

# École Pour l'Informatique et les Techniques Avancées – EPITA

Masters program – 18 March 2022

Course: Data Privacy by Design

# Data Privacy by Design (PbD)

Course schedule (tentative)

Date & Time	No.	Topics	Duration (in hours)
04/03/2022 14:30–17:30	1	Data & its types, Information & knowledge, Introduction to Data Privacy by Design (PbD)	3 hours
18/03/2022 14:30–17:30	2	DPbd Case studies, Data privacy risks & solutions	3 hours
02/04/2022 10:00–13:00	3	Privacy Enhancing Technologies (PET's)	3 hours
22/04/2022 14:30–17:30	4	General Data Protection Regulation (GDPR), PbD and GDPR	3 hours
29/04/2022 14:30–17:30	5	Open session, Putting it all together, Quiz, Final project presentation	3 hours
<b>Total Lecture (hours)</b>			<b>15</b>

**Evaluation:** 10% Class attendance + 10% Class participation  
+ 30% Class/home exercises + 50% Final Evaluation

# Lecture 2 Outline

- ▶ **Data Privacy by Design (DPbD)**
  - Class exercise solution
  - Methodology
  - Take-away
- ▶ Data privacy risks and solutions
  - Top privacy risks
  - Crypto package
  - Class exercise 3
  - Data masking (Anonymization vs Pseudonymisation)
  - Data masking (common techniques)
- ▶ Putting it altogether
  - Class exercise 4

# Case Study 2:

## European Electronic Toll Service (EETS)

- ▶ Defined functionality:
  - Pay according to road use: time, distance, road type, ...
- ▶ Requirements:
  - Privacy & integrity risks to be mitigated:
    1. Third party access to traffic/location data of driver.
    2. Abuse of traffic data by authority performing the billing (location data cannot be easily anonymized).
  - The provider needs to know the final fee to charge
  - The provider must be reassured that this fee is correctly computed and users cannot commit fraud

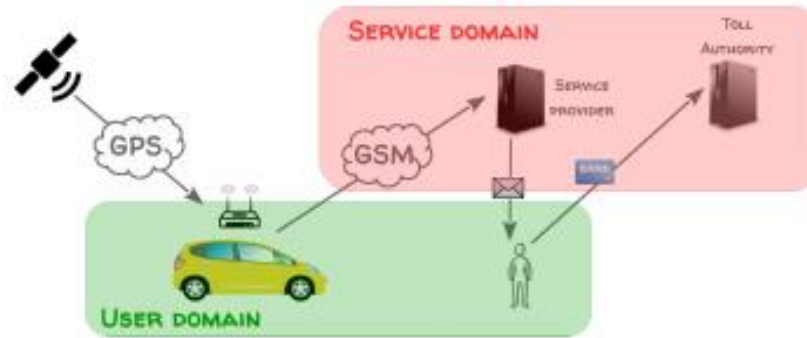
Note: Location as a means to compute above points -> not intrinsic

### Class exercise 2: activity

#### Form basic information model & perform the 4 activities:

1. Classify entities in domains
2. Identify necessary data for providing the service
3. Distribute data in architecture
4. Select technological solutions

RECAP



## Activity 1: Classify Entities in domains

**User domain:** Components under control of the user e.g., user devices, GPS receiver, ...

**Service domain:** Components outside the control of the user, e.g., backend system, ...

## Activity 2: Identify necessary data for providing to the service domain

Location data – to compute bill

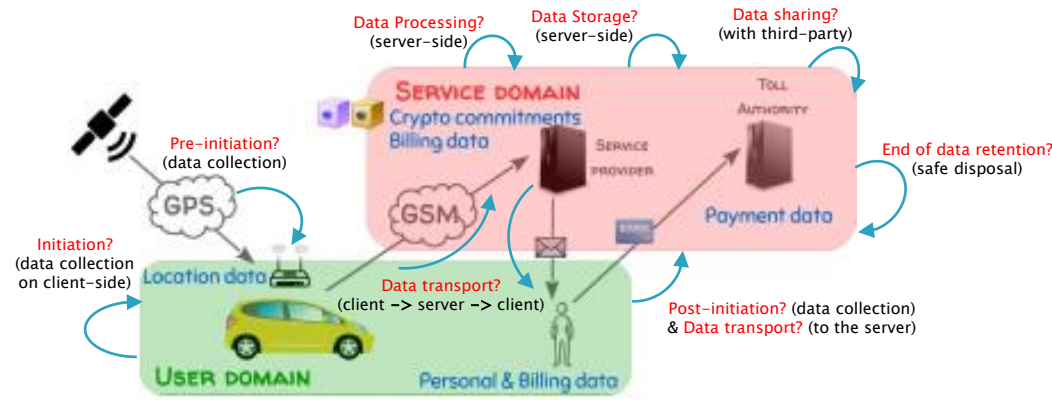
Billing data – to charge user

Personal data – to send bill

Payment data – to perform payment, ...

