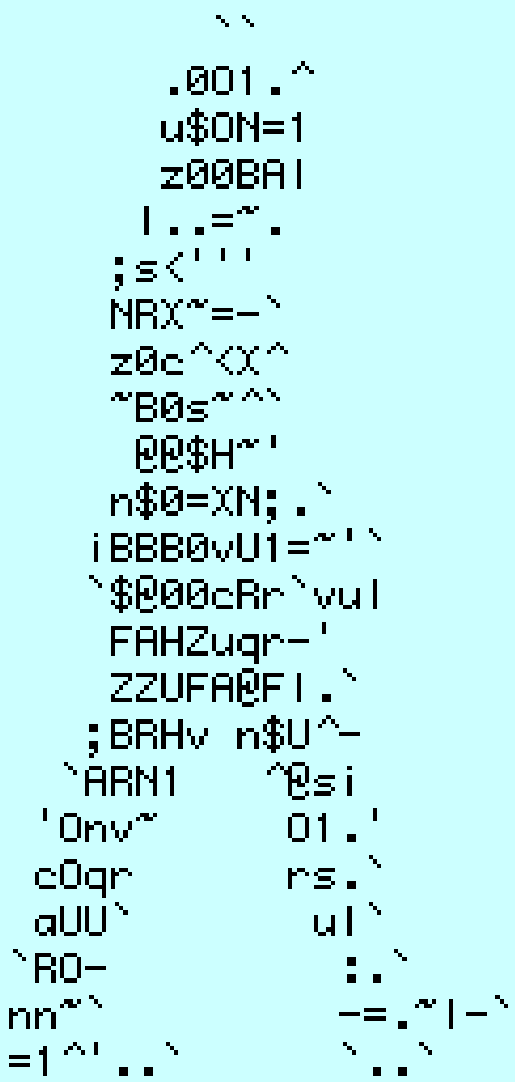


Crypto

for Cybersecurity

Michał Szychowiak, PhD

https://www.cs.put.poznan.pl/mszychowiak

Agenda

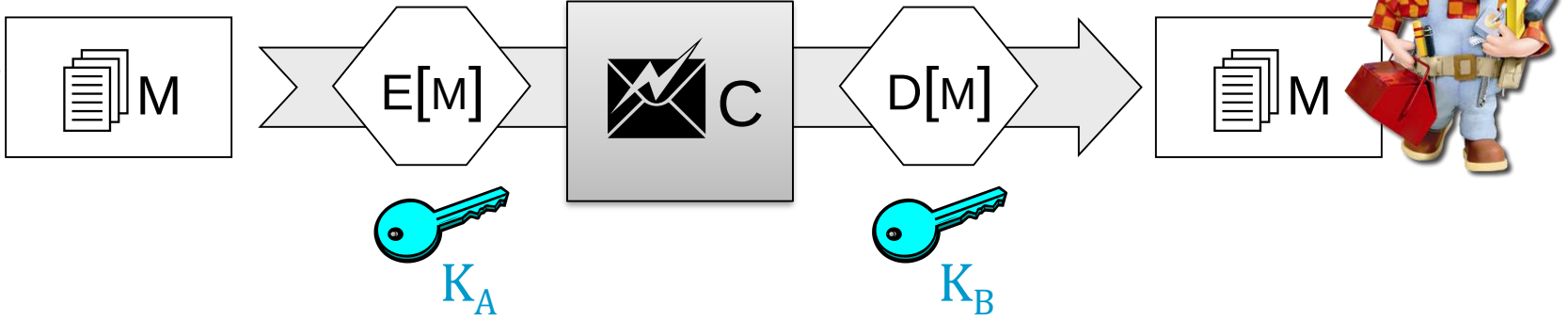
κρυπτός (kryptós) – hidden

γράφειν (gráphein) – writing

1. **Cryptology**
   - Cryptography
   - Cryptanalysis
2. **Symmetric/asymmetric crypto**
3. **Digital signatures**
4. **Key-distribution and certification (PKI)**
5. **Crypto applications**
6. **Near future of cryptology**
7. **Steganography**

Cryptography

clear-text message
plain-text message



$$E_{K_A}[M] = C \quad \Rightarrow \quad D_{K_B}[C] = M$$

Cryptanalysis

clear-text message
plain-text message

M

cipher-text

C

M

K_B

M

cryptanalyst

Cryptanalysis

Crypto attacks:

- brute-force cryptanalysis
- known-text cryptanalysis (w/ *cribs*)
- chosen-plaintext cryptanalysis
- black-box / white-box analysis (→ *white-box cryptography*)
- side-channel attacks on hardware crypto-devices
- …

Auguste Kerckhoffs

"It must not be required to keep the system secret, and it must be able to fall into the hands of the enemy without harm."

Journal des Sciences Militaires, Janvier 1883: La Cryptographie Militaire

Kerckhoffs's principle

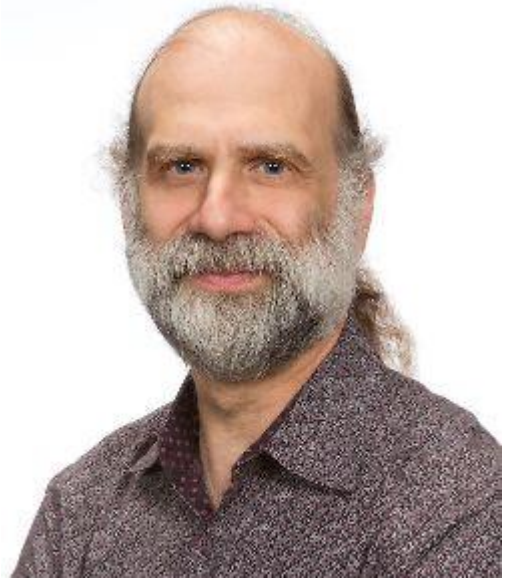
A cryptosystem should be secure even if everything about the system, except the key, is public knowledge

= no “security by obscurity”

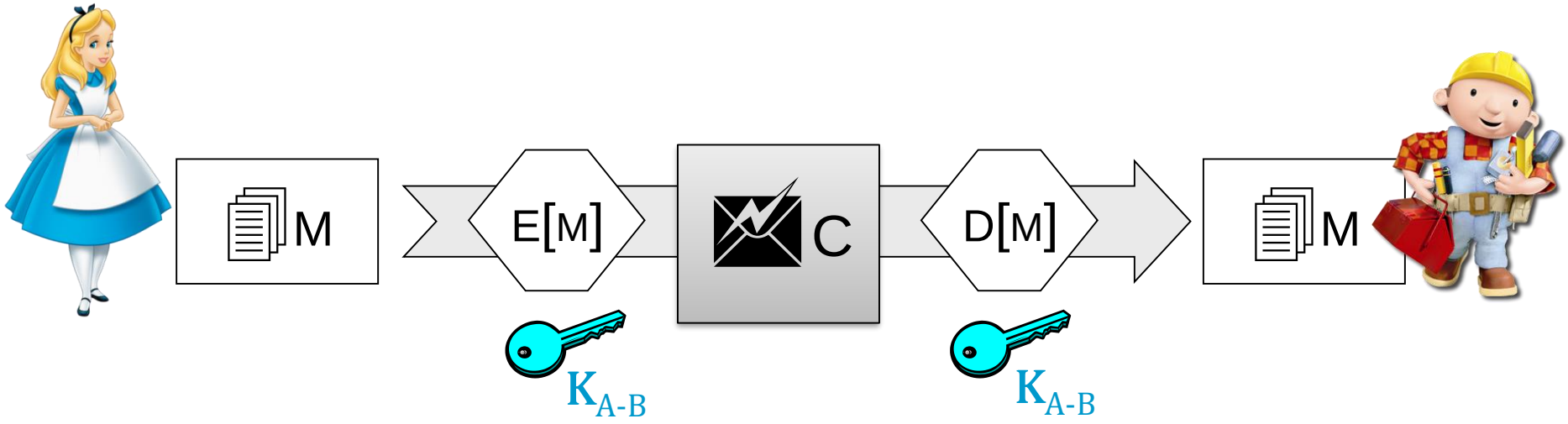
Schneier's principle ☺

"Anyone who creates his or her own cryptographic primitives is either a genius or a fool. Given the genius/fool ratio for our species, the odds aren't very good."

Bruce Schneier



Symmetric crypto



$$E_K[M] = C \quad \Rightarrow \quad D_K[C] = M$$

Symmetric crypto

Main problems:

- key secrecy
- key distribution
- key scale

Symmetric crypto

Main problems:

Symmetric crypto

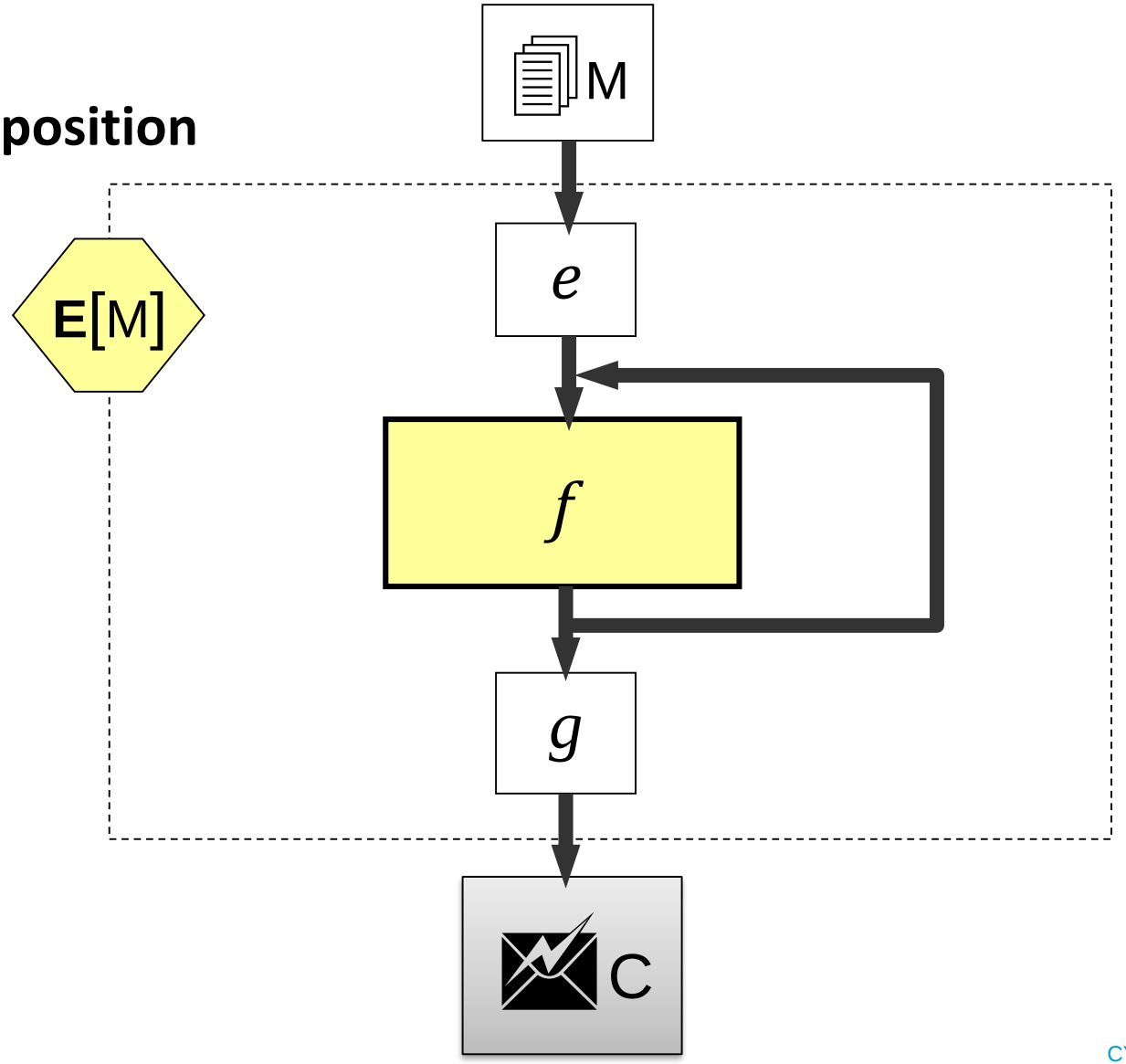
Main problems:

- key secrecy
- key distribution
- key scale

- authenticity?
- non-repudiation?

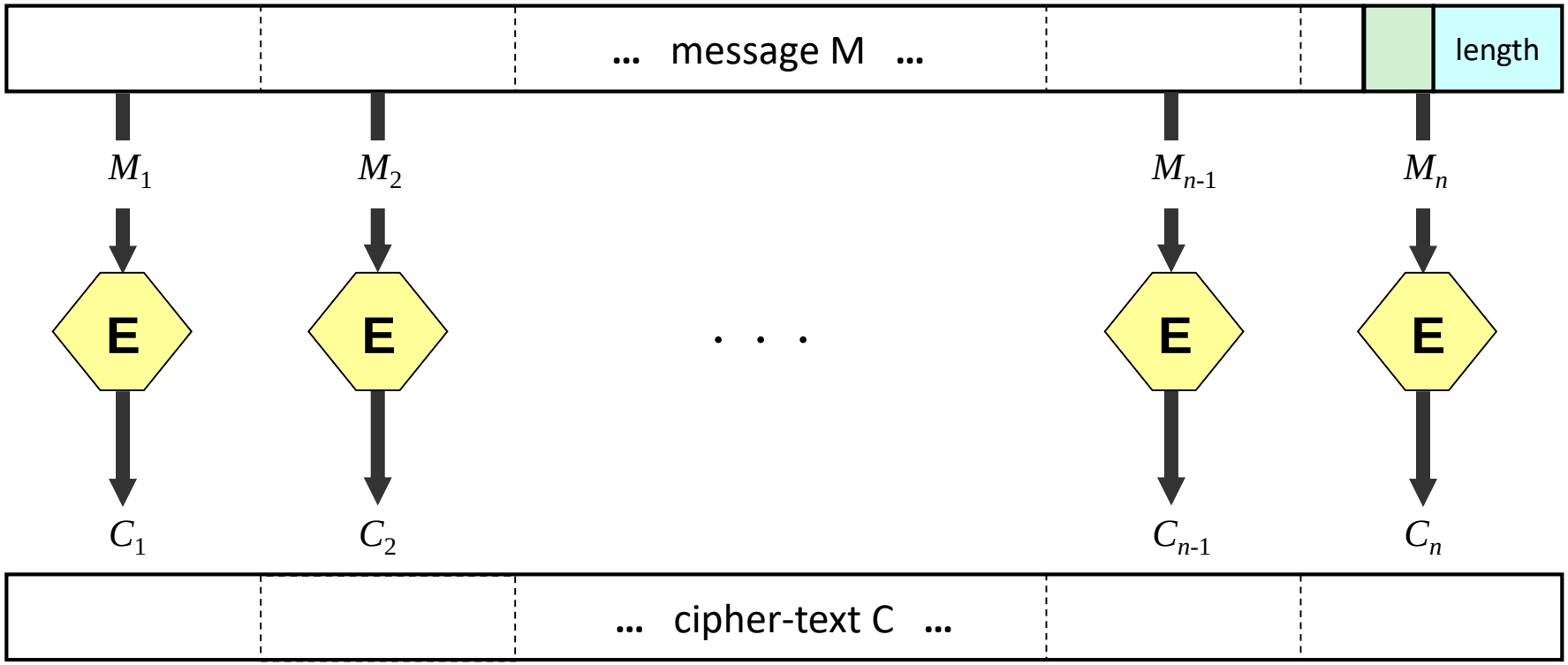
Algorithms

Basic composition



Algorithms

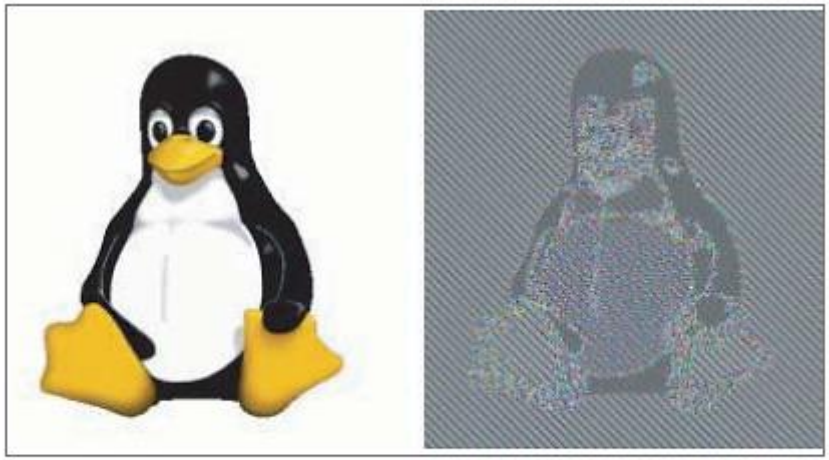
Block cipher



Algorithms

Block cipher

- to simple = easy to crack, e.g. ECB (*Electronic Code Book*) mode:

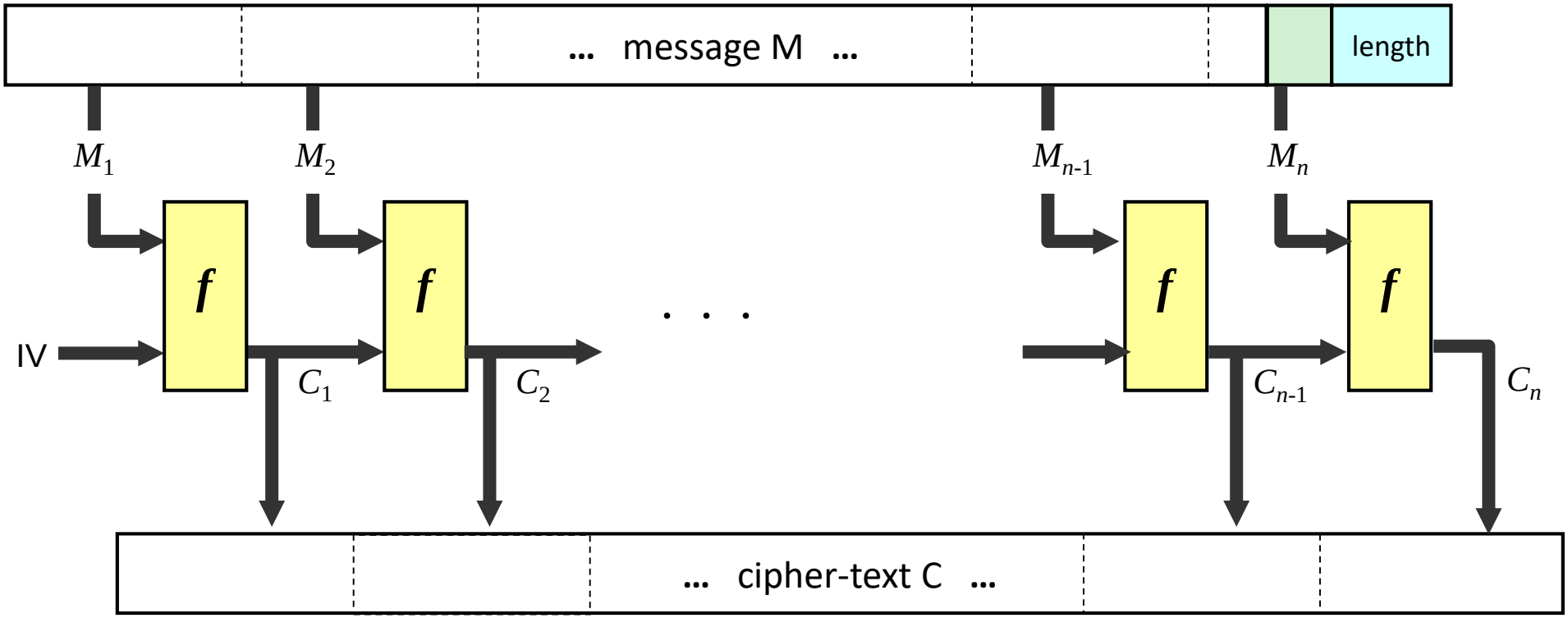


Patterns within plaintext encrypted using an ECB cipher may be visible within the ciphertext.

