

Ark

a new Bitcoin layer 2 protocol

Who am I?

Who am I?

- Steven Roose

Who am I?

- Steven Roose
- Bitcoin dev for over 10 years

Who am I?

- Steven Roose
- Bitcoin dev for over 10 years
- Liquid team @ Blockstream

Who am I?

- Steven Roose
- Bitcoin dev for over 10 years
- Liquid team @ Blockstream
- rust-bitcoin

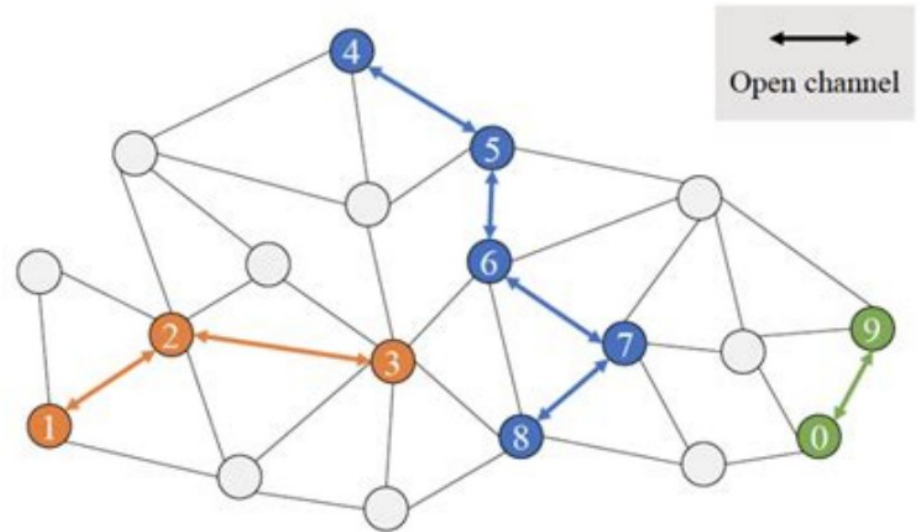
Lightning Network

Lightning Network

- off-chain payment protocol

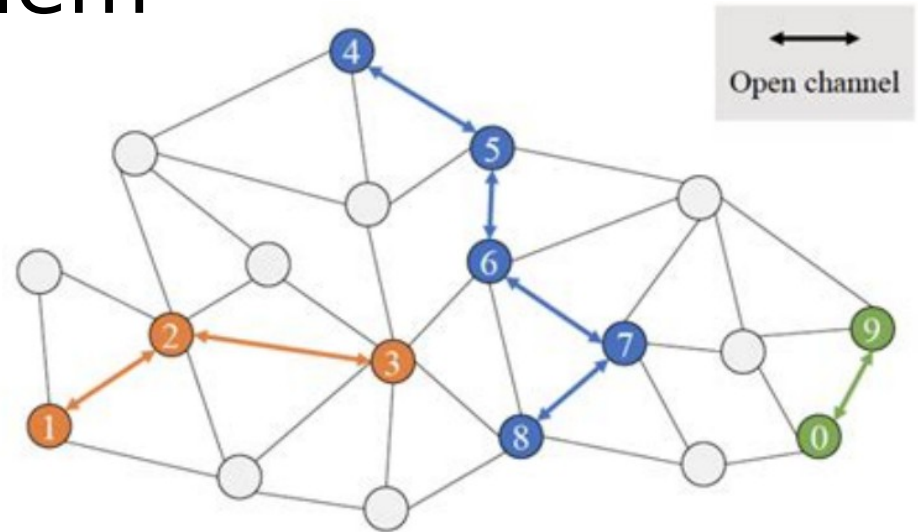
Lightning Network

- off-chain payment protocol
- connected graph of two-party channels



Lightning Network

- off-chain payment protocol
- connected graph of two-party channels
- inbound liquidity problem



What is Ark?

What is Ark?

- new layer 2 protocol for Bitcoin

What is Ark?

- new layer 2 protocol for Bitcoin
 - interoperable with Lightning

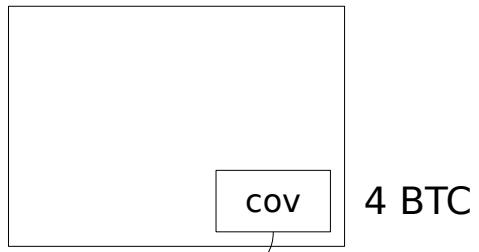
What is Ark?

- new layer 2 protocol for Bitcoin
 - interoperable with Lightning
- sharing UTXOs with many users: VTXOs

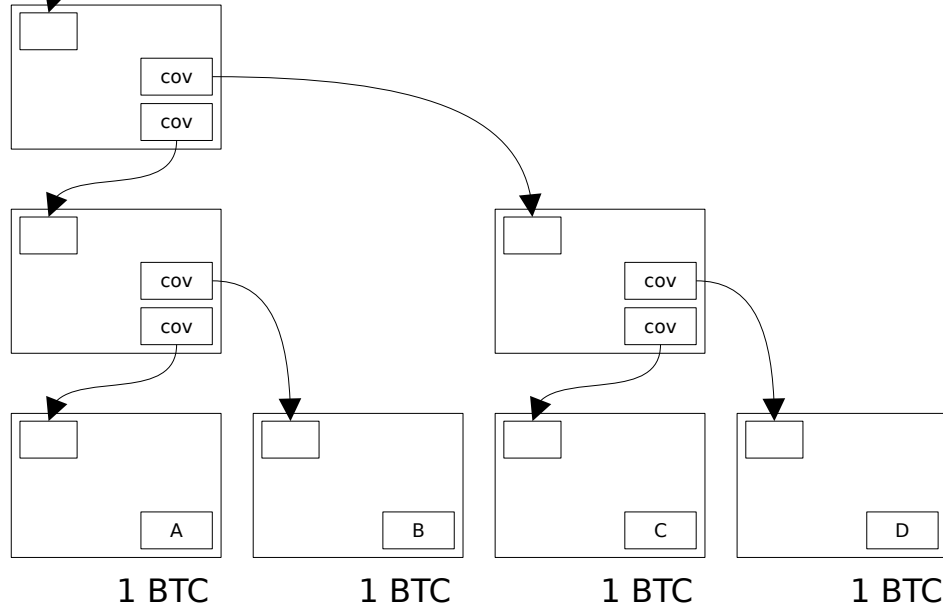
What is Ark?

- new layer 2 protocol for Bitcoin
 - interoperable with Lightning
- sharing UTXOs with many users: VTXOs
 - exchanging VTXOs for new VTXOs

on-chain

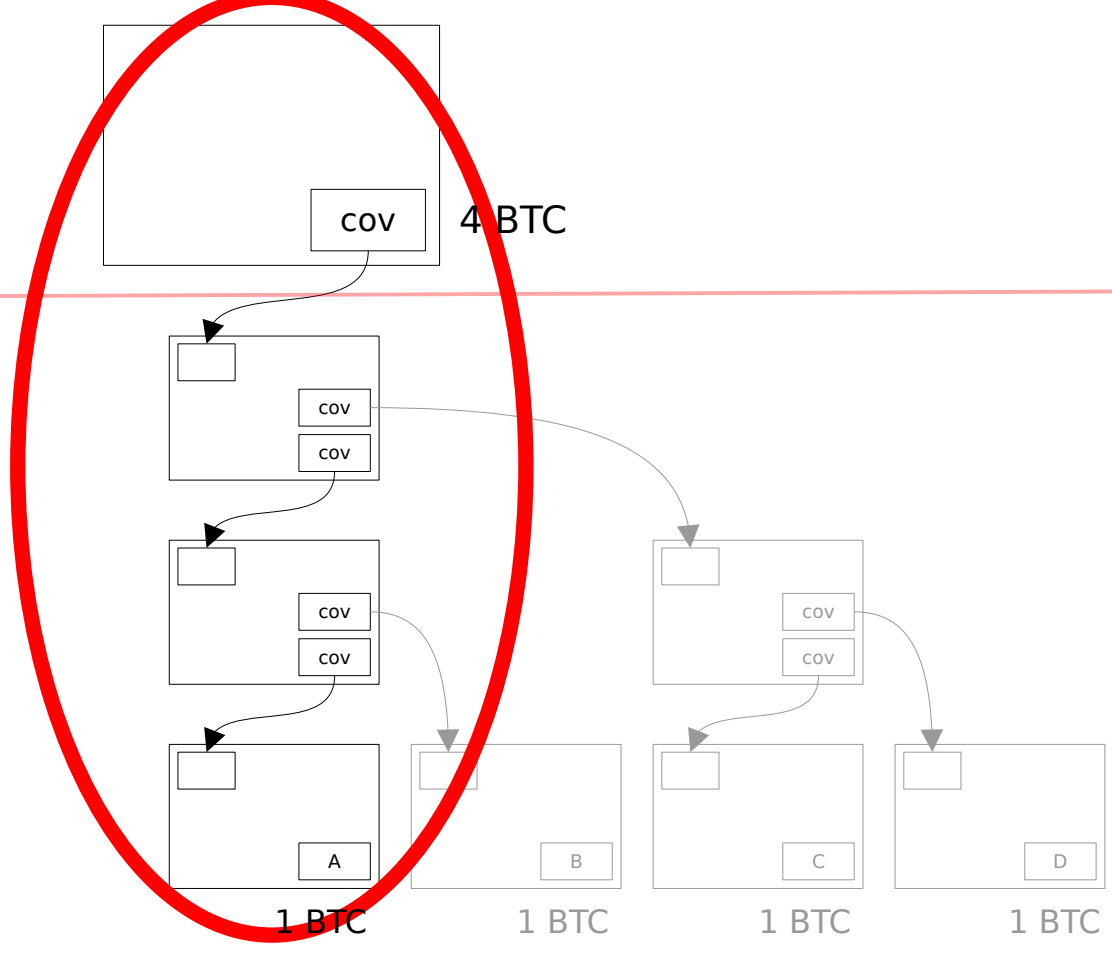


off-chain



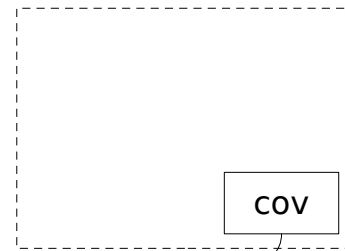
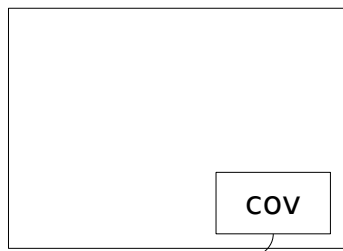
on-chain

off-chain

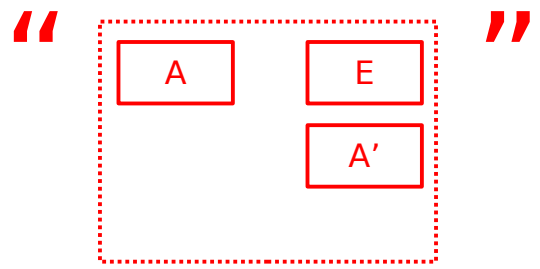
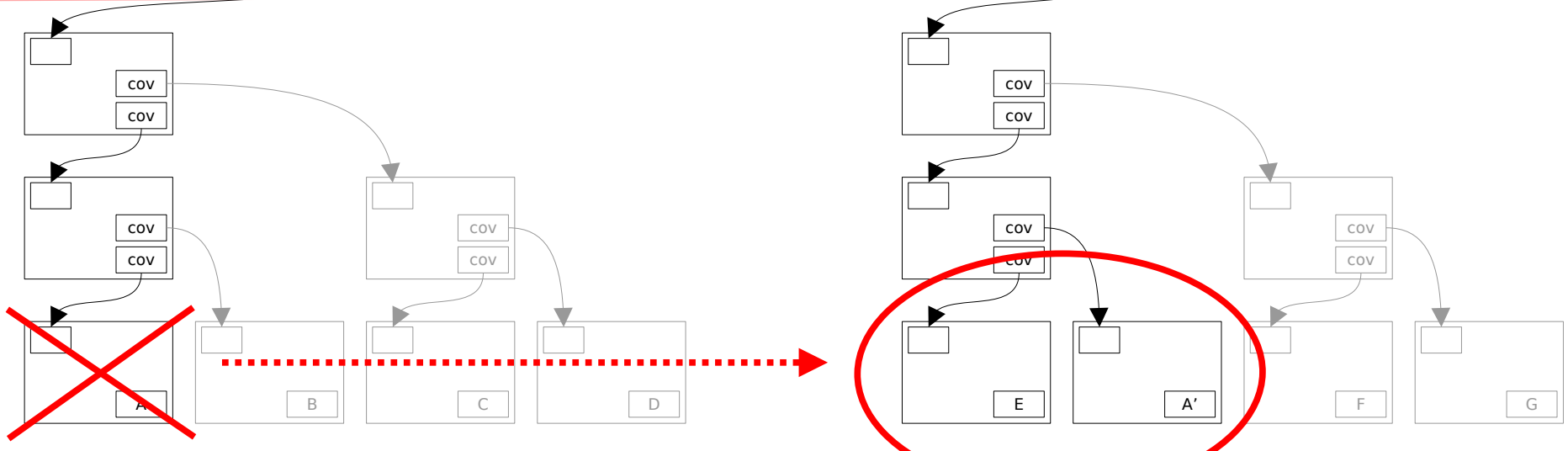


vTXO

on-chain



off-chain



What is (an) Ark?

What is (an) Ark?

- series of “Ark rounds”

What is (an) Ark?

- series of “Ark rounds”
 - atomic spending VTXOs to create new ones

What is (an) Ark?

- series of “Ark rounds”
 - atomic spending VTXOs to create new ones
 - one on-chain Bitcoin tx per round

What is (an) Ark?

- series of “Ark rounds”
 - atomic spending VTXOs to create new ones
 - one on-chain Bitcoin tx per round
- single service provider: “ASP”

What is (an) Ark?

- series of “Ark rounds”
 - atomic spending VTXOs to create new ones
 - one on-chain Bitcoin tx per round
- single service provider: “ASP”
 - coordinates & provides liquidity

What is (an) Ark?

- series of “Ark rounds”
 - atomic spending VTXOs to create new ones
 - one on-chain Bitcoin tx per round
- single service provider: “ASP”
 - coordinates & provides liquidity
 - users always 100% in control of money

What is (an) Ark?

What is (an) Ark?

- efficient UTXO-style off-chain txs

What is (an) Ark?

- efficient UTXO-style off-chain txs
- only client-server interactions
 - much simpler than Lightning

What is (an) Ark?

- efficient UTXO-style off-chain txs
- only client-server interactions
 - much simpler than Lightning
- anyone can receive (no liquidity required!)

What is (an) Ark?

- efficient UTXO-style off-chain txs
- only client-server interactions
 - much simpler than Lightning
- anyone can receive (no liquidity required!)
- VTXO expiry

Lightning

Lightning

- make Lightning payments from Ark

Lightning

- make Lightning payments from Ark
 - similar to regular Ark tx

Lightning

- make Lightning payments from Ark
 - similar to regular Ark tx
 - ASP functions as LSP*

Lightning

- make Lightning payments from Ark
 - similar to regular Ark tx
 - ASP functions as LSP*
- create Lightning channels inside Ark

Lightning

- make Lightning payments from Ark
 - similar to regular Ark tx
 - ASP functions as LSP*
- create Lightning channels inside Ark
 - Ark as “channel factory”

Lightning

- make Lightning payments from Ark
 - similar to regular Ark tx
 - ASP functions as LSP*
- create Lightning channels inside Ark
 - Ark as “channel factory”
 - cheap channels with expiry

Privacy

Privacy

- the ASP has full insight in txs

Privacy

- the ASP has full insight in txs
- solution: blinded coinjoins

Privacy

- the ASP has full insight in txs
- solution: blinded coinjoins
 - entire Ark round as anonymity set

Privacy

- the ASP has full insight in txs
- solution: blinded coinjoins
 - entire Ark round as anonymity set
 - similar to WabiSabi coin mixing

Thanks

- full technical explanation:
 - <https://roose.io/presentations>
- <https://arkpill.me/>
- Questions?

Understanding Ark

a new Bitcoin layer 2 protocol

Who am I?

Who am I?

- Steven Roose

Who am I?

- Steven Roose
- Bitcoin dev for over 10 years

Who am I?

- Steven Roose
- Bitcoin dev for over 10 years
- Liquid team @ Blockstream

Who am I?

