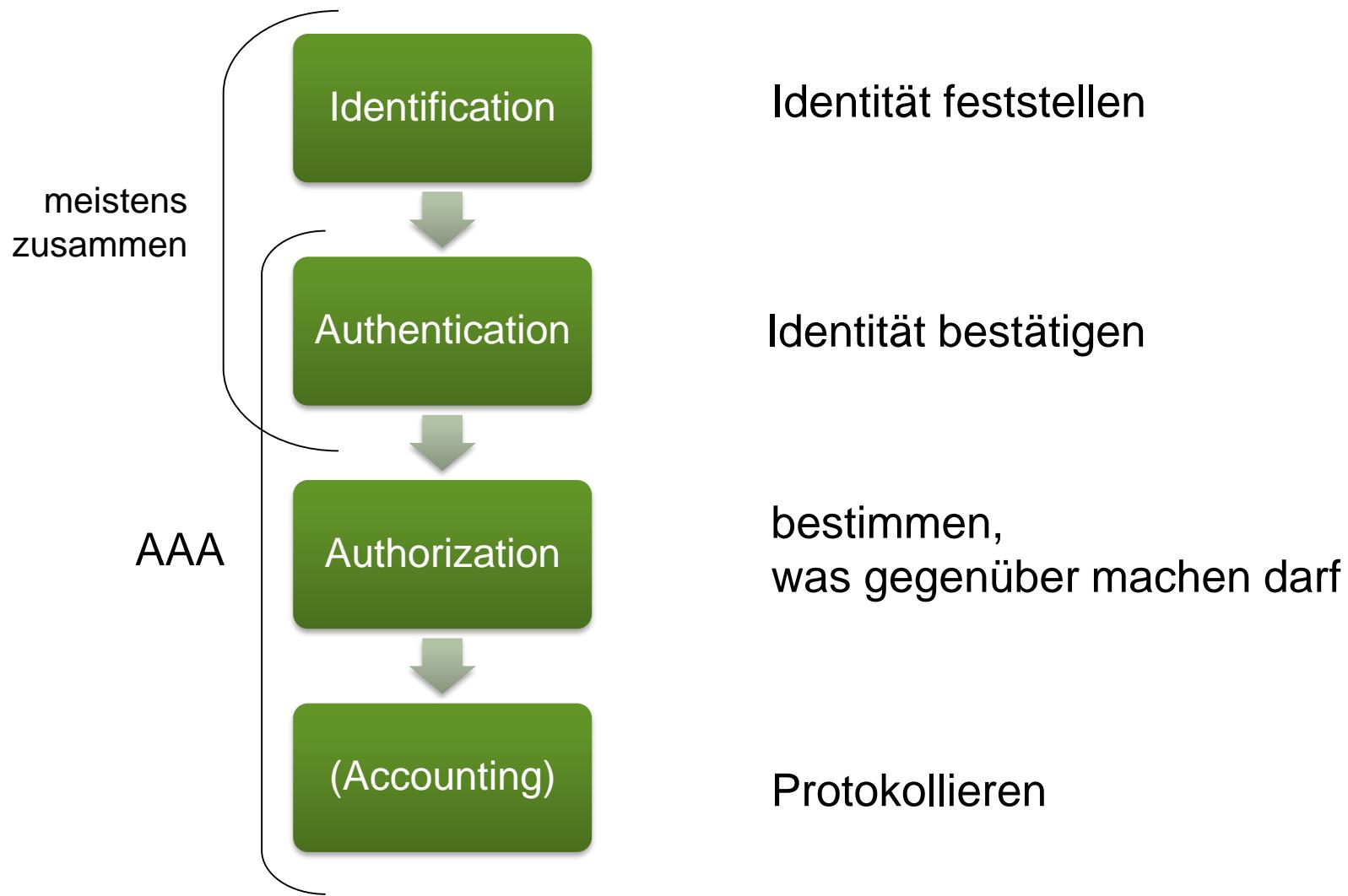


# Computersystemsicherheit



Prof. Marc Fischlin, Wintersemester 18/19

04  
Authentisierung und Autorisierung



# Identifikation ≠ Authentisierung



„offener“ SMTP-Server  
(Senden von E-Mails ohne Authentisierung)



E-Mail-Server

```
telnet open.smtp.server.com 25
Trying www.ddd.ddd.ddd...
Connected to open.smtp.server.com.
Escape character is '^].
220 www.ddd.ddd.ddd
HELO localhost
250 www.ddd.ddd.ddd
MAIL FROM:bob@provider.de
250 2.1.0 Ok
RCPT TO:alice@mail.de
250 2.1.5 Ok
DATA 354 End data with <CR><LF>.<CR><LF>
Subject: Hallo Alice

Bitte schicke mir deine Bankdaten
...
```



Thunderbird

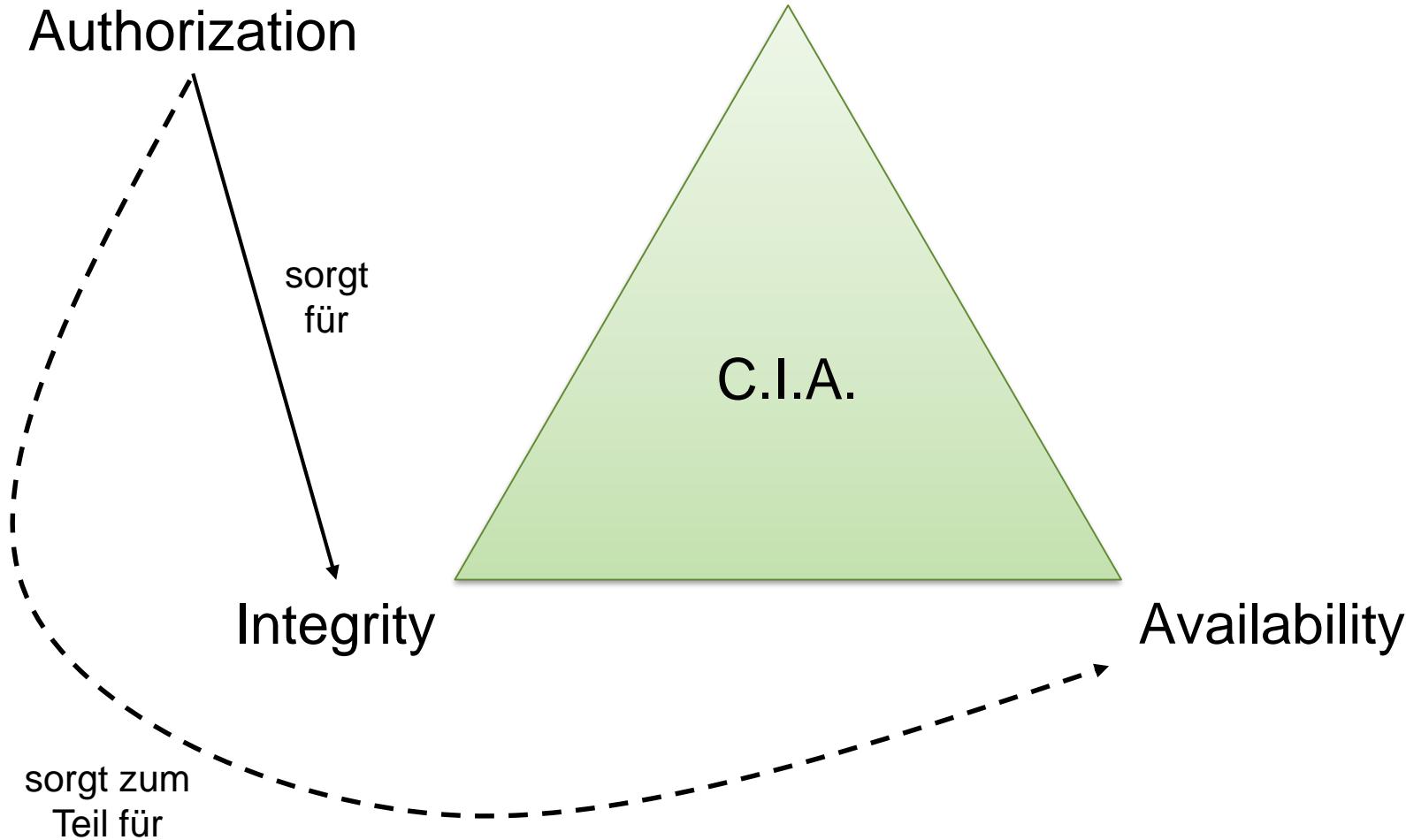
Von: Bob <bob@provider.de>  
Betreff: Hallo Alice  
An: Alice <alice@mail.de>

Alice identifiziert  
Absender Bob  
vom E-Mail-Header

Authentication  
+  
Authorization

sorgt  
für

→ Confidentiality



# Mittel zur Authentisierung

Authentication

Was man weiß

- z.B. Passwörter

Fdfjd#s3

Was man hat

- z.B. Chipkarte



Was man ist

- z.B. biometrische Merkmale



Weitere Faktoren:  
Wo man ist, Was man tut

Multi-faktor Authentisierung:  
Kombinationen von mehreren Faktoren

# Was man weiß

Authentication

Was man weiß

- z.B. Passwörter

Fdfjd#s3

## Vorteile

+einfach zu ändern

+einfach mitnehmbar

## Nachteile

-kann vergessen werden

-leicht zu duplizieren

Weitere Faktoren:  
Wo man ist, Was man tut

Multi-faktor Authentisierung:  
Kombinationen von mehreren Faktoren

# Was man hat

Authentication

Was man weiß

- z.B. Passwörter

Fdfjd#s3

Was man hat

- z.B. Chipkarte



## Vorteile

+einfach mitnehmbar

+nicht leicht zu duplizieren

## Nachteile

-einfach übertragbar

-einfach zu stehlen/verlieren

Weitere Faktoren:

Wo man ist, Was man tut

Mehrere Faktoren Authentisierung:

Kombinationen von mehreren Faktoren

### Vorteile

- +nicht übertragbar
- +individuell

### Nachteile

- oft leicht fälschbar
- unveränderbar
- Privacy-Probleme

Was man ist

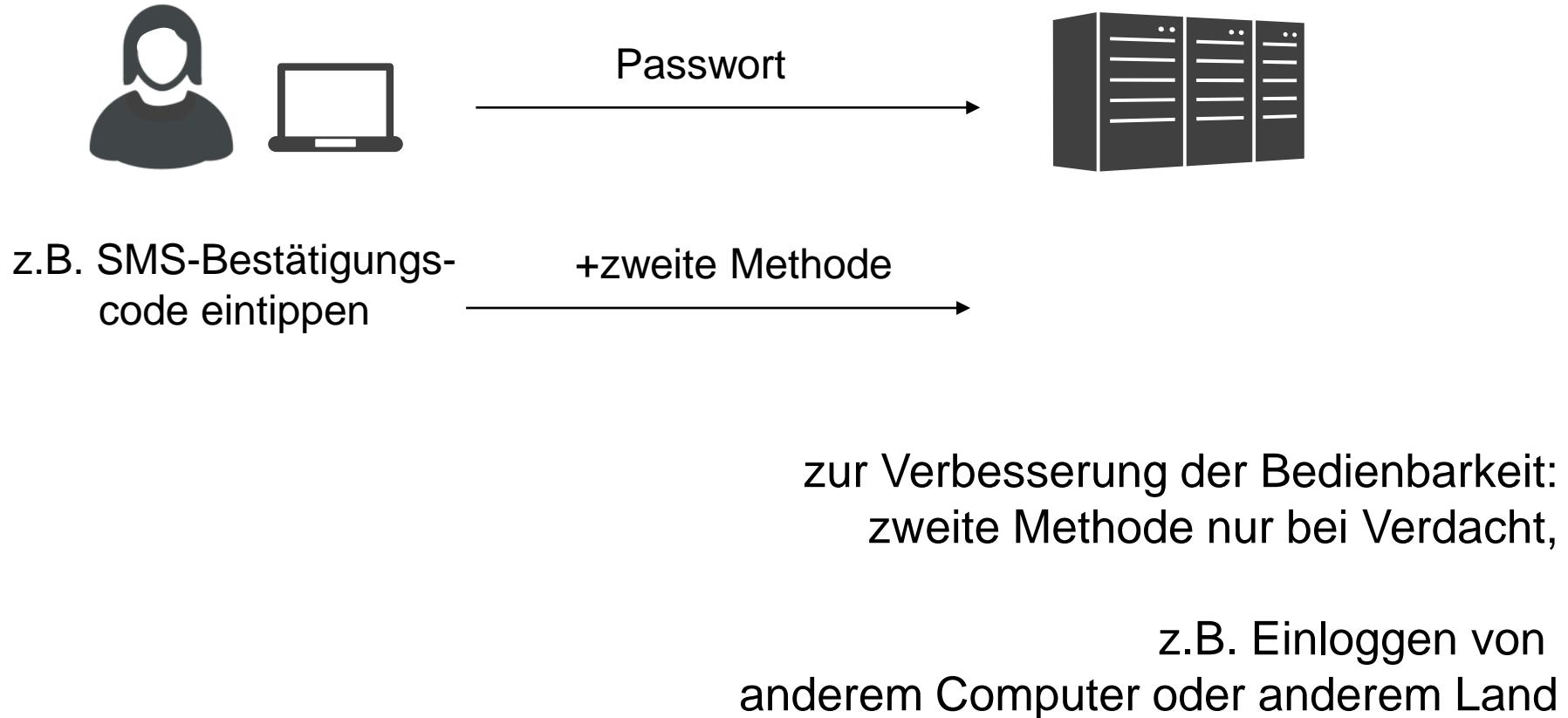
- z.B. biometrische Merkmale



Weitere Faktoren:  
Wo man ist, Was man tut

Multi-faktor Authentisierung:  
Kombinationen von mehreren Faktoren

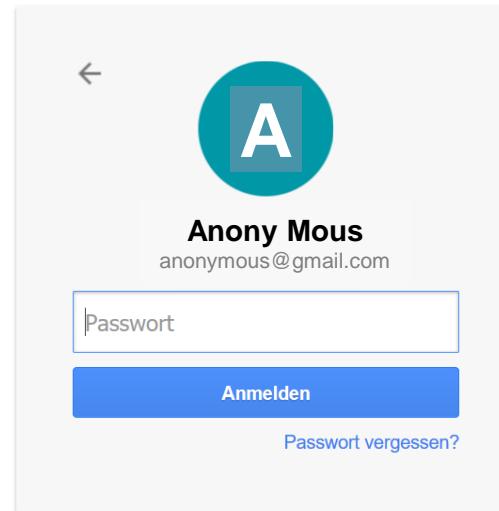
# Heute meistens 2-Faktor-Authentisierung



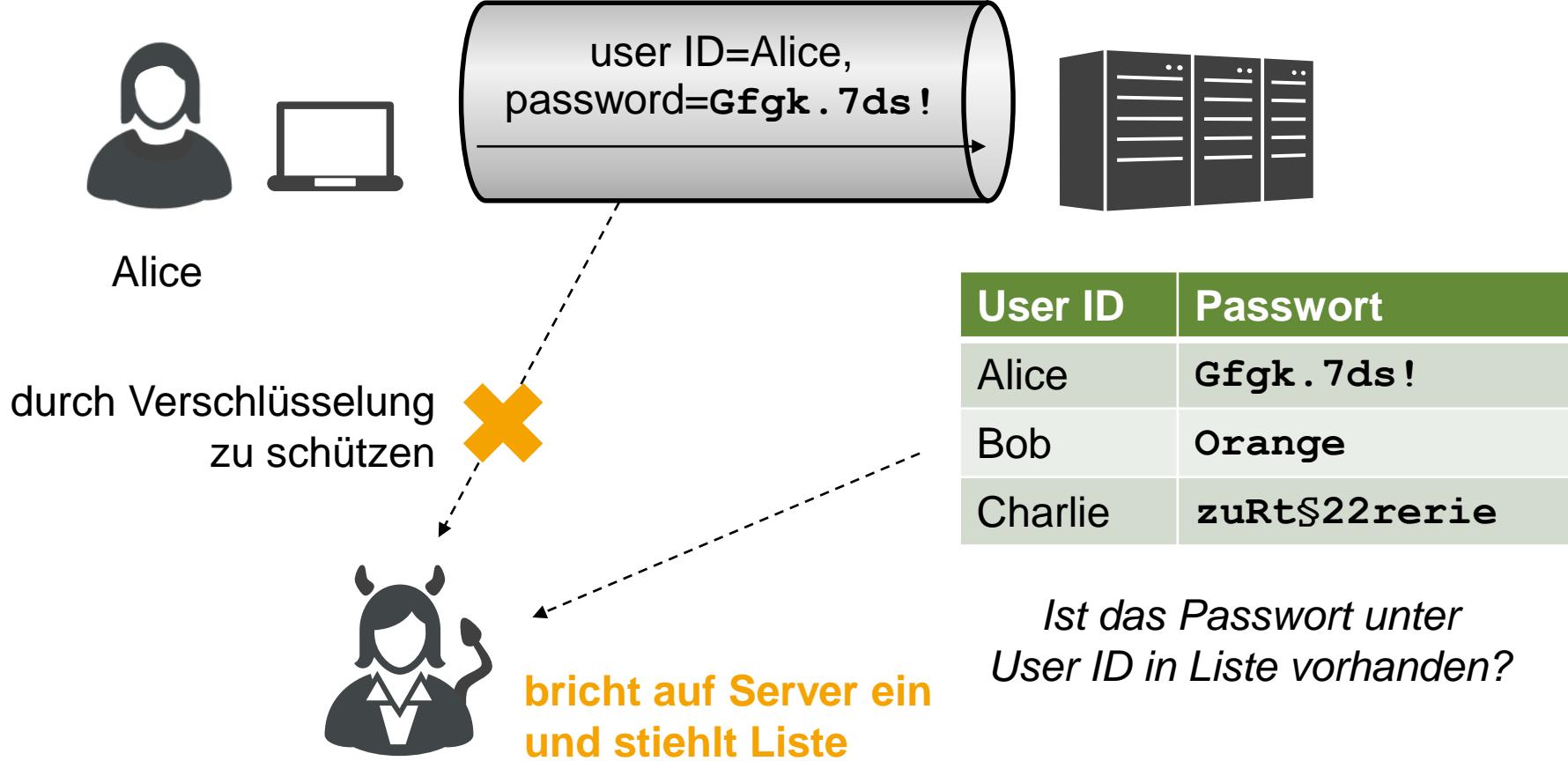
# Passwörter



Zum Hinzufügen eines weiteren Kontos anmelden



# Schlechte Lösung #1: Klartext



# Beispiel von 2016



The screenshot shows a news article from 'THE STACK' website. The header includes navigation links for NEWS, PARTNERS, EDUCATION, MAGAZINE, and ABOUT, along with a search bar and an advertisement for 'Electric'. The main content is under the 'SECURITY' category, with the title 'Russia's largest social network hacked – 100 million clear-text passwords stolen'. The article is attributed to 'Martin Anderson' on 'Mon 6 Jun 2016 11.21am'.

