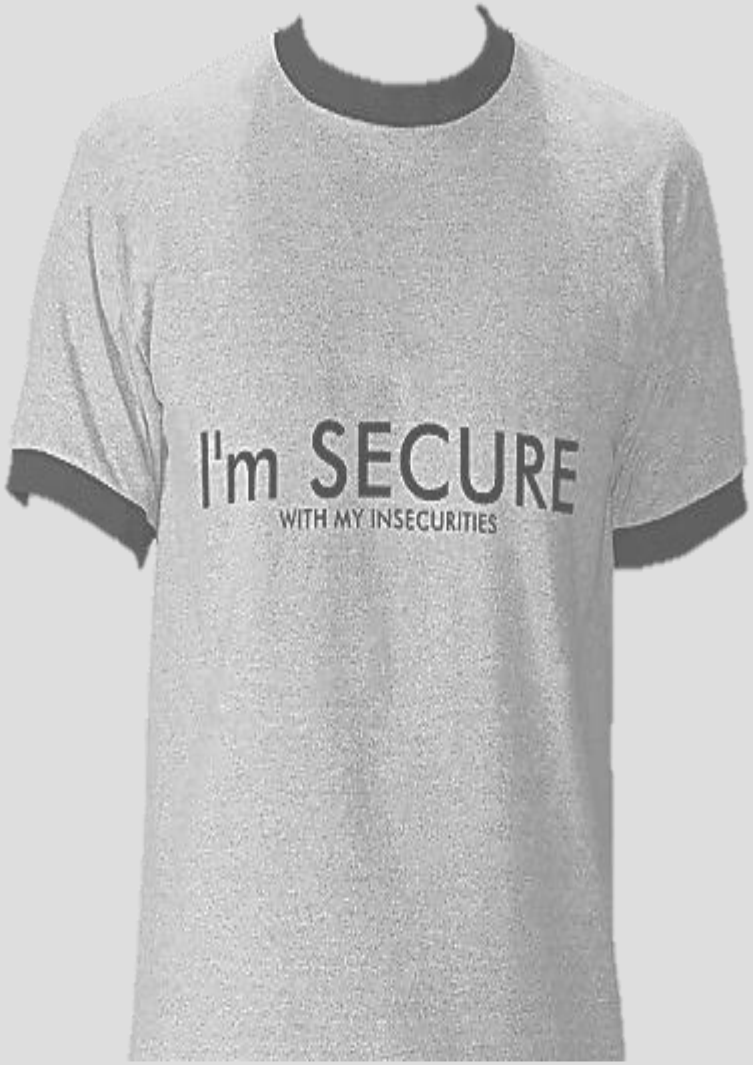
Secure Communications

Securing Internet Traffic



Mark Cummins
Institute of Technology Blanchardstown

mark.cummins@itb.ie
+353 1 885 1156

Launch of the Web Browsers

The first web browser, WorldWideWeb, launches in 1991.

Mosaic web browser in 1993 – one of the first graphical web browsers – led to an explosion in web use.

Netscape Navigator is released in 1994, which quickly became the world's most popular browser, accounting for 90% of all web use at its peak

Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156

Launch of the Web Browsers

Microsoft responded with its Internet Explorer in 1995, initiating the industry's first browser war

Opera debuted in 1996; although it has never achieved widespread use

First Attempts at Internet Security

- Early research efforts toward transport layer security included the **Secure Network Programming** (SNP) API, at the University of Texas.

- In 1993 they explored the approach of having a secure transport layer API closely resembling Berkeley sockets, to facilitate retrofitting preexisting network applications with security measures

Introducing SSL

- As there was not yet a standard for securing HTTP when use of the web for commercial transactions began, SSL was developed.
- The SSL protocol was originally developed by Netscape (the leading web browser at the time)
- It was designed to prevent eavesdropping, tampering, or message forgery for communicating client/server applications.

