Funktionen
- Konstruktion von Hash-Funktionen
- Algorithmen:
 - MD4
 - MD5
 - Whirlpool

Hash-Funktionen zur Integritätssicherung

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



- Beispiel Software-Distribution:

The screenshot shows a web page with a "downloads" section and a "Current Sources" section. A large yellow arrow points from the "Current Sources" section to the "Downloads" section.

downloads

Show page source | 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](#) if you use one.
- See [this page](#) to know how to use the driver.

Current Sources

This tarball contains the latest Linux sources.

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

SHA1: 16eed1a8cf06eb8274ae382150b56589b23adf77
MD5: f7a24ed8fad122c4187d06bfd6f998b4

Current Sources

This tarball contains the latest Linux sources.

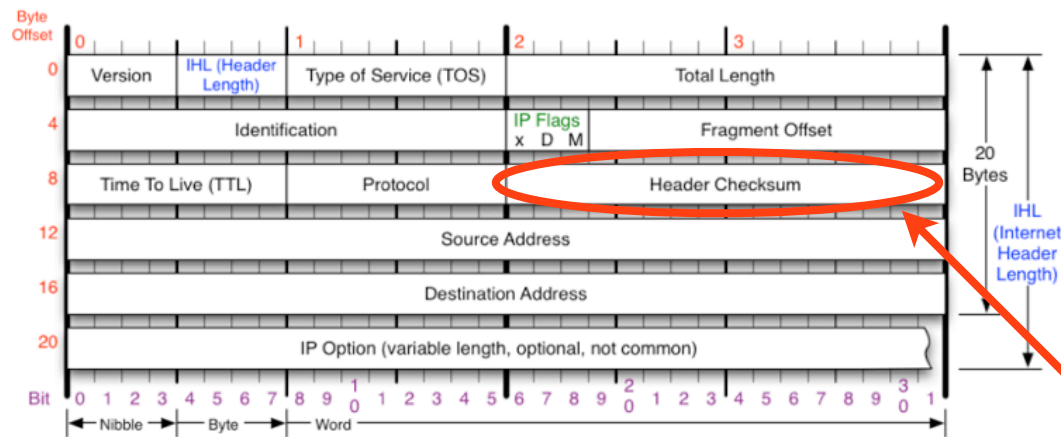
[aircrack-ng-1.1.tar.gz](#)

SHA1: 16eed1a8cf06eb8274ae382150b56589b23adf77

MD5: 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
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	RFC 791 Please refer to RFC 791 for the complete Internet Protocol (IP) Specification.

16-bit Addition /
Einerkomplement

- *Kryptographische* Prüfsummen sollen auch absichtliche Manipulationen erschweren

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
dddab486fd466bb1fcl1a126d75919a3f 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 gegebenem m einfach zu berechnen
- Bei gegebenem $h = H(m)$ für $m \in A_1^*$ ist es praktisch unmöglich, ein 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?

□ Antwort: 253

$$P = 1 - \left(1 - \frac{1}{365}\right)^n \quad (\text{ab } n=253 \text{ ist } P > 0,5)$$

- Wie viele Personen brauchen Sie, damit mit Wahrscheinlichkeit $p > 0,5$ zwei Personen am selben Tag Geburtstag haben?

□ Antwort: 23

$$P = 1 - \frac{n! \cdot \binom{365}{n}}{365^n} \quad (\text{ab } n=23 \text{ ist } P > 0,5)$$

- Wie können Sie dieses Wissen für Angriffe gegen Hash-Funktionen nutzen?

*Eine Kollision zu finden ist deutlich einfacher
als zu einem gegebenen Hash-Wert einen passenden Text!*

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

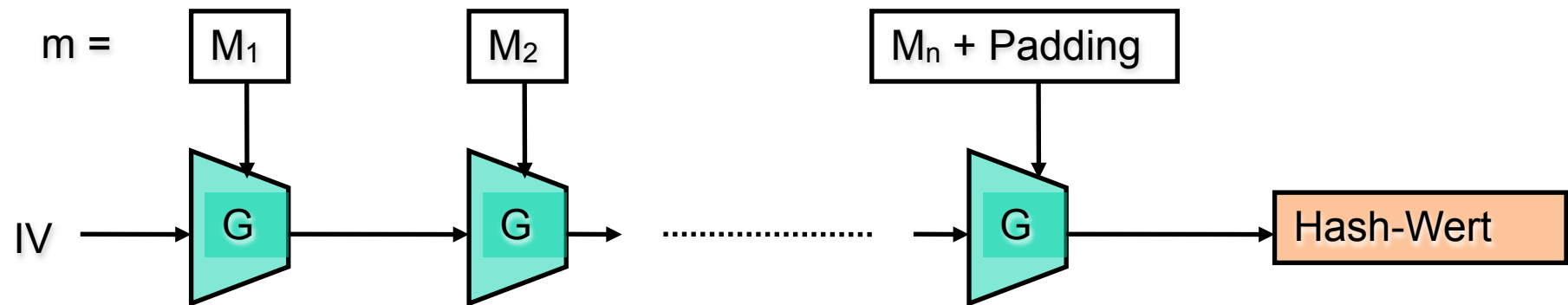
■ [Stal 98]

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 } . He { has taken }
responsibility for { the whole of } our interests in { watches and jewellery }
{ 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 }
{ attached }
{ 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

