

UNIDAD 1

INTRODUCCIÓN A
BLOCKCHAIN

ASPECTOS TEÓRICOS



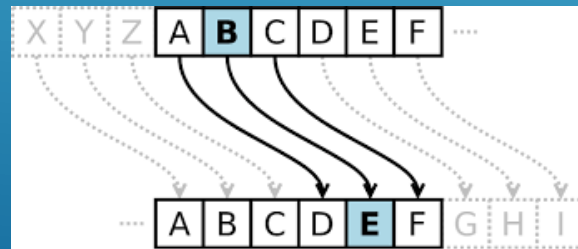
CRIPTOLOGÍA = CRIPTOGRAFÍA + CRIPTOANÁLISIS

CRIPTOGRAFÍA: Estudia el diseño y propiedades de algoritmos y mecanismos que ofrecen confidencialidad, integridad y autenticación.

CRIPTOLOGÍA: Es el opuesto a la criptografía, se ocupa de romper esos procedimientos para así recuperar la información, su objetivo es buscar el punto débil de los criptosistemas para así reducir o eliminar la seguridad que teóricamente aporta este.



Una escíjala.



Cifrado César



Máquina enigma.
Utilizada por los
alemanes durante
la segunda guerra
mundial.

CRIPTOGRAFIA EN LA BLOCKCHAIN

El interés global que despierta la cadena de bloques se debe a que busca mediante la confianza en la tecnología, poder desplazar a intermediarios o terceras partes innecesarias de los procesos administrativos y/o productivos. Si bien la cadena de bloques no nace a partir de una nueva tecnología, se crea con la conjunción de tres tecnologías conocidas existentes: redes P2P, criptografía, y programas (los contratos inteligentes). De estos tres la criptografía es el elemento principal, debido a que es el que nos aporta la confianza. (Holbrook, 2020)

Holbrook, J. (2020). Architecting Enterprise Blockchain Solutions. John Wiley & Sons.

SERVICIOS DE LA CRIPTOGRAFÍA

Confidencialidad: La confidencialidad garantiza que la información está disponible solo para entidades autorizadas.

Integridad: La integridad garantiza que la información solo sea modificada por entidades autorizadas.

Autenticación: La autenticación proporciona seguridad sobre la identidad de una entidad o la validez de un mensaje. Hay dos tipos de mecanismos de autenticación: la autenticación de entidad y la autenticación de origen de los datos.

No repudio: El no repudio es la seguridad de que una entidad no pueda negar un compromiso o acción realizada con anterioridad, proporcionando evidencia irrefutable.

Transparencia: La transparencia, o rendición de cuentas, es la garantía de que existen registros detallados de las acciones de una entidad.

PRIMITIVAS CRIPTOGRÁFICAS

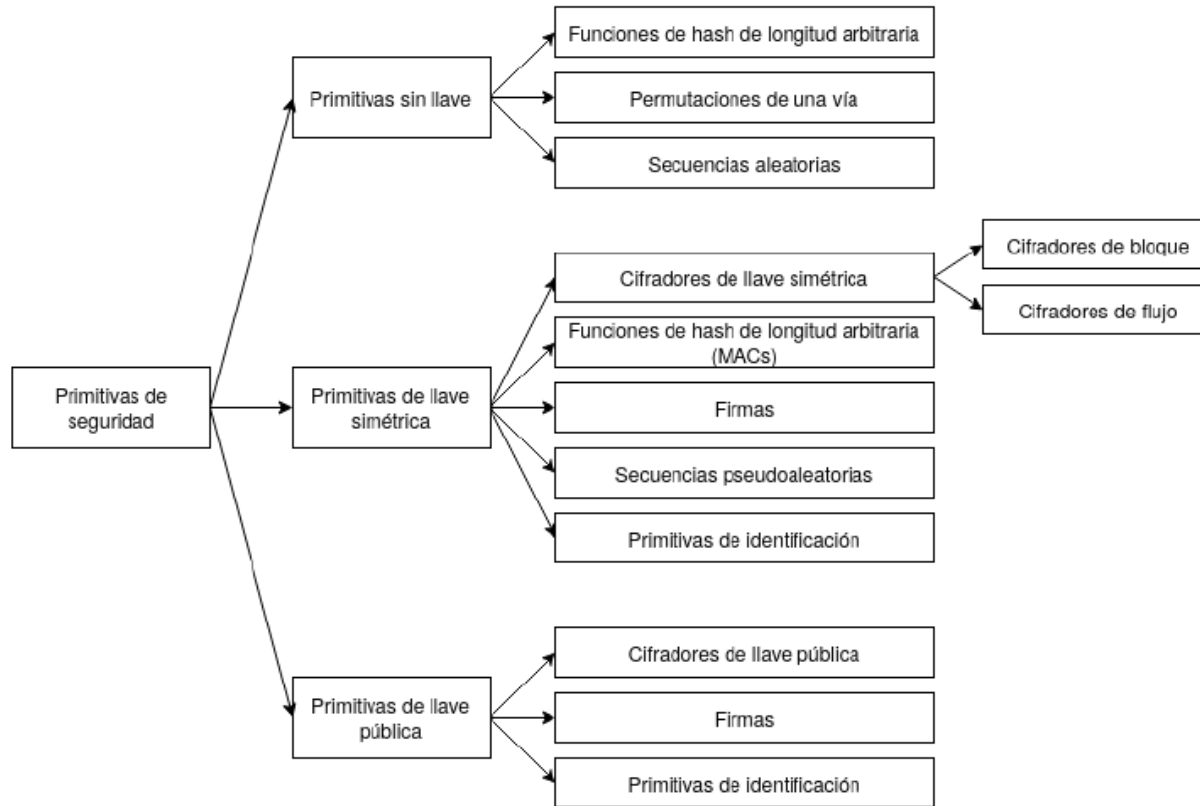
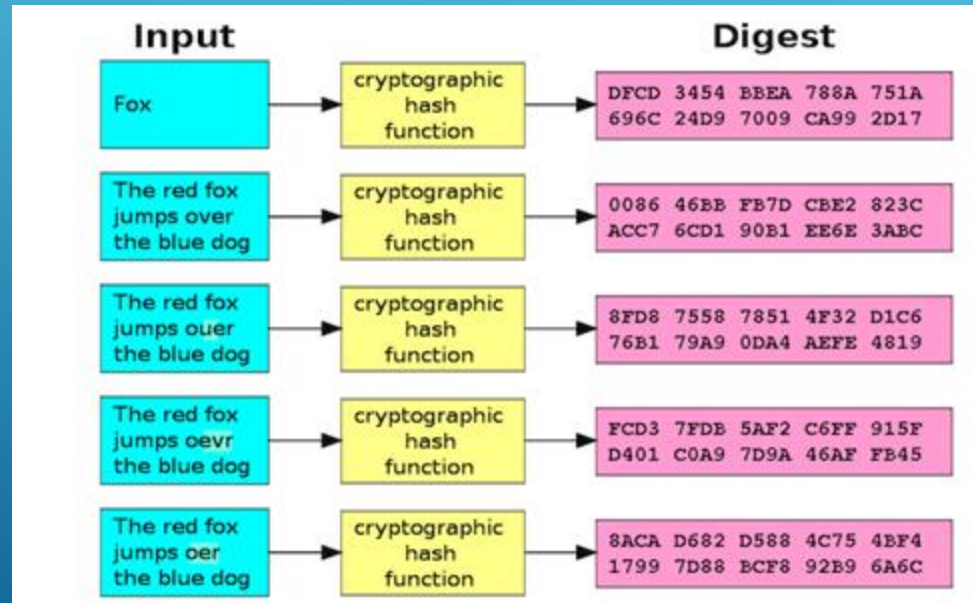


