

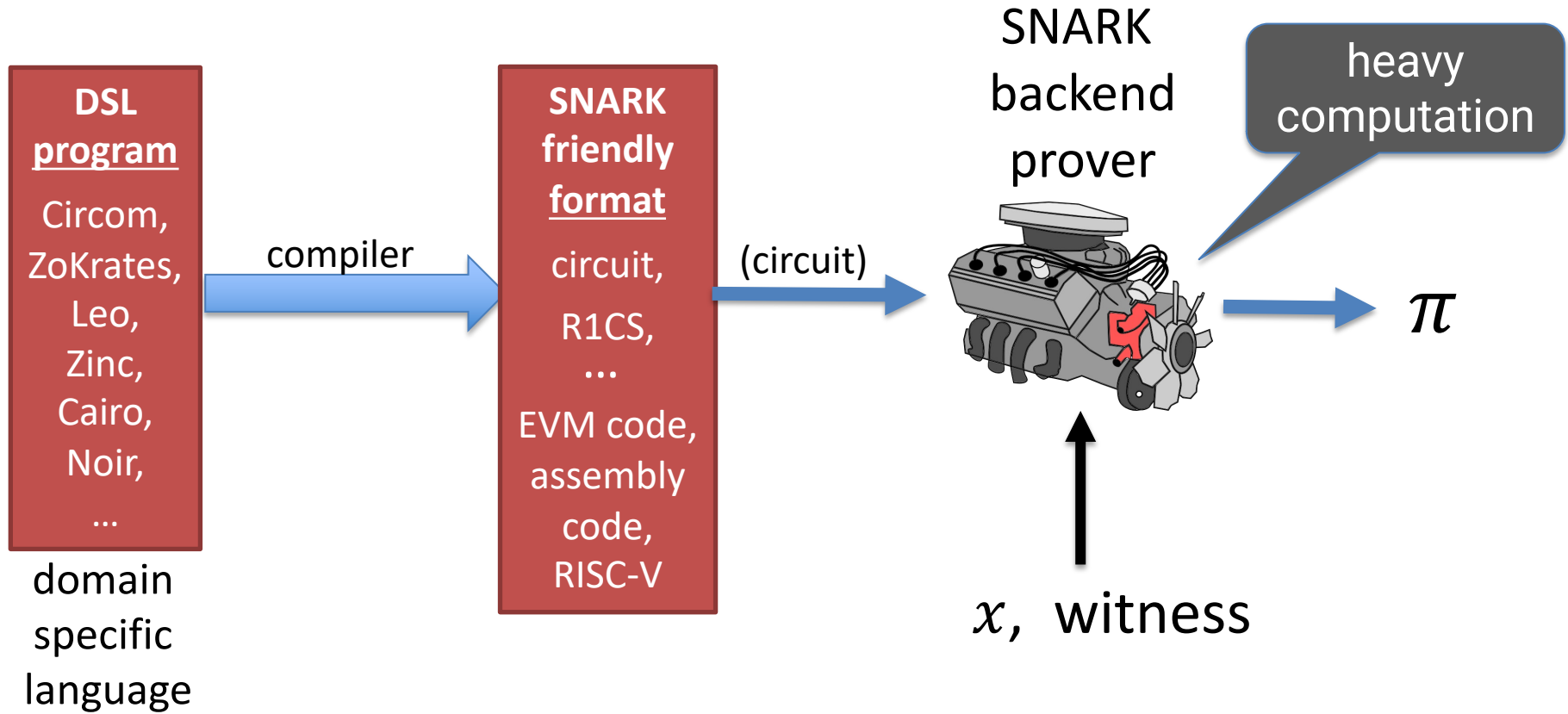
CS251 Fall 2022
(cs251.stanford.edu)



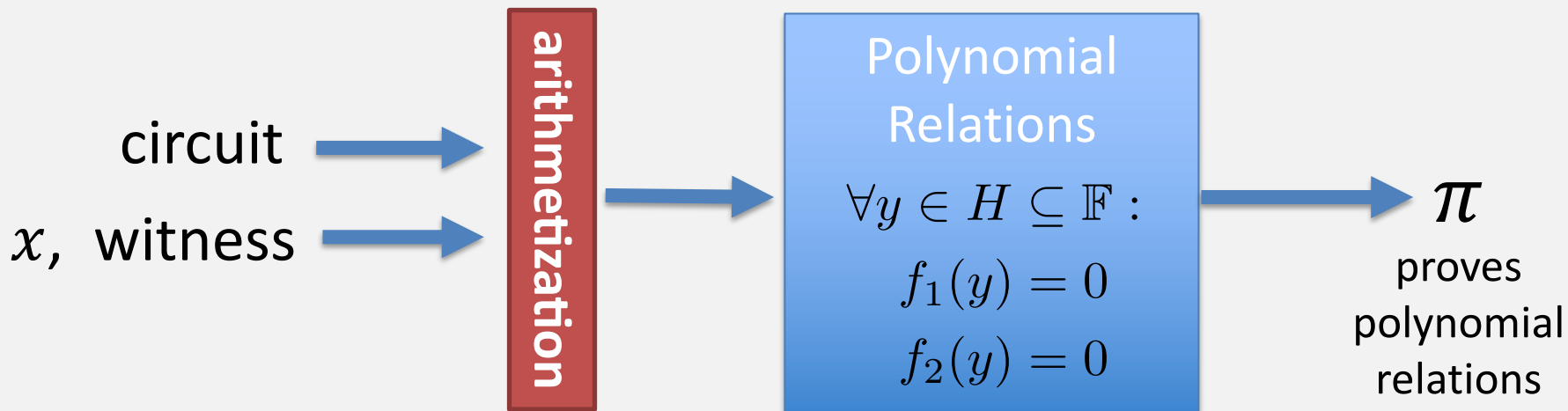
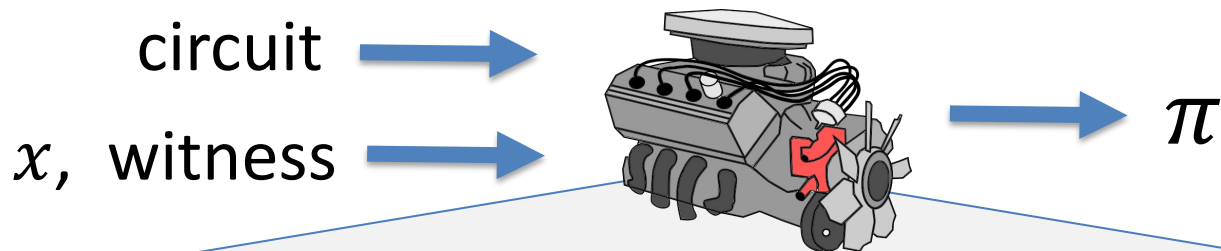
Scaling the blockchain part I: Payment Channels and State Channels

