Informatik 6ai Schuljahr 2024/2025 als PDF exportieren

Informatik 6. Klasse - Schuljahr 2024/25

Lehrplan

- Lehrplaninhalte

Themengebiete

- 7) Web Design Grundlagen (HTML, CSS)
- 8) Netzwerksicherheit

Leistungsbeurteilung

1/3 - Test (SA)

- 2x Tests pro Semester
  - 1. Test – Do, 17.10.2024 - Themengebiet 7 - 8 (bis Symmetrische Verschlüsselung inkl. Skytale, Cäsar, Vigenere)
  - 2. Test – Do, 12.12.2024 - Themengebiet ??
  - 3. Test – Do, ??.03.2025 - Themengebiet ??
  - 4. Test – Do, ??.05.2025 - Themengebiet ??

1/3 - Mitarbeit (MA)

- Aktive Mitarbeit im Unterricht (aMA)
- Mündliche Stundenwiederholungen (mMA)
- Schriftliche Stundenwiederholungen (sMA)

1/3 - Praktische Arbeiten (PA)

- 1x praktischer Arbeitsauftrag pro Woche via Google Classroom

Leistungsstand

Den aktuellen Leistungsstand könnt ihr jederzeit einsehen!

Lehrplaninhalte

Die nachstehenden Lehrplaninhalte werden in der angegebenen Reihenfolge (=Priorität) durchgenommen werden. Im Unterrichtsfach IT-Labor werden die Themen projektbasiert behandelt.

5 + 6. Klasse IT-Labor/WPF

1. Videobearbeitung mit DaVinci Resolve
2. Hardware – PC Systeme (Laptop, Spielekonsole, Handy,..) kennenlernen, Komponenten identifizieren
3. 3D – Modellierung & Animation & Druck
4. Mediendesign (Canva – Flyer, Plakate) & Inkscape Vektorgrafiken (Logo) & Bild (Pixlr) & Podcast bzw. Audiobearbeitung / Videoblog
5. Kamerasysteme (Handy + Gimbal, Drohne, GoPro,..)
6. Netzwerktechnik – Praxis
7. Robotik (Wetterstation,..)
8. Handy-App Programmierung
9. VR – Programmierung
10. Unity – 3D Spielprogrammierung

5. Klasse

1. Zahlensysteme
2. Informationseinheiten
3. Zeichencodierung
4. Schaltalgebra
5. Algorithmik und Programmierung Basics (C++ Grundstrukturen bis Funktionen)
6. Tabellenkalkulation & Datenanalyse (Datenimport, filtern, exportieren, SVERWEIS, WENN-DANN, Pivot-Tabellen)

6. Klasse

1. Webentwicklung Basics (HTML & CSS)
2. Datenschutz- & Sicherheit & Lizensierung & Kryptographie & Verschlüsselung
3. Netzwerktechnik Theorie & Simulation
4. Algorithmik und Programmierung Datenstrukturen (C++ Rekursionen, Arrays + Sortieralgorithmen)

7. Klasse

1. Algorithmik und Programmierung Objektorientierung (Klassen, Vererbung)
2. Betriebssysteme (Windows / Linux – Scripts, Serverdienste – Webserver, DB-Server) & Virtualisierung

3. Datenbanksysteme (ER-Modelle, Relationenmodell, SQL)
4. Textverarbeitung (mehrseitige Dokumente, VWA Vorlagen,..)

8. Klasse

1. Webentwicklung Advanced (PHP Formulare & DB-Connection, Javascript, Frameworks Bootstrap, CMS - Systeme)
2. Grundlagen der KI

8) Netzwerksicherheit

Zweifel zu haben ist ein unangenehmer Zustand, sich in Sicherheit zu wiegen ist ein absurder Zustand (Voltaire)

Not, Person und Zeit, machen die Gesetze eng und weit.

Es gibt keine Sicherheit, nur mehr oder weniger Unsicherheit (Josef Maler)

Sei vorsichtig, öffne keinem Fremden die Haustür.

kryptographie.mp4

Definition

Netzwerksicherheit steht als Begriff stellvertretend für sämtliche Schutzmaßnahmen, um IT-

Infrastrukturen gegen unbefugte Zugriffe, Schäden und Verluste abzusichern. Diese Maßnahmen können technischer oder organisatorischer Natur sein und sorgen dafür, dass ein Netzwerk vertraulich, integer und verfügbar bleibt. So zählen Verschlüsselungstechnologien und Firewalls ebenso zur professionellen Netzwerksicherheit wie Sicherheits- und Passwortrichtlinien und Security-Schulungen.

Um ein IT-Netzwerk umfassend zu sichern, braucht es mehrere Schutzschichten, die jeden Bereich im Netzwerk bedenken. Außerdem ist es sinnvoll, Netzwerk und Sicherheit individuell aufeinander abzustimmen. Wir zeigen Ihnen, welche Security-Schichten notwendig sind und wie umfangreich Ihre IT-Security je nach Unternehmensart und Angriffsrisiken ausfallen sollte.

Schutz des eigenen IT-Netzwerkes

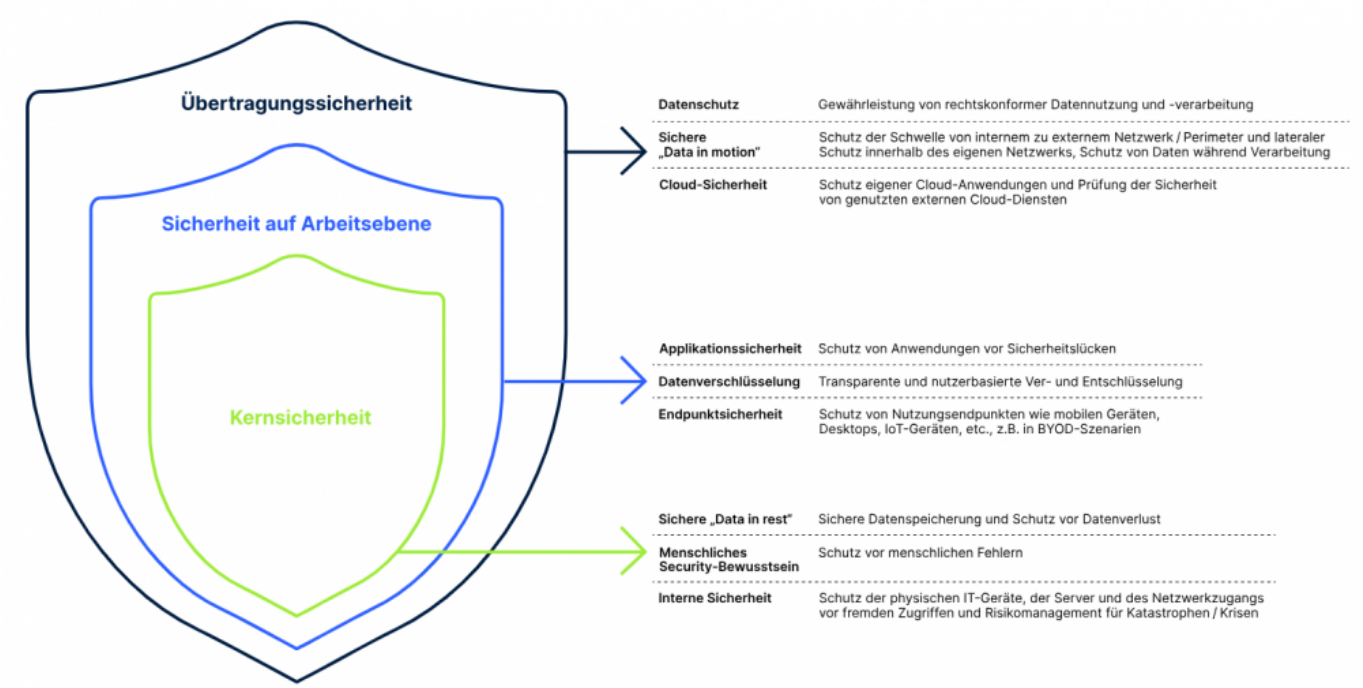
Ein IT-Netzwerk besteht aus einer Vielzahl an Einzelkomponenten: Wichtige Daten lagern auf internen oder externen Servern, Mitarbeitende sind mit Desktop-PCs, Laptops und Smartphones ausgestattet, ein Gateway oder ein Router sorgt für den Internetzugang und je nach Unternehmensgröße vernetzen ein oder mehrere Switches intern weitere Geräte (LAN) wie beispielsweise Access Points für eine professionelle WLAN-Abdeckung oder auch Arbeitsgeräte oder Drucker.

Bei IT-Netzwerken, die lokal weit verteilt und stark verzweigt sind, bedarf es zum Schutz einer mehrschichtigen und skalierbaren IT-Security, die im besten Fall möglichst automatisiert für mehr Sicherheit im Netzwerk sorgt.

Schichten und Bestandteile professioneller IT-Security

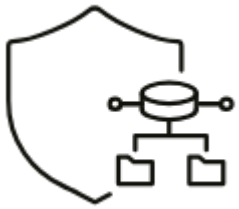
Zum Schutz des eigenen IT-Netzwerkes zählt zum einen die Absicherung sämtlicher sich im Netzwerk befindlichen Daten, Geräte, Server und Applikationen (Datensicherheit, interne Sicherheit und Applikationssicherheit), zum anderen aber auch Perimeter Security, also die Absicherung der Schwelle zwischen internem und externem Netz – demnach auch sichere Übertragungswege und Netzwerkzugänge (Endpunktsicherheit und Datenverschlüsselung).

Zu guter Letzt darf aber auch die Sicherheit der bewegten Daten und externen Cloud-Anwendungen nicht außer Acht gelassen werden (Sicherheit von „data in motion“, Datenschutz und Cloud-Sicherheit), ebenso wenig wie die menschliche Komponente (menschliches Security-Bewusstsein).



Kernsicherheit

Sichere „Data in rest“



Sichere Datenspeicherung und Schutz vor Datenverlust durch

- Strenge Passwort-Richtlinien (Komplexität, Ablaufdatum, Zwei-Faktor-Authentifizierung)
- Sicherheitseinstellungen für Dateien
- Getrennte Datenlagerung
- Klar definierte Zugriffs- und Bearbeitungsrechte
- Archive
- Datenbackups

Interne Sicherheit

Schutz der physischen IT-Geräte, Server und des Netzwerkzugangs vor fremden Zugriffen sowie Risikomanagement für Katastrophen / Krisen durch

- Zugangskontrollen auf Firmengelände
- ggf. Live-Monitoring mit Kameraüberwachung und Aktivitäten-Logs
- Professionelle Next-Generation Web Application UTM-Firewall
- Regelmäßige Security-Patches, Funktionsfähigkeitschecks und Software Updates
- Netzwerksegmentierung

Menschliches Security-Bewusstsein

Schutz vor menschlichen Fehlern durch

- Umfangreiche IT-Security- und Compliance-Schulungen, insb. zu E-Mail-, Passwort- und Social Media-Sicherheit, Vertraulichkeitsregelungen und Verhalten im Ernstfall
- Tests wie z.B. Phishing-Simulationen
- Vertraulichkeitsklassifikationen für Dateien und Informationen
- Gut zugängliche Sicherheitsrichtlinien
- Regelmäßige Erinnerungen und Auffrischungen

IT-Sicherheit auf Arbeitsebene

Endpunktsicherheit

Schutz von Nutzungsendpunkten wie mobilen Geräten, Desktops, IoT-Geräten und -Sensoren, etc., insb. in dezentralen und hybriden Arbeitsumgebungen (BYOD, Remote Work, externe Dienstleister, etc.) durch

- Nutzung von VPN-Clients und ZTNA (Zero-Trust-Network-Access)
- Multifaktor-Authentifizierung von Nutzer:innen
- Übersichtliches, zentral gemanagtes Asset-Inventar aller Geräte und virtuellen Ressourcen
- Individuelle, feingranulare Zugriffsrechte pro Nutzer:in
- Regelmäßige Systemupdates und Securitypatches
- Klare Nutzungs- und Sicherheitsrichtlinien
- Abschalten nicht zwingend notwendiger Ports

Datenverschlüsselung

Transparente und nutzerbasierte Ver- und Entschlüsselung im Hintergrund durch

- VPN- oder ZTNA-Netzwerke
- Verschlüsselungsparameter und - algorithmen nach aktuellem BSI-Standard
- Regelmäßige Überprüfung der aktuellen Standards
- Konzept zur Schlüsselverwaltung (PKI - Public Key Infrastructure) inklusive regelmäßiger Audits

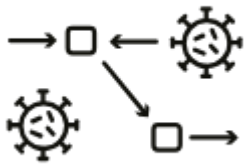
Applikationssicherheit

Schutz von Anwendungen vor Sicherheitslücken durch

- Zugriff auf ausschließlich vertrauenswürdige und geprüfte, freigegebene Apps
- Gezielter Einsatz von VPNs
- Zugangskontrolle und Application Monitoring / Steering
- Automatische Sessionterminierung bei Nicht-Nutzung
- Regelmäßige (Security-)Updates
- Aufbewahrungsrichtlinien

Übertragungssicherheit

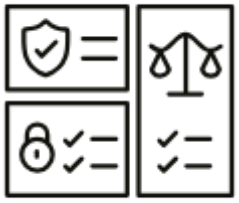
Sichere „Data in motion“



Schutz der Schwelle von internem zu externem Netzwerk (Perimeter) und lateraler Schutz innerhalb des eigenen Netzwerks durch

- Mehrschichtige Verschlüsselung von Daten während Übertragung
- Ausschließliche Nutzung von VPN- oder ZTNA-Verbindungen
- Netzwerksegmentierung durch VLANs
- Caching Routines zur Vermeidung von öffentlichem Zugang
- Regelmäßige Penetrationstests
- Nutzer- und systemspezifische Zugriffsschlüssel nur auf benötigte Daten

Datenschutz



Gewährleistung von rechtkonformer Datennutzung und -verarbeitung durch

- DSGVO-konforme Anwendungen und Systeme
- Backdoor-freie Netzwerkkomponenten
- In der EU gehostete Clouddienste

Cloud-Sicherheit

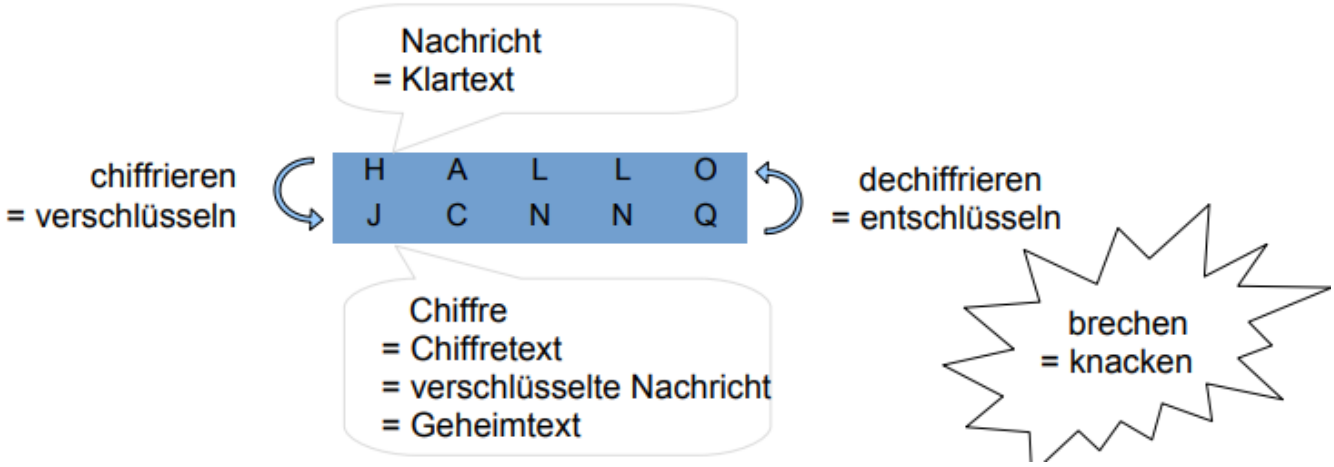
Schutz eigener Cloud-Anwendungen und sorgfältige Prüfung der Sicherheit von genutzten externen Cloud-Diensten durch

- Klare Klärung von Security-Angelegenheiten zwischen Cloud-Anbieter und -Nutzer
- Physische Host-Zugangskontrollen
- Sichere, DSGVO-konforme Infrastruktur
- Georedundanz des Hosts
- Security Patches
- Zugangskontrollen zu Unternehmensdaten in der Cloud
- Backdoor-Freiheit

Kryptologie

Umgangssprachlich werden kryptologische Begriffe oft nicht eindeutig verwendet. Daher ist inbesonderem Maße auf eine korrekte Verwendung der Begriffe zu achten. Hier eine kurze Zusammenfassung:

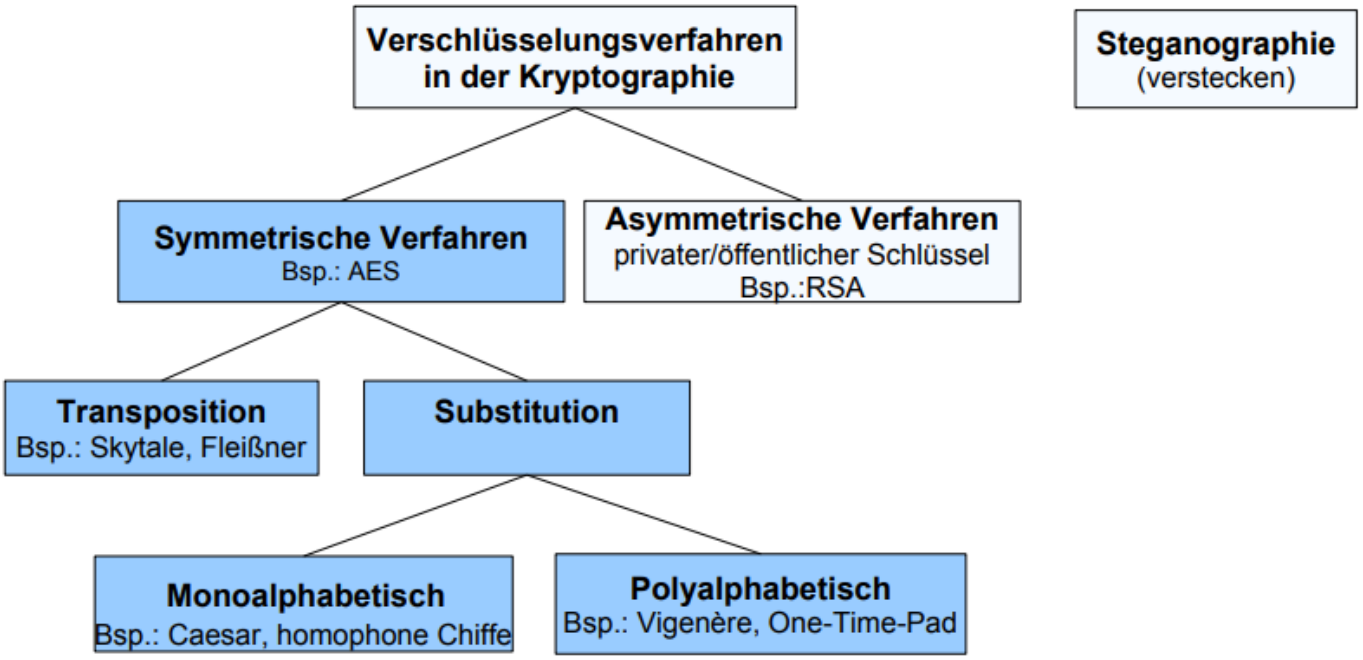
- Beim **Datenschutz** wird die Person mit ihren Rechten geschützt (Persönlichkeitsrecht,Urheberrecht,...).
- Bei der **Datensicherheit** werden die Daten vor unberechtigten Zugriffen geschützt.



Die Kryptologie ist die Wissenschaft der Verschlüsselung und Entschlüsselung von Informationen sowie Analyse kryptografischer Verfahren. Sie umfasst die

- **Kryptografie**: Lehre von den Methoden zur Ver- und Entschlüsselung von Nachrichten zum Zweck der Geheimhaltung von Informationen gegenüber Dritten.
- **Kryptoanalyse**: Analyse und Bewertung der Sicherheit von kryptografischen Verfahren gegen unbefugte Angriffe.

Bei einer **Verschlüsselung** ist das Verfahren (meist) bekannt, der Schlüssel ist geheim. Es geht um den Austausch von Informationen, die nicht für alle bestimmt sind.. Bei einer **Codierung** ist das Verfahren bekannt, und die Anleitung zum Codieren und Decodieren öffentlich. Einen Schlüssel gibt es nicht, und die ausgetauschten Informationen sind nicht geheim. (Blindenschrift, Morsecode, ...).



Kryptografie

Kurz: Kryptografie ist die Lehre der Verschlüsselung von Daten.

Lang: Kryptografie ist eine Wissenschaft, die sich mit Methoden beschäftigt, die durch Verschlüsselung und verwandte Verfahren Daten von unbefugten Manipulation schützen sollen.

Ursprung: Das Wort Kryptografie kommt aus dem Griechischen, wo kryptein „verstecken“ und gráphein „schreiben“ bedeutet.

Die beiden **wichtigsten Hilfsmittel der Kryptografie** sind:

- Die **Mathematik**, denn nur mit Hilfe von mathematischen Kenntnissen ist es möglich, Verfahren zur sicheren Verschlüsselung von Daten zu entwickeln
- Und der **Computer**, weil er die Verschlüsselungsverfahren ausführt und wichtige Dienste bei der Untersuchung von kryptografischen Methoden auf Schwachstellen leistet.

Motive der Kryptografie

- **Vertraulichkeit**

Geheimhaltung ist die offensichtlichste und bekannteste Anwendung kryptografischer Verfahren

- **Intigrität**

Für den Empfänger nachprüfbar sein, das er die Nachricht unversehrt erhalten hat

- **Authentizität**

Identität des Absenders einer Nachricht soll für den Empfänger nachprüfbar sein

- **Gültigkeit**

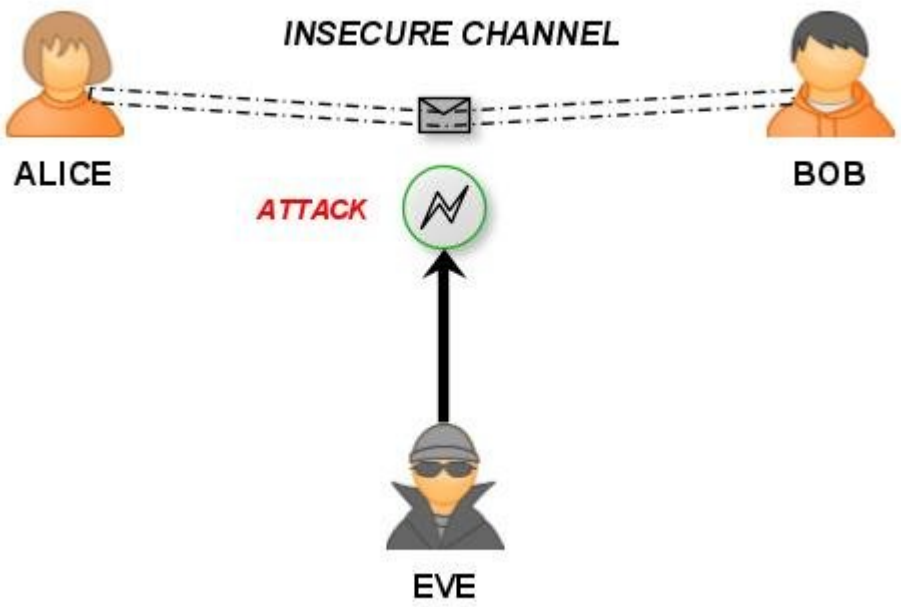
Nachricht kann durch zwischenzeitliche Ereignisse ihre Bedeutung verlieren

- **Nichtabstreitbarkeit**

Dass ist Absender einer Nachricht seine Urheberschaft später nicht verleugnen kann.

Allgemeines Modell

2 Personen (Alice und Bob) tauschen Daten über einen abhörbaren Kanal aus, heute meist das Internet, es kann dies aber auch eine Telefonleitung, eine Funkverbindung oder der Transport einer Diskette sein. Eine „böse“ gesinnte Person (Mallory/Eve) kann den Übertragungskanal beliebig beeinflussen. Er kann die Daten abfragen, mitlesen, analysieren, manipulieren und weiterleiten.



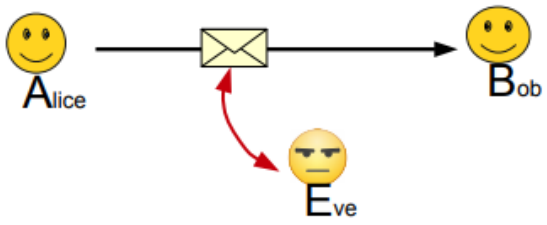
Auf Basis dieses einfachen Modells, kann die Kryptografie durch Verschlüsselung und ähnliche Maßnahmen verhindern dass

- Mallory mit den abgefangenen Daten etwas anfangen kann,
- Mallory übertragene Daten unbemerkt verändert,
- Mallory sich unbemerkt Alice gegenüber als Bob ausgibt (und umgekehrt)
- Alice unerkannt behaupten kann, dass eine von ihr gesendete Nachricht in Wirklichkeit eine Fälschung von Mallory sei.

Die Kryptografie kann aber nicht verhindern, dass

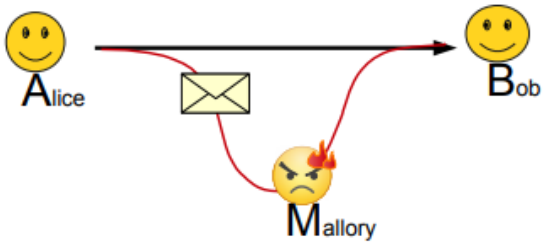
- Mallory Nachrichten verändert (er kann es nur nicht unbemerkt),
- Mallory Daten abfängt (er nur nichts von verschlüsselten Daten),
- Mallory die Leitung zerstört (physikalisch, durch Softwarefehler u.ä.).

Szenarien:	**Gefahren:**	**Ziele** der Kryptologie



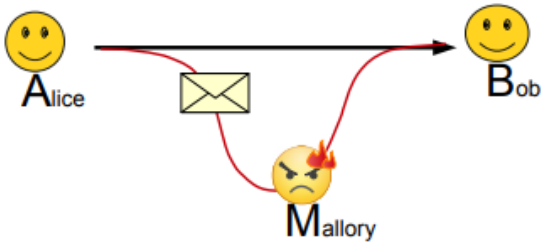
mitlesen => **Vertraulichkeit**

Können wirklich nur Alice und Bob die Nachricht lesen?



ändern => **Integrität**

Ist die Nachricht unverändert?
Sind die Daten original?



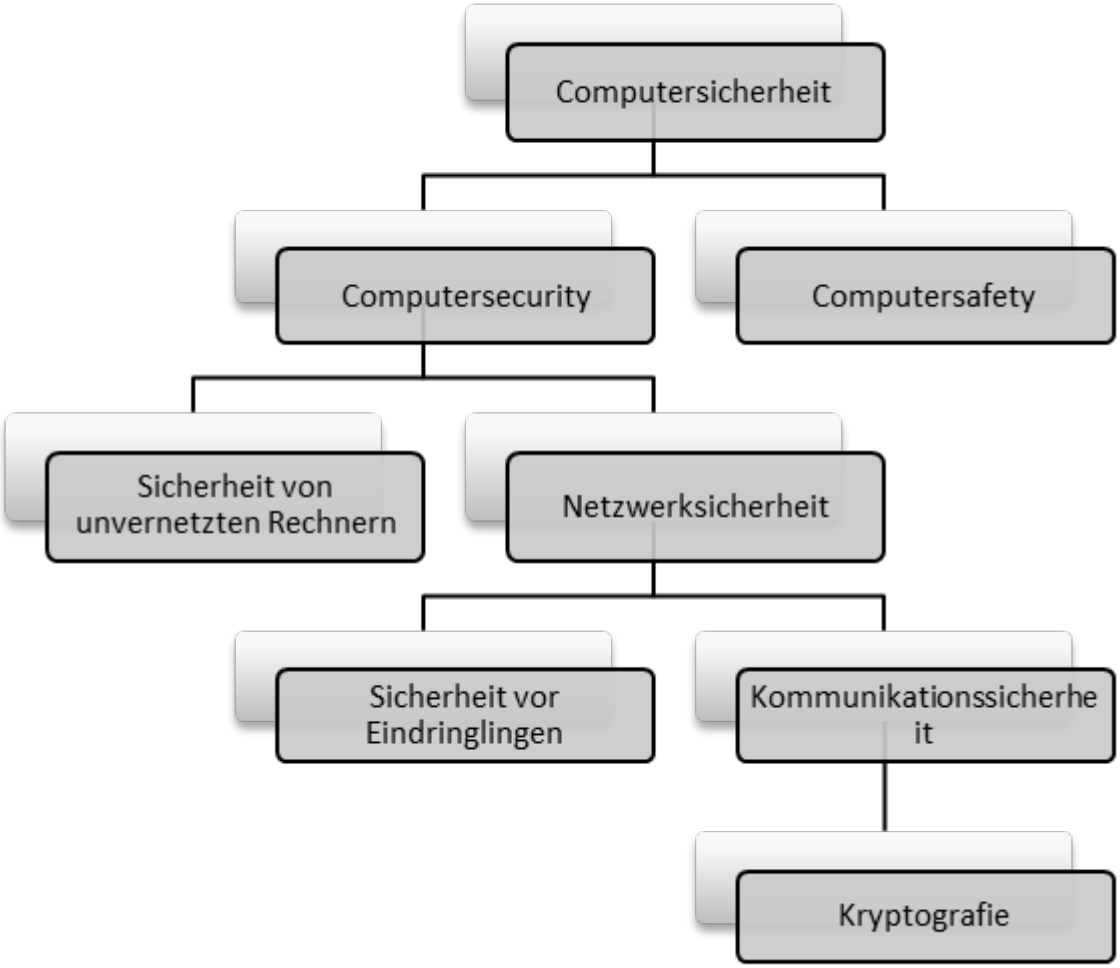
als A ausgeben
=> **Authentizität**
Kommt die Nachricht wirklich von Alice?
Landet die Nachricht wirklich bei Bob?

=> **Verbindlichkeit**
Kann Bob beweisen, dass die Nachricht von Alice kommt, selbst wenn sie es abstreitet? ('*Habe ich nie gesagt.*')
Kann Alice beweisen, dass Bob die Nachricht erhalten hat? ('*Habe ich nicht bekommen.*')

=> Unterschiedliche **Ziele** erfordern unterschiedliche **Verfahren**.

Bsp.: Eine Verschlüsselung liefert Vertraulichkeit, aber keine Authentizität.

Teilgebiet der Computersicherheit



- In der Computer-Safety geht es um den Schutz vor unbeabsichtigten Schäden. Dazu gehören defekte Geräte, unbeabsichtigtes Löschen, Übertragungsfehler, Festplatten-Crashes, Blitzeinschläge, Überschwemmungen, falsche Bedienung, defekte Speichermedien und Ähnliches.
- Die Computer-Security dagegen bezeichnet die Sicherheit vor absichtlichen Störungen. Dazu gehören die Sabotage von Hardware, Hackereinbrüche, das Schnüffeln in geheimen Dateien und dergleichen.

Der Bereich der Netzwerksicherheit beschäftigt sich vor allem mit zwei Sicherheitsfragen:

- Wie kann ein vernetzter Computer davor beschützt werden, dass ein Unbefugter über das Netzwerk darauf zugreift (man spricht dabei von hacken oder cracken).
- Wie können Nachrichten, die den Computer verlassen vor einem Abhörer oder Manipulierer geschützt werden (Kommunikationssicherheit).

Gründe für Kryptografie

- **Wirtschaftsspionage**

Staatliche Geheimdienste sind nach dem Ende des kalten Krieges vermehrt auf neue

Beschäftigungsbereiche verlegt worden. Wirtschaftliche Interessen gelten heute als häufiger Grund Spionage zu betreiben. Weltweit führend – nicht nur im Bereich der Wirtschaftsspionage – ist die amerikanische Geheimorganisation NSA (National Security Agancy). Die NSA ist der weltweit größte Arbeitgeber von Mathematikern und größter Hardwareabnehmer. Firmen nutzen heute auch die Kenntnisse von Hackern um die Konkurrenz auszuspionieren. Der geschätzte Schaden durch Wirtschaftsspionage übersteigt in Ländern wie Deutschland die Milliardengrenze. Auch wenn nur ein Teil davon über das Internet erfolgt, so ist die Gefahr vielfach unterschätzt oder nicht bewusst.

- **Kommerzielle Nutzung des Internets**

Ein guter Grund für den Einsatz von Kryptografie ist die Tatsache, dass sich in einem abhör- und manipulationssicheren Internet mittels Online-Shops, Auktionsbörsen, OnlineBanking u. ä. eine große Menge an Geld verdienen/bewegen lässt.

- **Privatsphäre**

Es gibt auch Gründe für den Einsatz von Kryptografie, die keine kommerziellen Gedanken verfolgen. So ist es das Recht jedes Bürgers, eine Privatsphäre zu behalten durch den Einsatz von Kryptografie möglich. Ein privater Brief wird ja auch in einem Umschlag versandt! Es ist im Internet auf jeden Fall einfacher, abgefangene Daten maschinell auszuwerten, als dies mit herkömmlicher Briefpost der Fall war.

Bei allen Vorteilen der Kryptografie darf aber nicht vergessen werden, dass sie auch Gefahren bringt: Kriminelle können durch den Einsatz geeigneter Verschlüsselungsverfahren nach Belieben Nachrichten austauschen.

Anwendungen

- **Passwörter**

Kryptografie wird häufig eingesetzt, um die Authentizität von Passwörtern zu überprüfen und gleichzeitig gespeicherte Passwörter zu verschleiern. Auf diese Weise können Dienstanbieter Passwörter authentifizieren, ohne eine Klartextdatenbank mit allen Passwörtern führen zu müssen, die für Hacker anfällig sein könnte.

- **Sicheres Surfen im Internet**

Beim Surfen auf sicheren Websites schützt die Kryptografie die Benutzer vor Lauschangriffen und „Man-in-the-Middle“-Angriffen (MitM). Die Protokolle Secure Sockets Layer (SSL) und Transport Layer Security (TLS) basieren auf der Verschlüsselung mit öffentlichen Schlüsseln, um die zwischen Webserver und Client gesendeten Daten zu schützen und sichere Kommunikationskanäle

herzustellen.

- **Elektronische Signaturen**

Elektronische Signaturen, oder E-Signaturen, werden zum Unterzeichnen wichtiger Dokumente im Internet verwendet und gelten oftmals als rechtsverbindlich. Mit Kryptografie erstellte elektronische Signaturen können validiert werden, um Betrug und Fälschungen zu verhindern.

- **Kryptowährungen**

Kryptowährungen wie Bitcoin und Ethereum beruhen auf einer komplexen Verschlüsselung von Daten, deren Entschlüsselung erhebliche Mengen an Rechenleistung erfordert. Durch diese Entschlüsselungsprozesse erfolgt das sogenannte „Minting“ neuer Coins, die dann in Umlauf gebracht werden. Kryptowährungen stützen sich zudem auf fortschrittliche Kryptografie, um Krypto-Wallets zu sichern, Transaktionen zu verifizieren und Betrug zu verhindern.

- **Authentifizierung**

In Situationen, in denen eine Identitätsauthentifizierung erforderlich ist, wie z. B. bei der Anmeldung bei einem Online-Bankkonto oder beim Zugriff auf ein sicheres Netzwerk, kann die Kryptografie bei der Verifizierung der Identität von Benutzern und der Authentifizierung ihrer Zugriffsberechtigungen helfen.

- **Sichere Kommunikation**

Ganz gleich, ob es um den Austausch von Staatsgeheimnissen oder um eine private Unterhaltung geht – die End-to-End-Verschlüsselung wird zur Authentifizierung von Nachrichten und zum Schutz von Zwei-Wege-Kommunikation wie Videokonferenzen, Sofortnachrichten und E-Mails verwendet. Die End-to-End-Verschlüsselung bietet ein hohes Maß an Sicherheit und Privatsphäre für die Nutzer und wird häufig in Kommunikations-Apps wie WhatsApp und Signal verwendet.

