ALBERT-LUDWIGS-
UNIVERSITÄT FREIBURG

Communication Systems

SSL

University of Freiburg
Computer Science
Computer Networks and Telematics
Prof. Christian Schindelhauer

CoNe
Freiburg

IIF
INSTITUT FÜR
INFORMATIK
FREIBURG

Organization

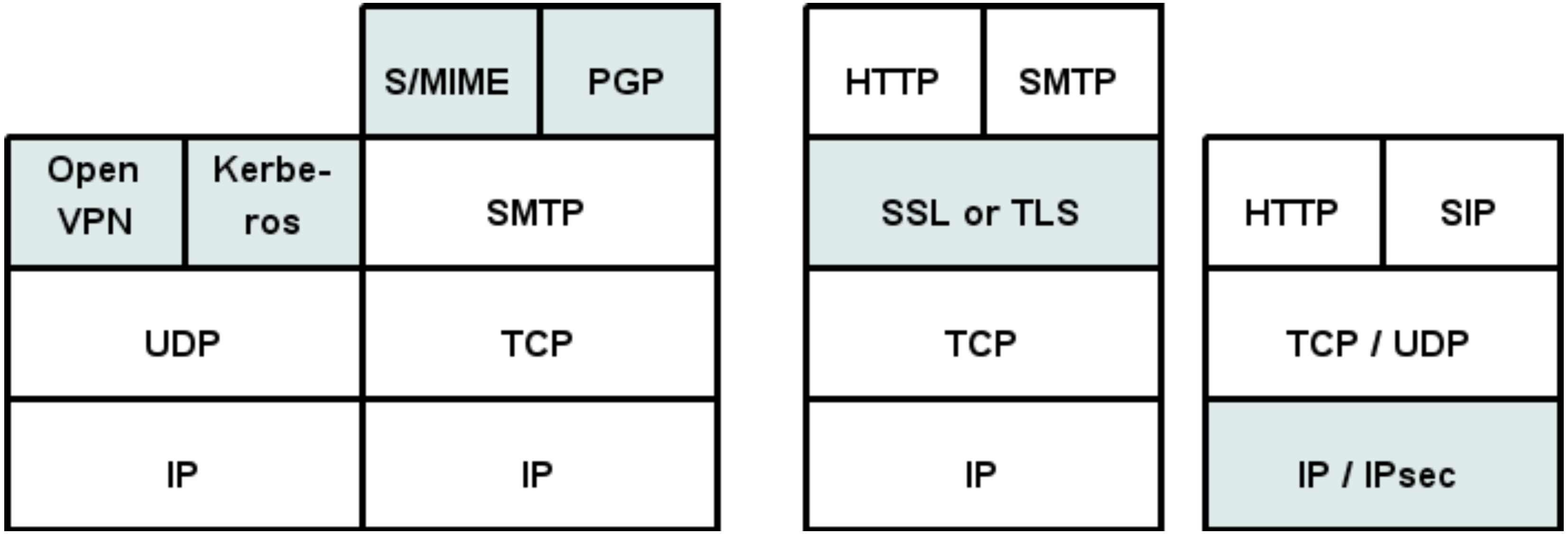
- I. Data and voice communication in IP networks
- **II. Security issues in networking**
- III. Digital telephony networks and voice over IP

Network Security Goals

- **Confidentiality**: only sender, intended receiver should "understand" message contents
    - sender encrypts message
    - receiver decrypts message
    - Privacy: hide `who is doing what with whom`
- **Authentication**: sender, receiver want to confirm identity of each other
- **Integrity**: sender, receiver want to ensure messages are not altered (in transit, or afterwards) without detection
- **Access and Availability**: services must be accessible and available to users

Network Security on Different Layers

- Security measures could be hooked to different layers of the stack
  - Link layer: one `hop` (e.g. wireless link)
  - IP Layer (IP-Sec): transparent to application (next Friday)
  - Transport Layer (SSL/TLS): easy, widely used
  - Application Layer (PGP, S/MIME)



