

Sicherheit in Kommunikationsnetzen (Network Security)

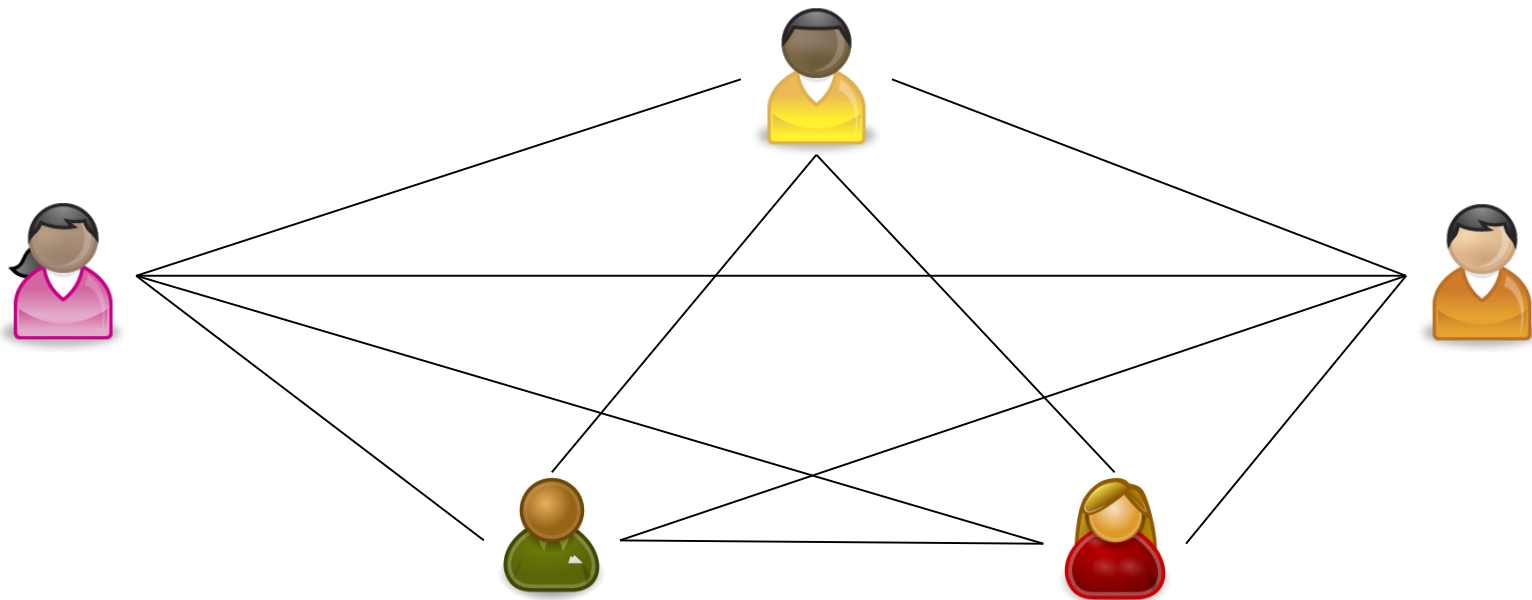
Key Management

Dr.-Ing. Matthäus Wander

Universität Duisburg-Essen

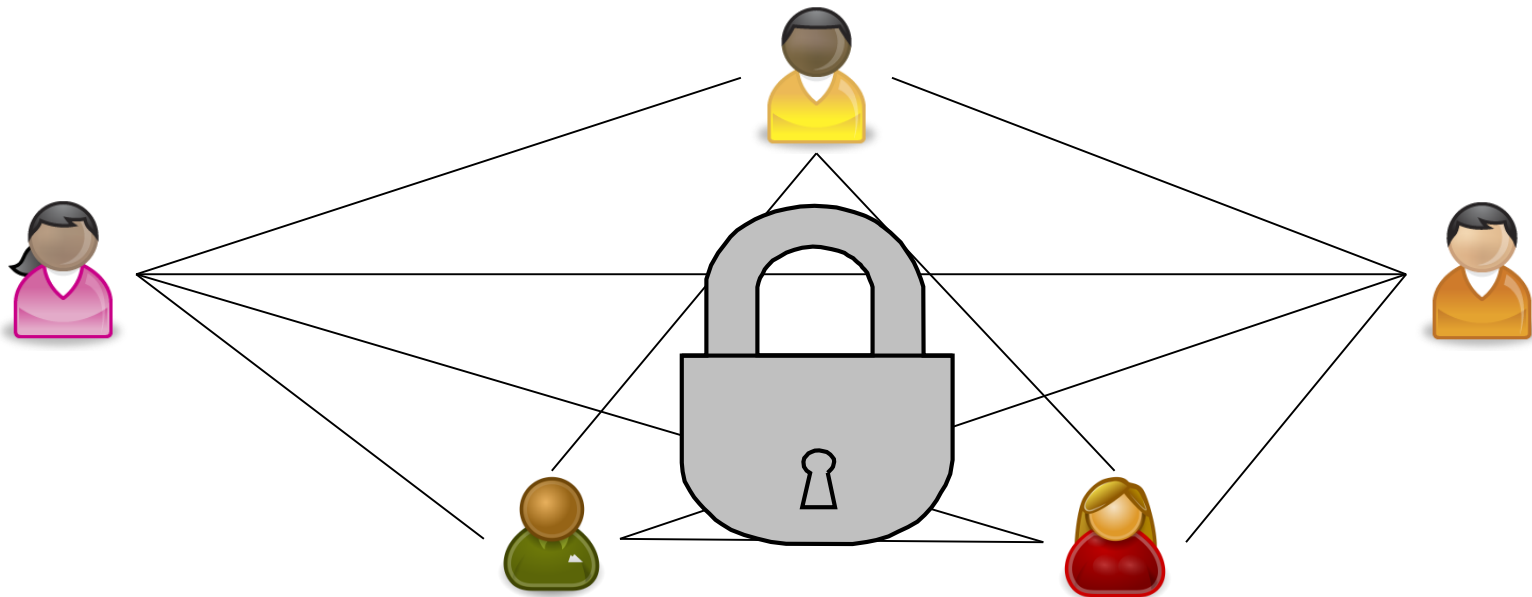
Symmetric Key Distribution

- How to negotiate **secret keys** between users?
 - Problem: scalability (one key for every pair of users)
 - Impractical with lots of users



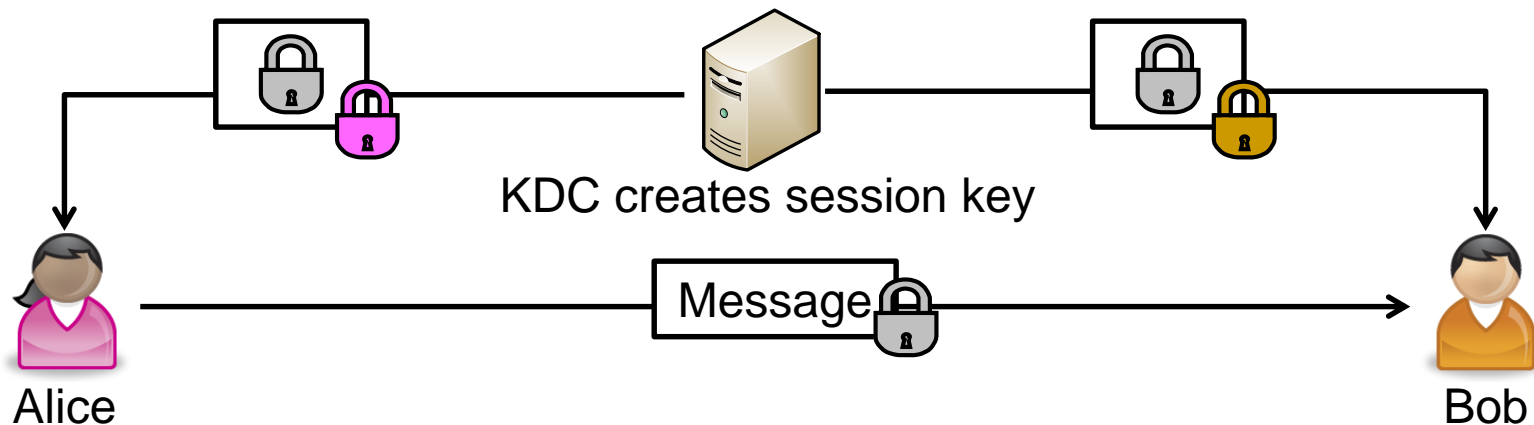
Group Key

- Idea: one group key
 - Adding new user: send key to new user
 - Remove user: re-distribute new group key
 - Each user is fully trusted, key disclosure fatal for all



