TECHNISCHE UNIVERSITÄT DRESDEN

Faculty of Computer Science Institute of Systems Architecture, Chair of Privacy and Data Security

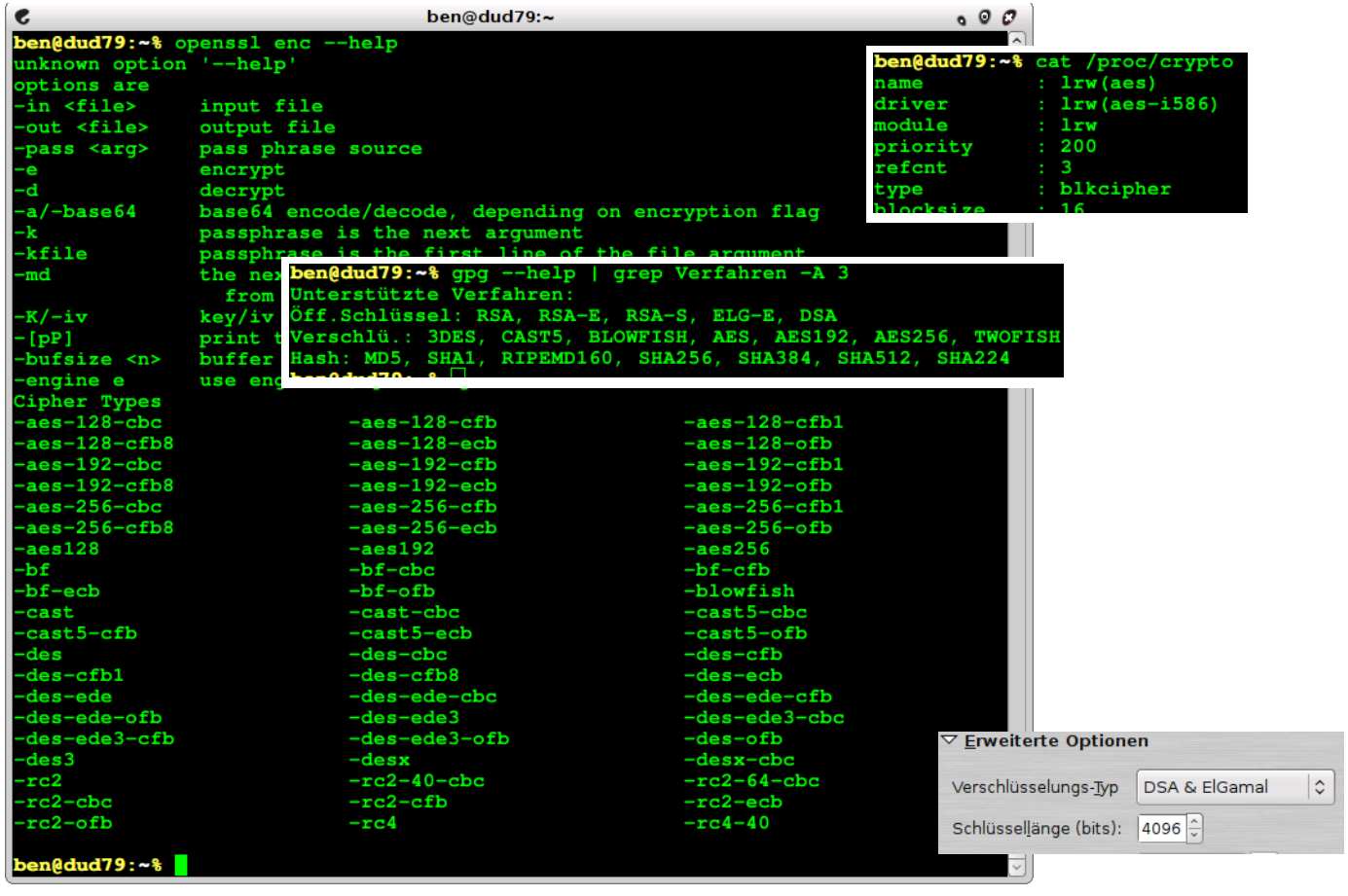
Kurze Einführung in kryptographische Grundlagen

Was ist eigentlich AES,RSA,DH,ELG,DSA,DSS,ECB,CBC

Benjamin.Kellermann@tu-dresden.de
D19E 04A8 8895 020A 8DF6
0092 3501 1A32 491A 3D9C

Dresden, 03.10.2009

Worum geht es?



Worum geht es?

Worum geht es?

Schlüssel

Zufall am Rechner

- Münze werfen, Würfeln
- Zeit seit Systemstart oder zwischen Tastenanschlägen
- Benutzer nach seed fragen

Schlüssel

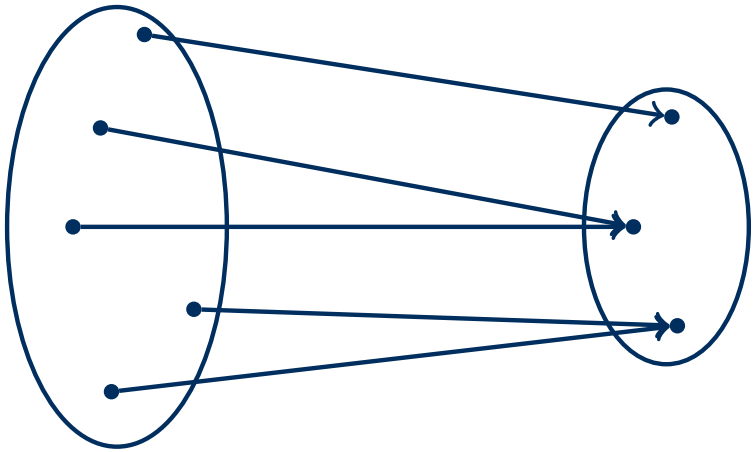
Zufall am Rechner

- Münze werfen, Würfeln
- Zeit seit Systemstart oder zwischen Tastenanschlägen
- Benutzer nach seed fragen

Seed = Passwort

- zufällig ist zufällig ist zufällig
  - Passwortgenerator benutzen
- lang genug
  - aufschreiben
  - Passwortmanager benutzen
  - Gedächtnis trainieren

Was ist ein Hash?



- Abbildung von großer auf kleine Menge
- schwer umkehrbar
- kollisionsresistent

Wofür brauch ich das überhaupt?

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
mQGiBEcFJrkRBACDnfVuIgwhAGbBCQ5Vn9cu5R2ngY+YmfbcqYgDrJIT0LF0w6u3
IzKOd1seHih5zURjsm0KsOz38szvbms8IcJoL6LPs04QI8BJmkDS1qZAzXkdtSuV
zF5QdezMczmJHpu4TSVPCrN2PG0OD8k57T4llG78ubEhfWAPPNKQWP9nDwCgwgpz
7X7iSOJOWf2j7/exefwPrzED/0
171c2fvnG2s/GF9V0HHYH+BSoW
MANNBhdNhbBWbkLQGUkrghSBoi
dAhCA/9gCsOHNEk+G4lOR65AIB
9I7SydZ8cmUvXO6jjocQmRdypZ
hJ4F6M9IK9BYbRD1BRMVgNfLFb
SGVpbmUgPEhlaW5yaWNoQGhlaW
AgQVAggDBBYCAwECHgECF4AACg
co+BLvsAnAkvHonVK5C+cMY5Jt
=AgZE
-----END PGP PUBLIC KEY BLO
```

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
mQGiBEcFJm8RBADE3d+8rooGxa6p9EFWLpmjy5Uv8hbL7iime71BvECpBrNMf8h9
+Skvt1Ad37JUGbgOCVvEqbdhqYifdIbTGqCt7UjpLdBHKEqkT+InZvJb3qQzcwDB
1e2rSkiWPyt/xR9pz6oUJ8sPF8V/4M2RQDKB2pNcfncH6qUnZY0CWy1QCwCg39ne
4/kzlv7OVf7lwLY0iATJJBsD/3X5M/MKNuH20Sx1S1mKcVPjcm7ATNuOvJs5DZJ3
qDI873Uk5QiUpsZrYLgm9YqAHSS0hK8mpBUTLizEs12R0/m3SNp/Yfnac1hmxhZI
3DLgTgPTScrr1Qoh9A7N9ZIYr8G5d3JNCr1gU60jIFI2/AzXs0j/L/DxuY7ayNEW
NNnWBACMxFo4lvGZ8IIpMb8gXYOFyAiv7aSNUl7wj9eJDodFmS9tmrPO2aX9/zmZ
cw38w0YalXNaVmcHa9ubVow5wbl9gA04gGLuAgnpBQkk2inTIw88X0zMDtCcPZfV
JyD/yts/ML5OcVhQjcOcwu48FTElSBg7soqecnHC+19UC9Kr5bQiSGVpbnJpY2gg
SGVpbmUgPEhlaW5yaWNoQEhlaW5lLmRlPohgBBMRAgAgBQJHBSZvAhsDBgsJCAcD
AgQVAggDBBYCAwECHgECF4AACgkQiZ5hM6LVrV59jQCgz+rXBs+ZJzKQGNgQx2I6
xjgdz4AAnjrnybS8ekLk5JIvBXIrgnM6f2ck
=52iW
-----END PGP PUBLIC KEY BLOCK-----
```

Sind beide Schlüssel gleich?

