IIT KHARAGPUR

NPTEL ONLINE
CERTIFICATION COURSES

NPTEL

E-BUSINESS

PROF. MAMATA JENAMANI
DEPARTMENT OF INDUSTRIAL AND SYSTEMS ENGINEERING
IIT KHARAGPUR

Week 6: Lecture 1

SECURITY CATEGORIES

We are going to learn

- Security categories
- Different types of attacks

Security Categories

- **Legitimate use**
  - identification, authentication and authorization.
- **Confidentiality**
  - protecting the content of messages or data transmitted over the Internet from the unauthorized people.
- **Integrity**
  - preventing data from being modified by an attacker. Transmitting information over the Internet
- **Availability**
  - systems, data, and other resources are usable when needed despite subsystem outages and environmental disruptions.
- **Non-repudiation**
  - preventing the sender of a message from denying having sent it.
- **Auditing or Traceability:**
  - process of examining transactions.

Authentication

- A process by which two parties involved in a dialogue are given a guarantee that they are indeed interacting with whom they think they are interacting.
- Confirming the Identity of the interacting party
- Server Authentication
  - Ex: Purchasing the book from the right server and not the imposter
- Client Authentication
  - Ex: Communicating with the right client in case of Internet Banking

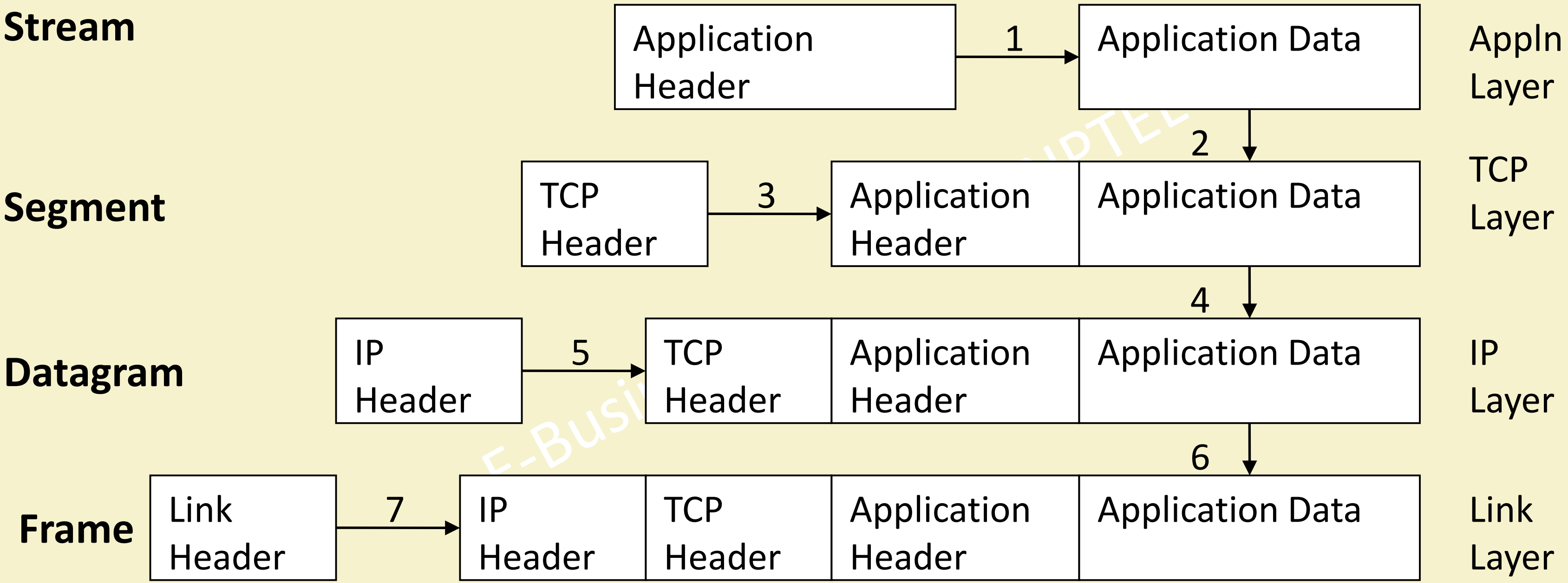
Confidentiality

- Protecting the content of messages or data transmitted over the Internet from the unauthorized people.
- Ex: Protecting credit card information during transmission over the Internet

Data Integrity

- Related to preventing data from being modified by an attacker
- Ex: Modifying a book order or changing the delivery address
- Attacker
  - Active: Modifies a packet
  - Passive: Only listens
- Types of Attacks
  - Sniffing
  - IP Spoofing

Processing at Each Layer



Stream — Application Header →1→ Application Data — Appln Layer

↓2

Segment — TCP Header →3→ Application Header | Application Data — TCP Layer

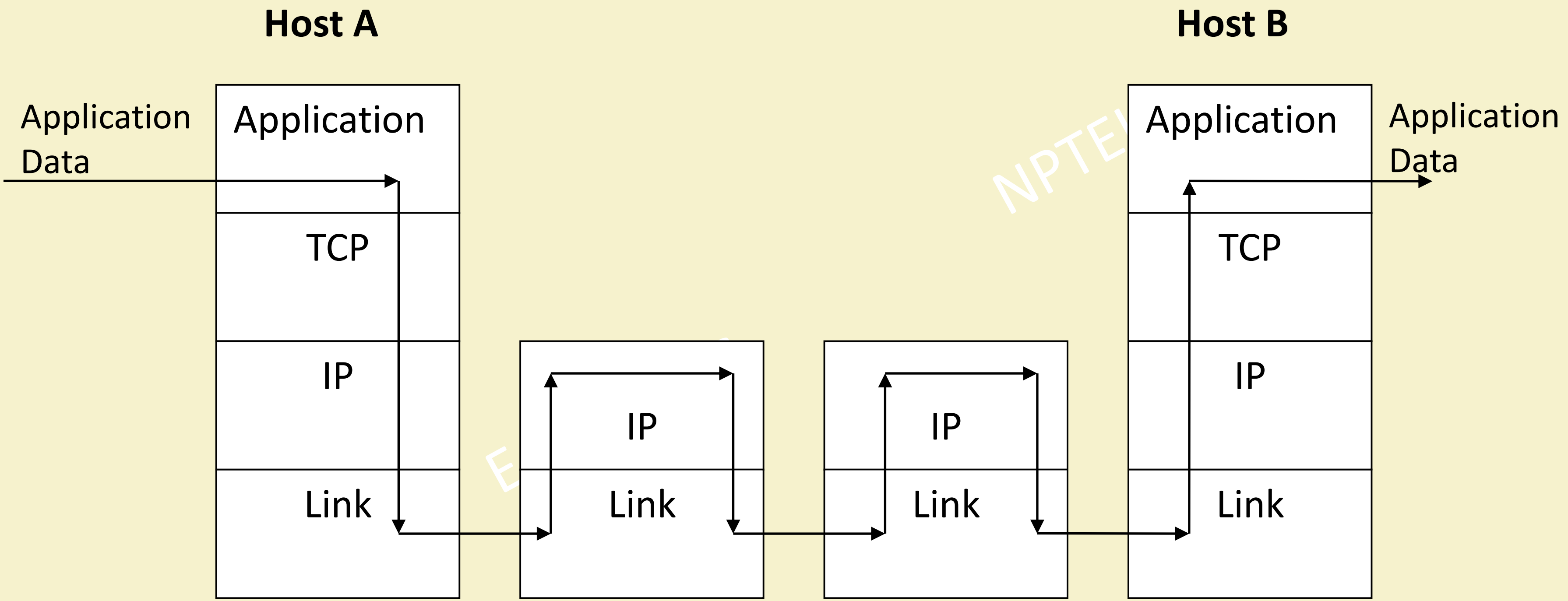
↓4

Datagram — IP Header →5→ TCP Header | Application Header | Application Data — IP Layer