Open VPN | Kerbe-ros | S/MIME | PGP
UDP | SMTP
IP | TCP
IP

HTTP | SMTP
SSL or TLS
TCP
IP

HTTP | SIP
TCP / UDP
IP / IPsec

SSL (Secure Socket Layer)

- Transport layer security service, yields secure channel
  - Secure byte stream
  - Optional public-key server authentication
  - Optional client authentication
- Development started by Netscape to offer secure Internet business
  - Used/Implemented with HTTP first (HTTPS, port 443)
  - Hash: combined MD5 & SHA
  - Encryption: Diffie Helman, RSA & DES, RC4
- Version 3 designed with public input; subsequently became Internet standard TLS (Transport Layer Security)

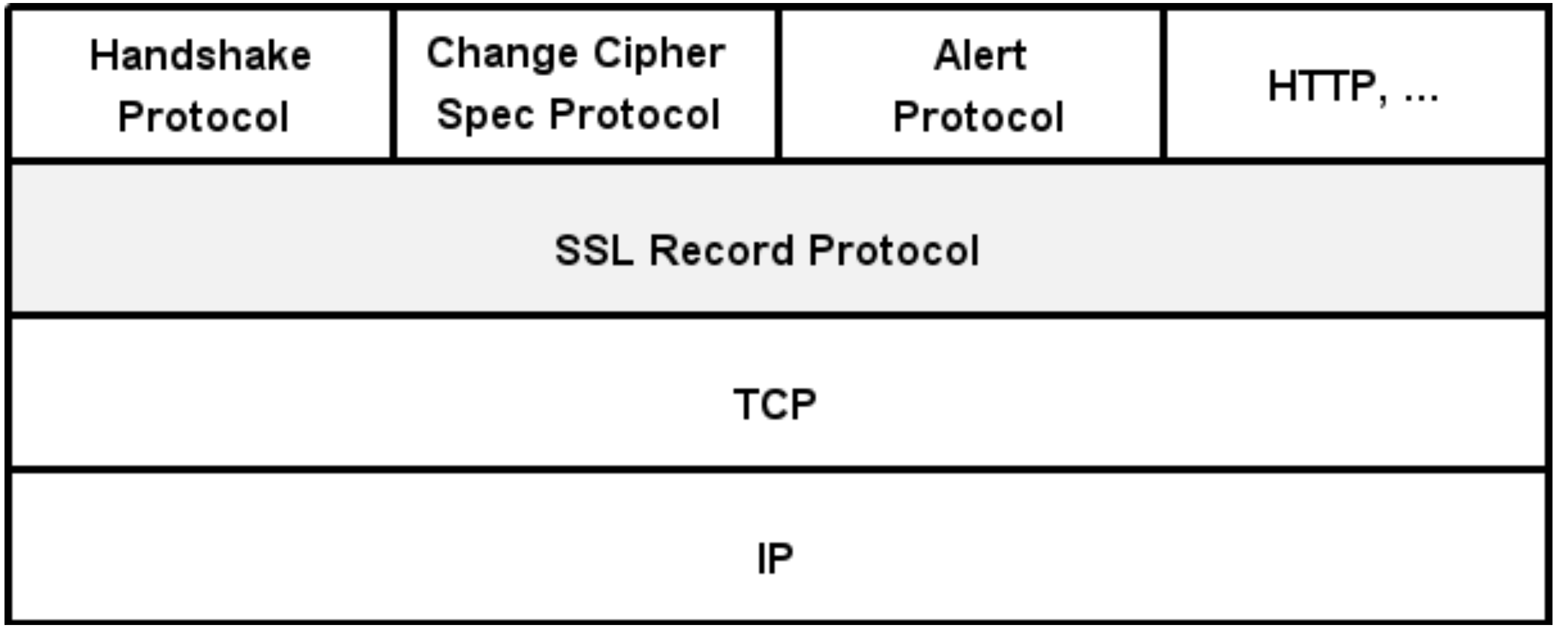
SSL (Secure Socket Layer)

