CORONA EMERGENCY LECTURE

Security of Distributed Software

Prof. Dr.-Ing. Martin Gaedke
Chemnitz University of Technology
Department of Computer Science
Professorship of Distributed and Self-organizing Systems

http://vsr.informatik.tu-chemnitz.de

Chapter 3

OWASP

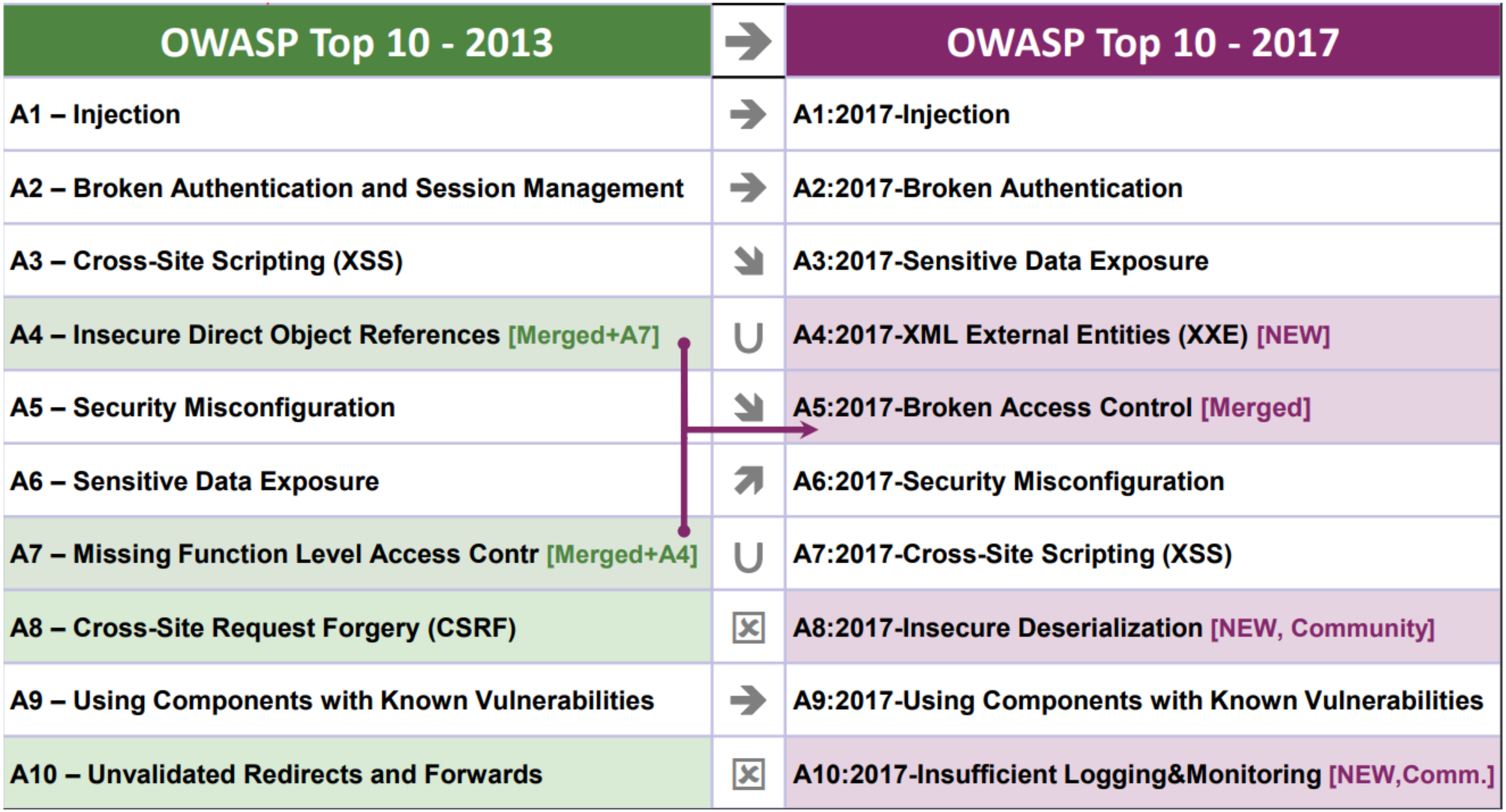
OWASP

Achtung: Die dargestellten Werkzeuge und Verfahren dienen lediglich zur Demonstration (Lehrzwecke) der Gefahrenpotenziale. Gesetzliche Bestimmungen sind zu beachten.

- The Open Web Application Security Project
  - worldwide not-for-profit charitable organization focusing on improving the security of software
  - issues software tools and knowledge-based documentation on application security
- Demo: https://www.owasp.org/

TOP 10 Report

Achtung: Die dargestellten Werkzeuge und Verfahren dienen lediglich zur Demonstration (Lehrzwecke) der Gefahrenpotenziale. Gesetzliche Bestimmungen sind zu beachten.



OWASP Top 10 - 2013	→	OWASP Top 10 - 2017
A1 – Injection	→	A1:2017-Injection
A2 – Broken Authentication and Session Management	→	A2:2017-Broken Authentication
A3 – Cross-Site Scripting (XSS)	↘	A3:2017-Sensitive Data Exposure
A4 – Insecure Direct Object References [Merged+A7]	∪	A4:2017-XML External Entities (XXE) [NEW]
A5 – Security Misconfiguration	↘	A5:2017-Broken Access Control [Merged]
A6 – Sensitive Data Exposure	↗	A6:2017-Security Misconfiguration
A7 – Missing Function Level Access Contr [Merged+A4]	∪	A7:2017-Cross-Site Scripting (XSS)
A8 – Cross-Site Request Forgery (CSRF)	☒	A8:2017-Insecure Deserialization [NEW, Community]
A9 – Using Components with Known Vulnerabilities	→	A9:2017-Using Components with Known Vulnerabilities
A10 – Unvalidated Redirects and Forwards	☒	A10:2017-Insufficient Logging&Monitoring [NEW,Comm.]

cf. **Homework**: Know what the Top10 looks like today
https://owasp.org/www-project-top-ten/

Discussion

WHICH RISKS DO YOU KNOW?

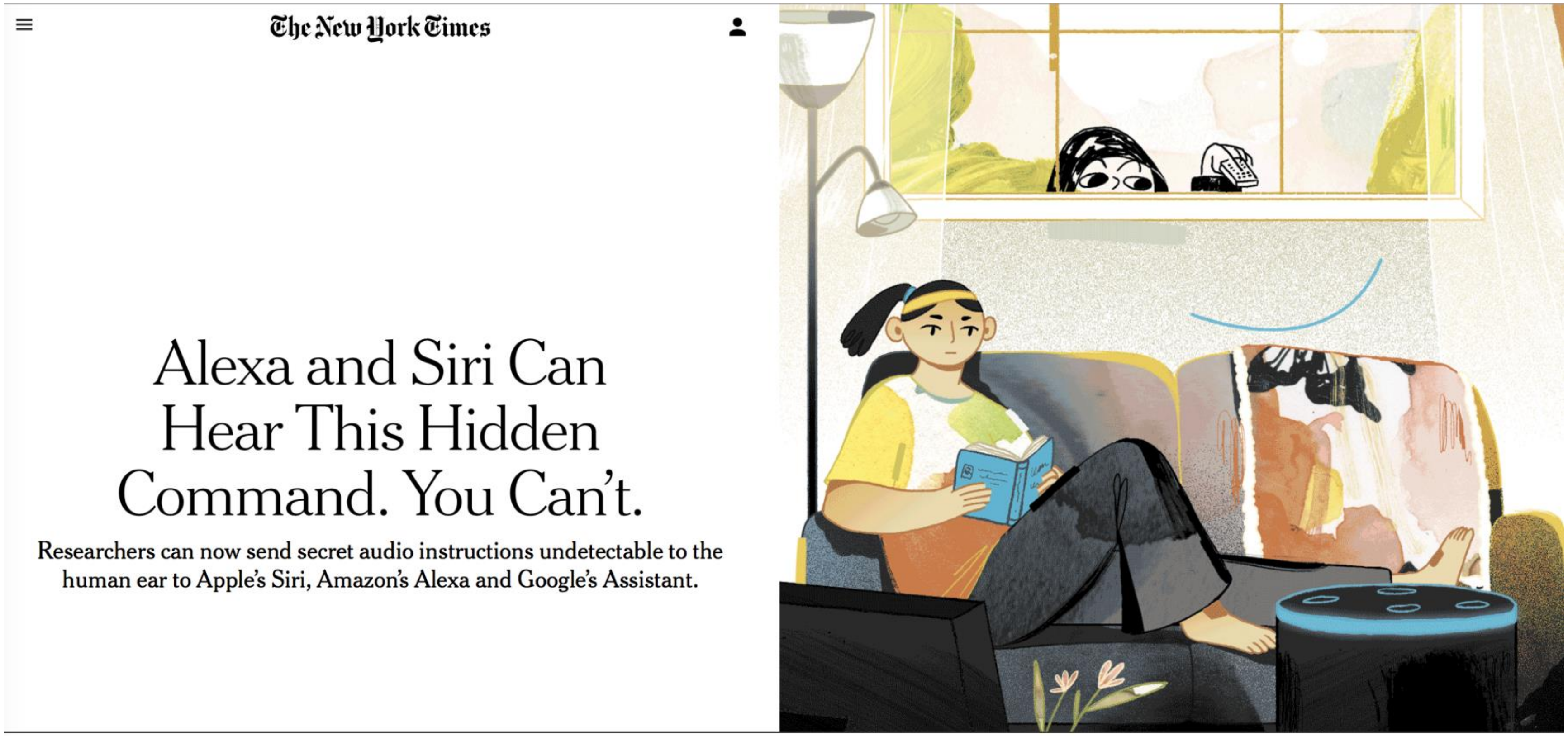
Typical Break-in

Achtung: Die dargestellten Werkzeuge und Verfahren dienen lediglich zur Demonstration (Lehrzwecke) der Gefahrenpotenziale. Gesetzliche Bestimmungen sind zu beachten.

- Dropper
- Rootkit
- Environment scan
- Key logger

- Malware:
  - UID/PWD via e-mail to ".ru"
  - Various, including e.g. Battlefield server, dropper

Another Attack on IoT



The New York Times

Alexa and Siri Can Hear This Hidden Command. You Can't.

Researchers can now send secret audio instructions undetectable to the human ear to Apple's Siri, Amazon's Alexa and Google's Assistant.

Source: https://www.nytimes.com/2018/05/10/technology/alexa-siri-hidden-command-audio-attacks.html

Homework

- Continue to read more about GDPR:
  - https://www.eugdpr.org/the-regulation.html
  - Google will also find lots of interesting places ;-)
- To be more concrete:
  - http://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32016R0679&qid=1490179745294&from=en
  - Chapter 1, Article 1 *Subject-matter and objectives* – Chapter 4, Article 43.
  - You should be able to explain what the EU-GDPR´s general provisions and principles are all about, and what they mean for the data economy and for doing business (incl. to build software)
  - You should be able to state obligations and responsibilities of data controllers and data processors, and to differentiate these two roles
  - You should be able to state the rights of the data subject, and what that means

Part II

SECURITY MECHANISMS FOR DISTRIBUTED SOFTWARE

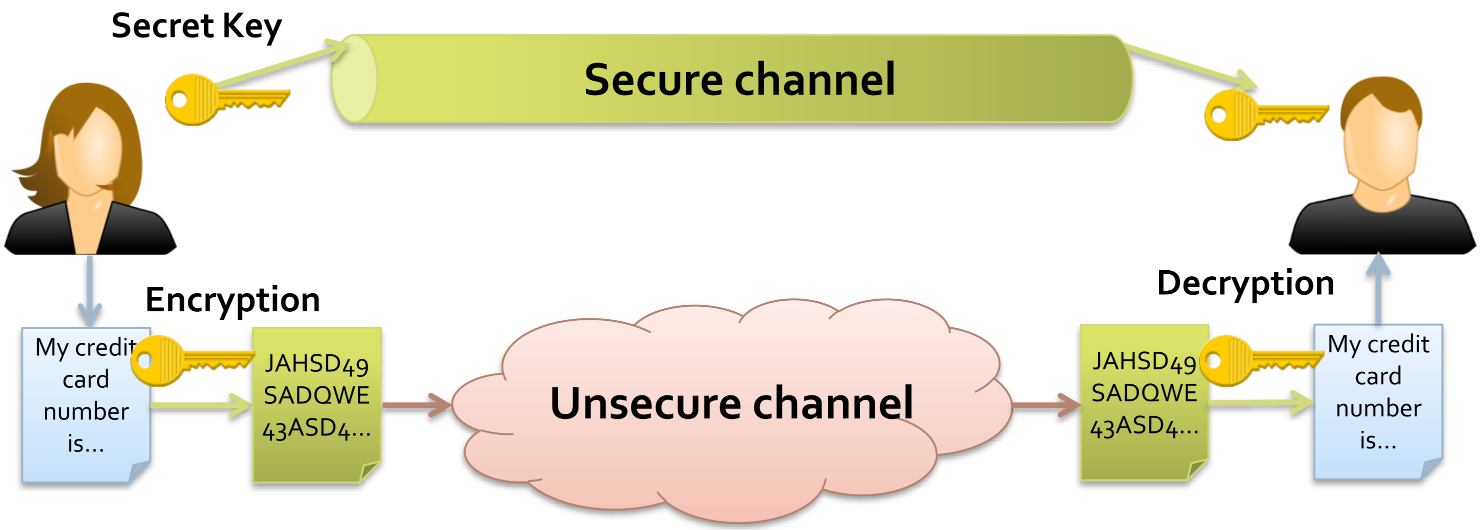
Chapter 1

CRYPTOGRAPHY – A VERY BRIEF INTRODUCTION

Introduction

- Cryptography is a broad field, which is only briefly touched in this lecture. For more information see the corresponding lectures and literature.
- Cryptographic methods used in this lecture:
  - One key (symmetric algorithms)
  - Two keys (asymmetric algorithms)
  - One-way hash functions

Symmetric Methods



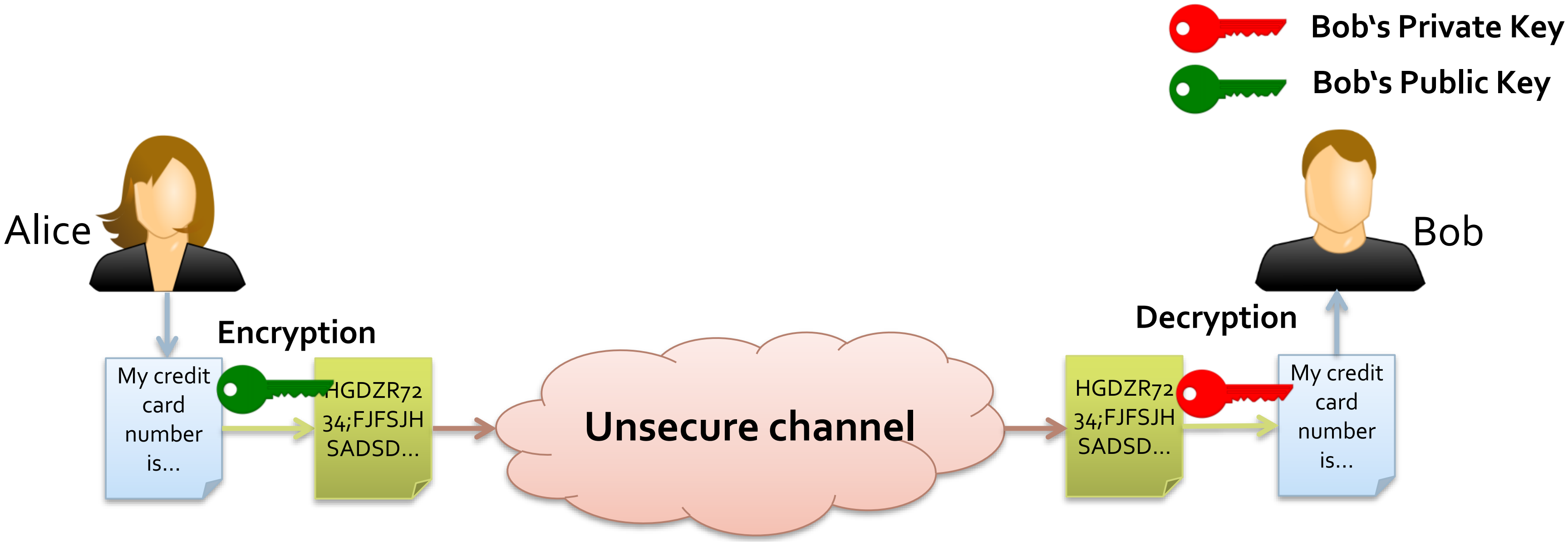
+

- Fast algorithms
- Easy to implement in hardware

−

- Secure channel is required
- Key management is necessary

Asymmetric Methods


Bob's Private Key
Bob's Public Key
Alice
Bob
Encryption
Decryption
My credit card number is...
HGDZR72 34;FJFSJH SADSD...
Unsecure channel
HGDZR72 34;FJFSJH SADSD...
My credit card number is...

