Mid-term 25th October

Everyone in CM3

2 Groups – 2 exams

10:15-11:30	From Abbey to Jalal
11:45-13:00	From Jeanmonod to Zrouga

You need to be seated at that point

1h15 for the exam

1st group: you **CANNOT** leave before 11:30
and you **CANNOT** use your phone/computer

Mid-term 25th October

- Sit on the right seat (closest to the window) of each column (we select your column) →

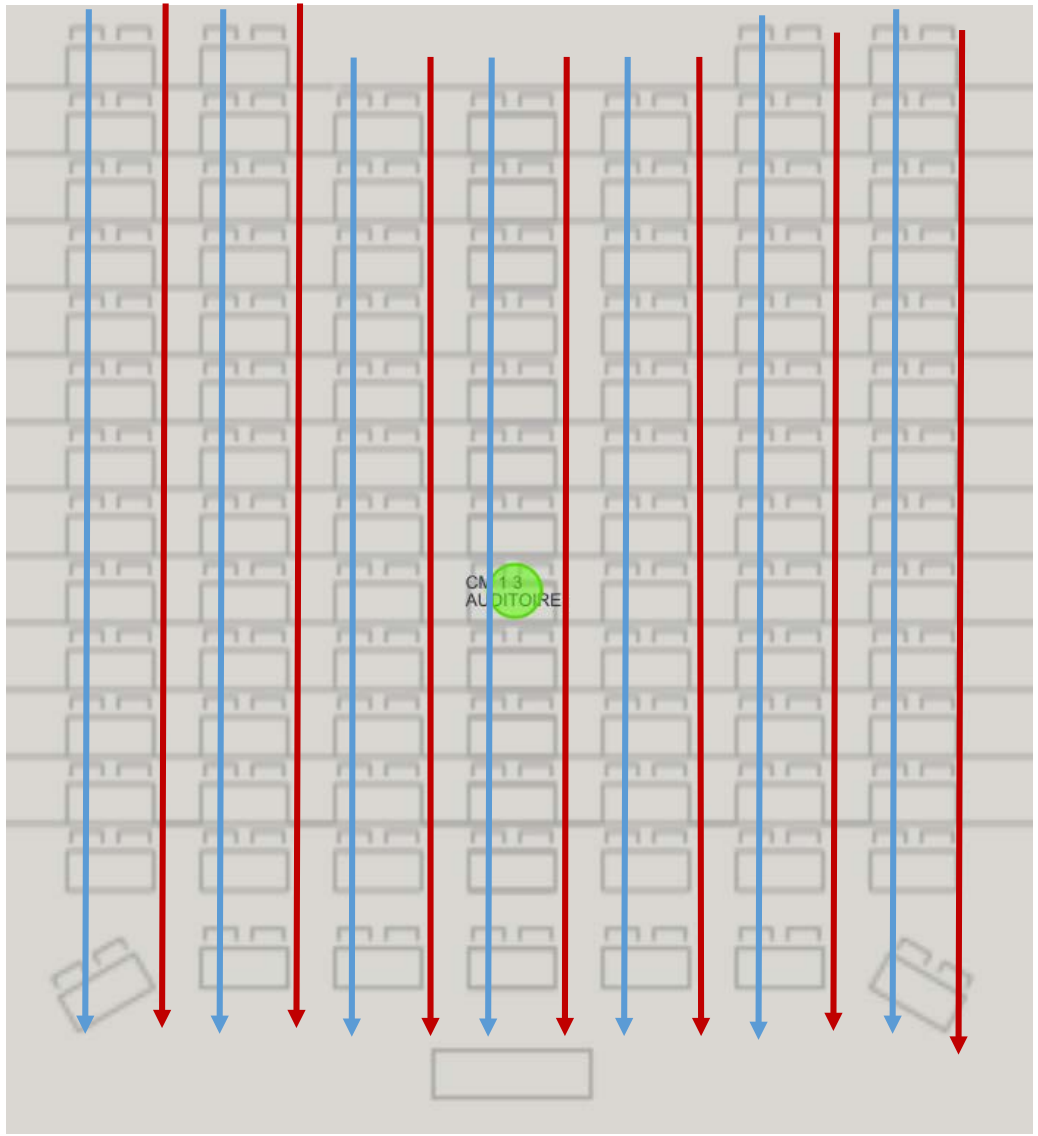
- Put your backpack/coat on the corridor to the left →

- Have your CAMIPRO on the desk

- At 10:15 /11:30 we will provide the exams From top to bottom.

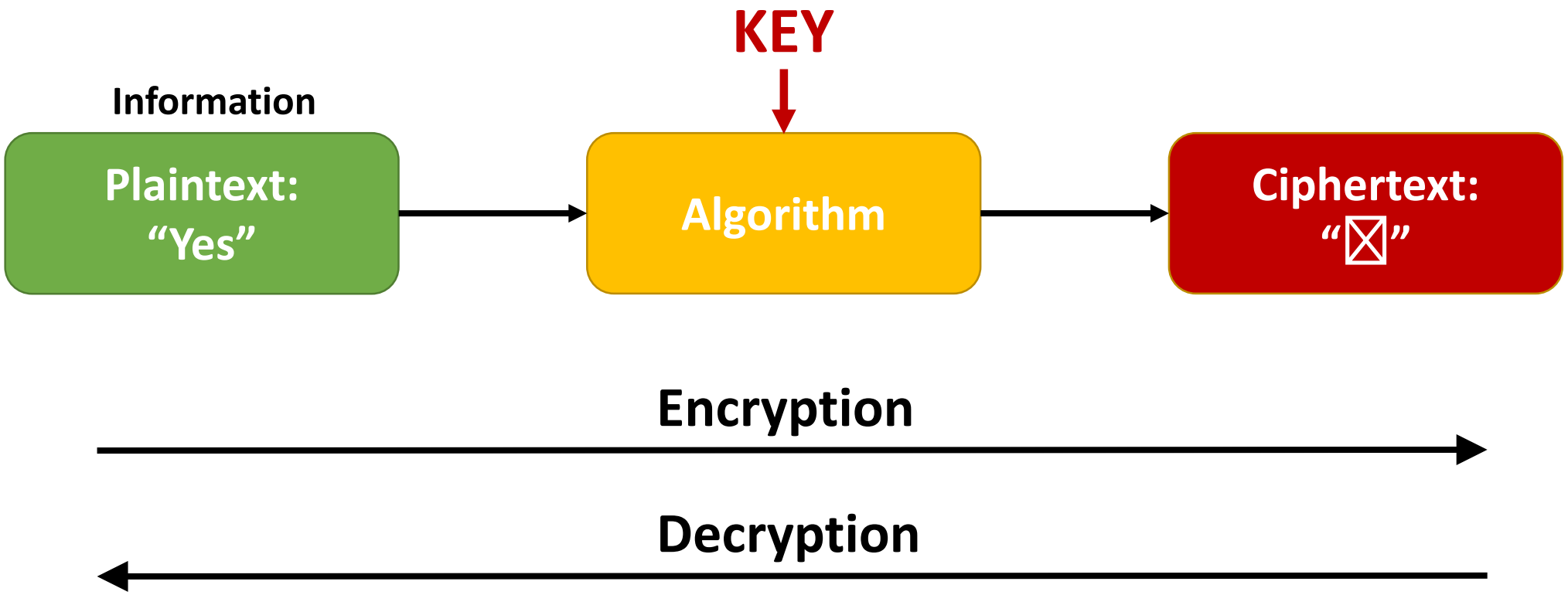
- 1h15 later we will take the exams from top to bottom

- As soon as your exam is taken, take your stuff and leave.



Last week

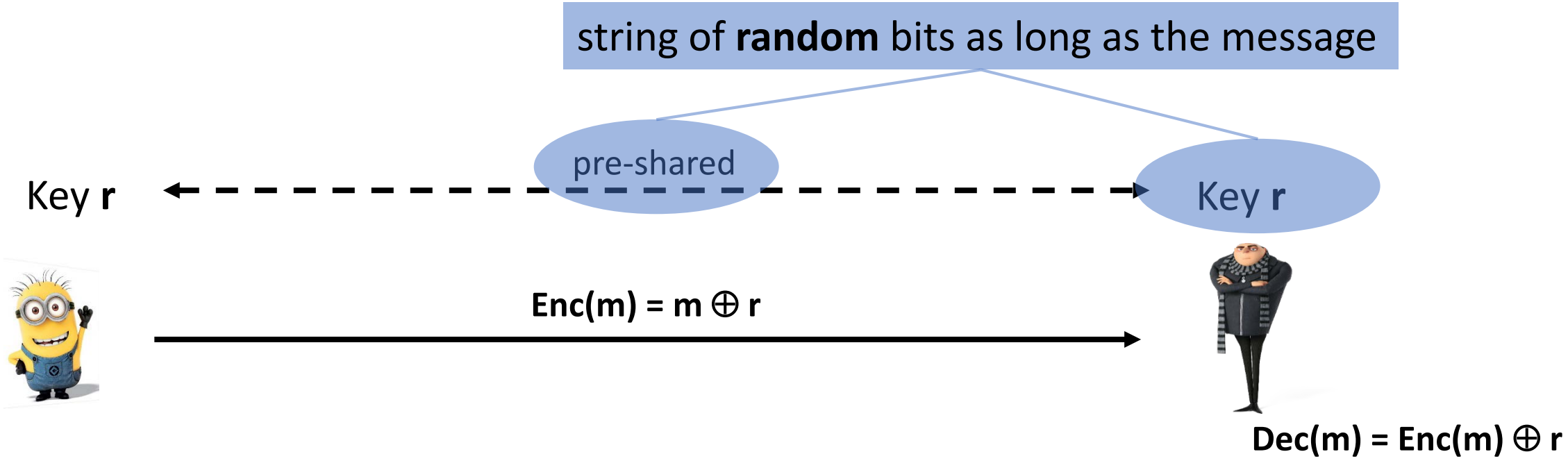
Confidentiality: information cannot be accessed by unauthorized parties



As opposed to encoding, encryption cannot be reversed without a KEY

Last week

One Time Pad – perfect secrecy

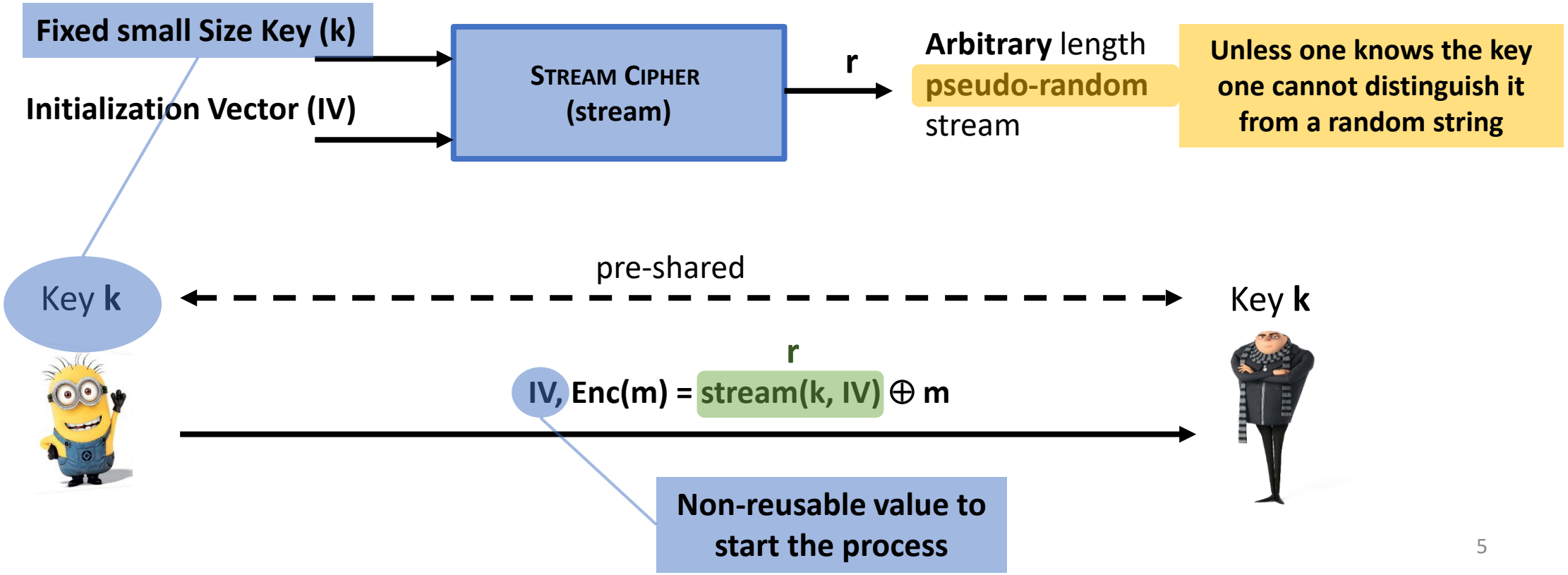


Message	YEAH
Binary (ASCII)	01111001011001010110000101101000
Pad	01110101000111010100101001001010
Encryption	00001100011110000010101100100010

Last week

SYMMETRIC CRYPTOGRAPHY
Encryption of plaintext and decryption of ciphertext are done using **THE SAME KEY**

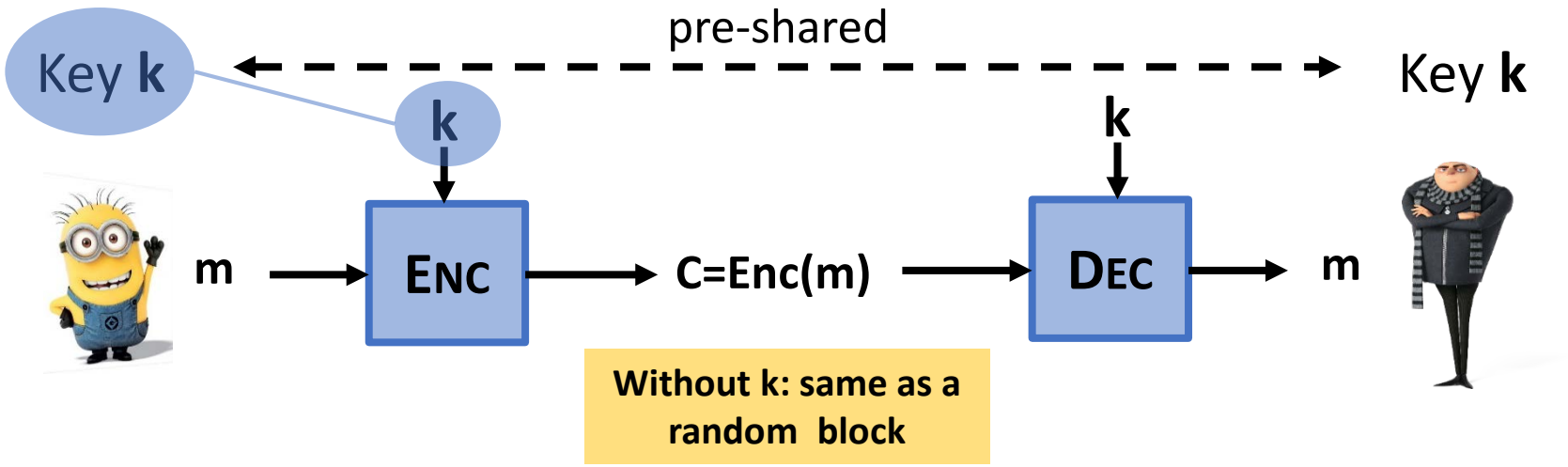
Stream cipher: a cheap One Time Pad



Last week

SYMMETRIC CRYPTOGRAPHY
Encryption of plaintext and decryption of ciphertext are done using **THE SAME KEY**

Block cipher



Messages are longer than a block! **Modes of operation** (CBC, CTR)

Last week

Integrity → Message Authentication Code

Fixed (short) Size Key (k)

Message (m)

MESSAGE AUTHENTICATION CODE

Short output (MAC)

Hard to generate (m, MAC(k;m)) without knowing k

Key **k** ← pre-shared → Key **k**

m, MAC(k;m)

Two properties:

m cannot be tampered with

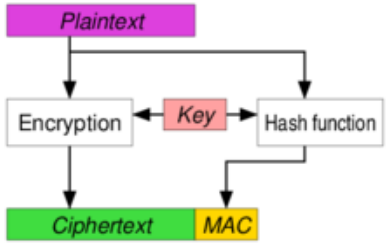
message comes from Bob/Gru

Mutual authentication but not to 3rd parties!

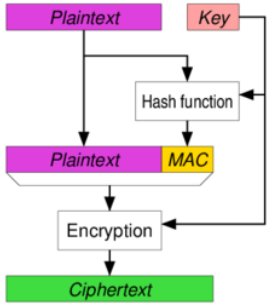
Message can be repudiated!

Last week

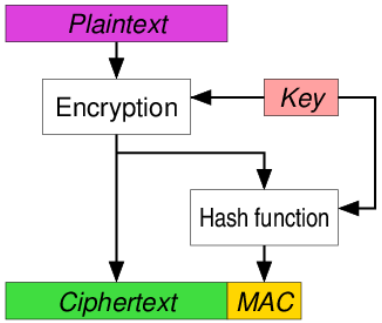
Confidentiality + Integrity



ENCRYPT-AND-MAC ✗



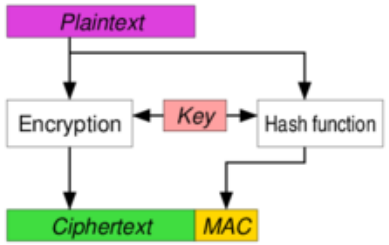
MAC-THEN-ENCRYPT ✗



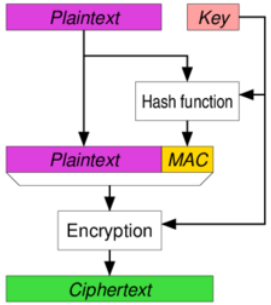
ENCRYPT-THEN-MAC ✓

Last week

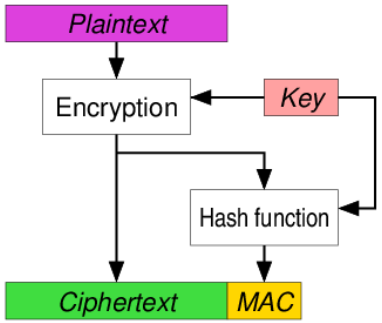
Confidentiality + Integrity



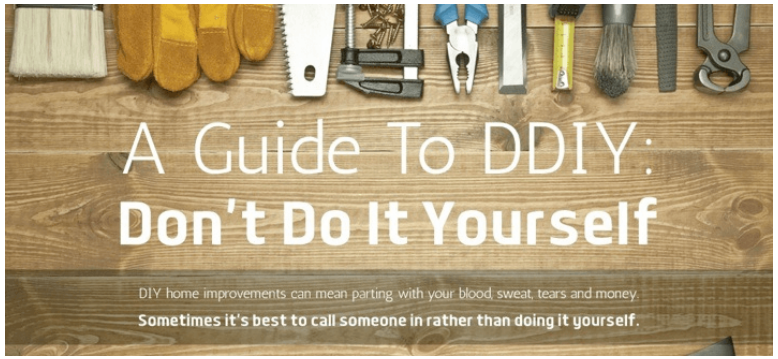
ENCRYPT-AND-MAC ✗



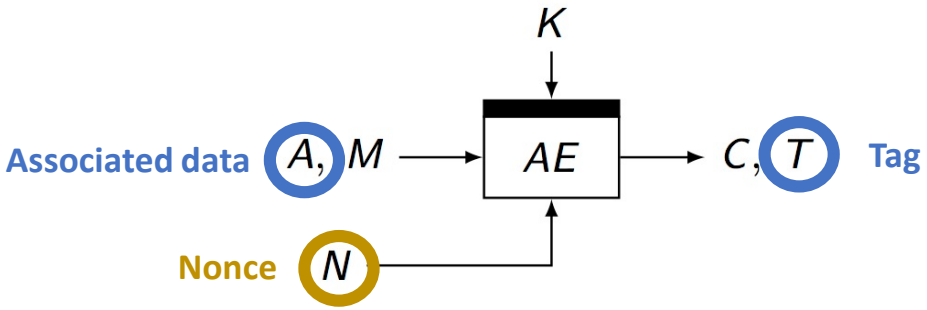
MAC-THEN-ENCRYPT ✗



ENCRYPT-THEN-MAC ✓



Authenticated Encryption with Associated Data (AEAD)



GCM, EAX: modes of operation that automatically provide confidentiality and integrity without the need of compose primitives.
Require extra input (A), and provide extra output (T) that enable to check integrity.