Steganografie

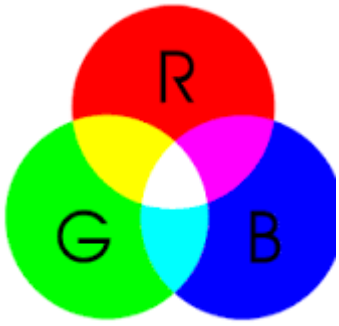
Ein steganografisches Verfahren verheimlicht, dass überhaupt geheime Daten existieren. Der Gedanke dahinter: Wo niemand geheimen Daten vermutet, wird sie auch niemand suchen. Steganografie-Software versteckt die geheimen Daten in einer anderen Datei. Also so genannte Trägerdateien dienen in der Regel meist Bilder, Sound-, Text- und Video-Dateien. Dieses Verfahren kann man nicht nur zum Schutz von Daten benutzen, sondern es wird auch zur Kenntlichmachung von Urheberrechten verwendet. Wer von der Verschlüsselung nichts weiß, nutzt die betreffende Trägerdatei ohne Einschränkungen mit der passenden Anwendung. Nur wer über die Verschlüsselung informiert ist und zudem Zugriff auf den verwendeten Kodierungs-Schlüssel hat, kann die in der Trägerdatei enthaltenen Informationen entschlüsseln und für sich nutzbar machen.

„Vertraue keinem Verschlüsselungsverfahren, das du nicht selbst geknackt hast.“

Beispiel:

Nehmen wir folgendes Bild als Grundlage:

Nun nehmen wir pro Pixel den Hexadezimalen Farbcode und rechnen diesen in das Binäre Zahlensystem um. Dann ergeben sich folgende Werte für die ersten acht Pixel im Bild.



Ein hexadezimaler Farbcode hat 6 Ziffern (z.B.: #ed1c24). Immer zwei Ziffern bilden eine Farbe ab (Rot, Grün, Blau). Eine Hex-Ziffer kann 16 mögliche Zeichen annehmen. Sprich Ein Farbcode (z.B.: Rot) hat nun 16*16 = 256 Möglichkeiten (0-255). Für 256 Möglichkeiten benötigen wir insgesamt 8 Bits. Nachdem wir bei RGB drei Farben darstellen und beliebig kombinieren können, benötigen wir 3×8 = 24Bits. Somit kann man 3^24 = 16777216 = ca. 16,8 Mio. verschiedene Farben mit dem RGB-Modell darstellen.

```
1.Pixel: 111011010001110000100100
2.Pixel: 111011010001110000100100
3.Pixel: 111011010001110000100100
4.Pixel: 111011010001110000100100
5.Pixel: 111011010001110000100100
6.Pixel: 111011010001110000100100
7.Pixel: 111011010001110000100100
8.Pixel: 111011010001110000100100
```

Jetzt nehmen wir beispielsweise einen Text, welcher lautet „Das hier ist ein geheiner Text.“ und rechnen auch hier erstmal das erste Zeichen des Textes in das Binär-System um. Es ergeben sich hierbei folgende Werte für, in dem Fall, den Buchstaben „D“.

ASCII Table

Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char
0	0	0		32	20	40	[space]	64	40	100	@	96	60	140	`
1	1	1		33	21	41	!	65	41	101	A	97	61	141	a
2	2	2		34	22	42	"	66	42	102	B	98	62	142	b
3	3	3		35	23	43	#	67	43	103	C	99	63	143	c
4	4	4		36	24	44	$	68	44	104	D	100	64	144	d
5	5	5		37	25	45	%	69	45	105	E	101	65	145	e
6	6	6		38	26	46	&	70	46	106	F	102	66	146	f
7	7	7		39	27	47	'	71	47	107	G	103	67	147	g
8	8	10		40	28	50	(	72	48	110	H	104	68	150	h
9	9	11		41	29	51	)	73	49	111	I	105	69	151	i
10	A	12		42	2A	52	*	74	4A	112	J	106	6A	152	j
11	B	13		43	2B	53	+	75	4B	113	K	107	6B	153	k
12	C	14		44	2C	54	,	76	4C	114	L	108	6C	154	l
13	D	15		45	2D	55	-	77	4D	115	M	109	6D	155	m
14	E	16		46	2E	56	.	78	4E	116	N	110	6E	156	n
15	F	17		47	2F	57	/	79	4F	117	O	111	6F	157	o
16	10	20		48	30	60	0	80	50	120	P	112	70	160	p
17	11	21		49	31	61	1	81	51	121	Q	113	71	161	q
18	12	22		50	32	62	2	82	52	122	R	114	72	162	r
19	13	23		51	33	63	3	83	53	123	S	115	73	163	s
20	14	24		52	34	64	4	84	54	124	T	116	74	164	t
21	15	25		53	35	65	5	85	55	125	U	117	75	165	u
22	16	26		54	36	66	6	86	56	126	V	118	76	166	v
23	17	27		55	37	67	7	87	57	127	W	119	77	167	w
24	18	30		56	38	70	8	88	58	130	X	120	78	170	x
25	19	31		57	39	71	9	89	59	131	Y	121	79	171	y
26	1A	32		58	3A	72	:	90	5A	132	Z	122	7A	172	z
27	1B	33		59	3B	73	;	91	5B	133	[	123	7B	173	{
28	1C	34		60	3C	74	<	92	5C	134	\	124	7C	174	|
29	1D	35		61	3D	75	=	93	5D	135	]	125	7D	175	}
30	1E	36		62	3E	76	>	94	5E	136	^	126	7E	176	~
31	1F	37		63	3F	77	?	95	5F	137	_	127	7F	177	

Laut der ASCII-Tabelle hat der Buchstabe D den dezimalen Wert 68. Binär ergibt sich somit folgende Ziffernfolge:

```
01000100
```

Jetzt wirds interessant. Wir nehmen jetzt quasi pro Pixel im Bild die schwächste Bit (die letzte) und ändern es so um, wie wir es brauchen. Das heißt, bei dem ersten Pixel, nehmen wir, in dem Fall die Null und schauen, ob das mit dem ersten Bit des Buchstaben „D“ übereinstimmt.

Danach kommt der nächste Pixel. Auch hier nehmen wir das die schwächste Bit (auch wieder eine Null) und schauen ob dieses mit dem zweiten Bit des Buchstaben „D“ übereinstimmt. Da die zweite Bit des Buchstaben „D“ allerdings eine 1 ist ändern wir die letzte Bit des zweiten Pixels auf eine 1 ab. Somit verändert sich die Farbe fast garnicht. Hier ein vergleich der Farbe (vorher/nachher):

Vorher:

```
111011010001110000100100
```

Nachher:

```
111011010001110000100101
```

Dieses vorgehen wenden wir nun an jedem Pixel des Bildes vor, bis der ganze Satz im Bild versteckt ist. In folgendem Beispiel ist im Bild der folgende Text versteckt:

```
Steganographie ist im Grunde genommen eine Technik um Daten jeglicher Art zu
verstecken. Das Wort Steganographie stammt vom griechichen Wort „steganos“
ab, was so viel wie Verbergen heißt. Es gibt verschiedene Arten der
Steganographie. Die am verbreitetste Art der Steganographie ist mittlerweile
die technische Steganographie. Hier werden meistens bestimmte Daten
innerhalb eines Bildes versteckt.
Eine spezifische Art dieser Kunst/Technik wurde auch beim weltbekannten
Internet-Rätsel „Cicada 3301“ verwendet.
Im Folgenden werde ich versuchen eine bestimmte Art der Steganographie zu
erklären.
```

Originalbild:

Bild mit Steganografie:

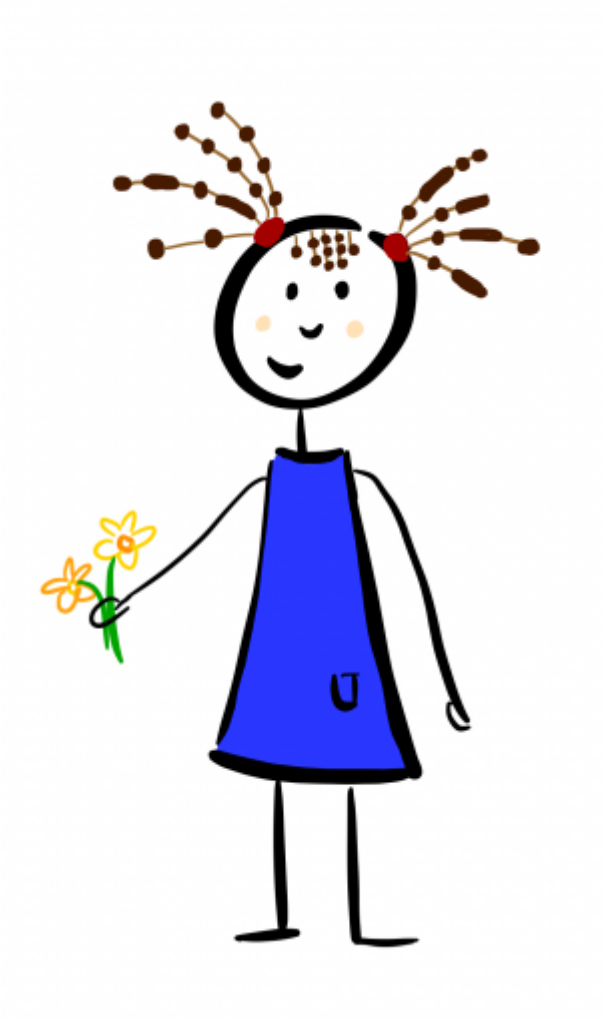
Wie man Sieht kann so eine Nachricht oder eine Datei vollkommen unerkannt übertragen werden. Diese Technik kann so weit geführt werden, dass ganze Bilder in anderen Bildern versteckt werden.

Je mehr Informationen in dem Grundbild sind desto Mehr Daten können darin versteckt werden.

Weitere Beispiele

In dem Bild des Mädchens wurde ihr Name verborgen. Die Nachricht ist mit einem bekannten Verfahren codiert.

Tipp: Um die Nachricht zu lesen, benötigst du den „Morse-Code“, bei dem jeder Buchstabe durch eine Kombination aus Punkten und Strichen ersetzt wird!



Aufgabe 1) Finde die verbotene Nachricht!

Aufgabe 2) Erstelle selbst ein Bild mit einer geheimen Nachricht!

Mehr zu Steganographie

Kryptografie

Die Kryptografie basiert auf mathematischen Verfahren. Die Sicherheit eines Kryptosystems lässt sich also mathematisch beweisen und berechnen. Die mathematische Beweisführung einer gewissen Sicherheit beruht jedoch oft nur auf Annahmen. Zum Beispiel: „Solange diese Bedingung erfüllt ist, ist dieses Verschlüsselungsverfahren sicher.“ Das hat Konsequenzen. Denn ein ungeschickt implementiertes Kryptosystem kann ein eigentlich sicheres Verschlüsselungsverfahren unsicher machen.

Wie sicher ein kryptografisches Verfahren ist, ist zu allen Zeiten immer zu optimistisch gewesen. Prinzipiell neigen wir zur Selbstüberschätzung, was die Sicherheit einer Technik angeht. Dabei zeigt die Erfahrung, dass kein Aufwand zu groß ist, um ein Verfahren zu brechen. Die Fragestellung ist nur, ob sich der Aufwand, in Erwartung des Inhalts verschlüsselter Daten, lohnt.

Kerckhoffs Prinzip

Die Sicherheit des Verschlüsselungsverfahrens beruht nur auf der Geheimhaltung des Schlüssels und nicht auf der Geheimhaltung des Verschlüsselungsverfahrens! Die Sicherheit eines Systems sollte nie allein von der Geheimhaltung der Funktionsweise abhängig sein (sonst: Security by Obscurity).

Das Gegenprinzip „security by obscurity“ besagt, dass man Sicherheit dadurch gewinnen will, indem man den Verschlüsselungsvorgang verschleiert. Dieses Gegenprinzip hat sich vielfach als wenig tauglich erwiesen. Verfahren kann man meist nicht geheimhalten (irgendjemand hält sich nicht an die Geheimhaltung). Zudem ist es oft möglich, durch eine Art Reverse-Engineering das benutzte Verfahren zu rekonstruieren.

Bei der Entwicklung neuer Verfahren versucht man daher gar nicht erst, die Verfahren selbst geheim zu halten. Im Gegenteil, die Verfahren werden zur öffentlichen Diskussion allen Expertinnen zur Verfügung gestellt. Nur die Verfahren, die eine solche Prüfung bestehen, haben eine Chance, in moderen Chiffriersystemen verwendet zu werden.

Gute kryptografische Verfahren erfüllen heute in der Regel also die folgenden Kriterien:

Sie beruhen auf dem Kerckhoffs-Prinzip. Sie werden von Kryptologinnen (bzw. -analytikerinnen) weltweit untersucht. Sie durchlaufen erfolgreich alle möglichen Angriffszenarien.

- 8.1.2.1) Symmetrische Kryptografie
  - 8.1.2.1.1) Cäsar-Chiffre
  - 8.1.2.1.2) Vigenere-Chiffre
  - 8.1.2.1.3) One Time Pad (Vernam-Chiffre)
  - 8.1.2.1.4) Skytale
  - 8.1.2.1.5) DES
  - 8.1.2.1.6) AES
- 8.1.2.2) Asymmetrische Kryptografie
  - 8.1.2.2.1) RSA
- 8.1.2.3) Hybride Chiffriersysteme

Symmetrische Kryptografie/Verschlüsselung

Die Verschlüsselungsverfahren, die mit einem geheimen Schlüssel arbeiten, der zum Ver- und Entschlüsseln dient, nennt man symmetrische Verfahren oder Secret-Key-Verfahren. Üblich sind auch die Bezeichnungen Secret-Key-Kryptografie und Secret-Key-Verschlüsselung. Fast alle symmetrischen Verfahren sind auf ressourcenschonende Umgebungen optimiert. Sie zeichnen sich durch geringe Hardwareanforderungen, geringen Energieverbrauch und einfache Implementierung in Hardware aus.

Prinzip

Die Verschlüsselungsverfahren der symmetrischen Kryptografie arbeiten **mit einem einzigen Schlüssel**, der bei der Ver- und Entschlüsselung vorhanden sein muss. Diese **Verfahren sind schnell** und bei entsprechend **langen Schlüsseln bieten sie auch eine hohe Sicherheit**.

Der **Knackpunkt liegt in der Schlüsselübergabe** zwischen den Kommunikationspartnern. Vor der sicheren Datenübertragung mit Verschlüsselung müssen sich die Kommunikationspartner auf den Schlüssel einigen und austauschen. Wenn der Schlüssel den selben Kommunikationspfad nimmt, wie die anschließend verschlüsselten Daten, dann besteht die Gefahr, dass ein Angreifer in Besitz des Schlüssels gelangt, wenn er die Kommunikation abhört. Wenn der Angreifer den Schlüssel hat, dann kann er nicht nur die Daten entschlüsseln, sondern auch selber Daten verschlüsseln, ohne das es die Kommunikationspartner bemerken. Knackpunkt ist der unsichere Schlüsselaustausch und die Authentifizierung der Kommunikationspartner.

Sicher ist die Schlüsselübergabe nur dann, wenn sich zwei Personen persönlich treffen und den Schlüssel austauschen oder der Schlüssel einen anderen Weg nimmt (Seitenkanal), wie es die Daten tun. Ein Möglichkeit wäre der postalische Weg (Brief, Einschreiben mit Rückschein). Allerdings nicht per E-Mail (Postkarten-Effekt). Zur Unsicherheit trägt außerdem bei, wenn einer der Kommunikationspartner den Schlüssel nur ungenügend sicher aufbewahrt.

Der sichere Schlüsselaustausch ist eines der vielen Probleme der Kryptografie. Mit der asymmetrischen Kryptografie versucht man dieses Problem zu lösen. Weil die asymmetrische Kryptografie weit komplexere Verfahren umfasst, kombinieren die übliche kryptografischen Protokolle sowohl symmetrische als auch asymmetrische Verfahren.

Vorteile

- Gleicher Schlüssel zum Verschlüsseln und Entschlüsseln
- Je zwei Teilnehmer benötigen einen Schlüssel
- Beide müssen den Schlüssel stets geheim halten
- Anzahl der Schlüssel wächst quadratisch mit der Teilnehmerzahl

Nachteile

- Sichere Verteilung des Schlüssels (Telefon, schriftlich,...)
- Nicht geeignet für Digitale Signatur

Symmetrische Verschlüsselungsverfahren

Jede symmetrische Verschlüsselung basiert auf einem bestimmten Algorithmus. Bei einem Verschlüsselungsalgorithmus bzw. Chiffre wird in den Klartext eine Geheiminformation, den Schlüssel, eingebracht und so der Geheimtext gebildet. Der Schlüssel kann ein Passwort, eine geheime Nummer oder auch nur eine zufällige Bitfolge sein.

Monoalphabetische Substitutionschiffren

Die einfachste Art der Verschlüsselung erreicht man, in dem man jeden Buchstaben ein festes Symbol zuordnen. Diese Verfahren sind monoalphabetisch. Sie sind bei genügend Verschlüsselungsmaterial leicht durch eine Häufigkeitsanalyse zu brechen. In jeder Schriftsprache kommen bestimmte Buchstaben häufiger vor. Man kann also mit einfachen statistischen Mitteln eine Kryptoanalyse machen. Mit Computer-Unterstützung geht es automatisch und noch schneller.

- 8.1.2.1.1) Cäsar-Chiffre

Polyalphabetische Substitutionschiffren

Wesentlich schwieriger sind polyalphabetische Geheimtexte. Hier kann ein Buchstabe mehreren Symbole entsprechen. Statistische Verfahren funktionieren hier nicht mehr so einfach.

- 8.1.2.1.2) Vigenere-Chiffre
- 8.1.2.1.3) One Time Pad (Vernam-Chiffre)

Permutationschiffren

Eine Umordnung, eine Permutation einer gegebenen Zeichenfolge, nennt man Permutations- oder Transpositionschiffre. Dies trifft in diesem Fall auf die Skytale zu. Permutationschiffren werden auch als Transposition bezeichnet. Die Skytale ist ein Spezialfall der Transposition. Denkbar wäre nämlich eine Permutationschiffre, die zur Erstellung des Geheimtextes erst den ersten, dann den 47-ten, danach den 32-ten Buchstaben nimmt, usw. Bei der Skytale wird jedoch, wie oben als Matrix betrachtet, die Nachricht zeilenweise aufgetragen und chiffriert liegt diese spaltenweise vor. Die Skytale ist also letztendlich eine einfache Matrixtransposition.

Bei einer Permutations-Chiffre werden somit die Buchstaben den Klartext nicht ersetzt sondern durcheinander gewürfelt. Fast man zB immer 5 Buchstaben des Klartextes zusammen und lässt sich durch die Permutations-Chiffre mit dem Schlüssel (4,1,2,5,3), dann erhält man zB folgenden Chiffretext:

```
IE SBGE TZIW EARNT LVU ENNUTS: EHOLEC, ZDI UE EENB DGRIENS; NWS ANI
EFAEANNG.

ES GIBT ZWEI ARTEN VON LEUTEN: SOLCEH DIE ZU ENDE BRINGEN, WAS SIE ANFANGEN.
```

Die Art der Verschlüsselung lässt sich durch Probieren – abhängig von der Schlüssellänge – mehr oder weniger schnell knacken. Auch die Häufigkeitsanalyse liefert wieder Rückschlüsse über die verwendete Sprache etc.

- 8.1.2.1.4) Skytale

Operationen

Alle gängigen symmetrische Verfahren arbeiten ausschließlich mit Bit-weisen Operationen. Hier werden Schlüssel, Klartext und Geheimtext in Form von Bitfolgen verarbeitet. In dem die Funktionen nahezu beliebig miteinander kombiniert werden, lassen sich neu symmetrische Verfahren in nahezu beliebiger Zahl entwickeln und mit bekannten Angriffen auf Schwächen testen. In der Regel kombinieren symmetrische Verschlüsselungsalgorithmen Substitutionschiffren und Permutationschiffren miteinander und wiederholen den Vorgang mehrmals (Runden), wobei eine härtere Verschlüsselung entsteht. Typische Bestandteile von symmetrischen Verschlüsselungsalgorithmen sind:

- Exklusiv-oder-Verknüpfung
- Permutation: Reihenfolge einer Bit-Folge wird verändert.
- Substitution: Eine Bit-Folge wird durch eine andere ersetzt.

Erfahrungsgemäß sind für eine wirkungsvolle Verschlüsselung keine aufwendigen Funktionen notwendig. Insbesondere beim Hardware-nahen Programmieren oder der Implementierung in Hardware ist das von Vorteil, weil sich so eine hohe Geschwindigkeit erreichen lässt. Beim praktischen Einsatz von Verschlüsselungsalgorithmen stellt sich auch immer die Frage, wie groß die Rechenleistung für die Verschlüsselung ist. Generell gilt, je schneller ein Verschlüsselungsverfahren arbeitet, desto niedriger sind die Hardwarekosten.

Moderne(re) Verschlüsselungsverfahren

Bei den symmetrischen Verschlüsselungsverfahren gilt der AES als Maß der Dinge. Es gibt aber auch weitere...

- 8.1.2.1.5) DES
- 8.1.2.1.6) AES
- 3DES - Triple DES
- IDEA - International Data Encryption Algorithm
- RC4 (Rivest-Cipher 4)

- Blowfish (von Bruce Schneier)
- RC5, RC5a, RC6 (Rivest-Cipher 5 bzw. 5a bzw. 6)
- A5 (GSM)
- Serpent
- Twofish (von Bruce Schneier)
- MARS
- SAFER/SAFER+
- CAST (Carlisle Adams und Stafford Tavares)
- MAGENTA
- MISTY1
- Camellia
- Ascon

Quellen

- Elektronik Kompendium
- Kryptografie.de
- Cryptool
- Wikipedia

Cäsar Chiffre

Die Cäsar-Chiffre ist eines der einfachsten, aber auch unsichersten Verfahren, um Texte zu verschlüsseln. Das Verfahren wurde nach dem römischen Kaiser Julius Cäsar benannt, der auf diese Weise bereits vor über 2000 Jahren Nachrichten verschlüsselt haben soll.

Die Cäsar-Chiffre ist eine monoalphabetische Substitution, das heißt, jeder Buchstabe des Textes wird durch genau einen anderen Buchstaben des Alphabets ersetzt. Dieser Austausch geschieht jedoch nicht zufällig, sondern basiert auf zyklischer Rotation des Alphabets um *k* Zeichen, wobei *k* der verwendete Schlüssel ist.

Als eines der einfachsten und unsichersten Verfahren dient es heute hauptsächlich dazu, Grundprinzipien der Kryptologie anschaulich darzustellen. Der Einfachheit halber werden oftmals nur die 26 Buchstaben des lateinischen Alphabets ohne Unterscheidung von Groß- und Kleinbuchstaben als Alphabet für Klartext und Geheimtext verwendet und Sonderzeichen, Satzzeichen usw. nicht beachtet.

Verschlüsselung

Die Verschlüsselung einer Nachricht erfolgt buchstabenweise mit einem Schlüssel *k* aus der Menge Z_{26} = *{0,1,2,3...25}*, wobei der Wert *k = 0* nicht sinnvoll ist, da der Originaltext in diesem Fall keine Änderung erfährt.

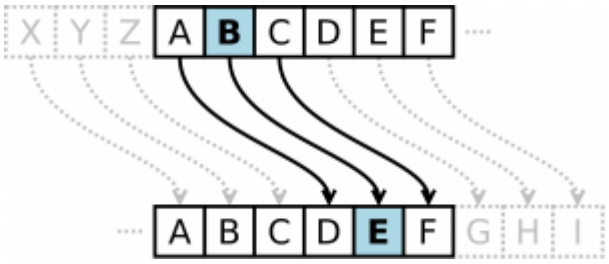
Für einen gegebenen Buchstaben wird zunächst anhand der folgenden Tabelle seine Position *m* im Alphabet bestimmt.

A	B	C	D	E	F	G	H	I	J	K	L	M
0	1	2	3	4	5	6	7	8	9	10	11	12
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
13	14	15	16	17	18	19	20	21	22	23	24	25

Anschließend erhält man den Wert *c* des verschlüsselten Buchstaben durch folgende kurze Berechnungsformel:

```
c = (m+k) mod 26
```

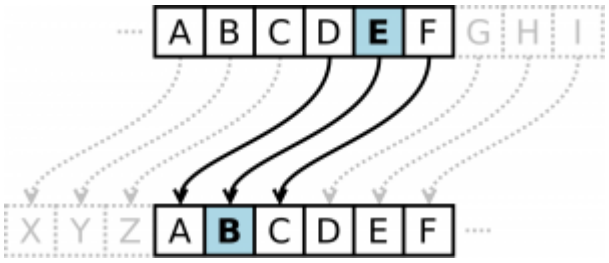
Mit Hilfe obiger Tabelle kann dieser Wert *c* wieder in einen Buchstaben transformiert werden.



Entschlüsselung

Die Entschlüsselung einer Nachricht erfolgt ähnlich wie die Verschlüsselung mit Schlüssel , wir verwenden jedoch die Formel:

```
m = (26+c-k) mod 26
```



Beispiel

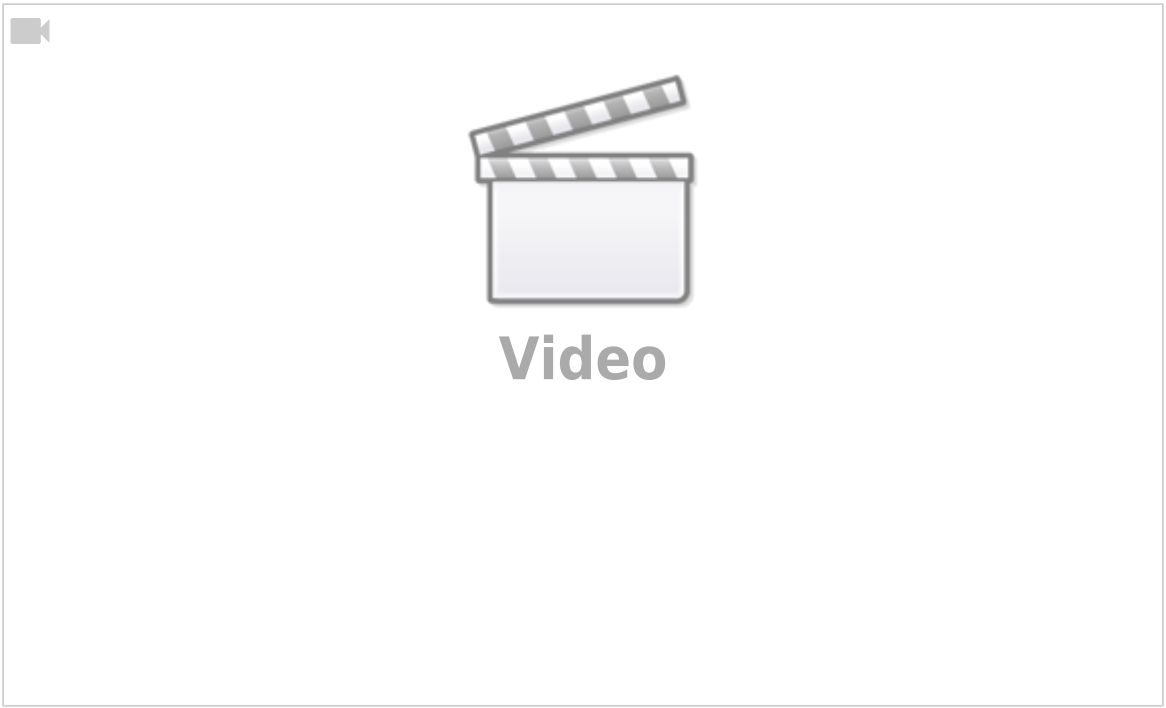
Der Satz „OTTO KOMMT“ wird mit dem Schlüssel *k=3* verschlüsselt.

unverschlüsselt	O	T	T	O		K	O	M	M	T
m	14	19	19	14		10	14	12	12	19
$c \equiv (m+k) \bmod 26$	17	22	22	17		13	17	15	15	22
verschlüsselt	R	W	W	R		N	R	P	P	W

Statt nur über dem Alphabet Z_{26} kann man analog allgemeine Cäsar-Chiffre über beliebigen endlichen Alphabeten Z_{26} = {0,1,..a-1} definieren.

Klartext: H A L L O

Geheimtext: K D O O R



Vignere Chiffre

Die Vigenère-Chiffre (auch: Vigenère-Verschlüsselung) ist eine aus dem 16. Jahrhundert stammende Handschlüsselmethode zur Verschlüsselung von geheim zu haltenden Textnachrichten.

Es handelt sich um ein monographisches polyalphabetisches Substitutionsverfahren. Der Klartext wird in Monogramme (Einzelzeichen) zerlegt und diese durch Geheimtextzeichen substituiert (ersetzt), die mithilfe eines Kennworts aus mehreren (poly) unterschiedlichen Alphabeten des „Vigenère-Quadrats“ ausgewählt werden. Dabei handelt es sich um eine quadratische Anordnung von untereinander stehenden verschobenen Alphabeten (siehe Bild).

Die Vigenère-Chiffre steht im Gegensatz zu den einfacheren monoalphabetischen Substitutionsmethoden, bei denen nur ein einziges (mono) Alphabet verwendet wird. Aufgrund ihrer für die damalige Zeit als besonders hoch eingeschätzten kryptographischen Sicherheit wurde sie auch als le chiffre indéchiffrable (frz. für „die unentzifferbare Chiffre“) bezeichnet, eine aus damaliger Sicht vielleicht zutreffende, aber aus heutiger Sicht falsche Beurteilung.

Methode

Ausgehend vom Standardalphabet mit seinen 26 Großbuchstaben werden alle möglichen Caesar-verschobenen Alphabete daruntergeschrieben. Man erhält eine quadratische Anordnung von 26 × 26 Buchstaben, ursprünglich als Tabula recta, später auch als carré de Vigenère (frz. für „Vigenère-Quadrat“) bezeichnet. In der folgenden Darstellung sind der Deutlichkeit halber oberhalb des eigentlichen Quadrats eine Zeile mit den Klartextbuchstaben und links eine Spalte mit den Schlüsselbuchstaben ergänzt worden, die prinzipiell nicht benötigt werden.

Vigenère-Quadrat

		Klartext																							
		A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	**W**	X	Y	Z
Schlüssel	A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
	B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
	C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
	D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
	E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
	F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
	G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
	H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
	I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
	J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
	K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
	L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
	M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
	N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
	O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
	P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
	Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
	R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
	S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
	T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
	U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
	V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
	W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
	X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
	Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
	Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

Zur Verschlüsselung eines Klartextes wie beispielsweise des Satzes „Werde Mitglied bei Wikipedia“ benötigt der Verschlüssler zunächst einen Schlüssel. Idealerweise sollte dieser möglichst lang sein und aus einer möglichst „zufälligen“ Buchstabenfolge bestehen. Erreicht die Länge des Schlüssels die des Klartextes und wird der Schlüssel nicht mehrfach verwendet, dann erhält man ein tatsächlich „unknackbares“ Verfahren, wie es aber erst Jahrhunderte später, im Jahr 1882, vom amerikanischen Kryptologen Frank Miller (1842–1925) vorgeschlagen wurde, und das heute als One-Time-Pad (Abkürzung: OTP, deutsch: „Einmalschlüssel-Verfahren“) bezeichnet wird. Zur Zeit von Vigenère und noch bis ins 20. Jahrhundert hinein wurden allerdings regelmäßig relativ kurze und häufig auch leicht

zu erratende Schlüssel benutzt, die zudem mehrfach verwendet wurden. Ein Beispiel wäre die Verwendung von WILLKOMMEN als Schlüsselwort.

Als praktisches Hilfsmittel kann der Verschlüssler den zu verschlüsselnden Text in eine Zeile schreiben und darüber das Kennwort so oft wiederholen, wie es nötig ist:

```
WILLKOMMEN WILLKOMMEN WILLK
WerdeMitgl iedbeiWiki pedia
```

Die entsprechenden Geheimtextbuchstaben kann er nun leicht mithilfe des Vigenère-Quadrats ermitteln. Dazu sucht er den Kreuzungspunkt der durch den jeweiligen Schlüsselbuchstaben gekennzeichneten Zeile und der Spalte des Quadrats, die oben durch den Klartextbuchstaben gekennzeichnet ist. Beispielsweise zur Vigenère-Verschlüsselung des ersten Buchstabens W des Textes sucht er den Kreuzungspunkt der Zeile W mit der Spalte W und findet als Geheimtextbuchstaben das S. Der auf diese Weise vollständig verschlüsselte Geheimtext lautet:

```
SMCOOAUFKY EMOMOWIUOV LMOTK
```

Üblicherweise wird er in Gruppen fester Länge, beispielsweise in Fünfergruppen übertragen. Diese Maßnahme dient auch dazu, die Länge des Kennworts (hier zehn) nicht zu verraten. Der zu übermittelnde Geheimtext lautet hier:

```
SMCOO AUFKY EMOMO WIUOV LMOTK
```

Der befugte Empfänger ist, wie der Absender, im Besitz des geheimen Kennworts (hier: WILLKOMMEN) und kann durch Umkehrung der oben beschriebenen Verschlüsselungsschritte aus dem Geheimtext durch Entschlüsselung mithilfe des Kennworts den ursprünglichen Klartext wieder zurückgewinnen:

```
SMCOOAUFKYEMOMOWIUOVLMOTK
WILLKOMMENWILLKOMMENWILLK
WERDEMITGLIEDBEIWIKIPEDIA
```

Vigenere Chiffre Erklärung

Kryptoanalyse

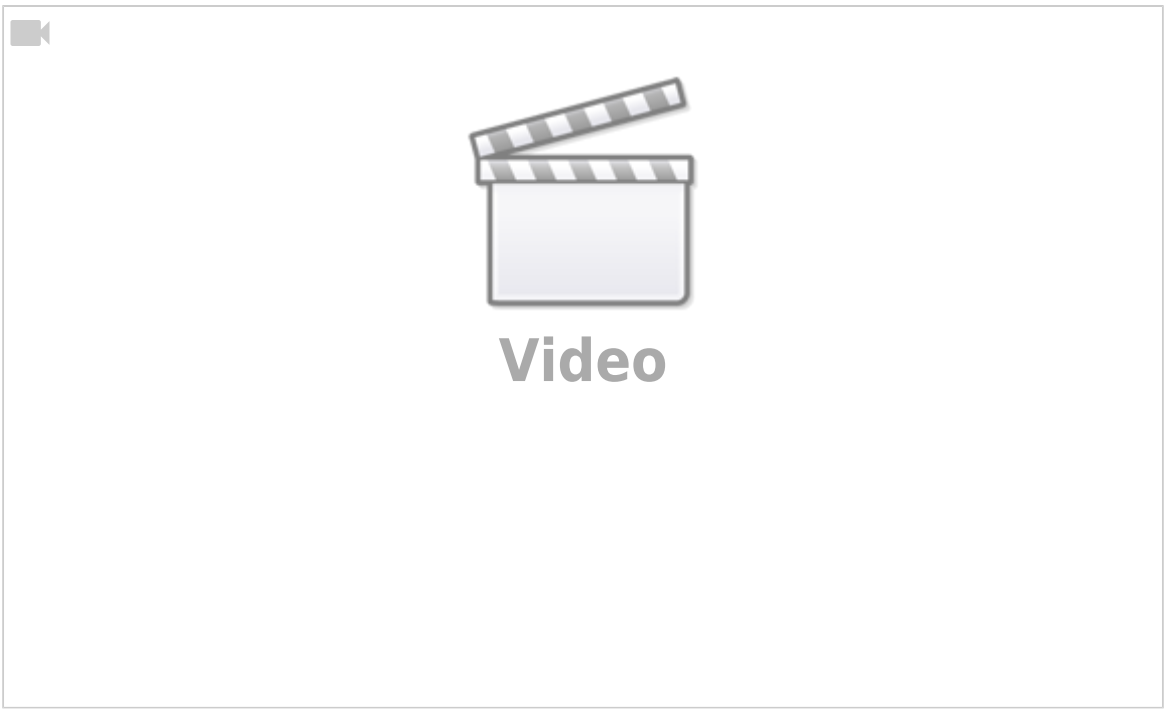
Vorteile einer polyalphabetischen Methode wie der Vigenère-Chiffre gegenüber den in den damaligen Jahrhunderten üblichen einfachen monoalphabetischen Methoden – dazu gehören auch die damals sehr beliebten Nomenklatoren – ist das durch die Verwendung von vielen unterschiedlichen Alphabeten bewirkte Abschleifen des bei den monoalphabetischen Verfahren so verräterischen Häufigkeitsgebirges. Der systematische Wechsel der Alphabete stärkt das Verfahren gegenüber statistischen Angriffsmethoden. Auch der erst im 20. Jahrhundert entwickelte Koinzidenzindex, ein universell einsetzbares kryptanalytisches Hilfsmittel, wird bei polyalphabetischen Verfahren wesentlich abgeschwächt. Lange wurde – abgesehen von Ausnahmen, in denen der Codeknacker das Schlüsselwort oder Teile des Klartextes erraten konnte – keine systematische Angriffsmethode gegen die Vigenère-Verschlüsselung gefunden, die sich über die Jahrhunderte den Ruf einer „unknackbaren Chiffre“ erwarb. Dennoch wurde sie nur selten verwendet und stattdessen lieber auf die althergebrachten Verfahren, wie Nomenklatoren, zurückgegriffen, wohl auch, weil viele Anwender die

Chiffre als zu kompliziert in der Anwendung empfanden.

