IT-Sicherheit

- Sicherheit vernetzter Systeme -

Kapitel 4: Grundlagen der Kryptologie

Inhalt

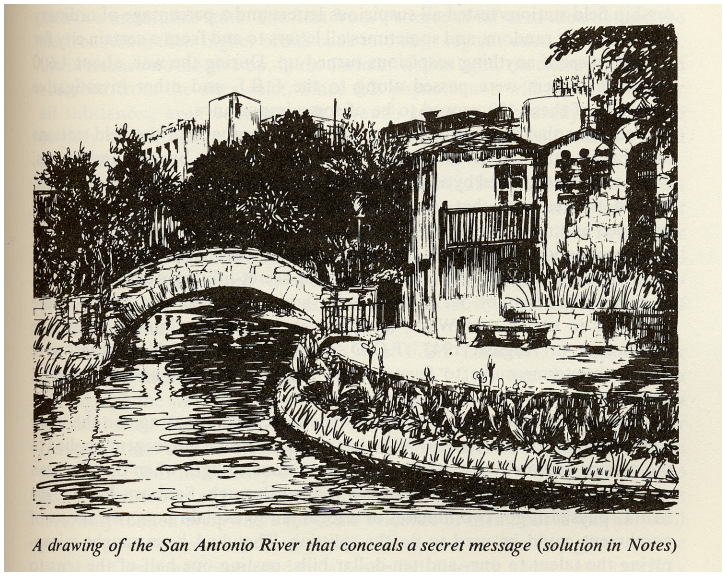
1. Kryptologie: Begriffe, Klassifikation
2. Steganographie
3. Kryptographie
   1. Begriffe und Definitionen
      - Kryptosystem
      - Substitution, Permutation
      - Symmetrische / asymmetrische Kryptosysteme
      - Kryptoanalyse
   2. Symmetrische Kryptosysteme
      - Data Encryption Standard (DES)
      - Advanced Encryption Standard (AES)
   3. Asymetrische Kryptosysteme
      - RSA
   4. Hybride Kryptosysteme
   5. Digitale Signatur
   6. Kryptographische Hash-Verfahren

Kryptologie: Begriffe, Klassifikation

- **Kryptographie**: Lehre von den Methoden zur Ver- und Entschlüsselung von Nachrichten
- **Kryptoanalyse, Kryptanalyse**: Wissenschaft von den Methoden zur Entschlüsselung ohne im Besitz den Schlüssels zu sein (Angriffe auf Kryptographische Verfahren)
- **Kryptologie** = Kryptographie + Kryptoanalyse
- **Kryptographische Protokolle**: Protokolle, die kryptographische Techniken verwenden, um z.B. Schlüssel auszutauschen, Kommunikationspartner zu authentisieren, ...
- **Steganographie** (verdecktes Schreiben): Methoden die bereits die Existenz der geheimen Nachricht verbergen (geheime Nachricht in anderen nicht geheimen „Nachrichten“ verbergen) Unterscheidung in **linguistische** und **technische** Steganographie

Linguistische Steganographie

- **Semagramme**: Nachrichten, die in **Details** von Schriften oder Bildern verborgen sind.
- Bsp. aus David Kahn: *The Codebreakers,* Scribner, 1996



A drawing of the San Antonio River that conceals a secret message (solution in Notes)

- Wo verbirgt sich die Nachricht?
- Wie lautet diese?

Linguistische Steganographie (Forts.)

- **Maskierung (Open Code):**
  Nachricht verborgen in offen übertragener, unverfänglicher Nachricht
  (z.B. Husten in „Wer wird Millionär“)
  - **Stichworte**: Begriff, Satzteil oder Satz mit vorher vereinbarter Bedeutung;
    z.B. *HIGASHI NO KAZE AME* („Ostwind, Regen“) im japanischen Wetterbericht - zwei mal wiederholt - sollte „Krieg mit USA“ bedeuten
- **Jargon, Millieu Code**:
  Sondersprachen oder Sonderzeichen beruflicher oder gesellschaftlicher Art
  - Bettler, Vagabunden und Gauner:
    Rotwelsch (Deutschland), Argot (Frankreich), ...
    z.B. „Schnee“ für Kokain; „Kies“ für Geld; „abstauben“ ,.....

  Für Zensoren durch „gestelzte“ Sprache relativ leicht erkennbar

Technische Steganographie

- Herodot (490 v.Chr.): Nachricht auf den rasierten Schädel eines Sklaven tätowiert
- Alle Arten von „Geheimtinten“
- Steganographie in digitalen Bildern; Beispiele mit `outguess`

Original

Steganographie

Steganographie in Bildern

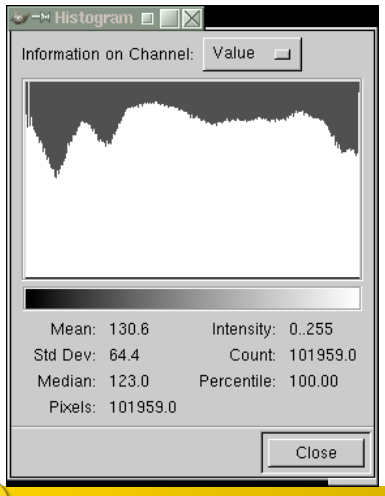
- **Cover** = Bild in das die Nachricht eingebettet wird
- Finde redundante Bits im Cover
  - Least Significant Bits
  - „Rauschen“
  - Nahe zusammenliegende Farben
- Kodieren der Nachricht in diesen redundanten Bits

- Steganographie führt zu “sehr geringen Veränderungen” im Bild

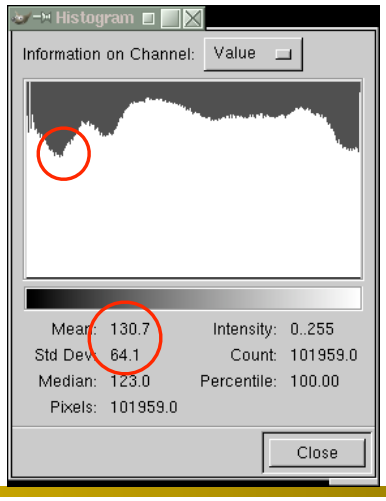
Steganographie; Veränderungen im Bild

- Histogramm:

Orginal

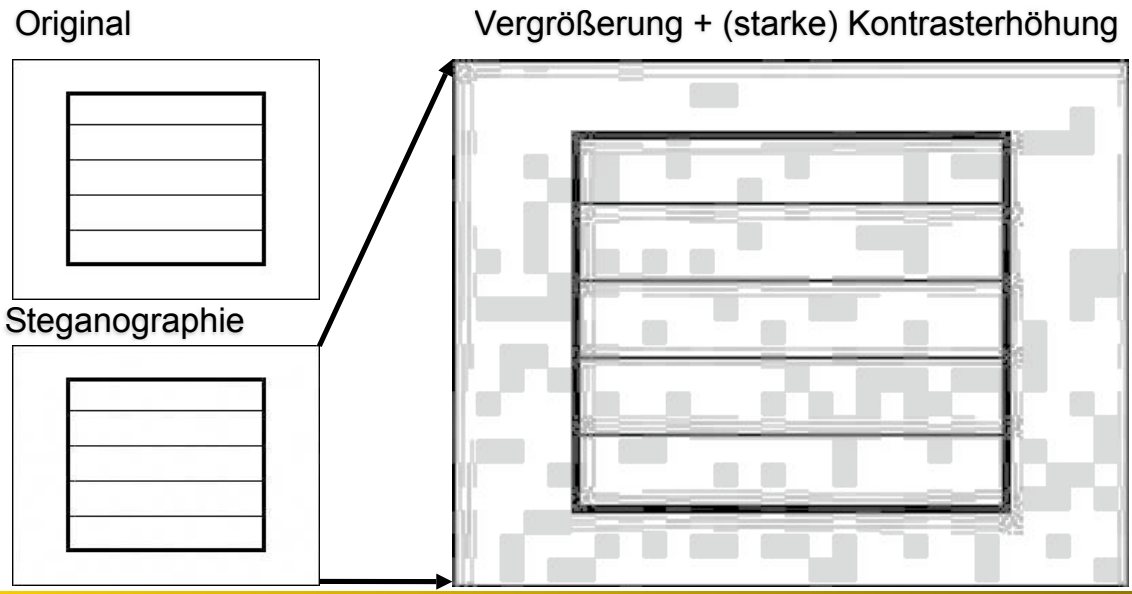


Steganographie



Steganographie; Merkmale

- Unterschiede bei “sehr strukturierten Bildern” mit hohem versteckten Datenvolumen evtl. erkennbar

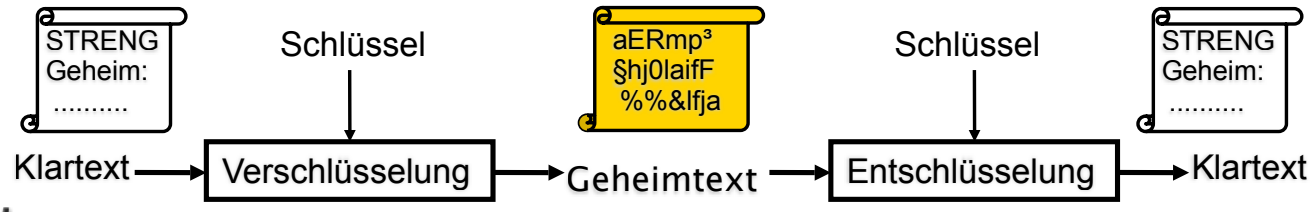


Inhalt

1. Kryptologie: Begriffe, Klassifikation
2. Steganographie
3. Kryptographie
   1. Begriffe und Definitionen
      - Kryptosystem
      - Substitution, Permutation
      - Symmetrische / asymmetrische Kryptosysteme
      - Kryptoanalyse
   2. Symmetrische Kryptosysteme
      - Data Encryption Standard (DES)
      - Advanced Encryption Standard (AES)
   3. Asymetrische Kryptosysteme
      - RSA
      - Digitale Signatur
   4. Hybride Kryptosysteme
   5. Kryptographische Hash-Verfahren

Kryptographie, Begriffe

- **Klartext** (**Plaintext**): die zu verschlüsselnde Nachricht
- **Geheimtext (Ciphertext):** verschlüsselte Nachricht
- **Verschlüsselung, Chiffrierung (Encryption)**: Vorgang der Klar- in Geheimtext überführt
- **Entschlüsselung, Dechiffrierung (Decryption):** Überführung von Geheim- in Klartext
- **Chiffriersystem (Cryptographic Algorithm, Cipher):** Algorithmisches Verfahren zur Ver- bzw. Entschlüsselung
- Benötigen **Schlüssel (Key)**



Kryptographisches System (Def.)

- Geg. zwei endliche Zeichenvorräte (Alphabete) A_1 und A_2
- Ein Kryptosystem (KS) ist gegeben durch ein Tupel

$$KS = (M, C, EK, DK, E, D)$$

1. Nicht leere endliche Menge von Klartexten $M \subseteq A_1^*$ mit A_1^* Menge aller Worte über dem Alphabet A_1
2. nicht leere endliche Menge von Krypto- bzw. Chiffrentexten $C \subseteq A_2^*$
3. der nicht leeren Menge von Verschlüsselungsschlüsseln EK
4. der nicht leeren Menge von Entschlüsselungsschlüsseln DK sowie einer Bijektion $f: EK \rightarrow DK$ Diese assoziiert zu jedem Verschlüsselungsschlüssel $K_E \in EK$ einen dazu passenden Entschlüsselungsschlüssel $K_D \in DK$, d.h. $f(K_E) = K_D$

Kryptographisches System (Def.); Forts.

