Introduction to Cryptography

CyberSisters, on 2024-04-24

$whoami

- Dr. Martin Schmiedecker
- Bosch Engineering by day
- online privacy by night
- digital forensics in between
- find me at schmiedecker.net

Cryptography

Crypto means Cryptography!

Outline

- Encryption methods:
  - Block ciphers
  - public key cryptography
- Hashing

Non-Outline

- History of cryptography
- The math behind most things
- Randomness
- Encrypted emails
- Post-Quantum stuff

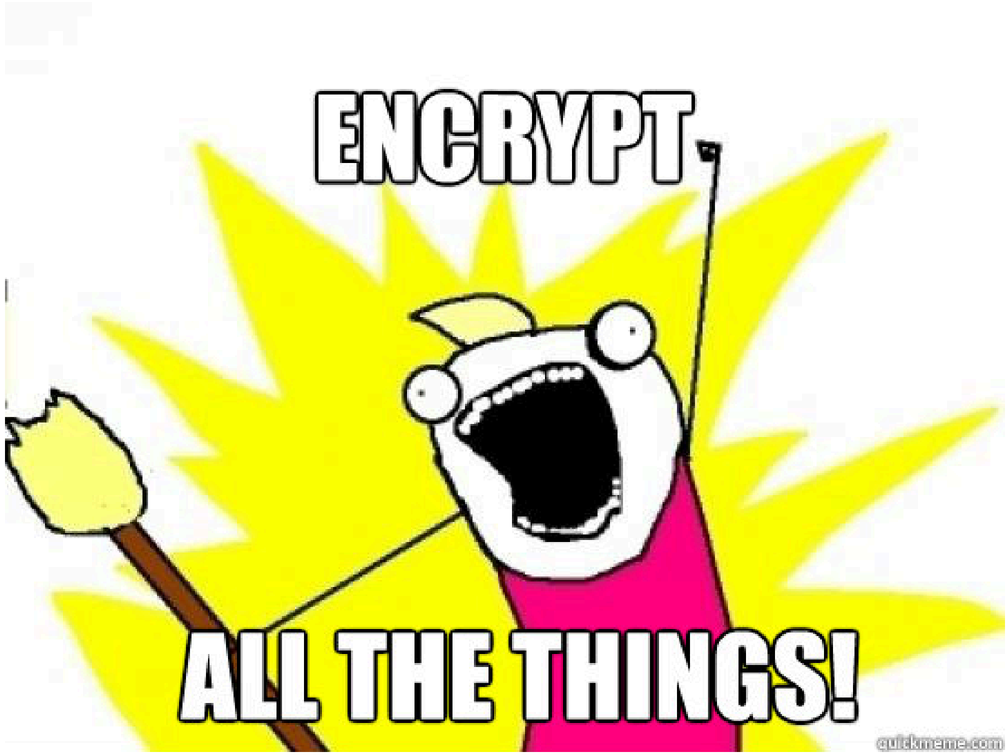
Disclaimer

- not a cryptographer!
- just a guy liking IT security
- please ask if something doesn't add up

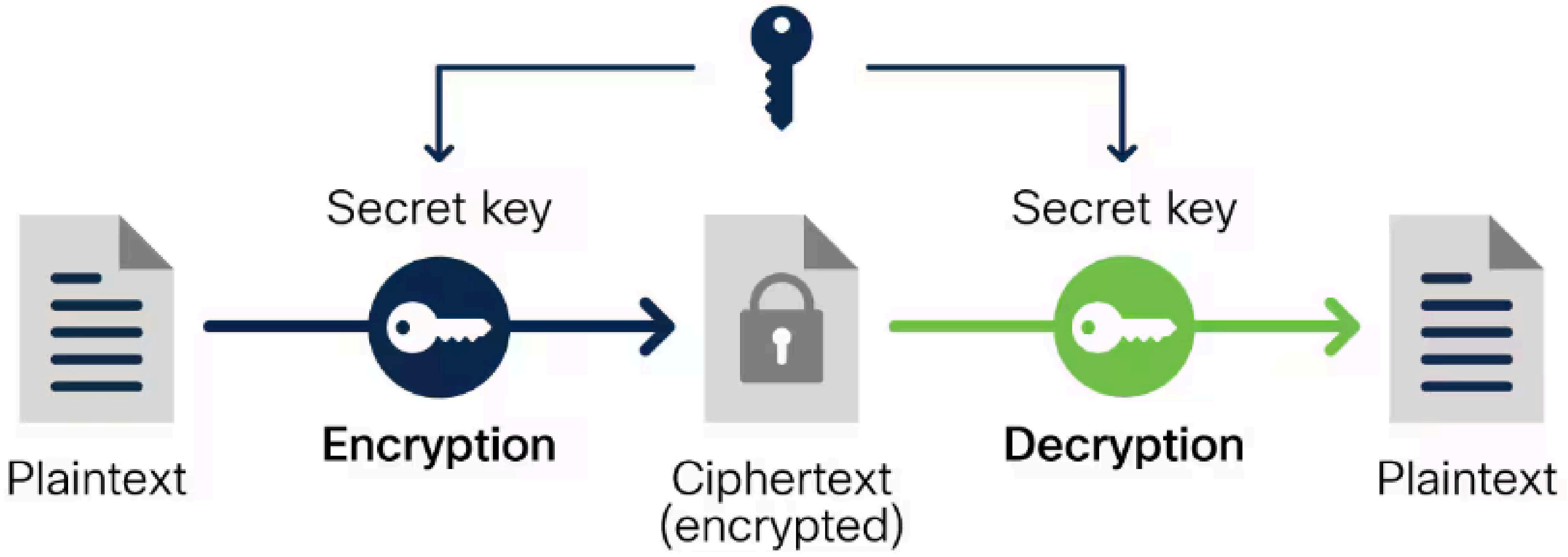
Encryption methods

Why cryptography?

- Confidentiality!
- Integrity!
- (Availability)



Symmetric encryption



Secret key

Secret key

Plaintext

Encryption

Ciphertext
(encrypted)

Decryption

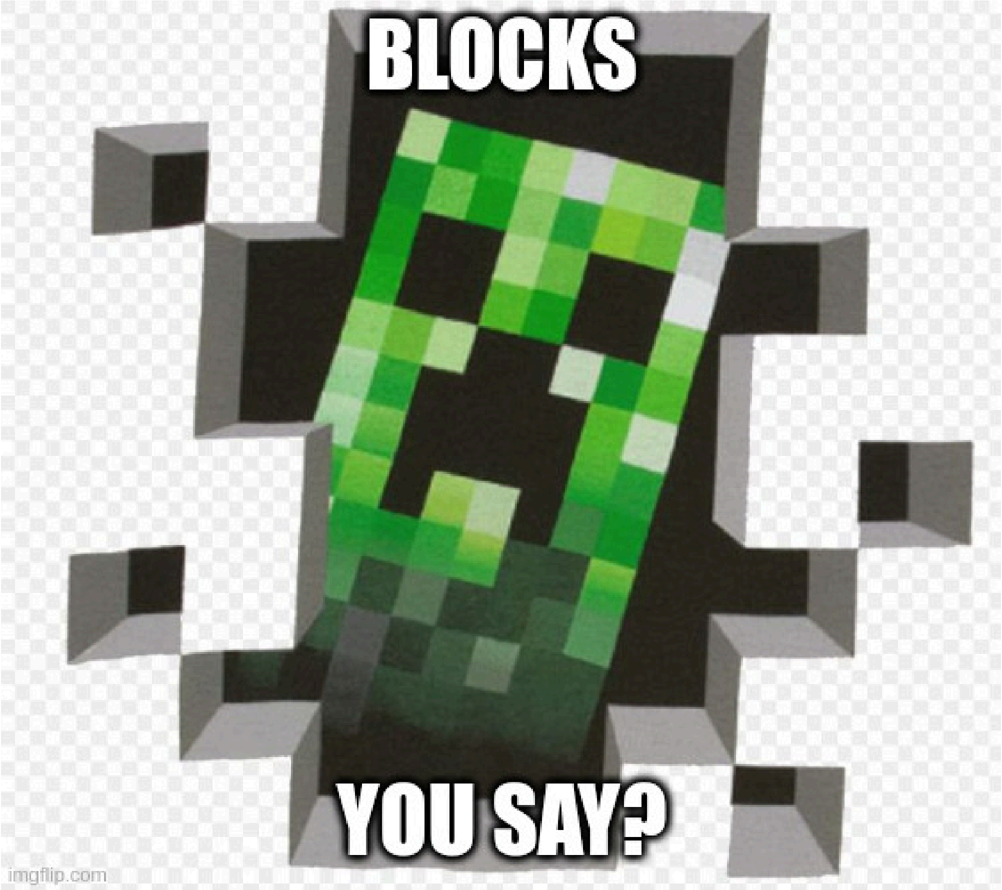
Plaintext

Howto Encryption?