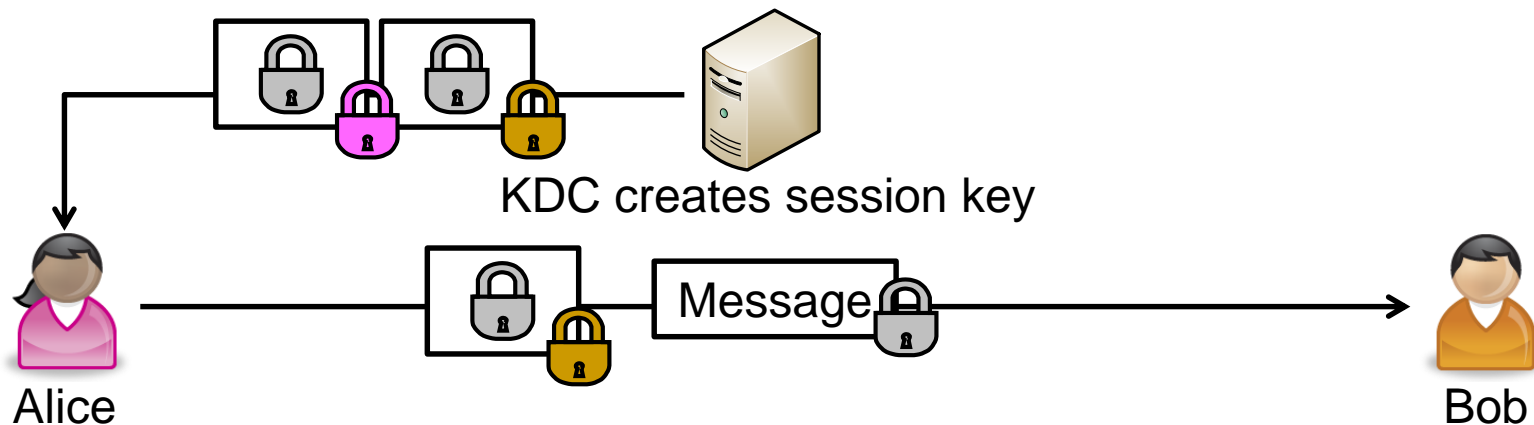
Key Distribution Center

- Idea: trusted **key distribution center** (KDC)
 - Negotiate master key between user and KDC
 - KDC creates on demand **session key** between Alice/Bob
 - Encrypt session key with **Alice's key** and **Bob's key**



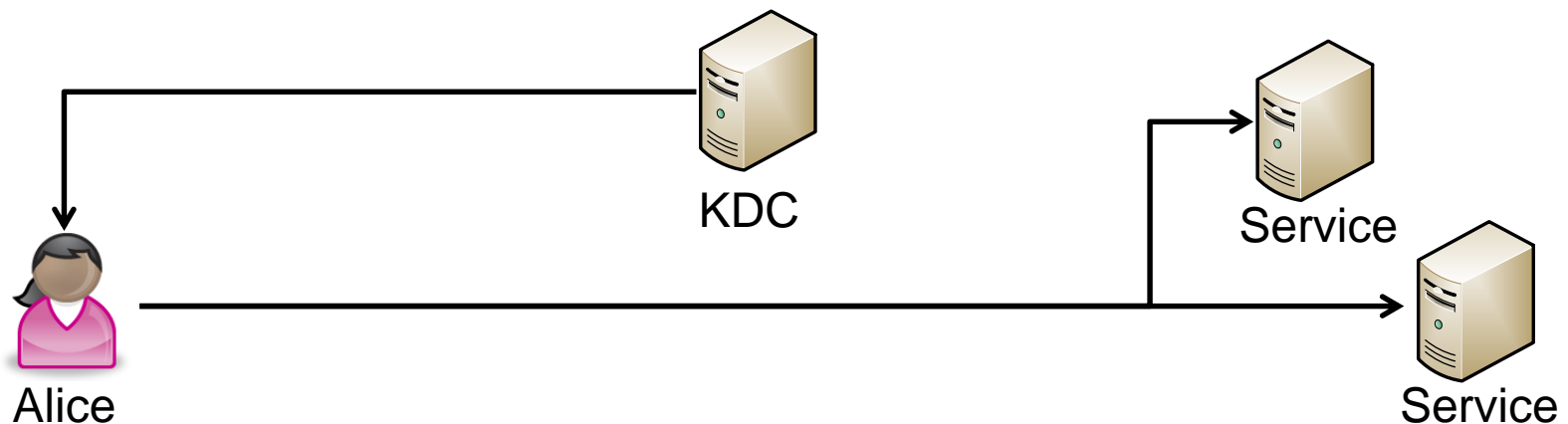
Key Distribution Center (2)

- KDC does not need to communicate with Bob
 - Piggyback session key for Bob over Alice's messages
- After key establishment, mutual authentication is needed
 - Prevent **replay attacks** with **nonce** and **timestamp**



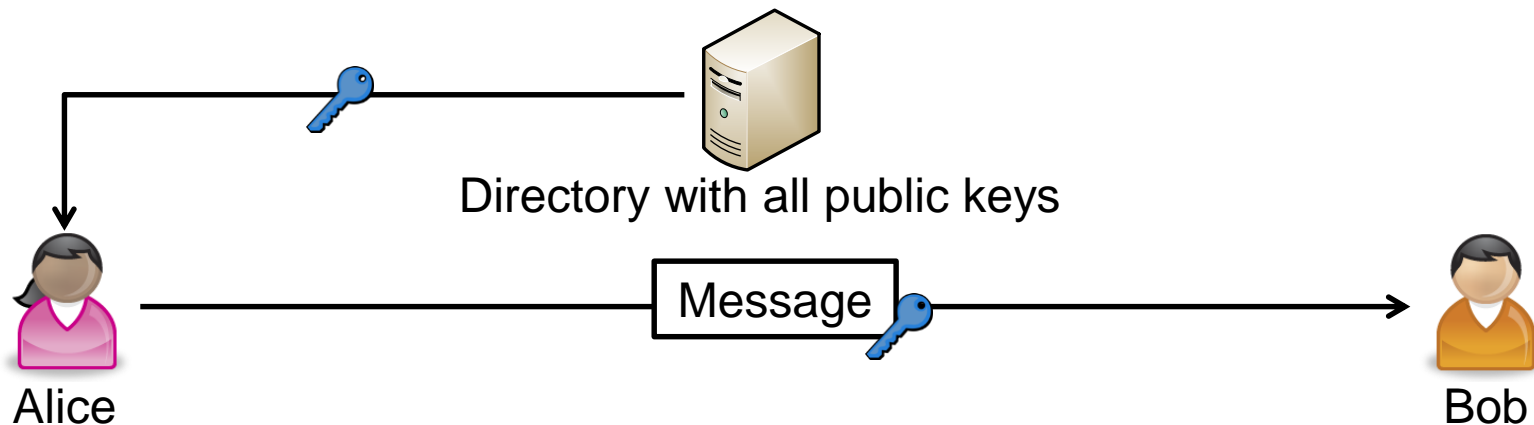
Key Distribution Center (3)

- With a KDC, we can implement **authentication** and **authorization** of users or clients
 - KDC checks: is the user allowed to access a service?
 - e.g. Kerberos: KDC creates session tickets
- Use case: **single sign-on** (centralized user login)



