Assignments

- Assignment of topic: Friday, November 27th.
- Deadline of first assignment: **Sunday, December 13th, 23:59.**

The deadlines are strict!
Cryptography in the TCP/IP stack

Application layer security (SSH, S-MIME, PGP, ...
Transport layer security (TLS/SSL, ...
Network layer security (IPsec, ...
Data-link layer security (WEP, WPA, WPA2, ...)
Cryptography in the TCP/IP stack

- Application layer security (SSH, S-MIME, PGP, ...)
- Transport layer security (TLS/SSL, ...)
- Network layer security (IPsec, ...)
- Data-link layer security (WEP, WPA, WPA2, ...)

Alice

- Application layer data (HTTP, SMTP...)
- Session (TCP, UDP, ...)
- IP packets
- Frames

Bob

- Application layer
- Transport layer
- Network layer
- Data-link layer
- Physical layer
Data-Link Layer Security

- Encrypt all network packets between network links, e.g., WPA2
- Point-to-point security between network interfaces
- Transparent encryption and decryption for higher layers
- Authentication between endpoints
Data-Link Layer Security

- encrypt all network packets between network links, e.g., WPA2
- point-to-point security between network interfaces
- transparent encryption and decryption for higher layers
- authentication between endpoints
Network Layer Security

- encrypt IP packets, main protocol IPsec
- point-to-point security between entities identified by IP addresses, e.g. routers, firewalls
- routers encrypt and decrypt unnoticed by higher layers
- authentication of routers to each other
Network Layer Security

- encryp IP packets, main protocol IPsec
- point-to-point security between entities identified by IP addresses, e.g. routers, firewalls
- routers encrypt and decrypt unnoticed by higher layers
- authentication of routers to each other
Transport Layer Security

- Encrypt sessions and messages, e.g., TLS/SSL
- Communication between web browser and server, or email clients and servers
- Entities identified by connections, port numbers
- Encrypt and authenticate sessions

Diagram:
- Internet
- Web server
- Application server

Department of Mathematics and Computer Science

Technische Universiteit Eindhoven
University of Technology
Transport Layer Security

- encrypt sessions and messages, e.g. TLS/SSL
- communication between web browser and server, or email clients and servers
- entities identified by connections, port numbers
- encrypt and authenticate sessions
Transport Layer Security

- encrypt sessions and messages, e.g. TLS/SSL
- communication between web browser and server, or email clients and servers
- entities identified by connections, port numbers
- encrypt and authenticate sessions
Current Efforts - Google

GFE = Google Front End Server
SSL Added and removed here!
SSL

PUBLIC INTERNET
Google Cloud

Traffic in clear text here.
Transport Layer Security

Internet

mail server

mail server

/ department of mathematics and computer science
Transport Layer Security

SMTP → transport layer security → mail server

Internet

mail server → IMAP → transport layer security → mail server
Application Layer Security

- add security to standard message formats (e.g. S/MIME, OTR)
- for email: entire link between two user mail clients is protected
- authentication of sender and receiver
- end users have control over their keys (but need to know what they are doing, how to use PKI)
- end-to-end security
Application Layer Security

- add security to standard message formats (e.g. S/MIME, OTR)
- for email: entire link between two user mail clients is protected
- authentication of sender and receiver
- end users have control over their keys (but need to know what they are doing, how to use PKI)
- end-to-end security
Network Layer Security
IPsec
Obvious first reflex: we want end-to-end security
IPsec

- Obvious first reflex: we want end-to-end security
- How many people here regularly encrypt e-mail?
IPsec

- Obvious first reflex: we want end-to-end security
- How many people here regularly encrypt e-mail?
- How many people here already did before their first “Security” lecture?

Problem with application-level security:
- Need to rewrite every single application.
- Need users to switch to secured applications.
- Need users to take care of keys.

Transport-layer security needs applications to be modified to use secure transport layer.

Idea of network-layer security: No need to change applications (or user behavior).

IPsec’s promise: Network security happening without you even noticing.
IPsec

- Obvious first reflex: we want end-to-end security
- How many people here regularly encrypt e-mail?
- How many people here already did before their first “Security” lecture?
- Problem with application-level security: users
  - Need to rewrite every single application.
  - Need users to switch to secured applications.
  - Need users to take care of keys.

