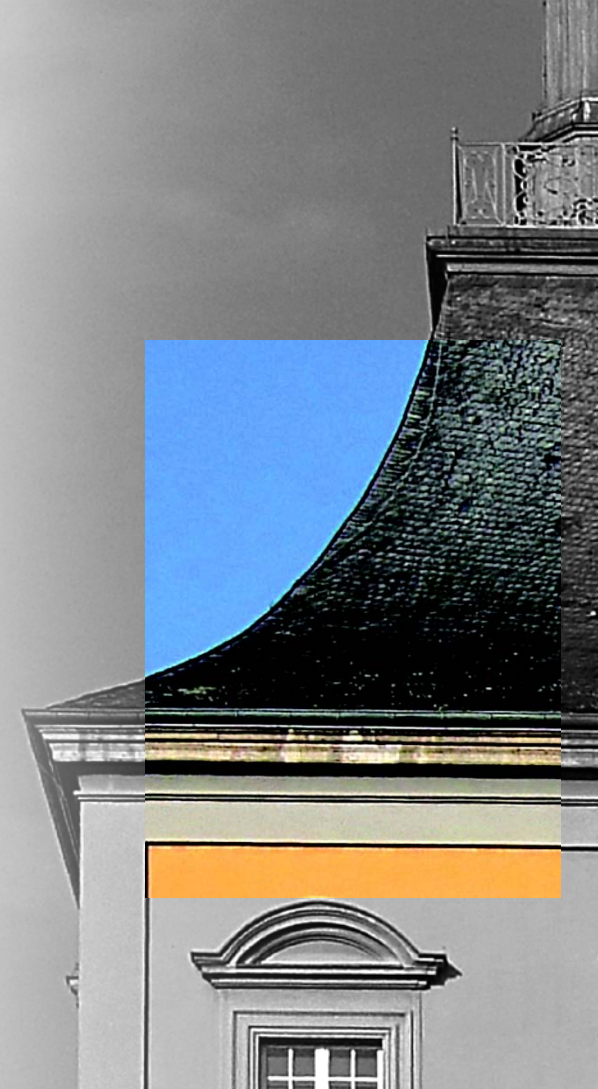


VORLESUNG
NETZWERKSICHERHEIT

SOMMERSEMESTER 2022

MO. 14-16 UHR



KAPITEL 3

PUBLIC-KEY- CRYPTOGRAPHY

Motivation zur kryptographischen Absicherung der Kommunikation:

- Inhalte sind vertraulich und nur für Berechtigte entschlüsselbar
- Daten bei Übermittlung und Speicherung nicht unbemerkt veränderbar
- Sender und Empfänger verifizieren sich gegenseitig als Urheber oder Ziel
- Urheberschaft einer Nachricht nicht abstreitbar



Verschlüsseln & Signieren

ACHTUNG: Nicht alle Ziele immer gleichzeitig erreichbar / gewünscht.

Asymmetrische Kryptographie

- Benötigt Schlüsselpaar
 - Öffentlicher Schlüssel
 - Privater Schlüssel
 - Öffentlicher Schlüssel von privatem Schlüssel abgeleitet
- Bekannte Algorithmen
 - DH (Diffie-Hellman; Schlüsseltausch)
 - ElGamal (ElGamal; Verschlüsseln & Signieren)
 - RSA (Rivest; Shamir; Adleman; Verschlüsseln & Signieren)
- Quiz: Wer ist auf dem Foto?

Asymmetrische Kryptographie

- Benötigt Schlüsselpaar
 - Öffentlicher Schlüssel
 - Privater Schlüssel
 - Öffentlicher Schlüssel von privatem Schlüssel
- Bekannte Algorithmen
 - DH (Diffie-Hellman; Schlüsseltausch)
 - ElGamal (ElGamal; Verschlüsseln & Signieren)
 - RSA (Rivest; **Shamir**; Adleman; Verschlüsseln & Signieren)
- Quiz: Wer ist auf dem Foto?



Absicherung von Kommunikation

- TLS (SSL)
- GnuPG
- S/MIME

Absicherung von Softwareinstallation

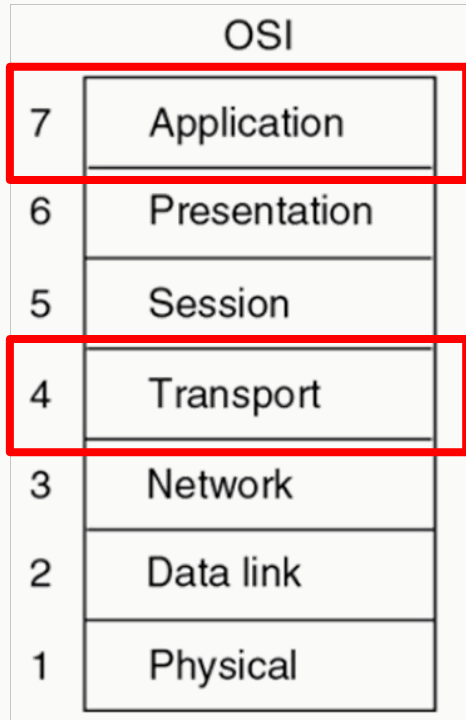
- GnuPG

Hintergrund

- Public Key Cryptography Standards (PKCS)

TRANSPORT LAYER SECURITY (TLS)

TRANSPORT LAYER SECURITY (TLS)



Was meint Transport Layer Security?

- Absicherung der Transportschicht?
 - Absicherung durch darunterliegende Schichten
- Absicherung durch die Transportschicht?
 - Absicherung der darüber liegenden Schichten

Vorgänger: Secure Sockets **Layer** (SSL)



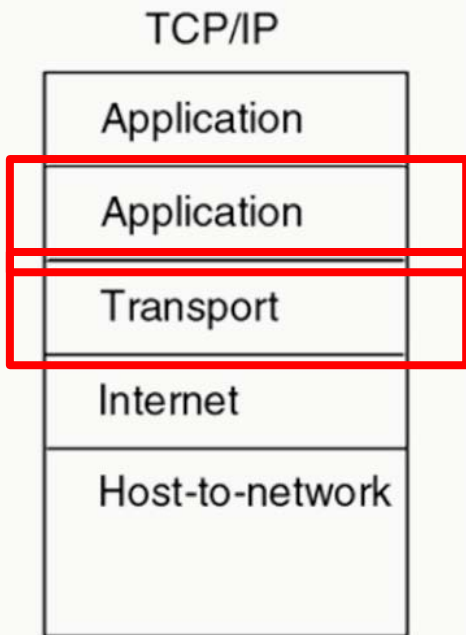
Eigene Schicht im Protokollstapel

Ziel: Absicherung der Anwendungsschicht

- OSI Layer 5/6 (Sitzungs- und Darstellungsschicht)

TRANSPORT LAYER SECURITY (TLS)

TLS im TCP/IP – Protokollstapel



- Betrachtung von TLS als Anwendung
- “Tunnel” von Anwendungsprotokollen durch TLS
- Bekannte Beispiele:
 - HTTP over TLS (HTTPS)
 - SMTP over TLS (SMTPS)
 - FTP over TLS (FTPS)

TRANSPORT LAYER SECURITY (TLS)

Historie

- 1994 SSLv1 (Netscape)
- 1995 SSLv2 (Netscape)
- 1996 SSLv3 (Netscape / Microsoft)
- 1999 TLSv1 (IETF Standard: RFC 2246)
- 2006 TLSv1.1 (RFC 4346)
- 2008 TLSv1.2 (RFC 5246)
- 2018 TLSv1.3 (RFC 8446)

TRANSPORT LAYER SECURITY (TLS)

Historie

- 1994 SSLv1 (Netscape)
- 1995 SSLv2 (Netscape)
- 1996 SSLv3 (Netscape / Microsoft)
- 1999 **TLSv1 (IETF Standard: RFC 2246)**
- 2006 TLSv1.1 (RFC 4346)
- 2008 TLSv1.2 (RFC 5246)
- 2018 TLSv1.3 (RFC 8446)

TLSv1 Updates:

- RFC 2712
- RFC 2817
- RFC 2818
- RFC 3268
- RFC 3546
 - Erweiterungen (z.B. SNI)
- RFC 5746
- RFC 6176
 - Prohibiting SSLv2
- RFC 7465
- RFC 7507
- RFC 7919

TRANSPORT LAYER SECURITY (TLS)

Historie

- 1994 SSLv1 (Netscape)
- 1995 SSLv2 (Netscape)
- 1996 SSLv3 (Netscape / Microsoft)
- 1999 TLSv1 (IETF Standard: RFC 2246)
- 2006 **TLSv1.1 (RFC 4346)**
- 2008 TLSv1.2 (RFC 5246)
- 2018 TLSv1.3 (RFC 8446)

TLSv1.1 Updates:

- RFC 4366
- RFC 4680
- RFC 4681
- RFC 5746
- RFC 6176
 - Prohibiting SSLv2
- RFC 7465
- RFC 7507
- RFC 7919

TRANSPORT LAYER SECURITY (TLS)

Historie

- 1994 SSLv1 (Netscape)
- 1995 SSLv2 (Netscape)
- 1996 SSLv3 (Netscape / Microsoft)
- 1999 TLSv1 (IETF Standard: RFC 2246)
- 2006 TLSv1.1 (RFC 4346)
- 2008 **TLSv1.2 (RFC 5246)**
- 2018 TLSv1.3 (RFC 8446)

TLSv1.2 Updates:

- RFC 5746
- RFC 5878
- RFC 6176
 - Prohibiting SSLv2
- RFC 7465
 - Prohibiting RC4
- RFC 7507
- RFC 7568
 - Deprecating SSLv3
- RFC 7627
- RFC 7685
- RFC 7905
- RFC 7919
- **RFC 8447**

TRANSPORT LAYER SECURITY (TLS)

Historie

- 1994 SSLv1 (Netscape)
- 1995 SSLv2 (Netscape)
- 1996 SSLv3 (Netscape / Microsoft)
- 1999 TLSv1 (IETF Standard: RFC 2246)
- 2006 TLSv1.1 (RFC 4346)
- 2008 TLSv1.2 (RFC 5246)
- 2018 **TLSv1.3 (RFC 8446)**

TLSv1.3 Updates:

- Bisher keine

Aufbau

- TLS definiert **zwei** eigene Schichten
 - Kontrollschicht
 - TLS Handshake Protocol
 - TLS Cipher Spec. Protocol
 - TLS Alert Protocol
 - TLS Application Data Protocol
 - Nutzdatenschicht
 - TLS Record Protocol

TLS HANDSHAKE PROTOCOL

- Ablauf
 - Cipher Auswahl / Abstimmung
 - **ACHTUNG:** Es gibt auch NULL-Encryption



NULL-Encryption Ciphers:

- TLS_NULL_WITH_NULL_NULL
- TLS_RSA_WITH_NULL_MD5
- TLS_RSA_WITH_NULL_SHA
- TLS_RSA_WITH_NULL_SHA256



TLS HANDSHAKE PROTOCOL

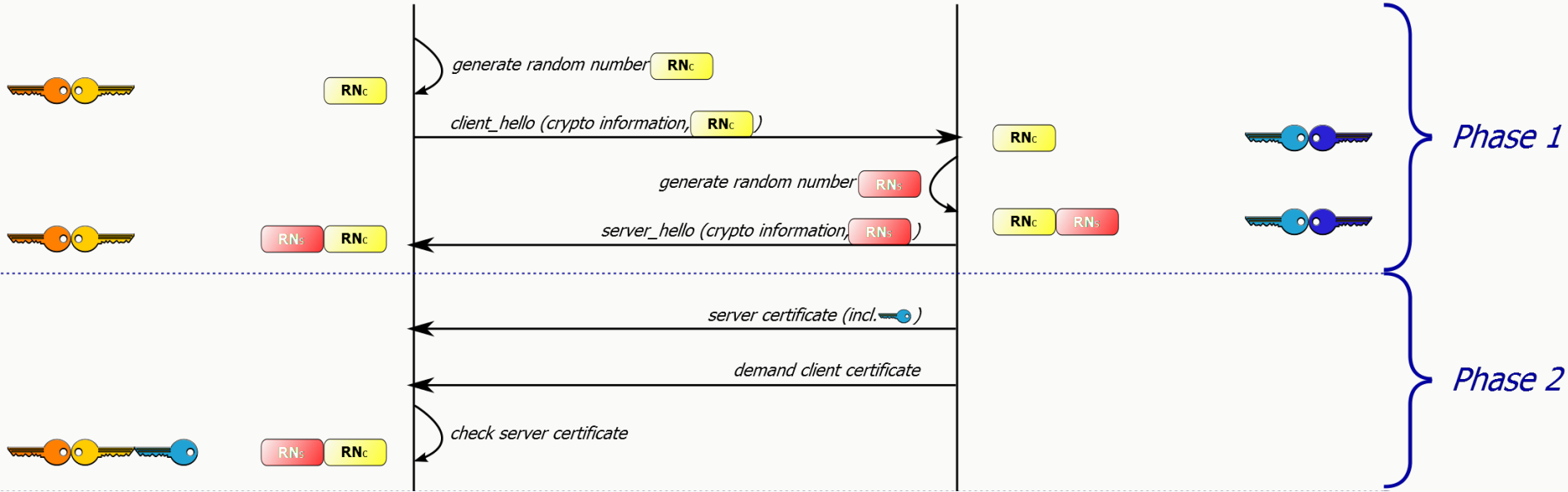
- Ablauf
 - Cipher Auswahl / Abstimmung
 - **ACHTUNG:** Es gibt auch NULL-Encryption
 - Schlüsselaustausch für asymmetrische Verschlüsselung
 - Serverauthentifikation
 - Clientauthentifikation
- } Authentifikation mittels X509v3 Zertifikat

