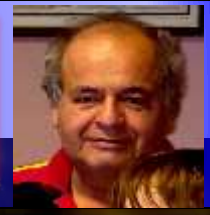
9 PROTOCOLO TCP SEGURO: TCP SSL



Manual de clases

Objetivo

- Describir cómo se puede utilizar la criptografía para mejorar los servicios de seguridad de TCP, incluyendo la confidencialidad, la integridad de los datos y la autenticación del punto terminal.

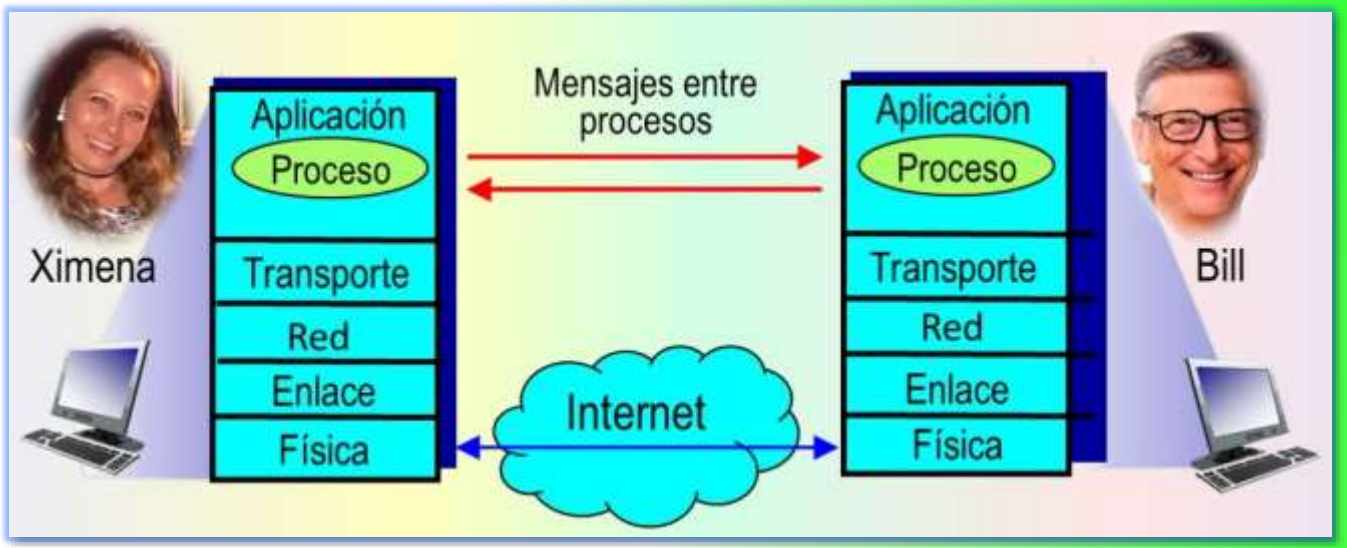
Última modificación:
7 de octubre de 2022

Tema 9 de:
SEGURIDAD EN REDES
Edison Coimbra G.

PROTOCOLO TCP SEGURO: TCP SSL

Concepto de procesos

(Kurose, 2017)

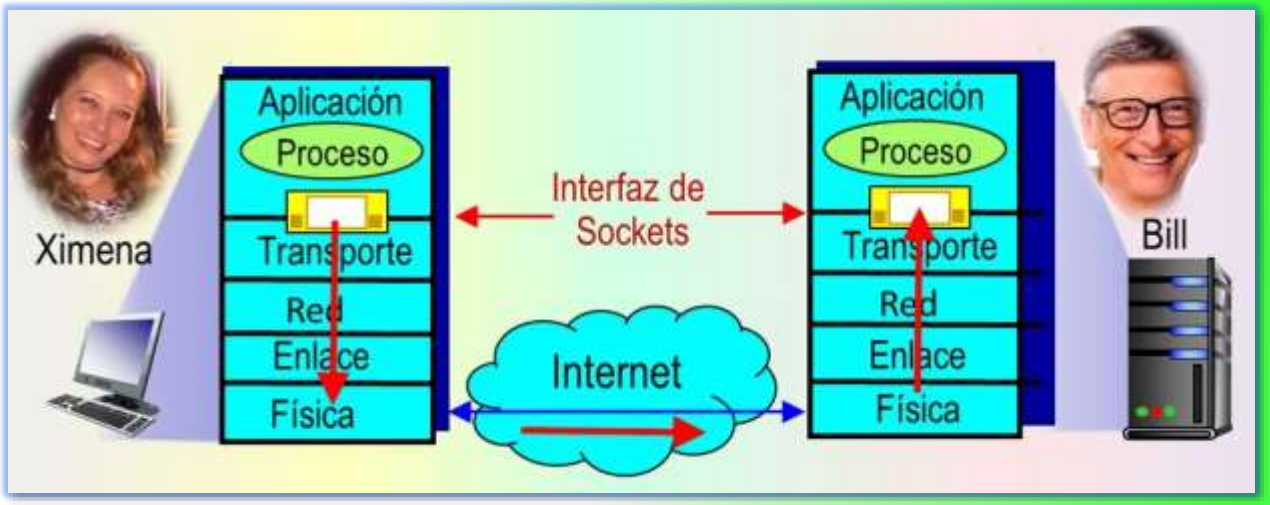


- **Antes de crear** una aplicación de red, se necesita disponer de conocimientos básicos sobre cómo se comunican entre sí los programas que se ejecutan en varios hosts. En la jerga de los sistemas operativos, los que se comunican no son programas, sino procesos.
- **Un proceso** puede interpretarse como un programa que se ejecuta dentro de un host.
  - ▶Si se ejecutan en el mismo host, se comunican entre sí mediante sistemas de comunicación interprocesos, aplicando reglas gobernadas por el sistema operativo del host.
  - ▶Si se ejecutan en host diferentes (con sistemas operativos potencialmente diferentes), se comunican entre sí intercambiando mensajes a través de la red.
- ▶**Un proceso emisor** crea y envía mensajes a la red.
- ▶**Un proceso receptor** recibe estos mensajes y responde devolviendo otros.
- **La figura ilustra** que los procesos que se comunican entre sí residen en la capa de aplicación de la pila de protocolos de cinco capas.

PROTOCOLO TCP SEGURO: TCP SSL

Interfaz entre el proceso y la red

(Kurose, 2017)