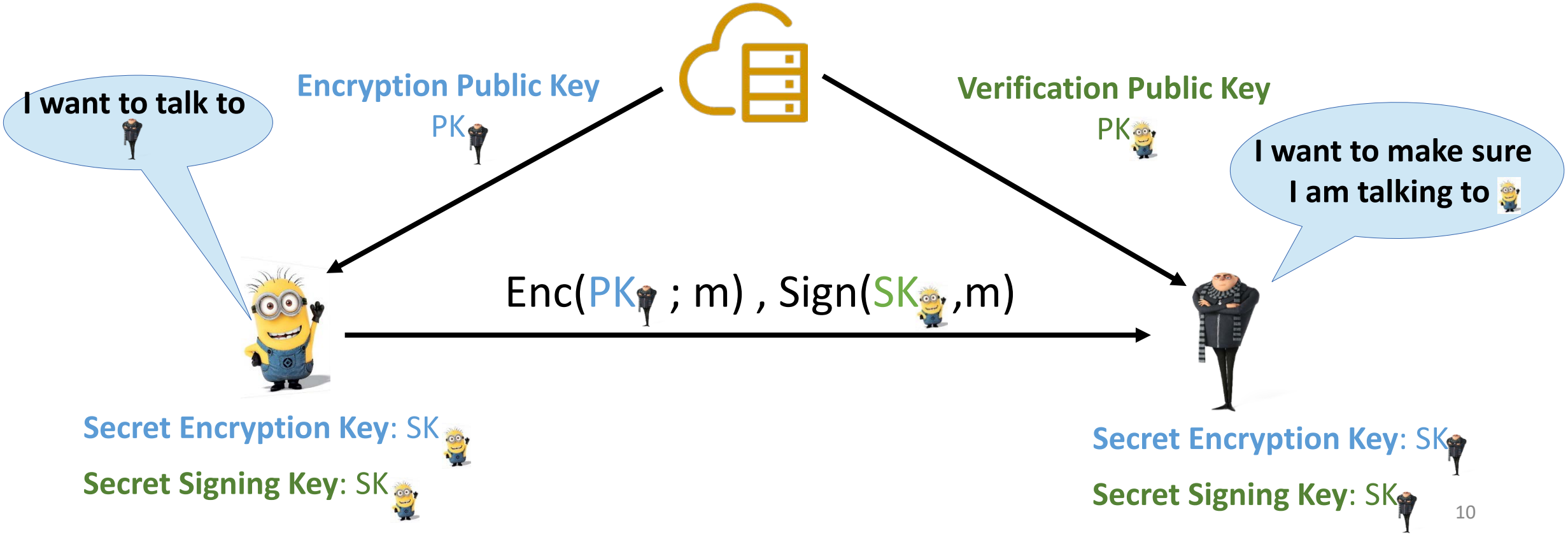
Last week

ASYMMETRIC CRYPTOGRAPHY

Users have two pairs of keys (secret key SK, public key PK)

Confidentiality — **Dec(SK,Enc(PK,m))= m**

Integrity/Authentication — **Sig(SK,m) = s; Verify(PK,Sig(SK,m)) = YES/NO**



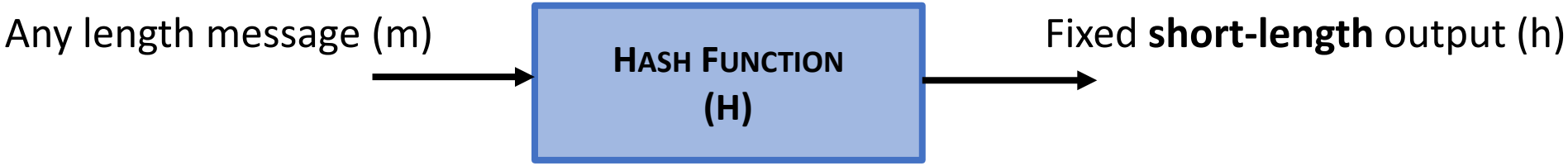
Asymmetric cryptography limitations

Computationally costly compared with most symmetric key algorithms of equivalent security

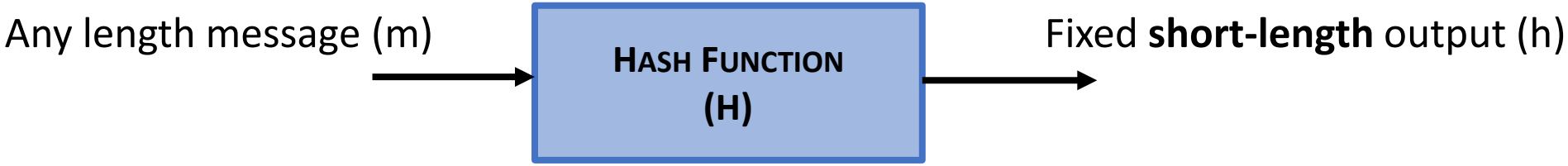
Signing and encrypting **are slow operations**

Not suitable to encrypt **large amounts of data**

Hash functions



Hash functions



THREE SECURITY PROPERTIES

PRE-IMAGE RESISTANCE

Given H(m), difficult to get m

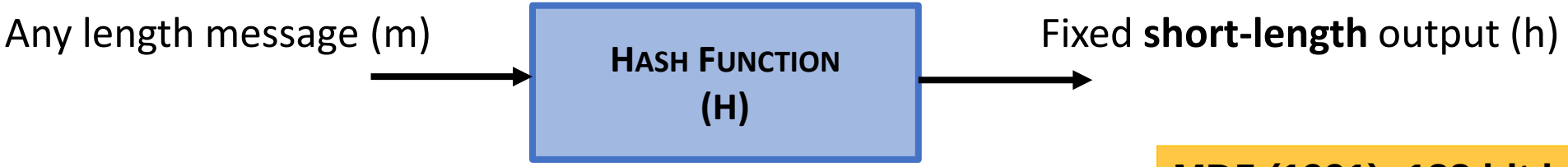
SECOND PRE-IMAGE RESISTANCE

Given H(m), difficult to get an m' such that H(m') = H(m)

COLLISION RESISTANCE

Difficult to find any m, m' such that H(m) = H(m')

Hash functions



MD5 (1991): 128 bit hash – insecure

SHA0, SHA1: 160 bits – insecure

SHA-2 (224/256 /384/512) – OK but slow

New NIST standard by competition

SHA-3 (224/256 /384/512)

THREE SECURITY PROPERTIES

PRE-IMAGE RESISTANCE

Given H(m), difficult to get m

SECOND PRE-IMAGE RESISTANCE

Given H(m), difficult to get an m' such that H(m') = H(m)

COLLISION RESISTANCE

Difficult to find any m, m' such that H(m) = H(m')

USES

Support digital signatures, build HMAC, password storage, file integrity, secure commitments, secure logging, blockchain,…

Hash functions

Any length message (m) → **HASH FUNCTION (H)** → Fixed **short-length** output (h)

THREE SECURITY PROPERTIES

PRE-IMAGE RESISTANCE

Given H(m), difficult to get m

SECOND PRE-IMAGE RESISTANCE

Given H(m), difficult to get an m' such that H(m') = H(m)

COLLISION RESISTANCE

Difficult to find

Don't design your own ⚠ **HMAC != H(K||M)**

MD5 (1991): 128 bit hash – insecure

SHA0, SHA1: 160 bits – insecure

SHA-2 (2) – OK but slow

Don't design your own ⚠

New NIST standard by competition

SHA-3 (224/256 /384/512)

USES

Support digital signatures, build HMAC, password storage, file integrity, secure commitments, secure logging, blockchain,…

Digital signatures

Examples:

NIST DSA

RSA-PSS

Don't design your own

Public Key Infrastructure

h=H(m)

m, Sign(SK ,h)

Secret Key: SK

Secret Key: SK

Digital signatures

Examples:

NIST DSA

RSA-PSS

Don't design your own

Public Key Infrastructure

Public Key

PK

I want to make sure I am talking to

h=H(m)

m, Sign(SK ,h)

Secret Key: SK

Secret Key: SK

h = H(m) -> Verify(PK , Sign(SK ,h))

Digital signatures

Examples:
NIST DSA
RSA-PSS

Don't design your own

Refresher

~~**PRE-IMAGE RESISTANCE**~~
~~Given H(m), difficult to get m~~

SECOND PRE-IMAGE RESISTANCE
Given H(m), difficult to get an m' such that H(m') = H(m)

COLLISION RESISTANCE
Difficult to find any m, m' such that H(m) = H(m')

Public Key Infrastructure

Public Key
PK

I want to make sure I am talking to

h=H(m)

m, Sign(SK ,h)

Secret Key: SK

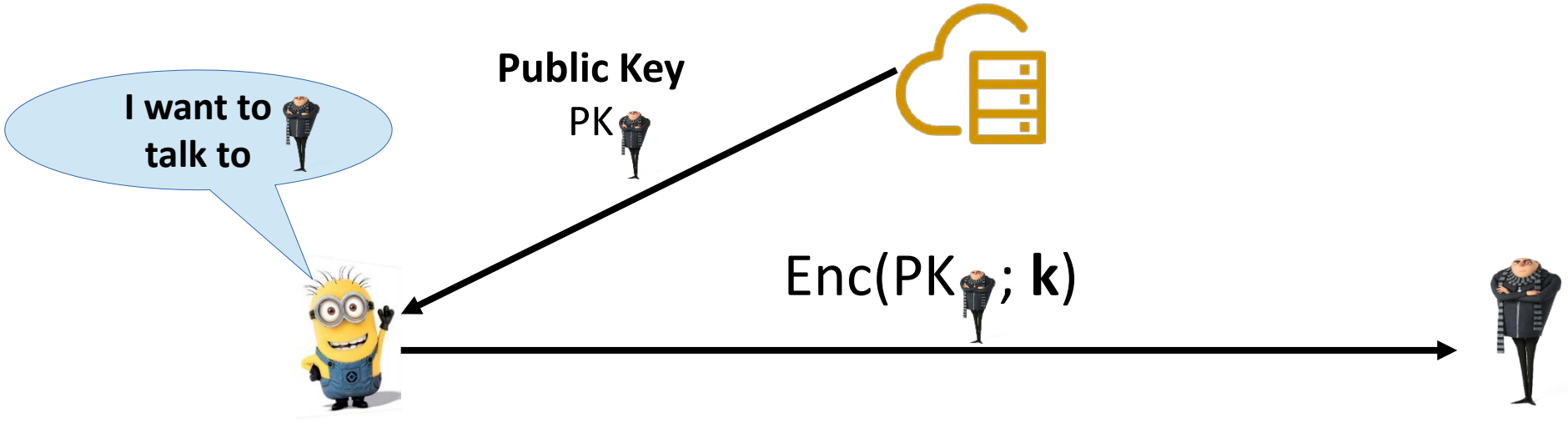
Secret Key: SK

h = H(m) -> Verify(PK , Sign(SK ,h))

Hybrid encryption

Asymmetric encryption **is slow,** but symmetric **is fast!**

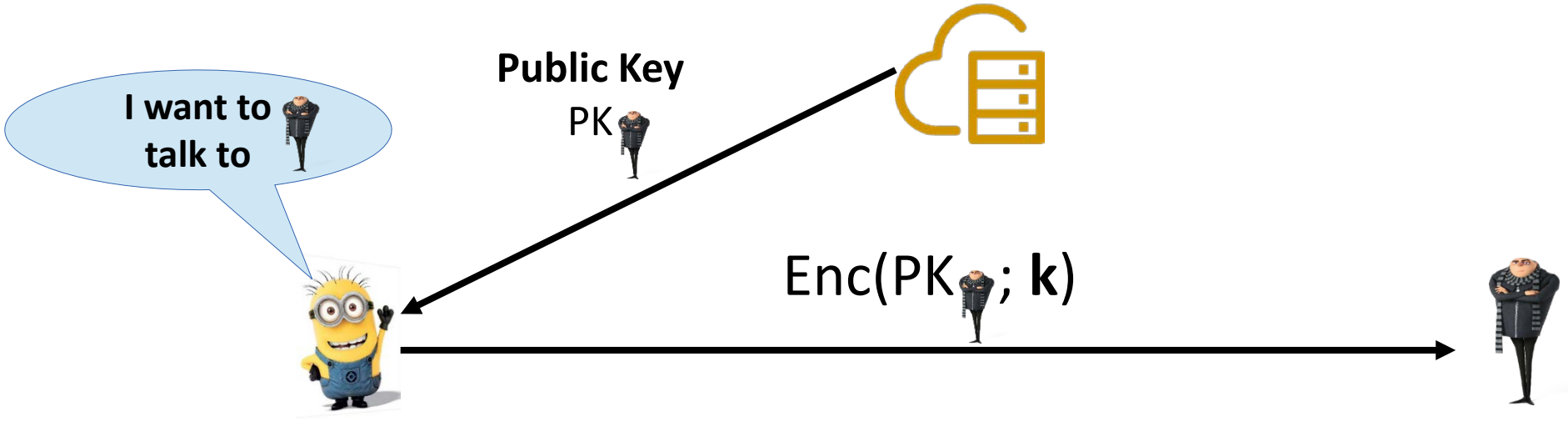
Step 1: establish a shared symmetric key k using “key transport”



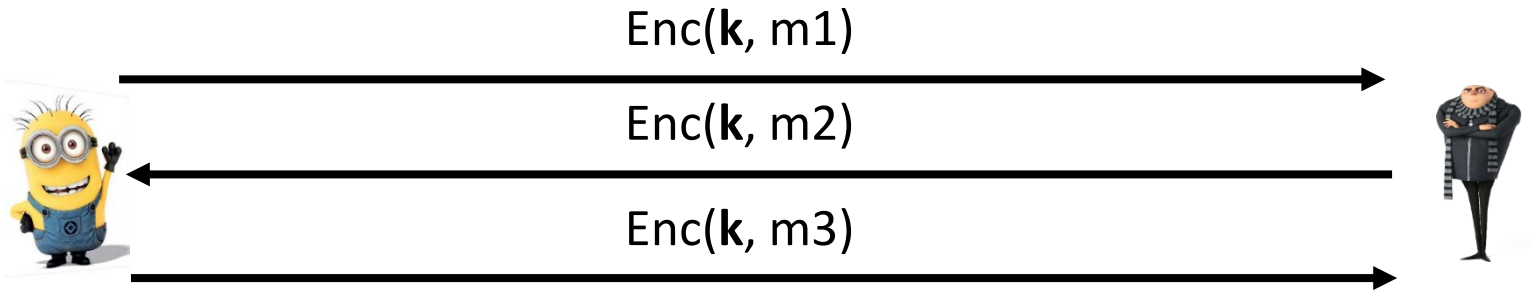
Hybrid encryption

Asymmetric encryption **is slow,** but symmetric **is fast!**

Step 1: establish a shared symmetric key k using “key transport”



Step 2: use the shared symmetric key k to encrypt the rest of the communication



Hybrid encryption

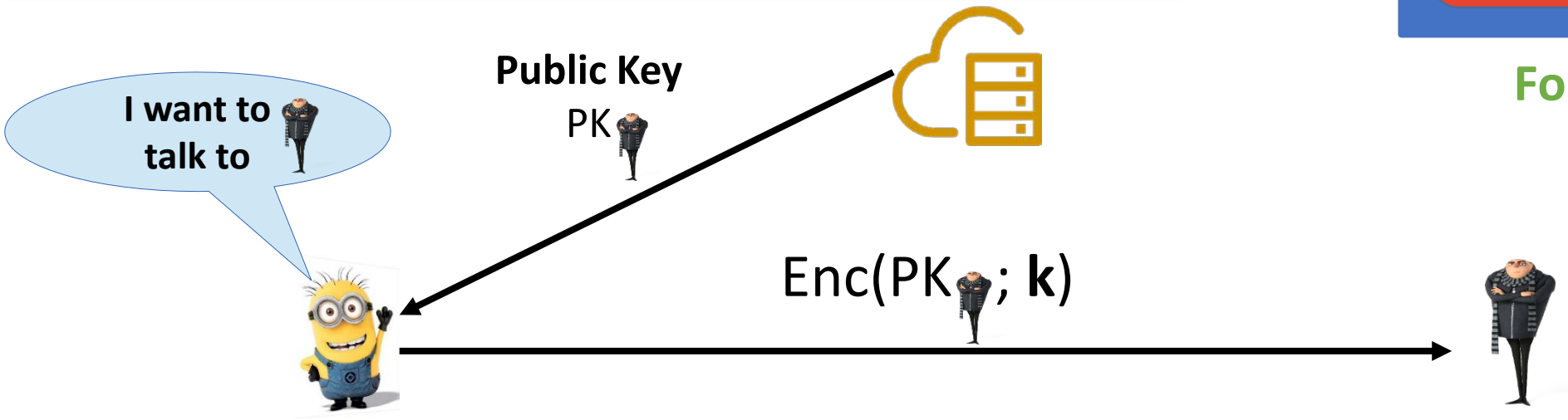
Asymmetric encryption **is slow,** but symmetric **is fast!**

Don't design your own

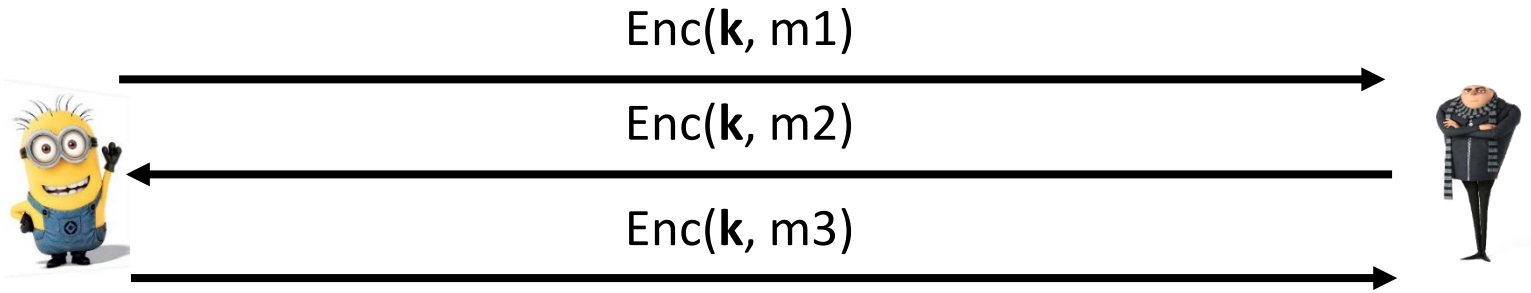
NOT SO SIMPLE!
e.g. ISO 9798-3
TLS

For authentication add signatures!!

