

Blockchains and Decentralized Applications

Thierry Sans

How Web3 Is Transforming The Future Of Business



Aleksandrs Malins Forbes Councils Member

Forbes Technology Council

COUNCIL POST | Membership (Fee-Based)

Apr 3, 2023, 06:45am EDT

Advantages Of Web3

From my experience implementing blockchain technology in projects, I've found that Web3 offers the following advantages.

- The main advantage is decentralization. This could refer to the networks, finance, identity, storage and more.
- It provides us with more secure and transparent systems because of blockchain implementation. Fewer people are involved in the processes, and they can view the transactions or transfers in real time.
- It offers the possibility to create digital assets such as NFTs to use in the metaverse.

Challenges Of Web3

The challenges I faced while developing Web3 projects included the following.

- The technology is complex, and it can be hard to find the right specialists.
- There's a lack of regulatory clarity (e.g., for game development and tokenization).

FORBES DIGITAL ASSETS • EDITORS' PICK

Web3 Growth Stymied By Scarcity Of Programmers

Nina Bambysheva Forbes Staff*I cover cryptocurrencies and other applications of blockchain*[Follow](#)

Aug 29, 2022, 07:30pm EDT

f

t

in



Developer shortage stands in the way of decentralized, blockchain-centric internet bliss. GETTY

Advocates of Web3, a catch-all term widely used to incorporate concepts of decentralized networks, cryptocurrencies and other blockchain-powered applications, have a grand vision for the future of the Internet and global finance.

One thing that stands in the way: a lack of people to make it happen.

Cryptocurrencies

Total cryptocurrency market cap



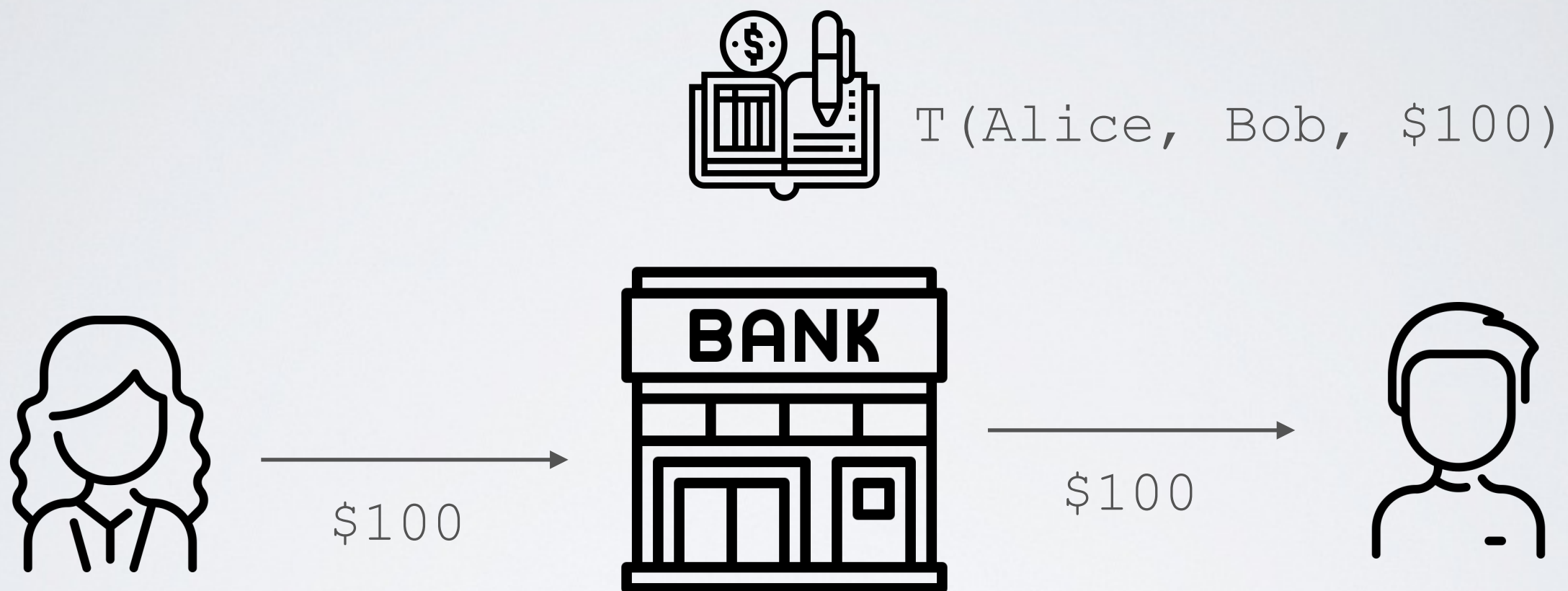
Bitcoin: A Peer-to-Peer Electronic Cash System