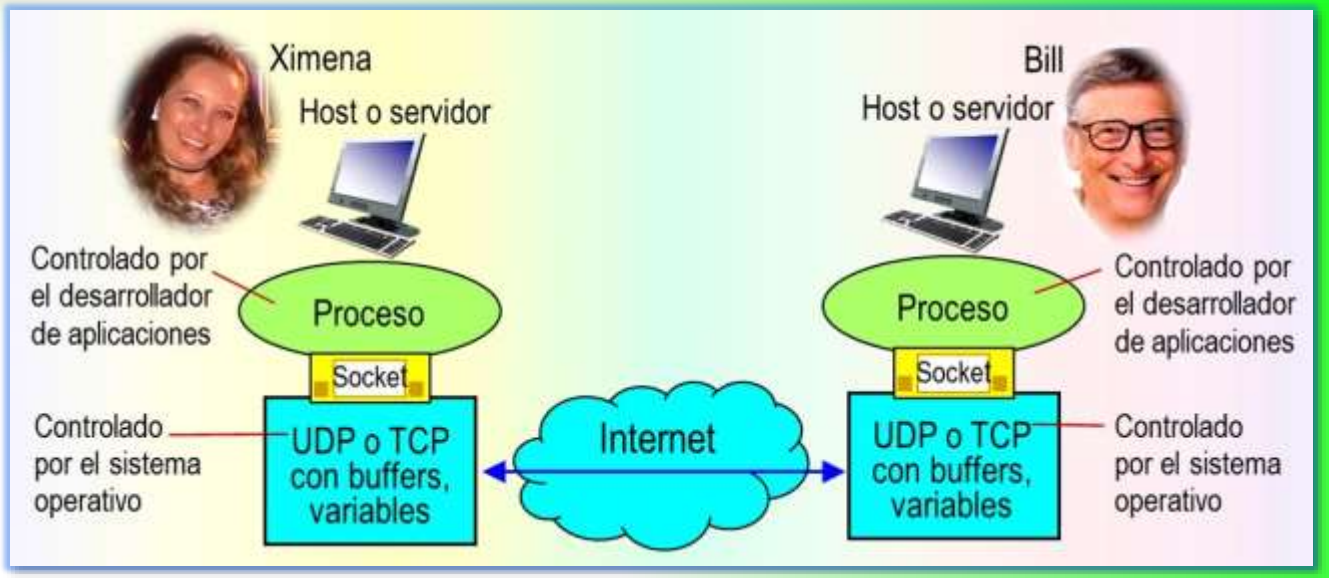


- **La mayoría de las aplicaciones** constan de parejas de procesos (cliente – servidor) que se comunican intercambiando mensajes.
- **Cualquier mensaje** enviado de un proceso al otro debe atravesar la red subyacente.
- **Un proceso** envía mensajes a la red y los recibe de la red a través una interfaz software denominada socket (enchufe).
- **¿Qué es un socket?** Una analogía para comprender el concepto de socket asociado a un proceso es la siguiente:
  - ▶Un proceso es análogo a una casa y un socket a la puerta de la casa. Cuando un proceso desea enviar un mensaje a otro que se ejecuta en otro host, envía el mensaje a través de su propia puerta (socket).
  - ▶El proceso emisor supone que existe una infraestructura de transporte al otro lado de la puerta que llevará el mensaje hasta la puerta del proceso de destino.
  - ▶Una vez que el mensaje llega al host de destino, éste pasa a través de la puerta (socket) del proceso receptor, actuando entonces el proceso receptor sobre el mensaje.

PROTOCOLO TCP SEGURO: TCP SSL

Comunicación entre sockets

(Kurose, 2017)



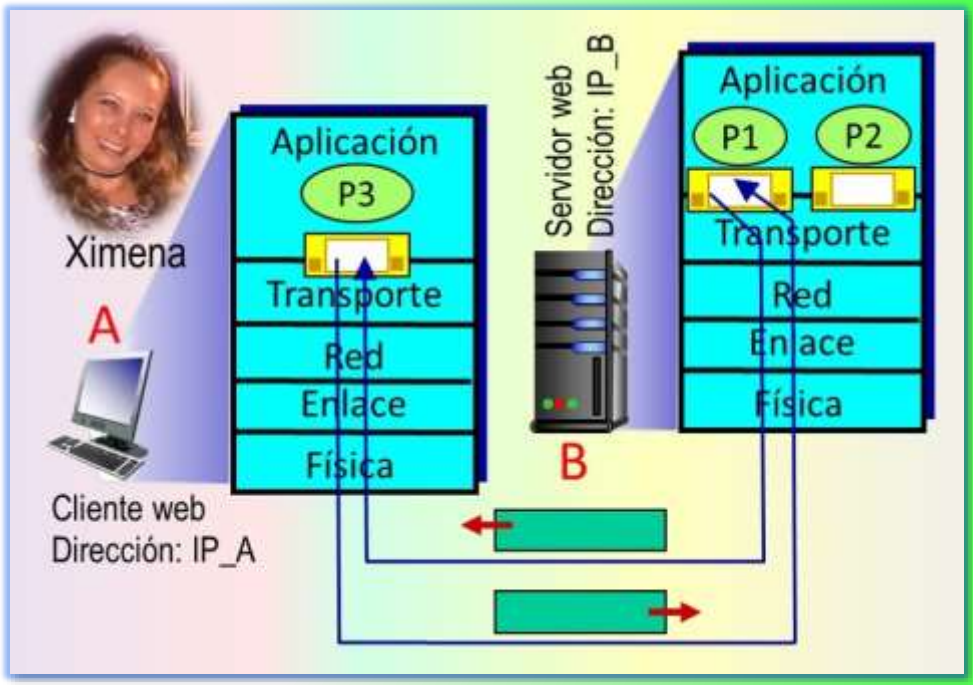
- **En la figura**, se ilustra la comunicación mediante sockets entre dos procesos que se comunican a través de Internet.
- **El protocolo de transporte** subyacente utilizado por los procesos puede ser UDP o TCP.
- **Un socket** es la interfaz entre la capa de aplicación y la de transporte de un host.
- **También se conoce** como Interfaz de programación de aplicaciones API entre la aplicación y la red, ya que el socket es la interfaz de programación con la que se construyen las aplicaciones de red.
  - ✉ Por ejemplo, un socket UDP se crea mediante la línea de código del programa Python:
    `clientsocket=socket(AF_INET,SOCKET_DGRAM)`
- **El desarrollador de la aplicación** tiene el control de todos los elementos del lado de la capa de aplicación del socket, pero no de los del lado de la capa de transporte, a excepción de:
  - ▶(1) La elección del protocolo de transporte que puede ser UDP o TCP y
  - ▶(2) Quizás la capacidad de fijar unos pocos parámetros, como los tamaños máximos del buffer y de segmento.
- **Una vez** que el desarrollador ha seleccionado un protocolo de transporte, la aplicación se construye utilizando los servicios de la capa de transporte proporcionado por dicho protocolo.

PROTOCOLO TCP SEGURO: TCP SSL

Direccionamiento de procesos

(Kurose, 2017)

- **Para enviar una carta** a un destino concreto, se necesita una dirección. De forma similar, para que un proceso que se está ejecutando en un host pueda enviar paquetes a otro proceso que se ejecuta en un host distinto, se necesita de una dirección del proceso receptor.
- **Para identificar** al proceso receptor se necesitan dos elementos de información:
  - ▶(1) La dirección IP del host.
  - ▶(2) Un identificador, llamado número de puerto, que especifique el proceso de recepción en el host de destino.
- **Por tanto,** además de conocer la dirección IP del host receptor, el host emisor también debe identificar al proceso receptor (o, para ser mas exactos, al socket receptor) que está ejecutándose en el host.
- **Esta información** es necesaria porque, en general, un host podría estar ejecutando varias aplicaciones de red a la vez.