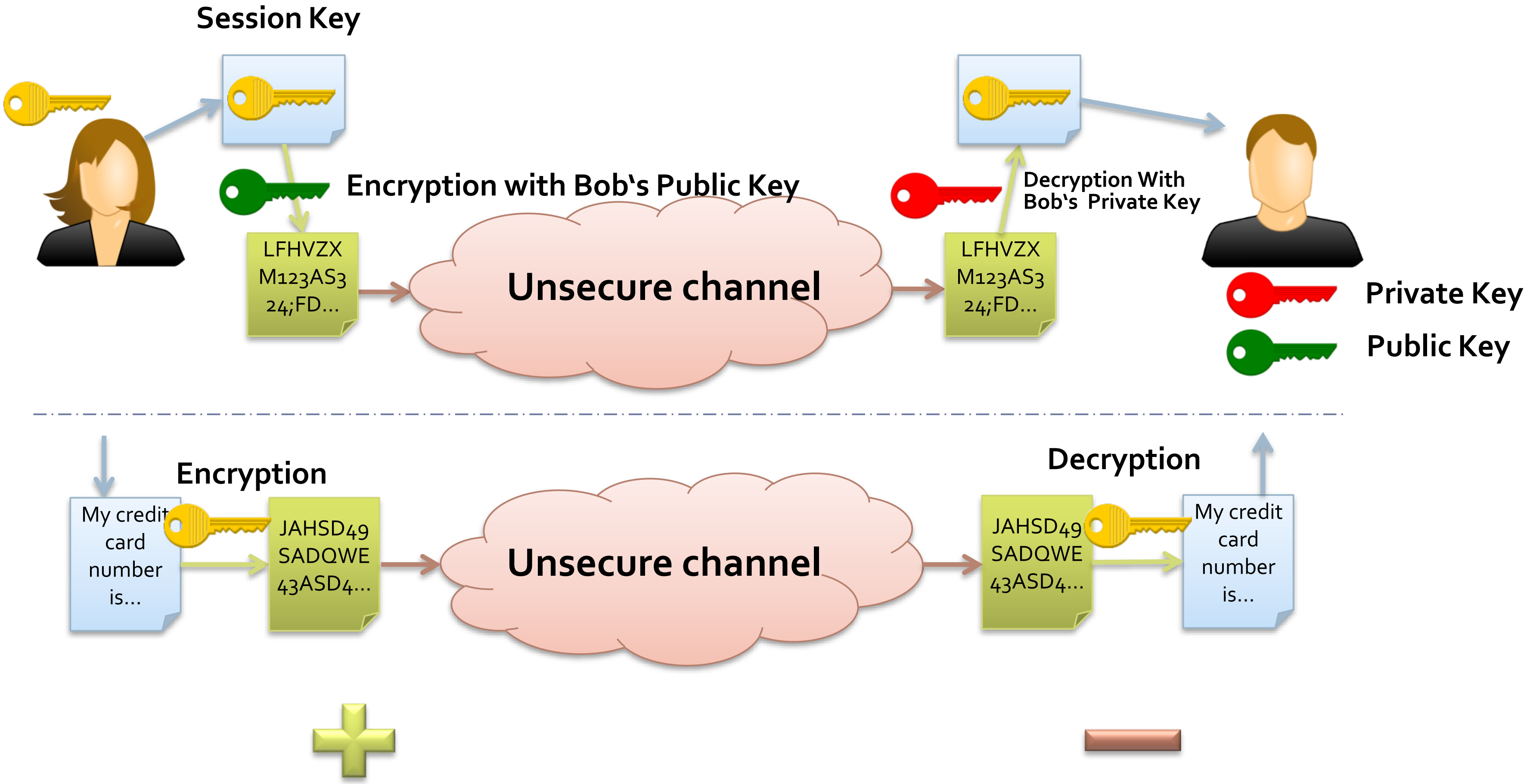
＋

- Secure channel is not necessary
- Efficient key distribution

−

- Complex encryption
- Authenticity proof of public keys is necessary

Hybrid Methods



- Performance
- Efficient key distribution

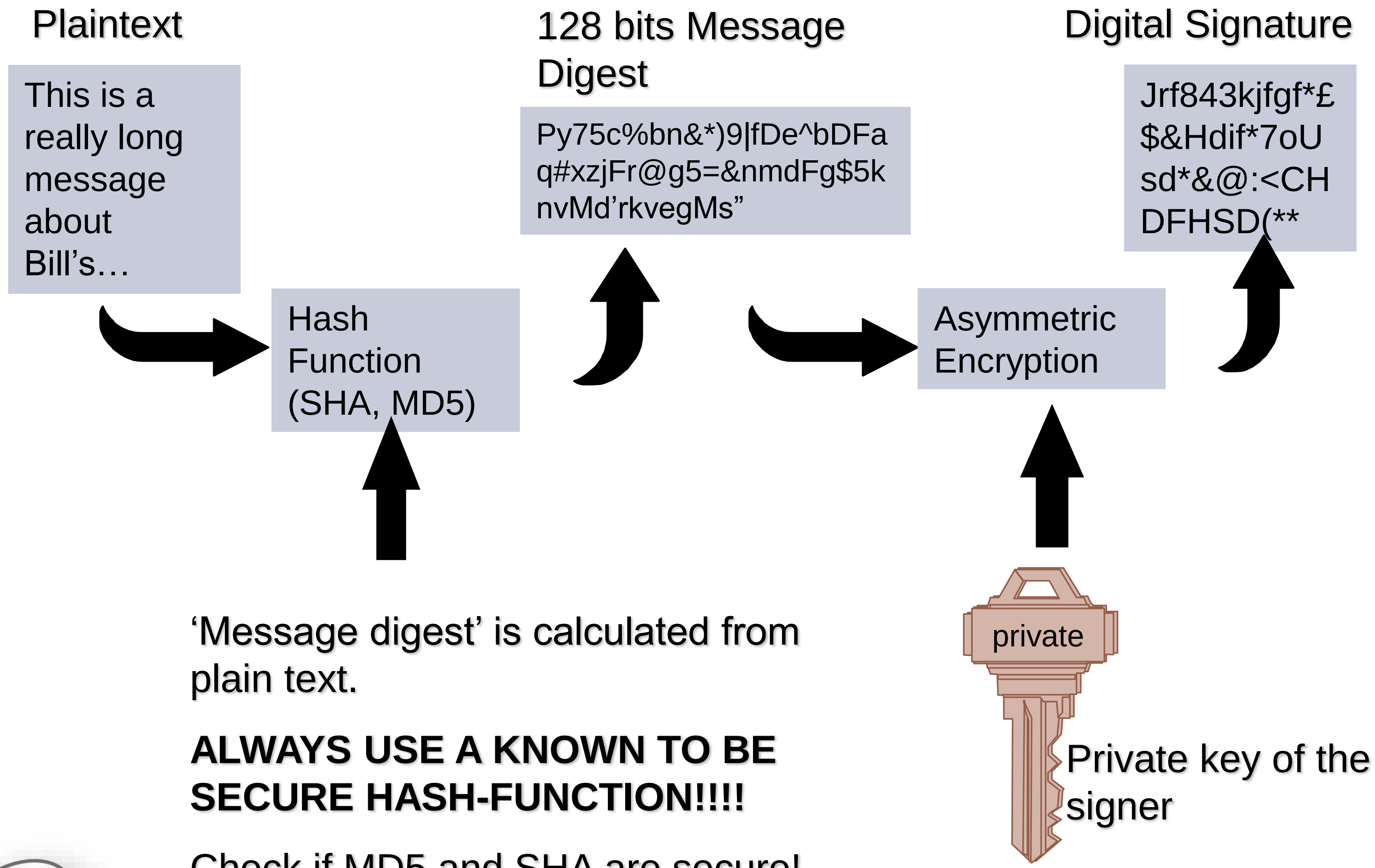
- Authenticity proof of public keys is necessary

One-way Hash Functions

- **Compression**
  Inputs of arbitrary length are mapped to outputs with fixed length
- **Irreversibility (surjective function)**
  Input can not be inferred from the output
- **Collision-resistant**
  Hash function h() is called collision resistant - if it is hard to find two inputs *a* and *b* such that $h(a) = h(b)$, and $a \neq b$

- Application: Digital signatures, authentication, integrity checks,…

Digital Signature Generation



Plaintext

This is a really long message about Bill's…

Hash Function (SHA, MD5)

128 bits Message Digest

Py75c%bn&*)9|fDe^bDFa q#xzjFr@g5=&nmdFg$5k nvMd'rkvegMs"

Asymmetric Encryption

Digital Signature

Jrf843kjfgf*£ $&Hdif*7oU sd*&@:<CH DFHSD(**

private

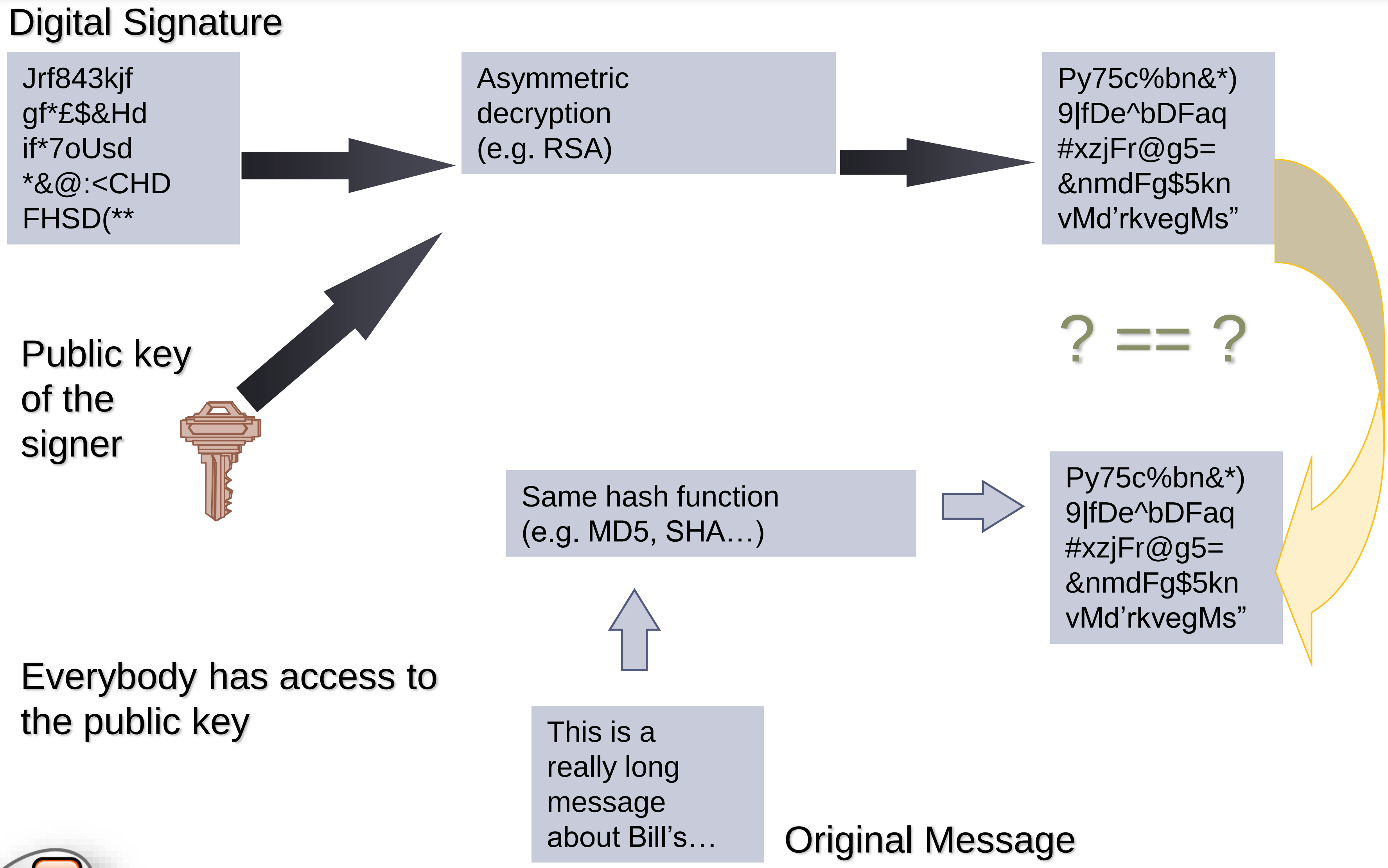
Private key of the signer

'Message digest' is calculated from plain text.

ALWAYS USE A KNOWN TO BE SECURE HASH-FUNCTION!!!!

Check if MD5 and SHA are secure!

Digital Signature Verification



Digital Signature

Jrf843kjf
gf*£$&Hd
if*7oUsd
*&@:<CHD
FHSD(**

Asymmetric decryption (e.g. RSA)

Py75c%bn&*)
9|fDe^bDFaq
#xzjFr@g5=
&nmdFg$5kn
vMd'rkvegMs"

? == ?

