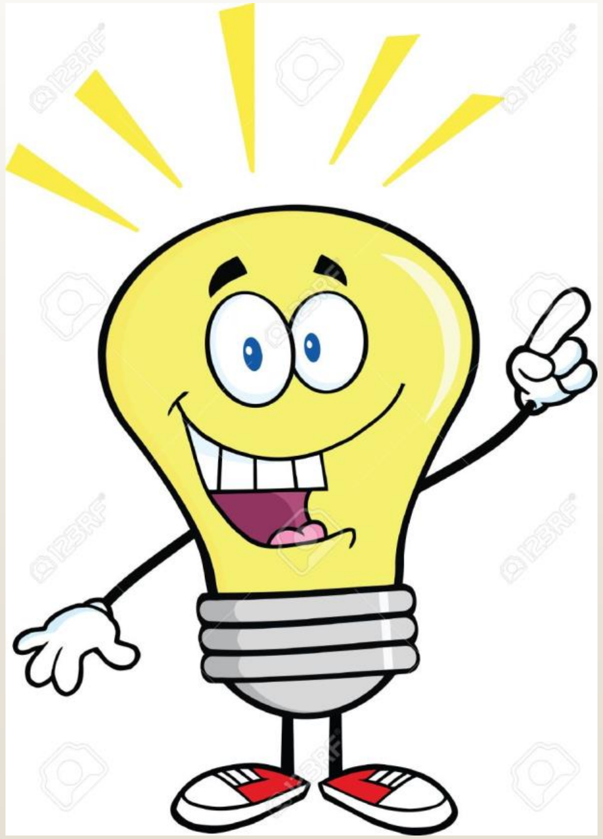
Network Security Foundations

Agenda

1. The OSI Model and TCP/IP Model
   - Why do we have various devices and they can still communicate with each other?
   - Application Layer
   - Transport Layer
   - Internet Layer
   - Network Access Layer
   - Data Encapsulation: Putting it all together
2. Cryptograhy Foundations
   - Why do we need cryptography?
   - Types of Cryptosystems
   - Symmetric Key Encryption
   - Asymmetric Key Encryption
3. HTTPS, SSL and TLS – Why and how do they work?
4. Computer Network Communications Example
   - How does it all work in real life?

The OSI Model and TCP/IP Stack

- BRAINSTORM: Why do we have various devices and they can still communicate with each other?



ANSWER:
They all implement the same networking model.

- <u>What is a networking model?</u>
- It's a comprehensive set of documents, where individually, each document describes one small function required for a network.
- Think of a networking model just like you think of an architectural pattern for building a house.

- <u>Do I really need it?</u>
- Of course, you can build a house without the blueprint, but it can ensure the house has the right foundation and structure so it won't fall down.
- The equivalent situation applies to computer networks.

Basic Concepts

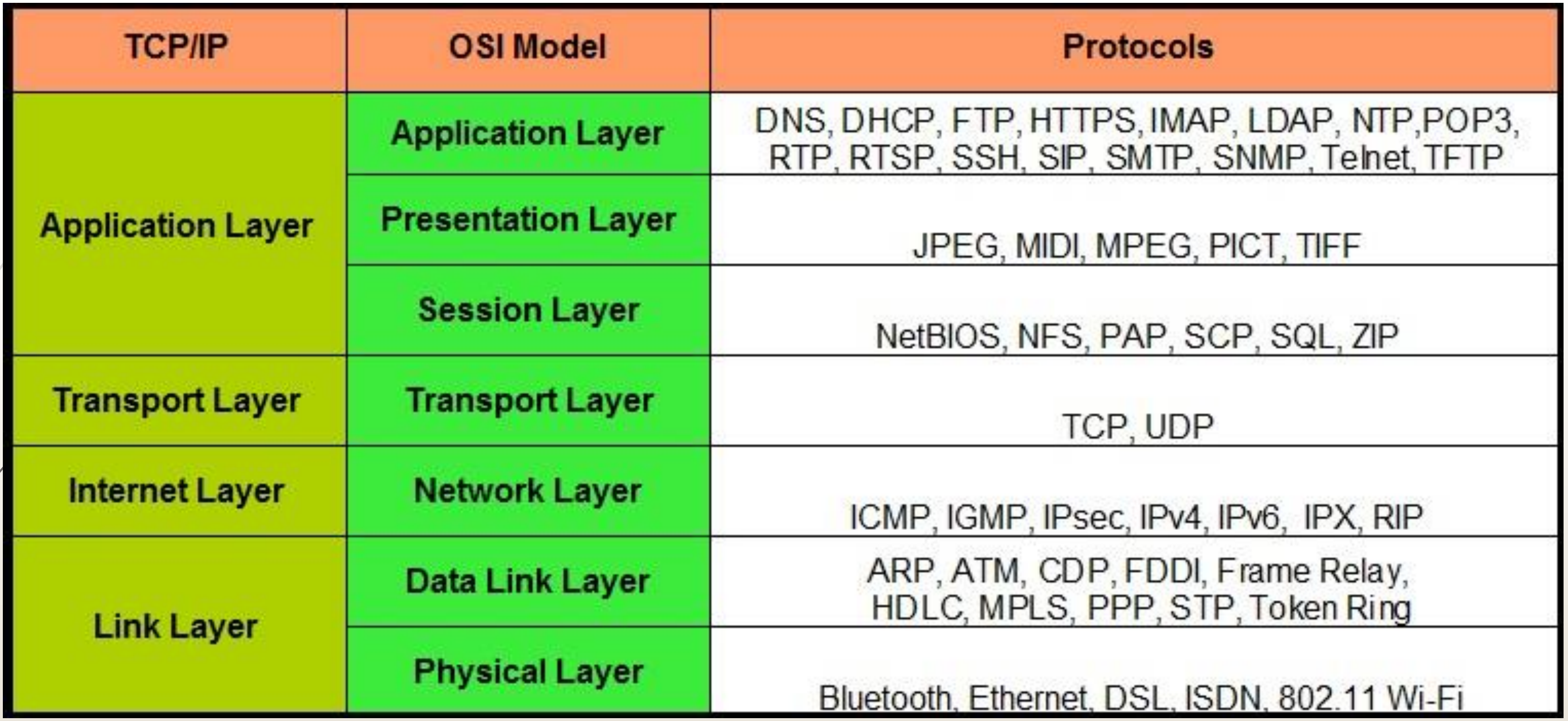
- Protocol: A set of logical rules that devices must follow to communicate.
- Layer: Each model breaks the functions into a smaller number of categories, called layers. Each layer includes protocols and standards that relate to that category of functions.
- Host: Any device that has an IP address and connects to any TCP/IP network.

