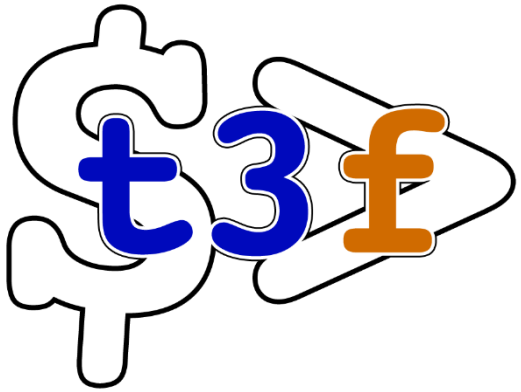
Leganés
6 Febrero 2014



Alberto Gómez Toribio (@gotoalberto)

Bitcoin protocol for developers

Leganés
6 Febrero 2014

Coinffeine

A P2P Bitcoin exchange

Coinffeine.com

Alberto Gómez @gotoalberto

"OpenData and Bitcoin Developer on fire.
Great coders are today's rock stars. That's it!"

Alvaro Polo @apolovald

"Software development enthusiast and aviation geek. Give me a higher-order function and I shall move the world."

Sebastián Ortega @_sortega

"Vocational coder since childhood, he will annoy you with concepts like "monad" and "actor model" but gets the work done."

Ximo Guanter

"Theoretical computer science believer by night, pragmatic ship-it developer by day. My brain runs on glucose, caffeine, general abstract nonsense and type theory."

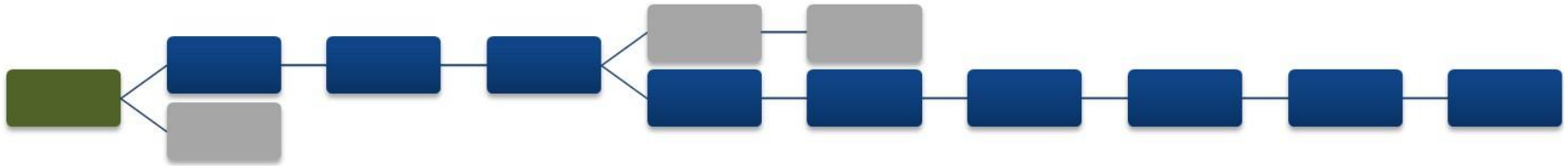
What is the Bitcoin Protocol?

- Network protocol
  - Blockchain minning
  - Fees and comitment rules
  - Sharing protocol: Gossip, Bittorrent, Gnutella...

- Transactions Mechanics
  - Validations of the money origin
  - Protocol to spend the money
  - Bitcoin Scripting

Network protocol

- Blockchain
  - The Blockchain is a distributed ledger book



  - Create a Block in chain require a proof of work

```
dc7047be… = SHA256(new_block_hash)
```

  - Forks are discarded, the complex chain wins.

Network protocol

- Other services: twister
  - The Blockchain is used to storage usernames.
  - Tweets are shared between users using a DHT protocol.
  - The older messages are discarded.
  - Incentive to generate blocks on Blockchain:
    Sponsored messages.

Network protocol

- Colored Coins: namecoin
  - The Blockchain is used to storage key-value strings, as internet web domains / IP.
  - You can register a domain spending Namecoins.
  - The fee by register a name is decreased in time.

Transactions Mechanic

- Transaction anatomy

```
{     txid      : 5c084b… ,
      locktime  : 0,
      vin       : [
                      {
                           txid           :  feff4b…
                           vout           :  0
                           scriptSig      {
                                             asm : d8f67a…
                                         …}
                      …}
                …]
      vout      : [
                       {
                           value          :  2.52
                           reqsigs        :  1
                           scriptPubKey   {
                                             asm : OP_DUP OP_HASH160 cb1f48…
                                                   OP_EQUALSVERIFY OP_CHECKSIG
                                         …}
                           addresses      :  [ 172D5w7C… ]
                       …}
                …]
 …}
```

Transactions Mechanic

- Transaction anatomy

```
{     txid      : 5c084b… ,
      locktime  : 0,
      vin       : [
                      {
                           txid           :  feff4b…
                           vout           :  0
                           scriptSig      :  {
                                               asm : d8f67a…
                                          …}
                      …}
                …]
      vout      : [
                       {
                           value          :  2.52
                           reqsigs        :  1
                           scriptPubKey   :  {
                                               asm : OP_DUP OP_HASH160 cb1f48…
                                                     OP_EQUALSVERIFY OP_CHECKSIG
                                          …}
                           addresses      :  [ 172D5w7C… ]
                       …}
                …]
 …}
```

Transaction Id, changes when a field is changed or tx is signed.

