Foto: kuu akura auf Unsplash

RSA-Verschlüsselung

Ein asymmetrisches Verschlüsselungsverfahren

INNOVET
KI B³
Künstliche Intelligenz in die berufliche Bildung bringen

INNOVET
GEFÖRDERT VOM
Bundesministerium für Bildung und Forschung
bibb Bundesinstitut für Berufsbildung

Gefördert als InnoVET-Projekt aus Mitteln des Bundesministeriums für Bildung und Forschung.

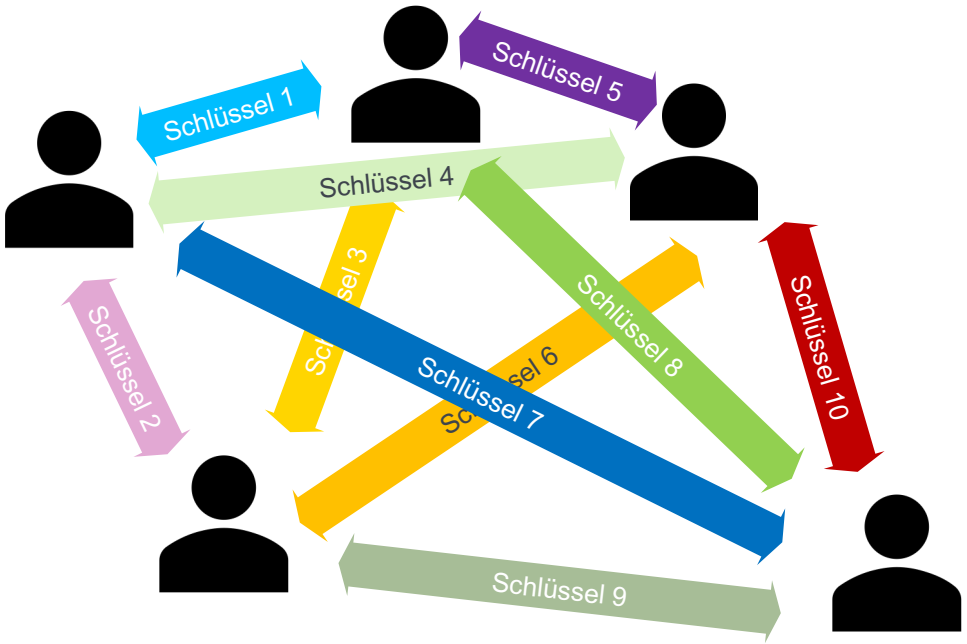
Symmetrische Verfahren

- Gleicher Schlüssel für Verschlüsselung und Entschlüsselung*
- Unpraktisch, wenn der Schlüssel mit anderen geteilt werden soll:

1. Wie kommt der Schlüssel sicher zum Empfänger?
2. „Schlüsselinflation“

* genauer: wer den Schlüssel zur Verschlüsselung kennt, kann auch entschlüsseln

Schlüsselinflation



Bei 5 Personen:

4+3+2+1=10 Schlüssel

Bei 25 Personen:

300 Schlüssel

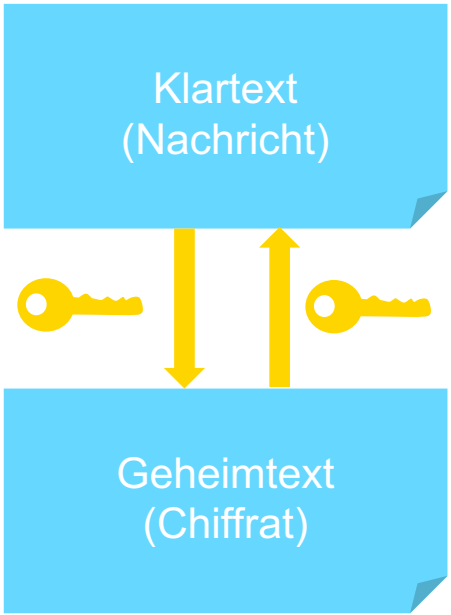
Bei 1000 Personen:

fast 500 000 Schlüssel

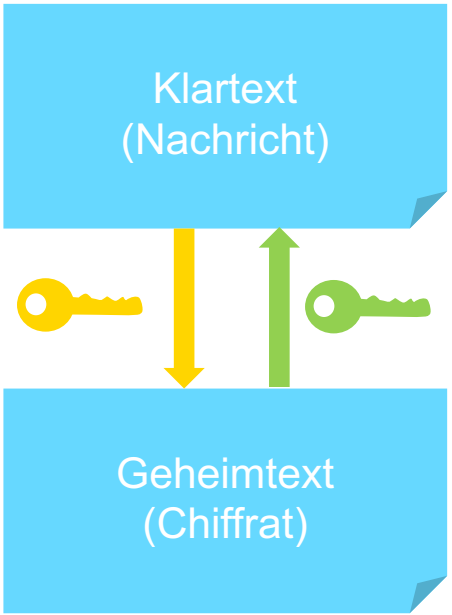
Asymmetrische Verfahren

Symmetrisch vs. asymmetrisch

Idee: verschiedene Schlüssel zur Verschlüsselung und zur Entschlüsselung



Symmetrisch:
„gleicher“ Schlüssel

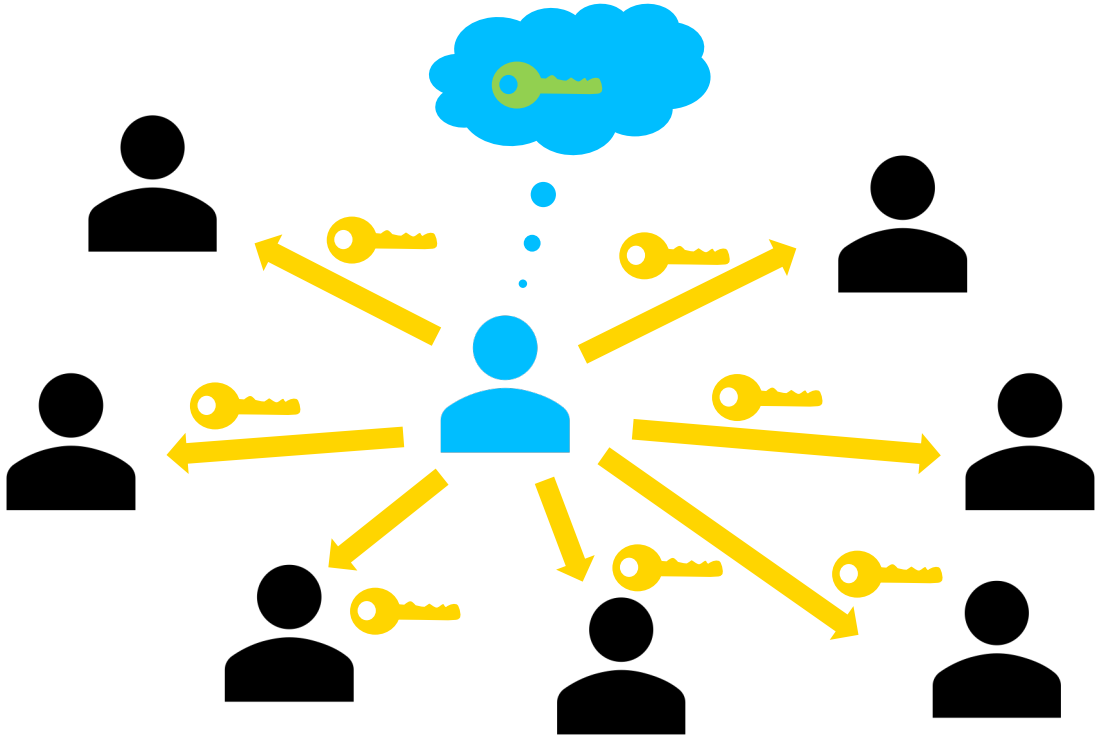


Asymmetrisch:
verschiedene Schlüssel

Schlüsselaustausch bei asymmetrischen Verfahren

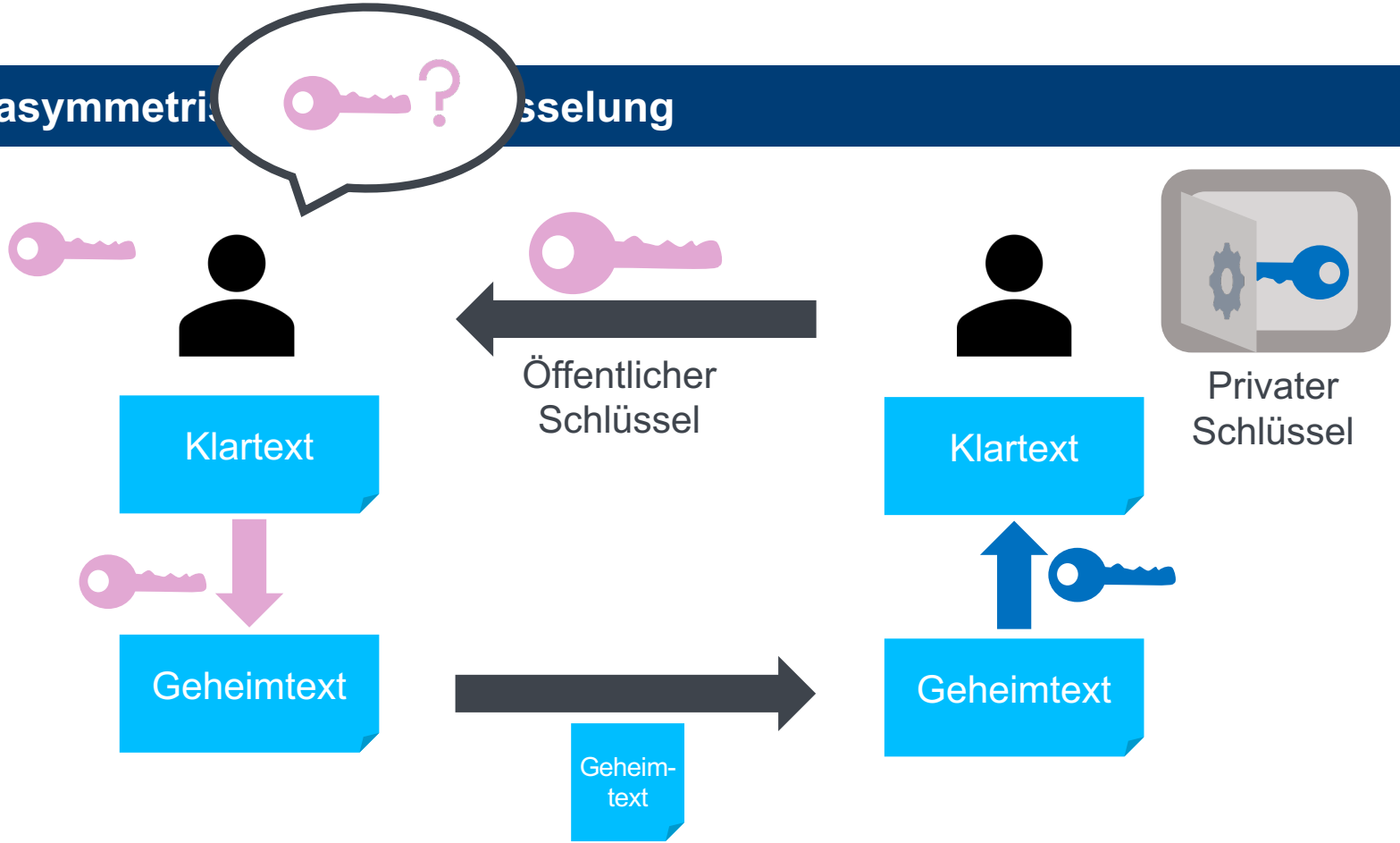
Schlüssel zur Verschlüsselung wird vom Empfänger beliebig verteilt