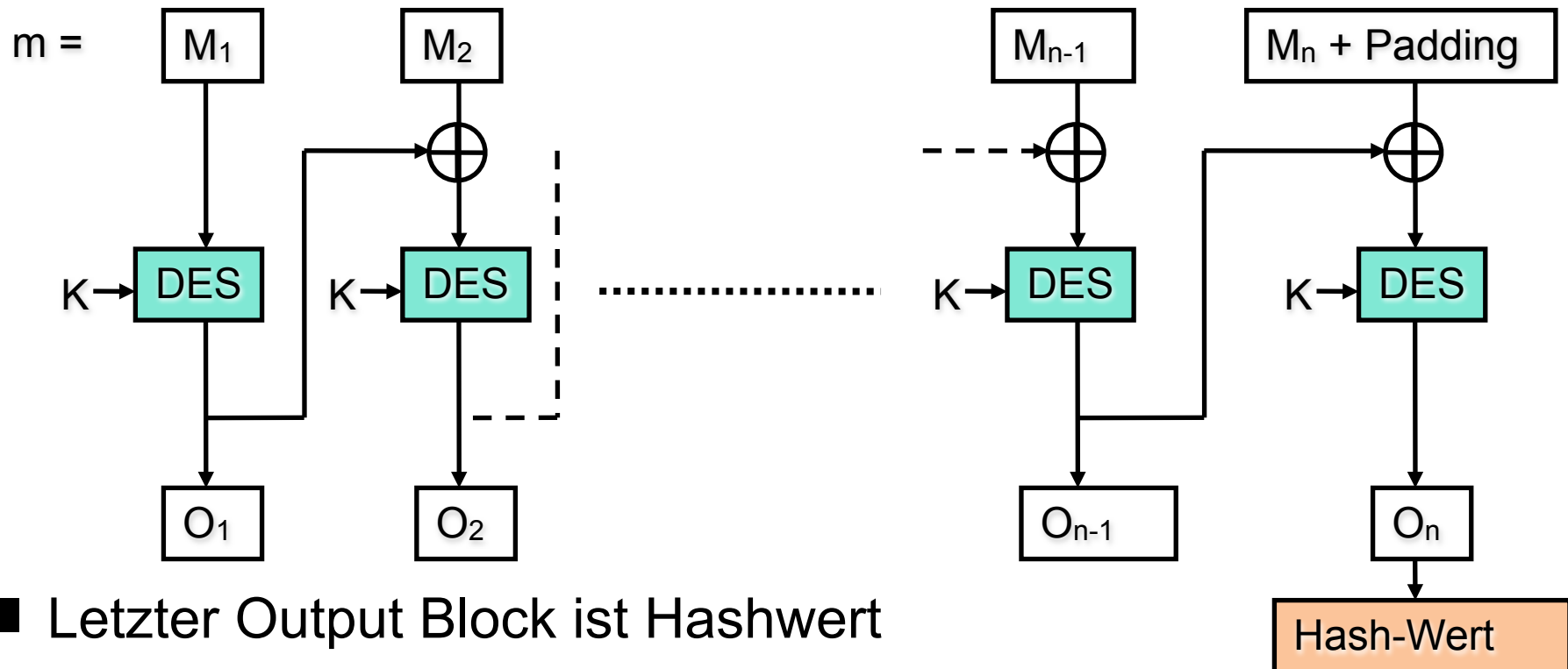
- Folge von Kompressionsfunktionen G
- Nachricht m wird in Blöcke M_i mit fester Länge y zerlegt
- Hash-Verfahren wird mit Initialisierungswert IV vorbelegt



- 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

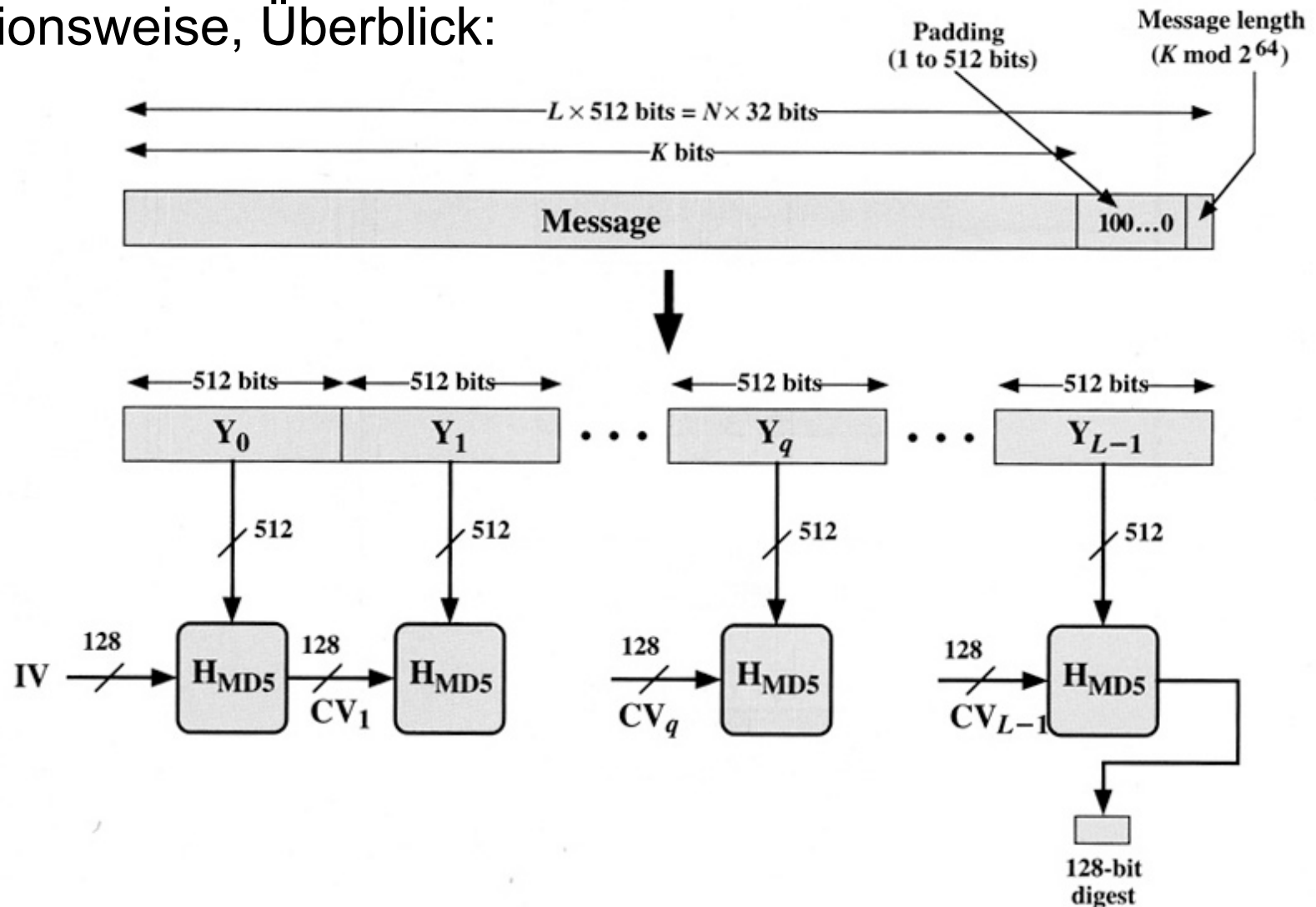
■ Länge des Hashwerts? 64 Bits

Hash-Funktionen: MD4 (1990)

- Entwickelt von Ron Rivest: MD4 = Message Digest Nr. 4
- Design-Kriterien:
 - Kollisionsresistenz: Es gibt kein besseres Verfahren als Brute Force, um zwei Nachrichten mit demselben MD4-Hash zu finden
 - Direkte Sicherheit: MD4 basiert auf keinerlei (Sicherheits-)Annahmen wie z.B. dem Faktorisierungsproblem
 - Geschwindigkeitsoptimiert für Software-Implementierungen
 - Bevorzugt Little Endian 32-Bit-Architekturen (Intel)
 - Einfach und kompakt
- Erfolgreiche Angriffe zeigen mangelnde Kollisionsresistenz:
 - Boer und Bosselaers brechen die beiden letzten Runden der insges. drei
 - Merkle greift erfolgreich die ersten beiden Runden an
 - Angriff auf alle 3 Runden gelingt nicht
 - Konsequenz: Rivest verbessert MD4; Ergebnis ist MD5
- MD4 ist Public Domain (IETF RFC1320);
sein Prinzip ist Basis für viele weitere Hash-Funktionen

