UNIT- IV

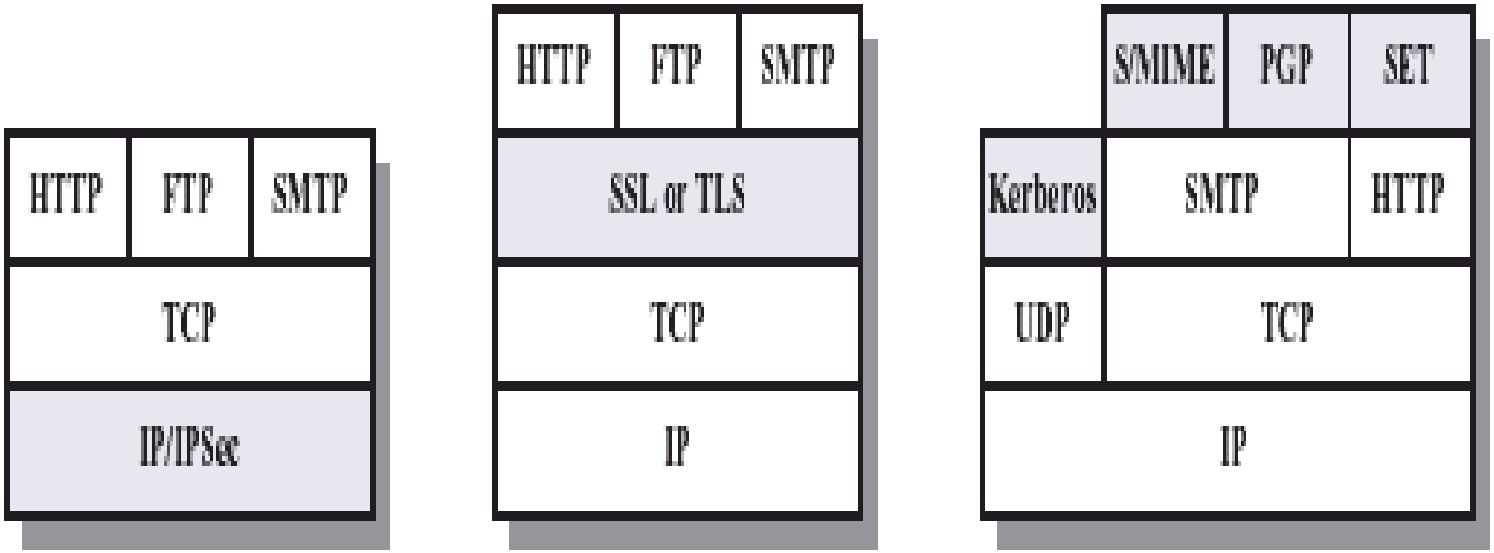
SLO - 1 & 2 :
Secure Sockets Layer(SSL)/ Transport Layer Security(TLS) Basic Protocol

Web Security Threats

- One way to group Web security threats(web) is in terms of passive and active attacks.
- Another way to classify Web security threats is in terms of the location of the threat:
  - Web server
  - Web browser
  - network traffic between browser and server.
- Issues of server and browser security fall into the category of computer system security.

Layers of Security

Relative Location of Security Facilities in the TCP/IP Protocol Stack



(a) Network Level (b) Transport Level (c) Application Level

IP/IPSec

- One way to provide Web security is to use IP security (IPsec) Figure(a)

Advantage of Ipsec :

- transparent to end users and applications
- provides a general-purpose solution.
- IPsec includes a filtering capability
  - » so that only selected traffic need incur the overhead of IPsec processing.

SSL/TLS

- Another relatively general-purpose solution is to implement security just above TCP - Figure (b)
- The foremost example of this approach is the Secure Sockets Layer (SSL) and the follow-on Internet standard known as Transport Layer Security (TLS).

At this level, there are two implementation choices.

1. For full generality, SSL (or TLS) could be provided as part of the underlying protocol suite and therefore be transparent to applications.
2. Alternatively, TLS can be embedded in specific packages.

SSL/TLS

- For example, virtually all browsers come equipped with TLS, and most Web servers have implemented the protocol.

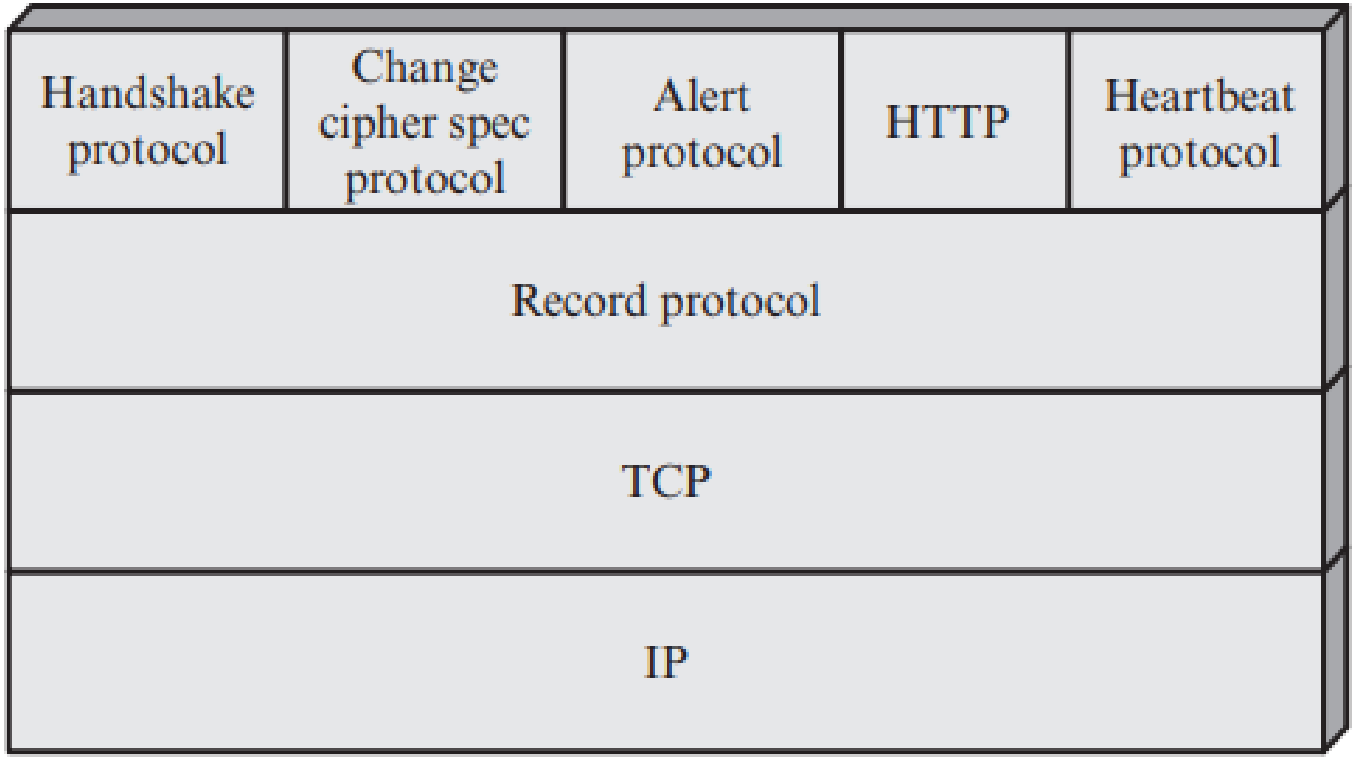
- Application-specific security services are embedded within the particular application.
  - Figure (c) shows examples of this architecture.
  - The advantage of this approach is that the service can be tailored to the specific needs of a given application.

SSL History

- Evolved through
  - Unreleased v1 (Netscape)
  - Flawed-but-useful v2
  - Version 3 from scratch
  - Standard TLS1.0
    - » SSL3.0 with minor tweaks, hence Version field is 3.1
- Defined in RFC2246, http://www.ietf.org/rfc/rfc2246.txt
- Open-source implementation at http://www.openssl.org/

TLS Architecture

- TLS is designed to make use of TCP to provide a reliable end-to-end secure service.
- TLS is not a single protocol but rather two layers of protocols

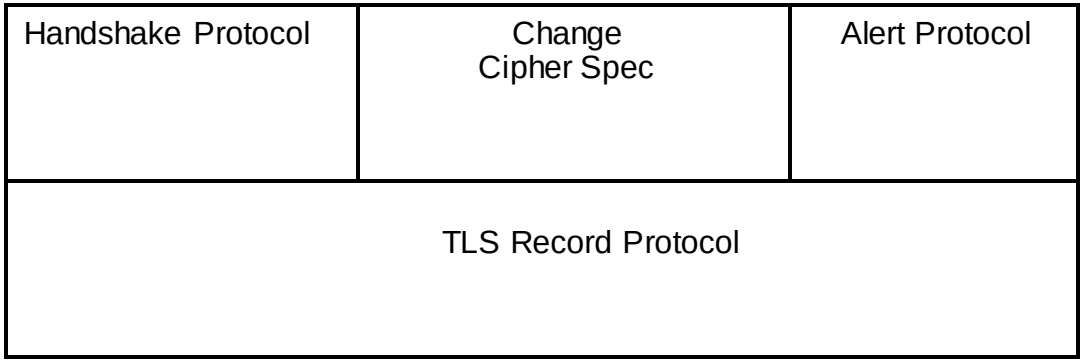


TLS Architecture

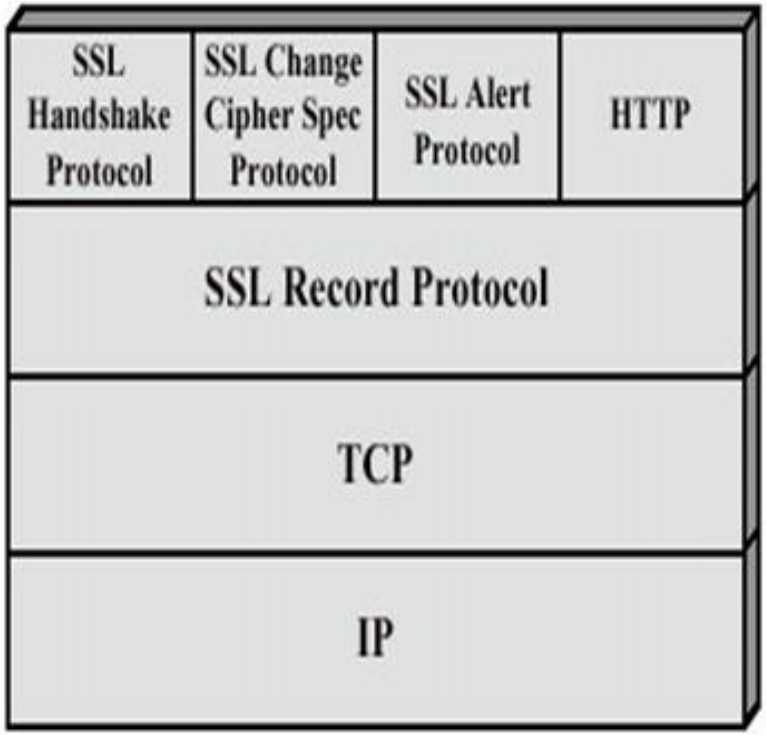
- The TLS Record Protocol provides basic security services to various higher layer protocols.
- In particular, the Hypertext Transfer Protocol (HTTP), which provides the transfer service for Web client/server interaction, can operate on top of TLS.
- Three higher-layer protocols are defined as part of TLS:
  - the Handshake Protocol;
  - the Change Cipher Spec Protocol; and
  - the Alert Protocol.

Architecture

- **Record Protocol to transfer application and TLS information**
- **A session is established using a Handshake Protocol**



Architecture(1)



- SSL vs. TLS
  - Layered on top of TCP to provide a reliable end-to-end secure service
  - SSL versions 1.0, 2.0, 3.0, 3.1
  - Netscape protocol
  - Later refitted as IETF standard TLS
  - TLS 1.0 very close to SSLv3.1

- uses TCP to provide a reliable end-to-end service. SSL is not single protocol but, Two layers of protocol

  - SSL Record Protocol
  - Three higher layer
    - » SSL Handshake protocol
    - » SSL Change Cipher Spec protocol
    - » SSL Alert protocol

Architecture(2) – Connection, Session

- Two important SSL concepts
  - SSL connection
    - » A transport (in the OSI 7 layer) that provides a suitable type of service
    - » Peer-to-peer relationships
    - » The connections are transient (ie. Temporary)
    - » Every connection is associated with one session
  - SSL session
    - » An association between a client and a server
    - » Sessions are created by the Handshake protocol
    - » Sessions define a set of cryptographic security parameters, which can be shared among multiple connections
    - » Sessions are used to avoid the expensive negotiation of new security parameters for each connection
- Between any pair of parties(applications such as HTTP on client and server), there may be multiple secure connections

Architecture(3)-States parameters

- There are actually a number of states associated with each session
- Once session is established, there is a current operating state for both read and write
- In addition, during the Handshake Protocol, pending read and write states are created
- Upon successful conclusion of the Handshake Protocol, the pending states become the current states

Session state parameters

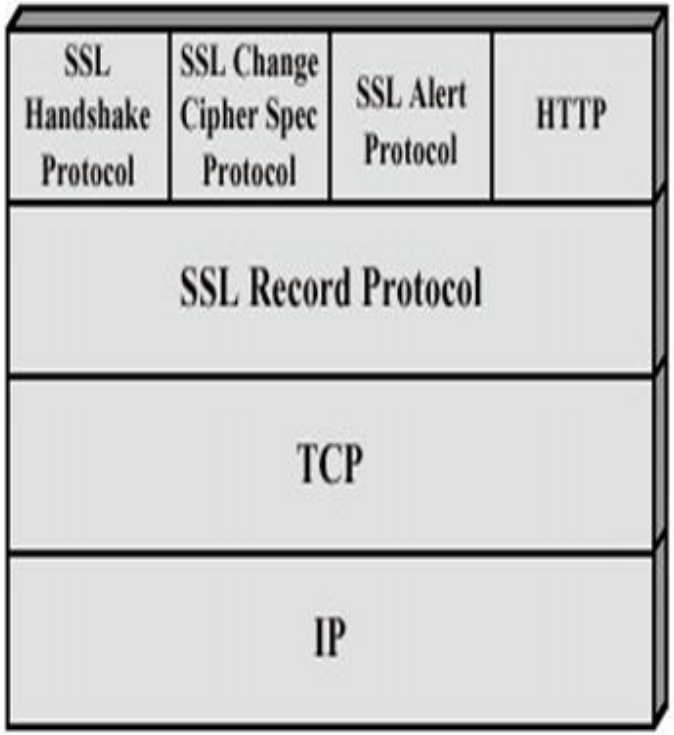
- Session Identifier (server to identify an active or resumable session state)
- Peer Certificate (X509.v3 certificate, may be null)
- Compression Method (algorithm used to compress data prior to encryption)
- Cipher spec (Specifies the bulk data encryption algorithm, AES,MD5,SHA-1, hash size)
- Master secret (48 byte secret shared by client and server)
- Is resumable (flag indicates , session can be used to initiate new connections.)

Architecture(4)-States parameters

Connection state parameters

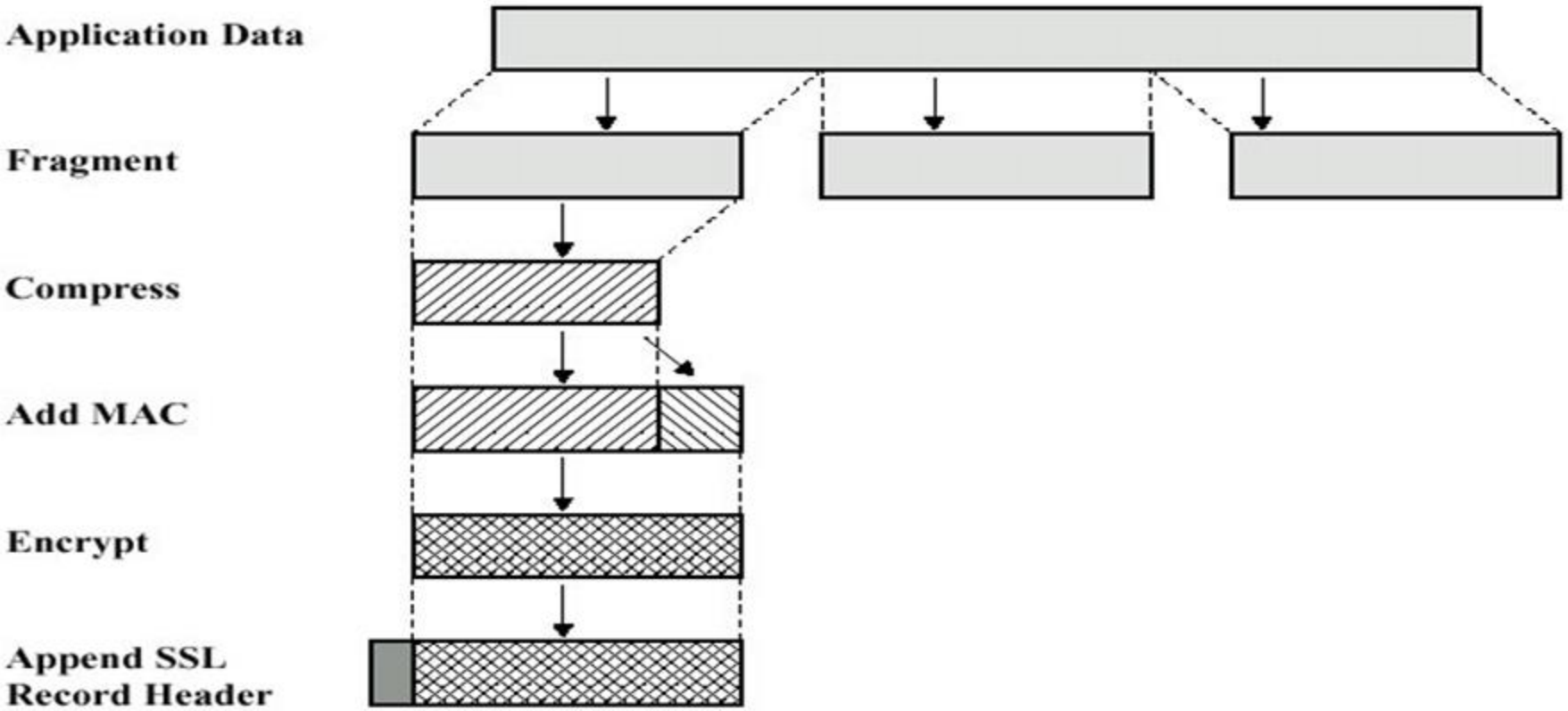
- Server and client random (byte sequence chosen by server and client for each connections)
- Server write MAC secret (secret key used MAC operations sent by the server)
- Client write MAC secret(The symmetric key used in MAC operations on data sent by the client.)
- Server write key (conventional encry . Key by server and decry. by client)
- Client write key (encry. Client and decry. Server) - Initialization vectors (IV) – (This field is first initialized by
- the TLS Handshake Protocol)
- Sequence numbers (Each party maintain a sequence number - each connection, Sequence numbers may not exceed 264 - 1)

SSL Record Protocol(1)



- Two services for SSL connections provided by SSL Record protocol
  - Confidentiality : The Handshake Protocol defines a shared secret key that is used for conventional encryption of TLS payloads.
  - Message Integrity : MAC
  - Handshake protocol defines both shared secret keys those are used for conventional encryption of SSL payloads and are used to form a message authentication code.

SSL Record Protocol(2)



- The Record Protocol takes an application message to be transmitted, fragments the data into manageable blocks, optionally compresses the data, applies a MAC, encrypts, adds a header, and transmits the resulting unit in a TCP segment
- Received data are decrypted, verified, decompressed, and reassembled before being delivered to higher-level users.

SSL Record Protocol(3)

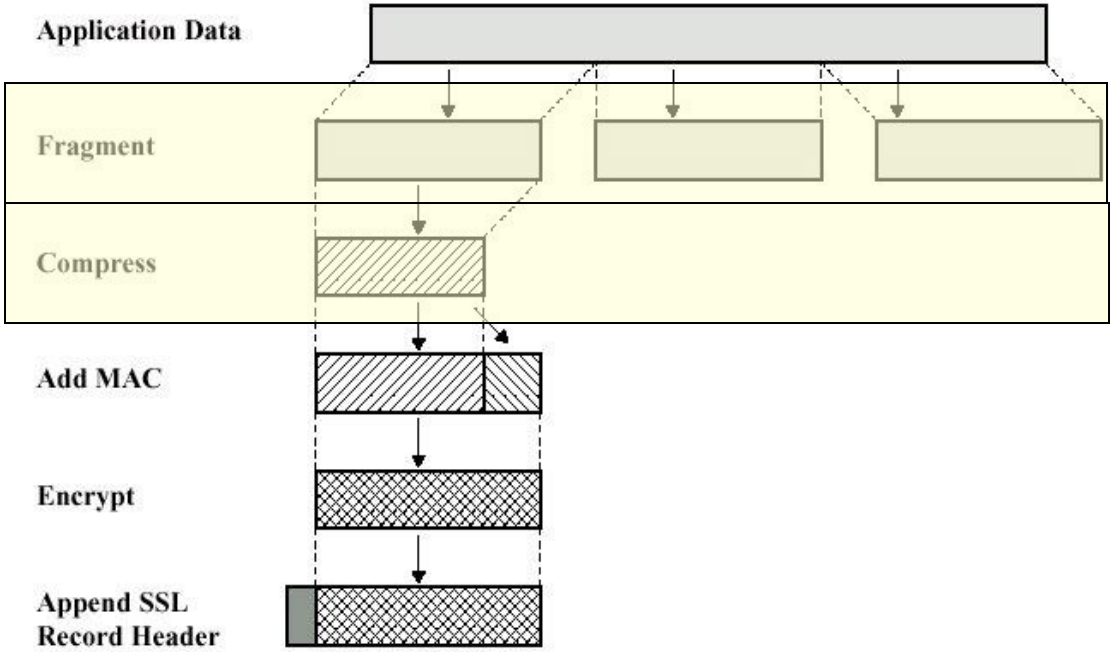


Figure 14.3 SSL Record Protocol Operation

- 1st : Fragmentation
  - 2^{14} bytes(16384 bytes) or less
- 2nd : Compression
  - Optionally applied
  - Must be losses
  - May not increase the content length by more than 1024bytes
  - In SSLv3, no compression algorithm is specified, so the default compression algorithm is null

SSL Record Protocol(4)

- 3rd : Add MAC
  - A shared secret key used

    hash(MAC_write_secret||pad_2||
    hash(MAC_write_secret||pad_1||seq_num||SSLCompressed.type||
    SSLCompressed.length ||SSLCompressed.fragment))
  - Where

    || = concatenation

    MAC_write_secret = shared secret key

    hash = cryptographic hash algo; MD5 or SHA-1

    pad_1 = the byte 0x36(0011 0110) repeated 48times(384 bits) for MD5 and 40times(320bits) for SHA-1

    pad_2= the byte 0x5c(0101 1100) repeated 48times for MD5 and 40times for SHA-1

    seq_num = the sequence number for this message

    SSLCompressed.type = the higher-level protocol used to process this fragment

    SSLCompressed.length = the length of the compressed fragment

    SSLCompressed.fragment = the compressed fragment (if compression is not used, the plaintext fragment)

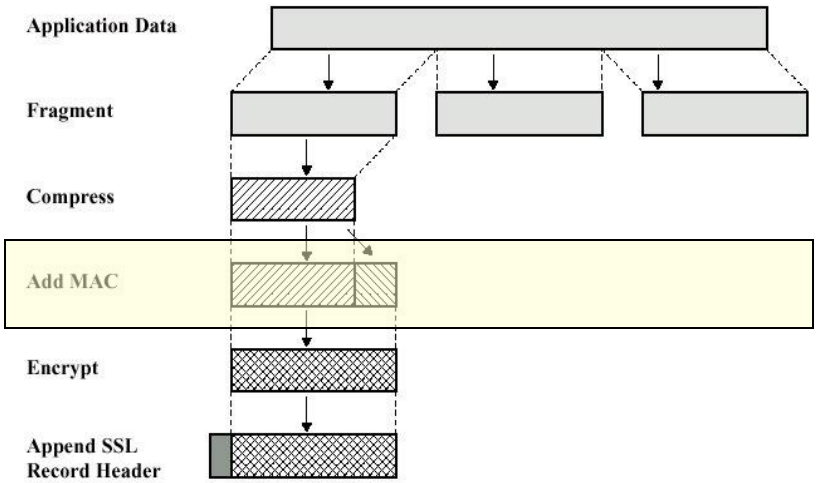


Figure 14.3 SSL Record Protocol Operation

SLO – 1 & SLO -2 :
Encoding, Encrypted Record

SSL Record Protocol(5)

- **4th : Encryption**
  - **Symmetric encryption**
  - **Algorithm used**
    - » **Stream cipher**
      - **The compressed message plus the MAC are encrypted**
      - **RC4-40, RC4-128**
    - » **Block cipher**
      - **Padding may be added**
      - **IDEA, RC2-40, DES-40, DES, 3DES,Fortezza**
      - **The total amount of padding is the smallest amount such that the total size of the data to be encrypted is a multiple of the cipher's block length**

        ex) Plain text : 58 bytes with a MAC of 20 bytes that is encrypted using a block length of 8 bytes = padding.length byte(1) + 1 byte padding

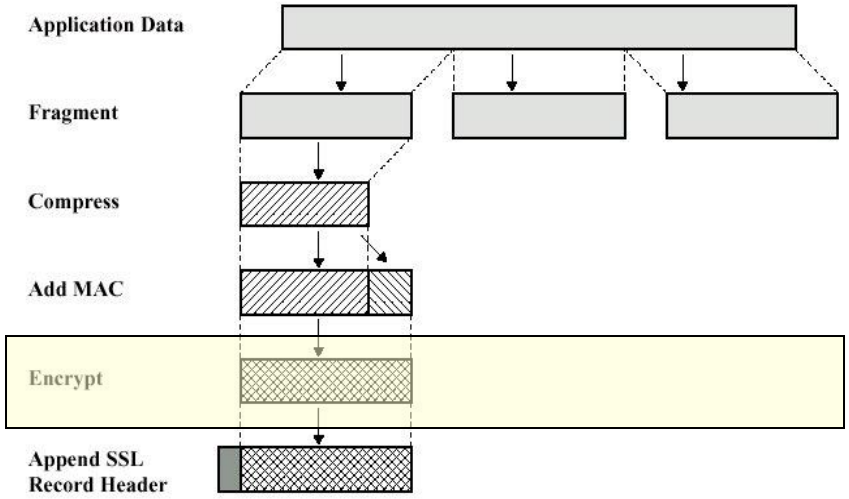


Figure 14.3 SSL Record Protocol Operation

SSL Record Protocol(6)

- **5th : Append SSL record header**
  - Content type(8bits)
    - » change_cipher_spec
    - » alert
    - » handshake
    - » application_data
  - Major version(8) – SSLv3
  - Minor version(8) - SSLv3 value 0
  - Compressed length(16) :
    length in bytes less than 2^{14} + 2048

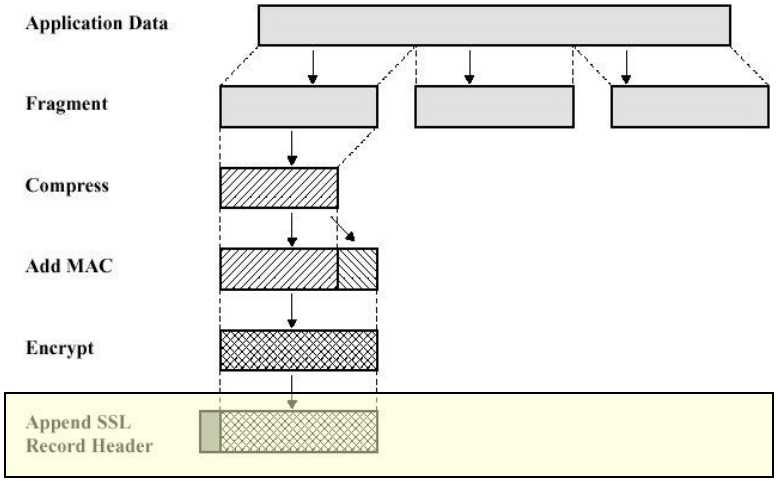


Figure 14.3 SSL Record Protocol Operation

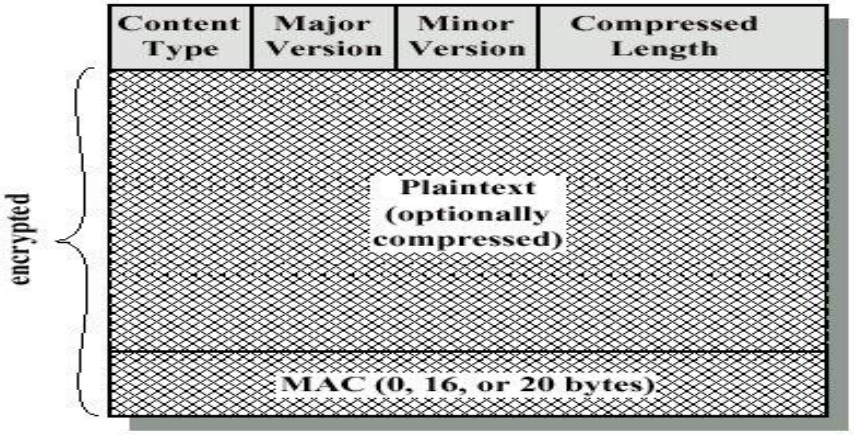
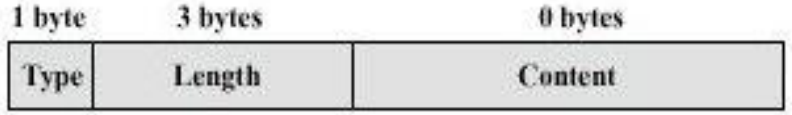


Figure 14.4 SSL Record Format

SLO - 1 :
Handshake Protocol

Handshake Protocol(1)