Transport-layer security needs applications to be modified to use secure transport layer.

Idea of network-layer security: No need to change applications (or user behavior).

IPsec’s promise: Network security happening without you even noticing.
- Obvious first reflex: we want end-to-end security
- How many people here regularly encrypt e-mail?
- How many people here already did before their first “Security” lecture?
- Problem with application-level security: users
  - Need to rewrite every single application.
  - Need users to switch to secured applications.
  - Need users to take care of keys.
- Transport-layer security needs applications to be modified to use secure transport layer.
IPsec

- Obvious first reflex: we want end-to-end security
- How many people here regularly encrypt e-mail?
- How many people here already did before their first “Security” lecture?
- Problem with application-level security: users
  - Need to rewrite every single application.
  - Need users to switch to secured applications.
  - Need users to take care of keys.
- Transport-layer security needs applications to be modified to use secure transport layer.
- Idea of network-layer security: No need to change applications (or user behavior).
- IPsec’s promise: Network security happening without you even noticing.
IP packet: IP header | IP data (payload)

IPsec was mandatory for IPv6 and is now optional; optional for IPv4.

IPsec provides cryptographic functionality to protect IP packets:
- packet integrity,
- packet origin authentication,
- confidentiality,
- some traffic flow confidentiality,
- protection against replay attacks.

IPsec protocols
- AH - Authentication Header,
- ESP - Encapsulating Security Payload.
IPsec – Modes of Operation

Transport mode:
- only the payload of the IP packet is protected,
- header is not encrypted in ESP, parts of it are authenticated in AH,
- data is protected from source to destination,
- header information is completely in the clear,
- used only between hosts.
IPsec – Modes of Operation

Transport mode:
▶ only the payload of the IP packet is protected,
▶ header is not encrypted in ESP, parts of it are authenticated in AH,
▶ data is protected from source to destination,
▶ header information is completely in the clear,
▶ used only between hosts.

Tunnel mode:
▶ entire IP packet is protected (i.e. IP header and data),
▶ becomes the payload of a new IP packet,
▶ may contain different source and destination addresses,
▶ provides data flow confidentiality to some extent,
▶ can be used between hosts, gateways or host-gateway.
IPsec – Modes of Operation

Transport mode:
- Host communicates directly with another host through a gateway.
- Data is sent directly between hosts through the Internet.

Tunnel mode:
- One host communicates with a gateway, which then communicates with another gateway.
- Data is encapsulated and sent through the Internet to the destination gateway, which then decapsulates the data and forwards it to the destination host.
- Hosts on the local network do not communicate directly with each other; all communications pass through the gateway.
IPsec – Modes of Operation

- Transport mode
  - Host communicates directly with gateway.
  - Internet acts as a透明中介.

- Tunnel mode
  - Host communicates through gateway.
  - Internet acts as a tunnel between gateways.
IPsec – Authentication Header

The Authentication Header provides

▶ data integrity,
▶ authentication of IP packets,
▶ protection against replay attacks.

First two by use of a Message Authentication Code (MAC), e.g. HMAC-SHA1-96.

IP packet is expanded with an AH that contains items such as:

▶ next header — type of the header following this header,
▶ payload length — length of AH,
▶ Security Parameter Index (SPI) — identifies a SA,
▶ sequence number,
▶ authentication data — contains the MAC of the packet, also called Integrity Check Value (ICV).
ICV (truncated HMAC) is computed over:

- immutable IP header fields (fields that do not change in transit), e.g., source address, IP header length,
- Auth. Header (except authentication data field),
- IP data.

Excluded fields are set to zero for HMAC computation.
ICV (truncated HMAC) is computed over:

- immutable IP header fields (fields that do not change in transit), e.g., source address, IP header length,
- Auth. Header (except authentication data field),
- IP data.

Excluded fields are set to zero for HMAC computation.
ICV (truncated HMAC) is computed over:

- immutable IP header fields (fields that do not change in transit), e.g., source address, IP header length,
- Auth. Header (except authentication data field),
- IP data.

Excluded fields are set to zero for HMAC computation.
Anti-replay protection prevents resending copies of authenticated packets.