Step 1: establish a shared symmetric key k using "key transport"



Step 2: use the shared symmetric key k to encrypt the rest of the communication



This process is repeated every time Bob wants to talk to Gru

Monday

Enc(PK  ; **k1**)

Enc(**k1**, m1)

Enc(**k1**, m2)

Enc(**k1**, m3)

Tuesday

Enc(PK  ; **k2**)

Enc(**k2**, m4)

Enc(**k2**, m5)

Enc(**k2**, m6)

Thursday

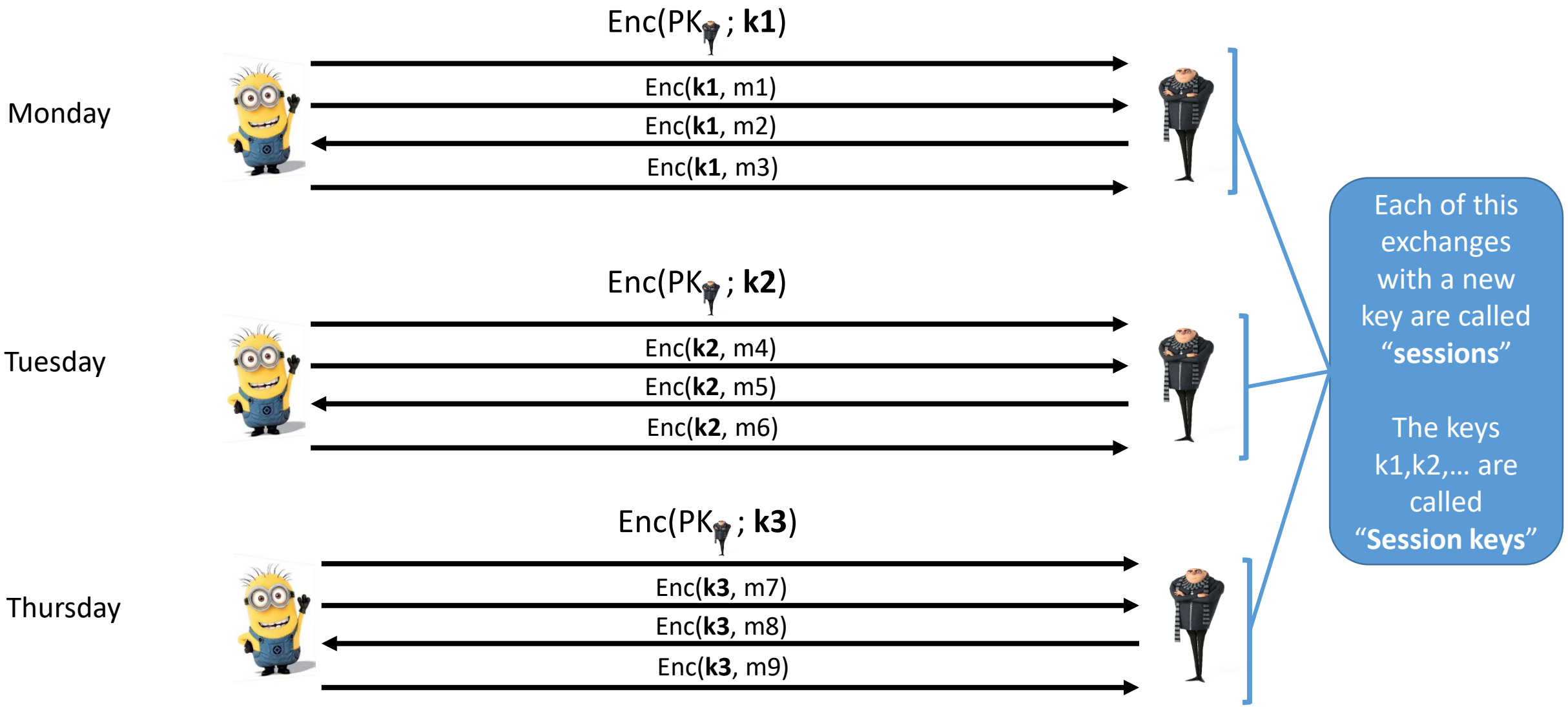
Enc(PK  ; **k3**)

Enc(**k3**, m7)

Enc(**k3**, m8)

Enc(**k3**, m9)

This process is repeated every time Bob wants to talk to Gru



What happens if the adversary gets access to Gru's asymmetric key on Thursday?

Monday

Enc(PK; **k1**)

Enc(**k1**, m1)

Enc(**k1**, m2)

Enc(**k1**, m3)

Tuesday

Enc(PK; **k2**)

Enc(**k2**, m4)

Enc(**k2**, m5)

Enc(**k2**, m6)

Thursday

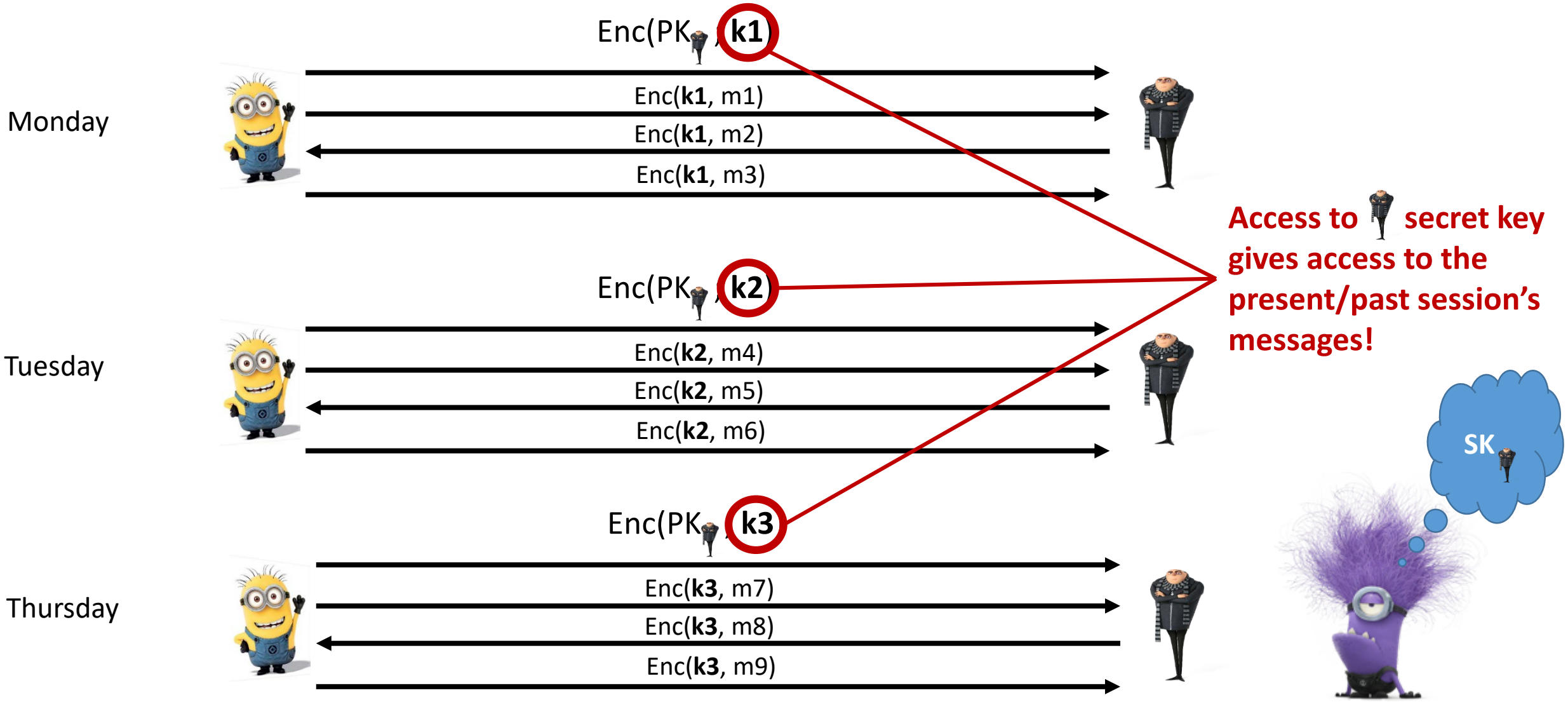
Enc(PK; **k3**)

Enc(**k3**, m7)

Enc(**k3**, m8)

Enc(**k3**, m9)

What happens if the adversary gets access to Gru's asymmetric key on Thursday?



What happens if the adversary gets access to Gru's asymmetric key on Thursday?

Monday

Enc(PK, **k1**)

Enc(**k1**, m1)

Enc(**k1**, m2)

Enc(**k1**, m3)

Tuesday

DESIRABLE PROPERTY

FORWARD SECRECY: the secrecy of the messages in a session is kept even if long term keys are compromised

If the adversary learns the key of Thursday's session, Monday and Tuesday should still be safe

Thursday

Enc(**k3**, m7)

Enc(**k3**, m8)

Enc(**k3**, m9)

Access to secret key gives access to the present/past session's messages!

SK

How can we obtain this property??

The math you need for the basics

Arithmetic modulo a number: clock arithmetic

6 (mod 12) = 6 (mod 12)

12 (mod 12) = 0 (mod 12)

14 (mod 12) = 2 (mod 12)

Arithmetic modulo a large prime p (>1024 bits)

Addition and multiplication (mod p) can be computed

Exponentiation can be computed [Given (a, x) → a^x mod p?]

Discrete logarithms are **HARD**! [Given (a, a^x mod p) → x?]

Basic Diffie-Hellman key exchange

Every time Bob wants to talk to Gru…

Shared **public** parameters p , g

Public Key
$P_b = g^x \bmod p$

Secret Key: x (random!)

Public Key
$P_a = g^y \bmod p$

Secret Key: y (random!)

Basic Diffie-Hellman key exchange

Every time Bob wants to talk to Gru…

Shared **public** parameters p , g

Because of the discrete logarithm hardness, an adversary observing these values cannot recover x and y, therefore cannot compute k

Public Key
$P_b = g^x \bmod p$

P_b

P_a

Public Key
$P_a = g^y \bmod p$

Secret Key: x (random!)

$(P_a)^x = \mathbf{g^{xy}}$ **(mod p)**

Secret Key: y (random!)

$(P_b)^y = \mathbf{g^{xy}}$ **(mod p)**

Basic Diffie-Hellman key exchange

Every time Bob wants to talk to Gru…

Shared **public** parameters p , g

Public Key
$P_b = g^x \bmod p$

Public Key
$P_a = g^y \bmod p$

Because of the discrete logarithm hardness, an adversary observing these values cannot recover x and y, therefore cannot compute k

P_b →

← P_a

Secret Key: x (random!)

$(P_a)^x = \mathbf{g^{xy}}$ **(mod p)**

Shared secret!!
$k = g^{xy}$ (mod p)
To encrypt messages for the session

Secret Key: y (random!)

$(P_b)^y = \mathbf{g^{xy}}$ **(mod p)**

Basic Diffie-Hellman key exchange

Every time Bob wants to talk to Gru…

Shared **public** parameters p , g

Public Key
$P_b = g^x \bmod p$

Public Key
$P_a = g^y \bmod p$

After the session is ended, delete the secrets x and y.
The key can never be recovered.
Forward secrecy is achieved!!

Secret Key: x (random!)

$(P_a)^x = \mathbf{g^{xy}}$ **(mod p)**

Shared secret!!
$k = g^{xy}$ (mod p)
To encrypt messages for the session

Secret Key: y (random!)

$(P_b)^y = \mathbf{g^{xy}}$ **(mod p)**

Summary of the lecture

Symmetric cryptography

- Confidentiality: Stream ciphers, Block ciphers (modes of operation!)
- Integrity / Authentication: Message Authentication Codes (MACs)

Asymmetric cryptography

- Confidentiality: Encryption
- Integrity / Authentication: Digital signatures

Hash functions

- Three security properties
- Support Digital Signatures + other functions

Hybrid encryption

best both worlds!

Forward secrecy

Diffie Hellman

Unanswered questions

- How do I build a block cipher?
- How do I build a stream cipher?
- How do I build a hash function?
- How do I implement those?

On the basis of this course: **Do not!**

And only use well established and standardised modes of operation and protocols

Use well established, audited libraries

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

SPRING
SECURITY AND PRIVACY ENGINEERING LABORATORY

Computer Security (COM-301)
Authentication

Carmela Troncoso
SPRING Lab
carmela.troncoso@epfl.ch

Some slides/ideas adapted from: Tuomas Aura, Yoshi Kohno, Trent Jaeger

Textbooks

Ross Anderson – Chapters:

- Protocols
- Passwords
- Biometrics

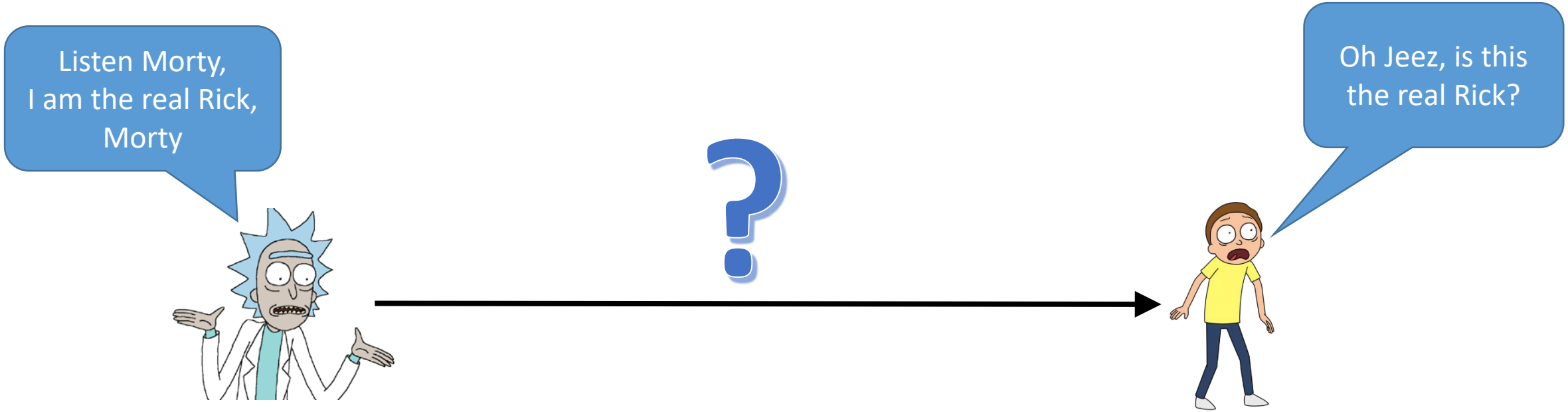
Dieter Gollmann - Chapter: Identification and authentication

Handbook of Applied Cryptography by A. Menezes, P. van Oorschot and S. Vanstone - http://cacr.uwaterloo.ca/hac/about/chap10.pdf

What is authentication?

AUTHENTICATION

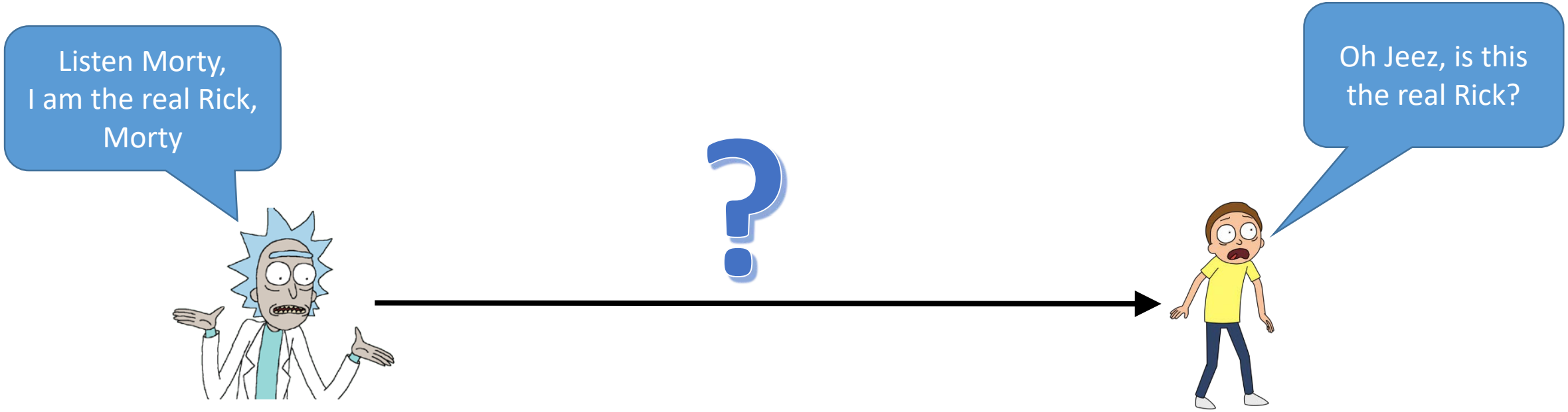
The process of verifying a claimed identity



What is authentication?

AUTHENTICATION

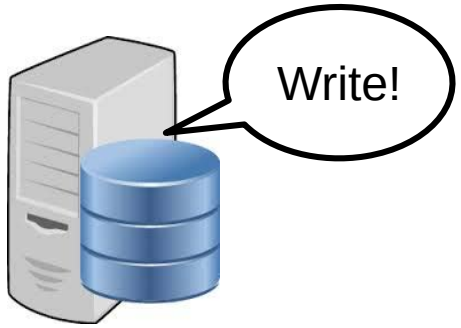
The process of verifying a claimed identity



!= MESSAGE AUTHENTICATION

The message comes from the designated sender, and has not been modified

Where does Authentication fit?



Authentication

Authorization (weeks 2 and 3)