Several Versions of SSL

- Version 1.0 was never publicly released
- Version 2.0 was released in February 1995 but "contained a number of security flaws (depreciated 2011 by IETF)
- Which ultimately led to the design of SSL version 3.0". (depreciated 2015 by IETF)
- SSL version 3.0 was released in 1996.

Introducing SSH

In 1995, Tatu Ylönen, a researcher at Helsinki University of Technology, Finland, designed the first version of the protocol (now called **SSH-1**)

The goal of SSH was to replace the earlier rlogin, TELNET and rsh protocols, which did not provide strong authentication or guarantee confidentiality.

Introducing TLS

TLS 1.0 was first defined as an upgrade to SSL Version 3.0.

As stated in the RFC, "the differences between this protocol and SSL 3.0 are not dramatic, but they are significant enough that TLS 1.0 and SSL 3.0 do not interoperate.“

TLS 1.0 does include a means by which a TLS implementation can downgrade the connection to SSL 3.0.

itb
Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156

Several Versions of TLS

Like its predecessor , SSL, TLS also has several versions

- TLS 1.0 (SSL3.1) was defined in RFC 2246 in January 1999
- TLS 1.1 (SSL 3.2) was defined in RFC 4346 in April 2006
- TLS 1.2 (SSL 3.3) was defined in RFC 5246 in August 2008

TLS 1.3
working draft
released july
2017

Introducing IPsec

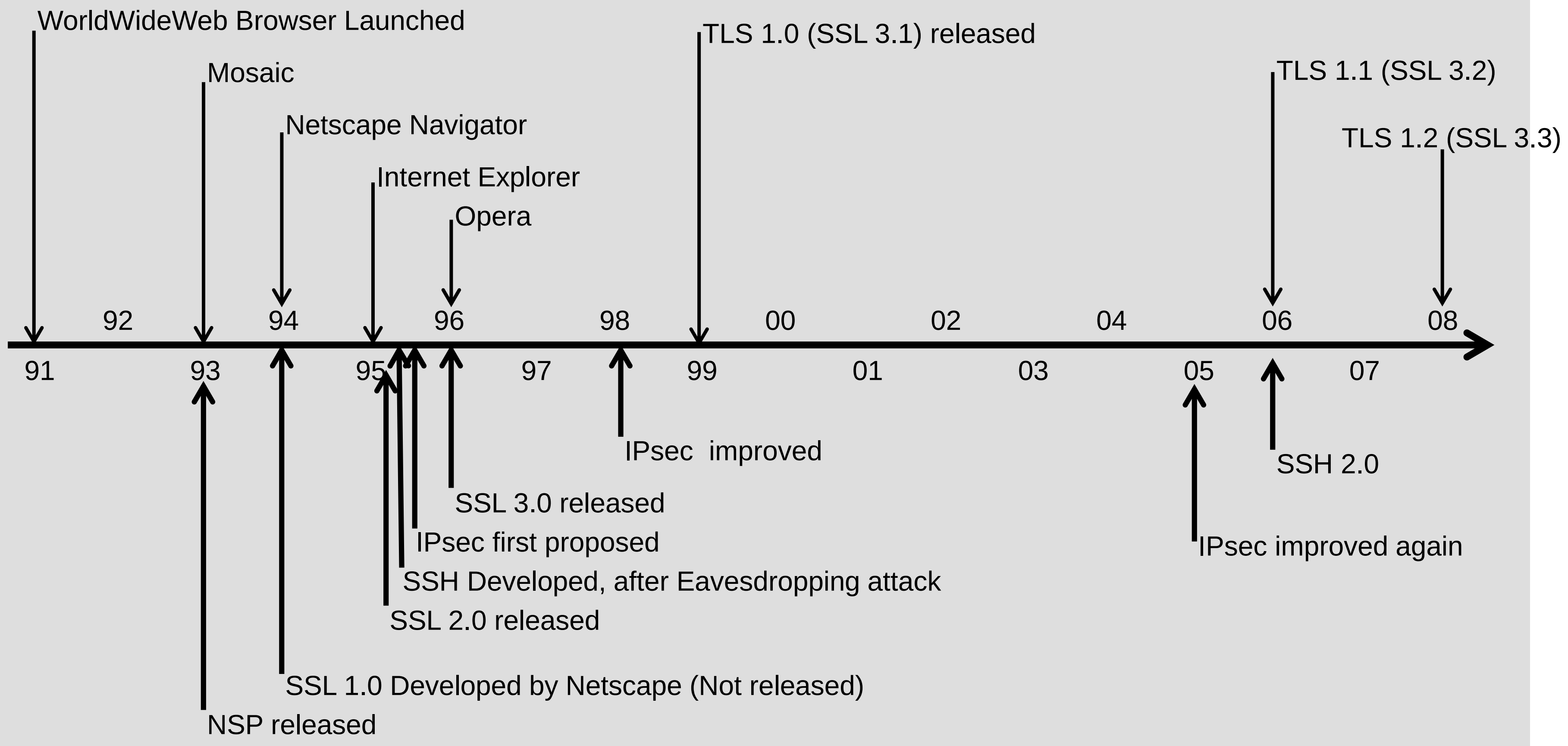
IPsec was developed in conjunction with IPv6

