SLEEP BETTER WITH

GOOD CRYPTOGRAPHY

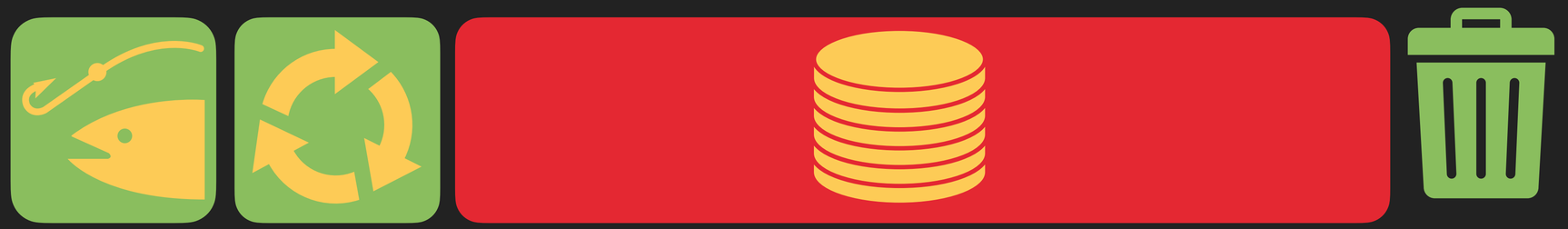
Jens Neuhalfen

https://github.com/neuhalje/presentation_content-encryption

WHO AM I?

- Jens Neuhalfen
- Age: Forty something
- IT since: ever
- Skills: Bridge between IT and business, IT-Security Management, writing software
- https://github.com/neuhalje

YOUR DATA



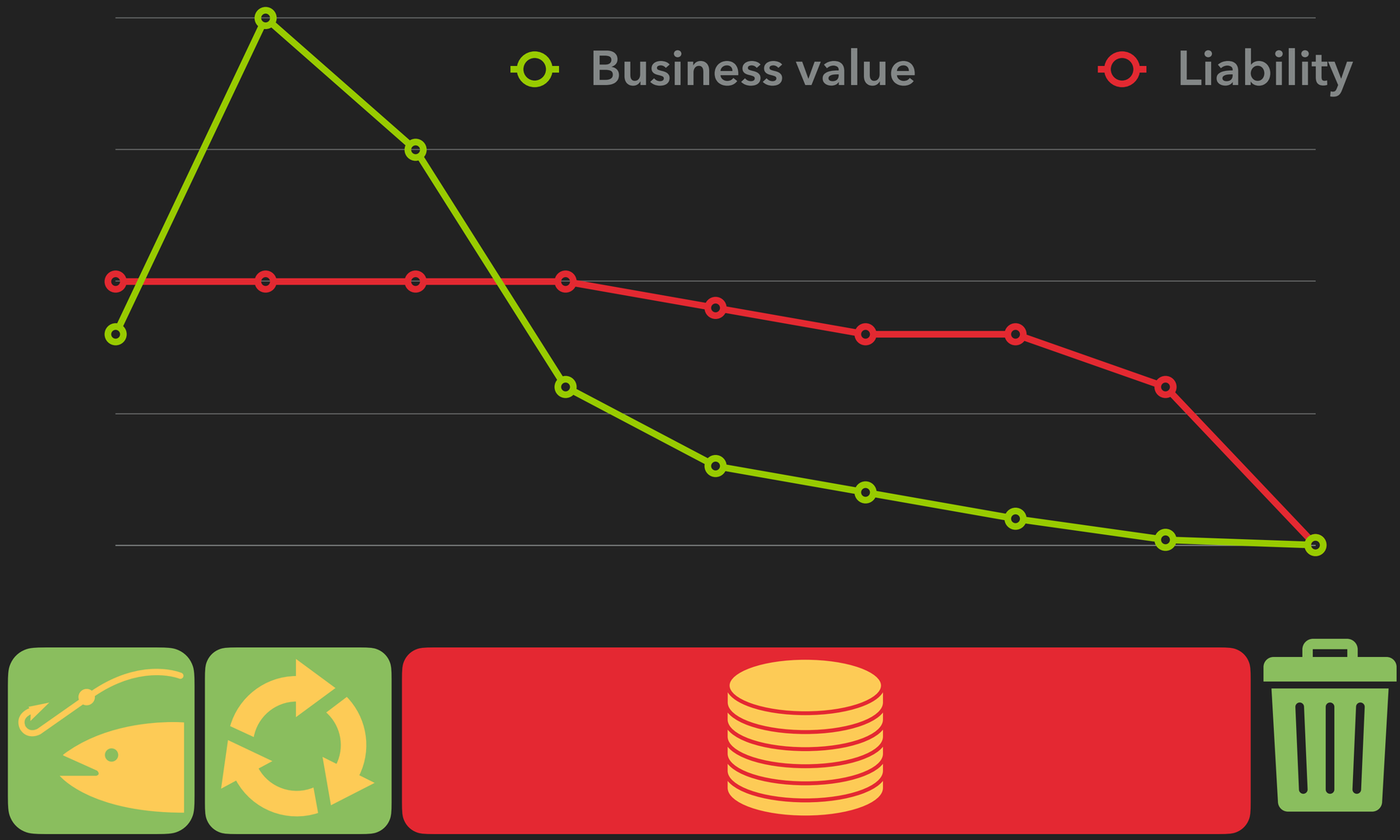
Collect

Process

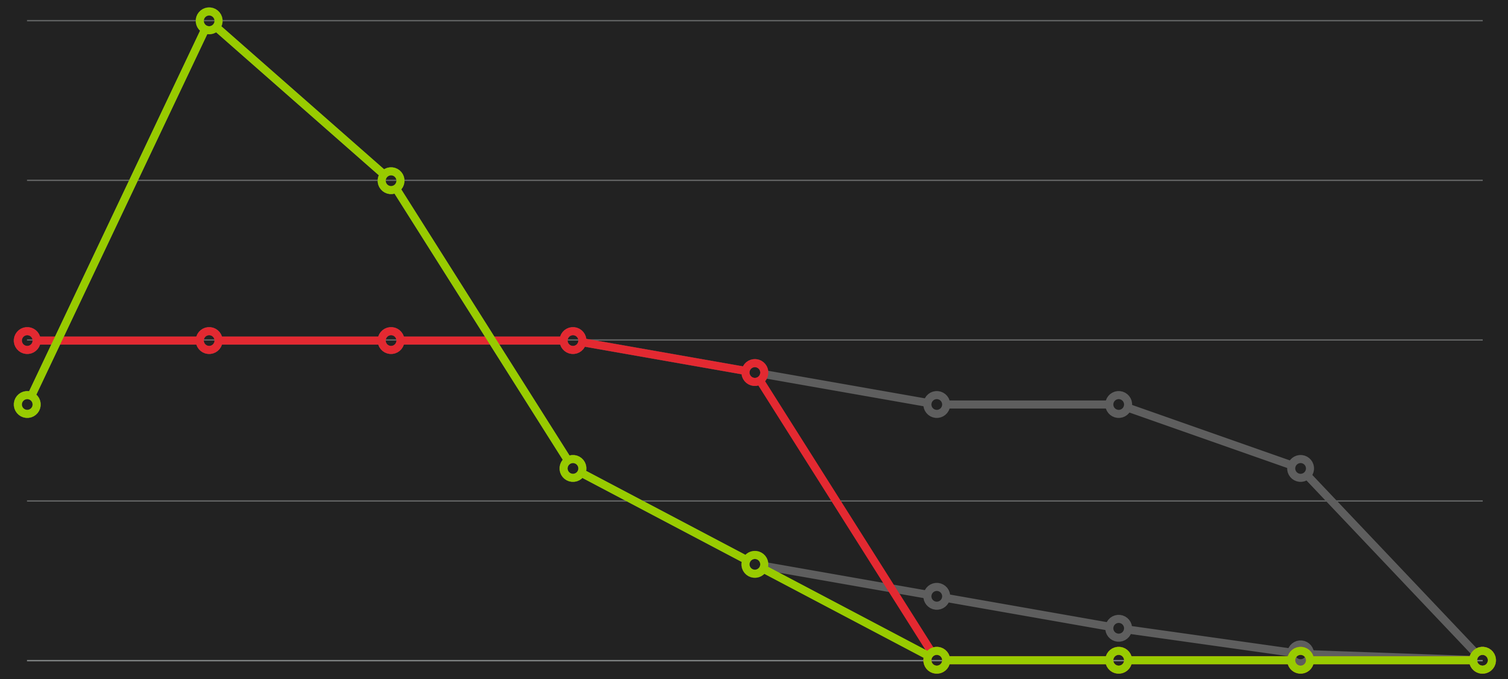
Store *but not use*

Delete

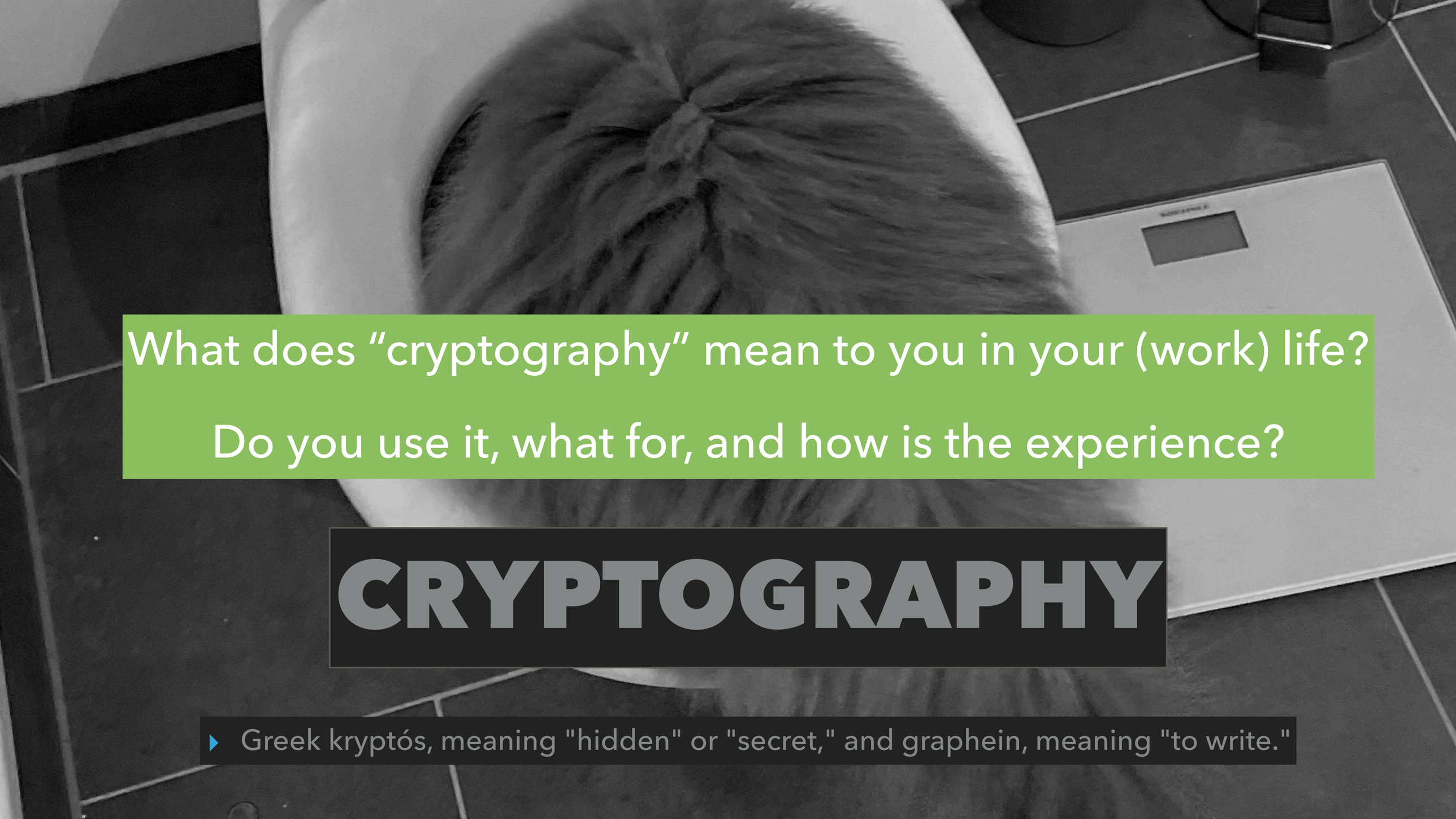
YOUR DATA



YOUR DATA



WHAT IS CRYPTOGRAPHY?



What does “cryptography” mean to you in your (work) life?

Do you use it, what for, and how is the experience?

CRYPTOGRAPHY

- Greek kryptós, meaning "hidden" or "secret," and graphein, meaning "to write."

I AM NOT A CRYPTOGRAPHER!

I AM NOT A LAWYER!

THIS TALK MADE ME A CRYPTOGRAPHY AND/OR LEGAL EXPERT

said no-one ever

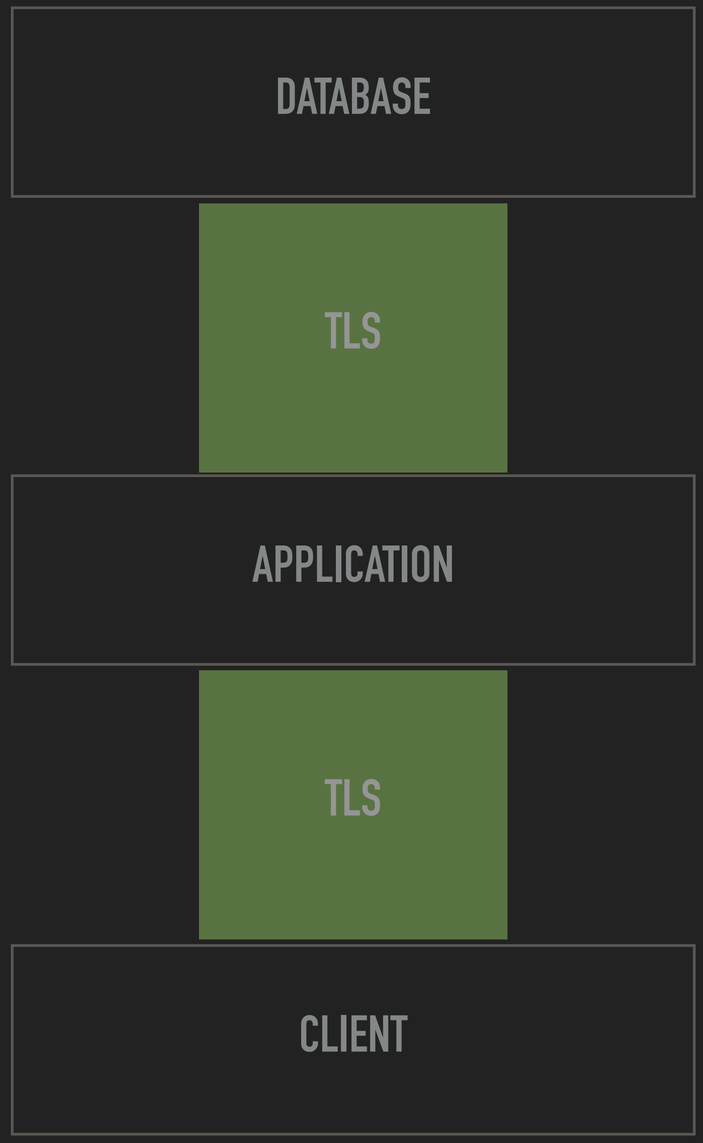
I AM NOT A LAWYER!

I AM NOT A CRYPTOGRAPHER!

YOUR FATHERS CRYPTO (*)

- CLIENT sends request
- APPLICATION applies logic
- DATABASE stores result
- Transport encrypted via TLS 🔒

SO, EVERYTHING IS SAFE?



