Computer and Network Security

Dr. Chan Yeob Yeun

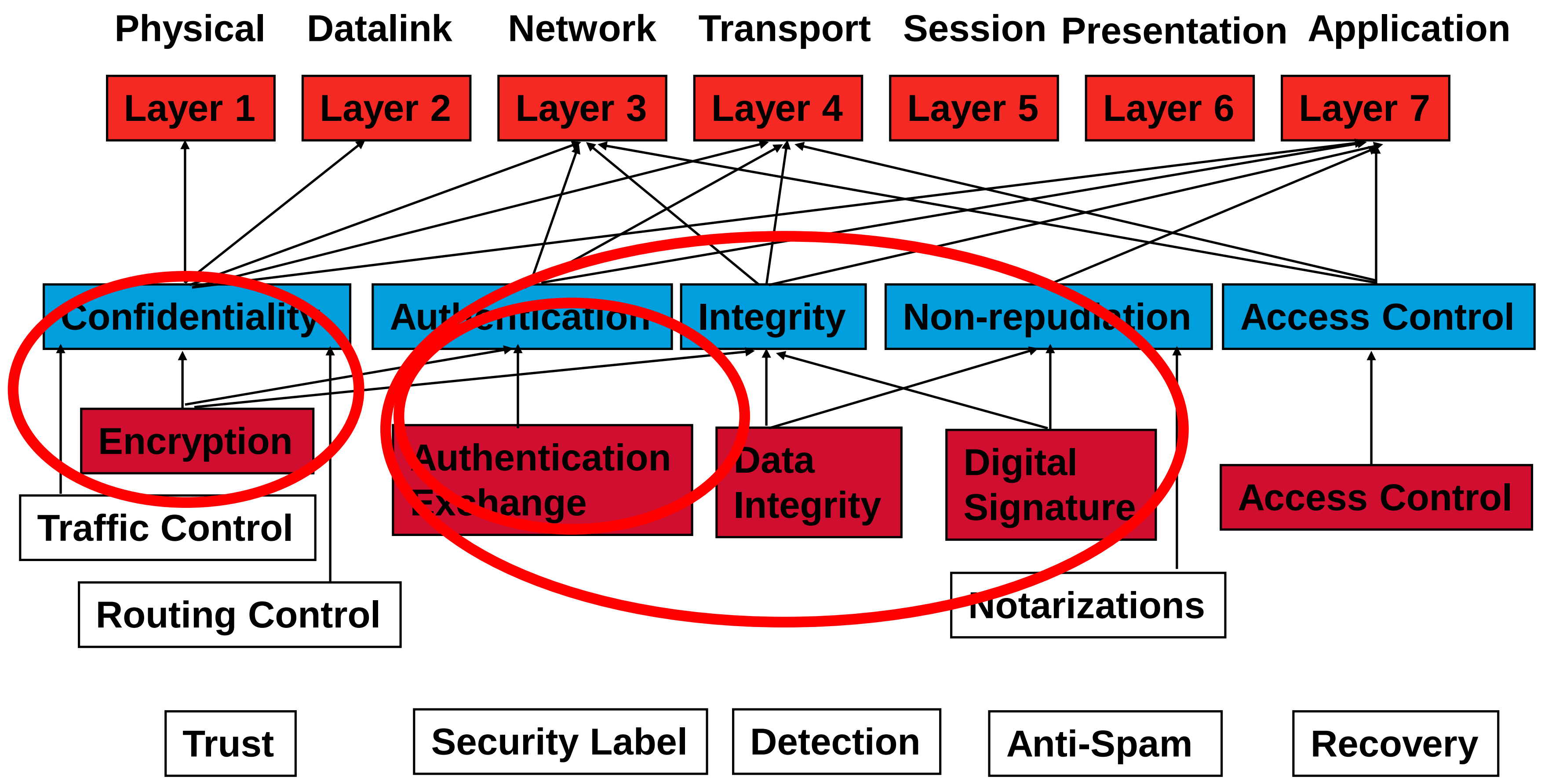
Week 8 - 9

جامعة خليفة
Khalifa University

Weekly Lecture Plan

Wk	Contents	Cmt	Wk	Contents	Cmt
1	**Introduction**		9	**Foundations of Network Security II**	
2	**Foundations of Computer Security**	**Tutorial** **Assig Plan**	10	Network-Based Threats and Attacks	
3	**Identification and Authentication I**		11	Network Security Protocols I	
4	**Identification and Authentication II**	**Quiz 1**	12	Network Security Protocols II	Quiz 3
5	**Access Control**		13	Firewalls	
6	**Modern Computer Attacks**		14	IDS / IPS	Assig Submit
7	**Malicious Code**	**Assig Confirm**	15	Revision and Presentation	
8	**Foundations of Network Security I**	**Quiz 2**	16	Exam	

What is Computer and Network Security ?