- Steven Roose
- Bitcoin dev for over 10 years
- Liquid team @ Blockstream
- rust-bitcoin

Understanding Ark

Understanding Ark

- no sales talk, just technical explanation

Understanding Ark

- no sales talk, just technical explanation
- from the ground up

Understanding Ark

- no sales talk, just technical explanation
- from the ground up
 - might slightly differ from other explanations

Understanding Ark

- no sales talk, just technical explanation
- from the ground up
 - might slightly differ from other explanations
- assume covenants*

Covenants

Covenants

- restriction on where the money in a UTXO can go

Covenants

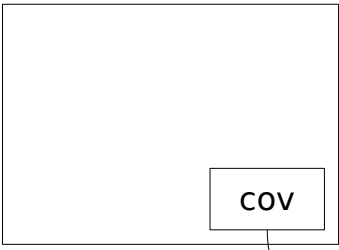
- restriction on where the money in a UTXO can go
- for now: an output that can only be spent using a single pre-specified transaction

off-chain

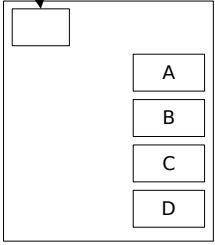
on-chain



on-chain



off-chain



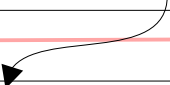
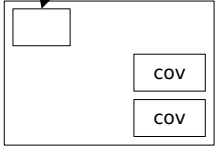
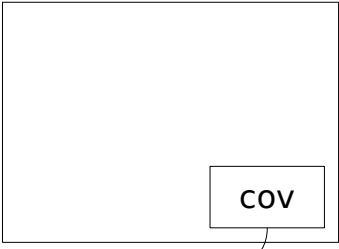
off-chain

on-chain



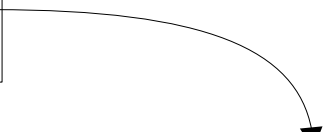
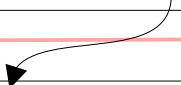
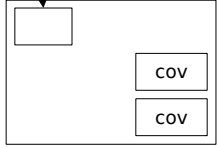
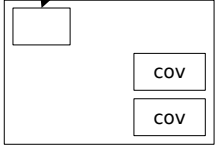
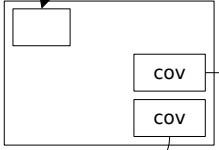
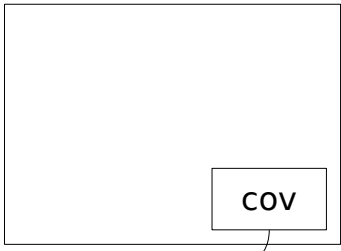
off-chain

on-chain

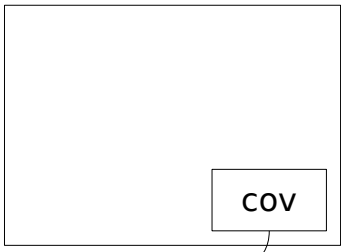


off-chain

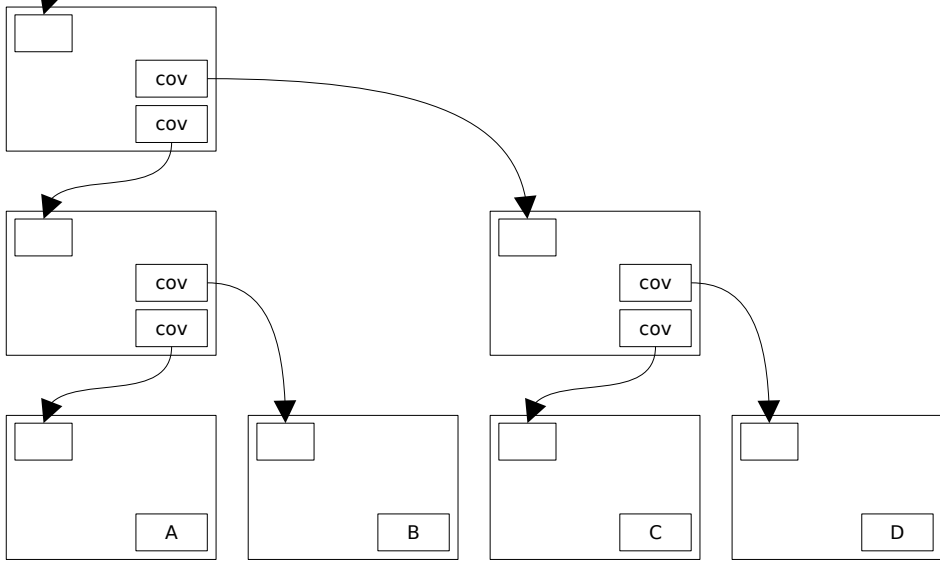
on-chain



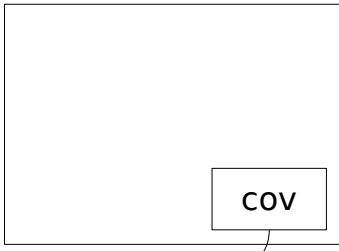
on-chain



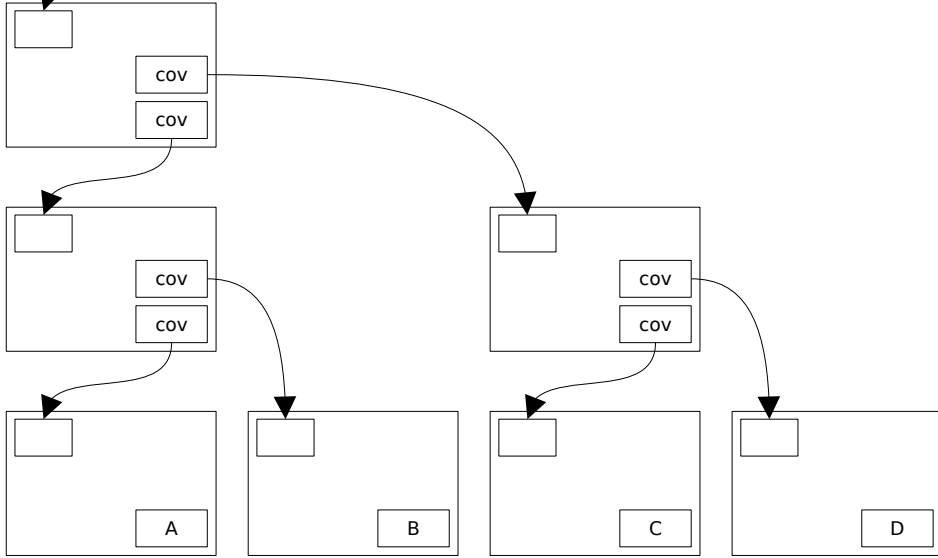
off-chain



on-chain

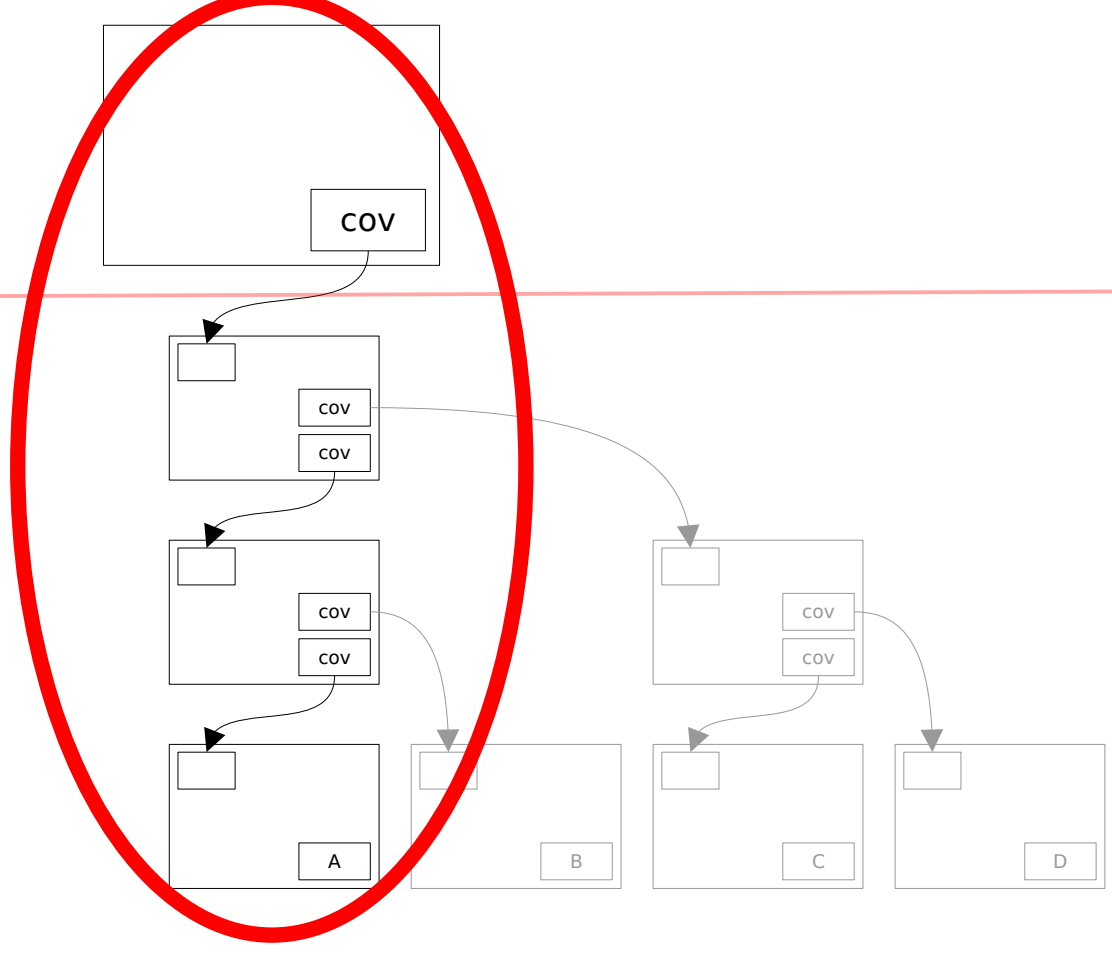


off-chain



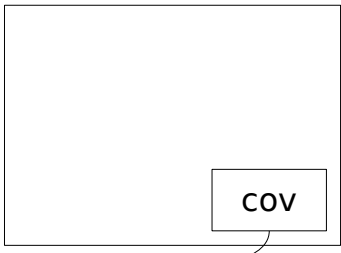
on-chain

off-chain

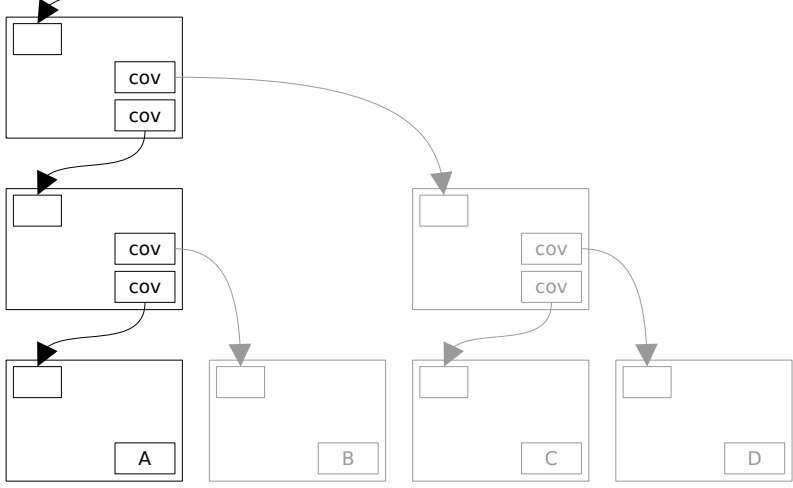


vTXO

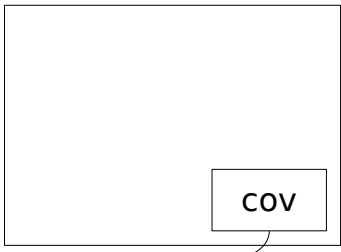
on-chain



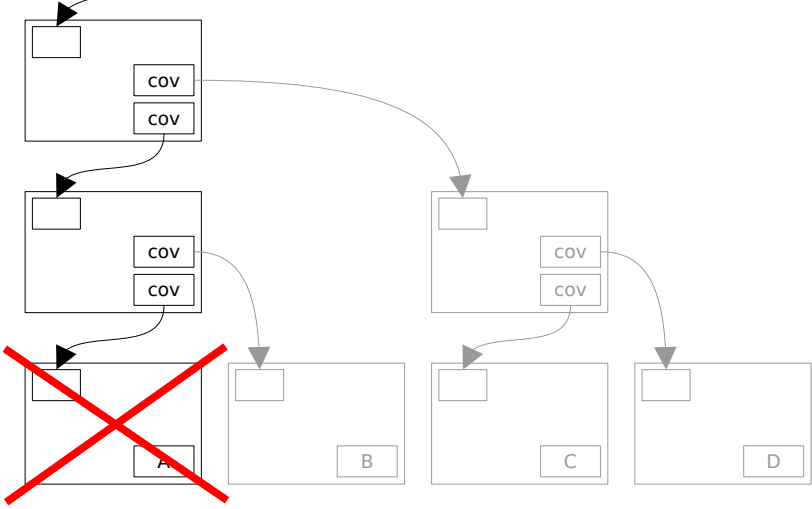
off-chain



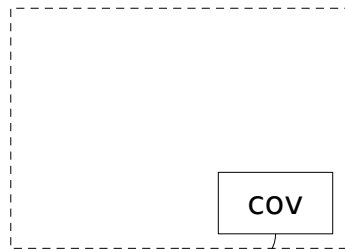
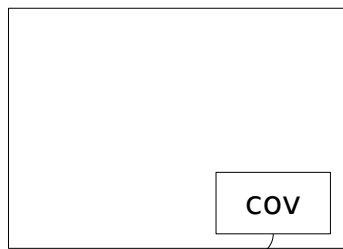
on-chain



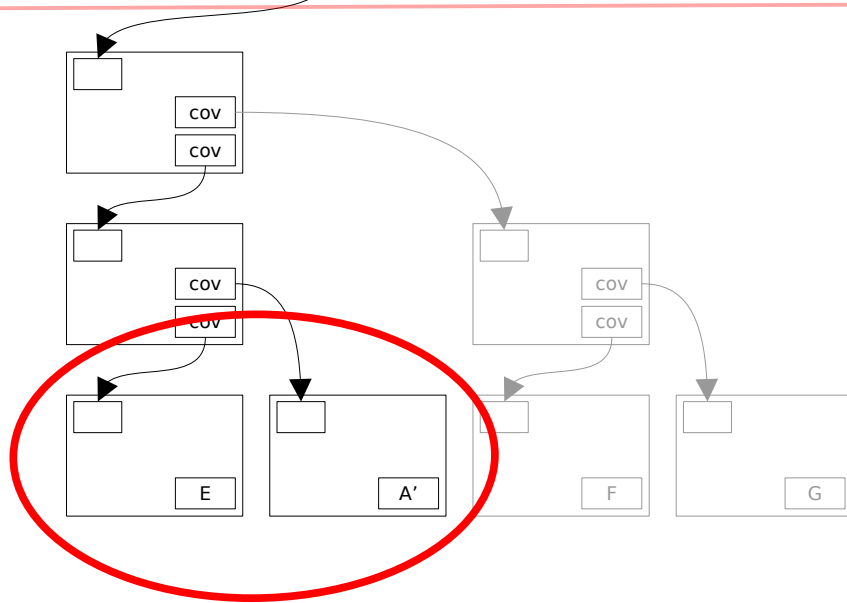
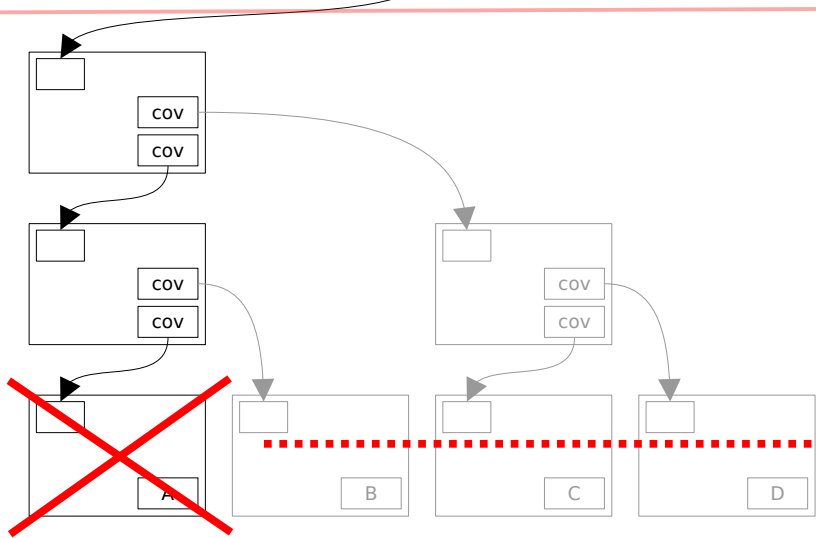
off-chain



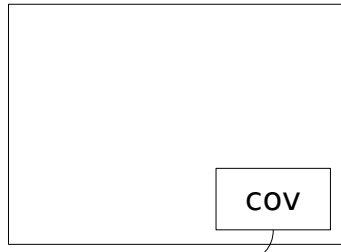
on-chain



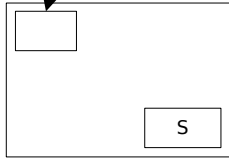
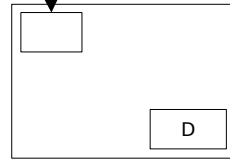
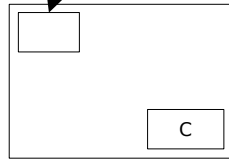
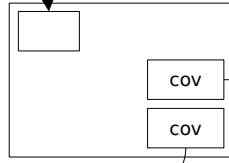
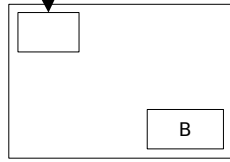
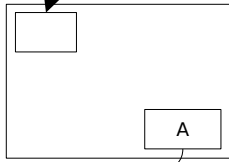
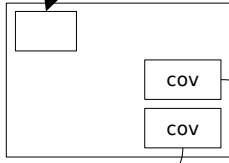
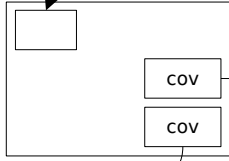
off-chain