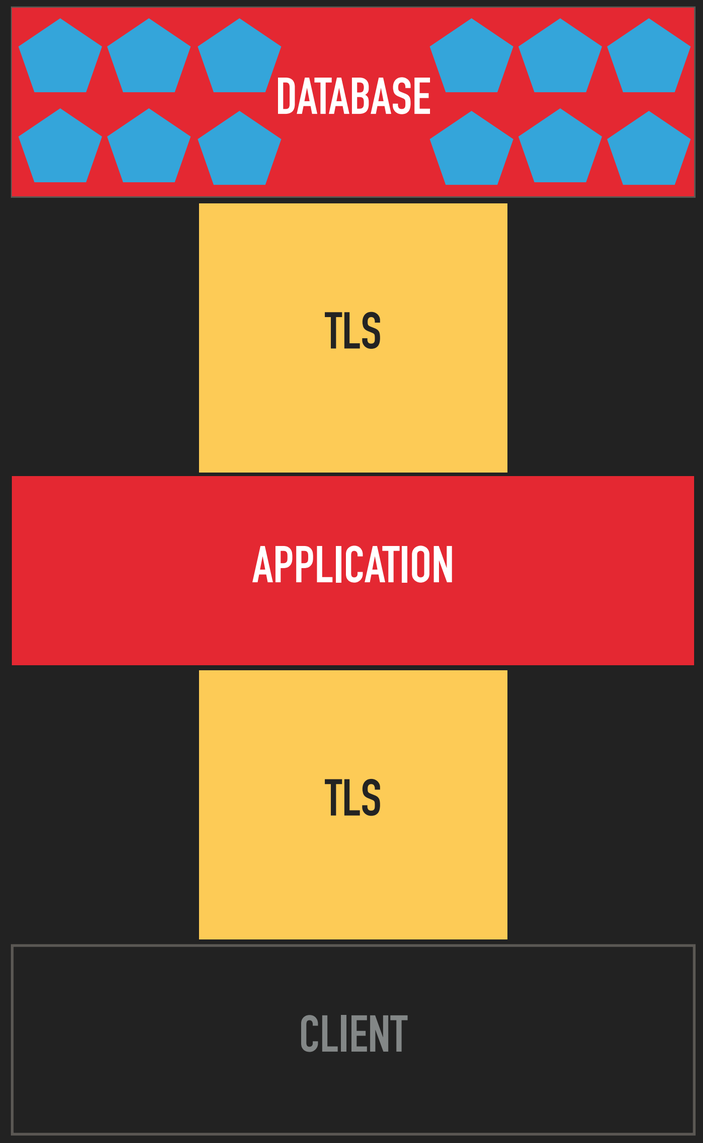
(*) I'm going to gloss over the whole cryptography nomenclature in these first slides. Bear with me.

WHAT ABOUT TLS?

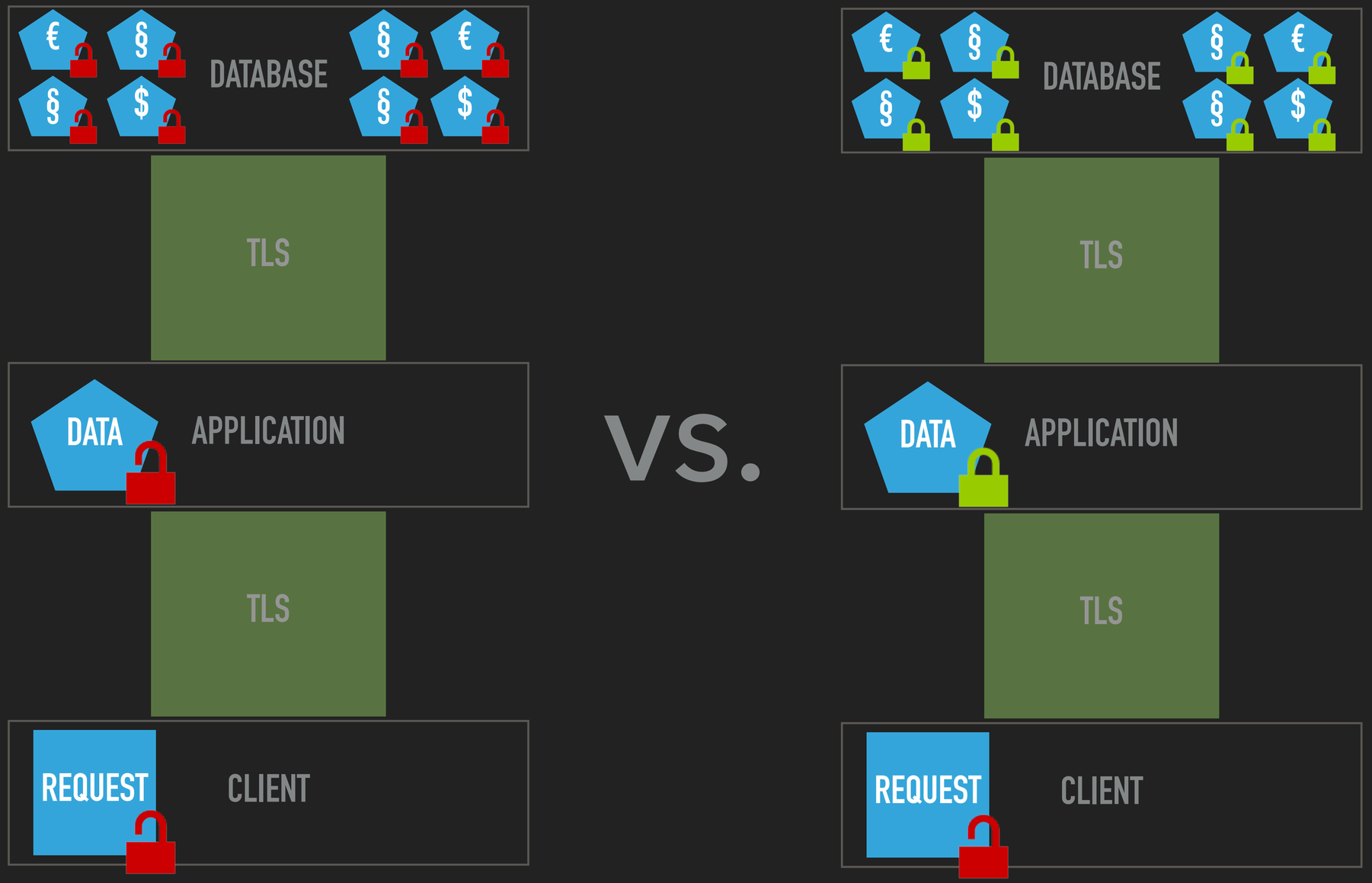
- Data is at rest for ~99.99998% of the time (*)
- Also: Heartbleed, POODLE, DROWN, Lucky13, Logjam, FREAK, …
- Also: Backups!

WHAT IS THIS ABOUT?

- Protect** data ‘itself’
- E.g. encrypted** data at rest
- Even: Protected** data while working with it



(*) 10s transit / 2 years storage (**) Cryptographically protected - details later!


DATABASE
TLS
DATA
APPLICATION
TLS
REQUEST
CLIENT
VS.
DATABASE
TLS
DATA
APPLICATION
TLS
REQUEST
CLIENT

WHY USE

CRYPTOGRAPHY?

IT'S THE LAW

- A lot of (here: German) laws take data protection seriously
- Bundesdatenschutzgesetz (BDSG)
- Telemediengesetz (TMG)
- Telekommunikationsgesetz (TKG)
- Strafgesetzbuch (StGB)
- Sozialgesetzbuch (SGB)
- EUDSGV

I AM NOT A LAWYER!

GENERAL DATA PROTECTION REGULATION

Applies to: You

- **Art. 32 Security of processing**

  Taking into account the state of the art, the costs of implementation [...] as well as the risk [...] for the rights and freedoms of natural persons, the controller and the processor shall implement [...] measures [...] including inter alia as appropriate

  A. the **pseudonymisation** and **encryption** of personal data;

  B. the ability to **ensure the ongoing confidentiality, integrity,** availability and resilience of processing systems and services;...

- Recital 83

GDPR

I AM NOT A LAWYER!

IT'S COMPLIANCE

- Company rules require encryption
- PCI DSS
- ISO 27001
- …

NOT CONVINCED YET?

ASK THE COMPETITION!

I grew tired of updating this!
a long time ago

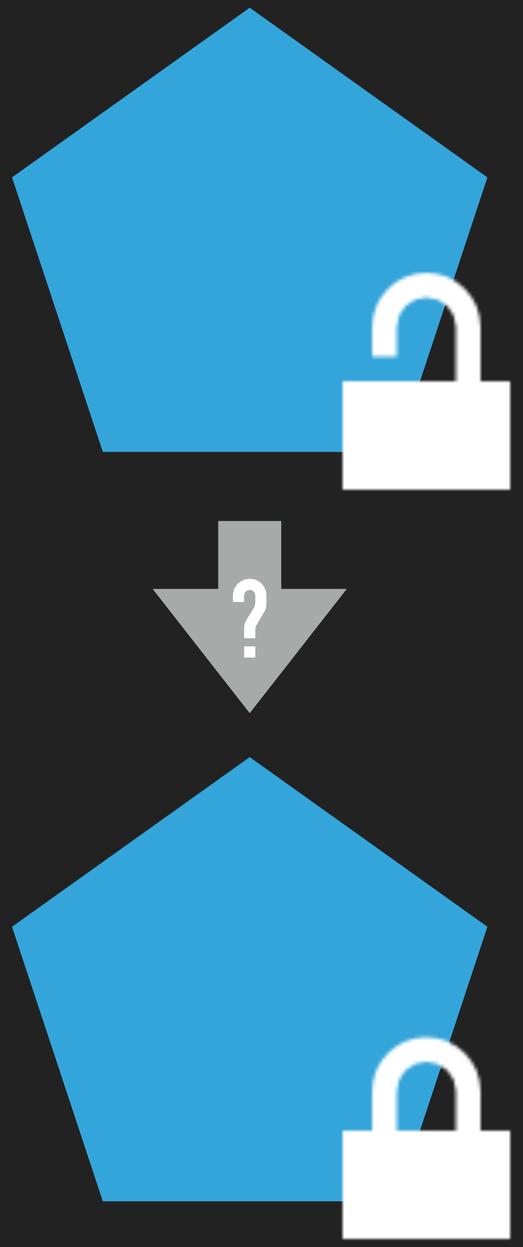
This could be YOUR CEO!

You do not want your CEO to look like that

Dido Harding (here still) CEO of Talk Talk

https://www.grahamcluley.com/hacked-talktalk-says-received-ransom-demand/

http://www.bbc.com/news/uk-34611857



HOW TO USE

CRYPTOGRAPHY

0 – REGULATIONS & INITIAL RISK ASSESSMENT

1 – DATA CLASSIFICATION

2 – DATA TREATMENT PLAN

3 – IMPLEMENTATION PLAN

0 – REGULATIONS & INITIAL RISK ASSESSMENT

1. List all relevant **laws, regulations**, etc. with a legal expert
2. List all **data items** (“Address”, “Bank Account”, …)
3. Estimate **initial risks** (damage & probability) for CIA violations with management and legal expert

1 – DATA CLASSIFICATION

Classify all data items with respect to regulations

Item	Regulation	
Customer		
- Name	GDPR	PII
- Bank Account	GDPR	PII / bank
…	…	…
Marketing E-Mail		
- Recipient	GDPR	PII
- Text	???	???
- Protocol	???	???

Classification depends on context!

E.g. GDPR Art. 9 “special categories of personal data”

THIS DEPENDS ON YOUR SPECIFIC APPLICATION/SCENARIO!

I AM NOT A LAWYER!

2 – DATA TREATMENT PLAN

Create a data treatment plan, and for each data item

- describe how (*) this data must be
- **... stored**
- **... transmitted**
- **... logged**
- **... backed up**
- and when it must be **deleted**!

(*) how: plain, masked (“XXXXX 123”), pseudonymised, anonymised, encrypted, hashed.

Also: consider integrity protection.

Item	Store	Transmit	Log	Backup	Delete
Customer					
- Name	Plain	Encrypted	Pseud.	Encrypted	GDPR
- Bank	Encrypted	Encrypted	Masked	Encrypted	GDPR
…	…	…			
Marketing E-Mail					
- Recipient	Plain	Encrypted	Pseud.	Encrypted	GDPR
- Text	???	???			
- Protocol	???	???			

Also: consider integrity protection.

THIS DEPENDS ON YOUR SPECIFIC APPLICATION/SCENARIO!

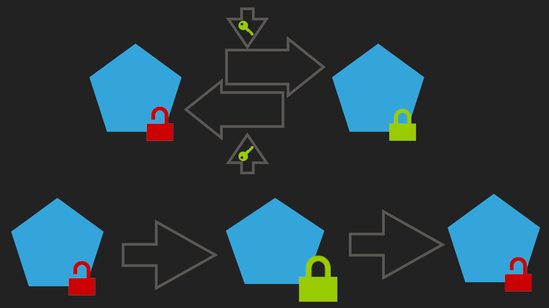
I AM NOT A LAWYER!

3 – IMPLEMENTATION PLAN

Update the requirements

- Merge Use Cases & data treatment plan
- Add cryptographic use cases

LONGER STORY: WRITE A CRYPTOGRAPHY & SECURITY CONCEPT



DES BLOWFISH AES

MD5 SHA-1 SHA-256

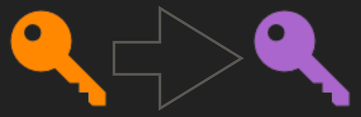
RSA-1024 RSA-2048 ?? POST QUANTUM ??

- [x] Data treatment ...
- [x] Use existing ...
- [x] ...

0X123456...

CRYPTO

PATTERNS



DATA

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

****

MOST IMPORTANT ADVICE: GET AN EXPERT OR AT LEAST READ AND UNDERSTAND THE DOCUMENTATION!

Like all power tools: Better RTFM than to lose an eye!

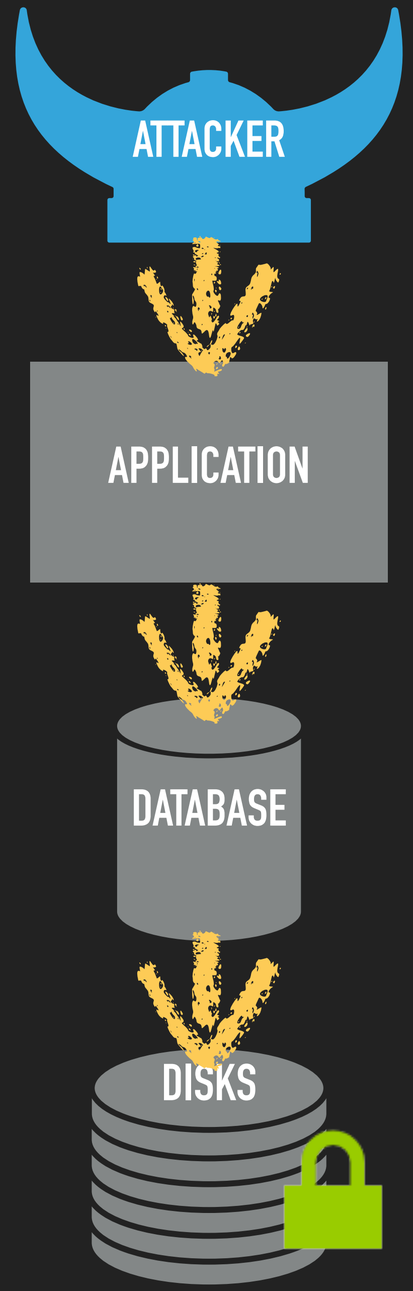
- At least be able to explain: “Hash vs. encryption”, “Integrity vs. encryption”, “Stream vs. block”, “Mode of operation”, “IV”, “Nonce”, “Padding”, “Key derivation”
- Identify and name your trust anchors

SOME WORDS THAT I MIGHT DROP AND THEN FORGET TO EXPLAIN

- **Cleartext**: What you can read. Not encrypted
- **Chiphertext**: Encrypted cleartext.
- **Hash**: Calculated from a text. Always the same length, regardless of the length of the text. Assumption: H(A) = H(B) => A = B
- **Key length**: length of the key/secret. E.g. “128 bits” for AES_128
- **Symmetric key length equivalence**: Asymmetric keys are much longer (e.g. RSA 3072) but scale differently. RSA3072 is ~128bit “symmetric key length”, RSA2048 is 112bits
- Generally you want key lengths >100 bit