<u>Two well-known networking models are:</u>

✓ The OSI Model
✓ TCP/IP Model

The OSI Model & TCP/IP Model Comparison

OSI Model	**TCP/IP Model**
Both define and reference a large collection of protocols that allow computers to communicate.	
	Protocols are defined in documents called Requests for Comments (RFC)
7 Layers	4 or 5 Layers
The top layers focus more on the applications that need to send and receive data.	
The lower layers focus on the need to somehow transmit the bits from one device to another.	

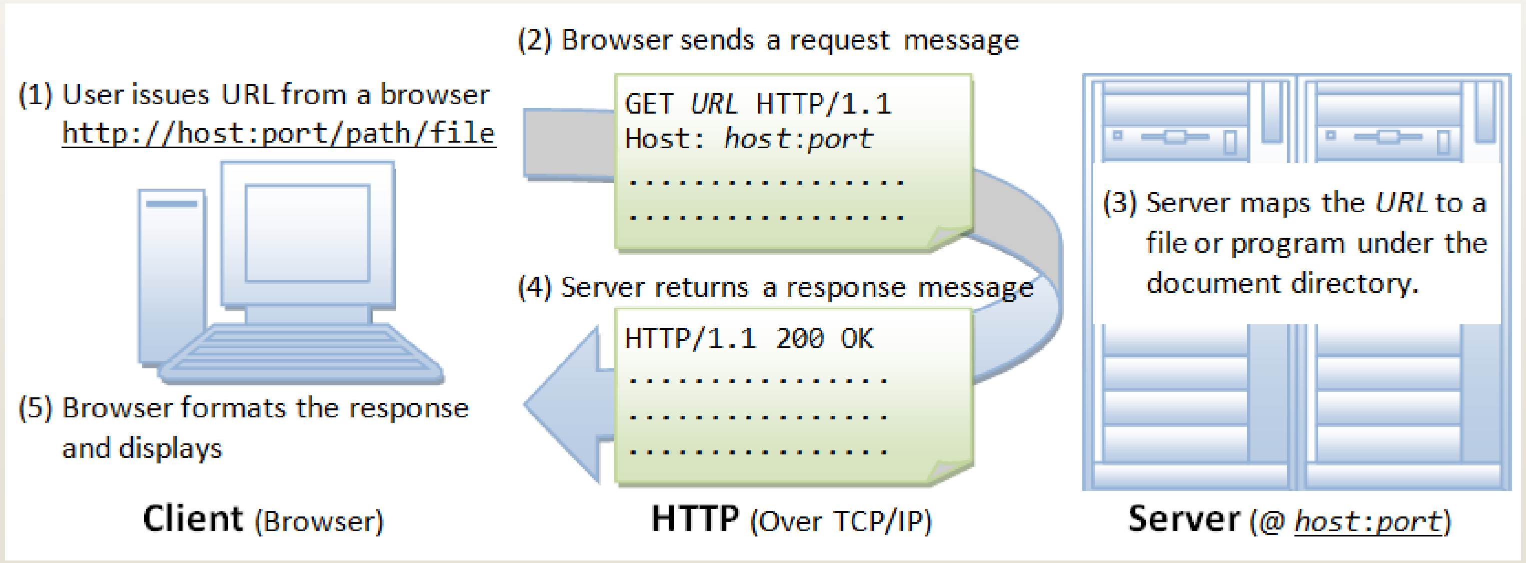


TCP/IP	OSI Model	Protocols
Application Layer	Application Layer	DNS, DHCP, FTP, HTTPS, IMAP, LDAP, NTP,POP3, RTP, RTSP, SSH, SIP, SMTP, SNMP, Telnet, TFTP
	Presentation Layer	JPEG, MIDI, MPEG, PICT, TIFF
	Session Layer	NetBIOS, NFS, PAP, SCP, SQL, ZIP
Transport Layer	Transport Layer	TCP, UDP
Internet Layer	Network Layer	ICMP, IGMP, IPsec, IPv4, IPv6, IPX, RIP
Link Layer	Data Link Layer	ARP, ATM, CDP, FDDI, Frame Relay, HDLC, MPLS, PPP, STP, Token Ring
	Physical Layer	Bluetooth, Ethernet, DSL, ISDN, 802.11 Wi-Fi

Application Layer

- Defines protocols providing services to the application software (processes) running on a computer, an interface between software running on a computer and the network itself.

  Example:
  HTTP defines how web browsers can pull the contents of a web page from a web server.


(1) User issues URL from a browser
http://host:port/path/file
(2) Browser sends a request message
(3) Server maps the URL to a file or program under the document directory.
(4) Server returns a response message
(5) Browser formats the response and displays
Client (Browser)
HTTP (Over TCP/IP)
Server (@ host:port)

```
GET URL HTTP/1.1
Host: host:port
................
................
```

```
HTTP/1.1 200 OK
................
................
................
```

INTERNAL

Transport Layer

- Provides host-to-host communication services for the application layer (applications) such as:
  - Connection-oriented data stream support
  - Reliability
  - Flow control
  - Multiplexing

Most common protocols:

- TCP (Transmission Control Protocol)
- UDP (User Datagram Protocol)

Many application layer protocols including HTTP, HTTPS, FTP and so on require a way to guarantee delivery of data across a network.

