DEVCON 8 · MUMBAI · 3–6 NOVEMBER 2026

# Your Device Doesn't Have to Trust Its Manufacturer

Ask the chain, not the supplier.

DON'T TRUST. VERIFY!

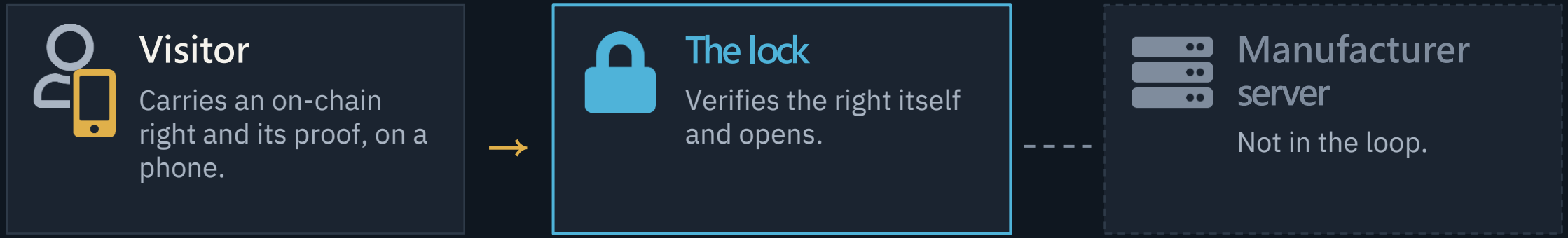


Jork Leonhardt · Co-CEO & Co-Founder, corpus.core

01 · THE LOCK

# The Lock That Opens Itself

A right on a blockchain, checked by the lock. No call to the manufacturer.



01 · THE LOCK

# The Lock That Opens Itself

A right on a blockchain, checked by the lock. No call to the manufacturer.

**Visitor**

Carries an on-chain right and its proof, on a phone.

→

**The lock**

Verifies the right itself and opens.




**slock.it, Devcon 1, London 2015: a smart lock controlled by a smart contract.**

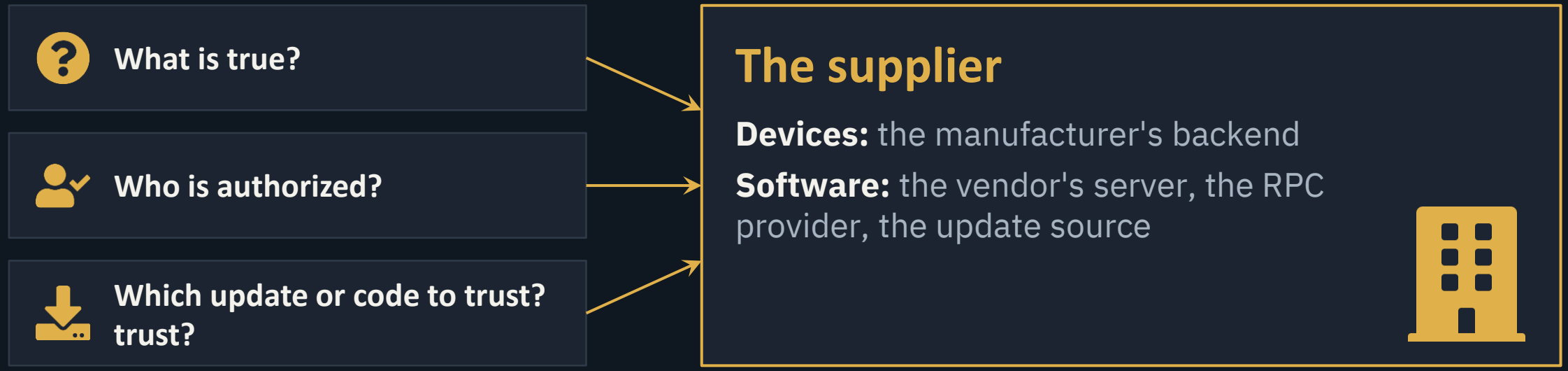
**Who here owns a device that needs the the manufacturer's server to work?**

**Who here owns a device that needs the the manufacturer's server to work?**

**And who here verifies what their RPC node tells them?**

# The Supplier Decides

Three decisions are made somewhere else




**A capture point inside decentralized systems.**

Manufacturers also provide cloud services, spare parts and repairs. This talk covers the three decisions.

