



Universidad del
Rosario



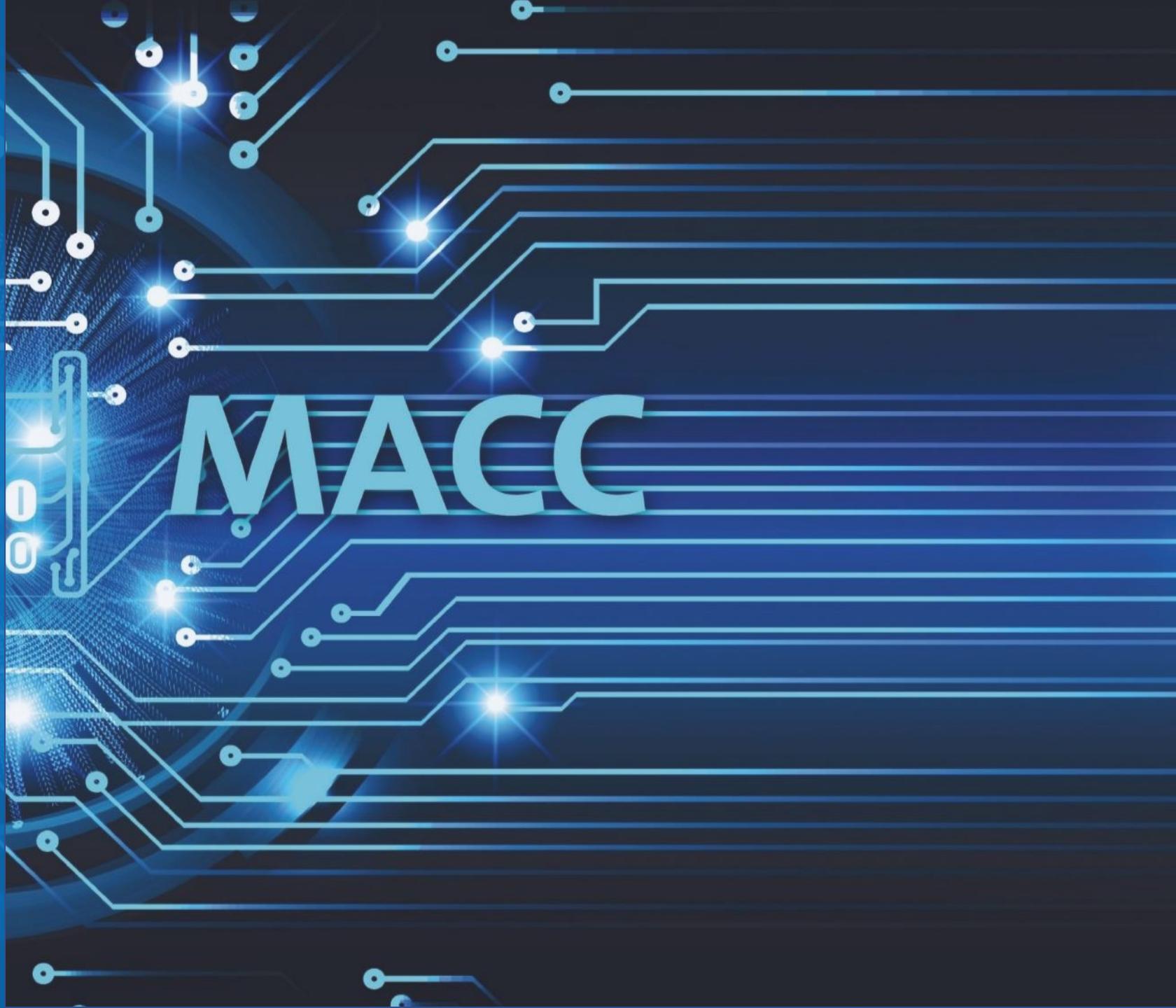
MACC
Matemáticas Aplicadas y
Ciencias de la Computación

Aplicaciones Blockchain

IV- Blockchhain, Bitcoin, Smart Contracts

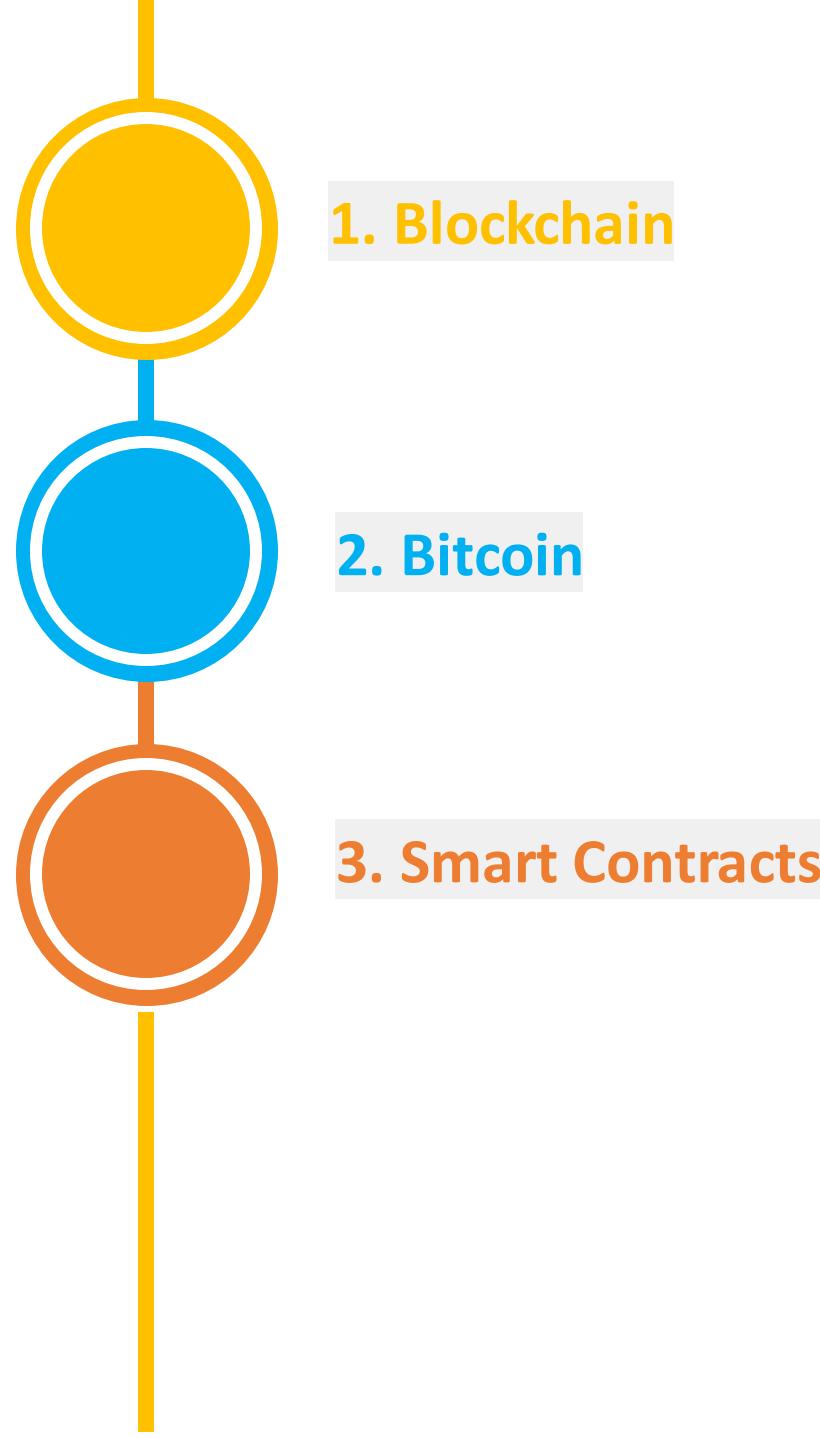
Carlos Castro
Valérie Gauthier
Martín Ochoa

Departamento MACC
Facultad de Economía
Universidad del Rosario



CONTENIDO

Aplicaciones Blockchain



1. Firmas

Mensaje m

Firma $f(m, k)$

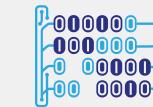
- Cualquiera puede verificar validez de firma.
- Solo poseedor de llave privada puede firmar.

1. Hashes

Mensaje m

Hash $h(m)$

- Mensaje arbitrariamente grande
- Hash $h(m)$ de tamaño constante
- Hash fácil de calcular, difícil de invertir
- Si cambiamos solo 1 bit de m , el hash cambia significativamente.