PATTERNS

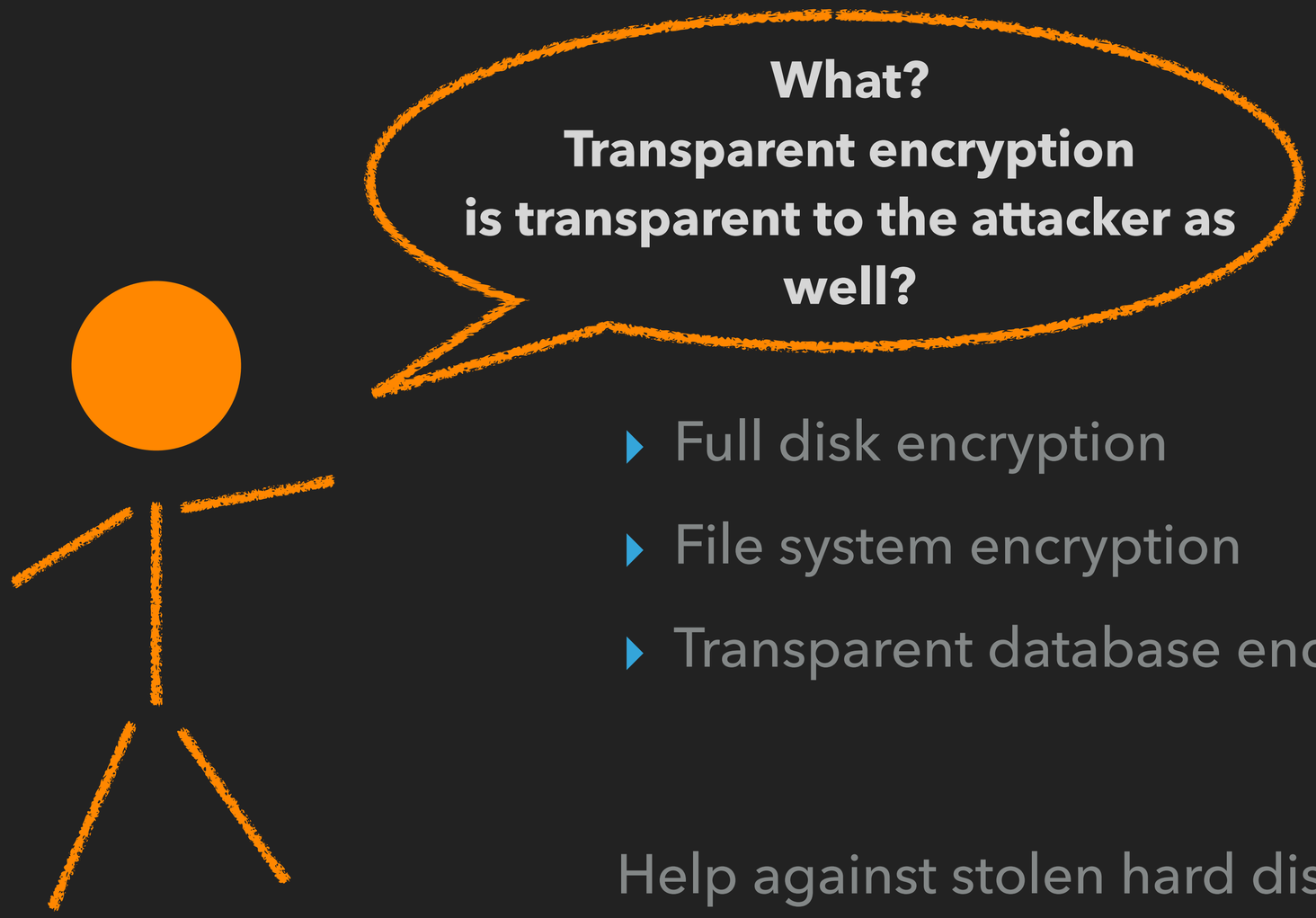
TRANSPARENT ENCRYPTION



The term “t**ransparent encryption**” is often used to describe encryption at the basic storage layer (disk, DB tables,…)

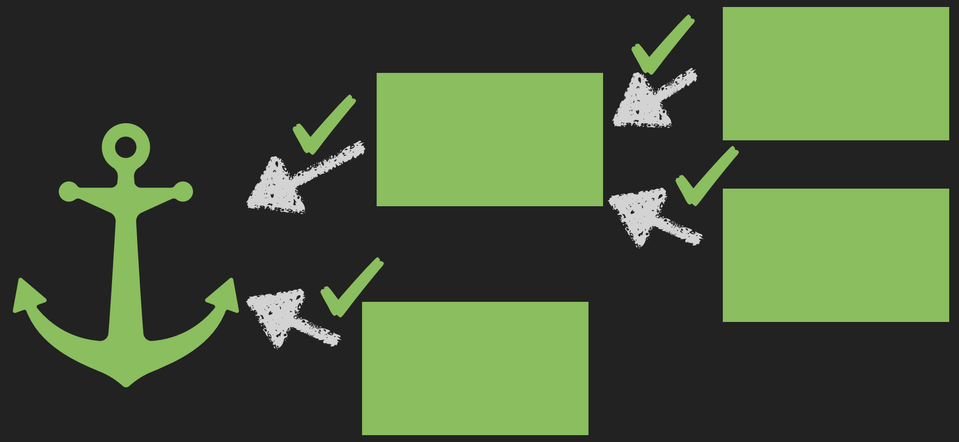
The disk is encrypted, so all data on the disk is encrypted as well. Does that mean that we have **effectively secured our data**?

TRANSPARENT ENCRYPTION



- Full disk encryption
- File system encryption
- Transparent database encryption

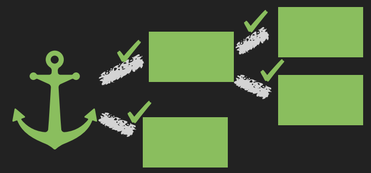
Help against stolen hard disks.



PATTERNS

TRUST ANCHORS

TRUST ANCHORS



Problem: You do not know when to start trusting / stop discussing

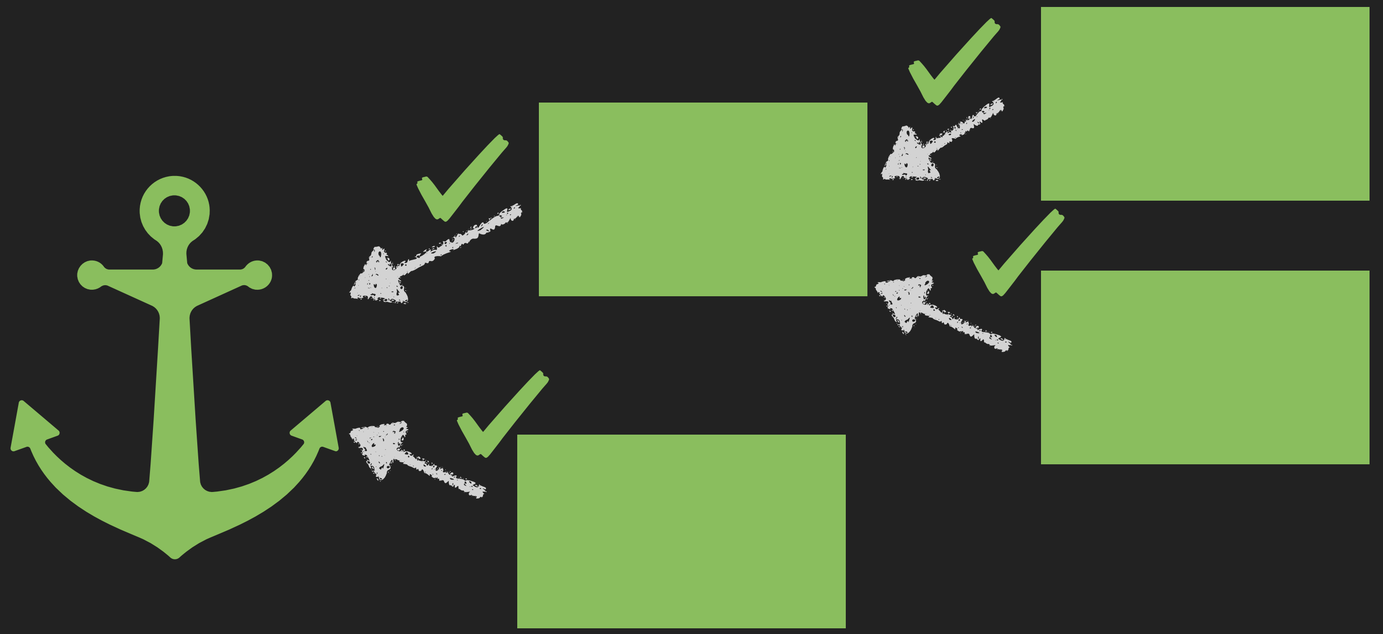
Solution: Define your trust anchors

Reasoning about the security of a system is impossible without

- knowing what you want to protect against (threat model)
- knowing what you can ultimately rely upon (trust anchor)

* *Threat modelling not discussed here but it is really important!*

TRUST ANCHORS



1. You trust your trust anchor(s) – they are secure by definition
2. You reason that something else is also trusted by relying on the trust anchor(s)
3. Therefore trust can be extended

TRUST ANCHORS: EXAMPLE

This is an example and might not meet your expectations

1. I trust my cloud provider
2. I trust that my cloud provider has a secure key storage
3. Data encrypted with keys stored in the keyring is secure
4. Therefore I can store sensitive data in my cloud

TRUST ANCHORS

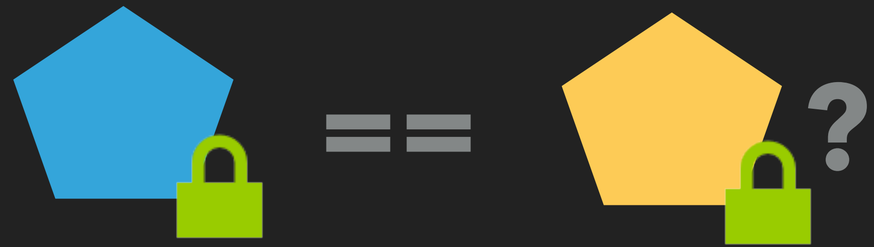
....

Microsoft Azure

....

any other ....

- http://time.com/collection/most-influential-people-2018/5217621/donald-trump-2/
- https://cloud.google.com/
- https://commons.wikimedia.org/wiki/File:Windows_Azure_logo.png
- https://www.bbc.com/news/25907502

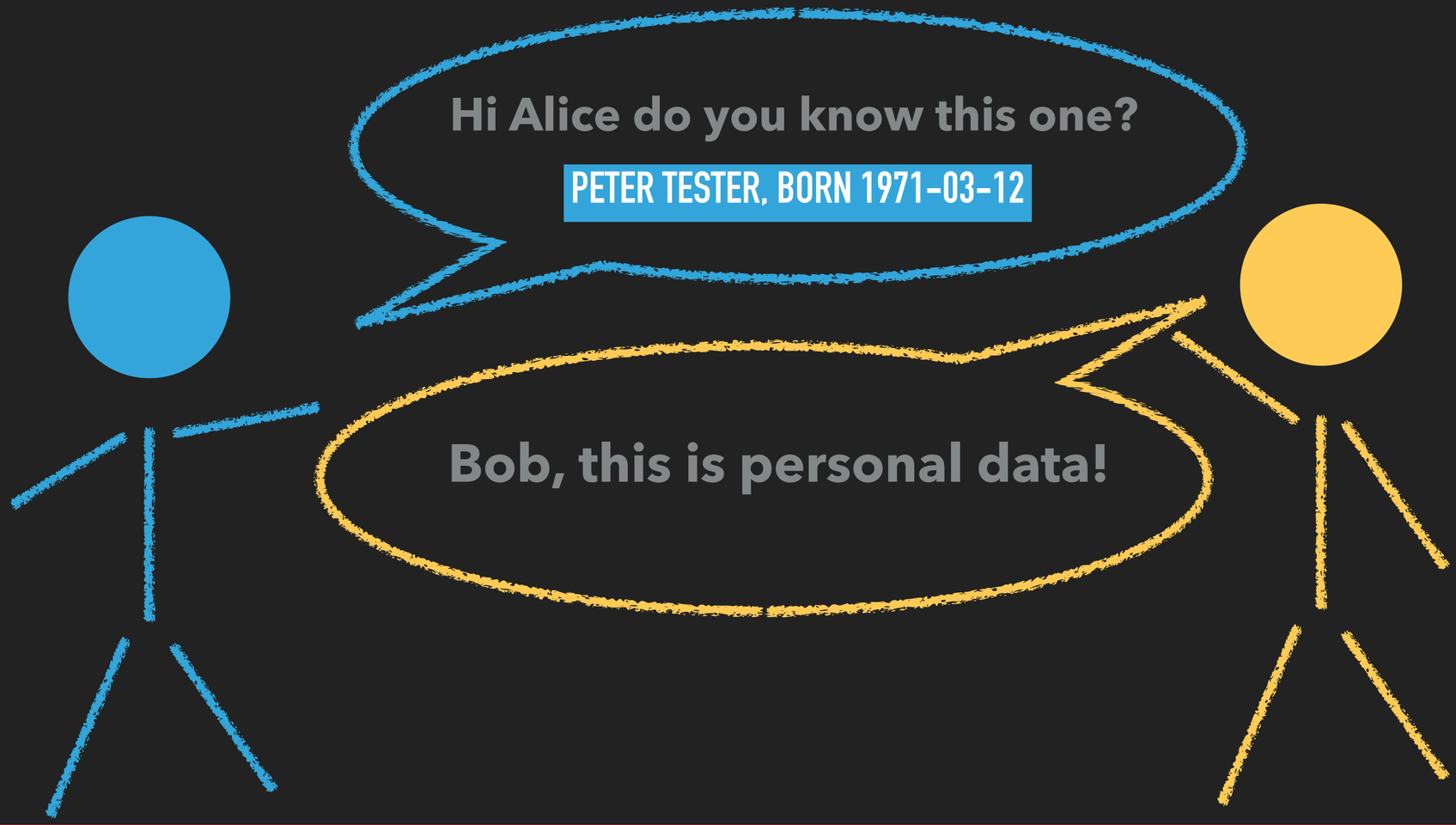


PATTERNS

COMPARING

COMPARE DATA

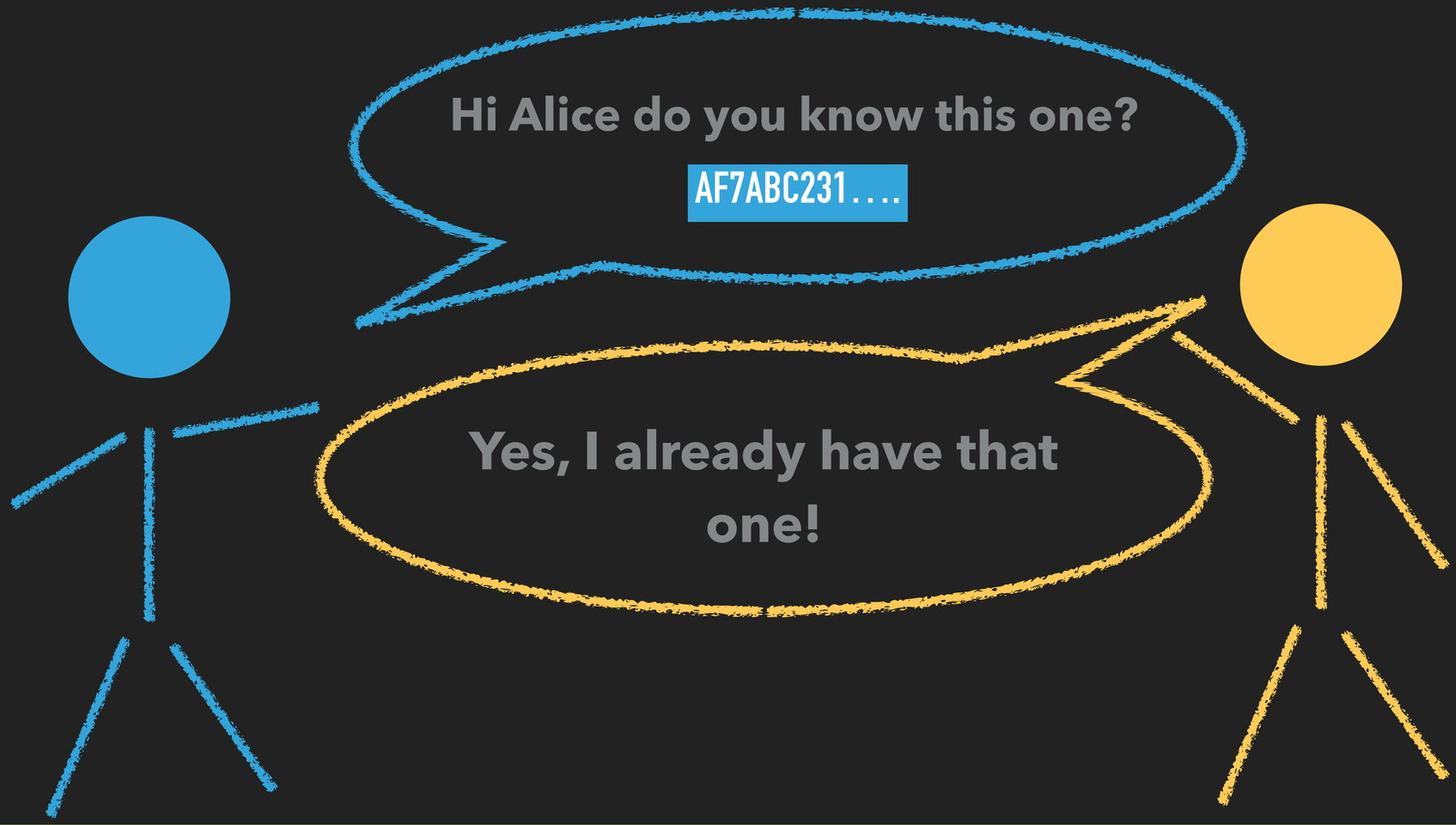
Problem: Securely compare two data items



CONFIDENTIAL DATA EXCHANGED!

