Bitcoin: un sistema de dinero electrónico Peer-to-Peer.

1. Uso de Peer-to-Peer.

Una versión puramente peer-to-peer de dinero electrónico permitiría pagos online que se enviarán directamente de una parte a otra sin tener que pasar a través de una institución financiera. Las firmas digitales proporcionan parte de la solución, pero el principal beneficio se pierde si un tercero de confianza sigue siendo necesario para evitar doble-gasto (dinero que se ha gastado dos veces). Se propone una solución al problema de doble-gasto mediante una red peer-to-peer.

2. Introducción.

El *Bitcoin* reposa sobre la confianza de las matemáticas para su encriptación y en sus algoritmos que están escritos en código abierto. La forma en que está pensado *Bitcoin* es matemáticamente robusta. Se basa en los mismos algoritmos de encriptación que permiten las comunicaciones seguras y el comercio en internet.

Matemáticamente los algoritmos se basan en aritmética elemental: en la dificultad de factorizar un número grande. El algoritmo es el mismo que para una solución sencilla, pero a partir de cierta magnitud los ordenadores tampoco tienen la potencia suficiente para descomponer números grandes. El truco fundamental consiste en utilizar como clave pública un número que es el producto de dos números (primos) enormes, y cualquiera de ellos es la clave privada. A partir de la clave pública no es factible encontrar la clave privada por la dificultad del problema de factorización. Sin embargo, sin dar la clave privada hay formas de demostrar que se posee esta, lo cual permite firmar documentos, encriptarlos para que sólo el receptor los lea (usando su clave pública y nuestra clave privada, y él lo podrá desencriptar usando su clave privada y nuestra clave pública), y también demostrar que nosotros somos los proprietarios de un *Bitcoin* cuyó código se conoce públicamente.

Todo esto, desde el punto de vista de la confianza, es mucho más robusto que la confianza en cualquier institución financiera o bancaria que cómo bien se sabe no se basa en verdades matemáticas.

Bitcoin utiliza **ECDSA**, (Elliptic Curve Digital Signature Algorithm) que es una modificación del algoritmo DSA que emplea operaciones sobre puntos de curvas elípticas en lugar de las exponenciaciones que usa DSA (problema del logaritmo discreto). La principal ventaja de este esquema es que requiere números de tamaños menores para brindar la misma seguridad que DSA.

3.Direcciones.

Los usuarios de Bitcoin pueden tener múltiples direcciones, y de hecho pueden generar ilimitadas direcciones nuevas, debido a que generarlas es relativamente instantáneo. Equivale a generar un par de claves pública/privada, y no requiere ningún contacto con nodos de la red. Crear direcciones para un sólo propósito/uso puede ayudar a preservar el anonimato de un usuario.

Las direcciones son secuencias alfanuméricas aleatorias de 33 caracteres de largo, en formato legible para personas, un ejemplo de dirección sería 1LtU9rMsQ41rCqsJAvMtw89TA5XT2dW7f9.

Utilizan una codificación en Base 58, De esta forma, se componen únicamente de caracteres alfanuméricos que se distinguen entre sí en cualquier tipo de letra. Las direcciones Bitcoin también incluyen un checksum de 32 bits para detectar cambios accidentales en la secuencia de caracteres.

4.Codificación Base58Check.

Las direcciones Bitcoin se codifican mediante una forma modificada de la codificación Base58 a la que se conoce como Base58Check.

De manera general, la codificación Base58Check se utiliza para codificar secuencias de bytes utilizadas en Bitcoin convirtiéndolas en un formato de texto legible para el ser humano. Una dirección Bitcoin es simplemente una cadena de texto codificada como Base58Check que contiene unos datos útiles de 20 bytes de longitud, que consisten en el *hash* de la clave pública asociada con la dirección.

La codificación Base58Check resulta de eliminar los siguientes seis caracteres del sistema Base 64: 0 (cero), I (i mayúscula), O (o mayúscula), l (L minúscula), + (más) y / (barra). que parecen iguales en algunas fuentes y que podrían utilizarse para crear números de cuenta visualmente idénticos.

Se incluyen cuatro bytes (32 bits) de un código de comprobación de errores basado en SHA-256. Este código puede emplearse para detectar automáticamente e incluso corregir errores tipográficos.

