Statechains:

Off-chain Transfer of UTXO Ownership

Ruben Somsen
Email: rsomsen@gmail.com
Twitter: @SomsenRuben

What Statechains Achieve

What Statechains Achieve

- L2 scaling by avoiding on-chain transactions

What Statechains Achieve

- L2 scaling by avoiding on-chain transactions
- Advantage over Lightning:

    unrestricted coin movement (has synergy)

What Statechains Achieve

- L2 scaling by avoiding on-chain transactions
- Advantage over Lightning:

  unrestricted coin movement (has synergy)
- Advantage over Federated Sidechains:

  federation doesn’t have full control

What Statechains Achieve

- L2 scaling by avoiding on-chain transactions
- Advantage over Lightning:

  unrestricted coin movement (has synergy)
- Advantage over Federated Sidechains:

  federation doesn’t have full control
- Unique limitation: can only move full UTXO amount

What Statechains Build On

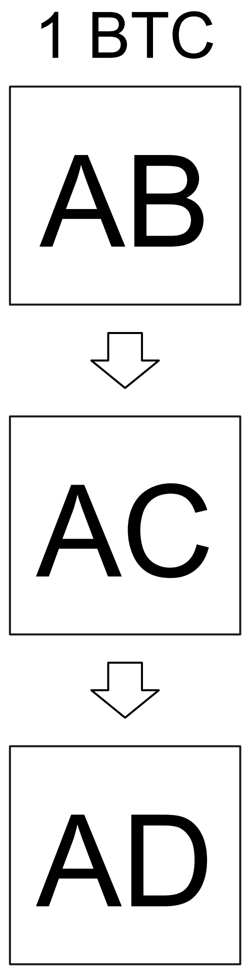
- Schnorr signatures
- Adaptor signatures
- Eltoo
- Graftroot

What Statechains Build On

- Schnorr signatures
- Adaptor signatures
- Eltoo
- Graftroot

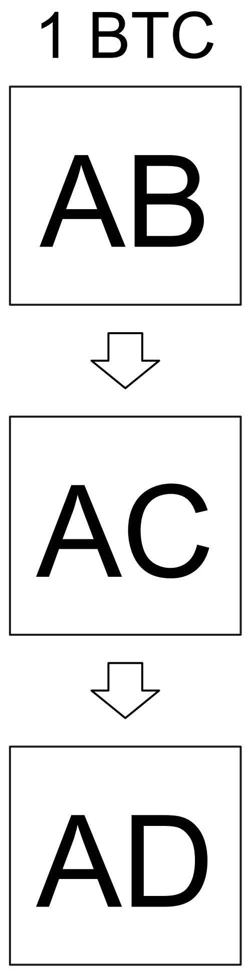
Works on any cryptocurrency that supports the above

High-level Overview



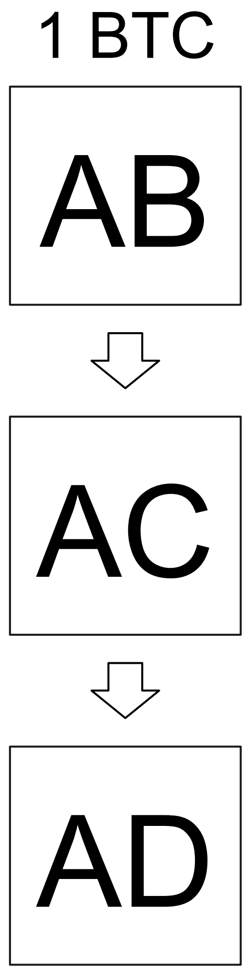
High-level Overview

- Change UTXO ownership off-chain



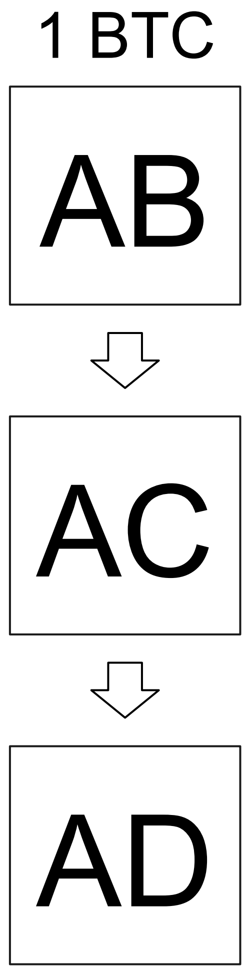
High-level Overview

- Change UTXO ownership off-chain
- Guaranteed on-chain redemption (D)



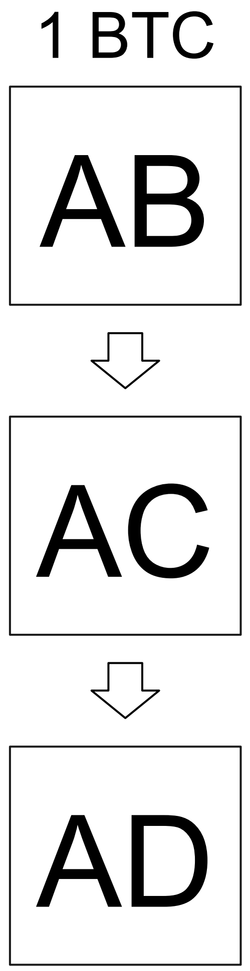
High-level Overview

- Change UTXO ownership off-chain
- Guaranteed on-chain redemption (D)
- Facilitated by “statechain entity” (A)



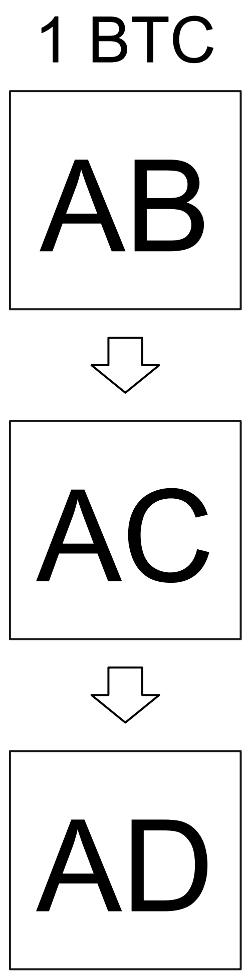
High-level Overview

- Change UTXO ownership off-chain
- Guaranteed on-chain redemption (D)
- Facilitated by “statechain entity” (A)
- A can collude with prior owners (B, C)

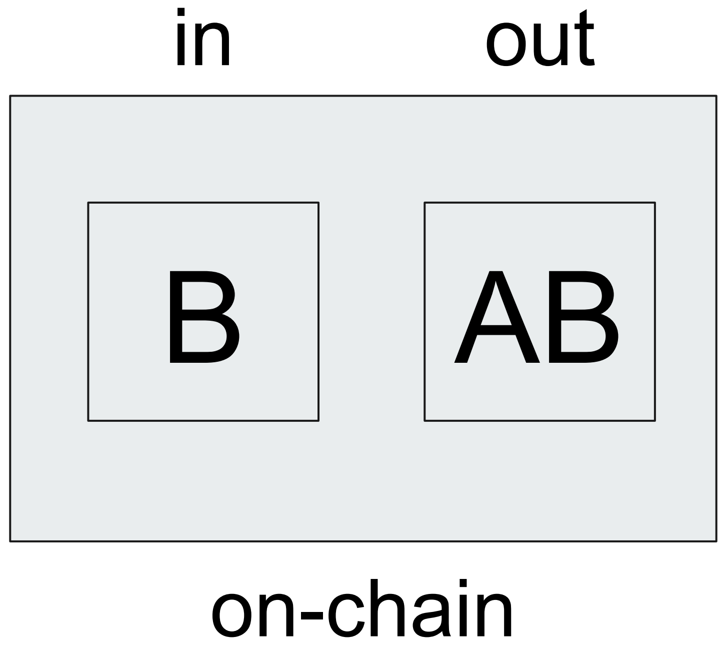


High-level Overview

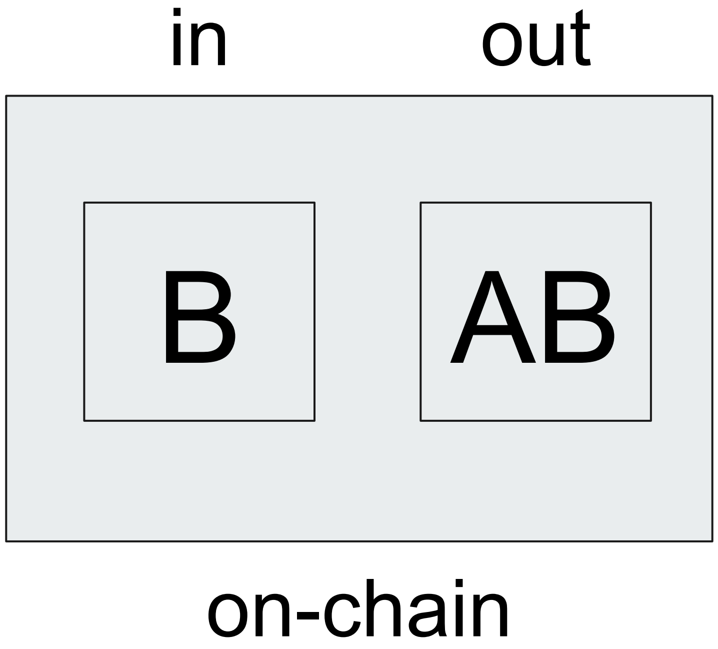
- Change UTXO ownership off-chain
- Guaranteed on-chain redemption (D)
- Facilitated by "statechain entity" (A)
- A can collude with prior owners (B, C)
- Collusion/cheating always provable



Bob locks up 1 BTC with Alice...