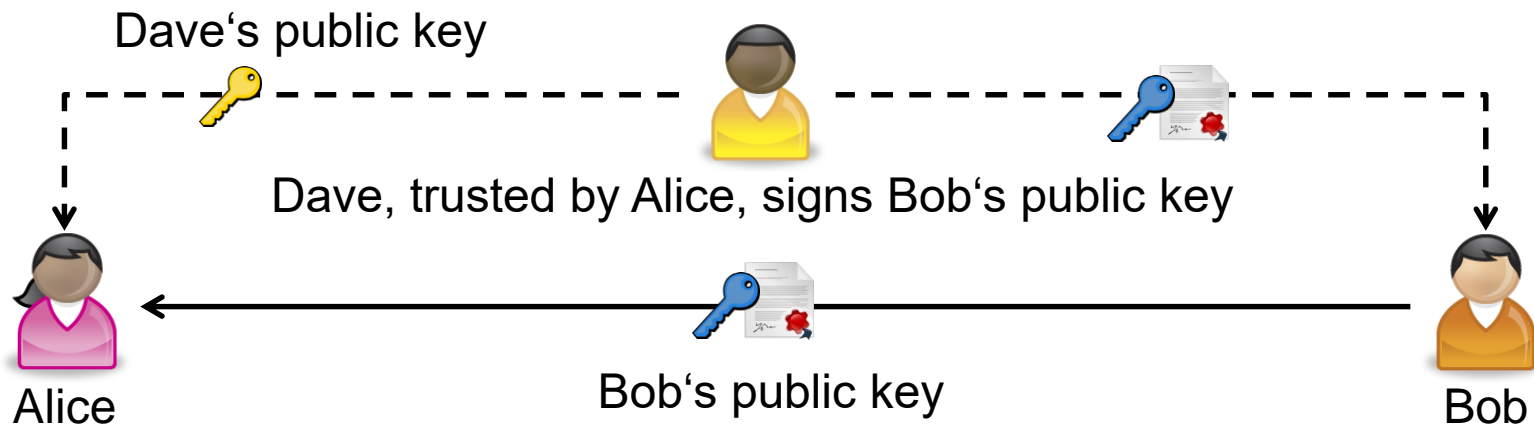
Public–Key Infrastructure

- Use a **public–key infrastructure**
 - Each user has a private/public key pair
- Problem: how to distribute public keys?
- Save public keys on a trusted directory server
 - Problem: bottleneck, scalability



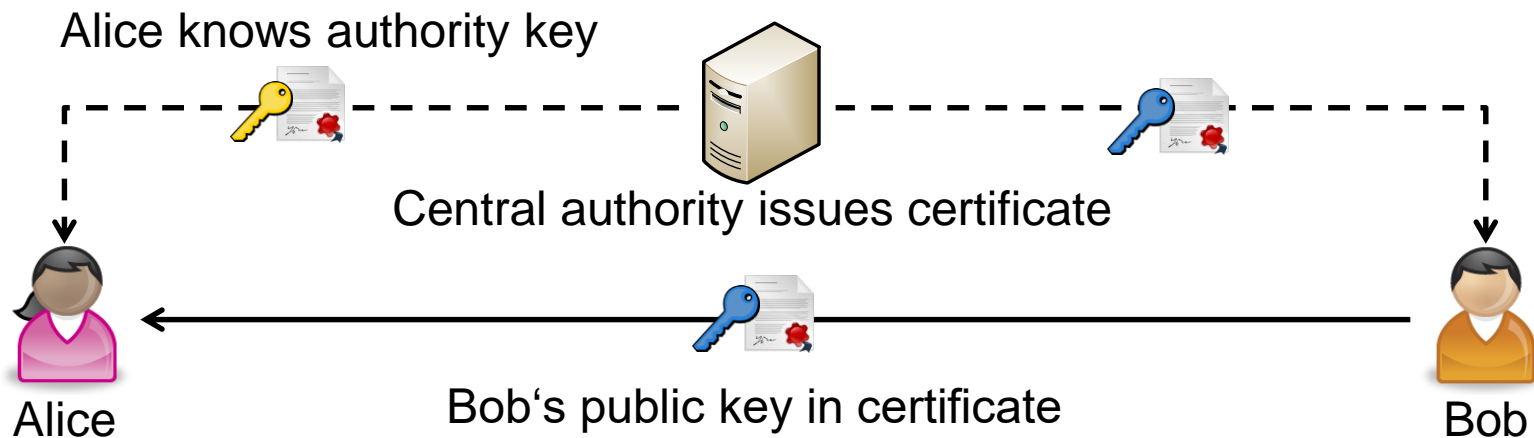
Public–Key Infrastructure (2)

- Distribute public keys directly between users
- How to authenticate public keys?
- **Web of trust:** transitive trust between users
 - Somebody you know attests the authenticity of somebody else and signs their public key



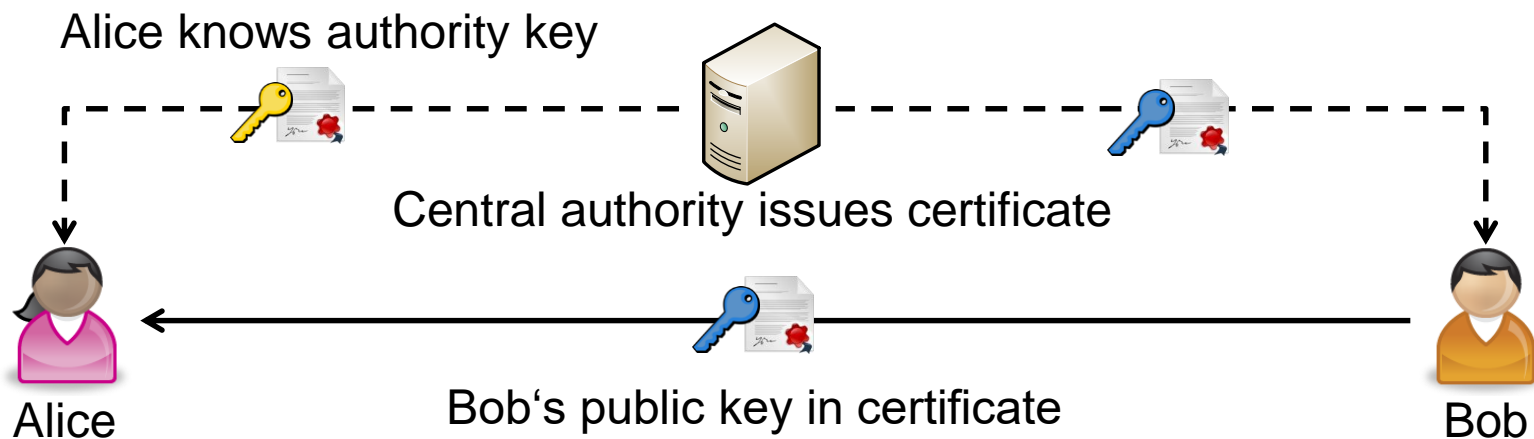
Public–Key Infrastructure (3)

- **Authority:** centralized trust
 - One (or a couple) of authorities attest the authenticity and sign public keys of all (or many) users
- Trusted authority issues **certificate** to Bob
 - Contains public key of Bob, signed by authority



