Para empezar a entender estos temas de los algoritmos de cifrado debemos saber el cómo se dividen y cuáles son las diferencias entre cada uno.

Existen dos tipos de cifrado según el tipo de sus claves. La criptografía cuyos algoritmos solo usan que es la simétrica y la criptografía de algoritmos de dos llaves, criptografía asimétrica.

Empezaremos con el cifrado simétrico.

Cifrado simétrico:

Este tipo de cifrado es de los primeros que existieron y se basa en utilizar una única clave secreta que se encarga de cifrar y descifrar la información y al ser una sola clave es necesario que el emisor como el receptor tengan esta para obtener la información correcta.
Cualquier otro usuario que quiera acceder al mensaje cifrado, deberán tener la contraseña de descifrado, de lo contrario el mensaje será erróneo o incompleto.
El método para cifrar los datos se basa en que el emisor va a cifrar el mensaje con su clave privada, lo enviará a través de un canal y el destinatario lo tendrá que descifrar con la misma contraseña o clave privada que ha usado el emisor.
Una de las desventajas de este tipo de algoritmos es que es antiguo y el problema que se presenta es la distribución de tener una clave privada para todos los usuarios, tanto emisores como receptores de la información, para cifrar y descifrar la información del mensaje.
Los ataques por fuerza bruta son unas de las desventajas de los algoritmos simétricos, hay que tener en cuenta que estos algoritmos son públicos y que la fuerza de estos depende directamente de lo complejo que sea el algoritmo internamente, y también de la longitud de la clave empleada para evitar estos ataques.
Por otro lado, las ventajas de los algoritmos simétricos es que son rápidos y con el tiempo se ha ido mejorando e incorporando a hardware como los procesadores de ordenadores, servidores, routers, etc. La velocidad también depende del algoritmo de cifrado simétrico a utilizar.

Estos algoritmos se aplican en una amplia gama de situaciones donde se requiere proteger la confidencialidad de la información como, por ejemplo:

- **Comunicaciones seguras**: En aplicaciones de mensajería instantánea y correos electrónicos, se utilizan para asegurar que los mensajes y las conversaciones permanezcan privadas entre el remitente y el destinatario. Aplicaciones como WhatsApp y Telegram utilizan este tipo de cifrado simétrico.
- **Almacenamiento de datos sensibles**: En dispositivos de almacenamiento, como discos duros, unidades USB y servicios de almacenamiento en la nube, se utilizan algoritmos de cifrado simétrico.

- **Protección de contraseñas**: En sistemas de autenticación y seguridad, como el almacenamiento de contraseñas en bases de datos, se utilizan algoritmos de cifrado simétrico para proteger las contraseñas de los usuarios.
- **Protocolos de seguridad de redes**: En protocolos de seguridad de redes, como SSL/TLS, SSH, se utilizan algoritmos de cifrado simétrico para proteger la comunicación entre dispositivos y servidores.

Ahora veremos algunos de los algoritmos simétricos más usados.

AES (Advanced Encryption Standard)

El algoritmo simétrico AES es el empleado actualmente en todos los canales y protocolos seguros como TLS, FTPES, redes privadas virtuales, etc. El cifrado de AES puede ser empleado tanto en software como en hardware, AES es un algoritmo de cifrado por bloques, **el tamaño fijo del bloque es de 128 bits**. La longitud de la clave se puede elegir, y se tiene disponible **128, 192 y 256 bits**, siendo la longitud de 128 bits el estándar, pero también se usa el de los 256 bits.

AES se encarga de generar una matriz de 4×4, cifra la información en un solo bloque (cifrado por bloques), y lo hace bloque a bloque en lo que se denomina "rondas". Los datos se convierten inicialmente en bloques, y luego éstos se cifran con la clave en diferentes rondas dependiendo del tamaño de la clave.

- Para una clave de 128 bits se aplican 10 rondas de cifrado y nos genera una combinación potencial de 3.4 x10^38 combinaciones.