Public key of the signer

Same hash function (e.g. MD5, SHA…)

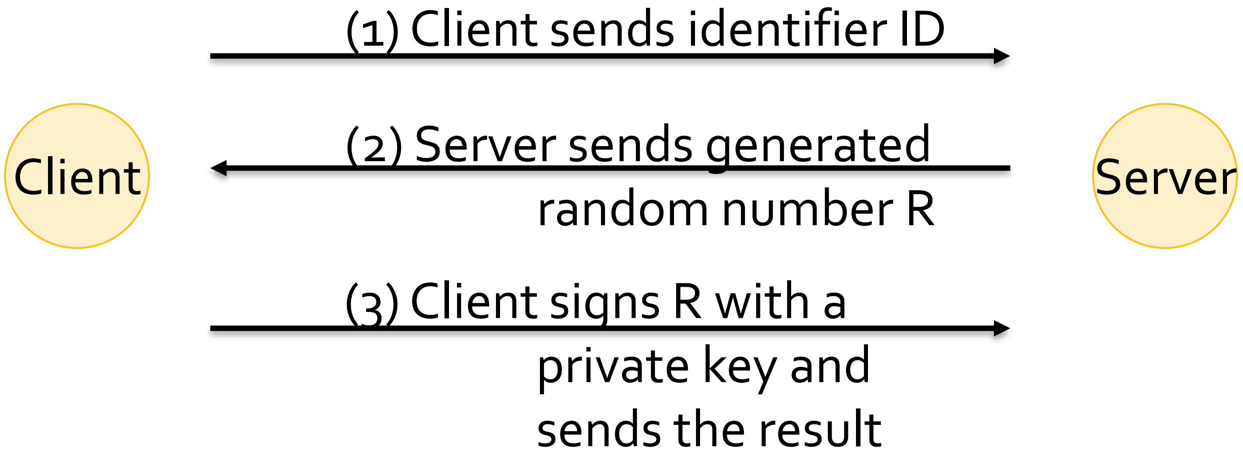
Py75c%bn&*)
9|fDe^bDFaq
#xzjFr@g5=
&nmdFg$5kn
vMd'rkvegMs"

Everybody has access to the public key

This is a really long message about Bill's…

Original Message

Challenge-Response with Public Key



Client

Server

(1) Client sends identifier ID

(2) Server sends generated random number R

(3) Client signs R with a private key and sends the result

(4) Server verifies the result using the public key of the client

Chapter 2

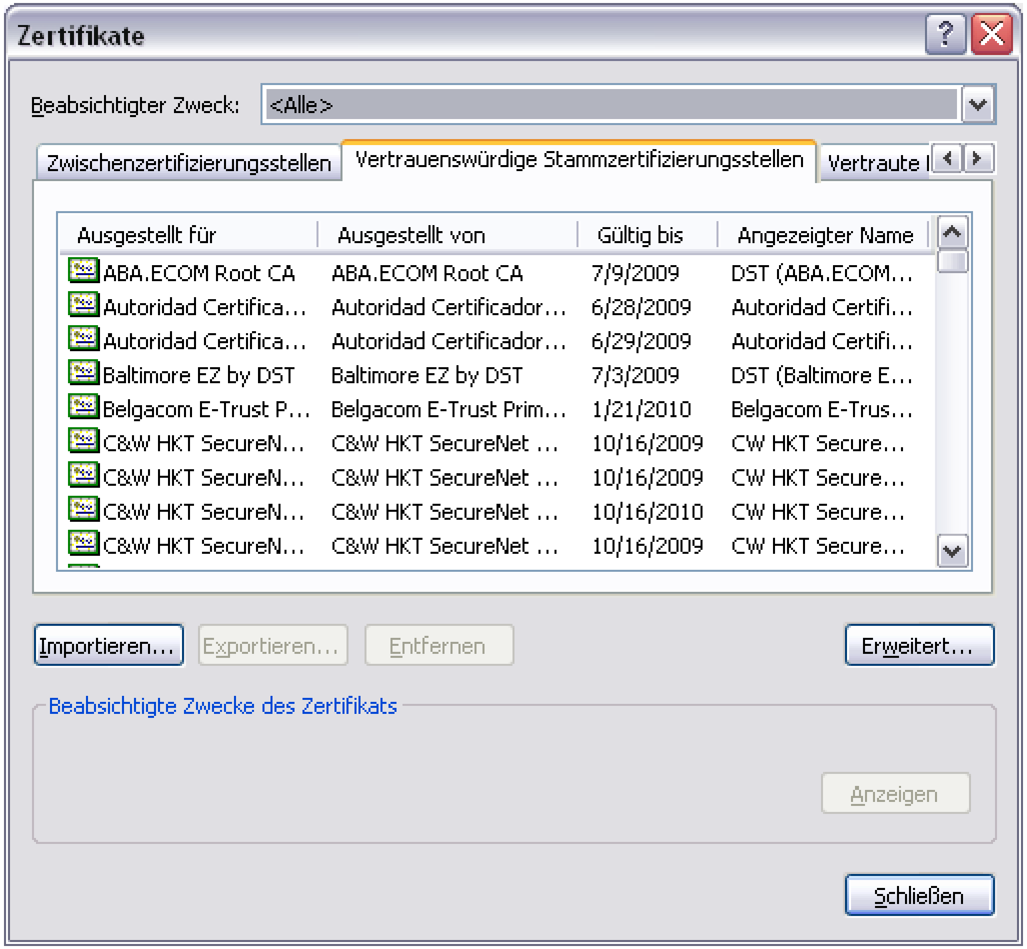
PUBLIC KEY INFRASTRUCTURE (PKI)

Public Key Infrastructure (PKI)

- Challenge: Management of public keys
- Binding the key to its owner
  - Certificate: digital certificate of public key assignment to a (legal) person
    - Example: X.509 Certificate (ITU, ISO/IEC)
  - Certification authority (CA): provides certificate issuing services; the certificates are usually signed with the private key of the CA
    - Reduce the problem of authentic key distribution to distribution of authentic keys of Cas
  - Service users must identify themselves to the CA

Root CAs

Internet Explorer: Extras → Internet options → Content → Certificates



Zertifikate

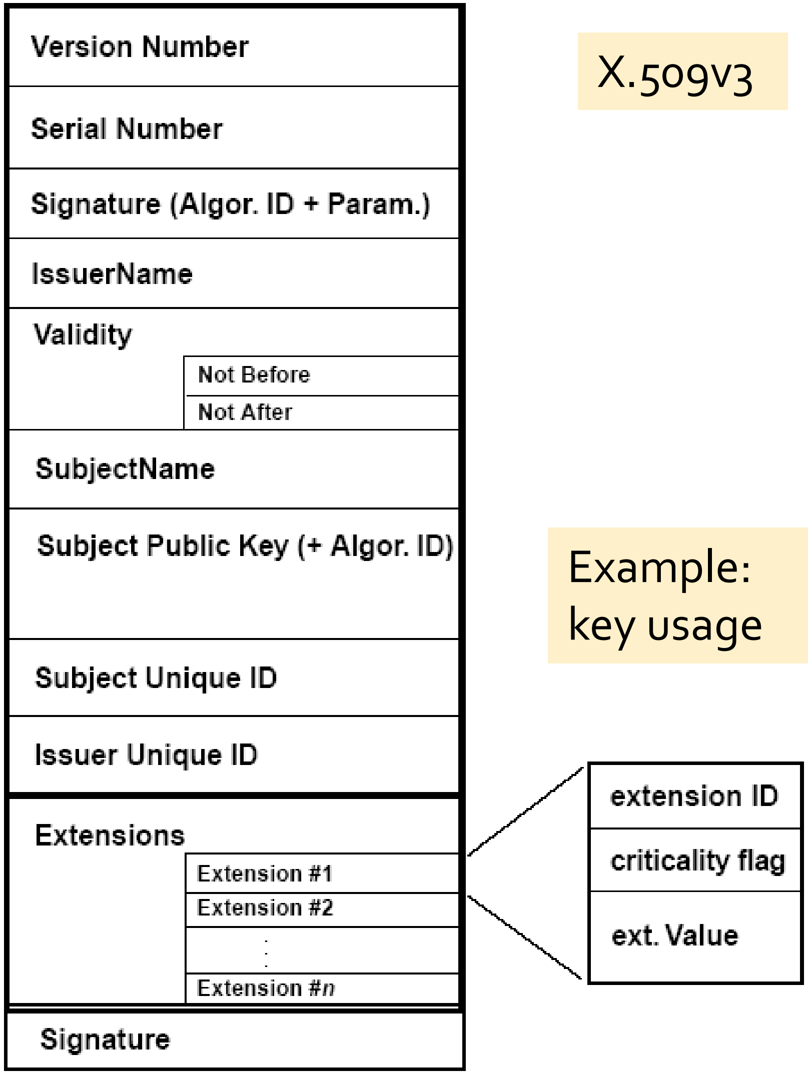
Beabsichtigter Zweck: <Alle>

Zwischenzertifizierungsstellen
Vertrauenswürdige Stammzertifizierungsstellen
Vertraute I

Ausgestellt für	Ausgestellt von	Gültig bis	Angezeigter Name
ABA.ECOM Root CA	ABA.ECOM Root CA	7/9/2009	DST (ABA.ECOM...
Autoridad Certifica...	Autoridad Certificador...	6/28/2009	Autoridad Certifi...
Autoridad Certifica...	Autoridad Certificador...	6/29/2009	Autoridad Certifi...
Baltimore EZ by DST	Baltimore EZ by DST	7/3/2009	DST (Baltimore E...
Belgacom E-Trust P...	Belgacom E-Trust Prim...	1/21/2010	Belgacom E-Trus...
C&W HKT SecureN...	C&W HKT SecureNet ...	10/16/2009	CW HKT Secure...
C&W HKT SecureN...	C&W HKT SecureNet ...	10/16/2009	CW HKT Secure...
C&W HKT SecureN...	C&W HKT SecureNet ...	10/16/2010	CW HKT Secure...
C&W HKT SecureN...	C&W HKT SecureNet ...	10/16/2009	CW HKT Secure...

X.509 Certificates

- Version Number: v1, v2 or v3
- Serial Number (uniquely assigned by the issuer)
- Signature Algorithm
- Issuer (X.500 Name)
- Validity (not before, not after)
- Subject Name
- Subject Public Key
- Unique identifier for issuer and subject



X.509v3

Example:
key usage

Certificate: Text Format

```
Certificate:
  Data:
    Version: 3 (0x2)
    Serial Number: 169910475 (0xa20a0cb)
    Signature Algorithm: sha1WithRSAEncryption
    Issuer: C=DE, O=Technische Universitaet Chemnitz, OU=Universitaetsrechenzentrum, CN=TU Chemnitz
Certification Authority - TUC/URZ CA G3/emailAddress=ca@tu-chemnitz.de
    Validity
      Not Before: Mar 21 12:21:17 2007 GMT
      Not After : Mar 19 12:21:17 2012 GMT
    Subject: C=DE, O=Technische Universitaet Chemnitz, OU=Universitaetsrechenzentrum, CN=wtc.tu-chemnitz.de
    Subject Public Key Info:
      Public Key Algorithm: rsaEncryption
        Public-Key: (2048 bit)
        Modulus:
          00:df:b6:c3:bf:ab:83:...
        Exponent: 65537 (0x10001)
    X509v3 extensions:
      X509v3 Subject Key Identifier:
        AB:BD:BB:D3:2F:9E:CA:8A:6C:C6:F6:8A:38:6E:74:8B:C0:A6:7B:61
      X509v3 Authority Key Identifier:
        keyid:E8:DA:B8:F2:47:DE:99:24:7D:67:40:89:27:67:71:0D:63:D8:A3:8E
              ..........................
```

Certificate: DER-Format

- Encoding ASN.1 structures in a binary stream

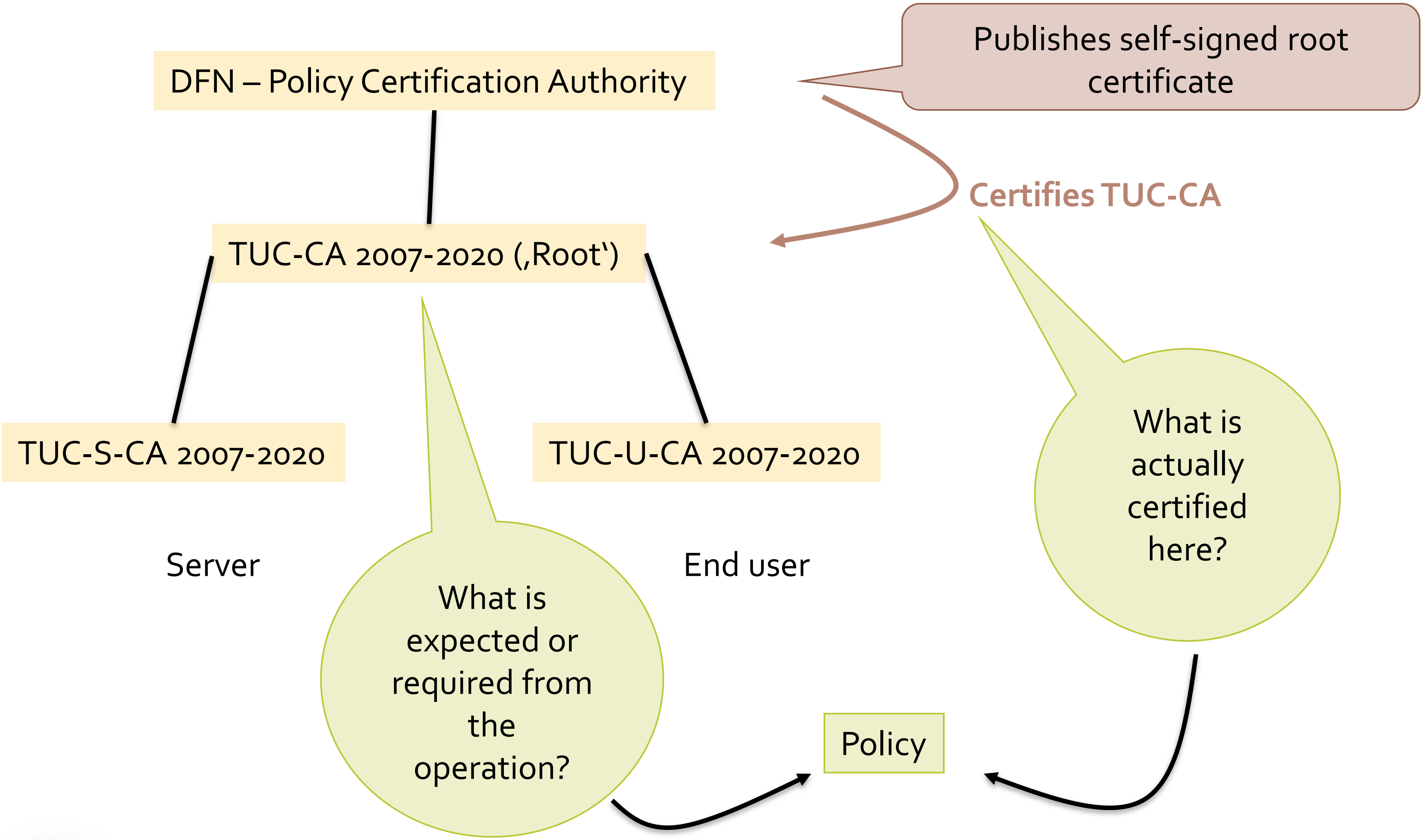
```
Attribute ::= SET {
        name IA5String,
        value INTEGER,
        flag BOOLEAN
}.
```

SET	**IA5String**	*Issuer*	**INTEGER**	4	**BOOLEAN**	*TRUE*
31 14	16 06	77 71 71 65 73 69	02 01	04	01 01	FF

Certificate: PEM-Format

```
-----BEGIN CERTIFICATE-----
MIIFiDCCBHCgAwIBAgIECkKcyjANBgkqhkiG9w0BAQUFADCBvTELMAkGA1UEBhMC
REUxKTAnBgNVBAoTIFRlY2huaXNjaGUgVW5pdmVyc2l0YWV0IENoZW1uaXR6MSMw
IQYDVQQLExpVbml2ZXJzaXRhZXRzcmVjaGVuemVudHJ1bTE8MDoGA1UEAxMzVFUg
Q2hlbW5pdHogQ2VydGlmaWNhdGlvbiBBdXRob3JpdHkgLSBUVUMvVVJaIENBIEcz
MSAwHgYJKoZIhvcNAQkBFhFjYUB0dS1jaGVtbml0ei5kZTAeFw0wNzA0MTYwNzAx
..................
LmRlL3R1LWNoZW1uaXR6LWNhL3B1Yi9jYWNlcnQvY2FjZXJ0LmNydDANBgkqhkiG
9w0BAQUFAAOCAQEAGuJSdTDZbQl6D9bonJcTOB9ZQLMTq3gQVrYtqR8lpsjBzi8E
JdTTTeNQ6K3ZhoiD+CwDG55qFGWKPQF3Mf5x+KTKfCIjgjnIrrBJyev72rVxqiKo
og7H1PbkUg1lbEimCSWg+Wx/BJSwpmddxnVYcEXMYxmGcdt66Swxlg+CowC5dVL9
6Hr98O+KtqI2NRhuH6aqUqumD8EP6YR6/oJZeo1SNM3y/QQEQoyVgPLob5uNofdr
g2twMUgptohGea3sPbmrlTLIuIvtAWqjaDidsyUKiuveWSlh4YbshZCNH+r6TzLe
zBo9/9WlA25buTEEDwGKCBOkN47rnnVLwcmqBw==
-----END CERTIFICATE-----
```

Example: DFN-PCA and TUC-CA



DFN – Policy Certification Authority

Publishes self-signed root certificate

Certifies TUC-CA

TUC-CA 2007-2020 (‚Root')

TUC-S-CA 2007-2020

TUC-U-CA 2007-2020

Server

End user

What is expected or required from the operation?