Transactions Mechanic

- Transaction anatomy

```
{     txid      : 5c084b… ,
      locktime  : 0,
      vin       : [
                      {
                           txid           :  feff4b…
                           vout           :  0
                           scriptSig      {
                                             asm : d8f67a…
                                         …}
                      …}
                …]
      vout      : [
                       {
                           value          :  2.52
                           reqsigs        :  1
                           scriptPubKey   {
                                             asm : OP_DUP OP_HASH160 cb1f48…
                                                   OP_EQUALSVERIFY OP_CHECKSIG
                                         …}
                           addresses      :  [ 172D5w7C… ]
                       …}
                …]
 …}
```

Locktime, Block or Time up to which this tx cannot be broadcasted.

Transactions Mechanic

- Transaction anatomy

```
{     txid      : 5c084b… ,
      locktime  : 0,
      vin       : [
                      {
                           txid           :  feff4b…
                           vout           :  0
                           scriptSig      :  {
                                             asm : d8f67a…
                                          …}
                      …}
                …]
      vout      : [
                       {
                           value          :  2.52
                           reqsigs        :  1
                           scriptPubKey   :  {
                                             asm : OP_DUP OP_HASH160 cb1f48…
                                                   OP_EQUALSVERIFY OP_CHECKSIG
                                          …}
                           addresses      :  [ 172D5w7C… ]
                       …}
                …]
 …}
```

TXid reference output for this input.

Transactions Mechanic

- Transaction anatomy

```
{     txid      : 5c084b… ,
      locktime  : 0,
      vin       : [
                      {
                           txid           :  feff4b…
                           vout           :  0
                           scriptSig      {
                                             asm : d8f67a…
                                         …}
                      …}
                …]
      vout      : [
                       {
                           value          :  2.52
                           reqsigs        :  1
                           scriptPubKey   {
                                             asm : OP_DUP OP_HASH160 cb1f48…
                                                   OP_EQUALSVERIFY OP_CHECKSIG
                                         …}
                           addresses      :  [ 172D5w7C… ]
                       …}
                …]
 …}
```

Output number reference in previous tx

Transactions Mechanic

- Transaction anatomy

```
{     txid      : 5c084b… ,
      locktime  : 0,
      vin       : [
                      {
                           txid          :  feff4b…
                           vout          :  0
                           scriptSig     {
                                              asm : d8f67a…
                                         …}
                      …}
                …]
      vout      : [
                       {
                           value         :  2.52
                           reqsigs       :  1
                           scriptPubKey  {
                                              asm : OP_DUP OP_HASH160 cb1f48…
                                                    OP_EQUALSVERIFY OP_CHECKSIG
                                         …}
                           addresses     :  [ 172D5w7C… ]
                       …}
                …]
 …}
```

ScriptSig which unlock the connected OutputScript

Transactions Mechanic

- Transaction anatomy

```
{     txid      : 5c084b… ,
      locktime  : 0,
      vin       : [
                      {
                           txid           :  feff4b…
                           vout           :  0
                           scriptSig      {
                                             asm : d8f67a…
                                         …}
                      …}
                …]
      vout      : [
                       {
                           value          :  2.52
                           reqsigs        :  1
                           scriptPubKey   {
                                             asm : OP_DUP OP_HASH160 cb1f48…
                                                   OP_EQUALSVERIFY OP_CHECKSIG
                                         …}
                           addresses      :  [ 172D5w7C… ]
                       …}
                …]
 …}
```

Value to transfer. The difference with the summatory of inputs values is the fee.

Transactions Mechanic

- Transaction anatomy

```
{     txid      : 5c084b… ,
      locktime  : 0,
      vin       : [
                      {
                           txid           :  feff4b…
                           vout           :  0
                           scriptSig      {
                                              asm : d8f67a…
                                         …}
                      …}
                …]
      vout      : [
                       {
                           value          :  2.52
                           reqsigs        :  1
                           scriptPubKey   {
                                              asm : OP_DUP OP_HASH160 cb1f48…
                                                    OP_EQUALSVERIFY OP_CHECKSIG
                                         …}
                           addresses      :  [ 172D5w7C… ]
                       …}
                …]
 …}
```

Number of signs required to broadcast this transaction. The signers are included in the "addresses" array.