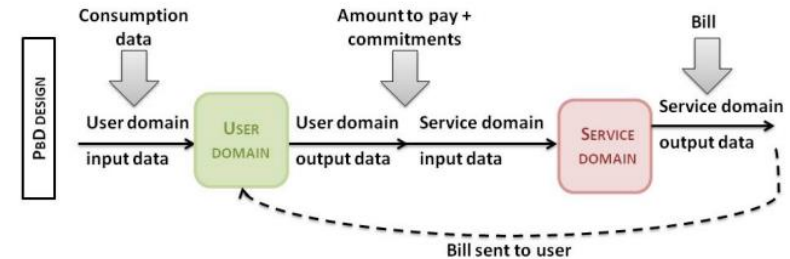
## Activity 3: Distribute data in architecture

SOLUTION 2/2



- Is location data needed or only the amount to bill?
- Crypto commitments?
  - ...

## What can go wrong?



## Risk analysis? (Likelihood vs Impact)

1. Data compromises
2. Service compromises
3. Data sharing
4. ...

## Activity 4: Select technological solutions

1. Not sending the data (local computations)
2. Encrypting the data / Advanced privacy-preserving protocols
3. Obfuscate the data / Anonymize the data
4. ...

Keeping as much data as possible out of the service domain for satisfying the data integrity requirements

Address threats in  
order of their priority  
(per risk analysis)

# No fixed methodology

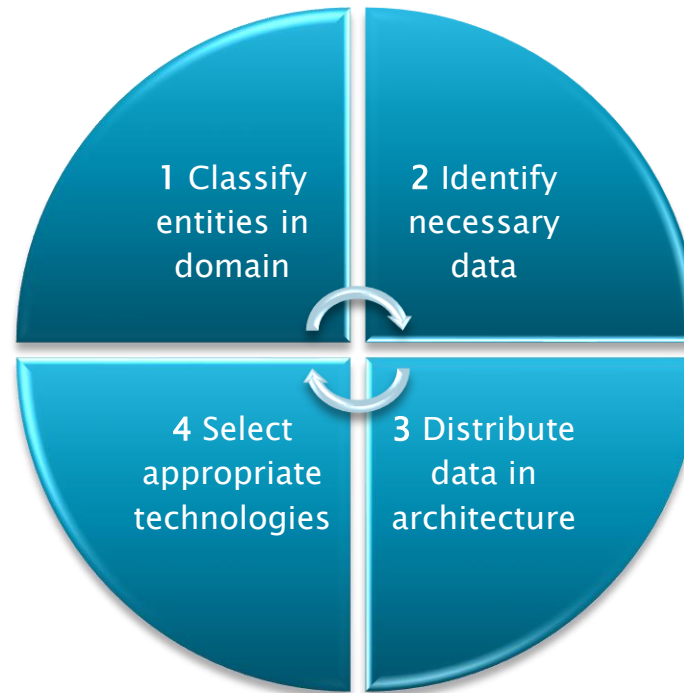
- ▶ Take **privacy considerations** and perform **risk management** at all levels of project management
- ▶ Assert data subject rights and integrate **appropriate controls to mitigate privacy risks** at all stages of development
  - E.g., requirements, specification, implementation, testing, deployment, maintenance
- ▶ Focus on raising **Transparency** of service/product:
  - Make your Privacy policy/Terms of service easy-to-understand
  - Take clear user consent (e.g. no pre-ticked boxes) with no shady tactics (e.g., clickbait)
  - ...



# Take away!

Assumptions:  
Functionality &  
requirement defined,  
Basic ref. model, ...

