TECHNISCHE UNIVERSITÄT DRESDEN

DRESDEN concept

Betriebssysteme und ***Sicherheit***

Stefan Köpsell, Thorsten Strufe

Modul 3: Mechanismen - Vertraulichkeit

Disclaimer: Inhalte übernommen aus Materialien von Dan Boneh, Winfried Kühnhäuser, Günter Schäfer, Andreas Pfitzmann, Elke Franz, Mitarbeitern des Lehrstuhls

Dresden, WS 14/15

The story so far…

Sicherheit als abstrakte Anforderung

Safety vs. Security

Datenschutz und Aspekte des Datenschutzes

Formale Ziele der IT Sicherheit

Bedrohungen – abstrakt und technisch

Bedrohungsanalysen

Angreifermodelle und Angriffstechniken

Risiko-Analysen

Was kann man tun: Sicherheitsmechanismen

Vertraulichkeit

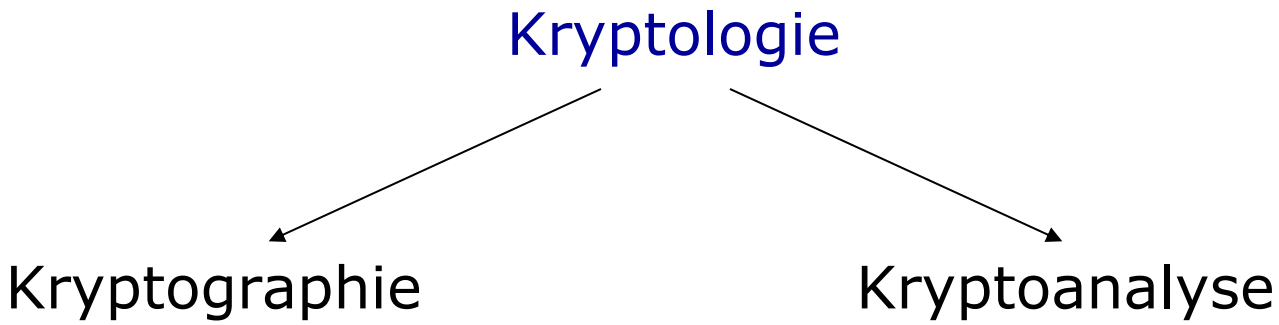
Schlüssel-Management

Identität und Authentisierung

Integrität

Zugriffskontrolle

Terminologie



Kryptographie (griech. „kryptos“+ „graphein“)

Wissenschaft von den Methoden der Ver- und Entschlüsselung von Informationen.

Kryptoanalyse (griech. „kryptos“+ „analyein“)

Wissenschaft vom Entschlüsseln von Nachrichten ohne Kenntnis dazu notwendiger geheimer Informationen.

Kryptologische Grundlagen

erreichbare Schutzziele:

Vertraulichkeit, Konzelation genannt

➔ Verschlüsselungs- / Konzelationssystem

Integrität (= keine *unerkannte* unbefugte Modifikation von Informationen), Authentikation genannt

➔ Authentikationssystem, digitale Signatur

durch Kryptographie unerreichbar:

Verfügbarkeit – zumindest nicht gegen starke Angreifer

Classifying Encryption Algorithms

Type of operation

- Substitution
- Transposition

Number of keys

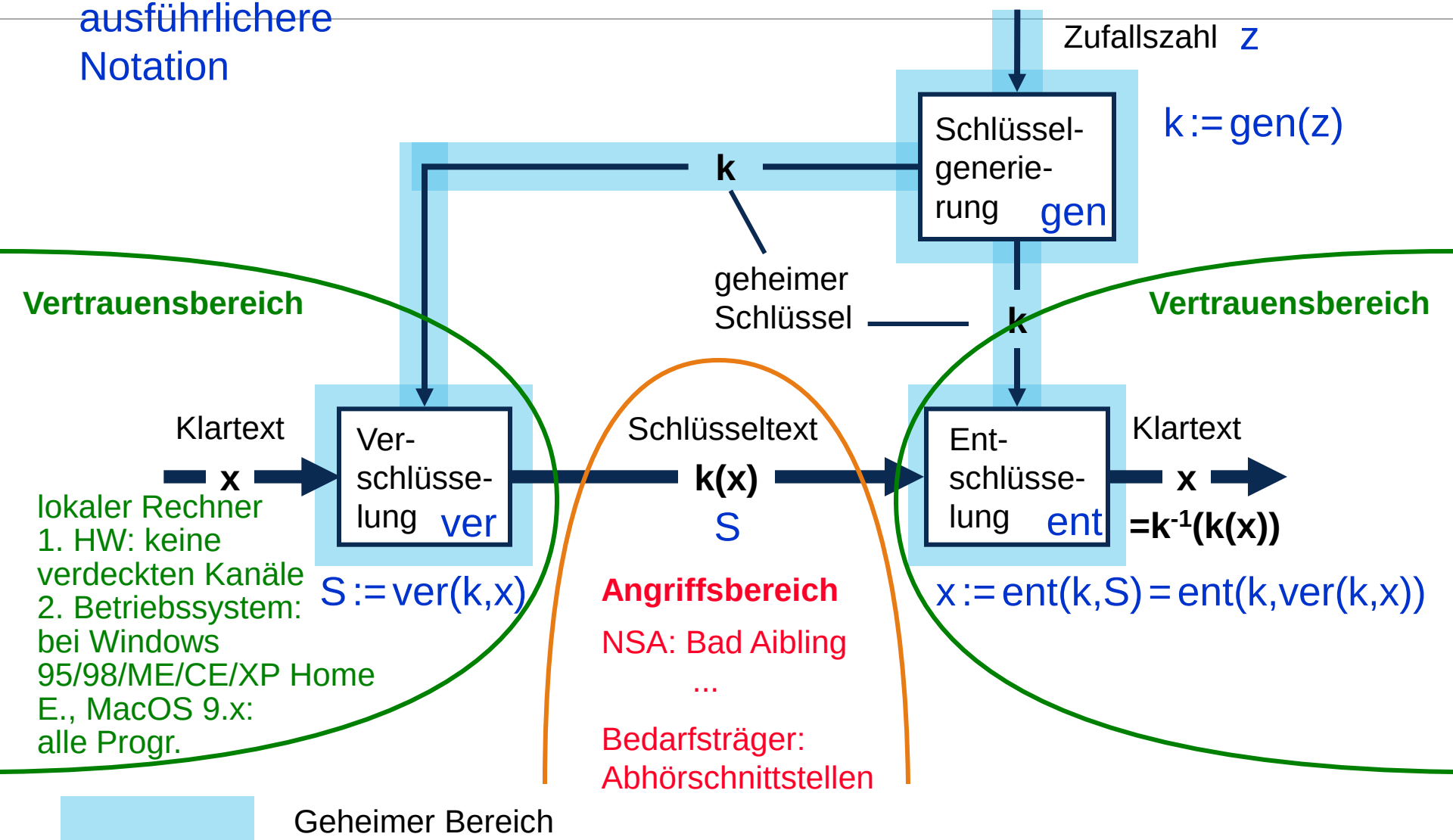
- Symmetric: secret key
- Asymmetric: „public key“, pair of public and private key

Processing of plaintext

- Stream ciphers: operate on streams of bits
- Block ciphers: operate on b-bit blocks

Symmetrisches Konzelationssystem

ausführlichere Notation



Undurchsichtiger Kasten mit Schloß; 2 gleiche Schlüssel

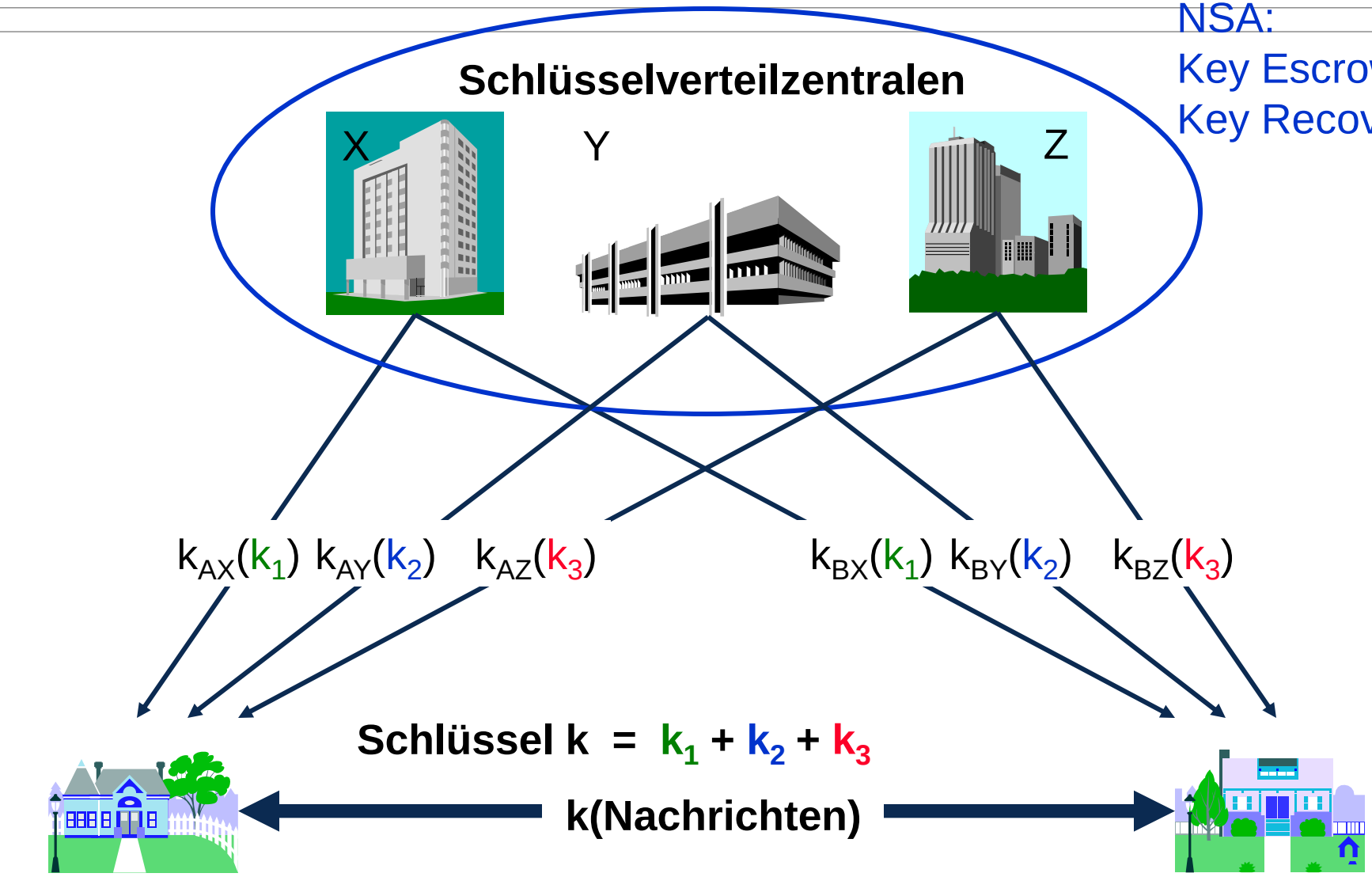
Kerckhoff's Prinzip (1883 von Auguste Kerckhoffs: "La cryptographie militaire")

"Sicherheit des Schutzmechanismus darf **ausschließlich** von der Geheimhaltung kryptographischer Schlüssel abhängen – nicht jedoch von der Geheimhaltung des Algorithmus"

- Daher:
  - nur veröffentlichte und wohl untersuchte Algorithmen und Protokolle verwenden
  - Algorithmen nicht selbst entwerfen und auf gar keinen Fall geheime, "hoch sichere" Algorithmen verwenden
  - Protokollentwurf ist sehr schwierig, da sehr komplex: durch die Kombination sicherer Baustein muß nicht notwendigerweise ein sicheres Gesamtsystem entstehen!

Schlüsselverteilung bei symmetrischem Kryptosystem

NSA:
Key Escrow
Key Recovery



Historische Verfahren

- Transpositionen
  Verwürfeln der Klartextzeichen, Permutation der Stellen des Klartextes (Permutationschiffren)

  Beispiel: Skytala (Matrixtransposition)

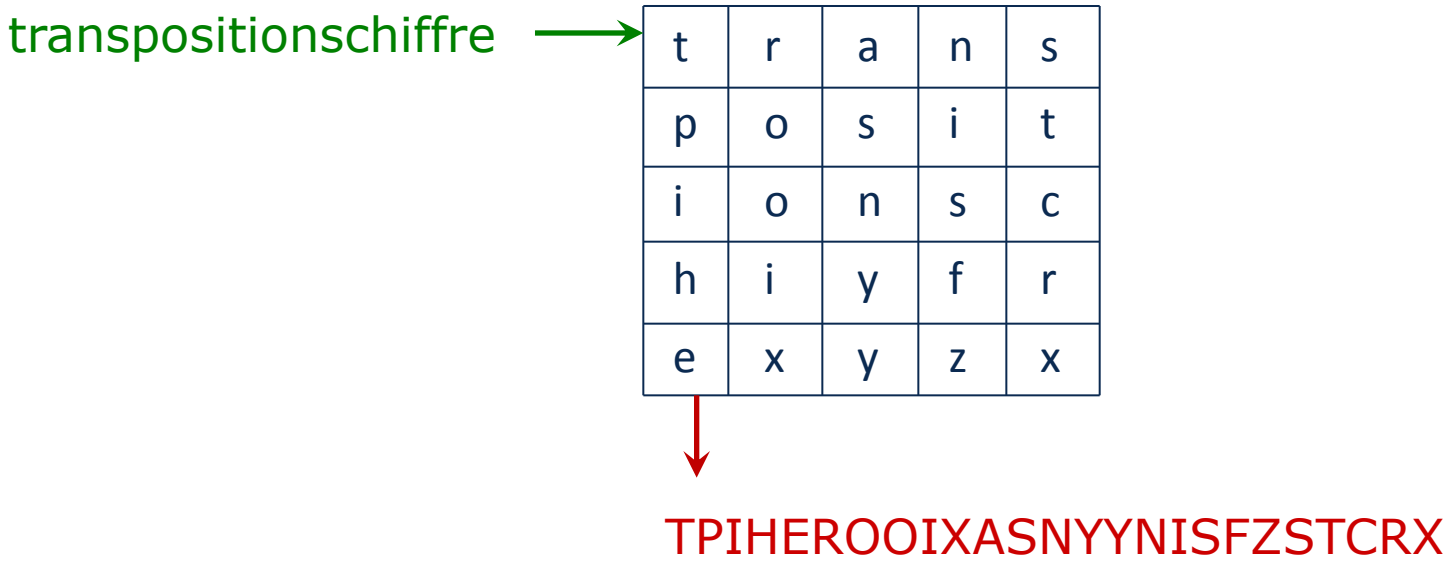
Skytale

[„Skytale“. Lizenziert unter Creative Commons Attribution-Share Alike 3.0 über Wikimedia Commons – http://commons.wikimedia.org/wiki/File:Skytale.png#mediaviewer/File:Skytale.png]

