Kurze Einführung in kryptographische Grundlagen.

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

Benjamin.Kellermann@gmx.de

GPG-Fingerprint: D19E 04A8 8895 020A 8DF6 0092 3501 1A32 491A 3D9C
git clone http://www.eigenheimstrasse.de/ben/silc_ta.git

26.10.2007 / SILC Themenabend

Worum geht es?

Wissen was dahintersteht!

Worum geht es?

Wissen was dahintersteht!

Worum geht es?

Wissen was dahintersteht!

Was schon immer mal gesagt werden musste!

Schlüssel

Was zu beachten ist

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

Seed = Passwort

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

Was schon immer mal gesagt werden musste!

Schlüssel

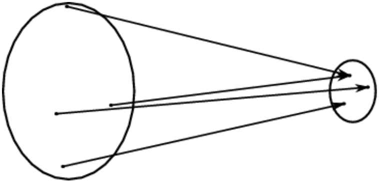
Was zu beachten ist

- Zufällig
    - Münze werfen, Würfeln
    - Zeit seit Systemstart oder zwischen Anschlägen
    - 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
- nicht umkehrbar
- kollisionsresistent

Wofür brauch ich das überhaupt?

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
mQGiBEcFJrkRBACDnfVuIgwhAGbBCQ5Vn9cu5R2ngY+YmfbcqYgDrJIT0LF0w6u3
IzKOd1seHih5zURjsm0KsOz38szvbms8IcJoL6LPs04QI8BJmkDS1qZAzXkdtSuV
zF5QdezMczmJHpu4TSVPCrN2PG0OD8k57T4llG78ubEhfWAPPNKQWP9nDwCgwgpz
7X7iSOJOWf2j7/exefwPrzED/0ltcZHgotqOBtIdVYWGmScAD2VAi7rFsGq60tIR
171c2fvnG2s/GF9V0HHYH+BSoW88E+0vGaApBzDkoSihEm//yo0i/79+5T+Vm7OF
MANNBhdNhbBWbkLQGUkrghSBoi+DnMWPBg+EFtdW41o4zrRwCmoiQbuA5GR+2n24
dAhCA/9gCsOHNEk+G4lOR65AIBUelZdzRka53fKCkLps48o+zdwPh98juJxE1Oc3
9I7SydZ8cmUvXO6jjocQmRdypZYIvzqLMwIMSFcQ1412T4fz7x99++e52l6J1Ucc
hJ4F6M9IK9BYbRD1BRMVgNfLFbt2TMaT81eDxqrb7j0nUwcWzbQiSGVpbnJpY2gg
SGVpbmUgPEhlaW5yaWNoQGhlaW5lLmRlPohgBBMRAgAgBQJHBSa5AhsDBgsJCAcD
AgQVAggDBBYCAwECHgECF4AACgkQkWvbn7lOZLFTCQCghZpkXjFL9qzqYS4RMWrX
co+BLvsAnAkvHonVK5C+cMY5JtL2/cEI/Tr+
=AgZE
-----END PGP PUBLIC KEY BLOCK-----
```

Sind beide Schlüssel gleich?

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
mQGiBEcFJm8RBADE3d+8rooGxa6p9EFWLpmjy5Uv8hbL7iime71BvECpBrNMf8h9
+Skvt1Ad37JUGbgOCVvEqbdhqYifdIbTGqCt7UjpLdBHKEqkT+InZvJb3qQzcwDB
1e2rSkiWPyt/xR9pz6oUJ8sPF8V/4M2RQDKB2pNcfncH6qUnZY0CWy1QCwCg39ne
4/kzlv7OVf7lwLY0iATJJBsD/3X5M/MKNuH2OSx1S1mKcVPjcm7ATNuOvJs5DZJ3
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-----
```

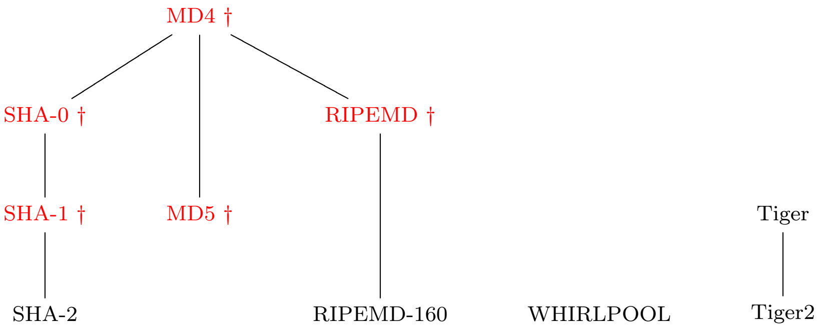
Wofür brauch ich das überhaupt?

