Introduction to Computer Security

Chapter 22: Internet Security Protocols and Standards

Chi-Yu Li (2020 Spring)
Computer Science Department
National Chiao Tung University

Outline

- Secure E-mail
- DomainKeys Identified Mail
- Secure Sockets Layer (SSL) and Transport Layer Security (TLS)
- HTTPS (HTTP over SSL)
- IPv4 and IPv6 Security

Secure E-mail

MIME

- An Internet mail format: an extension to the old RFC 822 specification
  - RFC 822 defines a simple heading with To, From, Subject
  - Assumes ASCII text format
- Provides new header fields
  - Defining information about the message body

S/MIME

- Secure/Multipurpose Internet Mail Extension
- Security enhancement to the MIME Internet e-mail format
  - Based on technology from RSA Data Security
- Provides the ability to sign and/or encrypt e-mail messages

MIME and S/MIME Message Examples

MIME

```
From: John Doe <example@example.com>
MIME-Version: 1.0
Content-Type: multipart/mixed;
        boundary="XXXXboundary text"

This is a multipart message in MIME format.

--XXXXboundary text
Content-Type: text/plain

this is the body text

--XXXXboundary text
Content-Type: text/plain;
Content-Disposition: attachment;
        filename="test.txt"

this is the attachment text

--XXXXboundary text--
```

S/MIME

```
Content-Type: multipart/signed;
protocol="application/pkcs7-signature"; micalg="sha1";
boundary="----Alexis"

------Alexis
Content-Type: text/plain

This is an example of a clear signed message.

------Alexis
Content-Type: application/pkcs7-signature; name="smime.p7s"
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename="smime.p7s"

MIHmBgkqhkiG9w0BBwKggdgwgdUCAQMxCTAHBgUrDgMCGjALBgkqhkiG9w0BBwEx
gbcwgbQCAQOAFFJz90NbzoEOUQOJxop1+azsEdH9MAcGBSsOAwIaMA0GCSqGSIb3
DQEBAQUABIGAmO1mvk4maqMgb+aYomDTO5fPnRK/0p8w4ExJnai+MknjHmo5VX99
J0zxf+myPuBn1ajZKxoDIRSHpeMp8oToAKUgWNQm7b/yAqRmtCQvmaeI7rnMULVU
cfQfsI0Hz/ujJO2wUUMjFKmLgo7Q/C7++JTJTad9+SADRcNnVEWGjMg=

------Alexis--
```

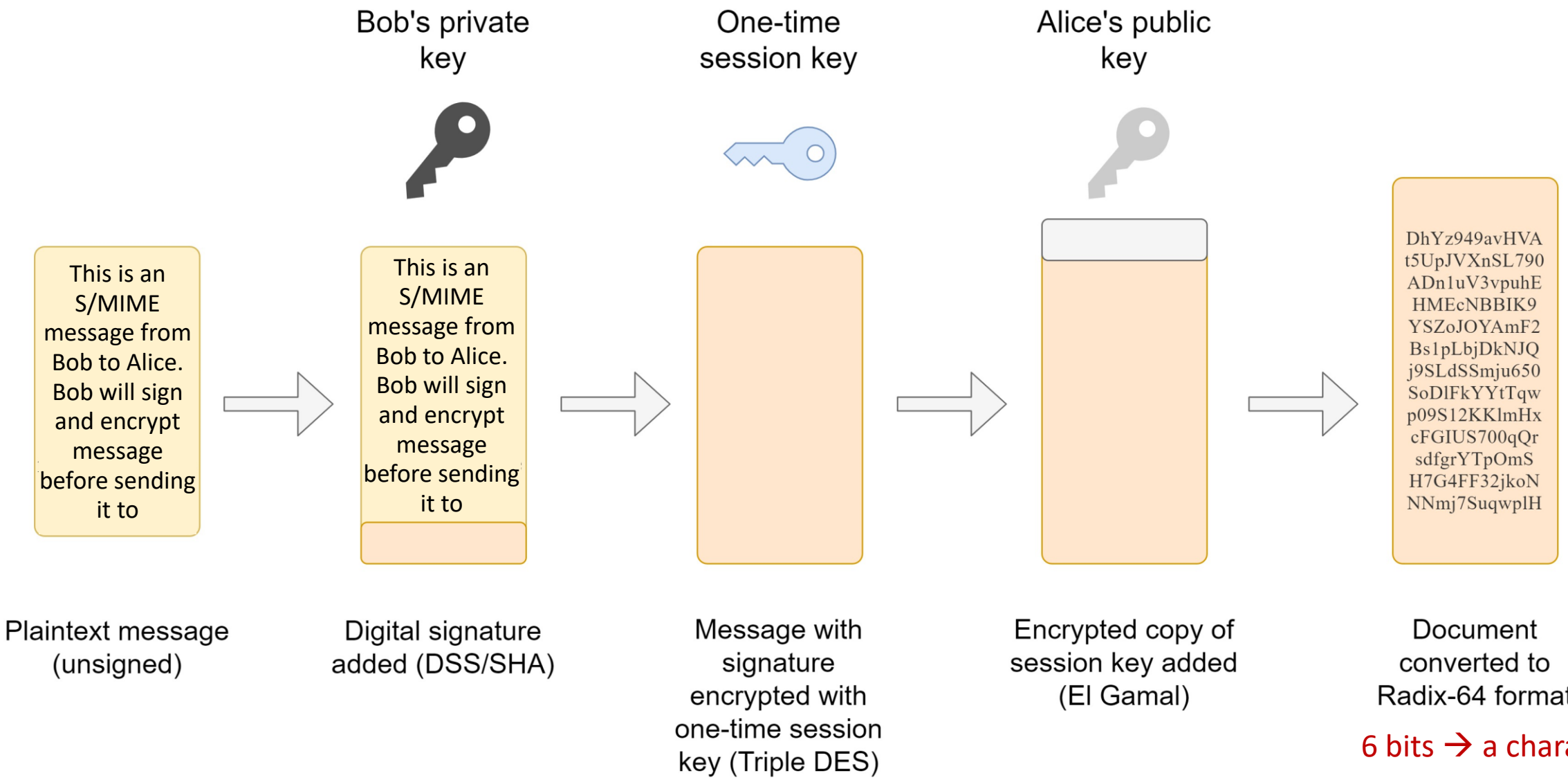
MIME Content Types

Type	Subtype	Description
Text	Plain	Unformatted text; may be ASCII or ISO 8859.
	Enriched	Provides greater format flexibility.
Multipart	Mixed	The different parts are independent but are to be transmitted together. They should be presented to the receiver in the order that they appear in the mail message.
	Parallel	Differs from Mixed only in that no order is defined for delivering the parts to the receiver.
	Alternative	The different parts are alternative versions of the same information. They are ordered in increasing faithfulness to the original, and the recipient's mail system should display the "best" version to the user.
	Digest	Similar to Mixed, but the default type/subtype of each part is message/rfc822.
Message	Rfc822	The body is itself an encapsulated message that conforms to RFC 822.
	Partial	Used to allow fragmentation of large mail items, in a way that is transparent to the recipient.
	External-body	Contains a pointer to an object that exists elsewhere.
Image	Jpeg	The image is in JPEG format, JFIF encoding.
	gif	The image is in GIF format.
Video	Mpeg	MPEG format.
Audio	Basic	Single-channel 8-bit ISDN mu-law encoding at a sample rate of 8 kHz.
Application	PostScript	Adobe Postscript.
	octet-stream	General binary data consisting of 8-bit bytes.

S/MIME Content Types

Type	Subtype	smime Parameter	Description
Multipart	Signed		A clear-signed message in two parts: one is the message and the other is the signature.
Application	pkcs7-mime	signedData	A signed S/MIME entity.
	pkcs7-mime	envelopedData	An encrypted S/MIME entity.
	pkcs7-mime	degenerate signedData	An entity containing only public-key certificates.
	pkcs7-mime	CompressedData	A compressed S/MIME entity.
	pkcs7-signature	signedData	The content type of the signature subpart of a multipart/signed message.

Typical S/MIME Process for Creating an S/MIME Message



6 bits → a character

S/MIME Functions

Enveloped data	Signed data	Clear-signed data	Signed and enveloped data
Encrypted content and associated keys	Encoded message + signed digest	Cleartext message + encoded signed digest	Nesting of signed and encrypted entities

Enveloped Data Using Public-key Infrastructure

- Default algorithms used for encrypting S/MIME messages are AES and RSA
  - M → AES(M) → AES(M) + RSA(K)
    - M: message
    - K: a pseudorandom secret key
    - Use recipient's public RSA key to encrypt K
- Public-key certificate: basic tool permits widespread use of S/MIME
  - Using certificates that conform to the international standard X.509v3

Signed and Clear-signed Data

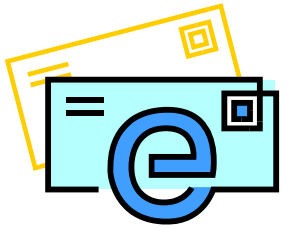
- Signed data ➔ Base64(content + sig)
- Clear-signed data ➔ content + Base64(sig)
- Default algorithms used for signing messages
  - DSS(SHA-1(Message), DSS-private-key)
  - DSS: Digital Signature Standard; SHA: Secure Hash Algorithm
- Alternative
  - RSA(SHA-1/MD5(Message), RSA-private-key)
- Radix-64/Base64: mapping content/sig into printable ASCII characters

Outline

- Secure E-mail
- DomainKeys Identified Mail
- Secure Sockets Layer (SSL) and Transport Layer Security (TLS)
- HTTPS (HTTP over SSL)
- IPv4 and IPv6 Security

DomainKeys Identified Mail (DKIM)

- A specification of cryptographically signing e-mail messages
  - A signing domain can claim responsibility for a message
  - RFC 4871 standard: DKIM signatures
- Has been widely adopted by a range of e-mail providers
  - e.g., gmail, yahoo, and many ISPs
- Why DKIM?