- Kryptosystem (KS)

$$KS = (M, C, EK, DK, E, D)$$

5. Dem injektiven Verschlüsselungsverfahren

$$E : M \times EK \rightarrow C$$

6. Dem Entschlüsselungsverfahren

$$D : C \times DK \rightarrow M$$

mit der Eigenschaft, dass für zwei Schlüssel $K_E \in EK$ und $K_D \in DK$ mit $f(K_E) = K_D$ gilt:

$$\forall m \in M : D(E(m, K_e), K_d)) = m$$

D.h. ein bel. Klartext m der mit einem Verschlüsselungs-schlüssel verschlüsselt wurde, kann mit dem passenden Entschlüsselungsschlüssel wieder entschlüsselt werden

Kryptosystem, Bsp.: Substitution

- **Substitution**: $f : A_1^n \rightarrow A_2^n$
- Alphabete: $A_1 = \{a, b, \ldots, z\} (= Z_{25}); A_2 = \{1,2,3,4,5\}$
- Verschlüsselungsverfahren: $E : A_1^1 \rightarrow A_2^2$
- Schlüssel $K_E = K_D$

	1	2	3	4	5
1	a	b	c	d	e
2	f	g	h	i	k
3	l	m	n	o	p
4	q	r	s	t	u
5	v	w	x	y	z

- Bsp.:
  4211421332435434 51 = iailhouse (Jailhouse)

Kryptosystem, Bsp.: Permutation

- **Permutation** als Spezialfall der Substitution: $f : A^n \rightarrow A^n$
  gleiche Wortlänge; gleiche Alphabete $A_1 = A_2 = \{a,b,...,z\}$
- $K_E = K_D$ (NEWYORK,1) (+ Alg. zur Anwendung)
  (Zur besseren Lesbarkeit, werden Chiffrentexte trotzdem oft in Großbuchstaben dargestellt.)

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
N	E	W	Y	O	R	K	A	B	C	D	F	G	H	I	J	L	M	P	Q	S	T	U	V	X	Z

  Zyklenschreibweise:
  (a,n,h) (b,e,o,i) (c,w,u,s,p,j) (d,y,x,v,t,q,l,f,r,m,g,k)
- Bsp.:
  QAOWI YOEMO NDOMP = thecodebreakers
  Chiffrentext wird in Blöcken übertragen
  Leer- und Satzzeichen werden nicht kodiert (Leerzeichen noch häufiger als „e“)

Kryptosystem: Symmetrische Verfahren

- Ver- und Entschlüsselungsschlüssel gleich, oder leicht voneinander ableitbar
- Kommunikationspartner teilen **gemeinsamen, geheimen Schlüssel (symmetrisch)**
- Setzt vorherige Verständigung (**Schlüsselaustausch**) voraus
- Protokoll:
  1. Alice und Bob vereinbaren („**out of band**“) gemeinsamen Schlüssel:
     $K_E = K_D = K_{A,B}$
  2. Alice verschlüsselt m: $C = E(m, K_{A,B})$ und sendet C an Bob
  3. Bob entschlüsselt C:

$$m = D(C, K_{A,B}) = D(E(m, K_{A,B}), K_{A,B})$$

- Beispiele: DES, AES, IDEA,.....

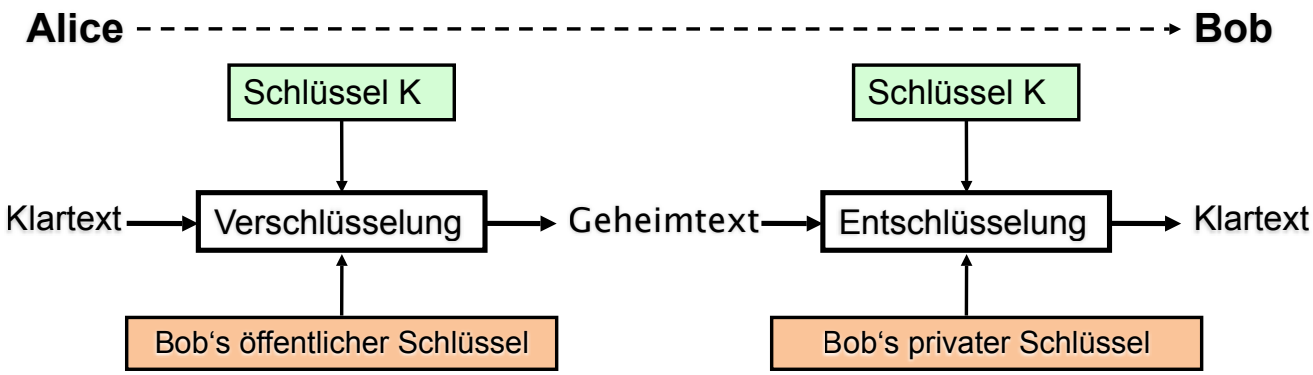
Kryptosystem: Asymmetrische Verfahren

- Jeder Partner besitzt **Schlüsselpaar** aus
  - persönlichem, **geheim** zu haltendem **Schlüssel** (*private key*) (wird NIE übertragen)
  - und **öffentlich** bekannt zu gebendem **Schlüssel** (*public key*) (kann über unsichere und öffentliche Kanäle übertragen werden)
- Protokoll:
  1. Alice und Bob erzeugen sich Schlüsselpaare:
     $(K_E^A, K_D^A); (K_E^B, K_D^B)$
  2. Öffentliche Schlüssel werden öffentlich zugänglich gemacht
  3. Alice will m an Bob senden; dazu benutzt sie B‘s öffentlichen Schlüssel
     $C = E(m, K_E^B)$
  4. Bob entschlüsselt die Nachricht mit seinem privaten Schlüssel:

     $m = D(C, K_D^B) = D(E(m, K_E^B), K_D^B)$
- Bsp.: RSA, ElGamal,...

Vergleich Symmetrische / Asymmetrische Verfahren

	Symmetrisch	Asymmetrisch
Schlüsselaustausch	Sicherer Kanal erforderlich	öffentlich
Schlüssellänge	128 bis 256 Bit	1024 bis 4096 Bit
Geschwindigkeit		Faktor 100 bis 1000 langsamer



Kryptoanalyse

- Wissenschaft von Methoden zur Entschlüsselung **ohne** Schlüssel
- Klassen kryptographischer Angriffe:
  - **Brute force; exhaustive search:** vollständiges Durchsuchen des Schlüsselraumes
  - **Klartext Angriff (ciphertext-only):** Dem Analytiker stehen mehrere Chiffren zur Verfügung. Ziel: Schlüssel und/oder Klartext berechnen
  - **Bekannter Klartext (known-plaintext):** Analytiker kennt Klartext-/ Chiffren-Kombinationen die mit selbem Schlüssel verschlüsselt wurden. Ziel: Schlüssel brechen oder Algorithmus der jede mit dem Schlüssel verschlüsselte Nachricht entschlüsseln kann
  - **Gewählter Klartext (chosen-plaintext):** Analytiker kann selber Klartexte wählen und diese verschlüsseln lassen.
  - **Gewählte Chiffre (known-ciphertext):** Angreifer kann sich zu ausgewählten Chiffren, den Klartext berechnen lassen
- Weitere Informationen: Vgl. F.L. Bauer: Entzifferte Geheimnisse

Einschub: Abschätzung Brute-Force Angriff

- Der Schlüssellänge sei 128 Bit
- Ihr Rechner ist in der Lage 50.000 Verschlüsselungs-operationen pro Sekunde durchzuführen
- Wie viele Jahre dauert ein Brute-Force-Angriff?
- Schlüsselraum $S = 2^{128} \approx 3,4 \cdot 10^{38}$
  340.282.366.920.938.463.463.374.607.431.768.211.456
- 1 Jahr hat rund 31.536.000 Sekunden
- (S / 50.000) / 31.536.000 Sekunden $\approx 2 \cdot 10^{27}$
  215.805.661.416.120.283.779.410.583,099802 Jahre
- Wieviele Schlüssel müssen Sie pro Sek. berechnen um „nur“ 100 Jahre zu brauchen?
- ~ 107.902.830.708.060.141.889.705.291.549,901
  $\approx 1,1 \cdot 10^{30}$

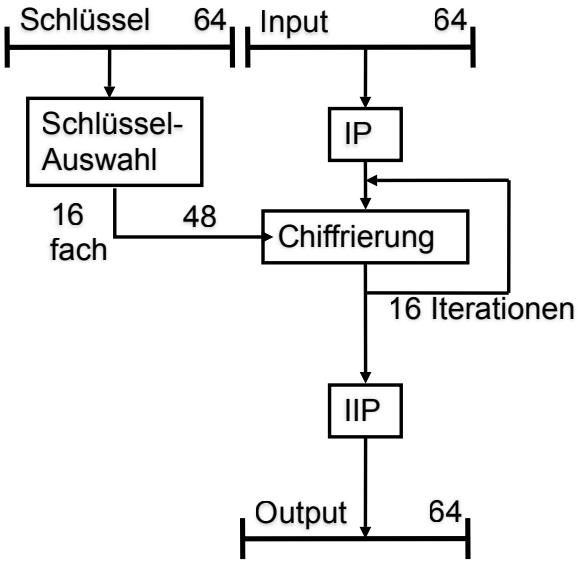
Inhalt

1. Kryptologie: Begriffe, Klassifikation
2. Steganographie
3. Kryptographie
   1. Begriffe und Definitionen
      - Kryptosystem
      - Substitution, Permutation
      - Symmetrische / asymmetrische Kryptosysteme
      - Kryptoanalyse
   2. Symmetrische Kryptosysteme
      - Data Encryption Standard (DES)
      - Advanced Encryption Standard (AES)
   3. Asymetrische Kryptosysteme
      - RSA
      - Digitale Signatur
   4. Hybride Kryptosysteme
   5. Kryptographische Hash-Verfahren

DES (Data Encryption Standard)

- 1977 vom NBS (National Bureau of Standards) heute National Institute of Standards (NIST) in USA zum Standard erklärt
- 2002 durch AES (Advanced Encryption Standard) ersetzt
- DES entwickelt von IBM aus dem 128 Bit Verfahren LUCIFER
- Klassifikation:
  - Symmetrisches Verfahren
  - mit Permutation, Substitution und bitweiser Addition modulo 2
  - Blockchiffre mit 64 Bit großen Ein- und Ausgabeblöcken
  - Schlüssellänge 64 Bit, davon 8 Paritätsbits, d.h. effektive Schlüssellänge 56 Bit

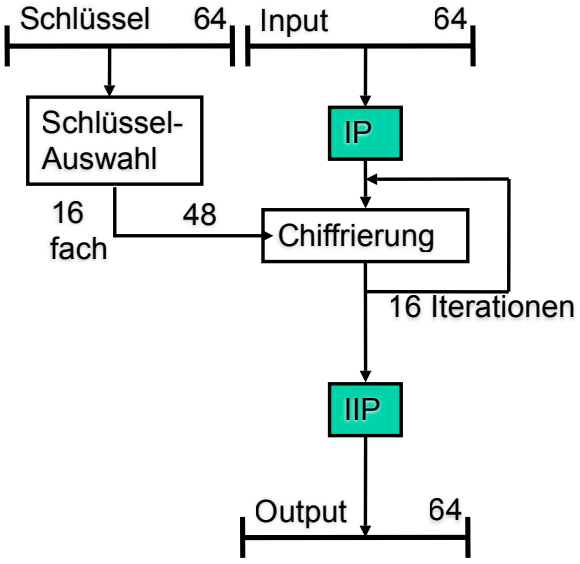
DES Funktion: Grundschema



- Ablauf der Verschlüsselung
1. Initialpermutation (IP) des Input-Blocks
2. 16 schlüsselabhängige Iterationen
   - 48 Bit lange Teilschlüssel
   - werden aus 64 Bit Schlüssel generiert
3. Inverse Initialpermutation (IIP) als Ausgangspermutation

- Entschlüsselung analog zur Verschlüsselung mit Schlüssel in umgekehrter Reihenfolge im Schritt 2.

DES Funktion: Grundschema



- Wie arbeiten Initialpermutation (IP) und Inverse Initialpermutation (IIP)?