- substitution
- secret key
- deterministic

Modern Encryption

- block of data
- 128, 192 or 256 bit
- AES, Salsa20, 3DES
- GOST in Russia, SM4 in China

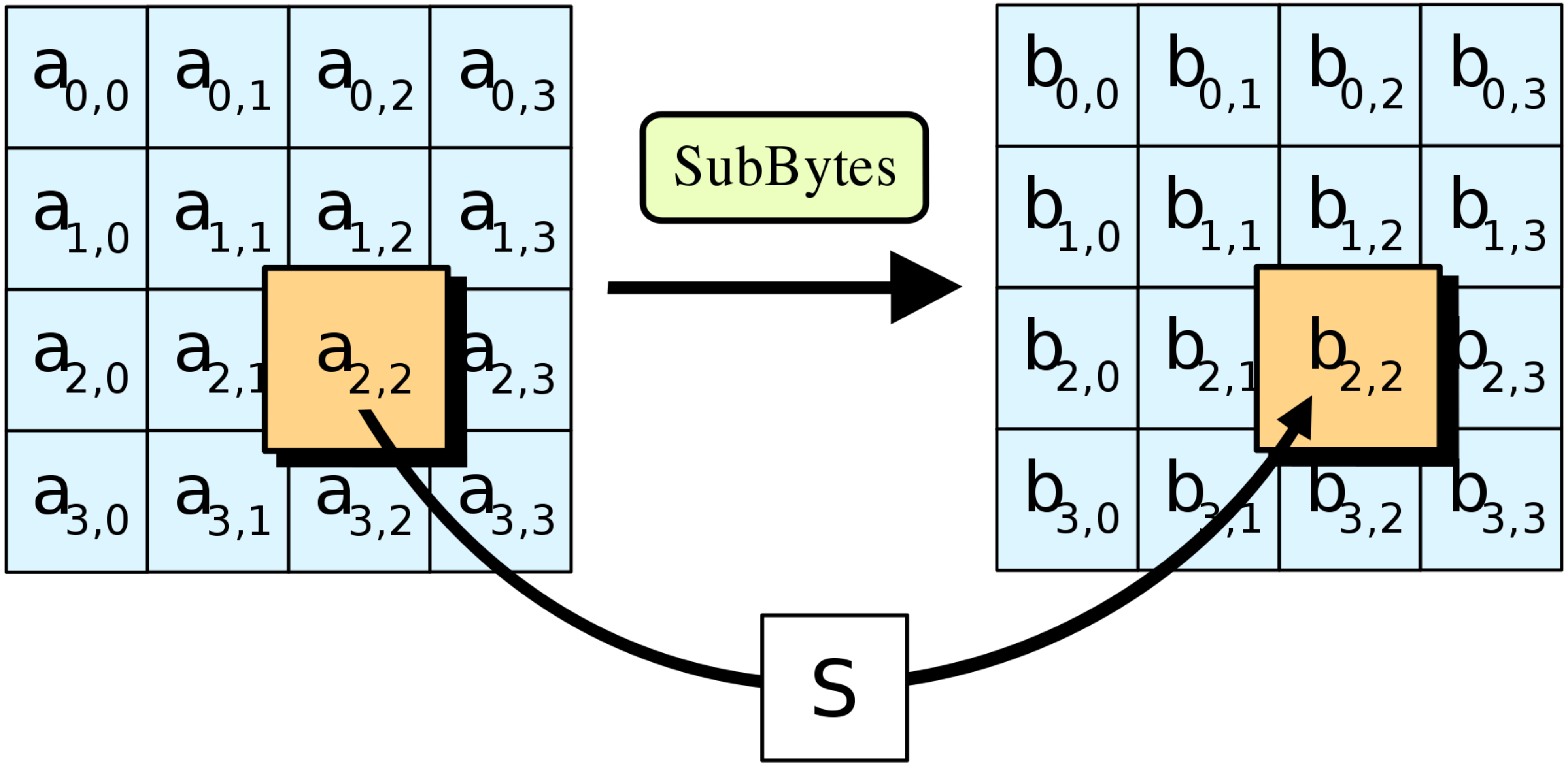


Best example: AES

- = Advanced Encryption Standard
- won the public NIST competition in 2001
- often in hardware (AES-NI in x86)
- blazing fast!
- still secure

Howto AES?

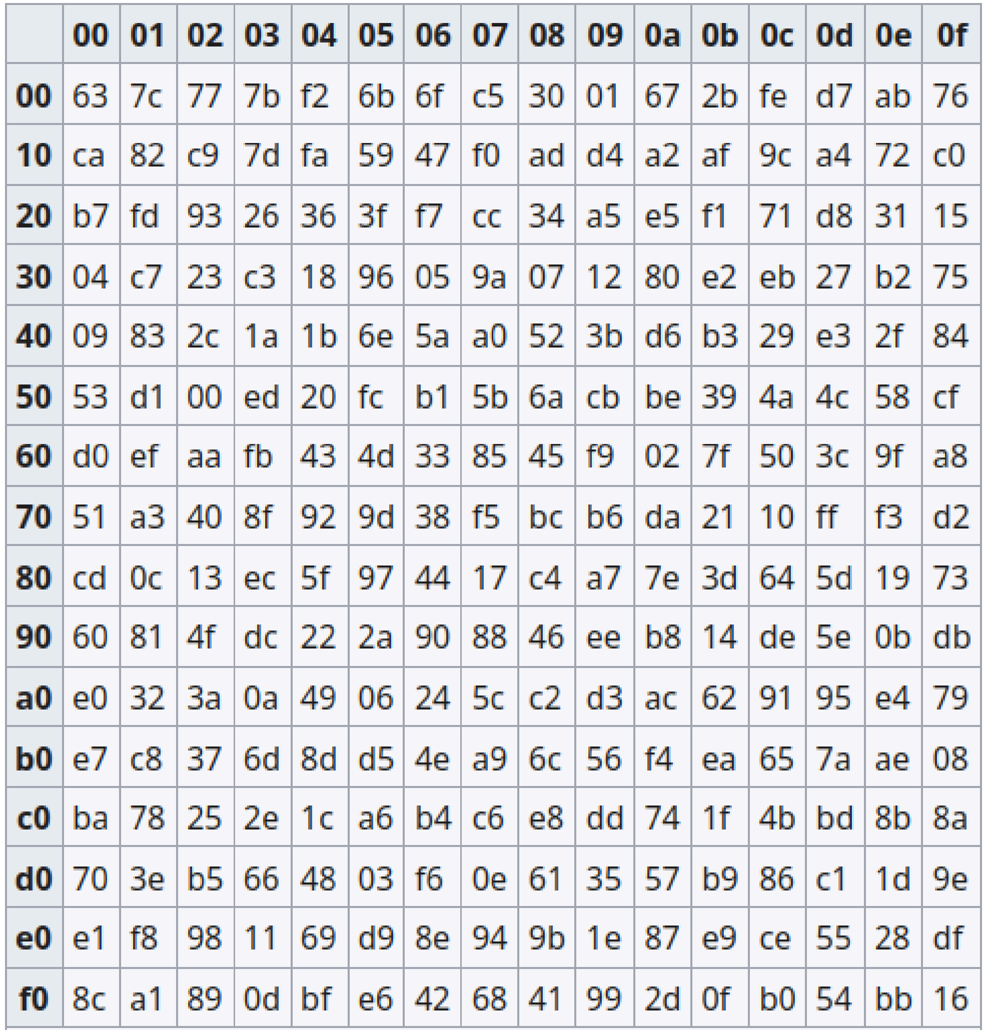
- block of 4x4 bytes
- one secret key, 128 bit
- 10 rounds
- same 4 steps: substitue, mix, shift, add round key
- (almost) symmetrical for en-/decryption



