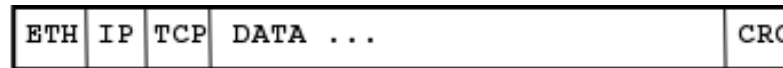
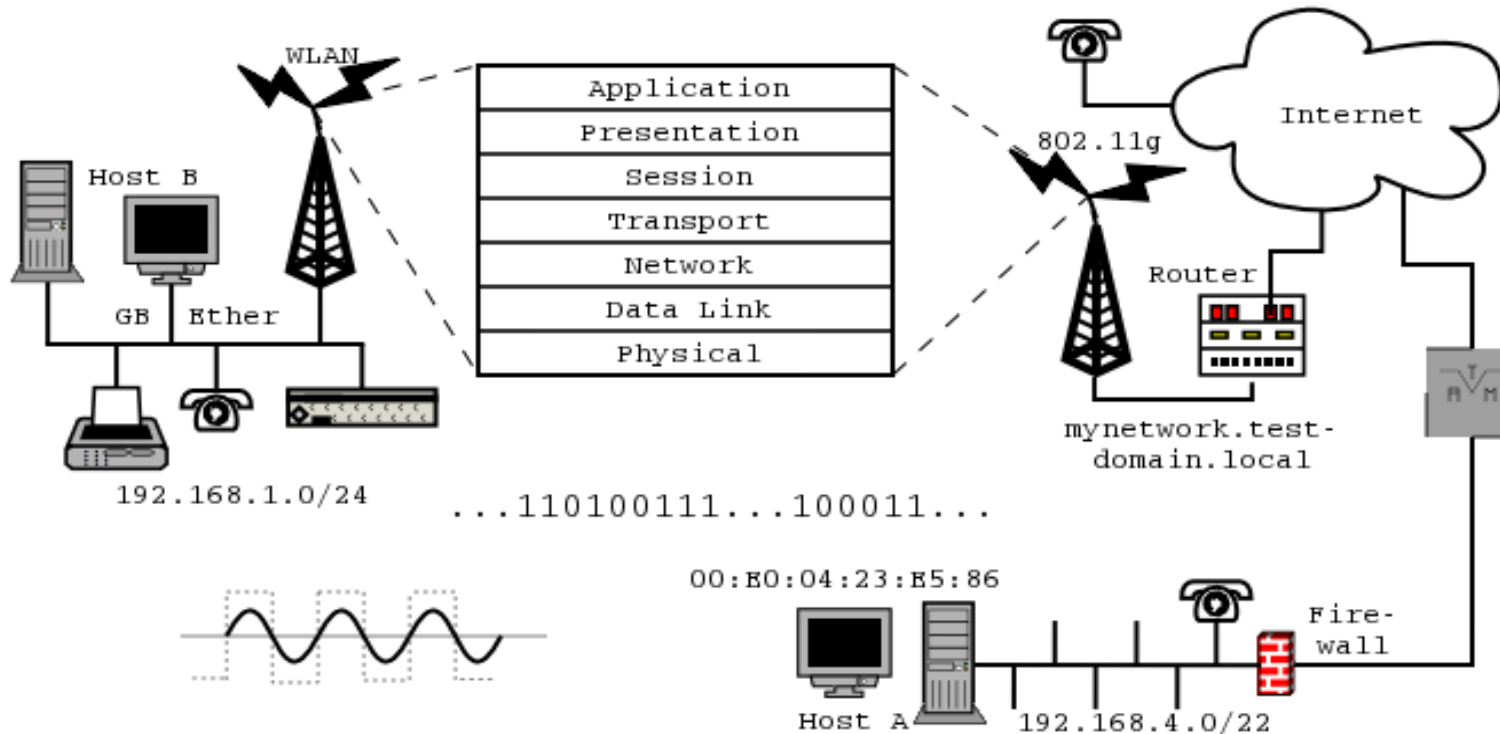


Communication Systems

16th lecture



... <http://www.ks.uni-freiburg.de/inetnetwork>



Chair of Communication Systems
Department of Applied Sciences
University of Freiburg
2009