1. Hashes

Download Apache OpenOffice

(Hosted by SourceForge.net - A trusted website)

Select your favorite operating system, language and version:

OS X (version >= 10.7) (DMG)

English [US]

4.1.5

[Download full installation](#)

[Download language pack](#)

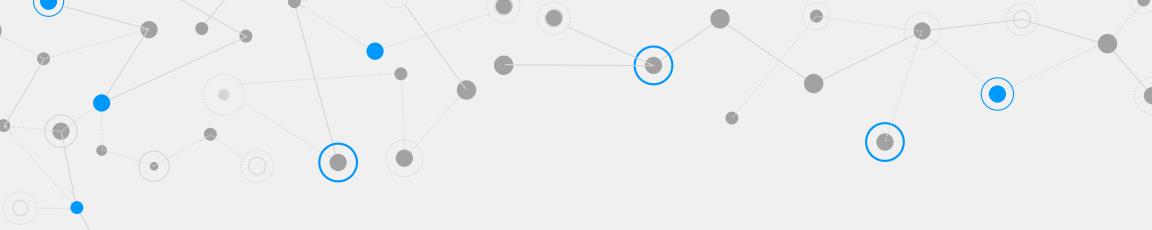
Release: Milestone AOO415m1 | Build ID 9789 | SVN r1817496 | Released 2017-12-30 | [Release Notes](#)

Full installation: File size ~ 163 MByte | Signatures and hashes: [KEYS](#) , [ASC](#) , [MD5](#) , [SHA256](#)

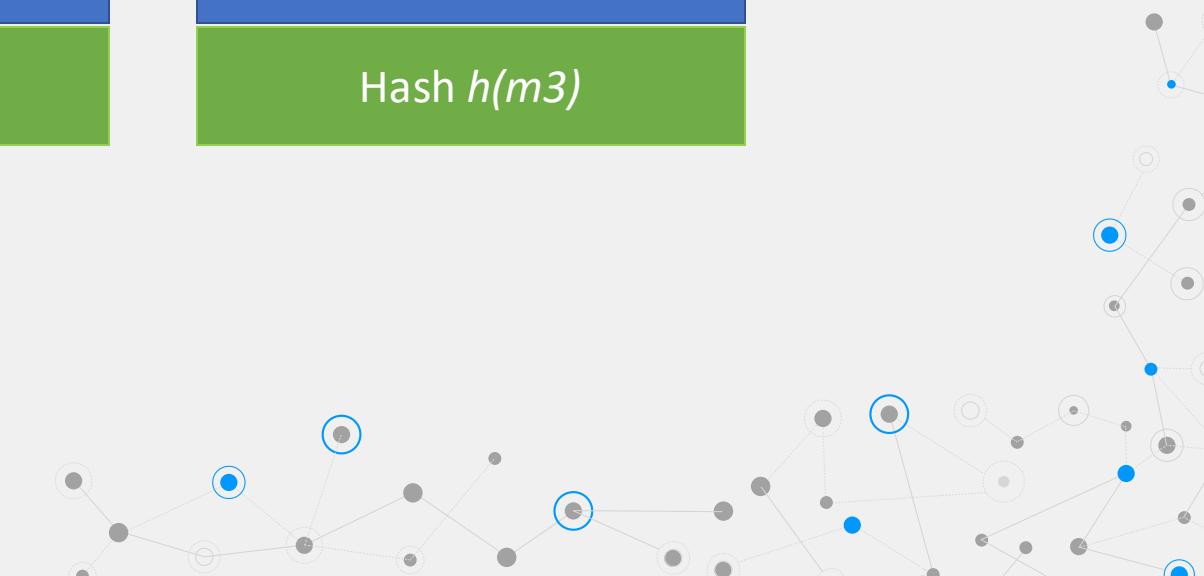
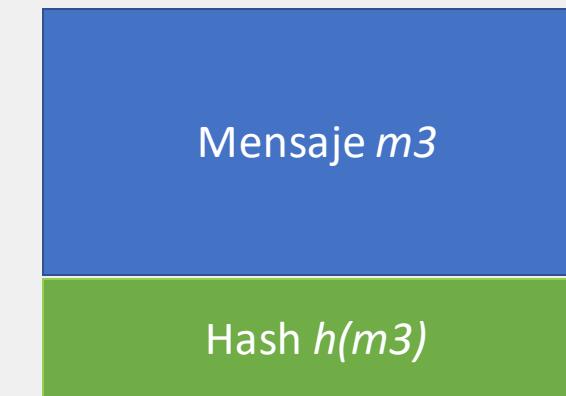
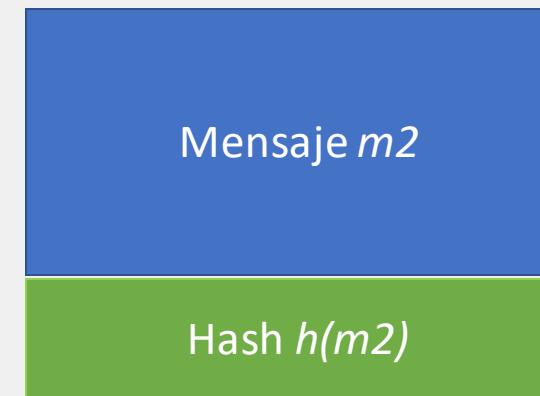
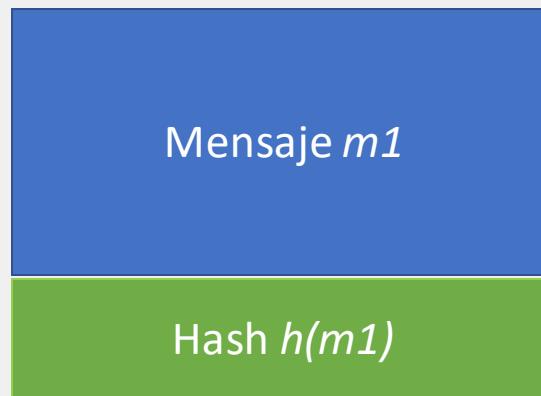
Language pack: File size ~ 17 MByte | Signatures and hashes: [KEYS](#) , [ASC](#) , [MD5](#) , [SHA256](#)

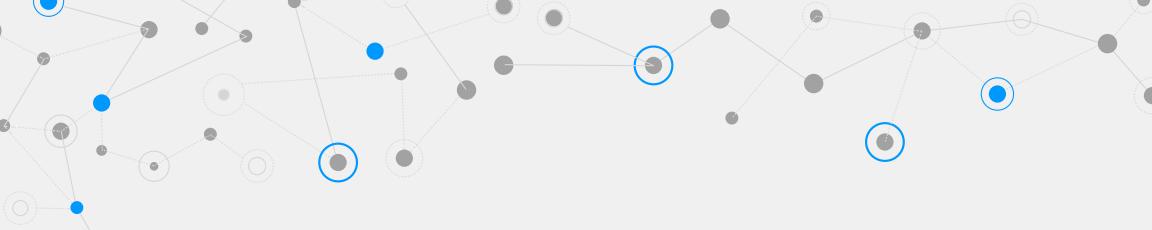
6b68adad6da7713e3fa62f01209826e0f5f87fc648d39f78e463b2b4b8223c59

