Caso de estudio: Bitcoin

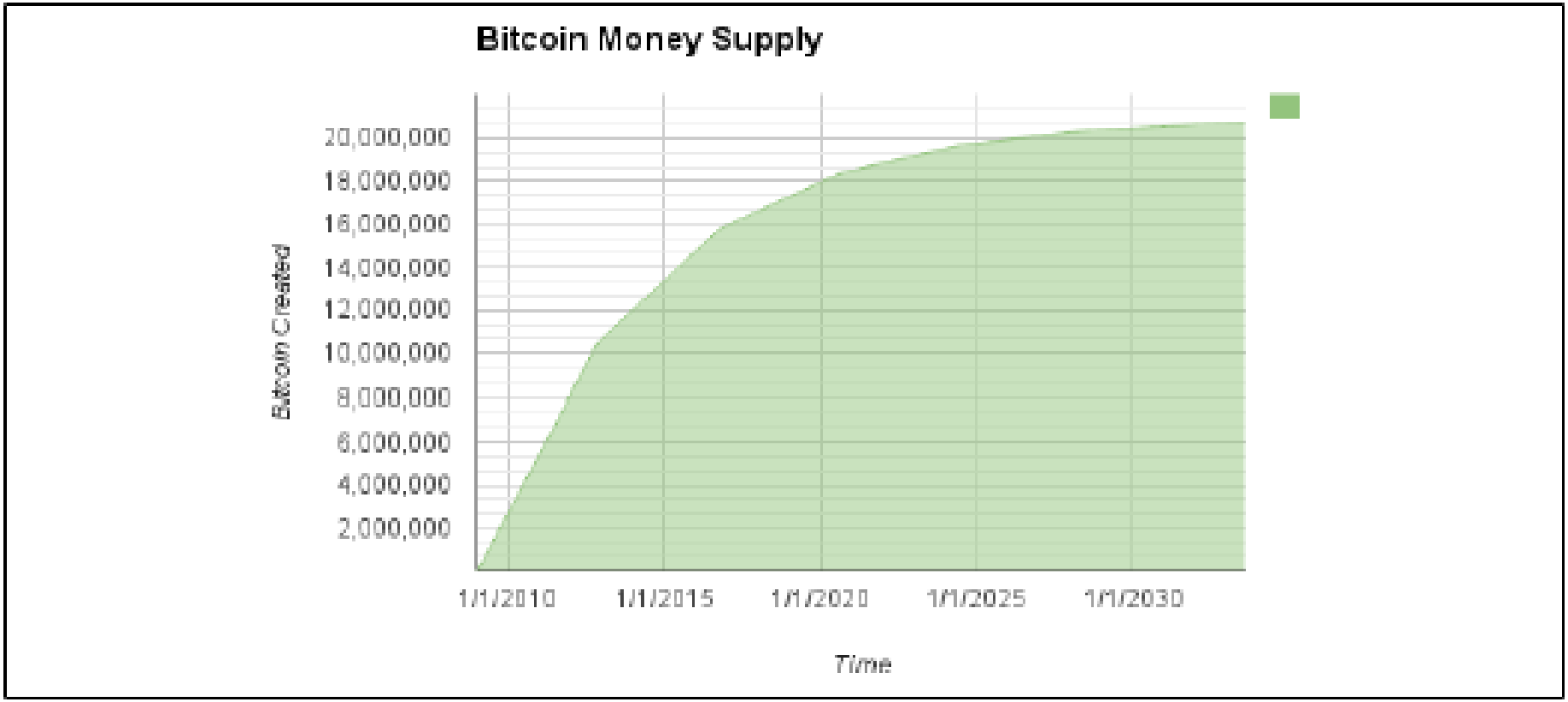
Bitcoin: motivación

- "Dinero electrónico": objetivo largamente buscado
- Criptografía de clave pública/privada permite afrontar el reto
  - En transacción electrónica, criptografía:
    - Asegura identidad de pagador y cobrador
    - Y la no manipulación (integridad) de la transacción
- Problema del *doble gasto*
  - ¿Cómo asegurar que hay fondos cuando emito 2 transacciones?
  - Entidad autorizada lo hace manteniendo histórico de transacciones
- Reto: eliminar entidad autorizada
  - La "comunidad" mantiene el histórico replicado
  - ¿Cómo consensuar el histórico ante nodos fraudulentos?
  - ¿Cómo fomentar la participación de la "comunidad"?

Bitcoin: introducción

- 1ª criptomoneda de carácter distribuido
  - Se gestiona sin la intervención de una autoridad central
  - Actualmente, hay otras criptomonedas (Ethereum, Litecoin…)
- Origen: artículo de "Satohsi Nakamoto" en 2008
  - Operativo desde el 3/1/2009
- Introduce su propio modelo de gestión del dinero
  - Moneda deflacionaria; llegará máximo en 2140: 21 millones de *bitcoins*
  - Cada 4 años se reduce a la mitad nº nuevas monedas que se crean
- Técnicamente: Sistema P2P desestructurado (*Blockchain*)
  - Cada nodo almacena una copia completa del "libro mayor contable"
  - Copias deben mantenerse sincronizadas aunque haya nodos malignos
  - Problema de consenso distribuido bizantino (tema sincronización)
- Estimula participación de comunidad pagando *bitcoins* (BTC)

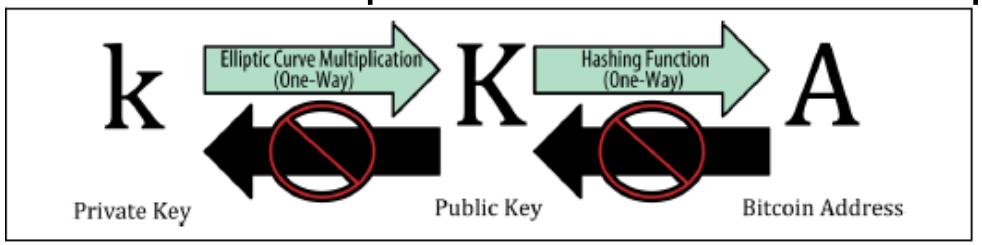
Bitcoin: Creación de *bitcoins*



Mastering Bitcoin: Andreas Antonopoulos

Bitcoin: claves, direcciones y billeteras

- Criptografía, evidentemente, juega un papel fundamental
- Antes de operar, cliente debe generar claves privada/pública
  - Y generar una *dirección bitcoin* a partir de la pública
    - Será la que se use como destino en las transacciones
    - Buena práctica:1 dir./transacción en vez de reusar como cuenta bancaria
  - Clave privada crítica: sin ella se pierde acceso al dinero



