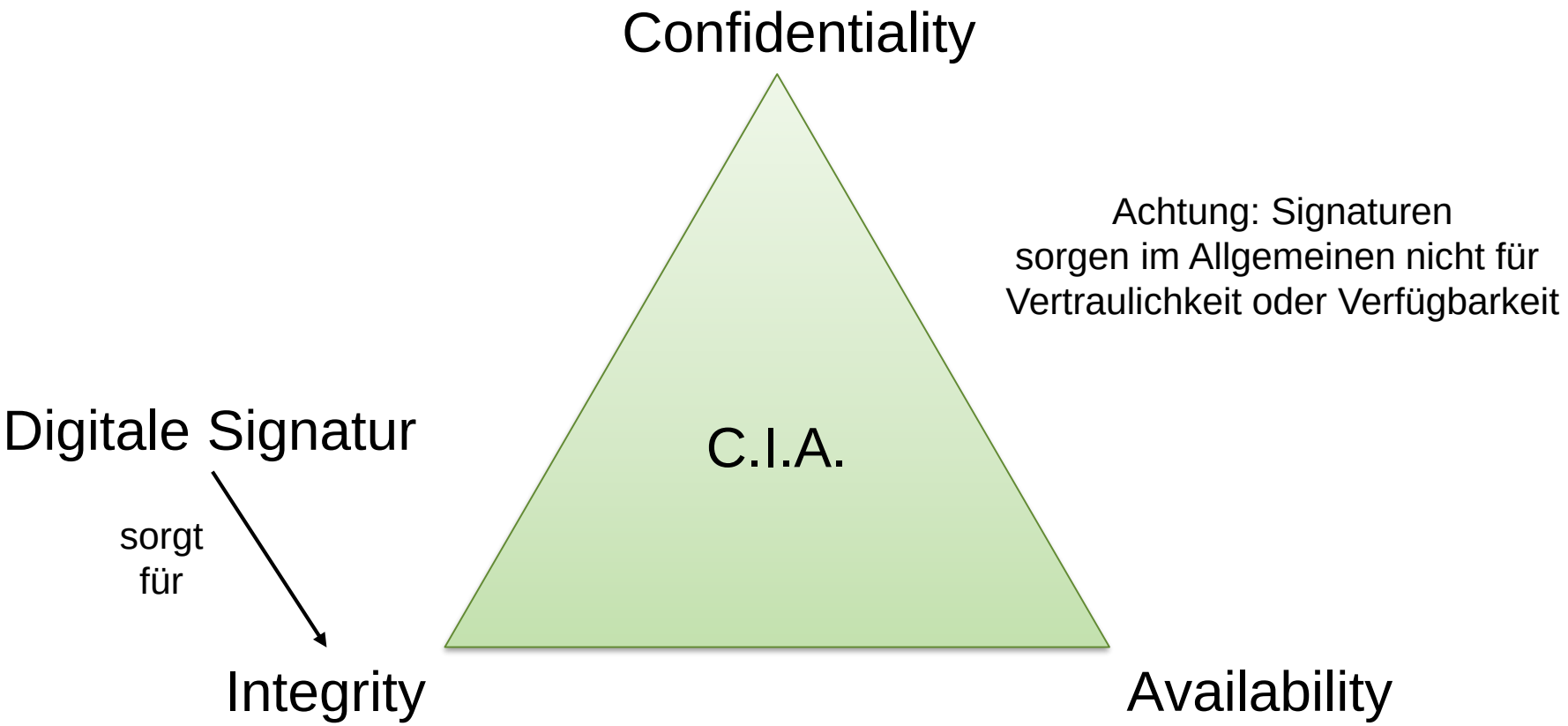
Computersystemsicherheit

Prof. Marc Fischlin, Wintersemester 18/19

03

Digitale Signaturen



Achtung: Signaturen
sorgen im Allgemeinen nicht für
Vertraulichkeit oder Verfügbarkeit

Digitale Signatur

sorgt
für

Anwendungsszenario

verändert Kommunikation

Alice

Eve

Bob

Bobs Public-Key pk

geheimer Schlüssel sk

Nachricht m

Ciphertext C←Enc(pk,m)

C*

Nachricht m*←Dec(k,C*)

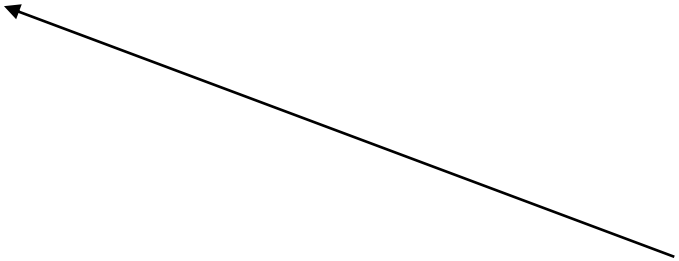
Woher weiß Bob, dass diese Nachricht wirklich von Alice stammt?

Integrität ist vielfältig

Sind eigentlich zwei Fragen:

(1) Woher weiß Bob, dass die Nachricht von Alice stammt („Integrität des Ursprungs“)?

(2) Woher weiß Bob, dass die Nachricht nicht verändert wurde („Integrität der Daten“)?



Woher weiß Bob, dass diese Nachricht wirklich von Alice stammt?

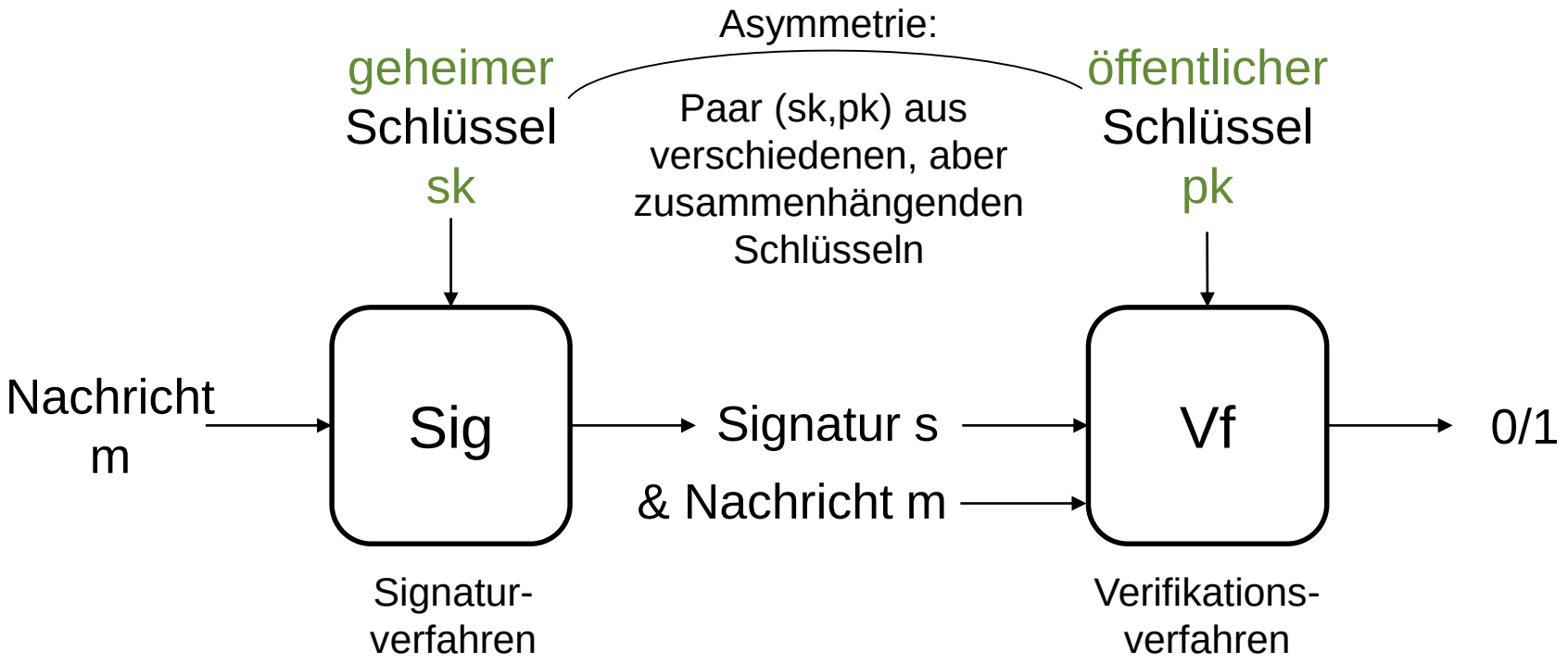
Beispiel

Unterschriftenvergleich beim Einkauf per EC-Karte

Integrität der Daten bereits gewährleistet

Unterschrift soll nur „Integrität des Ursprungs“ garantieren

Prinzip der digitalen Signatur



Funktionale Korrektheit (Vollständigkeit):
Für alle Nachrichten m und alle Paare (sk,pk) gilt: Vf(pk,m,Sig(sk,m))=1

Sicherheit

Angreifer soll keine Signatur
für Nachricht fälschen können.

Kerckhoffs-Prinzip: selbst wenn er alles kennt, außer dem geheimen Signier-Schlüssel

Einfache Folgerung: Digitale Signatur hängt stark von Nachricht ab. Sonst:

Alices geheimer Schlüssel sk

Alices öffentlicher Schlüssel pk

Nachricht m

Signatur s←Sig(sk,m)

m, s →

m*, s →

1 ←Vf(pk,m*,s)

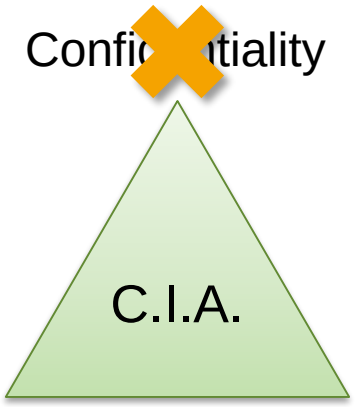
Beispiel

ASCII-“Hülle“ von zwei abgetrennten 4096-Bit-RSA-Signaturen

```
Dies ist ein Text, der
unterschrieben worden ist.

-----BEGIN PGP SIGNATURE-----
Version: GnuPG v2

iQIcBAABCAAGBQJX/3UaAAoJEA2zFmsNrpFKi
YcP/13TZo2TAbrGV+NXbhXLsyj3yEO6p1pK4k
viXPIZtJtou9R30ucjsHyha6lxmRAFbs3i1d7
B6rwIYTY5pV5pX8yJ1R0Q1ZfLLXFmu9R/VXgS
dqd9wbxYRGo3Ii0+e31Z309WKxzU+JBSYXY47
AE5iuVkcCRr3JssvNBz1NwESMFgVexd9GKA6O
nDVgru3eSg1oj+vVvg2+dcpkLLhIfhip57j4b
+eqPTGUpDUlHOpcoRD+hyiVdI4kHmc8bHQ5Kc
/DaEPCbapSV+2Tf7IyYlKl8HKgxA3s9i9gZaJ
54EauUQ3qQvDOSeg2KK6rKZBvBhKBKCfKHvR1
hKm+Yz8N5+Xu6oVjMapmzNtRTqFGQCb1Hlwcn
Xw3OoPdvBtTEOukWPIwNb58/SfBdKshcAnl5N
xmEIfx1oMLRoNTYI+ctymAAFY0wAW9f0R9vUF
/hhtxmIdv70j6D2Jp58JZdqADP9Vy/r1wHmJj
Daz9bU6FxecvlfGeS66pnMTVfJUagPkc3bWnN
nvpMMuonxxVycvarvwNhvHKoKgayPG/eOJ8WX
PSpuAd6u2yJmbhMBBXjYC51IgNyXdIPadf2EO
/rur6yIG0zz8NfucYxdgxaylxDuIhmP1e5kK1
iGJO3vUNqhiHFu3iX+bvZe1OC46XnJu/yAS2q
qgRiNpbX+NLgqd9oWmnTy=t8bg
-----END PGP SIGNATURE-----
```

```
Dies ist ein Test, der
unterschrieben worden ist.

-----BEGIN PGP SIGNATURE-----
Version: GnuPG v2

iQIcBAABCAAGBQJX/3XsAAoJEA2zFmsNrpFKu
04P/3v6zwEW1AoybdgFrPrRW1AMzkKXb7Pdr8
C7aPPfUQuiHMsxdKIUmOqxIXNf4HOdZ11h/xv
N0fsD3udSBkTYpaS2Ne6KygzQGeKTuj/IS+c1
lv0vQYfCzlZvMEO6xxdOtEebLLM5YqIkheQy6
G0OC4Ha94BVtbakJNY8wXpxxSbMRR110hnOUD
+kozrQfgU/RfqTHwXccJhNZOkVqX6e0UjWjot
aCymIdq/SWj4v0qHOAesTe9MI6WH8aeRkIv1R
eT66ENwS894RhnIroCBnjbQ1BxweGBwMXC+mx
OdxTVOljtElTksIk7kWpAzi053lAi7CXrvQvD
FvnWL5QsztenD4JRsgvyPeTWF3w67gw6DfuIz
oZCMICagftO33BdhhMvlc52sAc0l0NT+O9bCM
qNmBObPRJ6DvwwzBN+lD5mof+Evj0Md9xtsRI
Vx9l7w2RVvQ17/W9wfyd/FYEPwaYzKTKDr26/
Vs/7aekf9FzO8ZckVRTXcibqW9i7k1VEKSCi2
Bow4QC99Mw2XnZkLhWRXXfjhfErHGIp3833wR
rRFwHdhgsbUP5ybN8H8PFLN3FaQoeBFRxfvlV
VngmormQcCam6Saq8Dv61xmSQaKWgfznF5TKP
6ddxNUWkcWujYaVttqQfEMFQmKR0wn5tK+8bh
eBHenpwS08JSUVm17RDxO=xp98
-----END PGP SIGNATURE-----
```

Integrität vs. Vertraulichkeit



Wieviel Information
über die Nachricht m enthält die Signatur s?