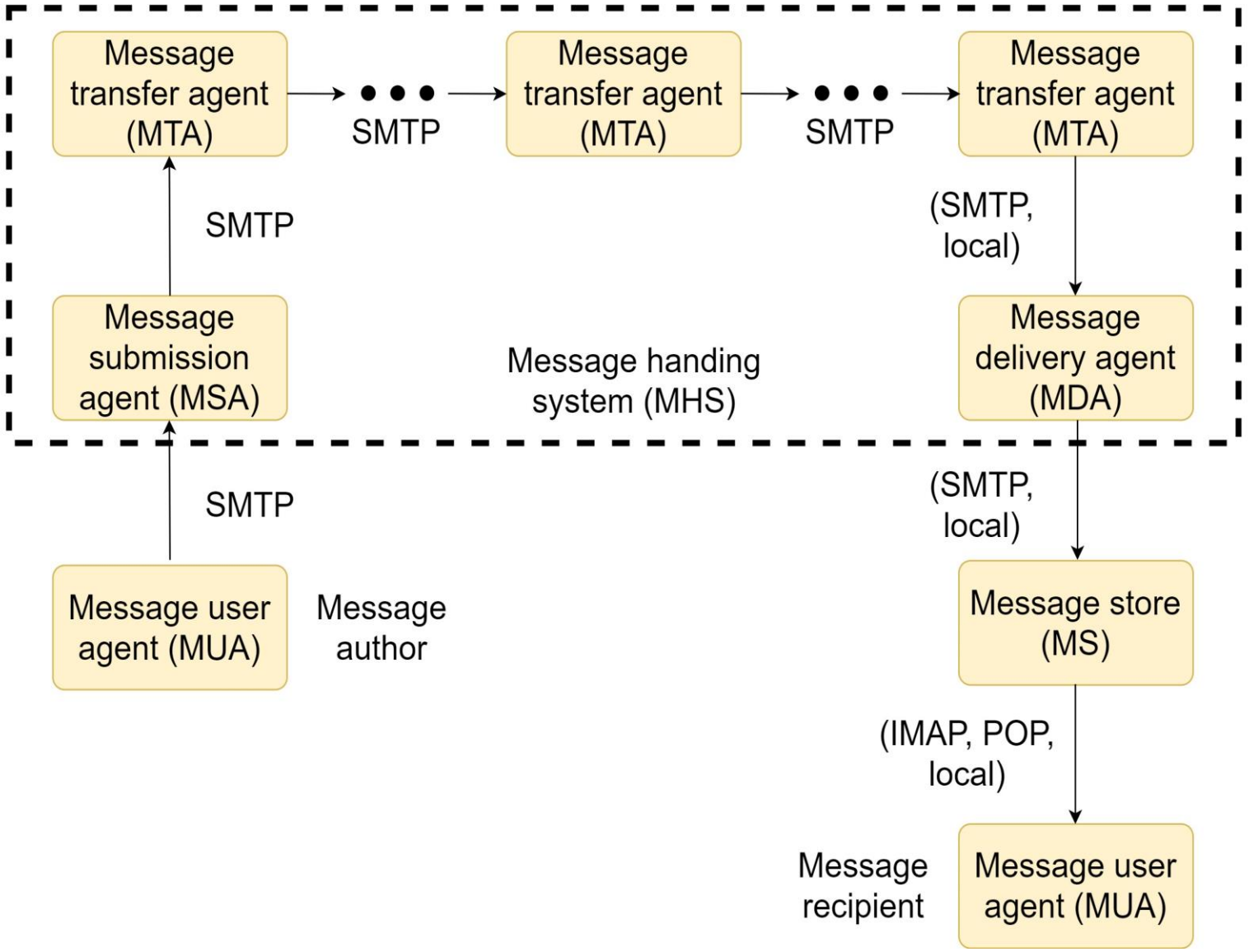
Why DKIM?

- An email authentication technique that is transparent to the end user

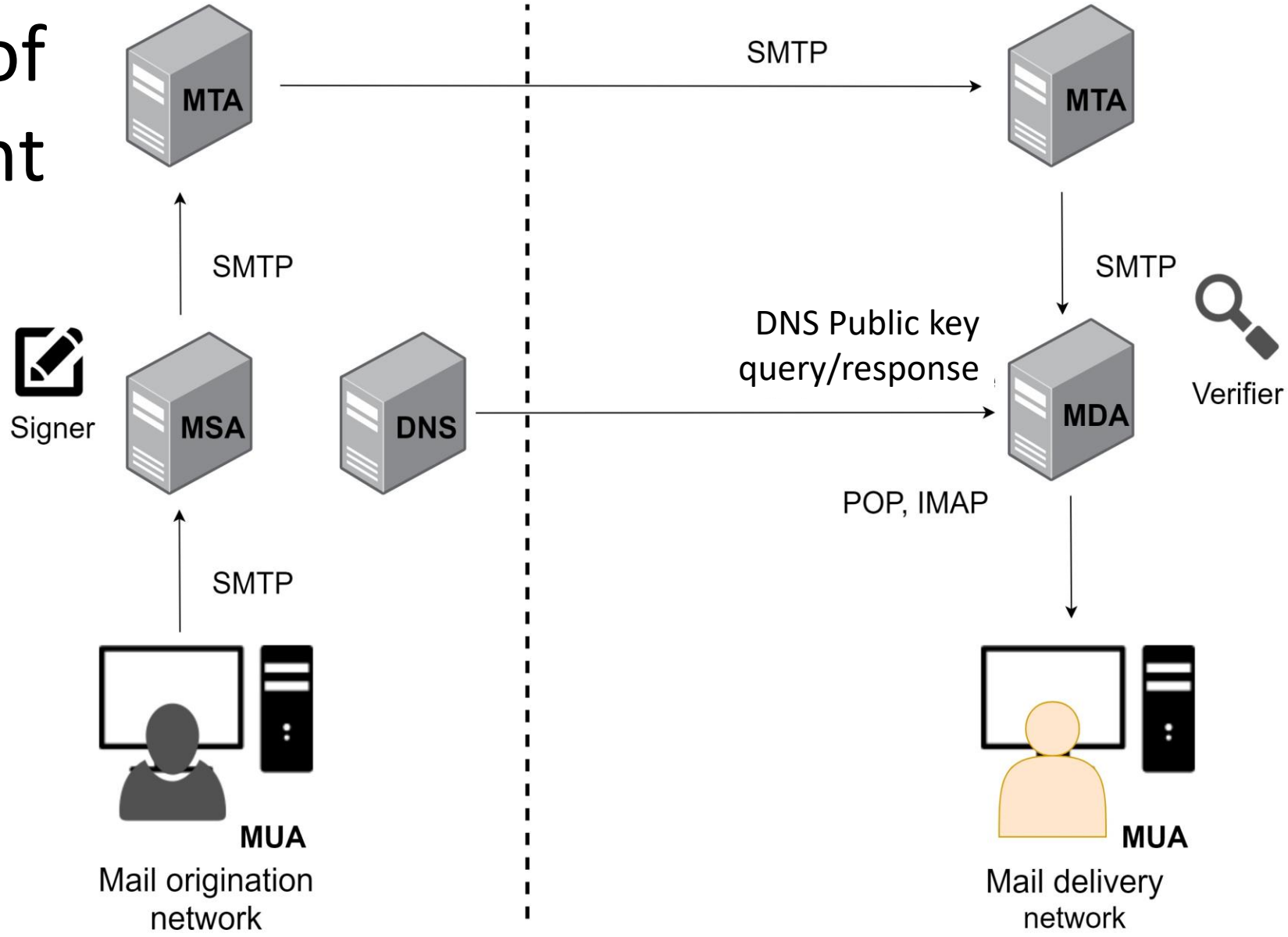
- Reasons
  - S/MIME depends on both the sending and receiving users employing S/MIME
    - But, the bulk of incoming mail does not use S/MIME
  - S/MIME signs only the message content
    - Header information concerning origin can be compromised
  - Transparent to the user; (s)he need take no action
  - Applied to all mails from cooperating domains
  - Preventing forgers from masquerading as good senders

Internet Mail Architecture

- User world: MUA
- Transfer world: MHS
  - Composed of MTAs



Simple Example of DKIM Deployment



DNS = domain name system
MDA = mail delivery agent
MSA = mail submission agent
MTA = message transfer agent
MUA = message user agent

Outline

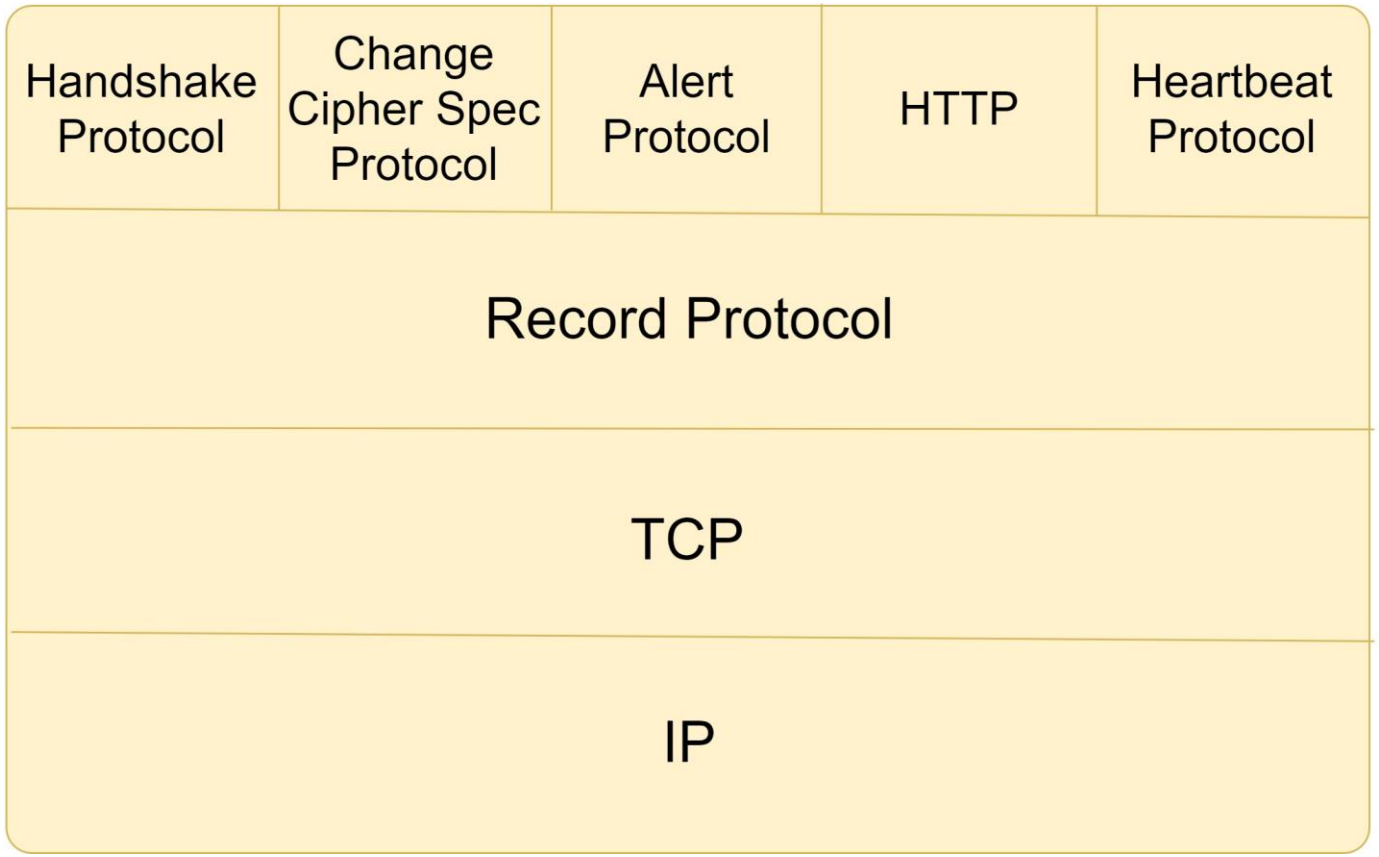
- Secure E-mail
- DomainKeys Identified Mail
- Secure Sockets Layer (SSL) and Transport Layer Security (TLS)
- HTTPS (HTTP over SSL)
- IPv4 and IPv6 Security