What is actually certified here?

Policy

What is Certified?

- The underlying policy’s requirements for technical components and certification methods comprise only of certification criteria "simple digital signature" or "advanced digital signature," according to § 2 No. 1 SigG 2001.
- Thereby, they are not "qualified" or "accredited" under § 2, No. 2 and 3 Signature Act 2001 and § 15 para 1 SigG 2001.
- In case of dispute, courts and experts have to check the legal value of the used keys, certificates and signatures.

DFN-PCA Requirements (1)

- A dedicated computer without any connection to a computer network is used for DFN-PCA services.
- Data exchange is performed via external data medium (e.g. floppy disk or magnetic tape); no automated data processing takes place. All the key carrying data mediums are kept in an unused condition in a safe place.
- The DFN-PCA secret keys for digital signature generation are created and used exclusively on dedicated computers.
- Backups for all relevant (digital) data will be created in regular, short intervals. The disk with the backup has to be kept at a remote location. A suitable backup concept for DFN-PCA is based on this data backup, this should in particular enable long storage periods of certificates and CRLs.

DFN-PCA Requirements (2)

- Secret signature keys of the PCA are only used to sign CA-Keys and revocation lists (CRLs) or to create cross-certificates.
- Asymmetric key pairs of the DFN-PCA for generating signatures have a length of at least 2048 bits RSA (or equivalent).
- Integrity of all data and programs on DFN-PCA computers is verified on a regular basis with the help of cryptographic applications.
- In addition, all data is treated confidentially by the PCA staff and all applicable statutory data protection regulations are complied with.

CA Requirements (1)

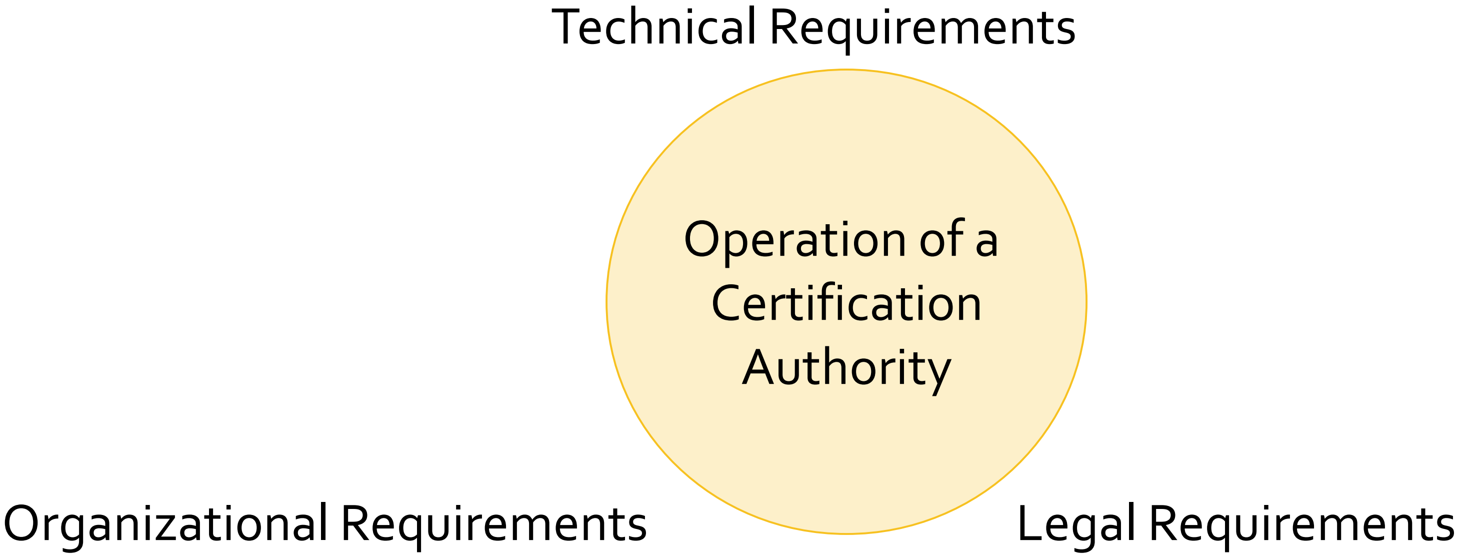
- CA services require the use of a computer, which is suitably protected against improper use. In particular, it is recommended to use a computer **without any network connection to protect it physically.**
- Secret key of the CA must be adequately protected and may not be given to third parties. The **responsibility** lies with the administrators of the CA, who are, therefore, advised to use **external peripheral devices** (eg smart card, floppy disk).
- The secret signature key of the CA must only be used to sign **CA- or End-user keys or revocation lists (CRLs)** or to create **cross-signed certificates**.

CA Requirements (2)

- Each CA must generate its asymmetric key pairs **by themselves**.
- Asymmetric key pairs of the CA for signature generation must have a **minimum length of 2048 bits RSA** (or equivalent).
- In case CA generates asymmetric key pairs for the end user, CA has to perform it on **a dedicated CA computer**.
- All data obtained during certification **must be treated as confidential** by the CA staff. CA **legal data protection regulations** are to be complied with.

Summary: CA Operation

- According to CA's requirements



Technical Requirements

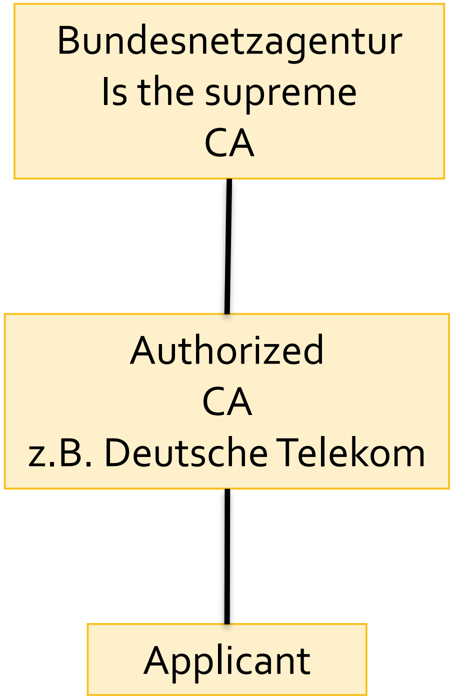
Operation of a Certification Authority

Organizational Requirements

Legal Requirements

Bundesnetzagentur (German Network Agency)

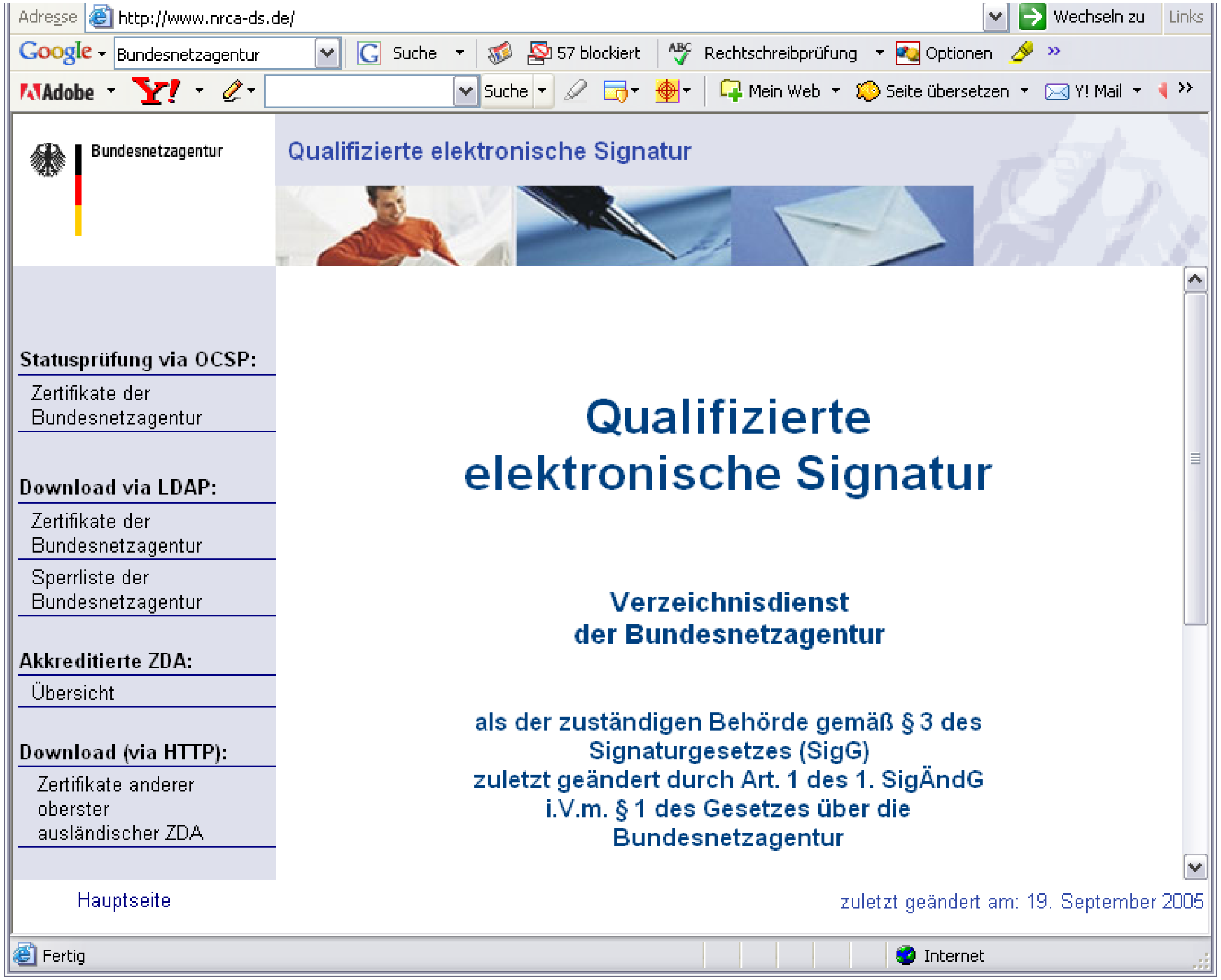
In Germany:



Bundesnetzagentur Is the supreme CA

Authorized CA z.B. Deutsche Telekom

Applicant



Bundesnetzagentur

Qualifizierte elektronische Signatur

Statusprüfung via OCSP:

Zertifikate der Bundesnetzagentur

Download via LDAP:

Zertifikate der Bundesnetzagentur

Sperrliste der Bundesnetzagentur

Akkreditierte ZDA:

Übersicht

Download (via HTTP):

Zertifikate anderer oberster ausländischer ZDA

Qualifizierte elektronische Signatur

Verzeichnisdienst der Bundesnetzagentur

als der zuständigen Behörde gemäß § 3 des Signaturgesetzes (SigG) zuletzt geändert durch Art. 1 des 1. SigÄndG i.V.m. § 1 des Gesetzes über die Bundesnetzagentur

Hauptseite

zuletzt geändert am: 19. September 2005

Certificate Revocation

- Such as with credit card and mobile phone
- OCSP: Online Certificate Status Protocol (RFC 2560)
- Addition or replacement of *Certificate Revocation Lists*

Procedure:

- Request whether the given certificate is valid
  - In general, request to the certificate issuer or a CA *designated responder*
- Response signed by certificate authority or CA DR:
  - Good, revoked, unknown

Chapter 3

SSL/TLS

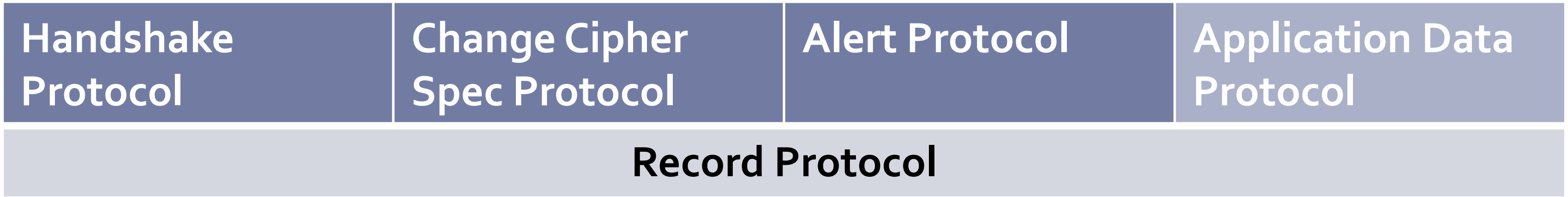
SSL/TLS – Overview

- Secure Sockets Layer (SSL)
  - Version 1.0 by Netscape Communications (1994)
- Transport Layer Security (TLS)
  - IETF-standard from the year 1999 (RFC 2246)
- Network protocol for secure data transfer
- Since Version 3.0 SSL is being further developed under the name TLS
  - Minor differences between SSL 3.0 & TLS 1.0
  - TLS 1.0 is presented as SSL 3.1

- TLS 1.3 finalized by March 21, 2018
  - Supports for performance and better security
  - https://datatracker.ietf.org/doc/draft-ietf-tls-tls13/
  - Why browsers don't support the latest TLS - and what a POODLE has to do with it. **HOMEWORK:** https://blog.cloudflare.com/why-tls-1-3-isnt-in-browsers-yet/

SSL/TLS – Architecture

- In OSI-model in layer 6
- In TCP/IP-model
  - Above the Transport layer (i.e. TCP,…)
  - Below the Application layer (i.e. HTTP,…)
- Basic idea: generic security layer
- Protocol consists of 2 layers:



Handshake Protocol

Change Cipher Spec Protocol

Alert Protocol

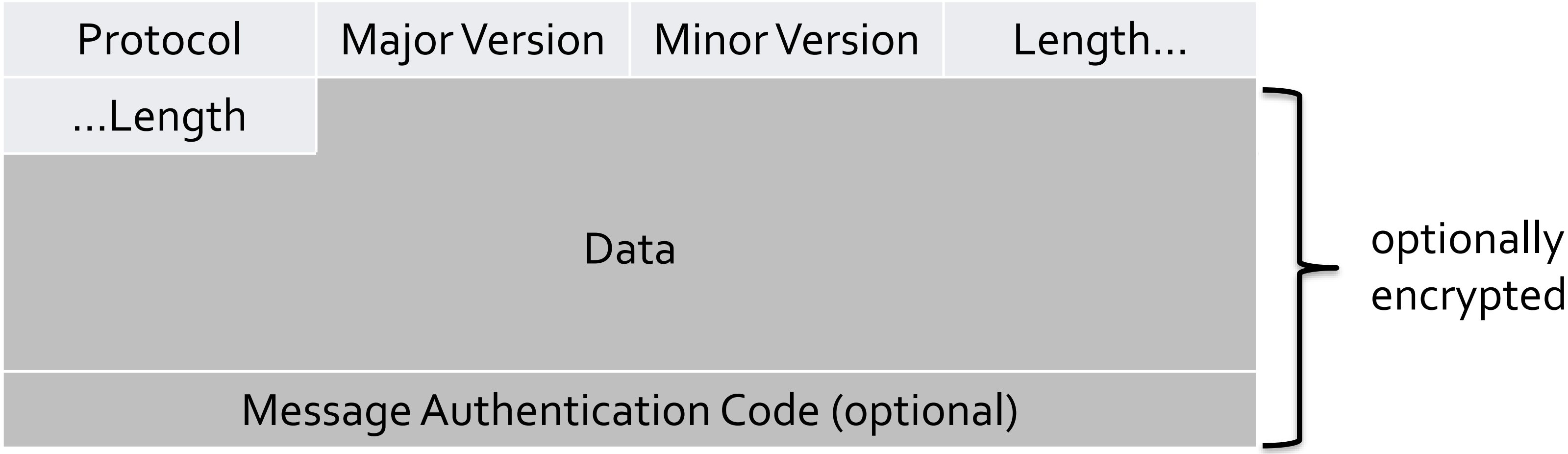
Application Data Protocol

Record Protocol

Record Protocol

- Represents the lower level of the TLS protocol
- Encapsulation of exchanged messages
- Decomposition into blocks for transmission
- End-to-End encryption
  - Symmetric algorithms
  - (see the following handshake protocol)
- Integrity and authenticity are ensured by cryptographic checksums

Record Protocol



Protocol

Major Version

Minor Version

Length...