Mastering Bitcoin: Andreas Antonopoulos

- Cliente no se da de alta en el sistema
  - Su dirección aparecerá como destino en una transacción
- Billetera (*wallet*): doble acepción
  - Contenedor de mis direcciones y claves (no de mis *bitcoins*)
  - Aplicación que ofrece interfaz para interacción con *bitcoin/blockchain*

Mastering Bitcoin: tipos de billeteras

paper wallet

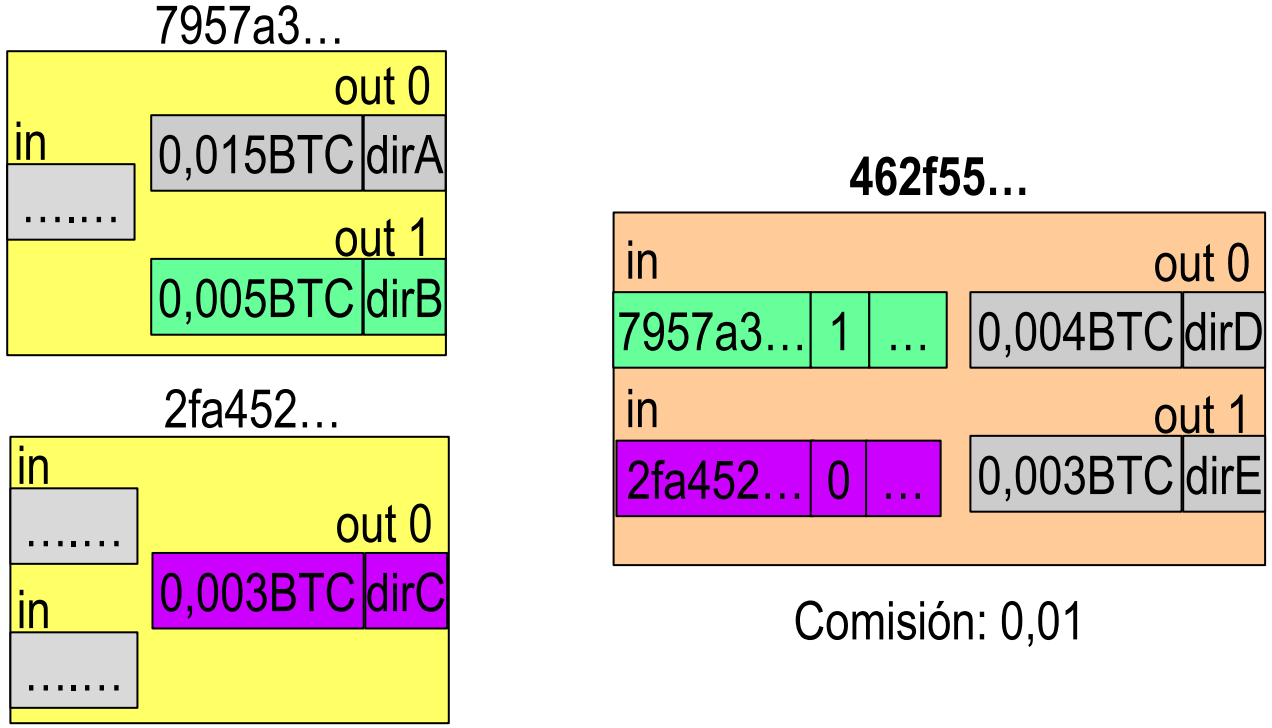
hardware wallet

app wallet

Bitcoin: transacciones

- El dinero se gestiona mediante transacciones encadenadas
- Transacción: un usuario envía dinero a otro usuario
  - Ese dinero proviene de una transacción que recibió el primer usuario
  - Segundo usuario podrá usar ese dinero en una transacción a tercero
- Modelo general de transacción con N entradas y M salidas
  - Transacción puede gastar N transacciones para pagar a M usuarios
  - Cada entrada hace referencia a salida de otra transacción: la gasta
    - Como un billete que no se puede dividir: necesidad de cambio
    - ID de transacción: *hash* de su contenido
  - Cada salida indica una cantidad y la dirección del destinatario
- Transacciones sin entradas: *coinbase*
  - Crean (acuñan) dinero y se lo pagan a usuarios indicados en salidas
- Saldo de usuario: transacciones no gastadas dirigidas a él

