UNIVERSIDAD CENTRAL DEL ECUADOR

COMPUTACIÓN

OCTAVO SEMESTRE

CRIPTOGRAFÍA Y SEGURIDAD DE LA INFORMACIÓN

TEMA: ACCESO WEB SEGURO (HTTPS)

Integrantes:

- Arias Basantes Joffre David
- Fiallos Checa Fátima Carolina
- Flores Armijo Byron Rigoberto
- Hurtado Tinoco Kevin David
- Lechon Lechon Cristian Alexander
- Pila Aguaisa Jordi Fernando
- Pujota Pineda Angelo Fabricio
- Tipán López Edgar Vinicio

26/05/2025

Contenido

1. Definición de HTTPS

HTTPS:

HTTPS, o HyperText Transfer Protocol Secure, es una extensión del Protocolo de Transferencia de Hipertexto (HTTP) que incorpora la Seguridad de la Capa de Transporte (TLS), anteriormente conocida como SSL. Según el documento estándar RFC 2818, publicado en mayo de 2000 por el IETF [13], HTTPS se define como HTTP sobre TLS, asegurando que las conexiones HTTP sean cifradas y autenticadas.

Es crucial para proteger datos sensibles, como contraseñas o detalles de tarjetas de crédito, especialmente en transacciones en línea. Por ejemplo, sitios web como bancos en línea destacan su importancia para evitar interceptaciones en redes públicas, como Wi-Fi [14]. El cifrado asegura que, incluso si los datos son interceptados, aparezcan como caracteres ilegibles, mejorando la seguridad del usuario.

2. Algoritmos criptográficos

2.1 Cifrado Simétrico

El cifrado simétrico es un método de encriptación en el que se utiliza una misma clave tanto para cifrar como para descifrar la información. Es uno de los sistemas más antiguos y eficientes para proteger datos, ya que suele ser más rápido que el cifrado asimétrico (de clave pública). [13]

2.1.1 AES (Advanced Encryption Standard)

AES (Advanced Encryption Standard) es un algoritmo de cifrado simétrico ampliamente utilizado para proteger datos electrónicos. Fue adoptado como estándar por el Instituto Nacional de Estándares y Tecnología (NIST) de EE.UU. en 2001, reemplazando al antiguo estándar DES (Data Encryption Standard), que se volvió vulnerable ante ataques modernos. [12]

1. **Funcionamiento**

   AES funciona mediante una serie de **rondas** de transformación sobre los datos. El número de rondas depende del tamaño de la clave

Cada ronda (excepto la última) consta de 4 operaciones:

- **SubBytes:** Sustitución de cada byte por otro según una tabla S-Box (sustitución no lineal).
- **ShiftRows:** Desplazamiento de filas de la matriz de estado (los datos) hacia la izquierda.
- **MixColumns:** Mezcla matemática de las columnas para mayor difusión (no se aplica en la última ronda).
- **AddRoundKey:** Se combina la matriz de estado con una subclave derivada de la clave original.

2. **Aplicaciones en HTTPS**

   AES se aplica para cifrar toda la comunicación entre el cliente (como un navegador) y el servidor web, asegurando que los datos como contraseñas, números de tarjetas o formularios no puedan ser leídos por terceros. Se usa en aplicaciones como banca en línea, tiendas virtuales, correo electrónico web, plataformas de redes sociales y servicios en la nube, donde la protección de la información es esencial.

2.1.2 ChaCha20

ChaCha20 es un algoritmo de cifrado simétrico de flujo, diseñado por Daniel J. Bernstein como una variante mejorada del algoritmo Salsa20. Es conocido por su alta velocidad, seguridad y eficiencia, especialmente en dispositivos móviles y sistemas con hardware limitado. [12]

A. Funcionamiento:

1. ChaCha20 toma como entrada:
   - Una clave de 256 bits
   - Un contador
   - Un vector de inicialización (nonce) de 96 bits

2. Genera un flujo de claves (key stream) mediante operaciones matemáticas seguras (suma, rotaciones y XOR).
3. Ese flujo se combina con los datos mediante XOR para cifrar o descifrar, bloque por bloque.
4. A diferencia de AES, ChaCha20 no necesita aceleración por hardware y ofrece un rendimiento uniforme en cualquier plataforma.

B. Aplicación en HTTPS

ChaCha20 se usa en HTTPS como parte de una suite de cifrado moderna como TLS_CHACHA20_POLY1305_SHA256.