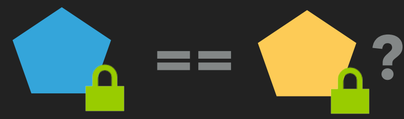
COMPARE DATA

Problem: Securely compare two data items



NO CONFIDENTIAL DATA EXCHANGED.

COMPARE DATA



Problem: Securely compare two data items

Solution: Normalise & hash data, compare hashes

LOREM IPSUM … == LOREM IPSUM …

=>★ sha256(LOREM IPSUM …) == sha256(LOREM IPSUM …)

<=> 4C53E9C9… == 4C53E9C9…

★ Collisions [**A != B** but **sha256(A) == sha256(B)**] are mathematically possible, but practically not relevant

COMPARE DATA

Problem: Securely compare two data items

Solution: Normalise & hash data, compare hashes

1 - Normalize

E.g. J. EDGAR HOOVER → HOOVER, JOHN EDGAR →* H160, J500 E326

2 - Hash

Use *hash(salt + data)* to prevent precomputing attacks. Use multiple iterations of hashing.

- *public salt* => treat hash as *pseudonymised*
- *secret salt*=> treat hash as *anonymised*

* *Soundex - but choose whatever normalisation works for you*

HASHING DOES NOT INCREASE THE ENTROPY! (THINK OF HASHED PHONE NUMBERS)

COMPARE DATA

Hashed personal data sometimes is no personal data!

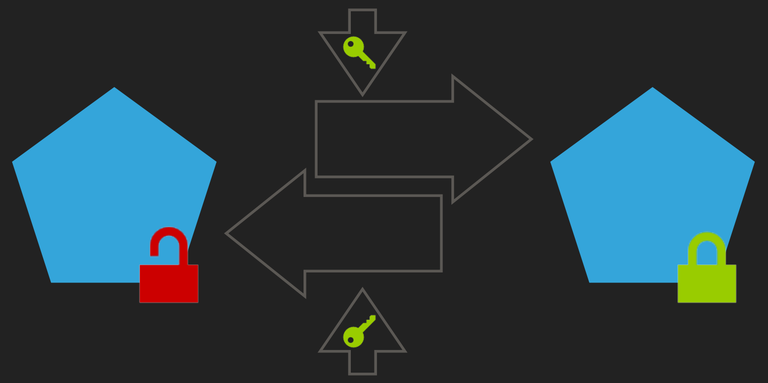
Dr. Grace Nacimiento von der Kanzlei KLEINER: „Der Einsatz von Salt-Wert und kryptographischer Hashfunktion durch Posteo sorgt dafür, dass eine vom Kunden eingegebene Mobilfunknummer anonymisiert wird, bevor eine Übermittlung an Posteo erfolgt. **Für die bei Posteo allein gespeicherten Hashwerte fehlt es daher an einem Personenbezug** […] Die gespeicherten Hashwerte sind daher kein Bestandsdatum im Sinne des § 95 TKG […]" ()

Bundesdatenschutzbeauftragte Andrea Voßhof:

„**Auch aus meiner Sicht ist der gespeicherte gesaltete Hashwert kein personenbeziehbares Datum**. (…) Zusammenfassend stelle ich fest, dass Posteo im Sinne des § 95 TKG keine Bestandsdaten erhebt".

https://posteo.de/bfdi_pruefbericht.pdf

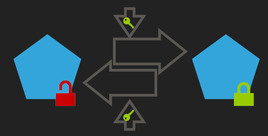
https://posteo.de/blog/bnetza-entscheidung-zu-posteo-kryptographisch-bearbeitete-daten-nicht-auskunftspflichtig



PATTERNS

STORING

SECURE MULTIPLE DATA RECORDS

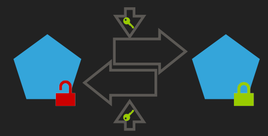


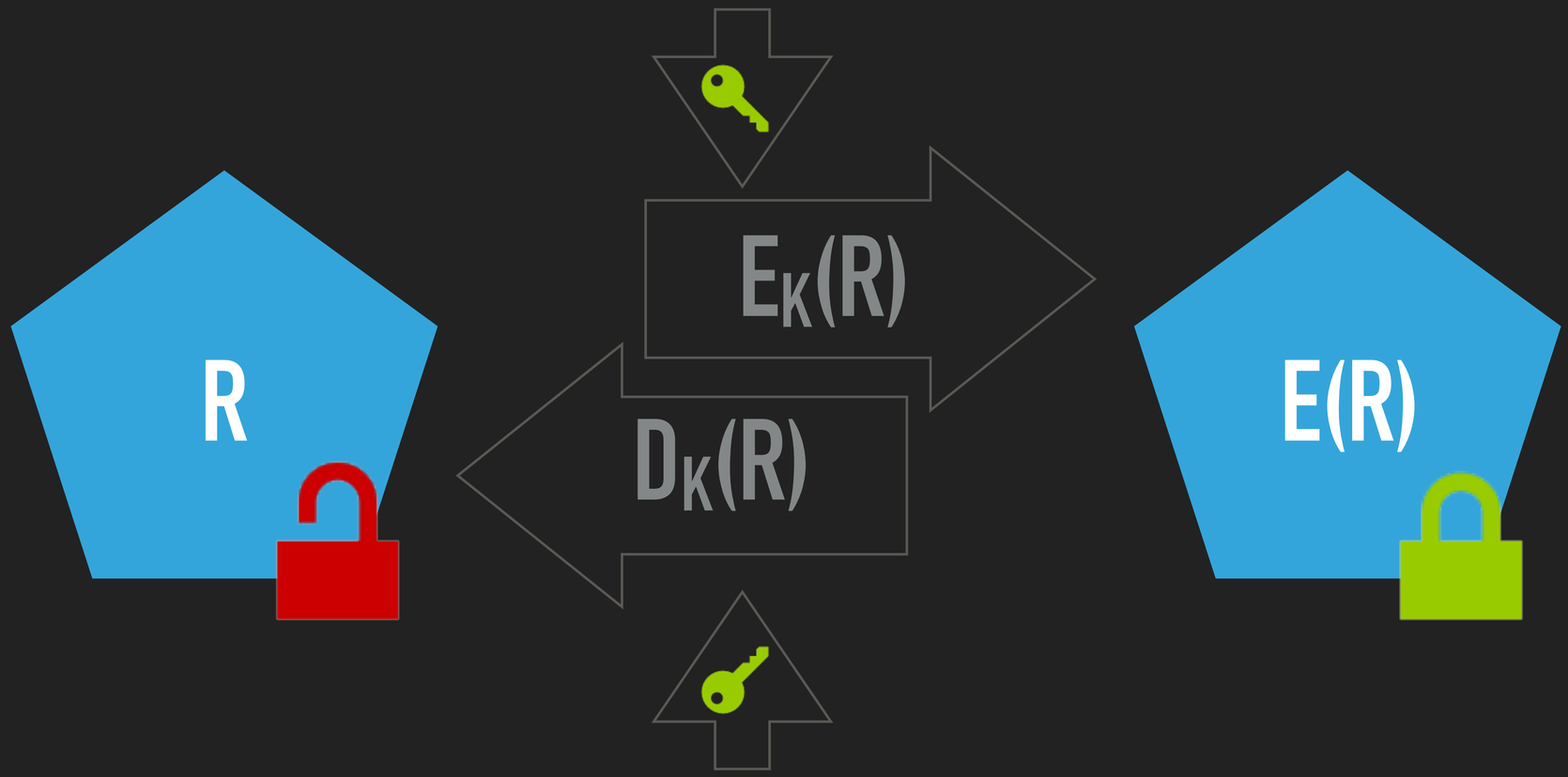
Problem: Multiple records are read and written by the same application.

Solution: Use symmetric encryption to protect the records.

Reading and writing plaintext record ‘R’ 

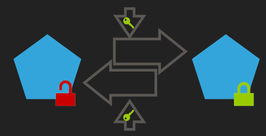
SECURE MULTIPLE DATA RECORDS

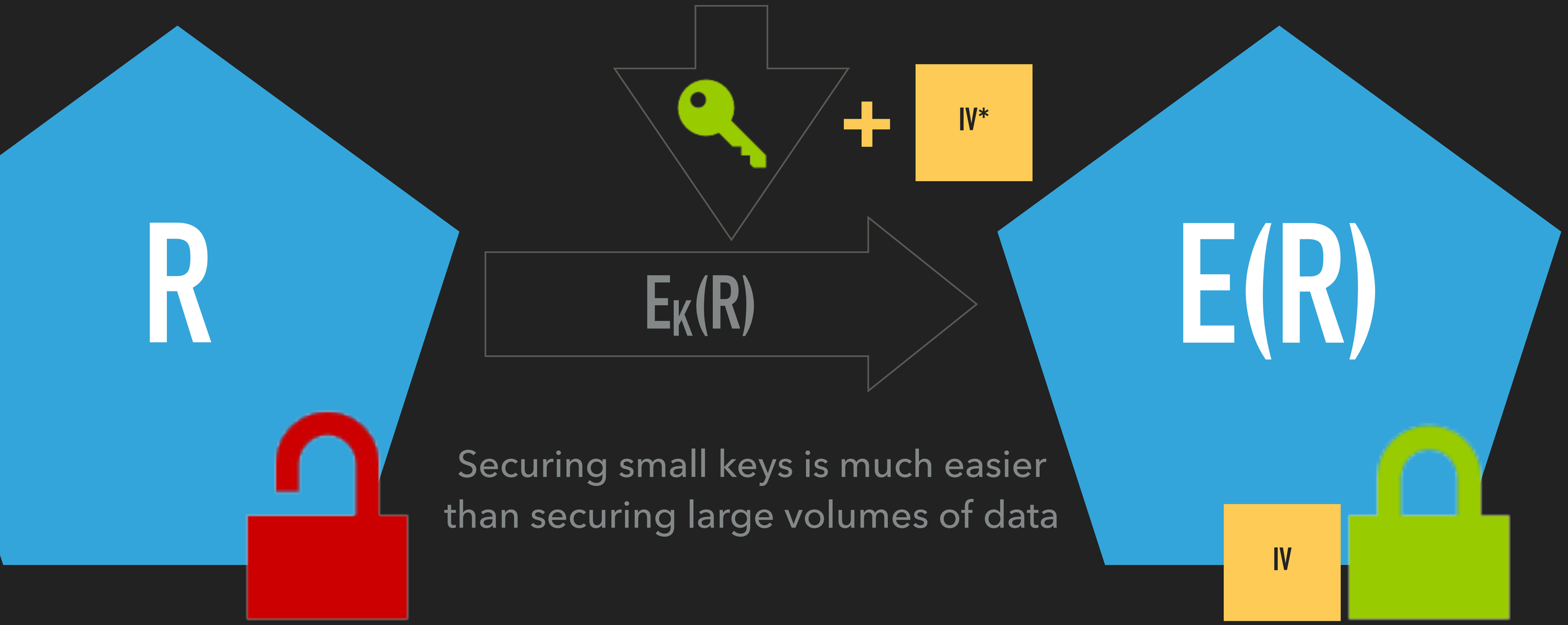




En-/decrypt Data ‘R’ (record) with key ‘K’.

SECURE SMALL KEY(S)





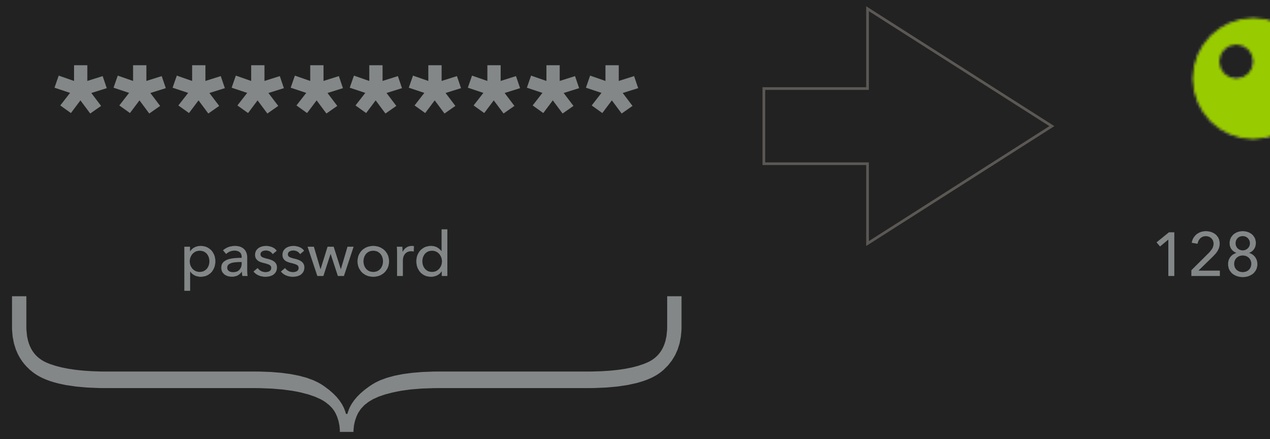
Securing small keys is much easier than securing large volumes of data

(*) IV - Initialisation Vector. Used in many cryptographic operations. Like a salt. Must be random, might be public.

PATTERNS

KEY DERIVATION 1: PASSWORD TO KEY

FROM PASSWORD TO KEY

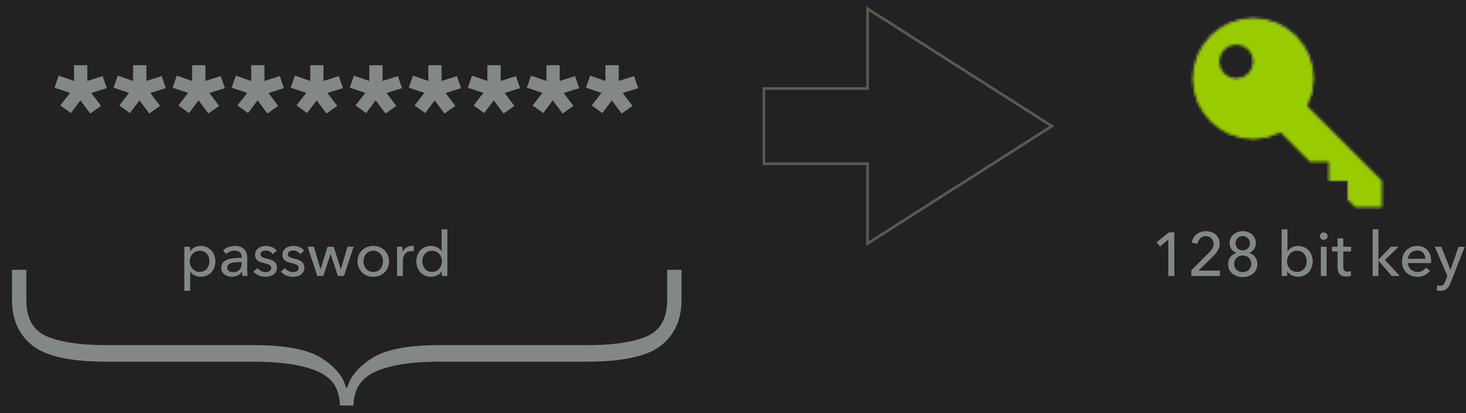


- A *random* password has
  ~ 6 bits per character (*)
- A 128 bit key needs >= 21 character passwords

Secure passwords are *very* long

(*) a-zA-Z0-9 –> ~64 different values per character (2^6). A 128 bit passwords must be >= 21 characters long (128/6)

FROM PASSWORD TO KEY



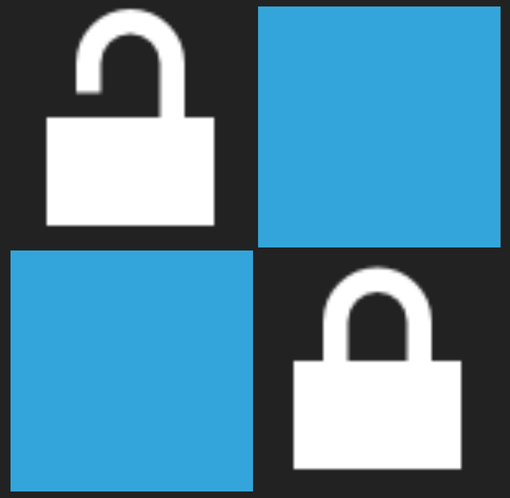
- `aw92SDAVglkqusabvgw38`
  - 128 bit
- `3o8uGsdA`
  - 8 chars, 48 bit (cracked in hours to days)

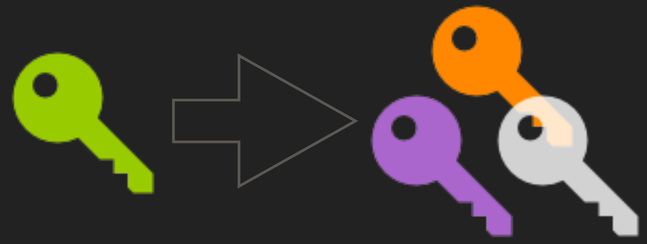
CREATE YOU PERSONAL PASSWORDS USING A PASSWORD MANAGER!

