La criptografía detrás de Bitcoin

Javier Aguilar Martín y Rafael González López

31 de marzo de 2018

Resumen

En este trabajo vamos a explorar el funcionamiento interno de Bitcoin, la criptomoneda más famosa y más valorada. Para ello, empezaremos presentando informalmente qué es Bitcoin y explicando los conceptos criptográficos previos que serán necesarios para entender el resto del trabajo. A continuación entraremos en detalle en los algoritmos usados por Bitcoin para poder considerarse funcional y seguro. Por último comentaremos algunas otras aplicaciones de las ideas que subyacen a esta moneda.

Índice

1. Preliminares

1.1. ¿Qué es Bitcoin?

Bitcoin (abreviado BTC) es una criptomoneda o criptodivisa, es decir, un medio de intercambio digital que usa las herramientas de la criptografía para garantizar la seguridad y el anonimato. Bitcoin no es la única criptomoneda existente, pero sí que fue la primera que empezó a operar en el año 2009. Algo que diferencia a esta moneda del dinero fiduciario (dinero usual) es su descentralización, ya que no existe un Banco Central de Bitcoin. Por ello no se puede manipular la cantidad total de Bitcoin existente, que está previamente fijada y es pública. Bitcoin a su vez se puede dividir en unidades menores hasta una fracción de 10^{-8} BTC llamada Satoshi (en honor a su creador, Satoshi Nakamoto).

Actualmente es reconocida como legal en varios países como Alemania o Japón, mientras que en otros como Bolivia está explíticamente prohibido. En muchos otros existe un vacío legal. Su valor a día de hoy alcanza los 5700€/BTC, lo cual lo convierte en un poderoso activo financiero.

Se puede conseguir este tipo de dinero intercambiándolo por bienes y servicios (como cualquier otro dinero) a través de un monedero o *wallet*, o bien obtenerlos directamente *minando* (por analogía al oro). Los procesos de intercambio y obtención de Bitcoin serán explicados más detalladamente en secciones posteriores. Antes de eso, aclararemos algunos conceptos que serán utilizados más adelante.

1.2. Funciones hash.

Definición 1.1.

Definición 1.2. Para un n fijo, llamamos **función hash** a una función $h : \mathcal{M} \to (\mathbb{F}_2)^n$ tal que:

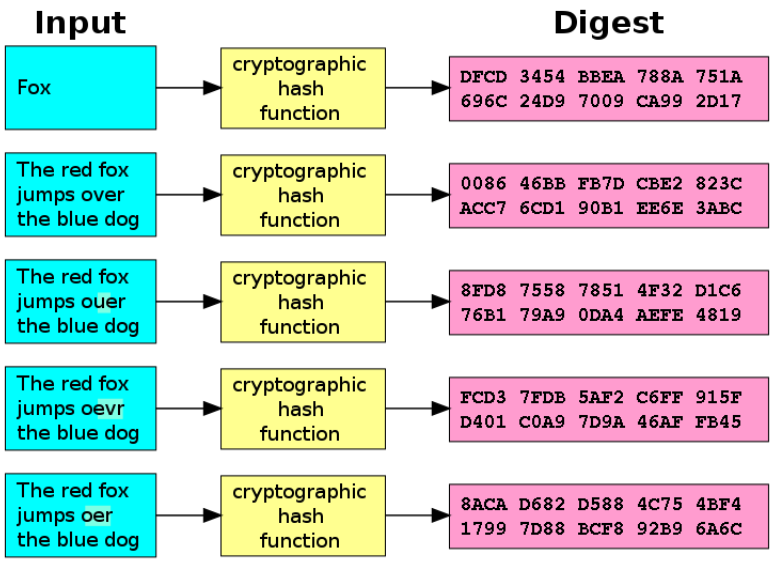
- h es resistente a preimagen: Dado $y = h(x)$ es computacionalmente infactible calcular x (es *one way*).
- h es resistente a colisiones: Es computacionalmente infactible calcular x y x' con $h(x) = h(x')$
- h es resistente a segunda preimagen: Dado $y = h(x)$, es computacionalmente infactible calcular x' con $h(x') = y$.

Al texto recibido se le suele llamar **input** o **mensaje**, mientras que el texto devuelto se puede llamar **output** o **digest**. Normalmente el digest se expresa en hexadecimal con 40 caracteres.

Algunos ejemplos de funciones hash son SHA1, CRC32 o SHA256. Uno podría preguntarse con qué frecuencia se producen colisiones, es decir, que dos entradas de texto tengan un mismo hash. En el caso del SHA1, si tenemos $1,71 \times 10^{17}$ mensajes, ¿cuál es la probabilidad de que tengamos una colisión? Tan solo 1 entre 100 billones, es decir, la misma que tu casa sea destruida por un impacto de meteorito.

Estas funciones tienen la propiedad de que un pequeño cambio en el mensaje recibido genera una salida totalmente distinta, lo que las vuelve impredecibles, como podemos observar en el siguiente ejemplo.

Ejemplo 1.3. En este ejemplo vemos el funcionamiento de la función SHA1.



