Hardware wallets

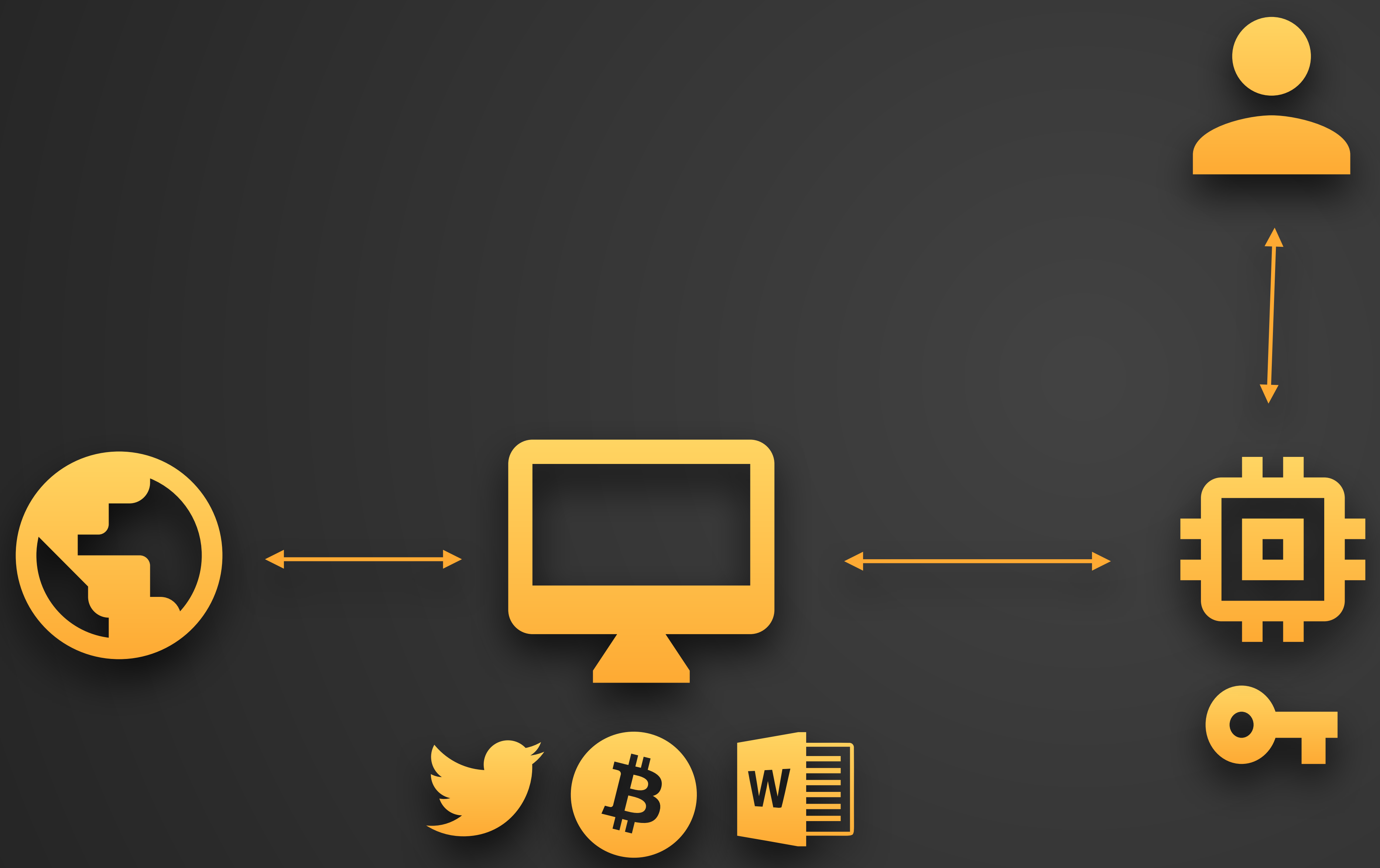
issues and best practices

crypto advance.