FROM PASSWORD TO KEY

- Key derivation functions (KDF) convert passwords to keys
- For good (21+ chars) passwords use HKDF (RFC5869)
- Else: use a KDF with brute force protection (*)
  - SCRYPT (RFC7914)
  - PBKDF2 (RFC2898)

(*) *Brute force protection*: The function is designed to be very slow (up to seconds). This prevents enumeration attacks.

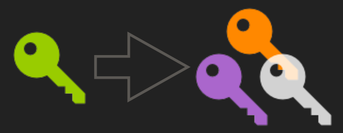




PATTERNS

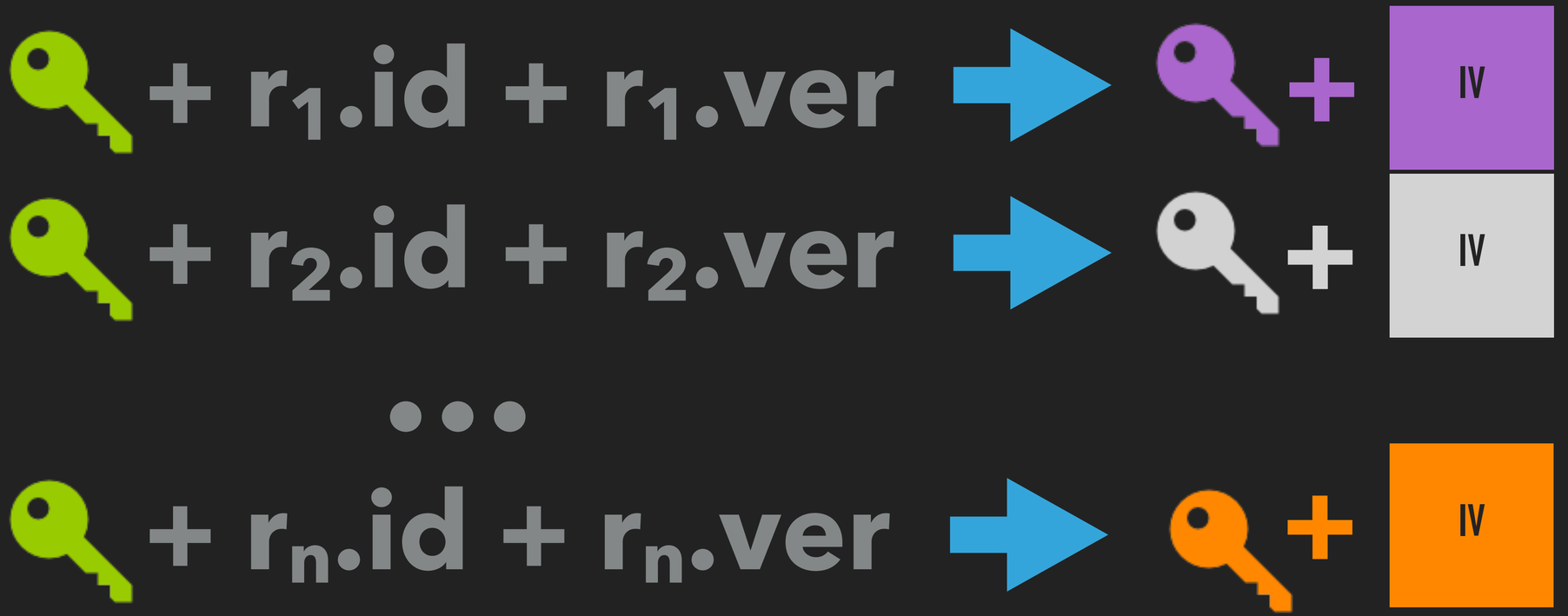
KEY DERIVATION 2: FROM 1 TO N

DERIVE PER RECORD KEYS



Problem: Use different keys for different records, only store master key.

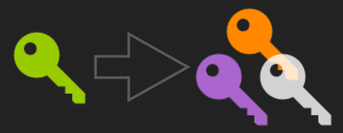
Solution: Use key derivation to derive per-record keys.



IMPORTANT: NEVER USE THE SAME KEY/IV TO ENCRYPT DIFFERENT DATA

MAKE SURE THAT THE MASTER KEY HAS ENOUGH ENTROPY FOR DERIVED KEY AND DERIVED IV

SOLUTIONS FOR DERIVING KEY(S)



```
// Input:
//  Master_key and
//  (DB) record_id target record DB id
// Output:
//  AES-Key and
//  salt for encrypting target record

// AES-Key and salt for target record. "||" concatenates
// AES-CBC uses 128 bit IV. AES-GCM uses a 96 bit IV
byte[32] keyAndIV = derive_key( master_key ||
                                record_id || record_version, 256 bit)

byte[16] derived_iv   = keyAndIV[0..15]
byte[16] derived_key  = keyAndIV[16..31]
```

- derive_key needs an additional *installation specific* salt of >= 128 bit. PBKDF2 with HMAC sha256 is an example of derive_key, as is scrypt or argon2.
- Use same process for decryption.
- No need to store the *generated* IV value.

IMPORTANT: NEVER USE THE SAME KEY/IV TO ENCRYPT DIFFERENT DATA

MAKE SURE THAT THE MASTER KEY HAS ENOUGH ENTROPY FOR DERIVED KEY AND DERIVED SALT

PATTERNS

PASSWORD VERIFICATION

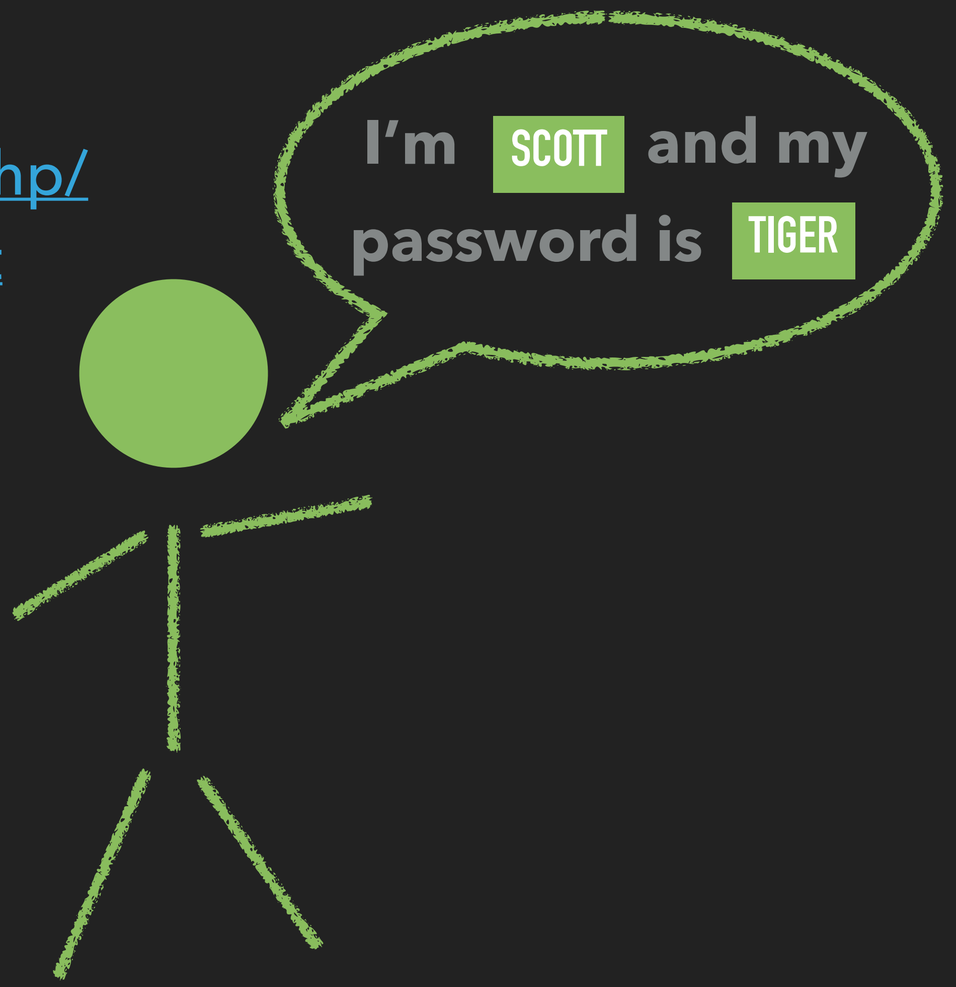
USER LOGIN

Problem: Login a user

Solution: Not in scope here

https://www.owasp.org/index.php/Password_Storage_Cheat_Sheet

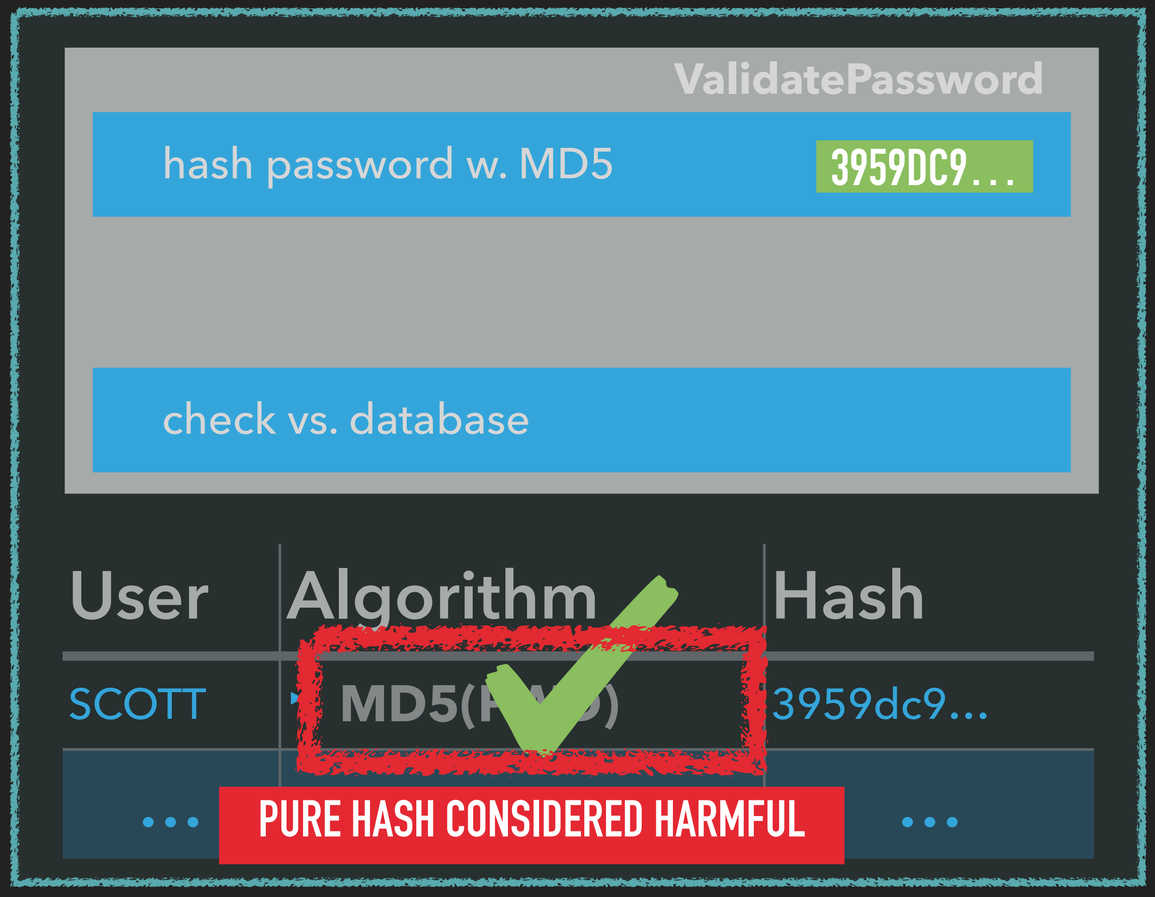
Also: OAUTH, Kerberos, …



USER LOGIN: MIGRATE PASSWORDS

Problem: Migrate password hashes to new algorithms

Solution: Chain hashing functions



Even consumer grade graphic cards calculates giga-hashes (2^30) per second.

HASHES – EVEN SALTED – OFFER NO PROTECTION AGAINST OFFLINE ATTACKS

SWITCH TO BRUTE-FORCE PROOF (== SLOWER) ALGORITHMS

INCREASE THE ENTROPY BY USING A PEPPER

- https://www.troyhunt.com/our-password-hashing-has-no-clothes/
- https://gist.github.com/epixoip/a83d38f412b4737e99bbef804a270c40
- http://cynosureprime.blogspot.de/2017/08/320-million-hashes-exposed.html
- https://www.owasp.org/index.php/Password_Storage_Cheat_Sheet
- https://en.wikipedia.org/wiki/Pepper_(cryptography)

USER LOGIN: MIGRATE PASSWORDS

- Online migration of passwords to a new hashing scheme

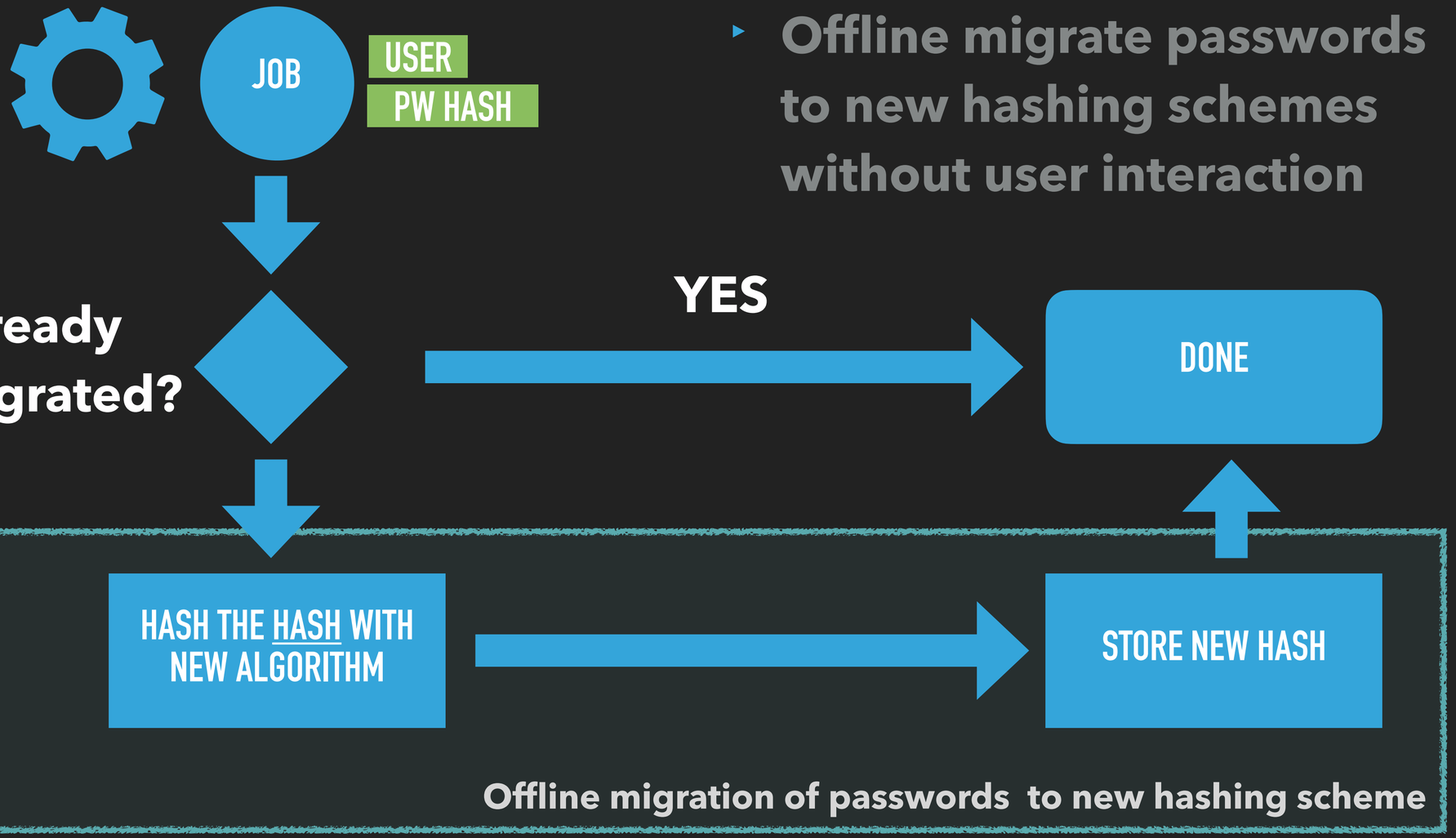
USER LOGIN: MIGRATE PASSWORDS

User	Algorithm	Hash	Last Login
SCOTT	PBKDF2(PWD)	3959dc9…	Now
PETER	MD5(PWD)	…	2 years ago
…	MD5(PWD)	…	4 months ago
…	MD5(PWD)	…	…
…	MD5(PWD)	…	…
…	MD5(PWD)	…	…

- How to migrate passwords of users that do not login frequently?