Un byte de información de versión/aplicación. Las direcciones Bitcoin tradicionales utilizan 0x00 para este byte. Las nuevas direcciones que permiten multifirma utilizan 0x05.

Cómo codificar una clave privada.

La codificación Base58Check se utiliza también para codificar claves privadas en el formato de importación de monedero. Este se forma exactamente igual que una dirección Bitcoin, excepto en que se utiliza 0x80 como byte de versión/aplicación y los datos útiles constan de 32 bytes en lugar de 20 (una clave privada en Bitcoin es un único entero de 32 bytes en formato *big-endian*). Este tipo de codificación da lugar siempre a una cadena de 51 caracteres que comienza por '5', o más concretamente '5H', '5J', o '5K'.

Tabla de símbolos Base58.

Valor	Carácter	Valor	Carácter	Valor	Carácter	Valor	Carácter
0	1	1	2	2	3	3	4
4	5	5	6	6	7	7	8
8	9	9	A	10	B	11	C
12	D	13	E	14	F	15	G
16	H	17	J	18	K	19	L
20	M	21	N	22	P	23	Q
24	R	25	S	26	T	27	U
28	V	29	W	30	X	31	Y
32	Z	33	a	34	b	35	c
36	d	37	e	38	f	39	g
40	h	41	i	42	j	43	k
44	m	45	n	46	o	47	p
48	q	49	r	50	s	51	t
52	u	53	v	54	w	55	x
56	y	57	z				

El algoritmo para codificar address_byte_string (consistente en 0x01 + hash + 4-byte_check_code) es:

```
code_string = "123456789ABCDEFGHJKLMNPQRSTUVWXYZabcdefghijkmnopqrstuvwxyz"
x = convert_bytes_to_big_integer(hash_result)

output_string = ""

while(x > 0){
    (x, remainder) = divide(x, 58)
    output_string.append(code_string[remainder])
}

repeat(number_of_leading_zero_bytes_in_hash){
    output_string.append(code_string[0]);
}
output_string.reverse();
```

5.Hash.

Un hash de un objeto (una cadena de texto, un número o cualquier cosa que se pueda representar en bits) es el equivalente de nuestra huella dactilar. Es una **identificación única y constante**. Dos objetos distintos tienen (teóricamente) *hashes* distintos. Además, tiene la peculiaridad de que es una función “de una vía”. Es decir, si tienes el objeto es muy fácil obtener su *hash*. Sin embargo, si tienes el *hash* es extremadamente difícil obtener el objeto original del que proviene. En el caso de Bitcoin, el algoritmo es SHA256.

6.Transacciones y bloques, los pilares de Bitcoin.

Definimos un monedero electrónico como una cadena de firmas digitales. Cada propietario transfiere la moneda al siguiente por la firma digital de un hash de la transacción anterior y la clave pública del siguiente propietario y la adición de estos hasta el final de la moneda. Un beneficiario puede verificar las firmas para comprobar la cadena de propiedad.

Los Bitcoins contienen la dirección pública de su dueño actual. Cuando un usuario *A* transfiere algo a un usuario *B*, *A* entrega la propiedad agregando la clave pública de *B* y después firmando con su clave privada. *A* entonces incluye esos Bitcoins en una *transacción*, y la difunde a los nodos de la red peer-to-peer a los que está conectado. Estos nodos validan las firmas criptográficas y el valor de la transacción antes de aceptarla y retransmitirla. Este procedimiento propaga la transacción de manera indefinida hasta alcanzar a todos los nodos de la red peer-to-peer.