DES: IP und IIP

- Initialpermutation IP

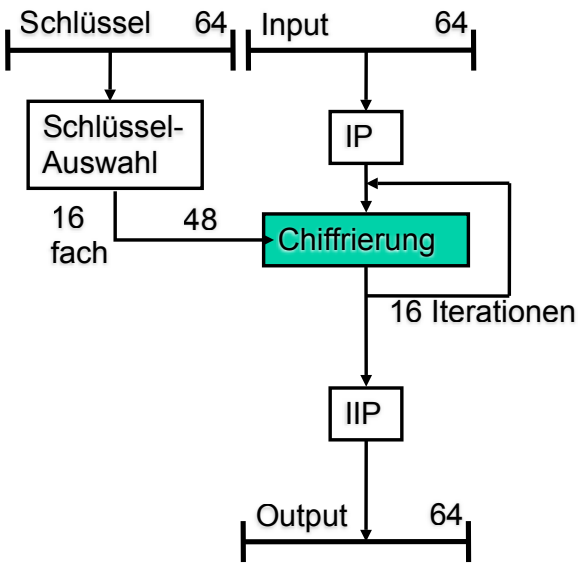
58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

- Inverse Initialpermutation IIP

40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

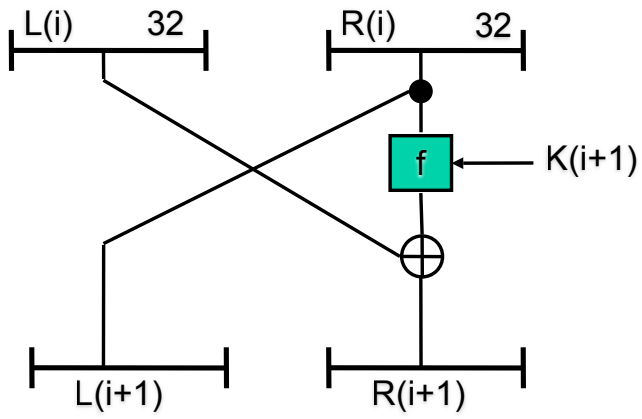
- D.h. aus Bit 58 des Input wird Bit 1, aus Bit 50 wird Bit 2,....., aus Bit 7 wird Bit 64

DES Funktion: Grundschema



DES Funktion: Verschlüsselungsiteration

- Ein Schritt (Runde) der Chiffrierung:

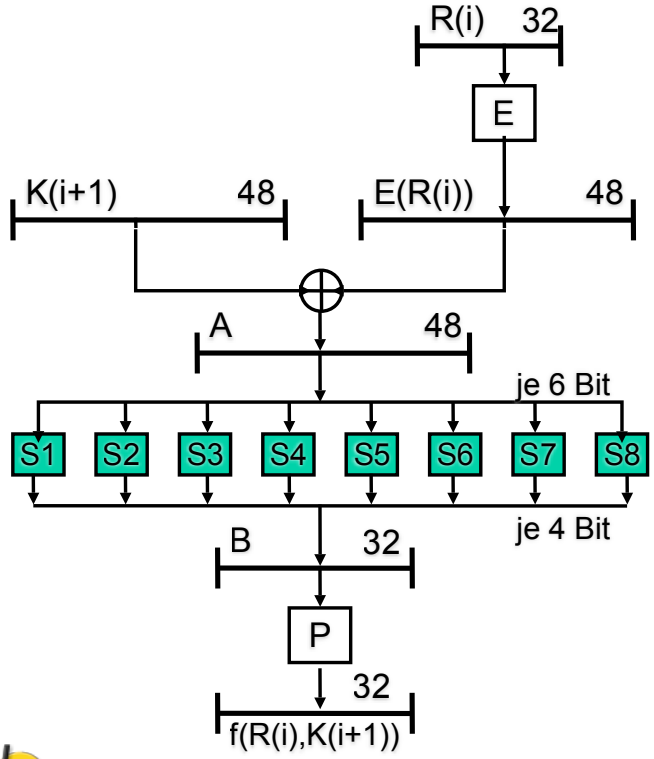


⊕ Addition modulo 2; entspr. XOR

- Verschlüsselungsblock (64 Bit) wird in linken (L) und rechten (R) Block a 32 Bit aufgeteilt
- Anwendung der Verschlüsselungs-iteration:
  L(0) = L und R(0) = R
  L(i+1) = R(i)
  R(i+1) = L(i) XOR f(R(i),K(i+1))

  für i=0,....,15
- Funktion f stellt Kern des Verfahrens dar.

DES Funktion f



- Rechter 32 Bit Input Block wird mittels Expansion E auf 48 Bit expandiert
- XOR Verknüpfung mit dem (Runden-) Schlüssel zum 48 Bit Block A
- A wird in 8 Blöcke zu 6 Bit aufgeteilt
- Jeder dieser Blöcke wird durch S-Box (Substitution) in 4 Bit Ausgabeblöcke (nichtlinear) abgebildet
- Konkatenation der acht 4 Bit Blöcke ergibt Block B der noch der Ausgangspermutation P unterworfen wird

Expansion E und Permutation P

- Expansion E:

32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29

  - Bit 32 aus R(i) wird Bit 1 von E(R(i))

- Ausgangspermutation P

16	7	20	21
29	12	28	17
1	15	23	26
5	18	31	10
2	8	24	14
32	27	3	9
19	13	30	6
22	11	4	25

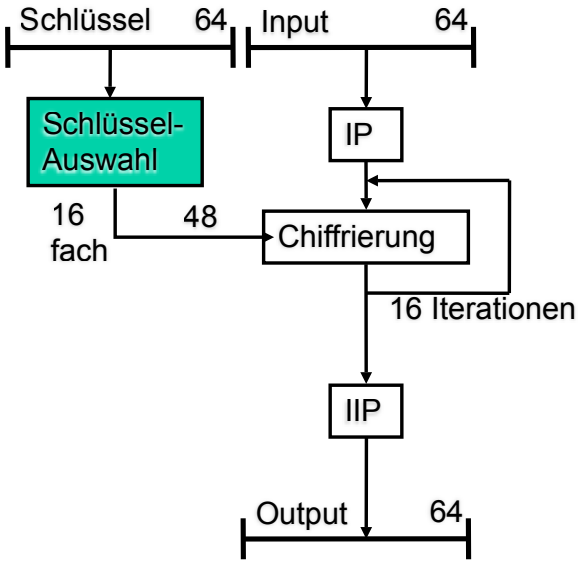
DES S-Boxen

- 6 Bit Input Block (i1,i2,i3,i4,i5,i6) wird auf 4 Bit Outputblock (o1,o2,o3,o4) abgebildet:
  - Redundante Bits (i1,i6) des Inputblocks bestimmten die Zeile der entspr. S-Box
  - Bits (i2,i3,i4,i5) bestimmen Spalte
  - Element in der Matrix bestimmt Outpublock
- Bsp. S-box S1:

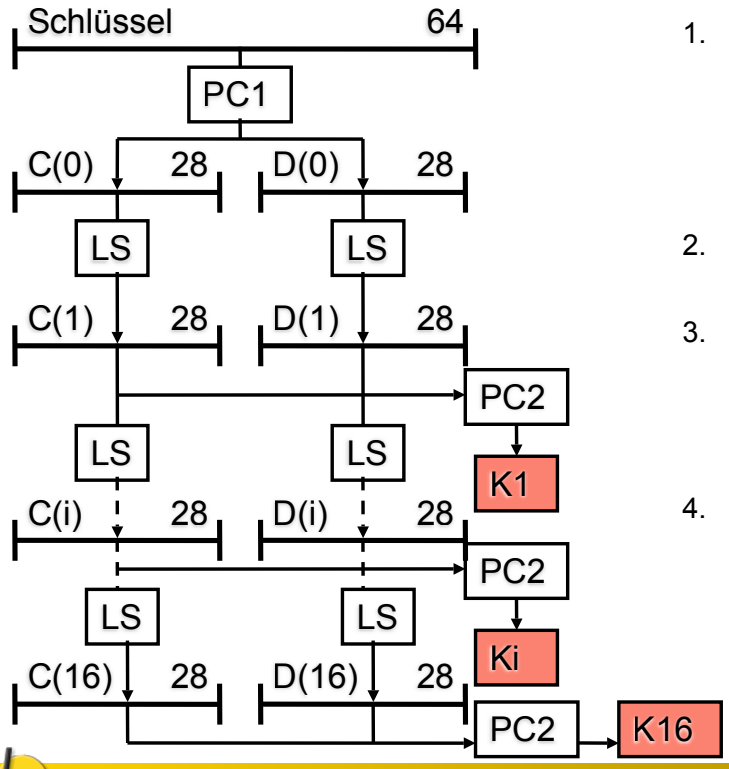
- Beispiel
  - S-Box S1
  - Input (0,1,1,0,1,1)
  - Zeile (0,1) = 1
  - Spalte (1,1,0,1) = 13
  - Output (5) = (0,1,0,1)

(i1,i6) \ (i2,i3,i4,i5)	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

DES Funktion: Grundschema

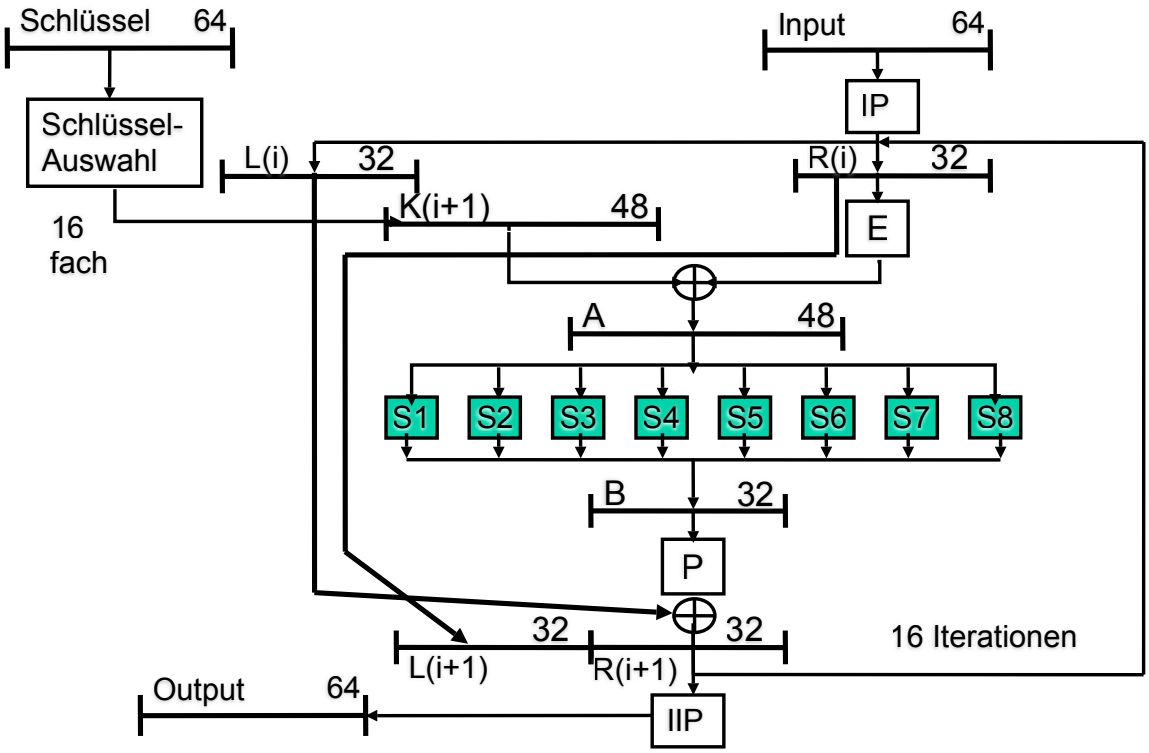


DES Schlüsselauswahl



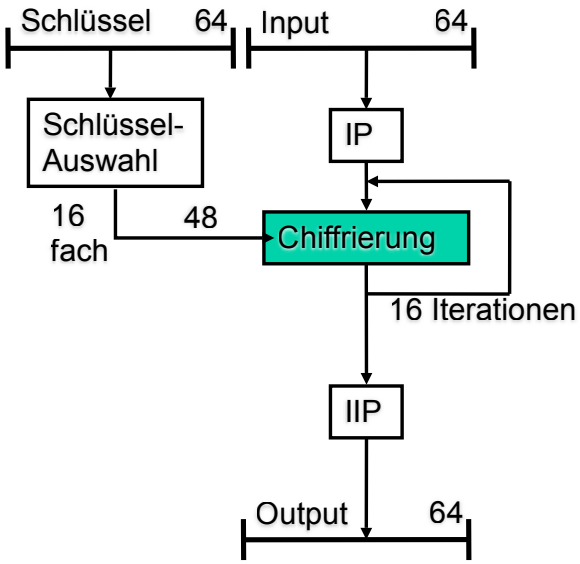
1. 64 Bit Schlüssel wird Permuted Choice 1 (PC1) unterworfen:
   - Key wird auf 56 relevante Bits gekürzt (jedes 8. Bit Parity)
   - Key wird permutiert
2. Schlüssel wird in zwei Teile a 28 Bit aufgeteilt
3. Blöcke werden zyklisch nach links geschiftet
   - In Runde 1,2,9 u. 16 um 1 Bit
   - 2 Bit sonst
4. Teilblöcke werden zusammen-gefasst und PC2 unterworfen:
   - Entfernen der Bits 9,18,22,25,35,38,43 u. 56
   - Permutation der verbleibenden 48 Bit