TLS HANDSHAKE PROTOCOL

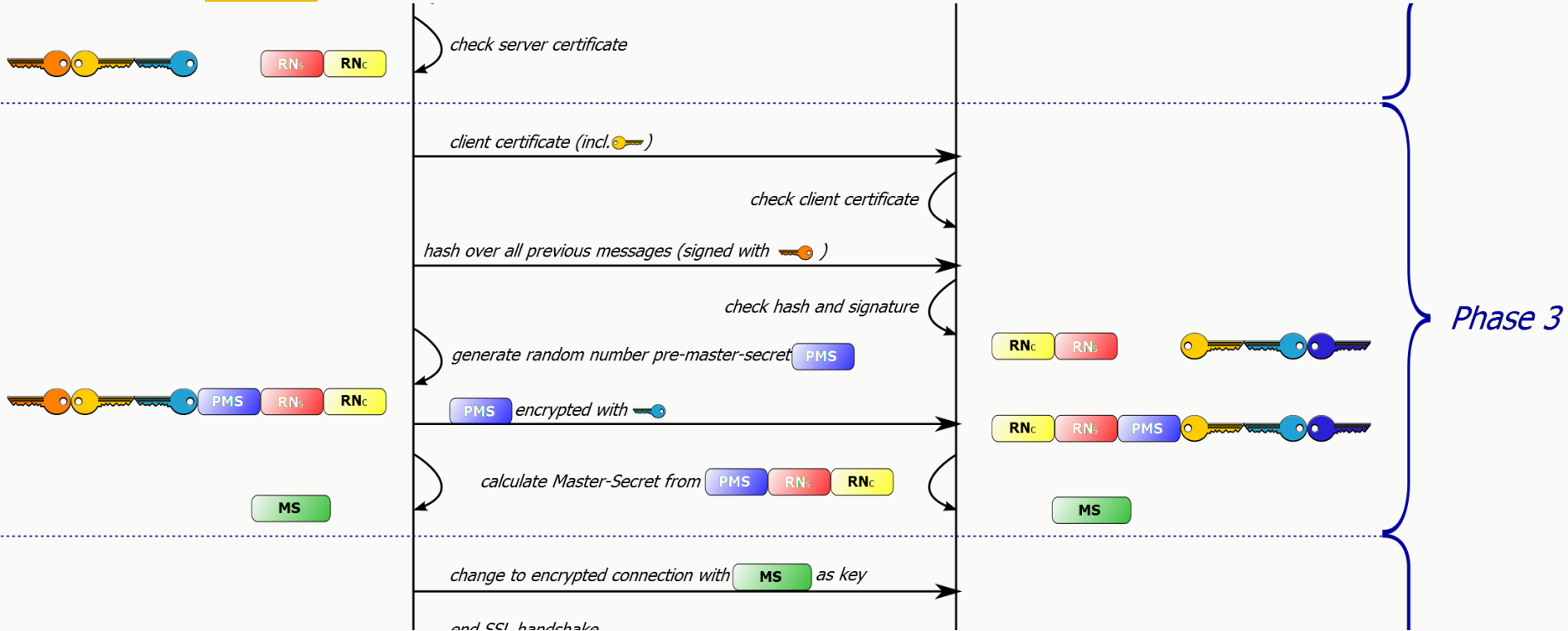
public key client 
private key client 

Client

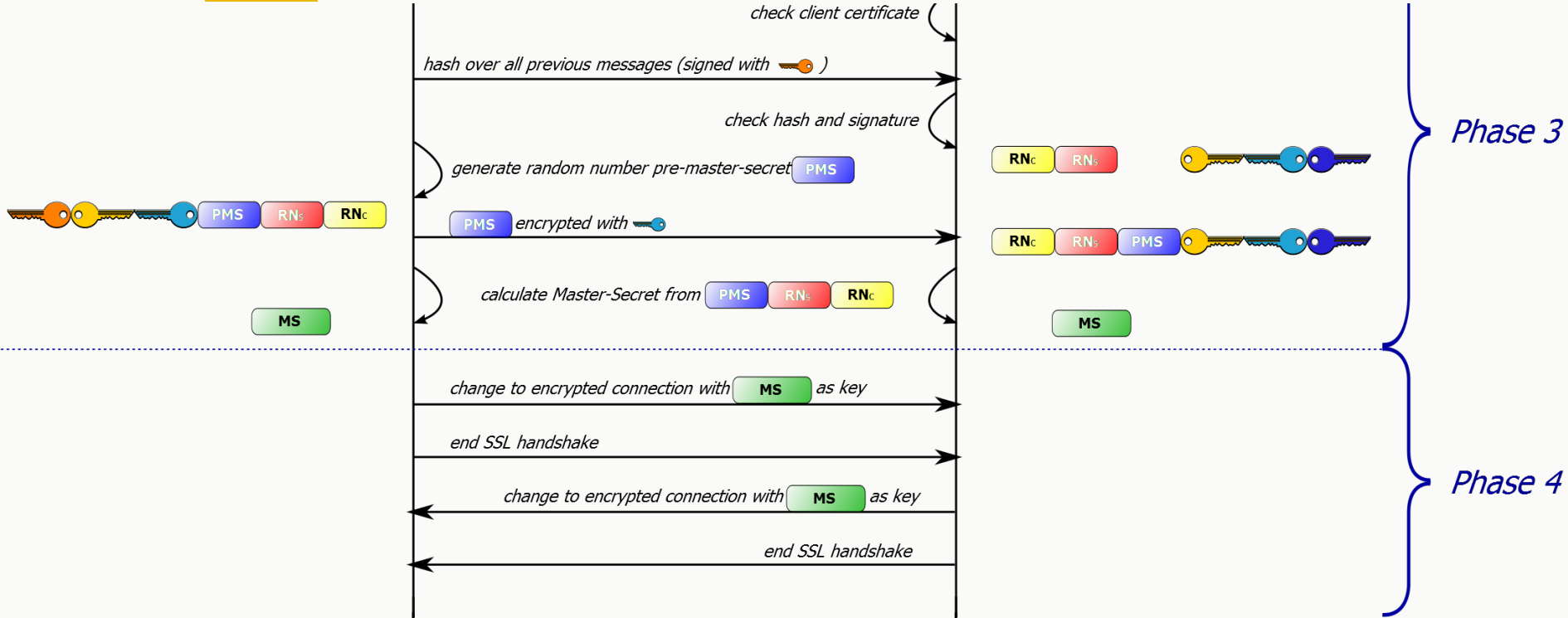
Server  public key server
 private key server



TLS HANDSHAKE PROTOCOL



TLS HANDSHAKE PROTOCOL



X509V3 (ISO/IEC 9594-8)

- ITU-T-Standard für Public-Key-Infrastrukturen
 - **ITU** = Internationale Fernmeldeunion der Vereinten Nationen
 - **ITU-T** = Standardisierungs-Einheit der ITU
 - **X** = „Data networks and open system communications“
- Spezifizierte Datentypen
 - Public-Key-Zertifikat
 - Attributzertifikat
 - Certificate Revocation List (CRL)
 - Attribute Certificate Revocation List (ACRL)

TLS RECORD PROTOCOL

Anwendungsdaten

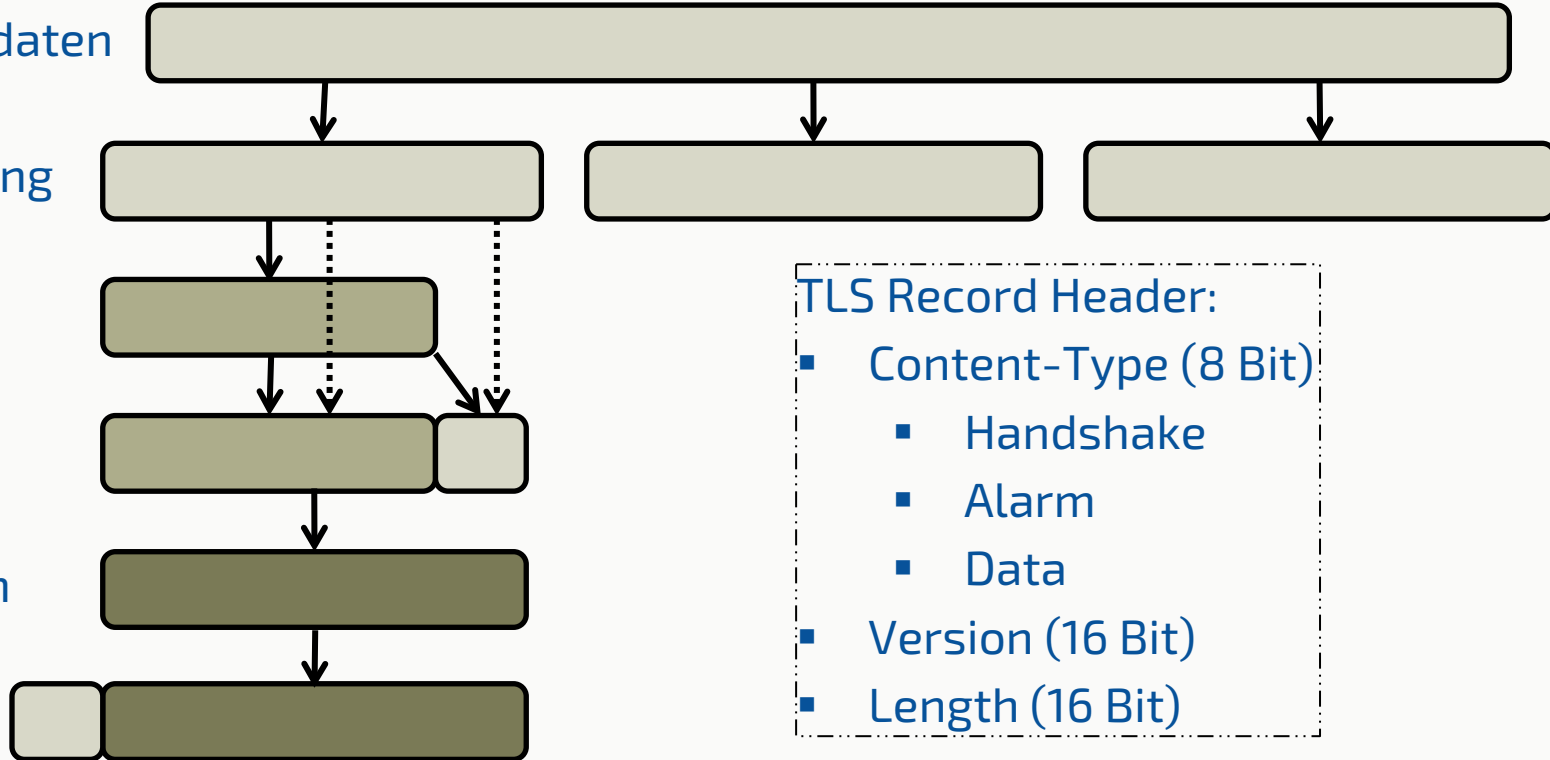
Fragmentierung

Kompression

MAC

Verschlüsseln

Header



Viele Standards, die heutiges Cryptographieumfeld prägen

- ITU-T (Vereinte Nationen)
 - X509-Zertifikate
- IEEE 802
 - 802.1X – Authentifikation am Ethernet-Port
- RSA Security Inc. Public-Key-Cryptography-Standard (PKCS)
 - 15 Standards und Definitionen für Public-Key-Crypto
- Request for Comments (RFC)
 - Organisationsübergreifende Veröffentlichung von Standards (bzw. Entwürfen und Updates)

- PKCS#1 – RSA public key crypto
- PKCS#2 – RSA encryption of message digests
- PKCS#3 – Diffie-Hellman key agreement
- PKCS#4 – RSA key syntax
- PKCS#5 – Password based cryptography specification
- PKCS#6 – Extended certificate syntax
- PKCS#7 – Cryptographic message syntax
- PKCS#8 – Private key information syntax

- PKCS#1 – RSA public key crypto
- ~~PKCS#2 – RSA encryption of message digests~~ Merged in PKCS#1
- PKCS#3 – Diffie-Hellman key agreement
- ~~PKCS#4 – RSA key syntax~~ Merged in PKCS#1
- PKCS#5 – Password based cryptography specification
- PKCS#6 – Extended certificate syntax
- PKCS#7 – Cryptographic message syntax
- PKCS#8 – Private key information syntax

- PKCS#9 – Selected attribute types
- PKCS#10 – Certification request standard
- PKCS#11 – Crypto token interface (cryptoki)
- PKCS#12 – Personal information exchange syntax
- PKCS#13 – Elliptic curve cryptography
- PKCS#14 – Pseudo random number generation
- PKCS#15 – Cryptographic token information format

PKCS#1 – RSA PUBLIC KEY CRYPTOGRAPHY

RFCs:

- | | | |
|------------|-------------|---------------|
| ▪ RFC 2313 | Version 1.5 | März 1998 |
| ▪ RFC 2437 | Version 2.0 | Oktober 1998 |
| ▪ RFC 3447 | Version 2.1 | Februar 2003 |
| ▪ RFC 8017 | Version 2.2 | November 2016 |

Definitionen:

- RSA Schlüsseltypen für öffentliche und private Schlüssel
 - Öffentlicher Schlüssel:
 - n : modulus
 - e : öffentlicher exponent
 - Privater Schlüssel
 - n : modulus
 - d : privater exponent
- “Multi-prime” RSA (ab PKCS#1 v2.1):
 - Modulus ist das Produkt von mehr als zwei Primfaktoren

Definitionen:

- Umwandlung von Datentypen (Integer \leftrightarrow Octet-String Primitive)
 - I2OSP
 - OS2IP
- Ver- und Entschlüsselung (Primitive und Operationen)
 - RSAEP $((n, e), m)$ mit m = Nachricht (Integer)
 - RSADP (K, c) mit K = privater Schlüssel & Parameter zur Erzeugung
- Signatur und Verifikation (Primitive und Operationen)
 - RSASP1 (K, m)
 - RSASV1 $((n, e), s)$

PKCS#1 – VERWENDET ASN.1

PKCS#1 sieht für die Repräsentation von Schlüsseln das ASN.1-Format vor:

- Abstract Syntax Notation One (ASN.1) – ITU-T-Standard (gemeinsam mit ISO)
- Definiert Repräsentation von
 - Schlüsseln (öffentlich/privat)
 - Zertifikatanfragen (CSR)
 - Zertifikaten
- Darstellungs-/Übertragungsformate:
 - DER
 - CER
 - PEM (nicht Teil von ASN.1) – oft Base64 encoded DER

Privacy Enhanced Mail
(definiert durch IETF)

- RFC 7468
- Encoding von
kryptografischem
Material

ABSTRACT SYNTAX NOTATION ONE (ASN1)

Ein Guter Einstieg (wer es wirklich wissen will):

- Olivier Dubuisson and Philippe Fouquart. **ASN.1: communication between heterogeneous systems**. San Francisco. 2001

Als OpenBook: <http://www.oss.com/asn1/resources/books-whitepapers-pubs/asn1-books.html#dubuisson>

Merkmale von ASN1:

- Beschreibt Datentypen (Syntax ähnlich einer BNF) und Encoding
- Zum Informationstausch zwischen unterschiedlichen Systemen
- Lange Versionshistorie (X.208 von Nov. 1988), aktuell: ASN1:2015

Datentypen

- Primitive Datentypen (Auswahl)
 - BIT STRING (binäre Zeichenfolge)
 - BOOLEAN
 - IA5String (IA5 kodierte Zeichenfolge)

ABSTRACT SYNTAX NOTATION ONE (ASN1)

Datentypen

- Primitive Datentypen
 - BIT STRING (BOOLEAN)
 - BOOLEAN
 - IA5String (IA5)

				b7	0	0	0	0	1	1	1	1
				b6	0	0	1	1	0	0	1	1
				b5	0	1	0	1	0	1	0	1
b4	b3	b2	b1		0	1	2	3	4	5	6	7
0	0	0	0	0	NUL	DLE	SP	0	@	P	'	p
0	0	0	1	1	SOH	DC1	!	1	A	Q	a	q
0	0	1	0	2	STX	DC2	"	2	B	R	b	r
0	0	1	1	3	ETX	DC3	#	3	C	S	c	s
0	1	0	0	4	EOT	DC4	&	4	D	T	d	t
0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u
0	1	1	0	6	ACK	SYN	&	6	F	V	f	v
0	1	1	1	7	BEL	ETB	'	7	G	W	g	w
1	0	0	0	8	BS	CAN	(8	H	X	h	x
1	0	0	1	9	HT	EM)	9	I	Y	i	y
1	0	1	0	10	LF	SUB	*	:	J	Z	j	z
1	0	1	1	11	VT	ESC	+	;	K	[k	{
1	1	0	0	12	FF	IS4	,	<	L	\	l	
1	1	0	1	13	CR	IS3	-	=	M]	m	}
1	1	1	0	14	SO	IS2	.	>	N	^	n	~
1	1	1	1	15	SI	IS1	/	?	O	_	o	DEL

Datentypen

- Primitive Datentypen (Auswahl)
 - BIT STRING (binäre Zeichenfolge)
 - BOOLEAN
 - IA5String (IA5 kodierte Zeichenfolge)
 - INTEGER
- Kombinierte Datentypen (Auswahl)
 - SEQUENCE (geordnete Abfolge verschiedener Typen)
 - SET (ungeordnete Abfolge verschiedener Typen)
 - SEQUENCE / SET OF <Typ> (Abfolge gleichen Typs)

ABSTRACT SYNTAX NOTATION ONE (ASN1)

```
RSAPrivateKey ::= SEQUENCE {  
    version          Version,  
    modulus          INTEGER, -- n  
    publicExponent   INTEGER, -- e  
    privateExponent  INTEGER, -- d  
    prime1           INTEGER, -- p  
    prime2           INTEGER, -- q  
    exponent1        INTEGER, -- d mod (p-1)  
    exponent2        INTEGER, -- d mod (q-1)  
    coefficient       INTEGER, -- (inverse of q) mod p  
    otherPrimeInfos  OtherPrimeInfos OPTIONAL  
}
```