The system needs to bind the actor to a **principal** before authorization

abstract entity that is authorized to act

(***users***, *connections, processes)*

The system needs to decide whether the **principal** is authorized

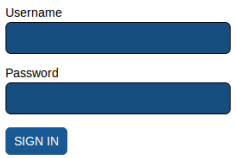
Access control

Ways to Prove Who You Are

TRADITIONAL

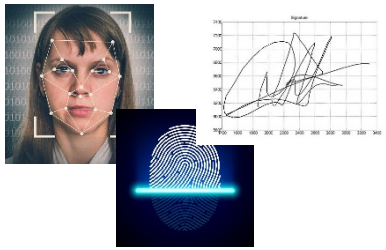
What you know

password, secret key



What you are

biometrics



What you have

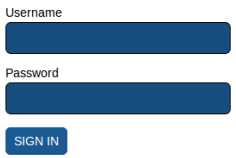
Smart card, secure tokens

Ways to Prove Who You Are

Traditional

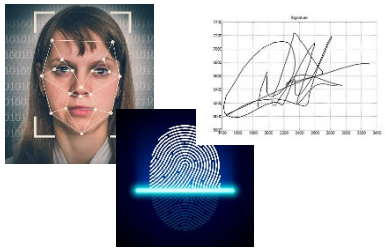
What you know

password, secret key



What you are

biometrics



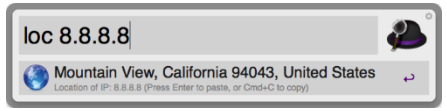
What you have

Smart card, secure tokens

Modern

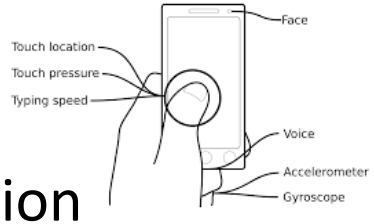
Where you are

location, IP address



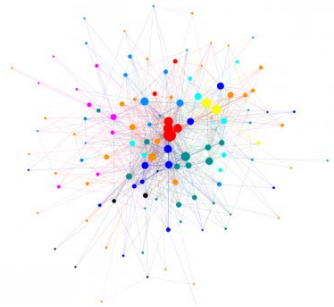
How you act

behavioural authentication



Who you know

social ties



Many others…

Ways to Prove Who You Are

Traditional

What you know

pas

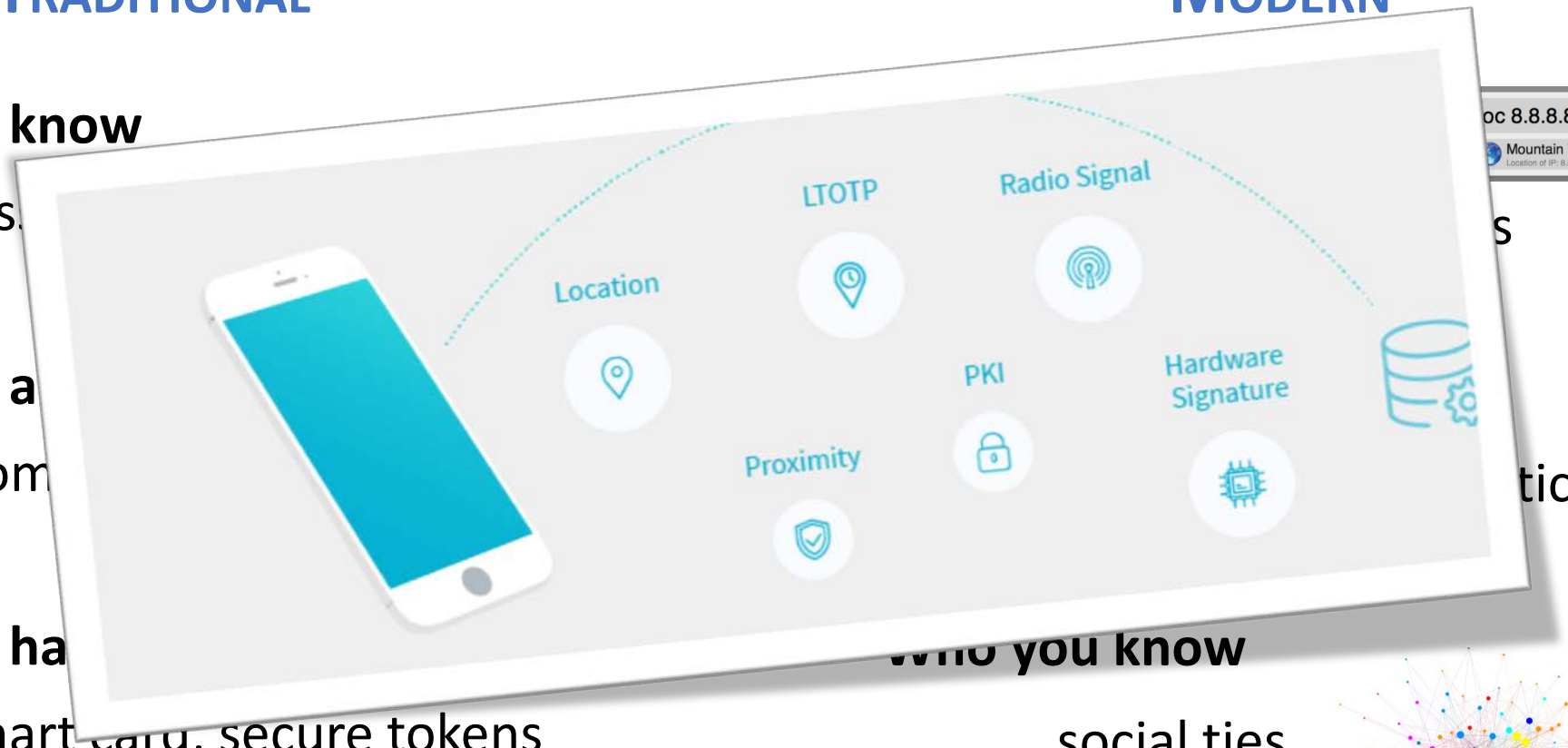
What you a

biom

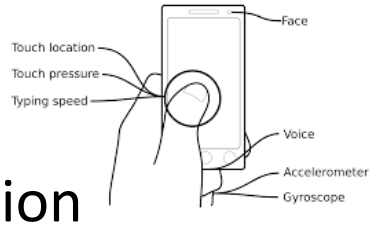
What you ha

Smart card, secure tokens

Modern



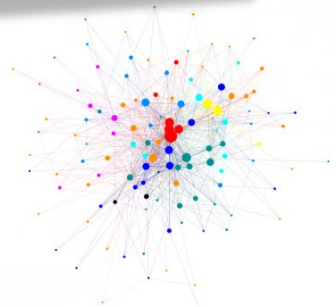
s



tication

Who you know

social ties



What you know: Passwords

PASSWORD
Secret shared between user and system

User has a secret password → System checks it to authenticate the user

What you know: Passwords

PASSWORD
Secret shared between user and system

User has a secret password → System checks it to authenticate the user

PROBLEMS TO BE SOLVED

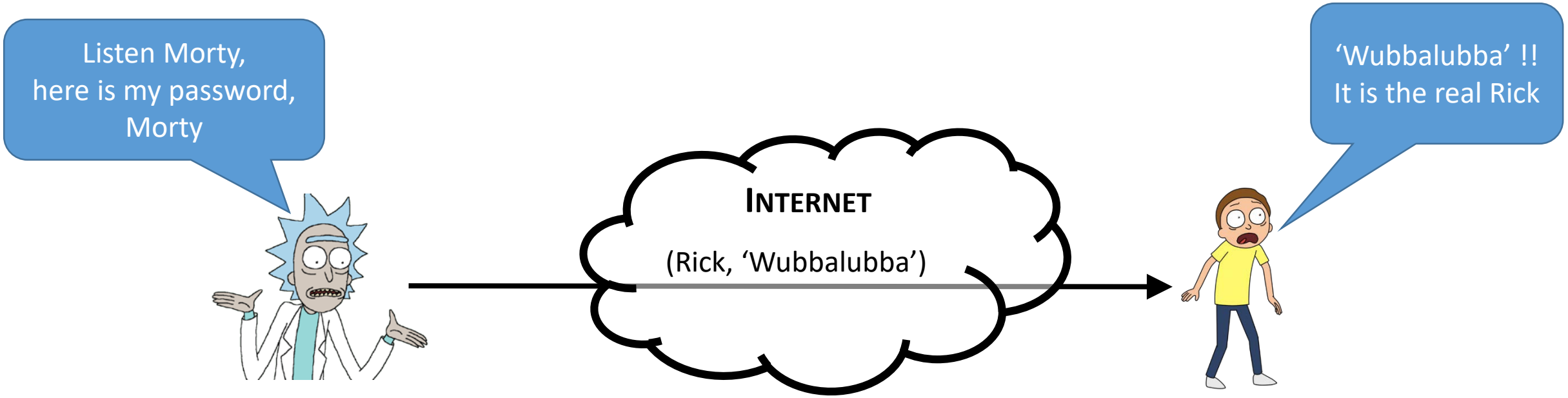
Secure transfer: the password may be eavesdropped when communicated

Secure check: naïve checks may leak information about the password

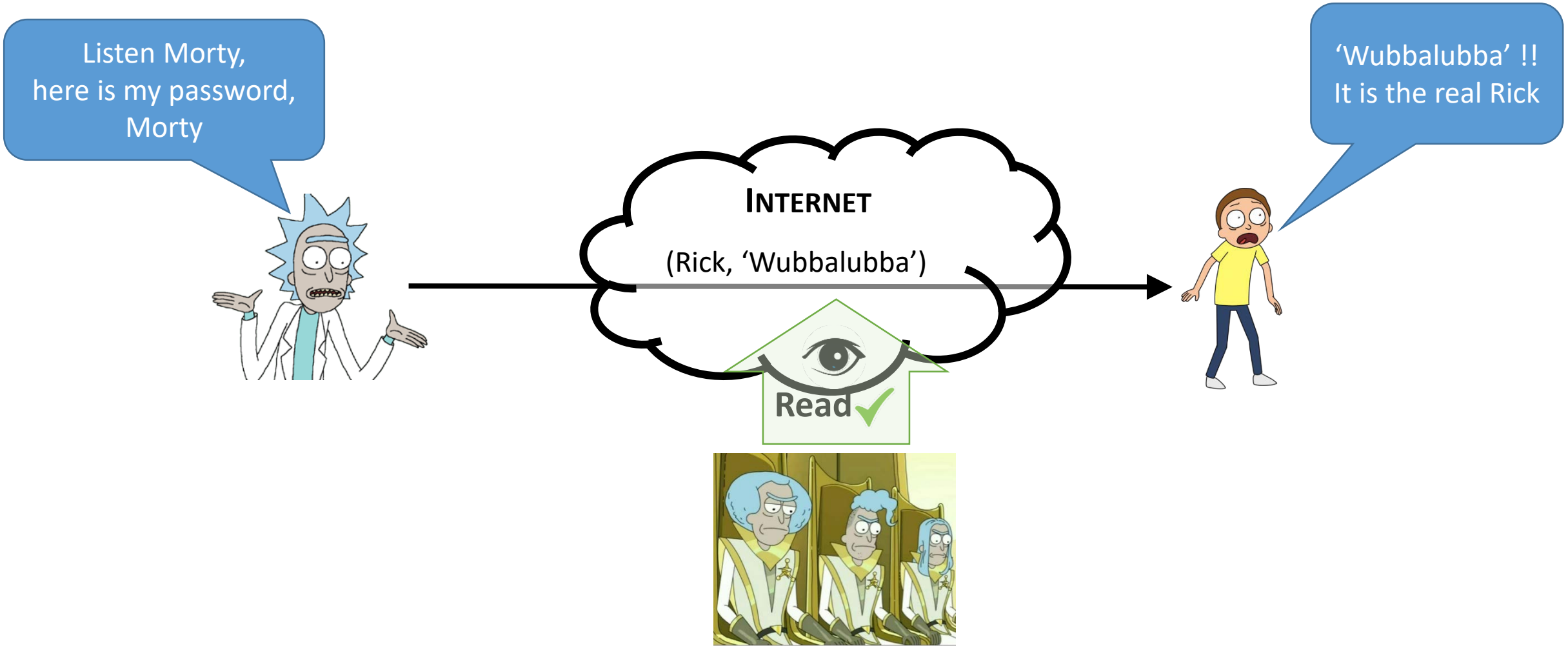
Secure storage: if stolen the full system is compromised!

Secure passwords: easy-to-remember passwords tend to be easy to guess

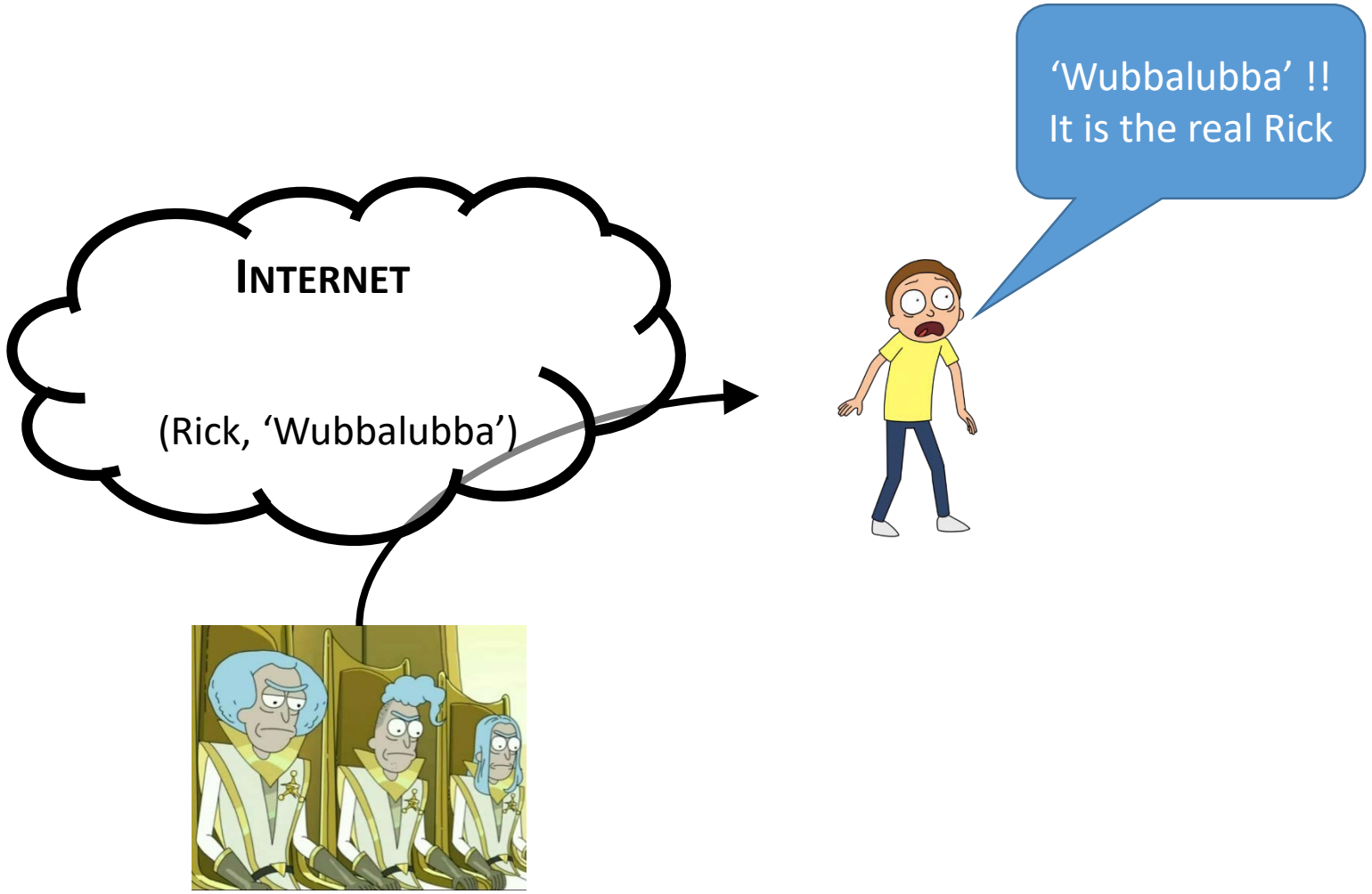
Secure transfer



Secure transfer

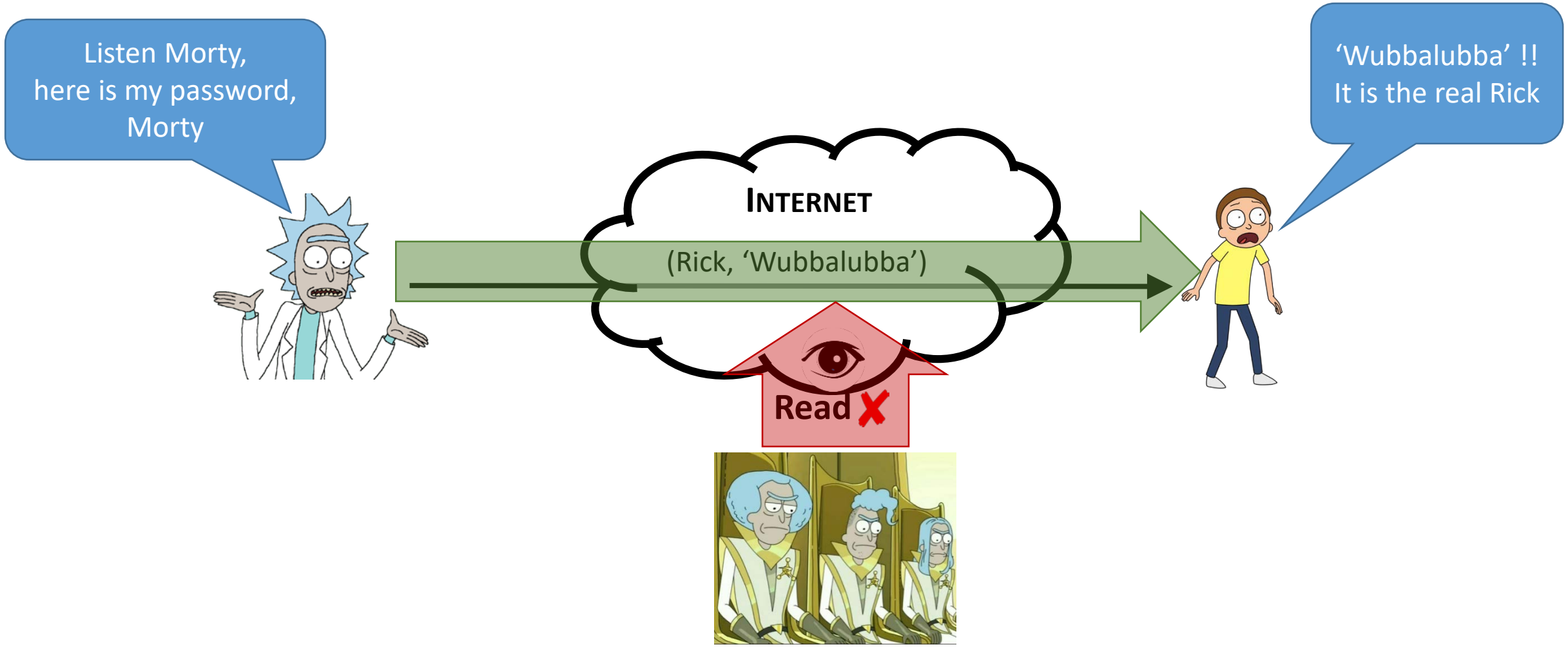


Secure transfer



Secure transfer

Encrypt the channel!!



Secure transfer

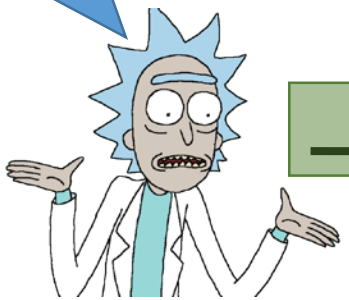
Encrypt the channel!!

TLS / HTTPS (HTTP-over-TLS)

***Roughly*... combine:**
- **Diffie Hellmann**
- **Digital signatures**
- **Hybrid encryption**

Week 9/10 !!

Listen Morty,
here is my password,
Morty



INTERNET

(Rick, 'Wubbalubba')

Read ✗

'Wubbalubba' !!
It is the real Rick

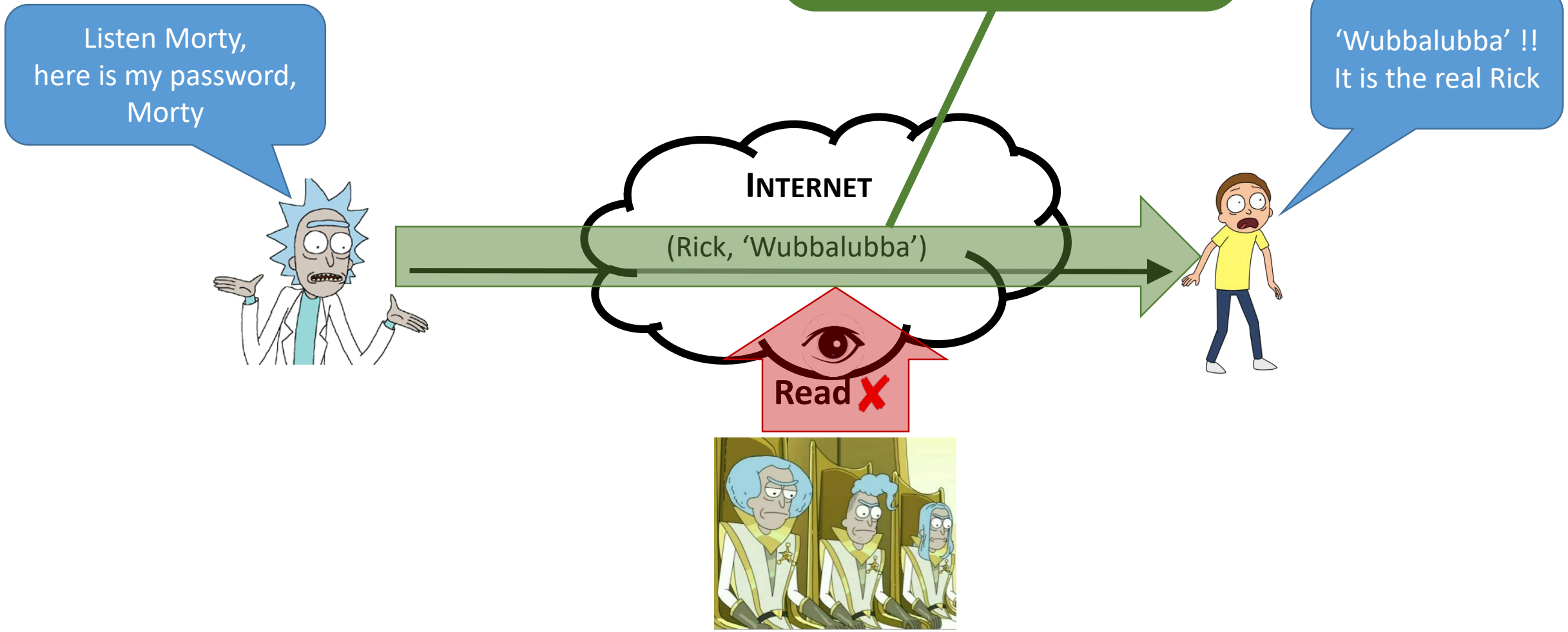
Secure transfer

Encrypt the channel!!