Communication Systems

Organization

- Welcome to the New Year!
- Reminder: Structure of “Communication Systems” lectures
 - I. Data and voice communication in IP networks
 - **II. Security issues in networking**
 - III. Digital telephony networks and voice over IP
- We are in second part, some motivation, security implications in the Christmas lecture last year
- We repeat on tunneling in the practical part
- Starting into theory of securing network connections on different layers of the protocol stack

Communication Systems

Q&A

- What does tunneling mean for the number of interfaces on a networked host?
 - Which problems might be introduced with protocol stacking?
 - How many layers of protocols are possible?
 - At which layer of the TCP/IP stack a tunnel could be implemented?
- Why would the IPv6 conversion most probably require protocol tunneling?
- It is possible to transport any protocol over any other?
 - USB over IP?
 - ISDN over IP?
 - Which limitations you can think of?

Communication Systems

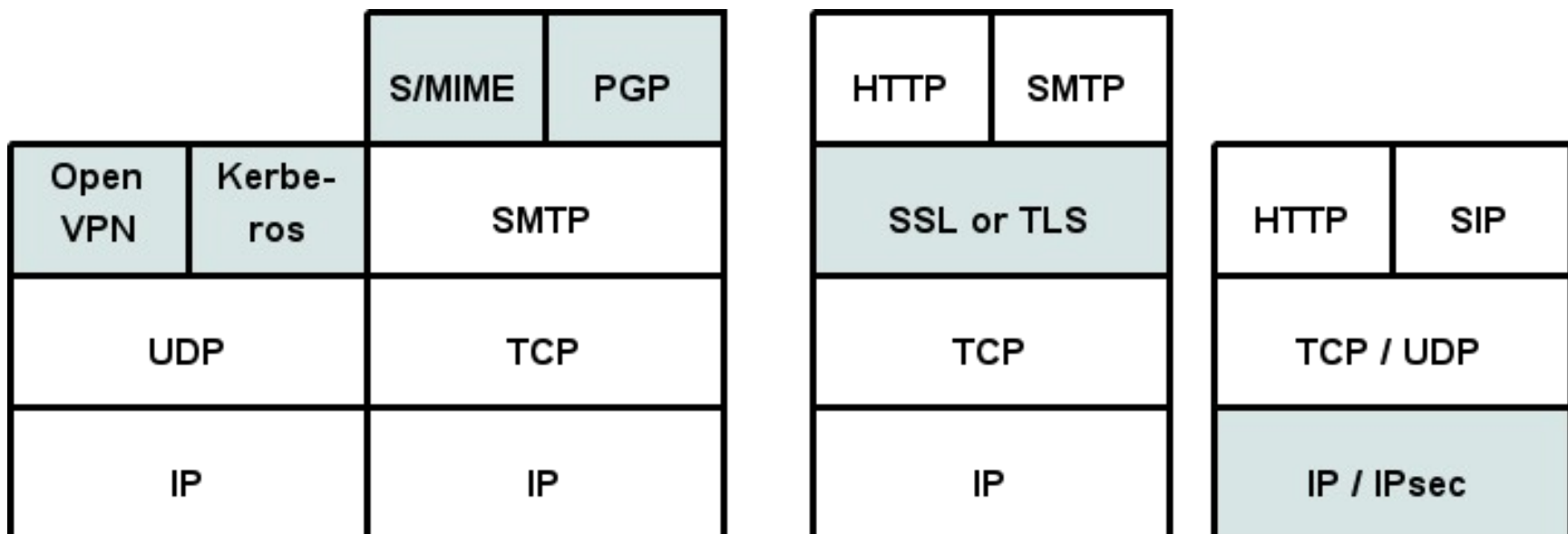
network security goals (recap)

- **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

Communication Systems

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)



Communication Systems

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)

