Progress on Scaling via Client-Side Validation

Peter Todd

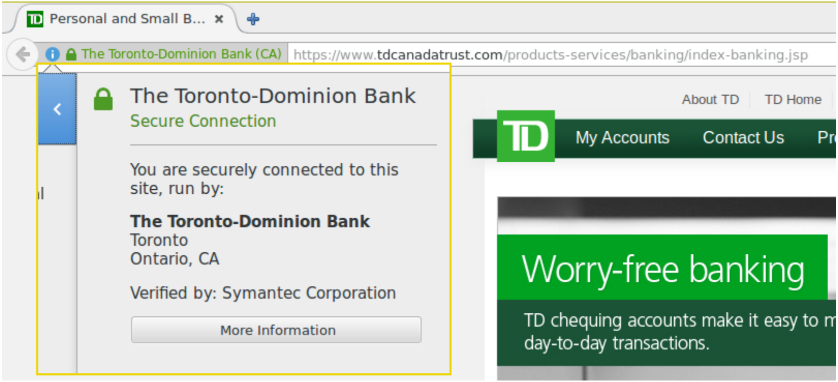
Oct 9th 2016

37EC 7D7B 0A21 7CDB 4B4E 007E 7FAB 1142 67E4 FA04

The Miner-Side Approach

```
{
  [0] "Chronos"
  (call 0x11d11764cd7f6ecda172e0b72370e6ea7f75f290
   0 0 0 32 0 0)
}
{
  ; Split the call data in groups of 32 bytes
  ; (2^256 = 2^8^32)
  ; Loop over this list with @i as an index
  (for () (< @i (/ (calldatasize) 32)) [i](+ @i 1) {
    ; Get the current hash
    [hash](calldataload (* @i 32))
    ; If the hash isn't already registered in
    ; storage, set a new entry
    (unless @@@hash [[@hash]](timestamp))
  })
}
```

What do we mean by ‘Client-Side’?



Personal and Small B... ×

The Toronto-Dominion Bank (CA) https://www.tdcanadatrust.com/products-services/banking/index-banking.jsp

The Toronto-Dominion Bank
Secure Connection

You are securely connected to this site, run by:

The Toronto-Dominion Bank
Toronto
Ontario, CA

Verified by: Symantec Corporation

More Information

Client-Side Validation Toolbox

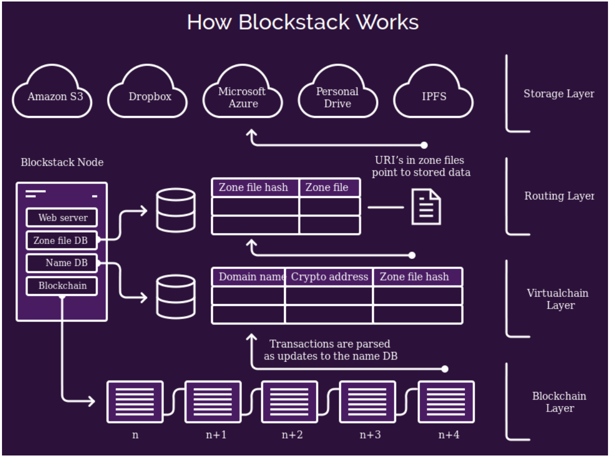
- Signatures
- Proof-of-Existence (Timestamping)
- Proof-of-Publication

Case Study: OpenTimestamps

```
$ git tag -v opentimestamps-client-v0.2.1
object fe19cd28c0685505ff3c2f6bfcb4d18abc85efa2
type commit
tag opentimestamps-client-v0.2.1
tagger Peter Todd <pete@petertodd.org> 1474872017 -0400

Release opentimestamps-client-v0.2.1
ots: Success! Bitcoin attests data existed as of
        Mon Sep 26 02:45:43 2016 EDT
ots: Good timestamp
gpg: Signature made Mon 26 Sep 2016 02:40:18 AM EDT
gpg:                using RSA key 6399011044E8AFB2
gpg: Good signature from "Peter Todd <pete@petertodd.org>"
gpg:                 aka "[jpeg image of size 5220]"
```

