Blockchains and Audit

CS 181S

November 28, 2018

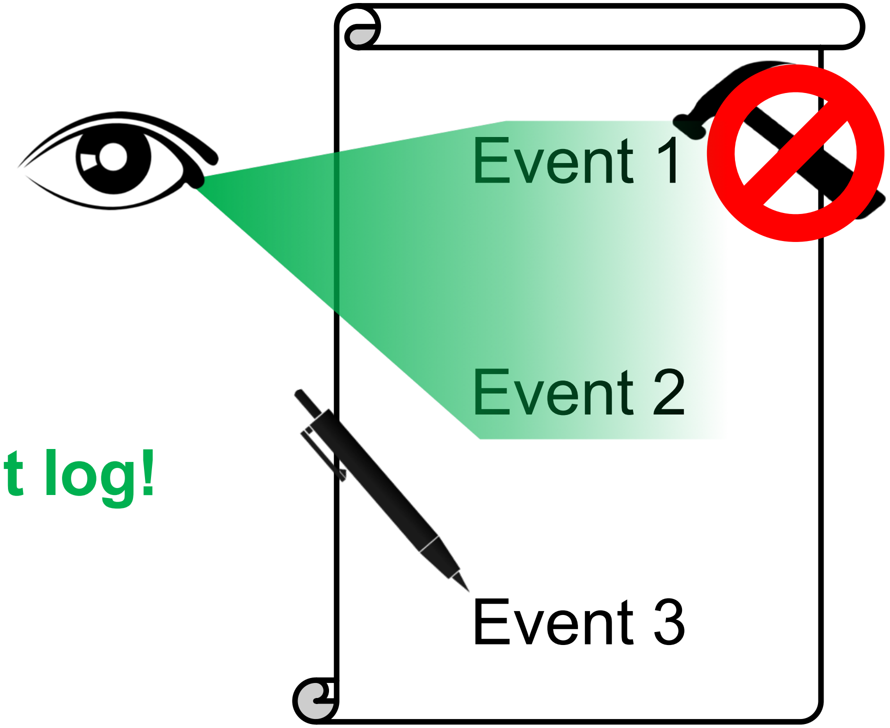
Blockchain: A public tamper-proof log

Publicly visible

Publicly writable

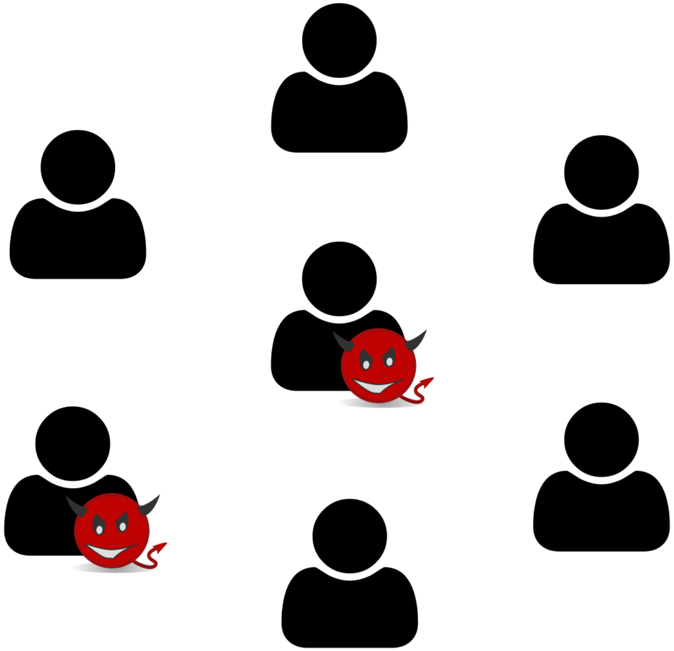
Unmodifiable

Useful for an audit log!