Satoshi Nakamoto
satoshin@gmx.com
www.bitcoin.org

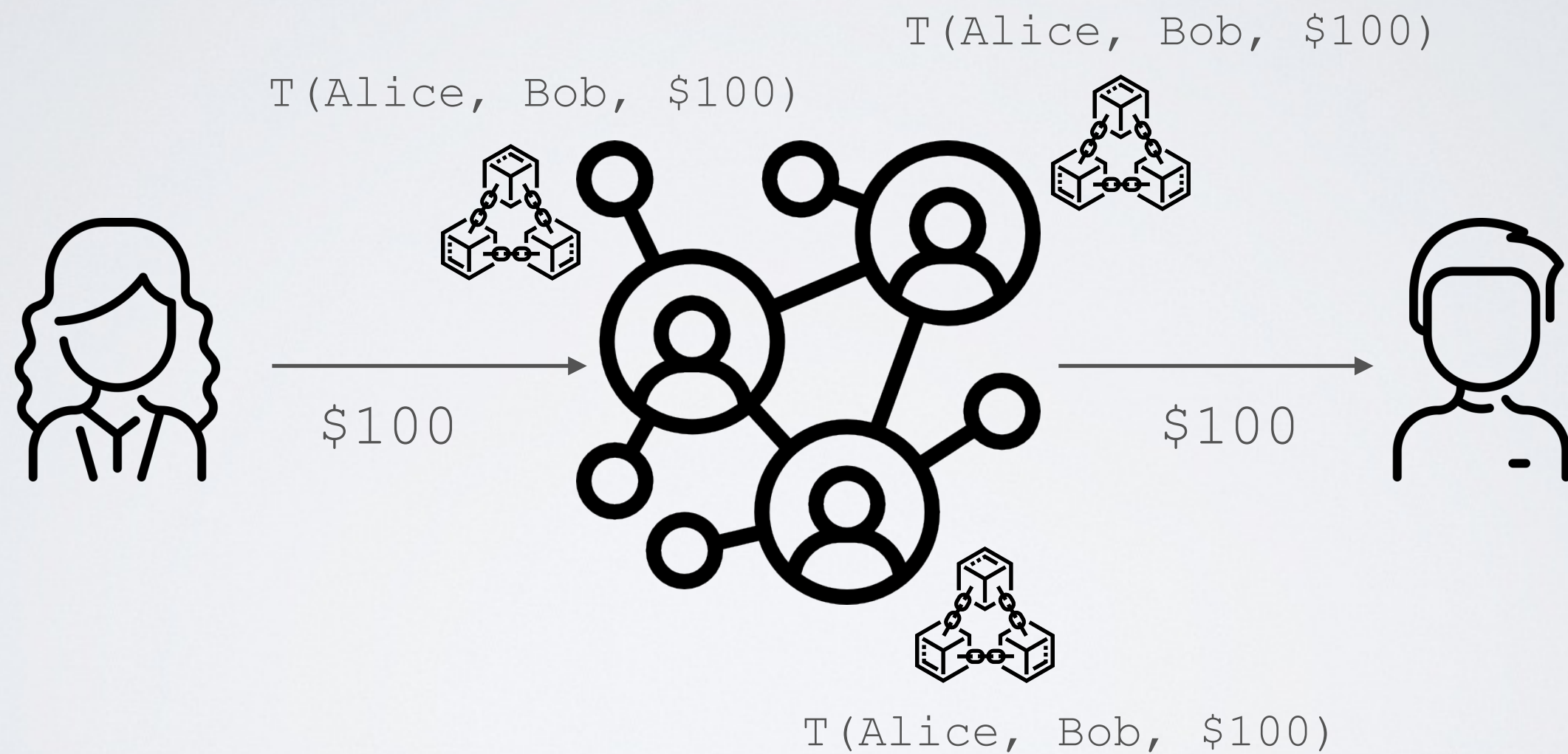
The original
Bitcoin paper
(2008)