Communication Systems

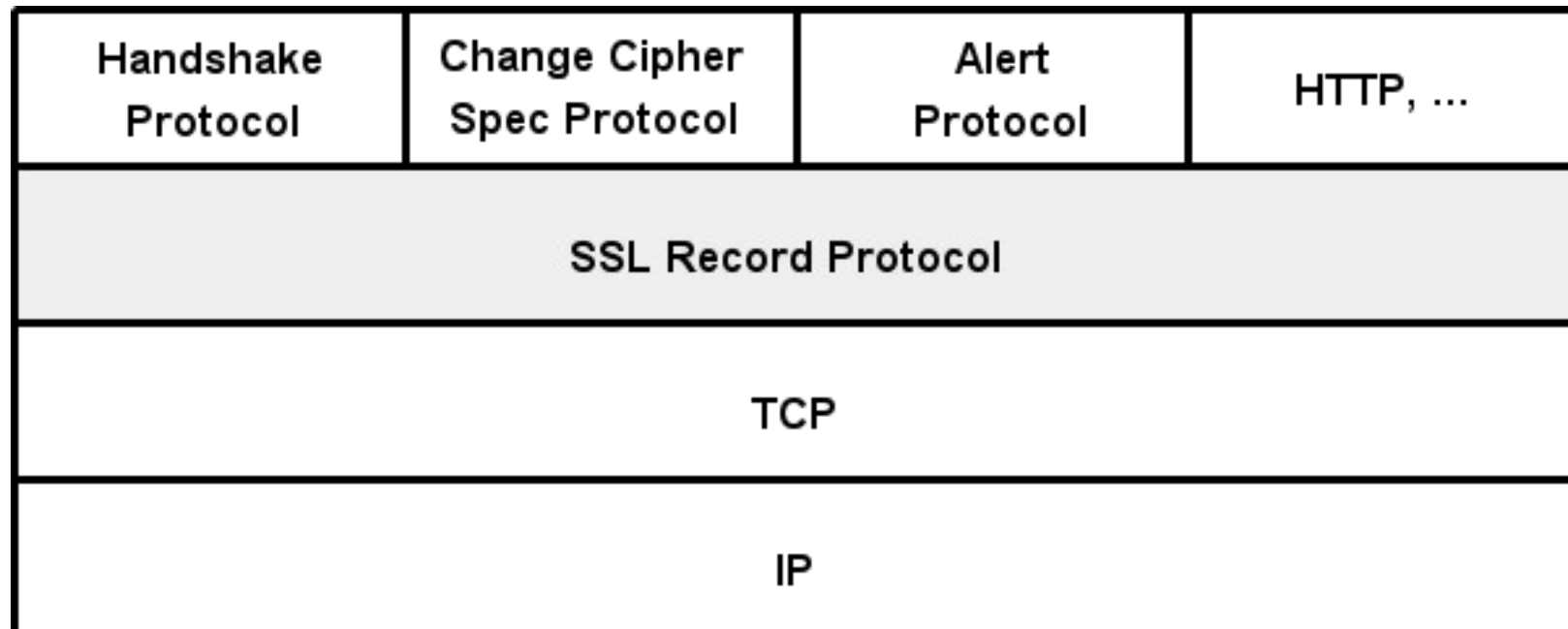
SSL (Secure Socket Layer)