↓6

Frame — Link Header →7→ IP Header | TCP Header | Application Header | Application Data — Link Layer

Transfer of Packet



IP Addresses

0 IPv4 Header Format 31

<table>
<tr><td colspan="3">Other Control fields</td></tr>
<tr><td colspan="3">Other Control fields</td></tr>
<tr><td>TTL</td><td>PID</td><td>Check Sum</td></tr>
<tr><td colspan="3">Destination Address</td></tr>
<tr><td colspan="3">Source Address</td></tr>
<tr><td colspan="3">Options and Padding</td></tr>
</table>

Data Integrity

- Types of Attacks
  - Sniffing
    - A packet sniffer is a program running in a network attached device that progressively receives all the data link layer frames passing by the device's network interface
  - IP Spoofing
    - Modifying the IP addresses

Availability

- Denial of Service Attacks
  - Creating extra legitimate workload that the server cannot handle
  - Ex. Partially opened TCP Connection

TCP

- TCP (Transmission control protocol)
  - Connection oriented
  - Handshaking
  - Source port, destination port, sequence number and acknowledgement.
  - Sliding window mechanism

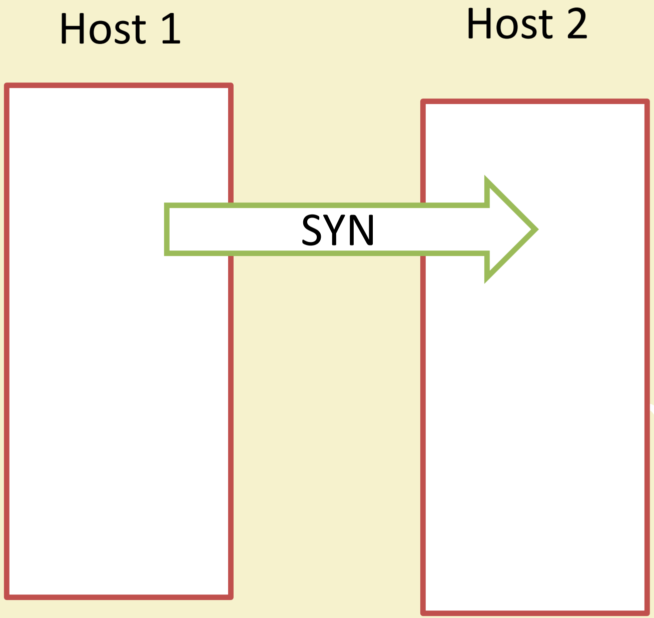
UDP Header Format

0	31
Source Port	Destination Port
Length	Checksum

TCP Header Format

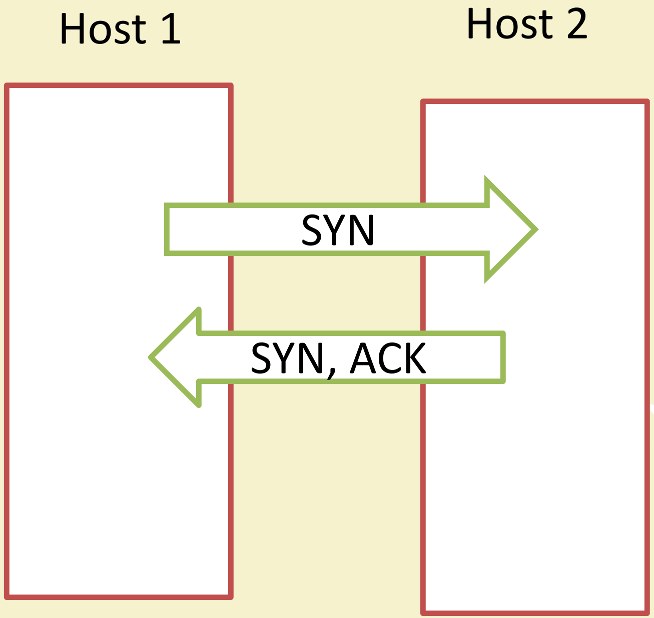
0		31
Source Port		Destination Port
Sequence Number		
Acknowledgement Number		
Off	Flags	Window
Check Sum		Urgent Pointers
Options and Padding		

Three way handshake in TCP



Step1: Host 1 wants to initiate a connection with Host 2, So Host 1 sends a segment with SYN(Synchronize Sequence Number). This segment will inform the Host 2 that Host 1 would like to start a communication with Host 2 and informs Host 2 what sequence number it will start its segments with.

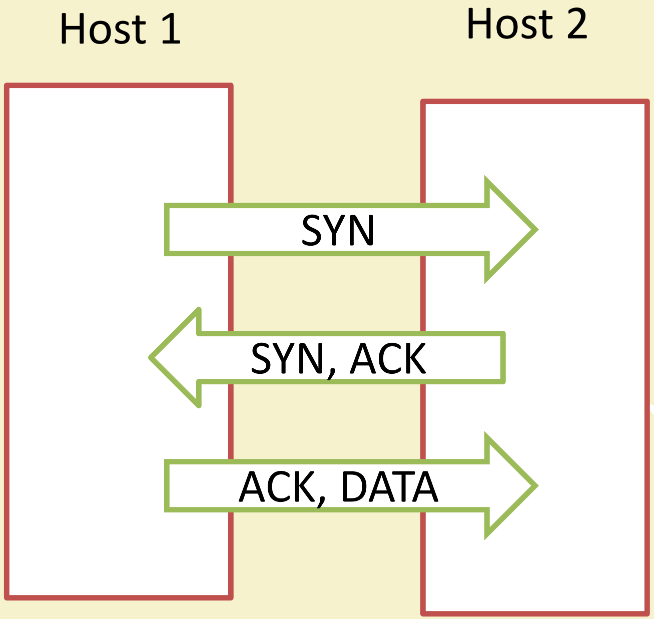
Three way handshake in TCP



Step2: Host 2 will respond to Host 1 with "Acknowledgment" (ACK) and SYN bits set. Host 2's ACK segment does two things;

1. It acknowledges Host 1's SYN segment.
2. It informs Host 1 what sequence number it will start its data with.

Three way handshake in TCP



Step 3:Now finally Host 1 Acknowledges Host 2's initial sequence Number and its ACK signal. Then Host 1 will start the actual data transfer.

Non-Repudiation

- An attribute of secure system that prevents the sender of a message from denying having sent it.
- Ex: Denying an order you have placed.