- Uses sequence number field.
- For each new SA, sequence counter set to 0.
- Keep track of overflow (sequence number is 32 bits), negotiate new SA when counter reaches $2^{32} - 1$.
- Check whether counter is in window of fixed size.
- Right edge = highest sequence number so far received (with valid authentication).
- Mark numbers of received packets with valid authentication.
- Advance window if new sequence number falls to the right of window and packet authenticates.
- Discard packet if number falls to the left of window or packet does not authenticate.
IPsec – Encapsulating Security Payload

The Encapsulating Security Payload provides:

- confidentiality, i.e. encryption with block cipher in CBC mode, e.g. AES-CBC,
- functionality as in AH like authentication, anti-replay (optional).

ESP adds an ESP header, encrypts the payload and adds an ESP trailer. An ESP packet contains:

- security parameter index (SPI),
- sequence number,
- payload data (encrypted),
- padding – to achieve data length a multiple of 32 bits (encrypted),
- padding length (encrypted),
- next header (encrypted),
- (optional) authentication data.
In transport mode, only data is encrypted, i.e. source and destination are in the clear.

In tunnel mode, the whole package is encrypted, i.e. real source and destination addresses are hidden.

Authentication not over IP header fields, only ESP header and data.
In transport mode, only data is encrypted, i.e. source and destination are in the clear.

In tunnel mode, the whole package is encrypted, i.e. real source and destination addresses are hidden.

Authentication not over IP header fields, only ESP header and data.
In transport mode, only data is encrypted, i.e. source and destination are in the clear.

In tunnel mode, the whole package is encrypted, i.e. real source and destination addresses are hidden.

Authentication not over IP header fields, only ESP header and data.
IPsec – Security Associations

- Concept to formalize unidirectional security relationships between two parties.
- Enforce security policy defined in Security Policy Database (SPDB).
- Security Association Database (SADB) contains list of active security associations (SA).

SA parameters:
- sequence number, sequence number overflow,
- anti-replay window,
- AH information: authentication algorithm, key, key lifetime, etc.,
- ESP information: encryption algorithm, key, key lifetime, etc.,
- lifetime of the SA,
- IPsec protocol mode (tunnel, transport),
- maximal packet size.
IPsec – Security Associations

- Concept to formalize unidirectional security relationships between two parties.
- Enforce security policy defined in Security Policy Database (SPDB).
- Security Association Database (SADB) contains list of active security associations (SA).

SA parameters:
- sequence number, sequence number overflow,
- anti-replay window,
- AH information: authentication algorithm, key, key lifetime, etc.,
- ESP information: encryption algorithm, key, key lifetime, etc.,
- lifetime of the SA,
- IPsec protocol mode (tunnel, transport),
- maximal packet size.
The Security Policy Database describes how to treat certain IP packets (BYPASS, DISCARD, PROTECT). Which SA to use for certain traffic is derived from selectors such as

- destination IP address,
- source IP address,
- transport layer protocol,
- source and destination ports.

Most selectors are read off from the IP packet (headers).
The Security Policy Database describes how to treat certain IP packets (BYPASS, DISCARD, PROTECT). Which SA to use for certain traffic is derived from selectors such as

- destination IP address,
- source IP address,
- transport layer protocol,
- source and destination ports.

Most selectors are read off from the IP packet (headers).

A security association can be used

- for one communication direction (bidirectional needs two SAs),
- for AH or ESP; can be combined, e.g. ESP then AH.

SAs are negotiated by public-key mechanisms (see below).
Combining Security Associations

Using ESP in encryption-only mode is insecure:
- manipulate data of inner encrypted IP packets,
- use padding checks to build an ESP trailer oracle.

Degabriele/Paterson *Attacking the IPsec Standards in Encryption-only Configurations*, IEEE Symposium on Security and Privacy 2007
Combining Security Associations

Using ESP in encryption-only mode is insecure:
- manipulate data of inner encrypted IP packets,
- use padding checks to build an ESP trailer oracle.