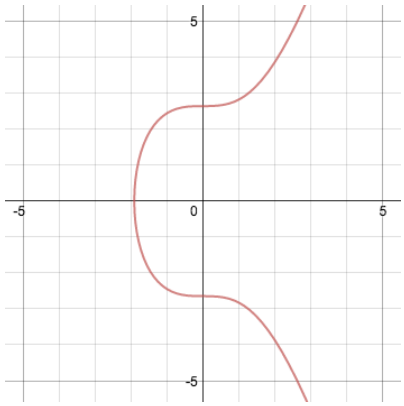
No todas las funciones hash son seguras[1], pero la que usa Bitcoin, SHA256, es una de las más seguras que existen.

[1] SHA1 es un ejemplo de esto https://shattered.io/

1.3. Curvas elípticas.

Definición 1.4. Una **curva elíptica** es una curva plana representada algebraicamente como una ecuación de la forma $y^2 = x^3 + ax + b$.

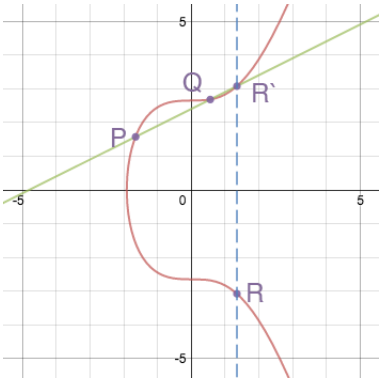
Ejemplo 1.5. Para $a = 0$ y $b = 7$, que es la que usa Bitcoin, tiene esta forma:



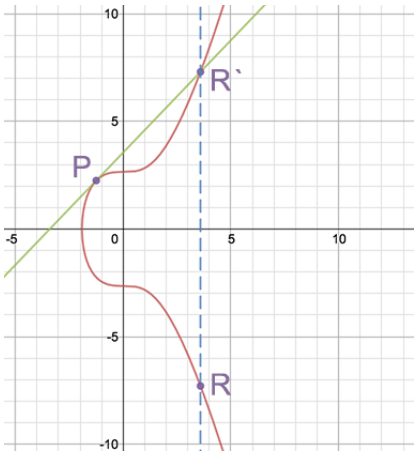
Los puntos de una curva elíptica tiene la propiedad de que forman un grupo si definimos una suma particular que vamos a ver a continuación.

1.3.1. Suma en una curva elípitca.

Para sumar dos puntos P y Q pertenecientes a la curva elíptica, en primer lugar tenemos que trazar la recta que pasa por dichos puntos y hallar la intersección de esta recta con la curva elíptica, al que llamaremos R'. Después debemos hacer una reflexión del punto de intersección con respecto al eje de abscisas (obsérvese que las curvas elípticas son simétricas con respecto a este eje). El punto obtenido es $R = P + Q$. Lo vemos más claramente en la siguiente imagen:



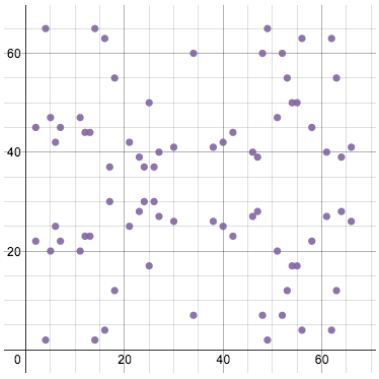
En el caso de que queramos sumar un punto P consigo mismo debemos hacer la operación anterior considerando la recta tangente a la curva y en P como se observa en la siguiente imagen:



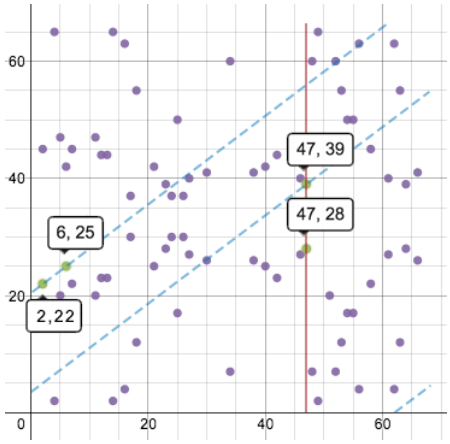
Como elemento neutro deberemos considerar al punto del infinito de la curva. Se puede observar que el inverso de un punto sería su simétrico con respecto al eje de abscisas. De forma natural se define el producto por escalar αP siempre que $\alpha \in \mathbb{Z}$ como la suma de α veces P (o de su inverso si $\alpha < 0$). Estas propiedades dotan a las curvas elípticas estructura de grupo.

1.3.2. Curvas elípticas sobre cuerpos finitos.

Realmente Bitcoin no usa la curva anterior sobre $\mathbb{R}$ tal como la hemos dibujado, sino sobre un cuerpo finito módulo $2^{256}–2^{32}–2^9–2^8–2^7–2^6–2^4 – 1$, que es primo. Además se selecciona un punto base para hacer las operaciones que tiene como orden un primo lo suficientemente grande. Tanto estos parámetros como la curva son los mismos para todos los usuarios. Por visualizar qué aspecto tiene una curva elíptica sobre un cuerpo finito, observémos este ejemplo en el que se toma como primo el número 67:



A pesar de lo diferente que se ve, conserva las propiedades esenciales del caso anterior. Además sigue siendo simétrica. La suma también cambia visualmente como podemos ver en esta imagen en la que sumamos $(2, 22) + (6, 25)$:



Con esto terminamos los preliminares y pasamos a explicar cómo funciona Bitcoin.