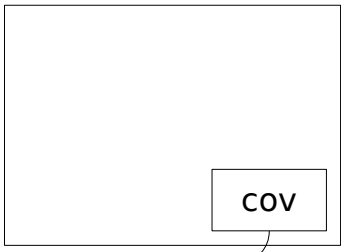
on-chain



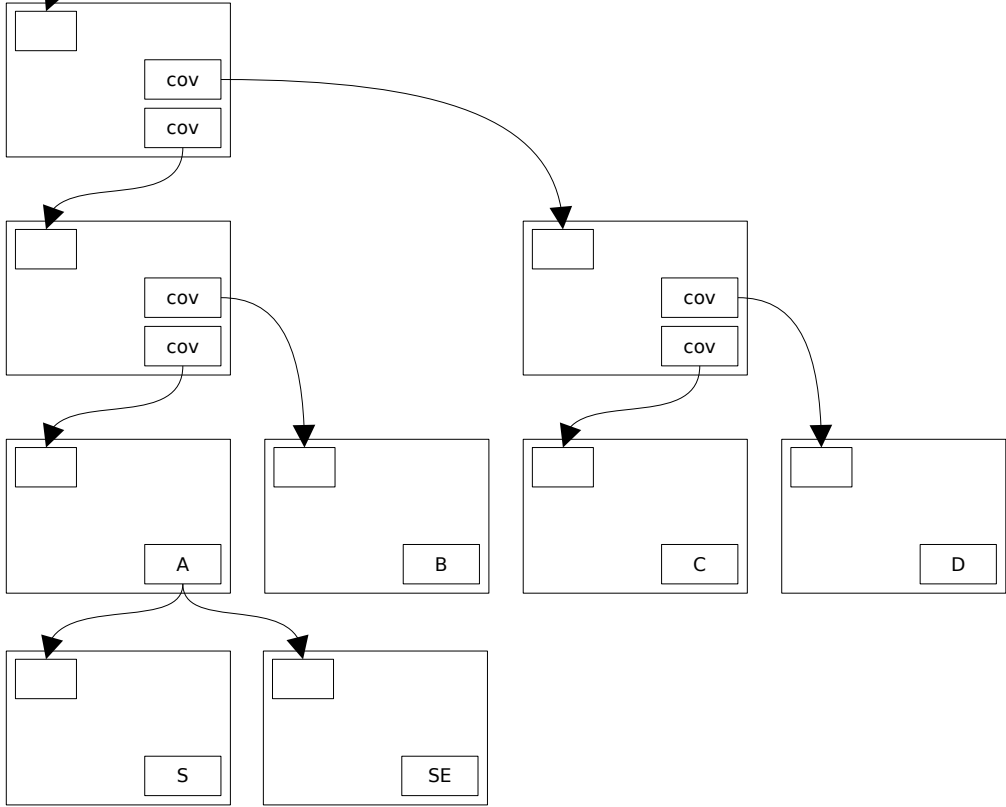
off-chain



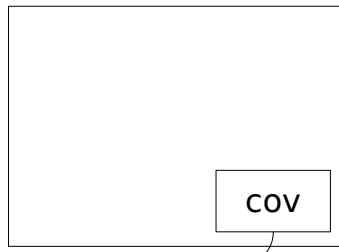
on-chain



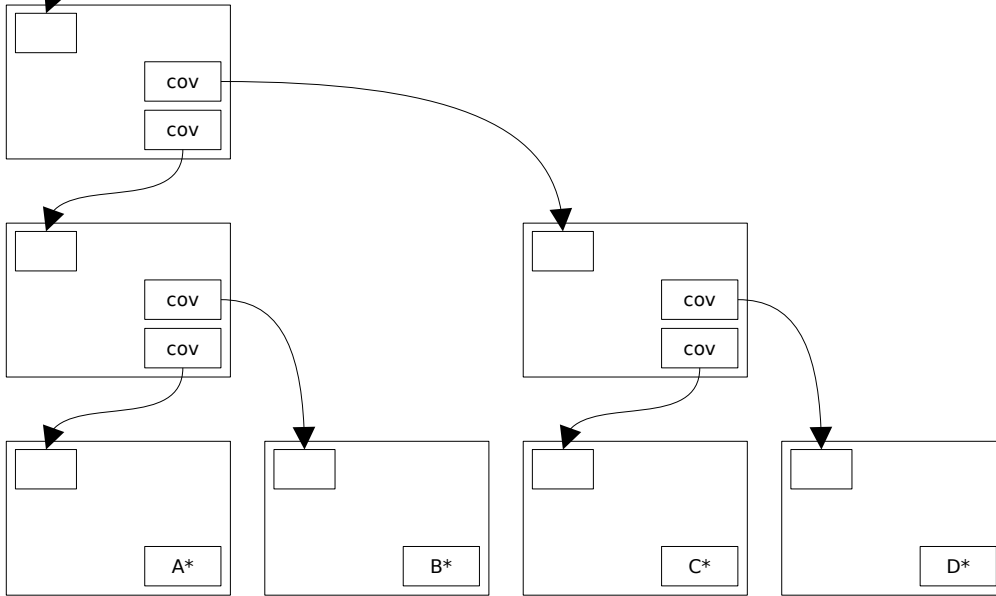
off-chain



on-chain



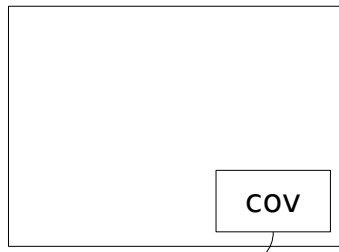
off-chain



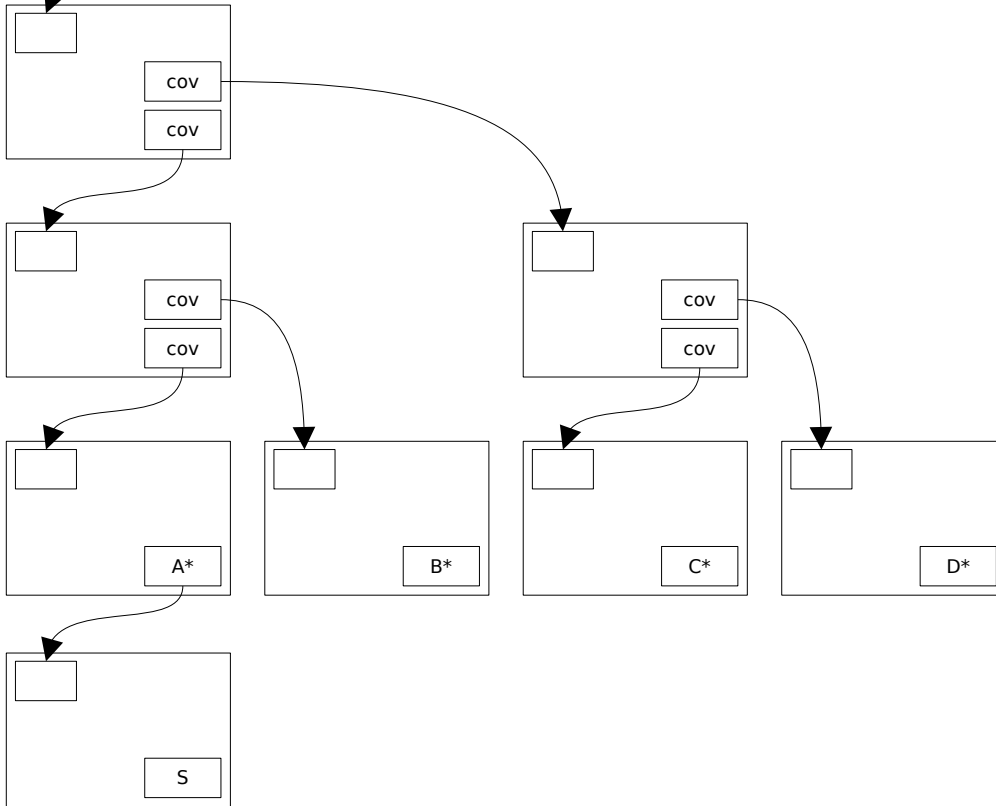
$S = \text{ASP pubkey}$

$A^* = A + S \text{ OR } (A \text{ after } 7\text{d})$

on-chain



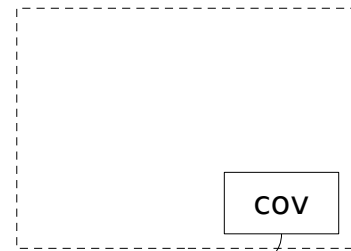
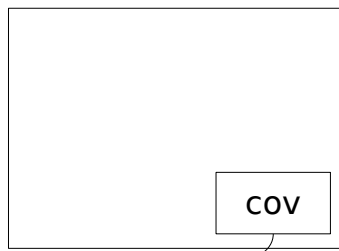
off-chain



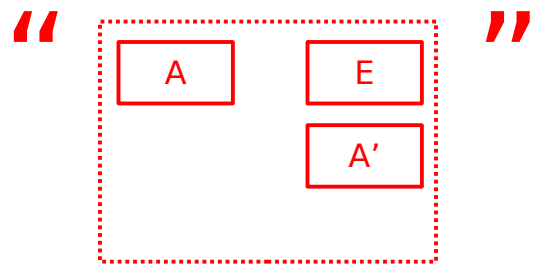
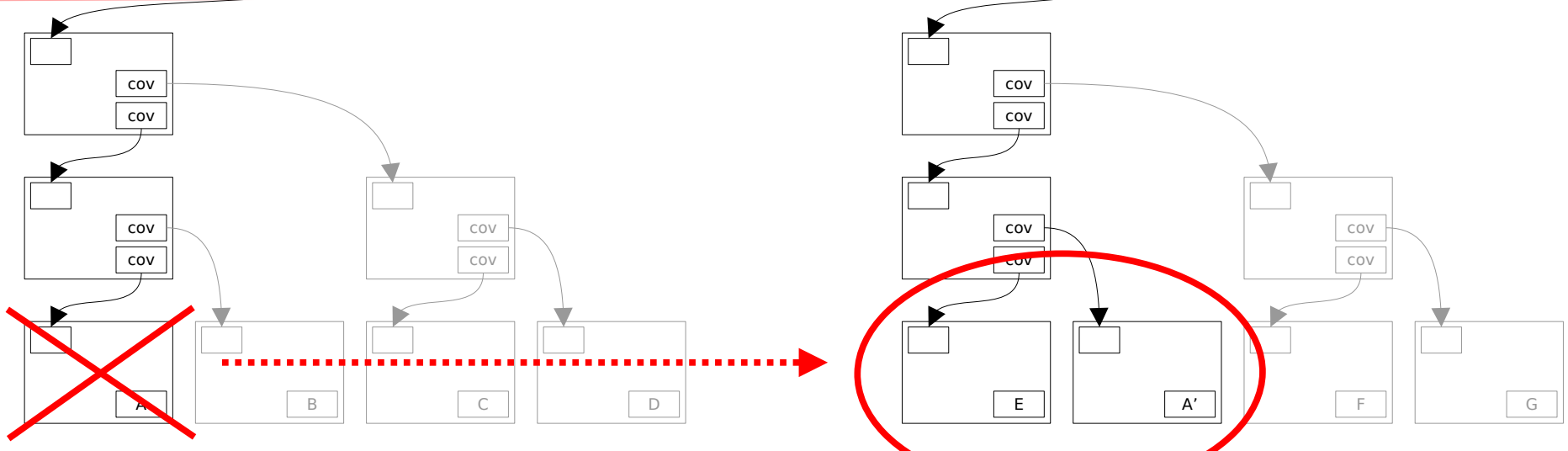
$S = \text{ASP pubkey}$

$A^* = A + S \text{ OR } (A \text{ after } 7d)$

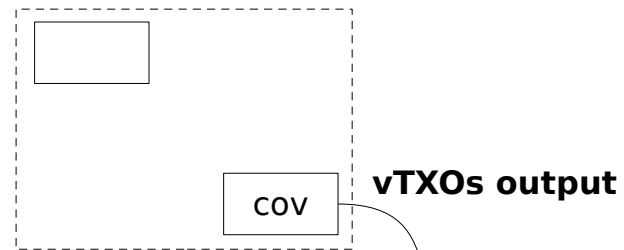
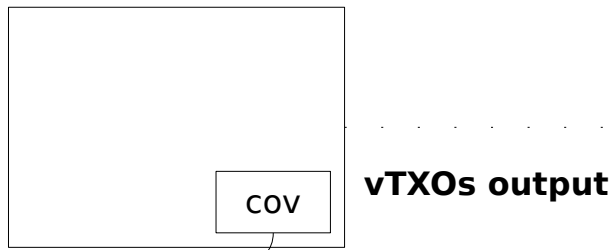
on-chain



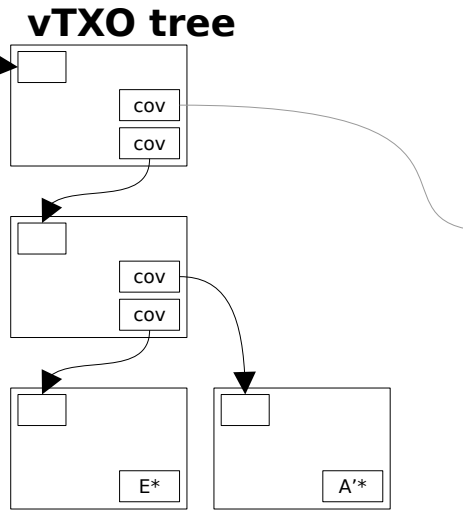
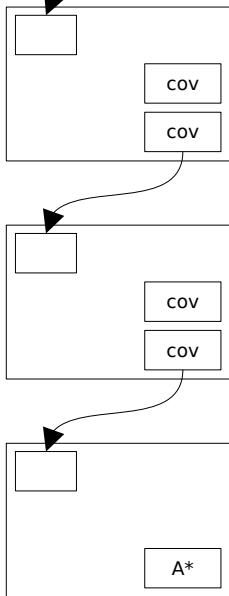
off-chain



on-chain

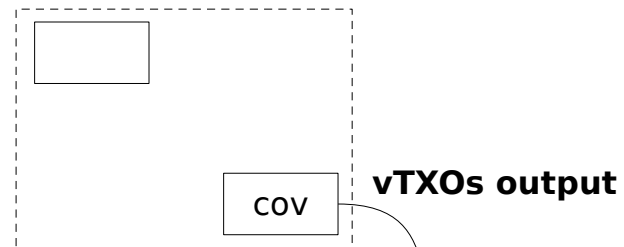
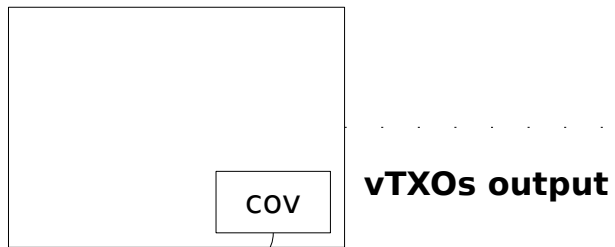


off-chain

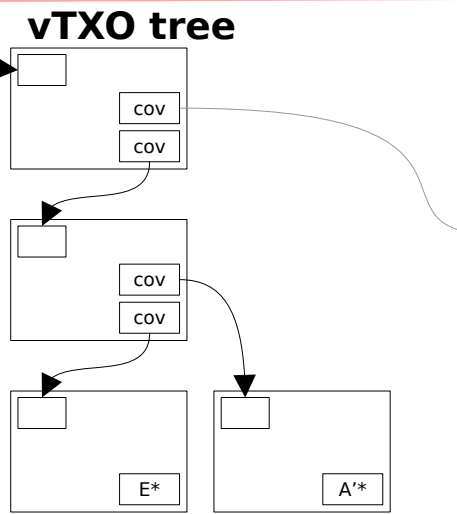
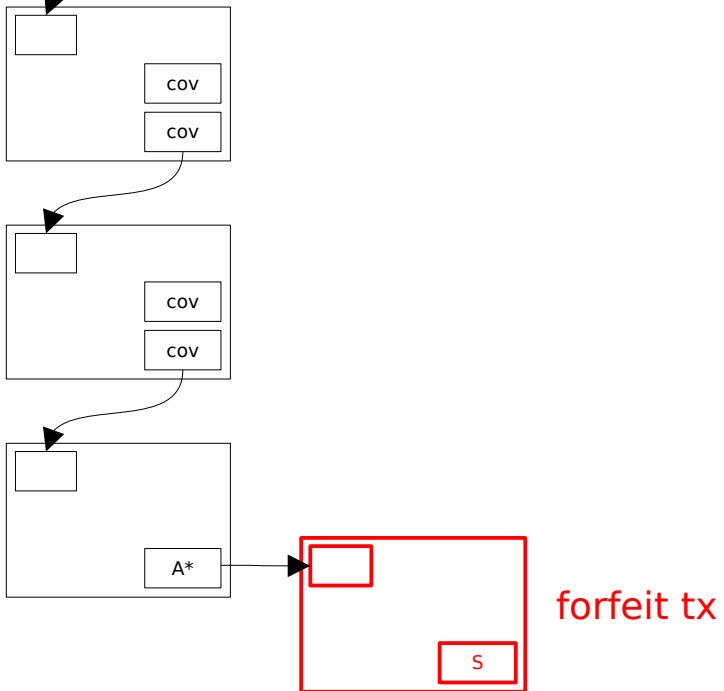