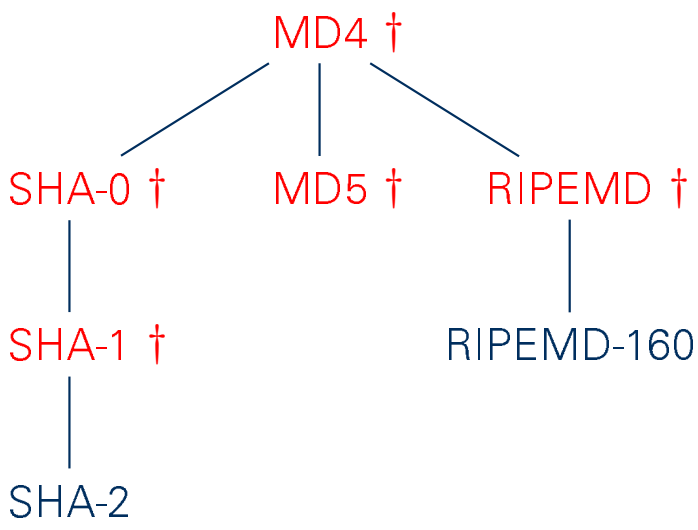
Wofür brauch ich das überhaupt?

Sind beide Schlüssel gleich?

```
B557 3B27 F1D1 1EA6 8BC1  F9C4 899E 6133 A2D5 AD5E
F719 38FB C85E 2B5F 7D86  A106 916B DB9F B94E 64B1
```

- zur Verifikation, ob Daten gleich sind

Überblick über Hashverfahren



WHIRLPOOL

Tiger

Tiger2

Sicherheit ≠ Verschlüsselung

Verschlüsselungsverfahren

- sicherstellen, dass niemand einen Text mitlesen kann

Sicherheit ≠ Verschlüsselung

Verschlüsselungsverfahren

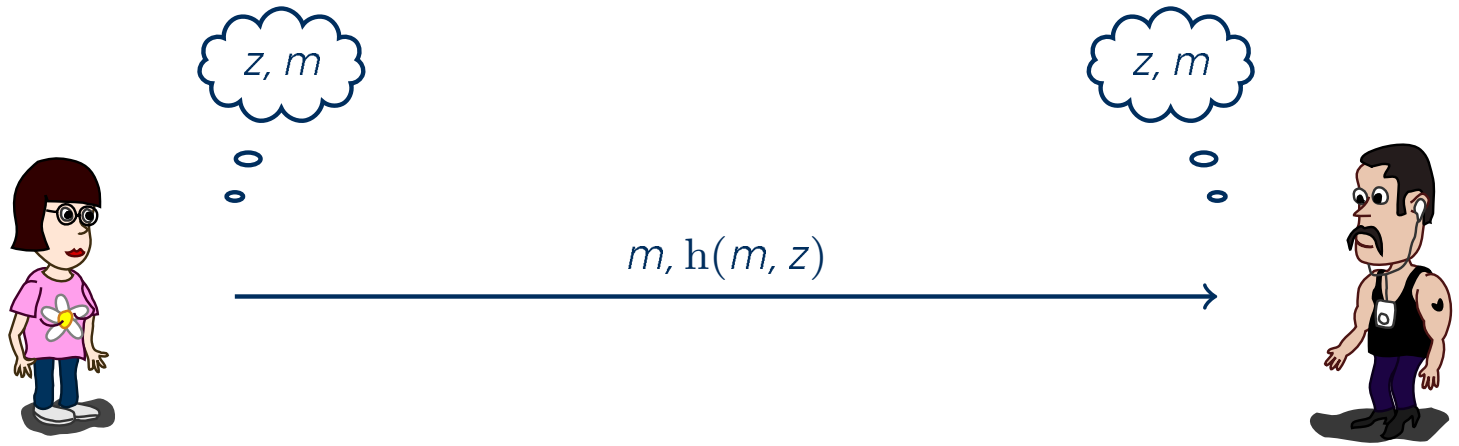
- sicherstellen, dass niemand einen Text mitlesen kann

Signaturverfahren

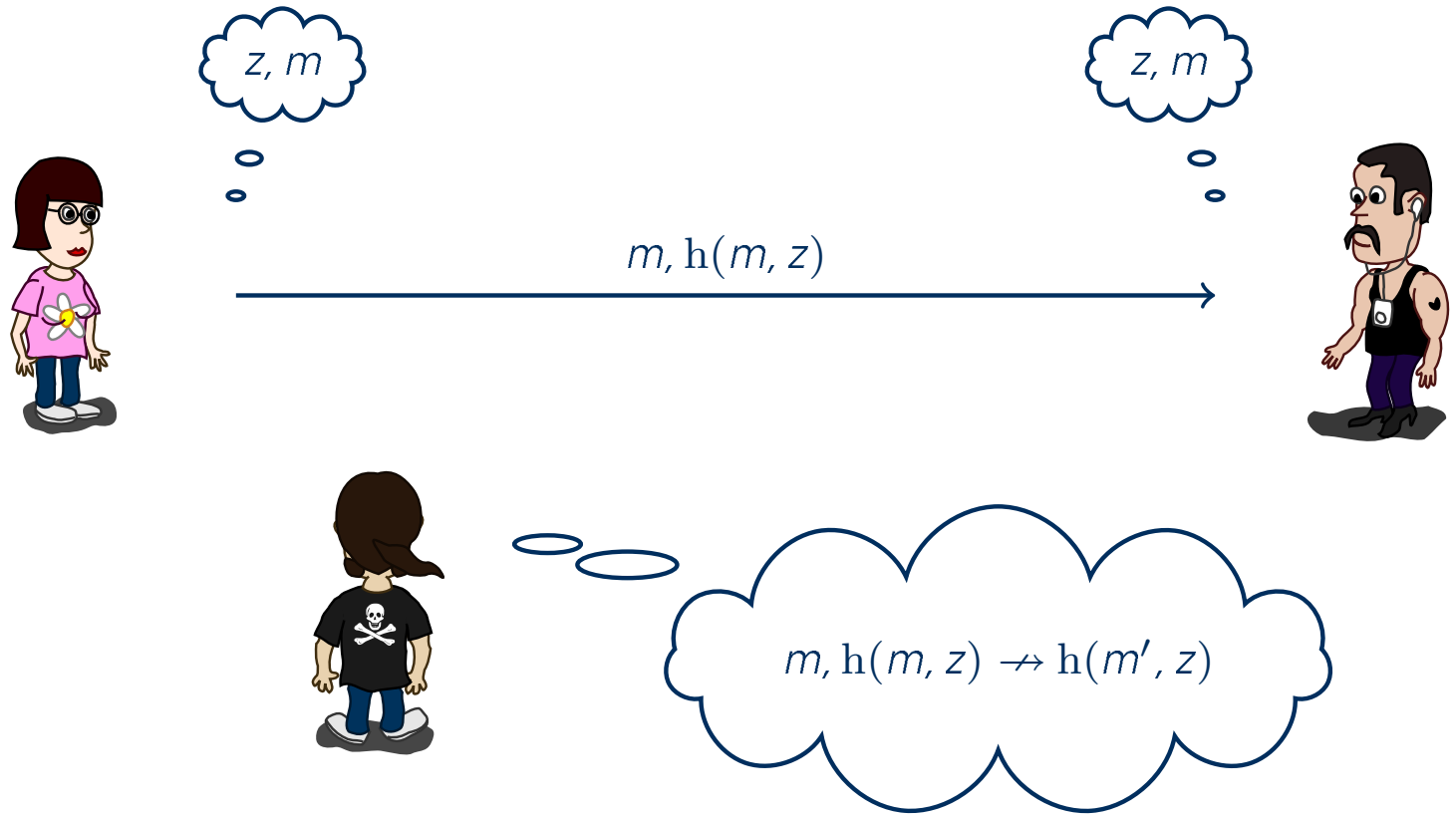
- Absender ist der, den man dafür hält
- kein Angreifer hat etwas verändert
- einem dritten etwas nachweisen (nur asymmetrisch)

Authentikation am Beispiel von HMACs

Authentikation am Beispiel von HMACs



Authentikation am Beispiel von HMACs



Symmetrische Verschlüsselung am Beispiel von Viginère-Chiffre

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

Nachricht		HALLO
Schlüssel	+	BGXWT
Schlüsseltext		

Symmetrische Verschlüsselung am Beispiel von Viginère-Chiffre

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

Nachricht		**HALLO**		7	0	11	11	14
Schlüssel	+	**BGXWT**	$+_{26}$	1	6	23	22	19
Schlüsseltext								

Symmetrische Verschlüsselung am Beispiel von Viginère-Chiffre

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

Nachricht		**HALLO**	$+_{26}$	7	0	11	11	14
Schlüssel	+	**BGXWT**		1	6	23	22	19
Schlüsseltext				8	6	8	7	7

Symmetrische Verschlüsselung am Beispiel von Viginère-Chiffre

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

Nachricht		**HALLO**		7	0	11	11	14
Schlüssel	+	**BGXWT**	$+_{26}$	1	6	23	22	19
Schlüsseltext		**JHJII**		8	6	8	7	7

