Trabajo de Fin de Grado

GRADO EN INFORMÁTICA

Facultad de Matemáticas e Informática

Universidad de Barcelona

Implementación de una plataforma web sobre Encriptado de Datos recomendada.

Autor: Asier Augusto Schaefer

Director: Nahuel Norberto Statuto
Realizado en: Departamento de Matemáticas e Informática

Barcelona June 12, 2024

Resumen

A medida que el tiempo pasa hemos ido evolucionando como sociedad y ha habido una transición de era constante. Actualmente nos encontramos en la era moderna y la seguridad digital se ha convertido en un pilar fundamental, donde la información fluye de manera constante a través de las redes. En este contexto, este Trabajo de Fin de Grado se centra en el estudio de la criptografía, una disciplina esencial para garantizar la seguridad de la información y este proyecto tiene como objetivo demostrar cuán importante es esta disciplina en base a explicar gran parte del marco teórico que abarca la criptografía.

El trabajo aborda tanto el cifrado simétrico como el asimétrico, así como los hashes, la infraestructura de clave pública, otras aplicaciones de la criptografía y el criptoanálisis entre otras cosas. Se explora la historia de la criptografía, destacando su origen en necesidades como la defensa de la información en Inglaterra durante la Segunda Guerra Mundial.

Por otro lado, se presenta una aplicación práctica de estos conceptos a través de una página web desarrollada como parte de este proyecto. Esta página, creada con VUE.js en el *frontend* y Python en el *backend* para la seguridad, permite a los usuarios introducir distintos tipos de datos sensibles que se encriptan utilizando el mejor algoritmo posible de encriptación y hash, incluyendo una explicación sobre qué algoritmo se ha utilizado y por qué. Además, hay una sección donde se hace una conclusión sobre cuál es el algoritmo más seguro de hash y de encriptación, ya que hay un algoritmo que es el que más se utiliza en la página web para encriptar datos.

La página web tiene como objetivo explicar cuáles son los algoritmos de hash y encriptación más utilizados hoy en día y los más seguros para proteger datos sensibles.

Resum

A mesura que el temps passa hem anat evolucionant com a societat i ha hagut una transició d'era constant. Actualment ens trobem a l'era moderna i la seguretat digital s'ha convertit en un pilar fonamental, on la informació flueix de manera constant a través de les xarxes. En aquest context, aquest Treball de Fi de Grau es centra en l'estudi de la criptografia, una disciplina essencial per garantir la seguretat de la informació i aquest projecte té com a objectiu demostrar com d'important és aquesta disciplina basant-se en explicar gran part del marc teòric que abasta la criptografia.

El treball aborda tant el xifrat simètric com l'asimètric, així com els hashes, la infraestructura de clau pública, altres aplicacions de la criptografia i el criptoanàlisi entre altres coses. S'explora la història de la criptografia, destacant el seu origen en necessitats com la defensa de la informació a Anglaterra durant la Segona Guerra Mundial.

D'altra banda, es presenta una aplicació pràctica d'aquests conceptes a través d'una pàgina web desenvolupada com a part d'aquest projecte. Aquesta pàgina, creada amb VUE.js al frontend i Python al backend per a la seguretat, permet als usuaris introduir diferents tipus de dades sensibles que s'encripten utilitzant el millor algoritme possible d'encriptació i hash, incloent una explicació sobre quin algoritme s'ha utilitzat i per què. A més, hi ha una secció on es fa una conclusió sobre quin és l'algoritme més segur de hash i d'encriptació, ja que hi ha un algoritme que és el que més s'utilitza a la pàgina web per encriptar dades.

La pàgina web té com a objectiu explicar quins són els algoritmes de hash i encriptació més utilitzats avui en dia i els més segurs per protegir dades sensibles.

Abstract

As time passes, we have been evolving as a society and there has been a constant era transition. Currently, we are in the modern era and digital security has become a fundamental pillar, where information flows constantly through networks. In this context, this Final Degree Project focuses on the study of cryptography, an essential discipline to guarantee information security and this project aims to demonstrate how important this discipline is based on explaining much of the theoretical framework that encompasses cryptography.

The work addresses both symmetric and asymmetric encryption, as well as hashes, public key infrastructure, other applications of cryptography and cryptanalysis among other things. It explores the history of cryptography, highlighting its origin in needs such as the defense of information in England during the Second World War.

On the other hand, a practical application of these concepts is presented through a website developed as part of this project. This page, created with VUE.js on the frontend and Python on the backend for security, allows users to enter different types of sensitive data that are encrypted using the best possible encryption and hash algorithm, including an explanation of which algorithm has been used and why. In addition, there is a section where a conclusion is made about which is the most secure hash and encryption algorithm, since there is an algorithm that is the most used on the website to encrypt data.

The website aims to explain which are the most used hash and encryption algorithms today and the safest to protect sensitive data.

Índice

1 Introducción

1.1 Objetivos

En la actualidad, la criptografía juega un papel crucial en la protección de la información y la privacidad en numerosos campos. El objetivo de este Trabajo de Fin de Grado es explorar y destacar la importancia y aplicabilidad de la criptografía, con el fin de concienciar al lector sobre su relevancia en el mundo actual.

Para ello, se realizará un análisis exhaustivo de los diferentes campos en los que la criptografía tiene utilidad, proporcionando una explicación detallada de cada uno de ellos. Además, se desarrollará una página web interactiva que permitirá a los usuarios introducir datos sensibles y observar el resultado de la encriptación o el hasheo de estos datos. Esta página web no solo mostrará el resultado de la encriptación, sino que también proporcionará una explicación detallada del algoritmo de encriptación más seguro utilizado y las razones de su elección.

Finalmente, la página web contará con una sección dedicada a la conclusión de cuál es el algoritmo más seguro para encriptar y cuál es el algoritmo más seguro para hashear. Esta sección proporcionará una visión clara y concisa de los algoritmos más seguros disponibles en la actualidad para la encriptación y el hasheo de datos.

1.2 Motivación

Desde siempre, he sentido una fascinación por la seguridad informática y la ciberseguridad con sus múltiples ramas y facetas, ha captado mi interés y me ha llevado a querer saber más. A lo largo de estos últimos años, he ido mejorando mi nivel en ciberseguridad y he descubierto que todas las ramas relacionadas con este mundo están intrínsecamente vinculadas a la criptografía. Cada implementación, cada proceso, necesita un sistema de seguridad fiable para resistir a los posibles atacantes informáticos y es aquí donde la criptografía entra en juego, protegiendo la información y los procesos.

Mi motivación para este trabajo es doble: por un lado, quiero hacer entender al mayor número de personas acerca de la importancia de la criptografía en nuestra sociedad digital y creo firmemente que una mayor conciencia sobre este tema puede llevar a una mayor seguridad y protección de la información personal y profesional y, por otro lado, este trabajo también es una oportunidad personal para poder aprender más sobre este sector, por lo que a través de la investigación y el desarrollo de este proyecto, espero adquirir un conocimiento más profundo de la criptografía y sus aplicaciones, lo que me permitirá contribuir de manera más efectiva a la ciberseguridad en el futuro.

En resumen, mi motivación para este Trabajo de Fin de Grado es tanto educativa como personal.

1.3 Estructura de la Memoria

Durante la elaboración de la memoria de este Trabajo de Fin de Grado, se ha ido ajustando y refinando continuamente hasta llegar a un guión definitivo. Esto se debe a que, a medida que se ha ido redactando, han surgido cambios y modificaciones. A continuación, se explicará brevemente la estructura final de esta memoria con cada apartado.

Primero, nos encontramos con una introducción, donde se explica cuáles han sido los objetivos principales de este proyecto de criptografía, de qué trata y la motivación por la cual se ha escogido este trabajo. Dentro de este apartado también encontramos la planificación a través de una metodología ágil y revisiones periódicas que se ha seguido durante la realización de este Trabajo de Fin de Grado.

En la siguiente sección, se introduce la criptografía, sus funcionalidades, su origen, los principios de Kerchkoffs, el nacimiento de la criptografía moderna y su historia.

Seguidamente, se presenta el marco teórico del proyecto. En él se definen conceptos que no se explican durante la redacción de otros puntos de la memoria, tanto en el ámbito de la criptografía simétrica y los hashes, la criptografía asimétrica, la infraestructura de clave pública (PKI), como en el criptoanálisis.

Después, se explican otras aplicaciones de la criptografía, como los algoritmos de cifrado IEEE 802.11 (WiFi), la seguridad en la capa de transporte (SSL/TLS) y las redes privadas virtuales (VPN).

En el siguiente apartado, se presentan los antecedentes de este proyecto de criptografía, mencionando y explicando brevemente las distintas funcionalidades ya implementadas y la estructura de la base de datos.

Acto seguido, se encuentra la estructura del código del proyecto y su despliegue. En este apartado, se explica el funcionamiento básico del frontend y su despliegue, así como del backend y su despliegue.

En el penúltimo capítulo, se detallan las conclusiones, los resultados obtenidos al finalizar el Trabajo de Fin de Grado y el trabajo futuro que podría ser realizado para mejorar este proyecto de criptografía.

Finalmente, se encuentra un anexo en el que se mencionan los documentos de despliegue realizados durante este trabajo y la bibliografía con los enlaces usados para la realización de esta memoria.

2 Introducción a la criptografía

2.1 La criptografía y sus funcionalidades

Antes de nada, es esencial poder responder a las preguntas más básicas sobre esta área, saber qué es la criptografía, cuáles son sus objetivos y qué funcionalidades tiene. Poder responder a estas preguntas consolidarán las bases para profundizar un poco más en este campo.

La criptografía es la técnica encargada de cifrar o codificar comunicaciones para alterar las representaciones lingüísticas de ciertos mensajes con el fin de hacerlos ininteligibles a receptores no autorizados. Explicado de forma más sencilla, se podría decir que es una técnica que se utiliza para proteger la información y funciona transformando la información original, conocida como texto plano, en un formato que no se puede leer o entender, llamado texto cifrado. Solo las personas que tienen una clave especial pueden convertir el texto cifrado de nuevo en texto plano.

El objetivo de la criptografía es ofuscar el contenido de un mensaje para que no pueda ser leído por destinatarios no deseados y sí por el destinatario previsto.

La criptografía engloba muchas funcionalidades, que se pueden resumir en mecanismos de seguridad: encriptación de mensajes, firma digital, autenticación mutua, intercambio de claves y en servicios de seguridad: confidencialidad, integridad, autenticación, no-repudio y control de acceso.

2.2 Origen de la criptografía

La escritura oculta tiene su origen en Egipto hace al menos 3.000 años. A lo largo de la historia, siempre ha habido una necesidad de mantener ciertas comunicaciones ocultas, como por ejemplo las misivas militares.

Un ejemplo destacado de esto se puede encontrar en la historia de la Segunda Guerra Mundial, cuando el Reino Unido se encontraba en una lucha constante por mantener sus comunicaciones seguras y la necesidad de transmitir mensajes de manera segura y eficiente era vital para el éxito de las operaciones militares.

Los alemanes, eran capaces de cifrar mensajes de una manera que era casi indescifrable, sin conocer la configuración exacta de la máquina utilizada para cifrar el mensaje original. En este escenario entra en juego una máquina llamada Enigma que fue inventada por Arthur Scherbius [1] .

En Bletchley Park, Inglaterra, un equipo de matemáticos y criptógrafos, liderado por Alan Turing, trabajó incansablemente para descifrar los códigos de Enigma y para

ello crearon una máquina, conocida como “la Bomba”, que podía descifrar los códigos de Enigma más rápido de lo que cualquier humano podría hacerlo.

Este trabajo de descifrado no solo cambió el curso de la guerra, sino que también sentó las bases para la criptografía moderna y la informática, así que, en cierto sentido, la necesidad de disponer de comunicaciones seguras durante la guerra llevó a algunos de los avances más significativos en la criptografía.

2.3 Principios de Kerchkoffs

En 1883, Auguste Kerckhoffs publicó un artículo que estableció las propiedades deseables para cualquier sistema criptográfico:

1. Si el sistema no es teóricamente irrompible, al menos debe serlo en la práctica: esto significa que, aunque un sistema criptográfico pueda ser descifrado teóricamente, debe ser tan difícil hacerlo que en la práctica sea considerado seguro.
2. La efectividad del sistema no debe depender de que su diseño permanezca en secreto: este es a menudo conocido como el “principio de Kerckhoffs” y establece que la seguridad de un sistema criptográfico debe depender únicamente del secreto de la clave, no del algoritmo o del diseño del sistema.
3. La clave debe ser fácilmente memorizable de manera que no haya que recurrir a notas escritas: esto es importante para evitar que las claves sean descubiertas por alguien más.
4. Los criptogramas deberán dar resultados alfanuméricos: esto significa que los mensajes cifrados deben ser legibles y comprensibles.
5. El sistema debe ser operable por una única persona: esto significa que una sola persona debe ser capaz de operar el sistema criptográfico sin necesidad de ayuda.
6. El sistema debe ser fácil de utilizar: esto es importante para asegurar que el sistema criptográfico se pueda usar de manera efectiva y eficiente.

2.4 Nacimiento de la criptografía moderna

La criptografía moderna nace con la Teoría de la Información, desarrollada por Claude Shannon en 1949 [2]. Esta teoría se centra en las leyes matemáticas que rigen la transmisión y el procesamiento de la información, considerando la información como una entidad independiente de su contenido o significado.