S = ASP pubkey
A* = A+S OR (A after 7d)

on-chain

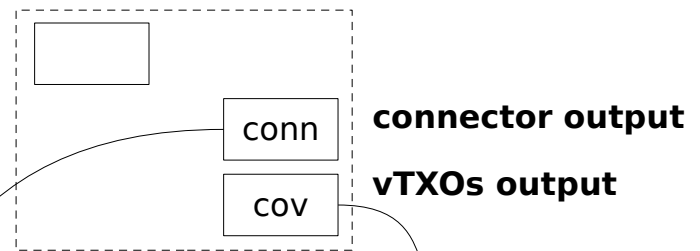
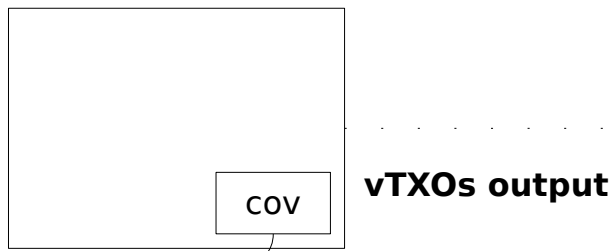


off-chain

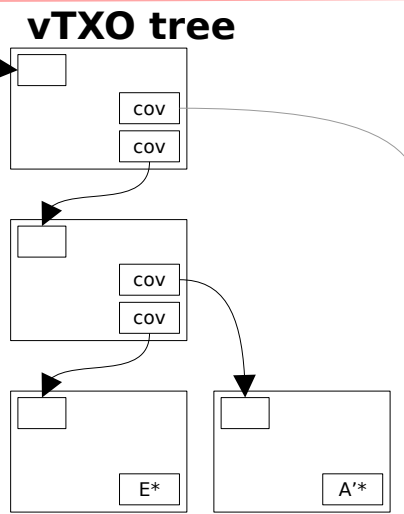
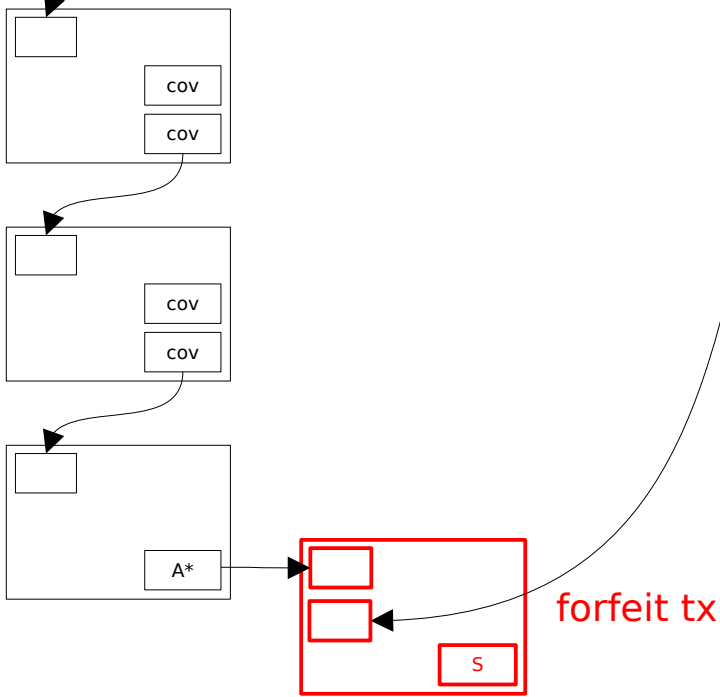


S = ASP pubkey
A* = A+S OR (A after 7d)

on-chain

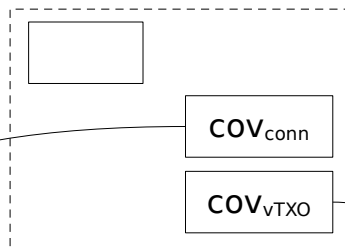
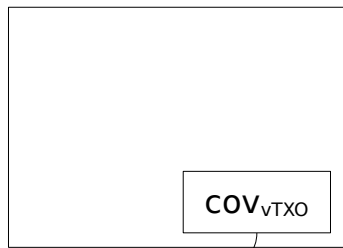


off-chain



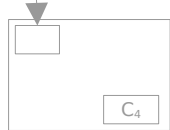
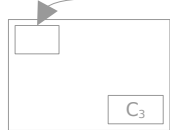
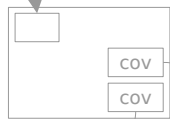
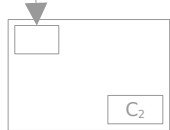
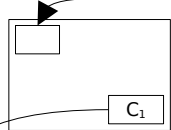
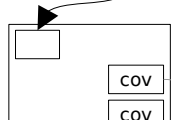
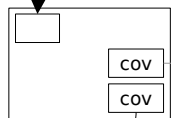
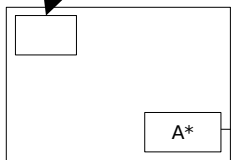
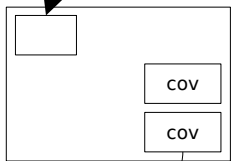
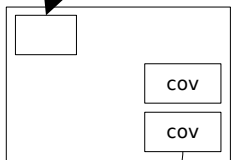
S = ASP pubkey
A* = A+S OR (A after 7d)

on-chain

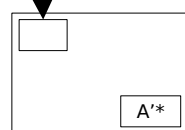
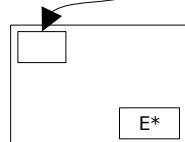
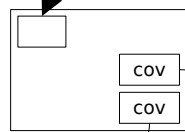
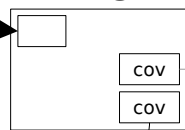


connector output
vTXOs output

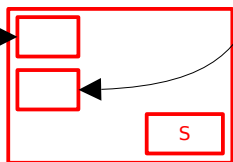
off-chain



vTXO tree



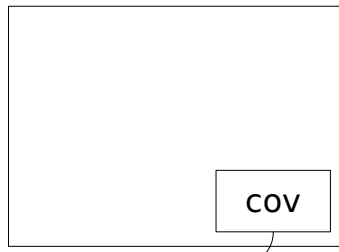
connector tree



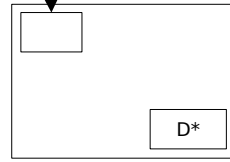
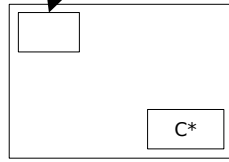
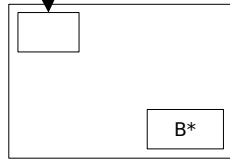
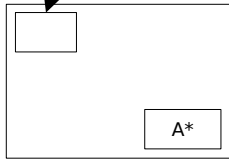
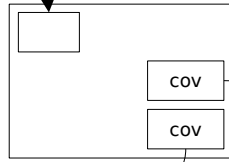
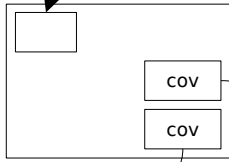
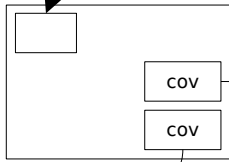
forfeit tx

S = ASP pubkey
 A^* = $A+S$ OR (A after 7d)

on-chain



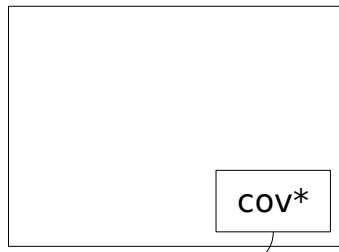
off-chain



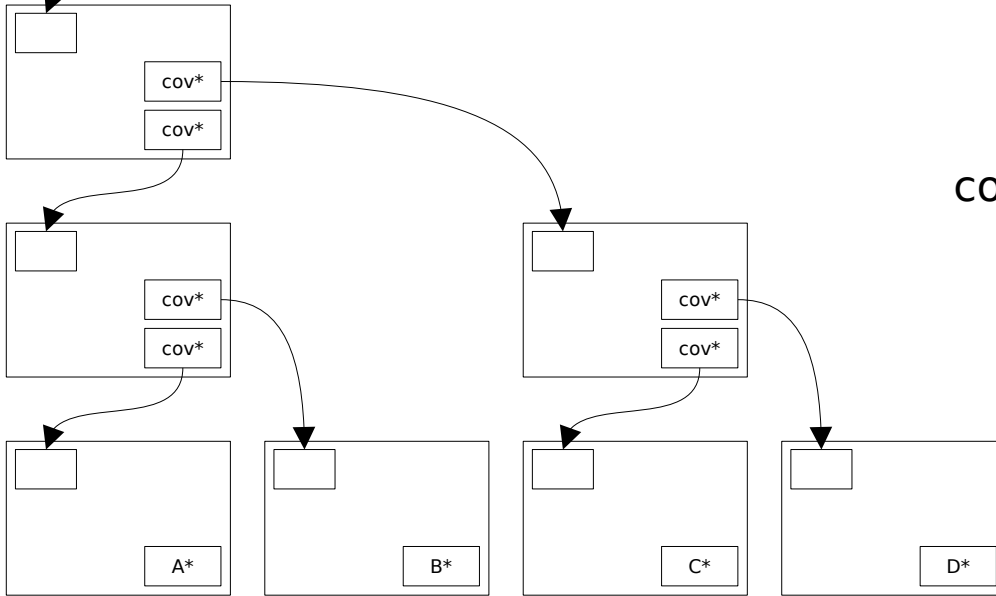
$S = \text{ASP pubkey}$

$A^* = A + S \text{ OR } (A \text{ after } 7\text{d})$

on-chain

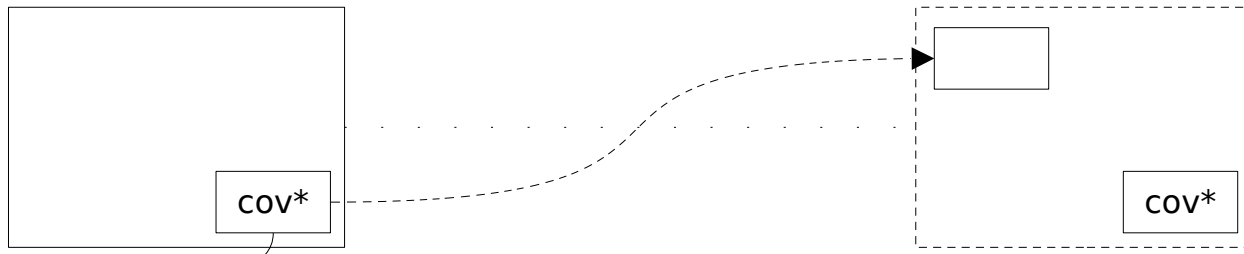


off-chain

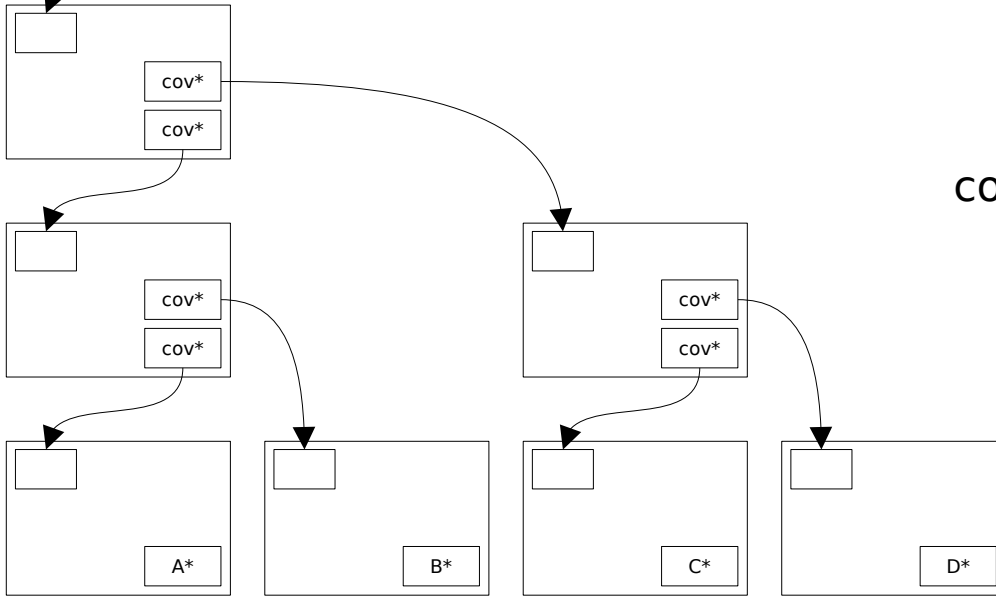


$S = \text{ASP pubkey}$
 $A^* = A + S \text{ OR } (A \text{ after } 7\text{d})$
 $\text{cov}^* = \text{cov} \text{ OR } (S \text{ after } 14\text{d})$

on-chain

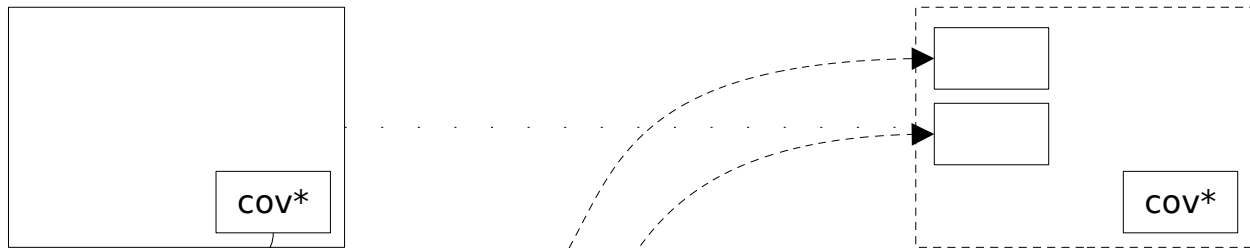


off-chain

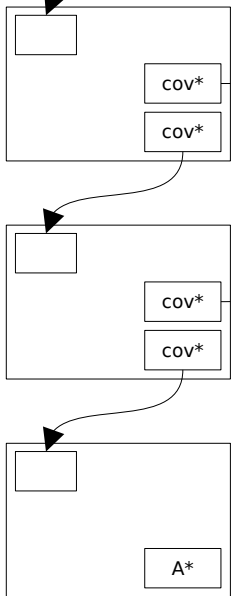


$S = \text{ASP pubkey}$
 $A^* = A + S \text{ OR } (A \text{ after } 7d)$
 $COV^* = COV \text{ OR } (S \text{ after } 14d)$

on-chain

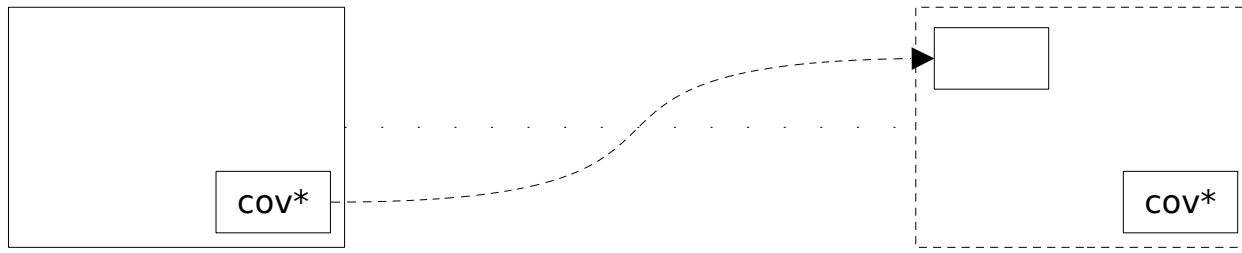


off-chain



$S = \text{ASP pubkey}$
 $A^* = A + S \text{ OR } (A \text{ after } 7d)$
 $COV^* = COV \text{ OR } (S \text{ after } 14d)$