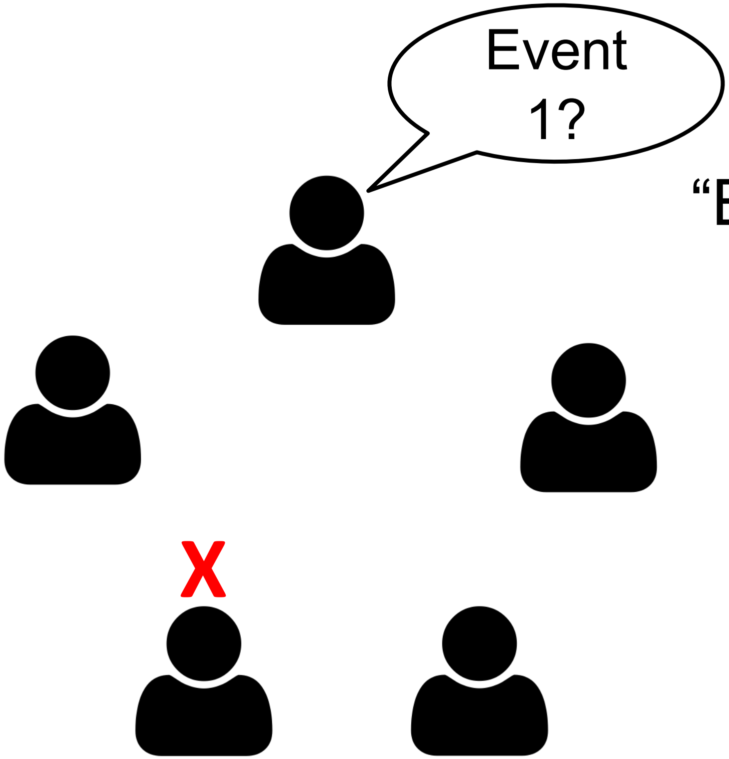
Preventing Tampering

vs



Traditional Consensus

Members Vote



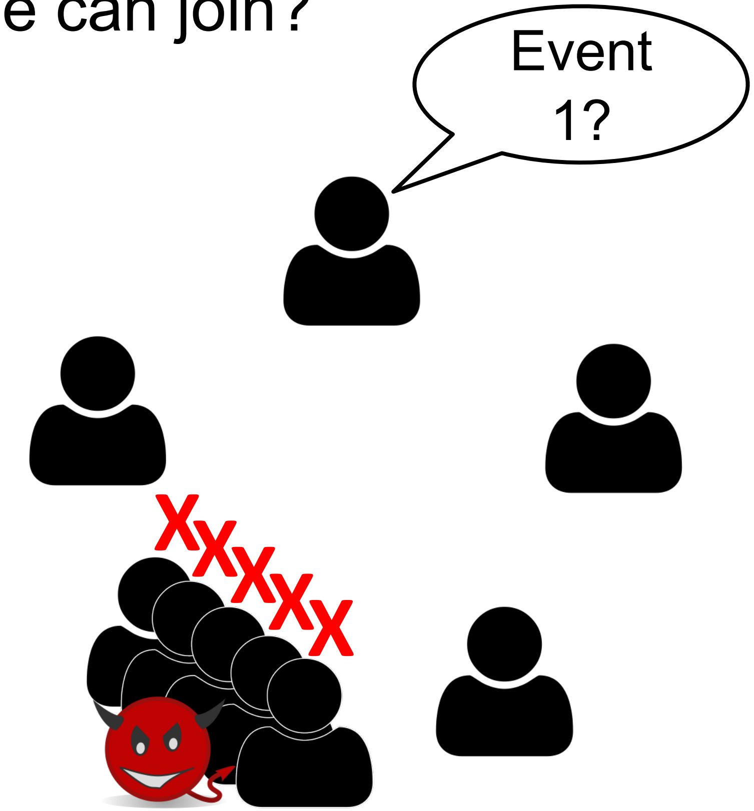
"Byzantine fault-tolerant (BFT) consensus"

Tolerates < 1/3 faulty

Must know who everyone is!

Sybil Attacks

What if anyone can join?



Defending against Sybil

Need a **scarce resource**

- BFT consensus uses identity – you only get one
- What else can we use?
  - Money (Proof of Stake)
  - Computational power (Proof of Work)

COMPUTATION AS A SCARCE RESOURCE: PROOF OF WORK

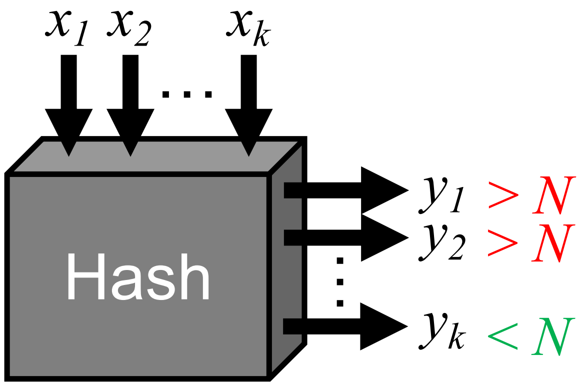
Proof of Work: The basics

Find x such that Hash(x) $< N$

- This could take a while…

What about replays?