Degabriele/Paterson *Attacking the IPsec Standards in Encryption-only Configurations*, IEEE Symposium on Security and Privacy 2007

Recall: SA can be combined for AH and ESP if ESP Auth. unavailable.
- For protecting communication against active adversary, need authenticated encryption (security vs. Chosen Ciphertext attacks).
- AH covers more header fields than authentication in ESP, in particular source and destination addresses in the IP header.
- Can do encrypt-then-MAC, i.e. first ESP then AH in transport mode, this gives authenticated encryption.
See RFC 4835 (now obsolete)

- Encryption: block ciphers in Cipher Block Chaining (CBC) mode, must have:
  - NULL encryption (RFC 2410),
  - AES-CBC with 128-bit keys,
  - TripleDES-CBC (168-bit keys).
See RFC 4835 (now obsolete)

- Encryption: block ciphers in Cipher Block Chaining (CBC) mode, must have:
  - NULL encryption (RFC 2410),
  - AES-CBC with 128-bit keys,
  - TripleDES-CBC (168-bit keys).

- Message authentication/integrity: Hash-based Message Authentication Code (HMAC), must have:
  - HMAC-SHA1-96,
  - HMAC-MD5-96.
## Crypto Algorithms (since 2014)

See RFC 7321

<table>
<thead>
<tr>
<th>Old Requirement</th>
<th>New Requirement</th>
<th>Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAY</td>
<td>SHOULD</td>
<td>AES-GCM with a 16 octet ICV</td>
</tr>
<tr>
<td>MAY</td>
<td>SHOULD</td>
<td>AES-GMAC with AES-128</td>
</tr>
<tr>
<td>MUST</td>
<td>MAY</td>
<td>TripleDES-CBC</td>
</tr>
<tr>
<td>SHOULD NOT</td>
<td>MUST NOT</td>
<td>DES-CBC</td>
</tr>
<tr>
<td>SHOULD</td>
<td>SHOULD</td>
<td>AES-XCBC-MAC-96</td>
</tr>
<tr>
<td>SHOULD</td>
<td>MAY</td>
<td>AES-CTR</td>
</tr>
</tbody>
</table>
Crypto Algorithms (since 2014)

See RFC 7321

<table>
<thead>
<tr>
<th>Old Requirement</th>
<th>New Requirement</th>
<th>Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAY</td>
<td>SHOULD</td>
<td>AES-GCM with a 16 octet ICV</td>
</tr>
<tr>
<td>MAY</td>
<td>SHOULD</td>
<td>AES-GMAC with AES-128</td>
</tr>
<tr>
<td>MUST</td>
<td>MAY</td>
<td>TripleDES-CBC</td>
</tr>
<tr>
<td>SHOULD NOT</td>
<td>MUST NOT</td>
<td>DES-CBC</td>
</tr>
<tr>
<td>SHOULD</td>
<td>SHOULD</td>
<td>AES-XCBC-MAC-96</td>
</tr>
<tr>
<td>SHOULD</td>
<td>MAY</td>
<td>AES-CTR</td>
</tr>
</tbody>
</table>

- These are symmetric algorithms, need a pre-shared secret key.
- Different options for key-agreement protocols: PSK, Internet Key Exchange (IKE, IKE2), Kerberos (KINK), IPSECKEY DNS records.
IPsec Problems

- Crypto of IPsec is not really state of the art.
IPsec Problems

- Crypto of IPsec is not really state of the art.
- IPsec ESP allows (in principle) encryption without authentication.
- Consequence: don’t use encrypt-only!
IPsec Problems

- Crypto of IPsec is not really state of the art.
- IPsec ESP allows (in principle) encryption without authentication.
- Consequence: don’t use encrypt-only!
- IPsec AH authenticates IP header (incl. source and dest.).
- NAT changes IP header (source or dest.).
- Possible to get IPsec through NAT, but requires extra effort.
IPsec Problems

- Crypto of IPsec is not really state of the art.
- IPsec ESP allows (in principle) encryption without authentication.
- Consequence: don’t use encrypt-only!
- IPsec AH authenticates IP header (incl. source and dest.).
- NAT changes IP header (source or dest.).
- Possible to get IPsec through NAT, but requires extra effort.
- Most important problem: It’s complicated!
“The first two generations of these documents (principally RFCs 1825–1829, published in 1995, and 2401–2412, published in 1998) are really only intended to provide a guide for implementors and are notoriously complex, difficult to interpret and lacking in overall structure.

... The third and latest incarnation of the core IPsec standards were published as RFCs 4301–4309 in December 2005, and are somewhat more accessible.