Bitcoin: anatomía de una transacción

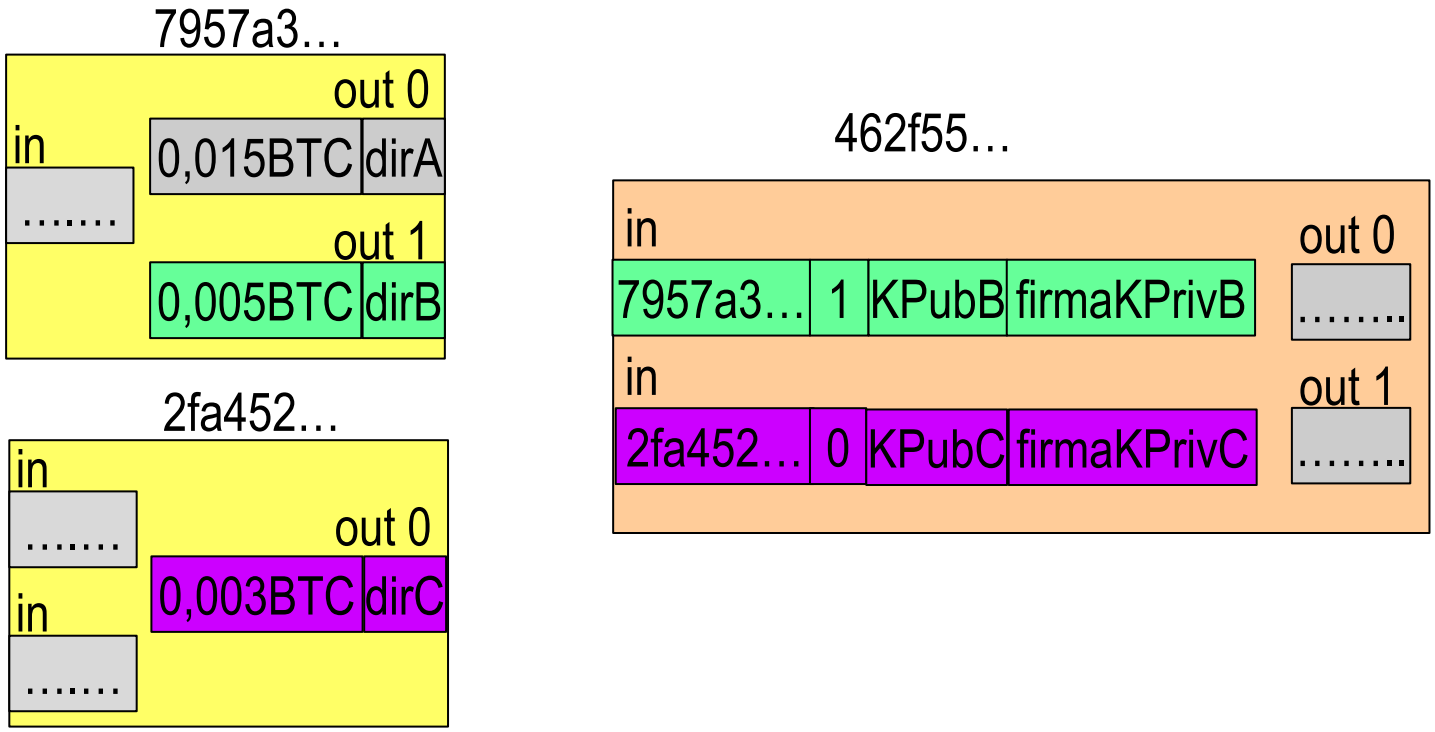


- Suma de entradas (0,08) ≥ Suma de salidas (0,07)
  - Si hay diferencia (0,01)⟶ comisión (*fee*) para minero que la procese
    - Mayor comisión ⟶ más estímulo para que minero la valide
    - Una transacción sin comisión podría no validarse nunca
    - ¿Cómo estimar la comisión adecuada? http://bitcoinfees.21.co

Bitcoin: verificación de transacción

- Salida transacción indica dirección *bitcoin* de quién puede gastarla
- Entrada de nueva transacción debe demostrar identidad
- Cuando se construye una nueva transacción en cada entrada
  - Clave pública correspondiente a dirección *bitcoin* de la salida asociada
  - Firma de *hash* de nueva transacción con clave privada correspondiente
    - Asegura identidad y también integridad de la transacción
- Para verificar la validez de una transacción por cada entrada
  - Comprueba que clave pública corresponde a la dir. bitcoin de la salida
  - Realiza el *hash* de la transacción
  - Aplica la clave pública a la firma
  - Comprueba que *hash* calculado = valor obtenido al aplicar clave pública

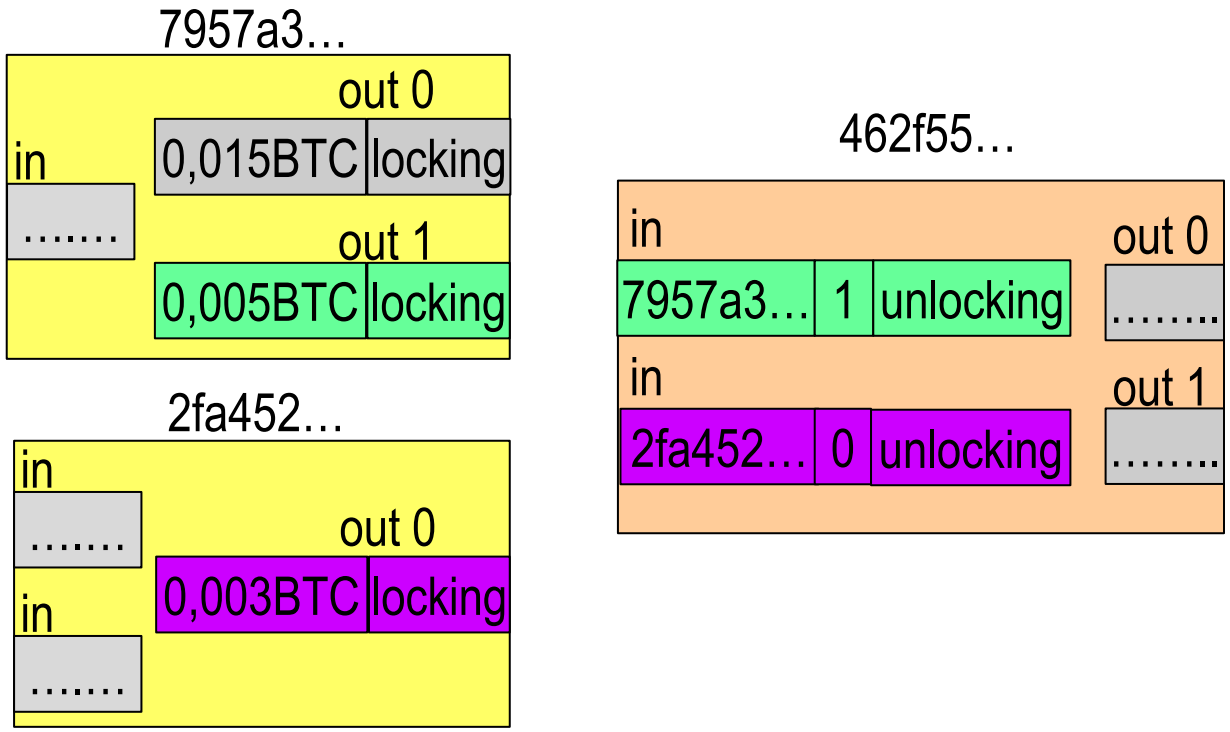
Bitcoin: información de verificación



Bitcoin: verificación programable

- La explicación sobre la verificación está simplificada
- En verdad, se trata de un mecanismo genérico programable
  - Cada salida incluye un *script* de *locking* (el candado)
  - Cada entrada un *script* de *unlocking* (la llave)
  - Escritos en *Script*: lenguaje no Turing completo; sin bucles; usa pila
- Cuando se verifica una transacción, por cada entrada:
  - Se ejecuta el *script* de *unlocking* de la entrada de la nueva transacción
    - Genera datos de verificación en la pila
  - Se ejecuta el *script* de *locking* de la salida de la transacción gastada
    - Comprueba si datos de verificación en pila son los requeridos
- En la transacción, realmente se almacenan:
  - dirección bitcoin en cada salida; clave pública y firma en cada entrada
  - Pero como operandos de las instrucciones del *script*:
    - p.e. PUSH clave pública