Transactions Mechanic

- Transaction anatomy

```
{     txid      : 5c084b… ,
      locktime  : 0,
      vin       : [
                      {
                           txid           :  feff4b…
                           vout           :  0
                           scriptSig      {
                                             asm : d8f67a…
                                          …}
                      …}
                …]
      vout      : [
                       {                          Script. It defines how the money can be spended and by whom.
                           value          :  2.52
                           reqsigs        :  1
                           scriptPubKey   {
                                             asm : OP_DUP OP_HASH160 cb1f48…
                                                   OP_EQUALSVERIFY OP_CHECKSIG
                                          …}
                           addresses      :  [ 172D5w7C… ]
                       …}
                …]
 …}
```

Transactions Mechanic

- Transaction anatomy

```
{     txid      : 5c084b… ,
      locktime  : 0,
      vin       : [
                      {
                           txid           :  feff4b…
                           vout           :  0
                           scriptSig      {
                                             asm : d8f67a…
                                         …}
                      …}
                …]
      vout      : [
                       {
                           value          :  2.52
                           reqsigs        :  1
                           scriptPubKey   {
                                             asm : OP_DUP OP_HASH160 cb1f48…
                                                   OP_EQUALSVERIFY OP_CHECKSIG
                                         …}
                           addresses      :  [ 172D5w7C… ]
                       …}
                …]
 …}
```

Bitcoin Script

- It Allows to define how an output will be spended and by whom.
- Is a non Turing-Complete language which is evaluated as a Stack, from left to right.
- It just allows to write pure functions (without context)
- Is a non Turing-Complete language which is evaluated as a Stack.

How Bitcoin Script works?

(Public Key Hash)

```
OP_DUP OP_HASH160 cb1f48…  OP_EQUALVERIFY OP_CHECKSIG
```

How Bitcoin Script works?

(Signature) **(Public Key)**

`bbba3f5… 172D5w7C…`

(Public Key Hash)

`OP_DUP OP_HASH160 cb1f48… OP_EQUALVERIFY OP_CHECKSIG`

How Bitcoin Script works?

(Signature) (Public Key) (Public Key Hash)

```
bbba3f5… 172D5w7C…  OP_DUP OP_HASH160 cb1f48…  OP_EQUALVERIFY OP_CHECKSIG
```

Stack

How Bitcoin Script works?

(Signature) (Public Key) (Public Key Hash)

```
bbba3f5… 172D5w7C…  OP_DUP OP_HASH160 cb1f48…  OP_EQUALVERIFY OP_CHECKSIG
```

Stack

How Bitcoin Script works?

(Public Key) (Public Key Hash)

172D5w7C… OP_DUP OP_HASH160 cb1f48… OP_EQUALVERIFY OP_CHECKSIG

Stack

(Signature) bbba3f5…

How Bitcoin Script works?

(Public Key) (Public Key Hash)

172D5w7C… OP_DUP OP_HASH160 cb1f48… OP_EQUALVERIFY OP_CHECKSIG

Stack

(Signature) bbba3f5…

How Bitcoin Script works?

(Public Key Hash)

```
OP_DUP OP_HASH160 cb1f48…  OP_EQUALVERIFY OP_CHECKSIG
```

Stack

(Public Key) `172D5w7C…` 

(Signature) `bbba3f5…`

How Bitcoin Script works?

(Public Key Hash)

```
OP_DUP OP_HASH160 cb1f48…  OP_EQUALVERIFY OP_CHECKSIG
```

Stack

(Public Key) `172D5w7C…`

(Signature) `bbba3f5…`

How Bitcoin Script works?

(Public Key Hash)

```
OP_HASH160 cb1f48…  OP_EQUALVERIFY OP_CHECKSIG
```

Stack

(Public Key) `172D5w7C…` 

(Public Key) `172D5w7C…`

(Signature) `bbba3f5…`

How Bitcoin Script works?

(Public Key Hash)

```
OP_HASH160 cb1f48…  OP_EQUALVERIFY OP_CHECKSIG
```

Stack

(Public Key) `172D5w7C…`

(Public Key) `172D5w7C…`

(Signature) `bbba3f5…`

How Bitcoin Script works?

(Public Key Hash)

`cb1f48…  OP_EQUALVERIFY OP_CHECKSIG`

Stack

(Public Key Hash) `cb1f48…`

(Public Key) `172D5w7C…`

(Signature) `bbba3f5…`

How Bitcoin Script works?

