Obliviously Managed Transactions

Presentation to October 2024 DeSci for Sustainability Conference

Geoff Goodell (University College London)

18 October 2024

g.goodell@ucl.ac.uk

Agenda

- Oblivious ledgers
- Using assets with oblivious ledgers
  - Creating assets
  - Updating assets
  - Transferring assets
- **Privacy:** Chaumian tokens as assets
- **Authoritativeness:** Oblivious distributed ledgers

Oblivious ledgers

Consider an associative array mapping keys k_i to values v_i, as follows:

$$(k_0 \to v_0, k_1 \to v_1, \cdots, k_n \to v_n) \quad (1)$$

Define a **root** G as the output of a well-known hash function h applied to the associative array:

$$G = h(k_0 \to v_0, k_1 \to v_1, \cdots, k_n \to v_n) \quad (2)$$

Consider a ledger operator who accepts contributions of key-value pairs from its clients and incorporates those pairs into an associative array.

An **oblivious ledger** L is a sequence of roots generated by a ledger operator at discrete time intervals, as follows:

$$G_{L,0}, G_{L,1}, \cdots, G_{L,t} \quad (3)$$

Oblivious ledgers are not new


NOTICES & LOST AND FOUND
(5100-5102)

Universal Registry Entries:
Zone 2 -
dS8492cgVOFAoP9kyE1XzMOrQ
HgEwzkVbVafNylkUz99qvq8/ME
p5y9EFSG8XxzMBalGQQ==
Zone 3 -
JnFCg+HCmvhj8GmmUP7VZna71
NgZup/RfuKUQNzCHWXMuqLK
durxHQV5pSHLqBGPRly+mg==
These base64-encoded values represent the combined fingerprints of all digital records notarized by Surety between 2009-06-03Z 2009-06-09Z.
www.surety.com 571-748-5800

Surety, an American firm founded in 1994, uses an oblivious ledger to provide a **notarisation** service wherein clients provide hashes of their transactions, and the ledger operator publishes a root once per week.

Notarisation requires a way to prove that a transaction has been included.

- With **Merkle trees**, proofs of <u>inclusion</u> can be done efficiently, $O(lg\ n)$.
- With **Merkle tries**, proofs of <u>exclusion</u> can be done just as fast.

It is a "blockchain" data structure, but not decentralised; the central operator (Surety) must be trusted to not equivocate.

Creating assets

The initial owner of an asset begins by creating a vector F_0 containing three fields, as follows:

$$F_0 \leftarrow \boxed{(u_0, G_{L,i}, k_1)} \tag{4}$$

- u_0 : an arbitrary message (may be empty)
- $G_{L,i}$: a reference to a specific root i of an oblivious ledger L
- k_1 : the public key matching a new, one-time private key k_1^*

The initial owner then creates an asset by combining the vector F_0 with a genesis signature, as follows:

$$A_0 \leftarrow (\boxed{F_0}, s(h(\boxed{F_0}), k_0)) \tag{5}$$

- $s(d, k)$: signed copy of some data d verifiable by a public key k
- k_0 : a long-term key held by some issuer (or the initial owner, if the owner is allowed to create assets)

Updating assets

To update an asset, an owner must **register** the update with an **integrity provider** (notarisation service). The owner at sequence number j must create an *update vector* F_j containing three fields, as follows:

$$F_j \leftarrow \boxed{(u_j, G_{L,i}, k_{j+1})} \tag{6}$$

- u_j : an indication of the type or nature of the update
- $G_{L,i}$: a reference to a specific root i of an oblivious ledger L (or empty, if the current ledger is to be retained)
- k_{j+1} : the public key matching a new, one-time private key k^*_{j+1}

The asset owner then signs the update vector, creating an update A_j:

$$A_j \leftarrow (\boxed{F_j}, s(h(\boxed{F_j}), k_j)) \tag{7}$$

Finally, the asset owner submits k_j and $s(h(\boxed{F_j}), k_j)$ to an integrity provider in exchange for a **proof of inclusion**.

Transferring assets

If an owner at sequence number j (the "old owner") wishes to transfer an asset to a new owner (at sequence number $j + 1$), then the owner of sequence number j must create an update vector:

$$F_j \leftarrow \boxed{(u_j, G_{L,i}, k_{j+1})} \quad (8)$$

To complete the transfer, k_{j+1} must be provided by the new owner. Then, the old owner creates the update:

$$A_j \leftarrow (\boxed{F_j}, s(h(\boxed{F_j}), k_j)) \quad (9)$$

Once the old owner shares $(A_0, \cdots, A_j)$ with the new owner, the new owner has **possession** of the asset.

