A Touch of Evil:

Cryptographic Hardware from Untrusted Components

Vasilios Mavroudis
Doctoral Researcher, UCL

BackdoorTolerance.org

Who we are

Vasilios Mavroudis
Doctoral Researcher, UCL

Andrea Cerulli
Doctoral Researcher, UCL

Dusan Klinec
Enigma Bridge

Petr Svenda
Assistant Professor, MUni
CTO, Enigma Bridge

Dan Cvrcek
CEO, Enigma Bridge

George Danezis
Professor, UCL

The Private Life of Keys

1. Someone designs an integrated circuit (IC)
2. IC is fabricated
3. IC is delivered to hardware vendor
4. Vendor loads firmware & assembles device
5. Device is sent to customer
6. Customer generates and stores key on the device
7. Keys are used to decrypt, sign, authenticate

The Private Life of Keys

1. Someone designs an integrated circuit (IC)
2. IC is fabricated
3. IC is delivered to hardware vendor
4. Vendor loads firmware & assembles device
5. Device is sent to customer
6. Customer generates and stores key on the device
7. Keys are used to decrypt, sign, authenticate

Any attack in these steps can compromise the key!

Hardware Security Modules

Physical computing device that safeguards and manages digital keys for strong authentication and provides cryptoprocessing.

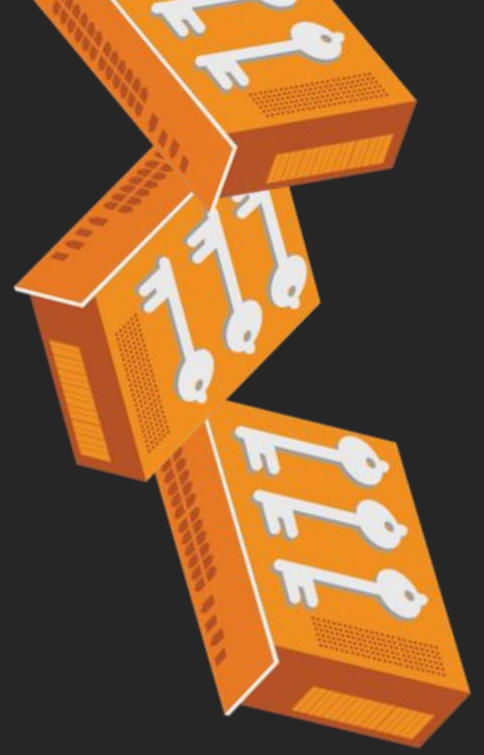
Features:

- Cryptographic key generation, storage, management
- Tamper-evidence, Tamper-resistance, Tamper-response
- Security Validation & Certification

Crypto Operations are carried out in the device

No need to output the private keys!

Hardware Security Modules



Common Applications

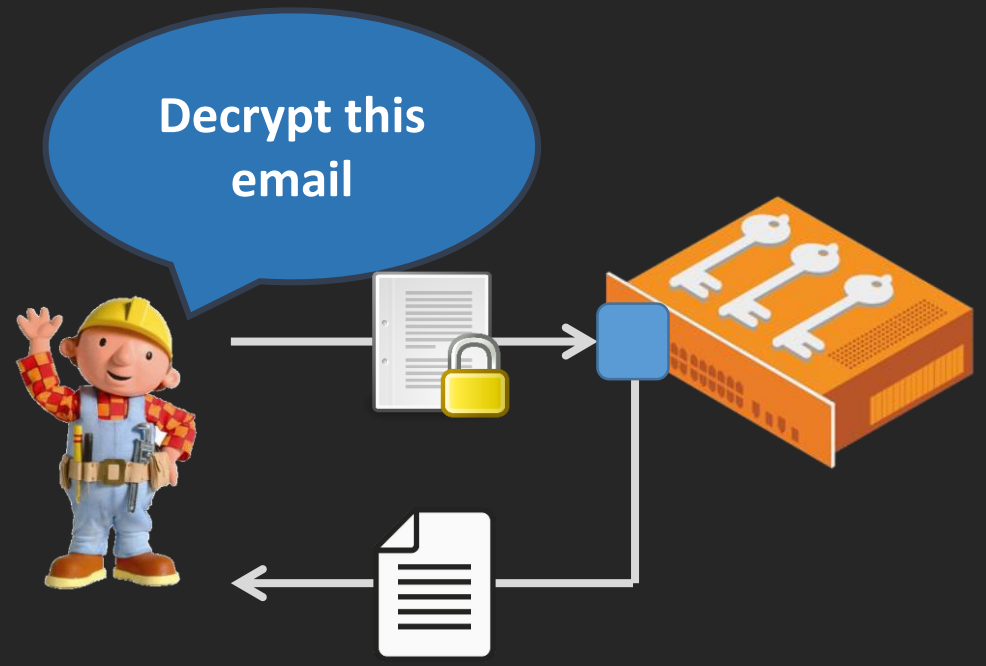
- Public Key Infrastructures
- Payment Processing Systems
- SSL Connections
- DNSSEC
- Transparent Data Encryption

Cost

- Hardware (>$10k)
- Integration Cost
- Operational/Support

HSM Simple Example

1. Bob uses a desktop app to request the decryption or signing of a document
2. The app interacts with the HSM through a crypto API (e.g., PKCS#11)
3. The HSM processes the request
4. Bob retrieves the request outcome



HSM Guarantees

1. Someone designs an integrated circuit (IC)
2. IC is fabricated
3. IC is delivered to hardware vendor
4. Vendor loads firmware & assembles device
5. Device is sent to customer
6. Customer generates and stores key on the device
7. Keys are used to decrypt, sign, authenticate

What could go wrong?

- Bugs

CVE-2015-5464
The HSM allows remote authenticated users to bypass intended key-export restrictions …

- Backdoors/HT?

THIS 'DEMONICALLY CLEVER' BACKDOOR HIDES IN A TINY SLICE OF A COMPUTER CHIP

NSA's Own Hardware Backdoors May Still Be a "Problem from Hell"

Expert Says NSA Have Backdoors Built Into Intel And AMD Processors

Snowden: The NSA planted backdoors in Cisco products

What could go wrong?

- Bugs

5-5464

The HSM allows remote authenticated users to bypass intended key export restrictions …

- Backdoors?

THIS REMOTELY CLEVER BACKDOOR DESCRIBED A TINY PIECE OF A COMPUTER CHIP

NSA's Own Hardware Backdoors May Still Be a "Problem from Hell"

Expert Says NSA Have Backdoors Built Into Intel And AMD Processors

Snowden: The NSA planted backdoors in Cisco products

ROCA

Proposed Solutions

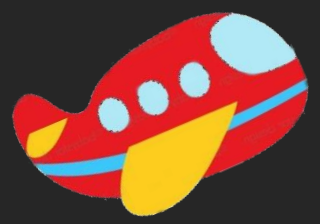
- Trusted Foundries
  - Very expensive
  - Prone to errors/bugs
- Split-Manufacturing
  - Still Expensive
  - Prone to errors/bugs
- Post-fabrication Inspection
  - Expensive (+ re-tooling)
  - A huge pain, doesn’t scale

Proposed Solutions

- Trusted Foundries
  - Very expensive
  - Prone to errors/bugs
- Split-Manufacturing
  - Still Expensive
  - Prone to errors/bugs
- Post-fabrication Inspection
  - Expensive (+ re-tooling)
  - A huge pain, doesn't scale

Arms Race

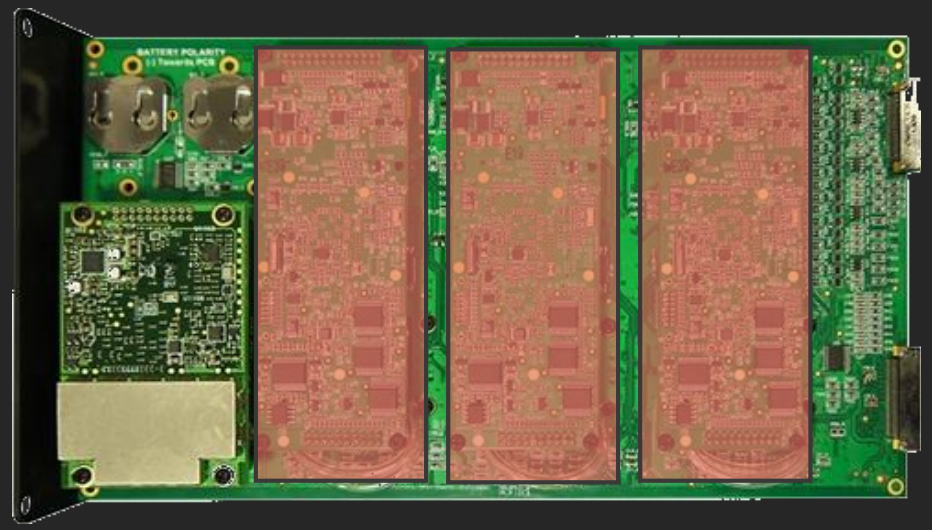
- Adversaries always one step forward
- Can never be 100% certain



A solution from the sky (not the cloud)

***Lockstep systems** are fault-tolerant computer systems that run the same set of operations at the same time in parallel.*

- Dual redundancy
  allows error detection and error correction
- Triple redundancy
  automatic error correction, via majority vote
  → Triple Redundant 777 Primary Flight Computer



Confidentiality of Keys?

Fault-tolerant systems are built for safety
and the computations are simply replicated.