- Add a nonce r
- Look for Hash($r \| x$) $< N$



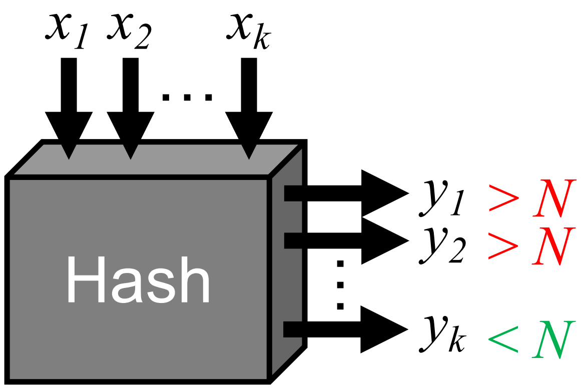
Proof of Work: Building a log

Make the nonce useful

Use a message digest!

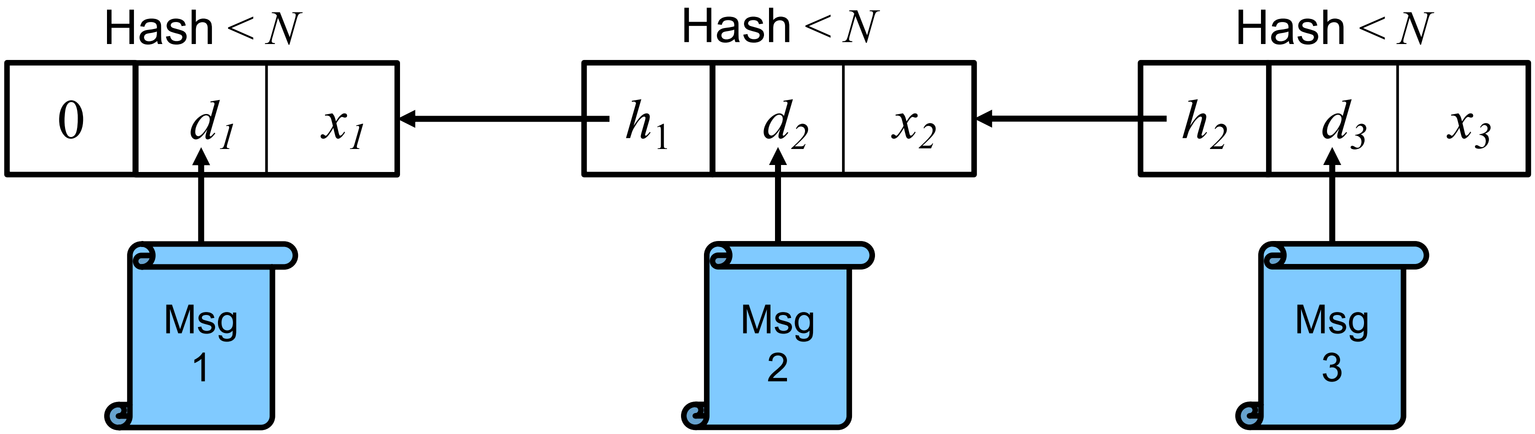
$d = \text{Digest}(m)$

Find x such that $\text{Hash}(d \| x) < N$



Proof of Work: Building a log

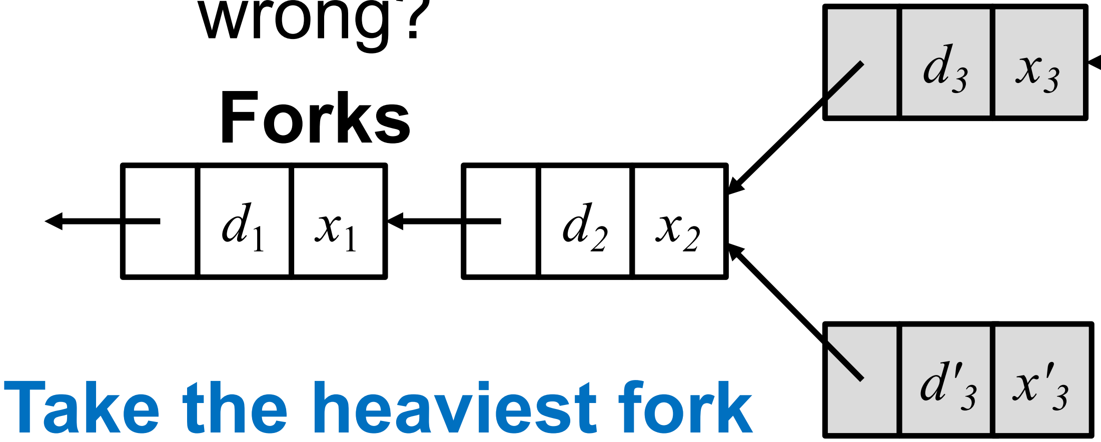
1. To add a message, generate a proof of work with that message
2. Connect each message to previous



Proof of Work: Coming to consensus

What can go wrong?

Forks



Take the heaviest fork (one with the most work)

Nakamoto Consensus

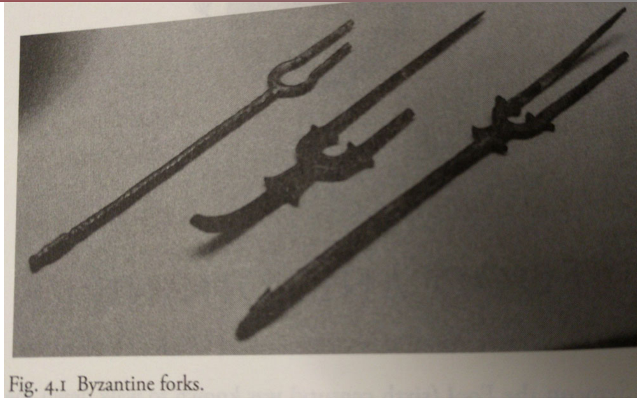


Fig. 4.1 Byzantine forks.

- If majority of computation is honest, honest parties will agree (eventually)
- Log is tamper-proof
  - It would require redoing all of the work to tamper

Blockchains for Audit

- **Individual accountability**
  - Everything is visible. Everyone is accountable.
- **Event reconstruction**
  - All of the events are there. Easy to reconstruct.
- **Real-time intelligence**
  - Miners can verify everything ~~as~~ it goes on the log.

before!

Not just a log!

Authoritative record

- Instead of logging events elsewhere, the blockchain can record the definition of events (e.g. transactions)
- Online validation can prevent illegal events from ever happening!

What restrictions make sense?

Transaction Processing System

- Each block has a limited number of transactions (1 MB)
- Transactions cannot create money
  - Except coinbase transaction to reward miner
- Coins can only be spent once (spending creates new unspent coins)
- To spend a coin conditions must be met (e.g., owner authorizes)

