

Blockstream

Building the blockchain ecosystem

Blockchain Techlab

2016-04-11

Jonas Nick

jonas@blockstream.com

Part 1: Blockchain

Peer-to-Peer Cash

- Ideal: Internet money without central control and anonymous

I've been working on a new electronic cash system that's fully peer-to-peer, with no trusted third party.

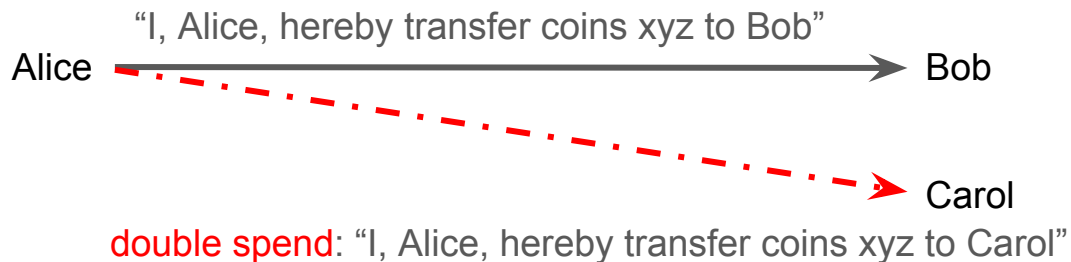
[...]

Satoshi Nakamoto

The Cryptography Mailing List

A toy currency

- Start with arbitrary bits that you call coins from now on
- Use cryptographic signatures to make forging messages impossible



- A central bank could tell which transaction came first.

A toy currency

- Decentralize control: Shared ledger
 - Every participant keeps a record of the transaction history
 - This works as long you know all the participants and trust a majority.
- But in open peer-to-peer systems
 - It is impossible to know all the participants.
 - It is impossible to meaningfully count votes.
- Want: dynamic membership of the participant set

Bitcoin

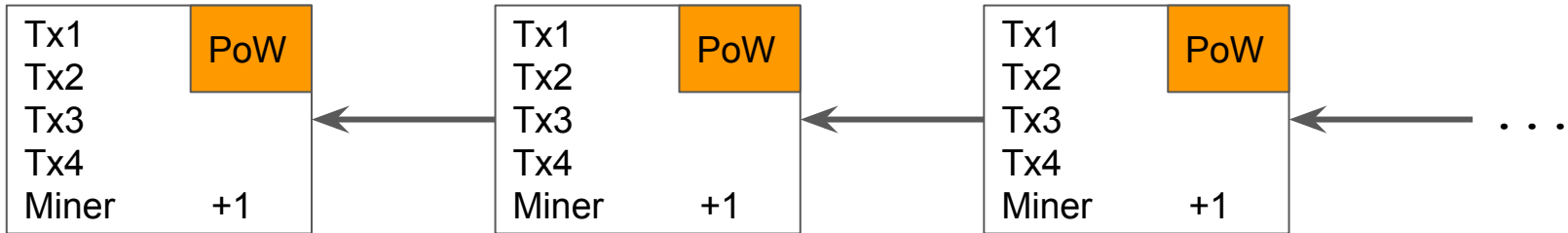
- Proof of Work: small proof that some amount of computation was done
- 1. Define that the “official” transaction history
 - a. is valid
 - b. has the most proof of work
- 2. Providing PoW (mining) to the official history is rewarded with coins

Effect:

1. Consensus on official history.
2. Incentivizes mining on a history. Incentivizes mining on the official history.

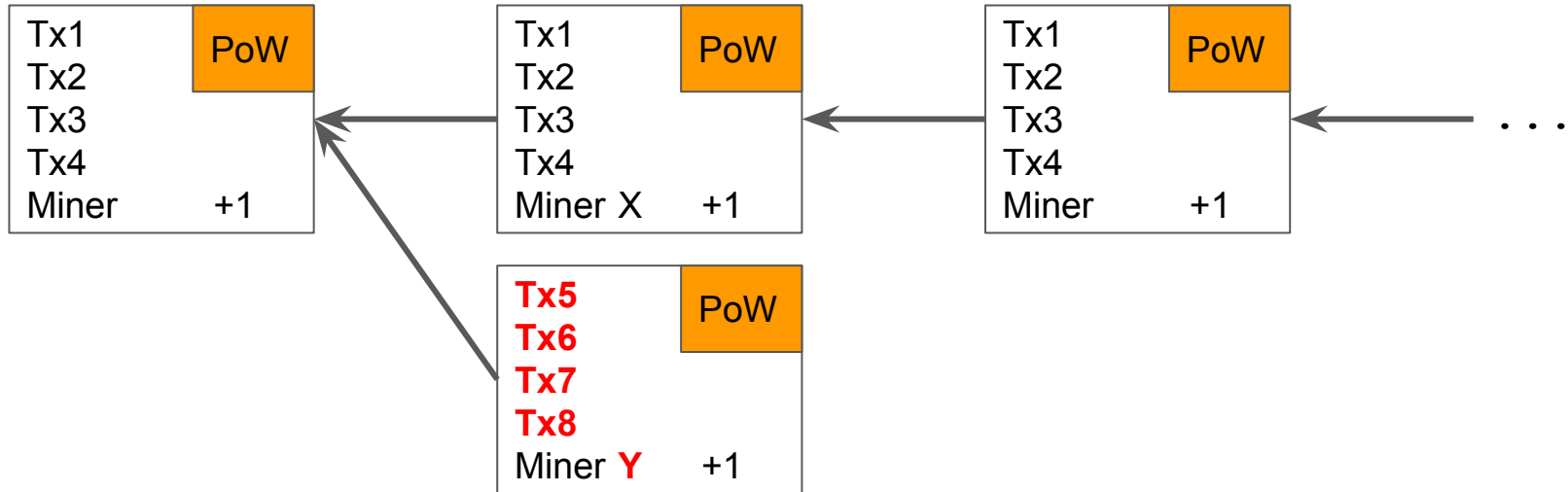
Mining

- History is represented as a chain of blocks.
 - Blocks contain transactions.
- Miners create blocks by collecting transactions.
- And attempt to solve the PoW function.
- Blocks are mined on expectancy every 10 minutes.
- The miner gets a mining reward.