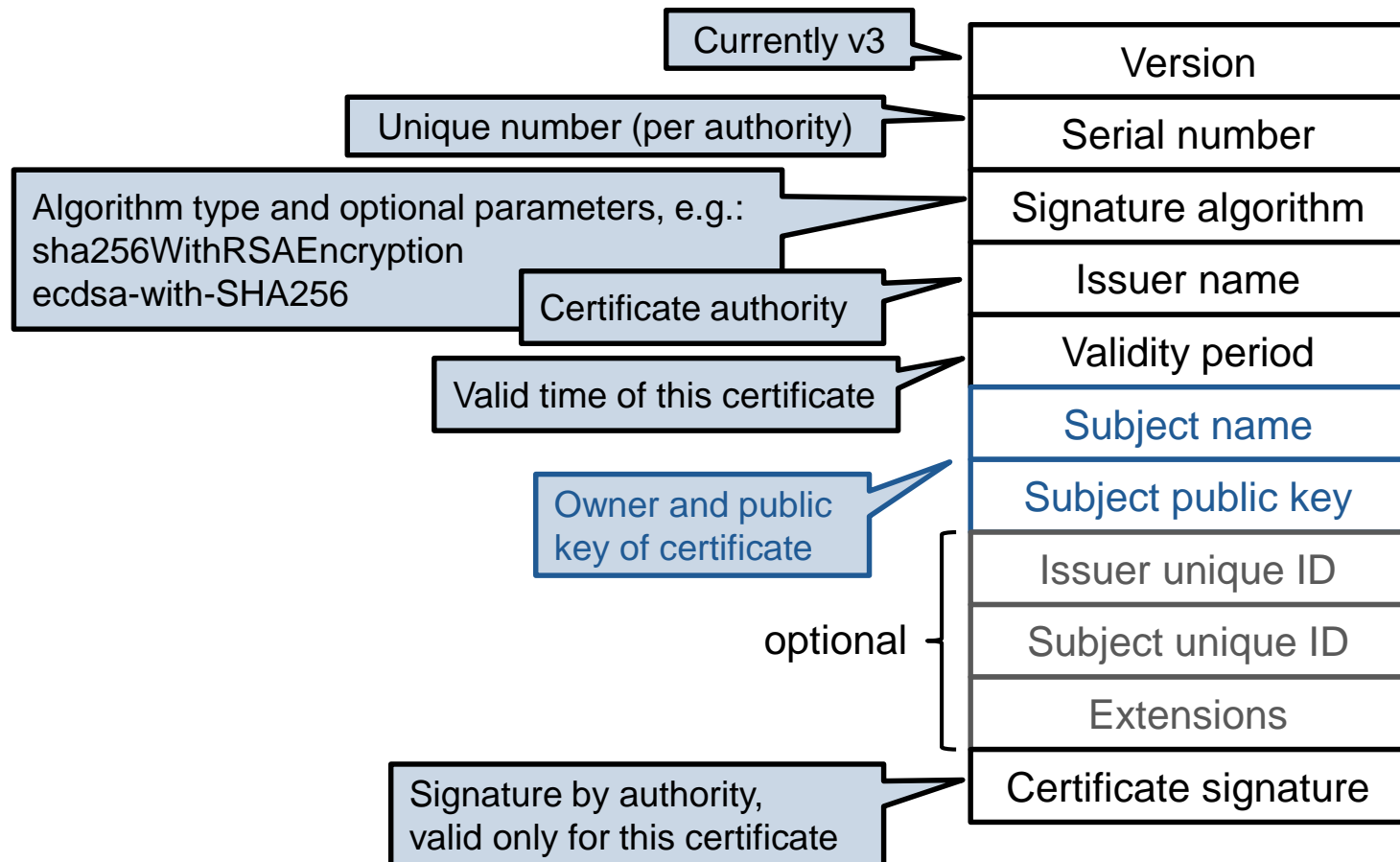
Public–Key Infrastructure (4)

- Alice has a list of trusted certificate authorities
- Certificates contain long–term keys
 - Bob’s certificate is valid for months or years
- Benefit: little interaction with authority required



X.509

- X.509 is a standard for digital certificates



Example Certificate for Secure Email

The image displays three overlapping screenshots of a Windows Certificate dialog box, illustrating the details of a certificate used for secure email.

Leftmost screenshot (General tab):

- Certificate Information:** This certificate is intended for the following purpose(s):
 - Protects e-mail messages
 - Proves your identity to a remote computer
 - 1.3.6.1.4.1.22177.300.1.1.4
- Issued to:** Matthaues Wander
- Issued by:** Universitaet Duisburg-Essen CA -G01
- Valid from:** 2015-06-08 to 2018-06-07
- Key icon: You have a private key that corresponds to this certificate.

Middle screenshot (Details tab):

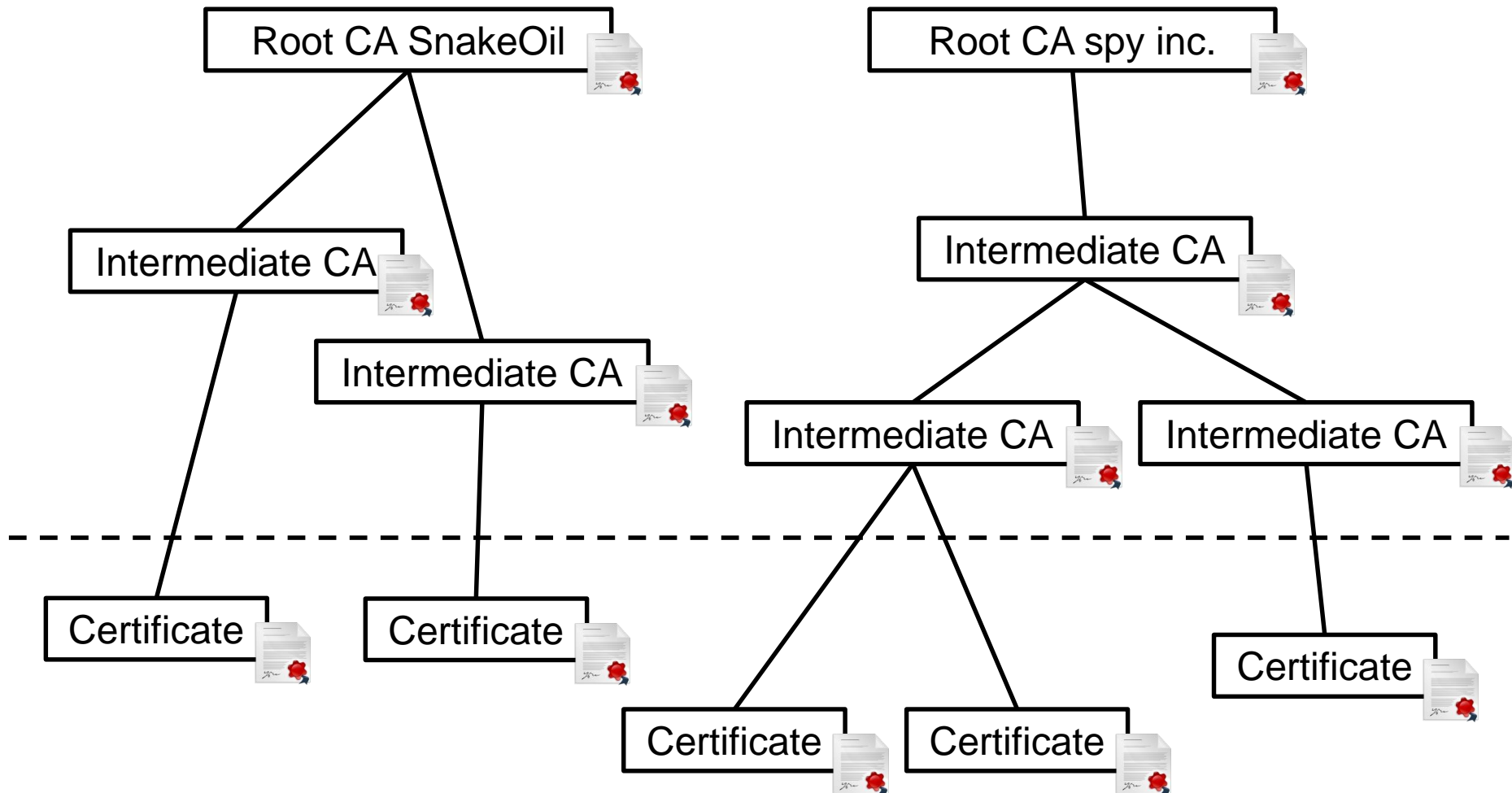
Field	Value
Signature hash algorithm	sha256
Issuer	caadmin@uni-duisburg-esse
Valid from	Montag, 8. Juni 2015 11:18
Valid to	Donnerstag, 7. Juni 2018 11:18
Subject	Matthaues Wander, Verteilte Systeme
Public key	RSA (2048 Bits)
Certificate Policies	[1]Certificate Policy:Policy Id:1.3.6.1.4.1.22177.300.1.1.4
Basic Constraints	Subject Type=End Entity, Path Length=0

Rightmost screenshot (Details tab):

Field	Value
Basic Constraints	Subject Type=End Entity, Path Length=0
Key Usage	Digital Signature, Non-Repudiation
Enhanced Key Usage	Client Authentication (1.3.6.1.4.1.3.2.1.1)
Subject Key Identifier	b2 e4 e6 b7 7b 13 fe 0f 52 79 ...
Authority Key Identifier	KeyID=66 66 bc 96 9b 48 4d e...
Subject Alternative Name	RFC822 Name=matthaues.wander@uni-due.de
CRL Distribution Points	[1]CRL Distribution Point: Distr...
Authority Information Access	[1]Authority Info Access: Acc...