Dentro de esta teoría, Shannon introdujo tres conceptos fundamentales para la criptografía: confusión, difusión y efecto avalancha.

1. **Confusión:** Este concepto se refiere a cómo cada bit del texto cifrado (la información codificada) debe depender de varias partes de la clave, de manera que se oscurezcan las conexiones entre ambos. En otras palabras, si cambiamos un solo bit en una clave, la mayoría o todos los bits en el texto cifrado se verán afectados. Esto ayuda a ocultar la relación entre el texto cifrado y la clave.
2. **Difusión:** Este concepto significa que si cambiamos un solo bit del texto sin formato (la información original), entonces aproximadamente la mitad de los bits del texto cifrado deberían cambiar. El propósito de la difusión es ocultar la relación estadística entre el texto cifrado y el texto plano. Por ejemplo, la difusión garantiza que cualquier patrón en el texto sin formato no sea evidente en el texto cifrado.
3. **Efecto Avalancha:** Este es un efecto deseado en la criptografía donde un pequeño cambio en la entrada (ya sea el texto sin formato o la clave) produce cambios drásticos en la salida (el texto cifrado). Esencialmente, es una extensión del concepto de difusión.

2.5 Historia de la criptografía moderna

Como ya se ha comentado antes, la criptografía ha sido una herramienta esencial para la seguridad y la privacidad de la información. En la era moderna, la criptografía ha evolucionado de formas simples de cifrado a sistemas complejos que son fundamentales para la seguridad de la información digital. En esta sección, nos centraremos en dos desarrollos clave de la criptografía moderna: el nacimiento de la criptografía simétrica y de la criptografía asimétrica.

La criptografía simétrica, también conocida como criptografía de clave secreta, es el método más antiguo de codificación de información. En este sistema, la misma clave se utiliza tanto para cifrar como para descifrar el mensaje. Un ejemplo clásico de criptografía simétrica es el cifrado César, utilizado por Julio César para comunicarse con sus generales. Es un tipo de cifrado por sustitución en el que cada letra del texto original se reemplaza por una letra cierto número de posiciones más adelante en el alfabeto. Por ejemplo, con un desplazamiento de 1, la A sería sustituida por la B, la B se convertiría en C y así sucesivamente.

Sin embargo, la verdadera era de la criptografía simétrica moderna comenzó con el desarrollo de la máquina Enigma durante la Segunda Guerra Mundial. Esta máquina, utilizada por los alemanes, empleaba un sistema de rotores y una clave que cambiaba diariamente para cifrar los mensajes. Como se ha comentado antes, pese a que finalmente fuese descifrada por los ingleses, Enigma marcó un hito en la historia de la criptografía simétrica.

Figure 1: Máquina enigma

La criptografía asimétrica, también conocida como criptografía de clave pública, revolucionó el campo de la criptografía. En este sistema, se utilizan dos claves diferentes: una para cifrar el mensaje y otra para descifrarlo. Esto resuelve el problema de tener que compartir una clave secreta de forma segura, como se requiere en la criptografía simétrica.

El nacimiento de la criptografía asimétrica se atribuye a Whitfield Diffie y Martin Hellman, quienes introdujeron el concepto en 1976. Su protocolo, conocido como intercambio de claves Diffie-Hellman, permitió a dos partes generar una clave compartida de forma segura a través de un canal inseguro. Este fue un avance monumental que sentó las bases para muchos de los sistemas de criptografía que usamos hoy en día, más adelante profundizaremos más sobre este algoritmo.

3 Marco teórico

3.1 Criptografía Simétrica y hashes

Antes de comenzar a comentar información sobre encriptación o hashes, es importante saber qué diferencia hay entre ambos conceptos.

3.1.1 Criptografía Simétrica

La criptografía de clave simétrica es el método criptográfico en el que se emplea la misma clave para cifrar y descifrar un mensaje. Una clave en criptografía es un código secreto que se utiliza para transformar un mensaje en una versión cifrada y luego volver a convertirlo en su forma original. En la criptografía simétrica la misma clave se utiliza tanto para el cifrado como para el descifrado del mensaje.

Hay dos tipos de cifrado de clave simétrica que son los más utilizados: cifrado de flujo y cifrado de bloque [3] [4].

El cifrado de flujo es un tipo de cifrado donde cada bit o byte del mensaje se cifra uno por uno con la clave, mientras que el cifrado de bloque cifra el mensaje en bloques de un tamaño determinado en lugar de uno por uno.

Ambos métodos utilizan la misma clave para el cifrado y el descifrado, pero difieren en cómo se aplica la clave al mensaje.

Las ventajas de utilizar el cifrado simétrico es que es un método simple y veloz a la hora de cifrar y descifrar. Por otro lado, el inconveniente es que las dos partes que se comunican tienen que ponerse de acuerdo sobre la clave (intercambio de claves).

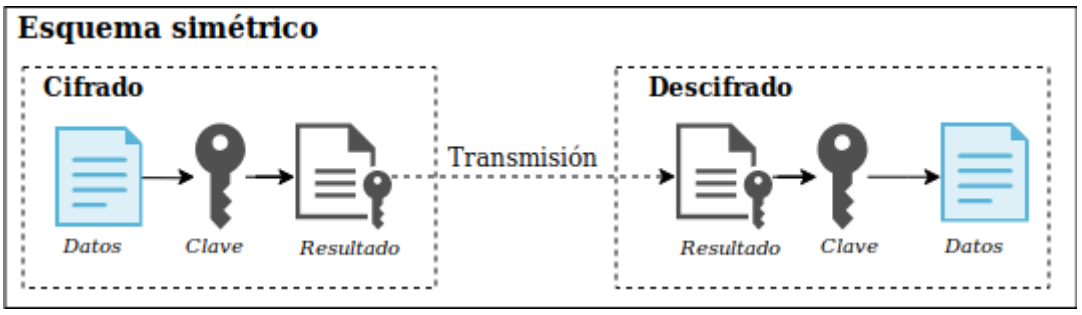

Figure 2: Esquema simétrico

3.1.1.1 Algoritmos de cifrado de flujo

Hay varios algoritmos de cifrado de flujo, pero sin lugar a duda, el algoritmo más importante entre todos estos y el más ampliamente utilizado es el RC4 [5].

Este algoritmo fue creado RSA Data Security® y se utiliza en muchos productos de software, así como en protocolos de internet para asegurar comunicaciones seguras a través de este.

RC4 es un esquema de cifrado de flujo simétrico extremadamente simple y como se acaba de mencionar, puede implementarse en software de forma muy eficiente. Esto lo ha convertido en uno de los esquemas de cifrado más utilizados del mundo.

Este algoritmo es considerado inseguro en ciertos contextos, pero pese a esto, sigue siendo ampliamente utilizado y es conocido por ser el mismo esquema de cifrado utilizado por WEP[1] y se estima que aún se utiliza en aproximadamente la mitad de las transmisiones TLS[2] que ocurren en el mundo actualmente.

Aunque RC4 tiene vulnerabilidades conocidas, como se ha mencionado antes, se sigue utilizando muy a menudo hoy en día y eso es debido a varios motivos. Para empezar, RC4 no está completamente roto, lo que significa que todavía puede proporcionar un nivel de seguridad aceptable en ciertos contextos. Además, RC4 es apreciado por su rapidez y facilidad de implementación, lo que lo hace atractivo para su uso.

La vulnerabilidad mencionada antes de RC4 está principalmente relacionada con la gestión de claves y el tratamiento del vector de inicialización. Esto significa que la clave con la cual se encripta un dato puede ser fácilmente desvelada y esto deja expuesto estos datos. Cuando se descubre la clave que se utiliza para encriptar los datos, entonces se puede utilizar esa misma clave para desencriptar los datos, ya que como se ha explicado, la encriptación simétrica utiliza la misma clave para encriptar y desencriptar los valores introducidos.

Una vez aclarado esto, se van a poner dos ejemplos visuales del uso de este algoritmo, para esto, nos apoyaremos en un encriptador/desencriptador online del algoritmo RC4. Cuando encriptamos una palabra o frase utilizando un encriptador online, la página web se encarga de crear una llave para poder encriptar el dato introducido mediante el uso de esta, o en otros casos, se pide al usuario que introduzca una llave manualmente, de forma que si luego se intenta desencriptar el dato introducido o cualquier otro creado en esa misma página, entonces el proceso será exitoso y se desencriptará correctamente el input, esto es debido a que la página web utilizará esa misma clave para ambos procesos.

[1] Wired Equivalent Privacy, es el estándar de seguridad Wi-Fi original, ahora considerado obsoleto y vulnerable.
[2] Transport Layer Security, es un protocolo de seguridad que protege la comunicación en la red mediante el cifrado de los datos transmitidos.

Vamos a utilizar la página web “https://www.flashbit.site/es/rc4/” para poner un ejemplo. En este, se va a introducir un input sencillo para encriptar:

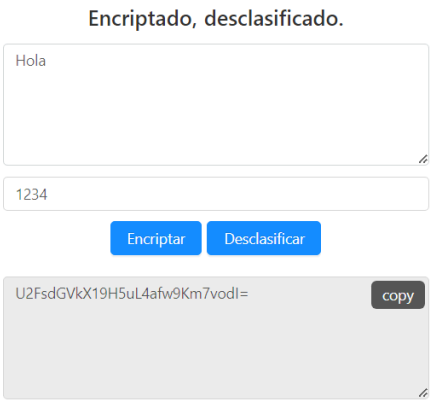

Figure 3: Encriptación de un dato con RC4

El dato introducido ha sido “Hola” y el valor encriptado saliente ha sido “U2FsdGVkX19H5uL4afw9Km7vodI=”. En este caso se ha introducido una clave manualmente, concretamente la clave es “1234”.

Si ahora probamos a desencriptar el resultado anterior utilizando la misma clave introducida previamente, podremos observar como el resultado de la desencriptación funciona:

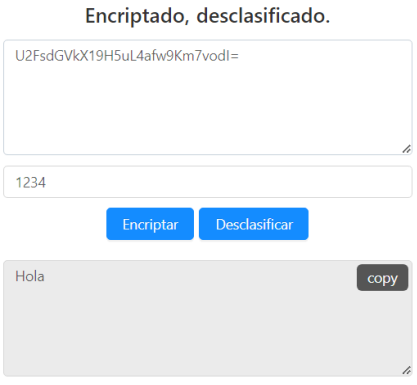

Figure 4: Desencriptación exitosa de un dato con RC4

Como podemos apreciar sí funciona y esto es debido a que la página web conoce la clave con la que se ha encriptado los datos.

Ahora en el segundo ejemplo que se puede apreciar en la figura 5, vamos a probar a desencriptar este mismo dato que acabamos de descifrar, pero usando una clave distinta:

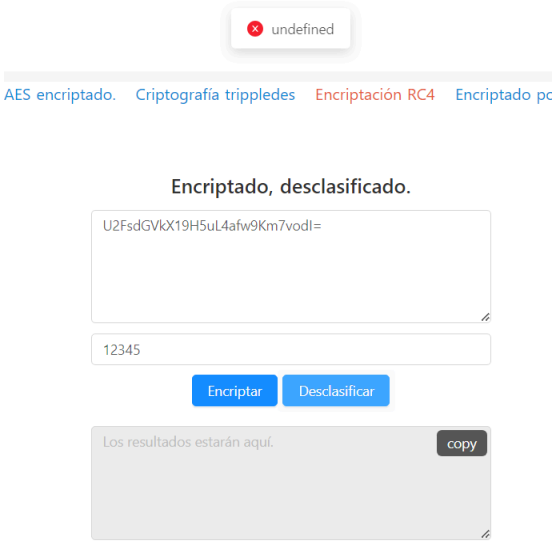

Figure 5: Desencriptación fallida de un dato con RC4

Nos sale el error “undefined”, que en cualquier caso quiere decir que ha habido un error y no se ha podido descifrar correctamente el dato cifrado. Esto es debido a que la clave no es la misma y como esta se desconoce, entonces es imposible conseguir el resultado del dato original en texto plano.

Con este ejemplo podemos llegar a la conclusión de que realmente el único problema del cifrado RC4 es que tiene vulnerabilidades con relación a lo fácil que puede llegar a ser descubrir la clave utilizada para encriptar los datos, y por tanto esto puede generar una posible brecha de seguridad cuando se utiliza este algoritmo.

3.1.1.2 Algoritmos de cifrado por bloques

Los algoritmos de cifrado por bloques más utilizados debido a que han sido estándares de la industria (NIST) siendo la referencia para sus desarrollos e implementaciones son: DES, 3DES y AES [6].

DES (Data Encryption Standard) es un algoritmo de cifrado simétrico por bloques que transforma bloques de 64 bits de datos en texto cifrado utilizando una clave de 56 bits. El problema principal de usar una clave relativamente tan pequeña es que este algoritmo se vuelve susceptible a ataques de fuerza bruta[3], lo cual permite al atacante probar todas las combinaciones posibles hasta encontrar la clave correcta

[3] Los ataques de fuerza bruta son intentos de descifrar una contraseña o clave mediante la prueba sistemática de todas las posibles combinaciones hasta encontrar la correcta.

que descifra el texto. Aunque fue considerado seguro en su momento, DES ya no se considera seguro contra ataques modernos debido a la longitud relativamente corta de su clave. De hecho, el NIST[4] ha deprecado su uso en nuevas aplicaciones desde 2017 y en aplicaciones antiguas en 2023.