source: “The Web Application Hacker's Handbook”

Algorithms

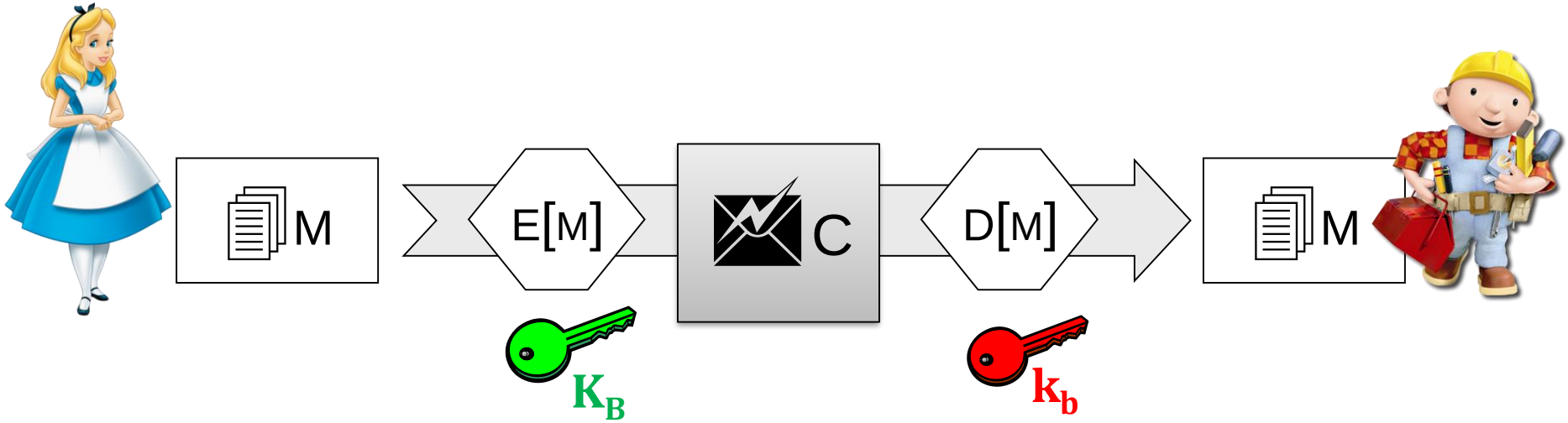
Feedback modes



V·T·E	Block ciphers (security summary)
Common algorithms	AES · Blowfish · DES (internal mechanics, Triple DES) · Serpent · Twofish
Less common algorithms	Camellia · CAST-128 · GOST · IDEA · RC2 · RC5 · RC6 · SEED · ARIA · Skipjack · TEA · XTEA
Other algorithms	3-Way · Akelarre · Anubis · BaseKing · BassOmatic · BATON · BEAR and LION · CAST-256 · Chiasmus · CIKS-1 · CIPHERUNICORN-A · CIPHERUNICORN-E · CLEFIA · CMEA · Cobra · COCONUT98 · Crab · Cryptomeria/C2 · CRYPTON · CS-Cipher · DEAL · DES-X · DFC · E2 · FEAL · FEA-M · FROG · G-DES · Grand Cru · Hasty Pudding cipher · Hierocrypt · ICE · IDEA NXT · Intel Cascade Cipher · Iraqi · Kalyna · KASUMI · KeeLoq · KHAZAD · Khufu and Khafre · KN-Cipher · Kuznyechik · Ladder-DES · Libelle · LOKI (97, 89/91) · Lucifer · M6 · M8 · MacGuffin · Madryga · MAGENTA · MARS · Mercy · MESH · MISTY1 · MMB · MULTI2 · MultiSwap · New Data Seal · NewDES · Nimbus · NOEKEON · NUSH · PRESENT · Prince · Q · RC6 · REDOC · Red Pike · S-1 · SAFER · SAVILLE · SC2000 · SHACAL · SHARK · Simon · SM4 · Speck · Spectr-H64 · Square · SXAL/MBAL · Threefish · Treyfer · UES · xmx · XXTEA · Zodiac
Design	Feistel network · Key schedule · Lai–Massey scheme · Product cipher · S-box · P-box · SPN · Confusion and diffusion · Avalanche effect · Block size · Key size · Key whitening (Whitening transformation)
Attack (cryptanalysis)	Brute-force (EFF DES cracker) · MITM (Biclique attack · 3-subset MITM attack) · Linear (Piling-up lemma) · Differential (Impossible · Truncated · Higher-order) · Differential-linear · Distinguishing (Known-key) · Integral/Square · Boomerang · Mod *n* · Related-key · Slide · Rotational · Side-channel (Timing · Power-monitoring · Electromagnetic · Acoustic · Differential-fault) · XSL · Interpolation · Partitioning · Rubber-hose · Black-bag · Davies · Rebound · Weak key · Tau · Chi-square · Time/memory/data tradeoff
Standardization	AES process · CRYPTREC · NESSIE
Utilization	Initialization vector · Mode of operation · Padding

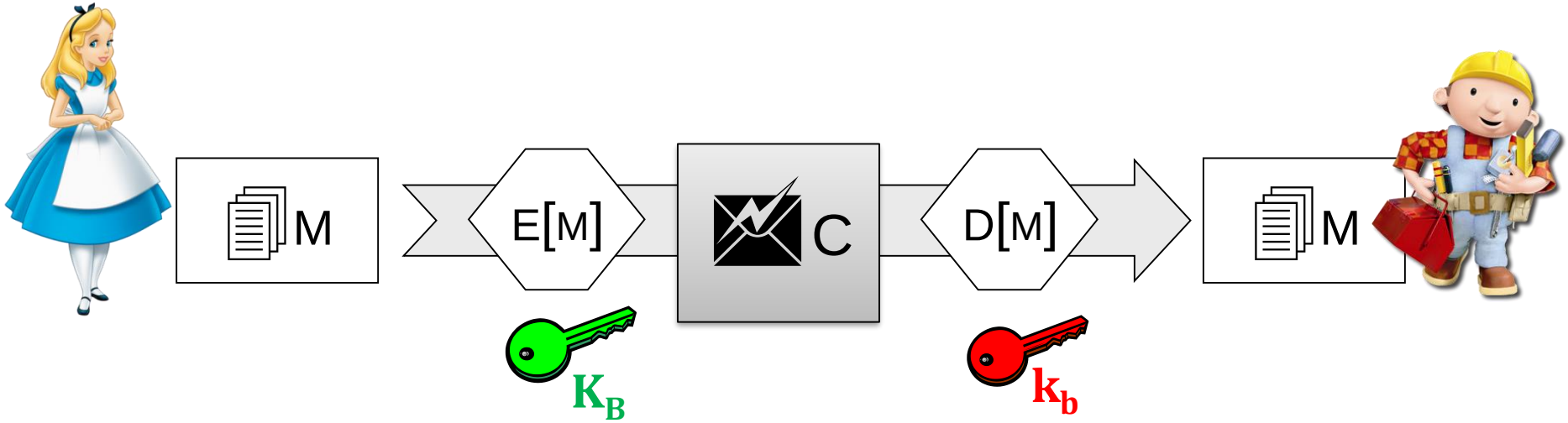
V·T·E		Stream ciphers
Widely used ciphers		RC4 · block ciphers in stream mode · ChaCha
eSTREAM Portfolio	Software	HC-256 · Rabbit · Salsa20 · SOSEMANUK
	Hardware	Grain · MICKEY · Trivium
Other ciphers		A5/1 · A5/2 · Achterbahn · E0 · F-FCSR · FISH · ISAAC · MUGI · Panama · Phelix · Pike · Py · QUAD · Scream · SEAL · SNOW · SOBER · SOBER-128 · VEST · VMPC · WAKE
Theory		shift register · LFSR · NLFSR · shrinking generator · T-function · IV
Attacks		correlation attack · correlation immunity · stream cipher attacks

Asymmetric crypto



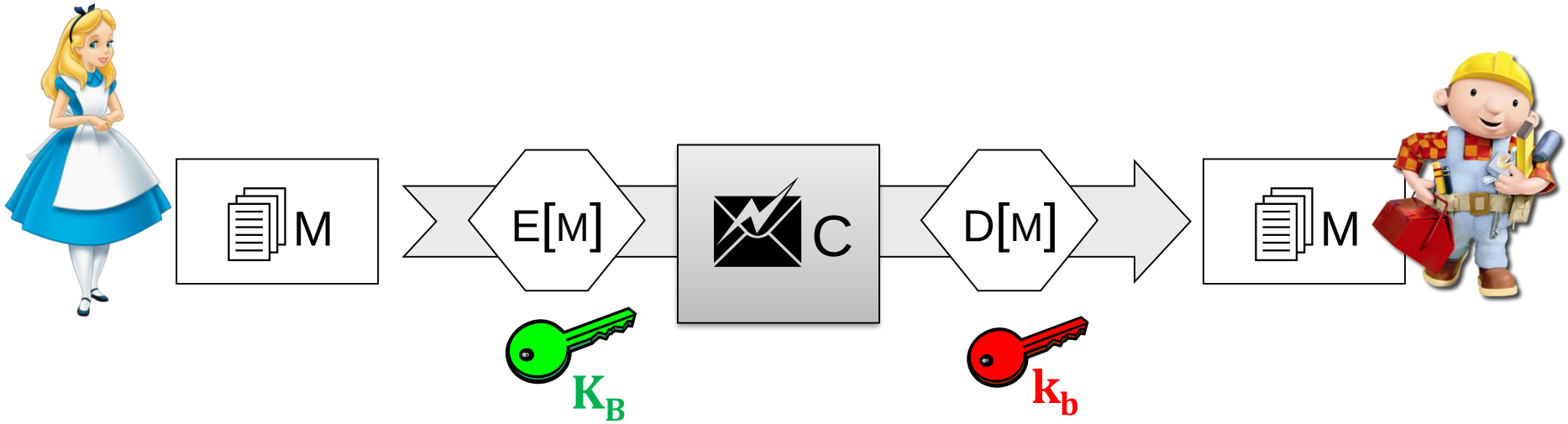
$$E_{K_B}[M] = C \quad \Rightarrow \quad D_{k_b}[C] = M$$

Asymmetric crypto



$$D_k[E_K[M]] = M$$

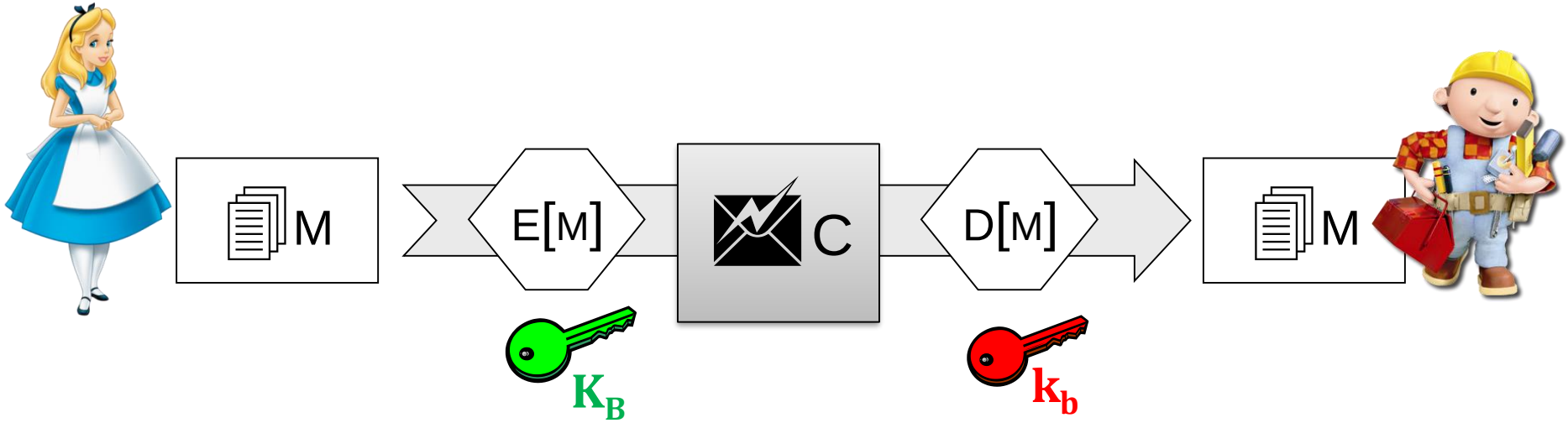
Asymmetric crypto

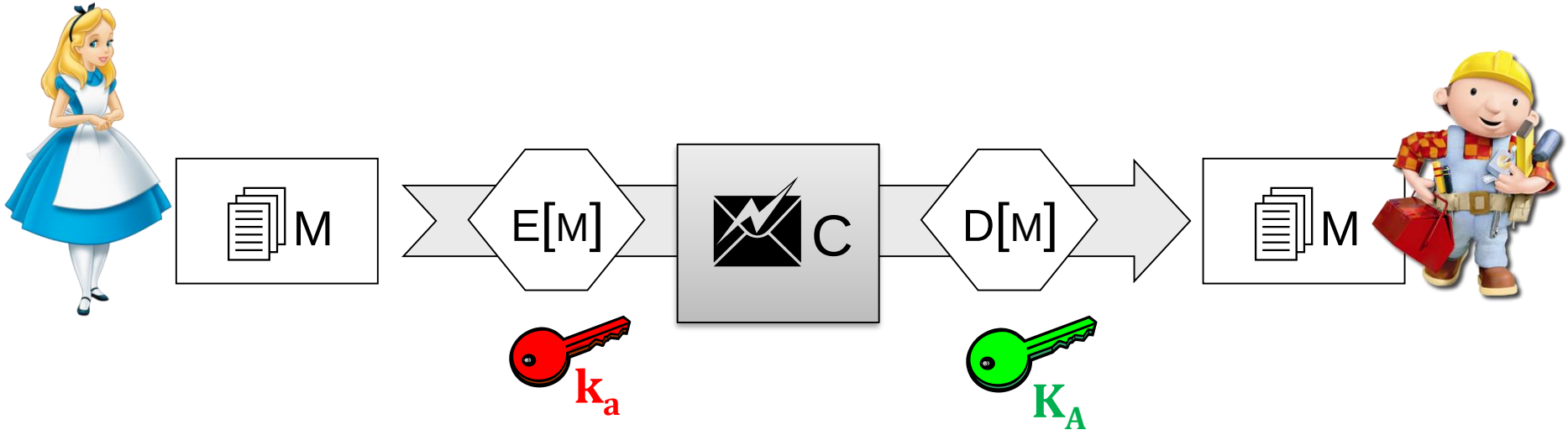


$$D_k[E_K[M]] = M$$

$$D_K[E_k[M]] = M$$

Asymmetric crypto





Symmetric crypto

Main problems:

- key secrecy
- key distribution
- key scale

- authenticity?
- non-repudiation?

Asymmetric crypto

Main problems:

- key secrecy ✓
- key distribution ✓
- key scale ✓

- authenticity? ✓
- non-repudiation? ✓

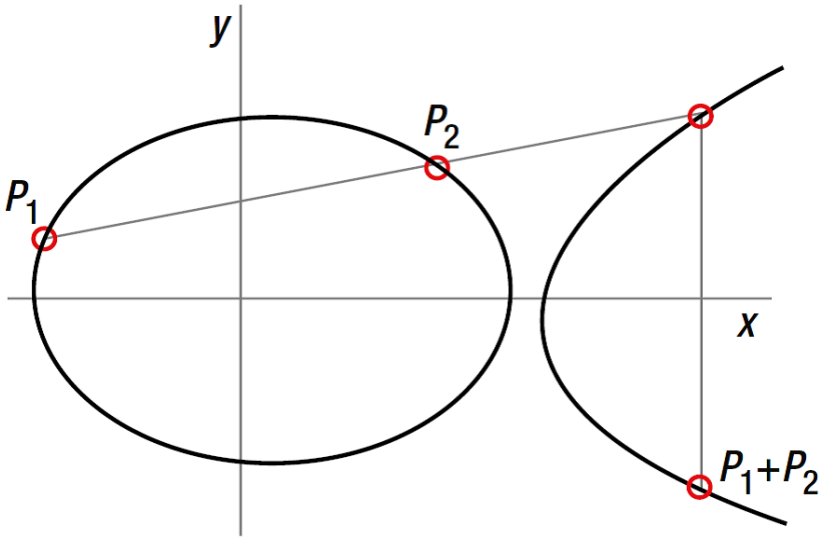
Algorithms

RSA (Rivest–Shamir–Adleman)

ELG (ElGamal)

ECC (Elliptic Curve Cryptography):

- Curve25519
- Edwards25519
- Curve448
- Curve41417
- Brainpool



➔https://arstechnica.com/information-technology/2013/10/a-relatively-easy-to-understand-primer-on-elliptic-curve-cryptography/

Checkpoint

Symmetric crypto:

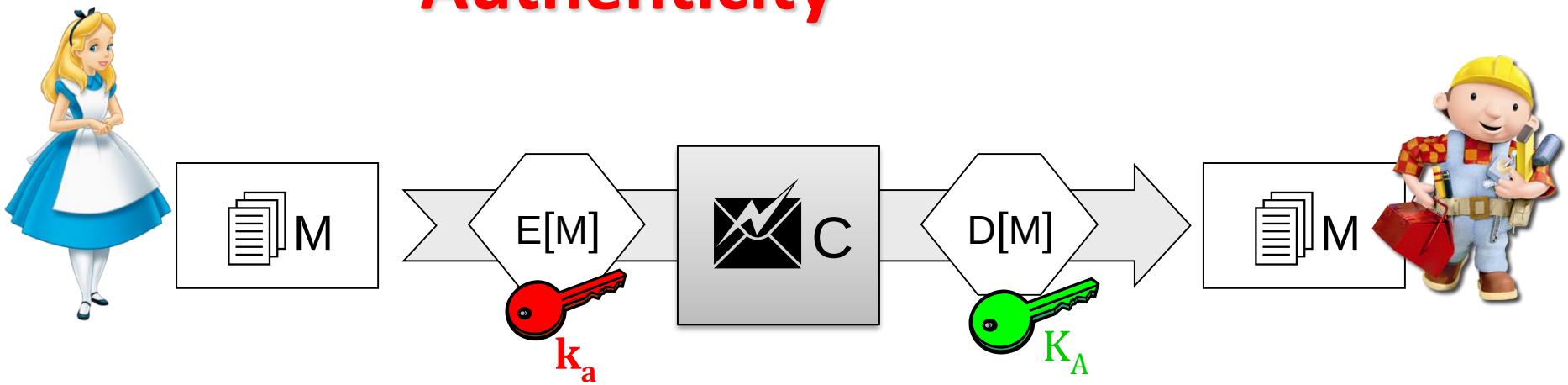
- key distribution sucks
- no authenticity
- forget about nonrepudiation

Asymmetric crypto:

- easy key distribution
- out-of-the-box authenticity
- possible nonrepudiation

Where is the hook?

Authenticity



Authenticity

M

h[M]

H

E[H]

k_a

M

+

MAC

D[H]

K_A

M

h[M]

H

= ?

H

MAC = *Message Authentication Code*

Authenticity

Digest

- one-way hash function $h[\mathrm{M}]$
- giving $\mathrm{H}=h[\mathrm{M}]$ of a fixed small size n
- possibly with additional input (*salt*, *challenge*, …)

+ crypto

- asymmetric: MAC – *message authentication code*
- symmetric: ICV – *integrity check value*

Message Digest

Not quite brand-new in IT

- *checksum*
- *data integrity check*
- *contraction function*
- *fingerprint, thumbprint*
- ...