Quelle: The Stack

“The fact that the passwords were stored as plain-text ...  
is quite shocking for a community of this size and scope.”

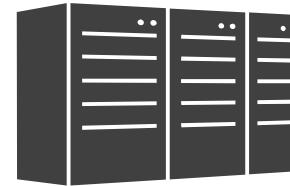
# Schlechte Lösung #2: Verschlüsselung

zusätzlich  
Schlüsselpaar  
(sk, pk)



Alice

user ID=Alice,  
password=Gfgk . 7ds !

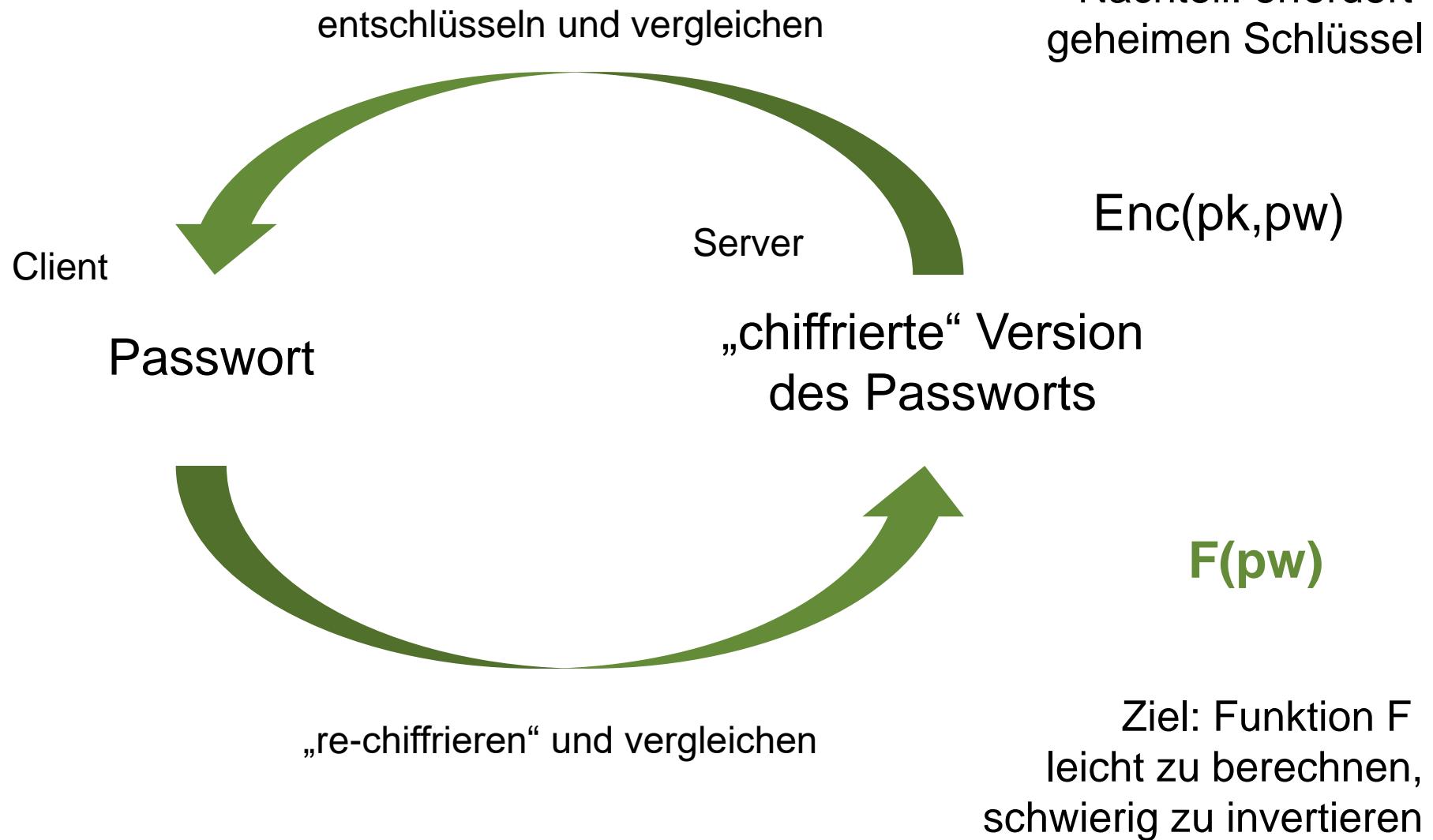


**bricht auf Server ein  
und stiehlt Liste  
und sk auf System**

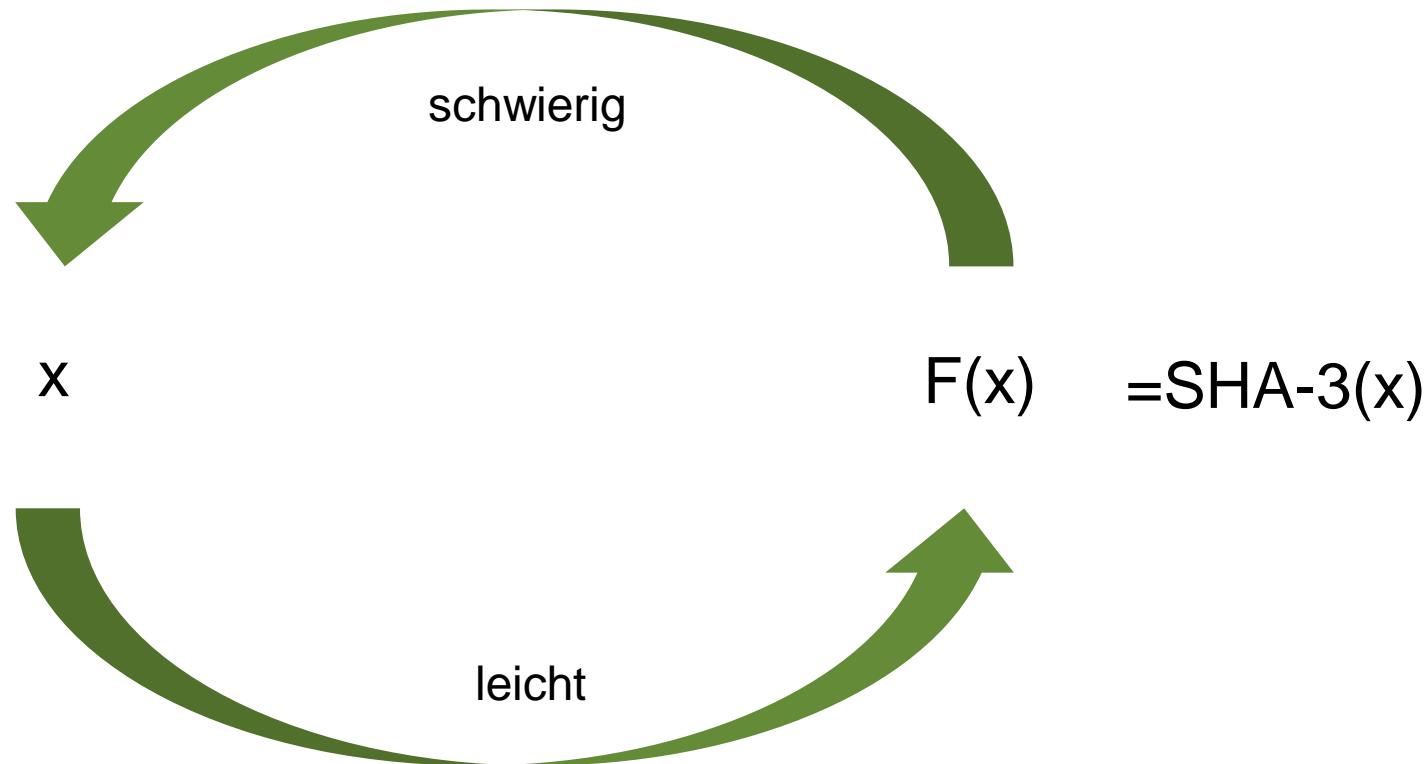
User ID	Passwort
Alice	E(pk , Gfgk . 7ds ! )
Bob	E(pk , Orange)
Charlie	E(pk , zuRt§22rerie)

*entschlüssle Passwort  
von User ID und vergleiche*

# Ansatz



# One-Way Funktionen

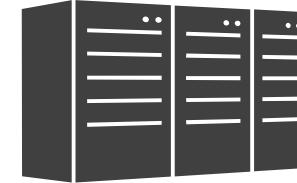


Hashfunktionen wie SHA-2, SHA-3  
gelten auch als One-Way-Funktionen

# Schlechte Lösung #3: Hashen



Alice



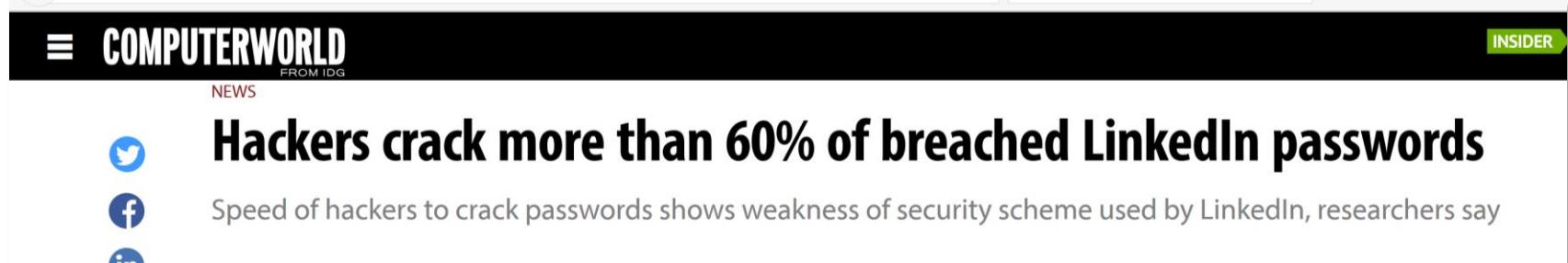
bricht auf Server ein  
und stiehlt Liste

User ID	Passwort
Alice	H (Gfgk . 7ds !)
Bob	H (Orange)
Charlie	H (zuRt\$22rerie)

Hashe Passwort  
und vergleiche

kann theoretisch Passwörter nicht rekonstruieren

# Beispiel von 2012



≡ COMPUTERWORLD  
FROM IDG  
NEWS  
INSIDER

**Hackers crack more than 60% of breached LinkedIn passwords**

Speed of hackers to crack passwords shows weakness of security scheme used by LinkedIn, researchers say

Quelle: Computerworld

“The breached LinkedIn member passwords were all hashed, or masked, using a hashing protocol known as SHA-1. Though SHA-1 offers a degree of protection against password cracking attempts, the protocol is by no means foolproof.”

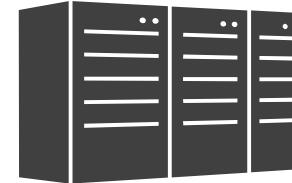
# Rainbow-Tables

Angreifer hat riesige vorberechnete Liste von (Ketten von) Passwörtern, die schnelle Suche unterstützt (Rainbow-Table)

Passwort	H(Passwort)
Apple	H (Apple)
Plum	H (Plum)
Orange	H (Orange)
...	...



bricht auf Server ein  
und stiehlt Liste



User ID	Passwort
Alice	H (Gfgk . 7ds ! )
Bob	H (Orange)
Charlie	H (zuRt\$22rerie)

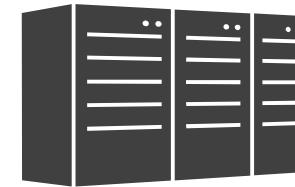
kann Passwörter rekonstruieren

# Lösung: Salted Hashing

Wählt zufälligen Salt S mit mindestens 64 Bits



Alice



User ID	Passwort
Alice	$H(S    Gfgk. 7ds!)$
Bob	$H(S    Orange)$
Charlie	$H(S    zuRtS22rerie)$



bricht auf Server ein und stiehlt Liste und Salt S

Hashe Passwort mit S und vergleiche

Hat wahrscheinlich keinen passenden Rainbow-Table für S

# Weitere Gegenmaßnahmen

Individuell: für jeden User eigenen Salt

Peppering: Salt(s) geheim halten

Iterationen: Hashfunktion iterieren  $h_{j+1} = H(\text{pw}, S, h_j)$  für  $j=1, 2, 3, \dots$ , um Aufwand für Angreifer merklich zu erhöhen

Beispiel: Braucht 100ms statt 1 $\mu$ s beim Einloggen, aber Angreifer braucht 270 Jahre statt 1 Tag zum Brechen

Ansatz wird von Kandidaten PBKDF2, bcrypt verwendet (einige tausend Iterationen)

aber nicht von Microsofts (unsalted) LM- und NT(LM) v1/v2

# Beispiel: Unix/Linux-Passwortdatei

Format von Einträgen in **/etc/passwd** :

Benutzername : Passwort : UserID : GroupID : Info : Heimatverzeichnis : Shell

```
root:x:1:1:::/bin/sh
lp:*:11:11::/var/spool/lp:/bin/false
boss:x:666:666:hq:/home/boss:/usr/bin/ksh
guest:x:1001:1000:no-office:/home/guest:/bin/sh
```



\* gesperrt;  
x ausgelagert in **/etc/shadow**,  
da **/etc/shadow** nur für bestimmte Prozesse lesbar,  
aber **/etc/passwd** für alle User/Prozesse

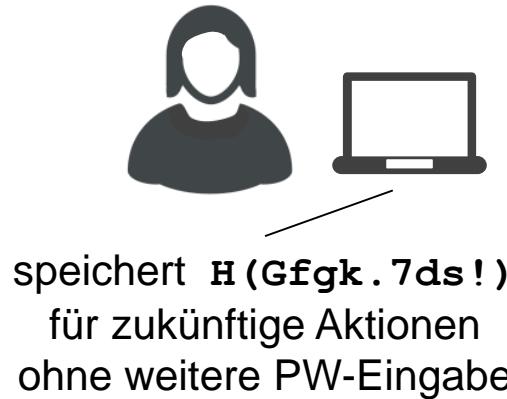
Format von Einträgen in **/etc/shadow** :

Name : \$Hashalgo\$Salt\$Passwort : LastChange: Min: Max: Warn: Inactive: Expire

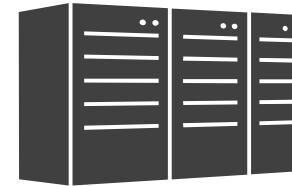
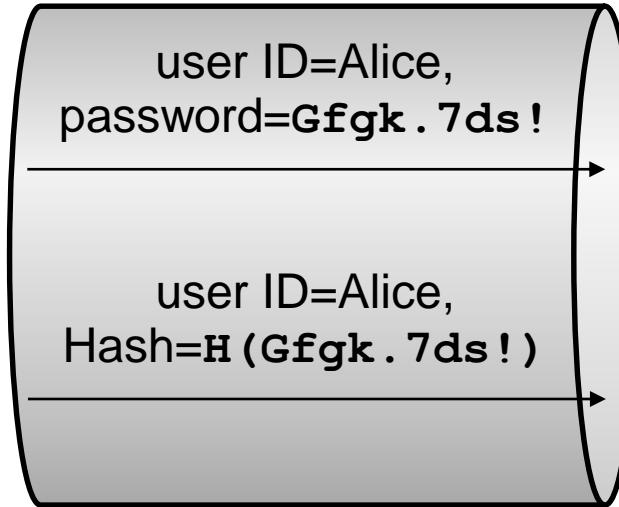
```
root:$1$TrIOigLp$PU00HaiLS15UY3CMVC0/g0:12375:0:99999:3:::
```

...