3DES (Triple DES) es una mejora del DES que aplica el algoritmo DES tres veces a cada bloque de datos. 3DES es más seguro que DES debido a su clave más larga de 168 bits, pero también es más lento. Hoy en día no se considera un buen algoritmo, ya que además de ser lento, también es vulnerable a un tipo de ataque llamado “Ataque de Sweet32”, que en resumen permite a un atacante obtener texto plano a partir de textos cifrados bajo ciertas condiciones. El NIST también ha deprecado su uso en nuevas aplicaciones desde 2017 y en aplicaciones antiguas desde 2023.

AES (Advanced Encryption Standard) es el estándar actual para el cifrado simétrico por bloques. AES puede manejar bloques de 128 bits y permite longitudes de clave de 128, 192 y 256 bits. AES es más seguro y eficiente que DES y 3DES, además es ampliamente utilizado en aplicaciones de seguridad de la información en todo el mundo.

3.1.1.3 Modos de operación de Cifrado por Bloques.

Los modos de operación del cifrado de bloques son técnicas que se utilizan para aplicar algoritmos de cifrado de bloques a datos de longitud mayor que el tamaño de bloque del algoritmo. Estos modos permiten el cifrado y descifrado de grandes cantidades de datos, en bloques, utilizando una clave secreta. Dicho de otra forma, más sencilla de entender, se podría decir que los modos de operación del cifrado de bloques son como recetas que nos dicen cómo usar las herramientas de cifrado para proteger mucha información, no solo un pequeño bloque.

Hay varios algoritmos para la transformación criptográfica, pero los más comunes son dos, primero hablemos del Modo de Cifrado de Bloque Electrónico (ECB). En este modo, cada bloque de texto plano se cifra de forma independiente con la misma clave, por lo que la ventaja de este modo es su simplicidad. Sin embargo, tiene una desventaja importante: si dos bloques de texto plano son idénticos, sus bloques cifrados también serán idénticos. Esto puede revelar patrones en el texto cifrado, lo que puede ser una vulnerabilidad.

[4] National Institute of Standards and Technology, es una agencia del gobierno de EE.UU. que desarrolla y promueve estándares de medición y tecnología.

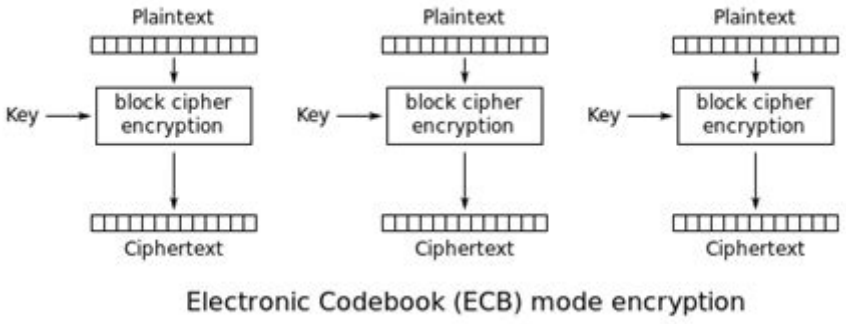

Figure 6: Modo de Cifrado de Bloque Electrónico

Por otro lado, el otro algoritmo también muy reconocido es el Modo de Cifrado de Bloque en Cadena (CBC). Este algoritmo agrega un paso adicional para hacer que el cifrado sea más seguro. Antes de cifrar cada bloque de texto plano, se combina con el bloque cifrado anterior mediante una operación de “O exclusivo” (XOR) y esto asegura que incluso si el texto plano tiene bloques idénticos, los bloques cifrados serán diferentes. La ventaja de CBC es que oculta los patrones en el texto plano, sin embargo, una desventaja es que un error en un bloque cifrado puede afectar a los bloques cifrados subsiguientes.

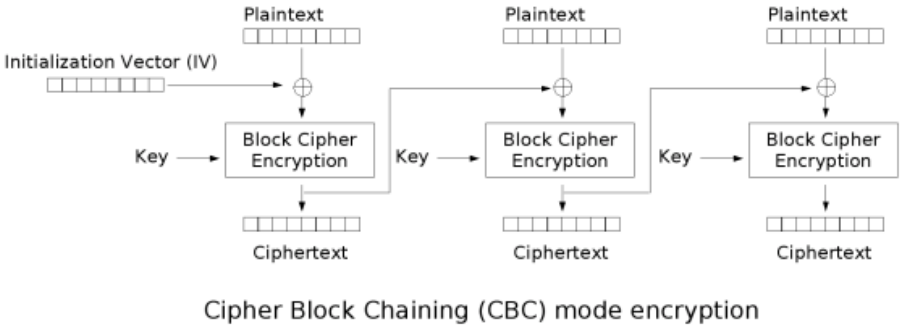

Figure 7: Modo de Cifrado de Bloque en Cadena

3.1.2 Hashing

El *Hashing* es el tipo más simple de operación criptográfica [7]. Por poner un ejemplo fácil de entender, se podría decir que un hash es como una máquina que transforma cualquier cosa que le des, sin importar su tamaño, en un resultado único y de tamaño constante, conocido como valor hash o simplemente, hash.

La referencia al “tamaño constante”, significa que no importa cuán grande o pequeño sea lo que le des a la máquina, siempre ~~te~~ dará un resultado del mismo tamaño.

Hay que pensar en este proceso como si se estuvieras asignando un número de serie único a un objeto. No importa cuán grande o pequeño sea el objeto, cada uno tiene su propio número de serie para identificarlo. Al igual que cada libro tiene un ISBN único, cada entrada de datos tiene un hash único.

Esta máquina toma cualquier cosa que se le de, desde un simple mensaje de texto hasta una biblioteca entera de libros electrónicos y la transforma en un hash único mediante una serie de cálculos complejos. Es importante destacar que incluso el cambio más mínimo que se le da a la máquina, como cambiar una sola letra, resultará en un hash completamente diferente.

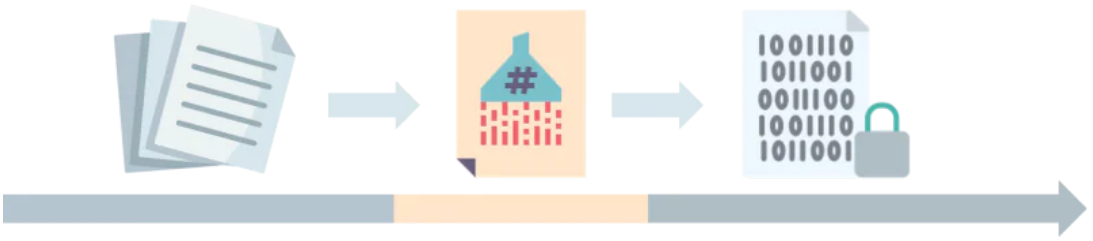

Figure 8: Proceso de hashing

Vamos a ejemplificar el uso de un hash. Para esto se va a utilizar la siguiente página web online de conversión de texto plano a hash MD5: "https://crypt.uticraft.com/es/online/cifrado/hash/a/md5". Si introducimos como dato de entrada la palabra de ejemplo "Brian", entonces nos sale lo siguiente:

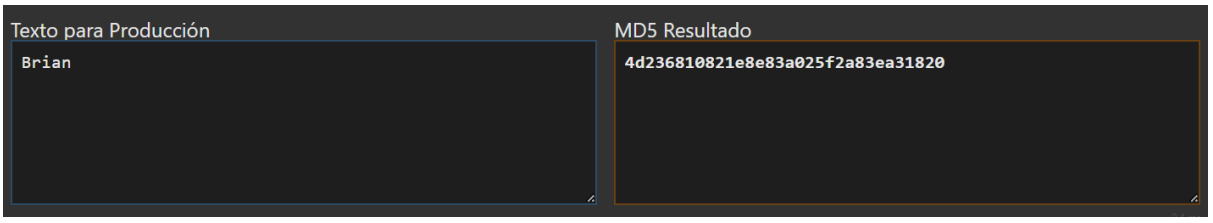

Figure 9: Encriptación de un dato con MD5

Se puede ver que el resultado es "4d236810821e8e83a025f2a83ea31820".

Ahora si se cambia el dato de entrada y se introduce el valor "Brain" por ejemplo, que no difiere mucho al anterior utilizado, entonces nos sale el siguiente output:

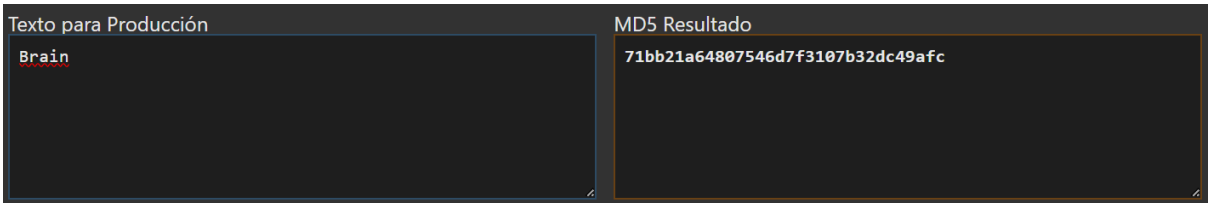

Figure 10: Encriptación de un dato con MD5

Se puede observar en la figura 9 que, en este caso, el resultado es “71bb21a64807546d7f3107b32dc49afc”.

Este valor resultante es totalmente distinto al anterior y el cambio del dato entrante es prácticamente insignificante en comparación a la diferencia que hay entre los dos resultados. Esta es la naturaleza de un hash, es capaz de cambiar el resultado de manera completa con tan solo un pequeño cambio en el valor del input.

Una vez introducido este nuevo concepto denominado hash, vamos a profundizar un poco más y ver los principales algoritmos de hash.

3.1.2.1 Implementaciones populares

Dentro de los algoritmos de hash, dos de las implementaciones más populares son los algoritmos de la familia SHA y MD5 [8].

Secure Hash Algorithm (SHA) es una de estas dos populares familias de funciones de hash criptográficas. Existen varias versiones de SHA, incluyendo SHA-1, SHA-2 y SHA-3, cada una con diferentes niveles de seguridad y tamaños de hash. La variante más popular entre esta familia de algoritmos es la conocida SHA-256.

A diferencia de SHA-1, que es vulnerable a ataques de colisión, SHA-256 es resistente a estos ataques, lo que significa que es muy poco probable que dos entradas diferentes produzcan el mismo hash. Además, es un algoritmo muy popular debido a su equilibrio entre eficiencia y seguridad con un hash de 256 bits y su uso generalizado en numerosas aplicaciones y protocolos de seguridad.

Por otro lado, está Message Digest Algorithm (MD5). Este algoritmo produce una salida de 128 bits. De hecho, este algoritmo no se considera seguro, pero hoy en día todavía se requiere por compatibilidad de algunas herramientas de seguridad. Este algoritmo a diferencia del anterior explicado, no es resistente a ataques de colisión, por lo que dos entradas iguales pueden producir la misma salida, lo cual convierte oficialmente a este algoritmo como inseguro. Su futura versión MD6 se ha desarrollado, pero aún es computacionalmente demasiado lenta, por lo que no es popular y aún está en proceso de mejora.

3.1.2.2 Funciones criptográficas basadas en hashes | HMAC

MAC, que significa Código de Autenticación de Mensajes, es como una firma secreta que se añade a un mensaje. Esta firma, que se crea usando una clave secreta, ayuda a confirmar que el mensaje no ha sido alterado y que proviene de la fuente correcta.

HMAC o Código de Autenticación de Mensajes basado en hash, es un tipo especial de MAC, utiliza funciones de hash, que son formas de convertir los datos en un conjunto único de caracteres, junto con una clave secreta. La fortaleza de HMAC depende de la función de hash que se utilice, el tamaño de la firma que se crea y la calidad de la clave secreta [9].

Poniendo un ejemplo para poder entenderlo vamos a usar la página web "https://gchq.github.io/CyberChef/" (CyberChef) para generar un HMAC para este ejemplo.

Imaginemos que se quiere enviar un mensaje secreto a un amigo con una clave secreta que sólo conocemos nosotros y nuestro amigo. Digamos que son "Hola, amigo" y "clave123" respectivamente. Usando la herramienta CyberChef para generar un HMAC de nuestro mensaje, primero introducimos "Hola, amigo" en el cuadro de entrada, seleccionamos por ejemplo SHA-256 como nuestra función de hash en la operación HMAC e introducimos "clave123" como nuestra clave secreta tal y como se ve en la figura 10:

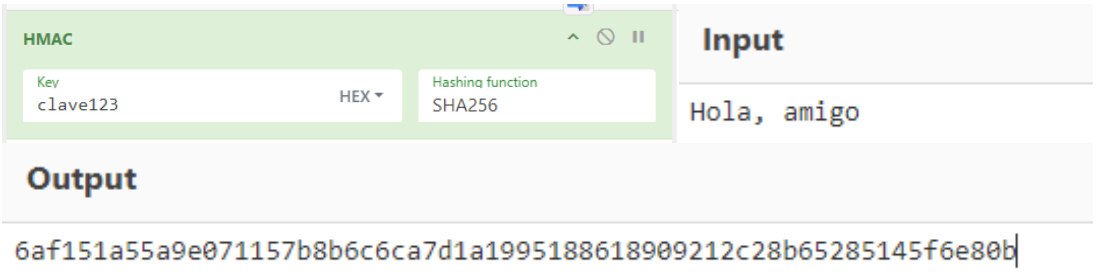

Figure 11: Generación de HMAC con CyberChef

CyberChef genera un HMAC para tu mensaje, en este caso es "6af151a55a9e071157b8b6c6ca7d1a1995188618909212c28b65285145f6e80b".
Este HMAC es un conjunto de caracteres que parece aleatorio, pero en realidad se genera a partir de nuestro mensaje y de nuestra clave secreta utilizando la función de hash SHA-256.