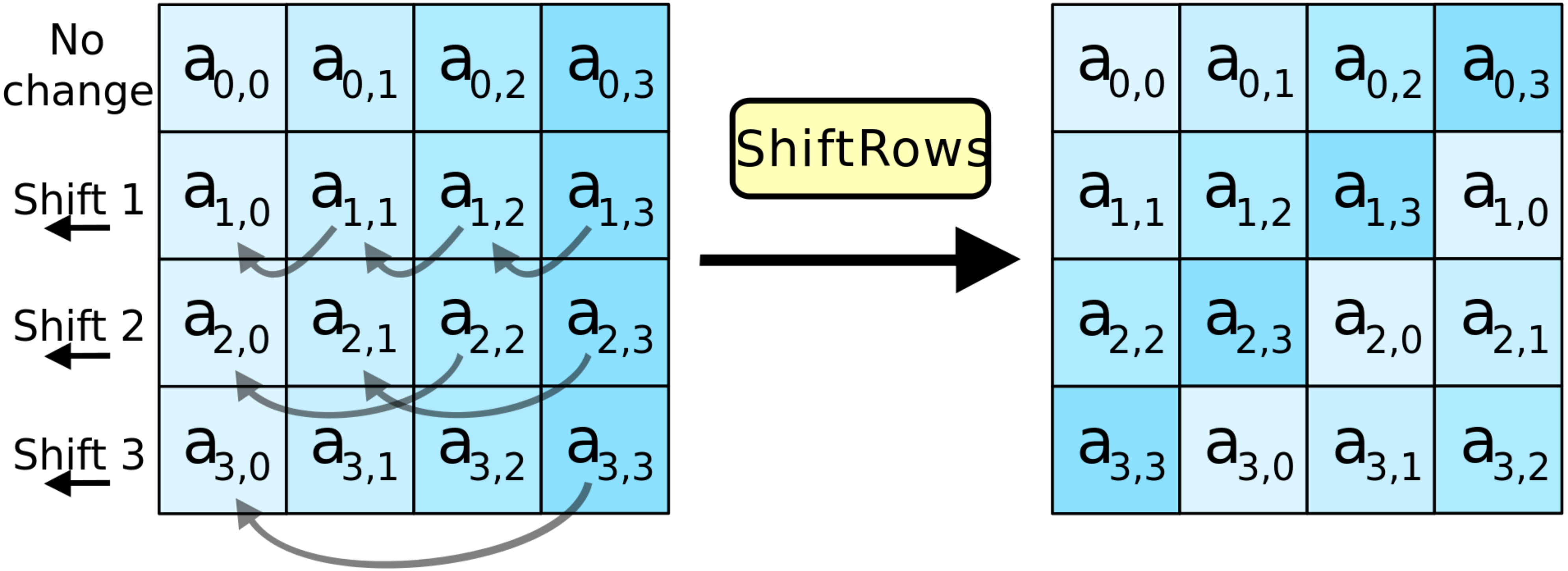
$$\begin{array}{cccc} a_{0,0} & a_{0,1} & a_{0,2} & a_{0,3} \\ a_{1,0} & a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,0} & a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,0} & a_{3,1} & a_{3,2} & a_{3,3} \end{array} \xrightarrow{\text{SubBytes}} \begin{array}{cccc} b_{0,0} & b_{0,1} & b_{0,2} & b_{0,3} \\ b_{1,0} & b_{1,1} & b_{1,2} & b_{1,3} \\ b_{2,0} & b_{2,1} & b_{2,2} & b_{2,3} \\ b_{3,0} & b_{3,1} & b_{3,2} & b_{3,3} \end{array}$$

$$a_{2,2} \xrightarrow{S} b_{2,2}$$

AES S-box



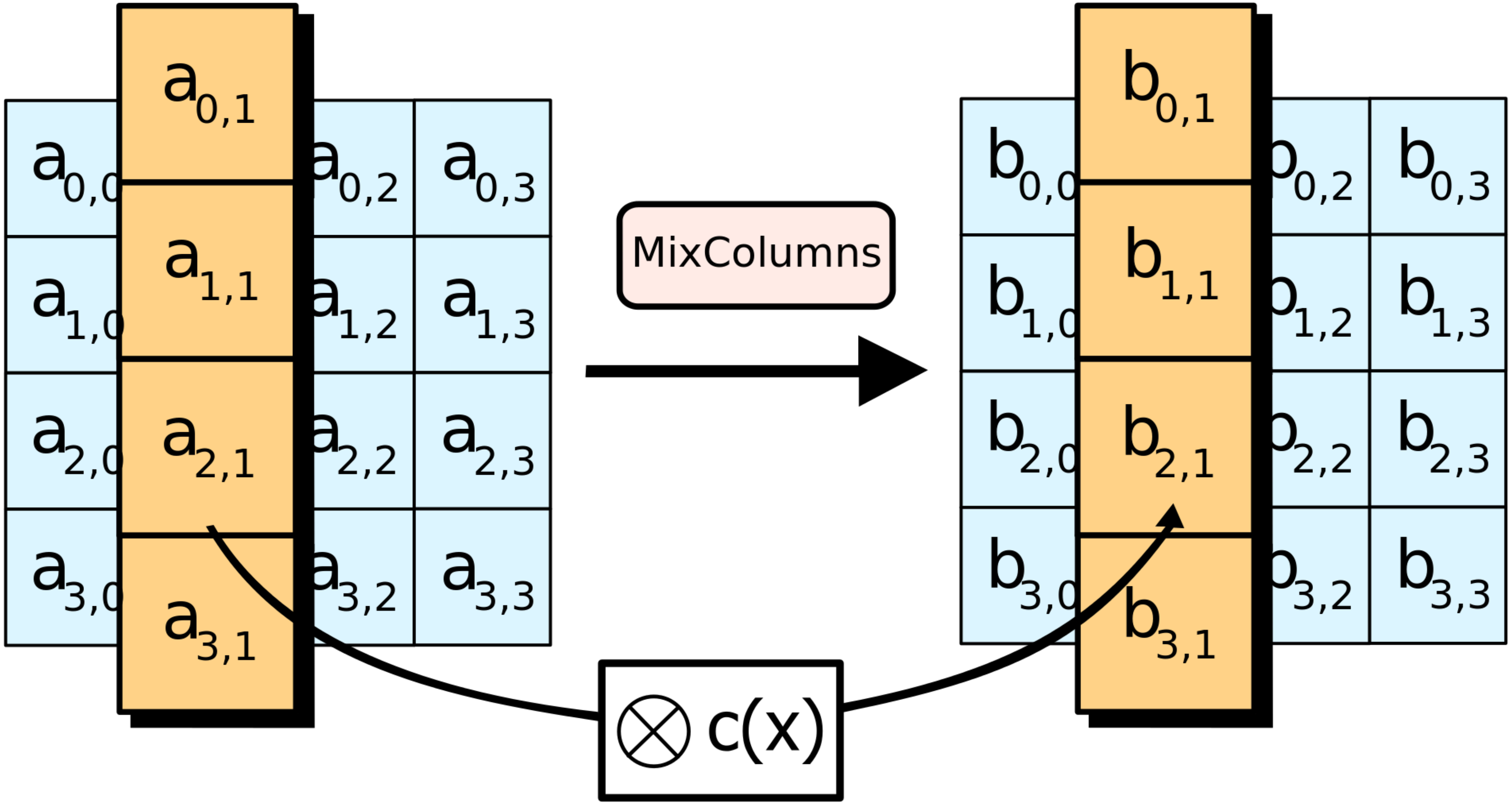
	00	01	02	03	04	05	06	07	08	09	0a	0b	0c	0d	0e	0f
00	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
10	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
20	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
30	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
40	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
50	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
60	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
70	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
80	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
90	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a0	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b0	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c0	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d0	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e0	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f0	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16



