Network Security (NetSec)

TECHNISCHE UNIVERSITÄT DARMSTADT

Summer 2015
Chapter 03: Application Level Security
Module 01: What (and what not) to Secure on APP Level

Prof. Dr.-Ing. Matthias Hollick

Technische Universität Darmstadt
Secure Mobile Networking Lab - SEEMOO
Department of Computer Science
Center for Advanced Security Research Darmstadt - CASED

Mornewegstr. 32
D-64293 Darmstadt, Germany
Tel.+49 6151 16-70922, Fax. +49 6151 16-70921
http://seemoo.de or http://www.seemoo.tu-darmstadt.de

Prof. Dr.-Ing. Matthias Hollick
matthias.hollick@seemoo.tu-darmstadt.de

CASED

Learning Objectives & Overview

SECURE MOBILE NETWORKING

Learning objectives

- Discuss a pragmatic model on securing networks and identify which security objectives can (should) be obtained at the application layer
- Critically discuss the trade-offs/limitations of application level security
- Understand selected security issues on application layer by studying protocols/mechanisms (in separate modules)

Outline

(1) Visualizing protection

(2) A pragmatic network security model (what, how, where to secure)

(3) Relationship between layers and requirement levels

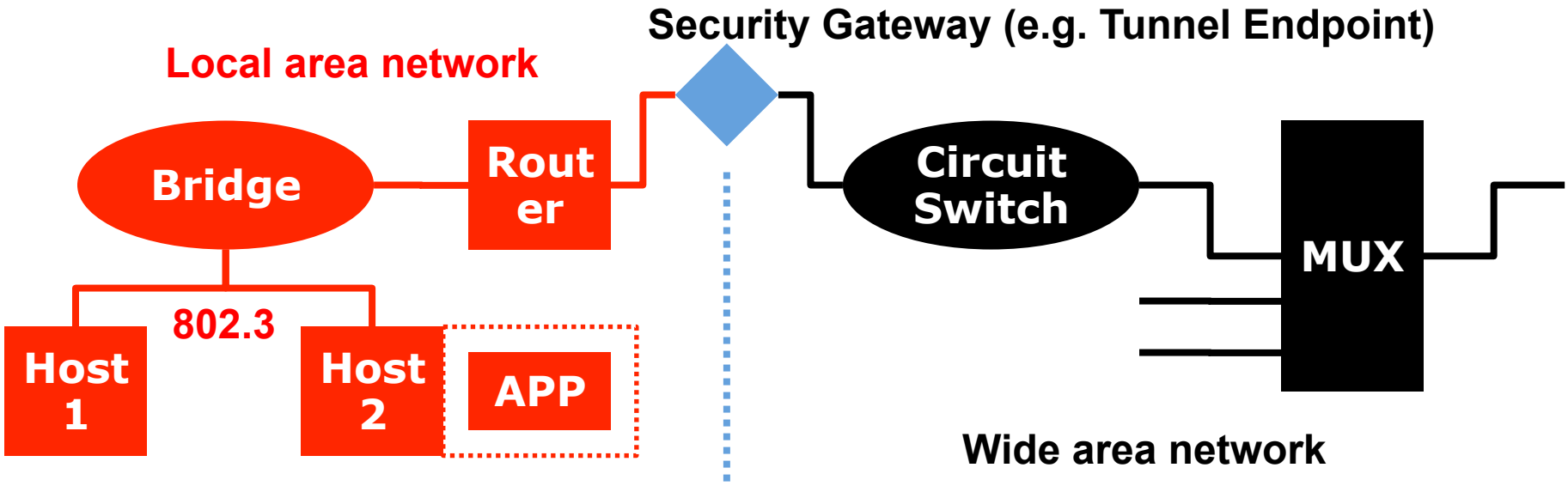
(4) Security on application layers vs. lower layers: trade-offs