DES: Zusammenfassung



DES Entschlüsselung

- Es gilt D = E
- DES wird für Ver- und Entschlüsselung prinzipiell gleich verwendet, außer
- Umkehrung der Schlüsselreihenfolge
- In Runde i wird K(17-i) verwendet



DES Stärken und Schwächen

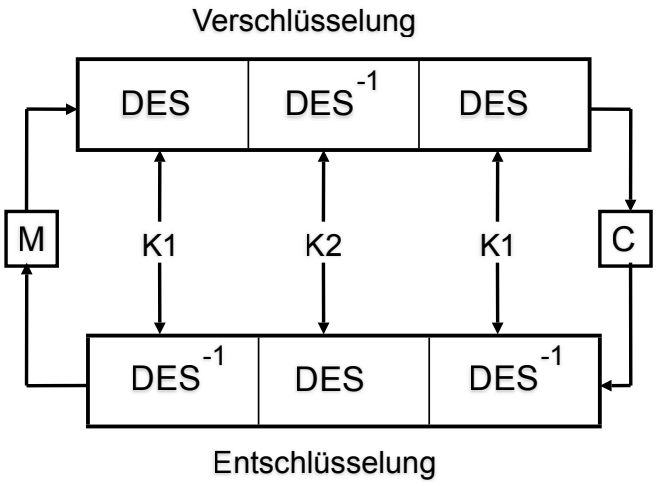
- Starker **Avalanche-Effekt** (Lawineneffekt; große Streuung) durch S-Boxen und Permutation P: Kleine Änderungen in der Eingabe, die nur eine S-Box betreffen breiten sich schnell aus. Eine Änderung eines Bits in der Eingabe verursacht eine Änderung von durchschnittlich 50% der Ausgabe
- 16 Iterationen: Known-plaintext Angriff auf DES mit < 16 Runden immer effizienter als Brute force
- Stark gegen analytische Angriffe Differentielle Kryptanalyse braucht 2^{47} gewählte Klartexte (chosen-pl.t.)
-

- (teilweise) geheimes Design
- Deutlich zu geringe Schlüssellänge Schlüsselraum der Größe 2^{56}
- 4 schwache Schlüssel mit: DES(DES(x,K),K) = x
- 6 semi-schwache Schlüsselpaare: DES(DES(x,K),K') = x
- Differentielle Kryptanalyse lässt sich in der Komplexität reduzieren auf 2^{37}
- Optimiert auf Implementierung in Hardware: Initialpermutation IP und inverse IP verbessern die Sicherheit nicht, sondern erhöhen nur den Aufwand für Software-Implementierungen

DES Varianten: Double und Tripple DES

- Double-DES:
  - DES(DES(m,K1),K2)
- Erwartete Komplexität:
  - bei Schlüssellänge n 2^{2n}
- Merkle und Hellman haben gezeigt, dass ein Known-Plaintext Angriff möglich ist mit 2^{n+1}
  - Allerdings liegt der Speicherverbrauch bei 2^n Blöcken, bei DES 2^56 * 64 Bit, d.h. 10^17 Byte, d.h. im Bereich von Peta Byte
- D.h. doppelte Ausführung von DES bringt **KEINE** merkliche Steigerung der Sicherheit

- Triple-DES



- Komplexität 2^{2n}

Einschub: US-CERT Alert TA07-334A

- Apple QuickTime RTSP Buffer Overflow
- Systems affected:
  - Apple Quicktime (prior and including Version 7.3) for Windows and Mac OS X
- Description
  - Buffer overflow in RTSP (Real Time Streaming Protocol) stream handling
  - Quicktime is part of iTunes; therefore iTunes is also affected
  - specially crafted HTML Page, email or something else which opens an specially craftes RTSP Stream with Quicktime
- Impact:
  - Remote Code Execution; DoS
- Solution:
  - No Patch available !!!!!!!
  - Workaround (see next slide)

Einschub: US-CERT Alert TA07-334A

- Workaround:
  - Block RTSP Port 554/tcp
  - Disable file association for QuickTime Files (registry change in Windows)
  - Disable QuickTime Active X Control in Internet Explorer
  - Disable QuickTime plug-in for Mozilla Firefox
  - Disable Java Script
  - Secure your web browser
  - Do not access Quick Time files from untrusted source

Advanced Encryption Standard (AES); Historie

- 1997 öffentliche Ausschreibung des Dept. Of Commerce (Request for Candidate Algorithms for AES):
  - Algorithmus öffentlich und nicht klassifiziert,
  - Mindestblocklänge 128 Bit, Schlüssellänge 128, 192 und 256 Bit
  - Weltweit frei von Lizenzgebühren,
  - nutzbar für 30 Jahre, effizient sowohl in SW als auch versch. HW
- Dreistufiges (Vor-)Auswahlverfahren
  1. Pre-Round 1 (1/97 – 7/98)
     - Call for Candidates
  2. Round 1 (8/98 – 4/99)
     - Vorstellung, Analyse und Test
     - Auswahl der Kandidaten für Round 2
  3. Round 2 (8/99 – 5/2000)
     - Analyse und Tests
     - Auswahl der Finalisten
- Endgültige Auswahl durch NIST

AES Kandidaten

- Pre-Round 1: 21 Kandidaten, 6 aus formalen Gründen abgel.

Algo.	Land	Autor(en)	Algo.	Land	Autor(en)
CAST-256	Kanada	Entrust	MAGENTA	Deutschland	Deutsche Telekom
CRYPTON	Korea	Future Systems	MARS	USA	IBM
DEAL	Kanada	R. Outbridge, L. Knudsen	RC6	USA	RSA Laboratories
DFC	Frankreich	CNSR	RIJNDAEL	Belgien	J. Daeman, V. Rijmen
E2	Japan	NTT	SAFER+	USA	Cylink
FROG	Costa Rica	TecApro	SERPENT	UK, Norwegen, Israel	R. Anderson, E. Biham u.a.
HPC	USA	R.Schroeppel	TWOFISCH	USA	B. Schneier, J. Kelsey, u.a.
LOKI97	Australien	L. Brown, J. Pieprzyk u.a.			

AES: Round 2 Finalisten und Ergebnis

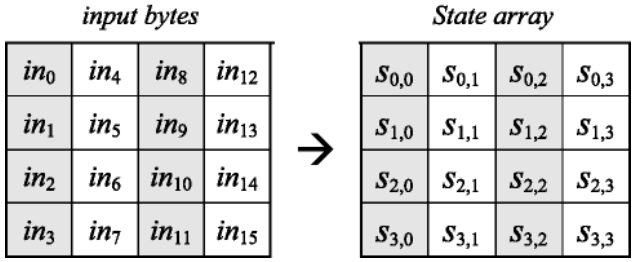
- Finalisten der Runde 2:

MARS	USA	IBM
RC6	USA	RSA Laboratories
RIJNDAEL	Belgien	J. Daeman, V. Rijmen
SERPENT	UK, Norwegen, Israel	R. Anderson, E. Biham, L. Knudsen
TWOFISCH	USA	B. Schneier, J. Kelsey, D. Whiting, D. Wagner, C. Hall, N. Feruson

- 2. Oktober 2000: Rijndael wird gewählt
- 26. Nov. 2001: Veröffentlichung des FIPS-197 (Federal Information Processing Std.) durch NIST (National Institute for Standards and Technology)
- 26. Mai 2002: Inkrafttreten des Standards
- Informationen: `www.nist.gov/aes` mit Link auf AES-Homepage

AES

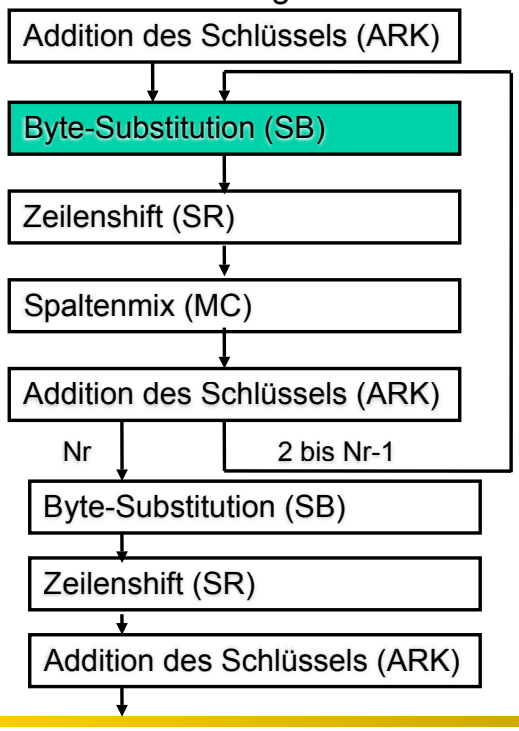
- Variable Blocklänge: 32*Nb Bits
- Variable Schlüssellänge: 32*Nk Bits
- Nb und Nk aus {4-8}; im Standard eingeschränkt auf 4,6 oder 8
- Variable Rundenanzahl Nr mit Nr = max(Nb, Nk) + 6
- Folgende Beispiele für Nb=Nk=4
- Rijndal arbeit auf sog. States:
  Input Bytes in0,in1,...,in15 werden in State kopiert:



- Runden arbeiten auf den States

AES: Ver- und Entschlüsselung

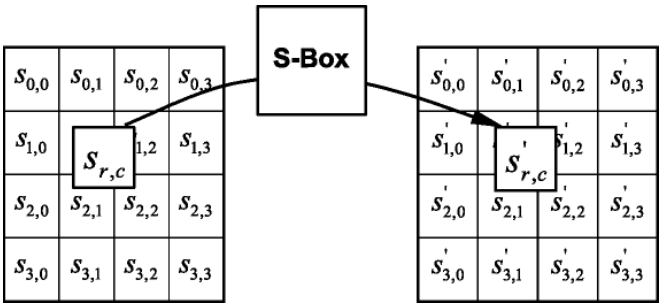
- Verschlüsselung



- Runden arbeiten auf sog. States
- Verschlüsselung
  - Ablauf der Runden 1 bis Nr-1:
    1. Byte-Substitution (`SubBytes`, SB)
    2. Zeilenshift (`ShiftRows`, SR)
    3. Spaltenmix (`MixColumns`, MC)
    4. Addition d. Rundenschlüssels (`AddRoundKey`, ARK)
- Entschlüsselung:
  - Runde 1 bis Nr-1:
    1. Addition d. Rundenschlüssels
    2. Inverser Spaltenmix
    3. Inverser Zeilenshift
    4. Inverse Byte-Substitution
- Letzte Runden Nr analog; aber **ohne** Spaltenmix