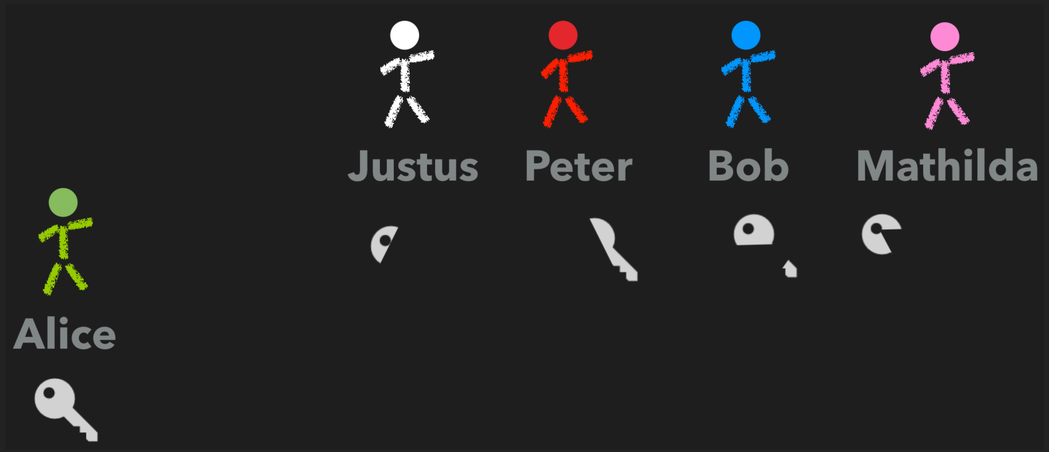
USER LOGIN: MIGRATE PASSWORDS

- Offline migrate passwords to new hashing schemes without user interaction



USER LOGIN: MIGRATE PASSWORDS

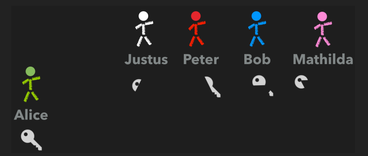
User	Algorithm	Hash	Last Login
SCOTT	PBKDF2(PWD)	3959dc9…	Now
PETER	PBKDF2(MD5(PWD))	…	2 years ago
…	PBKDF2(MD5(PWD))	…	4 months ago
…	PBKDF2(MD5(PWD))	…	…
…	PBKDF2(MD5(PWD))	…	…
…	PBKDF2(MD5(PWD))	…	…



PATTERNS

SECRET SHARING

SECRET SHARING



Problem: What happens when Alice forgets her password?

Solution: Use cryptographic secret sharing for recovery

Justus Peter Bob Mathilda

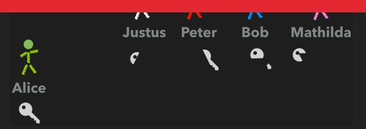
Alice

Alice trusts her friends only so far.
But she thinks it is very unlikely that three of them conspire together against her.

Alice will split her secret key (e.g. with Shamir) in such a way, that any three of her four trusted friends can restore the key.

DO NOT TRY THIS WITHOUT A CRYPTOGRAPHER

SECRET SHARING



Secret sharing (“t out of n”) shares a secret such, that the secret can be restored with any t (here 3) of the n (here 4) parts.

This can be used for secret recovery without a single point of trust (failure).

DO NOT TRY THIS WITHOUT A CRYPTOGRAPHER

DES BLOWFISH AES

MD5 SHA-1 SHA-256

RSA-1024 RSA-2048 ?? POST QUANTUM ??

PATTERNS

ALGORITHM ROLLOVER

ALGORITHM ROLLOVER

Problem: Algorithms must be changed and data migrated

Solution: Design for online data migration

Record-ID	…	Masterkey ID	(Data…)
B9E10DEE-C97E-…	…	B874920B-E801-…	…
FDE0C6E3-8BF0-…	…	9A6580FC-1248-…	…
…	…	9A6580FC-1248-…	…

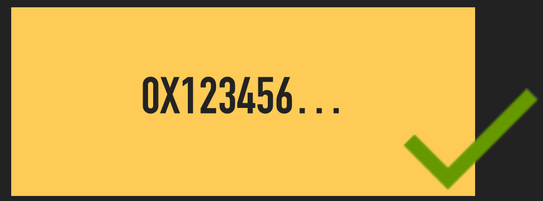
ALGORITHM ROLLOVER

Problem: Algorithms must be changed and data migrated

Solution: Design for online data migration

Record-ID	Algorithms	Masterkey ID	(Data…)
B9E10DEE-C97E-…	▸ PBKDF2(…) ▸ AES128–GCM	B874920B-E801-…	…
FDE0C6E3-8BF0-…	▸ SCRYPT(…) ▸ AES256–CBC ▸ PKCS#5	9A6580FC-1248-…	…
		9A6580FC-1248-…	

ATTENTION: REENCRYPTING OPENS A WINDOW OF ATTACK — BEST USE THE NEW ALGORITHM FOR NEW DATA!



PATTERNS

INTEGRITY

DATA INTEGRITY & ASSOCIATION

0X123456...

Problem: Sensitive data can be manipulated.

User	Salary	...
 Alice	3,141€	...
 Eve	2,718 €	...
...		...

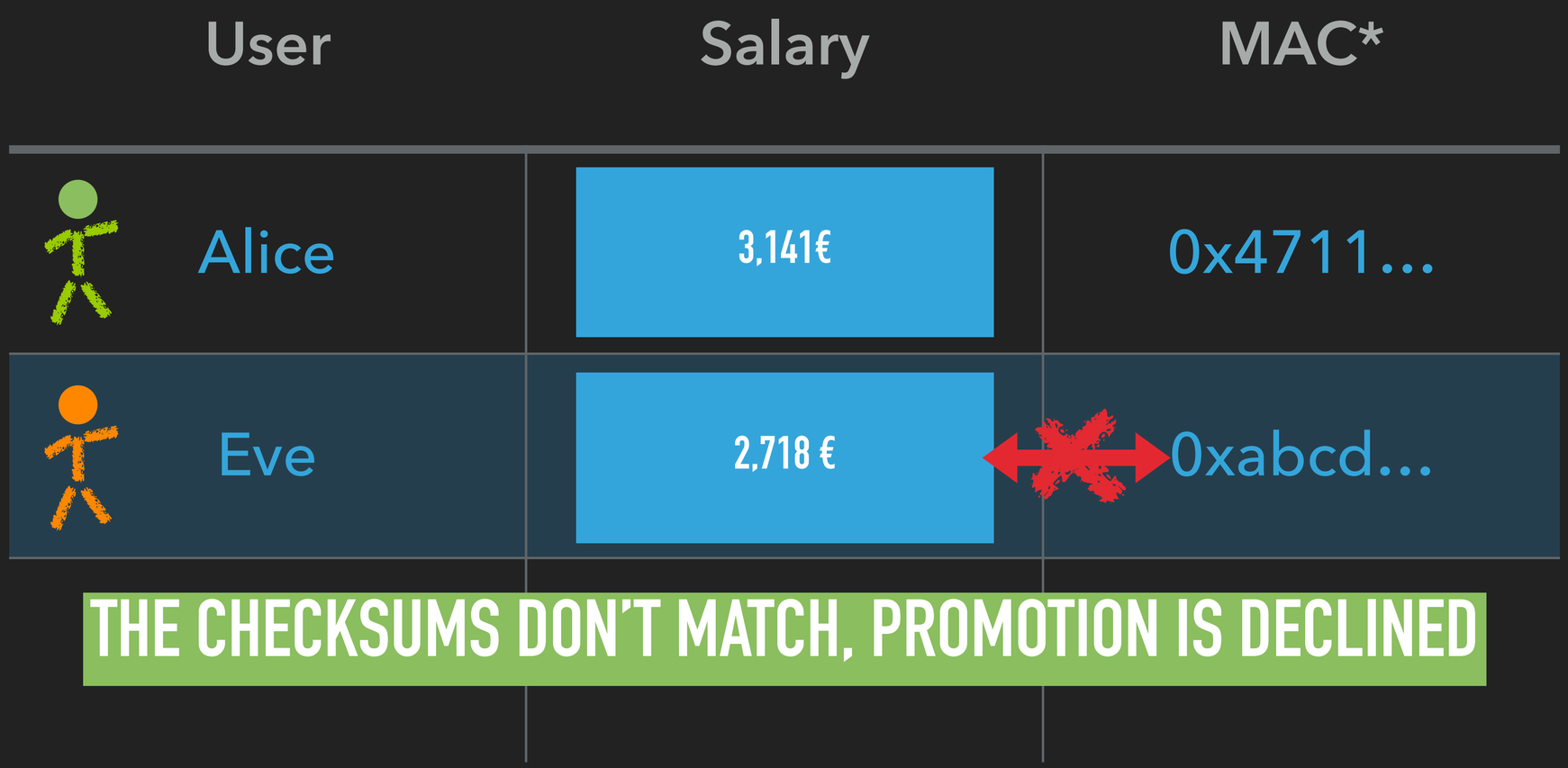
EVE GETS AN INSTANT PROMOTION

DATA INTEGRITY & ASSOCIATION

0X123456…

Problem: Sensitive data can be manipulated.

Solution: Use cryptographic checksums with a secret.



* Checksum with a secret: hmac, AEAD, public key signatures

DATA INTEGRITY & ASSOCIATION

0X123456…

Problem: Protected data can be “replayed”.

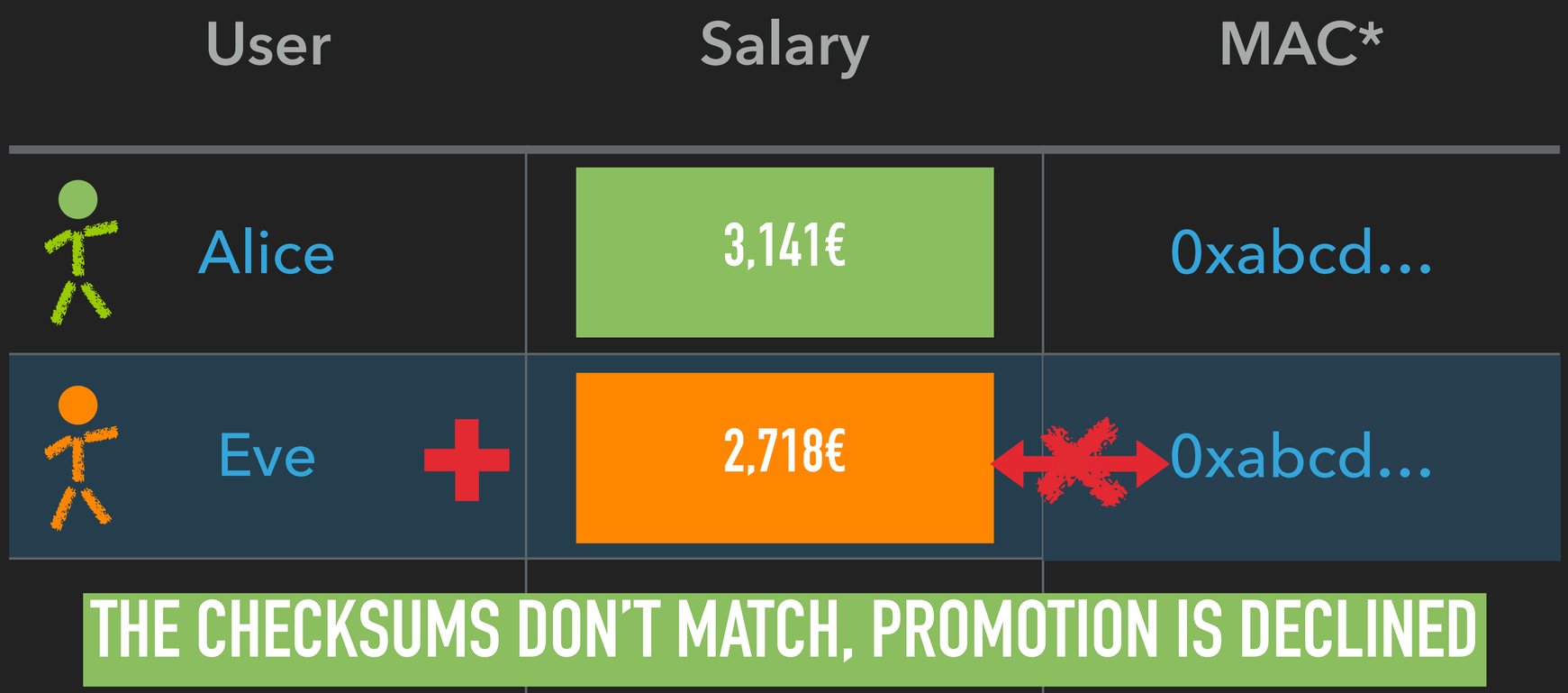
User	Salary	MAC*
 Alice	3,141€	0x4711…
 Eve	2,718 €	0x4711…
…	EVE GETS AN INSTANT PROMOTION	…

DATA INTEGRITY & ASSOCIATION

0X123456…

Problem: Protected data can be “replayed”.

Solution: Cryptographically bind data to context.



