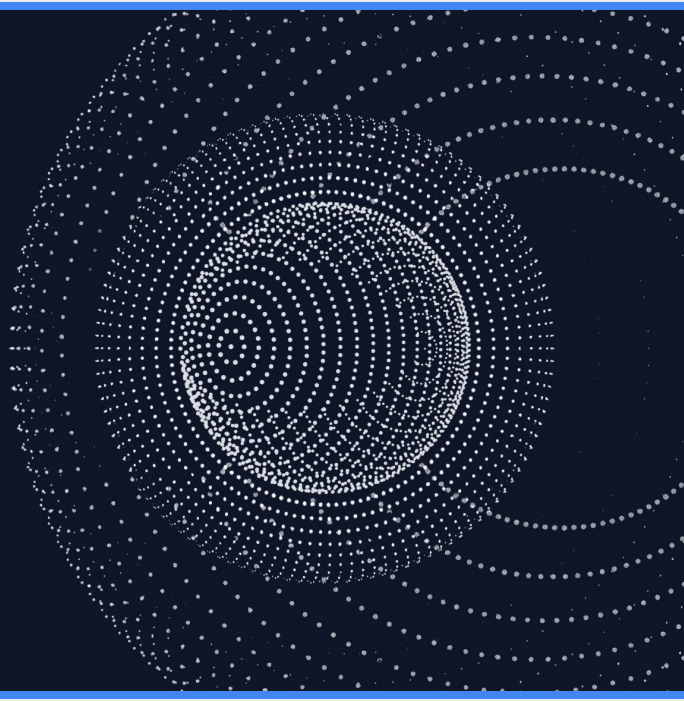
Fairgate

Trust-minimized bridging Bitcoin with BitVMX and Stylus



Sergio Demian Lerner | CTO, Fairgate Labs

NOV 20, 2025

Bridging Blockchains Requires Computation

- **Offchain Trusted Computation**:
  - TEE, Federated
- **Onchain Non-TC Space-bounded computation**
  - Bitcoin, Litecoin
- **Onchain Metered Computation**
  - Ethereum.
  - All work onchain
- **Proof of Computation Integrity**
  - SNARKs, STARKs, ZK-Rollups.
  - Most work off-chain, some work on-chain.
- **Disputable Computation (“Optimistic”)**
  - TrueBit, Optimistic Rollups (Arbitrum), **BitVM.**
  - Even less work onchain.

Computing on Arbitrum is cheap…

Computing on Bitcoin is impractical and expensive, but not impossible

The temporary change to script.cpp on August 15th, 2010

misc changes

```
94                 if (opcode == OP_CAT ||
95                     opcode == OP_SUBSTR ||
96                     opcode == OP_LEFT ||
97                     opcode == OP_RIGHT ||
98                     opcode == OP_INVERT ||
99                     opcode == OP_AND ||
100                    opcode == OP_OR ||
101                    opcode == OP_XOR ||
102                    opcode == OP_2MUL ||
103                    opcode == OP_2DIV ||
104                    opcode == OP_MUL ||
105                    opcode == OP_DIV ||
106                    opcode == OP_MOD ||
107                    opcode == OP_LSHIFT ||
108                    opcode == OP_RSHIFT)
109                    return false;
110
```

OPCODES

Constants

OP_0, OP_FALSE
N/A
OP_PUSHDATA1
OP_PUSHDATA2
OP_PUSHDATA4
OP_1NEGATE
OP_1, OP_TRUE
OP_2-OP_16

Bitwise logic

OP_INVERT
OP_AND
OP_OR
OP_XOR
OP_EQUAL
OP_EQUALVERIFY

Cryptographic

OP_RIPEMD160
OP_SHA1
OP_SHA256
OP_HASH160
OP_HASH256
OP_CODESEPARATOR
OP_CHECKSIG
OP_CHECKSIGVERIFY
OP_CHECKMULTISIG
OP_CHECKMULTISIGVERIFY
OP_CHECKSIGADD

Flow control

OP_NOP
OP_IF
OP_NOTIF
OP_ELSE
OP_ENDIF
OP_VERIFY
OP_RETURN

Arithmetic 2

OP_BOOLAND
OP_BOOLOR
OP_NUMEQUAL
OP_NUMEQUALVERIFY
OP_NUMNOTEQUAL
OP_LESSTHAN
OP_GREATERTHAN
OP_LESSTHANOREQUAL
OP_GREATERTHANOREQUAL
OP_MIN
OP_MAX
OP_WITHIN

Splice

OP_CAT
OP_SUBSTR
OP_LEFT
OP_RIGHT
OP_SIZE

Locktime

OP_CHECKLOCKTIMEVERIFY (previously OP_NOP2)
OP_CHECKSEQUENCEVERIFY (previously OP_NOP3)

Arithmetic

OP_1ADD
OP_1SUB
OP_2MUL
OP_2DIV
OP_NEGATE
OP_ABS
OP_NOT
OP_0NOTEQUAL
OP_ADD
OP_SUB
OP_MUL
OP_DIV
OP_MOD
OP_LSHIFT
OP_RSHIFT

Stack

OP_TOALTSTACK
OP_FROMALTSTACK
OP_IFDUP
OP_DEPTH
OP_DROP
OP_DUP
OP_NIP
OP_OVER
OP_PICK
OP_ROLL
OP_ROT
OP_SWAP
OP_TUCK
OP_2DROP
OP_2DUP
OP_3DUP
OP_2OVER
OP_2ROT
OP_2SWAP

Removed August, 2010

Emulation OP_MUL with OP_ADD

32-Bit Multiplication in 3575 bytes of Bitcoin Script

Small Script

Big Script

OP_CAT

Hashing

- OP_RIPEMD160
- OP_SHA1
- OP_SHA256
- OP_HASH160
- OP_HASH256

- OP_IFDUP
- OP_DEPTH

- OP_0, OP_FALSE
- OP_PUSHDATA1
- OP_1NEGATE
- OP_1, OP_TRUE
- OP_2-OP_16

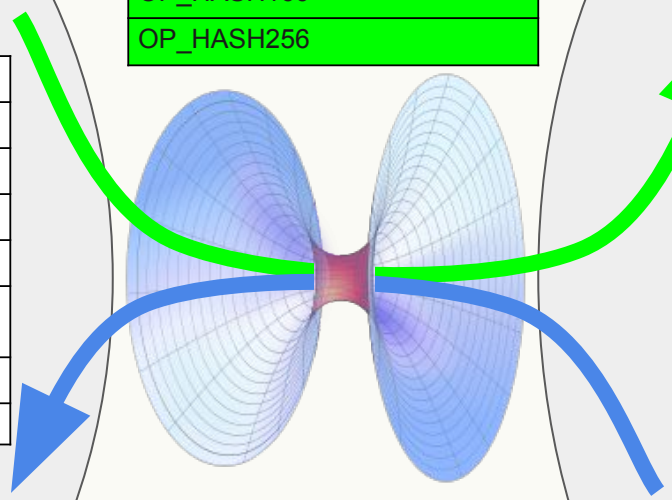
- OP_EQUAL
- OP_EQUALVERIFY

- OP_IF
- OP_NOTIF
- OP_VERIFY

- OP_CHECKLOCKTIMEVERIFY (previously OP_NOP2)
- OP_CHECKSEQUENCEVERIFY (previously OP_NOP3)

- OP_BOOLAND
- OP_BOOLOR
- OP_NUMEQUAL
- OP_NUMEQUALVERIFY
- OP_NUMNOTEQUAL
- OP_LESSTHAN
- OP_GREATERTHAN
- OP_LESSTHANOREQUAL
- OP_GREATERTHANOREQUAL
- OP_MIN
- OP_MAX
- OP_WITHIN

- OP_1ADD
- OP_1SUB
- OP_NEGATE
- OP_ABS
- OP_NOT
- OP_0NOTEQUAL
- OP_ADD
- OP_SUB



OP_SIZE

- OP_CHECKSIG
- OP_CHECKMULTISIG

- OP_SUBSTR
- OP_LEFT
- OP_RIGHT

- OP_PUSHDATA2
- OP_PUSHDATA4

- OP_CODESEPARATOR
- OP_CHECKSIGVERIFY
- OP_CHECKMULTISIGVERIFY

BitVM

Fairgate

What Enabled BitVM?

- Optimistic Protocols with Fraud Proofs
  - Blockstream sidechains (2014)
  - TrueBit (Ethereum)
  - Optimistic rollups on Ethereum (Arbitrum)
- Verification of Lamport/Winternitz signatures on Bitcoin
- Transferring state between Bitcoin transactions using Lamport Signatures.