- Uses TCP to provide a reliable end-to-end service
  - Not restricted for secure web (HTTP) transactions
  - Useful for any TCP based service to be secured: HTTP, IMAP, POP, NNTP, telnet, telephony signaling

- SSL implements two layers of protocols
- SSL session
  - Association between client & server
  - Created by the Handshake Protocol
  - Define a set of cryptographic parameters
  - May be shared by multiple SSL connections

SSL (Secure Socket Layer)

- **SSL connection**
  - A transient, peer-to-peer, communications link
  - Associated with one SSL session



Handshake Protocol

Change Cipher Spec Protocol

Alert Protocol

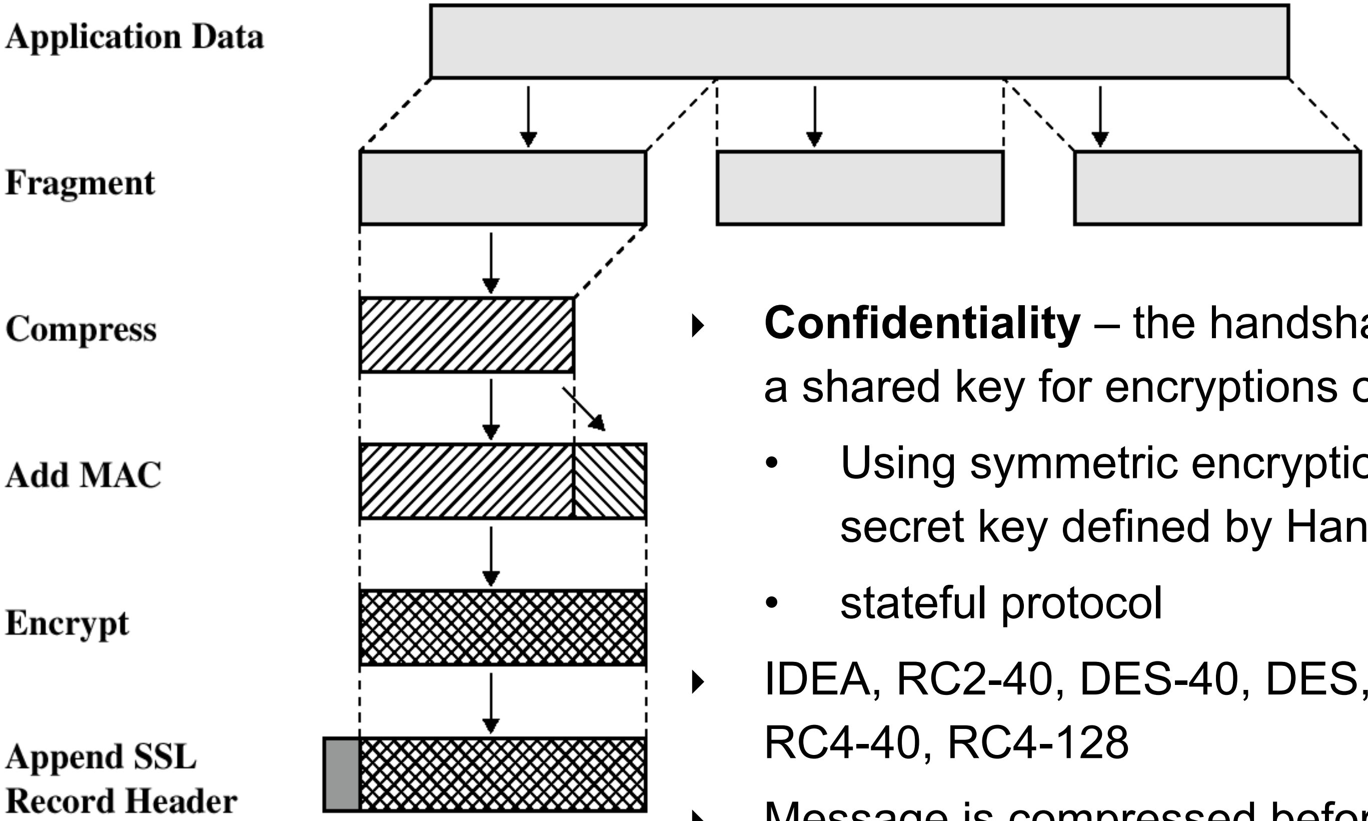
HTTP, ...