IPsec protocols were originally defined in RFC 1825 and RFC 1829, published in 1995.

In 1998, these documents were superseded by RFC 2401 and RFC 2412

In December 2005, new standards were defined in RFC 4301 and RFC 4309 which are largely a superset of the previous editions

Timeline of secure protocols



Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156

What does SSL/TLS give us?

It allows the connection between two mediums (client-server) to be encrypted. By encrypting the communication, we ensure that no third-party is able to read or tamper with the data that is being exchanged on our connection with the server.

With the use of Public Key Cryptography, SSL/TLS provides identification between the communicating parties. This means that one (most commonly the client), or both parties, know who they are communicating with.

What does SSL/TLS give us?

Perfect Forward Secrecy

Simply put, PFS’s primary job is to make sure that in the event of the private key of a server being compromised, an attacker will not be able to decrypt any previous TLS communications.

Perfect Forward Secrecy is possible by using **Diffie-Hellman ephemeral key exchange**, which provides new keys for every session and are valid as long as the session is active.

Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156

What does SSL/TLS give us?

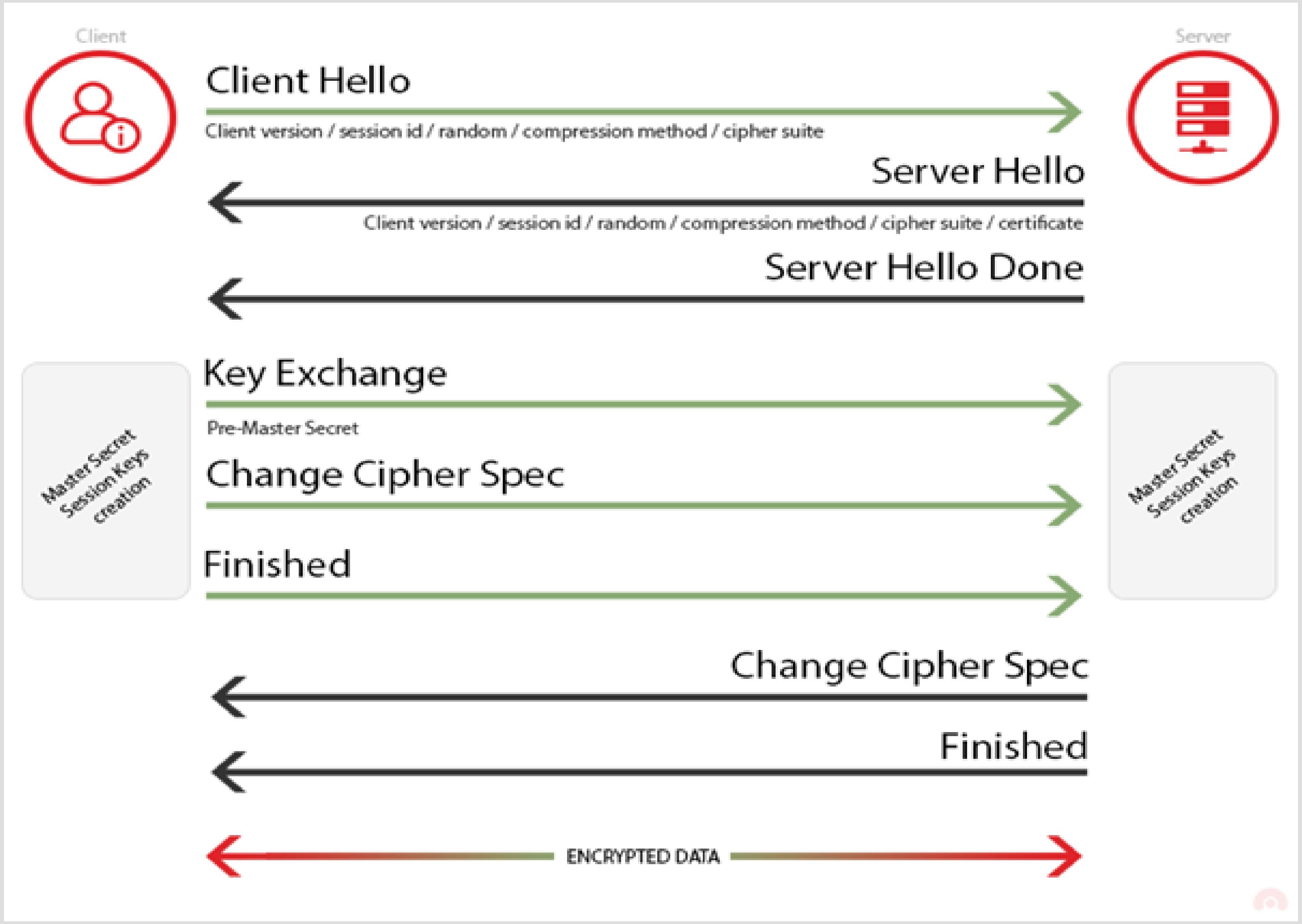
Application Security

SSL/TLS protocol can be used in different services such as web, mail, FTP, VoIP and VPN. Typically,

when a service uses a secure connection the letter S is appended to the service’s protocol name. For example: HTTPS, SMTPS, FTPS, SIPS.

Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156

SSL/TLS Handshake


Client
Server
Client Hello