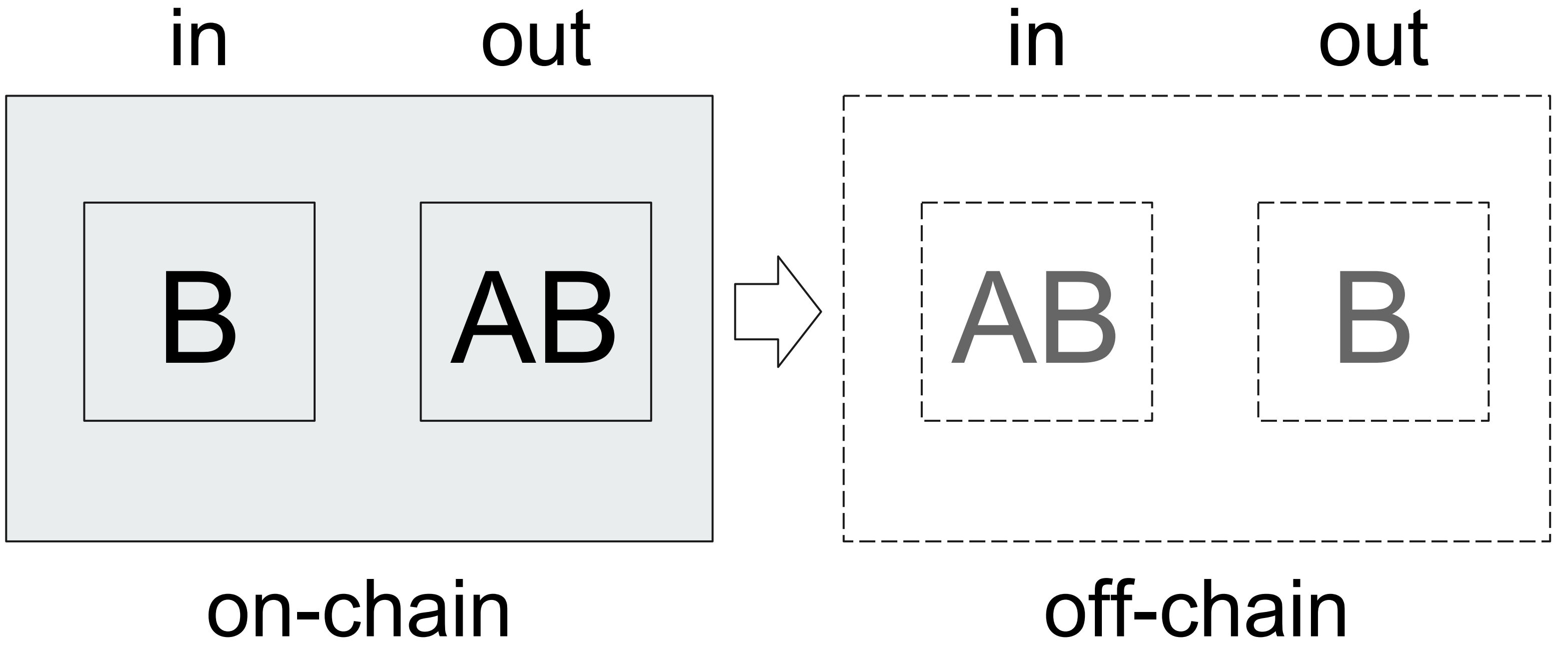


Bob locks up 1 BTC with Alice...

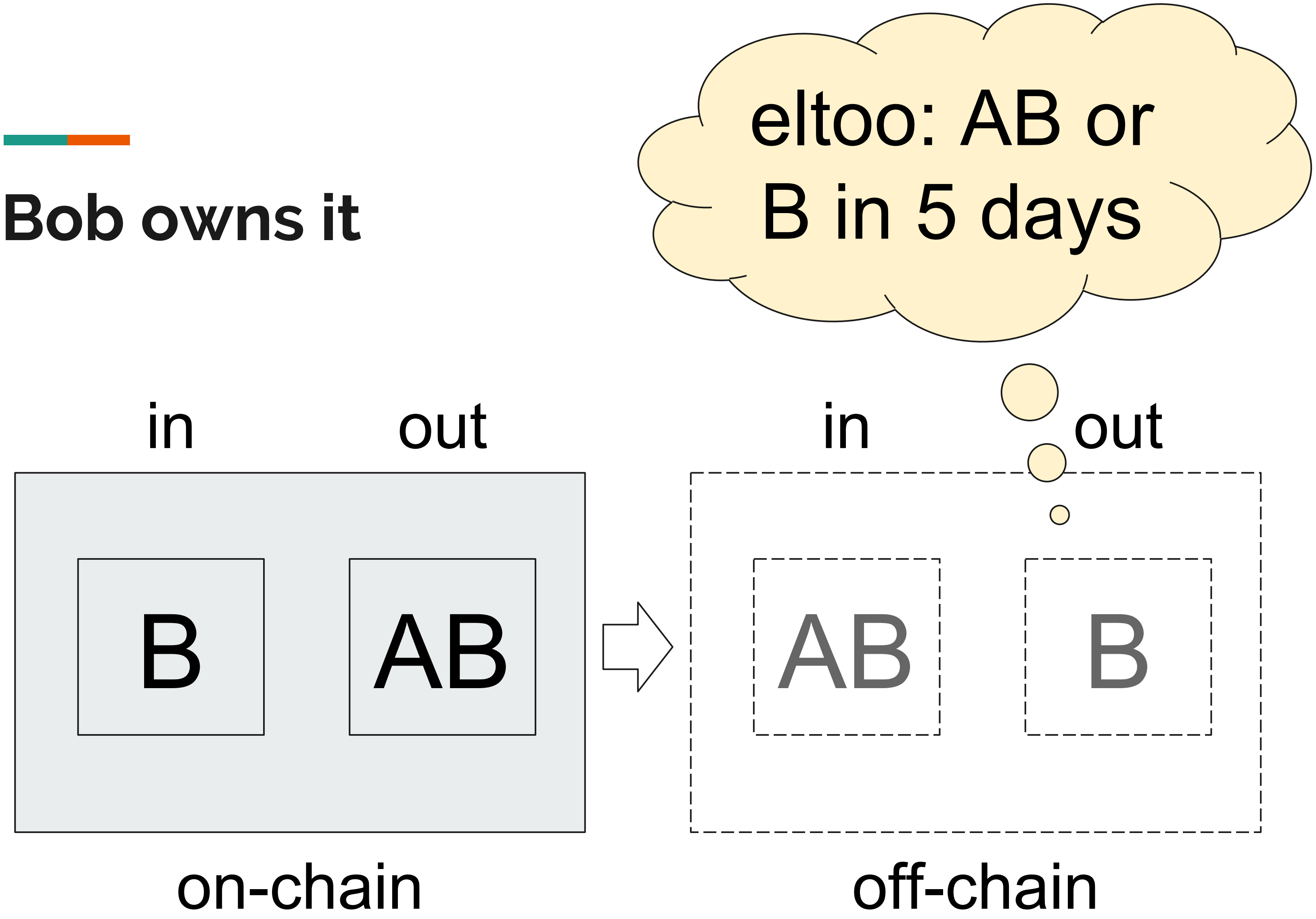


...but who owns it?

