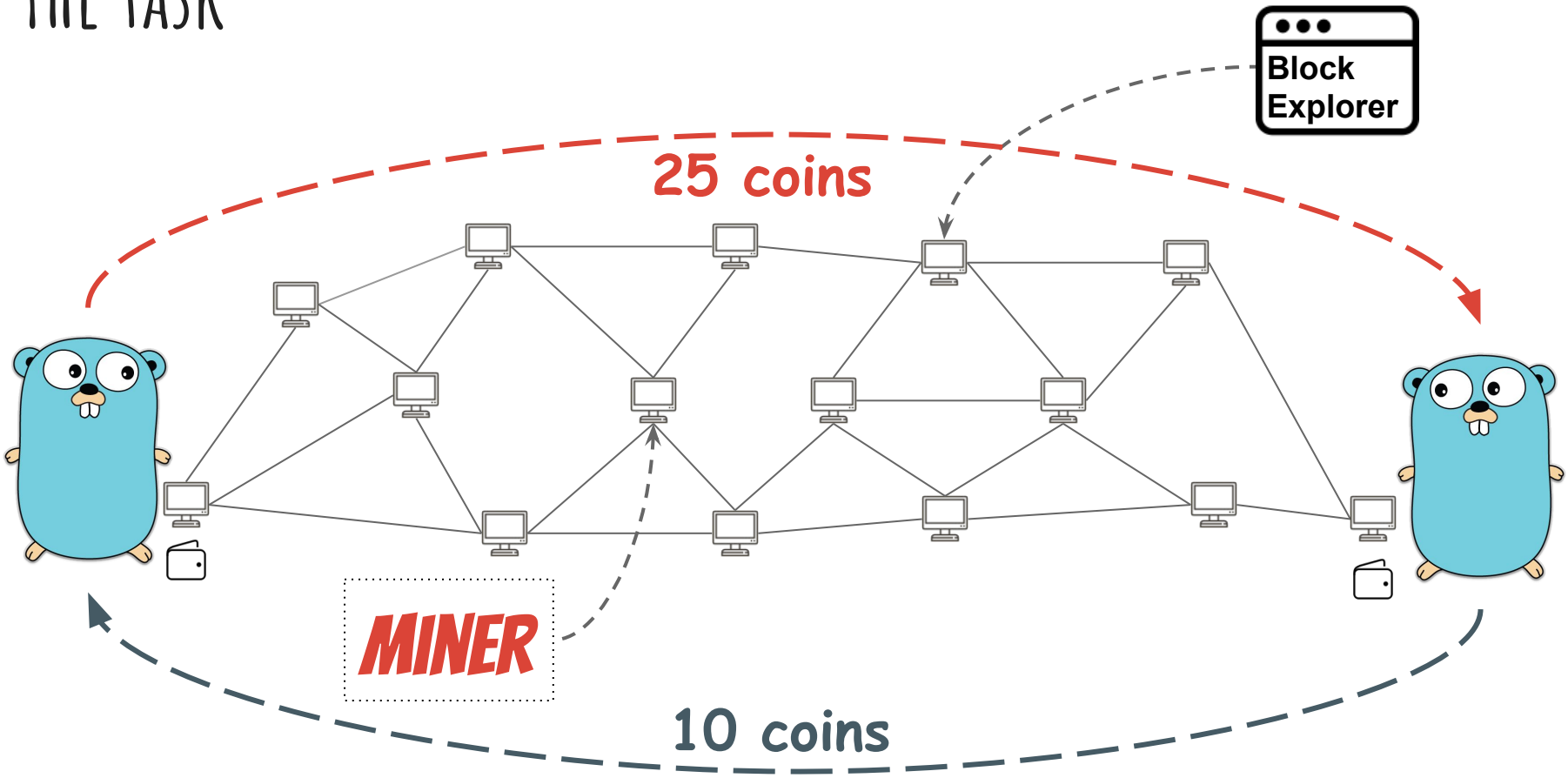
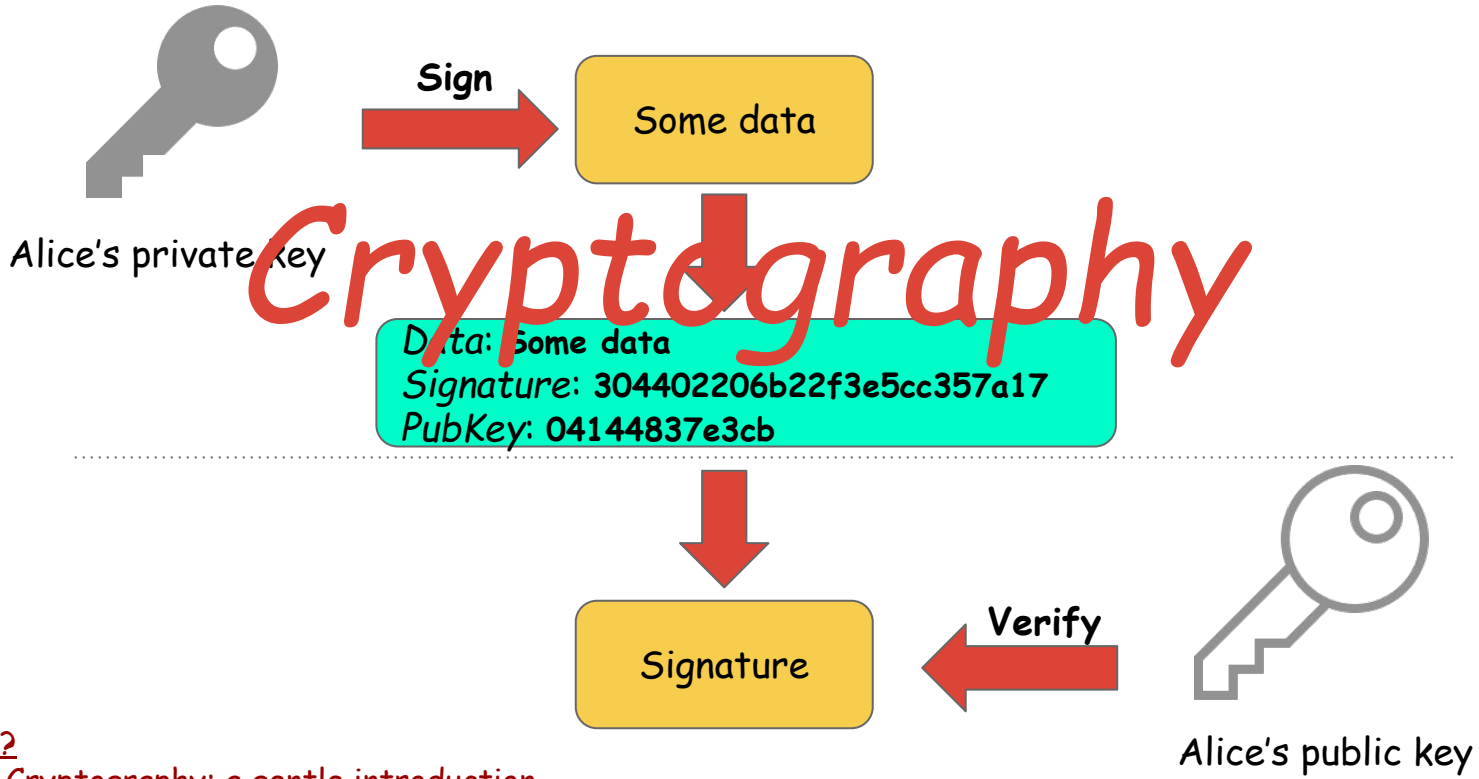
Cryptocurrency in Go