Case Study: Blockstack



URI's in zone files point to stored data

Zone file hash	Zone file

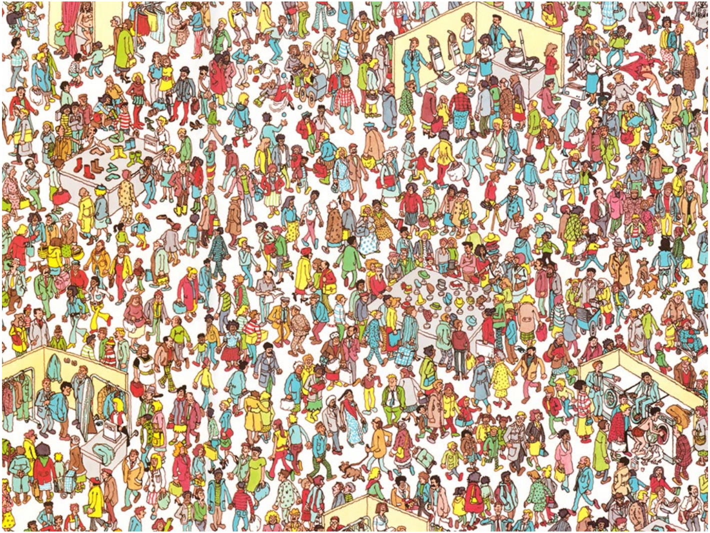
Domain name	Crypto address	Zone file hash

Transactions are parsed as updates to the name DB

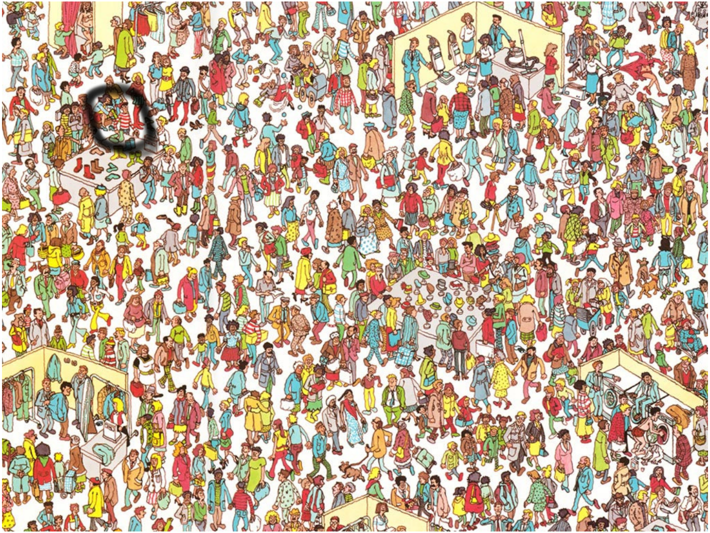
Client-Side Bitcoin

Do miners need to validate blocks?

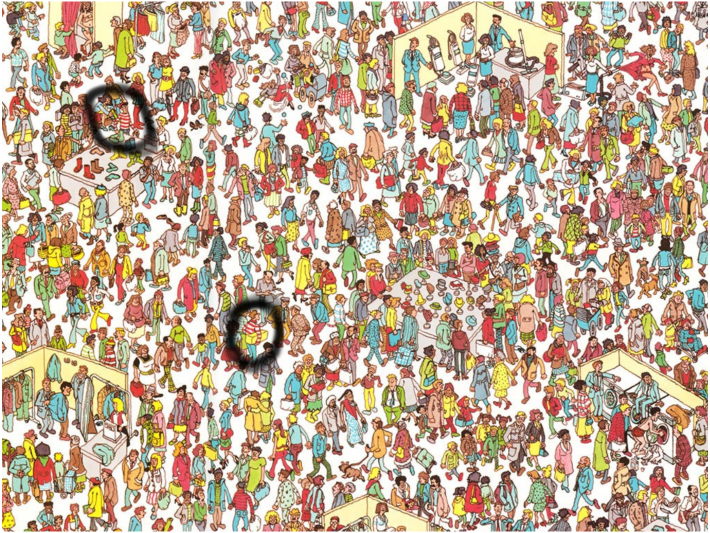
Where's Waldo? - The double-spend problem