Siguiendo con el proceso que se estaba describiendo antes, se envía el mensaje y el HMAC a nuestro amigo. Cuando nuestro amigo recibe el mensaje y el HMAC, entonces él puede usar la misma función de hash (SHA-256) y la misma clave

secreta (“clave123”) para generar un nuevo HMAC a partir del mensaje que recibió. Si el HMAC que nuestro amigo genera coincide con el HMAC que recibió, entonces él sabe que el mensaje no ha sido alterado y que realmente proviene de nosotros.

3.2 Criptografía Asimétrica

En términos generales, la criptografía de clave asimétrica es un método criptográfico en el que se utilizan dos claves diferentes para cifrar y descifrar un mensaje [10]. Una clave en criptografía es un código secreto que se utiliza para transformar un mensaje en una versión cifrada y luego volver a convertirlo en su forma original. En la criptografía asimétrica, una clave se utiliza para el cifrado y la otra para el descifrado del mensaje .

Las dos claves utilizadas en la criptografía asimétrica son la clave pública y la clave privada. La clave pública, como su nombre indica, es pública y puede ser compartida con cualquier persona. Por otro lado, la clave privada debe mantenerse segura y confidencial, ya que es la única que puede descifrar los mensajes cifrados con la clave pública correspondiente.

El proceso de intercambio de información de manera segura con este tipo de criptografía es bastante sencillo. Por poner un ejemplo, digamos que, si Marta quiere enviar un mensaje a Marc de manera segura, puede cifrar el mensaje con la clave pública de Marc. Una vez cifrado, el mensaje solo puede ser descifrado con la clave privada de Marc, que solo Marc conoce. De esta manera, incluso si alguien intercepta el mensaje cifrado, no podrá descifrarlo sin la clave privada de Marc.

Figure 12: Proceso de encriptación asimétrica

3.2.1 Ventajas y desventajas de la criptografía asimétrica

Las ventajas de utilizar la criptografía asimétrica incluyen la seguridad en la transmisión de datos, ya que incluso si la clave pública es conocida, el mensaje cifrado no puede ser descifrado sin la clave privada [11]. Además, no es necesario intercambiar claves de manera segura antes de la comunicación, ya que la clave pública puede ser compartida abiertamente.

Por otro lado, las desventajas de la criptografía asimétrica incluyen la complejidad y la lentitud en comparación con la criptografía simétrica. El proceso de cifrado y descifrado es más lento y requiere más recursos computacionales. Además, la gestión de las claves privadas es crítica, debido a qué si se pierde o se compromete, los datos cifrados pueden quedar expuestos.

3.2.2 Generador de Números Aleatorios (RNG)

Un generador de números aleatorios (RNG, por sus siglas en inglés) es como una máquina que produce números de una manera que no se puede predecir ni repetir[12]. Cada número que se produce es independiente de los números anteriores y no sigue ningún patrón. En otras palabras, es como si cada número que produce fuera el resultado de una tirada de dados: no importa cuántas veces tires los dados, no puedes predecir qué número saldrá a continuación.

Existen tres tipos de RNGs: los generadores de números aleatorios verdaderos (TRNGs), los generadores de números pseudoaleatorios (PRNGs) y los generadores de números pseudo aleatorios criptográficamente seguros (CSPRNGs).

Los **TRNGs** son dispositivos que generan números a partir de un proceso físico, como el ruido atmosférico o la desintegración radiactiva. Estos procesos son fundamentalmente aleatorios, lo que significa que los números que producen son verdaderamente aleatorios y no pueden ser predichos ni reproducidos.

Por otro lado, los **PRNGs** son algoritmos matemáticos que generan secuencias de números que parecen aleatorias. Sin embargo, dado que los algoritmos son deterministas, si conoces el estado inicial del algoritmo, puedes predecir todos los números que generará. Esto significa que los números que producen los PRNGs solo son pseudoaleatorios, no verdaderamente aleatorios.

Finalmente, los **CSPRNGs** son una clase especial de PRNGs que han sido diseñados para ser utilizados en criptografía. Los CSPRNGs tienen la propiedad de que, incluso si conoces todos los números que ha generado hasta ahora, no puedes predecir cuál será el próximo número. Esto los hace adecuados para tareas como generar claves criptográficas, donde la seguridad depende de que los números generados sean impredecibles.

3.2.3 Curva Elíptica

Antes de continuar con los principales algoritmos de encriptación asimétrica, hemos de dar contexto sobre dos conceptos fundamentales, el primero y del cual hablaremos en este apartado es la curva elíptica [13].

Una curva elíptica es una herramienta matemática que se utiliza para crear sistemas de cifrado muy seguros. Aunque el nombre de "curva" puede sugerir una forma ovalada, las curvas elípticas en realidad tienen una forma más compleja y pueden parecer más una serie de ondas que una elipse. Un ejemplo de esta curva se puede ver en la figura 13:

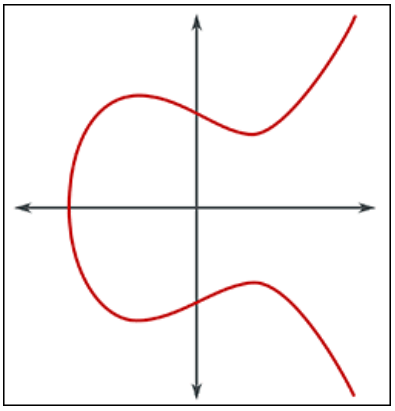

Figure 13: Ejemplo de curva elíptica

La criptografía de curva elíptica se basa en la dificultad de resolver ciertos problemas matemáticos que involucran estas curvas. Uno de estos problemas es el llamado problema del logaritmo discreto, que es fácil de realizar en una dirección, pero extremadamente difícil de invertir. Esto significa que, si tienes dos puntos en la curva puedes sumarlos fácilmente para obtener un tercer punto, pero si solo tienes el resultado, es muy difícil determinar cuáles eran los dos puntos originales.

Este tipo de operaciones unidireccionales son la base de muchos sistemas criptográficos, ya que permiten crear claves que pueden ser utilizadas para cifrar un mensaje de manera que solo puede ser descifrado con la clave correspondiente. En el caso de la criptografía de curva elíptica, las claves se generan a partir de puntos en la curva, lo que proporciona un alto nivel de seguridad.

Una de las grandes ventajas de las curvas elípticas es que prometen una seguridad más fuerte y mejor rendimiento con claves más cortas, necesitando menos potencia informática.

La fórmula del caso general de la curva elíptica es: $y^2 = x^3 + ax + b$

3.2.4 Firma Digital

Una firma digital es una técnica que se utiliza en la criptografía de clave asimétrica para verificar la autenticidad de un mensaje o documento, es como una firma manuscrita en el mundo físico, pero para documentos digitales. Cuando alguien firma digitalmente un documento, está utilizando su clave privada para crear una especie de sello único que está vinculado tanto al documento como a la persona que lo firma. Este sello es la firma digital.

La ventaja de la firma digital es que no puede ser falsificada. Si alguien intenta alterar el documento después de que ha sido firmado, la firma ya no será válida. Además, la firma digital no puede ser reutilizada o transferida a otro documento.

Para verificar una firma digital, se utiliza la clave pública del firmante. Si la firma es válida, entonces sabemos dos cosas: que el documento no ha sido alterado desde que fue firmado y que fue firmado por el propietario de la clave privada correspondiente a la clave pública utilizada para verificar la firma.

En resumen y para poder quedarse con una idea, se puede decir que las firmas digitales proporcionan una forma de garantizar la integridad y la autenticidad de los documentos digitales, es equivalente a la firma física de un documento [14].

3.2.5 Principales algoritmos de encriptación Asimétrica

A continuación se explicará cuáles son los principales algoritmos de encriptación asimétrica, desde los algoritmos relevantes más antiguos que se crearon y que fueron los promotores de los que hoy en día se utilizan, hasta los más seguros y modernos.

3.2.5.1 Diffie-Hellman. Algoritmo de compartición de claves.

El algoritmo de Diffie-Hellman es un método en la criptografía de clave asimétrica que permite a dos partes, cada una con su par de claves pública y privada, establecer un secreto compartido a través de un canal de comunicación no seguro.

Este algoritmo se basa en la dificultad de resolver el problema del logaritmo discreto en un campo finito, que es un problema matemático considerado difícil de resolver. En el algoritmo de Diffie-Hellman, cada parte genera un valor secreto y lo combina con un valor público acordado para generar una clave pública, luego intercambian estas claves públicas entre sí y una vez que cada parte ha recibido la clave pública de la otra, la combinan con su propio valor secreto para generar el secreto compartido

Para que se entienda mejor, vamos a explicar el proceso paso a paso:

Todo comienza con ambas partes acordando un número público que todos pueden ver. Luego, cada una elige un número secreto que no comparte con nadie. Después, cada una realiza una operación matemática con su número secreto y el número público, generando un nuevo número. Estos nuevos números se intercambian entre las dos partes. Cuando cada parte recibe el nuevo número de la otra, realiza otra operación matemática con ese número y su propio número secreto. El resultado de esta última operación es el mismo para ambas partes y ese es su "secreto compartido".

La ventaja del algoritmo Diffie-Hellman es que, aunque alguien vea los números públicos y los números intercambiados, no puede calcular el secreto compartido a menos que conozca los números secretos originales, por lo que esto garantiza seguridad.

Por otro lado, la desventaja es que este algoritmo es vulnerable al ataque *Man-in-the-Middle* [15]. Sin entrar mucho en detalle, ya que de esto se hablará luego en otra sección, se puede decir que este ataque funciona de forma que el atacante se sitúa entre las dos máquinas que se quieren intercambiar información y genera una clave con cada máquina, suplantando la identidad de la otra para interceptar y manipular las comunicaciones. La figura 14 ilustra la idea de este ataque mencionado:

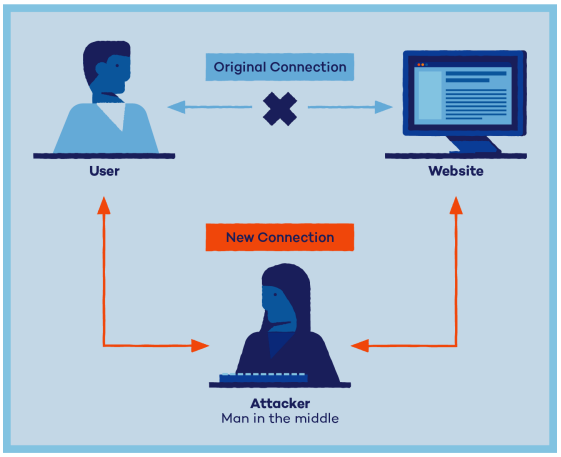

Figure 14: Ejemplo visual del ataque *Man-in-the-Middle*

3.2.5.2 Rivest Shamir Adleman (RSA)

Este algoritmo tiene una relevancia y aplicaciones significativas hoy en día.

En términos de relevancia, RSA es fundamental para mantener la seguridad de la información digital. La mayor parte de los sitios hoy en día corren sobre SSL/TLS[5] y

[5] SSL/TLS: Son protocolos de seguridad que proporcionan comunicaciones seguras en la red mediante el cifrado de los datos transmitidos entre dos sistemas.

permiten la autenticación mediante cifrado asimétrico basado en RSA. Actualmente, estas conexiones son cruciales para actividades como las transacciones bancarias en línea y el comercio electrónico.

Otra de las utilidades de RSA es el poder comprobar la autenticidad y la integridad de los documentos, se encarga de asegurarse que no han sido alterados desde su firma y que provienen del firmante correcto. Finalmente, RSA también se utiliza en la autenticación de usuarios en sistemas informáticos y redes, proporcionando una forma segura de verificar la identidad de los usuarios, protegiendo así los sistemas contra el acceso no autorizado.

Según se ha indicado, RSA tiene muchos usos hoy en día y se considera muy seguro y esto es debido a que la seguridad de RSA radica en que, aunque el proceso de cifrado es relativamente sencillo, el proceso inverso sin la clave privada es extremadamente difícil y consume mucho tiempo, incluso para los ordenadores más potentes.

La dificultad de desencriptar RSA sin la clave privada correcta radica en un problema matemático conocido como factorización de números grandes.
RSA se basa en el uso de dos números primos grandes para generar las claves pública y privada.

Si un intruso intenta descifrar un mensaje codificado con RSA pero sólo tiene acceso a la clave de acceso público, tendría que descomponer un valor específico en sus dos factores primos originales para calcular un valor crítico. Este proceso de descomposición es extremadamente arduo y requiere mucho tiempo, incluso para las computadoras más avanzadas, cuando el valor es lo suficientemente grande. Por lo tanto, a menos que el intruso tenga la clave privada, descifrar un mensaje codificado con RSA es prácticamente imposible. Esto es lo que hace que RSA sea tan seguro y resistente a la descodificación [16].