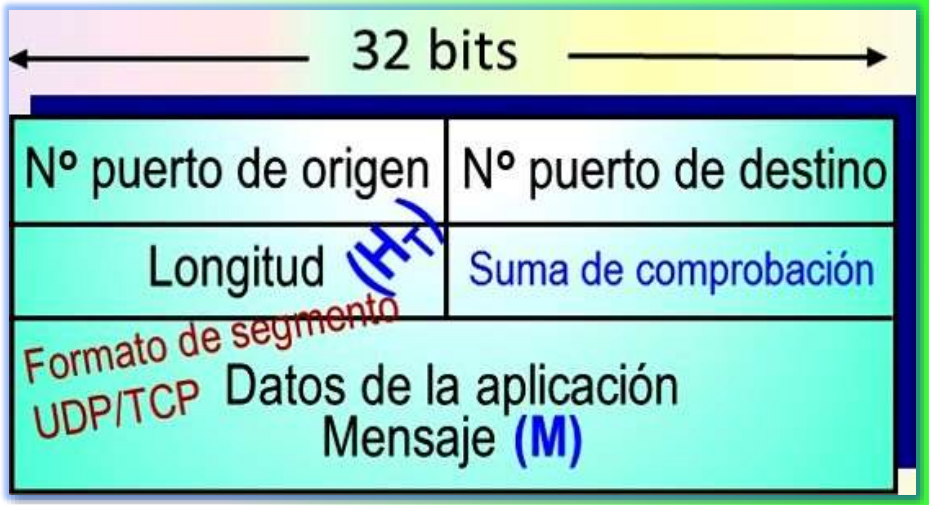


PROTOCOLO TCP SEGURO: TCP SSL

Números de puerto

(Kurose, 2017)

- **Cada segmento de transporte** tiene en su cabecera campos que identifican a un proceso, o más exactamente al socket al que tiene que entregarse el segmento ($H_T M$). Estos campos son el número de puerto de origen y el número de puerto de destino (ver el formato de segmento UDP/TCP de la figura).



- **Cada número de puerto** es un número en el rango de 0 a 65535 (16 bits).
  - ⊠Los números de puerto pertenecientes al rango de 0 a 1023 se conocen como números de puertos bien conocidos y son restringidos, lo que significa que están reservados para ser empleados por los protocolos de aplicación bien conocidos.
  - ⊠Por ejemplo HTTP utiliza el puerto 80, FTP utiliza el número de puerto 21, el proceso de servidor de correo SMTP se identifica mediante el puerto 25.
  - ⊠Al desarrollar una nueva aplicación, es necesario asignar un número de puerto a la aplicación.

PROTOCOLO TCP SEGURO: TCP SSL

Establecimiento de una conexión TCP

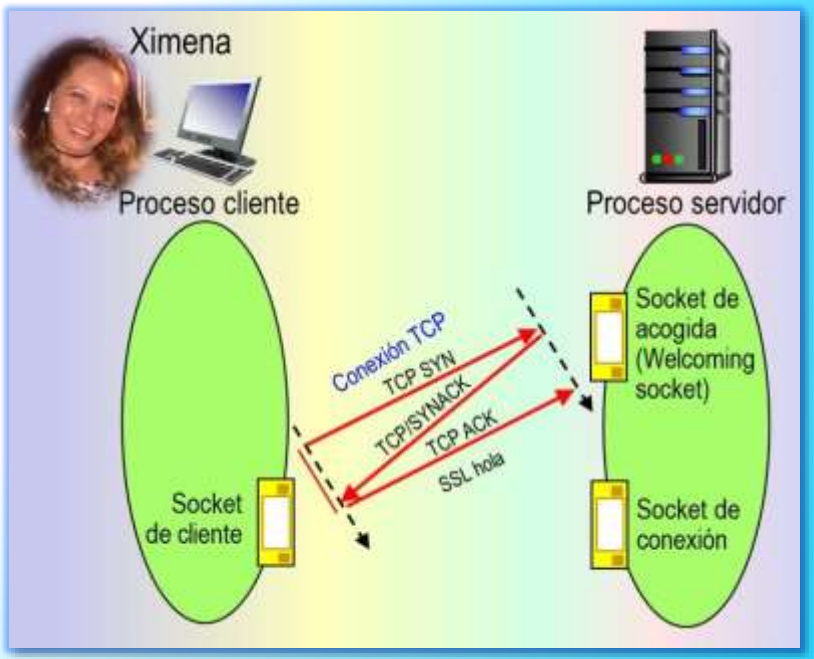
(Kurose, 2017)

- **Una sutil diferencia** entre un socket TCP y un socket UDP, es que en TCP se establecen conexiones identificadas por un conjunto de cuatro elementos: **(1)** dirección IP de origen, **(2)** número de puerto de origen, **(3)** dirección IP de destino, **(4)** número de puerto de destino. El establecimiento de la conexión tiene las siguientes fases.
- **Por tanto,** cuando un segmento TCP **(H_TM)** llega a un host procedente de la red, el host emplea los cuatro valores para dirigir (demultiplexar) el segmento al socket apropiado. En UDP, solo emplea dos, la dirección IP de destino y el número de puerto de destino.
- ▶**1. Proceso de conexión**. La aplicación del servidor TCP tiene un "socket de acogida", que está a la espera de solicitudes de conexión procedentes de clientes TCP en el puerto 12000. El cliente TCP crea un socket, que incluye su puerto origen, y envía un segmento de establecimiento de conexión con las líneas de código:

```
clientSocket=socket(AF_INET,SOCKET_DGRAM)
                    .clientSocket.connect((serverName,12000))
```

- ▶**2. Creación del socket de conexión**. El proceso servidor, que recibe la solicitud de conexión, con el puerto 12.000, crea un nuevo socket, el socket de conexión.

```
connectionlionSocket, addr=serverSocket.accept()
```

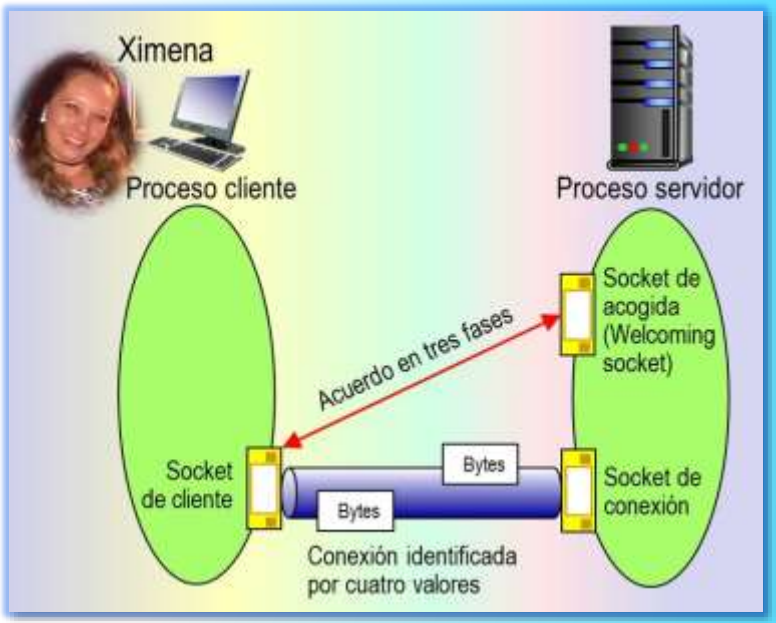


PROTOCOLO TCP SEGURO: TCP SSL

Establecimiento de una conexión TCP (cont.)