TLS / HTTPS (HTTP-over-TLS)

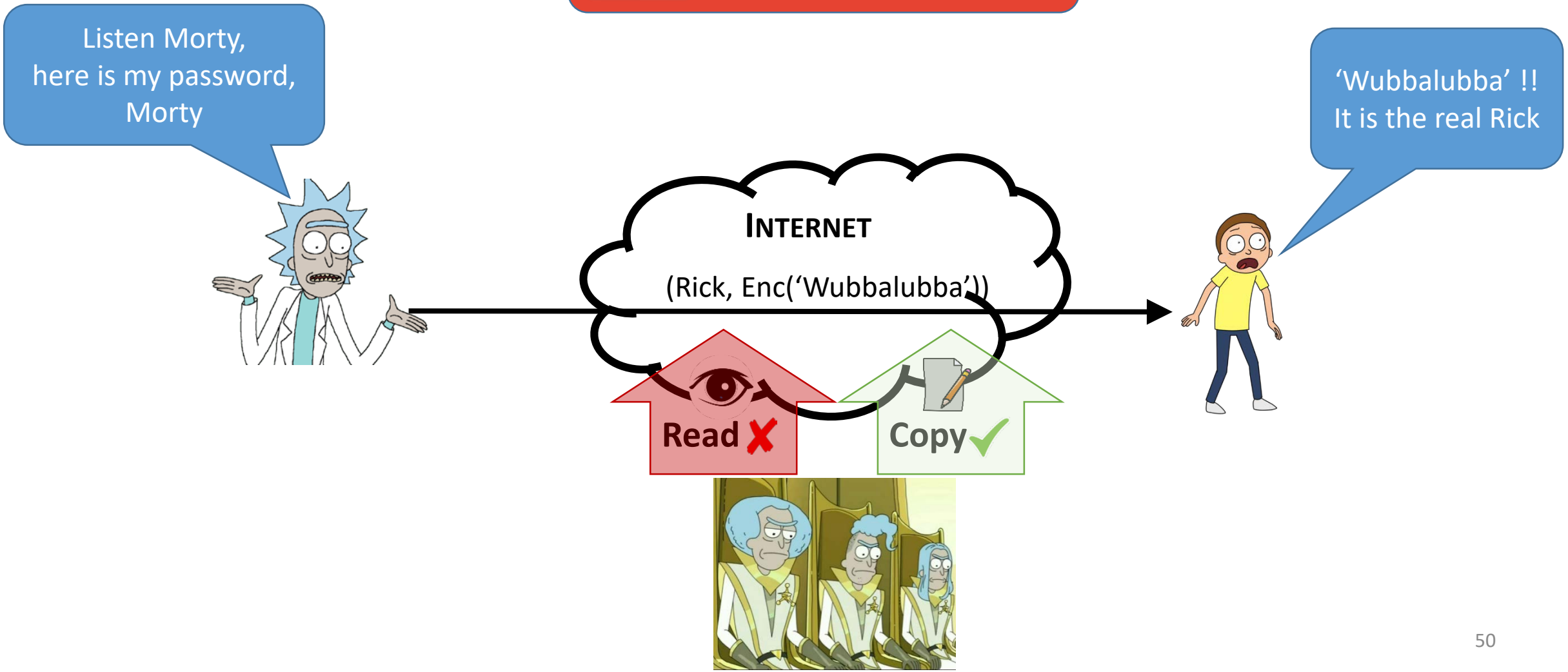
***Roughly*... combine:**
- **Diffie Hellmann**
- **Digital signatures**
- **Hybrid encryption**

Week 9/10 !!



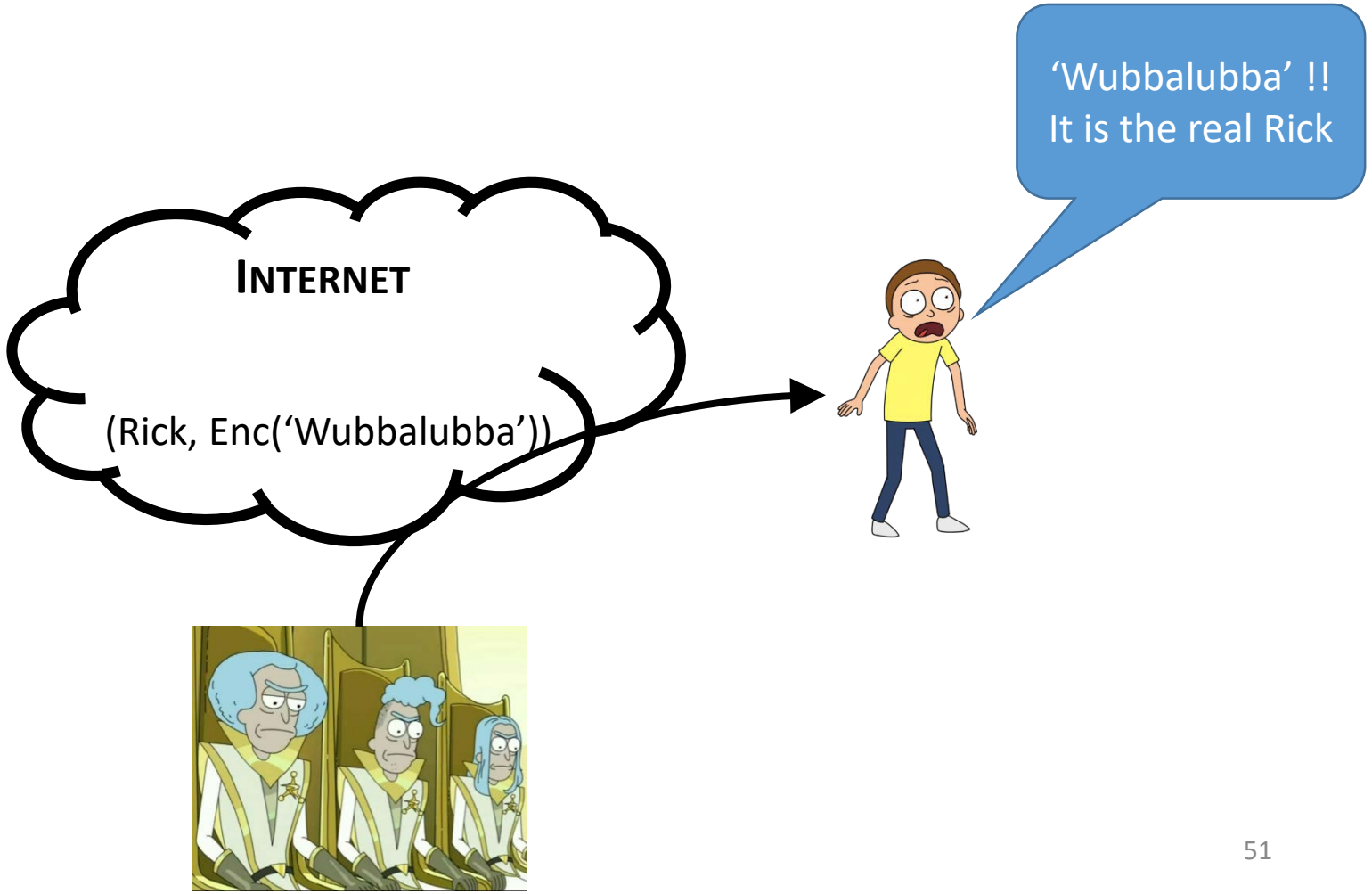
Secure transfer – Beware of replay attacks

One of the most difficult aspects in authentication



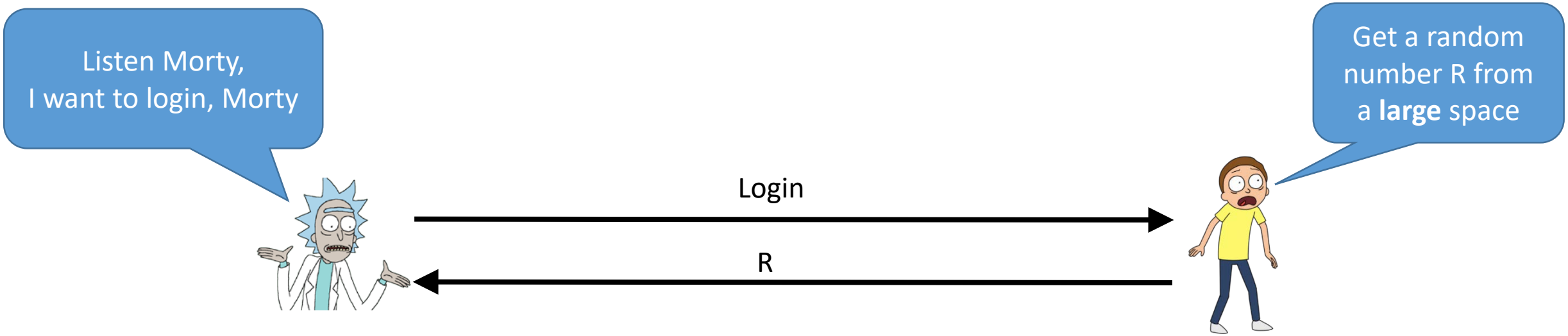
Secure transfer – Beware of replay attacks

One of the most difficult aspects in authentication



Challenge-Response protocols

Solution to replay attacks



Challenge-Response protocols

Solution to replay attacks

Listen Morty,
I want to login, Morty

Login →

← R

Get a random number R from a **large** space

Listen Morty,
here is my password,
Morty

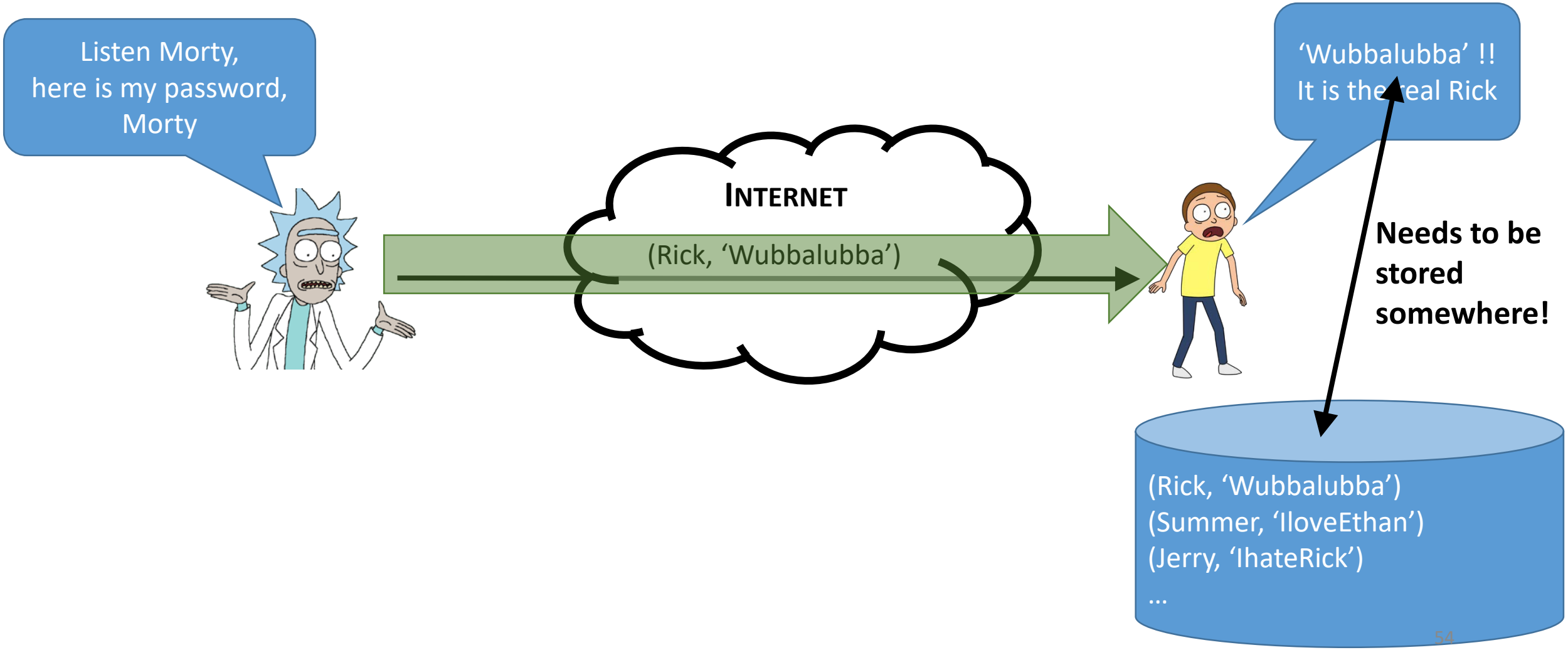
(Rick, Enc('Wubbalubba', **R**)) →

'Wubbalubba' !!
It is the real Rick
(delete R)

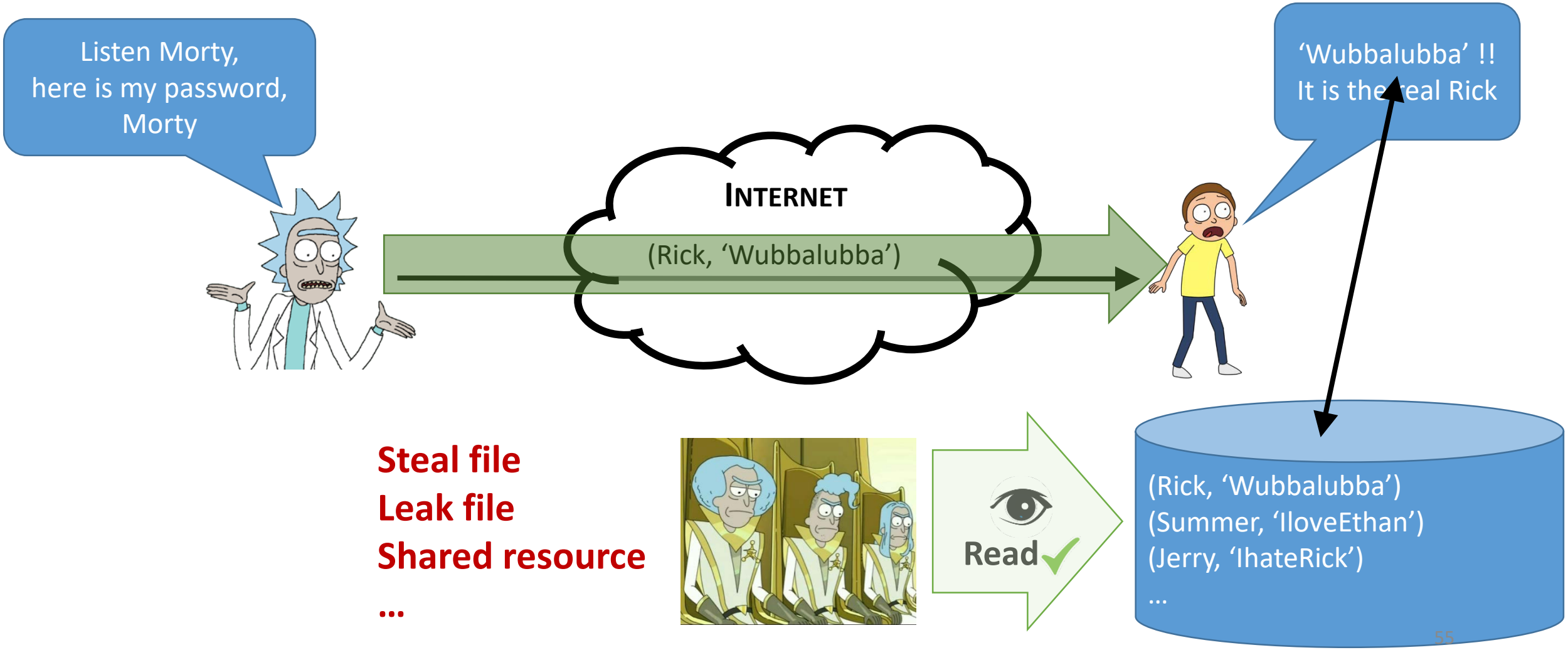
Replying login will not work!

Do NOT design your own authentication protocol

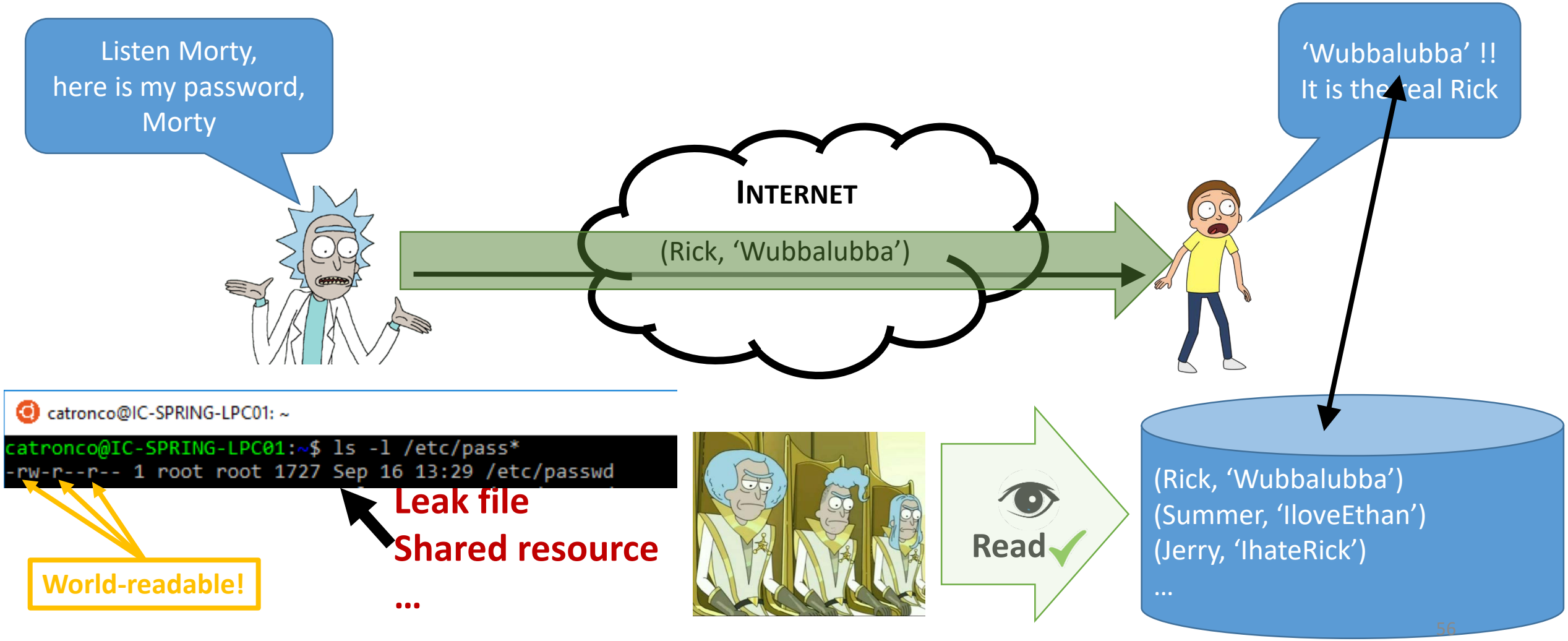
Secure storage



Secure storage



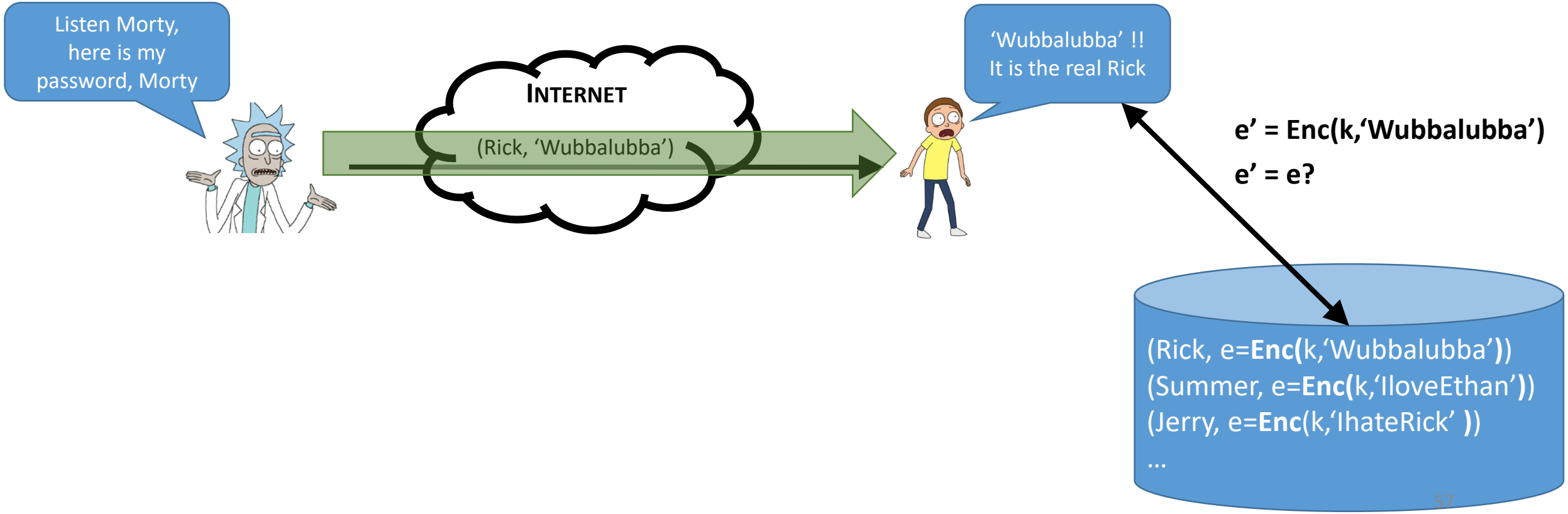
Secure storage



Secure storage - Do not store in the clear!