SSL Record Protocol

TCP

IP

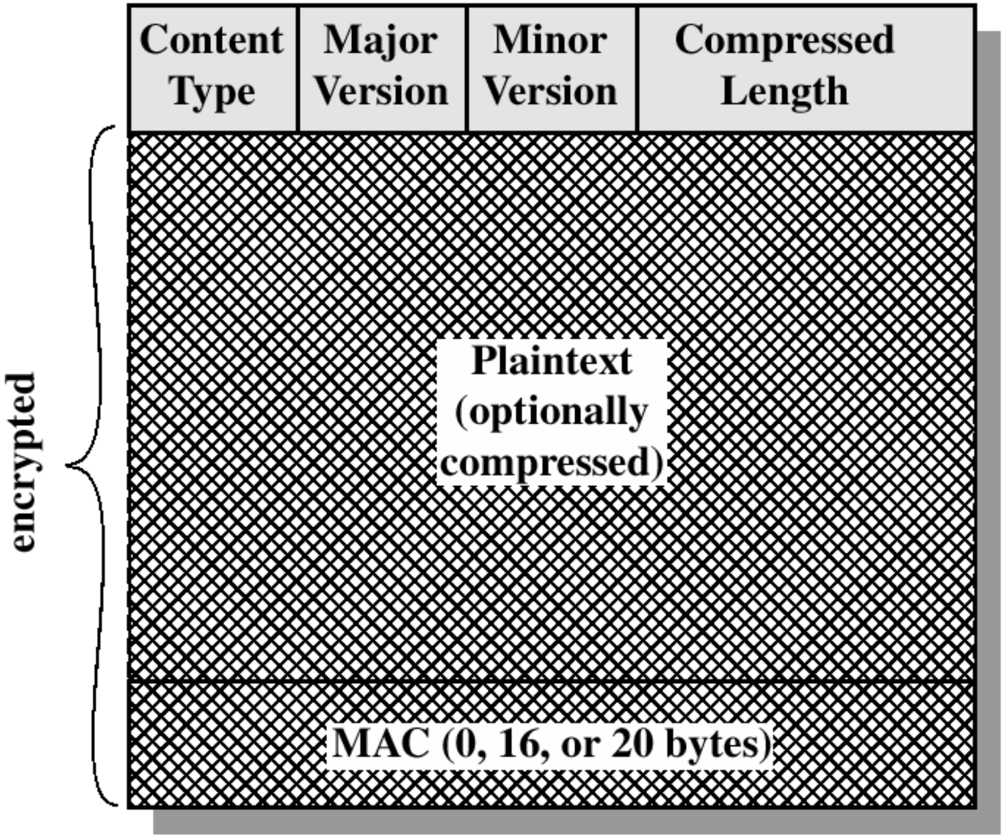
SSL record protocol



- **Confidentiality** – the handshake protocol defines a shared key for encryptions of SSL payloads
  - Using symmetric encryption with a shared secret key defined by Handshake Protocol
  - stateful protocol
- IDEA, RC2-40, DES-40, DES, 3DES, Fortezza, RC4-40, RC4-128
- Message is compressed before encryption

SSL record protocol and format

- The record format leads to
- **Message Integrity** – the handshake protocol defines a shared key used to form message authentication code (MAC)
  - Similar to HMAC but with different padding



