BitVM

Smarter Bitcoin Contracts

bitvm.org

Motivation

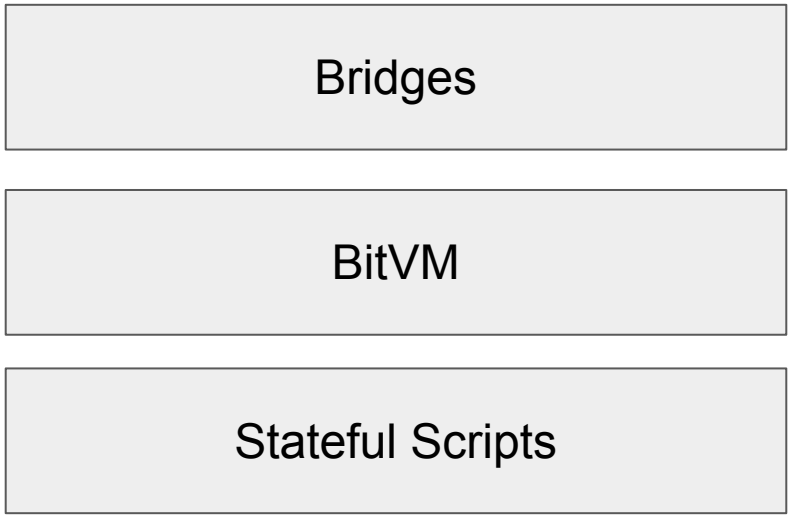
1 000 000 000

Bitcoiners

Bridging Bitcoin to other Systems

- Most users won't be able to afford mainchain fees
- BTC on sidechains, zk-rollups, zkCoins, ...
- Rapid innovation & cheap experimentation
- All L2s interconnected via Lightning
- How to build better bridges though?

Overview



Stateful Bitcoin Scripts

Introducing State with Signatures

- Idea: If we could sign a value...
- Enforce the same value for X in script1 and script2
  - Alice signs X=42 in script1
  - Bob uses that signature in script2
  - Note: Bob could be anyone
- How to sign a value though?
- We don't have CSFS...