... However, the new RFCs are still a long and complex set of documents, totalling over 300 pages.” – Paterson, 2006
“We are of two minds about IPsec. On the one hand, IPsec is far better than any IP security protocol that has come before: Microsoft PPTP, L2TP, etc. On the other hand, we do not believe that it will ever result in a secure operational system. It is far too complex, and the complexity has lead to a large number of ambiguities, contradictions, inefficiencies, and weaknesses. It has been very hard work to perform any kind of security analysis; we do not feel that we fully understand the system, let alone have fully analyzed it.”

– Ferguson, Schneier, 2003
Transport Layer Security
SSL/TLS
Timeline of SSL/TLS

- 1995: SSL 2.0
- 2000: SSL 3.0
- 2005: TLS 1.0
- 2010: TLS 1.1
- 2015: TLS 1.2 refined
- 2020: TLS 1.3 (draft)
Secure Sockets Layer (SSL) and Transport Layer Security (TLS):

- TLS is a variant of SSLv3,
- SSL originally designed for web environment by Netscape,
- design goals: security of web traffic, email, etc.,
- had to work well with HTTP,
- provides transparency for higher layers.
Secure Sockets Layer (SSL) and Transport Layer Security (TLS):

- TLS is a variant of SSLv3,
- SSL originally designed for web environment by Netscape,
- design goals: security of web traffic, email, etc.,
- had to work well with HTTP,
- provides transparency for higher layers.

SSL/TLS provides a secure channel between server and client:

- confidentiality,
- server/client authentication,
- message integrity.
SSL/TLS runs on top of TCP:

- Transparent for application layer protocols,
- SSL/TLS connection acts like a secured TCP connection,
- most protocols running over TCP can be run over SSL/TLS instead, e.g., HTTP → HTTPS, SMTP → SMTPS, ...
SSL/TLS runs on top of TCP:

- Transparent for application layer protocols,
- SSL/TLS connection acts like a secured TCP connection,
- most protocols running over TCP can be run over SSL/TLS instead, e.g., HTTP → HTTPS, SMTP → SMTPS, ...

Protocols in SSL/TLS:

- Record Protocol: data transfer. Compute MAC for integrity, encrypt MAC and data.
- Alert Protocol: alert the other side of exceptional conditions. E.g., errors and warnings.
SSL/TLS Handshake

- Client → Server: ClientHello
  - ClientRandom: random number,
  - Session ID (when resuming a session),
  - List of available CipherSuites:
    pk key exchange, pk auth, sym encryption, hash alg.

Example: TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA256

- ECDH: Elliptic curve Diffie Hellman key exchange.
- ECDSA: Elliptic curve digital signature algorithm.
- AES_128_CBC: AES with 128-bit key in CBC mode.
- SHA256: SHA with 256-bit output for HMAC.
SSL/TLS Handshake (cont.)

- Server → Client: ServerHello
  - ServerRandom: random number,
  - Session ID: implementation specific, random number,
  - Chosen CipherSuite.

- Server → Client: Certificate
  - Server sends server certificate to client, client obtains server’s public key and verifies certificate.

- Server → Client: ServerKeyExchange
  - for DHE: $P^a$, random $a$,
  - for ECDHE: $[a]P$, random $a$,
  - for RSA: –

- (Server → Client: CertificateRequest)

- Server → Client: ServerHelloDone
  - Message marks end of server messages.
SSL/TLS Handshake (cont.)

- Client → Server: ClientKeyExchange
  - for DH(E): $P^b$, random $b$,
  - for ECDH(E): $[b]P$, random $b$,
  - for RSA: random value encrypted with server public key.

- (Client → Server: CertificateVerify)

- Client → Server: ChangeCipherSpec
  - Notify that client switched to new CipherSuite.

- Client → Server: Finished
  - Encrypted Finished massaged containing hash over the previous handshake messages.

For DHE and ECDHE, client and server compute joint session key.
SSL/TLS Handshake (cont.)

- Server → Client: ChangeCipherSpec
  - Notify that client switched to new CipherSuite.
- Server → Client: Finished
  - Encrypted Finished message containing hash over the previous handshake messages.
SSL/TLS Handshake (cont.)

- Server → Client: ChangeCipherSpec
  - Notify that client switched to new CipherSuite.
- Server → Client: Finished
  - Encrypted Finished massaged containing hash over the previous handshake messages.