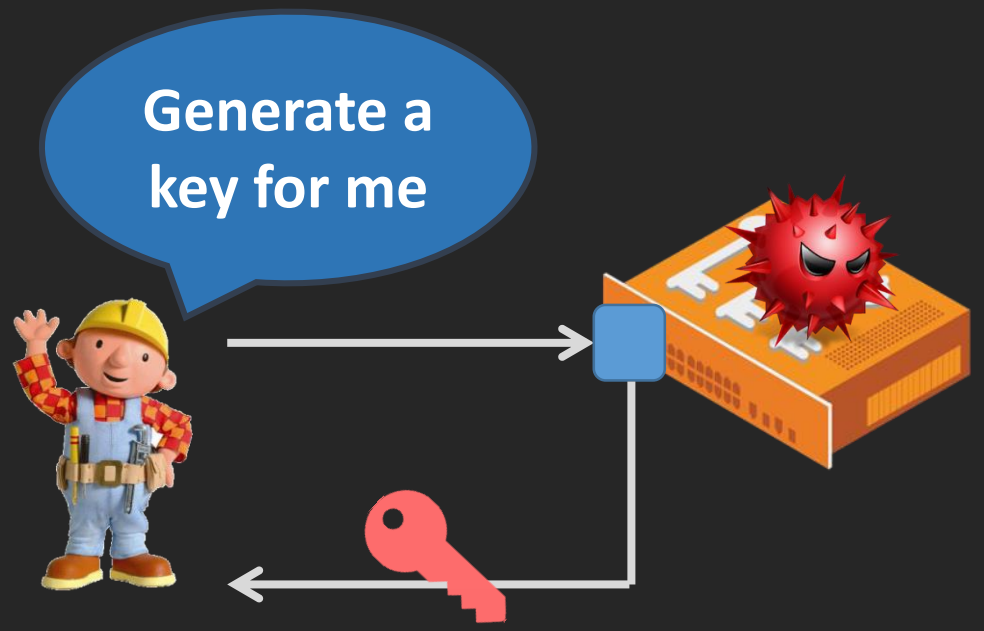
Fault-tolerant systems won't cut it:

- Increase the attack surface
- The private key is stored in each IC
- Device is as secure as its weakest link

Our Solution

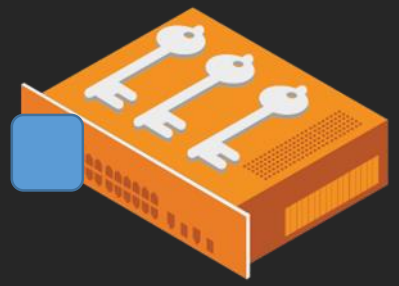
1. Someone designs an integrated circuit (IC)
2. IC is fabricated
3. IC is delivered to hardware vendor
4. Vendor loads firmware & assembles device
5. Device is sent to customer
6. Customer generates and stores key on the device
7. Keys are used to decrypt, sign, authenticate

Threat Model



Threat Model

Threat Model



Ingredients of the Solution

1. Hardware Components (IC)
   - Independent Fabrication
   - Non-overlapping Supply Chains
   - Programmable
   - Affordable
   - Bonus: commercial off-the-shelf

2. Cryptographic Protocols
   - No single trusted party
   - Full Distribution of Secrets
   - Distributed Processing
   - Provably Secure

Smart Cards

Many Independent Manufacturers

- Private Fabrication Facilities
- Disjoint Supply Chains (location, factories, design)

