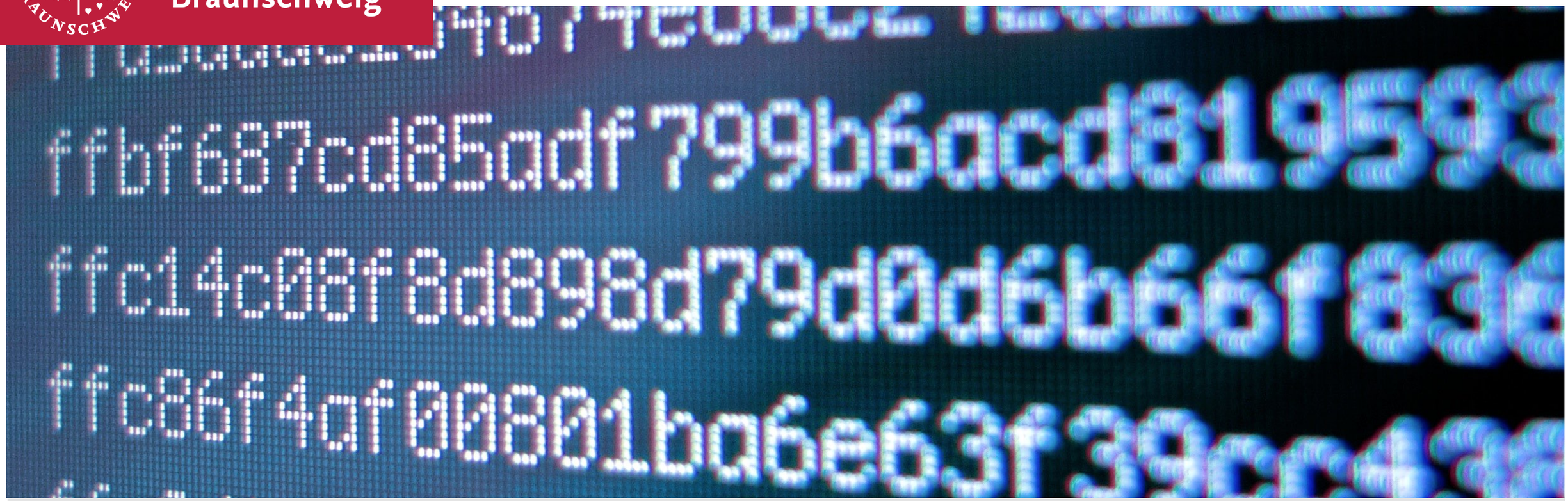
Technische Universität Braunschweig

IAS | INSTITUTE FOR APPLICATION SECURITY



Hybrid Cryptosystems

Vorlesung “Einführung in die IT-Sicherheit”