- En este caso, ChaCha20 cifra los datos y Poly1305 proporciona autenticación (verifica que los datos no hayan sido alterados).
- Es especialmente útil en dispositivos móviles (Android, iOS) y servidores que no tienen soporte de aceleración AES, ya que ofrece rendimiento más consistente y seguro.

2.2 Cifrado Asimétrico

El cifrado asimétrico, o criptografía de clave pública, utiliza pares de claves (pública y privada) para garantizar confidencialidad, autenticación e integridad en las comunicaciones digitales. Su seguridad se basa en problemas matemáticos complejos, como la factorización de números primos grandes o el logaritmo discreto en curvas elípticas [16].

2.2.1 RSA (Rivest-Shamir-Adleman)

El algoritmo RSA, introducido en 1977 por Rivest, Shamir y Adleman [16], es un sistema de cifrado asimétrico basado en la dificultad computacional de factorizar números enteros grandes en sus componentes primos. En HTTPS, RSA se emplea para autenticar servidores mediante certificados digitales y para intercambiar claves simétricas durante el protocolo TLS Handshake [9].

- **Funcionamiento:**
  1. Generación de claves: Se eligen dos números primos grandes $p\ y\ q$, se calcula $n = p \times q\ y\ \phi(n) = (p-1)(q-1)$.Luego, se selecciona un exponente público e coprimo con $\phi(n)$, y se determina el exponente privado $d\ tal\ que\ e \times d \equiv 1\ mod\ \phi(n)$.
  2. Cifrado: Un mensaje $M\ se\ cifra\ como\ C = M^e\ mod\ n$.
  3. Descifrado: $M = C^d\ mod\ n$ [16].
- **Seguridad:** La fortaleza de RSA depende de la longitud de la clave. Se recomiendan claves de 2048 bits o superiores para resistir ataques de factorización modernos [16].

2.2.2 ECDSA (Elliptic Curve Digital Signature Algorithm)

El ECDSA, estandarizado por NIST en 1999 [18], es un esquema de firma digital basado en la criptografía de curva elíptica (ECC). Ofrece un nivel de seguridad equivalente a RSA con

claves significativamente más cortas (por ejemplo, una curva de 256 bits equivale a una clave RSA de 3072 bits), lo que lo hace eficiente para dispositivos con recursos limitados [19].

- **Funcionamiento**:
    1. **Generación de claves**: Se selecciona una curva elíptica y un punto base G. La clave privada es un entero d, y la clave pública es el punto $Q = d \times G$.
    2. **Firma**: Para firmar un mensaje m, se genera un número aleatorio k, se calcula el punto $(x, y) = k \times G$, y se derivan las firmas $r = x \bmod n$ y $s = k^{-1}(H(m) + dr) \bmod n$, donde $H(m)$ es el hash del mensaje.
    3. **Verificación**: El receptor valida la firma usando la clave pública Q [19].
- **Aplicaciones en HTTPS**: ECDSA se utiliza en certificados TLS/SSL (por ejemplo, certificados X.509) y en el intercambio de claves ECDHE para proporcionar secreto hacia adelante [9][19].

2.3 Funciones Hash

Las funciones hash criptográficas son algoritmos que transforman datos de entrada de cualquier tamaño en una salida de longitud fija, conocida como valor hash. Estas funciones son fundamentales en la seguridad informática, ya que permiten verificar la integridad de la información y autenticar su origen.

Una característica clave de las funciones hash es que incluso una mínima modificación en los datos de entrada produce un hash completamente diferente, lo que facilita la detección de alteraciones. Además, están diseñadas para ser unidireccionales, es decir, resulta computacionalmente inviable reconstruir la entrada original a partir del hash generado.

Entre las funciones hash más utilizadas se encuentran SHA-256 y SHA-384, que ofrecen altos niveles de seguridad y son ampliamente empleadas en aplicaciones como firmas digitales, almacenamiento seguro de contraseñas y tecnologías blockchain [20][21].

2.3.1 SHA-256

SHA-256 (Secure Hash Algorithm 256) es una función hash criptográfica desarrollada por el NIST como parte de la familia SHA-2. Este algoritmo transforma datos de entrada de longitud variable en una cadena única y de longitud fija de 256 bits (32 bytes).