Client version / session id / random / compression method / cipher suite
Server Hello
Client version / session id / random / compression method / cipher suite / certificate
Server Hello Done
Key Exchange
Pre-Master Secret
Change Cipher Spec
Finished
Master Secret Session Keys creation
Master Secret Session Keys creation
Change Cipher Spec
Finished
ENCRYPTED DATA

Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156

SSL/TLS Handshake

1. The client starts the handshake with a **Client Hello**

Sending the following details:

- **SSL/TLS protocol versions it supports**
- **Compression method.**
- **Cipher suites.**
- **Session_id**

The client can request additional functionality for the connection and this can be done via “**extensions**”. If the server cannot provide the additional functionality, then the client can abort the handshake if needed.

Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156

SSL/TLS Handshake

1. The client starts the handshake with a **Client Hello**

   The client will send a list of all the SSL/TLS protocol **versions it supports**, with the preferred one being first on the list. The preferred one is usually, and should be, the latest available version.

   Client will also send the session id which will be used for the connection. If the **session_id** is not empty, the server will search for previously cached sessions and resume that session if a match is found.

Mark Cummins
Institute of Technology Blanchardstown

mark.cummins@itb.ie
+353 1 885 1156

SSL/TLS Handshake

1. The client starts the handshake with a **Client Hello**

   The client will also send the **Compression method**. This is the method that is going to be used for compressing the SSL packets. By using compression, we can achieve lower bandwidth usage and therefore, faster transfer speeds.