Once k_j and $s(h(\boxed{F_j}), k_j)$ are shared with the integrity provider specified in F_{j-1}, the new owner has **control** of the asset. (It is possible for the new owner to have control without possession.)

Privacy

Following Chaum[1], suppose that we have a function b such that:

$$b^{-1}(s(b(d),k)) = s(d,k) \tag{10}$$

Then, the initial owner of an asset can send $b(h(\boxed{F_0}))$ to an issuer, who will be able to create the signature $s(b(h(\boxed{F_0})),k_0)$. The initial owner can then apply b^{-1}, yielding $s(h(\boxed{F_0}),k_0)$.

If the asset A_0 represents a fungible token, then the initial owner can transfer this token without revealing its identity or any pseudonym.

Alternatively, the initial owner can furnish to the new owner a zero-knowledge proof linking the vector $\boxed{F_j}$ to the ledger G_L:

$$\textbf{zk}(\boxed{F_j}, G_L) \tag{11}$$

[1] D Chaum, Blind Signatures for Untraceable Payments. http://www.hit.bme.hu/~buttyan/courses/BMEVIHIM219/2009/Chaum.BlindSigForPayment.1982.PDF

Authoritative records

There is a persistent risk that notarisation services might **equivocate** by producing two different versions of some $G_{L,i}$.

In traditional records management systems, the **authoritativeness** of a record depends upon indexing the identity of the records manager in some juridical context.

In **distributed ledger systems**, the authoritativeness of a record depends upon the immutability of the ledger, which is an emergent property of the independence of its validators.

Preventing equivocation with DLT

From time to time, integrity providers may commit their roots $G_{L,i}$ to a distributed ledger, which can combine the roots produced by n different ledgers at a given time t, as follows:

$$G_{D,t} = h(k_{L_1} \to G_{L_1,t}, k_{L_2} \to G_{L_2,t}, \cdots, k_{L_n} \to G_{L_n,t}) \qquad (12)$$

Proofs of inclusion or exclusion of integrity provider roots can be furnished by participants in the DLT system, and they can be combined with proofs provided by integrity providers to provide assurance that the integrity providers did not equivocate.

In general, proofs can be **stacked**: roots of integrity providers or DLT systems can be committed into ledgers successively, without limit.

Further reading

G Goodell, D Toliver, and H Nakib. **'A Scalable Architecture for Electronic Payments.'** Presented at WTSC, Grenada, May 2022. In: S Matsuo et al., Financial Cryptography and Data Security. FC 2022 International Workshops. FC 2022. *Lecture Notes in Computer Science*, volume 13412, 2023. Springer, Cham. `https://doi.org/10.1007/978-3-031-32415-4_38`

G Goodell, H Nakib, and T Aste. **'Retail Central Bank Digital Currency: Motivations, Opportunities, and Mistakes.'** March 2024. To appear, *International Journal of Political Economy* (2025). `https://doi.org/10.2139/ssrn.4769226`

D Friolo, G Goodell, D Toliver, and H Nakib. **'Private Electronic Payments with Self-Custody and Zero-Knowledge Verified Reissuance.'** Working paper, October 2024. `https://doi.org/10.48550/arXiv.2409.01958`

G Goodell. **'Token-Based Payment Systems.'** July 2021. To appear, *Elgar Encyclopedia of Cryptocurrencies, Blockchain, and DLT*. `https://doi.org/10.48550/arXiv.2207.07530`

G Goodell, H Nakib, and P Tasca. **'A Digital Currency Architecture for Privacy and Owner-Custodianship.'** *Future Internet* 2021, 13(5), May 2021. `https://doi.org/10.3390/fi13050130`

G Goodell and T Aste. **'Can Cryptocurrencies Preserve Privacy and Comply with Regulations?'** *Frontiers in Blockchain*, May 2019. `https://doi.org/10.3389/fbloc.2019.00004`

Thank You

Be a part of this story!

The UCL **Future of Money** Initiative

Collaboration and partnership opportunities

`g.goodell@ucl.ac.uk`