OPTION 1

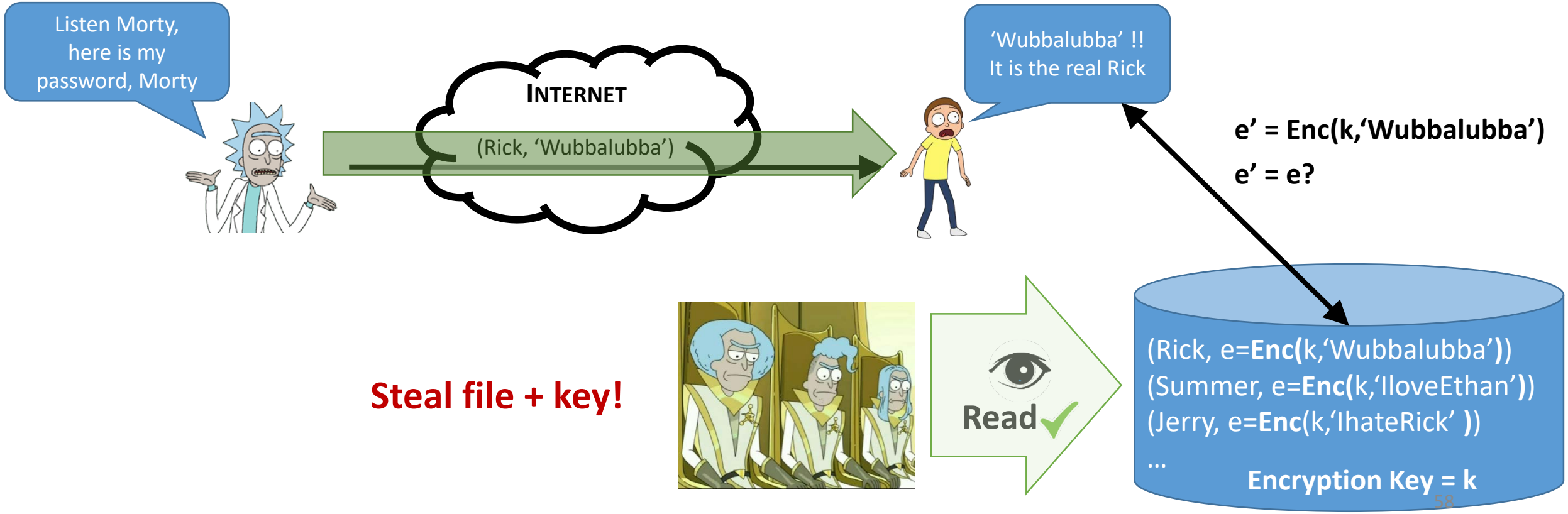
Store password encrypted



Secure storage - Do not store in the clear!

OPTION 1

Store password encrypted



Secure storage - Do not store in the clear!

OPTION 2

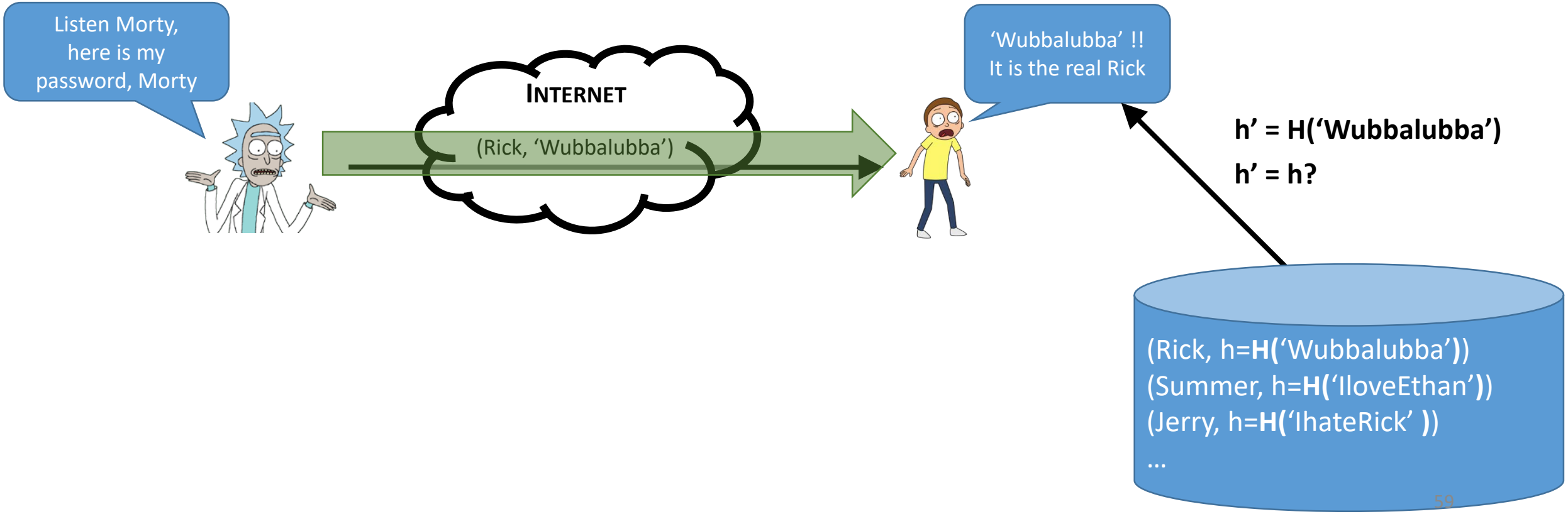
Store password as a "hash" of its value

Refresher

PRE-IMAGE RESISTANCE
Given H(m), difficult to get m

SECOND PRE-IMAGE RESISTANCE
Given H(m), difficult to get an m' such that H(m') = H(m)

COLLISION RESISTANCE
Difficult to find any m, m' such that H(m) = H(m')



Secure storage - Do not store in the clear!

Refresher

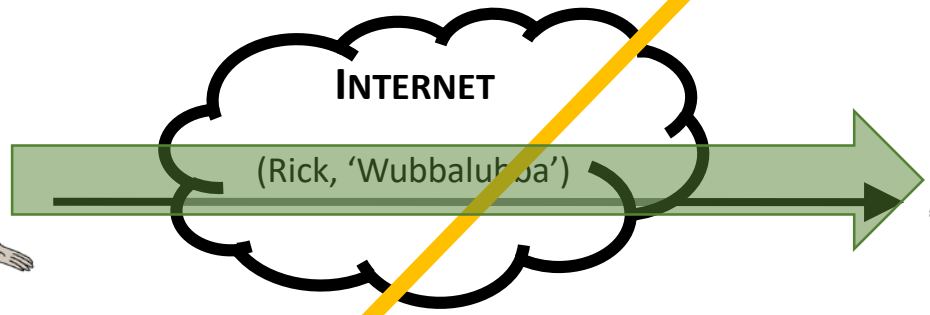
PRE-IMAGE RESISTANCE
Given H(m), difficult to get m

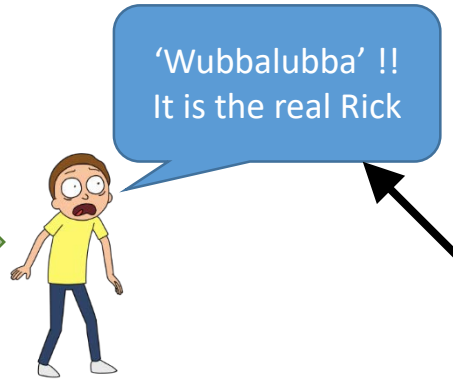
SECOND PRE-IMAGE RESISTANCE
Given H(m), difficult to get an m' such that H(m') = H(m)

~~**COLLISION RESISTANCE**~~
~~Difficult to find any m, m' such that H(m) = H(m')~~

OPTION 2

Store password as a "hash" of its value

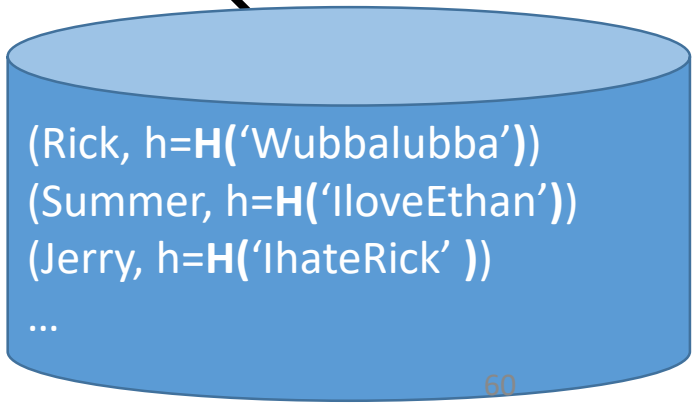




h' = H('Wubbalubba')

h' = h?

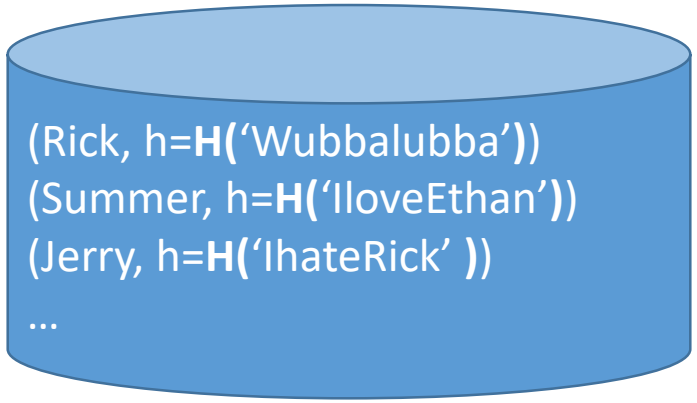
Cannot recover/produce valid password



Secure storage - Do not store in the clear!

OPTION 2

Store password as a “hash” of its value



OFFLINE ATTACKS – DICTIONARY ATTACK

Anyone can compute a hash

Passwords **not truly random**

- 52 upper- and lower-case letters, 10 digits and 32 punctuation symbols,
- 94^8 eight-character passwords (around 2^{52}) possibilities

Users use a **limited set** of passwords (reduced search space)

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

https://resources.infosecinstitute.com/10-popular-password-cracking-tools/#gref

Secure storage - Do not store in the clear!

OPTION 2

Store passwo

Kanye West accidentally reveals the code to unlock his iPhone is '000000' during Trump visit

(Rick, h=**H(**'Wubba
(Summer, h=**H(**'Ilov
(Jerry, h=**H(**'IhateR
…

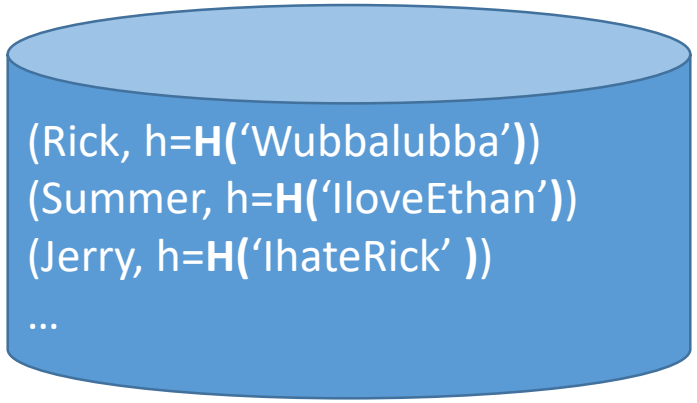
ts and 32 punctuation symbols,
^{52}) possibilities
educed search space)

https://en.wikipedia.org/wiki/List_of_the_most_common_passwords
https://resources.infosecinstitute.com/10-popular-password-cracking-tools/#gref

Secure storage - Do not store in the clear!

OPTION 2

Store password as a "hash" of its value



OFFLINE ATTACKS – DICTIONARY ATTACK

Anyone can compute a hash

Passwords **not truly random**

- 52 upper- and lower-case letters, 10 digits and 32 punctuation symbols,
- 94^8 eight-character passwords (around 2^{52}) possibilities

Users use a **limited set** of passwords (reduced search space)

Attacker can compute H(word) for every word in the dictionary and see if the result is in the password file!

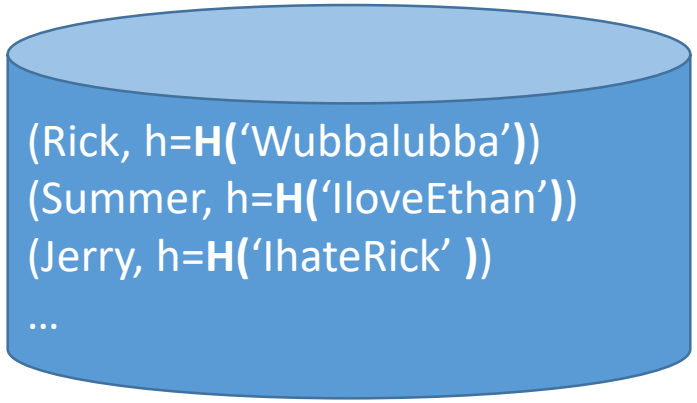
https://en.wikipedia.org/wiki/List_of_the_most_common_passwords

https://resources.infosecinstitute.com/10-popular-password-cracking-tools/#gref

Secure storage - Do not store in the clear!

OPTION 2

Store password as a "hash" of its value



OFFLINE ATTACKS – DICTIONARY ATTACK

Anyone can compute a hash

Passwords **not truly random**

- 52 upper- and lower-case letters, 10 digits and 32 punctuation symbols,
- 94^8 eight-character passwords (around 2^{52}) possibilities

Users use a **limited set** of passwords (reduced search space)

Attacker can compute H(word) for every word in the dictionary and see if the result is in the password file!

- Can **reuse** the dictionary
- **Parallel cracking** with GPU accelerates search
- Other tricks: rainbow tables, pre-computation,…

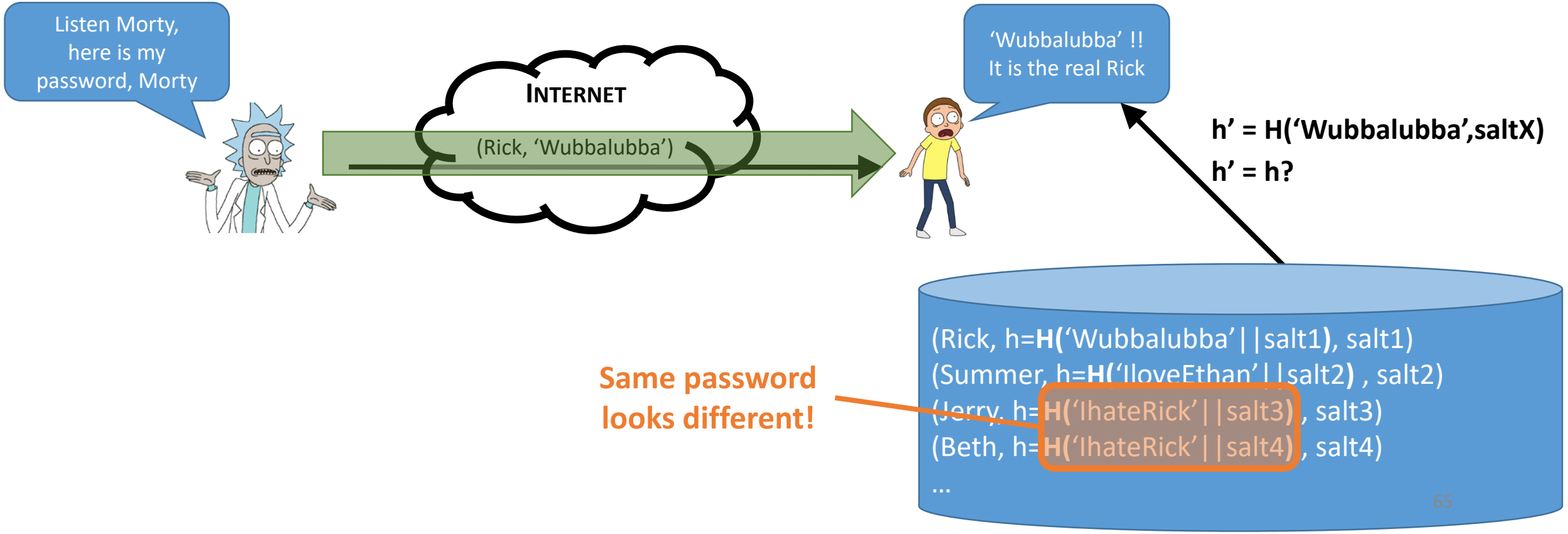
https://en.wikipedia.org/wiki/List_of_the_most_common_passwords

https://resources.infosecinstitute.com/10-popular-password-cracking-tools/#gref

Secure storage – Do this!

OPTION 3

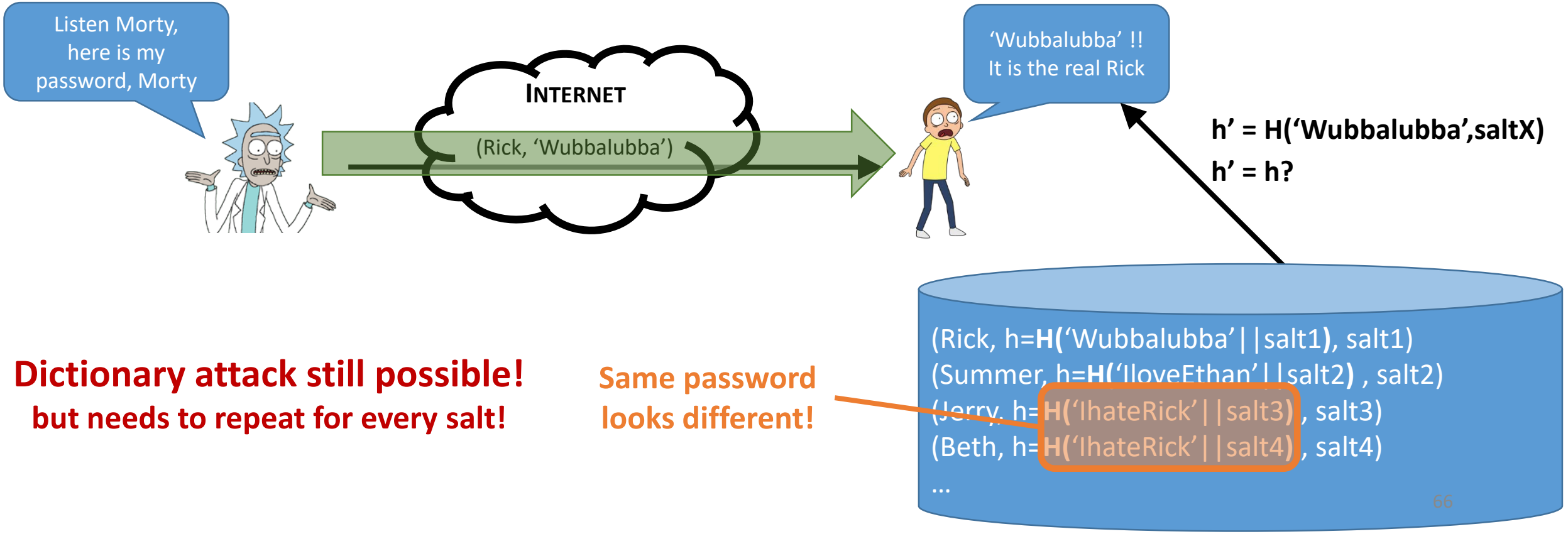
Store password as a “hash”+ “salt”



Secure storage – Do this!