2. Funcionamiento interno de Bitcoin.

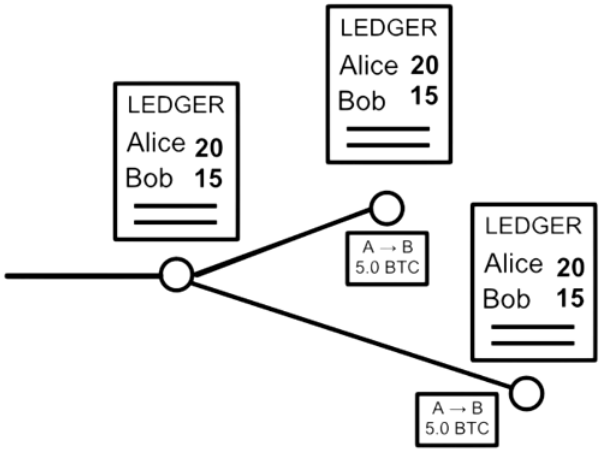
2.1. Transacciones, validación y firma.

Comencemos precisamente el concepto de transacción, la operación fundamental en el funcionamiento de Bitcoin.

Definición 2.1. Una **transacción** es un movimiento de Bitcoin de una direccion de origen a una dirección de destino.

Cada dirección de Bitcoin representa una clave pública. Para gastar Bitcoin es necesario conocer la clave privada asociada a la clave publica que contenga un saldo en Bitcoins. Entonces, se pueden gastar esos Bitcoins, es decir, transferirlos a otra dirección, firmando digitalmente con la clave privada la transmisión de esta información y enviando la nueva transacción a toda la red, la cuál está compuesta por nodos.

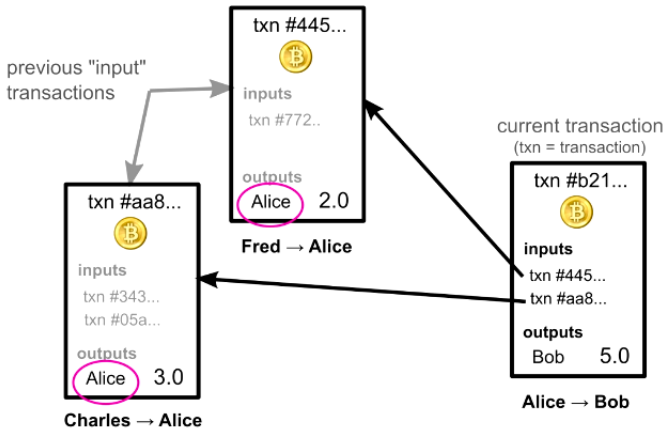
Definición 2.2. Los **nodos** de Bitcoin son las entidades que forman parte del protocolo distribuido para crear un registro común de información. Cada vez que se realiza una transacción, esta se envía a todos los nodos. Por ello, tienen la tarea de validar, afianzar, transmitir y almacenar. Esto se hará con ayuda de una base de datos llamada *blockchain*, que veremos más adelante.



Ahora vamos a ver detalladamente en qué consiste una transacción.

Supongamos que Alice quiere enviar 5 BTC a Bob. Para ello, Alice debe probar que tiene ese dinero y para ello el total de transacciones que haya realizado anteriormente debe sumar al menos esos 5 Bitcoins. Las transacciones usadas como referencia se llaman *inputs*. Los nodos que vayan a

verificar la transacción deben comprobar los inputs para asegurar que Alice era la receptora y que la suma asciende al menos a la cantidad que desea enviar.



Cada input debe usarse completamente, por lo que si se desea enviar una cantidad que no coincide exactamente con ninguna suma de transacciones previas se devolverá la "vuelta" como un segundo output además del dinero enviado a Bob.

A través de estas referencias a transacciones pasadas, la propiedad de Bitcoins se basa en una cadena de transacciones (veremos más adelante que es la cadena de bloques). Al instalar por primera vez un monedero de Bitcoin, el software comprueba automáticamente todas las transacciones anteriores para asegurarse de tener un registro fidedigno.

Una vez que una transacción ha sido usada como referencia se considera gastada y no puede volverse a usar. De otro modo podría cometer **doble gasto** aludiendo a la misma transacción varias veces. Por lo tanto, al validar una transacción también se comprueba que las transacciones no habían sido usadas antes. Debido a que las transacciones en la red de Bitcoin se cuentan por decenas de millones, se aligera el proceso mediante un índice de transacciones no usadas. En definitiva, tener Bitcoins significa que hay una lista de transacciones que apuntan a tu dirección y que no han sido gastadas.