No change	$a_{0,0}$	$a_{0,1}$	$a_{0,2}$	$a_{0,3}$
Shift 1	$a_{1,0}$	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$
Shift 2	$a_{2,0}$	$a_{2,1}$	$a_{2,2}$	$a_{2,3}$
Shift 3	$a_{3,0}$	$a_{3,1}$	$a_{3,2}$	$a_{3,3}$

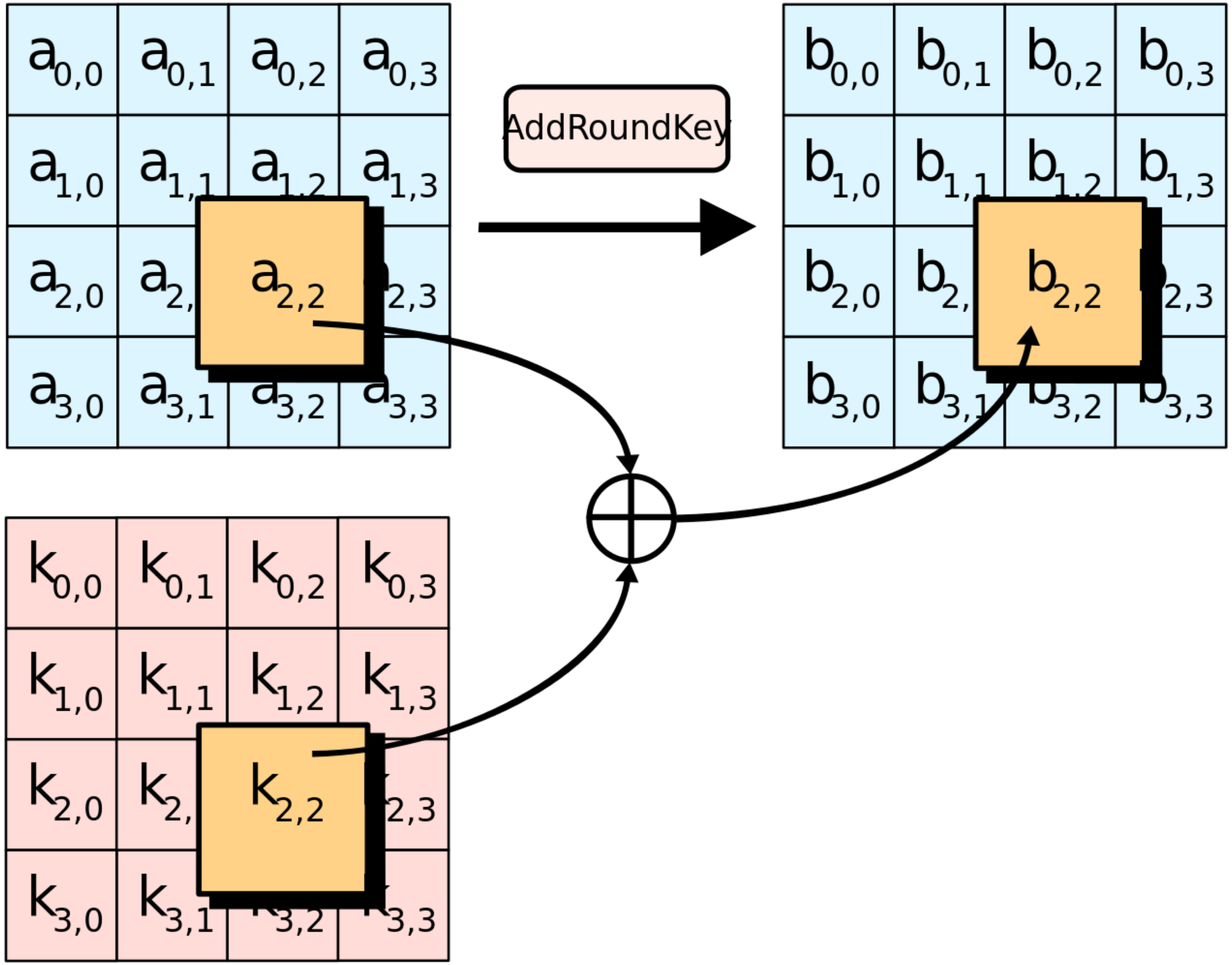
ShiftRows

$a_{0,0}$	$a_{0,1}$	$a_{0,2}$	$a_{0,3}$
$a_{1,1}$	$a_{1,2}$	$a_{1,3}$	$a_{1,0}$
$a_{2,2}$	$a_{2,3}$	$a_{2,0}$	$a_{2,1}$
$a_{3,3}$	$a_{3,0}$	$a_{3,1}$	$a_{3,2}$



$a_{0,0}$	$a_{0,1}$	$a_{0,2}$	$a_{0,3}$
$a_{1,0}$	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$
$a_{2,0}$	$a_{2,1}$	$a_{2,2}$	$a_{2,3}$
$a_{3,0}$	$a_{3,1}$	$a_{3,2}$	$a_{3,3}$

$b_{0,0}$	$b_{0,1}$	$b_{0,2}$	$b_{0,3}$
$b_{1,0}$	$b_{1,1}$	$b_{1,2}$	$b_{1,3}$
$b_{2,0}$	$b_{2,1}$	$b_{2,2}$	$b_{2,3}$
$b_{3,0}$	$b_{3,1}$	$b_{3,2}$	$b_{3,3}$



$a_{0,0}$	$a_{0,1}$	$a_{0,2}$	$a_{0,3}$
$a_{1,0}$	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$
$a_{2,0}$	$a_{2,1}$	$a_{2,2}$	$a_{2,3}$
$a_{3,0}$	$a_{3,1}$	$a_{3,2}$	$a_{3,3}$

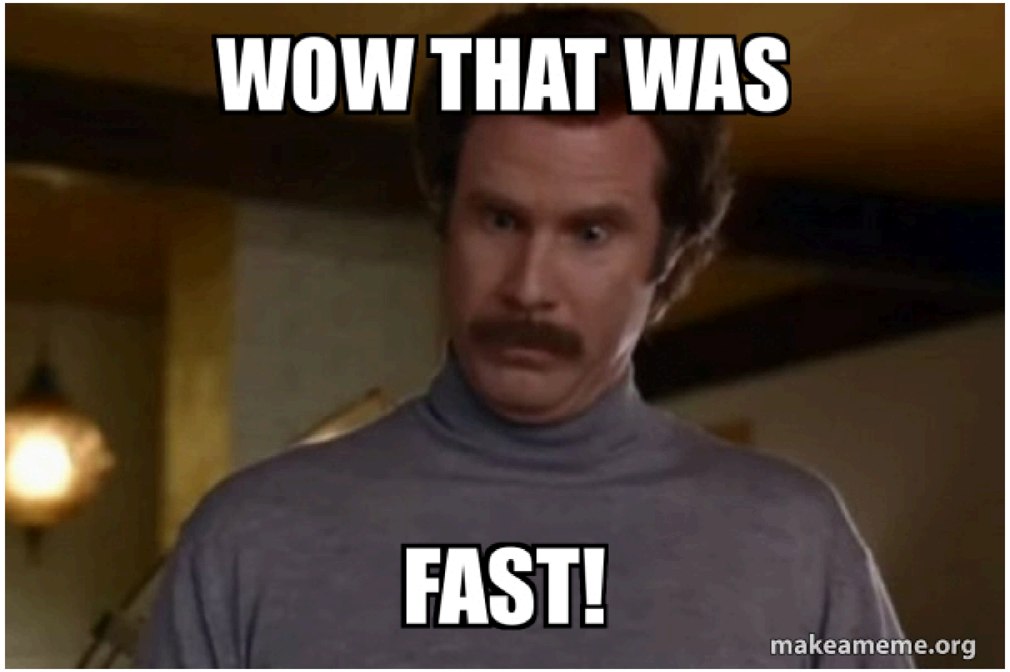
AddRoundKey

$b_{0,0}$	$b_{0,1}$	$b_{0,2}$	$b_{0,3}$
$b_{1,0}$	$b_{1,1}$	$b_{1,2}$	$b_{1,3}$
$b_{2,0}$	$b_{2,1}$	$b_{2,2}$	$b_{2,3}$
$b_{3,0}$	$b_{3,1}$	$b_{3,2}$	$b_{3,3}$