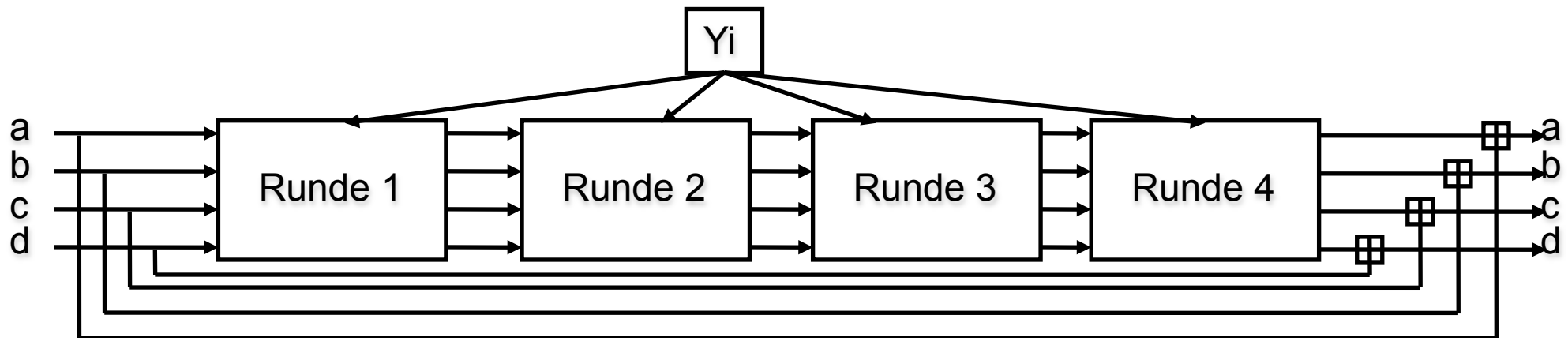
Hash-Funktionen: MD5 (1991)

- Länge 128 Bit, arbeitet auf 512 Bit Blöcken
- 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



MD5 Kompressionsfunktion (1)

- 4 Runden mit je einer nichtlinearen Funktion

$$F(X, Y, Z) = (X \wedge Y) \vee ((\neg X) \wedge Z)$$

$$G(X, Y, Z) = (X \wedge Z) \vee (Y \wedge (\neg Z))$$

$$H(X, Y, Z) = X \oplus Y \oplus Z$$

$$I(X, Y, Z) = Y \oplus (X \vee (\neg Z))$$

- Funktionen so gewählt, dass korrespondierende Bits von X, Y, Z und dem Ergebnis unabhängig voneinander sind

- In jeder Runde wird die Funktion 16 mal auf einen 32-Bit-Teilblock M_j von Y_i wie folgt angewendet

$$FF(a, b, c, d, M_j, s, t_i): \quad a = b + ((a + F(b, c, d) + M_j + t_i) \lll s)$$

$$GG(a, b, c, d, M_j, s, t_i): \quad a = b + ((a + G(b, c, d) + M_j + t_i) \lll s)$$

$$HH(a, b, c, d, M_j, s, t_i): \quad a = b + ((a + H(b, c, d) + M_j + t_i) \lll s)$$

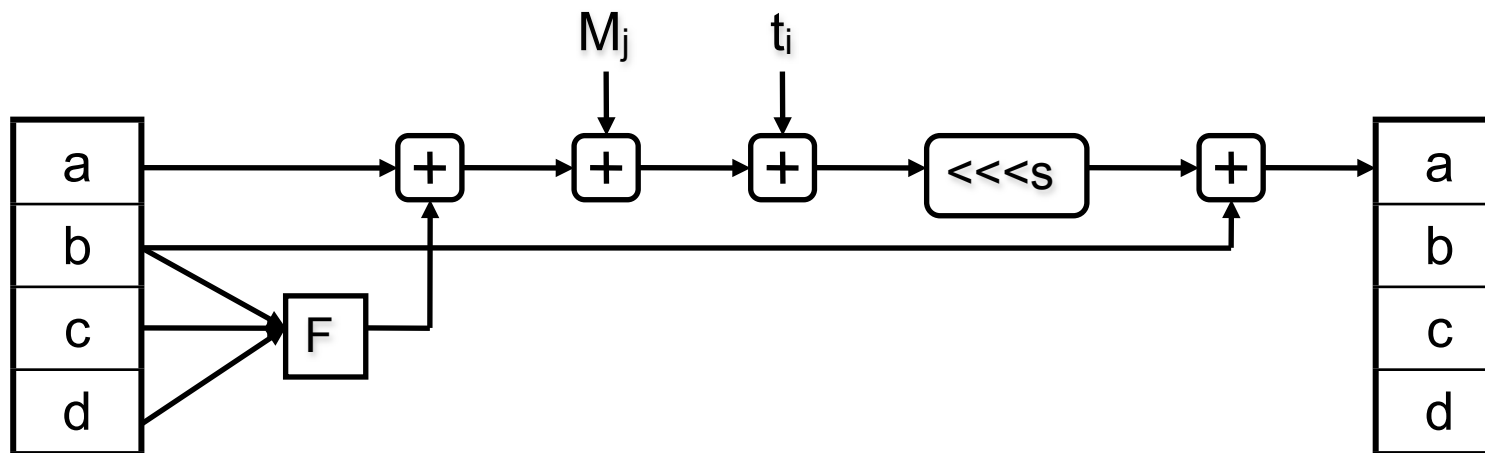
$$II(a, b, c, d, M_j, s, t_i): \quad a = b + ((a + I(b, c, d) + M_j + t_i) \lll s)$$

MD5 Kompressionsfunktion (2)

- $FF(a, b, c, d, M_j, s, t_i): a = b + ((a + F(b, c, d) + M_j + t_i) \lll s)$
 - + bezeichnet Addition modulo 2^{32}
 - $t_i = 2^{32} \text{abs}(\sin(i))$ mit i Grad im Bogenmaß; $0 \leq i < 64$ (i über 4 Runden)
 - $\lll s$ bezeichnet zirkulären Shift um s Bits (s variiert pro Operation)
 - Auswahl des Teilblocks M_j (Runde 1: $i = 0..15$, ..., Runde 4: $i = 48..63$)

Runde 1	Natürliche Ordnung	Runde 3	$(5 + 3i) \bmod 16$
Runde 2	$(1 + 5i) \bmod 16$	Runde 4	$7i \bmod 16$