Figura 7.1: Taxonomía de las primitivas criptográficas (Menezes y col., 1996)

(2.B) FUNCIÓN HASH

Una función criptográfica hash es un algoritmo matemático computacionalmente eficiente y unidireccional, que mapea cualquier bloque de entrada arbitrario de datos en una nueva serie de caracteres con una longitud fija. Independientemente de la longitud de los datos de entrada, el valor hash de salida tendrá siempre la misma longitud (generalmente entre 128 bits y 512 bits).



Podemos probar las funciones hash en https://emn178.github.io/online-tools/sha3_256.html y <https://sandbox.eth.build/>

LA FUNCIÓN HASH TIENE LAS SIGUIENTES CARACTERÍSTICAS ESENCIALES:

Determinista: Esto significa que para un mismo conjunto de datos de entrada siempre se ha de obtener el mismo valor del código hash. Se considera como una huella dactilar de los datos de entrada.

Si sabemos que $H(x) = H(y)$,

podemos asumir que $x = y$

Con la ventaja de que el hash es pequeño.

LA FUNCIÓN HASH TIENE LAS SIGUIENTES CARACTERÍSTICAS ESENCIALES:

Unidireccional: El camino inverso es prácticamente imposible (HIDING). No es posible tomar un hash (output) y obtener los datos que dieron origen al mismo (input).

Para un $H(x)$ conocido,

es inviable determinar x

Puzzle-friendly :

Dado $H(r \parallel x)$ con " r " aleatorio, es inviable encontrar x

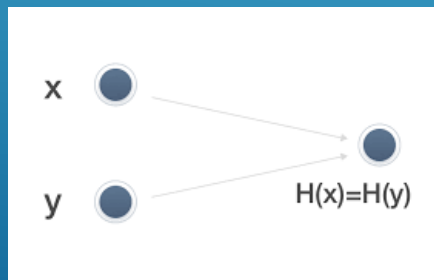
La propiedad puzzle-friendly implica que ninguna estrategia es mejor que intentar con valores aleatorios de x para resolver el "puzzle".

LA FUNCIÓN HASH TIENE LAS SIGUIENTES CARACTERÍSTICAS ESENCIALES:

Resistente a las colisiones: Hace referencia a que la probabilidad de que dos conjuntos de datos diferentes generen el mismo código hash ha de ser altamente improbable.

Nadie debería poder encontrar valores de x e y tal que:

$$x \neq y, H(x) = H(y)$$

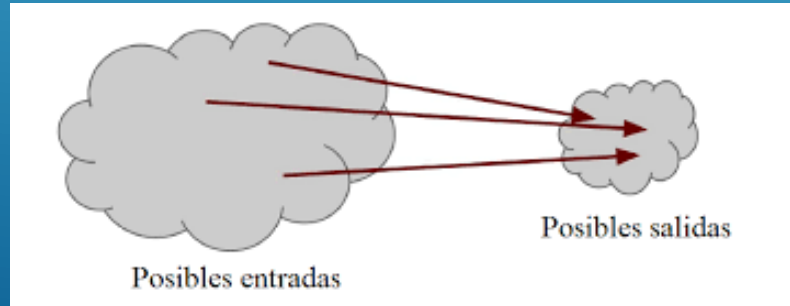


NINGUN HASH ES COLLISION-FREE

Sea H la función Hash()

tomamos 2^{130} inputs al azar


existe 99.8% de probabilidad de que dos de ellos tengan el mismo output, sin importar qué función H tomemos





COLLISION-FREE

La razón por la cual este método de encontrar colisiones no nos es relevante se debe a que nos tomaría todo el tiempo desde el nacimiento de la tierra hasta el día de hoy para realizarlo.



COLLISION-FREE

¿Existe algun otro metodo para probar las colisiones?

Para algunas funciones Hash() se encontraron métodos más rápidos, aunque para otras funciones Hash() aún no se las pudo encontrar.

Sin embargo **no existe prueba** alguna que demuestre que una función Hash() no puede tener un método de colisión más rápido.

Cuando decidimos confiar en una función Hash() es porque, luego de un gran esfuerzo por encontrar un método de collision, **no se ha encontrado ninguno**.