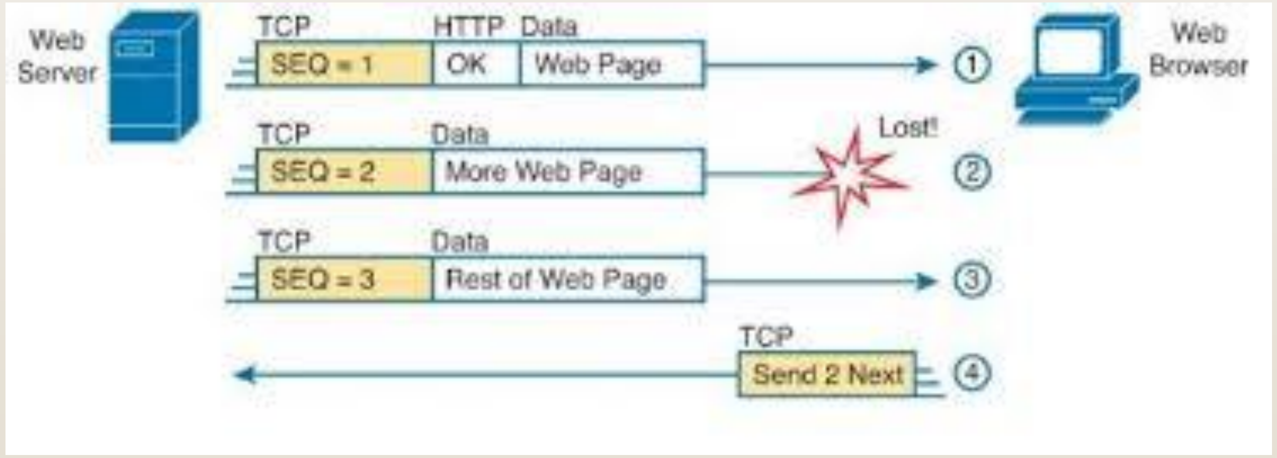
- How is it done that application layer protocols can transmit data through a reliable communication channel?

ANSWER:

- Adjacent-layer interaction: Each layer provides a service to the layer above it.

In case of TCP, it provides the following mechanisms:

- Reliability: Packets may be lost during transport due to errors, network congestion. TCP can verify that by the acknoledgment mechanism.
- Same order delivery: The network/internet layer doesn't generally guarantee packets of data will arrive in the same order that they were sent.



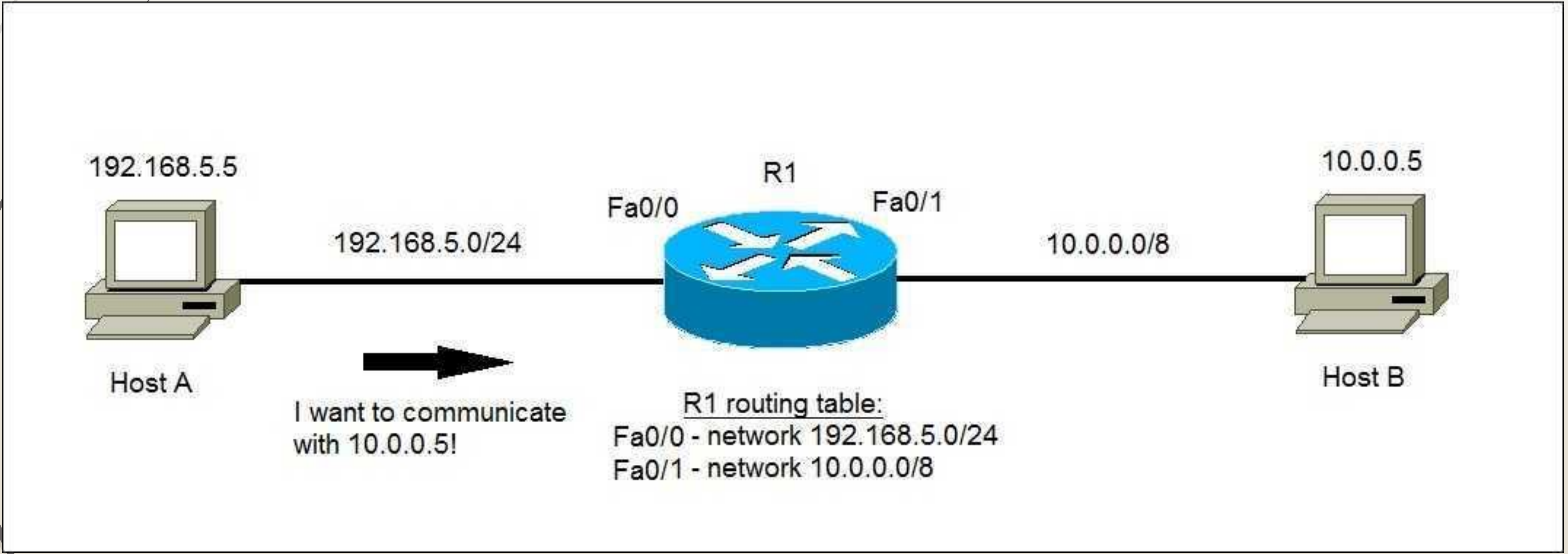
UDP

- UDP: User Datagram Protocol.
- What is it used for?
  - With UDP, processes can send messages (referred to as datagrams), to other hosts on an IP network.
- Details
  - Sends individual packets of data called datagrams.
  - Uses a connectionless communication model
  - Provides:
    - Checksums for data integrity
    - Port numbers for addressing different functions at the source and destination of the datagram.
  - No handshake dialogs
  - No guarantee of delivery, ordering, or duplicate protection.
  - Suitable for purposes where error checking and correction are either not necessary.
  - Avoids the overhead of such processing in the protocol stack.