Mining

- Miner attempting to rewrite the history always loses in the long run
 - As long as miner has less than 50% hash rate
- Miners can *not* spend your coins or include invalid transactions
 - f.e. A tx that send more coins than the attacker has available.



Blockchain technology

- 2 years ago: An application that uses Bitcoin in some way
- Now: **Consensus** on shared **state** with **immutable** rules in a **distributed** environment with potentially **dishonest** nodes.
- Goal: Reduce trust or expensive processes
- Can enable interactions that were previously impossible.

Workshop setup

- You start the alpha daemon
 - This is our Bitcoin fork that adds some features.
 - If not otherwise stated all functions we are going to look at apply to Bitcoin as well.
- You communicate to alphad with alpha-cli
- alphad connects you to a private altcoin network
- I mine blocks and I have all the coins.
- Further instructions: <https://jonasnick.github.io/workshop/tutorial-part-1.html>

Part 2: Transactions

Transactions

- Balance-based vs. UTXOs
- Balance-based (f.e. Ethereum)

Ledger state

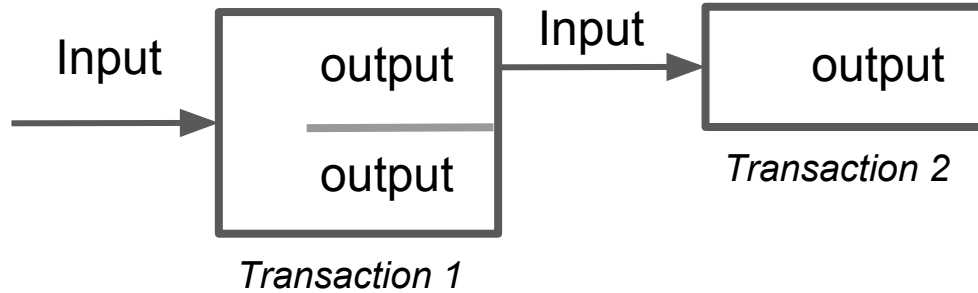
Alice	2
Bob	0

- Transaction: Alice 1 coin \longrightarrow Bob

New ledger state

Alice	1
Bob	1

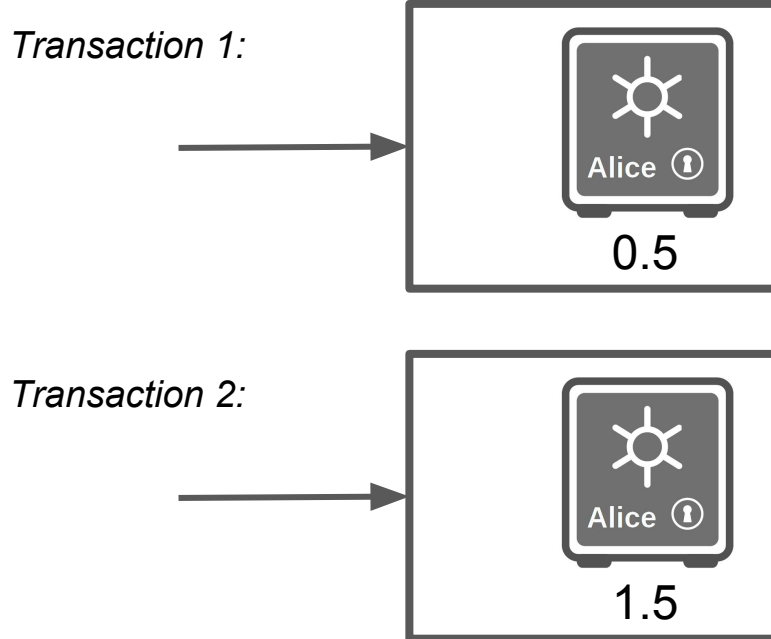
Transactions Inputs & Outputs



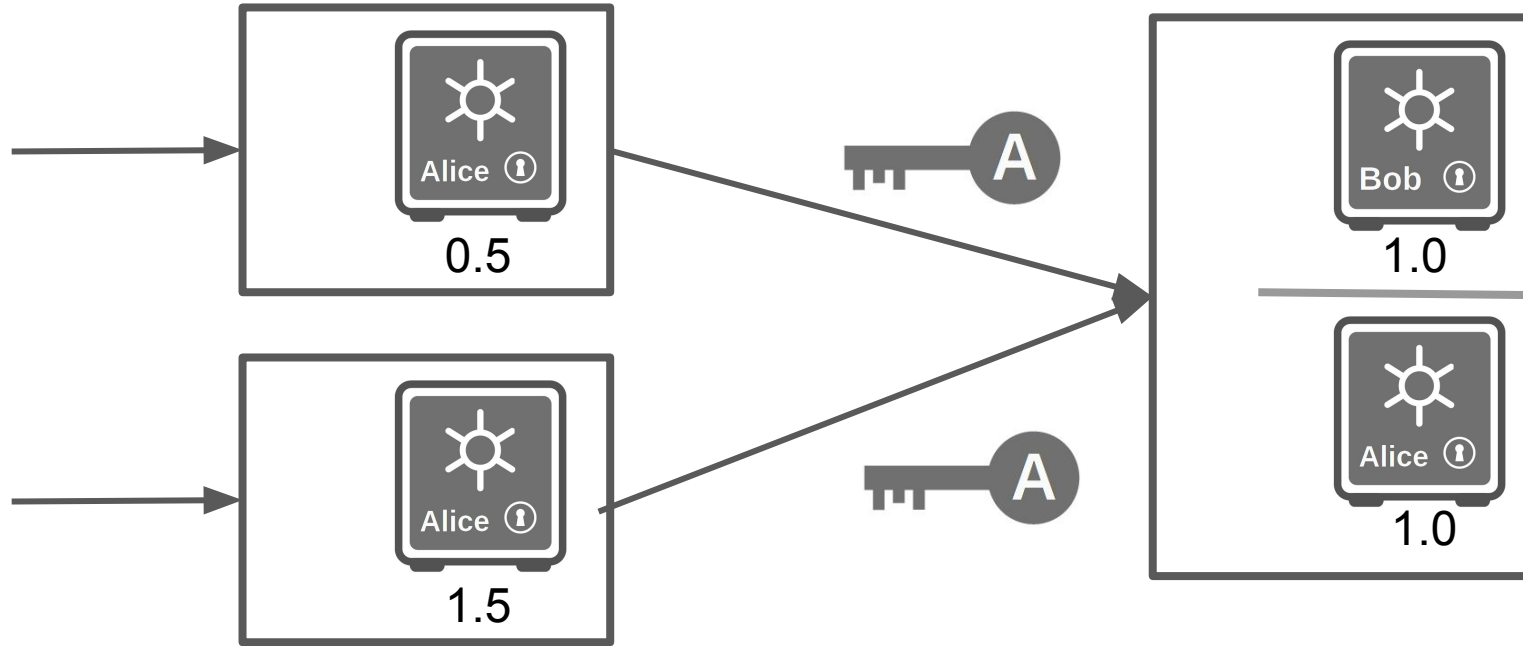
Transaction output: tuple of recipient and value
 input: tuple of txid, vout and signature