(Kurose, 2017)

- ▶**3.** Registro de identificadores de la conexión. A continuación, la capa de transporte en el servidor toma nota de los cuatro indicadores contenidos en el segmento de solicitud de conexión:
  - ▶ **(1)** La dirección IP del host de origen.
  - ▶ **(2)** El número de puerto de origen.
  - ▶ **(3)** La dirección IP del servidor de destino.
  - ▶ **(4)** El número de puerto de destino.
- ▶**4. Envío de datos**. El socket de conexión recién creado queda, entonces, identificado por cuatro valores. Así, todos los segmentos que lleguen y cuyo puerto de origen, dirección IP de origen, puerto de destino y dirección IP de destino se correspondan con estos cuatro valores serán enviados a este socket.
  - ✉Una vez establecida la conexión TCP, el cliente y el servidor podrán enviarse datos entre sí.

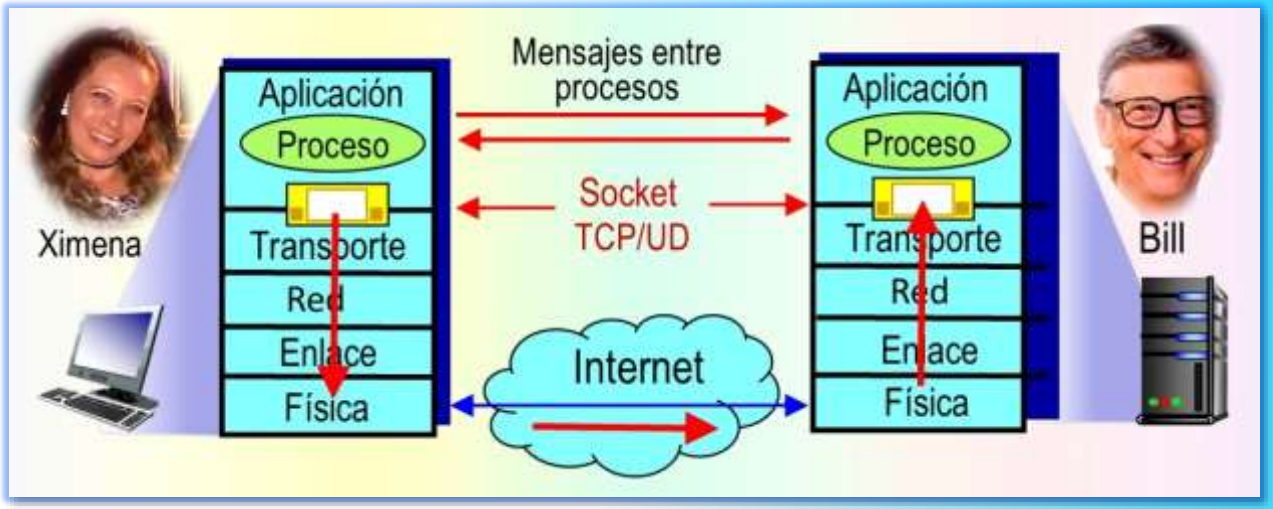


PROTOCOLO TCP SEGURO: TCP SSL

TCP seguro

(Kurose, 2017)

- **Ni TCP ni UDP** proporcionan ningún mecanismo de cifrado; los datos que el proceso emisor pasa al socket son los mismos datos que viajan a través de la red hasta el proceso de destino.
- **Por ejemplo**, si el proceso emisor envía una contraseña en texto legible (no cifrado) a su socket, esa contraseña viajará a través de los enlace entre el emisor y el receptor, pudiendo ser "husmeada" y descubierta en cualquiera de los enlaces intervinientes.



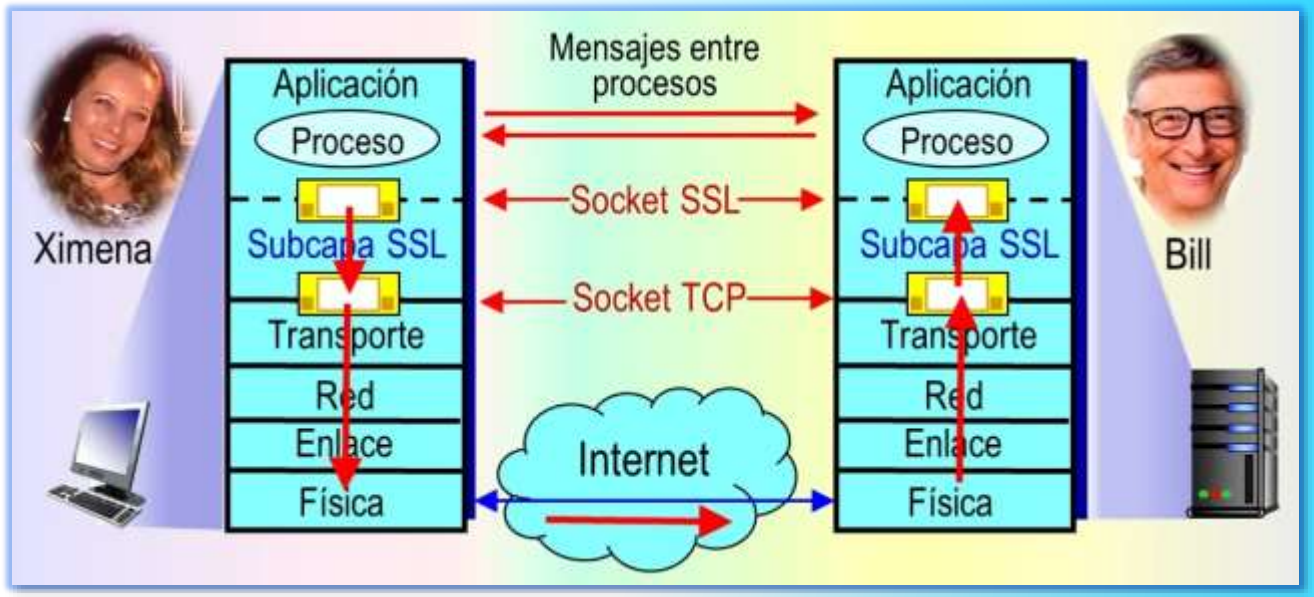
- **Puesto que la confidencialidad** y otras cuestiones de seguridad son críticas para muchas aplicaciones, la comunidad de Internet ha desarrollado una mejora para TCP, denominada SSL Capa de Sockets Seguros.

- **El TCP mejorado con SSL** no solo hace todo lo que hace el protocolo TCP tradicional, sino que también proporciona servicios críticos de seguridad proceso a proceso, entre los que se incluyen confidencialidad, integridad de los datos y autenticación del punto terminal.

PROTOCOLO TCP SEGURO: TCP SSL

El código SSL

(Kurose, 2017)



- **Aunque técnicamente** SSL reside en la capa de aplicación, desde la perspectiva del desarrollador, se trata de un protocolo de la capa de transporte.
- **Se debe destacar** que SSL no es un tercer protocolo de transporte de Internet, al mismo nivel que TCP o UDP, sino que es una mejora de TCP, que se implementa en la capa de aplicación.
- **En concreto**, si una aplicación desea utilizar los servicios de SSL, tiene que incluir **código SSL** (existen clases y librerías optimizadas) tanto en el lado del cliente como en el del servidor de la aplicación.
  - ▶**1.** SSL tiene su propia API, Interfaz de Programación de Aplicación de Socket, que es similar a la API de sockets del protocolo TCP tradicional. Cuando una aplicación utiliza SSL; el proceso emisor pasa los datos en texto legible al socket SSL.
  - ▶**2.** A continuación, la subcapa SSL cifra los datos en el host emisor y los pasa al socket TCP.
  - ▶**3.** Los datos cifrados viajan a través de Internet hacia el socket TCP del proceso receptor.
  - ▶**4.** El socket de recepción TCP pasa los datos cifrados a la subcapa SSL, que los descifra. Por último, la subcapa SSL pasa los datos en texto legible a través de su socket SSL al proceso receptor.