SSL and TLS

- Secure Socket Layer (SSL)
  - One of the most widely used security services
  - SSL3.0, SSL2.0 (deprecated in 2015)

- Transport Layer Security (TLS)
  - Becoming Internet standard RFC4346
  - General-purpose service: as a set of protocols that rely on TCP
  - TLS1.3, TLS1.2, TLS1.1 (new standards)
  - Two implementation choices
    - As part of the underlying protocol suite: transparent to apps
    - Be embedded in specific packages: e.g., most browsers come equipped with SSL

SSL/TLS Protocol Stack



TLS Concepts

TLS Connection

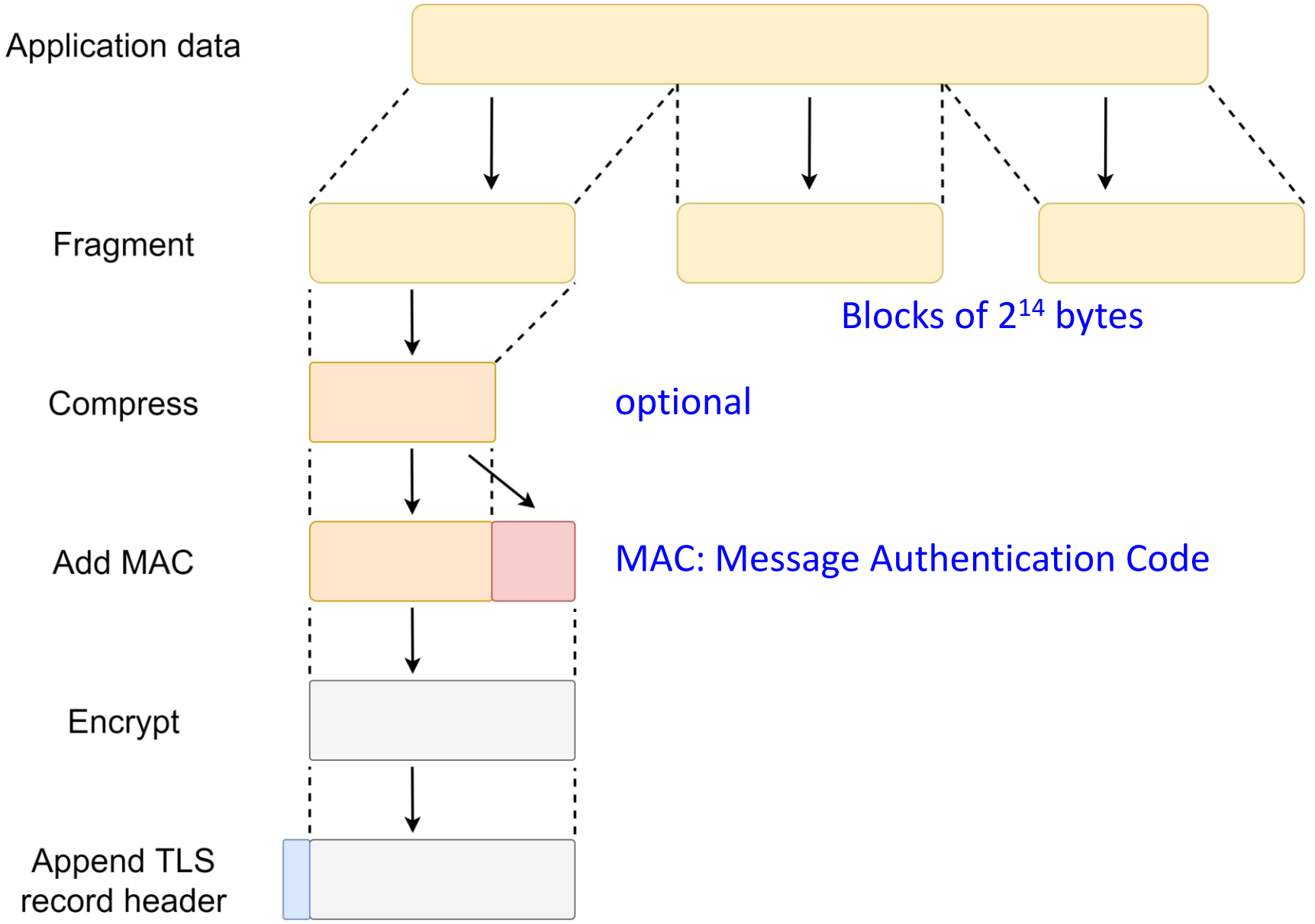
- A transport (in the OSI layering model definition) that provides a suitable type of service
- Peer-to-peer relationships
- Transient
- Every connection is associated with one session

TLS Session

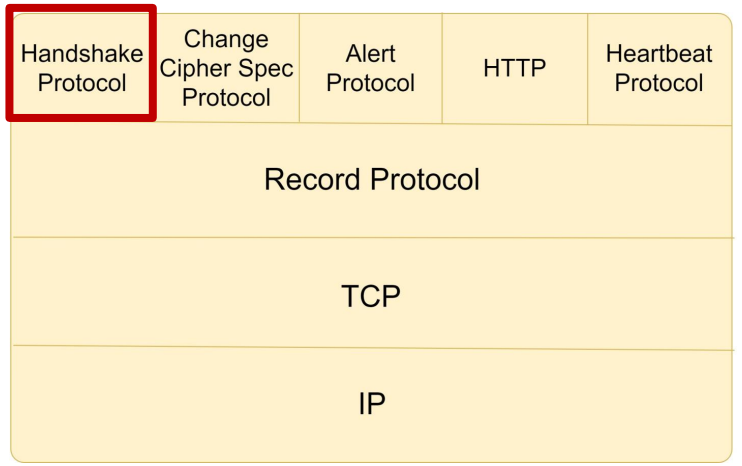
- An association between a client and a server
- Created by the handshake protocol
- Define a set of cryptographic security parameters
- Used to avoid the expensive negotiation of new security parameters for each connection

TLS Record Protocol Operation

- Two services
  - Confidentiality
  - Message integrity



Handshake Protocol

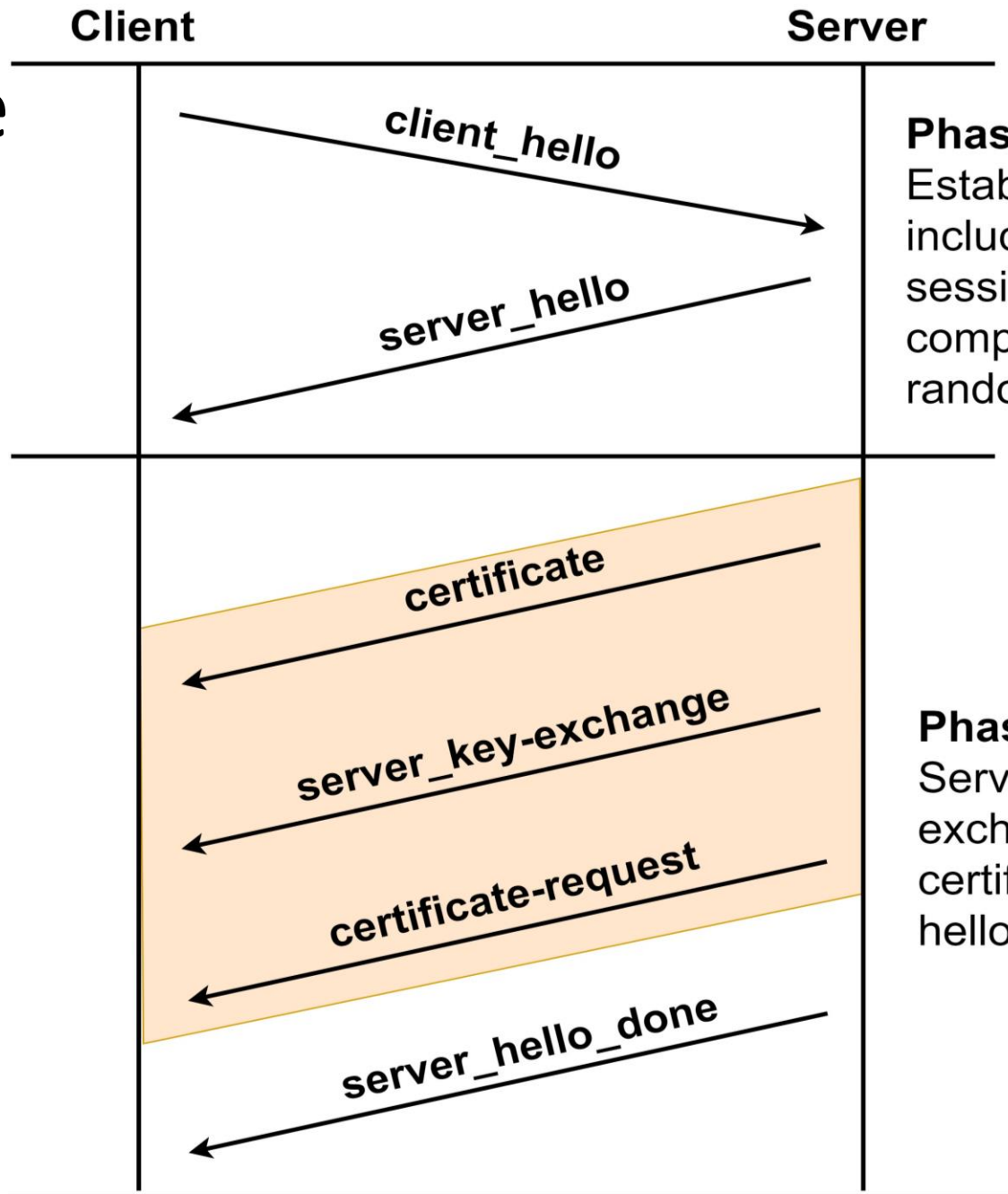


- Most complex part of TLS
- Used before any application data are transmitted
- Allows server and client to:

Authenticate each other → **Negotiate encryption and MAC algorithms** → **Negotiate cryptographic keys to be used**

- Comprises a series of messages exchanged by client and server
- Exchange has four phases