Unspent Transaction Outputs (UTXOs)

- Alice owns 2 coins = Alice can spend transaction outputs whose values sum to 2

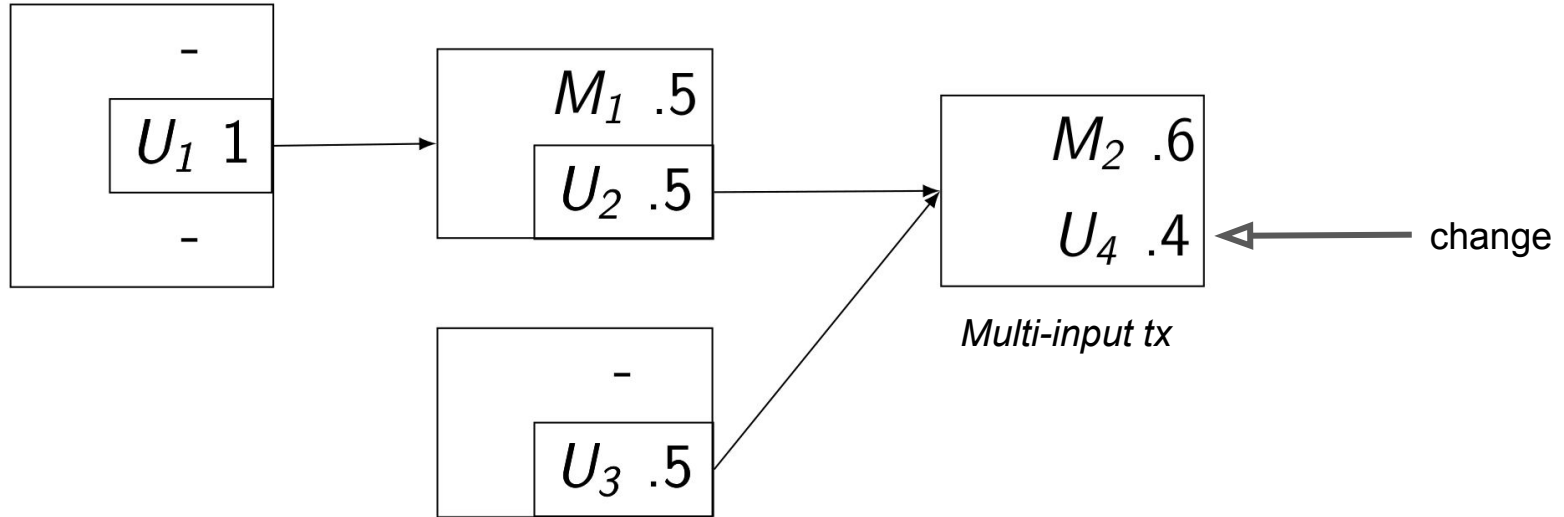


Spending Outputs



A Transaction Graph

User U with addresses U_i , Merchant M with addresses M_i



getrawtransaction <txid>

```
{
  "fee" : <fee>
  "vin" : [{
    "txid" : "<txid>",
    "vout" : <output_index>,
    "scriptSig" : {
      "asm" : "<scriptSig>",
    }
  }]
  "vout" : [{
    "value" <value>,
    "scriptPubKey" : {
      "asm" : "<scriptPubKey>",
    }
  }]
}
```

Elements feature: Confidential Transactions (CT)

Tx verification: $\text{input_value} = \text{output_value} + \text{fee}$

Verification with CT: $\text{Enc}(\text{input_value}) = \text{Enc}(\text{output_value}) + \text{fee}$

$\text{elements_address} = \text{bitcoin_address} + \text{blinding_pubKey}$

Without corresponding blinding private key, values are hidden (blinded).

Auditors can import private blinding key

-> Exercise <https://jonasnick.github.io/workshop/tutorial-part-2.html>

Part 3: Script

Cryptography Basics

- Cryptographic hash functions
 - `hash: {0,1}* -> {0,1}^n`
 - **Example:** `sha1("foo") =`
`f1d2d2f924e986ac86fdf7b36c94bcd32beec15`
 - collision resistant
- Public key cryptography
 - key pair: secret key `sk` and public key `pk`
 - cryptographic signature over message `m`
 - `sign(message, sk) -> sig`
 - `verify(message, pk, sig) -> {0, 1}`
 - Nobody can create a `sig` for a `pk` without the `sk`.