- The most complex part of SSL
- Allows sever and client
  - To authenticate each other
  - To negotiate an encryption and MAC algorithm and cryptographic keys to be used to protect data sent in an SSL record
- Used before any application data is transmitted
- Consists of a series of messages exchanged by client and server



(c) Handshake Protocol

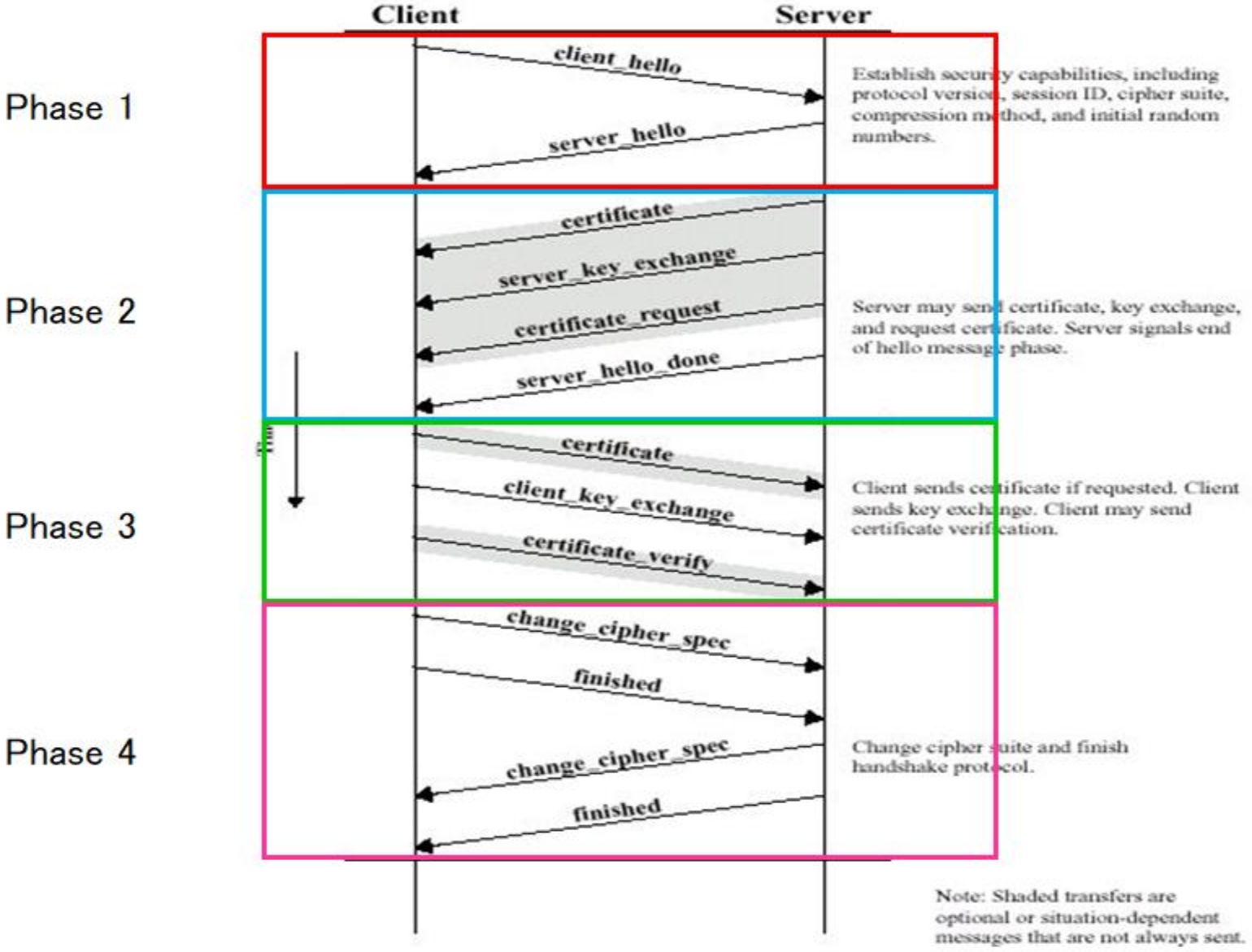
- Three fields
  - Type(1byte) : one of 10 messages
  - Length(3byte) – length of message in byte
  - Content(>=1 byte) : parameters associated with this message

Handshake Protocol(2)

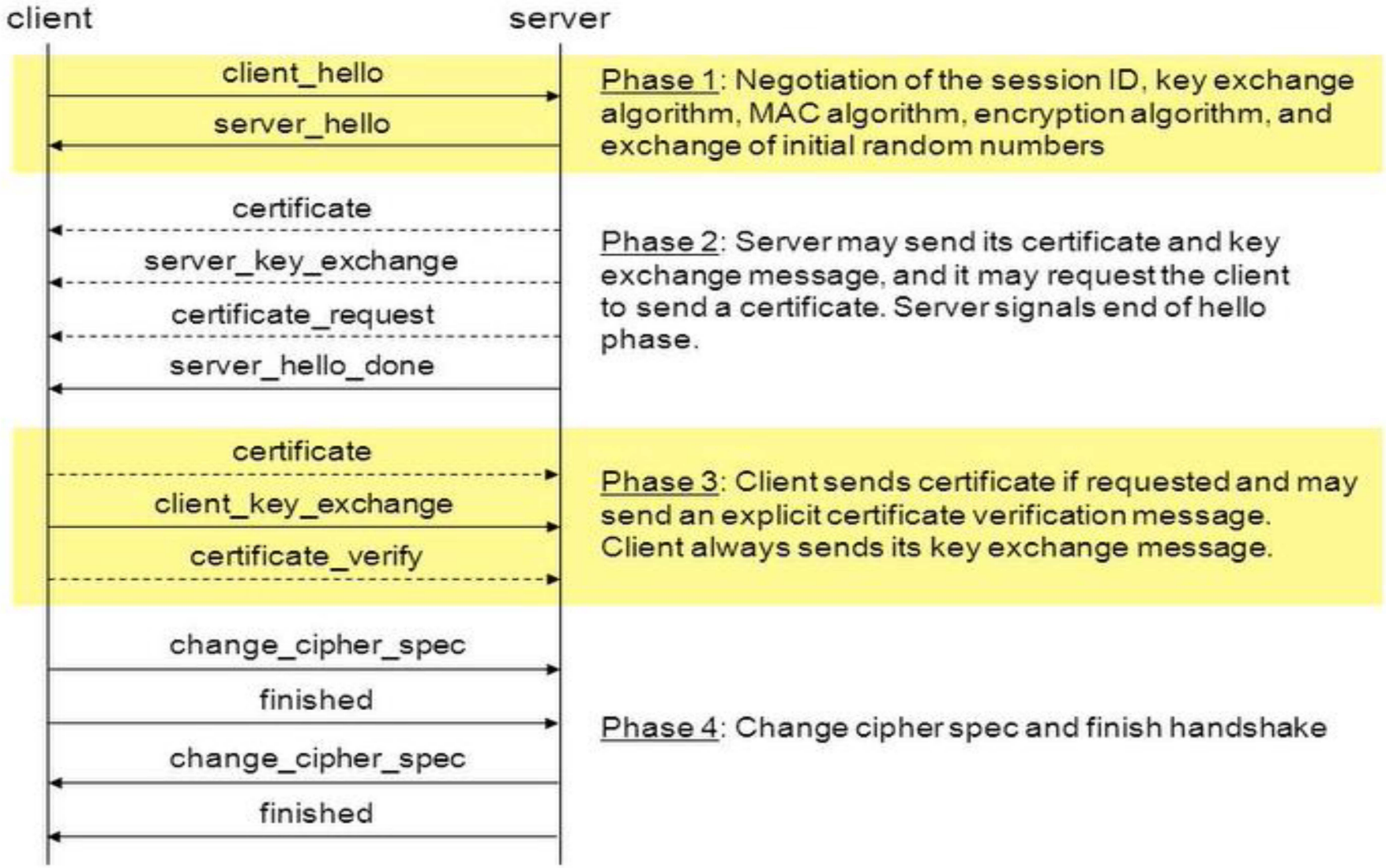
Table SSL Handshake Protocol Message Types

Message Type	Parameters
hello_request	null
client_hello	version, random, session id, cipher suite, compression method
server_hello	version, random, session id, cipher suite, compression method
certificate	chain of X.509v3 certificates
server_key_exchange	parameters, signature
certificate_request	type, authorities
server_done	null
certificate_verify	signature
client_key_exchange	parameters, signature
finished	hash value

Handshake Protocol ACTION



SSL Handshake Protocol – overview



Handshake Protocol(3)

- **Phase 1 –Establish Security Capabilities**
  - To initiate a logical connection and to establish the security capabilities that will be associated with it
  - The exchange is initiated by the client, which sends a "client_hello" message
    - Version :The highest SSL version
    - Random : 32bit timestamp & 28bytes Secure Random Number
    - SessionID : Variable-length session identifier
    - Cipher Suite : A list that contains the combinations of cryptographic algorithms supported by client, in decreasing order of preference
    - Compression Method : The list of compression methods supported by client
  - Server sends "server_hello" message with the same parameters
    - Version : The lower of the version suggested by the client and the highest supported by the server
    - Random : Generated by server
    - SessionID : If client is non-zero then,the same with the client. Otherwise, the value for new session
    - Cipher Suite : Single cipher suite selected by the server
    - Compression Method : The one supported by server

Handshake Protocol(4)

- Phase 1 –(continues)
  - Cipher Suite
    - » Key exchange methods supported
      - RSA : The secret key is encrypted with the receiver's public key. A public key certificate for the receiver's key should be available
      - Fixed Diffie-Hellman (DH): Server's certificate contains DH public parameters signed by CA. Client provides its DH public key parameters either in certificate or in a key exchange message
      - Ephemeral DH : DH public keys are exchanged, signed using sender's private RSA or DSS key. The receiver can use the corresponding public key to verify the signature
      - Anonymous DH : The base DH algo is used with no authentication. Vulnerable to man-in-the-middle attack
      - Fortezza
    - » CipherSpec
      - CipherAlgorithm : RC4, RC2, DES, 3DES, DES40, IDEA, Fortezza
      - MACAlgorithm : MD5, SHA-1
      - CipherType : Stream or block
      - IsExportable : True or false
      - HashSize : 0, 16(for MD5), or 20(for SHA-1) bytes
      - Key Material : A sequence of bytes used in generating the write key
      - IV Size : The size of IV for CBC

Handshake Protocol(5)

- **Phase 2 –Server Authentication and Key Exchange**
  - **The server begins by sending its certificate**
  - **Certificate(X.509) : except anonymous Diffie-Hellman**
  - **Server_key_exchange()**
    - » **Anonymous DH : prime + primitive root**
    - » **Ephemeral DH : prime + primitive root + signature**
    - » **RSA key exchange, in which the server is using RSA but has a signature-only RSA key: temporary RSA public key + signature**
    - » **Fixed DH or RSA key exchange : No need**
    - » **Signature is created by taking the hash of a message and encrypting it with the sender's private key**
      - **hash(ClientHello.random||ServerHello.random||ServerParams)**
  - **Certificate request : Non-anonymous server(server not using anonymous DH) can request certificate from the client**
    - » **Certificate_type : includes public key algorithm and its use**
    - » **Certificate_authorities : a list of the distinguished names of acceptable certificate authorities**
  - **Server_hello_done (no parameter)**

Phase : 3

Client authentication and key exchange

- certificate
  - sent only if requested by the server
  - may contain
    - public RSA or DSS key suitable for signing only, or
    - fix DH parameters
- client_key_exchange
  - always sent (but it is empty if the key exchange method is fix DH)
  - may contain
    - RSA encrypted pre-master secret, or
    - client one-time public DH value, or
    - Fortezza key exchange parameters
- certificate_verify
  - sent only if the client sent a certificate
  - provides client authentication
  - contains signed hash of all the previous handshake messages
    - if DSS: SHA-1 hash is signed
    - if RSA: MD5 and SHA-1 hash is concatenated and encrypted with the private key

MD5(master_secret | pad_2 | MD5(handshake_messages | master_secret | pad_1))

SHA(master_secret | pad_2 | SHA(handshake_messages | master_secret | pad_1))

Phase : 4

Finished messages

- finished
  - sent immediately after the change_cipher_spec message
  - first message that uses the newly negotiated algorithms, keys, IVs, etc.
  - used to verify that the key exchange and authentication was successful
  - contains the MD5 and SHA-1 hash of all the previous handshake messages:

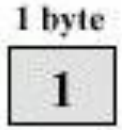
    MD5(master_secret | pad_2 | MD5(handshake_messages | sender | master_secret | pad_1)) |
    SHA(master_secret | pad_2 | SHA(handshake_messages | sender | master_secret | pad_1))

    where "sender" is a code that identifies that the sender is the client or the server (client: 0x434C4E54; server: 0x53525652)

SLO - 2 :
Change Cipher Spec & Alert

Change Cipher Spec & Alert Protocol(1)

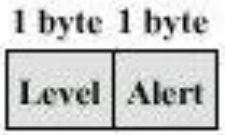
- **Change Cipher Spec Protocol**
  - Simplest protocol
  - Consists of a single message, which consists of a single byte with the value 1
  - To cause the pending state to be copied into the current state, which updates the cipher suite to be used on this connection



(a) Change Cipher Spec Protocol

- **Alert Protocol**
  - Used to convey SSL-related alerts to the peer entity
  - Consists of two bytes
    - » Level : conveys the severity of the message
      - First Byte takes the value : Warning(1) or Fatal(2) – immed. terminates the connection, other connection may continue, but no new connection may be established.

    Second Byte : Alert : a code that indicates the specific alert



(b) Alert Protocol

Alert Protocol(2)

- **Codes for alerts**
  - Alerts that are always fatal
    - unexpected_message : An in appropriate message was received
    - bad_record_mac : An incorrect MAC was received
    - decompression_failure : The decompression function received improper input
    - handshake_failure : Sender was unable to negotiate an acceptable set of security parameters given the options available
    - illegal_parameter : A field in a handshake message was out of range or inconsistent with other fields
  - The remainder of alerts
    - close_notify :Notifies the recipient that the sender will not send any more messages on this connection
    - no_certificate : May be sent in response to a certificate request if no appropriate certificate is available
    - bad_certificate : A received certificate was corrupt
    - unsupported_certificate :The type of the received certificate is not supported
    - certificate_revoked : revoked by its signer
    - certificate_expired : has expired
    - certificate_unknown : Some other unspecified issue arose in processing the certificate, rendering it unacceptable

SLO - 1 & 2 :
Computing the Keys

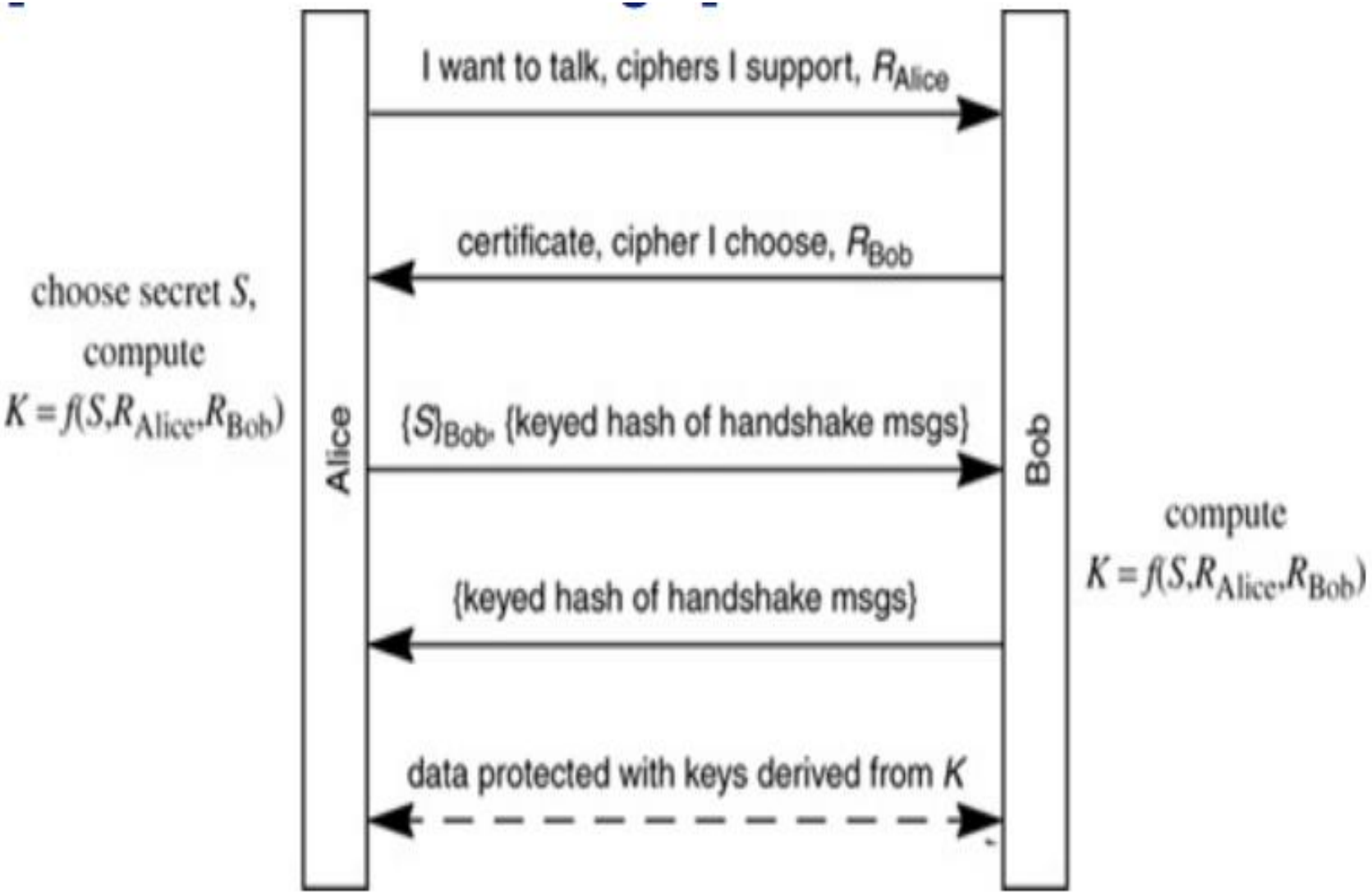
Computing the Keys

- The secret S sent in the first exchange is the pre master secret

  (Alice chooses a random number S (known as the pre-master secret) and sends it, encrypted with Bob's public key)

  She also sends a hash of the master secret K and the hand-shake messages, both to prove she knows the key and to ensure that tampering of the handshake messages would be detected)

The notation $f(K_{Alice\text{-}Bob}, R)$ means that R is cryptographically transformed, somehow, with Alice and Bob's shared secret $K_{Alice\text{-}Bob}$.


I want to talk, ciphers I support, R_{Alice}
certificate, cipher I choose, R_{Bob}
choose secret S,
compute
$K = f(S, R_{Alice}, R_{Bob})$
Alice
$\{S\}_{Bob}$, {keyed hash of handshake msgs}
Bob
compute
$K = f(S, R_{Alice}, R_{Bob})$
{keyed hash of handshake msgs}
data protected with keys derived from K

Computing the Keys…

- It is shuffled with the two Rs to produce the master secret K. In other words $K = f(S, R_{Alice}, R_{Bob})$.
- For each connection (including the first) the master secret is shuffled with the two Rs to produce the six keys used for that connection (for each side: encryption, integrity, and IV), i.e., each of the keys is $g_i(K, R_{Alice}, R_{Bob})$.
- Note that this is unnecessarily complex in the case of using RSA, since on the first connection, the Rs get shuffled in twice.

Computing the Keys...

- It would have been fine with RSA to just use S the way K is used; for each connection, combine S with that connection's Rs to produce the six keys used for that connection.
- However, there are other authentication methods that would wind up always computing the same pre-master secret S.
- For instance, with Diffie-Hellman using a fixed Diffie-Hellman number as your public key, the same two parties (Alice and Bob) will always compute the same S.

Computing the Keys...

- If S were kept around in memory, it's possible that malicious software might steal it.
- So it's safer to hash it with some nonces to produce K (the master secret), since if K is stolen, it will only affect communication between Alice and Bob during the single session.
- The keyed hashes are keyed with K, and are sent protected with the data keys (i.e., encrypted using the encryption key and IV, and integrity-protected with the integrity key).

Computing the Key

- The Rs are 32 octets long, and it is recommended (but not enforced) that the first 4 octets not be randomly chosen, but instead be the Unix time (seconds since January 1, 1970) when the message was generated.
- This ensures (assuming that at least a second has elapsed) that Alice and Bob will choose different 32-octet Rs each time a session under the same master secret is resumed, even if their random number generators are really bad or they're really unlucky.
- If the same Rs were chosen with the same master secret, an attacker might be able to successfully replay packets.

SLO - 1 & 2 :
Client Authentication

Client Authentication

- As deployed today it is unusual for clients to have certificates.
- However, it is possible in SSL/TLS for the server (Bob) to request that the client (Alice) authenticate herself.
- This is done by having Bob send a "certificate request" in message 2.
- Alice, upon seeing the request, sends her certificate, and her signature on a hash of the handshake messages, proving she knows the private key associated with the public key in the certificate.

SLO-1 & SLO-2 :
Public Key Infrastructure (PKI)

Overview

- Introduction
- Building Blocks
- Certificates
- Organization
- Conclusions

Public Key Infrastructure (PKI)

- PKI provides assurance of public key. It provides the identification of public keys and their distribution.

PKI comprises of the following components.

- Public Key Certificate, commonly referred to as 'digital certificate'.
- Private Key tokens.
- Certification Authority.
- Registration Authority.
- Certificate Management System.

Introduction

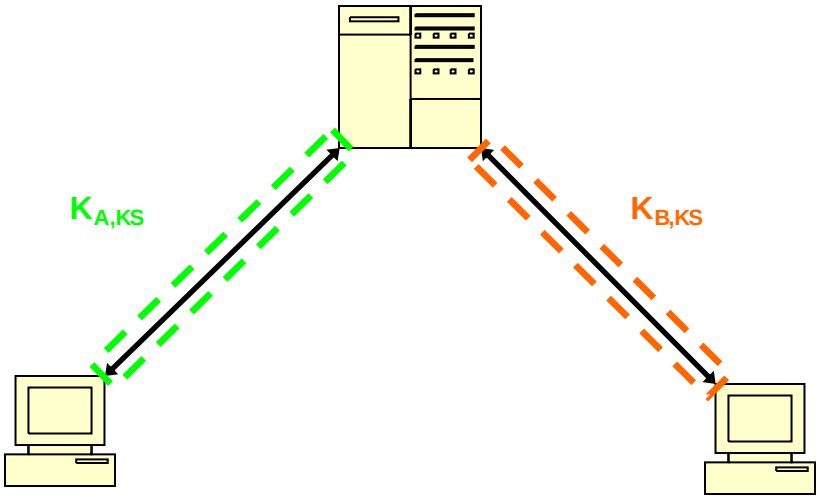
In the beginning there were shared secret keys

- Early cryptographic systems had to use the same key for encryption and decryption
- To establish an encrypted channel both users needed to find out this key in some secure fashion
  - **Limited –** Users could meet and exchange the key
  - **Flexible –** Users could use a key server

Introduction
Key Exchange – User to User

- This exchange eliminates a communication channel that could be attacked
- Limited - Users must meet all other users
- Users must recognize each other or show proper identification

Introduction
Key Exchange – Key Server



- Each user has set to up a key with the Key Server
- Key Server creates and transmits secure session keys to users
- Flexible – Users need only have a prior established key with the Key Server
  - For a system with n users only (n) meetings must occur
- Key Server takes care of the initial validation of user's identities

Building Blocks

- Cryptographic tools
- Putting them together
- Names
- Time
- A secure communication session

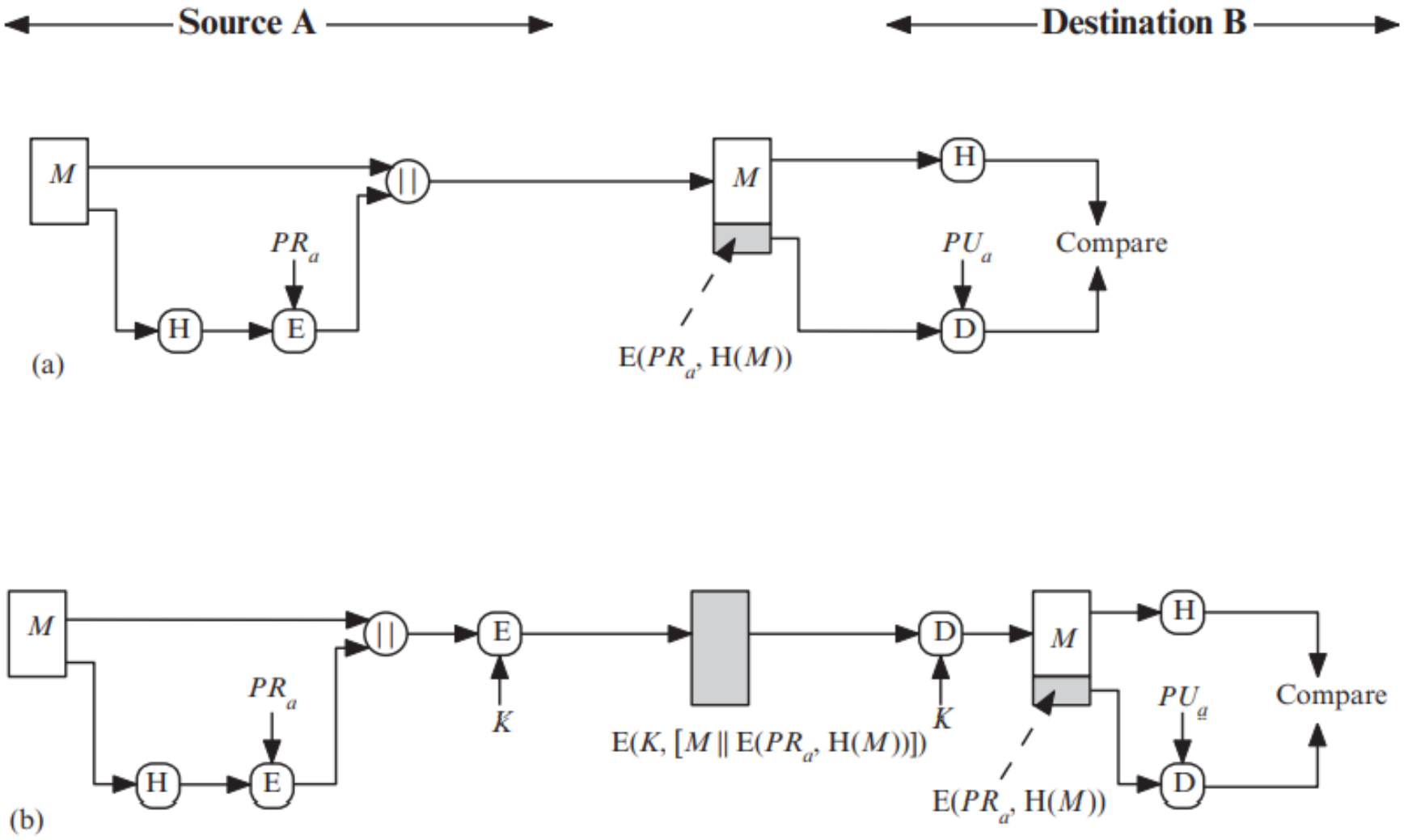
Building Blocks
Cryptographic Tools

- **Symmetric Key Cryptography**
  - Encryption: $C = SE_K(M)$
  - Decryption: $M = SD_K(C)$
  - Secure as long as only communicating users know K
  - Having K lets one read C
  - Fast to calculate
- **Public Key Cryptography**
  - Encryption: $C = PE_{K+}(M)$
  - Decryption: $M = PD_{K-}(C)$
  - Secure as long K- is only known by the receiver
  - Having K- lets one read C, but having K+ does not
  - Slow to calculate

Building Blocks
Cryptographic Tools

- **Digital Signatures**
  - Sign: $PE_{K-}(H(M)) = S$
  - Verify: $PD_{K+}(S) = H(M)$
  - Reliable as long as only the signer knows K-
  - Having K- allows one to sign, having K+ only allows one to verify the signature
  - Slow to calculate
  - K's + and - could just be a user's public and private keys

DIGITAL SIGNATURE



Simplified Examples of Digital Signatures

Building Blocks
Putting Them Together

- Symmetric cryptography is used for majority of communications
- Public Key cryptography is used for exchanging Symmetric keys
- Digital Signatures are used to validate Public Keys

Building Blocks
Names

- A name in PKI must be unique to a user
- Assigning these names presents similar difficulties as found in other areas of Distributed Systems
- Without proper and well thought out naming PKI is pretty much useless

Building Blocks
Time

- A PKI must know the current time
- Much of a PKI's security relies on having an accurate clock
- For the most part, time does not need to be known extremely reliably and being off by a minute will usually not be an issue

Building Blocks
A Secure Communications Session

- Alice and Bob wish to set up a secure communications channel
- They use Public Key Cryptography to exchange a Symmetric key
  - Alice: Private PK = $K-_A$, Public PK = $K+_A$
  - Bob: Private PK = $K-_B$, Public PK = $K+_B$
  - Time T and random Symmetric Key K_S
  - Simplified example:

  1: Alice -> Bob: $PE_{K+B}(Alice, T, K+_A, PE_{K-A}(T, K_S))$

  2: Bob -> Alice: $PE_{K+A}(T, K_S)$

  3: Alice <-> Bob: $SE_{KS}(M_i)$

Certificates

- What they are
- How they are issued
- How they are distributed
- How they are revoked

Certificates
What they are

- The issue with building a secure session is that it assumes that both Alice and Bob know each others public keys
- We need some way for them to learn this besides meeting each other
- a similar strategy to the Key Server but can we do better?