The RFC822 Name field is expanded to show: RFC822 Name=matthaues.wander@uni-due.de

X.509 Certificate Authorities



X.509 Certificate Authorities

- Hierarchy of **certificate authorities (CAs)**
 - Root CA certificates are installed on hosts
 - Root CA signs intermediate CA
 - Intermediate CA signs subject (e.g. user) certificate
- Each trusted CA (including intermediate) can issue certificate for **any** subject
 - With or without authorization of server owner
 - No transparency about issued certificates

(A few) Problems of X.509

- **Name constraints** suggested to restrict CAs
 - e.g. issue certificate only below *.domain.tld
 - Hardly supported by applications, i.e. ignored
- A few dozen root CAs shipped with operating system and applications (e.g. web browsers)
 - Managed primarily by vendors, not users
 - CAs occasionally get hacked (DigiNotar, StartCom, ...) or ignore security policies (e.g. backdated certificates)
 - Withdrawing CA trust can impact thousands of users

Suggested Solutions for X.509

- **Certificate Transparency** for CAs
 - Standard for publishing a log of issued certificates
 - Log is cryptographically secured
 - Monitors and auditors check logs for correctness
- **Certificate pinning** in applications
 - Hard-coded public keys or certificates in applications
 - Application checks whether received certificate strictly matches hard-coded certificate
 - ... or whether it is issued by a hard-coded CA

Certificate Revocation

- In case of a security mishap, certificates can be **revoked** (declared as invalid)
 - Various problems in practice, revocation often ignored
- CA publishes **Certificate Revocation List (CRL)**
 - Flat list, does not scale
- **Online Certificate Status Protocol (OCSP)**
 - Check revocation status via network request to CA
 - Problems: reliability, downgrade attacks, privacy
 - Mitigated by **OCSP stapling**: server appends cached OCSP response to certificate

Sicherheit in Kommunikationsnetzen (Network Security)