Bitcoin: verificación programable



Método de verificación descrito es el que se usa por defecto (P2PKH)
Pero hay otros como P2SH y pueden crearse nuevos al ser programable

Bitcoin: visión del usuario

- Aplicación *Wallet*: gestión como si fuera cuenta bancaria
- *Bitcoin* no gestiona balance de cada usuario
  - *Wallet* lo calcula: Σ salidas no gastadas asociadas a dirs. del usuario
  - *Wallet* guarda todas las direcciones asociadas a ese usuario
- Usuario *U* quiere realizar un pago de una cantidad *C* a *P*
  - *U* obtiene dirección *bitcoin* de *P* (de viva voz, correo, web, lee QR…)
  - *Wallet* selecciona suficientes salidas no gastadas de *U* para pagar *C*
    - Y dejar una propina razonable
  - Por cada salida elegida crea entrada asociada con *unlocking script*:
    - clave pública y firma con clave privada asociadas a esa dirección
  - Crea salida que especifica *C* en *satoshis* (1BTC=10^8 *satoshis*)
    - Y con un *locking script* que contiene dirección *bitcoin* de *P*
    - Posible salida adicional para el cambio
  - *Wallet* envía la transacción a la *Blockchain*

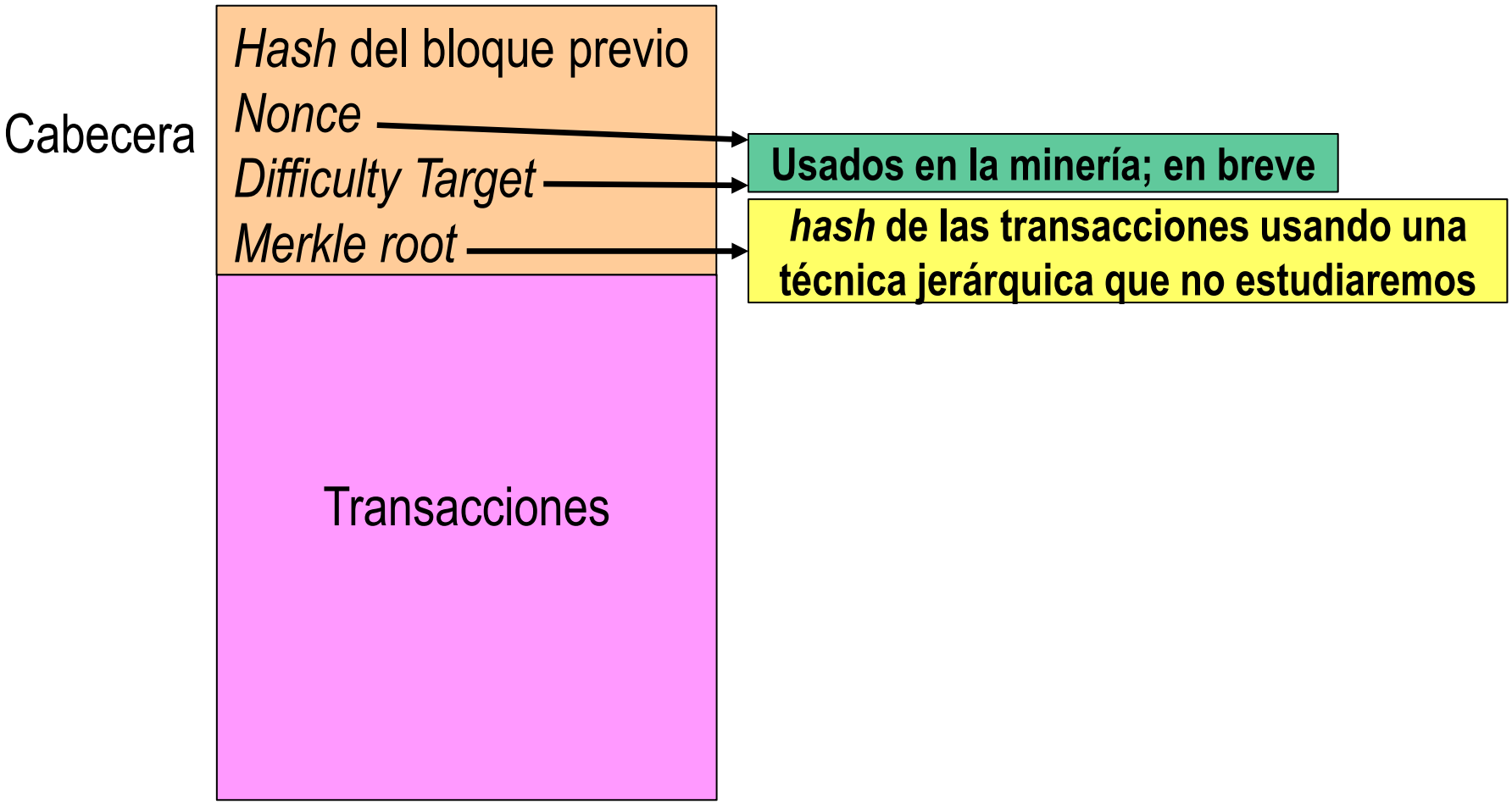
Bitcoin: Necesidad de consenso

- Cada nodo mantiene copia de todas las transacciones
  - Pero cómo resolver el problema del doble gasto
    - 2 transacciones *T1* y *T2* usan la misma salida no gastada
    - *Nodo A* recibe, y valida, primero *T1* y rechaza *T2*
    - *Nodo B* recibe, y valida, primero *T2* y rechaza *T1*
    - Copias de transacciones son diferentes en distintos nodos
  - Se requiere un consenso en orden de procesado de transacciones
- Consensuar transacción a transacción no parece razonable:
  - Las transacciones se agrupan en bloques
    - "Vamos cogiendo puñados de transacciones y metiéndolas en urnas"
    - Tamaño de bloque: originalmente 1MB
  - Y se crea una cadena/lista de bloques → *Blockchain*
  - No confundir cadena de transacciones y de bloques
    - 2 transacciones relacionadas pueden acabar en mismo o distinto bloque