Interrupted session can be resumed:

- Server and client are supposed to store session ID and MasterSecret,
- client sends session ID in ClientHello,
- reduced protocol: Hello, ChangeCipherSpec and Finished messages,
- new keying data is exchanged,
- new session keys are derived.
TLS 1.2 key derivation

Pseudo-random function:
Define a pseudo-random function (PRF) as

\[
\text{PRF}(\text{secret}, \text{label}, \text{seed}) = P_{\text{SHA256}}(\text{secret, label || seed}).
\]
TLS 1.2 key derivation

Pseudo-random function:

Define a pseudo-random function (PRF) as

\[ \text{PRF}(\text{secret, label, seed}) = P_{\text{SHA256}}(\text{secret, label} \ || \text{seed}). \]

The function \( P \) in PRF is an expansion function defined as

\[ \text{P}_H(\text{secret, seed}) = \text{HMAC}_H(\text{secret, A(1)} \ || \text{seed}) \ || \text{HMAC}_H(\text{secret, A(2)} \ || \text{seed}) \ || \ldots \]

and is iterated as often as necessary. The A(i) are defined as

\[ A(0) = \text{seed}, A(i) = \text{HMAC}_H(\text{secret, A(i-1)}). \]
Figure of the PRF at http://www.cs.bham.ac.uk/~mdr/teaching/modules06/netsec/lectures/tls/tls.html.
The message authentication code HMAC$_H$ is constructed from a hash function $H$ as

$$HMAC_H(k, m) = H((k \oplus opad) \ || \ H((k \oplus ipad) \ || \ m))$$

where

- $opad$ is the outer padding ($0x5c5c5c\ldots5c5c$),
- and $ipad$ is the inner padding ($0x363636\ldots3636$).
HMAC$_{SHA1}$

- XOR
  - i key pad
  - message
  - hash sum 1
  - hash sum 2

- XOR
  - o key pad
  - hash sum 1

- SHA1 - 1st pass
- SHA1 - 2nd pass

64 Byte
20 Byte
<= 64 Byte
<= 64 Byte
Computing the MasterSecret:

\[
\text{MasterSecret} = \text{PRF(PreMasterSecret, "master secret", ClientRandom || ServerRandom)[0..47];}
\]
**TLS 1.2 key derivation**

**Computing the MasterSecret:**

\[
\text{MasterSecret} = \text{PRF(PreMasterSecret, "master secret", ClientRandom || ServerRandom)[0..47]};
\]

**Computing the MAC keys for the Finished messages:**

\[
\text{PRF(MasterSecret, FinishedLabel, SHA256(handshake-messages))[0..95]}
\]

FinishedLabel: "client finished" of "server finished".
Computing the KeyBlock for session keys:

\[
\text{KeyBlock} = \text{PRF(MasterSecret, "key expansion", ServerRandom || ClientRandom)}
\]

Compute as many bits as needed to obtain six values from the key block:

- client MAC key,
- server MAC key,
- client encryption key,
- server encryption key,
- client IV, server IV.
SSL/TLS Record Protocol

Record protocol to exchange encrypted and authenticated data:

- Payload data is split into fragments which are protected and transmitted independently; when received, fragments are decrypted and verified independently.
- Each fragment is authenticated with a MAC which is appended; MAC is over a sequential number (anti-replay) and the content.
- Data fragment and MAC are encrypted.
- A record header is attached to the encrypted data, containing information necessary for interpreting the record such as type of data (e.g. Handshake or ApplicationData), length, and SSL version.
- (header || encrypted fragment and MAC) is sent.
# SSL/TLS Record

<table>
<thead>
<tr>
<th>Type</th>
<th>ChangeCipherSpec</th>
<th>Alert</th>
<th>Handshake</th>
<th>Application</th>
<th>Heartbeat</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hex</td>
<td>0x14</td>
<td>0x15</td>
<td>0x16</td>
<td>0x17</td>
<td>0x18</td>
</tr>
<tr>
<td>Dec</td>
<td>20</td>
<td>21</td>
<td>22</td>
<td>23</td>
<td>24</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Byte 0</th>
<th>Content type</th>
<th>Byte +1</th>
<th>Byte +2</th>
<th>Byte +3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bytes 1...4</td>
<td>Version (Major)</td>
<td>(Minor)</td>
<td>Length (bits 15..8)</td>
<td>Length (bits 7..0)</td>
</tr>
<tr>
<td>Bytes m...(p−1)</td>
<td>Protocol message(s)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bytes p...(q−1)</td>
<td>MAC (optional)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Padding (block ciphers only)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# SSL/TLS — Authentication, Key Exchange