Im schlimmsten Fall alles!

Beispiel:
Sig(sk,m) hängt m (ganz oder komprimiert) an Signatur s an

Macht beispielsweise GPG, wenn per **`gpg -s`** signiert wird

Moderne Signaturverfahren

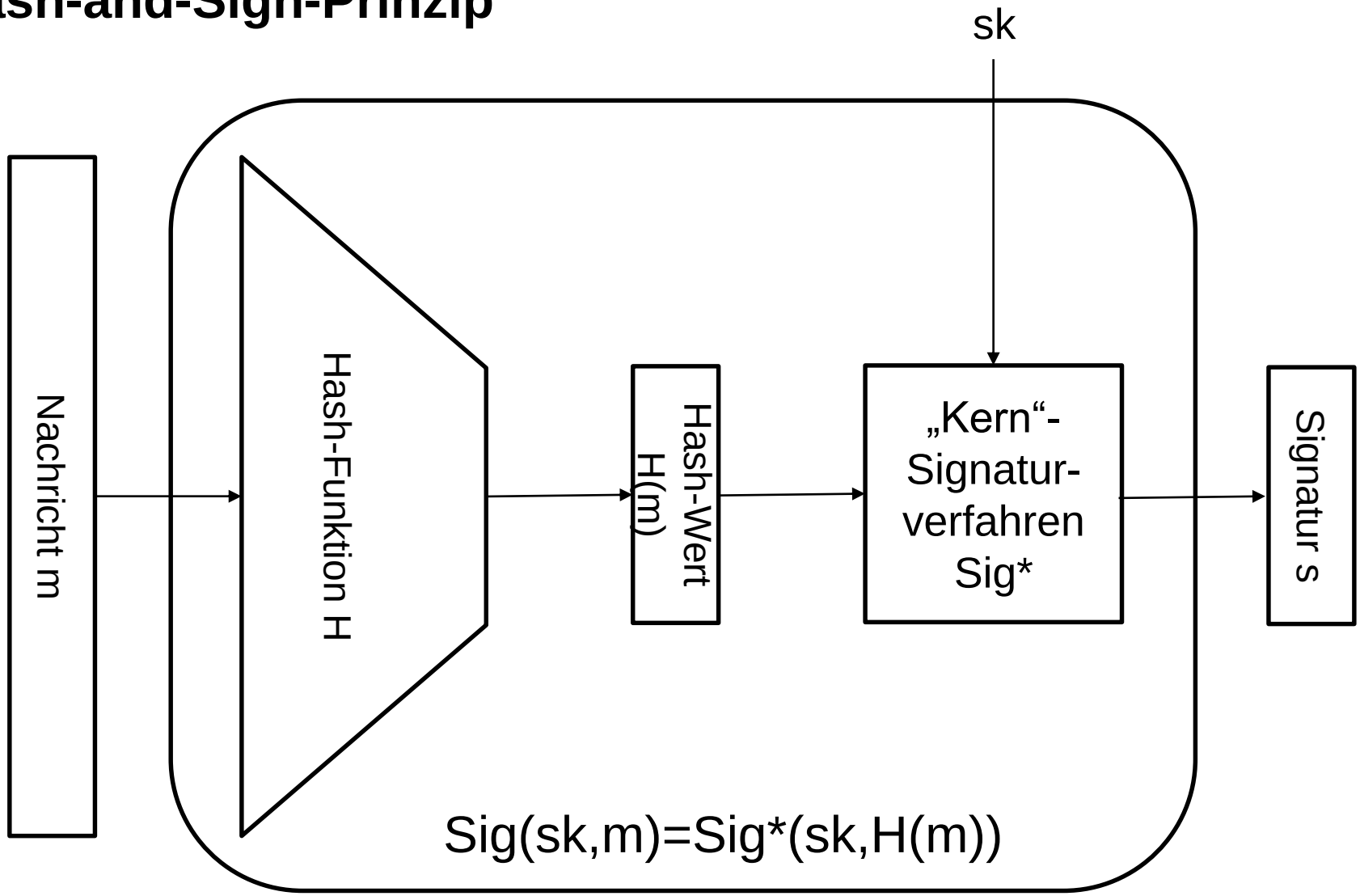
RSA-basierte Signaturen

Digital Signature Algorithm (DSA)

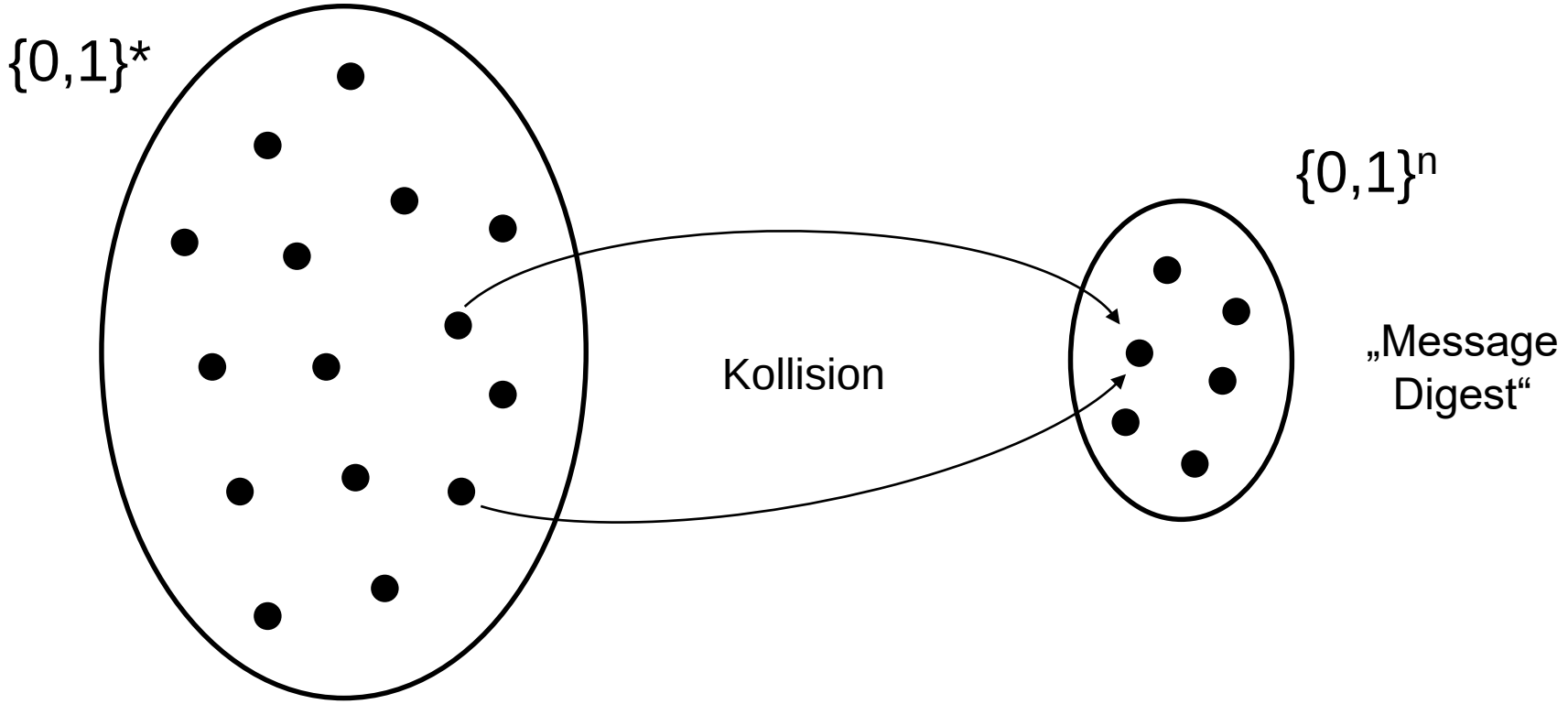
Diskreter-Logarithmus-basiert

Beide Verfahren folgen dem sogenannten „Hash-and-Sign“-Prinzip

Hash-and-Sign-Prinzip

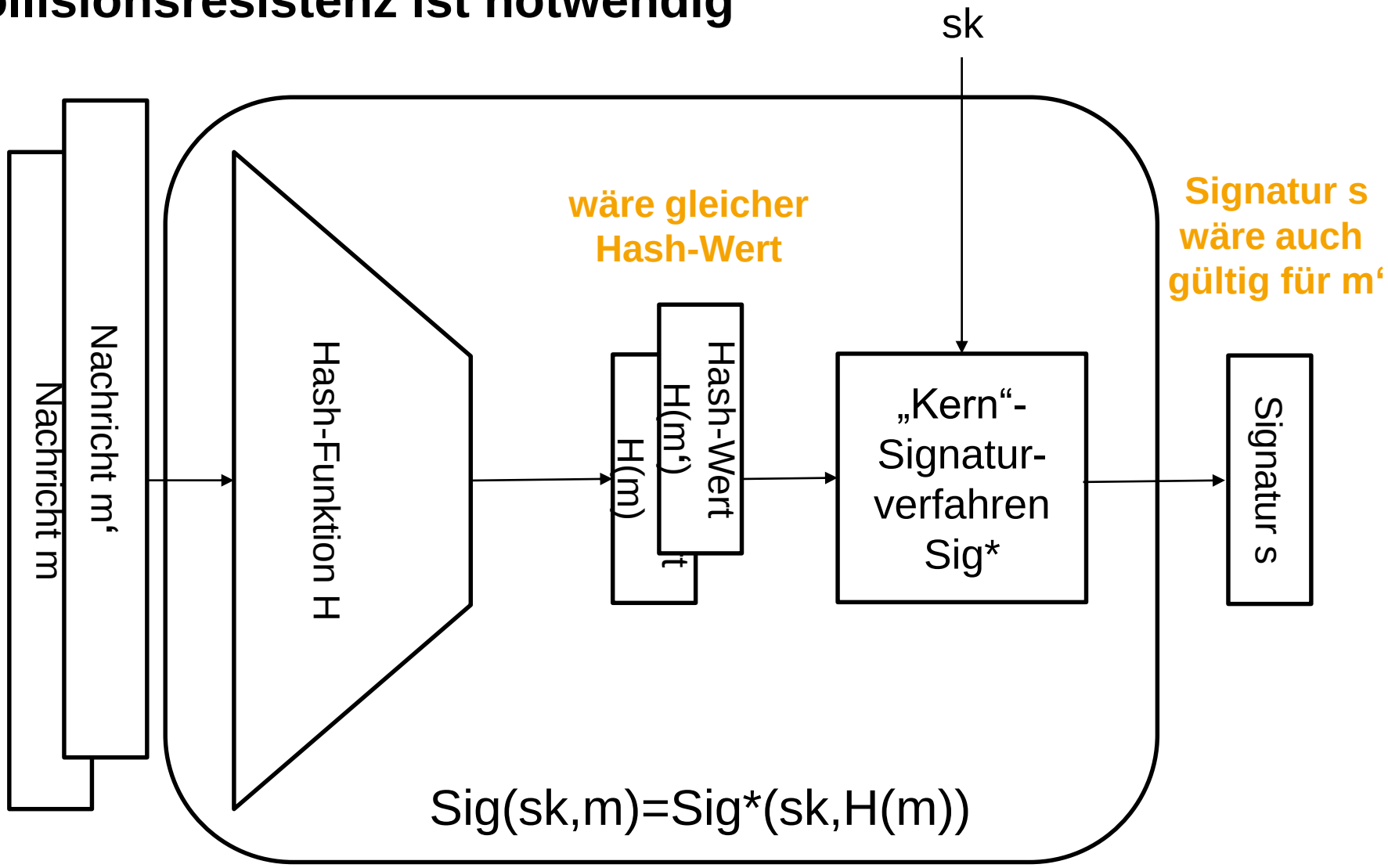


Hash-Funktion $H: \{0,1\}^* \rightarrow \{0,1\}^n$ üblich: n=160, 256, 384, 512



Kollisionsresistenz:
Nachrichten m≠m‘ mit H(m)=H(m‘) existieren
zwar notwendigerweise, sind aber schwierig zu finden

Kollisionsresistenz ist notwendig



Kandidaten in der Praxis

SHA=Secure Hash Algorithm

SHA-1: $\{0,1\}^{2^{64}-1} \rightarrow \{0,1\}^{160}$

1995 von der NIST standardisiert;
inzwischen nicht mehr verwenden wegen SHAttered-Angriffen

SHA-2: $\{0,1\}^{2^{128}-1} \rightarrow \{0,1\}^{n}$

2005 von der NIST für n=224, 256, 384, und 512 standardisiert;
für Übergangsphase bis SHA-3 fertig

SHA-3: $\{0,1\}^{*} \rightarrow \{0,1\}^{n}$

2015 von der NIST für n=224, 256, 384, und 512 standardisiert;
2012 durch öffentlichem Wettbewerb bestimmt (Sieger Keccak)

MD5 nicht mehr verwenden für Kollisionsresistenz!!!