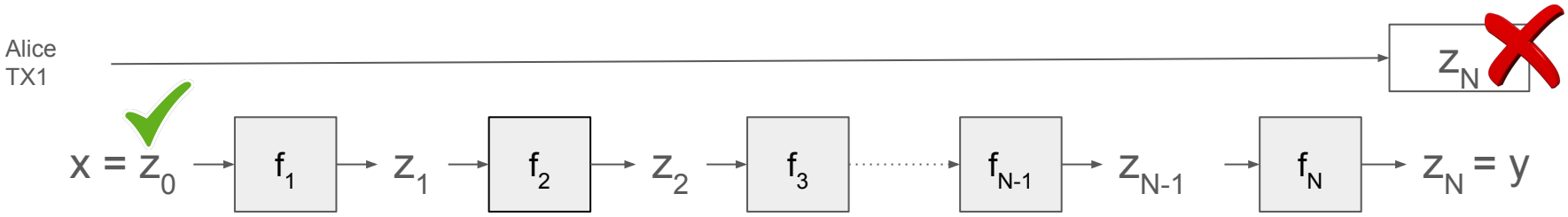
Active BitVM-like Protocols

- BitVM2 (1 round, finalized but deprecated)
- BitVM3s (Disposable GC, almost practical)
- **BitVMX-CPU** (RISC-V, BETA)
- **BitVMX-GC (Mixed Arithmetic Garbled Circuits)**

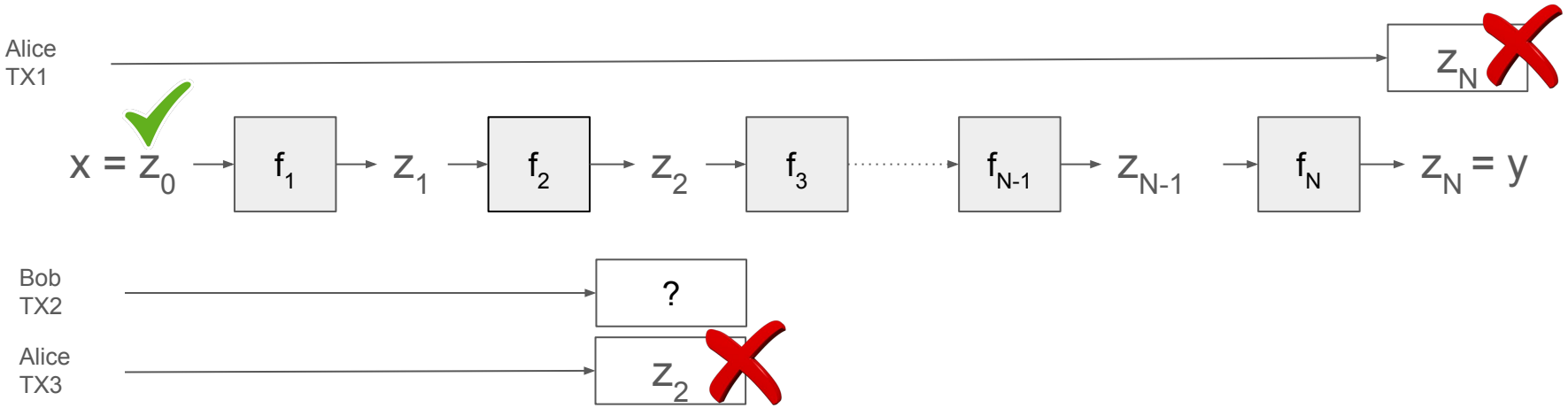
BitVMX-CPU

- We have a function f(x) that we want to compute on Bitcoin.
- We divide the function into N “smaller” functions such that...
  $y = f(x) = f_N (\ldots. f_2(f_1(x))$
- O sea:
  - $f_1(x) = z_1$
  - $f_2(z_1) = z_2$
  - $f_3(z_2) = z_3$
  - …
  - $f_N(z_{N-1}) = z_N = y$
- Let’s assume that the intermediate states z_i are small.