@StepanSnigirev

stepan@cryptoadvance.io

Overview



Secure the key

- Minimise the attack surface
- Avoid remote attacks
- Mitigate hardware attacks

Components

- Communication channel
- Microcontroller(s)
- Trusted peripherals

Overview

Protocol design

- PSBT
- Change detection
- Multisignature

Firmware security

- Secure boot
- Memory protection
- Supply chain

Secure key storage and generation

- Random numbers
- Hardware attacks
- PIN code and PIN counter
- Secure key storage

A few more hints

- Communication channels
- Hardware architecture
- Anti-tamper measures
- Platforms and languages

Protocol design

User needs to verify

- Total amount being spent and where it goes:
  - fee
  - destination address and amount
  - change address and amount (optional)

Hardware wallet knows

- BIP-32 root private key

Hardware wallet needs to know

- How to derive keys for signing
- What script is used in the prevouts
- What is the amount of prevouts (segwit)
- How to verify change address

Partially Signed Bitcoin Transactions

Global scope

- Unsigned transaction
- Cosigner's xpubs (new)

Inputs scope

- Full prevout (segwit) - amount & scriptPubKey
- Full previous transaction (legacy)
- Redeem script
- Witness script
- Derivation path (i.e. m/49h/0h/0h/0/32)

Outputs scope (if known)

- Redeem script
- Witness script
- Derivation path (i.e. m/49h/0h/0h/1/24)

Change detection

Input

3GGtfJQYAjxz4Wf29mStxrgL9HRgnjUS5s
P2SH_WPKH(m/49h/0h/0h/0/32)

Outputs: change or not?

3LeRQoJs8s8S3VQi8wUPBXEn2sSKLfCFti
P2SH_WPKH(m/49h/0h/0h/1/27)

bc1q7v4cs8dtxge2qvn6fz36th0vqhpgwhz3x2e86d
P2WPKH(m/49h/0h/0h/1/27)

36QzmK1agc7pRdb2ctSz1kvpydpwkQXVJj
(!) P2SH_WPKH(m/49h/0h/0h/1/99243234)

3DNUhBGTCgU85hzkj5coZSk8yErcsETQja
(!) P2SH_WPKH(m/49h/0h/1h/1/27)

Outputs scope (if known)

- Redeem script
- Witness script
- Derivation path (i.e. m/49h/0h/0h/1/24)

Multisig

Input

3GGtfJQYAjxz4Wf29mStxrgL9HRgnjUS5s

2 of 3 multisig:
- m/48h/0h/0h/1h/0/32
- <cosigner1>/48h/0h/0h/1h/0/32
- <cosigner2>/48h/0h/0h/1h/0/32

Outputs: change or not?

3LmuWLqGdJaZFpuVDZucrTiUqASgQTwZQM

2 of 3 multisig:
- m/48h/0h/0h/1h/1/27
- <cosigner1>/48h/0h/0h/1h/1/27
- <cosigner2>/48h/0h/0h/1h/1/27

Cosigners

- <cosigner1>/48h/0h/0h/1h : xpub1
- <cosigner2>/48h/0h/0h/1h : xpub2

Global scope

- Unsigned transaction
- Cosigner’s xpubs (new)

Outputs scope (if known)

- Redeem script
- Witness script
- Derivation path (i.e. m/49h/0h/0h/1/24)

Secure key storage and generation

Mnemonic phrase

Recovery phrase, BIP-39

Random sources:

- TRNG
- User input
- Two oscillators
- Antenna noise

Take a random number: →

f34b3e256b8b8bb9cf2f3e73e423521a

Convert to binary, add checksum:

11110011010 01011001111 10001001010 11010111000 10111000101 11011100111
00111100101 11100111110 01110011111 00100001000 11010100100 00110101100

Split, convert to words:

viable fly matter strike reward table device treat initial canal stand culture

* Now we also have Shamir Secret Sharing

Mnemonic phrase

Password and master key derivation

Take the mnemonic:

viable fly matter strike reward table device treat initial canal stand culture

Hash it 2048 times with the passphrase:

PBKDF2(password = mnemonic, salt = “mnemonic”+password, 2048).read(64)

Use the result as master private key:

chain code: 93fb9d28d8f8e60f0298f638b1c7340bb014f708daca29d47535dc0339b1ebd1
private key: ab819774d0cf931676302cc3b79d5e01127e91472543be4e84ebc5f7ff5676e4