Im Jahr 1854 fand der englische Wissenschaftler Charles Babbage (1791–1871) eine Lösung der Chiffre, die er jedoch nie publizierte. Der Erste, der eine allgemeingültige Angriffsmethode auf die Vigenère-Chiffre beschrieb, war der preußische Infanteriemajor und Kryptologe Friedrich Wilhelm Kasiski (1805–1881). Er veröffentlichte 1863 in Berlin sein Buch „Die Geheimschriften und die Dechiffrir-Kunst“ und erläuterte darin seine Idee zur Entzifferung von Vigenère-verschlüsselten Texten. Seine Entzifferungsmethode ist noch heute unter seinem Namen als Kasiski-Test bekannt. Als Erstes ist die Länge des verwendeten Schlüsselworts zu ermitteln. Dazu durchsuchte Kasiski den Geheimtext nach Buchstabenfolgen der Länge zwei (Bigramme) oder länger (Trigramme, Tetragramme etc.), die mehrmals vorkommen, genannt: „Doppler“. Anschließend bestimmte er den Abstand zwischen den Dopplern. Er erzeugte so eine möglichst vollständige Liste mit im Geheimtext auftretenden Dopplern und deren Abständen. In dieser suchte er mithilfe der Faktorisierung (Primfaktorzerlegung) nach gemeinsamen Längen, um so auf die vermutliche Schlüsselwortlänge zu schließen. Im Cryptologia-Artikel Breaking Short Vigenère Ciphers (siehe Literatur) ist die wichtige Seite 41 aus Kasiskis Buch abgebildet.[9] Nach der Untersuchung seines Vigenère-verschlüsselten Beispieltextes mit 180 Buchstaben zieht er das Fazit: „Hier kommt der Faktor 5 am häufigsten vor, der Schlüssel muß demnach 5 Buchstaben enthalten.“

Hat man die Schlüssellänge gefunden, so kann man im zweiten Schritt der Entzifferung den Geheimtext in seine Bestandteile zerlegen, die mit jeweils demselben Alphabet verschlüsselt wurden. In Kasiskis Beispielfall würde man den ersten, sechsten, elften Buchstaben und so fort als erste Gruppe betrachten. Die zweite Gruppe besteht aus dem zweiten, siebten, zwölften Buchstaben und so fort. Die dritte aus dem dritten, achten, dreizehnten und so weiter. Innerhalb jeder Gruppe liegt eine einfache Caesar-Verschlüsselung vor, die mithilfe der Häufigkeitsanalyse leicht zu knacken ist. In vielen Fällen entspricht schlicht der am häufigsten auftretende Geheimtextbuchstabe jeder Gruppe dem Klartext-„e“, also dem in den meisten europäischen Sprachen häufigsten Buchstaben. Hat man das „e“ identifiziert, dann ergeben sich unmittelbar alle anderen Buchstaben, denn die Vigenère-Chiffre benutzt ja nur verschobene Alphabete und keine verwürfelten, wie es der Namensgeber eigentlich vorgeschlagen hatte.

Nach „Rohrbachs Forderung“ sollte der Codeknacker zum Schluss seiner Arbeit noch versuchen, das Schlüsselwort zu erschließen. Erst dann gilt seine Arbeit als erfolgreich beendet. Im Idealfall gelingt ihm dies einfach mit Kenntnis des Klartextes durch anschließendes direktes Ablesen im Quadrat. In der Praxis wurden jedoch nicht immer plump einfache Wörter als Schlüssel benutzt. Dann gilt es, auch noch den Algorithmus zu erschließen, nach dem der Verschlüssler das Schlüsselwort (beispielsweise aus einem Merksatz) bildet und möglichst auch, wie und in welchem Rhythmus er es wechselt.



One Time Pad (Vernam-Chiffre)

Das One Time Pad (Abk. OTP, dt. Einmalverschlüsselung) Verschlüsselungsverfahren, auch Vernam Verschlüsselung nach seinem Erfinder.

Vernam arbeitete bei der US-amerikanischen Telefongesellschaft AT&T und war dort mit der damals noch neuen Fernschreiber-Technik betraut. Ein Problem war, dass man Fernschreiben leicht abhören konnte - besonders, wenn diese per Funk übertragen wurden. 1917 hatte er eine Idee für die Lösung: er wollte die Bits des Baudot-Codes, den man damals für Fernschreiben nutzte und die aus einer Null oder einer Eins bestanden, verschlüsseln, indem er jedes Bit mit einem zufälligen, anderen Bit kombinierte. Dazu benutzte er einen zweiten Lochstreifen mit Zufallsmuster zum ersten mit der Botschaft. Zuerst nahm er nur einen kurzen Lochstreifen, dessen Ende er an den Anfang klebte und so einen sich wiederholenden Endlosstreifen erhielt. Doch dann merkte er, dass absolute Sicherheit nur ein Schlüsselstreifen bieten konnte, der genau so lang war wie der Lochstreifen mit dem Klartext.

Der amerikanische Major (und später General) Joseph O. Mauborgne setzte die Idee 1918 als Erster für militärische Zwecke um und erweiterte sie um die Prämisse, dass ein Schlüsselcode zufällig und nur einmal benutzt werden darf. Das Verfahren wurde als One-time-system bekannt. Aus Gründen der Praktikabilität verwendete er allerdings einen sich wiederholenden Schlüssel. Alsbald beschäftigten sich auch die Deutschen mit dem Verfahren und setzten es im diplomatischen Dienst der Weimarer Republik ein.

One Time Pad ist also ein Verfahren, bei dem jedes Zeichen des Klartextes mit einem Zeichen eines Schlüssels kombiniert wird, um zu einem Chiffrat zu gelangen. Dies bedeutet aber auch, dass der Schlüssel genau so lang sein muss wie der zu verschlüsselnde Text.

Damit das Verfahren sicher ist, ist es außerdem wichtig, dass der Schlüssel rein zufällig ist und dass der Schlüssel nur ein einziges mal verwendet wird. Denn würde der Schlüssel zweimal verwendet und wäre dem Gegner bekannt, dass zweimal derselbe Schlüssel für zwei unterschiedliche Klartexte verwendet wurden, so ließe sich ein Datenstrom aus den Differenzen erstellen, der wiederum durch Häufigkeitsanalyse der verwendeten Zeichen angreifbar wäre.

Von Prinzip her könnte man die One Time Pad Verschlüsselung auch als polyalphabetische Substitution bezeichnen, bei dem für jedes Zeichens des Klartextes ein anderer Schlüssel verwendet wird.

Auf der anderen Seite stellt die Länge des Schlüssels doch einige Anforderungen bei längeren Texten, so dass der Schlüssel wohl zumeist der Output eines Pseudo-Zufallsgenerators sein wird, wobei dann der Terminus Stromchiffre wieder passen würde.

Ein Vorteil des One Time Pad Verfahrens ist außer der Sicherheit bei richtiger Anwendung auch, dass es leicht mit Papier und Bleistift bewerkstelligt werden kann. So war es im kalten Krieg unter Geheimdiensten oft eingesetzt. Dabei wurden die Zeichen einer Geheimbotschaft mittels Dekodierschablonen zu Ziffern umgewandelt, die dann mittels langen Ziffernkolonnen in einem Heft oder Block, sogenannte Wurmtabellen kombiniert wurden.

Z. B.: Klartext 6, Schlüsselziffer 7 ergibt 13 → 3 (Addition Modulo 10). Mit der Umkehrrechnung (Subtraktion Absolut), hier also 7 - 13 = -6 → 6 konnte dann wieder auf den Klartext entschlüsselt werden. Ein einmal verwendete Wurmtabelle wurde nach dem Verschlüsseln dann nach einem festen Muster (z. B. alle angefangenen Blätter) vernichtet.

Auch der Heiße Draht (das sogenannte Rote Telefon) zwischen dem amerikanischen Präsidenten und dem sowjetischen Generalsekretär wurde durch ein One Time Pad Verfahren gesichert.

Das OTP lässt sich auch einfach per Computer realisieren. Die Bits der dort vorliegende Binärdaten werden dann aber meistens mittels XOR verknüpft, weil dies weniger Rechenoperationen erfordert. Außerdem ist XOR eine reversible Operation und kann so für Ver- und Entschlüsselung zugleich eingesetzt werden.

Der Hauptnachteil des OTP in der modernen Umgebung liegt in der erforderlichen Schlüssellänge. Wollte man zum Beispiel eine gesamte Festplatte verschlüsseln, so bräuchte man eine zweite, mindestens genau so große, die den Schlüssel enthält. Noch dazu müsste der Schlüssel aus echten Zufallszahlen und nicht aus berechneten Pseudozufallszahlen bestehen, um wirklich sicher zu sein. Dies würde viel Aufwand bedeuten. Außerdem kann der Schlüssel bei dieser Größe nicht mehr gemerkt werden, so dass er an Medien gebunden ist, die dem Feind in die Hände fallen könnten.

Darum hat das OTP Verfahren in der Moderne zunehmend an Bedeutung verloren, insbesondere, wenn größere Datenmengen verschlüsselt werden müssen.

Beispiel

Eine einfache Handmethode zur Verschlüsselung ist beispielsweise die buchstabenweise Addition von Klartext und Schlüssel. Hierzu ersetzt man zunächst mithilfe einer beliebigen Substitutionstabelle die Buchstaben des Klartextalphabets durch Zahlen. Im einfachsten Fall ordnet man den 26 Großbuchstaben des lateinischen Alphabets Zahlen zu, die ihrer Position im Alphabet entsprechen. Mit anderen Worten, man nummeriert das Alphabet wie folgt durch:

```
A B C D E F G H I J  K  L  M  N  O  P  Q  R  S  T  U  V  W  X  Y  Z
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26
```

Jetzt ist eine buchstabenweise Addition leicht möglich. Beispielsweise ergibt die Addition von A und F den Buchstaben G, entsprechend ihren Platznummern 1 + 6 = 7. Falls die Summe den Wert 26 überschreiten sollte, so zieht man einfach 26 ab (Modulo-Operation) und erhält so wieder einen der 26 Alphabetbuchstaben. Beispielsweise X plus U ist numerisch 24 + 21 = 45, nach Abziehen von 26 ergibt sich 19 und damit der Buchstabe S, also X + U = S.

Die Zusammenhänge bei der Addition von Buchstaben lassen sich an der folgenden Tabelle, die Ähnlichkeit mit einer klassischen Tabula recta (Vigenere Quadrat) hat, übersichtlich darstellen.

Zur Verschlüsselung wird man einen zufälligen Schlüssel benutzen, der in diesem Beispielfall passenderweise ebenfalls aus den 26 Großbuchstaben zusammengesetzt ist und dessen Länge (mindestens) der Länge des zu verschlüsselnden Klartextes entspricht. Entscheidend für die Sicherheit der Verschlüsselung ist, dass die einzelnen Buchstaben des Schlüssels wirklich zufällig verteilt sind, unvorhersagbar sind und in keinerlei Zusammenhang untereinander stehen. Als Beispiel für einen zufälligen Schlüssel dient die folgende Buchstabenfolge:

```
S = WZSLXWMFQUDMPJLYQOXXB
```

Der Schlüssel S ist in diesem Beispiel recht kurz, er umfasst nur 21 Buchstaben und ist bei bestimmungsgemäßer Verwendung sehr schnell verbraucht, nämlich bereits nach Verschlüsselung eines Textes aus 21 Buchstaben.

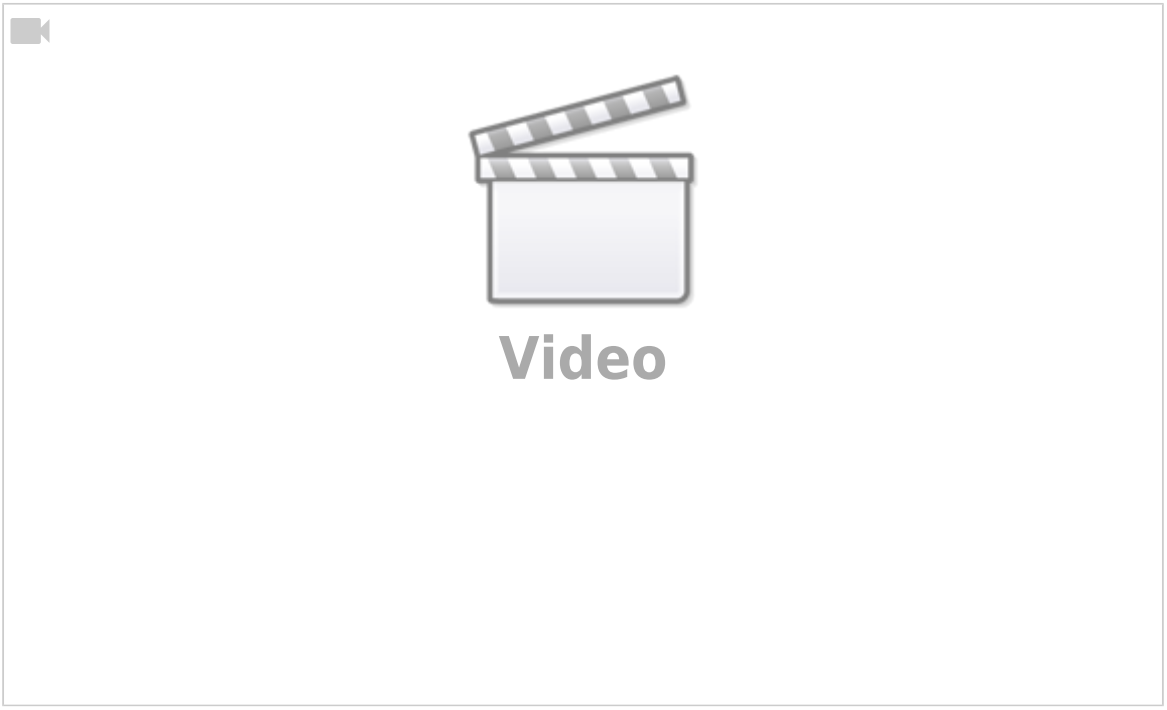
Beispielsweise soll der folgende Klartext K verschlüsselt werden:

```
K = ANGRIFFIMMORGENGRAUEN
```

Zur Verschlüsselung werden Klartext K und Schlüssel S, wie oben erläutert, buchstabenweise addiert. Als Summe (K + S = G) erhält man nach der so durchgeführten Einmalverschlüsselung den Geheimtext G:

```
G = XNZDGCSODHSEWOZFIPSCP
```

Der im Ergebnis erhaltene Geheimtext G ist von einem Zufallstext nicht zu unterscheiden und kann prinzipiell mit keiner noch so gearteten kryptanalytischen Angriffsmethode (weder jetzt noch in Zukunft) entziffert werden. Allein die Kenntnis des Schlüssels S erlaubt es, aus dem Geheimtext G durch Subtraktion des Schlüssels wieder den Klartext K zu gewinnen. Ohne den Schlüssel kann man prinzipiell alle denkbaren und mehr oder weniger sinnvollen Buchstabenkombinationen aus 21 Buchstaben konstruieren. Theoretisch könnte ein Angreifer dies probieren. Das wären aber 26^21 = 518 131 871 275 444 637 960 845 131 776 Möglichkeiten.



Skytale

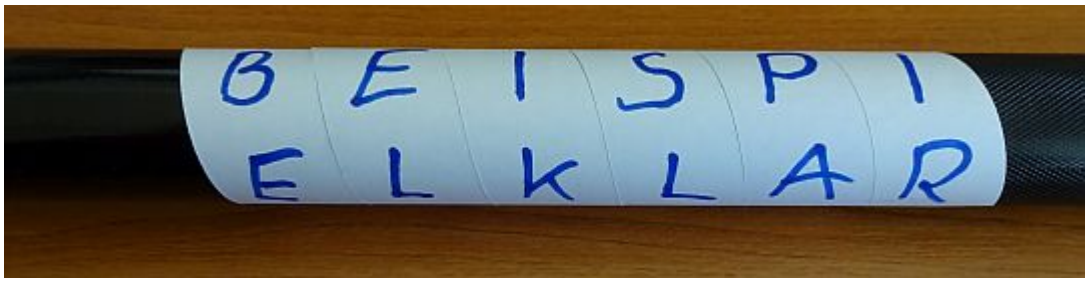
Die Skytale von Sparta (griech.: „scytale“; Stock, Stab) ist das älteste (5. Jh. v. Chr.) bekannte militärische Verschlüsselungsverfahren und basiert auf einem Stock mit einem bestimmten Durchmesser, auf den ein Lederstreifen wendelförmig gewickelt wurde. Dann wurde die Nachricht quer über den Stab auf das Leder geschrieben. Nach dem Abwickeln waren dann alle Buchstaben durcheinander und konnten erst wieder gelesen werden, wenn sie um einen Stab mit dem richtigen Durchmesser gewickelt wurden.

Der Durchmesser entspricht dem Versatz, also dem Schlüssel dieser Transpositions-Chiffre. Bitte beachten Sie, dass auch Leerzeichen mitkodiert werden, da diese ja eine Leerraum darstellen und somit einen Versatz bedeuten. Sollen keine Leerzeichen mitkodiert werden, löschen Sie diese vorher.

Die Chiffre ist nicht sonderlich sicher, denn man einfach Stöcke verschiedener Durchmesser ausprobieren. Oder mathematisch die Versätze durchrechnen. Dann muss man nur noch den Klartext erkennen.

Den Lederstreifen, auf den die Nachricht geschrieben war, konnte man auch umgedreht als Gürtel tragen, so dass er nicht weiter auffiel. Damit war die Skytale auch eine frühe Form der Steganografie.

Beispiel



Klartext:

```
BEISPIELKLARTEXT
```

Schlüssel:

```
6
```

Kodiert:

```
BETELEIKXSLTPAIR
```

Chiffrierung Versatz 6:

```
1 2 3 4 5 6
B E I S P I
E L K L A R
```

```
T E X T
^ ^ ^ ^ ^ ^ --- spaltenweise auslesen

BET ELE IKX SLT PA IR
```

Dechiffrierung per Bruteforce:

```
01: BETELEIKXSLTPAIR
02: BXESTLETLPEAIIKR
03: BITEKPTXAESILLRE
04: BLXPEESATILIEKTR
05: BLKLAEEXTITISPRE
06: BEISPIELKLARTEXT
07: BEIXLPIELKSTARTE
08: BTLIXLPIEEEKSTAR
09: BTLIXLPIREEEKSTA
10: BTLIXLPAIREEEKST
11: BTLIXLTPAIREEEKS
12: BTLIXSLTPAIREEEK
13: BTLIKXSLTPAIREEE
14: BTLEIKXSLTPAIREE
15: BTELEIKXSLTPAIRE
16: BETELEIKXSLTPAIR
```

DES (Data Encryption Standard)

- Entwickelt Anfang der 70er Jahre bei IBM („Lucifer“, „Feistelchiffre“)
- 1977 wurde er von der US-Standardisierungsbehörde NIST (National Institute of Standards and Technology) als Standard anerkannt
- NSA an der Entwicklung beteiligt und die Designkriterien unter Verschluss gehalten sowie Schlüssellänge verkürzt
- DES gilt als Muster aller modernen Chiffren
- DES ist eine Blockchiffre (64 Bit Blöcke)
- Schlüssellänge 56 Bit (+8 Bit Prüfsumme)
- Entwickelt für Hardwareverschlüsselung

Der Data Encryption Standard (DES) ist eine Blockchiffre mit 8 Byte Blocklänge und weit verbreiteter symmetrischer Verschlüsselungsalgorithmus und wurde als offizieller Standard für die US-Regierung im Jahr 1977 bestätigt und wird seither international vielfach eingesetzt. Seine Entstehungsgeschichte hat wegen der Beteiligung der NSA am Design des Algorithmus immer wieder Anlass zu Spekulationen über seine Sicherheit gegeben. Heute wird DES aufgrund der verwendeten Schlüssellänge von nur 56 Bits für viele Anwendungen als nicht ausreichend sicher erachtet.

Ursprünglich hieß der bei IBM unter der Leitung von Horst Feistel entwickelte Algorithmus Lucifer und bot eine Schlüssellänge von 16 Byte / 128 Bit. Die NSA soll dafür verantwortlich gewesen sein, dass die Schlüssellänge für DES auf 56 Bit gekürzt wurde, weil dies die Schlüssellänge ist, die man mit den Supercomputern bei der NSA in den 1970er-Jahren gerade noch so per Brute Force hätte knacken können.

Die Schlüssellänge kann durch Mehrfachanwendung des DES jedoch auf einfache Weise vergrößert werden. Als Triple-DES, auch als TDES, 3DES oder DESede bezeichnet, wird der DES weiterhin am häufigsten, zum Beispiel von Banken in Chipkartenanwendungen, eingesetzt, obwohl der TDES als offizieller Standard für die USA durch den Advanced Encryption Standard (AES) abgelöst wurde.

Weil die Schlüssellänge nur 56 Bit beträgt, konnte DES bereits durch Brute-Force-Angriffe gebrochen werden, indem systematisch alle möglichen Schlüssel (256 = ca. 72 Billiarden) getestet wurden. Die EFF baute 1998 eine etwa 250.000 Dollar teure Maschine mit dem Namen Deep Crack. Dieser Superrechner enthielt 1536 spezielle Krypto-Chips und konnte pro Sekunde etwa 88 Milliarden Schlüssel testen. Im Juli 1998 gelang es mit dieser Maschine, einen DES-Code in 56 Stunden zu knacken.

Die einzige andere öffentlich bekannte Maschine zum Brechen von DES ist COPACOBANA. Sie wurde 2006 an den Universitäten Bochum und Kiel gebaut. Im Gegensatz zu Deep Crack besteht eine COPACOBANA aus rekonfigurierbaren Hardware-Bausteinen, sog. FPGAs. COPACOBANA kann 65 Milliarden DES-Schlüssel pro Sekunde testen, woraus sich eine durchschnittliche Suchzeit von 6,4 Tagen für eine DES-Attacke ergibt. Durch den Einsatz rekonfigurierbarer Hardware kann COPACOBANA auch zum Brechen anderer Chiffren wie A5 eingesetzt werden. Die Material- und Herstellungskosten von COPACOBANA belaufen sich auf „nur“ etwa 10.000 Dollar.

Verfahren

Bei DES handelt es sich um einen symmetrischen Algorithmus, das heißt zur Ver- und Entschlüsselung

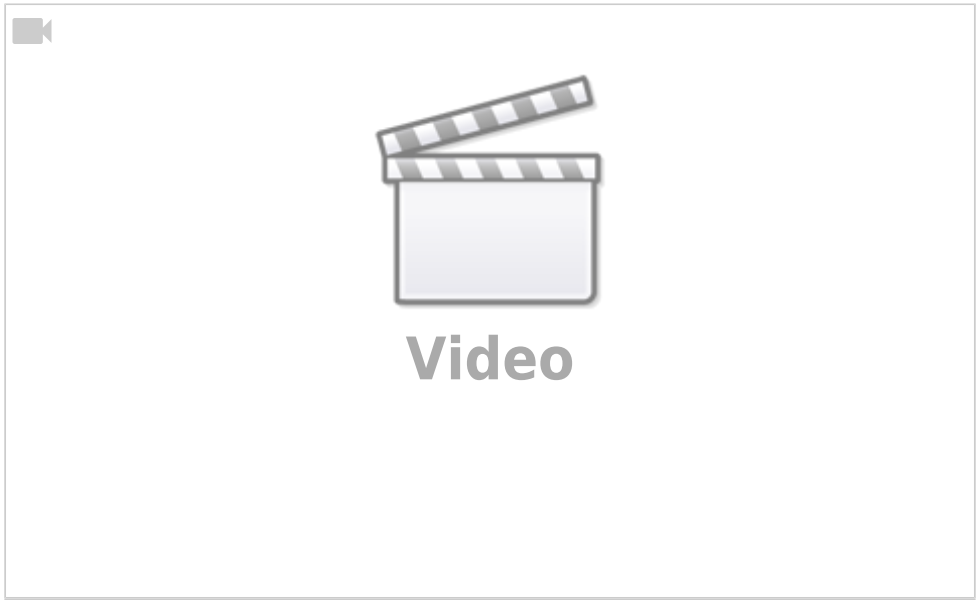
wird derselbe Schlüssel verwendet. DES funktioniert als Blockchiffre, jeder Block wird also unter Verwendung des Schlüssels einzeln chiffriert, wobei die Daten in 16 Runden von Substitutionen und Transpositionen (Permutation) nach dem Schema von Feistel verwürfelt werden.

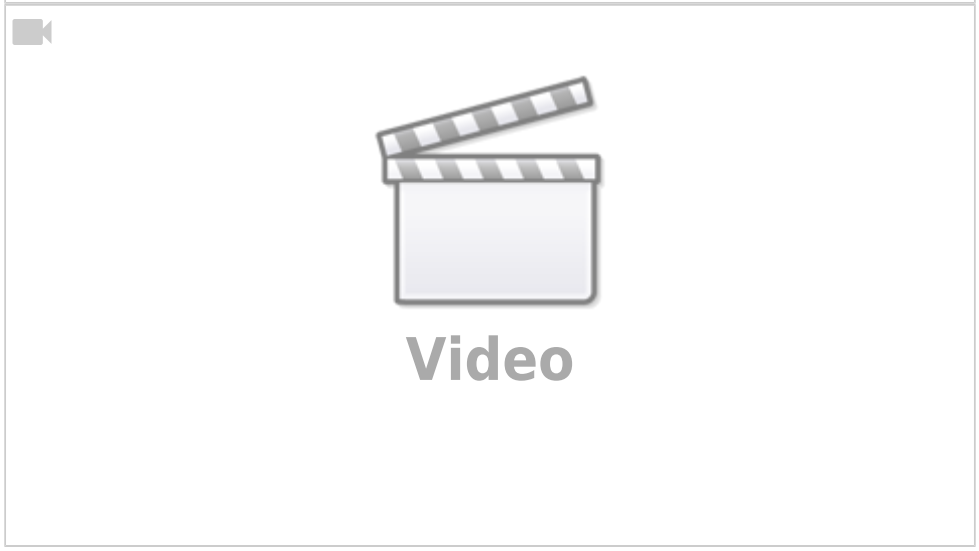
Die Blockgröße beträgt 64 Bits, das heißt ein 64-Bit-Block Klartext wird in einen 64-Bit-Block Chiffretext transformiert. Auch der Schlüssel, der diese Transformation kontrolliert, besitzt 64 Bits. Jedoch stehen dem Benutzer von diesen 64 Bits nur 56 Bits zur Verfügung; die übrigen 8 Bits (jeweils ein Bit aus jedem Byte) werden zum Paritäts-Check benötigt. Die effektive Schlüssellänge beträgt daher nur 56 Bits. Die Entschlüsselung wird mit dem gleichen Algorithmus durchgeführt, wobei die einzelnen Rundenschlüssel in umgekehrter Reihenfolge verwendet werden.

Auf den 64 Bit Block wird eine initiale Permutation angewandt. Danach wird der Block in zwei Teile aufgeteilt und jeder Teil in ein 32 Bit Register gespeichert, auf die das Prinzip eines Feistel-Netzwerkes angewandt wird.

Der DES-Algorithmus beschreibt zunächst nur, wie ein Datenblock mit 64 Bits verarbeitet wird. Zur Verarbeitung einer Nachricht beliebiger Länge lässt sich der DES wie auch jede andere Blockchiffre in verschiedenen Betriebsmodi verwenden. Für bestimmte Betriebsmodi, wie zum Beispiel ECB oder CBC, ist ein Auffüllen des Klartextes auf ein Vielfaches der vollen Blocklänge notwendig (Padding). Dies geschieht indem die Bitfolge 1000… angehängt wird.

Die genaue Spezifikation findet sich als FIPS 46-3 beim NIST.





AES (Advanced Encryption Standard) / Rijndael-Chiffre

AES steht für 'Advanced Encryption Standard', ist eine Blockchiffre und der Sieger-Algorithmus einer Ausschreibung in 2000 des NIST und gilt als Nachfolger von DES (Data Encryption Standard) von 1977. Der Algorithmus wurde von Joan Daemen und Vincent Rijmen entwickelt und die Chiffre wird deshalb auch Rijndael-Chiffre genannt. AES lässt einem die Wahl bei der Schlüssellänge von 128, 192 und 256 Bit. AES-192 und AES-256 sind in den USA für staatliche Dokumente mit höchster Geheimhaltungsstufe zugelassen.

AES benutzt eine Blocklänge von 16 Bytes, das heißt, dass ein Chiffrat 15 Zeichen länger werden kann als der ursprüngliche Klartext. Es empfiehlt sich, als Schlüssel den Hash eines Klartextpasswortes zzgl. eines (wenn gewünscht gehashten) Salts zu benutzen, z. B. SHA-256 für die 256-bit-Variante von AES. Dies ergibt eine gute Sicherheit..

AES hat sich mittlerweile als Standard durchgesetzt und neuere CPUs enthalten inzwischen spezielle Instruktionen, um die Verschlüsselung damit zu beschleunigen. Er wird u. a. bei der WLAN-Verschlüsselung WPA2, bei SSH, IPSec und in der IP-Telefonie benutzt.

Während Rijndael der Sieger-Algorithmus und zukünftiger Namensträger von AES wurde, waren die folgende vier weiteren Kandidaten in der engere Auswahl für AES gezogen worden, haben es letztendlich aber nicht geschafft: MARS, RC6, Serpent und Twofish. Weitere Kandidaten, die es nicht in die Endrunde schafften sind: CAST-256, CRYPTON, DEAL, DFC, E2, FROG, HPC, LOKI97, MAGENTA und SAFER+.

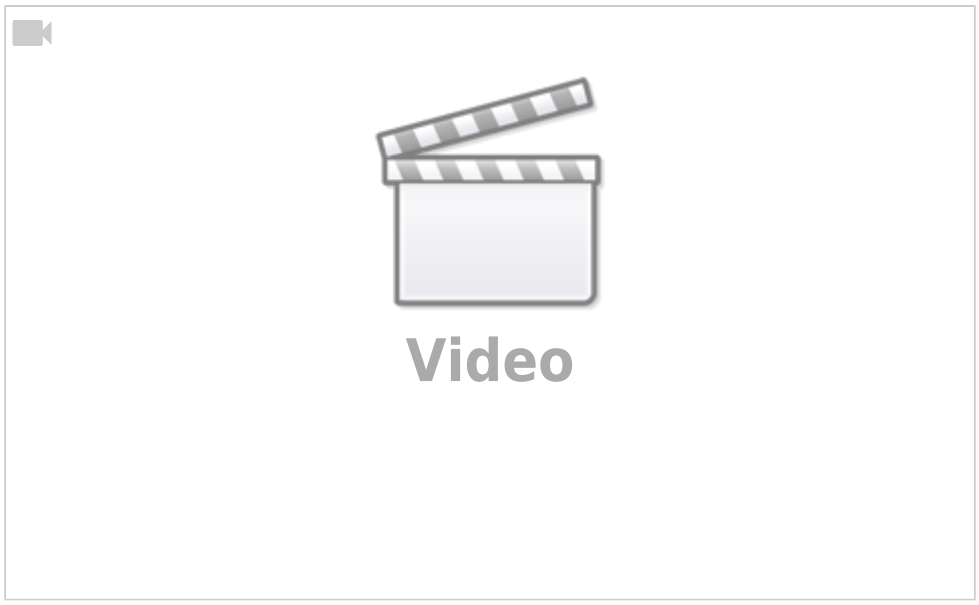
Beschreibung

Der in AES implementierte Algorithmus heißt Rijndael und ist ein als Substitutions-Permutations-Netzwerk entworfene Blockchiffre. Jeder Block wird zunächst in eine zweidimensionale Tabelle mit vier Zeilen geschrieben, deren Zellen ein Byte groß sind. Die Anzahl der Spalten variiert je nach Blockgröße von 4 (128 Bits) bis 8 (256 Bits). Jeder Block wird nun nacheinander bestimmten Transformationen unterzogen. Aber anstatt jeden Block einmal mit dem Schlüssel zu verschlüsseln, wendet Rijndael verschiedene Teile des erweiterten Originalschlüssels nacheinander auf den Klartext-Block an. Die Anzahl der Runden variiert und ist von Schlüssellänge und Blockgröße abhängig (bei AES also nur von der Schlüssellänge).

Eine S-Box (Substitutionsbox) mit 256 Bytes dient als Basis für eine monoalphabetische Verschlüsselung. Sie gibt an, wie in jeder Runde jedes Byte eines Blocks durch einen anderen Wert zu ersetzen ist. Typischerweise wird die S-Box in Blockchiffren eingesetzt, um die Beziehung zwischen Klar- und Geheimtext zu verwischen (in der kryptologischen Fachsprache Konfusion genannt). Die S-Box des AES setzt auch teilweise das Shannon’sche Prinzip der Diffusion um. Die Konstruktion der S-Box unterliegt Designkriterien, die die Anfälligkeit für die Methoden der linearen und der differentiellen Kryptoanalyse sowie für algebraische Attacken minimieren sollen.

Die genaue Spezifikation findet sich beim NIST.

Hier der Link zum Ausprobieren: Rijndael Animation



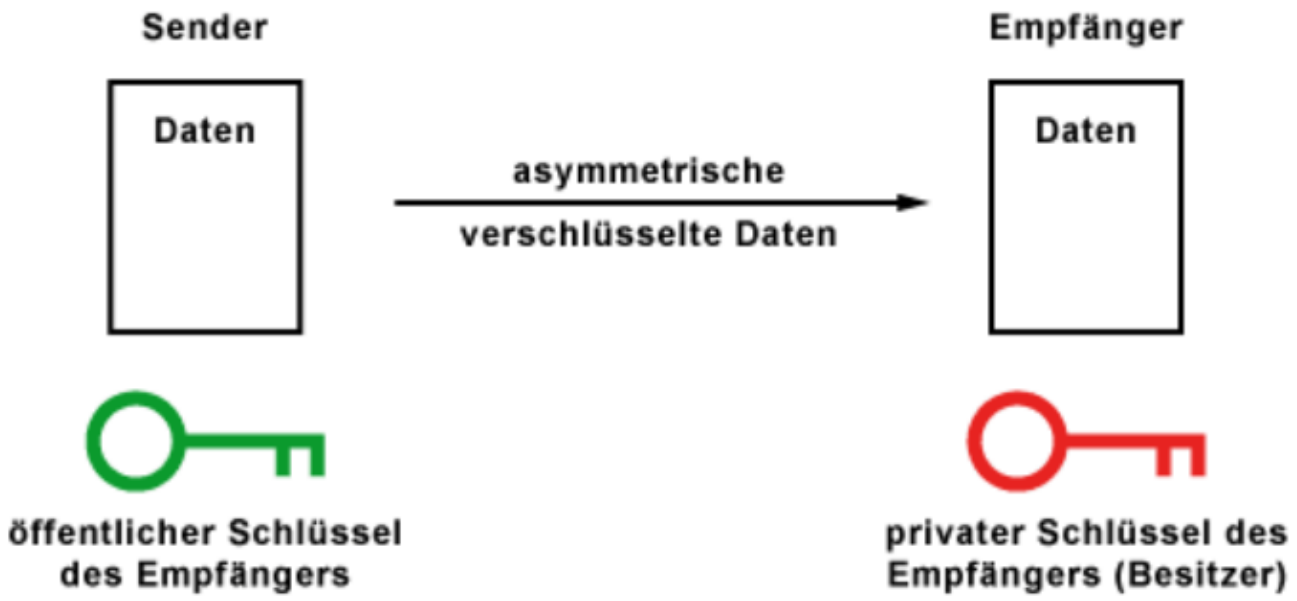
Asymmetrische Kryptografie (Verschlüsselung)

In der asymmetrischen Kryptografie arbeitet man nicht mit einem einzigen Schlüssel, sondern mit einem **Schlüsselpaar**. Bestehend aus einem **öffentlichen** und einem privaten Schlüssel. Man bezeichnet diese Verfahren als asymmetrische Verfahren oder **Public-Key-Verfahren**. Üblich sind auch die Bezeichnungen Public-Key-Kryptografie und Public-Key-Verschlüsselung.