# Pass-the-Hash-Angriffe (PtH)



unter Windows 10  
z.B. durch „Credential  
Guard“ erschwert



akzeptiert auch  
ge-haschte Passwörter

user ID=Alice,  
Hash=H(Gfgk . 7ds !)

bricht bei Client ein  
und stiehlt Hashwert



Beschreiben Sie den Unterschied zwischen der One-Way-Eigenschaft und der Kollisionsresistenz einer Hashfunktion.

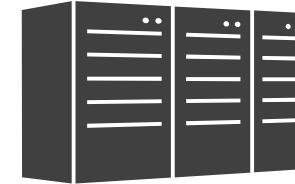


Warum verwendet man nicht ein Public-Key-Verschlüsselungssystem zum Hashen,  $H(\text{pw}):=\text{Enc}(\text{pk}, \text{pw})$ , und „wirft“ den geheimen Schlüssel  $\text{sk}$  weg?



Warum ist iteriertes Passwort-Hashing  $h_{j+1}=H(\text{pw}, S, h_j)$  besser als folgende Lösung:  
 $\text{Hash}(\text{pw}, S) = H(\text{pw}, S, 1) \oplus H(\text{pw}, S, 2) \oplus \dots \oplus H(\text{pw}, S, n)$

# Sicheres Verwalten ist das eine...



Passwort muss gut gewählt werden...

...sonst: (1) direktes Einloggen möglich,

user ID=Alice,  
PW=einfachesPW



oder (2) Angreifer findet Password schneller in erhaltener Liste

# Default-Passwörter

sind allgegenwärtig



**SPIEGEL ONLINE** DER SPIEGEL SPIEGEL TV Suche Anmelden

≡ NETZWELT Schlagzeilen | Wetter | DAX 10.696,19 | TV-Programm | Abo

Nachrichten > Netzwelt > Web > Hacker > Störung bei Twitter, Netflix & Spotify: Haushaltsgeräte legten Webseiten lahm

**Angriff auf Twitter und Co.**  
**Cyberattacke aus dem Babyfon**

Am Freitag waren Twitter, Netflix und Spotify stundenlang unerreichbar. Hinter der Störung steckt ein Angriff, bei dem internetfähige Haushaltsgeräte missbraucht wurden. Kurz vor der Wahl löst dieser Trend vor allem in den USA Besorgnis aus.



Quelle: Spiegel Online (10/2016)

Geräte haben einfache vorinstallierte Passwörter, die nicht geändert werden

admin mit <keins>  
guest mit 0000  
<keins> mit <keins>

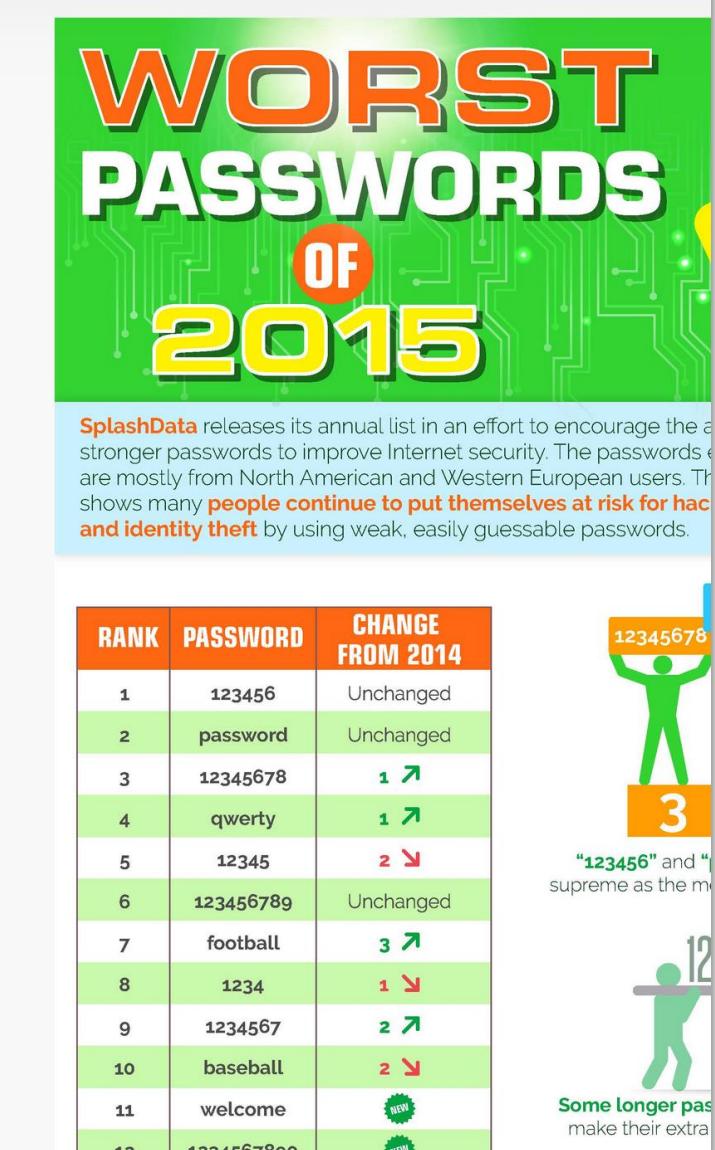
...

# Schwache Passwörter

## sind allgegenwärtig

- 0.5% of users have the password *password*;
- 0.4% have the passwords *password* or *123456*;
- 0.9% have the passwords *password*, *123456* or *12345678*;
- 1.6% have a password from the top 10 passwords
- 4.4% have a password from the top 100 passwords
- 9.7% have a password from the top 500 passwords
- 13.2% have a password from the top 1,000 passwords
- 30% have a password from the top 10,000 passwords

Quelle: xato.net (2011)

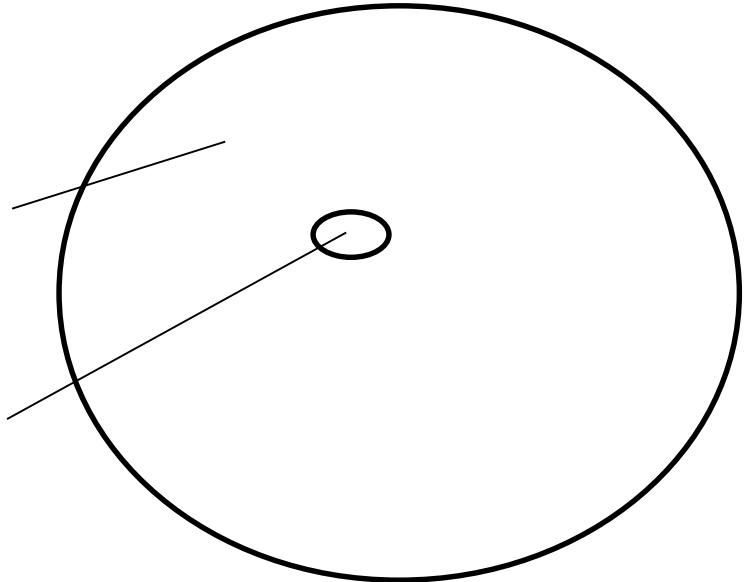


Quelle: SplashData

# Stärke von Passwörtern?

alle Passwörter mit 8 Stellen  
(64 Bits Entropie)

„menschliche“ Passwörter



Entropie = Maß für Information

$n$  Bits Entropie bedeutet, dass Angreifer  $2^n$  Möglichkeiten versuchen muss

NIST Special Publication 800-63 (von 2004)

8-stellige Passwörter mit Spezialsymbolen haben ca. 20-30 Bits Entropie  
(Berechnung ist allerdings umstritten)

# Gute Passwörter?

**Top Left Panel:** A complex password is shown as a tree diagram. The root node is labeled 'UNCOMMON (NON-GIBBERISH) BASE WORD'. It branches into 'CAPS?' (with a small box icon), 'COMMON SUBSTITUTIONS' (with a small box icon), 'ORDER UNKNOWN' (with a small box icon), 'NUMERAL' (with a small box icon), and 'PUNCTUATION' (with a small box icon). The password itself is 'Tr0ub4dor &3'. Below the tree is the text: '(YOU CAN ADD A FEW MORE BITS TO ACCOUNT FOR THE FACT THAT THIS IS ONLY ONE OF A FEW COMMON FORMATS.)'

**Top Middle Panel:** Entropy and cracking time are discussed. It shows a 5x5 grid of boxes. The text reads: '~28 BITS OF ENTROPY' and ' $2^{28} = 3$  DAYS AT 1000 GUESSES/SEC'. Below this is the note: '(PLAUSIBLE ATTACK ON A WEAK REMOTE WEB SERVICE. YES, CRACKING A STOLEN HASH IS FASTER, BUT IT'S NOT WHAT THE AVERAGE USER SHOULD WORRY ABOUT.)'

**Top Right Panel:** A stick figure says: 'WAS IT TROMBONE? NO, TROUBADOR. AND ONE OF THE 0s WAS A ZERO? AND THERE WAS SOME SYMBOL...' Below the figure is the text: 'DIFFICULTY TO REMEMBER: HARD'.

**Bottom Left Panel:** A simple password is shown as a tree diagram. The root node is 'correct horse battery staple'. It branches into 'FOUR RANDOM COMMON WORDS' (with a 4x4 grid of box icons). Below the tree is the text: '(YOU CAN ADD A FEW MORE BITS TO ACCOUNT FOR THE FACT THAT THIS IS ONLY ONE OF A FEW COMMON FORMATS.)'

**Bottom Middle Panel:** Entropy and cracking time are discussed. It shows a 5x5 grid of boxes. The text reads: '~44 BITS OF ENTROPY' and ' $2^{44} = 550$  YEARS AT 1000 GUESSES/SEC'. Below this is the note: '(PLAUSIBLE ATTACK ON A WEAK REMOTE WEB SERVICE. YES, CRACKING A STOLEN HASH IS FASTER, BUT IT'S NOT WHAT THE AVERAGE USER SHOULD WORRY ABOUT.)'

**Bottom Right Panel:** A stick figure says: 'THAT'S A BATTERY STAPLE. CORRECT!' Below the figure is the text: 'DIFFICULTY TO REMEMBER: YOU'VE ALREADY MEMORIZED IT'.

**Bottom Center Panel:** A summary statement: 'THROUGH 20 YEARS OF EFFORT, WE'VE SUCCESSFULLY TRAINED EVERYONE TO USE PASSWORDS THAT ARE HARD FOR HUMANS TO REMEMBER, BUT EASY FOR COMPUTERS TO GUESS.'

Quelle: [xkcd.com](http://xkcd.com)

# Schlechte Passwörter vermeiden!

**keine** kurzen Passwörter mit weniger als 8 Stellen

**nicht** mehrfach verwenden

**keine** persönlichen Daten wie Username, Geburtstag,...

**keine** einfachen Wörter aus Dictionaries

**keine** einfachen Transformationen wie **passw0rd**

lokale Tests beim Erstellen können  
viele dieser Regeln „erzwingen“

Passwortmanager helfen beim  
Verwalten komplizierter Passwörter

# Passwörter testen



OpenSource-Pakete zum Testen von Passwörtern:  
**hashcat**,  
**John the Ripper**,

...

verschiedene Modi:

Brute-Force (alle Kombinationen)  
bekannte Wörter/Dictionaries  
Kombinationen wie **passw0rd**  
Regeln

...

```
watchdog: temperature return trigger set to 75C
82a9ddda829eb7f8ffe9fbe49e45d47d2dad9664fb...:hashcat

Session.Name...: hashcat
Status.....: Cracked
Input.Mode...: Mask (?a?a?a?a?a?a?a) [7]
Hash.Target...: 82a9ddda829eb7f8ffe9fbe49e45d47d2dad9664fb...
Hash.Type....: SHA512
Time.Started...: Fri Aug 19 13:29:37 2016 (4 secs)
Speed.Dev.#1...: 1042.6 MH/s (96.54ms)
Speed.Dev.#2...: 1042.9 MH/s (98.27ms)
Speed.Dev.#3...: 31454.7 kH/s (84.55ms)
Speed.Dev.#*...: 2116.9 MH/s
Recovered.....: 1/1 (100.00%) digests, 1/1 (100.00%) salts
Progress.....: 8888422400/2799360000000 (0.32%)
Rejected.....: 0/8888422400 (0.00%)
Restore.Point..: 0/12960000 (0.00%)
HwMon.Dev.#1...: Temp: 68c Fan: 42% Util:100% Core:1821Mhz Mem:
HwMon.Dev.#2...: Temp: 66c Fan: 33% Util:100% Core:1847Mhz Mem:
HwMon.Dev.#3...: N/A

Started: Fri Aug 19 13:29:37 2016
Stopped: Fri Aug 19 13:29:47 2016
```

Quelle: hashcat

# Eine Frage der Sicherheit



## Hacker impersonated Sarah Palin to hijack e-mail password



BY THE ASSOCIATED PRESS

Thursday, September 18, 2008, 4:03 PM

**WASHINGTON** - Details emerged Thursday behind the break-in of Republican vice presidential candidate [Sarah Palin](#)'s e-mail account, including a first-hand account suggesting it was vulnerable because a hacker was able to impersonate her online to obtain her password.

Abfragen bei vergessenen Passwörtern oft nicht sicher

Quelle:  
Associated Press (2008)

“The hacker guessed that Alaska's governor had met her husband in high school, and knew Palin's date of birth and home Zip code. Using those details, the hacker tricked Yahoo Inc.'s service into assigning a new password,...”

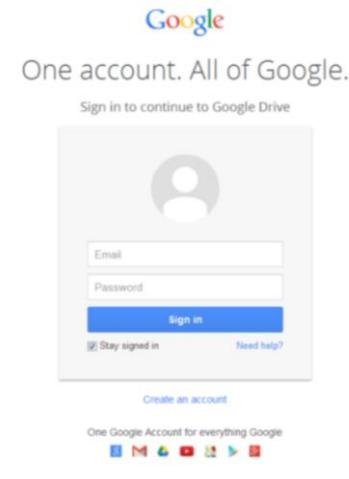
# Phishing for Passwords



(2) sendet gefälschte E-Mail

(1) setzt auf

(4) erhält Passwort



Preview unter Google Drive  
(gültige SSL-Verbindung zu Google!)



Thunderbird

Link auf Google Docs

Quelle:  
Symantec (2014)

(3) folgt Link und  
gibt Passwort preis

# Gone Phishing



“...an ‘urgent email password change request’ had a 28% average click rate.”

Quelle: Wombat: The State of the Phish (2016)

speziell „Spear Phishing“  
(gezielt z.B. als Vorgesetzter ausgeben)  
sehr erfolgreich



Ist `#gfFhH5/Un1.gg&` ein gutes Passwort?

