Solidus: Strong Confidentiality and Transparency for Blockchain Transactions

Ari Juels
Jacobs Institute, Cornell Tech
Initiative for CryptoCurrencies and Contracts (IC3)

Joint work with E. Cecchetti, F. Zhang, A. Kosba, and E. Shi
What is the blockchain?
Schools are recording students' results using bitcoin tech.
Blockchains: Abstraction
Blockchains: Abstraction
Blockchains: Abstraction

#1 Strict ordering of messages
Blockchains: Abstraction

#1 Strict ordering of messages
Blockchains: Abstraction

#2 Rule-based write, global read

Write Permission: Rule-based
Blockchains: Abstraction

#3 No message modification

MESSAGE 4
MESSAGE 3
MESSAGE 2
MESSAGE 1
Power of the Abstraction

“I, Alice, send 1 BTC to Bob --Signed Alice”
Power of the Abstraction

Message 4

Message 3

Message 2

Message 1

“Bob gave Dana 0.2 BTC”

“Bob gave Charlie 0.1 BTC”

“Alice gave Bob 1 BTC”

Alice: 2 BTC
Bob: 0.7 BTC
Charlie: 0.1 BTC
Dana: 0.2 BTC

...
Compare:
Execution, clearing, and settlement

- For transfer of financial instruments
- Up to three days to complete (T+3)
- Many middlemen
- Fragmented records
- Difficult to audit
Blockchains are much faster...

Alice: 2 BTC
Bob: 0.7 BTC
Carol: 0.1 BTC
Dana: 0.2 BTC

"Alice gave Bob 1 BTC"
"Bob gave Carol 0.1 BTC"
"Bob gave Dana 0.2 BTC"

...
and more transparent...

Alice: 2 BTC  
Bob: 0.7 BTC  
Carol: 0.1 BTC  
Dana: 0.2 BTC  

“Alice gave Bob 1 BTC”  
“Bob gave Carol 0.1 BTC”  
“Bob gave Dana 0.2 BTC”  

Message 1  
Message 2  
Message 3  
Message 4
Mail delivery in 19th-century United States

28 Oct. 1861

26 Oct. 1861
Shouldn’t blockchains just kill existing settlement systems?

28 Oct. 1861

26 Oct. 1861
The Challenge of Confidentiality
Transparency isn’t all good

“Alice gave Bob 1 BTC”

“Bob gave Carol 0.1 BTC”

“Bob gave Darlene 0.2 BTC”

Alice: 2 BTC  
Bob: 0.7 BTC  
Carol: 0.1 BTC  
Darlene: 0.2 BTC
Transparency vs. Confidentiality
Full transparency / auditability: All transaction data is on the blockchain!
Confidentiality and blockchain transparency?

Alice

Bob

Carol

Dana

$50

$200

$500

$500
Confidentiality idea #1: Pseudonyms à la Bitcoin

A

1 BTC

0.1 BTC

B

C

0.2 BTC

D
Confidentiality idea #1: Pseudonyms à la Bitcoin

A

1 BTC

0.1 BTC

C

B

D

0.2 BTC
Confidentiality idea #2: Confidential Transactions [Max15]

• Represent balance $a$ as commitment
  – $\text{Com}(a)$

• Pedersen commitments have homomorphic property:
  – $\text{Com}(a) \cdot \text{Com}(b) = \text{Com}(a+b)$
Idea #2: Confidential Transactions
[Max15]

• Represent balance $a$ as commitment
  – $\text{Com}(a)$

• Pedersen commitments have homomorphic property:
  – $\text{Com}(a) ~ \text{Com}(b) = \text{Com}(a+b)$

• Details on Pedersen:
  – Cycle subgroup $G$ of prime order $q$ for which discrete log is hard
  – Generators $g, h$ such that $\log_g h$ unknown
  – $\text{Com}(a) = h^{ag^r}$ for random $r$
Confidential Transactions

A
Com(a)

B
Com(b)
Confidential Transactions

To send $x$ confidentially...

A
Com(a)

B
Com(b)
Confidential Transactions

Com(a) → Com(x) → Com(b)

A

B
Confidential Transactions

\[ \text{Com}(a') \leftarrow \text{Com}(a) / \text{Com}(x) \]

\[ \text{Com}(b') \leftarrow \text{Com}(b) \text{ Com}(x) \]
Confidential Transactions

\[ \text{Com}(a') \text{ for } a' = a - x \]

\[ \text{Com}(b') \text{ for } b' = b + x \]
To prove correctness

\[ \text{Com}(x) + \text{ZK-proof: } x \leq a \]

\[ \text{Com}(a') \text{ for } a' = a - x \]

\[ \text{Com}(b') \text{ for } b' = b + x \]
But *transaction graph* still visible!