Chapter 03, Module 01

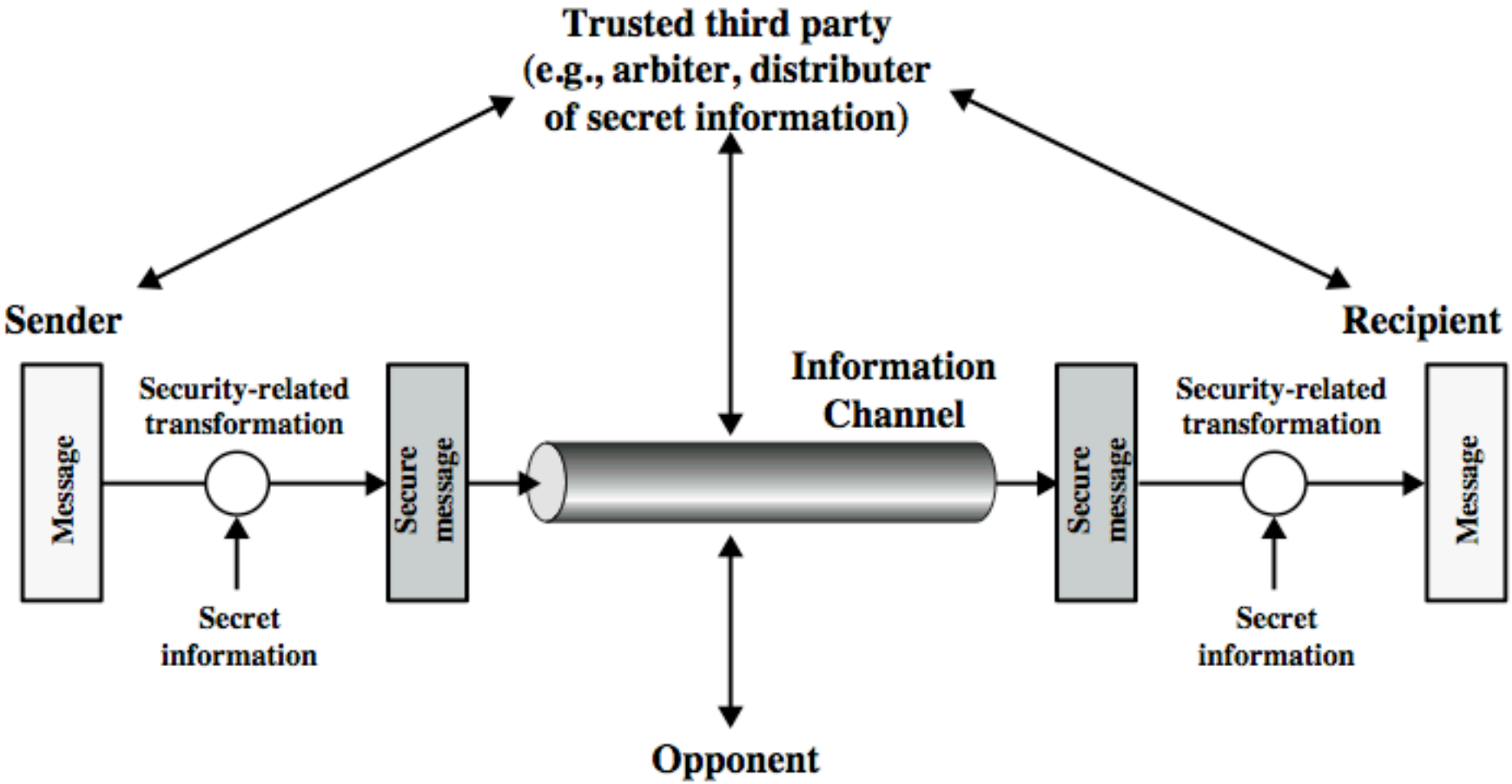
Visualizing Protection Areas

Secure vs. non-secure parts of systems are typically segregated

- Labeling (and drawing) systems/network elements as red and black usually refers to their “level of protection”
- Red signals or parts of a network are unencrypted/unprotected
- Black signals or parts of a network are encrypted/protected