Symmetrische Verschlüsselung am Beispiel von Viginère-Chiffre

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

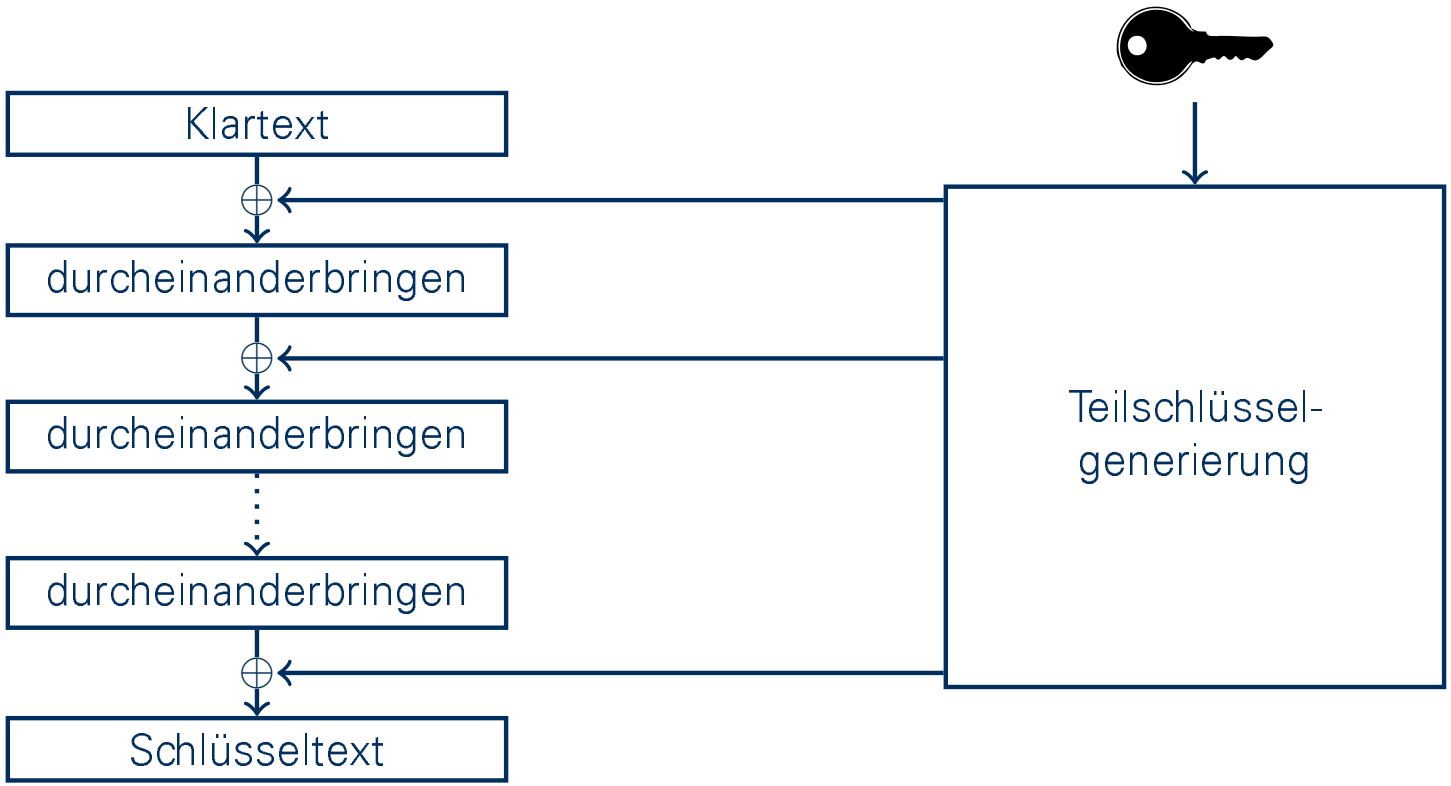
Nachricht		HALLO		7	0	11	11	14
Schlüssel	+	BGXWT	$+_{26}$	1	6	23	22	19
Schlüsseltext		JHJII		8	6	8	7	7
Schlüssel	−	BGXWT	$-_{26}$	1	6	23	22	19

Symmetrische Verschlüsselung am Beispiel von Viginère-Chiffre

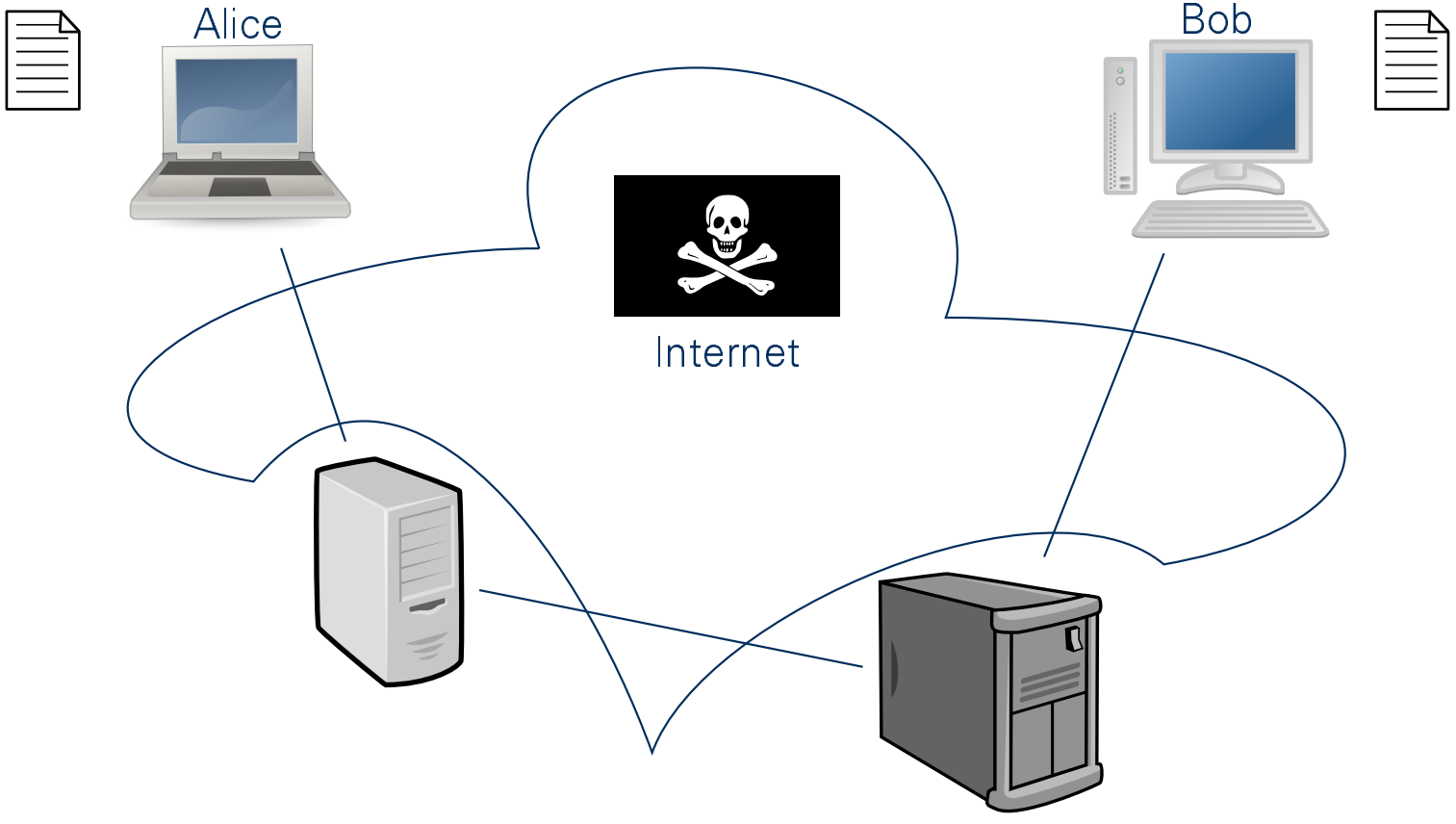
A	B	C	D	E	F	G	H	I	J	K	L	M
0	1	2	3	4	5	6	7	8	9	10	11	12
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
13	14	15	16	17	18	19	20	21	22	23	24	25

Nachricht		**HALLO**		7	0	11	11	14
Schlüssel	$+$	**BGXWT**	$+_{26}$	1	6	23	22	19
Schlüsseltext		**JHJII**		8	6	8	7	7
Schlüssel	$-$	**BGXWT**	$-_{26}$	1	6	23	22	19
Nachricht		**HALLO**		7	0	11	11	14