*Remember  
the Overall  
Goal, and 6  
strategies!*








*Covering full  
life-cycle  
using Agile  
approach*









# Lecture 2 Outline

- ▶ **Data Privacy by Design (DPbD)**
  - Class exercise solution
  - Methodology
  - Take-away
- ▶ **Data privacy risks and solutions**
  - Top privacy risks
  - Crypto package
  - Class exercise 3
  - Data masking (Anonymization vs Pseudonymisation)
  - Data masking (common techniques)
- ▶ **Putting it altogether**
  - Class exercise 4

# OWASP Top 10 Privacy Risks project

#	Type	Title	Frequency	Impact	Description
P1		Web Application Vulnerabilities	High	Very high	Vulnerability is a key problem in any system that guards or operates on sensitive user data. Failure to suitably design and implement an application, detect a problem or promptly apply a fix (patch) is likely to result in a privacy breach. This risk also encompasses the OWASP Top 10 List of web application vulnerabilities and the risks resulting from them.
P2		Operator-sided Data Leakage	High	Very high	Failure to prevent the leakage of any information containing or related to user data, or the data itself, to any unauthorized party resulting in loss of data confidentiality. Introduced either due to intentional malicious breach or unintentional mistake e.g. caused by insufficient access management controls, insecure storage, duplication of data or a lack of awareness.
P3		Insufficient Data Breach Response	High	Very high	Not informing the affected persons (data subjects) about a possible breach or data leak, resulting either from intentional or unintentional events; failure to remedy the situation by fixing the cause; not attempting to limit the leaks.
P4		Consent on Everything	Very high	High	Aggregation or inappropriate use of consent to legitimate processing. Consent is "on everything" and not collected separately for each purpose (e.g. use of website and profiling for advertising).
P5		Non-transparent Policies, Terms and Conditions	Very high	High	Not providing sufficient information to describing how data is processed, such as its collection, storage, and processing. Failure to make this information easily-accessible and understandable for non-lawyers.

P5		Non-transparent Policies, Terms and Conditions	Very high	High	Not providing sufficient information to describing how data is processed, such as its collection, storage, and processing. Failure to make this information easily-accessible and understandable for non-lawyers.
P6		Insufficient Deletion of Personal Data	High	High	Failure to effectively and/or timely delete personal data after termination of the specified purpose or upon request.
P7		Insufficient Data Quality	Medium	Very high	The use of outdated, incorrect or bogus user data. Failure to update or correct the data.
P8		Missing or insufficient Session Expiration	Medium	Very high	Failure to effectively enforce session termination. May result in collection of additional user-data without the user's consent or awareness.
P9		Inability of users to access and modify data	High	High	Users do not have the ability to access, change or delete data related to them.
P10		Collection of data not required for the user-consented purpose	High	High	Collecting descriptive, demographic or any other user-related data that are not needed for the purposes of the system. Applies also to data for which the user did not provide consent.

Version 2.0 – 2021

Source: <https://owasp.org/www-project-top-10-privacy-risks>

## Some Examples

### P2: Operator-sided Data Leakage

- ▶ Lack of awareness
- ▶ Poor access management
- ▶ Unnecessary copies of personal data
- ▶ ...

Dark archives

Duplicates

Shadow IT

No consistent security

...