- Beispiel: Elementarer Schritt in Runde 1 auf 32-Bit-Block



MD5: Rundenfunktion; 4 Runden mit 64 Schritten

■ Runde 1:

1. FF(a, b, c, d, M0, 7, 0xd76aa478)
2. FF(d, a, b, c, M1, 12, 0xe8c7b756)
3. FF(c, d, a, b, M2, 17, 0x242070db)
4. FF(b, c, d, a, M3, 22, 0xc1bdceee)

⋮

■ Runde 4:

60. II(a, b, c, d, M4, 6, 0xf7537e82)
61. II(d, a, b, c, M11, 10, 0xbd3af235)
62. II(c, d, b, a, M2, 15, 0x2ad7d2bb)
63. II(b, c, d, a, M9, 21, 0xeb86d391)

Sicherheit von MD5

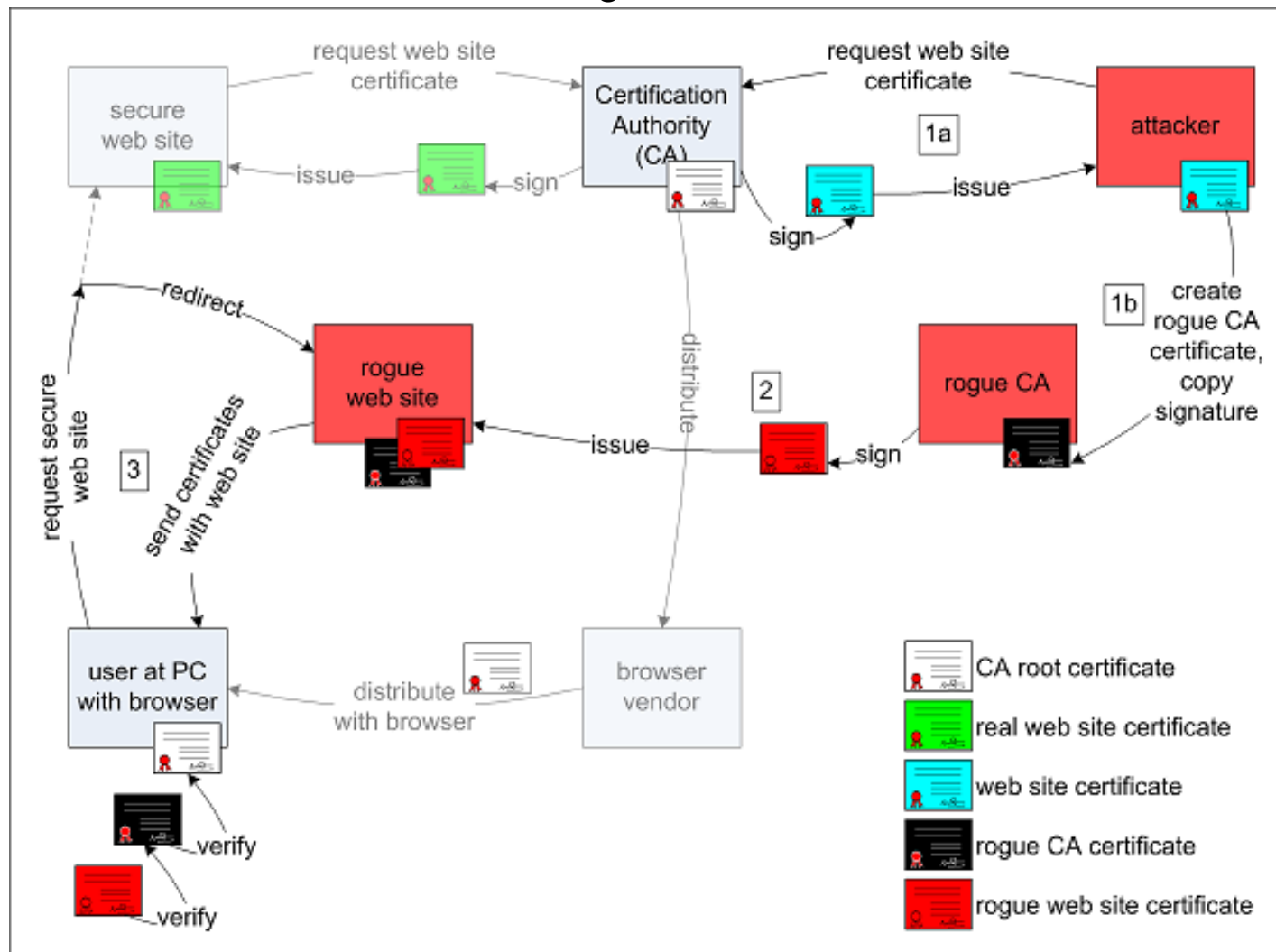
- Differentielle Kryptanalyse auf MD5 mit nur einer Runde [Bers 92]:
 - Für jede der 4 Runden einzeln möglich
 - Angriff auf alle 4 Runden konnte nicht gezeigt werden
- Pseudokollision [BoBo 93]:
 - Zwei verschiedene Variablenbelegungen von a,b,c,d führen für verschiedene Inputblöcke zum gleichen Outputblock
 - 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
 - Bisher kein Mechanismus zur Generalisierung des Angriffs auf gesamten MD5 mit IV gefunden

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 Wegener 2006/2007]
- ➔ MD5 (und SHA-1) nicht mehr verwenden!
- ➔ Algorithmen mit längeren Hash-Werten verwenden:
z.B. SHA-224, SHA-256, SHA-384, SHA-512, Whirlpool, o.ä.