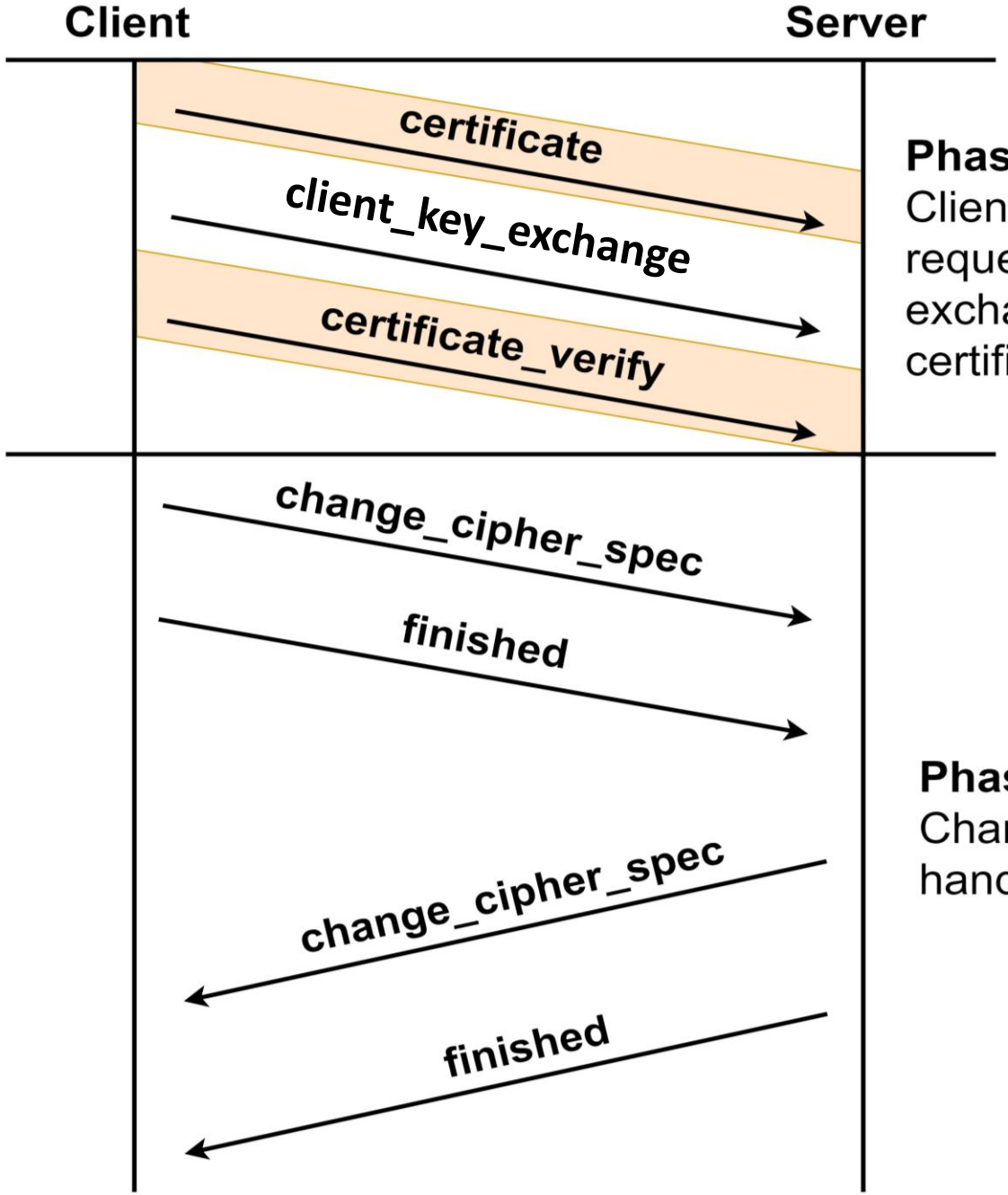
Handshake Protocol Action I



Phase 1
Establish security capabilities, including protocol version, session ID, cipher suite, compression method, and initial random number.

Phase 2
Server may send certificate, key exchange, and request certificate. Server signals end of hello message phase.

Handshake Protocol Action II



Phase 3
Client sends certificate if requested. Client sends key exchange. Client may send certificate verification.

Phase 4
Change cipher suite and finish handshake protocol.

ClientHello (RFC)

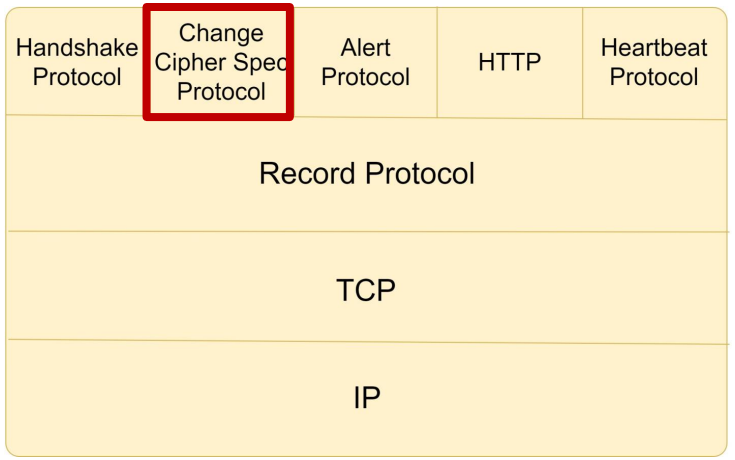
```
CipherSuite TLS_RSA_WITH_NULL_MD5                = { 0x00,0x01 };
CipherSuite TLS_RSA_WITH_NULL_SHA                = { 0x00,0x02 };
CipherSuite TLS_RSA_WITH_NULL_SHA256             = { 0x00,0x3B };
CipherSuite TLS_RSA_WITH_RC4_128_MD5             = { 0x00,0x04 };
CipherSuite TLS_RSA_WITH_RC4_128_SHA             = { 0x00,0x05 };
CipherSuite TLS_RSA_WITH_3DES_EDE_CBC_SHA        = { 0x00,0x0A };
CipherSuite TLS_RSA_WITH_AES_128_CBC_SHA         = { 0x00,0x2F };
CipherSuite TLS_RSA_WITH_AES_256_CBC_SHA         = { 0x00,0x35 };
CipherSuite TLS_RSA_WITH_AES_128_CBC_SHA256      = { 0x00,0x3C };
CipherSuite TLS_RSA_WITH_AES_256_CBC_SHA256      = { 0x00,0x3D };
```

- struct {
  - ProtocolVersion client_version;
  - Random random;
  - SessionID session_id;
  - CipherSuite cipher_suites <2..2^16-2>;
  - CompressionMethod compression_methods <1..2^8-1>;} ClientHello;

ServerHello (RFC)

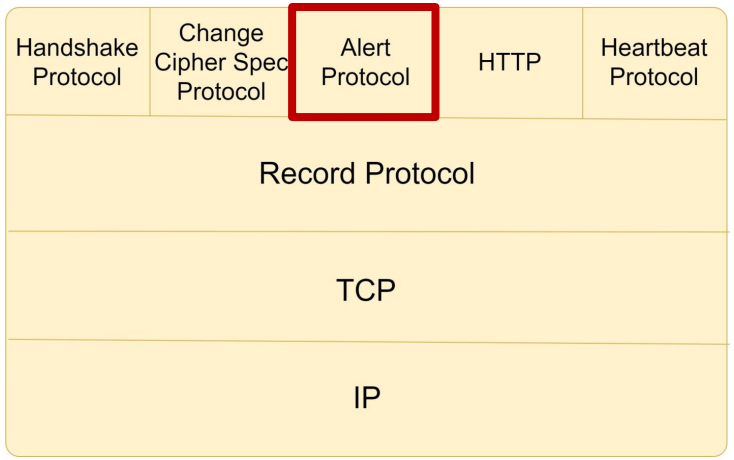
- struct {
  - ProtocolVersion server_version;
  - Random random;
  - SessionID session_id;
  - CipherSuite cipher_suite;
  - CompressionMethod compression_method;} ServerHello;

Change Cipher Spec Protocol



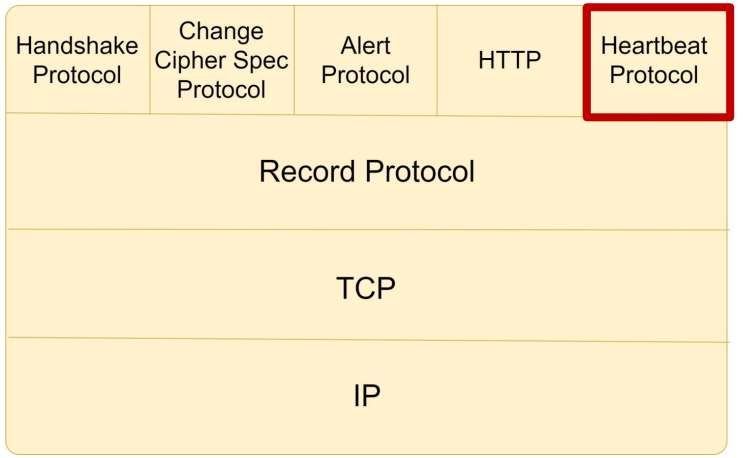
- One of four TLS specific protocols that use the TLS Record Protocol
  - the simplest
- Consists of a single message: a single byte with the value 1
- Purpose: cause pending state to be copied into the current state
  - Hence, updating the cipher suite in use

Alert Protocol



- Two bytes
  - First byte: warning (1), fatal (2)
    - For fatal, TLS implementation **terminates** the TLS connection.
    - For other TLS connections using the same TLS session, they may continue, but no new TLS connections may be established
  - Second byte: specific alert code (RFC5246-appendix)
    - close_notify(0), unexpected_message(10), etc.

Heartbeat Protocol

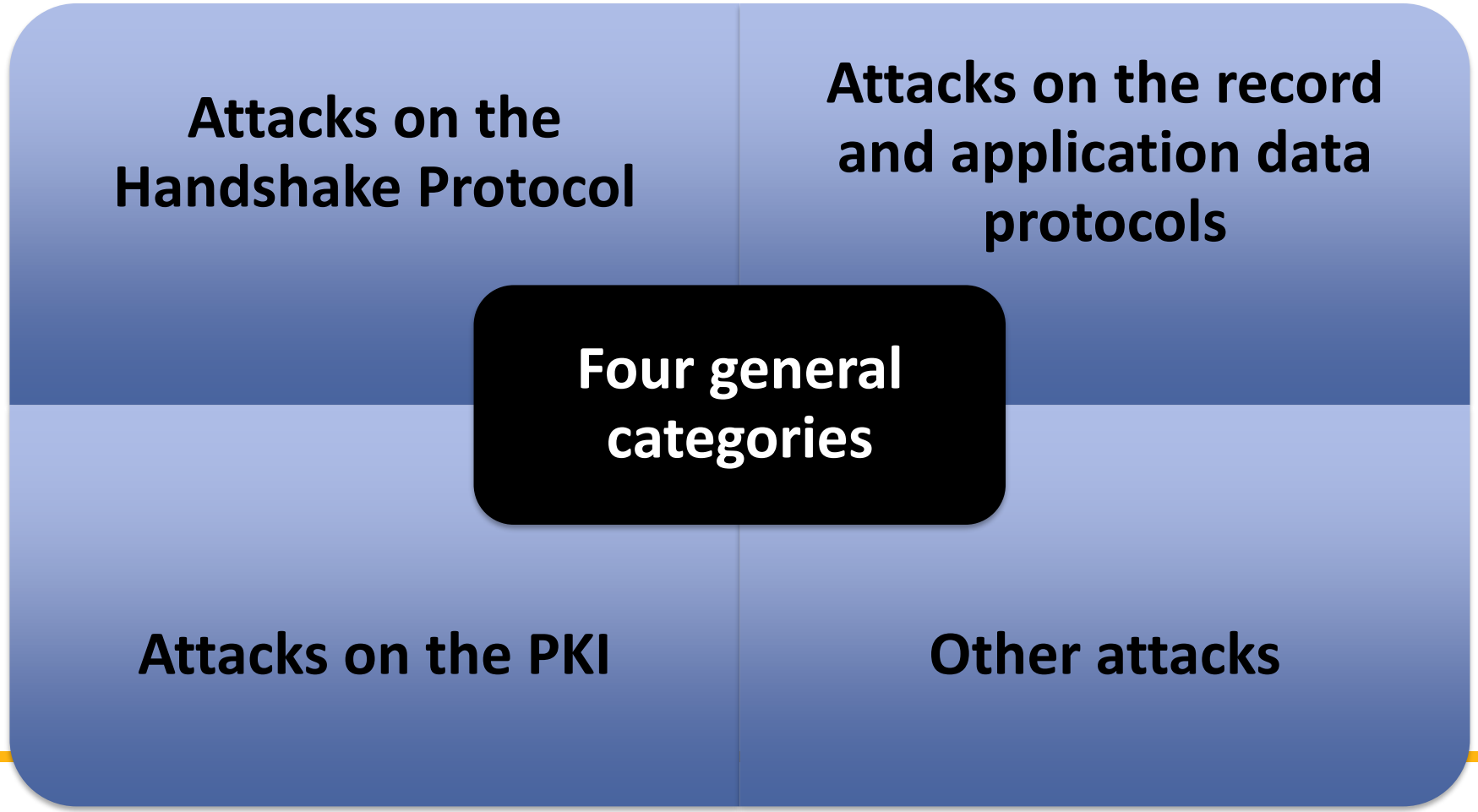


- A periodic signal generated by hardware or software
  - Indicates normal operation or synchronizes other parts of a system
  - Typically used to monitor the availability of a protocol entity
  - Timer: 1s to 60s (RFC6347)
  - Its Use is established during Phase 1 of the Handshake Protocol