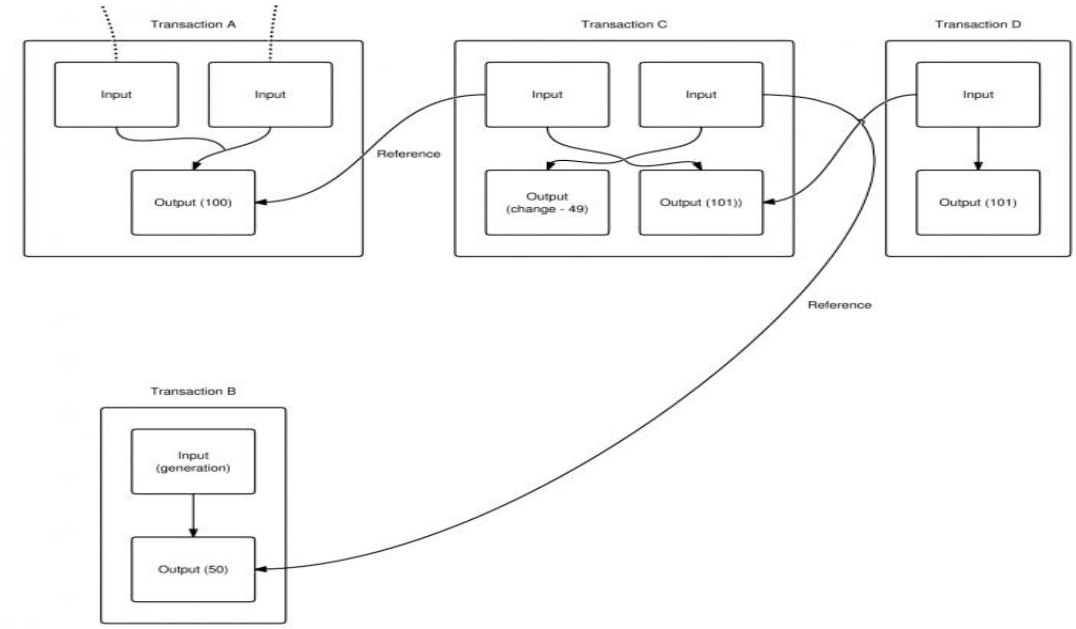

Imagen1:Transacción: A envía 100 BTC a C y C genera 50 BTC. C envía 101 BTC a D, y necesita enviarse a sí mismo el cambio. D envía los 101 BTC a otra persona, que aún no los ha gastado (no sale ninguna flecha de su output). Solamente la salida de D (101 BTC) y el cambio que volvió a C (49 BTC) son cantidades disponibles para gastar.

Bitcoin se basa en dos pilares fundamentales: uno es la **transacción**, y otro el **bloque**. La transacción es el envío de Bitcoins de un usuario a otro.

Una transacción tiene entradas y salidas: de donde viene el dinero y a dónde va. Su ID (identificación) es un *hash* combinado del ID de las entradas y del ID del destinatario (su clave pública). Así se fija de forma inequívoca quién es el destinatario y de dónde han salido las monedas. Después, ese ID se firma con la clave privada del emisor de la transferencia, quedando certificado que el dinero lo ha transferido su propietario.

Las transferencias se agrupan en bloques. Cada bloque tiene, además, un sello de tiempo, **un número de verificación (Nonce)** y el ID del bloque anterior. De esta forma, se genera una cadena de bloques, que contiene toda la historia de transferencias de Bitcoins.

Los bloques los generan los *mineros*, y antes de crearlos verifican la validez de todas las transferencias (es decir, que un usuario no haya gastado dinero que ya había transferido). Una transferencia que se ha quedado fuera de la cadena de bloques no es válida, y del mismo modo una transferencia dentro de la cadena se considera válida sin más operaciones.

Hash: 000000000043a8c0fd1d6f726790caa2a406010d19efd2780db27bdbbd93baf6
Previous block: 00000000001937917bd2caba204bb1aa530ec1de9d0f6736e5d85d96da9c8bba
Next block: 00000000000036312a44ab7711afa46f475913fbd9727cf508ed4af3bc933d16
Time: 2010-09-16 05:03:47
Difficulty: 712.884864
Transactions: 2
Total BTC: 100
Size: 373 bytes
Merkle root: 8fb300e3fdb6f30a4c67233b997f99fdd518b968b9a3fd65857bfe78b2600719
Nonce: 1462756097

Input/Previous Output	Source & Amount	Recipient & Amount
N/A	Generation: 50 + 0 total fees	Generation: 50 + 0 total fees
f5d8ee39a430...:0	1JBSCVF6VM6QjFZyTnbpLjoCJ...: 50	16ro3Jptwo4asSevZnsRX6vf..: 50

Imagen 2: Ejemplo de bloque de Bitcoin. El bloque contiene 2 transacciones, una de las cuales otorga el par del generador con 50 BTC.