DICHO LO ANTERIOR

```
1 String X;  
2 String Y;  
3 if (Hash(X) == Hash(Y)) {  
4     X = Y // es "razonable" asumir que son identicos  
5 }
```

APLICACIONES: ASUMIRO UN COMPROMISO
(COMMITMENT), OCULTAR INFORMACIÓN, VERIFICAR
INTEGRIDAD, ETC

OTRAS CARACTERÍSTICAS DE LOS HASH

- ⌞ Bajo costo/computacionalmente eficientes: los algoritmos generados por una función hash no requieren de una gran capacidad de cálculo o una gran cantidad de memoria para ejecutarse, deben ser eficientes incluso para mensajes de gran tamaño.
- ⌞ Tamaño fijo: no importa la cantidad de datos que se quiera convertir, ya que esta siempre será una cadena de tamaño fijo. Si se usa el SHA-256, siempre la cadena tendrá 64 caracteres.
- ⌞ Efecto avalancha: Los algoritmos han de ser sensibles a cualquier mínimo cambio en el conjunto de datos de entrada. Al realizar un pequeño cambio en la entrada, por ejemplo, un solo bit, los cambios en el código hash resultante han de ser enormes.

FUNCIONES HASH SEGÚN DISTINTAS CRYPTOS

Bitcoin: Sha-256

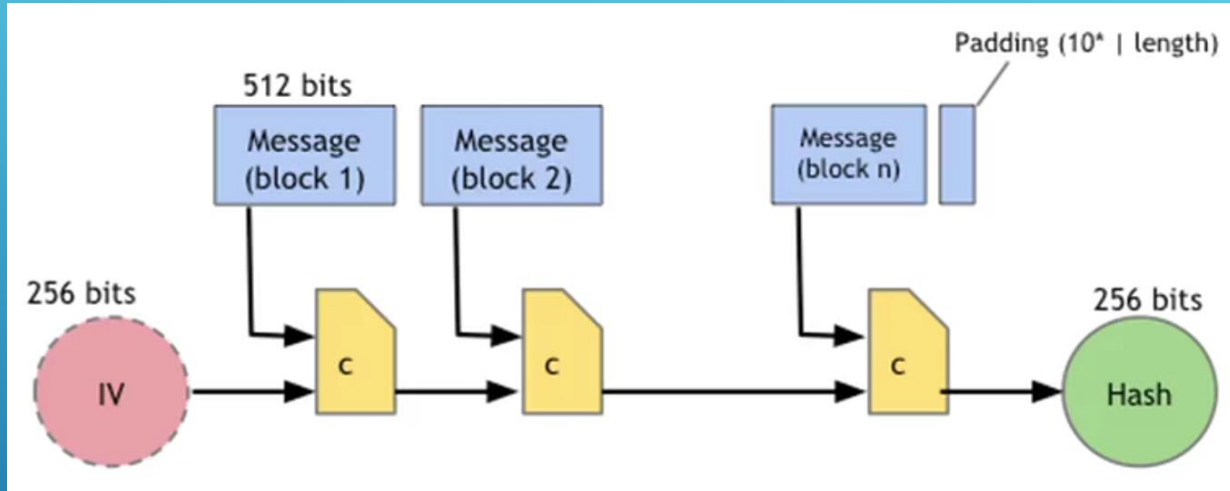
Ethereum: Ethash

Ripple: Ripple Protocol Consensus Alghoriyhm (RPCA)

LiteCoin/DogeCoin:Scrypt

Funciones hash != Algoritmo de minado

SHA 256



SHA-256 se utiliza en algunos de los protocolos de autenticación y cifrado más populares, incluidos SSL, TLS, IPsec, SSH y PGP. En Unix y Linux, SHA-256 se usa para el hash seguro de contraseñas. Las criptomonedas como Bitcoin usan SHA-256 para verificar transacciones.



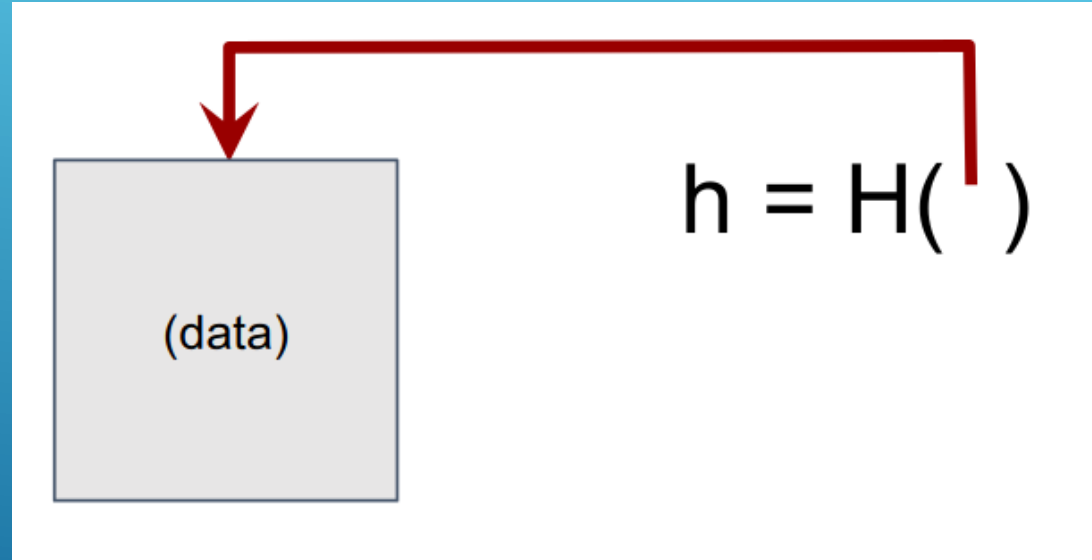
HASH POINTER

Un hash pointer es un puntero que representa la dirección donde los datos son almacenados y adicionalmente almacena el hash de los datos en esa ubicación.