Ahora que hemos dejado clara la idea general de cómo se produce una transacción, volvamos a la operación que quería realizar Alice. Sean (PK_A, SK_A) y (PK_B, SK_B) los pares de calve pública-privada de Alice y Bob respectivamente. La función $Add(PK)$ nos devuelve la dirección generada a partir de la clave pública PK aplicándole una función hash y añadiéndole delante un 1 o un 3 (esto se hace simplemente para indentificarlo como dirección pública de Bitcoin). Además será necesaria una firma digital, que explicaremos posteriormente, y que denotaremos $Sign_{SK}(m)$ donde m representa la transacción (ya que esta firma varía en cada transacción). Sea H una función hash. Supongamos

que Alice ya había recibido la cantidad de 5 BTC en una transacción T_0 a su dirección, $Add(PK_A)$. Como hemos explicado antes, $T_0 = \{input_0, output_0\}$. En este caso $input_0$ sería el registro anterior de transacciones y $output_0 = \{Add(PK_A), 5\}$.

Para enviar el dinero a Bob, Alice necesita crear una nueva transacción T_1 de la siguiente forma:

$$T_1 = \{input_1, output_1\}$$
$$input_1 = \{H(T_0), Sign_{SK_A}(T_0 + input), PK_A\}$$
$$output_1 = \{Add(PK_B), 5\}$$

Veamos por qué aparece cada uno de los términos en el $input_1$. En primer lugar, $H(T_0)$ es un hash de la transacción y actúa como puntero en la cadena de bloques como se verá en secciones posteriores. Después, se genera una nueva firma digital que solo se puede realizar con la clave privada SK_A que solo posee Alice. Por último es necesario que aparezca la clave pública de Alice para poder validar la transacción y comprobar que efectivamente era dinero proveniente de su dirección.

En el $output_1$ simplemente se indica a quién va dirigida la cantidad que se especifica, de forma que solo Bob podrá gastar esta cantidad en transacciones posteriores. Bob puede verificar que le han sido transferidos los fondos comprobando que $Add(PK_A)$ coincide con la dirección de destino de T_0 y también que la firma es correcta $Sign_{SK_A}(T_0 + input)$ usando PK_A.

Bitcoin permite realizar operaciones más complejas, como transacciones que requieren varias claves privadas, pero la base de todo es lo que hemos explicado. Nos queda pendiente mostrar cómo se calcula la firma digital.

2.1.1. Algoritmo de firma digital en curvas elípticas.

Para poder firmar primero se necesita tener las claves. El procedimiento por el cual se generan es ElGamal sobre una curva elíptica, que funciona de la misma forma que en cualquier otro grupo, pero con las operaciones explicadas en la primera sección. Por ello no vamos a detallar el proceso, pero sí vamos a señalar sus pasos.

En primer lugar se fija un **punto base** en la curva, llamémoslo P de orden N un primo bastante grande. Se elige como clave privada un número aleatorio $x \in \{1, \ldots, N-1\}$. A partir de este, claculamos la clave pública $Y = xP$. Esto indica que el número máximo de claves públicas (y por tanto de direcciones) es justamente el orden de P.

Debido a la complejidad computacional de la suma en curvas elípticas, es factible clacular la clave pública a partir de la clave privada y el punto base, pero es infactible calcular la clave privada a partir de la clave pública y el punto base.

En Bitcoin, tanto el punto base como su orden se representan como un *string* hexadecimal. Concretamente:

- Punto base: B04 79BE667E F9DCBBAC 55A06295 CE870B07 029BFCDB 2DCE28D9 59F2815B 16F81798 483ADA77 26A3C465 5DA4FBFC 0E1108A8 FD17B448 A6855419 9C47D08F FB10D4B8
- Orden: FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE BAAEDCE6 AF48A03B BFD25E8C D0364141

Así pues, las claves también se representarán de este modo. Ahora que ya tenemos las claves, podemos empezar a firmar. Sea pues P el punto base, N su orden, x la clave privada, m un mensaje que vamos a firmar y h una función hash que se usa para que el mensaje tenga el mísmo número de bits (256) que el orden de P. El algoritmo de la firma digital consiste en lo siguiente:

1. Calcular $e = h(m)$.
2. Sea z el conjunto de l dígitos más a la izquierda de e, donde l es la longitud en bits de N.
3. Elegir un número entero aleatorio $k \in \{1, \ldots, N-1\}$.
4. Calcular el punto $(a, b) = kP$ usando la multiplicación por escalar de la curva elíptica.
5. Encontrar $r \equiv a \mod N$. Si $r = 0$, vuelta al paso 1.
6. Encontrar $s \equiv (z + rx)k^{-1} \mod N$. Si $s = 0$, vuelta al paso 1.
7. La firma es el par (r, s).

Es importante que k sea diferente en cada firma para evitar la posibilidad de extraer la clave privada[2], ya que el resto de datos son conocidos. Dadas dos firmas (r, s) y (r, s') en las que se ha utilizado la misma k para dos mensajes conocidos m y m', un atacante puede utilizar la ecuación en el paso 6 para calcular k, ya que $s - s' \equiv k^{-1}(z - z') \mod N$. El atacante puede hallar entonces $k = \frac{z-z'}{s-s'} \mod N$. Como $s = k^{-1}(z+rx) \mod N$, ahora puede calcular la clave de privada $x = \frac{sk-z}{r}$ mód N.