Internet Layer

- Defines protocols and specifications used to transport packets/datagrams from the originating host to the destination host.
- The major protocol – IP, provides the most important features:
  - Addressing
  - Routing
- The Internet Layer works just like the postal service – to deliver messages to the correct destinations.

- Addressing: IP defines addresses so that each host computer can have a different IP address, just like the postal service defines addressing that allows unique addresses for house etc.
- Routing: IP defines the process of routing so that devices called routers can forward the packets of data so that they are delivered to the correct destinations.
- In routing, IP packets are forwarded from one device to another. Any device with an IP address can connect to the TCP/IP network and send packets.
- An IP address is basically used to uniquely identify a network interface (device) in a computer network.


192.168.5.5
Host A
192.168.5.0/24
I want to communicate with 10.0.0.5!
R1
Fa0/0
Fa0/1
R1 routing table:
Fa0/0 - network 192.168.5.0/24
Fa0/1 - network 10.0.0.0/8
10.0.0.0/8
10.0.0.5
Host B

TCP/IP Network Access/Link Layer

- Defines the protocols and hardware required to deliver data across some physical network.
- Defines physical media over which data can be transmitted.
- The network access layer provides services to the layer above in the model.
- When a host or router's IP process chooses to send an IP packet to another router or host, that host or router then uses network access layer details to send the packet to the next host/router.

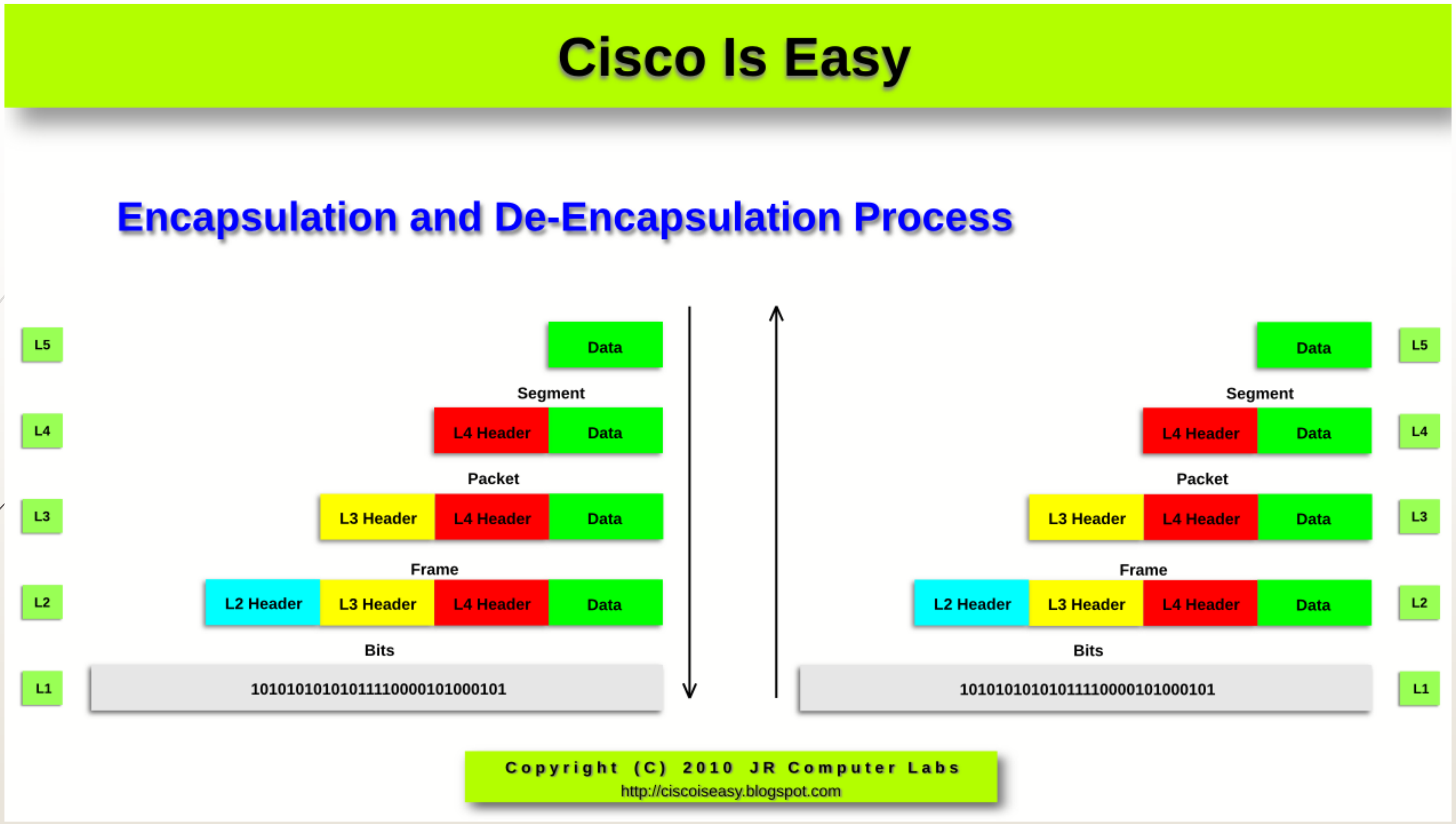
- Examples:
  - Ethernet (IEEE 802.3)
  - PPP (Point-to-point Protocol)
  - FDDI
  - IEEE 802.11 (WiFi)
  - Bluetooth