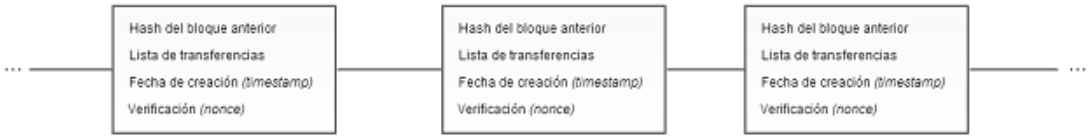

Imagen 3:Cadena de bloques de bitcoin.

7.La creación de la cadena de bloques: proof-of-work.

Todos los nodos que forman parte de la red Bitcoin mantienen una lista colectiva de todas las transacciones conocidas, la *cadena de bloques*. Los nodos generadores, también llamados *mineros*, crean los nuevos bloques, añadiendo en cada uno de ellos el hash del último bloque de la cadena más larga de la que tienen conocimiento, así como las nuevas transacciones publicadas en la red. Cuando un minero encuentra un nuevo bloque, lo transmite al resto de los nodos a los que está conectado. En el caso de que resulte un bloque válido, estos nodos lo agregan a la cadena y lo vuelven a retransmitir. Este proceso se repite indefinidamente hasta que el bloque ha alcanzado todos los nodos de la red. Eventualmente, la cadena de bloques contiene el historial de posesión de todas las monedas desde la dirección creadora a la dirección del actual dueño. Por lo tanto, si un usuario intenta reutilizar monedas que ya usó, la red rechazará la transacción.

El problema surge cuando un nodo malicioso intenta crear un bloque con una transferencia inválida (con dinero que se ha gastado dos veces) y después generar más bloques de forma masiva. Al generar bloques válidos rápidamente, la transferencia inválida queda enterrada en la cadena. Cuando el resto de los nodos reciban esta cadena, que será la más larga de todo el entorno, verificarán como mucho los últimos bloques, darán la rama como válida y esa transacción inválida pasará desapercibida**.**

Por lo tanto, hay que implementar un método que evite que se puedan generar bloques de forma indiscriminada, algo que obligue a los nodos a invertir tiempo en generar el bloque. Esto se conoce como *proof-of-work*. Lo cual consiste en encontrar el número de verificación del bloque (**Nonce**), de tal forma que el *hash* del bloque sea menor que un determinado valor , esto se hace para retardar la generación de un bloque a 10 minutos.

La red reajusta la dificultad cada 2016 bloques, es decir, aproximadamente cada 2 semanas, para que un bloque sea generado cada diez minutos. La cantidad de Bitcoins creada por bloque nunca es más de 50 BTC, y los premios

están programados para disminuir con el paso del tiempo hasta llegar a cero, garantizando que no puedan existir más de 21 millones de BTC.

Así, se consigue que el tiempo de generación de un bloque se mantenga estable a lo largo del tiempo independientemente de la capacidad de proceso de los nodos.

Esos 10 minutos que se tarda en verificar cada bloque son la barrera que impide a los atacantes tomar el control de Bitcoin. De esta forma, no pueden generar bloques rápidamente para ocultar transacciones y gastar dinero dos veces.

¿Cómo se almacena toda la historia de transacciones?