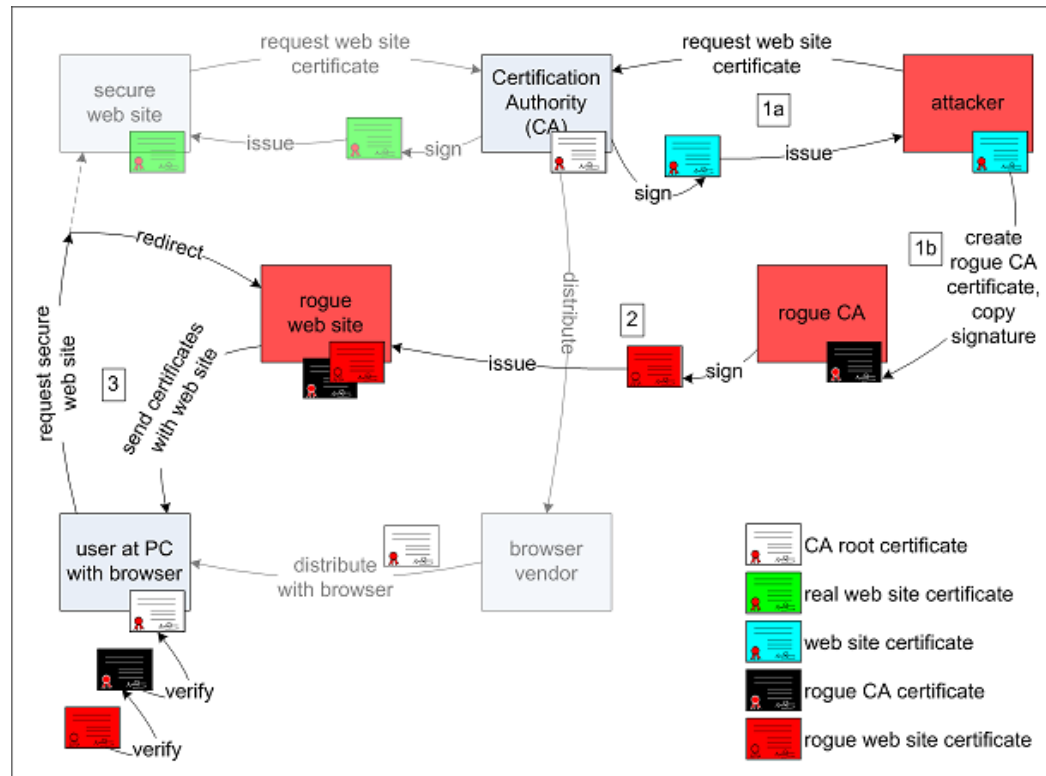
Sicherheit von MD5 (Forts.)

- 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

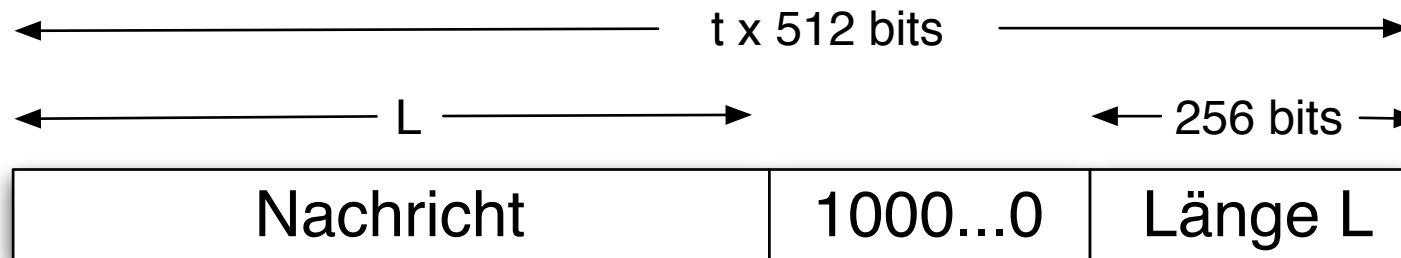
Whirlpool Hash-Funktion (2000)

- Entwickelt von P. Barreto und V. Rijmen
- im Rahmen des europäischen NESSI (**N**ew **E**uropean **S**chemes for **S**ignatures, **I**ntegrity, and **E**ncryption) entwickelt
- Struktur ähnlich zu AES, bzw. Rijndael
- Inzwischen Industriestandard im Rahmen von ISO/IEC 10118-3
 - Whirlpool ist nicht patentiert, die Referenzimplementierung ist Open Source
- 512 Bit lange Hashwerte bei max. Nachrichtenlänge 2^{256} Bits
- Design-Ziele:
 - Kollision zu finden benötigt $2^{n/2}$ Whirlpool-Operationen
 - Zu gegebenem Hash h eine Nachricht x zu finden mit $h = H(x)$ benötigt 2^n Whirlpool-Operationen
 - Zu gegebenen Hash h und geg. Nachricht m eine Nachricht x mit selbem Hashwert zu finden benötigt 2^n Whirlpool-Operationen
- Weit verbreitete Nutzung z.B. im Rahmen von TrueCrypt (2005)

Whirlpool: Überblick

- Whirlpool (WP(m)) arbeitet auf 512 Bit langen Teilblöcken M_i
- Verwendet Block Chiffre W
- Arbeitet intern mit 8x8 Byte Matrix CState