*Apache_OpenOffice_4.1.5_MacOS_x86-64_install_en-US.dmg

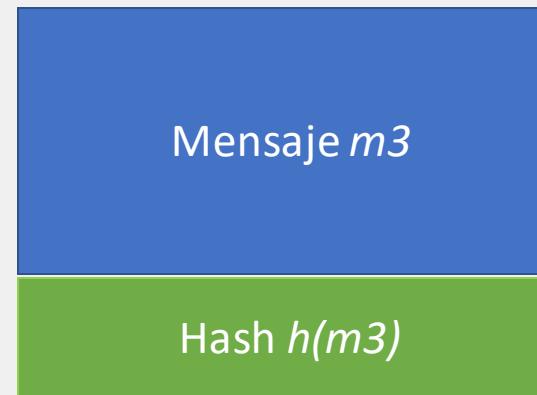


1. Cadena de transacciones 1



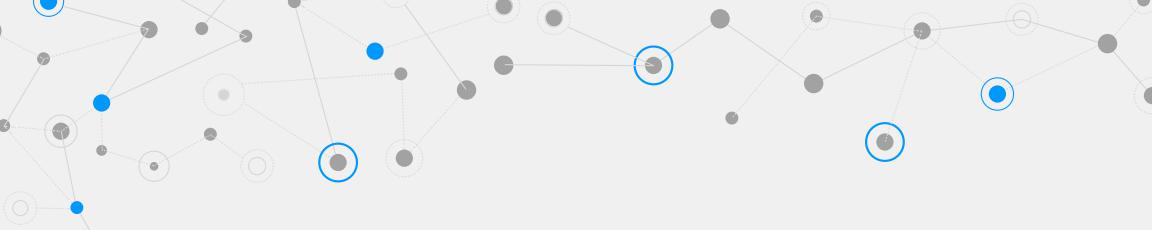


1. Cadena de transacciones 1

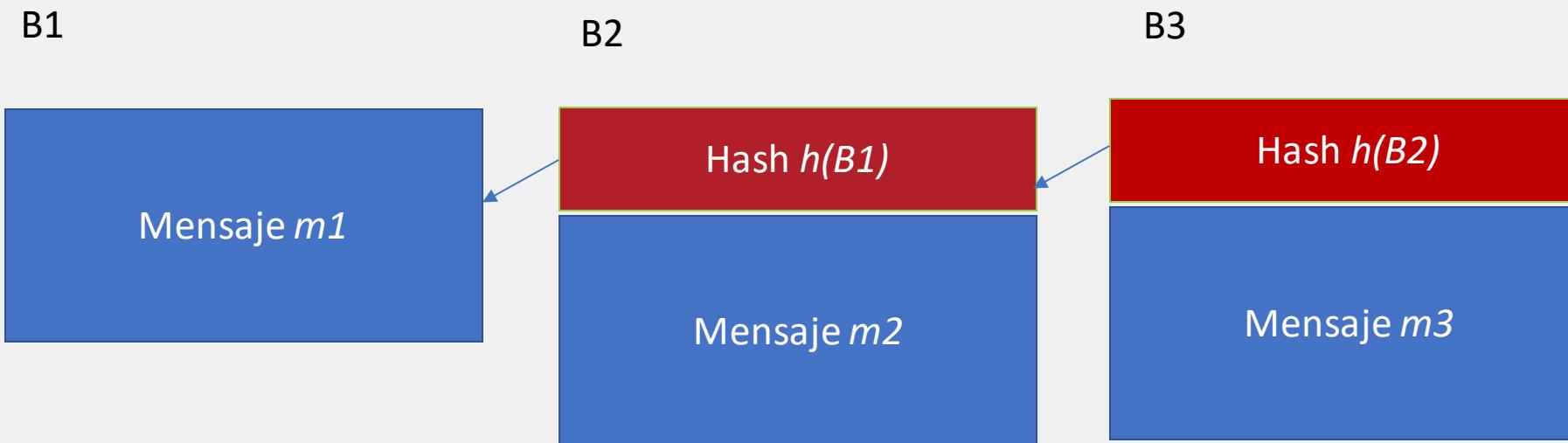


Puedo cambiar el orden de las transacciones, sin alterar los hashes.

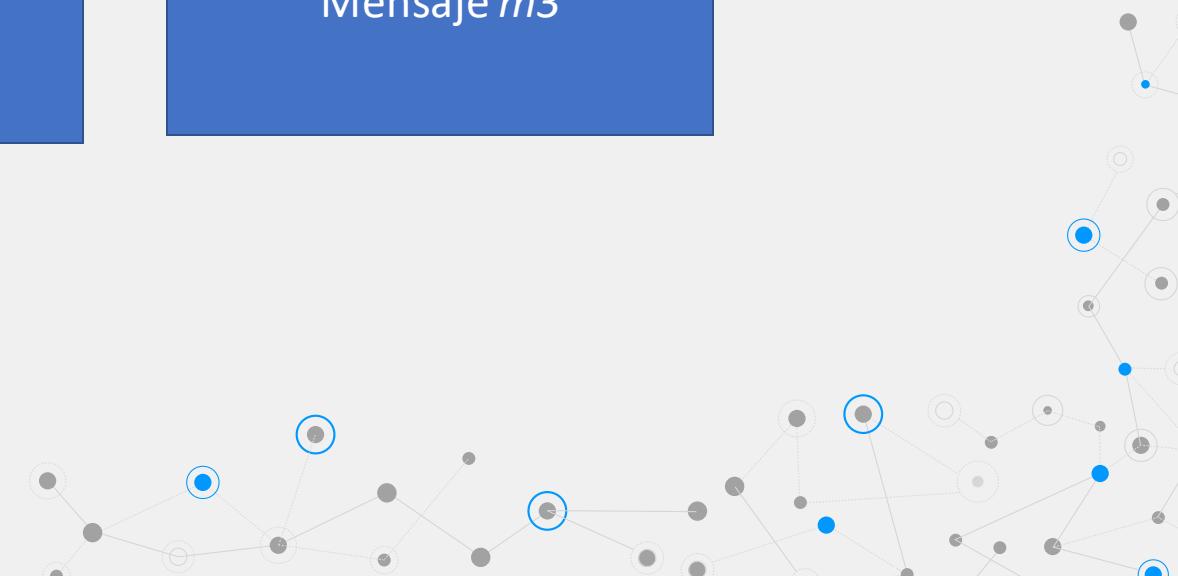


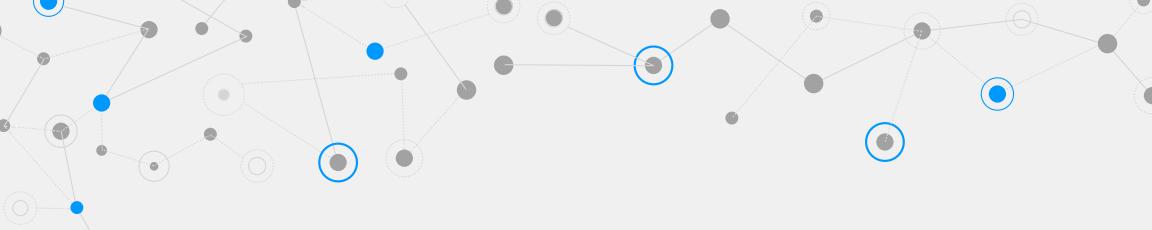


1. Cadena de transacciones 2



Que propiedades tiene esta estructura?





1. Cadena de transacciones 2

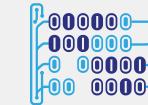
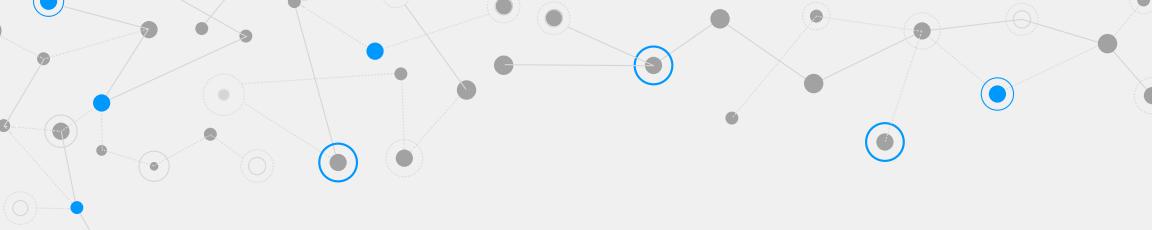
- Cualquier modificación (al menos 1 bit) a cualquier mensaje, invalida el hash del mensaje y el puntero del mensaje siguiente, por lo tanto “rompe” la cadena.
- No se puede cambiar el orden de los mensajes porque no serían validos los punteros.
- Integridad de los mensajes y de la secuencia temporal.



1. Taller

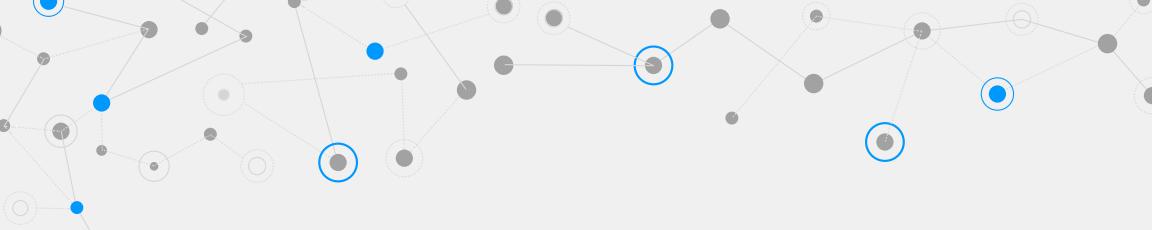
- A) Construya una cadena de bloques para los mensajes usando SHA-256:
 - Aplicaciones
 - Blockchain
 - UR

```
[[], 'Aplicaciones'],
['ed838cd18c5f039067d74444c860d9b7328fafad9adb97fa7eb7ab66ac
27a08c', 'Blockchain'],
['db04705af8ed390f49796895452741d9cd95e320b39e070791bfe07fc4
0a5abd', 'UR']]
```