Dan Boneh

... but first, last words on SNARKs (for now)

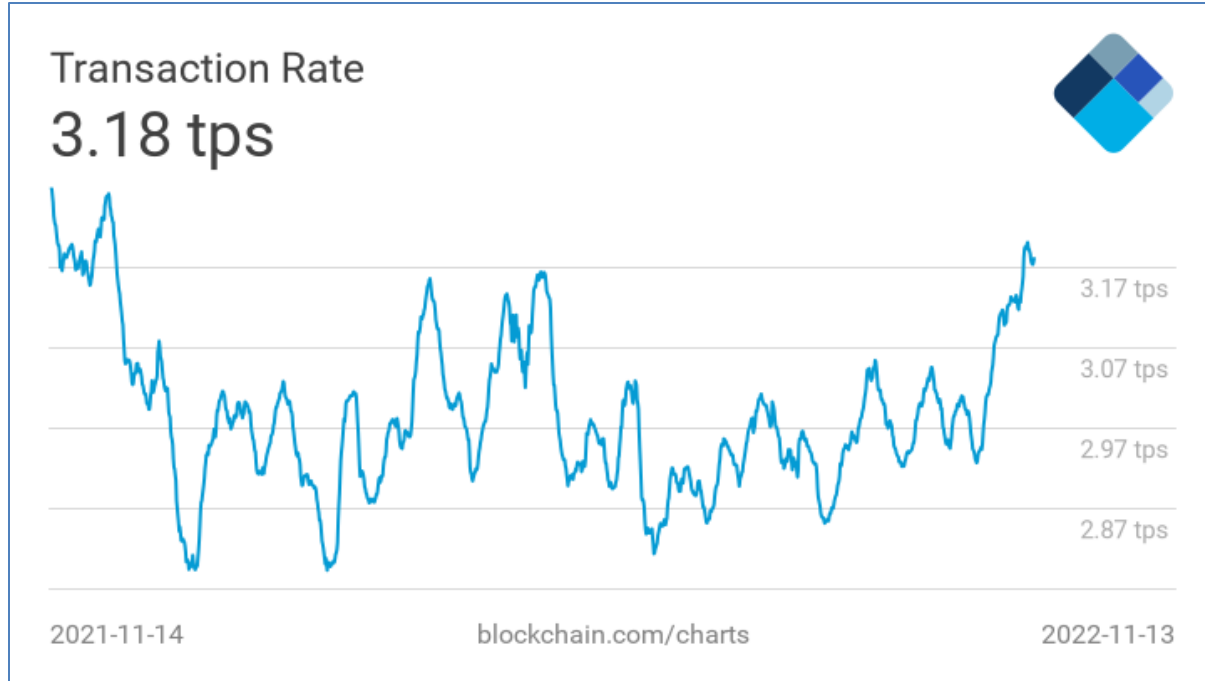


... but first, last words on SNARKs (for now)



Scaling the blockchain

Bitcoin Tx per second

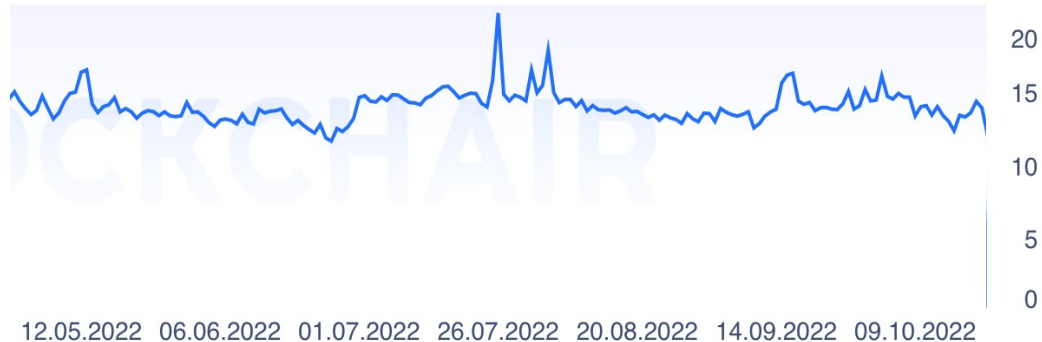


≈4200 Tx/block
1 block / 10 mins

⇒ max: 7 Tx/sec

Ethereum Tx per second

Ethereum avg Tx per second:



≈ 15 Tx/sec

Simple Tx: 21k Gas
max 30M Gas per block
⇒ max 1428 tx/block
1 Block/12s
⇒ max 119 tx/s

In comparison ...