Al igual que las claves, la firma se suele expresar en hexadecimal, pero a la hora de hacer los cálculos deben ser expresados como enteros. Para ver ejemplos consultar [4].

[2]Un error de este tipo fue usado en 2010 para extraer las claves de PS3 https://nakedsecurity.sophos.com/2012/10/25/sony-ps3-hacked-for-good-master-keys-revealed/

Llegado a este punto tenemos unos datos y una firma. Otra persona que tenga nuestra clave pública puede recibir los datos y la la clave pública y verificar que somos nosotros quienes lo enviamos. Veamos cómo se consigue.

Sea Y la clave pública y el resto de variables definidas como antes. En primer lugar se debe verificar que la clave pública es válida de la siguiente forma:

1. Comprobar que Y no es el punto base.
2. Comprobar que Y es un punto de la curva.
3. Comprobar que el orden de Y es N.

Una vez hecho esto, los pasos para verificar la firma son los siguientes:

1. Verificar que $1 \leq r, s \leq N - 1$. En caso contrario la firma no es válida.
2. Calcular $e = h(m)$ con la misma función hash de antes.
3. Tomar z como los l bits más a la izquierda de e del mismo modo que antes.
4. Calcular $w \equiv s^{-1} \mod N$.
5. Calcular $u \equiv zw \mod N$ y $v \equiv rw \mod N$.
6. Calcular el punto $(a, b) = uP + vY$.
7. Verificar que $r \equiv a \mod N$. La firma es inválida en caso contrario.

No es evidente que esta verificación realmente funcione, pero podemos probarlo. Denotemos $C = uP + vY$. Como $Y = xP$ obtenemos

$$C = (u + vx)P.$$

Sustituyendo los valores de u y v, y luego el de w:

$$C = (zw + rwx)P = (z + rx)s^{-1}P.$$

Introducimos ahora la definición de s:

$$C = (z + rx)(z + rx)^{-1}(k^{-1})^{-1}P = kP.$$

Por la definiciíon de r esto constituye la verificación. Por ello solo mensajes correctos serán verificados correctamente.

2.2. Blockchain, afianzamiento y minería.

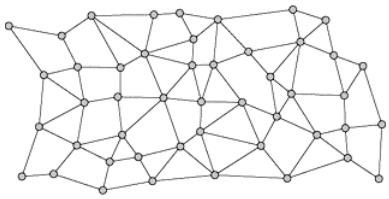
Comenzamos esta sección definiendo un concepto al que nos hemos referido varias veces y que es fundamental tanto en Bitcoin como en otras aplicaciones que han ido surgiendo a raíz del éxito de esta moneda.

Definición 2.3. Definimos la **cadena de bloques** o **blockchain** como el registro de contabilidad de Bitcoin, es decir, como el "libro" donde están apuntadas todas las transacciones desde que empezó a operar en 2009. Este registro único se genera, comparte y almacena de forma distribuida, de modo que todos los nodos están de acuerdo en su contenido sin la posibilidad de ser intervenido de ninguna forma.

Este registro público descentralizado es esencial y es lo que diferenció a Bitcoin de todas las monedas virtuales anteriores. Posee infinidad de aplicaciones que todavía se van descubriendo más allá de las criptodivisas y se cree que es una tecnología que cambiará el mundo. Esta cadena solo permite añadir información -no se puede quitar ni modificar nada-.

Los nuevos **bloques** que se van añadiendo contienen todas las transacciones que se han hecho desde que se añadió el bloque anterior. Cada bloque tiene un puntero que apunta al bloque anterior, formando así una cadena, y un valor de *nonce* (número aleatorio o pseudo-aleatorio de un solo uso) para crear bloques nuevos. ¿Pero cómo se generan estos bloques? Gracias a los mineros.

Definición 2.4. Los **mineros** son los usuarios que se dedican a validar las transacciones mediante una red P2P (abreviatura de *peer-to-peer*, esto es, una red en la que todos los nodos tienen la misma importancia) y creando nuevos bloques con las transacciones validadas. Cualquier usuario puede ser minero y por cada bloque generado se recibe un premio en forma de bitcoins. El nombre es por analogía con el oro.



La red P2P es una red distribuida, ni centralizada ni descentralizada.

Los Bitcoins se crean a partir de un tipo de transacción especial, la transacción de generación, que se incluye en cada bloque. Dicha transacción tiene una dirección de destino (que pertenece al minero que se ha generado el bloque) pero no tiene ninguna dirección de origen.

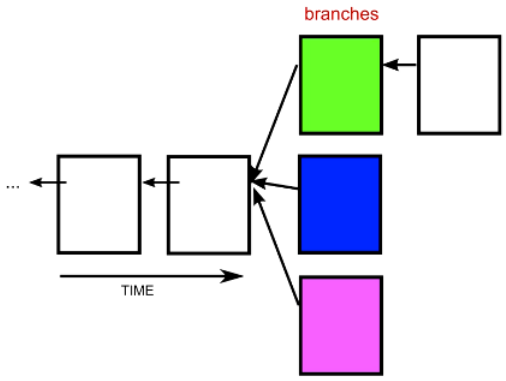
Actualmente, pues como veremos esta cantidad va decreciendo, se premia 12,5 BTC al minero o pool (agrupación de mineros que trabaja en conjunto) que resuelve el bloque. Más adelante explicaremos en profundidad en qué consiste la minería.