Network Model

- OSI Model
  - 7 layers
  - Old
  - Applications often have properties of several layers at once
    - Makes classification difficult, confusing
- TCP/IP Model
  - "DoD" model (Department of Defense)
  - 5 layers

OSI 7-Layer Model

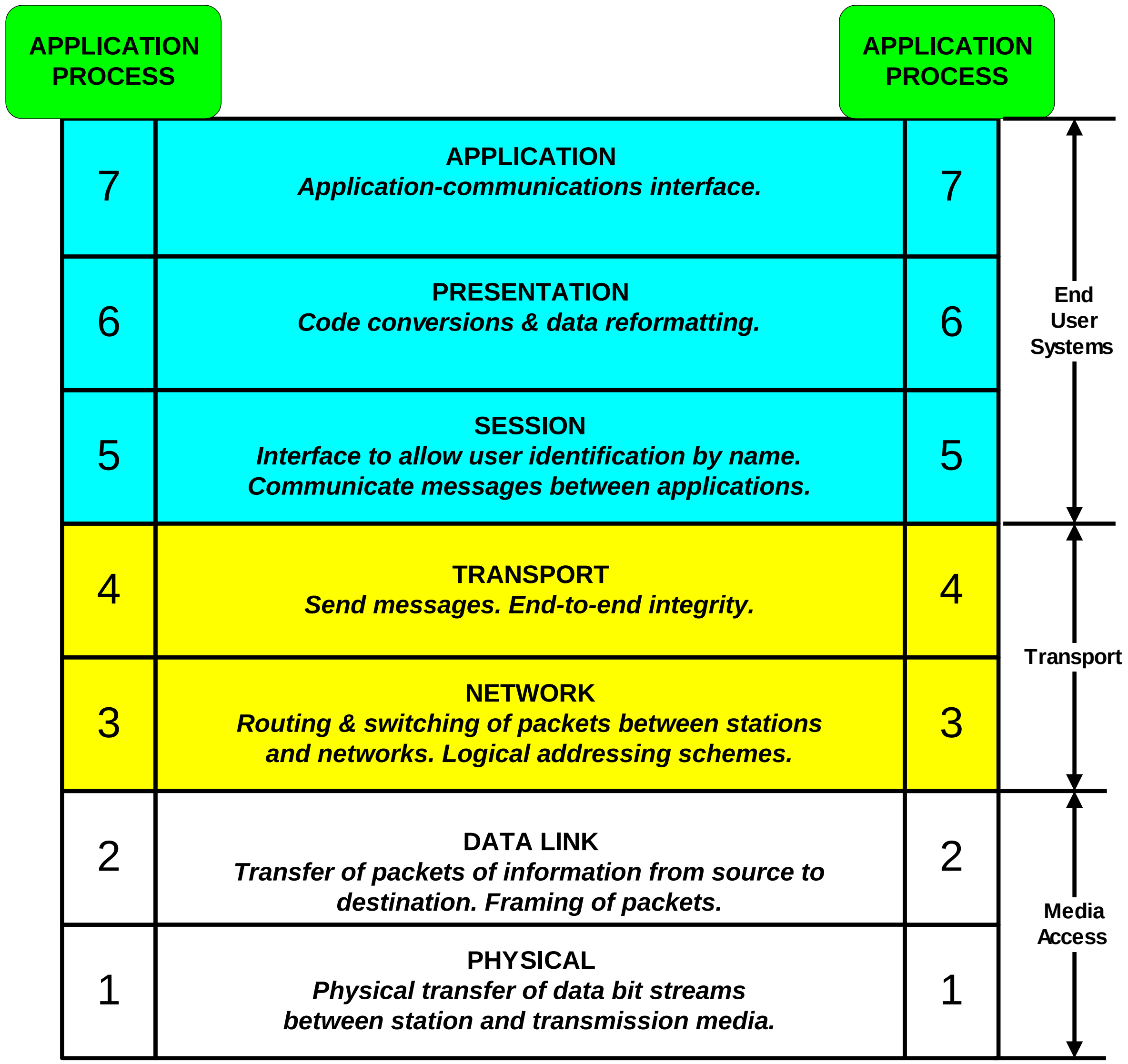
OSI: Open Systems Interconnection

- ISO standard

Layered approach provides:

- Simplification
- Abstraction
  - Each layer talks only to the equivalent layer somewhere else
- Division of responsibilities
- Standardization and interchangeability of equipment from different makers

OSI 7-Layer



APPLICATION PROCESS		APPLICATION PROCESS	
7	**APPLICATION** *Application-communications interface.*	7	End User Systems
6	**PRESENTATION** *Code conversions & data reformatting.*	6	
5	**SESSION** *Interface to allow user identification by name. Communicate messages between applications.*	5	
4	**TRANSPORT** *Send messages. End-to-end integrity.*	4	Transport
3	**NETWORK** *Routing & switching of packets between stations and networks. Logical addressing schemes.*	3	
2	**DATA LINK** *Transfer of packets of information from source to destination. Framing of packets.*	2	Media Access
1	**PHYSICAL** *Physical transfer of data bit streams between station and transmission media.*	1	

OSI vs. TCP/IP

OSI MODEL	TCP/IP
APPLICATION 7	APPLICATION *Worldwide Web: http* *Remote Login: telnet, rlogin* *Remote Commands: rexec, rsh* *File Transfers: ftp, rcp, tftp, UUCP* *E-mail: SMTP, POP3, IMAP* *Remote File Systems: NIS/NFS, RPC*
PRESENTATION 6	
SESSION 5	
TRANSPORT 4	TRANSPORT *Tranmission Control Protocol (TCP)* *User Datagram Protocol (UDP)*
NETWORK 3	INTERNETWORKING Internet Protocol (IP) Internet Control Message Protocol (ICMP)
DATA LINK 2	NETWORK INTERFACE & HARDWARE CONNECTIONS LAN: Ethernet, Token Ring, FDDI, ATM... WAN: SLIP/PPP, X.25, Frame Relay...
PHYSICAL 1	

Security Services and Mechanisms

Security services are Confidentiality, Integrity, Availability, Authentication and Access control.

Security mechanisms should provide the basis of provision of other security services in the majority of network security systems.

ISO 7498-2 distinguishes between *origin authentication* (verifying origin of received data) and *entity authentication* (identity verification)

We consider entity authentication.

Typically achieved by exchange of message called *authentication protocol*

Definitions

Entity authentication:

The corroboration that an entity is the one claimed

Unilateral authentication

Entity authentication which provides one entity with assurance of the other's identity but not vice versa

Mutual authentication

Entity authentication which provides both entities with assurance of each other's identity.

Use of Entity Authentication

Entity authentication only achieved for an instant in time

Typically provided at start of a connection.

If security (e.g. confidentiality, integrity) is required for information passed in a connection, then keys needed for cryptographic protection can be agreed/exchanged during protocol

Security Mechanisms

One security mechanism of entity authentication is "authenticated session key establishment".

Other mechanisms are:

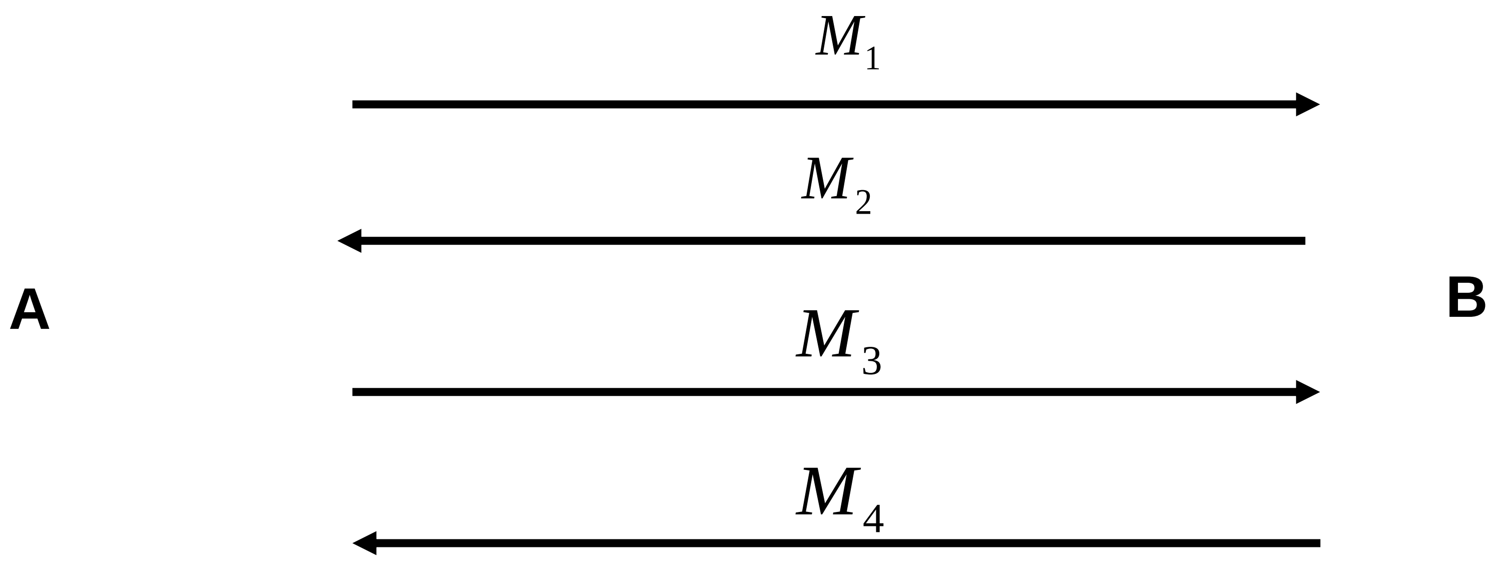
- Secure clock synchronization
- Secure RPC (Remote Procedure Call),
- Secure transactions.