Schlüssel zur Entschlüsselung bleibt beim Empfänger



Prinzip bei asymmetrischer Verschlüsselung

Prinzip bei asymmetri sselung



Bei der asymmetrischen Verschlüsselung braucht jeder Empfänger einen öffentlichen und einen privaten Schlüssel.

Der öffentliche Schlüssel kann bedenkenlos verteilt werden – es ist keine geschützte Übertragung nötig, da mit ihm nicht entschlüsselt werden kann.

Der private Schlüssel (engl.: Private Key) wird vom Empfänger zum Entschlüsseln von Nachrichten genutzt und bleibt geheim.

Der öffentliche Schlüssel (engl.: Public Key) des Empfängers wird vom Sender zum Verschlüsseln von Nachrichten für den Empfänger genutzt.

Das RSA-Verfahren

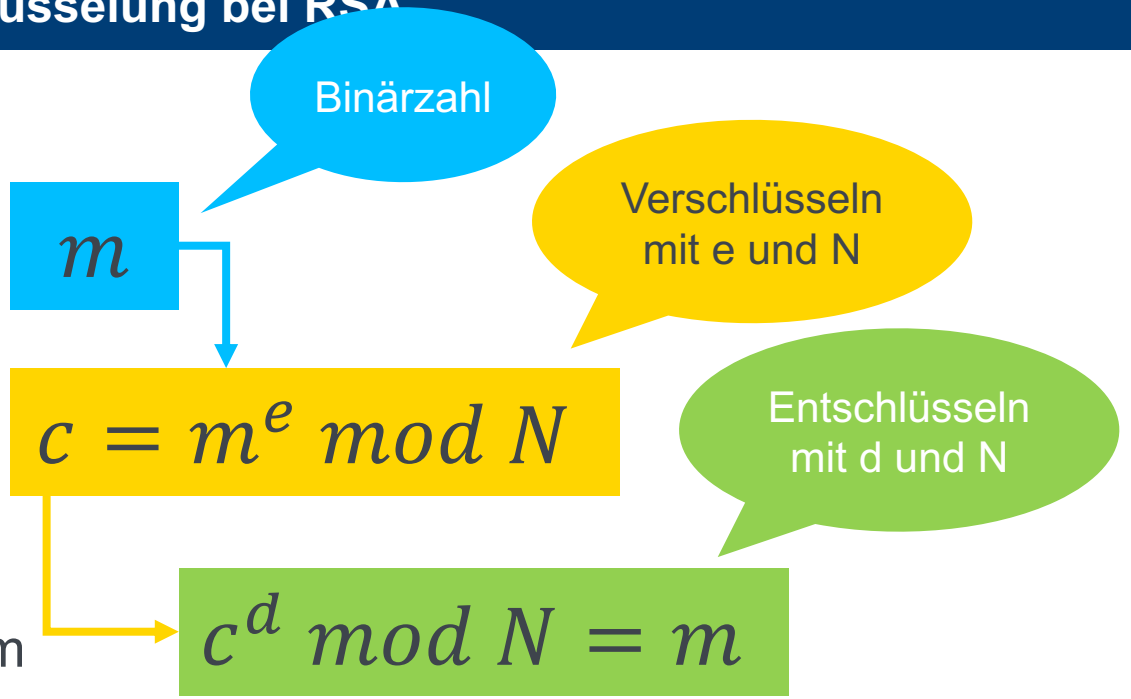
Rivest–Shamir–Adleman (nach den Erfindern)

Verschlüsselung und Entschlüsselung bei RSA

Nachricht m
(Klartext)

Chiffrat c
(„Geheimtext“)

entschlüsselte Nachricht m



Öffentlicher und privater Schlüssel bei RSA

Öffentlicher Schlüssel:

- Verschlüsselungsexponent e
- RSA-Modul N

Verschlüsseln mit e und N

$$c = m^e \ mod \ N$$

Entschlüsseln mit d und N

Privater Schlüssel:

- Entschlüsselungsexponent d
- (RSA-Modul N)

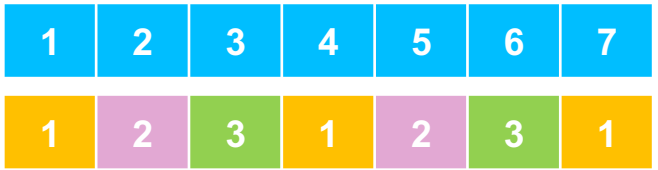
$$c^d \ mod \ N = m$$

Die Modulo-Operation

Wenn wir bis 7 zählen,
aber nur die Zahlen
1, 2 und 3 haben,
ist die 7 wieder die 1…

- Der „Rest“ beim Dividieren

7 : 3 = 2 Rest 1



- 7 mod 3 = 1

Ein Beispiel (mit kleineren Zahlen als üblich)

Öffentlicher Schlüssel

N = 187, e = 3

Privater Schlüssel

N = 187, e = 107

Länge von N: 8 Bit

Empfehlung des Bundesamts für Sicherheit und Informations-technik, Stand 2/24: 3000 Bit

65

$65^3 \ mod \ 187 = 274625 \ mod \ 187 = 109$

109

$109^{107} \ mod \ 187 = 65$

65

Klartext, hier A in ASCII-Kodierung

https://www.symbolab.com/solver/modulo-calculator

Wie bestimmt man die Schlüssel?

$$c = m^e \ mod \ N$$

$$c^d \ mod \ N$$

$$= \ m^{e^d} \ mod \ N$$

$$= \ m^{e \cdot d} \ mod \ N$$

Soll wieder m ergeben!