1. Expansion von m auf ein Vielfaches von 512 Bit; Aufteilen in Nachrichtenblöcke M_0 bis M_{t-1}

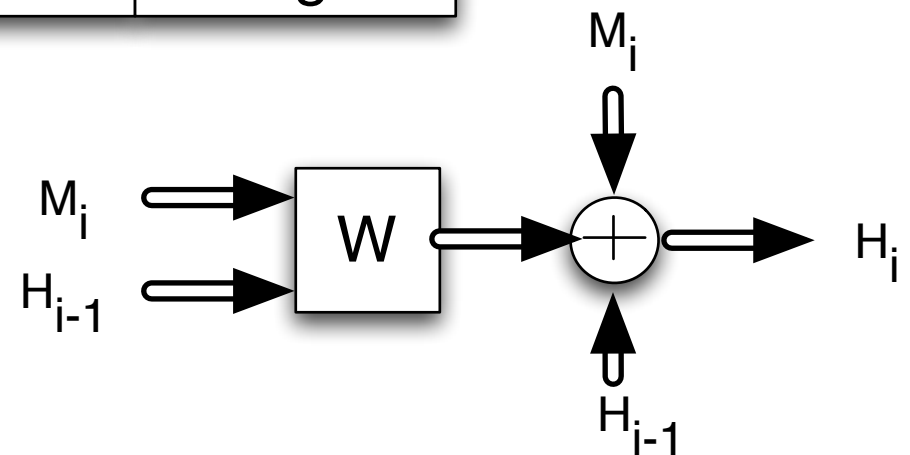


2. Initialisierung von $H_0 = 0$

3. For $0 \leq i \leq t-1$

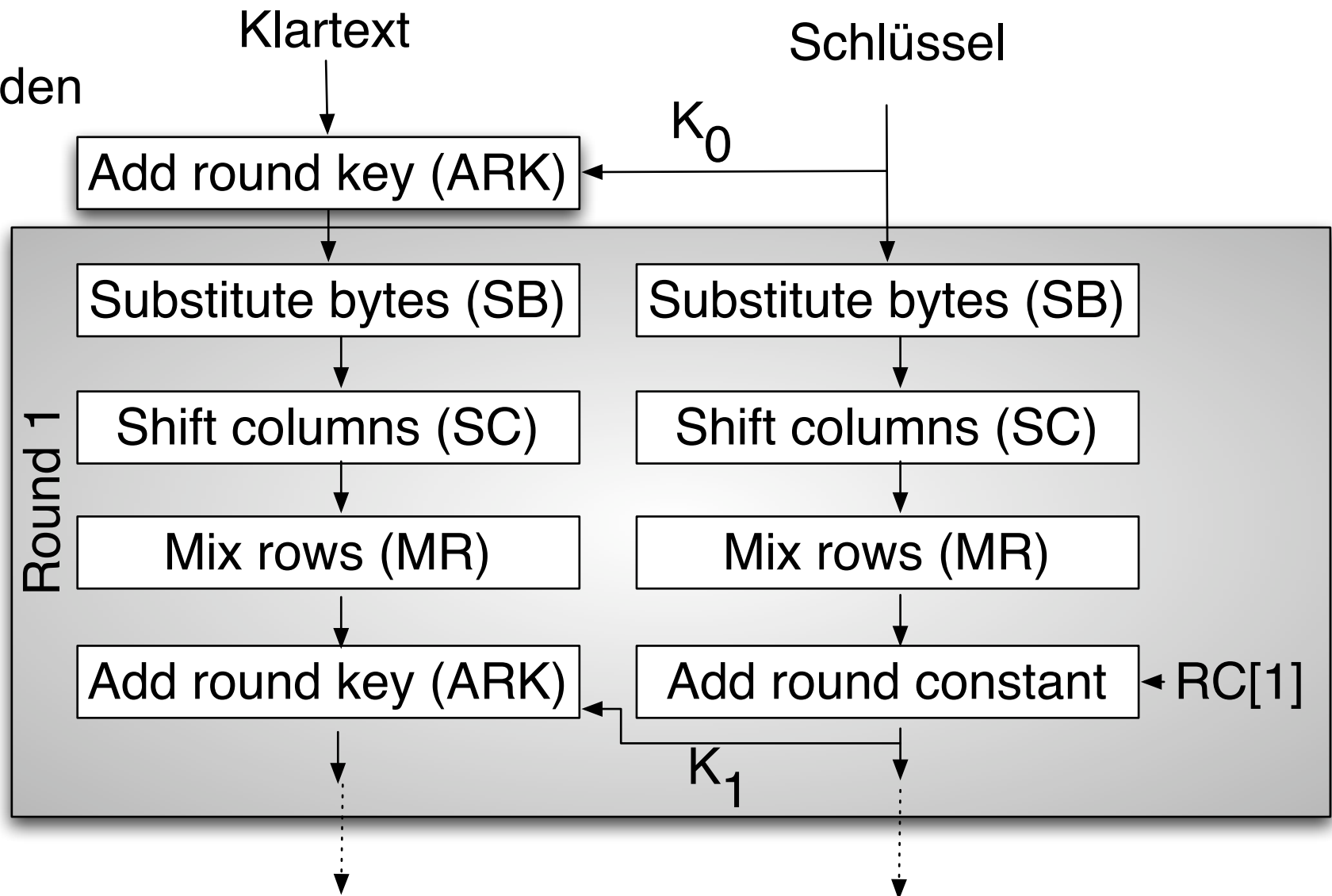
$$H_i = W_{H_{i-1}}(M_i) \oplus M_i \oplus H_{i-1}$$

