# Funciones hash

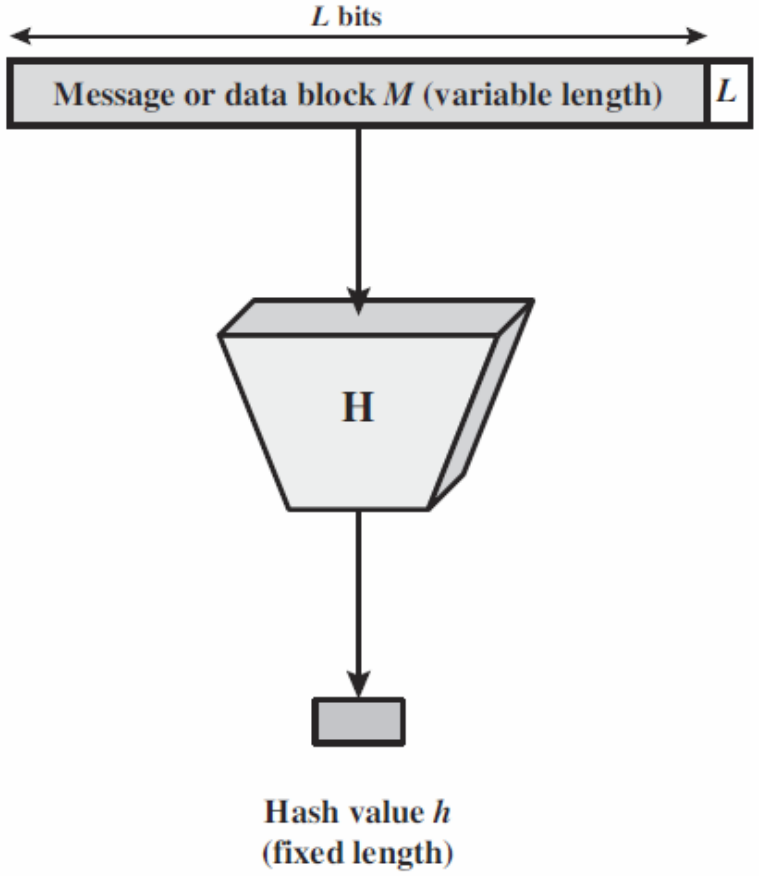
# Gestión de contraseñas

# Seguridad a nivel de transporte

# Principios de funciones hash

# Funciones Hash

- Una función hash (resumen) toma un bloque de entrada de longitud variable y produce un resumen de tamaño fijo
- Se encadenan de forma iterativa manteniendo el estado entre diversos bloques de entrada
- Suelen requerir un sistema de relleno (esquema Merkle-Damgard)



# Aplicaciones

- Autentificación de mensajes
  - La incorporación del resumen permite verificar la integridad del mensaje
- Firmas digitales
  - No es computacionalmente eficiente firmar el mensaje completo, se firma el resumen
- Otras
  - Gestión de contraseñas
  - Detección de intrusismo
  - Antivirus
  - PRNG
  - Etc.

# Requisitos

- Entrada de tamaño variable
  - H se puede aplicar a datos de cualquier tamaño
- Compresión
  - El tamaño de salida es fijo e independiente del de los datos de entrada
- Eficiencia
  - H(x) es relativamente eficiente de calcular tanto en software como en hardware
- Función de una vía
  - Para cada *h*, es computacionalmente difícil encontrar $m, H(m) = h$
- Colisión débil
  - Para un bloque x es computacionalmente difícil encontrar $y \neq x, H(y) = H(x)$
- Colisión fuerte
  - Es computacionalmente difícil encontrar una pareja $(x, y)$ tal que $H(x) = H(y)$
- Pseudoaleatoriedad
  - La salida de H cumple los tests de aleatoriedad