(Public Key Hash)

```
cb1f48…  OP_EQUALVERIFY OP_CHECKSIG
```

Stack

(Public Key Hash) `cb1f48…`

(Public Key) `172D5w7C…`

(Signature) `bbba3f5…`

How Bitcoin Script works?

`OP_EQUALVERIFY OP_CHECKSIG`

Stack

(Public Key Hash) `cb1f48...` 

(Public Key Hash) `cb1f48...`

(Public Key) `172D5w7C...`

(Signature) `bbba3f5...`

How Bitcoin Script works?

`OP_EQUALVERIFY OP_CHECKSIG`

Stack

(Public Key Hash) `cb1f48...` 

(Public Key Hash) `cb1f48...` 

(Public Key) `172D5w7C...`

(Signature) `bbba3f5...`

How Bitcoin Script works?

OP_CHECKSIG

Stack

(Public Key) 172D5w7C...

(Signature) bbba3f5...

How Bitcoin Script works?

OP_CHECKSIG

Stack

(Public Key) 172D5w7C...

(Signature) bbba3f5...

How Bitcoin Script works?

(EMPTY)

Stack

(EMPTY)

SUCCESS!

The sky is the limit! (Odd/Even Bet)

```
<Sig> <PubKeyB> <7bb> <5aa> OP_2DUP OP_HASH160 <aHash> OP_EQUALVERIFY OP_SWAP
OP_HASH160 <bHash> OP_EQUALVERIFY <1> OP_LEFT OP_SWAP <1> OP_LEFT OP_ADD <2>
OP_SWAP OP_MOD
OP_IF
    OP_DUP OP_HASH160 <pubKeyAHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ELSE
    OP_DUP OP_HASH160 <pubKeyBHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ENDIF
```

The sky is the limit! (Odd/Even Bet)

```
<Sig> <PubKeyB> <7bb> <5aa> OP_2DUP OP_HASH160 <aHash> OP_EQUALVERIFY OP_SWAP
OP_HASH160 <bHash> OP_EQUALVERIFY <1> OP_LEFT OP_SWAP <1> OP_LEFT OP_ADD <2>
OP_SWAP OP_MOD
OP_IF
    OP_DUP OP_HASH160 <pubKeyAHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ELSE
    OP_DUP OP_HASH160 <pubKeyBHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ENDIF
```

@_sortega

The sky is the limit! (Odd/Even Bet)

```
<Sig> <PubKeyB> <7bb> <5aa> OP_2DUP OP_HASH160 <aHash> OP_EQUALVERIFY OP_SWAP
OP_HASH160 <bHash> OP_EQUALVERIFY <1> OP_LEFT OP_SWAP <1> OP_LEFT OP_ADD <2>
OP_SWAP OP_MOD
OP_IF
    OP_DUP OP_HASH160 <pubKeyAHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ELSE
    OP_DUP OP_HASH160 <pubKeyBHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ENDIF
```

Stack

The sky is the limit! (Odd/Even Bet)

```
<PubKeyB> <7bb> <5aa> OP_2DUP OP_HASH160 <aHash> OP_EQUALVERIFY OP_SWAP
OP_HASH160 <bHash> OP_EQUALVERIFY <1> OP_LEFT OP_SWAP <1> OP_LEFT OP_ADD <2>
OP_SWAP OP_MOD
OP_IF
    OP_DUP OP_HASH160 <pubKeyAHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ELSE
    OP_DUP OP_HASH160 <pubKeyBHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ENDIF
```

Stack

```
<Sig>
```

The sky is the limit! (Odd/Even Bet)

```
                <7bb> <5aa> OP_2DUP OP_HASH160 <aHash> OP_EQUALVERIFY OP_SWAP
OP_HASH160 <bHash> OP_EQUALVERIFY <1> OP_LEFT OP_SWAP <1> OP_LEFT OP_ADD <2>
OP_SWAP OP_MOD
OP_IF
    OP_DUP OP_HASH160 <pubKeyAHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ELSE
    OP_DUP OP_HASH160 <pubKeyBHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ENDIF
```

Stack

```
<PubKey>
<Sig>
```

The sky is the limit! (Odd/Even Bet)

```
                    <5aa> OP_2DUP OP_HASH160 <aHash> OP_EQUALVERIFY OP_SWAP
OP_HASH160 <bHash> OP_EQUALVERIFY <1> OP_LEFT OP_SWAP <1> OP_LEFT OP_ADD <2>
OP_SWAP OP_MOD
OP_IF
    OP_DUP OP_HASH160 <pubKeyAHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ELSE
    OP_DUP OP_HASH160 <pubKeyBHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ENDIF
```