- Para una clave de 192 bits se aplican 12 rondas y nos genera una combinación potencial de 6.2 x10^57 combinaciones.

- Para una clave de 256 bits las rondas aplicadas son 14 y nos genera una combinación de 1.1 x10^77 combinaciones.

La forma en que se cifra es la siguiente:

Cada ronda consta de cuatro operaciones básicas:

- **Sustitución de bytes:** Se sustituyen los bytes del bloque de datos por otros bytes, según una tabla de sustitución fija conocida como S-box.
- **Desplazamiento de filas**: Todas las filas del texto recién encriptado se desplazan una posición, excepto la primera fila.
- **Mezcla de columnas**: Se realiza una operación de multiplicación de matrices entre cada columna del bloque de datos y una matriz fija.
- **Adición de la clave de ronda**: Se combina el bloque de datos con una subclave derivada de la clave original mediante una operación XOR.

- Este proceso se repite para las rondas que se hayan elegido.

Texto cifrado: Una vez completadas todas las rondas de cifrado, se obtiene el texto cifrado, que consiste en los bloques de datos cifrados.

Texto descifrado:

- Para descifrar el mensaje cifrado, se utiliza la misma clave de cifrado que se utilizó para cifrarlo.

- El proceso de descifrado con AES implica aplicar una serie de rondas de descifrado, que son similares a las rondas de cifrado, pero en orden inverso.

Este algoritmo se puede encontrar en diferentes aplicaciones como:

- **VPNs**: Como este proceso conecta a los usuarios con diferentes servidores, se utiliza el cifrado AES para proteger los datos del usuario contra filtraciones y ciberataques.
- **Gestores de contraseñas**: Los gestores de contraseñas se utilizan para almacenar de forma segura las credenciales de inicio de sesión bajo una única clave maestra. AES se utiliza en estos casos para proteger este tipo de software.
- **Wi-Fi:** la conexión inalámbrica a Internet suele utilizar muchos métodos de cifrado –como WPA2– y AES se encuentra a menudo en esas conexiones.
- **Aplicaciones móviles**: Cualquier aplicación que incluya mensajería o intercambio de fotos suele utilizar AES para ayudar a la seguridad de los datos.
- AES se utiliza en diferentes lenguajes de programación, como C, C++, Java o Python. También que está presente en programas de compresión de archivos como pueden ser WinZip, 7 Zip, etc.

ChaCha20

El algoritmo ChaCha20 es un algoritmo de cifrado simétrico que soporta claves de 128 y 256 bits y de alta velocidad, a diferencia de AES que es un cifrado por bloques, ChaCha20 es un cifrado de flujo. El número 20 viene de que ChaCha20 realiza 20 rondas de funciones no lineales para poder cifrar información.

ChaCha20 está diseñado para ser soportado por dispositivos que no posean una capacidad de procesamiento tan grande como lo son dispositivos IoT, teléfonos celulares, relojes inteligentes, etc.
Este algoritmo tiene un funcionamiento bastante sencillo, el mensaje es encriptado por medio de aplicar una operación XOR de los bits del mensaje con una serie de "llaves secretas", modificando el mensaje de tal forma que sea imposible leerlo, pero, que también pueda ser reversible aplicando el mismo método.

Chacha20 hace uso de un "generador de llaves secretas" el cual solo necesita de 2 cosas, una llave o clave de 256 bits que será la que usáremos para encriptar y desencriptar el mensaje, una variable llamada "*nonce*" el cual es un número aleatorio que solo debería ser utilizado una única vez y no volver a usarse en encriptaciones futuras.

- **Nonce**: es un valor arbitrario que se combina con la clave secreta para generar el flujo de cifrado.
- El **contador** es un valor que se incrementa cada vez que se cifra un nuevo bloque.

Juntos el nonce y el contador aseguran que el flujo de cifrado generado sea diferente para cada bloque.