OPTION 3

Store password as a “hash”+ “salt”



Secure storage – Do this!

OPTION 3

Store password as a “hash”+ “salt”

COMPLEMENTARY DEFENSES

- Use of hash functions designed to be **slow** (bcrypt, scrypt, argon2)
  Repeat several times (e.g., 1000)

Secure storage – Do this!

OPTION 3

Store password as a “hash”+ “salt”

COMPLEMENTARY DEFENSES

- Use of hash functions designed to be **slow** (bcrypt, scrypt, argon2)
  Repeat several times (e.g., 1000)
- Require specific elements in passwords
  Increase entropy

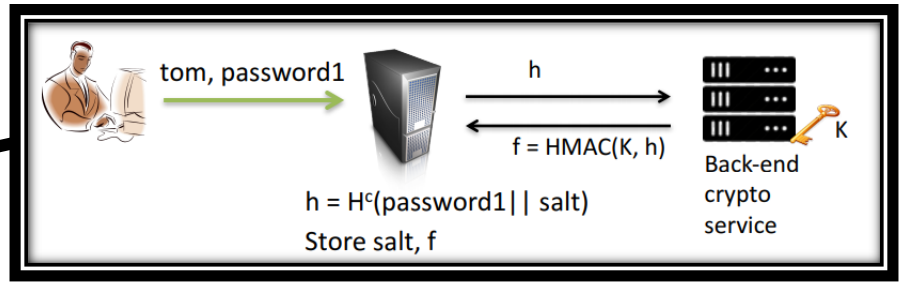
Secure storage – Do this!

OPTION 3

Store password as a “hash”+ “salt”

COMPLEMENTARY DEFENSES

- Use of hash functions designed to be **slow** (bcrypt, scrypt, argon2)
  Repeat several times (e.g., 1000)
- Require specific elements in passwords
  Increase entropy
- Split check, require a second server
  Invalidate offline attacks



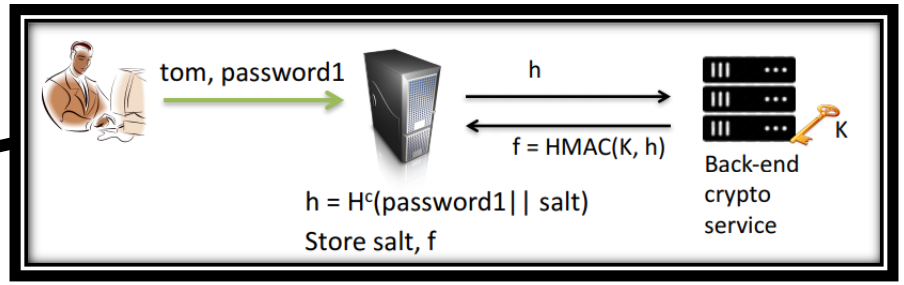
Secure storage – Do this!

OPTION 3

Store password as a “hash”+ “salt”

COMPLEMENTARY DEFENSES

- Use of hash functions designed to be **slow** (bcrypt, scrypt, argon2)
  Repeat several times (e.g., 1000)
- Require specific elements in passwords
  Increase entropy
- Split check, require a second server
  Invalidate offline attacks



- Access control! (`/etc/shadow` in UNIX only accessible by root)

Password database compromises

⋮	year	# stolen	% recovered	format
rockyou	2012	32.6 million	100%	plaintext (!)
LinkedIn	2012	117 million	90%	Unsalted SHA-1
Adobe®	2013	36 million	??	ECB encryption
YAHOO!	2014	~500 million	??	bcrypt + ??
ASHLEY MADISON® Life is short. Have an affair.®	2015	36 million	33%	Salted bcrypt + MD5

Source: Tom Ristenpart

Facebook password onion

```
$cur  = 'password'
$cur  = md5($cur)
$salt = randbytes(20)
$cur  = hmac_sha1($cur, $salt)
$cur  = remote_hmac_sha256($cur, $secret)
$cur  = scrypt($cur, $salt)
$cur  = hmac_sha256($cur, $salt)
```

Why this onion?

Source: Tom Ristenpart

Facebook password onion

```
$cur  = 'password'
$cur  = md5($cur)
$salt = randbytes(20)
$cur  = hmac_sha1($cur, $salt)
$cur  = remote_hmac_sha256($cur, $secret)
$cur  = scrypt($cur, $salt)
$cur  = hmac_sha256($cur, $salt)
```

Why this onion?

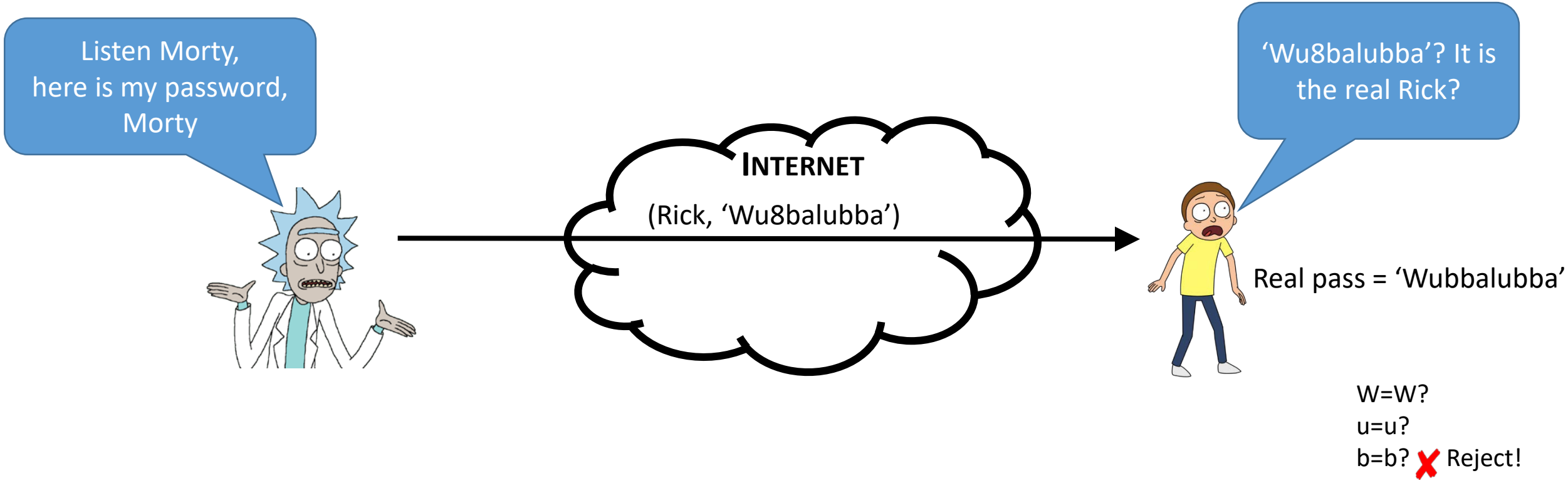
Coping with legacy!

Source: Tom Ristenpart

Secure checking

OPTION 1

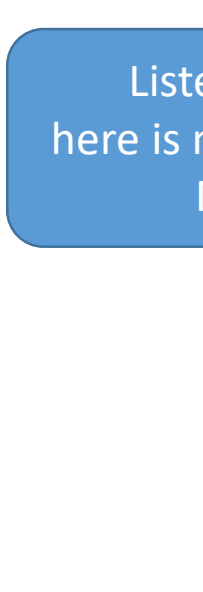
Check letter by letter



Secure checking

OPTION 1

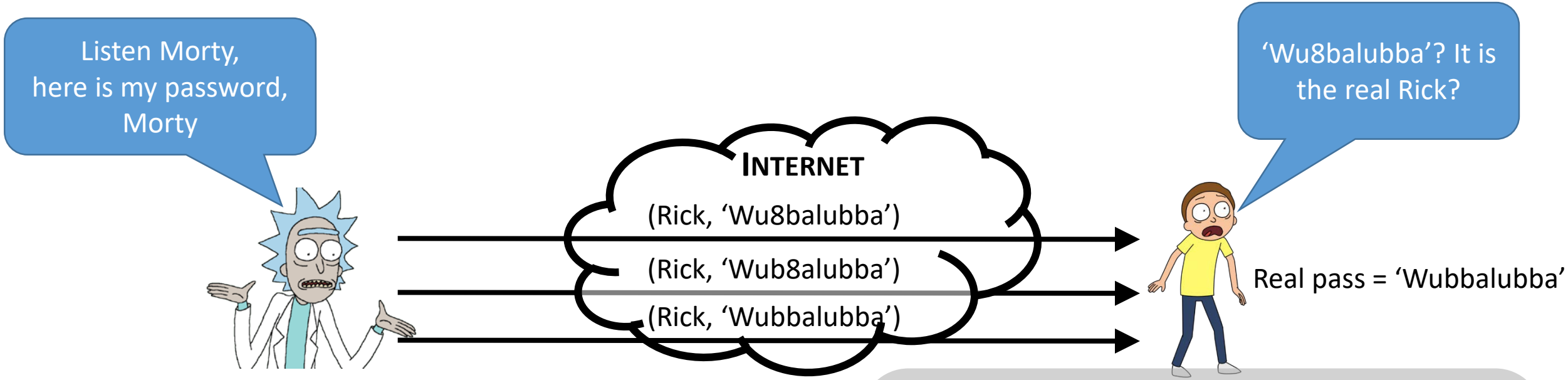
Check letter by letter



Secure checking

Option 1

Check letter by letter



Timing channel leaks information about the password

W=W?
u=u?
b=b?
b=b?
a=a?
…
b=b?
a=a? ✓ Accept!

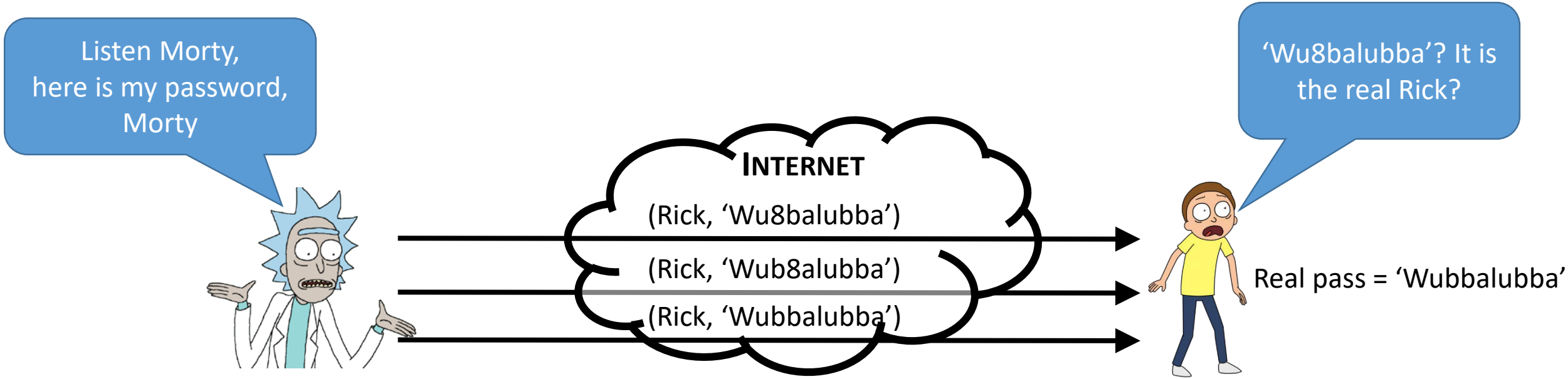
W=W?
u=u?
b=b?
8=b? ✗ Reject!

W=W?
u=u?
b=b? ✗ Reject!

Secure checking

Option 2

Always check everything



W=W?
u=u?
b=b?
b=b?
a=a?
…
b=b?
a=a? ✓ Accept!

W=W?
u=u?
b=b?
8=b?
a=a?
…
b=b?
a=a? ✗ Reject!

W=W?
u=u?
8=b?
b=b?
a=a?
…
b=b?
a=a? ✗ Reject!

Hashing will take care of equal operations!
Require same operations for accept and reject!!

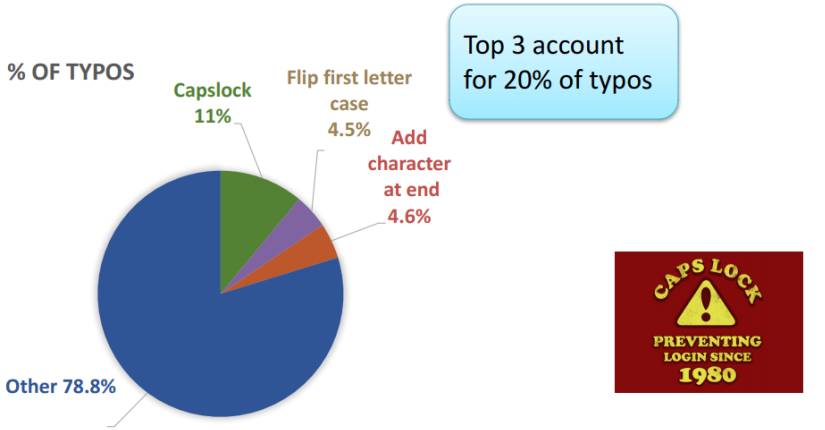
Secure checking – WHAT ABOUT ERRORS…?

Source: Tom Ristenpart

HASHES COMPLICATE ERROR CORRECTION!

Mechanical Turk transcription study

100,000+ passwords typed by 4,300 workers



Top 3 account for 20% of typos

Impact of Top 3 typos in real world

Instrumented production login of Dropbox to quantify typos

NOTE: We did not admit login using typo'd passwords

24 hour period:

- **3% of all users** failed to login due to one of top 3 typos
- 20% of users who made a typo would have saved at least 1 minute in logging into Dropbox if top 3 typos are corrected.

Allowing typos in password will add several person-months of login time every day.

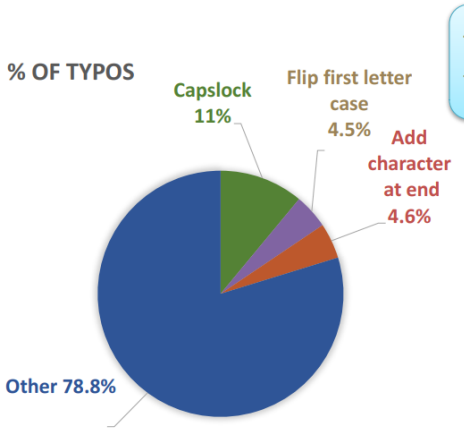
Secure checking – WHAT ABOUT ERRORS…?

Source: Tom Ristenpart

HASHES COMPLICATE ERROR CORRECTION!

Mechanical Turk transcription study

100,000+ passwords typed by 4,300 workers



Top 3 account for 20% of typos

Impact of Top 3 typos in real world

Dropbox

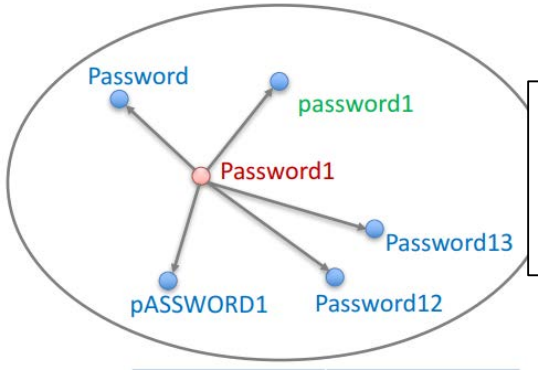
Instrumented production login of Dropbox to quantify typos

NOTE: We did not admit login using typo'd passwords

24 hour period:

- **3% of all users** failed to login due to one of top 3 typos
- 20% of users who made a typo would have saved at least 1 minute in logging into Dropbox if top 3 typos are corrected.

Allowing typos in password will add several person-months of login time every day.



Check also for typical errors

https://typtop.info/

Security must be taken into account when deciding how

Problems with passwords

Strong passwords are difficult to remember

- Written passwords
- Reuse across systems

Can be stolen

- Keylogger
- Shoulder surfing
- Phishing
- Social engineering

Problems with passwords

Strong passwords are difficult to remember

- Written passwords
- Reuse across systems

Can be stolen

- Keylogger
- Shoulder surfing
- Phishing
- Social engineering

Jul 6, 2017, 10:10am

Help! Hackers Stole My Password Just By Listening To Me Type On Skype!

Thomas Brewster Forbes Staff
Security
I cover crime, privacy and security in digital and physical forms.

For many, everyday life involves sitting in front of a computer typing endless emails, presentation documents and reports. Then there's the frequent typing of passwords just to get access to those files. But beware: researchers have hacked together a tool that can harvest what's being typed simply by listening to the sounds of the keys.

They've created the Skype&Type program for snooping on Skype

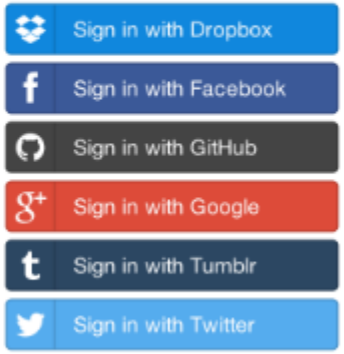
Authentication library

Dedicated security frameworks.

Embedded authentication libraries in web frameworks.

Cross-platform authentication libraries.

OAuth: performs the authentication in a third-party.