3.2.5.3 Digital Signature Algorithm (DSA)

Este algoritmo, al igual que RSA, también es un pilar fundamental en la criptografía asimétrica y se utiliza muy comúnmente en el día a día.

En términos de relevancia, la función principal y fundamental de DSA es para la creación de firmas digitales en documentos y software. Al igual que he explicado en el algoritmo RSA, esto permite verificar la autenticidad y la integridad de los documentos, asegurando que no han sido alterados desde su firma y que provienen del firmante correcto.

La seguridad de DSA radica en la dificultad de resolver el problema del logaritmo discreto sin la clave privada correcta. DSA se basa en el uso de un número primo grande y un generador de un subgrupo de orden primo para generar las claves pública y privada.

La clave pública, que se calcula con la siguiente fórmula: $y = g^x (mod\, p)$. Consta de un número *p* (un número primo grande), un número *g* (un generador de un subgrupo de orden primo) y un número *y* (g elevado a la clave privada x, módulo p).

La clave privada, que se utiliza para firmar los datos, es simplemente un número x elegido al azar del subgrupo de orden primo. Esta clave privada se mantiene en secreto.

Si un atacante quiere falsificar una firma DSA, pero sólo tiene la clave pública, tendría que resolver el problema del logaritmo discreto para calcular la clave privada x. Este problema es extremadamente difícil y consume mucho tiempo, incluso para los ordenadores más potentes, cuando p es un número suficientemente grande [17].

La diferencia principal del uso de las matemáticas entre los algoritmos RSA y DSA es que el primero se basa en la factorización de números primos grandes, mientras que el segundo se basa en el problema del logaritmo discreto. También es importante mencionar la diferencia a nivel de funcionalidad, ya que antes cuando se ha hablado del algoritmo RSA, se ha mencionado que uno de sus usos también podía ser en las firmas digitales. La diferencia es que RSA puede ser utilizado tanto para cifrado como para firmas digitales, mientras que DSA fue diseñado específicamente para firmas digitales [18].

3.2.5.4 Algoritmo de firma digital de curva elíptica (ECDSA)

Antes hemos estado hablando sobre lo que es la curva elíptica y cómo funciona, concretamente hemos hablado de por qué se utiliza en criptografía y ahora se explicará un algoritmo que se basa en este concepto explicado.

El algoritmo de firma digital de curva elíptica (ECDSA) es un sistema de criptografía asimétrica que genera un par de claves, una clave privada y una clave pública, que están vinculadas a través de una operación matemática compleja realizada sobre una función de curva elíptica.

En la generación de claves, se selecciona un número aleatorio como clave privada y luego se realiza una operación de multiplicación de puntos en la curva elíptica para generar la clave pública correspondiente. Para firmar un conjunto de datos, se genera un número aleatorio único y luego se realizan varias operaciones matemáticas, incluyendo multiplicaciones y divisiones en el campo finito de la curva elíptica, para generar dos valores que constituyen la firma. La verificación de la firma implica realizar varias operaciones matemáticas en la curva elíptica utilizando la clave pública, los datos y la firma. Si las operaciones resultan en el punto correcto en la curva, la firma es válida.

Estas operaciones garantizan que cada par de claves generadas resulte en firmas únicas e irrepetibles. Además, es prácticamente imposible falsificar las firmas

digitales, ya que la potencia computacional necesaria para hacerlo supera los límites actuales. Por lo tanto, el ECDSA proporciona una forma segura de verificar la autenticidad e integridad de los datos transmitidos [19]. Sin embargo, esta explicación que se acaba de hacer es una simplificación de las operaciones matemáticas reales que se realizan, las cuales son bastante complejas y se salen un poco de los límites de este trabajo de final de grado. Esto es debido a que además de requerir un entendimiento profundo de la criptografía, también se necesitan grandes conocimientos matemáticos para poder entender el proceso exacto que se sigue.

3.2.5.5 Edwards-curve Digital Signature Algorithm (EdDSA)

El algoritmo Edwards-curve Digital Signature es bastante parecido al anterior, ya que también se basa en la curva elíptica, concretamente se basa en curvas elípticas optimizadas para el rendimiento. Este algoritmo, al igual que el anterior, se basa en proporcionar una forma segura de firmar digitalmente documentos o datos, asegurando su autenticidad e integridad.

En este algoritmo se utilizan varias operaciones matemáticas complejas realizadas sobre una curva elíptica relacionadas con la suma de puntos y la multiplicación escalar en la curva de Edwards, en cambio, ECDSA utiliza una operación conocida como multiplicación de puntos en la curva elíptica.

Lo que nos proporciona el algoritmo EdDSA es una forma segura de verificar la autenticidad de los datos transmitidos. Cada par de claves generadas resulta en firmas únicas e irrepetibles [20].

3.2.5.6 Diferencias entre EdDSA y ECDSA

Hay cuatro grandes diferencias entre estos dos algoritmos a parte de la mencionada antes sobre las operaciones que realizan.

La primera gran diferencia es el tipo de curva, es decir, mientras que ECDSA puede operar sobre cualquier tipo de curva elíptica, EdDSA opera específicamente sobre las curvas de Edwards, que tienen propiedades que permiten implementaciones más rápidas y seguras de las operaciones de la curva elíptica.

Otra gran diferencia es respecto al proceso de generación de firmas. El proceso de generación de firmas es completamente distinto. ECDSA genera una firma única criptográficamente comprobable, pero EdDSA genera firmas distintas para cada documento e igualmente comprobables.

La tercera diferencia es respecto a la seguridad y rendimiento. EdDSA generalmente se considera más seguro y eficiente que ECDSA y esto se debe a algo que se acaba de mencionar hace un momento, que es el hecho de que las

curvas de Edwards permiten implementaciones más rápidas y seguras de las operaciones de la curva elíptica.

Y, por último, pero no menos importante, la última gran diferencia es el tamaño de las claves. Ambos algoritmos pueden lograr el mismo nivel de seguridad que RSA con claves significativamente más pequeñas debido a la complejidad computacional del problema del logaritmo discreto [21].

3.3 Public Key Infrastructure (PKI)

Antes de comenzar con la explicación de lo qué es PKI, es esencial entender qué es tanto un certificado digital cómo una Autoridad de Certificación, por lo que primero vamos a introducir estos conceptos.

3.3.1 Certificados Digitales

Un **certificado digital**, también conocido como **certificado electrónico**, es un archivo informático que se utiliza para identificar a una entidad (como una persona o una página web) en el mundo digital. Esto lo consigue asociando una clave pública a una entidad (datos que representan la identidad) y esta entidad es la que posee la clave privada asociada. Esta asociación la realiza una **Autoridad de Certificación**, que es una entidad confiable que verifica la identidad de la entidad que posee el certificado, de esta forma cuando se use la clave privada se puede asegurar la identidad del usuario [22].

La función principal de un certificado digital es proporcionar una forma segura de verificar la identidad de una entidad en las comunicaciones digitales. Esto es especialmente útil en las transacciones en línea, donde es importante asegurar que los datos se están enviando a la entidad correcta y que los datos no han sido alterados durante su transmisión. Además de servir como autenticación, también puede tener otros propósitos como: cifrado de datos, firma digital, etc [23].

En la figura 15 se puede apreciar un ejemplo visual del concepto teórico del funcionamiento de una Autoridad de Certificación:

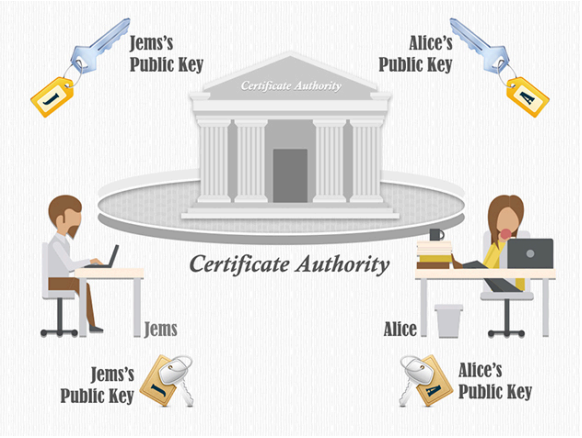

Figure 15: Ejemplo visual del funcionamiento de una Autoridad de Certificación

3.3.1.1 Ciclo de vida de un certificado

El ciclo de vida de un certificado se divide en cuatro fases: solicitud, renovación/revocación, emisión y uso.

En la fase de solicitud del certificado, el sujeto solicita el certificado a través de un CSR[6] (Certificate Signing Request) en el que incluye sus datos. Esta fase se hace automáticamente a través de la herramienta de administración de certificados (certlm).

La segunda fase es en la que se emite el certificado. Cuando el CSR llega, se verifica la identidad del usuario y se emite su certificado. Se pueden desplegar certificados de manera automática a través de GPOs [7] sin pasar por la solicitud.

La siguiente fase es la de uso del certificado y esta fase dura mientras el certificado está dentro de su periodo de validez y no se haya revocado.

La última fase de todas es la de Renovación/Revocación del certificado, esto ocurre cuando acaba el periodo de validez de un certificado. Cuando sucede esto, si es necesario renovarlo, entonces se puede habilitar automáticamente a través de GPO o realizar otro CSR. En caso de que un certificado esté corrupto o no se use, se puede revocar desde la CA[8] convirtiendo este certificado en inválido [24].

3.3.1.2 Revocación de certificado

[6] CSR: Es una solicitud generada por un usuario o una organización para una entidad de certificación, que contiene la clave pública y la información de identidad, utilizada para solicitar un certificado digital SSL/TLS.

[7] GPO: Una GPO (Group Policy Object) es un conjunto de reglas que permiten a los administradores de sistemas gestionar y configurar centralizadamente las cuentas de usuario y los equipos en una red de dominio.

[8] CA: Certification Authority (CA) es una entidad confiable que emite y gestiona certificados digitales para garantizar la autenticidad, confidencialidad e integridad de la comunicación en línea.

Esta fase que antes hemos introducido es en realidad más compleja de lo que puede parecer. Con la renovación de certificados se añade una capa adicional de complejidad al verificar la validez de los certificados: hay que comprobar que no se haya caducado y que no se haya revocado. Para esto existen 2 mecanismos: CRL y OCSP.

CRL (Certificate Revocation Lists), son listas que contienen los números de serie de los certificados digitales y su estado (revocado, válido o caducado). Estas listas están firmadas por la CA para dar validez a la propia lista. Antes de la llegada de OCSP, cada usuario tenía que descargar la CRL y mantenerla actualizada por su propia cuenta, mientras que ahora ya no hace falta.

De un modo más sencillo de explicar, se podría decir que es como un tablón de anuncios público donde se publican los certificados que ya no son válidos o de confianza. Cuando alguien quiere verificar un certificado, puede consultar este tablón para ver si el certificado está en la lista de revocados.

Por otro lado, está el OCSP (Online Certificate Status Protocol), que es un mecanismo que permite validar el estado de un certificado con una consulta a la CA. Se podría decir que es un protocolo moderno que permite hacer consultas a un OCSP, responder (que se encarga de contactar con la CRL) y devolver el estado del certificado en tiempo real.

De una forma más sencilla de explicar, se podría decir que es como un servicio de atención al cliente que responde a las consultas sobre el estado de los certificados. En lugar de buscar en una lista completa como en CRL, puedes preguntar directamente al servicio OCSP sobre un certificado específico y te dirá si es válido o no [25].

3.3.2 ¿Qué es PKI?

La PKI es un marco de seguridad que utiliza criptografía de clave pública para asegurar las comunicaciones y las transacciones digitales. Se compone de dos componentes principales: un certificado digital y una Autoridad de Certificación (CA).

Dicho de otra forma, PKI es el framework[9] subyacente (compuesto por políticas, funciones, procedimientos, hardware y software) que permite a las entidades (usuarios y equipos) intercambiar información utilizando certificados digitales. El funcionamiento de la PKI está basado en relaciones de confianza. Como se ha mencionado antes, está formada por una red de Autoridades de Certificación (CAs) que viene definida por su “jerarquía”. La jerarquía dentro de esta red se refiere a la estructura de confianza que se establece entre las CAs, donde algunas CAs pueden ser superiores (es decir, más confiables) que otras.

[9] Framework: Es un conjunto estructurado de librerías y herramientas de software que proporciona una base sobre la cual los desarrolladores pueden construir programas de manera más eficiente.

En lo alto de esta jerarquía se encuentra la **Root CA** (certificado auto-firmado). Desde la Root CA, la jerarquía se estructura en niveles donde cada sub-CA tiene un certificado firmado por su CA padre, esto se hace con el objetivo de establecer una cadena de confianza. Cuando una Autoridad de Certificación padre firma el certificado de una sub-CA, está esencialmente validando la identidad de la sub-CA y estableciendo confianza en ella. Esto significa que cualquier entidad que confíe en la CA padre también puede confiar en la sub-CA [26].