Como hemos visto antes, el funcionamiento básico de Bitcoin se basa en las transacciones. Estas se envían de una dirección a otra a través de los nodos, que son los encargados de validar -mediante los mineros- las transacciones y almacenarlas en la blockchain. Ahora explicaremos todas estas funciones de forma detenida.

Además de validar las transacciones, es necesario crear los bloques que las afianzan. Este proceso es el que permite construir el registro común único -la cadena de bloques- y se realiza de manera totalmente distribuida en Bitcoin. El afianzamiento en Bitcoin se basa en una **prueba de trabajo** (proof-of-work).

Definición 2.5. Un **sistema de prueba de trabajo** o **POW** es un sistema que requiere que el cliente del servicio realice algún tipo de trabajo que tenga cierto coste y que es verificado fácilmente en la parte del servidor. Normalmente el trabajo consiste en realizar un cómputo en el ordenador del cliente. En el caso de Bitcoin, el sistema de prueba de trabajo permite la transferencia de valor de manera directa entre los participantes de una transacción sin necesidad de depender de ninguna organización central de confianza.

Esta validación puede ocurrir simultáneamente con varios bloques, en cuyo caso se formaría una rama y se continúa a partir del bloque que llega primero.



¿En qué consiste dicha prueba? La prueba de trabajo que utiliza Bitcoin consiste en encontrar un valor de nonce para el nuevo bloque de tal manera que el hash del bloque sea inferior a un valor objetivo fijado. Si H es la función hash, lo que un minero es buscar un valor de un nonce tal que:

$$H(H(Data + Nonce)) < Dif$$

La dificultad es un valor que se ajusta en función de la capacidad computacional de la red. Por las propiedades de las funciones hash, la unica manera de conseguir un hash inferior al valor objetivo es ir probando diferentes valores de nonce, hasta dar con uno que genere el hash buscado. De este modo, si un atacante quiere modificar la cadena de bloques, ya sea para anular transacciones, ya sea para tener control de lo que se anota en el registro común, necesitará un poder computacional superior al 50 % de la red. La red computacional de Bitcoin es la más potente que existe en el mundo, por encima de Google o la NSA.

¿Y cómo varía la dificultad del problema? Como se busca un nonce tal que $H(H(Data + Nonce)) < Dif$, bastará con fijar un valor de Dif cada vez más pequeño. En la representación hexadecimal del hash (o en otra cualquiera) esto se consigue eligiendo una cadena de caracteres que empiece por una determinada cantidad de ceros. Cuantos más ceros, más pequeño es el valor, por lo que a grandes rasgos se trata de buscar el valor del nonce que te dé un hash con la cantidad buscada de ceros iniciales.

Ejemplo 2.6. Vamos a ver un ejemplo simplificado de cómo se construye un bloque. Para ello vamos a usar hashes reducidos, en lugar de los reales de 40 caracteres. Supongamos que el valor del hash del anterior bloque es $00000000000001adf44c7d69767585$ (comienza con 13 ceros). Consideremos dos hashes de transacciones esperando a ser incluidas en el bloque $5572eca4dd4$ y $db7d0c0b845$. Por último sea el hash de la transacción por la cual el minero intenta recibir recompensa $916d849af76$. El nuevo bloqué tendrá todos los hashes mencionados de la siguiente forma:

$$00000000000001adf44c7d69767585 - -5572eca4dd4 - db7d0c0b845 - 916d849af76 - -$$

Solo falta completarlo con el valor adecuado de nonce. Por ejemplo empezamos con el valor $nonce = 1$, con el que resultaría el bloque

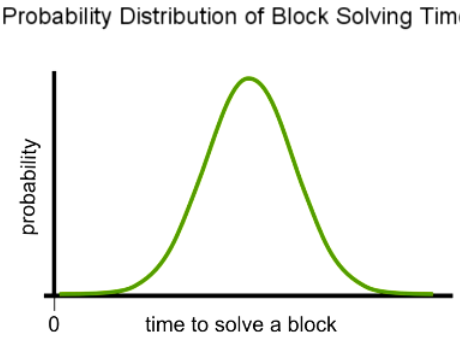
$$00000000000001adf44c7d69767585 - -5572eca4dd4 - db7d0c0b845 - 916d849af76 - -1$$

Al aplicarle el hash nos da como resultado

$$8b9b994dcf57f8f90194d82e234b72ac$$

que no puede ser válido porque tiene menos de de 13 ceros (ninguno de hecho). Podemos continuar probando hasta llegar al nonce adecuado. Si somos los primeros en encontrarlo, se añadirá nuestro bloque y recibiremos la recompensa.