Week 6: Lecture 2

SECURITY TERMINOLOGIES

We are going to learn

- Security terminologies
- Security Jargons

Cryptography

- Cryptography is a technique by which data, called *plaintext*, is scrambled or *encrypted* in such a way that it becomes extremely difficult, expensive and time consuming for an unauthorized person to unscramble or *decrypt* it.
- The encrypted text is called the ciphertext

The steps in Cryptography

- Encryption

  $EncryptedMsg$ = Encrypt (Msg, key_e)

- Decryption

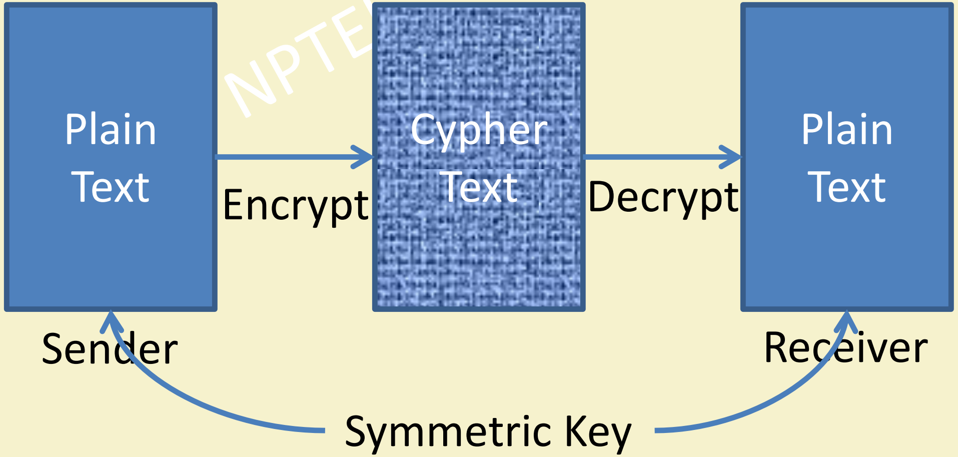
  Msg = Decrypt ($EncryptedMsg, key_d$)

Types of cryptographic Algorithm

- Symmetric key cryptographic Algorithm
- Asymmetric key cryptographic Algorithm

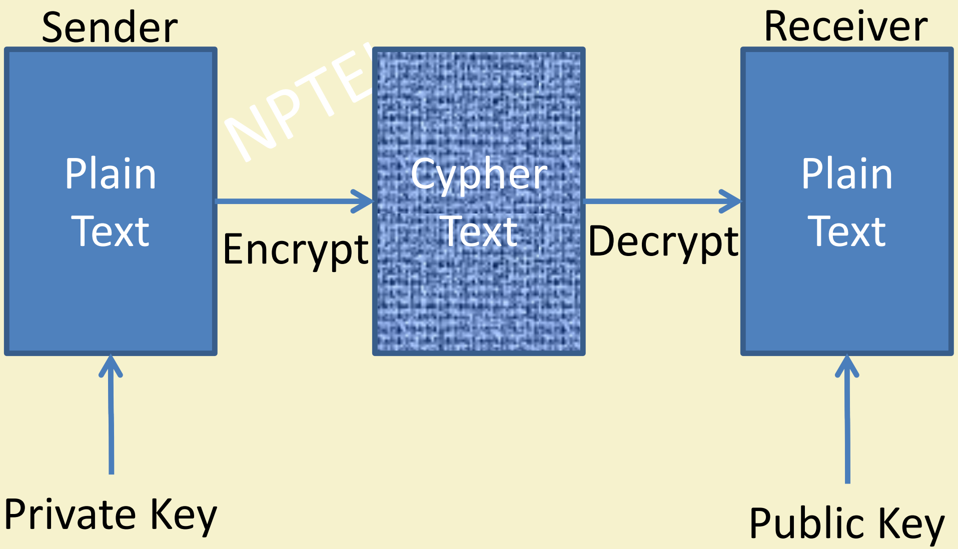
Types of cryptographic algorithms

- *$key_e = key_d$*
  - Same key for encryption and decryption
  - Ex: DES (Data Encryption Standard), TDES, IDEA, RC2, RC4, RC5
  - Hardware implementation is 100 times faster than the SW implementation
  - Problems of key distribution
  - Cannot be used for authentication or non-repudiation process



Asymmetric key cryptographic algorithms

- *key_e<> key_d*
  - Different key for encryption and decryption
  - Public (Known to everybody) and private key (Known to the owner)
  - Ex: RSA (Ron Rivest, Adi Shamir and Leonard Adleman), patented till 2000
  - Much slower than symmetric key cryptographic algorithms (RSA is 100 times slower than DES)
  - Private Key operation time slower than public key operation
  - Time grows with length of the key in bits



Other terminology associated with cryptography

- **Strength:** The strength of encryption is determined by the key size. Asymmetric algorithms require large keys

1024 bits	Low-strength asymmetric key
2048 bits	Medium-strength asymmetric key
4096 bits	High-strength asymmetric key

  - Symmetric keys are smaller: 256 bit keys give you strong encryption.

Other terminology associated with cryptography

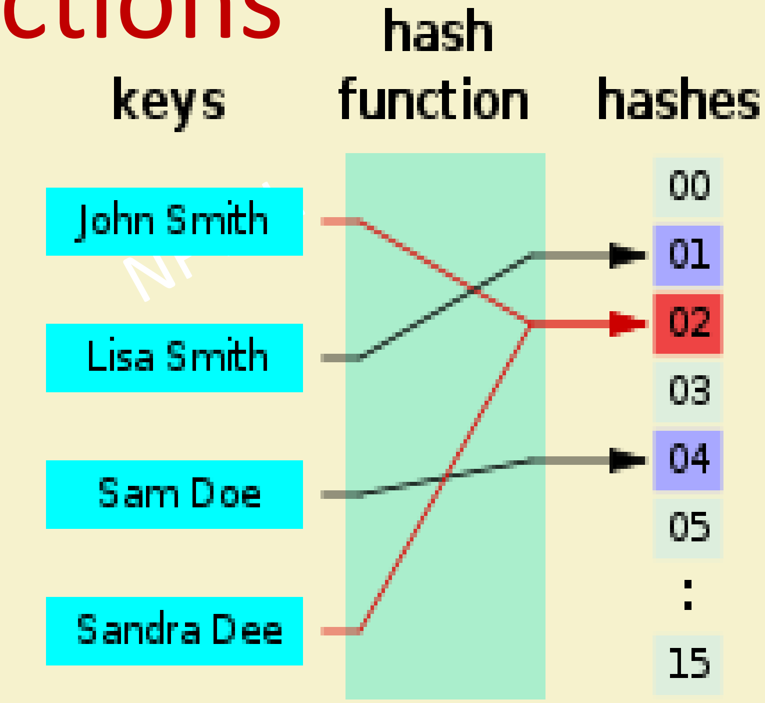
- **Block cipher algorithm:** These algorithms encrypt data by blocks. For example, the RC2 algorithm from RSA Data Security Inc. uses blocks 8 bytes long. Block algorithms are typically slower than stream algorithms.
- **Stream cipher algorithm:** These algorithms operate on each byte of data. Stream algorithms are typically faster than block algorithms.

Points to note

- Generally bulk data transfer is not done by asymmetric key cryptography.
- Session key exchange is done using RSA followed by DES for bulk data transfer

Hash Functions

- A hash function is any function that can be used to map data of arbitrary size to data of fixed size.
- The values returned by a hash function are called hash values, hash codes, digests, or simply hashes.