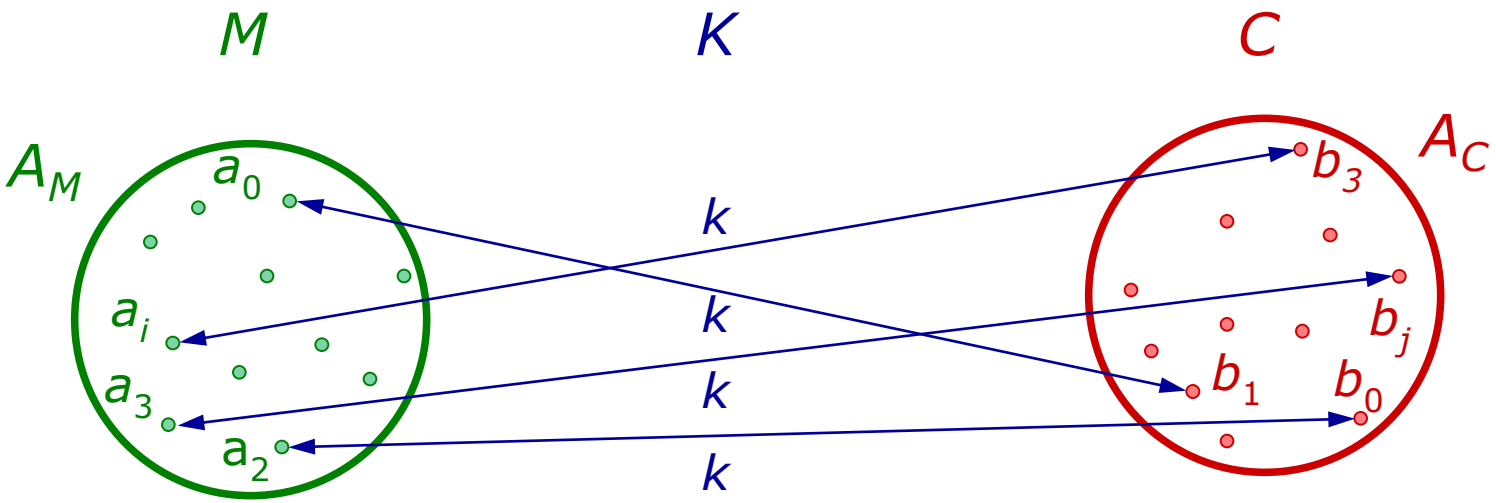
Historische Verfahren

Transpositionen
Verwürfeln der Klartextzeichen, Permutation der Stellen des Klartextes
(Permutationschiffren)

Beispiel: Skytala (Matrixtransposition)



Einfache Substitutionen: MM-Substitution (monoalphabetisch, monographisch)



$$\text{enc}(k, m) = \text{enc}(k, m_0)\,\text{enc}(k, m_1)\,\ldots\,\text{enc}(k, m_{l-1})$$

Einfache Substitutionen

Verschiebechiffre

- Nachricht: Buchstaben A .. Z ($A_M = A_C = \{A:Z\}$)
- Schlüssel s: Verschiebung, $s \in \{0:n\text{-}1\}$, $n = 26$
- Cäsarchiffre für $s = 3$

Nachricht	a	b	c	d	e	f	g		…		x	y	z
Schlüsseltext	D	E	F	G	H	I	J		…		A	B	C

Einfache Substitutionen

Kryptoanalyse der Verschiebechiffre

- Nur 26 verschiedene Schlüssel
- Vollständige Suche möglich
  → Durchprobieren aller möglichen Schlüssel

Beispiel:

$c =$ FMTKOJVIVGTNZDNOYVNOCZHVYZMCZPODBZIQJMGZNPIB

Verschiebechiffre: Vollständige Suche

s	m
0	FMTKOJVIVGTNZ
1	ELSJNIUHUFSMY
2	DKRIMHTGTERLX
3	CJQHLGSFSDQKW
4	BIPGKFRERCPJV
5	AHOFJEQDQBOIU
6	ZGNEIDPCPANHT
7	YFMDHCOBOZMGS
8	XELCGBNANYLFR
9	WDKBFAMZMXKEQ
10	VCJAEZLYLWJDP
11	UBIZDYKXKVICO
12	TAHYCXJWJUHBN
13	SZGXBWIVITGAM
14	RYFWAVHUHSFZL
15	QXEVZUGTGREYK
16	PWDUYTFSFQDXJ
17	OVCTXSEREPCWI
18	NUBSWRDQDOBVH
19	MTARVQCPCNAUG
20	LSZQUPBOBMZTF
21	**KRYPTOANALYSE**
22	JQXOSNZMZKXRD
23	IPWNRMYLYJWQC
24	HOVMQLXKXIVPB
25	GNULPKWJWHUOA

$s = 21$ ergibt den einzigen sinnvollen Text → $s = 21$

Einfache Substitutionen

Statistische Analysen

- Möglichkeiten der vollständigen Suche sind eingeschränkt
- Aufwand für eine allgemeine Substitution: $|A|!$

How large is large? (Some context)

Reference Numbers Comparing Relative Magnitudes			
Reference	***Magnitude***		
Seconds in a year		$\approx 3 * 10^7$	
Seconds since creation of solar system		$\approx 2 * 10^{17}$	$\approx 4.6 * 10^9$ y
Clock cycles per year (50 MHz computer)		$\approx 1.6 * 10^{15}$	
Instructions per year (i7 @ 3.9 GHz)	$\approx 2^{62}$	$\approx 4.3 * 10^{18}$	
Binary strings of length 64	2^{64}	$\approx 1.8 * 10^{19}$	
Binary strings of length 128	2^{128}	$\approx 3.4 * 10^{38}$	
Binary strings of length 256	2^{256}	$\approx 1.2 * 10^{77}$	
Number of 75-digit prime numbers		$\approx 5.2 * 10^{72}$	
Electrons in the universe		$\approx 8.37 * 10^{77}$	

Attacking Cryptography: Brute Force Attack

Brute force attack: try all keys until intelligible plaintext found:

- Cryptographic algorithms can be attacked by brute force
- On average, half of all possible keys will have to be tried

Average Time Required for Exhaustive Key Search

Key Size [bit]	*Number of keys*	*Time required at 1 encryption / μs*	*Time required at 10^6 encryption / μs*
32	$2^{32} = 4.3 * 10^9$	2^{31} μs = 35.8 minutes	2.15 milliseconds
56	$2^{56} = 7.2 * 10^{16}$	2^{55} μs = 1142 years	10.01 hours
128	$2^{128} = 3.4 * 10^{38}$	2^{127} μs = $5.4 * 10^{24}$ years	$5.4 * 10^{18}$ years
PM 26 chars	$26! = 4 * 10^{26}$	2^{88} μs = $6.4*10^{12}$ years	$6.4*10^6$ years

i7 could get to around 10^4 encryptions/μs
GPU can perform around $7x10^5$ hashes

Time since human/chimpanzee lines diverged:
$5×10^6$ years,
Homo sapiens: $5×10^4$ years

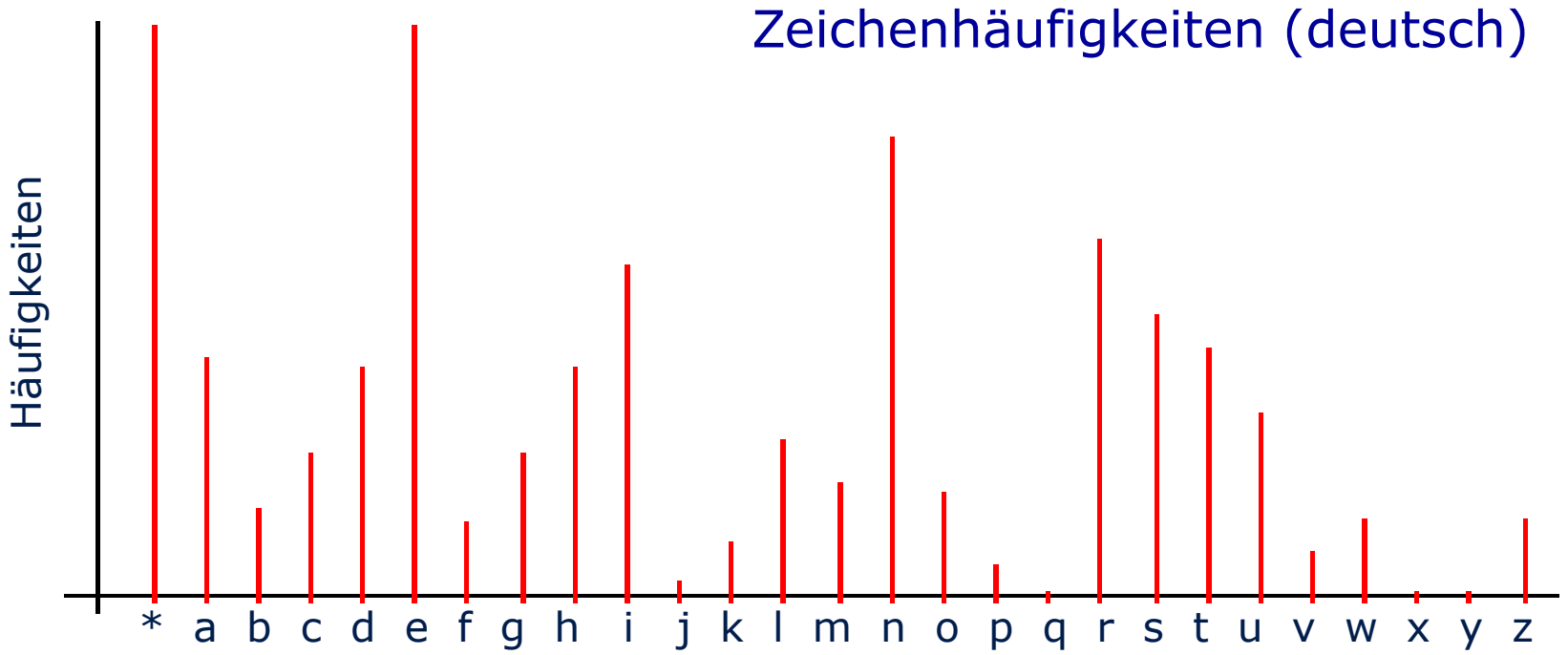
Einfache Substitutionen

Statistische Analysen

- Möglichkeiten der vollständigen Suche sind eingeschränkt
- Aufwand für eine allgemeine Substitution: $|A|!$
- Angriffspunkt: Einfache Substitutionen übertragen statistische Eigenschaften der Klartexte in die Schlüsseltexte

$m =$	B	E	I	S	P	I	E	L
Polybios: $c =$	12	15	24	44	41	24	15	32
Permutation (Bsp. F. 13): $c =$	J	W	X	Z	S	X	W	E
Verschiebechiffre ($s = 3$): $c =$	E	H	L	V	S	L	H	O

Einfache Substitutionen



Zeichenhäufigkeiten (%)

	Deutsch		Englisch	
*	15.15	-	19.25	-
a	4.58	5.40	6.60	8.17
b	1.60	1.89	1.21	1.49
c	2.67	3.15	2.25	2.78
d	4.39	5.17	3.43	4.25
e	15.35	18.10	10.26	12.70
f	1.36	1.60	1.80	2.23
g	2.67	3.15	1.63	2.02
h	4.36	5.14	4.92	6.09
i	6.38	7.52	5.63	6.97
j	0.16	0.19	0.12	0.15
k	0.96	1.13	0.62	0.77
l	2.93	3.45	3.25	4.03
m	2.13	2.51	1.94	2.41

	Deutsch		Englisch	
n	8.84	10.42	5.45	6.75
o	1.90	2.24	6.06	7.51
p	0.50	0.59	1.56	1.93
q	0.01	0.01	0.08	0.10
r	6.86	8.08	4.84	5.99
s	5.39	6.35	5.11	6.33
t	4.73	5.57	7.31	9.06
u	3.48	4.10	2.23	2.76
v	0.72	0.87	0.77	0.96
w	1.42	1.67	1.91	2.36
x	0.01	0.01	0.12	0.15
y	0.02	0.02	1.59	1.97
z	1.42	1.67	0.06	0.07

Häufigkeiten von Bi- und Trigrammen

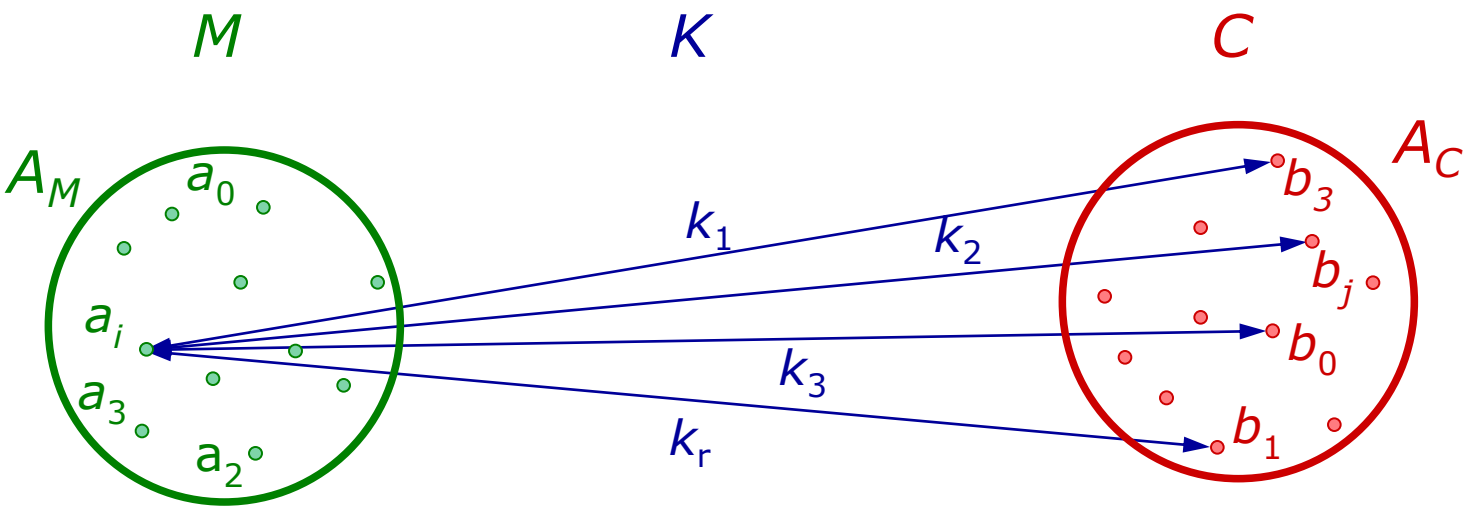
Rang				
1	EN	TH	EIN	THE
2	ER	HE	ICH	ING
3	CH	IN	NDE	AND
4	ND	ER	DIE	HER
5	EI	AN	UND	ERE
6	DE	RE	DER	ENT
7	IN	ED	CHE	THA
8	ES	ON	END	NTH
9	TE	ES	GEN	WAS
10	IE	ST	SCH	ETH
11	UN	EN	CHT	FOR
12	GE	AT	DEN	DTH
13	ST	TO	INE	HAT
14	IC	NT	NGE	SHE
15	HE	HA	NUN	ION

Rang				
16	NE	ND	UNG	INT
17	SE	OU	DAS	HIS
18	NG	EA	HEN	STH
19	RE	NG	IND	ERS
20	AU	AS	ENW	VER
21	DI	OR	ENS	TTH
22	BE	TI	IES	TER
23	SS	IS	STE	HES
24	NS	ET	TEN	EDT
25	AN	IT	ERE	EST
26	SI	AR	LIC	THI
27	UE	TE	ACH	HAD
28	DA	SE	NDI	OTH
29	AS	HI	SSE	ALL
30	NI	OF	AUS	ATI

Deutsch

Englisch

PM Substitutionen: polyalphabetisch, monographisch

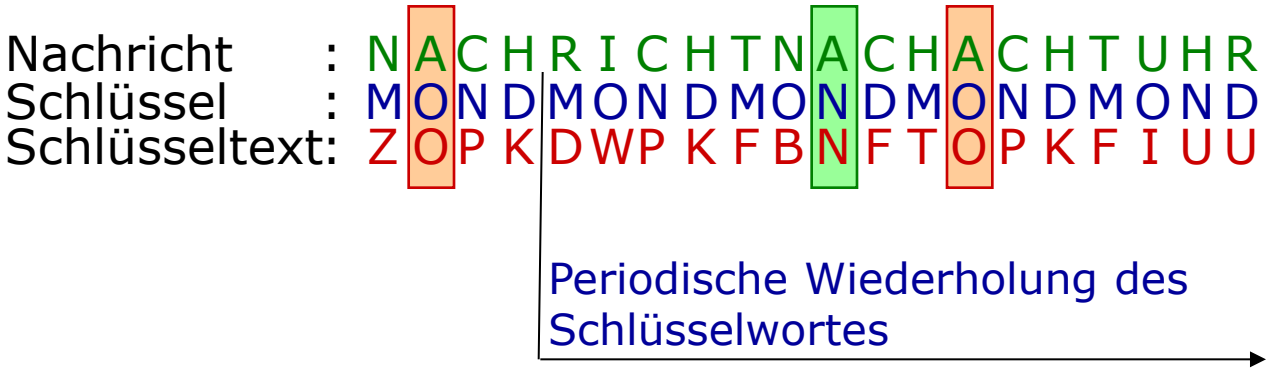


$$\text{enc}(k, m) = \text{enc}(k_0, m_0)\ \text{enc}(k_1, m_1)\ \ldots\ \text{enc}(k_l, m_l)$$

Polyalphabetische Substitutionen

Vigenère-Chiffre

- Schlüsselwort (bestehend aus den Buchstaben A..Z), das periodisch wiederholt wird
- Verschlüsselung einzelner Zeichen entspricht prinzipiell der Verschiebechiffre



Klassische Darstellung: Vigenère-Tableau

	a	b	c	d	e	f	…	z
A	A	B	C	D	E	F	…	Z
B	B	C	D	E	F	G	…	A
C	C	D	E	F	G	H	…	B
D	D	E	F	G	H	I	…	C
E	E	F	G	H	I	J	…	D
F	F	G	H	I	J	K	…	E
…	…	…	…	…	…	…	… … …	…
Z	Z	A	B	C	D	E	…	Y

Polyalphabetische Substitutionen – Kryptoanalyse

- Statistische Eigenschaften des Klartextes werden nicht in den Schlüsseltext übertragen
- Unter bestimmten Bedingungen sicher (Schlüssellänge!)