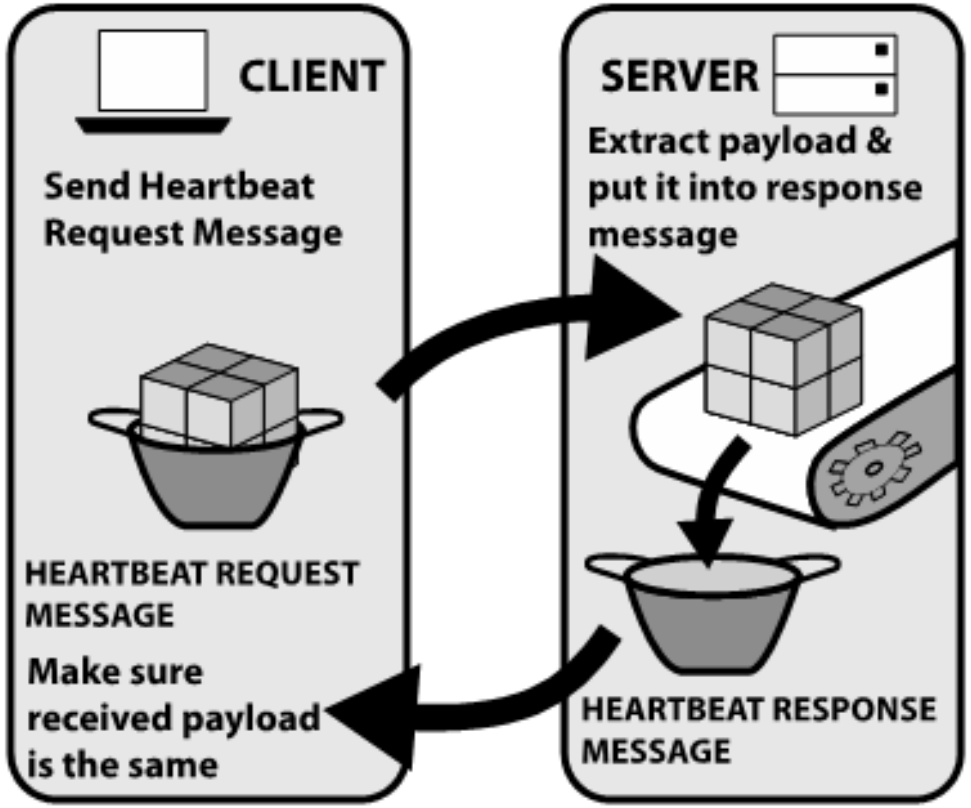
- Serves two purposes:
  - Assures the sender that the recipient is still alive
  - Generates activity across the connection during idle periods

```
struct {
   HeartbeatMessageType type;
   uint16 payload_length;
   opaque payload[HeartbeatMessage.payload_length];
   opaque padding[padding_length];
} HeartbeatMessage;
```

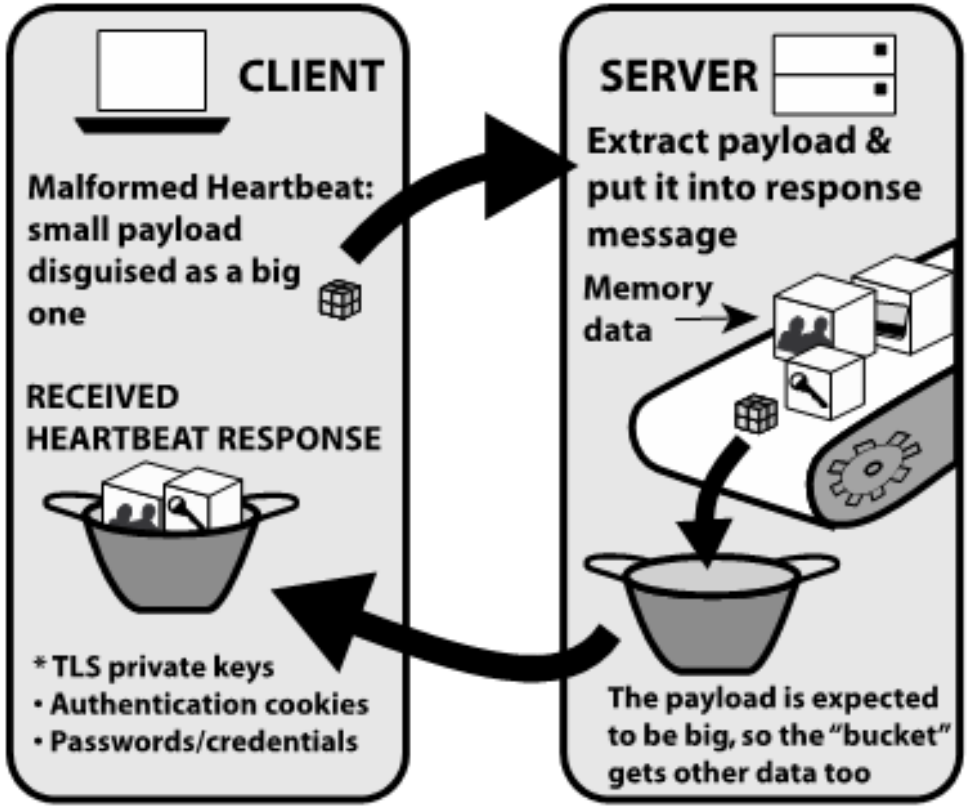
SSL/TLS Attacks



The Heartbleed Exploit (Source: BAE Systems)



(a) How TLS Heartbeat Protocol works



(b) How TLS Heartbleed exploit works

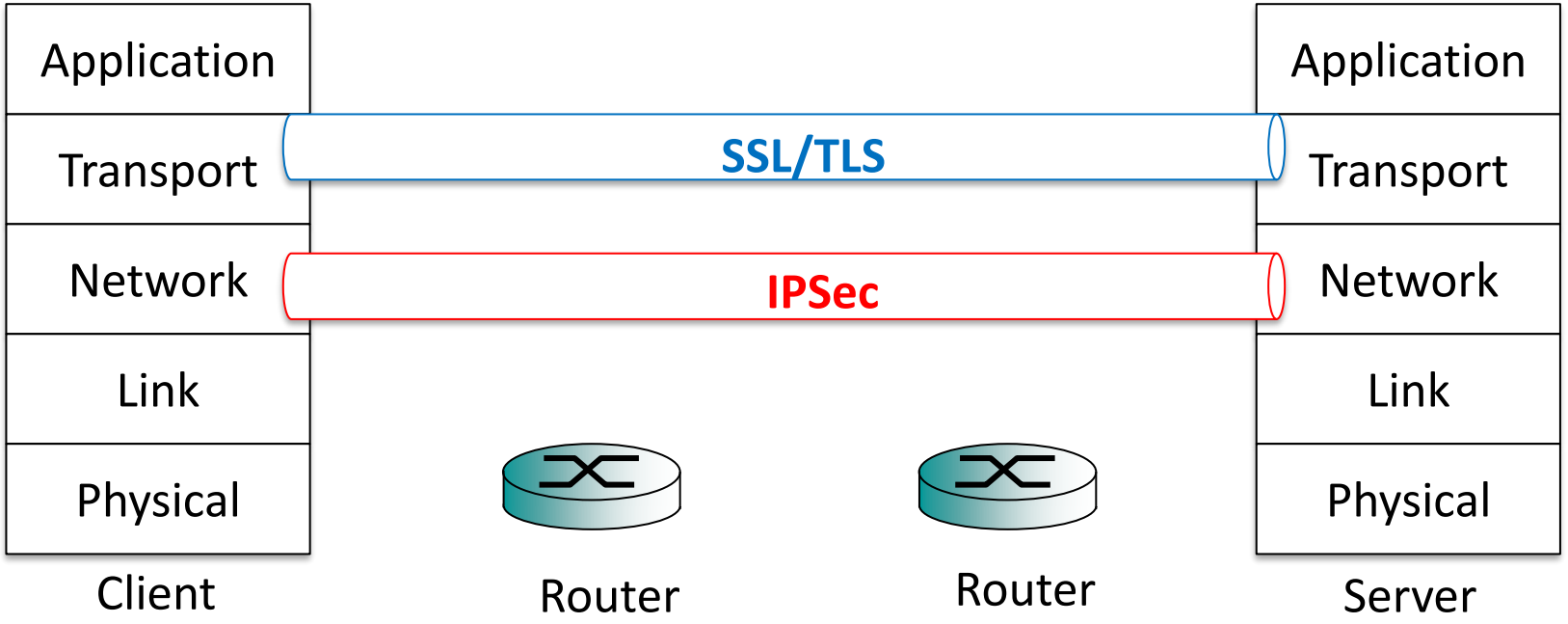
Outline

- Secure E-mail
- DomainKeys Identified Mail
- Secure Sockets Layer (SSL) and Transport Layer Security (TLS)
- HTTPS (HTTP over SSL)
- IPv4 and IPv6 Security

HTTPS (HTTP over SSL/TLS)

- Combination of HTTP and SSL
  - HTTP over TLS (RFC2818)
  - Secure communication between a Web browser and a Web server
- Built into all modern Web browsers
  - URL addresses begin with https://
- Connection initiation and closure
  - HTTP client act as TLS client
  - TLS handshake ➔ HTTP request
  - Three levels: HTTP, TLS session, TCP

IPSec v.s. SSL/TLS



Applications of IPSec

- Secure branch office connectivity over the Internet
- Secure remote access over the Internet
- Establishing extranet and intranet connectivity with partners
- Enhancing electronic commerce security

Benefits of IPSec

- Strong security applied to all traffic at a firewall or router
- Resistant to bypass at a firewall
- Transparent to apps
  - No need to change software
- Transparent to end users
  - No need to train users on security mechanisms

Benefits of IPSec (Cont.)