$ on 4 June

$ on 6 June

$ on 7 June
Transaction graph leakage: Example [FKP15]
Another example

- WannaCry ransomware
- Bitcoin addresses:
  - 13AM4VW2dhxYgXeQepoHkHSQuy6NgaEb94
  - 12t9YDPgwueZ9NyMgw519p7AA8isjr6SMw
  - 115p7UMMngoj1pMvkpHijcRdfJNXj6LrLn
- Showed they didn’t know who paid ransom!
Confidentiality idea #3: Zcash [BCGGMTV14]

To spend coin \( C \) ("pour"):

- Create new coins \( C_1 \) and \( C_2 \) (worth \( \$v_1 \) and \( \$v_2 \)) assigned to receiver
- Reveal: \( \#sn \), Com(\( C_1 \)), and Com(\( C_2 \))
Zcash

Com(C) (#sn, $v, sk)

Merkle Tree

Com(C₁)  Com(C₂)

List of spent coin #s

...  #sn

Publish ZK *Proof* that:

– You know secret key sk for C
– There exists valid coin C in Merkle tree with #sn
– C₁ and C₂ are validly formed
– $v = v₁ + v₂
• Pros:
  • Transaction graph / amount confidentiality

• Cons:
  • Computing zk-SNARK **Proof** is expensive
    • 1min 3sec on Intel Core i7-4770 @ 3.40GHz, 8 threads
  • Lack of real transparency
    • Values *not on blockchain* (only commitments)
Solidus
Solidus setting:
Traditional financial industry

$ x = \text{any of shares of stock, cryptocurrency, dollars, commodities, etc.}$
Solidus setting:
Traditional financial industry

Blockchain / distributed ledger is permissioned, i.e., run by banks
Solidus: Transparency

Blockchain
Transparency

1. Auditor can decrypt all transactions and balances.
2. Everyone can verify that all transactions are valid.
Solidus: Confidentiality
Solidus: Confidentiality

User $A_1$  User $A_2$  User $A_3$  …

User $B_1$  User $B_2$  User $B_3$  …

Blockchain

$\$x$
1. Everyone learns which banks performed transactions.
2. A bank learns balance updates to its users’ accounts.
3. No one learns anything else!
Why are transparency + confidentiality hard?

\[
\begin{align*}
A_1 &\quad A_2 &\quad A_3 &\quad \ldots \\
\mathdollar{a}_1 &\quad \mathdollar{a}_2 &\quad \mathdollar{a}_3 &\quad \ldots \\
\mathdollar{a}_1 &\quad \mathdollar{a}_2 - x &\quad \mathdollar{a}_3 &\quad \ldots \\

B_1 &\quad B_2 &\quad B_3 &\quad \ldots \\
\mathdollar{b}_1 &\quad \mathdollar{b}_2 &\quad \mathdollar{b}_3 &\quad \ldots \\
\mathdollar{b}_1 &\quad \mathdollar{b}_2 &\quad \mathdollar{b}_3 + x &\quad \ldots \\
\end{align*}
\]
No transaction-graph confidentiality!

Blockchain
Naïve solution: Shuffle & Prove

User $A_1$, User $A_2$, User $A_3$, ...

$a_1$, $a_2$, $a_3$
Naïve solution: Shuffle & Prove
Naïve solution: Shuffle & Prove
Naïve solution: Shuffle & Prove

Transaction-graph confidentiality!
Naïve solution: Shuffle & Prove

- How to ensure valid shuffle of accounts?
- ZK proof of correct shuffle
  [FS01,N01,G03]

Transparency?
Naïve solution: Shuffle & Prove

- **Problem**: Shuffle cost is *linear* in number of elements $n$
- What if bank has thousands or millions of customers?

Transparency?
Key observation

Transaction-graph confidentiality goal:
Conceal pattern of accesses to accounts
Key observation

Confidentiality requirement:
Conceal pattern of accesses to memory
Key observation

Confidentiality requirement:
Conceal pattern of accesses to memory
Key observation

Standard solution: Oblivious RAM (ORAM)

Goal: conceal pattern of accesses to memory
ORAM in a nutshell

- Memory is shuffled after each access
- To achieve $O(\log n)$ shuffling cost (bandwidth), memory organized typically in tree...
ORAM in a nutshell
ORAM in a nutshell
ORAM in a nutshell
ORAM in a nutshell

1
M[1]

4
M[4]

M[3]

2
M[2]
ORAM in a nutshell

3

1

4

M[4]

M[1]

2

M[2]

M[3]
ORAM in a nutshell
Looks like solution! But...