SHA-256 es ampliamente utilizado en aplicaciones como blockchain, firmas digitales, almacenamiento seguro de contraseñas y protocolos de seguridad como TLS/SSL.

Su diseño garantiza que incluso un cambio mínimo en la entrada origine un hash completamente diferente. Además, es resistente a colisiones, ataques de preimagen y ataques de cumpleaños, lo que lo convierte en una opción robusta para validar la integridad y autenticidad de los datos [20].

2.3.2 SHA-384

SHA-384 también forma parte de la familia SHA-2, pero genera un resumen de mensaje más largo, de 384 bits (48 bytes). Esta mayor longitud proporciona un nivel de seguridad criptográfica superior, especialmente frente a ataques avanzados.

Aunque comparte la misma estructura interna que SHA-256, su salida más extensa lo hace más adecuado para entornos que demandan mayor robustez, tales como la generación de certificados digitales, la firma electrónica de documentos sensibles y aplicaciones gubernamentales o bancarias.

Su eficiencia y seguridad lo convierten en una opción recomendada cuando se requieren garantías criptográficas adicionales [21].

2.4 Intercambio de Claves

2.4.1 Diffie-Hellman (DH)

Este algoritmo es un criptosistema perteneciente a la rama de la criptografía asimétrica y su seguridad se basa en la dificultad para calcular logaritmos discretos.

2.4.1.1. Aplicación Crítica del Diffie-Hellman: Seguridad en Comunicaciones Digitales

Una de las aplicaciones más importantes del Intercambio de Claves Diffie-Hellman es en la seguridad de las comunicaciones digitales, donde se utiliza para establecer una clave simétrica segura. Esto es esencial para protocolos como HTTPS, SSH, y VPNs, donde la privacidad y la integridad de los datos son **primordiales.**

2.4.2 ECDH (Elliptic Curve Diffie-Hellman)

El protocolo Elliptic Curve Diffie-Hellman (ECDH) es una técnica criptográfica avanzada que permite a dos partes, establecer una clave secreta compartida a través de un canal inseguro,

aprovechando las propiedades de la criptografía de curva elíptica. Introducido como una variante del protocolo Diffie-Hellman original de 1976, el ECDH mejora la seguridad y la eficiencia al usar curvas elípticas, permitiendo tamaños de clave más pequeños para el mismo nivel de seguridad, lo que lo hace ideal para dispositivos con recursos limitados.

2.4.2.1. Variantes y Aplicaciones Prácticas

El ECDH tiene dos variantes principales, cada una con casos de uso diferentes:

- **ECDH Estático**: Utiliza pares de claves pública-privada a largo plazo, a menudo certificados por una autoridad de confianza. Es adecuado para escenarios donde el secreto hacia adelante no es prioritario, pero carece de resistencia a la compromisión de claves, lo que puede exponer comunicaciones pasadas si una clave es comprometida.
- **ECDH Efímero (ECDHE)**: Emplea claves temporales por sesión, proporcionando secreto hacia adelante. Es común en protocolos como TLS 1.2 y 1.3, donde las claves de sesión se renegocian frecuentemente, y requiere autenticación adicional (por ejemplo, vía certificados) para prevenir ataques de man-in-the-middle.

3. Protocolos criptográficos

3.1 TLS (Transport Layer Security)

El protocolo Transport Layer Security (TLS) es un marco criptográfico crítico diseñado para proteger las comunicaciones a través de Internet. Funciona principalmente a través de dos subprotocolos: el protocolo TLS Handshake, que establece una sesión segura, y el protocolo de registro TLS, que garantiza la confidencialidad e integridad de los datos durante la transmisión. El TLS ha evolucionado considerablemente, con la última versión, la TLS 1.3, que mejora la seguridad y el rendimiento, a la vez que aborda las vulnerabilidades presentes en las versiones anteriores [1].

El protocolo TLS es parte integral del funcionamiento de HTTPS, ya que proporciona cifrado y autenticación para comunicaciones web seguras. Cuando un usuario accede a un sitio web HTTPS, el protocolo TLS inicia un proceso de enlace que establece una conexión segura y garantiza que los datos transmitidos entre el cliente y el servidor permanezcan confidenciales y a prueba de manipulaciones. Se detallan los componentes clave del uso de TLS en HTTPS.