$k_{0,0}$	$k_{0,1}$	$k_{0,2}$	$k_{0,3}$
$k_{1,0}$	$k_{1,1}$	$k_{1,2}$	$k_{1,3}$
$k_{2,0}$	$k_{2,1}$	$k_{2,2}$	$k_{2,3}$
$k_{3,0}$	$k_{3,1}$	$k_{3,2}$	$k_{3,3}$

How fast is fast?

- >2 gbit AES-128
- per second
- on just one core
- on my notebook

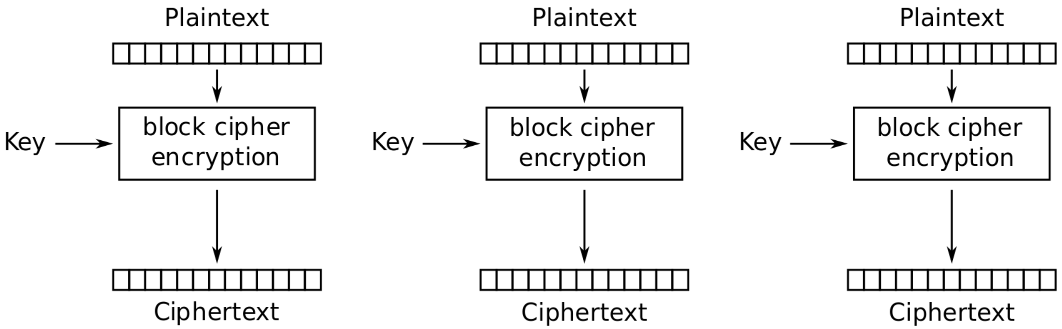


AES 192 & 256

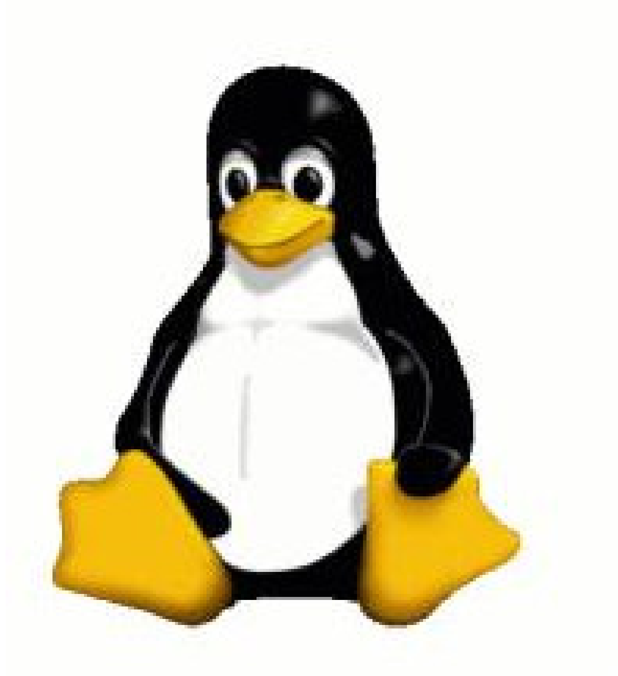
- AES standardized also with larger keys
- 192 bit and 256 bit
- 12 respectively 14 rounds then

"Mode of operation"

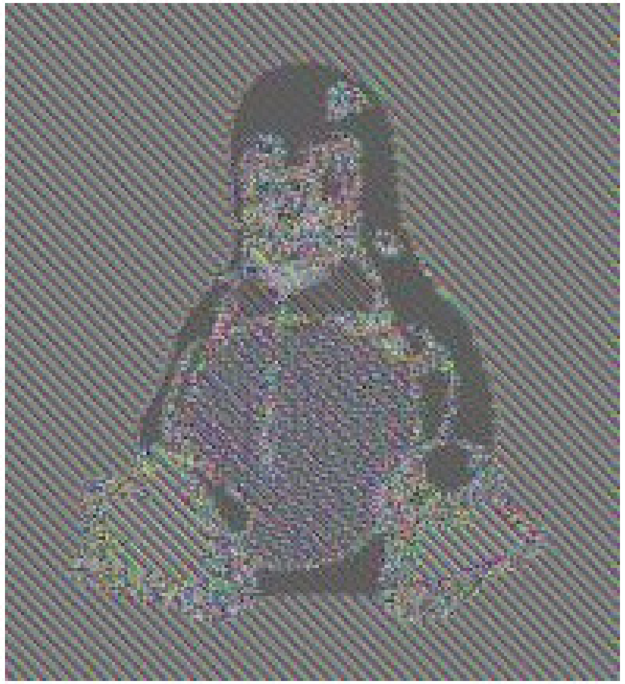
- plaintext > blocksize?
- encrypt each block individually is bad
- same plaintext blocks have same ciphertext



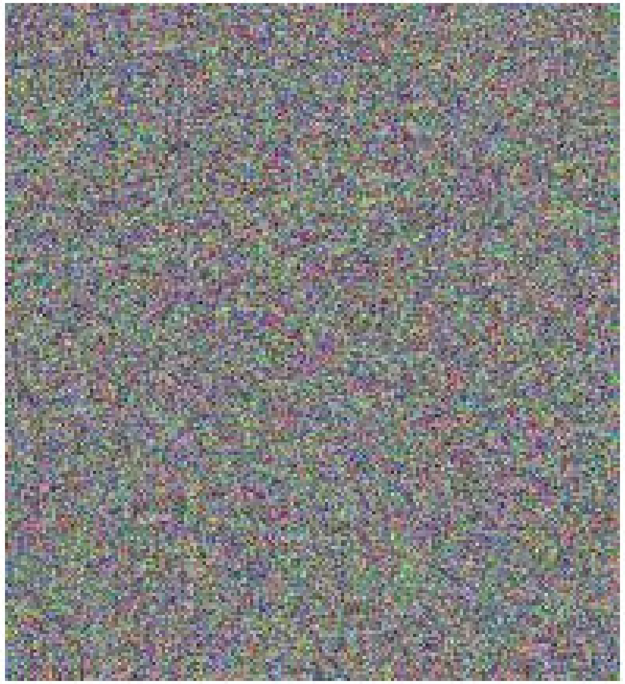
Electronic Codebook (ECB) mode encryption