Hergeleitet aus dem Satz von Euler:

Dies klappt genau dann, wenn

$$e \cdot d \ mod \ \phi(N) = 1$$

$\phi(N)$
Eulersche Phi-Funktion

Das multiplikative Inverse finden

Entschlüsselung mit d ergibt wieder m, wenn $e \cdot d \; mod \; \phi(N) = 1$

d das **multiplikative Inverse** zu e

- d existiert nur, wenn e und $\phi(N)$ keine gemeinsamen Teiler haben
- d kann für e relativ leicht berechnet werden, wenn man $\phi(N)$ kennt
- $\phi(N)$ kann man nur berechnen, wenn man die Teiler von N kennt
- Suche nach Teilern vor allem bei großen Teilern extrem aufwändig

Die Sicherheit von RSA kommt daher, dass man zum Knacken des privaten Schlüssels die Teiler des RSA-Moduls kennen muss.

Es dauert bei großen Teilern allerdings extrem lange, sie zu bestimmen. Daher kann man in der Praxis aus dem öffentlichen Schlüssel nicht den privaten Schlüssel herleiten.

Die Wahl der Schlüssel bei RSA

Nicht speichern!	Öffentlich: RSA-Modul	Nicht speichern!	Öffentlich: Verschlüsselungs-exponent	Geheim: Entschlüsselungs-exponent
p, q	N	$\phi(N)$	e	d
Beliebige Primzahlen Möglichst groß	$p \cdot q$	$(p-1) \cdot (q-1)$ Da p und q Primzahlen	frei gewählt kleiner als N, kein gemeinsamer Teiler mit $\phi(N)$ Sonst gibt es kein d	mithilfe des sog. erweiterten Euklidischen Algorithmus und $\phi(N)$ berechnet

Zur Erzeugung der Schlüssel wählt man zwei sehr große Primzahlen. Multipliziert ergeben diese dann den RSA-Modul.

So sind die Teiler des RSA-Modul bekannt, und die Bestimmung des zum Verschlüsselungsexponenten passenden Entschlüsselungsexponenten ist leicht.

Die verwendeten Primzahlen sowie der Wert der Eulerschen Phi-Funktion müssen natürlich geheim bleiben.

Ein Beispiel (mit kleineren Zahlen als üblich)

Nicht speichern! ✗	Öffentlich: RSA-Modul	Nicht speichern! ✗	Öffentlich: Verschlüsselungs-exponent	Geheim: Entschlüsselungs-exponent
p, q	N	$\phi(N)$	e	d
Beliebige Primzahlen 11, 17	$p \cdot q$ 187	$(p-1) \cdot (q-1)$ $10 \cdot 16 = 160$	kleiner als N, kein gemeinsamer Teiler mit $\phi(N)$ 3	107

Ein Beispiel (mit kleineren Zahlen als üblich)