Visa: up to 24,000 Tx/sec (regularly 2,000 Tx/sec)

PayPal: 200 Tx/sec

Ethereum: 15 Tx/sec

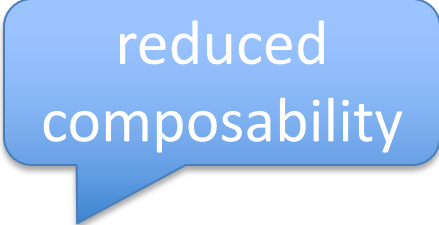
Bitcoin: 7 Tx/sec

Goal: scale up blockchain Tx speed

How to process more Tx per second

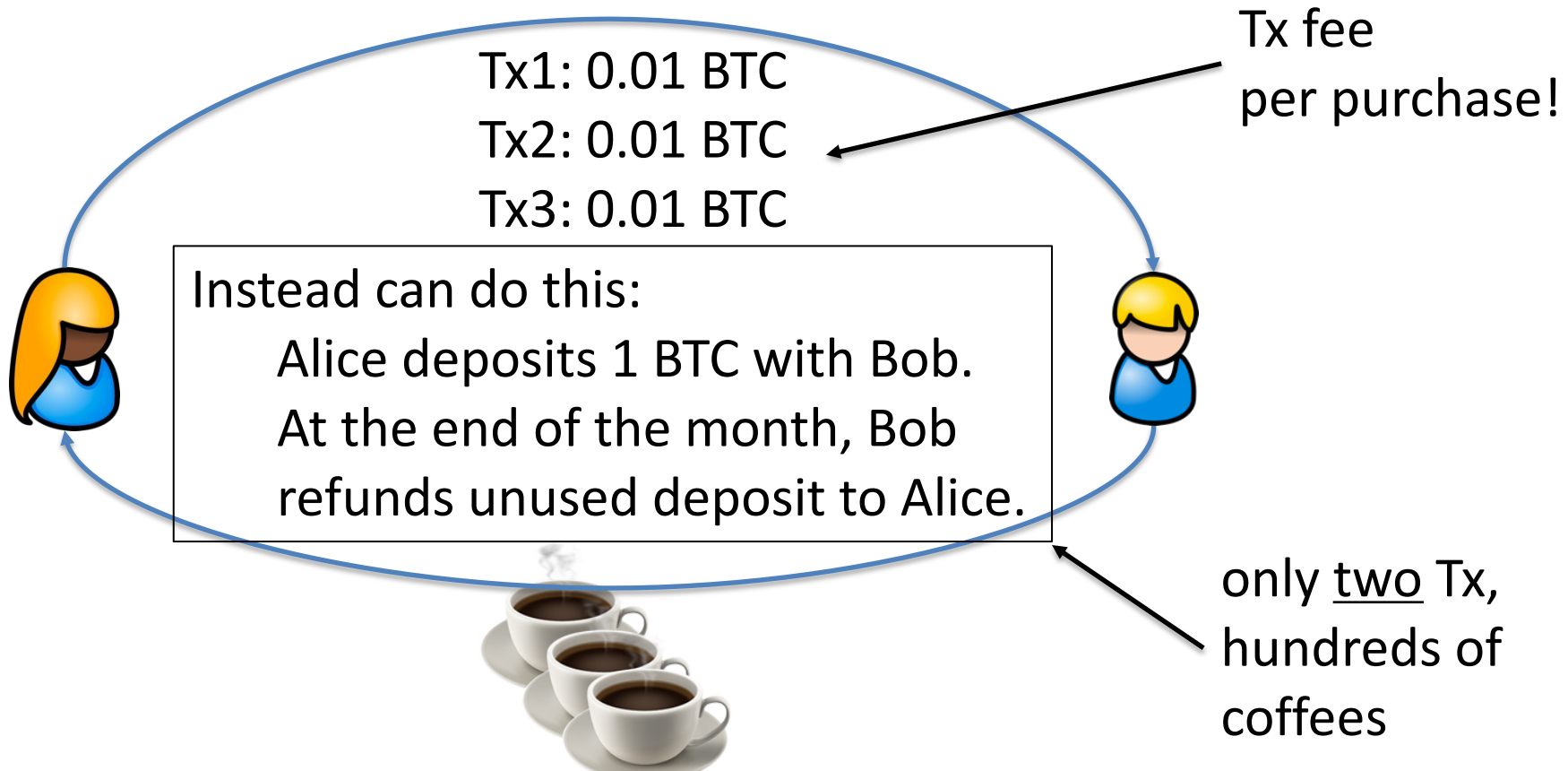
Many ideas:

- Use a faster consensus protocol
- Parallelize: split the chain into independent **shards**
- Rollup: move work somewhere else (next lecture)
- Today: payment channels, reduce the need to touch the chain



reduced
composability

Payment Channels: the basic idea



Unidirectional Payment Channel

UTXO A:
1 BTC

Bob does not post on chain

Problem: Alice spends UTXO A
before Bob publishes Tx3

Post Tx3 on Blockchain



Tx1: 0.99 to Alice / 0.01 to Bob from UTXO A
signed by Alice

Tx2: 0.98 to Alice / 0.02 to Bob from UTXO A
signed by Alice

Tx3: 0.97 to Alice / 0.03 to Bob from UTXO A
signed by Alice

A solution?

UTXO A:
1 BTC

Send 1BTC
to addr AB

UTXO AB: 2-2 Multisig
1 BTC



Problem:
What if Bob never publishes Tx3?

Sign Tx3 and publish on chain
(Tx3 signed by
both Alice & Bob)



TX1: from AB: 0.99 to Alice / 0.01 to Bob
signed by Alice

...