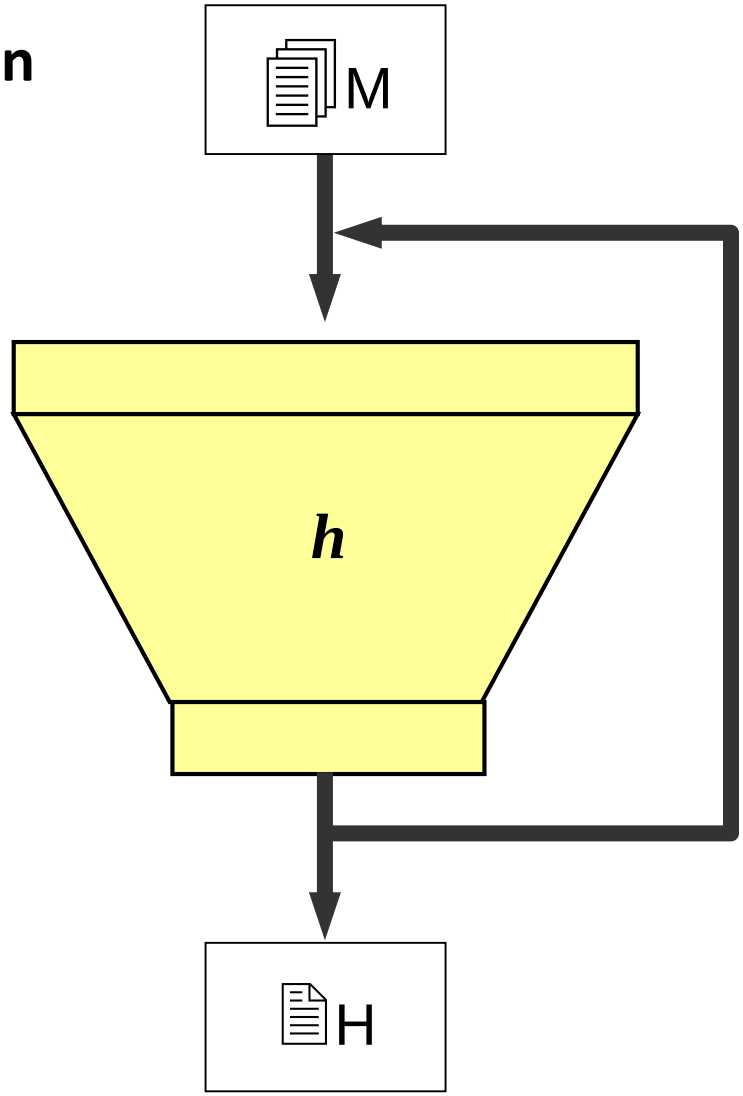
Message Digest

Required properties

- compression: |H| < |M|
- easy calculation
- preimage resistance: given H, difficult to find M:: H=h[M]
- 2nd preimage resistance: given M_1, difficult to find M_2:: $h[M_1] = h[M_2]$
- collision resistance: difficult to find (M_1, M_2)::
  $M_1 \neq M_2$ & $h[M_1] = h[M_2]$

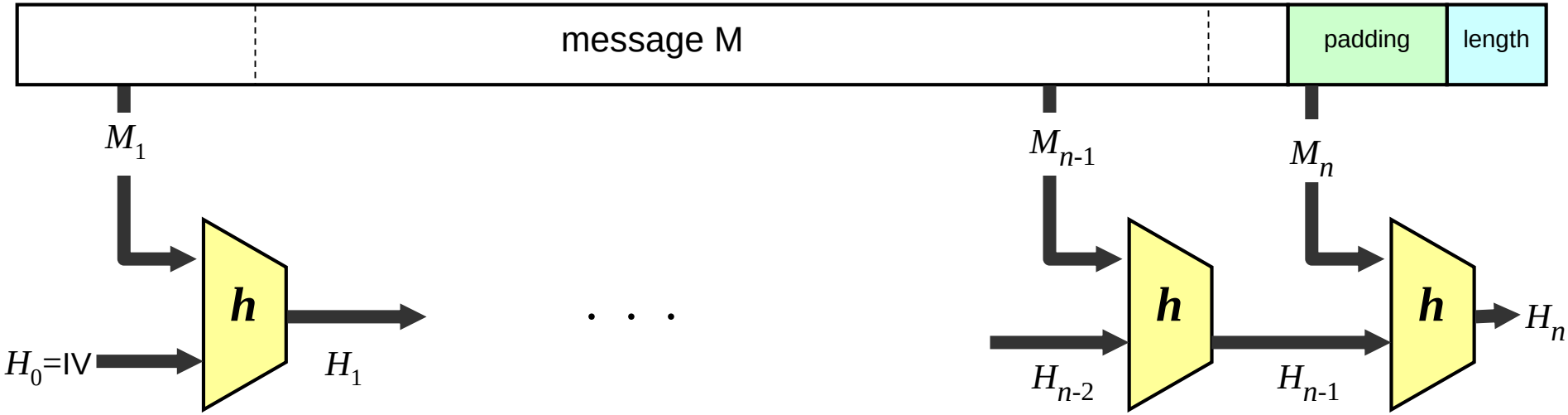
Message Digest

Basic composition



Message Digest

Merkle-Damgård construction:



➔ https://researchgate.net/publication/322094216_Merkle-Damgard_Construction_Method_and_Alternatives_A_Review

Hash Algorithms

MD (*Message Digest* by Ron Rivest)

- MD4 (RFC 1320) , MD5 (RFC 1321) — 128 bit

SHA (*Secure Hash Algorithm* by NSA)

- SHA-1 (by NSA; NIST std. DSS) — 160 bit

Hash Algorithms

MD (*Message Digest* by Ron Rivest)

- MD4 (RFC 1320) , MD5 (RFC 1321) — 128 bit

SHA (*Secure Hash Algorithm* by NSA)

- SHA-1 (by NSA; NIST std. DSS) — 160 bit
- SHA-2 (by NSA): SHA-224, SHA-256, SHA-384, SHA-512

Hash Algorithms

MD

SHA

NIST National Institute of Standards and Technology
Information Technology Laboratory

SEARCH: Search

CONTACT SITE MAP

Computer Security Division
Computer Security Resource Center

CSRC Home | About | Projects / Research | Publications | News & Events

Cryptographic Hash & SHA-3 Standard Development

- Pre-SHA3 Competition (2004-2007)
- SHA-3 Competition (2007-2012)
  - Submission Requirements
  - Round 1
  - Round 2
  - Round 3
- SHA-3 Standardization (2013-2015)
- SHA-3 Derived Functions (2016)
- NIST Policy on Hash Functions
- Hash Forum
- Contacts

CSRC HOME > GROUPS > CT > HASH PROJECT > SHA-3

SHA-3 COMPETITION (2007-2012)

***Research Results on SHA-1 Collisions* (2017)**

NIST announced a public competition in a Federal Register Notice on November 2, 2007 to develop a new cryptographic hash algorithm, called SHA-3, for standardization. The competition was NIST's response to advances made in the cryptanalysis of hash algorithms.

NIST received sixty-four entries from cryptographers around the world by October 31, 2008, and selected fifty-one first-round candidates in December 2008, fourteen second-round candidates in July 2009, and five finalists – BLAKE, Grøstl, JH, Keccak and Skein, in December 2010 to advance to the third and final round of the competition.

Throughout the competition, the cryptographic community has provided an enormous amount of feedback. Most of the comments were sent to NIST and a public hash forum; in addition, many of the cryptanalysis and performance studies were published as papers in major cryptographic conferences or leading cryptographic journals. NIST also hosted a SHA-3 candidate conference in each round to obtain public feedback. Based on the public comments and internal review of the candidates, NIST announced Keccak as the winner of the SHA-3 Cryptographic Hash Algorithm Competition on October 2, 2012, and ended the five-year competition.

Hash Algorithms

MD (*Message Digest* by Ron Rivest)

- MD4 (RFC 1320) , MD5 (RFC 1321) — 128 bit

SHA (*Secure Hash Algorithm* by NSA)

- SHA-1 (by NSA; NIST std. DSS) — 160 bit
- SHA-2 (by NSA): SHA-224, SHA-256, SHA-384, SHA-512

  ➔ https://sha256algorithm.com
- SHA-3 (Keccak, 2012; FIPS 202 std., 2015)

  ➔ http://keccak.noekeon.org

  ➔ https://csrc.nist.gov/CSRC/media/Publications/fips/202/final/documents/fips_202_draft.pdf

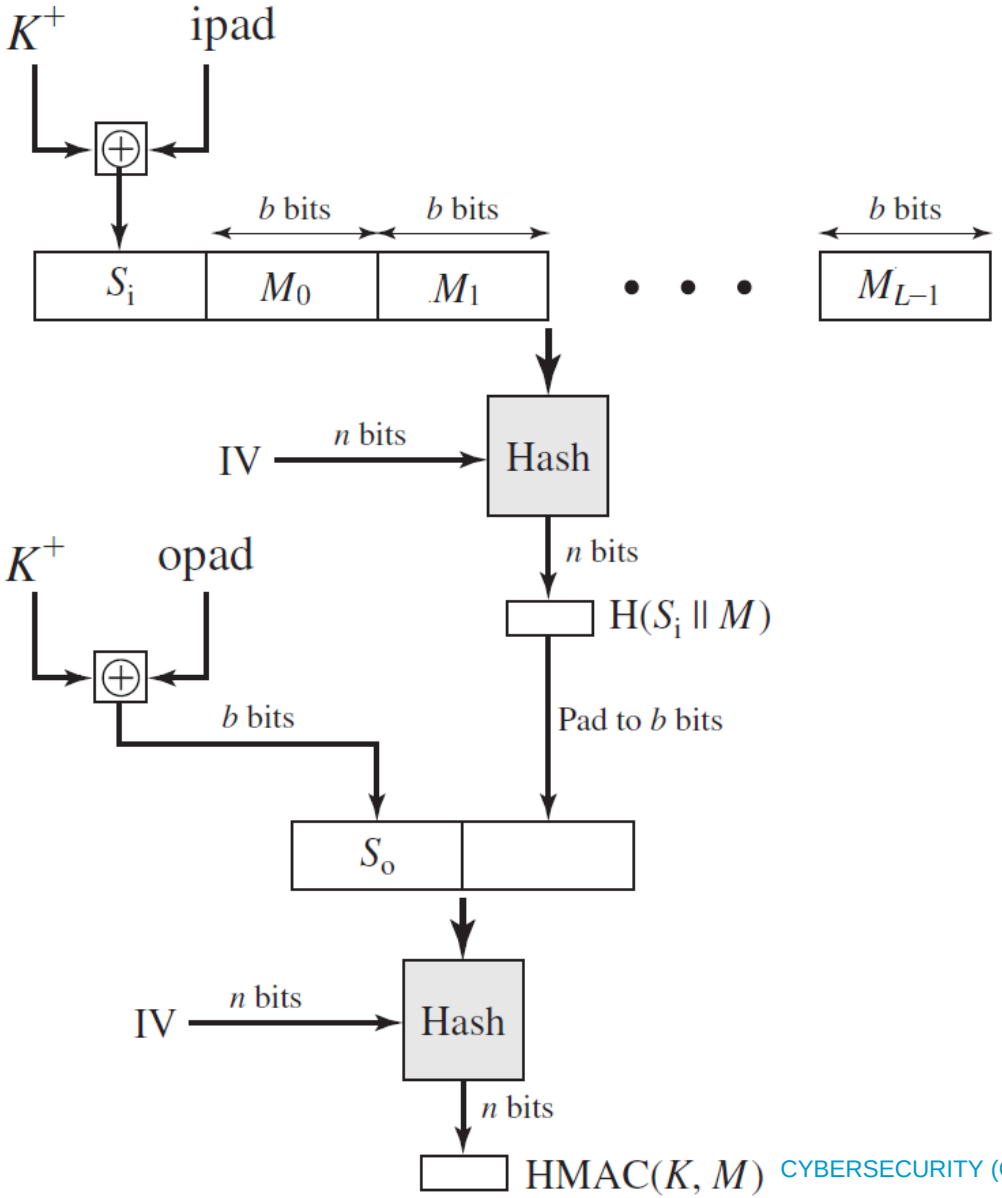
Whirlpool

. . .

Authenticity

HMAC (*Hashing MAC*)

- RFC 2104, FIPS 198
- HMAC-MD5
- HMAC-SHA1
- HMAC-SHA-256



Authenticity

ElGamal

- $HMAC_k[M] = (a, b)$ ::

$$\begin{cases} a = g^k \bmod p \\ b :: M = (xa + kb) \bmod p\text{-}1 \end{cases}$$

DSA (*Digital Signature Algorithm*)

- ElGamal tuned down by NSA (FIPS 186, FIPS 186-4 2003)
- today:
  - ECDSA
  - EdDSA (RFC 8032, Twisted Edwards curves)
  - Ed25519 (Edwards25519 curve)

Authenticity

Poly1305 authenticator, RFC 7539

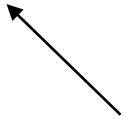
- fast $HMAC_k[M]$
- by Daniel J. Bernstein (Rumba20, Salsa20, ChaCha20)

AES used for MAC

- AES-CMAC (CBC MAC), AES-GMAC

AEAD (Authenticated Encryption with Associated Data)

- AES-CCM (CM + CBC MAC), AES-GCM (G/CM + GMAC)
- AES-Poly1305
- ChaCha20-Poly1305



TLS, SSH, IPsec, 802.1ae (=MACsec), 802.11i, 802.11ad (=WiGig)

Authenticated Encryption

"Nil cryptographiae sine veritate"

Encrypted Cookies Are Not Enough

- Developers often encrypt cookie payloads assuming it cannot be changed
- Encryption **does not provide integrity**! Attackers can modify an encrypted cookie without knowing the key

```
def encryptCookie(payload, key, iv):
    obj = AES.new(key, AES.MODE_CTR, iv)
    str1 = padding(payload)
    ciphertext = obj.encrypt(str1)
    return ciphertext

AuthCookieVal = encryptCookie("Role:Reviewer", "aiBuacoM8", "mee0epJee")
```

Bit-flip to Victory

Cookie payload = "Role:**Reviewer**" provides the cookie value (hex) below

`set-cookie: auth=de6dd89e66232da8a4dac92845;` ← This isn't signed!

Attacker:

By gathering cookies from various roles, looking for patterns and bit-flipping with XOR, a new valid cookie can be crafted without knowing the encryption key

`de6dd89e66232da8a4dac92845 XOR 13011b000b`

`Cookie: auth=de6dd89e66302cb3a4d1` ← Outcome used to set attacker's cookie

Decrypts to "Role:**Admin**"

Authenticated Encryption

Nothing is perfect

"Message Franking via Committing Authenticated Encryption", Grubbs, Lu, Ristenpart, IACR CRYPTO17, https://eprint.iacr.org/2017/664

"Fault Attacks on Nonce-based Authenticated Encryption (…)", Dobraunig, Mangard, Mendel, Primas, SAC 2018, https://eprint.iacr.org/2018/852.pdf

"Fast Message Franking: From Invisible Salamanders to Encryptment", Dodis, Grubbs, Ristenpart, Woodage, IACR CRYPTO18, https://eprint.iacr.org/2019/016

"Fault attacks on authenticated encryption modes for GIFT", Liu, Guan, Hu, 2001, https://ietresearch.onlinelibrary.wiley.com/doi/full/10.1049/ise2.12041

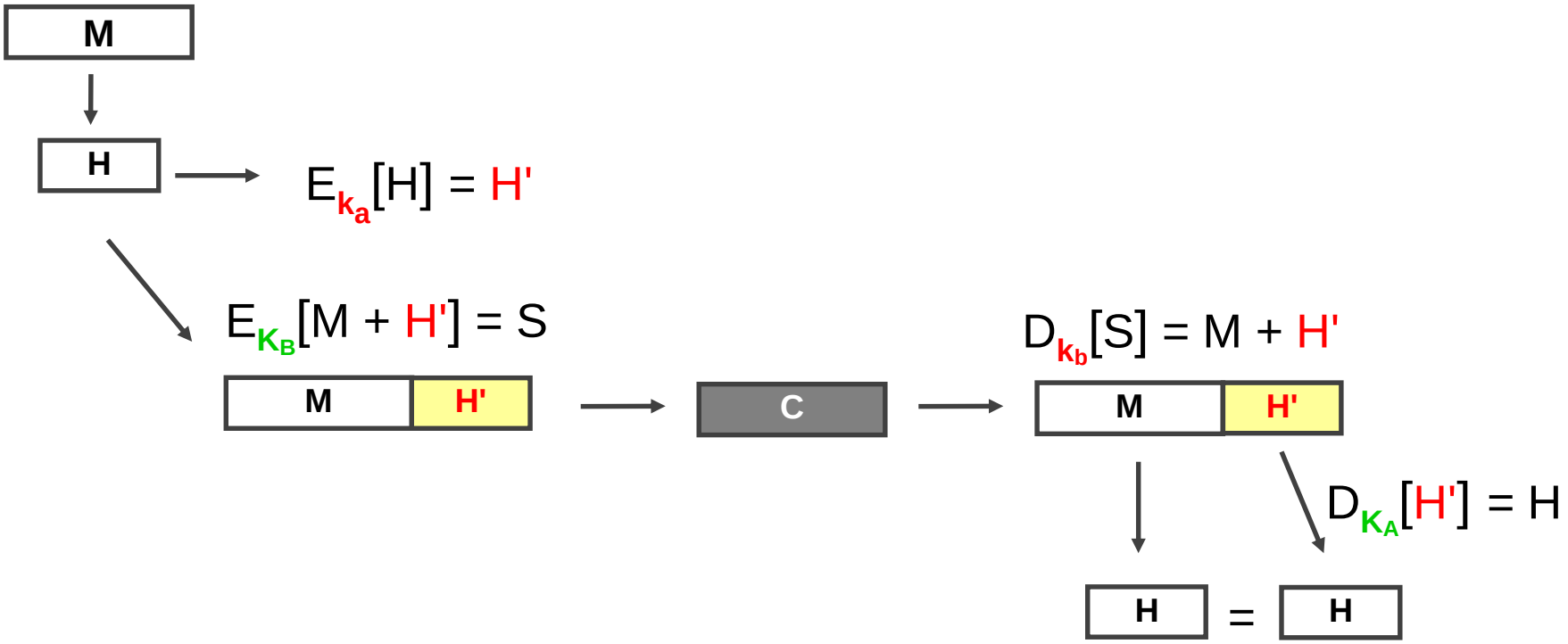
. . .