Script Evaluation: Pay-to-pubkey (P2PK)



= Bitcoin script `<pubKey> OP_CHECKSIG`



= Bitcoin script `<sig>`



= `<sig> <pubKey> OP_CHECKSIG`

= `true`

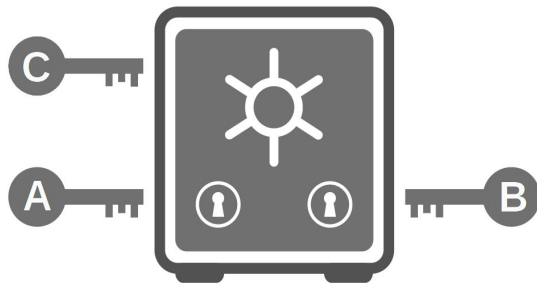
Pay-to-pubkey-hash (P2PKH)

scriptPubKey: `OP_DUP OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG`

scriptSig: `<sig> <pubKey>`

Stack	Script
	<code><sig> <pubKey> OP_DUP OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG</code>
<code><sig> <pubKey></code>	<code>OP_DUP OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG</code>
<code><sig> <pubKey> <pubKeyHash></code>	<code><pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG</code>
<code><sig> <pubKey></code>	<code>OP_CHECKSIG</code>

Multisig



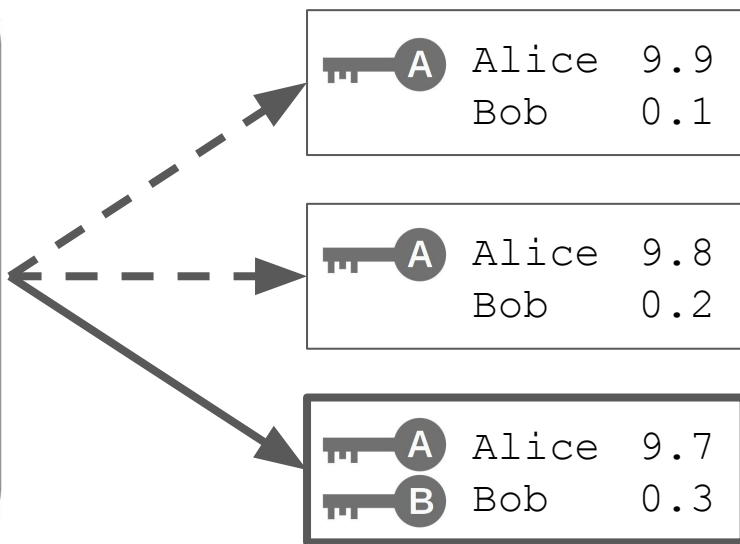
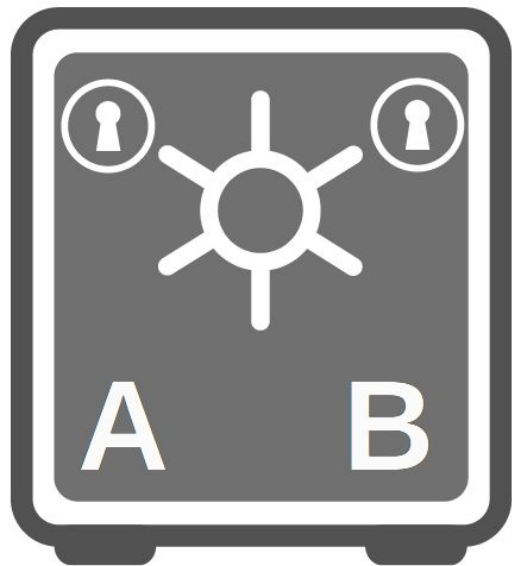
2 of 3 Multisig Output

Use cases: Wallet security, Escrow, Micropayment Channels

```
scriptPubKey: <m> <pubKey_1> ... <pubKey_n> <n> OP_CHECKMULTISIG  
scriptSig:    <sig_1> ... <sig_m>
```

Micropayment Channels

Setup: Alice creates transaction with 10 bitcoin to a 2-of-2 multisig with Bob



Alice pays by signing tx
and sending it directly to Bob

Closing the Channel:
Bob signs tx and broadcast
to miners

Micropayment Channel

- Problem: If Bob vanishes, Alice's coins are lost
- CheckLockTimeVerify
 - 12345 OP_CLTV
 - script evaluation fails if blockchain < 12345 blocks
- Idea: After some time, Alice gets refund
- -> Exercise <https://jonasnick.github.io/workshop/tutorial-part-3.html>

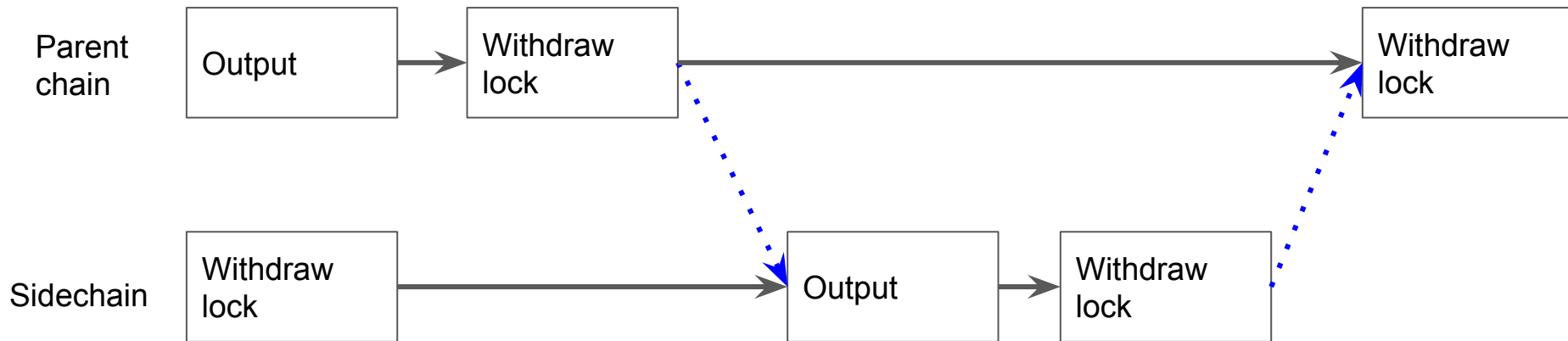
Part 4: Sidechains

Sidechains

- Observations
 - a. There is no single blockchain that meets all requirements.
 - b. Blockchains make different trade offs.
 - c. New blockchain rules need consensus, slow process.
 - d. Creating new blockchains from scratch is a huge challenge
 - Network effect, security
- Interoperability
 - a. Pass information from chain to chain in a trustless and automated way.
 - b. Leverage security from a different chain.
 - c. Common API.