https://en.wikipedia.org/wiki/Hash_function

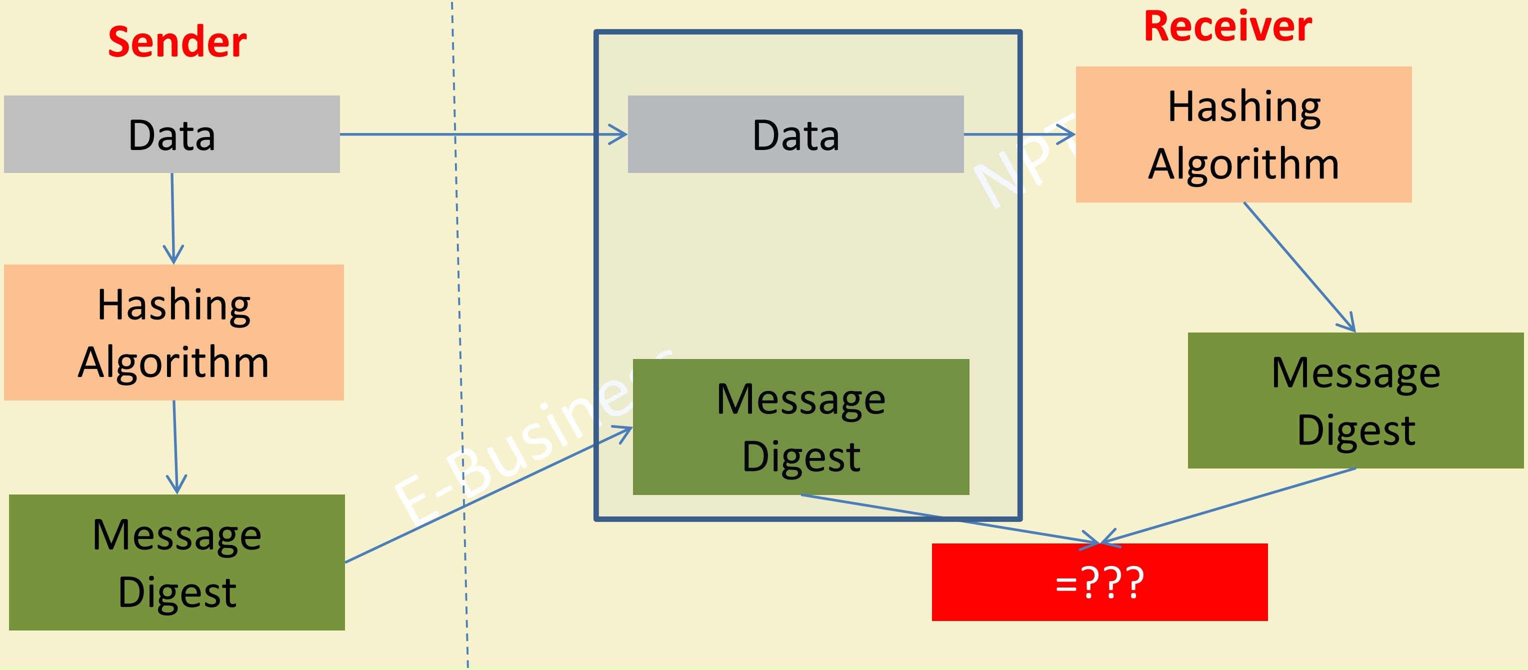
Use of Hash Functions

- Not used for Encryption
- Used for authentication and data integrity
  - – Masquerade – Insertion of message from fraudulent source
  - – Content Modification – Changing content of message
  - – Sequence Modification – Insertion, deletion and reordering sequence
  - – Timing Modification – Replaying valid sessions

Property of a Hash function

- It should be easy to compute h(Msg)
  - Msg is the message to be sent
- It should be hard to obtain Msg given h(Msg)
- It should be very hard to find another message Msg$^{/}$ such that h(Msg) = h(Msg$^{/}$)

Data Integrity through hash function



Applications of hash functions

- Public Key Algorithms
  - Password Logins
  - Encryption Key Management
  - Digital Signatures
- Integrity Checking
  - Virus and Malware Scanning
- Authentication
  - Secure Web Connections

Variants of hashing algorithms

- MD4 and MD5 by Ron Rivest (1990,1994)
- SHA-0, SHA-1 by NSA (1993, 1995)
- RIPEMD-160 (1996)
- SHA-2 (2002 – 224, 256, 385, 512)
- Whirlpool
- Tiger
- GOST-3411
- SHA-3

Security Jargons

- ***Adware*** — a general term used for software that invades your computer in the form of persistent pop-up ads.
- ***Cracker*** — someone who looks for and breaks into computers or networks without authorization, either for the fun of it or to steal valuable information such as credit card numbers; also called a "black hat" hacker.

https://ittraining.iu.edu/workshops/win_security/terminology.html

Security Jargons

- ***Firewall*** — software, hardware or both used to block unauthorized access to a machine or a network. A firewall can be internal (on an individual machine) or external (a separate piece of hardware on a network protecting multiple machines)

https://ittraining.iu.edu/workshops/win_security/terminology.html

Security Jargons

- ***Hacker*** — a general term used for anyone who spends time poking into computers and operating systems, trying to discover their vulnerabilities.
- ***Intruder*** — any unauthorized individual who tries to access a computer system from outside; also referred to as an attacker.

Security Jargons

- ***Malware*** —A new term which is emerging to refer to any software written with malicious intent. Term is derived from **mal**icious soft**ware**.
- ***Probe*** — a program used to gather information about a system or its users.
- ***Risk*** — the probability that a vulnerability will cause a harmful result.

Security Jargons

- ***Trojan horse*** **—** "back door" software program that allows intruders to take remote control of a computer without the owner’s knowledge. Trojans can be installed on computers through thousands of free software packages that can be downloaded from the Internet.

Security Jargons

- ***Rootkit*** — an especially heinous Trojan Horse program or group of programs that can completely hide itself from a virus scan program by integrating itself into the core of the operating system. Rootkits typically start themselves before the machine's operating system making them capable of hiding multiple files, registry keys and/or programs from the operating system and thus the machine's virus scan software.

Security Jargons

- ***Social Engineering*** — the practice of obtaining confidential information by manipulation; for example, people claiming to be administrators may trick computer users in to divulging sensitive information such as passwords.
- ***Phishing*** — a form of social engineering where an attacker tries to fraudulently acquire sensitive information, such as a password, bank account number, social security number, etc., by masquerading as a trustworthy entity with official looking electronic communication (email, instant message, etc.).

Security Jargons

- ***Spyware*** **—** a general term used for software that performs certain “secret” behaviours such as advertising or collecting personal information, generally without obtaining your consent.
- ***System Compromise*** **—** a violation of security policy in which disclosure of sensitive information may have occurred.
- ***Threat*** **—** any event that may harm a system by means of destruction, disclosure, modification of data, and/or denial of service.

Security Jargons

- ***Virus* —** a piece of code that replicates by attaching itself to another object. It can attack the registry, replace system files, or take over email programs in its attempt to replicate itself.
- ***Vulnerability* —** a weakness in security procedures that may be used to violate a system security policy.
- ***Worm* —** an independent program that replicates by copying itself from one computer to another, usually over a network or through email attachments. A particularly common use of worms is to make computers spew out so much bad network traffic that they cause networks and servers to fail.