All in one

k_a
K_A

confidentiality
integrity
authenticity
nonrepudiation

k_b
K_B



ATTACKS

Collisions

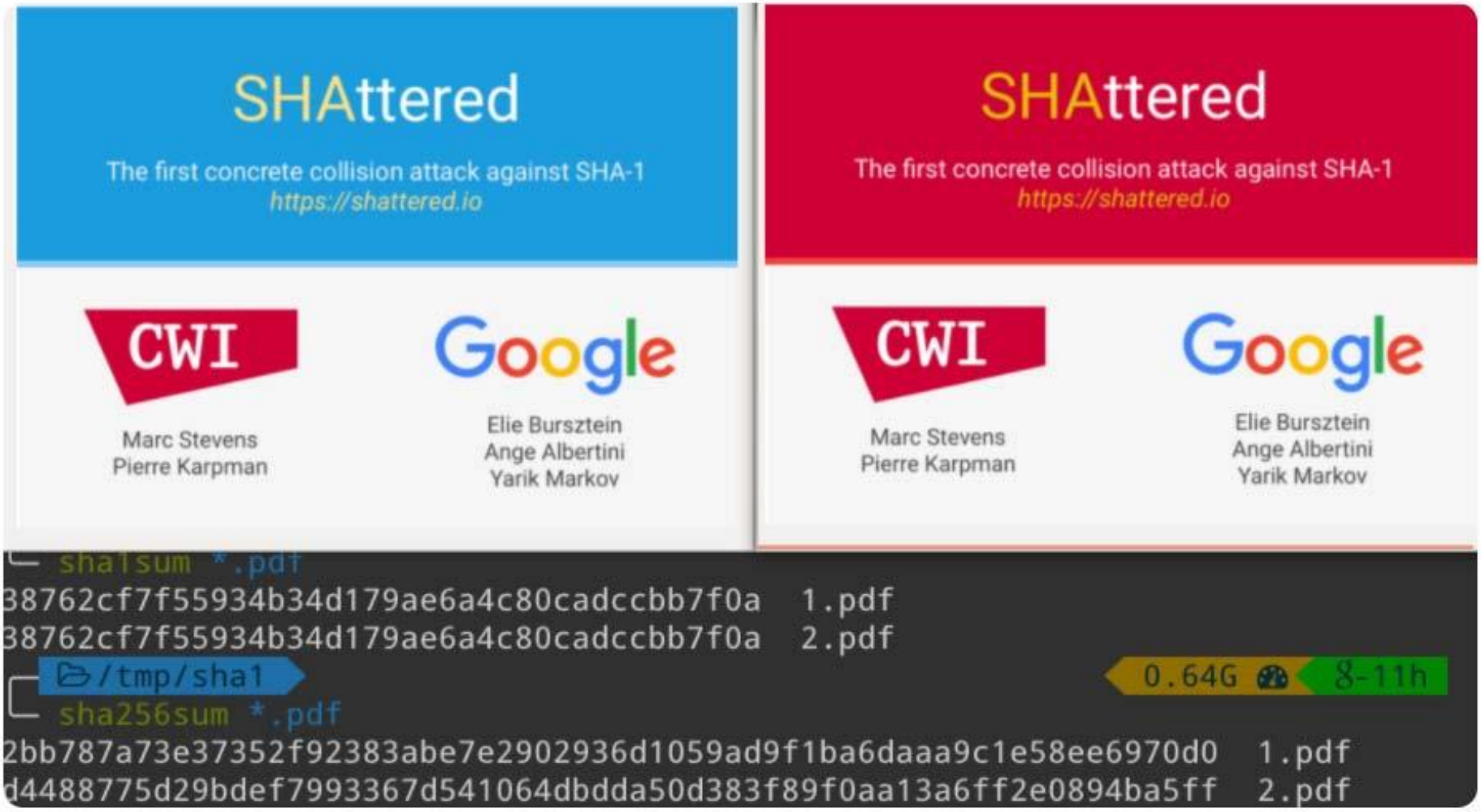
Birthday Attack

- theoretically: 128-bit digest has a collision resistance of $2^{64} = \frac{128}{2}$
- ☹ MD5
- ☹ SHA-1
- ☹ … → https://eprint.iacr.org/2004/199.pdf

nonetheless

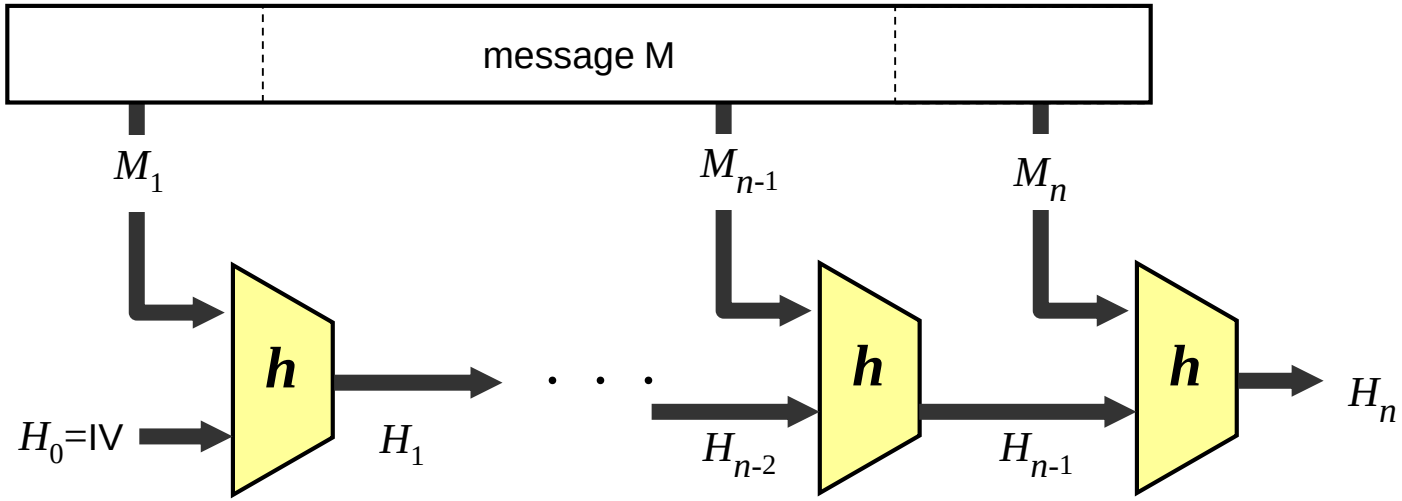
- preimage resistance and 2nd preimage resistance are more important

Collisions


SHAttered
The first concrete collision attack against SHA-1
https://shattered.io
CWI
Marc Stevens
Pierre Karpman
Google
Elie Bursztein
Ange Albertini
Yarik Markov
SHAttered
The first concrete collision attack against SHA-1
https://shattered.io
CWI
Marc Stevens
Pierre Karpman
Google
Elie Bursztein
Ange Albertini
Yarik Markov
sha1sum *.pdf
38762cf7f55934b34d179ae6a4c80cadccbb7f0a 1.pdf
38762cf7f55934b34d179ae6a4c80cadccbb7f0a 2.pdf
/tmp/sha1
0.64G
8-11h
sha256sum *.pdf
2bb787a73e37352f92383abe7e2902936d1059ad9f1ba6daaa9c1e58ee6970d0 1.pdf
d4488775d29bdef7993367d541064dbdda50d383f89f0aa13a6ff2e0894ba5ff 2.pdf

Message Digest

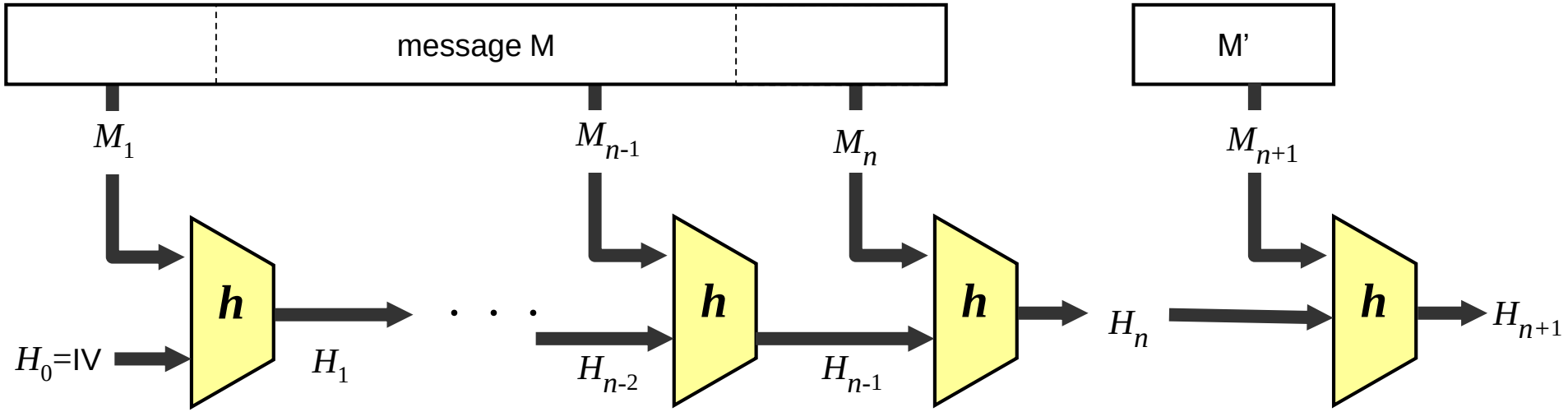
Merkle-Damgård construction



$$H_n = \boldsymbol{h}(M_n, H_{n-1})$$

Length extension attack

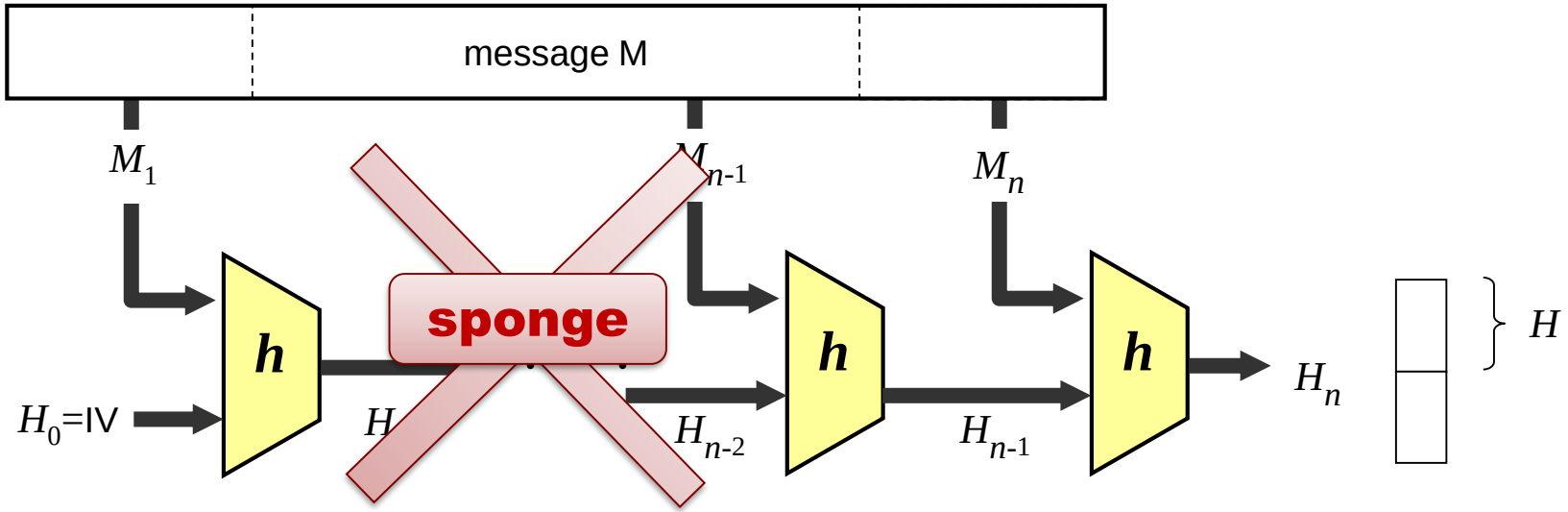
Merkle-Damgård construction



$$H_{n+1} = \boldsymbol{h}(M_{n+1}, H_n)$$

Length extension attack

Solution (→ SHA-3)



$$H_{n+1} = \boldsymbol{h}(M_{n+1}, H_n = ?)$$

Randomness

IV = nonce !

- always: unique
- sometimes: uniform distribution (e.g. for ECDSA, but not EdDSA)

→ "Nonce-Disrespecting Adversaries: Practical Forgery Attacks on GCM in TLS"

Randomness

Sample use of RNG:

```
RNGCryptoServiceProvider random_generator;
byte[] nonce = new Byte[32];   // 32B * 8b = 256-bit nonce
random_generator = new RNGCryptoServiceProvider();
random_generator.GetNonZeroBytes(nonce);
```

Randomness

RNG of a kind:

```
int getRandomNumber()
{
    return 4;  // chosen by fair dice roll.
               // guaranteed to be random.
}
```

Source: http://www.xkcd.com

Randomness

New solution required:

- nonce misuse resistant algorithms (→ Lightweight Crypto for IoT)
- Synthetic IV (SIV):

  (pre-)nonce + plaintext M_1 = IV

Key problems

Fundamental problems:

- shared secret (symmetric) key?
- safe key storage?
- secure transport (distribution)?
- key change and renewal?
- key revocation?
- key recovery?

KEY DISTRIBUTION

Diffie-Hellman

Whitfield Diffie, Martin Hellman
(Stanford Univ.)

k_a

$K_A = \alpha^{k_a} \bmod p$

α , p

k_b

$K_B = \alpha^{k_b} \bmod p$

$\Phi_A = K_B^{k_a} = (\alpha^{k_b})^{k_a} \bmod p$

$\Phi_B = K_A^{k_b} = (\alpha^{k_a})^{k_b} \bmod p$

$\Phi_A = \alpha^{(k_a \cdot k_b)} \bmod p = \Phi_B$

Φ session key Φ

Diffie-Hellman

Man-in-the-Middle attack!

- consider altering $K_A = \alpha^{k_a} = 1$, and $K_B = \alpha^{k_b} = 1$

Solution:

- DH + authentication = MAC of K_A and K_B

Group Key Exchange (GKE)

- Authenticated Group Key Exchange (AGKE)
- Stateful Group Key Exchange (stGKE)
- Stateful Authenticated Group Key Exchange (stAGKE)

Shamir's Secret Sharing

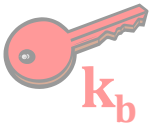
- a scheme for distributing a single secret among a group of individuals
- breaks the original secret into n parts, which can then be distributed
- k parts are needed to recalculate the original secret value

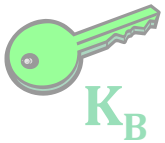
```
~ $ echo 'this is a secret' | ssss-split -n 5 -t 2
Generating shares using a (2,5) scheme with dynamic security level.
Enter the secret, at most 128 ASCII characters: Using a 128 bit security level.
1-4054162f42f328c2ecbff990e9e1996f
2-93285deac4d6406cde841b05b350f61f
3-22039b5646ca98093092ba897ac02cb0
4-35d0ca61c89c9130baf3de2f06322866
5-84fb0cdd4a80495554e57fa3cfa2f2c9
~ $ ssss-combine -t 2
Enter 2 shares separated by newlines:
Share [1/2]: 5-84fb0cdd4a80495554e57fa3cfa2f2c9
Share [2/2]: 4-35d0ca61c89c9130baf3de2f06322866
Resulting secret: this is a secret
```

Public key distribution

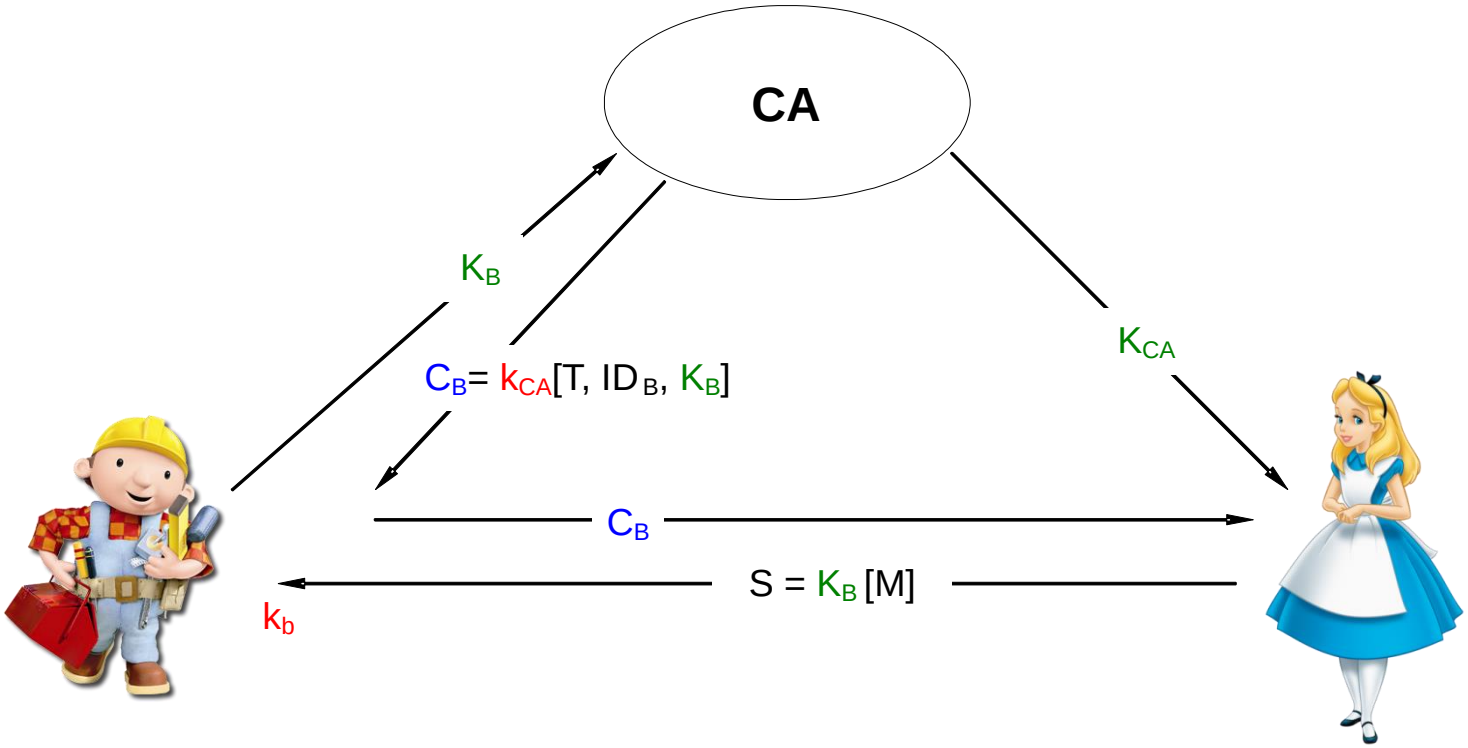
Where to get K_B from?

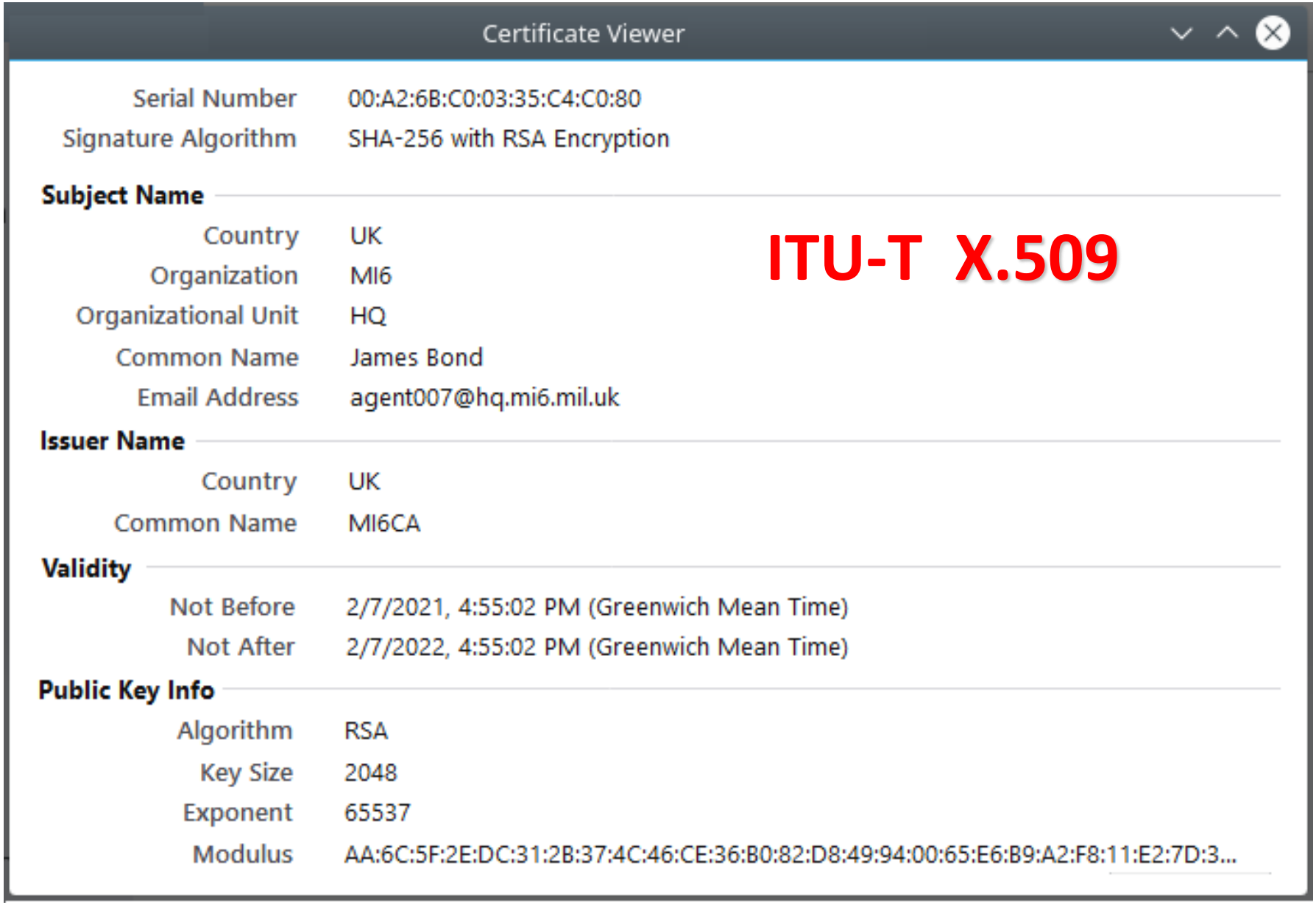
1. From Bob himself, in person?
2. From a (public) repository?