- 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

Communication Systems

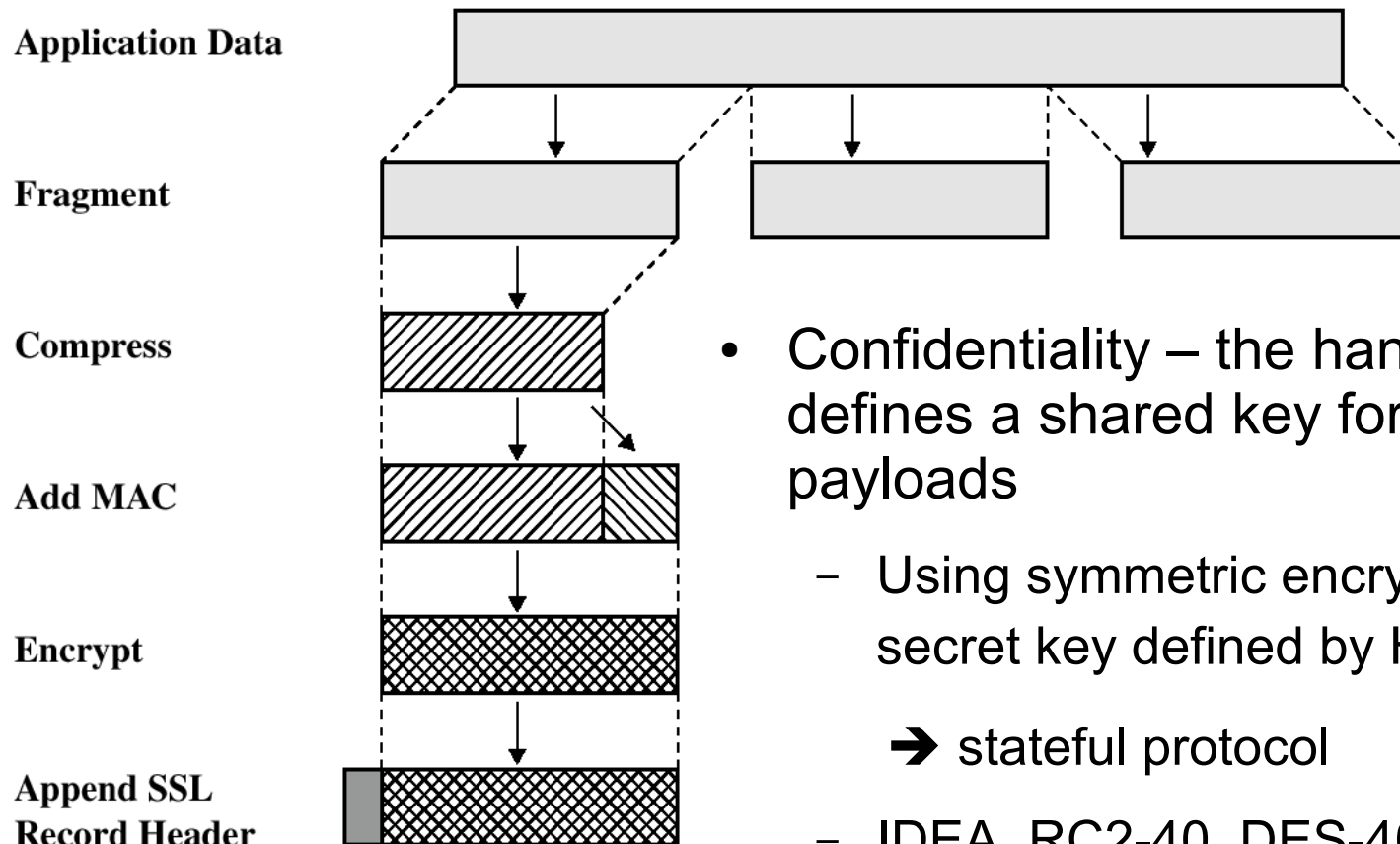
SSL (Secure Socket Layer)