AES: Bytesubstitution (`SubBytes()`)

- Angewandt auf jedes Byte des State



- Byte S(r,c) wird als Kodierung eines Polynoms im Körper GF(2^8) aufgefasst
- Substitution:
  1. Bestimme multiplikatives Inverses von S(r,c) in GF(2^8)
  2. Affine Transformation des Inversen:
     - Matrixmultiplikation
     - XOR mit {63}

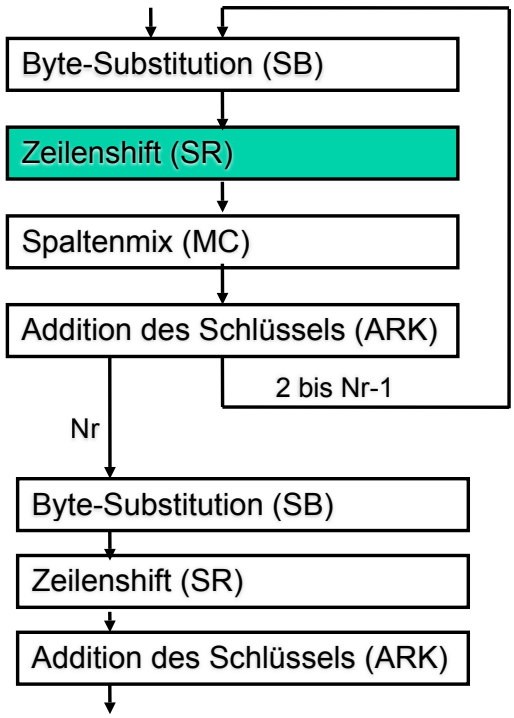
AES: Bytesubstitution Implementierung

- Vorausberechnung für alle 256 möglichen Polynome und Speicherung in S-Box (aus FIPS197)

x \ y	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

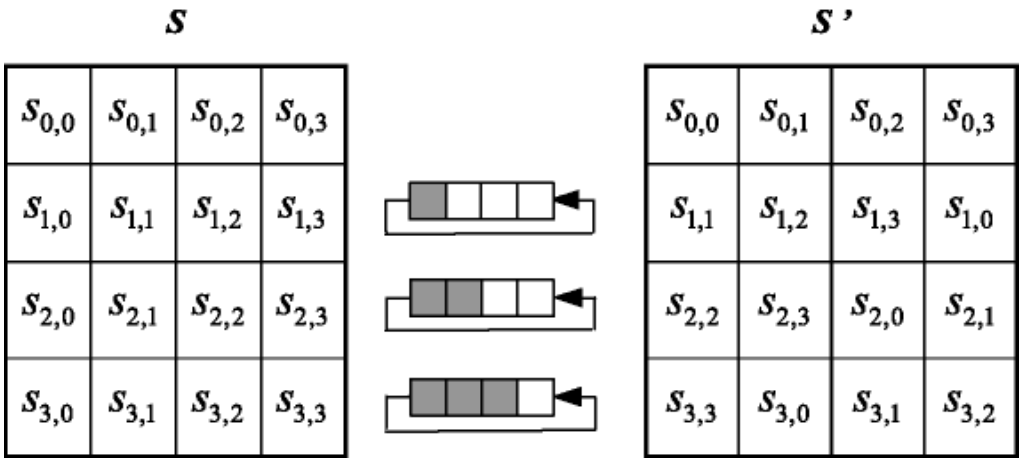
S-box: substitution values for the byte xy (in hexadecimal format).

AES: Verschlüsselung

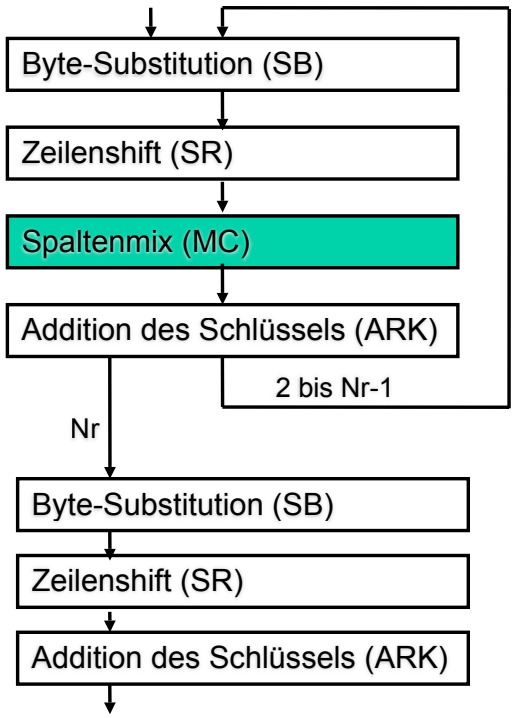


AES Zeilenshift (`ShiftRows()`)

- Zyklischer Shift der letzten drei Zeilen des State:Z
  - Zeile 1 bleibt unverändert
  - Zeile 2 um 1 Byte
  - Zeile 3 um 2 Byte
  - Zeile 4 um 3 Byte

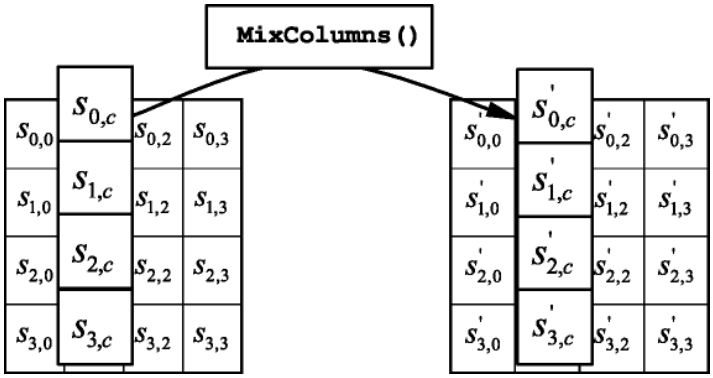


AES: Verschlüsselung



AES Spaltenmix (`MixColumns()`)

- Angewendet auf jede Spalte des State



- Jede Spalte wird als Polynom vom Grad 3 mit Koeffizienten aus GF(2^8) aufgefasst:
  - Multiplikation mit dem festen Polynom a(x) modulo x^4+1

$$a(x) = \{03\}x^3 + \{01\}x^2 + \{01\}x + \{02\} .$$

AES Spaltenmix

- Darstellbar als Matrizenmultiplikation:

$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix} \quad \text{für } 0 \le c < Nb.$$

Ausmultipliziert:

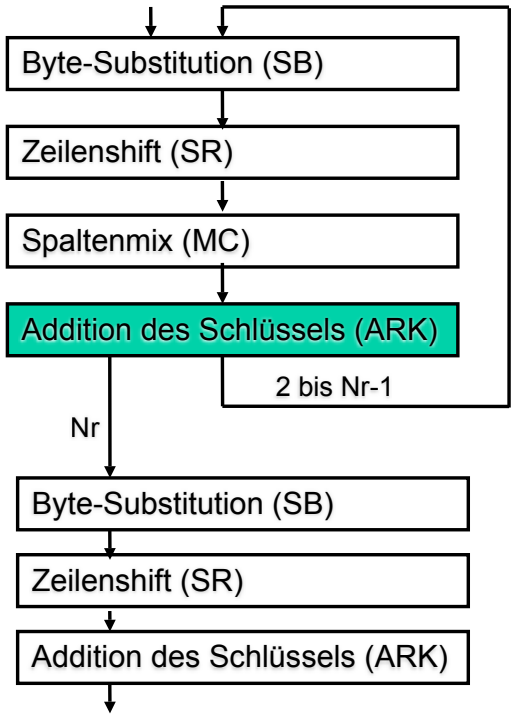
$$s'_{0,c} = (\{02\} \bullet s_{0,c}) \oplus (\{03\} \bullet s_{1,c}) \oplus s_{2,c} \oplus s_{3,c}$$

$$s'_{1,c} = s_{0,c} \oplus (\{02\} \bullet s_{1,c}) \oplus (\{03\} \bullet s_{2,c}) \oplus s_{3,c}$$

$$s'_{2,c} = s_{0,c} \oplus s_{1,c} \oplus (\{02\} \bullet s_{2,c}) \oplus (\{03\} \bullet s_{3,c})$$

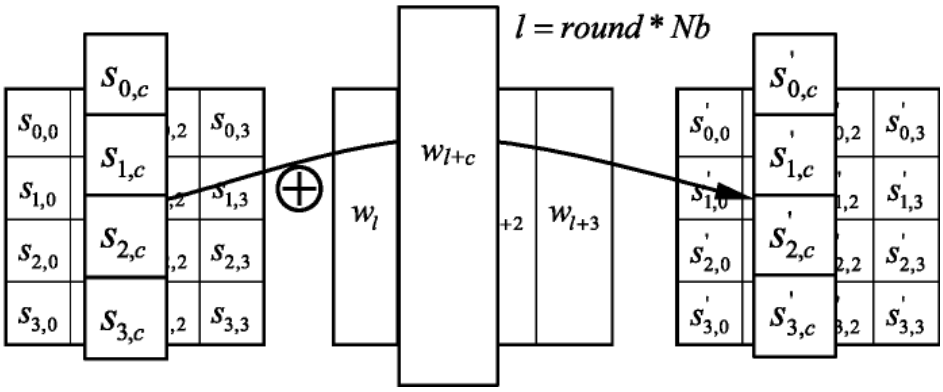
$$s'_{3,c} = (\{03\} \bullet s_{0,c}) \oplus s_{1,c} \oplus s_{2,c} \oplus (\{02\} \bullet s_{3,c}).$$

AES: Verschlüsselung



AES: Addition des Rundenschlüssels

- Funktion `AddRoundKey()`
- Jede Spalte des State wird mit einem „Wort“ des Rundenschlüssels XOR verknüpft

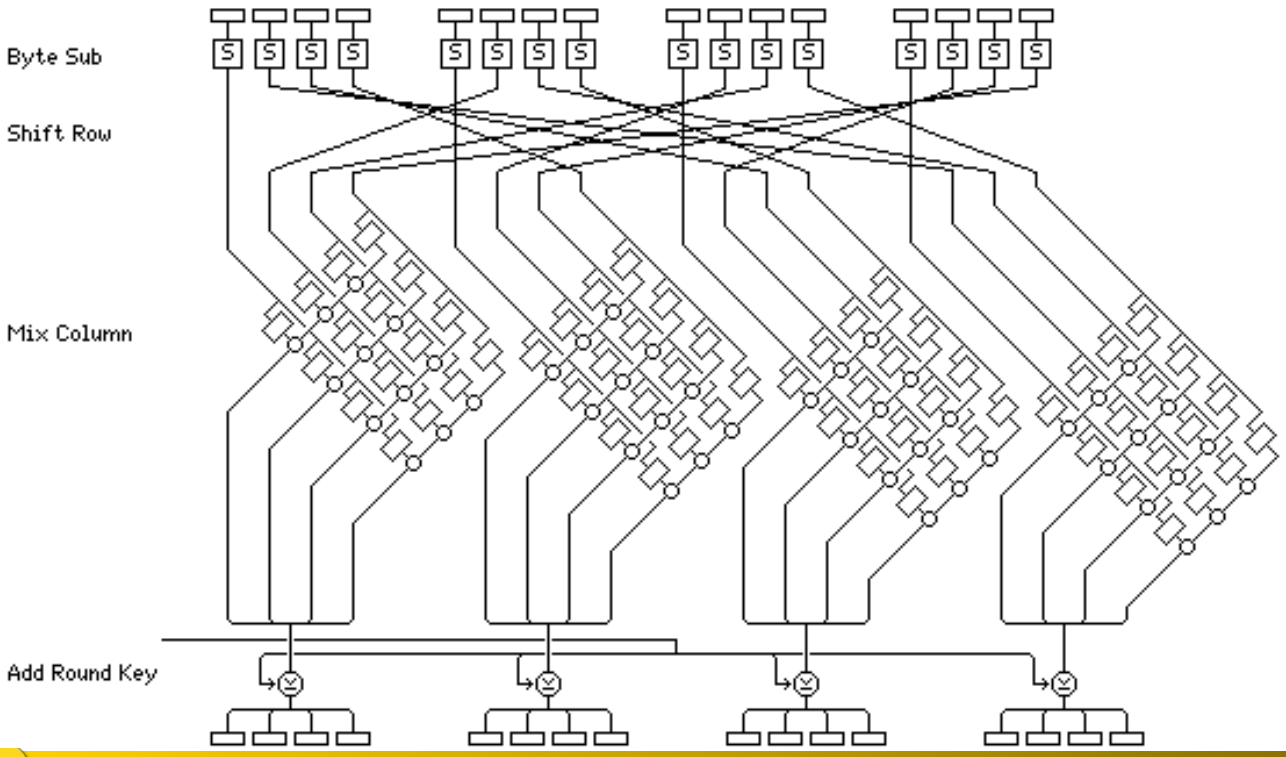


Schlüsselauswahl

- Schlüssel k besteht aus 32 * Nk Bits bzw. 4 * Nk Bytes
- Ein Wort W[i] besteht aus aus 4 Bytes
- W[0] sind die ersten 4 Byte des Schlüssels, W[1] die zweiten 4 Bytes, ..., W[Nk-1] die letzten 4 Bytes
- Insgesamt müssen Nb * (Nr + 1) Wörter berechnet werden
- Im folgenden sei Nk <= 6:
  für i = Nk bis Nb * (Nr +1) gilt:
  - W[i] = W[i-Nk] XOR `SubByte`(W[i-1])