Schlüsselerzeugung mit OpenSSL

```
$> openssl genrsa
```

RSA PRIVATE KEY

Schlüsselerzeugung

\$> openssl genrsa

```
[matze@tschita] /tmp $ openssl genrsa
Generating RSA private key, 2048 bit long modulus (2 primes)
.....+++++
e is 65537 (0x010001)
-----BEGIN RSA PRIVATE KEY-----
MIIEpAIBAAKCAQEA008YK4TzuJDJmVjZXRnlrBF4d74tx+23SV31tOMEmi3cDVTf
nNKBINR7GfwoT6EO587rBaPSIRJd8pa346iojBonCqpMRwEskAij28cNYiLueILO
CncBlaKGqPPGOZF8wNokvWH77kiUAJCBHqvtjhLQuCgZH17CzBKM2HkUTcaH0ErE
XqOalApy7tPymI2ue5+ClFjRU+9lKaj/acUGANADUmAy8K4icUGI/vSQcXWlxLK
D+shXYyy1FKow0iJfVn5A89UXWTZdp3RSHrPB6Q3y2lael8AdERzzNIlOphy2Ep1
GfRtbaf3N6WVYDYQIKO+qWFMcj77talmsn4gSIwIDAQBAoIBAH6fnvQ1L3ciC+8n
f9prxPPfZDkdFYIAyRyF2X8TquZZJmW4V+c5nXd23G2t1NoiWTPXog42hOycfP+p
oXgi0R1jbpHNivfhldqbctV4amSsWgqNF7rdpWltfxQvQFGvBPrn2Ihd9xvHLQVJ
kpU9WJUSYVB5dtR9jG2NkCkIJUqU3aGIDsjm4F4xqvDzrqJleO1lZaVtWuTSvU8I
J3pnZ5wL5MGA9PWWW/5EOqe2h2Cuqech0+baOUeGcGNpgE9GmeP8A6m+uyTcodlq
9D/Yebgw17IKTr3OcoQvpwMFDWMihlWJfBQrzKOPm+pJqdBYZefyAdsXM594ZXaZ
E0Y9ccECgYEA+5xcCIDKDBbKzkSE4Qt8cenYBbIoIF8t/CB9HyubwPmTjOFK2VIN
r3nml2oXccKPfXopIpMhSLn7rvax+aUq4j1LQJS0ig2TW4upmOU4DZs2sPox5d5n
YE6hwalUja/uElzCZYqH0/gUBfqNQfTPEI2Dcnq2dcVjEtRiH8yCMwcCgYEA1v7B
agNTpiz4qS9nAfh3vVVRjiBicHr+zd1MXTj37qweB3Y3xHq7M3swt8Ln+y0kcbCF
xNfNBseNGr9Duux1VaK4x+/YX2OXub7bTuXdDf1fL7YMrDMPx3NIMkBoJ/Em8sOo
QelserHXIb2bMYJFWxGkDQilyhNKFjAyLfm1QUCgYEAifZXZHnCOjjThTx+2+Lv
borHoSXP6K4nye2iTDgchDiVFjdmUj1a92/Q549NoZv8D+JO8d1Y78bC4VH1jpxp
IjgmMhgX0mbVOOogqWxuSs+jsnDNsWw8pMsY2NafPT2NPmKrWQAXsDujxw92Kwcw
DrR5sejlaTg9NdLspu9NzsCgYBvWA6pLChjnQTG8iLB1YGF0wirN8Bjin9t9H85
BGj69a6zpJjrK7jJx7IKaWmlHtLyAj6lHCiIsaE1Sfr0z2WIjTZvbZq8hCKU8tY3
fTPho2+VvkQ3tHKC+ZcjtJYgLR4vBTGLWfWJOBM5qzfys3n/XyHSe+DFXBriuH5Dw
IrxLVQKBgQDEXVEVznj/okAec/Y/gwH6qnry7rcEYy7stfcYlVQxOGOKHVbaQ62J
5HIv6474Buqmh38lyzpjpojZhesQzotBltQzitZYdUwjSL/iNcahi2wpXAxxr0AYM
w9RgDqSB4pm7AzHz4oia4cNP74udVYp4poolVZXrDD8PaiOXdG1/Pg==
-----END RSA PRIVATE KEY-----
```

RSA PRIVATE KEY

Schlüsselerzeugung mit OpenSSL (4096 Bits)

```
$> openssl genrsa 4096
```

RSA PRIVATE KEY

Schlüsselerzeuger

\$> openssl genrsa

```
[matze@schita] /tmp $ openssl genrsa 4096
Generating RSA private key, 4096 bit long modulus (2 primes)
.....+++++
e is 65537 (0x010001)
-----BEGIN RSA PRIVATE KEY-----
MIIEQgIBAAKCAgEApIL8sYFqJdVwMhTP2GgEqC8KRE2eNtWdy6fI2QcMuzVRvcY
vH9/mknDHGIMSCCAN6K4Nma1nlqP5UgksRpIgrbmQRJh6MUKL+Xhs3r6VDZpqKdI
kFDsnkd7ILfFYRQsg1012xeFku9cGott1Vguaz7OG37CKJgf2+vLnN4RNOBns12Kw
V0bTW/pSktq8FDxcmZESL+nqoy1LlWktbpVQ0/etICHL2FN4ODKuZj3ad3nJKIaJE
nkbSSWsfz+W2GyyuhHdC8g+EmBwLm7vRtHNUpt22tgcVN8PKKdWu263qzZX004
5BPQwduHn1kZ258JSE370wMR/+daI07Hs06jgfSuttl1Nid1x5Y8agct8u1
ueBdKncRE7vzspS7J5M20AHJVD5BNvLWUp8ECLFQWtew8SbjSnbwge0/74e1TF2
Tn9qkWO3G91v3rHMq7/fa05ccj1R8xp78v371WlaRocQ85Mkt1Fcud+Og0Uht6
hi8/7gkY5/1WGBQJmkoYns7BgObaax0tJr51eaaKQ/BNK3pyypzDzPoV9pugLzo
c92LwLurIMvi326JhgMRJ2w7VOB1/3VvuBmfRdt657ytw9v8Y12YpUL50eS90mG1
40iasosmhcmSgcc+Zc/0xcD1zU1u2Fh1awtaNm0KP0wjeR/aUZ/S9ZnHD0CAWEA
AQKCAgBAQ8r2erTRTbAege9Tte5q4S0r2i0jwk9dxUtrDHALoDdU/Hc/NXYNzbH
v6QEWze2z1Q0RvD1Wq1PnD1wu0fbZj513JwzjoKAJP9wWz70o11VT/mxiciuSMLVl
ctLabTtY1P+SnTKT1OYODhpx5VqhJs02M6wU72rXWkv3vFH11fa8GrD031p+JMTQH
pU0WwPAlKkognuBpWQcZqS1Z1U1Vjgt4e00b6xwVf5nuKJ11D5j4u8oJw0QP08
dxy8obeS11EQG63shmd6EHEAKAQ0BeQfRQ2agnAchGvRI6AuaZ30Rjgo61LGTOP
A+wp+5C5LtaFxmNn9DdsWanHrTwe7XtSDf0uKND0a1Aja72B0XPyPjocgfg/RJCY
OS351JmmkEjyVELXasNouLAUdm84BJCJBDKbzqCT0M34tROHvPw1NH2IAecPfkC
CK05gmZy4t5vEnHete2Rkt+7aL8l9mCtFVNLH8pWSP+c6n4FFKdJfRyYkwa6R
wmh50ADyb1FFB1KwMP/8CpFuzB3ycW0x8MR+3gK04/Vapj9dtIWC3MSM1RoS8zQp
x4J3GJ3G1ZG1Nzre78Sat1x6IgPpLFejFDQg1og5c+EtYRR7jx8KbHu7rJtBNEatH
jWU0gQ1w9QXK2V1mKvC/hlmucv8tFkVb4pejh1v63/1qtC4QKCAQEALVW11Noo
P1E1R7BuYed1AY5+HoAlmbzKrq6BGM12DVXe01YrS7S1lgHgLFBoorCcuUGogXR
WdyceV07PpgQ4HwLzdNsc42rTvRk3z+rvdVEXY4Sw7PY7jHpoUca4FGeyLVgtBm2
21PjKdq052HY0UxgpTp2fMwGtnYsdMoJWyQODClyTdkyXJLyOquvHra41202am
/6U//Sh11+vok1VDgdgbBV+hsPNIby0Pansp9s1z8JbsJoghb/WdyJMAw0p0FvkPvt
/rm1ZkausUvpI2xJwTY9C1PkE4Wod+va5E/rw1Cj3twpvXTITmyVyh4drosP8o
UXsozQF1C42VfWKAQEAYm193Kbzq25eU14EKVU1LrAHPH00uKmI9w9p7634BLS
vSSN8ges2vJmASH7zKjFR0MQKtOZa0s3k05P7Kkwo2Us8WymuWtN9XIWxerKtAH
D2fTaQc01fyBtLwDQtztYsptzeKolvcqk8RYWynbJt5XA7g7SXzeKt7SAkrMw
qg7JWJAT01GyctTnlQouW0RAVbmdE1IWMFaxyTOCJA+PMF3BNQ/DAoq1mw+4J8sg
8T984ZnfiIghak9unTVrVrvtKq/Abg5WIC0102dpFdd5r1Q0UHD9XGd1IAe0
QdPzsaBMBwmlXedw/06Uu607jE8vcs+u49H89wKcQZ1SHmdg/WyB3z+14
h8q2wClnzyedndekv8wnYjBvEXStZienGvTohBP8RGjoxURKJMQGuctE5bst1
SmgctKfys5Pw7P3o6gLYex0D600hH3fKE09AtpxczbFG61jv3yie+1zyGdyYUS4V
wo/VH4U0zs5hnnWQzBapbbjshnu52YUFP8bmb7Vhc/dBLnSPDPsWh0XQp8ngBkt
EE5rcoez9u3s81cphdYXbQw77xbqv5e9vMjQFW4uRk1M66+e3vUkFall14Cw
OHuueyB1PFEOfF1IPAuuEVK0K0e0Q0Udlb9rI3HncrSuzqWw4E3CtpvQ5BRDNKca
s0ANtWKAQEAj5T0mm00iN/wEustTCZpHM188x90+3xyVSYIH0ras11X15UvXvc
bg7FQ0oxvPt3/RnmiMmR71D121WUn+512rNprGon6RQ6VFYxzyDHcMyb5o2aue2
9QDh1+6R3DixR0Cwk45j7MU1Bjsu6FvkZ1nlu6uChHo249IjB2nZag5roDbIFh6P
Dho9SviGmRkHpKjPukLU1LALjL7PRI1G6TAv1S0me4whJPA3dg5x3MRQpDhDS/G
NvdAmNR4+PyW4k0D1HVjocHsGvxtJMrq14fx78LG2SnB3of/PMGR5QCZc1wpYhJ
y3uZiYV0HspidUhj5Z/xzDh1UOF0r0AAwKAQBN7Jr9mVdfwq1/tlQwcaxR5ZtQ
81PyOYK/NmrfGmsk1gSR73qGTOWt/Y+c4WU+cmZ409Yxj2M2atgQOM7NmC/qy4
ZKz5SzgmW8P77SKa/0uohp44tjYVUD0cAdMPZz/bgnCnEp7/7aUFIxQ9ncx
zyvtafcm/Fp7VtounpRjYpkL6Z0/hv46bWnZAKYrYf7i0ciZ+e0B+Jr0M
DXh95E1jUusHqo10hNHOVYN7ZJc7Ipe1HdejteWqRt+ryiKcmSchM1S9yuot8
qzd5ciX5ZzwaegHU0qkXS4T/Hmby70KvL8ULG1j4wdgi2PD0jOTeWgcmZK
-----END RSA PRIVATE KEY-----
```

RSA PRIVATE KEY

Schlüsselerzeugung mit OpenSSL (4096 Bits)

\$> openssl genrsa 4096 **ACHTUNG:** deprecated!

\$> openssl genpkey -algorithm rsa -pkeyopt rsa_keygen_bits:4096

RSA PRIVATE KEY

Schlüsselerzeug

\$> openssl genrsa

\$> openssl genrsa

```

[matze@tschita] /tmp $ openssl genrsa
.....++Generating RSA private key, 2048 bit long modulus (2 primes)
++++
-----BEGIN PRIVATE KEY-----
MIIEpAIBADANBgkqhkiG9w0BAQIwDQYJKoZIhvcNAQEFBQAwggEiIge35527 (0x010001)
/YP+bkc3YCBaVxRP1vK9lu/81-----BEGIN RSA PRIVATE KEY-----
W+374zMvEQQsCqyarJlAs8kyIIEpAIBAAKCAQEA008rk4rzuoDJMvjZXrnlrBF4d74tx+23SV31tOMEmi3cdVTF
KJOosLdkSnPFBj18clWo6/c31nNKBINR7GfwoT6EO587rBaPSIRJd8pa346iojBonCgpMRWeskAij28cNYiLueILO
DAOPZcRpm/7suu5SjWeQSRgEcncBlaKGqPPGOZF8wnOkvWH77kIUAJCBHQvtjhlQuCgZH17CzBKM2HkUTcaH0ErE
1qCR17p15eh5UuNudy9WVmlXxqOalApy7tPymI2ue5+ClfjRUP+9lKaj/acUGANADUMay8K4icUGI/vSQcXWlXkLK
Yw8KnKbLagMBAAECGgEAQdkZD+shXyYy1FKow0iJFVn5A89UXWTZdp3RSHrPB6Q3y21ael8AdErzzNI1Ophy2Epl
P21YBTONav3tPu6Blwh4h61LlGfRtbaf3N6WVYDYQIKO+qWfMCJ7taImSn4gSiWIDAQAABoIBAH6fnvQ1L3ciC+8n
9G5fi+taV7yAqGLrOonVBbuEuf9prxPPfZDkdFYIAyRyF2X8TqfZJmW4V+c5nXd23G2t1NoiWTPXoq42hOycfP+p
uXSU3CPDVfGxck0BPpm8c4CQoxgi0R1jbpHNivfhldqgbctV4amSsWgqNF7rdpWlftfxQvQFGvBPrn2IhD9xvHLQVJ
km7cPAG9f/fKw1sWaicmGC/s'kpU9WJUSYVB5dtr9jG2NkCkIJUqU3aGIDsjm4F4xqvDzrqJ1e011ZaVtWuTSvU8I
SZxpZbnQivnpfcZ2RdbH6sCRlJ3pnZ5wL5MGA9PWWW/5EOqe2h2Cugech0+ba0UEgCGNpgE9GmeP8A6m+uyTcodlq
LoykhDYhIjchMlrw2U7ArIBv9D/Yebgw17IKtr30coQvpwMFDWMihlWJfBQrzKOPm+pJqdBYZefyAdsXM594ZxaZ
k1URhvnDp0a0xd/fuGqK2TQ3E0Y9ccECgYEAA+5xcCIDkDbbKzKSE4Qt8cenYBbIoIF8t/GB9HyubwPmTjOFK2VIN
KE1qW2jwhscoQrGxVe9Y3ZWZr3nml2oXccKPFxopIpMhSLn7rvax+aUq4jllQJS0ig2TW4upmOU4DZs2sPox5d5n
FFVO2SQFJv/k3JBEEmOzSXfcaYE6hwalUja/uElzCZYqH0/gUBfqNQFTPEI2Dcnq2dcvJEtRiH8yCMwcCgYEA1v7B
zVwIop8qXn+Ma5119ZAjprVNagNTpiz4qS9nAfH3vVVRjiBicHr+zd1MXTj37qweB3Y3xHq7M3swt8Ln+y0kcbCF
62aElncuFQKBgElCEPoaafgjXnfnBsenGr9DuuX1VaK4x+/YX2OXub7bTuXddflfL7YMRDMPx3NIMkBoJ/Em8s0o
4eheOV2JXS+86P/rYesndoywQelserHXIb2bMYJwFWXGKDQilyhNKFjAyLfm1QUcGyEaifZXZHnCojjThTx+2+Lv
j6JehmpWrOWIDGTfEl6k28nliIjgmMhgX0mbVOOggWxuSs+jsnDNsWw8pmsY2NafPT2NPMKrWQAXsDujxw92Kwcw
MDZ1ISNjtQ+uA+UumpvLEbDclDrR5sej1aTg9NdLspuD9NzsCgYBvWA6pLcHjnQTG8iLB1YGF0wirN8Bjin9t9H85
MJZCxiM047X1Jh/icvS/b3SjBGj69a6zpJjrK7jJx7IKaWmlHtLyAj61HciIsaE1sfr0z2WIjTzvbzq8hCKu8ty3
LSY4vXKvQWyVE9TTisFH+GjpfTPho2+VqQ3tHKC+ZcjtJYgLR4vBTGLWFWJOBM5qzfys3n/XyHSe+DFXBriuh5Dw
VwhqPU9etCzTWgpgZxjIP7YP:IrXLVQKBGQDEXVEVznj/okAec/Y/gwH6qnry7rcEYy7stfcYlVQxOGOKHVbaQ62J
zeL06ADP4ogDZd8wtJ+DIjXY:5Hiv6474BuqmH38lyzpjpoJZhESQzotBitQzitZYdUwJSL/iNcahi2wpXaxr0AYM
aQaqDIVxspSRW5kvtML0g==w9RgDqSB4pm7AzHz4oia4cNP74udVYp4poolVZXRDD8PaiOXDg1/Pg==
-----END PRIVATE KEY-----

```

RSA PRIVATE KEY

Schlüsselerzeugung mit OpenSSL (4096 Bits)

\$> openssl genrsa 4096 **ACHTUNG:** deprecated!