Data Encapsulation

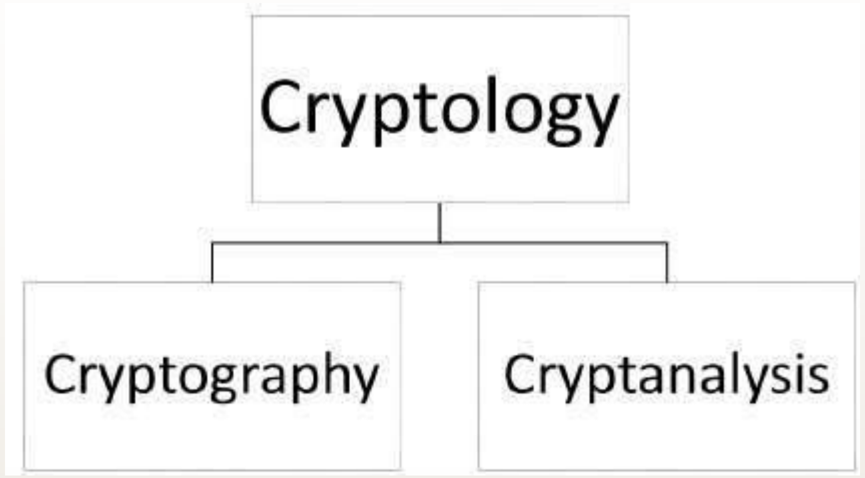
- Encapsulation: The process of putting headers (and sometimes trailers) around some data.
- Protocols define headers and trailers for the same general reason, but headers exist at the beginning of the message and trailers exist at the end.

- Steps

1. Create and encapsulate the application data with any required application header e.g. An HTTP request with some headers and contents of a web page.
2. Encapsulate the data supplied by the application header inside a transport layer header e.g. A TCP header (SEGMENT)
3. Encapsulate the data supplied by the transport inside an IP header (PACKET)
4. Encapsulate the data supplied by the Internet layer inside a data link layer header and trailer e.g. Ethernet, WiFi (IEEE 802.11)
5. Transmit the bits. The physical layer encodes a signal onto the medium to transmit the frame.


Cisco Is Easy
Encapsulation and De-Encapsulation Process
L5
Data
L4
Segment
L4 Header
Data
L3
Packet
L3 Header
L4 Header
Data
L2
Frame
L2 Header
L3 Header
L4 Header
Data
L1
Bits
10101010101011110000101000101
Data
L5
Segment
L4 Header
Data
L4
Packet
L3 Header
L4 Header
Data
L3
Frame
L2 Header
L3 Header
L4 Header
Data
L2
Bits
10101010101011110000101000101
L1
Copyright (C) 2010 JR Computer Labs
http://ciscoiseasy.blogspot.com

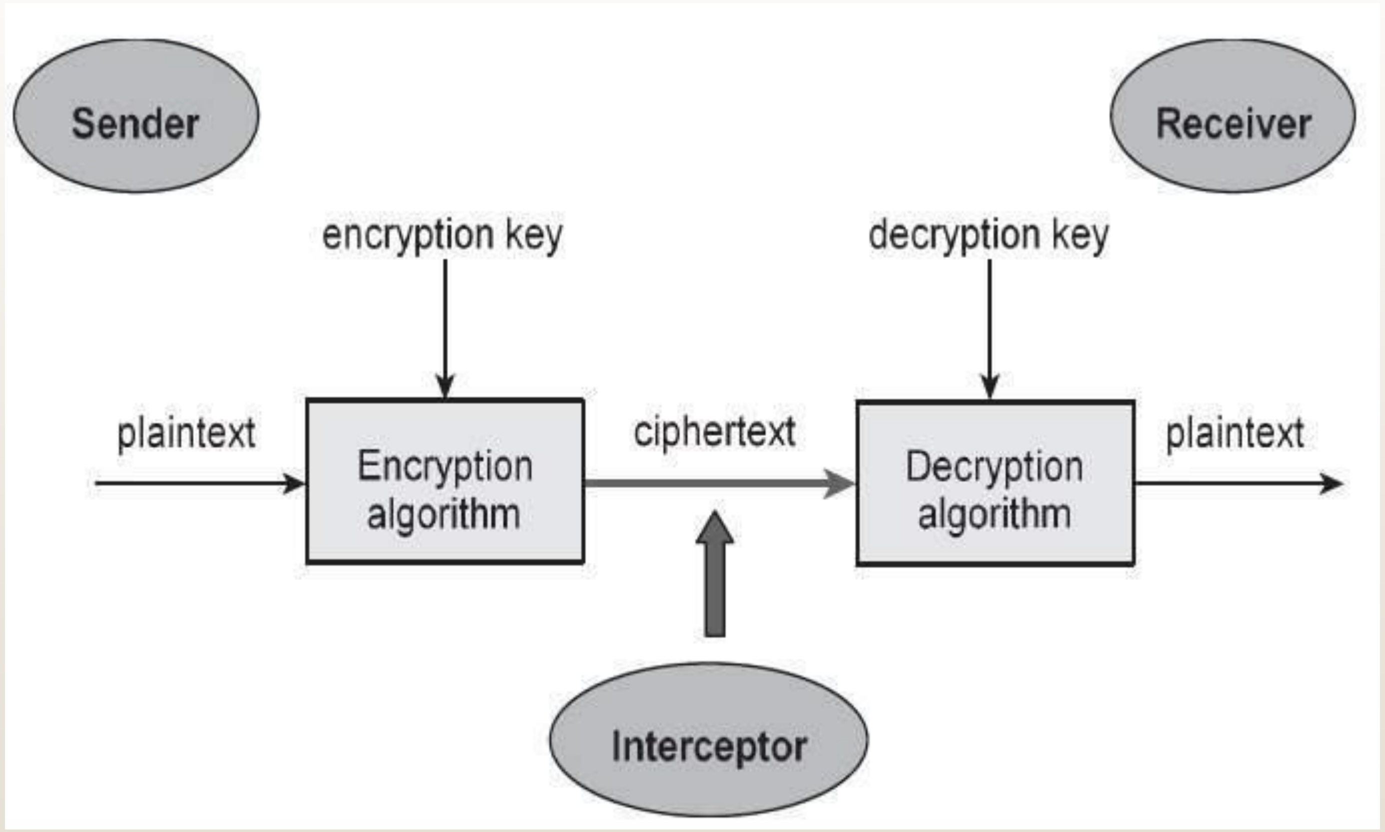
Cryptograhpy Foundations



- Cryptograhpy: The art and science of making a cryptosystem that's capable of providing information security.
- It deals with the actual securing of digital data, referrring to the design of mechanisms based on math algorithms providing information security services
- Cryptoanalysis: The art and science of breaking the cipher text.
- The cryptograhic process results in the cipher text for transmission or storage.