`date` erzeugt das Datum auf Sekundengenauigkeit



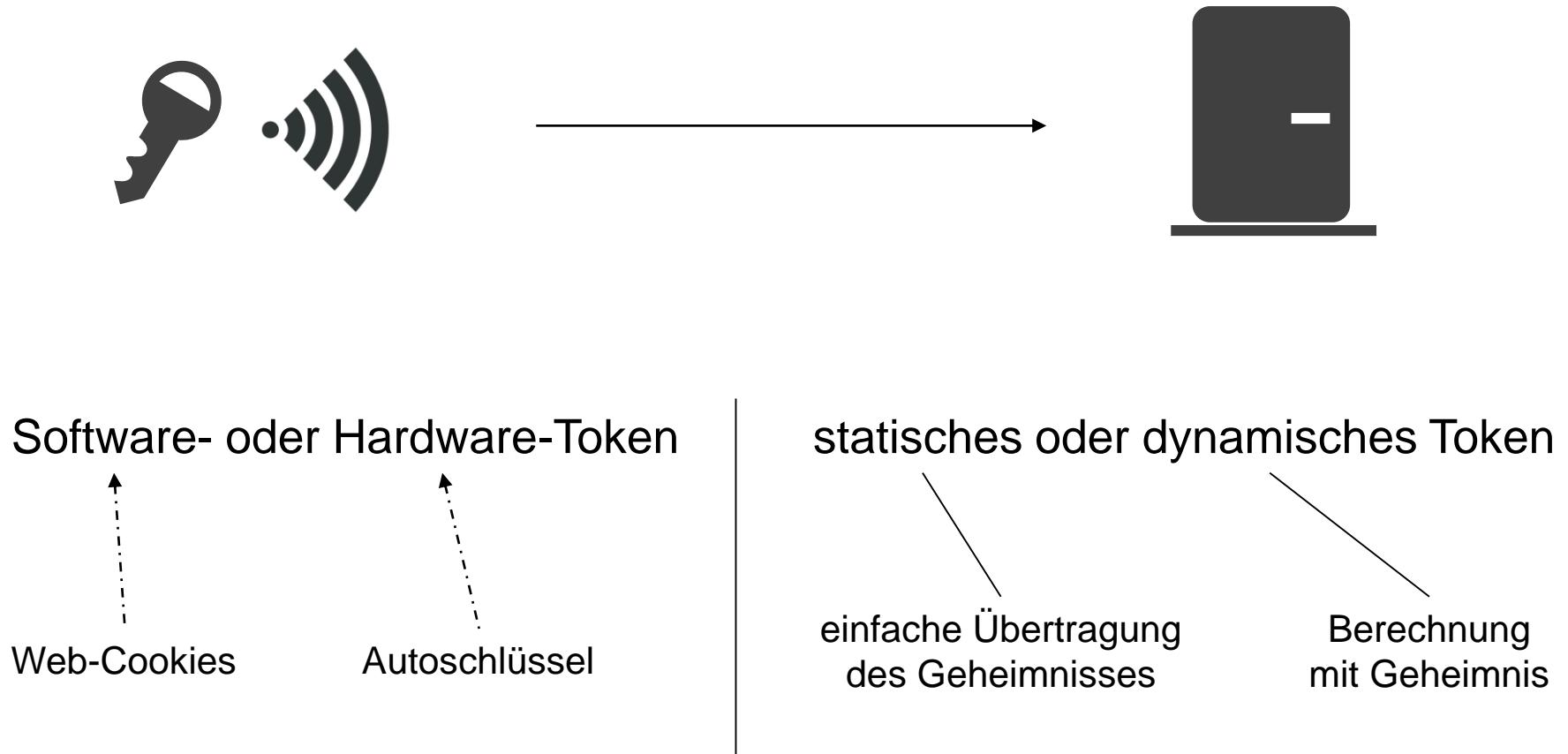
Was halten Sie von folgender Vorgehensweise, um Passwörter zu erzeugen: `date | md5sum` ?  
Wie viel Entropie erwarten Sie?



Welche 2-Faktor-Authentisierungsmethoden kennen Sie noch?

# Was man hat (Tokens)

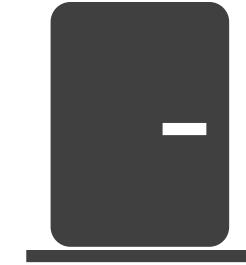
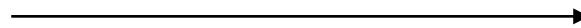
# Token-basierte Authentisierung



# Replay-Angriffe bei statischen Tokens



(Hash von) Geheimnis



(Hash von) Geheimnis



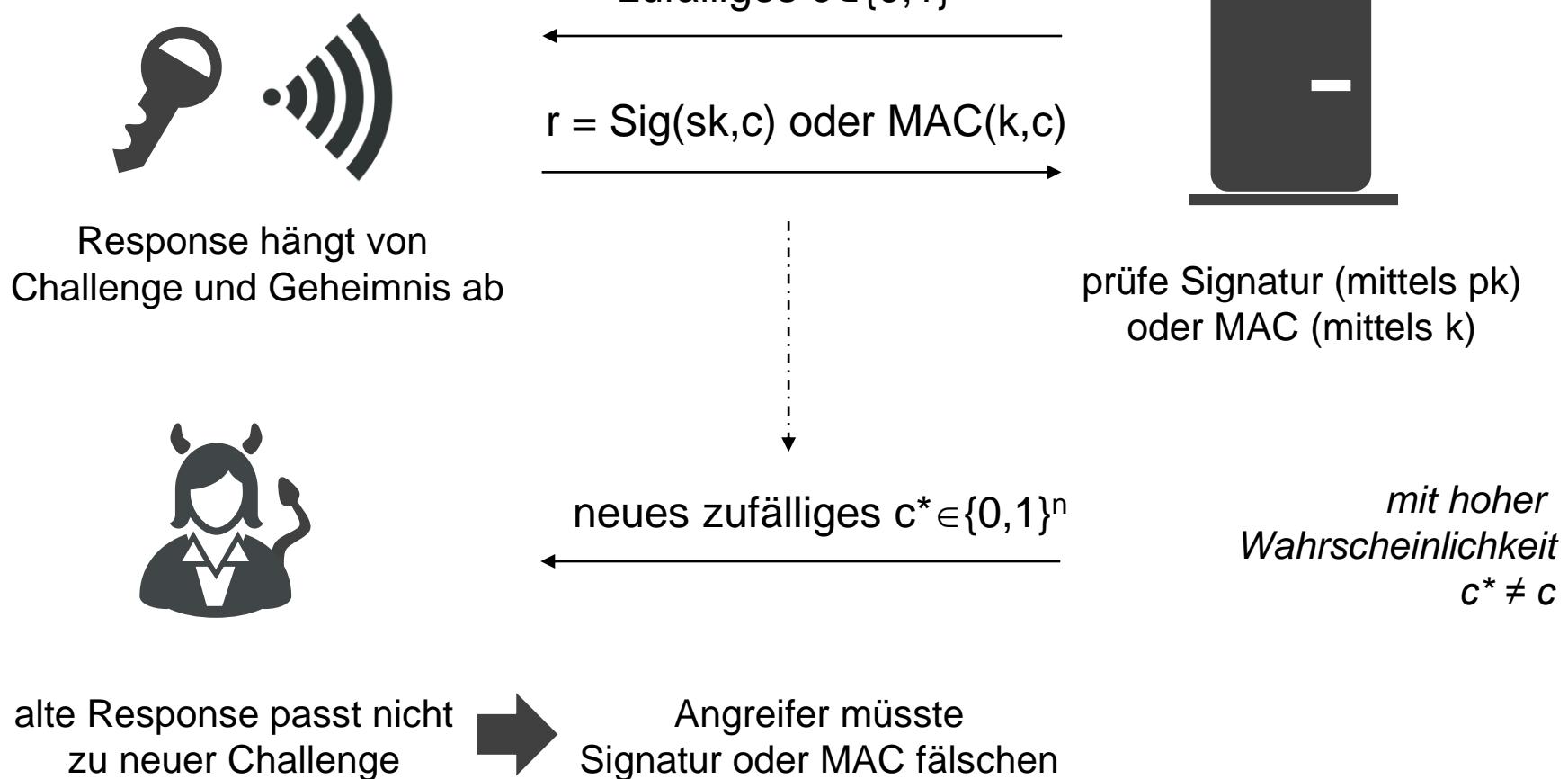
Absichern der Übertragung des Geheimnisses oft zu teuer

# Challenge-Response bei dynamischen Tokens



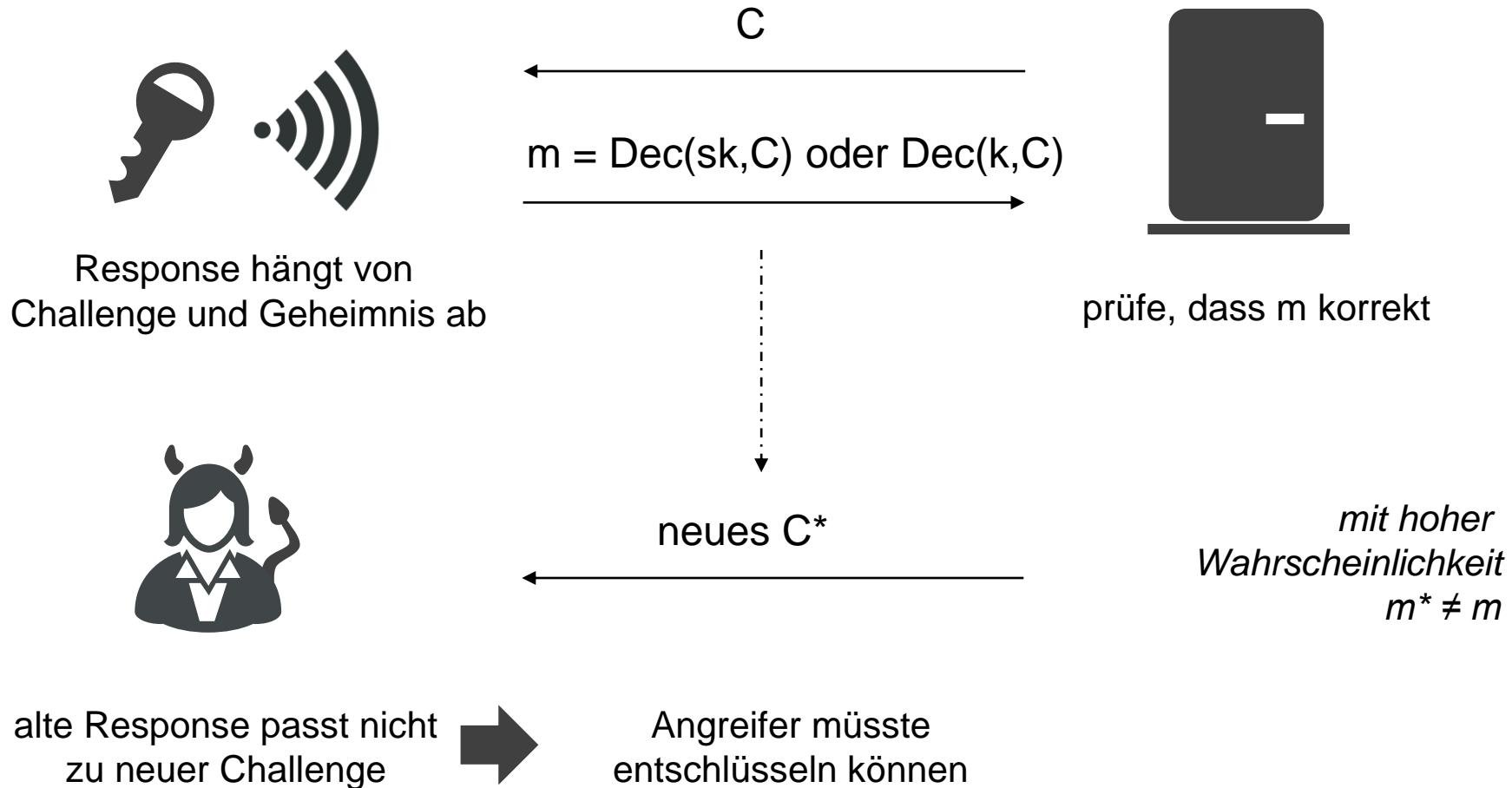
# Implementierung per Signaturen

$n$  meistens 64, 128 oder 256



# Implementierung per Verschlüsselung

wähle  $m \in \{0,1\}^n$  zufällig  
 $C = E(pk, m)$  oder  $E(k, m)$



# Beispiel neuer Personalausweis

Extended Access Control (vereinfacht!)

$sk_C, pk_C$  mit Zertifikat



prüfe Zertifikat

wähle zufälliges  $r$

prüfe Signatur

Challenge-Response mit Sig

prüfe  $epk_T$   
bilde  $K = DH(sk_C, epk_T)$   
 $t = MAC(K, epk_T)$

Challenge-Response mit MAC

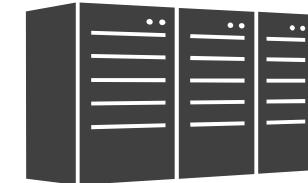
Berechtigungszertifikat  
für Signaturschlüssel  $pk_T$

Compr( $epk_T$ )

$r$

$s$

$sk_T, pk_T$  mit Zertifikat



wähle DH-Schlüsselanteil  
( $esk_T, epk_T$ )