This is where Certificates come in…

Certificates
What they are

- A Certificate is a combination of a user's public key, unique name, Certificate start and expiration dates, and possibly other information
- This Certificate is then digitally signed, by some Trusted 3rd Party, with the signature being attached to the rest of the Certificate
- This Signed Certificate is commonly referred to as just the user's Certificate
- The Certificate for a user Bob, signed by signer Tim, in essence states

  "I Tim certify that this Public Key belongs to Bob"

Certificates
How they are issued

- The users of a PKI must place their trust in a 3rd Party to carefully verify a user's identity before signing his or her public key
- Each user generates their own Public-Private Key pair and Certificate
- A user then verifies them self to the 3rd Party and shows his or her Certificate's content. At this point the third party will sign the Certificate.

Certificates
How they are distributed

- Users are free to distribute their signed Certificates over any medium, public or private, without concern
- Other users may acquire this Certificate from any source and check the 3rd Party's signature for tampering
- If the signature is good then the other users know that the 3rd Party affirms that the Certificate belongs to the user who is listed in the Certificate

Certificates
How they are Revoked

- Periodically Certificates may become compromised, requiring a Certificate Revocation
- A Certificate Revocation message is simply a message signed by $K-_i$ (the private version of the Certificate's $K+_i$) saying that the Certificate is revoked
- A PKI will have a database of revoked Certificates (a Certificate Revocation List, CRL) that users may access periodically for the latest list of revoked Certificates
- An alternative to certificate revoking is to set the expiration time to very shortly after the issue time.
- Thus every key in this system is revoked so rapidly that we do not need to worry what may happen to the compromised key

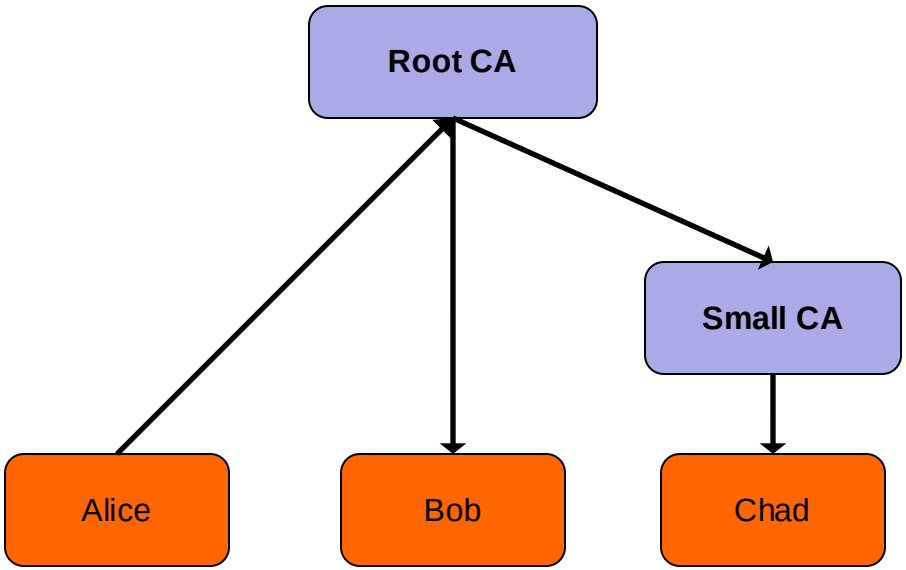
Organization

- What is “Trust”?
- How do we organize a PKI to disseminate trust?

Organization Trust

- Trust is based on real world contractual obligations between a 3rd Party and users [2]
- This Trusted 3rd Party is referred to as a Certificate Authority (CA)
- In other models trust is based on personal relationships that don't have a contractual basis (e.g. PGP)
- Users may allow a CA to delegate their trust
- This delegation of trust is what allows us to build large PKI's

Organization Trust

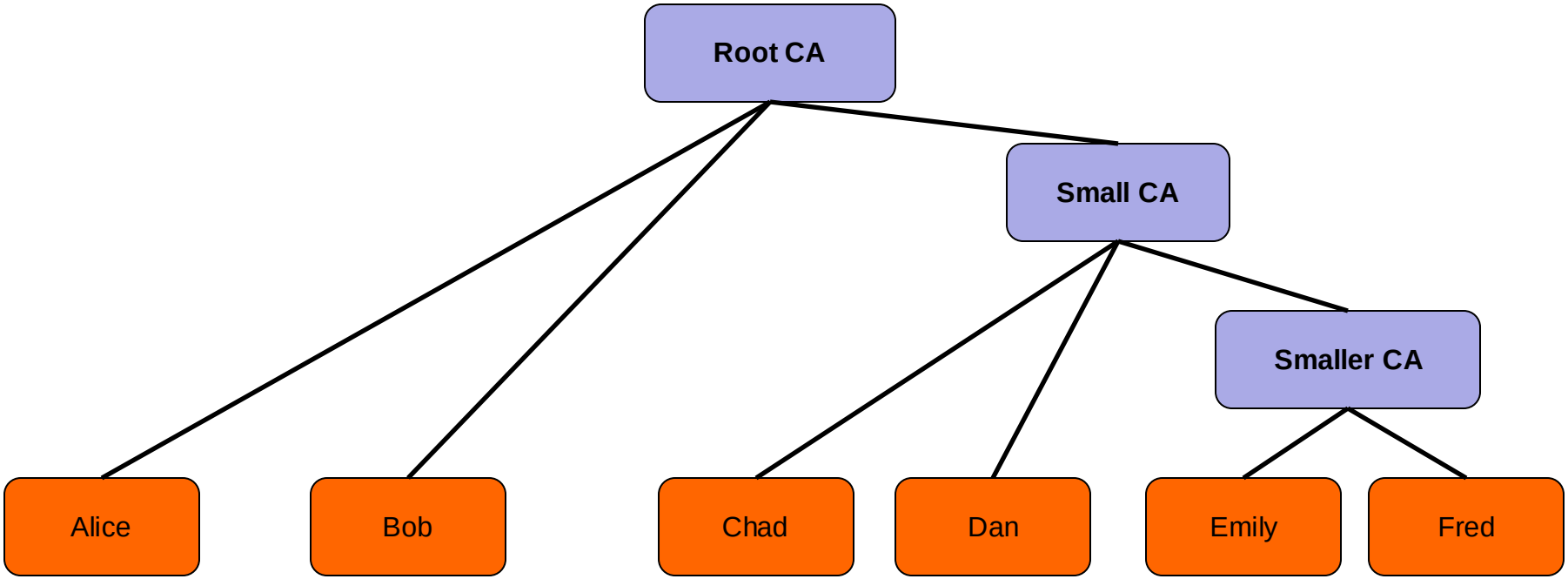


- If Alice trusts Root CA then she trusts Bob's Certificate signed by Root CA
- If Alice trusts Root CA to delegate her trust to others then she trusts Chad's Certificate signed by Small CA

Organization
Organizing a PKI

- A PKI may be organized based on a variety of models using delegation of trust
  - Strict Hierarchy
  - Networked
  - Web Browser
  - PGP

Organization
Strict Hierarchy

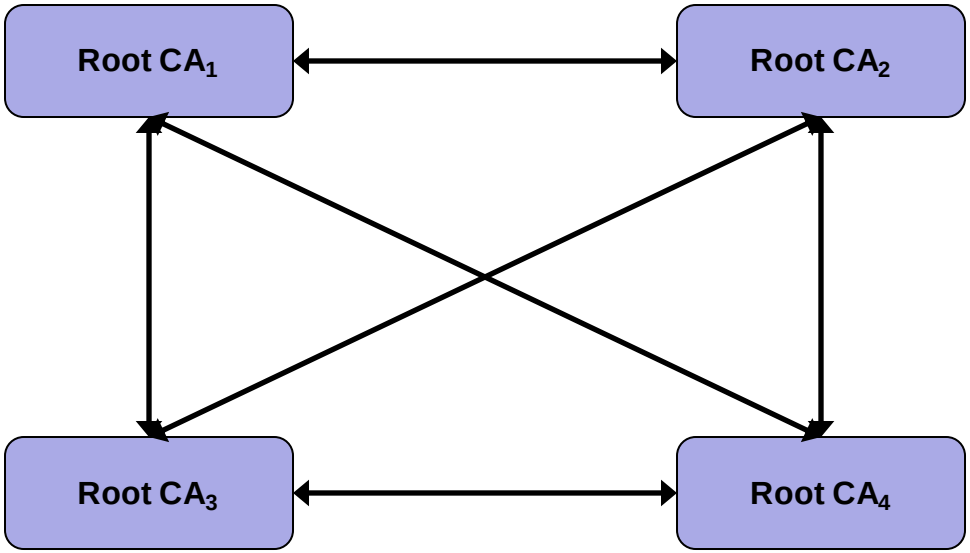


- All users trust Root CA
- Root CA may delegate that trust to other CA's who in turn may be allowed to delegate that trust
- In this way a PKI may grow without all the burden being placed on Root CA

Organization Networked

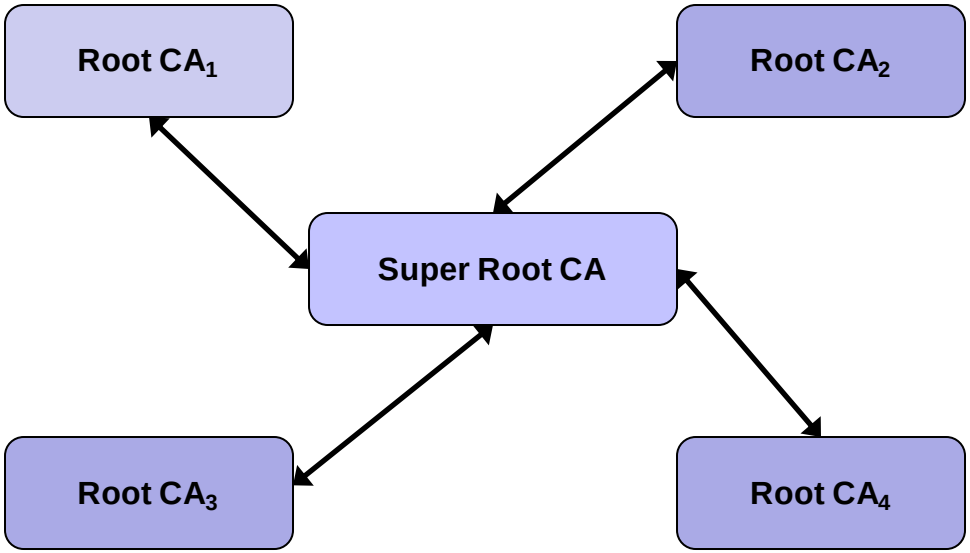
- The Networked model addresses what to do when two or more PKIs wish to join together or merge
- Two techniques
  - Mesh
  - Hub-and-Spoke
- We only need the Root CAs of each PKI to participate in this model

Organization
Networked – Mesh



- Every Root CA signs every other Root CA's Certificate
- Hard to join a large numbers of CAs

Organization
Networked – Hub-and-Spoke



- The Root CAs come together to create the Super Root CA
- Each Root CA signs the Super Root CA's certificate while the Super Root CA signs each of theirs
- Easier to join large numbers of CAs
- Question becomes, Who gets to manage the Super Root CA?

Conclusions

- A PKI allows us to take the concept of a Key Server and apply it to Public Keys
- It allows greater flexibility then a Key Server in that users do not need to communicate with the Root CA every time a Session Key is needed
- There are a vast variety of models for disseminating trust in a PKI
- Even though PKIs look like an amazing idea, in practice there are numerous problems implementing them on a large scale
  - Who does everyone trust?
  - What format do people use?
  - Security of the multitude of programs that rely on PKIs

SLO-1 & SLO-2 :
Attacks Fixed in v3

Downgrade Attack

- In SSLv2, there is no integrity protection for the initial handshake, so an active attacker can remove the cipher suites with strong encryption from the list of requested cipher suites, causing Alice and Bob to agree upon a weaker cipher.
- In SSLv3 this was fixed by adding a finished message to the end of the initial handshake in which each side sends a digest of the messages in the handshake.

Truncation Attack

- SSLv3 added a finished message to indicate that there is no more data to send.
- V2 depended on the TCP connection closing.
- But the TCP connection close is not cryptographically protected, so an attacker could close the connection by sending a TCP close message, and the other side would not be able to know that the session was abnormally terminated.

SLO-1 & SLO-2 :
Exportability

Exportability

- The simplest thing to have done with export controls was to always use weak crypto, even domestically.
- But it was legal to deploy strong crypto domestically, so complex mechanisms were built in order to make something that could be as secure as possible within the political constraints, and have the domestic and exportable versions interoperate.

Exportability in SSLv2

- SSLv2 domestic clients supported 128-bit encryption.
- But the U.S. government limited exportable cryptographic keys to 40 bits.
- Some "exportable" suites were defined for SSLv2 which effectively used a 40-bit encryption key even though the actual cryptographic algorithm used a 128-bit key.
- In these exportable suites, instead of sending a 128-bit secret encrypted with the server's public key, the client sends a 40-bit secret encrypted with the server's public key, and sends an additional 88 bits in the clear.

Exportability in SSLv2

- So in a non-exportable suite, there will be 128 secret bits.
- In an exportable suite, there will be 40 secret bits and 88 non-secret bits.
- So in either case there are 128 bits, which are known in SSLv2 as the **client master key** (not a terrific name for a quantity in which most of the bits are not secret).
- This key is used to **produce the session keys**.

Exportability in SSLv2

- Exportable servers had 512-bit RSA keys.
- Domestic servers had large RSA keys (e.g., 1024 bits).
- If a domestic server were talking to an exportable client, the client would wind up sending its 40-bit secret in the domestic server's large RSA key, which was technically illegal but apparently nobody noticed until after the protocol was approved and shipped.
- At that point, they agreed to let SSLv2 continue to be exported, with clients that would encrypt a 40-bit key in a 1024-bit RSA key, though they warned the designers they would not approve the same approach in SSLv3.

Exportability in SSLv3

- In SSLv3 the pre-master secret sent by the client (and encrypted with the server's public key) is always a 48-octet number, but only 46 octets are random.
- The other two octets are used for version number.
- In the domestic version the pre-master secret is hashed with the Rs to produce the master secret, and the master secret is hashed with the Rs to produce the stream from which the six data protection keys are extracted.

Exportability in SSLv3

- The IVs are not secret.
- The export rules would have allowed strong IVs, but non-secret IVs are not a security vulnerability anyway.
- 40-bit encryption keys wasn't sufficient to meet export restrictions.
- It was also a crime to send the 40-bit secret encrypted with an RSA key larger than 512 bits.
- A domestic server would have a 1024-bit RSA key, and might be speaking to an exportable client.
- It wouldn't be legal for the client to send its pre-master secret encrypted with a 1024-bit RSA key.

Exportability in SSLv3

- SSLv3 solves this problem by having the server (Bob) generate at random a 512-bit RSA key.
- This key can be changed periodically and is signed by Bob using his long-term key. The 512-bit key is known as the **ephemeral key.**
- In SSLv2 the client chose the cipher suite, but in SSLv3, since the server chooses the cipher suite, Bob figures out from the client's list of cipher suites in message 1 (and his own cipher suite capabilities) if he needs to be speaking exportable crypto.
- So in message 2, if Bob chooses an exportable cipher suite, then, in addition to sending his regular certificate (the one certifying his long-term key), Bob sends his 512-bit ephemeral key, signed using his long term key.
- The client then sends the pre-master secret encrypted with Bob's ephemeral key.

SLO-1 & SLO-2 :
SET

Secure Electronic Transaction

- **SET**
  - Open encryption specification, credit card transactions
  - SETv1
    - » by MasterCard and Visa in February 1996.
  - A wide range of companies were involved
    - » IBM, Microsoft, Netscape, RSA, Terisa, and Verisign ….
  - First products are available in 1998.
  - is not a payment system.
  - is **security protocols**, formats.
- **SET provides**
  - A secure communication channel
  - Trust by the use of X.509v3 digital certificates
  - Ensures privacy
- **SET is defined in**
  - Book1 : Business Description (80 pages)
  - Book2 : Programmer's Guide (629 pages)
  - Book3 : Formal Protocol Definition (262 pages)

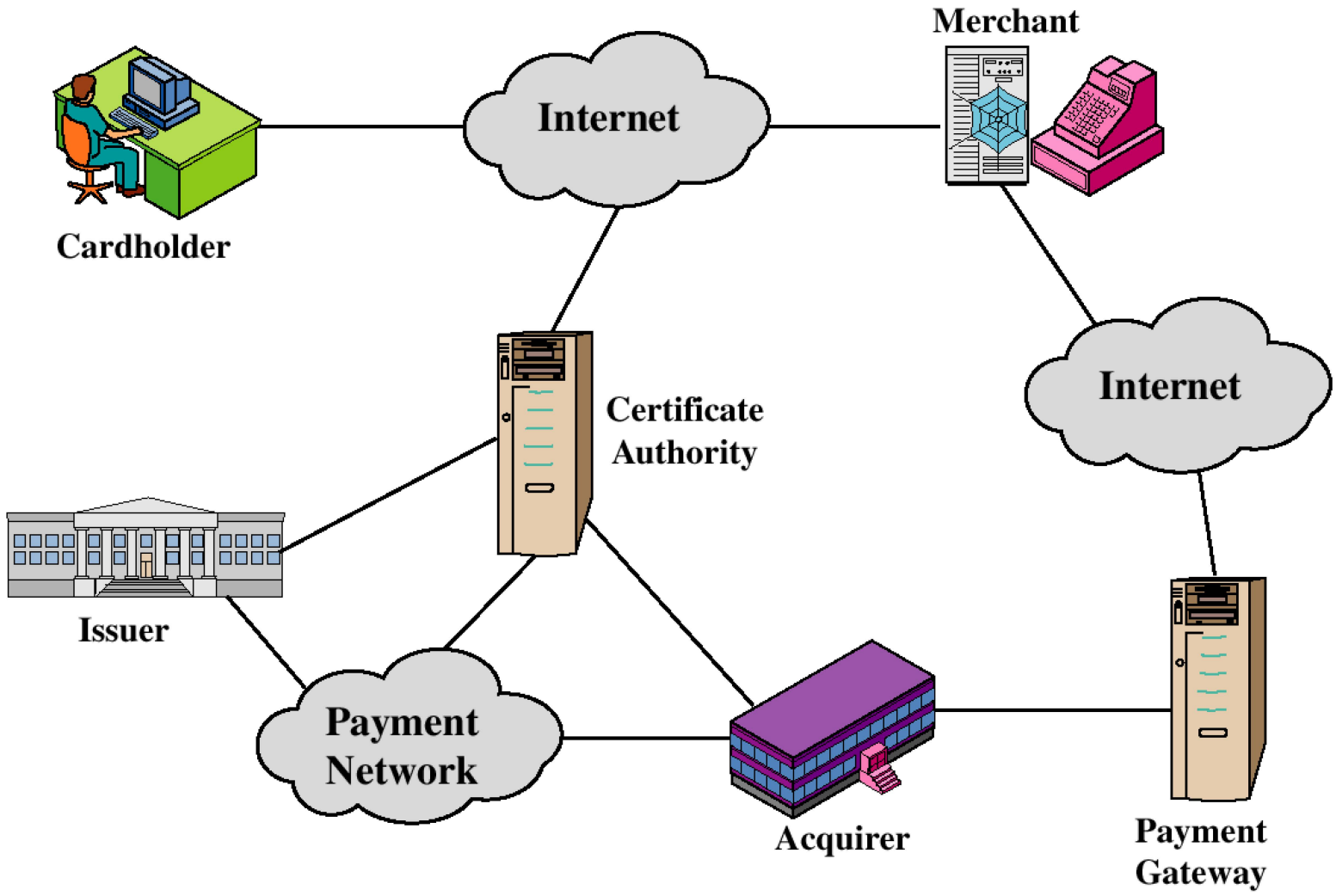
Business Requirements for SET

- Provide confidentiality of payment and ordering information.
- Ensure the integrity of all transmitted data.
- Provide authentication
  - a cardholder is a legitimate user of a credit card account.
  - a merchant can accept credit card transactions.
- Ensure the use of the best security practices and system design
- Create a protocol that neither depends on transport security mechanisms nor prevents their use.
- Facilitate and encourage interoperability among software and network providers.

Key Features of SET

- **Confidentiality of information**
  - Cardholder account and payment information is secured.
  - Merchant can not learn the cardholder's credit card number.
  - Using DES for confidentiality.
- **Integrity of data**
  - Type of data : Order information, personal data, payment instructions.
  - RSA digital signatures using SHA-1 or HMAC using SHA-1.
- **Cardholder account and Merchant authentication**
  - X.509v3 digital certificates with RSA signatures
- **Interoperability**
  - SET uses specific protocols and message formats.

SET Participants(1)



SET Participants(2)

- **Cardholder**
  - An authorized holder of a payment card (e.g., MasterCard, Visa).
- **Merchant**
  - A person or organization that has goods or services to sell.
- **Issuer (Financial institution ex. Bank)**
  - provides the cardholder with the payment card.
  - is responsible for the payment of the debt of the cardholder.
- **Acquirer (Financial institution - processes the payment)**
  - establishes an account with a merchant
  - Processing payment card authorizations and payments.
- **Payment Gateway**
  - Processing merchant payment messages.
  - Interfaces between SET and bankcard payment networks.
- **Certification Authority (CA)**
  - Issue X.509v3 public-key certificates.
  - The success of SET will depend on the existence of a CA infrastructure available.

Major Payment Gateways Supported in India

- PayUMoney.
- CCAvenue.
- EBS.
- ICICI Payseal.
- DirecPay.
- PayPal.

Sequence of events for transaction

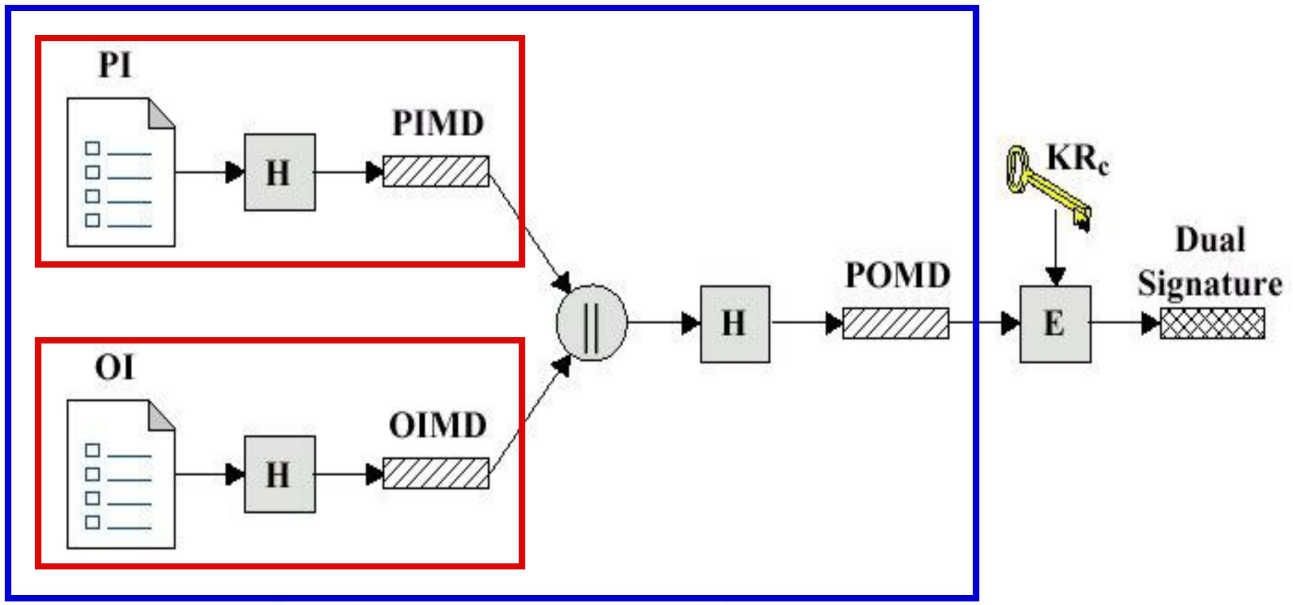
1. The customer opens an account.
2. The customer receives a certificate.
3. Merchants have their own certificates.
   - Two certificates : One for signing message, One for key exchange
4. The customer places an order.
   - Merchant returns an order form containing the list of items, their price, a total price, and an order number.
5. The merchant is verified.
   - The merchant sends a copy of its certificates. So customer verifies that a store is valid or not.
6. The order and payment are sent.
7. The merchant requests payment authorization.
   - Merchant sends the payment information to the payment gateway, requesting authorization.(customer's available credit is sufficient for this purchase)
8. The merchant confirms the order. (Confirmation of the order to the customer)
9. The merchant provides the goods or service.
10. The merchant requests payment.

Dual Signature(1)

- **Dual Signature**
  - An Important innovation introduced in SET.
  - Link two messages that are intended for two different recipients.
  - Customer -> Merchant : Order Information (OI)
    - » Merchant does not need to know the details of the customer's credit card number
  - Customer -> Bank : Payment Information (PI)
    - » the bank does not need to know the details of the customer's order
- The customer is afforded extra protection in terms of privacy by keeping these two items separate.
- The two items must be linked.
  - order information and payment information.

Dual Signature(2)

$$DS = E_{KR_c}[H(H(PI) \| H(OI))]$$



PI = Payment Information
OI = Order Information
H = Hash function (SHA-1)
|| = Concatenation

PIMD = PI message digest
OIMD = OI message digest
POMD = Payment Order message digest
E = Encryption (RSA)
KR_c = Customer's private signature key

Figure 14.9 Construction of Dual Signature

Dual Signature(3)

- **Customer makes dual signature.**
  - $DS = E_{KRc}[H(H(PI)||H(OI))]$
- **Merchant verifies the signature.**
  - Use DS, OI, PIMD, and customer's public key
  - Merchant computes and Compares two quantities;
    - » $H(PIMD||H(OI))$ and $DS = D_{KUc}[DS]$
- **Bank verifies the signature.**
  - Use DS, OIMD, PI, and customer's public key
  - Merchant computes and Compares two quantities;
    - » $H(H(PI)||H(OIMD))$ and $DS = D_{KUc}[DS]$

※ Customer has linked the OI and PI and can prove the linkage.

SET Transaction Types(1)

- **Cardholder registration**
  - Cardholders must register with a CA before they can send SET messages to merchants.
- **Merchant registration**
  - Merchants must register with a CA before they can exchange SET messages with customers and payment gateways.
- **Purchase request**
  - Message from customer to merchant containing OI for merchant and PI for bank.
- **Payment authorization**
  - Exchange between merchant and payment gateway to authorize a given amount for a purchase on a given credit card account.
- **Payment capture**
  - Allows the merchant to request payment from the payment gateway.

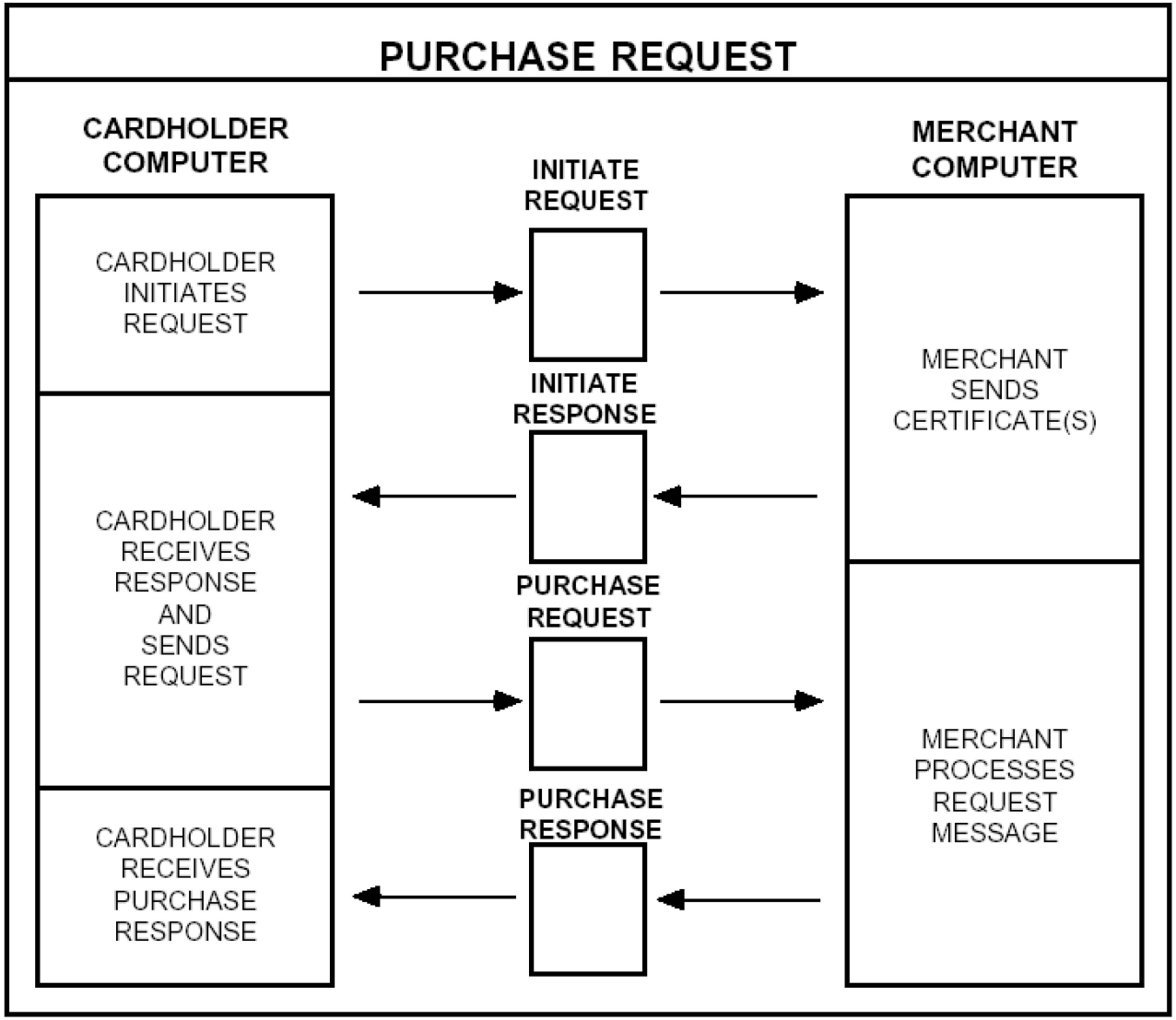
SET Transaction Types(2)

Certificate inquiry and status	If the CA is unable to complete the processing of a certificate request quickly, it will send a reply to the cardholder or merchant indicating that the requester should check back later. The cardholder or merchant sends the *Certificate Inquiry* message to determine the status of the certificate request and to receive the certificate if the request has been approved.
Purchase inquiry	Allows the cardholder to check the status of the processing of an order after the purchase response has been received. Note that this message does not include information such as the status of back ordered goods, but does indicate the status of authorization, capture and credit processing.
Authorization reversal	Allows a merchant to correct previous authorization requests. If the order will not be completed, the merchant reverses the entire authorization. If part of the order will not be completed (such as when goods are back ordered), the merchant reverses part of the amount of the authorization.
Capture reversal	Allows a merchant to correct errors in capture requests such as transaction amounts that were entered incorrectly by a clerk.