Proceso de apretón de manos TLS

- El cliente envía un mensaje de «ClientHello» al servidor, indicando las versiones de TLS y los conjuntos de cifrado compatibles.
- El servidor responde con un «ServerHello» y selecciona la versión de TLS y el conjunto de cifrado para la sesión.
- El servidor envía su certificado digital, que incluye su clave pública, al cliente para su autenticación.
- El cliente verifica el certificado comparándolo con autoridades de certificación (CA) confiables y genera una clave de sesión para el cifrado [2].

Administración de certificados

- HTTPS se basa en un sólido ecosistema de certificados, con un número significativo de certificados emitidos por varias CA, incluida Let's Encrypt.
- Las auditorías periódicas de la validez de los certificados son cruciales, ya que los estudios muestran que un pequeño porcentaje de dominios utilizan certificados caducados o auto firmados [2].

Implicaciones de seguridad

- A pesar de la adopción generalizada del TLS, las vulnerabilidades persisten y requieren actualizaciones y evaluaciones de seguridad continuas para mitigar los riesgos asociados a los protocolos anticuados [2].

3.2 X.509

El protocolo X.509 es un componente clave de la infraestructura de clave pública (PKI) que facilita las comunicaciones seguras a través de Internet. Define el formato de los certificados de clave pública, que vinculan la identidad de una entidad con su clave pública, lo que permite realizar transacciones y autenticarse de forma segura. El protocolo se utiliza ampliamente en varios protocolos de seguridad, incluido el TLS, en el que los navegadores web validan los certificados X.509 para garantizar la autenticidad de los servidores web [3].

El protocolo X.509 es parte integral de HTTPS y proporciona un marco para una comunicación segura a través de certificados digitales. Estos certificados autentican la identidad de los sitios web y garantizan que los usuarios se conecten a servidores legítimos. El proceso incluye varios pasos clave, que se describen a continuación.

Emisión y estructura del certificado

- Los certificados X.509 son emitidos por las autoridades de certificación (CA) y vinculan una clave pública a la identidad de una entidad, e incluyen detalles como la longitud de la clave y el algoritmo utilizado.
- Cada certificado tiene una fecha de caducidad, lo que garantiza que no se utilicen certificados obsoletos, lo cual es crucial para mantener la seguridad [4].

Proceso de validación

- Durante una conexión HTTPS, el cliente valida el certificado del servidor comparándolo con una lista de CA confiables. Esto implica comprobar la firma, el período de validez y el estado de revocación del certificado.
- Sin embargo, los estudios muestran que un porcentaje significativo de certificados (el 21,5%) no son válidos desde el punto de vista sintáctico, lo que genera posibles vulnerabilidades de seguridad [4].

Desafíos y alternativas

- Los sistemas tradicionales basados en CA se enfrentan a desafíos como la ineficiencia de los métodos de revocación, como el OCSP y las CRL, que pueden provocar demoras y problemas de privacidad.
- Los enfoques alternativos, como el uso de DNS en lugar de HTTPS (DoH) para la autenticación de claves públicas, tienen como objetivo reducir la dependencia de las CA y mejorar la seguridad [4].

3.3 OCSP (Online Certificate Status Protocol)

El Online Certificate Status Protocol (OCSP) es un protocolo utilizado para verificar en tiempo real el estado de revocación de certificados digitales X.509 en una infraestructura de clave pública (PKI). OCSP permite consultas en línea para determinar si un certificado es válido, ha sido revocado o su estado es desconocido [10].

Funcionamiento de OCSP

- **Solicitud del Cliente:** El cliente envía una solicitud OCSP al servidor, incluyendo el número de serie del certificado que se desea verificar [11].
- **Respuesta del Servidor OCSP:** El servidor responde con el estado del certificado, puede ser: Good, Revoked y Unknown [11].

- **Verificación de la Respuesta:** El cliente verifica la firma digital de la respuesta OCSP para asegurarse de su autenticidad y que proviene de una fuente confiable [11].

Importancia de OCSP en la seguridad de la información.

- **Verificación en tiempo real:** Proporciona información actualizada sobre el estado de los certificados [12].
- **Reducción de sobrecarga**: Al evitar la descarga completa de listas CRL, OCSP reduce la carga de red y mejora la eficiencia [12].
- **Mejora la experiencia del usuario:** Ofrecer respuestas rápidas sobre el estado de los certificados [12].