### P5: Non-transparent Policies, Terms & Conditions

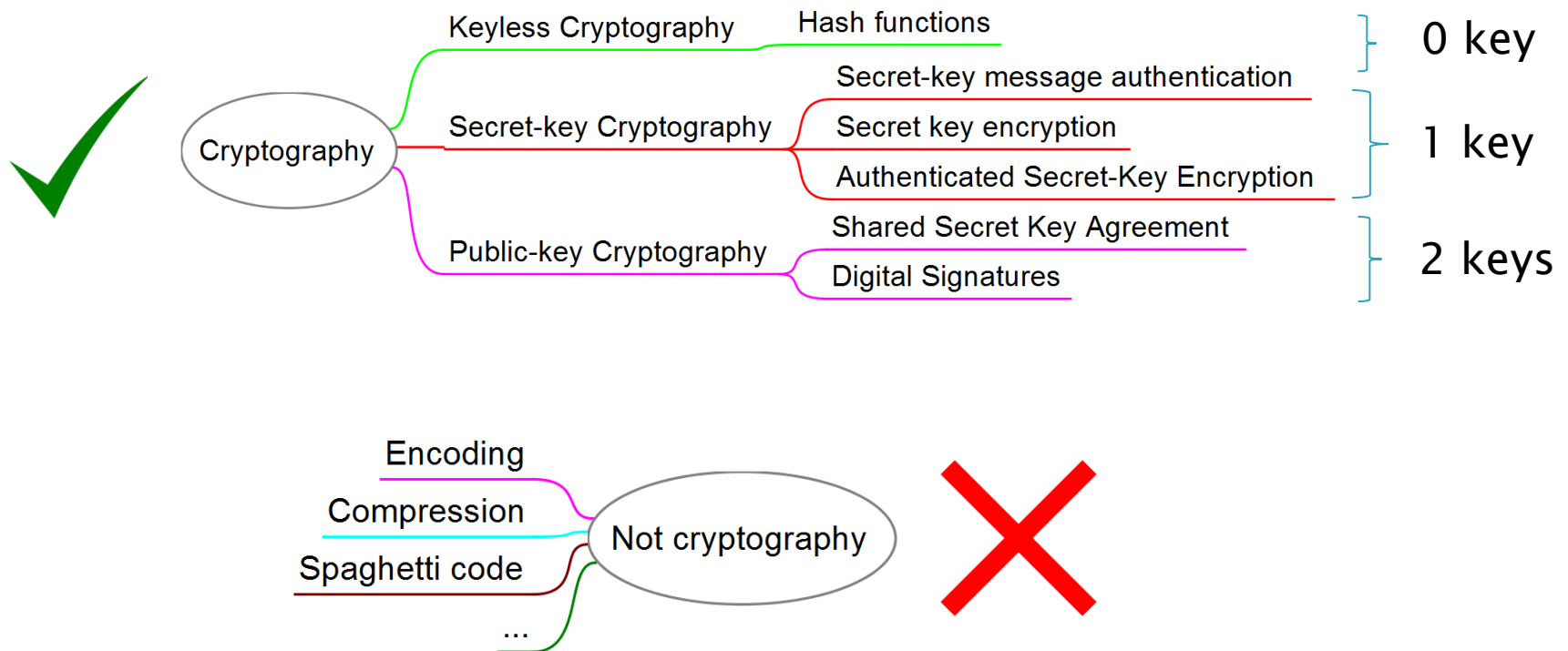
- ▶ Privacy Policies, Terms & Conditions are not up-to-date, inaccurate, incomplete or hard to find  
e.g., Check <https://priebot.org/polis>
- ▶ Data processing is not explained sufficiently
- ▶ Conditions are too long and users do not read them
- ▶ ...

✓ *"I have read and agree to the terms and conditions"*  
Is the **Biggest Lie** on the web.



I confessed  
**BiggestLie.com**

# Overview of cryptography concepts



# First Rule of Cryptography

Don't Implement it Yourself!

- ▶ Best left to the experts
  - Feel free to tinker
  - But don't deploy your experiments in production
- ▶ Always use a publicly scrutinized high-level crypto-library
- ▶ Crypto-library comparison:

Bouncy Castle	Legion of the Bouncy Castle Inc.	Java, C#	Yes	MIT License	Yes	Yes	Java 1.58 / August 18, 2017; 2 months ago <sup>[3]</sup> Java BC-FJA 1.0.0 / November 11, 2016; 11 months ago <sup>[4]</sup> FIPS 1.8.1 / December 28, 2015; 22 months ago <sup>[5]</sup> C# BC-FNA 1.0.1 / December 28, 2016; 10 months ago <sup>[6]</sup>
---------------	----------------------------------	----------	-----	-------------	-----	-----	--

Source: [https://en.wikipedia.org/wiki/Comparison\\_of\\_cryptography\\_libraries](https://en.wikipedia.org/wiki/Comparison_of_cryptography_libraries)

# Cryptographic feature?

- Simply put: Using math to secure an application
- Cryptographic algorithms can generally be grouped by two criteria's:
  1. How much information must be supplied by the developer?
  2. What is the intended goal? (primitives to achieve):
    - Confidentiality? (use encryption)
    - Integrity? (use signature)
    - Authenticity? (use signature)
    - Non-repudiation? (use signature)
    - Deniability (or repudiation)? (use signature)

# Keyless Cryptography

## ► Hash function (one-way data transformations):

This is Data Privacy by Design class.