Ein fundamentales Problem der Kryptografie ist, dass sich die Kommunikationspartner auf einen gemeinsamen Schlüssel verständigen müssen. Man bezeichnet das als **Schlüsselaustauschproblem**.

Während ein manueller Schlüsselaustausch durch ein persönliches Treffen oder per Telefon bei einer handvoll Kommunikationspartner sicherlich kein Problem wäre. Wird es bei vielen Schlüsseln oder vielen Kommunikationspartnern schnell unübersichtlich und aufwendig. Hier kommt das Thema **Schlüsselverwaltung und -verteilung** zum Tragen. Alternativ bestünde die Möglichkeit einen Authentifizierungsserver einzusetzen. Beispielsweise Kerberos. Alternativ bietet sich die asymmetrische Kryptografie an.

Prinzip der asymmetrischen Kryptografie (Public-Key-Verfahren)

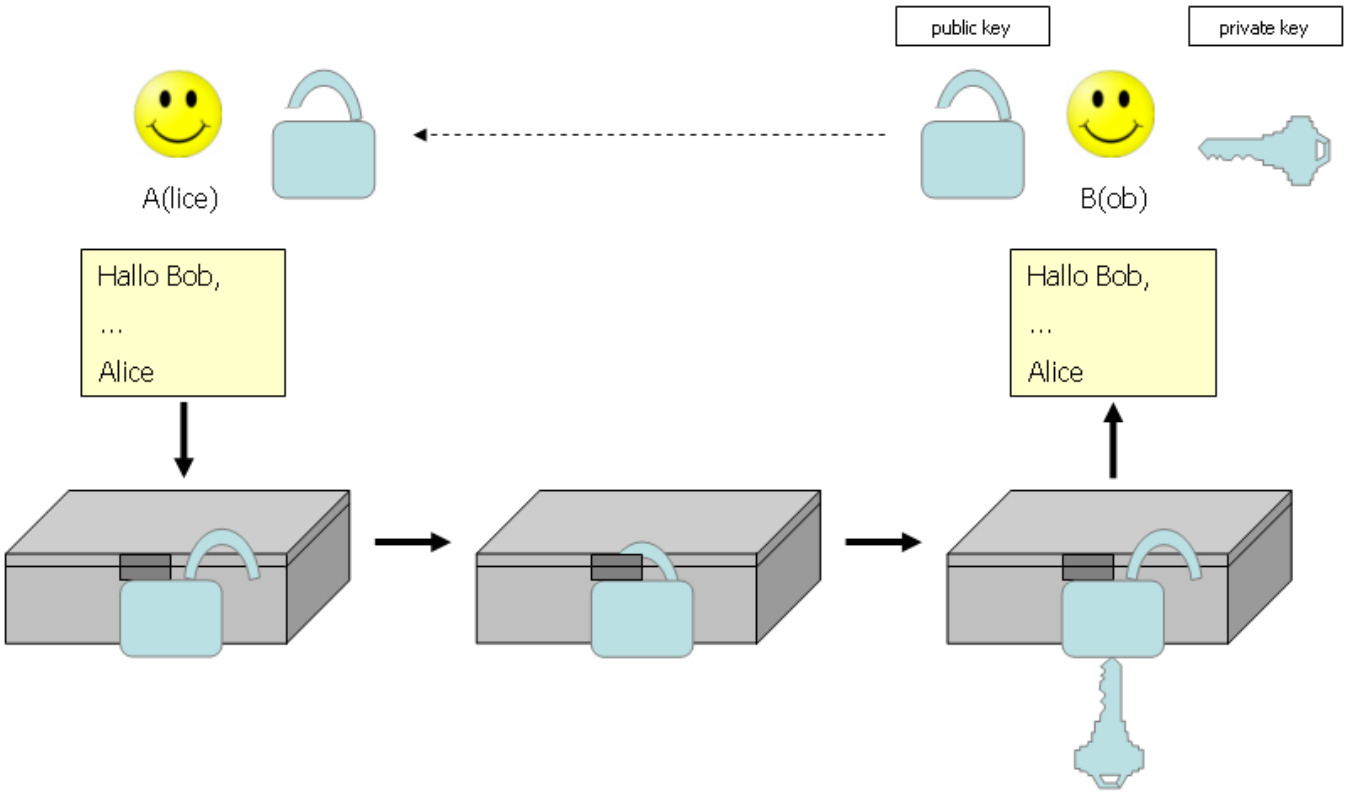


Asymmetrische Verschlüsselungsverfahren arbeiten mit Schlüsselpaaren. Ein Schlüssel ist der **öffentliche Schlüssel (Public Key)**, der andere ist der **private Schlüssel (Private Key)**. Dieses Schlüsselpaar hängt über einen **mathematischen Algorithmus** eng zusammen. **Daten**, die mit dem **öffentlichen Schlüssel verschlüsselt** werden, können nur mit dem **privaten Schlüssel entschlüsselt** werden. Deshalb muss der private Schlüssel vom Besitzer des Schlüsselpaares geheim gehalten werden.

Der konkrete Anwendungsfall sieht so aus: Will der Sender Daten verschlüsselt an den Empfänger senden, benötigt er den öffentlichen Schlüssel des Empfängers. Mit dem öffentlichen Schlüssel können die Daten verschlüsselt, aber nicht mehr entschlüsselt werden (Einwegfunktion). Nur noch der Besitzer des privaten Schlüssels, also der richtige Empfänger kann die Daten entschlüsseln. Wichtig bei diesem Verfahren ist, dass der private Schlüssel vom Schlüsselbesitzer absolut geheim gehalten wird. Kommt eine fremde Person an den privaten Schlüssel muss sich der Schlüsselbesitzer ein neues Schlüsselpaar besorgen.

Das Problem bei der asymmetrischen Kryptografie ist die Verteilung der öffentlichen Schlüssel. Typischerweise erfolgt die Übergabe des öffentlichen Schlüssels beim Erstkontakt. Doch hierbei stellt sich die Frage, ob dieser Schlüssel tatsächlich der echte Schlüssel des Kommunikationspartner ist.

Hinweis: Asymmetrische Verfahren benötigen viel mehr Rechenleistung als symmetrische Verfahren. Wenn man RSA und AES miteinander vergleicht, dann ist RSA ungefähr um den Faktor 1.000 langsamer als AES.



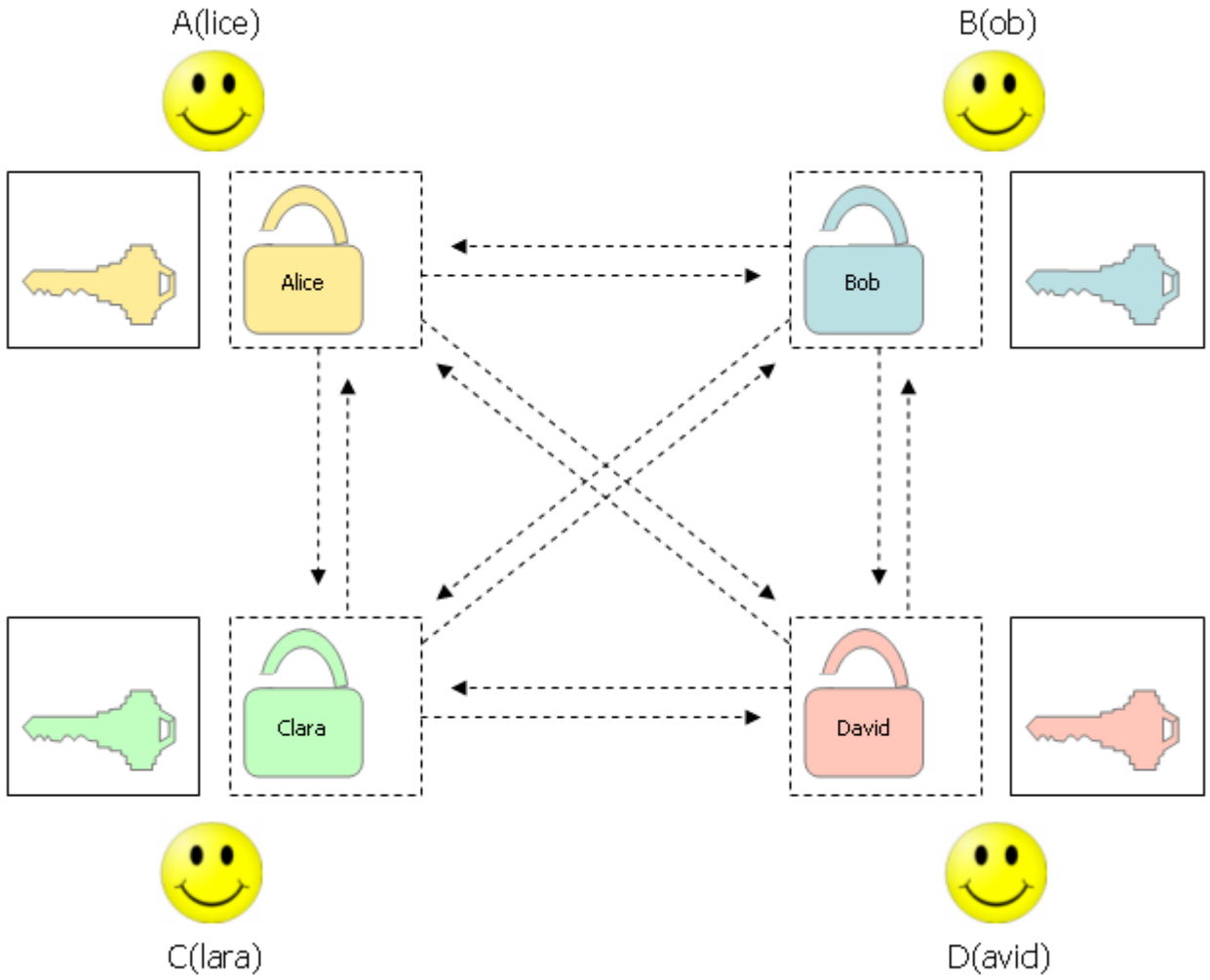
Vereinfachtes Beispiel

Ich verteile Sparschweine, zu deren Schloss nur ich den Schlüssel habe. Wenn mir jemand die Nachricht „135“ senden möchte, nimmt er eines dieser Sparschweine, steckt einen Zettel mit „135“ hinein und schickt es mir per Post. Ich öffne mit meinem Schlüssel das Sparschwein und kann den Zettel lesen.

Ein Schlüsselpaar für jede Kommunikationsteilnehmer/in

Wenn mehrere Personen verschlüsselte Nachrichten austauschen wollen, dann reicht es, wenn jede

Person ein Schlüsselpaar aus öffentlichem und privatem Schlüssel besitzt.



Bei 4 Personen A(lice), B(ob), C(lara) und D(avid), die alle miteinander kommunizieren wollen, werden somit 4 Schlüsselpaare benötigt. Bei n Personen benötigt man entsprechend genau n Schlüsselpaare. Es werden also viel weniger Schlüssel benötigt als bei symmetrischen Chiffriersystemen.

Einwegfunktion und Falltürfunktion

Damit so ein Verfahren funktioniert, muss Folgendes gelten:

- Mithilfe der öffentlich verfügbaren Informationen (insbesondere dem öffentlichen Schlüssel) muss es in sinnvoller Zeit möglich sein, den Klartext zum Geheimtext zu verschlüsseln. Außerdem muss es mit dem privaten Schlüssel in sinnvoller Zeit möglich sein, den Geheimtext zu entschlüsseln. Ansonsten könnte man das Verfahren nie anwenden.
- Mithilfe der öffentlich verfügbaren Informationen darf es – ohne den privaten Schlüssel – nicht in sinnvoller Zeit möglich sein, den Geheimtext zu knacken, also den Klartext zu rekonstruieren. Daraus folgt, dass auch der private Schlüssel nicht mithilfe der öffentlichen Informationen in sinnvoller Zeit berechnet werden kann.

Asymmetrische Chiffriersysteme nutzen Funktionen mit den oben dargestellten Eigenschaften – sogenannte Einwegfunktionen. In eine Richtung lassen sie sich in sinnvoller Zeit berechnen, aber die Umkehrung ist – ohne den privaten Schlüssel als Hilfsmittel – kaum möglich.

Auf den ersten Blick klingt es verwunderlich, dass hier immer von „in sinnvoller Zeit“ die Rede ist. Das liegt daran, dass man ja z.B. alle möglichen privaten Schlüssel durchprobieren könnte. Es ist also nicht möglich, das Knacken ganz auszuschließen. Aber, wenn es mit aller Rechenleistung der Welt viele Hundert Millionen Jahre dauern würde, gilt das als ausreichend sicher.

Bei der asymmetrischen Verschlüsselung geht es darum, eine **Funktion** zu wählen, die **sehr einfach zu rechnen ist, aber deren Umkehrung dagegen sehr aufwendig**. Realisiert wird das mit **Modulo-Rechenarten**.

Einige davon sind tatsächlich sehr einfach zu rechnen, während die Umkehrung sehr aufwendig ist. Sie entsprechen also einer **Einwegfunktion**.

Es gibt allerdings auch Funktionen, bei denen sich mit einer **zusätzlichen Information die Umkehrung abkürzen** lässt. In so einem Fall spricht man von einer **Falltürfunktion**.

Mathematische Grundlagen

Modulo

Bei der Modulo Operation wird der Rest einer Division berechnet. Der Rückschluss vom Ergebnis (Rest) auf die Ausgangszahl ist nicht möglich.

Beispiel:

```
17 % 5 = 2
6 % 2 = 0
18 % 6 = 0
```

Potenzieren

Potenzieren im Bereich natürlicher, ganzer, reeller oder sogar komplexer Zahlen ist eine grundlegende Operation der Mathematik:

$g^e = y$

Beispiel:

```
10^2 = 100
```

Logarithmieren

Die Umkehrfunktion zum Potenzieren in Bezug auf das Auffinden der Exponenten ist das Logarithmieren.

$\log_g y = e$

Beispiel: $\log_{10} 100 = 2$

Hierbei ist es nicht unüblich, dass dieser Exponent außerhalb des Definitionsbereichs liegt (z.B. $\log_{10} 8 \approx 0, 9$).

Nichtsdestotrotz lässt sich dieser Exponent mit Hilfe von Computern in sehr schneller Zeit (näherungsweise) errechnen. Beispielhaft wäre da die Berechnung über die Potenzreihenentwicklung.

Modulo Addition/Subtraktion

Ist bei der Addition das Ergebnis größer oder gleich n, dann zieht man n davon ab. Ist analog dazu, das Ergebnis bei der Subtraktion kleiner null, dann zählt man n dazu.

Beispiel:

```
(3+5) % 7 = 1
(2+2) % 13 = 4
(3-6) % 9 = 6
(3+6) % 9 = 0
```

Modulo Multiplikation

Beispiel:

```
(4*5) % 7 = 20 -7 -7 = 6
```

Modulo Division

Fragestellung: gibt es zu jeder Zahl a eine Zahl b mit der Eigenschaft a*b=a (mod n)

Gibt es eine solche Zahl, dann nennt man sie das zu a inverse Element und schreibt dafür a-1.

Es gibt auch eine Modulo-Division, weil $b*a^{-1}$ (mod n) gleich ist wie b/a (mod n).

Potenzieren mit Modulo

Beispiel:

```
3^4 % 7 = 3*3*3*3 % 7 = 4
```

In der Kryptografie spielt vor allem die Umkehrung dieser Rechenart eine Rolle!

Diskreter Logarithmus (Logarithmieren mit Modulo)

Das Problem des diskreten Logarithmus beschreibt ein mathematisches Problem. Eine der beiden

Umkehrungen der Potenzierung mit Modulo ist der Modulo-Logarithmus, der auch als diskreter Logarithmus bezeichnet wird. Sind g, x und p gegeben, dann ist der diskrete Logarithmus die Zahl x, für die gilt:

$g^x \pmod p = S$

g ... Generator p ... Primzahl

$\{g,p,S\} \in \mathbb{N}$

$\{x\} \in \mathbb{N}_0$

x ⇒ geheimer Schlüssel
S ⇒ öffentlicher Schlüssel

Ein Angreifer, der diese Gleichung nach dem geheimen Schlüssel nach x auflösen könnte, wäre in der Lage die geheime Nachricht zu entschlüsseln.

Bei kleinen Zahlen kann man durch systematisches Ausprobieren dieses Problem lösen.

Beispiel

g=8, p=29 und S=21 ⇒ $8^x \bmod 29 = 21$

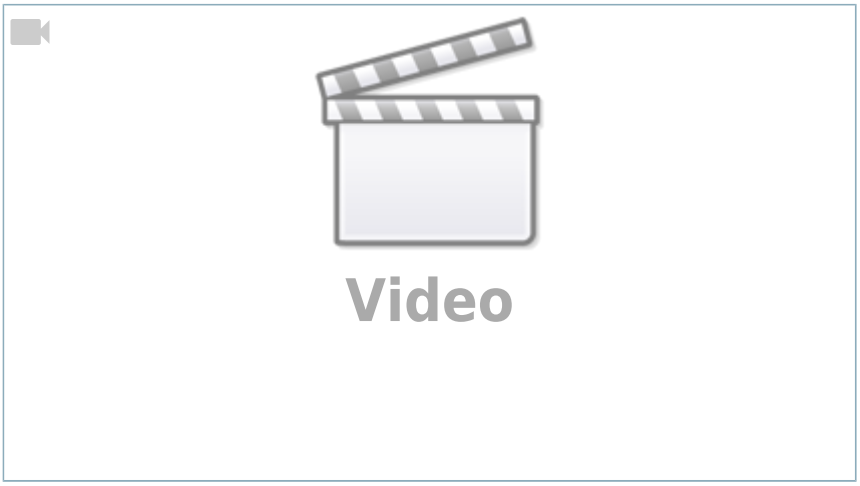
n	$8^n \bmod 29$
0	1
1	8
2	6
3	19
4	7
5	27
6	13
...	...
15	21
...	...
28	1
29	8

Der diskrete Logarithmus von b zur Basis x, kurz $\log_x b$, ist die kleinste natürliche Zahl g:

$g=\log_x b \pmod n$

Die Frage, wann zu gegebenen Zahlen g, S und n der diskreter Logarithmus existiert, ist kein triviales Problem, in der Kryptografie hat man jedoch nur mit solchen Problemstellungen zu tun, in denen der diskrete Logarithmus existiert.

Sind die Zahlen groß genug, so schaffen es auch nicht die leistungsfähigsten Computer durch systematisches Ausprobieren zu einem Ergebnis zu kommen. Bis jetzt ist kein Algorithmus bekannt, der dieses Problem für große Zahlen lösen kann.



Der **diskrete Logarithmus** fällt hier als **Einwegfunktion** besonders auf, weil man diesen sehr leicht berechnen kann. Umgekehrt ist es schlichtweg nicht möglich eine große Zahl in praktikabler Zeit zurückzurechnen. Man bezeichnet das als Diskreter-Logarithmus-Problem. Viele asymmetrische Verfahren basieren darauf. Allerdings bedeutet das nicht, dass nicht doch irgendwann ein Weg gefunden wird, den diskreten Logarithmus zu lösen.

Faktorisierungsproblem

Eine weitere Einwegfunktion ist das Multiplizieren von Primzahlen. Während die Multiplikation für einen Computer kein Problem darstellt, ist der umgekehrte Weg, beim dem das Primzahlprodukt in seine Faktoren zerlegt werden soll, nicht in akzeptabler Zeit machbar. Man spricht von Faktorisierung und in dem Zusammenhang vom Faktorisierungsproblem.

Beispiel:

n = p * q

p=17
q=19

n = p*q = 17*19 = 323

Wenn man 17 x 19 berechnet (beides Primzahlen), dann kommt 323 heraus. Und jetzt soll man die beiden unbekannten Faktoren (17 und 19) daraus zurückberechnen. Es gibt im Prinzip nur einen Weg. Man muss alle Möglichkeiten durchprobieren. Bei hinreichend großen Primzahlen dauert das ewig. Damit ist das Faktorisierungsproblem gemeint.

n	q	n/q=p
323	2	161,5
323	3	107,6666..
323	5	64,6
323	7	46,142...

323	9	35,88..
323	11	29,363...
323	13	24,846...
323	**17**	**19**

Alle gängigen asymmetrische Verfahren basieren auf komplexen mathematischen Berechnungen, die gemeinsam haben, dass es für sie noch keine Vereinfachung gibt. Schlüssel, Klartext und Geheimtext stellen große Zahlen bzw. Zahlenpaare dar. Die Verfahren sind aber nur so lange sicher sind, bis jemand eine Vereinfachung gefunden hat.

Weil es nur begrenzt geeignete mathematische Berechnungen mit Einwegfunktion gibt, lassen sich nicht beliebig viele asymmetrische Verfahren entwickeln.

Diffie-Hellman-Schlüsseltausch

Zurück zum Schlüsseltauschproblem: Alice und Bob können Einweg- und Falltürfunktionen zur Lösung des Schlüsseltauschproblems verwenden.

Ein Verfahren, das den diskreten Logarithmus zur Lösung des Schlüsseltauschproblems verwendet, wurde von den Kryptografen Whitfield Diffie und Martin Hellman erfunden.

Der Algorithmus mit Farben

Die Mathematik hinter dem Algorithmus ist leider nicht ganz einfach zu verstehen. Dafür gibt es aber eine schöne Analogie mit Farben, die man zum Verständnis nutzen kann.

Der Ablauf:

1. Alice und Bob einigen sich auf eine gemeinsame (öffentliche) Farbe.
2. Jeder wählt sich zudem eine geheime weitere Farbe.
3. Alice und Bob mischen sich aus ihrer geheimen und der öffentlichen Farbe eine weitere Farbe.
4. Sie schicken jeweils die neu gemischte Farbe zu ihrem Kommunikationspartner.
5. Am Ende mischt jeder seine geheime Farbe in die zuvor ausgetauschten Mischfarbe.

Die gemeinsame Farbe (der gemeinsame Cocktail) am Ende des Prozesses besteht also aus drei gemischten Farben. Und Eve in der Mitte? Sie hat nur die öffentliche Farbe und zwei gemischte Farben. Und wie du vielleicht aus dem Kunstunterricht weißt, ist es ziemlich schwierig, aus einer Mischfarbe zu erkennen, welche Ausgangsfarben genau darin stecken.

Exkurs Diffie-Hellmann von INF-Schule

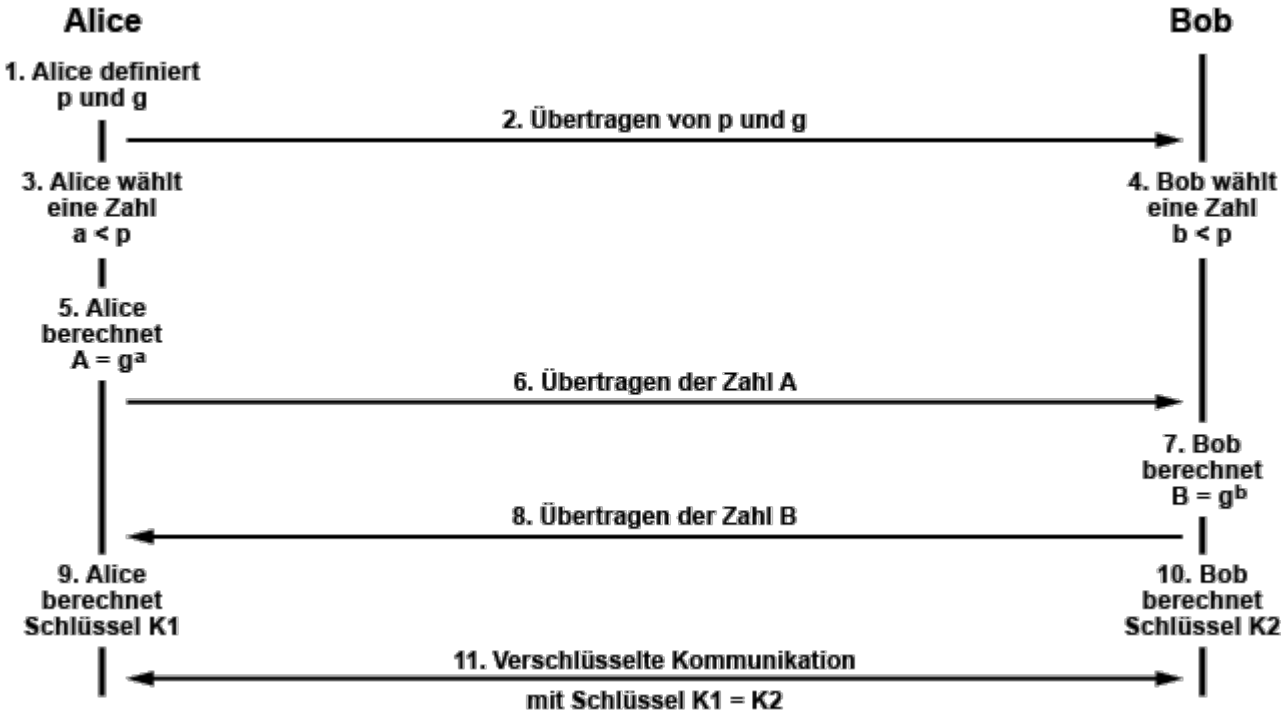
Diffie-Hellmann-Schlüsseltausch Algorithmus

Der Algorithmus mit Zahlen

In der folgenden Beschreibung ist von Alice und Bob die Rede. Beide stehen beispielhaft für zwei Kommunikationspartner, die ihre Kommunikation verschlüsseln wollen und den dazu notwendigen

geheimen Sitzungsschlüssel zum Ver- und Entschlüsseln vorab austauschen müssen. Um den Sitzungsschlüssel vor einem Angreifer zu schützen, der eventuell die Kommunikation abhört oder aufzeichnet, in der Hoffnung den Sitzungsschlüssel abgreifen zu können, vereinbaren sie den Schlüsselaustausch nach Diffie-Hellman-Merkle.

Das mathematische Verfahren beruht auf dem sog. „modularen Potenzieren“. Auch wenn der mathematische Hintergrund nicht so einfach zu verstehen ist, kannst du das Verfahren aber sicherlich einmal hier nachvollziehen. Die wesentlichen Schritte sind (ganz analog zu den Farben oben) die Folgenden.



1. Alice definiert p und g

Zuerst müssen sich Alice und Bob auf eine große Primzahl p und eine natürliche Zahl g, die ein Generator aus Gruppe Z(p) sein sollte, einigen. Die Zahl g kann aber auch einen Wert kleiner p annehmen. Weil Alice die Kommunikation zu Bob aufbaut, legt typischerweise Alice die Zahlen p und g fest. Diese Vorgehensweise kann in der Praxis auch anders erfolgen. Für dieses Beispiel wählt Alice **p = 11** und **g = 7**.

2. Alice schickt p = 11 und g = 7 zu Bob

Beide Werte dürfen bekannt sein und können deshalb über einen unsicheren Kanal übertragen werden.

3. Alice wählt eine Zahl a < p

Alice erzeugt nun zusätzlich eine Zufallszahl a, die kleiner als die gewählte Primzahl p sein muss (1 … p - 1). Für dieses Beispiel wählen wir a = 3.

4. Bob wählt eine Zahl b < p

Bob erzeugt nun zusätzlich eine Zufallszahl b, die kleiner als die gewählte Primzahl p sein muss (1 ... p - 1). Für dieses Beispiel wählen wir b = 6.

5. Alice berechnet A

$$A = g^a \bmod p$$
$$A = 7^3 \bmod 11 = 2$$

6. Alice schickt die Zahl A = 2 zu Bob

Übertragung der Zahl A = 2 über den unsicheren Kanal zu Bob.

7. Bob berechnet B

$$B = g^a \bmod p$$
$$B = 7^6 \bmod 11 = 4$$

8. Bob schickt die Zahl B = 4 zu Bob

Übertragung der Zahl B = 4 über den unsicheren Kanal zu Alice.

9. Alice berechnet Schlüssel K1

$$K1 = B^a \bmod p$$
$$K1 = 4^3 \bmod 11 = 9$$

10. Bob berechnet Schlüssel K2

$$K2 = A^b \bmod p$$
$$K2 = 2^6 \bmod 11 = 9$$

11. Verschlüsselte Kommunikation mit Schlüssel K1 = K2

Beide kommen auf das gleiche Ergebnis und haben so einen gemeinsamen geheimen Schlüssel. Dieser Schlüssel kann zum Beispiel in einem symmetrischen Verfahren als temporärer Sitzungsschlüssel genutzt werden.

$$K1 = K2$$

Der Angreifer kennt nur p und g. Außerdem A und B, da beide Werte übertragen werden. Allerdings kann er den Schlüssel K nur berechnen, wenn er a und b hat. Da diese Werte nicht übertragen werden, muss der Angreifer sich den Schlüssel anders berechnen, was bei einer ausreichend großen Primzahl fast unmöglich ist. Dieses Verfahren beruht darauf, dass es mit wenig Rechenleistung möglich ist, die Potenz g^x mod p zu errechnen. Aber der umgekehrte Weg von g^x auf x zu schließen ist sehr schwierig (diskreter Logarithmus).

RSA - Verfahren

RSA ist das ein asymmetrisches kryptografisches Verfahren bzw. ein Public-Key-Verfahren von den Kryptografen Ron Rivest, Adi Shamir und Leonard Adleman aus dem Jahr 1977. Kein anderes asymmetrisches Verfahren ist so vielseitig einsetzbar, so gut erforscht und so einfach zu implementieren, wie RSA. An RSA kommt man im Zusammenhang mit asymmetrischen Verfahren einfach nicht vorbei. Im Vergleich zum Diffie-Hellman-Schlüsselaustausch eignet sich RSA auch für die Verschlüsselung und als Signaturverfahren. Der RSA-Algorithmus basiert auf dem Faktorisierungsproblem.

Das RSA-Verfahren ist ein modernes asymmetrisches kryptografisches Verfahren, das zum Verschlüsseln und zum digitalen Signieren verwendet wird. Um es zu durchschauen, musst du dich mit zahlentheoretischen Überlegungen und Zusammenhängen auseinandersetzen.

RSA-Verfahren Step-by-Step

Hybride Chiffriersysteme

Fazit von symmetrische Verschlüsselungsverfahren

Die Algorithmen, die bei modernen symmetrischen Chiffriersystemen benutzt werden, sind sehr effizient. Man kann mit symmetrischen Chiffriersystemen daher recht schnell einen gegebenen Text verschlüsseln.

Schwierig ist bei symmetrischen Chiffriersystemen in der Regel der Schlüsselaustausch zwischen Sender/in und Empfänger/in. Da beide denselben Schlüssel benutzen, muss er sicher zwischen den Kommunikationspartner/innen ausgetauscht werden.

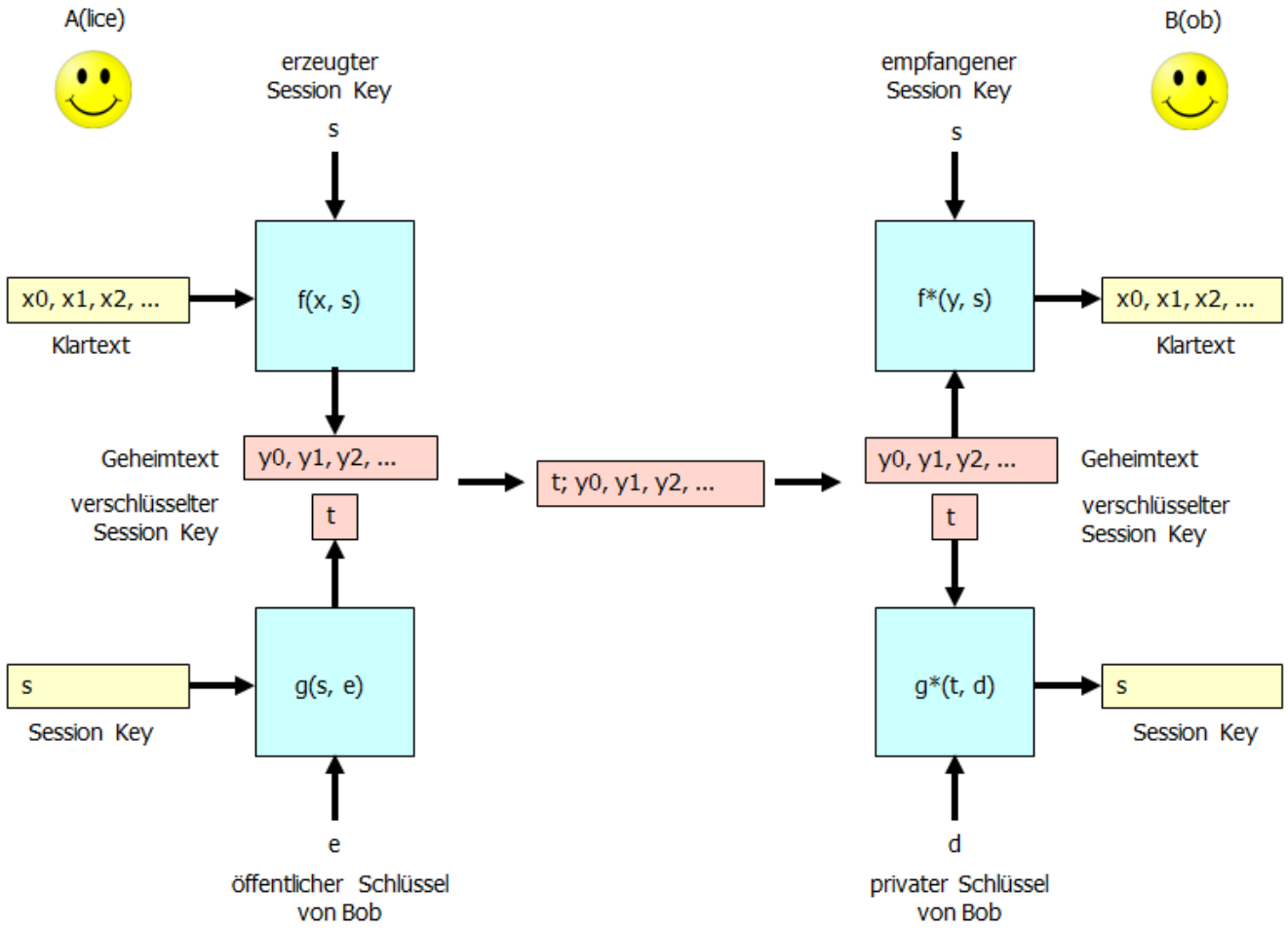
Fazit von asymmetrische Verschlüsselungsverfahren

Die Algorithmen, die bei modernen asymmetrischen Chiffriersystemen benutzt werden, sind recht aufwendig. Bei langen Texten dauert es daher eine Weile, bis der gesamte Text verschlüsselt ist.

Da bei asymmetrischen Chiffriersystemen ein Schlüsselpaar erzeugt wird und einer der beiden Schlüssel veröffentlicht wird, ist allerdings kein sicherer Schlüsselaustausch zwischen Sender:in und Empfänger:in erforderlich.

Kombination aus symmetrischen und asymmetrischen Verschlüsselungsverfahren

Die Stärken von symmetrischen und asymmetrischen Chiffriersystemen lassen sich nutzen, wenn man beide Systeme geeignet kombiniert. Die folgende Abbildung zeigt die Struktur eines kombinierten (man sagt auch hybriden) Chiffriersystems:



Wenn Alice Bob eine Nachricht senden möchte, erzeugt Alice in einem ersten Schritt einen sogenannten Session Key.

Dieser Session Key wird als Schlüssel zum Verschlüsseln der Nachricht benutzt. Dabei kommt ein schnelles symmetrisches Verfahren zum Einsatz.

Als nächstes wird der Session Key selbst verschlüsselt. Alice benutzt hierzu den öffentlichen Schlüssel von Bob bei einem asymmetrischen Chiffrierverfahren.

Die verschlüsselte Nachricht und der verschlüsselte Session Key werden jetzt an Bob geschickt.

Bob benutzt seinen privaten Schlüssel, um den Session Key zu rekonstruieren.

Anschließend kann Bob den Session Key - zusammen mit dem von Alice benutzten symmetrische Chiffrierverfahren - benutzen, um den Geheimtext zu entschlüsseln.