Mnemonic phrase

Problems and discussions

Depends on the dictionary:

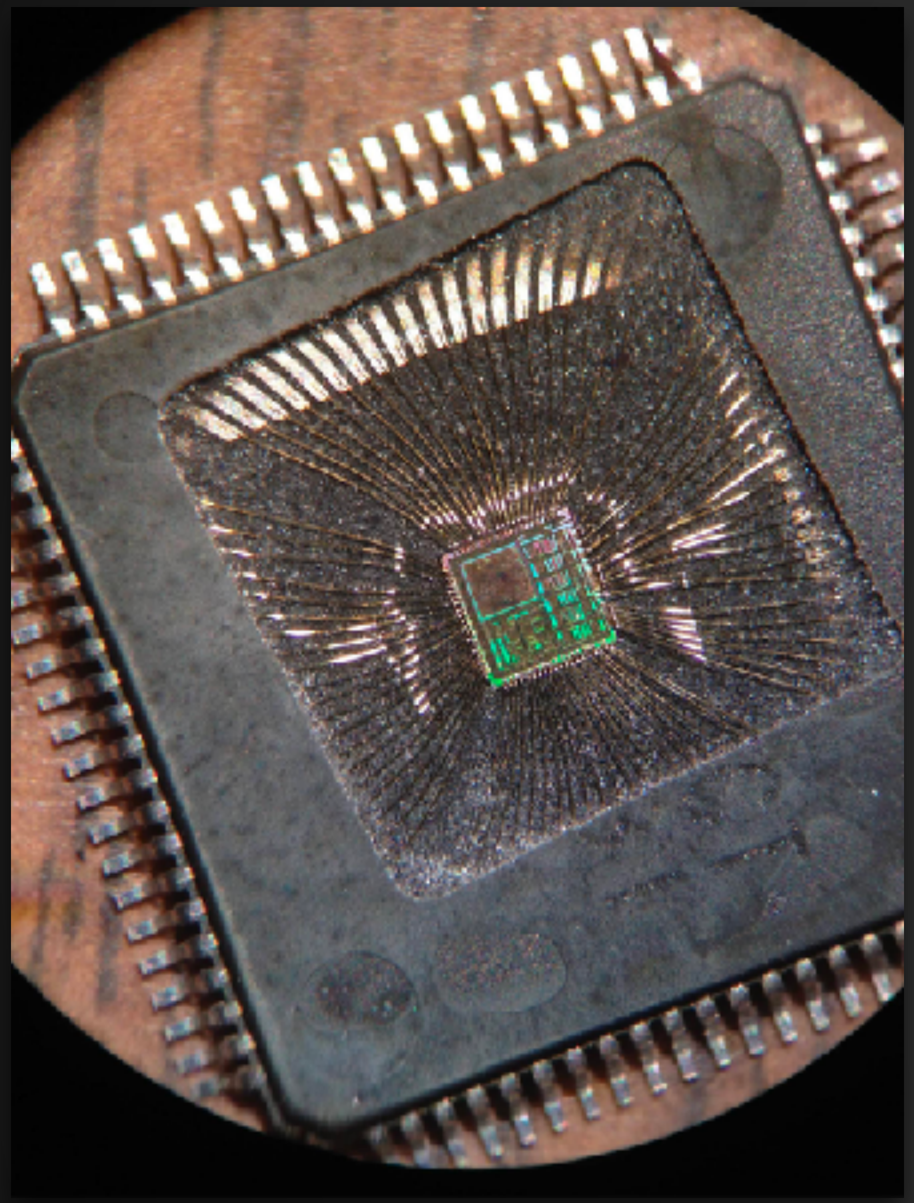
- Limited set of languages
- Only English is widely supported