- Routing apps: prevent attackers from disrupting communications or diverting some traffic
  - A router advertisement from an authorized router
  - A neighbor advertisement from an authorized router
  - A redirect message from the router to which the initial packet was sent
  - A routing update is nor forged

The Scope of IPSec

- Two main functions
  - Encapsulating Security Payload (ESP): a combined authentication/encryption function
  - A key exchange function
- VPN: both authentication and encryption are generally desired
- Authentication Header (AH): authentication-only function (deprecated)

Security Associations

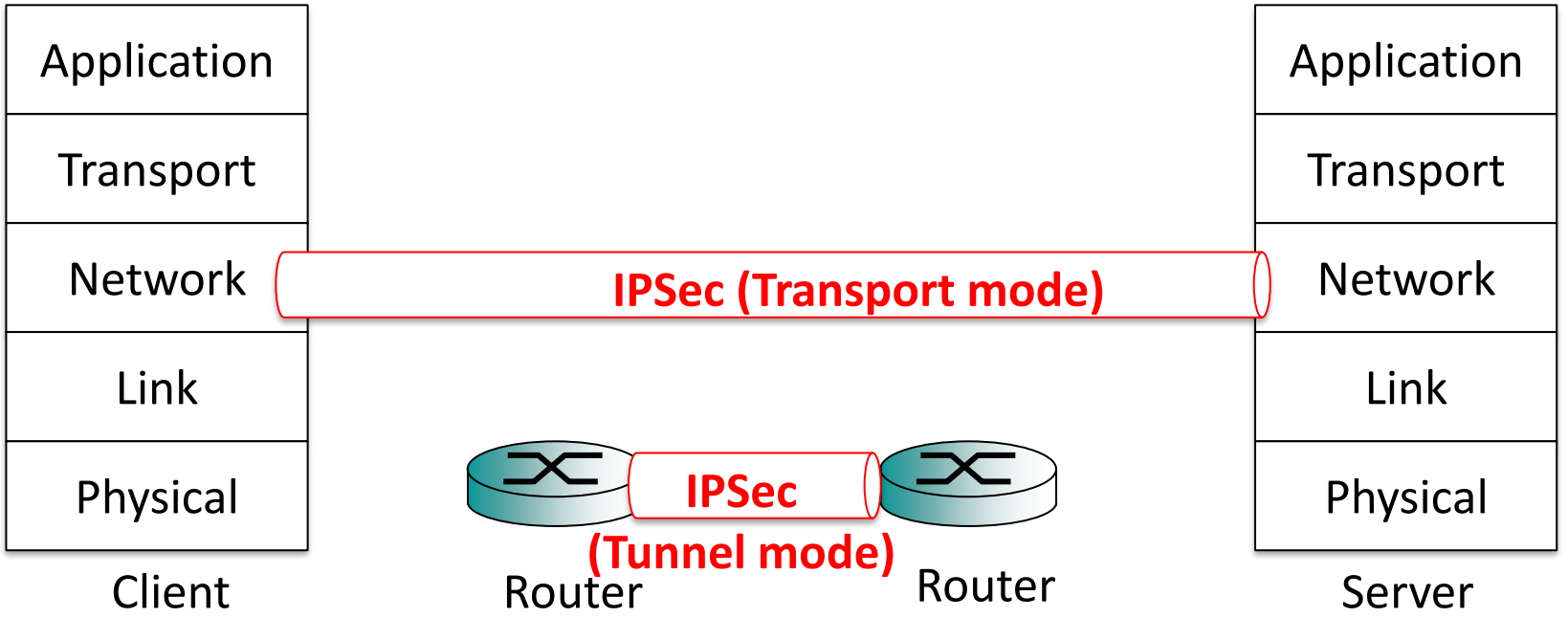
- A key concept of IPSec
  - One-way relationship between a sender and a receiver
  - Two-way secure exchange: two SAs are required
- Uniquely identified by three parameters
  - Security parameter index (SPI)
  - IP destination address
  - Protocol identifier: AH or ESP

Security Associations (Cont.)

- Characterized by the following parameters
  - Sequence number counter: 32-bit
  - Sequence counter overflow: A flag ➔ whether overflow ➔ an auditable event
  - Antireplay window: defining a sliding window
  - AH information
    - Algorithm, keys, key lifetimes, etc.
  - ESP information
    - Algorithm, keys, init values, key lifetimes, etc.
  - Lifetime of this security association
  - IPSec protocol mode: tunnel or transport
  - Path MTU

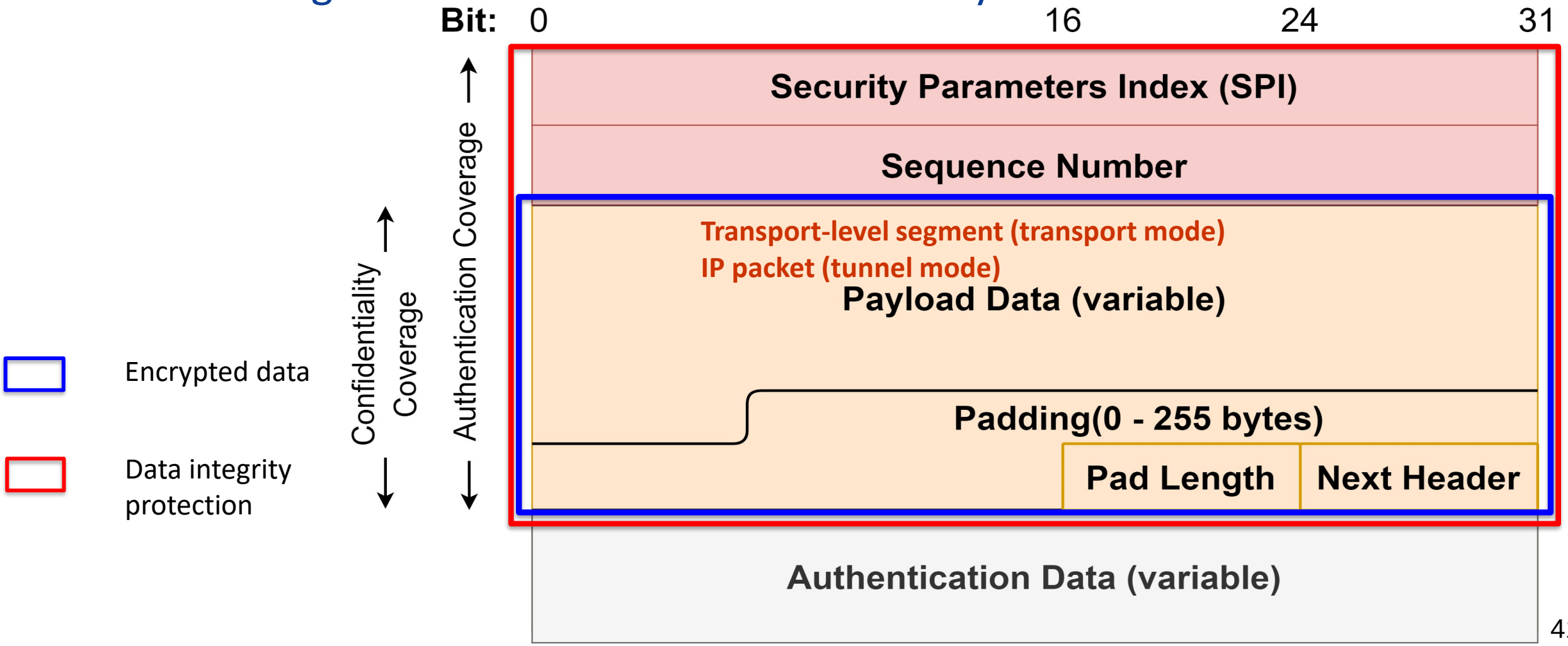
Two IPSec Operation Modes

- Transport and Tunnel modes



Encapsulating Security Payload

- Providing authentication and confidentiality services



Transport and Tunnel Modes

Transport Mode

- Protection: the payload of an IP packet
- Typically used for end-to-end communication between two hosts
- ESP protects the IP payload but not the IP header

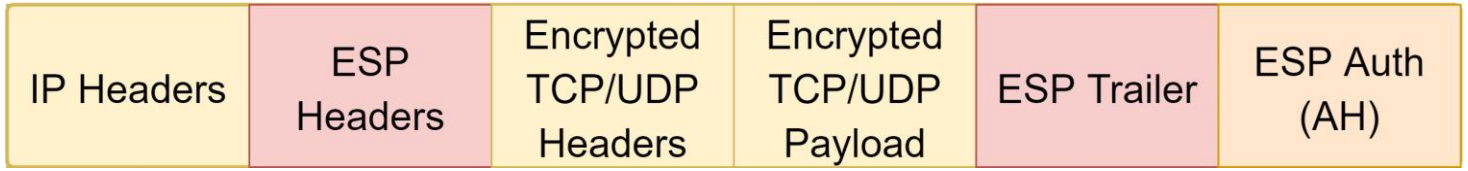
Tunnel Mode

- Protection: the entire IP packet
- Entire original packet travels through a tunnel from one point to another
- Used when one or both ends of a security association are a security gateway
- Hosts on networks behind firewalls may engage in secure communications without implementing IPSec