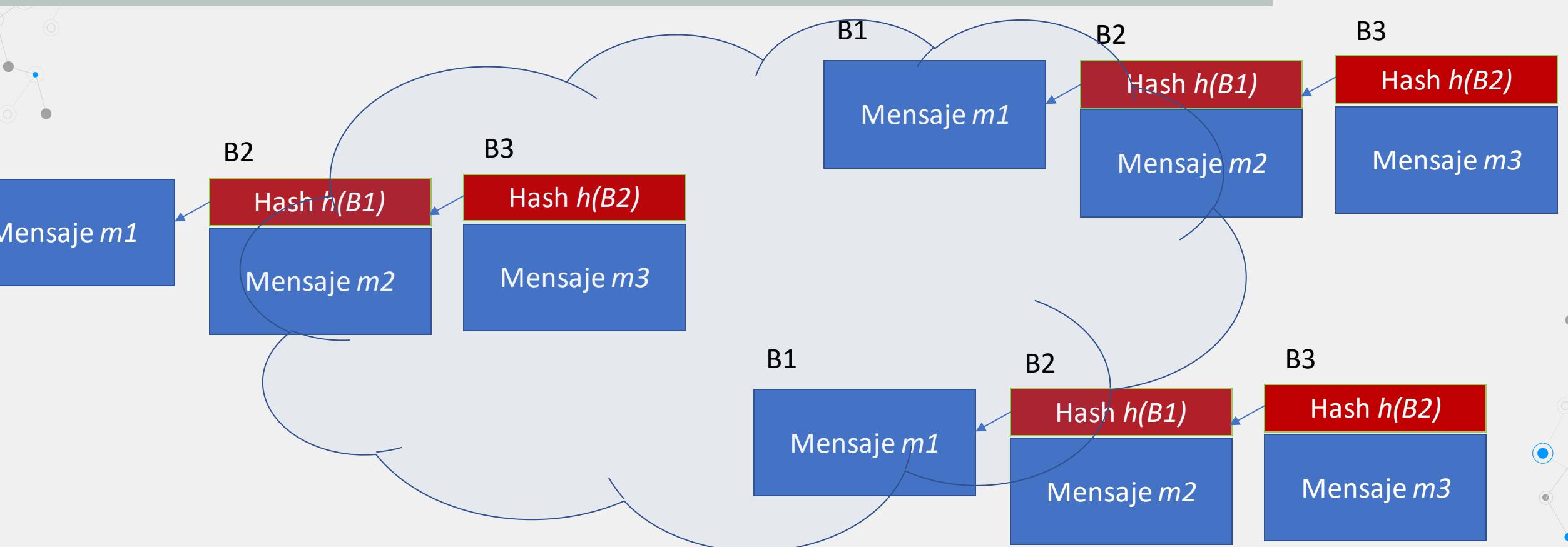


1. Taller

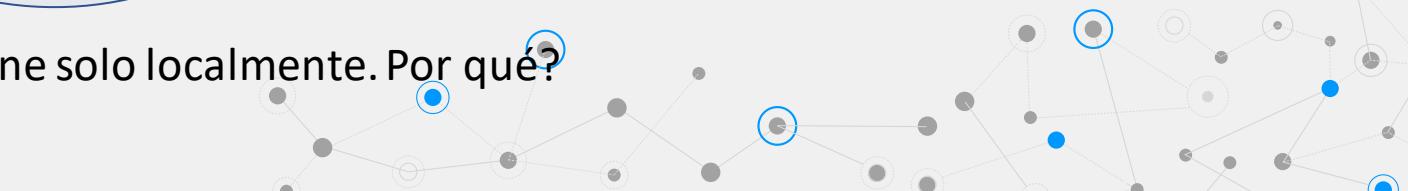
- B) Construya una función que verifique la integridad de la cadena.
- C) Corrompa la cadena y compruebe que está corrupta usando la función del punto B.



1. Cadena de transacciones

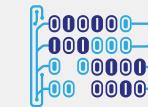
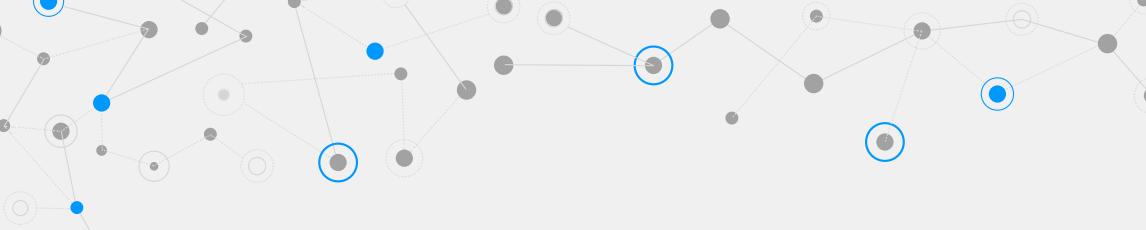


Esta estructura no tiene sentido si se mantiene solo localmente. Por qué?



1. Cadena de transacciones 2

- La idea de blockchain desde su concepción es la descentralización.
- Si un grupo de usuarios mantiene copias de la cadena de bloques, un atacante debería comprometer todos los nodos para poder cambiarla.
- Cómo actualizar la cadena? (añadir un bloque).



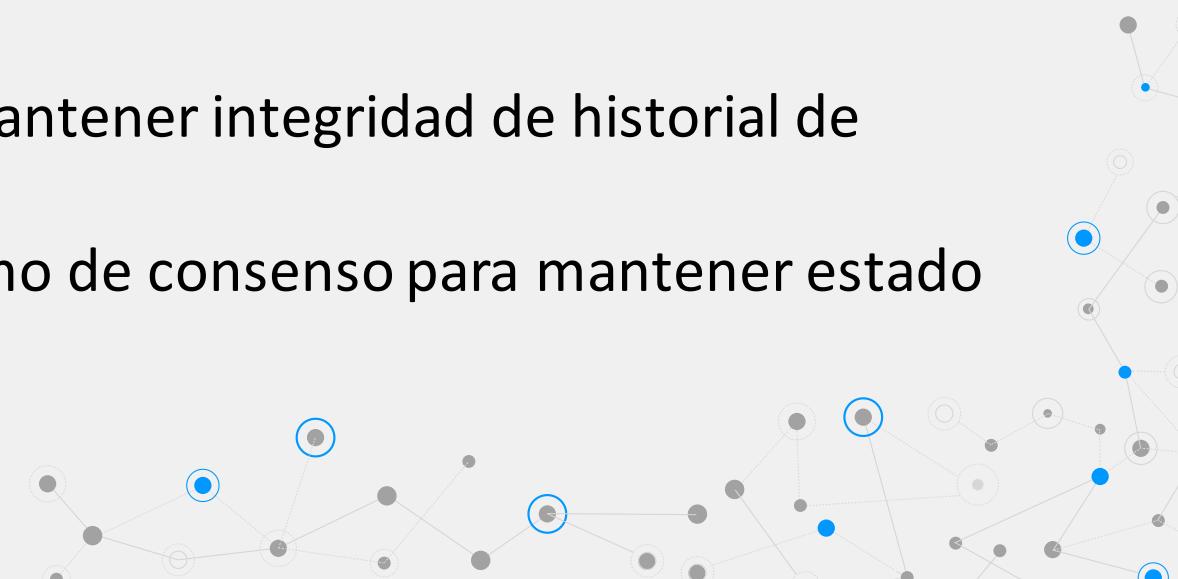
2. Bitcoin

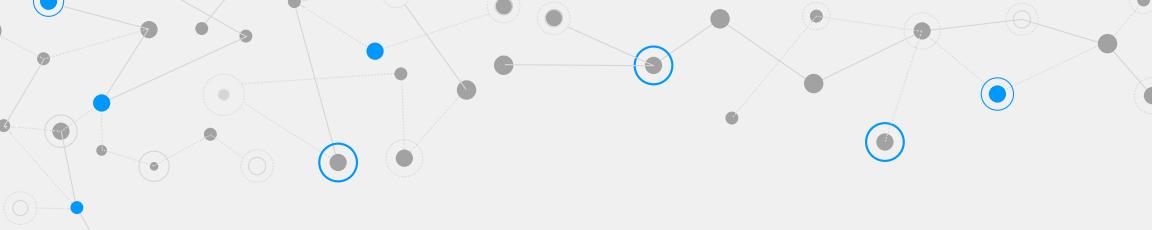
La innovación propuesta por Satoshi Nakamoto en el paper original de bitcoin:

<https://bitcoin.org/bitcoin.pdf>

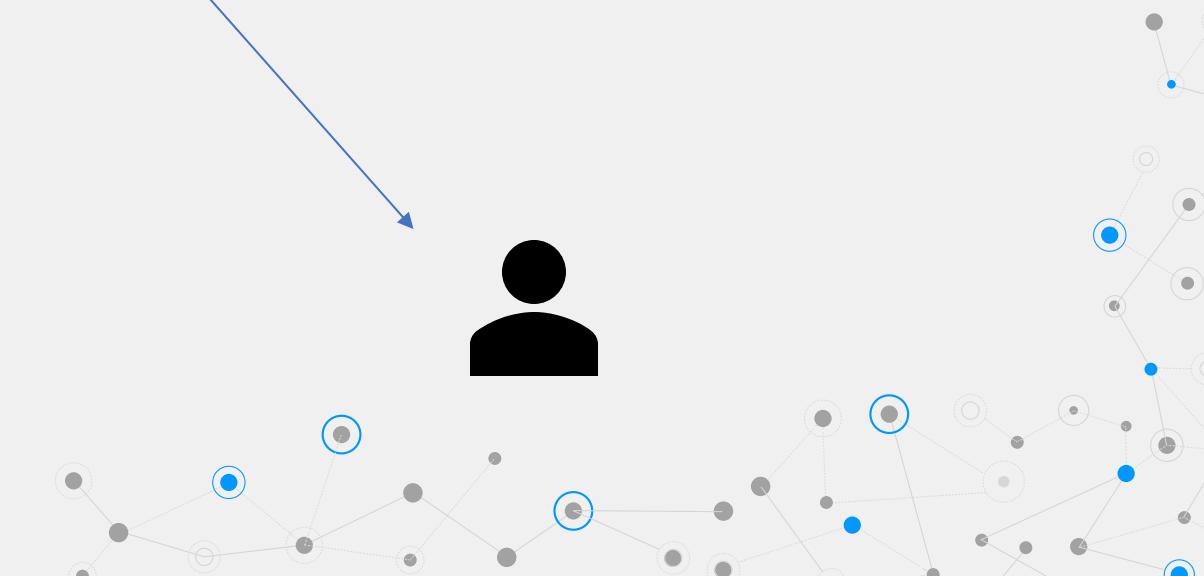
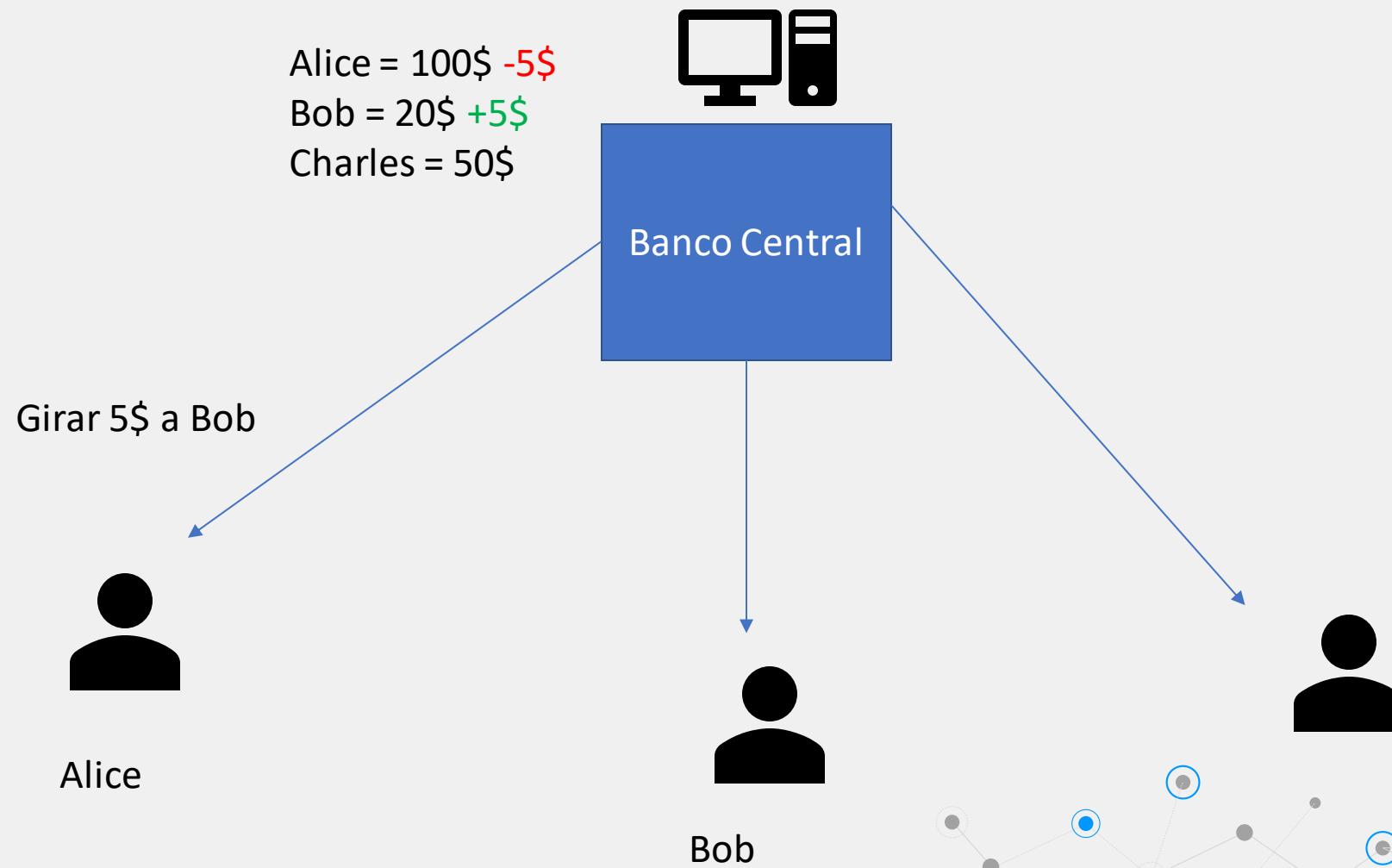
Es fundamentalmente:

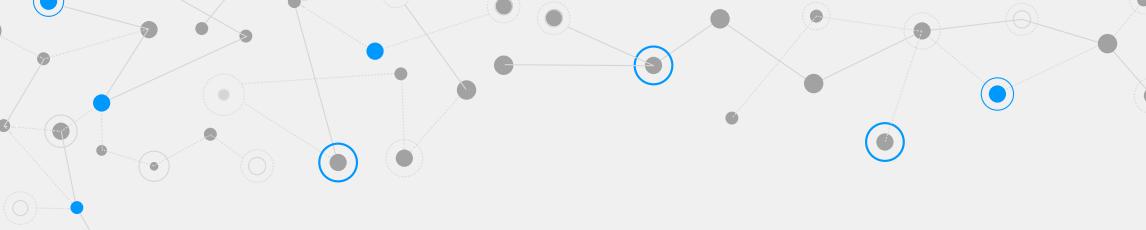
- La idea de usar cadenas de bloques para mantener integridad de historial de transacciones.
- El uso de "Proof of Work" como un algoritmo de consenso para mantener estado global descentralizado.





2. En una aplicación centralizada





2. Bitcoin

Problema fundamental de las cripto-monedas hasta 2009:

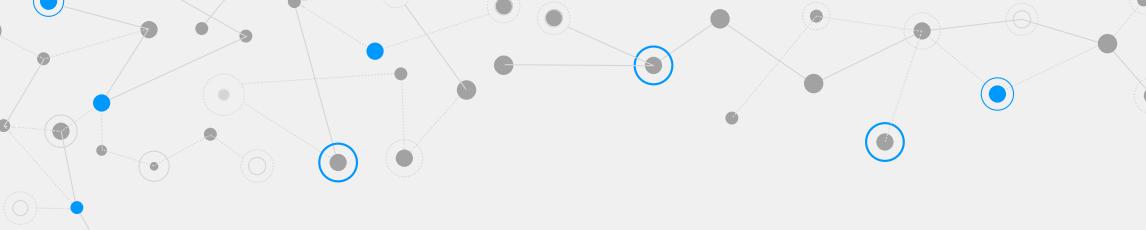
- Supongamos que una moneda es un mensaje criptográfico firmado por alguna autoridad:

$\text{Sign}\{1 \text{ moneda a nombre de Alice}\}_{\text{CA}}$

- Supongamos que tengo una llave privada/llave pública que permite “firmar” transacciones:

Le giro 1 moneda $\text{Sign}\{1 \text{ moneda a nombre de Alice}\}_{\text{CA}}$
a Bob:::(Firmado por Alice)

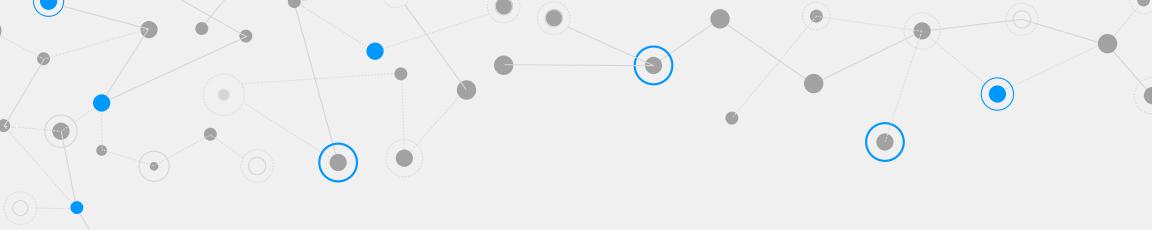




2. Bitcoin

Cómo controlo que una moneda no se gaste dos veces?

- Soluciones previas a Bitcoin involucran una autoridad central que de alguna manera previene este fenómeno.
- Cómo lo resuelve Bitcoin?