Welche Hash-Funktionen von MD5, SHA-1, SHA-2 und SHA-3 können Sie noch als kollisionsresistent ansehen?

Ist folgende Hashfunktion $H(m1,m2)=m1 \oplus m2$ für $m1,m2 \in \{0,1\}^{256}$ kollisionsresistent?

Was ist mit folgender Idee? Da H(m) quasi eindeutig ist, kann man H(m) als Signatur zur Nachricht m benutzen.

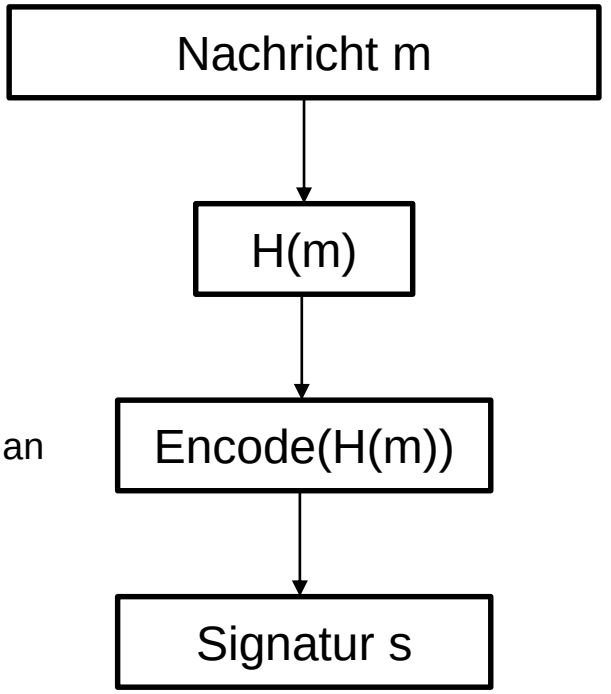
RSA-Signaturen sk=(N,d)

1. Hashe Nachricht m auf H(m)

2. Kodiere „kurzen“ Hashwert auf RSA-Länge

   sichere Optionen für Kodierungen gibt z.B. das BSI an

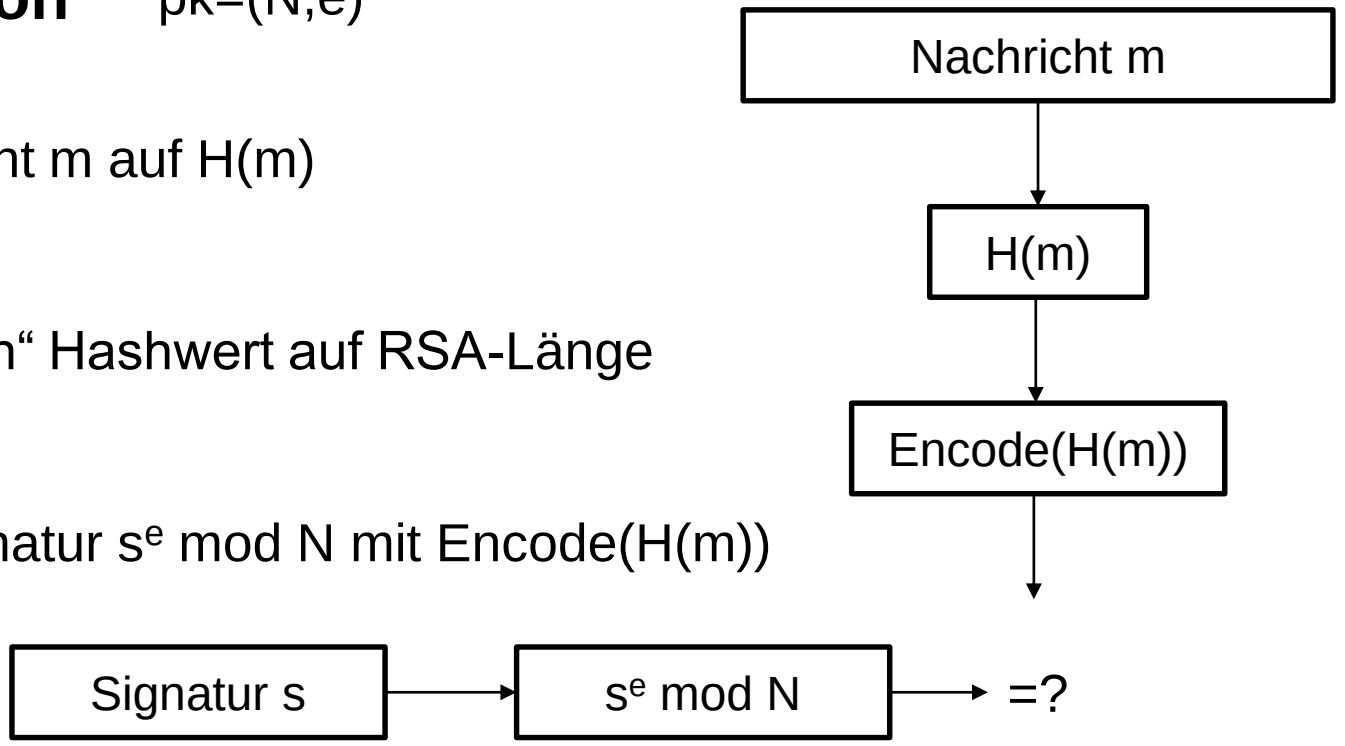
3. Wende RSA-Schlüssel $(\bullet)^d \bmod N$ an



$$s = (\text{Encode}(H(m))^d \bmod N$$

RSA-Verifikation pk=(N,e)

1. Hashe Nachricht m auf H(m)
2. Kodiere „kurzen“ Hashwert auf RSA-Länge
3. Vergleiche Signatur s^e mod N mit Encode(H(m))



Funktional korrekt, da für korrekte Signatur $s=(Encode(H(m))^d \text{ mod } N$ gilt:

$$s^e = (Encode(H(m))^d)^e = Encode(H(m)) \text{ mod } N$$

DSA-Signaturen

Unterschrift zu Nachricht m durch Schlüssel sk=(g,x,p,q):

1. Wähle zufälliges $k \leftarrow \{1,2,\ldots q-1\}$
2. Berechne $r = (g^k \bmod p) \bmod q$
3. Berechne $s = k^{-1} \bullet (H(m) + xr) \bmod q$
4. Gib aus S=(r,s)

mod-Operator!

$g \in \{1,2,\ldots p-1\}$
Generator primer Ordnung q,
d.h. $g, g^2, g^3, \ldots, g^{q-1} \neq 1$ und $g^q = 1$

k^{-1} Inverses zu k mod q,
d.h. $k \bullet k^{-1} = 1 \bmod q$

von der NIST 1991 entwickelt und
1994 als DSS (Digital Signature Standard) standardisiert

DSA-Verifikation

Prüfe Unterschrift S=(r,s) zu Nachricht m mit Schlüssel pk=(y,p,q,g) mit $y=g^x$:

1. Berechne $v = H(m) \cdot s^{-1} \bmod q$
2. Berechne $w = r \cdot s^{-1} \bmod q$
3. Prüfe, dass $(g^v \cdot y^w \bmod p) = r \bmod q$?

mod-Operator!

Zur Erinnerung:

Funktionale Korrektheit: $r = (g^k \bmod p) \bmod q$ und $s = k^{-1} \cdot (H(m) + xr) \bmod q$

Für korrekt gebildete Signatur gilt:

$$g^v \cdot y^w = g^{H(m) \cdot s^{-1}} \cdot y^{r \cdot s^{-1}} = g^{s^{-1} \cdot (H(m)+xr)} = g^k \bmod p$$

und damit auch Gleichheit mod q

Playstation-3-Angriff 2010

Quelle: Wikipedia

Ziel: Kontrolle erlangen/
Code aufspielen

akzeptiert aber nur mit
DSA-signierten Code
unter öffentlichem
Sony-Schlüssel

Sony unterschreibt auch Code per DSA,
aber immer mit gleichem Zufallswert

DSA-Signaturen mit schwachem Zufall

Quelle: Wikipedia

Ziel: Kontrolle erlangen/
Code aufspielen

Gegeben zwei DSA-Signaturen (r_1,s_1) und (r_2,s_2) für verschiedene Nachrichten $m_1 \neq m_2$ mit **gleichem** Zufallswert k:

$r_1 = (g^k \bmod p) \bmod q$ und $s_1 = k^{-1} \cdot (H(m_1) + x r_1) \bmod q$

$r_2 = (g^k \bmod p) \bmod q$ und $s_2 = k^{-1} \cdot (H(m_2) + x r_2) \bmod q$

$r_1 = r_2$

Berechne $s_1 - s_2 = k^{-1} \cdot (H(m_1) + x r_1) - k^{-1} \cdot (H(m_2) + x r_2)$

$= k^{-1} \cdot ((H(m_1) + x r_1) - (H(m_2) + x r_2))$

$= k^{-1} \cdot (H(m_1) - H(m_2)) \bmod q$

ergibt

$k = (s_1 - s_2)^{-1} \cdot ((H(m_1) - H(m_2)) \bmod q$

ergibt

$x = (k \cdot s_1 - H(m_1)) \cdot r_1^{-1} \bmod q$

Kontrollübergabe

Quelle: Wikipedia

Ziel: Kontrolle erlangen/
Code aufspielen

unterschreibt eigenen Code

Angreifer kennt Sonys geheimen Signaturschlüssel

$$x = (k \cdot s_1 - H(m_1)) \cdot r_1^{-1} \bmod q$$

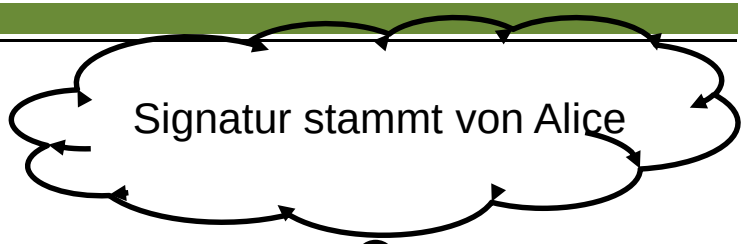
Wie funktioniert eine RSA-Signatur?

Diskutieren Sie folgende Aussage:
„Signaturen sind die Umkehrung von Verschlüsselung.“

Diskutieren Sie folgende Aussage:
„Jedes Signatur-Verfahren ist vom Typ Hash-and-Sign.“

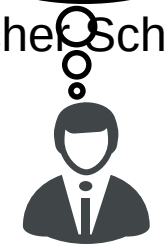
Zertifikate

Wie den Schlüssel zuordnen?



Schlüsselpaar (sk,pk)

Öffentlicher Schlüssel pk



Nachricht m

Signatur s←Sig(sk,m)

m, s

1 ←Vf(pk,m,s)

Verbindung zu Alice?

Tatsächlich aber lediglich:
Signatur stammt von der Person,
die unter dem Schlüssel pk signieren kann.