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



Communication Systems

SSL record protocol

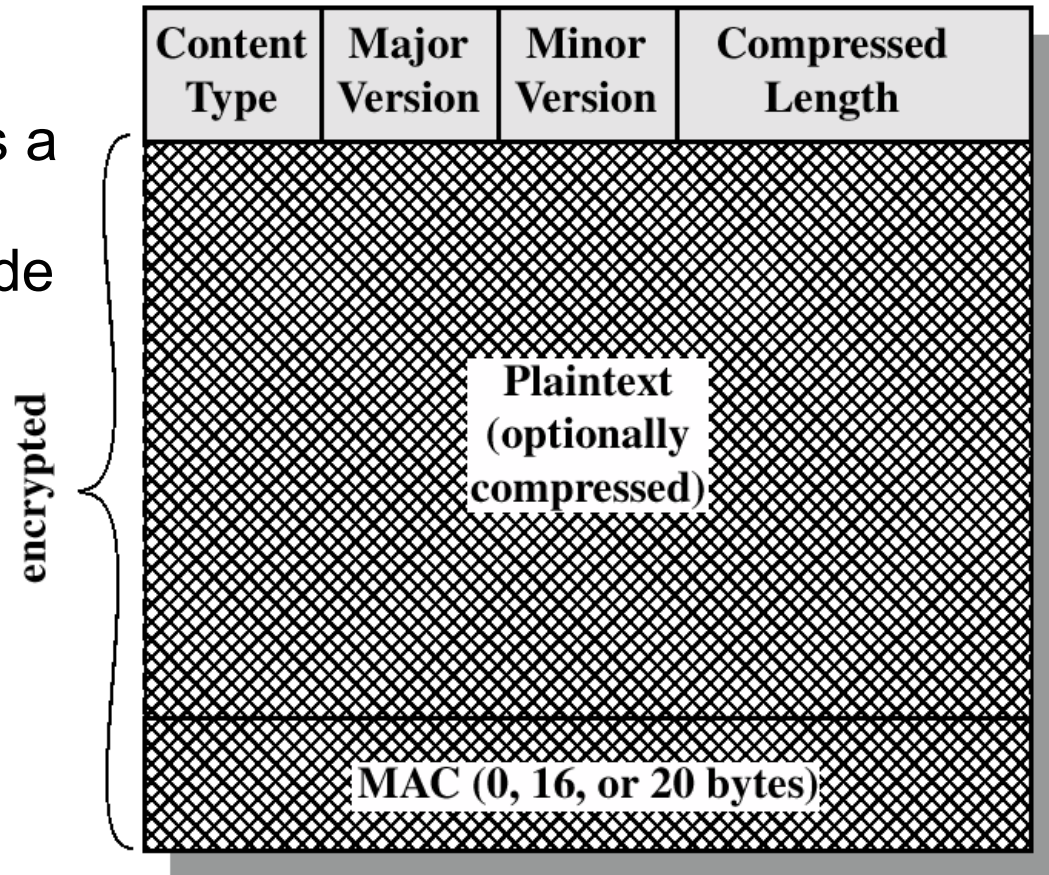


- Confidentiality – the handshake protocol defines a shared key for encryptions of SSL payloads
 - Using symmetric encryption with a shared secret key defined by Handshake Protocol
 - ➔ stateful protocol
 - IDEA, RC2-40, DES-40, DES, 3DES, Fortezza, RC4-40, RC4-128
 - Message is compressed before encryption

Communication Systems

SSL record protocol and format

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



Communication Systems

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

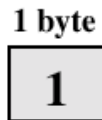
Communication Systems

SSL encryption

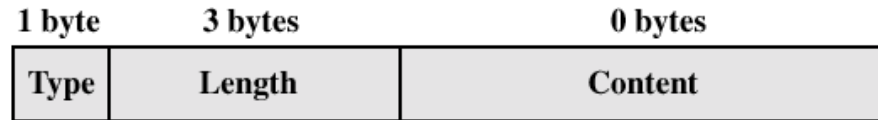
- Fragment size $2^{14} = 16384$ bytes
 - Compression must be lossless and must not increase length more than 1024
 - No compression algorithm specified in SSLv3 – default no compression
 - Block Cipher Encryption Methods
 - IDEA (128) RC2-40, DES-40, DES (56), 3DES (168)
 - Stream Cipher Encryption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

Communication Systems

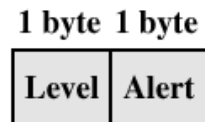
SSL payload / Change Cipher Specification Protocol