PKCS#1

\$> openssl genpkey -algorithm rsa -pkeyopt rsa_keygen_bits:4096

PKCS#8

RSA PRIVATE KEY – ASN1

version
modulus
publicExponent
privateExponent
prime1
prime2
exponent1
exponent2
coefficient

Version,
INTEGER, n
INTEGER, e
INTEGER, d
INTEGER, p
INTEGER, q
INTEGER, d mod (p-1)
INTEGER, d mod (q-1)
INTEGER, (inverse of q) mod p

```
-----BEGIN RSA PRIVATE KEY-----
MIIEpAIBAAKCAQEA...
00000000 00 32 04 a6 02 01 00 02 02 01 01 00 05 de 4d e2
00000010 03 34 40 01 a3 76 80 01 01 01 01 01 01 01 01
00000020 af d2 f1 f8 39 2c d1 5f c5 70 02 68 ec d0 a8 ad
00000030 f2 c7 f1 21 02 d2 b6 be f6 91 10 eb 73 56 ec 09
00000040 4e 72 e4 04 01 0c 02 02 12 b3 d2 17 86 6c 02 a4
00000050 24 24 20 4a 57 b6 a3 28 a2 c4 1c f2 16 51 74
00000060 fb 54 5b ed 49 4a e5 19 4b f7 7c 77 52 52 ba 56
00000070 69 9b 87 3e 74 07 e2 a5 f5 35 8d e5 26 4f
00000080 1c 0b e0 01 5c 07 75 79 2f 0c aa 68 94 8c 08 03
00000090 d2 b8 09 01 c1 53 75 5b 55 aa e7 4d 5b 88 0a 34
000000a0 bd 6e 01 1b 3b 2a 10 2a 3c 3d 65 8b 04 ff
000000b0 14 10 aa 12 9b 26 b0 46 5e 10 00 55 70 00 44
000000c0 b6 9b a4 57 9f 19 23 75 72 e8 85 0c 38 aa 71 65
000000d0 8c 3a 2b b5 43 20 8a 08 ab 07 66 99 8b 33 3c
000000e0 6f d2 b5 07 9f 04 22 09 1b 10 14 ef cd 55
000000f0 8e e3 13 6c dc 04 1f 40 62 c6 28 aa e2 b2 04 2d
00000100 da b5 9d 3d 57 45 8b 53 fe 75 53 ff 02 03 0a 9b
00000110 01 02 82 01 01 00 9a 68 7a 13 5a 3e 3f 8e 19 a0
00000120 00 44 e0 c6 15 22 20 a8 ec ac e7 bd 76 00 e4 34
00000130 08 26 15 8d 40 7c fc a5 d5 95 b6 dc 06 0a
00000140 74 48 51 3c 3a 9a 5a 39 d1 4f ad e7 a5 0d 4f bc
00000150 5e fa 02 17 44 68 ff 1a fe 69 e7 52 00 0a 20 aa
00000160 45 de d2 39 88 84 15 d0 a1 e6 02 15 8b 6c 3a
00000170 6d 50 22 39 8c 55 e5 0b fa ad 09 17 e7 8b fa 44
00000180 d6 0e 56 6a ec 3b 8f 61 3e ac 61 d0 a7 18 0c 02
00000190 54 fd db 9b fb 21 ff c4 aa fd ac 06 39 c4 82 20
000001a0 29 20 2e f7 04 a3 da 40 0a 17 02 0c 0c 0c 0c
000001b0 c3 c2 99 65 8c e7 db fa d7 0d c5 ab aa 23 66 b6
000001c0 24 c7 15 1d e2 32 e4 a4 5f 8c 0c 0c 0c 0c 0c
000001d0 8f 12 d1 0a e4 41 4a 8a 7b 24 13 78 33 67 17
000001e0 51 bf aa 49 aa a4 30 2c 06 f7 db a3 44 e3 ec 40
000001f0 01 44 3a bc 8e 28 4e 58 b6 97 d2 84 22 b1 35 ad
00000200 09 3a 7b 09 29 fe c5 2a 8c 36 3c 3b 27 8a ad da
00000210 69 07 af c8 da d1 02 81 81 00 e4 1e 23 ff e2 7e
00000220 90 03 5c ab bc 11 b0 20 8a 1b da c1 c6 e5 6c e2
00000230 14 e2 64 3c 09 29 c0 4d 5e 14 c1 c6 e5 6c e2
00000240 3a b8 dc ab 04 13 d1 c1 8f 5e 22 0f 8d 67 31 e6
00000250 13 16 f7 11 3b 9b 5a 39 d1 4f ad e7 a5 0d 4f bc
00000260 c3 c2 99 65 8c e7 db fa d7 0d c5 ab aa 23 66 b6
00000270 db e3 da f4 e7 4d 50 34 31 e5 29 d2 05 db bb ec
00000280 30 5e 46 67 82 38 5b 68 16 65 8f aa 1f 0d 13
00000290 9a f4 e7 25 ad 04 69 02 81 81 00 e4 1e 23 ff e2 7e
000002a0 02 88 04 8d 77 77 66 90 df 54 a1 14 89 05 03 46
000002b0 bd af 77 a9 ff 67 6a 39 5e 11 c2 66 38 e5 84 2c
000002c0 a6 0c 0d 3b 9f c6 4d 5e 14 c1 c6 e5 6c e2
000002d0 2b 21 aa 8a e5 6c 5f af 04 4e 5c 0e 1a 32 a1 47
000002e0 39 3d 3a 32 a2 3f df 41 56 2b 3d 69 aa 07 a3
000002f0 07 44 4f 8f 8e 8d 82 3b 9a 17 0c 0c 0c 0c 0c
00000300 b2 26 90 26 cd 4a 8b b5 ef d4 a6 1a 1a 37 02 35
00000310 0a 51 74 93 78 28 2b a2 5f b4 d6 46 08 27 02 81
00000320 81 00 d1 e4 ab c3 0c c8 ab bc 75 80 09 0e de
00000330 9d 4c 16 c2 ka 9e d8 04 55 35 0a 4a bb 83 9a 01
00000340 d1 fa 93 c2 22 7f e0 07 07 35 06 5e 0f 26 64 84
00000350 04 e5 e7 7e 27 1c 36 8a 61 49 87 4a 8e 4a 03
00000360 97 20 5a b9 bc 8a 46 42 2a d1 2f ca 23 a6 52 46
00000370 39 1d 2a 4b ec 89 39 c9 87 5d 5d ad 2a 3a 55 11
00000380 54 31 11 e3 80 52 3b 9a 17 0c 0c 0c 0c 0c 0c
00000390 2a 19 02 81 81 00 e4 1e 23 ff e2 7e
000003a0 54 19 02 81 81 00 e4 1e 23 ff e2 7e
000003b0 c3 c2 99 65 8c e7 db fa d7 0d c5 ab aa 23 66 b6
000003c0 1b 44 e7 86 2a 52 90 b6 a5 20 87 75 c1 da 19 da
000003d0 27 68 8a 0a 16 3b b7 eb 57 d1 94 21 aa 75 b2 3f
000003e0 77 01 02 12 c3 4a 8a 7b 24 13 78 33 67 17
000003f0 13 ba a2 0f 53 56 75 e5 e4 a1 3e 13 82 e0 17 91
00000400 01 ab 1a 81 e3 a2 aa 71 72 4b d7 88 2f 65 64 37
00000410 8e 1f 12 14 5a 75 18 07 0c 0c 0c 0c 0c 0c
00000420 48 4e a1 3a 98 5b 02 81 81 00 e4 1e 23 ff e2 7e
00000430 5e 64 c3 7b b7 ab 51 d2 80 02 75 68 13 87 11 24
00000440 8c 3a 2b b5 43 20 8a 08 ab 07 66 99 8b 33 3c
00000450 0b 04 d5 b7 e1 7b 88 2a 5c 99 08 99 04 0a c0 8d
00000460 3b 56 42 c3 55 55 47 d5 42 f7 07 99 aa 2a 3b 26
00000470 17 35 44 79 24 7d 6a 1a db b7 c9 e3 68 4f
00000480 88 9b 50 09 9a 7d dc 28 71 9b 77 47 7d ad b2
00000490 75 b3 ea 16 c0 ef e0 25 18 94 9a 4b 43 9d bf 29
000004a0 02 13 0c ac 7a 09 09 17 87 34
000004aa -----END RSA PRIVATE KEY-----
```

[illegible]

- Angriff gegen PKCS#1 (Version 1.5) – Bleichenbacher Attack
 - Chosen-Ciphertext-Angriff
 - Idee: PKCS#1 definiert **0x00 0x02** als Nachrichten-Prefix bei Padding
 - Einsatzbeispiel:
Eavesdropping des Austauschs eines Session-Keys in TLS
(Session-Key < RSA-Schlüssel)
 - Methode:
 - Ciphertext anpassen, bis der Server beim Entschlüsseln einen Erfolg meldet
 - Session-Key aus mitgelesenem Datenpaket entschlüsseln
 - Komplexität: 30.000 – 130.000 Versuche bis zum Erfolg

- Standardisierung des DH-Schlüsseltauschs (vgl. TLS-Handshake Phase 1)
 - Generierung der DH-Parameter durch eine dritte Instanz
 - Wahl einer Primzahl p
 - Wahl der Basis g mit $1 < g < p$
 - Optional: Wahl des Längen-Parameters l des Secrets
(Wahl der Primzahl oder Länge des Secrets als Performance-Setting)
 - DH – Phase I
 - Erzeugung des Secrets pro Partner und Octet-to-String-Konvertierung
 - DH – Phase II
 - Exponentiation des Partner-Secrets => Ableitung des PSK

Historie

- Version 2 im September 2000
- NIST-Empfehlung im Jahr 2010
- Update zu RFC 8018 im Januar 2017
- Password based Cryptography Specification
 - Ableitung eines kryptografischen Schlüssels aus einem Passwort
 - Key-Derivation-Function (KDF) basierend auf kryptografischen Hashes
 - Verwendung eines Salts
 - Variation in der Anzahl der Iterationen

Verwendung eines Salts

- $DK = KDF(P, S)$; DK = Derived Key, P = Password, S = Salt
- Vorteile:
 - Salts erschweren die Vorberechnung sogenannter Rainbow-Tables für Hash-Verfahren
 - Salts verringern die Wahrscheinlichkeit von Kollisionen des Hash-Verfahrens
- Empfehlungen:
 - Zufällige Wahl des Salts mit einer Länge von mindestens 64 Bit (8 Zeichen)
 - Optional: Hinzufügen einer festen Bytefolge (z.B. des Zwecks des DK)
 - Keine versehentliche Nutzung desselben Salts für unterschiedliche Zwecke

Anzahl der Iterationen

- $DK = KDF(KDF(KDF(\dots KDF(P,S))) \dots)$
- Vorteile:
 - Erhöht die Komplexität der Berechnung eines Schlüssels (Stichwort: Bruteforce)
 - c Iterationen erhöhen die Sicherheit des Schlüssels um $\log_2(c)$
- Empfehlungen
 - NIST: [...] A minimum iteration count of 1,000 is recommended. For especially critical keys, or for very powerful systems or systems where user-perceived performance is not critical, an iteration count of 10,000,000 may be appropriate.

Key Derivation Functions

- PBKDF1 und PBKDF2 (Password Based Key Derivation Function)

DEPRECATED

- Ablauf:
 1. Wahl von Salt s und Anzahl der Iterationen c
 2. Wahl der Länge des resultierenden Schlüssels (DKLen)
 3. Führe KDF^c mit (P, S) durch und
 4. Gebe den resultierenden Schlüssel (Derived Key) aus

PBKDF2

- KDF basiert auf Pseudozufallsfunktionen (PRF; z.B. Hash oder HMAC)
- DK wird aus allen Zwischenergebnissen der PRF erstellt (concat)
 - Dabei ist hlen die Länge des resultierenden Hashes
 - $DK = T_1 \parallel T_2 \parallel \dots \parallel T_l$ mit $l = DKLen / hlen$
 - $T_x = F(P, S, c, i), i = x$ und $F = U_1 \oplus U_2 \oplus \dots \oplus U_c$ mit
 - $U_1 = PRF(P, S \parallel INT(i))$
 - $U_2 = PRF(P, U_1)$
 - ...
 - $U_c = PRF(P, U_{c-1})$

$\oplus = \text{XOR}$

REKURSION VERHINDERT PARALLELISIERUNG!

PBES1/PBES2 (Password Based Encryption Scheme)

- Kombiniert die Verschlüsselung einer Nachricht m mit dem Schlüssel aus der entsprechenden PBKDF1/2
- Verschlüsselung mit beliebigem Verfahren, z.B. Blockchiffre (DES oder RC2)
- Parameter (Beispiel für $DKLen=32$):
 - $K = DK_{0..15}$
 - $IV = DK_{16..31}$
 - Padding (aufgefüllt mit 01,0202,..., 08^8 , ... nach Anzahl benötigter Zeichen)

PBMAC (Password Based Message Authentication Code)

- MAC-Berechnung analog zu PBES basierend auf unterliegender MAC-Funktion

Definiert Erweiterungen zu X.509-Zertifikaten

- Aktuelle Version 1.5 (November 1993)
- Erlaubt das (beliebige) Hinzufügen von Informationen zu X.509-Strukturen
 - Genutzt etwa für E-Mail-Adressen
- Definiert die Zertifizierung (Signatur) hinzugefügter Informationen
- **DEPRECATED** seit X.509V3

Als RFC 2315 veröffentlicht im März 1998

- Syntax für Daten, auf denen Kryptografie angewandt wurde
- Grundlage für die Verschlüsselung und Digitale Signatur von Nachrichten
- Basiert auf Zertifikaten aus Public-Key-Infrastrukturen (PKI)
- Anwendung: S/MIME, OpenSSL-Verschlüsselung, PKCS#12
- Definiert sechs Inhaltstypen:
 - Data
 - Signed Data
 - Enveloped Data
 - Signed-and-enveloped data
 - Digested data
 - Encrypted data

- Basistyp (Data)
 - Enthält beliebige Daten (Octet-Strings)
- Erweiterte Typen (die fünf anderen)
 - Definiert ASN1-Syntax für die entsprechenden Anwendungsfälle
 - Für Signaturen z.B. Informationen über den „Signer“
- Verschlüsselung ist kompatibel mit der von PEM, falls
 1. Content-Info ist vom Typ Data
 2. Kryptografische Parameter sind kompatibel zu PKCS#1 RSA-Encryption
- Padding: Analog zu PKCS#5 (allerdings für Blockgrößen bis 255 Byte)

PKCS#8 – PRIVATE KEY INFORMATION SYNTAX

Definiert die Speicherung (allgemeiner) privater Schlüssel

- Motivation: PKCS#1 speichert nur RSA-Schlüssel
- RFC 5208 von Mai 2008
- Datentypen:
 - Privater Schlüssel
 - Verschlüsselter privater Schlüssel (mit PKCS#5 (PBES) verschlüsselt)
- OpenSSL ersetzt **genrsa** mit **genpkey**
\$> openssl genpkey -algorithm rsa

PKCS#8 – PRIVATE KEY INFORMATION SYNTAX

Definiert die Speicherformate

- Motivation: RFC 5208

- RFC 5208 von

- Datentypen:

- Privater Schlüssel

- Verschlüsselter privater Schlüssel

- OpenSSL ers

\$> openssl g

```
[matze@tschita] /tmp $ openssl genpkey -algorithm rsa
...+++++
-----BEGIN PRIVATE KEY-----
MIIEvgIBADANBgkqhkiG9w0BAQEFAASCBAgEAAoIBAQCgKL51+1n0QOMN
Cc4Tq7H14pYokets9V2mJw+MI3mGIoQesEGPh0vRYV1xN8vwC3OzXhs2TkVIBDd
q2H0kQ+aUKU8AC07yy52USfSMNKnJT+sT2H9Gvtiy7HF8fGEeK+AuLJyb3saVzvK
Jwct2D0dx+mZsP04iPhYHI4Klte2pgAyXNBG/6kFrBBBU9ZoyqjyWHcMNlLVyxR
fiCnGkeySIP+EdK2rnb/MBIznvh8jCxCXhojEZwf6FqjU0to3B6XWadNS4N5n13xN
R22rFlPnCcQj3FalqsupzR4P6vuY164JPIiOCmITFbK2fV58nqDcHxJs40MkiJ6EX
Uc01YBZtAgMBAAECCgEBAIHd7J6wJHBGlmzHZntmjsaI096Ij8TJYBYI88p5JvS
2PZZTI+o0N1R6YaoC4D0iDhkkGQsqzLyDSrwkFjSVqh3FRR1wDe+UvmlDGjulaBa
H6sEIXwsCLiVIFK0v3T+tg/VEzyfEF5nZV+hDagVoqTJOK0aSFjRdy1a2gHUEIGs
AYCD1tI3seM/95moS4yW2IaEjb/6Gj3/bkrdJz8DQd7W6a13WUezFrnb/DqBuzU4
TJ3FdIqRJ0UNy8CxrZKGAxFl15rbvnka42Df84Y1x0cmUA4x6nGKA6r8ZkVcIQS0
JiNBD0KR55UkxIRKcxGNsVFUBML90ZhlD9VE40cNQGECgYEA0Vuv8qGrnad7jY7m
YS0udhidMBkxm7hV/8Ffq5zdiaJGFwDbcBhCHx5HLY2+//suaPvmqWWVhn5ktELS
y3XUYiFxxQpbToZtbOcHn0GhZIdEvBsTeJhcnsnISu3Xld17OmQDsuUzmt0BceNWX
JKoAvYxbmJ4I4jYgyc4spBPHGc0CgYEAw9cXw9byGGFT8L3f1oWnfKs/vIcDJKiP
OuOAlqBlKpSLM17UHXG5/1PmLbcHCq6QBz4eAHnTmOyIxZQTKsxt6evr1fgA3B3p
6uMS5irvO4rbWS4kM5ce+WRjRn8sJaF6tDhNfXZM/ek1AO11fSPzCX2K7y90YjgX
X5blQLsTteECgYEAqDYdr4LYvaDe2m9ECQkJJlroKLEs35apHAqJk2hgh6pYMCg3
OqvAZtI9F9h1GwDxiBuUQ/NRid9vvrEzTL5BRaRnjYdTL+bFRqcplm81joIEhw0x3
223cwrX3WlQ2JedqsGrPjh9ZP6prtzhfCnY0Jaa75XgKyrYJWG0qDIRPj1UCgYBw
kSQB130Ucm0zRZBx2CUrTau3C8vMuG6LpNT2v38m+HcCShEuU35burflsvyDMU9s
0li4Z84ualc7ah/NJEKRYKAQPxsPaspXzunADxGvoeLu4czBMq4FH6TIkrSndKf5
CkvPb75VfESRIeNiCNXbsKO5VaeLpT4AhbvdsxufQQKBGA4imlKQBZG+QPv/D32D
PEe+T7EVwh0+zORL1zSacRJYbzh4lEQslmMLiH9awtOS5jPMMuQxaj+eq6ZCsM0
3dKHZJ9W3zCW6PvGkCWavwymTGInc64PcRc4XuM2iBqWJJDz7WsPxUDrMkAVvfET
oyY9tgfflZoE8JW1JqOewBj
-----END PRIVATE KEY-----
```

(chlüsselt)

PKCS#8 – PRIVATE KEY INFORMATION SYNTAX

Definiert die Spei

```
[matze@tschita] /tmp $ openssl genpkey -algorithm rsa
.....+++++
-----BEGIN PRIVATE KEY-----
[matze@tschita] /tmp $ openssl genpkey -algorithm rsa | openssl asn1parse
.....+++++
0:d=0  hl=4  l=1214  cons: SEQUENCE
4:d=1  hl=2  l= 1    prim: INTEGER           :00
7:d=1  hl=2  l= 13   cons: SEQUENCE
9:d=2  hl=2  l= 9     prim: OBJECT            :rsaEncryption
20:d=2  hl=2  l= 0    prim: NULL
22:d=1  hl=4  l=1192  prim: OCTET STRING      [HEX DUMP]:308204A40201000282010100B4E891757D491E97F1B275ACAE17EDDF071E04EB400C
5F10C5C2E8A916FB87DF98FA9623586C88D4E59FB5C40E99A76CA7A026D877365333123A40002DCEA837B99FA1D9F05D2505E8550B1ED763B4BF1BF817C7BA1
635AF561B4F86AC9B9B9990671A7F56B1CC0259A10796FFC22390ED03BC9529E23F602CEFFFE3C35142EF8BB19BC9120DE40CB0713018C7B698D3CB7B1F584C
463DE5A9932394F36D468BC2FA00123A18B1AEA2D627604EDE0FA7571AAFC2A643CF803745F02030100010282010100A5B0BC48D74EB8EE8137E9FF16146490
94B2BC914B9A99A284A78918F0CB5A3646F4053725BF24C0071AF2D533BEC0F23BD033990D8DEB0A4FDF4CBD95E1B0086C9442380F05CDF23E1FEC208A54D74
FA7438699C57EC6B66257FE3C658B99981566662E3E8FDA162782B4D5E60C3DCA1D93557BB7D877BE62CB40343B4F59FC3823A29C024B0C358A6460B0D60F2
31DBED70315F92C912A79275931FDA529A7A6F045DFE3860749ABCB79C9642117E9CED1BCF7C8731A69D48102818100DD490FF7386B4CC844C2A869A646EFC
8D79AFD9CE80E4D270C2DBB6AB98BB079360B64C6C2B04134185B85B0D82CEB8CABE75159E9EF5787AA7589298AD0EAA5575C6E857E4D68431BE45807BFAEB
6B308702818100D149F2785BD77F524BDA934A46F3F2A56D90ACA0F0435C332DC9F5BECF05ED2C5B49AD7E007474571F20A54A68449EE133DCB074B42D7B17D
3D89E197E9AB14F071ADB537B1FAAA106B7238CE64DAD0111FA3639B36649EDC7272F0A7E4A9ACD3CCE5A5BD78AC34B6902818100A7E3B60C50AF003B7607E
AC5506B7ABCFB1553805CC4BF71343634DAEE6F00696DD8021593716D16AA58DC245D197094CA79D4E45B204CBF6A56462B3629C94C3BA6A5438268CB355E72
BFEFE839096BAE30281804E92507157298427454AFDD8F8E244CA4E63EE2B4D883C690A5BB3E19A4B434B4FCA4D53EC9FCBBD99760C17EF2533F0A023CE2B4
C4B653B6953065A9DE318AD5C4036B47E20F90FC90CCC4CFA04654E767D3D1458E6905A201FCCF4B4D5810FBFE8B55A2A423F61028180252C6C551310AD892
405EB250E27AE29CC636FE2A1AE836F453C2244ED07161E2898577CACE8F03E2E701870FB0845C8F18F44A9DB365AD8F3D493DEB82ACC41EAE1FAAF85E143D
88549884E1C27E3D81E9
0114264a1c /an/ NOEKRIKAWKXSPASPAZUNADKXVOELU4CZBMQ4FNOTIKRSHUK15
CkvPb75VFESRIeNiCNXbsKO5VAeLpT4aHbvsvdxufQQKBgA4imlKQBgz+QpV/D32D
PEe+T7EVWh0+zORL1zsacRJYbzh4lEQslmMLiH9awtOS5jPMMuQxaj+eq6ZCsM0
3dKHhZJ9W3zCW6PvGkCWavwymTGInc64PcRc4XuM2iBqWJJDz7WSPxUDrMkAVvfET
oyY9tgfflZoE8JWlJqOeWBj
-----END PRIVATE KEY-----
```

PKCS#8 – PRIVATE KEY INFORMATION SYNTAX

Definiert die Speicherung (allgemeiner) privater Schlüssel

```
[matze@tschita] /tmp $ openssl genkey -algorithm rsa | openssl asn1parse
.....+++++
.....+++++
0:d=0  hl=4  l=1214  cons: SEQUENCE
4:d=1  hl=2  l= 1    prim: INTEGER           :00
7:d=1  hl=2  l= 13   cons: SEQUENCE
9:d=2  hl=2  l= 9     prim: OBJECT            :rsaEncryption
20:d=2  hl=2  l= 0     prim: NULL
22:d=1  hl=4  l=1192  prim: OCTET STRING      [HEX DUMP]:308204A0201000282010100B4E891757D491E97F1B275ACAE17EDDF071E04EB400C
5F10C5C2E8A916FB87DF98FA9623586C88D4E59FB5C40E99A76CA7A026D877365333123A40002DCEA837B99FA1D9F05D2505E8550B1ED763B4BF1BF817C7BA1
635AF561B4F86AC9B9B9990671A7F56B1CC0259A10796FFC22390ED03BC9529E23F602CEFFFE3C35142EF8BB19BC9120DE40CB0713018C7B698D3CB7B1F584C
463DE5A9932394F36D468BC2FA00123A18B1AEA2D627604EDE0FA7571AAFC2A643CF803745F02030100010282010100A5B0BC48D74EB8EE8137E9FF16146490
94B2BC914B9A99A284A78918F0CB5A3646F4053725BF24C0071AF2D533BEC0F23BD033990D8DEB0A4FDF4CBD95E1B0086C9442380F05CDF23E1FEC208A54D74
FA7438699C57EC6B66257FE3C658B99981566662E3E8FDA162782B4D5E60C3DCA1D93557BB7D877BE62CB40343B4F59FC3823A29C024B0C358A6460B0D60F2
31DBED70315F92C912A79275931FDA529A7A6F045DFE3860749ABCB79C9642117E9CED1BCF7C8731A69D48102818100DD490FF7386B4CC844C2A869A646EFC
8D79AFD9CE80E4D270C2DDB6AB98BB079360B64C6C2B04134185B85B0D82CEB8CABE75159E9EF5787AA7589298AD0EAA5575C6E857E4D68431BE45807BFAEB
6B308702818100D149F2785BD77F524BDA934A46F3F2A56D90ACA0F0435C332DC9F5EBBCF05ED2C5B49AD7E007474571F20A54A68449EE133DCB074B42D7B17D
3D89E197E9AB14F071ADB537B1FAAA106B7238CE64DAD0111FA3639B36649EDC7272F0A7E4A9ACD3CCE5A5BD78AC34B6902818100A7E3B60C50AF003B7607E
AC5506B7ABCFB1553805CC4BF71343634DAEE6F00696DD8021593716D16AA58DC245D197094CA79D4E45B204CBF6A56462B3629C94C3BA6A5438268CB355E72
BFEFE839096BAE30281804E92507157298427454AFDD8F8E244CA4E63EE2B4D883C690A5BB3E19A4B434B4FCA4D53EC9FCBBD99760C17EF2533F0A023CE2B4
C4B653B6953065A9DE318AD5C4036B47E20F90FC906CCC4CFA04654E767D3D1458E6905A201FCCF4B4D5810FBFEF8B55A2A423F61028180252C6C551310AD892
405EB250E27AE29CC636FE2A1AE836F453C224D4ED07161E2898577CACE8F03E2E701870FB0845C8F18F44A9DB365AD8F3D493DEE82ACC41EAE1FAAF85E143D
88549884E1C27E3D81E9
```

- Also auch Elliptische Kurven als Schlüsselmaterial möglich!

\$> openssl genkey -algorithm EC -pkeyopt ec_paramgen_curve:ED25519

PKCS#8 – PRIVATE KEY INFORMATION SYNTAX

Definiert die Speicherung (allgemeiner) privater Schlüssel

- Motivation: PKCS#8

```
[matze@tschita] /tmp $ openssl genpkey -algorithm ED25519
-----BEGIN PRIVATE KEY-----
MC4CAQAwBQYDK2VwBCIEIFWco1tk9oKqtbkVr7786OSDi4vk9ePJt0Q4StALZeFF
-----END PRIVATE KEY-----
```

- RFC 5208 von Mai 2008

```
[matze@tschita] /tmp $ cat ec.priv | openssl asn1parse
0:d=0 hl=2 l= 46 cons: SEQUENCE
2:d=1 hl=2 l=  1 prim: INTEGER           :00
5:d=1 hl=2 l=  5 cons: SEQUENCE
7:d=2 hl=2 l=  3 prim: OBJECT           :ED25519
12:d=1 hl=2 l= 34 prim: OCTET STRING    [HEX DUMP]:042079E877D76EA4344CA63E6B716EDEF87449898E39E6D524EE7E7C70CF2B0AC0F4
```

- Verschlüsselter privater Schlüssel (mit PKCS#5 (PBES) verschlüsselt)
- OpenSSL ersetzt genrsa mit genpkey
 - \$> openssl genpkey -algorithm rsa
 - Also auch elliptische Kurven als Schlüsselmaterial möglich!
 - \$> openssl genpkey -algorithm EC -pkeyopt ec_paramgen_curve:ED25519

Vielen Dank für die Aufmerksamkeit!

Fragen?

Nächste Vorlesung:

- Montag, 2. Mai 2022

Nächste Übung:

- Dienstag, 26. April 2022 – 16 Uhr
- Abgabe des Übungszettels bis morgen – 16 Uhr

BEVOR ES WEITER GEHT...

- Pizza-Challenge

- RSA-Dokumente:

<http://ftp.sunet.se/mirror/archive/ftp.sunet.se/pub/security/docs/PCA/PKCS/ftp.rsa.com/>

PKCS#9 - SELECTED ATTRIBUTE TYPES

RFC 2985 (November 2000)

Definiert zwei übergeordnete Objekte zur Verwendung in anderen Standards

- PKCSEntity
 - Allgemeines Objekt zur Speicherung von PKCS-Standardattributen
 - pKCS7PDU
 - userPKCS12
 - pKCS15Token
 - encryptedPrivateKeyInfo

PKCS#9 - SELECTED ATTRIBUTE TYPES

RFC 2985 (November 2000)

Definiert zwei übergeordnete Objekte zur Verwendung in anderen Standards

- NaturalPerson
 - emailAddress
 - unstructuredName
 - unstructuredAddress
 - dateOfBirth
 - Gender
 - countryOfCitizenship
 - countryOfResidence
 - pseudonym
 - serialNumber
 - ...

Verwendet etwa in PKCS#7, PKCS#10, PKCS#12, PKCS#15

– auch für kryptografische Informationen in Directories (LDAP)

Historie

- RFC 2314 – v1.5 (März 1998)
- RFC 2986 – v1.7 (November 2000)

Definiert die Syntax eines Zertifizierungsantrags (Certificate Signing Request, CSR)