Public key certificates




Certificate Viewer
Serial Number 00:A2:6B:C0:03:35:C4:C0:80
Signature Algorithm SHA-256 with RSA Encryption
Subject Name
Country UK
Organization MI6
Organizational Unit HQ
Common Name James Bond
Email Address agent007@hq.mi6.mil.uk
Issuer Name
Country UK
Common Name MI6CA
Validity
Not Before 2/7/2021, 4:55:02 PM (Greenwich Mean Time)
Not After 2/7/2022, 4:55:02 PM (Greenwich Mean Time)
Public Key Info
Algorithm RSA
Key Size 2048
Exponent 65537
Modulus AA:6C:5F:2E:DC:31:2B:37:4C:46:CE:36:B0:82:D8:49:94:00:65:E6:B9:A2:F8:11:E2:7D:3...
ITU-T X.509

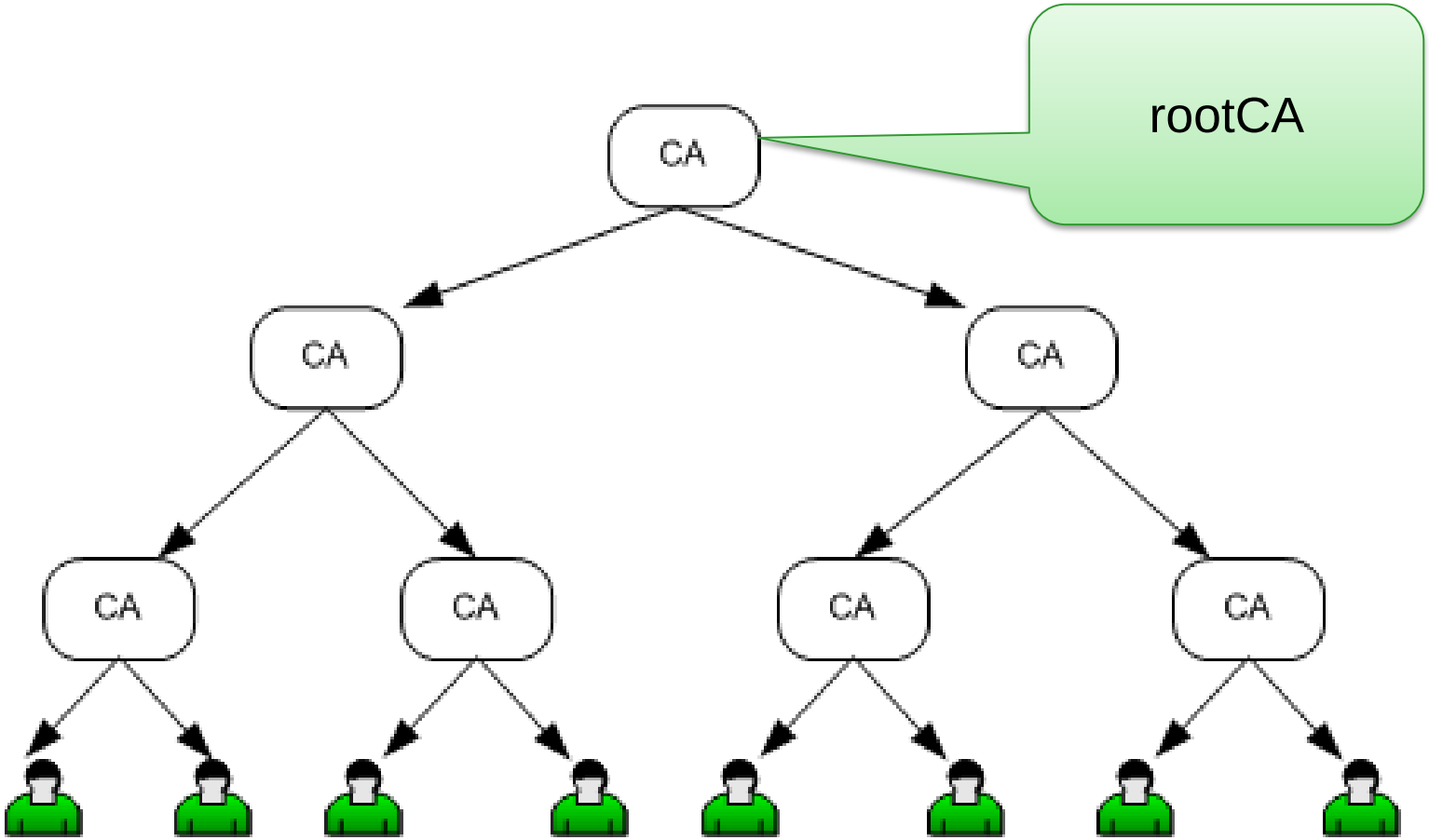
HOMEWORK

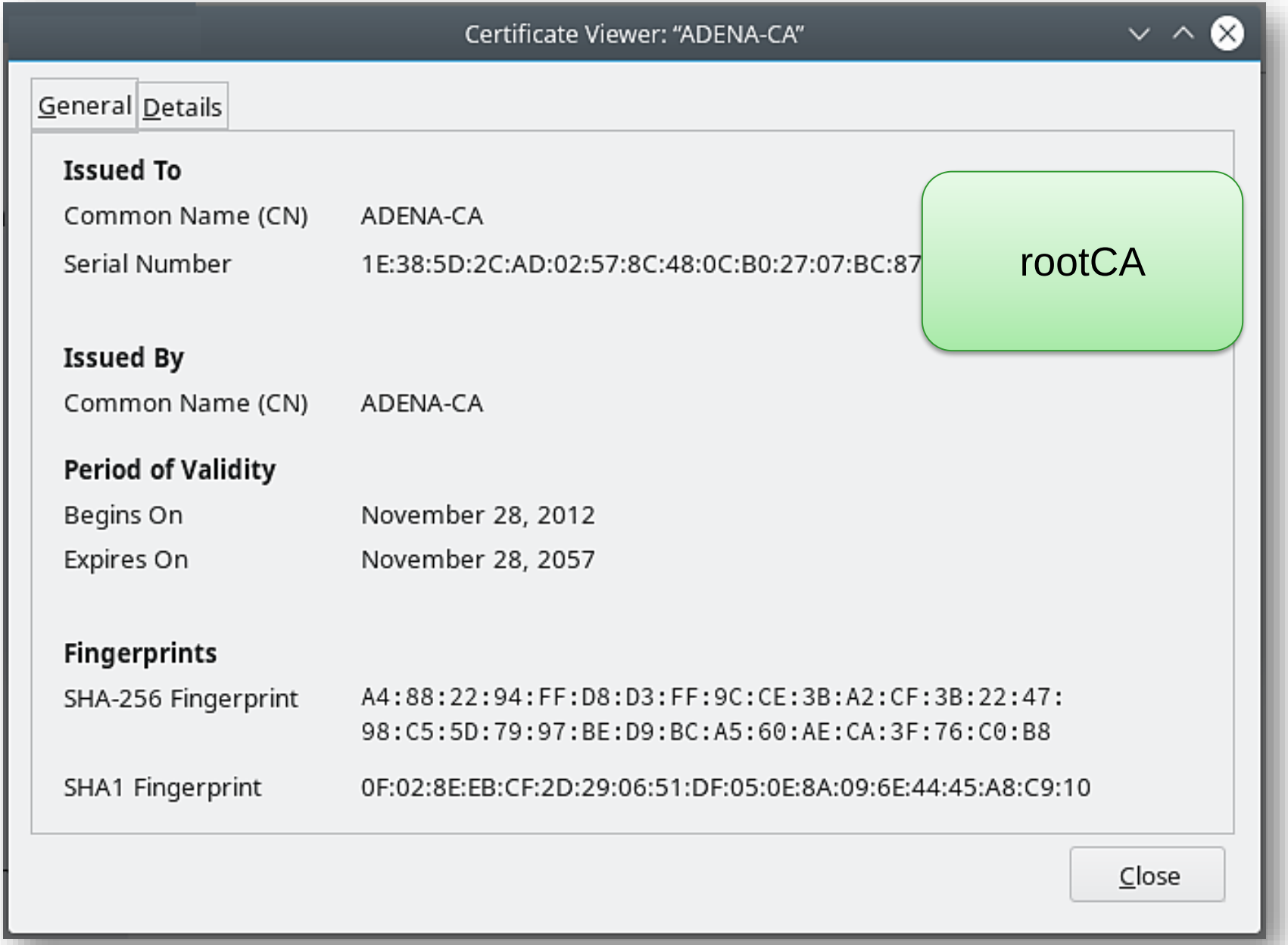
=

Half Of My Energy Wasted On Random Knowledge

→ **IEEE 1609.2 standard**

Public Key Infrastructure




Certificate Viewer: "ADENA-CA"
General
Details
Issued To
Common Name (CN) ADENA-CA
Serial Number 1E:38:5D:2C:AD:02:57:8C:48:0C:B0:27:07:BC:87
rootCA
Issued By
Common Name (CN) ADENA-CA
Period of Validity
Begins On November 28, 2012
Expires On November 28, 2057
Fingerprints
SHA-256 Fingerprint A4:88:22:94:FF:D8:D3:FF:9C:CE:3B:A2:CF:3B:22:47:
98:C5:5D:79:97:BE:D9:BC:A5:60:AE:CA:3F:76:C0:B8
SHA1 Fingerprint 0F:02:8E:EB:CF:2D:29:06:51:DF:05:0E:8A:09:6E:44:45:A8:C9:10
Close

Public key certificates



Domain Validation (DV)	**Automated**	**Fast**	**Free/Cheap**
Organisation Validation (OV)	Manual	Slow	Expensive
Extended Validation (EV)	Manual	Slow	Expensive

Public key certificates

Web of Trust

https://en.wikipedia.org/wiki/Key_signing_party

Web of Trust

https://xkcd.com

Recap

Key distribution:

- symmetric keys
  - — asymmetric crypto to the rescue → D-H
- asymmetric keys
  - — authenticity of the public key → certificates & PKI
    → Web of Trust

More standards

PKCS (*Public Key Cryptography Standard*)

PKCS#1 = RSA

PKCS#2 = Diffie-Hellman

PKCS#5 = Password-based Encryption Standard (PBKDF2 → RFC 2898)

PKCS#7 = digital signature format (→ RFC 2315)

PKCS#10 = certificate request format

PKCS#11 = cryptographic hardware API (→ TPM, HSM)

PKCS#12 = public key (certificate) and private key storage format

CRYPTO APPLICATIONS

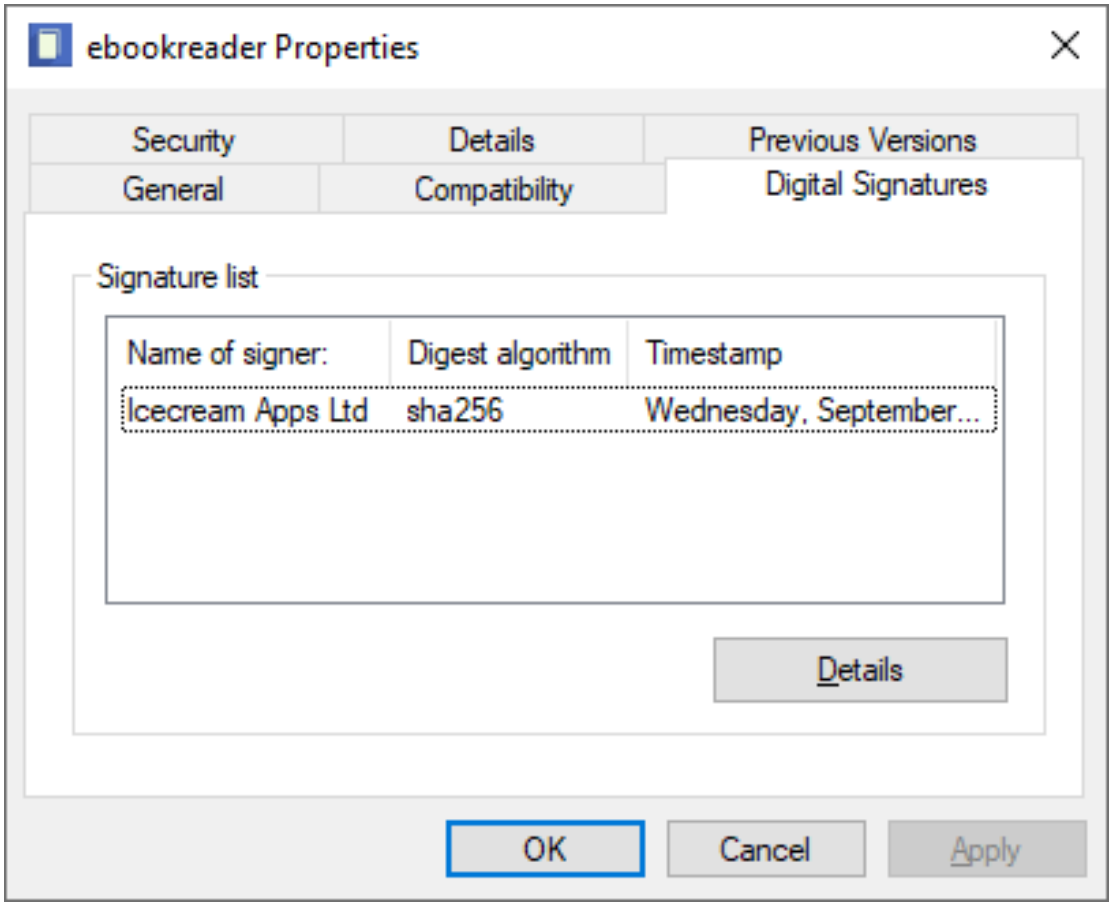
Software Code Integrity

Windows Authenticode:

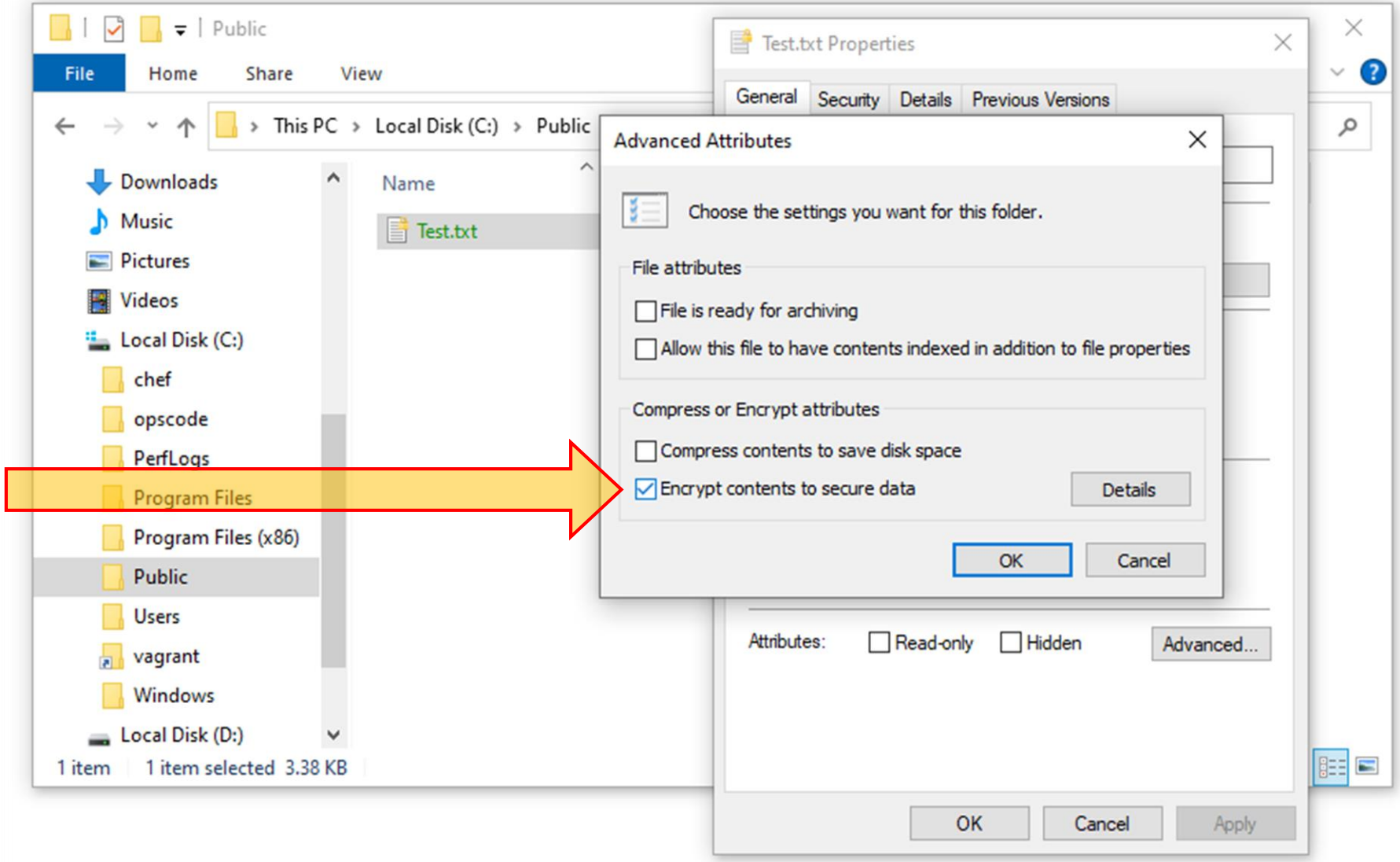
+

Software Code Integrity:

- user-mode (UMCI) = PE execs
- kernel-mode (KMCI) = drivers
- Windows 10 DeviceGuard = Hypervisor Enforced Code Integrity (HVCI)