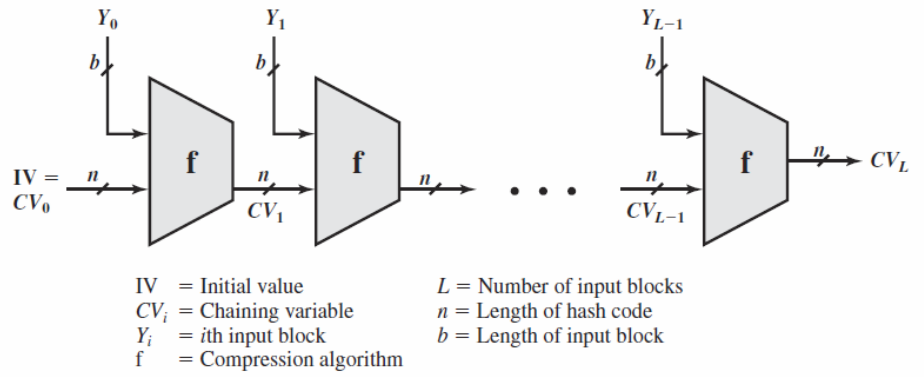
# Ataques

## Fuerza bruta

- Dependen únicamente de la longitud del resumen
- Preimagen
  - Es necesario hacer $2^{m-1}$ pruebas
- Resistencia a colisiones
  - Paradoja del cumpleaños $2^{m/2}$
  - 160 bits mínimo (80 bits)

## Criptoanálisis

- Se aplica a la estructura de la función hash
  - Merkle Damgard (esquema iterativo)



# Algunas funciones hash

- MD2 (128 bits, 1989)
  - Diseñado para procesadores de 8 bits
  - No recomendable por lentitud y seguridad
- MD4 (128 bits, 1990)
  - Base para la creación de otros hashes
  - Roto por Hans Dobbertin (RIPEMD-160)
- MD5 (128 bits, 1992)
  - Mejora sobre MD4
  - En 1996, Dobbertin encuentra colisiones
  - También se ha roto por preimagen (problemas en autoridades certificadoras)
- RIPEMD-160 (160 bits, 1996)
  - Creado por Dobbertin y otros
  - Basado en MD4
  - Considerado seguro en la actualidad

# Algunas funciones hash

- SHA-1 (160 bits, 1995)
  - Diseñado por NSA
  - Mejora sobre MD4 (ataques no desvelados)
  - Roto en 2011 por Marc Stevens
  - Es relativamente popular
- SHA-2 (2001)
  - Diseñado por NSA
  - Tamaños de 224, 256, 364 y 512 bits
  - Mejora sobre SHA-1
  - El de 512 y 384 funciona sobre 64 bits
  - Quasi ataques que hacen dudar de su seguridad en el futuro (SHA-3)