Con el hashpointer podemos:

- Obtener la información, y
 - Verificar que la misma no haya modificada
- 

HASH POINTERS

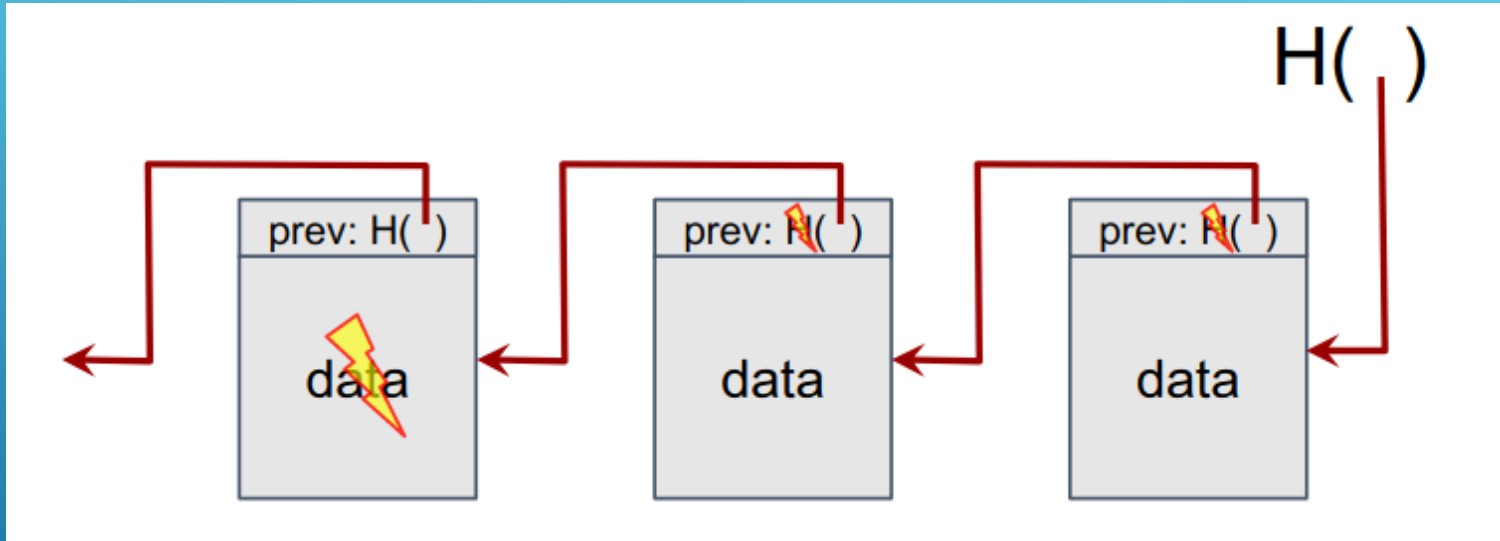


Credit: Authors of “Bitcoin and Cryptocurrency Technologies”



o de

HASH POINTERS – DETECTING TAMPERING

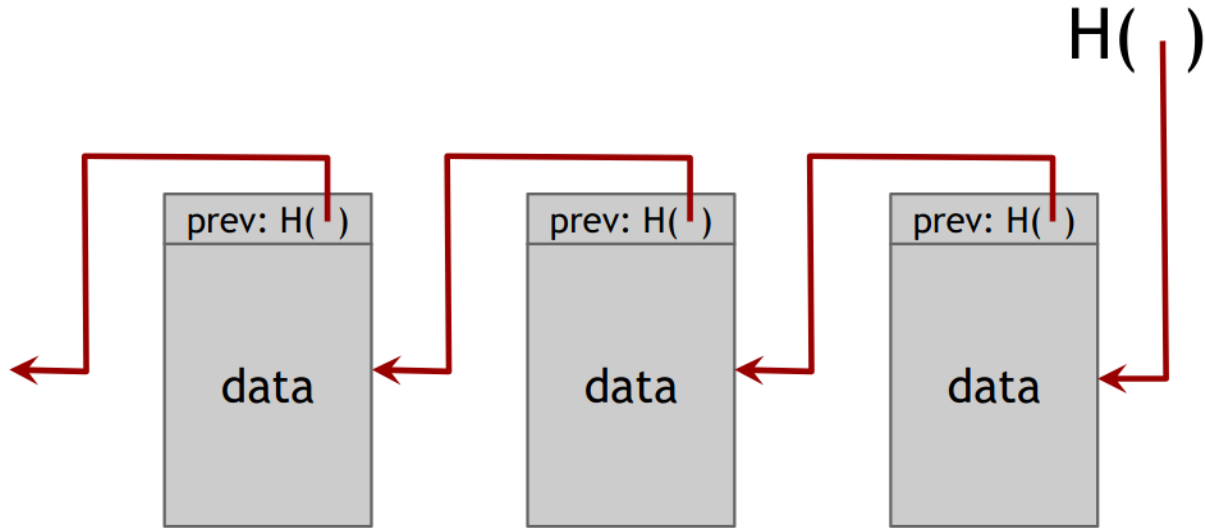


Si los datos se modifican, todos los hashpointers posteriores cambian. De lo contrario, ¡encontró una colisión de hash!