Prof. Dr. Martin Johns

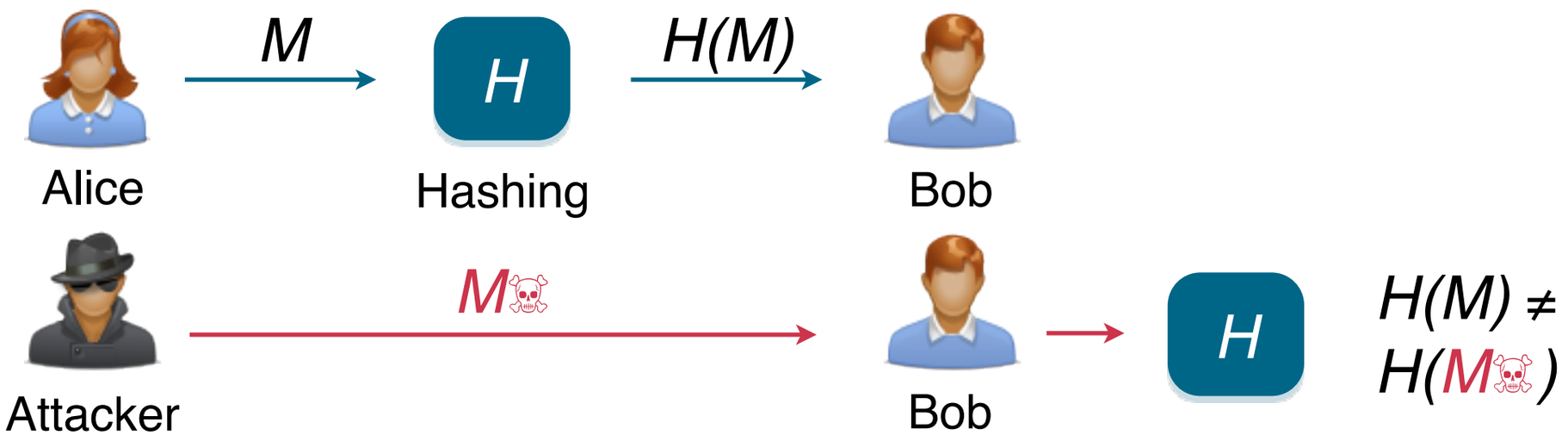
Overview

- **Topic of the unit**
  - Hybrid Cryptosystems
- **Parts of the unit**
  - Part #1: Cryptographic hash functions
  - Part #2: Digital signatures
  - Part #3: Hybrid cryptosystems
  - Part #4: SSL & PGP

Cryptographic Hash Functions

- **Cryptographic hash function** H(M) = D
  - "Hashes" message M to a fixed number of bits
  - Alternative names: message digest, fingerprint
- **One-way property**
  - Given message M: H(M) easy to compute
  - Given hash value D: hard to find M with H(M) = D
- **Keyless and keyed hash functions**
  - Hashing with key: $H_K(M) = D$

Applications of Hash Functions



- **Common application of cryptographic hash functions**
  - Checksums for large messages, e.g. DVD images
  - Example: download of DVD image from mirror sites
- Further applications in public-key cryptography

Birthday Attack

- **The Birthday Problem**
  - Group of people in a room
  - Probability of two people sharing the same birthday?
  - Surprising result: for 23 people already 50%
- **Where is the trick?**
  - Any person is matched against any other
  - Quadratic number of pairs to consider
  - Collisions more likely for pairwise matching

Collision Resistance

- **Weak collision resistance**
  - Computationally infeasible to find M’ for a given M such that M ≠ M’ and H(M) = H(M’)
  - Birthday attack possible
- **Strong collision resistance**
  - Computationally infeasible to find a pair M and M’ such that M ≠ M’ and H(M) = H(M’)
  - Birthday attack not possible

Secure Hash Algorithm

- **Group of standardized hash functions**
  - First version published by NIST in 1994
  - Extension SHA-2 published in 2002
  - Extension SHA-3 (Keccak family) published in 2015
- **Design of SHA-2**
  - Common hash sizes: 256 and 512 bits
  - Rounds: 64 and 80 depending on hash size
  - Network of arithmetic and logic operations
- Severe known attacks against SHA-1, e.g. collisions

Secure Hash Algorithm

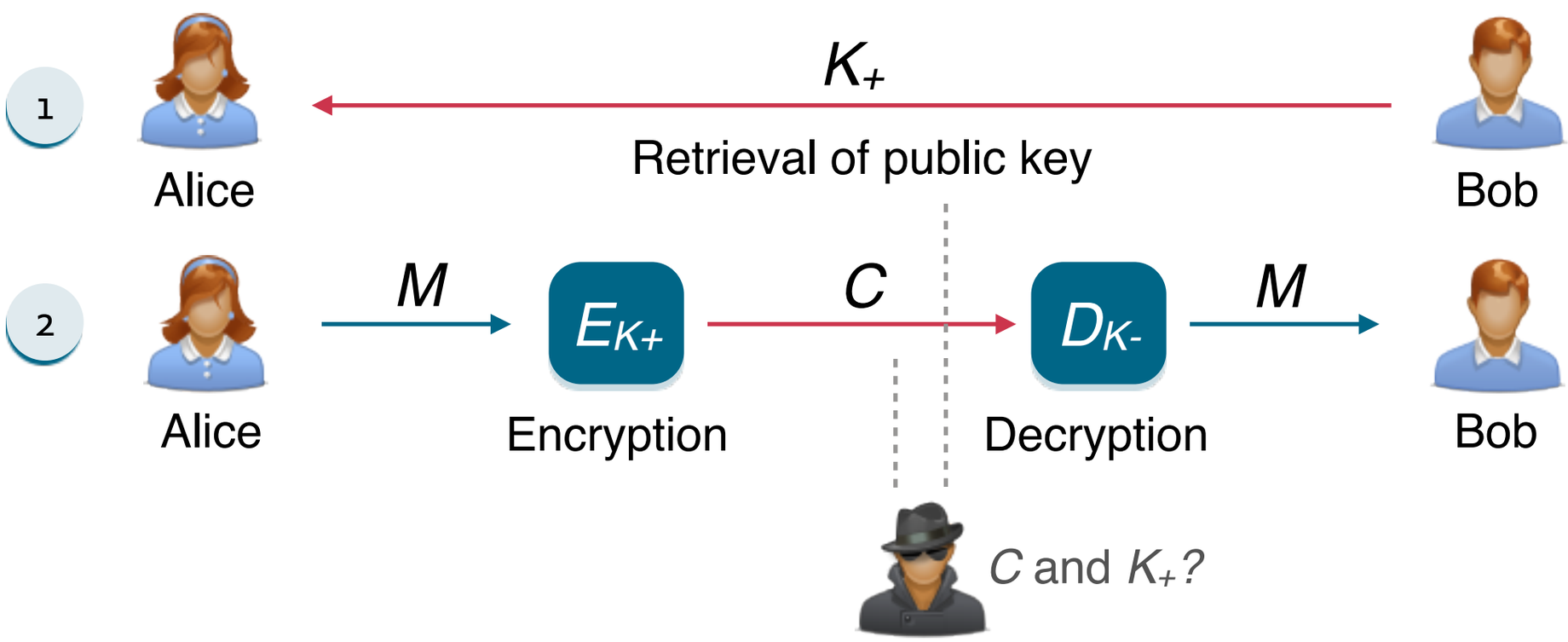
- **Group of standardized hash functions**
  - First version published by NIST in 1994
  - Extension SHA-2 published in 2002
  - Extension SHA-3 (Keccak family) published in 2015
- **Design of SHA-2**
  - Common hash sizes: 256 and 512 bits
  - Rounds: 64 and 80 depending on hash size
  - Network of arithmetic and logic operations
- Severe known attacks against SHA-1, e.g. [illegible]

```python
# Python
from Crypto.Hash import SHA256
hash = SHA256.new(msg)
hash.digest()
```