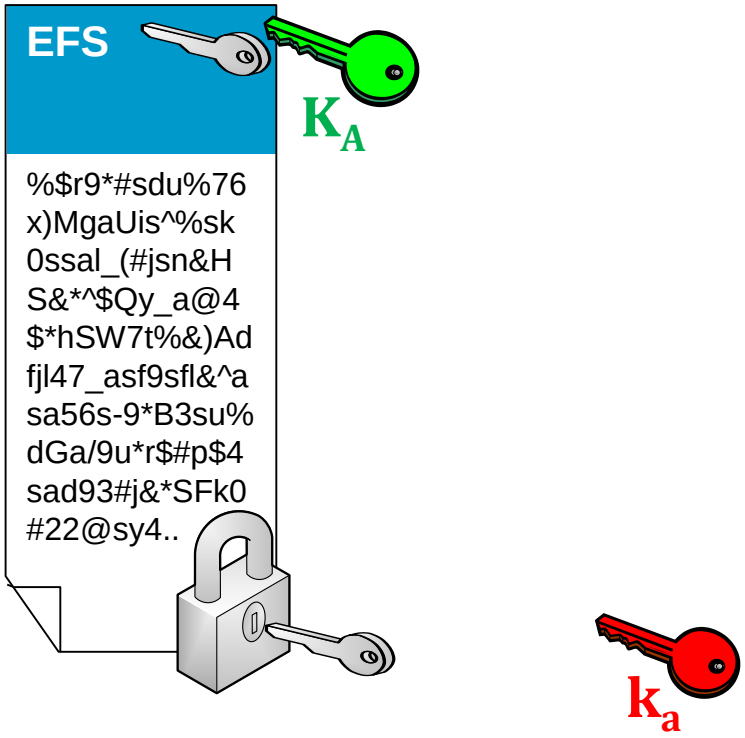


Windows EFS


Public
File
Home
Share
View
This PC
Local Disk (C:)
Public
Downloads
Music
Pictures
Videos
Local Disk (C:)
chef
opscode
PerfLogs
Program Files
Program Files (x86)
Public
Users
vagrant
Windows
Local Disk (D:)
Name
Test.txt
1 item
1 item selected 3.38 KB
Test.txt Properties
General
Security
Details
Previous Versions
Attributes:
Read-only
Hidden
Advanced...
OK
Cancel
Apply
Advanced Attributes
Choose the settings you want for this folder.
File attributes
File is ready for archiving
Allow this file to have contents indexed in addition to file properties
Compress or Encrypt attributes
Compress contents to save disk space
Encrypt contents to secure data
Details
OK
Cancel

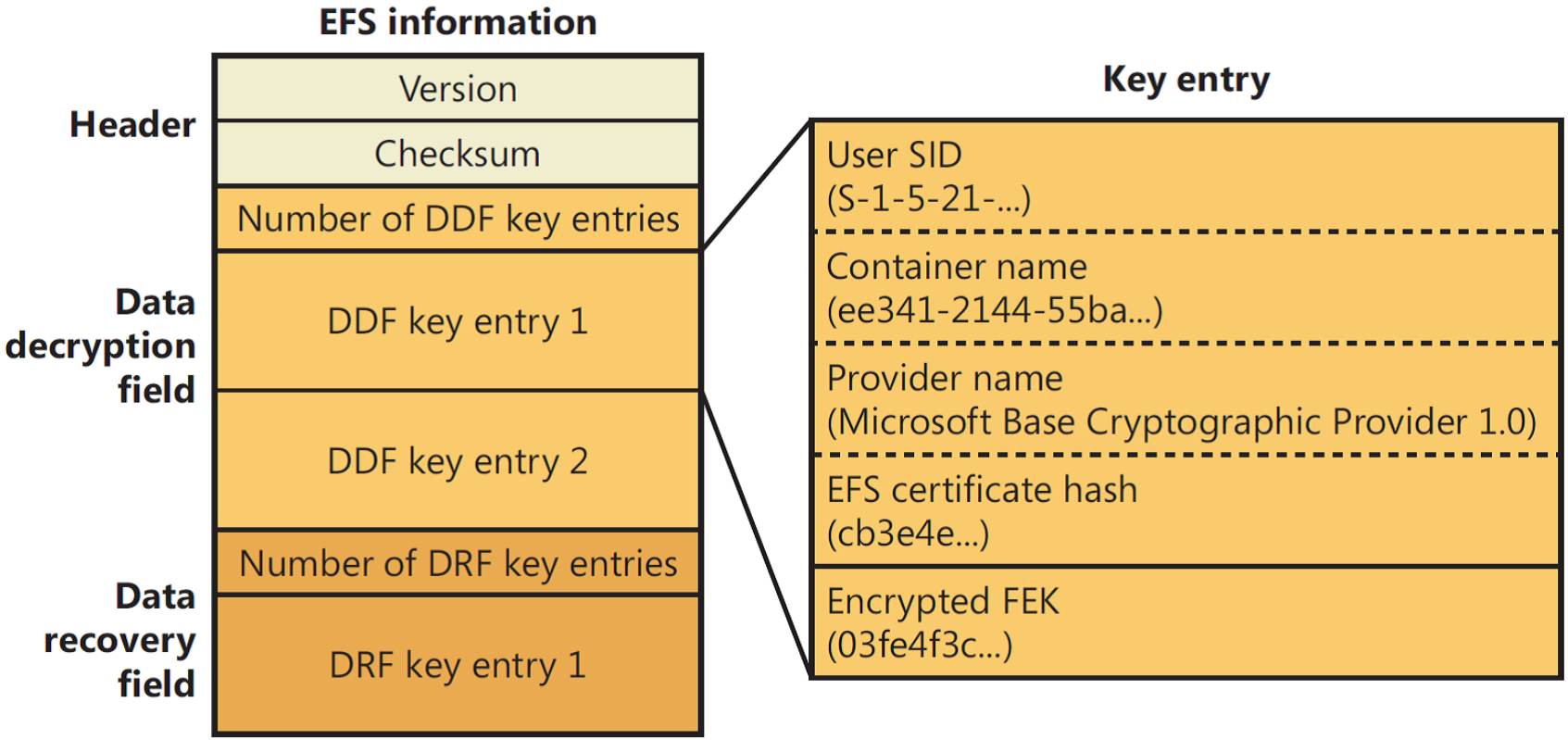
Windows EFS

FEK = File Encryption Key



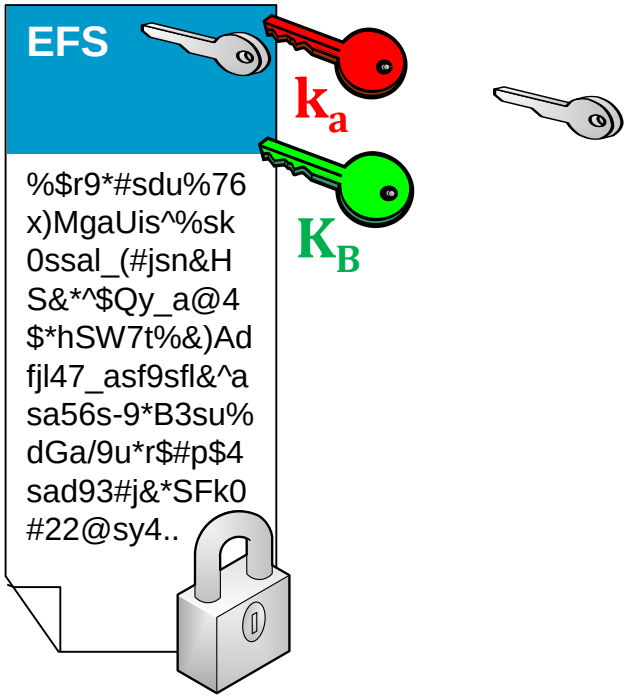
Windows EFS

FEK = File Encryption Key

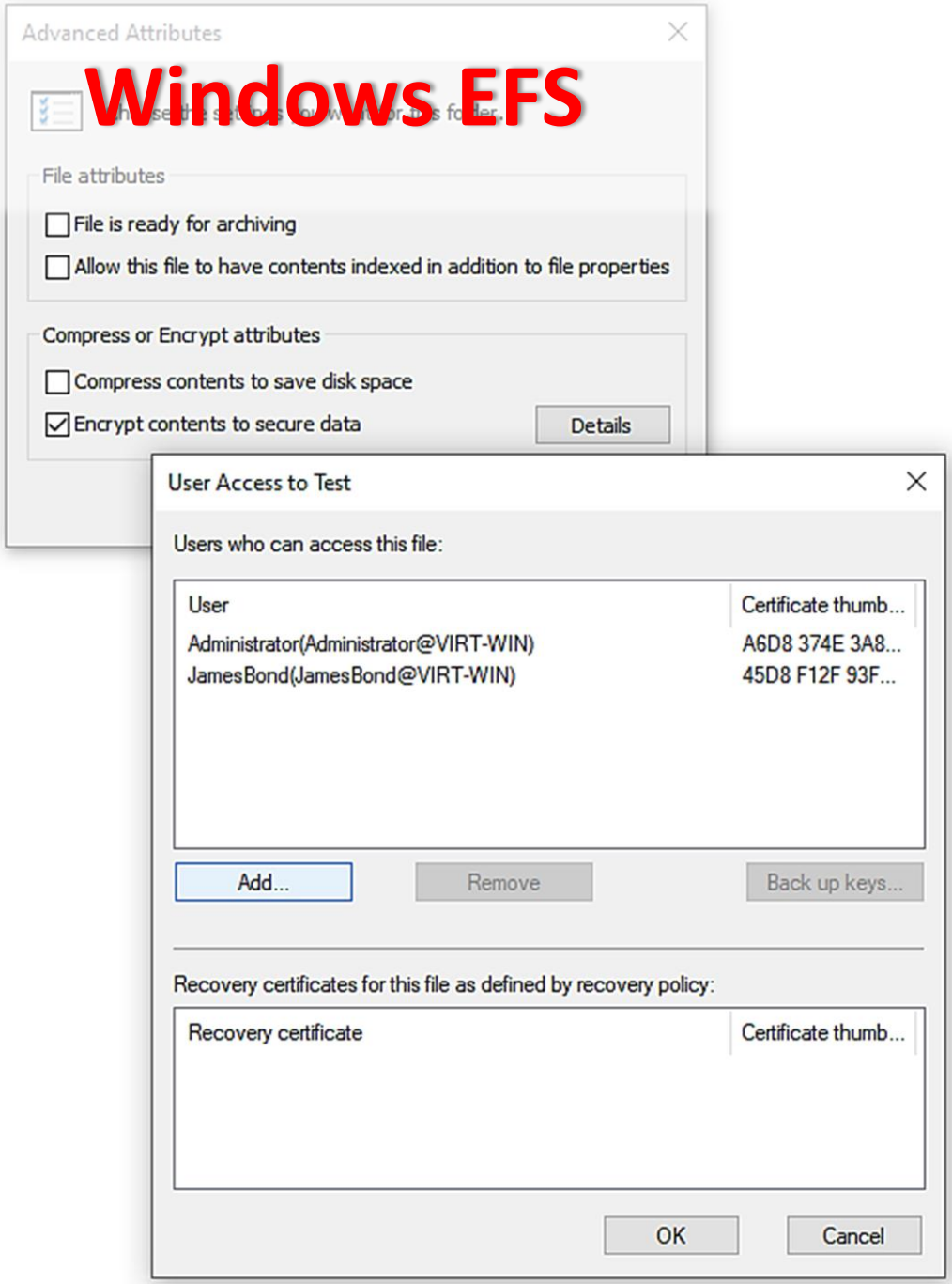


Windows EFS

FEK = File Encryption Key

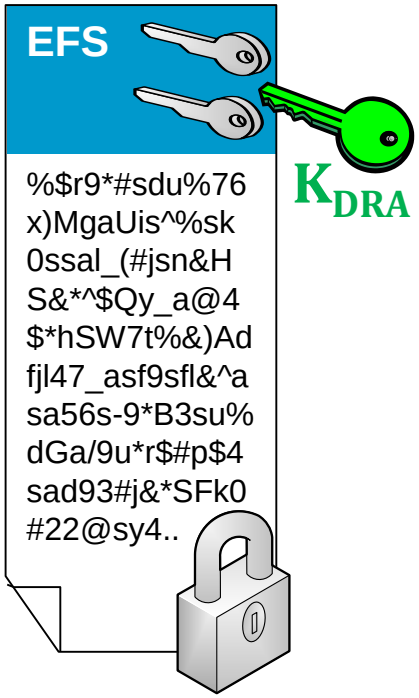


Windows EFS

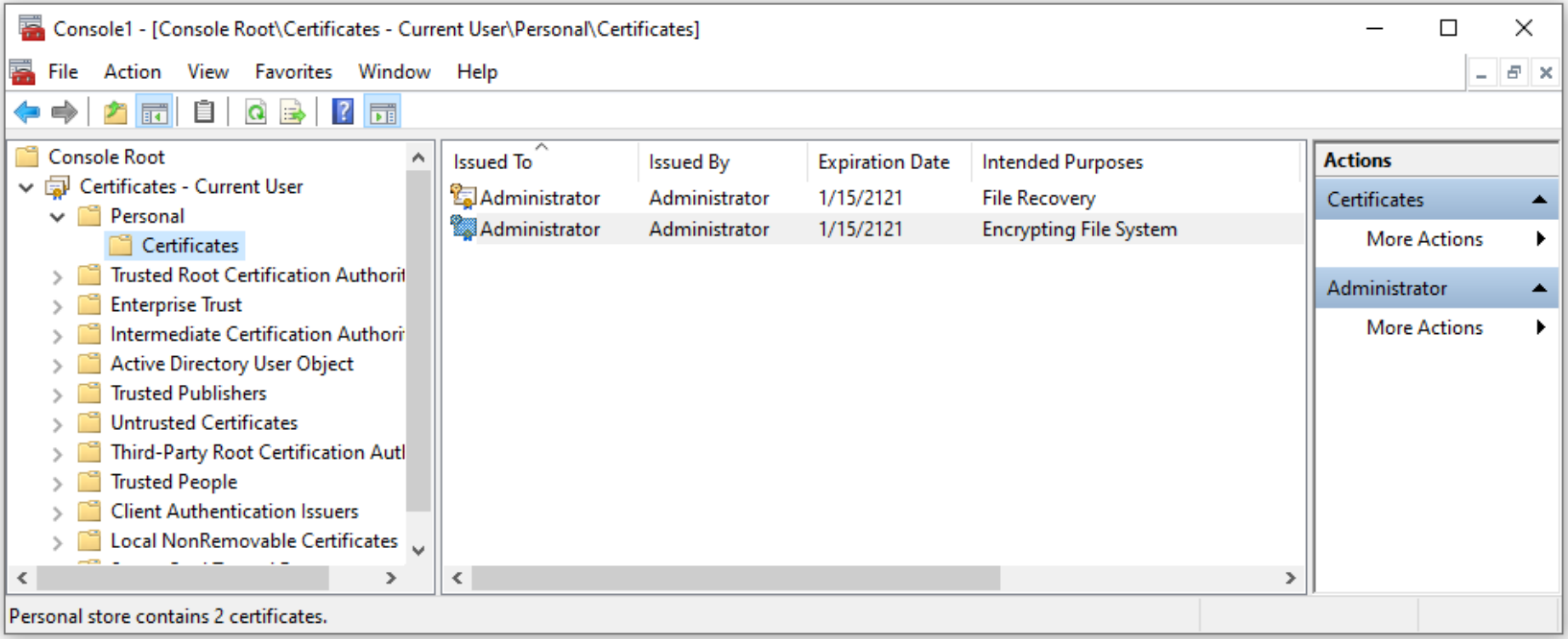

Advanced Attributes
File attributes
File is ready for archiving
Allow this file to have contents indexed in addition to file properties
Compress or Encrypt attributes
Compress contents to save disk space
Encrypt contents to secure data
Details
User Access to Test
Users who can access this file:
User
Certificate thumb...
Administrator(Administrator@VIRT-WIN)
A6D8 374E 3A8...
JamesBond(JamesBond@VIRT-WIN)
45D8 F12F 93F...
Add...
Remove
Back up keys...
Recovery certificates for this file as defined by recovery policy:
Recovery certificate
Certificate thumb...
OK
Cancel