Programmable Secure Execution Environment

- NIST FIPS140-2 standard, Level 4
- Common Criteria EAL4+/5+

Off-the-shelf Cost $1-$20

Multiparty Computation Protocols

Distributed Operations

- Random number Generation
- Key Pair Generation
- Decryption
- Signing

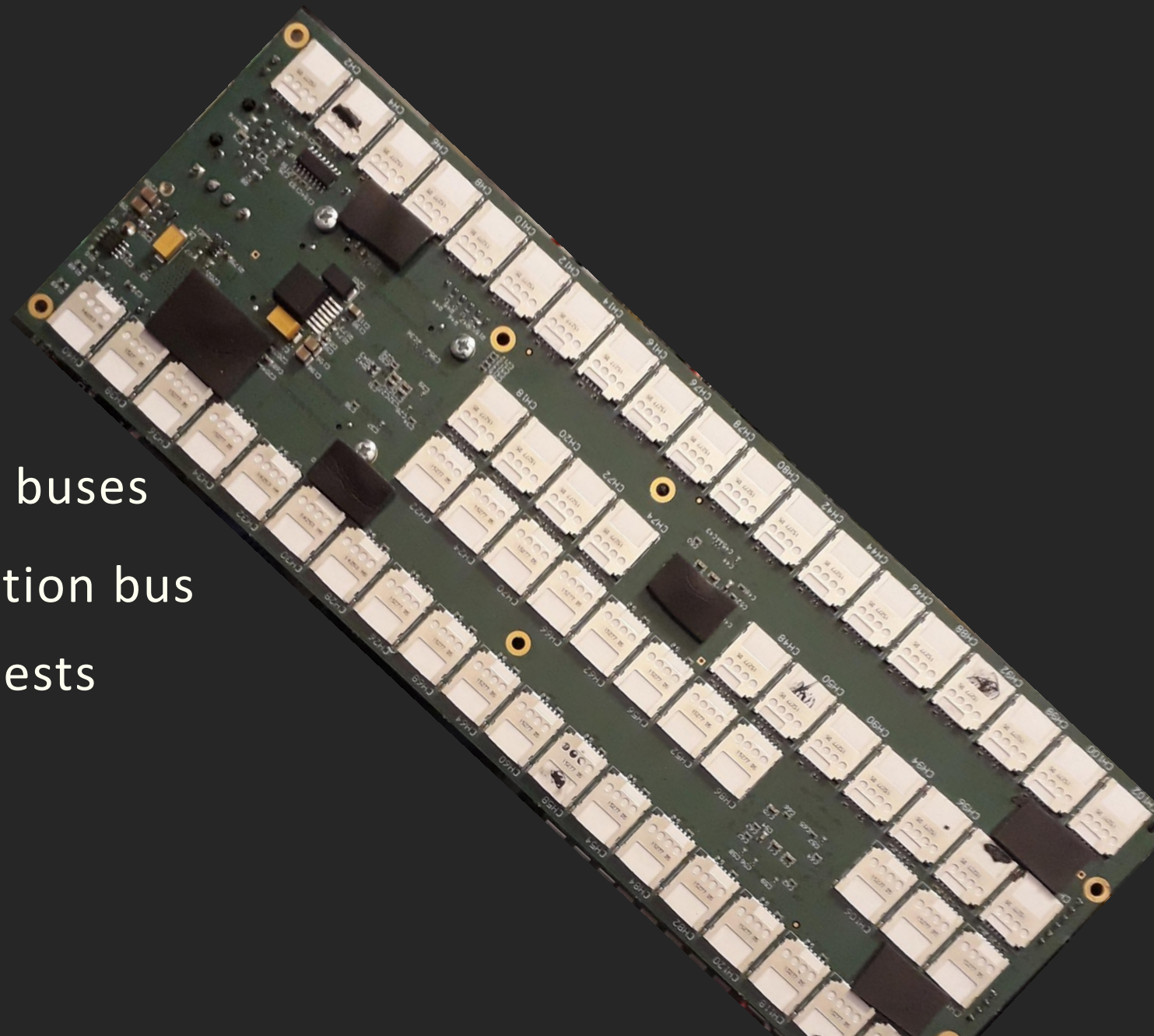
Provably Secure against

- All-1 Malicious & Colluding parties

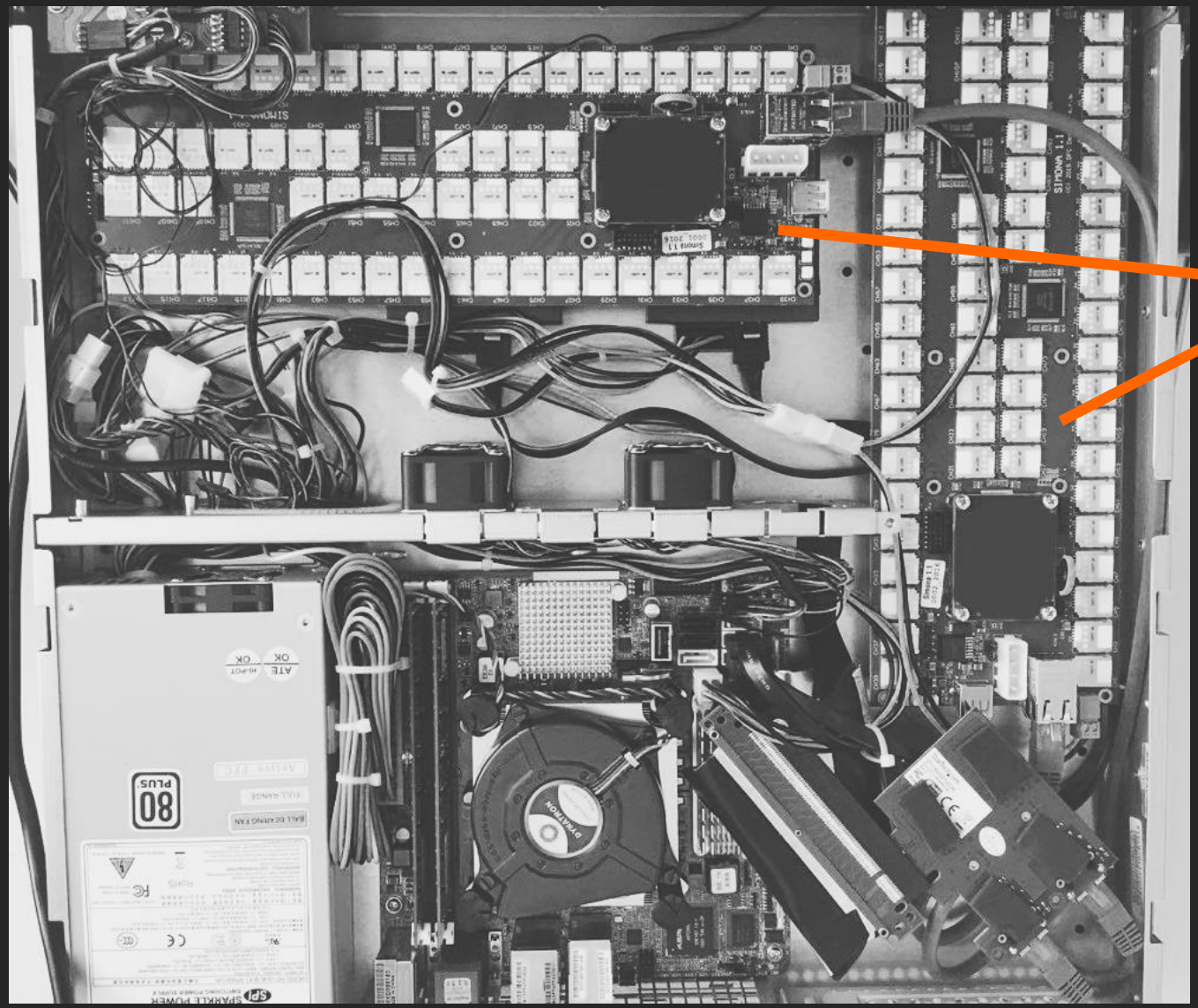
THE PROTOTYPE

Components

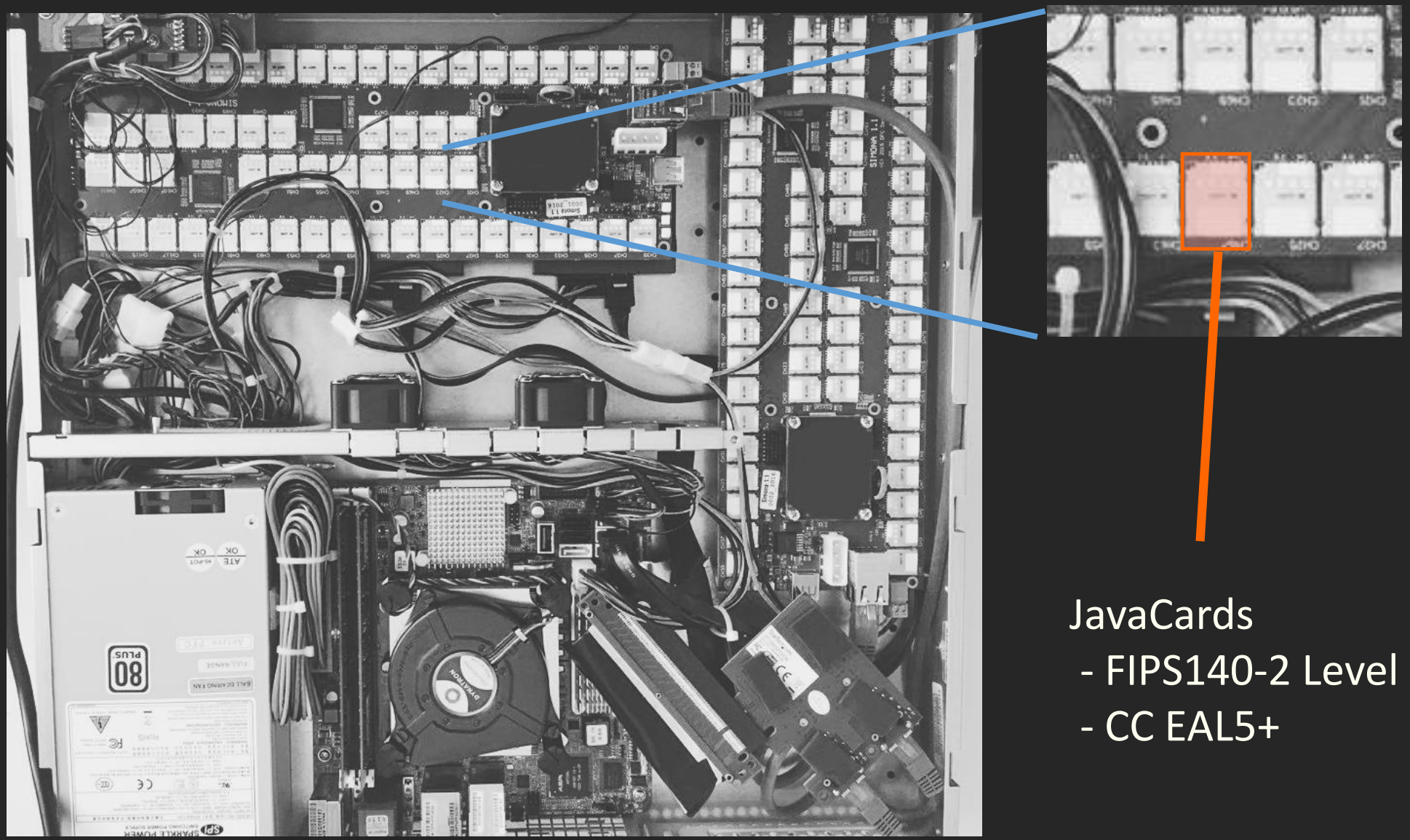
- 120 SmartCards
  - 40 quorums of 3 Cards
  - 1.2Mbps dedicated inter-IC buses
- FPGA manages the communication bus
  - 1Gbit/s bandwidth for requests



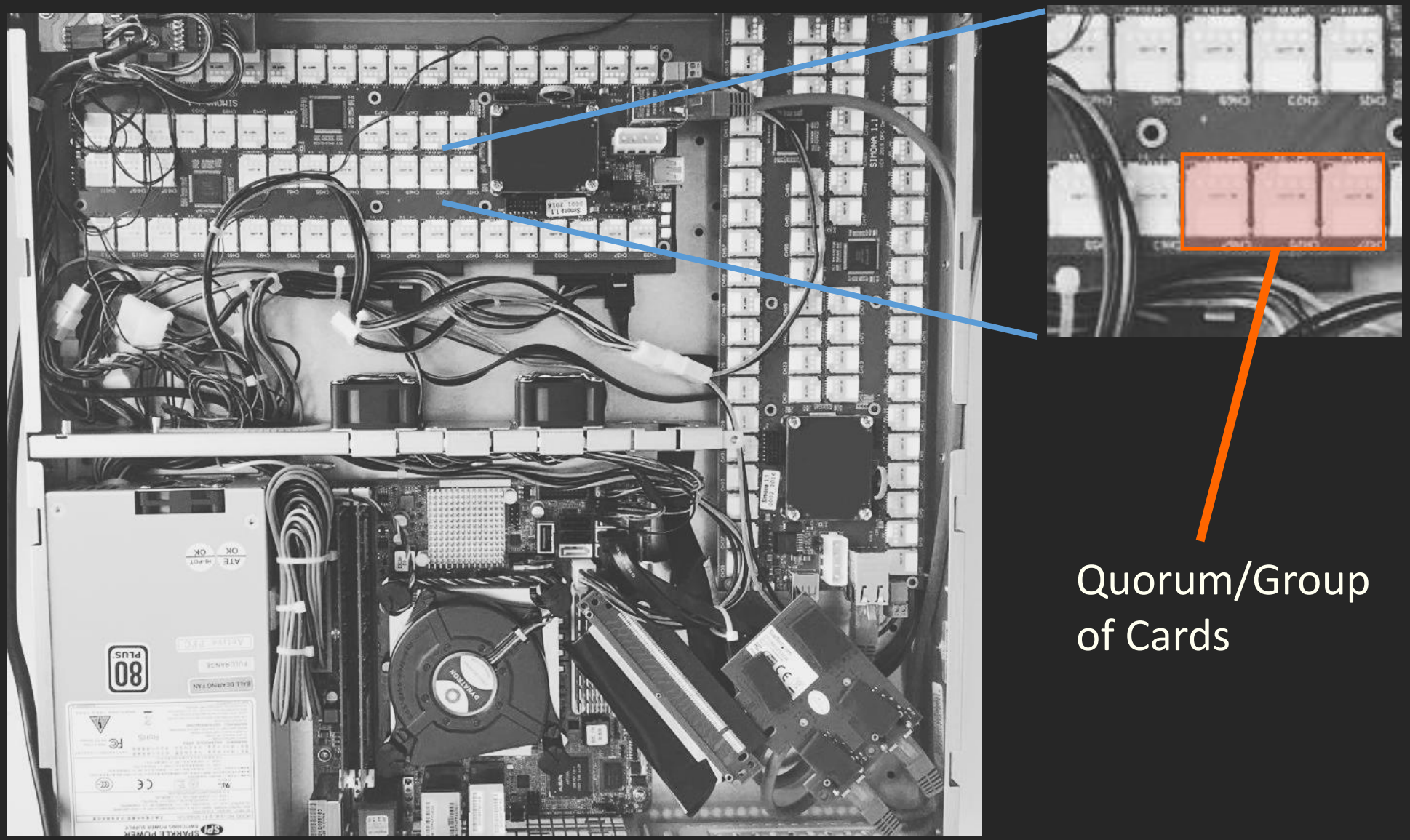
SIMONA 1.1
SIMONA 1.1
SPARKLE POWER
80 PLUS
DYNATRON