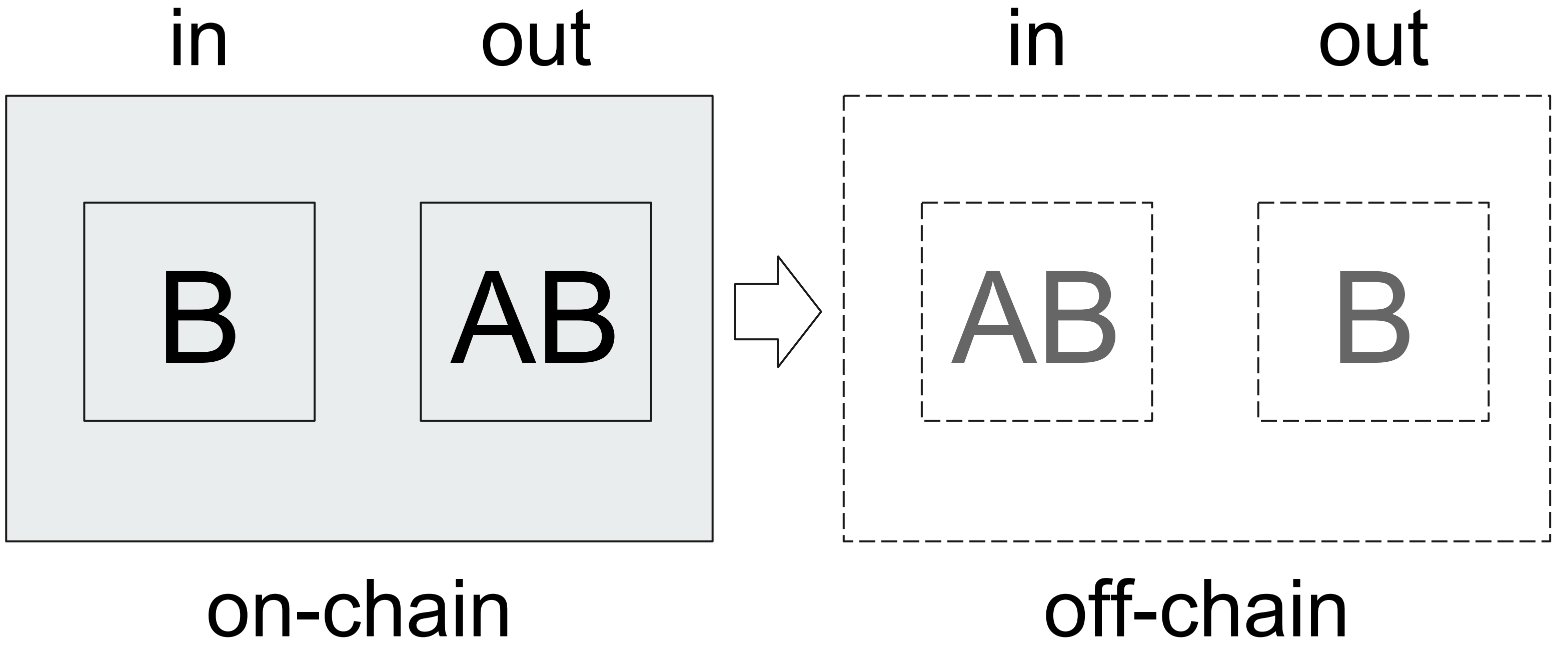
Bob owns it



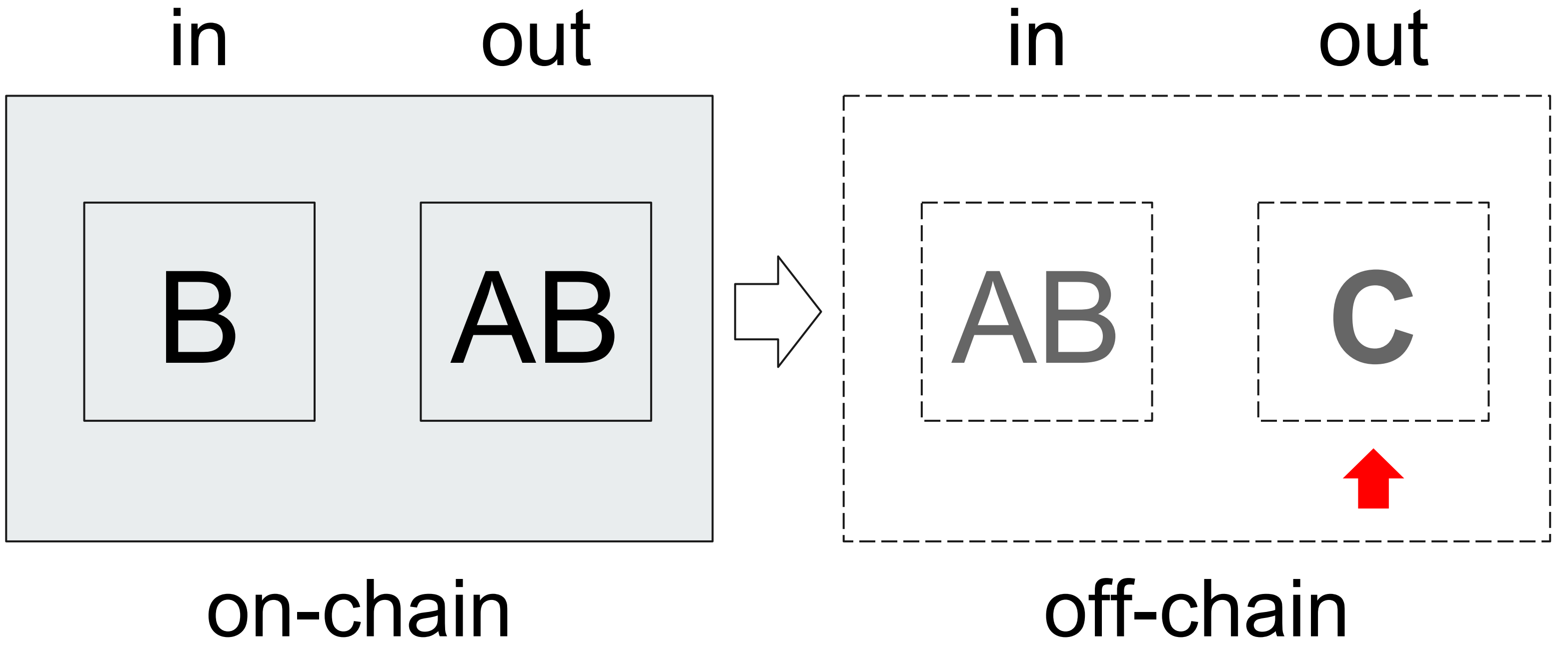
Bob owns it



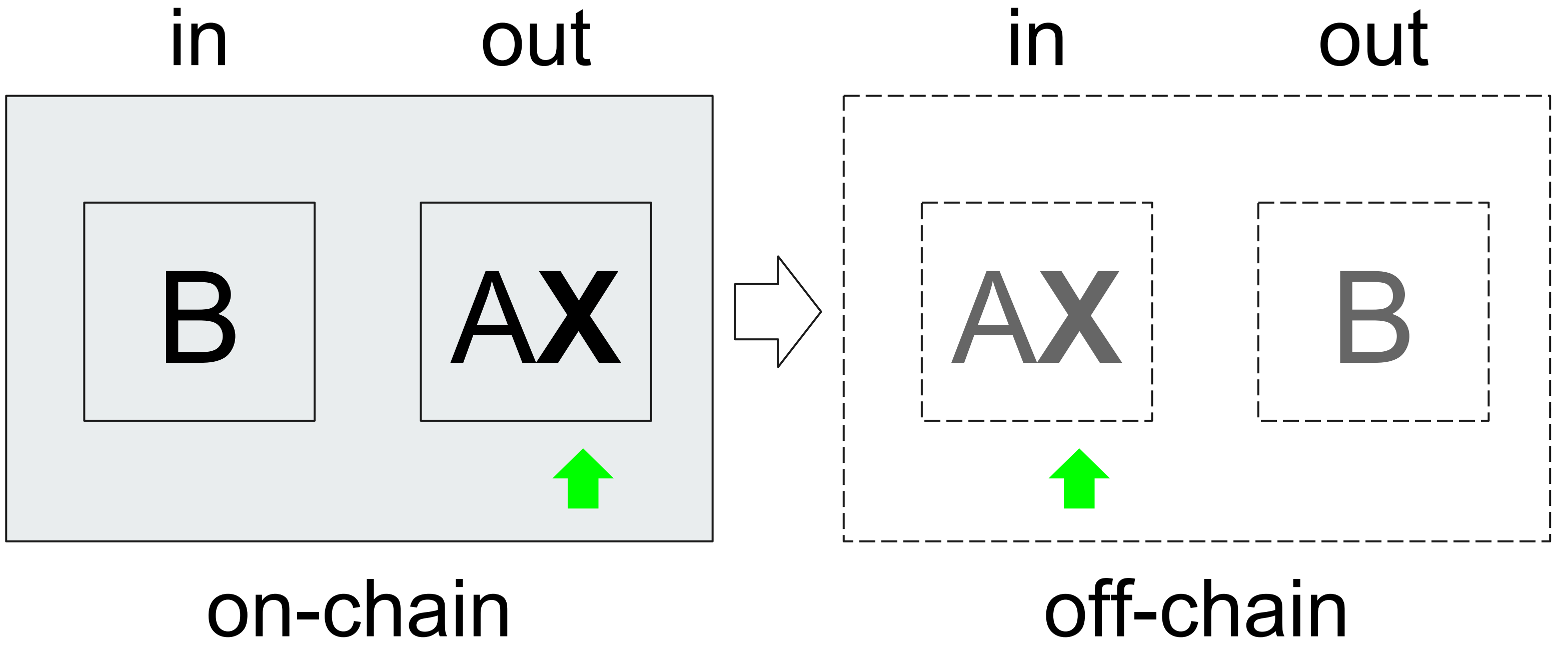
Can Bob transfer this off-chain to Carol?



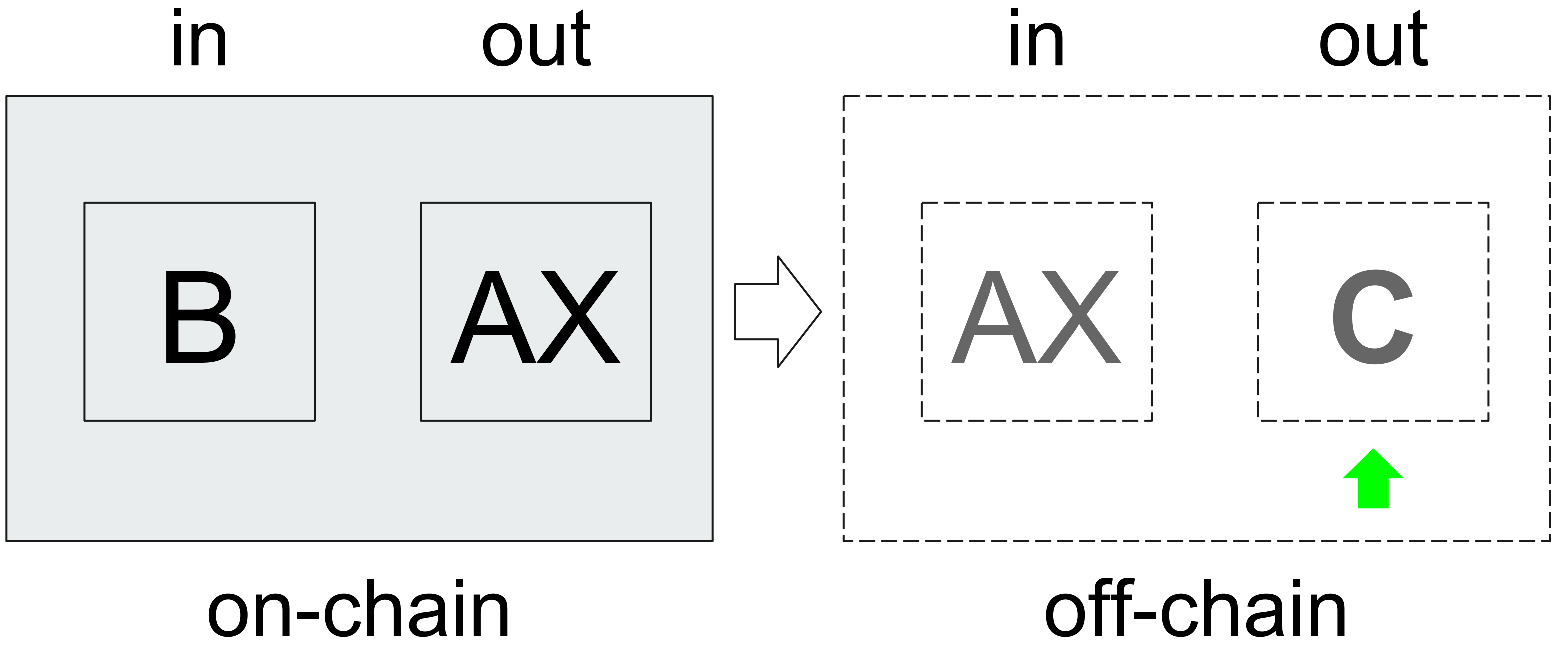
Can Bob transfer this off-chain to Carol?
Sort of, but Carol has no control over it



So Bob uses a transitory key X instead



So Bob uses a transitory key X instead **and passes the key on to Carol**



The role of the statechain entity (A)

The role of the statechain entity (A)

- Promises to *only* cooperate with the last owner

The role of the statechain entity (A)

- Promises to *only* cooperate with the last owner
- Can be a federation

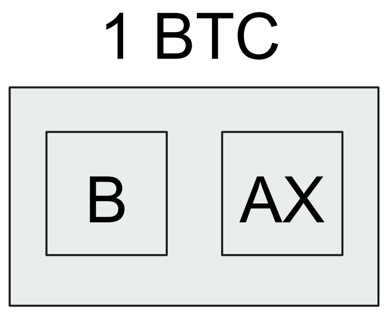
Schnorr:
A1+A2+A3=A

The role of the statechain entity (A)

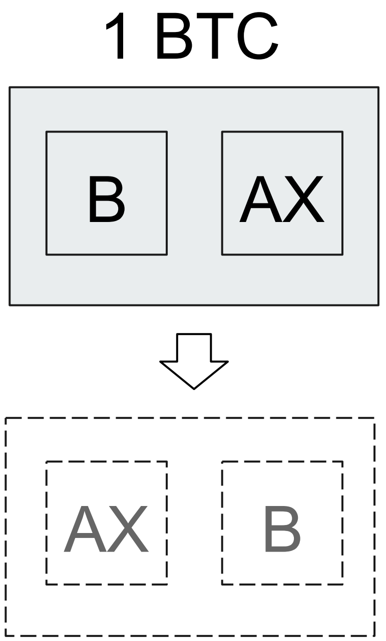
- Promises to *only* cooperate with the last owner
- Can be a federation
- Updates the statechain:
  - listing all UTXOs it controls (no duplicates)
  - every transfer has a signature (e.g. B to C)