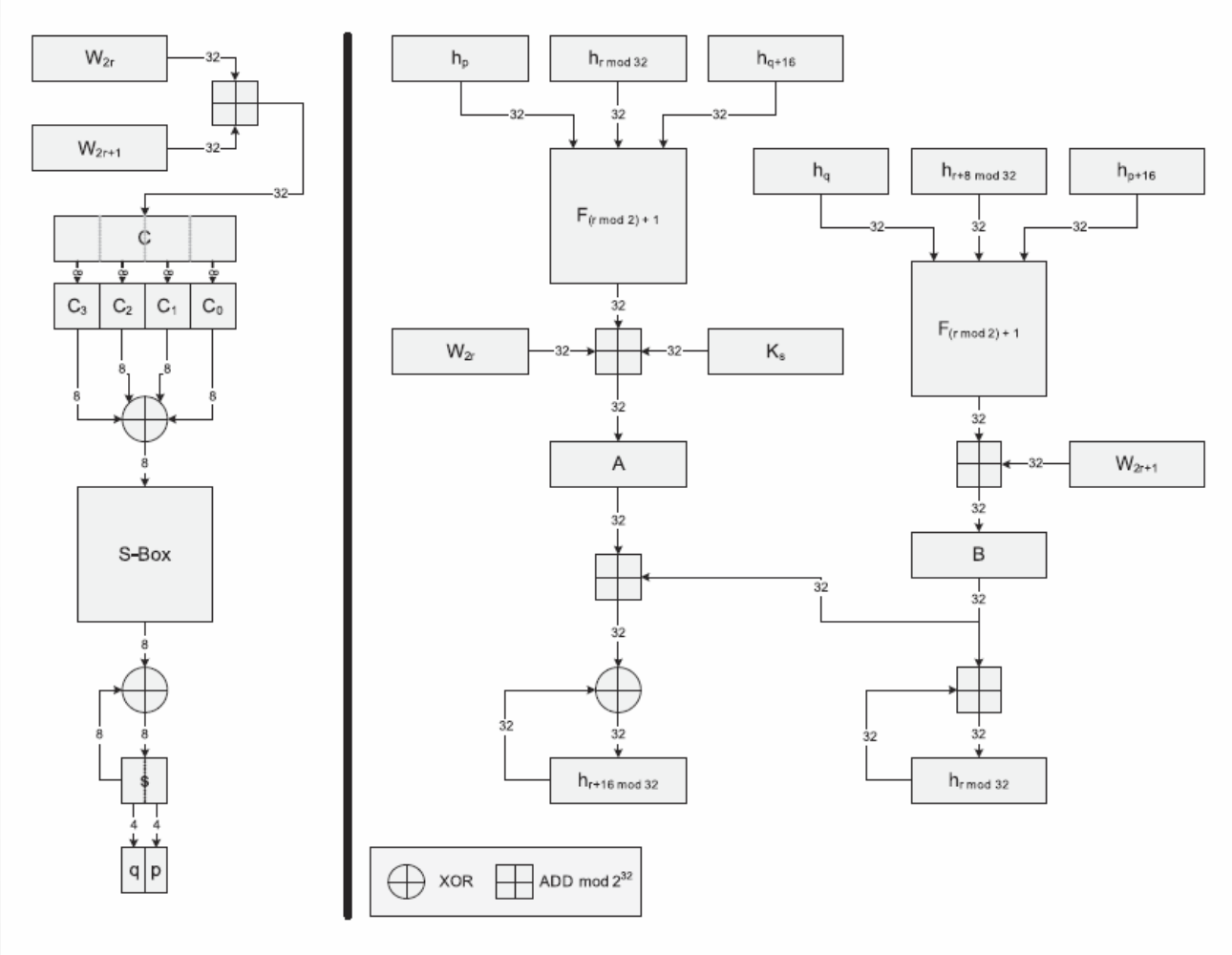
# SHA-3

- Anunciado en octubre 2012
- Originalmente, Keccak
- Diseñado por Bertoni, Daemen, Peeters, Van Assche
- Diseño completamente distinto a SHA-2 o SHA-1
- El objetivo es que sobreviva a los ataques contra SHA-2
- Usa un sistema de “esponja” con parámetros variables
- Se ha estandarizado con 224, 256, 384 y 512 bits de resumen*
- Velocidad de 12.5cpb
- No pretende sustituir a SHA-2 y coexisten ambos por el momento

# SHA-3

- Otros finalistas:
  - Blake
  - Grostl (Knudsen)
  - JH
  - Skein (Schneier)
- Tangle
  - Creada en la UA y CSI
  - Única función española
  - Colisiones en primera ronda
  - S-Box AES
  - Expansión por PRNG matricial
  - Función de ronda variable

# Tangle


W2r
W2r+1
32
C
C3
C2
C1
C0
8
S-Box
$
4
q
p
hp
hr mod 32
hq+16
F(r mod 2) + 1
hq
hr+8 mod 32
hp+16
W2r
Ks
A
B
W2r+1
hr+16 mod 32
hr mod 32
XOR
ADD mod 2^32

# Gestión de contraseñas

# Almacenamiento de contraseñas

- En claro
  - Ningún tipo de seguridad, cualquier acceso a la base de datos implica el robo de la identidad
- Cifrado
  - Todas las contraseñas se cifran con una única clave
  - La misma contraseña produce el mismo resultado cifrado
  - Posibilidad de precálculo
- Hashing
  - Similar al cifrado, pero no se puede descifrar
  - Ataque muy rápido basado en GPU
- Hashing + sal
  - Se concatena un valor aleatorio (sal) de tamaño significativo en bits para dificultar el precálculo
  - La sal se debe almacenar junto al resultado del hash para poder realizar la comprobación
- PBKDF + sal
  - Se sustituye la función hash por una de derivación de clave (Password Based Key Derivation Function), que realiza un gran número de iteraciones
  - Ralentiza en gran medida los ataques basados en GPU