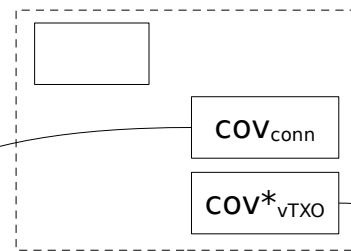
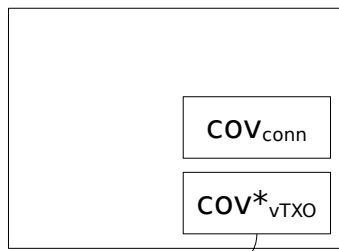
on-chain



off-chain

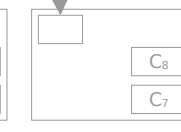
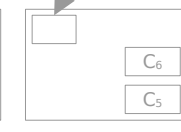
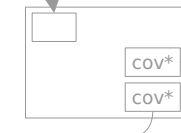
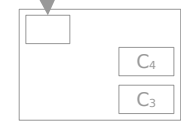
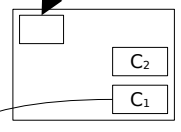
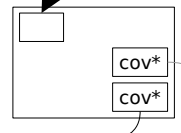
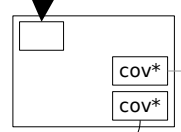
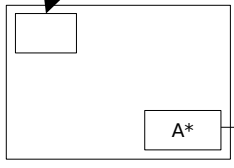
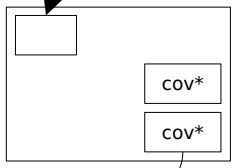
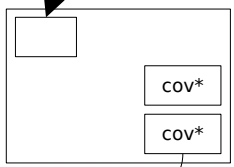
$S = \text{ASP pubkey}$
 $A^* = A + S \text{ OR } (A \text{ after } 7\text{d})$
 $\text{cov}^* = \text{cov} \text{ OR } (S \text{ after } 14\text{d})$

on-chain

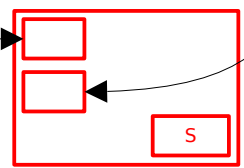
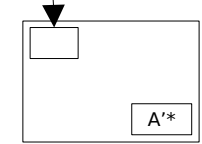
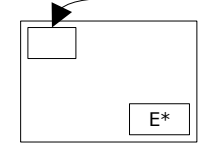
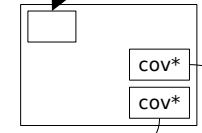
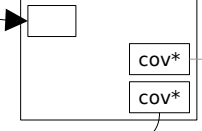


connector output
vTXOs output

off-chain



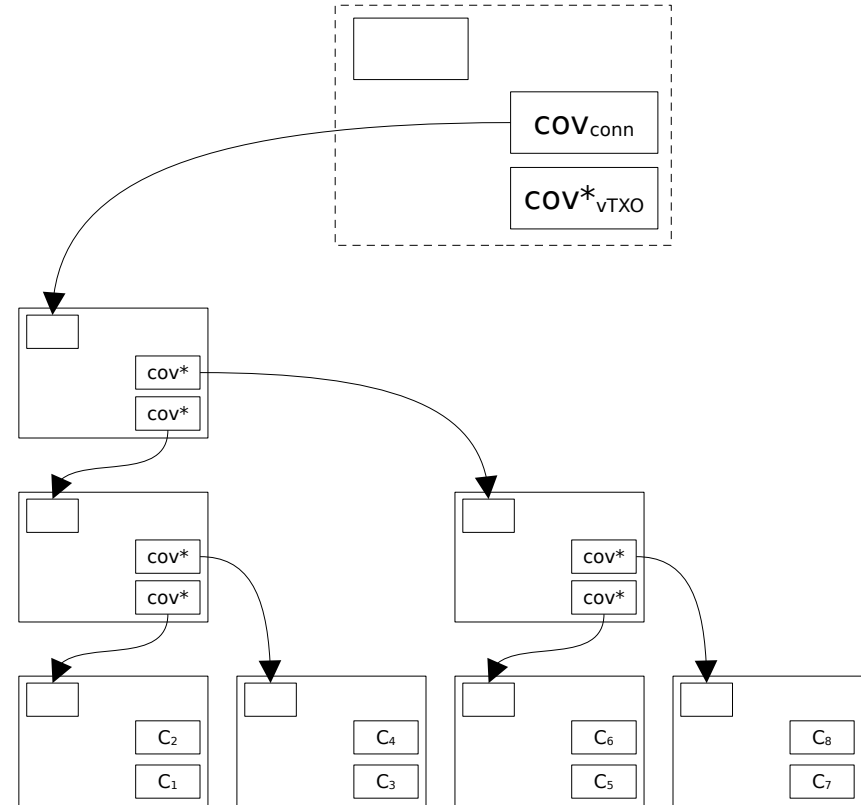
vTXO tree



forfeit tx

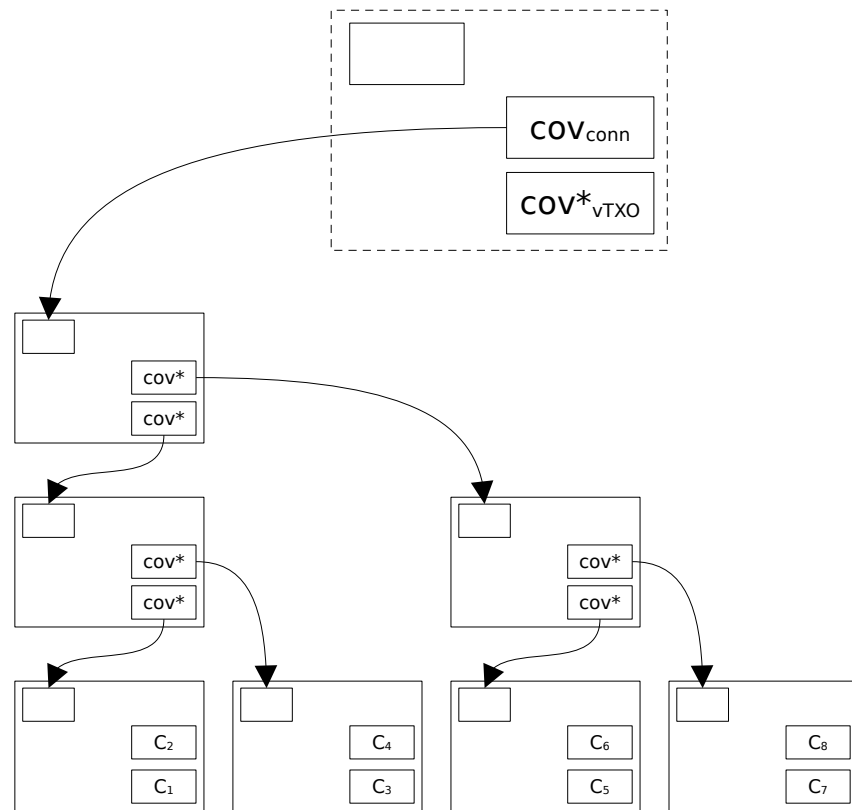
S = ASP pubkey
 A^* = $A+S$ OR (A after 7d)
 COV^* = COV OR (S after 14d)

Connectors



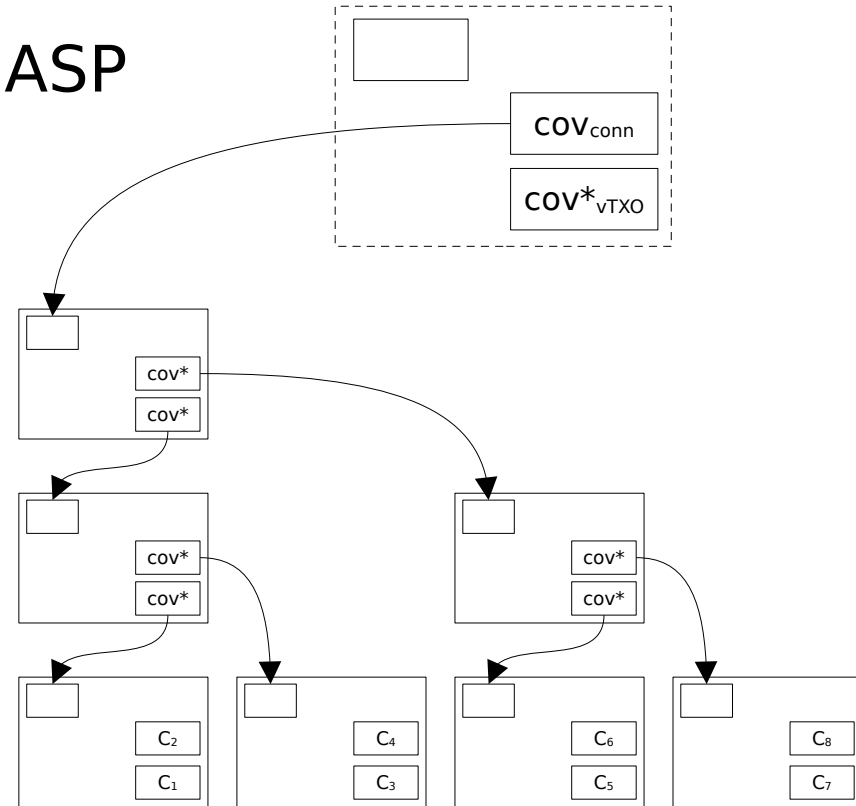
Connectors

- users only care about tx dep. chain



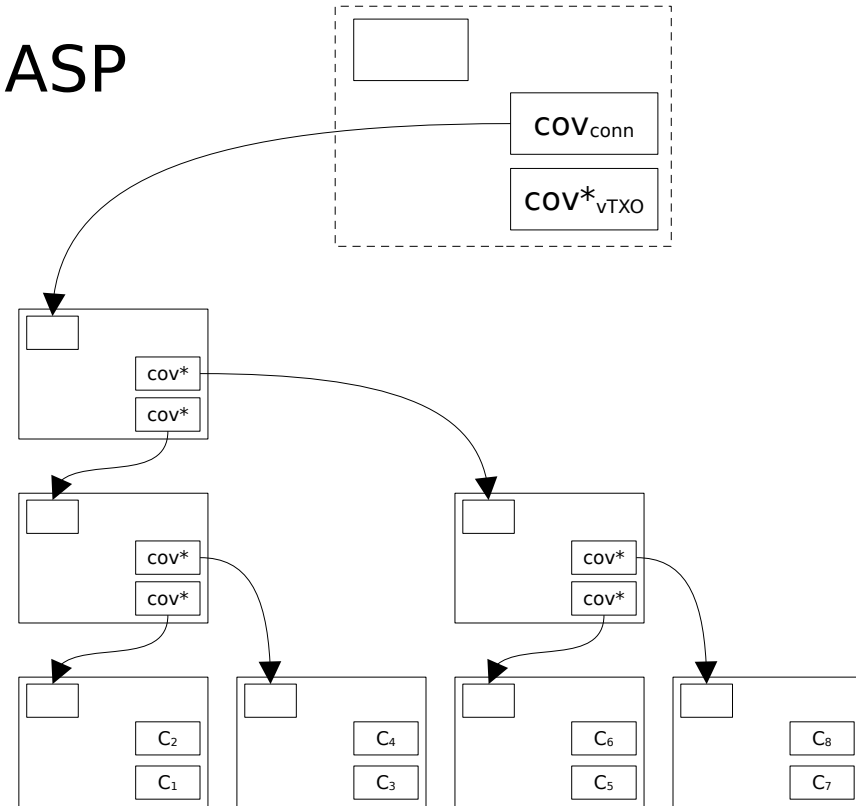
Connectors

- users only care about tx dep. chain
- simple 1-of-1 outputs owned by ASP



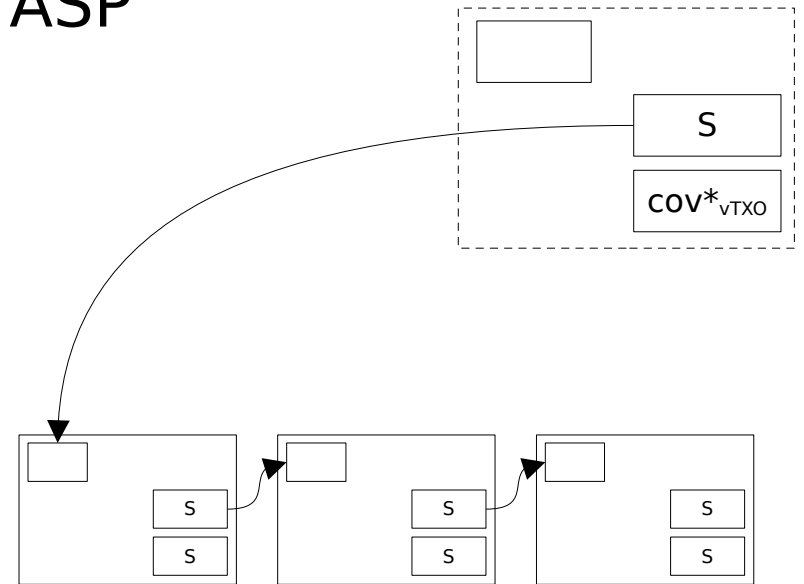
Connectors

- users only care about tx dep. chain
- simple 1-of-1 outputs owned by ASP
- ideally 0-value
 - rely on CPFP & package relay

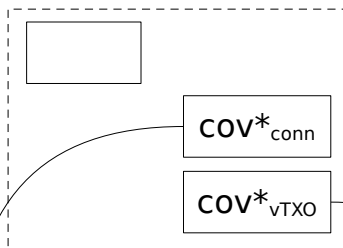
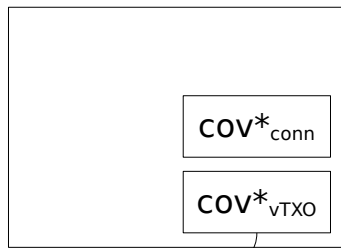


Connectors

- users only care about tx dep. chain
- simple 1-of-1 outputs owned by ASP
- ideally 0-value
- alternatively single chain
 - users sign multiple forfeit txs

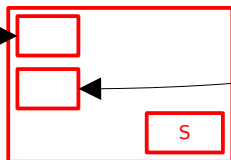
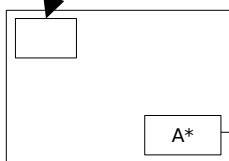
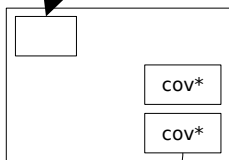
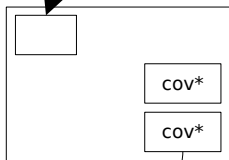


on-chain



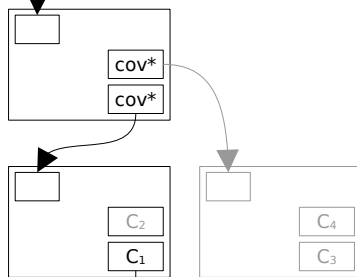
connector output
vTXOs output

off-chain

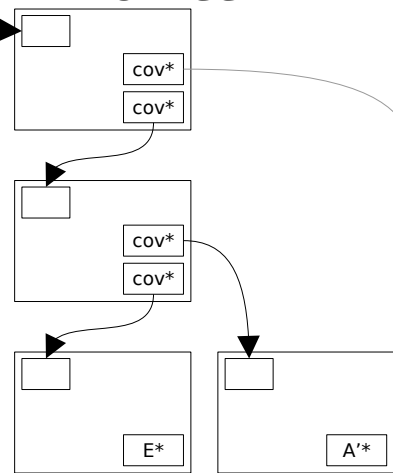


forfeit tx

connector tree



vTXO tree



S = ASP pubkey
 A^* = $A+S$ OR (A after 7d)
 COV^* = COV OR (S after 14d)

What is (an) Ark?

What is (an) Ark?

- series of “Ark rounds”

What is (an) Ark?

- series of “Ark rounds”
 - atomic spending vTXOs to create new ones

What is (an) Ark?

- series of “Ark rounds”
 - atomic spending vTXOs to create new ones
 - one on-chain Ark tx with vTXO & connector outputs

What is (an) Ark?

- series of “Ark rounds”
 - atomic spending vTXOs to create new ones
 - one on-chain Ark tx with vTXO & connector outputs
- single service provider: “ASP”

What is (an) Ark?

- series of “Ark rounds”
 - atomic spending vTXOs to create new ones
 - one on-chain Ark tx with vTXO & connector outputs
- single service provider: “ASP”
 - coordinates & provides liquidity*

What is (an) Ark?

- series of “Ark rounds”
 - atomic spending vTXOs to create new ones
 - one on-chain Ark tx with vTXO & connector outputs
- single service provider: “ASP”
 - coordinates & provides liquidity*
 - users always 100% in control of money

What is (an) Ark?

What is (an) Ark?

- efficient UTXO-style off-chain txs

What is (an) Ark?

- efficient UTXO-style off-chain txs
- only client-server interactions needed, no p2p

What is (an) Ark?

- efficient UTXO-style off-chain txs
- only client-server interactions needed, no p2p
- anyone can receive (no liquidity required)

What is (an) Ark?

- efficient UTXO-style off-chain txs
- only client-server interactions needed, no p2p
- anyone can receive (no liquidity required)
- flexible round times

What is (an) Ark?

- efficient UTXO-style off-chain txs
- only client-server interactions needed, no p2p
- anyone can receive (no liquidity required)
- flexible round times
 - confirmation when Ark tx confirms

What is (an) Ark?

What is (an) Ark?

- efficient UTXO-style off-chain txs
- only client-server interactions needed, no p2p
- anyone can receive (no liquidity required)
- flexible round times
- vTXO expiry

What is (an) Ark?

- efficient UTXO-style off-chain txs
- only client-server interactions needed, no p2p
- anyone can receive (no liquidity required)
- flexible round times
- vTXO expiry
 - watchtower-based automatic vTXO refresh?

Ark round flow

Ark round flow

- ASP announces start of new round

Ark round flow

- ASP announces start of new round
- spenders indicate input & output vTXOs

Ark round flow

- ASP announces start of new round
- spenders indicate input & output vTXOs
- ASP assembles an Ark tx and sends it to spenders

Ark round flow

- ASP announces start of new round
- spenders indicate input & output vTXOs
- ASP assembles an Ark tx and sends it to spenders
 - vTXO tree output

Ark round flow

- ASP announces start of new round
- spenders indicate input & output vTXOs
- ASP assembles an Ark tx and sends it to spenders
 - vTXO tree output
 - connector tree output

Ark round flow

- ASP announces start of new round
- spenders indicate input & output vTXOs
- ASP assembles an Ark tx and sends it to spenders
 - vTXO tree output
 - connector tree output
 - input liquidity & potential change output (can be a vTXO too!)