Digitale Signatur

Die digitale bzw. elektronische Signatur ist eine schlüsselabhängige Prüfsumme, die von einer Nachricht oder einem Dokument in Kombination mit einem Schlüssel erzeugt wird. Wird die Signatur an eine Nachricht oder ein Dokument angehängt, dann gilt das als unterschrieben. Für digitale Nachrichten und Dokumente werden digitale Signaturen verwendet, um ihre Echtheit glaubhaft und prüfbar zu machen. Die Echtheit der Signatur kann elektronisch geprüft werden.

Digitale Signaturen sind in der Datenübertragung deshalb notwendig, weil sich der Absender von Nachrichten und Dokumenten fälschen lässt. Beispielsweise ist es ganz einfach den Absender einer E-Mail zu fälschen. Das heißt, es ist möglich, dass sich jemand als eine andere Person ausgibt. Auch im wirklichen Leben kann man eine beliebige Absender-Adresse auf einen Brief schreiben. Um die Glaubwürdigkeit des Briefs zu unterstreichen setzen wir an das Briefende unsere Unterschrift. Genauso wird es mit der digitalen Signatur gemacht.

Digitale Signaturen gewährleisten:

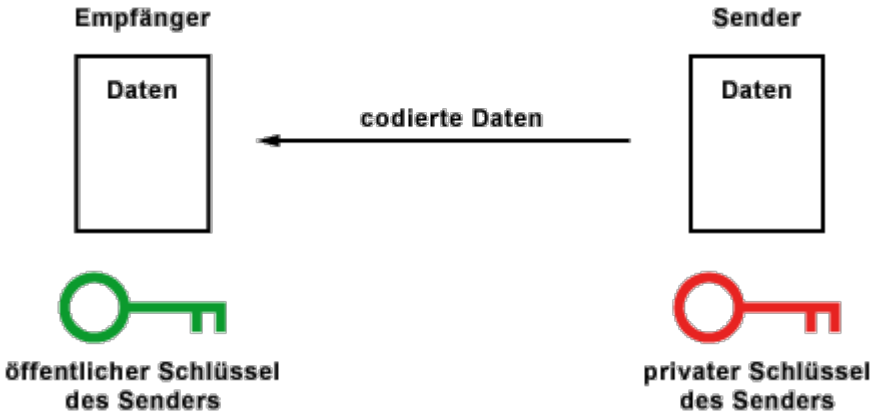
- **Integrität**: Die Nachricht, die man erhält, ist von keiner dritten Person manipuliert worden.
- **Authentizität**: Die Nachricht, die man erhält, stammt wirklich von der Person, die als Absender angegeben ist.
- **Verbindlichkeit**: Der Urheber kann nachträglich nicht bestreiten, die Nachricht verfasst zu haben.

dafür müssen folgende Anforderungen gelten:

- Die digitale Signatur darf nicht (unbemerkt) fälschbar sein.
- Die Echtheit der digitalen Signatur muss überprüfbar sein.
- Die digitale Signatur darf nicht von einem Dokument auf ein anderes (unbemerkt) übertragbar sein.
- Das signierte Dokument darf nicht (unbemerkt) veränderbar sein.

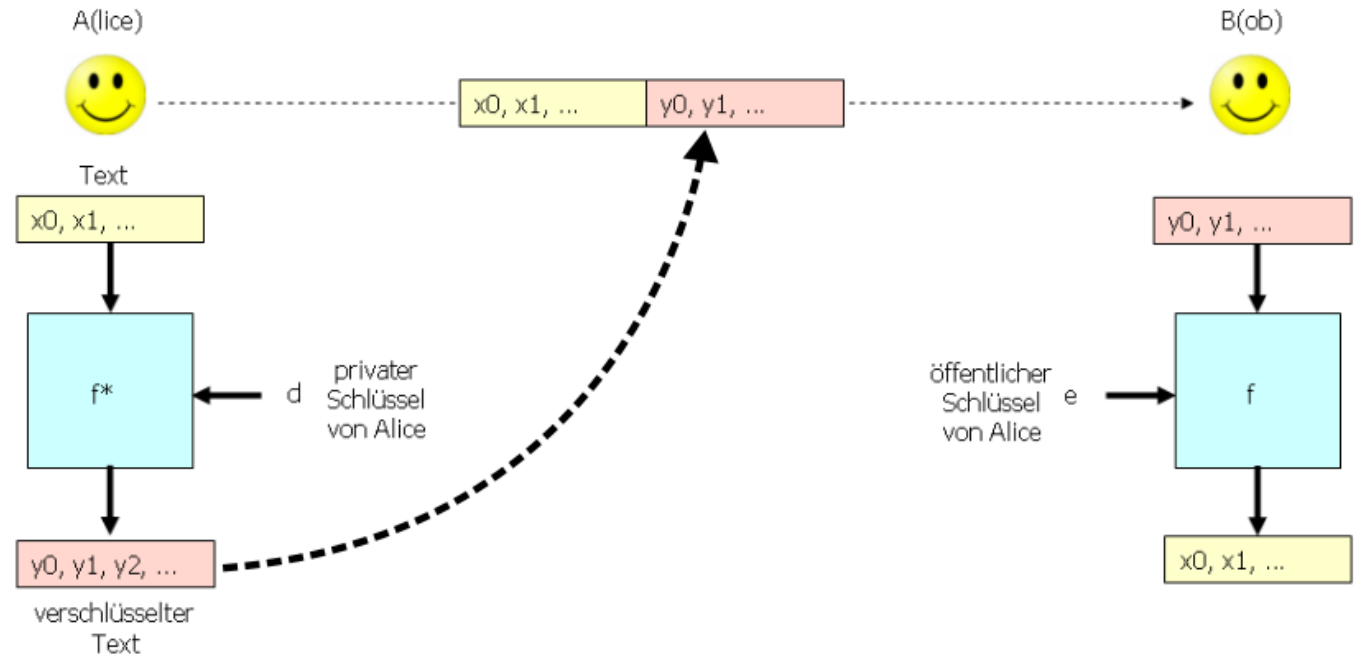
Eine Infrastruktur mit digitalen Signaturen macht nur dann Sinn, wenn die Signaturen in übertragenen Nachrichten und Dokumenten ständig geprüft werden. Nur dann kann erkannt werden, wenn eine Signatur, eine Nachricht oder ein Dokument gefälscht wurde. Wenn die Prüfung nicht erfolgt, dann bleiben Manipulationen „unbemerkt“, was nicht den Anforderungen der digitalen Signatur entspricht. Wird muss ein signiertes Dokument geändert werden, dann muss es erneut signiert werden, weil die alte Signatur nicht mehr zum Dokument passt.

Funktionsweise der digitalen Signatur



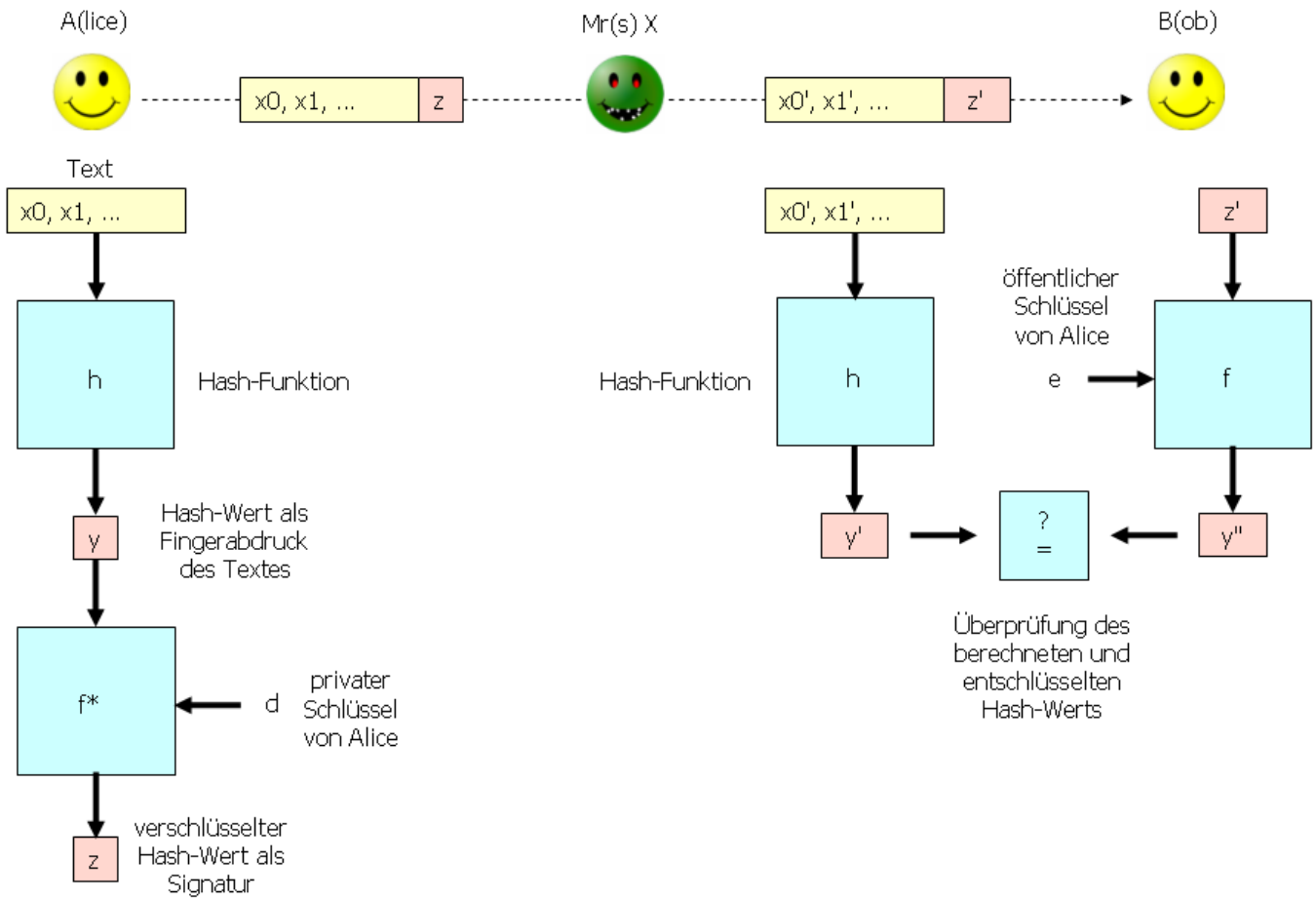
Die digitale Signatur basiert auf der asymmetrischen Kryptografie, wobei das verwendete **asymmetrische Verfahren umgekehrt** wird. Bei der asymmetrischen Verschlüsselung dient der öffentliche Schlüssel zum Verschlüsseln und der private Schlüssel zum Entschlüsseln. Bei der digitalen Signatur werden die Daten mit Kennzeichen versehen, die durch den privaten Schlüssel hinzugefügt werden. Mit dem öffentlichen Schlüssel kann man feststellen, ob die Daten von demjenigen stammen, der mit seinem privaten Schlüssel signiert hat und ob die Daten unverändert sind. Die Tatsache, dass der private Schlüssel durch seinen Besitzer geheim gehalten wird, erlaubt die Annahme, dass Daten, die mit dem privaten Schlüssel codiert sind, tatsächlich vom Schlüsselbesitzer stammen.

Um Nachrichten und Dokumente zu signieren muss dessen Ersteller die Nachricht mit seinem **privaten Schlüssel „entschlüsseln“**. Der dabei **entstandene „Klartext“ ist die digitale Signatur**. Sie wird an das Dokument angehängt. Die Signatur kann von jedem anderen mit dem **öffentlichen Schlüssel des Erstellers „verschlüsselt“ werden**. Dabei entsteht die ursprüngliche Nachricht, die mit der unverschlüsselten Nachricht verglichen werden kann. Sind beide gleich, ist das Dokument unverändert und korrekt signiert.



Blockweise erstellte Signaturen haben zusammen mindestens die gleiche Länge, wie die Nachricht oder das Dokument selbst. Das ist recht unpraktisch. Das Signieren kann ein erheblicher Aufwand bezüglich Aufwand und Rechenleistung sein. Ebenso die Prüfung der Signatur. Deshalb wird **nie die ganze Nachricht signiert**. Statt dessen wird **aus der Nachricht zuerst eine Prüfsumme gebildet**, die viel kürzer sein kann als die Nachricht selbst. Und erst dann wird das Signaturverfahren auf diese Prüfsumme angewendet. Das Verfahren, mit dem die Prüfsumme gebildet wird, ist eine

kryptografische Hash-Funktion. Das Ergebnis der kryptografischen Hash-Funktion ist der Hash-Wert oder auch nur Hash genannt.



Ein Signiersystem benutzt in der Regel eine Hash-Funktion, um eine Art Fingerabdruck des zu signierenden Textes zu erstellen. Mit Hilfe eines Schlüsselpaares bestehend aus einem öffentlichen und einem privaten Schlüssel wird aus dem Fingerabdruck dann die digitale Signatur erstellt. Die Vorgehensweise soll anhand des in der Abbildung gezeigten Szenarios beschrieben werden.

Alice will eine signierte Nachricht an Bob senden.

In einem ersten Schritt erzeugt sie mit Hilfe einer Hash-Funktion einen Fingerabdruck des zu versendenden Textes. Bei dem Fingerabdruck handelt es sich um ein Bitmuster, das dem Text zugeordnet wird.

Diesen Fingerabdruck verschlüsselt Alice mit ihrem privaten Schlüssel. Das Ergebnis ist ein Bitmuster, das die digitale Signatur zum vorgegebenen Text bildet.

Alice sendet jetzt den Text mit der digitalen Signatur an Bob.

Wie überprüft Bob die Integrität der erhaltenen Nachricht?

Bob benutzt dieselbe Hash-Funktion wie Alice, um einen Fingerabdruck zum übermittelten Text zu erzeugen.

Bob ist im Besitz des öffentlichen Schlüssels von Alice und benutzt ihn, um die übermittelte Signatur zu entschlüsseln.

Wenn die Nachricht nicht verändert wurde, dann erhält Bob durch die Entschlüsselung der Signatur den von Alice erzeugten Fingerabdruck zum versendeten Text. Dieser ist dann identisch mit dem von Bob bestimmten Fingerabdruck zum empfangenen Text.

Wenn die Nachricht in Teilen verändert wurde, dann müsste das Bob beim Vergleich der Fingerabdrücke auffallen. Wenn Mr(s) X. z.B. den Text abändert, dann ändert sich auch der Fingerabdruck zum Text. Mr(s) X. kann zwar einen Fingerabdruck zum veränderten Text erzeugen, Mr(s) X. kann ihn aber nicht passend verschlüsseln, da Mr(s) X. keinen Zugang zum privaten Schlüssel von Alice hat (davon gehen wir hier natürlich aus). Mr(s) X. ist demnach nicht in der Lage, ein stimmiges Paar bestehend aus einem veränderten Text und einer hierzu passenden mit dem privaten Schlüssel von Alice erzeugten Signatur zu erzeugen.

Bob kann zudem die Authentizität der Nachricht überprüfen, d.h., ob die Nachricht tatsächlich von Alice stammt. Nur Alice hat Zugriff auf ihren privaten Schlüssel (davon gehen wir hier aus). Wenn Bob den öffentlichen Schlüssel von Alice benutzt (und dieser tatsächlich auch von Alice stammt), dann passt dieser öffentliche Schlüssel nur zum privaten Schlüssel von Alice. Bob erhält nur dann identische Fingerabdrücke, wenn der gesendete Text mit dem privaten Schlüssel von Alice - also von Alice - signiert wurde.

Da nur Alice Zugriff auf ihren privaten Schlüssel hat, kann Alice nachträglich nicht bestreiten, die signierte Nachricht an Bob verschickt zu haben. Mit der digitalen Signatur wird also auch die Verbindlichkeit des Nachrichtenaustauschs garantiert.

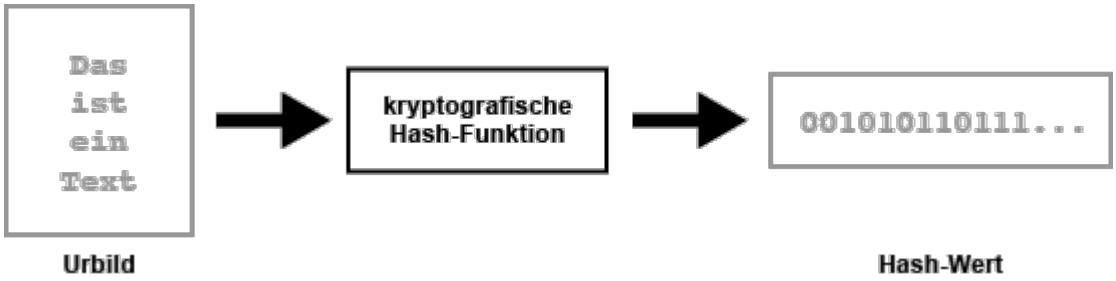
Bekannte Verfahren

Bekanntlich basieren Signaturverfahren auf asymmetrischen Verfahren, die sehr langsam arbeiten und von denen es nicht viele gibt. Das bekannteste und wohl am meisten eingesetzte Signaturverfahren ist RSA. Es gibt aber auch noch die Discrete Logarithm Signature Systems (DLSS). Dabei handelt es sich um eine Gruppe von Signaturverfahren auf Basis des diskreten Logarithmus. Dazu gehören ElGamal und DSA.

- RSA
- ElGamal
- DSA

Es gibt weitere DLSS-Verfahren, die in der Praxis nicht so häufig anzutreffen sind und deshalb hier nicht genannt werden.

Kryptografische Hash-Funktionen



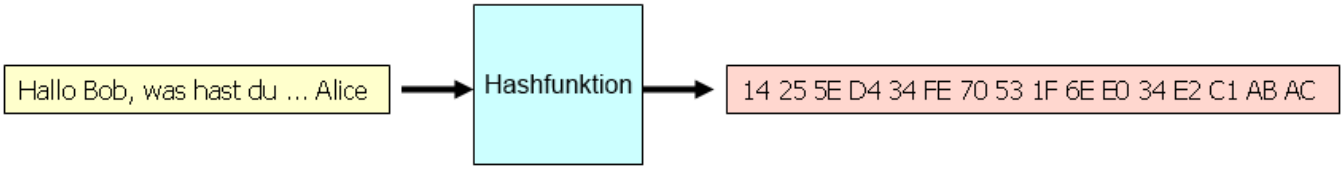
Kryptografische Hash-Funktionen sind ein wichtiges kryptografisches Instrument und bilden einen eigenen Bereich in der Kryptografie. Kryptografische Hash-Funktionen generieren aus beliebig langen Datensätzen eine Zeichenkette mit einer festen Länge (Angabe in Bit). Ein Datensatz kann ein Wort, ein Satz, ein längerer Text oder auch eine ganze Datei sein. Die erzeugte Zeichenkette wird als digitaler Fingerabdruck (Fingerprint), kryptografische Prüfsumme, Message Digest (MD) oder Message Authentication Code (MAC) bezeichnet. Gemeint ist damit in der Regel immer der sogenannte Hash-Wert oder auch nur Hash. Das ist ein digitaler Code, der nach Anwendung der kryptografischen Hash-Funktion als Ergebnis herauskommt.

Das Bilden eines Hash-Werts hat erst einmal nichts mit Kryptografie zu tun. Denn nicht alle Hash-Funktionen sind nach den Gesichtspunkten der Kryptografie eine kryptografische Hash-Funktion. Für „echte“ kryptografische Hash-Funktionen gibt es die unterschiedlichsten Begriffe und zusätzlich auch noch Produktbezeichnungen oder Leistungsmerkmale, die allerdings nichts darüber aussagen, ob sie kryptografischen Anforderungen entsprechen.

- Footprint-Funktion
- sichere Hash-Funktion
- Manipulation Detection Code (MDC)
- Message Integrity Code (MIC)
- Prüfsummenverfahren

Hash-Funktion

Eine Hash-Funktion ist eine Funktion, die Zeichenketten neue Zeichenketten einer fest vorgegebenen Länge zuordnet. Man nennt Funktionswerte von Hash-Funktionen auch Hash-Werte.



Im Prinzip erzeugt eine Hash-Funktion aus einem Datensatz, das als Urbild oder im Englischen Preimage bezeichnet wird, eine duale Zahl, die meist in hexadezimaler Schreibweise dargestellt und als Hash-Wert bezeichnet wird. Die Funktionsweise einer kryptografischen Hash-Funktion basiert auf einer Einwegfunktion, die sich sehr einfach rechnen lässt, aber deren Umkehrung dagegen sehr aufwendig bis unmöglich ist. Die Umkehrung vom Hash-Wert auf das Urbild zu schließen ist das was

man verhindern möchte.

Hash-Funktion als Einwegfunktion

Die in der Kryptologie benutzten Hash-Funktionen sind in der Regel Einwegfunktionen.

Bei einer Einwegfunktion ist es praktisch unmöglich, aus einem möglichen Zielwert einen Ausgangswert so zu bestimmen, dass der Zielwert Funktionswert zum Ausgangswert ist.

In mathematischer Kurzform kann man das so beschreiben: Eine Funktion f ist eine Einwegfunktion, wenn es praktisch unmöglich ist, zu gegebenem y aus der Zielmenge ein x aus der Definitionsmenge von f zu finden, so dass f(x) = y gilt.

Anforderungen an kryptografische Hash-Funktionen

- **Eindeutigkeit**: Eine identische Zeichenfolge muss zum selben Hash-Wert führen.
- **Reversibilität**: Der Hash-Wert darf nicht in die ursprüngliche Zeichenfolge zurückberechnet werden können.
- **Kollisionsresistenz**: Zwei unterschiedliche Zeichenfolgen dürfen nicht den gleichen Hash-Wert ergeben.

Nicht alle Hash-Funktionen erfüllen alle diese Anforderungen. Deshalb eignen sich nicht alle Hash-Funktionen für kryptografische Anwendungen, wie Authentisierung und Verschlüsselung.

Reversibilität

Grundsätzlich sollte es nicht möglich sein aus einem Hash-Wert die ursprünglichen Daten zurückzuberechnen. Weil mit der Zeit doch Möglichkeiten gefunden werden und die Rechenleistung steigt, gibt es immer bessere Verfahren aus einem Hash-Wert die ursprünglichen Daten zurück zu berechnen. Deshalb stellt sich mit der Zeit immer wieder heraus, dass Hash-Funktionen reversibel sind.

Kollisionsresistenz

Prinzipiell ist es so, dass ein Urbild beliebig viele Stellen und beliebig viele Werte einnehmen kann. Ein Hash-Wert ist allerdings auf eine bestimmte Länge begrenzt. So kann es vorkommen, dass ein beliebiger Hash-Wert unterschiedlichen Urbildern entspricht. Man spricht dann von einer Kollision. Bei einer guten Hash-Funktion sollte eine Kollision so wenige wie möglich vorkommen. Nehmen wir als Beispiel die Quersummenbildung. Hier kann es vorkommen, dass die Quersumme mehreren Zahlenwerten entsprechen kann. Aus Sicht der Kryptografie ist die Quersummenbildung also keine kryptografische Hash-Funktion. Die Kryptografie stellt an Hash-Funktionen und ihre Anwendungen höhere Anforderungen. Es sollte für einen Angreifer unmöglich sein Kollisionen zu erzeugen.

- Statistisch gesehen sollte jeder Hash-Wert etwa gleich oft vorkommen.
- Der Hash-Wert sollte auch bei kleinen Änderungen des Urbilds anders sein.

Um die Wahrscheinlichkeit von Kollisionen zu vermeiden, verwendet man immer bessere Verfahren, die meist längere Hash-Werte erzeugen. Beispielsweise sind die bekannten und beliebten Hash-Funktionen MD5 und SHA1 für Kollisions-Attacken verwundbar. Damit ist gemeint, dass ein anderer Datensatz den gleichen Hash-Wert erzeugen kann. Das heißt, dass ein MD5- oder SHA1-Hash nicht einzigartig ist. Besser ist es, SHA256 oder gleich SHA512 zu verwenden.

Angriffsszenarien

Bei einem **Urbildangriff** (engl. preimage attack) verfolgt der Angreifer das Ziel, zu einem gegebenen Hashwert einer unbekannten Nachricht (Erster Urbildangriff) oder zu einer gegebenen Nachricht selbst (Zweiter Urbildangriff) eine weitere Nachricht zu konstruieren, die denselben Hashwert besitzt.

Beispiel: Angenommen, ein Angreifer fängt ein signiertes Dokument ab. Er ist dann im Besitz des Dokumenttextes (z.B. „Hiermit bestelle ich 2 Konzertkarten zu je 40 €.“) sowie des zugehörigen Hashwerts. Der Angreifer versucht jetzt, aus der Nachricht oder dem Hashwert eine veränderte Nachricht mit demselben Hashwert zu erzeugen. Eine zusätzliche Schwierigkeit besteht darin, dass die neue Nachricht auch noch Sinn machen soll.

Eine andere Form von Angriff betrifft die Erzeugung von Kollisionen:

Bei einem **Kollisionsangriff** (engl. collision attack) verfolgt der Angreifer das Ziel, zwei verschiedene Dokumente zu konstruieren, die beide denselben Hashwert besitzen.

Beachte, dass es sich hier um unterschiedliche Angriffsszenarien handelt. Das zeigt sich auch in der Praxis. Während Kollisionsangriffe bei SHA-1 möglich sind, sind Urbildangriffe bei SHA-2 derzeit noch nicht möglich.

Kryptografische Hash-Funktionen

Kryptografische Hash-Funktionen bilden einen eigenen Bereich in der Kryptografie. An deren Entwicklung waren oft bekannte Kryptografen beteiligt, die man von anderen kryptografischen Verfahren her kennt.

- MD2 - Message Digest 2 mit 128 Bit
- MD4 - Message Digest 4 mit 128 Bit
- MD5 - Message Digest 5 mit 128 Bit
- RIPEMD
- RIPEMD-160
- Tiger
- WHIRLPOOL
- SHA-1 mit 160 Bit
- SHA-2 mit 224, 256, 384 und 512 Bit
- SHA-3 mit 224, 256, 384 und 512 Bit

Mehr Infos zu MD5 und SHA

Sicherheitsinfrastruktur

Hier geht es um die Frage, wie man gewährleisten kann, dass ein öffentlich bereit gestellter Schlüssel tatsächlich zu der Person gehört, die sich als Eigentümer ausgibt.

Mit Hilfe eines asymmetrischen Kryptosystems können Nachrichten in einem Netzwerk digital signiert und verschlüsselt werden. Sichere Kryptosysteme können bei geeigneter Wahl der Parameter (z. B. der Schlüssellänge) auch bei Kenntnis des Verfahrens (vgl. Kerckhoffs' Prinzip) zumindest nach heutigem Kenntnisstand nicht in überschaubarer Zeit gebrochen werden.

In asymmetrischen Kryptosystemen benötigt der Sender für eine verschlüsselte Übermittlung den öffentlichen Schlüssel (public key) des Empfängers. Dieser könnte z. B. per E-Mail versandt oder von einer Web-Seite heruntergeladen werden. Dabei muss sichergestellt sein, dass es sich tatsächlich um den Schlüssel des Empfängers handelt und nicht um eine Fälschung eines Betrügers.

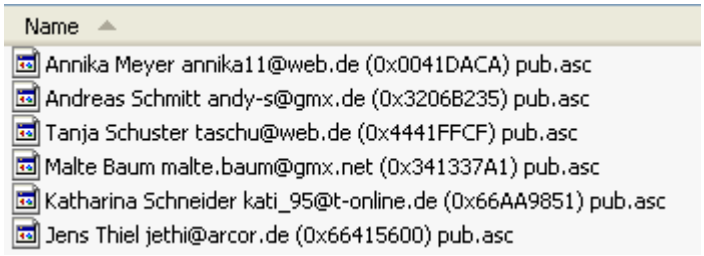
Hierzu dienen nun digitale Zertifikate, die die Authentizität eines öffentlichen Schlüssels und seinen zulässigen Anwendungs- und Geltungsbereich bestätigen. Das digitale Zertifikat ist selbst durch eine digitale Signatur geschützt, deren Echtheit mit dem öffentlichen Schlüssel des Ausstellers des Zertifikates geprüft werden kann.

Um die Authentizität des Ausstellerschlüssels zu prüfen, wird wiederum ein digitales Zertifikat benötigt. Auf diese Weise lässt sich eine Kette von digitalen Zertifikaten aufbauen, die jeweils die Authentizität des öffentlichen Schlüssels bestätigen, mit dem das vorhergehende Zertifikat geprüft werden kann. Eine solche Kette von Zertifikaten wird Validierungspfad oder Zertifizierungspfad genannt. Auf die Echtheit des letzten Zertifikates (und des durch dieses zertifizierten Schlüssels) müssen sich die Kommunikationspartner ohne ein weiteres Zertifikat verlassen können.

Vertrauen in Schlüssel

Blindes Vertrauen in Schlüssel als Problem

Andreas, Annika, Jens, Katharina, Malte und Tanja haben ihre öffentlichen Schlüssel in ein gemeinsam zugängliches Verzeichnis kopiert, so dass jeder sich die benötigten öffentlichen Schlüssel kopieren kann.



Bei einem erneuten Blick in das Verzeichnis taucht plötzlich ein weiterer Schlüssel auf.

Mit welchen öffentlichen Schlüssel soll Annika nun Jens die Nachricht verschlüsseln und schicken?

Schlüsselserver?

Öffentliche Schlüssel von Kommunikationspartnern kann man sich auch auf Schlüsselservern wie z.B. keys.openpgp.org besorgen. Aus Datenschutzgründen sind Recherchen auf diesen Servern heutzutage nur noch gezielt möglich. D.h. man muss nach einem Fingerabdruck (Fingerprint) einer Person suchen, den man erhalten hat, oder nach der E-Mail Adresse, die einem bekannt ist. Beispielsweise findet man den Fingerabdruck von Linus Neumann auf seiner Website. Zu diesem Fingerabdruck kann man dann auf https://keys.openpgp.org oder https://keyserver.ubuntu.com den passenden Key herunterladen.

Schlüssel überprüfen

Bevor du den öffentlichen Schlüssel einer anderen Person benutzt, solltest du dich von der Echtheit des Schlüssels überzeugen.

Wenn der öffentliche Schlüssel dir direkt von einem (vertrauenswürdigen) Bekannten übergeben wird, dann kannst du (in der Regel) von einem echten Schlüssel ausgehen.

Wenn du einen öffentliche Schlüssel über ein unsicheres Kommunikationsmedium (z.B. per E-Mail) erhalten hast oder von einem Schlüsselserver heruntergeladen hast, dann solltest du dich vergewissern, ob die Angaben zum Eigentümer stimmen. Du kannst dich z.B. mit der Person treffen (oder - wenn du die Stimme eindeutig erkennst - mit der Person telefonieren) und einen Datenabgleich machen.

Beim Datenabgleich solltest du den Fingerabdruck des Schlüssels genauestens überprüfen.

Schlüssel zertifizieren

Erst wenn du die Echtheit eines Schlüssels genauestens geprüft hast bzw. wenn du ganz sicher bist, dass ein Schlüssel zu einer bestimmten Person gehört, dann solltest du den betreffenden Schlüssel zertifizieren.

Einen Schlüssel zertifiziert man, indem man ihn mit einer digitalen Signatur versieht.

Web of Trust

Echtheit öffentlicher Schüssel als Problem

Zum Verschlüsseln von Dokumenten bzw. zum Überprüfen digitaler Signaturen benötigt man den öffentlichen Schlüssel des Kommunikationspartners. Aber, wie kommt man an den gewünschten öffentlichen Schlüssel? Oft ist eine persönliche Übergabe nicht möglich und man muss sich den öffentliche Schlüssel (über ein unsicheres Kommunikationsmedium) schicken lassen oder auf einen Schlüsselserver besorgen. In beiden Fällen ergibt sich dann das Problem, dass man nicht sicher sein kann, dass der erhaltene Schlüssel tatsächlich zu der angegebenen Person gehört.

Schlüssel zertifizieren

Wenn man (nach einer sorgfältigen Überprüfung) sicher ist, dass ein Schlüssel zu der angegebenen Person gehört, dann kann man den Schlüssel mit seiner digitalen Unterschrift beglaubigen / zertifizieren.

In der folgenden Abbildung kann man solche Zertifizierungen erkennen (z.B.: snerz@bvpk.org)

Search results for 'snerz@bvpk.org'

```
Type bits/keyID            cr. time   exp time   key expir
```

```
pub (4)dsa1024/39c40c2fca5b31804c6f82e184d2b243449d222e 2008-11-28T21:30:27Z

uid Sebastian Nerz <basti@tirsales.de>
sig cert  84d2b243449d222e 2008-11-28T21:33:32Z ____________________ ____________________ [selfsig]
sig cert  84d2b243449d222e 2008-11-28T21:30:27Z ____________________ ____________________ [selfsig]
sig cert  926ae196022ce281 2009-08-30T00:30:30Z ____________________ ____________________ 926ae196022ce281
sig cert  9f587ee862a25e4e 2009-08-30T11:44:39Z ____________________ ____________________ 9f587ee862a25e4e
sig cert  021b54a7fe1dd1df 2009-08-30T11:47:14Z ____________________ ____________________ 021b54a7fe1dd1df
sig cert  a48fb68bd58cb000 2009-08-30T14:49:28Z ____________________ ____________________ a48fb68bd58cb000
sig cert  2189b0bd2c8c1429 2009-09-02T10:00:21Z ____________________ ____________________ 2189b0bd2c8c1429
sig cert  208f84c45ff25b4d 2010-04-19T19:37:21Z ____________________ ____________________ 208f84c45ff25b4d
sig cert  c0ac6eb914fe16c1 2010-06-01T14:30:48Z ____________________ ____________________ c0ac6eb914fe16c1
sig cert  be7d227833742d65 2010-06-01T14:31:26Z ____________________ ____________________ be7d227833742d65
sig cert  4473759d0191d5ed 2010-12-27T03:24:49Z ____________________ ____________________ 4473759d0191d5ed
```

Klickt man auf die erste Zertifizierung, sieht man dass hier z.B. ein Benutzer mit dem Namen Matthias Binninger den öffentlichen Schlüssel von Sebastian Nerz mit seinem privaten Schlüssel signiert hat. Zusätzlich ist der öffentlichen Schlüssel von Sebastian Nerz mit seinem eigenen privaten Schlüssel signiert (selfsig). Eine solche Selbstzertifizierung wird vom benutzen System standardmäßig vorgenommen.

keyserver.ubuntu.com/pks/lookup

Search results for '0x926ae196022ce281'

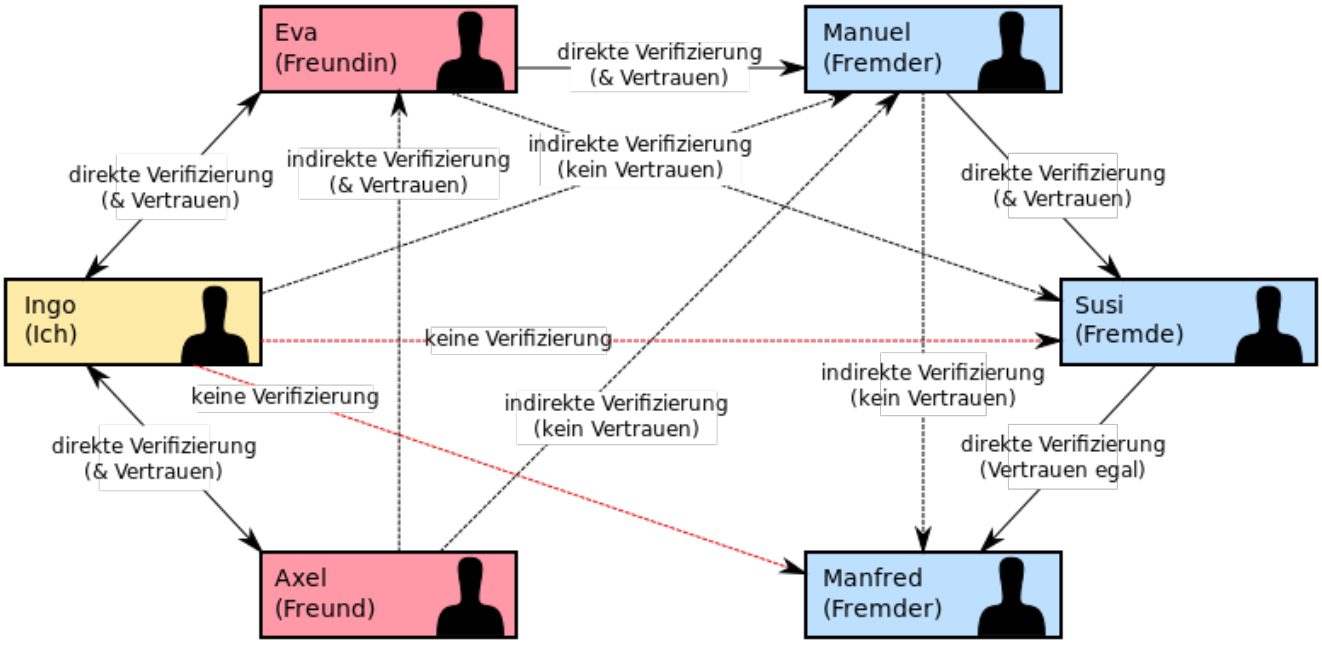
```
Type bits/keyID           cr. time   exp time   key expir
```

```
pub (4)rsa2048/926ae196022ce281 2009-08-28T01:35:51Z

uid Matthias Binninger <mail@matthias-binninger.de>
sig cert  926ae196022ce281 2009-08-28T01:39:35Z ____________________ ____________________ [selfsig]
sig cert  926ae196022ce281 2009-08-28T01:35:51Z ____________________ ____________________ [selfsig]
sig cert  f2bb81d8cc8d3de2 2009-08-30T08:40:39Z ____________________ ____________________ f2bb81d8cc8d3de2
sig cert  9f587ee862a25e4e 2009-08-30T10:26:06Z ____________________ ____________________ 9f587ee862a25e4e
sig cert  1133ff1848a587a7 2009-08-30T10:50:03Z ____________________ ____________________ 1133ff1848a587a7
sig cert  021b54a7fe1dd1df 2009-08-30T11:49:20Z ____________________ ____________________ 021b54a7fe1dd1df
sig cert  58c691ffedcb6ea7 2009-08-30T12:00:50Z ____________________ ____________________ 58c691ffedcb6ea7
sig cert  a48fb68bd58cb000 2009-08-30T14:49:16Z ____________________ ____________________ a48fb68bd58cb000
sig cert  7d5ba9edd43794cc 2009-08-30T20:51:57Z ____________________ ____________________ 7d5ba9edd43794cc
sig cert  f8c376a1a2c51749 2009-08-30T21:32:22Z ____________________ ____________________ f8c376a1a2c51749
sig cert  2189b0bd2c8c1429 2009-09-02T09:48:46Z ____________________ ____________________ 2189b0bd2c8c1429
sig cert  02907183f0818b12 2009-09-06T18:59:09Z ____________________ ____________________ 02907183f0818b12
sig cert  646cbf3ad40db3d7 2010-03-31T23:19:15Z ____________________ ____________________ 646cbf3ad40db3d7
```

Aufbau eines Vertrauensnetzes (Web of Trust)

Durch das Zertifizieren von Schlüsseln lässt sich ein Vertrauensnetz aufbauen.