4. $WP(m) = H_t$

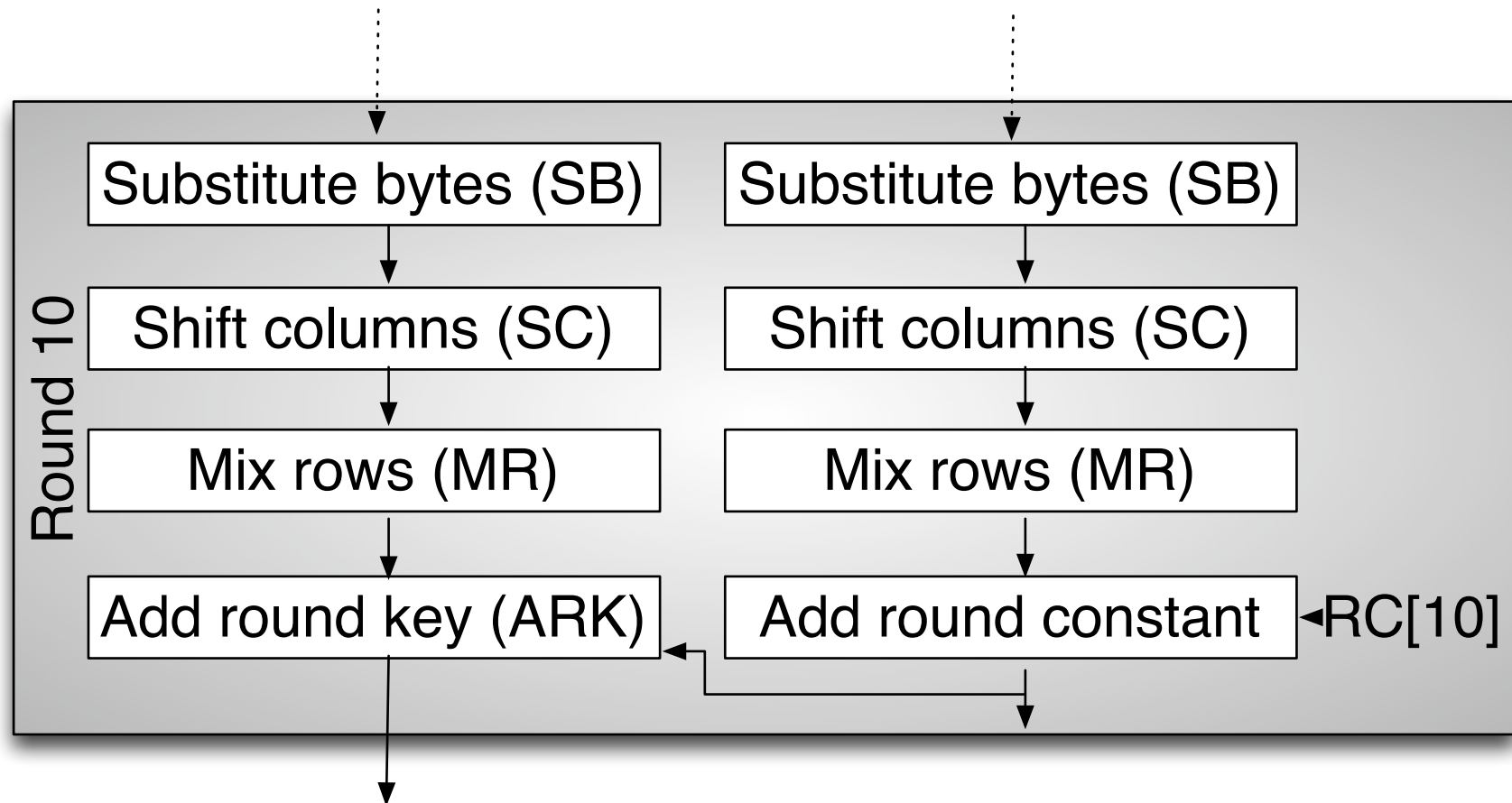


Whirlpool: Block Chiffre W

- Immer 10 Runden



Whirlpool: Block Chiffre W; Fortsetzung



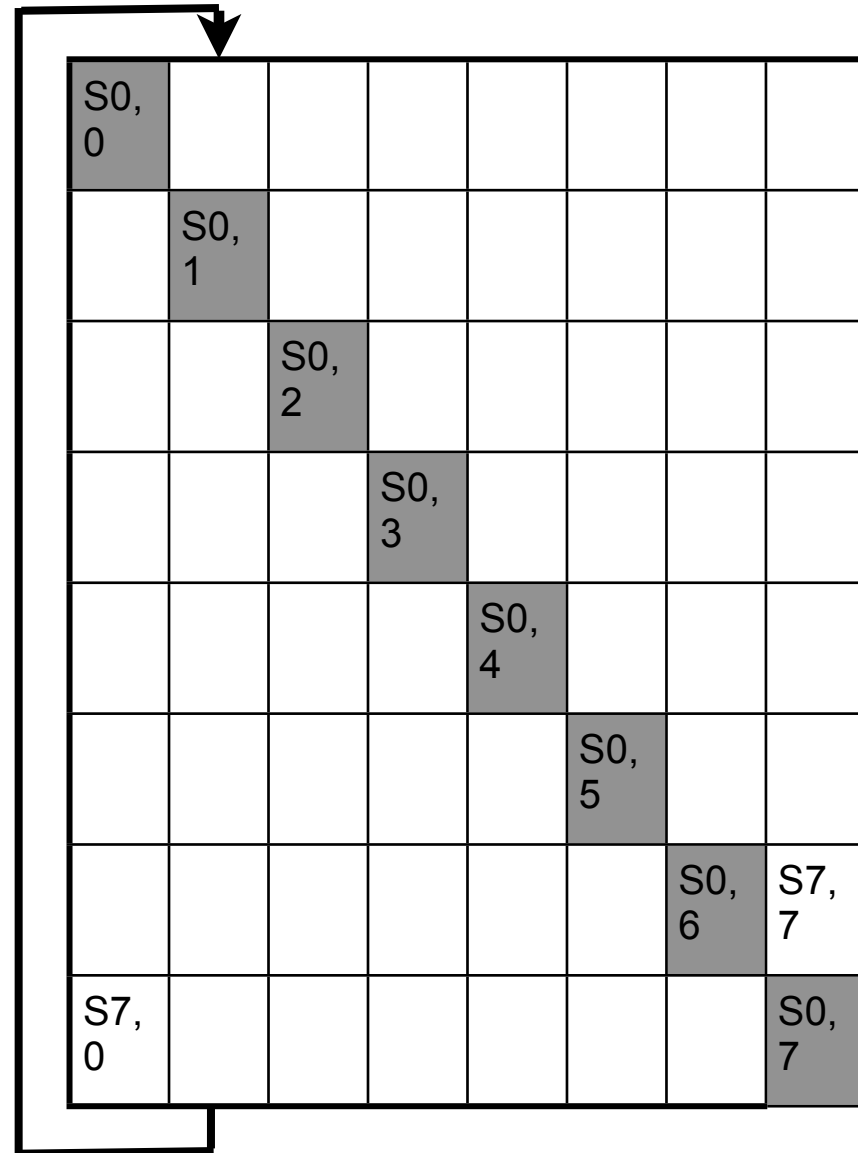
Whirlpool: Byte Substitution

- Mittels S-Box: auf Matrix-Elementen von CState
- 4 linken Bit bestimmen Spalte; 4 rechten Bit die Zeile

	00 _x	01 _x	02 _x	03 _x	04 _x	05 _x	06 _x	07 _x	08 _x	09 _x	0A _x	0B _x	0c _x	0d _x	0E _x	0F _x
00 _x	18 _x	23 _x	c6 _x	E8 _x	87 _x	B8 _x	01 _x	4F _x	36 _x	A6 _x	d2 _x	F5 _x	79 _x	6F _x	91 _x	52 _x
10 _x	60 _x	Bc _x	9B _x	8E _x	A3 _x	0c _x	7B _x	35 _x	1d _x	E0 _x	d7 _x	c2 _x	2E _x	4B _x	FE _x	57 _x
20 _x	15 _x	77 _x	37 _x	E5 _x	9F _x	F0 _x	4A _x	dA _x	58 _x	c9 _x	29 _x	0A _x	B1 _x	A0 _x	6B _x	85 _x
30 _x	Bd _x	5d _x	10 _x	F4 _x	cB _x	3E _x	05 _x	67 _x	E4 _x	27 _x	41 _x	8B _x	A7 _x	7d _x	95 _x	d8 _x
40 _x	FB _x	EE _x	7c _x	66 _x	dd _x	17 _x	47 _x	9E _x	cA _x	2d _x	BF _x	07 _x	Ad _x	5A _x	83 _x	33 _x
50 _x	63 _x	02 _x	AA _x	71 _x	c8 _x	19 _x	49 _x	d9 _x	F2 _x	E3 _x	5B _x	88 _x	9A _x	26 _x	32 _x	B0 _x
60 _x	E9 _x	0F _x	d5 _x	80 _x	BE _x	cd _x	34 _x	48 _x	FF _x	7A _x	90 _x	5F _x	20 _x	68 _x	1A _x	AE _x
70 _x	B4 _x	54 _x	93 _x	22 _x	64 _x	F1 _x	73 _x	12 _x	40 _x	08 _x	c3 _x	Ec _x	dB _x	A1 _x	8d _x	3d _x
80 _x	97 _x	00 _x	cF _x	2B _x	76 _x	82 _x	d6 _x	1B _x	B5 _x	AF _x	6A _x	50 _x	45 _x	F3 _x	30 _x	EF _x
90 _x	3F _x	55 _x	A2 _x	EA _x	65 _x	BA _x	2F _x	c0 _x	dE _x	1c _x	Fd _x	4d _x	92 _x	75 _x	06 _x	8A _x
A0 _x	B2 _x	E6 _x	0E _x	1F _x	62 _x	d4 _x	A8 _x	96 _x	F9 _x	c5 _x	25 _x	59 _x	84 _x	72 _x	39 _x	4c _x
B0 _x	5E _x	78 _x	38 _x	8c _x	d1 _x	A5 _x	E2 _x	61 _x	B3 _x	21 _x	9c _x	1E _x	43 _x	c7 _x	Fc _x	04 _x
c0 _x	51 _x	99 _x	6d _x	0d _x	FA _x	dF _x	7E _x	24 _x	3B _x	AB _x	cE _x	11 _x	8F _x	4E _x	B7 _x	EB _x
d0 _x	3c _x	81 _x	94 _x	F7 _x	B9 _x	13 _x	2c _x	d3 _x	E7 _x	6E _x	c4 _x	03 _x	56 _x	44 _x	7F _x	A9 _x
E0 _x	2A _x	BB _x	c1 _x	53 _x	dc _x	0B _x	9d _x	6c _x	31 _x	74 _x	F6 _x	46 _x	Ac _x	89 _x	14 _x	E1 _x
F0 _x	16 _x	3A _x	69 _x	09 _x	70 _x	B6 _x	d0 _x	Ed _x	cc _x	42 _x	98 _x	A4 _x	28 _x	5c _x	F8 _x	86 _x