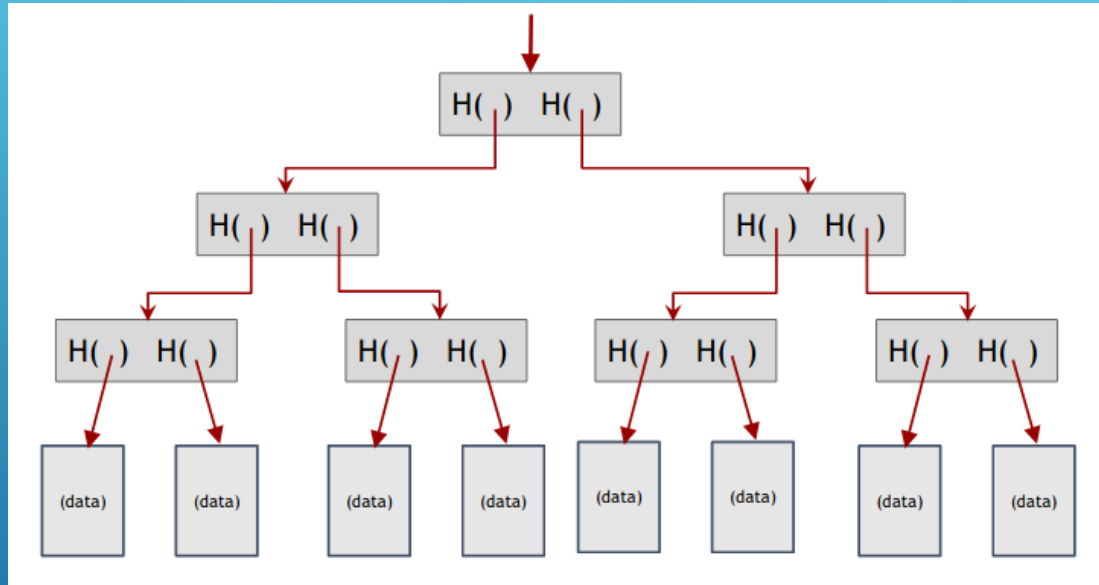
HASH POINTERS

BLOCKCHAIN: APPEND-ONLY HASH CHAIN

Linked list with hash pointers = “Blockchain”

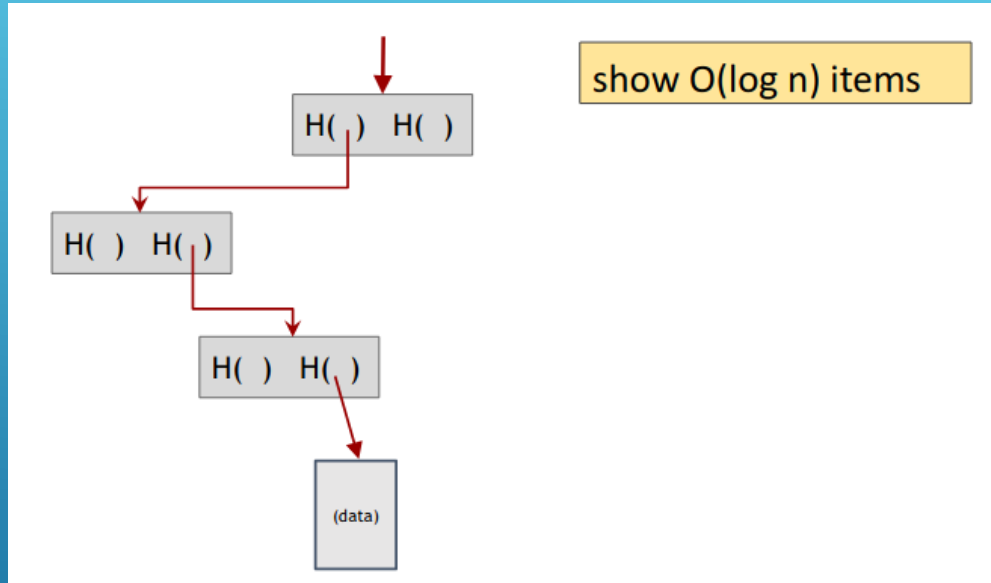


MERKLE TREE - TAMPERPROOF BLOCK OF DATA



Merkle tree es un árbol binario donde se realiza el hash de los nodos "hojas" y luego se vuelven a hashear los resultados de los nodos padres hacia arriba hasta llegar a la raíz o "merkle root".

PROVING MEMBERSHIP IN A MERKLE TREE



Maneja la integridad de una manera más eficiente.

MERKLE TREE - PROPIEDADES

- ↴ fácil de verificar la integridad de la información
 - ↵ un simple valor hash (raíz/root) es el representante de toda la información/data
 - ↵ si cualquier data es cambiada, el hash debe de ser cambiado
- ↴ fácil de verificar la inclusión de transacciones (Merkle Proof)
 - ↵ esto permite una simple verificación de sistemas de transacciones o pagos
 - ↵ ya no es un requisito alojar toda la cadena de información, el hash raíz te garantiza la integridad

SISTEMAS CRIPTOGRÁFICOS

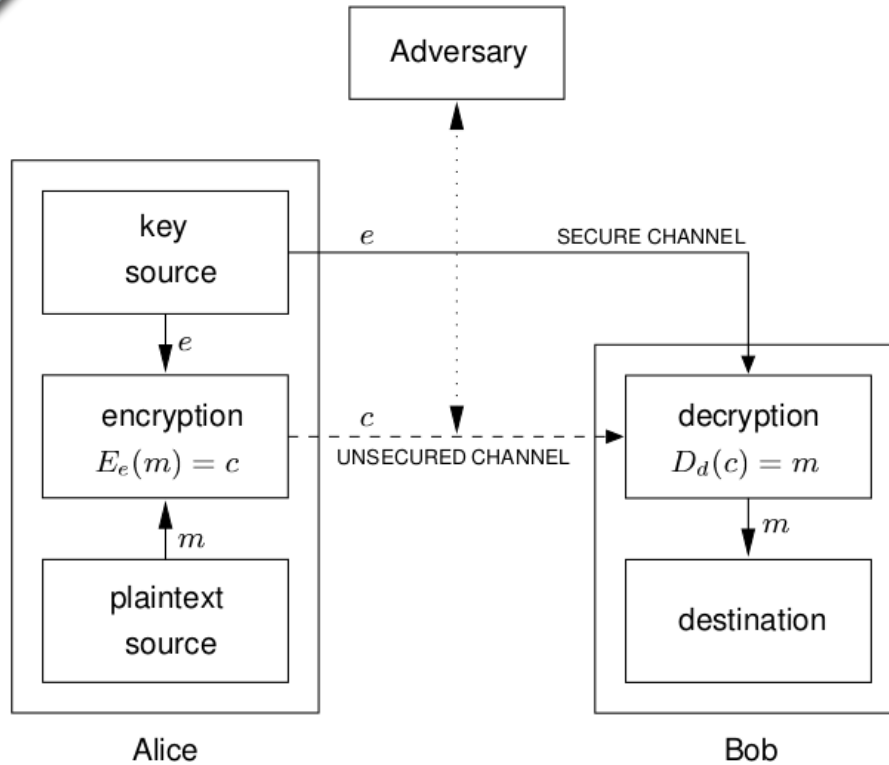
Existen dos tipos básicos de criptosistemas:

- Sistemas de cifrado simétrico (también conocidos como sistemas clave secreta o clave privada)
- Sistemas de cifrado asimétrico (también conocidos como sistemas de clave pública)

CIFRADO SIMÉTRICO

► En el cifrado simétrico, dos entidades comparten una sola clave de cifrado/descifrado.