Basic Concepts

- Cryptosystem: An implementation of cryptographic techniques and their infrastructure to provide information security services.



Components of a cryptosystem

- **Plaintext.** It is the data to be protected during transmission.
- **Encryption Algorithm.** It is a mathematical process that produces a cipher text for any given plaintext and encryption key. It is a cryptographic algorithm that takes plaintext and an encryption key as input and produces a cipher text.
- **Cipher text.** It is the scrambled version of the plaintext produced by the encryption algorithm using a specific encryption key. The cipher text is not guarded. It flows on public channel. It can be intercepted or compromised by anyone who has access to the communication channel.
- **Decryption Algorithm,** It is a mathematical process, that produces a unique plaintext for any given cipher text and decryption key. It is a cryptographic algorithm that takes a cipher text and a decryption key as input, and outputs a plaintext. The decryption algorithm essentially reverses the encryption algorithm and is thus closely related to it.

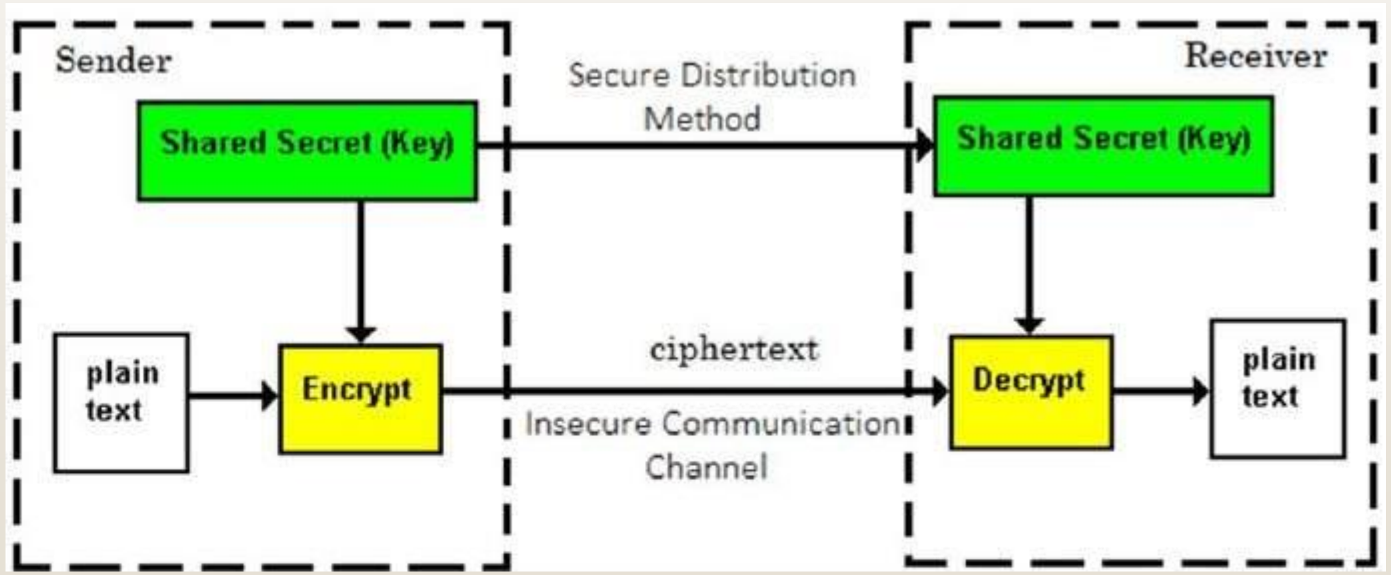
- **Encryption Key.** It is a value that is known to the sender. The sender inputs the encryption key into the encryption algorithm along with the plaintext in order to compute the cipher text.
- **Decryption Key.** It is a value that is known to the receiver. The decryption key is related to the encryption key, but is not always identical to it. The receiver inputs the decryption key into the decryption algorithm along with the cipher text in order to compute the plaintext.
- For a given cryptosystem, a collection of all possible decryption keys is called a **key space**.
- An **interceptor** (an attacker) is an unauthorized entity who attempts to determine the plaintext. He can see the cipher text and may know the decryption algorithm. He, however, must never know the decryption key.

Types of Cryptosystems

- Symmetric Key Encryption
- Asymmetric Key Encryption

Symmetric Key Encryption

- The encryption process where **same keys are used for encrypting and decrypting** the information.
- Symmetric cryptography -> The study of symmetric cryptosystems.
- Examples:
  - DES (Digital Encryption Standard)
  - Triple-DES (3-DES)
  - Blowfish