1.Möglichkeit: Direkte Bestätigung

Schlüsselpaar (sk,pk)

Öffentlicher Schlüssel pk

m, s

oder

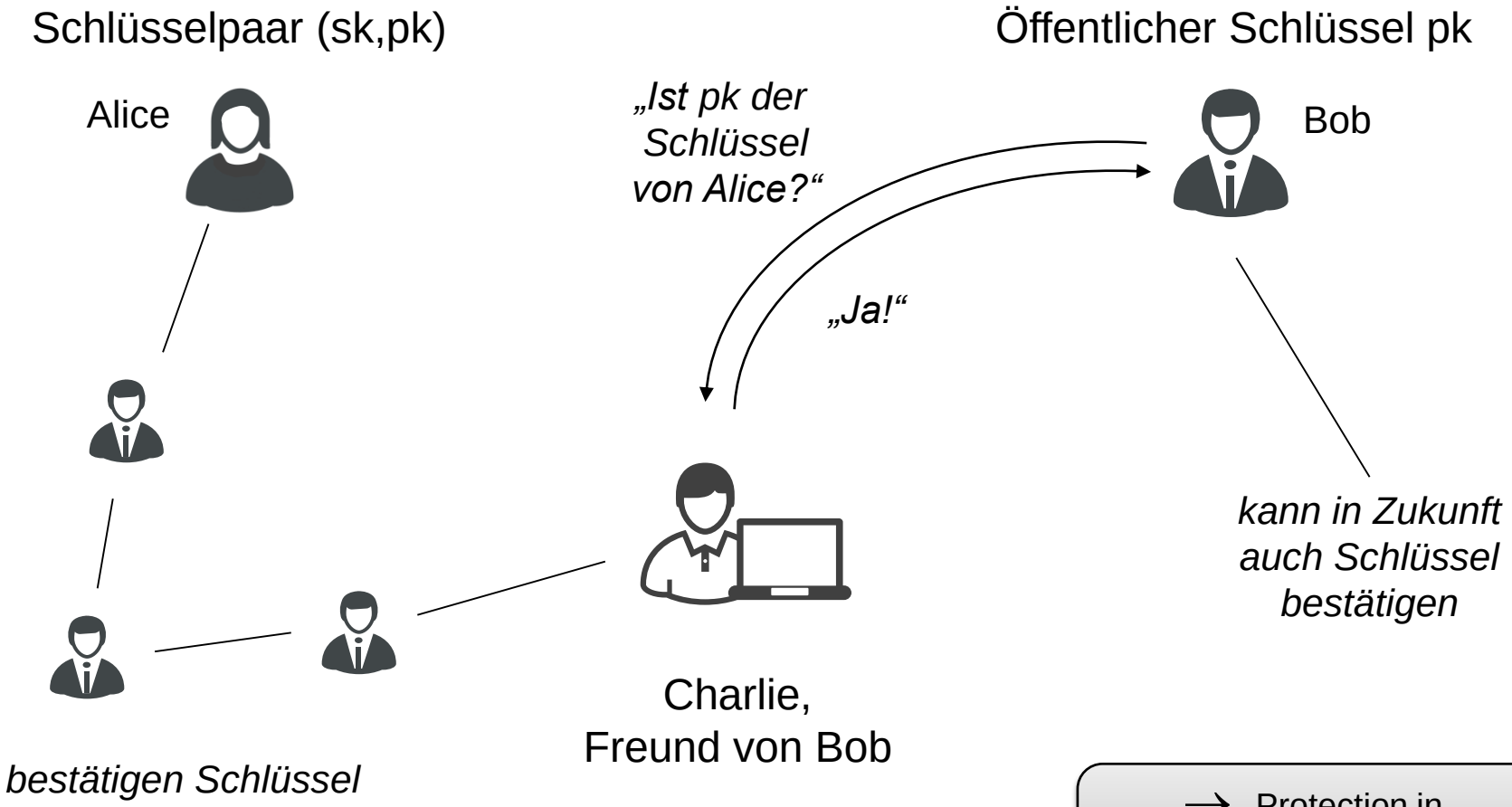
bestätige Schlüssel
über zusätzlichen
Kommunikationskanal

leichtere
Verifikation für Menschen
durch Hashwert H(pk)
~20-28 Zeichen

„Fingerprint“

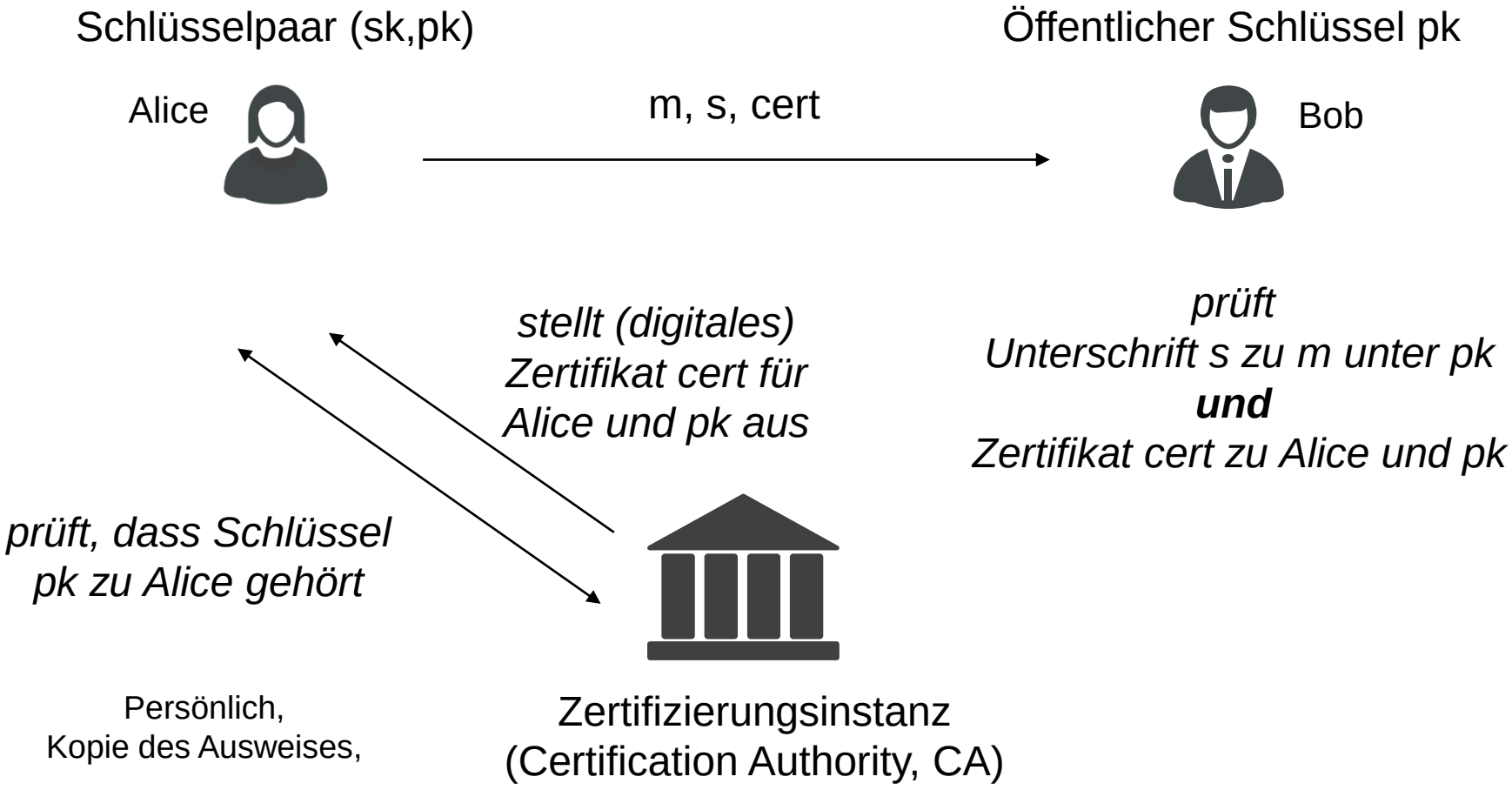
2.Möglichkeit: „Web of Trust“

GPG



→ „Protection in Networked Systems – Trust, Resilience, and Privacy“

3.Möglichkeit: Zertifikate



Zertifikate

Was steht in solchen Zertifikaten?

Wie erhält man eigentlich die Schlüssel und Zertifikate?

Wie prüft man elektronische Zertifikate?

Zertifikatsinhalt

heute: X.509-Zertifikate oder
„kürzere“ Card Verifiable Certificates (CVCs) für Smartcards

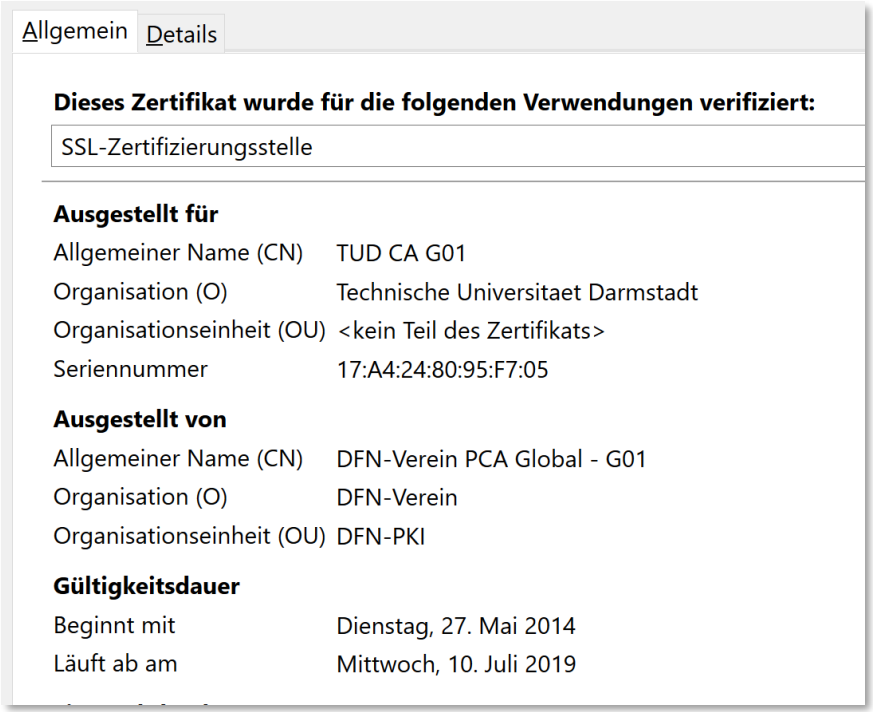
Inhaber üblicherweise common name (**CN**), Organisation (**O**), Land (**C**), …