Blockchain

- Estructura de datos que almacena histórico de transacciones
  - Desde el bloque génesis (0): solo 1 transacción *coinbase* de 50BTC
  - Hasta hoy: *https://blockchain.info*
  - Ocupa > 300GB, creciendo ≈ 1MB (1 bloque) cada ≈ 10 minutos
- Lista: cada bloque referencia al previo (así hasta bl. Génesis)
- Cada bloque guarda un conjunto de transacciones
  - Nº medio de transacciones por bloque ≈ 500
- Contenido bloque: cabecera + transacciones
- Bloque identificado por su *hash* (realmente, *hash* de cabecera)
  - También por posición en la lista: su altura (lista se visualiza vertical)
- Cabecera incluye: *Hash* de bloque previo
  - Asegura que no se puede cambiar *a posteriori* contenido del previo
  - Ni puede generarse un nuevo bloque antes de completarse el previo

Blockchain: estructura de un bloque



Blockchain: Minería de bloques

- Nodo crea un nuevo bloque *N+1* agrupando transacciones
  - Selecciona y verifica conjunto transacciones y las incluye en bloque
- Estímulo: crear, y poseer, nuevos BTC y quedarse comisiones
  - La primera transacción del bloque es de tipo *coinbase* y crea *bitcoins*
    - La incluye el propio minero para pagarse a sí mismo
    - Actualmente, 6,25 BCT por bloque (se reduce a la mitad cada 4 años)
    - A partir de 2140 la minería de un bloque solo otorgará comisiones
- Propaga el bloque a todos los demás nodos
- Cada nodo genera su propio bloque *N+1*
  - ¿Quién gana? ¿Cómo lograr consenso? ¿Votación?
    - Atacante puede controlar muchas IP
  - *Proof-of-work* (PoW)

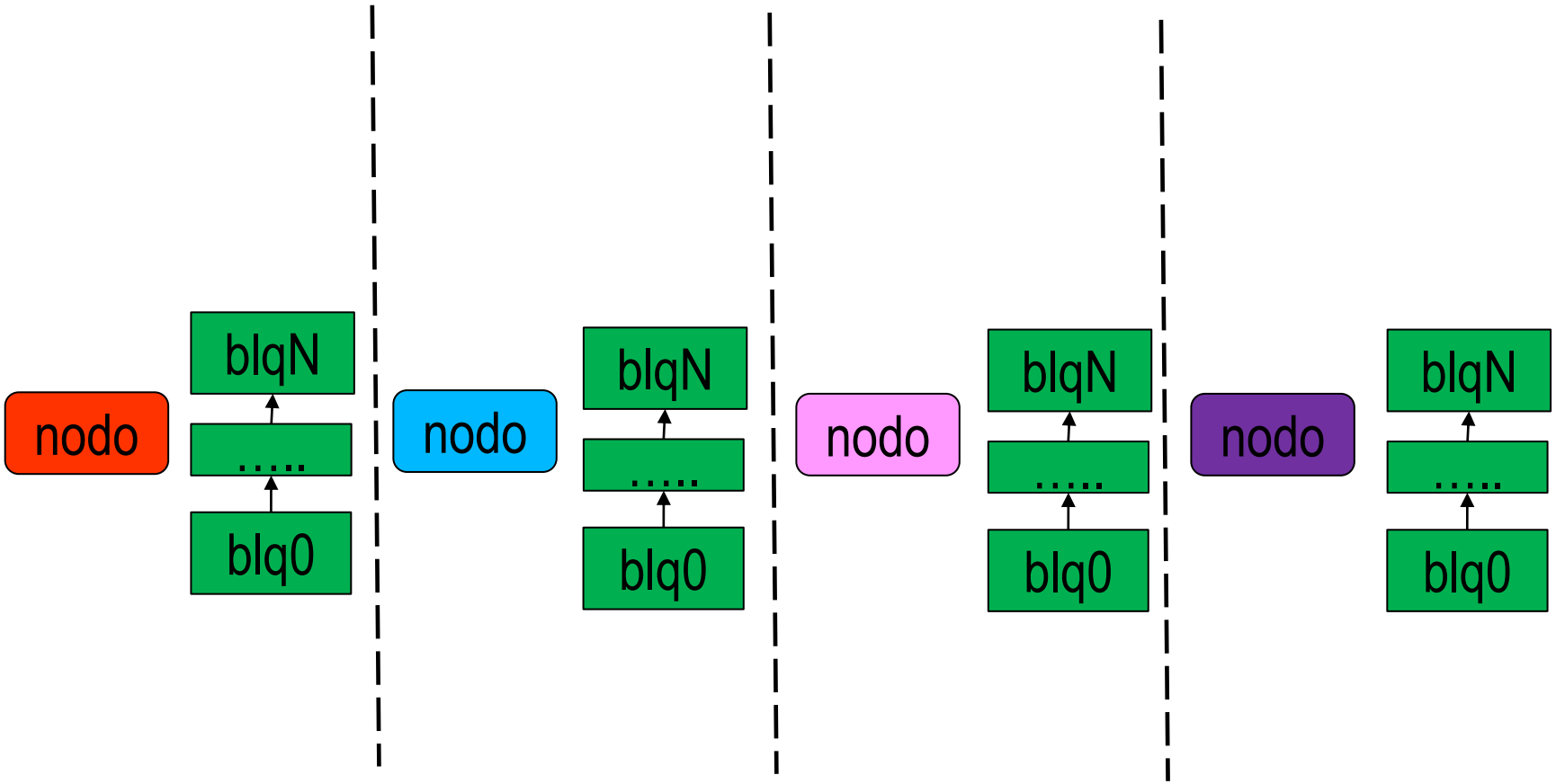
Metáfora del *Proof-of-work*

- Varias personas tienen que consensuar una decisión
- Van a correr un maratón y el que gane lo decide
- ¿Y si dos llegan a la vez?
  - Es muy difícil: por eso corremos un maratón y no 100 metros
  - Si ocurre, algunos harán caso a uno y otros al segundo (***fork***)
  - Más adelante, habrá que tomar otra decisión: nuevo maratón
  - Ganador toma la decisión actual pero también rompe empate previo
    - La decisión previa por la que optó el ganador actual es la consensuada
- PoW en bitcoin:
  - "Mineros" deben mostrar un gran esfuerzo de computación
    - Solución original al problema del consenso distribuido
    - Aunque puede tener consecuencias medioambientales importantes