Overview

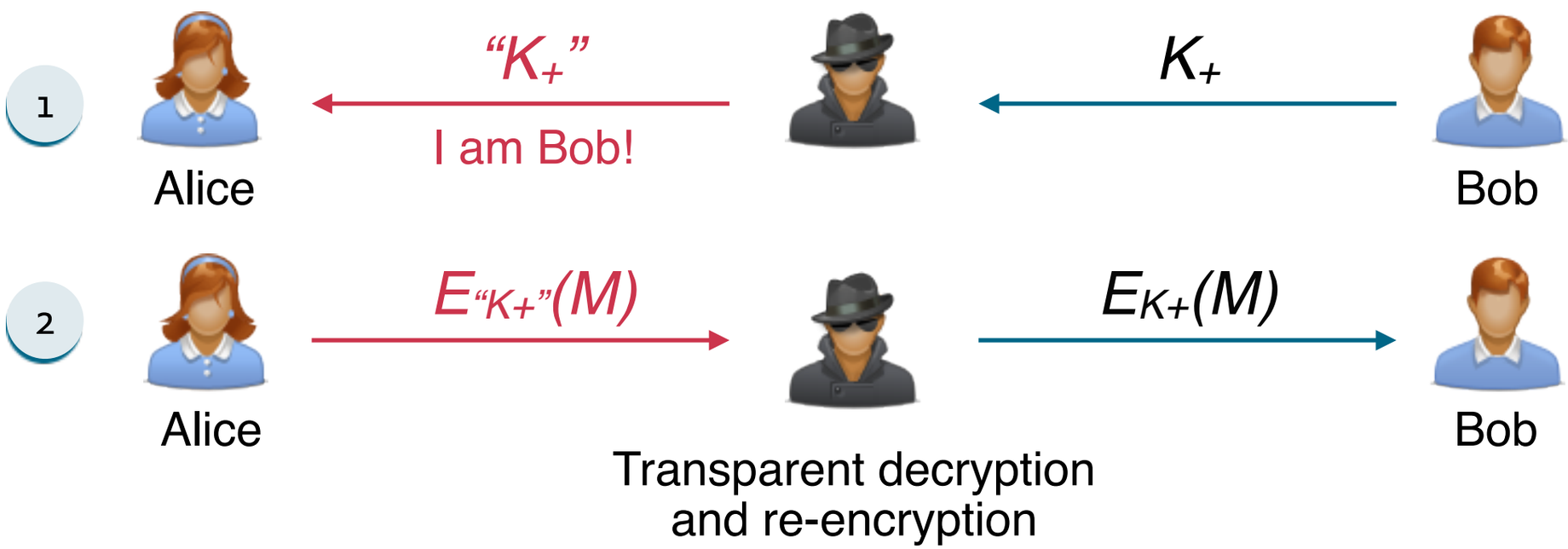
- **Topic of the unit**
  - Hybrid Cryptosystems
- **Parts of the unit**
  - Part #1: Cryptographic hash functions
  - Part #2: Digital signatures
  - Part #3: Hybrid cryptosystems
  - Part #4: SSL & PGP