GopherCon Russia 2018
Afanasev Stanislav, JUNO
17.03.2017

The task



Cryptography



Why not RSA?
Elliptic Curve Cryptography: a gentle introduction

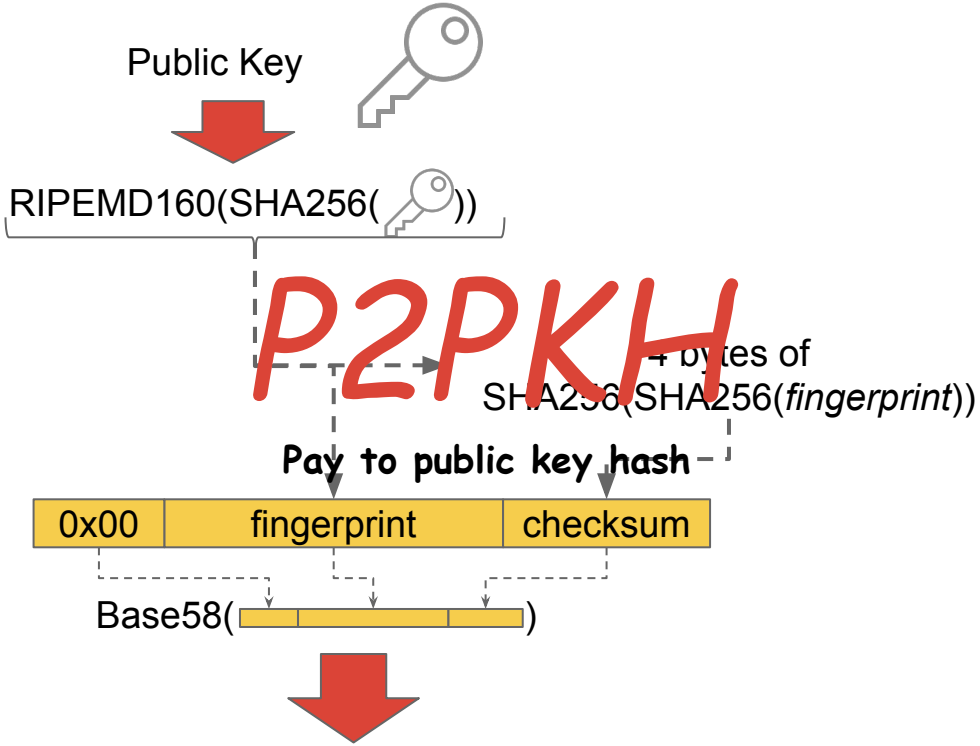
Cryptographic keys. Go

```
func main() {
	privKey, err := btcec.NewPrivateKey(btcec.S256())
	d := chainhash.DoubleHashB([]byte("some data"))

	signature, err := privKey.Sign(d)
	ok := signature.Verify(d, privKey.PubKey())
}
//Signature: e5cc357a17f18539e273235f8ad548ff5c44c9e8c3ae66dde732...
//Is sign valid: true
```



$ go get -u github.com/btcsuite/btcd/btcec

Addresses



Address
Base58Check encoding
Why does it use two hash functions (SHA-256 and RIPEMD-160)?

Addresses. Go

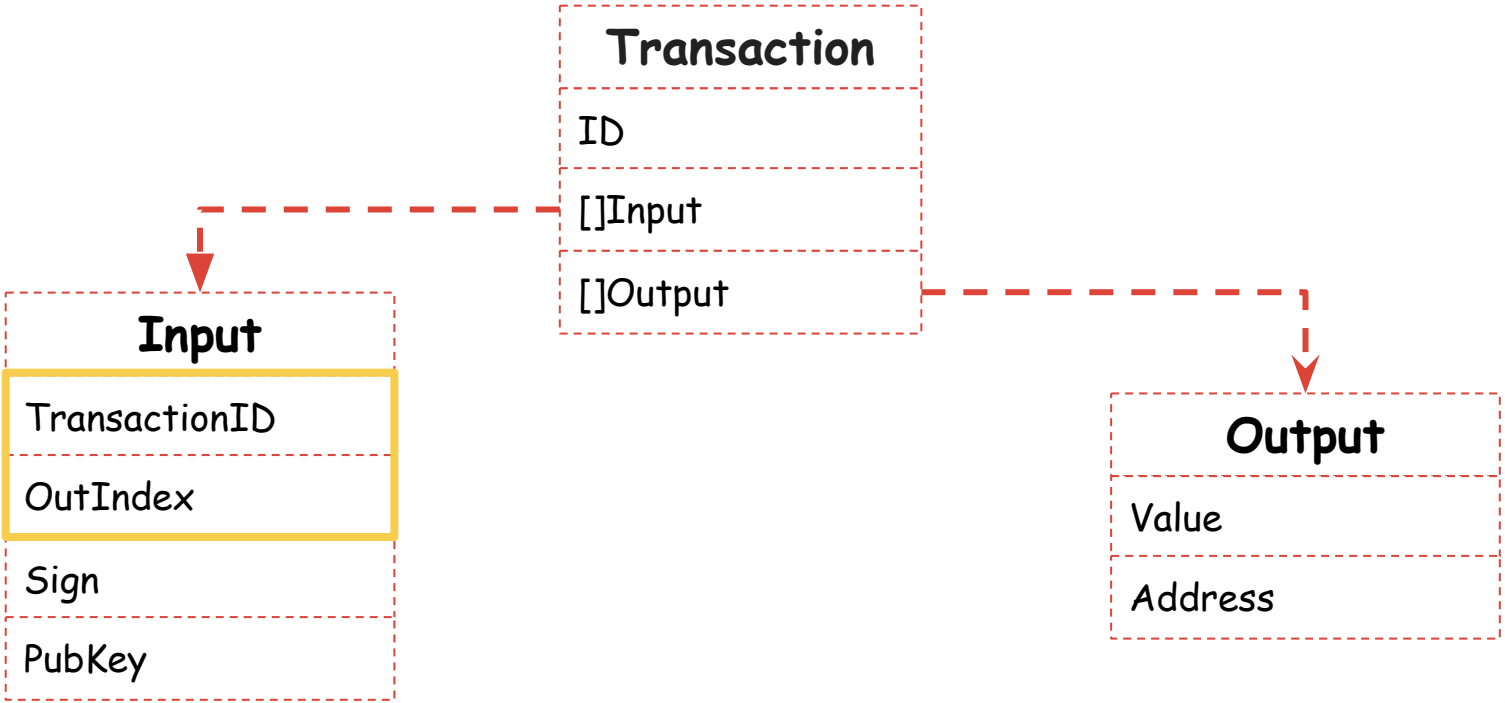
```
func main() {
        privKey, err := btcec.NewPrivateKey(btcec.S256())
        pubKey := privKey.PubKey().SerializeUncompressed()
        pubKeyHash := btcutil.Hash160(pubKey)
        addrPKH, err := btcutil.NewAddressPubKeyHash(pubKeyHash)

        addr := addrPKH.EncodeAddress()
}
//PubKey: 67bd04b262b7ef0fea7bd292cdc978344a578644251ed9...
//Address: 1Pwp7sqxYtrTkJExuZmHMfbh9ffbqzk6sH
```