El principal reto del cifrado simétrico consiste en la distribución y protección de las claves.



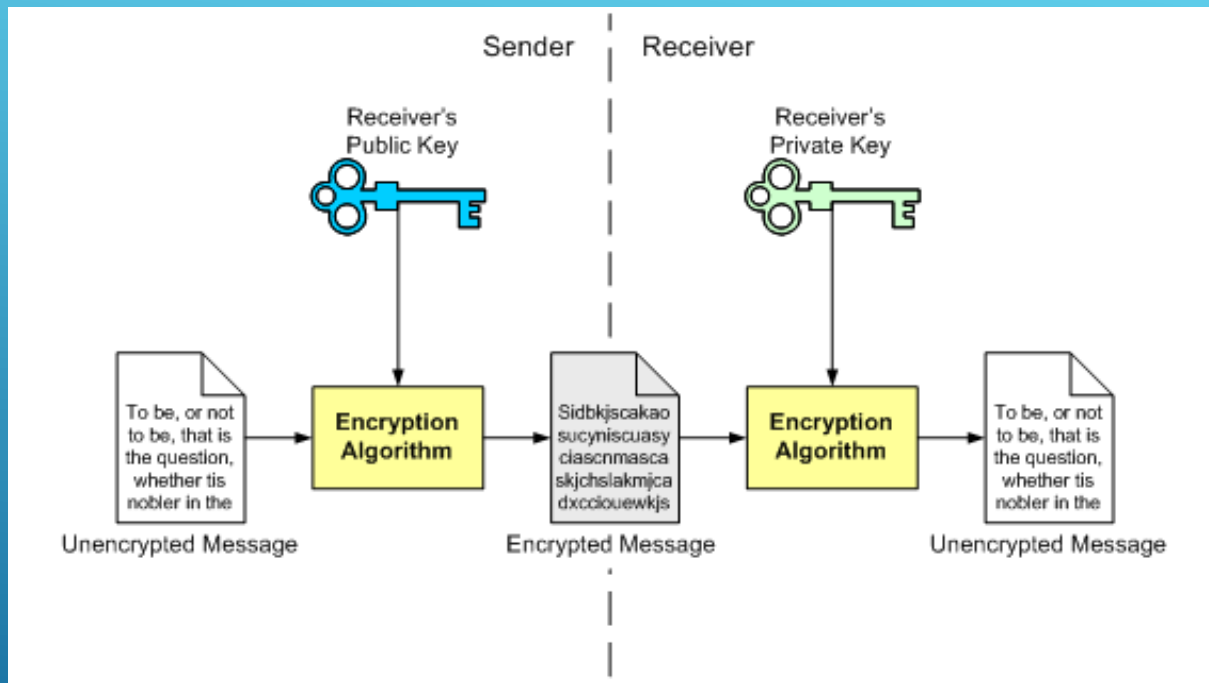
CRIPTOGRAFÍA ASIMÉTRICA

Los sistemas asimétricos utilizan dos claves, una privada y una pública (siendo una la inversa de la otra). Ambas pueden ser usadas para encriptar y desencriptar, dependiendo del modo de operación utilizado (encripción o autenticación).

Dichas claves están matemáticamente relacionadas entre sí y además:

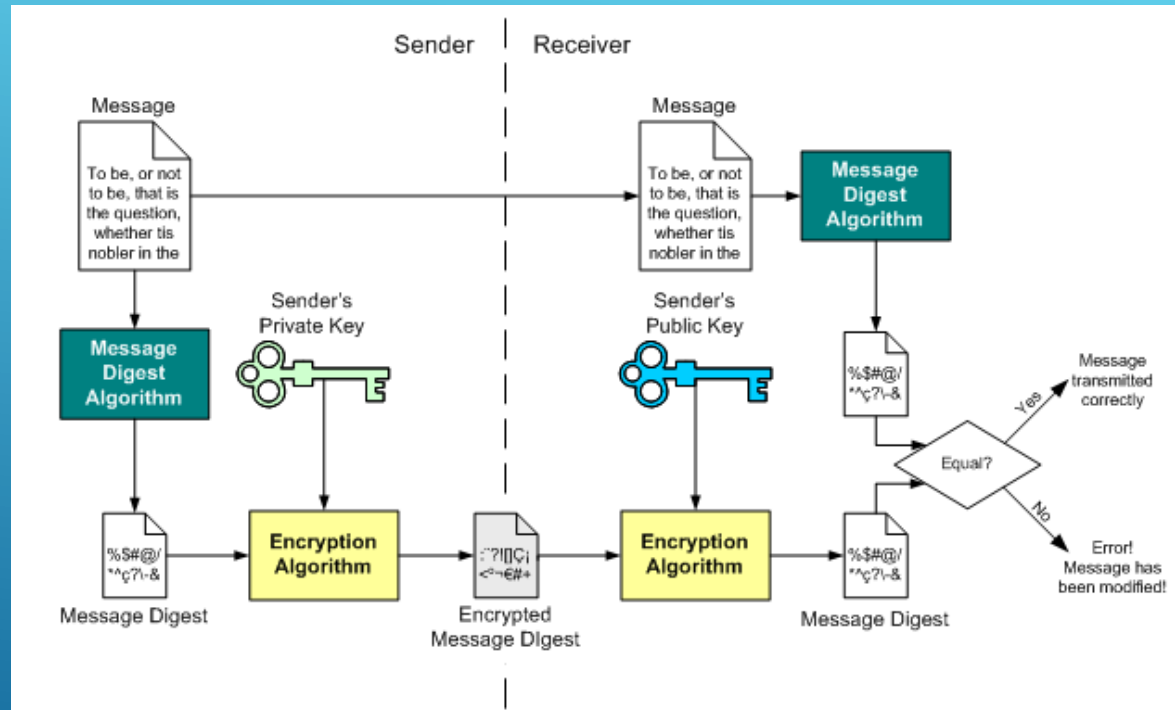
- La clave pública está disponible para todos.
- La clave privada es conocida sólo por el individuo dueño del par de claves.