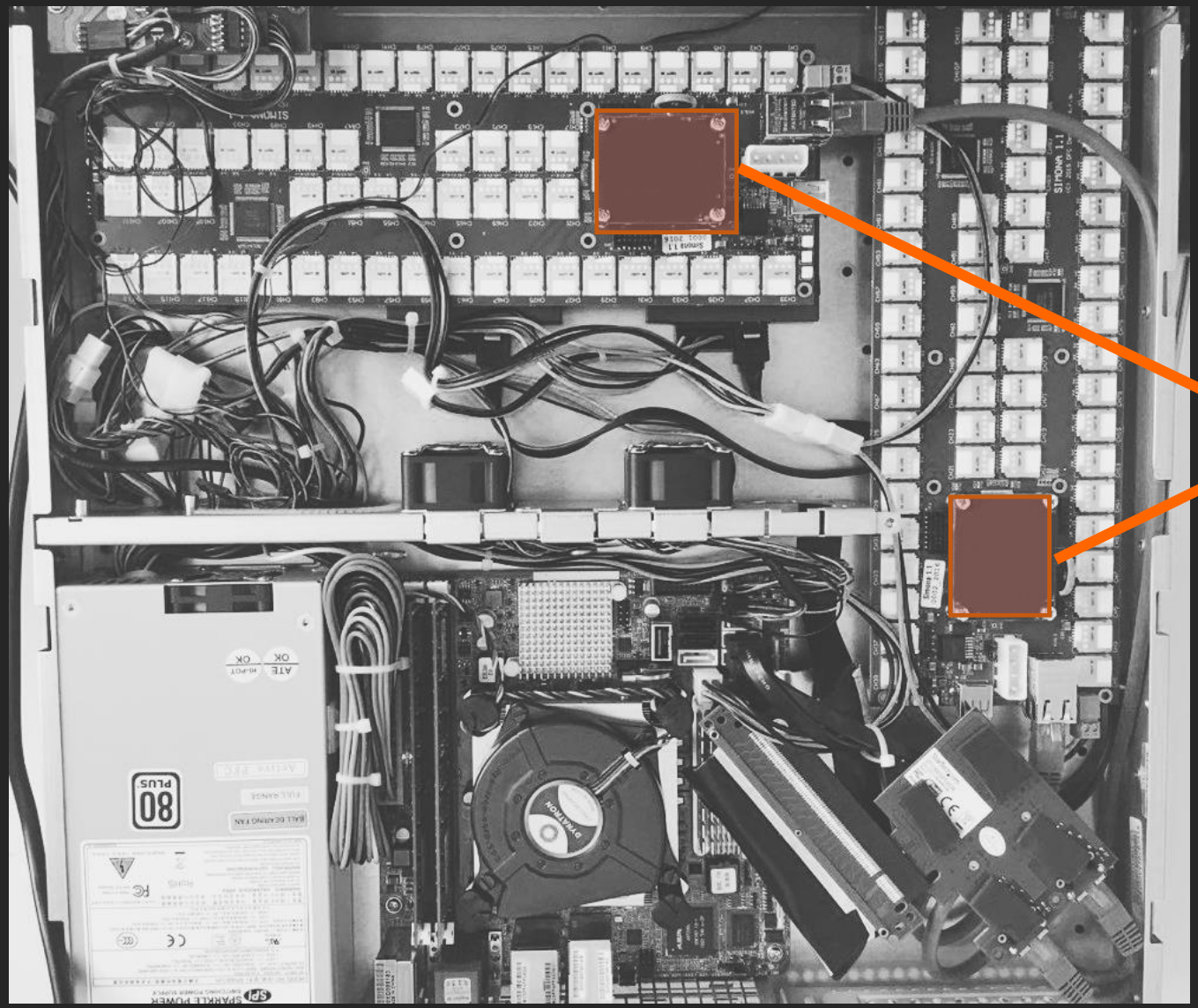


Custom boards
with 120 JCs



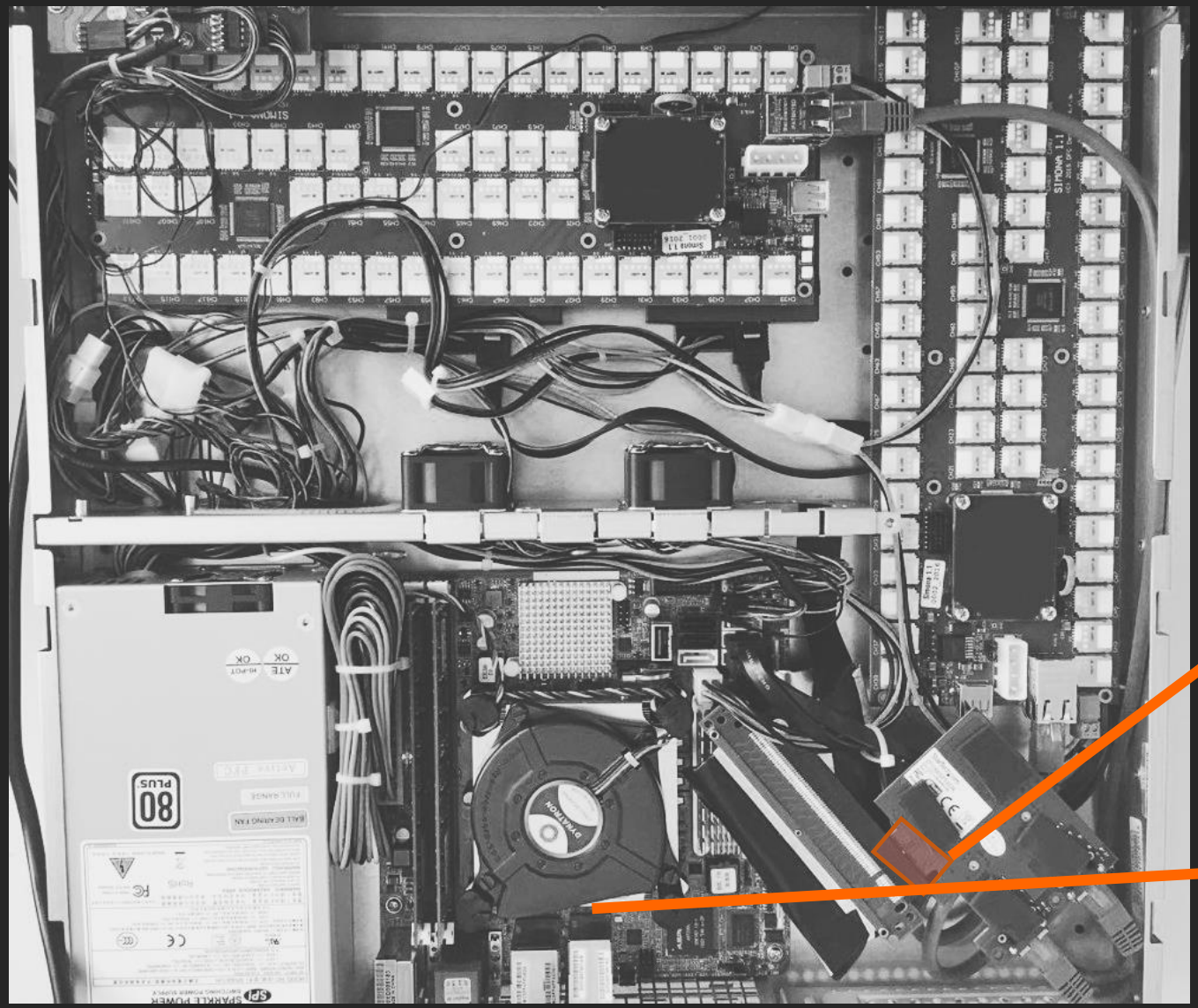
JavaCards
- FIPS140-2 Level 3
- CC EAL5+


Quorum/Group
of Cards



FPGA

JavaCard→TCP



Gigabit link to untrusted

Linux server

DEMO

PROTOCOLS

Sharing a Secret

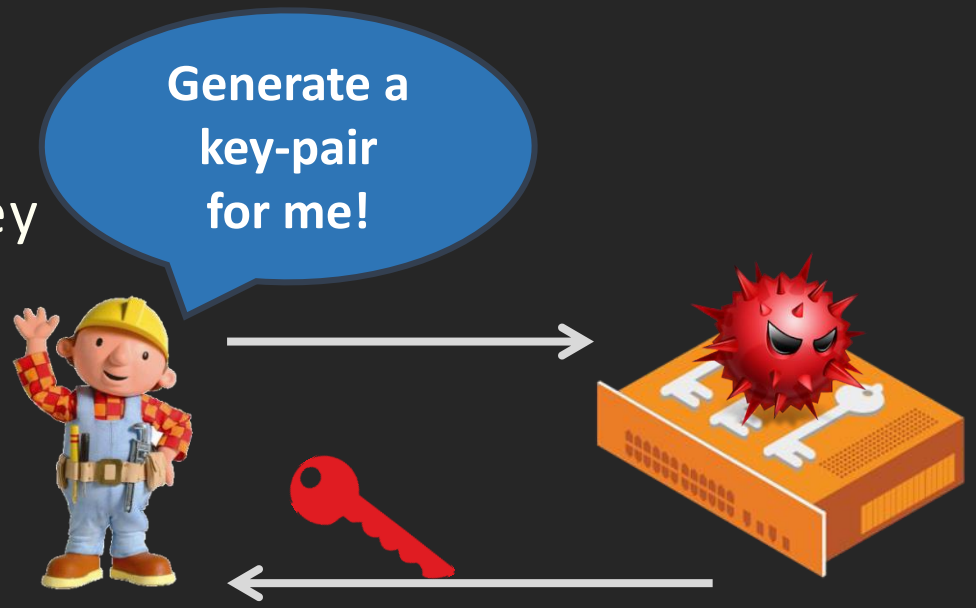
- Split a secret in *shares* and later *reconstruct* it
- Splitting Parameters:
  - How many shares the secret is split into (n)
  - How many shares you need to reconstruct the secret (t)
- Without *sufficient* shares not a single bit is leaked