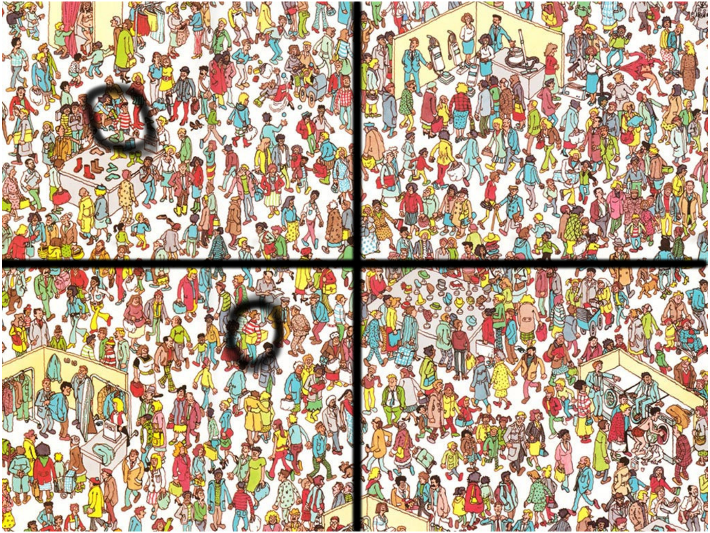
Where's Waldo? - The double-spend problem