- Antragsteller erstellt CSR
 - CertificateRequestInfo mit Attributen: Version, Subject (Name des Antragstellers), SubjectPKInfo (Public Key), ... (weitere Attribute etwa aus PKCS#9 – z.B. Challenge-Password zum späteren Widerrufen des Zertifikats)
 - Antragsteller signiert den CSR mit dem Private Key
 - CertificateRequest mit Attributen: CertificateRequestInfo, SignatureAlgorithm, Signature

```

1996:d=4 hl=2 l= 9 prim: OBJECT                               :rsaEncryption
207:d=4 hl=2 l= 0 prim: NULL
209:d=3 hl=4 l= 271 prim: BIT STRING
0000 - 00 30 82 01 0a 02 82 01-01 00 db 91 9c bf 82 f6 fe .0
0010 - 5a f1 28 ad a4 ee 49 b6-d1 f0 3a 60 02 61 bb 9d Z(.I.I.a.
0020 - 03 98 0e da e6 52 5b c5-b6 a0 ed d5 a7 0c e7 97 .R|.
0030 - ad 48 3d 0b 02 59 a9 b2-12 25 2a 3d 63 64 5b b7 .H.=.Y.%=cd[.
0040 - 94 d5 70 73 df 86 85 40-e2 ac 3b f3 00 1e 9c c4 .ps.].
0050 - 14 f9 19 67 65 78 93 b5-15 f2 26 5e 2c f6 55 5e .gex.].^UV
0060 - 15 19 b5 a4 f1 20 14 63-6e 13 f7 41 84 5b 89 7e .df.n.A[.
0070 - e2 79 8a f9 4b f2 98 a5-14 49 c9 01 94 55 e8 73 .y.K.I.U.s
0080 - 15 f2 12 64 66 0a 6e 17-5a 3d 33 73 1b al 35 44 .df.n.Z<3s.5D
0090 - 81 6c ff 03 7d 86 09 37-36 3f c1 b0 79 2f 62 63 .l|.76.y/bc
00a0 - 22 38 55 49 f6 f2 85-c8 3a aa d6 34 4b 75 2d .SUI.]=4ku-
00b0 - 26 ff 02 19 f2 cb 39 b6 4f 17 89 90 1c .tq.9.L
00c0 - 9f 36 a1 0e 43 ad 59 b2-94 a4 1b 35 cd 66 45 9e .6.C.Y.L.E.
00d0 - 89 3d 85 e0 25 5f fe 9e-b2 45 12 25 0b c0 05 fa .6.C.Y.E.%
00e0 - d8 44 9c 24 ae 5c 58 9b-e2 5f 8e 6d 79 db 51 43 .6.$h.E.%y.QC
00f0 - 64 63 9b 2d 44 02 5c 56-f9 a0 10 6b bd 5d 54 a0 .de.D.V.[.i]T.
0100 - e5 4f f9 48 29 7d 84 bb-cc 91 02 03 01 00 01 .o.H])
484:d=2 hl=2 l= 67 cons: cont [ 0 ]
486:d=3 hl=2 l= 31 cons: SEQUENCE
488:d=4 hl=2 l= 9 prim: OBJECT                               :unstructuredName
499:d=4 hl=2 l= 18 cons: SET
501:d=5 hl=2 l= 16 prim: UTF8STRING                          :Uni Bonn Company
519:d=3 hl=2 l= 32 cons: SEQUENCE
521:d=4 hl=2 l= 9 prim: OBJECT                               :challengePassword
532:d=4 hl=2 l= 19 cons: SET
534:d=5 hl=2 l= 17 prim: UTF8STRING                          :ChallengePassword
553:d=1 hl=2 l= 13 cons: SEQUENCE
555:d=2 hl=2 l= 9 prim: OBJECT                               :sha256WithRSAEncryption
566:d=2 hl=2 l= 0 prim: NULL
568:d=1 hl=4 l= 257 prim: BIT STRING
0000 - 00 12 ae af 4d e5 87 85-1b ae bc f0 9e f1 2b 58 .M.....+X
0010 - 7a 16 bf ab 40 f1 69 09-5c f4 77 38 84 fb 9e 85 z..q|\w8.
0020 - 45 39 22 3b 19 6c 07 f2-44 a0 ed 0e cd 33 bd 8a E9"=1.D.3.
0030 - a4 df cb 5b 23 08 c5 04-86 40 43 c1 3f 61 ff 38 .|(#.C.a.8
0040 - 2f be 69 f2 92 40 0f 6c-58 70 4a ef ff c2 91 30 /|.IX.J.0
0050 - 56 da bd cd 03 7d 0e 98-81 80 e5 01 da cc 45 03 V|.].E
0060 - 3a 47 71 91 03 87 a5 14-c6 04 a2 17 16 9d 64 .q.J.J.d
0070 - 56 c6 7d 2c 2c e5 45-48 27 44 42 8b 75 al VD. eHu.B.u
0080 - bb 21 3f 3f 84 2c c5-ee af 1b 4a ce 25 f5 fa .|??.D.
0090 - 2d d1 40 91 5e 24 74-b3 af ff 6d al 54 e4 a2 .inst..D.
00a0 - f0 da 47 2c 28 96 dd 99-c0 ac 91 96 be 8d 2d 31 .inst..-i
00b0 - b7 ab 73 0b al 4f ae 63-0f 20 41 75 76 b3 2e 54 .O.c.u.Av.T
00c0 - 68 cd 21 5d 76 96 34 b6-b5 4d 3b 4e 5e 2e 04 h.]-4..NV.J.
00d0 - ff 55 a2 d7 87 14 1a-19 09 a5 3f bb 0f 70 fb 15 U..J..p.
00e0 - f6 a0 ef 1f 91 08 e6 b0-01 c8 0e 99 0f 86 f1 59 .|.3..b.
00f0 - 82 bb 4c f9 e6 b2 a3 33-a9 af eb 15 f3 62 f1 bb L.
0100 - 4c

```

PKCS#11 – CRYPTO TOKEN INTERFACE (CRYPTOKI)

Kein RFC (Version 2.2 veröffentlicht von RSA Laboratories)

Pflege des Standards übergeben an die “Organization for the Advancement of Structured Information Standards” (OASIS)

- OASIS PKCS#11 Technical Committee

API zur Verwendung von Crypto Token

(Smartcards, Cryptokey, Yubikey, Hardware Security Modules (HSM) , ...)

- ANSI C (1990) Standard
- Abstrahiert das zugrundeliegende Hardware-Token für die Anwendung(en)

PKCS#11 – CRYPTO TOKEN INTERFACE (CRYPTOKI)

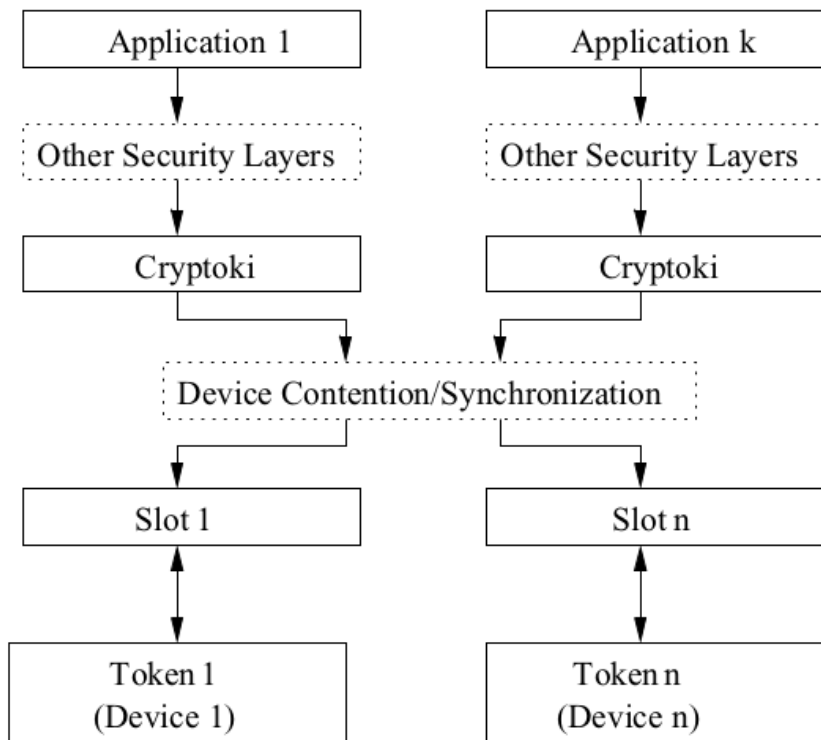
Kein RFC (Version 2.2)

Pflege des Standards
Structured Informa

- OASIS PKCS#11 Te

API zur Verwendung v
(Smartcards, Crypto

- ANSI C (1990) Sta
- Abstrahiert das z



advancement of

s (HSM) , ...)

e Anwendung(en)

Historie

- Version 1 (Juni 1999)
- Version 1.1 (Juli 2014)

Definiert ein portables Format zum Speichern und Transportieren von privaten Schlüsseln, Zertifikaten, Geheimnissen, etc.

Basiert auf Microsofts PFX-Dateiformat (und ist kompatibel dazu)

Verwendet in

Java (Java Key-Store) z.B. Tomcat;

Microsoft: z.B. IIS, Exchange, etc.

Unterstützung aber auch in gängigen Tools (Firefox, Chrome, Thunderbird, ...)

PKCS#12 – PERSONAL INFORMATION EXCHANGE SYNTAX

Lässt sich als “Erweiterung” von PKCS#8 betrachten, gespeicherte private Schlüssel können um weitere Informationen, etwa Zertifikate angereichert werden.

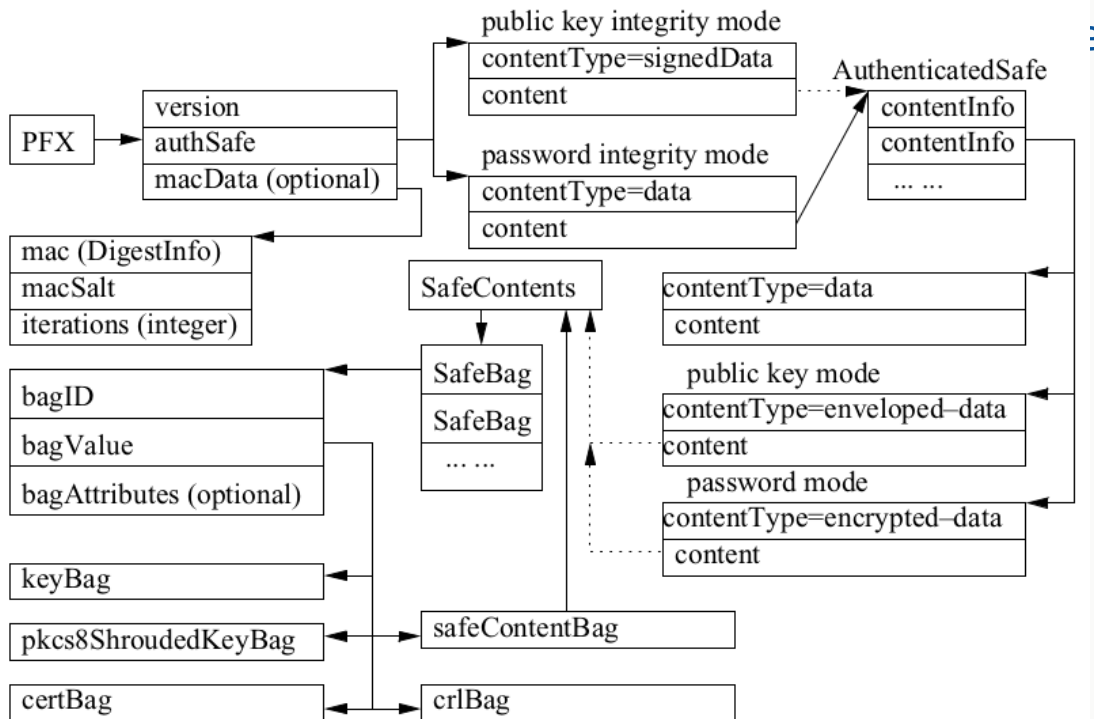
- Verbesserter Schutz der Informationen im Vergleich zu PKCS#8
 - Privacy-Mode / Verschlüsselung mit
 - Passwort
 - Public-Key **EMPFOHLEN**
 - Integrity-Mode / Signatur mit
 - Passwort (Passwort MAC)
 - Digitale Signatur (mit privatem Schlüssel) **EMPFOHLEN**

PKCS#12 – PERSONAL INFORMATION EXCHANGE SYNTAX

Der PKCS#12-Standard ist sehr komplex und erlaubt (fast) beliebige
Verschachtelungen von Schlüsselmaterial

PKCS#12 – PERSONAL INFORMATION EXCHANGE SYNTAX

Der PKCS#12-Struktur Verschachtelung



PKCS#12 – PERSONAL INFORMATION EXCHANGE SYNTAX

Der PKCS#12-Standard ist sehr komplex und erlaubt (fast) beliebige Verschachtelungen von Schlüsselmaterial

In normalen Anwendungsszenarien aber fast ausschließlich ein privater Schlüssel mit entsprechendem Zertifikat (inkl. Zertifikatskette)

Erstellen einer PKCS#12-Datei (Privater Schlüssel + Zertifikatskette)

```
$> openssl pkcs12 -export -in vlnwsi.crt -inkey vlnwsi.key -out vlnwsi.p12
```

Existiert nur als Entwurf (wurde nie veröffentlicht)

Wird nicht weiter gepflegt

Andere Elliptic-Curve-Standards

- ANSI X9.62 (ECDSA)
- NIST FIPS 186-3
- IEEE 1363

PKCS#13 – ELLIPTIC CURVE CRYPTOGRAPHY

Existiert nur als Entwurf (wurde nie veröffentlicht)

Wird nicht weiter gepflegt

Andere Elliptic-Curve-Standards

- ANSI X9.62 (1985)

- NIST

- IEEE

- BSI

- ISO

- SafeCurves



Keine existierenden Dokumente (reserviert)

Zufallszahlen kommen aus

- Pseudo Random Number Generators (PRNG)
 - Deterministische Zahlenfolgen mit guten zufälligen Eigenschaften
 - Guter Zufall, viele Werte
- True Random Number Generator (TRNG)
 - Nutzt (zufällige) physikalische Prozesse / Werte als Zufallswert
 - Echter Zufall, wenige Werte
- Optimaler Weise PRNG mit regelmäßigem TRNG Seed

PKCS#15 – CRYPTOGRAPHIC TOKEN INFORMATION FORMAT

Version 1.1 (Juni 2000; RSA Laboratories)

Kein RFC

Definiert die Datenstruktur auf Cryptotoken

Ziel: Interoperabilität über Software- und Hardware-Grenzen hinweg

Vier unterschiedliche Objekttypen

- Schlüssel
- Zertifikate
- Authentifikationsobjekte
- Datenobjekte

Objekte können privat oder öffentlich sein

Zugriff geschützt durch biometrische oder wissensbasierte Verfahren (z.B. PIN)

PKCS#15 – CRYPTOGRAPHIC TOKEN INFORMATION FORMAT

MF = Master File (root)

DF = Dedicated File (Verzeichnisse)

EF = Elementary File (Dateien)

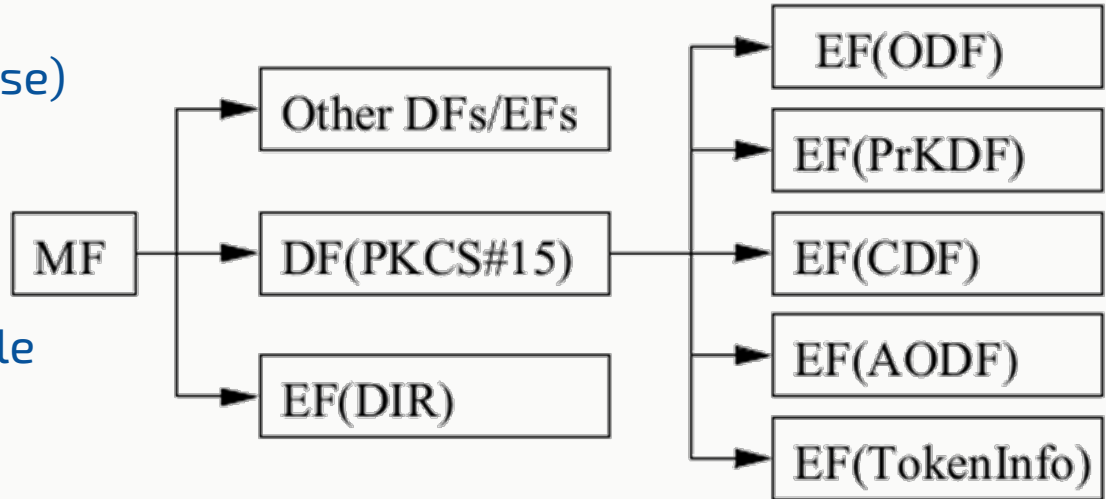
ODF = Object Directory File

PrKDF = Private Key Directory File

CDF = Certificate Directory File

AODF = Authentication Object Directory File

TokenInfo = Informationen über das Token (z.B. Seriennummer, unterstützte Dateitypen, implementierte Algorithmen)



PKCS#15 – CRYPTOGRAPHIC TOKEN INFORMATION FORMAT

\$ > pkcs15-tool --dump

PIN

Private Key

Zertifikat

```
AODF:
  Com. Flags : private, modifiable
  Auth ID   : 01
  Flags     : [0x32], local, initialized, needs-padding
  Length    : min_len:4, max_len:8, stored_len:8
  Pad char   : 0x00
  Reference : 1
  Encoding  : ASCII-numeric
  Path      : 3F005015

PrKDF:
  Com. Flags : private, modifiable
  Com. Auth ID: 01
  Usage      : [0x32E], decrypt, sign, signRecover, unwrap, derive, nonRep
  Access Flags: [0x1D], sensitive, alwaysSensitive, neverExtract, local
  ModLength  : 1024
  Key ref    : 0
  Native     : yes
  Path       : 3F00501530450012
  ID         : 45

X.509 Certificate [/C=BE/ST=...]
  Com. Flags : modifiable
  Authority  : no
  Path       : 3f0050154545
  ID         : 45
```

Digitale Signaturen:

- Ein Unterzeichner signiert eine Nachricht so, dass jeder überprüfen kann, dass die Nachricht nur vom Unterzeichner und sonst niemandem verändert werden konnte
- Message-Digest- und Public-Key-Algorithmen zum Hashen und Signieren des Hashes.
- Spezifikationen
 - Message-Digest-Algorithmen (PKCS#1)
 - Public-Key-Algorithmen (PKCS#1, PKCS#3, PKCS#13 [Entwurf])
 - Algorithmenunabhängige Syntax für digital signierten Nachrichten (PKCS#7)

Digitale Signaturen:

- Ein Unterzeichner signiert eine Nachricht so, dass jeder überprüfen kann, dass die Nachricht nur vom Unterzeichner und sonst niemandem verändert werden konnte
- Message-Digest- und Public-Key-Algorithmen zum Hashen und Signieren des Hashes.