...Length

Data

Message Authentication Code (optional)

optionally encrypted

Handshake Protocol

- Server and Client decide on
  - Mode of encryption
  - Type of message authentication
  - Secret key
- Authentication via certificates is possible (X.509v3)

Change Cipher Spec Protocol

- Change to the negotiated **Cipher Suite**
- Cipher Suite identifies a combination of four algorithms:
  - Key exchange
  - Authentication
  - Hash function
  - Encryption

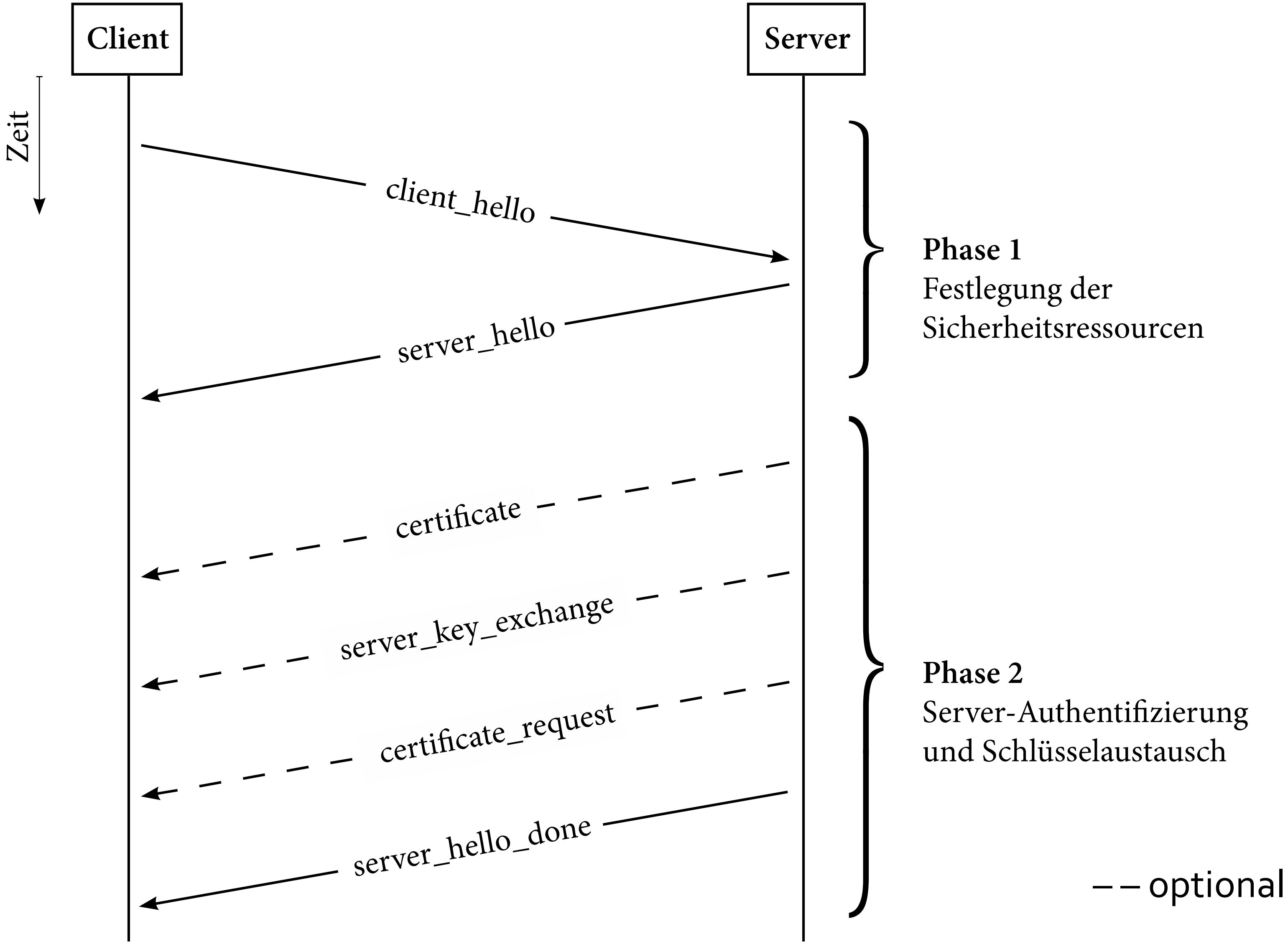
Alert Protocol

- Signaling on error states
- Protocol defines two fields:
  - Level of error alert
    - warning
    - fatal → connection is immediately interrupted
  - Type of error alert
    - detailed error description

Application Data Protocol

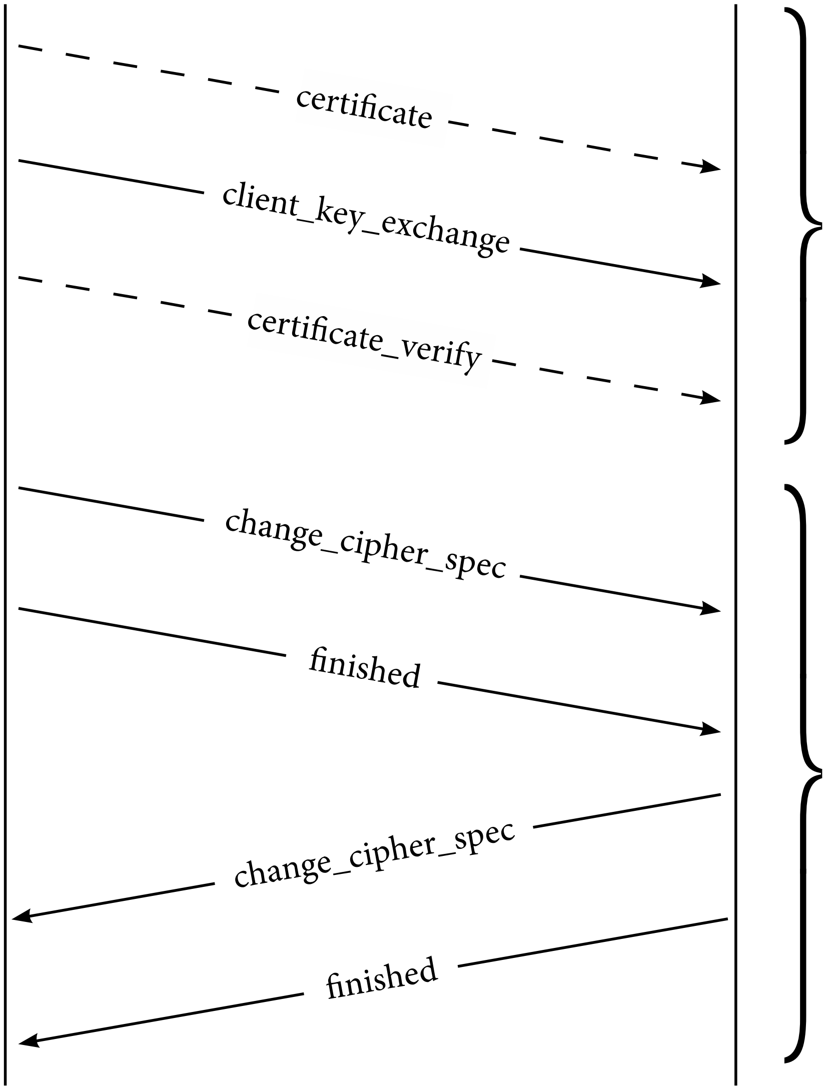
- Pass application data transparently
  - without consideration of its content!
- Based on security parameters data is…
  - fragmented
  - compressed
  - protected
  - encrypted

Handshake-Protocol – Part 1



Quelle: http://www.repges.net/SSL/Architektur_SSL/HANDSH_1/handsh_1.HTM

Handshake-Protocol – Part 2



Phase 3
Client-Authentifizierung
und Schlüsselaustausch

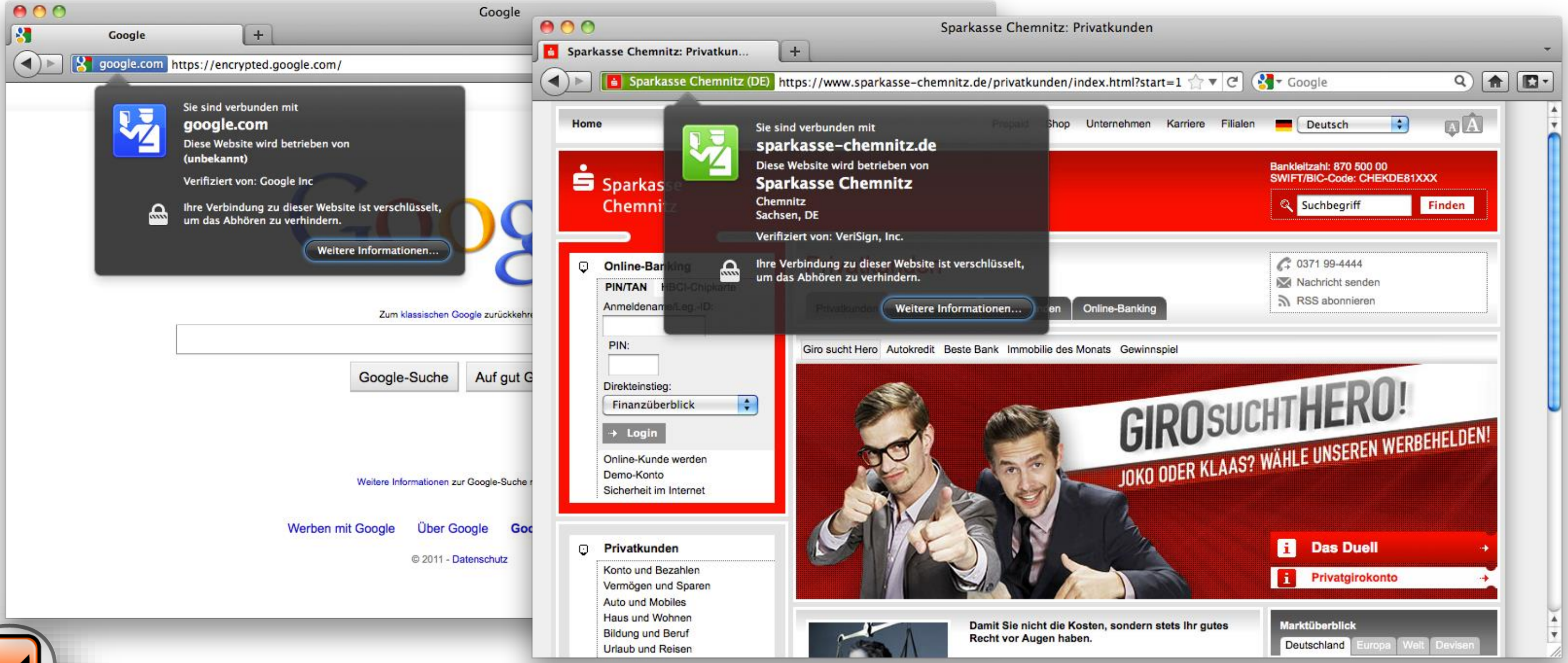
Phase 4
Beendigung
des Handshake

– – optional

Quelle: http://www.repges.net/SSL/Architektur_SSL/HANDSH_1/handsh_1.HTM

Hypertext Transfer Protocol Secure

- HTTP with additional transmission encryption by SSL/TLS
- Standard-Port: 443



Google

https://encrypted.google.com/

Sie sind verbunden mit
google.com
Diese Website wird betrieben von
(unbekannt)
Verifiziert von: Google Inc
Ihre Verbindung zu dieser Website ist verschlüsselt, um das Abhören zu verhindern.

Sparkasse Chemnitz: Privatkunden

https://www.sparkasse-chemnitz.de/privatkunden/index.html?start=1

Sie sind verbunden mit
sparkasse-chemnitz.de
Diese Website wird betrieben von
Sparkasse Chemnitz
Chemnitz
Sachsen, DE
Verifiziert von: VeriSign, Inc.
Ihre Verbindung zu dieser Website ist verschlüsselt, um das Abhören zu verhindern.

Bankleitzahl: 870 500 00
SWIFT/BIC-Code: CHEKDE81XXX

Damit Sie nicht die Kosten, sondern stets Ihr gutes Recht vor Augen haben.