Windows EFS

FEK = File Encryption Key

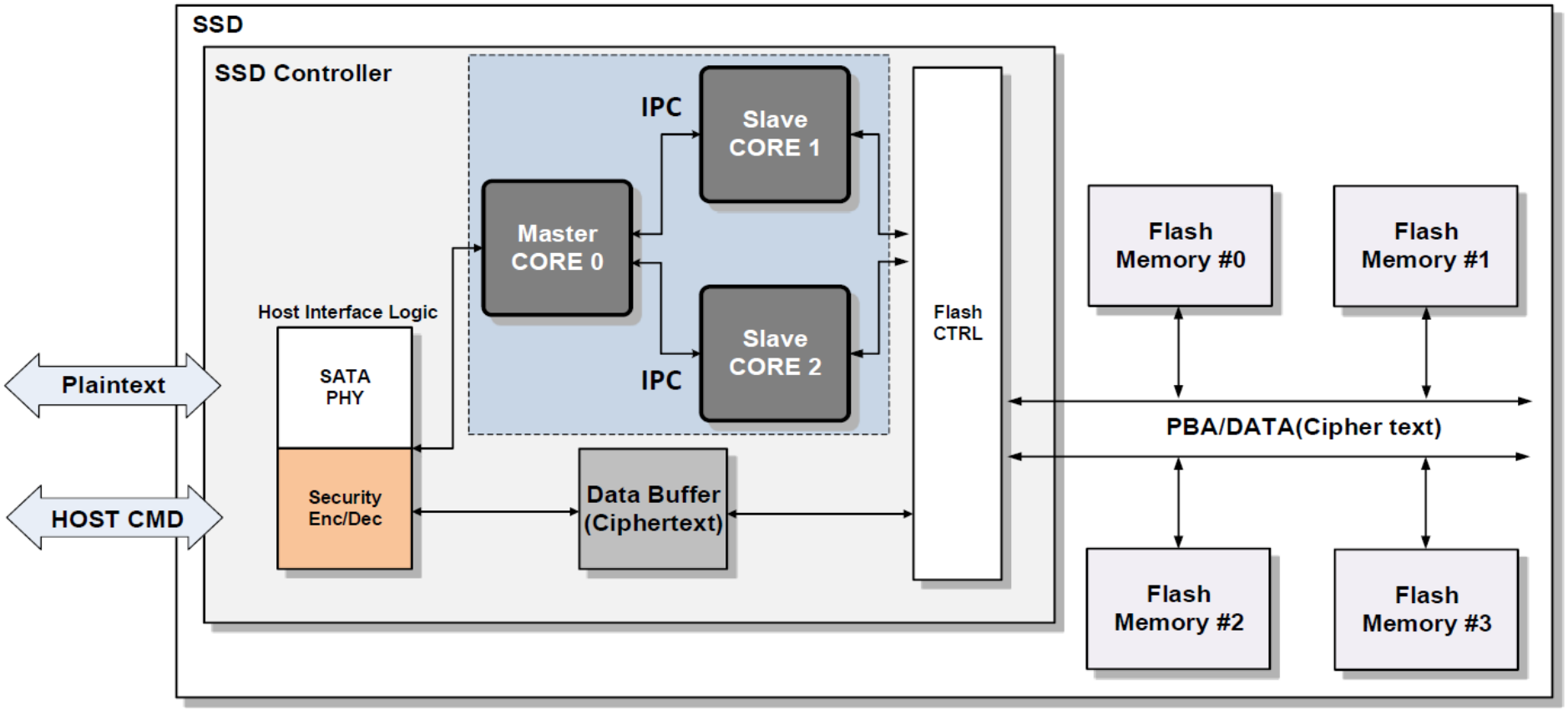


Windows EFS

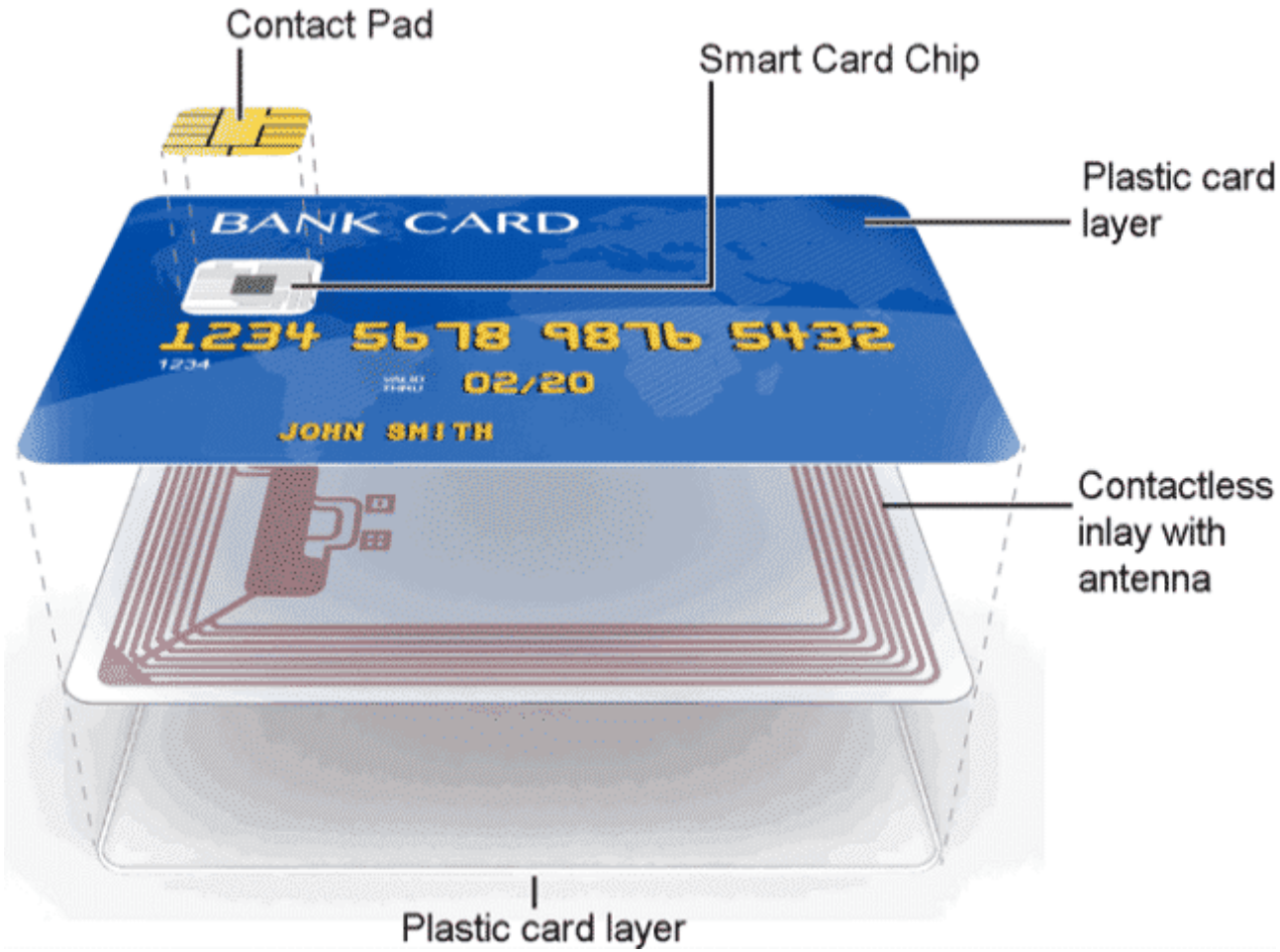

Console1 - [Console Root\Certificates - Current User\Personal\Certificates]
File Action View Favorites Window Help
Console Root
Certificates - Current User
Personal
Certificates
Trusted Root Certification Authorit
Enterprise Trust
Intermediate Certification Authori
Active Directory User Object
Trusted Publishers
Untrusted Certificates
Third-Party Root Certification Autl
Trusted People
Client Authentication Issuers
Local NonRemovable Certificates
Issued To
Issued By
Expiration Date
Intended Purposes
Administrator Administrator 1/15/2121 File Recovery
Administrator Administrator 1/15/2121 Encrypting File System
Actions
Certificates
More Actions
Administrator
More Actions
Personal store contains 2 certificates.

SSD Controllers

AES-XTS

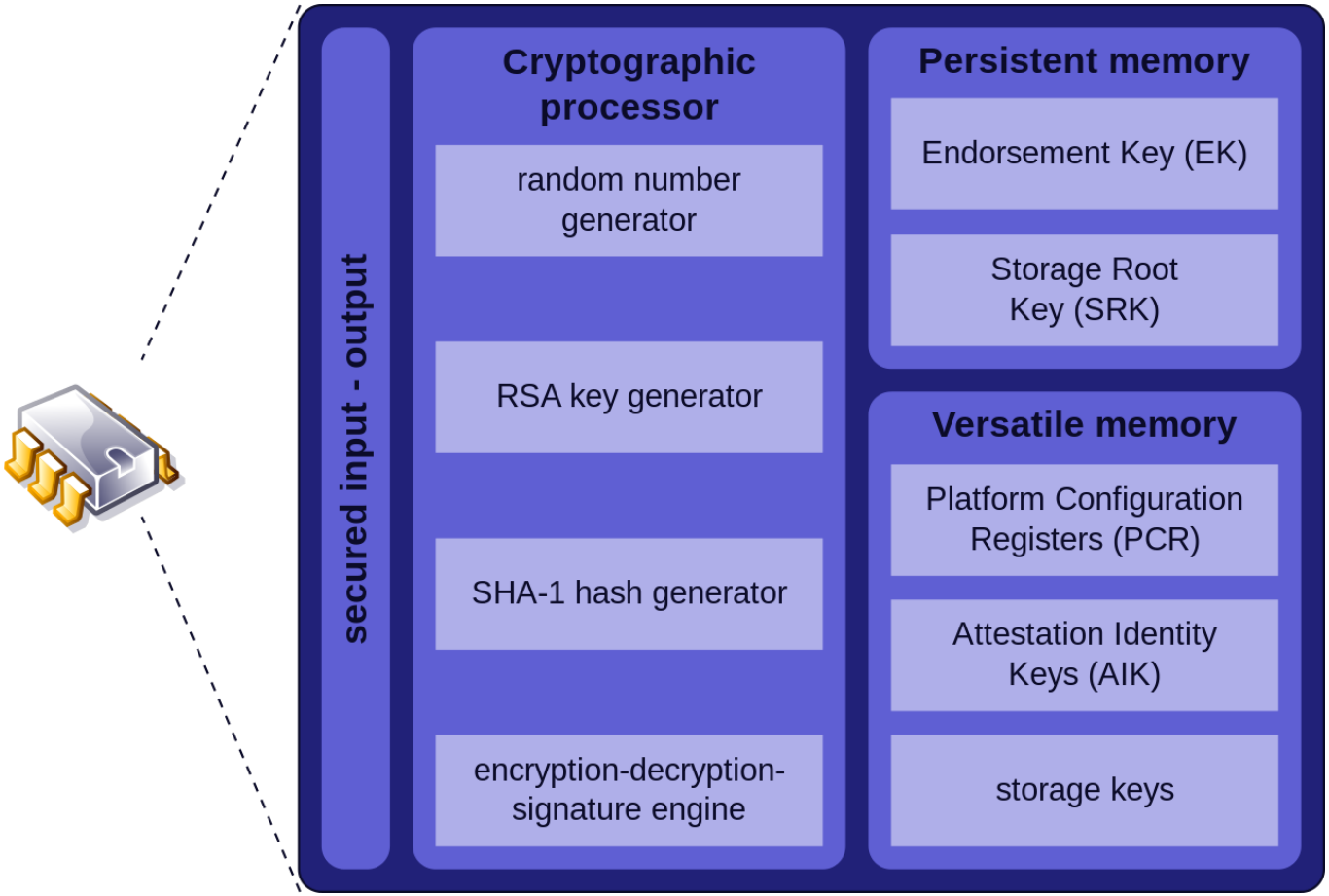


Smart Cards

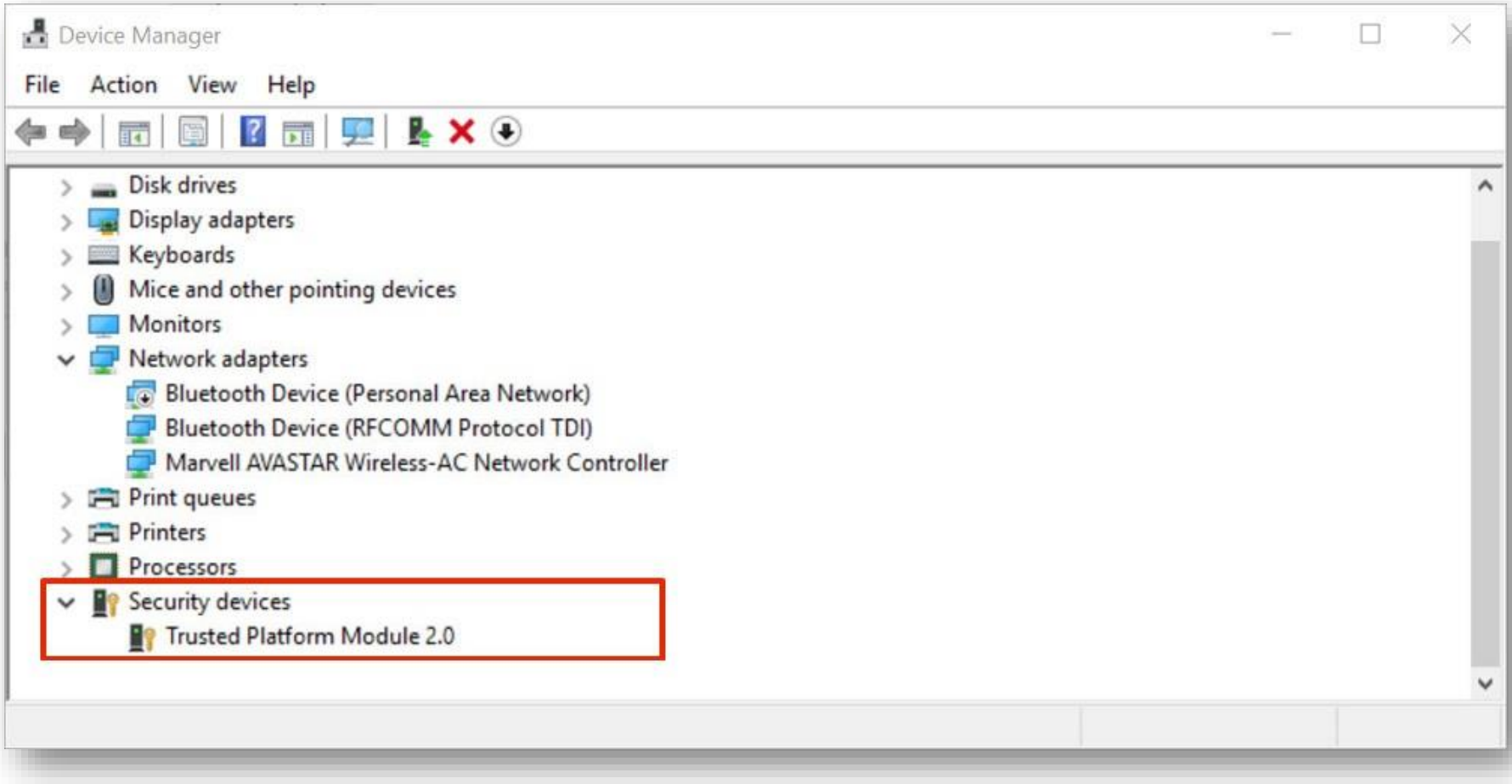

Contact Pad
Smart Card Chip
Plastic card layer
BANK CARD
1234 5678 9876 5432
1234
02/20
JOHN SMITH
Contactless inlay with antenna
Plastic card layer

Trusted Platform Module (TPM)

Trusted Platform Module (TPM)



Trusted Platform Module (TPM)



Trusted Platform Module (TPM)

→ https://trustedcomputinggroup.org/membership/certification/tpm-certified-products/

Trusted Platform Module (TPM)

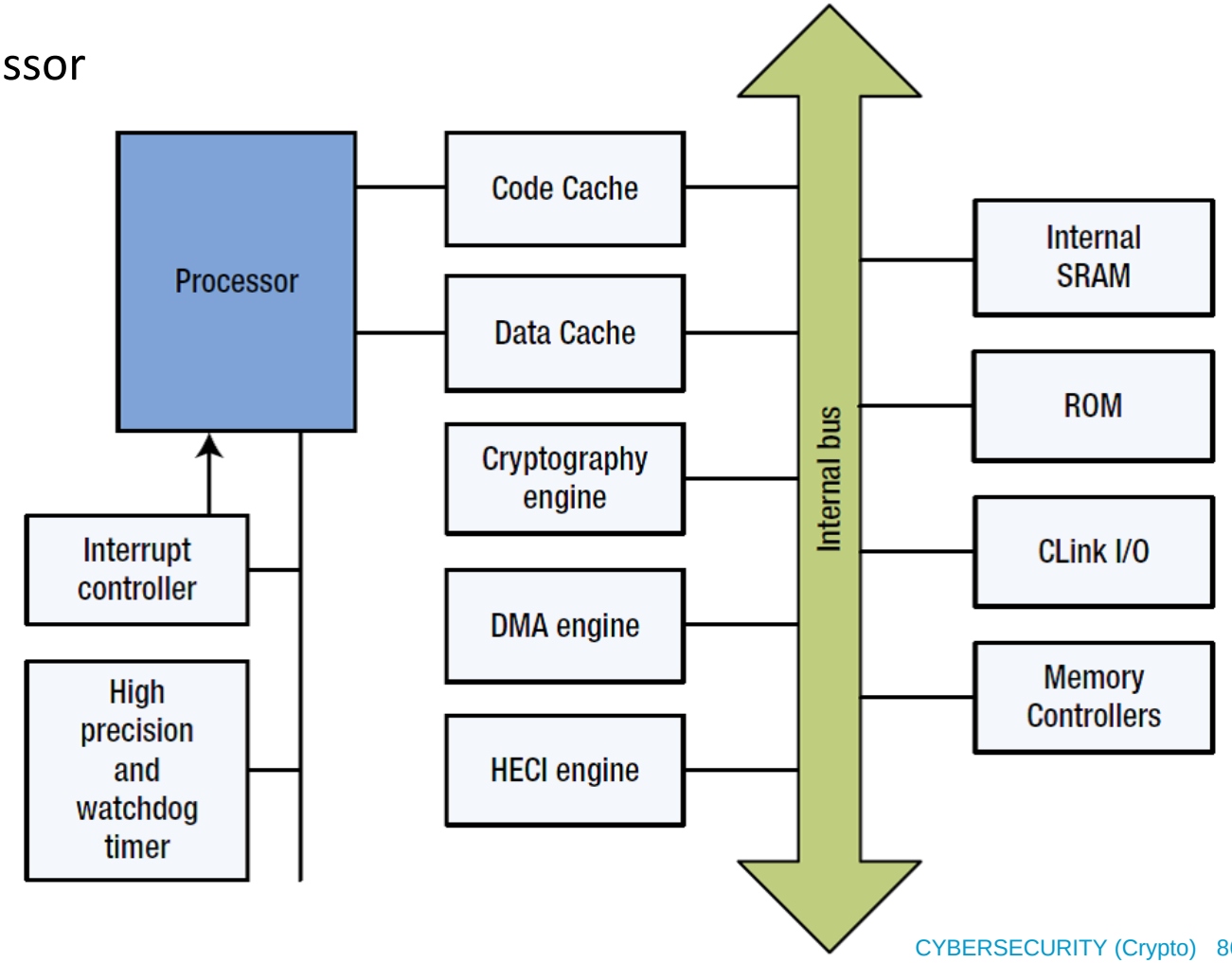
Platform configuration registers (PCRs)

```
PCR-00: A8 5A 84 B7 38 FC ... # BIOS
PCR-01: 11 40 C1 7D 0D 25 ... # BIOS Configuration
PCR-02: A3 82 9A 64 61 85 ... # Option ROM
PCR-03: B2 A8 3B 0E BF 2F ... # Option ROM Configuration
PCR-04: 78 93 CF 58 0E E1 ... # MBR
PCR-05: 72 A7 A9 6C 96 39 ... # MBR Configuration
```

Hardware

Platform Embedded Security (Intel TXT, AMD PSP, …)

- chipset coprocessor
- TPM
- BootGuard
- crypto engine



Hardware Secure Module (HSM)

Hardware Secure Module (HSM)

Cloud HMS

Provider	Dedicated HSM option	Key management service
Amazon Web Services	CloudHSM	Amazon KMS
Microsoft Azure	---	Key Vault (software keys)
Google Compute Platform	---	Cloud KMS
IBM Cloud	Cloud HSM	Key Protect

New trends

Threshold Signature Scheme (TSS)

- new cryptographic primitive for distributed key generation and signing: https://academy.binance.com/en/articles/threshold-signatures-explained
- best-suited to use in blockchain

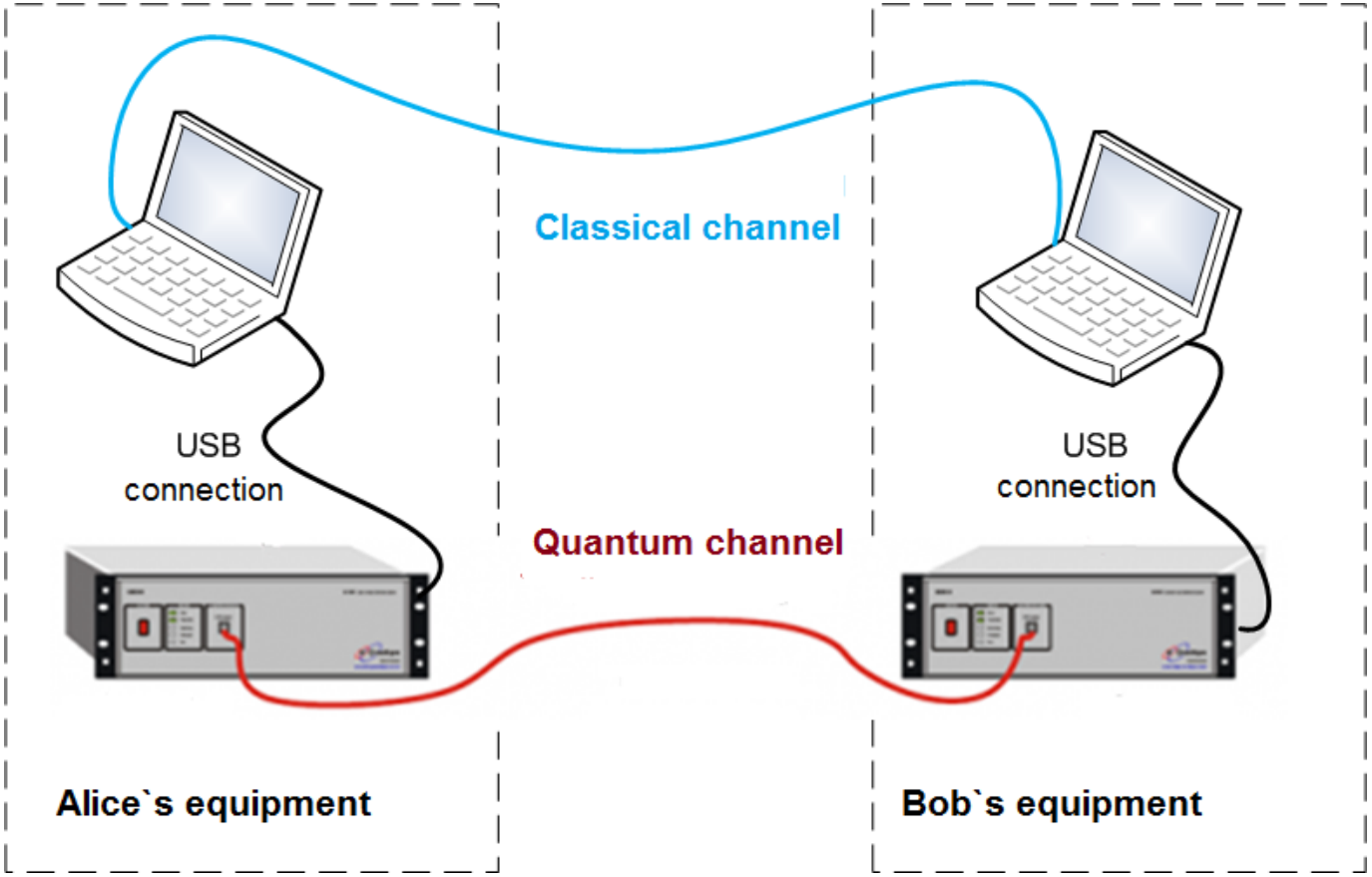
New trends

Homomorphism (malleability)