Spezifikationen

- Syntax für private Schlüssel (PKCS#1, PKCS#8)
- Syntax für verschlüsselte private Schlüssel (PKCS#8)
- Methoden zum Ableiten geheimer Schlüssel aus Passwörtern (PKCS#5)

Digitale Umschläge

- Ein Absender versiegelt eine Nachricht, dass nur der Empfänger diese öffnen kann. Die Nachricht ist verschlüsselt mit einem geheimen Schlüssel und dieser Schlüssel ist verschlüsselt mit dem öffentlichen Schlüssel des Empfängers

Spezifikationen

- Algorithmenunabhängige Syntax für digitale Umschläge (PKCS#7)
- Syntax für private Schlüssel (PKCS#1, PKCS#8)
- Syntax für verschlüsselte private Schlüssel (PKCS#8)
- Methoden zum Ableiten geheimer Schlüssel aus Passwörtern (PKCS#5)

Digitale Zertifikate

- Eine Zertifizierungsstelle (CA) signiert eine spezielle Nachricht, die mindestens den Namen und den öffentlichen Schlüssel einer Person enthält so, dass jeder verifizieren kann, dass diese spezielle Nachricht nur von der CA verändert wurde und der öffentliche Schlüssel somit vertrauenswürdig ist. Die spezielle Nachricht wird Zertifikatanfrage (Certificate Signing Request; CSR) genannt und wird mit einem digitalen Signaturalgorithmus signiert.

Spezifikationen

- Algorithmenunabhängige Syntax für Zertifikatanfragen (PKCS#10)
- Syntax für öffentliche Schlüssel (PKCS#1)
- Spezifische Signaturalgorithmen (PKCS#1)

Schlüsseltausch

- Zwei Kommunikationspartner einigen sich auf einen gemeinsamen geheimen Schlüssel, ohne vorherige Absprachen. Typischerweise gibt es dafür Algorithmen mit zwei Phasen: Ein Kommunikationspartner initiiert den Schlüsseltausch in der ersten Phase. Anschließend tauschen beide Partner das Ergebnis der ersten Phase aus und berechnen in der zweiten Phase den gemeinsamen geheimen Schlüssel

Spezifikationen

- Algorithmenunabhängige Syntax für Nachrichten zum Schlüsseltausch (PKCS#3)
- Spezifische Algorithmen zum Schlüsseltausch (PKCS#3)

Verschlüsselung und Signatur von E-Mails

- Ursprünglich von RSA Data Security (PKCS#7) entworfen
- Mittlerweile Standard der Internet Engineering Task Force (IETF)
- PKCS#7 wird zu Cryptographic Message Syntax
- Mehrere RFCs mit diversen Weiterentwicklungen
 - RFC 2311 - Version 2 (März 1998)
 - RFC 2633 – Version 3 (Juni 1999)
 - ...
 - RFC 8551 – Version 4 (April 2019)

Zertifikatshandling und Sperrlisten (S/MIME Certificate Handling)

Historie

- RFC 2632 – Version 3 (Juni 1999)
- ...
- RFC 8550 – Version 4 (April 2019)

HIER NICHT WEITER RELEVANT

Schutzziele

- Zurechenbarkeit
- Integrität
- Nicht-Abstreitbarkeit
- Privacy / Datenschutz
- Vertraulichkeit

Zwei unterschiedliche „Güteklassen“ bei der Zertifikatausstellung

- Klasse 1: Die Absender-E-Mail-Adresse wird überprüft
- Klasse 2: Die Person hinter der Absender-E-Mail-Adresse wird überprüft

Best practices

- Verwendung unterschiedlicher Schlüssel für Verschlüsselung und Signatur
- Obwohl eine Nachricht nur für den Empfänger verschlüsselt wird, wird zumeist ein eigenes Schlüsselpaar gefordert.
 - Dieses wird benötigt, wenn der Absender die E-Mail später noch einmal lesen möchte (und sie verschlüsselt im Ordner Gesendet liegt)
- In der Regel werden die Zertifikatinformationen von der CA öffentlich verfügbar gemacht (z.B. über Verzeichnisdienste)

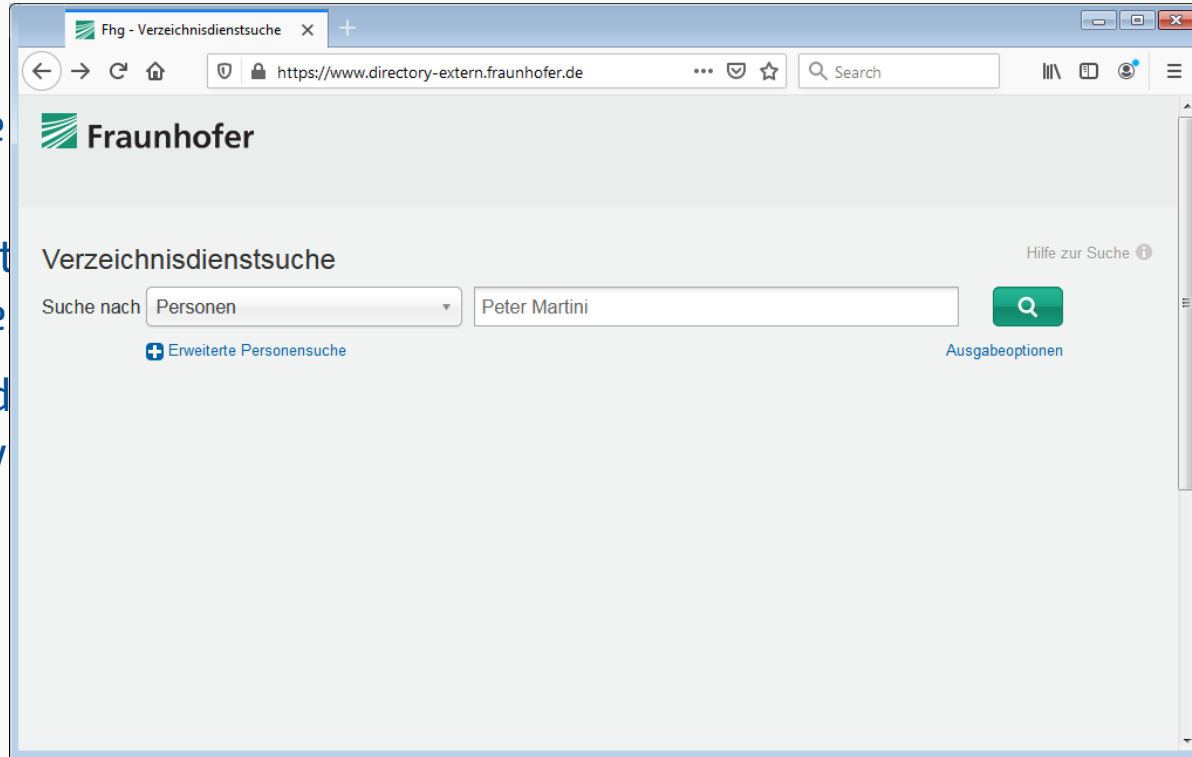
Nachteile von S/MIME

- Keine Überprüfung des E-Mail-Inhalts durch SPAM-Filter oder Virens Scanner möglich
 - Alternativ: Der private Schlüssel der Benutzer (aller?!) muss für die Prüfung hinterlegt werden
- E-Mail-Adressen sind öffentlich verfügbar (Beispiel: Fraunhofer Directory)
<https://www.directory-extern.fraunhofer.de/>

SECURE/MULTIPURPOSE INTERNET MAIL EXTENSIONS (S/MIME)

Nachteile von

- Keine Überprüfung möglich
- Alternative hinterlegen
- E-Mail-Adresse
<https://www.fraunhofer.de>





enscanner

die Prüfung

Directory)

SECURE/MULTIPURPOSE INTERNET MAIL EXTENSIONS (S/MIME)


 **Fraunhofer**

Prof. Dr. Peter Martini 


Institut	FKIE
Standort	Wachtberg
Fax	
E-Mail	peter.martini@fkie.fraunhofer.de
Adresse	Fraunhoferstraße 20 53343 Wachtberg
Zertifikat	<div><div>Gültiges Zertifikat</div><div>Anzeigen</div><div>Download</div></div>

[Zurück zur Suche](#)

SECURE/MULTIPURPOSE INTERNET MAIL EXTENSIONS (S/MIME)

 **Fraunhofer**


Verzeichnisdienstsuche


Hilfe zur Suche 

Suche nach

Personen

michael meier



 Erweiterte Personensuche

Ausgabeoptionen

3 Ergebnisse

Name	Institut	E-Mail
Michael Meier	FKIE	michael.meier@fkie.fraunhofer.de
Michael Meier	IZB	michael.meier@izb.fraunhofer.de
Paul Michael Meier	IPK	paul.michael.meier@ipk-projekt.fraunhofer.de

SECURE/MULTIPURPOSE INTERNET MAIL EXTENSIONS (S/MIME)

- Anwendungsbeispiele
 - Bei der Universität Bonn:
 - HRZ betreibt eine CA für die Universität Bonn über die DFN-PKI
<https://pki.pca.dfn.de/dfn-ca-global-g2/pub/>
 - Nutzbar für Client/User-Zertifikate
 - Nutzbar für Server-Zertifikate (Uni-Dienste)

SECURE/MULTIPURPOSE INTERNET MAIL EXTENSIONS (S/MIME)

- Anwendung Zertifikatdaten
Apply here for a new certificate.

- Bei der U Certificate profile **User**
The chosen "Certificate profile" determines the possible usages of the certificate. ([Description of certificate profiles \[German\]](#))

- HRZ be
- <https://>
- Nutzba
- Nutzba

Neuer Antrag

Create certificate request

The following data is used to create a new certificate request.

▼ Email addresses with domain names from this list can be used without further confirmation. Email addresses with all other domain names must be confirmed separately.

Name (CN)

Enter your first and last name(s) here. Do not use umlauts and diacritics. For group certificates, use prefix 'GRP:' or 'GRP - '. For pseudonym cer

Email

Email address

Organisational unit (OU, optional)

If you specify an organisational unit here, it will be included as OU-attribut in the certificate name.

Namespace (The chosen namespace will be used to complete the final certificate name.)

O=Rheinische Friedrich-Wilhelms-Universitaet Bonn,L=Bonn,ST=Nordrhein- Westfalen,C=DE

SECURE/MULTIPURPOSE INTERNET MAIL EXTENSIONS (S/MIME)

Ihre Daten

Enter further data below. What you enter here will not be found in the certificate.

Revoke-PIN

Revoke-PIN - Confirmation

This PIN is required if you want to revoke your certificate. Please make a note of this PIN.

Personal note (optional)

You may enter an optional note for this certificate application here. The comment will solemnly be saved in your local certificate application data file.

- ☐ I am committed to comply with the regulations contained in [Informationen für Zertifikatinhaber](#).
- ☐ Optional: I agree to the [publication of the certificate](#) with the contained names and e-mail addresses.
You can withdraw this agreement by sending an e-mail to pki@dfn.de.

Next

SECURE/MULTIPURPOSE INTERNET MAIL EXTENSIONS (S/MIME)

- Anwen

- Bei d

- HR

- htt

- Nu

- Nu

Ihre Daten

Enter further data below. What you enter here will not be found in the certificate.

Revoke-PIN

Revoke-PIN - at least 8 character

Revoke-PIN - Confirmation

Please enter the revoke-PIN again to confirm.

This PIN is required if you want to revoke your certificate. Please make a note of this PIN.

Personal note (optional)

You may enter an optional note for this certificate application here. The comment will solemnly be saved in your local certificate application data file.

Personal note (optional)

☐ I am committed to comply with the regulations contained in [Informationen für Zertifikatinhaber](#).

☐ Optional: I agree to the [publication of the certificate](#) with the contained names and e-mail addresses.
You can withdraw this agreement by sending an e-mail to pki@dfn.de.

Next

- Anwendungsbeispiele
 - Bei der Universität Bonn:
 - HRZ betreibt eine CA für die Universität Bonn über die DFN-PKI
<https://pki.pca.dfn.de/dfn-ca-global-g2/pub/>
 - Nutzbar für S/MIME-Zertifikate
 - Nutzbar für x509-Zertifikate für Webserver / Uni-Dienste
 - Volksverschlüsselung
 - Betrieb durch Fraunhofer SIT (Darmstadt)
<https://volksverschlueselung.de/>
 - Klasse-2-Zertifikate (Überprüfung der Person)

SECURE/MULTIPURPOSE INTERNET MAIL EXTENSIONS (S/MIME)

- Anwendungsbeispiele
 - Bei der Universität Bonn:
 - HRZ betreibt eine CA für die Universität Bonn über die DFN-PKI
<https://pki.pca.dfn.de/dfn-ca-global-g2/pub/>
 - Nutzbar für S/MIME-Zertifikate
 - Nutzbar für x509-Zertifikate für Webserver / Uni-Dienste
 - Volksverschlüsselung
 - Betrieb durch Fraunhofer SIT (nicht)
 - <https://volksverschluesselung.de/>
 - Klasse-2-Zertifikate (Überprüfung durch Person)

LÄUFT NUR
UNTER WINDOWS

■ Anwendung

■ Bei der

■ HRZ

<https://www.hrz.uni-bonn.de/>

■ Nutz

■ Nutz

■ Volksw

■ Betri

<https://www.hrz.uni-bonn.de/>

■ Klass

Neues Zertifikat

Vorbereitung

- ☒ Einführung
- ☒ Methode auswählen

Identitätsnachweis

- ☐ Identifikation
- ☐ E-Mail-Adresse wählen
- ☐ Verifikationscode

Beantragung

- ☐ Antrag abschicken
- ☐ Zertifikatserstellung
- ☐ Herunterladen

Identitätsnachweis

Bitte wählen Sie eine der verfügbaren Methoden zum Nachweis Ihrer Identität:

Personalausweis
Identitätsnachweis mit der Online-Ausweisfunktion des Personalausweises

Registrierungscode
Identitätsnachweis mit dem von uns erhaltenen 12-stelligen Registrierungscode

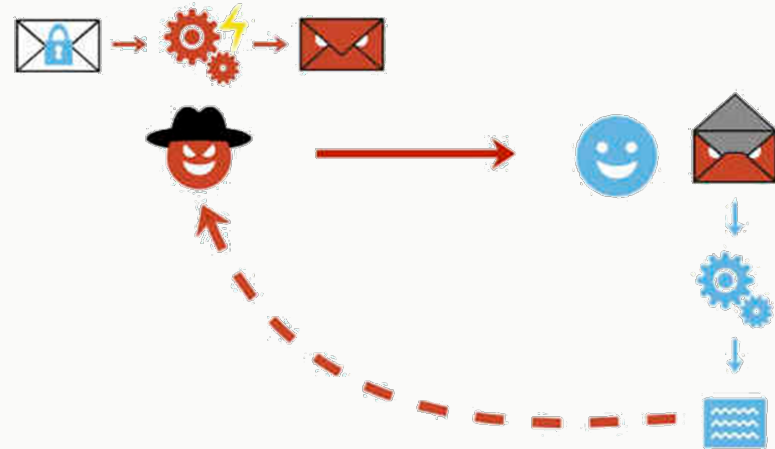
Zertifikatserneuerung
Identitätsnachweis erfolgt mit einem existierenden Authentifizierungszertifikat

Fraunhofer
SIT

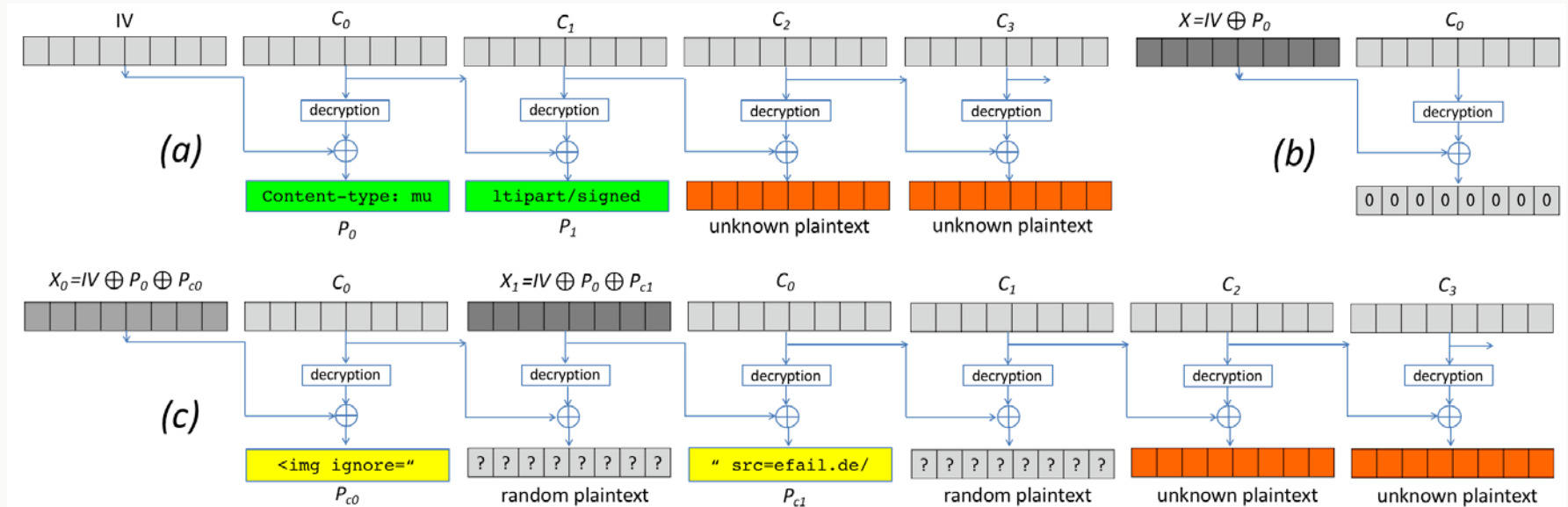
© 2015-2020 Fraunhofer-Institut für Sichere Informationstechnologie