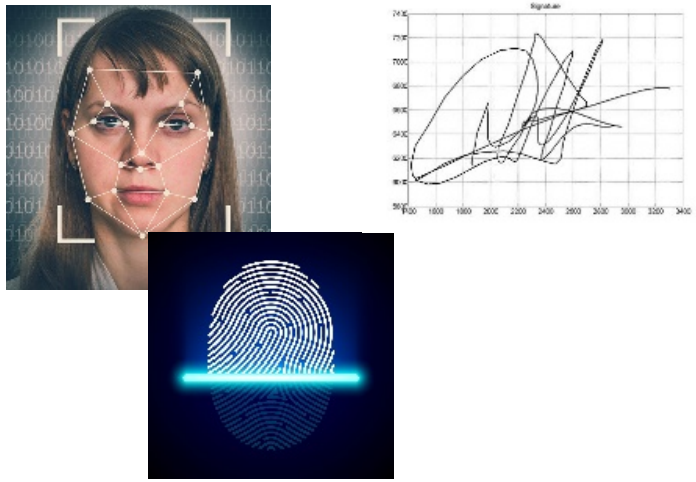


What you are: Biometrics

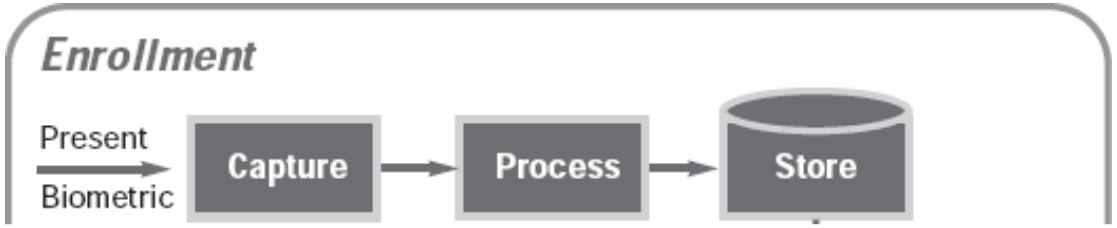
BIOMETRICS
is the measurement and statistical analysis of people's unique physical characteristics *(modern: also behavioral)*

Popular biometrics
- Fingerprint, face recognition, retina, voice, handwritten signature, DNA

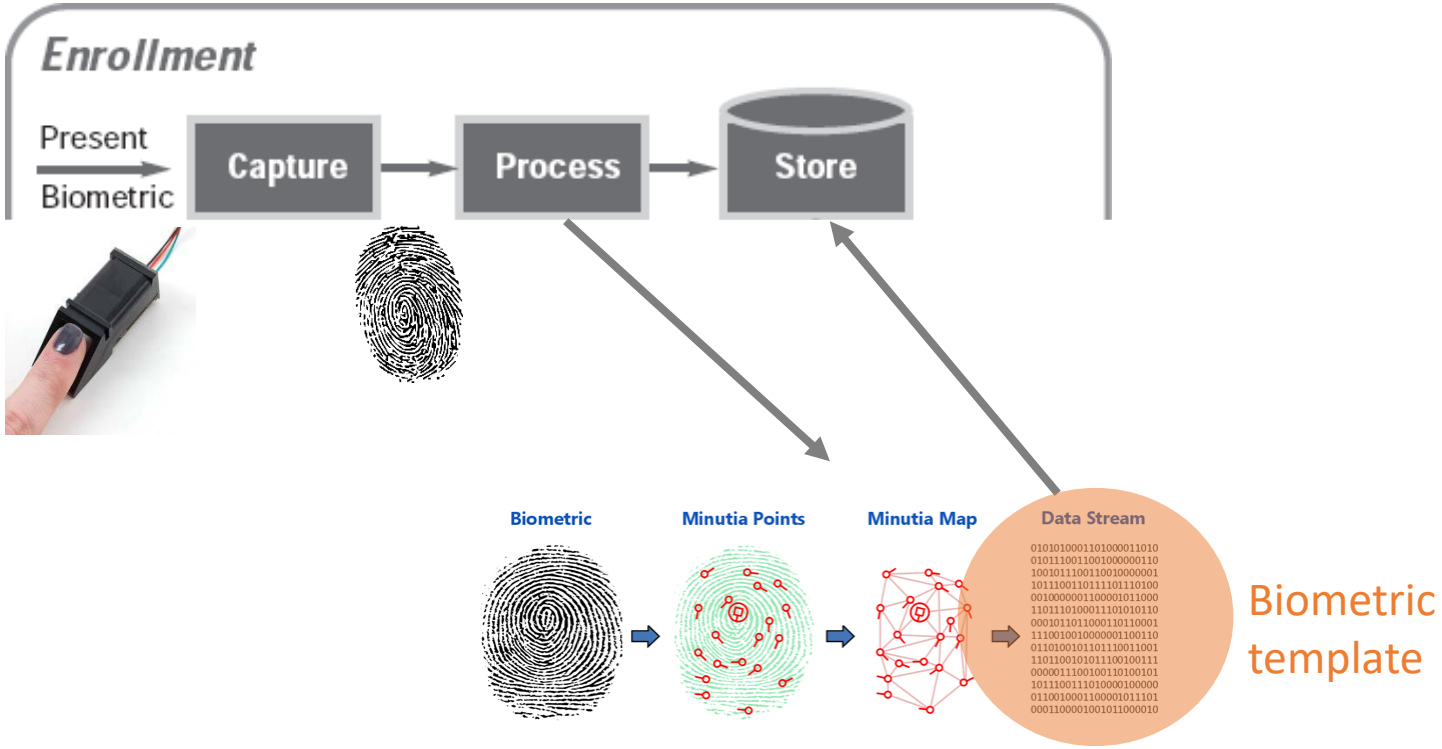
Advantages
- Nothing to remember
- Passive
- Difficult to delegate
- If method is very accurate, they are unique



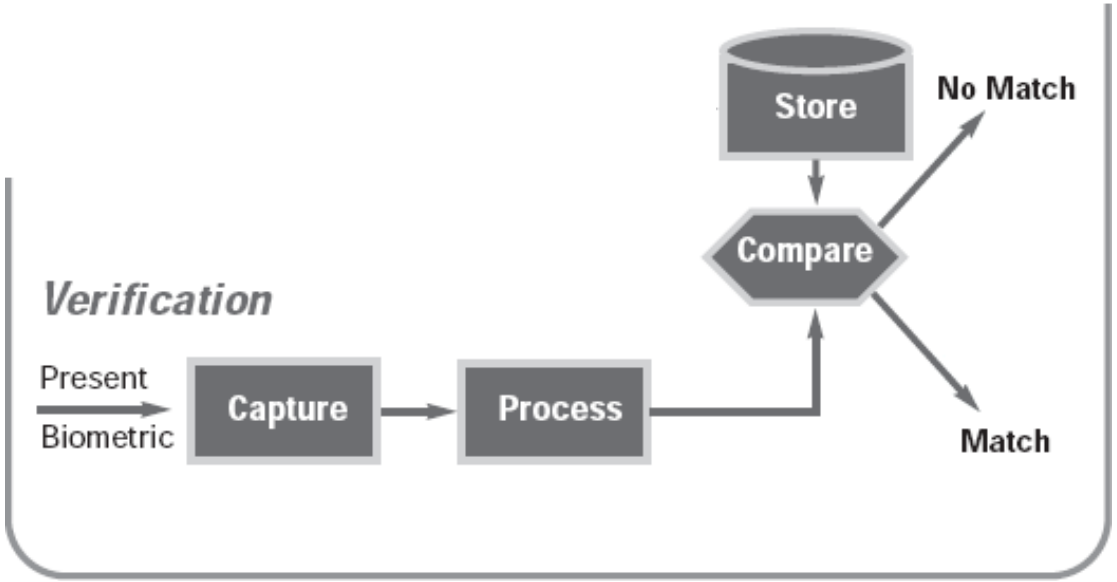
What you are: Biometrics



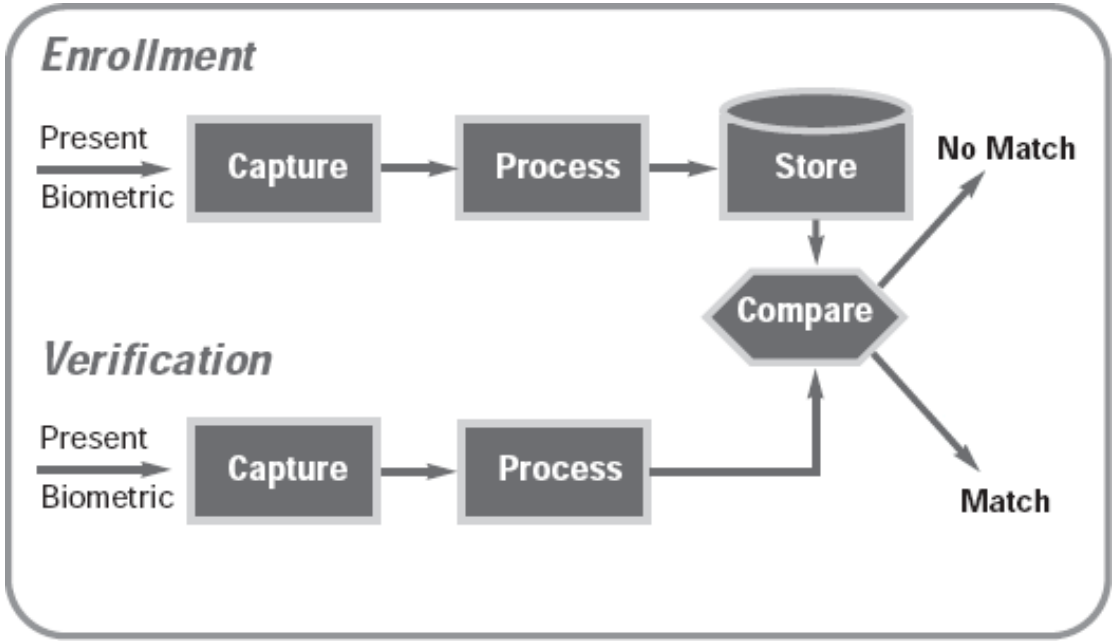
What you are: Biometrics



What you are: Biometrics



What you are: Biometrics

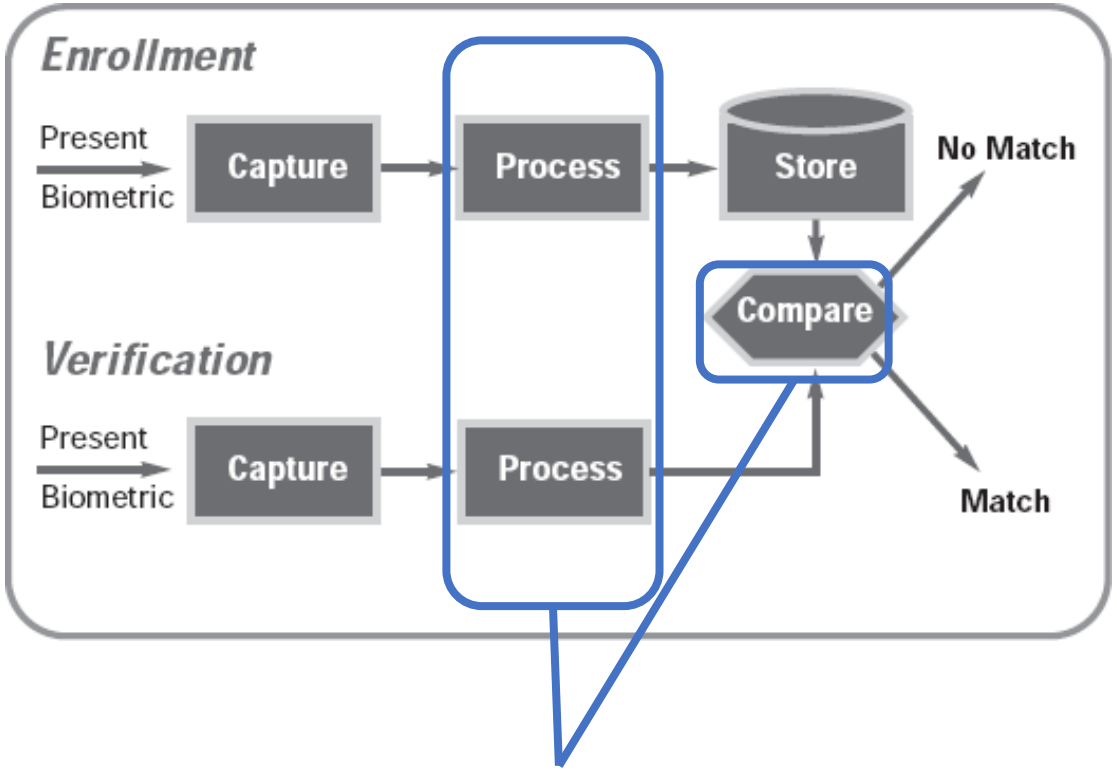


WHERE DO THESE PROCESSES HAPPEN?

CAPTURE	PROCESS	STORE	
Local	Local	Local	Fingerprint on a smartphone
Local	Local	Remote	Fingerprint on a door
Local	Remote	Remote	Fingerprint on a door

The configurations provide different security / privacy tradeoffs!

What you are: Biometrics

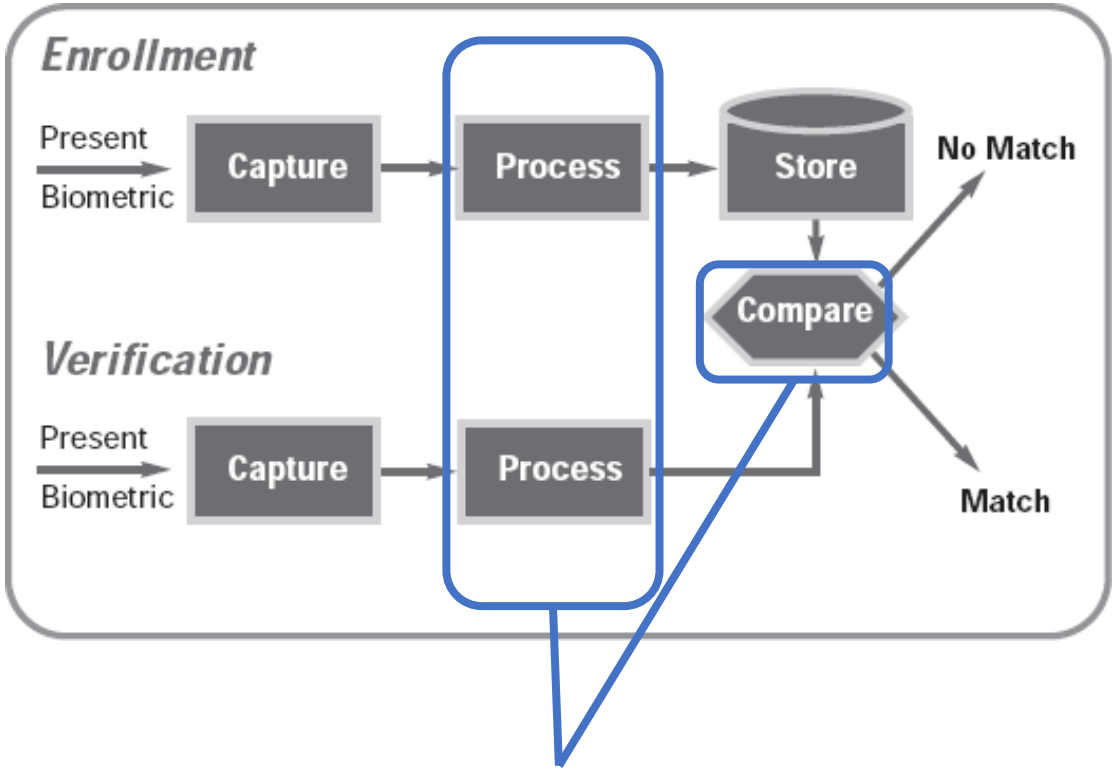


Parameters determine
false positives and false negatives

Wrong authentications accepted

True authentications rejected

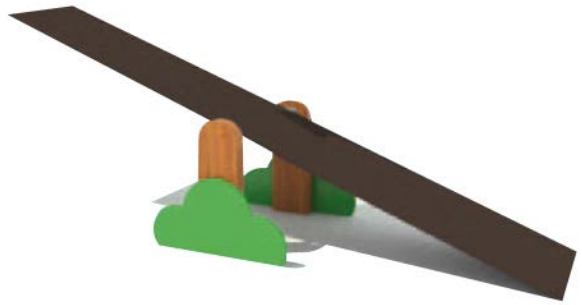
What you are: Biometrics



Parameters determine
false positives and false negatives

Wrong authentications accepted

True authentications rejected



Decreasing false negatives increases false positives!!

Configuration depends on applications

Bank: low false positive even if legitimate users need to repeat

Gym: low false negative even if some non-users get in

Problems with Biometrics

Hard to keep secret

Signature on ID card
Left on glasses, door handle, ...
Photos (nowadays, everywhere!)

Revocation is difficult (impossible?)

Sorry, your iris has been compromised, please create a new one... !

Identifiable and unique

Linking across systems

May reveal private information

Iris → disease
Face → identity

Not always universal or immutable

Fingerprints disappear, iris changes with lenses,…

Problems with Biometrics

Hard to keep secret

Signature on ID card
Left on glasses, door handle, ...
Photos (nowadays, everywhere!)

} **Liveness detection**

Revocation is difficult (impossible?)

Sorry, your iris has been compromised, please create a new one... !

Identifiable and unique

Linking across systems

May reveal private information

Iris → disease
Face → identity

Not always universal or immutable

Fingerprints disappear, iris changes with lenses,…

Problems with Biometrics

Hard to keep secret

Signature on ID card
Left on glasses, door handle, ...
Photos (nowadays, everywhere!)

Liveness detection

Revocation is difficult (impossible?)

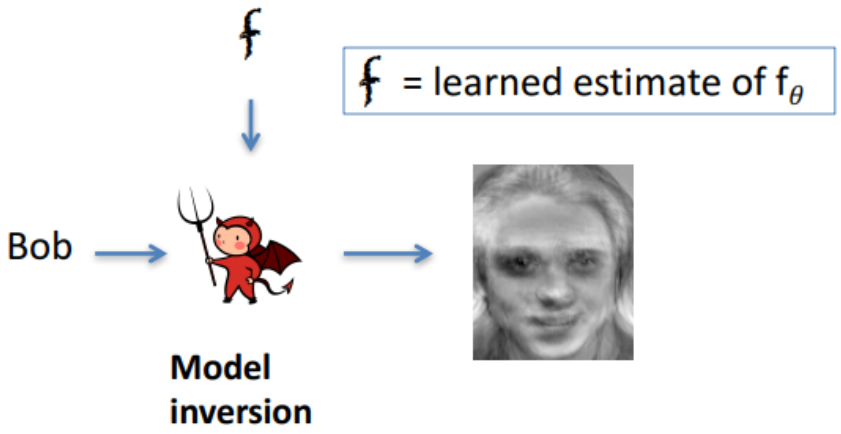
Sorry, your iris has been compromised, please create a new one... !

Identifiable and unique

Linking across systems

May reveal private information

Iris → disease
Face → identity



Source: Tom Ristenpart

Not always universal or immutable

Fingerprints disappear, iris changes with lenses, voice changes with a cold…

What you have: Tokens

What you have: Tokens

Contains a secret key to sign
(+ complex protocol)

Key shared token & server
Time synchronized with server
Algorithm obtains unique number key+time

What you have: 2FA – Two factor authentication

+ PIN = Two factor authentication

+ personal identification number = Two factor authentication

+ card = Two factor authentication

What you have: Secret key

Digital signatures can be used to authenticate parties

e.g., used in internet protocols HTTPS/TLS to authenticate the server

(and sometimes the client)

Building authentication protocols **is hard**! (see extra slides for examples)

defending from man in the middle

defending from replay attacks

Use well established protocols!! (TLS 1.3, ISO 9798-3)

Don't design your own

Summary of the lecture

AUTHENTICATION

The process of verifying a claimed identity

What you know

Passwords: how to store them securely

What you are

Biometrics: require tradeoffs, bring problems

What you have

Tokens, keys: require careful design of protocols!!

Beware of reply attacks!

Extra slides

(NOT FOR EXAM, JUST FOR FUN)

Problem with Diffie Hellmann

Secret “b”

g^b →

← g^a

Derive: $k = g^{ab}$

Secret “a”

Man in the middle is possible!!

What you know: A secret Key

PUBLIC

g, p, Cert{, sk }, Cert{ , sk }

Secret "b"

g^b →

← g^a

Derive: $k = g^{ab}$

Secret "a"

Man in the middle!!

What you know: A secret Key Station to Station protocol

PUBLIC

g, p, Cert{, sk }, Cert{ , sk }

Secret "b"
sk

Secret "a"
sk

g^b →

← g^a, $Enc(k, Sign_{sk}[g^a, g^b])$

$Enc(k, Sign_{sk}[g^a, g^b])$ →

Derive: $k = g^{ab}$

Authentic receiver but no freshness, keys can be replayed

What you know: A secret Key
ISO 9798-3

PUBLIC

g, p, Cert{, sk }, Cert{ , sk }

Secret "b"
sk
Random Nb

Nb, , g^b →

← Na, Nb, , g^a, $Sign_{sk}$[Na, Nb, g^a, g^b,]

Na, Nb, $Sign_{sk}$[Na, Nb, g^a, g^b,] →

Secret "b"
sk
Random Nb

Derive: K = H(g^{ab}, Na, Nb, A, B)

K Different every time!

Diffie-Hellman
Freshness
Authenticity

What you know: A secret Key
ISO 9798-3

PUBLIC

g, p, Cert{, sk }, Cert{ , sk }

Don't design your own authentication protocol

Secret "b"
Random "Nb"

Secret "a"
Random "Na"

Nb, , g^b →

← Na, Nb, , g^a, $Sign_{sk}$[Na, Nb, g^a, g^b,]

Na, Nb, $Sign_{sk}$[Na, Nb, g^a, g^b,] →

Derive: K = H(g^{ab}, Na, Nb, A, B)

K Different every time!

Diffie-Hellman
Freshness
Authenticity

The protocol also has problems:
Communication identities public
State is kept (remembering N)→DoS