Crypto Workshop

Einleitung

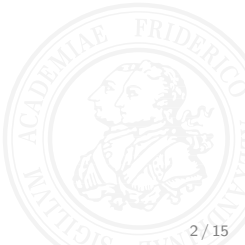
Christoph Egger
Dominik Paulus

11. Mai 2018

Agenda

Einleitung

- Was macht Crypto
- Die guten Klassiker



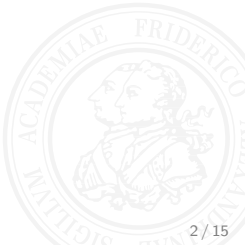
Agenda

Einleitung

- Was macht Crypto
- Die guten Klassiker

Symmetrische Crypto

- Was macht Verschlüsselung
- Was macht Verschlüsselung nicht



Agenda

Einleitung

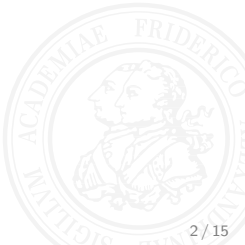
- Was macht Crypto
- Die guten Klassiker

Symmetrische Crypto

- Was macht Verschlüsselung
- Was macht Verschlüsselung nicht

Hash und MAC

- Datenintegrität
- Einseitige Authentifizierung



Agenda

Einleitung

- Was macht Crypto
- Die guten Klassiker

Symmetrische Crypto

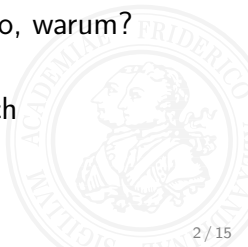
- Was macht Verschlüsselung
- Was macht Verschlüsselung nicht

Hash und MAC

- Datenintegrität
- Einseitige Authentifizierung

Diffie-Hellman

- Asymmetrische Crypto, warum?
- Key Agreement, DH-Schlüsselaustausch

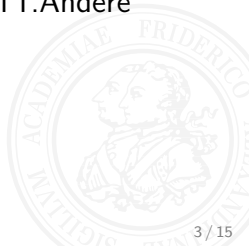


Resources

`https://workshop.faust.ninja`

Username: workshop
Password: gulasch

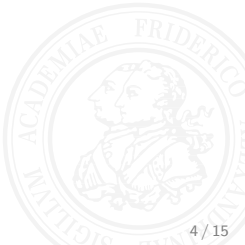
Alle Beispiele funktionieren mit `Python3` und verwenden `binascii`.Andere Sprachen gehen natürlich auch.



Destruktive Crypto

Wir reden heute *nur* darüber, wie man Kryptographie angreift. Daher:

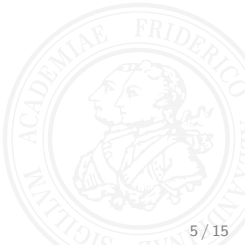
- Alles was wir heute besprechen machen wir kaputt!
- Aber: Wir haben in jeder Gruppe “gute” Algorithmen dabei



Was wir nicht tun

Kryptographie erschaffen

$\rightarrow$ Einführungsvorlesungen in die Kryptographie



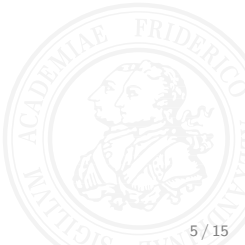
Was wir nicht tun

Kryptographie erschaffen

→ Einführungsvorlesungen in die Kryptographie

Kryptographie richtig einsetzen

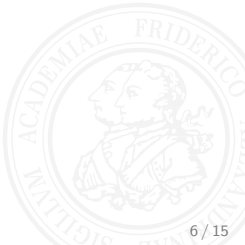
→ Cryptoparties



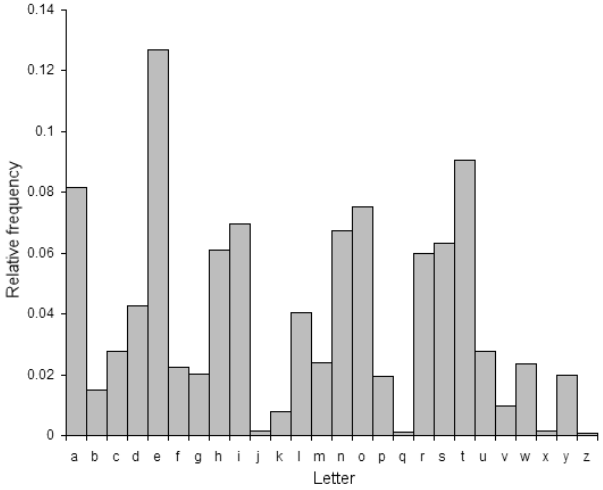
Caesar

```
def caesar(text):
  return bytes(((i - ord('a') + 3) % 26)
                + ord('A')
               for i in text)
```

- Einfache Substitutionschiffre
- Gaius Julius Caesar erster bekannter Nutzer
- Trivial unsicher (nächste Folie)
- Siehe auch: `rot13`

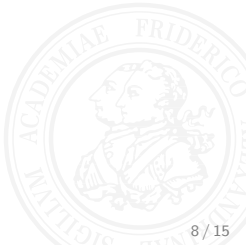


Frequenzanalyse



Demo

die nuetzlichkeit eines dings macht es zum gebrauchswert. aber diese nuetzlichkeit schwebt nicht in der luft. durch die eigenschaften des warenkoerpers bedingt, existiert sie nicht ohne denselben. der warenkoerper selbst, wie eisen, weizen, diamant usw., ...