Schnorr:
A1+A2+A3=A

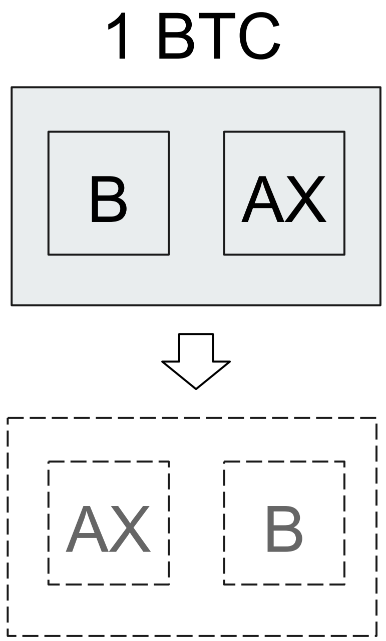
Bitcoin



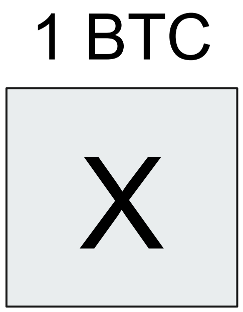
Bitcoin



Bitcoin

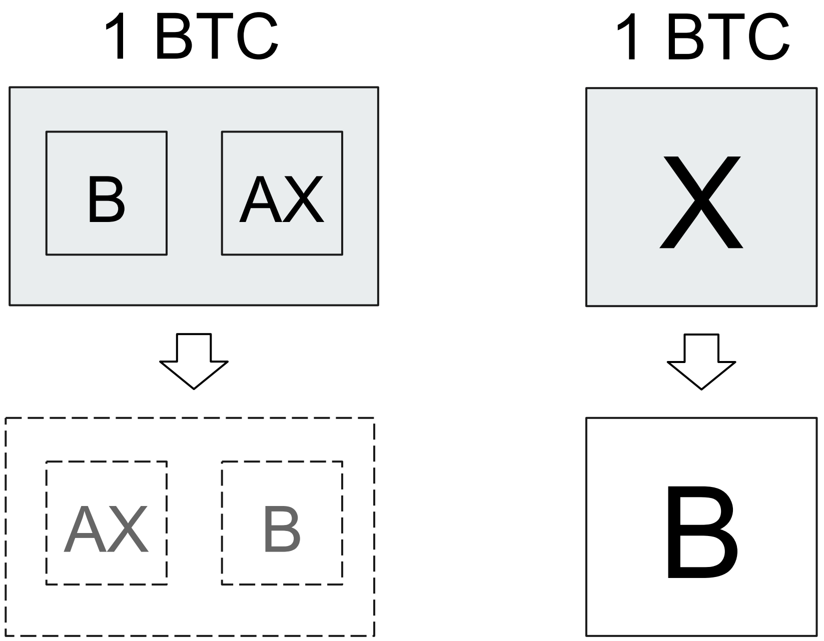


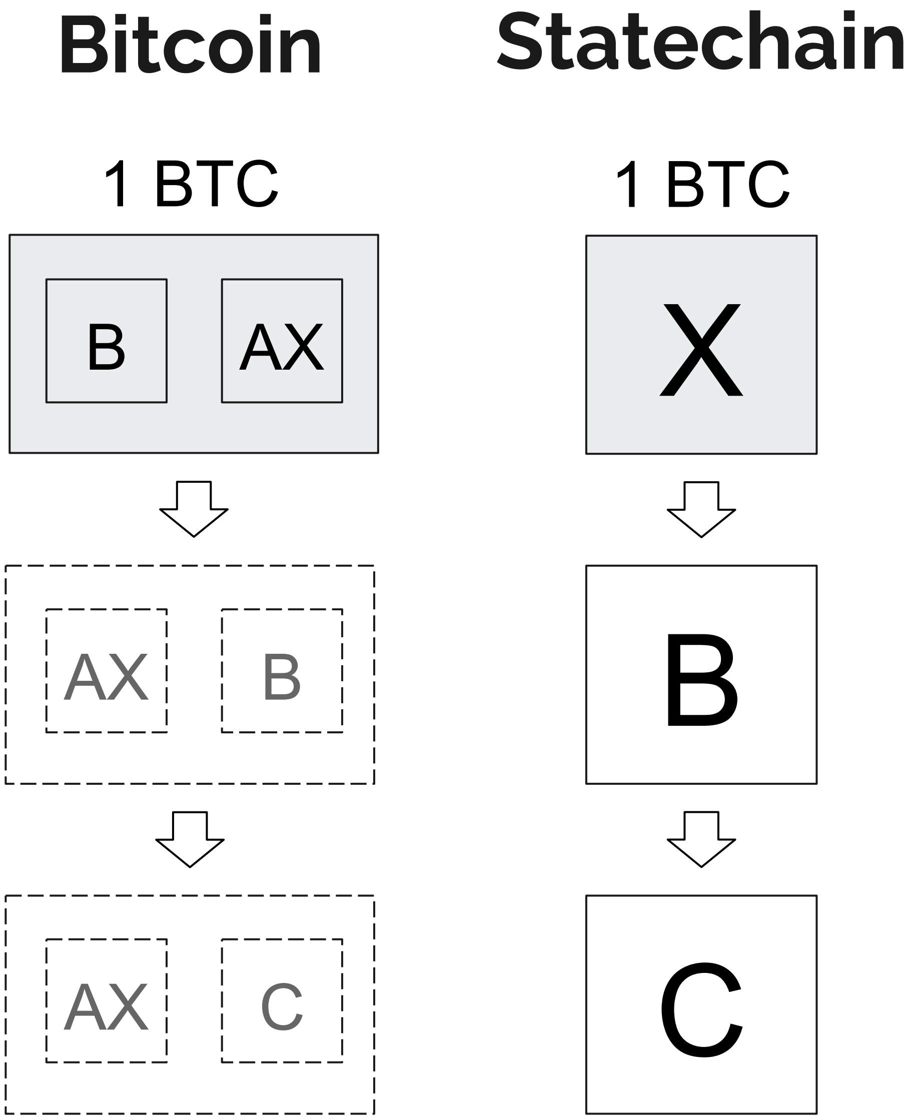
Statechain

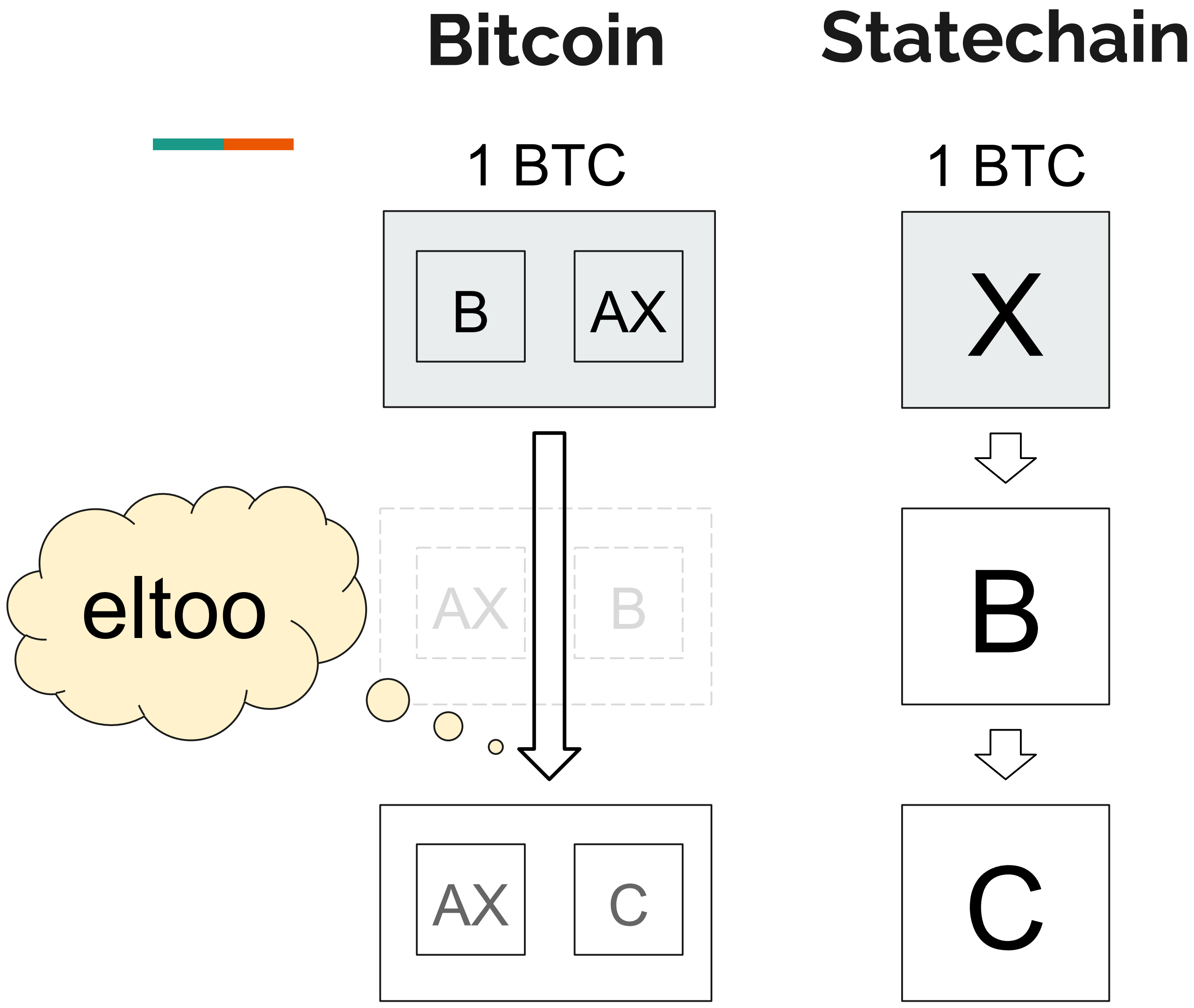


Bitcoin

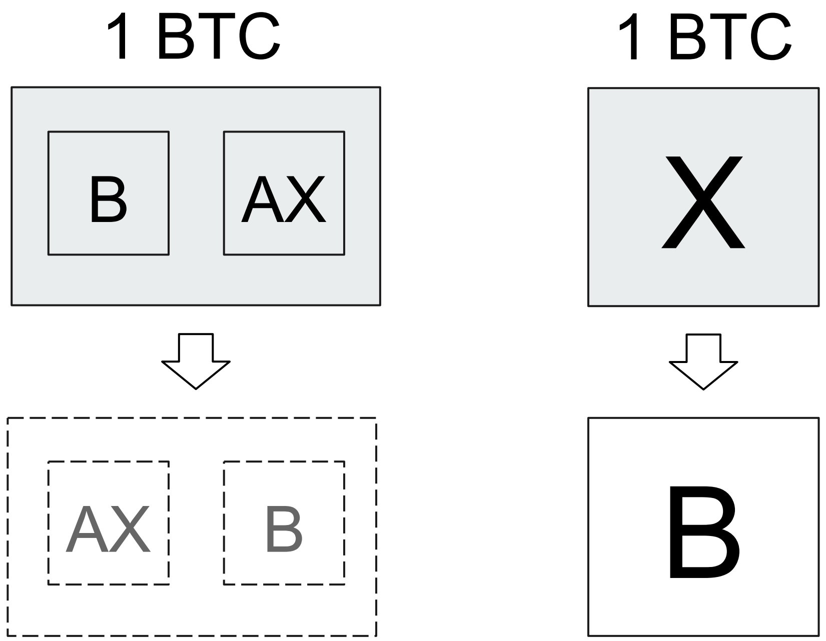
Statechain