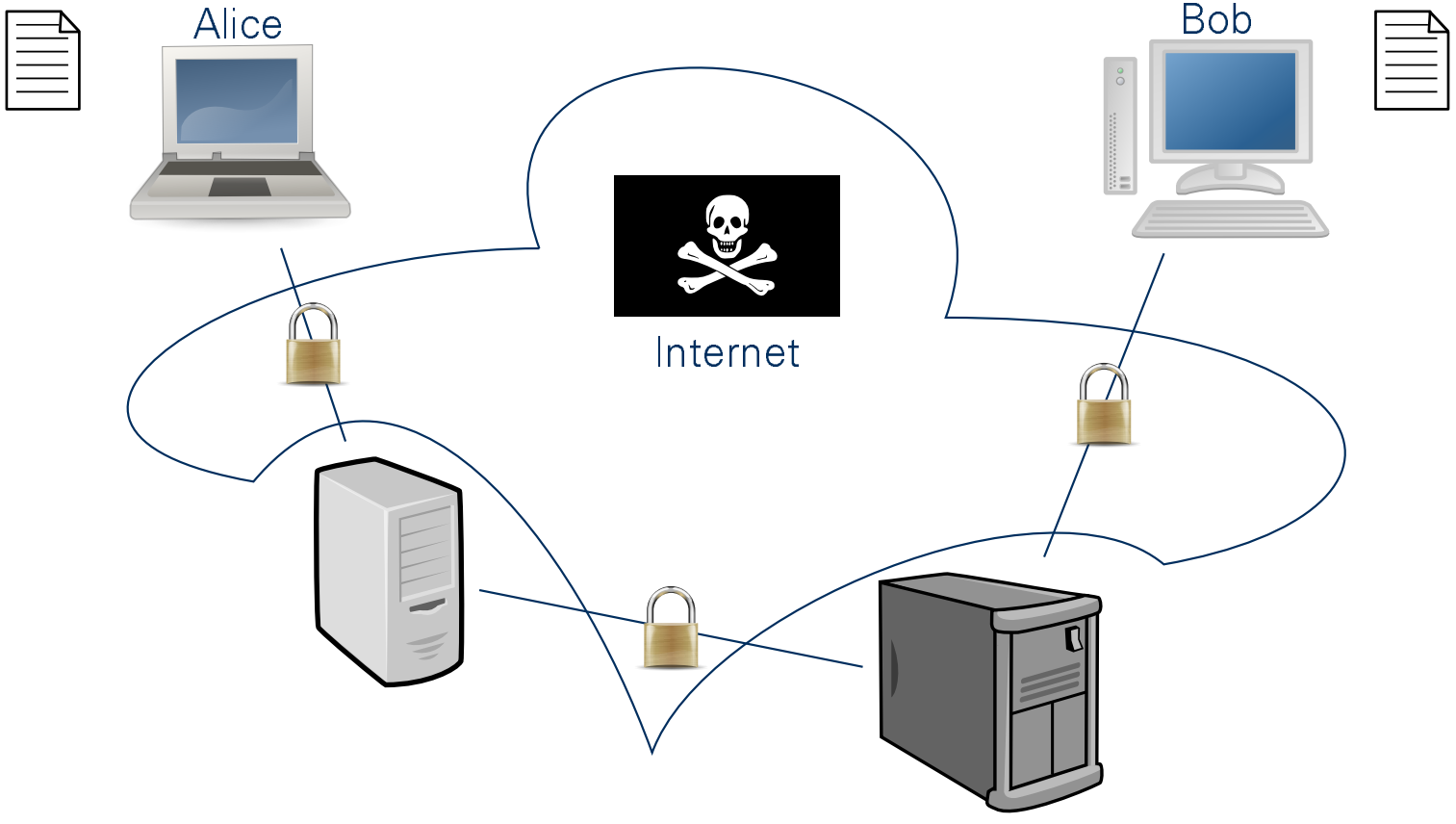
AES (Advanced Encryption Standard)



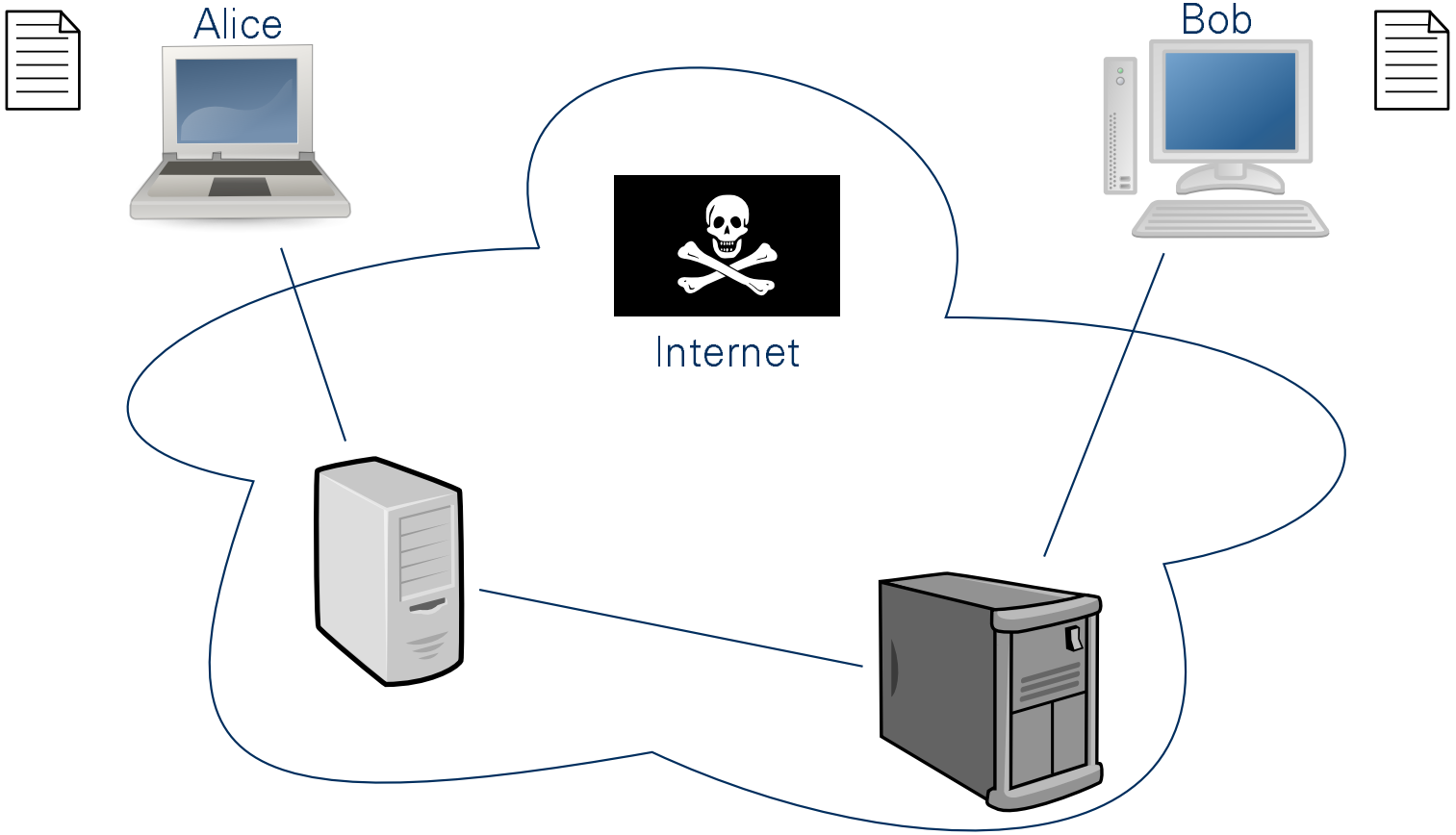
Verbindungs-/ Ende-zu-Ende-Verschlüsselung



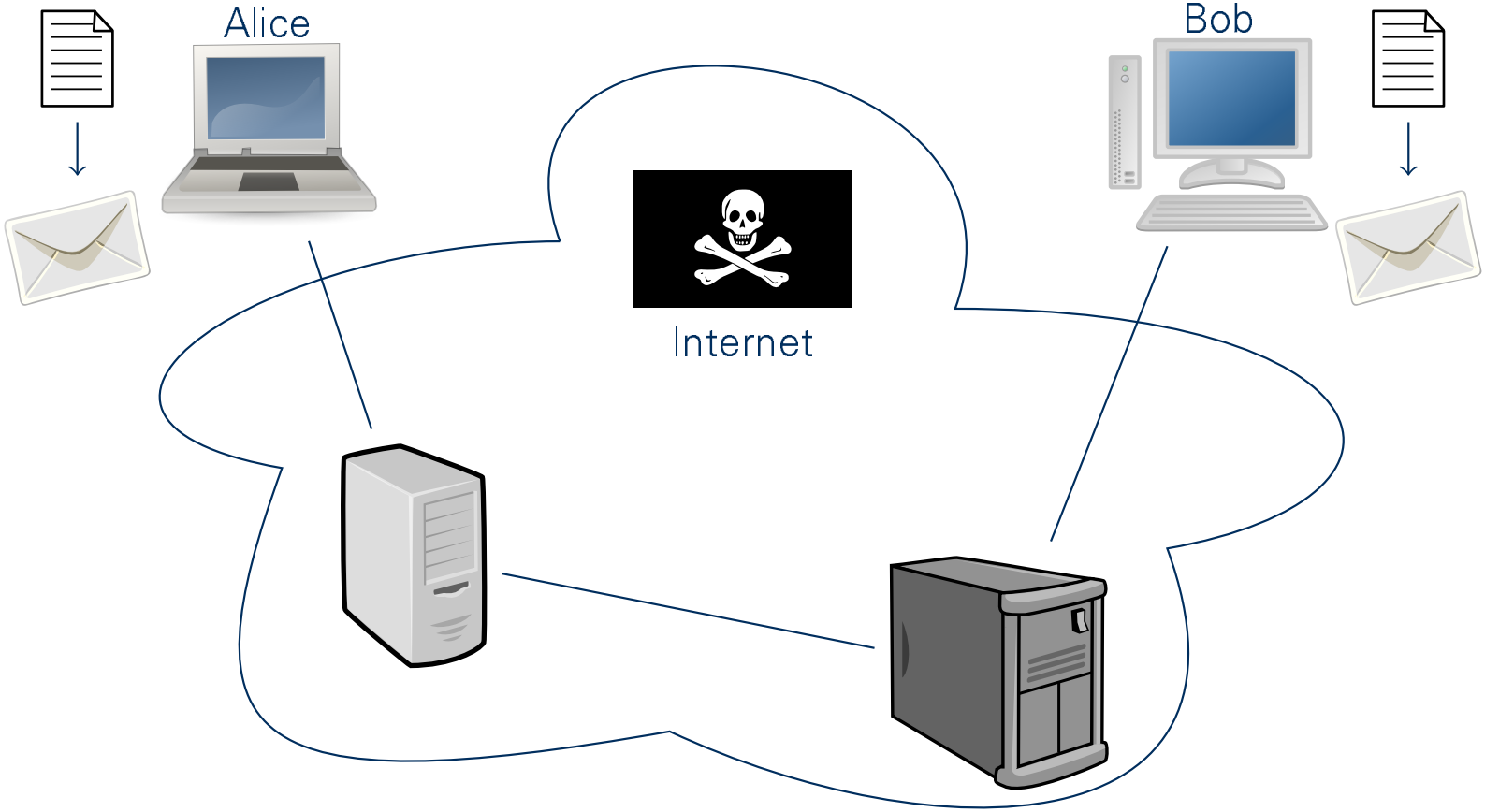
Verbindungs-/ Ende-zu-Ende-Verschlüsselung