Desafíos y avances en la implementación de OCSP

- **Carga en el Servidor:** Las solicitudes en tiempo real pueden generar una carga significativa en los servidores OCSP [12].
- **Privacidad del Usuario:** Las consultas OCSP pueden revelar información sobre los sitios web que un usuario visita [12].
- **Disponibilidad:** Si el servidor OCSP no está disponible, puede afectar la capacidad de verificar certificados [12].

3.4 PKI (Public Key Infrastructure)

La Infraestructura de Clave Pública (PKI), por su parte, es un marco que facilita la seguridad en comunicaciones digitales mediante criptografía de clave pública. Según un artículo académico titulado "Public Key Infrastructure: A Survey" de Albarqi et al. (2015) [15], publicado en el Journal of Information Security, PKI se define como un sistema desarrollado para soportar criptografía asimétrica, donde un mensaje se cifra con la clave pública del receptor y solo se puede descifrar con su clave privada.

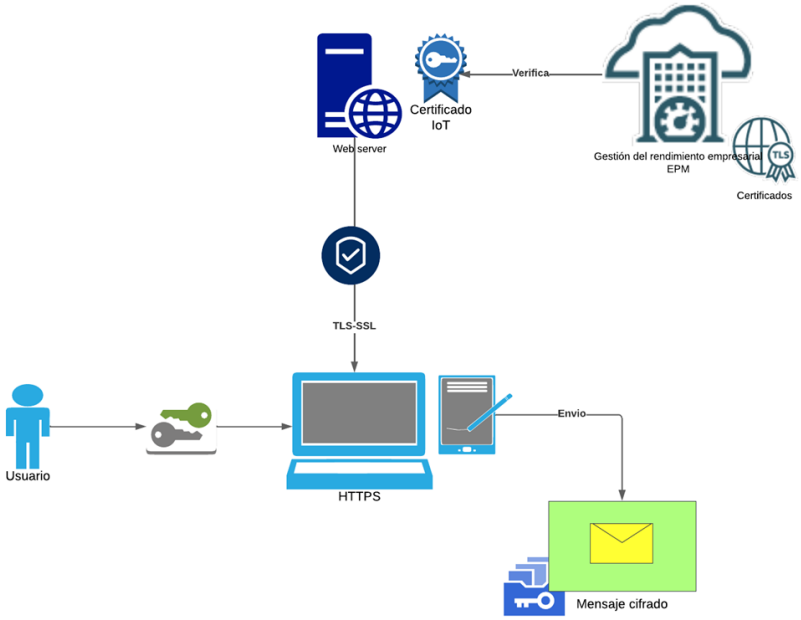
Este sistema resuelve problemas de gestión de claves mediante directorios y certificados, asegurando la autenticidad de las claves públicas a través de Autoridades de Certificación (CAs) [16].

PKI incluye componentes como la Autoridad de Registro (RA), políticas de seguridad y repositorios de certificados, esenciales para actividades como comercio electrónico, banca en línea y correo electrónico seguro [17]. Sus orígenes se remontan 1976 con el trabajo de

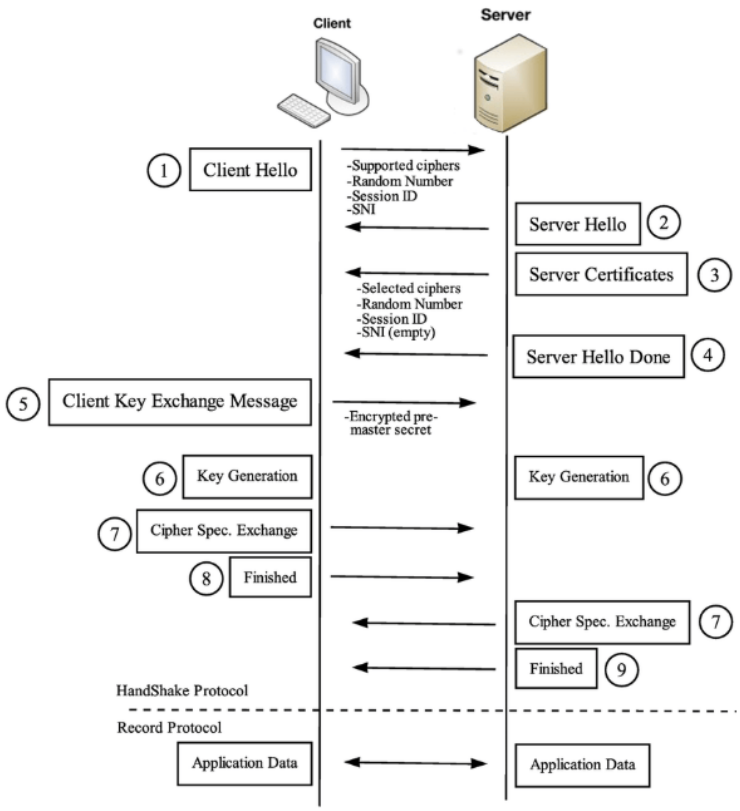
Diffie y Hellman, y estándares como X.500 y X.509 de la ITU en los años 80 y 90 consolidaron su uso.