Sidechains

- Use case: Add features to Bitcoin



Two-way peg

- Minimizes additional trust over Bitcoin's model
- Requires
 - Miner software upgrade
 - Softforking a new opcode into Bitcoin

Federated Peg

- Set of mutually distrusting functionaries
- Enforce the rules that Bitcoin is currently unable to.
- Uses m-of-n multisig instead of PoW.
- Auditable
- Allows creation of interoperable private chains.

Elements

<https://elementsproject.org> / <https://github.com/elementsproject/elements>

- Bitcoin Core code fork
- Uses federated peg
 - our public chain pegged to Bitcoin testnet
- Alpha released, Beta soon

Elements

<https://elementsproject.org> / <https://github.com/elementsproject/elements>

- Features

- Confidential Transactions (CT)
- More opcodes (OP_CSV, OP_CAT, ...)
- Segregated witness
- Federated peg
- Block signing
- Schnorr signatures
- Soon:
 - Asset Issuance
- In progress:
 - Full two-way peg
 - More powerful scripting system
 - More privacy
 - Better scalability

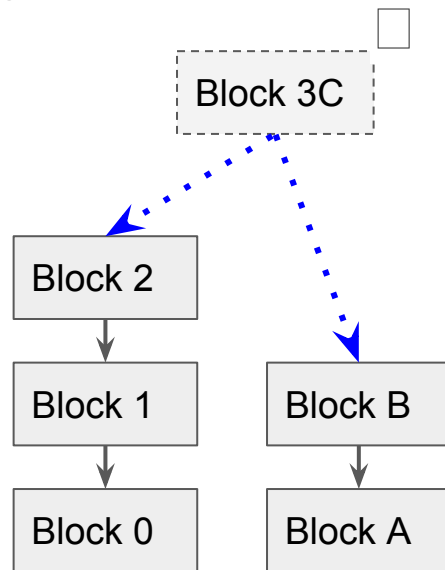
Liquid

<https://elementsproject.org/sidechains/liquid/>

- Production Bitcoin sidechain
- Based on elements
- Key feature: Improves interchange settlement lag (ISL)
 - Because Liquid uses federated Peg: improves latency, throughput
- + Elements features (CT)
- Primarily for Bitcoin exchanges, payment processors, traders
- Launch in summer 2016

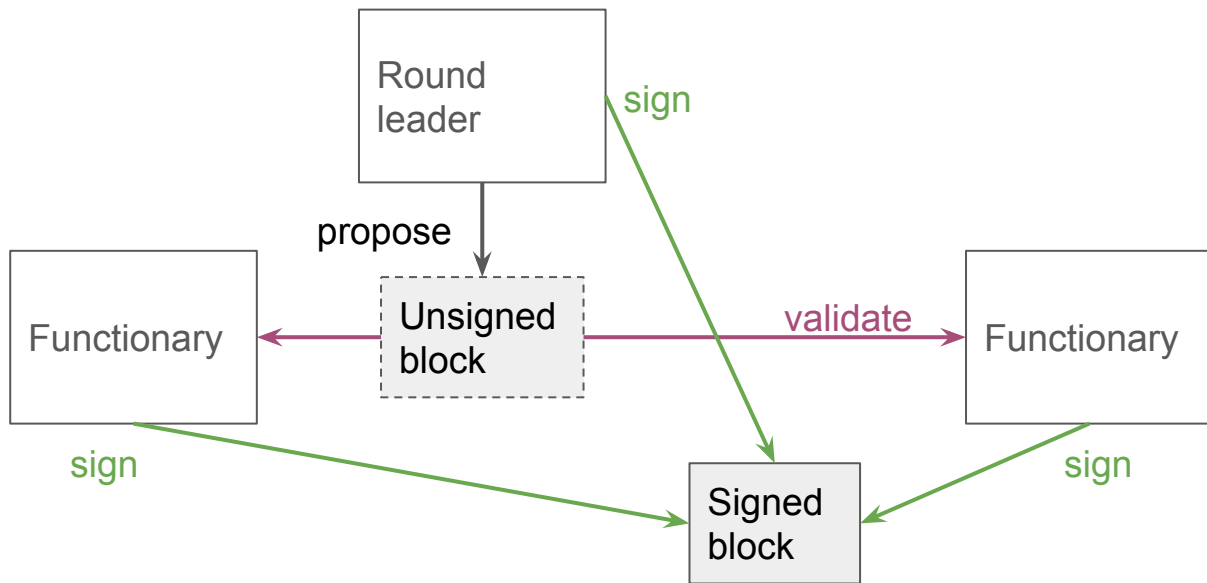
Block creation in sidechains

- Option: Leverage Bitcoin security with merged mining
 - Merged mined sidechain blocks are also valid bitcoin blocks.
 - When miner solves PoW for the block, work is sufficient for sidechain, parent chain or both.
 - Allows mining (securing) multiple chains at once instead of dividing work.
 - Miners collect transaction fee on the merge-mined chain
 - Requires miner software upgrade



Block creation in sidechains

- Option: Block signing
 - Valid blocks do not require PoW, but m-of-n multisignatures instead.
 - Separate software and network responsible for creating blocks.