Die vorliegende Abbildung geht von folgenden Zertifizierungen aus:

```
Ingo zertifiziert den öffentlichen Schlüssel von Eva und Axel.
```

```
Eva zertifiziert den öffentlichen Schlüssel von Ingo und Manuel.
...
```

Hierdurch werden Vertrauensbeziehungen erstellt:

```
Ingo vertraut den Angaben des öffentlichen Schlüssels von Eva und Axel.
Eva vertraut den Angeben des öffentlichen Schlüssels von Ingo und Manuel.
...
```

Nach dem Motto der Freund meines Freundes ist auch mein Freund pflanzen sich Vertrauensbeziehungen fort:

```
Ingo vertraut den Angaben des öffentlichen Schlüssels von Eva.
Eva vertraut den Angeben des öffentlichen Schlüssels von Manuel.
```

also:

```
Ingo vertraut den Angaben des öffentlichen Schlüssels von Manuel.
```

Insgesamt ergibt sich auf diese Weise ein Netzwerk von Vertrauensbeziehungen.

In einem Web of Trust funktioniert das so:

1. Alice erzeugt für sich ein Schlüsselpaar und signiert es. Außerdem schickt sie den öffentlichen Teil an einen Schlüsselserver (key server), damit andere Teilnehmer leichten Zugriff darauf haben.
2. Bob möchte mit Alice verschlüsselt kommunizieren. Dazu besorgt er sich Alices Schlüssel von einem Schlüsselserver, muss aber noch sicherstellen, dass er wirklich den richtigen Schlüssel bekommen hat: Ein Angreifer könnte sich für Alice ausgeben und einen von ihm erzeugten Schlüssel an den Schlüsselserver schicken. Jeder, der meint, eine Nachricht nur für Alice zu verschlüsseln, würde sie in Wirklichkeit für den Angreifer verschlüsseln.
3. Bob bittet Alice (z. B. bei einem Telefonanruf oder einem persönlichen Treffen) um den Fingerprint ihres öffentlichen Schlüssels. Diesen vergleicht er mit dem des Schlüssels, den er vom Schlüsselserver erhalten hat.
4. Stimmen beide Fingerprints überein, kann Bob davon ausgehen, den richtigen Schlüssel erhalten zu haben. Darum signiert er den öffentlichen Schlüssel von Alice (genauer: eine oder mehrere ihrer User-IDs) mit seinem privaten und schickt diese Signatur an den Schlüsselserver.
5. Möchte jetzt Carl mit Alice verschlüsselt kommunizieren, besorgt er sich genau wie Bob Alices öffentlichen Schlüssel vom Schlüsselserver. Dann stellt er fest, dass Bob Alices Schlüssel bereits überprüft hat. Wenn Carl Bobs Schlüssel schon kennt und er Bob vertraut, dass Bob vor der Signatur fremder Schlüssel eine gründliche Überprüfung durchführt, dann muss er nicht erst Alice treffen und diese Prüfung wiederholen. Er vertraut dem Schlüssel von Alice allein aufgrund Bobs vertrauenswürdiger Signatur. Wenn Carl sein Sicherheitsniveau erhöhen möchte oder er speziell Bobs Signaturen nur eingeschränkt vertraut, kann er sein Kryptosystem so konfigurieren, dass mehrere von ihm akzeptierte Signaturen vorhanden sein müssen, damit ein Schlüssel automatisch als gültig angesehen wird.

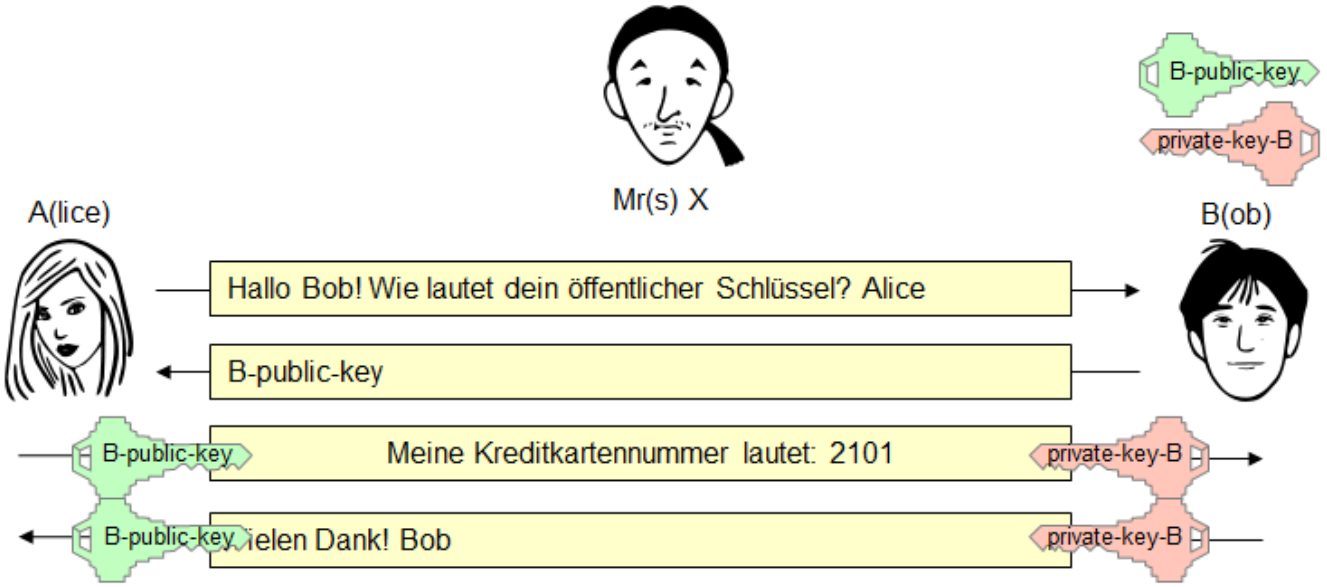
Datenschutzprobleme

Beachte, dass beim Veröffentlichen von Schlüsseln auch persönliche Daten veröffentlicht werden. Beachte auch, dass einmal veröffentliche Schlüssel mitsamt der persönlichen Daten nicht wieder gelöscht werden können.

(Wo)Man in the middle Angriff

Mr(s) X hört mit!

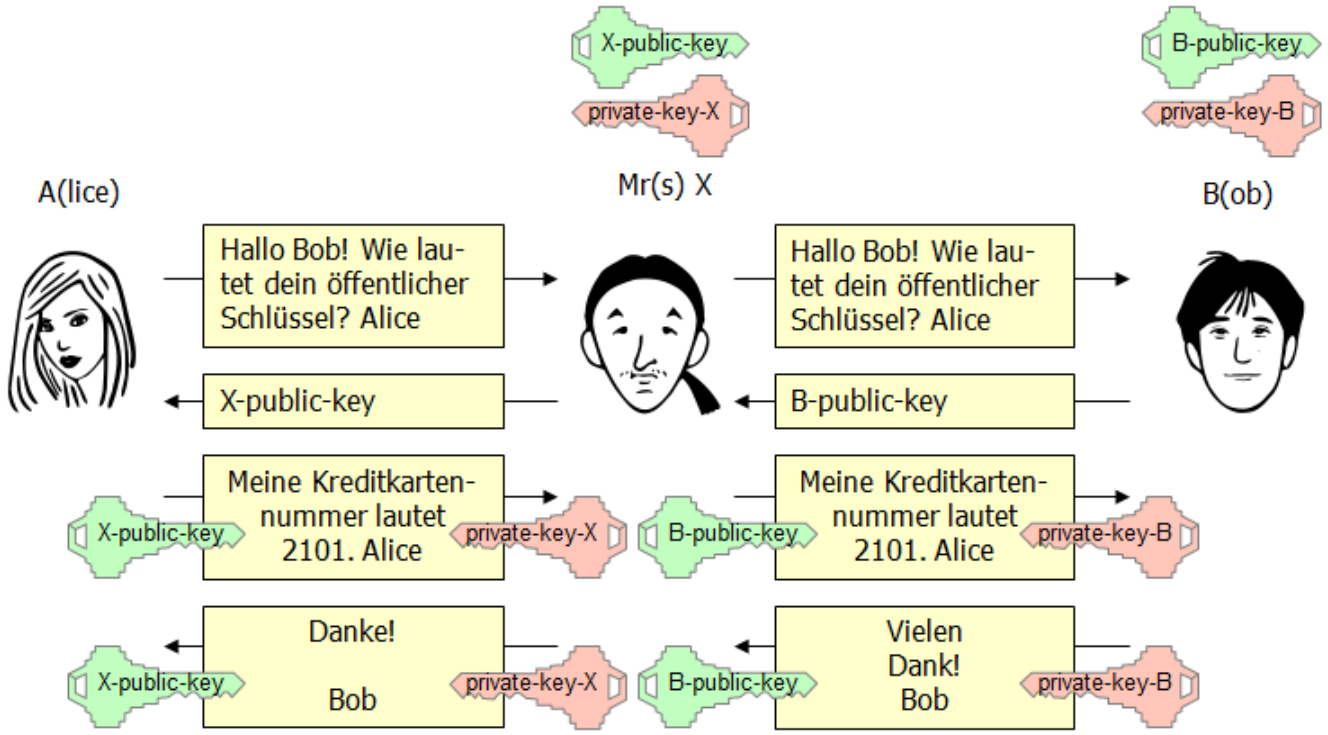
Alice möchte bei Bobs Internetversand eine Bestellung aufgeben und muss ihm deshalb die Nummer ihrer Kreditkarte übermitteln. Diese vertrauliche Information soll keinesfalls in die Hände von Mr(s) X fallen.



Bob hat sich ein Schlüsselpaar bestehend aus einem öffentlichen und einem privaten Schlüssel erzeugt. Bob schickt seinen öffentlichen Schlüssel auf Anfrage an Alice. Alice benutzt nun diesen Schlüssel, um ihre Kreditkartennummer an Bob zu verschicken. Bob bestätigt den Erhalt der Kreditkartennummer mit einer signierten Nachricht. Beurteile die Sicherheit des beschriebenen Szenarios: Kann sich Alice sicher sein, dass sie wirklich Bobs öffentlichen Schlüssel erhält?

Mr X schiebt sich dazwischen

Mr(s) X hat eine Möglichkeit gefunden, den Datenverkehr zwischen Alice und Bob abzufangen.

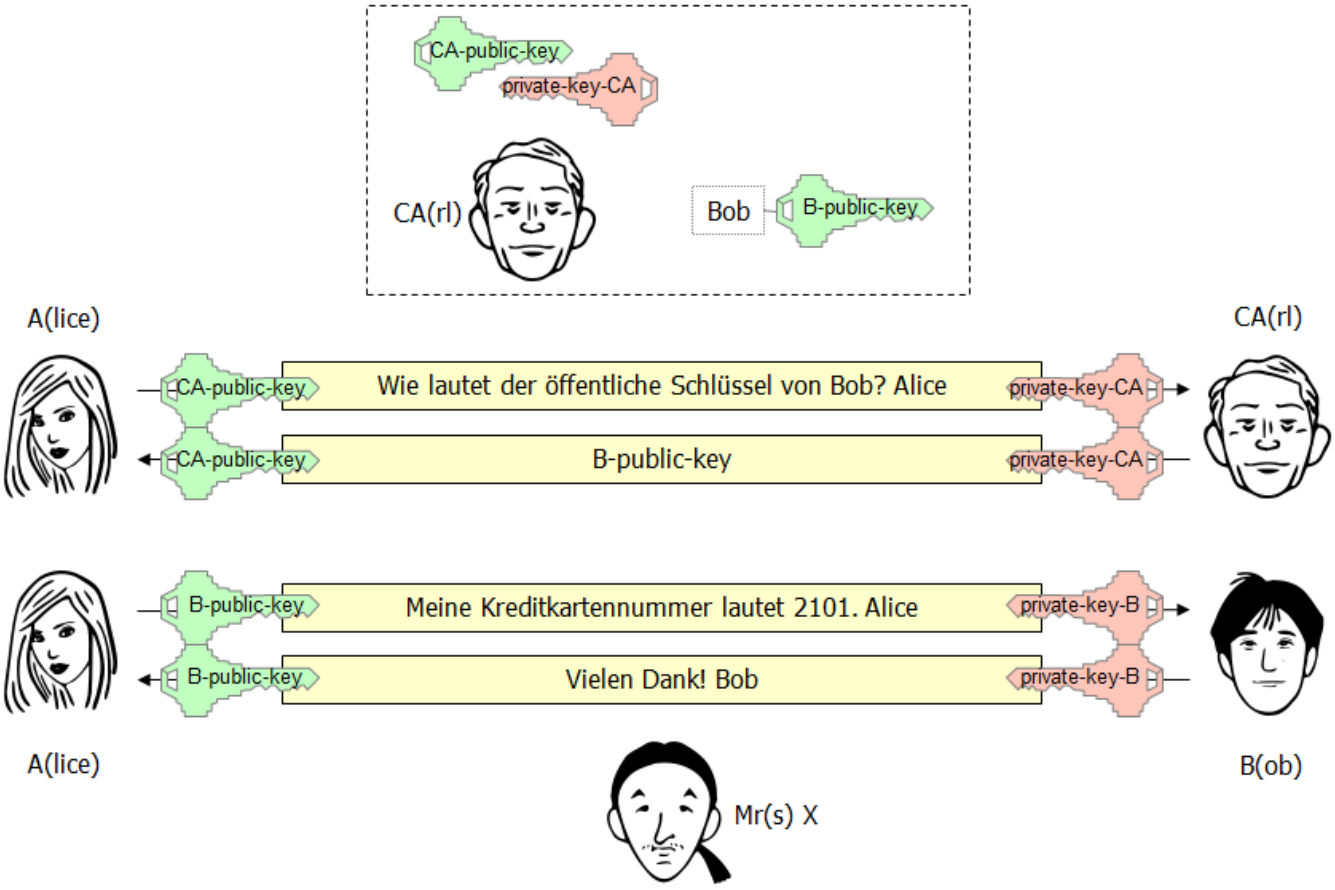


Warum merkt Alice oder Bob hier nichts vom (Wo)Man in the middle Angriff? Worin besteht die Schwierigkeit? Siehst du Lösungsansätze?

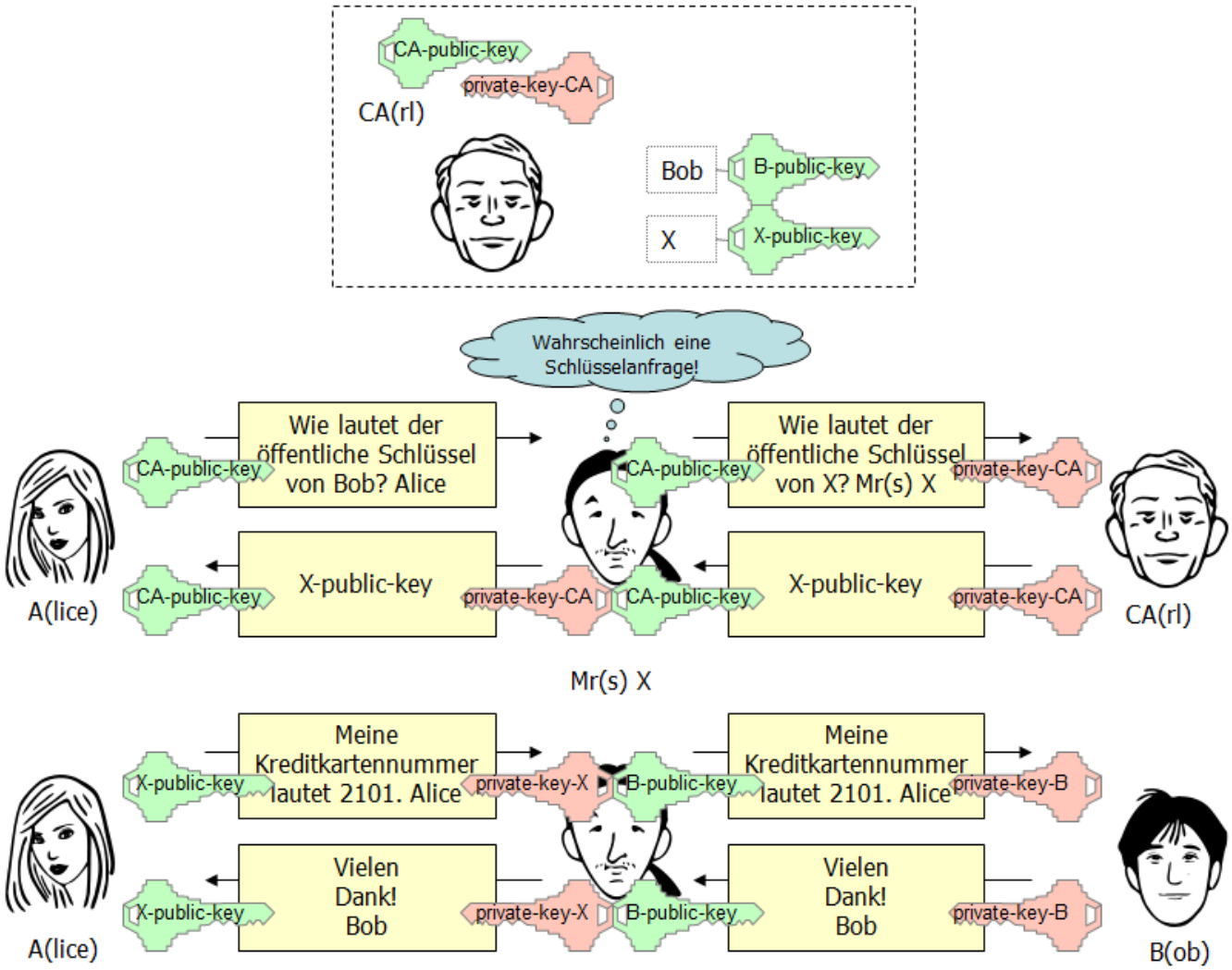
CA(rl) kommt ins Spiel!

CA(rl) bietet an, gegen ein Entgeld öffentliche Schlüssel zu verwalten. Er macht das aber nur, wenn man sie ihm vorab persönlich übergibt und sich ausweist. Als Gegenleistung überbringt CA(rl) seinen eigenen öffentlichen Schlüssel persönlich an alle gewünschten Personen.

Bob ist Internethändler und nimmt das Angebot von CA(rl) in Anspruch. Er hinterlässt seinen öffentlichen Schlüssel bei CA(rl) und beauftragt ihn, seinen - also CA(rl)s - öffentlichen Schlüssel an Alice zu übergeben.



- Wie erhält Alice hier den öffentlichen Schlüssel von Bob? Kann sie sicher sein, dass sie wirklich Bobs öffentlichen Schlüssel erhält?
- Mr(s) X kann sich auch hier zwischen Alice und Bob schieben. Analysiere hierzu die folgende Abbildung. Von welchen Annahmen geht man hier aus?

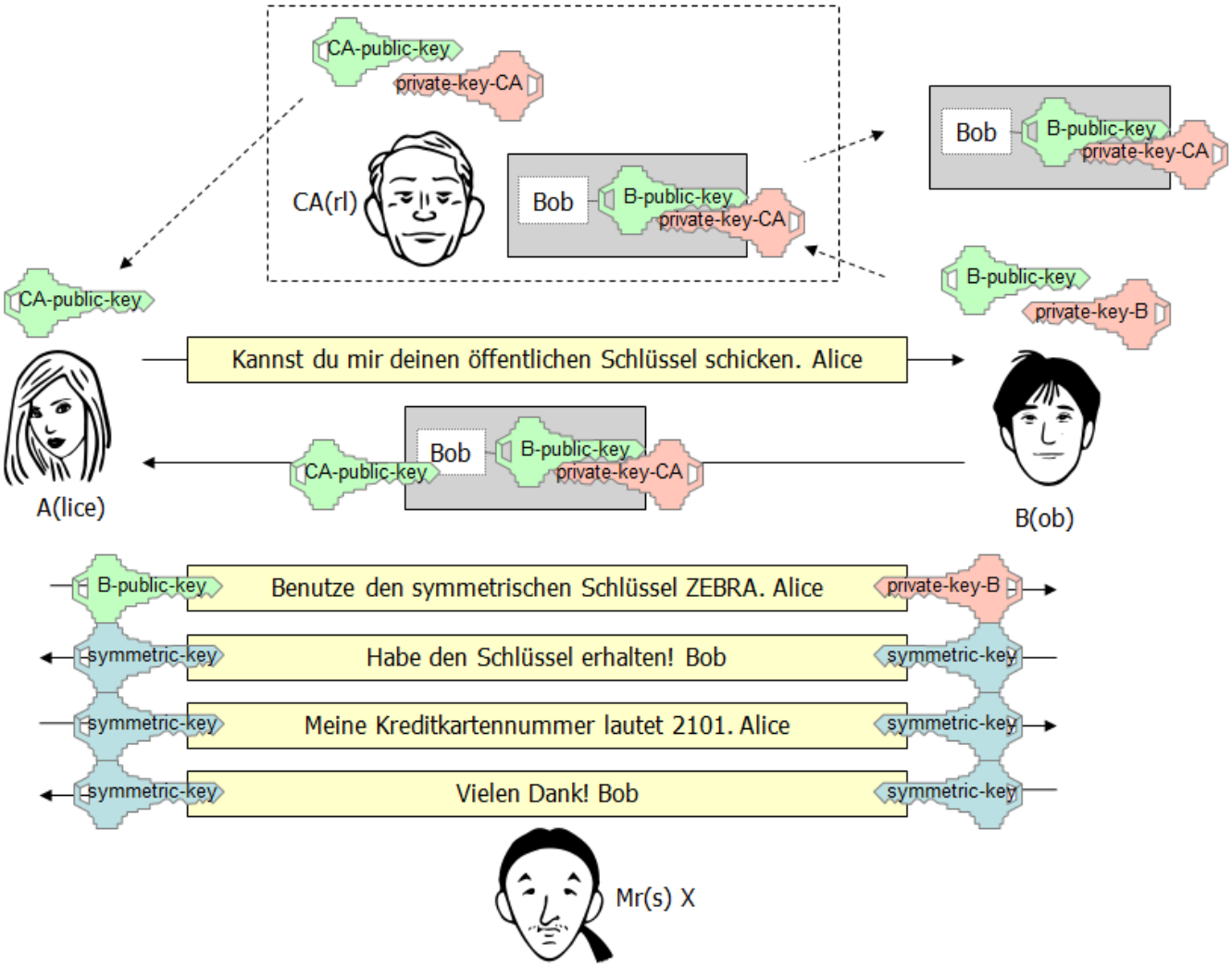


CA(rl) signiert die hinterlegten Schlüssel!

CA(rl) übernimmt jetzt die Rolle der sogenannten Certification Authority (CA). Seine Aufgabe ist es, die Echtheit von Schlüsseln zu gewährleisten.

CA(rl) bietet hier an, gegen ein Entgeld öffentliche Schlüssel zu signieren. Er macht das aber nur, wenn man sie ihm vorab persönlich übergibt und sich ausweist. Zusätzlich überbringt CA(rl) seinen eigenen öffentlichen Schlüssel persönlich an alle gewünschten Personen.

Bob ist Internethändler und nimmt das Angebot von CA(rl) in Anspruch. Er lässt seinen öffentlichen Schlüssel von CA(rl) signieren und beauftragt CA(rl), seinen - also CA(rl)s - öffentlichen Schlüssel an Alice zu übergeben.



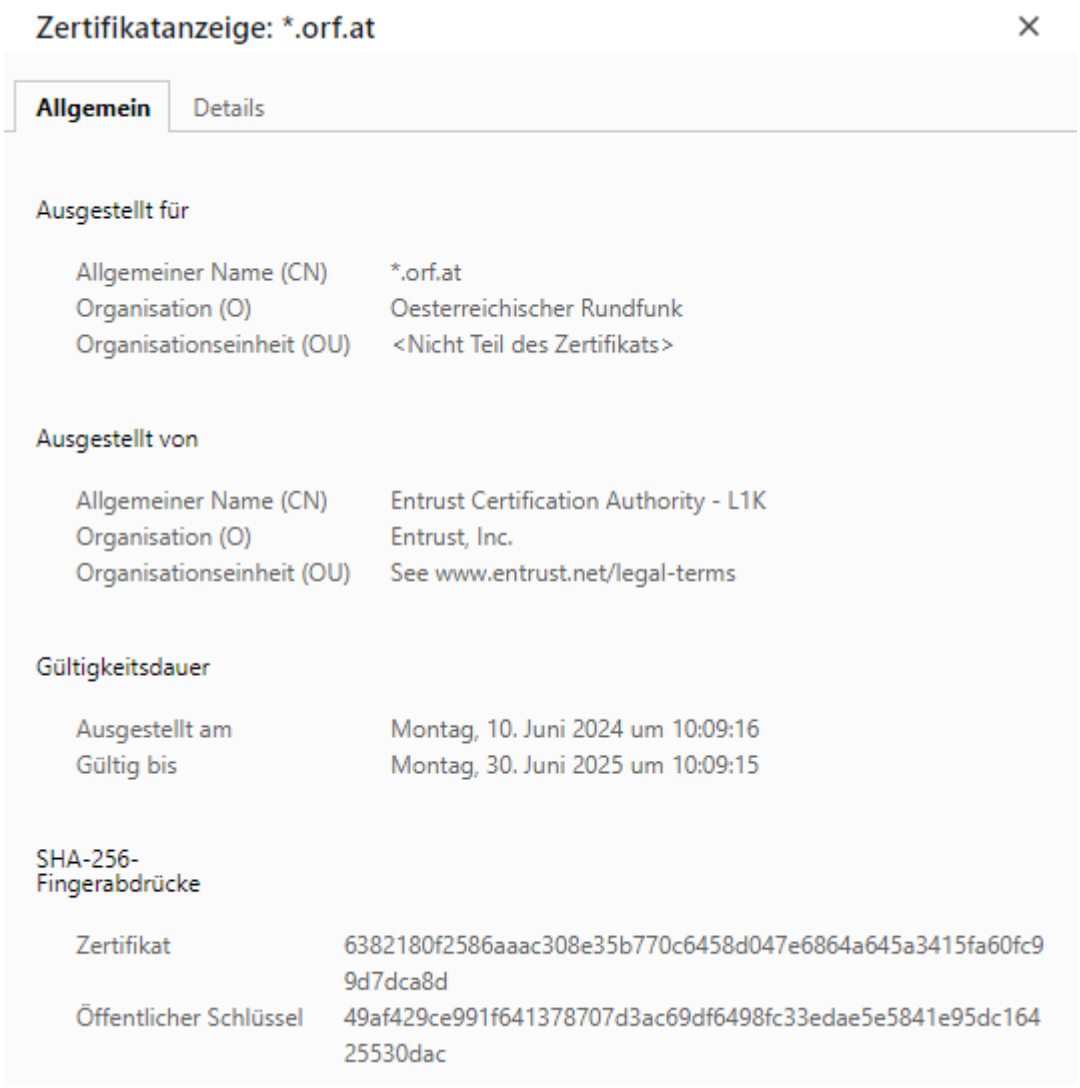
Was ist ein digitales Zertifikat

Wenn man den öffentlichen Schlüssel einer Person zum Verschlüsseln benutzt, dann sollte man auch sicher sein, dass dieser Schlüssel tatsächlich zur angegebenen Person gehört.

Ein **Public-Key-Zertifikat** dient dazu, die Zugehörigkeit eines öffentlichen Schlüssels zu einem bestimmten Eigentümer zu bestätigen.

Ein Public-Key-Zertifikat enthält in der Regel eine ganze Reihe von Informationen, u a.:

- den zu bestätigenden öffentlichen Schlüssel
- den Eigentümer des Schlüssels
- den Aussteller des Zertifikats
- die benutzten kryptografischen Verfahren
- die Gültigkeitsdauer des Zertifikats
- ...
- eine digitale Signatur des Ausstellers zur Bestätigung aller Informationen

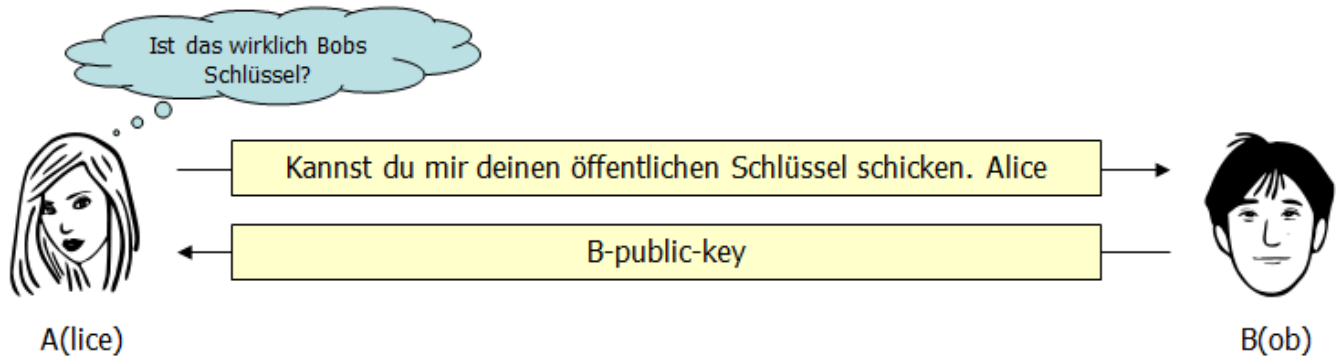


Public Key Infrastruktur (PKI)

Eine Public Key Infrastruktur (kurz: PKI) basiert auf einem asymmetrischen Kryptosystem, bei dem öffentliche und private Schlüssel zum Verschlüsseln und zum Signieren von Dokumenten benutzt werden.

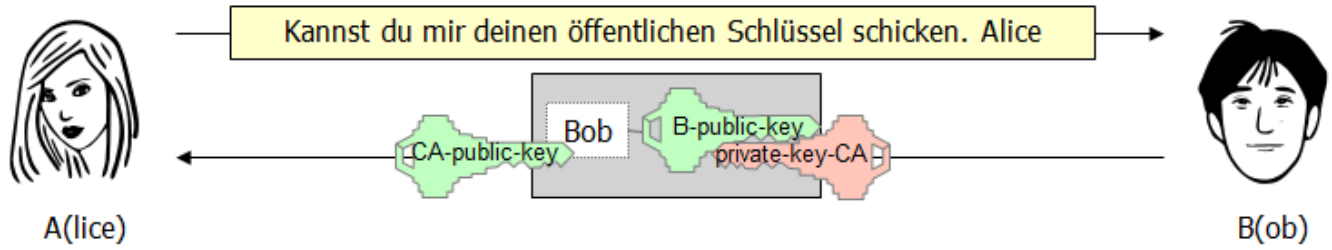
Zertifikate

Zum Verschlüsseln von Dokumenten bzw. zum Überprüfen digitaler Signaturen benötigt man den öffentlichen Schlüssel des Kommunikationspartners. Ein Grundproblem einer PKI besteht darin, an den öffentlichen Schlüssel des Kommunikationspartners zu gelangen. Oft ist eine persönliche Übergabe nicht möglich und der öffentliche Schlüssel muss über ein unsicheres Kommunikationsmedium übertragen oder öffentlich bereitgestellt werden.



Bei dieser (unpersönlichen) Weitergabe des öffentlichen Schlüssels muss dann sichergestellt werden, dass der übergebene öffentliche Schlüssel tatsächlich zu der Person gehört, die sich als Eigentümer ausgibt.

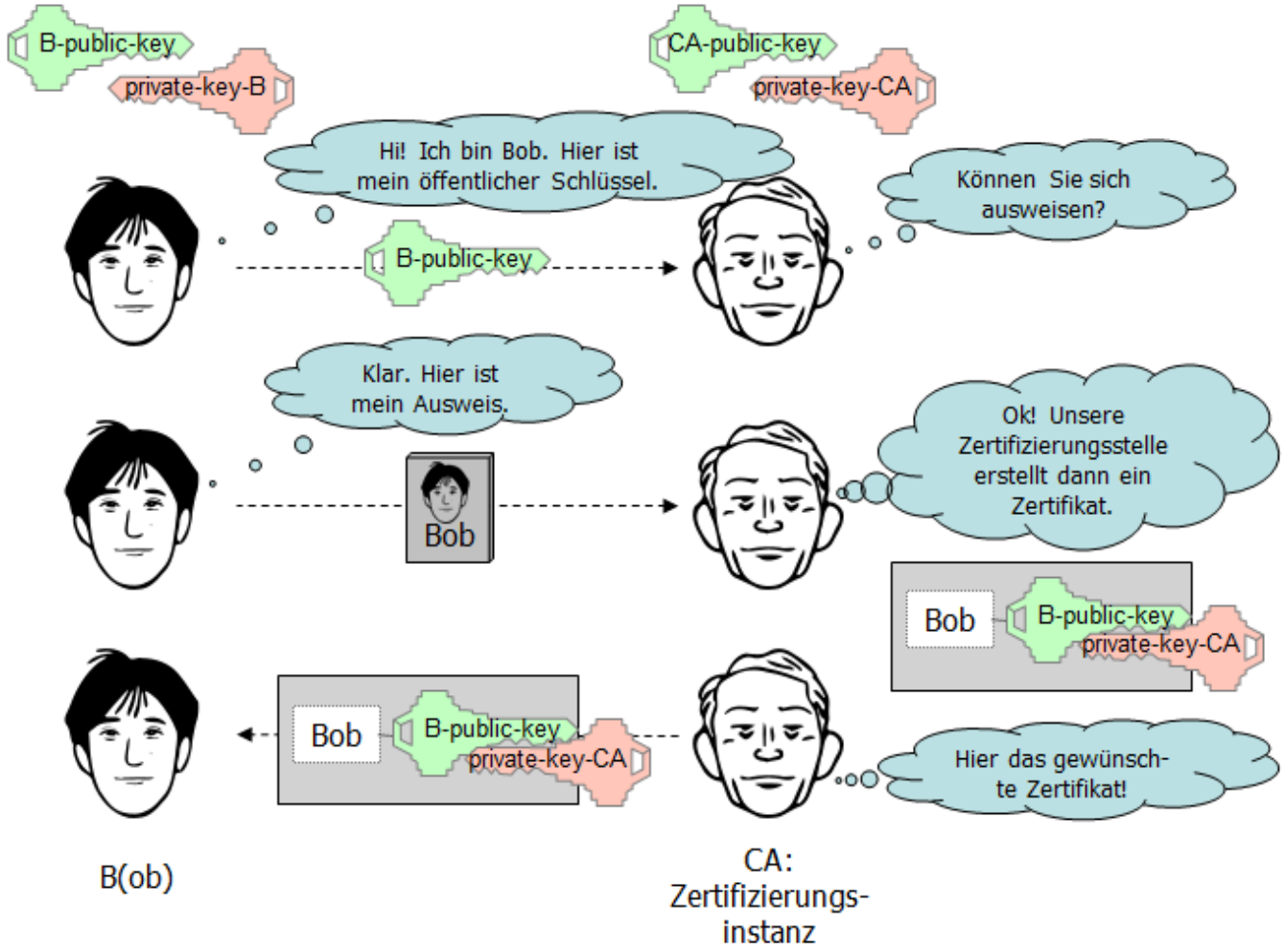
Hierzu benutzt man Zertifikate. Mit einem (Public Key) Zertifikat wird die Authentizität eines öffentlichen Schlüssels gewährleistet. Eine vertauenswürdige Instanz die **Zertifizierungsinstanz (CA - Certificate authority)** - beglaubigt mit ihrer digitalen Signatur, dass der öffentliche Schlüssel tatsächlich zur Person gehört, die sich als Eigentümer ausgibt.