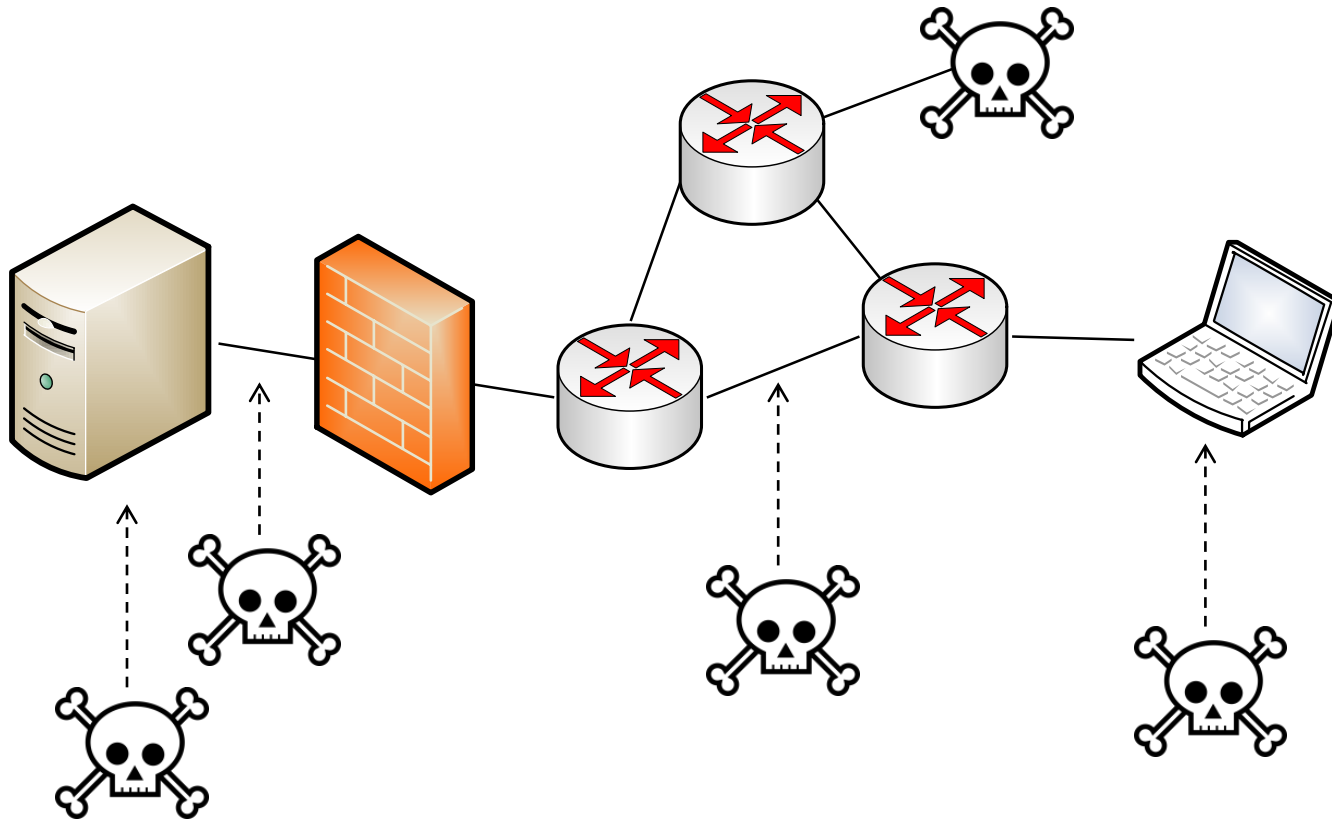
Security Infrastructures

Dr.-Ing. Matthäus Wander

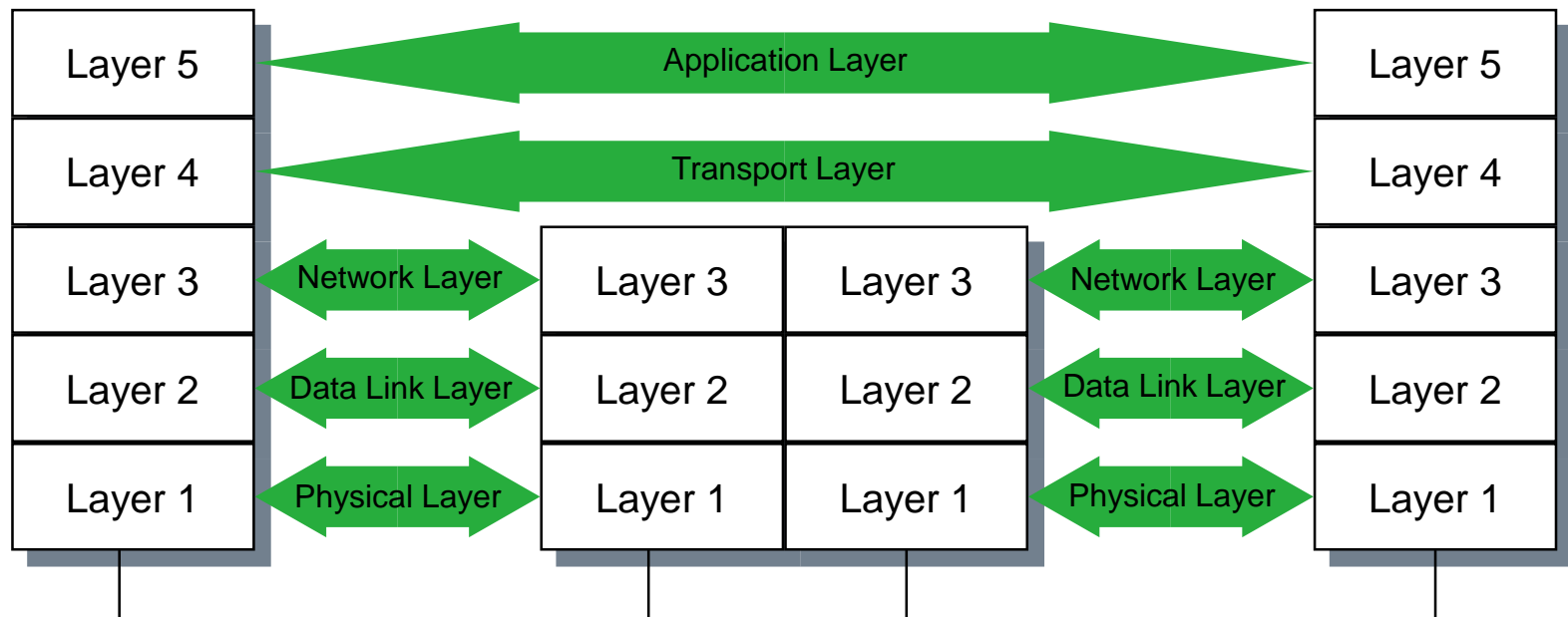
Universität Duisburg-Essen

Threat Model

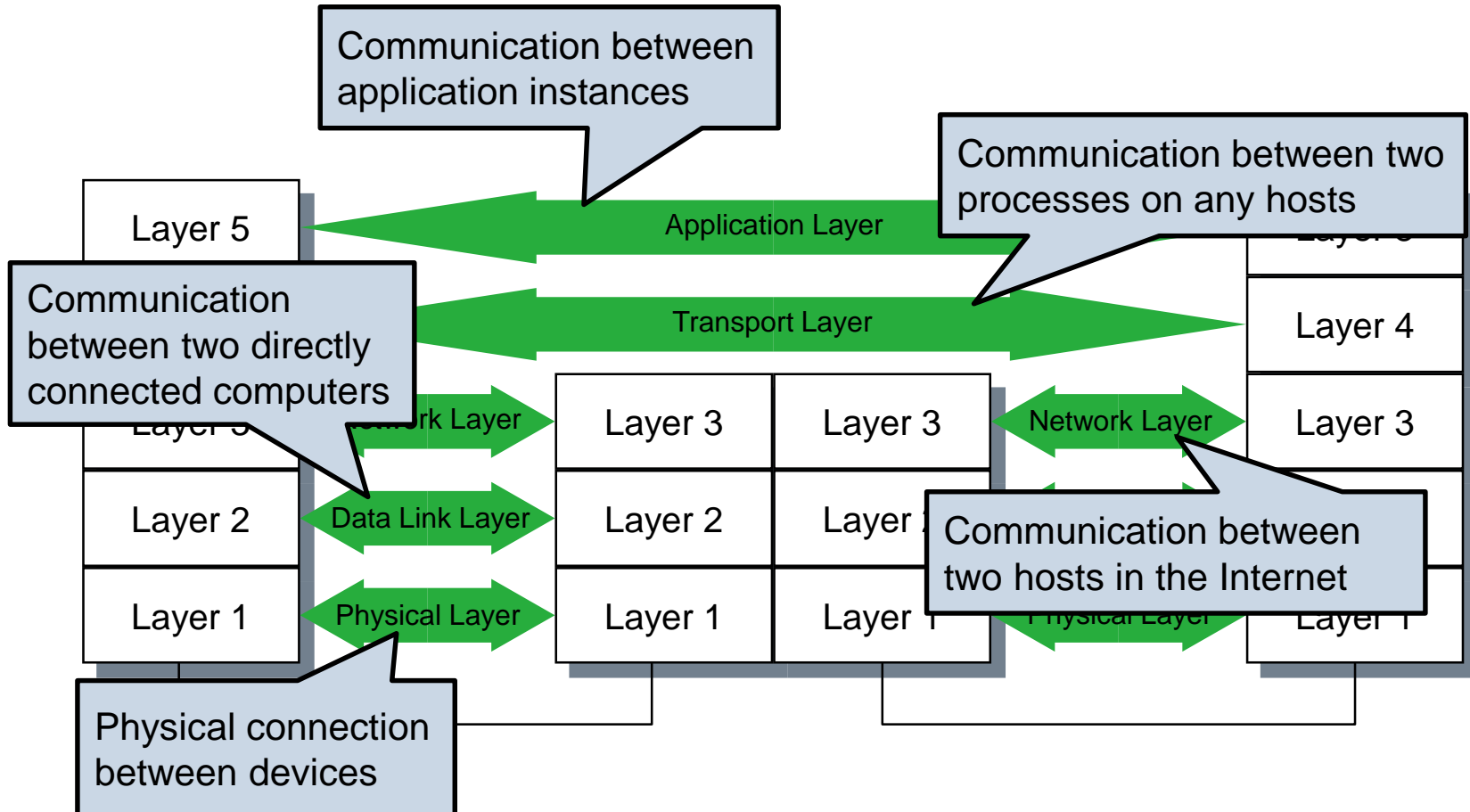
- Where is the attacker located?



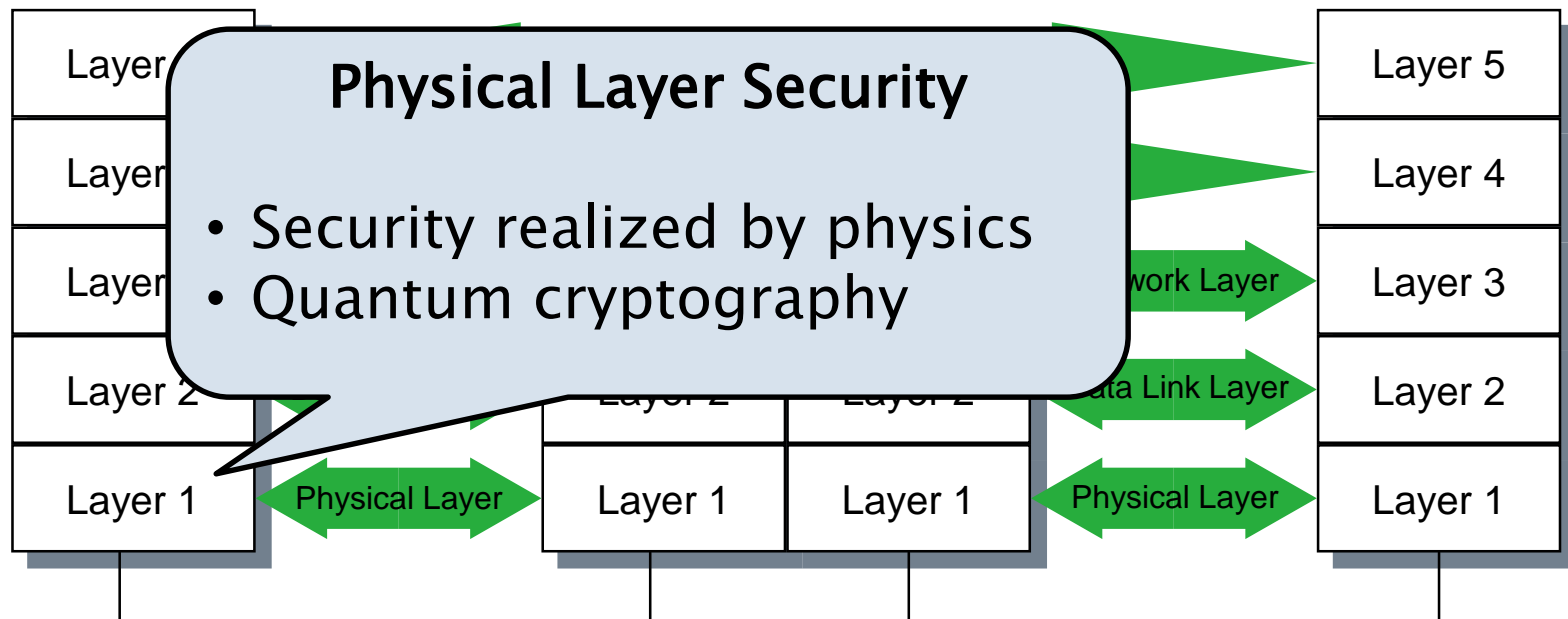
Protocol Layers



Protocol Layers (2)