Öffentlicher Schlüssel

N = 187, e = 3

Privater Schlüssel

N = 187, e = 107

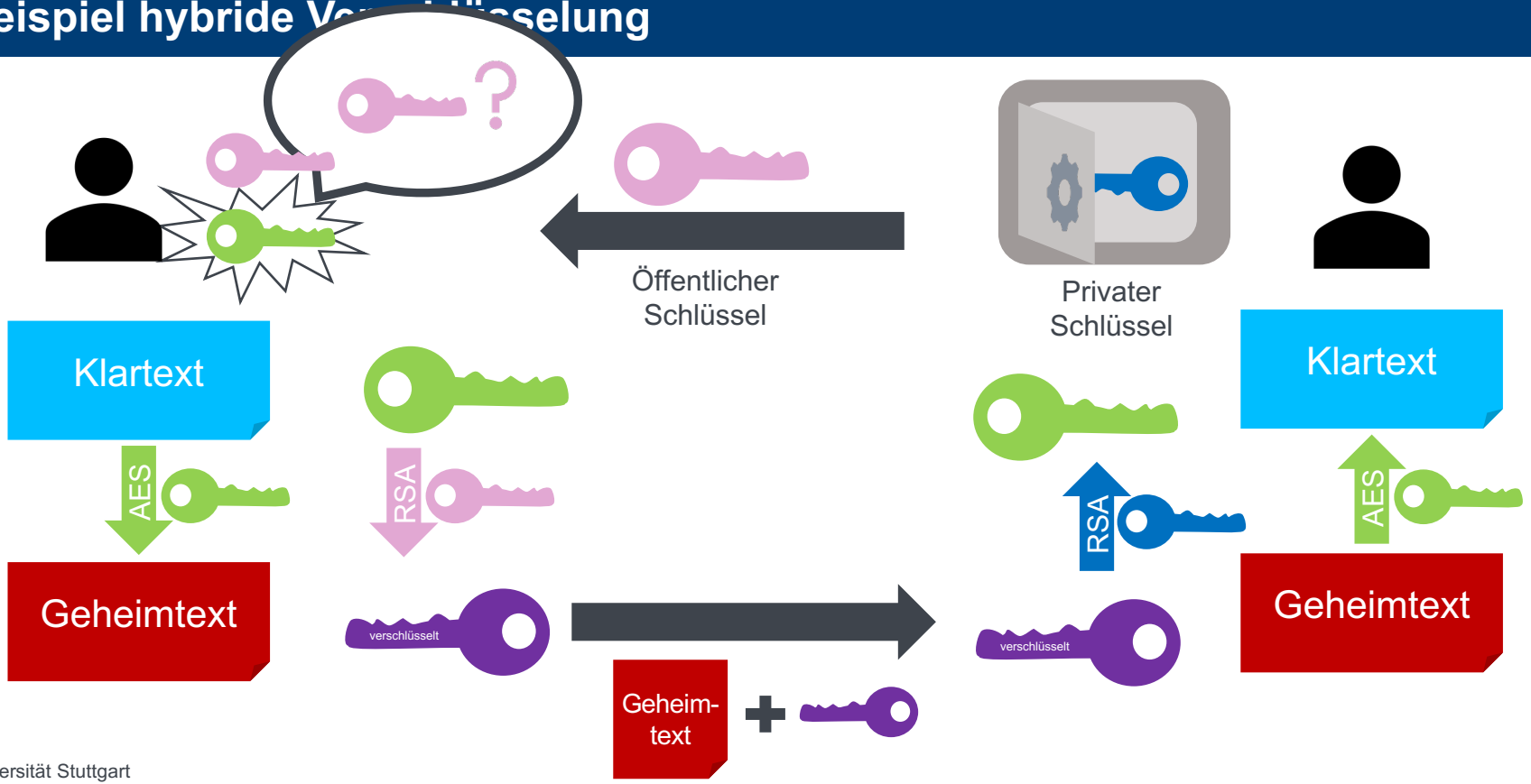
Öffentlich: RSA-Modul	Öffentlich: Verschlüsselungs-exponent	Geheim: Entschlüsselungs-exponent
N	e	d
$p \cdot q$ 187	kleiner als N, kein gemeinsamer Teiler mit $\phi(N)$ 3	 107

RSA in der Praxis

Modifikation von RSA in der Praxis

- RSA wird nicht genau wie bisher beschrieben zum Verschlüsseln der Nachricht eingesetzt
  - Angreifbar, insbesondere bei sehr kurzen Nachrichten
  - Verschlüsselung und Entschlüsselung sind sehr rechenintensiv
- Statt dessen z.B. möglich:
  - Hybrides Verfahren
  - Erzeugung eines nur für eine Nachricht gültigen Schlüssels (Session Key) für AES (!)
  - Verschlüsselung der Nachricht mit AES mit dem Session Key
  - Verschlüsselung des Session Key mit RSA mit dem öffentlichen Schlüssel des Empfängers
  - Übertragung von AES-verschlüsselter Nachricht und RSA-verschlüsseltem Session Key

Beispiel hybride Verschlüsselung



Zusammenfassung

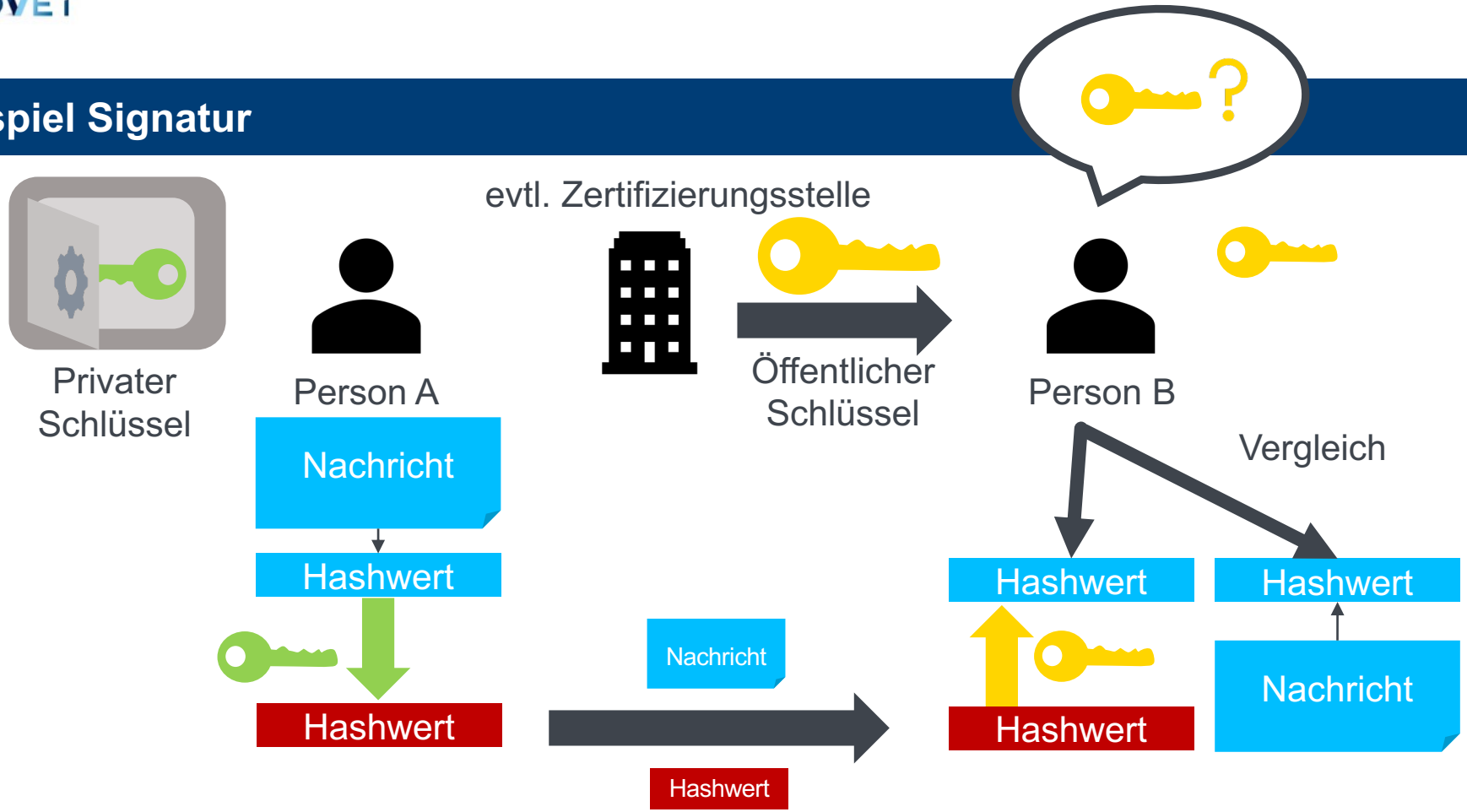
- RSA kann mit AES zu einem hybriden Verschlüsselungsverfahren kombiniert werden
- Weniger angreifbar als RSA alleine
- Effizienter, da AES weniger rechenintensiv