Abstract. A purely peer-to-peer version of electronic cash would allow online payments to be sent directly from one party to another without going through a financial institution. Digital signatures provide part of the solution, but the main benefits are lost if a trusted third party is still required to prevent double-spending. We propose a solution to the double-spending problem using a peer-to-peer network. The network timestamps transactions by hashing them into an ongoing chain of hash-based proof-of-work, forming a record that cannot be changed without redoing the proof-of-work. The longest chain not only serves as proof of the sequence of events witnessed, but proof that it came from the largest pool of CPU power. As long as a majority of CPU power is controlled by nodes that are not cooperating to attack the network, they'll generate the longest chain and outpace attackers. The network itself requires minimal structure. Messages are broadcast on a best effort basis, and nodes can leave and rejoin the network at will, accepting the longest proof-of-work chain as proof of what happened while they were gone.

A centralized ledger (Trusted Third Party)



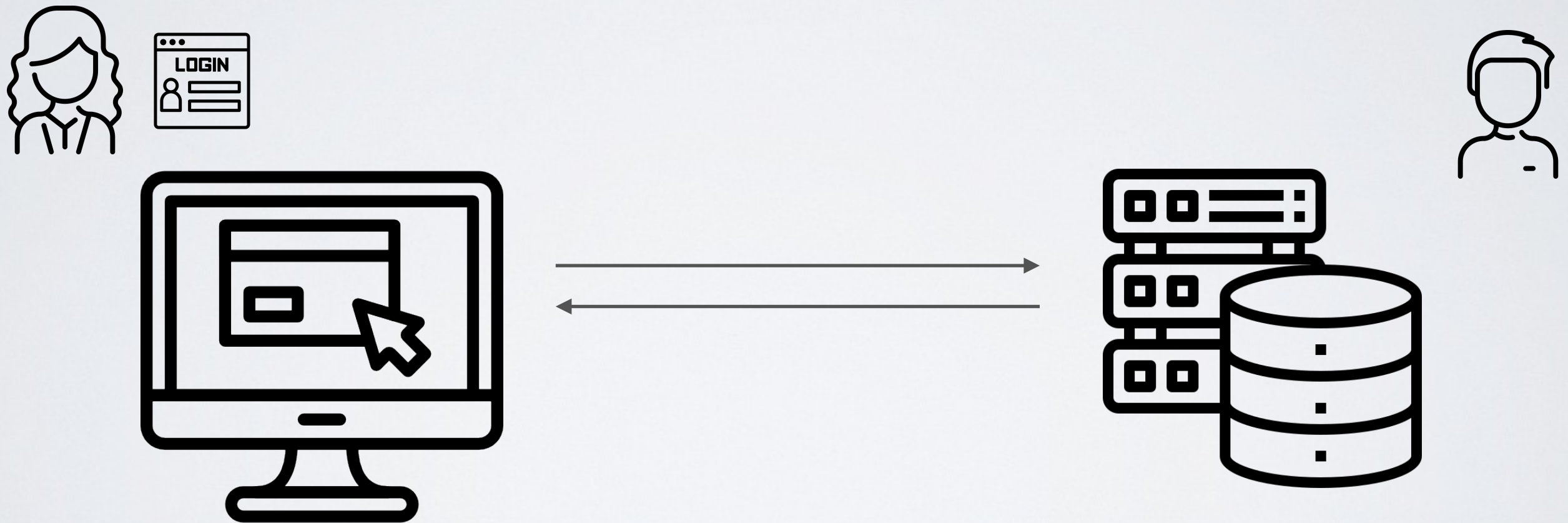
A decentralized ledger (Trustless)