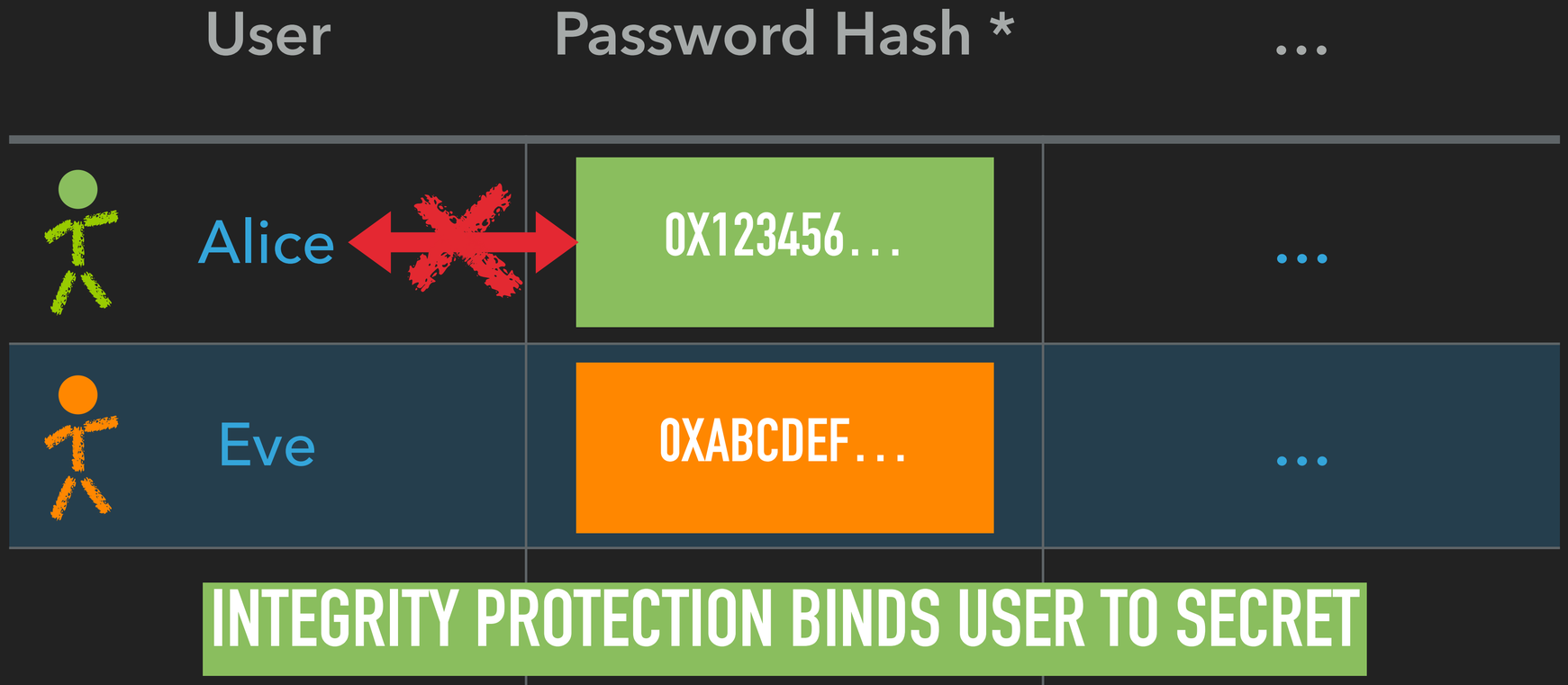
* including the username in the hash/hmac

DATA INTEGRITY & ASSOCIATION

0X123456…

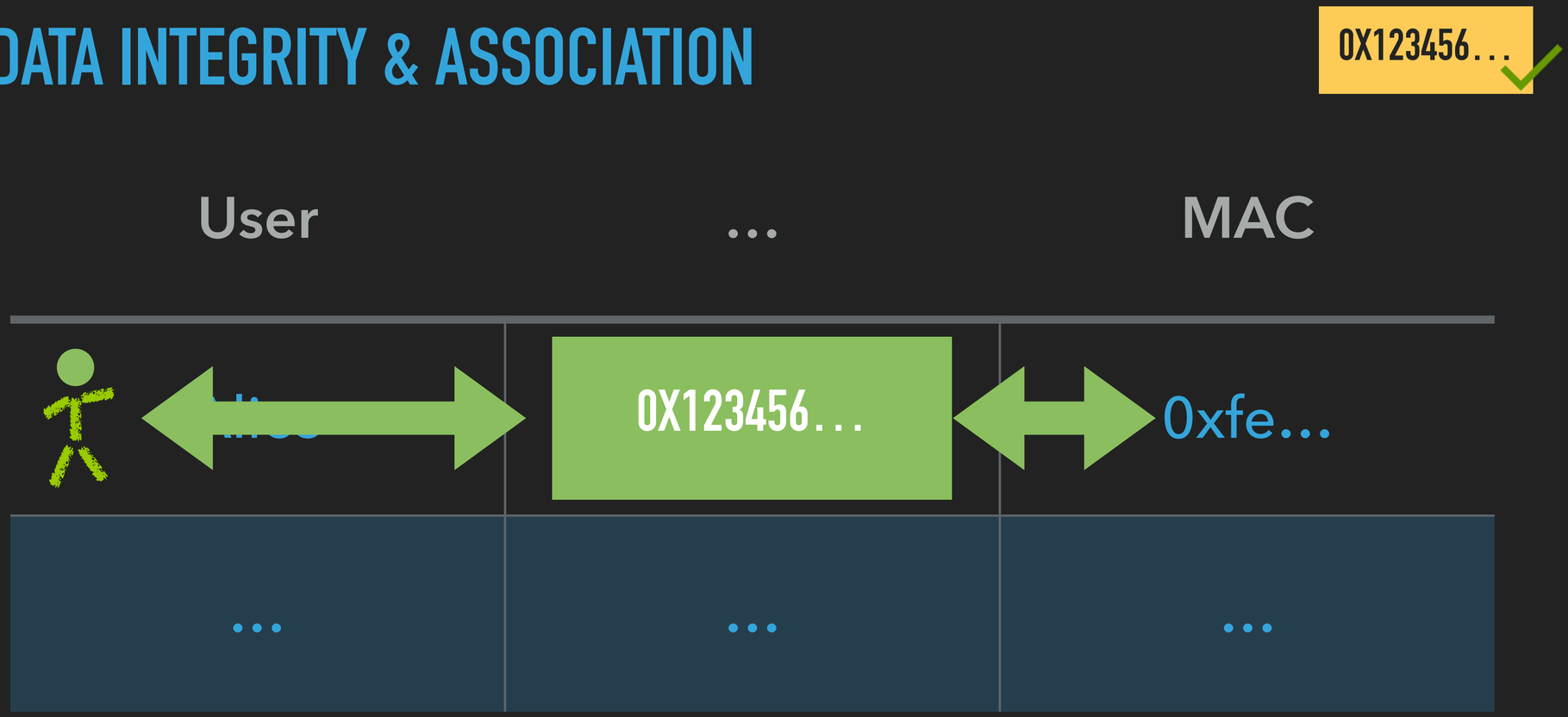
Problem: Protected data can be “replayed”.

Solution: Cryptographically bind data to context.



* including the username in the hash/hmac

DATA INTEGRITY & ASSOCIATION



- Add integrity checks to the data (HMAC, AEAD encryption, signatures)
- Include an association (here: “User”) in the integrity check

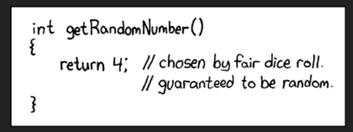
```
int getRandomNumber()
{
    return 4;  // chosen by fair dice roll.
               // guaranteed to be random.
}
```

https://xkcd.com/221/

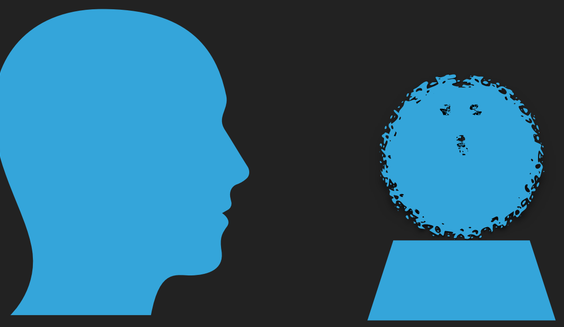
PATTERNS

ENTROPY

ENTROPY



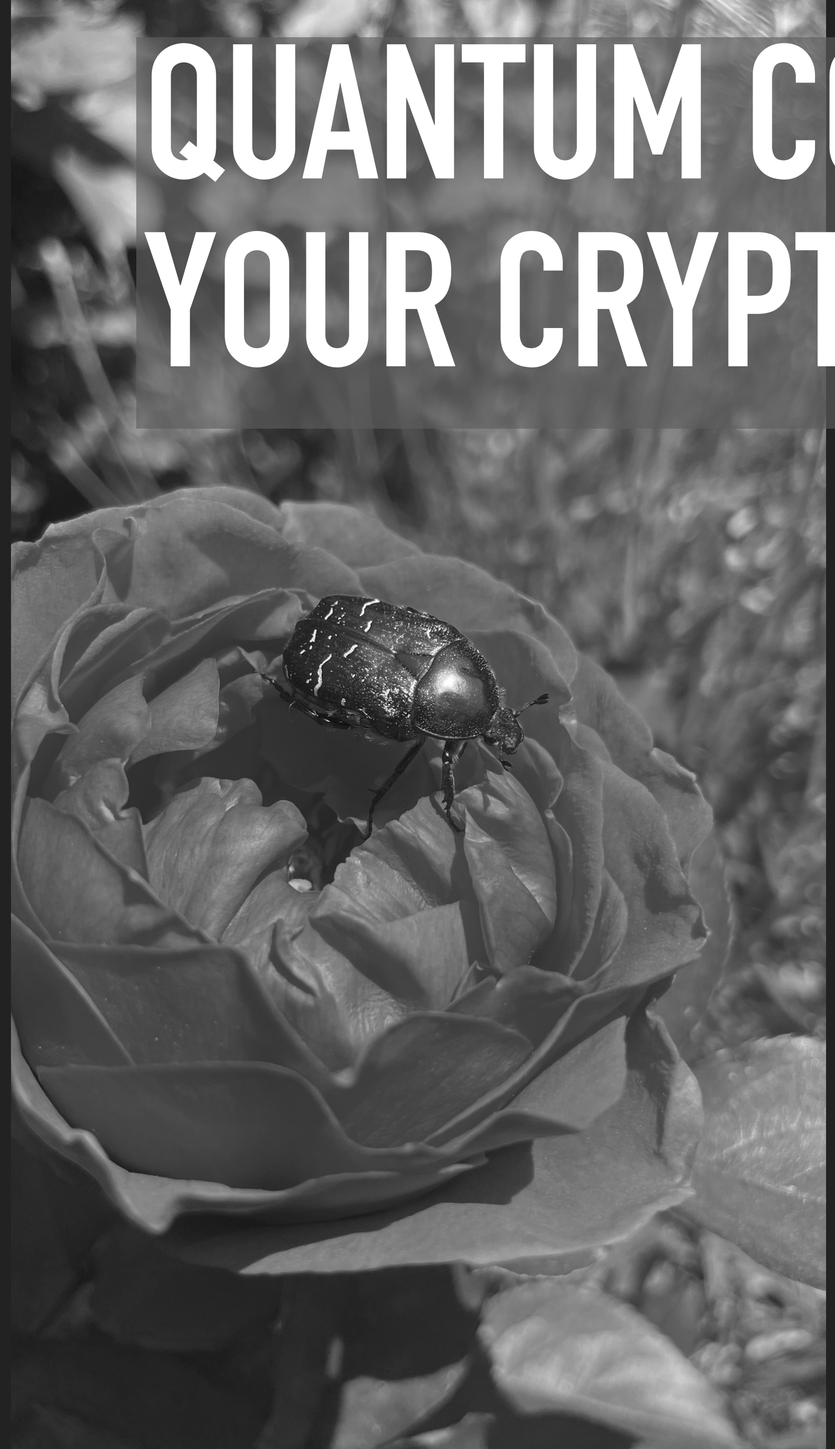
- Bad entropy compromises keys
- Computers are very bad at making things up! (not always)
- Entropy therefore often is limited (esp. after booting!)
- Use what the API provides (SecureRandom)
- **RTFM**



THE FUTURE

POST QUANTUM

QUANTUM COMPUTERS WILL BREAK YOUR CRYPTO (BE PREPARED)



Quantum computers are *very* good at solving puzzles

Cryptography is *all* about solving puzzles

	Security now	Quantum
AES_128	128	64 bit ⛔
AES_256	256	128 bit ✅
RSA 2048/3072	112/128	Trivial ⛔

Key length equivalent for RSA: https://en.wikipedia.org/wiki/Key_size

HOW TO SOLVE THE POST QUANTUM PROBLEM?

- Post quantum will come - likely in the next 5-15 years. Or much earlier (see link below)
- AES256 and other symmetric algorithms likely still secure (but key length greatly reduced: bit length/2)
- We have **no** quantum safe asymmetric algorithms
- **Solution**: Crypto Agility! Design everything in ways that allow algorithms to be replaced (as shown throughout the slides)

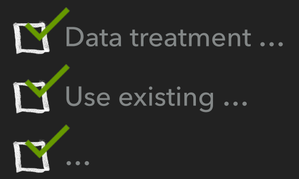
https://www.schneier.com/blog/archives/2023/01/breaking-rsa-with-a-quantum-computer.html

- [x] Data treatment …
- [x] Use existing …
- [x] …

PATTERNS

CRYPTO CHECKLIST

CRYPTO CHECKLIST



- [ ] Data treatment plan created and consequences accepted by management
- [ ] Trust anchors identified and named
- [ ] Sensitive operations (crypt,sign,..) require client authentication (applies to services too!)
- [ ] Existing (e.g. RFC 4880) protocols & formats used wherever possible
- [ ] Nonces used only once. Random salt used where possible
- [ ] Cryptographic concept written down & challenged in review
- [ ] (Master-)Key offsite backup established
- [ ] Key refresh after a few GiB of encrypted data implemented and tested
- [ ] Algorithm rollover implemented and tested
- [ ] Entropy source with enough entropy used
- [ ] Test cases include restore of old data (key/algorithm rollover)

SUMMARY

Regulations apply - whatever you do!

Encryption is not for free!

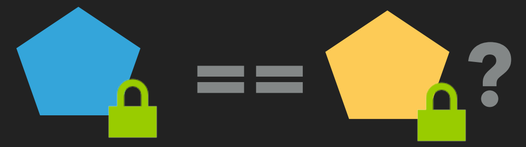
No encryption might be way more expensive!

Encryption is a safety net (*last* line of defence)

—> Assess risks & cost, plan, implement!

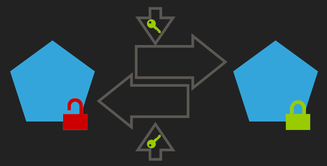
SUMMARY

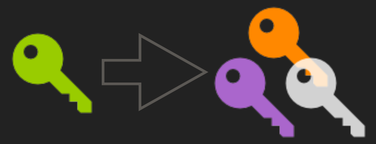
Comparing data



Transparent encryption

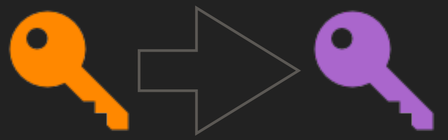
Storing data





Key derivation

Key refresh



Algorithm rollover

SUMMARY

Integrity

Moving Data

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

Entropy

Access Control

Secret Sharing

Crypto Checklist

- [x] Data treatment ...
- [x] Use existing ...
- [x] ...

Q & A

https://github.com/neuhalje/presentation_content-encryption

Something missing?

Boring?

Awesome?

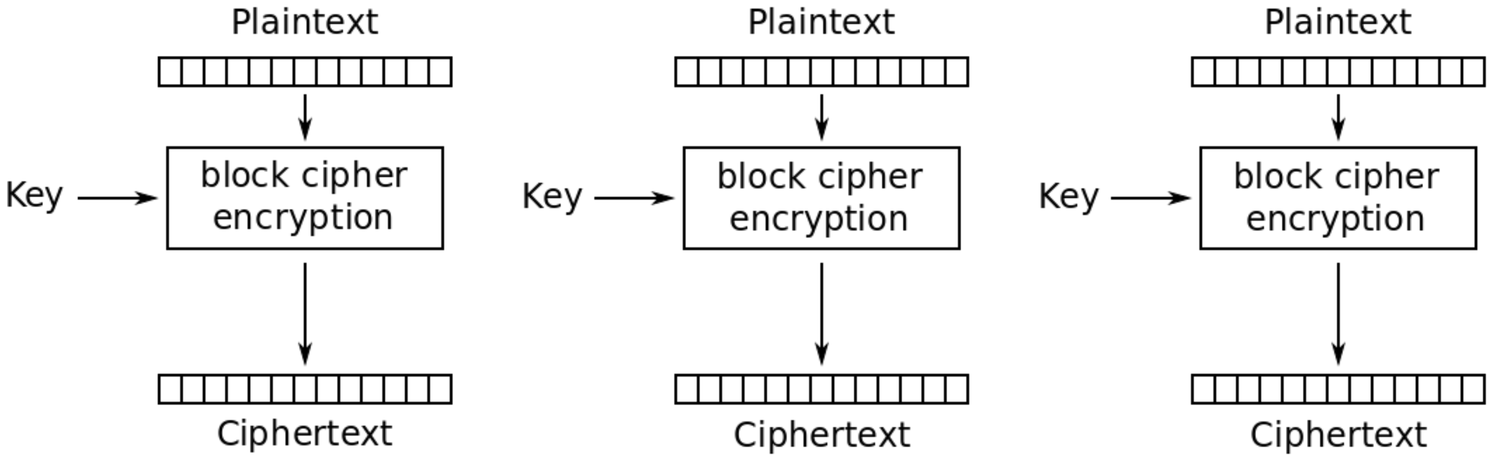
Feedback helps!