Stack

```
<7bb>
<PubKey>
<Sig>
```

The sky is the limit! (Odd/Even Bet)

```
                                OP_2DUP OP_HASH160 <aHash> OP_EQUALVERIFY OP_SWAP
OP_HASH160 <bHash> OP_EQUALVERIFY <1> OP_LEFT OP_SWAP <1> OP_LEFT OP_ADD <2>
OP_SWAP OP_MOD
OP_IF
    OP_DUP OP_HASH160 <pubKeyAHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ELSE
    OP_DUP OP_HASH160 <pubKeyBHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ENDIF
```

Stack

```
<5aa>
<7bb>
<PubKeyB>
<Sig>
```

The sky is the limit! (Odd/Even Bet)

```
                                    OP_HASH160 <aHash> OP_EQUALVERIFY OP_SWAP
OP_HASH160 <bHash> OP_EQUALVERIFY <1> OP_LEFT OP_SWAP <1> OP_LEFT OP_ADD <2>
OP_SWAP OP_MOD
OP_IF
    OP_DUP OP_HASH160 <pubKeyAHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ELSE
    OP_DUP OP_HASH160 <pubKeyBHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ENDIF
```

Stack

```
<5aa>
<7bb>
<5aa>
<7bb>
<PubKeyB>
<Sig>
```

The sky is the limit! (Odd/Even Bet)

```
                                                    <aHash> OP_EQUALVERIFY OP_SWAP
OP_HASH160 <bHash> OP_EQUALVERIFY <1> OP_LEFT OP_SWAP <1> OP_LEFT OP_ADD <2>
OP_SWAP OP_MOD
OP_IF
    OP_DUP OP_HASH160 <pubKeyAHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ELSE
    OP_DUP OP_HASH160 <pubKeyBHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ENDIF
```

Stack

```
<aHash>
<7bb>
<5aa>
<7bb>
<PubKeyB>
<Sig>
```

The sky is the limit! (Odd/Even Bet)

```
                                                    OP_EQUALVERIFY OP_SWAP
OP_HASH160 <bHash> OP_EQUALVERIFY <1> OP_LEFT OP_SWAP <1> OP_LEFT OP_ADD <2>
OP_SWAP OP_MOD
OP_IF
    OP_DUP OP_HASH160 <pubKeyAHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ELSE
    OP_DUP OP_HASH160 <pubKeyBHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ENDIF
```

Stack

```
<aHash>
<aHash>
<5aa>
<7bb>
<7bb>
<PubKeyB>
<Sig>
```

The sky is the limit! (Odd/Even Bet)

```
                                                                    OP_SWAP
OP_HASH160 <bHash> OP_EQUALVERIFY <1> OP_LEFT OP_SWAP <1> OP_LEFT OP_ADD <2>
OP_SWAP OP_MOD
OP_IF
    OP_DUP OP_HASH160 <pubKeyAHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ELSE
    OP_DUP OP_HASH160 <pubKeyBHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ENDIF
```

Stack

<5aa>

<7bb>

<7bb>

<PubKeyB>

<Sig>

The sky is the limit! (Odd/Even Bet)

```
OP_HASH160 <bHash> OP_EQUALVERIFY <1> OP_LEFT OP_SWAP <1> OP_LEFT OP_ADD <2>
OP_SWAP OP_MOD
OP_IF
    OP_DUP OP_HASH160 <pubKeyAHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ELSE
    OP_DUP OP_HASH160 <pubKeyBHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ENDIF
```

Stack

```
<7bb>
<5aa>
<7bb>
<PubKeyB>
<Sig>
```

The sky is the limit! (Odd/Even Bet)

```
            <bHash> OP_EQUALVERIFY <1> OP_LEFT OP_SWAP <1> OP_LEFT OP_ADD <2>
OP_SWAP OP_MOD
OP_IF
    OP_DUP OP_HASH160 <pubKeyAHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ELSE
    OP_DUP OP_HASH160 <pubKeyBHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ENDIF
```

Stack

```
<bHash>
<5aa>
<7bb>
<PubKeyB>
<Sig>
```

The sky is the limit! (Odd/Even Bet)

```
                    OP_EQUALVERIFY <1> OP_LEFT OP_SWAP <1> OP_LEFT OP_ADD <2>
OP_SWAP OP_MOD
OP_IF
    OP_DUP OP_HASH160 <pubKeyAHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ELSE
    OP_DUP OP_HASH160 <pubKeyBHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ENDIF
```