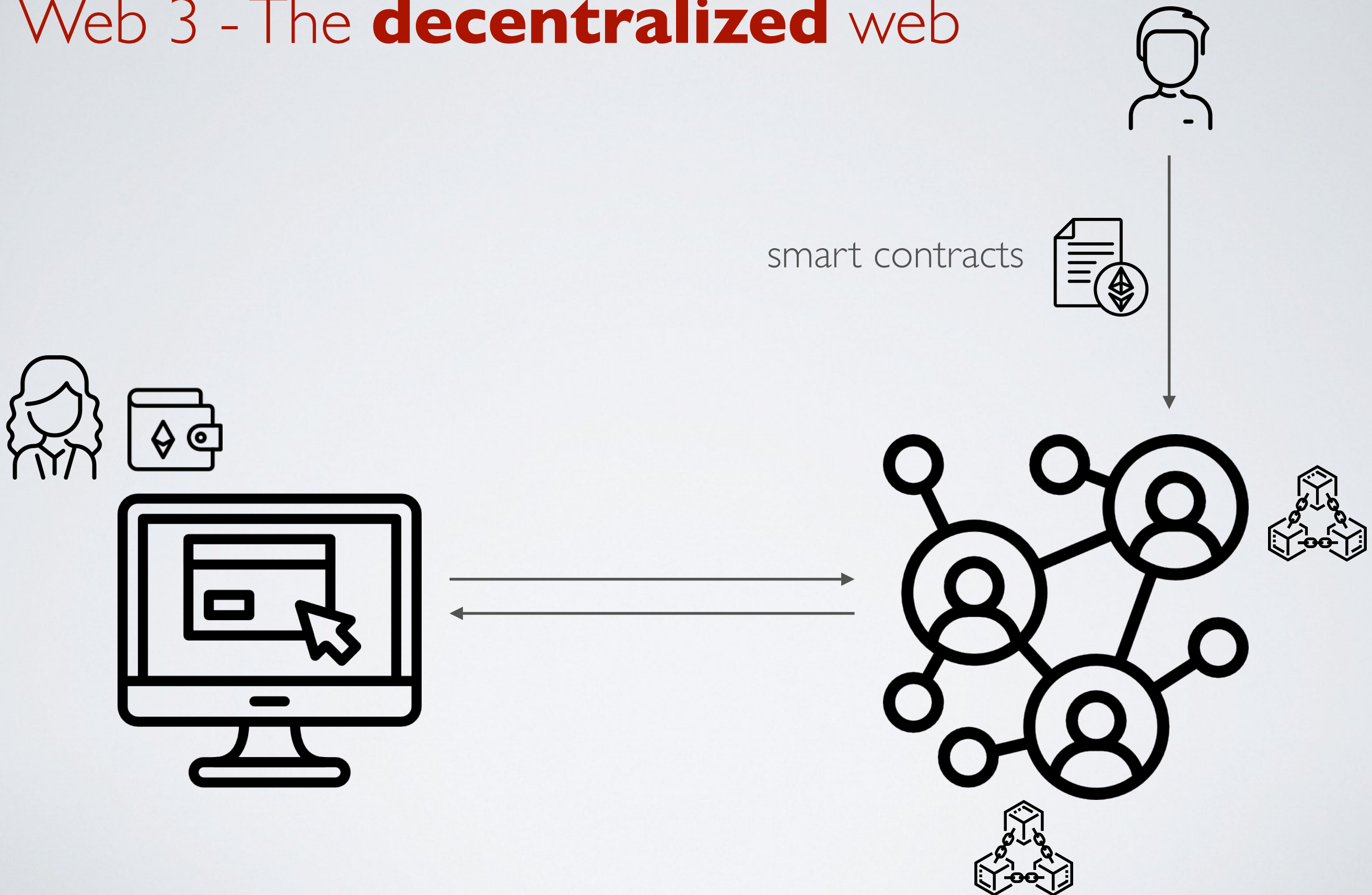
Beyond Cryptocurrencies

Towards Decentralized Applications

Web 2 - The centralized Web



Web 3 - The **decentralized** web



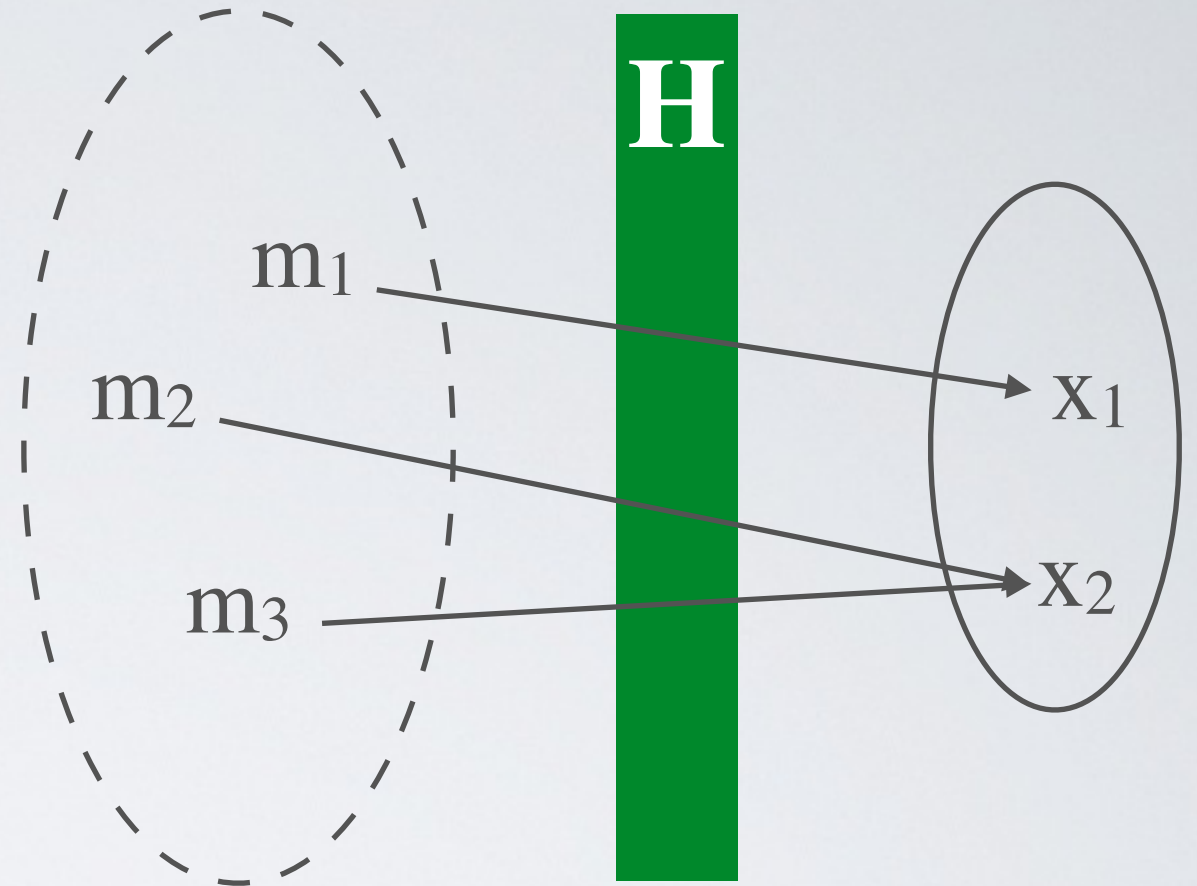
Cryptography Toolbox

The cryptography toolbox has many building blocks ...

... but here we only need:

- Hashing
- Digital Signature

Cryptographic Hashing



$H(m) = x$ is a hash function if

- m is a message of any length
- x is a message digest of a fixed length
- H is a non invertible function

➡ H is a lossy compression function
necessarily there exists x, m_1 and $m_2 \mid H(m_1) = H(m_2) = x$

Computational Properties



- ✓ Given H and m , computing x is **easy** (polynomial or linear)
- ⊙ Given H and x , computing m is **hard** (exponential)
- ⊙ Given H , m and x , it is hard (exponential) to find m' such that $H(m) = H(m') = x$
- ⊙ Given H , it is hard (exponential) to find m and m' such that $H(m) = H(m') = x$