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



2 Arten von Sicherheit

Verschlüsselungsverfahren

- sicherstellen, dass niemand einen Text mitlesen kann

Signaturverfahren

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

2 Arten von Sicherheit

Verschlüsselungsverfahren

- sicherstellen, dass niemand einen Text mitlesen kann

Signaturverfahren

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

Wie kann ich etwas signieren?

Symmetrische Authentikation am Beispiel von HMACs

- Nachricht: m; Hashfunktion: $\mathrm{h}(\ldots)$; Zufall: z (Schlüssel)

Alice

- kennt z, m
- berechnet $HMAC = \mathrm{h}(m, z)$

sendet $\longrightarrow$ $m, HMAC$

Bob

- kennt z, m
- überprüft Authentizität ($\mathrm{h}(m, z) \stackrel{?}{=} HMAC$)

Marvin

- kennt nur m, kennt z nicht!
- kann ohne Kentniss von z weder Authentizität überprüfen noch Nachricht fälschen ($\mathrm{h}(m', z)$ berechnen)

Wie kann ich etwas signieren?

Symmetrische Authentikation am Beispiel von HMACs

- Nachricht: m; Hashfunktion: $\mathrm{h}(\dots)$; Zufall: z (Schlüssel)

Alice

- kennt z, m
- berechnet $HMAC = \mathrm{h}(m, z)$

sendet $\rightarrow$ $m, HMAC$

Bob

- kennt z, m
- überprüft Authentizität $(\mathrm{h}(m, z) \stackrel{?}{=} HMAC)$

Marvin

- kennt nur m, kennt z nicht!
- kann ohne Kentniss von z weder Authentizität überprüfen noch Nachricht fälschen $(\mathrm{h}(m', z)$ berechnen)

Wie kann ich etwas signieren?

Symmetrische Authentikation am Beispiel von HMACs

- Nachricht: m; Hashfunktion: $\mathrm{h}(\dots)$; Zufall: z (Schlüssel)

Alice

- kennt z, m
- berechnet $HMAC = \mathrm{h}(m, z)$

sendet
$\longrightarrow$
$m, HMAC$

Bob

- kennt z, m
- überprüft Authentizität $(\mathrm{h}(m, z) \stackrel{?}{=} HMAC)$

Marvin

- kennt nur m, kennt z nicht!
- kann ohne Kentniss von z weder Authentizität überprüfen noch Nachricht fälschen ($\mathrm{h}(m', z)$ berechnen)

Wie kann ich etwas signieren?

Symmetrische Authentikation am Beispiel von HMACs

- Nachricht: m; Hashfunktion: $\mathrm{h}(\dots)$; Zufall: z (Schlüssel)

Alice

- kennt z, m
- berechnet $HMAC = \mathrm{h}(m, z)$

sendet
$\longrightarrow$
$m, HMAC$

Bob

- kennt z, m
- überprüft Authentizität $(\mathrm{h}(m, z) \stackrel{?}{=} HMAC)$

Marvin

- kennt nur m, kennt z nicht!
- kann ohne Kentniss von z weder Authentizität überprüfen noch Nachricht fälschen ($\mathrm{h}(m', z)$ berechnen)

Wie kann ich etwas signieren?

Symmetrische Authentikation am Beispiel von HMACs

- Nachricht: m; Hashfunktion: $\mathrm{h}(\dots)$; Zufall: z (Schlüssel)

Alice

- kennt z, m
- berechnet $HMAC = \mathrm{h}(m, z)$

sendet
$\rightarrow$
$m, HMAC$

Bob

- kennt z, m
- überprüft Authentizität $(\mathrm{h}(m, z) \stackrel{?}{=} HMAC)$

Marvin

- kennt nur m, kennt z nicht!
- kann ohne Kentniss von z weder Authentizität überprüfen noch Nachricht fälschen $(\mathrm{h}(m', z)$ berechnen)