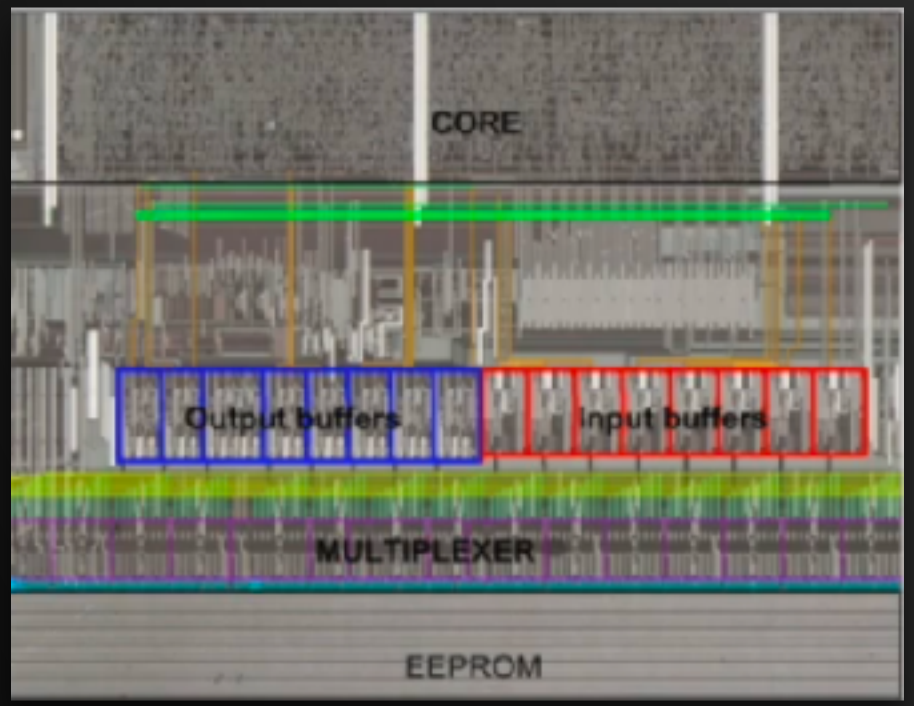
Hash-based checksum:

- Impossible to generate by a human
- Checksum is based on entropy

Types of hardware attacks

Even if your software is perfect, hardware is not



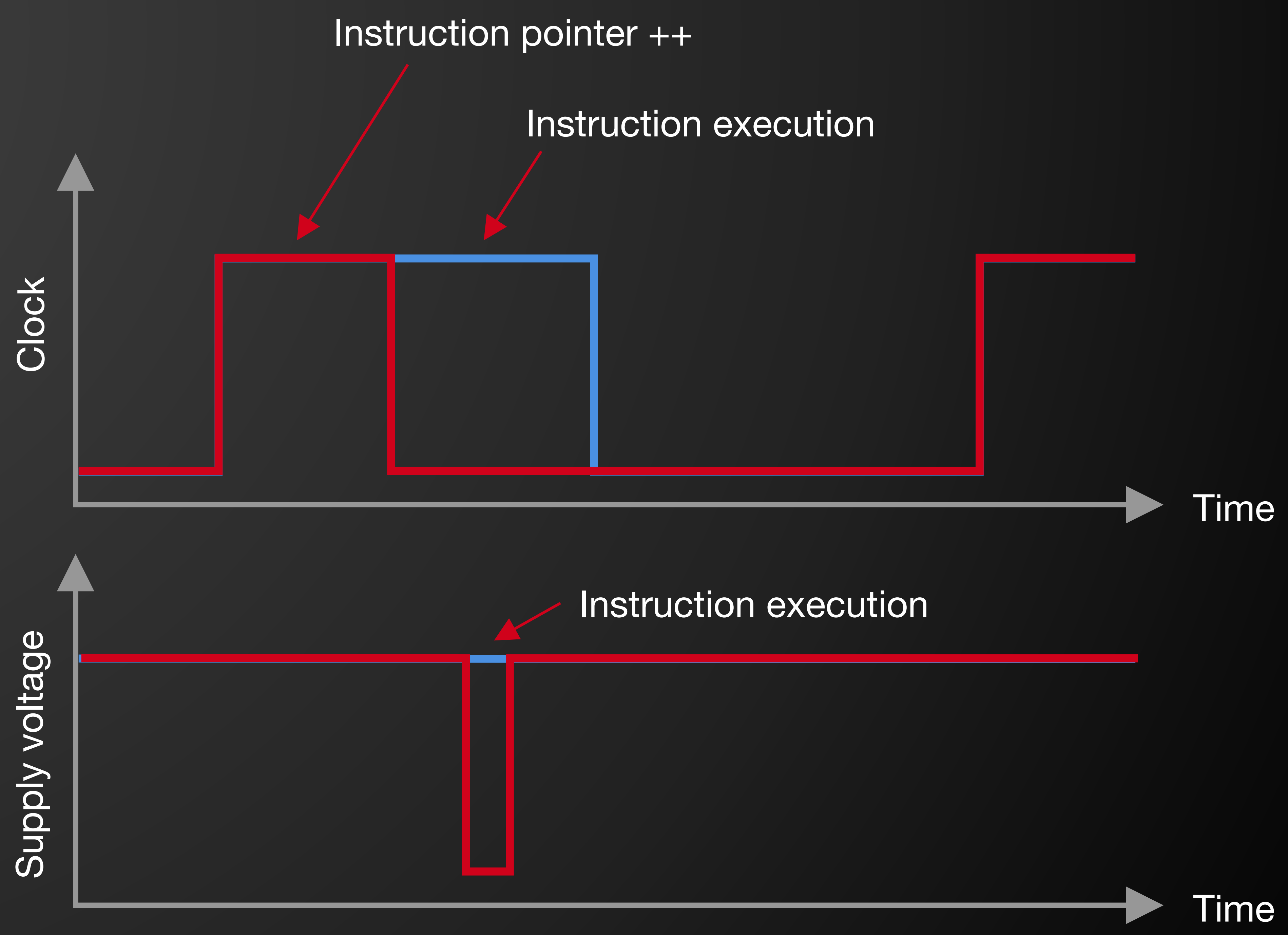


Side channels

- Timing attack
- Differential power analysis
- Data remanence attack
- Electromagnetic radiation
- much more…

Fault injection

- Clock glitching
- Voltage glitching
- Shooting with a laser
- much more…



PIN code

Best practices

- Do not store “correct PIN” - use cryptographic functions (i.e. HMAC)
  Side channel attacks (i.e. Trezor, March 2019)
- Increase the PIN counter before checking the PIN
  Simple conditional reset
- Use checksum for the PIN counter (i.e. use 01 for 0 and 10 for 1)
  Fault injection (i.e. laser beam on the memory region)
- Do not load private key to memory until PIN check is passed
  Data remanence attack (i.e. Trezor, August 2017)
- Encrypt the private key with a secret that includes PIN code
  Trezor Storage is a good library
- Think of device verification method
  Evil Maid attack, ColdCard’s double PIN technique is interesting

Key storage

Best practices

- Put secrets in flash & memory before everything else
  Glitching (i.e. Trezor, March 2019)
- Protect secrets with not readable bytes on edges
  Glitching
- Authenticate all information with private key (i.e. cosigners)
  Especially if it is stored on external chip
- Disallow access to secret keys from unsecure functions
  Use Memory Management Unit / Memory Protection Unit / TrustZone
- Physical isolation is the best
  Just add another microcontroller to do all insecure stuff
- Use a secure element if you can
  They are designed to protect secrets from hardware attacks

Firmware security

Best practices

- Use a secure bootloader that verifies the firmware
  Don't forget about key rotation & invalidation mechanism
- Use a unique secret & public key per device
  Helps against supply chain attack
- Don't allow firmware downgrades
- Disable debug interface, LOL

A few more hints

Best practices

- Unidirectional communication channel is the best
  SD card, QR codes, Audio modem with a switch
- Consider using PUFs and anti-tamper measures
  RAM PUF, anti-tamper switches, active meshes, tamper-evident resin
- Wisely choose a programming language
  Pick Python or Embedded Rust over C / C++
- Faster is sometimes better
  Fast MCUs, threading and small feature size makes attacker’s life harder
- Use standards
  Support multisig and BIPs

Questions?

I have a few toys with me ;)

crypto
advance.

@StepanSnigirev
stepan@cryptoadvance.io