Key Exchange with Public Keys



- **Asymmetric Cipher**
  - K_+ = public key of Bob K_- = private key of Bob
  - No exchange of shared key necessary

Man-in-the-Middle Attacks (MITM)



- **Common attack against asymmetric cryptosystems**
  - Interception of public key exchange by attacker
  - Transparent eavesdropping using forged keys

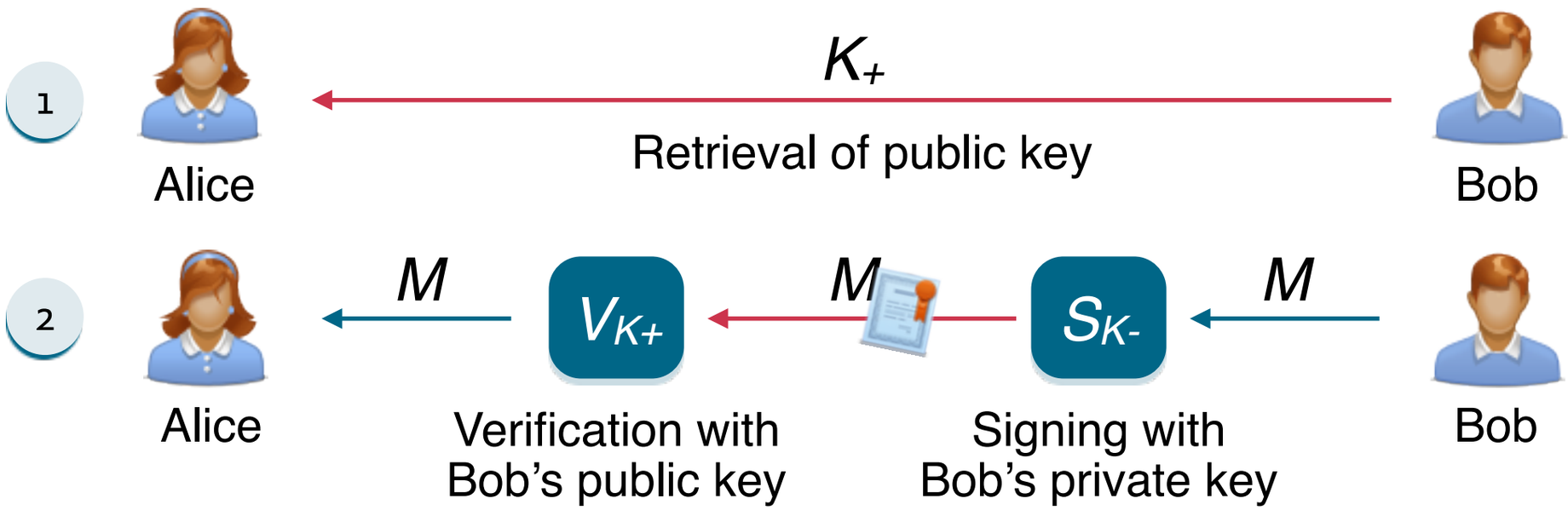
Key Fingerprints

- **Protection against MITM using key fingerprints (hashes)**
  - Manual comparison of public keys using hash values
  - Storage of approved public keys in database
  - Hen-egg problem: Secure exchange of fingerprints

- Example: **SSH client presents fingerprint for validation**

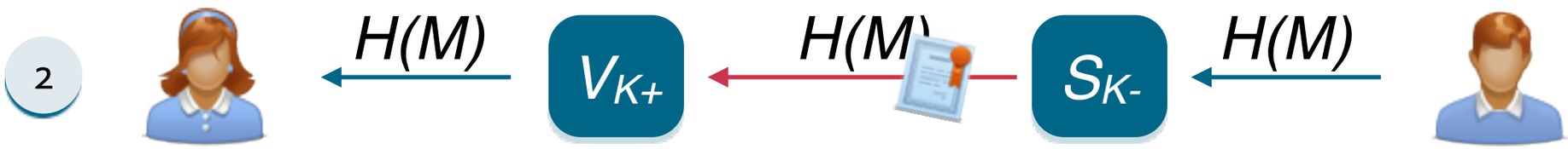
```
$ ssh root@life.ncsc.mil
The authenticity of host ‘life.ncsc.mil (144.51.1.10)’ can't be established.
RSA key fingerprint is 42:2d:21:ac:ce:1f:48:22:3f:21:53:a8:5f:27:0f:1a.
Are you sure you want to continue connecting (yes/no)?
```

Digital Signatures



- **Digital signing:** reverse application of public-key system
  - Signing = encryption with private key K_-
  - Verification = decryption with public key K_+

Signing and Hashing



- **Encryption and decryption of large messages inefficient**
  - Signing of hash H(M) instead of message M
  - One-way property: hard to find M' with H(M') = H(M)
  - Support for signing emails, images, videos, ...