   And the Client will also send its preferred **cipher suites**. Cipher suites are combination of cryptographic algorithms which are used to define the overall security of the connection to be established.

SSL/TLS Handshake

1. The client starts the handshake with a **Client Hello**

   A 32-byte **random** data of which 4 bytes represent the client's current datetime (in epoch format) and the remaining 28 bytes, a randomly generated number.

   The Client's random and Server's random will later be used to generate the key which the data will be encrypted with.

Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156

SSL/TLS Handshake

A sample cipher suite is the following:
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256

Let’s break this down so that it makes more sense.

- TLS is the protocol being used
- ECDHE is the key exchange algorithm
  (Elliptic curve Diffie–Hellman)
- ECDSA is the authentication algorithm
  (Elliptic Curve Digital Signature Algorithm)
- AES_128_GCM is the data encryption algorithm
  (Advanced Encryption Standard 128 bit Galois/Counter Mode)
- SHA256 is the Message Authentication Code (MAC) algorithm
  (Secure Hash Algorithm 256 bit)

Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156

SSL/TLS Handshake

2. The second step will be to reply with a **Server Hello**

After the server receives the Client Hello, If it finds an acceptable set of algorithms in the client’s request, it will respond by accepting those options and provide it’s certificate.

If the server does not meet the requirements of the client it will respond with a **handshake failure message**.

Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156

SSL/TLS Handshake

The **Server Hello** to the client includes the following:

- server_version
- random
- session_id
- compression_methods
- cipher_suites
- certificate
- Client Certificate (optional)
- Server Key Exchange
- Server Hello Done

Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156

SSL/TLS Handshake

The **Client Key Exchange** message is sent right after the **Server Hello Done** is received from the server.

If the server requests a Client Certificate, the Client Key Exchange will be sent after that.

During this stage, the client will create a pre-master key.

Before sending the pre-master secret to the server, the client encrypts it using server’s public key.

SSL/TLS Handshake

After the server receives the **pre-master secret key**, it uses its private key to decrypt it. Now both client and server will compute the master-secret key based on the random values exchanged earlier (ClientRandom and ServerRandom) using a Pseudorandom Function (PRF).

The **master-secret key**, which is 48 bytes in length, will then be used by both client and server to symmetrically encrypt the data for the rest of the communication.

Both Client and Server will use the master secret to generate the sessions keys which will be to encrypt/decrypt the data.