$s = Sig(sk_T, r, Compr(epk_T))$

$pk_C$  mit Zertifikat

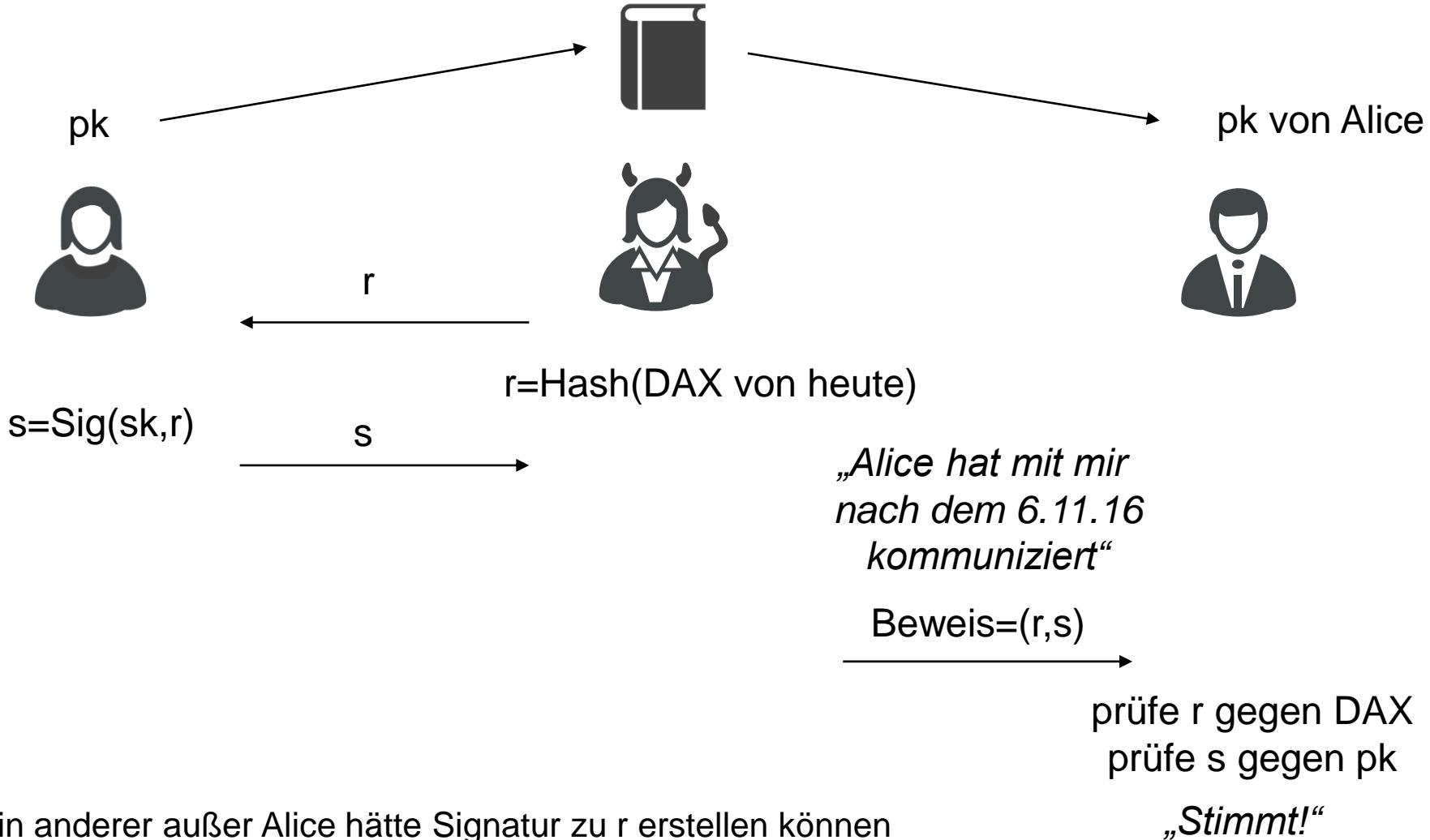
$epk_T$

$t$

prüfe Zertifikat

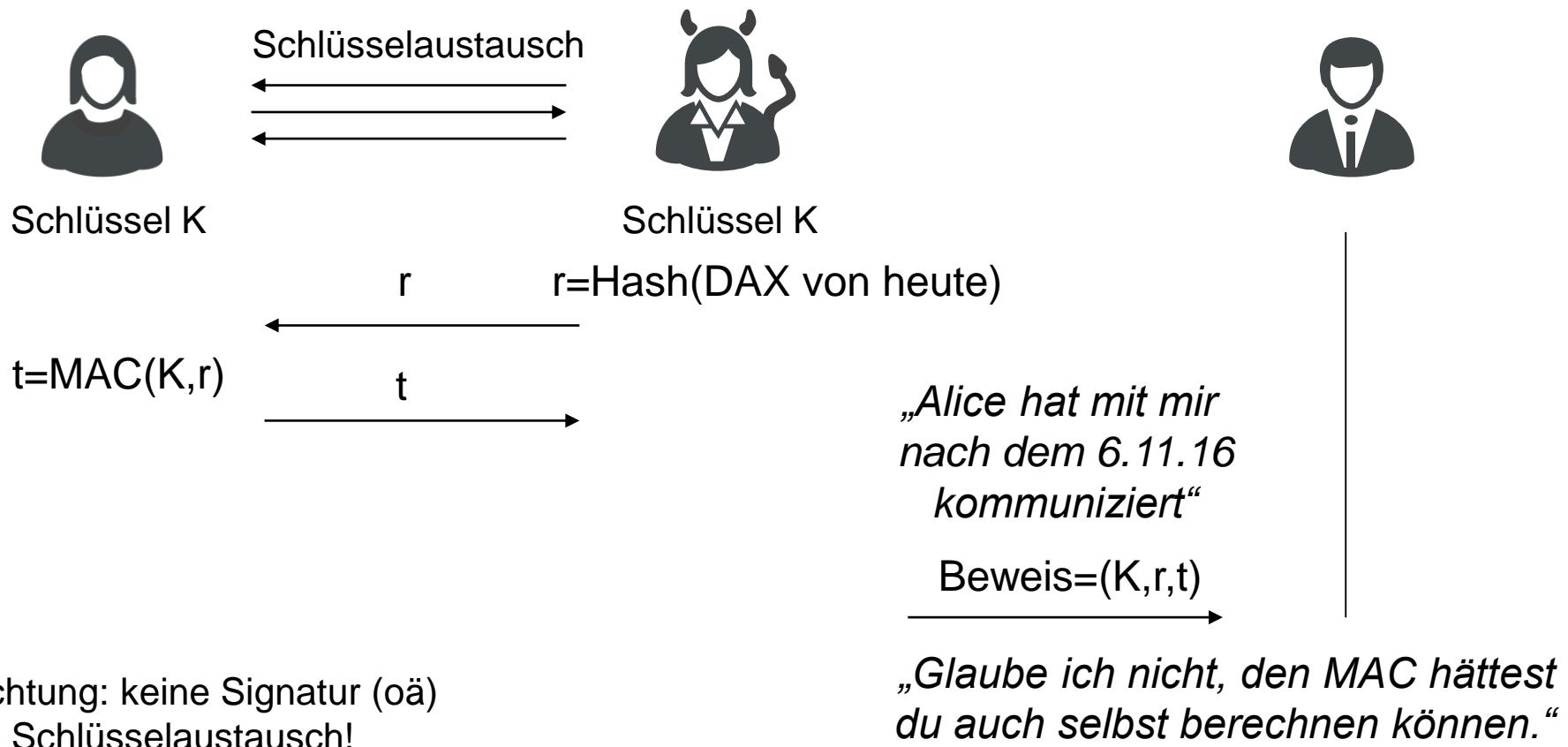
bilde  $K = DH(esk_T, pk_C)$   
prüfe MAC  $t$

# Nicht-Abstreitbarkeit von Signaturen in Challenge-Response-Verfahren



# Abstreitbarkeit von MACs

in Challenge-Response-Verfahren



# Biometrie (Was man ist)

# Biometrische Systeme

Fingerabdrücke

Retina und Iris

Gesichtserkennung

Handschrift

Sprache

...

Face Recognition  
+Fingerprints



Face Recognition

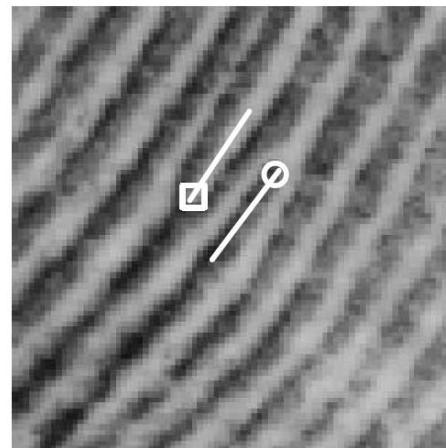
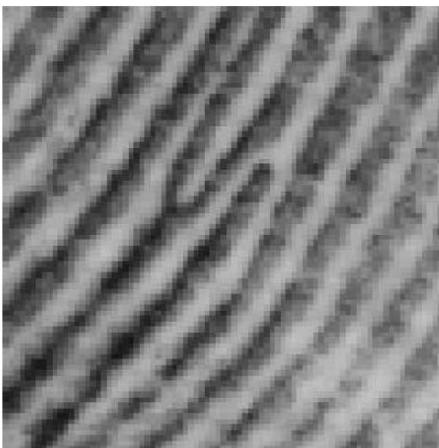


Quelle: US Customs  
& Border Protection

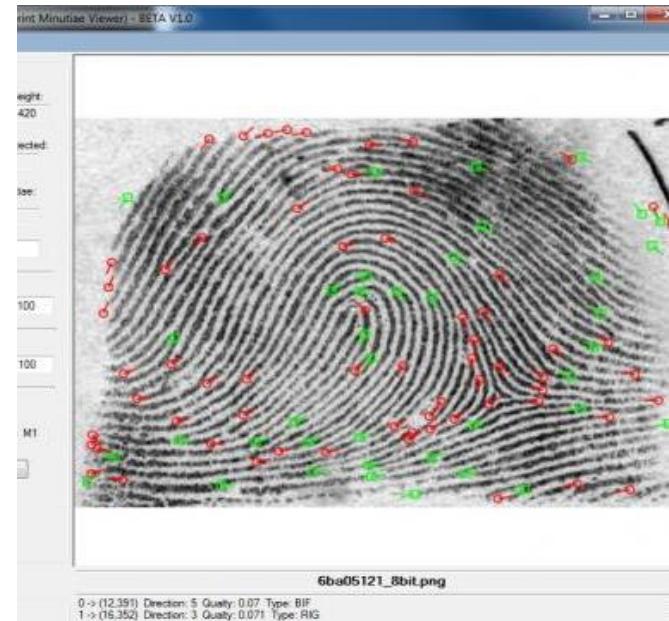
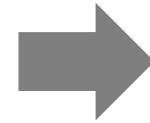
Quelle: easypass

# Fingerabdruck-Minutien (feine Merkmale)

Bifurcation (Verzweigung) Ridge Ending (Endpunkte)



Quelle: NIST Special Database 27



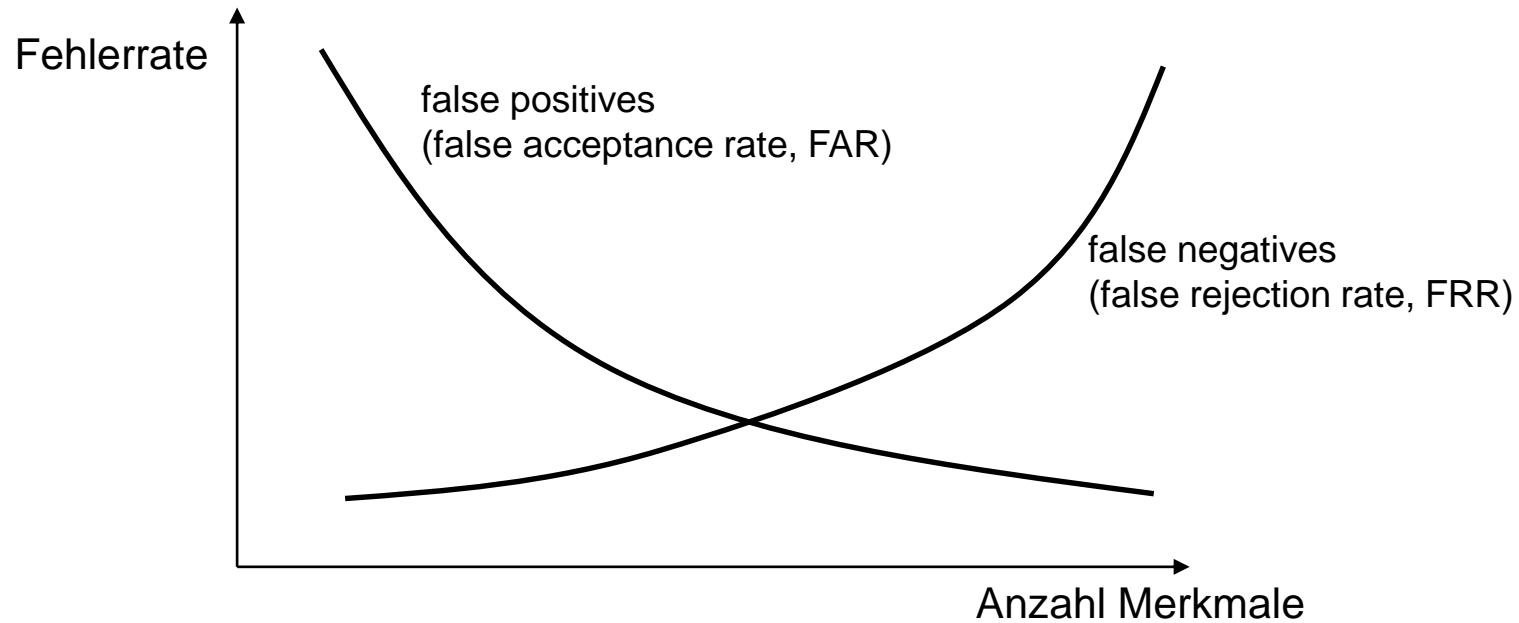
Quelle: NIST Fingerprint Minutiae Viewer (FpMV)

eventuell auch Merkmale wie Schleifen, Deltas oder Wirbel

# Fehler

false positives: wird akzeptiert, ist aber nicht die Person

false negatives: wird verworfen, obwohl die richtige Person



# Probleme Biometrischer Authentisierung

nicht widerrufbar

benötigt vertrauenswürdige Geräte vor Ort

oft leicht zu fälschen (Bild statt Gesicht, Fingerattrappe,...)

Hackers threaten to publish fingerprints



Quelle: Sidney Morning Herald

guten Tag, mein Name ist  
Dr. von der Leyen



Quelle: heise.de



Beschreiben Sie Challenge-Response mit Private-Key-Verschlüsselung.



Was ist bei folgendem 1-Runden Challenge-Response-Verfahren problematisch? Alice wählt sich ein zufälliges  $r$  selbst und sendet  $s = \text{Sig}(sk, r)$ .

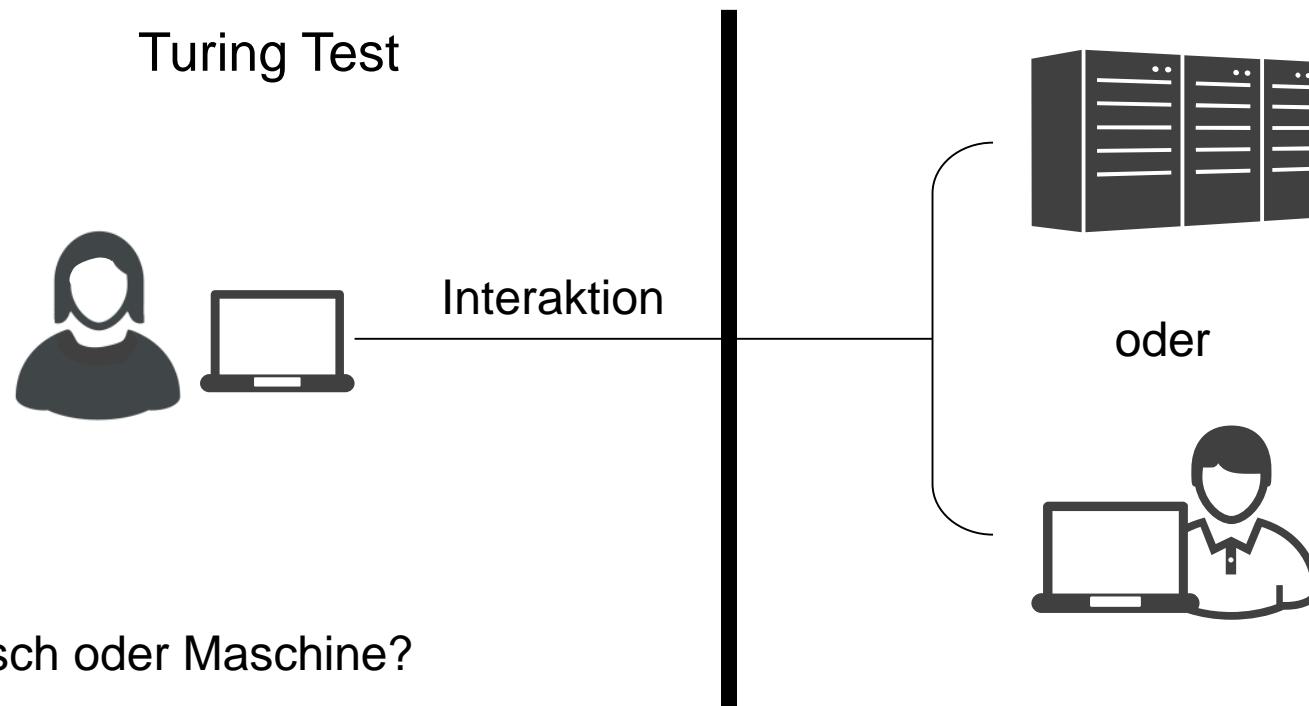


Benutzen Sie biometrische Authentisierung?

# CAPTCHAs

# CAPTCHAs

Completely Automated Public Turing test to Tell Computers and Humans Apart



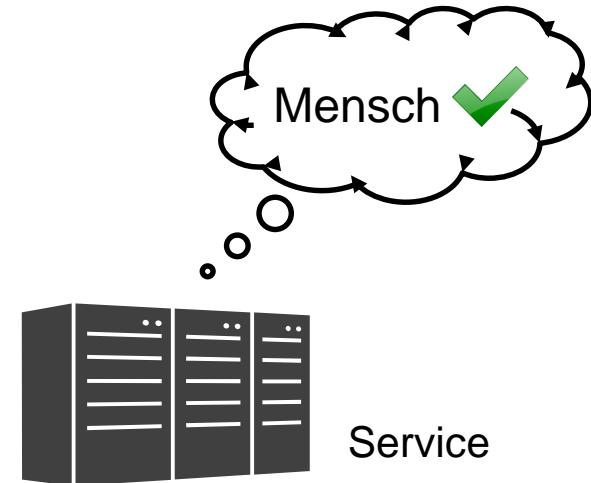
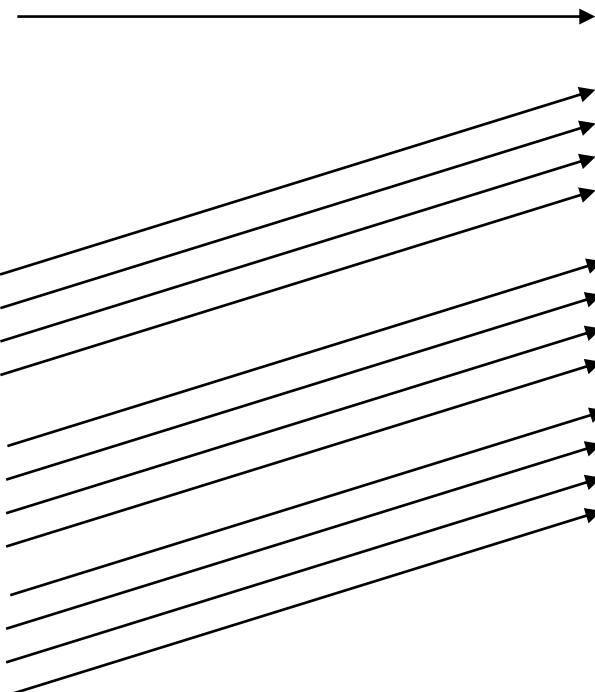
# Anwendung CAPTCHAs

autorisierte  
User

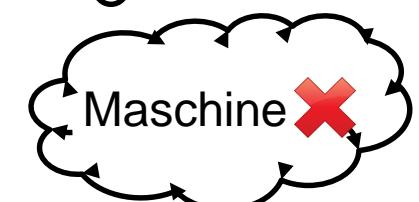


sollen DoS-Angriff verhindern

DoS-Angreifer



Service



# CAPTCHAs

Beispiel:  
Googles reCAPTCHA  
(ca. 2008)



Quelle: Wikipedia

reCAPTCHA änderte Aussehen mit Angriffen  
(z.B. Linie durch Buchstaben, Farbfleck, Hausnummern)



Figure 4: Examples of images from the hard CAPTCHA puzzles dataset.