Week 6: Lecture 3

DIGITAL SIGNATURE

We are going to learn

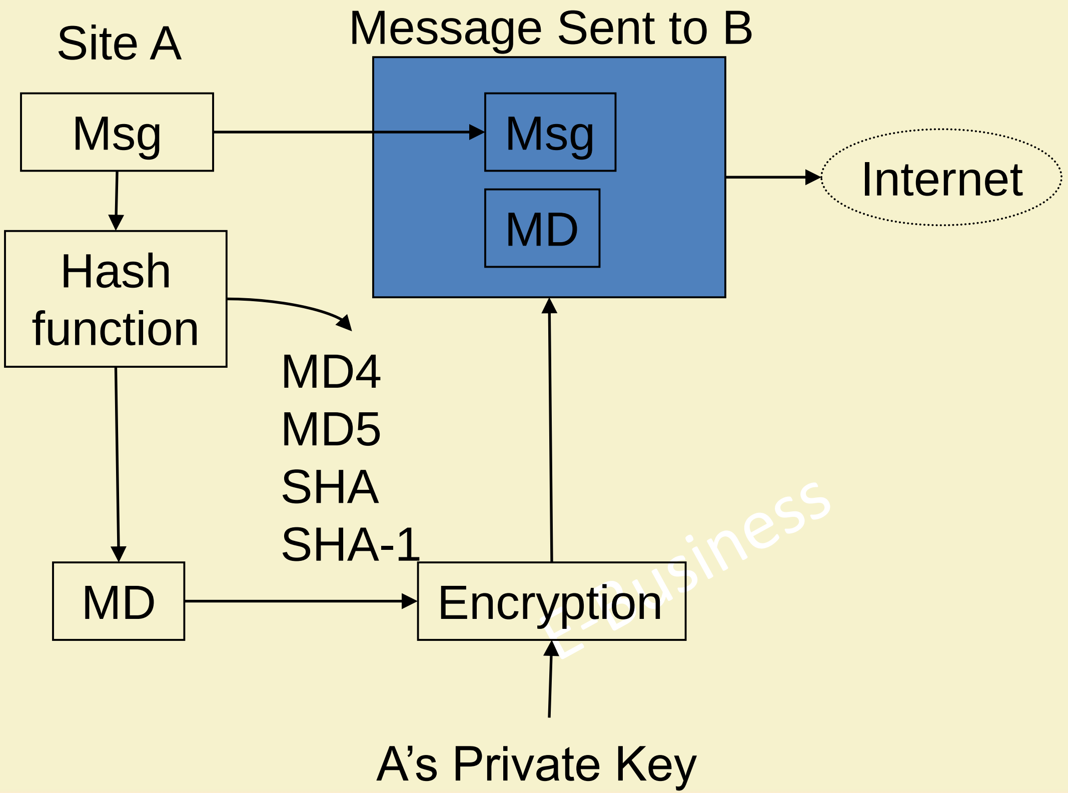
- Digital signature
- Digital certificates
- Public key infrastructure

What is digital signature

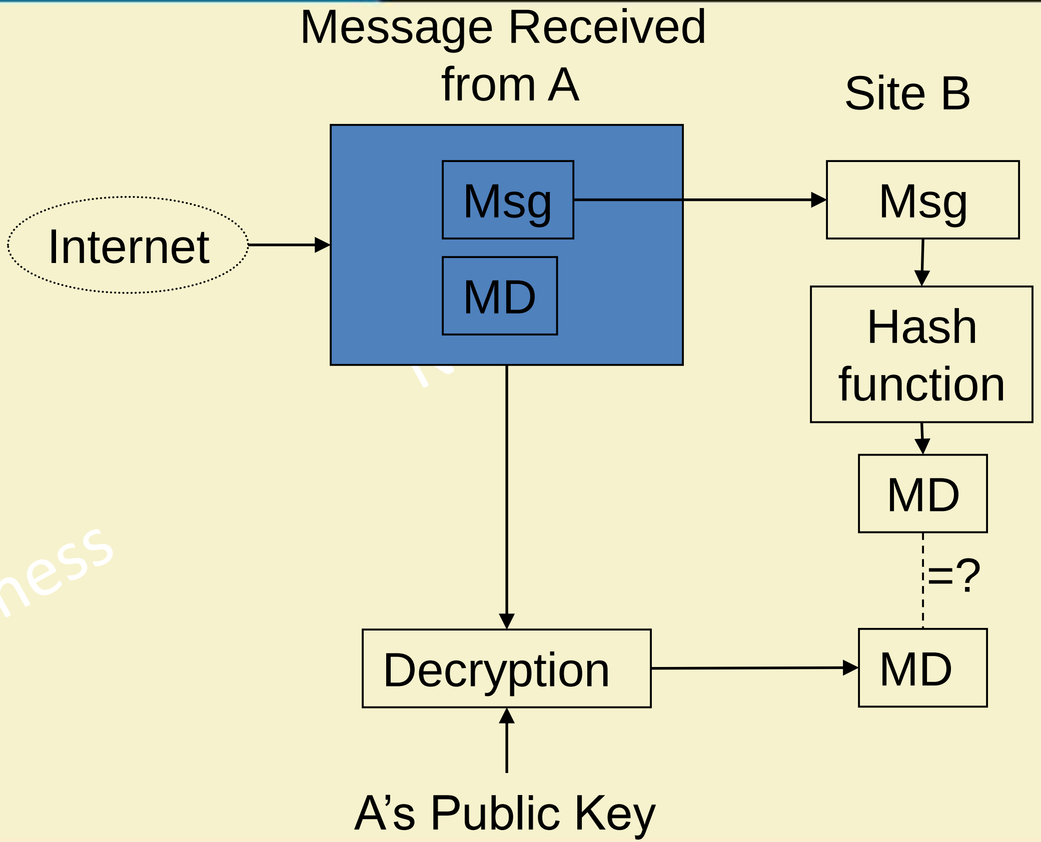
- A digital signature is a mathematical scheme for demonstrating the authenticity of digital messages or documents. A valid digital signature gives a recipient reason to believe that the message was created by a known sender (authentication), that the sender cannot deny having sent the message (non-repudiation), and that the message was not altered in transit (integrity).