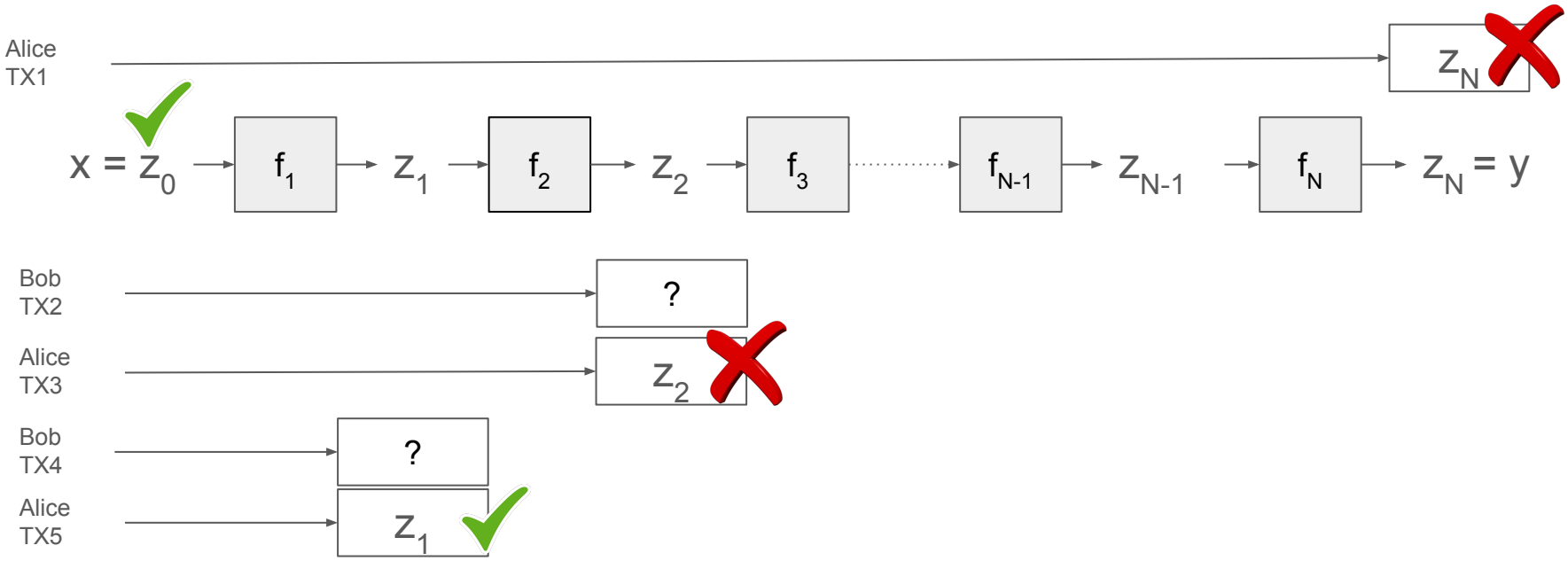
BitVMX-CPU



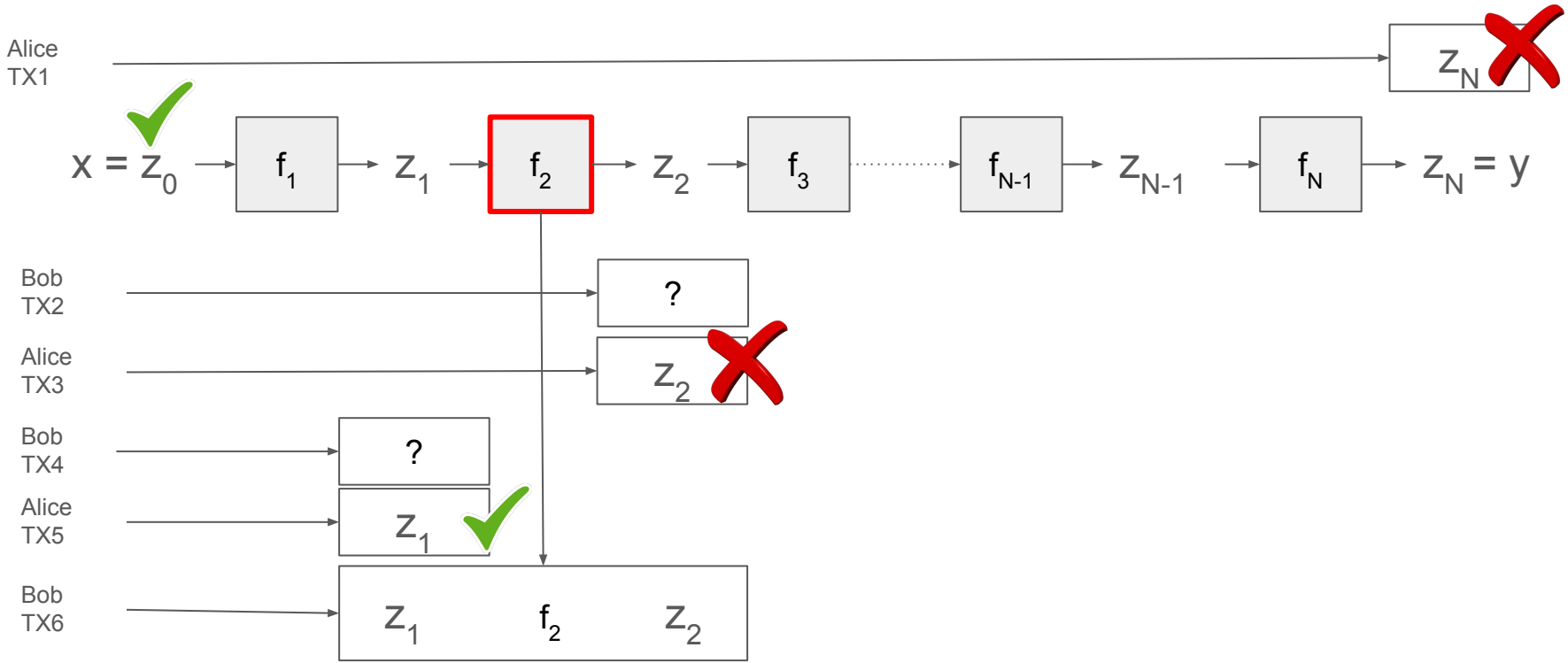
BitVMX-CPU


Alice
TX1
z_N
$x = z_0$
f_1
z_1
f_2
z_2
f_3
f_{N-1}
z_{N-1}
f_N
$z_N = y$
Bob
TX2
?
Alice
TX3
z_2

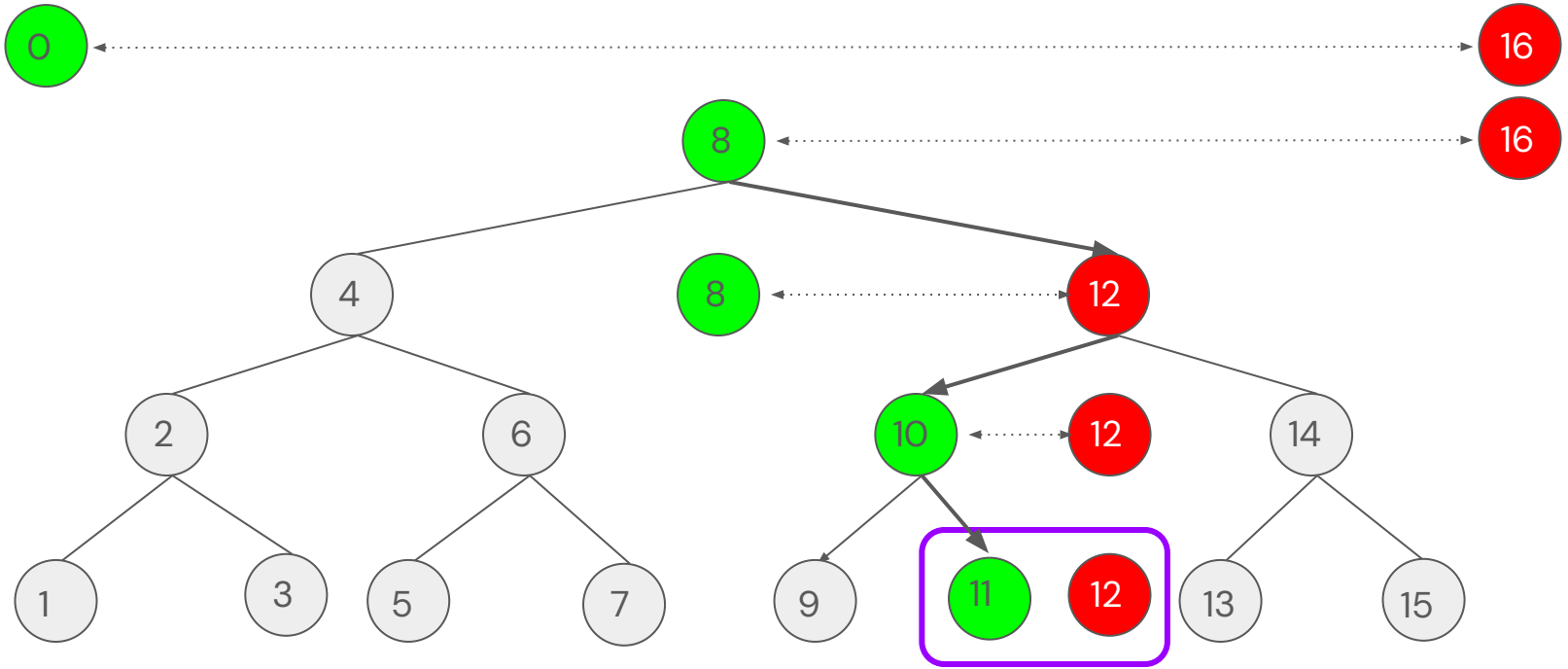
BitVMX-CPU


Alice
TX1
z_N
$x = z_0$
f_1
z_1
f_2
z_2
f_3
f_{N-1}
z_{N-1}
f_N
$z_N = y$
Bob
TX2
?
Alice
TX3
z_2
Bob
TX4
?
Alice
TX5
z_1

BitVMX-CPU


Alice
TX1
z_N
$x = z_0$
f_1
z_1
f_2
z_2
f_3
f_{N-1}
z_{N-1}
f_N
$z_N = y$
Bob
TX2
?
Alice
TX3
z_2
Bob
TX4
?
Alice
TX5
z_1
Bob
TX6
z_1 f_2 z_2

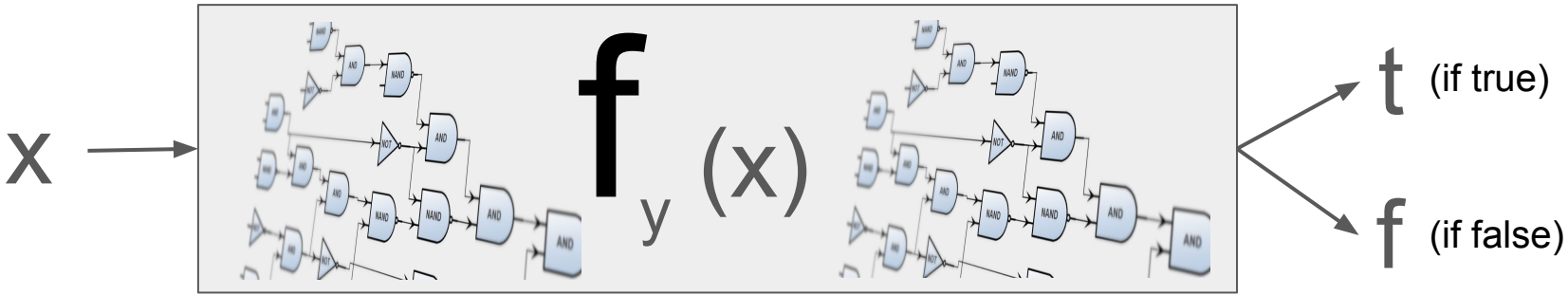
Malfunction Binary Search



BitVMX-CPU: RISC-V 32-Bit CPU

- 32 memory-mapped registers, referred to by number x0 – x31 (x0==0)
- Arithmetic operations can only be performed on registers
- Only memory actions are loads & stores
- 32-bit memory space
- Natively only support aligned accesses, unaligned access done by soft.
- Program Counter (PC)
- 2 mem word inputs and 1 mem word output + PC change per instruction
- Instructions are fixed, 32 bits long
- RV32I with M (mul/div) Extension