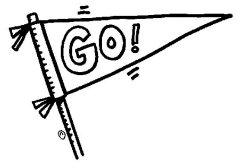




$ go get -u github.com/btcsuite/btcutil

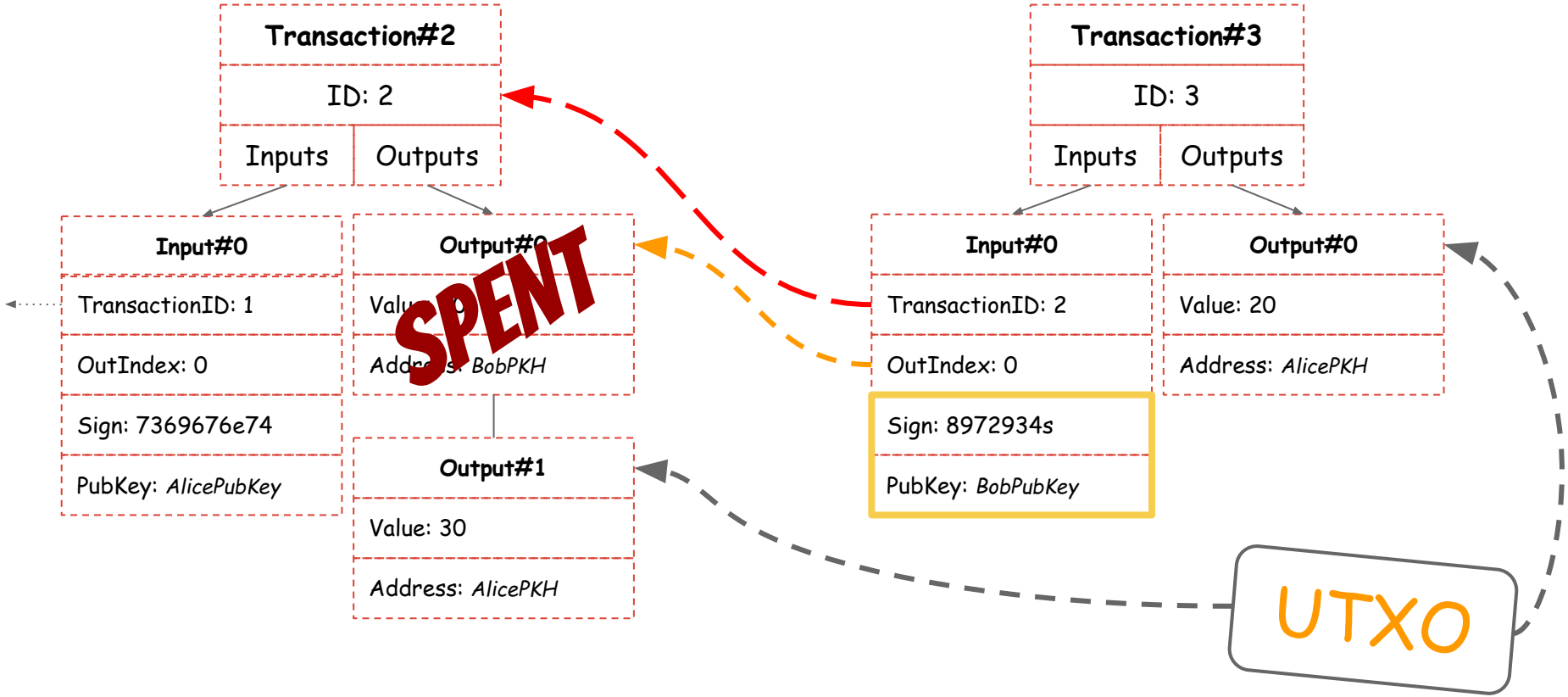
Transactions



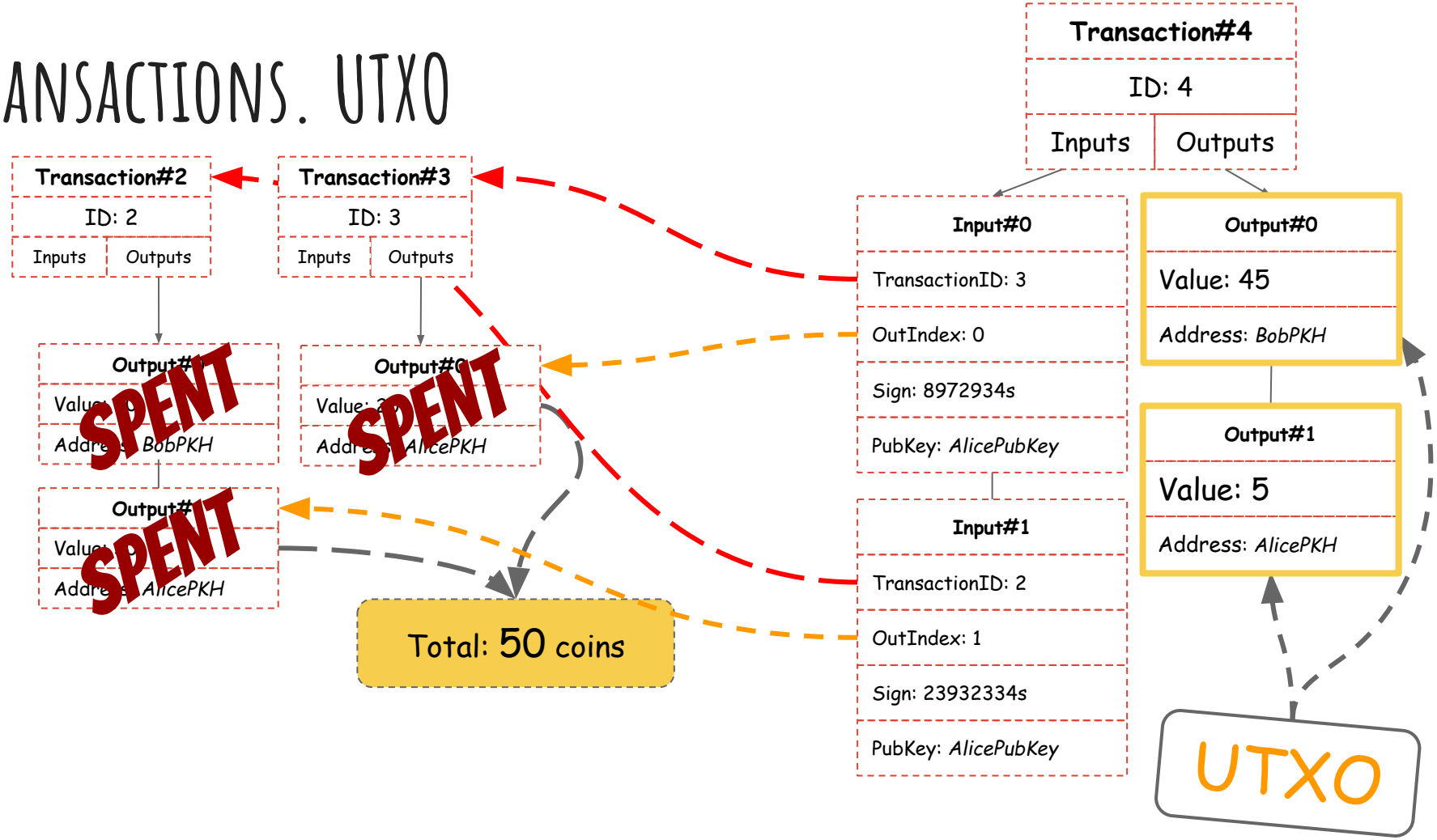