Como funciona el algoritmo

- Se inicia con una matriz de 4x4 bloques, cada uno de estos bloques de 32 bits, dando un total una matriz de 512 bits:
    - La primera fila de esta matriz se llena con una cadena que siempre es constante, la cual es "expand 32-byte k".
    - Las siguientes 2 filas se llenan con la clave que el usuario ingrese.
    - La última fila se llena con 2 variables, el contador y el nonce, estos se reparten entre las 4 últimas celdas.
- Ahora que se tiene la matriz preparada, se toma una "copia" de esta en un buffer que se usara más tarde.
- Se entra a un proceso llamado "Quarter-round", el cual se va a dedicar a mezclar toda la información de la matriz para obtener las llaves secretas con las que trabajará.

- La matriz se va a dividir de 2 formas distintas, por sus columnas o en diagonales.
- Una vez dividida la matriz toca hacer una Quarter-round, para lo cual tomaremos una columna o diagonal.
- Sigue una serie de operaciones binarias entre las celdas las cuales son las siguientes.
    - Suma binaria.
    - Operación XOR.
    - Desplazamiento de bit n a la izquierda.

Repitiendo los pasos

- Esta serie de operaciones se repiten un total de 20 veces, alternando en dividir la matriz en columnas o diagonales.
- Se toma la copia de la matriz original que se tomó antes y se le suma a la matriz mezclada para ahora sí, imposibilitar el obtener la matriz original sin la clave principal.
- Esta clave final termina por hacer XOR con una parte del mensaje original y en caso de necesitar más claves secretas para mensajes largos, se repite todo el proceso de generar una clave secreta, solo que el contador incrementa en 1 y el nonce genera un nuevo número aleatorio. Este proceso se repite hasta obtener nuestro mensaje encriptado.

Algoritmo asimétrico.

La criptografía de clave asimétrica también es conocida como clave pública, **emplea dos llaves diferentes en cada uno de los extremos de la comunicación para cifrarla y descifrarla**. Cada usuario tendrá una clave pública y otra privada.

La clave pública será accesible por todos los usuarios del sistema que quieran comunicarse. Esta clave no sirve para el proceso de descifrado. Un usuario necesita tener una clave secundaria, es decir, la clave privada, para descifrar esta información. De este modo, la clave privada sólo la tiene la persona responsable de descifrar la información, sin sacrificar la seguridad a medida que se escala la seguridad.

El proceso generalmente implica lo siguiente:

- **Cifrado y descifrado:** Para cifrar un mensaje, se utiliza la clave pública del destinatario. El mensaje cifrado solo puede descifrarse con la clave privada correspondiente del destinatario. Esto proporciona confidencialidad a los datos.
- **Firma digital:** para firmar digitalmente un mensaje, se utiliza la clave privada del remitente, y la firma puede ser verificada por cualquier persona con la clave pública del remitente. Esto garantiza la autenticidad e integridad del mensaje.

La estructura del funcionamiento del cifrado asimétrico funciona de esta forma:

- Mensaje + clave pública = Mensaje cifrado
- Mensaje encriptado + clave privada = Mensaje descifrado
- Mensaje + clave privada = Mensaje firmado
- Mensaje firmado + clave pública = Autenticación

Ventajas:

- **Facilidad en el intercambio de claves**: La clave pública de un usuario puede ser compartida abiertamente, lo que facilita el intercambio seguro de claves entre partes que desean comunicarse de manera segura.
- **Autenticación y no repudio:** Los algoritmos asimétricos permiten la autenticación de usuarios y la firma digital de documentos, lo que garantiza la integridad y autenticidad de la información. Además, el remitente no puede negar haber enviado un mensaje firmado digitalmente, lo que se conoce como no repudio.
- **Seguridad en la comunicación en línea**: Los algoritmos asimétricos proporcionan un nivel adicional de seguridad en la comunicación en línea al

cifrar datos de forma que solo el destinatario puede descifrarlos, incluso si el mensaje se intercepta durante la transmisión.