https://en.wikipedia.org/wiki/Digital_signature

Digital Signature generation process

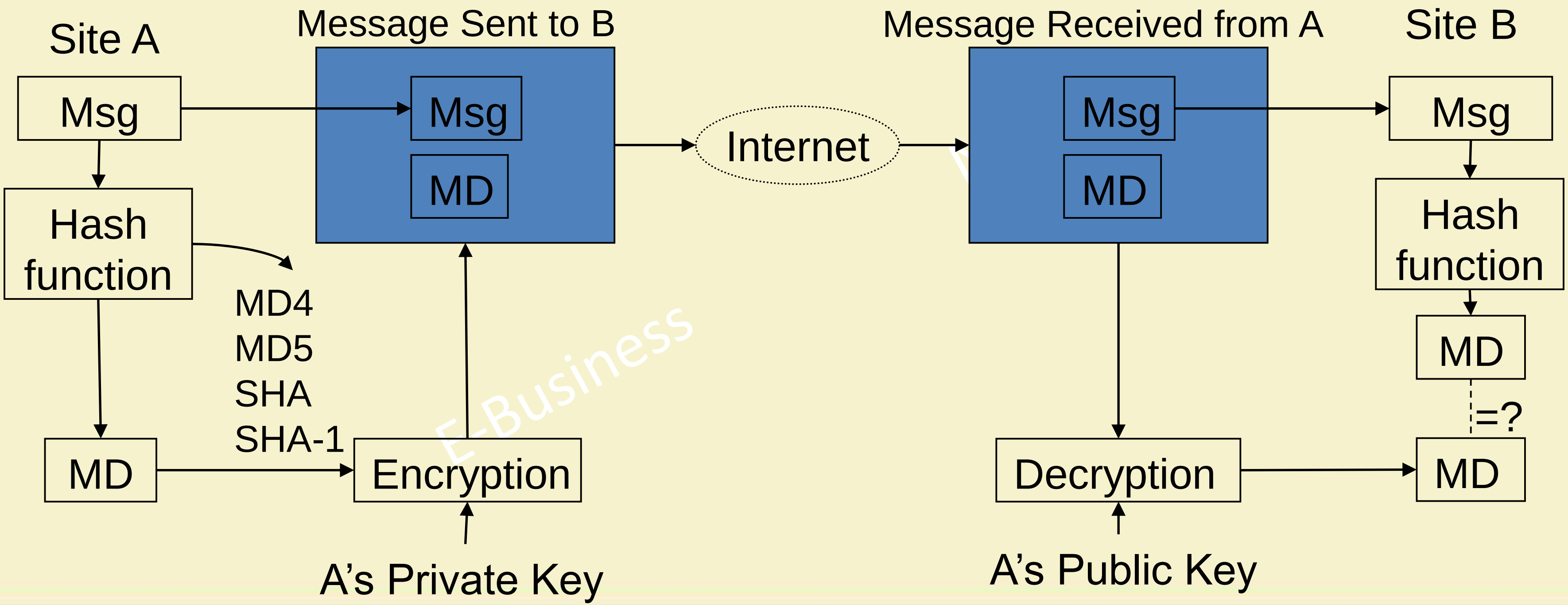


Digital Signature Verification



Security categories addressed by Digital Signature

–Authentication, Non-Repudiation and Data Integrity



Digital certificate

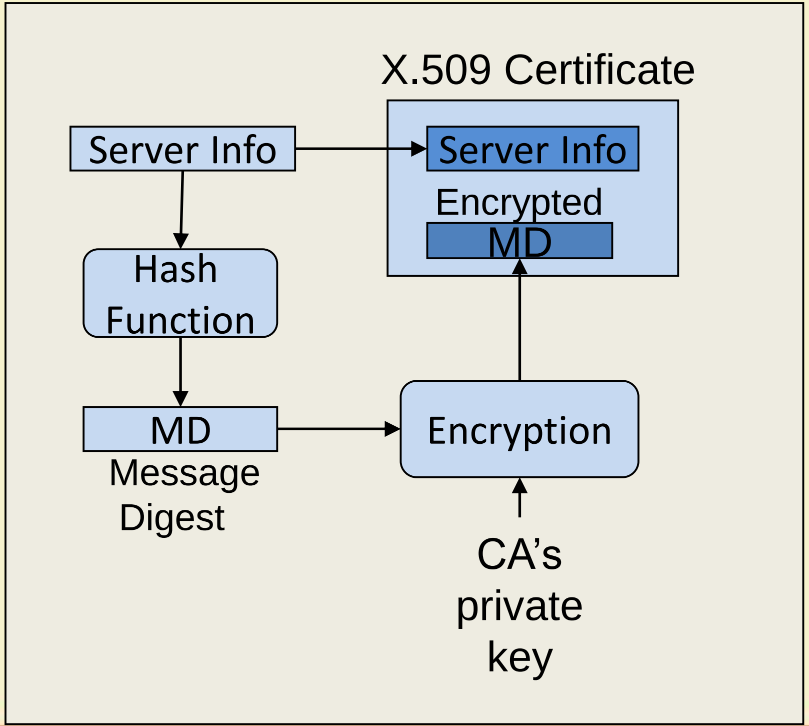
- A digital certificate, also known as a public key certificate or identity certificate, is an electronic document used to prove the ownership of a public key. The certificate includes information about the key, information about the identity of its owner (called the subject), and the digital signature of an entity that has verified the certificate's contents (called the issuer).
- Server certificate
- Client certificate

https://en.wikipedia.org/wiki/Public_key_certificate

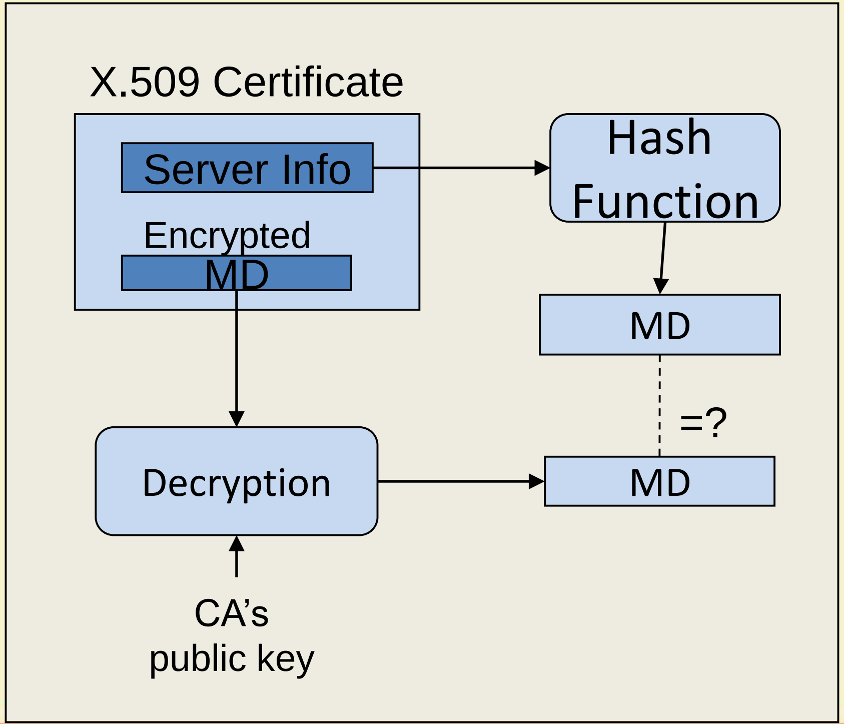
Public Key Infrastructure
-Solution to the e-Business security

- A public key infrastructure(PKI) is a foundation on which other applications, system, and network security components are built.
- Public Key cryptography supports security mechanisms such as integrity, authentication, and non-repudiation.
- To participate in a PKI, an end entity must enroll or register in a PKI. The result of this process is the generation of a public key certificate.
- The binding is declared when a trusted CA digitally signs the public key certificate with its private key.

Generation of server certificate



Verification of server certificate



Components of the server info

- Name
- Issuer CA
- Serial No
- Validity
- Public key of the server
- …

Important PKI Functions

- **Public key cryptography** – Includes the generation, distribution, administration, and control of cryptographic keys.
- **Certificate issuance** – Binds a public-key to an individual, organization, other entity, or to some other data—for example, an email or purchase order.
- **Certificate validation** – Verifies that a trust relationship or binding exists and that a certificate is still valid for specific operations.
- **Certificate revocation** – Cancels a previously issued certificate and either publishes the cancellation to a Certificate Revocation List or enables an Online Certificate Status Protocol process.

PKI infrastructure in India

- Controller of Certifying Authority (CCA) India is at the root
- **CAs licensed by the CCA**

  a. Safescrypt
  b. NIC
  c. IDRBT
  d. TCS
  e. MtnlTrustline
  f. iCertCA
  g. GNFC
  h. e-Mudhra CA
- Certificate Enrolment process
- CST (Cryptographic service provider) (http://cca.gov.in/rw/pages/index.en.do)