Script
Transaction

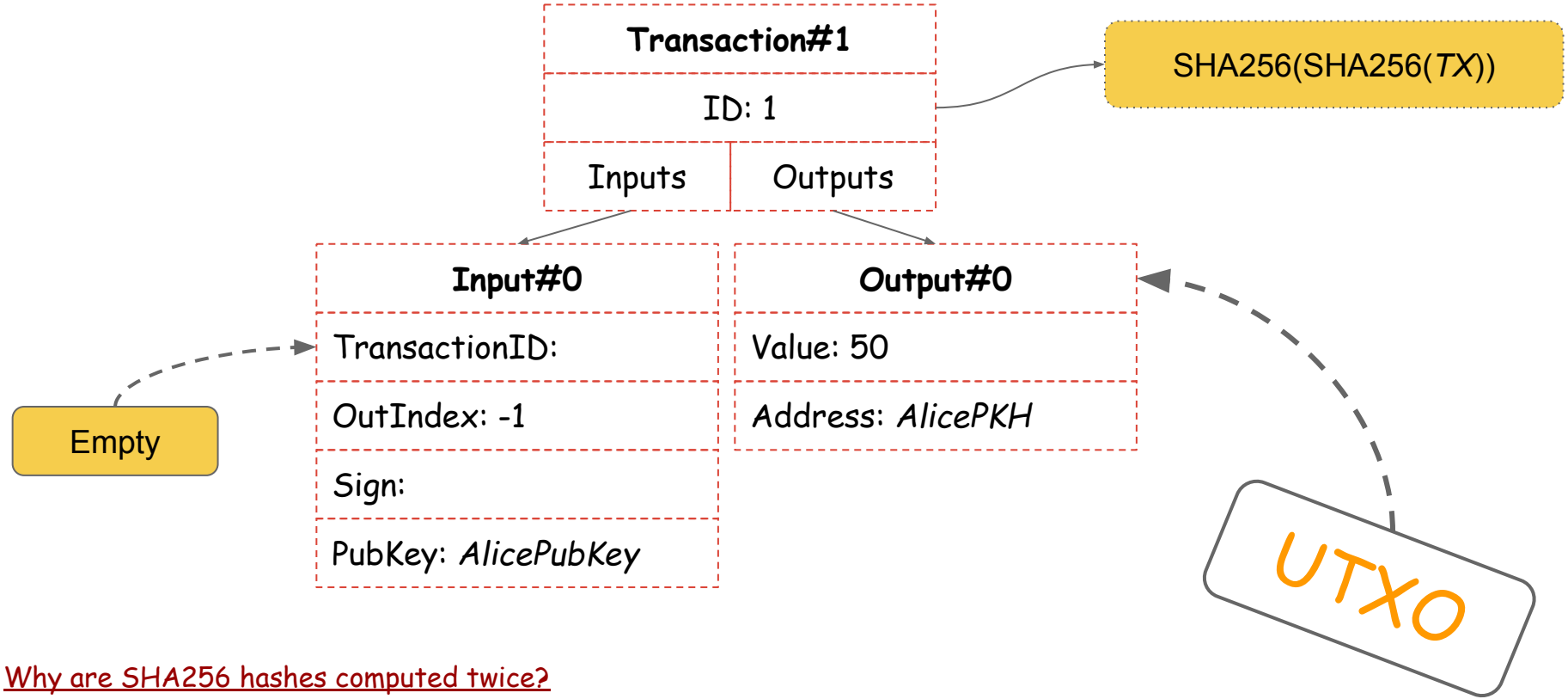
Transactions. UTXO



Transactions. UTXO



Transactions. Coinbase



Why are SHA256 hashes computed twice?