PROTOCOLO TCP SEGURO: TCP SSL

Origen del protocolo TCP mejorado

(Kurose, 2017)

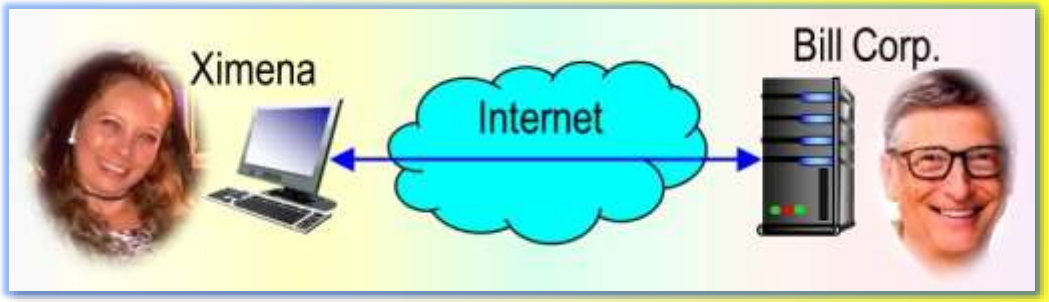
- **Las técnicas criptográficas** pueden proporcionar confidencialidad, integridad de los datos y autenticación del punto terminal a una **aplicación** específica. Ahora, se va a descender un nivel dentro de la pila de protocolos para examinar cómo se puede utilizar la criptografía para mejorar los servicios de seguridad de TCP, incluyendo la confidencialidad, la integridad de los datos y la autenticación del punto terminal.
- **Esta versión mejorada de TCP** se conoce comúnmente como **SSL**, Capa de Sockets Seguros. Una versión ligeramente modificada de SSL versión 3 denominada **TLS**, Seguridad de la Capa de Transporte, ha sido estandarizada por el IETF (RFC4346).
- **El protocolo SSL** fue diseñado originalmente por Netscape, pero las ideas básicas subyacentes para dotar de seguridad a TCP han terminado evolucionando a partir de los trabajos de Netscape. Desde su invención, **SSL** ha disfrutado de una amplia implantación; está soportado por todos los navegadores y servidores web más populares y es empleado por Gmail y la totalidad de los sitios de comercio electrónico de Internet (incluyendo Amazon, eBay y TaoBao).
- **Anualmente** se tranzan cientos de miles de millones de dólares a través de **SSL**. De hecho, si ha realizado en alguna ocasión una compra por Internet con su tarjeta de crédito, la comunicación entre su navegador y el servidor para dicha compra tuvo lugar, casi con total seguridad, sobre **SSL**. Puede verificar que su navegador está usando **SSL** viendo si el URL comienza por https: en lugar de por http:.

El protocolo TCP mejorado: TCP SSL

PROTOCOLO TCP SEGURO: TCP SSL

Necesidad de SSL

(Kurose, 2017)



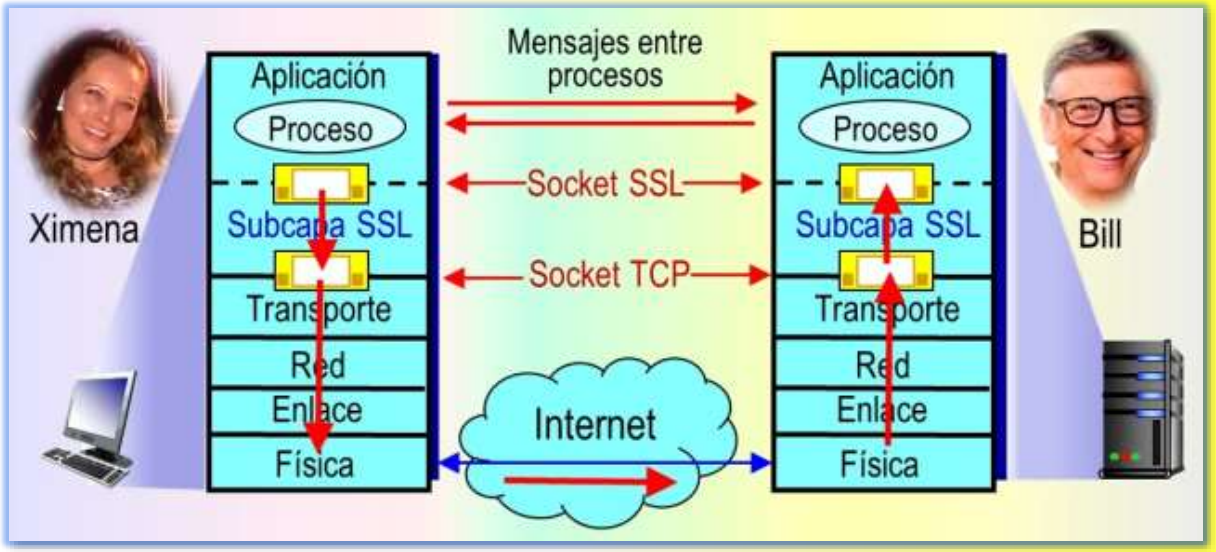
- **Para entender** la necesidad de SSL, analice un escenario de comercio electrónico por Internet. Imagine que Ximena está navegando por la Web y que accede al sitio web de la empresa Bill Corp. Que, entre otras cosas, se dedica a la venta de perfumes.
- **El sitio Bill Corp.** muestra un formulario en el que se supone que Ximena debe indicar el tipo de perfume y la cantidad deseada, junto con su dirección y el número de su tarjeta de crédito. Ximena introduce esta información, hace clic en el botón Enviar y espera recibir por correo ordinario los perfumes adquiridos; así como también el cargo correspondiente a ese pedido en el extracto de la tarjeta de crédito. Todo suena muy bien, pero si no se adopta ninguna medida de seguridad Ximena podría encontrarse con algunas sorpresas.
  - ▶Si no se utiliza ningún tipo de confidencialidad (cifrado), un intruso podría interceptar al pedido de Ximena y obtener la información de su tarjeta de crédito. El intruso podría entonces realizar compras a expensas de Ximena.
  - ▶Si no se utiliza ningún mecanismo que garantice la integridad de los datos, un intruso podría modificar el pedido de Ximena, haciéndole comprar diez veces más frascos de perfume de lo deseado.
  - ▶Si no se utiliza ningún tipo de autenticación del servidor, cualquier servidor podría mostrar el logotipo de Bill Corp. cuando en realidad se trataría de un sitio mantenido por un intruso, que se esté haciendo pasar por Bill Corp. Después de recibir el pedido de Ximena, el intruso podría tomar el dinero de Ximena y salir corriendo o podría efectuar un robo de identidad recopilando los datos relativos al nombre, la dirección y el número de tarjetas de crédito de Ximena.