BitVMX-Garbled Circuits Paradigm

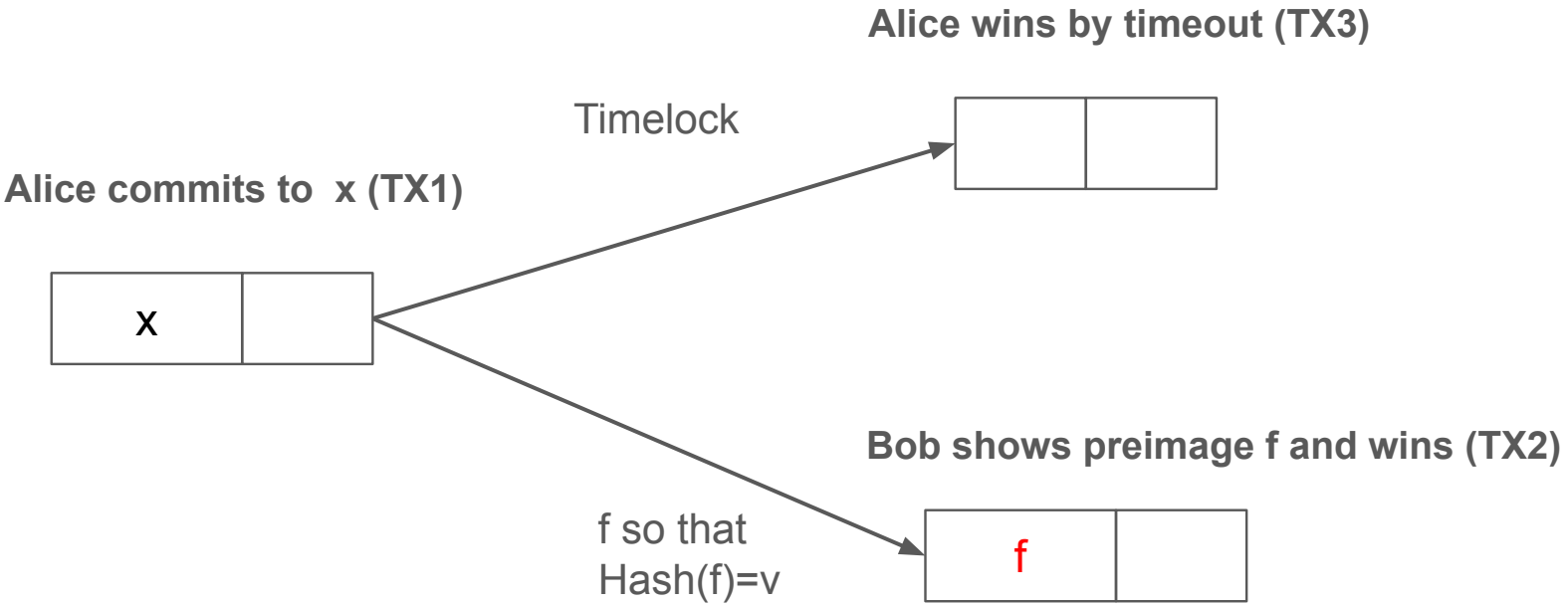


Conditional
Disclosure of Secrets

Alice
TX1 → X

Bob
TX2 → f

HTLCs in BitVM-Garbled Circuits



Program Sizes to Verify a Sidechain Peg-out

Verification method	Size
All	800 GB
SNARK	8 GB
BitVM2	4 MB
BitVMX-CPU	400 Kbytes
Garbled Circuits	4 Kbytes

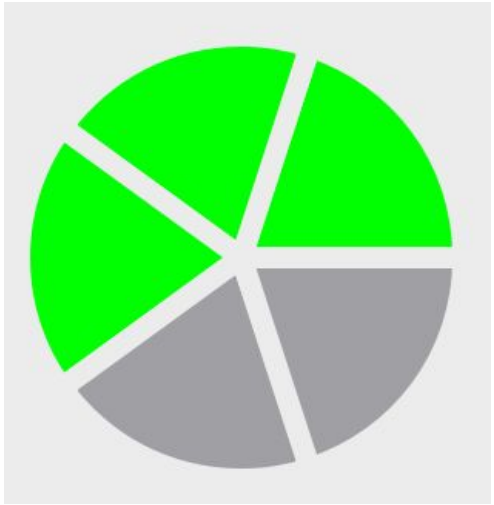
BitVM Limitations

- Protocol between N known participants
- Security condition: 1 of N is honest
- No contextual information (time, block hash)
- Minimum computation cost is non-negligible
- Persistent storage cost is non-negligible
- Builds immutable transaction graphs to emulate covenants
- Capital Efficient with TOOP and Trust-minimized Covenants 

1 of N Honesty

Fairgate

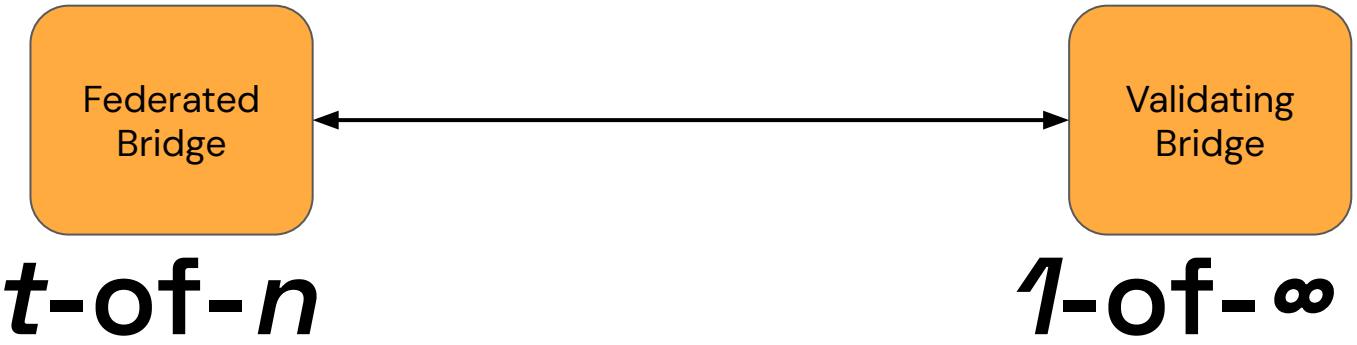
Federated Model (*t*-de-*N*)



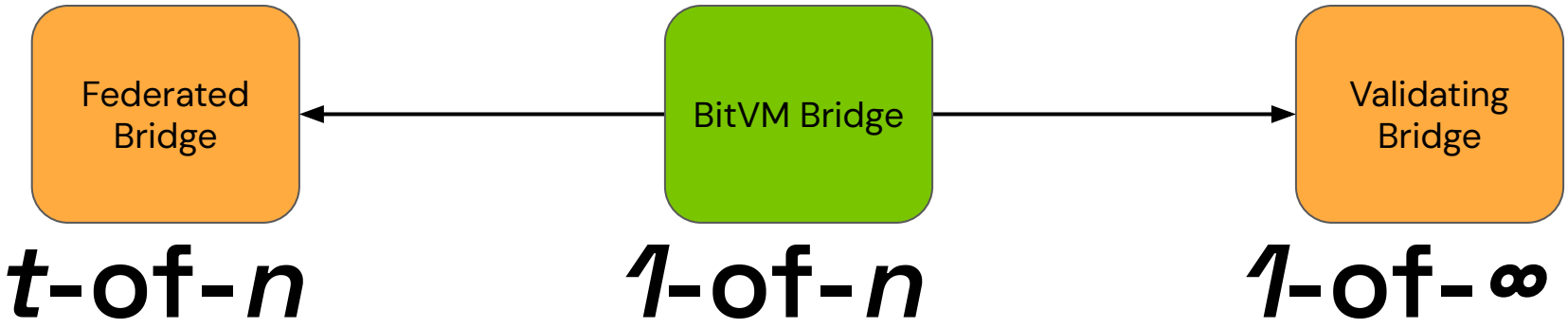
All federated bridges are based on the t-of-n security model. This model assumes that t members of a security committee of N are honest, in order to protect the funds. Normally, t represents the majority.

The committee cannot be permissionless: the anonymous addition of new members would dilute security through a Sybil attack.