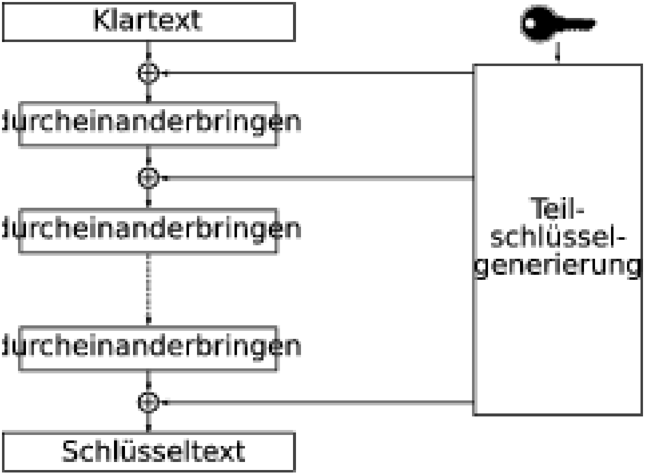
Wie kann ich überhaupt verschlüsseln?

symmetrische Kryptographie 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

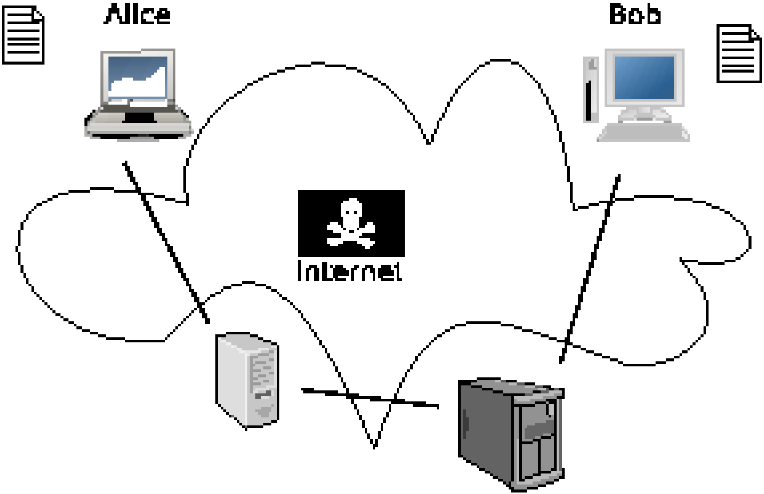
Nachricht		HALLO		7	0	11	11	14
Schlüssel	+	BGXWT	$+_{26}$	1	6	23	22	19
Schlüsseltext		JHJII		8	6	8	7	7
Schlüssel	-	BGXWT	$-_{26}$	1	6	23	22	19
Nachricht		HALLO		7	0	11	11	14

AES (Advanced Encryption Standard)

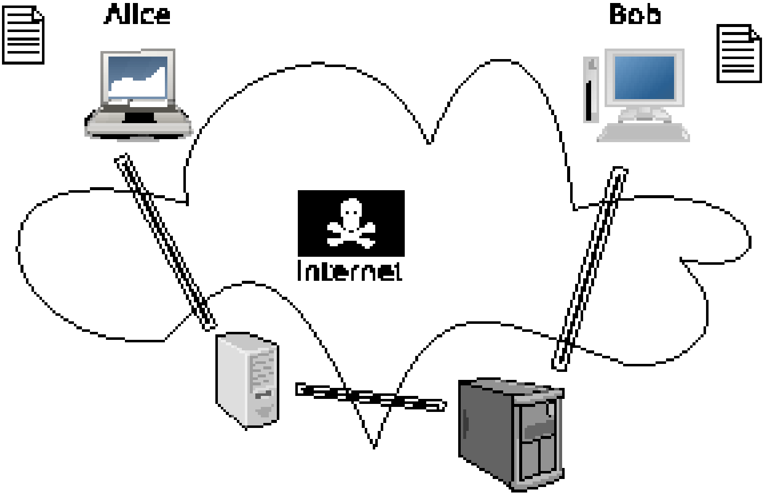


Was muss verschlüsselt werden?

Verbindungsverschlüsselung vs. Ende-zu-Ende-Verschlüsselung

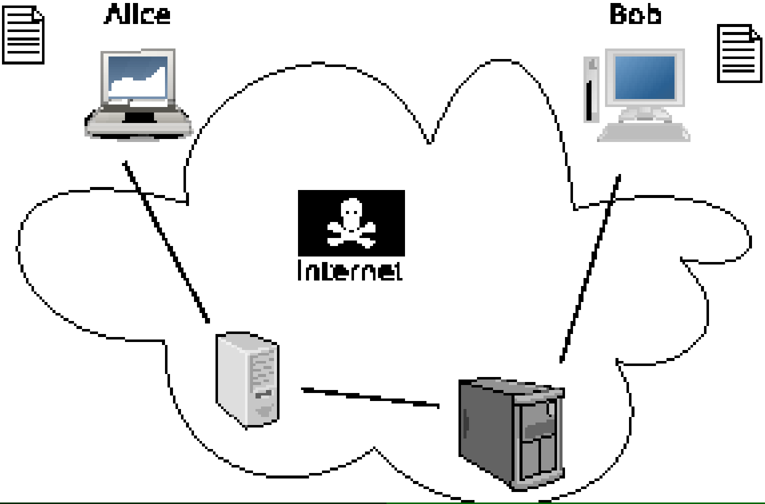


Was muss verschlüsselt werden?
Verbindungsverschlüsselung vs. Ende-zu-Ende-Verschlüsselung