SSL MAC calculation

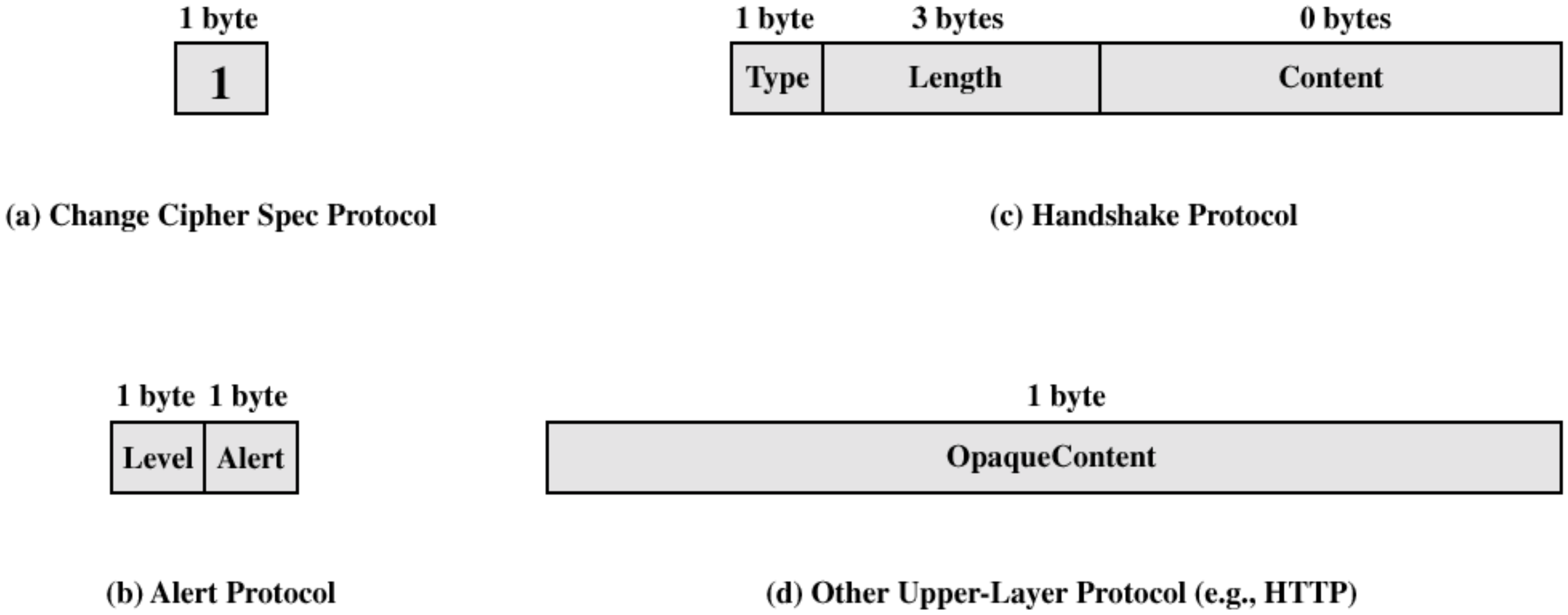
- Hash(MAC_secret_key || pad2 || hash(MAC_secret_key || pad1 || seqNum || SSLcompressed.type || SSLcompressed.length || SSLcompressed.fragment))
- Where:
  - Mac_secret_key –
  - pad1 = 0x36 repeated 48 times for MD5 40 times for SHA-1
  - pad2 = 0x5C repeated …
  - SSLcompressed.type = the higher level protocol used to process this fragment

SSL encryption

- Fragment size 2^{14} = 16384 bytes
  - Compression must be lossless and must not increase length more than 1024
  - No compression algorithm specified in SSLv3 – default no compression
  - Block Cipher Encryption Methods
    - IDEA (128) RC2-40, DES-40, DES (56), 3DES (168)
  - Stream Cipher Encryption choices
    - RC4-40, RC4-128

SSL payload / Change Cipher Specification Protocol

- Change Cipher Spec Protocol
  - consists of a single message of a single byte with value 1
  - it means copy pending state to current state



(a) Change Cipher Spec Protocol

(c) Handshake Protocol

(b) Alert Protocol

(d) Other Upper-Layer Protocol (e.g., HTTP)

SSL Alert Protocol

- Conveys SSL-related alerts to peer entity
- Severity
  - Warning or fatal: 1=warning, 2=fatal
- Specific alert
  - Unexpected message, bad record mac, decompression failure, handshake failure, illegal parameter
  - Close notify, no certificate, bad certificate, unsupported certificate, certificate revoked, certificate expired, certificate unknown
- Compressed and encrypted like all SSL data