Se puede apreciar en la figura 16 cómo está estructurada la jerarquía de una CA:

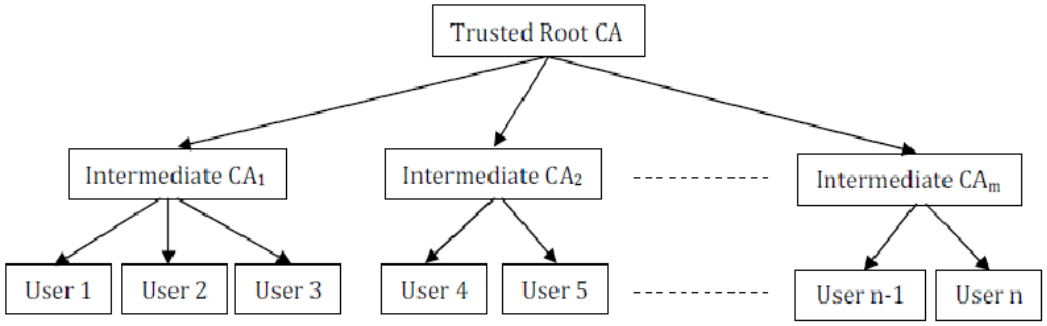

Figure 16: Estructura de la jerarquía de una CA

3.3.3 Tipos de PKIs

Existen tres tipos de PKIs, cada una con su propio conjunto de características y usos. Estas son la PKI Pública, la PKI Interna o Privada y la PKI Híbrida. La PKI Pública se utiliza en el ámbito público de Internet, mientras que la PKI Interna se utiliza dentro de una organización o empresa. Por otro lado, la PKI Híbrida combina elementos de las PKIs públicas e internas, aunque todas las PKIs trabajan para proporcionar un entorno seguro para las comunicaciones y transacciones digitales, se eligen en función de las necesidades de seguridad y las capacidades de la organización. A continuación, se detallará más sobre cada tipo de PKI y cómo funcionan.

PKI Pública: En este tipo de PKI, la criptografía de clave pública se utiliza para asegurar las comunicaciones en Internet. Las Autoridades de Certificación (CAs) emiten certificados digitales para sitios web, servidores de correo electrónico y otros servicios en línea. Cuando un usuario intenta conectarse a uno de estos servicios, el servidor presenta su certificado digital. El navegador del usuario verifica la firma digital en el certificado utilizando la clave pública de la CA que emitió el certificado. Si la verificación es exitosa, el navegador establece una conexión segura con el servidor. Los datos enviados a través de esta conexión se cifran con la clave pública del servidor y sólo pueden ser descifrados con la clave privada correspondiente del servidor [27].

PKI Interna o Privada: En una PKI interna, la criptografía de clave pública se utiliza para asegurar las comunicaciones dentro de una organización. La organización tiene su propia CA que emite certificados para sus propios servidores y usuarios. Cuando un usuario interno intenta conectarse a uno de estos servidores, el servidor presenta su certificado digital. El sistema del usuario verifica la firma digital en el certificado utilizando la clave pública de la CA interna. Si la verificación es exitosa, el sistema establece una conexión segura con el servidor. Los datos enviados a través de esta conexión se cifran con la clave pública del servidor y sólo pueden ser descifrados con la clave privada correspondiente del servidor [28].

PKI Híbrida: En una PKI híbrida, se utilizan elementos de las PKIs públicas e internas. La organización tiene su propia CA interna que emite certificados para las comunicaciones internas y también utiliza certificados de una CA pública para los servicios que se ofrecen al público a través de Internet. Cuando un usuario (ya sea interno o externo) intenta conectarse a uno de estos servicios, el servidor presenta el certificado digital correspondiente. El sistema del usuario verifica la firma digital en el certificado utilizando la clave pública de la CA correspondiente (ya sea la CA interna o la CA pública). Si la verificación es exitosa, el sistema establece una conexión segura con el servidor. Los datos enviados a través de esta conexión se cifran con la clave pública del servidor y sólo pueden ser descifrados con la clave privada correspondiente del servidor.

3.4 Criptoanálisis

3.4.1 ¿En qué consiste el criptoanálisis?

El criptoanálisis es una disciplina que se encarga de estudiar los sistemas criptográficos, con el objetivo de descifrar o entender los mensajes cifrados sin tener acceso a la información secreta que normalmente se necesitaría. En otras palabras, el criptoanálisis intenta “romper” los códigos o sistemas de cifrado para acceder a la información oculta. Se encargan de someter a pruebas a los algoritmos criptográficos para encontrar fallos de seguridad en ellos que permitan recuperar el contenido de un texto previamente cifrado.

El criptoanálisis se puede dividir en dos tipos principales: criptoanálisis de clave secreta y criptoanálisis de clave pública.

El criptoanálisis de clave secreta se aplica a los sistemas de cifrado simétrico, donde la misma clave se utiliza para cifrar y descifrar el mensaje. Los métodos comunes incluyen el análisis de frecuencia, donde se estudia la frecuencia de aparición de ciertos caracteres o grupos de caracteres y el ataque por fuerza bruta, que implica probar todas las posibles combinaciones de claves hasta encontrar la correcta.

Por otro lado, el criptoanálisis de clave pública se aplica a los sistemas de cifrado asimétrico, donde se utilizan claves diferentes para cifrar y descifrar el mensaje. Los métodos comunes incluyen el ataque de factorización, que intenta factorizar el producto de dos números primos grandes, y el ataque de logaritmo discreto, que intenta resolver el problema del logaritmo discreto.

Es importante destacar que un buen sistema criptográfico debe ser resistente a todos los tipos de ataques de criptoanálisis. La seguridad de un sistema criptográfico se mide a menudo por el tiempo y los recursos que se necesitarían para romperlo. En la práctica, un sistema se considera seguro si el tiempo requerido para romperlo excede la vida útil de la información que protege.

Además, el criptoanálisis no solo se utiliza para “romper” los códigos, sino también para probar la seguridad de los sistemas criptográficos. Antes de que se adopte un nuevo algoritmo criptográfico, se somete a un riguroso análisis para asegurarse de que es seguro contra los ataques conocidos. [29]

3.4.2 Ataques

En este apartado, vamos a explorar diferentes tipos de ataques de criptoanálisis: análisis de texto plano conocido, análisis de texto plano elegido, análisis de sólo texto cifrado, ataque de fuerza bruta, ataque man in the middle y análisis de texto plano elegido adaptativo. Cada uno de estos ataques tiene su propio enfoque y eficacia para comprometer la seguridad de un sistema criptográfico. A continuación, profundizaremos en cada uno de estos ataques.

Known-plaintext analysis (Análisis de texto plano conocido)**:** Este tipo de ataque ocurre cuando un atacante tiene acceso a un texto cifrado y a su correspondiente texto plano. El objetivo es descubrir la clave utilizada para el cifrado.

Chosen-plaintext analysis (Análisis de texto plano elegido)**:** En este caso el atacante puede elegir un texto plano y obtener su correspondiente texto cifrado. El objetivo es descubrir la clave de cifrado.

Ciphertext-only analysis (Análisis de sólo texto cifrado)**:** Este es el tipo de ataque más común y también el más difícil. El atacante sólo tiene acceso al texto cifrado y su objetivo es descifrarlo sin conocer la clave.

Brute force attack (Ataque de fuerza bruta)**:** Este ataque implica probar todas las posibles combinaciones de claves hasta encontrar la correcta. Es efectivo, pero requiere una gran cantidad de tiempo y recursos computacionales.

Man-in-the-middle attack (Ataque man in the middle)**:** En este ataque, el atacante se coloca entre el emisor y el receptor del mensaje cifrado, interceptando y posiblemente alterando la comunicación.

Adaptive chosen-plaintext analysis (Análisis de texto plano elegido adaptativo)**:** Este es un tipo de ataque chosen-plaintext en el que el atacante puede realizar sucesivas consultas de textos planos elegidos basándose en los resultados de las consultas anteriores.

Cada uno de estos ataques descritos tiene diferentes niveles de éxito dependiendo de la fortaleza del sistema criptográfico y de la cantidad de información a la que tenga acceso el atacante. Este es uno de los principales motivos por lo que es tan importante diseñar sistemas criptográficos robustos y mantener las claves de cifrado seguras [30] .

4 Otras aplicaciones de la criptografía

4.1 Algoritmo de cifrado IEEE 802.11 (WiFi)

El algoritmo IEEE 802.11 es un tipo de cifrado de redes LAN inalámbricas. Se utiliza para añadir seguridad, mediante un protocolo de autenticación, que solicita una contraseña o clave de red cuando un usuario o dispositivo intenta conectarse.

Este algoritmo ha pasado por una evolución constante, ya que el algoritmo original se quedó obsoleto e inservible por lo que se fue evolucionando a versiones posteriores hasta la final que se usa hoy en día.

Los diferentes tipos de cifrado para redes LAN son los siguientes: WEP (1999), WPA (2003), WPA2 (2004) y WPA3 (2018).

WEP (Wired Equivalent Privacy, 1999) fue la versión original y el primer intento de proporcionar seguridad a las redes 'Wi-Fi Sin embargo, tenía varias debilidades, incluyendo el uso de claves de cifrado estáticas, lo que lo hacía vulnerable a los ataques. Además, los paquetes de datos podían ser interceptados y descifrados por atacantes con suficiente tiempo y recursos. El tipo de cifrado que usa este algoritmo es RC4, y es un cifrado de flujo.

WPA (Wi-Fi Protected Access, 2003) surge para corregir las debilidades de WEP. Utilizaba un protocolo llamado TKIP (Protocolo de Integridad de Clave Temporal), que generaba una nueva clave para cada paquete de datos, lo que lo hacía más seguro que WEP. Sin embargo, TKIP todavía tenía algunas vulnerabilidades y no era tan seguro como podría ser. El tipo de cifrado que usa este algoritmo es RC4 + TKIP, con IVs[10] más largos y claves de 128 bits.

WPA2 (Wi-Fi Protected Access, 2004) surge para mejorar aún más la seguridad. Reemplazó TKIP con AES (Advanced Encryption Standard), un algoritmo de cifrado mucho más seguro. Además, WPA2 introdujo un proceso de autenticación mejorado y un “handshake[11]” de 4 vías para establecer una conexión segura. El tipo de cifrado que usa este algoritmo es CCMP[12] + AES, con claves de 128 bits y es un cifrado de bloques. La foto de la derecha refleja el funcionamiento de un “handshake” de 4 vías, donde el AP (Access Point) intercambia información con el cliente para asegurar la conexión.

[10] IV: IV (Vector de Inicialización), es un bloque de bits que se utiliza para permitir un cifrado en flujo o un cifrado por bloques, con un resultado independiente de otros cifrados producidos por la misma clave.

[11] Handhsake: Es un proceso de autenticación en el que dos dispositivos establecen una conexión segura intercambiando información de autenticación y claves de cifrado

[12] CCMP: Es un protocolo criptográfico utilizado para proteger la comunicación inalámbrica en redes Wi-Fi, garantizando la privacidad y la autenticidad de las comunicaciones.

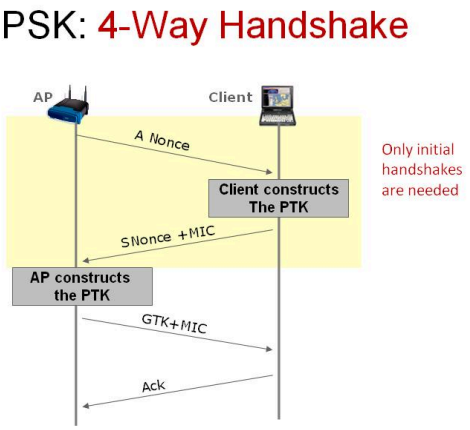

Figure 17: Ejemplo visual de un *handshake* de 4 vías

Finalmente llegamos a WPA3 (Wi-Fi Protected Access, 2018), es el protocolo más reciente y ofrece aún más mejoras de seguridad. Proporciona una protección más fuerte contra ataques de fuerza bruta y mejora la seguridad en redes públicas. También introduce un nuevo protocolo de handshake llamado Simultaneous Authentication of Equals (SAE), que es más seguro que el handshake de 4 vías utilizado en WPA2. El tipo de cifrado que usa este algoritmo es AES, con claves de 192 bits (WPA3-Enterprise) y es un cifrado de bloques. El único problema que tiene este algoritmo es que no es compatible con algunos sistemas [31].

4.2 Seguridad en la capa de transporte (SSL/TLS)

La Seguridad de la Capa de Transporte (TLS) y su antecesor Secure Sockets Layer (SSL) son protocolos criptográficos que proporcionan comunicaciones seguras en una red, comúnmente Internet. Estos protocolos utilizan la criptografía para asegurar la autenticación, la integridad de los datos y la confidencialidad.

En un sistema criptográfico asimétrico, se utilizan un par de llaves para lograr la autenticación. Estas llaves son asimétricas, una es tu llave pública, que es pública y la otra es tu llave privada. Durante el proceso de “handshake” o saludo inicial, el cliente y el servidor intercambian información adicional para establecer un secreto compartido. Un secreto compartido es una clave o dato que se mantiene en privado entre dos o más partes y se utiliza para cifrar y descifrar la información transmitida entre ellas, garantizando así la seguridad y la confidencialidad de la comunicación[32].

En la figura 17 se puede ver el proceso ilustrativo de cómo funciona un secreto compartido. Frecuentemente, este proceso se mueve del intercambio de criptografía asimétrica a criptografía simétrica.