Transactions. Go

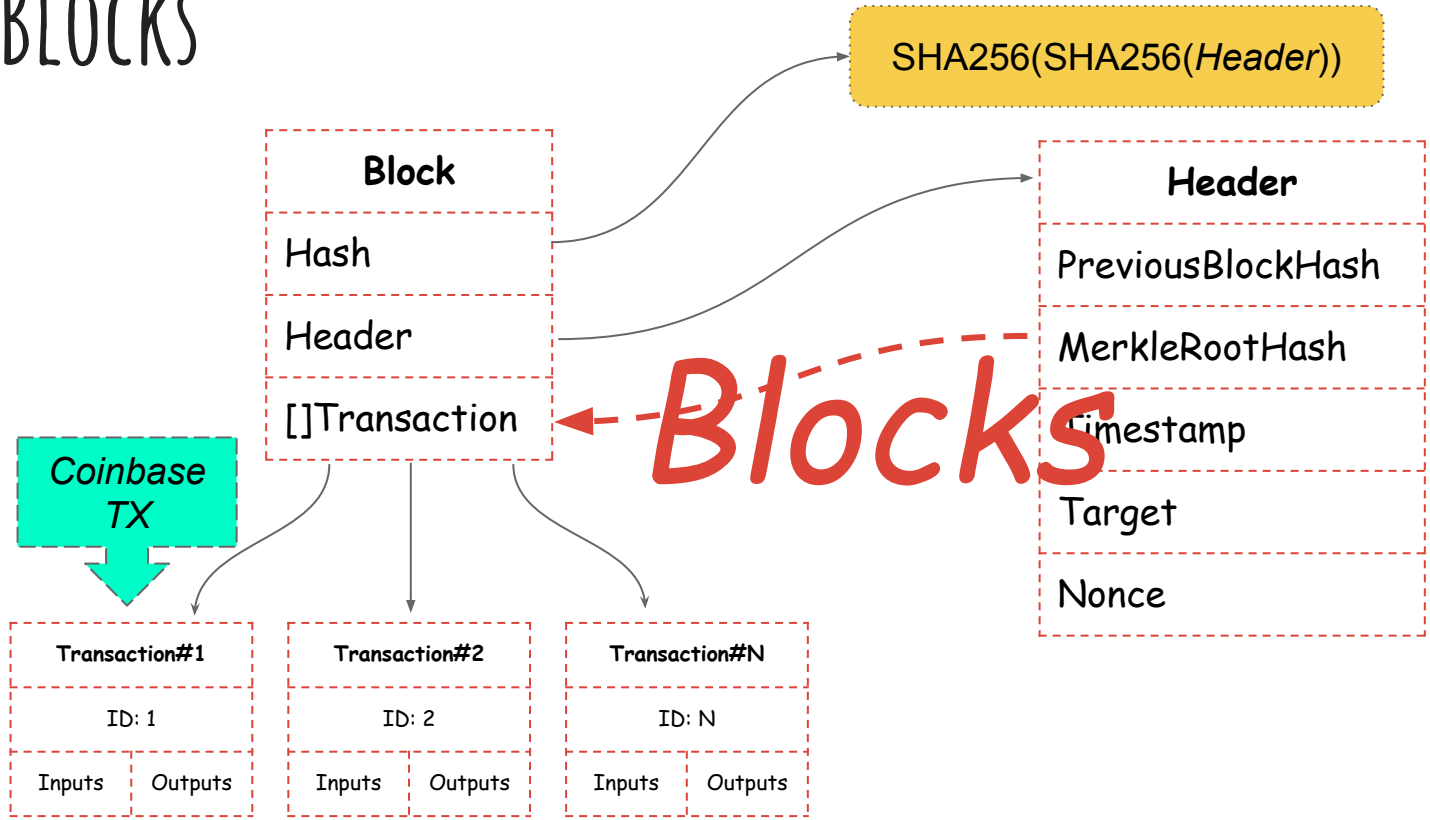
```
// Transaction represents a TX
type Transaction struct {
        ID      []byte
        Inputs  []Input
        Outputs []Output
}
```

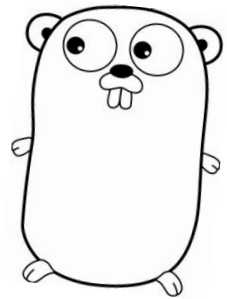
```
// Input represents an input
type Input struct {
        TransactionID []byte
        OutIndex      int64
        Sign          []byte
        PubKey        []byte
}
```

```
// Out represents an output
type Output struct {
        Value      int64
        Address    []byte
}
```

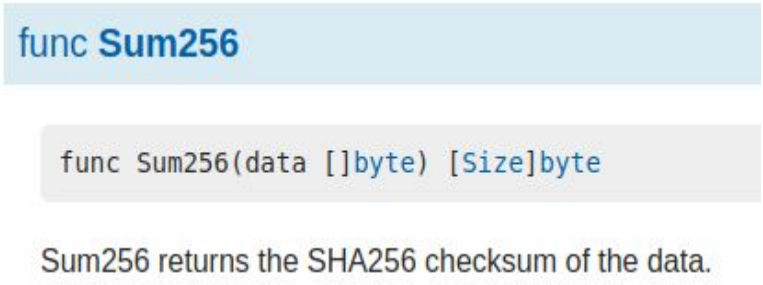
```
func New(us UTXOSet, from, to btcutil.AddressPubKeyHash, a amount.Amount) (Transaction, error) {}
func NewCoinBase(to btcutil.AddressPubKeyHash, reward amount.Amount) (Transaction, error) {}
func Sign(tx *Transaction, pk *btcec.PrivateKey) error {}
func Verify(tx *Transaction) error {}
```

Blocks


SHA256(SHA256(*Header*))
Block
Hash
Header
[]Transaction
Header
PreviousBlockHash
MerkleRootHash
Timestamp
Target
Nonce
Blocks
Coinbase TX
Transaction#1
ID: 1
Inputs
Outputs
Transaction#2
ID: 2
Inputs
Outputs
Transaction#N
ID: N
Inputs
Outputs