Last Module: Abstract Model for Network Security



Source: book of Stallings

What, How and Where to Secure

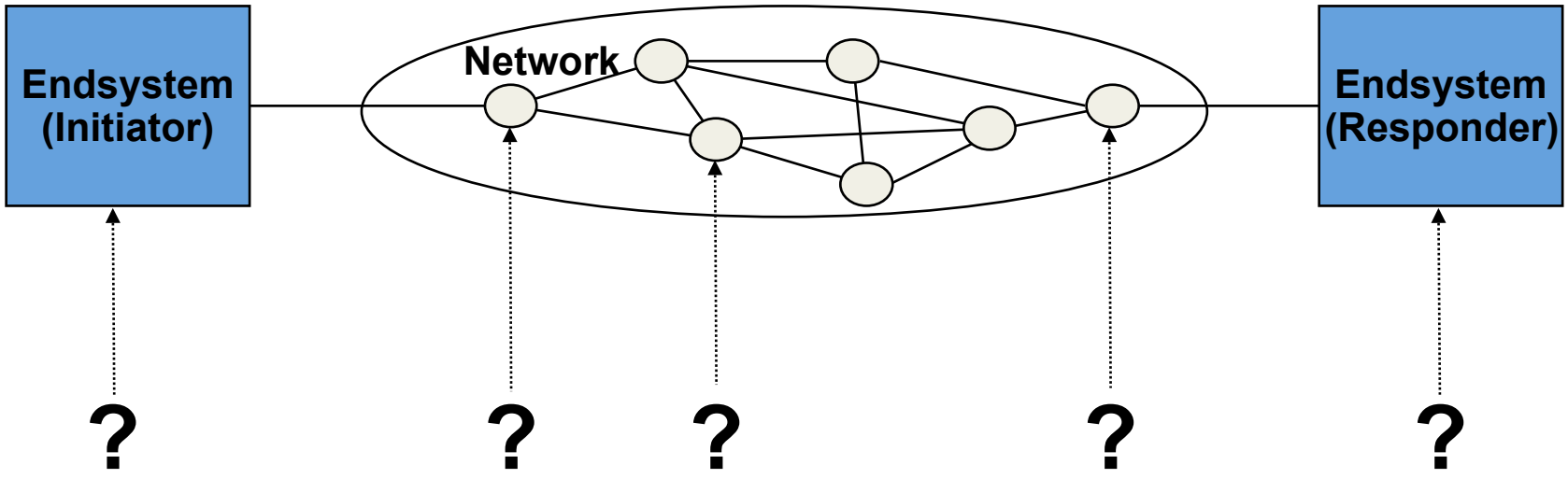
SECURE MOBILE NETWORKING

Mapping ISO Security services to Protocol Layers

Service	L1	L2	L3	L4	L5	L6	L7
Peer Entity Authentication			Y	Y			Y
Data Origin Authentication			Y	Y			Y
Access Control Services			Y	Y			Y
Connection Confidentiality	Y	Y	Y	Y			Y
Connectionless Confidentiality		Y	Y	Y			Y
Selective Field Confidentiality						Y	Y
Traffic Flow Confidentiality	Y		Y				Y
Connectionless Integrity		?	Y	Y			Y
Selective Field Integrity							Y
Non-repudiation, Origin							Y
Non-repudiation, Receipt							Y