Properties of an authentication protocol

We consider what properties we require an authentication protocol to satisfy

We set up a model of an authentication protocol and consider what we might require of a protocol in general.

Authentication protocol model



A and B wish to use an authentication protocol to provide either unilateral or mutual authentication. Suppose that protocol consists of a finite sequence of message M1, M2…

Assumptions for model

A starts protocol and sends B message M1

B then sends A message M2.

A then sends B message M3, and so on.

We always assume that A or B will not send message Mi until message Mi-1 has been correctly received.

Requirements I

After the protocol, A wants to be sure that”

1. M2, M4, … were all sent by B (as received),
2. M2, M4,… are “fresh”, i.e. not replays of old messages
3. M2, M4,… were intended for A, not for any other entity, and
4. M2, M4,… were only generated by B after M1, M3,… (respectively) were received correctly.

Requirement II

Similarly, B wants to be sure that

1. **M1, M3,…. were all sent by A (as received),**
2. **M1, M3,…, are “fresh”, i.e. not replay of old messages**
3. **M1, M3,…, were intended for B, and not for any other entity, and**
4. **M3, M5,…, were only generated by A after M2, M4,…(respectively) were correctly by A.**

Discussion

In some sense requirement (4) implies both (1) and (2) – except for the special case of B knowing message M1 was sent by A and is fresh (since property (4) has nothing to say about message M1).

A and B may not be sure of “beliefs” until completion of protocol.

Exact list of beliefs needed for a practical protocol may be a subset of these lists.

However, “generic” protocols (c.f. ISO/IEC 9798) should satisfy as many as possible.

Cryptographic Mechanisms

Authentication protocols require use of

- **Secret (or private) keys**
- **Cryptographic mechanisms**

So that message recipient knows:

- **Where it has come from (origin checking)**
- **Not been interfered with (integrity checking)**

Note that cryptography cannot provide freshness checking, i.e. the verification a protocol message is not simply a replay of a previously transmitted (valid) protocol message, protected using a currently valid key (consider this later)

Three approach

Three main possibilities for type of cryptographic mechanism to protect message of a protocol:

- **Encipherment,**
- **Integrity mechanism (MAC or Cryptographic Check Function),**
- **Digital signature**

Let’s consider each in turn more details.

Encipherment I

To protect a message in a protocol, the sender enciphers it with a secret key shared with the recipient.

Recipient can verify origin because, when deciphered it “make sense” (and only genuine sender knows key).

We assume that an interceptor cannot manipulate an enciphered message (without knowledge of the key used to encipher it) in such a way that it still “make sense” after decipherment.

This constrains the type of encipherment algorithm that is suitable for use in this application.

Encipherment II

Usual solution – add redundancy according to specified formula to message prior to encipherment.

Presence of redundancy can be automatically checked by recipient.

One popular approach is to append a Manipulation Detection Code (MDC) (e.g. a sort of checksum dependent on the entire message) to message before encipherment.