- **Escalabilidad:** Los sistemas de clave pública pueden admitir una amplia gama de usuarios sin necesidad de compartir secretos entre ellos, lo que los hace escalables y adecuados para implementaciones en grandes redes.

Desventajas:

- **Rendimiento computacional**: Los algoritmos asimétricos suelen ser más lentos en términos de rendimiento computacional en comparación con los algoritmos simétricos. Esto se debe a la complejidad matemática de las operaciones necesarias para cifrar y descifrar los datos.
- **Tamaño de clave**: Las claves asimétricas tienden a ser más largas en comparación con las claves simétricas, lo que puede resultar en un mayor consumo de recursos de almacenamiento y ancho de banda.
- **Vulnerabilidad a ataques de canal lateral**: Los algoritmos asimétricos pueden ser más susceptibles a ataques de canal lateral, como ataques de caché o temporales, que pueden permitir deducir información sobre las claves privadas.

Estos algoritmos se pueden encontrar en:

- **Protocolos de comunicaciones seguras:**
  - Protocolo SSH (Secure Shell): Utiliza intercambio de claves asimétrico (como Diffie-Hellman) para establecer un canal seguro para las comunicaciones remotas.
  - Protocolo IPsec (Seguridad de la Capa de Internet): Utiliza intercambio de claves asimétrico para establecer asociaciones de seguridad y garantizar comunicaciones VPN seguras.

- **Firmas digitales:**
  - Los algoritmos asimétricos se utilizan para crear y verificar firmas digitales, lo que garantiza la autenticidad, la integridad y la no repudiación de documentos electrónicos. Las firmas digitales son comúnmente utilizadas en contratos electrónicos, transacciones financieras y documentos legales.
- **Autenticación de usuarios:**
  - Los sistemas de autenticación de usuarios, como SSH y SSL/TLS, utilizan algoritmos asimétricos para autenticar a los usuarios y establecer conexiones seguras entre clientes y servidores.
- **Servicios en línea y aplicaciones web:**
  - Plataformas como PayPal, bancos en línea y otros servicios utilizan cifrado asimétrico para proteger las comunicaciones y transacciones financieras.
- **Dispositivos móviles e IoT:**
  - Los algoritmos asimétricos, especialmente ECC, se utilizan en dispositivos con recursos limitados para establecer comunicaciones seguras y actualizaciones de firmware seguras.

- **Blockchain y criptomonedas:**
    - Las criptomonedas como Bitcoin utilizan criptografía de curva elíptica (ECC) para generar y administrar las direcciones y firmas digitales en las transacciones.

Ahora hablaremos de algunos algoritmos asimétricos que se usan:

RSA(Rivest-Shamir-Adleman)

RSA es un algoritmo de cifrado de clave pública que lleva el nombre de sus inventores: Ron Rivest, Adi Shamir y Leonard Adleman. Fue presentado por primera vez en 1977 y desde entonces ha sido ampliamente usado en muchos lugares.
El propósito principal de RSA es permitir el cifrado y descifrado de datos utilizando dos claves diferentes: una clave pública para cifrar y una clave privada para descifrar. Esto resuelve el problema de distribución de claves que enfrentan los algoritmos simétricos.

RSA utiliza la factorización del producto de dos números primos para ofrecer un cifrado de 1024 bits y una longitud de clave de hasta 2048 bits.
Además de cifrar/descifrar, RSA también se utiliza ampliamente para firmas digitales y autenticación.
Funcionamiento:

Primero se tienen que generar las claves:

- Se eligen dos números primos grandes, p y q, de forma aleatoria.
- Se calcula n = p * q (n se llama el módulo)

- Con fi que es la función φ de Euler se calcula φ(n) = (p - 1) * (q – 1)

- Se escoge un entero positivo e menor que φ(n) o que sea co-primo con φ(n).

- Se calcula d, el módulo multiplicativo inverso de e, es decir, d = e^-1 mod φ(n).