Aussteller wie Inhaber

Seriennummer

Gültigkeitsdauer

…



Schlüssel- und Zertifikatsverteilung

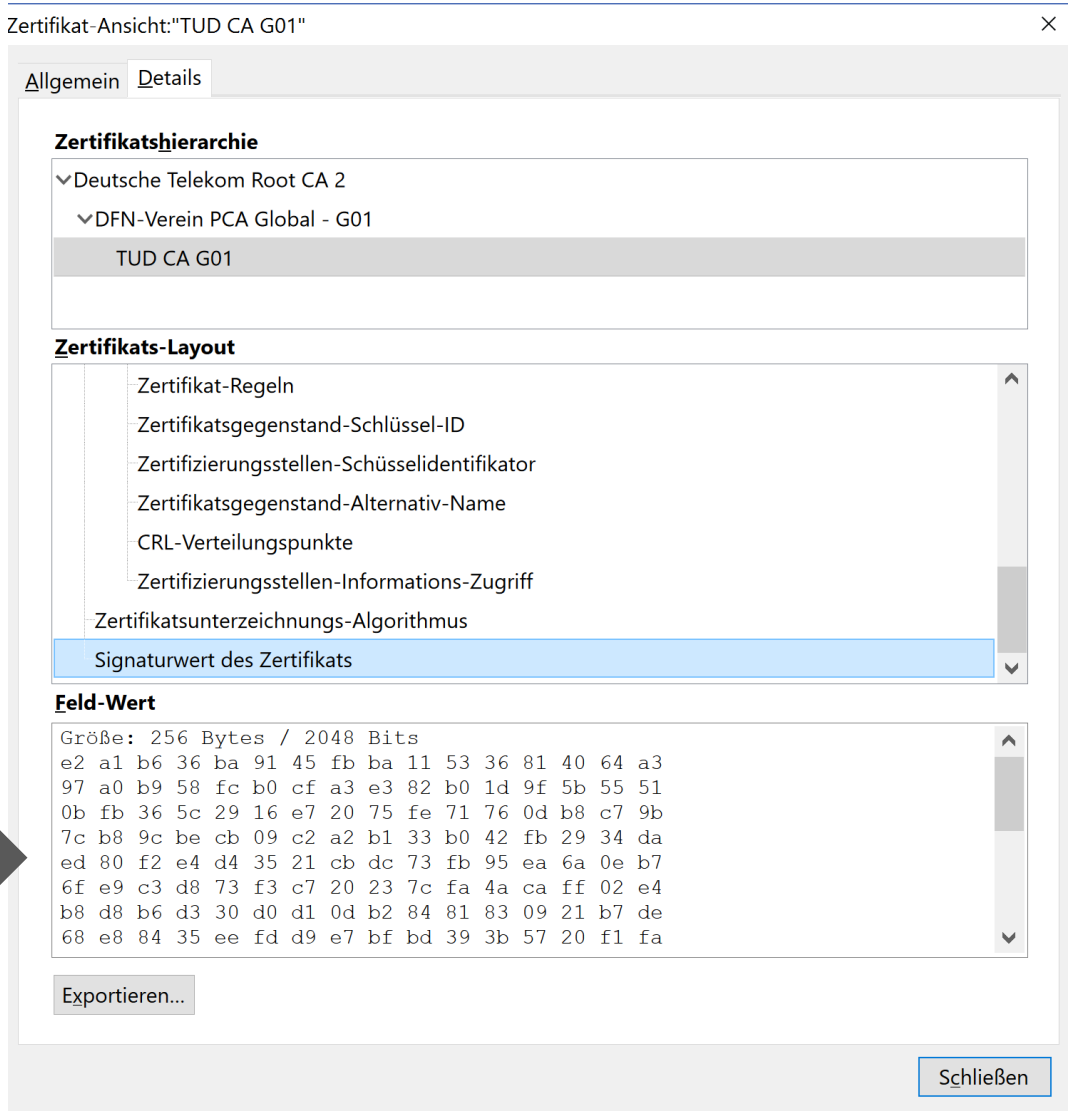
Öffentliches Verzeichnis

z.B. lokaler LDAP Server
oder
globaler Key Server

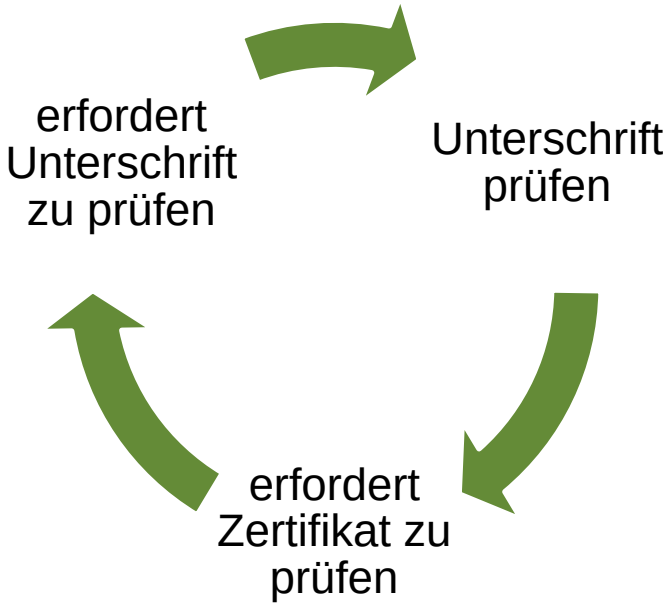
z.B. SKS-Keyservers für OpenPGP aktuell ca. 4,5 Mio Schlüssel

Zertifikatsprüfung

Ansatz:
Certification Authority unterschreibt Inhalt mit digitaler Signatur

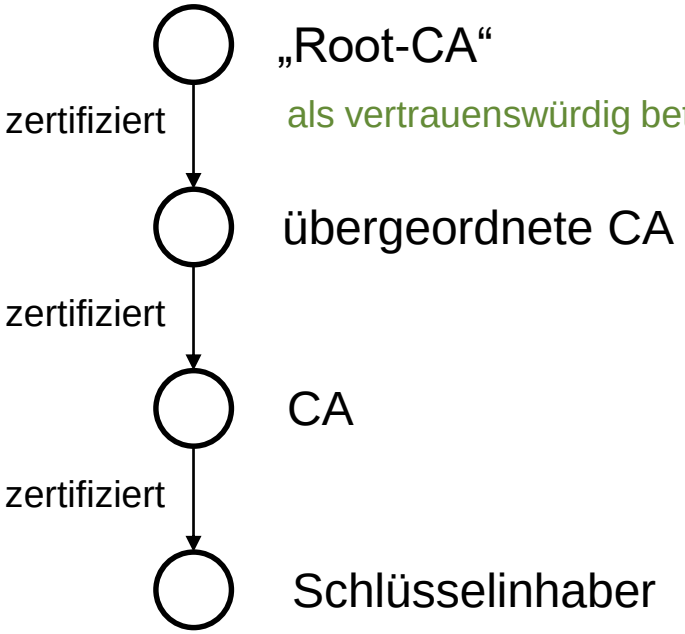


Endlosrekursion?



„Teufelskreis“ muss man aufbrechen, indem man einem öffentlichen Schlüssel einer Zertifizierungsinstanz vertraut

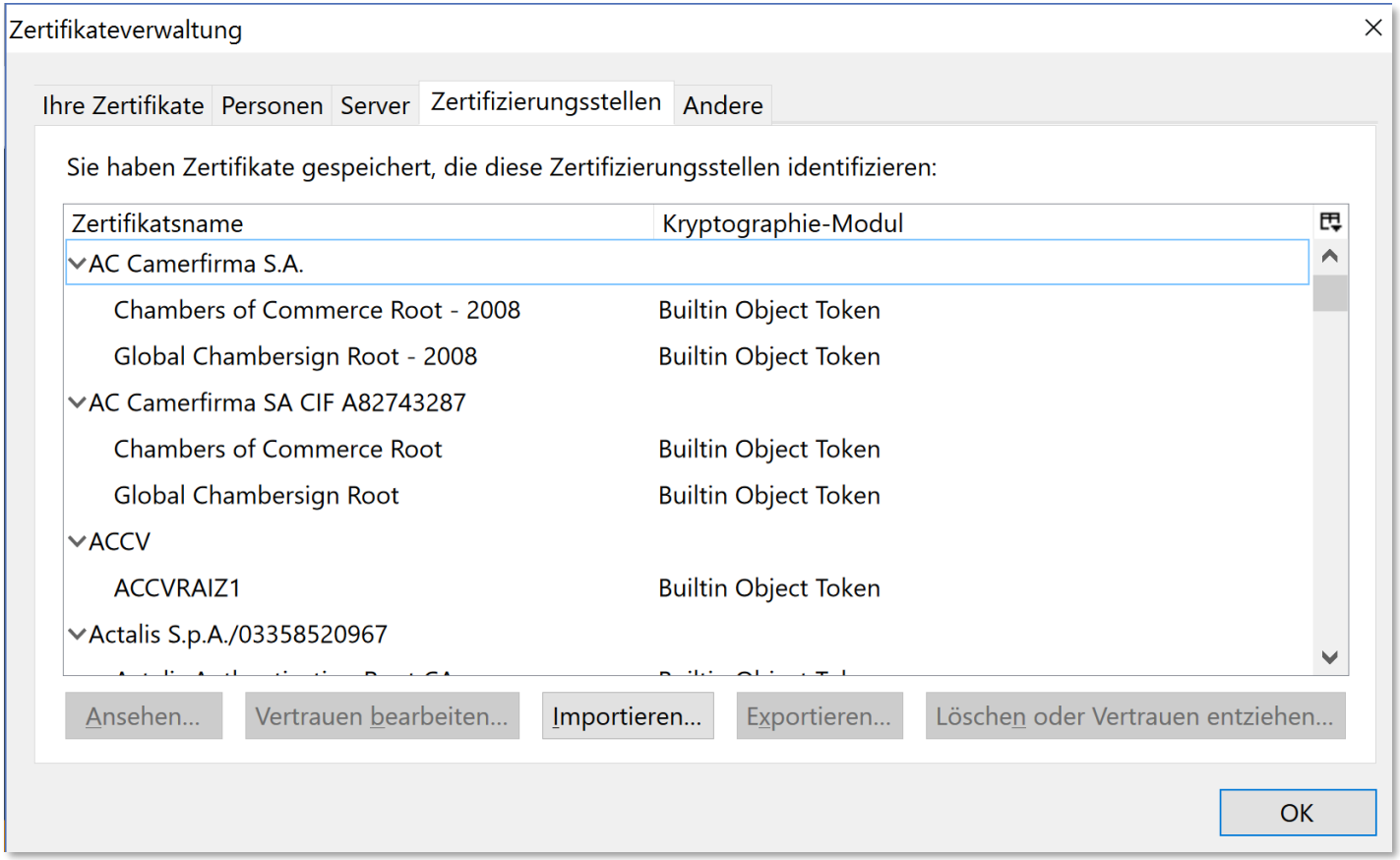
Zertifizierungshierarchie



ergibt Zertifikatskette

Vertrauenswürdige CAs

z.B. vorinstalliert im Web-Browser



Diginotar-Vorfall 2011

SPIEGEL ONLINE DER SPIEGEL SPIEGEL TV Anmelden

NETZWELT Schlagzeilen | Wetter | DAX 10.609,48 | TV-Programm | Abo

Nachrichten > Netzwelt > Web > Internet > Diginotar-Hack: Attacke auf das Sicherheitssystem des Web

Diginotar-Hack

Attacke auf das Sicherheitssystem des Web

Unbekannte Hacker bringen die Sicherheitsmechanismen des Internet ins Wanken: Sie konnten sich selbst mehr als 500 Web-Ausweise ausstellen und sich mit diesen erbeuteten Zertifikaten als Google, CIA oder Facebook ausgeben. Hinter dem Angriff wird Iran vermutet.

Von Matthias Kremp

Corbis

Netzwerkzentrale der CIA: Auch Geheimdienste sind von dem Hack betroffen

Hacker in Zertifizierungsinstanz „Diginotar“ eingebrochen

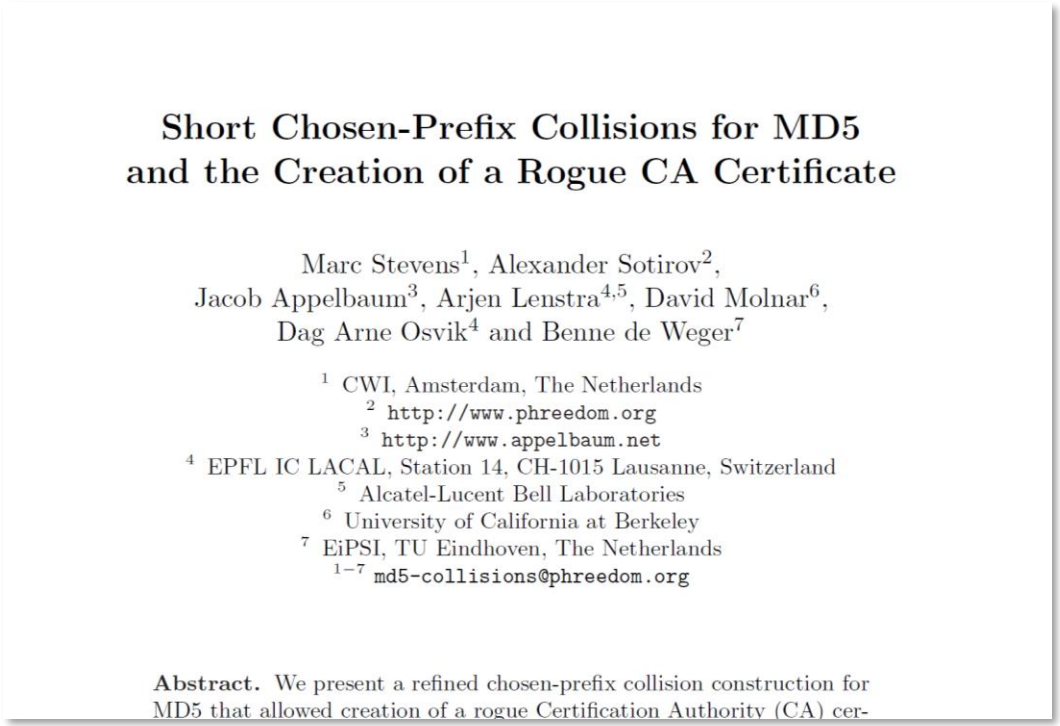
Zertifikate erstellt u.a. für

`google.com`

`microsoft.com`

`skype.com`

MD5-Zertifikatskollisionen 2009



Short Chosen-Prefix Collisions for MD5
and the Creation of a Rogue CA Certificate

Marc Stevens[1], Alexander Sotirov[2],
Jacob Appelbaum[3], Arjen Lenstra[4,5], David Molnar[6],
Dag Arne Osvik[4] and Benne de Weger[7]