# When the Vendor Holds the Key

Nobody attacked anything in these cases

| | Vendor | Device | When | What stopped working |
|---|---|---|---|---|
| | **The vendor decided who may use the device** | | | |
|  | **VanMoof** | E-bikes | July 2023 | Key in the vendor's app; repairs and spare parts stopped |
|  | **Fisker** | Electric cars | June 2024 | The app that unlocks the doors crashed |
|  | **Insteon** | Smart home hubs | April 2022 | Servers switched off without warning, hubs stopped |
| | **Where our approach ends** | | | |
|  | **Humane** | AI Pin | February 2025 | The AI service lived in the vendor's cloud |
|  | **Belkin Wemo** | Smart plugs | January 2026 | Cloud ended: remote access and voice control stopped |

**In all three, the decision who may use the device sat on the vendor's server.**

# Software Trusts Its Suppliers Too

Two cases of wrong data that was believed, one of a deceived signer

**NOVEMBER 2020**

## Infura

Provider on a minority chain. MetaMask broke, exchanges paused withdrawals.



**APRIL 2026**

## KelpDAO

About 292 million dollars. Poisoned RPC nodes, no contract bug found.



**FEBRUARY 2025**

## Bybit

About 1.5 billion dollars. Safe{Wallet} infrastructure compromised, signers deceived.



*Verifying chain data does not reach this one. The next talk does.*

**Infura and KelpDAO: wrong data that was trusted.**

# DON'T TRUST.
# VERIFY!

04 · THE PRINCIPLE

# Two Steps Back to the Consumer

Take the decision away from the supplier

**1** **Truth in public state**

The right, the state or the reference is written into public, verifiable state. For a lock: a smart contract. For software: the chain itself.



**2** **The consumer verifies**

Data and proof arrive from any source: phone, gateway, RPC node, USB stick. Anything that does not verify is rejected.



**Transport gives availability. Proofs give correctness.**

Something still delivers the data: no trusted backend, not no backend.

05 · THE TRUST ANCHOR

# One Assumption Remains

Ethereum's consensus cannot be verified from nothing




## One assumption remains: a recent checkpoint.

Much smaller than a server that answers every question. Anyone can compare it against other sources.

# Verification as a Component

Stateless: no full node, no persistent state, no background sync



**Access**



**Pay**



**Update**

**SDK · local verification**

**Consensus proof:** the block is in the finalized chain, down to a ZK sync proof of ≈350 bytes

**Execution proof:** a balance, storage slot or event belongs to that block



**Software**

Wallets, bridges and backends verify what their RPC returns.



**Devices**

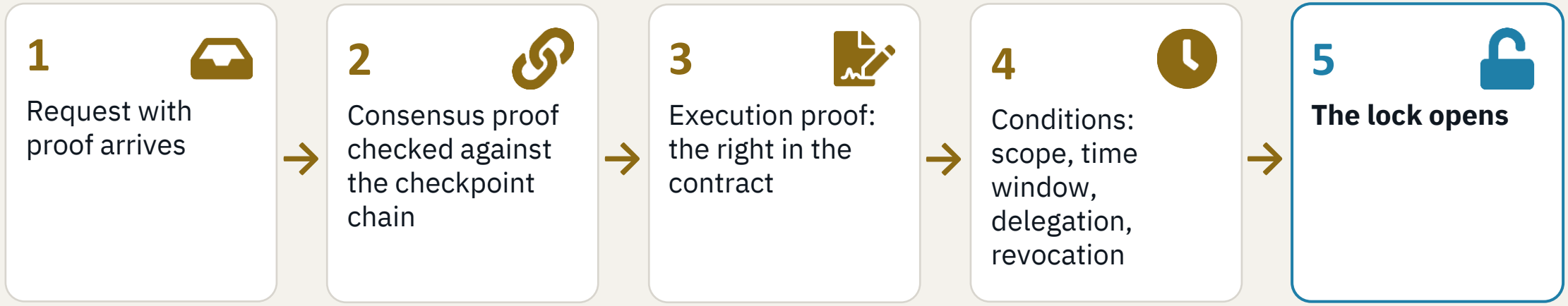
Runs on an ESP32: ≈256 kB flash, 128 kB RAM to prove one transaction.