In our case: One secret is split and stored in 3 smartcards

Classic Key Generation

Single IC System

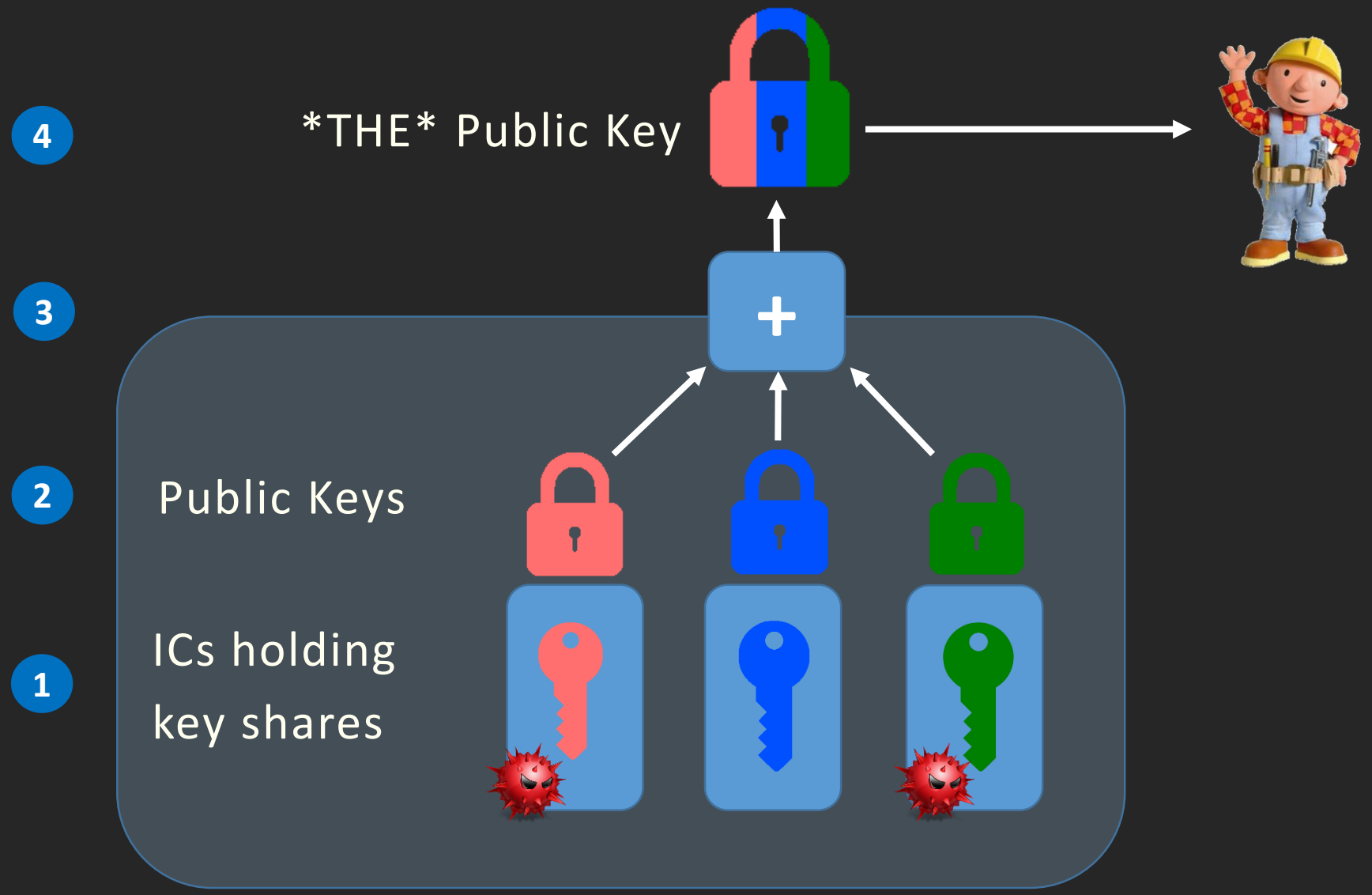
1. Bob asks for new key pair
2. Backdoored IC generates compromised key
3. Private Key is “securely” stored
4. Weak Public key is returned



Problems

- Malicious IC has full access to the private key
- Bob can’t tell if he got a “bad” key