Ark round flow

- ASP announces start of new round
- spenders indicate input & output vTXOs
- ASP assembles an Ark tx and sends it to spenders
 - vTXO tree output
 - connector tree output
 - input liquidity & potential change output (can be a vTXO too!)
- spenders sign their forfeit txs

Ark round flow

- ASP announces start of new round
- spenders indicate input & output vTXOs
- ASP assembles an Ark tx and sends it to spenders
 - vTXO tree output
 - connector tree output
 - input liquidity & potential change output (can be a vTXO too!)
- spenders sign their forfeit txs
 - using individually assigned connector output

Ark round flow

- ASP announces start of new round
- spenders indicate input & output vTXOs
- ASP assembles an Ark tx and sends it to spenders
 - vTXO tree output
 - connector tree output
 - input liquidity & potential change output (can be a vTXO too!)
- spenders sign their forfeit txs
 - using individually assigned connector output
- outputs of spenders that refuse signing are dropped

Ark round flow

- ASP announces start of new round
- spenders indicate input & output vTXOs
- ASP assembles an Ark tx and sends it to spenders
 - vTXO tree output
 - connector tree output
 - input liquidity & potential change output (can be a vTXO too!)
- spenders sign their forfeit txs
 - using individually assigned connector output
- outputs of spenders that refuse signing are dropped
 - ASP creates new Ark tx and spenders sign again, etc..*

Let's dig a little deeper

Lifting

Lifting

- mechanism to enter and exit the Ark (boarding?)

Lifting

- mechanism to enter and exit the Ark (boarding?)
- non-interactive lift-in

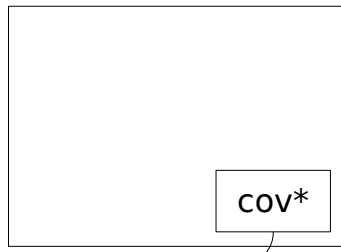
Lifting

- mechanism to enter and exit the Ark (boarding?)
- non-interactive lift-in
 - straight from on-chain to vTXO

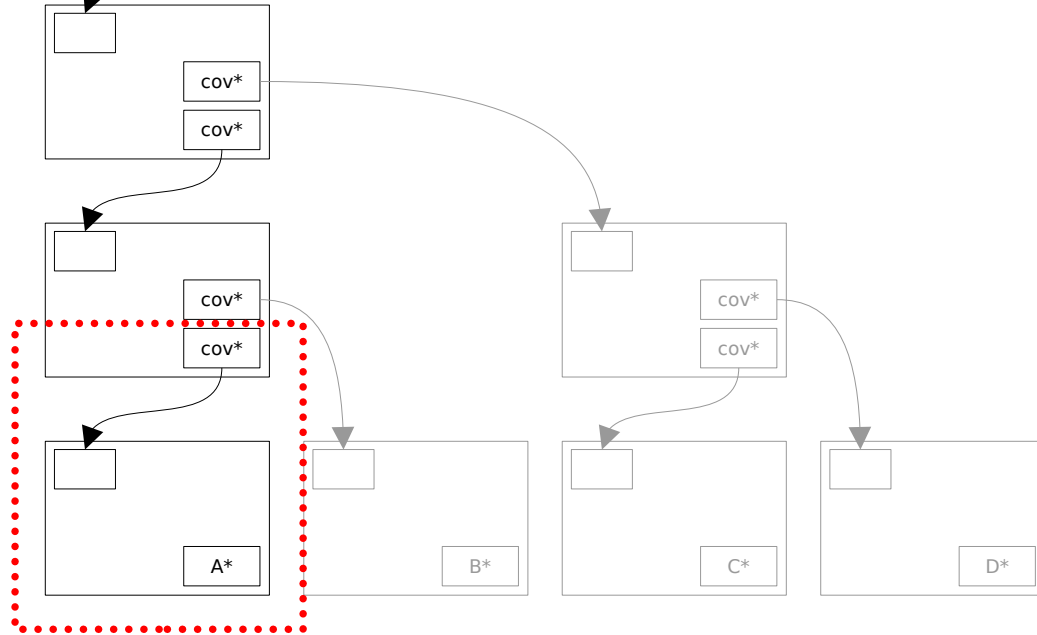
Lifting

- mechanism to enter and exit the Ark (boarding?)
- non-interactive lift-in
 - straight from on-chain to vTXO
- interactive lift-out

on-chain

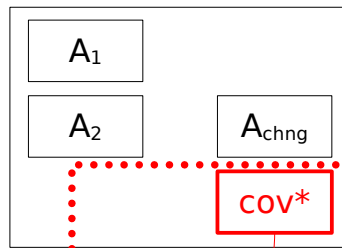
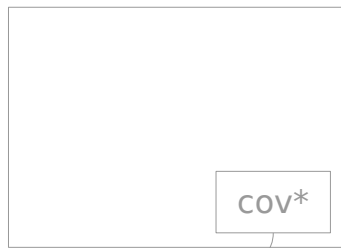


off-chain

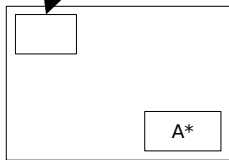
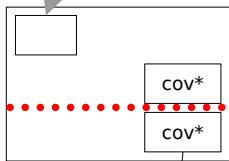
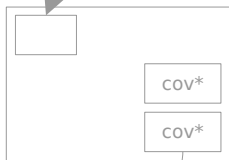


S = ASP pubkey
 A^* = $A+S$ OR (A after 7d)
 cov^* = cov OR (S after 14d)

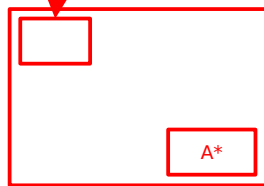
on-chain



off-chain

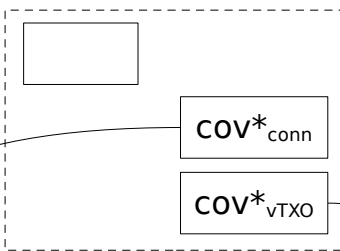
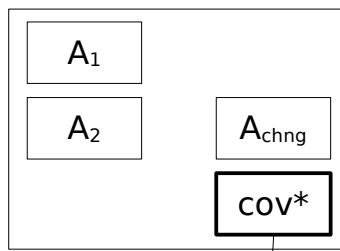


vTXO



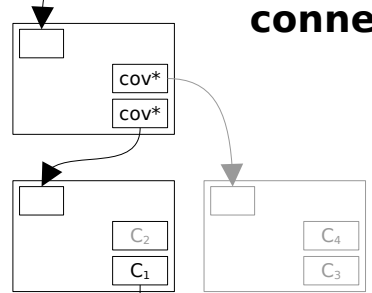
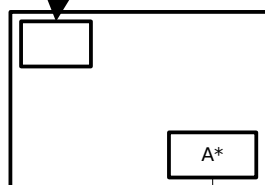
S = ASP pubkey
 A^* = $A+S$ OR (A after 7d)
 cov^* = cov OR (S after 14d)

on-chain



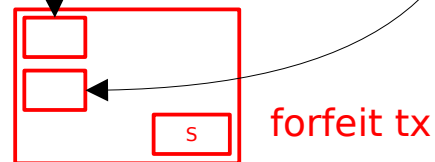
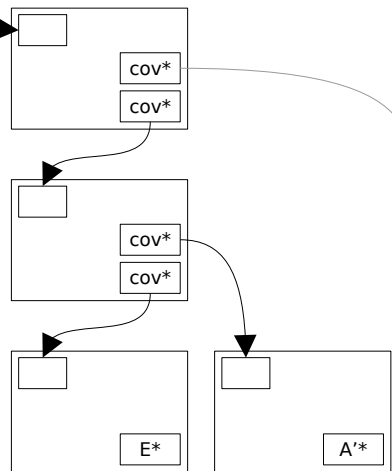
connector output
vTXOs output

off-chain



connector tree

vTXO tree



forfeit tx

$S = \text{ASP pubkey}$
 $A^* = A+S \text{ OR } (A \text{ after } 7d)$
 $\text{cov}^* = \text{cov} \text{ OR } (S \text{ after } 14d)$

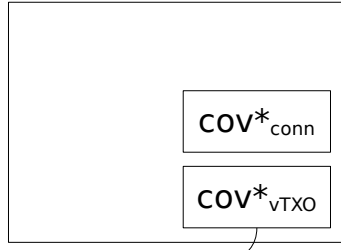
Lifting

- mechanism to enter and exit the Ark
- non-interactive lift-in
 - straight from on-chain to vTXO
- interactive lift-out

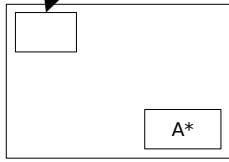
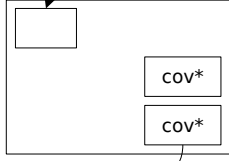
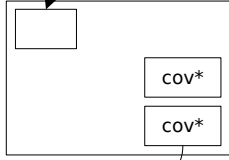
Lifting

- mechanism to enter and exit the Ark
- non-interactive lift-in
 - straight from on-chain to vTXO
- interactive lift-out
 - non-interactive unilateral exit always available

on-chain

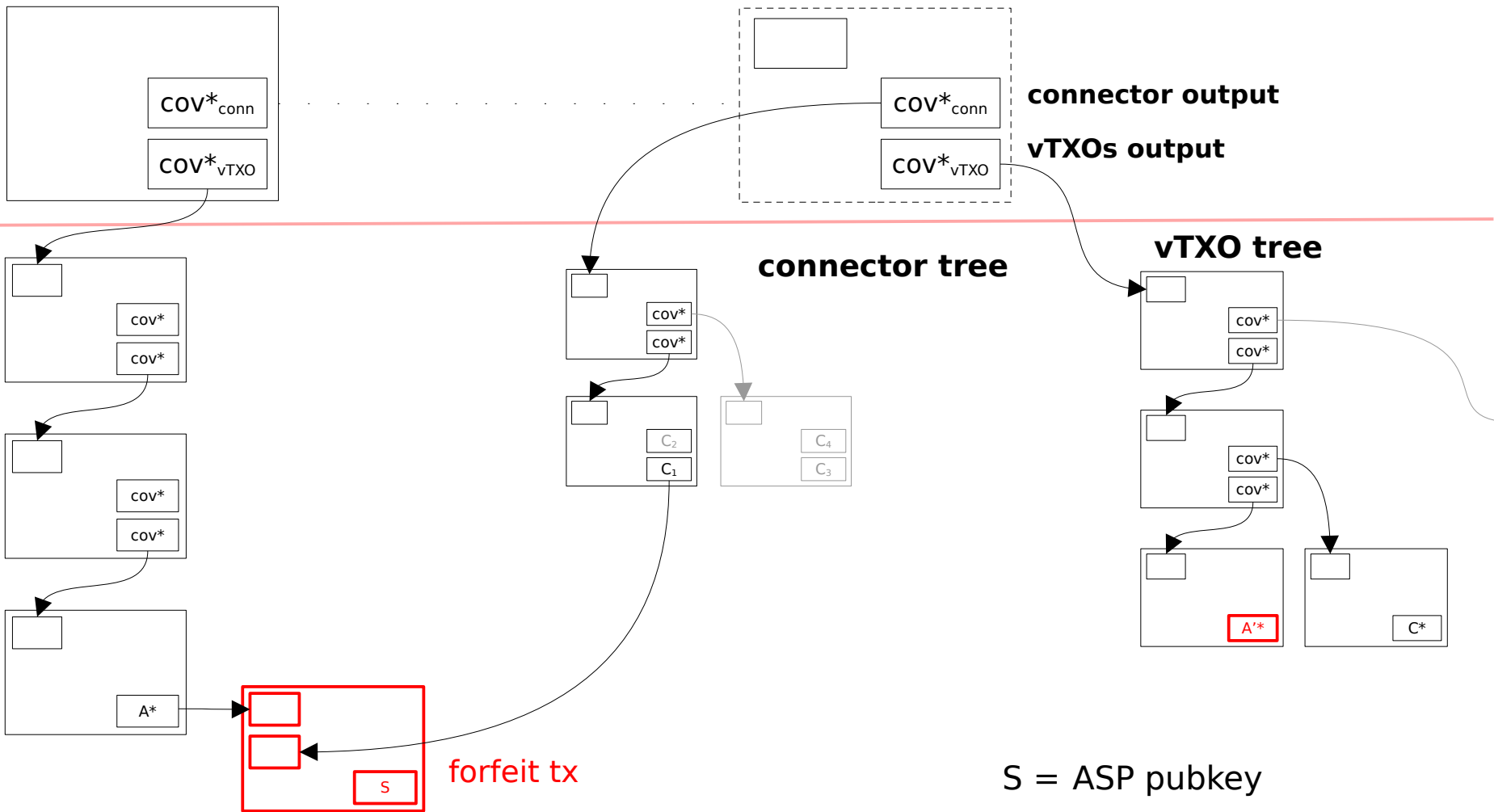


off-chain



on-chain

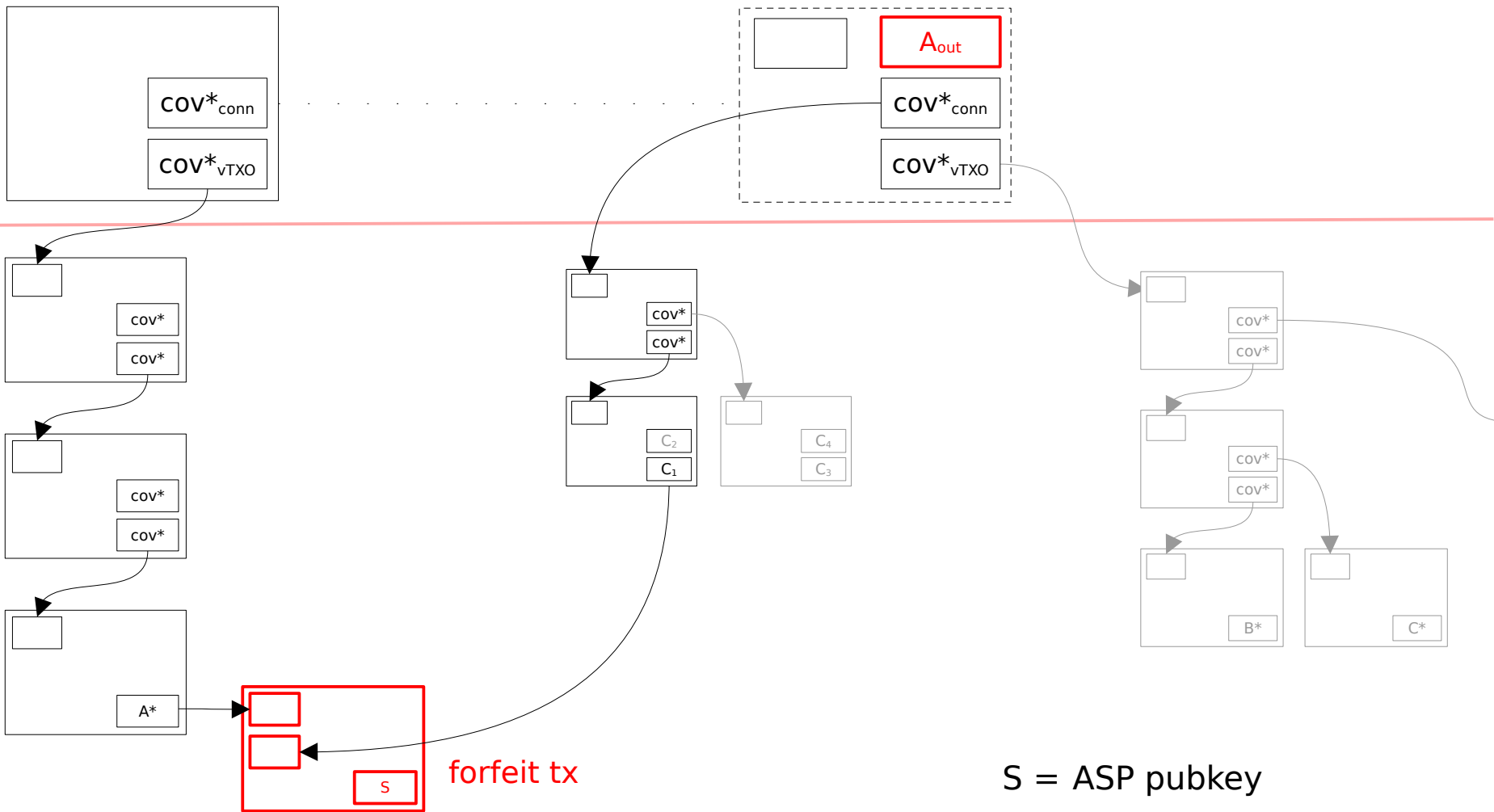
off-chain



S = ASP pubkey
 A^* = $A+S$ OR (A after 7d)
 cov^* = cov OR (S after 14d)

on-chain

off-chain



S = ASP pubkey
 A^* = $A+S$ OR (A after 7d)
 cov^* = cov OR (S after 14d)

Note on covenants

Note on covenants

- covenant construction desired

Note on covenants

- covenant construction desired
 - OP_CHECKTEMPLATEVERIFY (CTV)

Note on covenants

- covenant construction desired
 - OP_CHECKTEMPLATEVERIFY (CTV)
 - OP_TXHASH / OP_TX

Note on covenants

- covenant construction desired
 - OP_CHECKTEMPLATEVERIFY (CTV)
 - OP_TXHASH / OP_TX
 - OP_CHECKSIGFROMSTACK (on Liquid)