# PBKDF2

- Función de derivación de clave (key stretching)
- Dentro del Public Key Cryptographic Standards (PKCS) de RSA / IETF RFC 2898 [2000]
- Sustituye a PKBDF1 que produce una salida fija de 160 bits
- Itera una función pseudoaleatoria (hash, cifrador, HMAC, etc.) repetidas veces
- Se utiliza una sal (mínimo 64 bits) para dificultar el precálculo

- Parámetros:
  - Función pseudoaleatoria
  - Contraseña
  - Sal
  - Iteraciones
  - Longitud
- Utilización:
  - WiFi (WPA y WPA2)
  - WinZip
  - Apple iOS y Mac OS X
  - Microsoft Windows
  - LUKS (Linux)
  - Etc.

# BCRYPT

- PBKDF diseñada por Provos y Mazières [USENIX, 1999]
- Basada en el cifrador blowfish
- Modifican el algoritmo de derivación de subclaves para hacerlo mucho más costoso computacionalmente.
- Utiliza algo de memoria y es más resistente que PBKDF2 frente a ataques basados en GPU/ASIC
- Parámetros:
  - Contraseña
  - Sal
  - Iteraciones (potencia de 2)

# SCRYPT

- PBKDF diseñada por Colin Percival para el sistema de backup “tarsnap”
- Se publicó en 2012 como Internet Draft (IETF) pero ha expirado
- Scrypt intenta evitar los ataques basados en GPU/ASIC mediante el uso de grandes cantidades de memoria y funciones secuenciales (no paralelizables)
- Se basa en un vector gigante de valores pseudoaleatorios que se acceden y combinan de forma desordenada, obligando a mantener dicho vector en RAM
- Se uso más popular consiste en la prueba de trabajo (proof of work) de las criptomonedas litecoin y derivadas (dogecoin, entre otras)
- Al contrario que bitcoin, que se basa en un doble sha256, las monedas basadas en scrypt no se pueden minar fácilmente con GPU/FPGA/ASIC

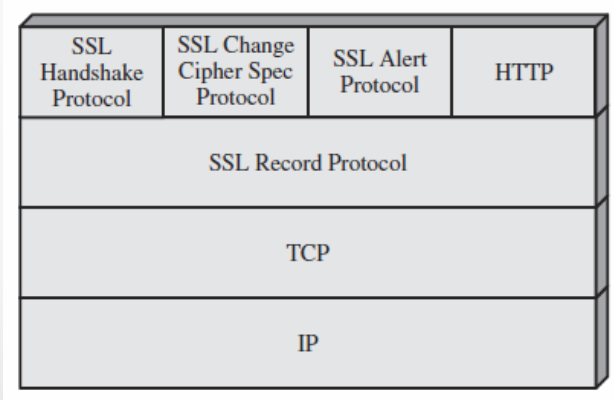
# El concurso PHC

- Concurso “semioficial” para algoritmos de tipo **password hashing**
- Se lleva a cabo por voluntarios durante 2014 – 2015
- Finalistas:
  - Catena
  - Lyra2
  - Makwa
  - Yescrypt
- Consultar password-hashing.net
- Ganador: ARGON2
  - Propuesto como nuevo estándar para gestión de contraseñas
  - Su popularidad es escasa de momento, siendo SCRYPT, BCRYPT o PBKDF más habituales
  - Han surgido algunos ataques de seguridad frente a ARGON2, aunque sin demasiado éxito