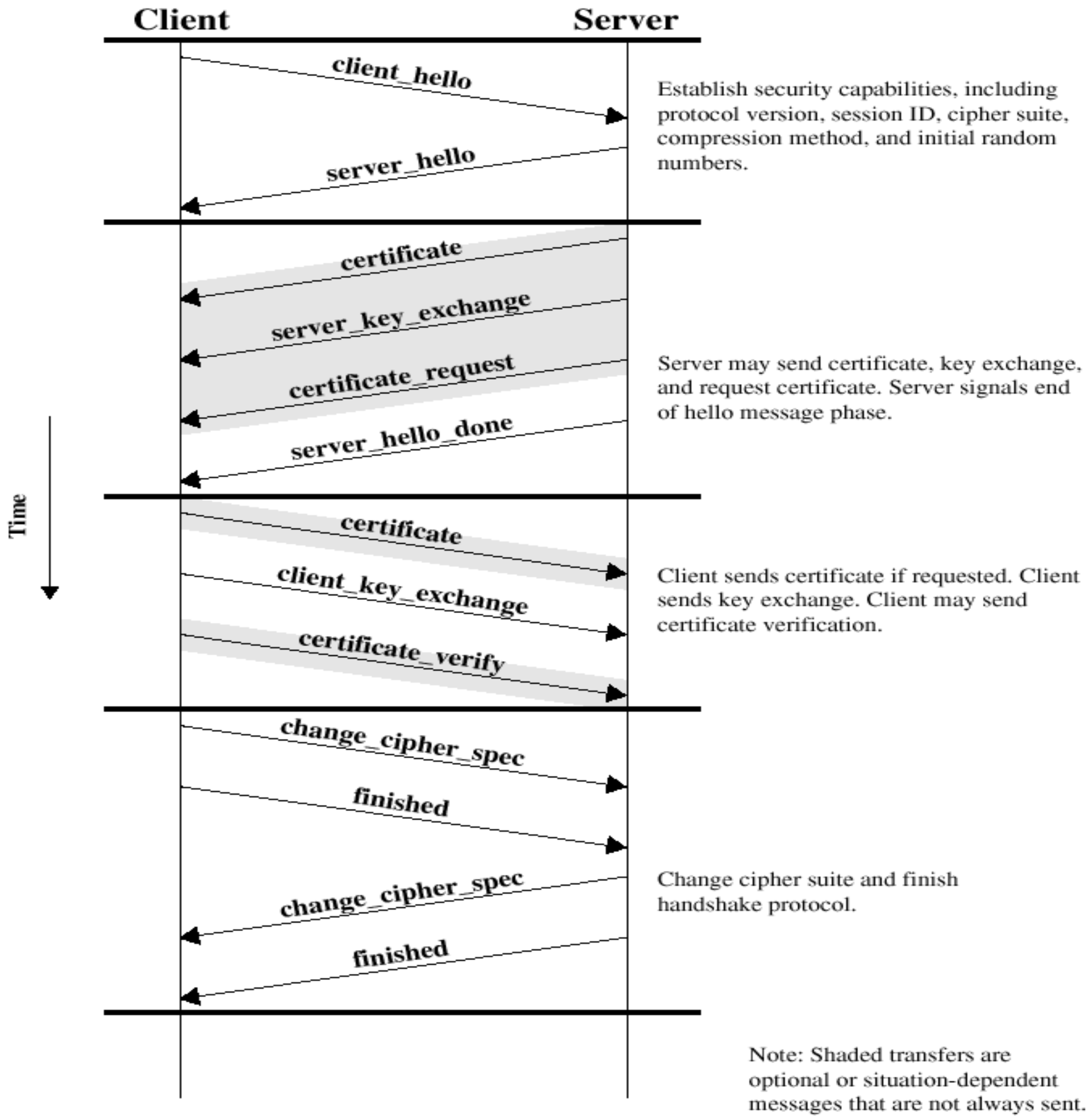
SSL Handshake Protocol

- Most complex part of SSL
  - Allows the server and client to authenticate each other
  - Negotiate encryption, MAC algorithm and cryptographic keys
  - Used before any application data are transmitted
- Message Fields
  - Type (8)
  - Length (24)
  - Content (≥ 1 byte) parameters
- Several Message types