SSL/TLS Handshake

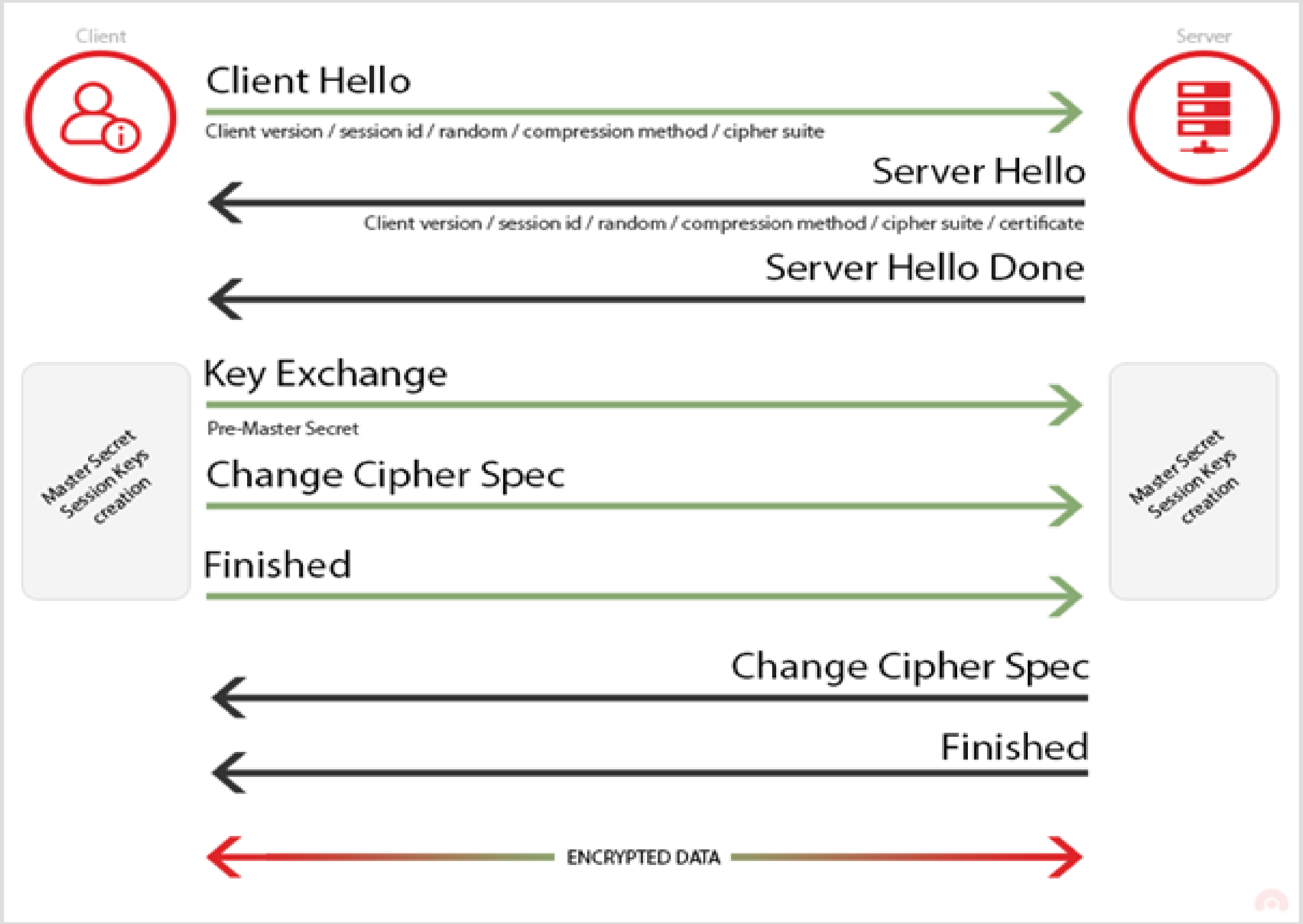
At this point, both the Client and the Server are ready to switch to a secure, encrypted environment.

The Change Cipher Spec protocol is used to change the encryption being used by the client and server.

Any data being exchanged between the two parties will now be encrypted with the symmetrical shared key they have.

The last message of the handshake process and first encrypted one in the secure connection is **Finished**, which is exchanged by both parties.

SSL/TLS Handshake


Client
Server
Client Hello
Client version / session id / random / compression method / cipher suite
Server Hello
Client version / session id / random / compression method / cipher suite / certificate
Server Hello Done
Key Exchange
Pre-Master Secret
Change Cipher Spec
Finished
Master Secret Session Keys creation
Master Secret Session Keys creation
Change Cipher Spec
Finished
ENCRYPTED DATA

Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156

Attacks against SSL/TLS

In February 2015, IETF issued an informational RFC summarizing the various known attacks against TLS/SSL.