# Seguridad a Nivel de Transporte

# SSL

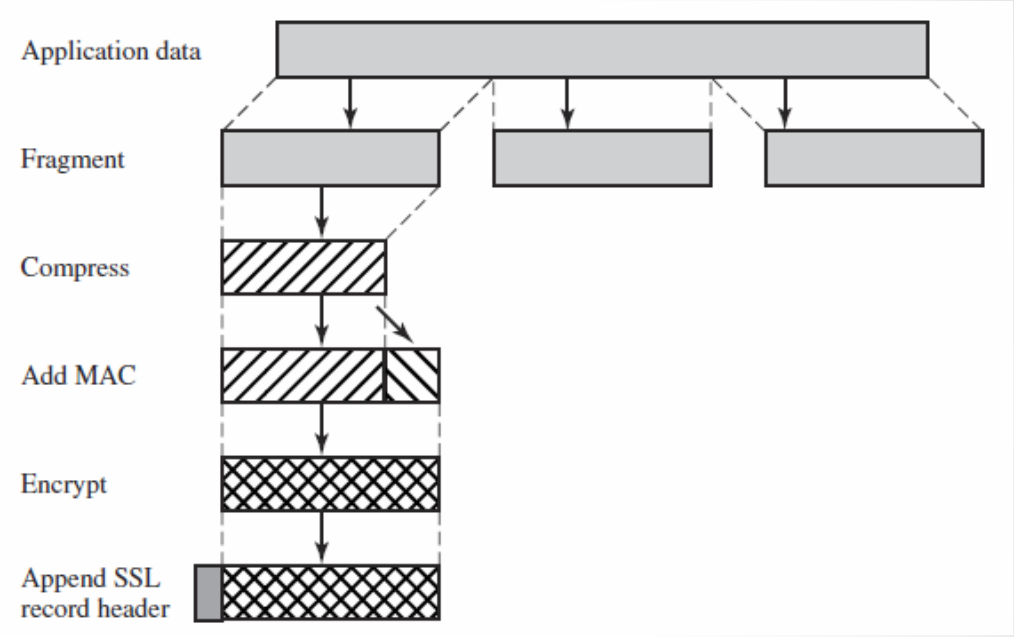
- SSL proviene de Netscape, la versión 3 se diseñó en abierto
- Está diseñado para proporcionar un servicio punto a punto seguro sobre TCP
- Son dos capas de protocolos y no un único protocolo




- Conexión:
  - Un transporte (OSI) que proporciona algún servicio
  - Es temporal y está asociada a una sesión
- Sesión:
  - Asociación entre un cliente y un servidor.
  - Creada por handshake
  - Define los parámetros de seguridad a utilizar por las múltiples conexiones
  - Evita tener que renegociar nuevos parámetros de seguridad para cada conexión

# SSL

- El protocolo de registro (record) ofrece:
  - Confidencialidad, cifrando los datos con criptografía simétrica y la clave negociada
  - Integridad, aplicando un MAC que autentifique el mensaje