Integrity mechanisms I

Various types of data integrity mechanism:

- **Message Authentication Code (MAC) is a function which takes as inputs a secret key and a message, and gives as output a fixed length MAC that is appended to the message as a “cryptographic check value”.**

- **One-way function. Message is appended to secret key and input to one-way function. Output is appended to message as check value**

- **One-way function is a function which is simple to compute yet computationally infeasible to invert.**

Integrity mechanisms II

Typically, a MAC is constructed using a block cipher (e.g. DES) in Cipher Block Chaining (CBC) mode.

Recipient can always re-compute the check value and hence verify the origin and integrity of a message. The appended check value provide both:

- **Origin checking, because the originator and verifier are assumed to be the only possessors of the secret key, and hence are the only ones capable of computing a valid check value for a message Integrity checking, because, without the secret key, the new check value corresponding to a changed message cannot be calculated.**

Digital signatures

Pair of keys:

- **Private “signing key” (defines signature transformation), and**
- **Public “verification key” (defines verification transformation).**

Signature typically like a MAC, i.e. it is appended to a message

Recipient uses public key to verify message, thereby enabling the recipient to check the origin and integrity of a message

Classification by cryptography

Can classify authentication protocols by cryptographic mechanism they use

ISO/IEC 9798:

- Part 1 (1991): General model
- Part2 (1994): Using symmetric enchiperment
- Part3 (1993): Using digital signature
- Part4 (1995): Using data integrity mechanism

Freshness mechanisms

Origin and integrity checking not enough also need means of checking “freshness” of message, to protect against replays

Two main methods

- Use of time-stamp (clock-based or logical time-stamps)
- Use of “nonce” or challenges (as in challenge-response protocol)

Clock-based protocol I

Inclusion of a date/time stamp in a message enables recipient to check it for freshness (as long as time-stamp protected by cryptographic means).

Requires securely synchronised clocks

Non-trivial to provide such clock (Note that clock drift of work-station typically 1 second/day)

Clock-base protocol II

Receiver will need to define time acceptance “window” either side of receiver’s current clock value because of

- Clock variations, hence no two clocks will be precisely synchronised, except perhaps at some instant in time.
- Messages take time to propagate from one machine to another, and this time will vary unpredictably.

Acceptance window allows for undetectable replays – hence need to store a log of recently received messages.

Synchronised clocks

How do you provide syncrhonised clocks?

One solution – at regular intervals use an authentication protocol not based on time-stamp (e.g. nonce-based) at regular intervals to distribute a master clock value which is then used to update each entity’s individual clock.

Another solution – have reliable access to accurate time source (e.g. national radio broadcast time)

Logical time-stamp I

Alternative to clocks – every pair of communicating entities store a pair of sequence numbers used only in communications between that pair

E.g. for communications between A and B, A must maintain two counter: N_{AB} and N_{BA}

Every time A sends B a message, value of N_{AB} is included, and N_{AB} is incremented

Logical time-stamp II

Every time A receives a message from B, then the sequence number put in the message by B (N say) is compared with N_{BA} (as stored by A):

If $N > N_{AB}$ then

- **The message is accepted as fresh and**
- **N_{BA} is reset to equal N,**

If $N \leq N_{BA}$ then

- **The message is rejected as an “old” message**

These sequence numbers take the role of what are known as logical time-stamps

All time stamp protocols have problems in providing property (4) from the list of desired properties for an authentication protocol

Nonce-based protocol I

Nonce-based protocols use a quite different mechanism to provide freshness checking

A sends B a nonce (Number used ONCE) as a challenge

B includes the nonce in the response to A

Because the nonce has never been used before, A can verify the freshness of B’s response (given message integrity is protected by cryptography).

Nonce-based protocols II

Up to A (who chooses nonce) to make sure it is “new”

Main property is “onetime” property, and thus, in theory, could be provided using a counter

However, many protocols (to operate correctly) also need nonces to be unpredictable to any third party

There are problems with the provision of properties (2) and (4) with some nonce-based protocols if nonces can be predicted by third parties

Nonce are typically chosen at random form a set sufficiently large to mean that the probability of the same nonce being used twice is effectively zero

Summary

Three main possibilities for type of cryptographic mechanism to protect message of a protocol:

- **Encipherment,**
- **Integrity mechanism (MAC or Cryptographic Check Function),**
- **Digital signature**

Cryptography cannot provide freshness checking

Origin and integrity checking not enough also need means of checking “freshness” of message, to protect against replays

- **Time-Stamp**
- **Sequence Number (Logical Time-Stamp)**
- **Nonce**