## The check: a few hashes and one signature.

colibri.stateless · open source (MIT) · Ethereum, Gnosis, OP Stack

# Access, Step by Step

Back to the lock




**If any step fails, nothing happens. No server decides.**

07 · ACCESS

# When the Vendor Is Gone

Replay VanMoof

### Today

The key lives in the vendor's app, on the vendor's server. Insolvency decides who may ride.



### With an on-chain right

The right lives in a contract the owner controls. Insolvency changes nothing.



**Guest access with a time window**

**Delegation and revocation**

**Offline use, proofs arrive later**

**The vendor is no longer part of the decision.**

08 · PAY

# Pay Without a Billing Backend

Has this payment really happened?

### A device with its own wallet

Keys stay on the device. Mainly stablecoins, so no exchange rate risk.



**Verified** Balance and settlement checked with proofs

**Limits** How much, how often, for what

**Use cases** Pay per use, time based billing, agents that pay

**The device verifies its own settlement. No billing backend.**

09 · UPDATE

# Which Code to Trust

Firmware and any software artifact with a hash

**Contract**

Approved hashes with conditions: hardware, version, device class



**Any source**

Vendor server, peer, gateway, USB stick



**Consumer**

Hashes the package, compares with the verified hash, checks eligibility



**Install or reject**

Nothing unverified runs



**Transport protects delivery. The contract decides what is legitimate.**

# The Same Three Questions Everywhere

Is this right valid? Was this payment settled? Is this update legitimate?

- Machines operated by different companies
- Shared charging infrastructure
- Robots
- Agents that act on hardware hardware
- **Software that acts on chain data**
- Agents that pay

**Software that acts on chain data starts with the wallet: the next talk.**

11 · THE LIMITS

# What This Does Not Solve

Better named here than found later

### Not solved

- Cloud services the vendor runs
- Spare parts and repairs
- What a signer is shown (next talk)
- Tampered hardware (Ledger, October 2026)
- Who publishes updates after the vendor is gone

### Limits

- No real time control loops or high frequency sensor data on chain
- The checkpoint remains an assumption
- Proofs cost bandwidth: ≈28 kB for an answer of a few bytes

**Verification protects a device from wrong data, not from tampered hardware.**

# Three Things to Take Home

1. You should not have to trust the manufacturer of your device or the vendor of your software your software for what is true, who may use it and what code runs.

2. Put the truth into public state, and let the consumer verify it.

3. Know the limits.

**Ask the chain, not the supplier.**

# DON'T TRUST.
# VERIFY!

**Ask the chain, not the supplier.**

**repository.colibri-stateless.tech**

Jork Leonhardt · Co-CEO & Co-Founder, corpus.core

BACKUP A

# Who Sets the Checkpoint?

Weak subjectivity checkpoint: a recent block whose sync committee is assumed correct

Public chain data, comparable against other sources

How it gets onto the device: the original checkpoint would be put on the device by the manufacturer installing colibri, the device would be able to get a current checkpoint from public APIs (e.g. checkpointz)

From there the device checks every committee change itself or via a zk proof with the aggregation of a long chain of committee changes

BACKUP B

# Offline Operation and Proof Sources

The source is interchangeable, because the device checks the proof and not the sender

Proof sources: local generator, prover service, untrusted RPC

Every response carries its proofs for consensus and execution

Decisions can wait until a valid proof is available

BACKUP C

# Supported Chains

The on-chain right has to live on one of these

In production: Ethereum, Gnosis, OP Stack (Base, OP Mainnet)

Planned: more EVM and non-EVM chains

A chain needs provable consensus and authenticated state