Read the RFC on Moodle

Mark Cummins
Institute of Technology Blanchardstown
s@itb.ie
+353 1 885 1156

Attacks against SSL/TLS

Attack	SSL 2	SSL 3	TLS 1	TLS 2	TLS 3
Renegotiation attack (2009)	NO	YES	YES	YES	YES
Downgrade attacks: FREAK attack (2014) Logjam attack (2015)	Certain OSs and Browsers using openSSL				
Drown attacks (2016)	Attacks Server Configurations				
BEAST (2011) CBC Vul in TLS 1	NO	YES	YES	NO	NO
CRIME (2012) (compression)	NO	NO	YES	YES	YES
BREACH (2013) (Crime on HTTP)					
Lucky Thirteen attack (2013)	Padding/Timing attack				
Poodle Attack (2014) CBC	NO	YES	NO	NO	NO
HeartBleed (2014)	Certain versions of OpenSSL				

Security issues with SSL

SSL 2.0 had many security weaknesses which SSL 3.0 aims to fix.

In export weakened modes, SSL 2.0 unnecessarily weakens the authentication keys to 40 bits.

SSL 2.0 uses a weak MAC construction,

SSL 2.0 feeds padding bytes into the
MAC in block cipher modes, but leaves the padding length field unauthenticated

Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156

Security issues with SSL

There is a ciphersuite rollback attack,
where an active attacker edits the list of ciphersuite preferences in the hello messages to invisibly force both endpoints to use a weaker form of encryption than they otherwise would choose

From a security standpoint, SSL 3.0 should be considered less desirable than TLS 1.0.

The SSL 3.0 cipher suites have a weaker key derivation process

Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156

Security issues with TLS

A vulnerability of the renegotiation procedure was discovered in August 2009 that can lead to plaintext injection attacks against SSL 3.0 and all current versions of TLS.

There are some attacks against the implementation rather than the protocol itself

Security issues with TLS

On September 23, 2011 researchers Thai Duong and Juliano Rizzo demonstrated a "proof of concept" called BEAST (using a Java Applet to violate "same origin policy" constraints) for a long-known CBC vulnerability in TLS 1.0.

Google included a patch in development version of their Google Chrome web browser to mitigate BEAST-like attacks.

Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156

Security issues with TLS

The TLS 1.0-specific BEAST attack can be prevented by removing all CBC ciphers from your list of allowed ciphers—leaving only the RC4 cipher, which is still widely supported on most websites.

Microsoft is releasing tools to support TLS 1.1 (which fixes this CBC attack vulnerability) on Microsoft servers and browsers.

Security issues with TLS

The tool is based on a blockwise-adaptive chosen-plaintext attack, a man-in-the-middle approach that injects segments of plain text sent by the target's browser into the encrypted request stream to determine the shared key.

The code can be injected into the user's browser through JavaScript associated with a malicious advertisement

Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156

Security issues with TLS

Using the known text blocks, BEAST can then use information collected to decrypt the target's AES-encrypted requests, including encrypted cookies, and then hijack the no-longer secure connection.

That decryption happens slowly, however; BEAST currently needs sessions of at least a half-hour to break cookies using keys over 1,000 characters long.

Security issues with TLS

The attack, according to Duong, is capable of intercepting sessions with PayPal and other services that still use TLS 1.0—which would be most secure sites, since follow-on versions of TLS aren't yet supported in most browsers or Web server implementations.

A bit more about TLS/SSL

In applications design, TLS is usually implemented on top of any of the Transport Layer protocols, encapsulating the application-specific protocols such as HTTP, FTP, SMTP, NNTP and XMPP.

Historically it has been used primarily with reliable transport protocols such as TCP.

Mark Cummins
Institute of Technology Blanchardstown

mark.cummins@itb.ie
+353 1 885 1156

A bit more about TLS/SSL

However, it has also been implemented with datagram-oriented transport protocols, such as UDP and the Datagram Congestion Control Protocol (DCCP), usage which has been standardized independently using the term Datagram Transport Layer Security (DTLS).

A prominent use of TLS is for securing World Wide Web traffic carried by HTTP to form HTTPS.

A bit more about TLS/SSL

Increasingly, SMTP is also protected by TLS.

TLS can also be used to tunnel an entire network stack to create a VPN, as is the case with OpenVPN.