  falls i mod Nk == 0 gilt
  - W[i] = W[i-Nk] XOR `SubByte`(`RotByte`(W[i-1]) XOR `Rcon`[i/Nk])
    mit `SubByte` = AES Byte-Substitution angew. auf Bytes eines Wortes
    mit `RotByte` = zyklischer Shift um 1 Byte (abcd wird bcda)
  - `Rcon[i]=(RC[i], 00, 00, 00)` (Rundenkonstante) mit
    RC[1] = {01}; RC[i] = {02} * RC[i-1]

AES Verschlüsselung

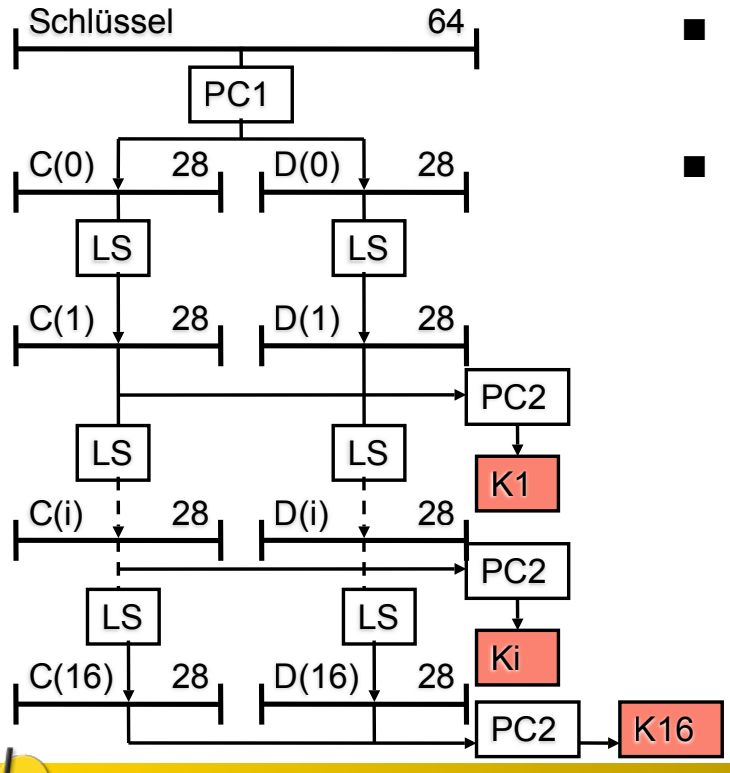


AES Design-Kriterien

- Design-Kriterien mussten offen gelegt werden
- Abschätzung und Stellungnahme zur Widerstandsfähigkeit gegen bekannte Angriffe

- Schlüsselauswahl mit nichtlinearer Durchmischung:
  wegen Verwendung der S-Box;
  damit widerstandsfähig gegen folgende Angriffe:
  - Kryptanalyst kennt Teile des Schlüssels und versucht den Rest zu berechnen
  - Zwei ähnliche Schlüssel haben **keine** große Zahl von gemeinsamen Rundenschlüsseln
  - Rundenkonstante verhindert Symmetrien im Verschlüsselungsprozess; jede Runde ist anders

Wdhlg.: DES Schlüsselauswahl



- Lediglich Permutation und Shift-Operationen

- Keine Substitution bzw. Anwendung von S-Boxen wie bei AES

AES Design-Kriterien (Forts.)

- Keine Feistel Chiffre, d.h. deutlich höhere Diffusion:
  nach 2 Runden hängt jedes Output Bit von jedem Input Bit ab
- Algebraische S-Box Konstruktion; Offengelegt; In hohem Maße nichtlinear
- Damit stabil gegen lineare und differentielle Kryptanalyse
- `ShiftRow` wurde eingefügt um zwei neue Angriffsarten zu verhindern (truncated differentials und Square attack)
- `MixColumn` für hohe Diffusion; Änderung in einem Input Byte verursacht Änderung in allen Output Bytes
- Auswahl von 10 Runden:
  Bei AES mit bis zu 7 Runden sind Angriffe bekannt die besser sind als Brute Force. Bei mehr als 7 Runden sind keine solchen Angriffe bekannt. D.h. 3 Runden „Reserve“ die sehr leicht erweitert werden können

Inhalt

Wdh.: Kryptographisches System (Def.)

- Geg. zwei endliche Zeichenvorräte (Alphabete) A_1 und A_2
- Ein Kryptosystem (KS) ist gegeben durch ein Tupel

$$KS = (M, C, EK, DK, E, D)$$

1. Nicht leere endliche Menge von Klartexten $M \subseteq A_1^*$
2. nicht leere endliche Menge von Krypto- bzw. Chiffretexten $C \subseteq A_2^*$
3. der nicht leeren Menge von Verschlüsselungsschlüsseln EK
4. der nicht leeren Menge von Entschlüsselungsschlüsseln DK sowie einer Bijektion $f : EK \rightarrow DK$ mit $f(K_E) = K_D$
5. Injektives Verschlüsselungsverfahren $E : M \times EK \rightarrow C$
6. Entschlüsselungsverfahren $D : C \times DK \rightarrow M$ mit $\forall m \in M : D(E(m, K_e), K_d)) = m$

Asymmetrisches Kryptosystem, Def.

- $KS = (M, C, EK, DK, E, D)$
- Schlüsselpaare müssen leicht (effizient) zu erzeugen sein, und es muss gelten:
  - $K_E = f(K_D), mit\ K_E \in EK,\ K_D \in DK$
  - $\forall m \in M: \quad D(E(m, K_E), K_D) = m$
  - K_E kann öffentlich bekannt gemacht werden
- Ver- und Entschlüsselungsfunktion E und D effizient zu berechnen
- K_D aus K_E nicht mit vertretbarem Aufwand berechenbar; Umkehrfunktion von f schwer zu berechnen; (Stichwort: **Einwegfunktionen**)
- Kryptosystem zur digitalen Signatur geeignet falls zus. gilt:
  - $\forall m \in M: \quad E(D(m, K_D), K_E) = D(E(m, K_E), K_D) = m$

RSA

- Benannt nach den Erfindern: Rivest, Shamir, Adleman (1978)
- Sicherheit basiert auf dem Faktorisierungsproblem:
  - Geg. zwei große Primzahlen p und q (z.B. 200 Dezimalstellen):
  - n=pq ist auch für große Zahlen einfach zu berechnen,
  - aber für gegebenes n ist dessen Primfaktorzerlegung sehr schwierig
- Erfüllt alle Anforderungen an asymmetrisches Kryptosystem
- In USA patentiert (Patent ist 2000 ausgelaufen)
- Große Verbreitung, verwendet in:
  - SSL (Secure Socket Layer)
  - PEM (Privacy Enhanced Mail)
  - PGP (Pretty Good Privacy)
  - GNUpg
  - ....

RSA Mathematische Grundlagen

- Restklassenarithmetik
- Restklassenring modulo m
- Multiplikatives Inverses
- Eulersche φ Funktion
- Einweg-Funktion (One way function)

Vgl. Tafel

- Einweg-Funktion mit Falltür (Trapdoor one way function)

RSA Funktionsweise

1. Wähle große Primzahlen p und q und berechne $n = pq$
   n wird als RSA-Modul bezeichnet
2. Wähle $d \in [0, n-1]$ so dass d relativ prim ist zu $\varphi(n)$
   - D.h. $ggT(\varphi(n), d) = 1$
   - $\varphi(n) = (p-1)(q-1)$

   Wähle d so dass gilt: $\max(p,q) < d < \varphi(n) - 1$
3. Wähle $e \in [0, n-1]$ mit $e \times d \equiv 1 \bmod \varphi(n)$
   d.h. e ist das multiplikative Inverse modulo $\varphi(n)$ zu d
4. Öffentlicher Schlüssel (e, n)
5. Geheimer Schlüssel (d, n)
6. Verschlüsseln eines Klartextes $m \in [0, n-1]$

$$E(m) = m^e \bmod n = c$$

7. Entschlüsseln:

$$D(c) = c^d \bmod n$$

Erfüllt RSA Anforderungen an asym. Kryptosystem?

- Es sind zwei Sachverhalte zu zeigen:

1. Der Chiffrentext kann auch wieder entschlüsselt werden:

$$D(E(m)) = m$$

2. Das Verfahren eignet sich zur digitalen Signatur

$$D(E(m)) = E(D(m))$$

- Beweis vgl. Tafel

Einschub: Nomenklatur für kryptologische Verfahren

- Für Verschlüsselungsverfahren wird künftig die folgende Notation verwendet:

A_p	Öffentlicher (public) Schlüssel von A
A_s	Geheimer (secret) Schlüssel von A
A_p{m}	Verschlüsselung der Nachricht m mit dem öffentlichen Schlüssel von A
A_s{m} oder A{m}	Von A erstellte digitale Signatur von m
S[m]	Verschlüsselung von m mit dem **symmetrischen** Schlüssel S

RSA Sicherheit / mögliche Angriffe

1. **Brute force**: Testen aller möglichen Schlüssel
2. **Chosen-Cipertext Angriff**: (vgl. Tafel)
3. **Mathematische Angriffe**:
   - Faktorisierung von n;
   - direkte Bestimmung von $\varphi(n)$ ohne Faktorierung;
   - direkte Bestimmung von d ohne Bestimmung von $\varphi(n)$
4. **Timing Angriff:** Info: [Koch 96, Kali 96]
   - Überwachung der Laufzeit von Entschlüsselungsoperationen
   - Über Laufzeitunterschiede kann privater Schlüssel ermittelt werden
   - Analogie: Einbrecher ermittelt Kombination eines Tresors mit Hilfe der mitgehörten Stellzeiten am Zahlenschloß
5. **Angriffe auf Signaturen** (vgl. spätere Folien zur dig. Signatur)
   - Existentielle Fälschung
   - Multiplikativität von RSA