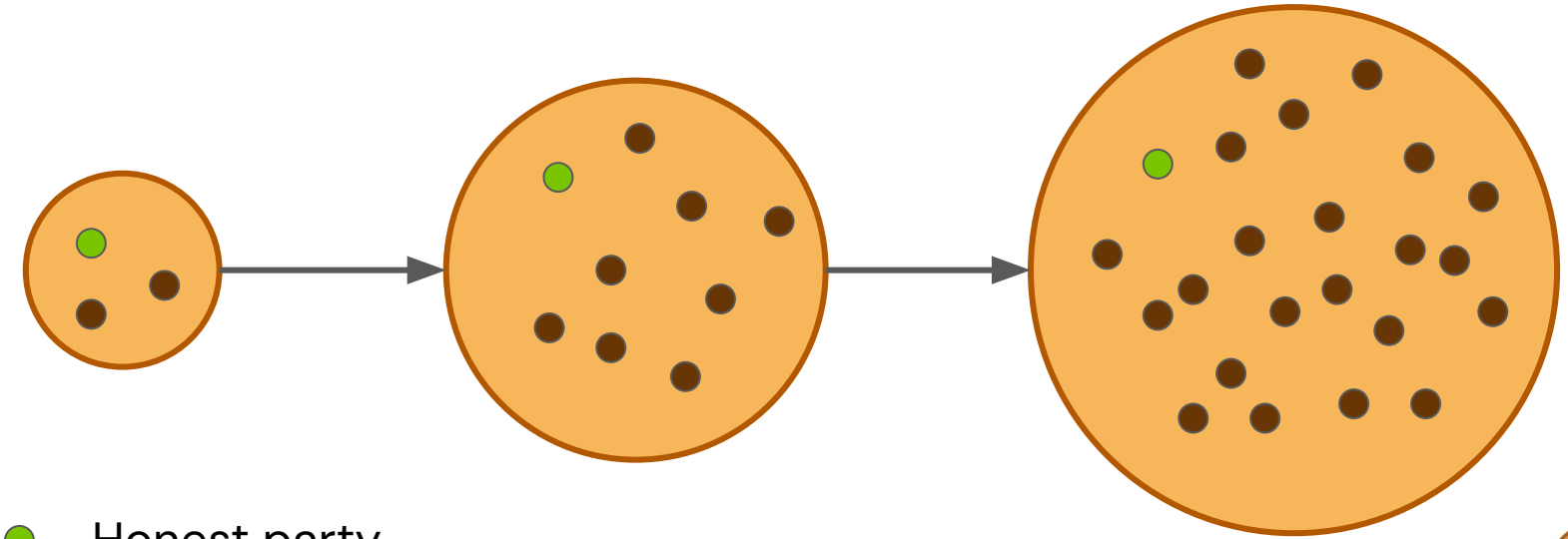
From *t*-of-*n* to *1*-of-∞



New *1*-of-*n* honesty assumption



Openness Without Security Dilution



- Honest party
- Party of unknown behaviour

BitVM 2025

Fairgate

BitVM/BitVMX Adoption on Bitcoin L2s

BitVMX: A Platform for Universal Computation on Bitcoin

BitVMX is a cutting-edge framework designed to optimistically execute arbitrary programs on Bitcoin, leveraging the N-party disputable computation paradigm introduced by BitVM.

With its foundation in secure, extensible, and open-source principles, **BitVMX paves the way for verifying computation on Bitcoin**.