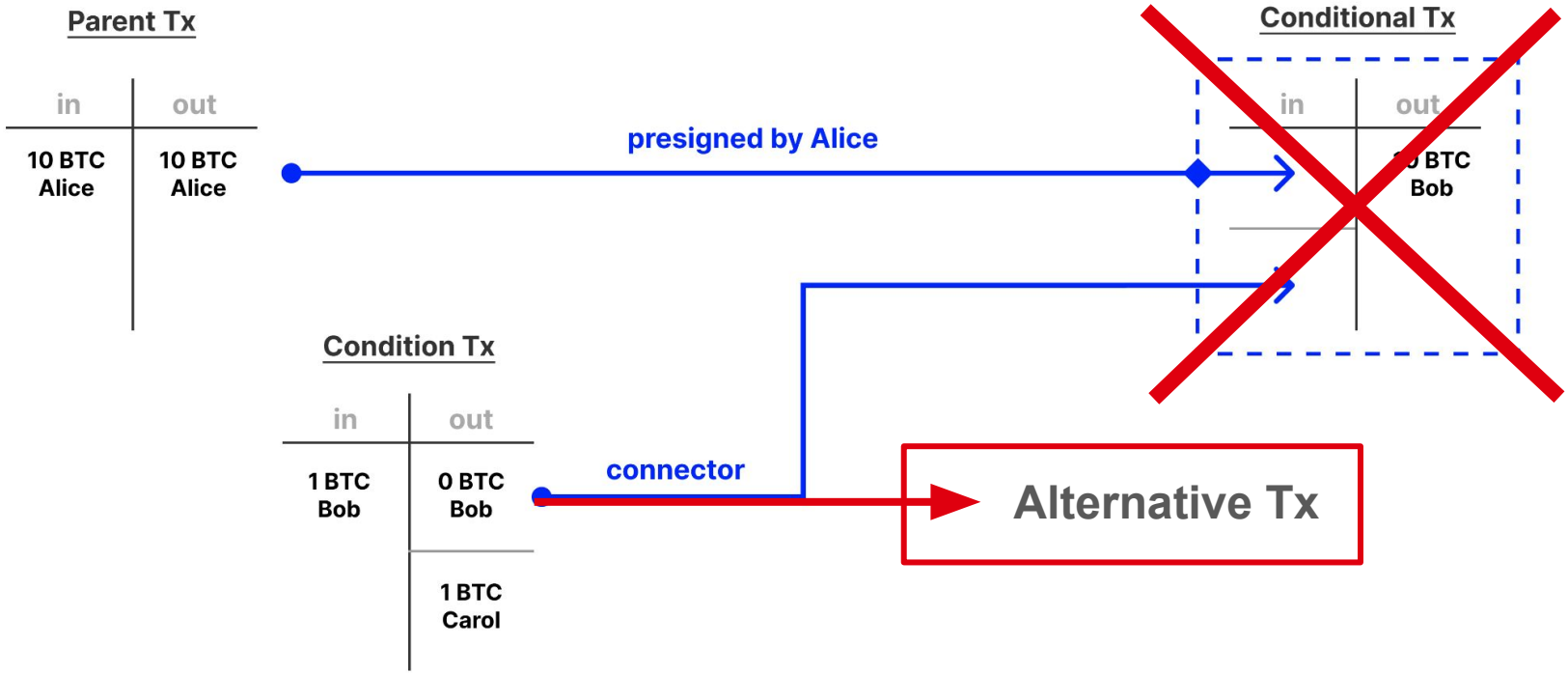
Lamport Signatures

- Conceptually very simple
- Require only hash functions
- Possible in Bitcoin today
- Main drawback: large
- But one can sign values of size u8, u32, u160, ...

Lamport Signature for a 1-bit Message

```
OP_HASH160
OP_DUP

<0xf592e757267b7f307324f1e78b34472f8b6f46f3>    // This is hash1
OP_EQUAL
OP_DUP

OP_ROT
<0x100b9f19ebd537fdc371fa1367d7ccc802dc2524>    // This is hash0
OP_EQUAL

OP_BOOLOR
OP_VERIFY

// Now the value of the bit commitment is on the stack. Either "0" or "1".
```

Introducing State with Connector Outputs



BitVM Architecture

The BitVM Paradigm

- Optimistic computation (only required in case of a dispute)
- Disprove a faulty assertion (much easier than execution)
- Universal computation enabled by SNARK verifier

Advanced Bitcoin Scripts

- Express complex Bitcoin Contracts
- Templating language for Script
  - Unroll loops
  - Compose functions
- Composite opcodes (xor, shift, mul, blake3, field arithmetic, ...)
- Statefulness
  - Lamport signatures (u8, u32, u160, ...)
  - Connector outputs
- Potentially complex scripts, complex Taptrees, and large TX graphs