Signatures and RSA

- **Signing and verification with RSA**
  - Signing with private key d: $s = H(M)^d \bmod n$
  - Verification with public key e: $H(M) = s^e \bmod n$
  - Security identical to regular encryption

- **Several other signature schemes exist**
  - Digital Signature Algorithm (DSA) by NIST
  - ElGamal and Schnorr signature scheme

Example: **PGP Signature**

Hash function

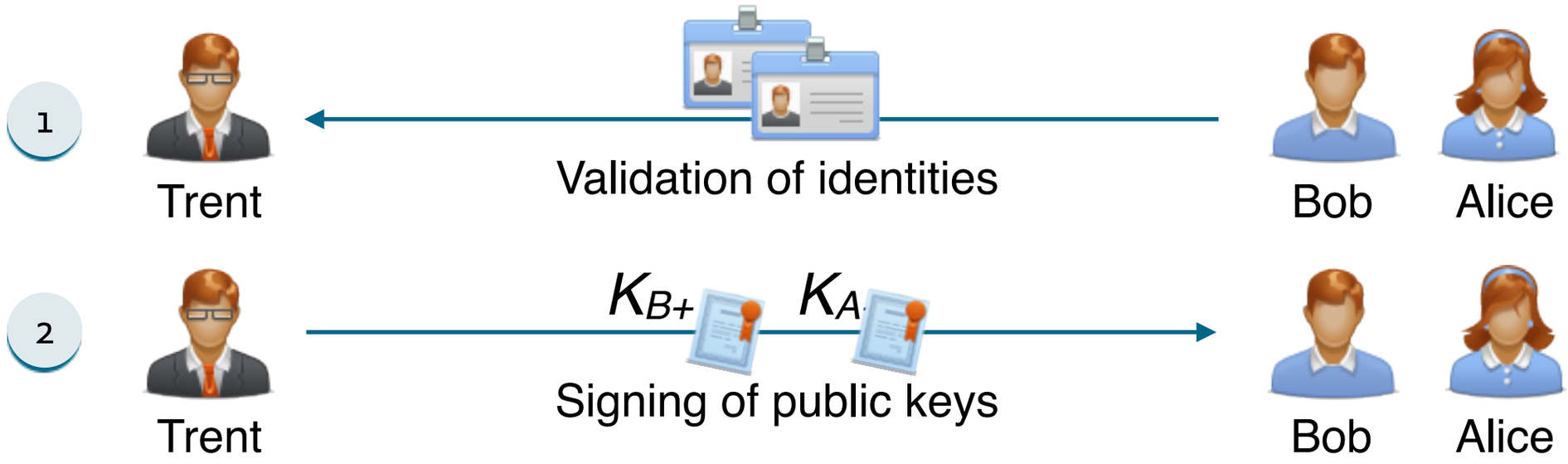
Signed hash

Message in cleartext

```
From: Christian Wressnegger <c.wressnegger@tu-bs.de>
Subject: Re: Einladung zum Projekttreffen
To: Konrad Rieck <k.rieck@tu-bs.de>

-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA256

Jo, wie vorgeschlagen im Lindner?

-----BEGIN PGP SIGNATURE-----
Version: GnuPG v2

iQIcBAEBCAAGBQJXAmNSAAoJEPmJxtgQxaKJw8IP/R6V1Kp8VgY1VUo/w9nI02Hv
T3Gim92BJnYXJQBXabU+VdWWMaQPn6Vlz8HbxYwC/V7APNlhAEJcI5cs9J355zX3
Lc9xEfd9EM/nJFUawjKcQtu/uJCIvgRcWaV7/JLdD0jcH8CSCJv9z37lTmVpVmuM
LKhpClSsZomG3A8qjz9VEpVFcfy3Fn9k2onRGFOknMQ1KkwSWktI4DuCYQLrd7Cr
5D8d1juAxc6GLMG/wz4pbIEiqFqEkGblKKEtl4DJh2bH7zG6777pAGsmdBBJhd22
BvSoLGBr3r4aeknkCKitwV2DvBb2hU0zXyGLzlp19kQw2YtaYB1oDa+M6ALSJlin
t3ISTul13/zVZOj4ftAuoPCyKvaYsUwJn0Ix8CsKFMSVdvvY0h8W0QcMVJQn1dSq
gLWRP84Jim/DNuduZfuwttqOrj8QaGIloNS4IJEghHsZKBnNtfW4voITd9dYPSCQ
ExbKlgtIlw4Nfj9kDP0fxtKhMU07JWQVqV2yr9D2sU7bGQeoiv30W11R+qvPbnXs
j/RjJhT61kZd/DVNULTabuHA/NmrfCAJ/lQm8XymZ+YJlYQkRUnVj8ajSAAVayQx
qpEJnbv5jqId2BIebHNHgvS7cqrMEtMdwCGoC3G8ikX9IXuWBpLQ/qnBno9PzXUV