2. Bitcoin

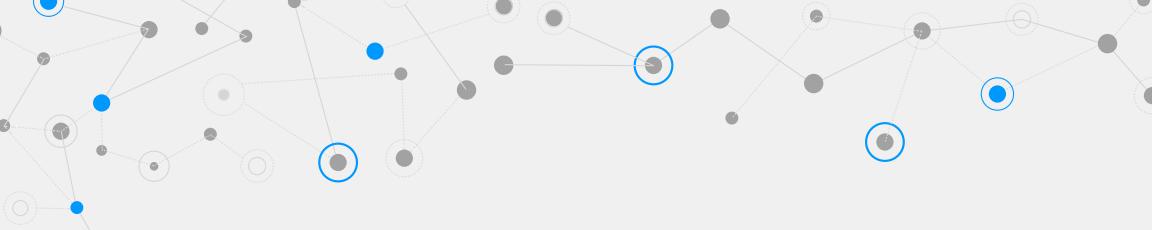
Por simplicidad, asumamos estructura de transacción (en realidad es un poco más complejo):

ID1	ID2	Valor
-----	-----	-------

Quién gira:
Identidad = Llave pública

A quién

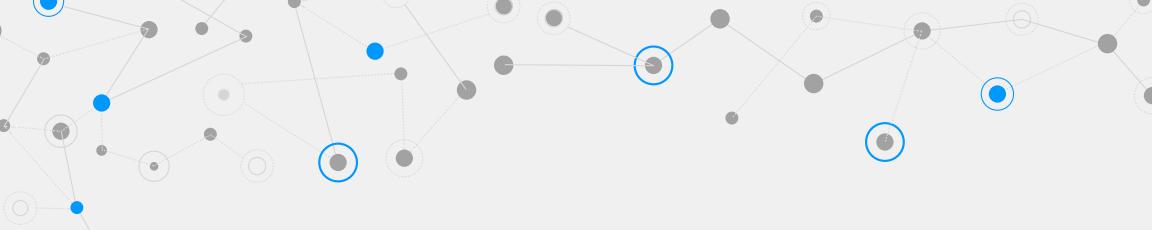
Cuánto



2. Bitcoin

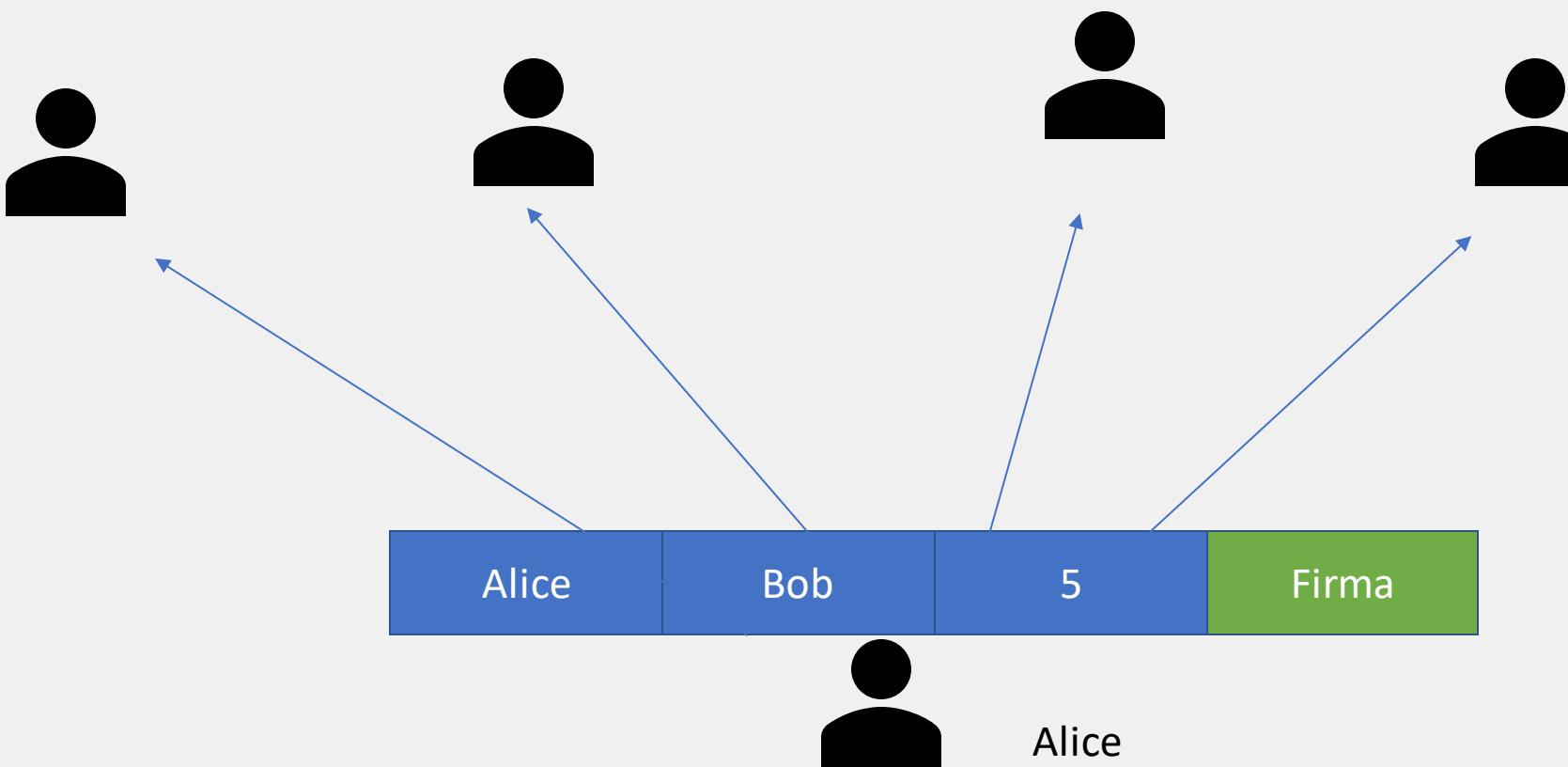
Asumamos que los bloques de la cadena de bloques son “estados de cuenta” globales:

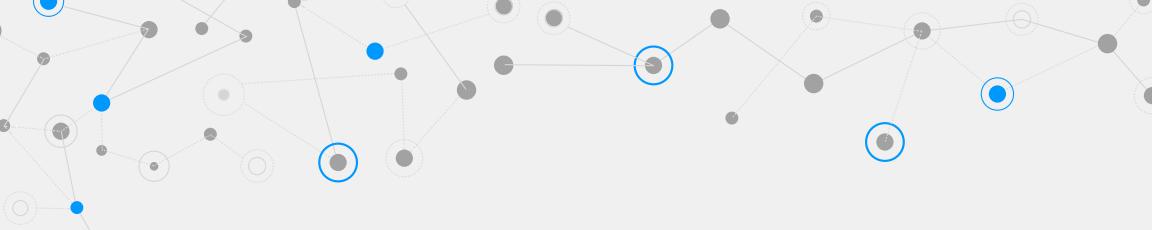
Alice	100\$
Bob	50\$
Charles	10\$



2. Bitcoin

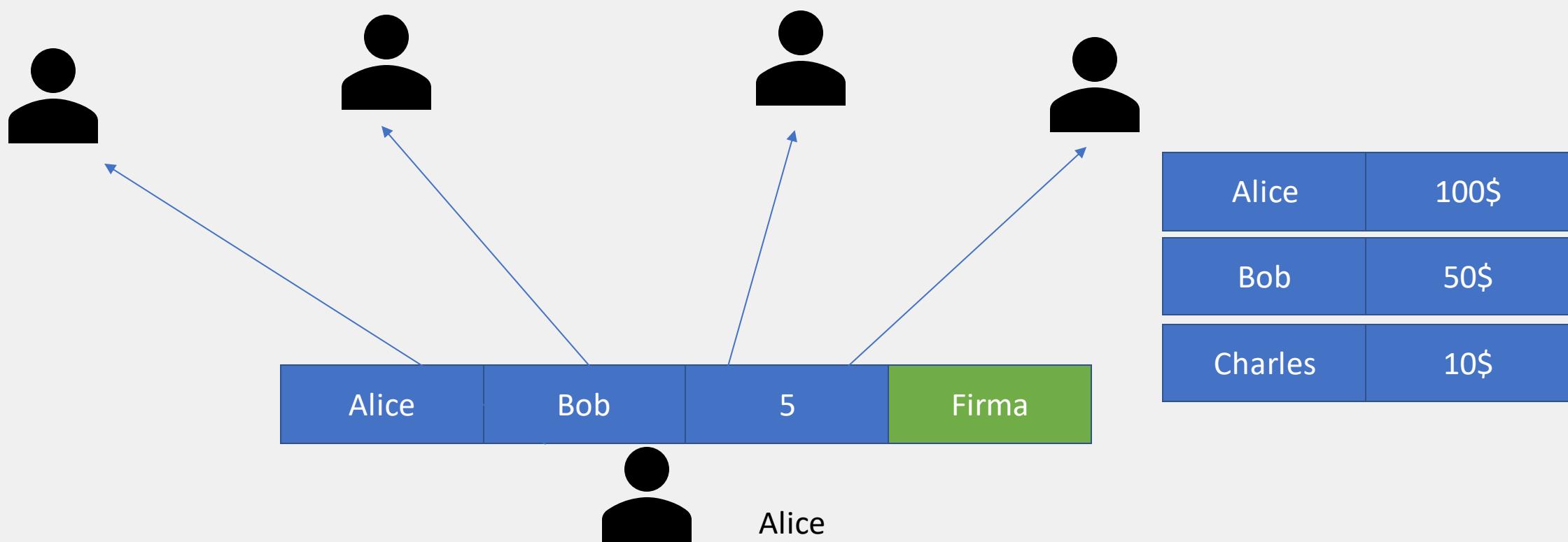
Si quiero hacer una transacción, lo comparto con todos los nodos de una red:

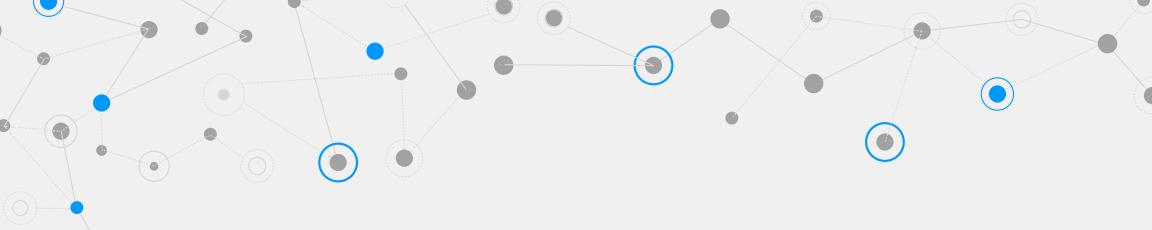




2. Bitcoin

Los nodos acumulan las transacciones que ven en un período de tiempo. Cualquier nodo puede validar si la transacción es autentica (firma), y si la cuenta tiene dinero (bloque anterior).