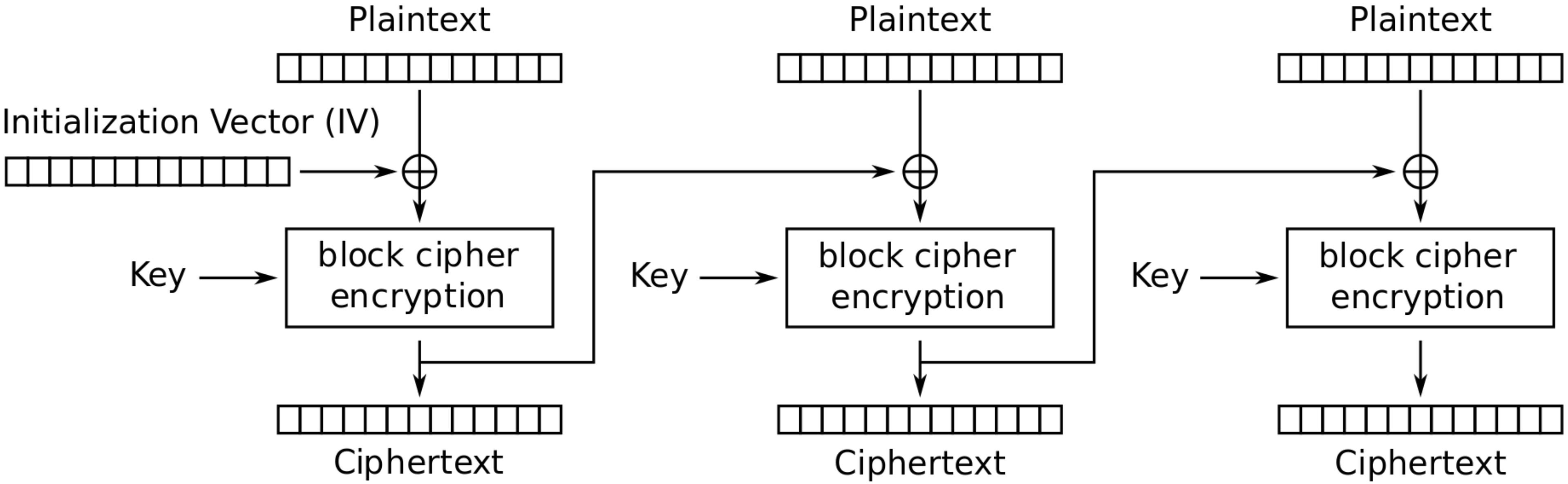
Original



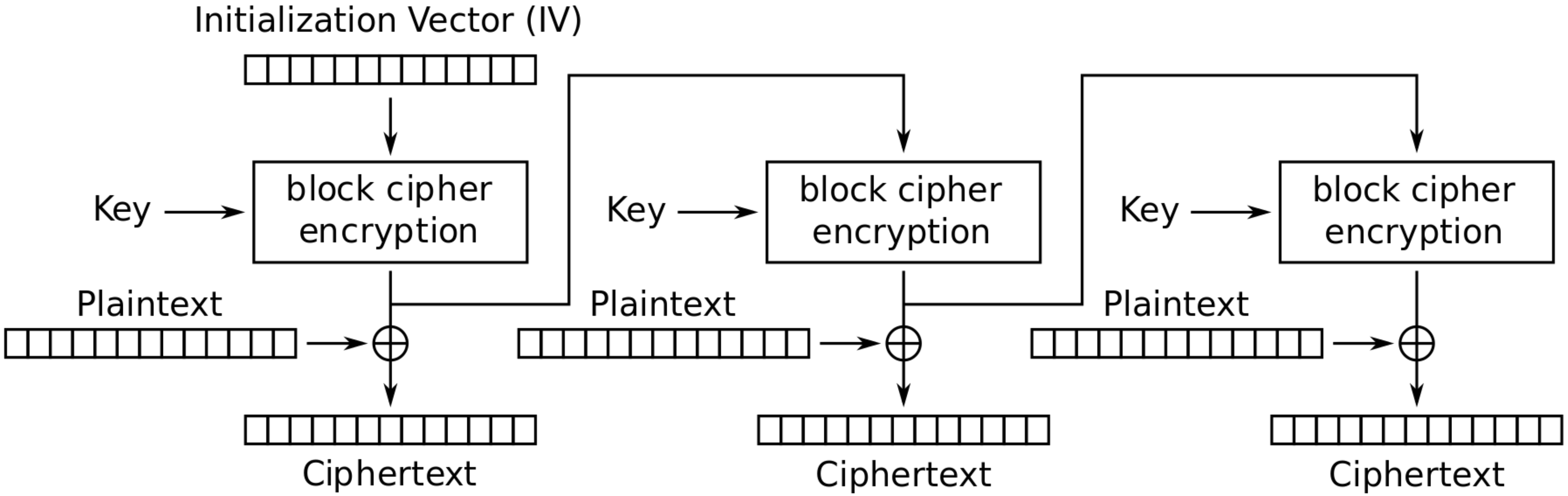
Encrypted using ECB mode



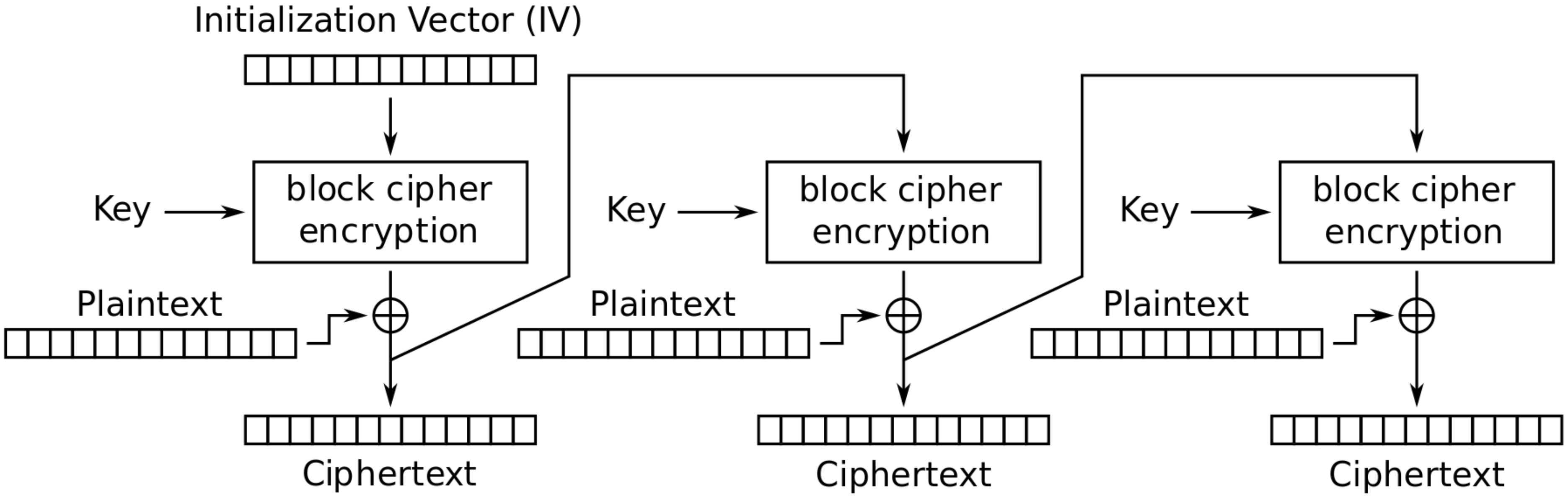
Modes other than ECB result in pseudo-randomness