BjNrNMv+PVxPdjqBCIbn
=3D4fj2
-----END PGP SIGNATURE-----
```

Public Keys and Signatures

- Problem: **Public keys not linked to identity of user**
- Solution: **Validation and signing of public key by third party**
  - Certification of link between identity and public key
  - Defense against Man-in-the-Middle attacks



Overview

- **Topic of the unit**
  - Hybrid Cryptosystems
- **Parts of the unit**
  - Part #1: Cryptographic hash functions
  - Part #2: Digital signatures
  - Part #3: Hybrid cryptosystems
  - Part #4: SSL & PGP

Hybrid Cryptosystems

- **Public-key cryptography computationally demanding**
  - Involved calculations with large numbers
- **Combination of symmetric and asymmetric cryptosystems**
  - Public-key algorithm for secure exchange of session key
  - Symmetric block cipher for efficient encryption of data

  → Best of both worlds: secure and efficient communication
- However: **attacker may target weakest link of cryptosystem**

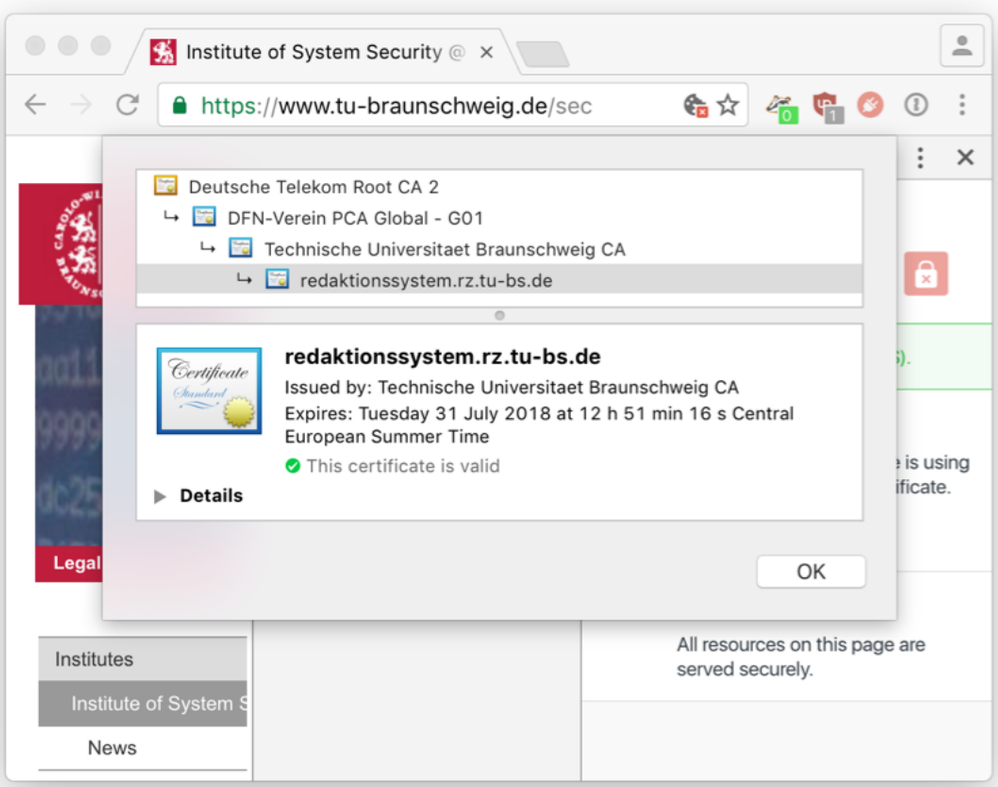
Forward Secrecy

- **Forward secrecy** = containment of compromises
  - Short-term session keys vs. long-term personal keys
  - Frequent changes of long-term keys not possible
  - Specific design of hybrid cryptosystems necessary
- Example: **Forward secrecy using Diffie-Hellman and RSA**
  - Key exchange using Diffie-Hellman algorithm
  - Random initialisation of key pairs for each session
  - Exchange signed using long-term RSA keys