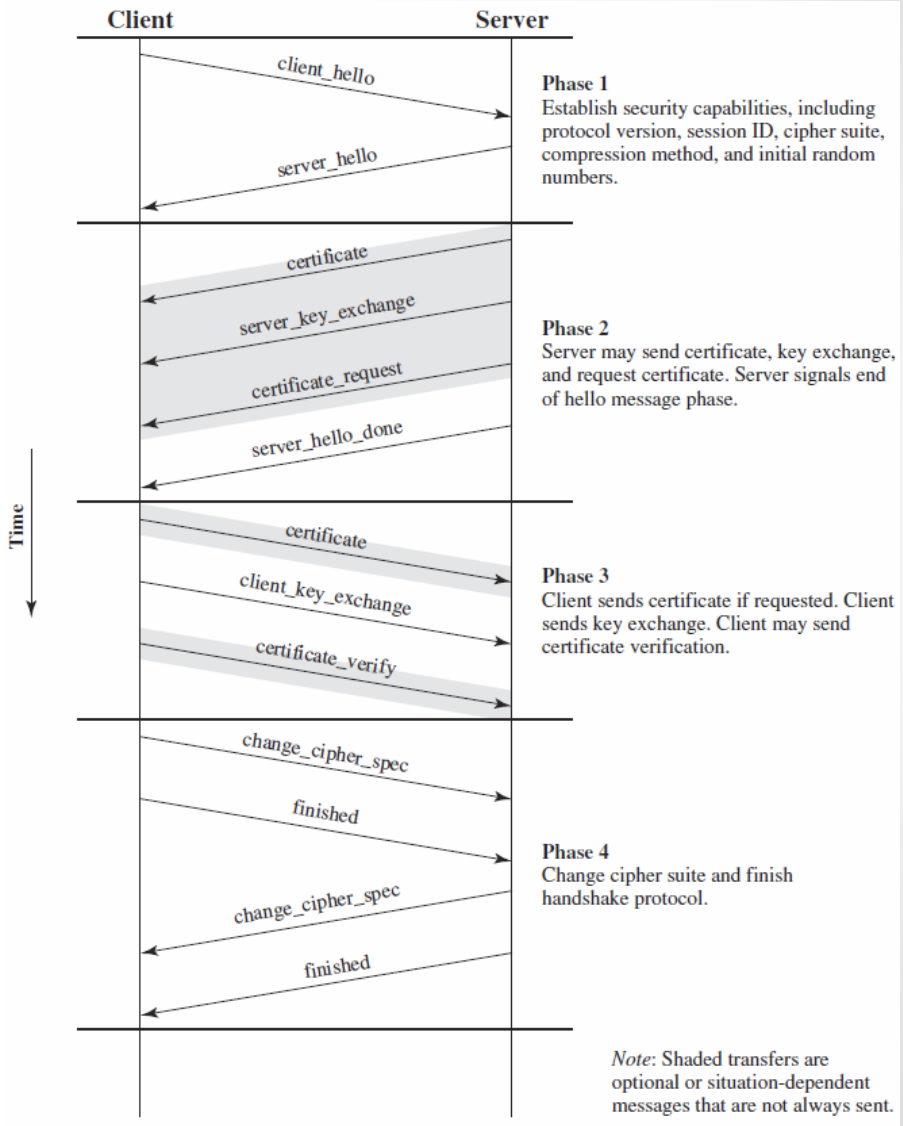


# SSL

- Cambio de cifrado (Change Cipher)
  - Un único mensaje de un byte con valor 1.
  - Actualiza la selección de criptosistema para esta conexión
- Alerta (Alert)
  - Permite enviar alertas
  - Cada mensaje consiste en dos bytes:
    - (1 aviso, 2 fatal)
    - Alerta específica

# SSL

- Saludo (Handshake)
  - Es el protocolo más complejo
  - Permite negociar algoritmos de cifrado y MAC y las claves a usar
  - Consiste en una serie de mensajes con la estructura:
    - Tipo (1 byte)
    - Longitud (3 bytes)
    - Contenido (>= 0 bytes)



# TLS

- TLS tiene como objetivo obtener una versión estándar de SSL.
- Existen varias diferencias
  - Mantiene el formato de registro pero el número de versión es 3.3
  - Usa HMAC (RFC 2104) y cubre los mismos campos más el de versión
  - Utiliza una función pseudoaleatoria (PRF) para expandir secretos en bloques de datos
  - Soporta todos los códigos de alerta menos el de “no_certificate”
  - Soporta los mismos cifradores excepto Fortezza (smart cards)