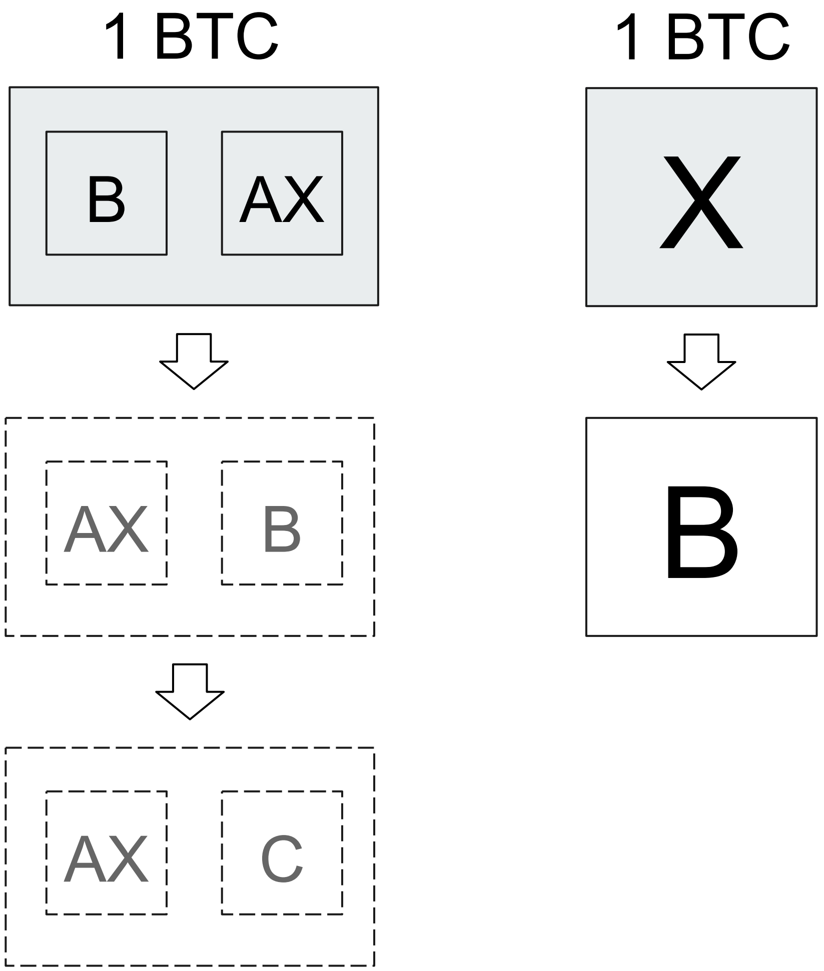

Bitcoin
Statechain
1 BTC
B
AX
AX
B
AX
C
1 BTC
X
B
C


Bitcoin
Statechain
1 BTC
1 BTC
B
AX
X
eltoo
AX
B
B
AX
C
C

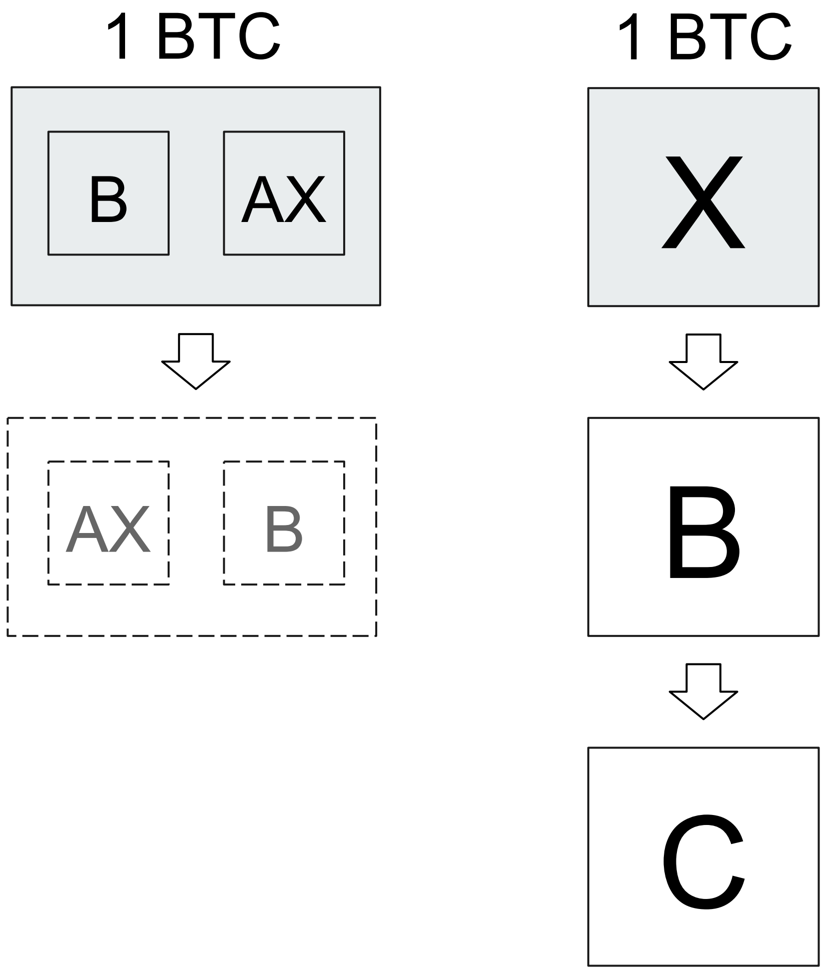
Problem: who goes first?



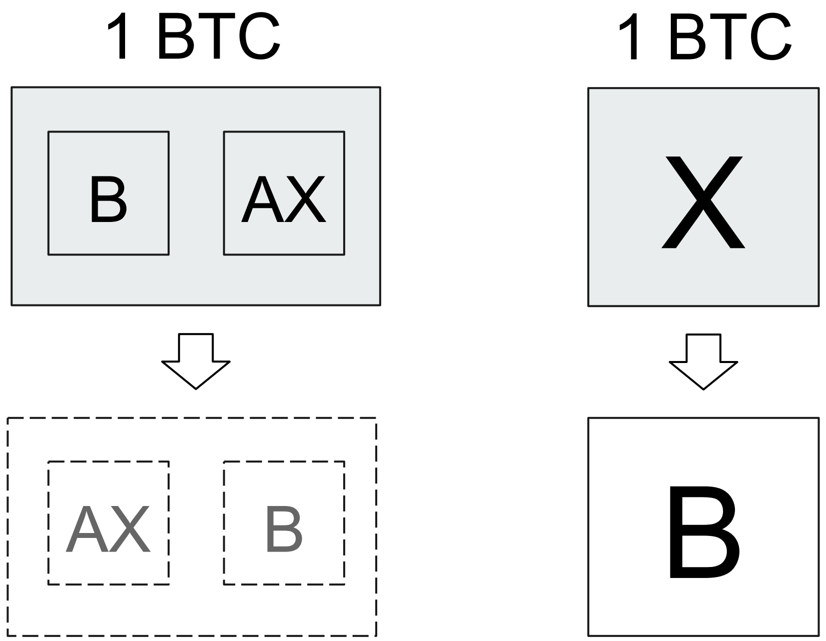
Statechain entity A goes first...?



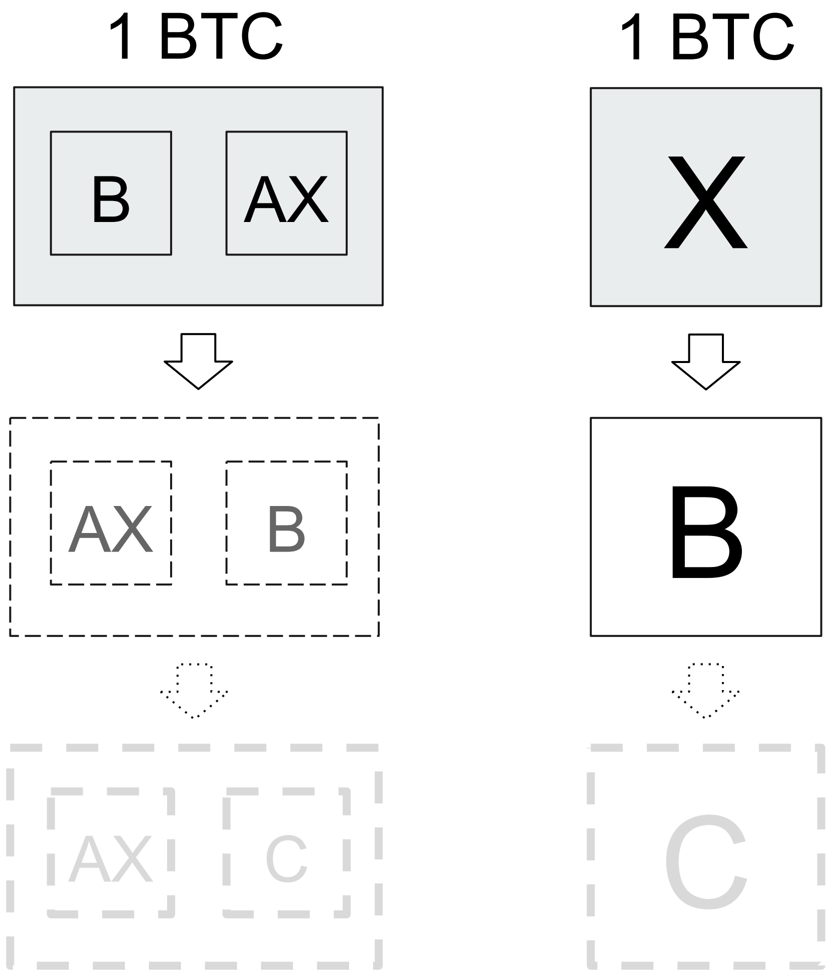
User B goes first...?