Public Key Infrastructure (PKI)

- **Management of trust using public-key cryptography**
  - Digital certificates (signatures) on keys, attributes, ...
  - Certificate authorities (CA) act as trusted parties
  - Chain of trust with multiple layers

- **Different architectures**
  - Hierarchical PKI, e.g. X.509 standard
  - Web of trust, e.g. PGP software



Overview

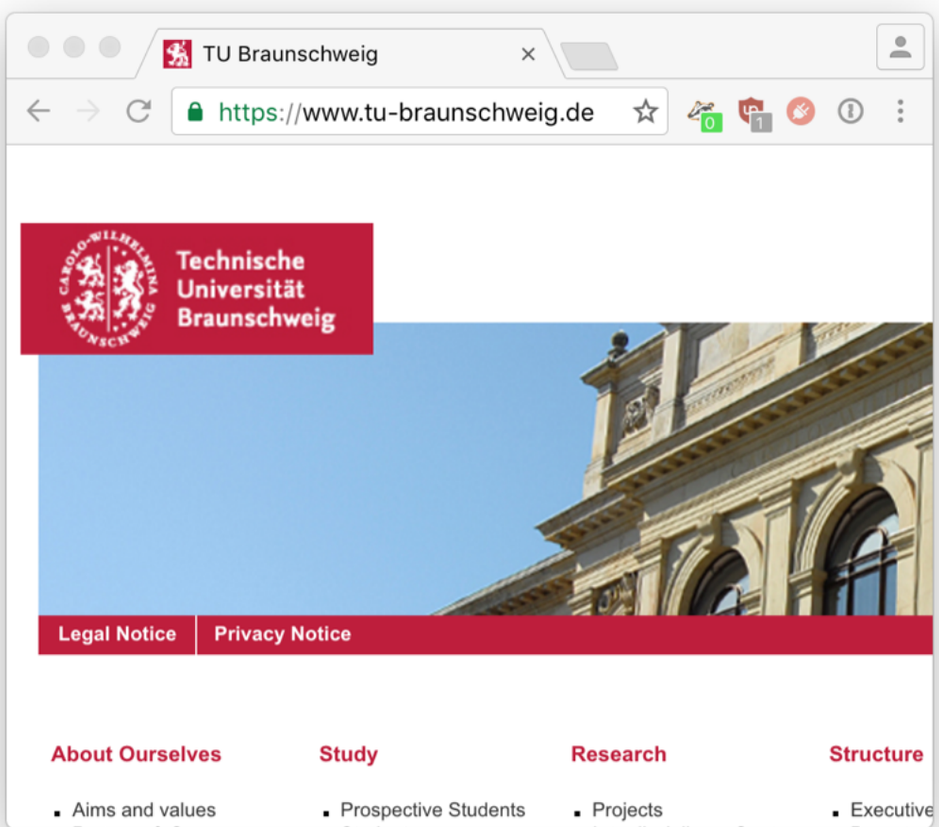
- **Topic of the unit**
  - Hybrid Cryptosystems
- **Parts of the unit**
  - Part #1: Cryptographic hash functions
  - Part #2: Digital signatures
  - Part #3: Hybrid cryptosystems
  - Part #4: SSL & PGP