CRYPTO/SHA256



func **Sum256**

```
func Sum256(data []byte) [Size]byte
```

Sum256 returns the SHA256 checksum of the data.

```
func main() {
	blockHeader := []byte("blockHeader1")
	sha256.Sum256(sha256.Sum256(blockHeader)[:])
}
// invalid operation sha256.Sum256(blockHeader)[:] (slice of unaddressable value)
```

https://github.com/golang/go/wiki/Hashing

PKG/HASH

```
type Hash interface {
        // Write (via the embedded io.Writer interface) adds more data to the running hash.
        // It never returns an error.
        io.Writer

        // Sum appends the current hash to b and returns the resulting slice.
        // It does not change the underlying hash state.
        Sum(b []byte) []byte

        // Reset resets the Hash to its initial state.
        Reset()

        // Size returns the number of bytes Sum will return.
        Size() int

        // BlockSize returns the hash's underlying block size.
        // The Write method must be able to accept any amount
        // of data, but it may operate more efficiently if all writes
        // are a multiple of the block size.
        BlockSize() int
}
```

https://github.com/golang/go/wiki/Hashing

CRYPTO/SHA256

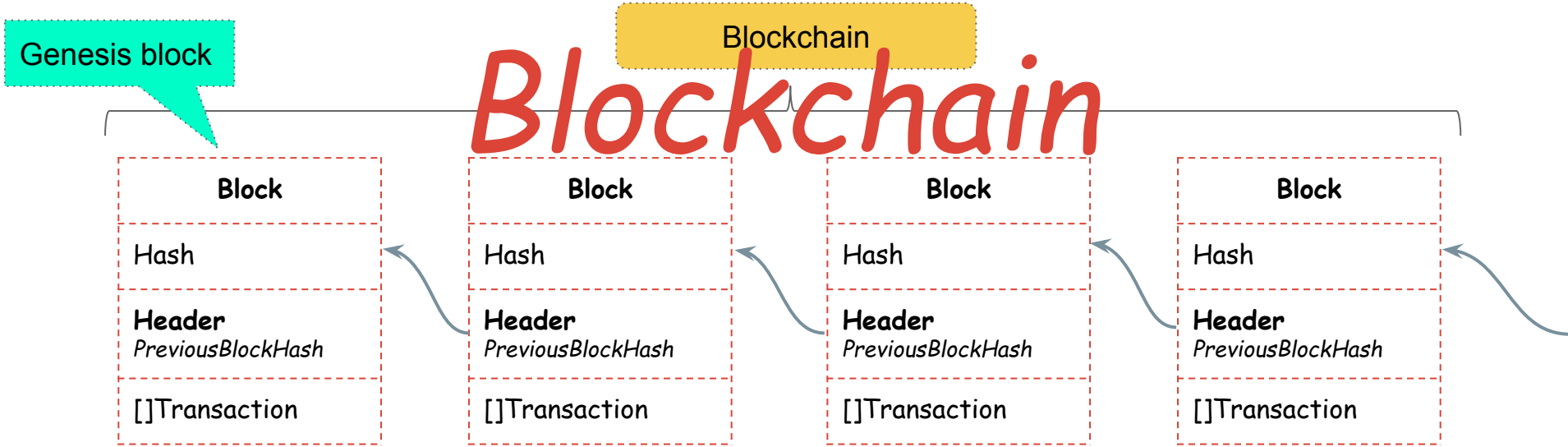
```go
func IneffectiveDoubleHash(b []byte) []byte {
	f := sha256.Sum256(b)
	s := sha256.Sum256(f[:])
	return s[:]
}
```

```go
func EffectiveDoubleHash(h hash.Hash, d, b []byte) []byte {
	h.Reset()
	h.Write(d)
	sum := h.Sum(b[:0])
	h.Reset()
	h.Write(sum)
	return h.Sum(b[:0])
}
```

```
//BenchmarkIneffectiveDoubleHash-4      1000000    1500 ns/op       32 B/op     1 allocs/op
//BenchmarkEffectiveDoubleHash-4        1000000    1427 ns/op        0 B/op     0 allocs/op
```

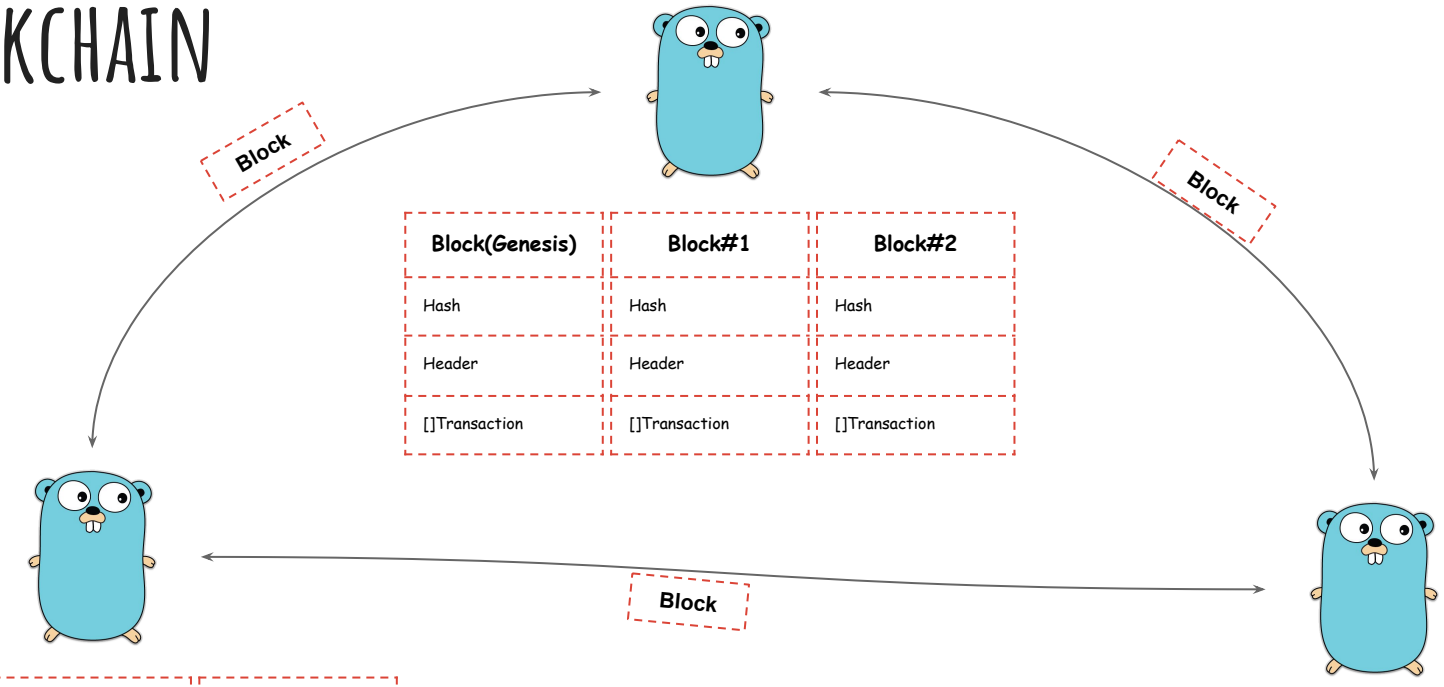
https://play.golang.org/p/vePTwCkQBFS
https://stackoverflow.com/a/45389087 - extra 40% of sha256 performance