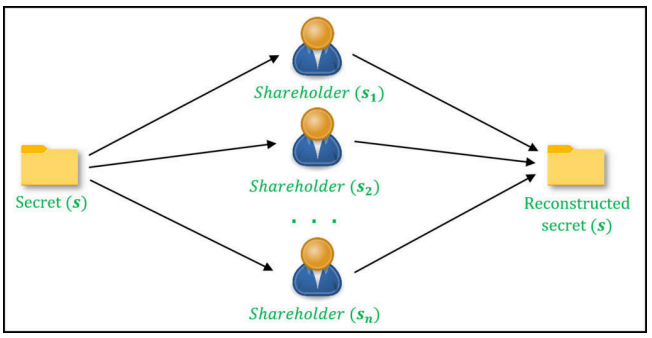

Figure 18: Ejemplo visual de un secreto compartido

Una vez que se ha establecido un secreto compartido, se utiliza la criptografía simétrica para cifrar los datos transmitidos entre el cliente y el servidor. Esto asegura que los datos no puedan ser leídos por terceros, incluso si logran interceptar la comunicación. Además, TLS y SSL también utilizan funciones de hash para garantizar la integridad de los datos. Esto significa que, si los datos son alterados durante la transmisión, el receptor será capaz de detectarlo.

Al cifrar los datos, TLS y SSL aseguran que la información transmitida entre el cliente y el servidor se mantenga confidencial. La figura 19 ilustra el proceso que sigue el protocolo SSL [33].

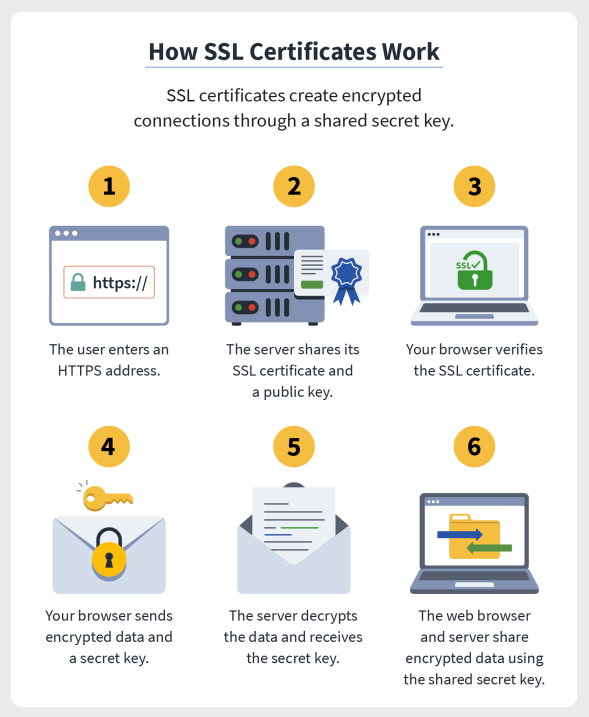

Figure 19: Ejemplo del proceso que sigue un certificado SSL

4.3 Virtual Private Network (VPN)

La explicación sencilla es que una VPN o Red Privada Virtual, es algo parecido a un túnel seguro en Internet. Cuando te conectas a una VPN, tu información se cifra y se envía a través de este túnel seguro, lo que protege tus datos de otras personas que podrían intentar verlos.

Técnicamente hablando, una VPN extiende una red privada a través de una red pública, permitiendo a los usuarios enviar y recibir datos a través de redes públicas o compartidas, como si sus dispositivos estuvieran directamente conectados a la red privada (remote access, site-to-site) [34].

Los dos tipos de VPN se basan en diferentes protocolos: OpenVPN, IPSec/L2TP, SSTP, PPTP y IKEv2.

OpenVPN, es un protocolo de VPN de código abierto que utiliza técnicas de VPN que engloban cifrado, túneles de datos, protocolos de VPN y cambio de dirección IP. Sirve para asegurar conexiones punto a punto y de sitio a sitio.

IPSec/L2TP, está basada en protocolos de cifrado de IPSec. IPSec es un conjunto de protocolos que proporciona seguridad a las comunicaciones IP. L2TP es un protocolo de túnel que no tiene cifrado integrado, por lo que se usa junto con IPSec para proporcionar seguridad.

SSTP es un protocolo de cifrado de las comunicaciones a nivel de socket. Un socket es un punto de conexión de comunicaciones que permite la transmisión de datos entre dos programas, ya sea en la misma máquina o en máquinas diferentes en una red, es lo que se utiliza por ejemplo cuando se intenta conectar a un servidor web para cargar una página, o a un servidor de correo para enviar un email.

PPTP es un protocolo muy antiguo desarrollado por Microsoft, que actualmente se encuentra en desuso por envejecimiento respecto a los protocolos de VPN.

IKEv2 es un protocolo que fue desarrollado por Cisco y Microsoft. Se encarga de realizar un *handshake* al inicio de una sesión Ipsec. Dicho de otra forma, es un protocolo de cifrado de solicitud y respuesta que establece y maneja la Asociación de Seguridad (SA) para la comunicación segura entre dos entidades de red.

5 Trabajo práctico

En esta sección se va ha hablar sobre la parte práctica realizada para este trabajo de final de grado, correspondiente a una página web donde se puede introducir datos sensibles o privados para ver como sería el resultado en formato hash o encriptado.
Es importante mencionar que para poder acceder a mi página web se ha de entrar al siguiente enlace: https://encryptdata.netlify.app/ [36] .

5.1 Funcionalidades

Esta página web tiene un objetivo sencillo, ilustrar de forma visual cómo se ven los datos encriptados o hasheados con el algoritmo más seguro/eficiente posible. Además de esto, otro gran objetivo de esta página web es concienciar de la diferencia entre hashear y encriptar, y también concienciar de cuáles son los algoritmos más seguros para cumplir con este propósito.

- **Introducción y detección de datos** de entrada como DNI, contraseña, tarjeta de crédito, documentos y datos personales (nombre, apellido, correo electrónico, dirección de vivienda, código postal, número de teléfono y fecha de nacimiento). Cada uno de estos datos corresponde con una sección de la página web, al entrar dentro de una de estas secciones, te sale la opción de introducir el dato correspondiente y encriptarlo o hashearlo.

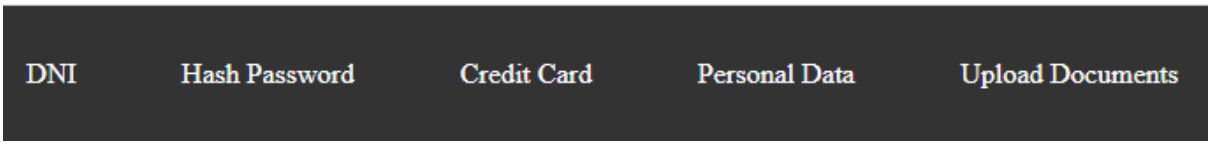

Figure 20: Secciones disponibles donde se introducen datos.

- **Subir un documento:** Esta funcionalidad permite al usuario poder subir un archivo de casi cualquier tipo de extensión para su procesamiento y encriptación posterior. En la sección de subir un documento, solo se permite encriptar el documento y no hashearlo, ya que en ningún caso tendría sentido hashear un documento, ya que entonces quiere decir que se volvería un documento indescifrable e ilegible, lo cual no tiene sentido ni tampoco relación con seguridad. La única opción que tiene sentido con un documento es poder encriptarlo. En la figura 21 se puede ver cómo se ve esta sección a la hora de introducir el documento.

Figure 21: Sección *Upload Documents*

- **Encriptación de datos:** Esta funcionalidad permite al usuario introducir y encriptar datos privados para luego ver el resultado por pantalla de cómo quedarían todos estos datos encriptados. En la figura 22 se puede ver cómo se ve una sección en la que se puede dar clic al botón “Encriptar” y seguidamente ver el resultado de esta encriptación:

Figure 22: Ejemplo encriptar datos en la sección *Personal Data*

- **Hashear datos:** Esta funcionalidad permite al usuario hashear los datos introducidos y ver el resultado por pantalla dándole clic al botón de “Hashear”. En la figura 23 se puede ver cómo se ve una sección en la que se puede dar clic al botón “Hashear” y seguidamente ver el resultado de este hash:

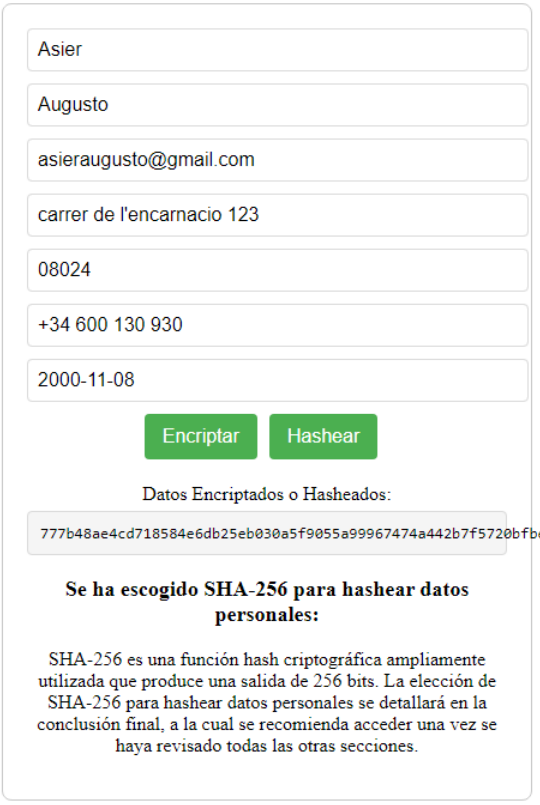

Figure 23: Ejemplo hashear datos en la sección *Personal Data*

- **Generación de clave segura en el "Backend":** Dentro del archivo python.py, el cual se encarga de hashear y encriptar los datos, también se generan las claves para poder encriptar estos datos. En caso de crear la clave y aplicar estos métodos de cifrado directamente desde el frontend, entonces la generación de la clave utilizada no sería segura. Esto es debido a que cualquier persona puede inspeccionar el código y acabar encontrando la clave que el frontend ha generado para encriptar los datos sensibles, esto obviamente supone un riesgo para la seguridad, por lo que gracias a este archivo python.py, el cual actúa como "backend", la generación de la clave deja de ser un problema.
- **Creación de una sección de conclusiones:** Se ha creado una sección donde se añaden las conclusiones extraídas tras haber navegado por la página web. En estas conclusiones se habla del algoritmo usado para hashear y encriptar datos, el cual es casi siempre el mismo debido a que es el mejor algoritmo para almacenar datos sensibles de forma segura y eficiente. En la figura 24 se puede apreciar como se ve la primera pantalla de la sección "Conclusión Final".

Figure 24: Sección “Conclusión Final”

5.2 Base de datos

En esta página web no ha hecho falta el uso de una base de datos, esto es debido a que la idea de esta parte práctica se basa en simples fines demostrativos. Estos fines incluyen saber cuál es el algoritmo más seguro de todos para encriptar o hashear datos, y otro gran objetivo es poder acabar de entender del todo como funciona y qué diferencia hay entre hashear y encriptar. Esto se puede lograr probando a encriptar datos y observar cuál es el resultado.

En cualquier caso, para poder lograr todo esto, no es necesario almacenar los datos, lo que nos interesa es ver el resultado y analizarlo, no almacenarlo.

6 Estructura del código y despliegue

Por un lado, en esta sección se va ha hablar de cómo está estructurado el frontend y el backend, de forma que se entienda un poco mejor el funcionamiento y estructura del código.

Por otro lado, también se explicará qué página web o herramienta se ha utilizado para desplegar tanto el frontend como el backend.

6.1 Frontend

En esta parte vamos a comenzar hablando sobre cómo se ha estructurado el código.

Por un lado decir que se ha creado un nuevo entorno de trabajo llamado ***frontend-env***, esto se ha hecho con el objetivo de incluir dentro las principales librerías que se han utilizado en el proyecto, con el fin de evitar problemas de dependencias entre ellas a la hora de desarrollar este.

Por otro lado, es importante mencionar que se ha utilizado **vue.js** para crear los componentes que forman esta página web.

En total hay 8 componentes que forman todas las secciones de la página web. Además de estos componentes, también se ha creado un archivo llamado **apiService.js**, el cual sirve para poder conectar el *frontend* con el *backend,* concretamente se encarga de manejar todas las llamadas a la API (Interfaz de Programación de Aplicaciones). Una vez terminada la explicación de cómo está estructurado el código pasaremos a hablar un poco más a fondo sobre el funcionamiento interno de este archivo mencionado.

El archivo **main.js** es el responsable de iniciar la aplicación y configurar cualquier funcionalidad global. Algunas de sus funcionalidades son: importaciones, creación de la aplicación, configuración de la aplicación, creación del router que nos permite enrutar las secciones disponibles, uso del router y montaje de la aplicación.

Otro archivo importante que tiene una relación indirecta con el recién mencionado es el archivo **routes.js**, se utiliza para definir las rutas de la aplicación. Cada ruta se mapea a un componente Vue, y cuando esa ruta se visita en el navegador, el componente correspondiente se renderiza.

Estos son los principales archivos utilizados en el *frontend*, ahora vamos a desarrollar un poco más el funcionamiento del archivo apiService.js.

En el contexto de mi página web, los métodos que están definidos en apiService.js hacen una solicitud HTTP al endpoint correspondiente en el servidor backend

(Python). Esta solicitud incluye los datos que pasamos como argumento y serán los que se procesen para luego devolver.