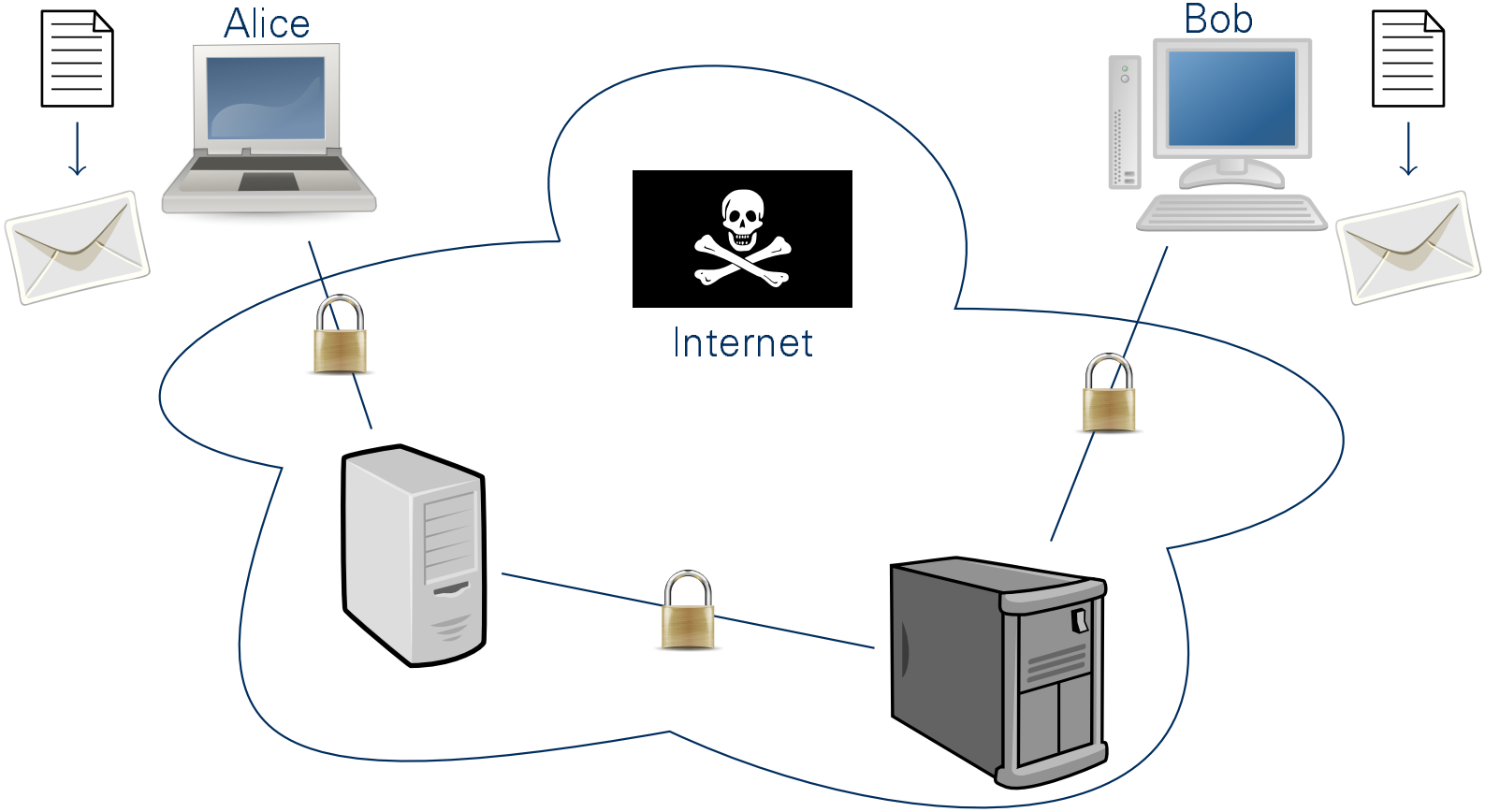
Verbindungs-/ Ende-zu-Ende-Verschlüsselung



Verbindungs-/ Ende-zu-Ende-Verschlüsselung



Verbindungs-/ Ende-zu-Ende-Verschlüsselung



Symmetrische Verfahren im Überblick

Algorithmus	Anmerkung
DES	gebrochen
RC2	gebrochen
RC4, ARC4, ARCFOUR	gebrochen
IDEA	patentiert
3DES	/* no comment */
Blowfish	Vorgänger von Twofish
RC6	in AES-Endrunde
MARS	in AES-Endrunde
Twofish	in AES-Endrunde
Serpent	in AES-Endrunde
AES, Rijndael	wohluntersucht

Nachteile Symmetrischer Verfahren

- Schlüsselaustausch sehr unpraktikabel
- viele Teilnehmer → viele Schlüsselpaare

RSA

Schlüsselgenerierung

- $n = p \cdot q$ (p, q sind große zufällige Primzahlen)
- c mit $\mathrm{ggT}(c, (p-1) \cdot (q-1)) = 1$
- $d = c^{-1} \bmod (p-1) \cdot (q-1)$

RSA

Schlüsselgenerierung

- $n = p \cdot q$ (p, q sind große zufällige Primzahlen)
- c mit $\mathrm{ggT}(c, (p-1) \cdot (q-1)) = 1$
- $d = c^{-1} \bmod (p-1) \cdot (q-1)$

öffentlich

- n, c

geheim

- d

RSA

Schlüsselgenerierung

- $n = p \cdot q$ (p, q sind große zufällige Primzahlen)
- c mit $\mathrm{ggT}(c, (p-1) \cdot (q-1)) = 1$
- $d = c^{-1} \bmod (p-1) \cdot (q-1)$

öffentlich

- n, c

geheim

- d

verschlüsseln

- m . . . Nachricht
- $x = m^c \bmod n$

entschlüsseln

- $m = x^d = (m^c)^d \bmod n$

RSA

Schlüsselgenerierung

- $n = p \cdot q$ (p, q sind große zufällige Primzahlen)
- c mit $\mathrm{ggT}(c, (p-1) \cdot (q-1)) = 1$
- $d = c^{-1} \bmod (p-1) \cdot (q-1)$

RSA

Schlüsselgenerierung

- $n = p \cdot q = 3 \cdot 11 = 33$
- c mit $\mathrm{ggT}(c, (p-1) \cdot (q-1)) = 1$
- $d = c^{-1} \bmod (p-1) \cdot (q-1)$

RSA

Schlüsselgenerierung

- $n = p \cdot q = 3 \cdot 11 = 33$
- $c = 3$ mit $\mathrm{ggT}(3, 2 \cdot 10) = 1$
- $d = c^{-1} \bmod (p-1) \cdot (q-1)$

RSA

Schlüsselgenerierung

- $n = p \cdot q = 3 \cdot 11 = 33$
- $c = 3$ mit $\mathrm{ggT}(3, 2 \cdot 10) = 1$
- $d = 7 = 3^{-1} \bmod 20$ $(3 \cdot 7 = 21 = 1 \bmod 20)$

RSA

Schlüsselgenerierung

- $n = p \cdot q = 3 \cdot 11 = 33$
- $c = 3$ mit $\mathrm{ggT}(3, 2 \cdot 10) = 1$
- $d = 7 = 3^{-1} \bmod 20$ $(3 \cdot 7 = 21 = 1 \bmod 20)$

verschlüsseln ($m = 31$)

$$
\begin{aligned}
x &= 31^3 && \bmod 33 \\
&= (-2)^3 && \bmod 33 \\
&= -8 && \bmod 33 \\
&= 25
\end{aligned}
$$