CRIPTOGRAFÍA ASIMÉTRICA - ENCRIPTACIÓN



El mensaje original es encriptado utilizando la clave pública del receptor, y no puede ser descifrado por nadie (incluyendo al que lo cifró), excepto un poseedor de la clave privada correspondiente.

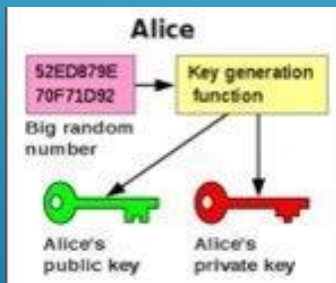
CRIPTOGRAFÍA ASIMÉTRICA- FIRMA DIGITAL



Un mensaje firmado con la clave privada del remitente puede ser verificado por cualquier persona que tenga acceso a la clave pública de dicho remitente.

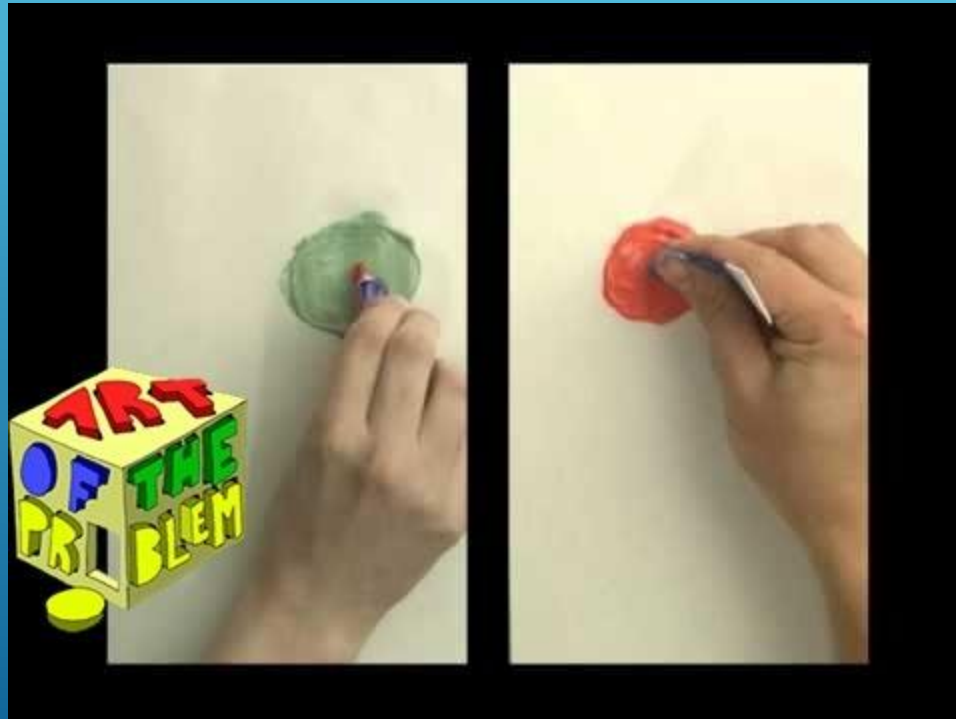
CRIPTOGRAFÍA ASIMÉTRICA

Además, los métodos criptográficos garantizan que esa pareja de claves sólo se puede generar una vez, de modo que se puede asumir que no es posible que dos personas hayan obtenido casualmente la misma pareja de claves.



HISTORIA DE CRIPTOGRAFÍA Y COLORES

https://www.youtube.com/watch?v=YEBfamv-_do

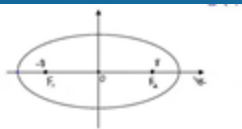
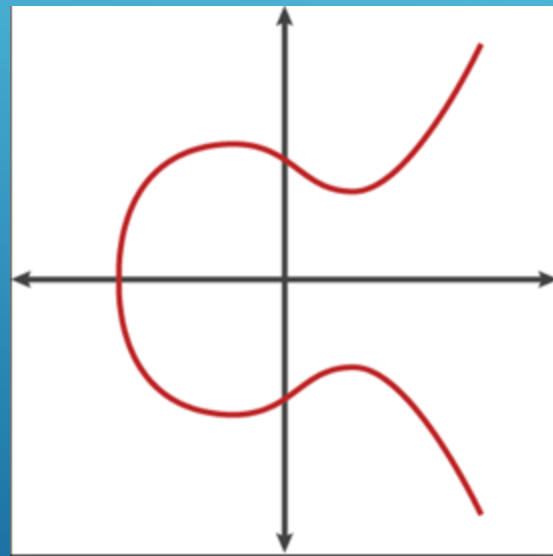