Una vez que la solicitud llega al servidor backend, se invoca el método correspondiente en el archivo app.py. Este método toma los datos recibidos, los encripta utilizando el algoritmo seleccionado y luego devuelve los datos encriptados o hasheados.

Finalmente los datos resultantes se envían de vuelta al frontend como respuesta a la solicitud HTTP inicial. Esta respuesta se recibe en el método correspondiente en apiService.js, y luego se maneja según sea necesario en el componente Vue.js.

6.2 Despliegue frontend

Netlify ofrece un plan gratuito que nos ha permitido desplegar el frontend de nuestra aplicación sin incurrir en ningún gasto. Este plan gratuito es más que suficiente para las necesidades de nuestro proyecto, lo que nos ha permitido utilizar esta plataforma sin coste alguno.

El despliegue de la aplicación en Netlify ha sido un proceso bastante sencillo. Con solo unos cuantos pasos, hemos podido conectar nuestro repositorio de código y configurar el proceso de construcción y despliegue. Además, Netlify se encarga automáticamente de la actualización de la aplicación cada vez que se realiza un cambio en el repositorio de código, lo que ayuda a ahorrar mucho tiempo y esfuerzo.

En resumen, Netlify nos ha proporcionado una forma eficiente y económica de desplegar el frontend de nuestra aplicación. El coste total ha sido de 0 euros, y la facilidad de implementación ha sido una gran ventaja.

6.3 Backend

El *backend* que he desarrollado para mi trabajo es relativamente simple, sobre todo en comparación con el *frontend*. Este *backend* está formado por un único archivo llamado **app.py**, en este archivo están implementados todos los métodos que se encargan tanto de encriptar como de hashear datos sensibles.

La idea de desarrollar un *backend* para esta aplicación es con el fin de securizar el proceso de proteger los datos, ya que como he explicado antes, si no tuviéramos un *backend*, una persona podría descubrir la clave creada por el *frontend* para encriptar los datos, lo cual no sería muy práctico ni seguro.

6.4 Despliegue backend

Netlify nos ha proporcionado un plan gratuito, el cual nos ha permitido poder usar esta plataforma sin incurrir en ningún gasto para la realización de este proyecto. Aunque Netlify es más conocido por alojar y desplegar aplicaciones frontend, también ofrece funcionalidades para el despliegue del backend a través de las funciones de Netlify, lo que nos ha permitido desplegar tanto la aplicación de gestión y visualización de archivos como la de cálculo de contornos.

El uso de Netlify para el despliegue del backend de nuestra aplicación ha sido una elección estratégica que nos ha permitido mantener todo nuestro stack de desarrollo en una sola plataforma, simplificando así el proceso de despliegue y mantenimiento. Además, la facilidad de uso y la integración con otros servicios, como los repositorios de código en GitHub, han hecho de Netlify una opción ideal para nuestro proyecto.

7 Conclusión

Recapitulando lo desarrollado en este proyecto y teniendo en cuenta los objetivos del mismo se puede considerar que se han conseguido cumplirlos.

Para empezar se ha explorado la vasta y amplia disciplina de la criptografía, desde sus orígenes hasta sus aplicaciones más modernas y avanzadas. Hemos visto cómo la criptografía ha evolucionado a lo largo de los años, adaptándose a las necesidades cambiantes de la sociedad y a los avances tecnológicos.

Hemos profundizado en los fundamentos de la criptografía simétrica y asimétrica, explorando desde los principios de Kerckhoffs y cómo han influido en el desarrollo de la criptografía moderna hasta el estudio de la generación de números aleatorios, curvas elípticas, firmas digitales y diversos algoritmos de encriptación asimétrica. Además de esto, también hemos abordado la infraestructura de clave pública (PKI) y hemos hablado de otros campos de la encriptación como el criptoanálisis y cómo la criptografía ayuda a proteger la seguridad de las redes WiFi y en general las comunicaciones digitales, analizando protocolos como SSL/TLS.

La conclusión principal a la que he llegado una vez finalizado este trabajo es que actualmente existe una gran desinformación en la era digital [35]. Esta desinformación incluye al campo de la criptografía y esto puede ser un gran problema debido a que a medida que la sociedad digital se desarrolla, también es de vital importancia desarrollar una seguridad que complemente todos estos avances, además de tomar precauciones y medidas de seguridad para mantener los datos protegidos. Hay grandes posibilidades de que estas medidas de seguridad no se hagan efectivas si no se está concienciado de la importancia de esto, por eso mismo se puede afirmar que la gran desinformación que nos acecha hoy en día es negativa y se ha de intentar remediar a futuro.

Este trabajo ha sido un viaje a través de la historia y la teoría de la criptografía y espero que haya proporcionado una visión clara y completa de esta disciplina que día a día cobra más importancia. Aunque este trabajo llega a su fin, el estudio y la exploración de la criptografía no lo hacen. Como dijo el famoso criptógrafo Auguste Kerckhoffs: “La criptografía es necesaria, pero no es infalible”. Por lo tanto, debemos seguir trabajando para fortalecerla y adaptarla a los desafíos del futuro.

Finalmente decir que desde un aspecto más personal, este proyecto ha sido una gran experiencia enriquecedora que me ha permitido un mayor entendimiento del amplio campo de la seguridad informática, concretamente de la rama de la criptografía, por lo que he desarrollado un gran interés durante el periodo de investigación para este trabajo de final de grado.

Agradecer a mi tutor Nahuel Norberto Statuto por la paciencia, la ayuda proporcionada y la tutoría realizada a lo largo de todo este proceso de la investigación de la criptografía.

Bibliografía

[1] **Autor desconocido. (Fecha desconocida).** Criptografía y la máquina Enigma. Universidad de Murcia. Recuperado de https://www.um.es/documents/3239701/10301477/criptografia.pdf/09a15c34-3998-4966-991f-fa9210511080

[2] **Jose Luis Perea Vega. (2024)**. Teoría de la información. Recuperado de https://www.aliat.click/BibliotecasDigitales/sistemas/Teoria_de_la_informacion.pdf

[3] **Jordi Herrera Joancomartí. (2022)**. Cifrado en flujo. Recuperado de https://www.aliat.click/BibliotecasDigitales/sistemas/Teoria_de_la_informacion.pdf

[4] **Jordi Herrera Joancomartí. (2022)**. Cifrado en bloque. Recuperado de https://www.aliat.click/BibliotecasDigitales/sistemas/Teoria_de_la_informacion.pdf

[5] **Schneier, B. (1996).** Applied Cryptography: Protocols, Algorithms, and Source Code in C. John Wiley & Sons.

[6] Cifrado DES, 3DES y AES:

https://askanydifference.com/es/difference-between-aes-and-3des/

[7] **Plaza Martín, F. J. (2023).** De Estepa a Salamanca. Miradas en torno a la lengua. Ediciones Universidad de Salamanca. https://doi.org/10.14201/0AQ0351

[8] SHA y MD5 algorithms: https://www.freecodecamp.org/espanol/news/md5-vs-sha-1-vs-sha-2-cual-es-el-hash-cifrado-mas-seguro-y-como-verificarlos/

[9] HMAC:

https://barcelonageeks.com/que-es-hmac-codigo-de-autenticacion-de-mensajes-basado-en-hash/

[10] **Plaza Martín, F. J. (2023).** De Estepa a Salamanca. Miradas en torno a la lengua. Ediciones Universidad de Salamanca. https://doi.org/10.14201/0AQ0351

[11] **Plaza Martín, F. J. (2023).** De Estepa a Salamanca. Miradas en torno a la lengua. Ediciones Universidad de Salamanca. https://doi.org/10.14201/0AQ0351

[12] **Bianco, A. M., & Martínez, E. J. (2004).** Probabilidades y Estadística (Computación). Facultad de Ciencias Exactas y Naturales. Universidad de Buenos Aires. Recuperado de https://www.dm.uba.ar/materias/probabilidades_estadistica_C/2004/1/PyEC08.pdf

[13] **Ivorra Castillo, C. (2020).** Curvas Elípticas. Universitat de València. Recuperado de https://www.uv.es/ivorra/Libros/Elipticas.pdf

[14] ¿Qué es la firma digital? https://ciberseguridad.com/servicios/firma-digital/

[15] **Plaza Martín, F. J. (2023).** De Estepa a Salamanca. Miradas en torno a la lengua. Ediciones Universidad de Salamanca. https://doi.org/10.14201/0AQ0351

[16] **Quijada Van Den Berghe, C., Alonso Pascua, B., Escudero Paniagua, F., Martín Gallego, C., & Garrido Vílchez, G. B. (Eds.). (2023).** De Estepa a Salamanca. Miradas en torno a la lengua. Ediciones Universidad de Salamanca. https://doi.org/10.14201/0AQ0351

[17] **Quijada Van Den Berghe, C., Alonso Pascua, B., Escudero Paniagua, F., Martín Gallego, C., & Garrido Vílchez, G. B. (Eds.). (2023).** De Estepa a Salamanca. Miradas en torno a la lengua. Ediciones Universidad de Salamanca. https://doi.org/10.14201/0AQ0351

[18] **Quijada Van Den Berghe, C., Alonso Pascua, B., Escudero Paniagua, F., Martín Gallego, C., & Garrido Vílchez, G. B. (Eds.). (2023).** De Estepa a Salamanca. Miradas en torno a la lengua. Ediciones Universidad de Salamanca. https://doi.org/10.14201/0AQ0351

[19] Libro ECDSA: https://libroblockchain.com/ecdsa/

[20] **Josefsson, S., & Liusvaara, I. (2017).** RFC 8032 - Edwards-Curve Digital Signature Algorithm (EdDSA). Internet Research Task Force (IRTF). Recuperado de https://datatracker.ietf.org/doc/html/rfc8032

[21] **Josefsson, S., & Liusvaara, I. (2017).** RFC 8032 - Edwards-Curve Digital Signature Algorithm (EdDSA). Internet Research Task Force (IRTF). Recuperado de https://datatracker.ietf.org/doc/html/rfc8032

[22] **Dueñas, D. (2022). Guía de lectura de Information Architecture. For the Web and Beyond de Rosenfeld, Morville y Arango (2015).** Universitat Oberta de Catalunya. Recuperado de https://openaccess.uoc.edu/bitstream/10609/145975/7/ddeduenasTFG0622memoria.pdf

[23] **Dueñas, D. (2022). Guía de lectura de Information Architecture. For the Web and Beyond de Rosenfeld, Morville y Arango (2015).** Universitat Oberta de Catalunya. Recuperado de https://openaccess.uoc.edu/bitstream/10609/145975/7/ddeduenasTFG0622memoria.pdf

[24] **Dueñas, D. (2022). Guía de lectura de Information Architecture. For the Web and Beyond de Rosenfeld, Morville y Arango (2015).** Universitat Oberta de Catalunya. Recuperado de https://openaccess.uoc.edu/bitstream/10609/145975/7/ddeduenasTFG0622memoria.pdf

[25] **Dueñas, D. (2022). Guía de lectura de Information Architecture. For the Web and Beyond de Rosenfeld, Morville y Arango (2015).** Universitat Oberta de Catalunya. Recuperado de https://openaccess.uoc.edu/bitstream/10609/145975/7/ddeduenasTFG0622memoria.pdf

[26] **DigiCert, Inc. (2020).** ¿NECESITA SEGURIDAD? TODO LO QUE NECESITA ES PKI. Recuperado de https://www.digicert.com/resources/pki-trust-ebook-es.pdf

[27] **DigiCert, Inc. (2020).** ¿NECESITA SEGURIDAD? TODO LO QUE NECESITA ES PKI. Recuperado de https://www.digicert.com/resources/pki-trust-ebook-es.pdf

[28] **DigiCert, Inc. (2020).** ¿NECESITA SEGURIDAD? TODO LO QUE NECESITA ES PKI. Recuperado de https://www.digicert.com/resources/pki-trust-ebook-es.pdf

[29] **Fúster Sabater, A., & Hernández Encinas, L. (2004).** Técnicas Criptográficas de Protección de Datos (3ª Edición actualizada). Grupo Editorial RA-MA. https://doi.org/10.14201/0AQ0351

[30] **Fúster Sabater, A., & Hernández Encinas, L. (2004).** Técnicas Criptográficas de Protección de Datos (3ª Edición actualizada). Grupo Editorial RA-MA. https://doi.org/10.14201/0AQ0351

[31] Algoritmo de cifrado IEEE 802.11 (WiFi): https://community.fs.com/es/article/wep-vs-wpa-vs-wpa2-vs-wpa3.html

[32] ¿Qué es un secreto compartido? https://techlib.net/techedu/secreto-compartido/

[33] **DigiCert, Inc. (2019).** Guía para principiantes sobre los certificados TLS/SSL. Recuperado de https://www.digicert.com/resources/beginners-guide-to-tls-ssl-certificates-whitepaper-es-2019.pdf

[34] ¿Qué es una VPN y cómo funciona?

https://www.xataka.com/basics/que-tipos-vpn-existen-que-se-diferencial-cual-te-conviene-tu-uso

[35] **Oficina de Ciencia y Tecnología del Congreso de los Diputados (Oficina C). (2023).** Informe C: Desinformación en la era digital. Recuperado de https://oficinac.es/sites/default/files/informes/OFICINAC_desinformaci%C3%B3n_20231214_web.pdf

[36] **Página web "encryptdata" que se ha credao para este trabajo:** https://encryptdata.netlify.app/