Distributed Key Generation

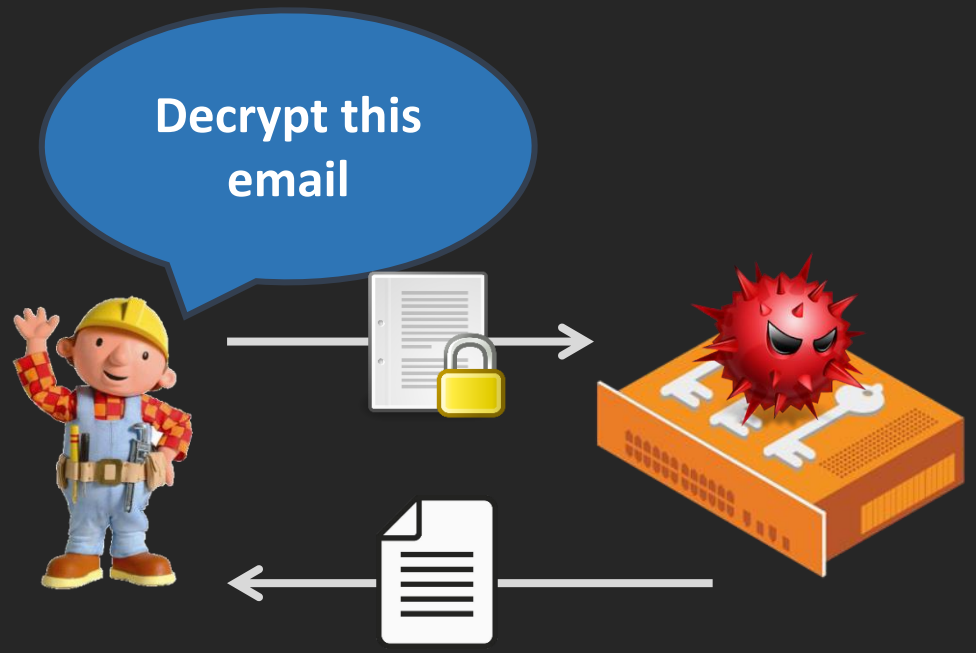


Distributed Key Generation

Classic Decryption

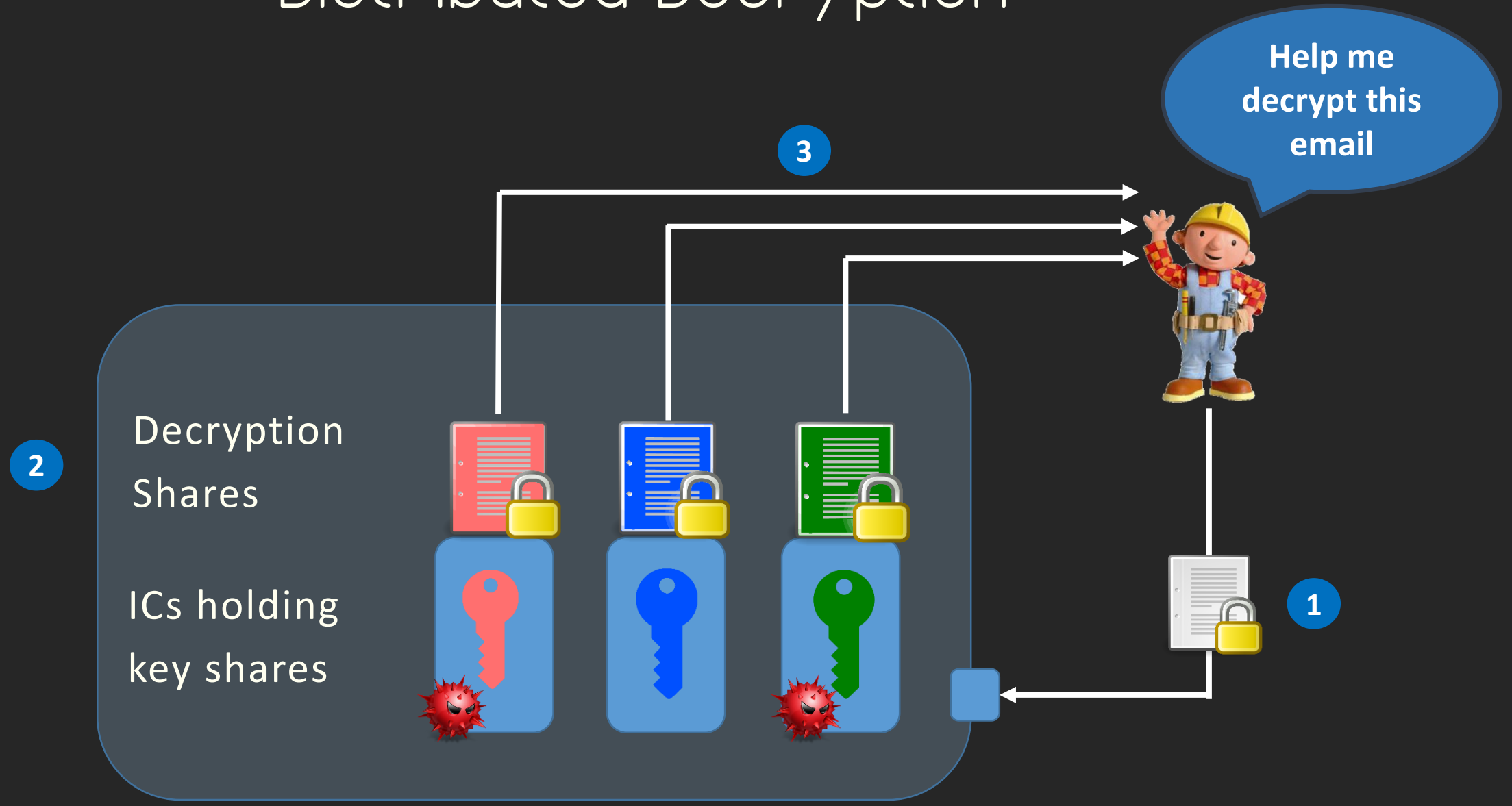
Single IC System

1. Bob asks for ciphertext decryption
2. Backdoored IC decrypts ciphertext
3. Bob retrieves plaintext

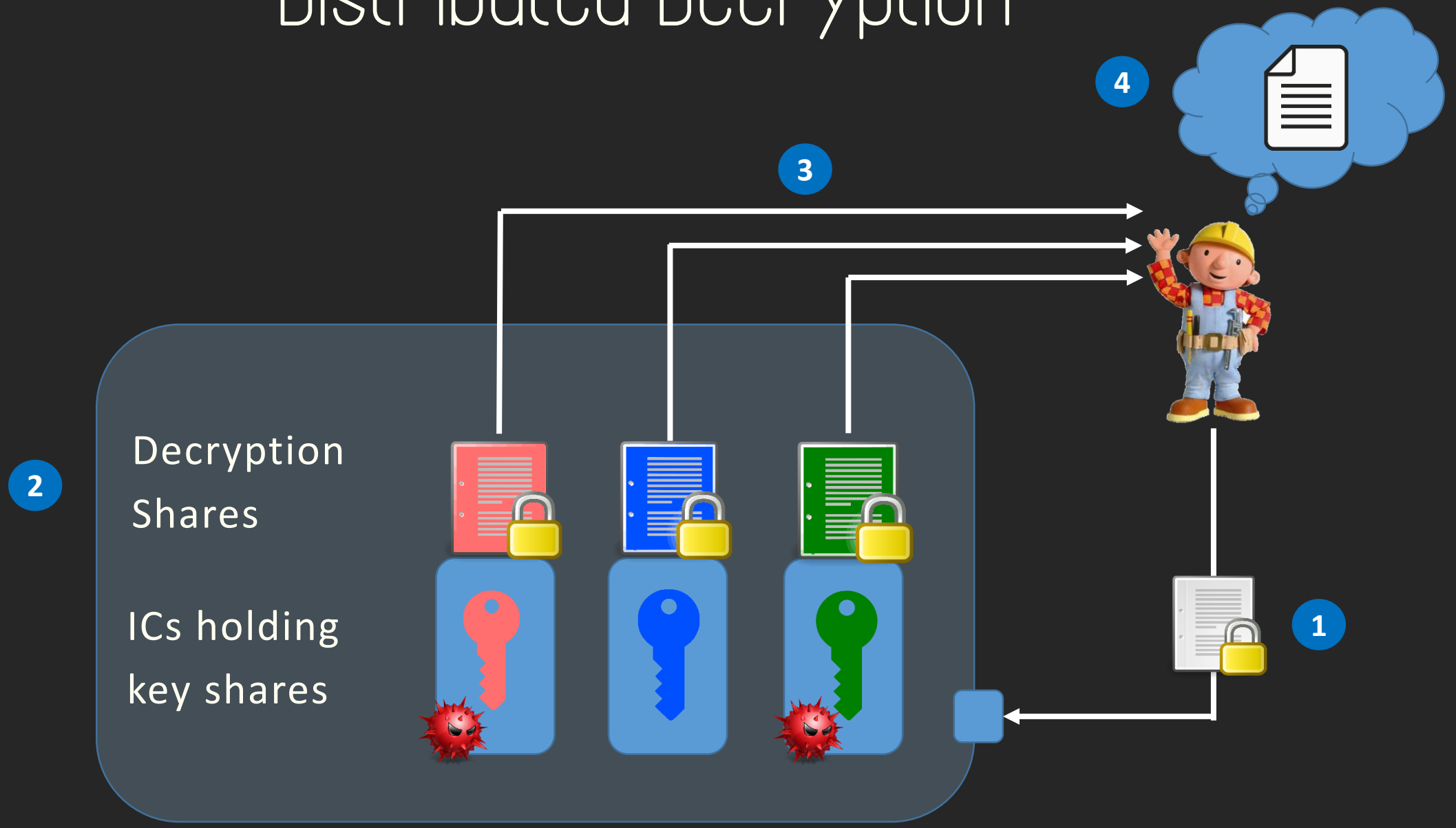


The IC needs full access to the private key to be able to decrypt ciphertexts.