- f. La clave pública es el par (e, n) esto es, el módulo y el exponente de cifrado

- la clave privada es (d, n) esto es, el módulo y el exponente de descifrado, que debe mantenerse en secreto.

Cifrado de un mensaje:

- Convertimos el mensaje que queremos cifrar en un número entero m, utilizando algún esquema de codificación como UTF-8.
- El texto cifrado c se calcula elevando m a la potencia del exponente de cifrado e y tomando el resultado módulo n: c = m^e mod n.

Descifrado de un mensaje:

- Obtención del texto cifrado: Recibimos el texto cifrado c.
- Cálculo del mensaje original: El mensaje original m se calcula elevando c a la potencia del exponente de descifrado d y tomando el resultado módulo n: m = c^d mod n

El funcionamiento es bastante complejo de entender ya que son muchas operaciones que se realizan, pero a la vez lo hace demasiado seguro según una investigación realizada en 2010, se necesitarían 1.500 años de potencia de cálculo para descifrar su versión de apenas 768 bits, si tenemos en cuenta que el algoritmo ofrece un cifrado de 1024 bits hasta 2048 bits.

es un algoritmo de cifrado más lento. Como requiere dos claves diferentes de una longitud extensa, el proceso de cifrado y descifrado es lento, pero el nivel de seguridad que proporciona para la información sensible es incomparable.

Este tipo de algoritmo se encuentra en:

- **Seguridad en comunicaciones**: SSL/TLS (Capa de Sockets Seguros/Seguridad de la Capa de Transporte): RSA se utiliza en el apretón de manos inicial para el intercambio de claves y la autenticación del servidor web durante el establecimiento de conexiones HTTPS seguras.
- **Autenticación de usuarios:** RSA se utiliza en sistemas de autenticación de usuarios, como SSH (Secure Shell), para verificar la identidad de los usuarios que intentan acceder a sistemas remotos. Los usuarios pueden autenticarse utilizando claves RSA o certificados digitales basados en RSA.
- **Firmas digitales**: RSA se utiliza para crear y verificar firmas digitales en documentos electrónicos, contratos y transacciones financieras, garantizando la autenticidad e integridad de los datos firmados.

EEC (Criptografía de Curva Elíptica)

Este método fue propuesto en 1985 por Neal Koblitz y Victor S. Miller, para ser ECC utiliza una operación matemática basada en curvas elípticas sobre un campo finito ECC se basa en funciones matemáticas que son simples de calcular en una dirección, pero muy difíciles de revertir. Resulta muy fácil crear una clave ECC, pero romperla es prácticamente imposible. En el caso de ECC, esta dificultad reside en la inviabilidad de calcular el logaritmo discreto de un elemento de curva elíptica aleatoria con respecto a un punto base conocido públicamente. Esta propiedad es conocida como el "problema de logaritmo discreto de curva elíptica"

Como funciona

Generación de claves:

El método utiliza una curva elíptica de la forma:

$$y2 = x3 + ax + b$$

Donde a y b son parámetros que determinan la forma y la posición de la curva. La curva también puede contener un punto llamado "punto en el infinito", que actúa como el elemento neutro en la operación de suma.

El primer paso es elegir una de las curvas del dominio, estos parámetros se deben elegir con especial cuidado ya que no todas las curvas poseen la misma fortaleza, por lo que es recomendable utilizar curvas con parámetros de dominios que ya estén validados.
Las más comunes son: secp256k1, secp256r1, secp384r1, secp521r1

Se elige un punto base G sobre la curva elíptica Este punto es el mismo para todos los usuarios que utilicen la curva y, por tanto, lo podemos considerar como un dato público base sobre la que se construye la criptografía de curva elíptica es la siguiente ecuación:

$$\mathbf{P = k*G}$$

Donde:

- **P** y **G** son puntos de la curva
- **G:** es el punto base o generador, que es conocido por todo el mundo y único para la curva que estamos utilizando.
- **P:** es la clave pública y consiste en un punto que ha sido generado iterando k veces sobre G.