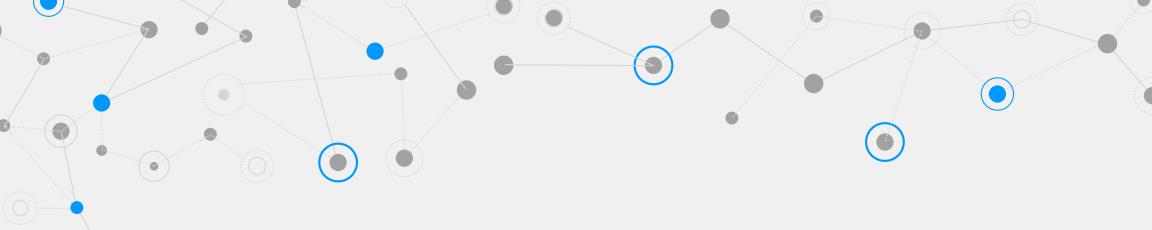




2. Bitcoin

Para generar un nuevo bloque valido, los nodos compiten por encontrar un valor, *nonce*, talque el siguiente puntero (hash) sea menor a un cierto valor.





2. Bitcoin

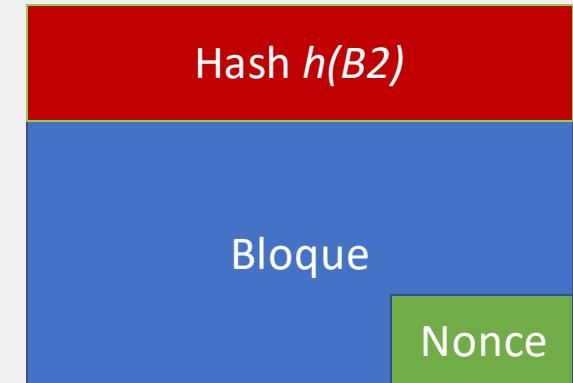
Cuando alguien encuentra un nonce para añadir bloque, lo propaga en la red.

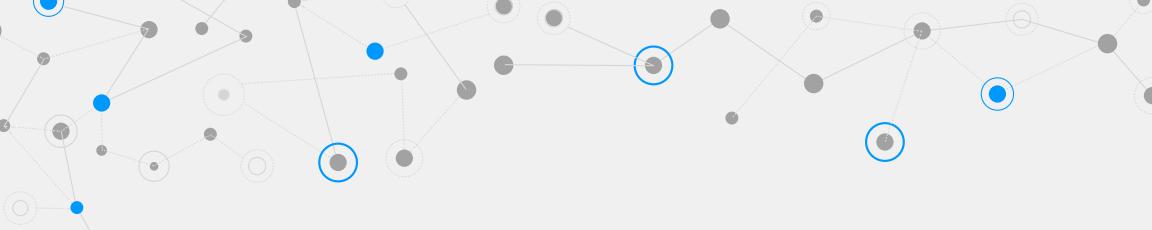
Todos los nodos pueden verificar:

- Que el nonce y el hash resultante son validos.
- Que el bloque resultante es valido
(las transacciones eran correctas)

Si alguna de las anteriores no se cumple, pueden rechazar la actualización. Si la aceptan, añaden el nodo a su cadena local (consenso implícito).

B3

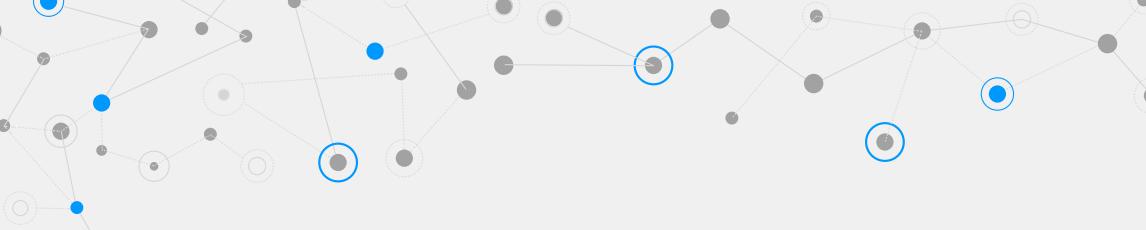




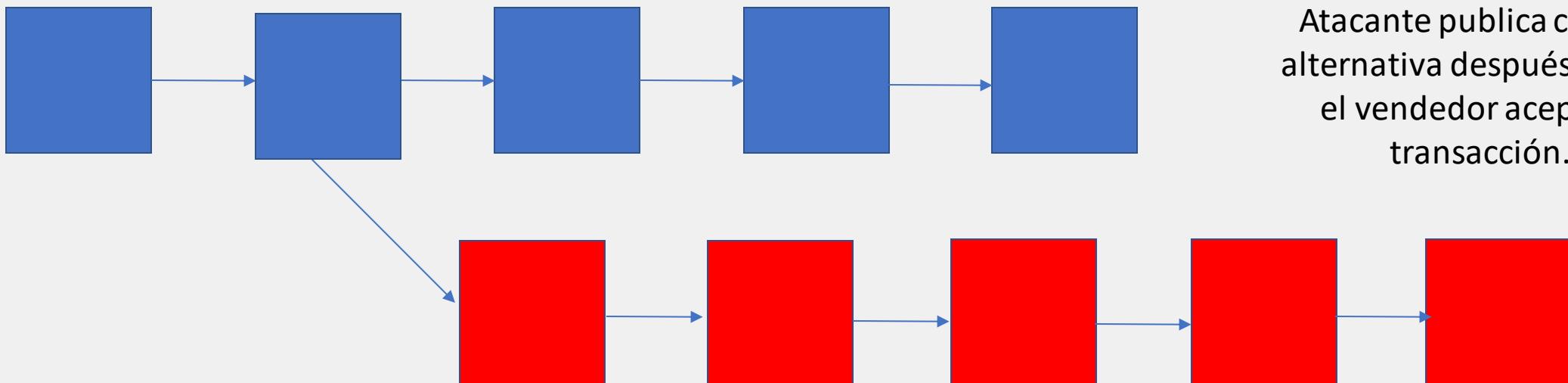
2. Bitcoin: Seguridad de Bitcoin

La seguridad de Bitcoin quiere decir:

- Nadie puede gastar más de lo que tiene.
- Solo quién posee la llave privada puede firmar transacciones.
- El estado global se mantiene siempre y cuando la mayoría de los nodos cooperen y acepten solo bloques validos.
- Si una mayoría decide aceptar bloques inválidos, se rompe la cadena.

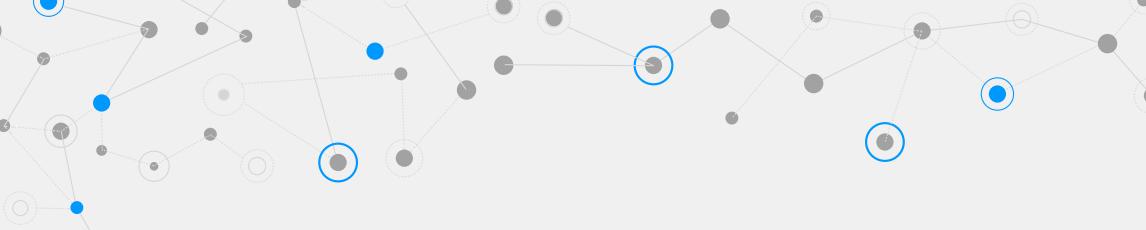


2. Ataque del 51%



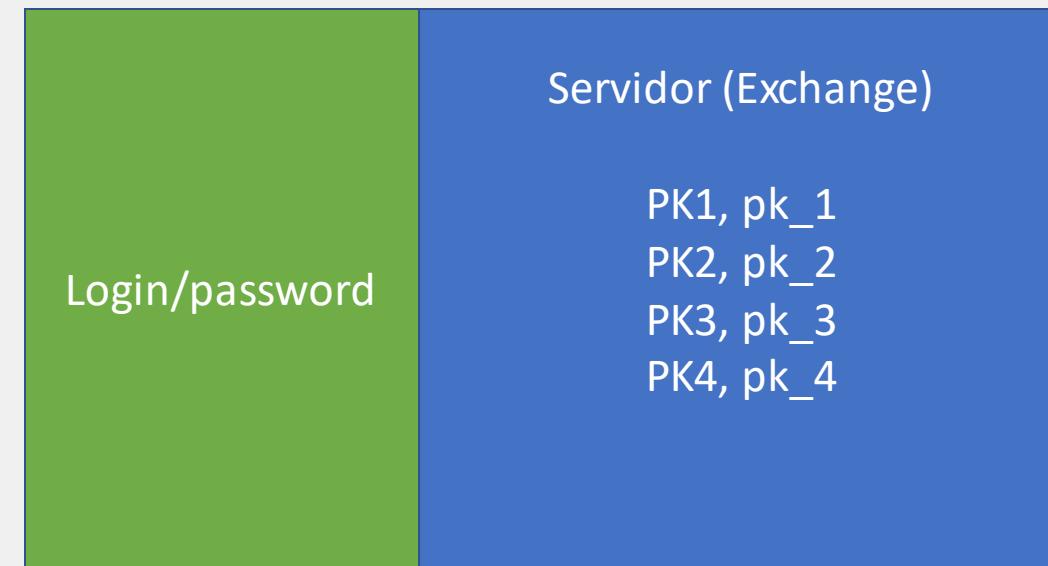
Atacante publica cadena alternativa después de que el vendedor aceptó la transacción.