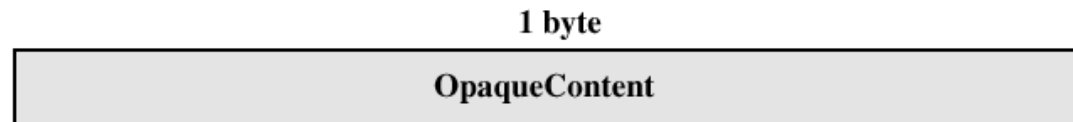
(a) Change Cipher Spec Protocol



(c) Handshake Protocol



(b) Alert Protocol



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

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

Communication Systems

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

Communication Systems

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

Communication Systems

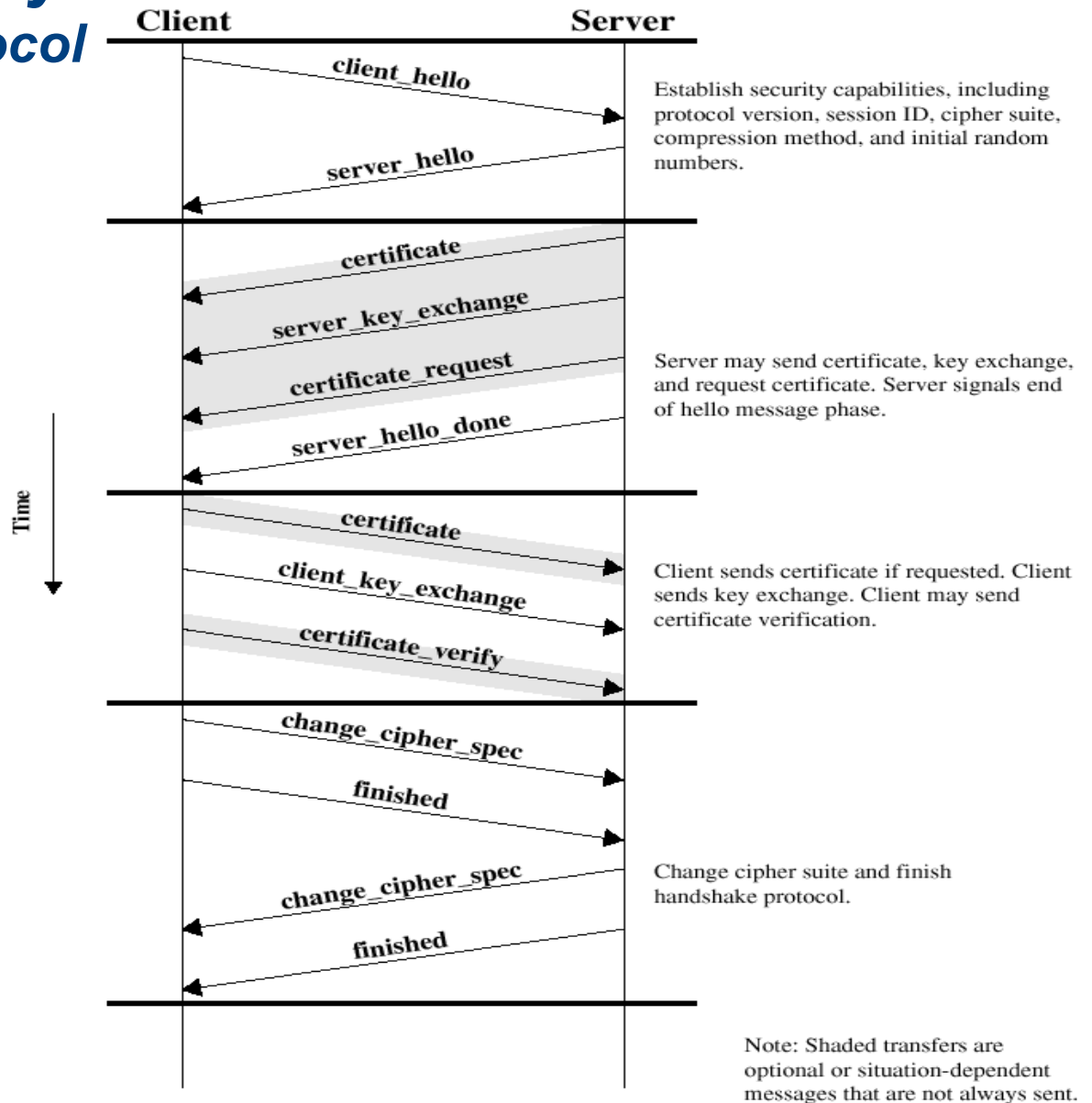
SSL Handshake Protocol – message types

- 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)

Communication Systems

SSL Handshake Protocol

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



Communication Systems

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 → 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

Communication Systems

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
 1. RSA – secret key is encrypted with receiver's RSA public key
 2. Fixed Diffie-Hellman
 3. Ephemeral Diffie Hellman
 4. Anonymous Diffie-Hellman
 5. Fortezza

Communication Systems

SSL Handshake Protocol – Phase 1

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

Communication Systems

SSL Handshake Protocol – Phase 2

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

Communication Systems

SSL Handshake Protocol – Phase 3

- Client Authentication and Key Exchange
- Client verifies server certificate and checks the server hello parameters
 - 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”

Communication Systems

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

Communication Systems

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

Communication Systems

Literature and practical part

- General reading on network security “Security in Computer Networks” (chapt. 7 in Kurose&Ross)
- For next lecture (overview) e.g.: “Understanding PKI – Concepts, Standards, and Deployment Considerations”, 2nd ed. By Adams&Lloyd)
- Lots of online resources

- Start the usual lecture pool environment for experimenting
 - Setup a VLAN with ID 260 and start the DHCP client appropriately
 - As SSH server is used – please change your password of your environment in the virtual machine to avoid unwanted remote administration