RSA zum Erstellen von Signaturen

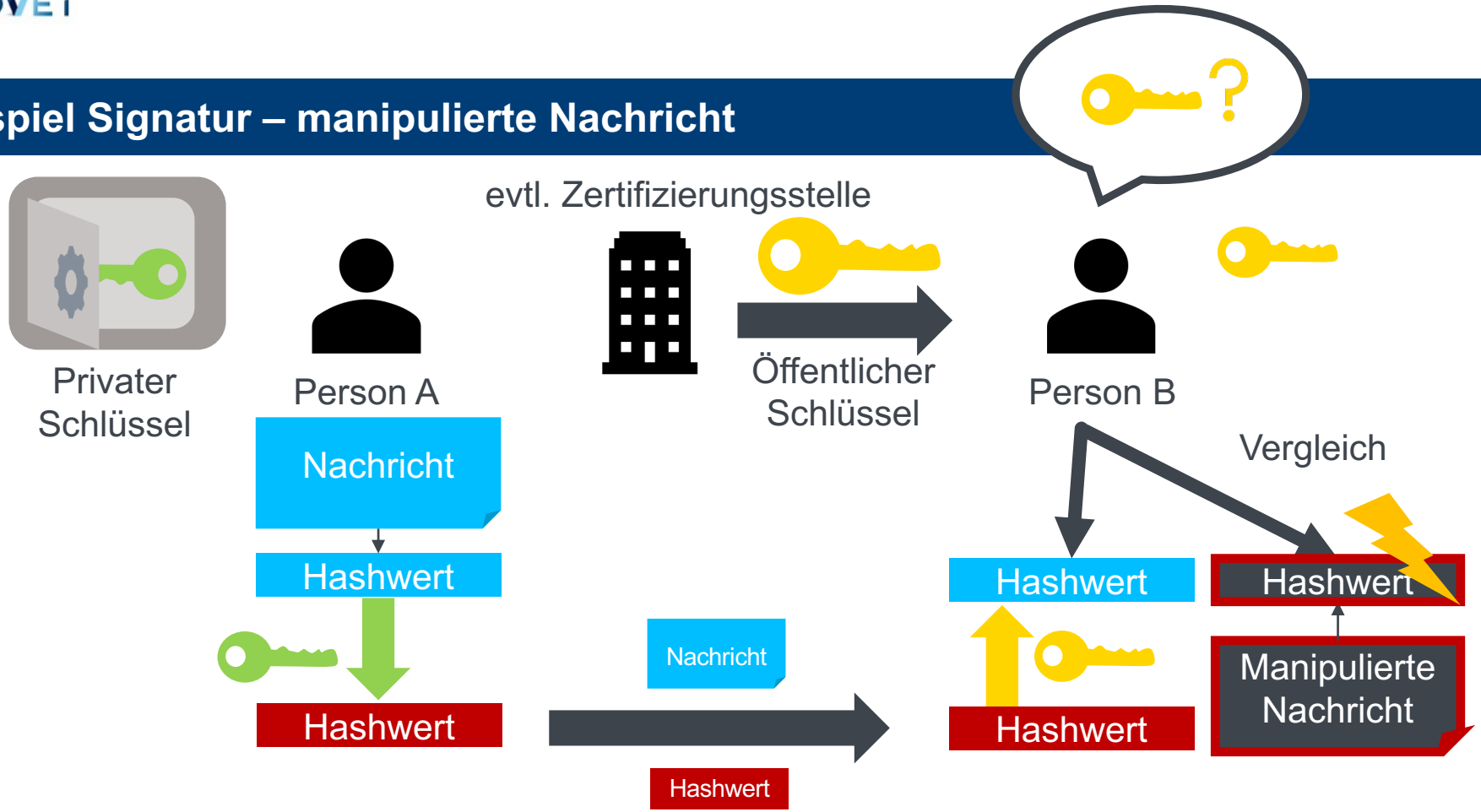
Idee digitale Signaturen

- Sicher stellen, dass Nachricht nicht abgefangen und manipuliert wurde
- Sicher stellen, dass Nachricht wirklich vom Absender kommt
- Sender
  - Aus der Nachricht eindeutigen Hashwert berechnen
  - Mit dem privaten Schlüssel verschlüsselt mitschicken
- Empfänger
  - Entschlüsselt den Hashwert mit dem öffentlichen Schlüssel des Senders
  - Berechnet den Hashwert der Nachricht selbst und vergleicht