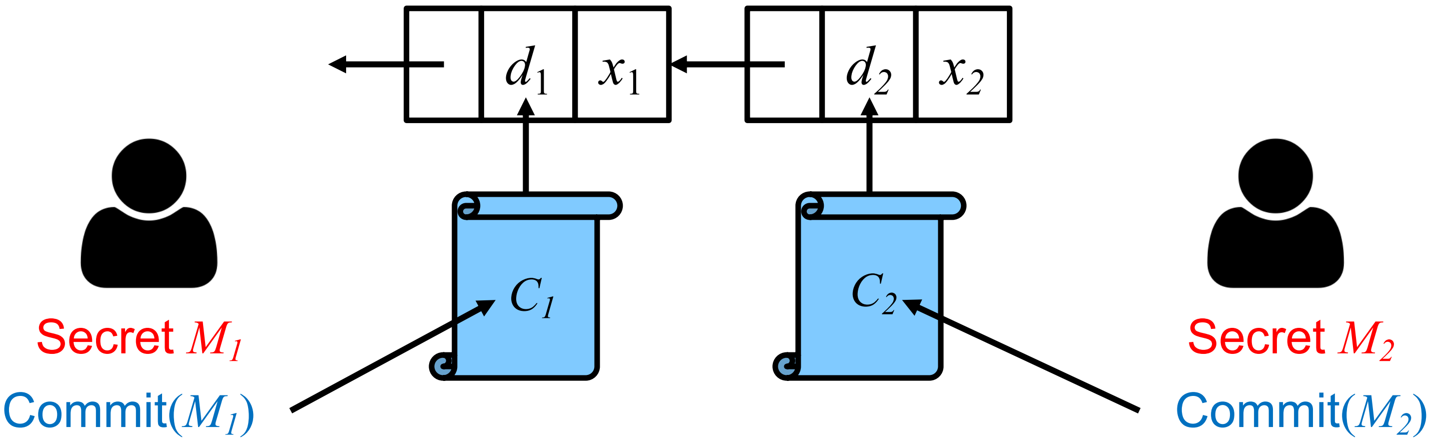
Bitcoin

BLOCKCHAINS AND CONFIDENTIALITY

What do we do with private data?

Cannot put it on the blockchain – everything is public

Only publish commitments



Commitment Schemes

- A commitment scheme Com is a two-phase, two-party protocol such that:
  - Secrecy: receiver does not learn anything about x from Com(x)
  - Binding: sender cannot produce alternative x' such that Com(x) = x'

- Example Protocol
  1. B->A: r
  2. A: choose random bit b. If b=0, Com(x) = Hash(x) else Com(x) = Hash(x) xor r
  3. A->B: Com(x)

     …

  1. A->B: x

What do we do with private data?

Cannot put it on the blockchain – everything is public

Only publish commitments

- Still tamper-proof
- No longer able to see actions
  - Cannot reconstruct events X
  - Cannot perform online validation X

Doing better with private data

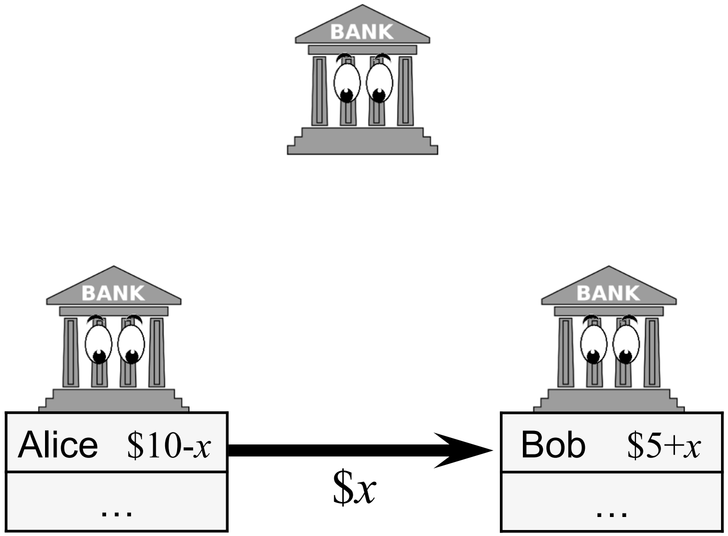
Verify data validity without leaking secrets

Ongoing research with two main tools

1. Heavy-duty cryptographic constructs
   - Complex zero-knowledge proofs
2. Trusted hardware
   - Places trust in hardware instead of crypto or a large group