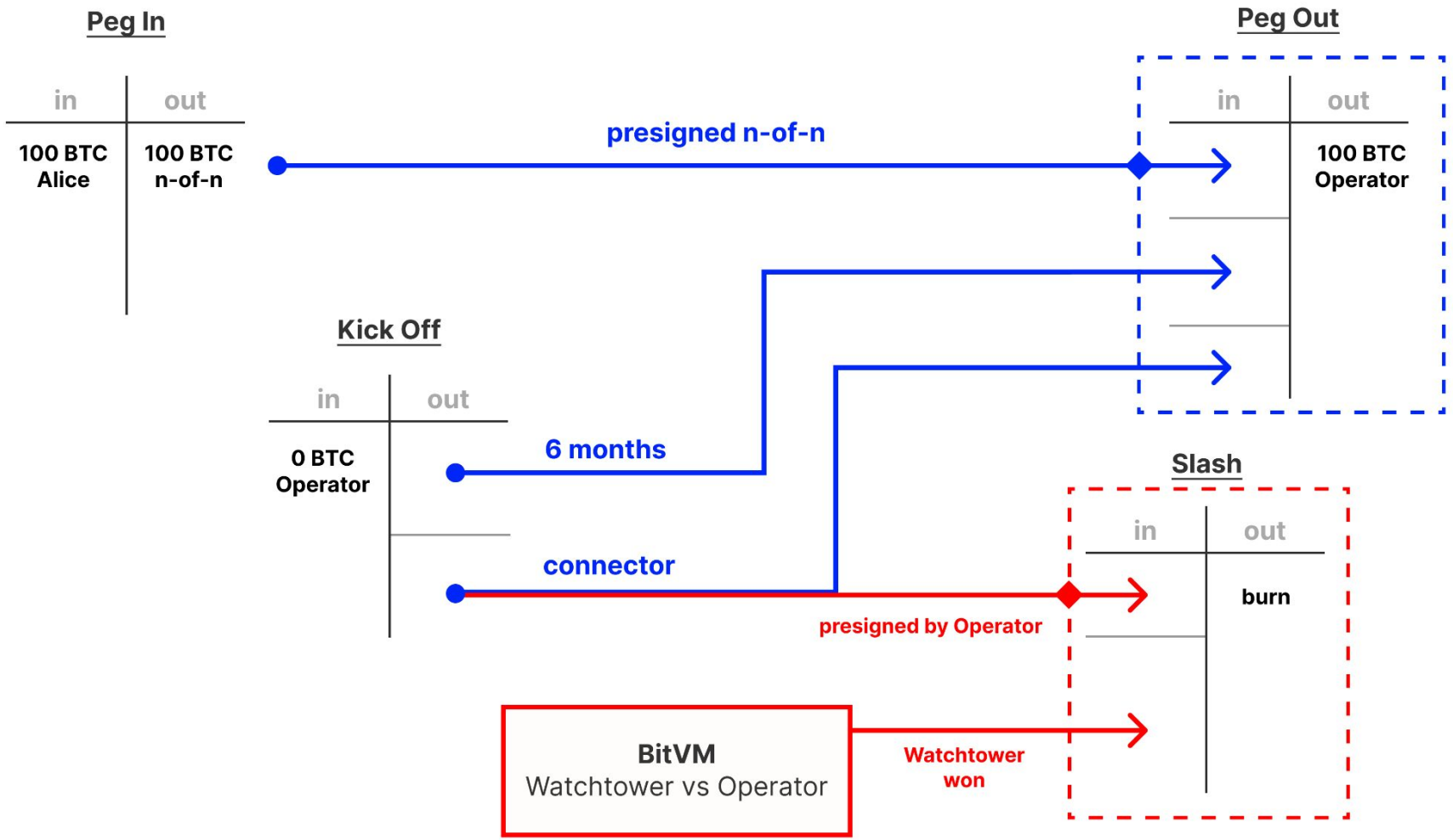
BitVM Bridges

BitVM Bridges

- Bridge BTC to any other system
- Idea: a bit clunky is fine
- Bridge is used rarely. Only large amounts
- End users use cross-chain swaps (LN)
- Fixed set of operators, but anyone can be a verifier