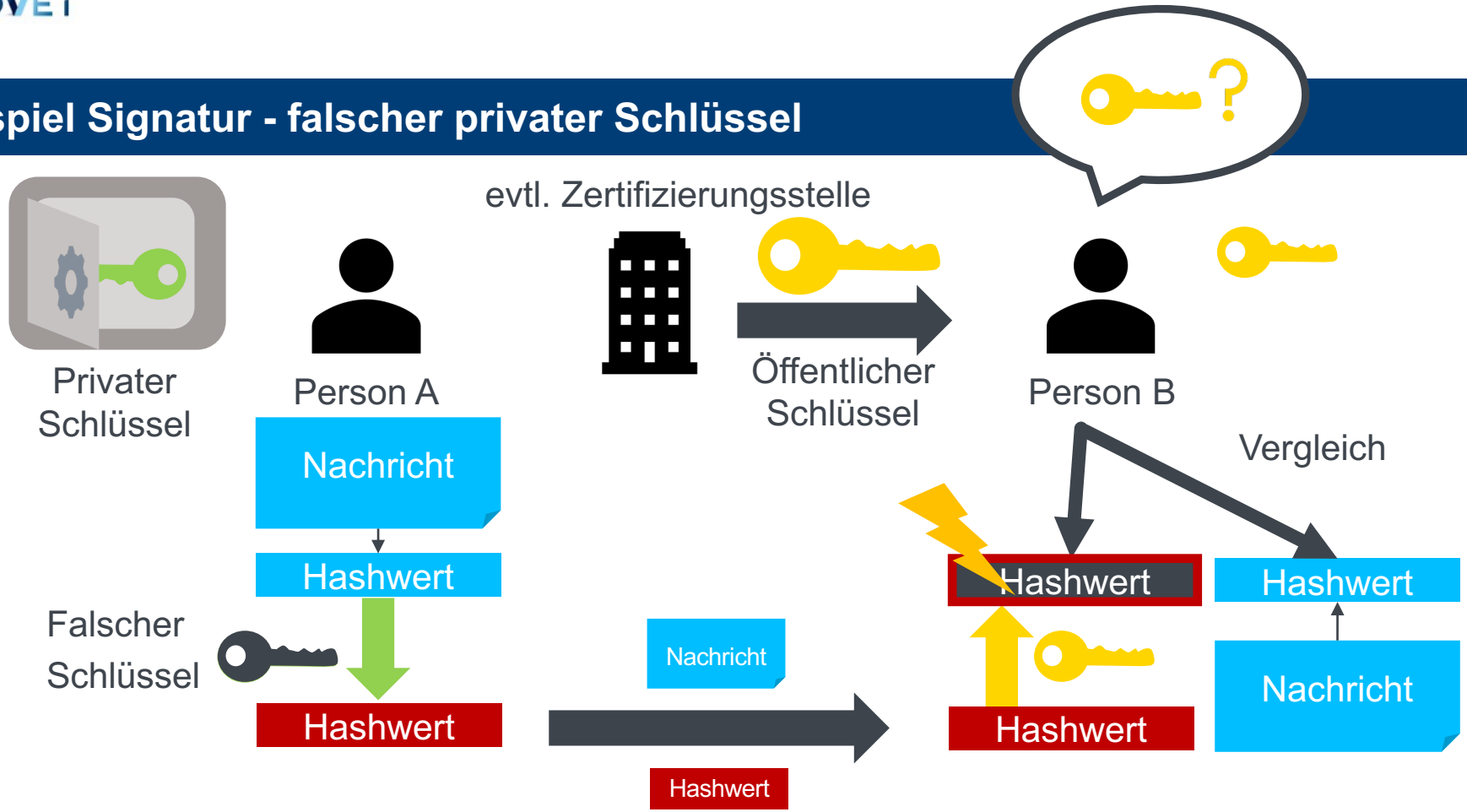
Beispiel Signatur



Beispiel Signatur – manipulierte Nachricht



Beispiel Signatur - falscher privater Schlüssel



Öffentliche und private Schlüssel können auch genutzt werden, um Nachrichten mit digitalen Signaturen zu versehen.

Hierbei wird ausgenutzt, dass der private Schlüssel nur dem echten Sender bekannt ist. Passt er zum öffentlichen Schlüssel, stammt die Nachricht wirklich vom Sender – die Authentizität ist gegeben.

Der verschlüsselt mitgeschickte Hashwert zur Nachricht garantiert dagegen, dass die Nachricht unterwegs nicht manipuliert wurde – die Integrität ist gegeben.