Calculate Hashes Copy to clipboard (undo)

NTLM	6E341BAA05096D837B43EA5080C4C816	MD2	cc0f51ed125450215a649baa892cb48c	MD4	afe2f55637e1a13f5e6cc9d6a590989b
MD5	540b5d69a321042984c98f8994b7de6e	MD6-128	b4a5e1cd6161702fa321cae28579face	MD6-256	f114e1f2a12b5b41ee7778074ee1f0e47485e988c01
MD6-512	c16d393114e9aacc1898467d200ace8442b9099c36f	RipeMD-128	3fd5b5711d440890c9008a983d35406e	RipeMD-160	be2d8b3fc73ec0ee2cc6bed20042ed8c7c114e50
RipeMD-256	5a14ef4e4eeda7308a8328fa9c5c5651d5a136d1f3i	RipeMD-320	af37f0f8306e365978bd824c0488d4ff0da781dd42d7	SHA1	4aa1dcef070491c74e8a89c9aa1377ec3669c5ad
SHA3-224	bcee92ed497cb795ff3965c73c4113c08e58848c519i	SHA3-256	b485d662ff996ff676a38682bacd87ac838fe43668cc	SHA3-384	399aec902cc8cb4b1aa8c92b9c620a6e4c3bc8d8929
SHA3-512	d0a9d6255a6b59d7e6fed2b584254bf7c11fae91c26i	SHA-224	74c4b30f47a5a220454b868c21f84371cf3e8a55ecec	SHA-256	6fbe260de57d81c95fe2373282288bf2d3d3dc13a9cf
SHA-384	e349d39221c43d1becc036341878978cb4cd497082f	SHA-512	248bbe7aa87e28ebc7864a56c472e1d3844be57a1f	CRC16	d0cc
CRC32	d93abf03	Adler32	f57e0d06	Whirlpool	32d12503915cce79427b52e64f87feb2516375a3ac4

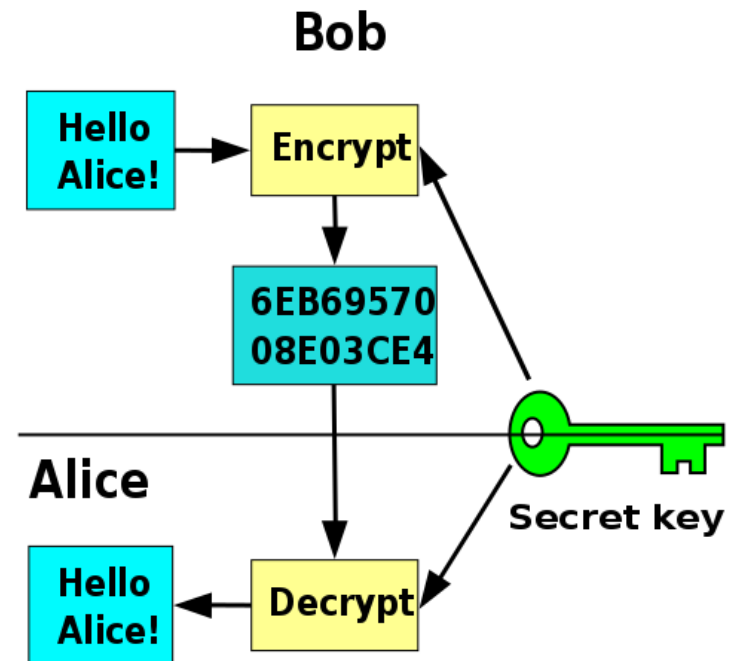
Source: <https://www.browsersling.com/tools/all-hashes>

- Accepts one input & returns a fixed-size output (depending on the algorithm)
- Any change to the input will result in a drastically different hash output
- Cannot easily be reversed from hash output to the original message – and that's the goal