Rootstock Labs

FairGate LABS

BitVMX FORCE

FOUNDING MEMBERS

Rootstock Labs

Fairgate

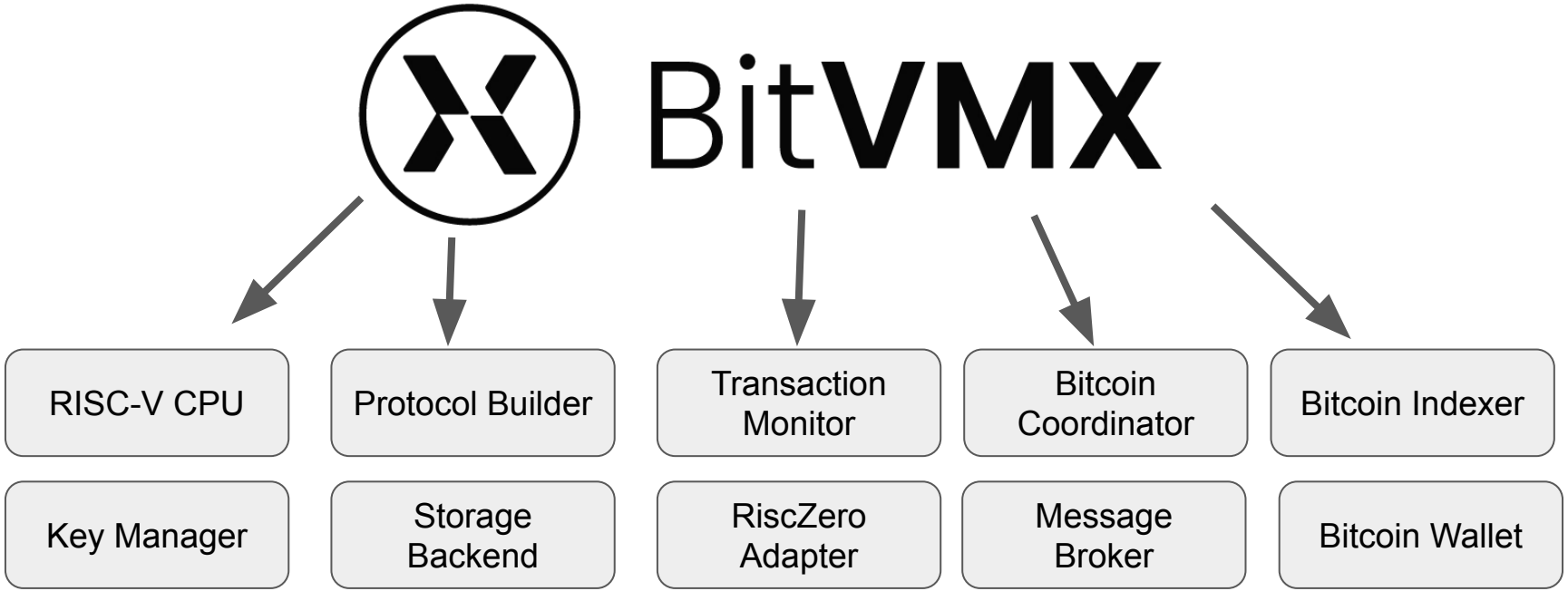
INPUT | OUTPUT

BitVMX-CPU

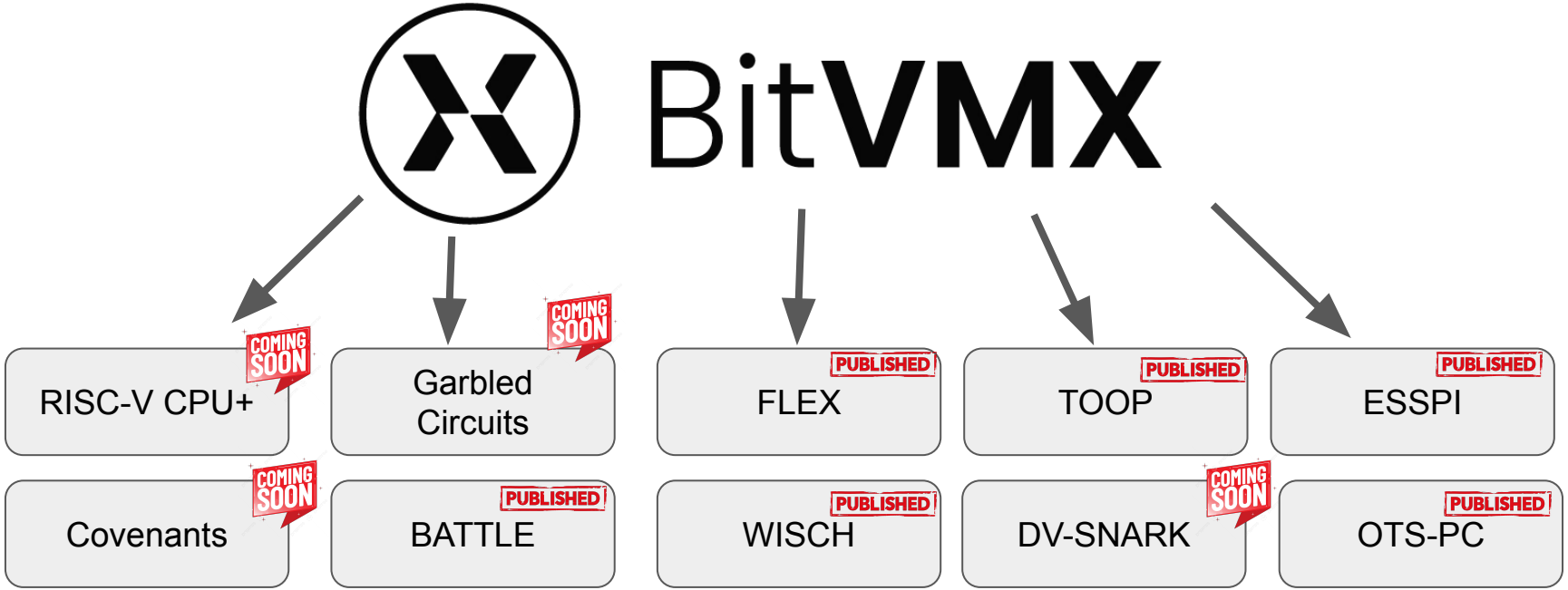
BitVMX Platform

BitVMX-GC

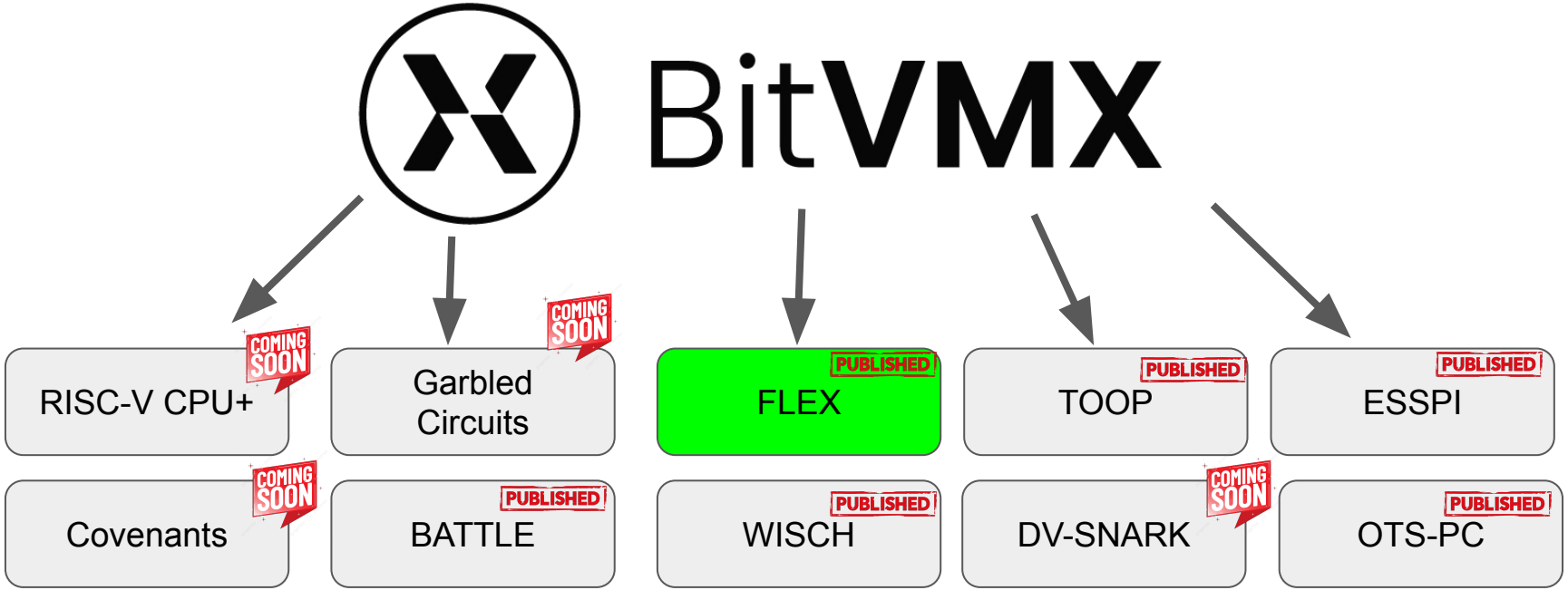
Current Components in BitVMX Platform



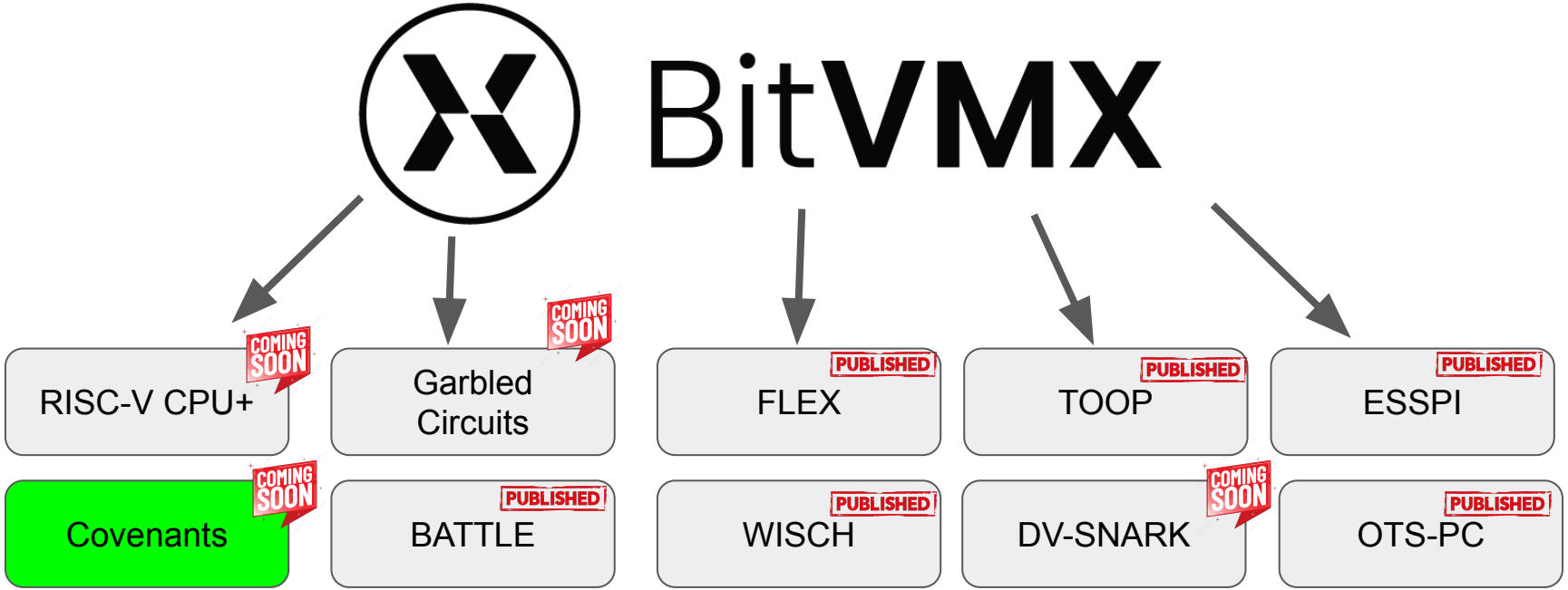
New Components Coming to the BitVMX Family 2026



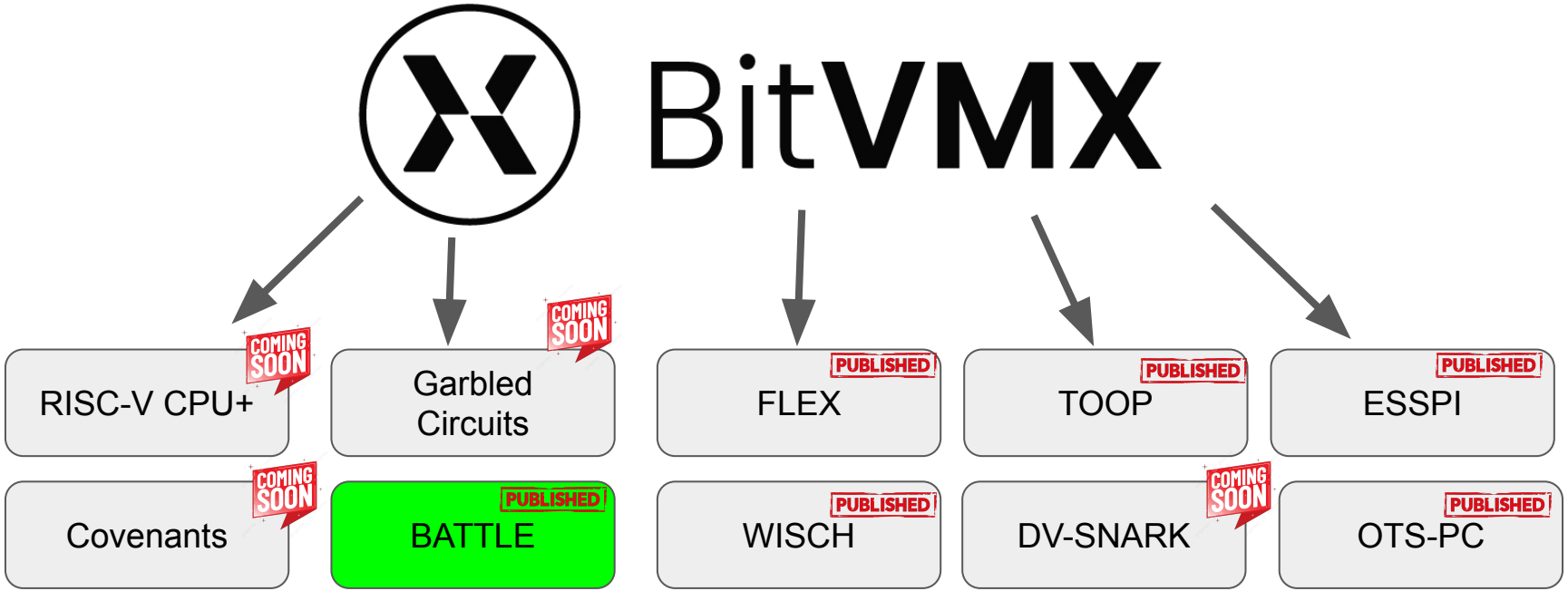
New Components Coming to the BitVMX Family 2026