Stack

```
<bHash>
<bHash>
<5aa>
<7bb>
<PubKeyB>
<Sig>
```

The sky is the limit! (Odd/Even Bet)

```
                                    <1> OP_LEFT OP_SWAP <1> OP_LEFT OP_ADD <2>
OP_SWAP OP_MOD
OP_IF
    OP_DUP OP_HASH160 <pubKeyAHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ELSE
    OP_DUP OP_HASH160 <pubKeyBHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ENDIF
```

Stack

```
<5aa>
<7bb>
<PubKeyB>
<Sig>
```

The sky is the limit! (Odd/Even Bet)

```
                                        OP_LEFT OP_SWAP <1> OP_LEFT OP_ADD <2>
OP_SWAP OP_MOD
OP_IF
    OP_DUP OP_HASH160 <pubKeyAHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ELSE
    OP_DUP OP_HASH160 <pubKeyBHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ENDIF
```

Stack

```
<1>
<5aa>
<7bb>
<PubKeyB>
<Sig>
```

The sky is the limit! (Odd/Even Bet)

```
                                                    OP_SWAP <1> OP_LEFT OP_ADD <2>
OP_SWAP OP_MOD
OP_IF
    OP_DUP OP_HASH160 <pubKeyAHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ELSE
    OP_DUP OP_HASH160 <pubKeyBHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ENDIF
```

Stack

```
<5>
<7bb>
<PubKeyB>
<Sig>
```

The sky is the limit! (Odd/Even Bet)

```
                                                    <1> OP_LEFT OP_ADD <2>
OP_SWAP OP_MOD
OP_IF
    OP_DUP OP_HASH160 <pubKeyAHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ELSE
    OP_DUP OP_HASH160 <pubKeyBHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ENDIF
```

Stack

```
<7bb>
<5>
<PubKeyB>
<Sig>
```

The sky is the limit! (Odd/Even Bet)

```
                                                    OP_LEFT OP_ADD <2>
OP_SWAP OP_MOD
OP_IF
    OP_DUP OP_HASH160 <pubKeyAHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ELSE
    OP_DUP OP_HASH160 <pubKeyBHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ENDIF
```

Stack

```
<1>
<7bb>
<5>
<PubKeyB>
<Sig>
```

The sky is the limit! (Odd/Even Bet)

```
                                                    OP_ADD <2>
OP_SWAP OP_MOD
OP_IF
    OP_DUP OP_HASH160 <pubKeyAHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ELSE
    OP_DUP OP_HASH160 <pubKeyBHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ENDIF
```

Stack

```
<7>
<5>
<PubKeyB>
<Sig>
```

The sky is the limit! (Odd/Even Bet)

```
                                                                    <2>
OP_SWAP OP_MOD
OP_IF
    OP_DUP OP_HASH160 <pubKeyAHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ELSE
    OP_DUP OP_HASH160 <pubKeyBHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ENDIF
```

Stack

<12>

<PubKeyB>

<Sig>

The sky is the limit! (Odd/Even Bet)

```
OP_SWAP OP_MOD
OP_IF
    OP_DUP OP_HASH160 <pubKeyAHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ELSE
    OP_DUP OP_HASH160 <pubKeyBHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ENDIF
```

Stack

```
<2>
<12>
<PubKeyB>
<Sig>
```

The sky is the limit! (Odd/Even Bet)

```
OP_MOD
OP_IF
    OP_DUP OP_HASH160 <pubKeyAHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ELSE
    OP_DUP OP_HASH160 <pubKeyBHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ENDIF
```

Stack

```
<12>
<2>
<PubKeyB>
<Sig>
```

The sky is the limit! (Odd/Even Bet)

```
OP_IF
    OP_DUP OP_HASH160 <pubKeyAHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ELSE
    OP_DUP OP_HASH160 <pubKeyBHash> OP_EQUALVERIFY OP_CHECKSIG
OP_ENDIF
```

Stack

```
<0>
<PubKeyB>
<Sig>
```

The sky is the limit! (Odd/Even Bet)

```
OP_DUP OP_HASH160 <pubKeyBHash> OP_EQUALVERIFY OP_CHECKSIG
```

Stack

```
<PubKeyB>
<Sig>
```

(Standard Script)

The sky is the limit! (Odd/Even Bet)

- Possible real use:
  - Random user to pay the transaction fees.
  - Help to define asymetric scenario (secrets)

Context: Oracles

- The oracle contract allow define how the money is spended including a external state.
  - Allows to make reversible transactions.
  - Allow to pay, only if a external condition is true, for example the result of a search in google, or the API SEUR response.