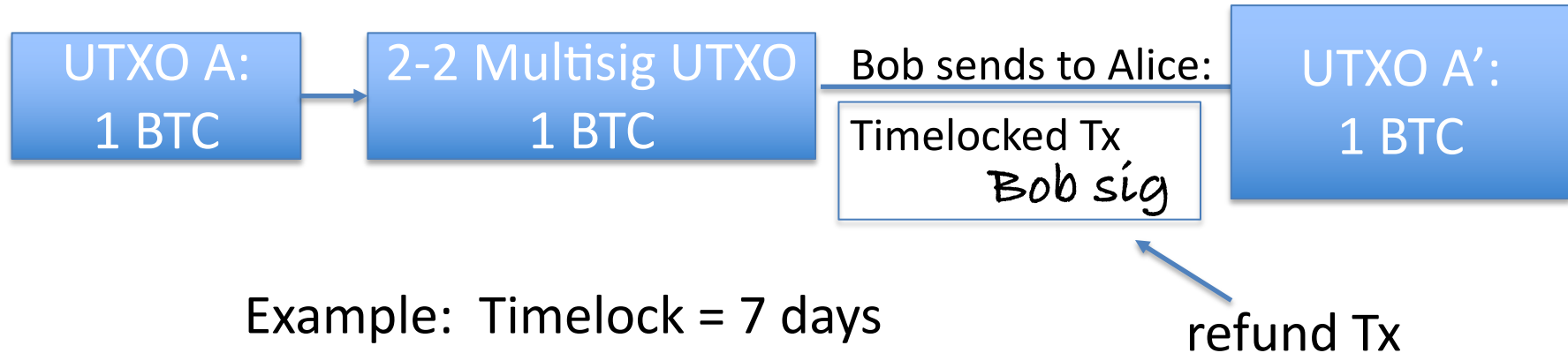
TX3: from AB: 0.97 to Alice / 0.03 to Bob
signed by Alice

Alice can't get
her 0.97 BTC back

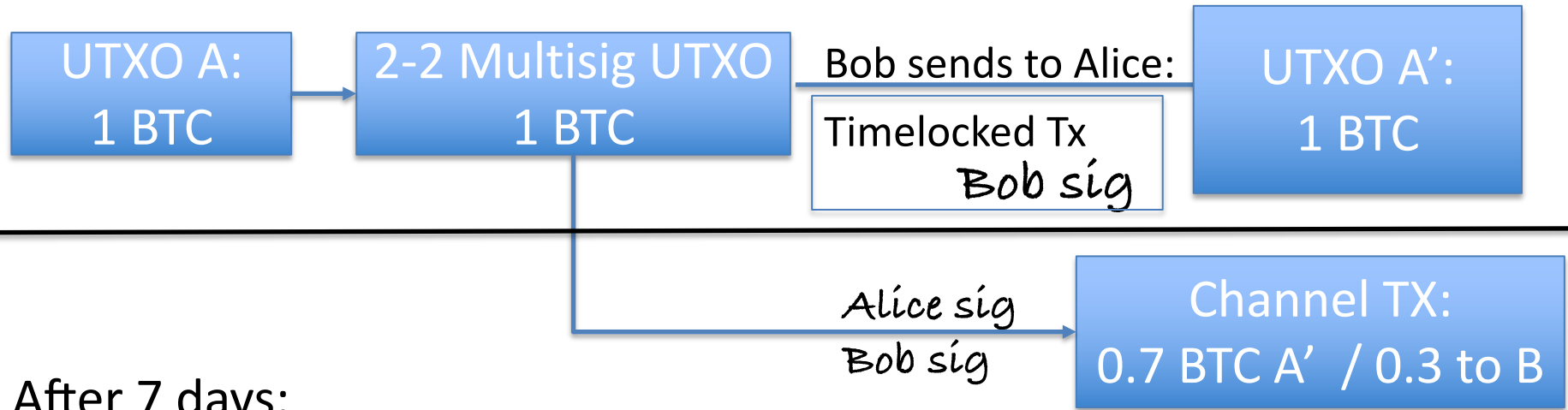
Unidirectional Payment Channel

Alice needs a way to ensure refund if Bob disappears

- Basic idea: If Bob doesn't publish Tx3 after some time, Alice gets 1 BTC refunded
- Refund transaction signed before funding Account AB



Unidirectional Payment Channel



After 7 days:

- If Alice and Bob cooperate, close channel using multisig.
- Otherwise, Alice closes channel using refund Tx, gets 1 BTC.

Note: refund TX from Bob determines lifespan of channel
Once Alice sends 1 BTC to Bob, the Channel is "exhausted"

Payment Channel in Solidity

```
1 pragma solidity >=0.4.24 <0.6.0;
2
3 contract SimplePaymentChannel {
4     address payable public sender; // The account sending payments.
5     address payable public recipient; // The account receiving the payments.
6     uint256 public expiration; // Timeout in case the recipient never closes.
7
8     constructor (address payable _recipient, uint256 duration)
9         public
10        payable
11    {
12        sender = msg.sender;
13        recipient = _recipient;
14        expiration = now + duration;
15    }
16
17    /// the recipient can close the channel at any time by presenting a
18    /// signed amount from the sender. the recipient will be sent that amount,
19    /// and the remainder will go back to the sender
20    function close(uint256 amount, bytes memory signature) public {
21        require(msg.sender == recipient);
22        require(isValidSignature(amount, signature));
23        recipient.transfer(amount);
24        selfdestruct(sender);
25    }
26
27    /// if the timeout is reached without the recipient closing the channel,
28    /// then the Ether is released back to the sender.
29    function claimTimeout() public {
30        require(now >= expiration);
31        selfdestruct(sender);
32    }
33 }
34
35 }
```