RSA: Chosen-Ciphertext Angriff

Vgl. Tafel

RSA Mathematische Angriffe

- Mathematische Angriffe lassen sich auf Faktorisierung zurückführen
- Schnellster bekannter Algorithmus General Number Field Sieve (GNFS), vgl. [Silv 01]
  - Laufzeitkomplexität: $L(N) = e^{(c+o(1))\bullet\sqrt[3]{\log(N)}\bullet\sqrt[3]{\log(\log(N))^2}}$
  - Speicherplatzkomplexität: $\sqrt{L(N)}$
- Faktorisierung wird einfacher falls:
  - Anzahl der Ziffern von p und q große Unterschiede aufweisen (z.B. |p| = 10 und |q| = 120)
  - Falls $d < 1/3 \bullet \sqrt[4]{n}$ kann d leicht berechnet werden
  - Die ersten m/4 Ziffern oder die letzten m/4 Ziffern von p oder q sind bekannt

RSA Schlüssellängen

- RSA Challange: Belohnung für das brechen von RSA Schlüsseln, z.B. durch Faktorisierung (2007 eingestellt)

Dezimalstellen	Bits	Datum	Aufwand	Algorithmus
100	332	April 1991	7 Mips Jahre	Quadratisches Sieb
110	365	April 1992	75 Mips J.	
120	398	Juni 1993	830 Mips J.	
129	428	April 1994	5000 Mips J.	
130	431	April 1996	1000 Mips J.	General Number Field Sieve (GNFS)
140	465	Februar 1999	2000 Mips J.	
155	512	August 1999	8000 Mips J.	
160	530	April 2003	k.A.	GNFS(Lattice Sieve)
174	576	Dez. 2003	k.A.	GNFS(Lattice/Line Sieve)
193	640	Nov. 2005	30 2,2 GHz Opteron J.*	GNFS

* Halb so lange wie für RSA-200 Challange

RSA Schlüssellängen

- RSA Challange: Belohnung für das brechen von RSA Schlüsseln, z.B. durch Faktorisierung (2007 eingestellt)

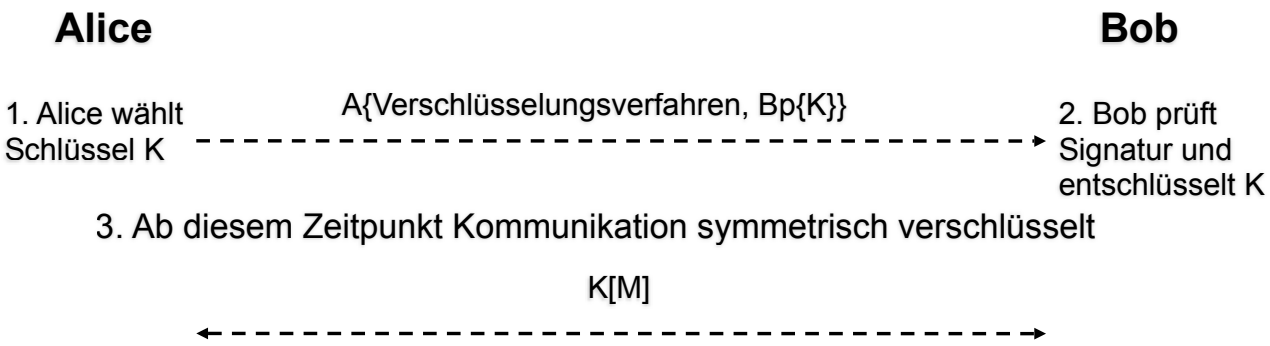
Dezimalstellen	Bits	Datum	Aufwand	Algorithmus
100	332	April 1991	7 Mips Jahre	Quadratisches Sieb
110	365	April 1992	75 Mips J.	
155	512	August 1999	8000 Mips J.	
160	530	April 2003	k.A.	GNFS(Lattice Sieve)
174	576	Dez. 2003	k.A.	GNFS(Lattice/Line Sieve)
193	640	Nov. 2005	30 2,2 GHz Opteron J.*	GNFS

d.h. 512 Bit Schlüssel ist nicht mehr sicher!

* Halb so lange wie für RSA-200 Challange

Hybride Kryptosysteme

- Vereinen Vorteile von Symmetrischen und asymmetrischen Verfahren
- Asymmetrisches Verfahren zum Schlüsselaustausch
- Symmetrisches Verfahren zur Kommunikationsverschlüsselung



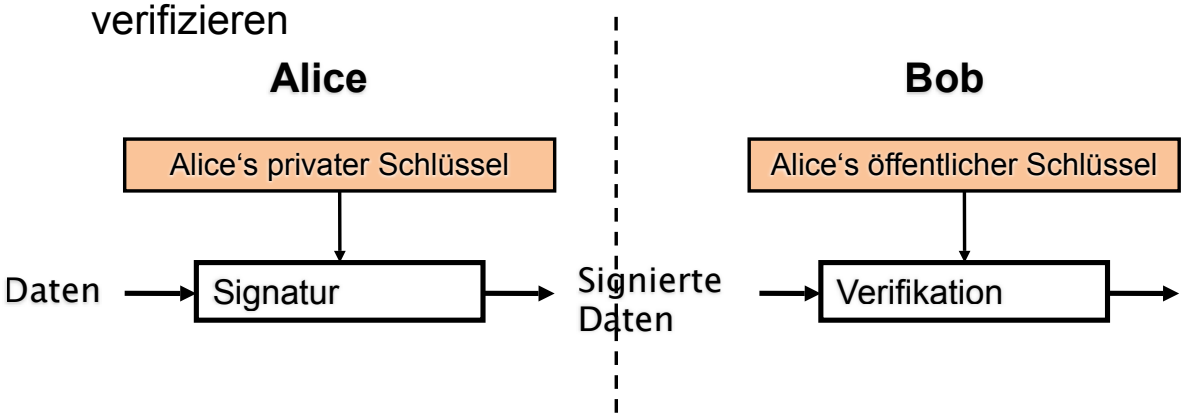
- Beispiele für hybride Verfahren: PGP, ssh,...

Einschub: US-CERT Alert TA07-345A

- Microsoft Updates for Multiple Vulnerabilities
- Systems affected:
  - Microsoft Windows
  - Microsoft Internet Explorer
- Description
  - DirectX vulnerability (specially crafted media files)
  - Vulnerability in Windows Media File Format
  - Cummulative security update for Internet Explorer (Uninitialized Memory Corruption Vulnerability)
- Impact:
  - Remote Code Execution; DoS
- Solution:
  - Apply updates from Microsoft

Digitale Signatur

- Alice „signiert“ Daten mit ihrem privaten Schlüssel
- Jeder kann die Signatur mit Alice‘ öffentlichem Schlüssel verifizieren



- Assymetrische Verfahren sind im Vergleich sehr langsam
- Daher i.d.R. nicht Signatur der gesamten Daten
- Lediglich kryptographischer Hash-Wert der Daten wird signiert (digitaler Fingerabdruck der Daten)

Digitale Signatur: Analogie zur Unterschrift

- Anforderungen an die (analoge) Unterschrift:

1. **Perpetuierungsfunktion:** Unterschrift kann nicht gefälscht werden und ist dauerhaft
2. **Echtheitsfunktion:** Die Unterschrift ist authentisch
3. Die Unterschrift kann **nicht wiederverwendet** werden.
4. **Abschlußfunktion:** Unterschrift kann später nicht verändert werden
5. **Beweisfunktion:** Unterzeichner kann seine Unterschrift später nicht leugnen

- Bei der Unterschrift auf Papier ist keine dieser Anforderungen vollständig erfüllt!
- Trotzdem wird die Unterschrift im Rechtsverkehr akzeptiert
- Ihre Funktion wird durch Rahmenbedingungen gesichert

Digitale Signatur: Erfüllung der Anforderungen?

1. **Perpetuierungsfunktion:** Fälschungssicher und dauerhaft
2. **Echtheitsfunktion:** authentisch
3. **Nicht wiederverwendbar**
4. **Abschlußfunktion:** nicht veränderbar
5. **Beweisfunktion:** Unterschrift nicht leugnen

- Lassen sich diese Funktionen bei der digitalen Signatur realisieren?

1. Solange privater Schlüssel geheim gehalten wird
2. Abhängig von zweifelsfreier Zuordnung des Schlüsselpaares zu einer Identität (Zertifizierung, CA)
3. Digitale Signatur „beinhaltet“ den Dateninhalt
4. vgl. 3.
5. Jeder kann Signatur bzw. Echtheit mit öffentlichem Schlüssel des Unterzeichners verifizieren

RSA Signaturangriffe

- Existentielle Fälschung:
  - Mallet wählt beliebige Zahl r mit 0 <= r < n
  - M behauptet r sei ein von Alice signiertes Dokument (z.B. abzuhebender Geldbetrag)
  - Empfänger verifiziert und falls m=A_p{r} sinnvollen Text ergibt, war der Angriff erfolgreich
- Multiplikativität von RSA
  - Mallet wählt zwei Nachrichten m1 und m2 und lässt Alice signieren
  - Mallet berechnet A{m1}*A{m2} mod n
  - Es gilt $A\{m1\} \bullet A\{m2\} \bmod n = (m1 \bullet m2)^{As} \bmod n$
  - D.h. M kann aus zwei signierten Dokumenten ein drittes, gültig signiertes Dokument erzeugen, ohne im Besitz des Schlüssels von Alice zu sein
- Gegenmaßnahme:
  Nicht das gesamte Dokument, sondern nur Hash-Wert signieren!

Inhalt

1. Kryptologie: Begriffe, Klassifikation
2. Steganographie
3. Kryptographie
   1. Begriffe und Definitionen
      - Kryptosystem
      - Substitution, Permutation
      - Symmetrische / asymmetrische Kryptosysteme
      - Kryptoanalyse
   2. Symmetrische Kryptosysteme
      - Data Encryption Standard (DES)
      - Advanced Encryption Standard (AES)
   3. Asymetrische Kryptosysteme
      - RSA
      - Digitale Signatur
   4. Hybride Kryptosysteme
   5. Kryptographische Hash-Verfahren

Kryptographische Hash-Funktionen: Grundlagen

- Hash-Funktionen bilden „Universum“ auf endlichen Bildbereich ab
- Hash-Funktion h 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-Funktionen H:
  - Eingabe: beliebig langes Wort m aus dem Universum U
  - Ausgabe: Hash-Wert H(m) mit fester Länge
  - H soll möglichst kollisionsresistent sein

Einsatz kryptographischer Hash-Funktionen

- Integritätssicherung („Digitaler Fingerabdruck“):
  1. Alice erzeugt Nachricht m, berechnet H(m)=h und überträgt (m,h) an Bob (mindestens h muss gesichert werden, z.B. durch Verschlüsselung)
  2. Bob empfängt (m‘,h) und berechnet h‘=H(m‘)
  3. Falls h=h‘ kann davon ausgegangen werden, dass m=m‘, d.h. m wurde **nicht** verändert

- Digitale Signatur:
  - In der Praxis wird nicht die Nachricht m digital signiert
  - Stattdessen wird H(m) digital signiert {H(m)}
  - Übertragen wird dann (m,{H(m)})
  - Empfänger kann Quelle der Nachricht zweifelsfrei feststellen
  - Empfänger kann Integrität der Nachricht belegen

Def. Kryptographische Hashfunktion

- Schwache Hash-Funktion H:
  1. H besitzt die Eigenschaften einer Einwegfunktion
  2. Hash-Wert H(m) = h mit |h|=k ist bei gegebenem m einfach zu berechnen
  3. Bei gegebenem h = H(m) für $m \in A_1^*$ ist es praktisch unmöglich ein m‘ zu finden mit:

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

- Starke Hash-Funktion H:
  1. H ist eine schwache Hash-Funktion
  2. Es ist praktisch unmöglich eine Kollision zu finden, d.h. ein Paar verschiedene Eingabewerte m und m‘ mit:

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

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
- Wie viele Personen brauchen Sie, damit mit Wahrscheinlichkeit p > 0,5 zwei Personen am selben Tag Geburtstag haben
  - Antwort: 23
- 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 m-Bit langen Hash eine Nachricht M.
2. Mallet erzeugt 2^(m/2) Variationen der Nachricht M

- Die Wahrscheinlichkeit für eine Kollision ist größer 0,5.