2 Schritte:

1. Ermittlung der Schlüssellänge r

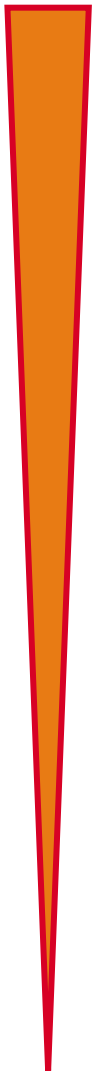
→ Vereinfachung auf Analyse von r einfachen Substitutionen

2. Analyse der einfachen Substitutionen

Schlüssel	k_0	k_1	k_2	…	k_{r-1}
Schlüsseltext	c_0	c_1	c_2	…	c_{r-1}
	c_r	c_{r+1}	c_{r+2}	…	$c_{r+(r-1)}$
	c_{2r}	c_{2r+1}	c_{2r+2}	…	$c_{2r+(r-1)}$
	…	…	…	…	…

einfache Substitution mit Schlüssel k_0

Angriffsziel/ -erfolg



a) Schlüssel (total break)

b) zum Schlüssel äquivalentes Verfahren (universal break)

c) einzelne Nachrichten,

z.B. speziell für Authentikationssysteme

c1) eine gewählte Nachricht (selective break)

c2) irgendeine Nachricht (existential break)

Angriffstypen

Schwere

a) passiv

- **a1) reiner Schlüsseltext-Angriff (ciphertext-only attack)**
- **a2) Klartext-Schlüsseltext-Angriff (known-plaintext attack)**

b) aktiv

(je nach Kryptosystem; asym.: eins von beiden: b1 oder b2; sym.: ggf. beides: auch b1 und b2)

- **b1) Signatursystem: Klartext → Schlüsseltext (Signatur) (chosen-plaintext attack)**
- **b2) Konzelationss.: Schlüsseltext → Klartext (chosen-ciphertext attack)**

Adaptivität

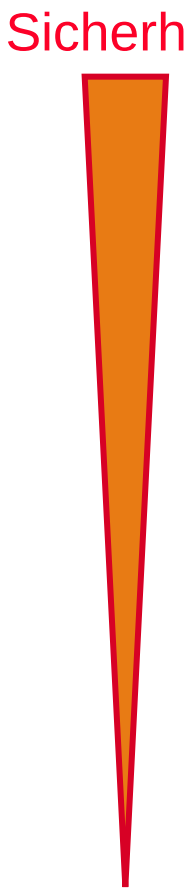
- **nicht adaptiv**
- **adaptiv**

Kriterium: Handlung Erlaubnis

passiver Angreifer ≠

beobachtender Angreifer

Sicherheitsklassen kryptographischer Systeme



1. **informationstheoretisch sicher**
2. **kryptographisch stark**
3. **wohluntersucht**
4. **wenig untersucht**
5. **geheim gehalten**

Sicherheit: Was kann man bestenfalls erreichen?

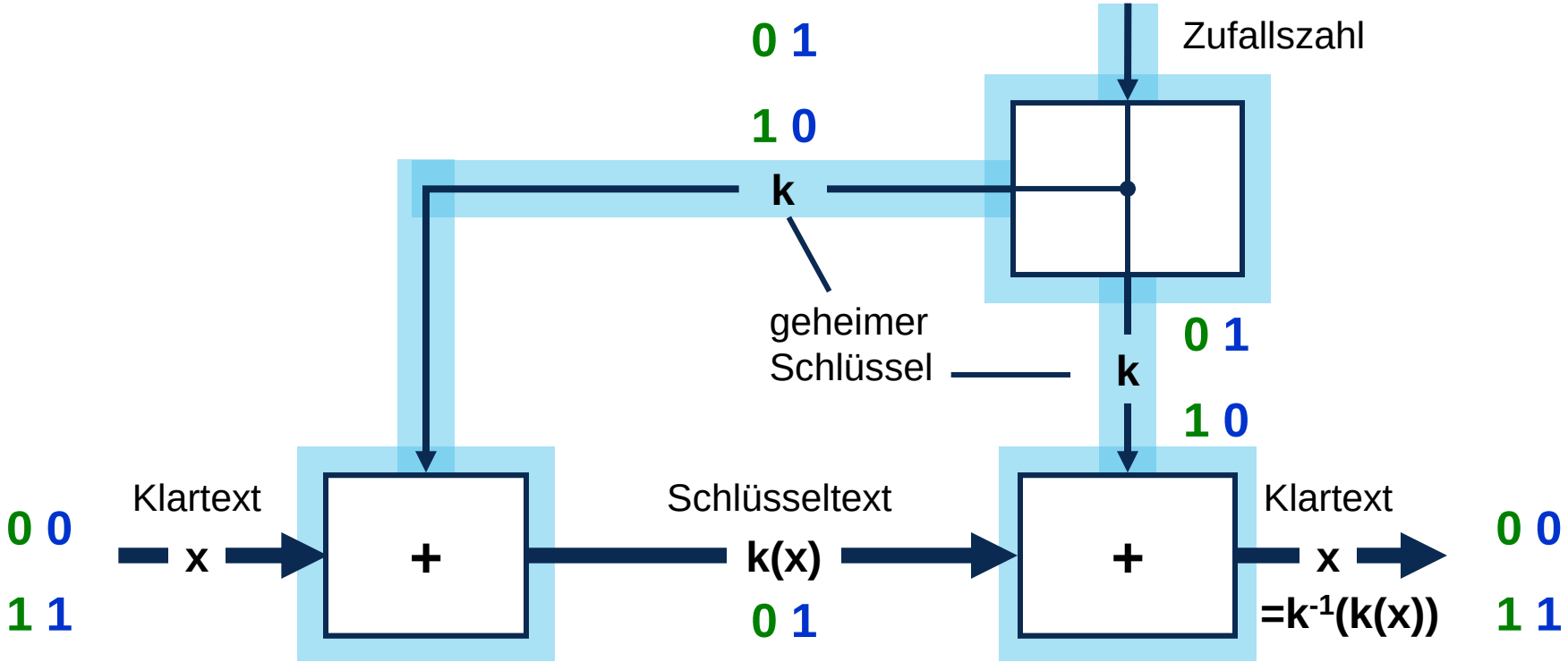
Informationstheoretische Sicherheit [Shannon, 1949]

Ein unbeschränkter Angreifer (beliebig viel Zeit und Ressourcen) darf aus der Beobachtung von Schlüsseltext keinerlei Informationen über Klartext oder Schlüssel erhalten.

→ apriori wissen = a posteriori Wissen

- *a priori* Wissen:
  Wissen des Angreifers über mögliche Nachrichten *vor* einer Beobachtung (als bekannt vorausgesetzt)
- *a posteriori* Wissen:
  Wissen des Angreifers über mögliche Nachrichten *nach* der Beobachtung des gesendeten Schlüsseltexts

Bsp. Vernam-Chiffre (=one-time-pad)



Geheimer Bereich

Für informationsth. Sicherheit *müssen* Schlüssel so lang sein

Sei $\mathcal{K}$ Schlüsselmenge, $\mathcal{X}$ Klartextmenge und S Menge der mindestens einmal auftretenden Schlüsseltexte.

$|S| \geq |\mathcal{X}|$ damit eindeutig entschlüsselbar (k fest)

$|\mathcal{K}| \geq |S|$ damit hinter jedem Schlüsseltext jeder Klartext stecken kann (x fest)

also $|\mathcal{K}| \geq |\mathcal{X}|$.

Falls Klartext geschickt codiert, folgt:

Schlüssel mindestens so lang wie Klartext.

Definitionen für informationstheoretische Sicherheit

1. Definition für informationstheoretische Sicherheit

(alle Schlüssel mit gleicher Wahrscheinlichkeit gewählt)

$$\forall S \in \mathcal{S}\ \exists\, const \in \mathbb{N}\ \forall x \in \mathcal{X}\colon |\{k \in \mathcal{K} \mid k(x) = S\}| = const. \qquad (1)$$

Die a-posteriori-Wahrscheinlichkeit eines Klartextes x, wenn der Angreifer den Schlüsseltext S gesehen hat, ist $W(x|S)$.

2. Definition

$$\forall S \in \mathcal{S}\ \forall x \in \mathcal{X}\colon W(x|S) = W(x). \qquad (2)$$

Beide Definitionen sind äquivalent:

Nach Bayes gilt:

$$W(x \mid S) = \frac{W(x) \bullet W(S \mid x)}{W(S)}$$

(2) ist also äquivalent zu

$$\forall S \in \mathcal{S}\ \forall x \in \mathcal{X}\colon W(S|x) = W(S). \qquad (3)$$

Wir zeigen, dass dies äquivalent ist zu

$$\forall S \in \mathcal{S}\ \exists\, const' \in \mathbb{R}\ \forall x \in \mathcal{X}\colon W(S|x) = const'. \qquad (4)$$

Beweis

(3)⇒(4) ist klar mit $const' := W(S)$.

Umgekehrt zeigen wir $const' = W(S)$:

$$W(S) = \sum_x W(x) \bullet W(S|x)$$

$$= \sum_x W(x) \bullet const'$$

$$= const' \bullet \sum_x W(x)$$

$$= const'.$$

(4) sieht (1) schon sehr ähnlich: Allgemein ist

$$W(S|x) = W(\{k \mid k(x) = S\}),$$

und wenn alle Schlüssel gleichwahrscheinlich sind,

$$W(S|x) = |\{k \mid k(x) = S\}| / |\mathcal{K}|.$$

Dann ist (4) äquivalent (1) mit

$$const = const' \bullet |\mathcal{K}|.$$

Sicherheitsdefinitionen

[http://de.wikipedia.org/wiki/Sicherheitseigenschaften_kryptografischer_Verfahren]

Semantische Sicherheit (SEM)

- jede Information, die aus dem Schlüsseltext abgeleitet werden kann, kann auch aus der Länge des Schlüsseltext abgeleitet werden

Real-Or-Random-Sicherheit (ROR)

- ununterscheidbar, ob Schlüsseltext zu einer selbst gewählten Nachricht gehört oder einer zufälligen Nachricht gleicher Länge

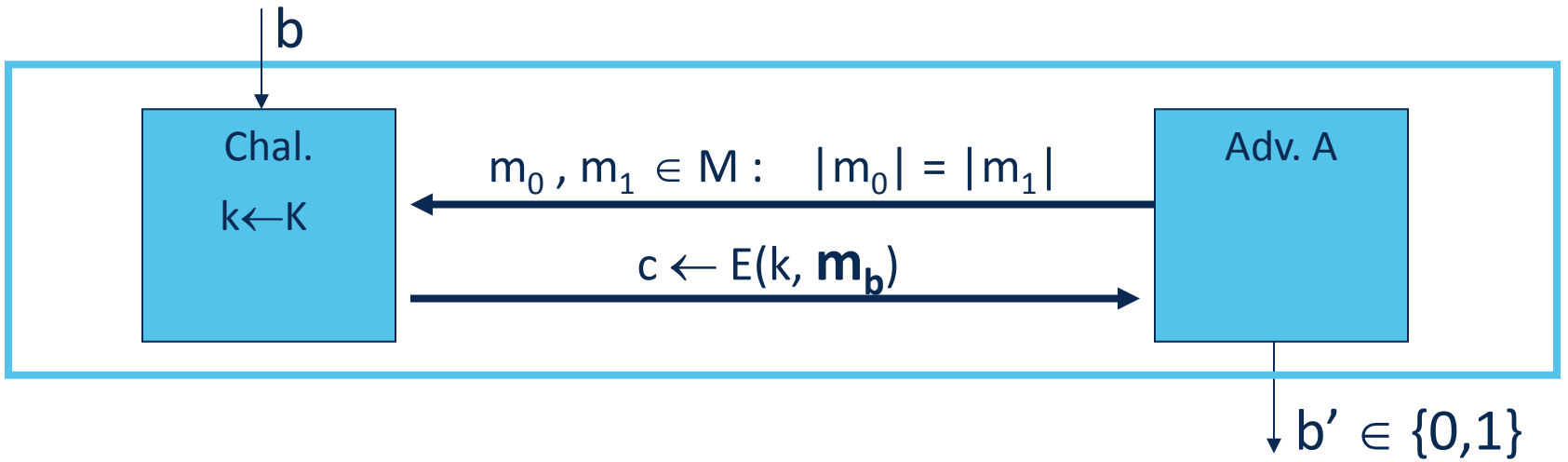
Ciphertext Indistinguishability (IND)

- ununterscheidbar zu welcher von zwei selbstgewählten Klartexten ein Schlüsseltext gehört

➔ die 3 Definitionen sind äquivalent

Semantic Security (for one time key)

For b=0,1 define experiments EXP(0) and EXP(1) as:



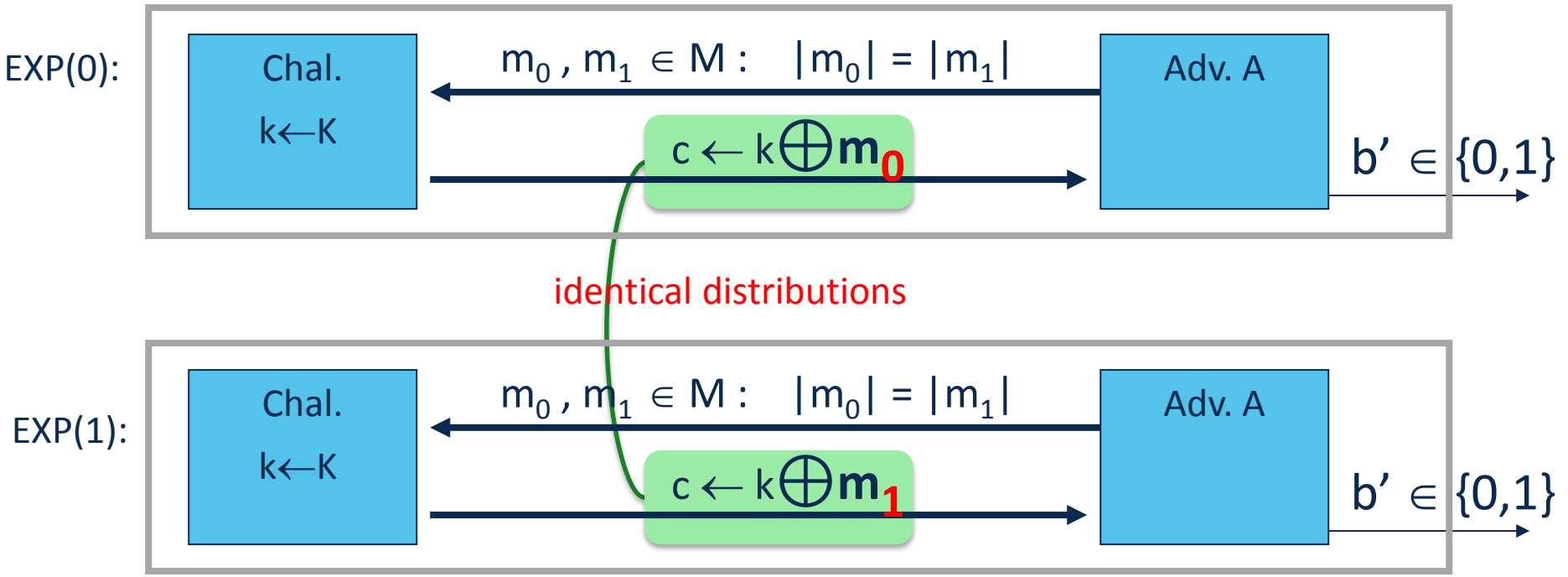
for b=0,1: W_b := [event that EXP(b) = 1]

$$Adv_{SS}[A,E] := |\Pr[W_0] - \Pr[W_1]| \quad \in [0,1]$$

Again:

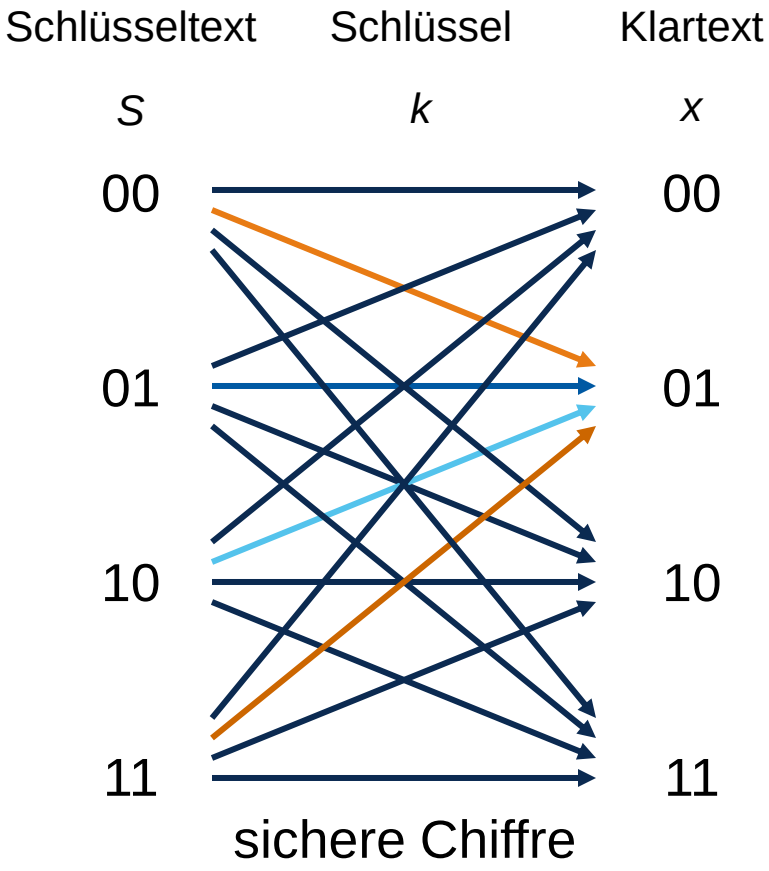
E is called ***semantically secure*** if for all eff. A, $Adv_{SS}[A,E]$ is negligible

OTP is semantically secure



For **all** A: $Adv_{SS}[A,OTP] = \left| Pr[\, \mathbf{A}(k \oplus \mathbf{m_0}) = 1 \,] - Pr[\, \mathbf{A}(k \oplus \mathbf{m_1}) = 1 \,] \right|$

Wie passen die verschiedenen Verteilungen zusammen?



sichere Chiffre

gleich-verteilt

gleichverteilt, aber *nicht* unabhängig von den Schlüsseltexten

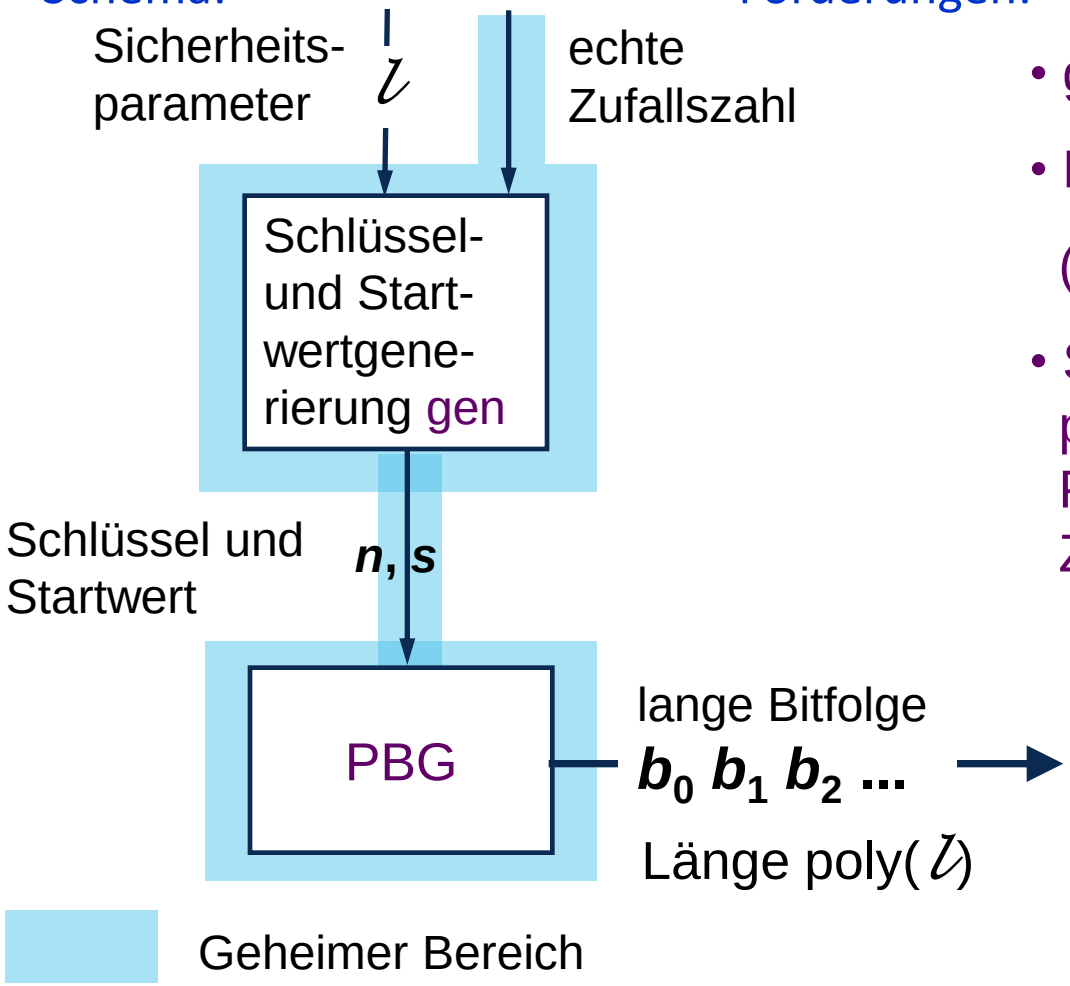
ungleich verteilt

Gleichverteilte Schlüsseltexte entschlüsselt mit gleichverteilten Schlüsseln kann ungleich verteilte Klartexte dann und nur dann ergeben, wenn die Gleichverteilungen nicht unabhängig voneinander sind, d.h. die Schlüsseltexte wurden aus Klartexten und Schlüsseln berechnet.

Der Pseudozufallsbitfolgengenerator (PBG)

Idee: kurzer Startwert (seed) → lange Bitfolge (soll zufällig sein aus Sicht von polynomialen Angreifern)

Schema:



Forderungen:

- gen und PBG sind effizient
- PBG ist deterministisch

  (⇒ Folge reproduzierbar)
- Sicher: Kein probabilistischer polynomialer Test kann PBG-Folgen von echten Zufallsfolgen unterscheiden

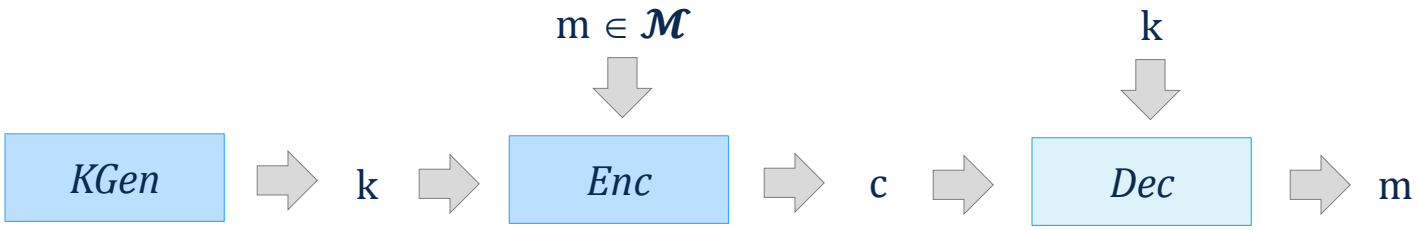
Confidential Communication

Spaces

$\mathcal{M}$	plaintext space (e.g. words over an alphabet)
$\mathcal{C}$	space of ciphertexts
$\mathcal{K}$	space of keys

Algorithms of a private-key (symmetric) encryption scheme

KGen	generates some (usually random) *key* k
Enc	encrypts a *plaintext* m using key k and outputs the ciphertext c
Dec	decrypts a *ciphertext* c using key k and outputs the plaintext m



Correctness $\quad \text{for all } k \in \mathcal{K}, m \in \mathcal{M} \ : \ \text{Dec}(k, \text{Enc}(k, m)) = m$

Perfect Secrecy

Observation: Patterns are your enemy!

Concept:

- Long key (long/no periodicity)
- No recognizable pattern

Gilbert Vernam
(1890-1960)

Key Generation

choose $k = (k_1, \ldots, k_n)$ where each k_i is truly random permutation

Encryption

Let $m = m_0 \ldots m_n$.

$Enc(k, m) = c_0 \ldots c_n$ where $c_i = f(k_i, m_i) = k_i \oplus m_i$

The One-Time-Pad (Vernam cipher)

Truly random key, as long as the message:

m =	A	T	T	A	C	K	T	H	E	C	I	T	Y	A	T	T	W	E	L	V	E
k =	P	S	P	I	U	H	G	D	S	P	H	G	D	S	P	I	W	E	E	W	O

(+ mod 26)

c =	P	L	I	I	W	R	Z	K	W	R	P	Z	B	S	I	B	S	I	P	R	S

k =	Y	H	P	R	S	R	G	F	F	D	D	X	N	S	Q	I	S	P	W	N	F
m =	R	E	T	R	E	A	T	F	R	O	M	C	O	A	S	T	A	T	T	E	N

(+ mod 26)

… or any other message of the same length, for that matter

Now, what are the two problems with this method?

Information Theoretic Security

Shannon (1949): „CT should not reveal ***any*** information about PT“

Def: A cipher (E,D) over ($\mathcal{K}$, $\mathcal{M}$, $\mathcal{C}$) has ***perfect secrecy*** if

$\forall\ m_0, m_1 \in \mathcal{M}$ (with $len(m_0) = len(m_1)$)
$\forall\ c \in \mathcal{C}$ and $k \xleftarrow{R} \mathcal{K}$:

$$Pr[E(k,m_0) = c] = Pr[E(k,m_1) = c]$$

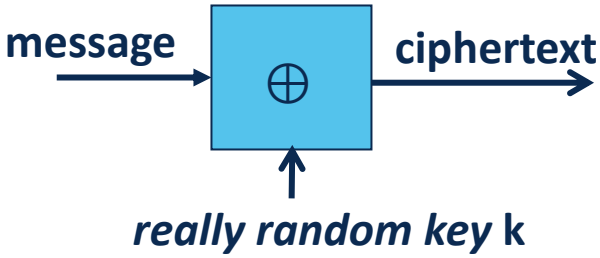
So being an attacker, what do I learn?

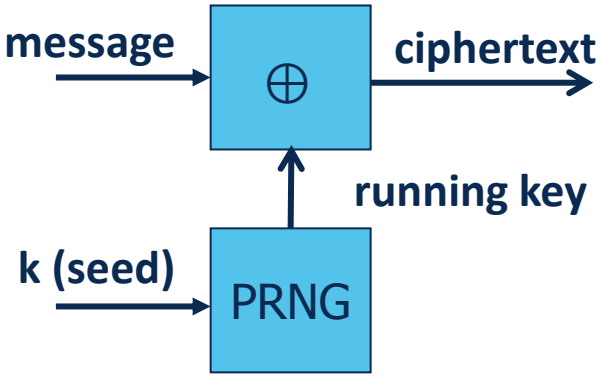
No CT attack can tell if msg is m_0, m_1 (or any other message)

→ No CT only attacks

Stream ciphers in general

OTP:





Idea: replace random by „pseudorandom“ key

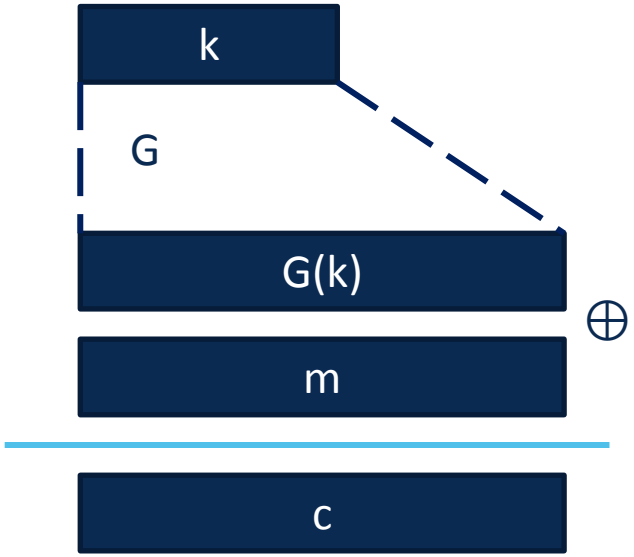
PRNG is a function G: $\{0,1\}^s \rightarrow \{0,1\}^n$ n >> s

Det. algorithm from seed space to key space (looking random)

Making the OTP practical

$C := E(k,m) = m \oplus G(k)$

$D(k,c) = c \oplus G(k)$



Can this achieve perfect secrecy?