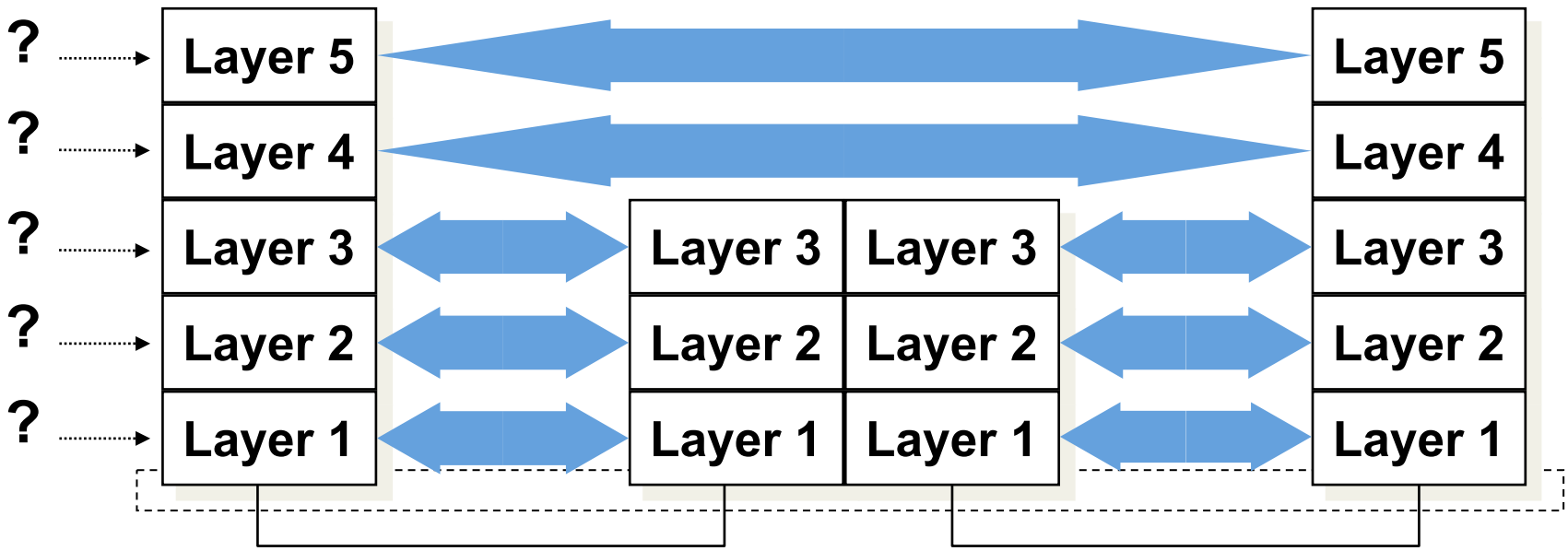
What, How and Where to Secure

Two dimensions on how to integrate security services into communications architectures



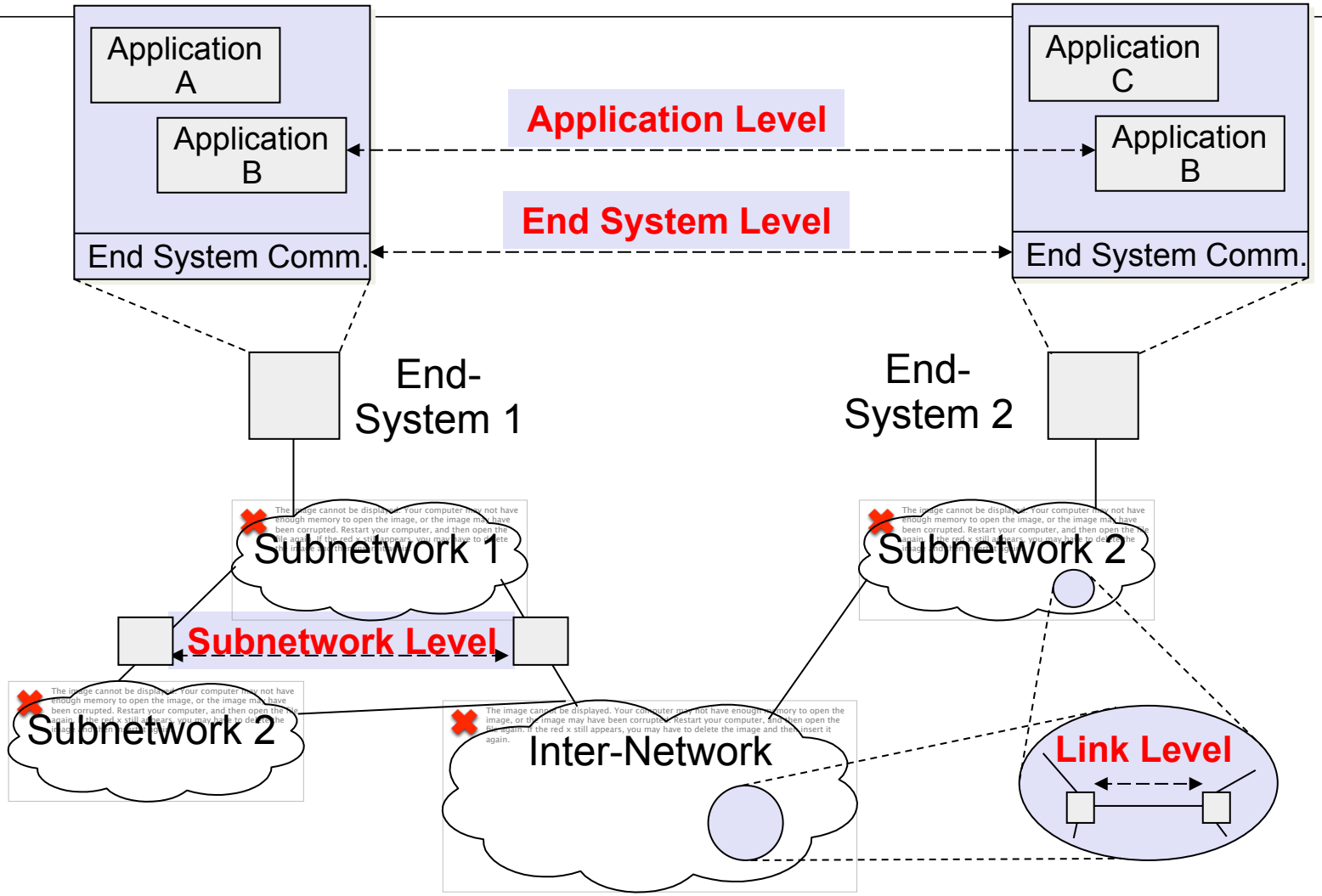
Dimension 1: Which security service should be realized in which node?