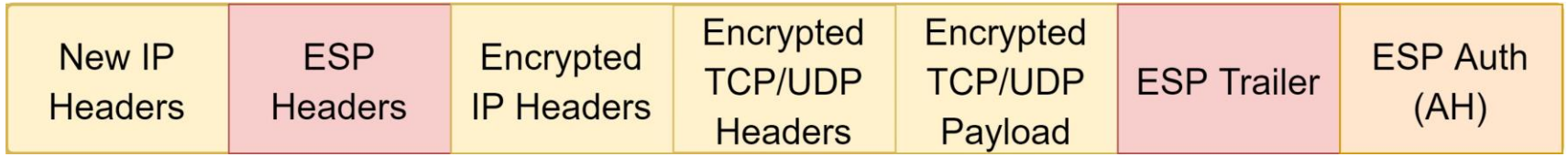
IPSec: AH + ESP

- IP AH only

- IP AH + ESP
  - Transport mode

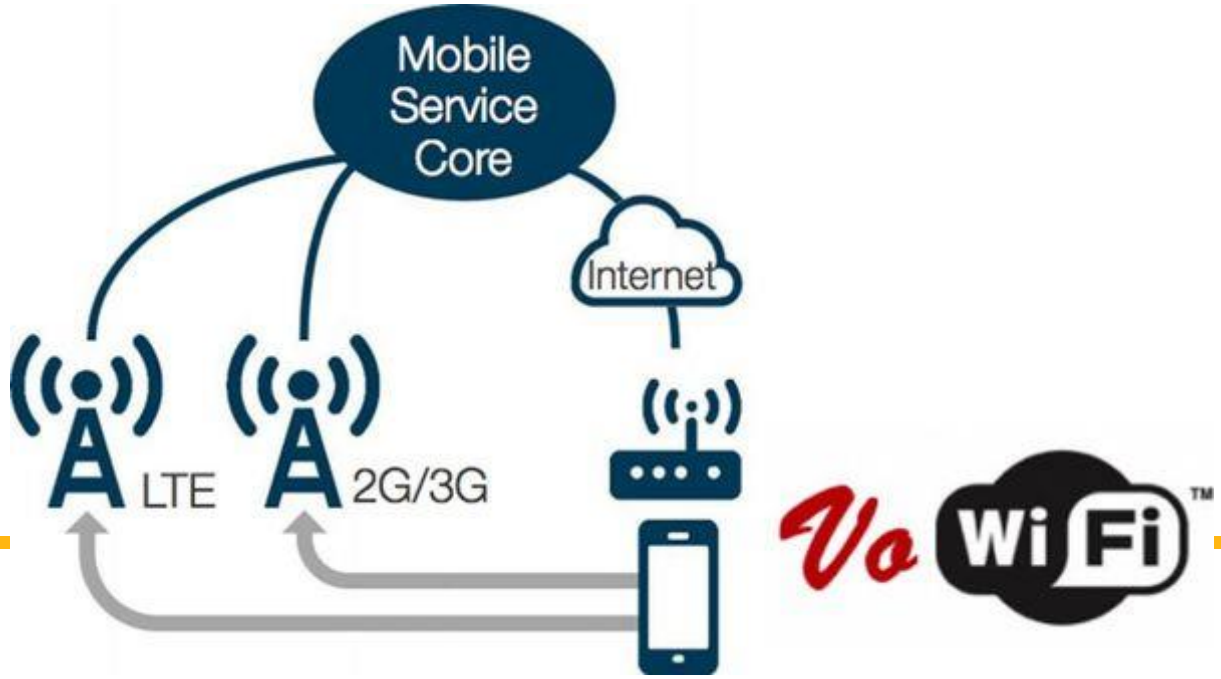


  - Tunnel mode

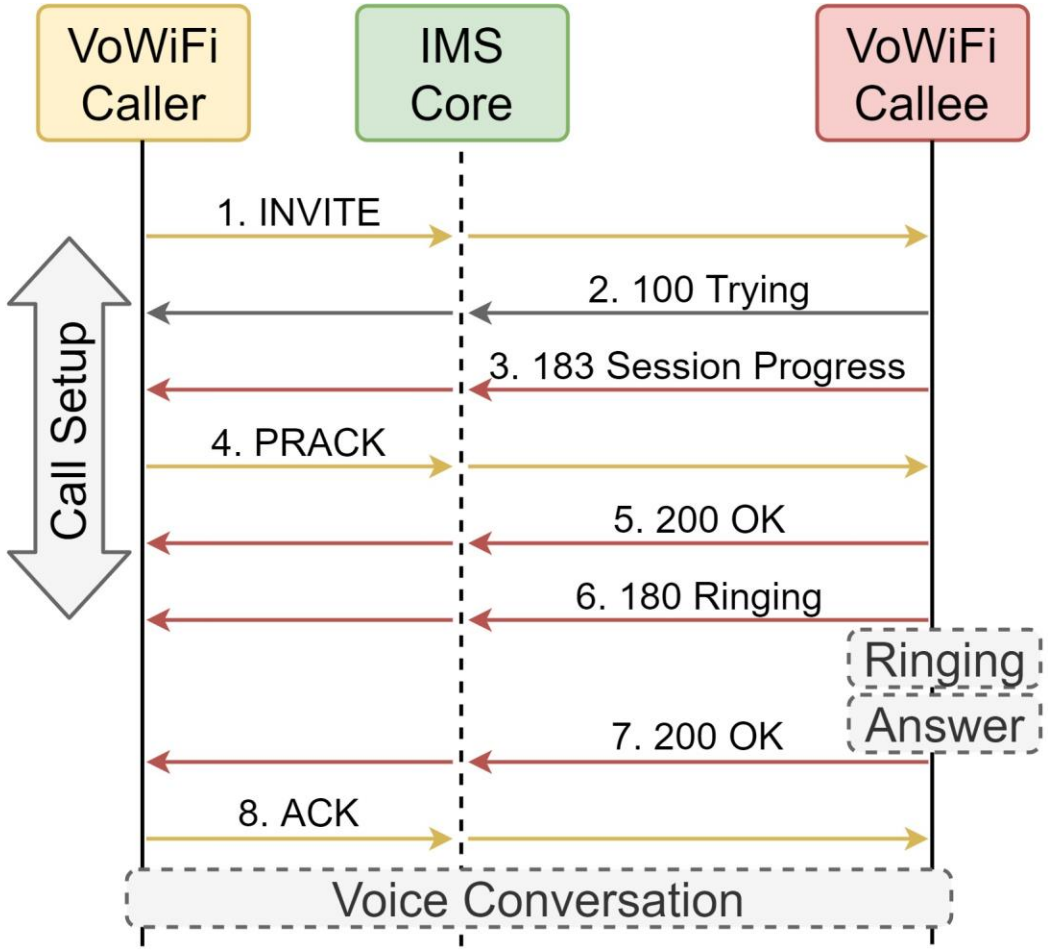


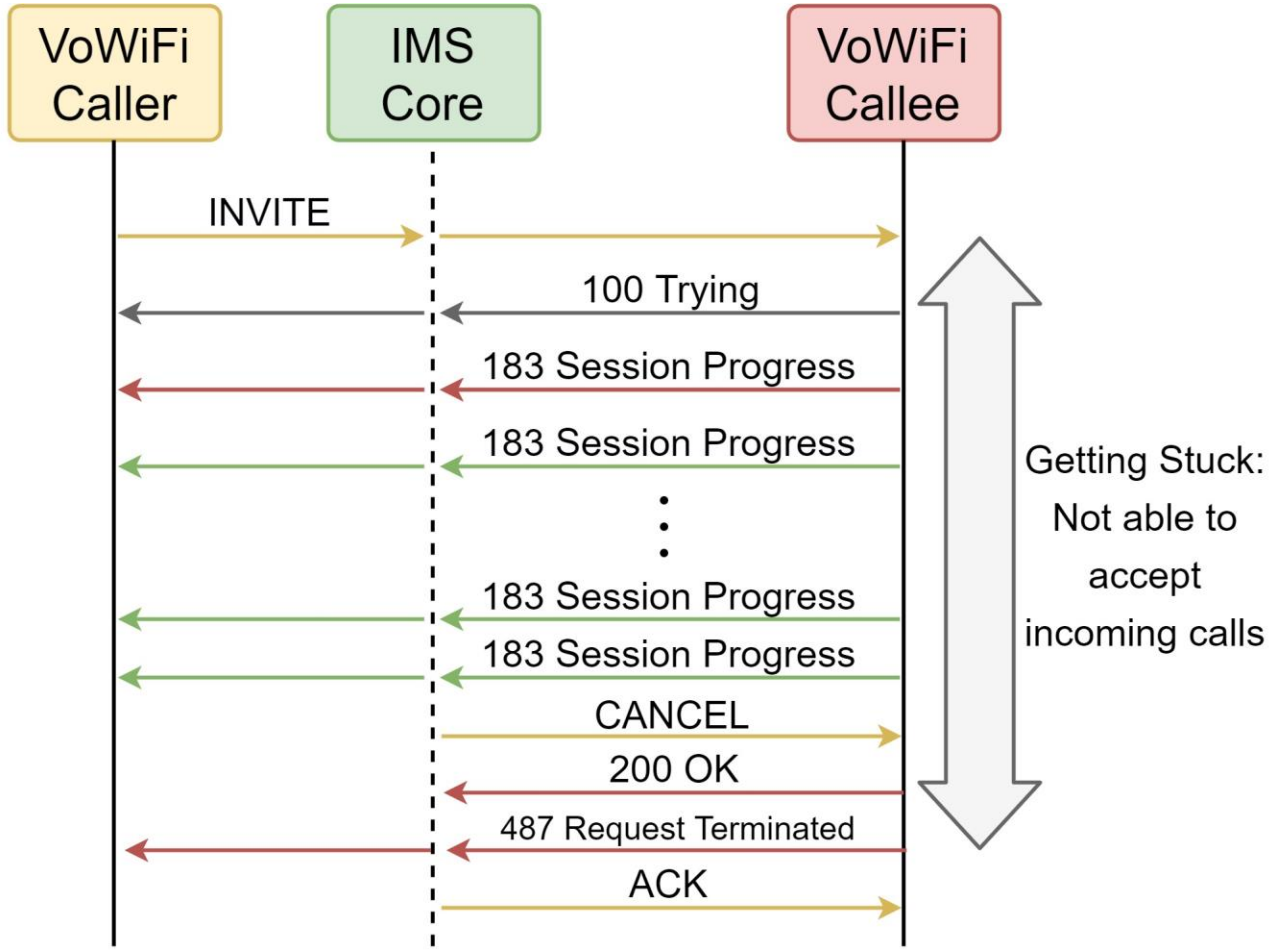
Case Study: VoWi-Fi Session Hijacking

- VoWi-Fi (Voice over Wi-Fi): telephony calls over Wi-Fi
  - Complements VoLTE (Vocie over LTE) for poor-signal areas
  - Supported by a system called IMS (IP Multimedia Subsystem)



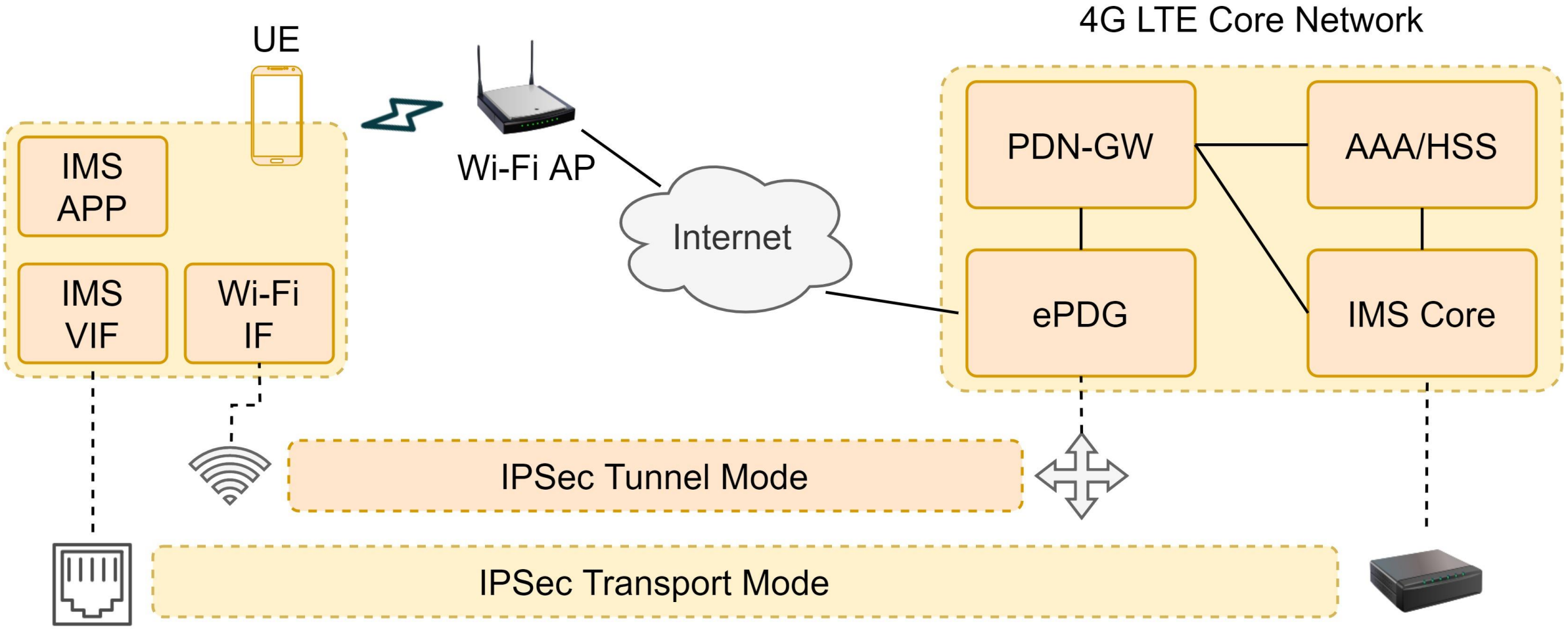
A Vulnerability at the Call State Machine