When compared against traditional IPsec VPN technologies, TLS has some inherent advantages in firewall and NAT traversal that make it easier to administer for large remote-access populations.

itb
Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156

A bit more about TLS/SSL

TLS is also a standard method to protect Session Initiation Protocol (SIP) application signalling.

TLS can be used to provide authentication and encryption of the SIP signaling associated with VoIP and other SIP-based applications.

Mark Cummins
Institute of Technology Blanchardstown

mark.cummins@itb.ie
+353 1 885 1156

So how does SSL/TLS work?

The TLS protocol allows client/server applications to communicate across a network in a way designed to prevent eavesdropping and tampering.

A TLS client and server negotiate a stateful connection by using a handshaking procedure.

During this handshake, the client and server agree on various parameters used to establish the connection's security.

Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156

So how does SSL/TLS work?

The handshake begins when a client connects to a TLS-enabled server requesting a secure connection and presents a list of supported CipherSuites (ciphers and hash functions).

From this list, the server picks the strongest cipher and hash function that it also supports and notifies the client of the decision.

The server sends back its identification in the form of a digital certificate. The certificate usually contains the server name, the trusted certificate authority (CA) and the server's public encryption key.

Mark Cummins
Institute of Technology Blanchardstown

mark.cummins@itb.ie
+353 1 885 1156

So how does SSL/TLS work?

The client may contact the server that issued the certificate and confirm the validity of the certificate before proceeding.

In order to generate the session keys used for the secure connection, the client encrypts a random number with the server's public key and sends the result to the server.

From the random number, both parties generate key material for encryption and decryption.

So how does SSL/TLS work?

This concludes the handshake and begins the secured connection, which is encrypted and decrypted with the key material until the connection closes.

If any one of the above steps fails, the TLS handshake fails and the connection is not created.

Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156

Who supports SSL/TLS?

All the most recent web browsers support TLS: Apple's Safari supports TLS, but it’s not officially specified which version.

On operating systems (Safari uses the TLS implementation of the underlying OS)

Mozilla Firefox, versions 2 and above, support TLS 1.0. As of September 2011, Firefox does not support TLS 1.1 or 1.2.

itb

Mark Cummins
Institute of Technology Blanchardstown

mark.cummins@itb.ie
+353 1 885 1156

Who supports SSL/TLS?

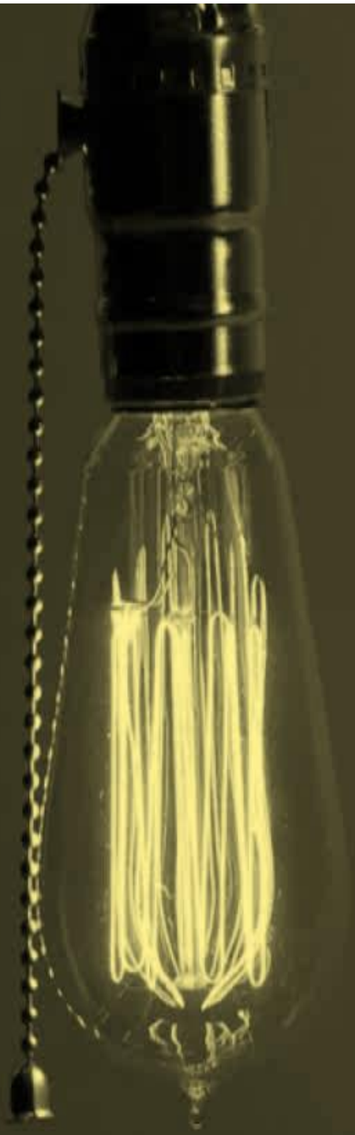
Microsoft Internet Explorer always uses the TLS implementation of the underlying Microsoft Windows Operating System

IE8 in Windows 7 and Windows Server 2008 R2 supports TLS 1.2.

As of Presto 2.2, featured in Opera 10, Opera supports TLS 1.2.

Google's Chrome browser supports TLS 1.0, but not TLS 1.1 or 1.2.

Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156



Thank You