TLS

Transport Layer Security.

TLS es capa de aplicación en TCP / IP y capa de presentación en el modelo OSI.

# Características

* **Evolución del SSL** (Secure Socket Layer) el cual actualmente es obsoleto.
* Ampliamente utilizado para proteger tráfico web entre un servidor HTTP y un browser. La idea es encriptar la capa de transporte.
* Usa encriptación simétrica y asimétrica. Primero se realiza un intercambio de claves de manera asimétrica (más segura) y una vez establecida la conexión segura, se pasa a la encripción simétrica para la transferencia de datos (más eficiente).

# Handshake

Funciona en el puerto 443 (HTTP sobre TLS = HTTPS).

1. En azul: Se establece el Handshake de 3 vías de TCP (SYN, SYN-ACK, ACK).
2. En naranja, el intercambio TLS:
   * En el ACK el cliente va enviar un mensaje de TLS llamado ClientHello, indicando la solicitud de una conexión segura
   * El servidor contesta con ServerHello y le presenta un certificado, que el cliente va a poder usar para verificar que el servidor es quien dice ser.
   * El cliente verifica el certificado y manda al servidor su CipherSpec (cuales son los mecanismos de encripción que soporta).
   * El servidor responde indicándole que encripción utilizaran.
3. En gris: Comienza la transmisión de datos con HTTP con encripción simétrica (GET, POST, etc).

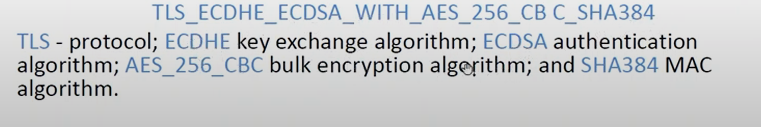
Se usan muchos mensajes. Esto puede ser problemático si los servidores están muy lejos entre sí (geográficamente).

## Explicación de mensajes:

* Client Hello: Indica la solicitud de una conexión segura.
* CipherSuite: es una lista de algoritmos criptográficos ordenados por orden de preferencia. El servidor elegirá el mayor que pueda soportar. Es una combinación de:
  + key exchange algorithm – cómo se intercambiarán las claves simétricas.
  + authentication algorithm – cómo se autenticará.
  + bulk encryption algorithm – algoritmo de clave simétrica a utilizar.
* Message Authentication Code (MAC) – método para chequear integridad.
* Server Hello: El servidor acepta utilizar TLS en esta conexión, informando el Cipher seleccionado.
* Certificate: El servidor envía la cadena de certificados.
* Server Key Exchange / Client Key Exchange: Permite a ambos interlocutores encriptar el intercambio de claves-mensajes.
* Change Cipher Spec: Estos mensajes finalizan el handshake.

La autenticación del cliente (TLS mutual authentication) es opcional. EL servidor puede solicitarle al cliente su certificado luego del Server Hello.

Ejemplo, con los diferentes mecanismos de encripcion:



# Certificados

El certificado es un objeto que identifica el servidor/cliente con varios atributos, entre ellos:

* Issuer / Emisor. El certificado es legítimo si fue emitido por una autoridad certificante confiable para el cliente. El issuer o certificate authority o CA es el emisor.
* Subject y Subject alternative name (SAN): indican a quién fue extendido

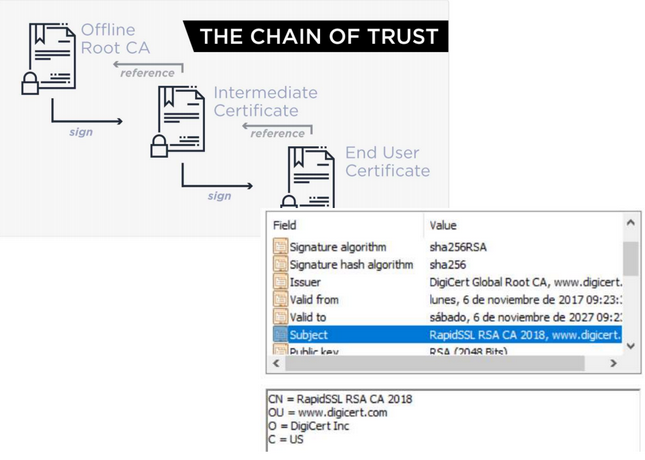
El cliente puede confiar o no en el certificado.

El certificado se compra a una autoridad certificante. Los certificados duran entre 1 y 2 años, luego hay que cambiarlo.

Hay un repositorio de root CAs (autoridades certificantes) en el cliente (el browser), que se consideran confiables.

El certificado puede estar vinculado a certificados intermedios o directo al root. Si hay intermedios, la cadena completa tiene que estar indicada en el certificado.

La cadena de confianza hace referencia a que el certificado puede estar firmado por autoridad de certificación intermedia (CA), puede que el issuer no confiemos pero si en la CA.

Un certificado es una clave pública. Para que sirva, en el servidor tiene que estar la clave privada también. Esta permite completar el handshake con un cliente.

Uno podría ponerse su propio generador de certificadso, pero el broswer no va a tener el root CA instalado y lo va a considerar no confiable.