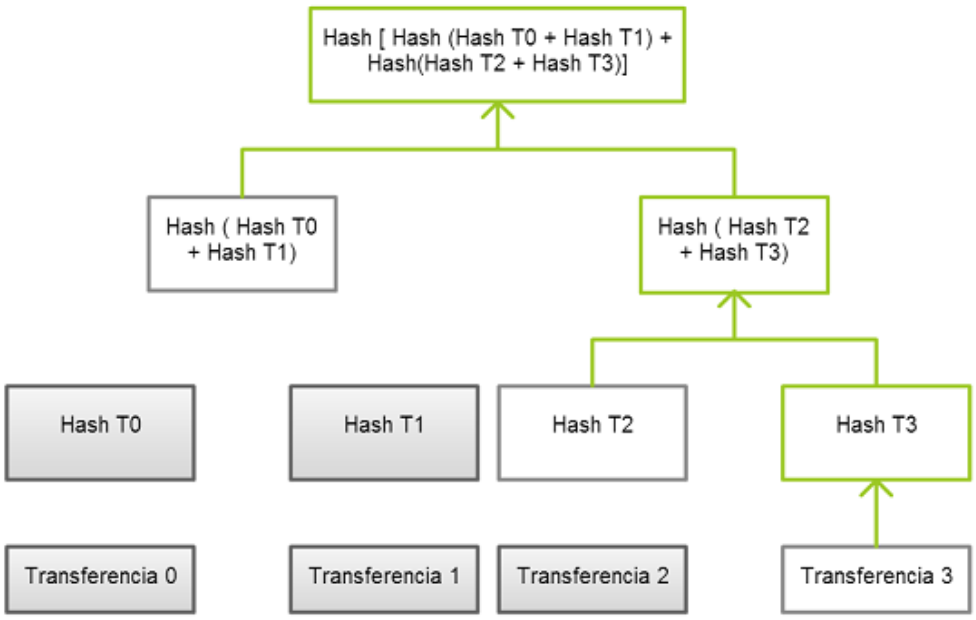

Imagen 4: Funcionamiento de un árbol hash. Los recuadros verdes son los hashes generados, los grises los que se guardan y los sombreados no se usan por lo tanto no se guardan.

Otro problema de Bitcoin es el espacio de almacenamiento. ¿Cómo se puede guardar toda la cadena de bloques (que es considerablemente larga) sin desperdiciar espacio en disco?

La solución es usar un árbol de *hashes* o un árbol Merkle. Los hashes de las transferencias se van combinando dos a dos en forma de árbol binario (cada nodo tiene dos hijos), como se ve en la figura. Así, cuando no se necesitan dos hermanos (dos nodos que comparten el mismo padre), se pueden borrar y quedarse con el nodo padre sin perder la posibilidad de verificar el resto de nodos del árbol. Esto reduce considerablemente el espacio necesario para almacenar toda la historia, y de hecho permite quedarse sólo con las transferencias más recientes y olvidarse del resto.

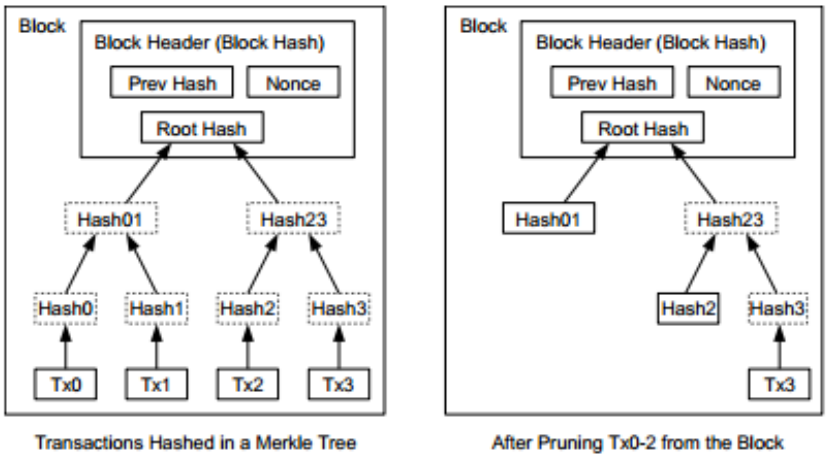

Imagen 5: Árbol de *hashes* o árbol Merkle.

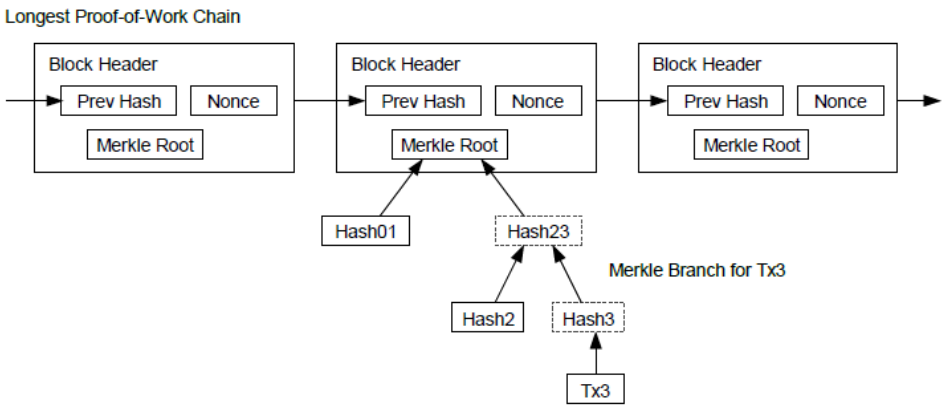

Imagen 6: cadena de bloques: proof-of-work.