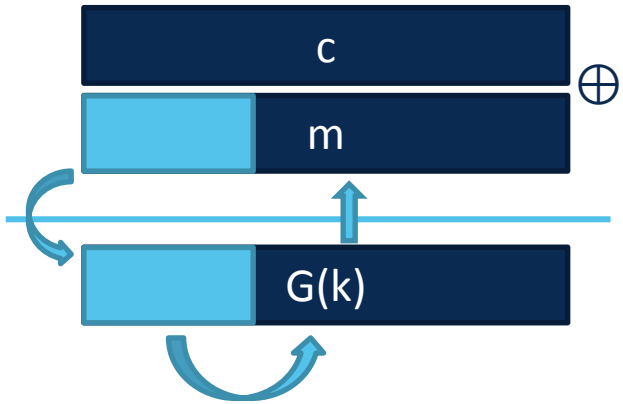
So, how „secure“ is it then?

The problem of predictable PRNG

What does „PRNG is predictable “ mean?

Suppose ∃i: $G(k)|_{1,\dots,i} \xrightarrow{alg.} G(k)|_{i+1,\dots,n}$

Then:



We call a PRNG ($G: K \longrightarrow \{0,1\}^n$) ***predictable***, if:

∃ efficient algorithm A and ∃ $0 \leq i \leq n-1$ such that

$$\Pr_{k \leftarrow \mathcal{K}}[A(G(k)|_{1,..,i}) = G(k)|_{i+1}] > ½ + \varepsilon \quad \text{(for non-negl. } \varepsilon\text{)}$$

PRNG is ***unpredictable***: ∀i: no adv. can predict bit (i+1) for n.n. ε

Secure ciphers: another attempt

Recall Shannons perfect secrecy:

(E,D) has perfect secrecy, if $\forall m_0, m_1 \in \mathcal{M}\ (|m_0|=|m_1|)$

$$\{E(k, m_0)\} = \{E(k, m_1)\} \text{ where } k \xleftarrow{R} \mathcal{K}$$

Let‘s say instead:

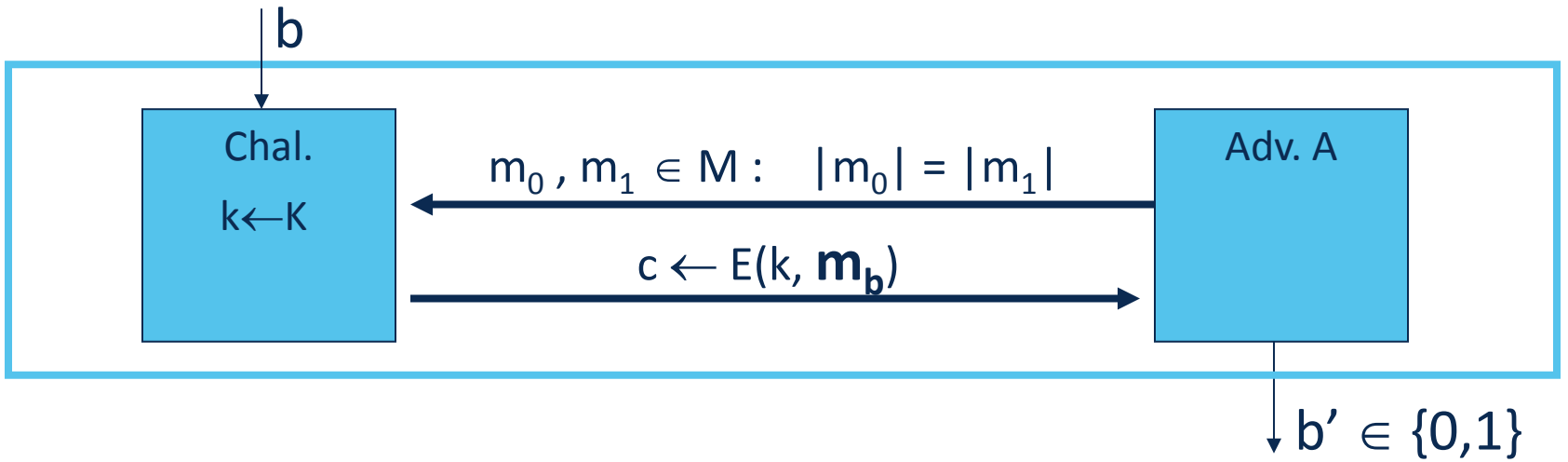
(E,D) has „some“ secrecy, if $\forall m_0, m_1 \in \mathcal{M}\ (|m_0|=|m_1|)$

$$\{E(k, m_0)\} \approx_p \{E(k, m_1)\} \text{ where } k \xleftarrow{R} \mathcal{K}$$

under given m_0 and m_1

Semantic Security (for one time key)

For b=0,1 define experiments EXP(0) and EXP(1) as:



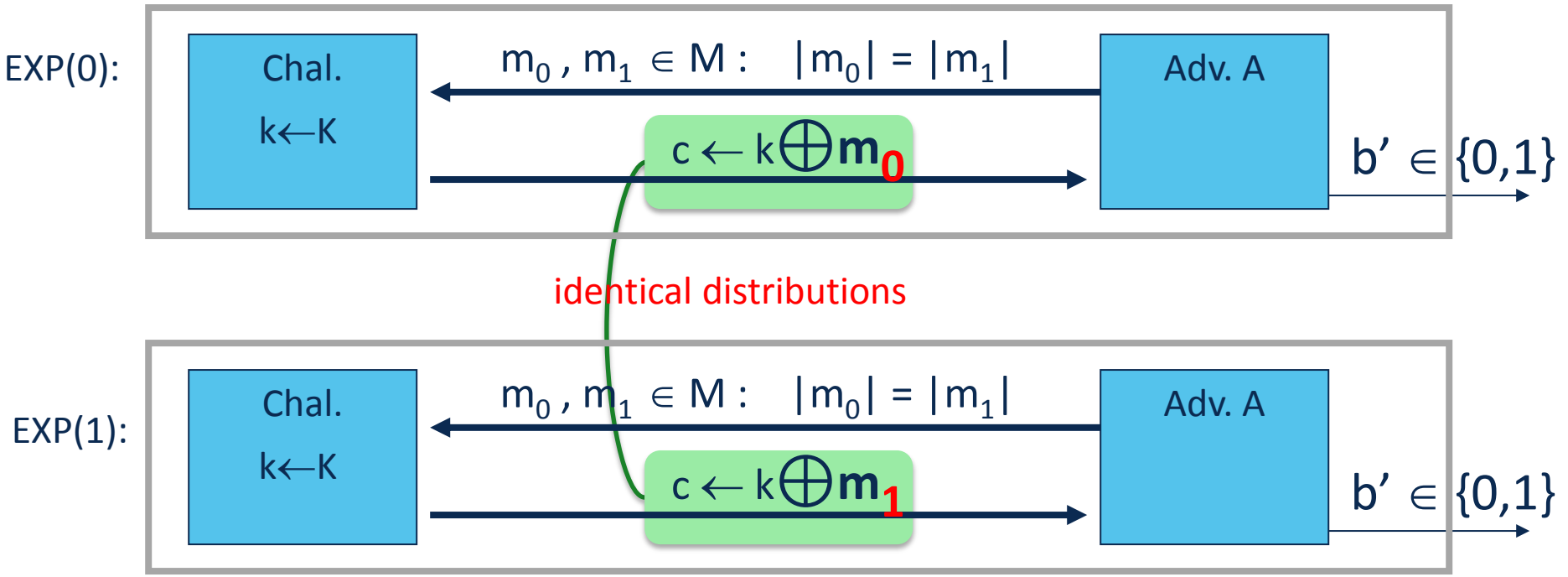
for b=0,1: W_b := [event that EXP(b) = 1]

$$Adv_{SS}[A,E] := |\Pr[W_0] - \Pr[W_1]| \quad \in [0,1]$$

Again:

E is called ***semantically secure*** if for all eff. A, $Adv_{SS}[A,E]$ is negligible

OTP is semantically secure



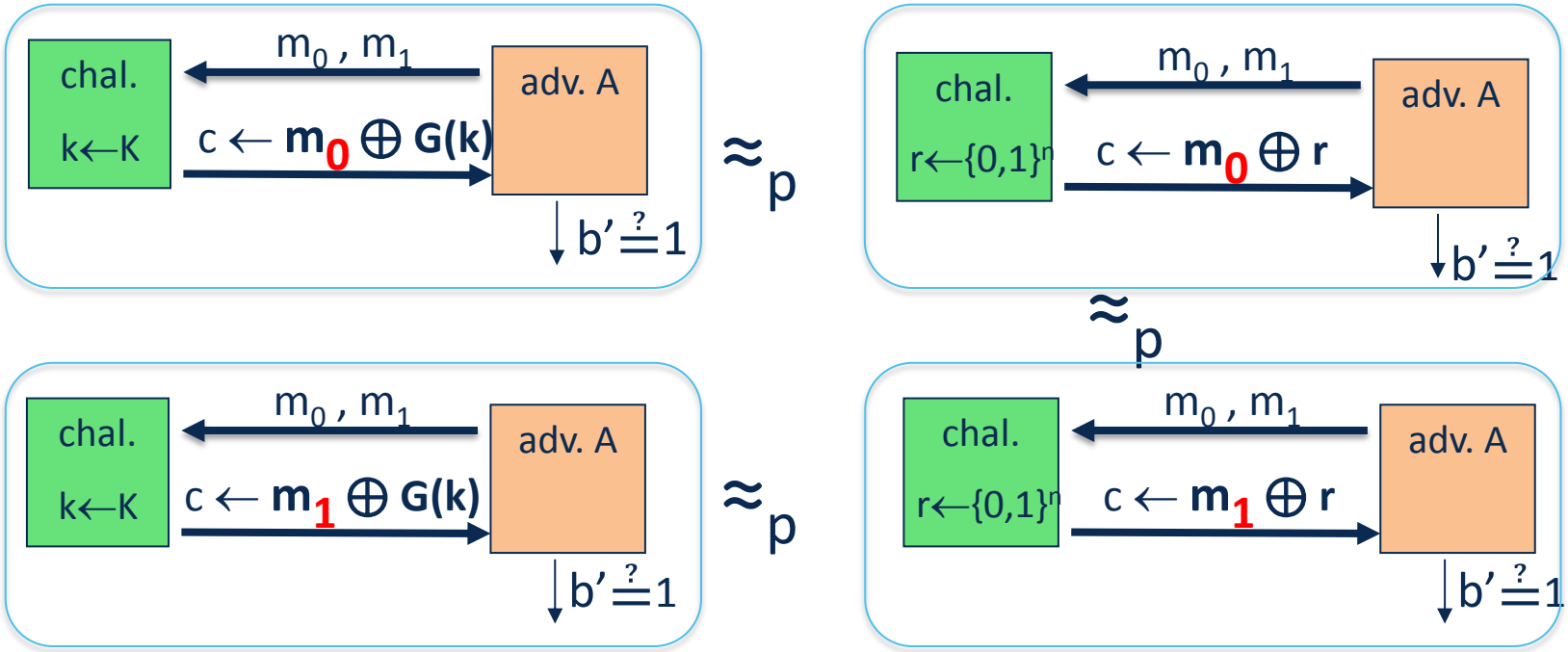
For **all** A: $Adv_{SS}[A,OTP] = \left| Pr[\, \mathbf{A}(k \oplus \mathbf{m_0}) = 1 \,] - Pr[\, \mathbf{A}(k \oplus \mathbf{m_1}) = 1 \,] \right|$

So, are stream ciphers semantically secure?

We have shown:

- Unpredictable PRNG are secure
- OTP (XOR with truly random bitstring) is secure

Proof intuition:



Sicherheit von Strom-Chiffren

Idee:

Wenn kein probabilistischer polynominaler Test Pseudozufallsfolgen von echten Zufallsfolgen unterscheiden kann, dann ist eine Stromchiffre, die auf einem sicheren PRNG basiert gegen polynomiale Angreifer so gut wie echtes one-time-pad.

Trotzdem gilt: wir implementieren NIEMALS eine Chiffre!!!1!

(Sonst ist der Angreifer ein Test !)

Konstruktion geht also mit jedem guten PRNG

Two known problems of OTP constructions

The ***two*** time pad:

Assume you use a stream cipher ***key*** more than once:

$$c_1 = m_1 \oplus \text{PRNG}(k)$$

$$c_2 = m_2 \oplus \text{PRNG}(k)$$

How can this be attacked?

$$c_1 \oplus c_2 = (m_1 \oplus \text{PRNG}(k)) \oplus (m_2 \oplus \text{PRNG}(k)) = 0 \oplus m_1 \oplus m_2$$

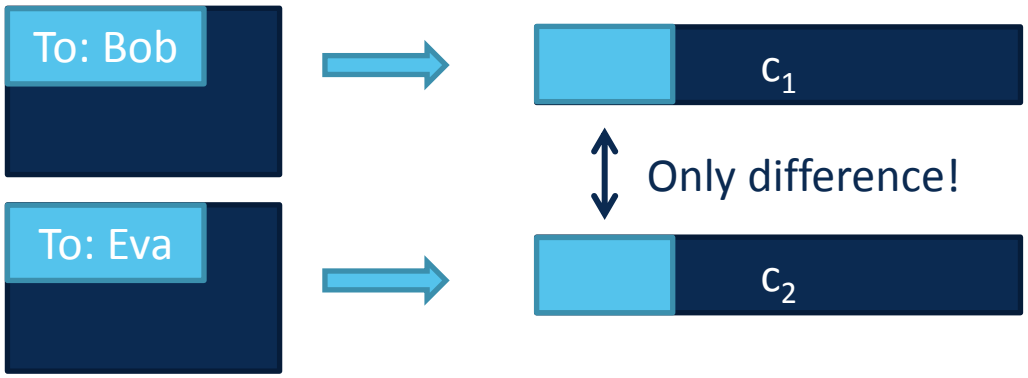
ASCII and natural language highly redundant and regular,

deriving m_1, m_2 from $m_1 \oplus m_2$ is easy…

Two Time Pads in the real world

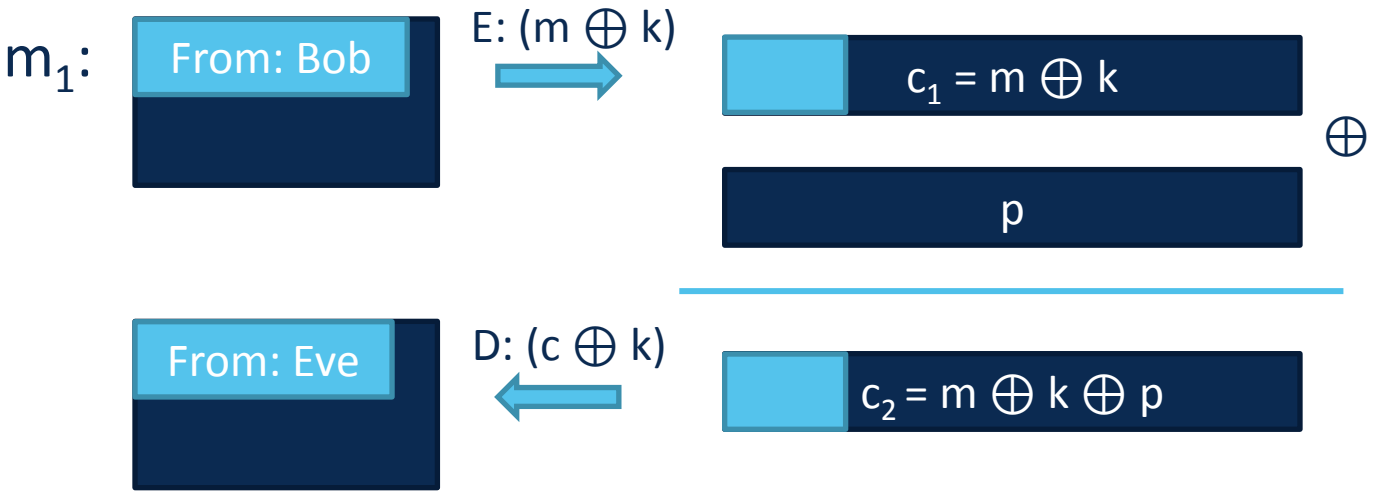
Examples are plentiful:

- Project Venona
- MS-PPTP (Win NT: two streams of messages enc with same key)
  - Lesson learned: use separate key per direction!
- WEP (802.11b)
  - $PRG(IV||k)$ to avoid identical keys (k=long time key)
  - IV 24 bits and commonly reset to 0 after power cycle
  - After each cycle ($2^{24} \approx 16M$) again a two time pad
- Disk encryption:



Lack of integrity (Malleability)

Assuming a similar example as before:



In this simple example:

Bob = 42 6F 62; Eve = 45 76 65; Bob ⊕Eve = 07 19 07 (p:= 0 0 0 0 07 19 07)

Lesson: Modification is undetected and has predictable impact!