What, How and Where to Secure



Dimension 2: Which security service should be realized in which layer?

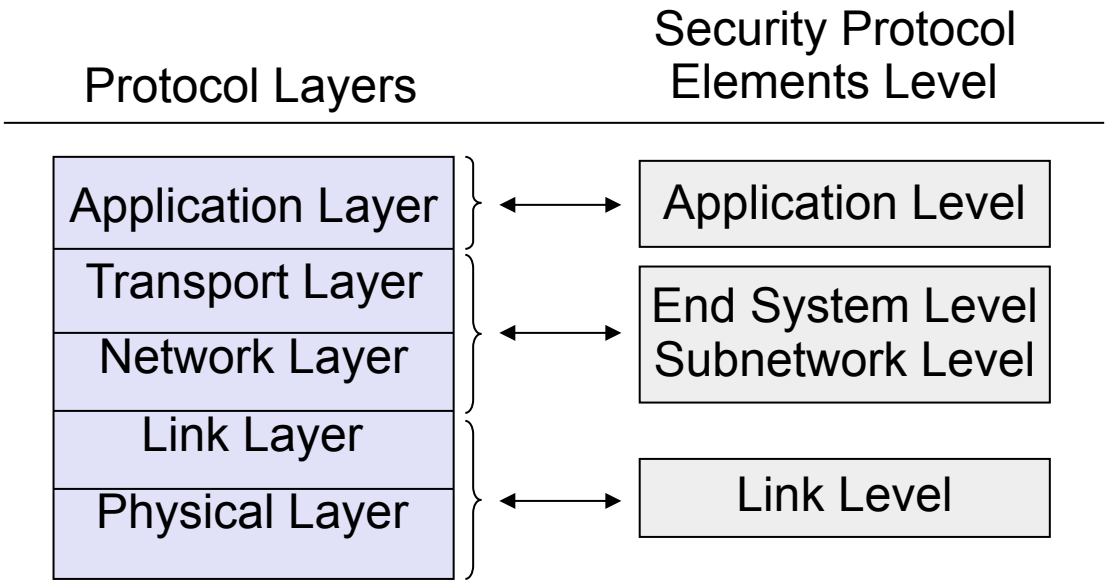
A Pragmatic Model for Network Security



Relationships Between Layers & Requirements Levels

Typically, relations between protocol layers and the protocol element security requirements levels are not one-to-one:

- Security mechanisms for end system and subnetwork level reqs. can be realized in the transport and/or the network layer
- Link level reqs. can be met by integrating security mechanisms or using “special functions” of the either the link layer and/or the physical layer



SECURE MOBILE NETWORKING

Trade-offs to place Security on APP Level

Considerations Regarding Specific Levels

Application level:

- This level might be the only appropriate level, for example because:
  - A security service is application specific, e.g. access control for a networked file store
  - A security service needs to traverse application gateways, e.g. integrity and / or confidentiality of electronic mail
  - Semantics of data is important, e.g. for non-repudiation services
  - It is beyond the reach of a user / application programmer to integrate security at a lower level

Considerations Regarding Specific Levels

End system level:

- This level is appropriate when end systems are assumed to be trusted and the communication network is assumed to be untrusted
- Further advantages of end system level security:
  - Security services are transparent to applications
  - The management of security services can be more easily given in the hands of one system administrator

Integration into Lower Protocol Layers vs. Applications

Benefits of integrating security services into lower network layers:

- Security:
  - The network itself also needs to be protected
  - Security mechanisms realised in the network elements (esp. in hardware) are often harder to attack for network users
- Application Independence:
  - Basic network security services need not be integrated into every single application
- Quality of Service (QoS):
  - QoS preserving scheduling of the communication subsystem can also schedule encryption of co-existing data streams
  - Example: simultaneous voice call and FTP transfer
- Efficiency:
  - Hardware support for computationally intensive encryption / decryption can be easier integrated into protocol processing

Integration into End Systems vs. Intermediate Systems

Integration into end systems:

- Can be done generally either on the application or end system level
- In some special cases also a link level protection might be appropriate, e.g. when using a modem to connect to a dedicated device

Integration into intermediate systems

- Can be done on all four levels:
  - Application / “end system” level: for securing management interfaces of intermediate nodes, not for securing user data traffic
  - Subnetwork / link level: for securing user data traffic

Depending on the security objectives an integration in both end systems and intermediate systems might be appropriate

Application-level Security Trade-offs

SECURE MOBILE NETWORKING

Developer's perspective

Developers' Perspective

Pros/cons

- Everything is under control of the application developer
- We have to modify every single application
- Application designers are not necessarily security experts

The general approach

- Use a security software package
- Use provided functions to
  - perform key exchange; encrypt/decrypt messages
- Link application software with the security library

Different levels of abstraction are possible

- `s = new SecureSocket(), s.send(m), ...`
- `s = new Socket(), c=new Security(DES, K), m=c.encrypt(m), s.send(m), ...`

Examples: The Java security APIs, Crypto++ Library, ...

Developers' Perspective

Developers' perspective: Summary

- A standard security package provides normally basic cryptographic functions
- However it is still required to define and implement a communication protocol
- Unless, the library abstracts from the communication aspects as well.
  - In this case the library implements an existing transport layer security protocol (e.g. SSL/TLS)

Remaining protocol elements to design

- Key exchange?
- Key management (public key, session key, ...)
- Time stamps, sequence numbers, ...

Summary of c03m01

Integration of security services into communications architectures is guided by two main questions:

- Which security service into which node?
- Which security service into which layer?

These design choices can also be guided by looking at a pragmatic model of networked computing which distinguishes four different levels on which security services may be realized:

- Application / end system / subnetwork / link and physical level

As there are various reasons for and against each option, there is no single solution to this design problem

The developers might need to get security aware

In this course we will, therefore, study some examples of security services integration into network architectures in order to better understand the implications of the design choices made

Acks & Recommended Reading

Selected slides of this chapter courtesy of

- Günther Schäfer (TU Ilmenau) and Jens Schmitt (TU Kaiserslautern)
- Utz Roedig (ULancaster)

Recommended reading

- [KaPeSp2002] Charlie Kaufman, Radia Perlman, Mike Speciner: Network Security – Private Communication in a Public World, 2nd Edition, Prentice Hall, 2002, ISBN: 978-0-13-046019-6
- [Stallings2014] William Stallings, Network Security Essentials, 4th Edition, Prentice Hall, 2014, ISBN: 978-0-136-10805-4
- [Schäfer2003] G. Schäfer. Netzsicherheit - Algorithmische Grundlagen und Protokolle. dpunkt.verlag, 2003.

Copyright Notice

This document has been distributed by the contributing authors as a means to ensure timely dissemination of scholarly and technical work on a non-commercial basis. Copyright and all rights therein are maintained by the authors or by other copyright holders, notwithstanding that they have offered their works here electronically.

It is understood that all persons copying this information will adhere to the terms and constraints invoked by each author's copyright. These works may not be reposted without the explicit permission of the copyright holder.

Contact

Prof. Dr.-Ing. Matthias Hollick
Department of Computer Science

SEEMOO
Mornewegstr. 32
64293 Darmstadt/Germany
matthias.hollick@seemoo.tu-darmstadt.de

Phone +49 6151 16-70920
Fax +49 6151 16-70921
www.seemoo.tu-darmstadt.de