Digital Signatures

$(pk_A, sk_A) = \text{generateKeyPair}()$

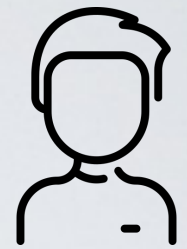


$\text{sig} = \text{sign}(m, sk_A)$

m, sig, pk_A



pk_A



pk_A

$\text{verify}(m, \text{sig}, pk_A)$

Only Alice can sign a message m with her secret key sk_A

➔ Everybody can verify m using Alice's public key pk_A

Computational Properties

- ✓ $(pk, sk) = \text{generateKeyPair}()$
is easy to compute (polynomial)
- ✓ $sig = \text{sign}(m, sk_A)$
is easy to compute (either polynomial or linear)
- ✓ $\text{verify}(m, sig, pk_A)$
is easy to compute (either polynomial or linear)
- Finding a matching key sk , given pk is hard (exponential)
- Forging a valid signature without knowing sk is hard (exponential)

Naive Cryptocurrency

Alice's Blockchain

oldH: null
from: null
to: aGks5
amount: 100
newH: uUiN1

Genesis Block

Alice's Keypair

pk_B: aGks5
sk_B: ...



Alice's Tracker

aGks5: 100



Bob's Keypair

pk_B: bJR5H
sk_B: ...



Bob's Tracker

aGks5: 100

Bob's Blockchain



Alice's Blockchain

oldH: null
from: null
to: aGks5
amount: 100
newH: uUiN1

Genesis Block

oldH: uUiN1
from: aGks5
to: bJR5H
amount: 20
newH: dSm3LJ

Alice's Key pair

pk_B: aGks5
sk_B: ...



Alice's Tracker

aGks5: 80
bJR5H: 20



Bob's Keypair

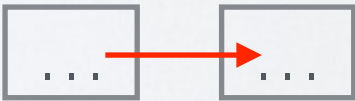
pk_B: bJR5H
sk_B: ...



Bob's Tracker

aGks5: 80
bJR5H: 20

Bob's Blockchain



Transaction validation

A transaction is valid if

- the oldHash match the last block hash (chaining)
- the signature of the issuer (from) is valid
- the account has the necessary balance

Alice's Blockchain

oldH: null

from: null
to: aGks5
amount: 100

newH: uUiN1

Genesis Block

oldH: uUiN1

from: aGks5
to: bJR5H
amount: 20

newH: dSm3LJ

oldH: dSm3LJ

from: bJR5H
to: m406Z
amount: 10

newH: 7fLvX

Alice's Key pair

pk_B: aGks5
sk_B: ...



Alice's Tracker

aGks5: 80
bJR5H: 10
m406Z: 10



Malory's Keypair



pk_M: m406Z
sk_M: ...

Malory's Tracker

aGks5: 80
bJR5H: 10
m406Z: 10

Mallory's Blockchain



Bob's Keypair

pk_B: bJR5H
sk_B: ...



Bob's Tracker

aGks5: 80
bJR5H: 10
m406Z: 10

Bob's Blockchain



Concurrency issues

oldH: 7fLvX
from: aGks5
to: m406Z
amount: 10
newH: P+q2C

Alice's Key pair

pk_B: aGks5
sk_B: ...



Solution: consensus algorithm

Bob's Keypair

pk_B: bJR5H
sk_B: ...



oldH: 7fLvX
from: bJR5H
to: m406Z
amount: 10
newH: vVxL6

Malory's Keypair

pk_M: m406Z
sk_M: ...