← sender creates contract with funds, specifies duration

← verify sender's signature on amount

← send all funds to sender

Bidirectional Payment Channel

Alice and Bob want to move funds back and forth

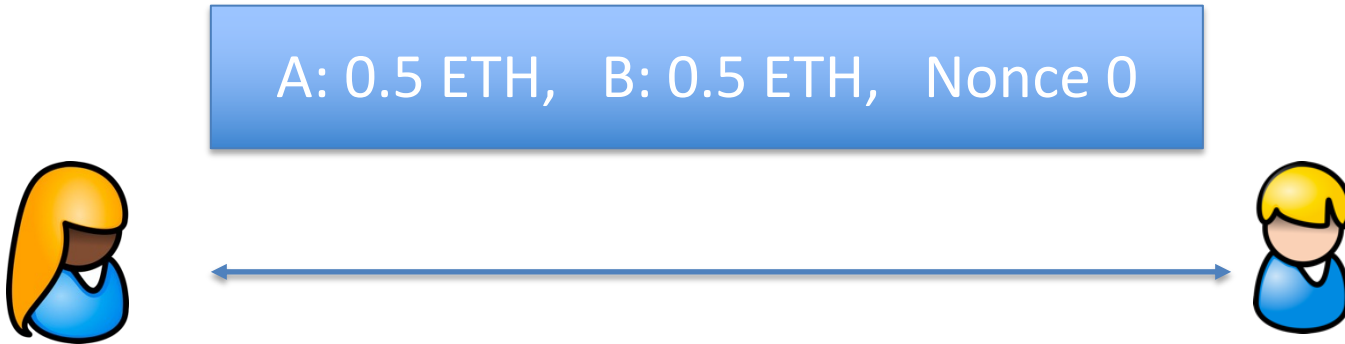


Two Unidirectional Channels?

Not as useful because Channels get exhausted

Bidirectional Payment Channel

On Ethereum: create a shared contract, each contributes 0.5 ETH:



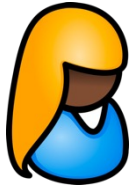
Off chain: Bob sends 0.1 ETH to Alice by both signing new state:

A: 0.6, Bob: 0.4, Nonce 1
Alice sig, Bob sig

Bidirectional Payment Channel

On chain contract does not change:

balance: 1 ETH, Nonce 0



Off chain: Alice and Bob can move funds back and forth by sending updated state sigs to each other:

A: 0.3, Bob: 0.7, Nonce 7

Alice sig, Bob sig

(7th transfer)

Eventually: Alice wants to close payment channel

Alice does: sends latest balances and signatures to contract
⇒ starts challenge period (say, 3 days)

on chain:

A: 0.3 ETH, B: 0.7 ETH, Nonce 7

(pending close)



if Bob does nothing for 3 days:

⇒ funds disbursed according to Alice's submitted state

if Bob submits signed state with greater nonce (e.g., nonce=9)

⇒ funds disbursed according to Bob's submitted state

Watchtowers



Bidirectional channel requires
Bob to constantly check that Alice
did not try to close the channel
with an old stale state
⇒ post latest state if she did

Watchtowers outsource this task

Bob sends latest state to
watchtower.



Trusted for availability

Main points: summary

Payment channel between Alice and Bob:

- One on-chain Tx to create channel (deposit funds);
- Alice & Bob can send funds to each other **off-chain**
... as many Tx as they want;
- One on-chain Tx to close channel and disburse funds

⇒ only two on-chain Tx

A more general concept: State Channels

Smart contract that implements a game between Alice and Bob.

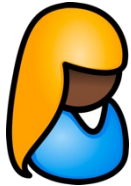
Begin game & end game: on chain. **All moves are done off-chain.**



State Channels

Can be used to implement any 2-party contract off chain!

two Tx on-chain: contract creation and termination



Bidirectional channels on Bitcoin

The Lightning Network

Bidirectional payment channels on Bitcoin

Problem: no updatable state in UTXOs \Rightarrow
much harder to implement a bidirectional channel

Solution:

- When updating the channel to Alice's benefit,
Alice gets TX that invalidates Bob's old state

UTXO payment channel concepts

Will create UTXO that can be spent in one of two ways: (using if opcode)

- **Relative time-lock:** UTXO contains a number t .
A properly signed Tx can spend this UTXO
 t blocks (or more) after it was created (CLTV opcode)
- **Hash lock:** UTXO contains a hash image X .
A properly signed Tx can spend this UTXO
by presenting x s.t. $X = \text{SHA256}(x)$.

(x is called a hash preimage of X)