Distributed Decryption



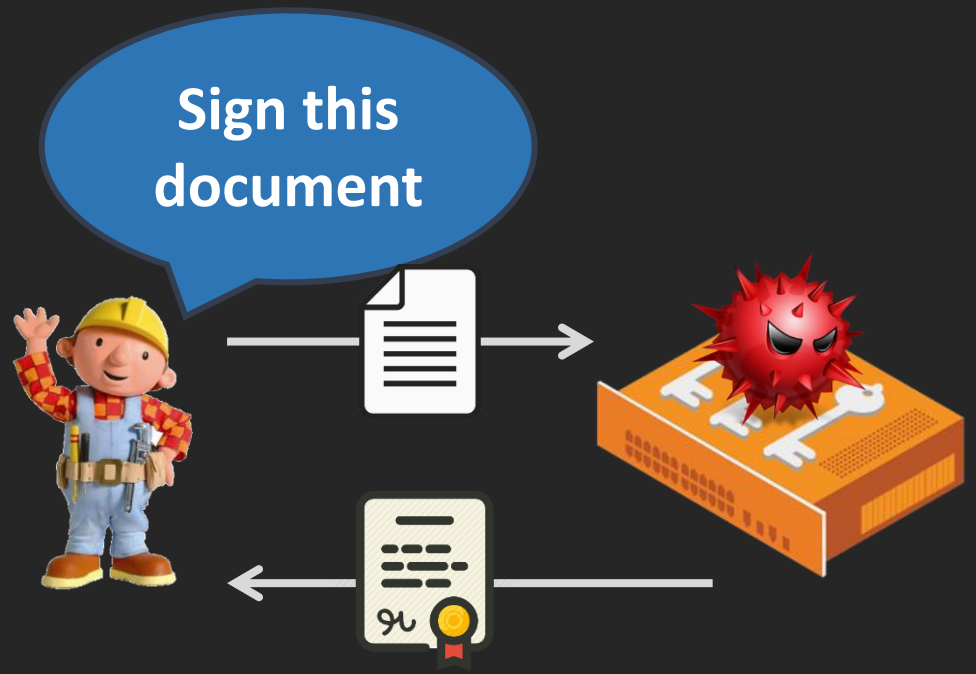
Distributed Decryption



Classic Signing

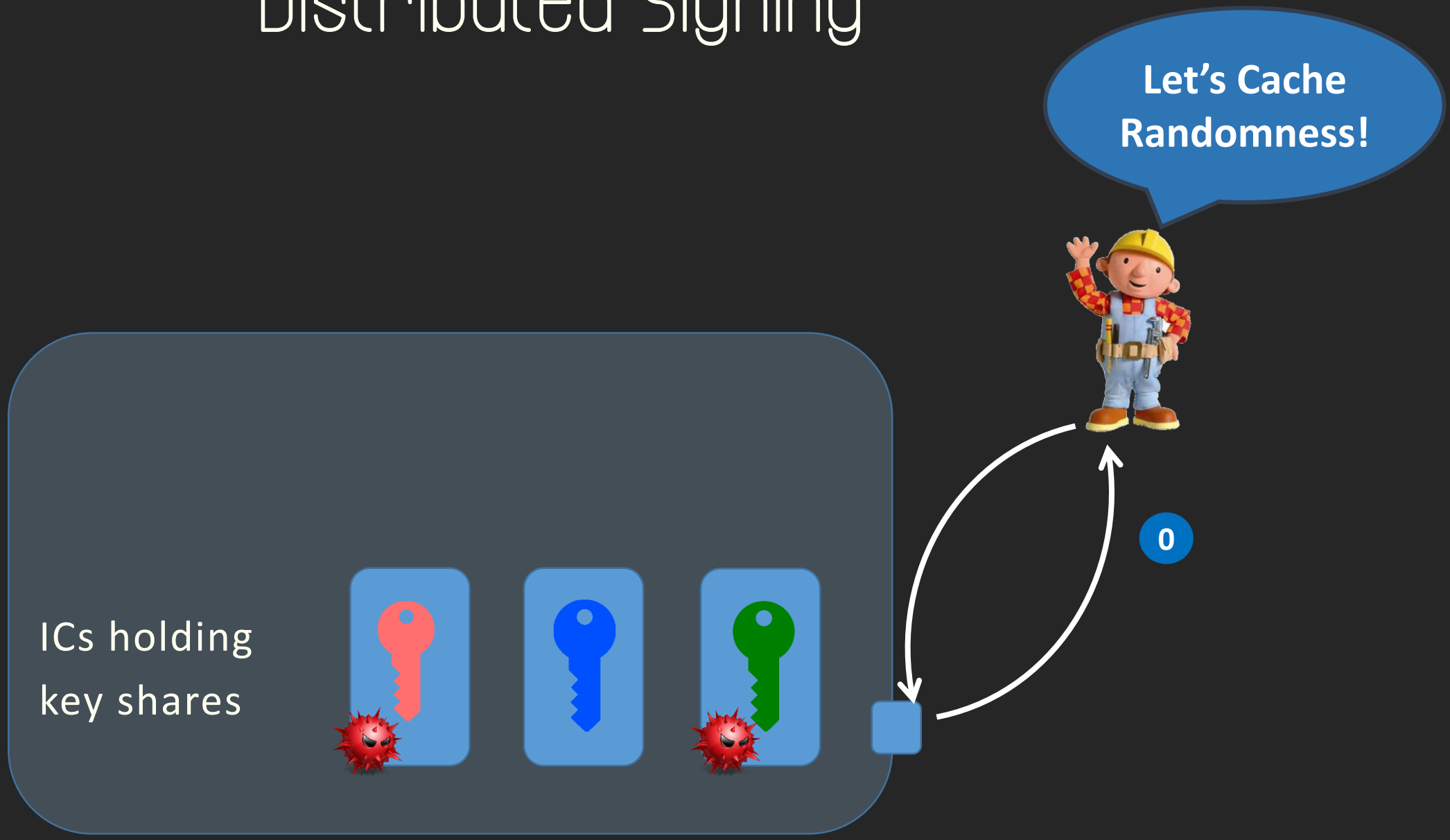
Single IC System

1. Bob asks for document signing
2. Backdoored IC signs the plaintext
3. Bob retrieves signature

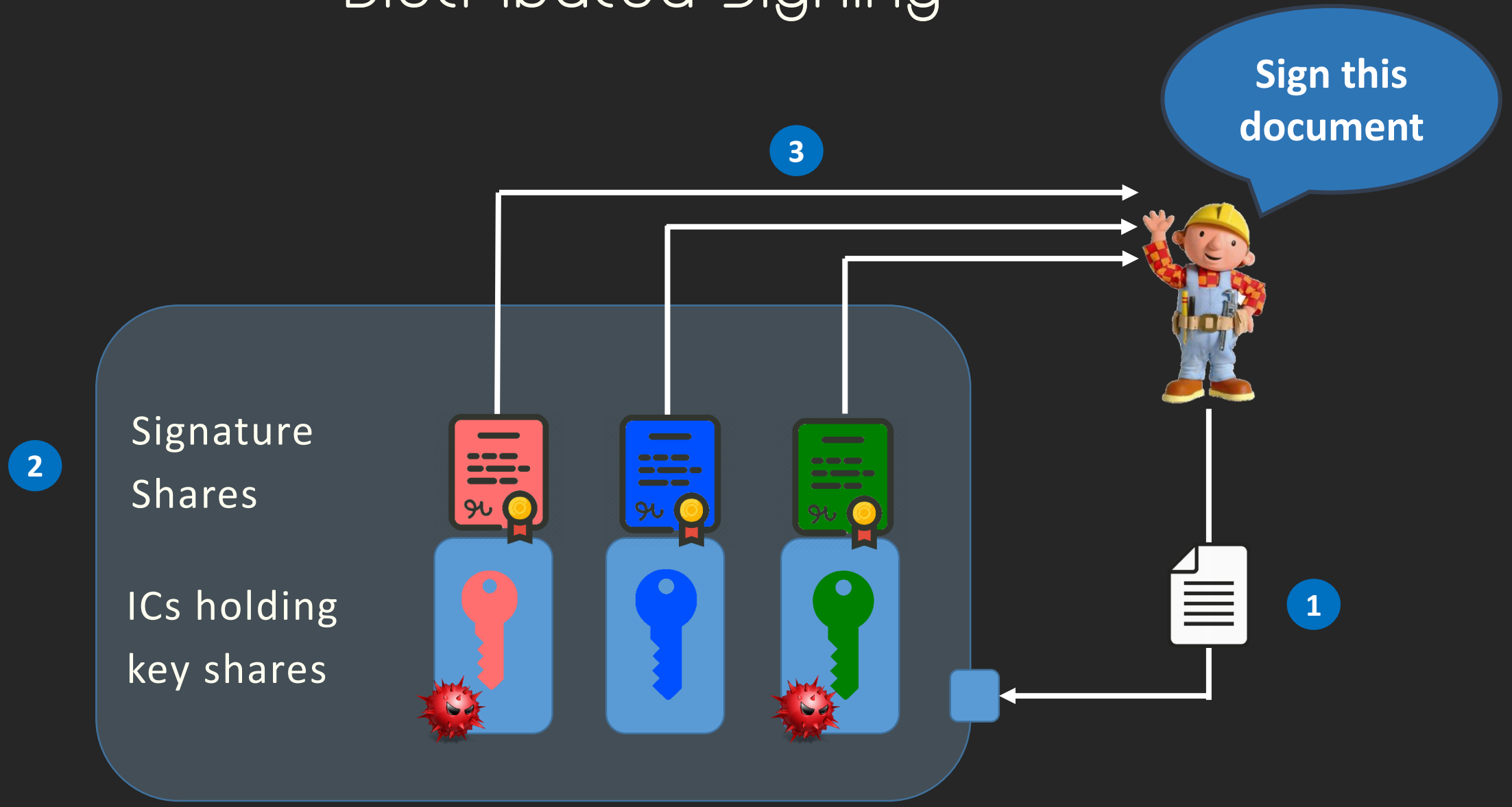


The IC needs full access to the private key to be able to sign plaintexts.