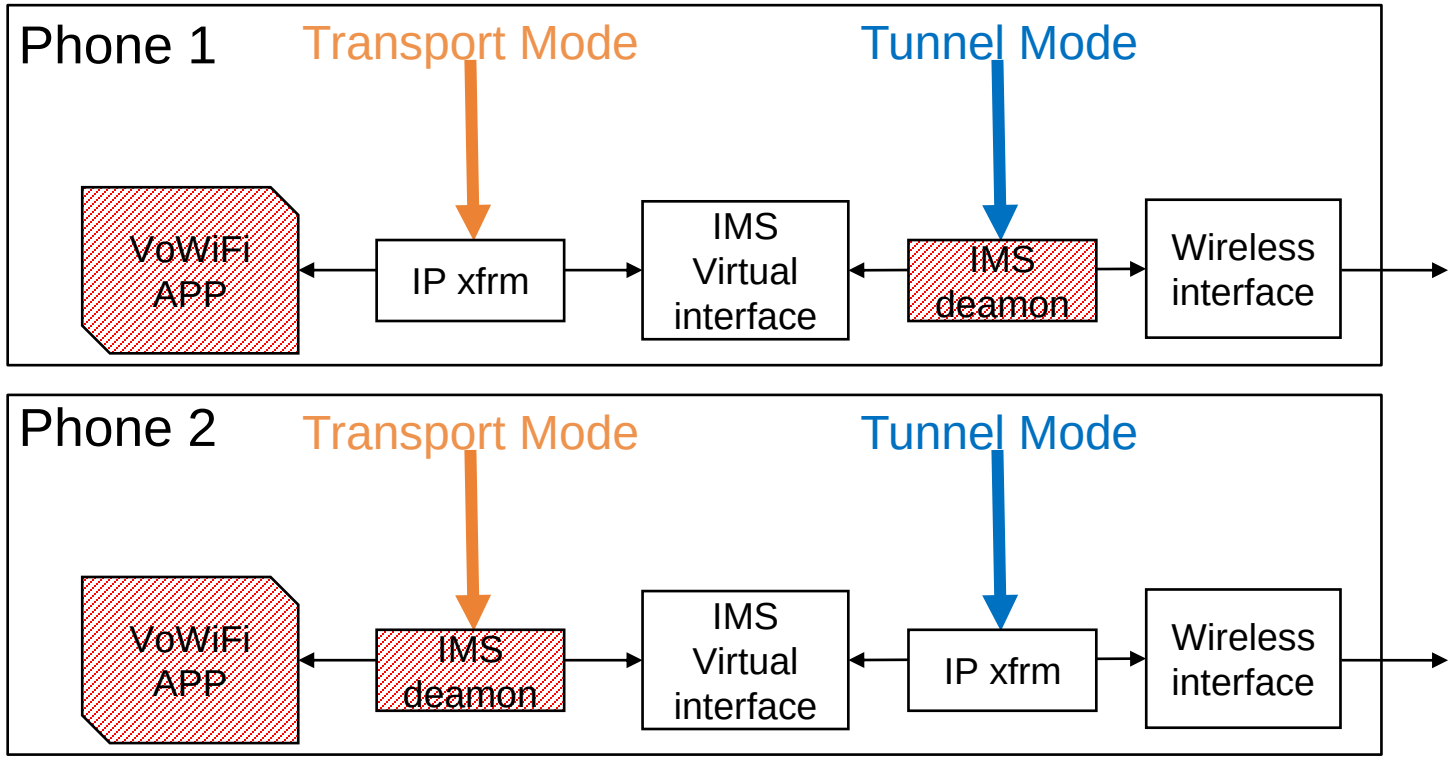


VoWi-Fi SIP Sessions Protected by IPSec

- How to hijack the session to exploit the vulnerability?



IPSec Implementation Analysis



- XFRM ("transform"): The Linux kernel's IP framework for transforming packets (such as encrypting payloads)
  - Support IPSec

Example: XFRM IPSec SAs

- Phone 1

```
dreamlte:/ # ip xfrm state
src 10.159.204.230 dst 100.64.45.207
        proto esp spi 0x00017502 reqid 512 mode transport
        replay-window 4
        auth-trunc hmac(sha1) 0x35a5dccbf4cfa5f2c54999b6216d4cec 96
        enc ecb(cipher_null)
        anti-replay context: seq 0x5, oseq 0x0, bitmap 0x0000001f
        sel src 0.0.0.0/0 dst 0.0.0.0/0
src 100.64.45.207 dst 10.159.204.230
        proto esp spi 0x77b513cc reqid 511 mode transport
        replay-window 4
        auth-trunc hmac(sha1) 0x35a5dccbf4cfa5f2c54999b6216d4cec 96
        enc ecb(cipher_null)
        anti-replay context: seq 0x0, oseq 0x0, bitmap 0x00000000
        sel src 0.0.0.0/0 dst 0.0.0.0/0
src 10.159.204.230 dst 100.64.45.207
        proto esp spi 0x00017501 reqid 510 mode transport
        replay-window 4
        auth-trunc hmac(sha1) 0x35a5dccbf4cfa5f2c54999b6216d4cec 96
        enc ecb(cipher_null)
        anti-replay context: seq 0x24, oseq 0x0, bitmap 0xffffffff
        sel src 0.0.0.0/0 dst 0.0.0.0/0
src 100.64.45.207 dst 10.159.204.230
        proto esp spi 0x80cfc9f0 reqid 509 mode transport
        replay-window 4
        auth-trunc hmac(sha1) 0x35a5dccbf4cfa5f2c54999b6216d4cec 96
        enc ecb(cipher_null)
        anti-replay context: seq 0x0, oseq 0x33, bitmap 0x00000000
        sel src 0.0.0.0/0 dst 0.0.0.0/0
```

Example: XFRM IPSec SAs

- Phone 2

```
ASUS_Z01KDA:/ # ip xfrm state
src 223.22.236.1 dst 192.168.0.121
        proto esp spi 0x1f35c02b reqid 0 mode tunnel
        replay-window 0 flag af-unspec
        auth-trunc hmac(sha1) 0xc9d7d3aa2f57e7b0cf2216d795478a04d7efa010 96
        enc cbc(des3_ede) 0x43098b1d8fae1344671ba8e61e955c0fc5ee8706c6ac8cbb
        encap type espinudp sport 4500 dport 42987 addr 0.0.0.0
        anti-replay context: seq 0x0, oseq 0x0, bitmap 0x00000000
src 192.168.0.121 dst 223.22.236.1
        proto esp spi 0xef011233 reqid 0 mode tunnel
        replay-window 0 flag af-unspec
        auth-trunc hmac(sha1) 0x02027e6811161a69e9b16088d89c5dbd5cd58583 96
        enc cbc(des3_ede) 0x961028b4d0a30ebeadaa031918dd5f623204a30d3f3fbd52
        encap type espinudp sport 42987 dport 4500 addr 0.0.0.0
        anti-replay context: seq 0x0, oseq 0x65, bitmap 0x00000000
```

IPSec SAs Negotiation

Source	Destination	Protocol	Length	Info
100.72.58.140	10.159.204.229	SIP	557	Request: REGISTER sip:ims.mnc092.
10.159.204.229	100.72.58.140	SIP	894	Status: 401 Unauthorized 11030230

```
Session Initiation Protocol (REGISTER)
  Request-Line: REGISTER sip:ims.mnc092.mcc466.3gppnetwork.org SIP/2.0
  Message Header
    Via: SIP/2.0/TCP 100.72.58.140:5060;branch=z9hG4bK-524287-1---0fe6d123f0b
    Require: sec-agree
    [truncated]Security-Client: ipsec-3gpp;prot=esp;mod=trans;spi-c=58645;sp
```

Source	Destination	Protocol	Length	Info
100.72.58.140	10.159.204.229	SIP	557	Request: REGISTER sip:ims.mnc092.m
10.159.204.229	100.72.58.140	SIP	894	Status: 401 Unauthorized 110302303

```
Session Initiation Protocol (401)
  Status-Line: SIP/2.0 401 Unauthorized 11030230325
  Message Header
    Security-Server: ipsec-3gpp;alg=hmac-sha-1-96;prot=esp;mod=trans;ealg=null
```

IPSec Transport Packet

Source	Destination	Protocol	Length	Info
100.64.54.225	10.156.204.230	SIP/SDP	1224	Request: INVITE sip:+[redacted]023292
10.156.204.230	100.64.54.225	SIP	464	Status: 100 Trying |
10.156.204.230	100.64.54.225	SIP/SDP	212	Status: 183 Session Progress |
100.64.54.225	10.156.204.230	SIP	1096	Request: PRACK sip:sgc_c@10.156.2[redacted]
10.156.204.230	100.64.54.225	SIP	720	Status: 200 OK |
10.156.204.230	100.64.54.225	SIP	840	Status: 180 Ringing |

```
▾ Encapsulating Security Payload
    ESP SPI: 0xc0448453 (3225715795)
    ESP Sequence: 9
    ESP Pad Length: 1
    Next header: TCP (0x06)
    Authentication Data: 273cc5e6fa6ef23184580d6a
▾ Session Initiation Protocol (INVITE)
  › Request-Line: INVITE sip:+[redacted]023292;phone-context=ims.mnc[redacted].mcc[redacted].3g
  › Message Header
  ▾ Message Body
    ▾ Session Description Protocol
        Session Description Protocol Version (v): 0
      › Owner/Creator, Session Id (o): 24465697594557 1535544072365129 0 IN
        Session Name (s): FORGED SIP
```

How to Fabricate a IPSec-protected SIP Message?

- SIP message content
  - caller and callee numbers
  - Static info: SIP server/proxy addresses and supported features, etc.
  - Dynamic info: tag, call id, branch id, etc.
- ESP integrity protection
  - Obtain the IPSec security parameters from the SAD (Security Association DB)
    - SPI and hash algorithm/keys
  - Keep track of the sequence number of IPSec session
  - Apply default ESP padding algorithm
  - Generate authentication data using the hash algorithm/keys

Questions?