- allows processing of encrypted data without needing to first decrypt it
- best-suited to use in Cloud computing
- so far additive homomorphism and multiplicative homomorphism
- RSA and ElGamal are partially homomorphic (multiplication only)

New trends

Quantum Key Distribution (QKD)



HOMEWORK

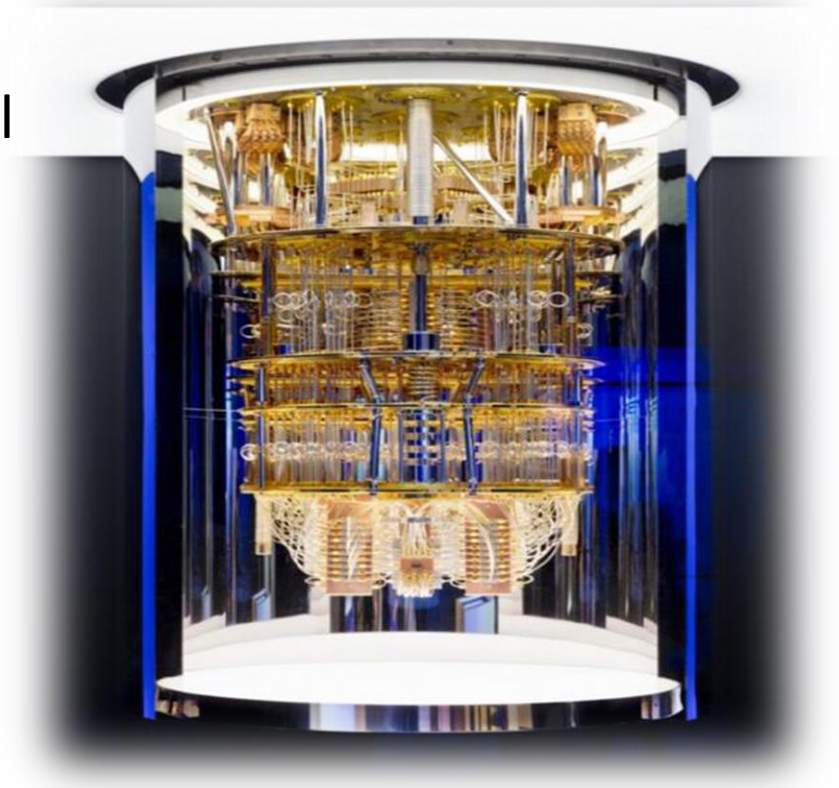
=

Half Of My Energy Wasted On Random Knowledge

Quantum Computing

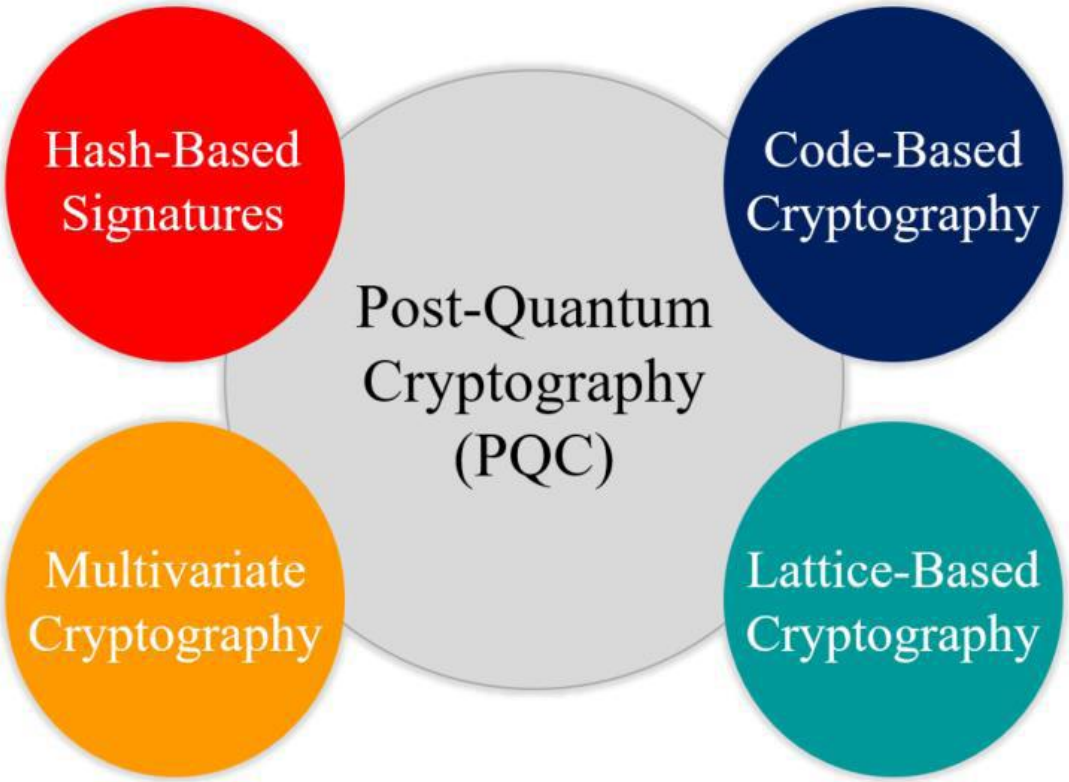
— What's its the impact on classical cryptosystems:

- symmetric
- asymmetric
- hash functions



New trends

Post-Quantum Crypto



Remember!

Strong crypto is not a silver bullet!

WSJ CYBERSECURITY

Capital One Breach Highlights Shortfalls of Encryption

Capital One said in a statement this week that it uses encryption "as a standard," but the method used by the hacker "enabled the decrypting of data." The bank didn't respond to questions about its encryption practices.

ars TECHNICA

BIZ & IT TECH SCIENCE POLICY CARS GAMING & CULTURE FORUMS

OVERFLOWING THE BUFFER —

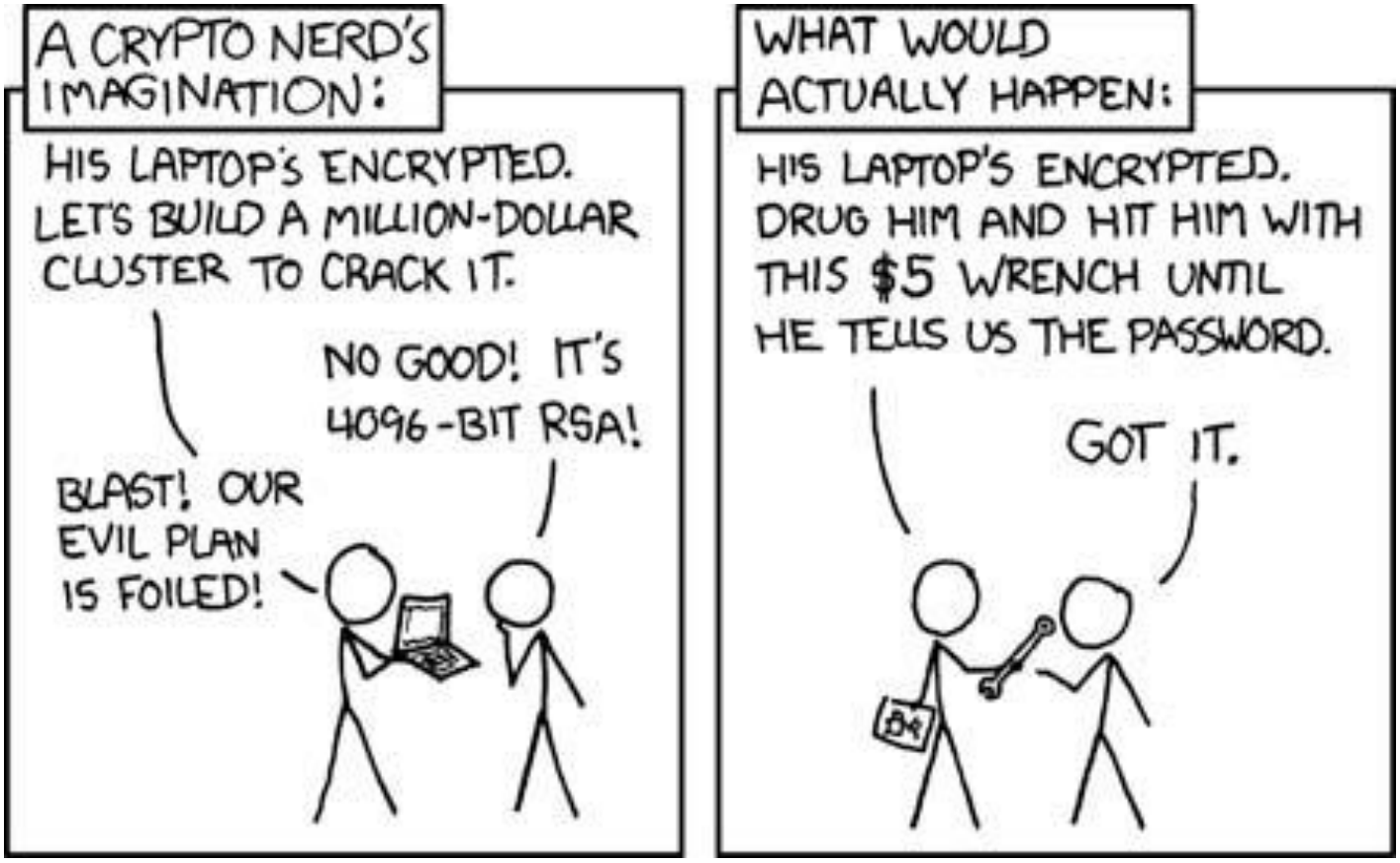
Google's Project Zero discloses Windows 0day that's been under active exploit

Security flaw lets attackers escape sandboxes designed to contain malicious code.

DAN GOODIN - OCT 30, 2020 7:38 PM UTC

CVE-2020-117087 stems from a buffer overflow in a part of Windows used for cryptographic functions. Its input/output controllers can be used to pipe data into a part of Windows that allows code execution. Friday's post indicated the flaw is in Windows 7 and Windows 10, but made no reference to other versions.

"The Windows Kernel Cryptography Driver (cng.sys) exposes a \Device\CNG device to user-mode programs and supports a variety of IOCTLs with non-trivial input structures," Friday's Project Zero post said. "It constitutes a locally accessible attack surface that can be exploited for privilege escalation (such as sandbox escape)."



https://xkcd.com


“And now for something completely different.”
Monty Python

STEGANOGRAPHY

Steganography

στεγανός (steganós) – covered

γράφειν (gráfein) – writing

Trithemius – „Steganographia, hoc est ars per occultam ...” 15/16 c

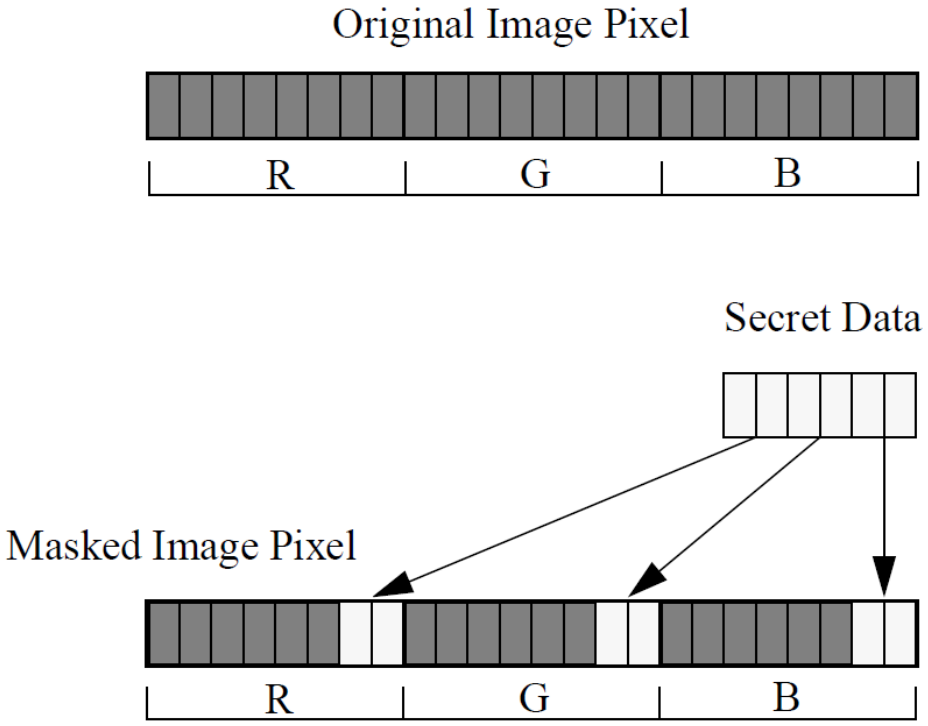
- hiding secret data within an non-secret object
- to avoid detection

→ watermarking:

  - genuineness proofs → monetary
  - intellectual property protection → multimedia

Steganography

RGB noise – LSB (Least Significant Bits)

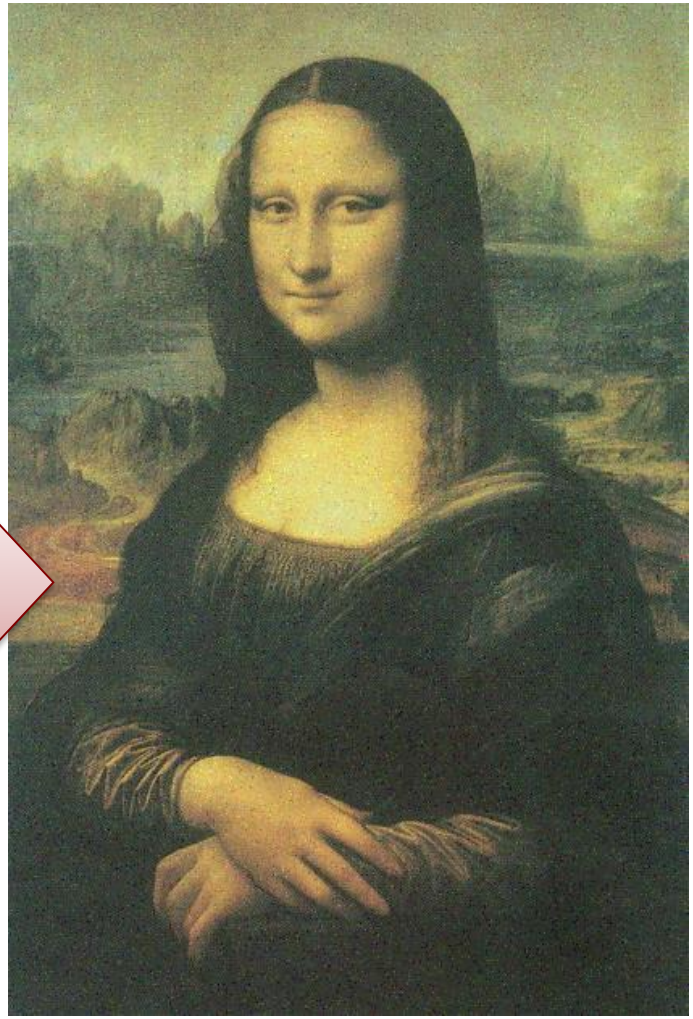


Steganography

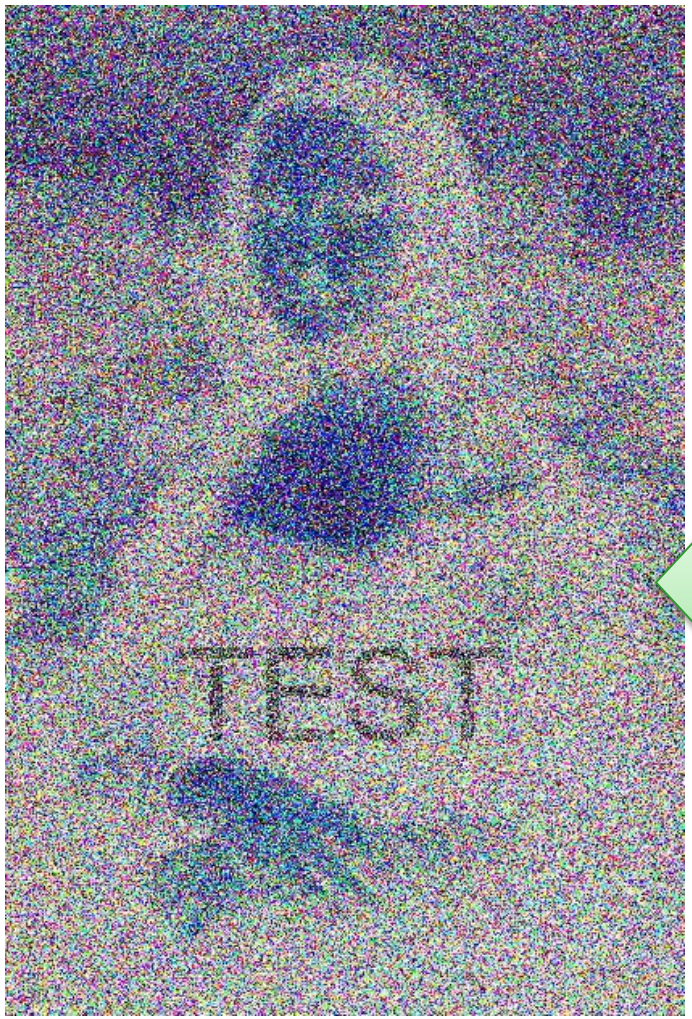
TEST

MASK

FILTER

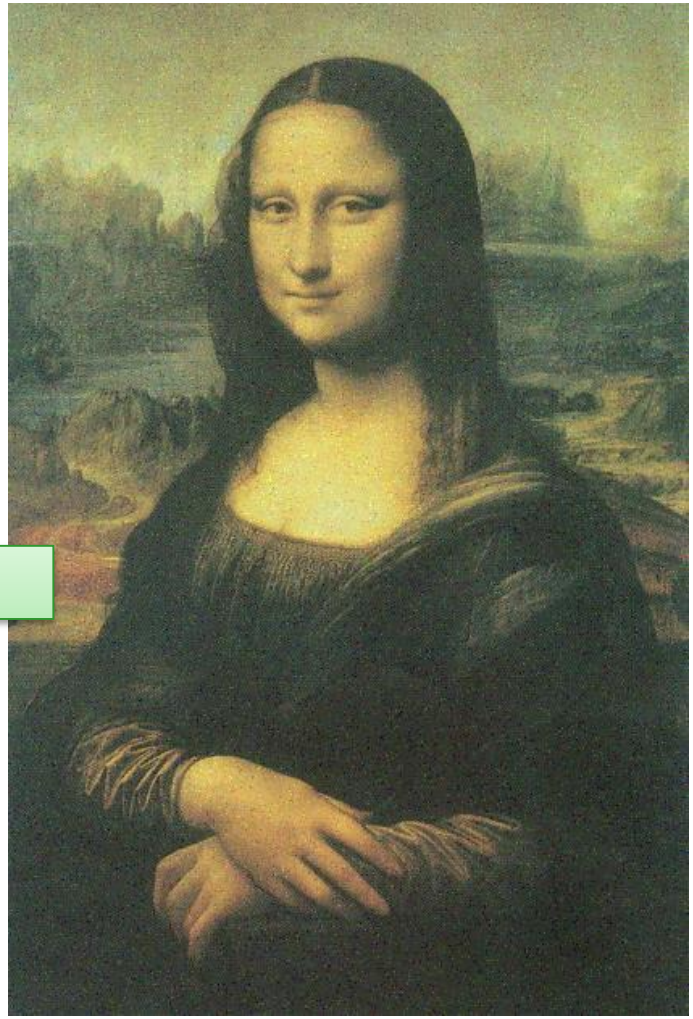


Steganography



UNMASK

FILTER




"That's all Folks!"