El protocolo TCP mejorado: TCP SSL

PROTOCOLO TCP SEGURO: TCP SSL

Panorámica de SSL

(Kurose, 2017)



- **SSL solventa** los problemas mencionados. Se emplea para proporcionar seguridad a las transacciones que tienen lugar a través de HTTP. Sin embargo, puesto que SSL dota de seguridad a TCP, puede ser empleado por cualquier aplicación que se ejecute sobre TCP.
- **SSL proporciona** una Interfaz de programación de aplicaciones API simple con sockets, similar y análoga a la API de TCP. Cuando una aplicación desea utilizar SSL, la aplicación incluye clases/bibliotecas SSL.
  - ✉Aunque SSL reside técnicamente en la capa de aplicación, desde la perspectiva del desarrollador, se trata de un protocolo de transporte que proporciona servicios TCP mejorados con servicios de seguridad.
- **Se describirá** la versión simplificada de SSL, lo cual permitirá obtener una panorámica general del por qué y el cómo de SSL. La versión simplificada de SSL se compone de tres fases:
  - ▶**Fase 1**. Pasos del acuerdo
  - ▶**Fase 2**. Deducción de las claves
  - ▶**Fase 3**. Transferencia de datos
- **Se describirán** las tres fases de una sesión de comunicación entre la cliente Ximena y un servidor Bill Corp. El servidor Bill Corp. tiene una pareja de claves privada/publica y un certificado que asocia su identidad con su clave pública.

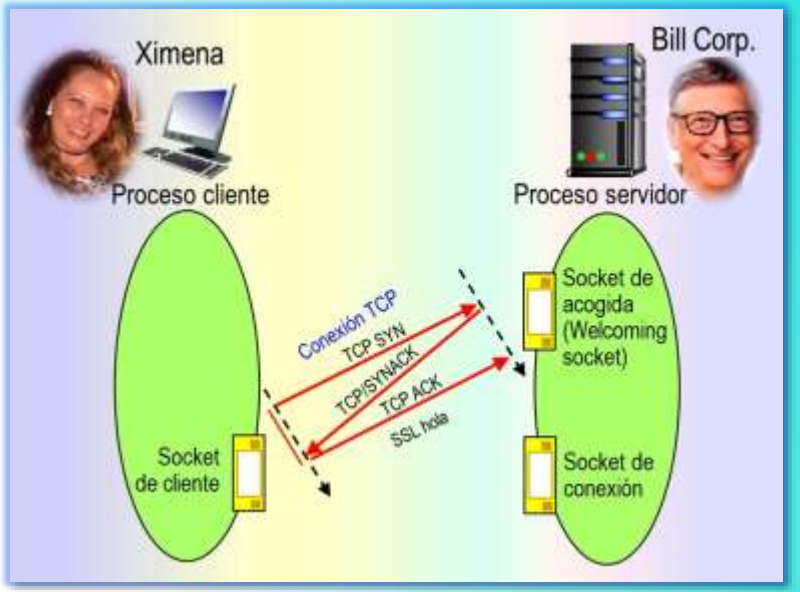
3.- FASES DEL PROTOCOLO SSL

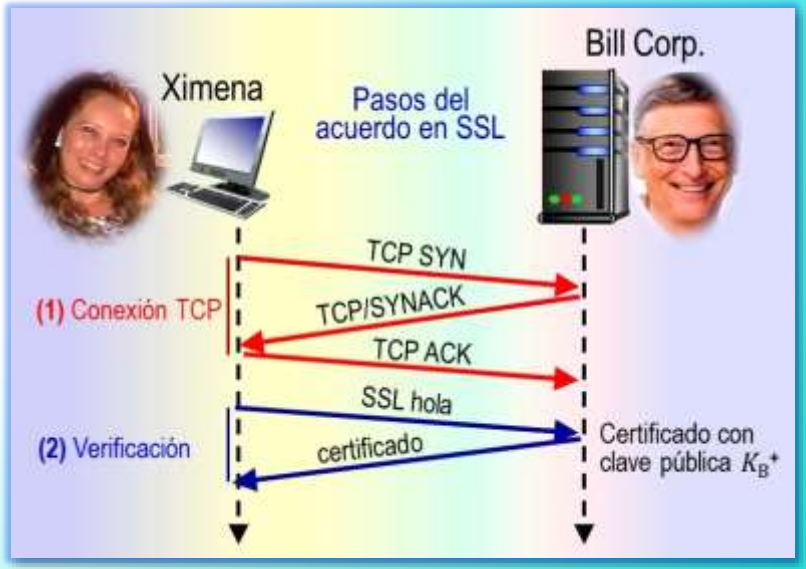
PROTOCOLO TCP SEGURO: TCP SSL

Fase 1. Pasos del acuerdo

(Kurose, 2017)

- ▶**(1)** Ximena establece una conexión TCP con Bill Corp.
  - Solicitud de establecimiento de conexión TCP SYN al socket de acogida.
  - Creación del socket de conexión y registro de identificadores de la conexión TCP/SYNACK.
  - Listo para el envío de datos TCP ACK, una vez que se ha establecido la conexión TCP.
- ▶**(2)** Ximena verifica que Bill Corp. es realmente Bill Corp.
  - Ximena envía a Bill Corp. un mensaje de saludo, SSL hola.
  - Bill Corp. responde con su certificado que contiene una clave publica.
  - Dado que el certificado ha sido emitido por una autoridad de certificación, Ximena puede estar segura de que la clave publica del certificado pertenece a Bill Corp.





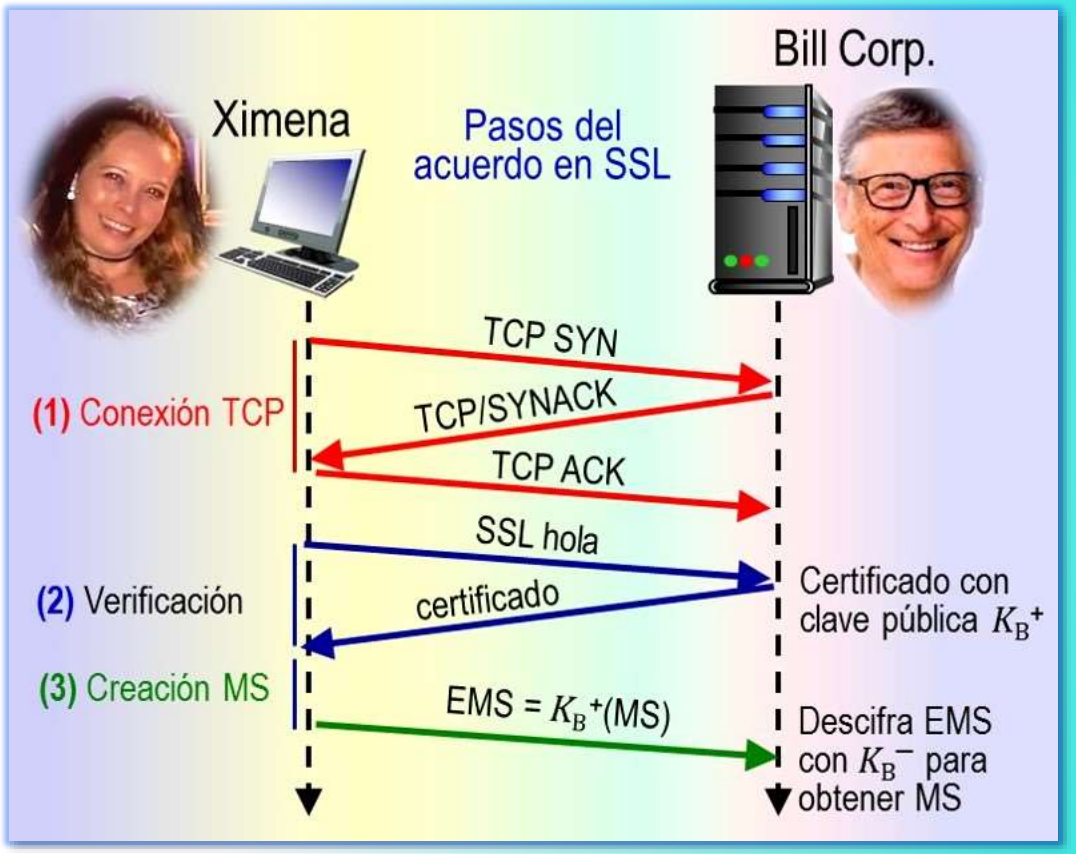
Fases del protocolo SSL

PROTOCOLO TCP SEGURO: TCP SSL

Fase 1. Pasos del acuerdo (cont.)