Example script

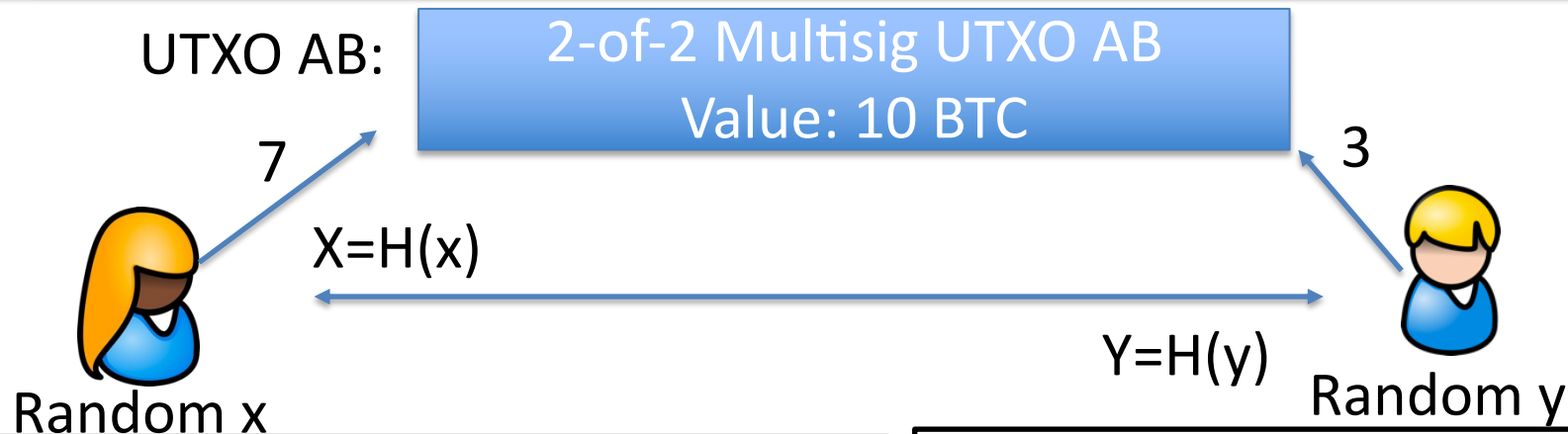
Example locktime redeem script: two ways to redeem UTXO

```
OP_IF
  OP_HASH256 <digest> OP_EQUALVERIFY      // redeem by providing <digest> preimage
OP_ELSE
  <num> OP_CLTV OP_DROP                    // redeem by waiting <num> blocks
OP_ENDIF

<2> <sender pubkey> <recipient pubkey> <2> OP_CHECKMULTISIG    // check 2-of-2 signature
```

This is called a **hash-timelock contract** (HTLC).

UTXO Payment Channel



TX1: input: UTXO AB

Out1: pay 7 → A

Out2: either 3 → B, 7 day timelock
or 3 → A now, given y s.t. $H(y)=Y$

Alice sig

TX2: input UTXO AB

pay 3 → B

either 7 → A, 7 day timelock
or 7 → B now, given x s.t. $H(x)=X$

Bob sig

Alice can sign and post Tx2, wait 7 days, and get her funds back

UTXO Payment Channel Update

2-of-2 Multisig UTXO AB
Value: 10 BTC



x

$$X' = H(x')$$



Alice sends 1BTC to Bob (off chain)

Random x'

TX3 input: UTXO AB

Out1: pay 6 \rightarrow A

Out2: either 4 \rightarrow B, 7 day timelock
or 4 \rightarrow A now, given y s.t. $H(y)=Y$

Alice sig

TX4 input: UTXO AB

pay 4 \rightarrow B

either 6 \rightarrow A, 7 day timelock
or 6 \rightarrow B now, given x' s.t. $H(x')=X'$

Bob sig

Security: ways to close the channel?

Alice has TX2, TX4, x , x'

TX2: (stale state)

pay 3 \rightarrow B

either 7 \rightarrow A, 7 day timelock

or 7 \rightarrow B now, given x s.t. $H(x)=X$

Bob sig

TX4: (current state)

pay 4 \rightarrow B

either 6 \rightarrow A, 7 day timelock

or 6 \rightarrow B now, given x' s.t. $H(x')=X'$

Bob sig

Bob has TX1, TX3, y , x

TX1: (stale state)

pay 7 \rightarrow A

either 3 \rightarrow B, 7 day timelock

or 3 \rightarrow A now, given y s.t. $H(y)=Y$

Alice sig

TX3: (current state)

pay 6 \rightarrow A

either 4 \rightarrow B, 7 day timelock

or 4 \rightarrow A now, given y s.t. $H(y)=Y$

Alice sig

Security: ways to close the channel?

Alice has TX2, TX4, x , x'

Bob has TX1, TX3, y , x

TX2:
pay 3 →
either 7
or 7 → E
Bob sig

The good case:
Alice can post Tx4 or Bob can post Tx3 to chain and
close channel after 7 days
A gets 6, B gets 4

lock
 $H(y)=Y$

TX4: (current state)
pay 4 → B
either 6 → A, 7 day timelock
or 6 → B now, given x' s.t. $H(x')=X'$
Bob sig

TX3: (current state)
pay 6 → A
either 4 → B, 7 day timelock
or 4 → A now, given y s.t. $H(y)=Y$
Alice sig

Security: ways to close the channel?

Alice has TX2, TX4, x , x'

TX2: (stale state)

pay 3 \rightarrow B

either 7 \rightarrow A, 7 day timelock

or 7 \rightarrow B now, given x s.t. $H(x)=X$

Bob sig

Bob has TX1, TX3, y , x

TX1: (stale state)

pay 7 \rightarrow A

either 3 \rightarrow B, 7 day timelock

or 3 \rightarrow A now, given y s.t. $H(y)=Y$

Alice sig

TX

pa

eit

or

Bob sig

The bad case (Alice cheats):

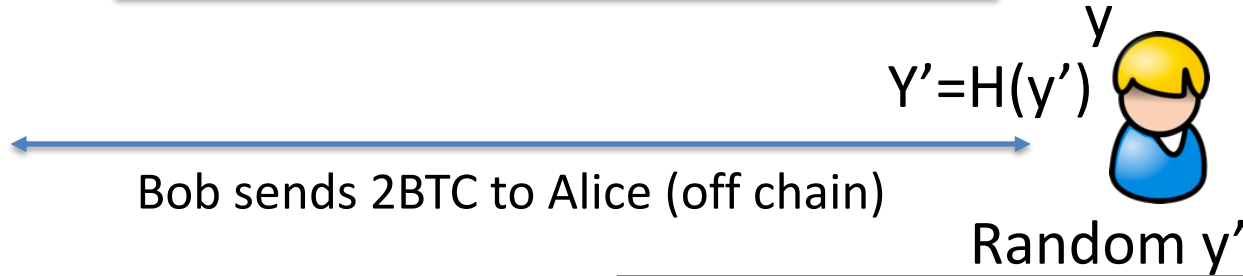
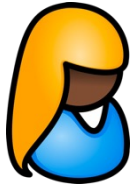
if Alice posts the stale Tx2 then Bob will use x to take all 10 BTC