Der Empfänger eines Zertifikats hat jetzt die Möglichkeit, den signierten Schlüssel (mit Angabe des Eigentümers) zu überprüfen.

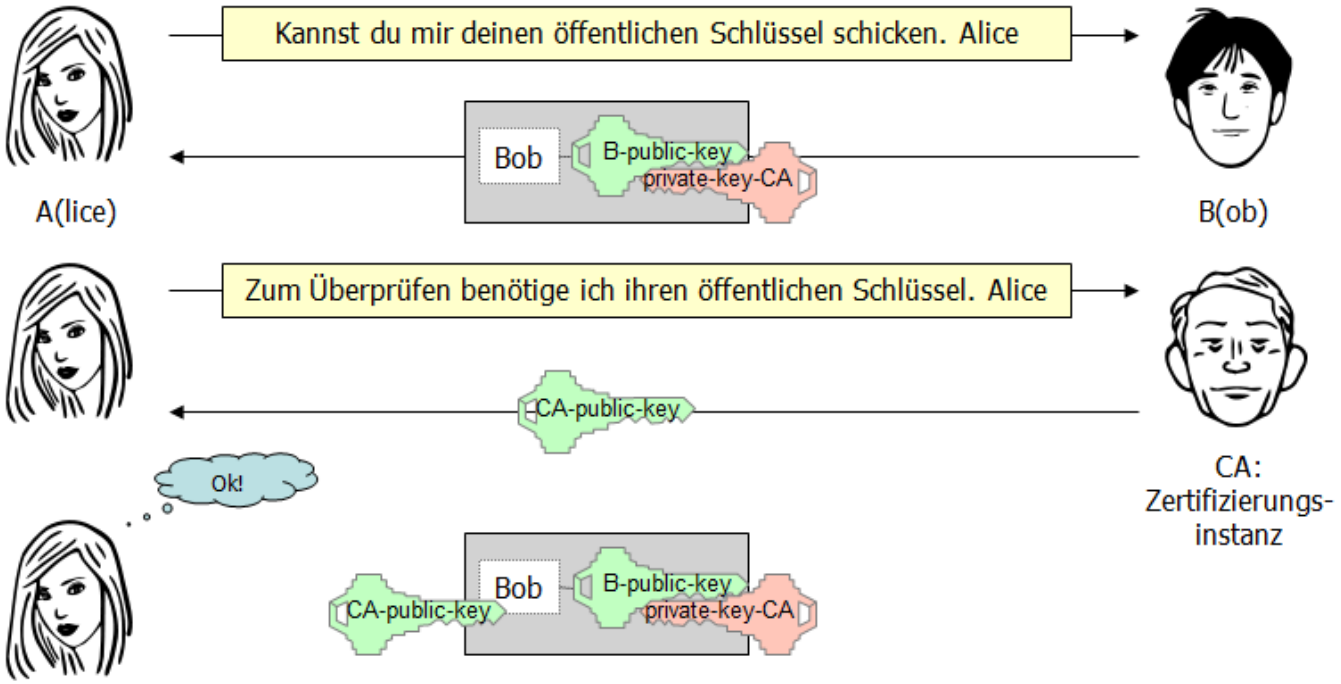
Ausstellung von Zertifikaten

Zertifikate werden von sogenannten Zertifizierungsstellen (engl. certification authority, kurz CA) ausgestellt. Das sind vertrauenswürdige Organisationen, die gegen eine Gebühr vorgelegte öffentliche Schlüssel überprüfen und signieren. Diese Zertifizierungsstellen müssen strenge Auflagen erfüllen und werden in Österreich von der Rundfunk & Telekom Regulierungs GmbH, kurz RTR überwacht.

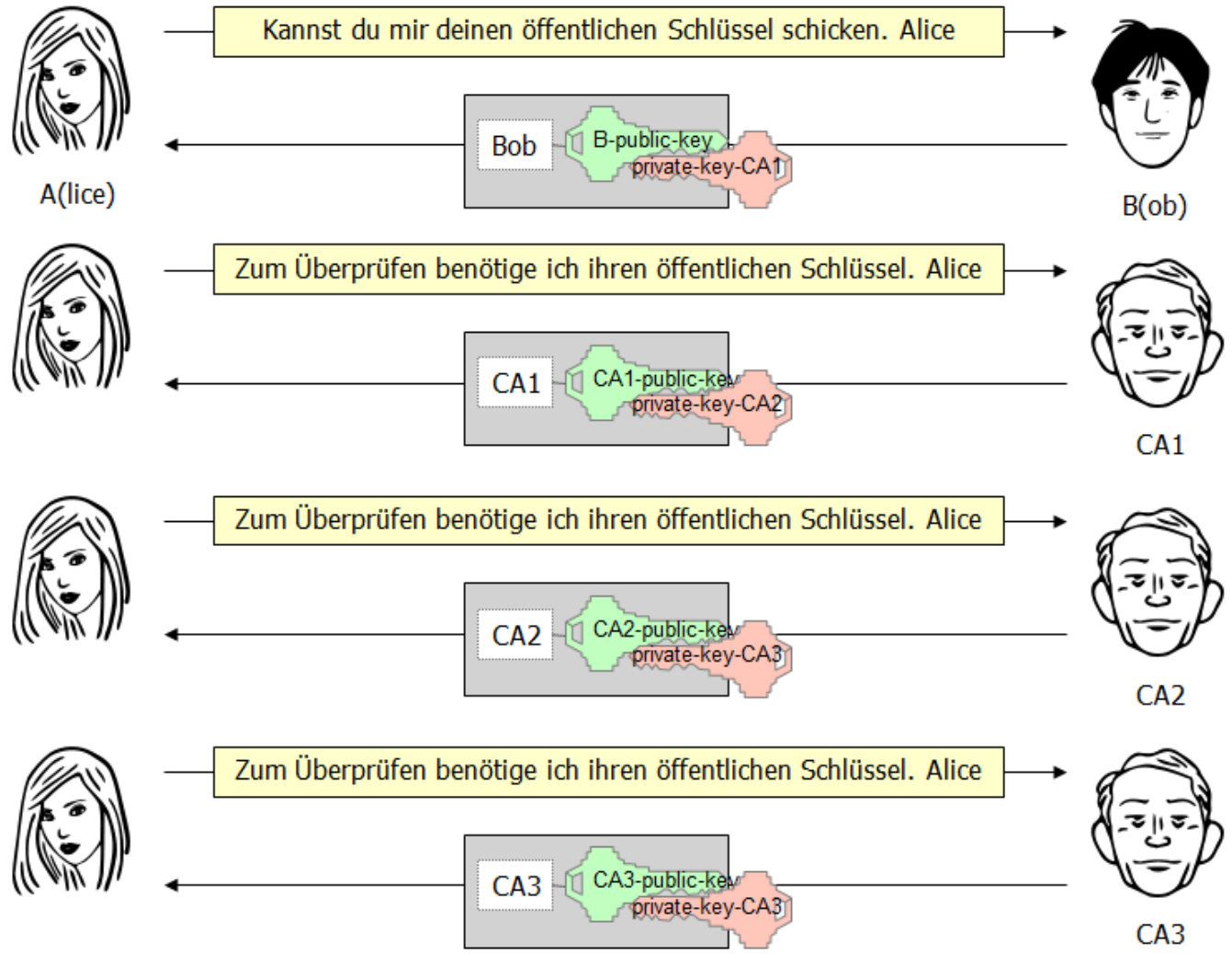


Überprüfung von Zertifikaten

Zertifikate werden überprüft, indem man die digitale Signatur der Zertifizierungsstelle überprüft. Hierzu benötigt man den öffentlichen Schlüssel der Zertifizierungsstelle.

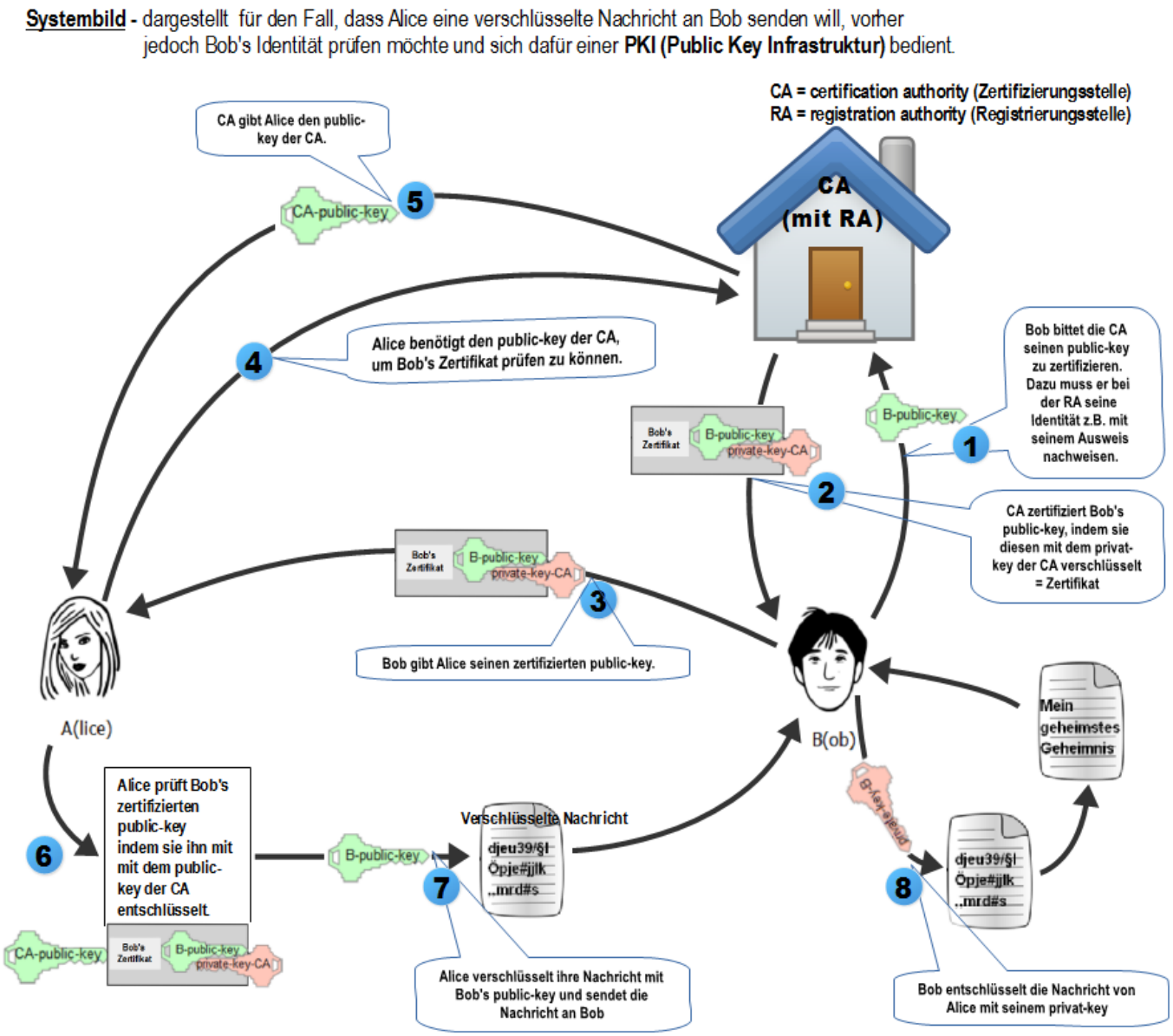


Nur, wer garantiert, dass der öffentliche Schlüssel der Zertifizierungsstelle tatsächlich zur Zertifizierungsstelle gehört? Das macht man natürlich wieder mit Zertifikaten.



Es ergibt sich eine ganze Hierarchie von Zertifizierungsstellen, die sich schrittweise Zertifikate ausstellen. Damit die Kette irgendwann zu einem Ende kommt, gibt es oberste Zertifizierungsstellen, die sogenannte Wurzelzertifikate ausstellen. Das sind Zertifikate, bei denen die Zertifizierungsstelle sich selbst das Vertrauen ausstellt.

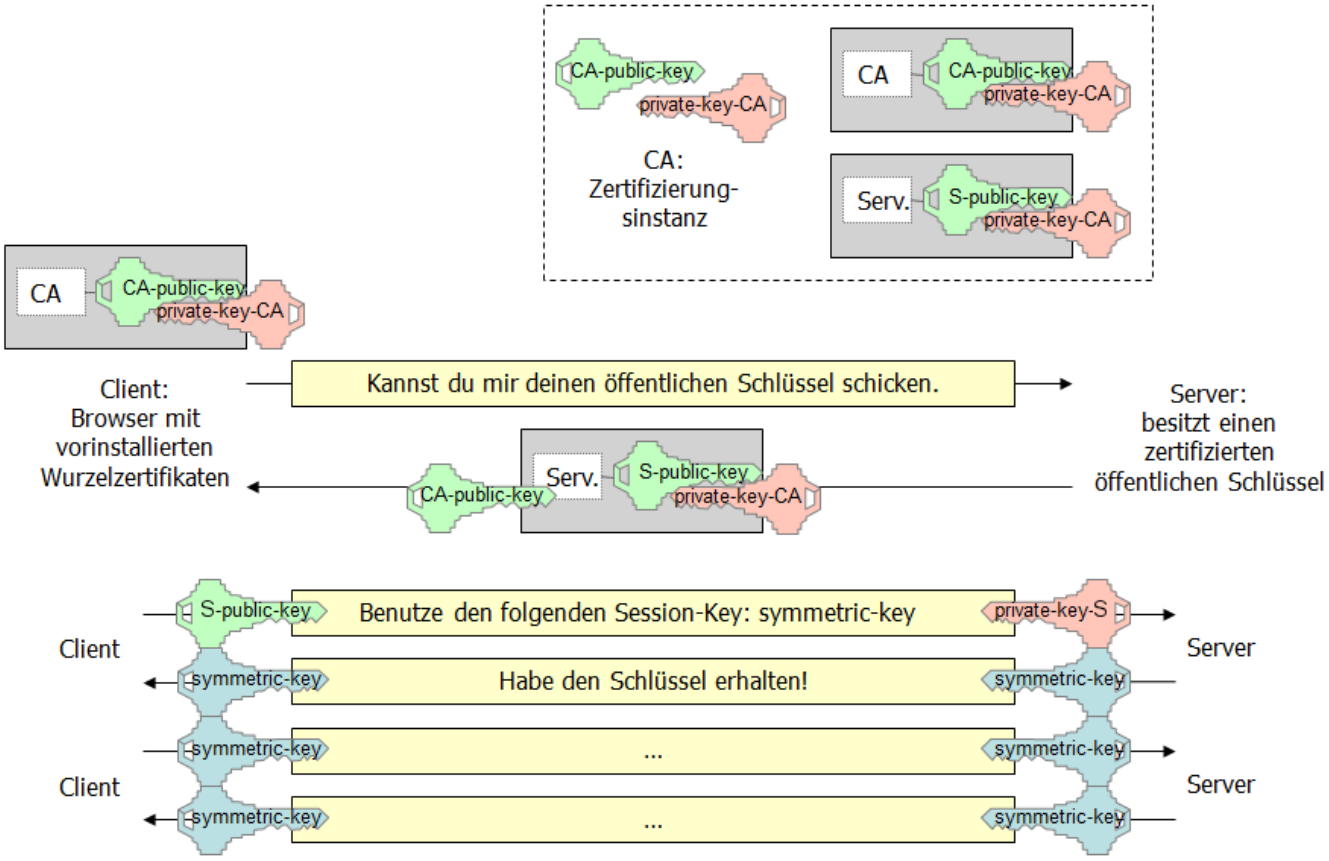
In der folgenden Abbildung wird das Zusammenspiel aller Beteiligten nochmal verdeutlicht.



Übertragung von Webseiten

HTTPS (Abkürzung für HyperText Transfer Protocol Secure) ist eine Protokoll zur sicheren Übertragung von Webseiten im Internet.

Die folgende Abbildung zeigt (in vereinfachter Form), wie Daten bei diesem Protokoll übertragen werden.



Beim Verbindungsaufbau sendet der Server ein Server-Zertifikat (mit seinem öffentlichen Schlüssel) an den Client (Browser).

Der Browser überprüft das übermittelte Server-Zertifikat - in der Regel mit einem vorinstallierten Wurzelzertifikat. Der Browser hat also vorinstalliertes Vertrauen in bestimmte Zertifizierungsstellen.

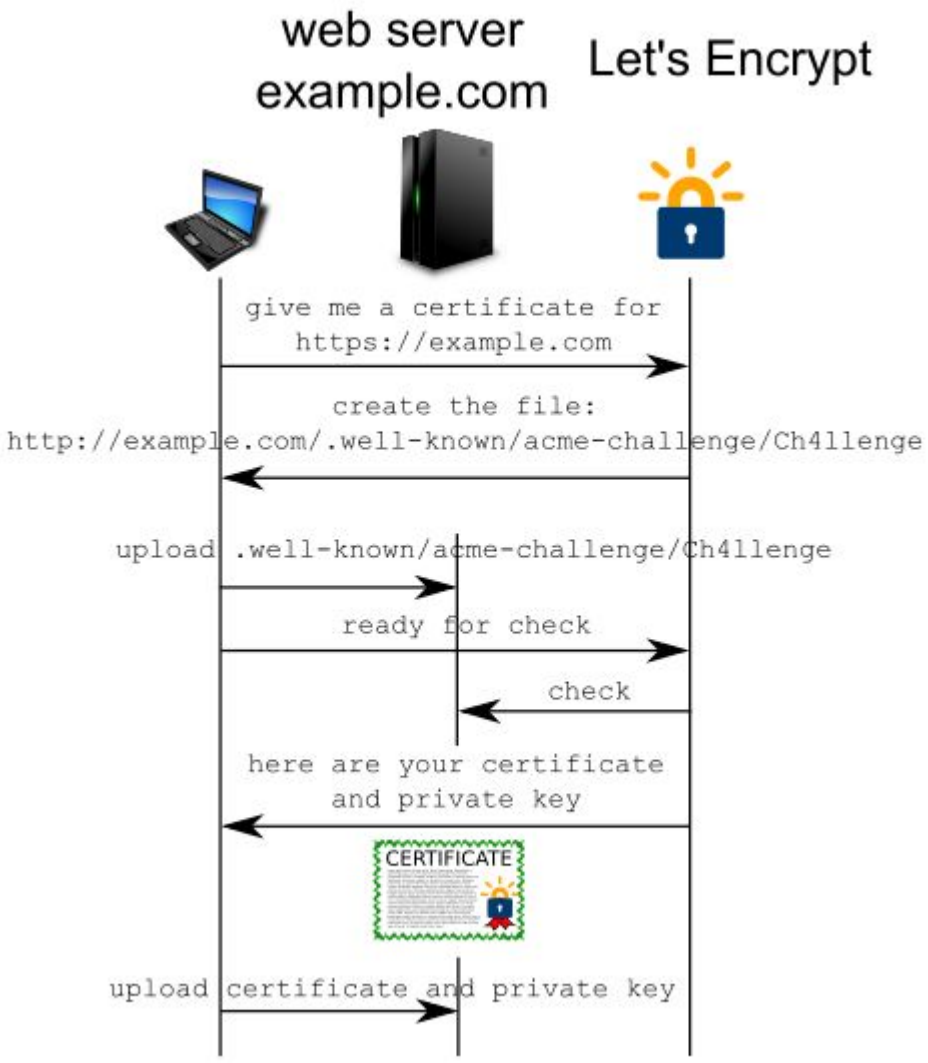
Wenn die Überprüfung zu einem positiven Ergebnis gekommen ist, dann erzeugt der Browser einen Session-Key und übermittelt ihn - natürlich verschlüsselt - an den Server. Nachdem der Server den Erhalt bestätigt hat, kann der sichere Nachrichtenaustausch beginnen.

Let's encrypt

Für die Verschlüsselung des Datenverkehrs zwischen Webbrowser und Webserver werden normalerweise teure Zertifikate benötigt. Seit Ende 2015 gibt es mit Let's Encrypt ein Projekt, das kostenlose Zertifikate vergibt. Während des Zertifizierungsprozesses muss lediglich nachgewiesen werden, dass man Zugriff auf den Webserver hat und dort zum Beispiel Dateien speichern darf. Ein

kleiner Nachteil dieses vereinfachten Zertifizierungsprozesses ist, dass unklar bleibt, welche Person oder welches Unternehmen das Zertifikat beantragt hat und wer hinter einer Webseite steckt. Trotzdem ist es aus meiner Sicht sinnvoll den Datenverkehr überhaupt erst einmal zu verschlüsseln, so dass der Datenverkehr nicht so leicht abgehört oder manipuliert werden kann. Außerdem bewertet die Suchmaschine Google eine Webseite mit HTTPS in seinen Trefferlisten etwas besser - als Anreiz zum Umstieg. Eine verschlüsselte Verbindung wird im Webbrowser Firefox links neben der Adressleiste mit einem grünen Schloss symbolisiert.

Während des Zertifizierungsprozesses müssen auf dem Webserver Dateien als „Challenge“ hochgeladen werden, um nachzuweisen, dass das Zertifikat zum Webserver passt.



Let's Encrypt ist eine Zertifizierungsstelle welche durch Spenden finanziert wird. Ziel des Projekts ist es, das World Wide Web sicher zu machen. Um die Privatsphäre der User zu wahren, sollen verschlüsselte Verbindungen zum Normalfall werden - und das so einfach wie nur möglich.

Passwörter

Was Sie sich immer schon gefragt haben:

- Was sind gute Passwörter?
- Wie merke ich mir meine Passwörter?
- Wie werden Passwörter sicher auf Computern gespeichert?
- Kann die Systemadministratorin eigentlich mein Passwort auslesen?
- Sind Fingerabdruckscanner besser als Passwörter?
- Warum sind die Passwörter beim Online-Banking eigentlich so lächerlich kurz und einfach?
- Was versteht man unter 2-Faktor-Authentifizierung?

Auf diese Fragen soll in dem folgenden Kapitel eingegangen werden.

Empfehlungen

Als Empfehlung für gute Passwörter liest man häufig, dass ein gutes Passwort

- mindestens 8 Zeichen lang sein sollte.
- aus Groß- und Kleinbuchstaben, Sonderzeichen und Ziffern bestehen sollte.
- nach einem möglichst komplexen Muster gebildet werden sollte.

Ein gutes Passwort wäre somit zum Beispiel:

f8$)?,G3

Liest man dann weiter, so folgen meist die Empfehlungen:

- Niemals für verschiedene Dienste das gleiche Passwort verwenden.
- Passwörter nicht auf Zetteln oder in Dateien auf einem Computer notieren.
- Passwörter unbedingt in regelmäßigen Abständen ändern.

Ein typischer Computernutzer kommt heute wahrscheinlich leicht auf 10-20 oder sogar noch deutlich mehr Passwörter, die er oder sie sich auf diese Weise merken müsste. Damit ist die Umsetzung der obigen Forderungen für die meisten Nutzer völlig unrealistisch.

Als Konsequenz werden dann meist:

- Viel zu einfache Passwörter verwendet, in denen dann oft auch noch leicht zu erratende persönliche Daten wie Namen oder Geburtstage vorkommen.
- Das gleiche Passwort mehrmals bei verschiedenen Diensten verwendet.
- Passwörter nie oder viel zu selten geändert.
- Passwörter in einfachen Textdateien auf unsicheren Computern gespeichert.

Ein einfacher Ausweg aus dem beschriebenen Dilemma ist es, sogenannte Passphrasen statt Passwörter zu verwenden. Im Folgenden sollen nun solche Passphrasen und deren Eigenschaften genauer betrachtet werden.

Aufgabe 1)

Wie viele mögliche Passwörter (PIN 4-stellig) gibt es bei der EC-Karte?

a) wenn man sich an gar nichts mehr erinnern kann?

Man muss immer Anzahl der Möglichkeiten pro Stelle betrachten (Variation mit Zurücklegen).

$10*10*10*10 = 10^4 = 10000$

b) wenn man noch weiß, dass unter den richtigen Ziffern genau eine 3 war?

Da die Ziffer 3 nur genau einmal vorkommt, gibt es für die verbleibenden Stellen noch 9 (statt vorher 10) mögliche Ziffern (in jeweils 4 Anordnungen), somit gilt:

4 * 9^3 = 2.916.

c) wenn man noch weiß, dass diese eine 3 definitiv an der dritten Stelle stand?

Da sicher ist, dass die 3 an der dritten Stelle stand, sucht man hier nun die Anzahl der möglichen Ziffernkombinationen für 3 aus 10 (eine Stelle ist ja bekannt und fällt somit komplett weg) im Modell der Variation ohne Zurücklegen.

Es ergibt sich: 9^3 = 729.

Wie versuchen Banken, dennoch eine gewisse Sicherheit zu garantieren?

Nach 4 falschen Versuchen wird zumeist die Karte eine Zeit lang (z.B. 2 Tage) gesperrt.

Aufgabe 2)

Wie viele achtstellige Passwörter gibt es über dem Alphabet der 95 sinnvoll über die Tastatur einzugebenden ASCII-Zeichen? Wie lange würde ein Brute-Force-Angriff im sogenannten Worst-Case dauern, wenn ein Angreifer Hundert Milliarden (100 000 000 000) Passwörter pro Sekunde testen kann?

95^8 = 6 634 204 312 890 625 / 100 000 000 000 = 66 342,04312890625s = 1105,7 min = 18,43 h

==⇒ https://www.passwortcheck.ch

Passphrasen

Unter der Annahme, dass ein Angreifer 100 Milliarden Passwörter pro Sekunde testen kann, dauert ein Brute-Force-Angriff auf ein 8-stelliges Passwort über dem Alphabet aller 95 ASCII-Zeichen im Worst-Case gerade einmal

$95^8 / 10^{11}$ s = 18,4 h.

Ein entsprechender Angriff auf ein 15-stelliges Passwort über dem Alphabet {a, ..., z} würde

$26^{15} / 10^{11}$ s = 194 127 Tage ~ 531 Jahre

dauern.

Damit sind Passwörter, die aus

- mindestens 8 Zeichen bestehen,
- Groß- und Kleinbuchstaben, alle möglichen Sonderzeichen und Ziffern beinhalten
- und nach einem möglichst zufälligen Muster gebildet werden,

für einen Menschen schwer zu memorieren und für einen Angreifer leicht zu dechiffrieren.

Im Vergleich dazu sind etwas längere Passwörter, selbst wenn sie lediglich aus den Kleinbuchstaben {a, ..., z} gebildet werden, für einen Menschen deutlich leichter zu memorieren, und für einen Angreifer sehr viel schwer zu dechiffrieren.

Relativ lange Passwörter, die aber dennoch gut für memorieren sind, werden oft als sogenannte Passphrasen bezeichnet.

Beim Erstellen solcher Passphrasen sind der Phantasie keine Grenzen gesetzt, so dass eine mögliche Passphrase zum Beispiel sein könnte:

```
nadaborder4fanta=hinreichend4heitERsicht
```

Wird gefordert, dass eine Passphrase unbedingt Sonderzeiten und Großbuchstaben enthalten muss, so lassen sich diese meist leicht in eine beabsichtigte Passphrase einbauen.

Noch besser ist natürlich, solche Passphrasen zufällig zu generieren, etwa mittels des sogenannten Diceware-Verfahrens.

Aufgabe 1)

Recherchiere im Internet, was man unter dem sogenannten Diceware-Verfahren versteht. Erzeuge mittels des Diceware-Verfahrens eine Passphrase, die aus mindestens 6 Diceware-Wörtern besteht. Verwende einen herkömmlichen Spielwürfel, um echte Zufallszahlen zu generieren.

Aufgabe 2)

Die folgende Passphrase wurde zufällig mit dem Diceware-Verfahren erzeugt:

tintenfleck-biergarten-hinauf-selber-backt-verarbeiten-parkbank-fragst-bringen-gerahmt

Beurteile die Sicherheit dieser Passphrase.

Wortliste-Deutsch

Entropie von Passwörtern

Mit der sogenannten Passwort-Entropie wird versucht, eine Aussage darüber zu treffen, wie gut oder schlecht es durch einen Angreifer vorhersagbar ist.

Die Entropie eines Passworts wird dabei von den verwendeten Zeichen bestimmt. Sie kann zum Beispiel durch die Verwendung von Groß- und Kleinbuchstaben, Ziffern sowie Sonderzeichen erweitert werden. Dazu kommt die Länge des Passworts, die eine sehr große Rolle spielt. Die Entropie eines Passworts besagt, wie schwierig es ist, es durch Erraten, Brute Force, einen Wörterbuchangriff oder andere Methoden zu knacken.

Die Entropie von Passwörtern wird meist in Form von Bits angegeben. Ein bereits bekanntes Passwort hat beispielsweise eine Entropie von 0 Bit. Ein Passwort, das in der Hälfte der Fälle direkt vorausgesagt werden kann, hat eine Entropie von 1 Bit. Die Entropie eines Passwortes kann berechnet werden, indem die Entropie für jedes verwendete Zeichen bestimmt wird. Für ein Kennwort mit einer Entropie von zum Beispiel 30 Bit werden 2 hoch 30 Versuche benötigt, um alle Möglichkeiten durchzurechnen.

Beispiel:

Die Anzahl an 20-stelligen Passphrasen über dem Alphabet A={a,b,c,...,z} der Kleinbuchstaben beträgt:

$(\#A)^{20} = 26^{20}$

Sei nun e die Länge der Bitfolge, mit der die gleiche Anzahl an Passphrasen dargestellt werden kann, also:

$2^e = 26^{20}$

$e = \log_2(26^{20}) = 20\ \log_2(26) = 20\ \log(26)/\log(2) = 94$

Damit können durch Bitfolgen der Länge 94 Bit genauso viele Passphrasen dargestellt werden wie durch 20 Stellen über dem Alphabet A der Kleinbuchstaben. Man sagt, eine 20-stellige Passphrase über dem Alphabet A der Kleinbuchstaben enthält 94 Bit Entropie.

Aufgabe 1)

Wie viel Bit Entropie enthält ein 8-stelliges Zufallswort über

a) dem Alphabet A={a,b,c,...,z} der Kleinbuchstaben?

b) dem Alphabet der ASCII-Zeichen?

Aufgabe 2)

Wie lang muss ein Passwort über dem Alphabet A={a,b,c,...,z} der Kleinbuchstaben gewählt werden, damit es genau so viel Entropie enthält wie

a) ein 8-stelliges Passwort über den Alphabet der ASCII-Zeichen?

b) ein 10-stelliges Passwort über den Alphabet der ASCII-Zeichen?

c) ein n-stelliges Passwort über den Alphabet der ASCII-Zeichen?

Aufgabe 3)

Unter der Voraussetzung, dass eine Angreiferin Kenntnis darüber hat, dass das Diceware-Verfahren verwendet wurde (und sogar auch noch die verwendete Wortliste kennt): Wie viel Bit Entropie enthält ein Zufallswort, das nach dem Diceware-Verfahren generiert worden ist? Welche Entropie ergibt sich daraus für eine Passphrase, die aus 8 Zufallswörtern gebildet wurde?

Aufgabe 4)

Wie viele Passwörter müssen mittels des Diceware-Verfahrens zu einer Passphrase zusammengesetzt werden, die mindestens 128 Bit Entropie enthalten soll?

Zufallspasswörter und Passwortmanager

Viele Menschen benötigen heute leicht 20-30 oder sogar noch mehr Passwörter, so dass es auch bei der Verwendung des Diceware-Verfahrens oft nicht mehr gelingt, sich eine so große Anzahl an Passwörtern auswendig zu merken.

Daher ist es oft nützlich, einen sogenannte Passwortmanager zu verwenden, mit dessen Hilfe man auch eine große Anzahl an Passwörtern noch effizient verwalten kann, ohne sie alle auswendig lernen zu müssen. Die einzelnen Passwörter werden dabei durch ein sogenanntes Hauptpasswort geschützt verschlüsselt auf der Festplatte gespeichert. Typischerweise kann man sich Passwörter auch gleich automatisiert generieren lassen und diese bei Bedarf mittels eines ausgeklügelten Cut-and-Paste-Systems in Formularfelder übertragen, ohne sie mühsam in Handarbeit abtippen zu müssen.

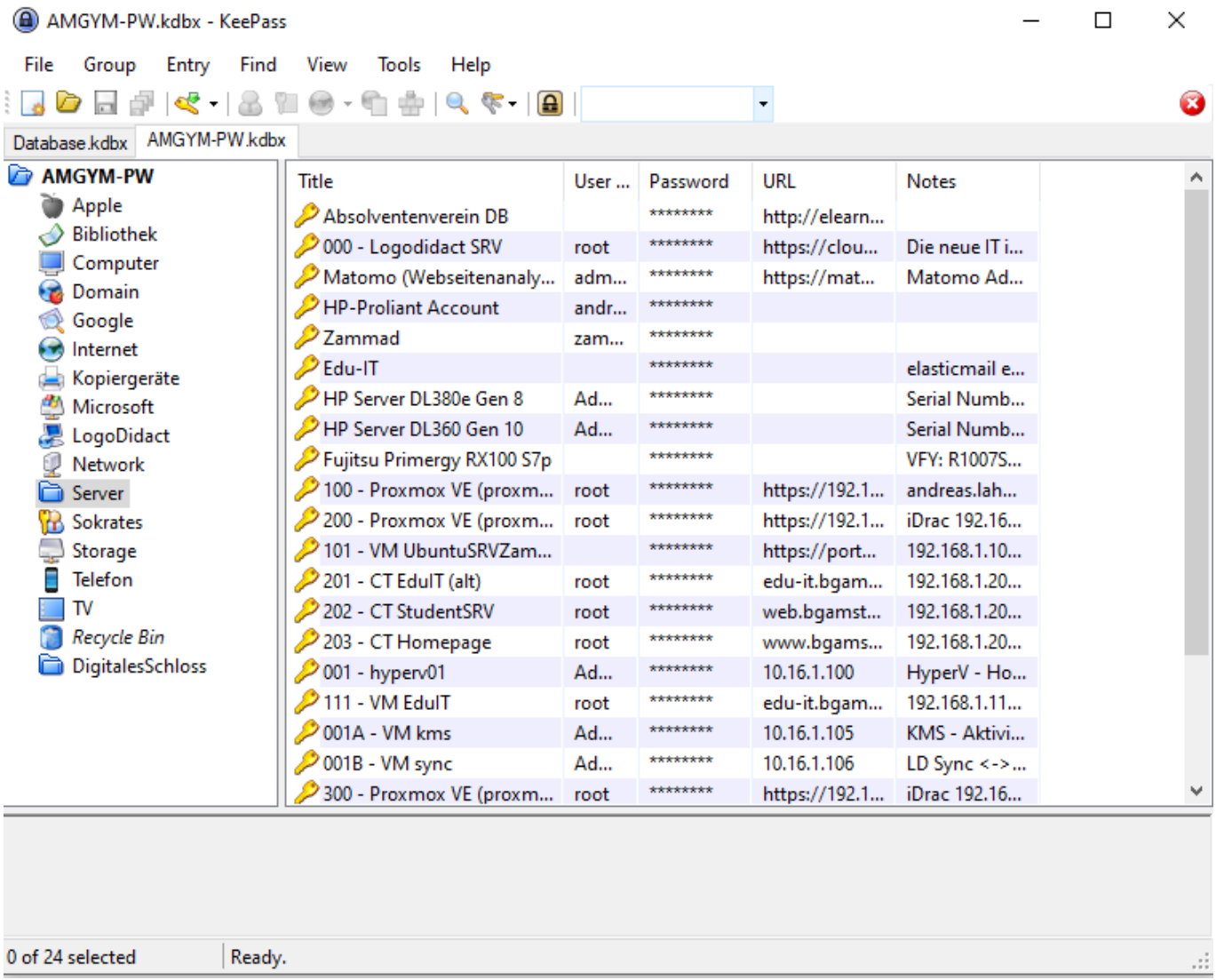
Ein weit verbreiteter Passwort-Safe ist die freie Software KeePass, die unter einer freien GPL-Lizenz für alle gängigen Betriebssysteme verfügbar ist.