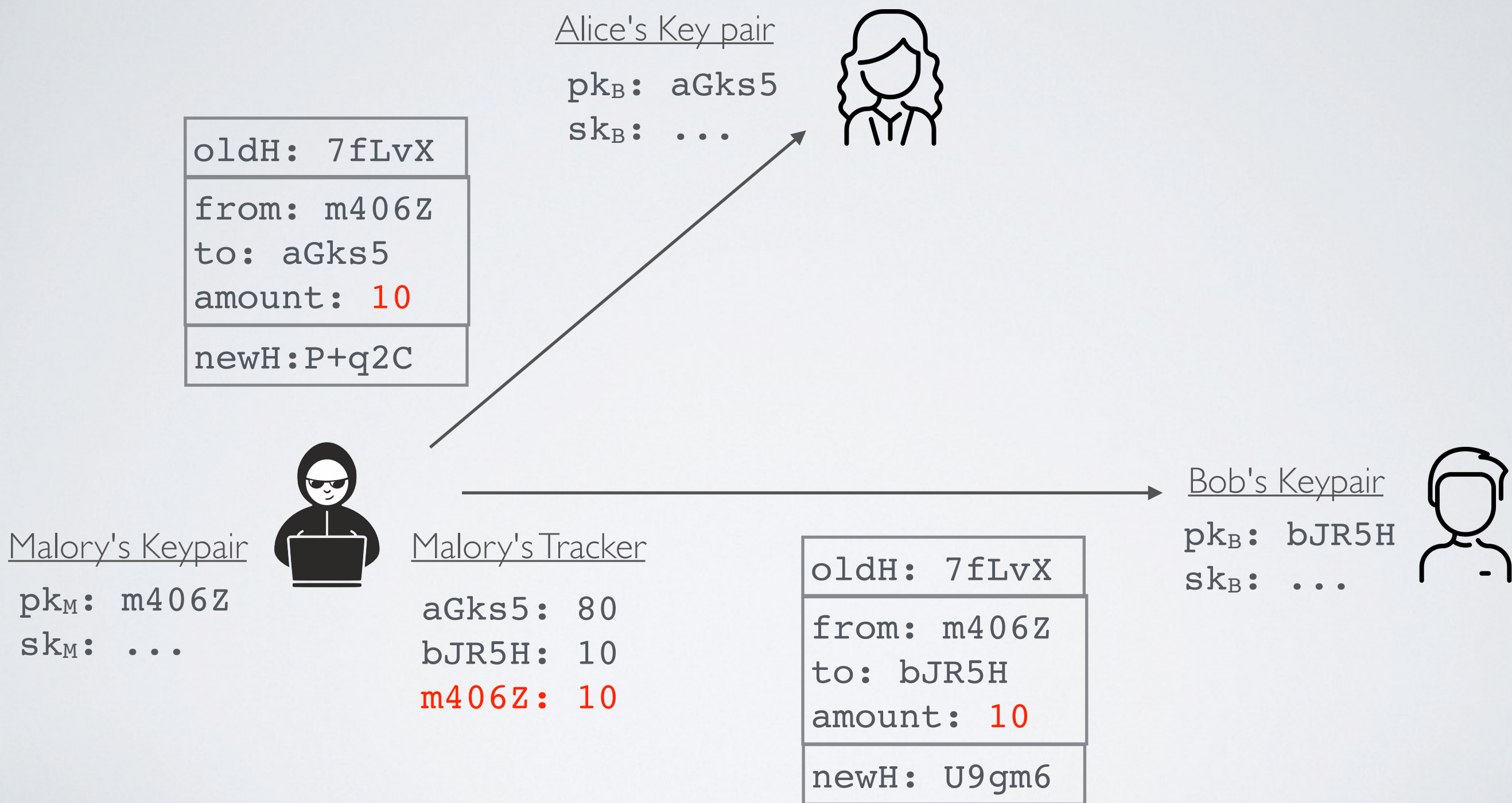
Malory's Tracker

aGks5: ?
bJR5H: ?
m406Z: ?

Mallory's Blockchain



Double Spending Attack



The original Bitcoin paper (2008)

Bitcoin: A Peer-to-Peer Electronic Cash System

Satoshi Nakamoto
satoshin@gmx.com
www.bitcoin.org

Abstract. A purely peer-to-peer version of electronic cash would allow online payments to be sent directly from one party to another without going through a financial institution. Digital signatures provide part of the solution, but the main benefits are lost if a trusted third party is still required to prevent double-spending.

We propose a solution to the double-spending problem using a peer-to-peer network. The network timestamps transactions by hashing them into an ongoing chain of hash-based proof-of-work, forming a record that cannot be changed without redoing the proof-of-work. The longest chain not only serves as proof of the sequence of events witnessed, but proof that it came from the largest pool of CPU power. As long as a majority of CPU power is controlled by nodes that are not cooperating to attack the network, they'll generate the longest chain and outpace attackers. The network itself requires minimal structure. Messages are broadcast on a best effort basis, and nodes can leave and rejoin the network at will, accepting the longest proof-of-work chain as proof of what happened while they were gone.