Blockchain



Blockchain technology

Blockchain

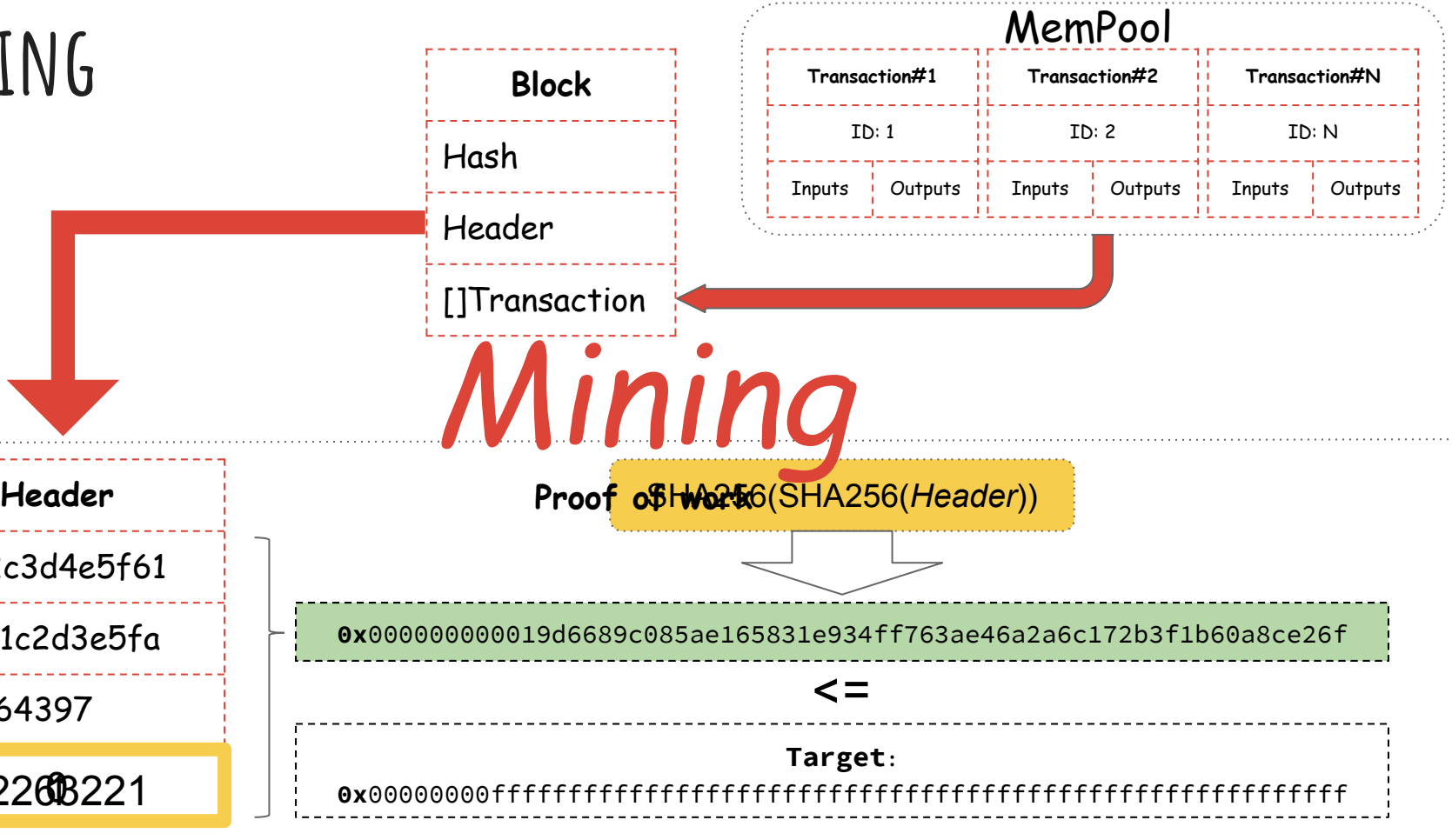


Block(Genesis)	Block#1	Block#2
Hash	Hash	Hash
Header	Header	Header
[]Transaction	[]Transaction	[]Transaction

Block(Genesis)	Block#1	Block#2
Hash	Hash	Hash
Header	Header	Header
[]Transaction	[]Transaction	[]Transaction

Consensus rules

Mining



PoW vs PoS

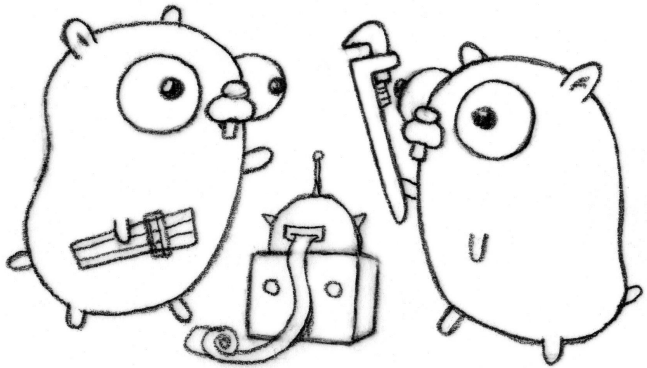
MATH/BIG

```
func main() {
	blockHash, _  := hex.DecodeString("0001AA")
	targetHash, _ := hex.DecodeString("0001FF")

	blockHashInt := big.NewInt(0).SetBytes(blockHash)
	targetHashInt := big.NewInt(0).SetBytes(targetHash)
}
// BlockHashInt: 426
// TargetHashInt: 511
// BlockHashInt <= TargetHashInt: true
```