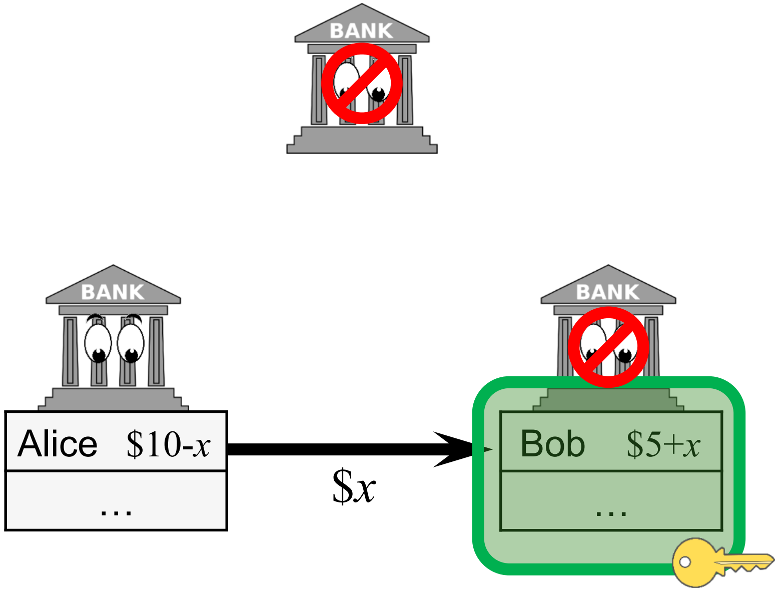
Cryptographic Example: Solidus

Bank-base confidential transactions



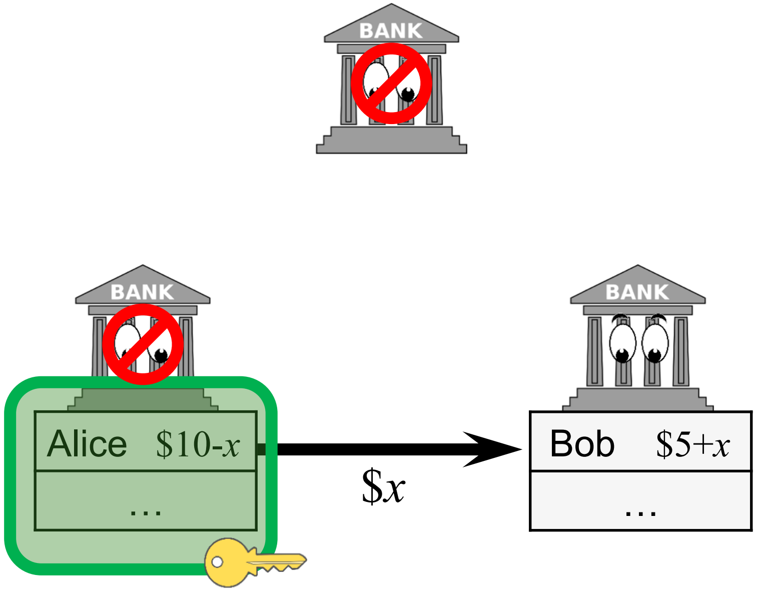
Cryptographic Example: Solidus

Sending bank can see sender and value



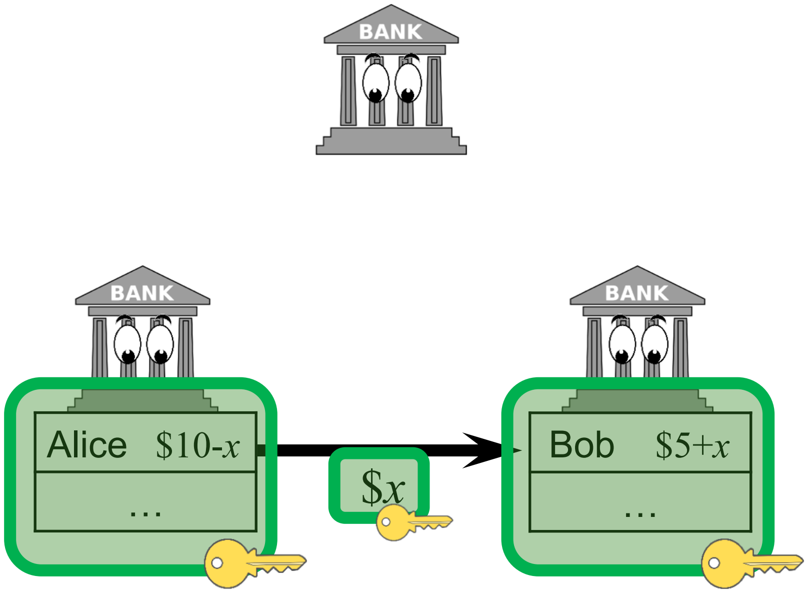
Cryptographic Example: Solidus

Receiving bank can see recipient and value



Cryptographic Example: Solidus

Everyone can see banks involved



Cryptographic Example: Solidus

Strong publicly verifiable integrity guarantees

- Sender authorized transaction
- Sender had money to send
- Transaction value was not negative
- Transaction was processed correctly

Can (provably) furnish transaction details to external auditor

Trusted Hardware

Special machine instructions

- Isolate process from the surrounding system
- Can remotely attest that they’re running specific code
  - Uses (literally) hard-wired keys in the CPU

Trustworthy code can operate on secret data and attest to correctness

Examples:

- Intel Software Guard eXtensions (SGX)
- ARM TrustZone