Where's Waldo? - The double-spend problem

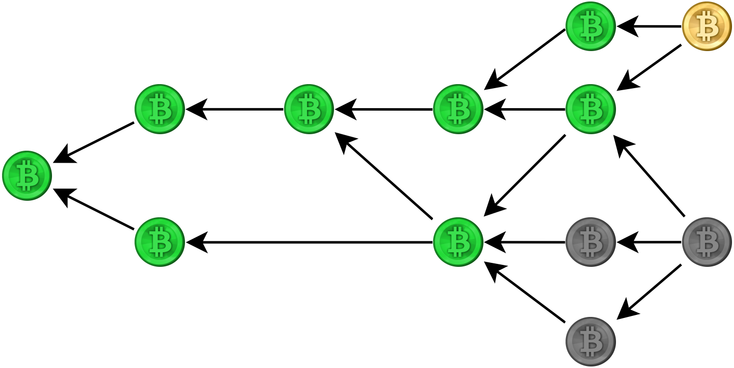


Where's Waldo? - Split chains

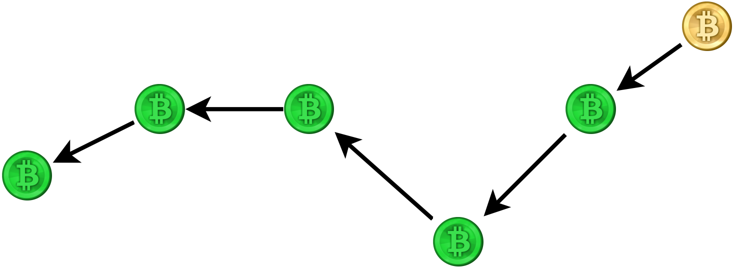


Where's Waldo? - Sharded chains

Transaction History Proof



Transaction History Linearization



Transaction History Linearization

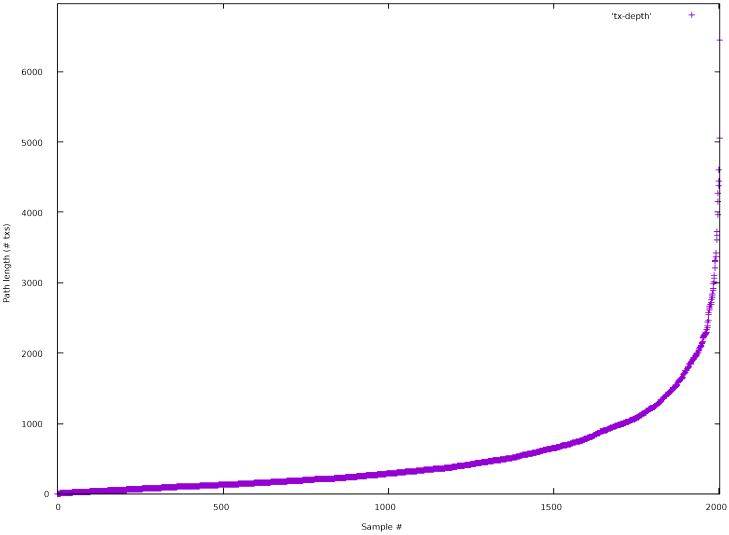
$$x = \sum \text{fake inputs} \tag{1}$$

$$y = \sum \text{real inputs} \tag{2}$$

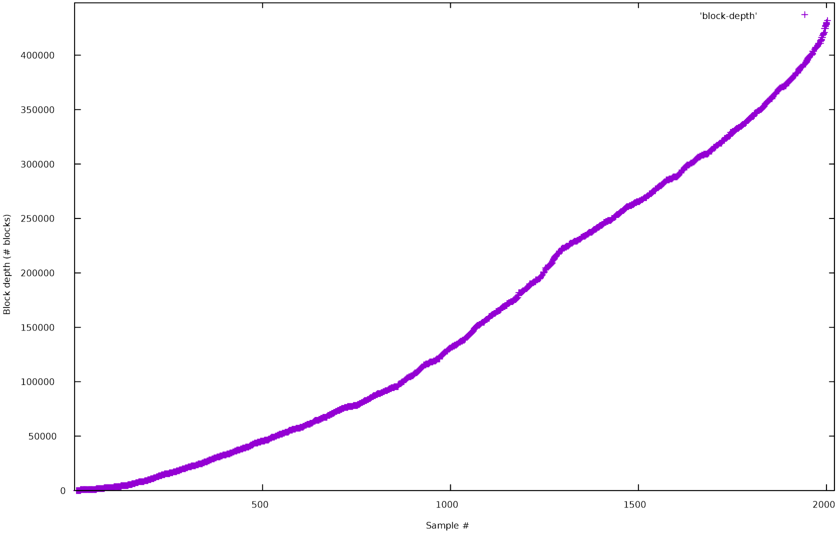
Transaction History Linearization

$$
\begin{aligned}
E_x &= x(1 - \frac{x}{x+y}) - y(\frac{x}{x+y}) && (3)\\
&= x(\frac{x+y}{x+y} - \frac{x}{x+y}) - y(\frac{x}{x+y}) && (4)\\
&= x(\frac{y}{x+y}) - y(\frac{x}{x+y}) && (5)\\
&= 0 && (6)
\end{aligned}
$$

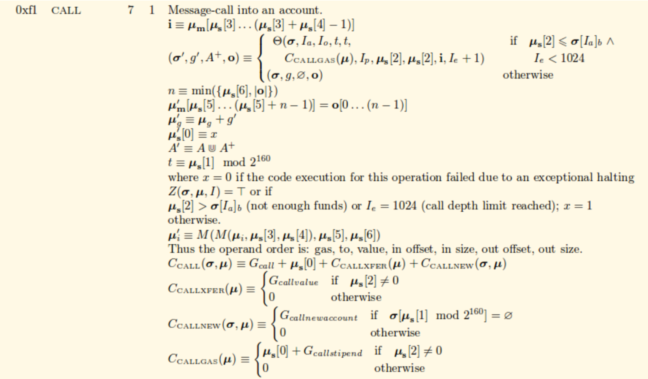
Linearization Simulation



Linearization Simulation



Defining Protocols



0xf1 CALL 7 1 Message-call into an account.