New Components Coming to the BitVMX Family 2026



New Components Coming to the BitVMX Family 2026



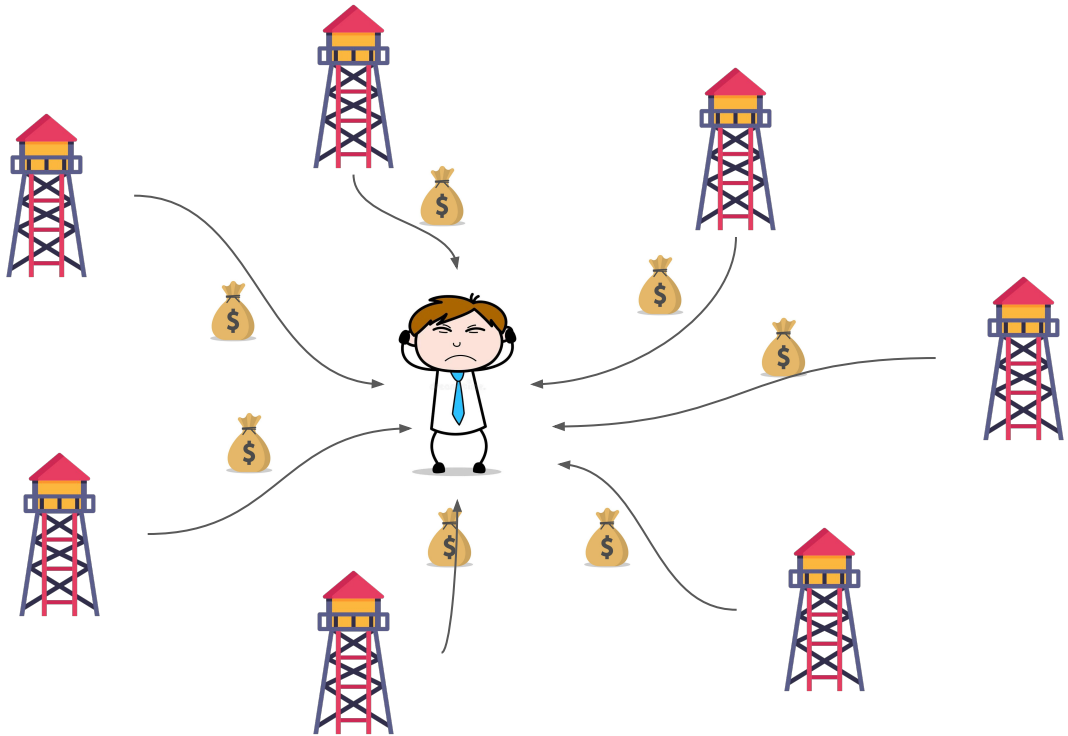
BitVMX **BATTLE**

Fairgate

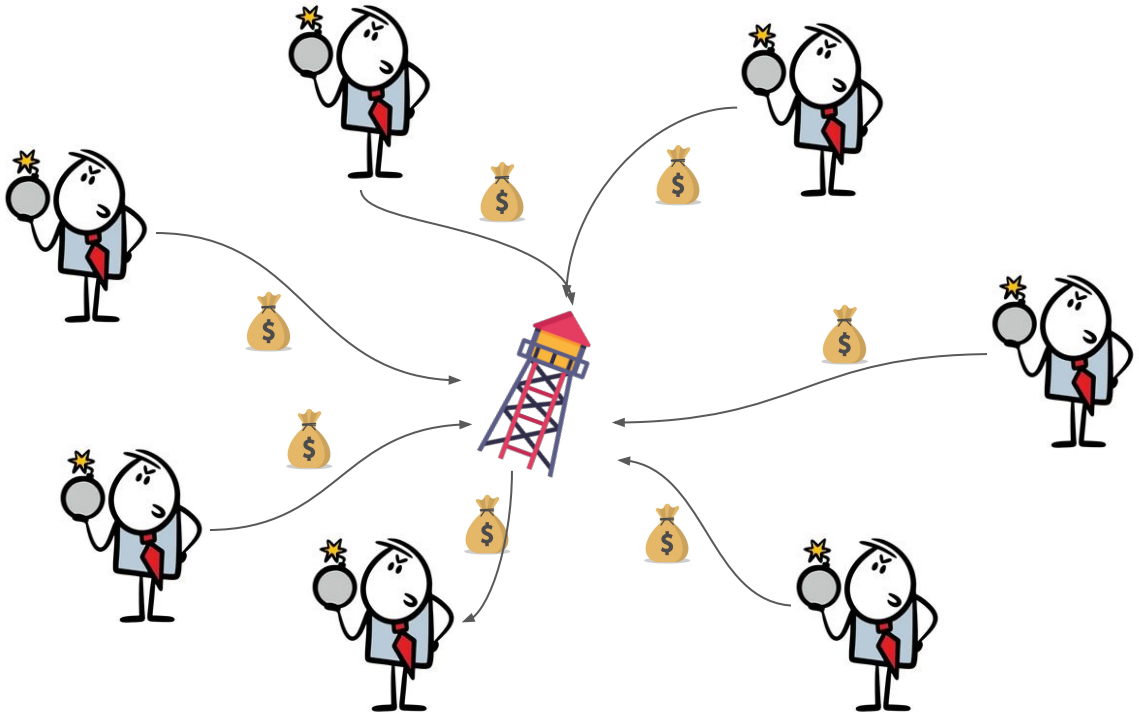
PROBLEM

- Most BitVM-based protocols are restricted to N < 20 watchtowers
- How to Scale an Optimistic Protocol on Bitcoin to $N \approx 1000$
- Most BitVM-based protocols are restricted to N < 20

PROBLEM 1: Watchtowers against one Claimant



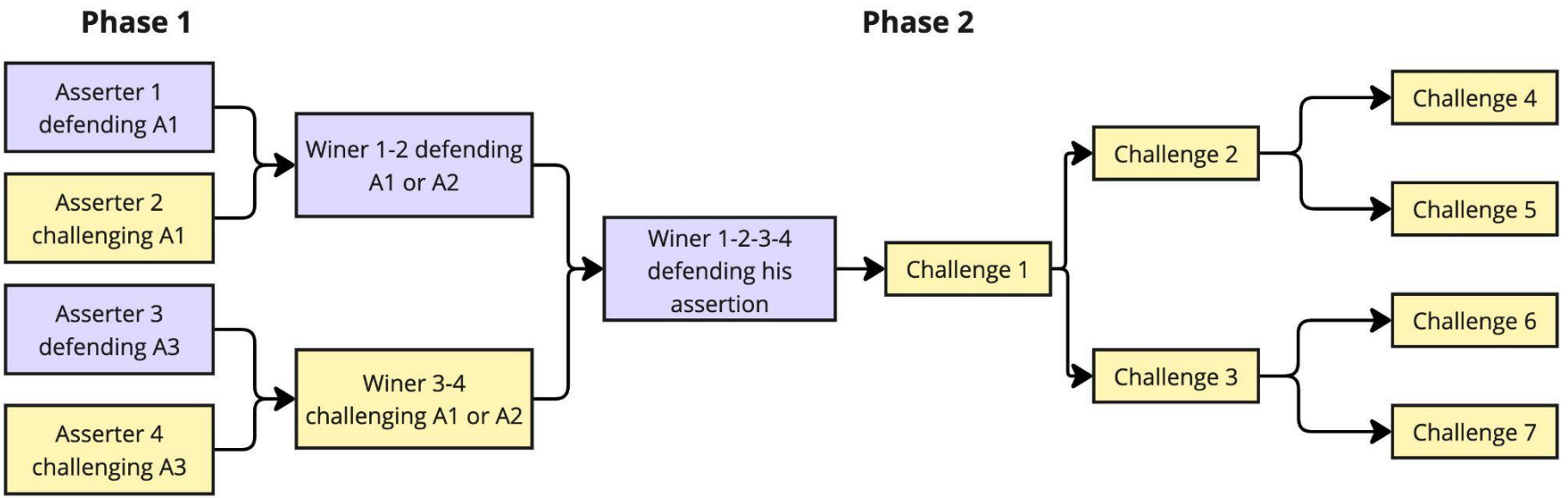
PROBLEM 2: Claimants against one Watchtower



Properties of BATTLE for Bitcoin

- Solves N-party Disputes in O(Log(N)) time
- Requires O(1) amount in security bonds
- Security bonds are only required on dispute with FLEX
- Setup Cost is $O(N^2)$ Practical for $N \approx 1000$ operators
- Tournament-based, similar to:
  - Arbitrum’s BoLD
  - Cartesi’s "Dave"
- More efficient than Arbitrum Classic, or Optimism’s fraud proof protocol (OPFP)

BATTLE Protocol Phases



Conflicting assertions compete.