# Secret Key Cryptography

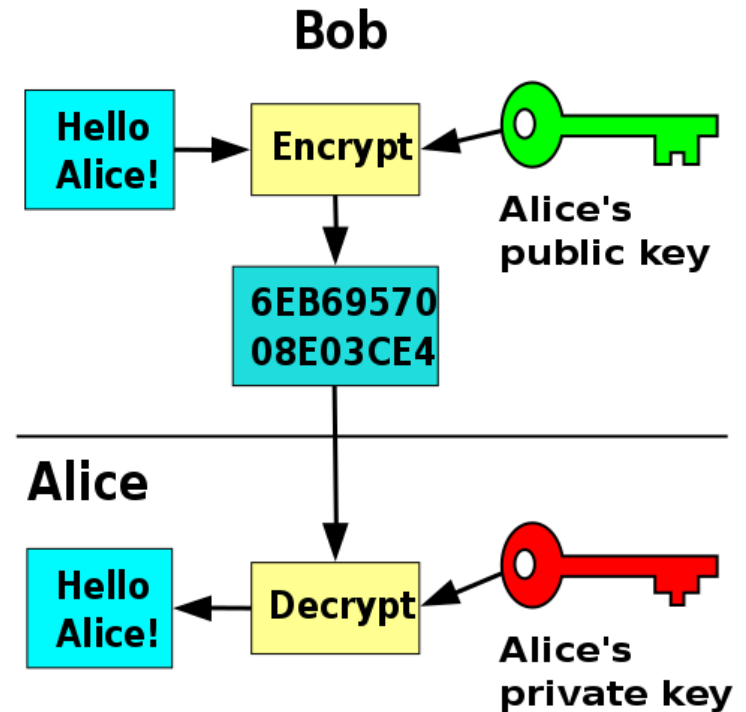
- Typically require two pieces of input: The message and a secret key
- A secret key should be a unique string of random bytes
- The secret key must be only known to sender and intended recipient, and nobody else!



Source:  
[https://commons.wikimedia.org/wiki/File:Symmetric\\_key\\_encryption.svg](https://commons.wikimedia.org/wiki/File:Symmetric_key_encryption.svg)

# Public Key Cryptography

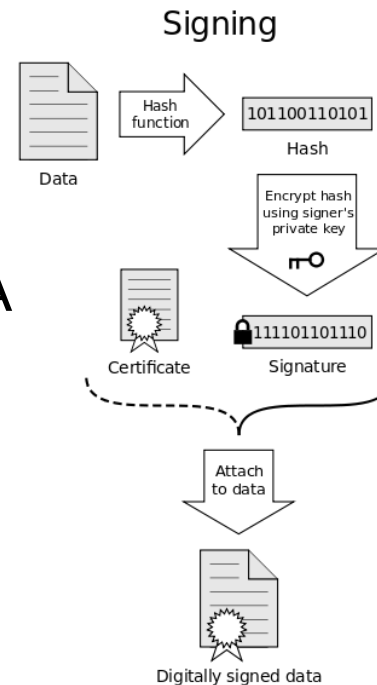
- ▶ Unlike secret key encryption, in public key cryptography, each participant has two keys (or a keypair):
  - **Private key:** never shared, used for:
    - Signing a message
    - Decrypting a message
  - **Public key:** mathematically related to the private key, shared with everyone
    - Used for encrypting a message
    - Used for verifying digital signatures



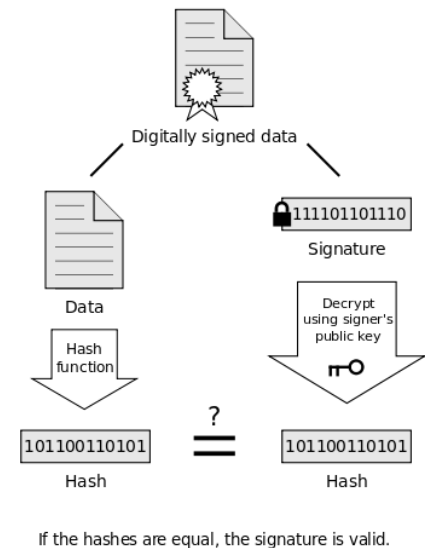
Source: [https://en.wikipedia.org/wiki/Public-key\\_cryptography#/media/File:Public\\_key\\_encryption.svg](https://en.wikipedia.org/wiki/Public-key_cryptography#/media/File:Public_key_encryption.svg)

# Digital Signatures

- ▶ A digital signature is calculated from a **message** and a **private key**:
  - Algorithm such as EdDSA (Edwards-curve Digital Signature Algorithm) or RSA (Rivest-Shamir-Adleman) are commonly used
  - Anyone else with a copy of respective **public key** can verify that a particular message was signed by someone's private key



## Verification



Source:

[https://commons.wikimedia.org/wiki/File:Digital\\_Signature\\_diagram.svg](https://commons.wikimedia.org/wiki/File:Digital_Signature_diagram.svg)

# Crypto hashes & Password hashes (don't confuse them)

Simple Hashes	Password Hash (schemes)
<ul style="list-style-type: none"><li>• Fast</li><li>• Only one input: The message</li><li>• E.g., SHA, Whirlpool, ...</li></ul>	<ul style="list-style-type: none"><li>• Intentionally slow</li><li>• At least three inputs:<ol style="list-style-type: none"><li>1. The password</li><li>2. A per-user salt</li><li>3. A cost factor (how expensive to make the computation) i.e., random/fixed iterations</li></ol></li><li>• E.g., Scrypt, Bcrypt, ...</li></ul>