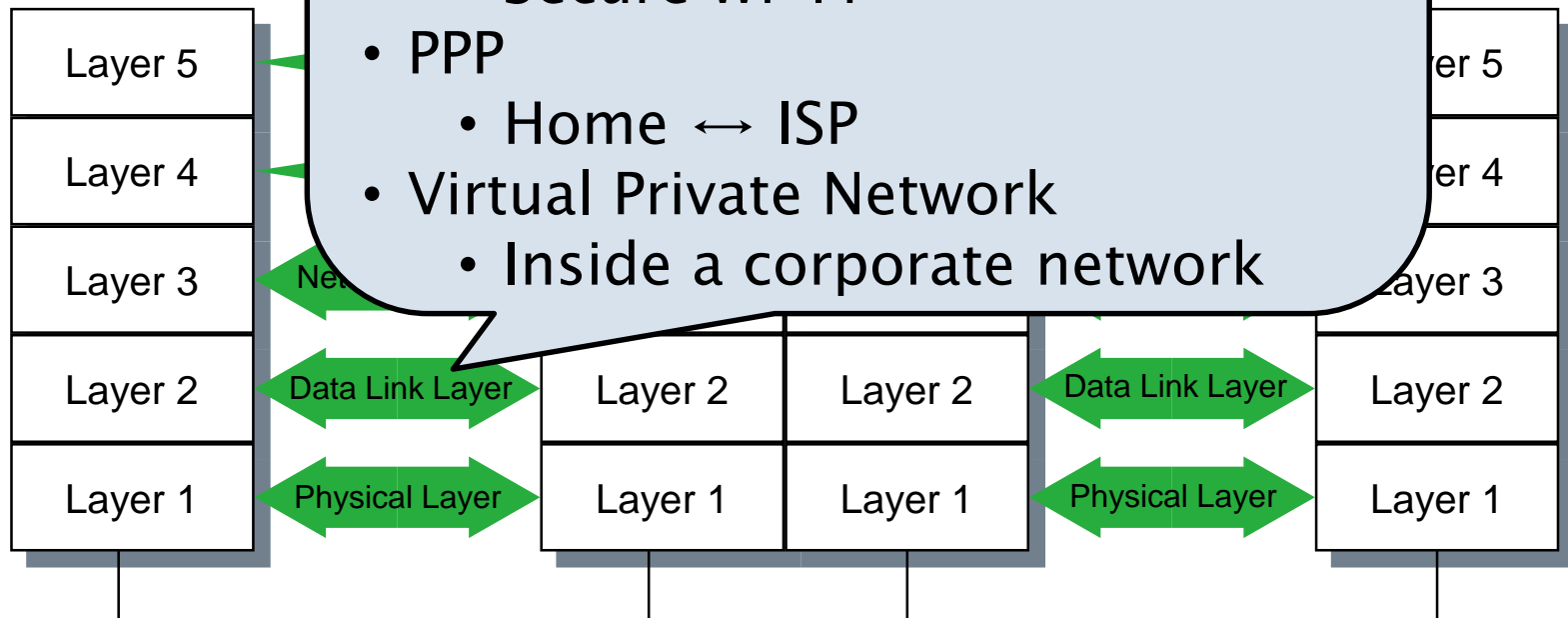
Protocol Layers (3)



Protocol L

Data-Link Layer Security

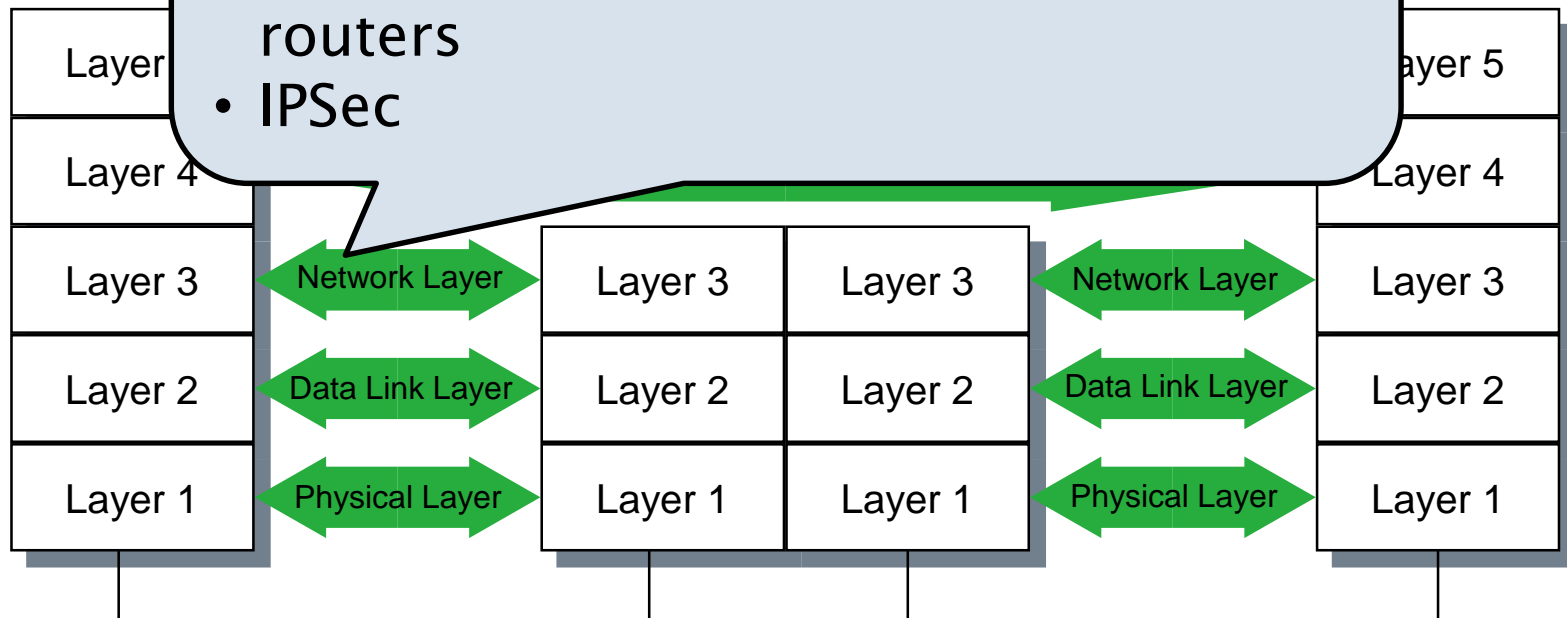
- Implemented by network interface cards, switches, etc.
- WPA2
 - Secure Wi-Fi
- PPP
 - Home ↔ ISP
- Virtual Private Network
 - Inside a corporate network



Protocol Layers (5)

Network Layer Security

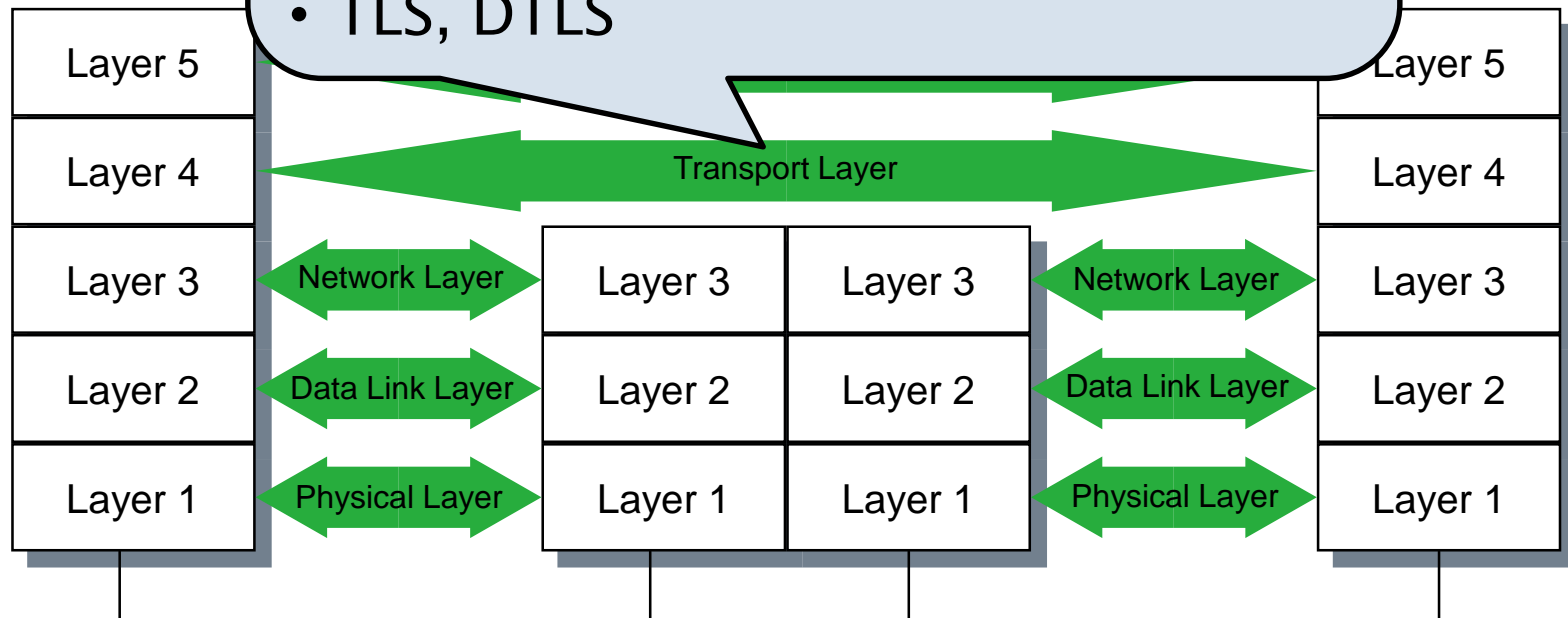
- Security on the IP packet level
- Implemented by operating system and routers
- IPSec