FEEDBACK

https://github.com/neuhalje/presentation_content-encryption

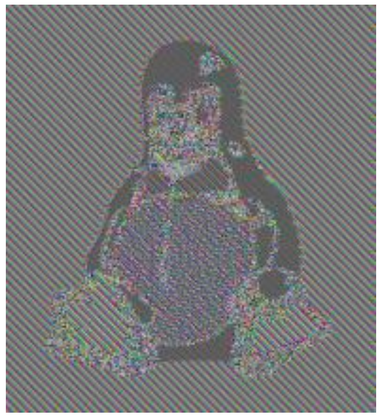
BACKUP

BLOCK CIPHERS AND THE IV

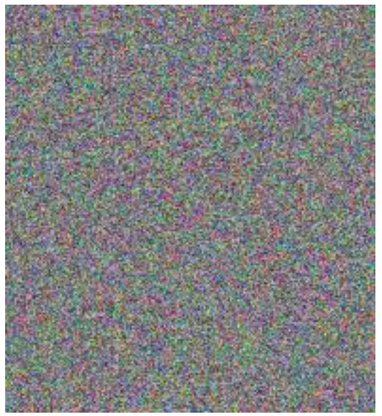


Electronic Codebook (ECB) mode encryption

Original



ECB



CBC (or other)

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

PATTERNS

ACCESS CONTROL

ACCESS CONTROL

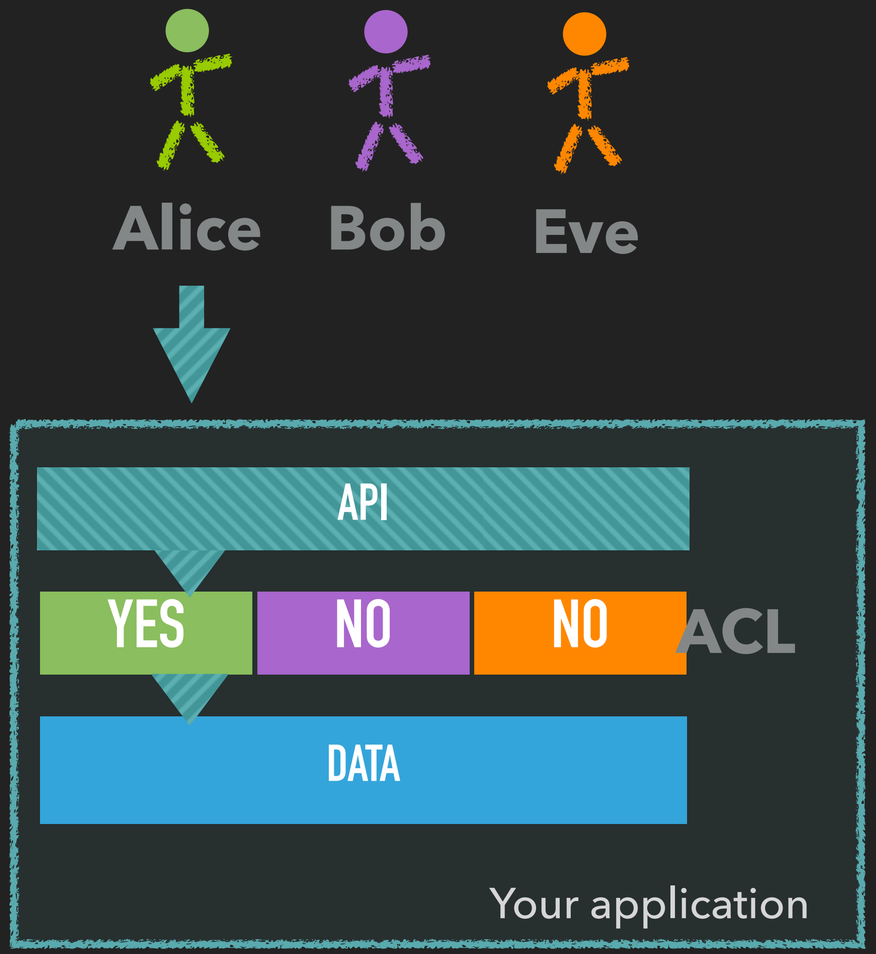
Problem: Make sure that data can only be accessed by some users

Solution: Use cryptographic access controls

Applies primarily to

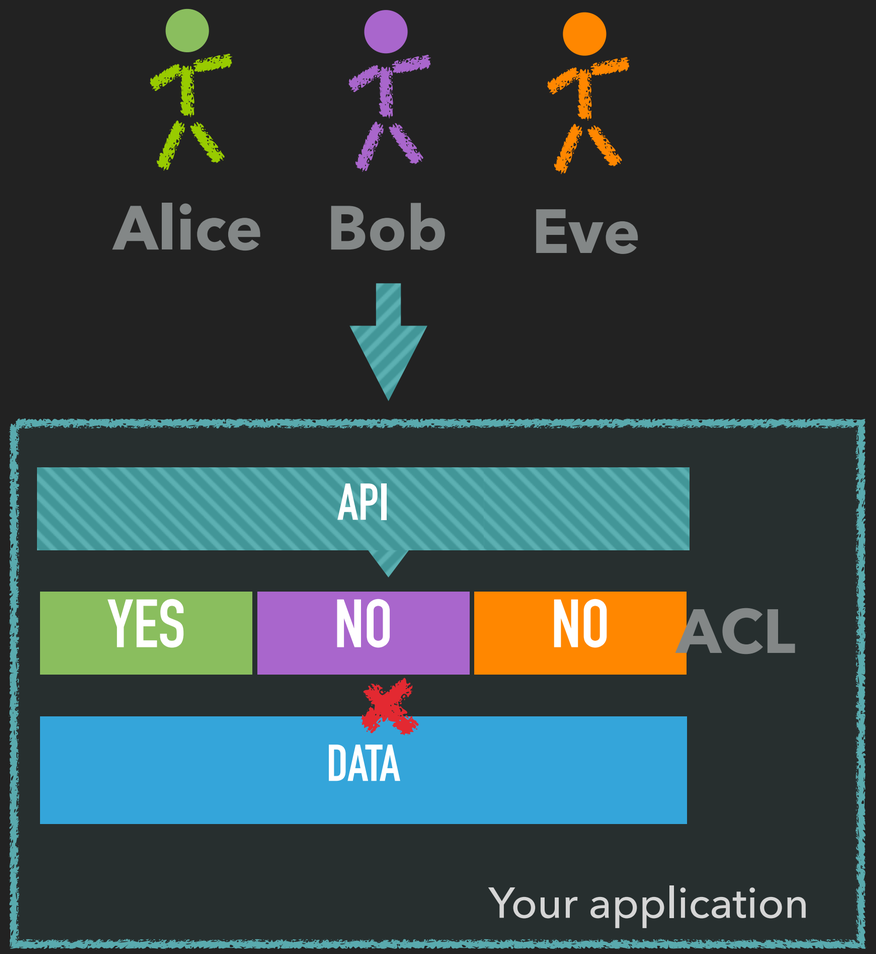
- Personal data (e.g. health data, personell records, …)
- Top Secret data (e.g. company secrets)
- When “provable” access control is required

ACCESS CONTROL



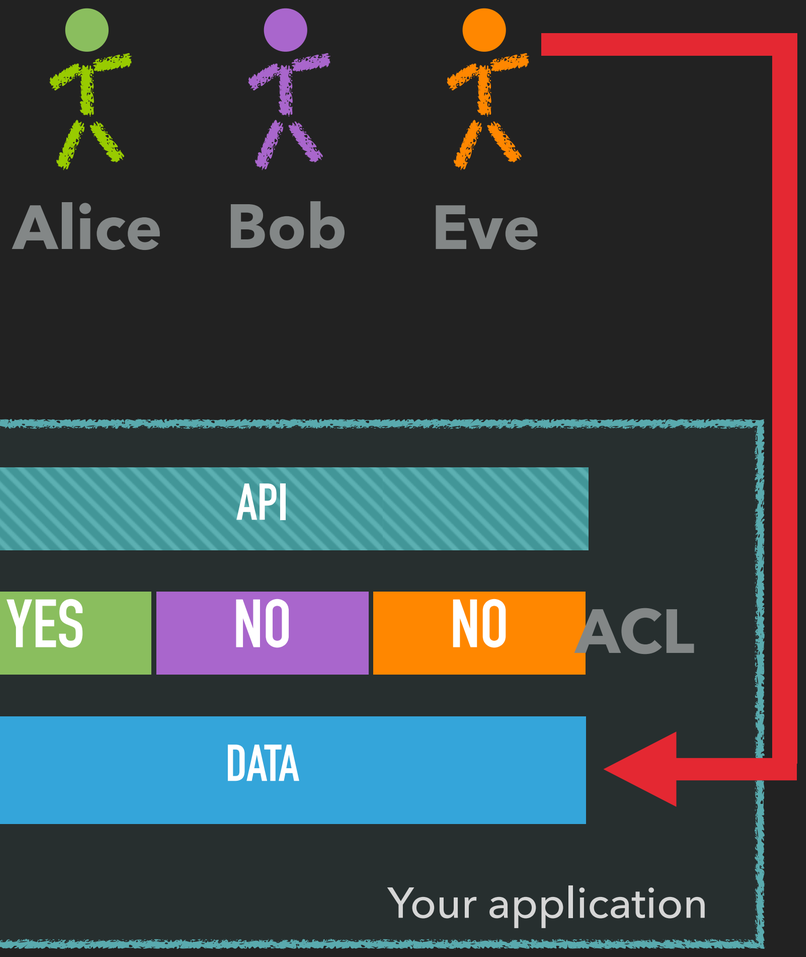
1. Alice tries to read data
2. Application validates ACL
3. Alice is granted access

ACCESS CONTROL



1. Bob tries to read data
2. Application validates ACL
3. Access is denied

ACCESS CONTROL

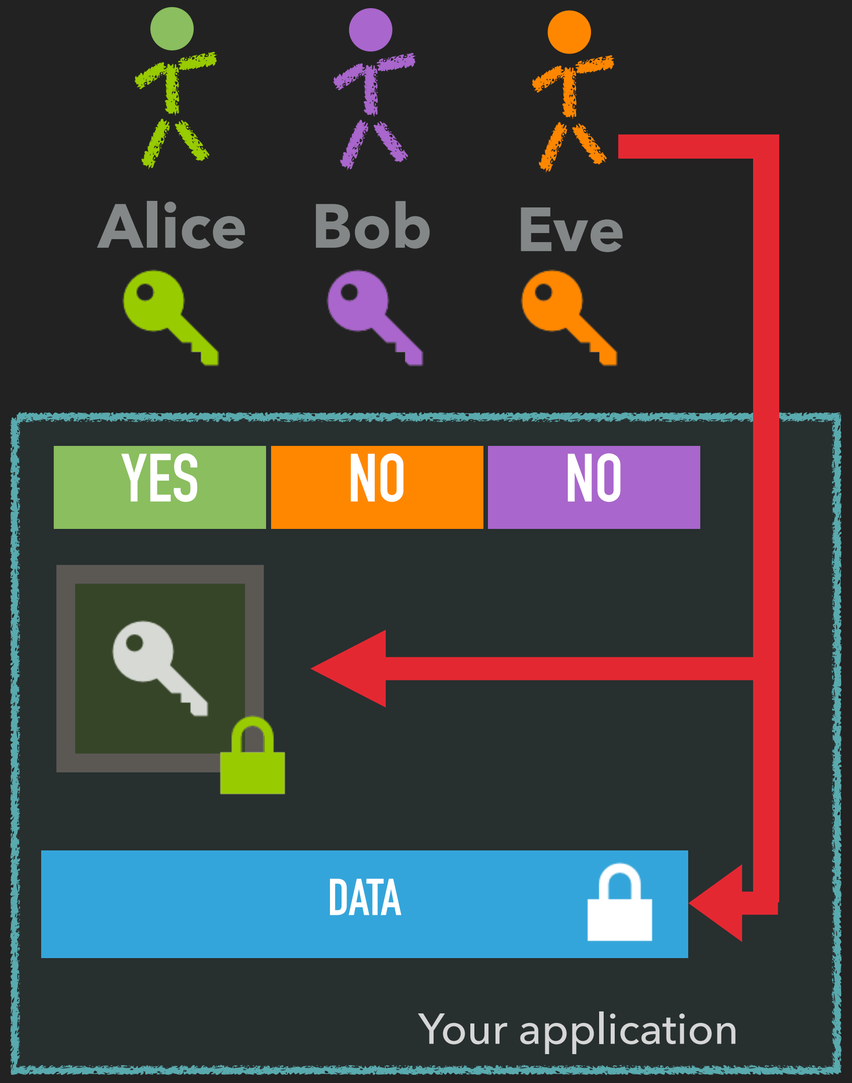


1. Eve exploits application
2. Application ????
3. Eve has access to data

ACCESS CONTROL

1. Alice tries to read data
2. Application validates ACL
3. Alice is granted access
4. Alice decrypts the “data key”
5. Alice decrypts data

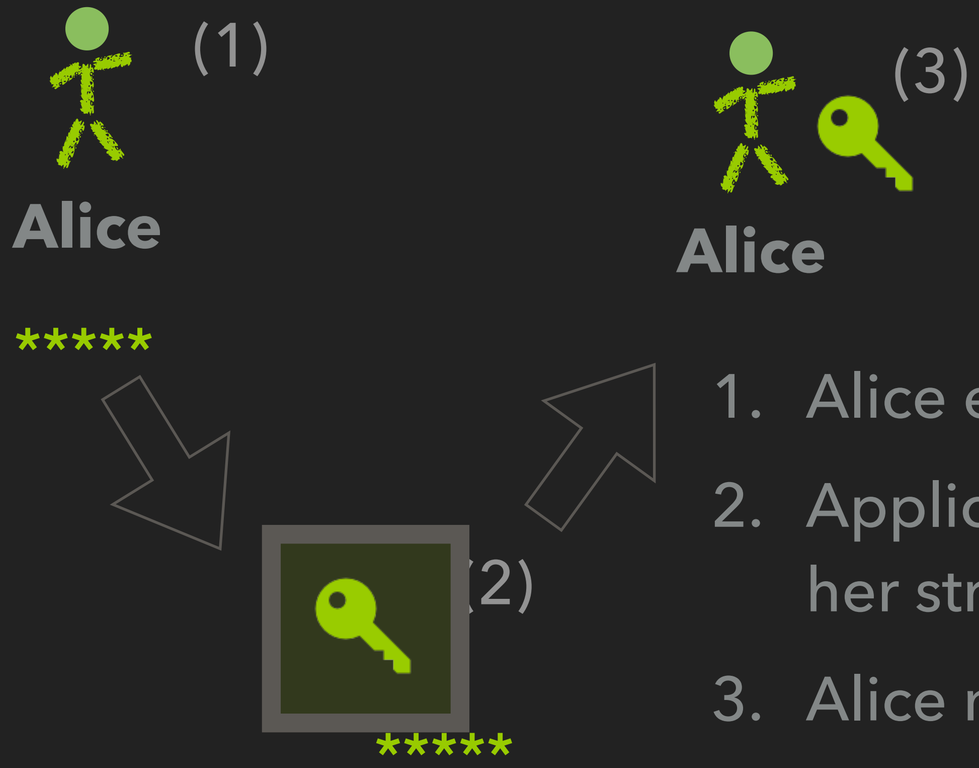
ACCESS CONTROL



1. Eve exploits application
2. Eve gets encrypted data key
   AND
   Eve gets encrypted data

Eve cannot decrypt the data!

ACCESS CONTROL

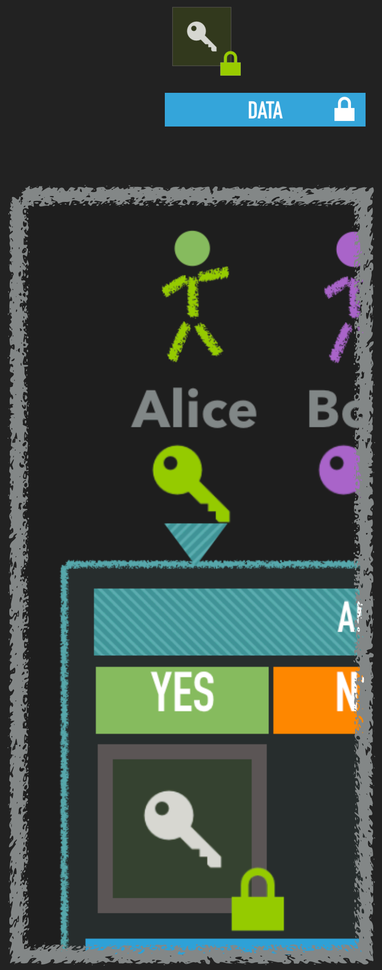


1. Alice enters her password
2. Application decrypts her strong long-term key
3. Alice now has her long-term key

Alice's long term secret key
encrypted with her password.

DECOUPLE PASSWORD AND LONGTERM KEY TO ALLOW EASY PASSWORD CHANGES

ACCESS CONTROL

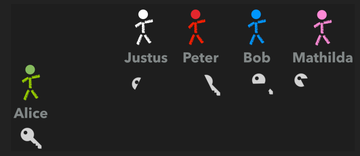


Situation:

- ✓ All data is secured under Alices key
- ✓ Alice can change her password by reencrypting her longterm key
- ⚠ Alice may forget her password

Question:

What happens when Alice forgets her password?



~BACKUP