Chapter 4

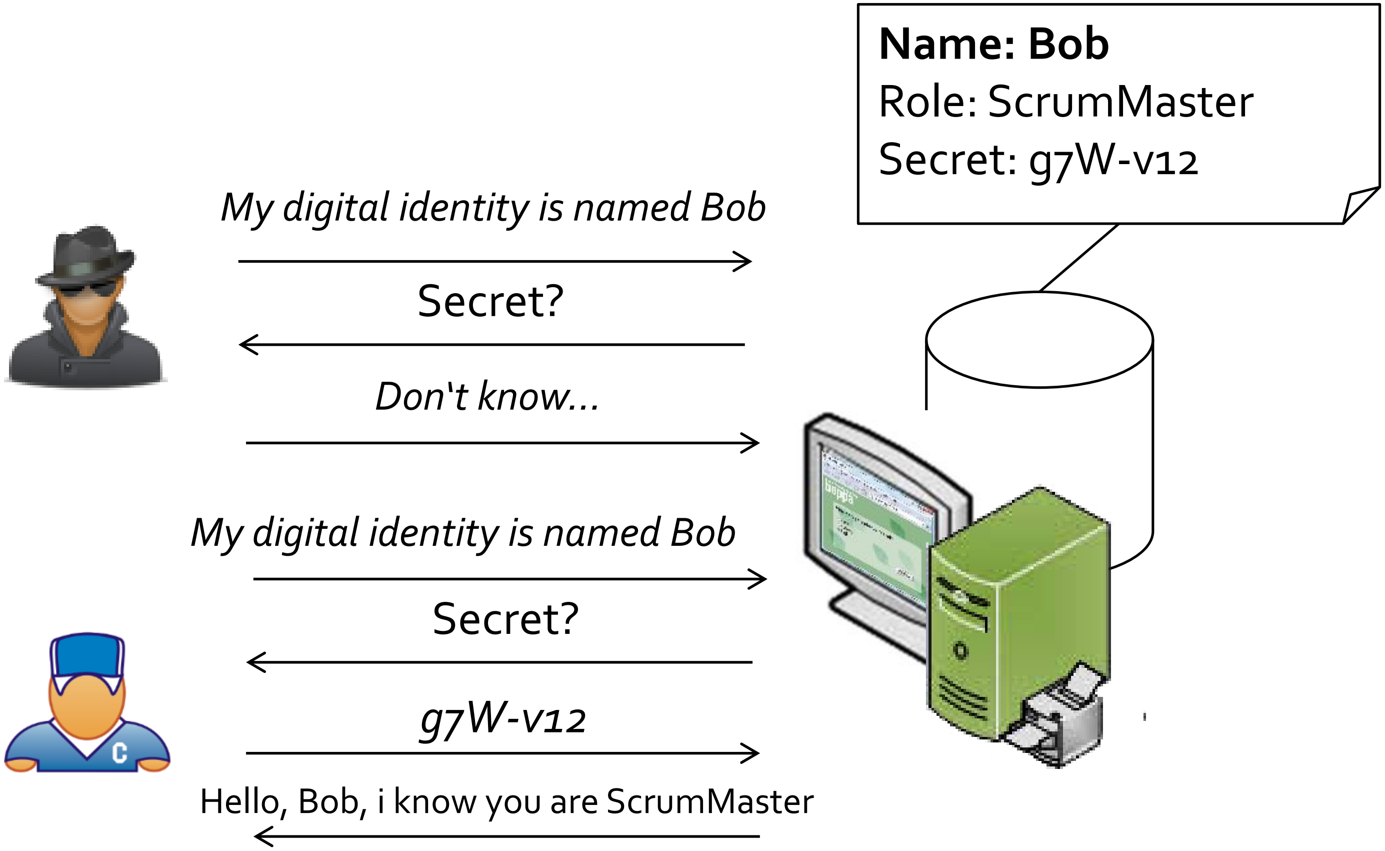
AUTHENTICATION

Section

INTRODUCTION

Terminology

- **Authentication** is the process of verification if someone is the one who he claims to be



Name: Bob
Role: ScrumMaster
Secret: g7W-v12

My digital identity is named Bob

Secret?

Don't know…

My digital identity is named Bob

Secret?

g7W-v12

Hello, Bob, i know you are ScrumMaster

Authenticators

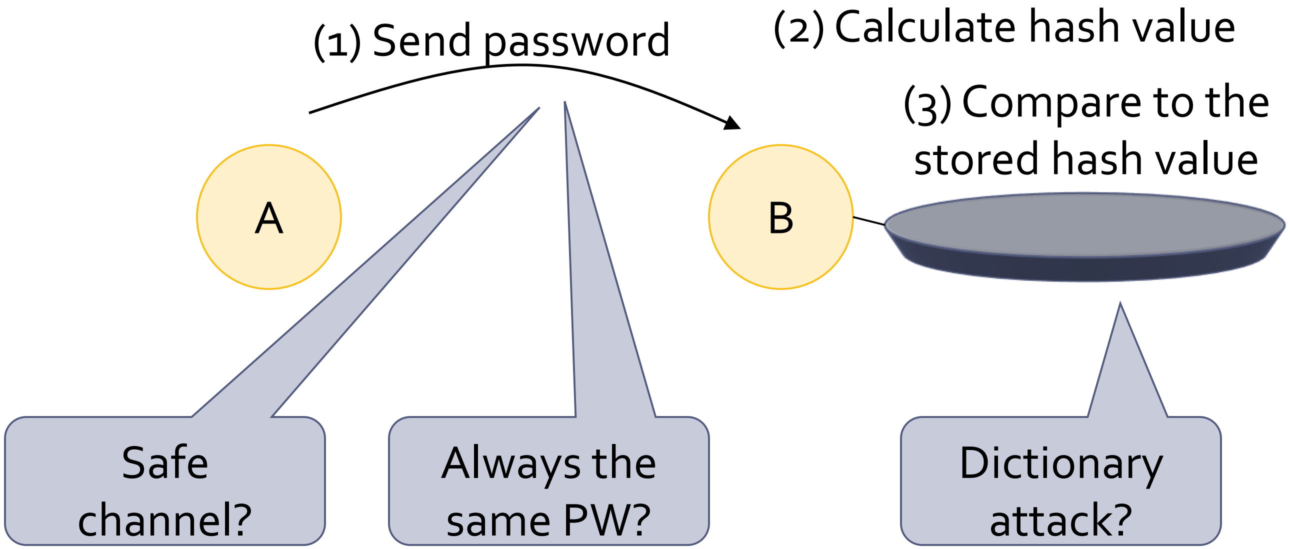
- What you know: knowledge-based
  - PIN, passwords
  - Challenge-Response
- What you are: biometrics
  - Fingerprint, iris, voice, signature, keystroke behavior
- What you have: ownership-based
  - Something that you do not notice, but what is stored on a medium
  - IDs, magnetic cards, certificates, smart cards
- Multi-factor authentication
  - Combination of different types of authentication
  - 2-Factors: deposit card + PIN, credit card + signature
  - 2-Factors: password + PIN send by SMS
  - 3-Factors: password + smart card + fingerprint

Knowledge-based Authentication

Knowledge-based authentication using passwords:

- Alice agrees with Bob on a secret password p for authentication of Alice to Bob.
- Bob applies a one-way or cryptographic hash function H on the password, and stores the image value H (p).

Authentication:



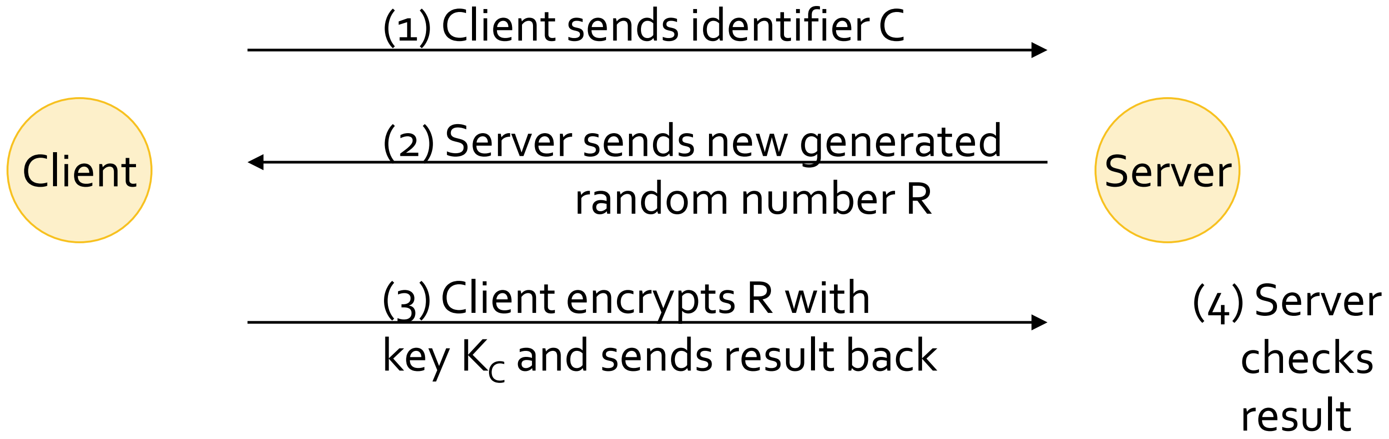
Corresponding Password Policy

- How long should a password be? Which character set should be used?
- System generated or user generated?
- How often should the password be changed?
- For which roles / functions must separate rules exist?
- How many attempts to enter the right password?
- Are the used hash functions still safe?

- Example: Passwords must have between 6 and 8 printable ASCII characters except @, #, %, " and $ and must contain at least two _different_ non-alphanumeric characters. Usable characters are: ! & ' ( ) * + , - . / : ; < = > ? [\ ] ^ _ ` { | } ~

Challenge-Response Method

Client and server share a common secret K_C and an encryption method E:



(1) Client sends identifier C

Client

(2) Server sends new generated random number R

Server

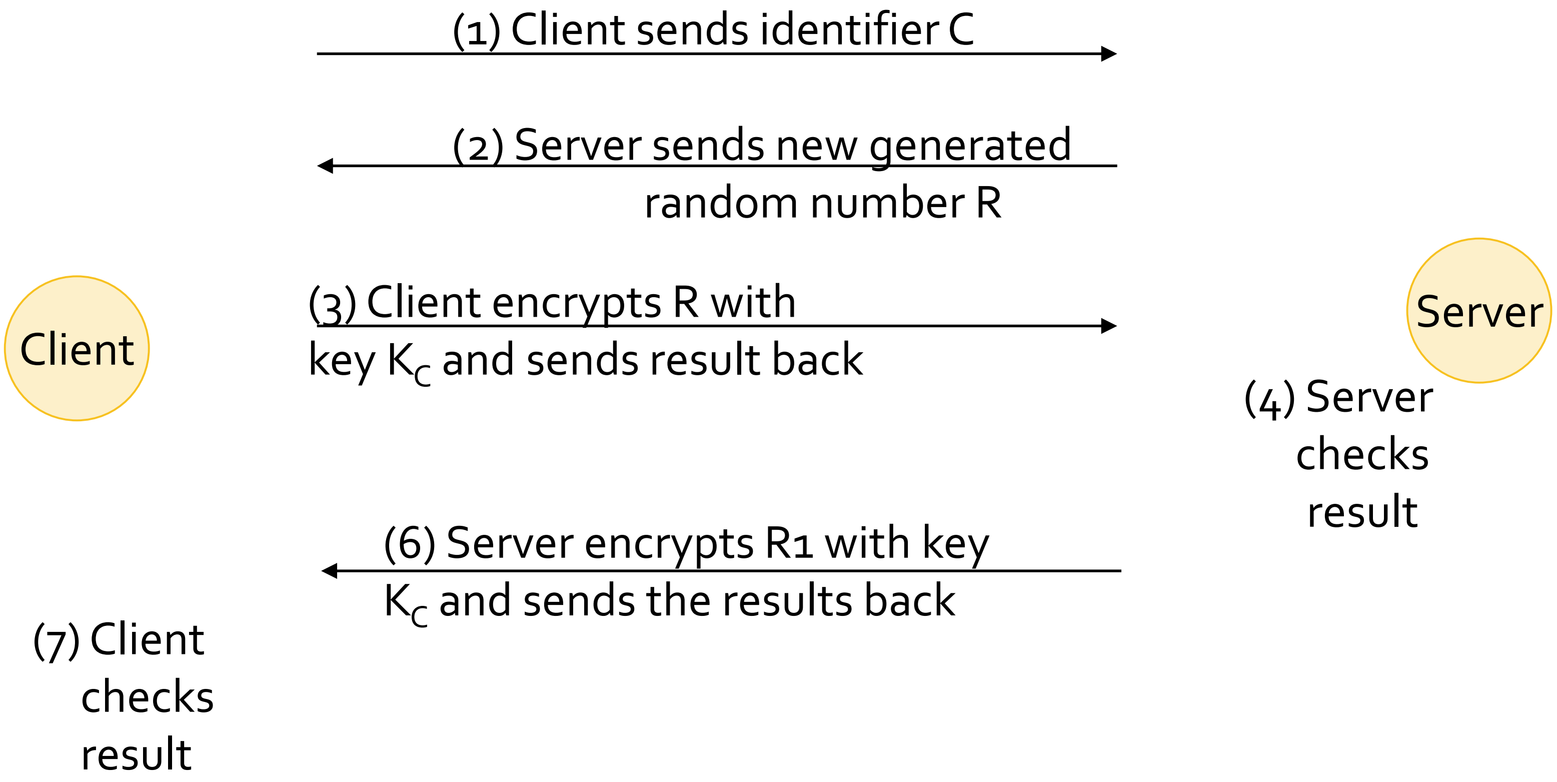
(3) Client encrypts R with key K_C and sends result back

(4) Server checks result

- Ways to Attack:
  - Listening to the channel and cryptanalysis
  - Man-in-the-middle attack
  - Attack on the server that has stored the key

Authentication on Both Sides

- Each partner checks whether each of the others knows the secret

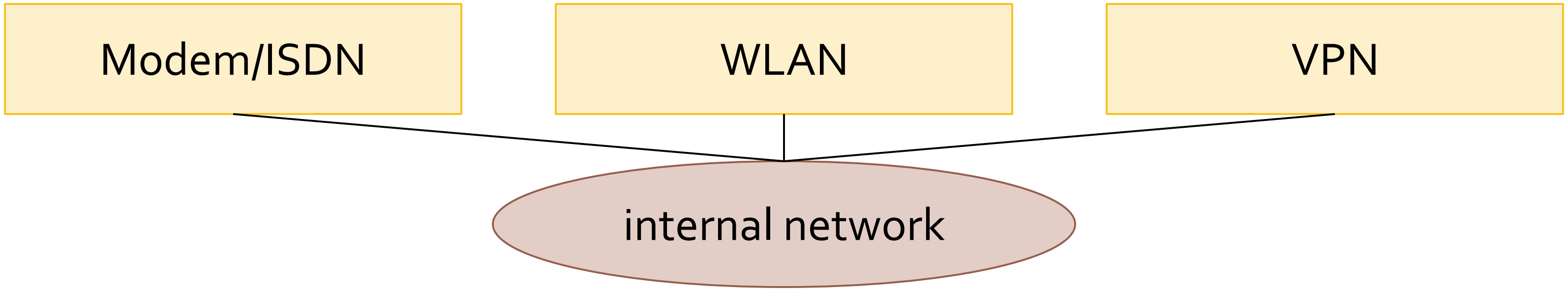


(1) Client sends identifier C
(2) Server sends new generated random number R
Client
(3) Client encrypts R with key K_C and sends result back
Server
(4) Server checks result
(6) Server encrypts R1 with key K_C and sends the results back
(7) Client checks result

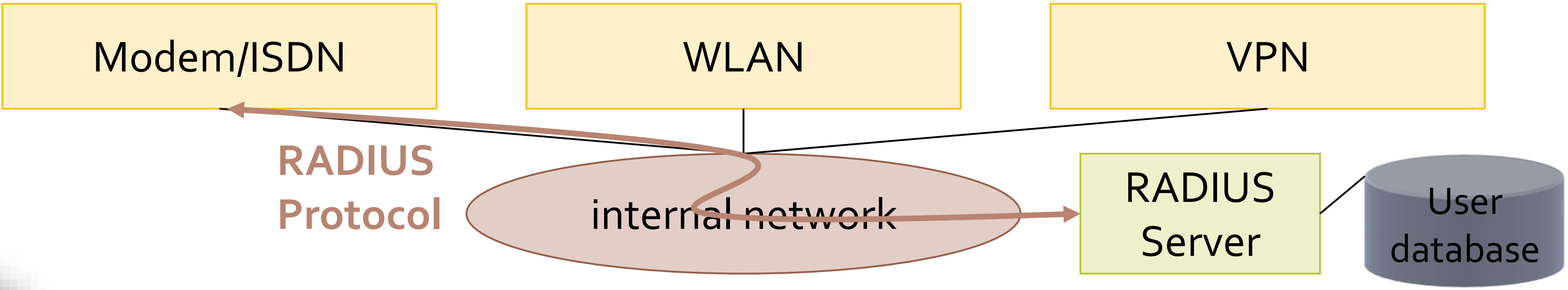
Section

AUTHENTICATION IN DISTRIBUTED SYSTEMS

Remote Access



- Problem: Number of access points, authentication is required at each of them. Is it necessary to store and manage authentication data at each access point?
- Idea: Centralization by “remote authentication”
- RADIUS: Remote Authentication Dial In User Service

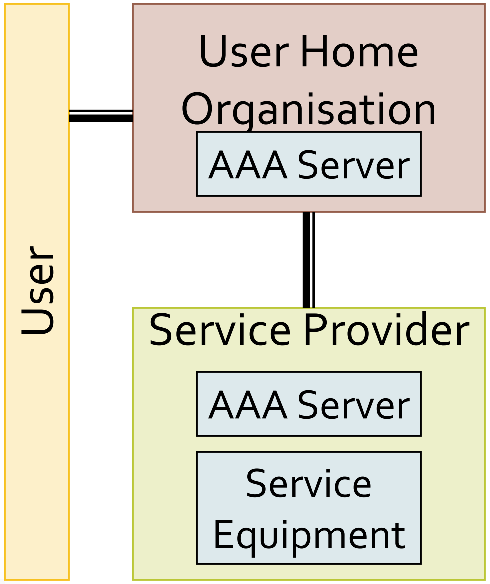


RADIUS

- UDP-based Client-Server-Protocol
- RADIUS client and RADIUS server share a pre-agreed secret S
- Password Authentication Protocol (PAP):
  - Client encrypts user password with the help of the secret S and sends it to the RADIUS server.
  - The radius-server decrypts the password and checks it.
  - If the password is correct an "access accept message" will be sent to the client, otherwise an "access reject message" will be sent.
  - To ensure the authenticity of the RADIUS server, the client sends a random number with its request ("Request Authenticator"). This number will be encrypted by the server with the secret S and sent back in the access message.
  - Alternative: Challenge Handshake Authentication Protocol (CHAP)
  - Challenge response method
- RADIUS does not use message encryption!