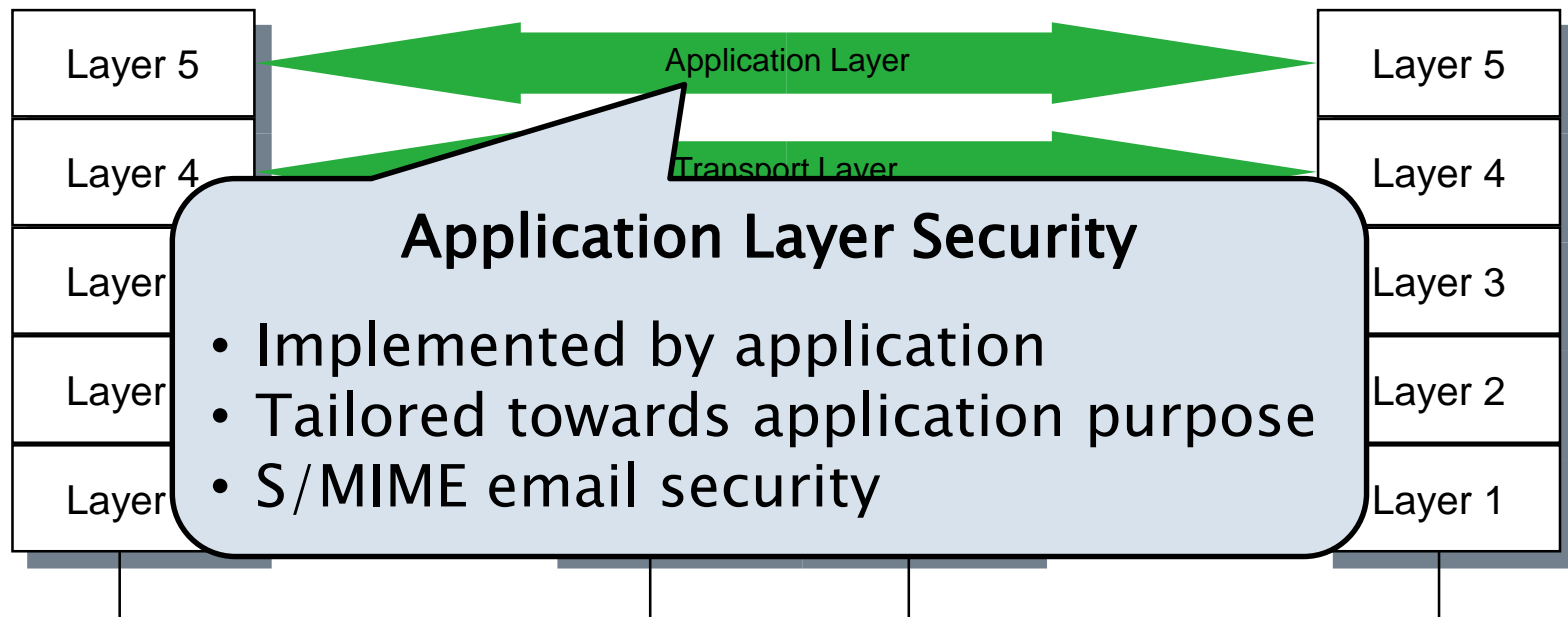
Protocol

Transport Layer Security

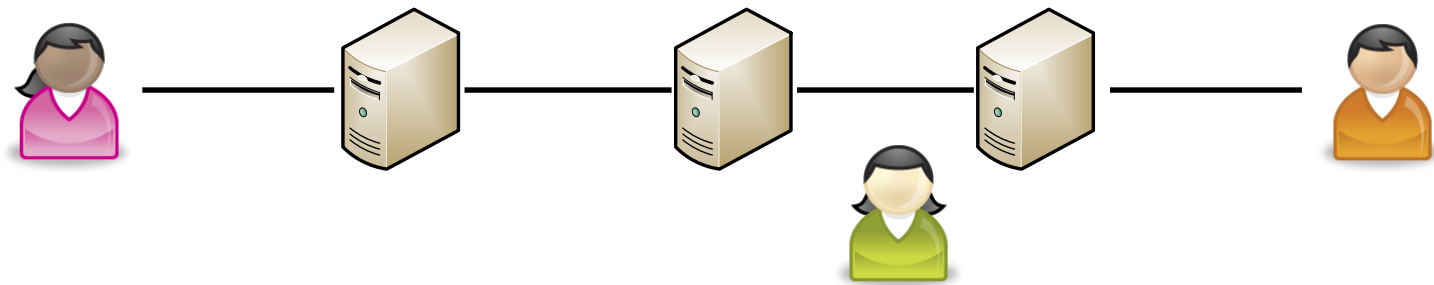
- Implemented by operating system or by application
- Works on top of Sockets
- TLS, DTLS



Protocol Layers (7)

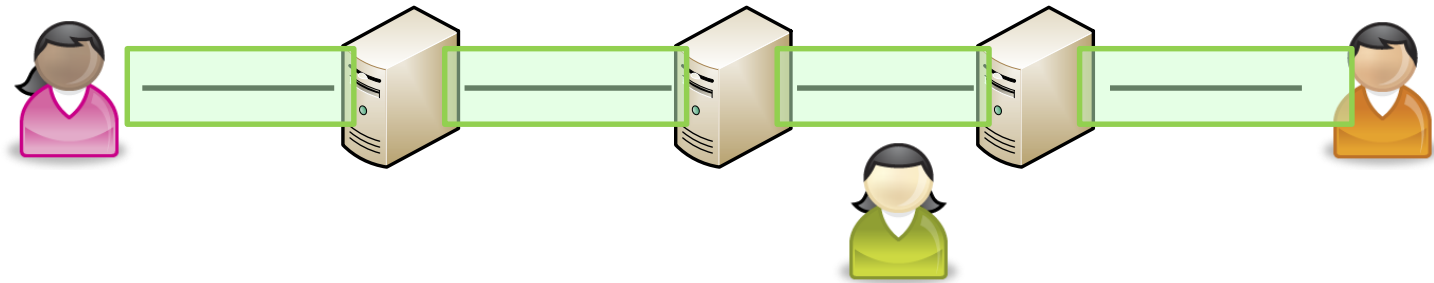


System Model



- Who talks with whom?
- Who is trusted?
- Who needs access to which data?
 - Are the servers only for storage and communication?
 - Or also for processing of our data?

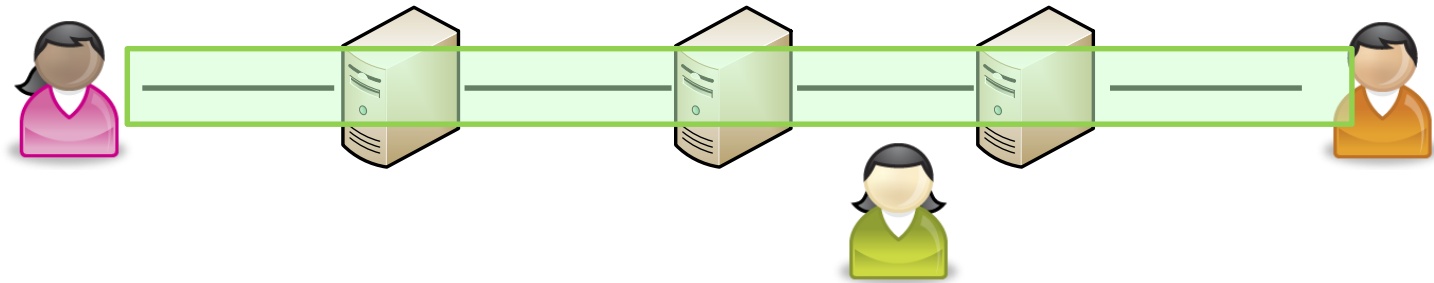
System Model (2)



- Point-to-point security

- Secures communication link between two components
- Each component is trusted to comply with security
- Servers process plaintext messages
- In multi-hop communication, each link is independent from the others (but all of them need to be secure!)

System Model (3)



- End-to-end security

- Secures communication between two endpoints
- Only endpoints are trusted to comply with security
- Servers forward secure messages and see metadata, but cannot tamper nor assist with the contents
- Usually harder to achieve than point-to-point security