(Kurose, 2017)

- ▶**(3)** Ximena crea clave maestra MS y la envía cifrada a Bill Corp.
  - Ximena genera una clave maestra MS, la cual solo se utilizará para esta sesión SSL.
  - Ximena cifra la MS con la clave publica de Bill Corp. para crear la clave maestra cifrada EMS y se la envía a Bill Corp.
  - Bill Corp. descifra la clave maestra cifrada EMS con su clave privada para obtener la clave maestra MS.
- **Después de este tercer paso**, tanto Ximena como Bill Corp. (y nadie más) conocen la clave maestra MS para esta sesión SSL. Ambos emplearan esta clave maestra secreta MS para generar todas las claves simétricas que necesiten para la sesión SSL.



Fases del protocolo SSL

PROTOCOLO TCP SEGURO: TCP SSL

Fase 2. Deducción de las claves

(Kurose, 2017)

- **En principio**, la clave maestra MS, ahora compartida por Ximena y Bill Corp., podría emplearse como la clave de sesión simétrica para todos los cifrados y comprobaciones de integridad de los datos subsiguientes. Sin Embargo, generalmente, se considera más seguro que Ximena y Bill Corp. utilicen claves criptográficas distintas y también que empleen distintas para el cifrado y las comprobaciones de integridad.
- **Por tanto**, tanto Ximena y Bill Corp. utilizan la clave maestra MS para generar cuatro claves:
  - ▶E_X = clave de cifrado de sesión. Para cifrar los datos que Ximena envía a Bill Corp.
  - ▶E_B = clave de cifrado de sesión. Para cifrar los datos que Bill Corp. envía a Ximena.
  - ▶M_X = clave MAC de sesión (Código de Autenticación de Mensaje) para verificar la integridad de los datos que Ximena envía a Bill Corp.
  - ▶M_B = clave MAC de sesión (Código de Autenticación de Mensaje) para verificar la integridad de los datos que Bill Corp. envía a Ximena.
- **Tanto** Ximena como Bill Corp. generan las cuatro claves a partir de la clave maestra MS. Esto podría hacerse simplemente dividiendo la clave maestra en cuatro claves.
- **Al terminar** la fase de deducción de claves, tanto Ximena como Bill Corp. disponen de las cuatro claves.

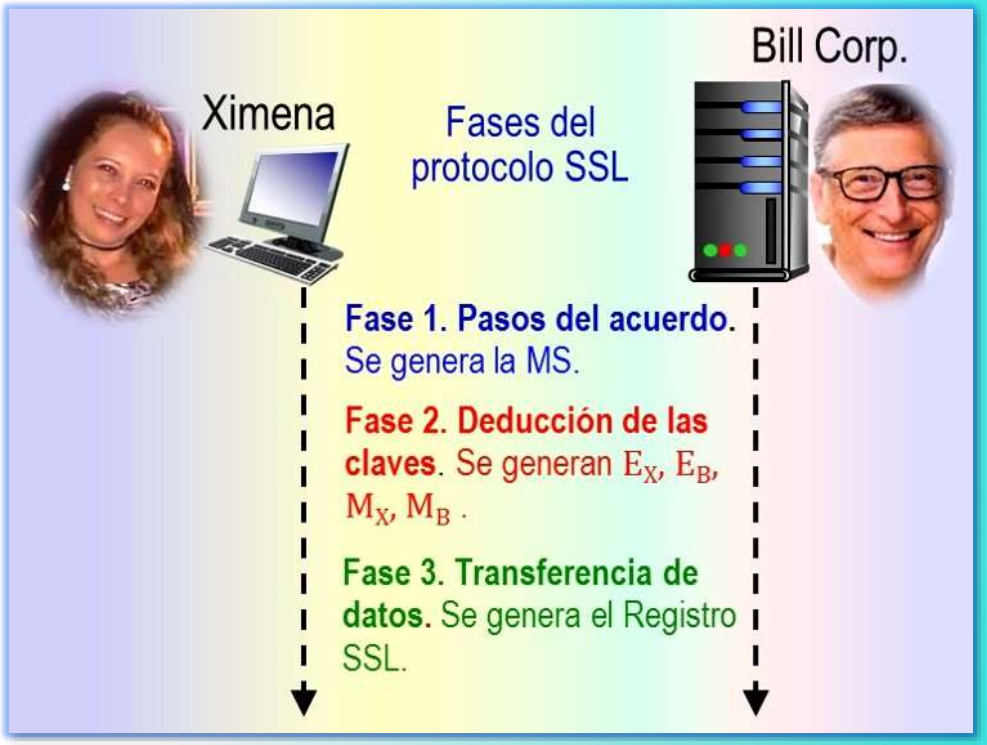
Fases del protocolo SSL

PROTOCOLO TCP SEGURO: TCP SSL

Fase 3. Transferencia de datos

(Kurose, 2017)

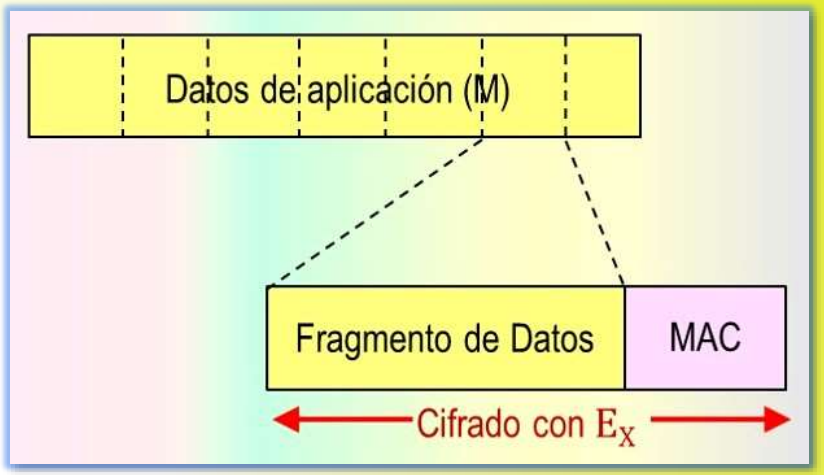
- **Ahora que Ximena y Bill Corp.** comparten las cuatro mismas claves de sesión (E_X, E_B, M_X, y M_B), pueden comenzar a enviar datos de forma segura a través de la conexión TCP.
- **Puesto que TCP** es un protocolo de flujos de bytes, una técnica natural seria que SSL cifrara los datos de aplicación sobre la marcha y luego pasara esos datos cifrados también sobre la marcha a TCP.
- **Pero si se hace eso**, ¿dónde se incluye el valor MAC necesario para comprobar la integridad de los datos? Realmente, no es deseable tener que esperar a que termine la sesión TCP para verificar la integridad de todos los datos que Ximena ha estado enviando a lo largo de la sesión completa. ¿Cómo resolver este problema?