SET Transaction Types(3)

Credit	Allows a merchant to issue a credit to a cardholder's account such as when goods are returned or were damaged during shipping. Note that the SET *Credit* message is always initiated by the merchant, not the cardholder. All communications between the cardholder and merchant that result in a credit being processed happen outside of SET.
Credit reversal	Allows a merchant to correct a previously request credit.
Payment gateway certificate request	Allows a merchant to query the Payment Gateway and receive a copy of the gateway's current key-exchange and signature certificates.
Batch administration	Allows a merchant to communicate information to the Payment Gateway regarding merchant batches.
Error message	Indicates that a responder rejects a message because it fails format or content verification tests.

Purchase Request(1)



Purchase Request(2)

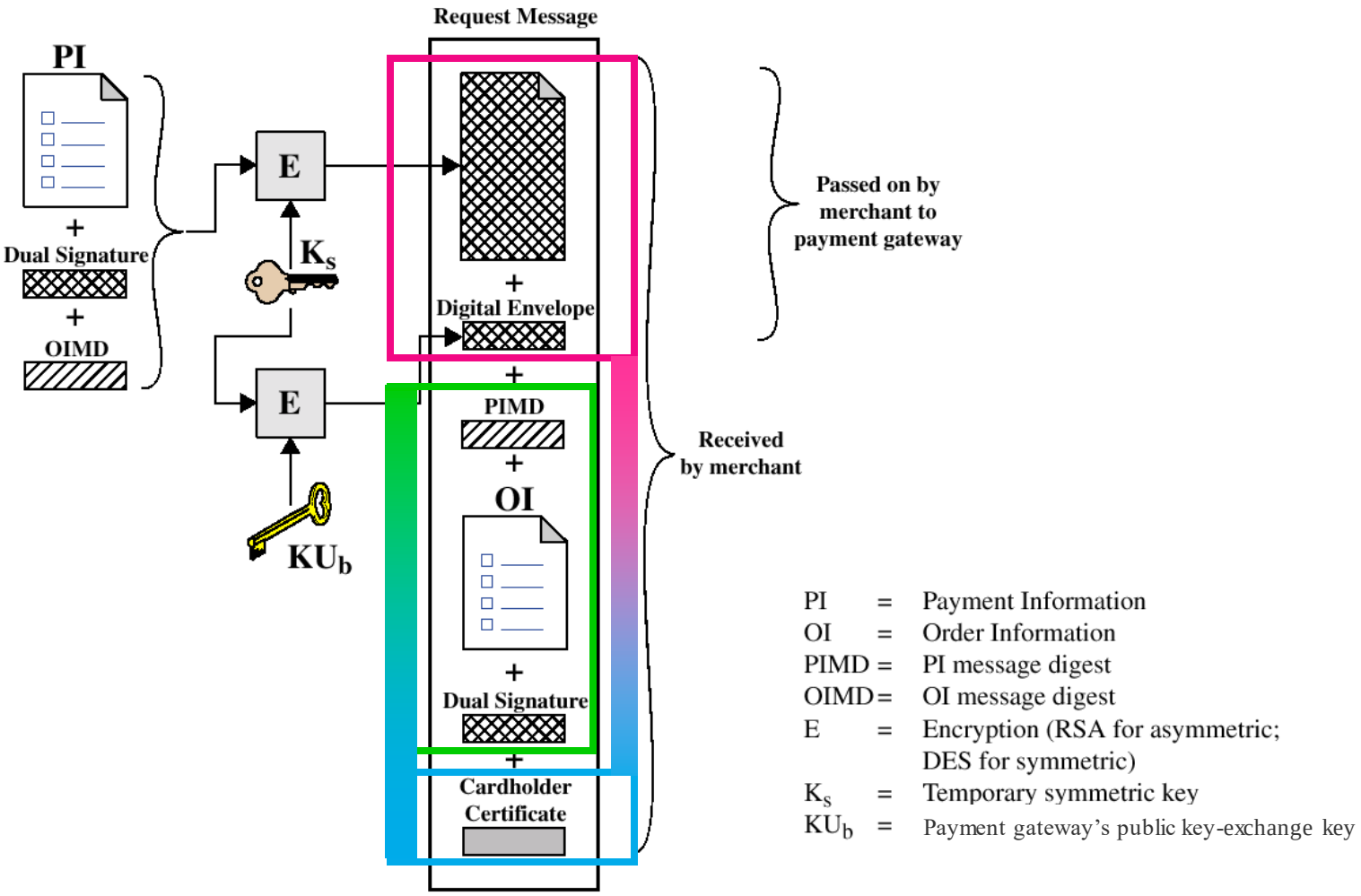
- **All of the preceding occurs without the use of SET.**
  - Customer : visit the cyber store, browsing, selecting, and ordering.
  - Merchant : sending order form to customer

Purchase request exchange consists of 4 messages.

  - Initiate Request, Initiate Response, Purchase Request, and Purchase Response

- **Initiate Request message**
  - The customer requests the certificates of the merchant and the payment gateway.
  - Brand of the credit card that the customer is using.
  - ID of customer and Nonce.
- **Initiate Response message**
  - Merchant signs message with his private signature key.
  - Nonce from the customer and another Nonce by merchant.
  - Transaction ID for this purchase transaction.
  - Merchant's signature certificate and Payment gateway's key exchange certificate.

Purchase Request message(1)

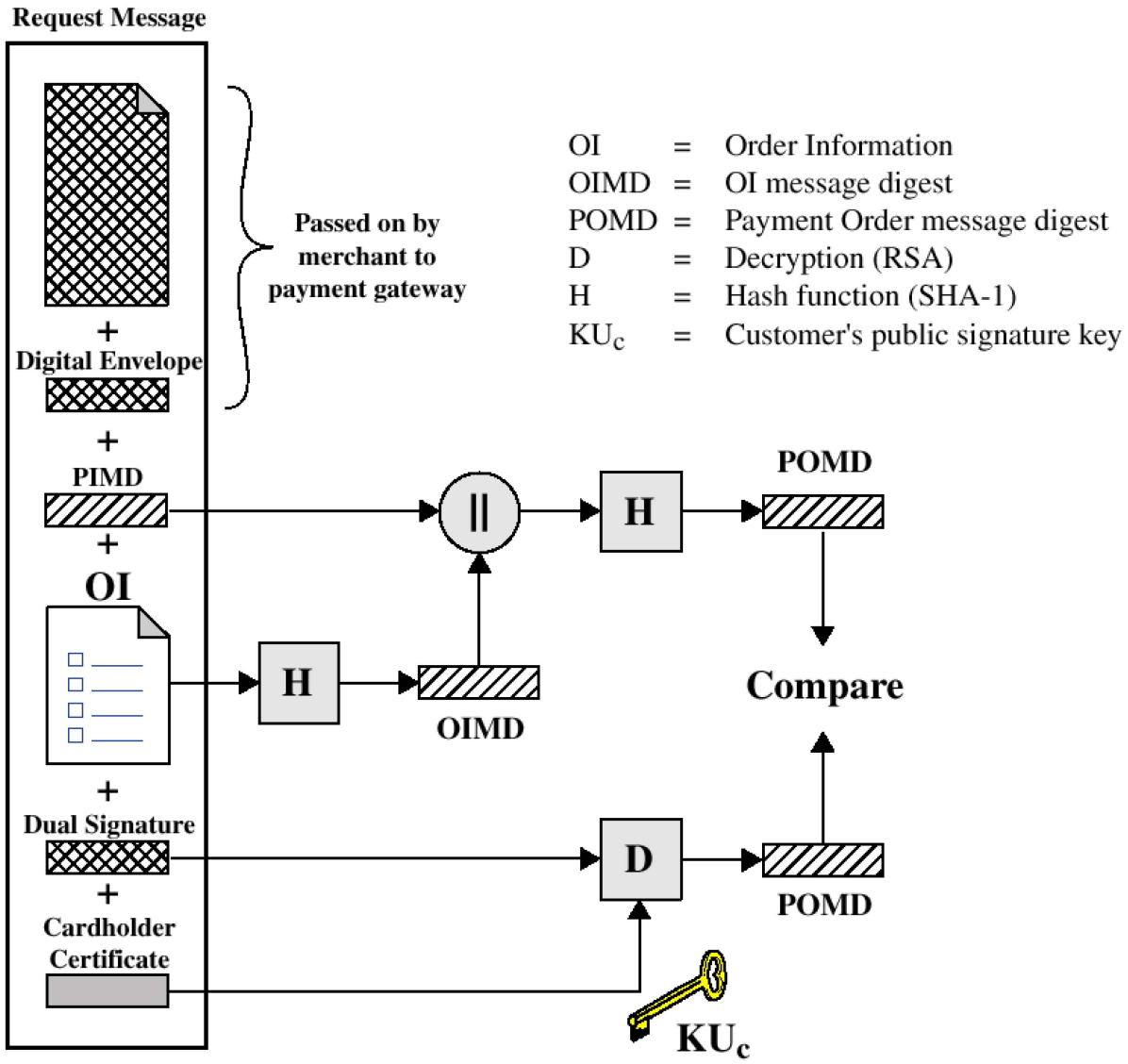


Cardholder sends Purchase Request.

Purchase Request message(2)

- **Purchase-related information**
  - The merchant sends this information to the payment gateway.
  - The PI
  - The dual signature signed with the customer's private signature key.
  - The OI message digest (OIMD)
    - » The OIMD is needed for the payment gateway to verify the dual signature.
  - The digital envelope
    - » This is formed by encrypting Ks with the payment gateway's public key exchange key.
- **Order-related information**
  - This information is needed by the merchant.
  - The OI
  - The dual signature
  - The PI message digest (PIMD)
- **Cardholder's certificate**

Purchase Request message(3)

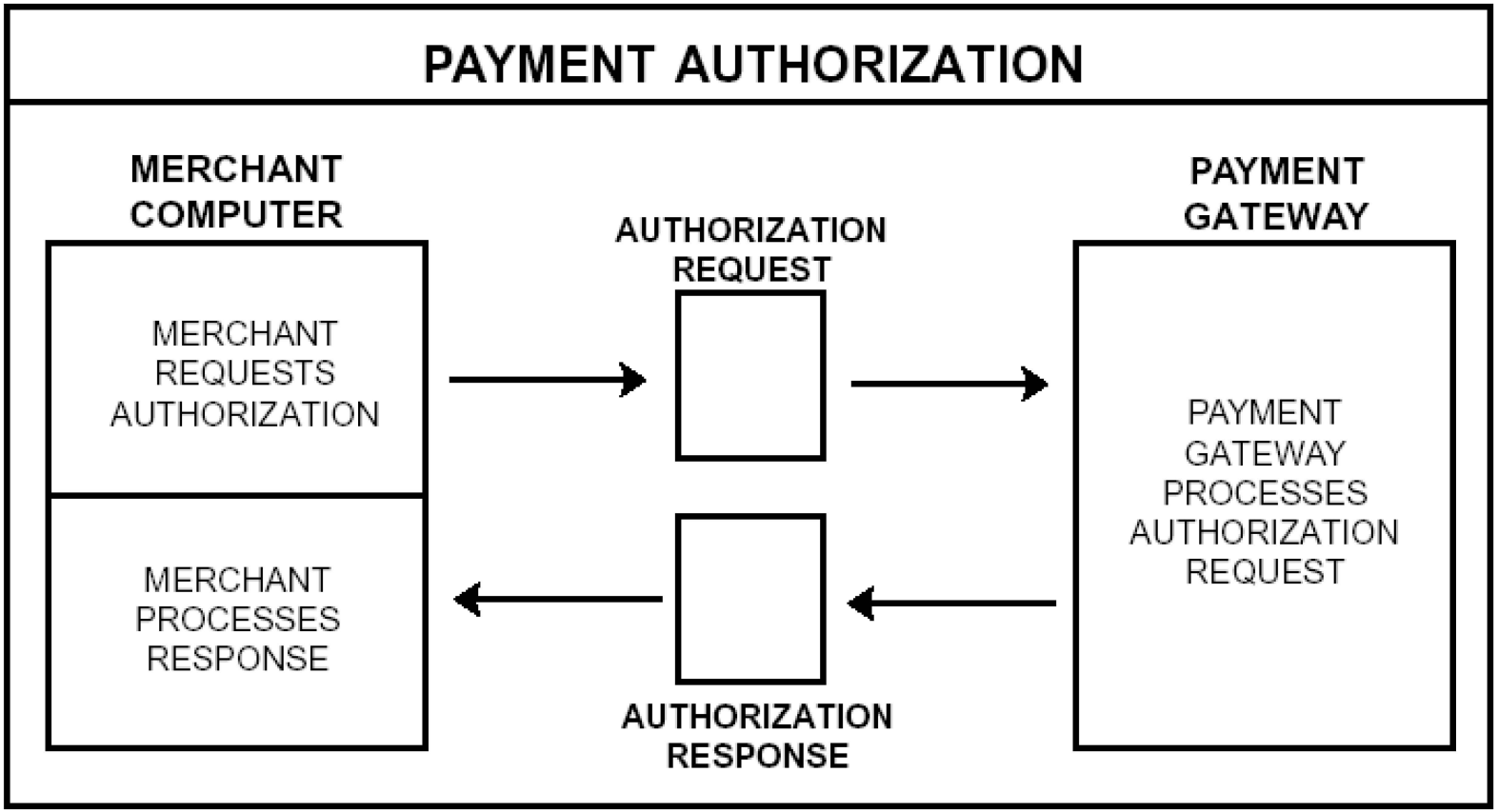


Merchant Verifies Customer Purchase Request.

Purchase Request message(4)

- **Merchant processes purchase request message.**
  - Verifies the cardholder certificates by means of its CA signatures.
  - Verifies the dual signature using the customer's public signature key.
  - Processes the order and forwards the payment information to the payment gateway for authorization.
  - Sends a purchase response message to the cardholder.
- **Purchase response message**
  - It contains that response block and merchant's signature certificate.
  - Response block is signed by the merchant using its private signature key.
- **Cardholder processes purchase response message.**
  - Verifies the merchant's certificate and then verifies the signature on the response block.

Payment Authorization



Authorization Request message

- **Purchase-related information**
  - The PI
  - The dual signature
  - The OI message digest (OIMD)
  - The digital envelope
- **Authorization-related information**
  - An authorization block that includes the transaction ID
  - A digital envelope
- **Certificates.**
  - The cardholder's signature key certificate : used to verify the dual signature.
  - The merchant's signature key certificate : used to verify the merchant's signature in authorization block.
  - The merchant's key-exchange certificate : needed in the payment gateway's response.

Actions of the Payment Gateway (after receiving authorization request message)

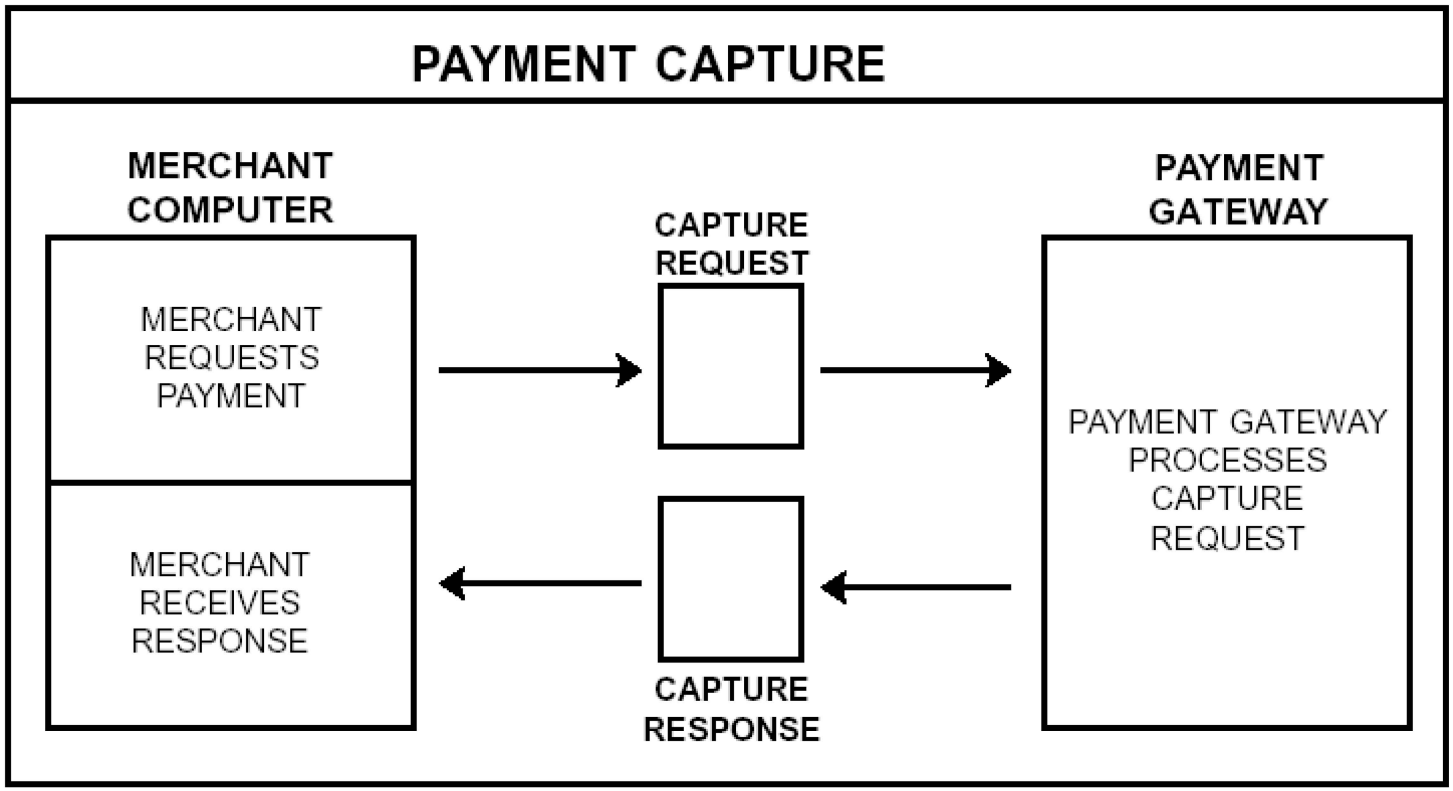
1. Verifies all certificates.
2. Decrypts the digital envelope of the authorization block to obtain the symmetric key and then decrypts the authorization block.
3. Verifies the merchant's signature on the authorization block.
4. Decrypts the digital envelope of the payment block to obtain the symmetric key and then decrypts the payment block.
5. Verifies the dual signature on the payment block.
6. Verifies that the transaction ID received from the merchant matches that in the PI received (indirectly) from the customer.
7. Requests and receives an authorization from the issuer.

Authorization Response message

- **Authorization-related information**
  - An authorization block, signed with the payment gateway's private signature key and encrypted with a one-time symmetric key generated by the payment gateway.
  - A digital envelope that contains the one-time symmetric key encrypted with the merchant's public key-exchange key.
- **Capture token information**
  - A capture token, signed with the payment gateway's private key and encrypted with a newly generated one-time symmetric key.
  - A digital envelope that contains this one-time symmetric key and cardholder account information encrypted with the payment gateway's public key-exchange key.
- **Certificate**
  - The payment gateway's signature key certificate.

※ With the authorization from the gateway, the merchant can **provide the goods or services to the customer.**

Payment Capture(1)



Payment Capture(2)

- **Capture Request message**
  - Capture request block, signed and encrypted.
    - » Payment amount, transaction ID
  - The encrypted capture token received in the authorization response for this transaction.
  - The digital envelope
  - Certificates
- **Capture Response message**
  - Capture response block, signed and encrypted.
  - The digital envelope
  - Certificates

※ The merchant stores the capture response to be used for reconciliation with payment received from the acquirer.


UNIT- V

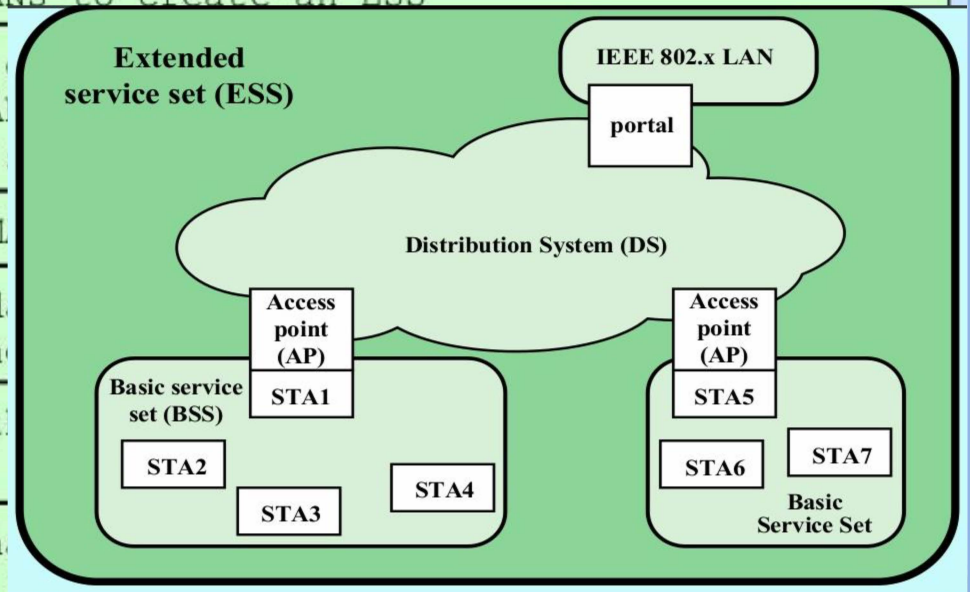

SLO : 1 & SLO :2
Wireless Security :IEEE 802.11 LAN Security

IEEE 802.11 WIRELESS LAN OVERVIEW

- IEEE 802 committee for LAN standards
- IEEE 802.11 formed in 1990's
  - charter to develop a protocol & transmission specifications for wireless LANs (WLANs)
- since then demand for WLANs, at different frequencies and data rates, has exploded
- hence seen ever-expanding list of standards issued

IEEE 802 Terminology

Access point (AP)	Any entity that has station functionality and provides access to the distribution system via the wireless medium for associated stations
Basic service set (BSS)	A set of stations controlled by a single coordination function
Coordination function	The logical function that determines when a station operating within a BSS is permitted to transmit and may be able to receive PDUs
Distribution system (DS)	A system used to interconnect a set of BSSs and integrated LANs to create an ESS
Extended service set (ESS)	A set of one integrated LA layer at any
Frame	Synonym for M
MAC protocol data unit (MPDU)	The unit of d entities usin
MAC service data unit (MSDU)	Information t users
Station	Any device th and physical



Wi-Fi Alliance

- 802.11b first broadly accepted standard
- Wireless Ethernet Compatibility Alliance (WECA) industry consortium formed 1999
  - to assist interoperability of products
  - renamed Wi-Fi (Wireless Fidelity) Alliance
  - created a test suite to certify interoperability
  - initially for 802.11b, later extended to 802.11g
  - concerned with a range of WLANs markets, including enterprise, home, and hot spots

Wireless Network

Wireless networks are categorized into three groups based on their coverage range:

- **Wireless Wide Area Networks (WWAN)**

  WWAN includes wide coverage area technologies such as 2G cellular, Cellular Digital Packet Data (CDPD), Global System for Mobile Communications (GSM), and Mobitex.

- **Wireless Local Area Network (WLAN)**

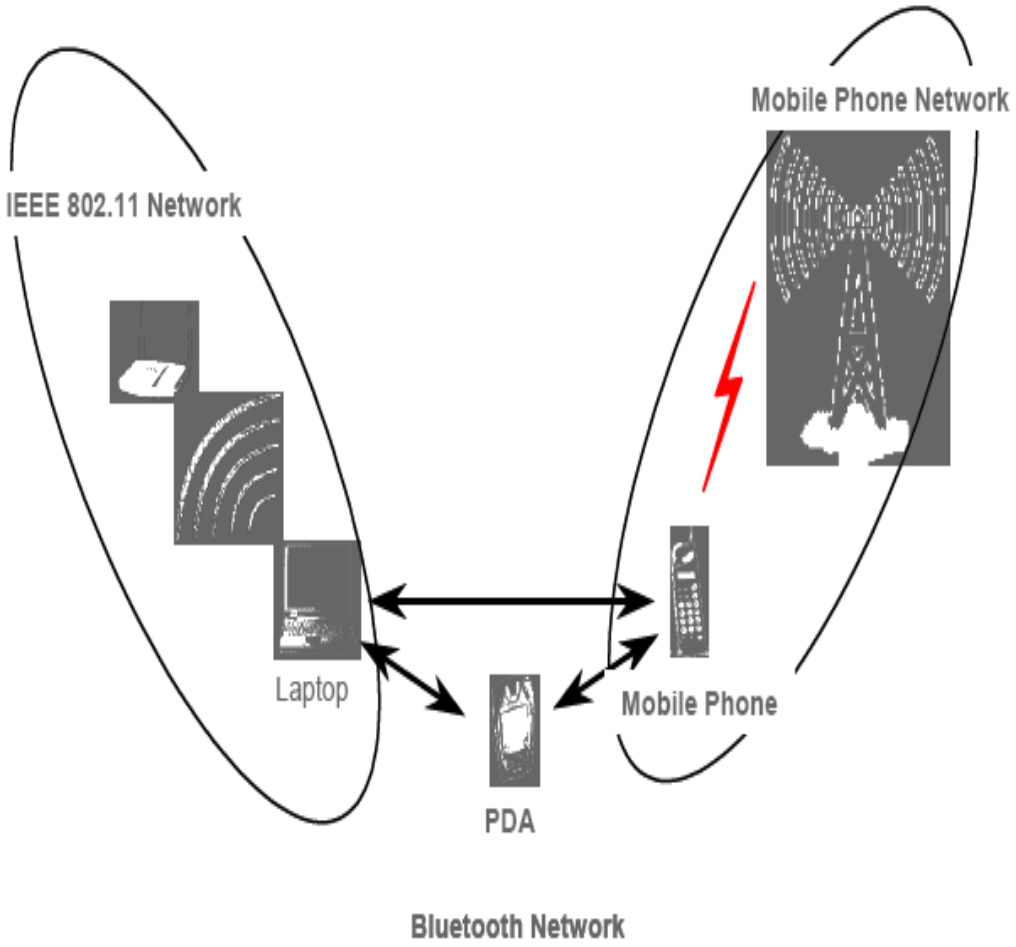
  WLAN, representing wireless local area networks, includes 802.11, HiperLAN, and several others.

- **Wireless Personal Area Networks (WPAN)**

  WPAN, represents wireless personal area network technologies such as Bluetooth and IR(**Infrared** (**IR**) remote controls).

Wireless Network

- Wireless LAN
- Adhoc Network
- Wireless devices
  - Personal digital assistants (PDA)
  - Smart phone
- Wireless Standards
  - IEEE 802.11
  - IEEE 802.15 (Bluetooth)



Wireless security threats

- Security Self-Assessment Guide for Information Technology Systems (SSAGIT) states that information must be protected from unauthorized, unanticipated, or unintentional modification.
    - Authenticity
    - Non-repudiation
    - Accountability
    - Availability
- Risks in wireless networks are equal to the sum of the risk of operating a wired network (as in operating a network in general) plus the new risks introduced by weaknesses in wireless protocols

Wireless security threats

- All the vulnerabilities that exist in a conventional wired network apply to wireless technologies.
- Malicious entities may gain unauthorized access to an agency's computer or voice (IP telephony) network through wireless connections, potentially bypassing any firewall protections
- Sensitive information that is not encrypted (or that is encrypted with poor cryptographic techniques) and that is transmitted between two wireless devices may be intercepted and disclosed
- Denial of service (DoS) attacks may be directed at wireless connections or devices.
- Malicious entities may steal the identity of legitimate users and masquerade as them on internal or external corporate networks

Wireless security threats

- Sensitive data may be corrupted during improper synchronization
- Malicious entities may be able to violate the privacy of legitimate users and be able to track their physical movements.
- Malicious entities may deploy unauthorized equipment (e.g., client devices and access points) to secretly gain access to sensitive information
- Handheld devices are easily stolen and can reveal sensitive information
- Data may be extracted without detection from improperly configured devices
- Viruses or other malicious code may corrupt data on a wireless device and be subsequently introduced to a wired network connection.

Wireless security threats

- Malicious entities may, through wireless connections, connect to other agencies for the purposes of launching attacks and concealing their activity

- Interlopers, from inside or out, may be able to gain connectivity to network management controls and thereby disable or disrupt operations.

- Malicious entities may use a third party, untrusted wireless network services to gain access to an agency's network resources.

- Internal attacks may be possible via ad hoc transmissions.