RSA

Schlüsselgenerierung

- $n = p \cdot q = 3 \cdot 11 = 33$
- $c = 3$ mit $\mathrm{ggT}(3, 2 \cdot 10) = 1$
- $d = 7 = 3^{-1} \bmod 20$ $(3 \cdot 7 = 21 = 1 \bmod 20)$

verschlüsseln ($m = 31$)

$$
\begin{aligned}
x &= 31^3 && \bmod 33 \\
&= (-2)^3 && \bmod 33 \\
&= -8 && \bmod 33 \\
&= 25
\end{aligned}
$$

entschlüsseln

$$m \equiv 25^7 \equiv (-8)^7 \equiv \left((-2)^3\right)^7$$

RSA

Schlüsselgenerierung

- $n = p \cdot q = 3 \cdot 11 = 33$
- $c = 3$ mit $\mathrm{ggT}(3, 2 \cdot 10) = 1$
- $d = 7 = 3^{-1} \bmod 20$ $(3 \cdot 7 = 21 = 1 \bmod 20)$

verschlüsseln ($m = 31$)

$$
\begin{aligned}
x &= 31^3 && \bmod 33 \\
&= (-2)^3 && \bmod 33 \\
&= -8 && \bmod 33 \\
&= 25
\end{aligned}
$$

entschlüsseln

$$
\begin{aligned}
m &\equiv 25^7 \equiv (-8)^7 \equiv \left((-2)^3\right)^7 \\
&\equiv (-2)^{21} \equiv -2 \cdot (-2)^{20}
\end{aligned}
$$

RSA

Schlüsselgenerierung

- $n = p \cdot q = 3 \cdot 11 = 33$
- $c = 3$ mit $\mathrm{ggT}(3, 2 \cdot 10) = 1$
- $d = 7 = 3^{-1} \bmod 20$ $(3 \cdot 7 = 21 = 1 \bmod 20)$

verschlüsseln ($m = 31$)

$$
\begin{aligned}
x &= 31^3 && \bmod 33 \\
&= (-2)^3 && \bmod 33 \\
&= -8 && \bmod 33 \\
&= 25
\end{aligned}
$$

entschlüsseln

$$
\begin{aligned}
m &\equiv 25^7 \equiv (-8)^7 \equiv \left((-2)^3\right)^7 \\
&\equiv (-2)^{21} \equiv -2 \cdot (-2)^{20} \\
&\equiv -2 \cdot \left((-2)^5\right)^4 \equiv -2 \cdot (-32)^4
\end{aligned}
$$

RSA

Schlüsselgenerierung

- $n = p \cdot q = 3 \cdot 11 = 33$
- $c = 3$ mit $\mathrm{ggT}(3, 2 \cdot 10) = 1$
- $d = 7 = 3^{-1} \bmod 20$ $(3 \cdot 7 = 21 = 1 \bmod 20)$

verschlüsseln ($m = 31$)

$$
\begin{aligned}
x &= 31^3 && \bmod 33 \\
&= (-2)^3 && \bmod 33 \\
&= -8 && \bmod 33 \\
&= 25
\end{aligned}
$$

entschlüsseln

$$
\begin{aligned}
m &\equiv 25^7 \equiv (-8)^7 \equiv \left((-2)^3\right)^7 \\
&\equiv (-2)^{21} \equiv -2 \cdot (-2)^{20} \\
&\equiv -2 \cdot \left((-2)^5\right)^4 \equiv -2 \cdot (-32)^4 \\
&\equiv -2 \cdot 1^4 \equiv 31
\end{aligned}
$$

Diffie Hellmann

Diskrete-Logarithmus-Annahme

- $h = g^x \bmod p$
- Trotz Kenntnis von h, g, p ist x schwer zu berechnen!

Diffie Hellmann

Diskrete-Logarithmus-Annahme

- $h = g^x \bmod p$
- Trotz Kenntnis von h, g, p ist x schwer zu berechnen!

Alice	öffentlich	Bob
Zufall: z_A	Primzahl: p, Generator: g	Zufall: z_B

Diffie Hellmann

Diskrete-Logarithmus-Annahme

- $h = g^x \bmod p$
- Trotz Kenntnis von h, g, p ist x schwer zu berechnen!

Alice	öffentlich	Bob
Zufall: z_A	Primzahl: p, Generator: g	Zufall: z_B
	$g^{z_A} \bmod p \quad \leftrightarrows \quad g^{z_B} \bmod p$	

Diffie Hellmann

Diskrete-Logarithmus-Annahme

- $h = g^x \bmod p$
- Trotz Kenntnis von h, g, p ist x schwer zu berechnen!

Alice	öffentlich	Bob
Zufall: z_A	Primzahl: p, Generator: g	Zufall: z_B
	$g^{z_A} \bmod p \quad \leftrightarrows \quad g^{z_B} \bmod p$	
$(g^{z_B})^{z_A} \bmod p$		$(g^{z_A})^{z_B} \bmod p$

ElGamal

geheimer Schlüssel

- Zufall: z_A

öffentlicher Schlüssel

- g, p, g^{z_A}

ElGamal

geheimer Schlüssel

- Zufall: z_A

öffentlicher Schlüssel

- g, p, g^{z_A}

Nachricht verschlüsseln

- Zufall z_B wählen
- Nachricht mit $g^{z_A \cdot z_B}$ verschlüsseln
- g^{z_B} mod p zusammen mit verschlüsselter Nachricht verschicken

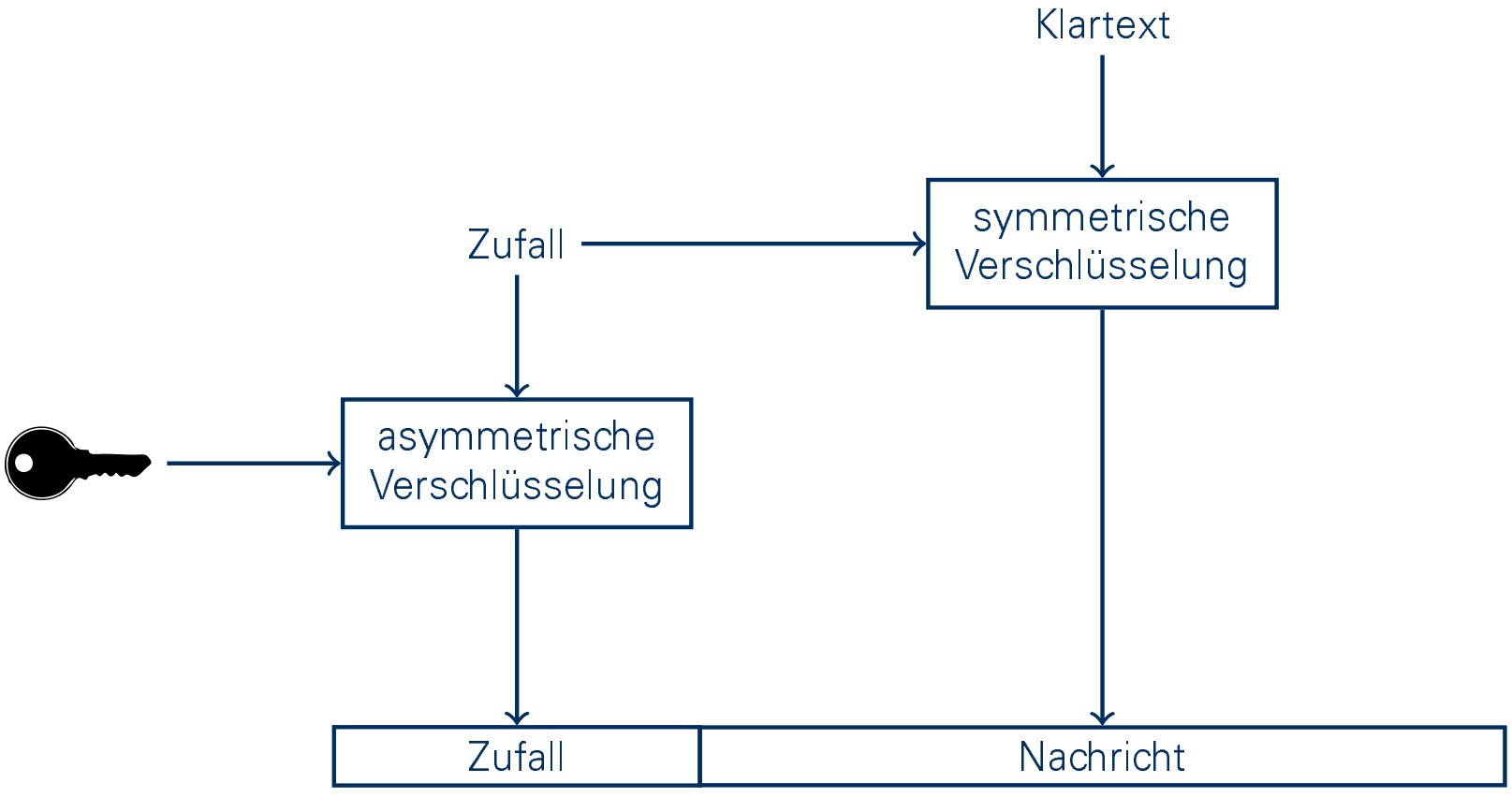
Asymmetrische Verfahren im Überblick

- RSA
- ELG/ElGamal (DSA/DSS)
- Kryptosysteme auf Basis elliptischer Kurven

asymmetrisch vs. symmetrisch

	asymmetrisch	symmetrisch
Schlüsselaustausch	gut	schlecht
Performance	schlecht	gut