\Rightarrow sending x to Bob revokes the stale Tx2 held by Alice

Alice sig

UTXO Payment Channel Update

2-of-2 Multisig Address C:
Value: 10 BTC



Random y'

TX5 input: UTXO AB

pay 8 → A

either 2 → B, 7 day timelock

or 2 → A now, given y s.t. $H(y') = Y'$

Alice sig

TX6 input: UTXO AB

pay 2 → B

either 8 → A, 7 day timelock

or 8 → B now, given x s.t. $H(x') = X'$

Bob sig

Security: ways to close the channel?

Alice has TX2, TX6, x , x' , y

TX2:

pay 3 \rightarrow B

either 7 \rightarrow A, 7 day timelock
or 7 \rightarrow B now, given x s.t. $H(x)=X$
Bob sig

TX6:

pay 2 \rightarrow B

either 8 \rightarrow A, 7 day timelock
or 8 \rightarrow B now, given x' s.t. $H(x')=X'$
Bob sig

Bob has TX3, TX5, y , y' , x

TX3:

pay 6 \rightarrow A

either 4 \rightarrow B, 7 day timelock
or 4 \rightarrow A now, given y s.t. $H(y)=Y$
Alice sig

TX5:

pay 8 \rightarrow A

either 2 \rightarrow B, 7 day timelock
or 2 \rightarrow A now, given y' s.t. $H(y')=Y'$
Alice sig

Security: ways to close the channel?

Alice has TX2, TX6, x , x' , y

Bob has TX3, TX5, y , y' , x

TX2:
pay 3 \rightarrow B
either 7 \rightarrow A, 7 day timelock
or 7 \rightarrow B now, given x s.t. $H(x)=X$
Bob sig

TX3:
pay 6 \rightarrow A
either 4 \rightarrow B, 7 day timelock
or 4 \rightarrow A now, given y s.t. $H(y)=Y$
Alice sig

TX6:
pay 2 \rightarrow B
either 8 \rightarrow A, 7 day timelock
or 8 \rightarrow B now, given x' s.t. $H(x')=X'$
Bob sig

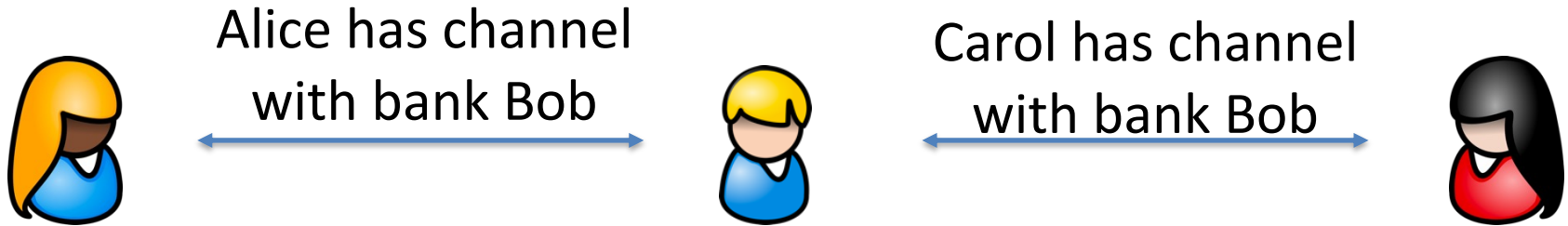
TX5:
pay 8 \rightarrow A
either 2 \rightarrow B, 7 day timelock
or 2 \rightarrow A now, given y' s.t. $H(y')=Y'$
Alice sig

The bad case (Bob cheats):