# Common pitfalls & good practices

- Encoding and compression algorithms are both reversible, keyless transformations of information – and are not cryptographic
- Managing/protecting keys is hard!
  - Key life cycle management (generation, distribution, rotation/destruction), compromise (or recovery, zeroization), storage, sharing agreement, ...
  - Make sure all of key management aspects are well-thought beforehand
- Do not just assume that your code providing crypto. commitments will just work
  - Perform proper testing and code review
- Check ‘OWASP Top 10 privacy risks project’ while determining possible data privacy risks: [https://owasp.org/www-pdf-archive/OWASP\\_Top\\_10\\_Privacy\\_Countermeasures\\_v1.0.pdf](https://owasp.org/www-pdf-archive/OWASP_Top_10_Privacy_Countermeasures_v1.0.pdf)
- Always use trusted crypto library that has been scrutinized by experts
- Other: Instead of posting hashes, the software vendor can instead digitally sign their software package with their Private key and share their public key far and wide
  - When user downloads the file, the respective signature should also be downloaded and by using the verified public key, authenticity can be checked e.g., Minisign, GPG signature, ...
- ...

# Recommended ciphers & key length

Obsolete/ <del>Discouraged</del>	Encouraged
RC4, DES, TDES, <del>IDEA</del>	AES (128, 256, ...)
RSA ( <del>768</del> , <del>1024</del> ), El Gamal – DSA	RSA (2048, 4096, ...)
ECC ( <del>130</del> , <del>160</del> )	ECC (224), ECDSA
MD5, SHA-0, <del>SHA-1</del>	SHA-2, SHA-3, Whirpool, Argon2

# The crypto package

- Don't implement it yourselves
- +
- Use right key management tactics / appropriate system architecture
- +
- Use valid (non-obsolete) ciphers
- +
- Use valid (non-obsolete) cipher key lengths
- +
- Use reputed, publicly accepted and open-source cipher implementation
- +
- Keep crypto. libraries up-to-date
- +
- Perform testing & code review



# Class exercise 3

1. Watch following video:

CPDP 2020: Privacy enhancing technologies and AI:  
<https://www.youtube.com/watch?v=plOoeLB370A>

2. Write 10 points that you learned out of that video in a .txt or .doc file.

3. Save your .txt or .doc file and upload it to the 'Teams' assignment section (using your EPITA account).

Deadline: See 'Teams' Assignment section

# Data masking

- ▶ Process of obfuscating original/sensitive data
- ▶ The two main categories include:



## Anonymization

Information rendered anonymous, such that the data subject is no longer identifiable



## Pseudonymization

Information rendered neither anonymous nor directly identifying

# Anonymization vs Pseudonymisation

	Anonymization	Pseudonymization
Key difference (under usage scenario)	Anonymous data cannot be re-identified	Pseudonymous data allows for some form of re-identification
Data	Anonymization is mainly used for sensitive personal information such as: Names, IDs (CC, ID, ...), Addresses, Phone numbers, etc	Any data

# Anonymization & PbD

## Identity

First name: Bob  
Last name: Dyer  
Credit Card: 125 968

**Unique identifiers:**  
Apply Anonymization/  
Pseudonymisation?

## Other data

Age: 36  
Gender: Male  
Nationality: Finnish  
Lang: C, Assembly  
Company: XXX

**Quasi-identifiers:**  
Apply Anonymization/  
Pseudonymisation?

## Full data

First name: Bob  
Last name: Dyer  
Age: 36  
Credit Card: 125 968  
Gender: Male  
Nationality: Finnish  
Lang: C, Assembly  
Company: XXX

*Pseudonymized data can be  
attributed when the identity  
is added to the data*

Anonymization Techniques: Noise addition, Aggregation, ...  
-> Pseudonymisation techniques can be extended to achieve anonymization

# Pseudonymisation & PbD

- ▶ Pseudonymisation often satisfy requirements to implement “*privacy by design and by default*” and therefore is encouraged
  - Use of pseudonymous data is emphasized where personal data is used for historical or scientific research or for statistical purposes
- ▶ Data masking techniques:
  - Scrambling/Obfuscation (e.g., Name: Bob → BBO)
  - Encryption/Hashing (e.g., Name: Tyler → 8cbx2)
  - Substitution and/or shuffling (e.g., Credit Card no. : 125 978 → X25 798; X=1, 97→79)
  - Tokenization (e.g., Credit Card: 125 968 → akjcn809)
  - Blurring (approximation): (e.g., Age: 36 → Above 30)
  - ...