Hybride Kryptographie



Betriebsmodi

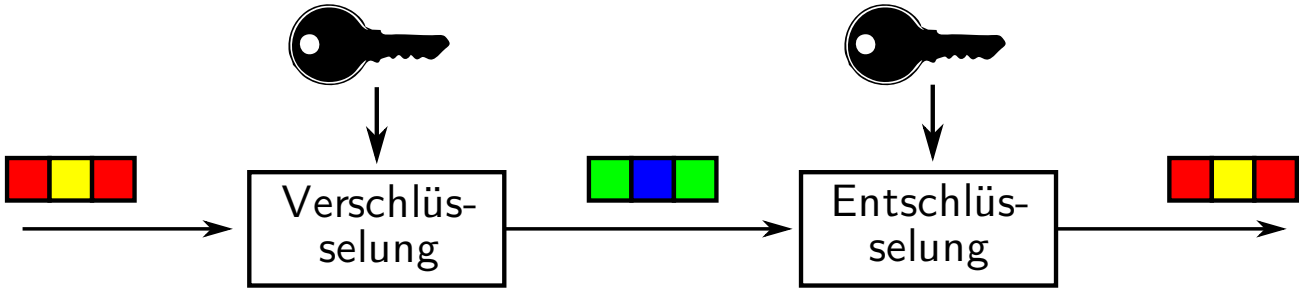
Wozu?

- Verfahren brauchen feste Blöcke
- Länge von Nachrichten nicht vorhersagbar

Beispiel

- $m =$ „5 ist Quersumme von 23!“
  - 23 chars
  - 8-Bit kodiert: $23 \cdot 8 = 184$ Bit
- AES: Blockgröße von 128, 192 oder 256 Bit

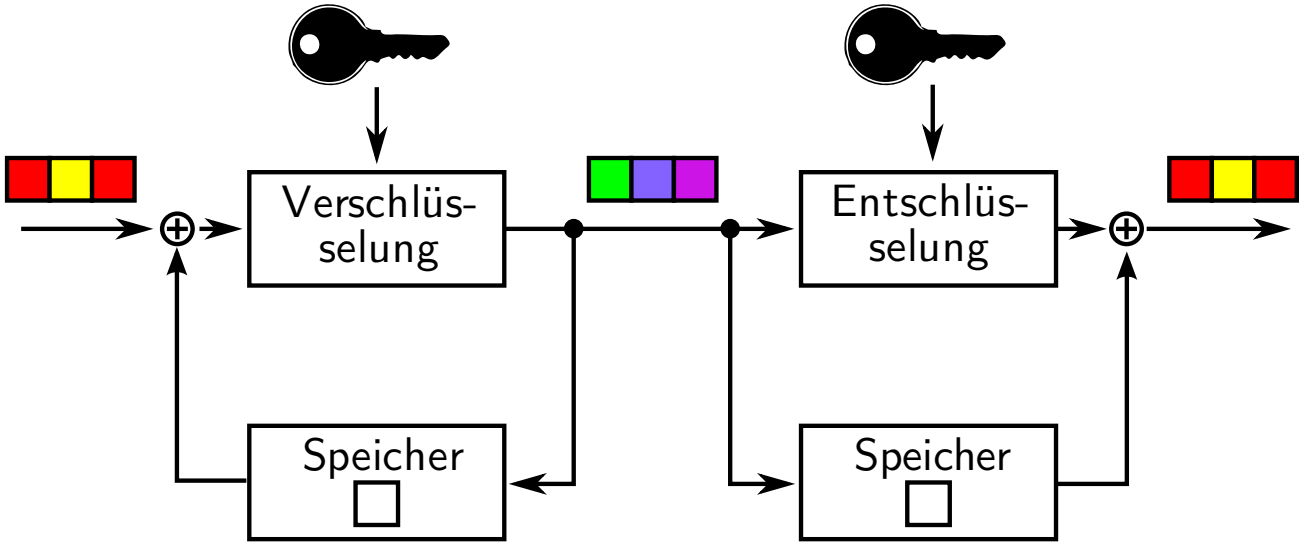
ECB (Electronic Code Book)



Nachteil

- gleiche Blöcke sehen verschlüsselt gleich aus

CBC (Cipher Block Chaining)



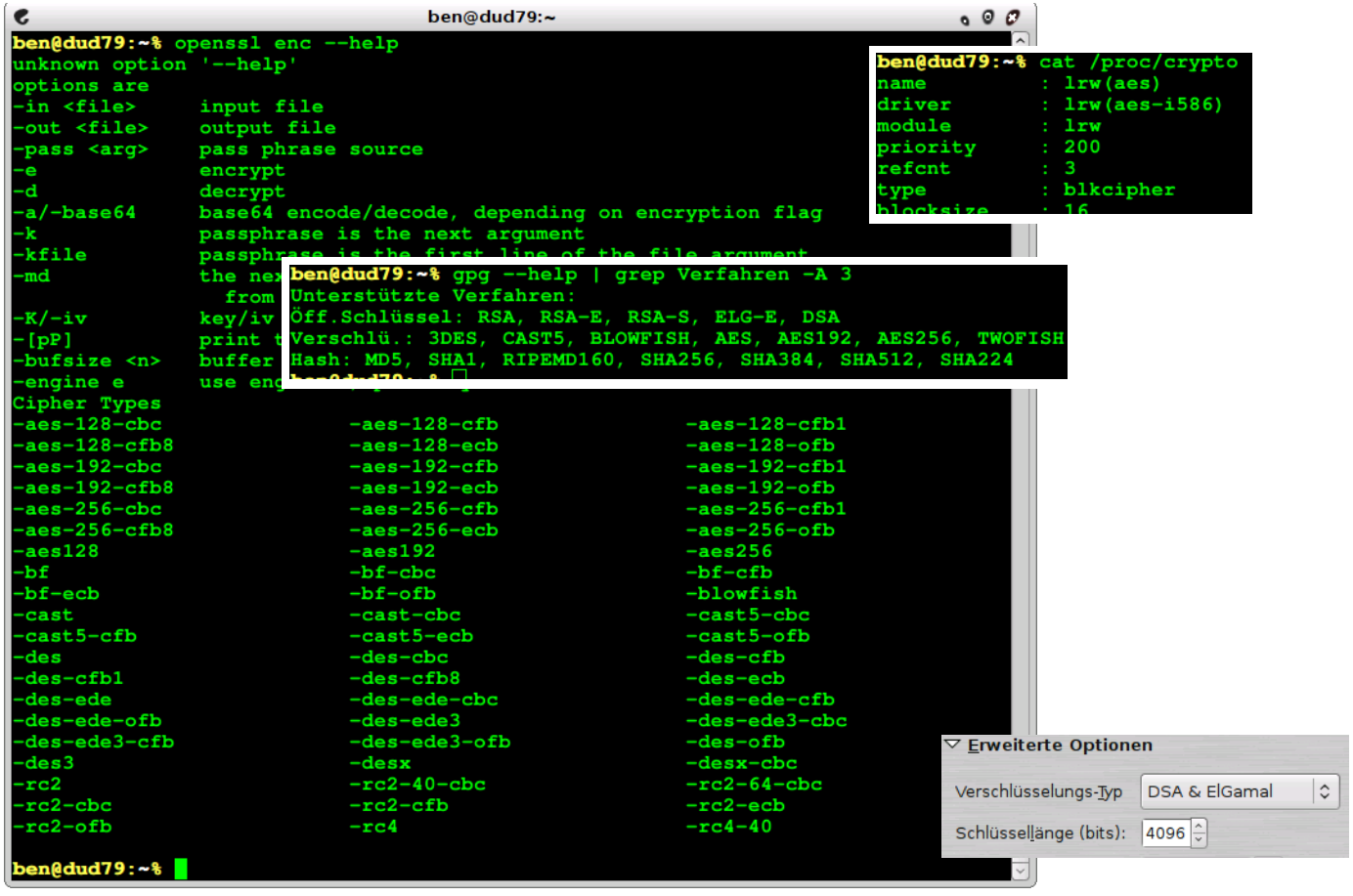
Vorteil

- gleiche Blöcke sehen verschlüsselt unterschiedlich aus

Weitere Betriebsmodi

- ECB (Electronic Codebook)
- CBC (Codebook Chaining)
- CTR (CBC im Counter Mode)
- CBCR (Channel Byte Count Register)
- OFB (Output Feedback)
- CFB (Cipher Feedback)
- LRW (Liskov-Rivest-Wagner)