https://golang.org/pkg/math/big/

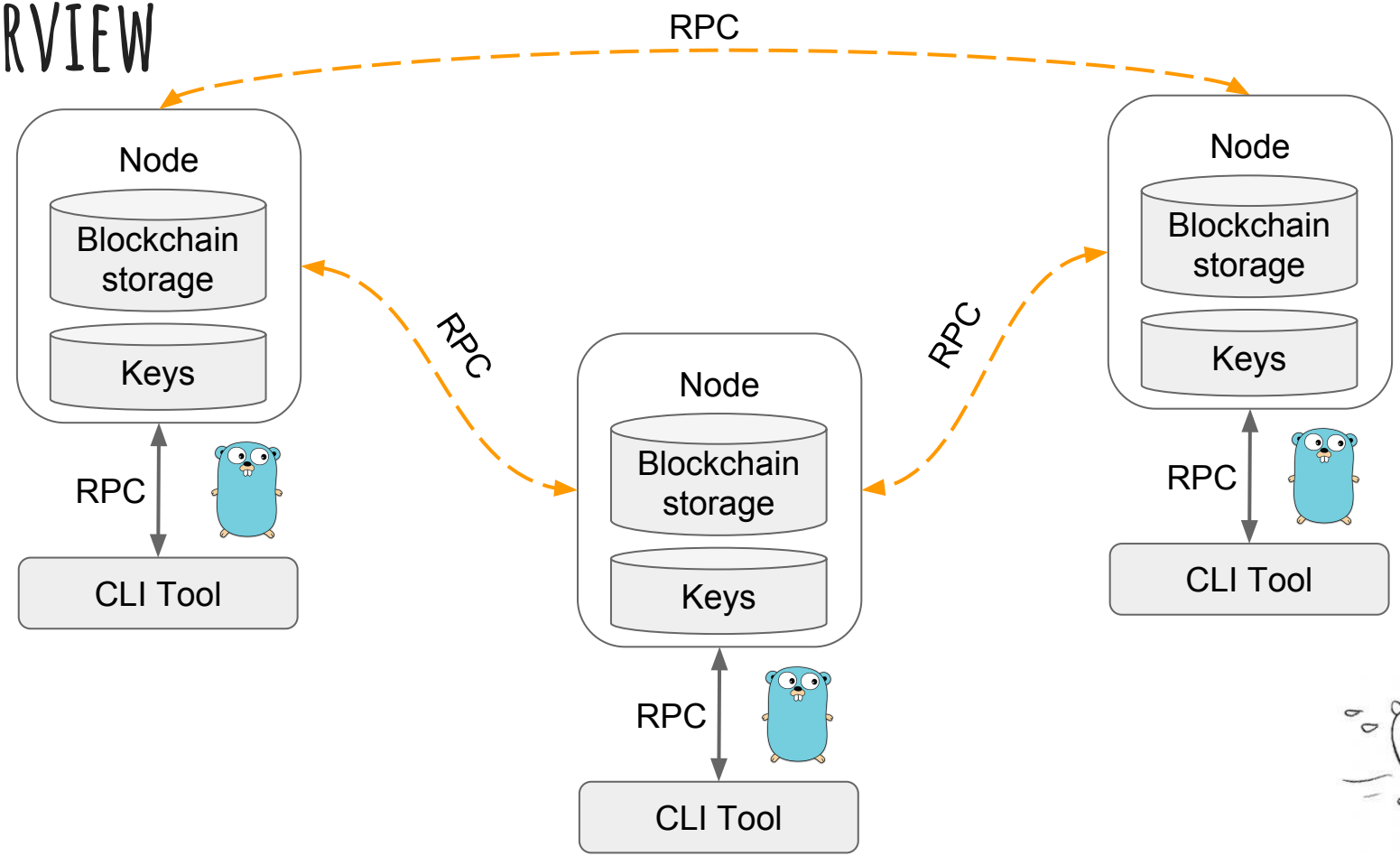
Block explorer. Go

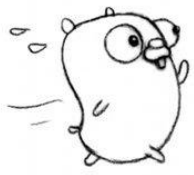
```
func main() {
    be := blockexplorer.New(storage, memPool)
    http.HandleFunc("/", be.ViewIndex)
    http.HandleFunc("/tx/", be.ViewTX)
    http.HandleFunc("/block/", be.ViewBlock)
    http.ListenAndServe(l, nil)
}
```



https://golang.org/pkg/net/http
https://golang.org/pkg/html/template/

Overview



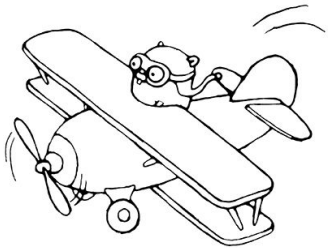


Network Protocol

The plan of demo

https://gcoin.website

Miner's node
0.0.0.0:30000



Alice's node
0.0.0.0:10000

Bob's node
0.0.0.0:20000

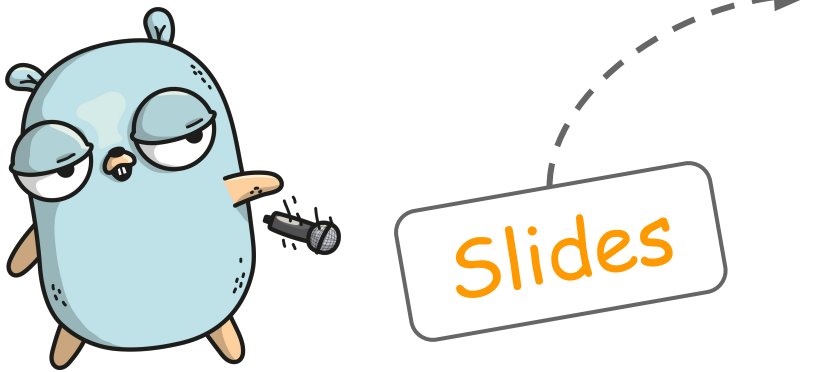
Transactions

DEMO

The end. Thank you!

@superstas88 - Twitter

https://github.com/superstas/gcoin



Framework/Projects

- https://github.com/tendermint/tendermint
- https://github.com/cosmos/cosmos-sdk
- https://github.com/hyperledger/fabric-sdk-go

- https://github.com/btcsuite/btcd
- https://github.com/ethereum/go-ethereum
- https://github.com/decred

Manuals/Documentation/Courses

- https://jeiwan.cc/posts/building-blockchain-in-go-part-1/
- https://chainhero.io/2017/07/tutorial-build-blockchain-app/
- https://goo.gl/7nRq9r

- https://en.bitcoin.it/wiki/Network
- https://en.bitcoin.it/wiki/Protocol_documentation
- https://github.com/bitcoinbook/bitcoinbook
- https://bitcoin.org/en/developer-guide#p2p-network

- https://www.coursera.org/learn/cryptocurrency