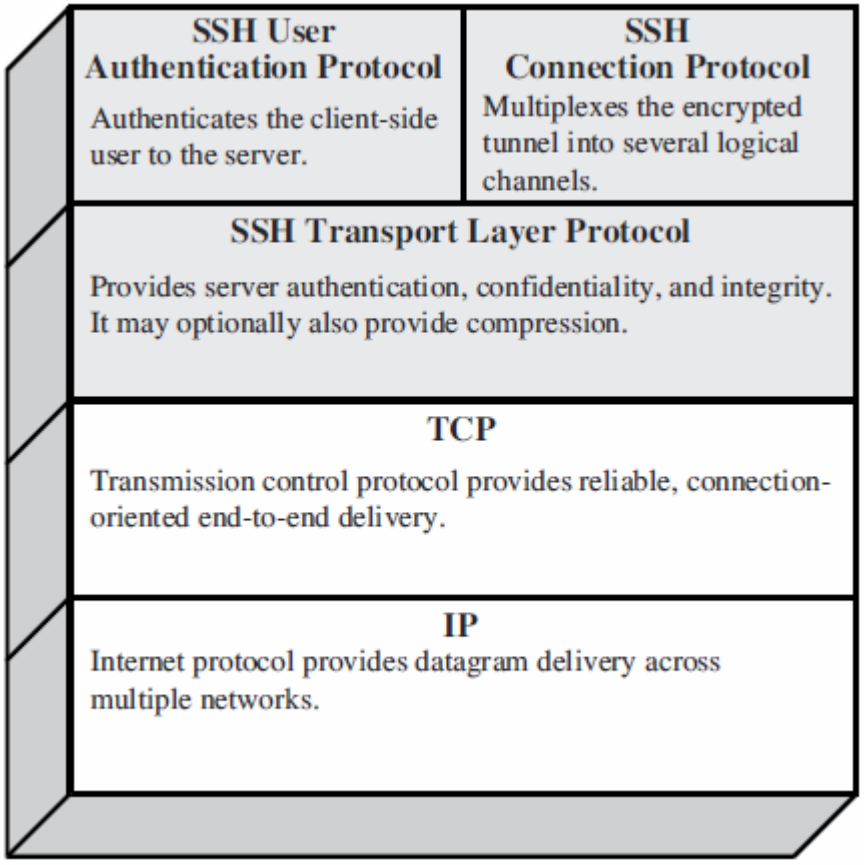
# HTTPS

- Significa HTTP sobre SSL
- Implementa una comunicación segura entre el navegador y el servidor web
- La principal diferencia para el usuario es que las URL empiezan por https://
- Con HTTPS se cifran:
  - URL
  - Contenido del documento
  - Contenidos de formularios
  - Cookies en ambas direcciones
  - Cabecera HTTP
- No hay diferencia entre SSL y TLS y ambos son HTTPS
- Interesante estudiar SPDY y HTTP2

# SSH

- Secure Shell (SSH) es un protocolo para asegurar las comunicaciones en red
- Originalmente diseñado para permitir login remoto seguro (sustituyendo TELNET, etc.)
- Se puede utilizar para otras conexiones más generales:
  - Transferencia de ficheros (SFTP)
  - Email
  - Port forwarding
  - Etc.
- Se organiza como tres protocolos por encima de TCP
  - Capa de transporte
    - Autentificación de servidor
    - Confidencialidad
    - Integridad
    - Compresión (opcional)
  - Autentificación
    - Autentifica a los usuarios frente al servidor
  - Conexión
    - Multiplexa múltiples canales lógicos sobre una única conexión SSH

# SSH


SSH User Authentication Protocol
Authenticates the client-side user to the server.
SSH Connection Protocol
Multiplexes the encrypted tunnel into several logical channels.
SSH Transport Layer Protocol
Provides server authentication, confidentiality, and integrity. It may optionally also provide compression.
TCP
Transmission control protocol provides reliable, connection-oriented end-to-end delivery.
IP
Internet protocol provides datagram delivery across multiple networks.

# Ampliación

## Otros materiales

- Se puede consultar los capítulos 13, 16 y 17.5 del libro de Lucena
  *(en los materiales de UACloud)*
- También se puede consultar los capítulos 9 y 13 de *"Handbook of Applied Cryptography"*
  *(más avanzado y en inglés)*
- Se puede investigar los candidatos a las competiciones PHC y SHA-3.

## Cuestiones

- ¿Qué ventajas aporta un esquema PBKDF frente a un sistema de contraseñas basado en hash convencional?
- También puede ser interesante estudiar la relación entre funciones hash/PBKDF y criptomonedas (Bitcoin, Litecoin, etc.).
- ¿Qué aplicaciones se benefician de protocolos como SSL/TLS, HTTPS o SSH?