Note on covenants

- covenant construction desired
 - OP_CHECKTEMPLATEVERIFY (CTV)
 - OP_TXHASH / OP_TX
 - OP_CHECKSIGFROMSTACK (on Liquid)
 - SIGHASH_ANYPREVOUT* (aka SIGHASH_NOINPUT)
- possible on Inquisition testnet or Liquid right now

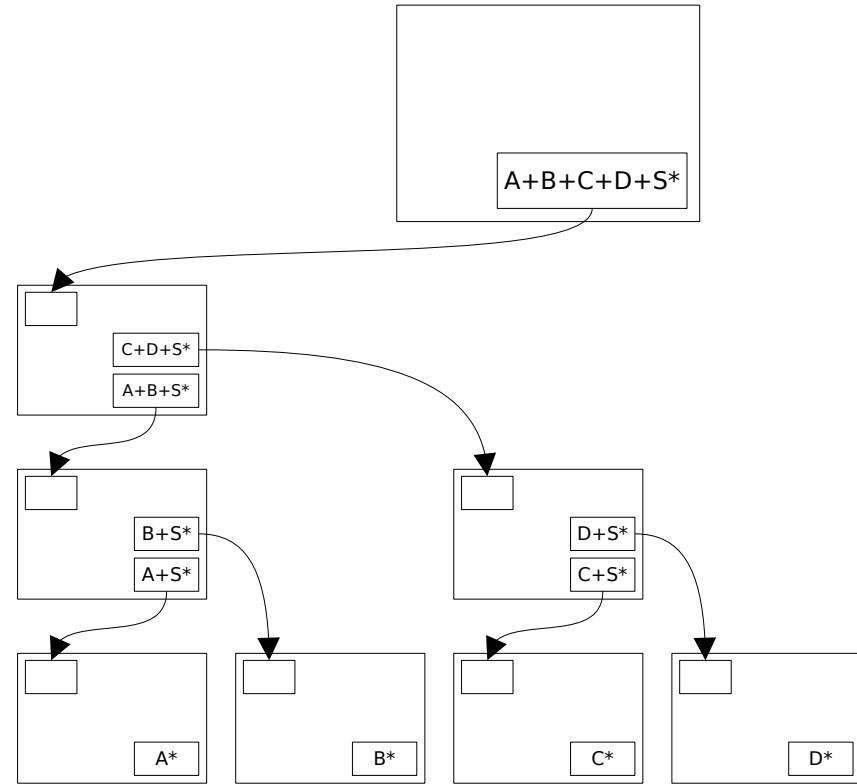
Without covenants: clArk

Without covenants: clArk

- use multisigs instead of covenants

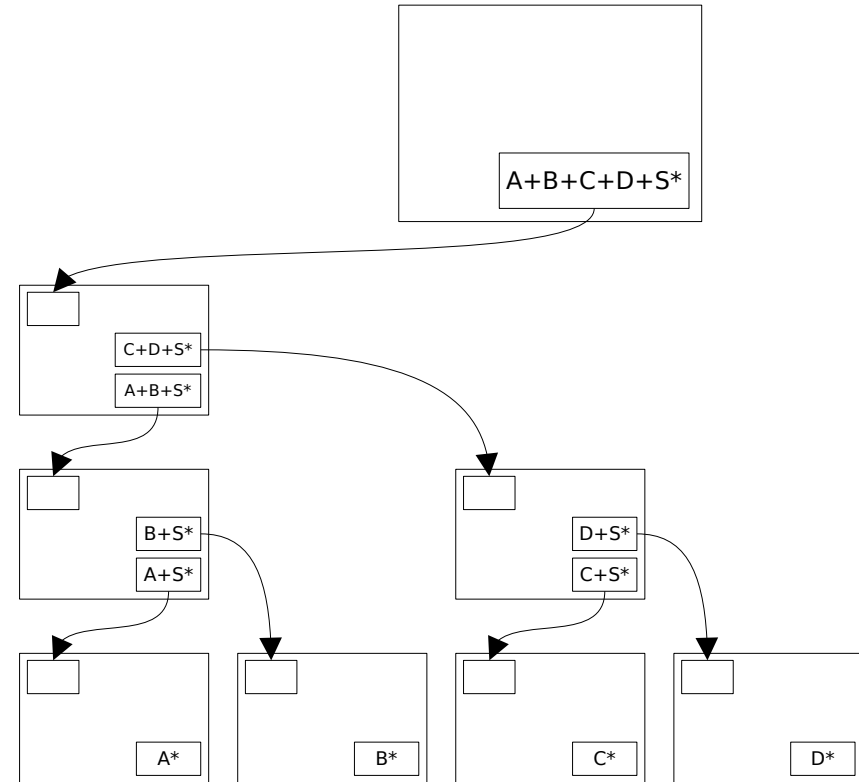
Without covenants: clArk

$S = \text{ASP pubkey}$
 $A+B+S^* = A+B+S \text{ OR } (S \text{ after } 14\text{d})$
 $A^* = A+S \text{ OR } (A \text{ after } 7\text{d})$



Without covenants: clArk

- use multisigs instead of covenants



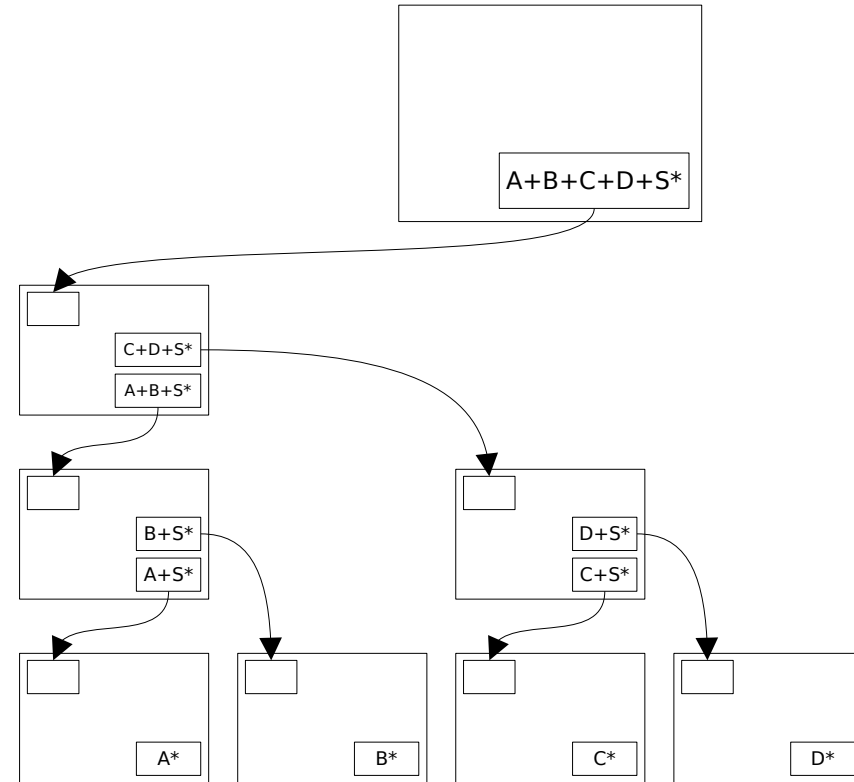
S = ASP pubkey

$A+B+S^* = A+B+S$ OR (S after 14d)

$A^* = A+S$ OR (A after 7d)

Without covenants: clArk

- use multisigs instead of covenants
 - all receivers cosign with ASP



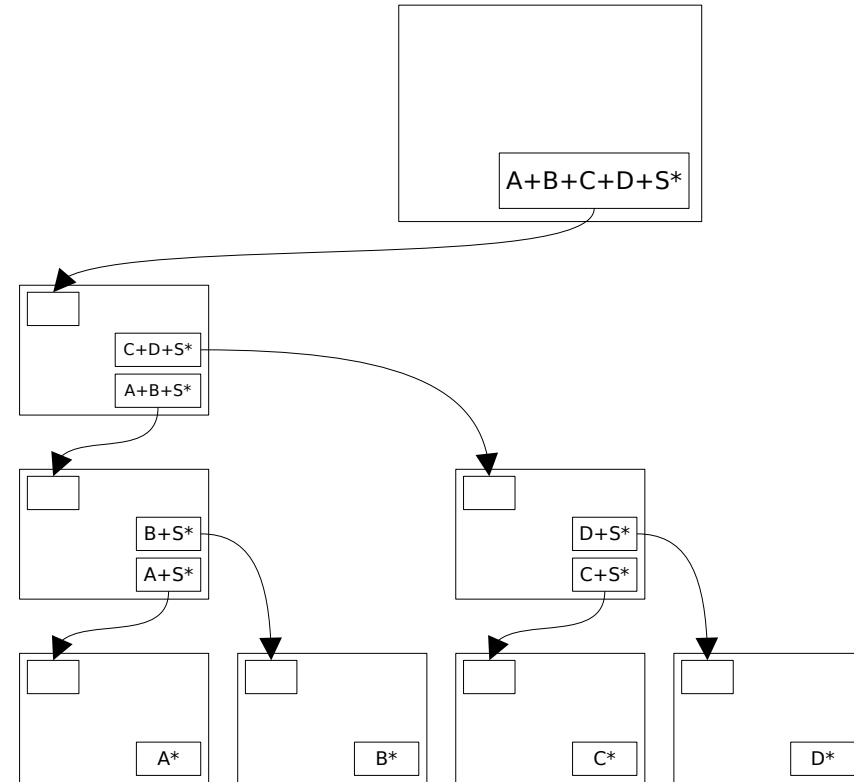
S = ASP pubkey

$A+B+S^* = A+B+S$ OR (S after 14d)

$A^* = A+S$ OR (A after 7d)

Without covenants: clArk

- use multisigs instead of covenants
 - all receivers cosign with ASP
 - requires receivers online



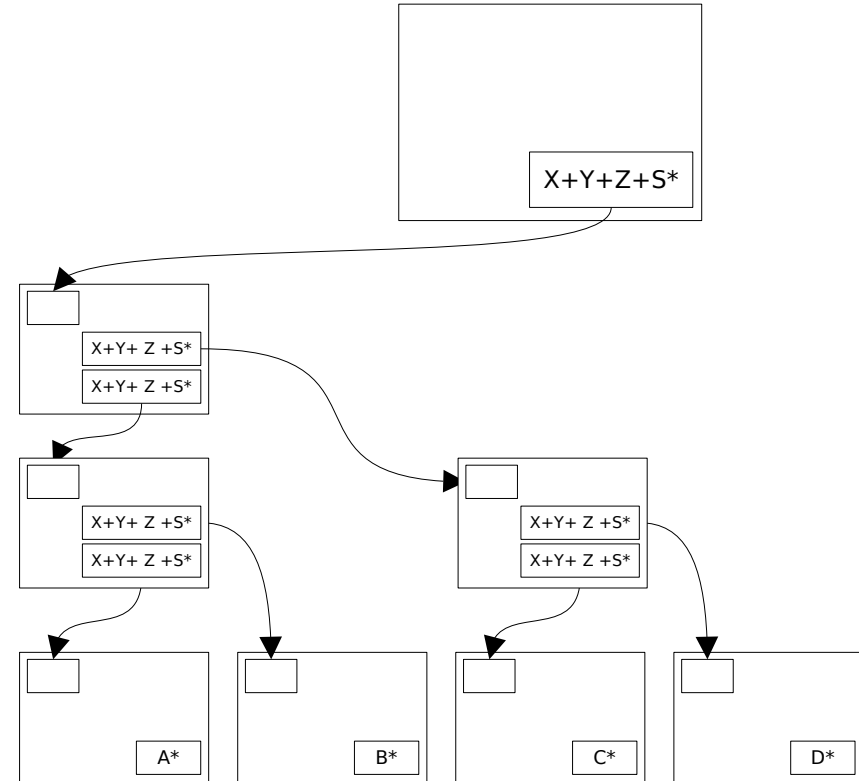
S = ASP pubkey

$A+B+S^* = A+B+S$ OR (S after 14d)

$A^* = A+S$ OR (A after 7d)

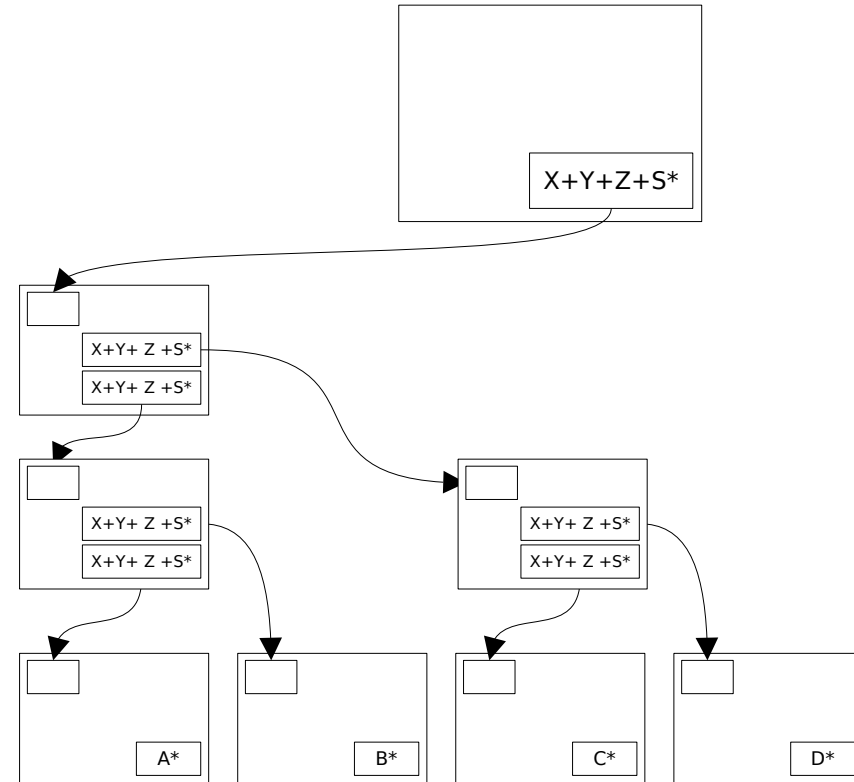
Without covenants: clArk

$S = \text{ASP pubkey}$
 $A+B+S^* = A+B+S \text{ OR } (S \text{ after } 14\text{d})$
 $A^* = A+S \text{ OR } (A \text{ after } 7\text{d})$



Without covenants: clArk

- use multisigs instead of covenants
 - all receivers cosign with ASP
 - requires receivers online
 - (how about all senders?)



S = ASP pubkey

$A+B+S^* = A+B+S$ OR (S after 14d)

$A^* = A+S$ OR (A after 7d)

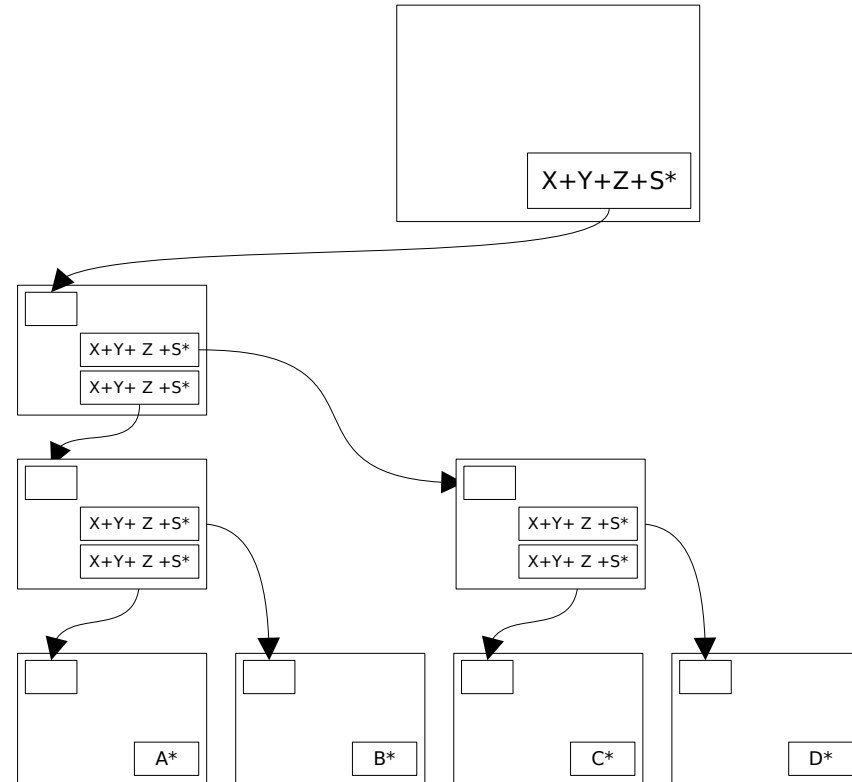
Without covenants: clArk

- use multisigs instead of covenants
 - all receivers cosign with ASP
 - requires receivers online
 - (how about all senders?)
- possible on Bitcoin today

$S = \text{ASP pubkey}$

$A+B+S^* = A+B+S \text{ OR } (S \text{ after } 14\text{d})$

$A^* = A+S \text{ OR } (A \text{ after } 7\text{d})$

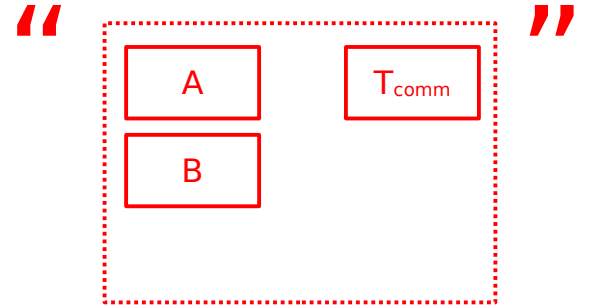


You said Lightning?