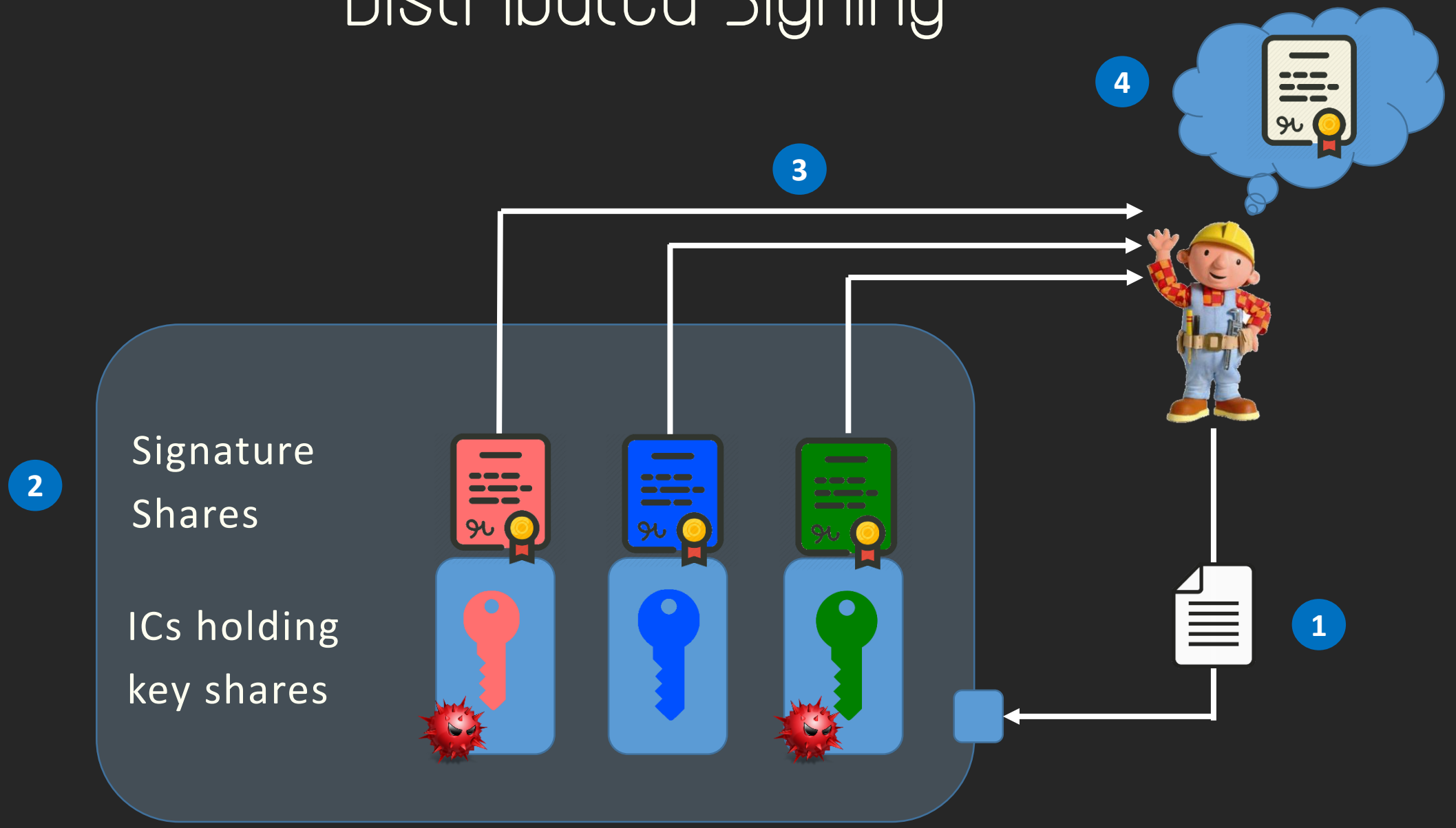
Distributed Signing



Distributed Signing



Distributed Signing



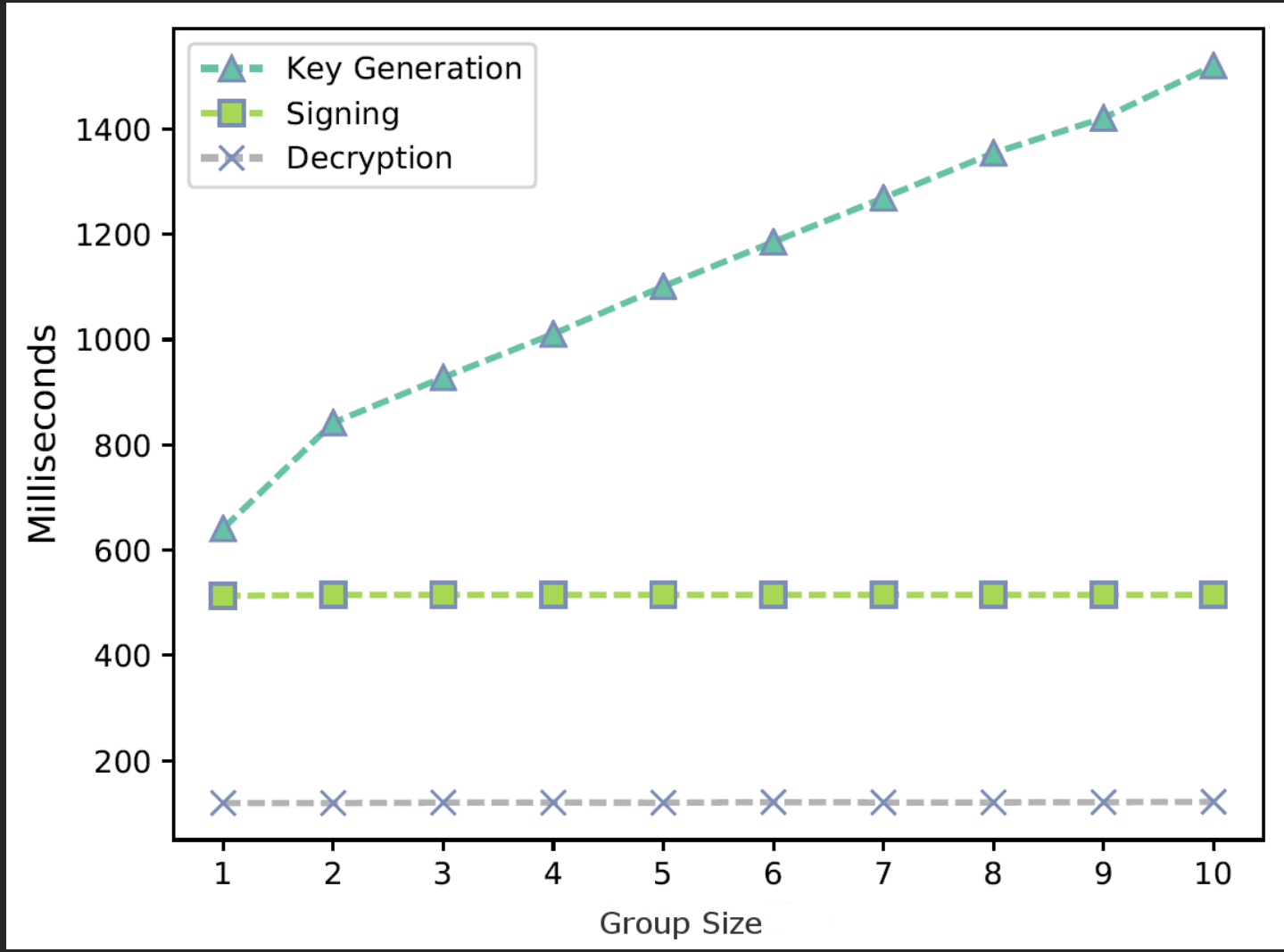
Technical Details

- Elliptive Curve Crypto (SecP256r1)
- El-Gamal for Encryption; a Schnorr variant for signing
- JavaCards are peculiar:
  - → They feature an ECC-capable crypto coprocessor
  - → Devs have access only to high-level methods (e.g., ECDSA)
  - → We had to build *JCMathLib*

OpenCryptoJC.org

PERFORMANCE

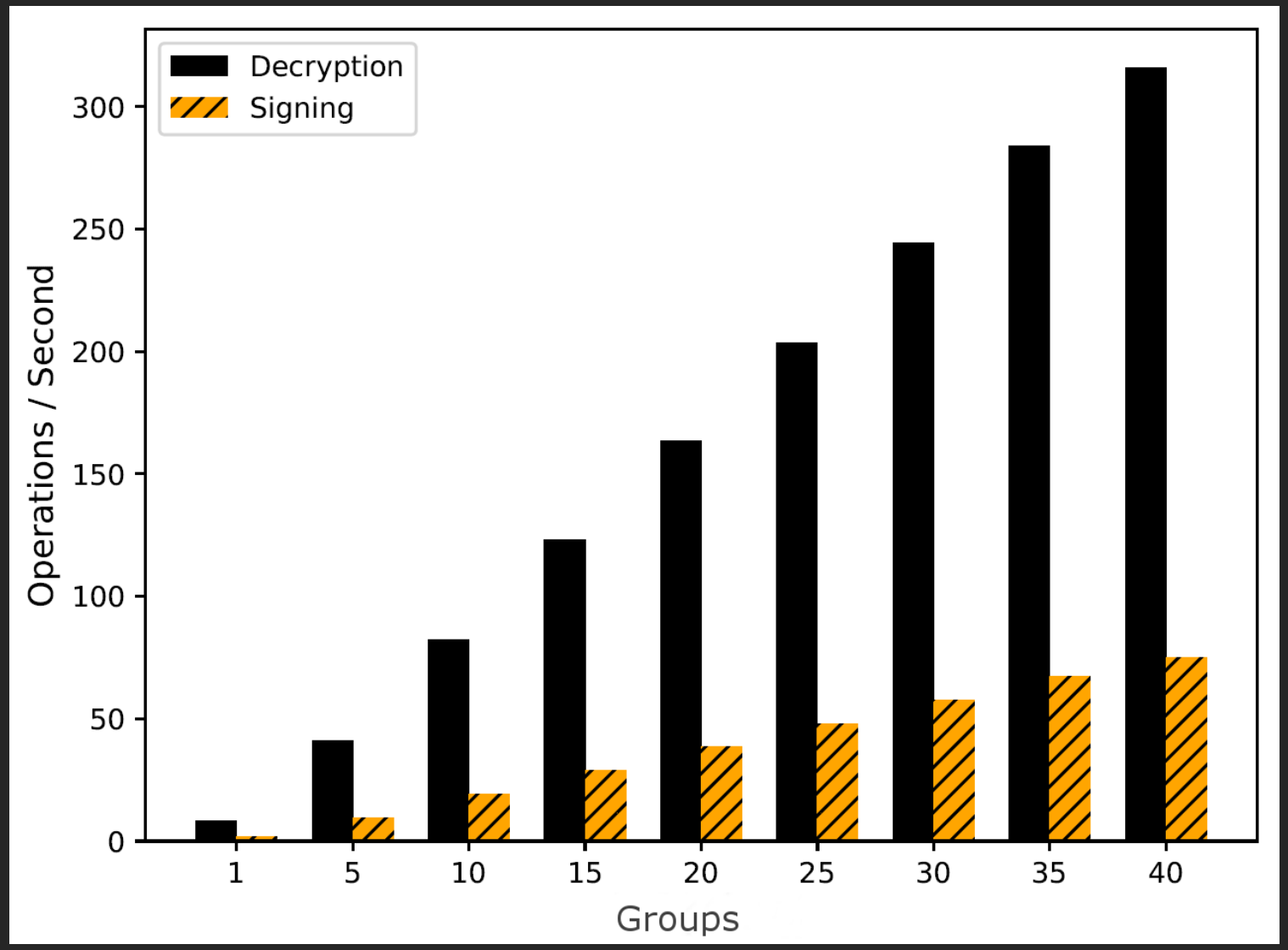
Tolerance vs Runtime



Does it scale?

- A single quorum (3 smartcards) can’t serve thousands of requests
- We could add more smartcards…

Scalability



DIY

Poor man’s HSM (~60$)

1. Buy a USB hub
2. 3-4 card readers (or more)
3. Buy cards from various manufacturers
4. Download our MPC applet
5. Review the code – single point of failure
6. Install the applet into your cards
7. Enjoy your homemade HSM!

Key Takeaways

New cryptographic device architecture:

- Tolerates faulty & malicious h/w components
- Decent Performance; Scales nicely; just keep adding ICs
- Commercial off-the-shelf components
- Existing hardware-trojan detection techniques are very welcome!

A Touch of Evil:

Cryptographic Hardware from Untrusted Components

BackdoorTolerance.org

Vasilios Mavroudis
Doctoral Researcher, UCL
v.mavroudis@cs.ucl.ac.uk

Panoramix