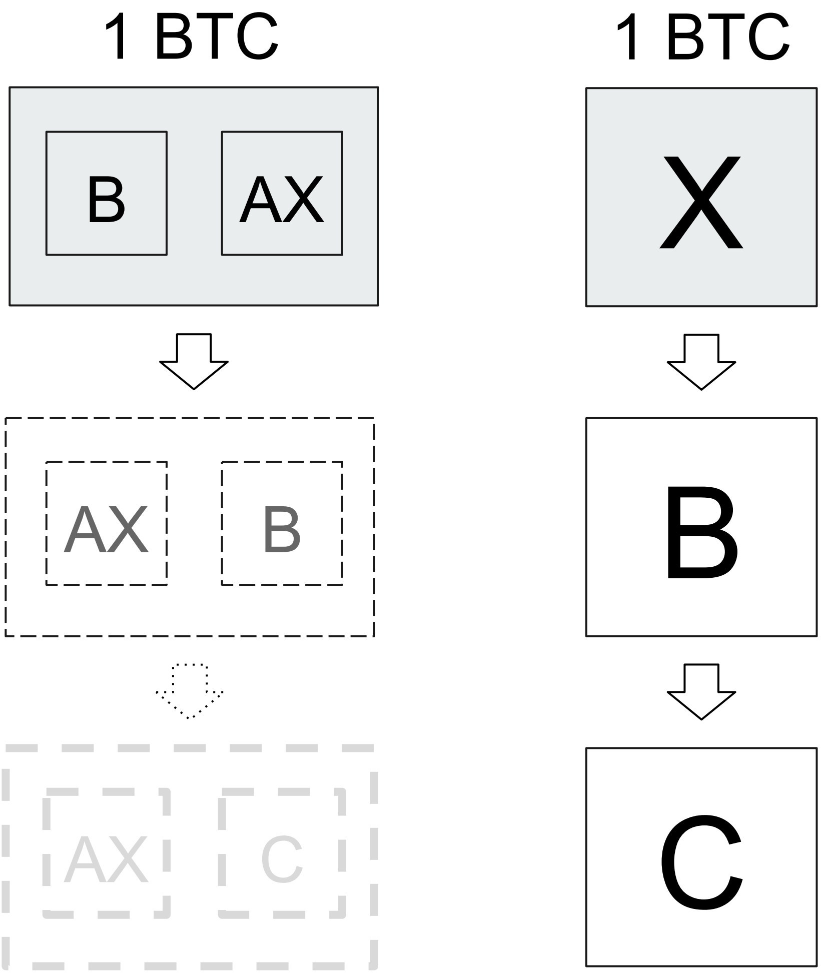
Solution: Adaptor Signatures



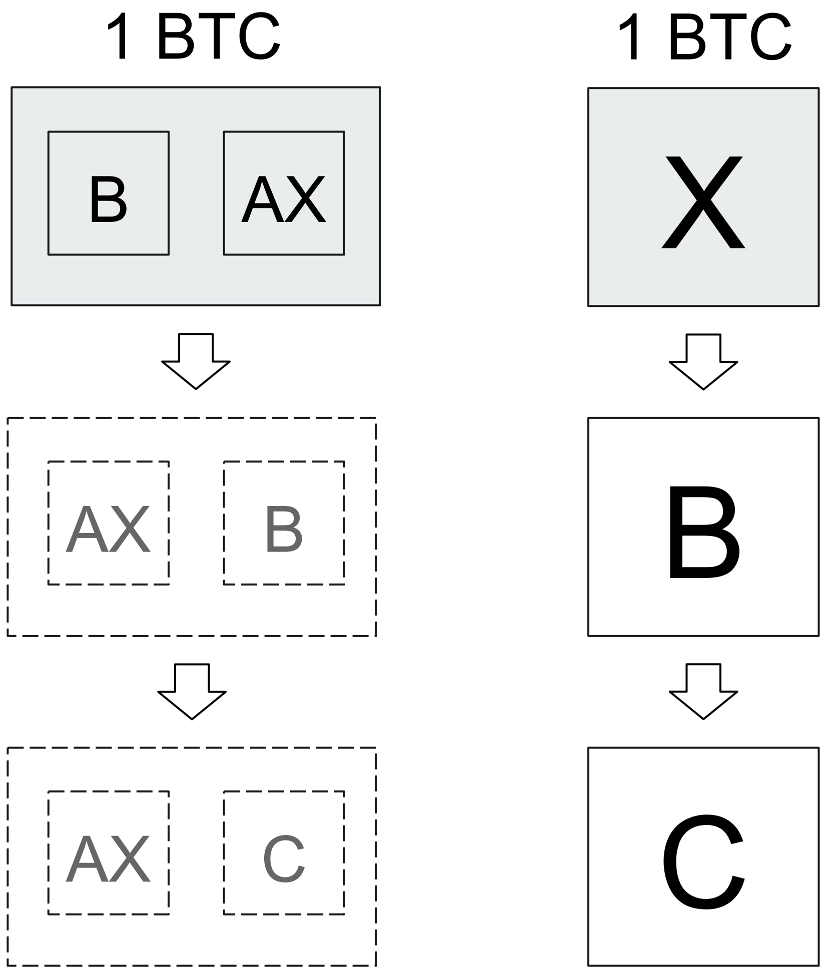
Everyone shares an incomplete signature



Completing the statechain signature...



…automatically completes the bitcoin signature



Security Model

Bitcoin:	*A + X*
Statechain:	*~~B / C / D~~ / E (knows X)*
Owner:	*Prev.* *Last*

Security Model

Bitcoin:	*A + X*	
Statechain:	*~~B / C / D~~*	*/ E (knows X)*
Owner:	*Prev.*	*Last*

- Moving the coins always requires the permission of:
  a. Statechain entity A (typically a federation)
  b. a transitory key holder (who held the UTXO)

Security Model

Bitcoin:	*A + X*
Statechain:	*~~B / C / D~~ / E (knows X)*
Owner:	*Prev. Last*

- Moving the coins always requires the permission of:
  a. Statechain entity A (typically a federation)
  b. a transitory key holder (who held the UTXO)
- Entity must cooperate with LAST transitory key holder

Security Model

Bitcoin:	*A + X*	
Statechain:	*~~B / C / D~~*	*/ E (knows X)*
Owner:	*Prev.*	*Last*

- Moving the coins always requires the permission of:
   a. Statechain entity A (typically a federation)
   b. a transitory key holder (who held the UTXO)
- Entity must cooperate with LAST transitory key holder
- Failure to do so will produce evidence of fraud

Worst case scenario

Worst case scenario

- Entity obtains a bunch of transitory keys (X, Y, Z…)

Worst case scenario

- Entity obtains a bunch of transitory keys (X, Y, Z...)
- Proceeds to (provably) steal the coins

Worst case scenario

- Entity obtains a bunch of transitory keys (X, Y, Z...)
- Proceeds to (provably) steal the coins
- Uncompromised transitory keys withdraw on-chain

Worst case scenario

- Entity obtains a bunch of transitory keys (X, Y, Z...)
- Proceeds to (provably) steal the coins
- Uncompromised transitory keys withdraw on-chain

Harmless without transitory keys (weak assumption):

- The statechain entity gets hacked

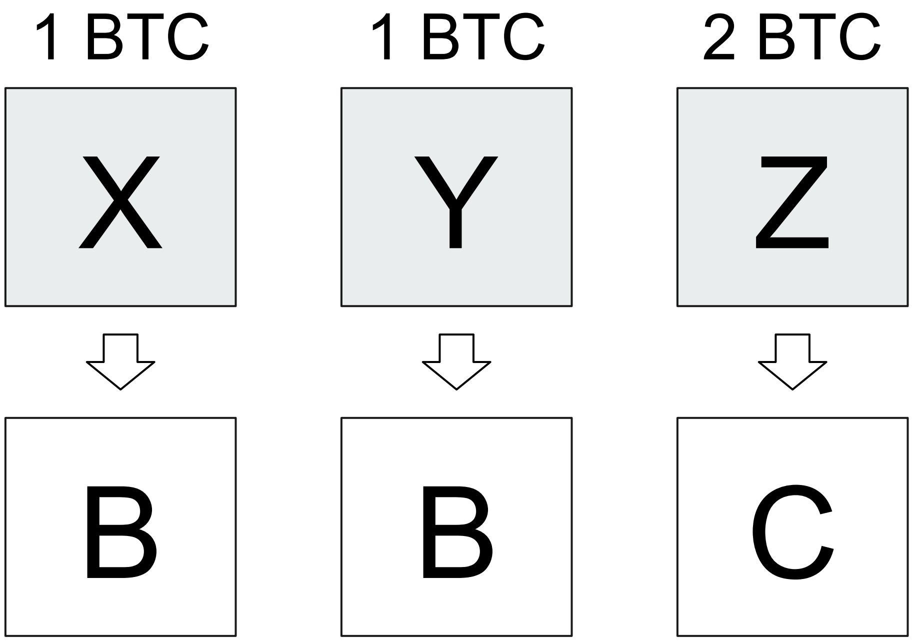
Worst case scenario

- Entity obtains a bunch of transitory keys (X, Y, Z...)
- Proceeds to (provably) steal the coins
- Uncompromised transitory keys withdraw on-chain

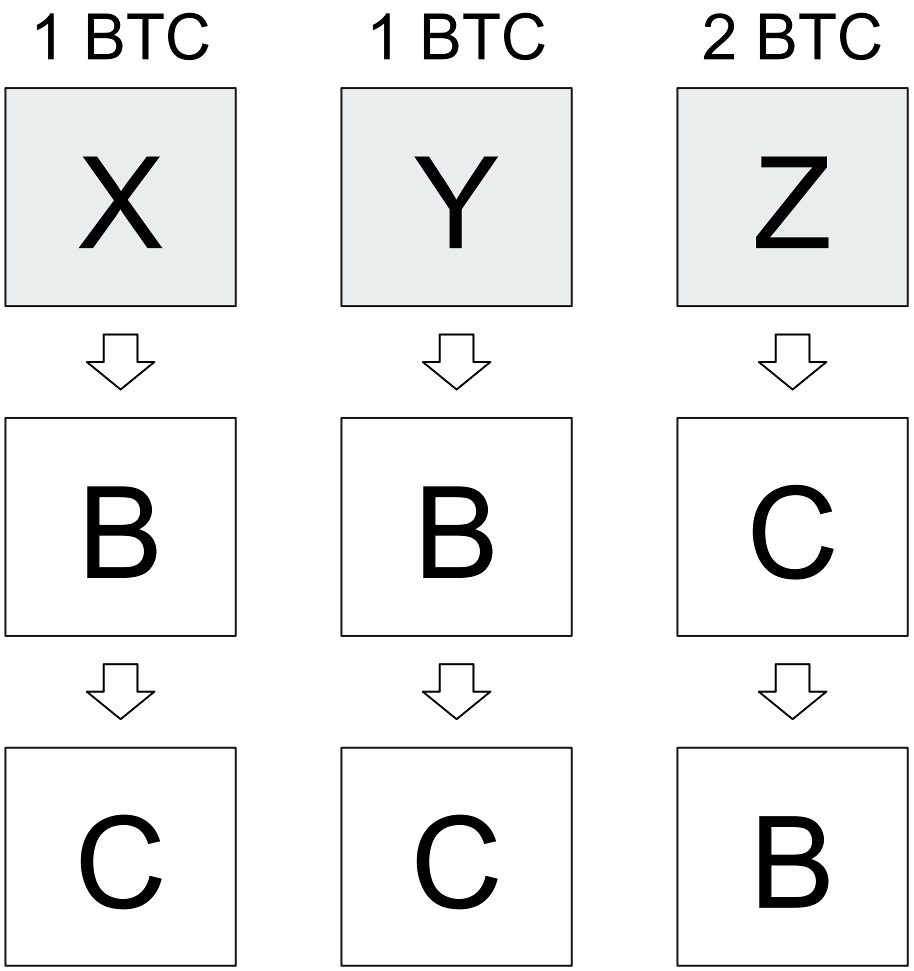
Harmless without transitory keys (weak assumption):

- The statechain entity gets hacked
- Court order to freeze/confiscate coins

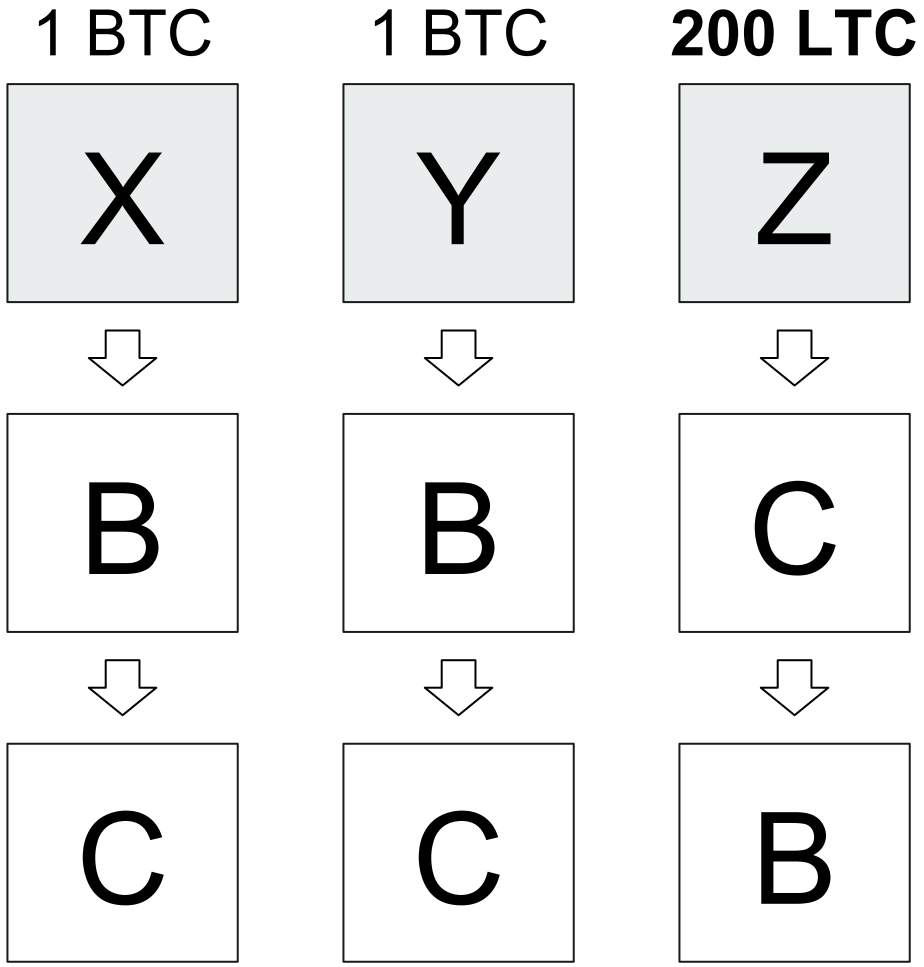
Swapping to smaller amounts



Swapping to smaller amounts



Possible with other coins



Microtransactions

“Anything smaller than an economically viable UTXO”