4. Diseño esquemático de funcionamiento

1. **Cliente (Navegador):** El proceso comienza cuando el usuario accede a una página web mediante el protocolo https://
2. **Servidor web:** El servidor responde con un mensaje, proporciona el contenido cifrado.
3. **Certificado digital:** Garantiza que el servidor es autentico y contiene su clave pública.
4. **Autoridad de certificación (CA):** Verifica el certificado usando una firma digital de una autoridad de certificación.
5. **Intercambio de claves TLS/SSL:** Comienza el intercambio de claves para establecer una clave de sesión simétrica, utilizando criptografía asimétrica.
6. **Cifrado de los datos HTTPS activo:** Protege la trasferencia de los datos atreves de claves simétrica.



El cliente y el servidor establecen una conexión segura utilizando protocolos TLS, esto también se lo conoce como TLS Handshake [9].



TLS handshake protocol

5. Usos del acceso web seguro (HTTPS)

5.1 Comercio Electrónico (E-Commerce)

Ganarse la confianza del usuario es especialmente importante para los negocios en línea, como las tiendas de comercio electrónico. Los clientes potenciales necesitan la seguridad de que sus datos de pago no se verán comprometidos. Los propietarios de sitios web sin HTTPS no solo ponen en riesgo la privacidad de sus clientes, sino también su propia reputación. Los atacantes pueden acceder fácilmente a la información del cliente a través de conexiones no seguras. Una brecha de seguridad de este tipo podría disuadir a los usuarios de realizar futuras transacciones con la empresa debido a la pérdida de confianza.

5.2 Portales Bancarios o financieros

Un usuario accede a su cuenta bancaria en línea para realizar transacciones o consultar su saldo.

Importancia de HTTPS:

- **Seguridad en la autenticación:** HTTPS garantiza que las credenciales de acceso se transmitan de forma segura, evitando su robo.
- **Protección contra suplantación:** Ayuda a prevenir que los usuarios sean redirigidos a sitios falsos que imitan al banco legítimo.
- **Cumplimiento normativo:** El uso de HTTPS es esencial para cumplir con regulaciones de seguridad financiera y proteger la información del cliente.

5.3 Inicios de sesión en plataformas web

HTTPS se utiliza para todos los sitios web en los que un usuario introduce datos. Un campo de aplicación importante es la banca online. En cualquier lugar donde se utilice una cuenta protegida por contraseña, sería sensato tener una conexión HTTPS. Esto incluye el acceso a redes sociales, o cuentas de correo electrónico y de compras, en las que de otro modo se podría causar un gran daño personal con la adquisición ilegal de datos personales. La información personal también puede ser enviada sin una cuenta. Si, por ejemplo, un vuelo o unas vacaciones enteras se reservan en línea, entonces los datos aplicables deben ser comunicados a los proveedores de una manera segura.

6. Referencias bibliográficas

[1] S.-W. Han, H. Kwon, C. Hahn, D. Koo, and J. Hur, "A survey on MITM and its countermeasures in the TLS handshake protocol," *International Conference on Ubiquitous and Future Networks*, pp. 724–729, Jul. 2016, doi: 10.1109/ICUFN.2016.7537132.

[2] S. Keshvadi and Y. Sharma, "Exploring HTTPS Certificate Ecosystem: Analyzing the Entire IPv4 Address Space," pp. 547–552, Sep. 2023, doi: 10.1109/ccece58730.2023.10288980.

[3] A. S. Wazan, R. Laborde, D. W. Chadwick, F. Barrere, and A. Benzekri, "TLS Connection Validation by Web Browsers: Why do Web Browsers Still Not Agree?," *Computer Software and Applications Conference*, vol. 1, pp. 665–674, Jul. 2017, doi: 10.1109/COMPSAC.2017.240.