Week 6: Lecture 4

PROTOCOLS FOR SECURITY: TLS

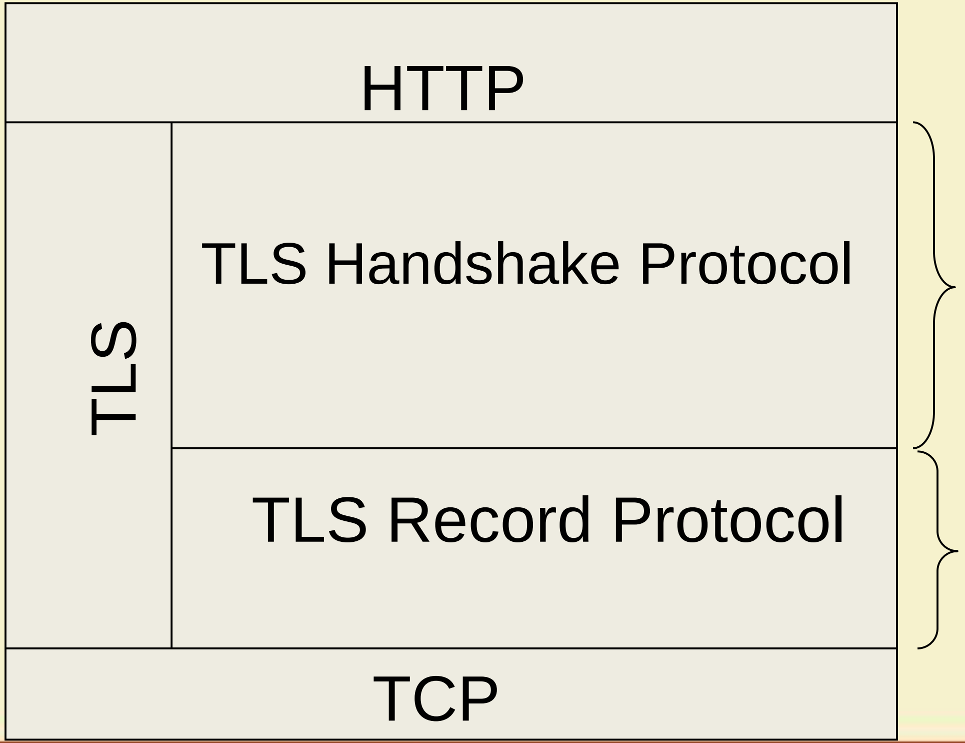
We are going to learn

- Steps involved in TLS protocol

Secure Socket Layer (SSL)
Transport Layer Security (TLS)

- SSL First developed by Netscape
  - Superceded by Transport Layer Security (TLS) Protocol
  - Minor changes over SSL 3.0
  - IETF RFC 2246
- A session layer protocol and runs on top of TCP
- Authentication, Confidentiality, Non-repudiation

HTTP, TCP – TLS in context



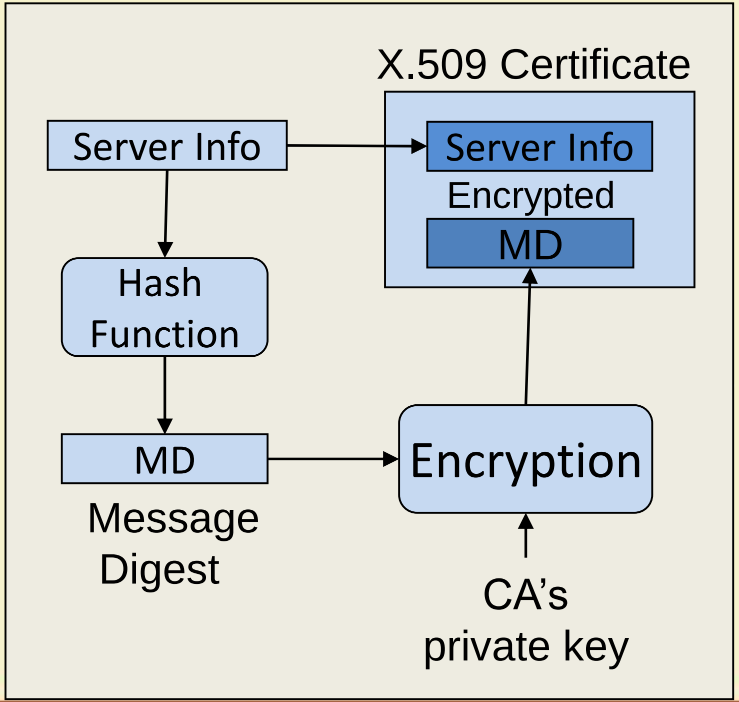
-Negotiation of cryptographic and compression algorithms
-Exchange of secrets through PK
-Generation of secrete key

-Encryption/decryption
-Message authentication
-Compression/Decompression

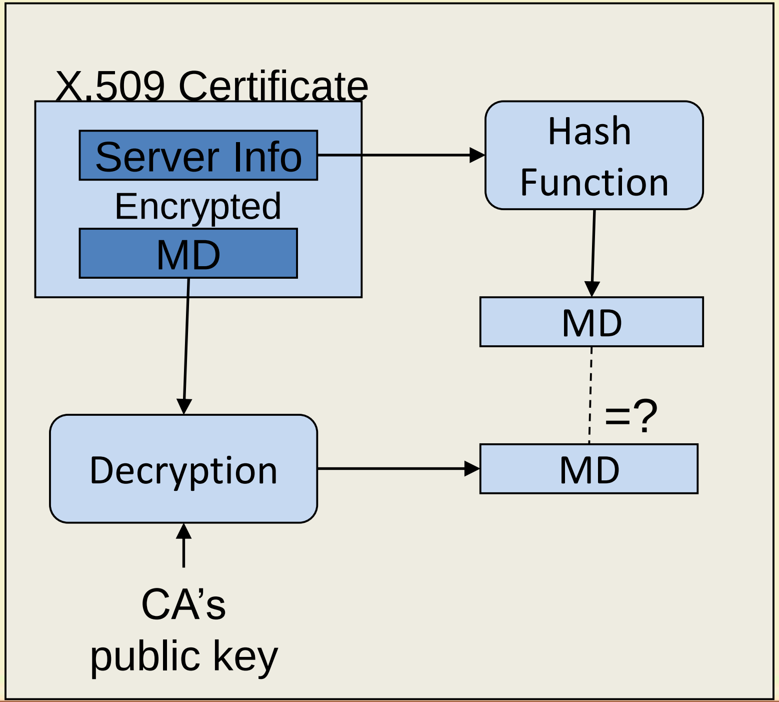
TLS – Handshake Protocol

- Selection of PK algorithm (e.g. RSA) and the key used for transmission of the shared secrete.
- Selection of bulk encryption algorithm (ex. DES) and the session key to be used during the session by the Record Protocol
- Message authentication code (MAC) to be used by the record protocol (Ex. MD5)
- Compression algorithm used by the Record Protocol
- Server authenticates itself to the client and the client occasionally authenticates itself during handshaking

Generation of server certificate



Verification of server certificate



Components of the server info

- Name
- Issuer CA
- Serial No
- Validity
- Public key of the server
- …

TLS Record Protocol

- Encryption/decryption
  - DES, TDES, RC4
- Message authentication
  - MD5, SHA, SHA1
- Compression/Decompression
  - Lempel-Ziv-Stac (LZS) compression
  - Overhead is negligible

Understanding Client Server Interaction

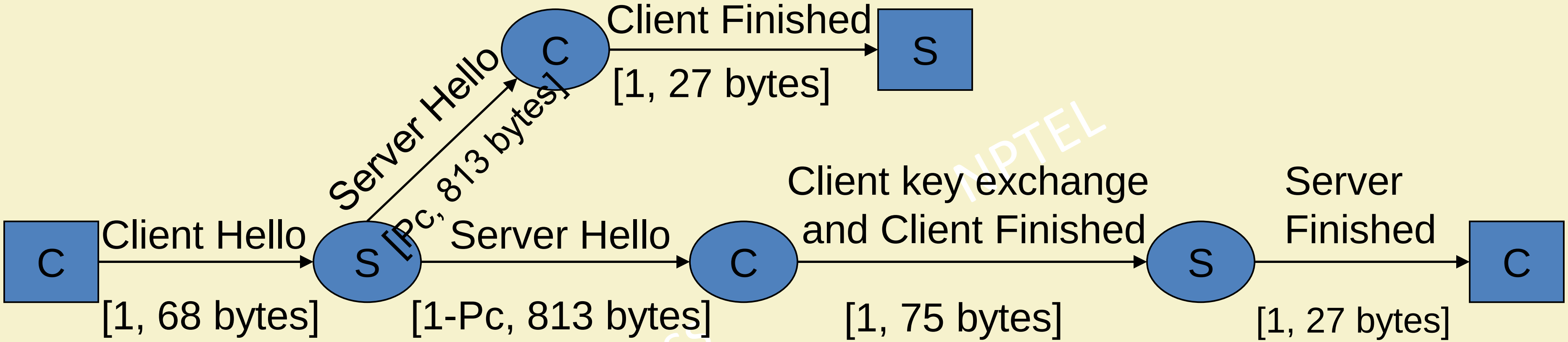
Client-Server Interaction

- Any e-business function is implemented through a set of client-server interactions.
  - HTTP request response, Remote Procedure Calls (RPC), Object Request Broker (ORB), CORBA, Web Services etc.
- C/S interaction starts with a client sending a request to the primary server and get a response.
- The primary server in tern may act as a client to anther server, send a service request and get a response.
- Many tools can be used to model this interaction
  - UML sequence diagrams, Message sequence charts, Specification and description language etc.
- But none of these tools annotated with performance related parameters
  - Client-Server Interaction Diagram