New Hands-on Setup

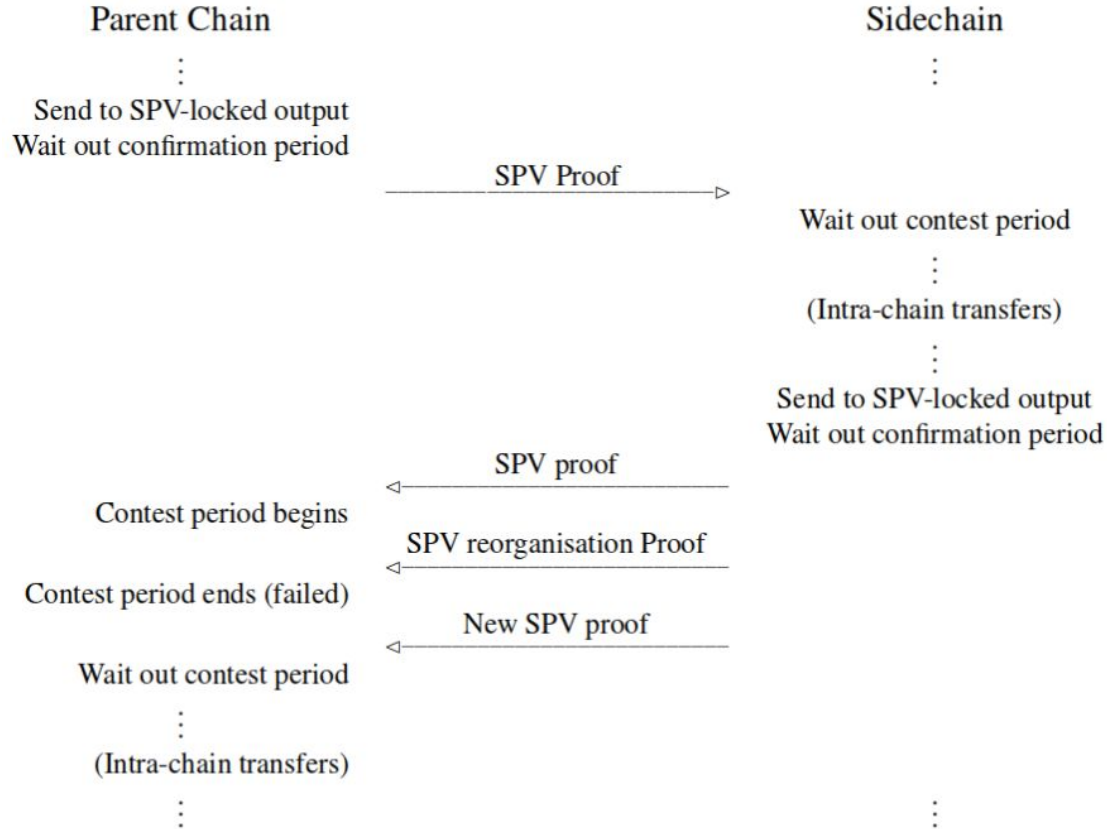
- Create own sidechain that uses block signing and federated peg.
 - Feature: 2MB blocks
- You are only user of your chain (initially).
- No coins without peg-in.

Part 5: Peg mechanisms

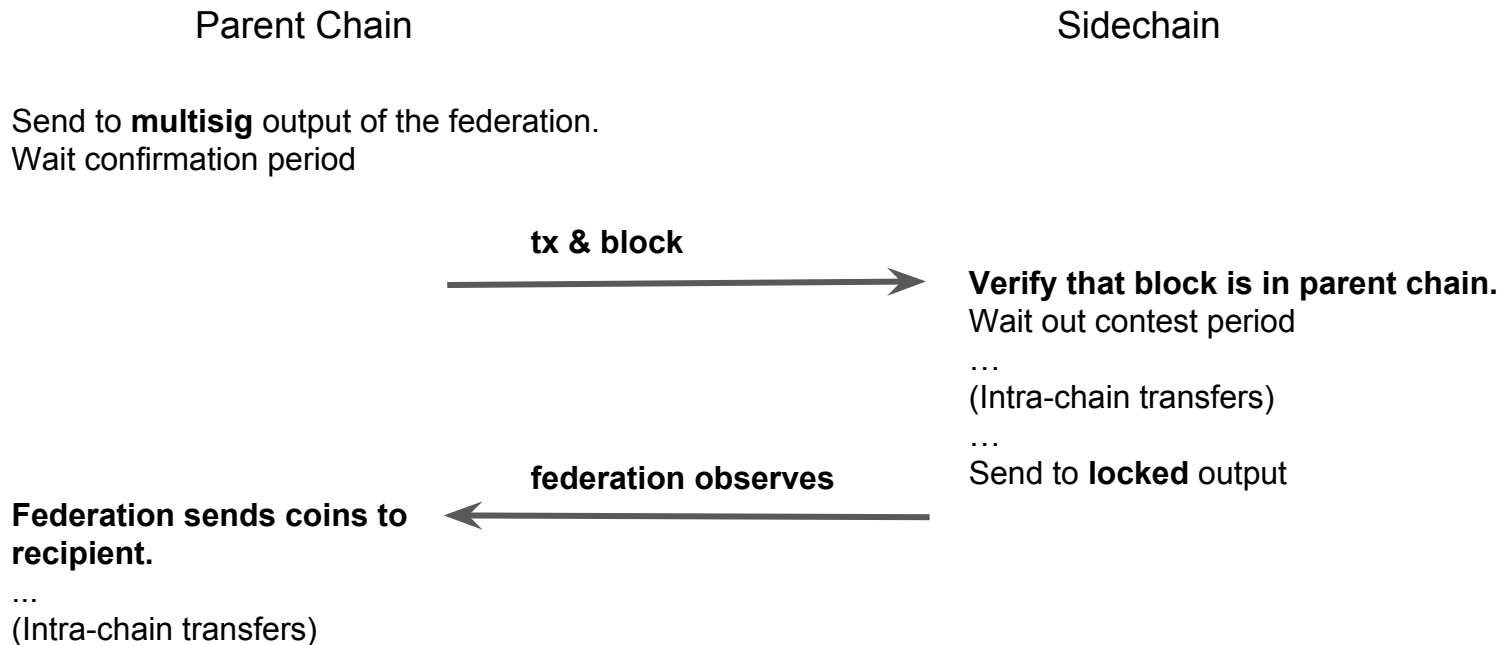
SPV Proof

- Small Proof “There is x work on top of block y in chain z”.
- Verifier can’t check validity of the block
 - But in the long run can always distinguish between majority chain and attacker chain.
- Usually accompanied by Merkle proof that tx is in block.
- Is used in lightweight wallets.
- Is used in two-way peg.
 - Proof that a lock transaction took place.
 - Pegged chains should not be required to validate each other’s rules.