Blockchain: PoW

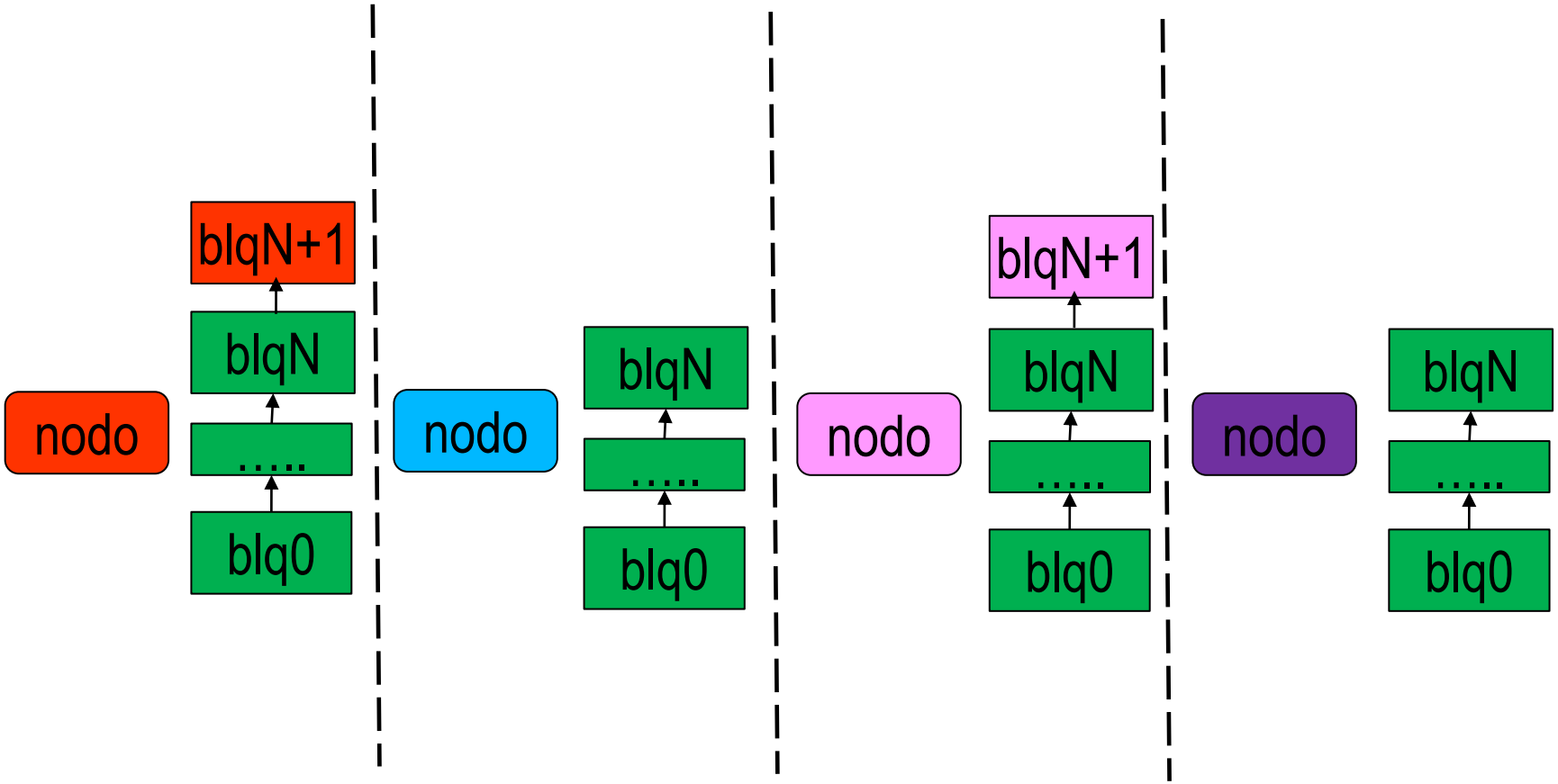
- Minero, después de rellenar bloque, debe realizar cómputo
  - Conseguir que *hash* de bloque sea < cierto valor: *Difficulty Target*
  - Prueba y error: modificando campo *Nonce* hasta que se cumpla
  - Cálculo artificial, innecesario y que requiere muchísimo cómputo
- Objetivo: generar un bloque cada ≈ 10 minutos
  - Cada 2016 bloques (≈ 2 semanas) se recalcula *Difficulty Target*
    - Todo nodo lo reajusta para conseguir ritmo de generación deseado
- Nodos pueden completar generación prácticamente a la vez
  - Nodos reciben varias versiones del bloque en distinto orden: ***fork***
  - Nodo se acaba quedando con *fork* más largo (otros descartados)
- ¡Bloque incluido en Blockchain puede ser descartado!
  - Bloque se considera definitivamente validado si hay 6 posteriores
- Minería de bloques como "negocio": ASIC, *mining pools*…