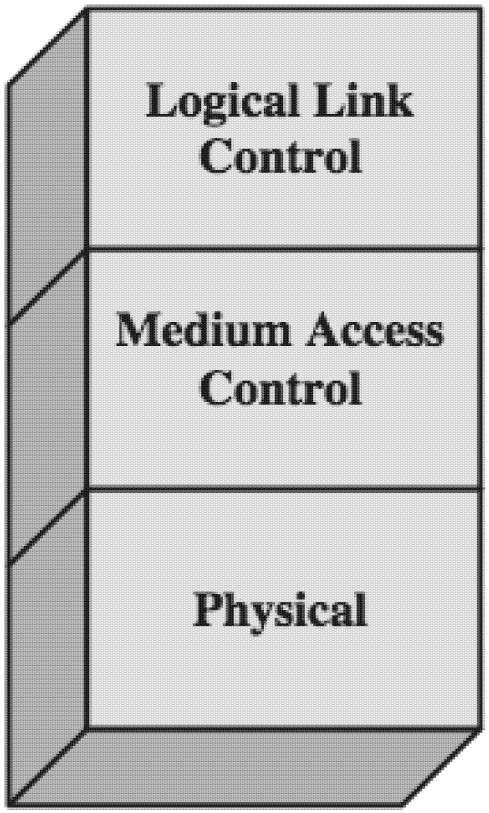
Wireless LAN Characteristics

Characteristic	Description
Physical Layer	Direct Sequence Spread Spectrum (DSSS), Frequency Hopping Spread Spectrum (FHSS), Orthogonal Frequency Division Multiplexing (OFDM), infrared (IR).
Frequency Band	2.4 GHz (ISM band) and 5 GHz.
Data Rates	1 Mbps, 2 Mbps, 5.5 Mbps (11b), 11 Mbps (11b), 54 Mbps (11a)
Data and Network Security	RC4-based stream encryption algorithm for confidentiality, authentication, and integrity. Limited key management. (AES is being considered for 802.11i.)
Operating Range	Up to 150 feet indoors and 1500 feet outdoors.[9]
Positive Aspects	Ethernet speeds without wires; many different products from many different companies. Wireless client cards and access point costs are decreasing.
Negative Aspects	Poor security in native mode; throughput decrease with distance and load.

Benefits of WLAN

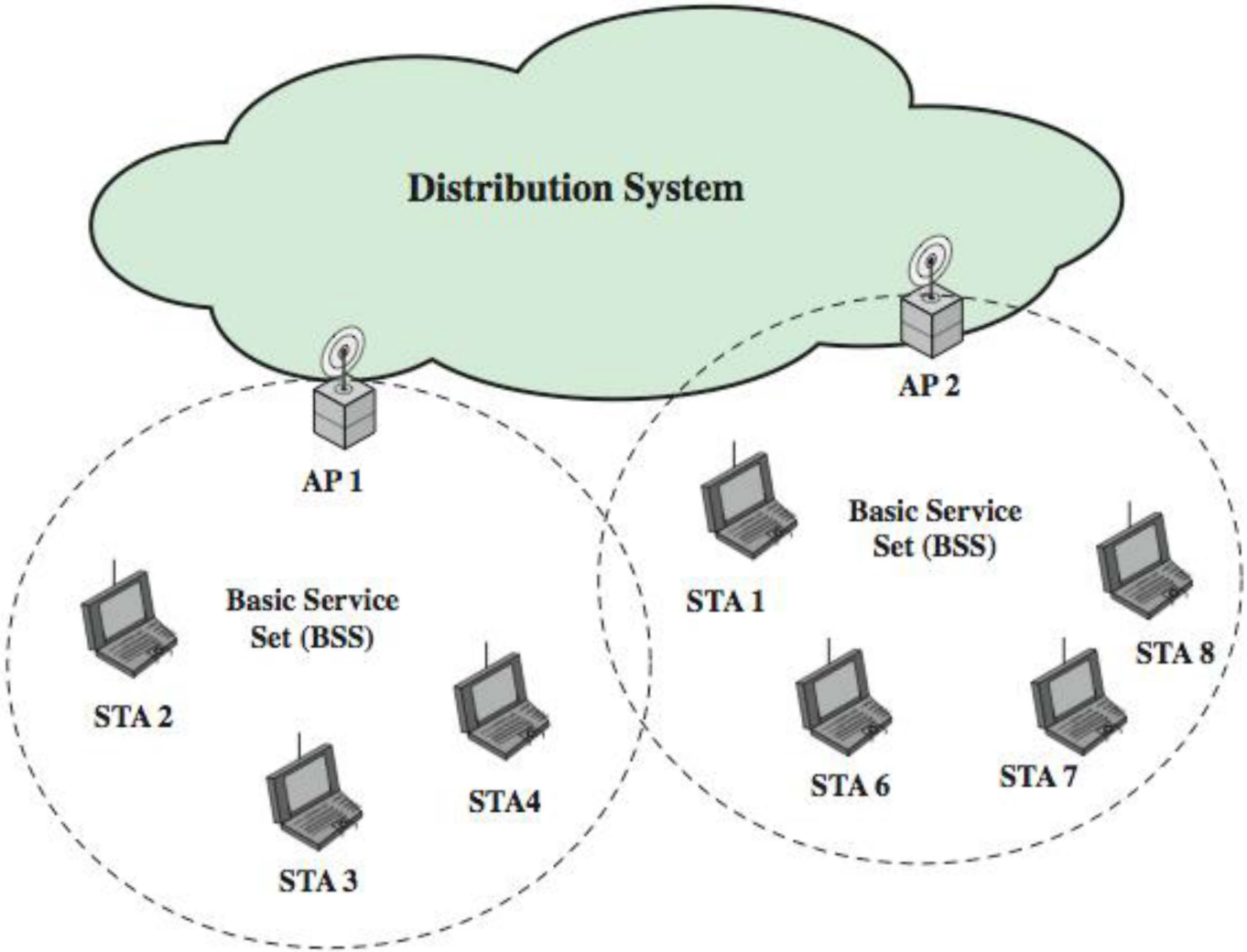
User mobility, Rapid installation, Flexibility, Scalability

IEEE 802 Protocol Architecture



	General IEEE 802 functions	Specific IEEE 802.11 functions
Logical Link Control	Flow control Error control	
Medium Access Control	Assemble data into frame Addressing Error detection Medium access	Reliable data delivery Wireless access control protocols
Physical	Encoding/decoding of signals Bit transmission/ reception Transmission medium	Frequency band definition Wireless signal encoding

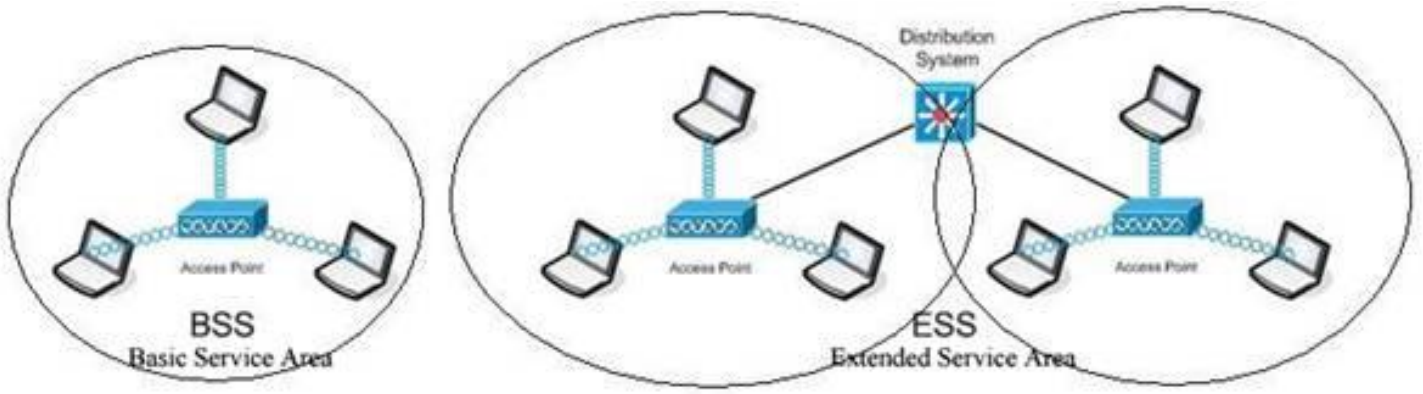
Network Components & Architecture



Wireless LAN Architecture

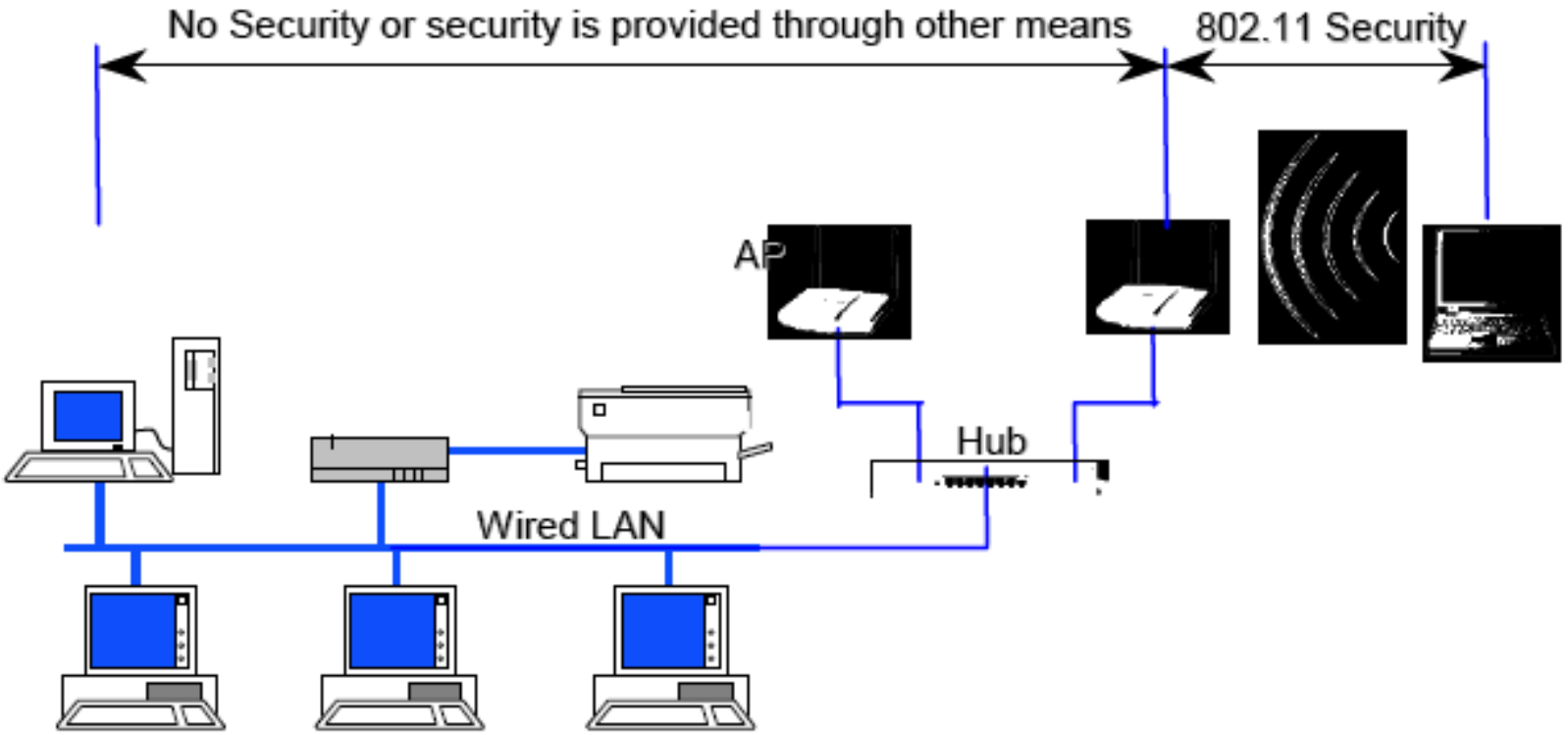
- Infrastructure WLAN
  - Access Point
  - Basic Service Set (BSS)
  - Extended Service Set (ESS)
  - Remote Authentication Dial in User service (RADIUS)

Functionality – Authentication/Authorization/Accounting



Wireless LAN Security

- It is largely provided by the Wired Equivalent Privacy (WEP) protocol to protect link level data during wireless transmission between clients and access points



Wireless LAN Security

Three basic security services provided by the IEEE 802.11 WEP

- **Authentication** – verifying the identity of client stations
- **Confidentiality or privacy** – preventing the information compromise
- **Integrity –** Ensuring the content of message is not modified

Weakness of WEP

- Weakness in key management – Single key for all access points and client radios, Authentication and encryption keys are same
- Shared key authentication failure – no knowledge of secret to gain network access
- Weakness in encryption – short 24 bit IV, Short 40 bit encryption scheme
- Given c_1 and c_2 with same IV, $c_1 \oplus c_2 = p_1 \oplus p_2$ $[p_1 \oplus S \oplus p_2 \oplus S]$, leading to statistical attacks to recover plaintexts

Wireless LAN Security

- In order to solve the weakness of WEP, Wi-Fi Protected Access (WPA) was developed.
- The current state of **802.11i** standard is referred to as Robust Security Network

802.11i RSN defines the following services with high complexity

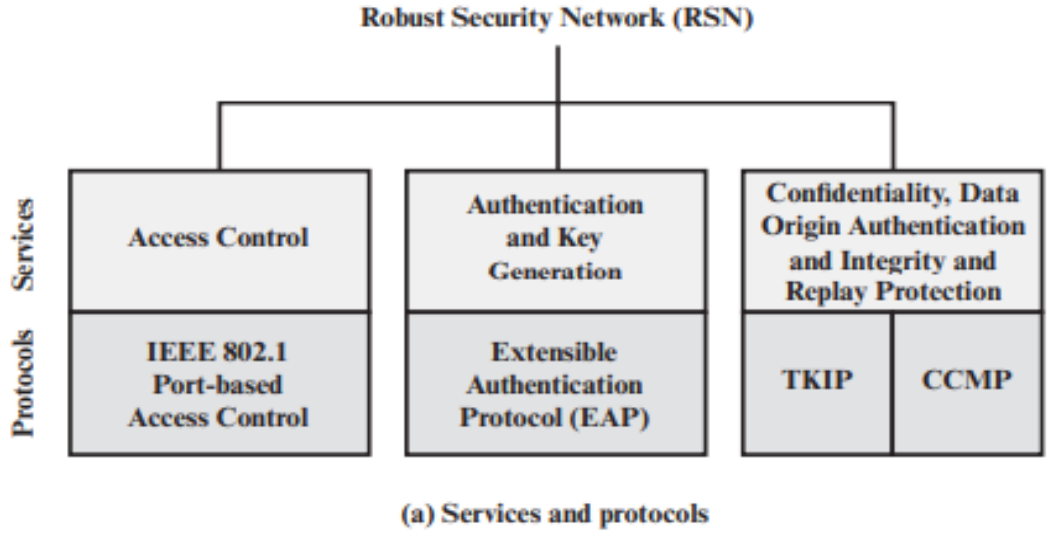
- Authentication – Mutual authentication, temporary keys between client and AP
- Access Control – enforces authentication function, routes the message properly, facilitates key exchanges
- Privacy with message integrity – MAC level data are encrypted along with a message integrity code

Comparison of WEP and WPA

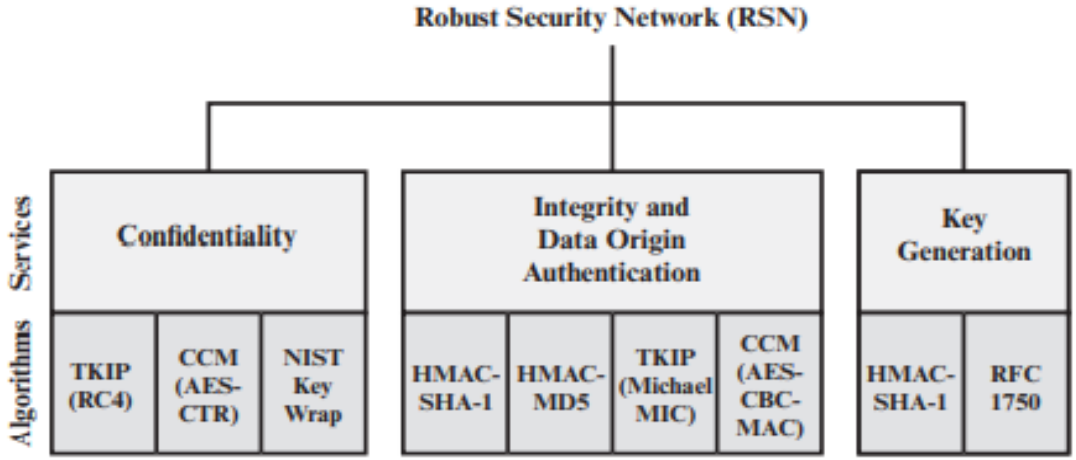
	WEP	WPA
Encryption	Flawed	Fixes all WEP flaws
	40-bit keys	128-bit keys
	Static-same keys used by everyone on network	Dynamic session keys. Per-user, per-session, per-packet keys
	Manual distribution	Automatic Distribution
Authentication	Flawed, uses WEP key itself	Strong user authentication using 802.1X and EAP

Wireless LAN Security

- CBC-MAC → Cipher block chaining message authentication code
- CCM – Counter mode with cipher block chaining message authentication code
- CCMP – Counter mode with cipher block chaining MAC protocol
- TKIP – Temporal Key Integrity Protocol



(a) Services and protocols



(b) Cryptographic algorithms

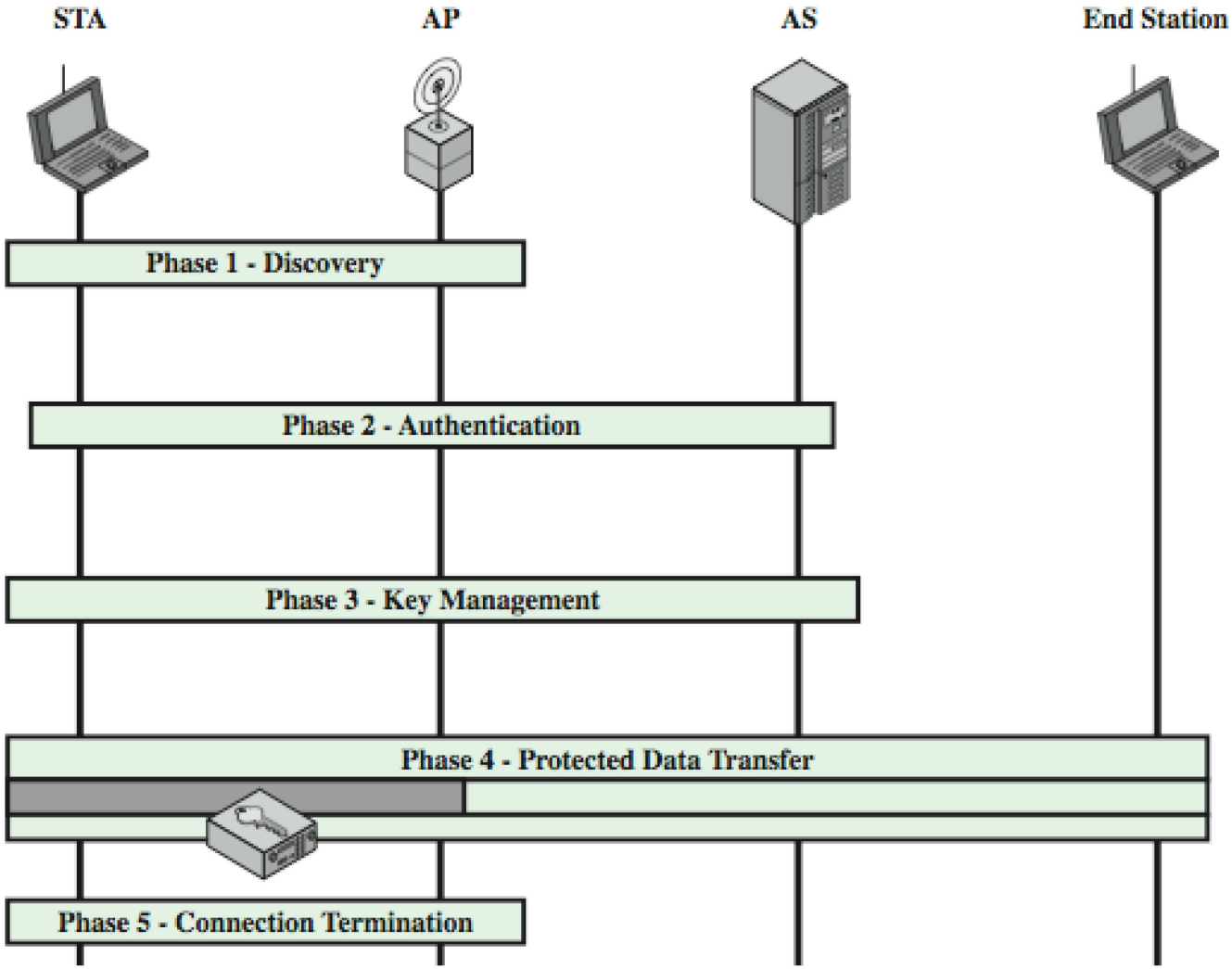
Possible message exchanges in WLAN

- Two wireless stations in the same BSS communicating via the access point (AP) for that BSS.
- Two wireless stations (STAs) in the same ad hoc Independent BSS (IBSS) communicating directly with each other
- Two wireless stations in different BSSs communicating via their respective APs across a distribution system
- A wireless station communicating with an end station on a wired network via its AP and the distribution system.

Possible message exchanges in WLAN

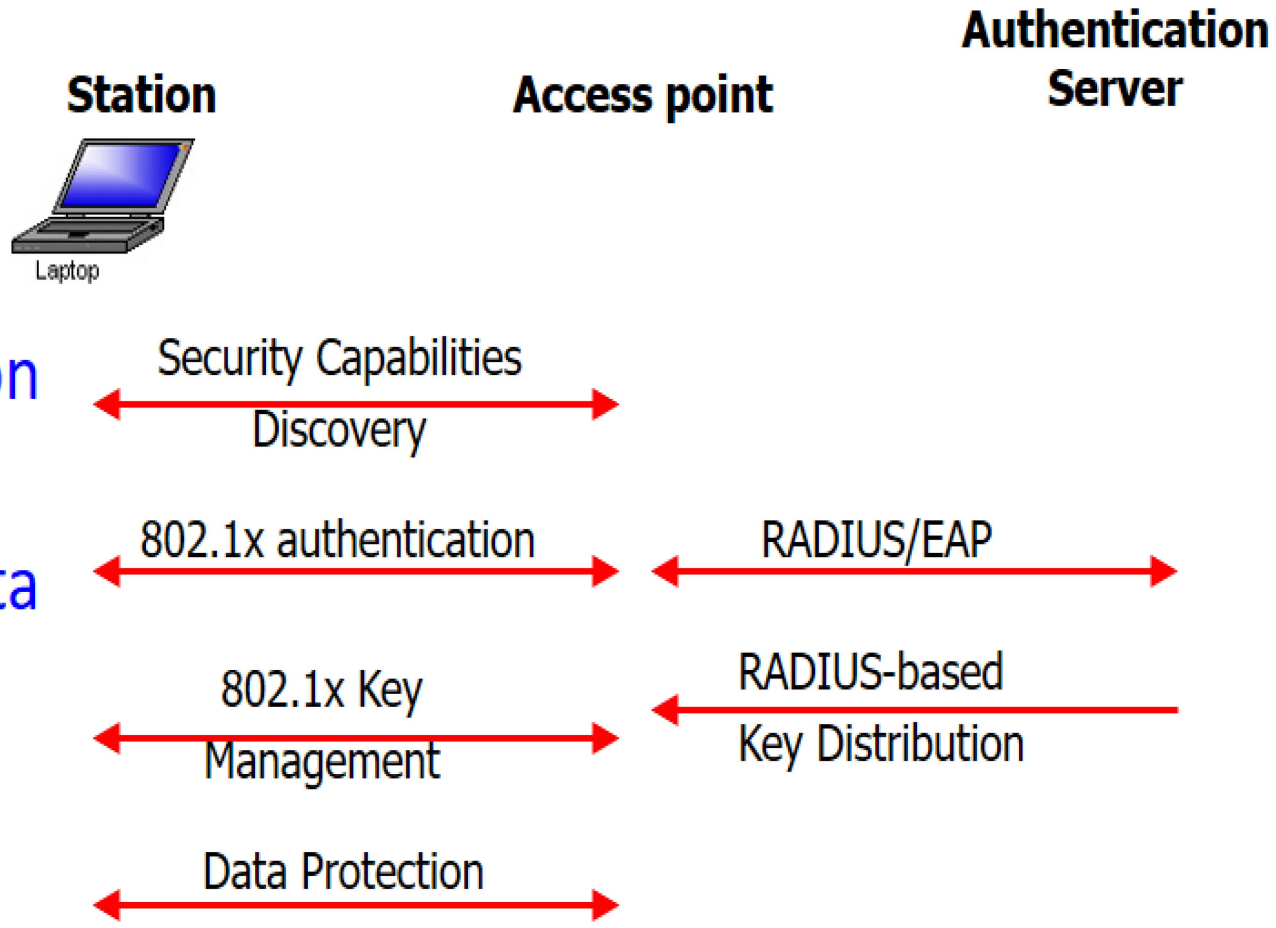
- Two wireless stations in the same BSS communicating via the access point (AP) for that BSS.
- Two wireless stations (STAs) in the same ad hoc Independent BSS (IBSS) communicating directly with each other
- Two wireless stations in different BSSs communicating via their respective APs across a distribution system
- A wireless station communicating with an end station on a wired network via its AP and the distribution system.

802.11i Phases of Operation



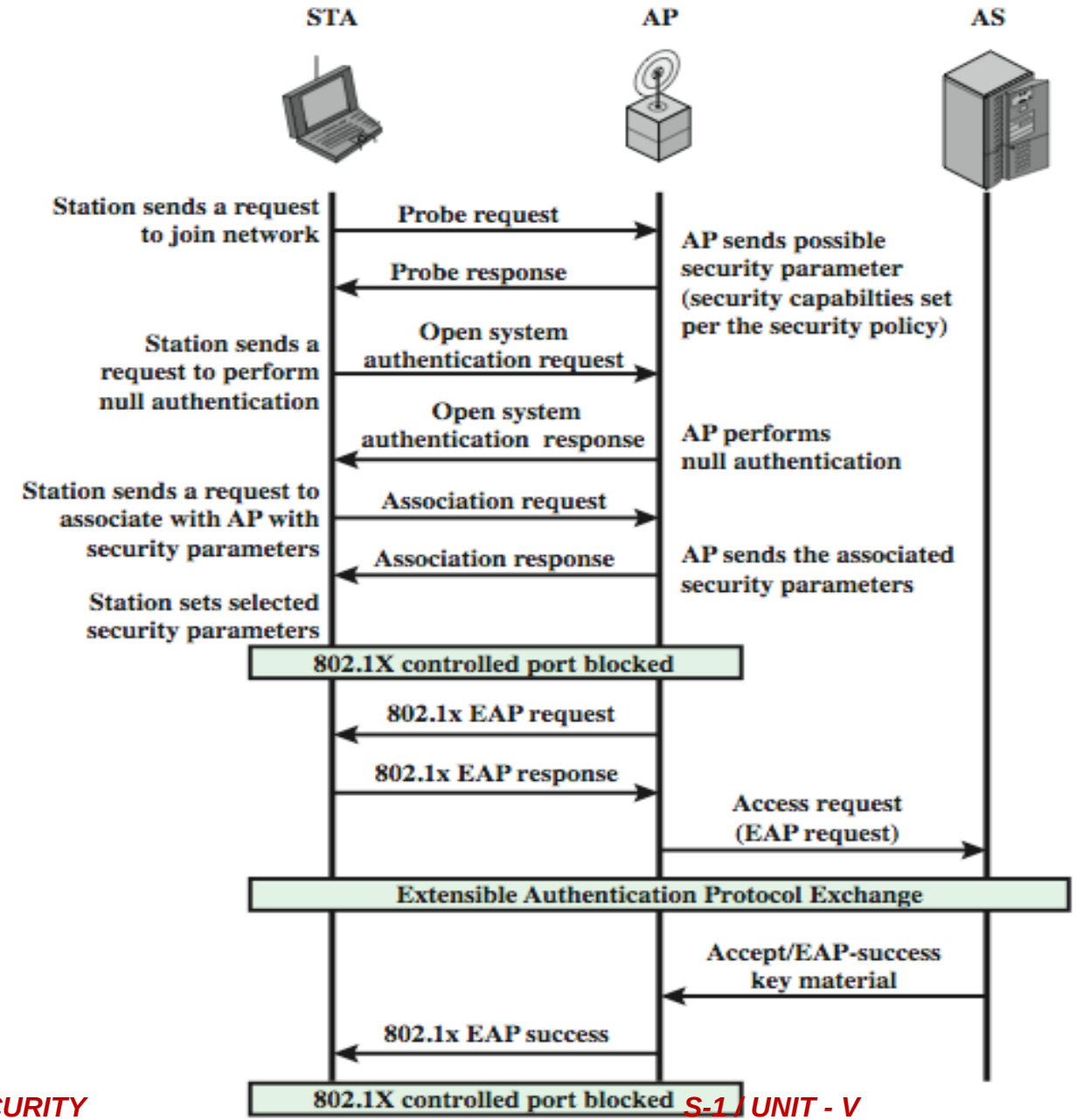
802.11i Phases of operation

- Discovery
- Authentication
- Key generation and distribution
- Protected data transfer
- Connection termination



802.11i Discovery and Authentication Phases

The purpose of this phase is for an STA and an AP to recognize each other, agree on a set of security capabilities, and establish an association for future communication using those security capabilities.