Quelle: ZDnet



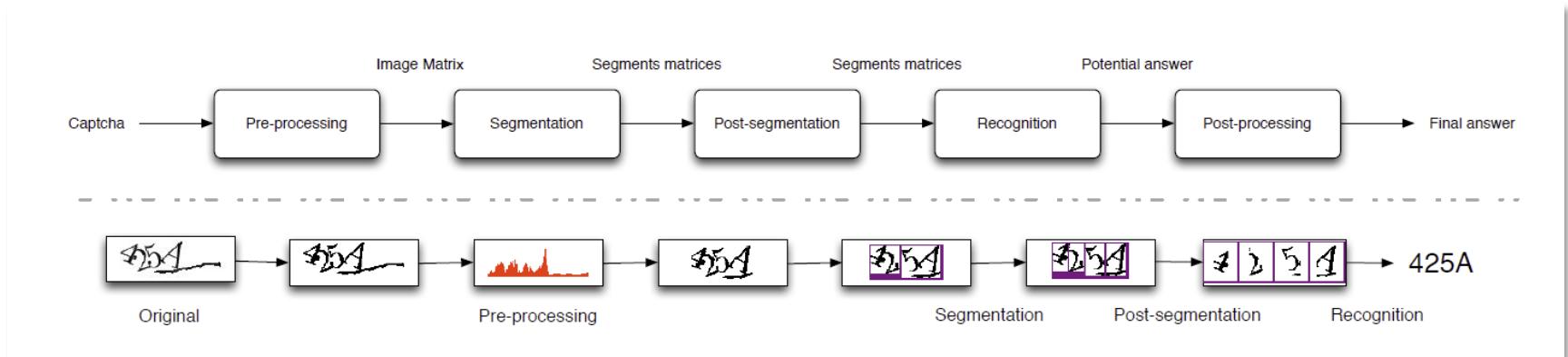
Quelle: ZDnet

auch Audio-CAPTCHAs für Anwender mit Seh Einschränkung  
(meist „verrauschte Ansage von Buchstaben“)

# CAPTCHAs werden gebrochen



Von Hand, für spezielles CAPTCHA, per Segmentierung und Recognition:



Quelle: Burzstein et al., The End is Nigh: Generic Solving of Text-based CAPTCHAs

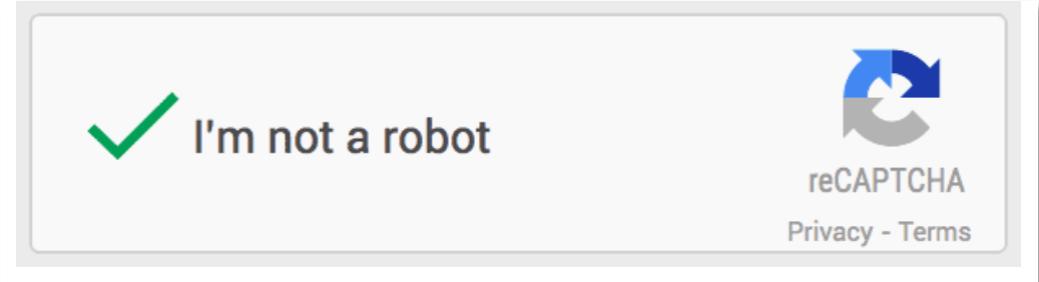
oder

billige Online-Services,  
um CAPTCHAs zu lösen



# Angenehmere CAPTCHAs

Letzte Version:  
Googles  
noCAPTCHA reCAPTCHA



Quelle: reCAPTCHAs

Unterscheidung per Browser-Informationen, Cookies, Mausbewegungen,...

nur bei zu hohem Risiko noch ein „ursprüngliches“ CAPTCHA

# Angriffe heute



## Allgemeine Machine-Learning-Ansätze

### I Am Robot: (Deep) Learning to Break Semantic Image CAPTCHAs

Suphanee Sivakorn, Iasonas Polakis and Angelos D. Keromytis

*Department of Computer Science*

*Columbia University, New York, USA*

[suphanee](mailto:suphanee@cs.columbia.edu), [polakis](mailto:polakis@cs.columbia.edu), [angelos](mailto:angelos@cs.columbia.edu)@cs.columbia.edu

*Abstract*—Since their inception, captchas have been widely used for preventing fraudsters from performing illicit actions. Nevertheless, economic incentives have resulted in an arms race, where fraudsters develop automated solvers and, in turn, captcha services tweak their design to break the solvers. Recent work, however, presented a generic attack that can be applied

accounts and posting of messages in popular services. According to reports, users solve over 200 million reCaptcha challenges every day [2]. However, it is widely accepted that users consider captchas to be a nuisance, and may require multiple attempts before passing a challenge. Even simple challenges deter a significant amount of users from

Bild-basierte CAPTCHAs per Deep Learning gebrochen  
auch noCAPTCHA reCAPTCHA betroffen

# Autorisierung

# Autorisierung und Zugriffskontrolle

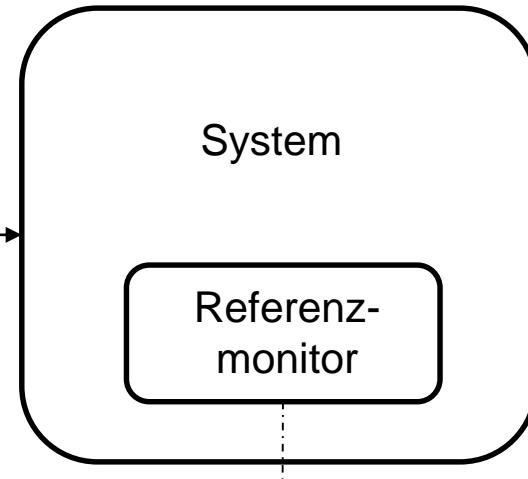
Autorisierung:  
lege fest, was Subjekt darf



Subjekt

Zugriffsmodus  
(Lesen, Schreiben,  
Löschen, Erzeugen,...)

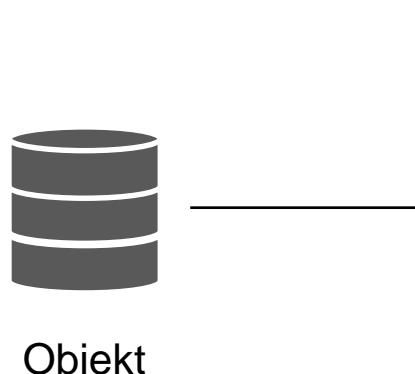
Access Control:  
setzt Zugriffsregeln um



stets aktiv  
nicht manipulierbar  
korrekt

erlaubt/lehnt ab

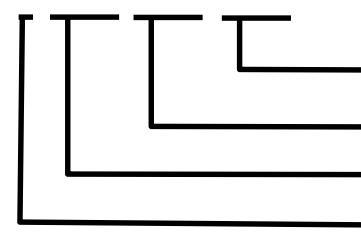
# Zugriffskontrolle via Access Control Lists (ACLs)



Objekt	User ID	Zugriffsmodus	allow/deny
	Alice	Lesen	Allow
	Alice	Schreiben	Deny
	Alice	Ausführen	Deny
	Bob	Lesen	Allow
	...	...	...

Beispiel: Dateirechte bei Unix-basierten Systemen:

`-rwxr-x---` **userA groupB ... file.txt**



- Rechte für andere (hier: nichts)
- Rechte für Gruppe (hier: Lesen, Ausführen)
- Rechte für Besitzer (hier: Lesen, Schreiben, Ausführen)
- Dateityp (z.B. `l` für Links)

# Modelle für Zugriffskontrollen

## Discretionary Access Control (DAC)

- Eigentümer des Objekts legt Zugriffsrechte für Subjekte fest

*Beispiel: Unix-Dateien*

## Mandatory Access Control (MAC)

- Autorität setzt Zugriffsrechte fest

*Beispiel: Geheimhaltung von staatlichen Dokumenten*

Festsetzung

Granularität

## Role-Based Access Control (RBAC)

- Zugriffsrechte durch Rolle festgelegt (i.d.R. mit MAC)

*Beispiel: unterschiedlicher Zugriff für Webdesigner, Administrator, ...*

## Attribute-Based Access Control (ABAC)

- feinere Zugriffsrechte gemäß logischer Formel (i.d.R. mit RBAC)

*Wenn MANAGER und Modus=Lesen, dann...*

# Beispiel: Bell-LaPadula (BLP)

In den 1970'ern für US Air Force entwickelt

einfache Mechanismen mit formalem Sicherheitsmodell

Ziel: Vertraulichkeit

Zugriffsrechte per  
Access Control Matrix (ACM)  
 $(M[s,o])_{s,o}$

	Objekt 1	Objekt 2	...
Subjekt 1	read, write	read	...
Subjekt 2	read	-	...
Subjekt 3	read, write	read, write	...
...	...	...	...

# Formalisierung in Bell-LaPadula

Objekt o wird klassifiziert:

classification(o)  $\in$  Markierung

Subjekt s erhält Freigabe:

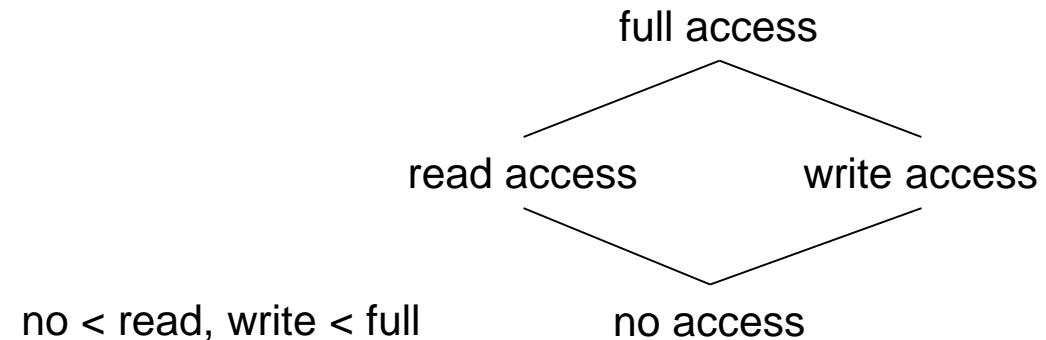
clearance(s)  $\in$  Markierung

Menge der Sicherheitsmarkierungen bildet eine partielle Ordnung  $\leq$  :

Reflexivität:  $x \leq x$

Antisymmetrie: Wenn  $x \leq y$  und  $y \leq x$ , dann  $x=y$

Transitivität: Wenn  $x \leq y$  und  $y \leq z$ , dann auch  $x \leq z$



# Bell-LaPadula Regeln

bestimmen, wann die ACM M Vertraulichkeit garantiert

## Simple Security Property („no read-up“):

Wenn **read** in Matrix  $M[s,o]$ , dann  $\text{classification}(o) \leq \text{clearance}(s)$

## \*-Property („no write-down“):

verhindert, dass Subjekte mit höheren Rechten in Objekte mit niedrigeren Rechten schreiben und damit Informationen preisgeben

Wenn **write** in Matrix  $M[s,o]$ , dann  $\text{clearance}(s) \leq \text{classification}(o)$

## Discretionary Security Property:

Für alle Subjekte, Objekte und Zugriffsmodi gibt es einen Matrixeintrag

# Grenzen des Bell-LaPadula-Modells

kein Integritätsschutz: „write-up“ (z.B. Anfügen) ist grundsätzlich erlaubt

keine Änderungen von Zugriffsrechten möglich

daher zahlreiche Erweiterungen des Modells:

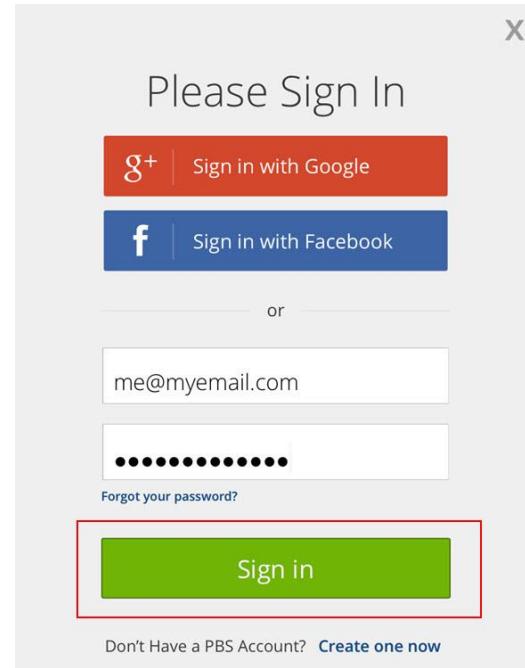
(weak/strong) tranquility bzgl. Änderungen in Matrix M

Biba-Modell: kein „write-up“, kein „read-down“

...

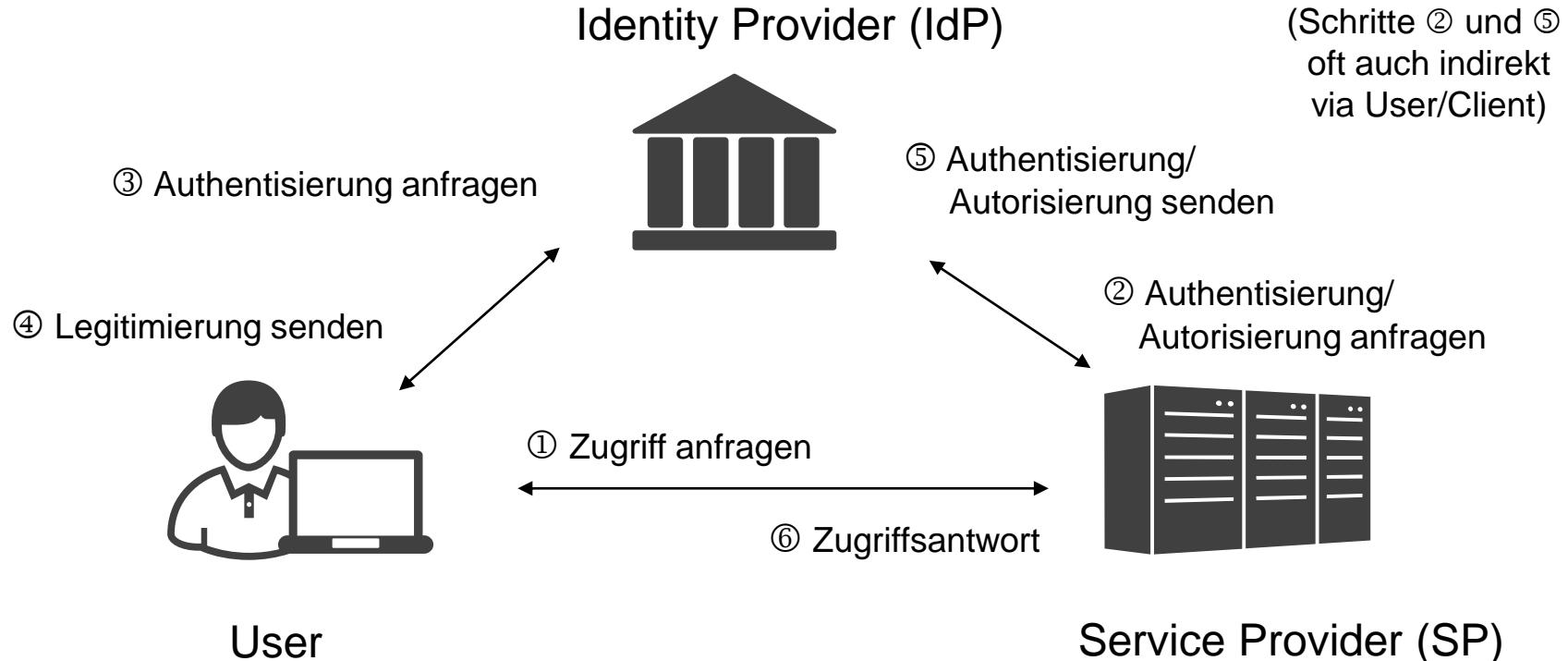
→ „IT-Sicherheit“

# Federated Identity Management (FIdM)



Quelle: pbs.org

# Grundprinzip des FldM



wichtig: Standardisierung zur Interoperabilität (SAML, OAUTH2.0,...)