[4] J. Simaltoga, "X.509 Digital Certificate," Apress eBooks, 2022, pp. 45–73. doi: 10.1007/978-1-4842-7486-6_5.

[5] "¿Qué es el Protocolo HTTPS y por qué es tan importante?" The #1 Platform for Website User Experience. Accedido el 25 de mayo de 2025. [En línea]. Disponible:

https://es.ryte.com/wiki/HTTPS#:~:text=protección%20de%20datos.-,Uso%20y%20relevancia,un%20pequeño%20icono%20de%20candado.

[6] C. Chipeta. “What is HTTPS? How it Works and Why It's So Important | UpGuard”. Third-Party Risk and Attack Surface Management Software | UpGuard. Accedido el 25 de mayo de 2025. [En línea]. Disponible: https://www.upguard.com/blog/what-is-https

[7] “Criptografía: Modelo compacto del algoritmo Diffie-Hellman utilizando cuaterniones”. DSpace Angular: Home. Accedido el 25 de mayo de 2025. [En línea]. Disponible: https://repositorio.uai.edu.ar/handle/123456789/2781

[8] “Software educativo para el aprendizaje de Criptografía de Curva Elíptica”. Inicio. Accedido el 25 de mayo de 2025. [En línea]. Disponible: https://reunir.unir.net/handle/123456789/12792

[9] E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.3," RFC 8446, 2018. DOI: 10.17487/RFC8446

[10] M. Myers, R. Ankney, A. Malpani, S. Galperin y C. Adams, "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP," RFC 2560, IETF, Jun. 1999. [En línea]. Disponible en: https://www.rfc-editor.org/rfc/rfc2560.html.

[11] A. Deacon y R. Hurst, "The Lightweight Online Certificate Status Protocol (OCSP) Profile for High-Volume Environments," RFC 5019, IETF, Sep. 2007. [En línea]. Disponible en: https://datatracker.ietf.org/doc/html/rfc5019.

[12] K. Ivanov, "Autonomous collision attack on OCSP services," arXiv preprint, arXiv:1609.03047, 2016. [En línea]. Disponible en: https://arxiv.org/abs/1609.03047.

[13] “RFC 2818: HTTP Over TLS”. IETF Datatracker. [En línea]. Disponible: https://datatracker.ietf.org/doc/html/rfc2818

[14] “Public Key Pinning Extension for HTTP”. RFC Editor. [En línea]. Disponible: https://www.rfc-editor.org/rfc/pdfrfc/rfc7469.txt.pdf

[15] A. Albarqi, E. Alzaid, F. A. Ghamdi, S. Asiri, and J. Kar, “Public Key Infrastructure: a survey,” Journal of Information Security, vol. 06, no. 01, pp. 31–37, Jan. 2015, doi: 10.4236/jis.2015.61004.

[16] H. Li and Y. Wang, “Public-Key infrastructure,” in Springer eBooks, 2003, pp. 39–70. DOI: 10.1007/978-3-662-05322-5_3.

[17] J. Huang and D. M. Nicol, “An anatomy of trust in public key infrastructure,” International Journal of Critical Infrastructures, vol. 13, no. 2/3, p. 238, Jan. 2017, doi: 10.1504/ijcis.2017.088234.

[18] National Institute of Standards and Technology (NIST), Digital Signature Standard (DSS), FIPS PUB 186-4, 2013. [En línea]. Disponible: https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf.

[19] D. Johnson, A. Menezes, and S. Vanstone, “The Elliptic Curve Digital Signature Algorithm (ECDSA),” International Journal of Information Security, vol. 1, no. 1, pp. 36–63, 2001. DOI: 10.1007/s102070100002.

[20]“¿Qué es una función criptográfica de hash?” SSL.com. Accedido el 25 de mayo de 2025. [En línea]. Disponible: https://www.ssl.com/es/artículo/¿Qué-es-una-función-hash-criptográfica?/

[21] “SHA-384 — PyCryptodome 3.23.0 documentation”. Welcome to PyCryptodome’s documentation — PyCryptodome 3.23.0 documentation. Accedido el 25 de mayo de 2025. [En línea]. Disponible: https://pycryptodome.readthedocs.io/en/latest/src/hash/sha384.html