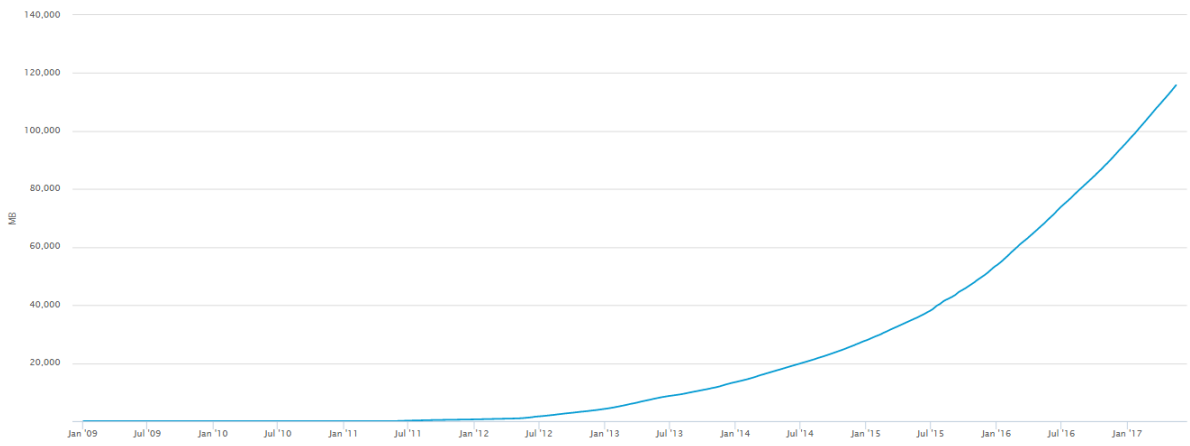
La gráfica RX 480 8GB GDDR5 permite minar a 29 MH/s.



La resolución de un bloque ocurre, más o menos, cada diez minutos.

2.3. Almacenamiento

Como hemos visto cuando hemos hablado acera de la blockchain, el almacenaje de la cadena de bloques se lleva a cabo con mucha redundancia pues todos los nodos completos de la red contienen una copia entera de la cadena de bloques (y sus transacciones). Esto permite a estos nodos validar de manera correcta cada nueva transacción. Tener que mantener una copia completa de la cadena puede suponer un problema para los nodos operando en dispositivos ligeros como, por ejemplo, dispositivos móviles. En Febrero de 2014, después de 5 años de operaciones con Bitcoin, la cadena de bloques ocupaba unos 13 GB. El día 17 de Mayo de 2017 ocupaba 115 GB. Aunque de momento no se ha implementado ninguna solución, se discute activamente la posibilidad de incorporar un algoritmo de poda de la cadena, de manera que no sea necesario guardar todas las transacciones. Así, transacciones antiguas podrían ser eliminadas, guardando de ellas solo su hash, para preservar la integridad de la cadena.



Evolución del peso en MB de la blockchain de BTC.

2.4. Halving

El sistema de Bitcoin ha sido diseñado de modo que la recompensa por minar se reduce a la mitad cada 210000 bloques. Este proceso se conoce como **Halving** o **Halvening**. Cuando el bloque 0 (conocido como *bloque génesis*) fue minado a principios de 2009 la recompensa fue de 50 BTC. El primer Halving tuvo lugar a finales de 2012, con lo que la recompensa fue de 25 BTC. El último Halvening se produjo en 2016, por lo que actualmente la recompensa es de 12,5 BTC.

En la introducción comentamos que la cantidad de Bitcoin que se podrá minar es fija (concretamente 21 millones de BTC), y aunque el motivo de trasfondo es puramente económico, está directamente relacionado con el Halving. La explicación se encuentra en la progresión geométrica que sigue la recompensa.

Definición 2.7. Una progresión geométrica es una sucesión construida de forma recursiva como:

$$x_0 = a \in \mathbb{R}$$
$$x_{n+1} = rx_n, r \in \mathbb{R}.$$

En el caso de las recompensas por minar Bitcoin, $r = \frac{1}{2}$ y $a = 50 \cdot 210000$ (el total de Bitcoins emitidos mientras la recompensa era de 50 BTC).

Proposición 2.8. La suma de una progresión geométrica en la que $0 < r < 1$ tiene la siguiente fórmula que no vamos a probar:

$$S = \sum_{n=0}^{+\infty} x_n = \frac{a}{1-r}.$$

Aplicando esto a Bitcoin obtenemos precisamente $S = \frac{10500000}{\frac{1}{2}} = 21000000$. Nótese que la cantidad total de Bitcoin no llegará a ser exactamente 21 millones (concretamente será 20999999.97480000 BTC, esto es, 2520000 Satoshis menos de los 21 millones de BTC) pues la cantidad mínima con la que se puede recompensar es con 10^{-8} BTC = 1 Satoshi. De hecho se puede calcular cuándo se empezará a recompensar con 1 Satoshi, lo cual será al acanzar el bloque 6720000. Por lo tanto, a partir del bloque 6930000 no habrá recompensa en absoluto. Cuando ese momento llegue, no se crearán más Bitcoins y los mineros solo obtendrán beneficios a través de comisiones en las transacciones. Para entonces se espera que el valor del Bitcoin sea tan alto que estas comisiones sean más que suficientes para compensar a los mineros.