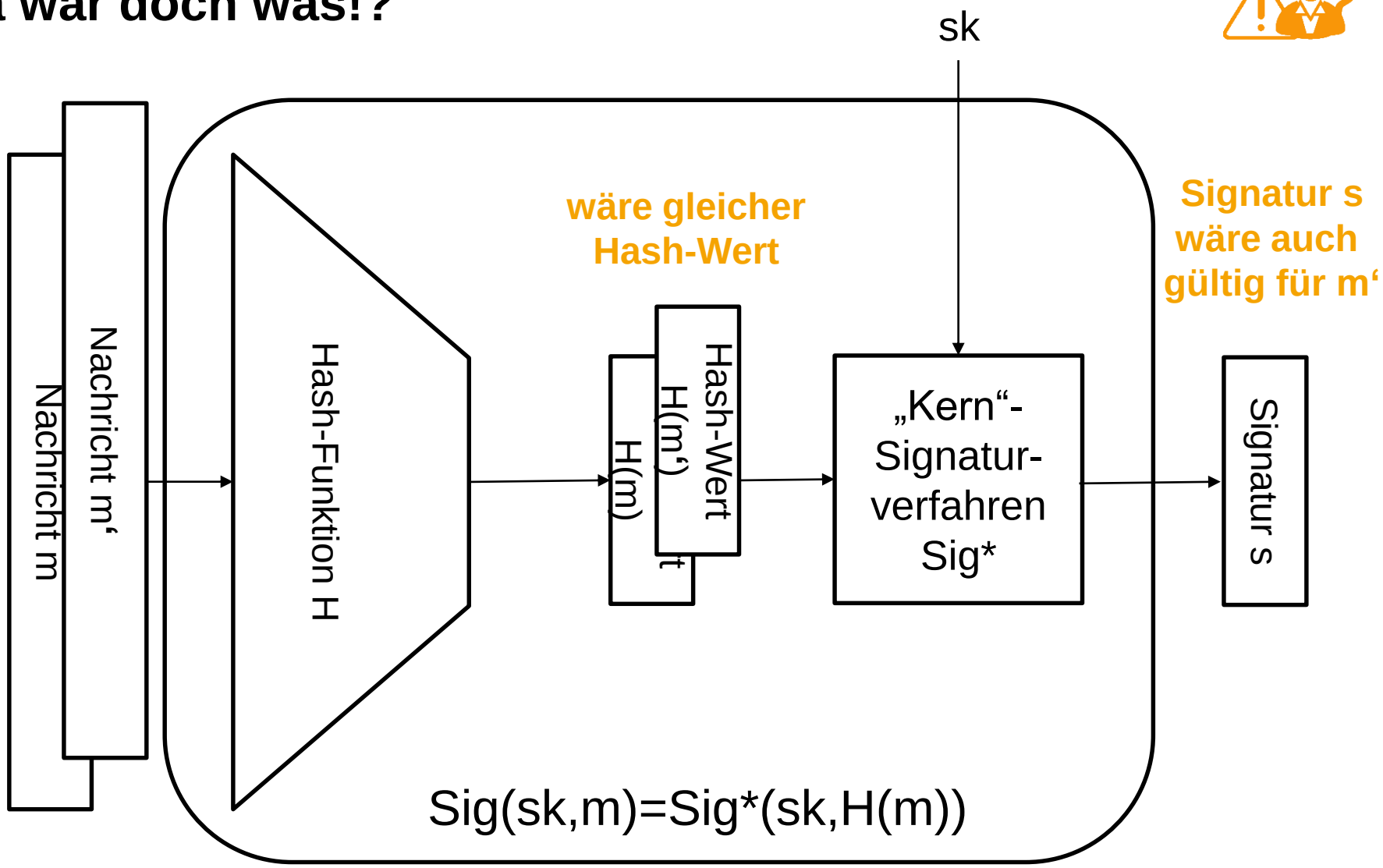
[1] CWI, Amsterdam, The Netherlands
[2] http://www.phreedom.org
[3] http://www.appelbaum.net
[4] EPFL IC LACAL, Station 14, CH-1015 Lausanne, Switzerland
[5] Alcatel-Lucent Bell Laboratories
[6] University of California at Berkeley
[7] EiPSI, TU Eindhoven, The Netherlands
[1–7] md5-collisions@phreedom.org

Abstract. We present a refined chosen-prefix collision construction for MD5 that allowed creation of a rogue Certification Authority (CA) cer-

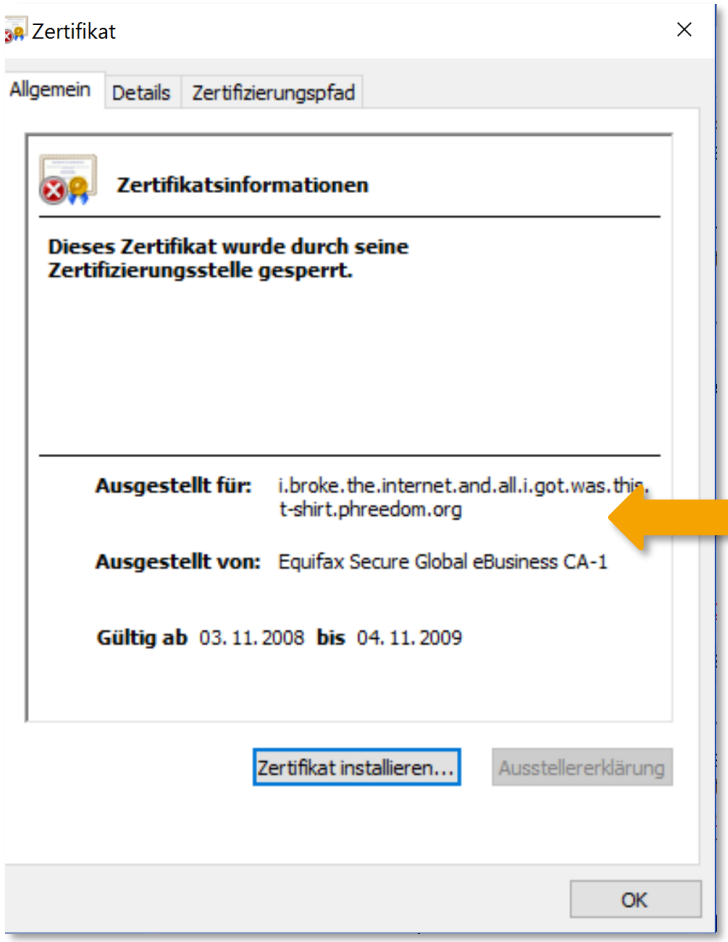
Angriff auf CAs,
die noch schwache
MD5-Hashfunktion für
Zertifikatsunterschriften
verwendet

MD5 galt seit 2005 als nicht mehr kollisionsresistent,
wurde aber noch 2009 von Zertifizierungsinstanzen
zum Signieren unter **`md5RSA`** genutzt,
da man trotzdem keine speziellen Kollisionen
für Zertifikate und ihre Struktur erwartete

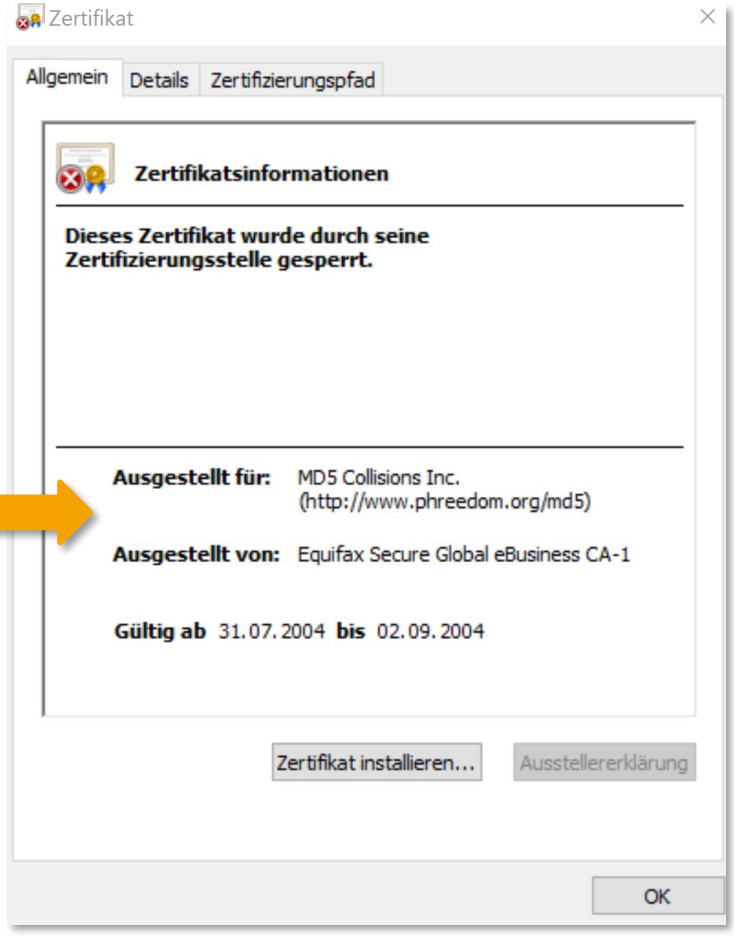
Da war doch was!?



MD5-Zertifikatskollisionen



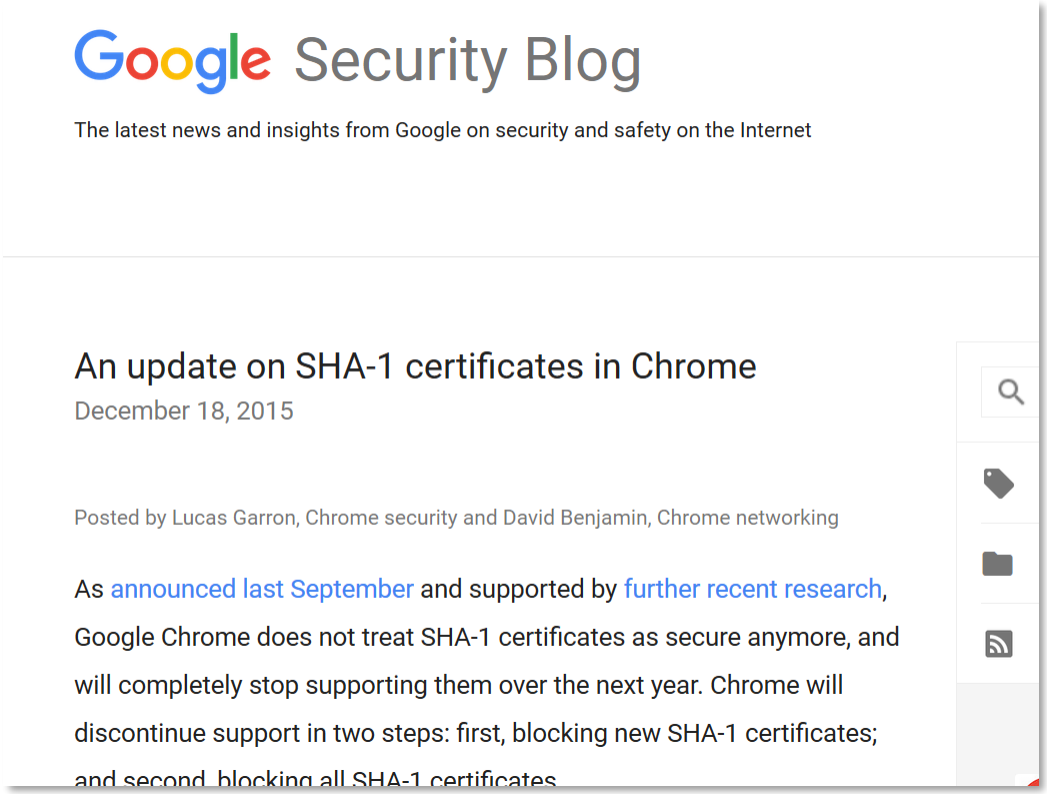
verschiedene Inhaber und Schlüssel



Zertifikat, von einer echten CA ausgestellt

Zertifikat mit gleichem MD5-Hashwert erzeugt

SHA1-Zertifikate auch nicht mehr unterstützt



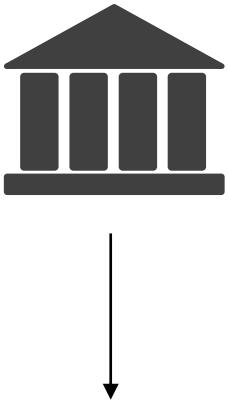
Chrome, Firefox,…
zeigen Webseiten
mit SHA1-Zertifikaten
seit 2015/16 als unsicher an

Browser-Test, ob SHA1 als unsicher erkannt wird

Zertifikate revozieren

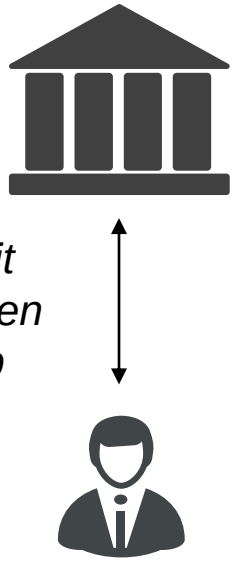
→ „Public-Key Infrastrukturen“

Certificate Revocation Lists (CRLs)



veröffentlicht unterschriebene Liste gesperrter Zertifikate

Online Certificate Status Protocol (OCSP)



fragt Gültigkeit eines bestimmten Zertifikats ab

In der Praxis meist OCSP, die aber eventuell selbst mit CRLs arbeiten

Im Web-Browser
(Beispiel vom Deutschen Forschungsnetz DFN)

Beispiel OCSP-fähiges, widerrufenes Zertifikat

Nennen Sie (die) drei Möglichkeiten, um einen öffentlichen Schlüssel zu prüfen.