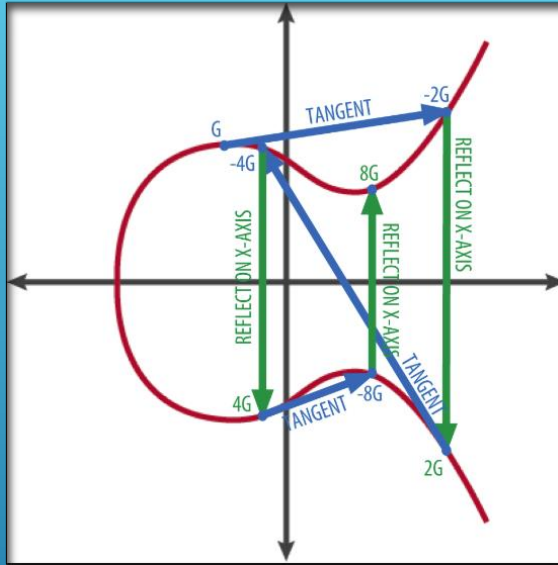
CURVAS ELIPTICAS

En matemáticas, las curvas elípticas se definen mediante ecuaciones cúbicas (de tercer grado). Han sido utilizadas para probar el último teorema de Fermat y en factorización de enteros. Se emplean también en criptografía de curvas elípticas. Estas curvas no son elipses.

Ej:

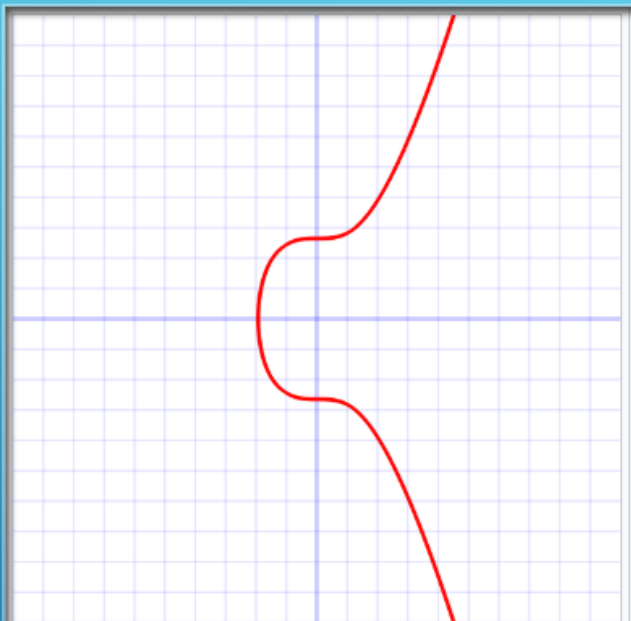
$$y^2 = x^3 - x + 1$$





- Llave Privada: la cantidad de iteraciones
- Llave Pública: posición X;Y del resultado final

¿COMO FUNCIONA?

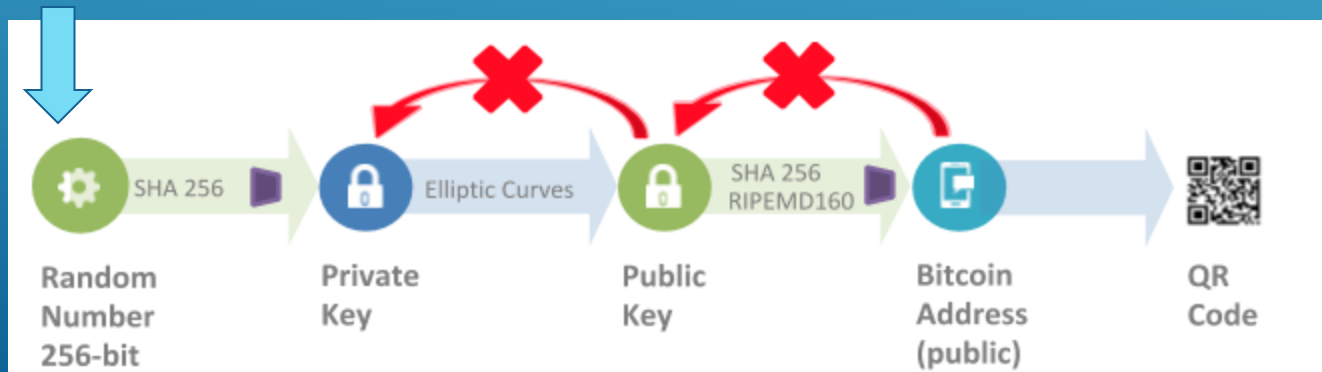


This is a graph of secp256k1's elliptic curve $y^2 = x^3 + 7$ over the real numbers. Note that because secp256k1 is actually defined over the field \mathbb{Z}_p , its graph will in reality look like random scattered points, not anything like this.

CUAL ES LA CURVA ELÍPTICA REALMENTE USADA

¿CÓMO OBTENER LOS 3 ELEMENTOS?

- ↓ PrivateKey = Hash(<algun_contenido>)
- ↓ PublicKey = CurvaEliptica(PrivateKey)
- ↓ Address = Hash(PublicKey)



PRIVATE KEY

La PrivateKey debe de ser un número dentro de los límites definidos como input validos en la función de curva elíptica Secp256k1 que es la función utilizada por la mayoría de las criptomonedas (incluyendo Bitcoin, Ethereum y Zcash).

Es un número entero cuyo valor maximo es el resultado de $2^{256} - 2^{32} - 2^9 - 2^8 - 2^7 - 2^6 - 2^4 - 1$ lo que da un numero de 77 cifras (o 32 bytes)

PUBLIC KEY

Luego de aplicar la función de curva elíptica tomando como parámetro la privateKey, vamos a obtener un resultado que es el punto de intersección (en coordenadas X;Y) para con la curva.

Tal output de la función tendrá 64 bytes de información.

Blockchain 101: History



- *May 11, 2008*: First concrete implementation by Homer Simpson.
- *October 31, 2008*: Satoshi Nakamoto releases computer-based version.

We will closely follow Bitcoin and Cryptocurrency Technologies by Narayanan, Bonneau, Felten, Miller, Goldfeder (NBFMG) for several lectures.

<http://bitcoinbook.cs.princeton.edu/>

MUCHAS GRACIAS!!

Several thin, white, parallel diagonal lines are positioned in the bottom right corner of the image, extending from the right edge towards the center.