Si el atacante puede generar una cadena alternativa puede "reversar" transacciones
-> La red empieza a ampliar la cadena más larga



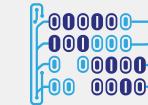
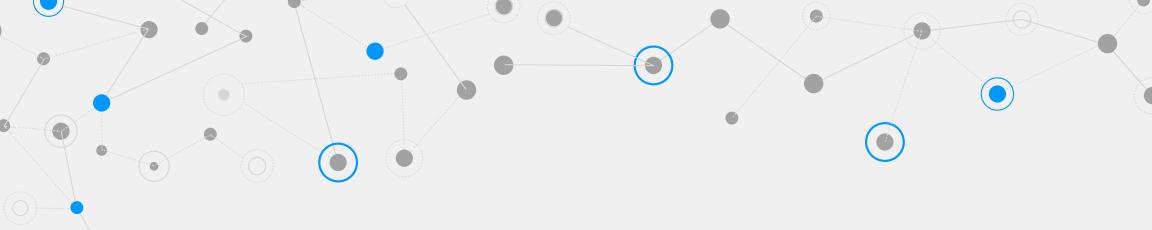
2. Bitcoin: Seguridad de Bitcoin

A la fecha, los problemas de seguridad más frecuentes con bitcoin son el robo de llaves privadas.



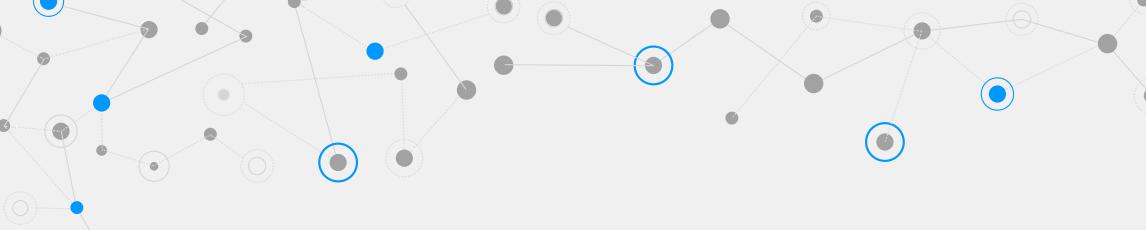
Exfiltración datos





2. Taller: Proof of Work

- Construya una función que encuentre un Nonce tal que el hash del mensaje:
"Aplicaciones Blockchain UR"
concatenado con el Nonce tenga un 0 como prefijo.
- Extienda su función a 2,3,4 0's como prefijo. ¿Qué observa?



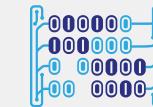
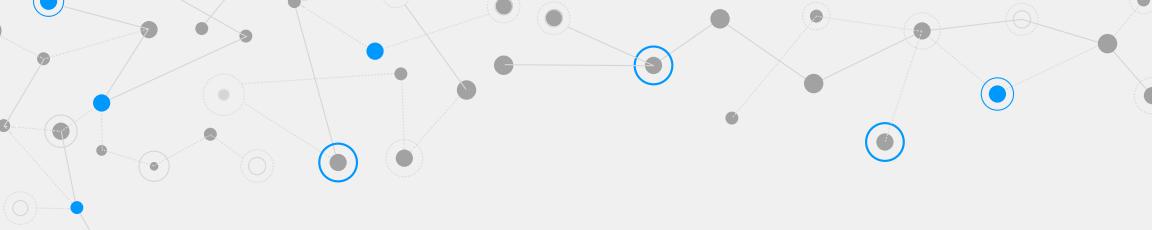
3. Smart Contracts

Generalización de idea de Blockchain donde un programa se ejecuta de forma distribuida.

Ethereum permite escribir programas "Turing Completos"

- Basicamente tiene un lenguaje de scripting (Solidity, parecido a JavaScript) que tiene
 - Ciclos
 - If-else
 - Variables
 - Llamadas a otros contratos (librerías)
- Se puede calcular cualquier cosa (como una función de hash, un algoritmo de ordenamiento etc.).





3. Smart Contracts

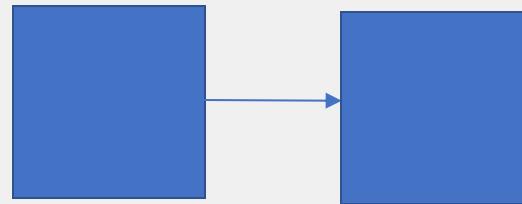
Ejemplos:

- Contratos entre dos personas: Alice le presta 1 ETH a Bob, Bob en 1 mes paga un interés a Alice de 0.1 ETH.
- Cuando existan donaciones por 100 ETH a una cierta cuenta, automáticamente gire la mitad a fundación A y la otra mitad a la fundación B.
- Si mi vuelo está retrasado más de 2 horas, automáticamente el seguro me paga en ETH.



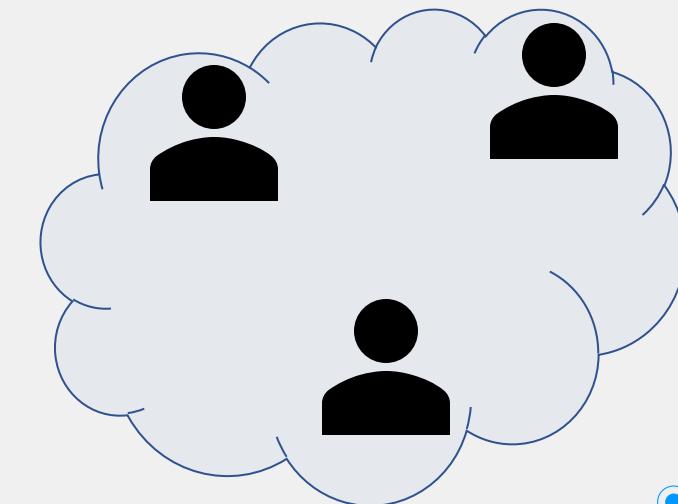
3. Smart Contracts

¿Cómo se ejecutan los contratos?



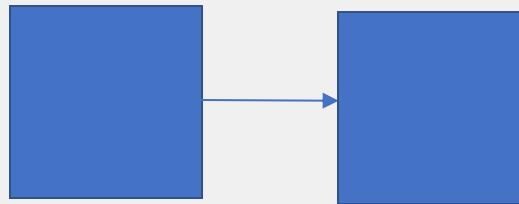
Un nuevo bloque es un nuevo "estado". Al ejecutar un programa con ciertos inputs (estado inicial) se llega un nuevo estado.

- Los mineros (quienes generan nuevos bloques) ejecutan el contrato.
- Toda la red puede verificar que el contrato fue ejecutado correctamente.



3. Smart Contracts

¿Cómo evitar abusos o programas que no paren?



Cada programa tiene un "precio" para ejecutarse. Este es el incentivo para que el minero ejecute la función.

- El precio en Ethereum se llama "Gas"
 - El "Gas" tiene un equivalente en ETH.



3. Smart Contracts : seguridad

Ethereum hereda muchas de las propiedades de seguridad de Bitcoin (aunque los detalles del consenso son diferentes).

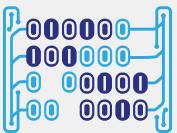
Sin embargo, Ethereum añade un grado de complejidad que puede introducir problemas:

- ¿Cómo estamos seguros de que la lógica del programa que escribimos no tiene errores?
- En la práctica errores sutiles en la programación de contratos han dado pie a ataques.
 - Ejemplo más famoso es la app "Parity" (Dos ataques en 2017: perdidas por 300MM USD).
 - Para ver más detalles leer: <https://eprint.iacr.org/2016/1007.pdf>

Gracias



Universidad del
Rosario



MACC
Matemáticas Aplicadas y
Ciencias de la Computación