Cipher Block Chaining (CBC) mode encryption



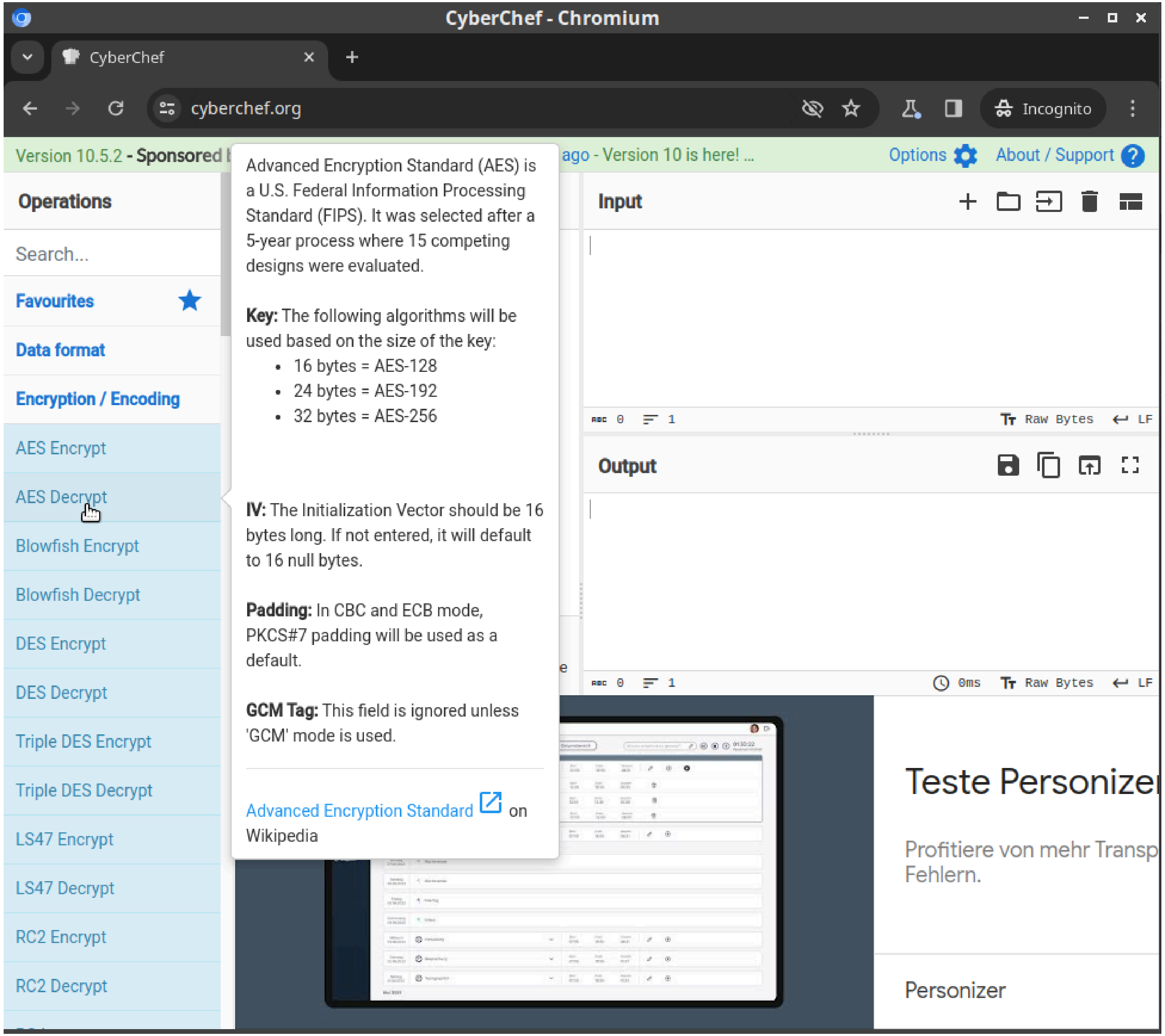
Output Feedback (OFB) mode encryption



Cipher Feedback (CFB) mode encryption

Were that all?

- XTS-AES: used for hard drive encryption
- AES-GCM: can be calculated in parallel & provides authenticity
- even more ...



Advanced Encryption Standard (AES) is a U.S. Federal Information Processing Standard (FIPS). It was selected after a 5-year process where 15 competing designs were evaluated.

Key: The following algorithms will be used based on the size of the key:

- 16 bytes = AES-128
- 24 bytes = AES-192
- 32 bytes = AES-256

IV: The Initialization Vector should be 16 bytes long. If not entered, it will default to 16 null bytes.

Padding: In CBC and ECB mode, PKCS#7 padding will be used as a default.

GCM Tag: This field is ignored unless 'GCM' mode is used.

Advanced Encryption Standard on Wikipedia

Lets test things

- plaintext is CyberSisters!!1
- key is 0102030405060708 (UTF8)
- IV is 0000000000000000 (UTF8)
- AES in CBC, Raw, Hex

- ciphertext should be 050aa1e9cdbca9040f5fe898cfb9934d

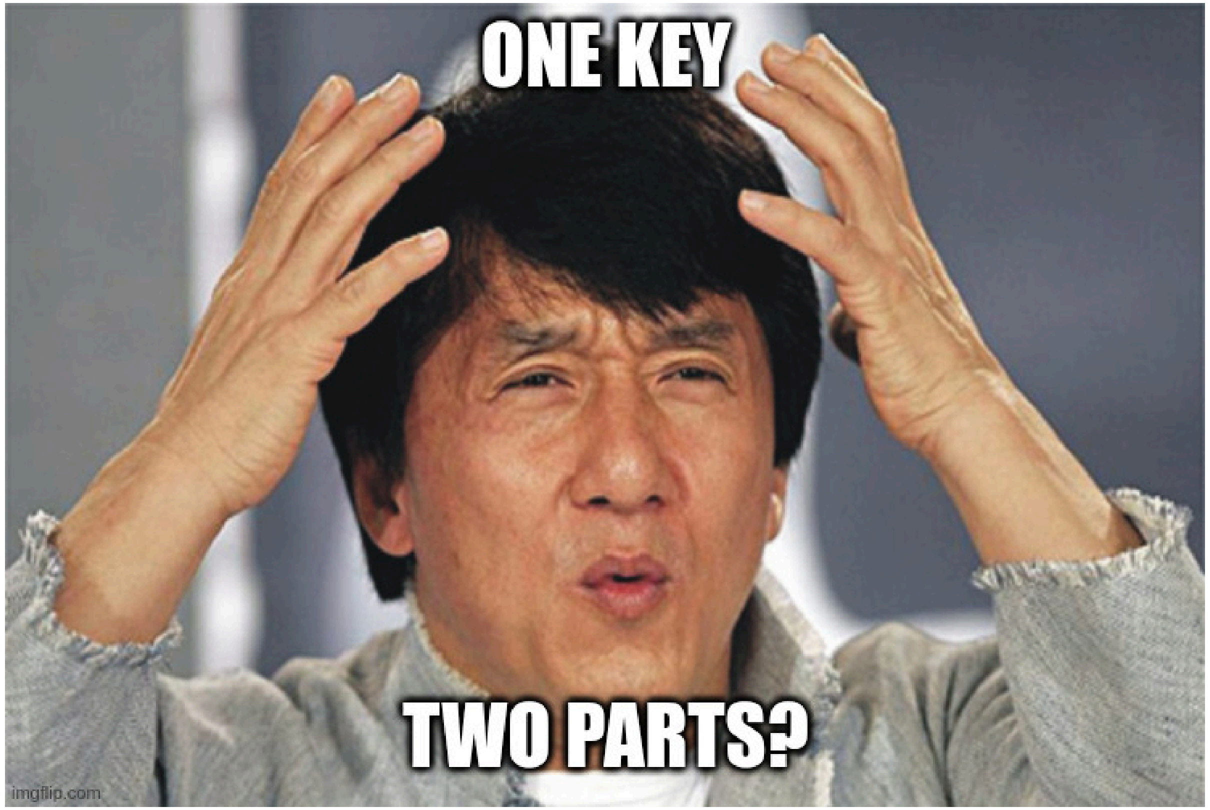
Lets test things

Same settings:

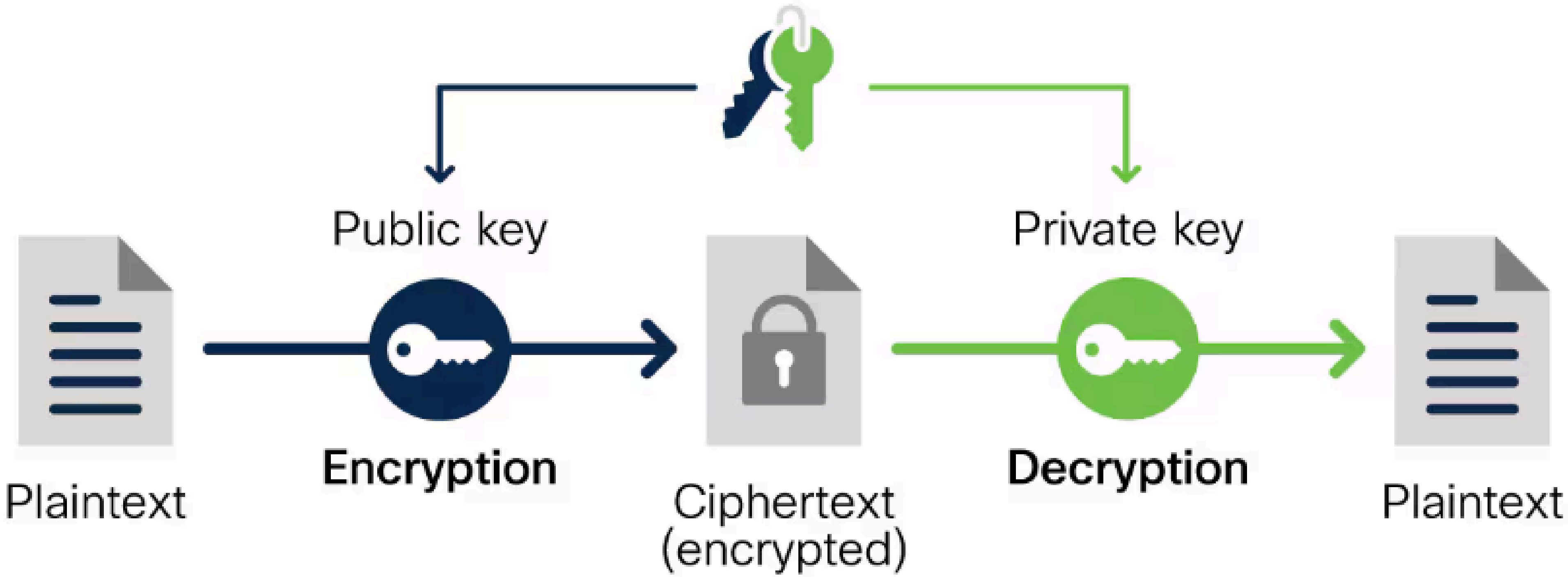
- what is ba7730f0758060ef48d93b993acb27f8775e41b0399194aabaae1716366826fa?

Asymmetric Cryptography

- key consists of two parts
- one is public, one is private
- entirely different math
- one cannot calculate the private part
  from the public one



Asymmetric encryption



Public Key Cryptography

- allows many nifty use-cases
- signatures!
- encryption (but slow)
- key exchange protocol

Diffie-Hellmen key exchange

- published in 1976
- two parties communicate
- obtain a shared secret
- only known to both of them
- attacker can observe all messages

RSA

- 1977: Rivest, Shamir, Adelman
- multiplication of two (large) primes
- modulo *n*
- 1024-4096 bit keys

RSA

used in e.g.:

- TLS, for e.g. HTTPS or VPN
- encrypted emails
- software signing
- IPSec, SSH, …

Elliptic curves (ECC)

- faster then RSA
- keys smaller
- different math!
- example: ed25519 signature scheme

Example Public Key Crypto

- SSH relies on public/private keys
- (passwords stink)
- ssh-keygen for interactive key pair creation

Example Public Key Crypto

- *ssh-keygen*
- *ssh-keygen -b 4096*
- *ssh-keygen -t ed25519*

Hashing

- when things are too big
- function for truncation
- create a unique digest
- one-way!
- short & deterministic

Hashing

given a cryptographic hash value:

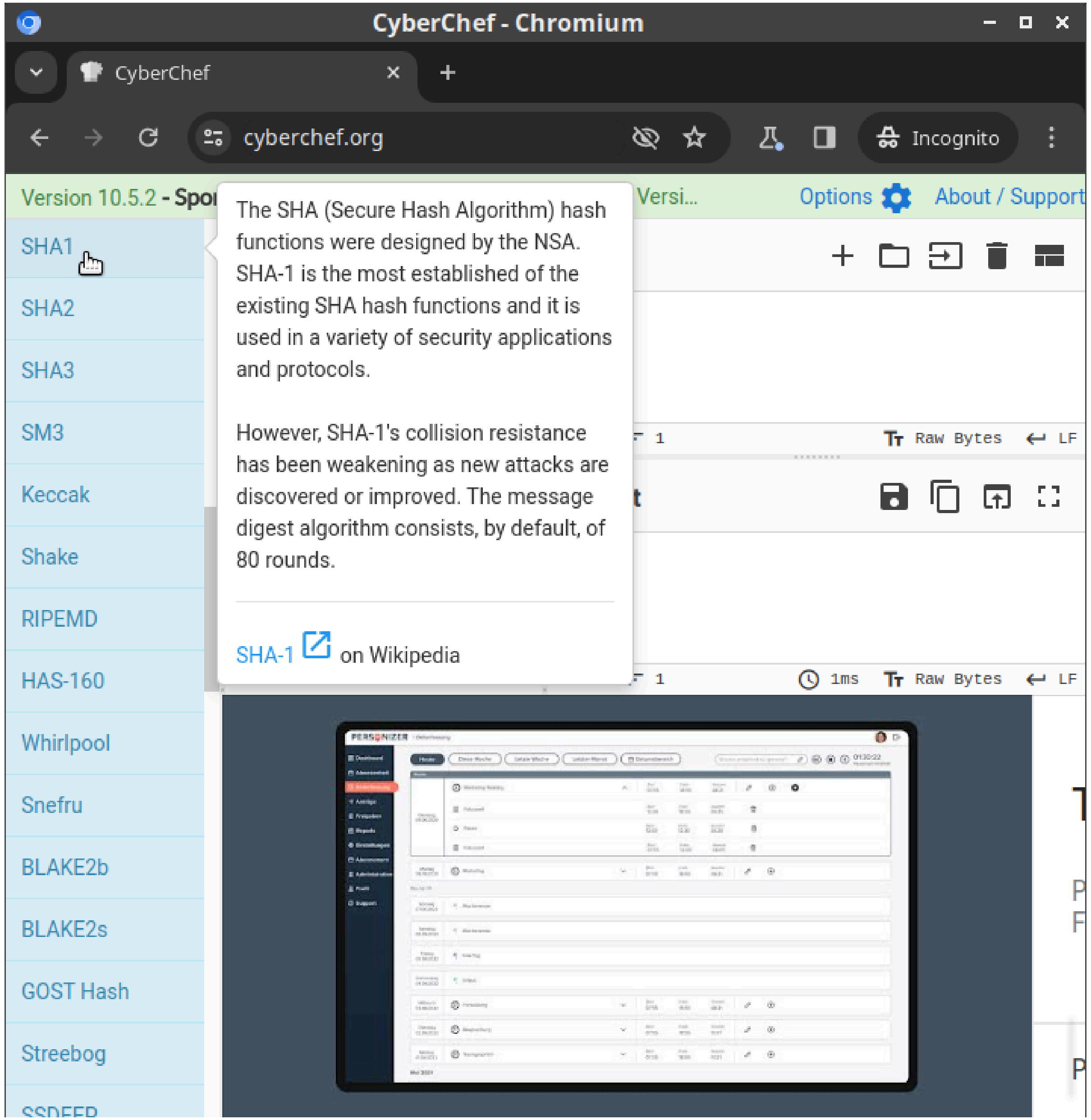
- one-way: should be hard to find original input
- should be hard to find other input with same hash value
- should be hard to find two inputs with same hash value

Hashing

- used in TLS, certificates, code singing, ...
- storing credentials = passwords
- file sharing
- proof-of-work

Hashing

- (MD5, SHA-1)
- SHA-256, SHA-512
- GOST, SM3
- Keccak
- bcrypt, argon2, ...



The SHA (Secure Hash Algorithm) hash functions were designed by the NSA. SHA-1 is the most established of the existing SHA hash functions and it is used in a variety of security applications and protocols.

However, SHA-1's collision resistance has been weakening as new attacks are discovered or improved. The message digest algorithm consists, by default, of 80 rounds.

SHA-1 on Wikipedia

That's all!