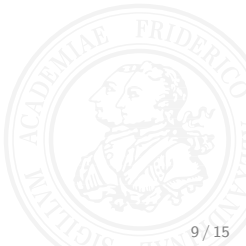
Demo

die nuetzlichkeit eines dings macht es zum gebrauchswert. aber diese nuetzlichkeit schwebt nicht in der luft. durch die eigenschaften des warenkoerpers bedingt, existiert sie nicht ohne denselben. der warenkoerper selbst, wie eisen, weizen, diamant usw., ...

GLH QXHWCOLFKNHLW HLQHV GLQJV PDFKW HV CXP JHEUDXFKVZHUW. DEHU GLHVH QXHWCOLFKNHLW VFKZHEW QLFKW LQ GHU OXIW. GXUFK GLH HLJHQVFKDIWHQ GHV ZDUHQNRHUSHUV EHGLQJW, HALVWLHUW VLH QLFKW RKQH GHQVHOEHQ. GHU ZDUHQNRHUSHU VHOEVW, ZLH HLVHQ, ZHLCHQ, GLDPDQW XVZ., ...

Deutsch:	E	N	R	I	T	S	A	D	H
	148	89	71	70	59	59	59	46	38

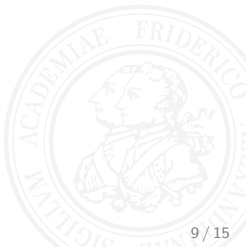
Ciphertext:	H	L	Q	U	W	V	D	G	K
	144	70	64	61	58	57	40	40	40



Deutsch:	E	N	R	I	T	S	A	D	H
	148	89	71	70	59	59	59	46	38

Ciphertext:	H	L	Q	U	W	V	D	G	K
	144	70	64	61	58	57	40	40	40

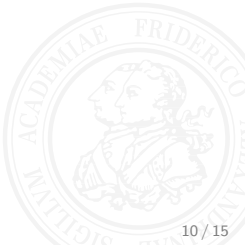
Plaintext:	e	i	n	r	t	s	a	d	h
	144	70	64	61	58	57	40	40	40



One-Time-Pad

```
def onetimepad(key, text):
  return bytes(a^b for (a,b) in
                    zip(key,text))
```

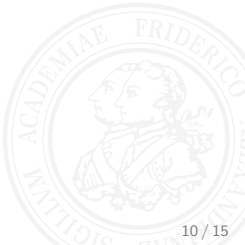
- Perfekt sichere Verschlüsselung



One-Time-Pad

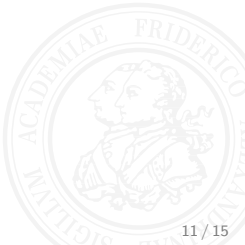
```
def onetimepad(key, text):
  return bytes(a^b for (a,b) in
                    zip(key,text))
```

- Perfekt sichere Verschlüsselung
- … falls der Schlüssel genau einmal verwendet wird



Fragen?

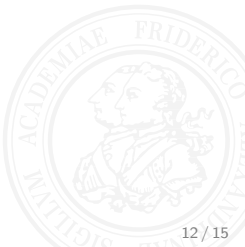
42



Caesar Challenge

Deutsch:	E	N	R	I	T	S
	148	89	71	70	59	59

Ciphertext	O	\n	D	C	Y	X
	153	133	95	77	75	65

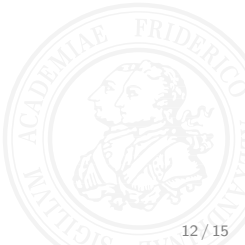


Caesar Challenge

Deutsch:	E	N	R	I	T	S
	148	89	71	70	59	59

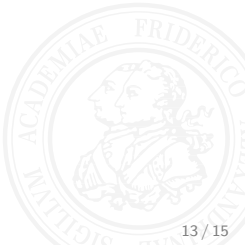
Ciphertext	O	\n	D	C	Y	X
	153	133	95	77	75	65

Plaintext:	e	' '	n	i	s	r
	153	133	95	77	75	65



One-Time-Pad Challenge

- Key reuse: Key wird sowohl für die Flagge als auch unsere selbstgewählte Eingabe genutzt
- Sende eine (beliebige) Nachricht (`known`)
- Query die Flagge
- XOR der Antworten ist $\texttt{known} \oplus \texttt{flag}$



```
nc = nclib.Netcat(('localhost', 4202))
nc.recv()
# extract the key, because key xor 0  = key
nc.send(b'C ' + unhexlify('00000000000000000000'))
key = nc.recv().decode().strip()

nc = nclib.Netcat(('localhost', 4202))
nc.recv()
nc.send(b'F')
flag_cipher = nc.recv().decode().strip()

# extract the flag, because cipher xor key = plain
flag = bytes(a^b for (a, b) in
             zip(unhexlify(flag_cipher),
                 unhexlify(key)))
print(flag.decode())
```

Fragen?

42