Zusammenfassung zu Stromchiffren

Alice, Bob und Eve sind Ihnen gute Bekannte

Sie kennen die Konzepte von Substitution und Permutation

Sie haben einige klassische Chiffren kennengelernt

Sie kennen die Schwachstellen historischer Chiffren

Sie haben das OTP kennengelernt (und wissen, wie sicher es ist und warum)

Informationstheoretische und semantische Sicherheit können Sie erklären

Sie kennen Konstruktionen für Stromchiffren

Sie verstehen das Two-Time-Pad Problem und Malleability

Teil 2: Block-Chiffren

Erinnerung an Funktionstheorie

Pseudo-Zufallsfunktionen (PRF) und Pseudo-Zufallspermutationen (PRP)

Grundidee der Block-Chiffren

Substitutions-Permutations Netzwerk

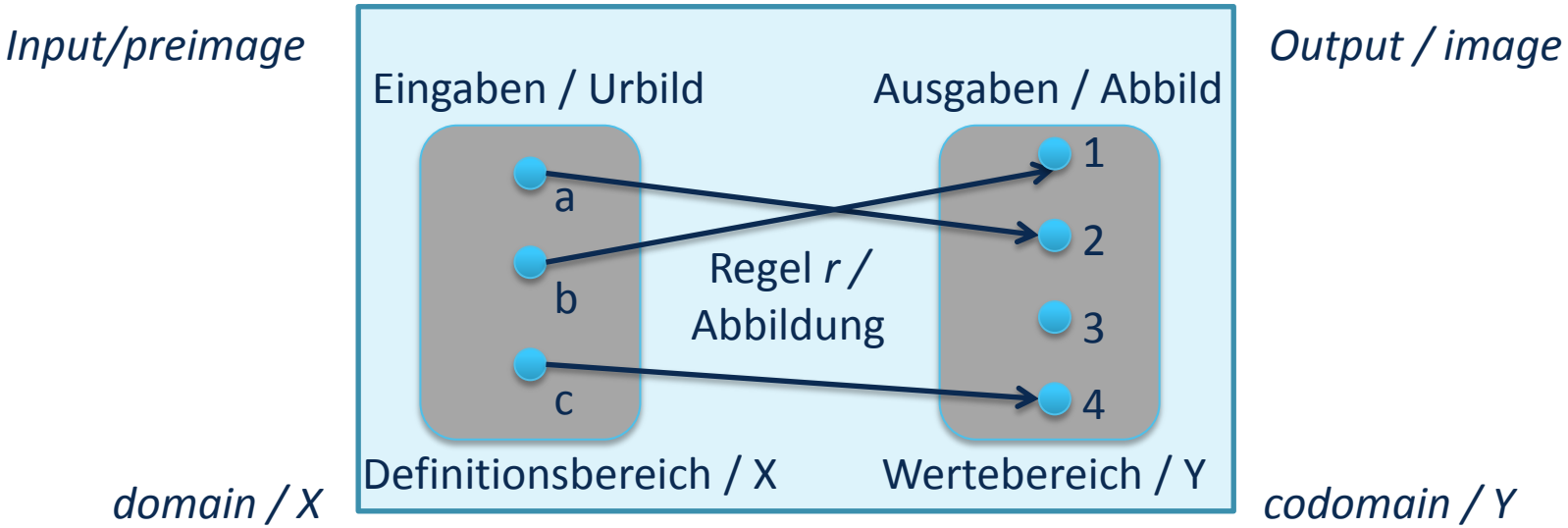
Feistel Netzwerke

Zwei Beispiele: DES und AES

Verschlüsselung langer Sequenzen (Operationsmodi)

ECB, DCM, CBC mit zufälligem IV, OFB, R-CTR

Kleine Erinnerung zu Funktionen



$f: X \longrightarrow Y$

$X = \{a,b,c\}$

$Y=\{1,2,3,4\}$

$y=f(x)$

$Im(f) = \{1,2,4\}$

Pseudo Random Functions and Permutations

Pseudo Random Functions (PRF):

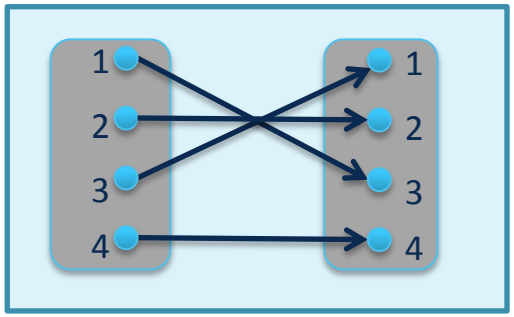
$$F: K \times X \longrightarrow Y$$

on „domain“ X and „range“ K, with „efficient“ algorithm to evaluate F(k,x)

Permutations:

A *permutation* π is a *bijective function* from a domain to itself:

$$\pi: X \longrightarrow X \qquad Im(f) = X$$



Pseudo Random Permutation (PRP):

Permutation $E: K \times X \longrightarrow X$

- has efficient deterministic algorithm to evaluate $E(k,x)$ ***and***
- efficient inversion algorithm $D = E^{-1}$

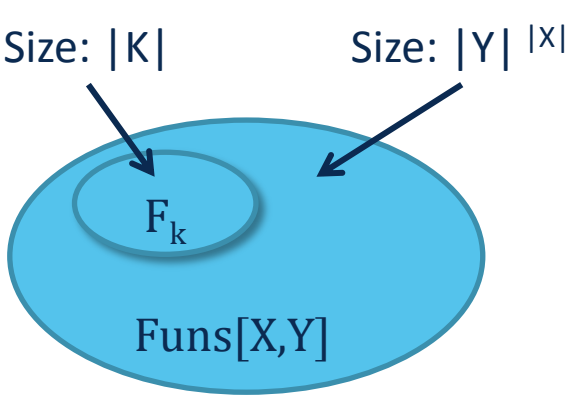
Security of PRFs

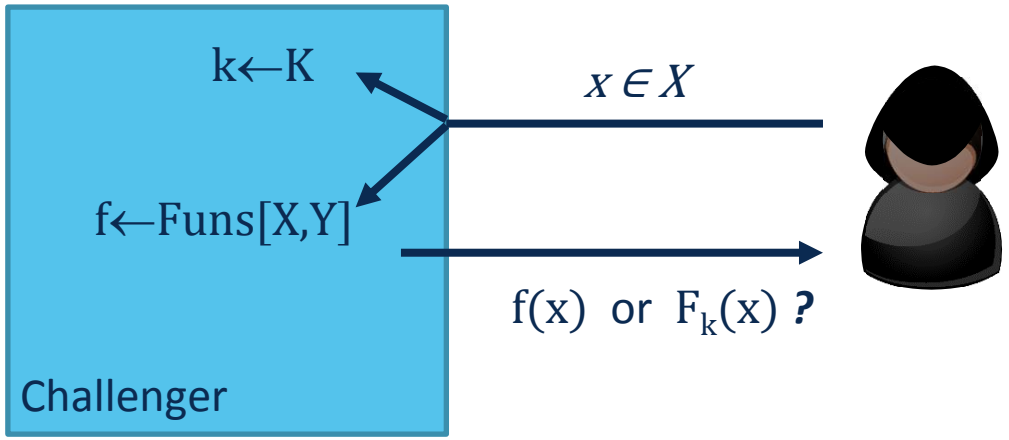
A PRF is secure, if it is indistinguishable from a random function:

Consider

Funs[X,Y]: the set of ***all*** functions from X to Y

PRF $F_k = \{F(k, \cdot) \text{ s.t. } k \in K\} \subseteq \text{Funs}[X,Y]$





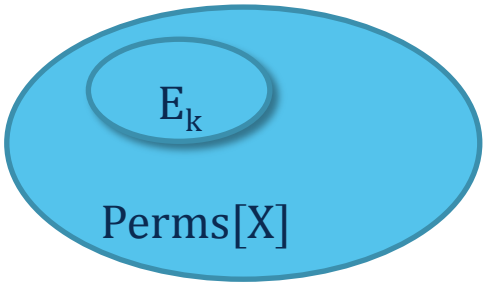
Security of PRPs

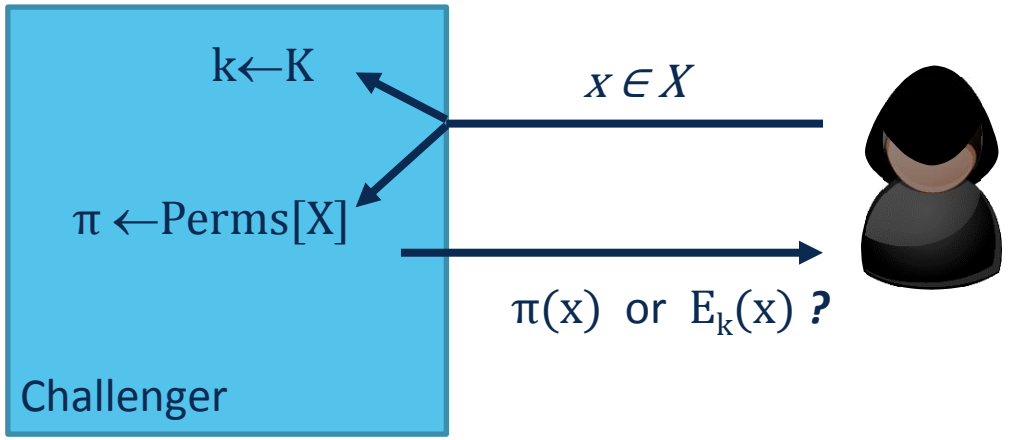
A PRP is secure, if it is indistinguishable from a random permutation:

Consider

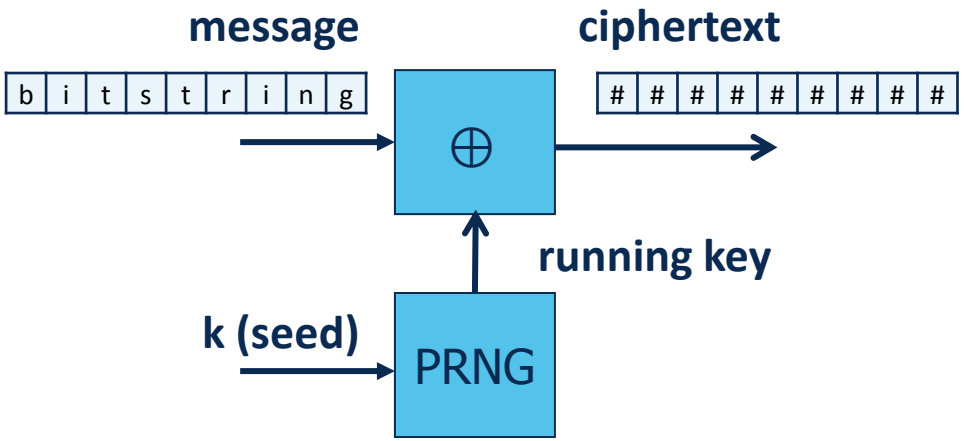
$\mathrm{Perms}[X]$: the set of ***all*** one-to-one functions from X to X

PRP $E_k = \{E(k, \cdot) \text{ s.t. } k \in K\} \subseteq \mathrm{Perms}[X]$



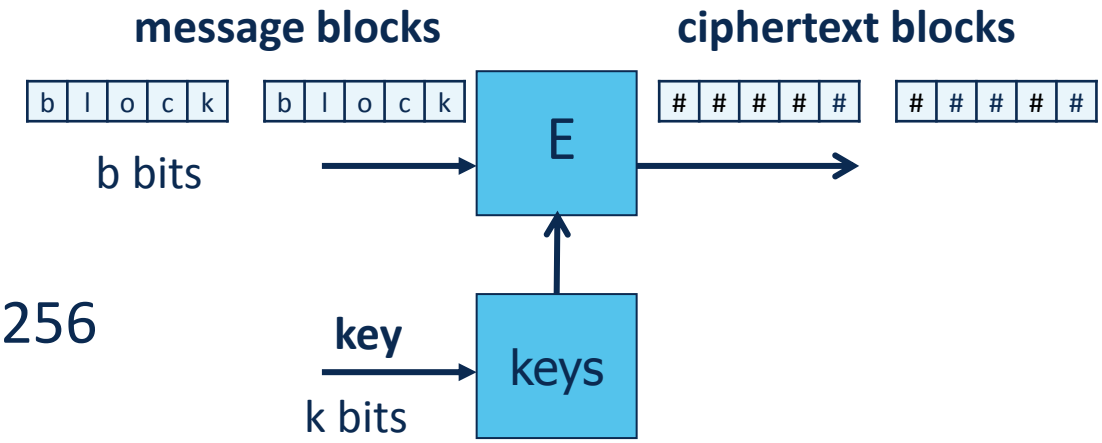


Stream Ciphers and Block Ciphers



Goal:

Build a secure PRP for b-bit blocks



Examples:

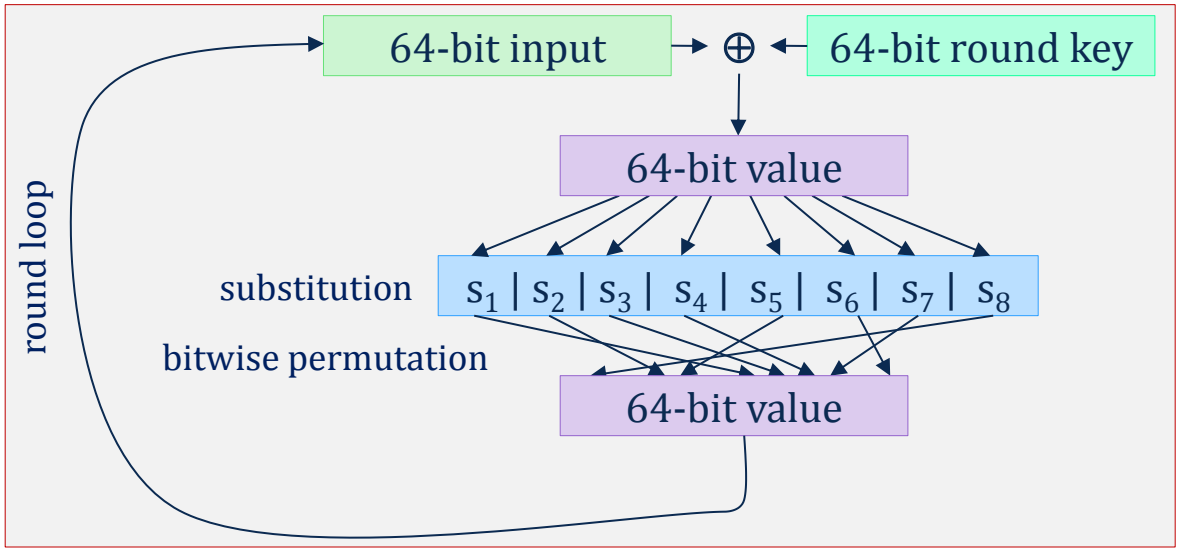
3DES: n = 64, k = 168

AES: n = 128, k = 128,192,256

Interlude: Substitution – Permutation Nets

SPN implement the Confusion – Diffusion Paradigm:

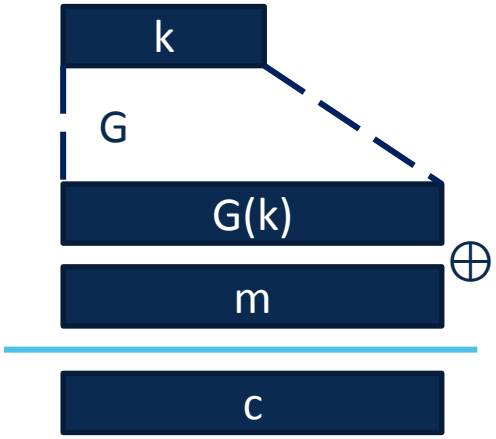
- Round keys k_i are derived from k, then usually $\oplus$-ed with intermediate round output
- round functions f_i are *fixed, invertible* substitution boxes (S-Box)



Rounds and Round Keys: Key Expansion

Recall from stream ciphers:

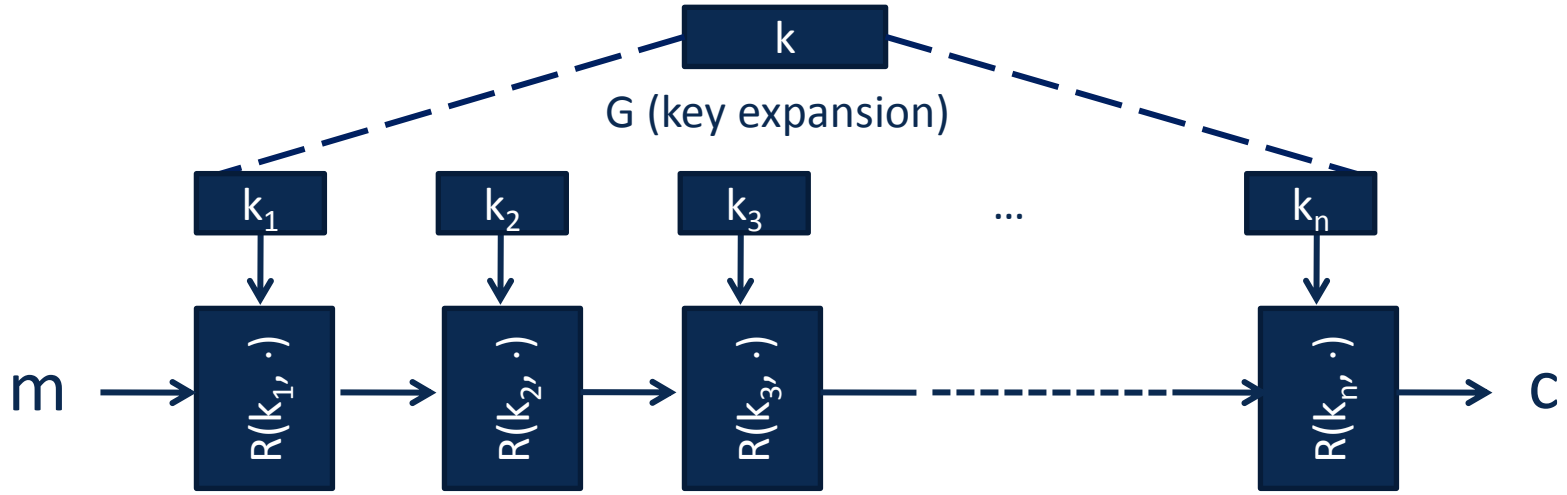
Short key expanded to encrypt bitstream



Idea:

Perform several keyed permutations in rounds

Expand key to round keys as parameters for random permutations



Feistel Networks

Horst Feistel

Goal:

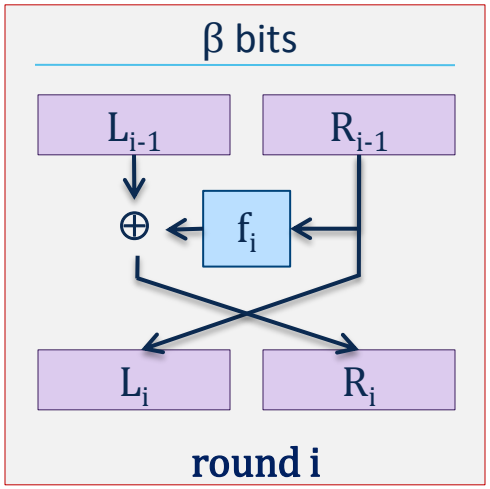
Create a PRP from arbitrary (non-invertible) functions

Idea:

$R_i = f_i(R_{i-1}) \oplus L_{i-1}$ $\qquad$ $L_i = R_{i-1}$

with round function f_i (possibly non-invertible),

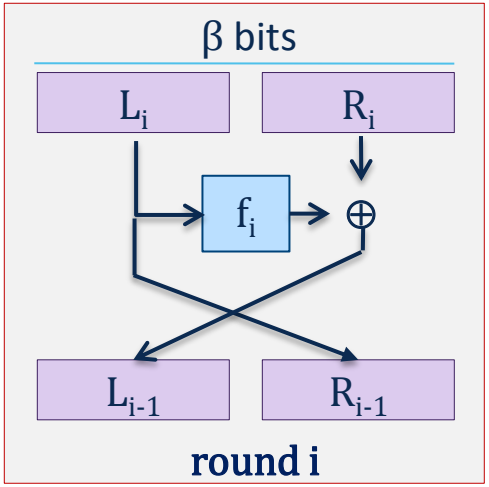
keyed with round key k_i



Inverting is easy (basically identical, f_1 to f_d reversed):

$R_{i-1} = L_i$

$L_{i-1} = R_i \oplus f_i(L_i)$



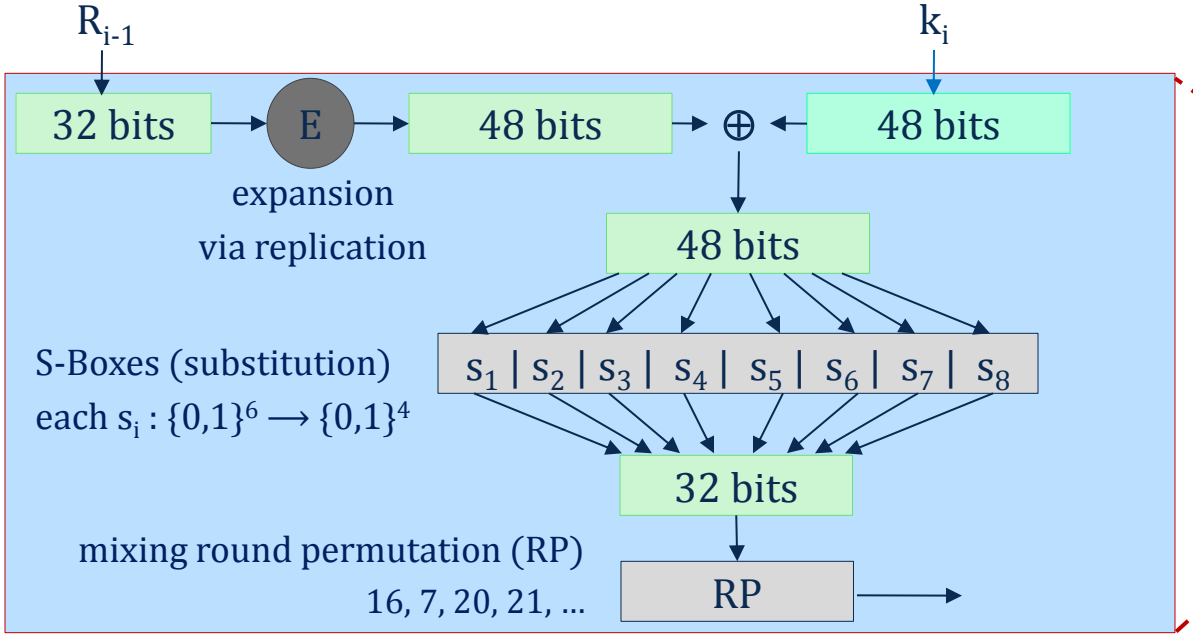
Luby-Rackoff '85: a 3 round Feistel-Network

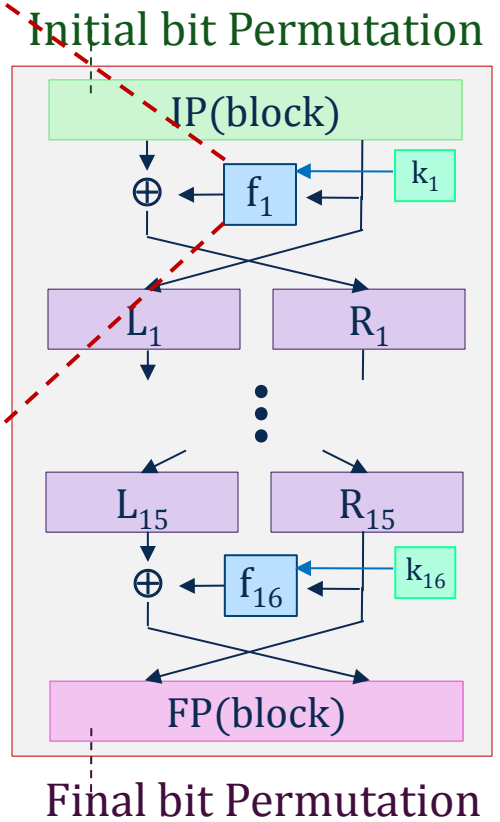
F: $K^3 \times \{0,1\}^{2n} \longrightarrow \{0,1\}^{2n}$, built using PRF, is a PRP

Lucifer and the Data Encryption Standard

„Lucifer“ at DES challenge (16 rounds; b,k = 128 bit FN, IBM)

Standardized as DES after adaptation (b=64, k = 56,…, due to NSA)





S_5	Middle 4 bits of input															
	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
Outer bits 00	0010	1100	0100	0001	0111	1010	1011	0110	1000	0101	0011	1111	1101	0000	1110	1001
Outer bits 01	1110	1011	0010	1100	0100	0111	1101	0001	0101	0000	1111	1010	0011	1001	1000	0110
Outer bits 10	0100	0010	0001	1011	1010	1101	0111	1000	1111	1001	1100	0101	0110	0011	0000	1110
Outer bits 11	1011	1000	1100	0111	0001	1110	0010	1101	0110	1111	0000	1001	1010	0100	0101	0011

The Advanced Encryption Standard

1997: NIST publishes request for proposal

1998: 15 submissions

1999: NIST chooses 5 finalists

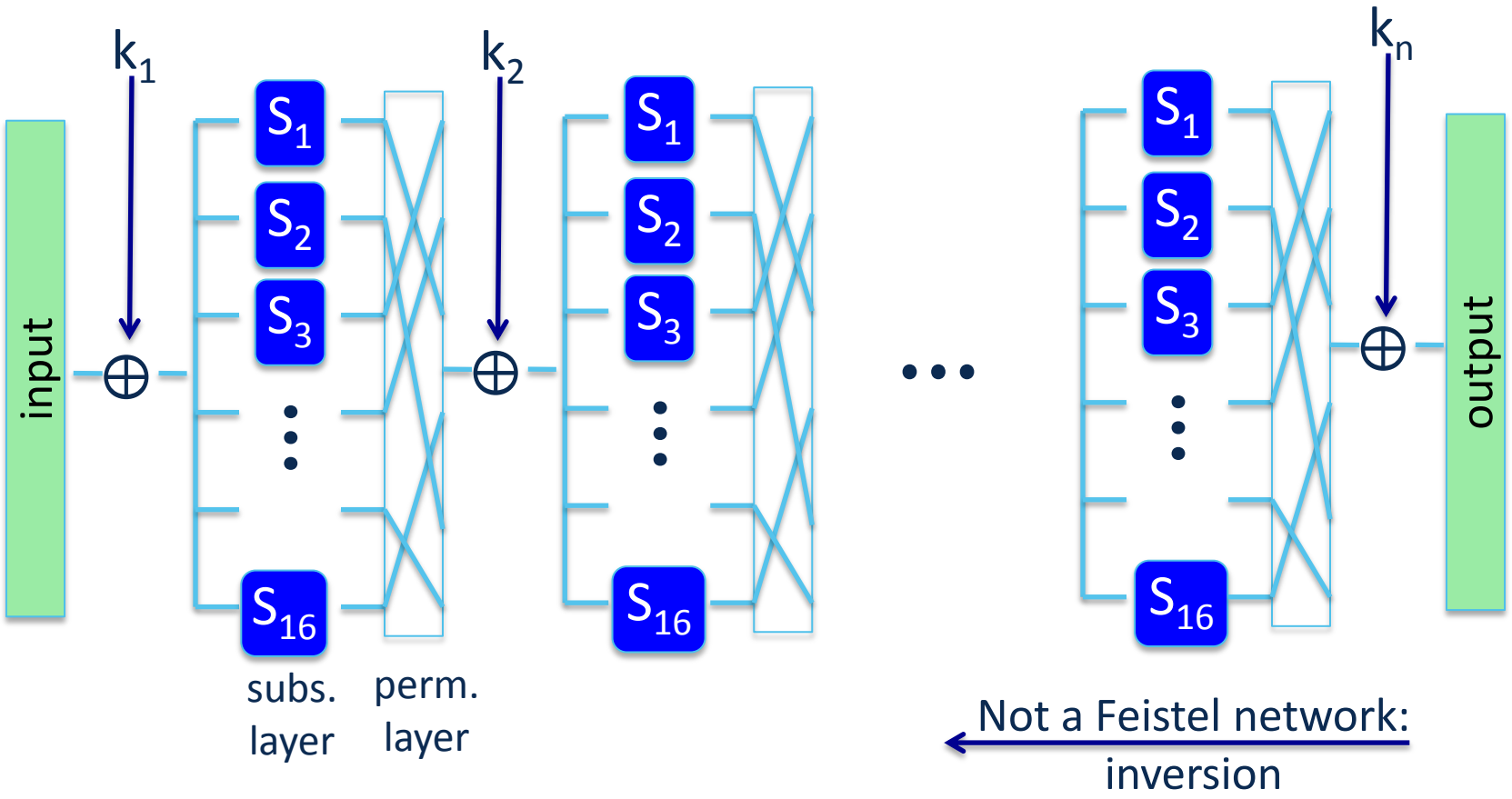
(Mars: IBM, RC6: RSA, Rijndael: Rijmen/Daemen – Belgium, Serpent: Anderson/Biham/Knudsen, Twofish: Bruce Schneier et al.)