Microtransactions

“Anything smaller than an economically viable UTXO”

- Required if Statechain entity wants to charge fees

Microtransactions

“Anything smaller than an economically viable UTXO”

- Required if Statechain entity wants to charge fees
- Needed when swapping between multiple currencies

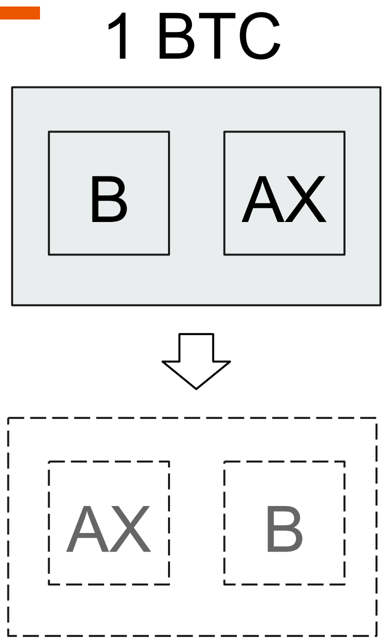
Microtransactions

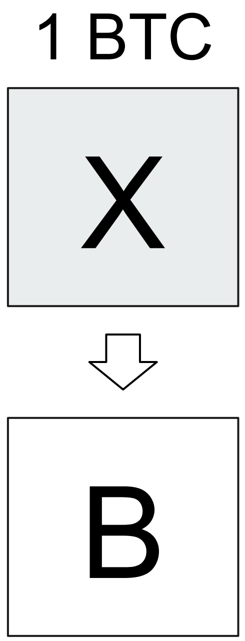
"Anything smaller than an economically viable UTXO"

- Required if Statechain entity wants to charge fees
- Needed when swapping between multiple currencies
- Ideally solved without trusting the statechain entity

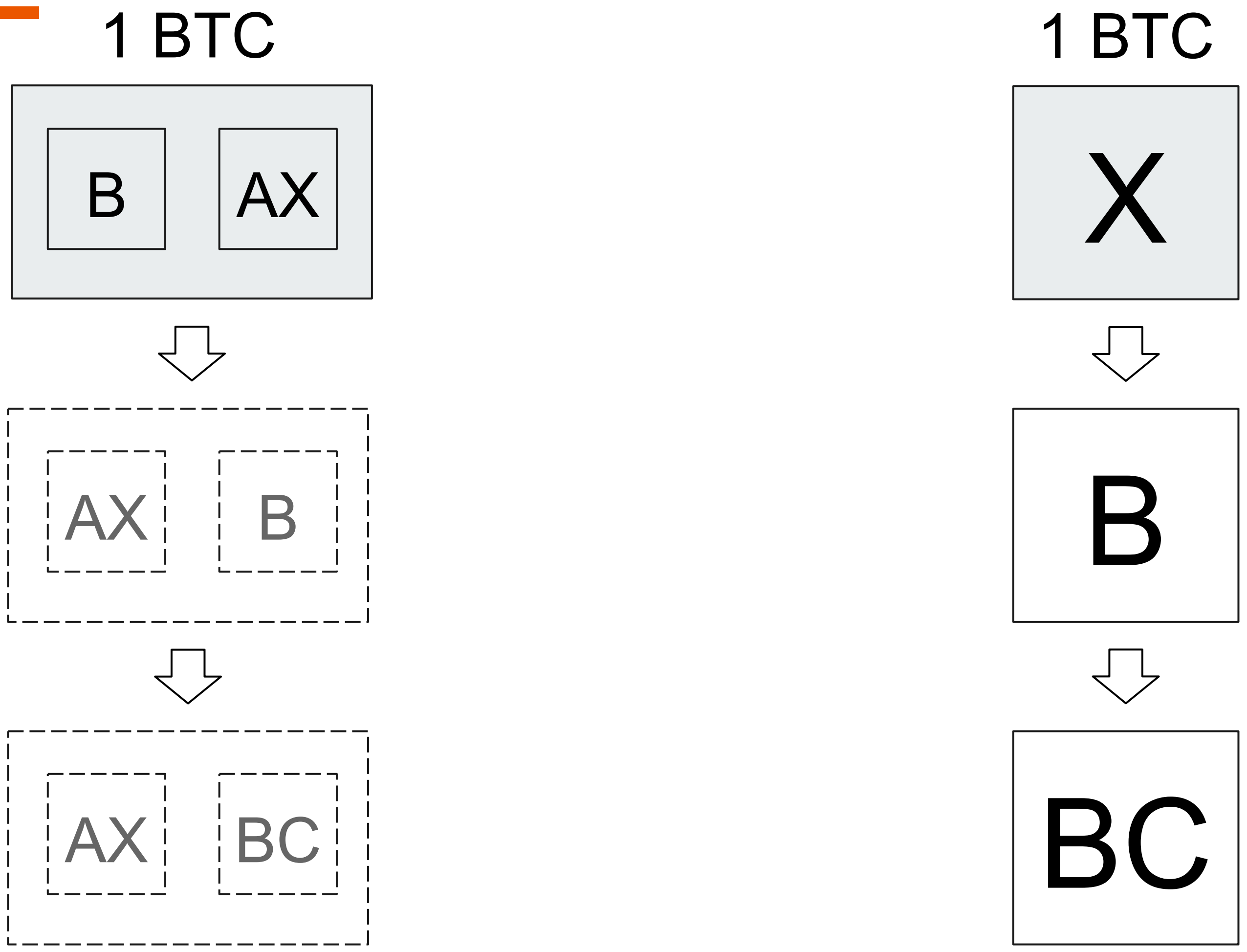
  (important legal reason: no custody over ANY coins)

Lightning Channel Creation

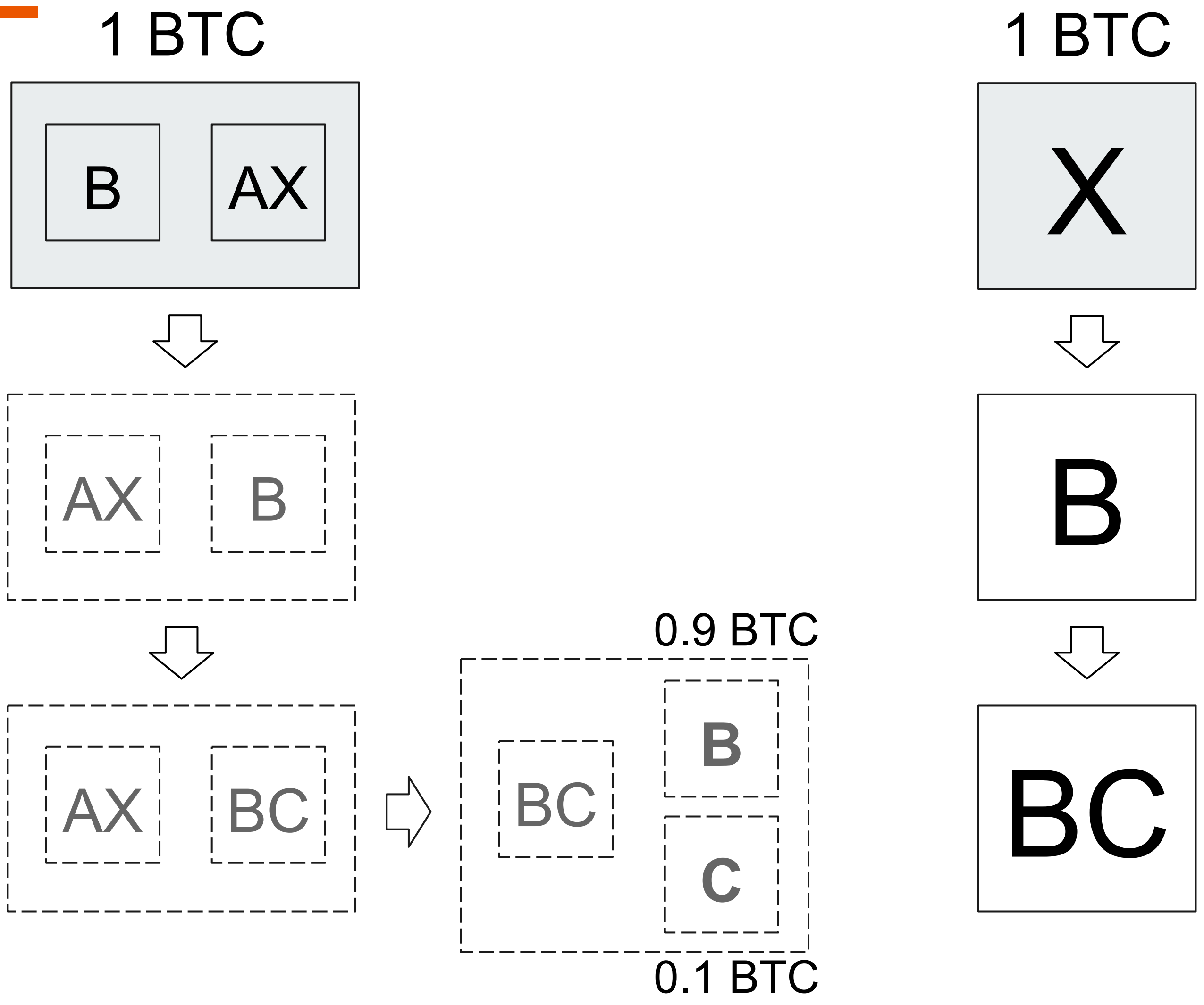




Lightning Channel Creation



Lightning Channel Creation



Lightning on Statechains

Lightning on Statechains

- Channel updated together with multi atomic swap

Lightning on Statechains

- Channel updated together with multi atomic swap
- Small channels: up to the amount of the smallest UTXO

Lightning on Statechains

- Channel updated together with multi atomic swap
- Small channels: up to the amount of the smallest UTXO
- Uncooperative close similar to regular Eltoo

Lightning on Statechains

- Channel updated together with multi atomic swap
- Small channels: up to the amount of the smallest UTXO
- Uncooperative close similar to regular Eltoo
- Close/reopen channel low-friction: it’s all off-chain!