Client Server Interaction Diagram

- Two types of nodes
  - Client / Server
- Start node and finished nodes are shown as squares and internal nodes as circles
- Directed Arcs
  - Indicate a message being sent to another node
  - Labeled with the probability of that path and the size of the message

CSID for TLS Handshake



The *Client Hello* message

- A random number generated by the client (28 bytes)
- Time measured at the client (4 bytes)
- A session Id (0 to 32 byte)
- Cipher suit (set of cryptographic algorithms for key exchange, bulk encryption etc.) supported by the client (2 byte)
- Compression algorithm supported by the client (1 byte)
- Protocol Version (1 byte)
- Total 68 bytes (Max)

The *Server Hello* message

- X.509 Server Certificate (750 bytes)
- A server random number (28 bytes)
- A server session ID (0-32 bytes)
- Cipher suit supported by the server (2 bytes)
- Compression method supported by the server (1 bytes)
- Total 813 bytes

Client key exchange and *Client Finished* Message

- Client authenticates the server using its (server's) certificate
- Generates a premaster secrete and key to be used for bulk encryption using premaster secrete, server random number and client random number and encrypts using servers public key(48 bytes)
- Client finished message 27 bytes.

The *Server Finished* Message

- The server receives the *Client Key Exchange Message* and decrypts the premaster secrete using its private key.
- The server generates the key using the premaster secret and the client and server random numbers.
- Encrypts all the messages received from the client using the bulk encryption key and sends the message back to the client for the verification purpose in a *server finished* message (27 bytes).

Using the cached session state at the server

- The client sends the old *client session id* (it wants to reuse), a new random number and the other details to the server.
- If the client session id is cached at the server, it (server) sends back the same client id (in place of the server session id), a new server random number along with the other details.
- The client confirms with a *client finished* message.

Overhead due to TLS

- Increase in Round Trip Times (RTTs)
- Byte overhead
- Processing time at the client
  - Handshaking + Decryption + Verification
  - Handshaking
    - verification of server certificate + Encryption of master secretes with server's public key + Bulk Encryption Key generation from the master secrete
- Processing time at the server
  - Handshaking + Encryption + Message Digest Generation
  - Handshaking time
    - Decryption of master secretes with private key + Bulk Encryption Key generation from the master secrete

Week 6: Lecture 5

IMPACT OF SECURITY PROTOCOL ON SERVER PERFORMANCE

We are going to learn

- Steps involved in TLS protocol

Understanding the Effect of TLS on Web Server Throughput

- Web Server Throughput = Number of Completed Queries / Observation Period
- Throughput = 1/service demand
- Service Demand: Average response time per service request
- Web Server Throughput = 1/ service demand at the bottleneck device
- Bottleneck Resource: Device with the highest service demand
- Bottleneck Resource could be:
  - Server, storage, Network, Client

Example

- Average size of a file requested by the client = 16, 385 bytes
- Average CPU time for accessing a file at the server when secure connection is not required = 0.002 sec
- Average time for a disk access = 0.01 sec
- Average Network Delay = 0.001737 sec

- Find the throughput
  - insecure connection (Without TLS)
  - Secure connection for all the pages
  - Secure connection for a part of the site
  - Using cryptographic accelerator
  - Using cached session states

Throughput with insecure connection (Without TLS)

- Disk is the bottleneck device
- Throughput = 1/0.01 = 100 requests per second

Secure connection for all the pages

- Service demand at the client = Time for handshaking + decryption +verification
- Time for client side handshaking (In msec)

Key Size (bits)	Verification of server certificate	Encryption of the master secrete	Key generation	Total Time
512	2.4	1.31	0.10	3.81
768	3.61	2.61	0.10	5.87
1024	7.09	5.20	0.10	12.36

- Encryption/decryption and message digest generation/verification (In mbps)

Encryption/Decryption		MD Generation/ Verification	
RC4	140	MD5	180
DES	40	SHA	130
TDES	15	SHA1	130

- Assuming a 1024 bit key, RC4 and MD5 algorithms
- Service time at the client = Time for handshaking + decryption +verification

  =0.01239

  + (16, 384 * 8)/140*10^6

  + (16, 384 * 8)/180*10^6

  = 0.01405 sec

- Service demand at the server CPU = (Time for handshaking + encryption + MD generation)+ Actual service Time
- Time for server side handshaking (In msec)

Key Size (bits)	Decryption of the master secrete with private key	Key generation	Total Time
512	10.13	0.10	10.23
768	23.66	0.10	23.76
1024	47.93	0.10	48.03

- Encryption/decryption and message digest generation/Verification (In mbps)

Encryption/Decryption		MD Generation/ Verification	
RC4	140	MD5	180
DES	40	SHA	130
TDES	15	SHA1	130

- Service demand at the server CPU = (Time for handshaking + encryption + MD generation)+ Action service Time

=(0.04808

+ (16, 384 * 8)/140*10^6

+ (16, 384 * 8)/180*10^6)

+ 0.002

= 0.051694 sec

- Service Demand at various resources
  - Client: 0.01405 sec
  - Network: 0.001737 sec
  - Server CPU: 0.051694 sec
  - Server Disk: 0.01 sec
- Bottleneck resource: Server
- Throughput = 1/0.05169 = 19.3 requests per sec
- 20% of the throughput without TLS

Secure connection for a part of the site

- Assume 40% of the requests are for the insecure documents
- Service demand at the server CPU is the weighted average of the service demand for both the secure and insecure documents
- Service Demand at Server CPU = 0.4*0.002+0.6*0.051694 = 0.0318 sec
- CPU is still the bottleneck
- Throughput = 1/0.0318=31.43

Using cryptographic accelerator

- cryptographic accelerator-A PK Math processor
- Assume
  - Cryptographic accelerator increases handshake protocol by a factor of 50
  - it is used by both client and the server
- Service time at the client = Time for handshaking + decryption +verification

  $=0.01239/50 + (16, 384 * 8)/140*10^6 + (16, 384 * 8)/180*10^6$

  $= 0.001405$ sec

- Service time at the server CPU = (Time for handshaking + encryption + MD generation)+ Actual service Time

  =(0.04808/50

  + (16, 384 * 8)/140*10^6

  + (16, 384 * 8)/180*10^6)

  + 0.002

  = 0.004625

- Bottleneck is the Disk
  - Maximum throughput is 100 request per sec

Using Cached session states

- Assume 60% of the connections are for the cached states
- Service time at the server CPU for sessions with cached states = (Time for session key generation + encryption + MD generation)+ Actual service Time

  =(0.0001

  $+ (16, 384 * 8)/140*10^6$

  $+ (16, 384 * 8)/180*10^6)$

  + 0.002 = 0.00376

- Service time at the server CPU = 0.6*0.00376+0.4*0.05169 = 0.0229
- Throughput = 1/0.0229 = 43.67