# Common data masking techniques

Category	Sub-category	Techniques	Application scenario
Anonymization	Randomization	Noise addition	Numeric data
		Permutation	Numeric data needs to be reversible
		Differential privacy	Big data statistics
	Generalization	Aggregation	Big data statistics
		K-anonymity	
		L-diversity	
		T-closeness	
Pseudonymization		Encryption (AES256)	Data needs to be reversible
		Hash (HMAC-SHA256)	Fixed length value
		Tokenization	Keep data format such as ID

# Lecture 2 Outline

- ▶ **Data Privacy by Design (DPbD)**
  - Class exercise solution
  - Methodology
  - Take-away
- ▶ **Data privacy risks and solutions**
  - Top privacy risks
  - Crypto package
  - Class exercise 3
  - Data masking (Anonymization vs Pseudonymisation)
  - Data masking (common techniques)
- ▶ **Putting it altogether**
  - **Class exercise 4**



# Putting it all together!

Data is a commodity

+

Data Privacy by design (PbD) is essential

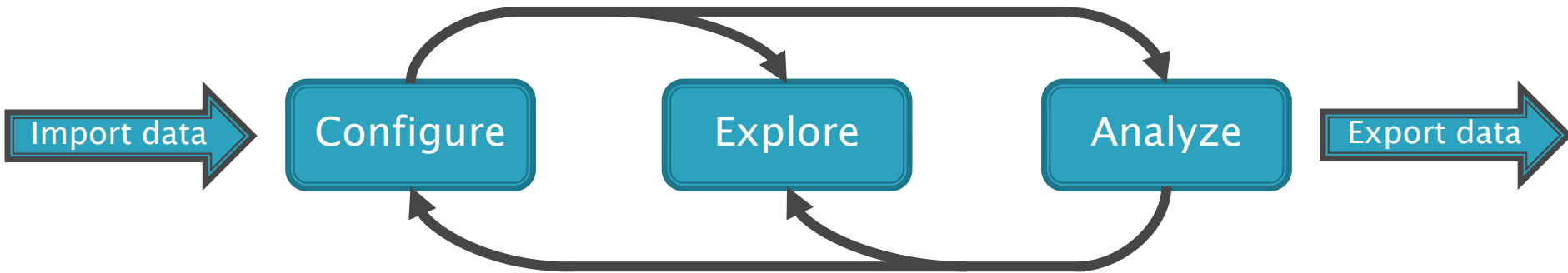
+

Always remember the Crypto package!

+

To avoid data privacy risks, use appropriate  
Anonymization/pseudonymisation techniques

# Class exercise 4 (ARX introduction)



- ▶ Iterative process to successively refine transformation until desired result is obtained
  1. Define transformation model, privacy and coding model [wizard assistance]
  2. Filter and analyze the solution space, and organize transformations [privacy and utility measures]
  3. Compare and analyze input and output, regarding risks and utility

# Class exercise 4 (task)



## ► Use 'Arx.deidentifier.org'

[Opensource; Apache "License" Version 2.0]

1. Download (and install) Arx:  
<https://arx.deidentifier.org/downloads/>
2. Create a new project
3. Import **any example data set** (e.g., spreadsheet, csv, db file)
  - Preferably containing ~1000 entries
4. Perform data masking (using **any transformation/technique of your choice**)
5. Export/download your project (firstname\_lastname) & upload it to the 'Teams' assignment section (using your EPITA account)
  - If file size is  $\geq 50\text{MB}$ , then upload it to OneDrive (make sure to grant read access to 'mohammad-salman.nadeem@epita.fr') and include the access link in 'Teams' assignment section

Deadline: See 'Teams' Assignment section

# Lecture 2 ends here

- ▶ Course Slides: Go to MS Teams:  
'Data Privacy by Design Spring (S1) Spring 2022'  
–> Files section
- ▶ Send your questions by email:  
mohammad-salman.nadeem@epita.fr  
OR via direct message using MS Teams
- ▶ Thank You!