  (e.g. adding/removing bitcoins)

Potential Use Cases

Potential Use Cases

- Off-chain value transfer

Potential Use Cases

- Off-chain value transfer
- Platform for Lightning channels

Potential Use Cases

- Off-chain value transfer
- Platform for Lightning channels
- Betting channels (multisig, Discreet Log Contracts)

Potential Use Cases

- Off-chain value transfer
- Platform for Lightning channels
- Betting channels (multisig, Discreet Log Contracts)
- Fork-agnostic ETF (UTXOs don’t move)

Further topics

Further topics

- Non-interactive version

Further topics

- Non-interactive version
- Use HSM to transfer transitory key (attestation)

Further topics

- Non-interactive version
- Use HSM to transfer transitory key (attestation)
- Graftroot withdrawal (allows redeeming forks)

Further topics

- Non-interactive version
- Use HSM to transfer transitory key (attestation)
- Graftroot withdrawal (allows redeeming forks)
- Succinctly store and relay statechain (per UTXO)

Further topics

- Non-interactive version
- Use HSM to transfer transitory key (attestation)
- Graftroot withdrawal (allows redeeming forks)
- Succinctly store and relay statechain (per UTXO)
- Variant using blind signatures:

  Entity unaware which UTXOs it holds (unblind p2p)

Paper: goo.gl/RWQ4ue

Email: rsomsen@gmail.com

Twitter: @SomsenRuben

Thank You

Paper: goo.gl/RWQ4ue

Email: rsomsen@gmail.com

Twitter: @SomsenRuben

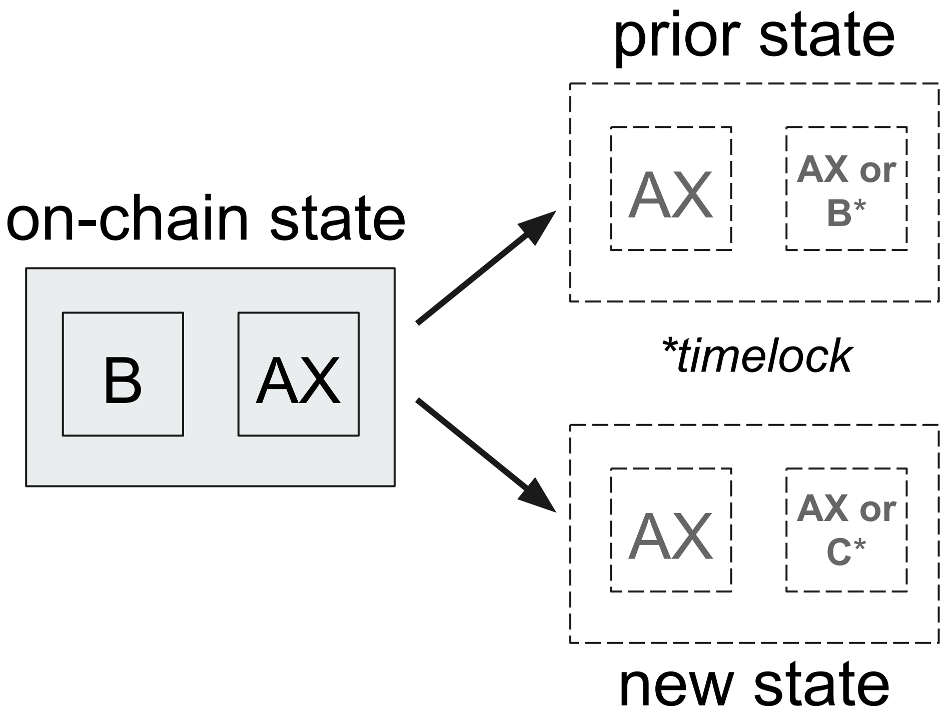
Thank You

Paper: goo.gl/RWQ4ue

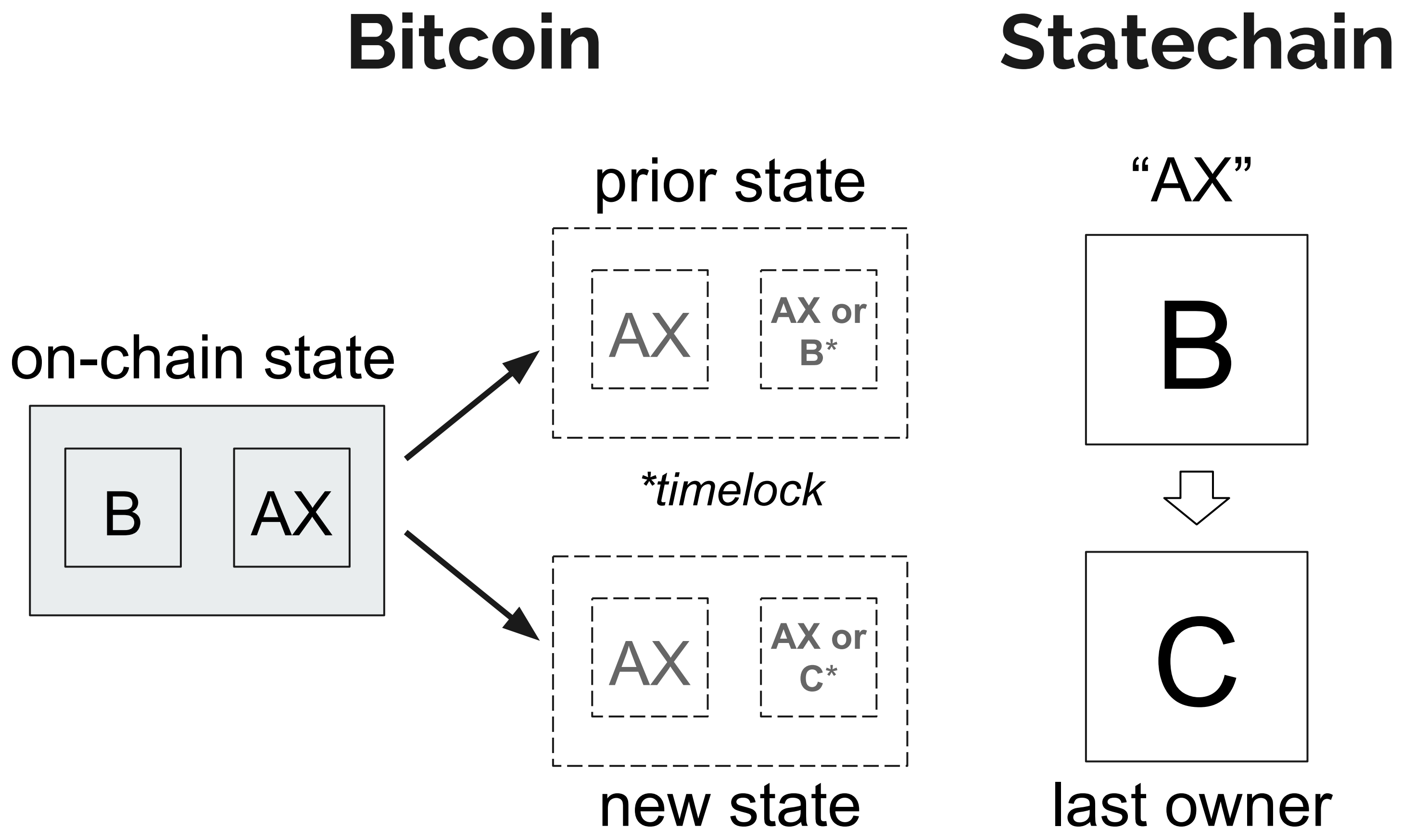
Email: rsomsen@gmail.com

Twitter: @SomsenRuben

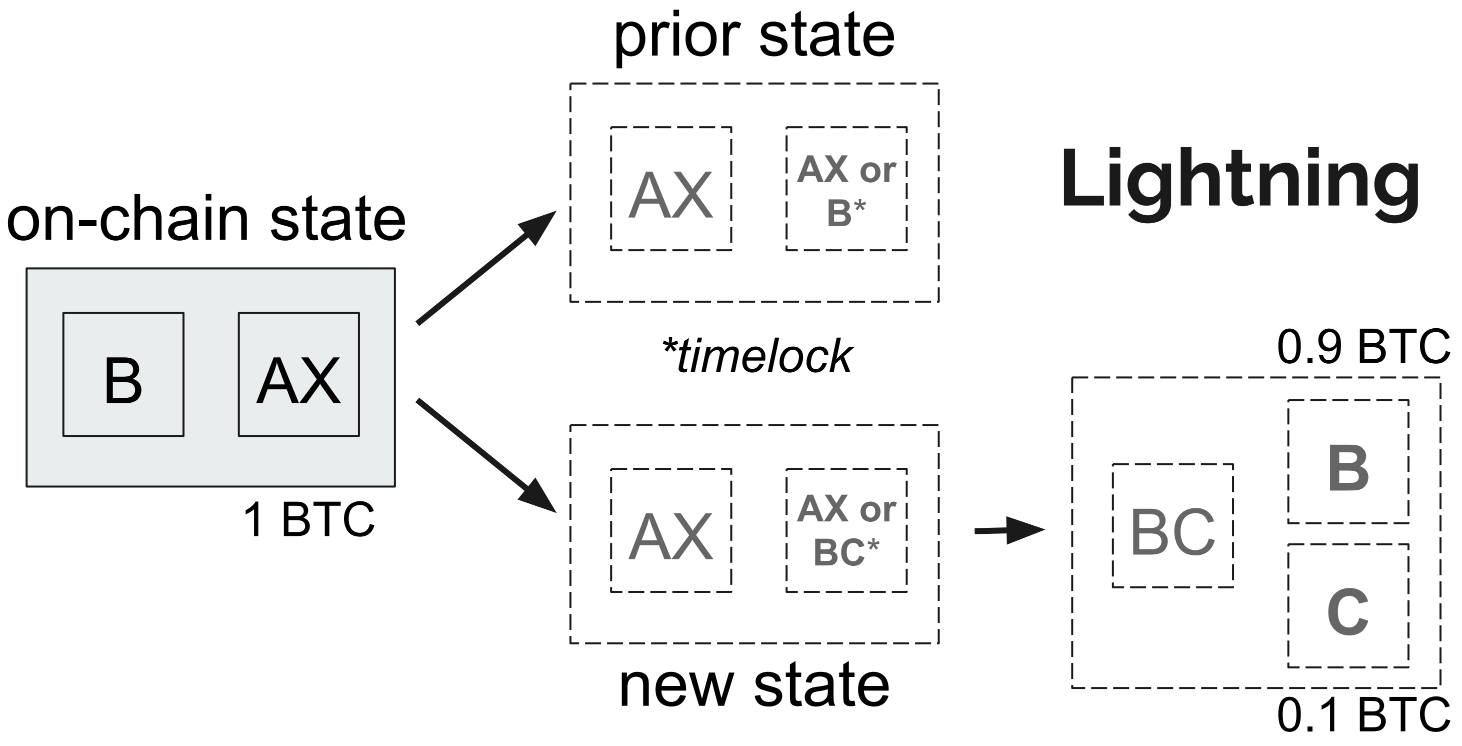
Bitcoin



prior state
AX
AX or B*
on-chain state
B
AX
*timelock
AX
AX or C*
new state


Bitcoin
Statechain
on-chain state
B
AX
prior state
AX
AX or B*
*timelock
AX
AX or C*
new state
"AX"
B
C
last owner

Lightning


prior state
on-chain state
B
AX
1 BTC
AX
AX or B*
*timelock
AX
AX or BC*
new state
0.9 BTC
BC
B
C
0.1 BTC