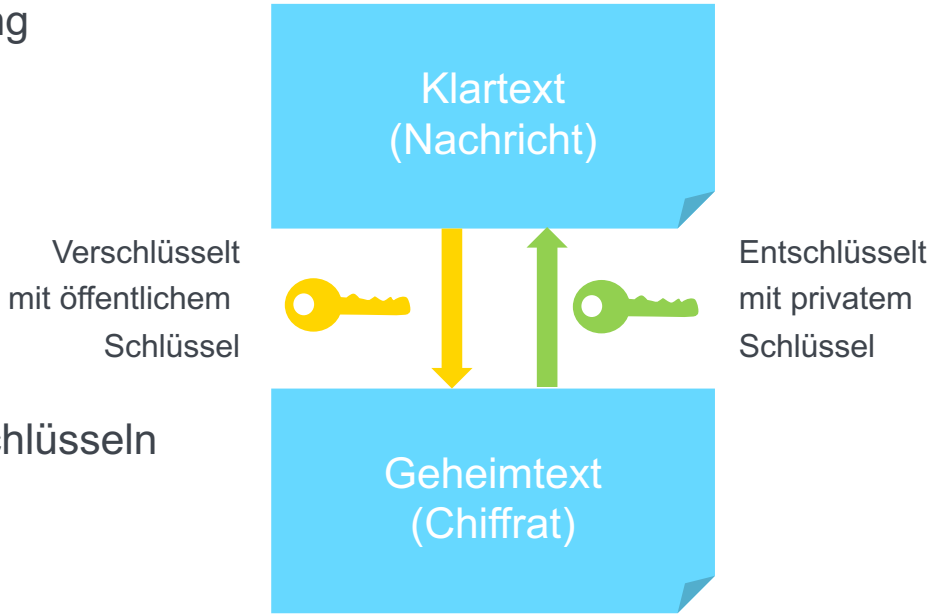
Denn wenn sich die Nachricht seit dem Absenden verändert hat, verändert sich auch der Hashwert und passt dann nicht mehr zum verschlüsselten originalen Hashwert.

Erinnerung: Prinzip bei Verschlüsselung von Nachrichten

- Öffentlicher Schlüssel zur Verschlüsselung
- Privater Schlüssel zur Entschlüsselung
- Funktioniert auch anders herum:

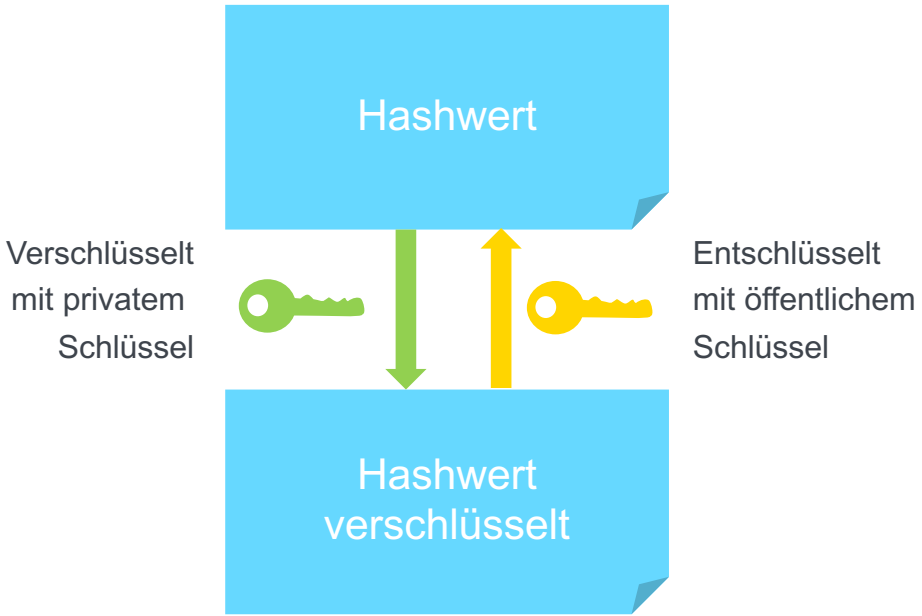
$$m^{e \cdot d} \bmod N = m$$

- Aber: es soll ja nur die eine Person entschlüsseln können, für die die Nachricht gedacht ist
- Daher bisher sinnvoll:
  Entschlüsselung mit privatem Schlüssel



Prinzip bei der Erstellung von Signaturen

- Signatur soll von jedem beliebigen Empfänger entschlüsselt werden können
- Aber nur von einer Person korrekt verschlüsselt („unterschrieben“) werden können: vom Sender



Anmerkung

- In der Praxis wird auch hierfür eine modifizierte, sicherere Variante von RSA verwendet

INNOVET
KI B³
Künstliche Intelligenz in die berufliche Bildung bringen

Dr. Antje Schweitzer

Universität Stuttgart
Institut für Maschinelle Sprachverarbeitung

Universität Stuttgart
Institut für Maschinelle Sprachverarbeitung
Institut für Software Engineering

IHK Industrie- und Handelskammer Reutlingen
Reutlingen | Tübingen | Zollernalb

IHK Region Stuttgart

IHK Industrie- und Handelskammer Karlsruhe

LMU Ludwig-Maximilians-Universität München

GEFÖRDERT VOM
INNOVET
Bundesministerium für Bildung und Forschung
bibb Bundesinstitut für Berufsbildung

Gefördert als InnoVET-Projekt aus Mitteln des Bundesministeriums für Bildung und Forschung.

Lizenzbestimmungen

“RSA-Verschlüsselung" von Antje Schweitzer, KI B³ / Uni Stuttgart

Das Werk - mit Ausnahme der folgenden Elemente:

- Logos der Verbundpartner und des Förderprogramms
- im Quellenverzeichnis aufgeführte Medien

ist lizenziert unter:

 CC BY 4.0 (https://creativecommons.org/licenses/by/4.0/deed.de)

(Namensnennung 4.0 International)

Quellenverzeichnis

Titelfoto: kuu akura (https://unsplash.com/de/@akurakuu), ohne Titel, auf Unsplash (https://unsplash.com/de/fotos/text-symbol-pnK6Q-QTHM4), lizenziert unter Unsplash-Lizenz (https://unsplash.com/license). Bildausschnitt verändert.