AAA-Architecture

- AAA: Authentication, authorization, accounting
- RADIUS is a simplified implementation of this architecture
- AAA Authorization Framework, RFC 2904, August 2000
  - Recent development in the context of DIAMETER



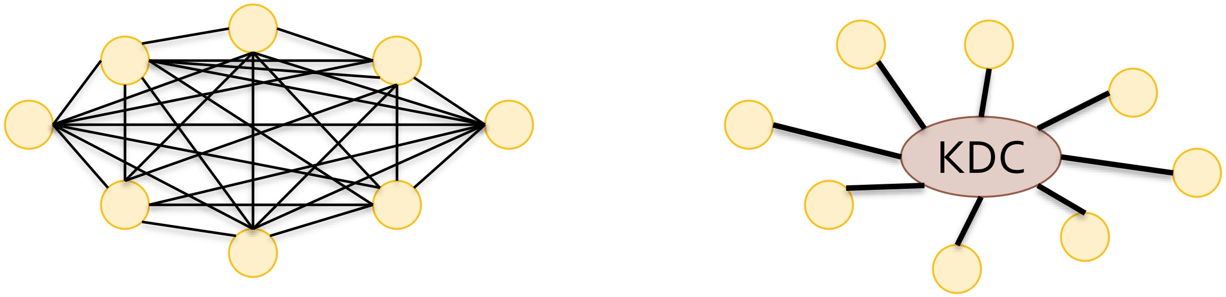
- UHO and Service Provider can be located in the same or different "administrative domains"
- Various processes possible:

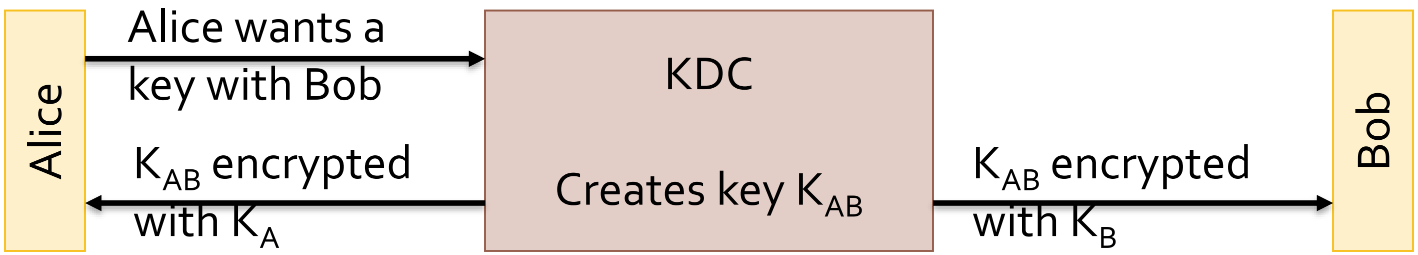
  U → UHO → SP → UHO → U

  U → SP → UHO → SP → U

  U → UHO → U → SP → U
- How do you deal with multiple SPs?

Key Distribution Center (KDC)





Alice

Alice wants a key with Bob

K_{AB} encrypted with K_A

KDC

Creates key K_{AB}

K_{AB} encrypted with K_B

Bob

- Centralization reduces complexity from (N-1)! to N
- Single-Sign-On
- Drawbacks: “Single point of failure“, performance bottleneck

Section

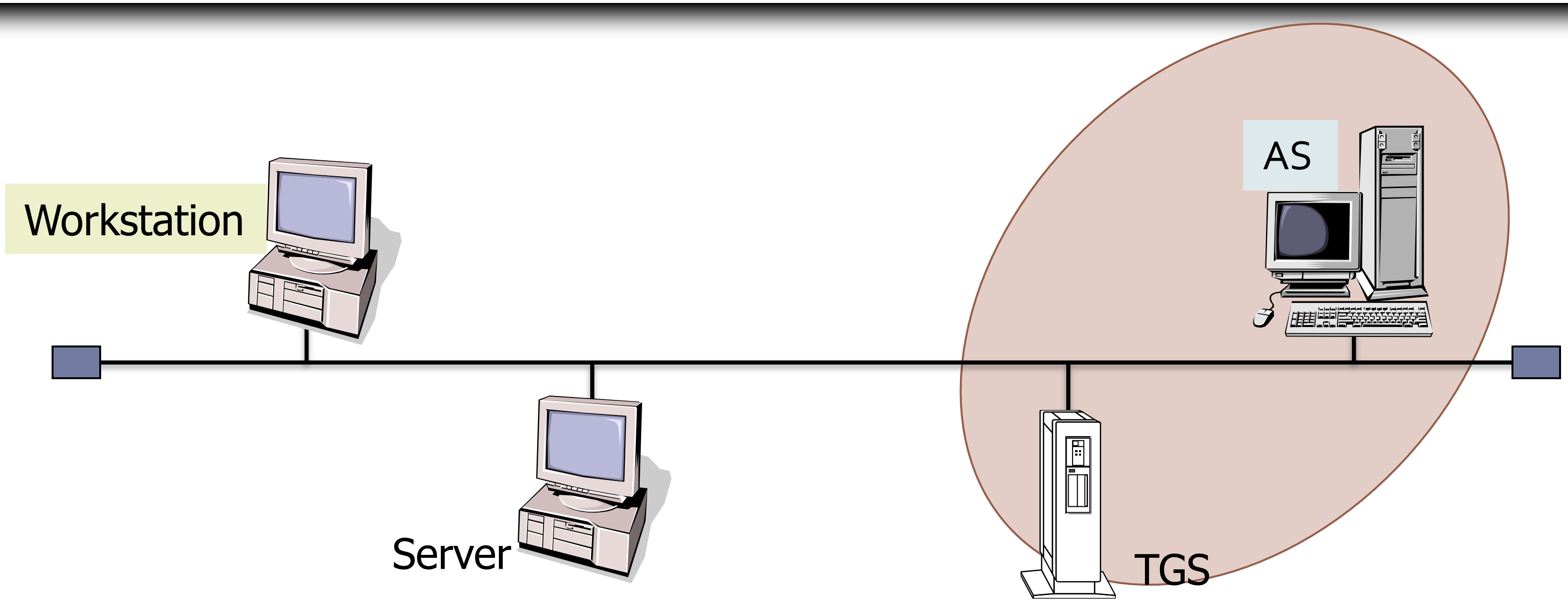
KERBEROS

Kerberos

- Works according to the KDC principle
- Developed at MIT within the Athena project
  - First big Client-Server-campus network
  - Approx. 25.000 users
  - 1.200 computers
- Operational since 1986
- Public Domain
- Current version (5) is standardized in RFC 1510

- **User**
  wants to use a certain service
- **Client**
  is the local Kerberos application
- **Server**
  provides the desired service
- **Authentication Server (AS)**
  is used for primary user authentication
- **Ticket Granting Server (TGS)**
  issues tickets for certain services
- **KDC** includes AS and TGS

Kerberos V4 Accreditation

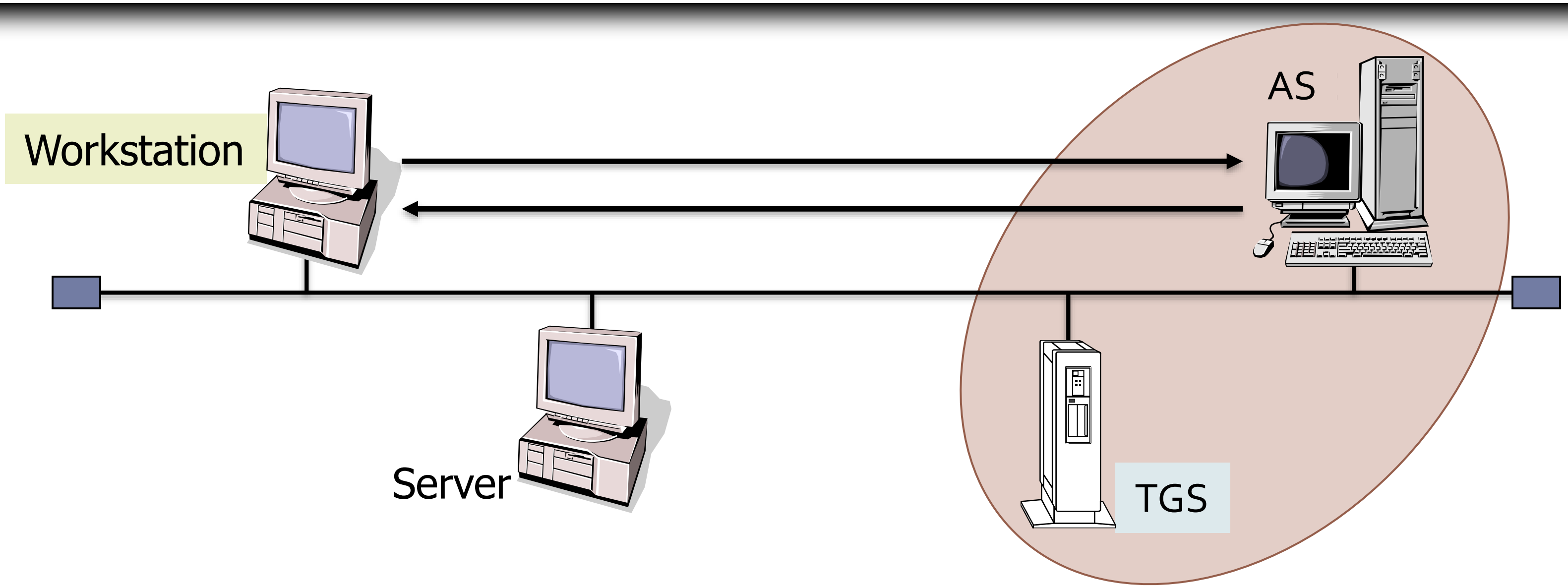


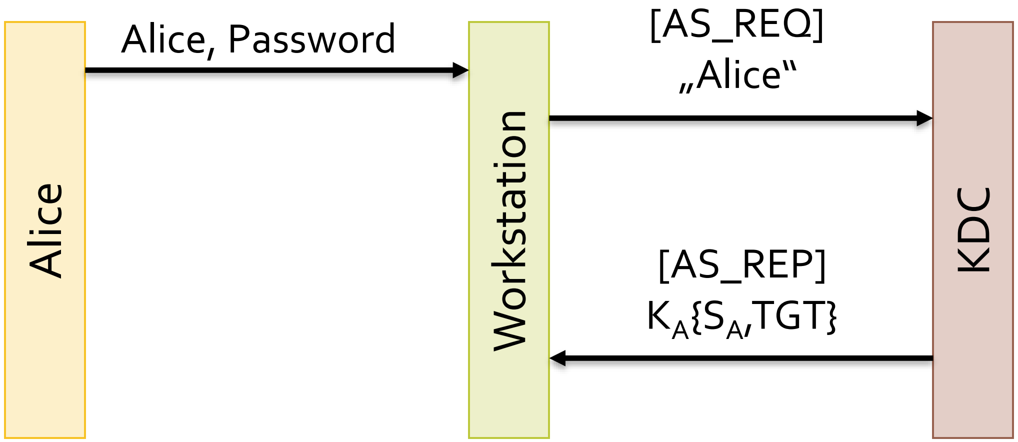
- User and his passwords are provided to the AS.
- TGS and its secret key are also accredited by the AS.
- Server and its secret key are made known to the TGS.