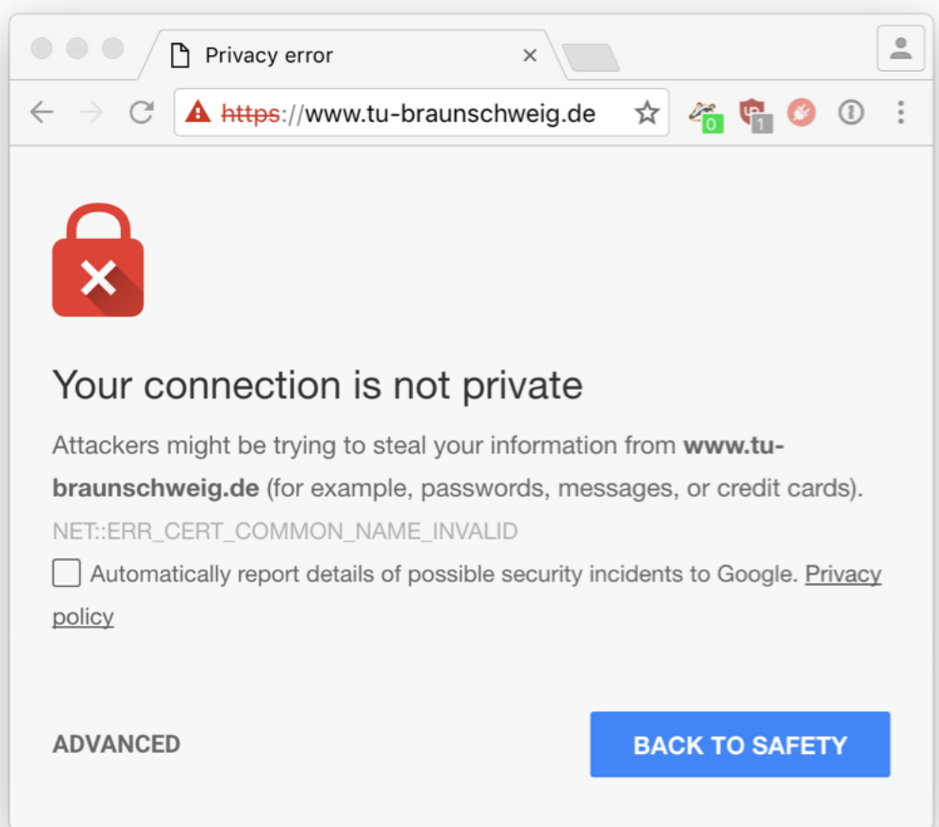
SSL: Secure Sockets Layer

- **Standard protocol for secure communication**
  - Developed in 1994 for first web browser
  - Peer-to-peer encryption (transport ⇔ application layer)
  - Standardized as Transport Layer Security (TLS); RFC 2246
- **SSL Features**
  - Hybrid cryptosystem with several algorithms
  - Support for digital certificates (X.509 standard)
  - Easy to integrate: Several extended protocol: <name>S
- Free implementation: **OpenSSL library**

SSL: In the Web

- **HTTPS:** Secure Sockets Layer (SSL) added to HTTP protocol
  - Transparent protection layer around Web communication





SSL: No more MITM?

- **Case: Forged Google certificate**
  - Issued by legitimate CA
  - Valid for *.google.com
  - Used by unknown holder

- **Large-scale attack against CA**
  - Reported in August 2011
  - Break-in at CA DigitNotar
  - 539 forged certificates

SSL: Countermeasures

- **Certificate pinning**
  - Restrict chain of trust to specific “pinned” certificates
  - Pinning of root, intermediate or end certificates possible
  - Example: HTTP Public Key Pinning in Chrome & Firefox
- **Certificate transparency**
  - Tracking of new certificates in public, append-only logs
  - Implementation of logs using Merkle hash trees
  - Regular monitoring and auditing of logs by third parties

PGP: Pretty Good Privacy

- **Classic tool for public-key cryptography**
  - Developed by P. Zimmermann in 1991
  - Widely used for encryption and signing of emails
  - Standardized as OpenPGP; RFC 4880
- **PGP features**
  - Hybrid cryptosystem with several algorithms
  - Management of trust using web-of-trust model
  - Easy integration in several mail clients
- Free implementation: **GNU Privacy Guard (GPG)**

PGP: Keysigning Party

- **Public event for mutual signing of PGP keys**
  - Validation of identity (e.g. using passport)
  - Signing of public keys with private key
  - Keysigning party in the lecture (→ canceled due to Covid19)

Problems with PGP

- **EFAIL: Attacks against OpenPGP and S/MIME**
  - Encrypted data “piggybacked” with other email to victim
  - Transport of decrypted data back to attacker, e.g. via links
  - Many email clients vulnerable (see Poddebniak et al., Sec’18)
- **Attacks spoofing signatures of OpenPGP and S/MIME**
  - Forgery of signatures using insecure APIs and clever tricks
  - Many email clients vulnerable (see Müller et al., Sec’19)

Summary

Summary

- **Public-key cryptography**
  - Man-in-the-middle attacks and defenses
  - Digital signing and verification of data
- **Implementations**
  - Hybrid cryptosystems ~ Symmetric and asymmetric
  - Public Key Infrastructures (PKI) for managing trust
- **Don’t roll your own crypto!**