No solo se conoce la cantidad máxima que habrá de Bitcoin, sino también cuándo se alcanzará esta cantidad. Como hemos visto en la sección de minería, el sistema ha sido diseñado para que se produzca aproximadamente un bloque cada 10 minutos. Para ello, a medida que la capacidad computacional de la red aumenta, también se incrementa la dificultad de generar los bloques. ¿Por qué 10 minutos? En realidad esta cantidad es arbitraria y cada criptodivisa establece el suyo, pero tiempos demasiado cortos podrían generar inestabilidad y tiempos muy largos retrasarían las confirmaciones.

Un bloque cada 10 minutos supone unos 144 bloques por día. Así que el tiempo para alcanzar 6930000 se necesitan $\frac{6930000}{144} = 48125$ días, algo menos de 132 años. Esto quiere decir que en torno al año 2140 se habrán minado todos los Bitcoins previstos.

3. Problema de los generales bizantinos

El problema de los generales bizantinos es un experimento mental creado para ilustrar el dilema de lograr un consenso entre un conjunto de entidades con un objetivo común cuando entre ellas pueden existir traidores, es decir, entidades con objetivos opuestos que intenten dinamitar el proceso. Ademas, se supone que las comunicaciones entre dichas entidades son limitadas e inseguras.

Problema 3.1. El problema se presenta como una analogía con un escenario de guerra, donde un grupo de generales bizantinos se encuentran acampados con sus tropas alrededor de una ciudad enemiga que desean atacar. Después de observar el comportamiento del enemigo, los generales deben comunicar sus observaciones y ponerse de acuerdo en un plan de batalla común que permita atacar la ciudad y vencer. Para ello, los generales se comunican únicamente a través de mensajeros.

Existe la posibilidad que algunos de los generales sean traidores y, por lo tanto, decidan enviar mensajes con información errónea con el objetivo de confundir a los generales leales. Un algoritmo que solucione el problema debe asegurar que todos los generales leales acuerdan un mismo plan de accion y que unos pocos traidores no pueden conseguir que el plan adoptado por los generales leales sea equivocado.

El desarrollo que hemos efectuado acerca de la blockchain su tecnología impide que una persona o un grupo pueda crear bloques falsos es considerada como la primera solución práctica al problema.

4. Aplicaciones de la Blockchain

Terminamos este trabajo mencionando algunas aplicaciones de la tecnología fundamental bajo Bitcoin: la Blockchain.

- **Gestión de identidades**. La tecnología blockchain permite a los usuarios crear su propia identidad digital a prueba de manipulación. Según los expertos, esta especie de ID basado en blockchain reemplazará pronto a los nombres de usuario y contraseñas en línea.

  Podremos utilizar nuestra identidad blockchain para acceder a aplicaciones y sitios web, firmar documentos digitales, etc. Ya hay algunas compañías que ofrecen este tipo de servicios, como: Onename, Keybase o ShoCard.

- **Alquiler de propiedades y economía colaborativa**. El sistema anterior se puede hacer aún más complejo combinándolo con un contrato inteligente de alquiler. Por ejemplo, si un propietario de un piso o un vehículo quiere alquilarlo, bastaría con elaborar y almacenar en la blockchain un contrato inteligente en el que el propietario fija un precio para el alquiler por un tiempo.

  En el momento en que el usuario realiza el pago con una transacción registrada en la blockchain, el contrato inteligente se ejecutaría permitiendo el acceso a la propiedad a dicho usuario concreto por el tiempo estipulado. Una compañía que ofrece este tipo de servicios es Slock.

- **Votar por Internet**. La blockchain resuelve uno de los grandes problemas de los sistemas de votación por Internet: el anonimato del voto.

  Por su propia estructura y funcionamiento, la blockchain puede garantizar que una persona no pueda votar más de una vez en una misma elección, al tiempo que garantiza la privacidad de su voto. Además, al no haber ninguna autoridad central que gestione la votación no es posible manipularla.

  El voto electrónico mejoraría la rapidez y abarataría considerablemente el coste de unas elecciones y referéndums, lo que permitiría hacer referéndums con más frecuencia y mejorar así la democracia.

Referencias

[1] Cristina Pérez-Solà & Jordi Herrera-Joancomartí, *Bitcoins y el problema de los generales bizantinos*, 2014. https://web.ua.es/en/recsi2014/documentos/papers/bitcoins-y-el-problema-de-los-generales-bizantinos.pdf

[2] Kahn Academy. https://www.khanacademy.org/economics-finance-domain/core-finance/money-and-banking/bitcoin/v/bitcoin-what-is-it

[3] Bitcoinwiki, https://en.bitcoin.it/wiki/How_bitcoin_works.

[4] Eric Rywalder, *The math behind Bitcoin*, Coindesk, 2014. http://www.coindesk.com/math-behind-bitcoin/

[5] Alex Gorale, *Explaining the math behind Bitcoin.* Cryptocoinsnews, 2014. https://www.cryptocoinsnews.com/explaining-the-math-behind-bitcoin/

[6] Scott Driscoll, *How Bitcoin works under the hood.* ImponderableThings, 2013. http://www.imponderablethings.com/2013/07/how-bitcoin-works-under-hood.html

[7] Bitcoin Forum, https://bitcointalk.org/index.php?topic=1473285.0