Blockchain: Fork



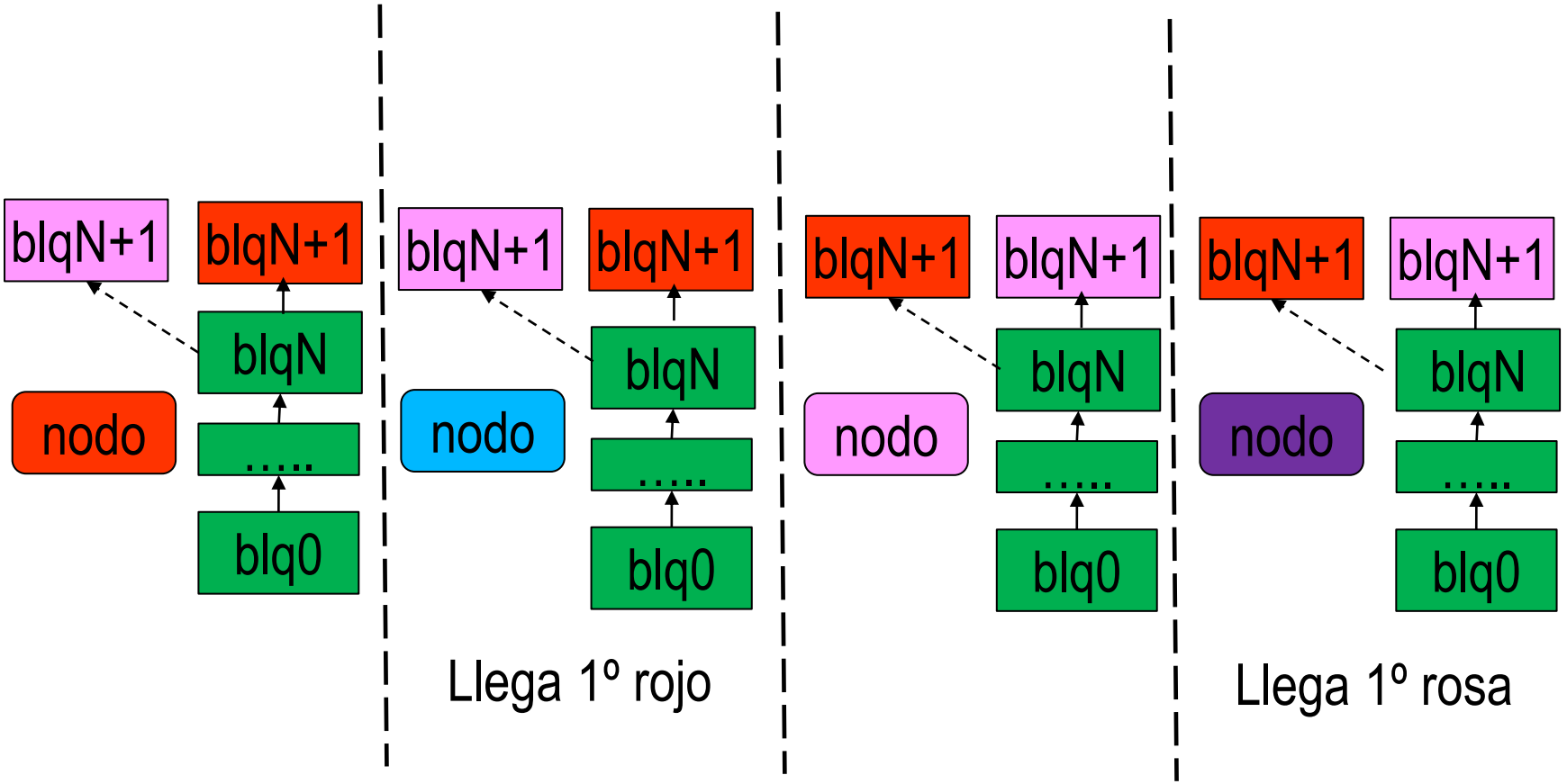
Situación inicial

Blockchain: Fork



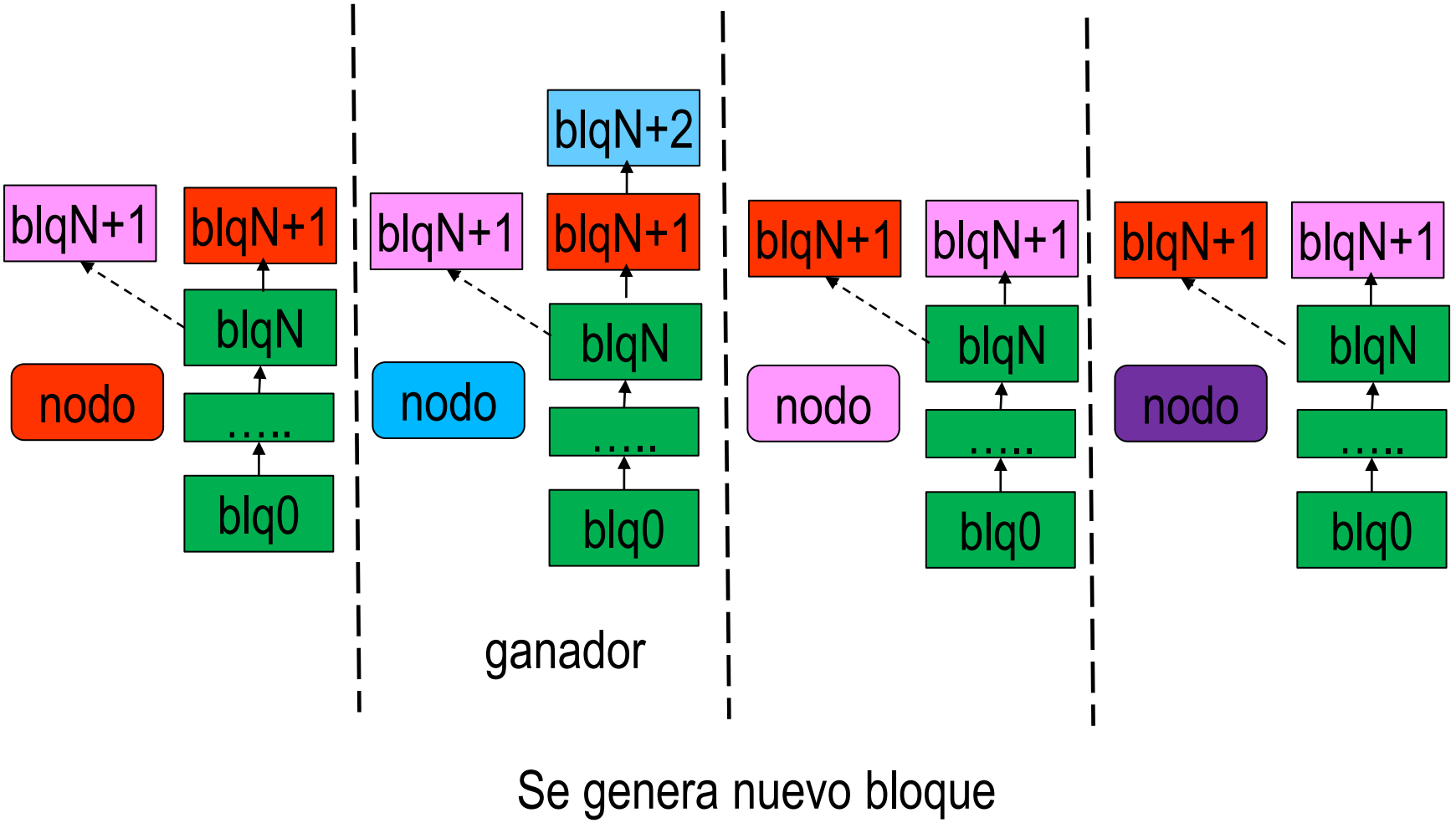
Empate: 2 ganadores

Blockchain: Fork



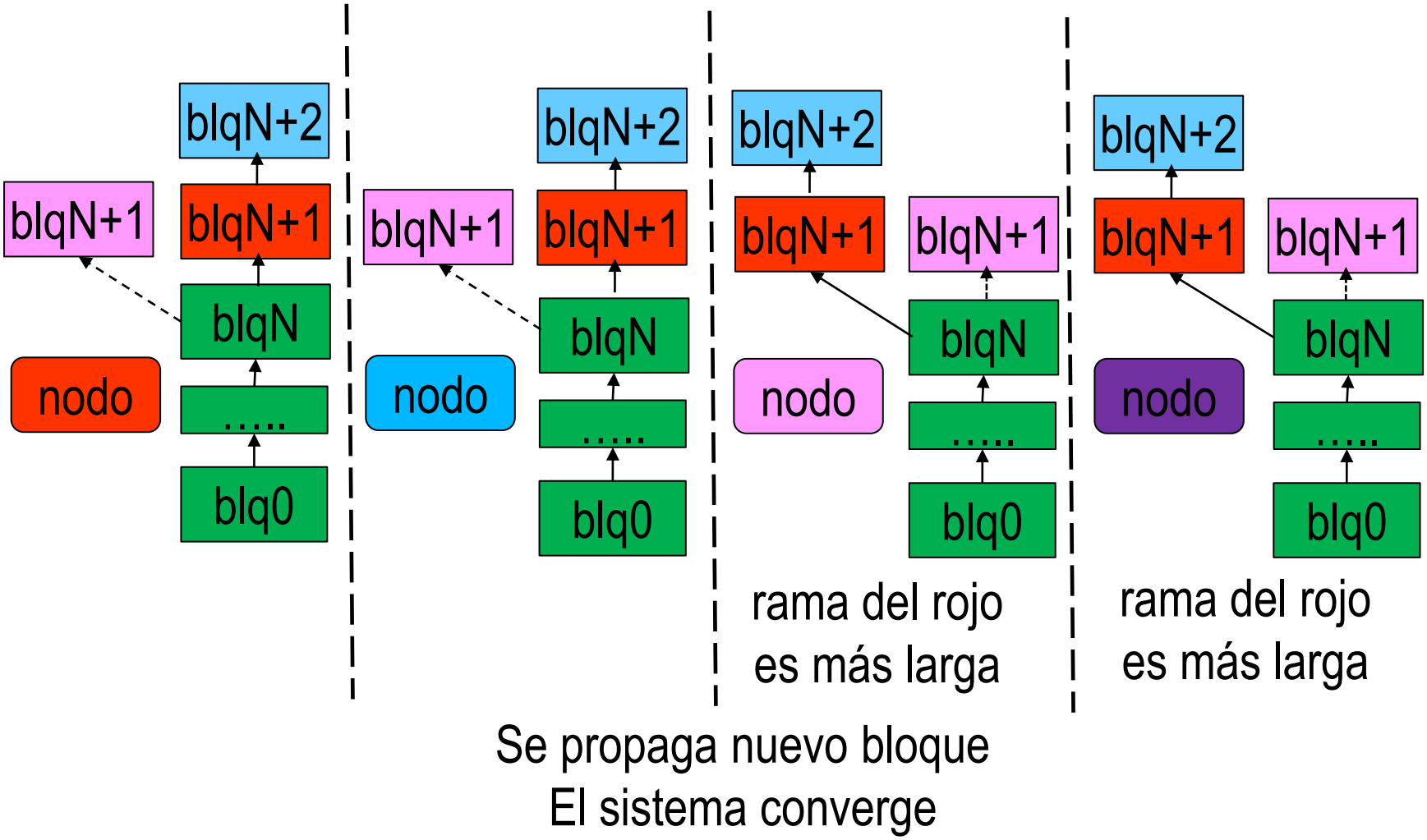
Se propagan nuevos bloques

Blockchain: Fork



Se genera nuevo bloque

Blockchain: Fork



Blockchain: Arquitectura P2P

- Nuevo nodo debe averiguar IP de al menos 1 nodo en bitcoin
  - *DNS seed*: nombre máquina con muchas IP de nodos asociadas
    - *host dnsseed.bluematt.me*
- Al conectarse (puerto TCP 8333) descubre nuevos nodos
  - A partir de entonces, intercambio de IP de "vecinos" entre nodos
  - Para asegurar conectividad ante desconexión de nodos
- Nuevo nodo debe descargar toda la *blockchain* (y validarla)
  - Proceso lento y consumidor de recursos
- Comunicación entre nodos por "inundación"
  - Al recibir transacción, si no la ha recibido antes:
    - La valida y la transmite a los nodos vecinos
  - Lo mismo para los bloques

Blockchain: Tipos de nodos

- No todos los nodos involucrados tienen misma funcionalidad
- Nodo estándar:
  - Mantiene una copia completa de la *blockchain*
  - Participa en el proceso de propagar transacciones y bloques