Können Sie sich vorstellen, warum Zertifizierungsinstanzen anno 2009 noch MD5 verwendet haben, obwohl MD5 quasi 2005 gebrochen war?

Diskutieren Sie Vor- und Nachteile von CRLs gegenüber OSCP zur Zertifikationsüberprüfung.

Elektronische Signaturen

digitale Signatur (mathematisch)
vs.
elektronische Signatur (juristisch)

Signaturgesetz 1997 bzw. 2001

Gesetz zur Rechtssicherheit für elektronische Signaturen

(einfache) elektronische Signatur
mit Daten verknüpft, dient der Authentisierung

fortgeschrittene Signatur
dem Unterzeichner zuzuordnen
Unterzeichner hat Mittel zur Erstellung unter alleiniger Kontrolle
mit den unterzeichneten Daten sicher verknüpft

qualifizierte Signatur (kann Schriftform ersetzen)
wie fortgeschrittene Signatur
und qualifiziertes Zertifikat und sichere Signaturerstellungseinheit

Folgen

Signaturgesetz

Im Sinne dieses Gesetzes sind

1. "elektronische Signaturen" Daten in elektronischer Form, die anderen elektronischen Daten beigefügt oder logisch mit ihnen verknüpft sind und die zur Authentifizierung dienen,
2. "fortgeschrittene elektronische Signaturen" elektronische Signaturen nach Nummer 1, die
   a) ausschließlich dem Signaturschlüssel-Inhaber zugeordnet sind,
   b) die Identifizierung des Signaturschlüssel-Inhabers ermöglichen,
   c) mit Mitteln erzeugt werden, die der Signaturschlüssel-Inhaber unter seiner alleinigen Kontrolle halten kann, und

hat Signatur-Komponente, also:

Personalausweisgesetz (seit 2009)

§ 1 Ausweispflicht; Ausweisrecht

(1) Deutsche im Sinne des Artikels 116 Abs. 1 des Grundgesetzes sind verpflichtet, einen Ausweis zu besitzen, sobald sie 16 Jahre alt sind und der allgemeinen Meldepflicht unterliegen oder, ohne ihr zu unterliegen, sich überwiegend in Deutschland aufhalten. Sie müssen ihn auf Verlangen einer zur Feststellung der Identität berechtigten Behörde vorlegen. Vom Ausweisinhaber darf nicht verlangt werden, den Personalausweis zu hinterlegen oder in sonstiger Weise den Gewahrsam aufzugeben. Dies gilt nicht für zur Identitätsfeststellung berechtigte Behörden sowie in den Fällen der Einziehung und Sicherstellung.

eIDAS

seit 1.Juli 2016

electronic identification and trust services for electronic transactions

Europäische Verordnung, die die nationalen Signaturgesetze ablöst

sehr ähnlich zu Signaturgesetz

definiert weitere Authentisierungsarten neben elektronischer Signatur:

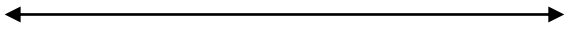
- elektronische Siegel (für juristische Personen/Services)
- Zeitstempel
- Zustellungsservices
- Webseitenauthentisierung
- …

Message Authentication Codes
(„symmetrische Signaturen“)

Brauchen wir immer Signaturen?

Hier muss schon der Schlüssel dem angedachten Partner zugeordnet werden können

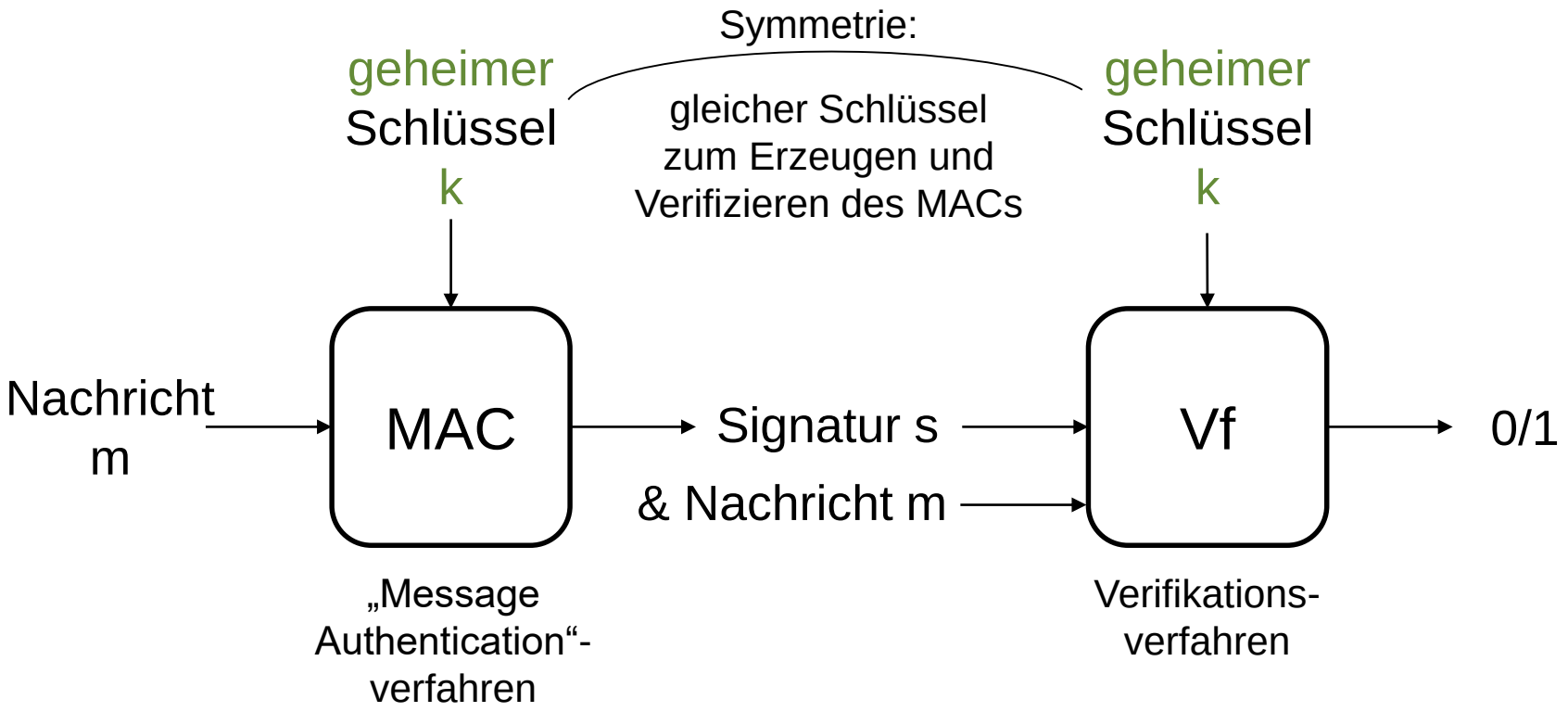
Statt Public-Key-Verschlüsselung verwenden wir die schnellere symmetrische Verschlüsselung mit dem ausgehandelten Schlüssel k



Symmetrische Verschlüsselung mittels k

Ansatz: weiteren, dem Partner auch zuzuordnenden Schlüssel k* aushandeln und zum schnelleren Integritätsschutz verwenden!

Prinzip der Message Authentication Codes (MACs)



Funktionale Korrektheit (Vollständigkeit):
Für alle Nachrichten m und alle Schlüssel k gilt: Vf(k,m,MAC(k,m))=1

MACs in der Praxis

HMAC (Hash-Funktions-basierter MAC für SHA-1, SHA-2):

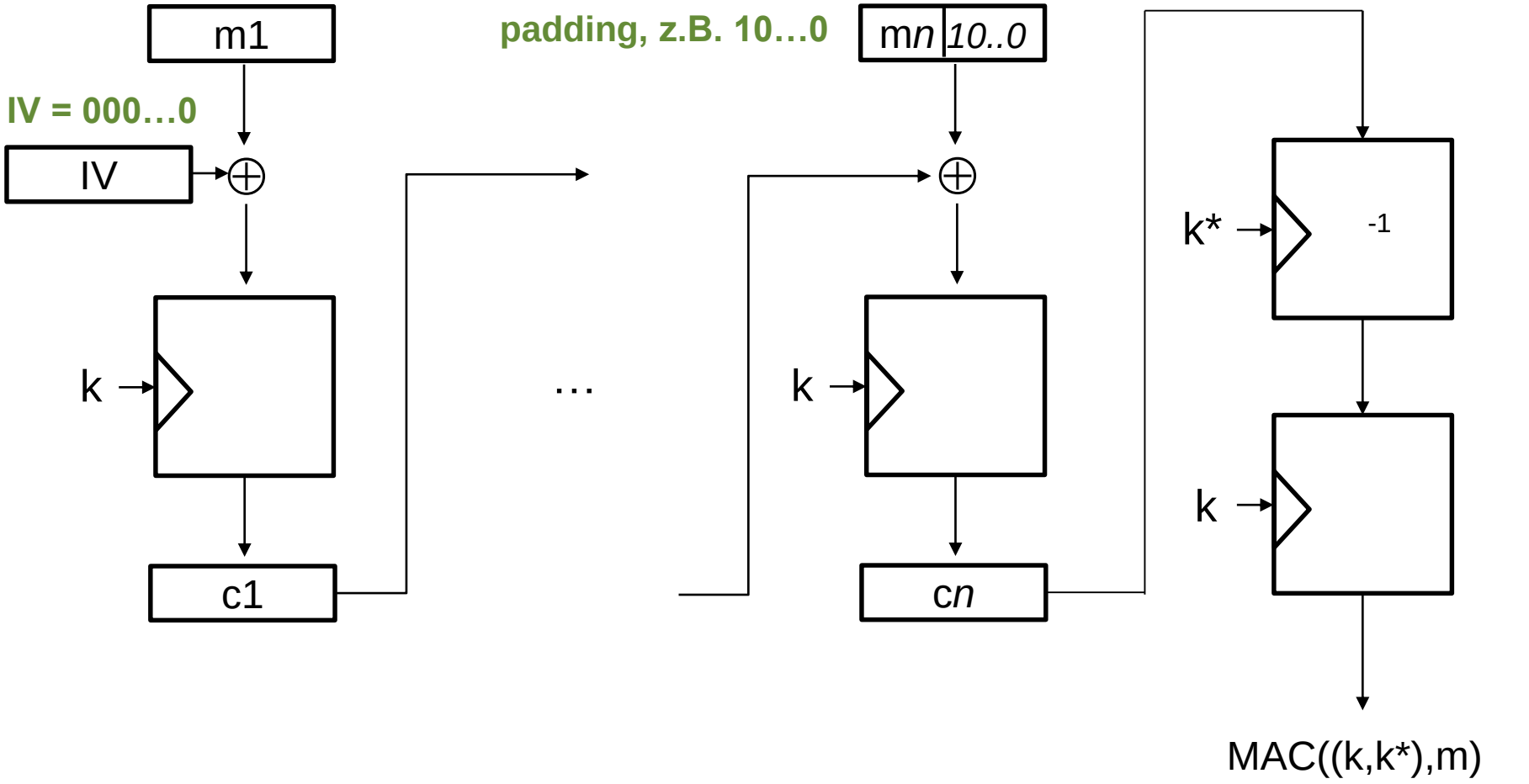
$$HMAC(k,m) = H (k \oplus opad \,||\, H(k \oplus ipad \,||\, m))$$

opad = **0x5c 0x5c 0x5c** …
Ipad = **0x36 0x36 0x36** …

Prinzipiell auch für SHA-3 geeignet, dort kann man aber wegen anderer Struktur der Hash-funktion sogar H(k||m) direkt benutzen

MACs in der Praxis

CBC-MAC (für 3DES, AES) im Retail Mode:



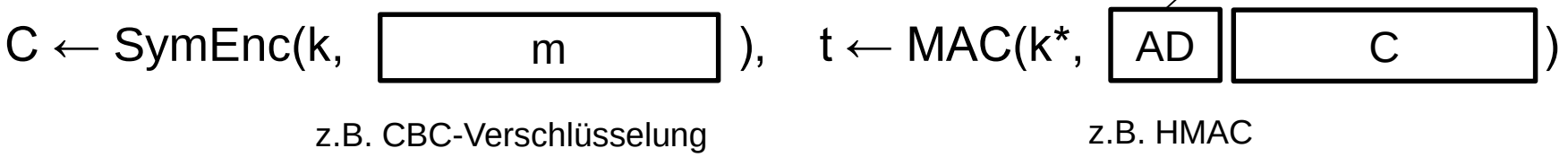
evtl. noch „abschneiden“, z.B. auf 96 Bits

Authenticated Encryption (with associated data, AEAD)

Kontextinformation
z.B. TCP Header

Kombination für Vertraulichkeit und Integrität

Encrypt-then-MAC:



Ausgabe = (AD, C, t)

Entschlüsseln: nur wenn Vf(k*,AD|C,t)=1, dann m←Dec(k,C) ausgeben

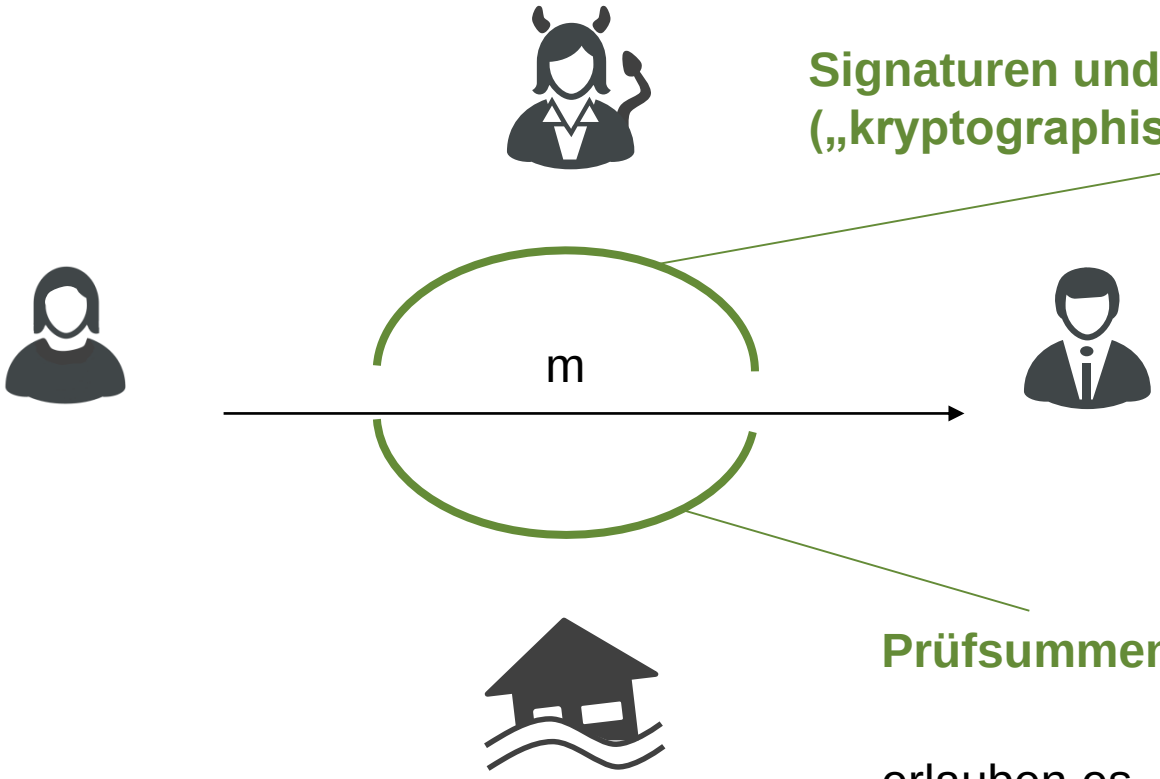
schnellere „Verzahnung“ von Enc und MAC in speziellen Modi wie Galois/Counter Mode

Warum kann man MACs nicht als elektronische Signatur verwenden?

Warum ist folgendes „hybride“ Signaturverfahren keine gute Idee? Um m zu signieren wähle man Schlüssel k, berechne s=Sig(sk,k) und t=MAC(k,m) und gebe (s,t,k) aus.

*Warum kann man nicht den letzten Block einfach so als CBC-MAC der Nachricht m nehmen?

Prüfsummen



Signaturen und MACs („kryptographische Prüfsummen“)

Prüfsummen

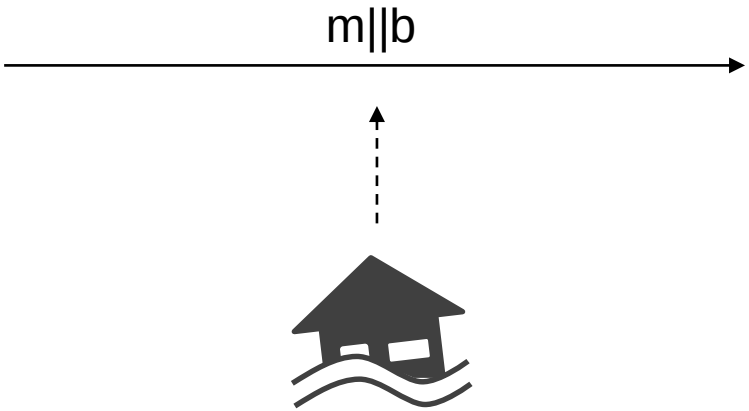
erlauben es,
(unabsichtliche) Fehler zu erkennen
und evtl. sogar zu korrigieren

Paritätsprüfsumme

Sender hat Nachricht $m = b_0 | b_1 | \ldots | b_{n-1}$ aus Bits b_j

Berechnet Paritätsbit $b = b_0 \oplus b_1 \oplus \ldots \oplus b_{n-1}$

Sendet $m||b$



Empfänger prüft, dass
$b = b_0 \oplus b_1 \oplus \ldots \oplus b_{n-1}$
für $m = b_0 | b_1 | \ldots | b_{n-1}$

JA: gib m aus
NEIN: gib FEHLER aus

Wenn kein Bit in m|b „gekippt“, dann gibt Empfänger m aus
Wenn ein Bit in m|b „gekippt“, dann gibt Empfänger FEHLER aus
Wenn zwei oder mehr Bits in m|b „gekippt“, dann falsche Ausgabe

Erlaubt es, bis zu einen Fehler zu entdecken, aber nicht zu korrigieren

Grundlagen der Cyclic Redundancy Checks (CRCs)

Operieren über Polynomen $a_0x^0 + a_1x^1 + \cdots + a_{n-1}x^{n-1}$

mit Koeffizienten über {0,1}

\+ entspricht damit $\oplus$

Insbesondere: $x^i + x^i = 0$ und damit $\sum a_i x^i + \sum b_i x^i = \sum (a_i \oplus b_i) \cdot x^i$

Polynomdivision $A(x)/B(x)$

$A(x) = x^4 + x^3 + 1$ und $B(x) = x^2 + 1$

$A(x)/B(x) = x^2 + x + 1$, Rest x

Cyclic Redundancy Check (CRC)

1. Schritt: wähle geeignetes CRC-Polynom, z.B.

$$C(x) = x^0 + x^1 + x^2 + x^4 + x^5 + x^7 + x^8 + x^{10} + x^{11} + x^{12} + x^{16} + x^{22} + x^{23} + x^{26} + x^{32}$$

„Ethernet-CRC32“

2.Schritt:
Betrachte Nachricht $m = b_0|b_1 \ldots |b_{n-1}$ als Polynom

$$M(x) = b_0 x^0 + b_1 x^1 + \cdots + b_{n-1} x^{n-1}$$

wobei n-1 Vielfaches vom Grad des CRC-Polynoms sein muss, sonst muss man m mit 0en auffüllen

3.Schritt:
Sende m und CRC(m) = Divisionsrest von $M(x)/C(x)$

CRC checken

Empfänger prüft bei Erhalt von m*,
ob CRC(m*) mit erhaltener Summe übereinstimmt

Für geeignete CRC-Polynome
kann man jede ungerade Anzahl von Bitflips in Nachricht feststellen

Vor allem Fehlerbündel („burst errors“)
von zusammenhängenden Bitfehlern werden erkannt

Angriff auf WEP

ab 2001

WEP = Wired Equivalent Privacy

Ehemaliges Protokoll zur Absicherung von WLANs

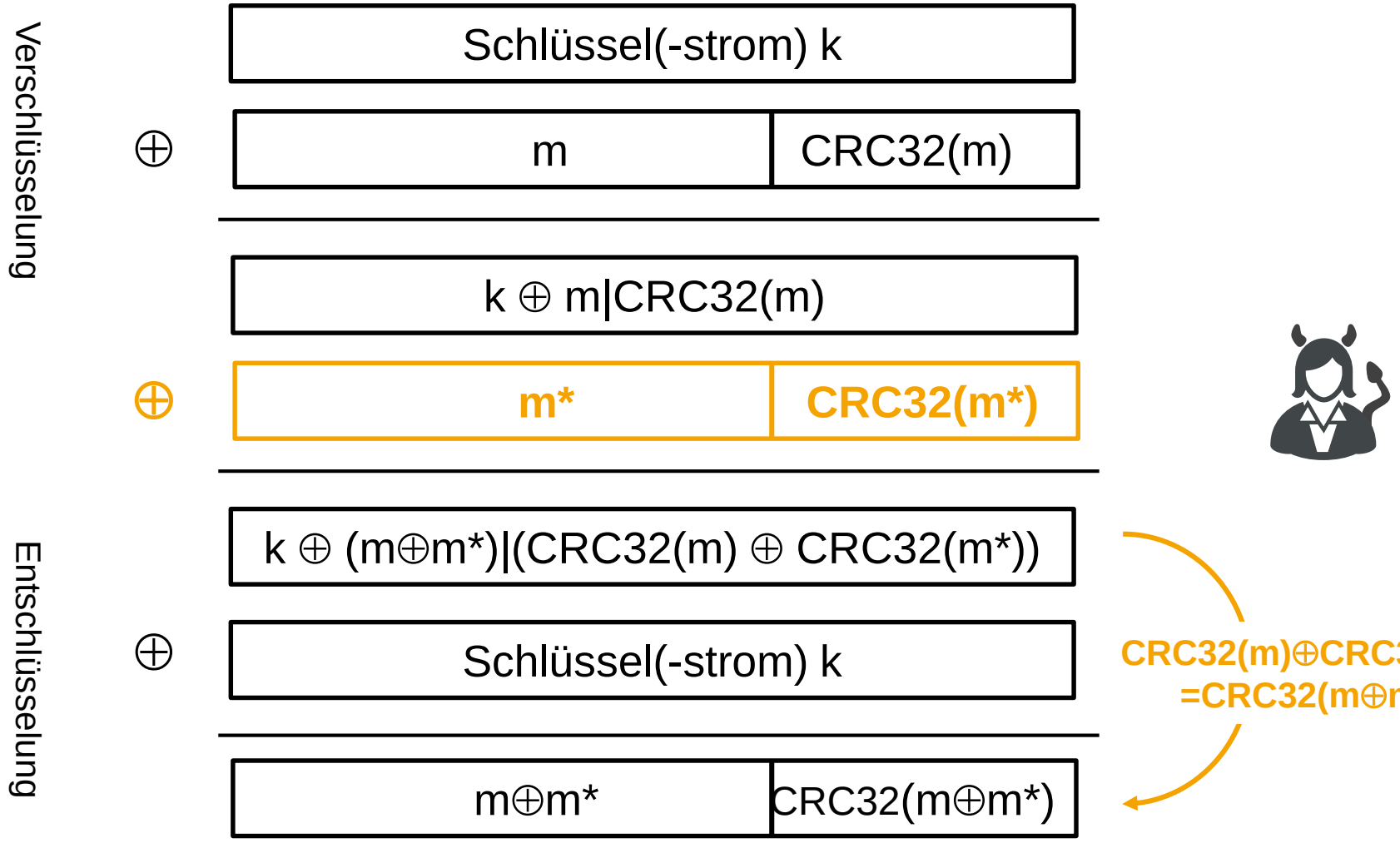
Inzwischen WPA2 und dazwischen das auch als unsicher geltende WPA

Wi-Fi Protected Access

WEP(m)

Hier nur speziell der Angriff auf vermeintliche Integrität durch CRC in WEP

(Fehlender) Integritätsschutz in WEP



Was Sie gelernt haben sollten

Digitale Signaturen

Hashfunktionen

Kandidaten in der Praxis (Signaturverfahren + Hashfunktionen)

Zertifizierung von öffentlichen Schlüsseln

Elektronische Signaturen

Message Authentiction Codes (MACs)

Kandidaten in der Praxis

Prüfsummen

Unterschied Prüfsummen vs. Signaturen und MACs