Whirlpool: Shift Column

S0, 0	S0, 1	S0, 2	S0, 3	S0, 4	S0, 5	S0, 6	S0, 7
S7, 0							S7, 7



Whirlpool: Mix Row

- Matrixmultiplikation für jede Zeile von CState

$$\begin{bmatrix} s'_{i,0} \\ s'_{i,1} \\ s'_{i,2} \\ s'_{i,3} \\ s'_{i,4} \\ s'_{i,5} \\ s'_{i,6} \\ s'_{i,7} \end{bmatrix}^T = \begin{bmatrix} s_{i,0} \\ s_{i,1} \\ s_{i,2} \\ s_{i,3} \\ s_{i,4} \\ s_{i,5} \\ s_{i,6} \\ s_{i,7} \end{bmatrix}^T \cdot A = \begin{bmatrix} s_{i,0} \\ s_{i,1} \\ s_{i,2} \\ s_{i,3} \\ s_{i,4} \\ s_{i,5} \\ s_{i,6} \\ s_{i,7} \end{bmatrix}^T \cdot \begin{bmatrix} 01_x & 01_x & 03_x & 01_x & 05_x & 08_x & 09_x & 05_x \\ 05_x & 01_x & 01_x & 03_x & 01_x & 05_x & 08_x & 09_x \\ 09_x & 05_x & 01_x & 01_x & 03_x & 01_x & 05_x & 08_x \\ 08_x & 09_x & 05_x & 01_x & 01_x & 03_x & 01_x & 05_x \\ 05_x & 08_x & 09_x & 05_x & 01_x & 01_x & 03_x & 01_x \\ 01_x & 05_x & 08_x & 09_x & 05_x & 01_x & 01_x & 03_x \\ 03_x & 01_x & 05_x & 08_x & 09_x & 05_x & 01_x & 01_x \\ 01_x & 03_x & 01_x & 05_x & 08_x & 09_x & 05_x & 01_x \end{bmatrix}$$

Whirlpool: Add round key / constant

- CState wird mit K_i XOR-verknüpft
- Berechnung von K_i

- Berechnung von $RC[r]$ für Runde r :

$$RC[0] = K$$

for $1 \leq r \leq 10$

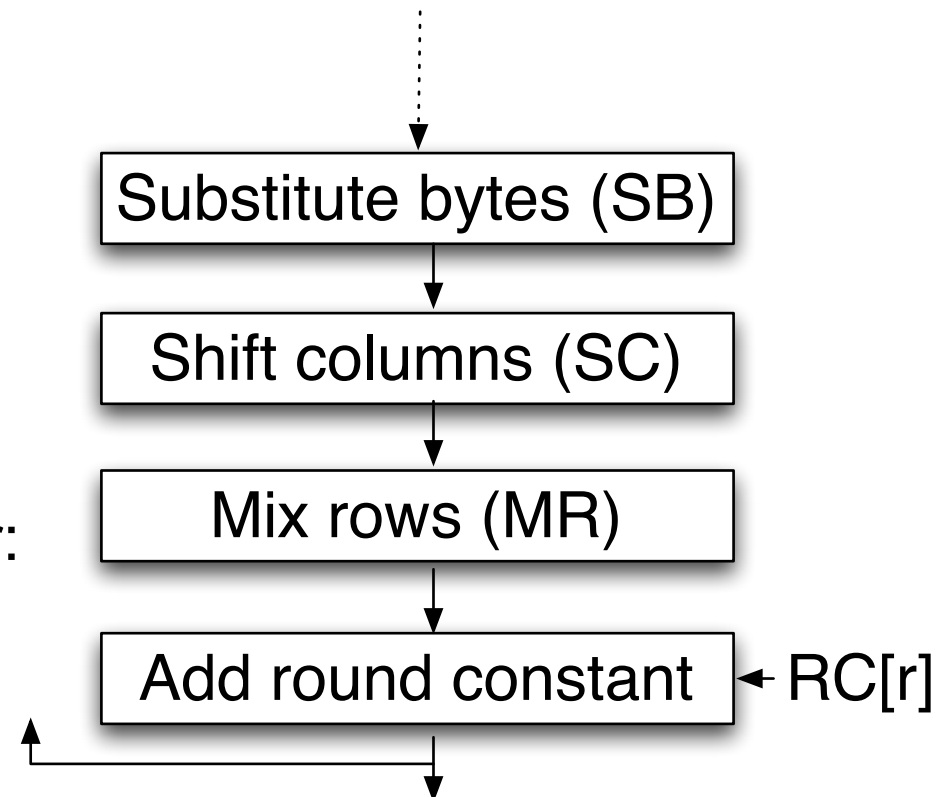
$$RC[r]_{0,j} = Sbox[8(r-1)+j]$$

für

$$0 \leq j \leq 7 \text{ u. } 1 \leq r \leq 10$$

$$RC[r]_{i,j} = 0 \quad \text{für}$$

$$1 \leq i \leq 7, 0 \leq j \leq 7 \text{ u. } 1 \leq r \leq 10$$



Vergleich Rijndael und W

	Rijndael	W
Blockgröße [bits]	128, 160, 192, 224 oder 256	immer 512
Rundenanzahl	10, 11, 12, 13 oder 14	immer 10
Schlüsselauswahl	ausgezeichneter Algorithmus (a priori)	Rundenfunktion von W
Reduktionspolynom	$x^8+x^4+x^3+x+1$ (0x11b)	$x^8+x^4+x^3+x^2+1$ (0x11d)

Sicherheit von Whirlpool

- Algorithmus wurde im Rahmen des NESSIE-Projekts evaluiert
 - Versteckte Schwächen wurden nicht gefunden
 - Statistische Analyse: Whirlpool sollte stochastisch sein
 - 512 Bit Länge bietet sonst nur SHA-512
 - Resistent gegenüber bekannten Angriffen
 - Bisher kein erfolgreicher Angriff
-
- ABER: Algorithmus noch relativ jung und bisher noch nicht so weit verbreitet