- Nodo minero: Nodo estándar + funcionalidad de minería
- *Simplified Payment Verification* (SPV):
  - Cliente ligero que incluye solo la funcionalidad de la billetera
  - No mantiene copia de la *blockchain*
  - Solo interesado en bloques que contienen transacciones del usuario
  - Mecanismos como árboles Merkle (no los estudiamos)
    - Le permiten trabajar solo con las cabeceras de los bloques de interés
    - KBs en vez de MBs

Proof-of-stake (PoS)

- PoW tremendo gasto energético para cálculos superfluos
- Búsqueda de alternativas fuera de *bitcoin* que no lo requieran
  - Pero mantengan incentivos para que haya mineros
  - Y desanimen a los mineros tramposos
- *Proof-of-stake*
  - Un minero debe comprar ese derecho
    - Adquiriendo y bloqueando cantidad significativa de capital en esa moneda
  - Crítica: solo los ricos pueden hacerse más ricos minando
    - Aunque eso ya ocurría con PoW por el coste de la minería
  - Mineros validan las transacciones sin esfuerzo computacional añadido
    - Criterio para elegir el ganador: p.e. de forma aleatoria
  - Si minero intenta hacer trampas, será sancionado y además:
    - impacto negativo en credibilidad de la moneda, que afecta a su inversión