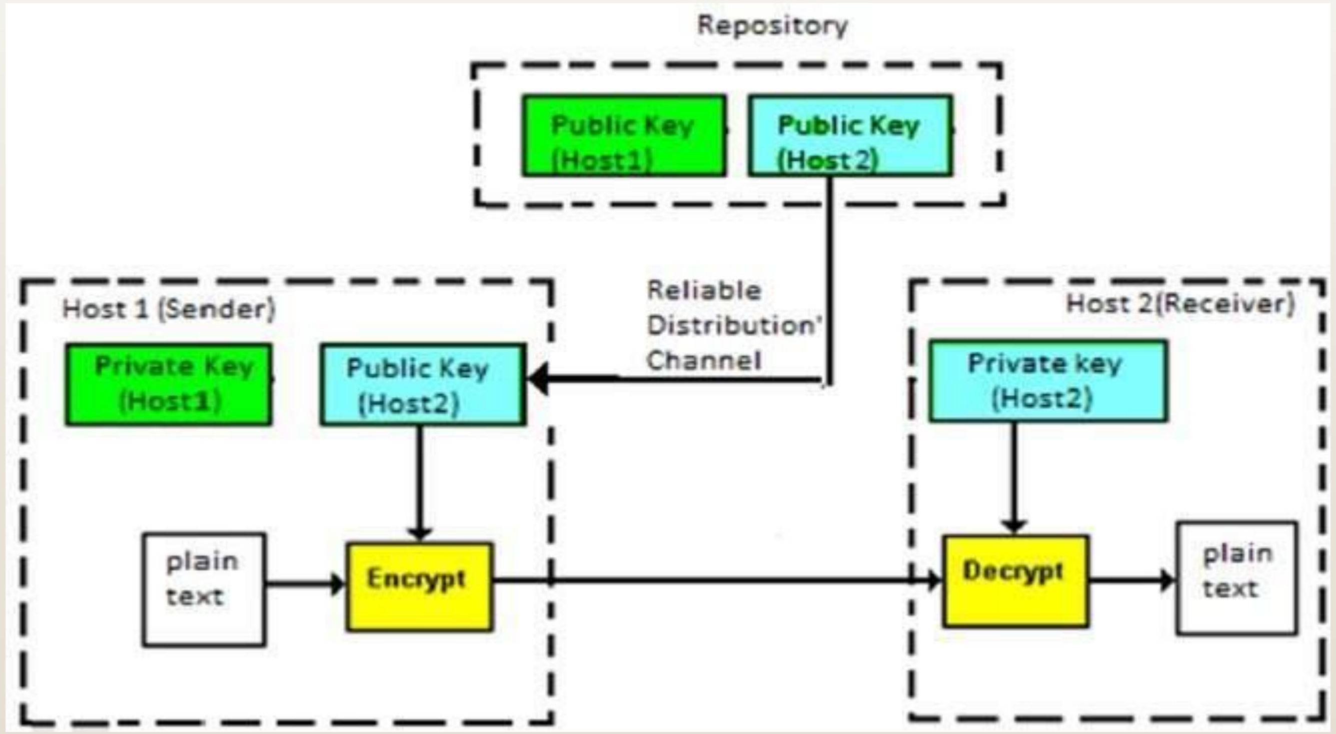


Symmetric Key Encryption Features

- Persons using symmetric key encryption must share a common key prior to exchange of information.
- Keys are recommended to be changed regularly to prevent any attack on the system.
- The length of a key (number of bits) is smaller, so the process of encryption-decryption is faster than asymmetric key encryption.
- Challenges:
    - **Key establishment** – Before any communication, both the sender and the receiver need to agree on a secret symmetric key. It requires a secure key establishment mechanism in place.
    - **Trust Issue** – Since the sender and the receiver use the same symmetric key, there is an implicit requirement that the sender and the receiver 'trust' each other. For example, it may happen that the receiver has lost the key to an attacker and the sender is not informed.

Assymetric Key Encryption

- The encryption process where **different keys are used for encrypting and decrypting the information**
- Though the keys are different, they are mathematically related and hence, retrieving the plaintext by decrypting cipher text is feasible.
- Examples: RSA, ElGamal



Assymetric Key Encryption Features

- Every user in this system needs to have a pair of dissimilar keys, **private key** and **public key**.
- When one key is used for encryption, the other can decrypt the cipher text back to the original plaintext.
- It requires to put the public key in a public repository and the private key as a well-guarded secret -> **Public Key Encryption**.
- Though public and private keys of the user are related, it is computationally not feasible to find one from another -> That's the strength of this scheme.
- When *Host1* needs to send data to *Host2*, it obtains the public key of *Host2* from repository, encrypts the data, and transmits.
- *Host2* uses its private key to extract the plaintext.

HTTPS, SSL and TLS – Why and how do theywork?

- What is HTTPS?
  - Also known as HTTP over Transport Layer Security (TLS),HTTP over SSL, and HTTP Secure
  - It’s a communications protocol for secure communication over a computer network.
  - Operates in the Application Layer of the TCP/IP suite.
  - Consists of HTTP + a connection encrypted by **TLS** or **SSL**.
  - Motivation: Authentication of the visited website and protection of the privacy and integrity of the exchanged data
  - Default port: 443, URLs: https://
- Why?
  - HTTP isn’t encrypted and is vulnerable to man-in-the-middle and eavesdropping attacks, which can let attackers gain access to website accounts and sensitive information.