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>SSL 2.0</th>
<th>SSL 3.0</th>
<th>TLS 1.0</th>
<th>TLS 1.1</th>
<th>TLS 1.2</th>
<th>TLS 1.3</th>
</tr>
</thead>
<tbody>
<tr>
<td>RSA</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>DH-RSA</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>DHE-RSA</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>ECDH-RSA</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>ECDH-ECDSA</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>DH-DSS</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>DHE-DSS</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>ECDHE-ECDSA</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>ECDHE-ECDSA</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>PSK</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>PSK-RSA</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>DHE-PSK</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes*</td>
</tr>
<tr>
<td>ECDHE-PSK</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes*</td>
</tr>
<tr>
<td>SRP</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>SRP-DSS</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>SRP-RSA</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Kerberos</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>DH-ANON</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>ECDH-ANON</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>GOST</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

*Session Resumption
<table>
<thead>
<tr>
<th>Cipher</th>
<th>Strength (bits)</th>
<th>SSL 2.0</th>
<th>SSL 3.0</th>
<th>TLS 1.0</th>
<th>TLS 1.1</th>
<th>TLS 1.2</th>
<th>TLS 1.3</th>
</tr>
</thead>
<tbody>
<tr>
<td>AES GCM</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>Secure</td>
<td>Secure</td>
<td></td>
</tr>
<tr>
<td>AES CCM</td>
<td>256, 128</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>Secure</td>
<td>Secure</td>
<td></td>
</tr>
<tr>
<td>AES CBC</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>Depends</td>
<td>Secure</td>
<td>Secure</td>
<td></td>
</tr>
<tr>
<td>Camellia GCM</td>
<td>256, 128</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>Secure</td>
<td>Secure</td>
<td></td>
</tr>
<tr>
<td>Camellia CBC</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>Depends</td>
<td>Secure</td>
<td>Secure</td>
<td></td>
</tr>
<tr>
<td>ARIA GCM</td>
<td>256, 128</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>Secure</td>
<td>Secure</td>
<td></td>
</tr>
<tr>
<td>ARIA CBC</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>Depends</td>
<td>Secure</td>
<td>Secure</td>
<td></td>
</tr>
<tr>
<td>SEED CBC</td>
<td>128</td>
<td>N/A</td>
<td>N/A</td>
<td>Depends</td>
<td>Secure</td>
<td>Secure</td>
<td></td>
</tr>
<tr>
<td>3DES EDE CBC</td>
<td>112</td>
<td>Insecure</td>
<td>Insecure</td>
<td>Low/Dep.</td>
<td>Low</td>
<td>Low</td>
<td></td>
</tr>
<tr>
<td>GOST CNT</td>
<td>256</td>
<td>N/A</td>
<td>N/A</td>
<td>Secure</td>
<td>Secure</td>
<td>Secure</td>
<td></td>
</tr>
<tr>
<td>IDEA CBC</td>
<td>128</td>
<td>Insecure</td>
<td>Insecure</td>
<td>Depends</td>
<td>Secure</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>DES CBC</td>
<td>40</td>
<td>Insecure</td>
<td>Insecure</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>RC2 CBC</td>
<td>56</td>
<td>Insecure</td>
<td>Insecure</td>
<td>Insecure</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>ChaCha20-Poly1305</td>
<td>256</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>Secure</td>
<td>Secure</td>
<td></td>
</tr>
<tr>
<td>RC4</td>
<td>40</td>
<td>Insecure</td>
<td>Insecure</td>
<td>Insecure</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>NULL</td>
<td>–</td>
<td>N/A</td>
<td>Insecure</td>
<td>Insecure</td>
<td>Insecure</td>
<td>Insecure</td>
<td></td>
</tr>
</tbody>
</table>
Choice of topic: before Thursday, November 26th, 23:59.
Assignment of topic: Friday, November 27th.

The deadlines are strict!