Two-way peg

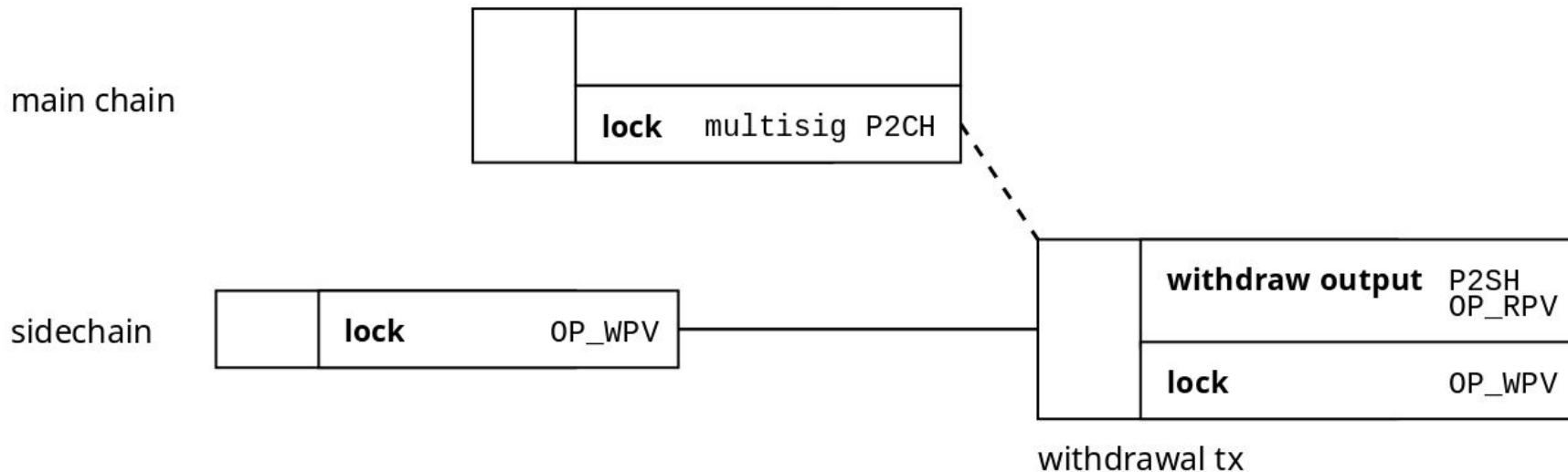


Federated peg

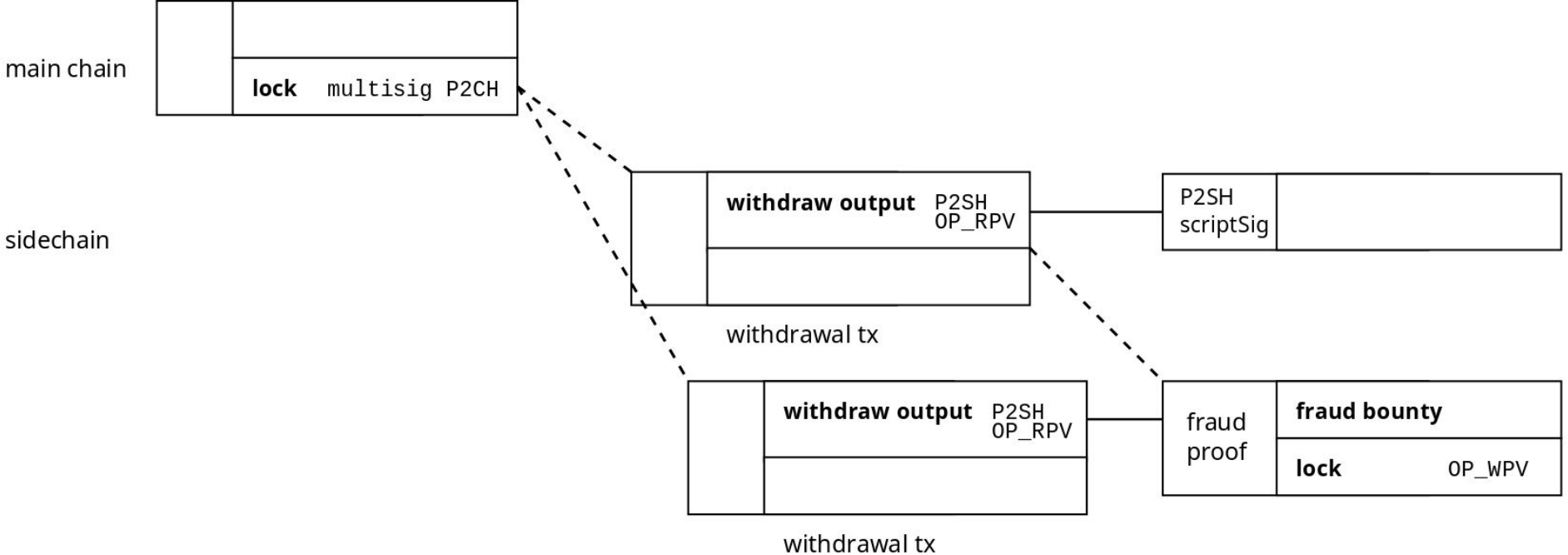


Peg in Elements Alpha

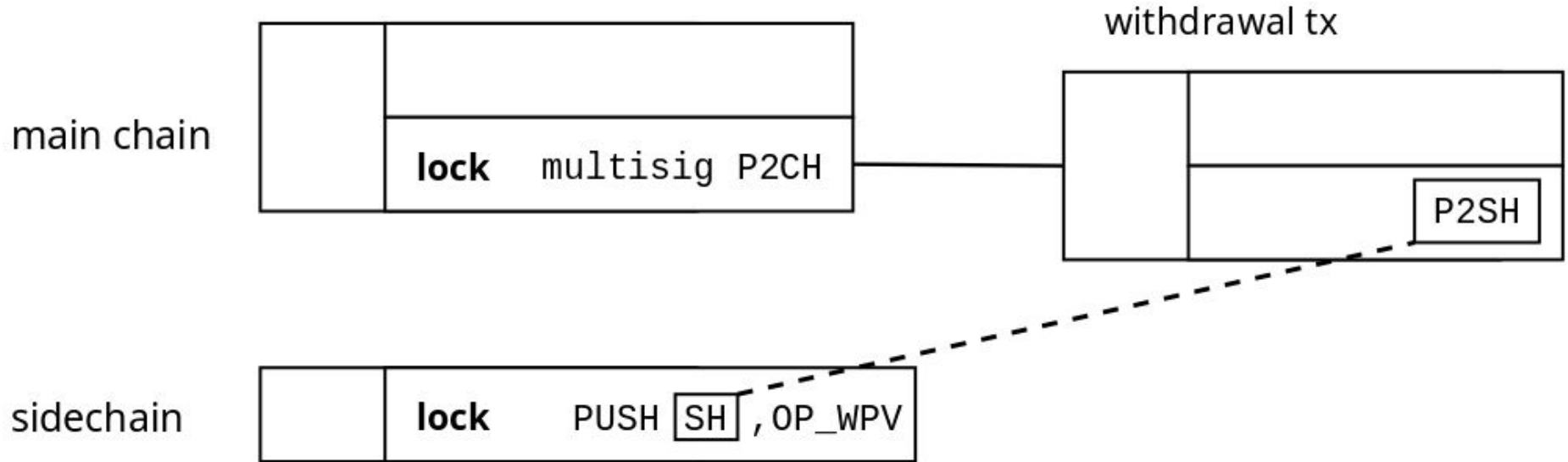
- Genesis block:
 - 21 million coins, OP_WITHDRAWPROOFVERIFY (OP_WPV) locked
 - OP_WPV is special, must be spent by tx with withdraw output and relock
 - Withdraw output: OP_IF ... OP_RPV OP_ELSE 144 OP_CSV P2SH OP_ENDIF



Fraud Proofs



Peg-out in Elements Alpha

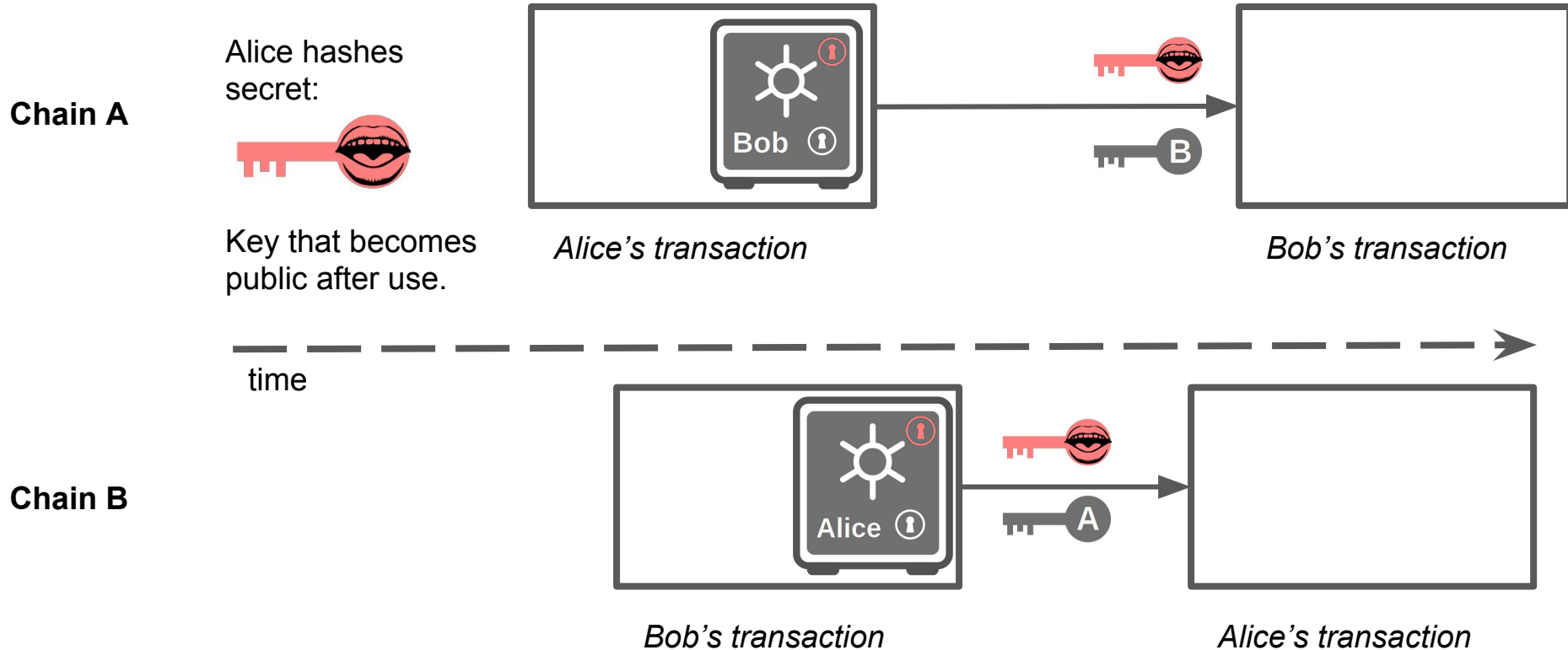


Withdrawwatcher

- Members of the federation run withdrawwatcher program
 - Separate network
- Identifies outputs on the parent chain
- Watches sidechain for withdraw transactions.
- Round leader proposes parent chain transaction.
- Functionaries validate tx, sign and collect signatures.
- Broadcast to the network if enough signatures.

Atomic Cross Chain Swap (ACCS)

Alice (Chain A) swaps coins with Bob (Chain B) relatively quickly and without requiring trust. Simplified.



Conclusion

- Bitcoin is a versatile platform for blockchain apps.
- Elements Project adds state of the art features and can be starting point for custom blockchains.
- Sidechains will power production applications by summer.
- A lot of research and innovation is happening in this space.