Weiter

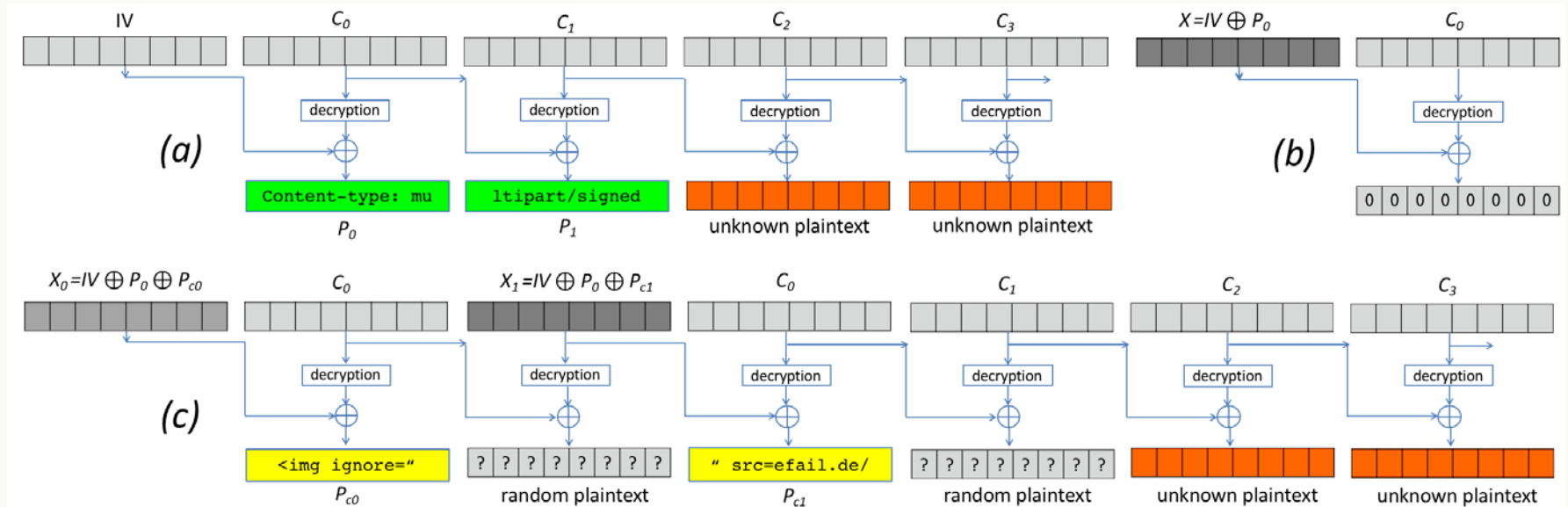
- Angriffe gegen S/MIME
 - E-Fail
 - CVE-2017-17688 , CVE-2017-17689
 - Änderung des Ciphertexts, so dass die Nachricht nach dem Entschlüsseln an den Angreifer gesendet wird
 - CBC/CFB Gadget Attack



- Angriffe gegen S/MIME
- E-Fail



- Angriffe gegen S/MIME
- E-Fail



SECURE/MULTIPURPOSE INTERNET MAIL EXTENSIONS (S/MIME)

```
From: attacker@efail.de
To: victim@company.com
Content-Type: multipart/mixed;boundary="BOUNDARY"

--BOUNDARY
Content-Type: text/html


--BOUNDARY--
```

SECURE/MULTIPURPOSE INTERNET MAIL EXTENSIONS (S/MIME)

```
From: attacker@efail.de  
To: victim@company.com  
Content-Type: multipart/mixed;boundary="BOUNDARY"
```

```
--BOUNDARY  
Content-Type: text/html
```

```

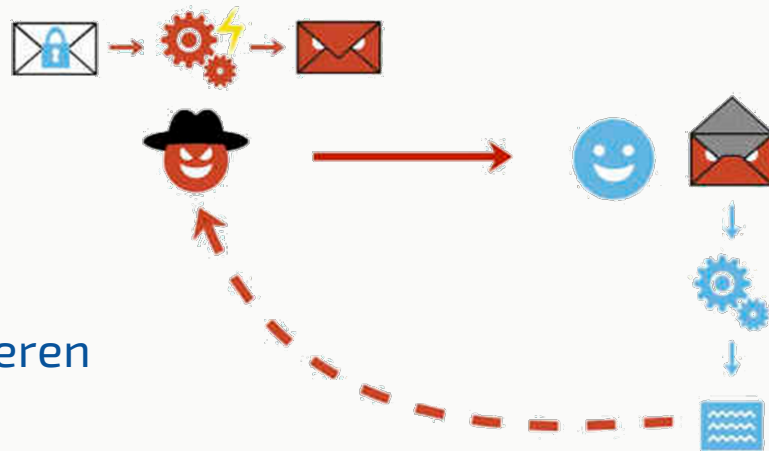
```

```
--BOUNDARY  
Content-Type: text/html
```

```
">
```

```
--BOUNDARY--
```


- Angriffe gegen S/MIME
 - E-Fail
 - CVE-2017-17688 , CVE-2017-17689
 - Änderung des Ciphertexts, so dass die Nachricht nach dem Entschlüsseln an den Angreifer gesendet wird
 - CBC/CFB Gadget Attack
 - Direct Exfiltration
 - Problem: Aktive Inhalte
 - Skripte ausführen / HTML interpretieren
 - Bilder nachladen

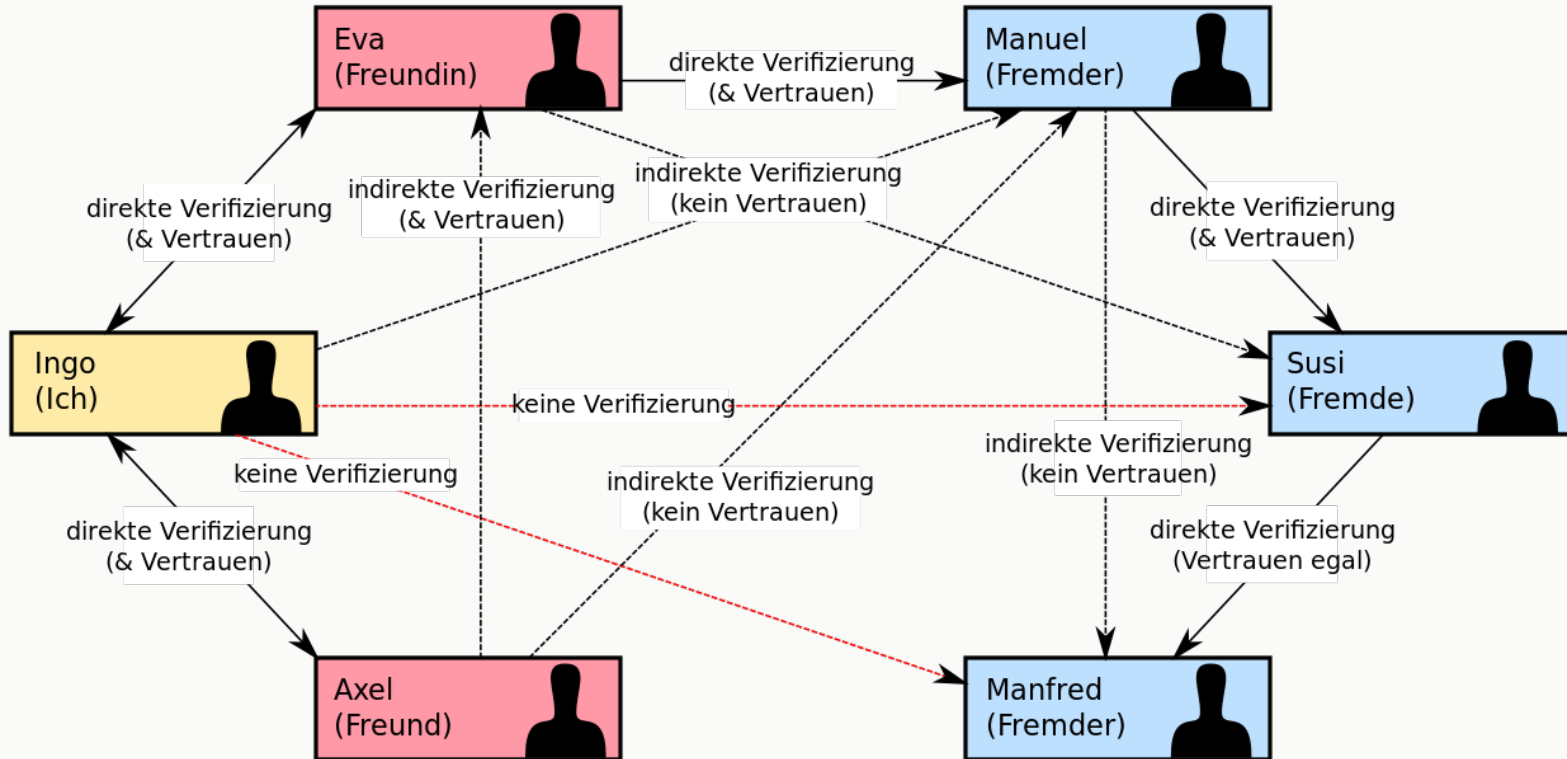


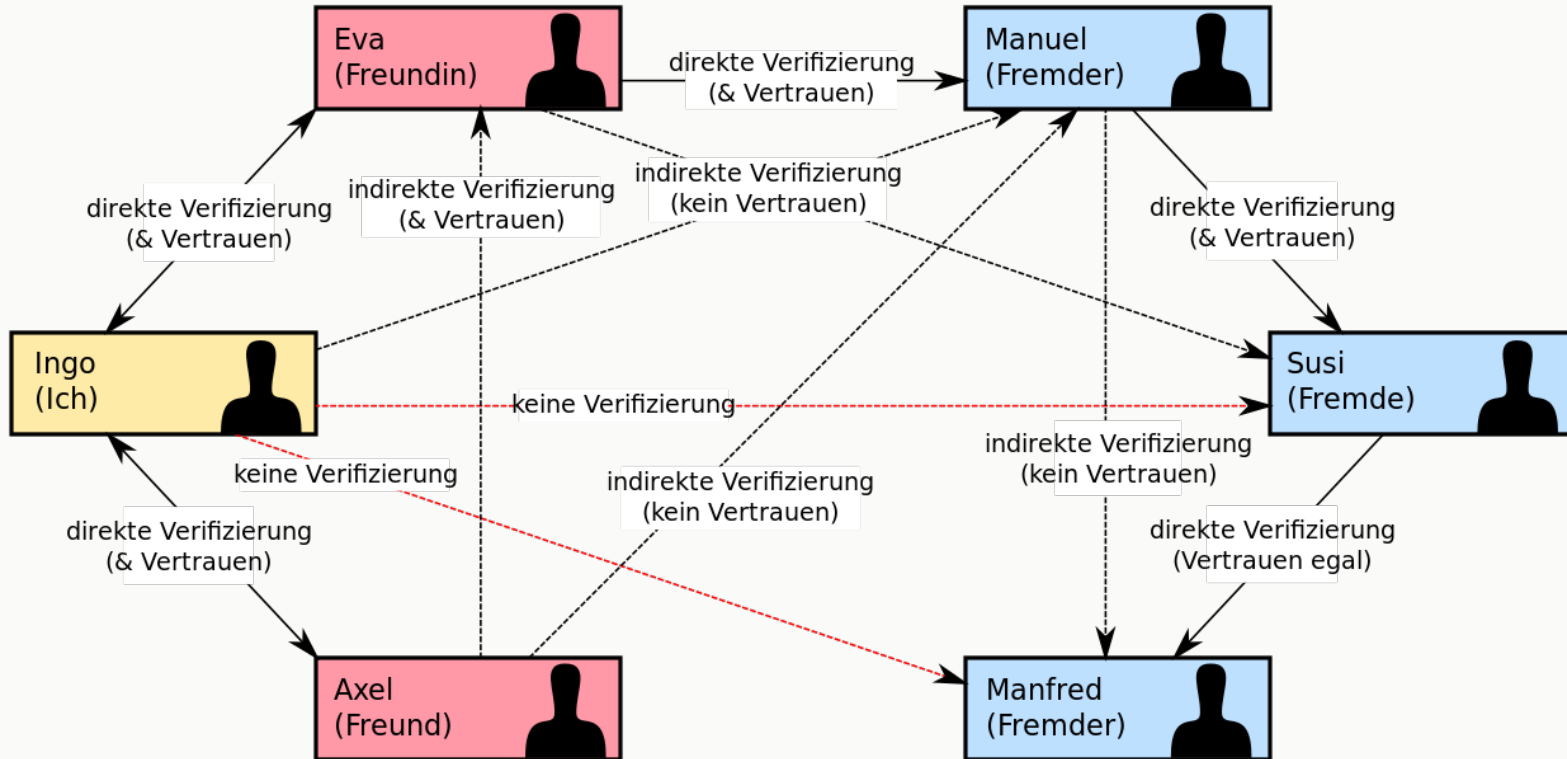
- RFC 4880 (OpenPGP)
 - Basiert auf dem kommerziellen Produkt „Pretty Good Privacy (PGP)“
 - Ähnlich wie S/MIME, hybride Verschlüsselung
 - Unterstützt mehr Algorithmen als S/MIME und PGP

- RFC 4880 (OpenPGP)
 - Basiert auf dem kommerziellen Produkt „Pretty Good Privacy (PGP)“
 - Ähnlich wie S/MIME, hybride Verschlüsselung

	S/MIME	OpenPGP	PGP 5.x
Asymmetrische Verschlüsselung	RSA	RSA, ElGamal, Elliptic Curves, Diffie-Hellman	RSA, ElGamal
Asymmetrische Signatur	RSA	RSA, DSA, ElGamal, ECDSA	RSA, DSA
Symmetrische Algorithmen	TripleDES, DES, RC2	TripleDES, CAST5, IDEA, Blowfish, SaferSK128, Twofish	TripleDES, IDEA, CAST5
Hash-Algorithmen	MD5, SHA-1	MD5, SHA-1, RIPE/MD-160, MD2, Double-width SHA	MD5, SHA-1

- RFC 4880 (OpenPGP)
 - Basiert auf dem kommerziellen Produkt „Pretty Good Privacy (PGP)“
 - Ähnlich wie S/MIME, hybride Verschlüsselung
 - Unterstützt mehr Algorithmen als S/MIME und PGP
 - Web-of-Trust statt hierarchischer PKI





- RFC 4880 (OpenPGP)
 - Basiert auf dem kommerziellen Produkt „Pretty Good Privacy (PGP)“
 - Ähnlich wie S/MIME, hybride Verschlüsselung, gleiche Schutzziele
 - Unterstützt mehr Algorithmen als S/MIME und PGP
 - Web-of-Trust statt hierarchischer PKI
 - Programmier-API seit Version 2.0 (libgcrypt)
- Ebenfalls, wie S/MIME, verwundbar gegen Efail!
 - Update: Modification Detection Codes (MDC) verhindern Änderungen des Ciphertexts

Vielen Dank für die Aufmerksamkeit!

Fragen?

Nächste Vorlesung:

- Montag, 9. Mai 2022

Nächste Übung:

- Dienstag, 3. Mai 2022 – 16 Uhr
- Abgabe des Übungszettels bis morgen – 16 Uhr