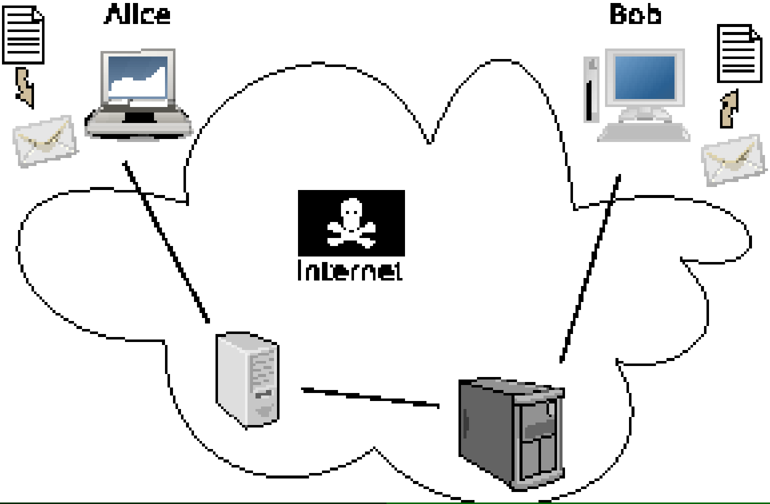
Was muss verschlüsselt werden?

Verbindungsverschlüsselung vs. Ende-zu-Ende-Verschlüsselung



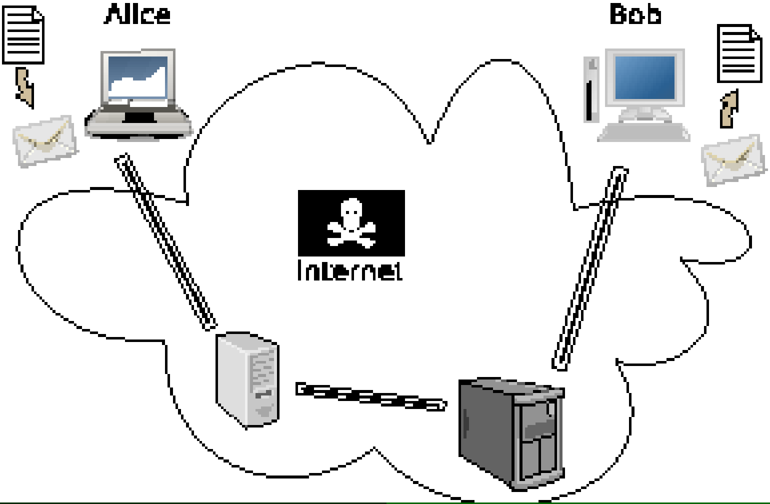
Was muss verschlüsselt werden?

Verbindungsverschlüsselung vs. Ende-zu-Ende-Verschlüsselung



Was muss verschlüsselt werden?

Verbindungsverschlüsselung vs. 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
- bei vielen Teilnehmern werden viele Schlüsselpaare benötigt

RSA

bekanntestes und wohluntersuchtestes asymmetrisches Kryptographieverfahren

Schlüsselgenerierung

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

öffentlich

- n, c

geheim

- d

verschlüsseln

- $m \ldots \mathrm{Nachricht}$
- $x = m^c \mod n$

entschlüsseln

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

RSA

bekanntestes und wohluntersuchtestes asymmetrisches Kryptographieverfahren

Schlüsselgenerierung

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

öffentlich

- n, c

geheim

- d

verschlüsseln

- $m \ldots \text{Nachricht}$
- $x = m^c \mod n$

entschlüsseln

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

RSA

bekanntestes und wohluntersuchtestes asymmetrisches Kryptographieverfahren

Schlüsselgenerierung

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

öffentlich

- n, c

geheim

- d

verschlüsseln

- $m \ldots \text{Nachricht}$
- $x = m^c \mod n$

entschlüsseln

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

RSA
Beispiel

Schlüsselgenerierung

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

verschlüsseln $(m = 31)$

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

entschlüsseln

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

RSA
Beispiel

Schlüsselgenerierung

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

verschlüsseln $(m = 31)$

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

entschlüsseln

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

RSA
Beispiel

Schlüsselgenerierung

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

verschlüsseln ($m = 31$)