Context: Oracles

- Example of a reversible payment using a Oracle
  - First you generate (in private) a multisig transaction as this:

```
TX1
in  { 1 BTC BOB }
out { 1 BTC MULTISIGVERIFY BOB SAM }
```

  - Then you must obtain signed from counterpart:

```
TX2
in  { TX1[0] }
out { 1 BTC

<SeurAPI(trackID) == 'RETURNED'>
OP_HASH160 <ExternalScriptHash> OP_EQUALSVERIFY
<OraclePubKey> <SamPubKey> 2 OP_CHECKMULTISIGVERIFY }
```

```
TX3
in  { TX1[0] }
out { 1 BTC

<SeurAPI(trackID) == 'DELIVERED'>
OP_HASH160 <ExternalScriptHash> OP_EQUALSVERIFY
<OraclePubKey> <BobPubKey> 2 OP_CHECKMULTISIGVERIFY }
```

Context: Oracles

- Execution

```
TX2
in  { TX1[0] }
out { 1 BTC
<SeurAPI(trackID) == 'RETURNED'>
OP_HASH160 <ExternalScriptHash>
OP_EQUALSVERIFY
<OraclePubKey> <SamPubKey> 2
OP_CHECKMULTISIGVERIFY }
```

Context: Oracles

- Execution

```
<OracleSig> <SamSig> <SeurAPI(trackID) == 'RETURNED'>
OP_HASH160 <ExternalScriptHash> OP_EQUALSVERIFY
<OraclePubKey> <SamPubKey> 2 OP_CHECKMULTISIGVERIFY
```

Stack

Context: Oracles

- Execution

```
<SamSig> <SeurAPI(trackID) == 'RETURNED'>
OP_HASH160 <ExternalScriptHash> OP_EQUALSVERIFY
<OraclePubKey> <SamPubKey> 2 OP_CHECKMULTISIGVERIFY
```

Stack

```
<OracleSig>
```

Context: Oracles

- Execution

```
                    <SeurAPI(trackID) == 'RETURNED'>
OP_HASH160 <ExternalScriptHash> OP_EQUALSVERIFY
<OraclePubKey> <SamPubKey> 2 OP_CHECKMULTISIGVERIFY
```

Stack

```
<OracleSig>
<SamSig>
```

Context: Oracles

- Execution

```
OP_HASH160 <ExternalScriptHash> OP_EQUALSVERIFY
<OraclePubKey> <SamPubKey> 2 OP_CHECKMULTISIGVERIFY
```

Stack

```
<SeurAPI(trackID) == 'RETURNED'>
<OracleSig>
<SamSig>
```

Context: Oracles

- Execution

```
<ExternalScriptHash> OP_EQUALSVERIFY
<OraclePubKey> <SamPubKey> 2 OP_CHECKMULTISIGVERIFY
```

Stack

```
<ScriptHash>
<OracleSig>
<SamSig>
```

Context: Oracles

- Execution

```
                                        OP_EQUALSVERIFY
<OraclePubKey> <SamPubKey> 2 OP_CHECKMULTISIGVERIFY
```

Stack

```
<ExternalScriptHash>
<ScriptHash>
<OracleSig>
<SamSig>
```

Context: Oracles

- Execution

```
<OraclePubKey> <SamPubKey> 2 OP_CHECKMULTISIGVERIFY
```

Stack

```
<OracleSig>
<SamSig>
```

Context: Oracles

- Execution

SUCCESS!

Stack

Fees? Micro Payment Channels!

- Transactions from a checkpoint witout broadcast
- Allows commit money safely

- Allows to pay on demand per second

- For example: decentralized WiFi Hotspots or subcontracting services in third world countries.