2000: NIST chooses Rijndael as AES

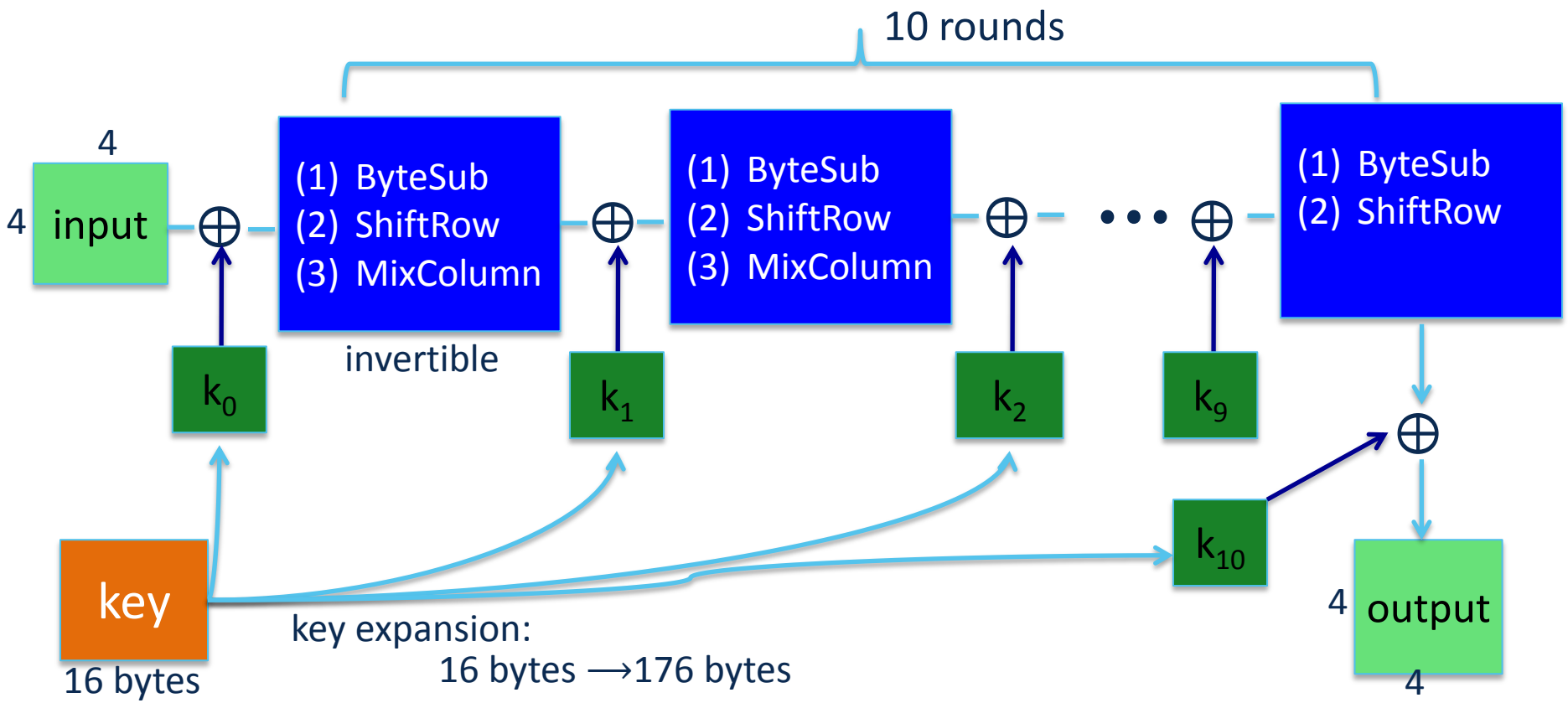
Key sizes: 128, 192, 256 bits Block size: 128 bits

Best known (theoretical) attacks in time $\approx 2^{99}$

AES Substitution-Permutation Network



AES-128 scheme



Building Block Ciphers (Modes of Operation)

So far we have seen PRFs and PRPs (3DES, AES)

Goal:

Build „secure“ encryption from secure PRPs

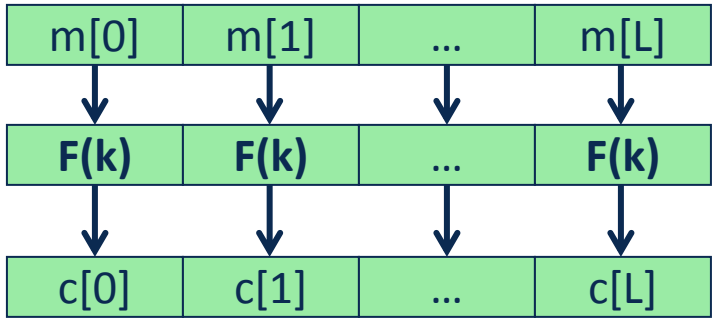
Only one-time keys for the moment:

- Adversary can submit only two messages
- Sees only one ciphertext
- Aims at learning about PT/k from CT (semantic security)

Electronic Code Book Mode

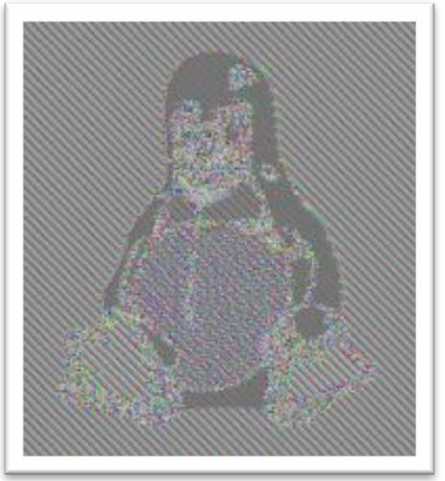
No semantic security!

Encrypt each block with the keyed PRP:



ECB encryption is *deterministic*

⇒ identical PT is encrypted to identical CT:



Is this "secure" (how)?

Consider: Eve can send two messages and needs to distinguish which one has been encrypted…

Deterministic Counter Mode

Semantically secure under one-time-key,
Insecure under many-time-keys

XOR each block with changing keys:

Encrypt key and counter, take a PRF F: K × $\{0,1\}^n \longrightarrow \{0,1\}^n$

$$E_{DETCTR}(k,m) =$$

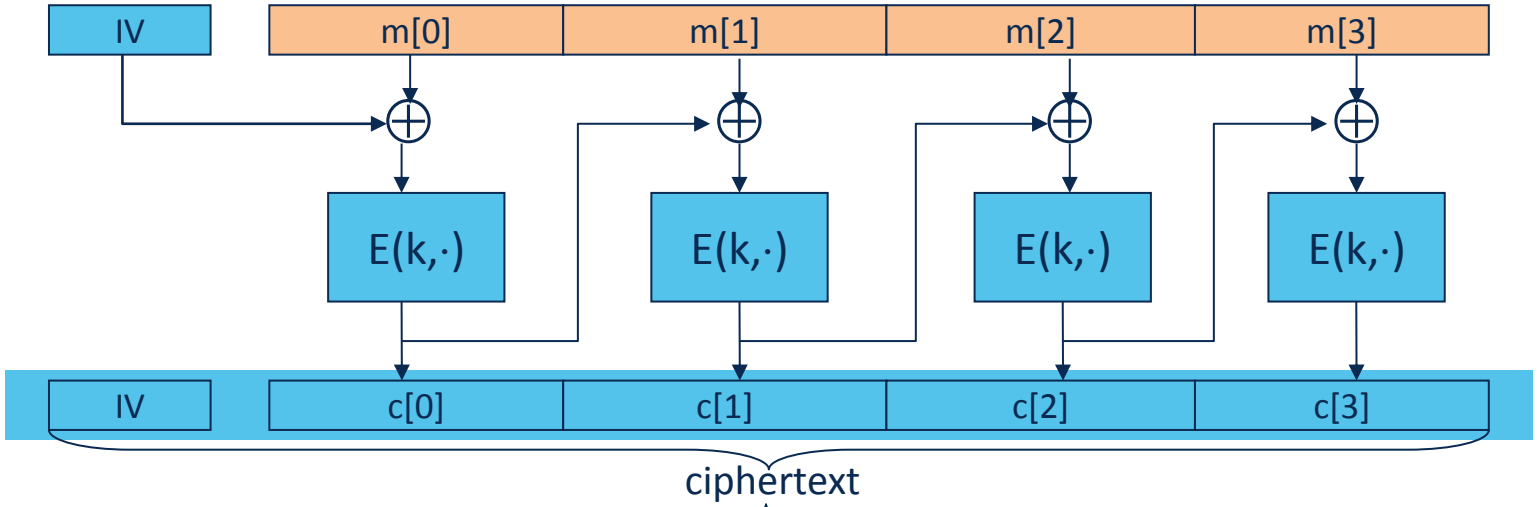
$$D_{DETCTR}(k,m) = ?$$

Remarks:

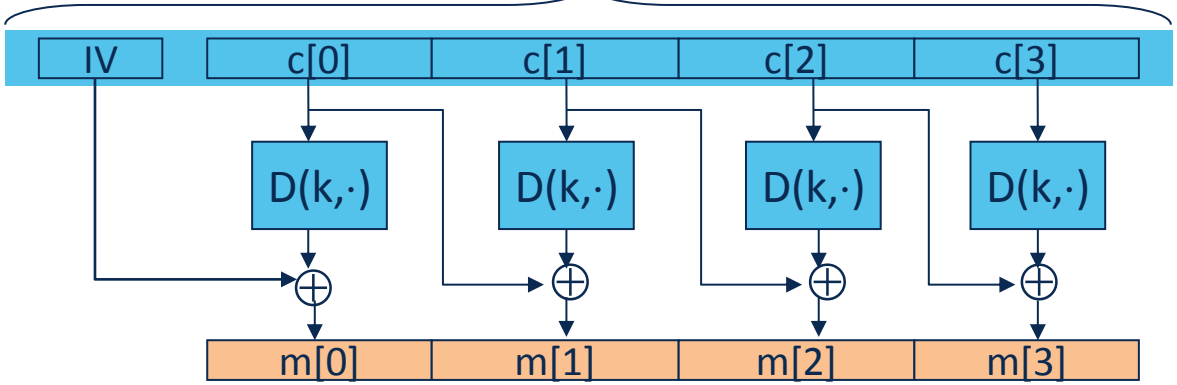
- Assuming each key is used only once: can Eve learn anything?
- Can she learn anything if she can submit several challenges (key is used more than once)?
- Solution: Random/Nonce-based Encryption

CBC with random IV

Let (E,D) be a PRP. $E_{CBC}(k,m)$: choose **random** IV∈X and do:



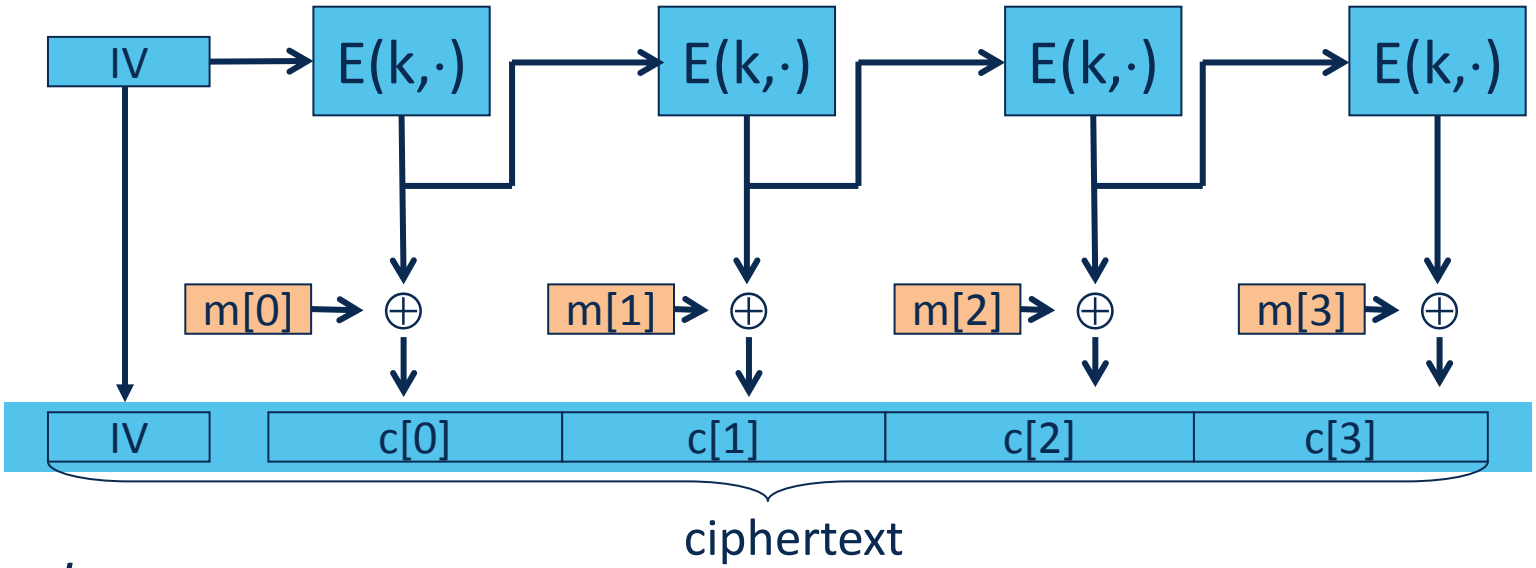
Decrypt?



Dependency between subsequent packets, D can be parallelized

Output Feedback Mode

Let (E,D) be a PRF $E_{CBC}(k,m)$: choose **random** IV∈X and do:



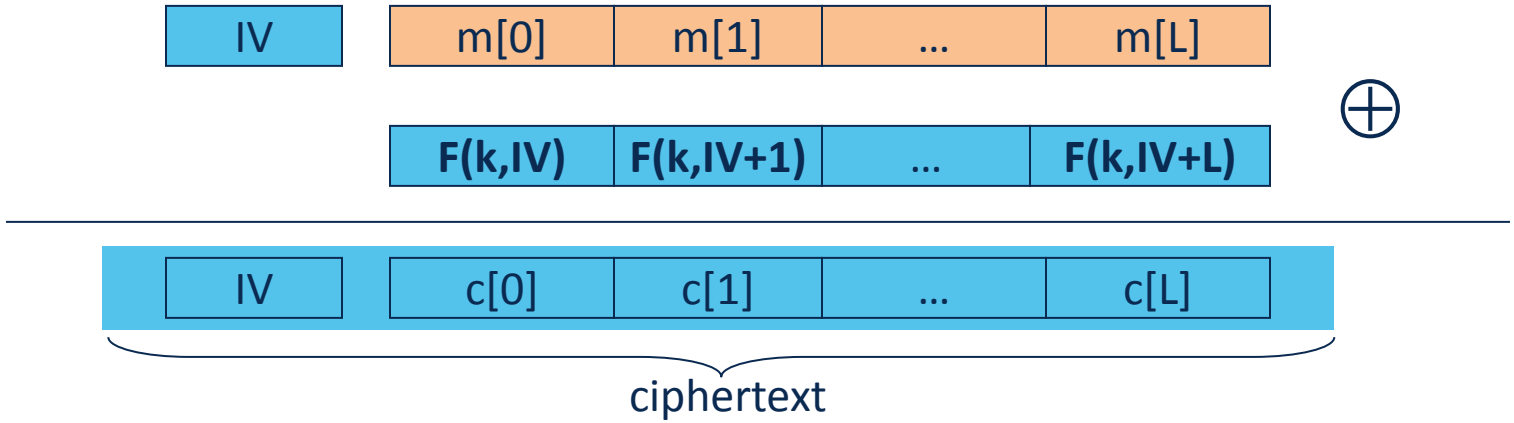
Remarks:

- Dependency between subsequent packets
- E and D cannot be parallelized, ***but***
- E(k,E(k,E(k,…E(k,IV))) can be precomputed

Randomized Counter Mode R-CTR

Let F: $K \times \{0,1\}^n \longrightarrow \{0,1\}^n$ be a secure PRF.

E(k,m): choose a random $IV \in \{0,1\}^n$ and do:



Variation: Choose 128 bit IV as: nonce || counter, to avoid repetition

Remarks:

- E, D can be parallelized and F(k,IV+i) can be precomputed
- R-CTR allows random access, any block can be decrypted on its own
- Again: F can be any PRF, no need to invert

Zusammenfassung

Sie kennen Kryptologie, Kryptographie und Kryptoanalyse

Sie wissen was Strom- und Block-Chiffren sind

Sie verstehen die unterschiedlichen Angreifermodelle

Sie kennen perfekte und semantische Sicherheit und Sie wissen, wie diese nachgewiesen werden

Sie kennen das One-Time-Pad

Sie können Feistel-Netzwerke und 3DES erklären

Sie kennen und verstehen AES

Sie kennen unterschiedliche Operationsmodi und kennen ihre Eigenschaften