# Security Assertion Markup Language (SAML)

XML-basierte Sprache zum  
Austausch von Authentisierungs-/Autorisierungsinformation

meist über Webbrowser/HTTP(s)

„SAML token“:

```
<saml:Assertion ...>
  <saml:Issuer>
  <ds:Signature>
  <saml:Subject>
  <saml:Conditions>
  <saml:Advice>
  <saml:AuthnStatement>
  <saml:AuthzDecisionStatement>
  <saml:AttributeStatement>
</saml:Assertion>
```

Assertion ist Aussage über  
Authentisierung („User authentisiert“), oder  
Attribute („User hat Attribut“), oder  
Autorisierung („User darf X“)

# Angriffe auf SAML-Implementierung



## On Breaking SAML: Be Whoever You Want to Be

Juraj Somorovsky<sup>1</sup>, Andreas Mayer<sup>2</sup>, Jörg Schwenk<sup>1</sup>, Marco Kampmann<sup>1</sup>, and Meiko Jensen<sup>1</sup>

<sup>1</sup>Horst Görtz Institute for IT-Security, Ruhr-University Bochum, Germany

<sup>2</sup>Adolf Würth GmbH & Co. KG, Künzelsau-Gaisbach, Germany

{*Juraj.Somorovsky, Joerg.Schwenk, Marco.Kampmann, Meiko.Jensen*}@rub.de,  
*Andreas.Mayer@wuerth.com*

### Abstract

The Security Assertion Markup Language (*SAML*) is a widely adopted language for making security statements about subjects. It is a critical component for the development of secure web applications.

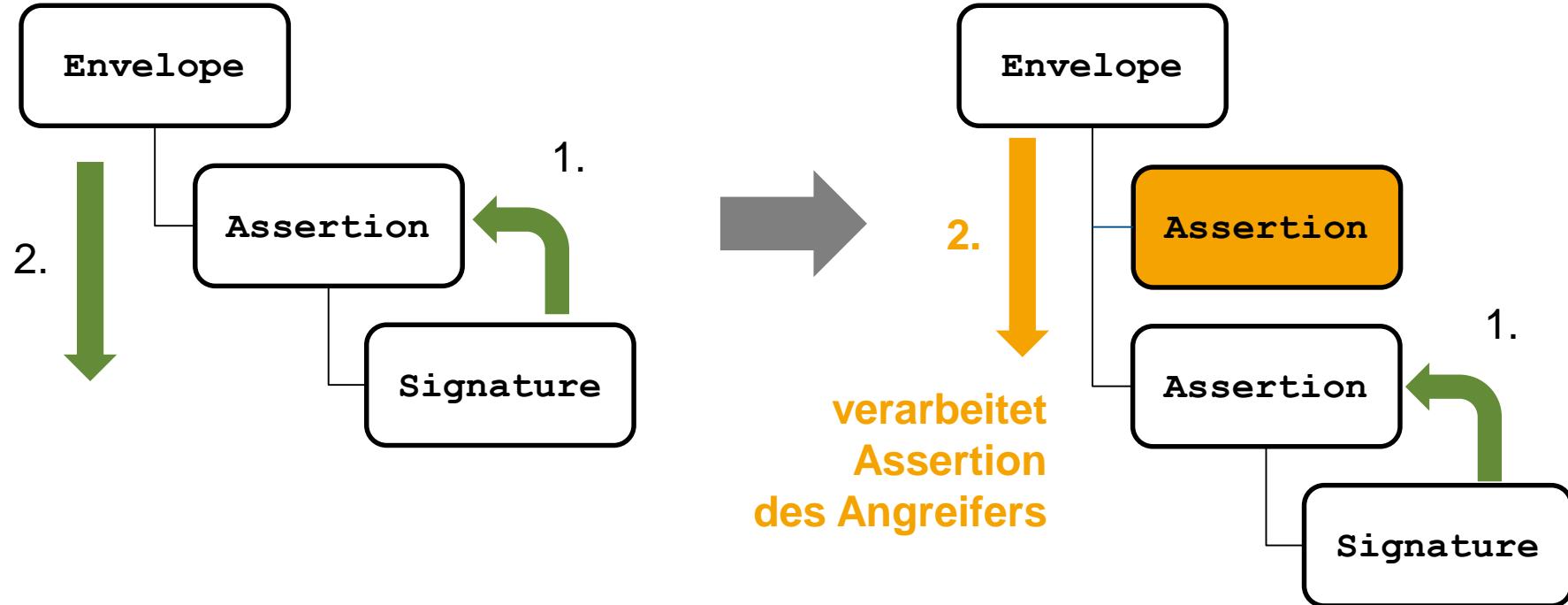
standard, this shall be achieved by using XML Signatures, which should either cover the complete SAML assertion, or an XML document containing it (e.g. a SAML Authentication response).

USENIX 2012

Angriff auf schlechte Verarbeitung gemäß SAML

Provider von SAML wurden informiert und Verfahren vor Veröffentlichung gefixt

# Angriffe auf SAML-Implementierung (vereinfacht!)



zwei „unabhängige“ Schritte in SAML:

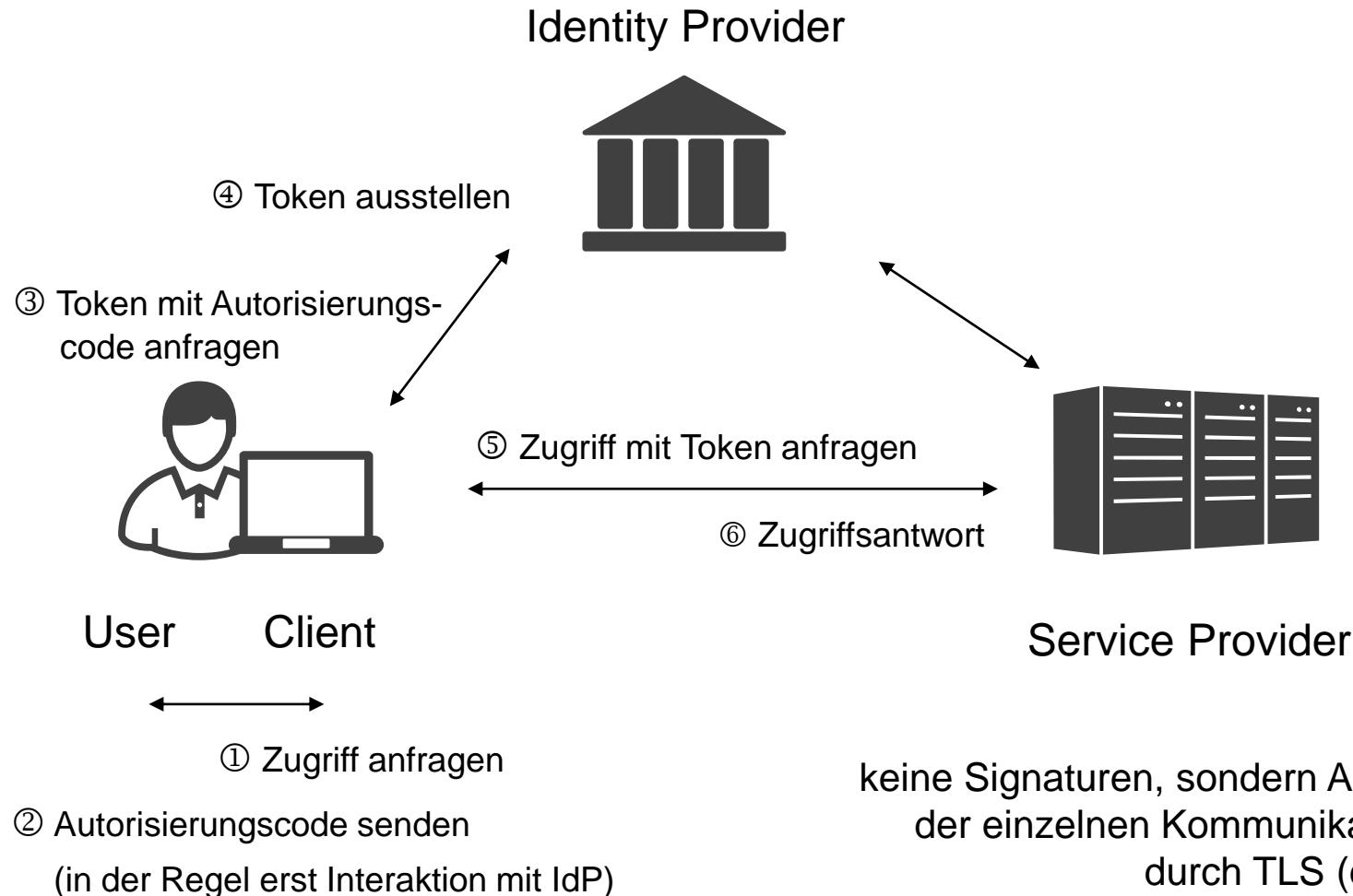
1. Signaturprüfung
2. Verarbeitung der Assertion(s)

Signaturprüfung nur über die zugehörige Assertion

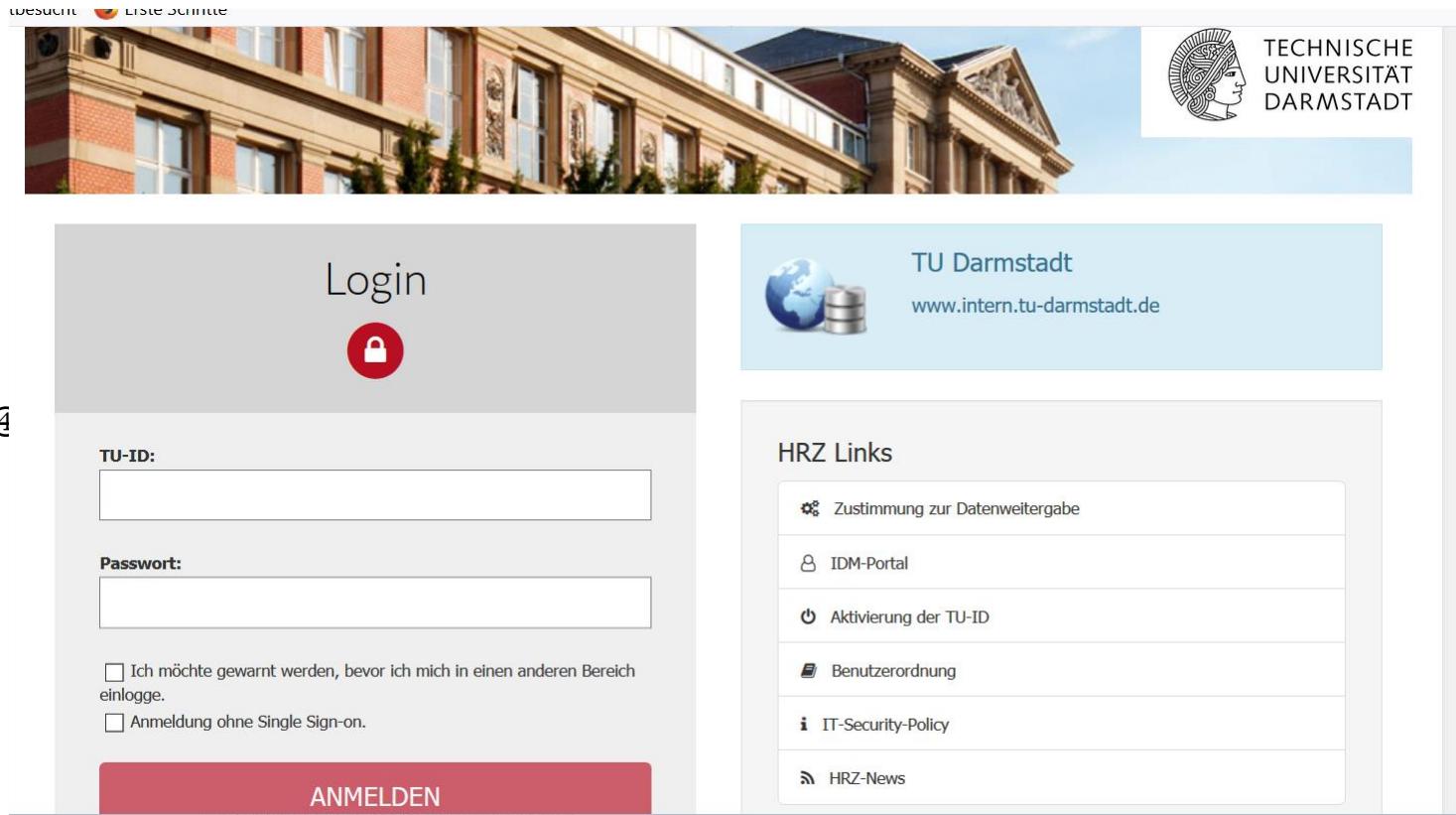
# OAuth2.0-Autorisierungsprotokoll

OAuth: 2006, OAuth2.0: 2012

Autorisierung (!)  
man kann aber in der Regel  
Authentisierung darüber laufen lassen



# Single-Sign-On (SSO)



anfragen

④

WESCHITZ LISTE SCHritte

## Login



**TU-ID:**

**Passwort:**

Ich möchte gewarnt werden, bevor ich mich in einen anderen Bereich einlogge.  
 Anmeldung ohne Single Sign-on.

**ANMELDEN**



**TU Darmstadt**  
[www.intern.tu-darmstadt.de](http://www.intern.tu-darmstadt.de)

**HRZ Links**

-  Zustimmung zur Datenweitergabe
-  IDM-Portal
-  Aktivierung der TU-ID
-  Benutzerordnung
-  IT-Security-Policy
-  HRZ-News

Single-Sign-On:  
Authentisierung bleibt bestehen  
(z.B. Cookie in Webbrowser des Users)

# Content-Management an der TU Darmstadt



Wir sind zur Zeit dabei, das Skript aus dem letzten Sommersemester zu überarbeiten. Eine erste stabile Version (mit leichten Änderungen) ist nun bereits verfügbar. Diese wird veraussichtlich laufend aktualisiert.

## Skript

Zugriffsgeschützter Absatz: [Melden Sie sich an](#), um diesen Absatz zu sehen.

## Übungen

Zugriffsgeschützter Absatz: [Melden Sie sich an](#), um diesen Absatz zu sehen.

## Lösungen

Zugriffsgeschützter Absatz: [Melden Sie sich an](#), um diesen Absatz zu sehen.

eigene Homepage  
an der TU Darmstadt

per Single-Sign-On...



Wir sind zur Zeit dabei, das Skript aus dem letzten Sommersemester zu überarbeiten. Eine erste stabile Version (mit leichten Änderungen) ist nun bereits verfügbar. Diese wird veraussichtlich laufend aktualisiert.

## Skript

[cryptoplexy](#) (PDF-Datei, 1178kB)

## Übungen

- [1exercise](#) (PDF-Datei, 173kB)
- [2exercise](#) (PDF-Datei, 197kB)
- [3exercise](#) (PDF-Datei, 163kB)
- [4exercise](#) (PDF-Datei, 171kB)
- [5exercise](#) (PDF-Datei, 137kB)
- [6exercise](#) (PDF-Datei, 182kB)

[7exercise](#) (PDF-Datei, 160kB)



...oder direkt URL eintippen  
(sofern bekannt),  
ohne Single-Sign-On

[http://www.cryptoplexy.informatik.tu-darmstadt.de/media/crypt/teaching\\_1/cryptoplexy/skript/cryptoplexy.pdf](http://www.cryptoplexy.informatik.tu-darmstadt.de/media/crypt/teaching_1/cryptoplexy/skript/cryptoplexy.pdf)