Kerberos V4: Ticket

- Ticket is both user- and server-related:
  - Username
  - Name of the server/service
  - Address of the user computer (optional in V 5)
  - Session key
  - Ticket lifetime (expiry date in V 5)
  - Date of ticket issue

Kerberos V4: Ticket-Granting-Ticket




Alice

Workstation

KDC

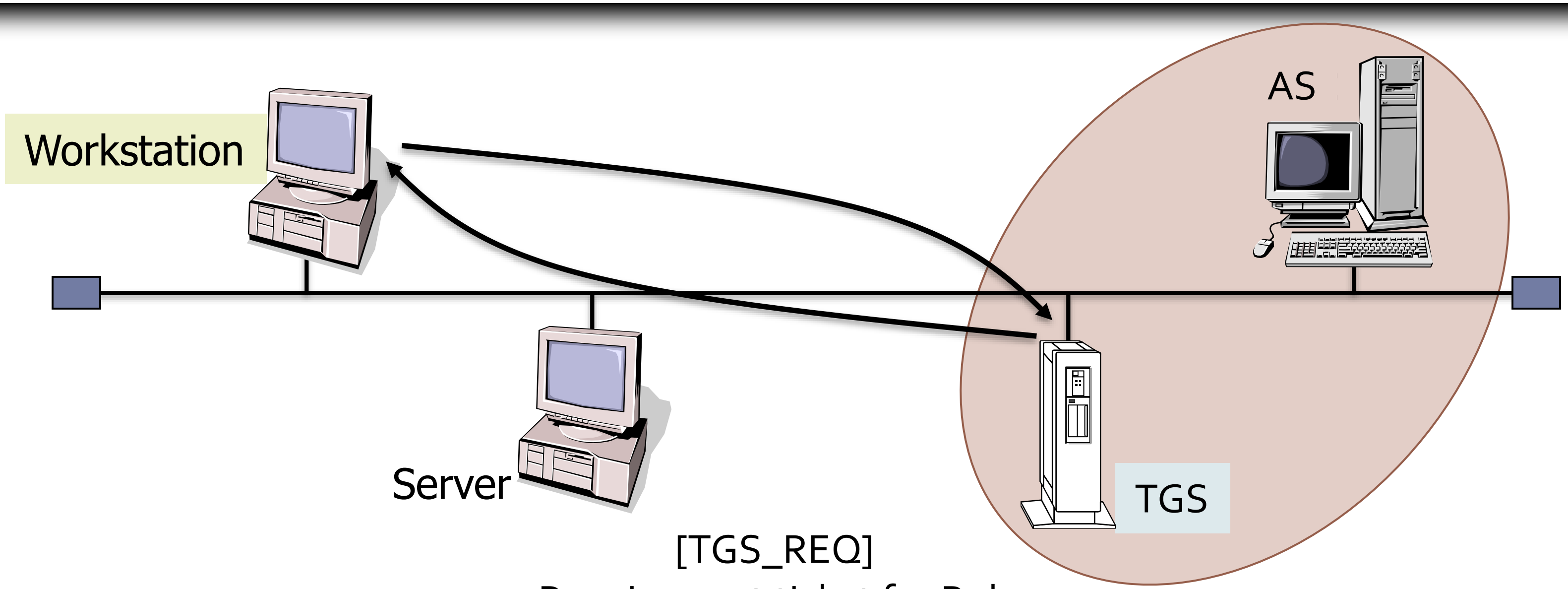
Alice, Password

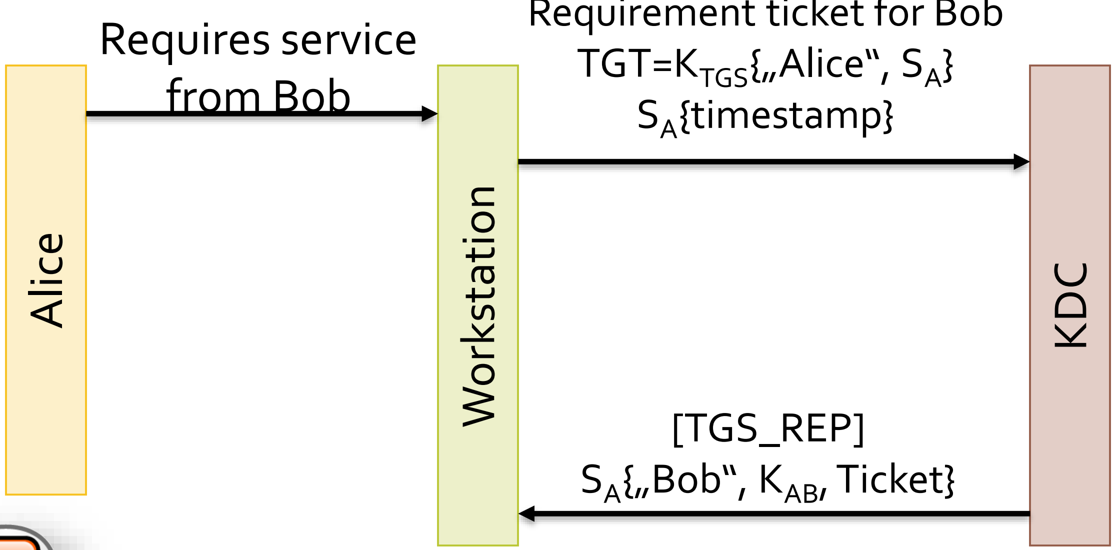
[AS_REQ]
„Alice"

[AS_REP]
$K_A\{S_A, TGT\}$

- Generate session key S_A
- Get master key K_A from Alice
- TGT = $K_{TGS}\{$„Alice", $S_A\}$

Kerberos V4: Service-Ticket





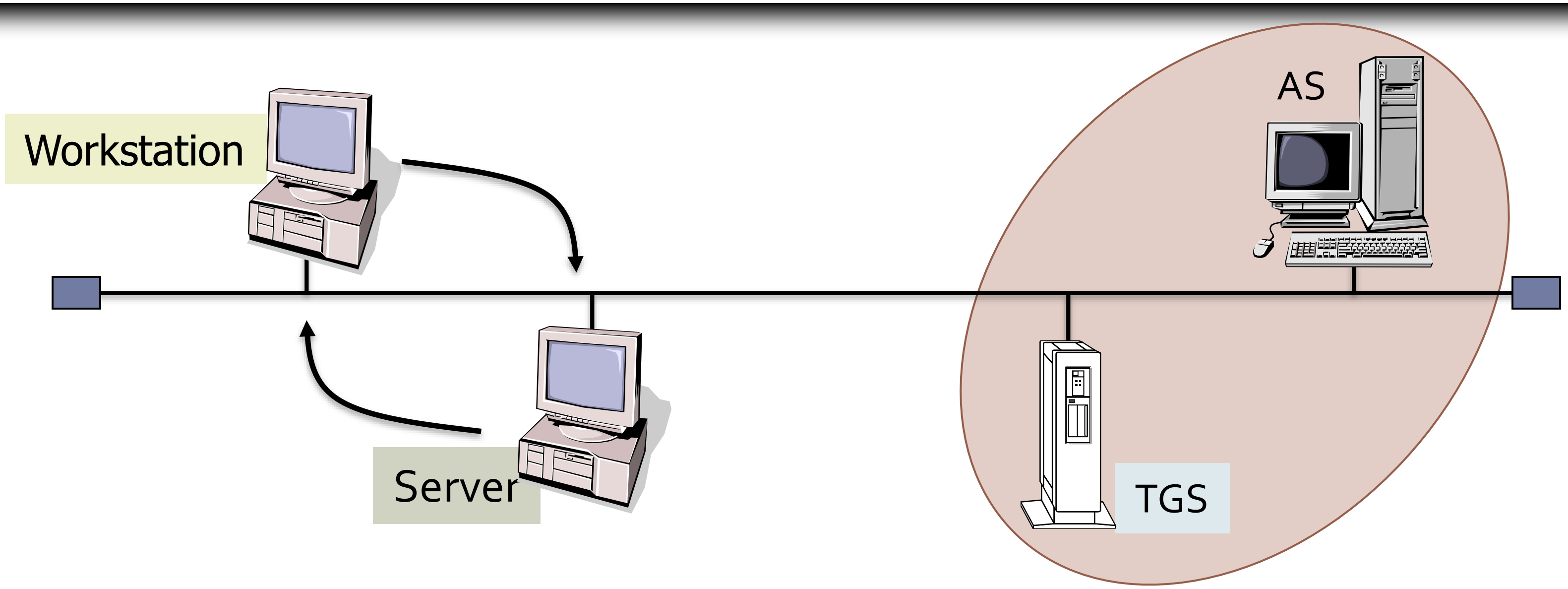
Requires service from Bob

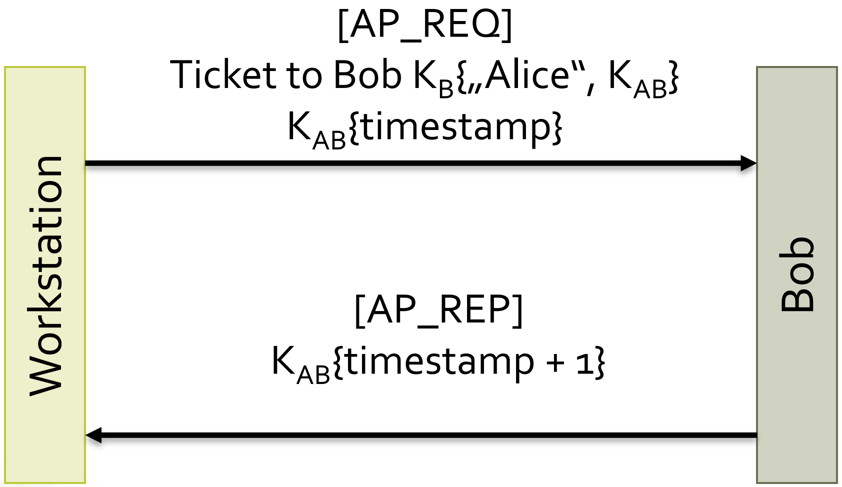
[TGS_REQ]
Requirement ticket for Bob
TGT=K_{TGS}{„Alice“, S_A}
S_A{timestamp}

[TGS_REP]
S_A{„Bob“, K_{AB}, Ticket}

- Generate key K_{AB}
- Decrypt TGT for S_A
- Decrypt S_A{timestamp}
- Verify timestamp
- Get Bob‘s master key K_B
- Ticket = K_B{„Alice“, K_{AB}}

Kerberos V4: Service Use



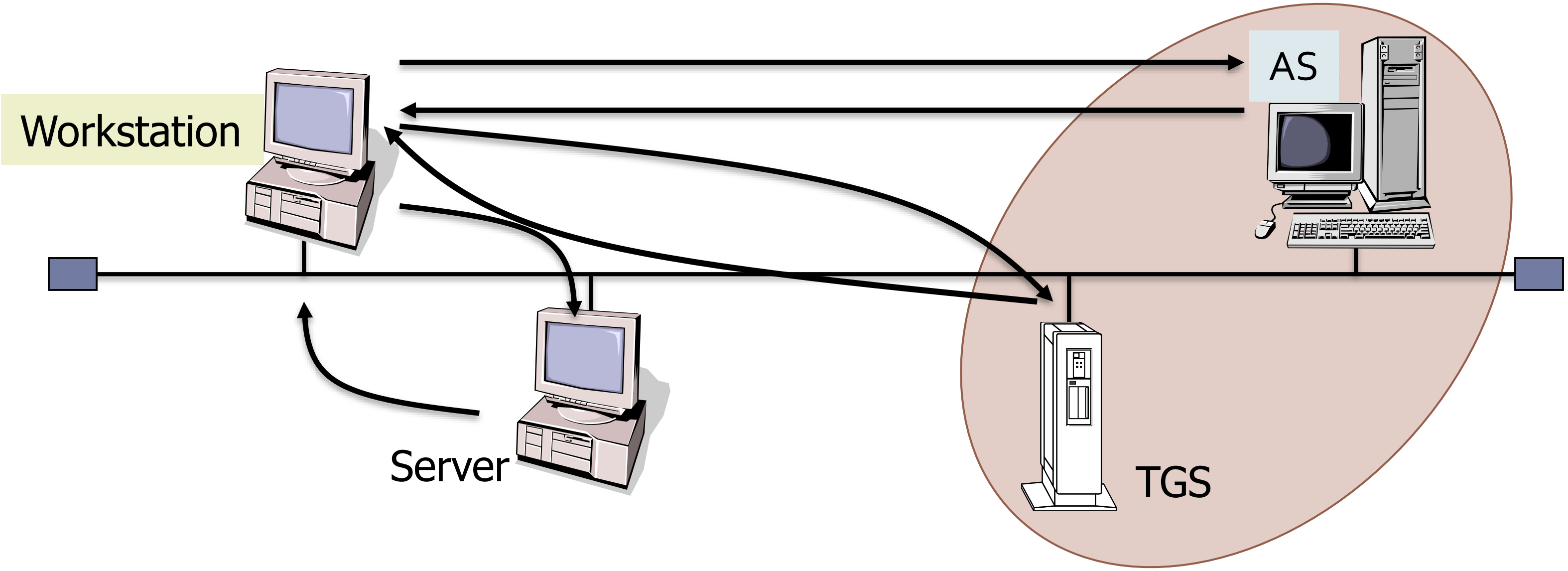


[AP_REQ]
Ticket to Bob K_B{„Alice“, K_{AB}}
K_{AB}{timestamp}

[AP_REP]
K_{AB}{timestamp + 1}

- Decrypt ticket for K_{AB}
- Decrypt K_{AB}{timestamp}
- Verify timestamp

Kerberos V4: Protocol Summary



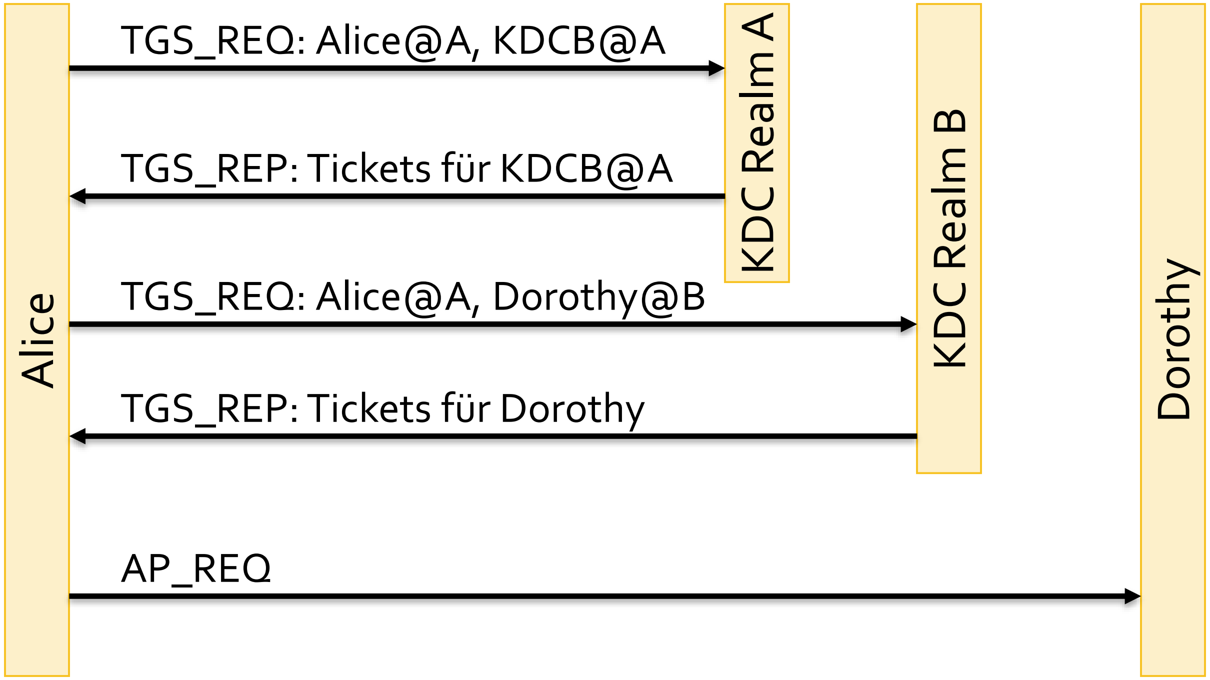
- Two-stage protocol:
  - Client rarely communicates with the KDC.
  - The actual secret key (password) is rarely used.
  - A single TGT is valid for multiple TGS requests.

Kerberos V4: KDC Replication

- Availability: a single KDC constitutes a "Single Point of Failure".
- Availability: a single KDC constitutes a bottleneck.
- Replication of KDCs:
  - *One* shared KDC Master Key
  - Use of identical databases
- One KDC holds the *Master Copy* of the database Master Key; the others synchronize.
- Confidentiality of database Master Key transmission is ensured by hashing with the KDC Master Key; integrity-enforcing procedures must be performed.

Kerberos V4: “Realms“

- Realm → “territory“
- Requires the naming schema to be able to differentiate between Realms.
- Idea: KDC of Realm B acts as a resource in Realm A
  - Holds a shared secret with the KDC of Realm A
- Then: Transitivity utilization


Alice
KDC Realm A
KDC Realm B
Dorothy
TGS_REQ: Alice@A, KDCB@A
TGS_REP: Tickets für KDCB@A
TGS_REQ: Alice@A, Dorothy@B
TGS_REP: Tickets für Dorothy
AP_REQ