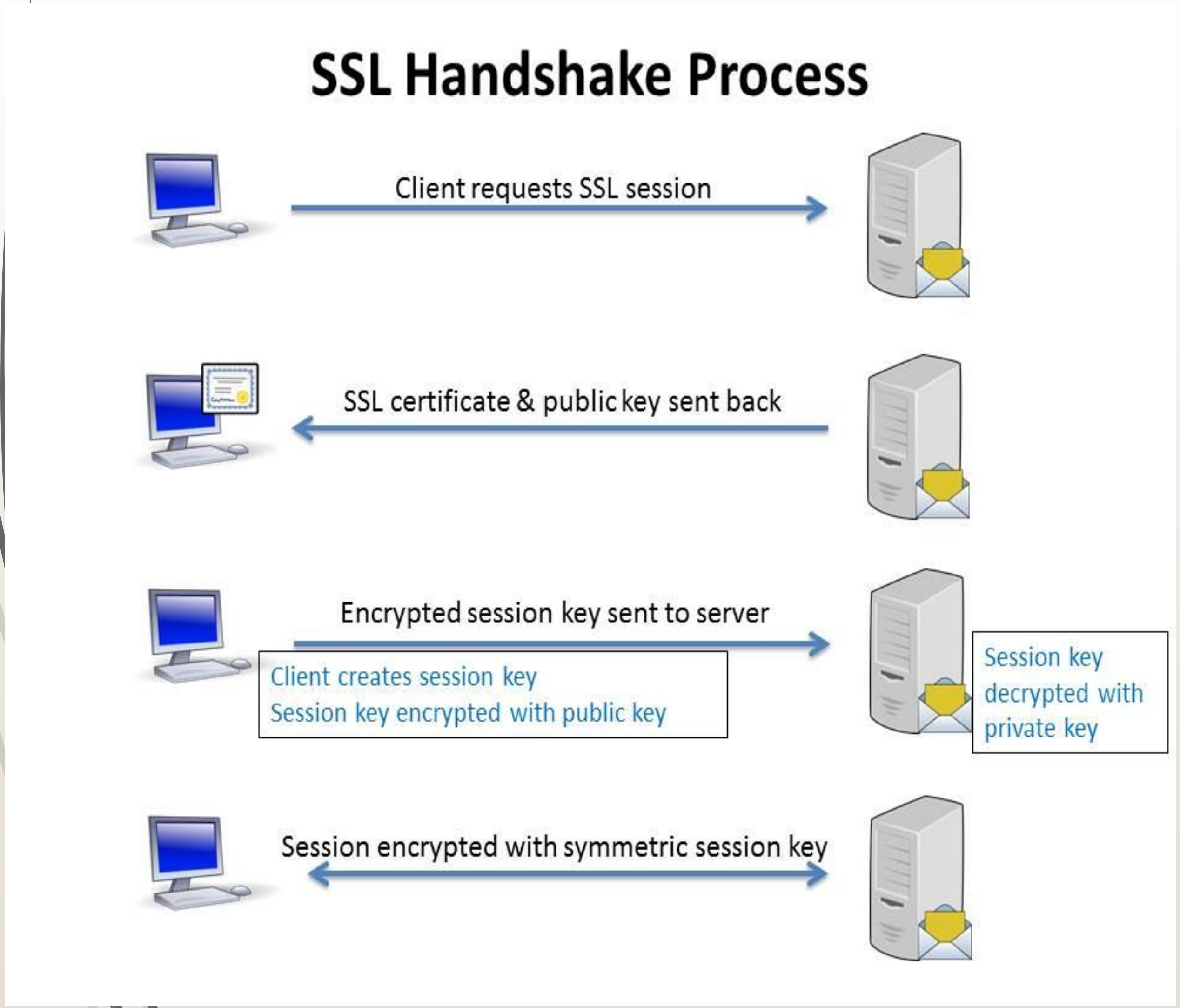
TLS/SSL

- TLS (Transport Layer Security)
- SSL (Secure Sockets Layer)
- Who are they?
  - They're cryptographic protocols providing communications security over a computer network from web browsing, email, Internet faxing, VoIP, IM (Instant Messaging: online chats).
  - They secure all communications between two communicating applications (processes)

TLS/SSL Properties

- TLS/SSL operates directly on top of TCP, so protocols on the higher layer such as HTTP can be left unchanged providing a secure connection.
- If TLS/SSL is used correctly, All an attacker can see on the cable is:
  - The IP address and port number you're connected to,
  - How much data you're sending.
  - Which encryption and compression is used.
- An attack may also terminate the connection, but both sides will not it has been interrruptеd by a 3rd party.
- In typical use, the attacker will also be able to figure out which host name you're connecting to.
- Although HTTPS itself does not expose the host name, your browser will usually need to make a DNS request first to find out what IP address to send the request to.

TLS/SSL Handshake Process


SSL Handshake Process
Client requests SSL session
SSL certificate & public key sent back
Encrypted session key sent to server
Client creates session key
Session key encrypted with public key
Session key decrypted with private key
Session encrypted with symmetric session key

http://blogs.mdaemon.com/wp-content/uploads/2016/08/How-SSL-Works.jpg

1. Build a TCP connection (establish TCP sockets on both sides of the communication channel).
2. Start the TLS/SSL handshake. The client sends:
   - Version of TLS/SSL it's running.
   - What ciphersuites it wants to use.
   - What compression methods it wants to use.
3. The server checks what the highest TLS/SSL version is supported by them both, picks a ciphersuite from one of the client's options and optionally a compression method.

4. Once the basic setup's done, the server sends its certificate.
   - The certificate must be trusted by either the client itself or a party that the client trusts e.g. If the client trusts GeoTrust, then the client can trust the certificate from Google.com, because GeoTrust cryptographically signed Google's certificate.
5. Once the certificate's verified and being certain this server really is who it claims to be, a key is exchanged.

- The key can be a public key e.g. „PreMasterSecret" or nothing, depending on the ciphersuite.
- Both the client and the server can compute the key for the symmetric encryption.

6. The client tells the server that from now on, all communication will be encrypted, and sends and encrypted and authenticated message to the server (MAC: Message Authentication Code).
7. The server verifies that the MAC is correct, and that the message can be correctly decrypted. It then returns a message, which the client verifies as well.
8. The handshake is finished and the two hosts can communicate securely.

Computer Network Communications Example

- How does it all work in real life?

References

- Cisco CCNA ICND1, Wendell Odom
- https://en.wikipedia.org/wiki/Internet_protocol_suite
- https://www.tutorialspoint.com/cryptography/
- https://security.stackexchange.com/questions/20803/how-does-ssl-tls-work
- https://en.wikipedia.org/wiki/Transport_Layer_Security
- https://en.wikipedia.org/wiki/User_Datagram_Protocol