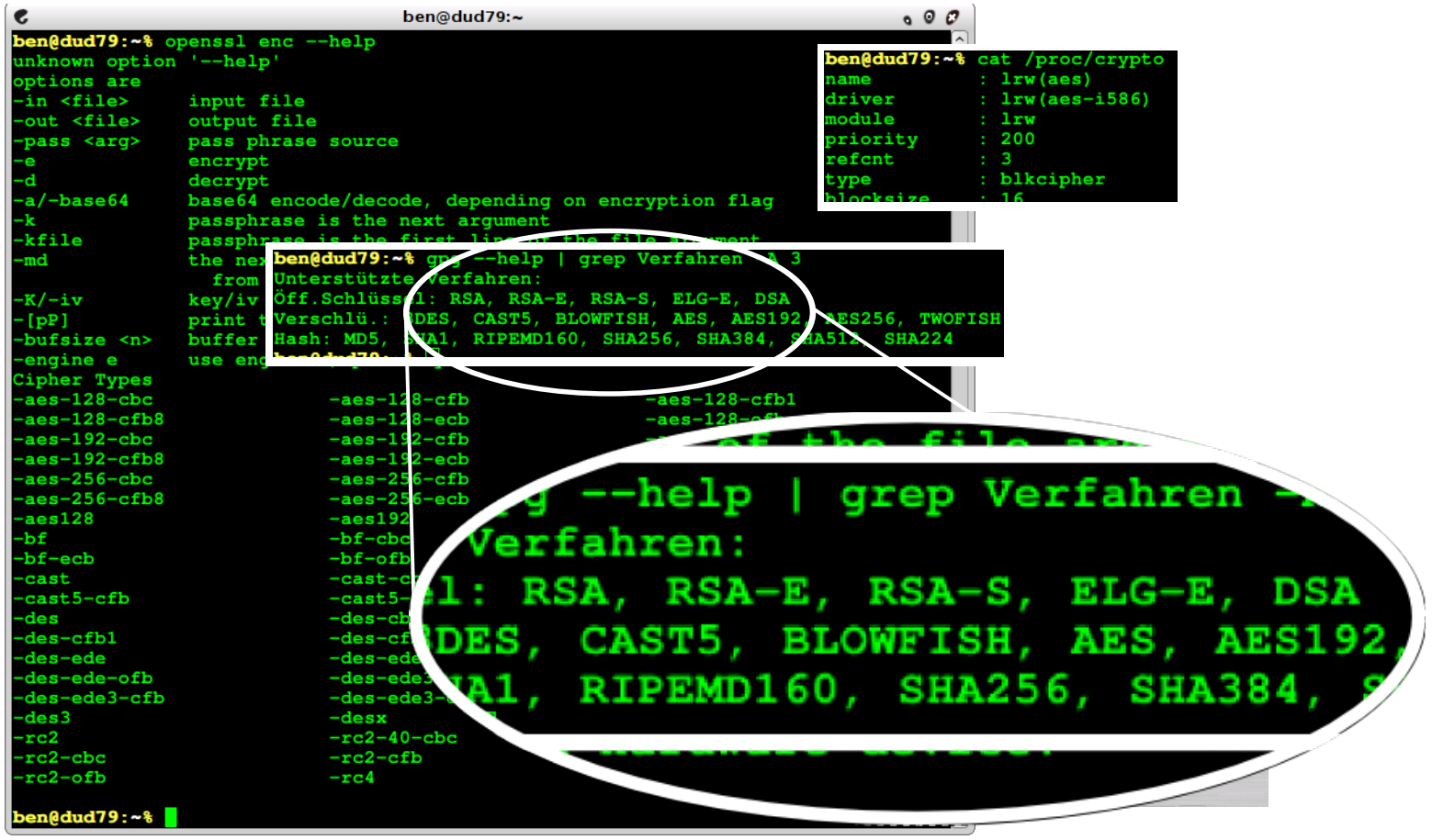
Alles klar?


ben@dud79:~
ben@dud79:~% openssl enc --help
unknown option '--help'
options are
-in <file>
-out <file>
-pass <arg>
-e
-d
-a/-base64
-k
-kfile
-md
-K/-iv
-[pP]
-bufsize <n>
-engine e
Cipher Types
-aes-128-cbc
-aes-128-cfb
-aes-128-cfb1
-aes-128-cfb8
-aes-128-ecb
-aes-128-ofb
-aes-192-cbc
-aes-192-cfb
-aes-192-cfb1
-aes-192-cfb8
-aes-192-ecb
-aes-192-ofb
-aes-256-cbc
-aes-256-cfb
-aes-256-cfb1
-aes-256-cfb8
-aes-256-ecb
-aes-256-ofb
-aes128
-aes192
-aes256
-bf
-bf-cbc
-bf-cfb
-bf-ecb
-bf-ofb
-blowfish
-cast
-cast-cbc
-cast5-cbc
-cast5-cfb
-cast5-ecb
-cast5-ofb
-des
-des-cbc
-des-cfb
-des-cfb1
-des-cfb8
-des-ecb
-des-ede
-des-ede-cbc
-des-ede-cfb
-des-ede-ofb
-des-ede3
-des-ede3-cbc
-des-ede3-cfb
-des-ede3-ofb
-des-ofb
-des3
-desx
-desx-cbc
-rc2
-rc2-40-cbc
-rc2-64-cbc
-rc2-cbc
-rc2-cfb
-rc2-ecb
-rc2-ofb
-rc4
-rc4-40
ben@dud79:~% cat /proc/crypto
name : lrw(aes)
driver : lrw(aes-i586)
module : lrw
priority : 200
refcnt : 3
type : blkcipher
ben@dud79:~% gpg --help | grep Verfahren -A 3
Unterstützte Verfahren:
Öff.Schlüssel: RSA, RSA-E, RSA-S, ELG-E, DSA
Verschlü.: 3DES, CAST5, BLOWFISH, AES, AES192, AES256, TWOFISH
Hash: MD5, SHA1, RIPEMD160, SHA256, SHA384, SHA512, SHA224
Erweiterte Optionen
Verschlüsselungs-Typ
DSA & ElGamal
Schlüssellänge (bits):
4096
ben@dud79:~%

Alles klar?


ben@dud79:~
ben@dud79:~% openssl enc --help
unknown option '--help'
options are
-in <file>
-out <file>
-pass <arg>
-e
-d
-a/-base64
-k
-kfile
-md
-K/-iv
-[pP]
-bufsize <n>
-engine e
Cipher Types
-aes-128-cbc
-aes-128-cfb8
-aes-192-cbc
-aes-192-cfb8
-aes-256-cbc
-aes-256-cfb8
-aes128
-bf
-bf-ecb
-cast
-cast5-cfb
-des
-des-cfb1
-des-ede
-des-ede-ofb
-des-ede3-cfb
-des3
-rc2
-rc2-cbc
-rc2-ofb
-aes-128-ecb
-aes-192-cfb
-aes-192-ecb
-aes-256-cfb
-aes-256-ecb
-aes192
-bf-ofb
-cast-cbc
-cast5-ecb
-des-cbc
-des-cfb8
-des-ede-cbc
-des-ede3
-des-ede3-ofb
-desx
-rc2-40-cbc
-rc2-cfb
-rc4
-des-ofb
-desx-cbc
-rc2-64-cbc
-rc2-ecb
-rc4-40
ben@dud79:~% gpg --help | grep
Unterstützte Verfahren:
Öff.Schlüssel: RSA, RSA-E, RSA
Verschlü.: 3DES, CAST5, BLOWF
Hash: MD5, SHA1, RIPEMD160, S
-aes-128-cfb
-aes-128-ecb
-aes-192-cfb
-aes-192-ecb
-aes-256-cfb
-aes-256-ecb
aes192
DSA & ElGamal
Schlüssellänge (bits):
4096
ben@dud79:~%

Alles klar?


ben@dud79:~
ben@dud79:~% openssl enc --help
unknown option '--help'
options are
-in <file>
-out <file>
-pass <arg>
-e
-d
-a/-base64
-k
-kfile
-md
-K/-iv
-[pP]
-bufsize <n>
-engine e
Cipher Types
-aes-128-cbc
-aes-128-cfb8
-aes-192-cbc
-aes-192-cfb8
-aes-256-cbc
-aes-256-cfb8
-aes128
-bf
-bf-ecb
-cast
-cast5-cfb
-des
-des-cfb1
-des-ede
-des-ede-ofb
-des-ede3-cfb
-des3
-rc2
-rc2-cbc
-rc2-ofb
-aes-128-cfb
-aes-128-ecb
-aes-192-cfb
-aes-192-ecb
-aes-256-cfb
-aes-256-ecb
-aes192
-bf-cbc
-bf-ofb
-des-ede3
-desx
-rc2-40-cbc
-rc2-cfb
-rc4
-aes-128-cfb1
ben@dud79:~% cat /proc/crypto
name : lrw(aes)
driver : lrw(aes-i586)
module : lrw
priority : 200
refcnt : 3
type : blkcipher
Unterstützte Verfahren:
Hash: MD5, SHA1, RIPEMD160, SHA256, SHA384, SHA512, SHA224
--help | grep Verfahren
Verfahren:
RSA, RSA-E, RSA-S, ELG-E, DSA
CAST5, BLOWFISH, AES, AES192,
RIPEMD160, SHA256, SHA384,
ben@dud79:~%

EOF

--verbose

- Wikipedia
-  bsi algorithmenkatalog
- Versuchsanleitungen zum Komplexpraktikum, Script Datensicherheit und Kryptographie:

   Datensicherheit Kryptographie
- Security and Cryptography, Montags 9:20–12:40 Uhr, TUD, Fakultät Informatik, E023