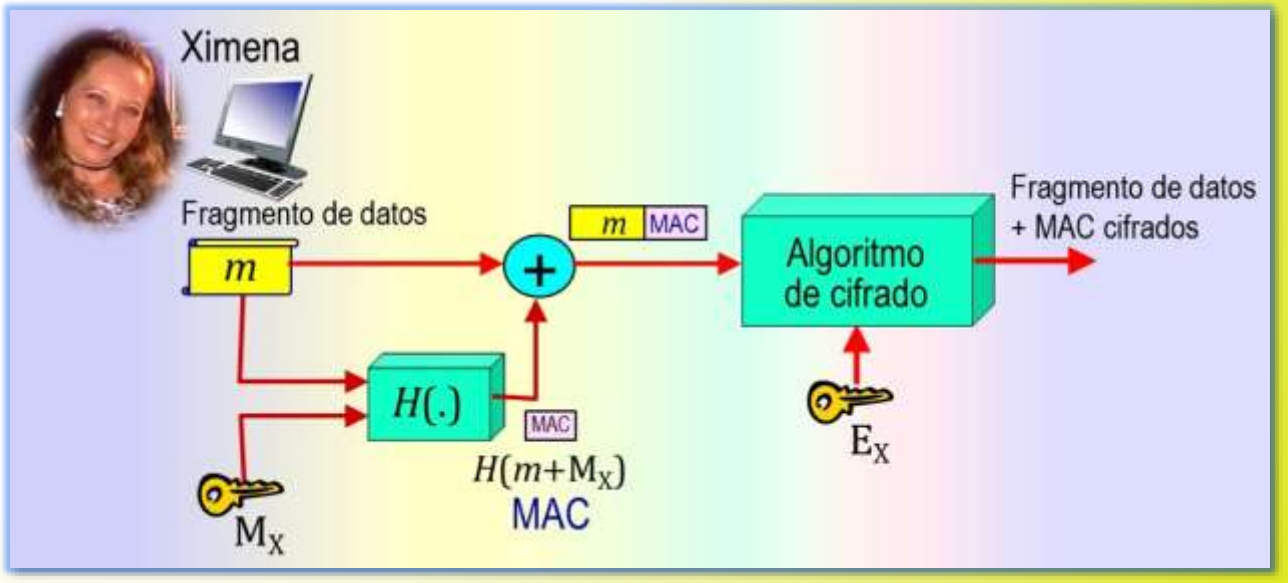
PROTOCOLO TCP SEGURO: TCP SSL

Fragmentación e inclusión del valor MAC

(Kurose, 2017)



- **Para incluir** el valor MAC necesario para comprobar la integridad de los datos, se procede a la fragmentación de los datos de aplicación.
  - ▶**1.** SSL divide el flujo de datos de aplicación en fragmentos de datos que conformaran registros SSL.
  - ▶**2.** Añade un código MAC a cada fragmento de datos para comprobar la integridad de los datos. Para crear el valor MAC, Ximena introduce los datos del fragmento (*m*) y la clave M_X en una función hash *H*(.).
  - ▶**3.** Luego cifra el fragmento de datos (*m*) junto con el código MAC. Para cifrar el paquete formado por el fragmento de datos y el valor MAC, Ximena utiliza su clave de cifrado de sesión E_X.



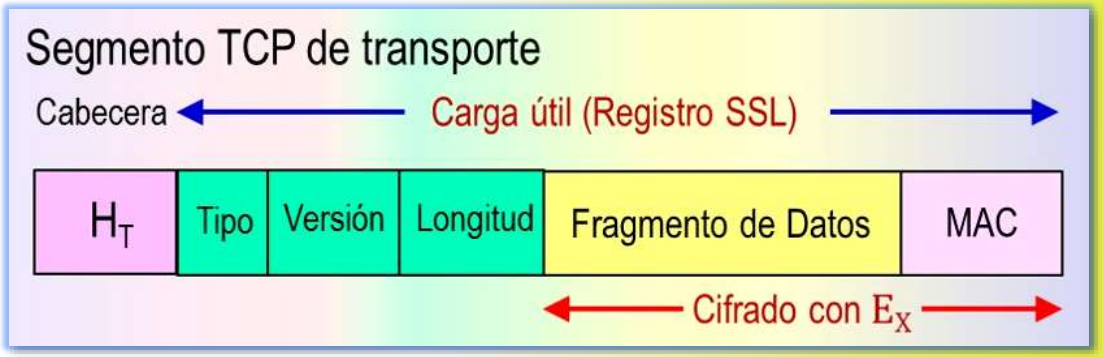
Fragmentación de los datos de aplicación

PROTOCOLO TCP SEGURO: TCP SSL

El número de secuencia SSL

(Kurose, 2017)

- **Aunque el método anterior** permite resolver bastantes de los problemas, sigue sin ser perfecto en lo que se refiere a proporcionar integridad de los datos para todo el flujo de mensajes. En particular, suponga que un atacante lleva a cabo un ataque por interposición y que tiene la capacidad de insertar, borrar y sustituir segmentos en el flujo de segmentos TCP enviados entre Ximena y Bill Corp.
- **El atacante**, por ejemplo, podría capturar dos segmentos enviados por Ximena, invertir el orden de los mismos, ajustar los números de secuencias TCP (que no están cifrados) y luego enviar los dos segmentos en orden inverso a Bill Corp. Se supone que cada segmento TCP encapsula exactamente un registro SSL.



- **La solución a este problema** consiste en utilizar números de secuencia. **SSL** hace esto de la forma siguiente:
  - ▶**1.** Ximena mantiene un contador de número de secuencia, que se inicializa en cero y que se incrementa cada vez que envía un registro SSL.
  - ▶**2.** Ximena no incluye realmente un número de secuencias en el propio registro, sino que cuando calcula el código MAC incluye el número de secuencia en el cálculo del código MAC .
  - ▶**3.** Ahora el valor MAC es un hash de: fragmento de datos + la clave MAC M_E + el número de secuencia actual.
  - ▶**4.** Bill Corp. controla los números de secuencias de Ximena, pudiendo verificar la identidad de los datos de un registro incluyendo el número de secuencia apropiado en el cálculo de MAC. Este uso de los números de secuencia SSL impide que el atacante lleve a cabo un ataque por interposición, tal como la reordenación o reproducción de segmentos. ¿Por qué ?

Referencias bibliográficas

PROTOCOLO TCP SEGURO: TCP SSL

Referencias bibliográficas

- CISCO (2015). *CCNA Routing and Switching. Introduction to Networks*. CISCO.
- CISCO (2016). *Introducción a las redes.* Madrid: Pearson Education, S.A.
- Forouzan, B. A. (2020). *Transmisión de datos y redes de comunicaciones.* Madrid: McGraw-Hill.
- Huawei Technologies (2020). *Basics of data communication networks*. Huawei.
- Kurose, J. Keith, R. (2017). *Redes de computadoras: un enfoque descendente*. Madrid: Pearson Education, S.A.

FIN

Tema 9 de:

SEGURIDAD EN REDES

Edison Coimbra G.