SSL Handshake Protocol – message types

- Message types (name (value)):
  - Hello-request (null)
  - Client-hello (version,random(32B), sessionID, cipher suite, compression method)
  - Server_hello (same as Client-hello)
  - 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)

SSL Handshake Protocol



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

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

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

Change cipher suite and finish handshake protocol.

Note: Shaded transfers are optional or situation-dependent messages that are not always sent.

- Colored messages are optional
- Phase 1-3 messages are plaintext

SSL Handshake Protocol – Phase 1

- Establish security capabilities
  - Client_hello
    - Version = highest SSL understood by client
    - Random 32 bit time stamp + 28 random bytes (secure random number generator)
    - sessionID: 0 to establish new connection, non-zero means update parameters of an existing session
    - Ciphersuite: sequence of cryptographic algorithms in decreasing order of preference (key exchange + CipherSpec)
    - Compression methods: sequence of compression methods

SSL Handshake Protocol – Phase 1

- Establish security capabilities
  - Server_hello is sent back
    - same as from client but confirmation to suggested values:
    - Highest common version, new random field, same sessionID if nonzero, new sessionID otherwise, the selected ciphersuite and the selected compression technique
- Key Exchange methods
  - RSA – secret key is encrypted with receiver's RSA public key
  - Fixed Diffie-Hellman
  - Ephemeral Diffie Hellman
  - Anonymous Diffie-Hellman
  - Fortezza

SSL Handshake Protocol – Phase 1

- CipherSpec follows containing the fields
  - Cipher algorithm
  - MAC algorithm
  - CipherType: block or stream
  - Hash size: 0, 16 for MD5 or 20 for SHA-1 bytes
  - Key material – sequence of bytes used to generate keys
  - IV size of Initial Value for Cipher Block Chaining (CBC)

SSL Handshake Protocol – Phase 2

- Server Authentication and Key Exchange
- Server sends
  - Certificate: X.509 certificate chain (not required for anonymous Diffie-Hellman)
  - Server_key_exchange (not always need e.g. fixed Diffie-Hellman) - Hash(Client_hello.random|| ServerHello.random||ServerParms)
  - Certificate_request: certificate type and certificate authorities
  - Server_hello_done: I’m done and I’ll wait on response

SSL Handshake Protocol – Phase 3

- Client Authentication and Key Exchange
- Client verifies server certificate and checks the server hello paramters
  - If not in list of CAs, may trust the new certificate
  - Client generates 48 byte pre-secret
- Client sends
  - pre-secret encrypted w/ server's public key in certificate
  - Certificate: if requested, client_key_exchange message must be sent
  - Certificate_verify message to provide explicit verification of client certificate
  - Session key now generated from master secret and client hello random provides "salt"

SSL Handshake Protocol – Phase 4, 5

- Finishing up: switch to next cipher_spec
- Client sends
  - Change_cipher_spec message
  - Finished message under new algorithms, keys (new cipher_spec)
- Server answers
  - Change_cipher_spec message
  - Finished message under new algorithms, keys (new cipher_spec)
- Phase 5: Now encrypted application data could be exchanged between both parties

SSL Version 3 and Transport Layer Security

- IETF standard RFC 2246 similar to SSLv3
- With minor differences
  - in record format version number
  - uses HMAC for MAC
  - a pseudo-random function expands secrets
  - has additional alert codes
  - some changes in supported ciphers
  - changes in certificate negotiations
  - changes in use of padding

ALBERT-LUDWIGS-
UNIVERSITÄT FREIBURG

Communication Systems

University of Freiburg
Computer Science
Computer Networks and Telematics
Prof. Christian Schindelhauer

CoNe
Freiburg

IIF
INSTITUT FÜR
INFORMATIK
FREIBURG