- Wie können 2^(m/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

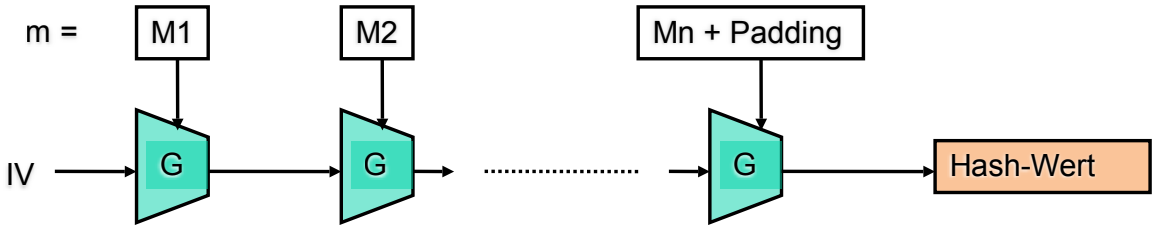
- [Stal 98]

Dear Anthony,

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

Konstruktion kryptographischer Hash Funktionen

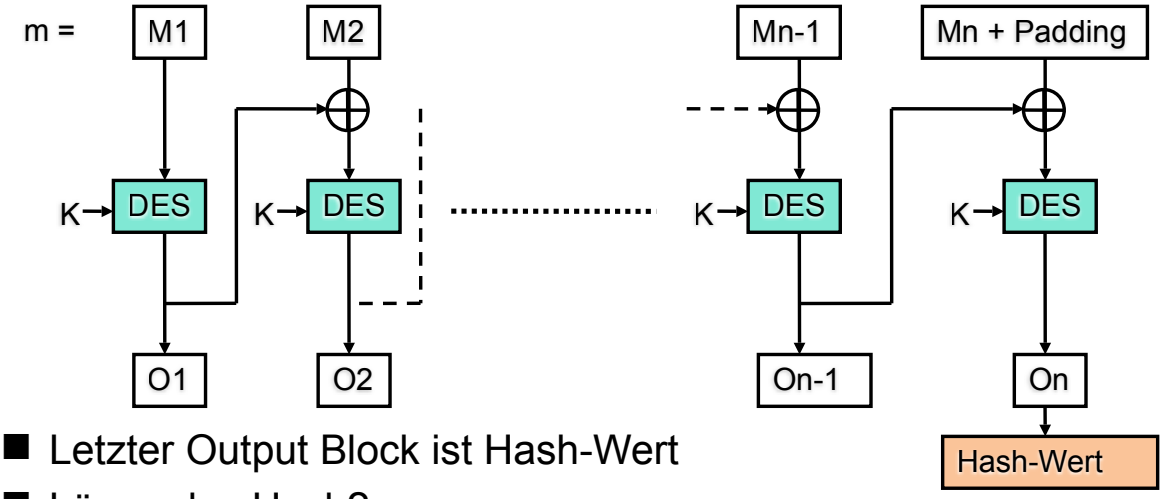
- Folge von Kompressionsfunktionen G
- Nachricht m wird in Blöcke Mi mit fester Länge zerlegt
- Hash Verfahren wird mit Initialisierungswert IV vorbelegt



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

DES als Kompressionsfunktion

- DES im Cipher Block Chaining (CBC) Mode



- Letzter Output Block ist Hash-Wert
- Länge des Hash?

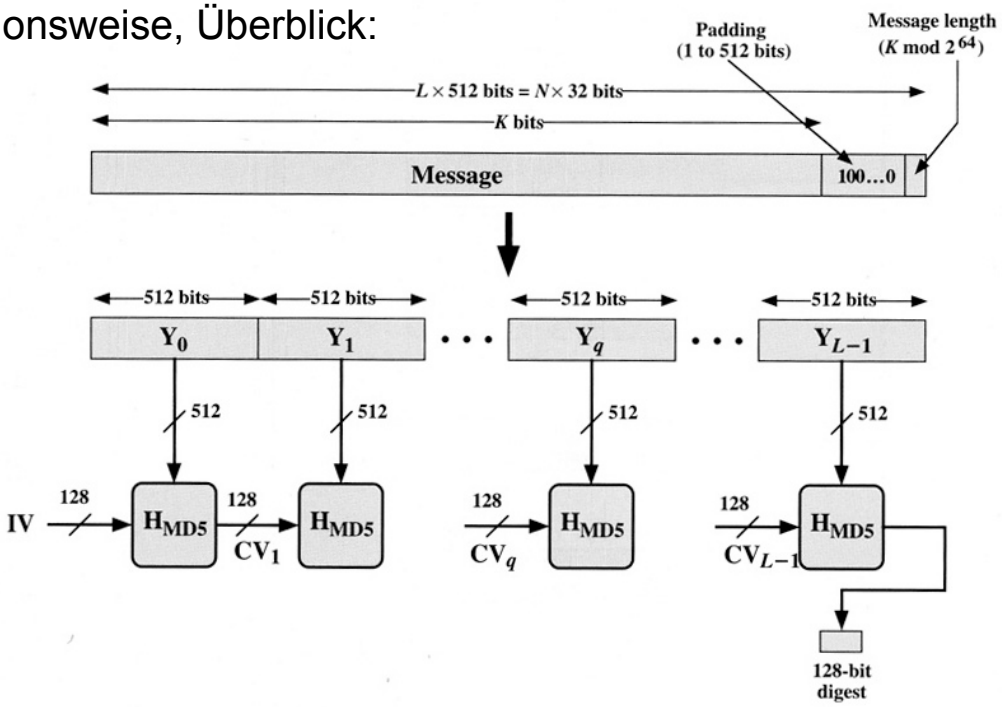
  64 Bit

Hash-Funktionen: MD4

- 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
  - 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
- Trotzdem: Rivest verbessert MD4; Ergebnis MD5

Hash-Funktionen: MD5

- Länge 128 Bit, arbeitet auf 512 Bit Blöcken
- Funktionsweise, Überblick:

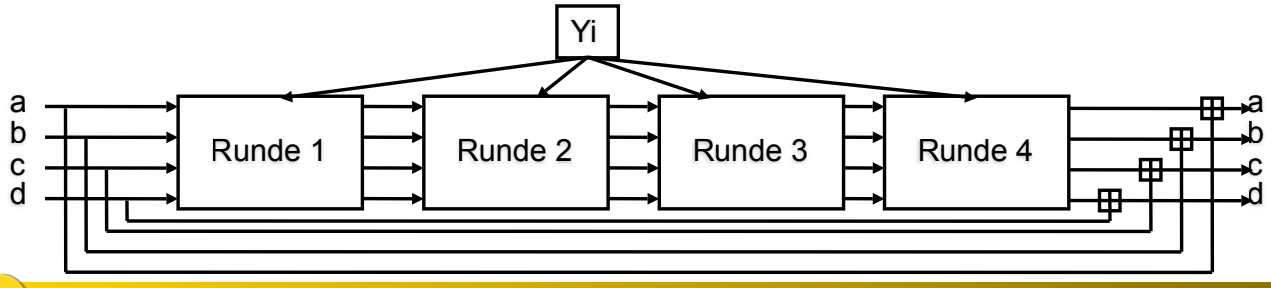


MD5 Ablauf

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 Mj von Yi wie folgt angewendet

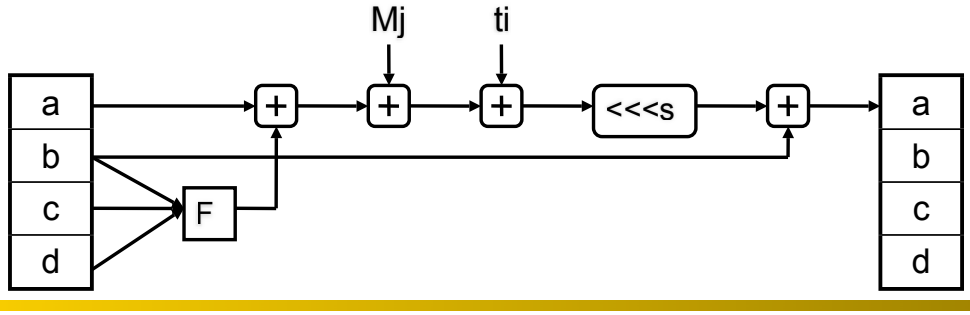
$$FF(a,b,c,d,M_j,s,t_i): \quad a = b + ((a + F(b,c,d) + M_j + t_i) <<< s)$$
$$GG(a,b,c,d,M_j,s,t_i): \quad a = b + ((a + G(b,c,d) + M_j + t_i) <<< s)$$
$$HH(a,b,c,d,M_j,s,t_i): \quad a = b + ((a + H(b,c,d) + M_j + t_i) <<< s)$$
$$II(a,b,c,d,M_j,s,t_i): \quad a = b + ((a + I(b,c,d) + M_j + t_i) <<< s)$$

MD5 Kompressionsfunktion (2)

- $FF(a,b,c,d,M_j,s,t_i): \quad a = b + ((a + F(b,c,d) + M_j + t_i) <<< s)$
  - + bezeichnet Addition modulo 2^32
  - $t_i = 2^{32} abs(\sin(i))$ mit i Grad im Bogenmaß; 0 <= i < 64 (i über 4 Runden)
  - <<< s bezeichnet zirkulären Shift um s Bits
  - Auswahl des Teilblocks Mj

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



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
  - Im Moment scheint 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
  - Bisher gefährlichster bekannter Angriff
  - Bisher kein Mechanismus zur Generalisierung des Angriffs auf gesamten MD5 mit IV gefunden

Vergleich der gebräuchlichsten Hash-Funktionen

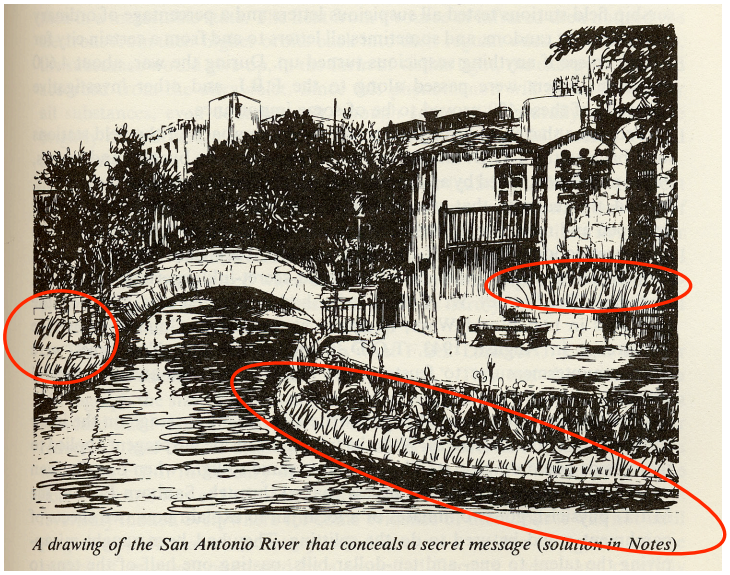
	MD5	SHA-1	RIPEMD-160
Hash-Länge [Bit]	128	160	160
Blocklänge [Bit]	512	512	512
Anzahl der Schritte	64 4 Runden a 16	80 4 Runden a 20	160 5 x 2 Runden a 16
Maximale Nachrichtenlänge [Bits]	Unendlich	2^64 – 1	2^64 - 1
Primitive Logische Funktionen	4	4	5
Performance Mbps*	32,4	14,4	13,6

- Prinzipiell gilt: Je länger der Hash-Wert umso sicherer das Verfahren

* Auf einem 266 MHz Pentium [Stal 98]

Linguistische Steganographie

- Bsp. aus David Kahn: *The Codebreakers,* Scribner, 1996



A drawing of the San Antonio River that conceals a secret message (solution in Notes)

- Nachricht als Morsezeichen kodiert.
- Kurze und lange Grashalme
- Nachricht lautet:

  Compliments of CPSA MA to our chief Col Harold R. Shaw on his visit to San Antonio May 11th 1945