Discovery Phase

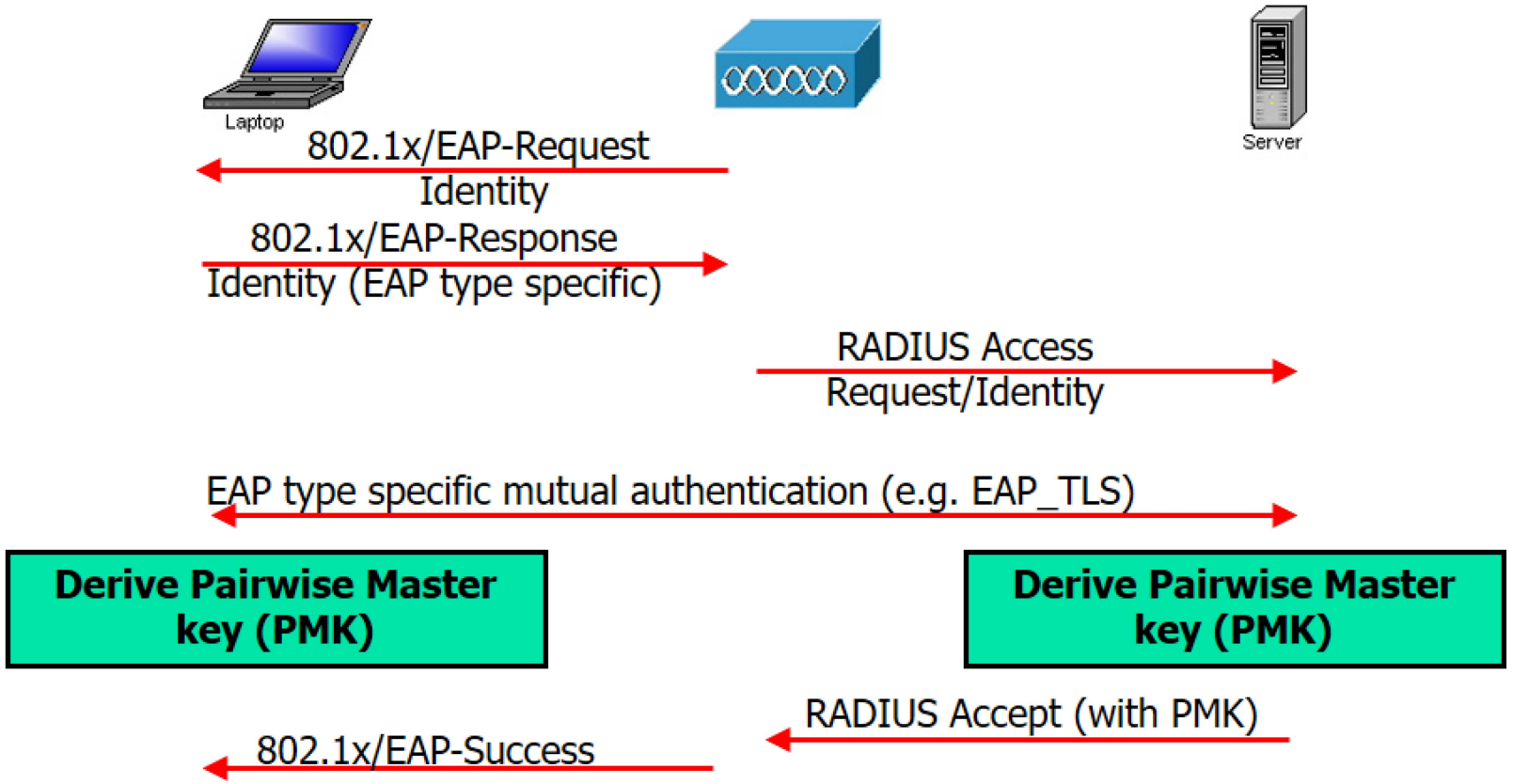
- security capabilities between STA and AP
  - Confidentiality protocol
  - MAC protocol data unit (MPDU) integrity protocols
  - Authentication methods
  - Cryptographic key management approach
- Three type of message exchanges in discovery phase (MPDU Exchanges)
  - Network and security capability discovery
  - Open system authentication
  - Association

Mobile Station (MS) and AP connection

- Discovery process
  - AP broadcasts beacon from periodically to discover the mobile nodes in its coverage area
    - Beacon frame contains the SSID (Service Set ID) of broadcasting AP, its data rates etc.
  - Mobile node sends a probe request frame
- Station association
  - A station willing to have association with AP will send Associate request frame
  - AP will send the Associate response frame if it accepts the request
  - Before association, it requires authentication
  - A station can have the association with only one AP
- Protocols
  - Wired Equivalent Privacy (WEP)
  - WiFi Protocol Access (WPA) – also called Temporal key integrity protocol (TKIP)
  - IEEE 802.11i – WPA2

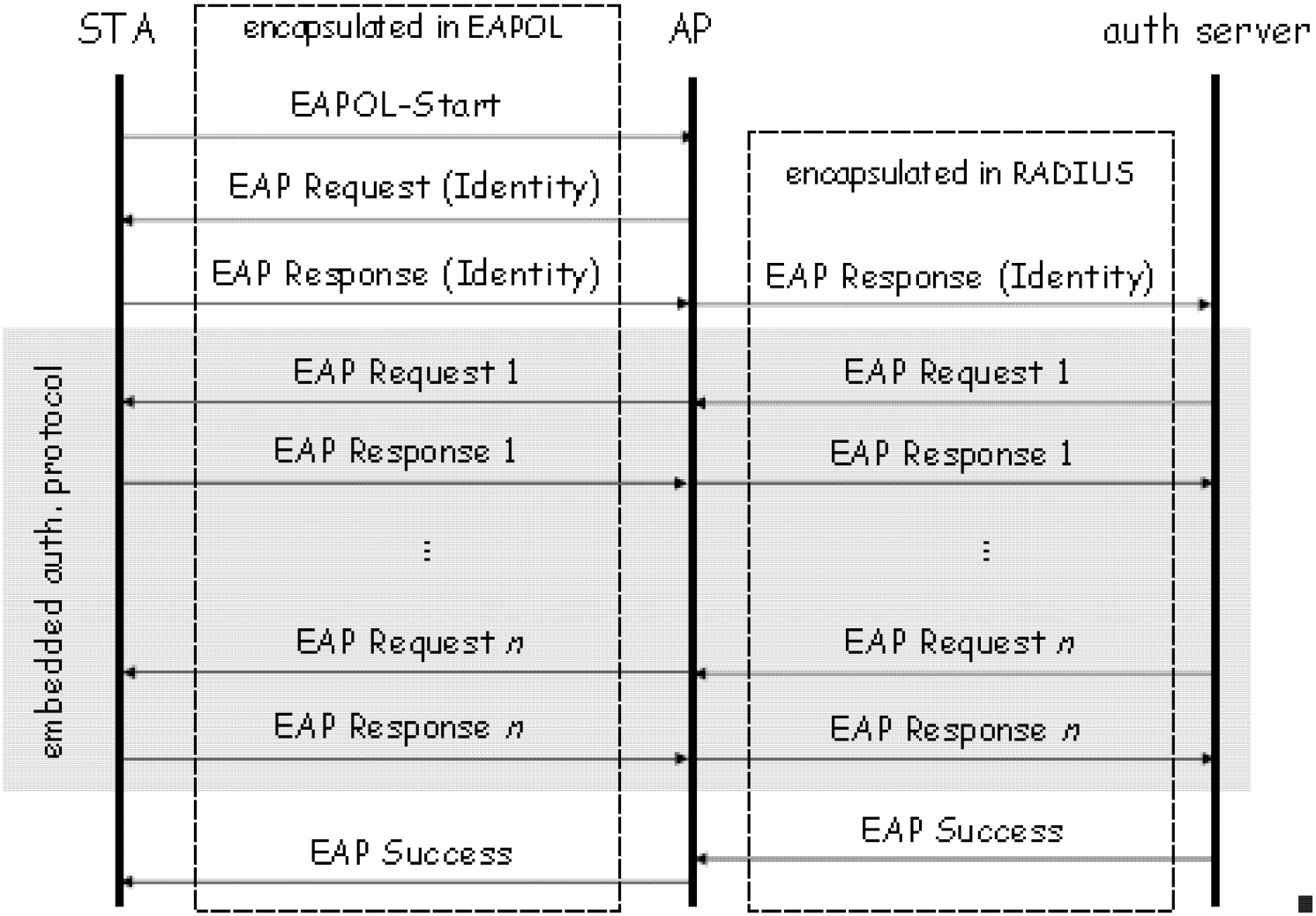
Authentication phase

- Mutual authentication between STA and Authentication server (AS) located in DS



EAP – Extensible Authentication Protocol

- This standard defines the Extensible Authentication Protocol (EAP), which uses a central authentication server to authenticate each user on the network.
- EAP is an 802.1x standard that allows developers to pass security authentication data between authentication server, access point (AP) and wireless client


STA
encapsulated in EAPOL
AP
auth server
EAPOL-Start
EAP Request (Identity)
encapsulated in RADIUS
EAP Response (Identity)
EAP Response (Identity)
EAP Request 1
EAP Request 1
EAP Response 1
EAP Response 1
EAP Request n
EAP Request n
EAP Response n
EAP Response n
EAP Success
EAP Success
embedded auth. protocol

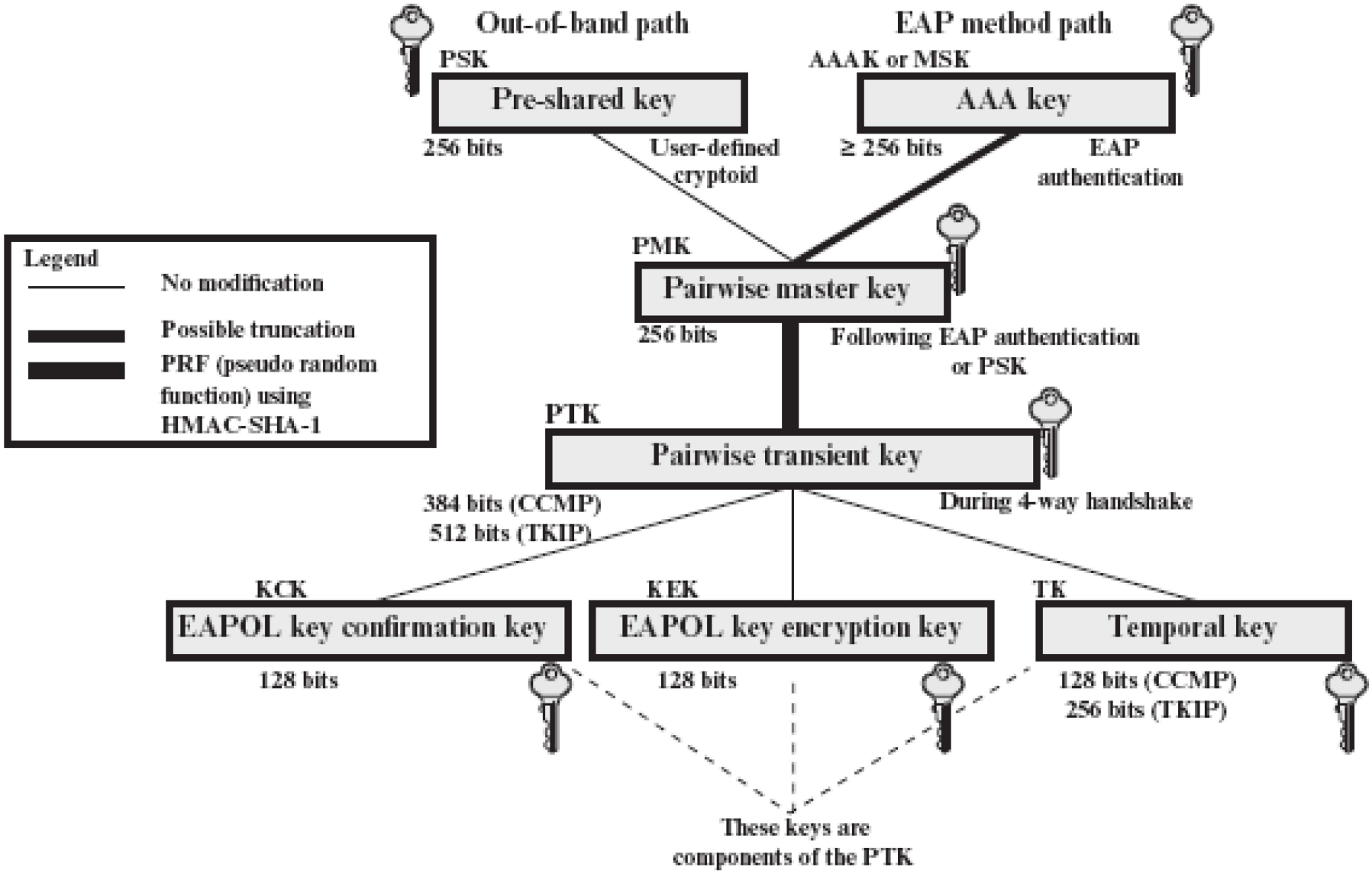
EAP – Protocol supporting

- **EAP – MD5**
  - AS challenges the MS to transmit the MD5 of the user's password. User can send hash of password to AP. But attacker could eavesdrop on such message exchange and replay the hashed password, thus impersonates like the owner of the password
  - No support for authentication of AP to the station
- **EAP – TLS**
  - It provides mutual authentication and agreement on a master session key.
  - It requires AP and MS to have digital certificates
- **EAP – TTLS (Tunneled TLS)**
  - First the AP authenticates itself to MS
  - Second Both AP and MS construct the secure tunnel between then
  - Third, Over the secure tunnel, the MS authenticates itself to AP by sending its user name and password
- **EAP – PEAP**
  - (protected EAP) proposed by microsoft, Cisco and RSA security
  - Similar to EAP-TTLS

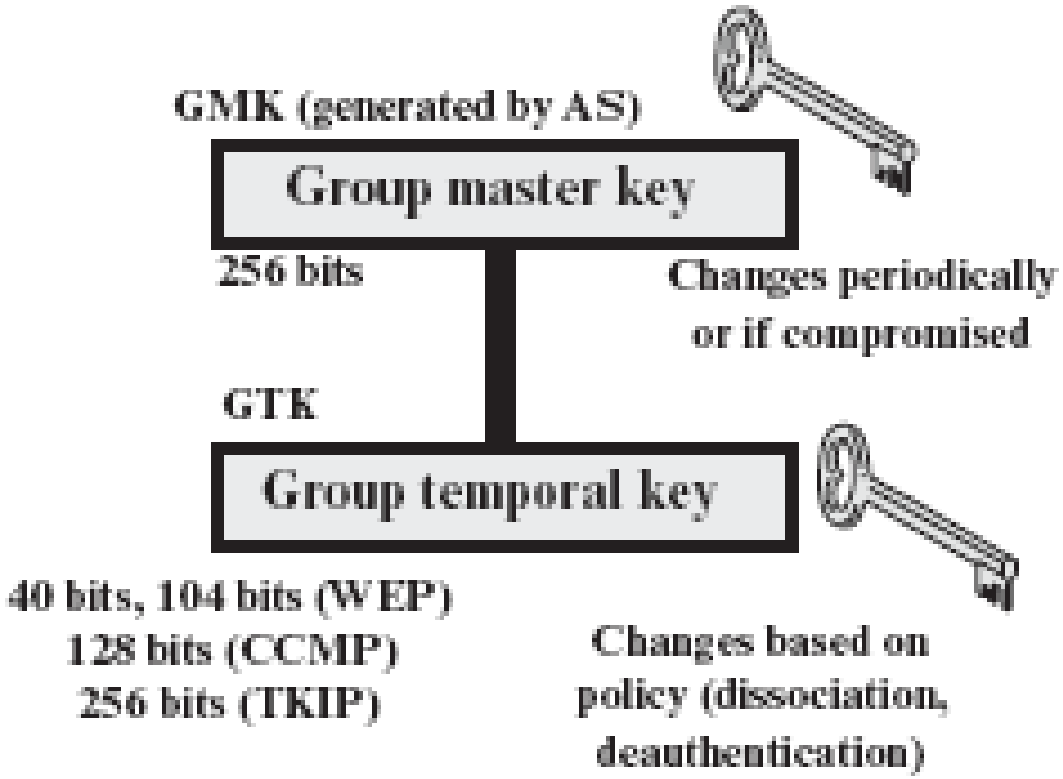
3. Key management

- Two types of Keys
  - Pairwise key – to protect the traffic between MS and AP
  - Group key – To protect the broadcast and multicast traffic between an AP and multiple MS
- Master Session Key
  - The MS and AS agrees on MSK as part of authentication
- Pairwise Master key (PMK)
  - AS supplies the MSK to both MS and AP
  - AP and MS will derive PMK from MSK
- Pairwise Transient Key (PTK)
  - 256 bit PMK is used to derive 384 bit PTK.
  - PTK is pseudo random function of PMK, two nonce chosen by AP and MS, MAC address of MS
  - From 384 bit PTK, 128 bit chunks are extracted to generate Temporal Key (TK), Key Confirmation Key (KCK) and Key Encryption Key (KEK)
- Temporal key
  - Used for both encryption and integrity protection of data between the AP and MS
- Key Confirmation Key
  - Used to integrity-protect some messages which carries group keys
- Key Encryption Key (KEK)
  - Used to encrypt the message containing the group key

Pairwise key



Group key


GMK (generated by AS)
Group master key
256 bits
Changes periodically
or if compromised
GTK
Group temporal key
40 bits, 104 bits (WEP)
128 bits (CCMP)
256 bits (TKIP)
Changes based on
policy (dissociation,
deauthentication)

Key Hierarchy

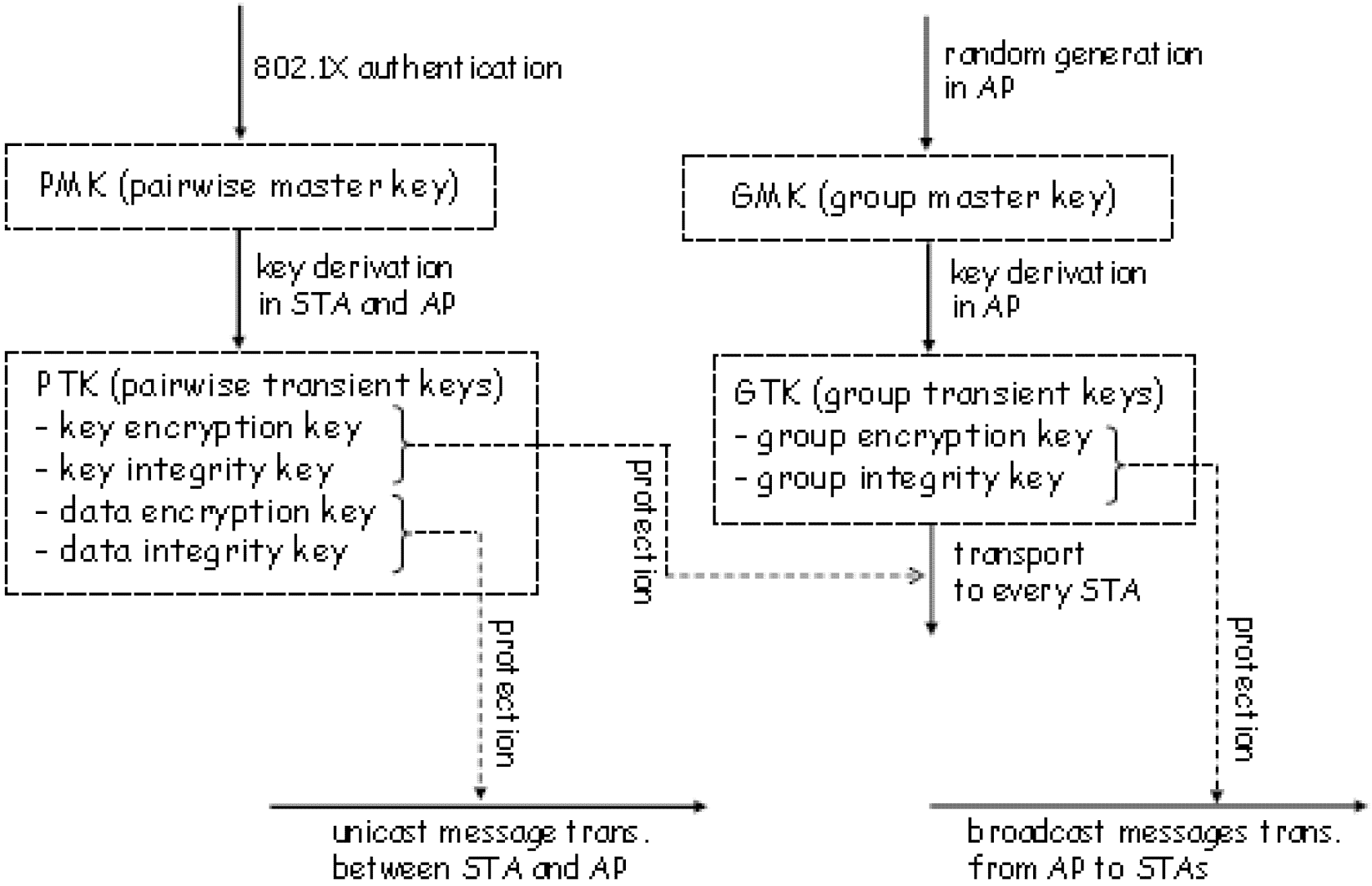


Table 18.3 IEEE 802.11i Keys for Data Confidentiality and Integrity Protocols

Abbreviation	Name	Description / Purpose	Size (bits)	Type
AAA Key	Authentication, Accounting, and Authorization Key	Used to derive the PMK. Used with the IEEE 802.1X authentication and key management approach. Same as MMSK.	≥ 256	Key generation key, root key
PSK	Pre-shared Key	Becomes the PMK in pre-shared key environments.	256	Key generation key, root key
PMK	Pairwise Master Key	Used with other inputs to derive the PTK.	256	Key generation key
GMK	Group Master Key	Used with other inputs to derive the GTK.	128	Key generation key
PTK	Pair-wise Transient Key	Derived from the PMK. Comprises the EAPOL-KCK, EAPOL-KEK, and TK and (for TKIP) the MIC key.	512 (TKIP) 384 (CCMP)	Composite key
TK	Temporal Key	Used with TKIP or CCMP to provide confidentiality and integrity protection for unicast user traffic.	256 (TKIP) 128 (CCMP)	Traffic key
GTK	Group Temporal Key	Derived from the GMK. Used to provide confidentiality and integrity protection for multicast/broadcast user traffic.	256 (TKIP) 128 (CCMP) 40,104 (WEP)	Traffic key
MIC Key	Message Integrity Code Key	Used by TKIP's Michael MIC to provide integrity protection of messages.	64	Message integrity key
EAPOL-KCK	EAPOL-Key Confirmation Key	Used to provide integrity protection for key material distributed during the 4-Way Handshake.	128	Message integrity key
EAPOL-KEK	EAPOL-Key Encryption Key	Used to ensure the confidentiality of the GTK and other key material in the 4-Way Handshake.	128	Traffic key / key encryption key
WEP Key	Wired Equivalent Privacy Key	Used with WEP.	40,104	Traffic key

Four way handshake

- objective:
  - prove that AP also knows the PMK (result of authentication)
  - exchange random values to be used in the generation of PTK

- protocol:

AP : generate ANonce

AP → STA : ANonce | KeyReplayCtr

STA : generate SNonce and compute PTK

STA → AP : SNonce | KeyReplayCtr | MIC_{KIK}

AP : compute PTK, generate GTK, and verify MIC

AP → STA : ANonce | KeyReplayCtr+1 | $\{GTK\}_{KEK}$ | MIC_{KIK}

STA : verify MIC and install keys

STA → AP : KeyReplayCtr+1 | MIC_{KIK}

AP : verify MIC and install keys

MIC – Message Integrity Check

4. Protected data transfer phase

- Two methods to protect the data transmitted in 802.11
  - Temporal Key Integrity Protocol (TKIP)
  - Counter Mode – CBC MAC Protocol (CCMP)

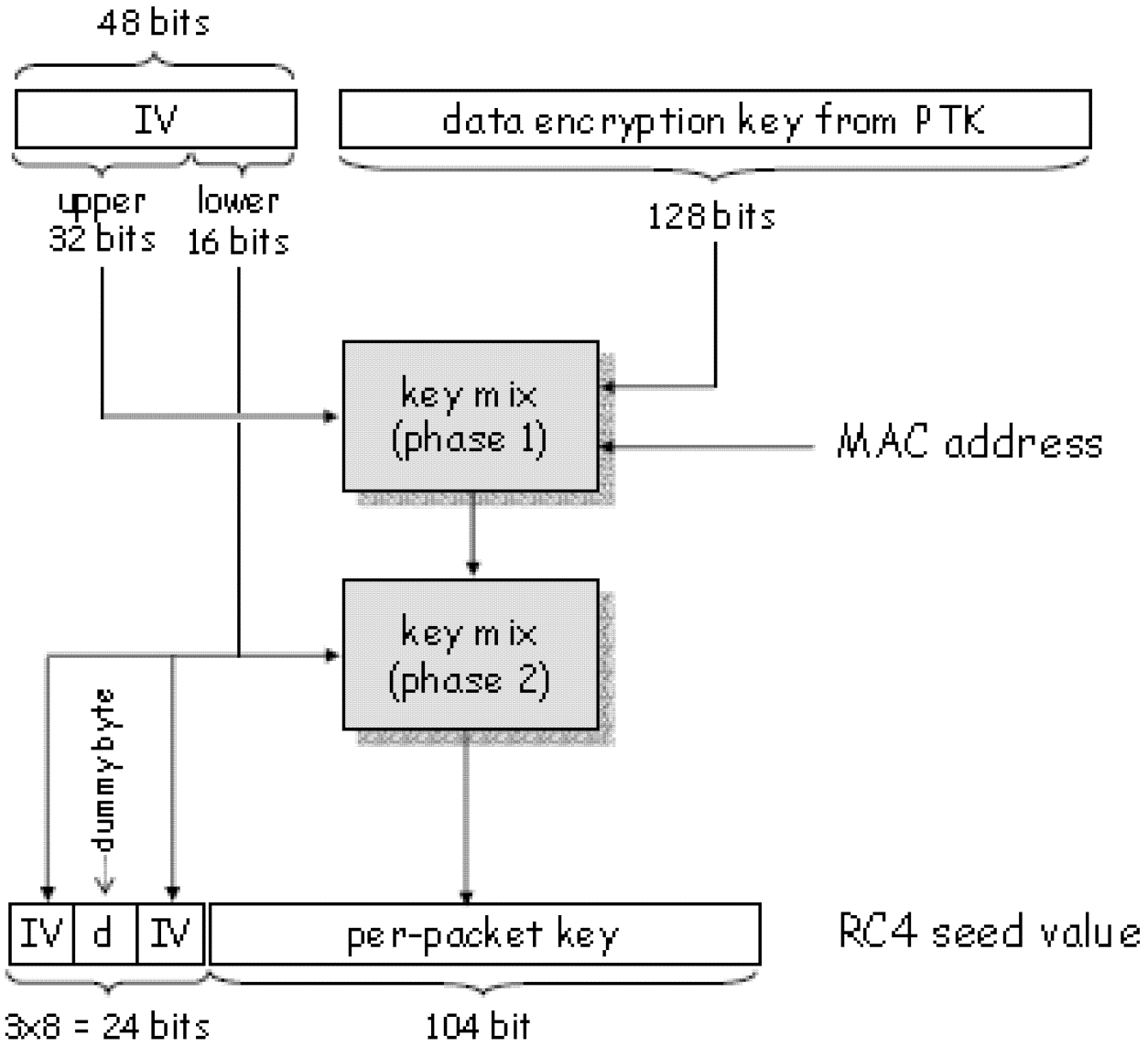
Temporal Key Integrity Protocol (TKIP)

runs on old hardware (supporting RC4), but …

WEP weaknesses are corrected

- new message integrity protection mechanism called Michael
  - MIC value is added at SDU level before fragmentation into PDUs
  - implemented in the device driver (in software)
- use IV as replay counter
- increase IV length to 48 bits in order to prevent IV reuse
- per-packet keys to prevent attacks based on weak keys

TKIP


48 bits
IV
data encryption key from PTK
upper
32 bits
lower
16 bits
128 bits
key mix
(phase 1)
MAC address
key mix
(phase 2)
dummy byte
IV
d
IV
per-packet key
RC4 seed value
3x8 = 24 bits
104 bit

Counter mode with CBC MAC Protocol (CCMP)

- CCMP means CTR mode and CBC-MAC
  - integrity protection is based on CBC-MAC (using AES)
  - encryption is based on CTR mode (using AES)
- CBC-MAC
  - CBC-MAC is computed over the MAC header, CCMP header, and the MPDU (fragmented data)
  - mutable fields are set to zero
  - input is padded with zeros if length is not multiple of 128 (bits)
  - CBC-MAC initial block:
    - flag (8)
    - priority (8)
    - source address (48)
    - packet number (48)
    - data length (16)
  - final 128-bit block of CBC encryption is truncated to (upper) 64 bits to get the CBC-MAC value
- CTR mode encryption
  - MPDU and CBC-MAC value is encrypted, MAC and CCMP headers are not
  - format of the counter is similar to the CBC-MAC initial block
    - "data length" is replaced by "counter"
    - counter is initialized with 1 and incremented after each encrypted block

SLO : 1 & SLO :2
AUTHENTICATION and CONFIDENTIALITY

AUTHENTICATION

1. Pre-WEP Authentication

a. Early versions of 802.11 use naïve approaches: knowledge of SSID sufficed for a station to be authenticated to the AP

➢ Drawbacks: An attacker could easily sniff the value of SSID from frames such as the beacon or probe response and then use it for authentication.

b. Another approach was to restrict admission to the WLAN by MAC address.

✓ The AP would maintain a list of MAC addresses (access control list) of stations permitted to join the WLAN.

✓ valid MAC addresses could be obtained by sniffing the wireless medium.

✓ The attacker could then modify his network card to spoof a valid MAC address. So, neither of these approaches was truly secure.

Authentication in WEP

2. Authentication in WEP

➢ In WEP, the station authenticates itself to the AP using a challenge—response protocol.

➢ Basically, the AP generates a challenge (nonce) and sends it to the station.

➢ The station encrypts the challenge and sends it to the AP.

➢ The stream cipher, RC4, is used for encryption.

➢ Response From Station: the station computes a key stream, which is a function of a 40-bit shared secret, S, and a 24-bit Initialization Vector (IV).

➢ The challenge is then XORed with the keystream to create the response.

RESPONSE = CHALLENGE (XOR) KEYSTREAM(S, IV)

Authentication in WEP

➢ The response together with the IV is sent by the station to the AP.

➢ The shared secret, S, is common to all stations authorized to use the WLAN.

Drawbacks:

➢ All an attacker needs to do is to monitor a challenge—response pair.

➢ From this, he can compute the keystream.