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

entschlüsseln

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

RSA
Beispiel

Schlüsselgenerierung

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

verschlüsseln ($m = 31$)

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

entschlüsseln

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

RSA
Beispiel

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

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

RSA
Beispiel

Schlüsselgenerierung

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

verschlüsseln ($m = 31$)

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

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

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

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

Diffie Hellmann

Diskrete-Logarithmus-Annahme

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

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

Diffie Hellmann

Diskrete-Logarithmus-Annahme

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

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

Diffie Hellmann

Diskrete-Logarithmus-Annahme

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

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

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

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

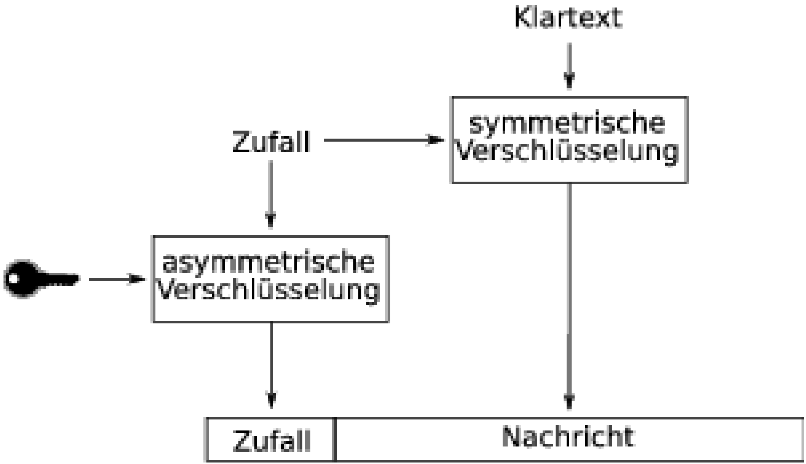
Asymmetrische Verfahren im Überblick

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

asymmetrisch vs. symmetrisch

	asymmetrisch	symmetrisch
Schlüsselaustausch	gut	schlecht
Performance	schlecht	gut

die Vorteile beider nutzen



Betriebsmodi

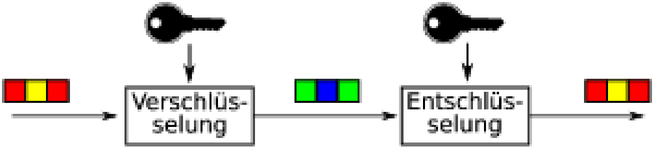
Wozu?

- Verfahren brauchen feste Blöcke
- Länge von Nachrichten nicht vorhersagbar

Beispiel

- Nachricht „5 ist Quersumme von 23!“ besteht aus 23 chars $\rightarrow$ 8-Bit kodiert $\rightarrow 23 \cdot 8 = 184$ Bit
- Verschlüsselung mit AES benötigt Blockgröße von 128, 192 oder 256 Bit.

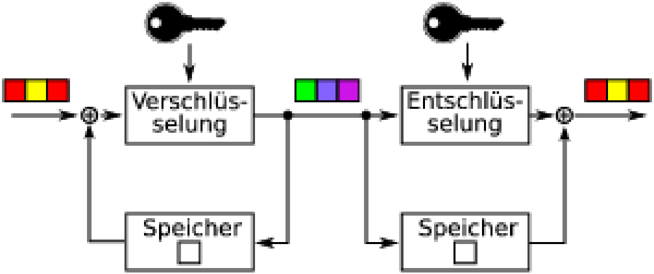
ECB (Electronic Code Book)



Nachteil

- gleiche Blöcke sehen gleich aus

CBC (Cipher Block Chaining)



Vorteil

- gleiche Blöcke sehen 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)

Worum geht es?

Wissen was dahintersteht!

Worum geht es?

Wissen was dahintersteht!

Worum geht es?

Wissen was dahintersteht!

EOF

--verbose

- Wikipedia
- Versuchsanleitungen zum Komplexpraktikum:
  http://www.inf.tu-dresden.de/index.php?node_id=1358&ln=de
  Script Datenschutz und Datensicherheit:
  http://www.inf.tu-dresden.de/index.php?node_id=483&ln=de
- Security and Cryptography, Montags 9:20–12:40 Uhr, TUD, Fakultät Informatik, E023