- **K:** será la clave privada del emisor y es un entero que define el número de iteraciones que se van a realizar sobre el punto G para obtener el punto P.

1. **Clave privada:** k
2. **Clave pública**: P = k*G

Sonara muy raro pensar que si se tiene a P y G como elementos públicos es fácil encontrar K y descifrar el mensaje, pero no es tan sencillo como parece. Dado las combinaciones que se hicieron y la complejidad del algoritmo es imposible tener el equipo adecuado para llegar a la respuesta adecuada.

Ejemplo:

- Supongamos que elegimos un número k de tamaño 128 bits, eso significa que el atacante debe realizar 2^128 operaciones de suma sobre el punto G, hasta encontrar el punto P, en ese momento habría encontrado el número k es la clave privada.
- Suponiendo que se tiene una capacidad de cómputo que permite realizar 1 millón de sumas por segundo, es capaz de realizar unas 220 operaciones por segundo y un año tiene unos 225 segundos.
- En un año se podrán hacer 220 x 225 = 245 sumas.
- Se necesitarían 30.000.000.000.000.000 millones de años para realizar 2128 sumas, teniendo en cuenta que la edad del universo es de 14.000 millones de años, la probabilidad de encontrar k por fuerza bruta es mínimas.
- Esto es lo que hace que el algoritmo sea completamente seguro y no sea vulnerado de esta forma.

Cifrado:

- El remitente obtiene la clave pública Q del destinatario.
- Elige un número entero aleatorio d.
- Calcula el punto C1 = d * G y el punto C2 = M + d * Q, donde M es el mensaje representado como un punto sobre la curva.
- El texto cifrado es el par de puntos (C1, C2).

Descifrado:

- El destinatario recibe el texto cifrado (C1, C2).
- Utilizando su clave privada k, calcula el punto M' = C2 - k * C1.
- M' es el mensaje descifrado, representado como un punto sobre la curva.

ECC en este sentido aporta mayor seguridad y beneficios como los siguientes:

- La ECC permite que las claves sean más pequeñas en extensión en comparación con otros sistemas criptográficos.

- El ECC escala mejor, mientras que el RSA se vuelve más lento y pesado.

Una comparación con el algoritmo anteriormente hablado seria este

Tamaño de clave ECC	**Tamaño de clave RSA**
160 bits	1024 bits
224 bits	2048 bits
256 bits	3072 bits
384 bits	7680 bits
521 bits	15360 bits

Los tamaños de clave más pequeños de ECC significan que se puede lograr un cifrado más fuerte con menos potencia informática

Donde se utilizan:

- **Protocolos de comunicación seguros**: la ECC se utiliza en varios protocolos de comunicación seguros para proporcionar cifrado, firmas digitales e intercambio de claves. Algunos ejemplos son la seguridad de la capa de transporte (TLS) que se utiliza en la navegación web segura, el intérprete de órdenes seguro (SSH) para inicio de sesión remoto seguro y las redes privadas virtuales (VPN) para una comunicación de red segura.
- **Tecnología de criptomonedas y cadena de bloques**: muchas criptomonedas, como Bitcoin, Ethereum y Litecoin, utilizan la criptografía de curva elíptica para generar pares de claves públicas y privadas, así como para firmar transacciones. La ECC ofrece la seguridad criptográfica necesaria para proteger los conjuntos digitales y garantizar la integridad de las redes de cadena de bloques.
- **Tarjetas inteligentes y sistemas integrados**: la ECC se utiliza comúnmente para proteger los sistemas de pago, los sistemas de control de acceso, los pasaportes electrónicos y otras aplicaciones que requieren soluciones criptográficas seguras y compactas.
- **Firmas y certificados digitales:** la ECC se puede utilizar para generar firmas digitales, que se utilizan para verificar la autenticidad y la integridad de los documentos y mensajes digitales. Las firmas digitales basadas en la ECC también se utilizan en los sistemas de infraestructura de clave pública (PKI) para emitir y validar certificados digitales.