➢ To authenticate himself to the AP, he needs to XOR the challenge from the AP with the computed keystream.

➢ It may also be possible for an attacker to obtain S itself.

➢ By eavesdropping on several challenge—response pairs between the AP and various stations, an attacker could launch a dictionary attack and eventually obtain S.

Authentication and key agreement in 802.11

3. Authentication and key agreement in 802.11

Authentication

➢ 802.11i uses IEEE 802.1x — a protocol that supports authentication at the link layer.

Three entities are involved:

1. Supplicant (the wireless station).
2. Authenticator (the AP in our case).
3. Authentication server.

➢ Different authentication mechanisms and message types are defined by the Extensible authentication Protocol (EAP) standardized by Internet Engineering Task Force (IETF).

➢ EAP is not really an authentication protocol but rather a framework upon which various authentication protocols can be supported.

➢ EAP exchanges are mostly comprised of requests and responses.

➢ For example one party requests the ID of another party.

➢ The latter responds with its username or e-mail address.

➢ EAP also defines messages that may contain challenges and responses used in authentication protocols.

➢ The AP broadcasts its security capabilities in the Beacon or Probe Response frames.

Authentication and key agreement in 802.11

➢ The station uses the Associate Request frame to communicate its security capabilities.

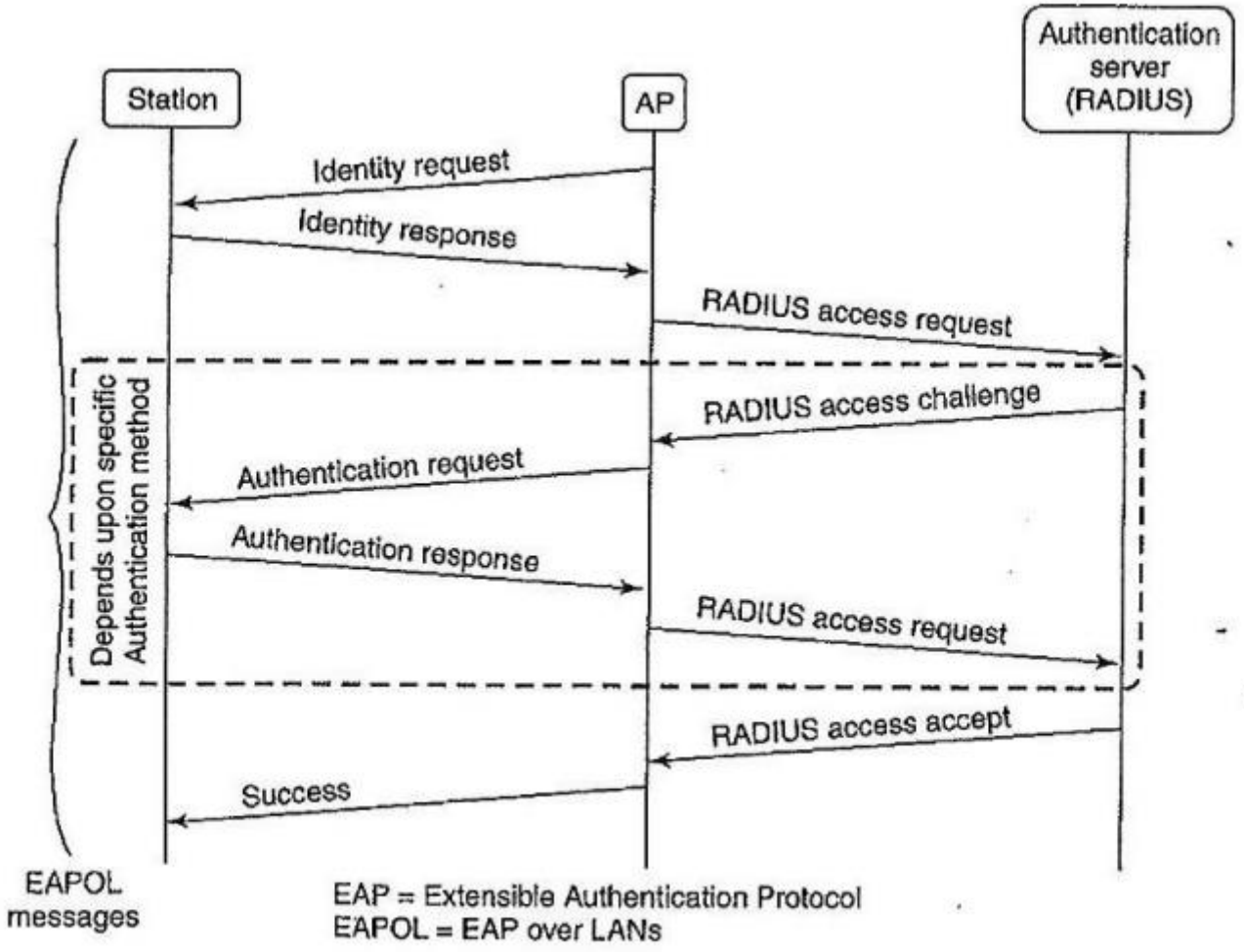
➢ 802.11i authentication takes place after the station associates with an AP. IEEE 802.11i

➢ The generic authentication messages in IEEE 802.11i are shown in Fig.(next slide)

➢ The protocol used between the station and the AP is EAP but that used between the AP and the authentication server depends upon the specifics.

➢ For example, the authentication server is often a RADIUS server which uses its own message types and formats. (RADIUS stands for Remote Authentication Dial in User Service. It is a client—server protocol used for authentication, authorization, and accounting.)

Authentication messages in IEEE 802.11i



Authentication and master session key exchange in 802.11i

Authentication Methods

➢ The main authentication methods supported by EAP include the following:

1. EAP-MDS
2. EAP-TLS
3. EAP-TTLS
4. EAP-PEAP

EAP-MDS, EAS-TLS

1. EAP-MDS

✓ This is most basic of the EAP authentication methods.

✓ Here, the authentication server challenges the station to transmit the MD5 hash of the user's password.

✓ The station prompts the user to type his/her password.

✓ It then computes the hash of the password and sends this across.

✓ This method is insecure since an attacker could eavesdrop on such a message exchange and then replay the hashed password thus impersonating the owner of the password.

✓ Also, this method does not support authentication of the AP to the station.

2. EAP-TLS

✓ EAP-TLS is based on the SSL/TLS protocol

✓ most secure and provides mutual authentication and agreement on a master session key.

✓ It requires the AP as well as the user (station) to have digital certificates. ✓ It is relatively straightforward to equip each AP with a digital certificate and a corresponding private key but extending the via to each user of the WLAN may not be feasible.

EAP-TTLS, PEAP

3. EAP-TTLS

✓ (tunnelled TLS) requires certificates only at the AP end.

✓ The AP authenticates itself to the station and both sides construct a secure tunnel between themselves.

✓ Over this secure tunnel, the station authenticates itself to the AP.

✓ The station could transmit attribute-value pairs such as user_name = akshay

password = 4rP#mNaS&7

4 Protected EAP (PEAP)

✓ This was proposed by Microsoft, Cisco, and RSA Security, is very similar to EAPTTLS.

✓ In PEAP, the secure tunnel is used to start a second EAP exchange where in the station authenticates itself to the authentication server.

CONFIDENTIALITY

CONFIDENTIALITY

Data Protection in WEP (wired equivalent privacy).

➢ WEP was designed to provide message confidentiality, integrity, and access control but it failed on all three counts.

➢ In this section, we show how plaintext can be recovered and messages can be modified due to flawed design decisions in WEP. ➢ There are many lessons to be learned from WEP — the most important being how not to design protocols for security.

CONFIDENTIALITY

WEP Encryption

➢ WEP uses the stream cipher, RC4, for encrypting messages.

➢ It generates a pseudo-random keystream, KS, which is a function of a static secret shared between the two communicating parties.

➢ In order to have KS vary from message to message, a random per-message initialization vector, IV, is also used to generate KS.

➢ Early implementations of WEP used a 40-bit secret, S, concatenated with a 24-bit IV to create, in effect, a "64-bit key."

➢ KS is xored with the plaintext, P, to obtain the ciphertext, C

or $C = P \oplus KS(S, IV)$

CONFIDENTIALITY

- The plaintext includes
  - Message to be send
  - Integrity: which is a 32 bit checksum computed on the message.
  - The IV chosen by the sender is included in each frame as shown below

CONFIDENTIALITY

- The plaintext p is obtained as follows:

The receiver will generates KS from the shared secret Sand the IV retrieved from the received frame. It recovers the plain text from the following equation

$$\mathcal{P} = C \oplus KS\ (S, IV)$$

Known plaintext attack

➢ The first problem with WEP is the possibility of keystream re-use.

➢ Since the IV is 24 bits in length, there are only 224 distinct keystreams that could be constructed given a secret S.

➢ Suppose an attacker finds two frames which were encrypted using the same IV.

➢ Let their ciphertexts be C and C'.

➢ Let the corresponding plaintexts be P and P'. using

$$\mathcal{P} \oplus \mathcal{P}' = C \oplus C'$$

$$\mathcal{P}' = \mathcal{P} \oplus C \oplus C'$$

➢ Thus knowing c,c', and p, we can obtain p'
which is called as known plaintext attack.

SLO :1
Cellphone Security

Cellphone Security

→ One of the most widely used cellular networks is the Global System for Mobile Communications (GSM).

→ The goals of GSM or 2G are better quality for voice, higher speeds for data and other new applications.

→ The successor to GSM is Universal Mobile Telecommunications System (UMTS) or 3G.

→ It provides services such as mobile Internet, multimedia messaging, videoconferencing etc.

Entities Involved

→ At the lowest level, a cellphone is connected to a base station by a radio link.

→ Multiple base stations are, in turn, connected to and controlled by a base station controller.

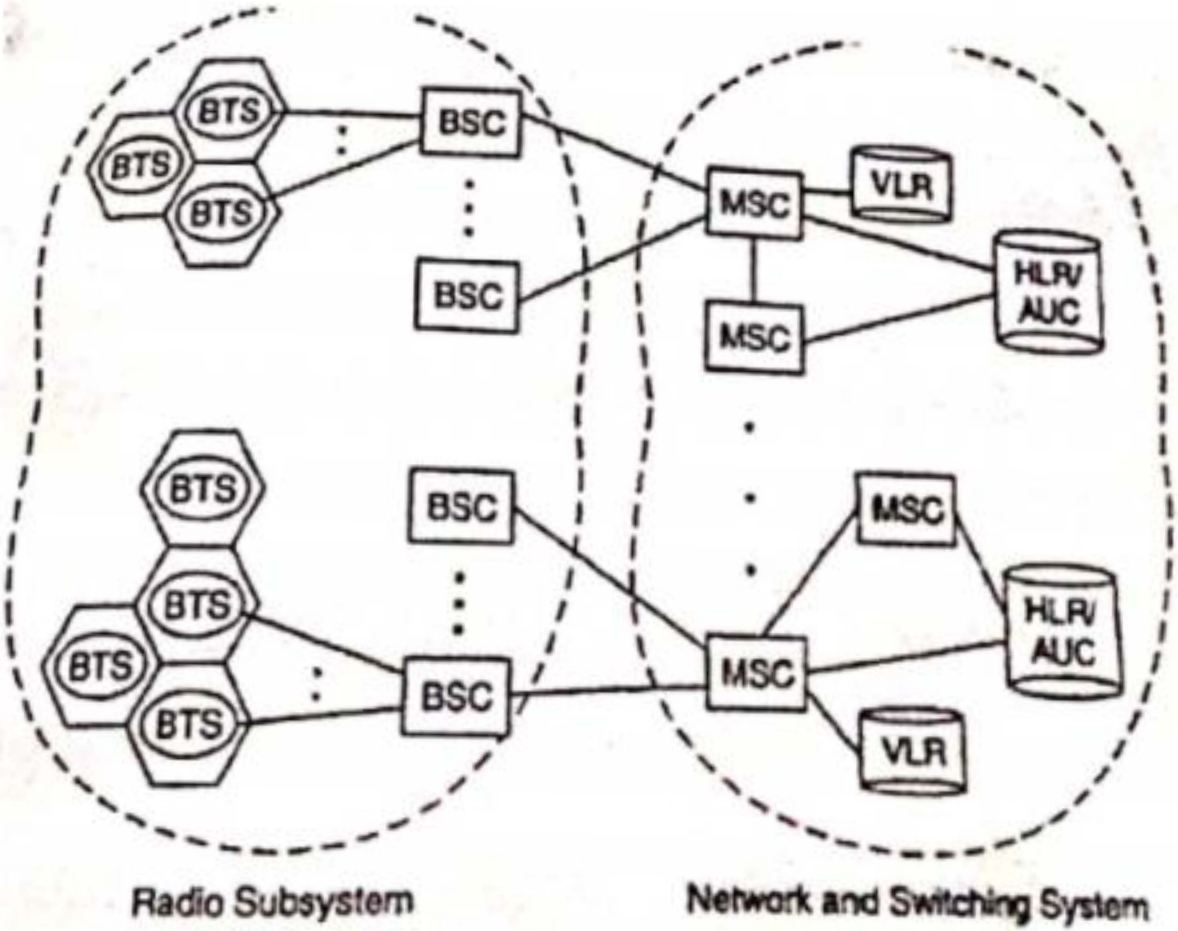
→ The connection between a base station and its controller could be a microwave link, optical link etc.

→ Multiple base station controllers are connected to a Mobile Switching Centre (MSC).

→ The MSC forwards an incoming call to the MSC where the call recipient is located.

→ The MSC also handles call billing and accounting functions.

→ MSCs are connected to each other through wired networks such as the Packet Switched Telephone network (PSTN).


BTS
BTS
BTS
BSC
BSC
MSC
VLR
HLR/
AUC
MSC
BTS
BTS
BTS
BTS
BSC
BSC
MSC
MSC
HLR/
AUC
VLR
Radio Subsystem
Network and Switching System

→ A user's home network is the one with whom the user has a subscription.

→ There is a one-to-one mapping between a network and an MSC.

→ An MSC has a database, called the Home Location Register (HLR), containing information about each of its subscribers.

→ This includes information such as the subscriber's mobile number, services subscribed to, and a secret key stored in the mobile and known only to the HLR.

→ HLR contains information regarding roaming subscribers, current location location of a subscriber etc.

→ A subscriber may avail services of other networks that have a roaming agreement with the subscriber's home network.

→ each cellular network maintains a database of users currently visiting that network together with the list of services that the subscriber is entitled to.

→ this database is referred to as a Visitor Location Register (VLR).

→ 2G technology introduced the idea of a Subscriber Identity Module (SIM) card.

→ It stores 3 secrets and performs cryptographic operations on these secrets.

→ the secrets are

- a unique 15-digit subscriber identification number called the International Mobile Subscriber Identity (IMSI)
- a 128-bit subscriber authentication key denoted K, known only to the SIM and the HLR of the subscriber's home network.
- a PIN known to the phone's owner and used to unlock the SIM, intended to prevent stolen phones being used.

Security Goals

→ The main security goals in GSM/UMTS are
- authentication, integrity and confidentiality.

User identity Confidentiality:- one way for the attacker to identify a caller is through the IMSI transmitted by the cellphone when a call is made.

→ to protect user privacy, GSM requires that the IMSI be used, during initial authentication to a foreign network.

→ a temporary Mobile Subscriber Identity (TMSI) is assigned to a user.

→ this has a limited-time validity.

→ when a user changes location and moves to a new network, the user's cellphone will have to be re-authenticated & a new TMSI assigned.

Message Confidentiality:- one of the most important security goals is to protect the confidentiality of user messages so that an attacker will not be able to make sense of the contents of messages.

Entity Authentication :- The MSC needs to be sure that the call is billed to the person making the call.

Message origin authentication and message integrity

→ For each signaling message between the cellphone and the base station, the recipient needs to verify that the message has been received without error.

→ Both parties should be able to verify that each message was indeed created by the party at the other end and not by an attacker.

SLO :2
GSM (2G) Security

GSM (2G) Security

→ There are two principal tasks involved in providing security in GSM.

a) entity authentication and key agreement

b) message protection.

1) Entity Authentication and Key Agreement

The main steps in authentication are

Step 1: Authorization Request from Cellphone

→ the cellphone sends to the base station the encryption algorithms that it can support.

→ It also sends its IMSI/TMSI to the MSC.

→ If the cellphone is away from its home network, the IMSI will be received by the MSC of the visited network.

→ The latter communicates the IMSI to the MSC/HLR of the cellphone's home network with a request to provide a challenge that

will be used by the cellphone to authenticate itself.

Step 2: Creation and Transmission of Authentication Vectors

→ The MSC for the home network receives the IMSI of the cellphone.

→ This is used to index into the HLR where it obtains the key K_i.

→ K_i is shared between a SIM and the HLR of its home network.

→ The MSC/HLR generates a 128-bit random number, RAND, which functions as the challenge in the challenge-response authentication protocol.

→ SI computes two quantities XRES and Kc

$$XRES = A3(RAND, K_i)$$

$$Kc = A8(RAND, K_i)$$

→ A3 & A8 are two keyed hash functions.

→ XRES is the expected response to the challenge-response authentication protocol.

→ Kc is the encryption key.

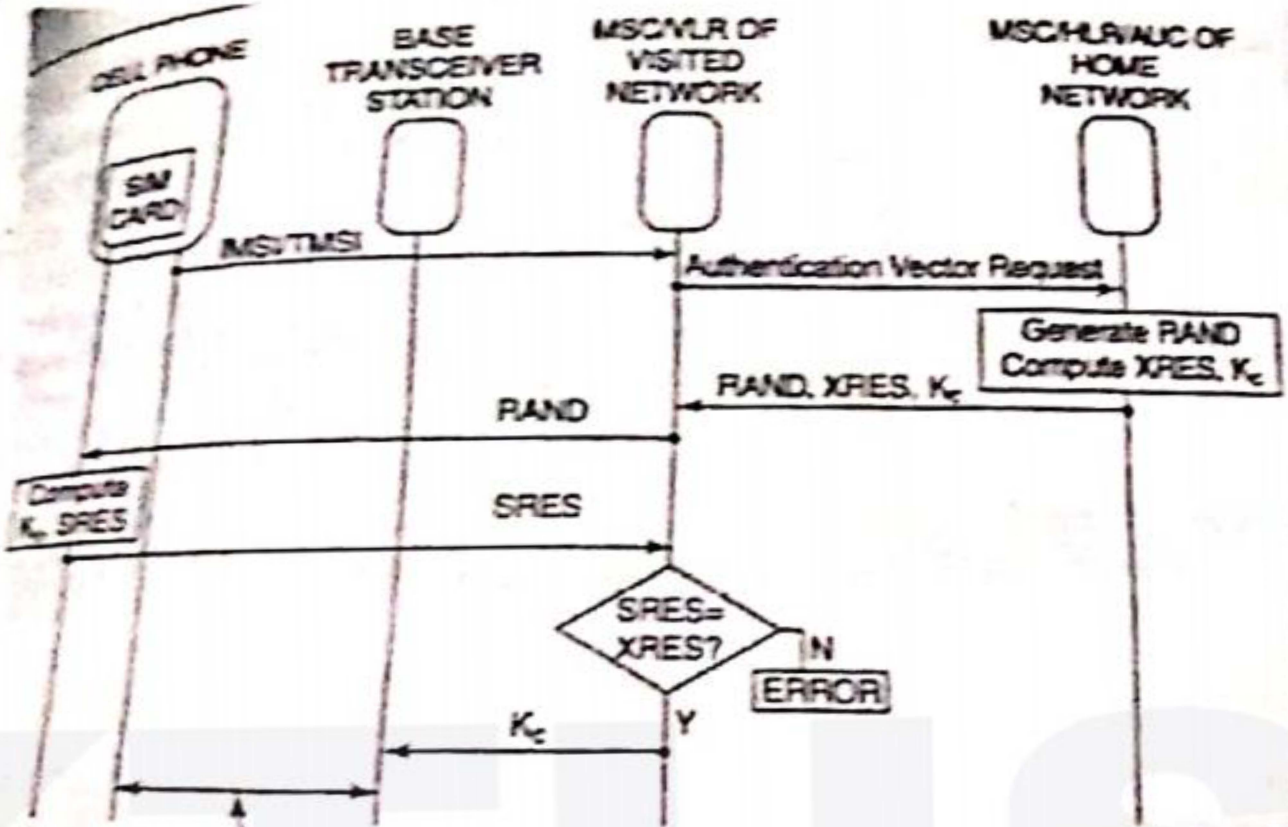
→ The HLR creates five authentication triplets, each triplet is

$$\langle RAND, XRES, Kc \rangle$$

→ The triplets are sent to the MSC of the home network by the HLR.

→ If the cellphone is visiting a "foreign" network the MSC forwards the triplets to the MSC of the visited network.

→ The MSC then sends the challenge (RAND) from the first triplet to the base station who forwards it to the SIM on the cellphone



Step 3: Cellphone Response

→ Once the SIM has received RAND, it proceeds to compute SRES.

→ SRES stands for "signed response".

→ The cellphone sends SRES to the base station who forwards it to the MSC

→ The MSC checks if SRES is equal to XRES - its expected response.

→ If they are equal the MSC concludes that the SIM knows Ki and represents a genine subscriber

Step 4 : Computation / Receipt of Encryption key

→ The SIM computes Kc.

→ On the network side, the MSC extracts Kc from its authentication triplet and communicates it to the base station.

→ All user messages between the cellphone & the base station are encrypted using Kc.

Encryption

→ Encryption of message between the cellphone and the base station is performed by a stream cipher.

→ The key stream of this cipher is denoted by A5.

→ The keystream is a function of the 64-bit encryption key, Kc, and a 22-bit frame buffer.

KEYSTREAM = A5 (Kc, FRAME#)

→ The frame buffer is regenerated for each frame transmitted.

→ So the keystream changes for each frame sent during a call.

→ In most stream ciphers, the ciphertext is the ⊕ of the plaintext and the keystream.

Problems And Drawbacks

→ The algorithms A_3, A_5 and A_8 are based on Comp-128, a keyed hash function.

→ There have been several attacks on A_3 and A_8 to get the value of K_i;

→ With access to the SIM, one can obtain K_i using a side channel attack that involves 8 chosen plaintexts.

→ Once K_i is known the SIM can be cloned thus defeating one of the security goals of GSM.

→ Several versions of A5 such as A5/1, A5/2 and A5/3 are in use.
→ There were several successful attacks on all versions of A5.
→ By eavesdropping the first two minutes of conversation, a ciphertext-only attack on A5/2 can reveal the encryption key in a few milliseconds.
→ The SIM authenticates itself to the network but authentication of the network to the SIM is not part of the GSM protocol.
→ This could result in a false base station attack in which an attacker poses as a base station by sending more powerful beacon signals than the base station.
→ In such cases the attacker can spoof a cipher mode command from the base station.
→ The attacker's cipher mode command instructs the cellphone to suppress encryption.
→ So the attacker can easily trap the connection.

SLO : 1 & SLO : 2
Security in UMTS (3G)

UMTS- Universal Mobile Telecommunications System

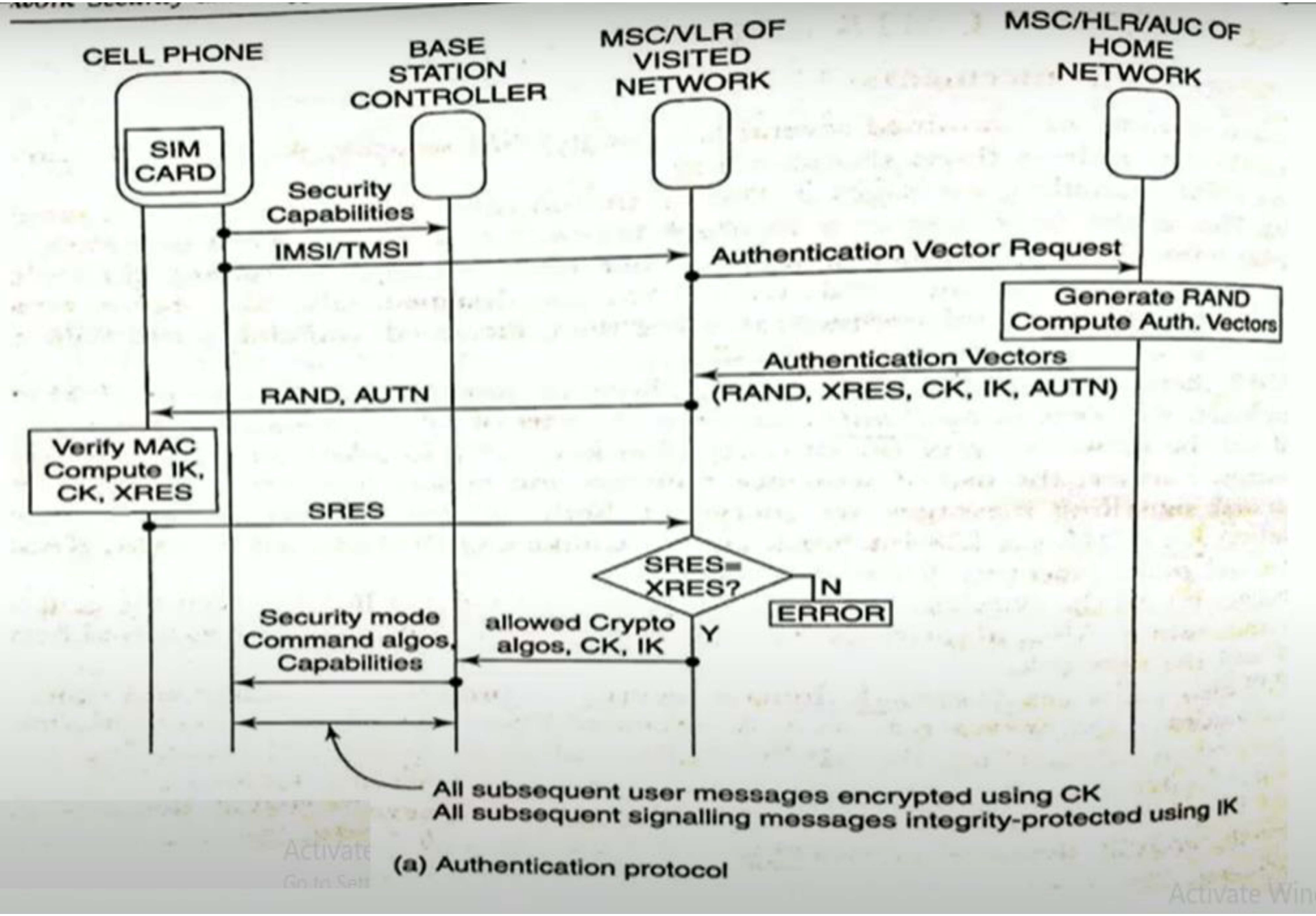
- UMTS is a 3G mobile cellular system for networks based on the GSM standard.
- It provide services such as Mobile Internet, Multimedia Messaging, video conferencing etc.

Security Enhancements in UMTS / 3G

- Signalling messages in UMTS are individually authenticated and integrity protected. Hence , the false base station attack is not possible in UMTS.
- UMTS supports mutual authentication. The SIM card and the network agree on an encryption key and also a key for integrity protection of messages.

UMTS

- Use of sequence numbers and nonces prevent replay attacks in 3G.
- Data and signalling messages are encrypted. Both integrity protection and encryption are based on a KASUMI- a128 bit block cipher which had withstood public scrutiny for several years.
- Messages on all the wireless links are encrypted.
- Algorithms for encryption and integrity may be negotiated between SIM and network.
- UMTS also addresses network domain security protecting signalling and other data in the provider domain.
- UMTS security architecture is carefully designed to maximize compatibility with GSM.



(a) Authentication protocol

Authentication and Key agreement

Step 1:- authorization request from cell phone

- The mobile station and the base station to establish a Radio Resource Control connection (RRC connection). During the establishment of the connection the mobile station sends its security capabilities to the base station.
- Security features include UMTS integrity and encryption algorithms supported and possibly GSM encryption capabilities as well.
- The mobile station sends its temporary identity TMSI current on the network.
- If the network cannot solve the TMSI, he asks the mobile station to send its permanent identity and the mobile stations responding to the request with the IMSI.
- The visited network requests authentication of the home network of the mobile station data.

Step 2:- creation and transmission of authentication vector

- HLR generates a random number 'RAND', which function as the challenge in the *challenge response authentication protocol*
- It also computes MAC, authentication token AUTN, anonymity key AK, integrity check key IK and cipher key CK.
- The keys and XRES are derived using derived hash functions F2, F3,F4 and F5.
  - XRES = F2(RAND, Ki)
  - CK = F3(RAND, Ki)
  - IK = F4(RAND, Ki)
  - AK = F5(RAND, Ki)
  - MAC = F1(RAND, Ki, AMF, SQN)

(AMF → Authentication management field. Life time of key, SQN → sequence number)

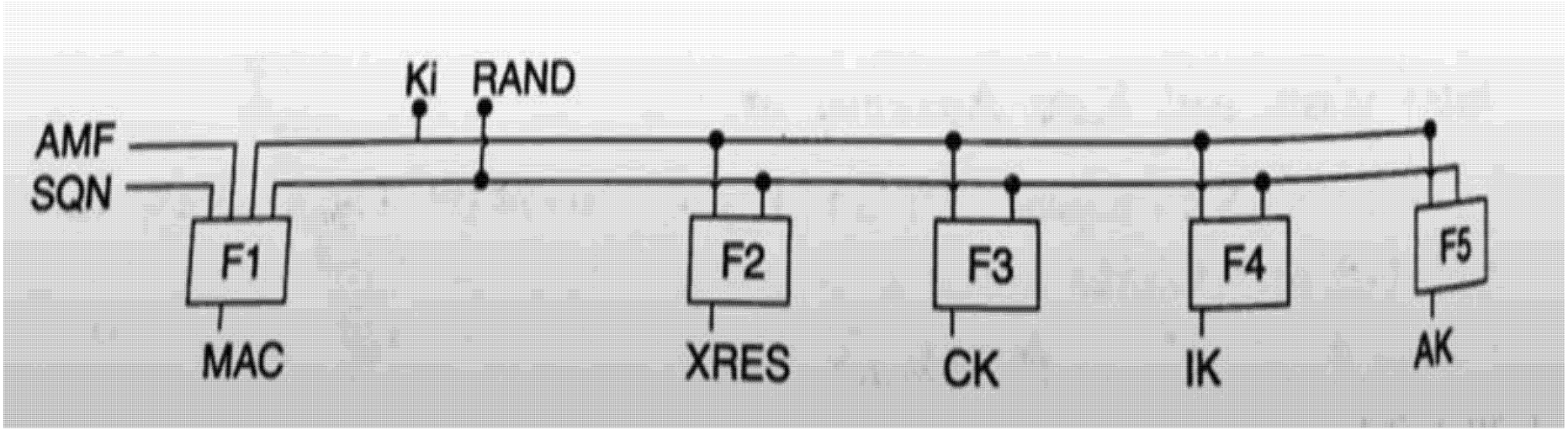
- HLR then creates a Authentication token

  AUTN = <SQN ⊕ AK, AMF,MAC>

- HLR then creates 5 Authentication vectors

  <RAND, XRES, CK, IK, AUTN>

- Authentication vectors are forwarded to MSC/VLR of visited network.
- MSC/VLR dispatches RAND and AUTN to BST which is forwarded to SIM.