# Kerberos – Authentisierungsverfahren

Ende 1980 vom MIT entwickelt,  
aktuelle Version V datiert auf 2005



Quelle: <https://web.mit.edu/kerberos/>

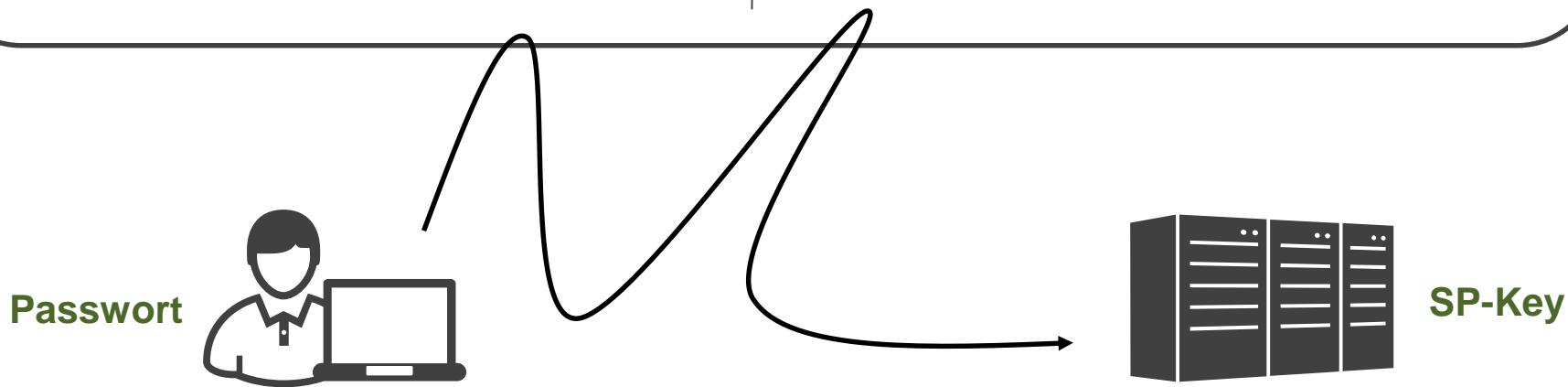
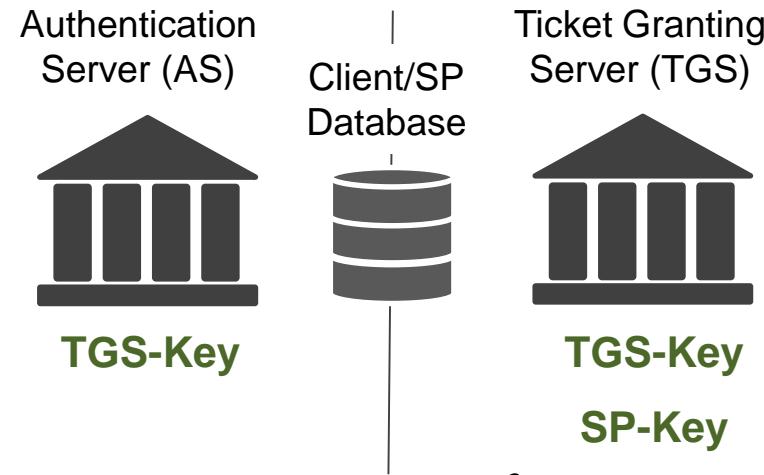
basiert auf symmetrischer Verschlüsselung

einfache SSO-Lösung

immer noch sehr verbreitet

# Kerberos-Protokoll (vereinfacht)

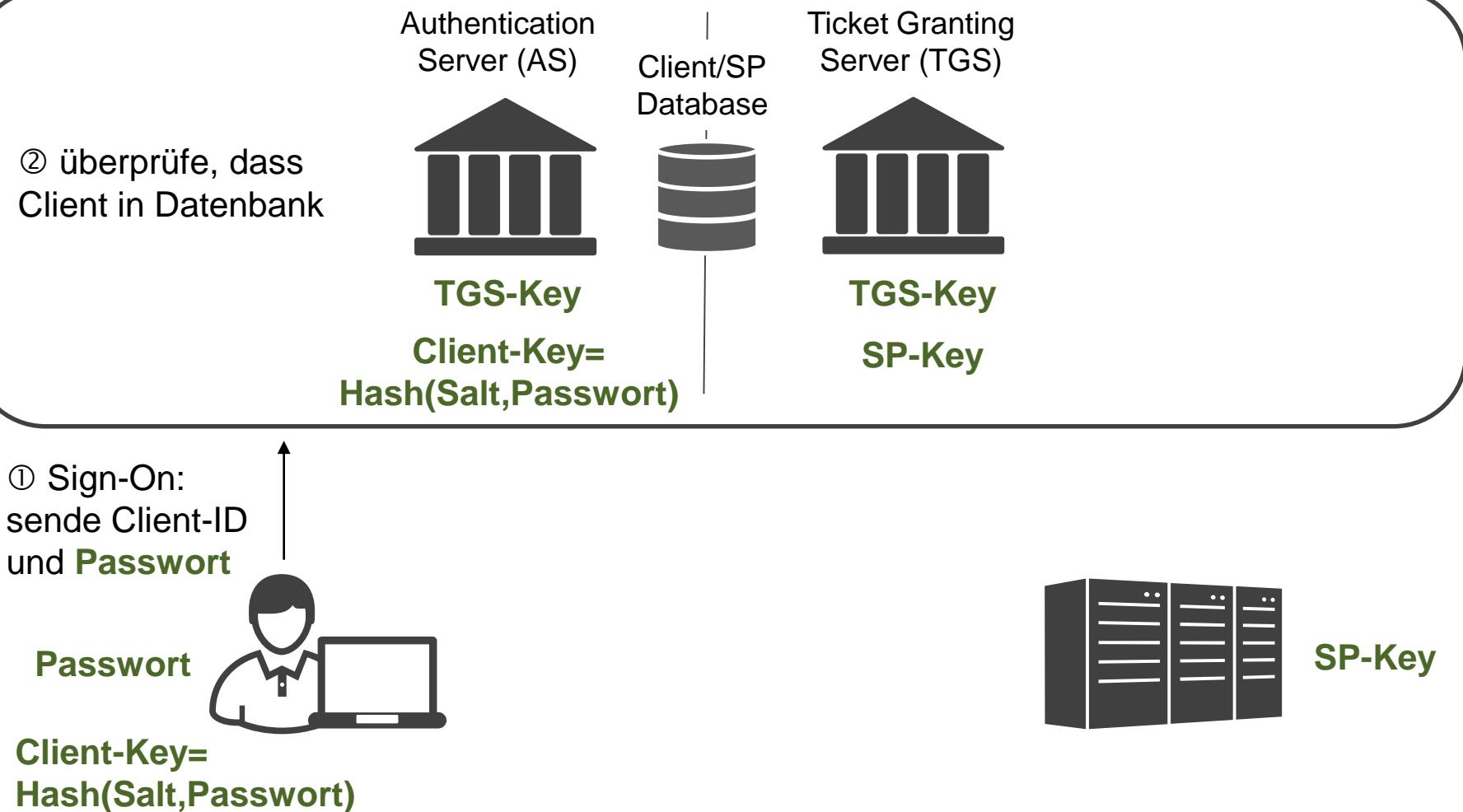
Key Distribution Center (KDC)



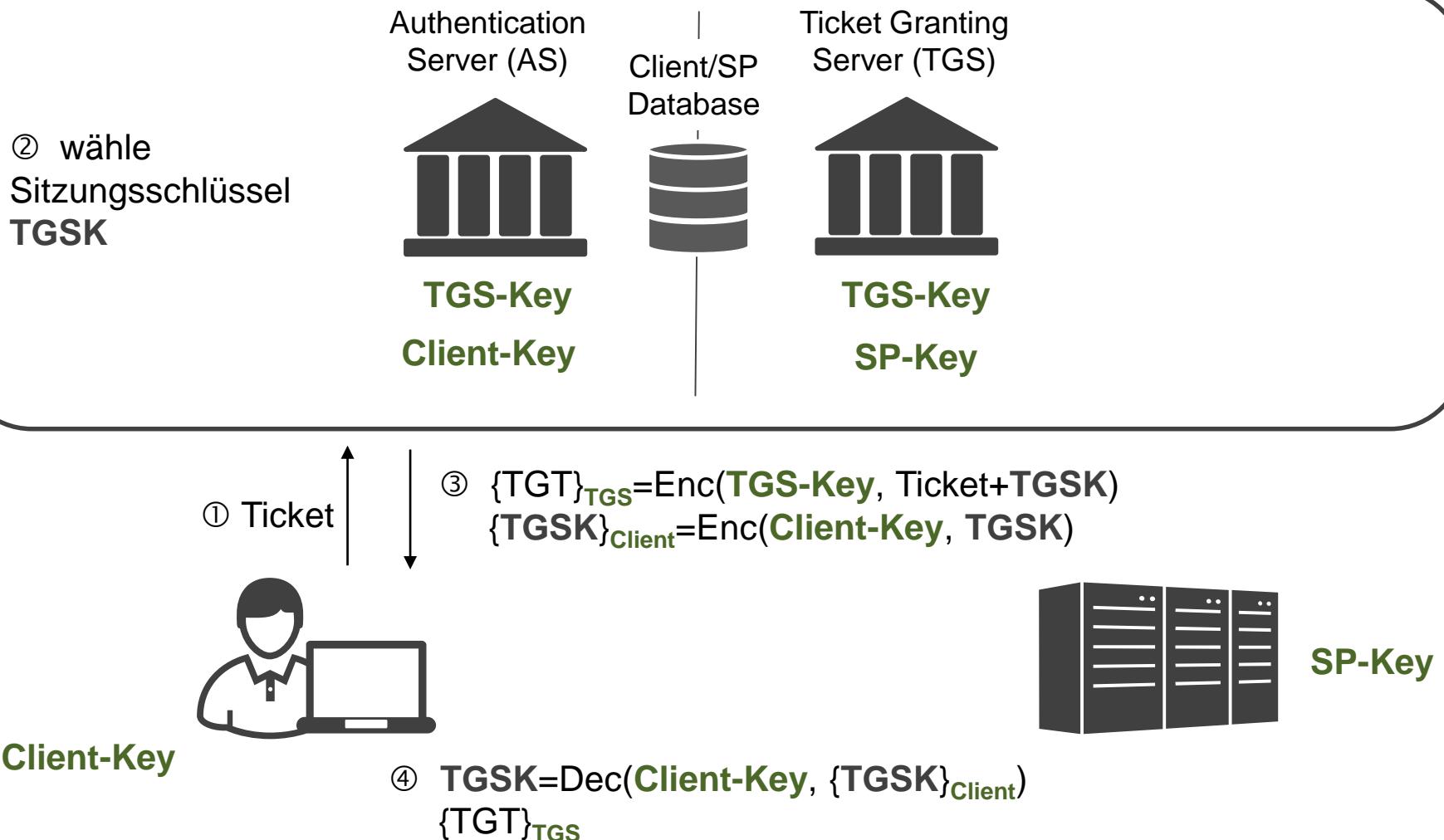
sende Ticket (Lifetime, Timestamp, Client-ID,...)  
via Client an jeweils nächsten Partner

# Kerberos (vereinfacht): Sign-On

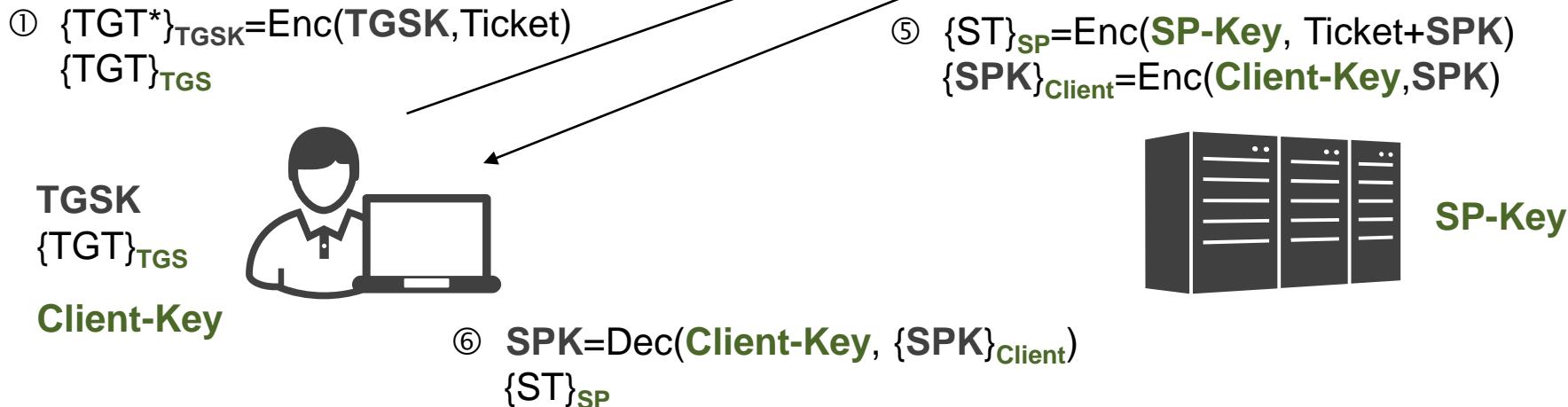
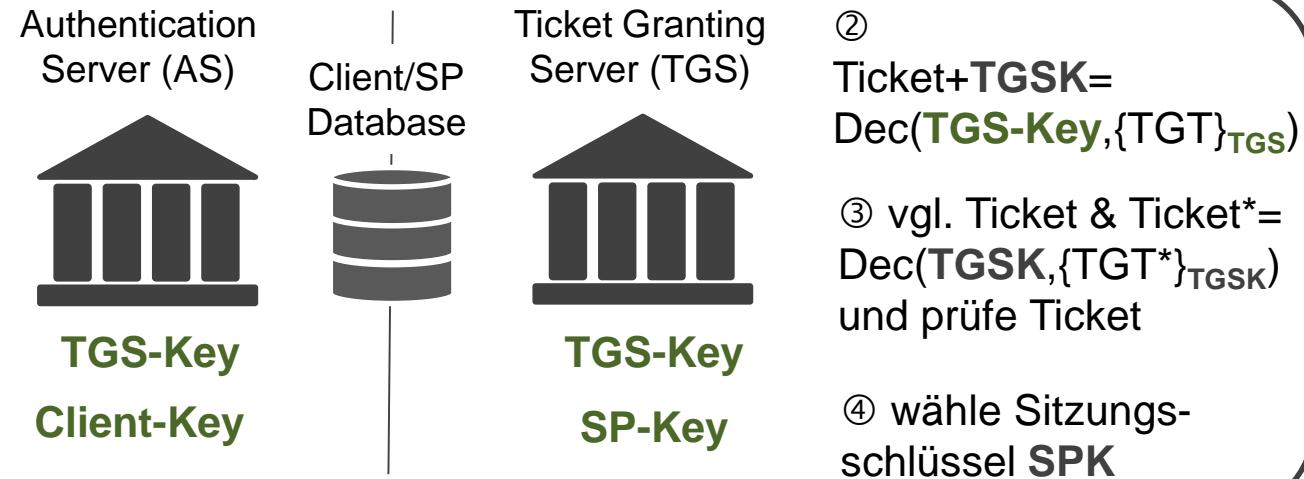
Key Distribution Center (KDC)



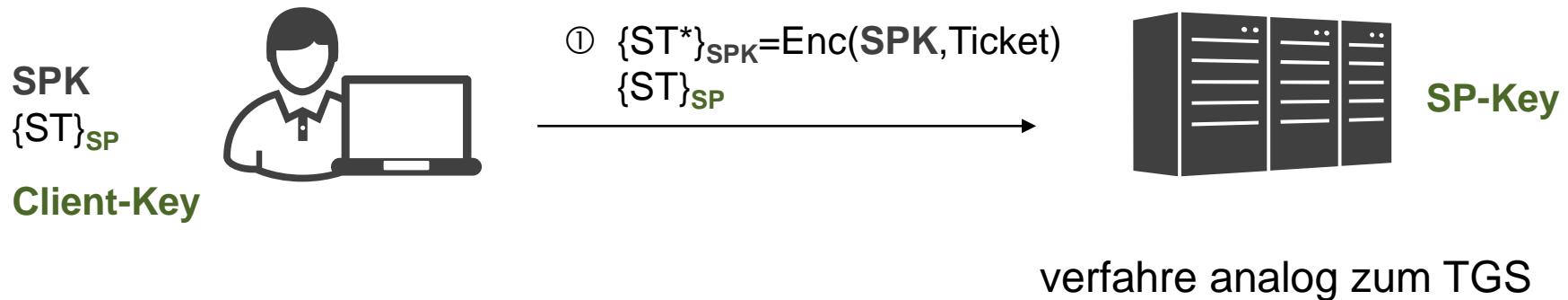
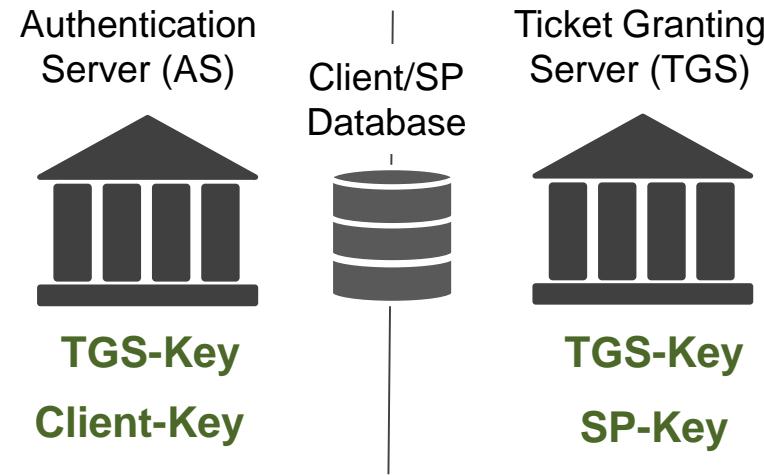
# Kerberos (vereinfacht): Ticket Granting Ticket (TGT)



# Kerberos (vereinfacht): Service Ticket Granting



# Kerberos (vereinfacht): Service Request



# Was Sie gelernt haben sollten

# Unterschiede Identifikation, Authentisierung, Autorisierung

Mittel zur Authentisierung

Speichern von Passwörtern

Angriffe auf Passwörter

Challenge-Response-Verfahren

Vor- und Nachteile von Biometrie

CAPTCHAs

Autorisierung

Prinzipien von Bell-LaPadula

Federated Identity Management

Kerberos