ORAM:
- No transparency
  - Client proves nothing
  - Symmetric-key encryption
- Minimal client memory
- (Some) data can be hidden on client

Solidus:
- Needs transparency
  - Bank proves valid operations
  - Public-key encryption for audit
- Ample bank memory
  - Can store everything locally!
- No hiding data with bank, i.e., no private stash
Great! But...

Solidus requires a fundamental reworking of ORAM

- Solidus:
  - Needs transparency
  - Bank proves valid operations
  - Public-key encryption for audit
  - Ample bank memory
  - Can store everything locally!
  - No hiding data with bank, i.e., no private stash

- ORAM:
  - No transparency
  - Client proves nothing
  - Symmetric-key encryption
  - Minimal client memory
  - (Some) data can be hidden on client
Publicly Verifiable Oblivious RAM Machine (PVORM)

- Private memory $M$ represented as ciphertext $C$
- A function $f(u, M) \rightarrow M'$ for valid operations on $M$
- A function $\text{Update}(sk, u, C) \rightarrow C'$ that:
  - Applies $f$ to $M$
  - Outputs a new ciphertext $C'$ on $M'$
  - Outputs publicly verifiable $proof$ that $C \rightarrow C'$ from $f$
How do we implement PVORM?

• El Gamal ciphertexts
  – Supports variant of Confidential Transactions
    • To hide transaction amounts
  – Supports Generalized Schnorr Proofs (GSPs)
    • Can avoid zk-SNARKs as ZKPs
• Circuit ORAM [WCS16] structure
  – Swaps elements in tree; easy to prove correct with GSPs [JJ99]
• Security proof for PVORM implementation under DDH in the ROM
Algorithm ElGamal-Swap((c₀, c₁), pk, s):
parse (c₀, c₁) = ((α₀, β₀), (α₁, β₁));
r₀ ← Z_p, r₁ ← Z_p;
if s = NoSwap
  c₀' = (α₀', β₀') ← (α₀pk₀, β₀g₀);
  c₁' = (α₁', β₁') ← (α₁pk₁, β₁g₁)
else // s = Swap
  c₀' = (α₀', β₀') ← (α₁pk₀, β₁g₀);
  c₁' = (α₁', β₁') ← (α₀pk₁, β₁g₁);
output (c₀', c₁')

Bulk of the work
Solidus: The big picture

\[ f(x, M) = \text{transaction execution} \]
Transaction lifecycle

User $A_i$

Bank $A$

Bank $B$

Request

Verify & Prepare

PVORM Update

Sign

Settle

Ledger

Time
Solidus: Properties

• Transparency:
  – Anyone can verify that transactions are valid
  – Auditor can decrypt PVORM

• Confidentiality:
  – Transaction-graph confidentiality:
    • Only a user’s bank can identify a participating user
  – Transaction amounts are concealed
  – Bank-to-bank transactions can be hidden with padding

• Proof of security in Universal Composability framework
Implementation and performance

- ORAM is “practical” in academic sense
- PVORM practical for real blockchain applications in Solidus...

Solidus performance. Each bank has one c4.8xlarge EC2 instance. It runs a ZooKeeper node and maintains a PVORM with size-3 buckets, a size-25 stash, and capacity $2^{15}$. 
Implementation and performance

• Solidus is highly parallelizable!
  – In principle, banks scale up arbitrarily...
  – Bandwidth is bottleneck (1000 tx/sec over fast link)
• Also implemented zk-SNARK PVORM
  – Proofs about 100x slower, but 100x+ more compact
  – Verification about 100x faster

<table>
<thead>
<tr>
<th>Number of Threads</th>
<th>1</th>
<th>4</th>
<th>36</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proof Time (sec)</td>
<td>65.45</td>
<td>24.53</td>
<td>13.76</td>
</tr>
<tr>
<td>Verification Time</td>
<td>0.0065 sec</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Proof Size</td>
<td>288 bytes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Peak Memory Use</td>
<td>7.2 GB</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

zk-SNARK-based PVORM performance on c4.8xlarge EC2 instance.
How did Solidus get started?

• The one approach to confidentiality I didn’t mention... the current industry approach
• JPMC Quorum, Digital Asset platform, etc.

“I, Alice, send $1 to Bob”
--Signed Alice
To learn more
Visit www.initc3.org or ic3.cornell.edu