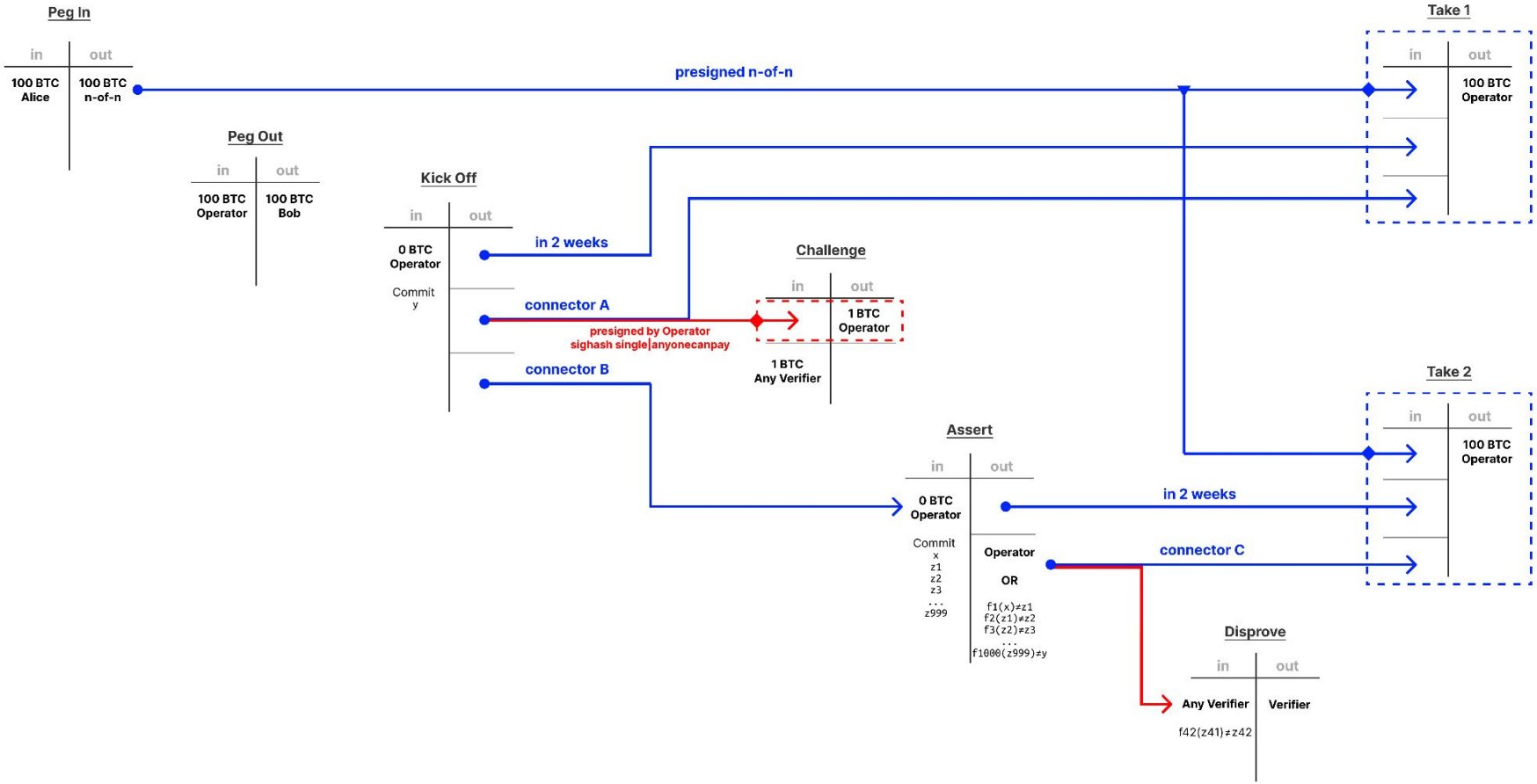
BitVM Bridge Guarantees

- A federation, but a single honest member suffices
  - 1000 cosigners for trusted setup
  - 100 bridge operators
- Guarantees
  - safe: nobody can steal deposits (1 of 1000 cosigners)
  - live: nobody can stop a valid peg out (1 of 100 operators)
- *You* can become a member. So you don't have to trust anyone


Peg In
in
out
100 BTC
Alice
100 BTC
n-of-n
Peg Out
in
out
100 BTC
Operator
presigned n-of-n
Kick Off
in
out
0 BTC
Operator
6 months
connector
Slash
in
out
burn
presigned by Operator
BitVM
Watchtower vs Operator
Watchtower
won

SNARK Verifier in Script

- Pairing-based proofs are constant size (Groth16, fflonk, ...)
- Implementation in Script is huge (gigabytes!)
- Script size limit is 4 MB (a full block)
- Idea: commit to 1000 intermediate results
  - f(x) = y
    - f1(x) = z1
    - f2(z1) = z2
    - f3(z2) = z3
    - ...
    - f1000(z999) = y
- Disproving a single step suffices
  - For example f42(z41) ≠ z42
- Every f_i can be up to 4 MB. That's 4 GB in total!


Peg In
in
out
100 BTC
Alice
100 BTC
n-of-n
presigned n-of-n
Take 1
in
out
100 BTC
Operator
Peg Out
in
out
100 BTC
Operator
100 BTC
Bob
Kick Off
in
out
0 BTC
Operator
Commit
y
in 2 weeks
connector A
connector B
presigned by Operator
sighash single|anyonecanpay
Challenge
in
out
1 BTC
Operator
1 BTC
Any Verifier
Assert
in
out
0 BTC
Operator
Commit
x
z1
z2
z3
...
z999
Operator
OR
f1(x)≠z1
f2(z1)≠z2
f3(z2)≠z3
...
f1000(z999)≠y
in 2 weeks
connector C
Take 2
in
out
100 BTC
Operator
Disprove
in
out
Any Verifier
Verifier
f42(z41)≠z42

Limitations

- Complexity
- Balancing incentives: Loser should pay winner's fees + bounty
- If incentives are balanced the chain is not needed
- Operator has to front capital for 2 weeks
- But no 1:1 collateral required
- For every peg-in all 1000 parties have to pre-sign 100 peg-out TXs
- Federation can censor peg-ins

Summary & Outlook

- BitVM enables smarter Bitcoin contracts
- Use case: trust-reduced bridges for L2s
- Limitation: practical but clunky
- Requires no softfork
  - better: TXHASH, OP_MUL, OP_BLOCKHASH
- Draft version completed
- "Reckless" mainnet version this year

Questions?

bitvm.org

Thank you!

bitvm.org