Step 3:- verification of authentication Token and cellphone response

- SIM first calculate AK

  AK = F5(RAND, Ki)
- Then retrieve first element of

  AUTN = <SQN ⊕ AK, AMF,MAC> i.e, SQN ⊕ AK
- Then retrieve SQN by

  (SQN ⊕ AK) ⊕ AK
- Then check for difference

  computed SQN – stored SQN
- The result should be +ve and in acceptable range

- If SQN is acceptable then compute MAC

  MAC= F1(RAND, Ki, AMF, SQN)
- If computed MAC and received MAC are same, then SIM assume that
  - Vector is created by authenticated HLR
  - Vector is not a replay of previous one
- Then update SQN
- SIM compute SRES = F2(RAND, Ki)
- Then send SRES to MSC/VLR
- MSC/VLR compare "SRES == XRES", if same SIM is authenticated or else "ERROR"
- SIM computes CK and IK for all future messages

Step 4:- Agreement on encryption and integrity check algorithm

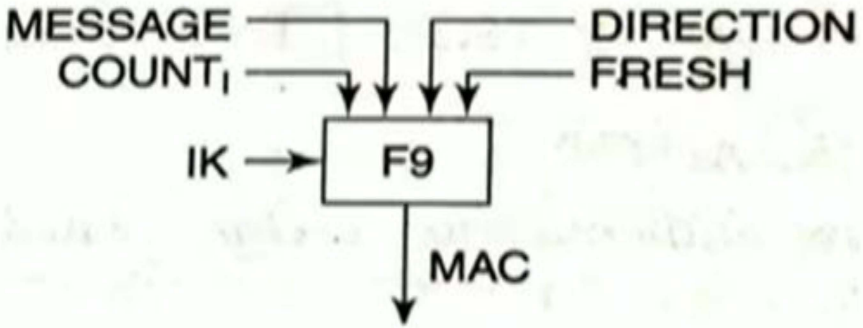
- MSC/VLR send a list of permitted MAC algorithms & encryption algorithms to BSC
- BSC decide which can be supported and send to cellphone
- Cellphone and BSC share CK and IK for encryption & integrity protection
- The mobile station confirms the protection of the integrity and verify the accuracy of the safety functions.

Integrity protection and encryption in UMTS

Integrity protection

MAC is used for

- Message origin authentication
- Integrity protection
- Protecting signaling messages



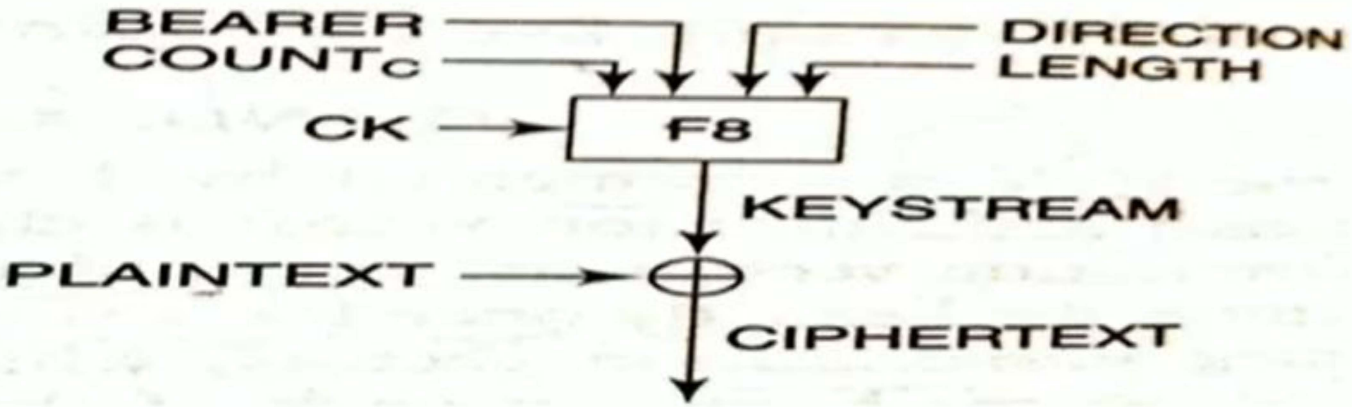
Per-message MAC is computed as

$$MAC = F9 (IK, COUNT_i, FRESH, DIRECTION, message)$$

- $COUNT_i$ → sequence number
- FRESH → a random number, to prevent "replay attack"
- DIRECTION → specify whether message was originated at "Cellphone or BSC"

Encryption

- Key stream = F8 (CK, $COUNT_C$, BEARER, DIRECTION, LENGTH)
- CK → Cipher key
- $COUNT_C$ → Frame count
- BEARER → radio channel indication
- F8 and F9 are based on KASUMI – an eight round fiestel cipher with 64-bit block size and 128-bit keys.



UMTS PROS & CONS

Advantages of UMTS

- Faster data rates.
- Support multimedia applications such as video and photography.
- Value added services like Mobile TV, GPS, Video Call & Video Conference.
- High speed mobile internet access.
- Increased capacity.

Disadvantages of UMTS

- Requires 3G compatible handsets.
- The cost of upgrading to 3G device is expensive.
- Power consumption is high.
- 3G requires closer base stations which is expensive.

SLO :1
Wireless LAN vulnerablities

Wireless LAN Vulnerabilities

→ In wireless LANs, there is much scope for misbehaving nodes to launch a variety of attacks.

Frame Spoofing

Premature Termination of Connections.

→ A station needs to authenticate and then associate with an Access Point (AP) before they can exchange data frames with each other.

→ At any point in time, either party can terminate the connection by transmitting a Deauthentication frame.

→ The recipient of a management frame relies on the Sender Address field in the frame to identify the originator of the message.

→ An attacker can spoof the Sender Address in the frame.

→ For ex: the attacker can fabricate a Deauthentication frame with

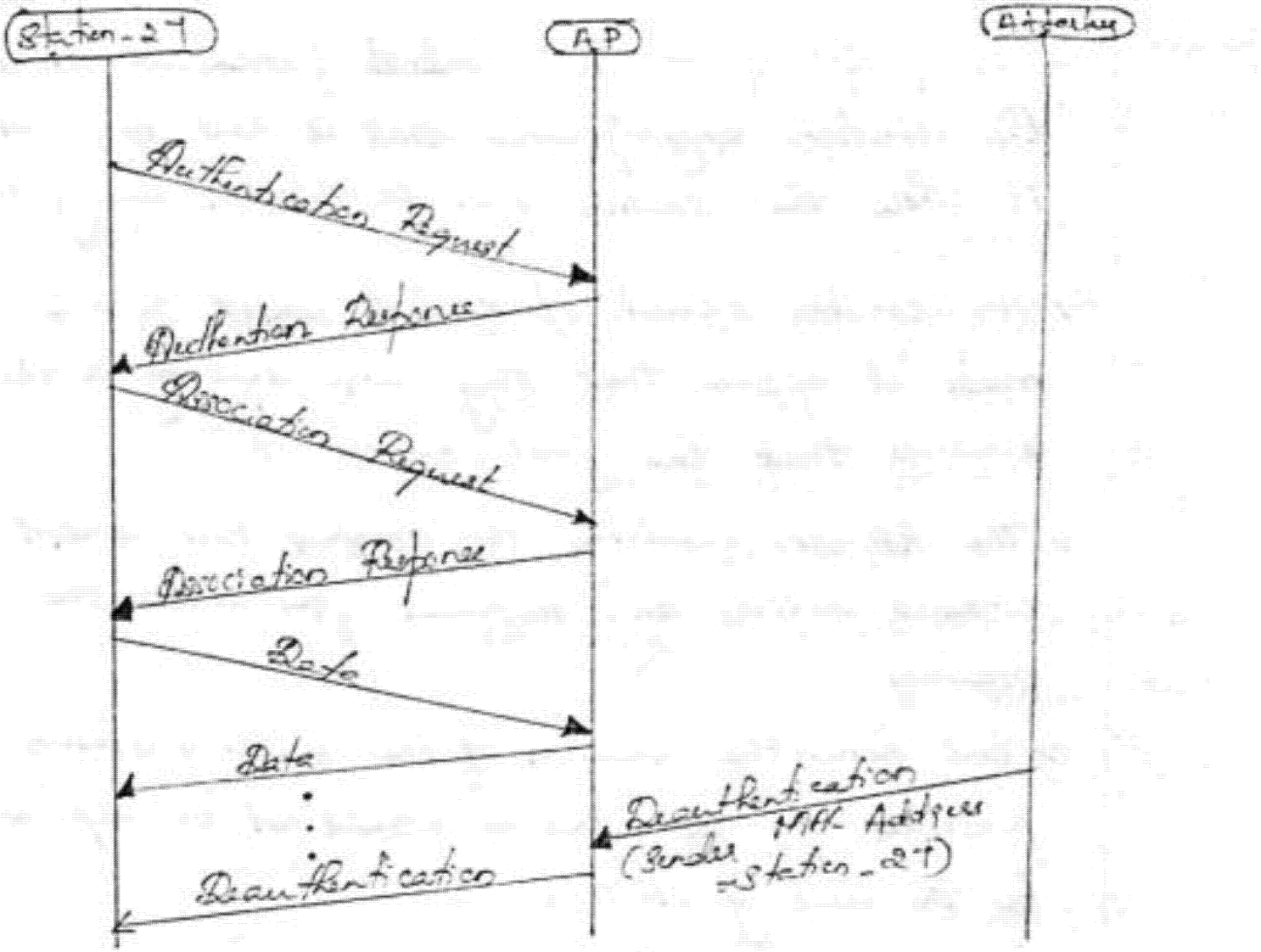
Sender Address = Station_27

Receiver Address = AP

→ The address used are 48-bit MAC address.

→ When the AP receives the above frame, it thinks that Station_27 wishes to terminate the existing connection to itself.

→ The AP sets the state of the connection between itself and Station_27 to be "Unauthenticated and Unassociated".



Spoofed Deauthentication Frame

→ Station-27 would have to go through the time-consuming process of de-authenticating and de-associating itself to the AP if it wished to resume the communication.

→ The attacker could repeatedly transmit such Deauthentication frames to the AP thus effectively slowing down or even preventing communications between station_27 and the AP.

Spoofing Power Management Control frames

→ A mobile station works on battery.

→ To save power, a mobile station powers off its transceiver.

→ It informs the AP that it is in power-saving mode so that the AP can buffer all frames intended for it.

→ When the station wakes up, it informs the AP that it is now in the active state using a Poll Control frame.

→ On receipt of the Poll Control frame, the AP delivers the station any frames that it had buffered for it while the station was in power-saving mode.

→ An attacker could spoof Poll Control frames and make it appear that they were sent by a sleeping station that has just woken up.

→ The AP on receiving the spoofed Poll Control frame, would deliver any buffered frames to the sleeping station.

→ But, since the receiver of the sleeping station is powered off, the frames would not be captured by the sleeping station.

→ When the sleeping station actually wakes up, it may send a Poll Control frame to retrieve frames buffered for it while it was asleep.

→ Since all the frames buffered for it have already been transmitted by the AP, it will not receive the frames destined for it while it was asleep.

SLO :2
Phishing

Vulnerabilities

- A vulnerability is a **weakness** or lacuna in a policy, procedure, protocol, hardware or software within an organization that has the potential to cause it damage or loss.

Vulnerability Types

- Human Vulnerabilities
  - Induced by careless/unthinking human behaviour
  - Ex. clicking on a link in an e-mail message from a questionable source
  - Related to **phishing** and cross-site scripting attacks

Vulnerability Types…

- Protocol Vulnerabilities
  - Attacks on commonly used networking protocols such as TCP, IP, ARP, ICMP and DNS
  - Ex. Connection hijacking caused by ARP spoofing, etc.
  - **Denial of Service Attacks** (DoS) which exploit the 3-way TCP handshake
  - Pharming attacks exploit vulnerabilities in DNS

Vulnerability Types…

- Software Vulnerabilities
  - Caused by sloppy software
  - Software may perform as expected under normal conditions but when provided with a specific input, it turns malicious
  - Examples include **Buffer Overflow** vulnerability, **Cross-site Scripting** (XSS) vulnerability and **SQL Injection** vulnerability

Phishing

- Phishing is the fraudulent attempt to obtain sensitive information such as usernames, passwords and credit card details by disguising oneself as a trustworthy entity in an electronic communication.
- Typically carried out by email spoofing or instant messaging it often directs users to enter personal information at a fake website which matches the look and feel of the legitimate site.
- Email spoofing is one of the easiest types of phishing used to get data from users without their knowledge.

It can be done in different ways:

– Sending an email through a familiar username,

– Impersonating the identity of an organization and asking employees to share internal data.

Phishing

- Here is an example

From: **avlor.hr@gmail.com**

To: bruce@avlorcloud.info

Subject: Message from human resources

Dear Bruce,

An information document has been sent to you by the Human Resources Department. **Click here to Login** to view the document.

Thank you

HR Department

Avlorcloud University of California

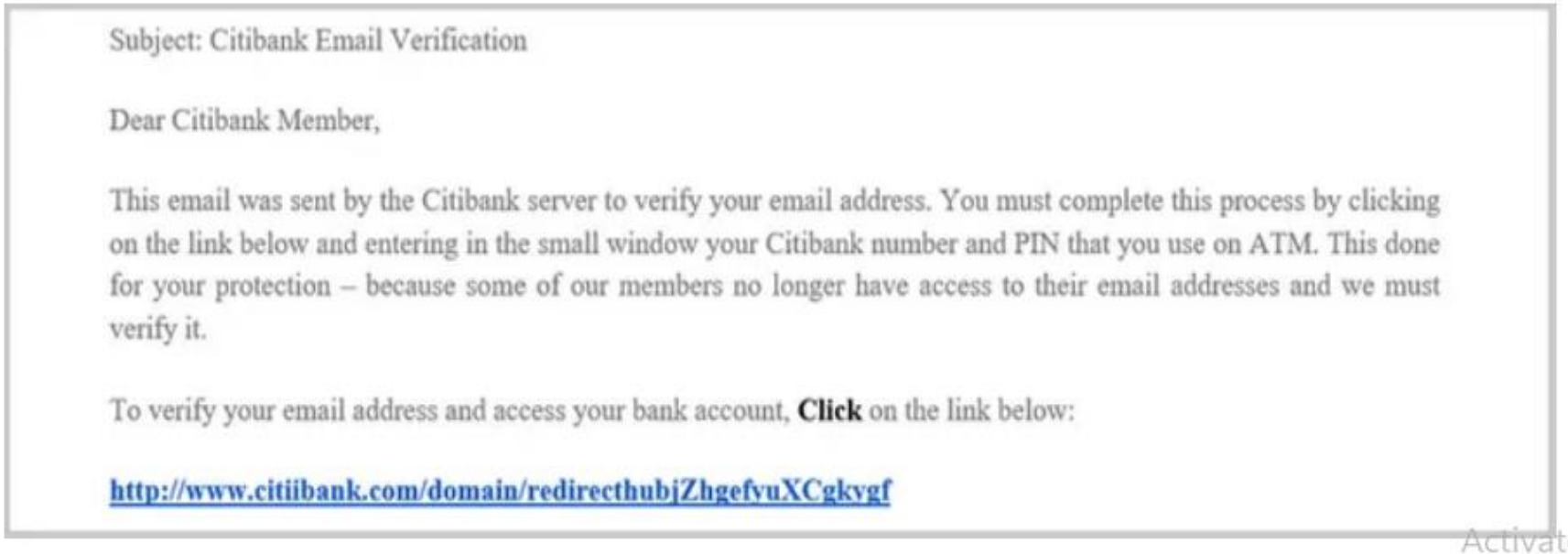
Just by seeing the company's name and the urgency of action, some users may click on the link.

Phishing

- How to prevent email phishing? The best way to prevent these attacks is by carefully reading the sender's email address.
- If you are not sure about the characters in an email address, then copy and paste it in the notepad to check the use of numeric or special characters.

Phishing

- Misspelled URL Hackers buy domains that sound similar to popular websites.
- Then, they phish users by creating an identical website, where they ask targets to log in by submitting personal information.
- In the example below, you can see that there's a typo in the link that people can easily miss: "www.citiibank.com…" instead of "www.citibank.com…



Subject: Citibank Email Verification

Dear Citibank Member,

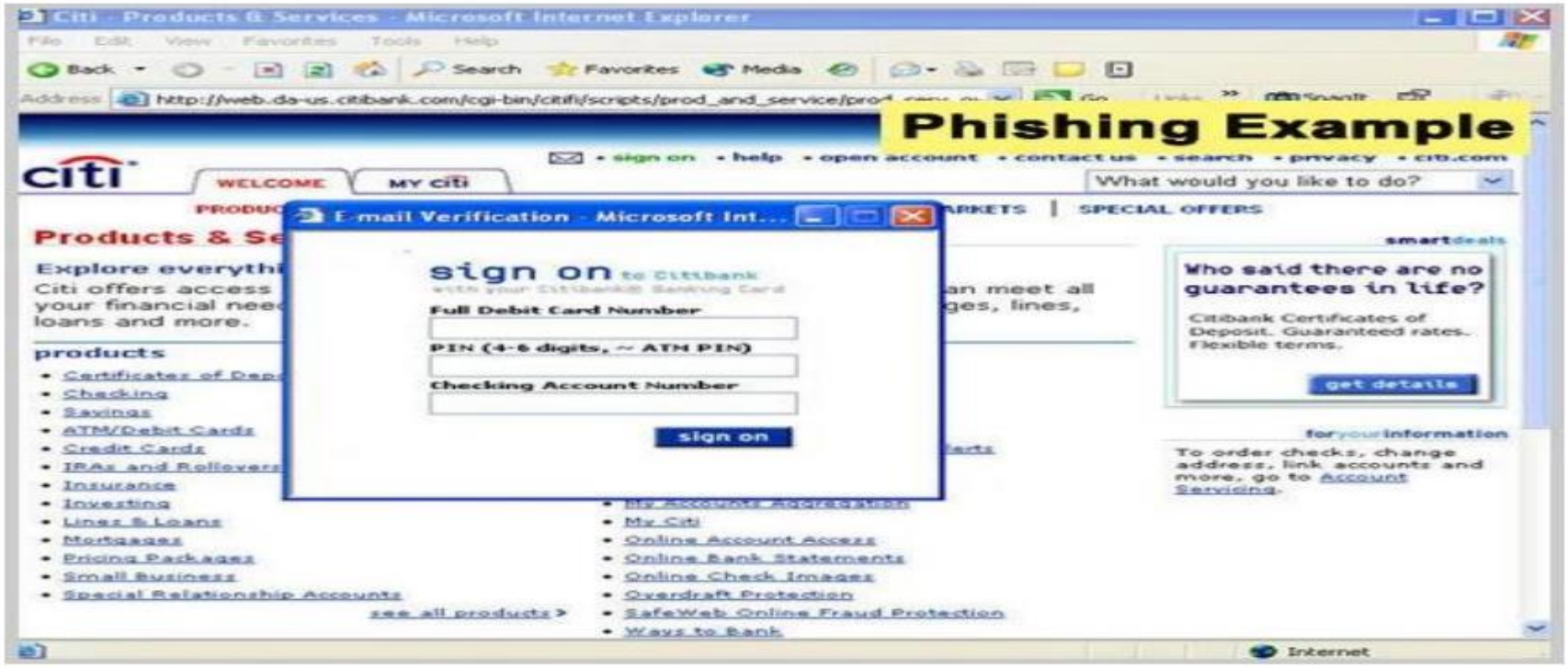
This email was sent by the Citibank server to verify your email address. You must complete this process by clicking on the link below and entering in the small window your Citibank number and PIN that you use on ATM. This done for your protection – because some of our members no longer have access to their email addresses and we must verify it.

To verify your email address and access your bank account, **Click** on the link below:

http://www.citiibank.com/domain/redirecthubjZhgefvuXCgkvgf

Phishing

- Pop-Up Messages: In-Session Phishing Pop-up messages are the easiest way to run a successful phishing. Through pop-up messages, attackers get a window to steal the login credentials by redirecting them to a fake website. This technique of phishing is also known as “In-session phishing.” Look at the pop-up window given below. In this example, doesn’t the foreground pop-up seem legitimate enough to mislead customers?



SLO : 1 & SLO : 2
Buffer Overflow

Buffer Overflow (BOF)

- The BOF vulnerability is one of the oldest and, by far, the most common of software vulnerabilities.
- As early as 1988, the **Morris worm** was one of the first to exploit this vulnerability.
- Since then, many creative ways of converting such a **vulnerability into an exploit** have been devised.

Buffer Overflow (BOF) …

- A buffer overflow (BOF) occurs when the space allocated to a variable (typically an array or string variable) is insufficient to accommodate the variable in its entirety.

- For example, a certain amount of buffer space is allocated for an array. If **array bounds are not checked** while populating it, the array may overflow into contiguous memory and corrupt it.

- Interestingly, this could cause an attacker to subvert the normal flow of a program. Malicious code supplied by the attacker in the buffer could be executed.

Exploiting Stack Overflows

- Provide input to a buffer on the stack which includes malicious code (often called **shellcode**)
- Overflow the buffer so that the **return address** to the calling program **is overwritten** with the address of the malicious code
- That way, when the called function terminates, it will not return to the calling program. Instead, the malicious code will be executed

Buffer Overflow Defences

There are many defences against BOF.

Some of the best known are

- Make the **stack non-executable**. This prevents malicious code on the stack from being executed. However, exploits like return into LibC are still possible
- Compiler-based option: Place a "**canary variable**" on the stack between the local variables and the return address. If a BOF modifies the return address, the canary will be corrupted. This will be detected by the compiler and the program will be aborted.

Related Attacks

- **Heap Overflow:** A program's dynamically allocated variable are stored on the heap. Buffers in this area may also be overflown leading to Heap buffer overflow attacks.

- **Format String Attacks**:

  C language printf( ), for example, uses a format string as function parameter. An attacker may pass a malicious string as input parameter enabling the attacker to read or write arbitrary locations in memory.

SLO : 1 & SLO : 2
Format String Attacks, Cross-site Scripting(XSS)

Format String

Format String Attacks:

C language printf( ), for example, uses a format string as function parameter. An attacker may pass a malicious string as input parameter enabling the attacker to read or write arbitrary locations in memory.

Cross-site Scripting Attacks

- A web site is said to have a cross-site scripting vulnerability if it inadvertently **includes malicious scripts** crafted by an attacker **in pages returned** by it.

- For example,

  <SCRIPT> Malicious Code </SCRIPT>

- The malicious code may, for example, read browser cookies on the victim's machine and ship these off to an attacker's web server

Persistent XSS Attack

- The malicious code (scripts) on a web page is saved on the web server.
- When an innocent user downloads the web page, the malicious scripts execute on that user's browser.
- Example: Users update their profile on a social networking site. These profiles may be read (downloaded) by other users through their browsers

Non-persistent XSS Attack

- Exploits the fact that some **servers echo back** certain user input back to the client without validating it
- For example, a user may be asked for personal details in an HTML form. Suppose he enters his name as “Prashant”. The server then responds with “Hello Prashant”
- Note that the server has echoed back his name
- Now, what would happen if, instead of Prashant, the user enters

  <SCRIPT>alert(‘Fire!’)</SCRIPT>

Overcoming XSS

- **Validate** and **filter** all user input. (Should this be done at the client or server?)

- One strategy is to make a **blacklist** of all user input that should be filtered out. For example, single/double quotes, angular brackets, etc. should not appear in an e-mail address input from the user.

- A better solution in most cases is the equivalent of a **whitelist approach** - specify precisely what user input is expected. This is often accomplished by the use of a **regular expression**.

SLO : 1 & SLO : 2
SQL Injection

SQL Injection

- SQL is standard query language for accessing and manipulating databases.

What does SQL do?

- Executes queries
- Insert update and delete record
- Create new database
- Create new tables
- Create stored procedures
- Create Views
- Set permission on tables, procedures, and views

SQL Injection

- SQL injection is a code injection technique, used to attack datadriven applications, in which malicious SQL statements are inserted into an entry field for execution.
- This is a method to attack web applications that have a data repository.
- The attacker would send a specially crafted SQL statement that is designed to cause some malicious action.

Attack Intent

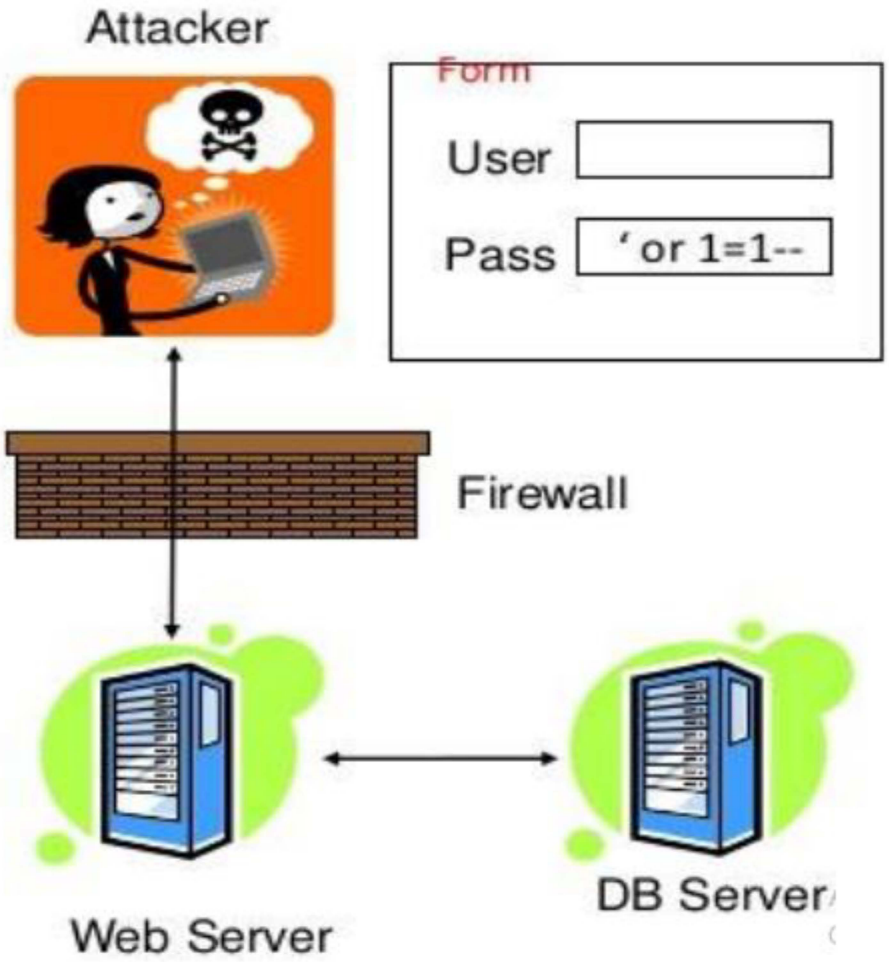
Determining database schema

- Extracting data
- Adding or modifying data
- Bypassing authentication
- In August 17, 2009, the United States Justice Department charged an American citizen Albert Gonzalez and two Russians with the theft of 130 million credit card numbers using an SQL injection attack.

SQL Injection

How SQL Injection works?

1. App sends form to user.
2. Attacker submits form with SQL exploit data.
3. Application builds string with exploit data.
4. Application sends SQL query to DB.
5. DB executes query, including exploit, sends data back to application.
6. Application returns data to user.



SQL Injection

- Form parameters may be passed as a **query string** in an extended URL to the server as in
  www.iitb.ac.in?s_ID=08935710&passwd=4ep*NdF

- The server application retrieves the form parameters and uses them to build an **SQL query** such as

  select *s_ID, gpa*
  from students09
  where *s_ID* = 08935710 **and** *passwd* = '4ep*NdF'

Constructing an SQL query directly from user input (Example 1)

select *s_ID, gpa*

from students09

where *s_ID* = 123 **and** *passwd* = 'abc' or 'x' = 'x'

Constructing an SQL query directly from user input (Example 2)

select *s_ID, gpa*

from students09

where *s_ID* = 123 or 1=1 - - **and** *passwd* = ' abc '

Constructing an SQL query directly from user input (Example 3)

select *s_ID, gpa*

from students09

where *s_ID* = 123; DROP TABLE students09; - - **and** *passwd* = ' abc '

SQL Injection

Defence Against SQL Injection

1. Comprehensive data sanitization

- Web sites must filter *all* user input
- For example, e-mail addresses should be filtered to allow only the characters allowed in an e-mail address.
- Its SQL injection defenses can catch most attempts to sneak SQL through web channels.

SQL Injection

Defence Against SQL Injection

2. Use a web application firewall

- A popular example is the free, open source module ModSecurity.
- ModSecurity provides a sophisticated and ever-evolving set of rules to filter potentially dangerous web requests.

SQL Injection

Defence Against SQL Injection

3. Limit database privileges by context

- Create multiple database user accounts with the minimum levels of privilege for their usage environment.
- For example, the code behind a login page should query the database using an account limited only to the relevent credentials table.
- This way, a breach through this channel cannot be leveraged to compromise the entire database.