Micro Payment Channels Mechanic

TX1

```
IN  {0: 1 BTC                    }
OUT {0: 1 BTC MULTISIG BOB SAM }
```

Micro Payment Channels Mechanic

TX1

```
IN   {0: 1 BTC                    }
OUT  {0: 1 BTC MULTISIG BOB SAM   }
```

TX2

```
    IN   {TX1[0]: 1 BTC      }
    OUT  {     0  : 1 BTC BOB }

    LOCKTIME : 18
```

Micro Payment Channels Mechanic

TX1

```
IN   {0: 1 BTC                    }
OUT  {0: 1 BTC MULTISIG BOB SAM   }
```

TX2

```
    IN   {TX1[0]: 1 BTC      }
    OUT  {     0  : 1 BTC BOB }

    LOCKTIME : 18
```

Micro Payment Channels Mechanic

TX1

```
IN  {0: 1 BTC                    }
OUT {0: 1 BTC MULTISIG BOB SAM }
```

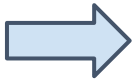
TX2

```
    IN  {TX1[0]: 1 BTC      }
    OUT {     0 : 1 BTC BOB }

    LOCKTIME : 18
```

Micro Payment Channels Mechanic

TX1

```
IN  {0: 1 BTC                          }
OUT {0: 1 BTC MULTISIG BOB SAM }
```



```
IN  {TX1[0] 1 BTC    }
OUT {0: 0,1 BTC BOB
        0,9 BTC SAM }

LOCKTIME : 10
```

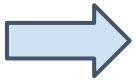
TX2

```
    IN  {TX1[0]: 1 BTC      }
    OUT {     0 : 1 BTC BOB }

    LOCKTIME : 18
```

Micro Payment Channels Mechanic

TX1

```
IN   {0: 1 BTC                          }
OUT  {0: 1 BTC MULTISIG BOB SAM }
```



```
IN   {TX1[0] 1 BTC    }
OUT  {0: 0,2 BTC BOB
         0,8 BTC SAM }

LOCKTIME : 8
```

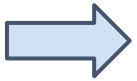
TX2

```
    IN   {TX1[0]: 1 BTC      }
    OUT  {     0 : 1 BTC BOB }

    LOCKTIME : 18
```

Micro Payment Channels Mechanic

TX1

```
IN   {0: 1 BTC                          }
OUT  {0: 1 BTC MULTISIG BOB SAM }
```



```
IN   {TX1[0] 1 BTC    }
OUT  {0: 0,3 BTC BOB
         0,7 BTC SAM }

LOCKTIME : 6
```

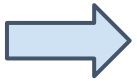
TX2

```
    IN   {TX1[0]: 1 BTC      }
    OUT  {     0 : 1 BTC BOB }

    LOCKTIME : 18
```

Micro Payment Channels Mechanic

TX1

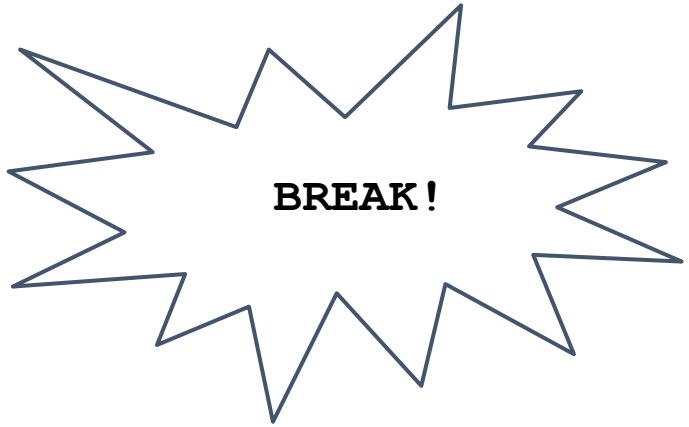
```
IN   {0: 1 BTC                          }
OUT  {0: 1 BTC MULTISIG BOB SAM }
```



```
IN   {TX1[0] 1 BTC    }
OUT  {0: 0,4 BTC BOB
         0,6 BTC SAM }

LOCKTIME : 4
```

TX2

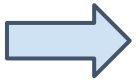
```
    IN   {TX1[0]: 1 BTC      }
    OUT  {     0 : 1 BTC BOB }

    LOCKTIME : 18
```



Micro Payment Channels Mechanic

TX1

```
IN   {0: 1 BTC                          }
OUT  {0: 1 BTC MULTISIG BOB SAM }
```



```
IN   {TX1[0] 1 BTC    }
OUT  {0: 0,4 BTC BOB
         0,6 BTC SAM }

LOCKTIME : 4
```

TX2

```
IN   {TX1[0]: 1 BTC      }
OUT  {     0 : 1 BTC BOB }

LOCKTIME : 18
```

THANKS!

- BigData is the current wave, P2P is the next.
- APIs and P2P are the next challenge on financial world.
- Bitcoin technology and protocol is here to stay.