You said Lightning?

- Lightning channel commitment vTXOs

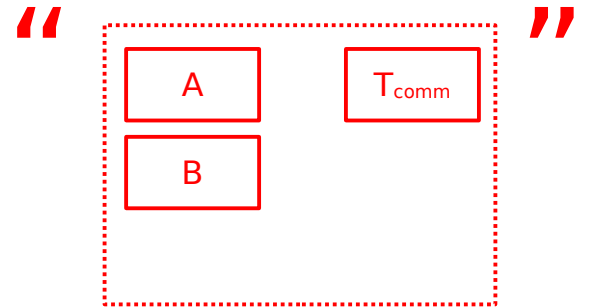
$$T_{\text{comm}} = A+B$$



You said Lightning?

- Lightning channel commitment vTXOs
 - Ark as “channel factory”

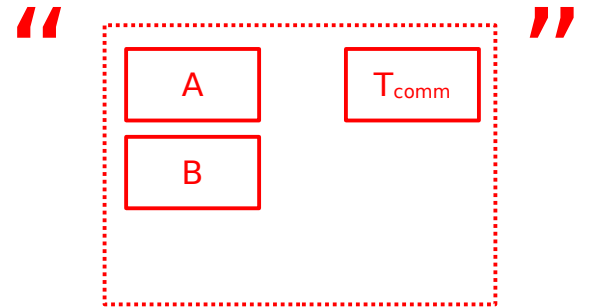
$$T_{\text{comm}} = A+B$$



You said Lightning?

- Lightning channel commitment vTXOs
 - Ark as “channel factory”
 - requires periodic channel refresh

$$T_{\text{comm}} = A+B$$



You said Lightning?

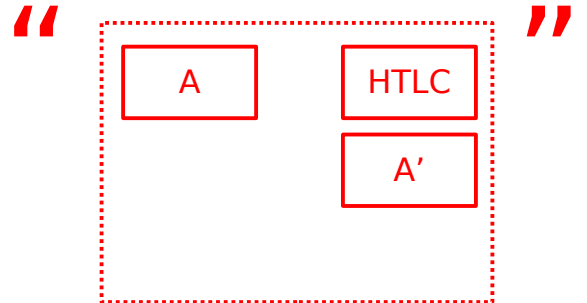
You said Lightning?

- Lightning channel commitment vTXOs

You said Lightning?

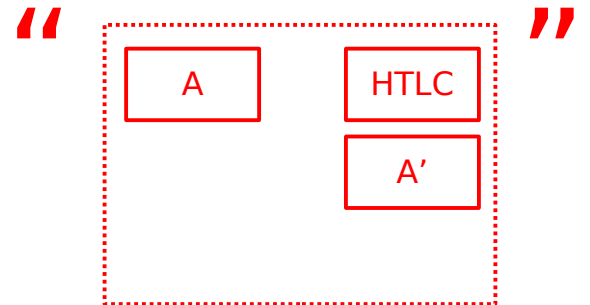
- Lightning channel commitment vTXOs
- HTLCs can be added directly as vTXOs

HTLC = (S AND preimage) OR (A after 24h)



You said Lightning?

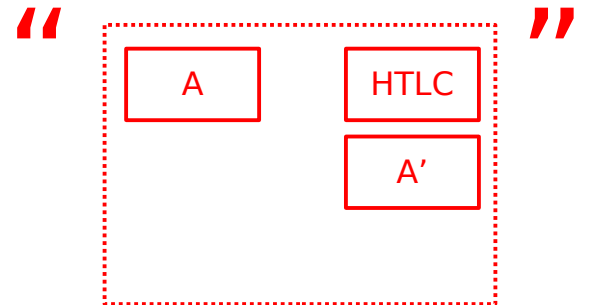
- Lightning channel commitment vTXOs
- HTLCs can be added directly as vTXOs
 - Lightning payment in Ark round



HTLC = (S AND preimage) OR (A after 24h)

You said Lightning?

- Lightning channel commitment vTXOs
- HTLCs can be added directly as vTXOs
 - Lightning payment in Ark round
 - ASP acts as LSP and forwards payment



HTLC = (S AND preimage) OR (A after 24h)

Addressing

Addressing

- $A^* = A + S \text{ OR } 0R$ (A after 7d)

Addressing

- $A^* = A+S$ OR (A after 7d)
- in theory many Miniscript policies seem possible
→ $\text{or}(\text{and}(P, \text{key}(S)), \text{after}(7d, P))$

Addressing

- $A^* = A+S$ OR (A after 7d)
- in theory many Miniscript policies seem possible
→ $\text{or}(\text{and}(P, \text{key}(S)), \text{after}(7d, P))$
- ideally a single Schnorr (FROST?) pubkey
→ optimal for taproot keyspend

Addressing

- $A^* = A+S \text{ OR } (A \text{ after } 7d)$
- in theory many Miniscript policies seem possible
→ $\text{or}(\text{and}(P, \text{key}(S)), \text{after}(7d, P))$
- ideally a single Schnorr (FROST?) pubkey
→ optimal for taproot keyspend
- receiver gives a pubkey/policy to sender

Privacy

Privacy

- currently the ASP has full insight in txs

Privacy

- currently the ASP has full insight in txs
- solution: blinded coinjoins “à la WabiSabi”

Privacy

- currently the ASP has full insight in txs
- solution: blinded coinjoins “à la WabiSabi”
 - spenders get blinded tokens for input vTXOs

Privacy

- currently the ASP has full insight in txs
- solution: blinded coinjoins “à la WabiSabi”
 - spenders get blinded tokens for input vTXOs
 - redeem blinded tokens for output vTXOs

Privacy

- currently the ASP has full insight in txs
- solution: blinded coinjoins “à la WabiSabi”
 - spenders get blinded tokens for input vTXOs
 - redeem blinded tokens for output vTXOs
 - fixed denominations for vTXO values

But wait...

But wait...

- vTXO output pubkey reuse is a problem

But wait...

- vTXO output pubkey reuse is a problem
 - ASP can deanonymise receivers & targetted senders

But wait...

- vTXO output pubkey reuse is a problem
 - ASP can deanonymise receivers & targetted senders
- need new pubkey each vTXO & round attempt

But wait...

- vTXO output pubkey reuse is a problem
 - ASP can deanonymise receivers & targetted senders
- need new pubkey each vTXO & round attempt
 - paynym/greenaddress-like solution

But wait...

- vTXO output pubkey reuse is a problem
 - ASP can deanonymise receivers & targetted senders
- need new pubkey each vTXO & round attempt
 - paynym/greenaddress-like solution
 - out-of-band from sender to receiver? nostr?

But wait...

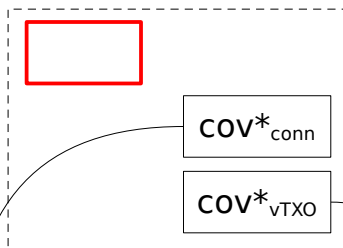
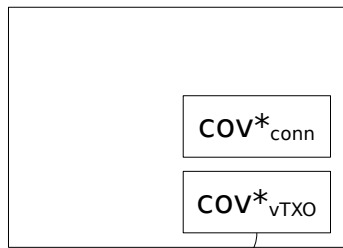
- vTXO output pubkey reuse is a problem
 - ASP can deanonymise receivers & targetted senders
- need new pubkey each vTXO & round attempt
 - paynym/greenaddress-like solution
 - out-of-band from sender to receiver? nostr?
 - using deterministic round-specific entropy?

Existing challenges

Existing challenges

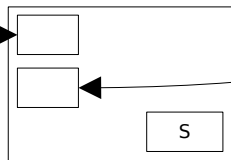
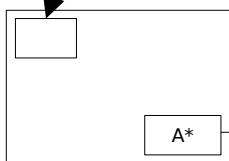
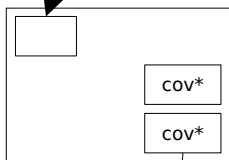
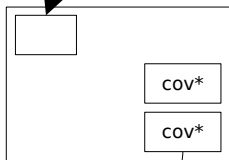
- ASP can double spend txs in mempool

on-chain



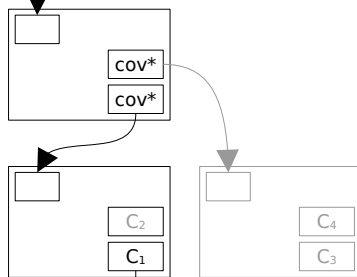
connector output
vTXOs output

off-chain

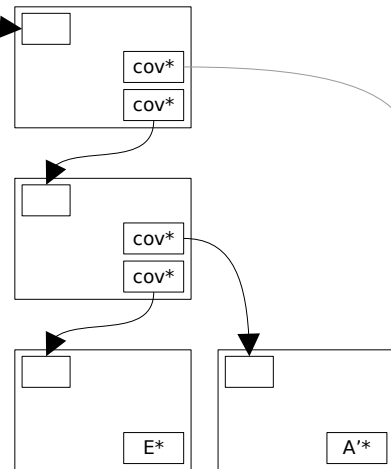


forfeit tx

connector tree



vTXO tree



S = ASP pubkey
 A^* = $A+S$ OR (A after 7d)
 cov^* = cov OR (S after 14d)

Existing challenges

Existing challenges

- ASP can double spend txs in mempool

Existing challenges

- ASP can double spend txs in mempool
 - no inherent incentive

Existing challenges

- ASP can double spend txs in mempool
 - no inherent incentive
 - disincentive because of HTLCs

Existing challenges

- ASP can double spend txs in mempool
 - no inherent incentive
 - disincentive because of HTLCs
 - ✓ LN-on-Ark txs don't care about confirmations

Existing challenges

- ASP can double spend txs in mempool
 - no inherent incentive
 - disincentive because of HTLCs
 - ✓ LN-on-Ark txs don't care about confirmations
 - double-spend prevention with bond?

Existing challenges

Existing challenges

- ASP can double spend txs in mempool
- high liquidity requirement

Existing challenges

- ASP can double spend txs in mempool
- high liquidity requirement
 - increases with vTXO velocity

Existing challenges

- ASP can double spend txs in mempool
- high liquidity requirement
 - increases with vTXO velocity
 - depends on vTXO expiration parameter

Existing challenges

- ASP can double spend txs in mempool
- high liquidity requirement
 - increases with vTXO velocity
 - depends on vTXO expiration parameter
 - ASP can charge fees based on vTXO age

Existing challenges

Existing challenges

- ASP can double spend txs in mempool
- high liquidity requirement
- DoS by forcing many round restarts

Existing challenges

- ASP can double spend txs in mempool
- high liquidity requirement
- DoS by forcing many round restarts
 - penalties for abandoning a round

Existing challenges

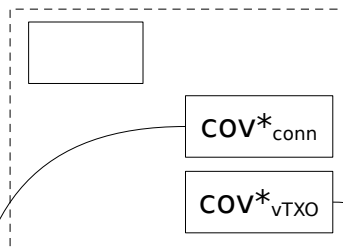
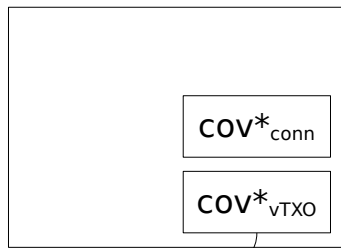
- ASP can double spend txs in mempool
- high liquidity requirement
- DoS by forcing many round restarts
 - penalties for abandoning a round
 - attack incentive is small with larger round times

NEW: the Somsen Shortcut

NEW: the Somsen Shortcut

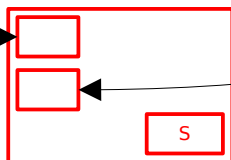
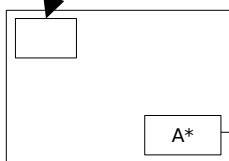
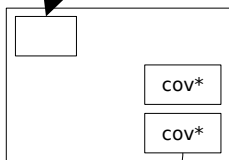
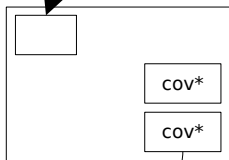
- send vTXOs outside Ark round

on-chain



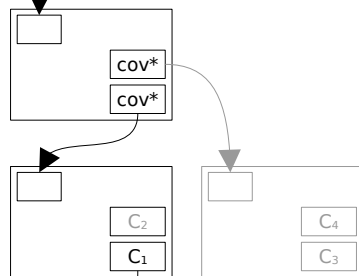
connector output
vTXOs output

off-chain

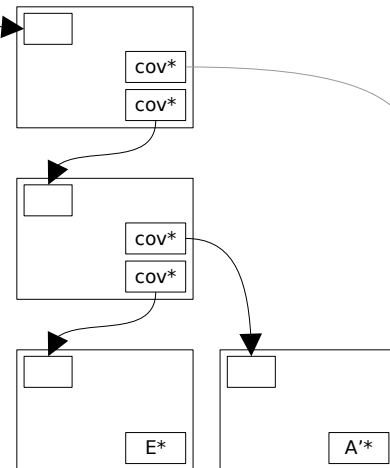


forfeit tx

connector tree

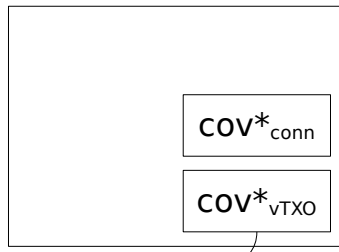


vTXO tree

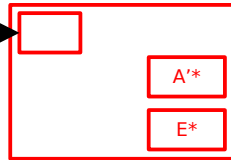
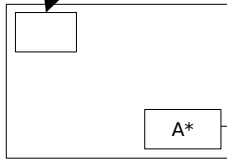
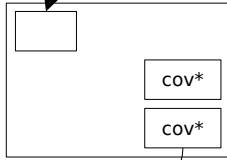
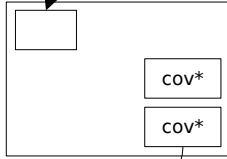


S = ASP pubkey
 A^* = $A+S$ OR (A after 7d)
 cov^* = cov OR (S after 14d)

on-chain



off-chain



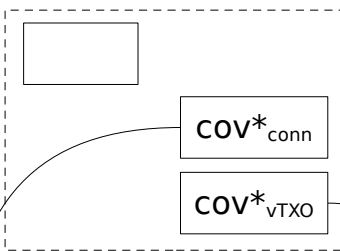
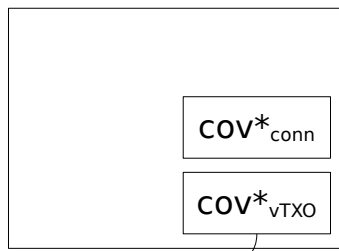
shortcut tx

$S = \text{ASP pubkey}$
 $A^* = A + S \text{ OR } (A \text{ after } 7d)$
 $cov^* = cov \text{ OR } (S \text{ after } 14d)$

NEW: the Somsen Shortcut

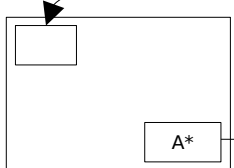
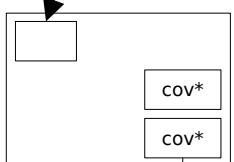
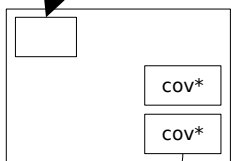
- send vTXOs outside Ark round
 - “building a state-chain from a vTXO”

on-chain

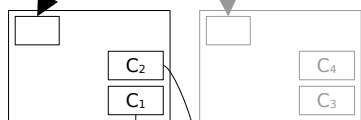
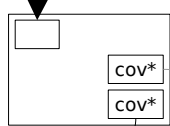


connector output
vTXOs output

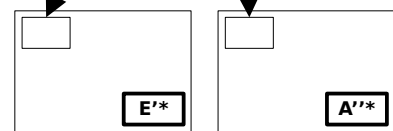
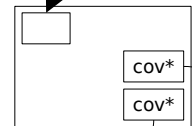
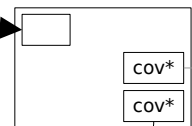
off-chain



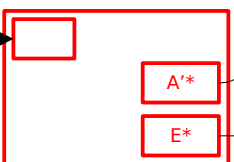
connector tree



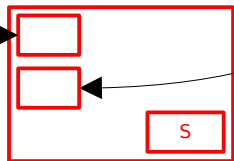
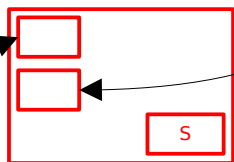
vTXO tree



shortcut tx



forfeit txs



$S = ASP \text{ pubkey}$
 $A^* = A+S \text{ OR } (A \text{ after } 7d)$
 $cov^* = cov \text{ OR } (S \text{ after } 14d)$

NEW: the Somsen Shortcut

NEW: the Somsen Shortcut

- send vTXOs outside Ark round
 - “building a state-chain from a vTXO”

NEW: the Somsen Shortcut

- send vTXOs outside Ark round
 - “building a state-chain from a vTXO”
- makes clArk more feasible

Thanks

Thanks

- <https://roose.io/presentations>

Thanks

- <https://roose.io/presentations>
- <https://arkpill.me/>

Thanks

- <https://roose.io/presentations>
- <https://arkpill.me/>
- Questions?