$$\mathbf{i} \equiv \boldsymbol{\mu}_{\mathbf{m}}[\boldsymbol{\mu}_{\mathbf{s}}[3] \dots (\boldsymbol{\mu}_{\mathbf{s}}[3] + \boldsymbol{\mu}_{\mathbf{s}}[4] - 1)]$$

$$(\boldsymbol{\sigma}', g', A^{+}, \mathbf{o}) \equiv \begin{cases} \Theta(\boldsymbol{\sigma}, I_a, I_o, t, t, & \text{if} \quad \boldsymbol{\mu}_{\mathbf{s}}[2] \leqslant \boldsymbol{\sigma}[I_a]_b \wedge \\ \quad C_{\textsc{callgas}}(\boldsymbol{\mu}), I_p, \boldsymbol{\mu}_{\mathbf{s}}[2], \boldsymbol{\mu}_{\mathbf{s}}[2], \mathbf{i}, I_e + 1) & \quad I_e < 1024 \\ (\boldsymbol{\sigma}, g, \varnothing, \mathbf{o}) & \text{otherwise} \end{cases}$$

$$n \equiv \min(\{\boldsymbol{\mu}_{\mathbf{s}}[6], |\mathbf{o}|\})$$

$$\boldsymbol{\mu}'_{\mathbf{m}}[\boldsymbol{\mu}_{\mathbf{s}}[5] \dots (\boldsymbol{\mu}_{\mathbf{s}}[5] + n - 1)] = \mathbf{o}[0 \dots (n - 1)]$$

$$\boldsymbol{\mu}'_g \equiv \boldsymbol{\mu}_g + g'$$

$$\boldsymbol{\mu}'_{\mathbf{s}}[0] \equiv x$$

$$A' \equiv A \Cup A^{+}$$

$$t \equiv \boldsymbol{\mu}_{\mathbf{s}}[1] \mod 2^{160}$$

where $x = 0$ if the code execution for this operation failed due to an exceptional halting $Z(\boldsymbol{\sigma}, \boldsymbol{\mu}, I) = \top$ or if $\boldsymbol{\mu}_{\mathbf{s}}[2] > \boldsymbol{\sigma}[I_a]_b$ (not enough funds) or $I_e = 1024$ (call depth limit reached); $x = 1$ otherwise.

$$\boldsymbol{\mu}'_i \equiv M(M(\boldsymbol{\mu}_i, \boldsymbol{\mu}_{\mathbf{s}}[3], \boldsymbol{\mu}_{\mathbf{s}}[4]), \boldsymbol{\mu}_{\mathbf{s}}[5], \boldsymbol{\mu}_{\mathbf{s}}[6])$$

Thus the operand order is: gas, to, value, in offset, in size, out offset, out size.

$$C_{\textsc{call}}(\boldsymbol{\sigma}, \boldsymbol{\mu}) \equiv G_{call} + \boldsymbol{\mu}_{\mathbf{s}}[0] + C_{\textsc{callxfer}}(\boldsymbol{\mu}) + C_{\textsc{callnew}}(\boldsymbol{\sigma}, \boldsymbol{\mu})$$

$$C_{\textsc{callxfer}}(\boldsymbol{\mu}) \equiv \begin{cases} G_{callvalue} & \text{if} \quad \boldsymbol{\mu}_{\mathbf{s}}[2] \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

$$C_{\textsc{callnew}}(\boldsymbol{\sigma}, \boldsymbol{\mu}) \equiv \begin{cases} G_{callnewaccount} & \text{if} \quad \boldsymbol{\sigma}[\boldsymbol{\mu}_{\mathbf{s}}[1] \mod 2^{160}] = \varnothing \\ 0 & \text{otherwise} \end{cases}$$

$$C_{\textsc{callgas}}(\boldsymbol{\mu}) \equiv \begin{cases} \boldsymbol{\mu}_{\mathbf{s}}[0] + G_{callstipend} & \text{if} \quad \boldsymbol{\mu}_{\mathbf{s}}[2] \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

Thank you!