Bob posts the stale Tx3 \Rightarrow Alice will use y to take all 10 BTC $=Y$

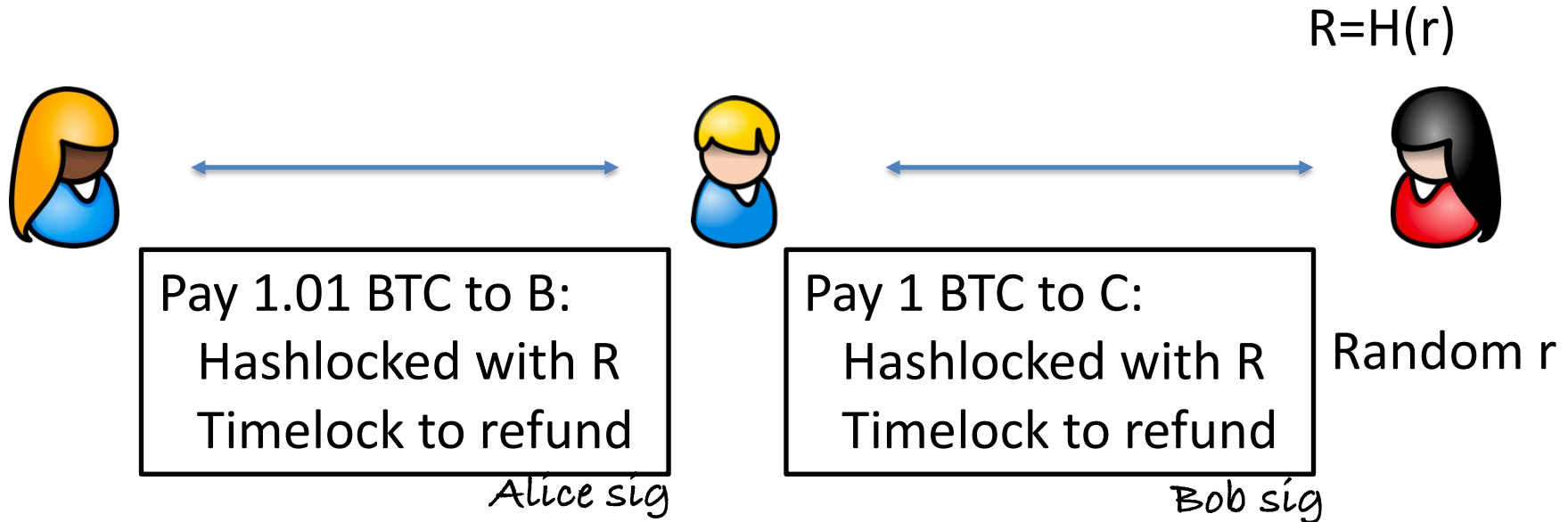
Multihop payments

Multi-hop payments



Alice wants to pay Carol through *untrusted* intermediary Bob

Multi-hop payments (briefly)



B can claim 1.01 BTC with r

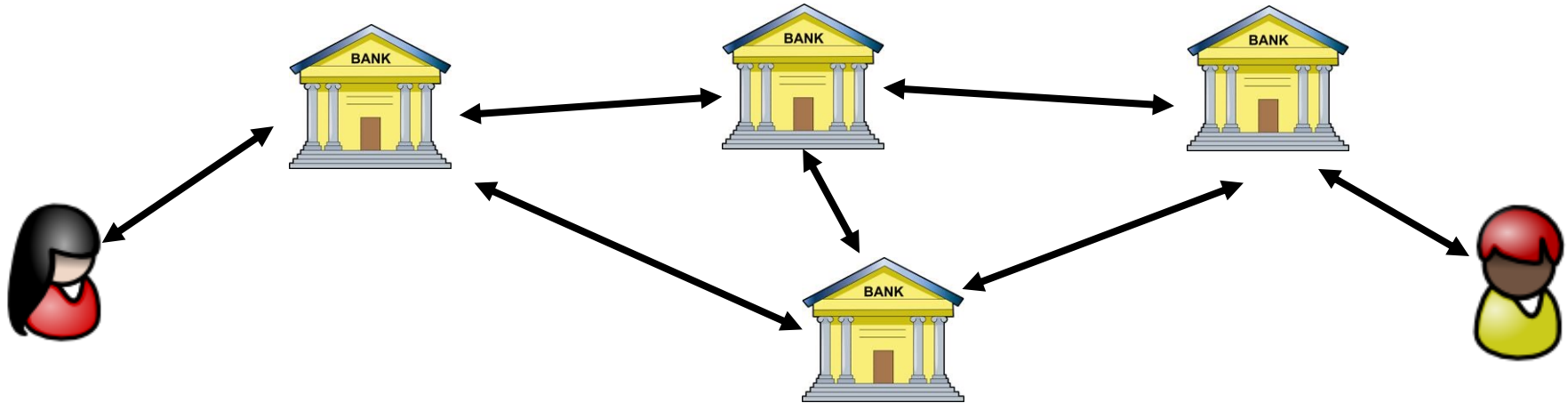
C can claim 1 BTC with r

if Carol never claims, Bob gets funds back after timelock

The lightning network

The network: lots of open bi-directional payment channels.

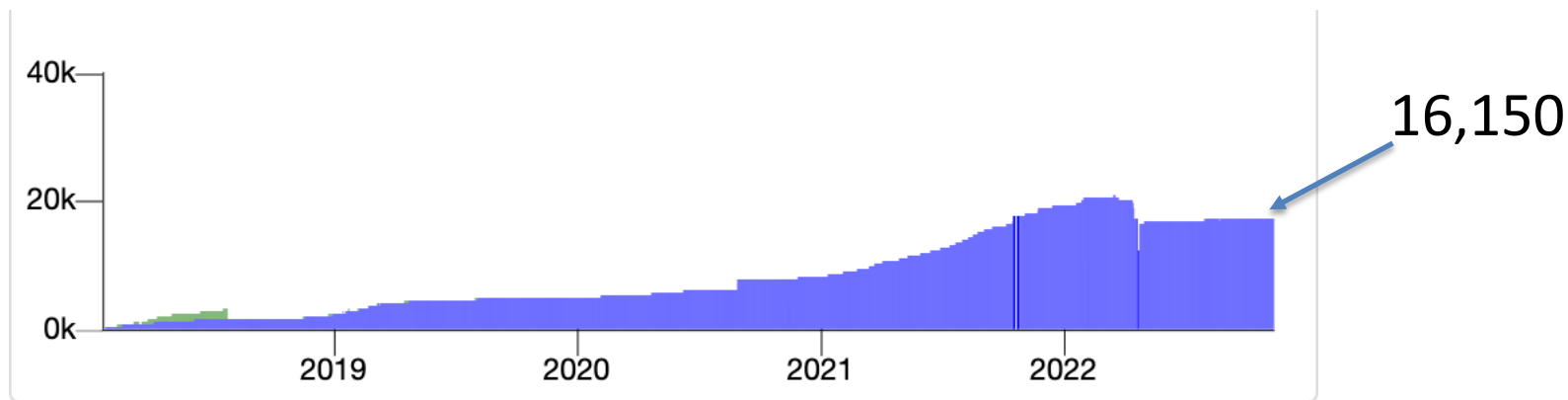
Alice want to pay Bob: she finds a route to Bob through the graph



Many extensions possible: multi currency hubs, credit hubs, ...

Stats

nodes in lightning network (Nov. 2022)



Number of channels: 77K

END OF LECTURE

Next lecture: scaling via Rollups