Typischer Funktionsumfang eines guten Passwort-Safes in Stichworten:

- Hauptpasswort
- generieren von Pseudozufallspasswörtern
- Speichermöglichkeit von benötigten URLs und Kommentaren
- automatisches Übertragen von Passwörtern in Formularfelder per Tastenkombination
- Speicherung der Versionsgeschichte bei Passwortänderungen
- Beachtung des Kerckhoffs'schen Prinzips durch Veröffentlichung als freie Software

Bei der Auswahl eines konkreten Passwort-Safes sollte man unbedingt darauf achten, dass dieser unter einer freien Softwarelizenz veröffentlicht ist. Denn nur in diesem Fall ist sichergestellt, dass unabhängige Sicherheitsfachleute langfristig die Möglichkeit haben, Funktion und Vertrauenswürdigkeit der Software überprüfen und kontrollieren zu können. Aber Achtung: Die Möglichkeit zu einer solchen Kontrolle bedeutet noch nicht, dass diese auch in allen Fällen immer durchgeführt wurde. Man kann sich aber an den Empfehlungen von renommierten Expert:innen orientieren.

Die Sicherheit der gespeicherten Passwörtern hängt natürlich entscheidend von der Sicherheit des Hauptpasswortes ab. Deshalb sollte man als Hauptpasswort eine hinreichend lange Passphrase wählen, welche man zum Beispiel mittels des Diceware-Verfahrens generieren kann.



Speicherung von Passwort-Dateien

In Betriebssystemen oder bei Diensten im Internet müssen in irgendeiner Form Informationen über die Passwörter der angemeldeten Nutzer:innen gespeichert werden. Ein erster Ansatz dazu könnte zunächst sein, Passwörter einfach im Klartext in einer einfachen Textdatei zu speichern:

Beispiel für eine Klartext-Passwortdatei passwd.txt:

```
mustermann:qwertz
falken:joshua
administrator:passw123
root:fmvku76%d
edward:jdjhfZhd8/6ei:dk*askd_ddk?(djsh%hdZH5:dk
```

Die Benutzer:in mit dem Login „mustermann“ hat also das Passwort „qwertz“ usw.

Der Nachteil einer solchen Speicherung im Klartext ist natürlich, dass jede, die Lesezugriff auf die Passwortdatei bekommt, sofort die Passwörter von allen Nutzer:innen im Klartext auslesen kann.

Deshalb sollten von verantwortungsbewussten Systemadministrator:innen Passwörter von Nutzer:innen niemals im Klartext in Passwortdateien abgespeichert werden.

Speicherung als Hashwerte

Passwörter lassen sich auch so abspeichern, dass sie nicht im Klartext ausgelesen werden können. Dazu wird eine kryptographische Einwegfunktion, eine sogenannte Hashfunktion, verwendet. Mittels einer solchen Hashfunktion lassen sich Klartextpasswörter leicht und effizient zu Hashwerten verrechnen. Umgekehrt ist es aber praktisch unmöglich, aus einem vorgegebenen Hashwert das Klartextpasswort wieder effizient zurückzuberechnen. Da der Hashwert eines Passwortes praktisch einmalig sind, wird er anschaulich oft auch als dessen Fingerabdruck bezeichnet.

Das sogenannte SHA1-Hashverfahren ordnet einem beliebig langen Text einen 20 Byte langen hexadezimalen Hashwert zu. Für das Passwort des Benutzers „mustermann“ aus dem obigen Beispiel ergibt sich damit der folgende Hashwert:

```
sha1('qwertz') = 8c829ee6a1ac6ffdbcf8bc0ad72b73795fff34e8
```

Damit bekommt die gesamte Passwortdatei aus dem obigen Beispiel die folgende Form:

```
mustermann:8c829ee6a1ac6ffdbcf8bc0ad72b73795fff34e8
falken:d6955d9721560531274cb8f50ff595a9bd39d66f
administrator:4de730479c592c0619802013bc9883dfbde67fea
root:277c21563ade912ffa1f183f04ed482648790fed
edward:712778715c24c68886ecf69d7191cc2acec05919
```

Enthält eine Passwortdatei oder eine Datenbank statt der Klartext-Passwörter nur ihre Hashwerte, so kann eine potentielle Angreiferin mit diesen Hashwerten erst einmal nichts anfangen, weil sich

Hashwerte nicht zurückrechnen lassen. Das gleiche gilt auch für die Systemadministrator:in des Computersystems, die Passwörter nicht im Klartext auslesen kann, obwohl sie natürlich Zugriff auf die Passwortdatei hat. Beachte an dieser Stelle jedoch, dass die Nutzer:in beim späteren Einloggen nicht etwa den Hashwert, sondern ihr Klartext-Passwort in das Passwort-Eingabefeld eingibt. Anschließend wird der Hashwert des eingegebenen Passworts berechnet. Nur wenn dieser berechnete Hashwert mit dem gespeicherten übereinstimmt, wird der Zugang gewährt.

Anmerkung 1

Berechnung von SHA1-Hashwerten in einer GNU/Linux-Shell:

```
echo -n 'qwertz' | sha1sum
```

Berechnung von SHA1-Hashwerten in Python:

```
import hashlib
hashlib.sha1("qwertz".encode('utf-8')).hexdigest()
```

Anmerkung 2

Aufgrund der Entwicklung immer schnellerer Rechner entspricht das SHA1-Verfahren nicht mehr den heutigen Anforderungen an ein sicheres Hashverfahren. Es wurde im obigen Beispiel lediglich zur Illustration der prinzipiellen Funktionsweise von Passwortdateien verwendet. Aktuelle Verfahren wie SHA512 erzeugen sehr viel längeren Hashwerte, so dass die Übersichtlichkeit der obigen Beispiel-Passwortdatei (für eine menschlichen Leserin) damit erschwert wäre.

Pepper

Werden gehashte Passwörter nach dem oben beschriebenen einfachen Hashverfahren gespeichert, so ergibt sich allerdings das Problem, dass sich die Hashwerte typischer Passwörter vorherberechnen und in sogenannten Rainbowtables speichern lassen. Wird dann zu einem späteren Zeitpunkt eine gehashte Passwortdatei erbeutet, so können die dort gespeicherten Hashwerte einfach mit den vorherberechneten verglichen werden, ohne dass dazu eine erneute zeitaufwendige Berechnung nötig wäre. Findet der Angreifer eine Übereinstimmung von Hashwerten, so hat er damit auch das zugehörige Klartextpasswort gefunden.

Um solche Angriffe auf Passwortdateien mittels Rainbowtables zu erschweren, wird ein sogenanntes Pepper eingesetzt. Dazu werden die Klartextpasswörter pw um ein hinreichend lange Zeichenfolge, das sogenannte Pepper p, ergänzt. Anschließend wird auf die konkatenierte Zeichenfolge p+pw ein Hashverfahren wie z.B. SHA1 angewandt.

```
    pw = 'qwertz'
    p    = '9cm9hsgkdcucz7ckdje-z7652cv_mnbsdsj_'
    sha1(p+pw) = sha1('9cm9hsgkdcucz7ckdje-z7652cv_mnbsdsj_qwertz')
               = 5f684e914ba9840cd0a0b2af79251c476ec5ffc2
```

Mit dem Pepper p = '9cm9hsgkdcucz7ckdje-z7652cv_mnbsdsj_' bekommt die gesamte Passwortdatei aus dem obigen Beispiel damit die folgende Form. Beachte, dass für alle Passwörter das gleiche Pepper p verwendet wird.

```
    mustermann:5f684e914ba9840cd0a0b2af79251c476ec5ffc2
    falken:08eb33988697de0ad395ba94290e2a324887d0ab
    administrator:a1becea837841197fb847940653c66b84859a576
    root:162b0d49d63b2b280e313ec7ae012ce7871b578a
    edward:6d47b7c8a88bc2e633d05d8defb0b76405859044
```

Wird nun die Passwortdatei durch einen Angreifer erbeutet, so ist für diesen nicht erkennbar, dass ein Pepper verwendet wurde. Aufgrund der hohen Entropie der um das Pepper verlängerten Klartextpasswörter funktioniert auch ein Angriff über vorherberechnete Rainbowtables nicht mehr.

Eine Schwachstelle des Hashen von Passwörtern mit Pepper entsteht aber, sobald ein Angreifer zusätzlich zur Passwortdatei auch noch das Pepper p erbeutet. Denn dann kann er z.B. Wörterbuch- oder Bruteforceangriffe starten, indem er die zu testenden Passwörter einfach vor dem Hashen um das Pepper ergänzt und damit einen Rainbowtable für das erbeutete Pepper berechnet.

Ein Ausweg daraus bietet wiederum das Hashen der Passwörter mit einem sogenannten Salt:

Salt

Während beim Hashen von Passwörtern mit einem Pepper für die gesamte Passwortdatei ein festes Pepper verwendet wird, wird für Passwortdateien mit Salt für jedes Passwort ein unterschiedlicher sogenanntes Salt s verwendet.

```
    pw = 'qwertz'
    s    = 'dkkfjerfjc983883(7akfrjklfds8/akdjf=adfsh*'
    sha1(s+pw) = sha1('dkkfjerfjc983883(7akfrjklfds8/akdjf=adfsh*qwertz')
               = b1abab1a8efe6bd81df13f6f2fa3ae2aadce960d
```

Die Saltwerte werden dann jeweils zusammen mit dem Hashwert in der Passwortdatei gespeichert, so dass sich für das obige Beispiel dann eine Datei der folgenden Form ergibt:

```
mustermann:dkkfjerfjc983883(7akfrjklfds8/akdjf=adfsh*:b1abab1a8efe6bd81df13f
6f2fa3ae2aadce960d
falken:dk38d7/%&jdjdsjd9djsu;(djja8nvmnbx:a53fba661a4e031a7de12bfc55a422d8fb
fad6e2
administrator:aslkfa8e4jfsaksakfdjlalkdsfjdsa:286752d79187249ffa28d2068e460c
2c12a8f093
root:ivcjcxy8737dki9cyyxvjlkj28347askf:2e4fc670daa272fac4bf7d9c67fbdadddf558
78a
edward:aslkfj8vcx7j38h2983hfnc832fjc:55f2ee3ce9296cb9fbaa49581c998c26cbb0c6c
e
```

Da für alle Passwörter unterschiedliche Saltwerte verwendet werden, müsste bei einem Rainbowtable-Angriff für jedes Passwort ein eigener kompletter Rainbowtable berechnet werden. Dies ist so aufwendig, dass ein solcher Angriff nicht effizient möglich ist.

Zwei-Faktor-Authentifizierung (2FA)

Die Zwei-Faktor-Authentifizierung (auch: Zwei-Schritte- oder Zwei-Wege-Authentifizierung bzw. 2FA) ist eine zusätzliche Sicherheitsmaßnahme zum Schutz von Benutzerkonten: Selbst wenn Ihre Passwörter in die falschen Hände gelangen, haben Unbefugte auf diese Weise keinen Zugriff auf Ihre Accounts.

Zusätzlich zum Passwort müssen Sie beim Login eine weitere Sicherheitskomponente eingeben, etwa einen PIN-Code. Dieser Code wird z. B. an die in Ihrem Konto hinterlegte Handynummer gesendet.

Eine weitere Möglichkeit sind für kurze Zeit gültige Einmalkennwörter, die von sogenannten Authentifizierungs-Apps erstellt werden. Solche Apps bieten den Vorteil, dass Sie damit Ihre mit 2FA geschützten Konten übersichtlich verwalten können.

Bekannte Apps für die Zwei-Schritte-Authentifizierung sind z. B. Google Authenticator (für iOS/Android), Microsoft Authenticator (für iOS/Android) oder Twilio Authy (für iOS/Android/Desktop).

Die Zwei-Faktor-Authentifizierung wird inzwischen von den meisten großen Online-Diensten unterstützt – von Apple über Google oder Microsoft bis hin zu Twitter, Facebook und Instagram.

bsifb_animation_zwei-faktor-authentisierung.mp4

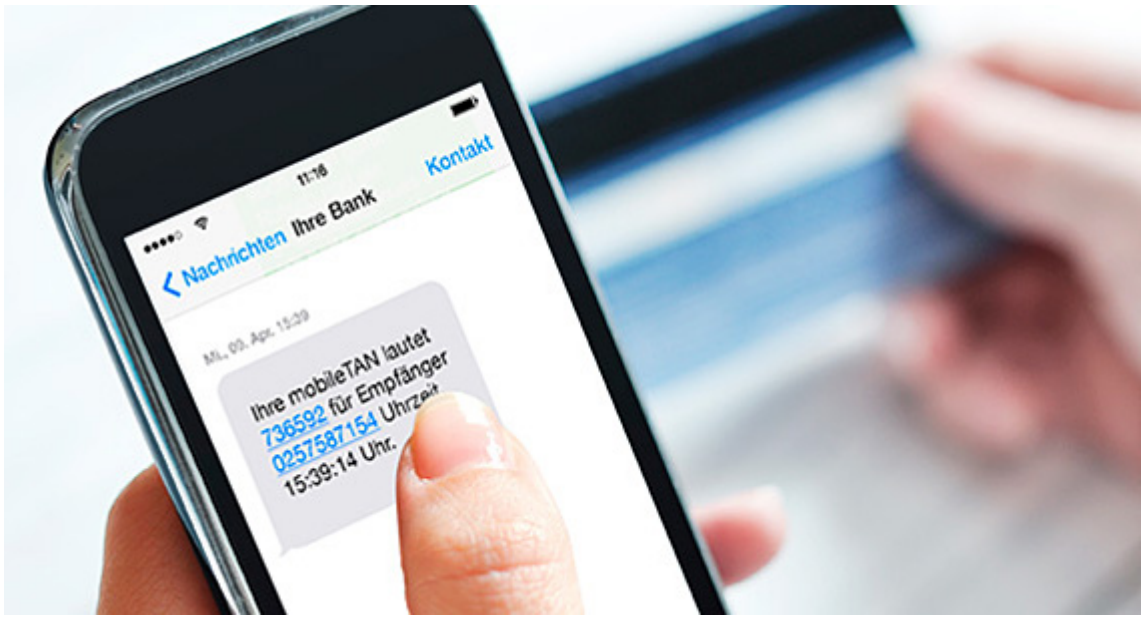
Grundsätzlich gibt es drei unterschiedliche Möglichkeiten, um sich auszuweisen:

- durch Wissen (z. B. Passwort)
- durch Besitz (z. B. Bankomatkarte, Token)
- durch biometrische Merkmale (z. B. Fingerabdruck)

„Token“ ist ein Überbegriff für jegliche Technologien, mit deren Hilfe sich eine Person authentisieren kann. Hierzu zählen sowohl Hardware-Tokens als auch Software-Tokens. Hardware-Tokens sind beispielsweise Smartcards, USB-Tokens und RFID-Chips, mit denen Einmalpasswörter generiert werden, oder auch Smartphones, an die ein Authentifikationsfaktor übertragen wird. Software-Tokens benötigen keine zusätzliche Hardware – hier werden via Software One-Time-Passwords (OTPs) generiert. Solche Einmalpasswörter sind Passwörter, die ein einziges Mal verwendet werden können und danach ihre Gültigkeit verlieren.

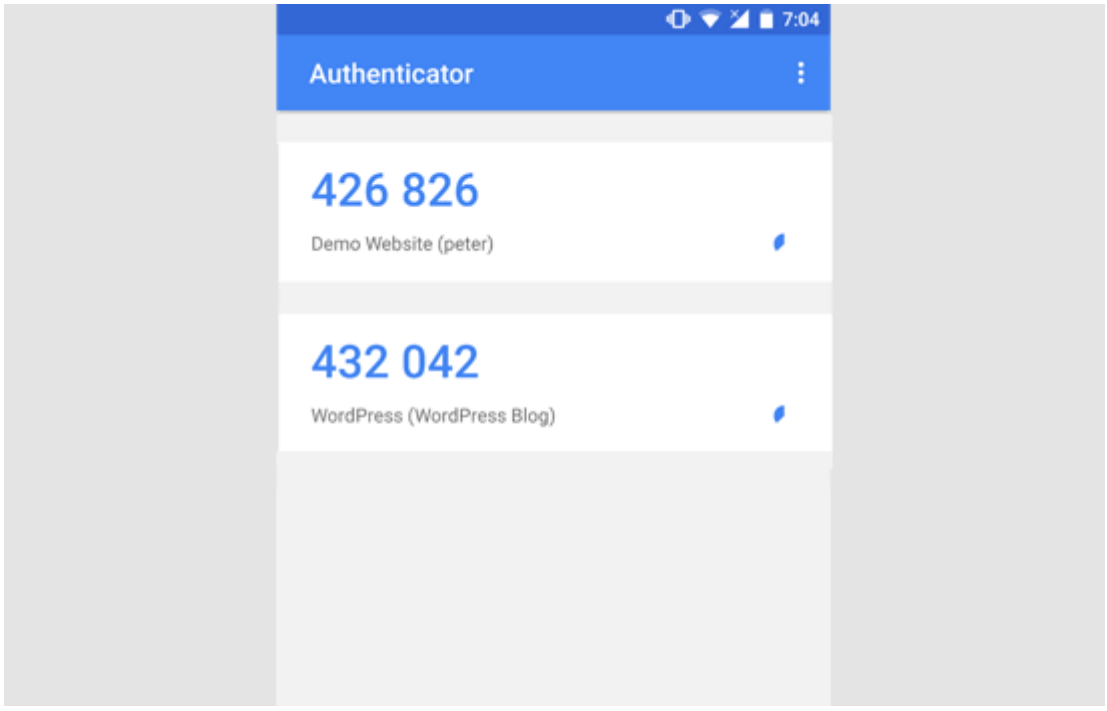
Eine bekannte Zwei-Faktor-Authentifizierung ist das Zusammenspiel von **Bankomatkarte** (der Besitz) und die dazugehörige **PIN** (das Wissen). Um Bargeld beheben zu können, werden beide Bestandteile benötigt. Das Gleiche gilt für Online-Überweisungen, bei denen die Anmeldedaten (das Wissen) und – zur Freigabe einer Überweisung – die TAN (der Besitz) vorliegen müssen. Früher funktionierte dies mit einer im Voraus generierten **TAN-Liste**, welche von der Bank postalisch der Kundin oder dem Kunden zugestellt wurde. Dabei war jede TAN ein einziges Mal gültig. Heutzutage sind die meisten Banken von Papier-TANs auf ihre digitalen Pendants umgestiegen: per SMS (je nach Bank wird dieses Verfahren unterschiedlich benannt: **smsTAN**, **mobileTAN**, **TAC-SMS**), TAN-Generator oder digitaler Signatur. Ein TAN-Generator ist ein selbstständiges Gerät, in das eine EC-Karte gesteckt wird und nach der Eingabe eines PIN-Codes eine TAN generiert wird. So ein TAN-Generator hat durch die zusätzliche Einbeziehung der EC-Karte und der fehlenden Vernetzung eine hohe Sicherheit. Nichtsdestotrotz sind smsTAN wegen ihrer einfachen Bedienung und schnellen Verfügbarkeit bei Kundinnen und Kunden am beliebtesten, da kein weiteres Gerät – außer dem in der Regel bereits vorhandenen Mobiltelefon – benötigt wird. So eine smsTAN hat einen

Gültigkeitszeitraum von typischerweise nur einigen wenigen Minuten, was einen Missbrauch stark einschränkt.



Die Verwendung eines Smartphones als Empfänger für OTPs wird auch als tokenlose Authentifizierung bezeichnet, da kein extra Hardware-Token von Anbietern zur Verfügung gestellt werden muss, sondern das Smartphone der Kundin oder des Kunden als Empfangs- oder Anzeigegerät für Tokens dient.

Die tokenlose Authentifizierung etabliert sich mittlerweile auch außerhalb der Bankenbranche und unternehmensinternen Systemen. Google unterstützt beispielsweise eine Zwei-Faktor-Authentifizierung als Anmeldeverfahren für seine Dienste, bei dem das Smartphone verwendet wird, um OTPs zu empfangen. Dies bedeutet, dass entweder ein SMS-Dienst verwendet wird, der bei einem Anmeldungsversuch in das Google-Konto ein OTP per SMS sendet, oder alternativ dazu eine von Google bereitgestellte App (Google Authenticator) verwendet wird, um gültige OTPs anzuzeigen. Die App-Lösung ist nicht auf eine Mobilfunkverbindung angewiesen und bietet daher mehr Flexibilität.

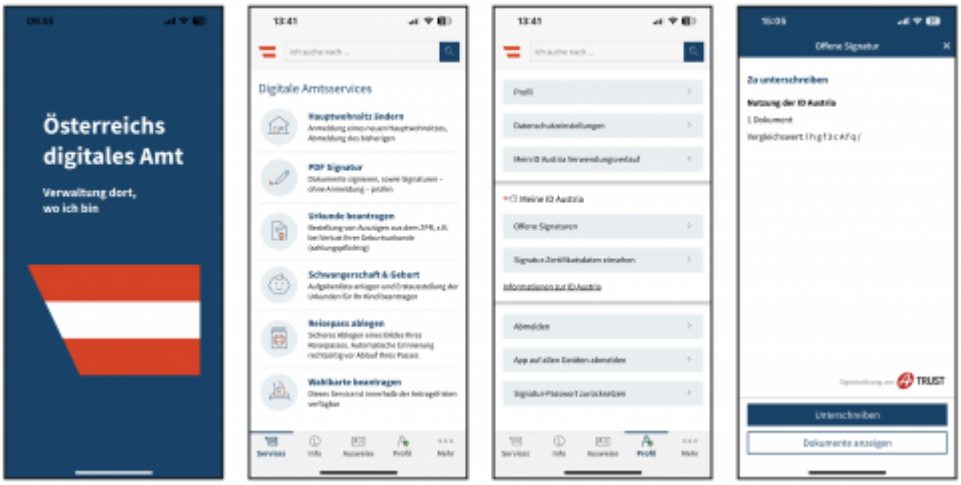


Das System der tokenlosen Authentifizierung mit dem Mobiltelefon erhöht jedoch nur dann die Sicherheit, wenn das Mobiltelefon nicht auch zur Anmeldung auf dem Konto verwendet wird, für welches das OTP angefordert wird. Der Sinn des Tokens ist es einen zweiten, unabhängigen Authentifizierungsfaktor zu liefern, der über einen gesonderten Kanal übertragen oder generiert wird. Falls aber zum Beispiel das Google-Konto über das Smartphone aufgerufen wird und das OTP für die Anmeldung ebenfalls auf das Smartphone gesendet wird, ist die Risikominimierung, die durch Verwendung der Zwei-Faktor-Authentifizierung entstehen sollte, nicht mehr gegeben, da beide Authentifizierungsschritte auf demselben Gerät stattfinden und somit eine mögliche Angriffsstelle für Kriminelle gegeben ist.

Jede Art des Tokens hat seine Vor- und Nachteile, daher ist es wichtig, einen gewissen Grad an Flexibilität zwischen den verschieden Methoden zu erlauben. Der Nachteil bei der Verwendung eines Smartphones ist, dass es zum Zeitpunkt der Authentifikation aufgeladen und verfügbar sein muss, da ansonsten kein OTP zugesendet oder angezeigt werden kann. Zu bedenken ist auch, dass es bei Verlust, Diebstahl oder Defekt des Smartphones nicht möglich sein wird, sich ohne größeren Aufwand in ein durch Zwei-Faktor-Authentifizierung gesichertes Konto einzuloggen.

In Österreich besteht eine weitere Möglichkeit der Zwei-Faktor-Authentifizierung. Im Zuge der Forcierung des e-Governments wurde mit der Bürgerkarte als Token eine elektronische Möglichkeit geschaffen, sich amtlich auszuweisen. Dies ist auch im österreichischen E-Government-Gesetz verankert. Bei der Bürgerkarte fallen keine zusätzlichen Kosten an, da die Funktion der Bürgerkarte auf der bereits verbreiteten „e-card“ lediglich aktiviert werden muss.

Auch an eine tokenlose Authentifizierung wurde gedacht und mit der „Handy-Signatur“ geschaffen. Das Mobiltelefon stellt dabei den Faktor des Besitzes dar und empfängt den Einmalcode (OTP), der fünf Minuten gültig ist. Ein weiterer Vorteil in Österreich: die Handy-Signatur und die Bürgerkarte können für eine rechtsgültige elektronische Unterschrift genutzt werden, diese ist durch das Signaturgesetz der eigenhändigen Unterschrift gleichgestellt.



Fazit

Die Zwei-Faktor-Authentifizierung bietet durch die Kombination mehrerer Authentifizierungsschritte eine höhere Sicherheit für Konten. Jede Nutzerin und jeder Nutzer muss für sich entscheiden, ob der Mehraufwand, der mit der Zwei-Faktor-Authentifizierung verbunden ist, für das jeweilige Konto sinnvoll ist. Falls eine tokenlose Authentifizierung verwendet wird, sollten Nutzerinnen und Nutzer unbedingt darauf achten, dass die OTPs nicht auf das Gerät geschickt werden, welches für die Anmeldung oder Überweisung verwendet wird.

Mögliche Fallstricke

Im Folgenden werden einige Beispiele für mögliche Fallstricke aufgezählt, durch die man schnell wieder die Sicherheit eines eigentlich starken Passwortes gefährden kann. Die Aufzählung ist beispielhaft und erhebt keinen Anspruch auf Vollständigkeit.

Überall gleiches Passwort

Verwendet man dasselbe Passwort mehrfach für unterschiedliche Dienste im Internet, so kann eine Angreifer, der dieses Passwort bei einem dieser Dienste erbeutet, sofort auch auf alle anderen zugreifen. Insbesondere hat man als Benutzer keinerlei Kontrolle darüber, ob ein Diensteanbieter mit den Passwörtern seiner Kunden auch tatsächlich verantwortlich umgeht und die Passwörter beispielsweise ausschließlich verschlüsselt speichert. In der Vergangenheit ist es leider recht häufig vorgekommen, dass Einbrecher in Computersysteme völlig unverschlüsselte Dateien oder Datenbanken mit den Passwörtern einer erschreckend großen Anzahl an Benutzern erbeutet haben. Aus Angst vor Reputationsverlust verschweigen Diensteanbieter solche Einbrüche oft über lange Zeit, so dass die Nutzer des Dienstes nicht einmal bemerken, dass ihr Passwort schon längst kompromittiert worden ist. Hinzu kommt, dass ein verantwortungsvoller und sicherheitsbewusster Umgang mit den Kundenpasswörtern oft sehr viel aufwendiger und kostenintensiver für den Diensteanbieter ist als eine einfache unverschlüsselte Speicherung. Daher sollte man unbedingt für alle Dienste im Internet unterschiedliche Passwörter verwenden. Zur effizienten Verwaltung der Passwörter kann man, wie oben beschrieben, ein Passwortspeicher-Programm einsetzten.

Die Sicherheitsfrage kann schnell wieder alles kaputt machen

Bei vielen Diensten im Internet kann man beim Einrichten eine Antwort auf eine sogenannte Sicherheitsfrage angeben, mit deren Hilfe man später ein vergessenes Passwort wieder zurücksetzen kann. Als Sicherheitsfrage wird dann oft zum Beispiel nach dem Mädchenname der eigenen Mutter oder nach dem Namen des ersten Haustieres gefragt. Da solche Antworten aber relativ leicht zu erraten sind, wird die Sicherheit des eigentlichen Passwortes durch solche Sicherheitsfragen schnell ad absurdum geführt. Ein möglicher Ausweg ist es, als Antwort auf die Sicherheitsfrage einfach auch wieder eine starke Zufallspassphrase hinreichender Länge zu verwenden. Diese Sicherheits-Passphrase kann man dann getrennt von dem eigentlichen Passort wiederum in einem Passwortspeicher oder auch ausgedruckt an einem sicheren Ort verwahren.

Testen von Passwörtern im Internet

Im Internet findet man oft kostenlose Seiten, auf denen man angeblich die Sicherheit von Passwörtern testen kann. Dazu soll man das Passwort auf der Seite eintippen, worauf dann meist eine Zeit berechnet wird, die angeben soll, wie lange Brute-Force-Angriff auf das Passwort dauern würde. Diese Zeit hängt dann oft lediglich von der Stellenzahl des eingegebenen Passwortes ab, so dass es völlig ausreichen würde, statt nach dem Passwort lediglich nach dessen Stellenzahl zu fragen. Da man als

Nutzer keinerlei Kontrolle darüber hat, ob der Anbieter einer solchen Seiten nicht vielleicht doch die eingegebenen Passwörter speichert, um sie in Rainbowtables zu verwenden oder um sie weiterzuverkaufen, sollte man kritisch hinterfragen, ob es tatsächlich sinnvoll ist, seine Passwörter solchen Seiten anzuvertrauen. Insbesondere sollte man auch kritisch hinterfragen, wie der Seitenbetreiber sein Angebot wohl finanziert und warum er es kostenlos im Internet anbietet.

Auswahl eines Passwortspeicher-Programms

Da in Passwortspeicher-Programmen besonders sensible Daten abgelegt werden, sollte man sorgfältig auswählen, welches Programm man dazu verwendet. Da bei proprietären Programmen der Quelltext in der Regel nicht veröffentlicht wird, ist hier das Kerckhoffs'sche Prinzip meist verletzt, sodass es keinerlei Möglichkeit gibt, die Vertrauenswürdigkeit solcher proprietärer Programme zu prüfen. Bei freier Software gibt es zwar eine solche Überprüfungsmöglichkeit. Allerdings ist dies alleine noch keine Garantie dafür, dass auch tatsächlich jemand diese Möglichkeit wahrgenommen und die Software ausgiebig getestet oder sogar vollständig auditiert hat. Eine gute Möglichkeit sich zu informieren, welche Programme in der IT-Security-Community als vertrauenswürdig gelten, bietet das https://www.kuketz-blog.de.

Internetbrowser bieten in der Regel auch die Möglichkeit, Passwörter verschlüsselt zu speichern. Allerdings sind Browser relativ komplexe und sehr vielseitige Programme, so dass sie im Vergleich zu einem auf Sicherheit optimierten Passwortspeicher-Programm auch fehleranfälliger sind. Hinzu kommt, dass Internet-Browser bei Angriffen aus dem Internet anders als externe Passwortspeicher auch oft das erste und damit einfachste Angriffsziel darstellen. Daher sollte man zumindest für wertvolle Passwörter tendenziell eher einen externen Passwortspeicher bevorzugen.

IOT (Internet of Things)

Ebenso wie im Internet sollte man auch zu Hause für verschiedene Dienste und Geräte stets unterschiedliche Passwörter verwenden. Besonders gefährdet für Angriffe sind billig hergestellte IOT-Geräte, da die Hersteller aus Kostengründen nach dem Verkauf teilweise nur mit großer zeitlicher Verzögerung oder sogar überhaupt keine Updatemöglichkeiten bei auftretenden Sicherheitslücken anbieten.

Fingerabdruck-Scanner

Auch durch die Verwendung eines Fingerabdruckscanners als alternative Authentifizierungsmöglichkeit kann man leicht die Sicherheit eines starken Passwortes wieder gefährden. Denn im Gegensatz zu Passwörtern lassen sich Fingerabdrücke kaum geheim halten und das Wechseln eines Fingerabdrucks bei Verdacht auf Missbrauch ist sogar überhaupt nicht möglich. Hinzu kommt, dass gerade die zu schützenden Geräte meist übersät sind von Fingerabdrücken, die eine potentieller Angreifer nutzen kann, um sich unbefugten Zugang zu verschaffen. Es sind Fälle bekannt, bei denen es Angreifern gelungen ist, Fingerabdruckscanner zu kompromittieren, indem sie einfach einen auf Folie ausgedruckten Fingerabdruck oder einen 3D-Druck eines Fingerabdrucks verwendet haben.

Online-Banking

Um zusätzliche Sicherheit zu gewinnen, werden beim Online-Banking sogenannte TANs (Transaction Numbers) verwendet. Dabei werden verschiedene Verfahren unterschieden.

PIN-TAN-Verfahren

Der Benutzer erhält eine Papierliste mit TANs, die er bei Bedarf eingeben kann. Da bei diesem Verfahren sowohl PIN (Personal Identification Number) als auch TAN in den Browser eingegeben werden, ist bei einem Man-in-the-Browser-Angriff gleich auch das PIN-TAN-Verfahren mit betroffen. Das PIN-TAN-Verfahren gilt daher allgemein als relativ unsicher, so dass es viele Banken heute nicht mehr verwenden.

Mobile-TAN-Verfahren

Die Benutzerin erhält bei Bedarf eine TAN auf ihr Mobiltelefon gesendet. Diese TAN tippt sie dann in den Browser mit ihrer Online-Banking-Sitzung ein. Hierbei sollte man darauf achten, unterschiedliche Geräte für das Online-Banking und den TAN-Empfang zu verwenden, da man ansonsten schnell die zunächst gewonnene zusätzliche Sicherheit wieder gefährdet, da ein Angreifer dann doch wieder nur ein Gerät kompromittieren müsste. Auch ist zu bedenken, dass sich ein Mobiltelefon nur schwer so einrichten lässt, dass es hohen Sicherheitsanforderungen genügt.

Chip-TAN-Verfahren

Die TAN wird mit einem externen Gerät, dem sogenannten TAN-Generator, erzeugt, in das man die EC-Karte einsteckt. Die Überweisungsdaten werden mittels eines sogenannten Flickercodes über den Bildschirm auf den TAN-Generator übertragen. Die generierte TAN ist dann ausschließlich für die eingegebenen IBAN und den eingegebenen Betrag gültig. Zur Sicherheit werden IBAN und Betrag noch einmal auf dem externen Gerät angezeigt und sollten von dort (und nicht von der Anzeige des Browsers!) noch einmal kontrolliert werden. Dieses Verfahren gilt allgemein als relativ sicher, da der externe TAN-Generator keine Verbindung zum Internet aufbaut, so dass sich ein potentieller Angreifer physikalischen Zugang zu dem Gerät und zu der EC-Karte verschaffen müsste.

2-Faktor-Authentifizierung

Wird hierbei nur ein einziges Gerät für beide Authentifizierungen verwendet, so gefährdet man wiederum die zunächst eigentlich durch den zweiten Faktor gewonnene Sicherheit, da ein Angreifer dann nur ein einziges Gerät kompromittieren müsste. Daher sollte man darauf achten, dass bei dem zweiten Faktor ein unterschiedliches Gerät involviert ist, so dass ein potentieller Angreifer dann auch tatsächlich die Kontrolle über beide beteiligten Geräte erlangen müsste.

Vertrauenswürdiges Computersystem

Passwörter sollten nur auf vertrauenswürdigen Computersystemen eingegeben werden. Insbesondere sollte man vorsichtig sein bei der Verwendung von fremden Rechnern, in Internetcafes sowie in öffentlichen Netzwerken. Im Zweifelsfall sollte man sein Passwort ändern, wenn man den Verdacht hat, dass es auf wenig vertrauensvollen Computersystemen mitgeschnitten worden sein könnte. Bei der Auswahl von Betriebssystemen und Software sollte man bedenken, dass proprietäre Software aufgrund der Geheimhaltung des Quelltextes und der damit einhergehenden Verletzung des Kerckhoffs'schen Prinzip meist nicht auf Hintertüren oder Schadsoftware untersucht werden kann.