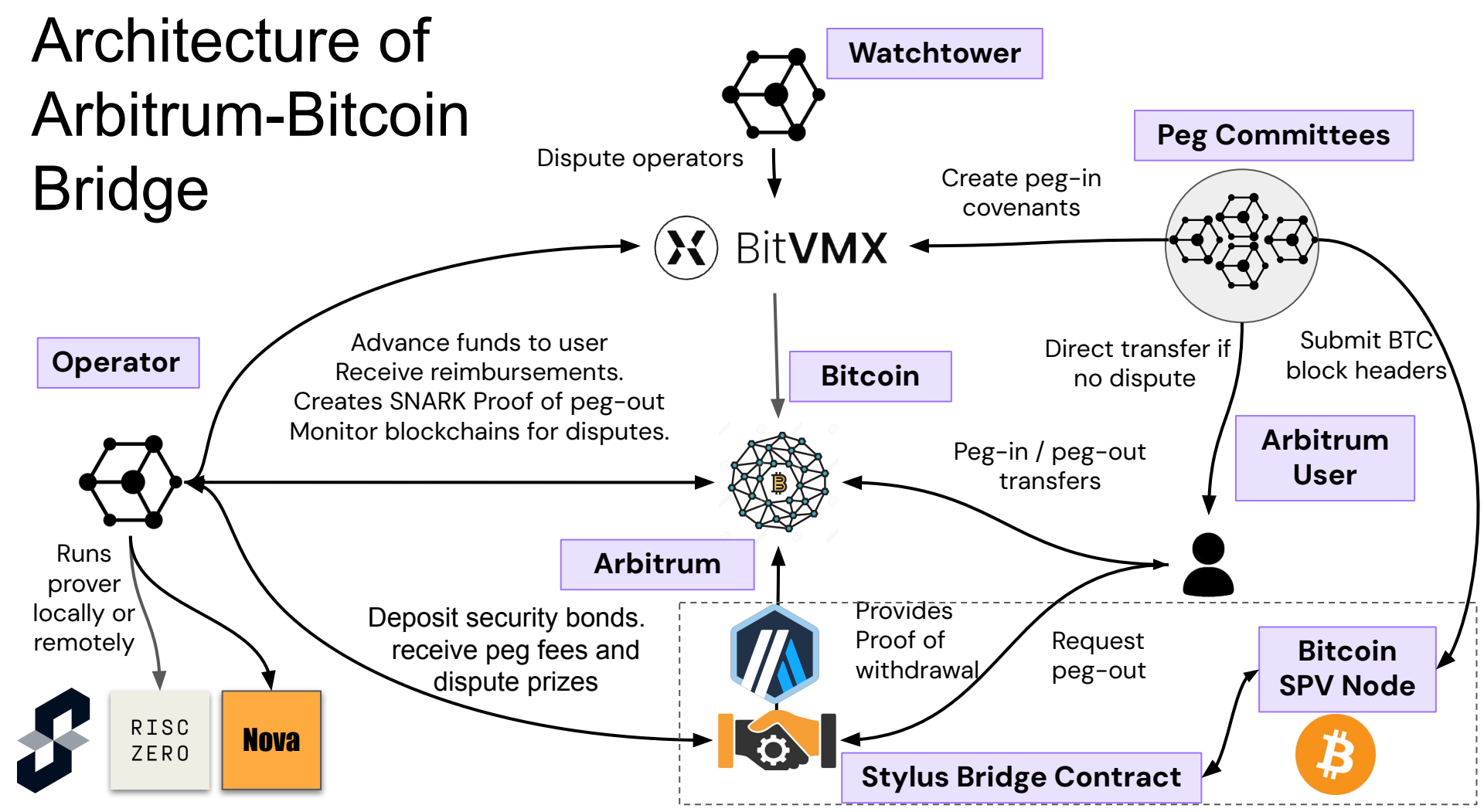
Winning assertion competes with challengers

BitVMX Bridge for Arbitrum using Stylus

FLEX

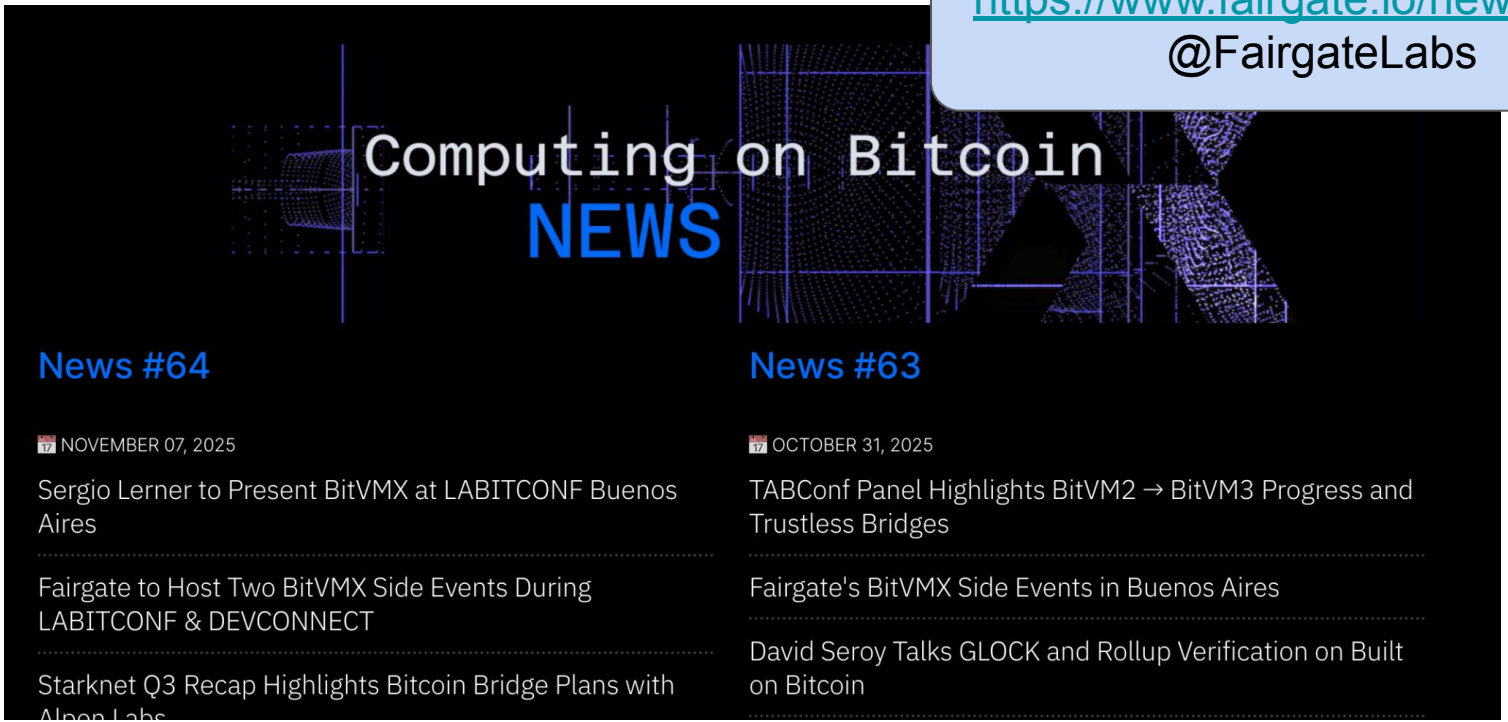
Fairgate

Architecture of Arbitrum-Bitcoin Bridge



BitVM/BitVMX in the news

https://www.fairgate.io/newsletters/
@FairgateLabs



Visit BitVMX.org
and Fairgate.io !

Q&A