Zupply: Anonymously Maintained Decentralized DAG Data Record Over Public Blockchains

Mohammadtaghi Badakhshan, Guang Gong

{mbadakhshan, ggong}@uwaterloo.ca

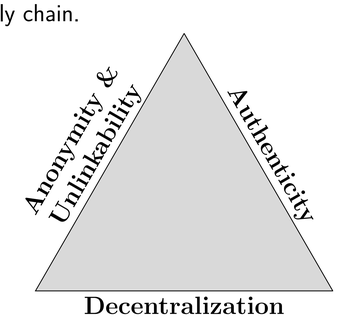
Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, N2L3G1, CANADA

Introduction

Zupply

Supply Chain Zero-knowledge Proof

Zupply is a decentralized **Anonymous Authentication Token framework** leveraging **zero-knowledge proofs** to ensure Authenticity, Anonymity, Unlinkability, and Decentralization of product history in **Supply Chain Management (SCM)** applications. The product history is structured as a **Directed Acyclic Graph (DAG)**, ensuring the secure and tamper-proof recording of events across the supply chain.



Problem Statement and Motivation

Centralized SCM systems, controlled by companies like Amazon, pose collaboration challenges, limit small retailers, and raise privacy concerns over sharing sensitive business data. Small entities struggle to compete, often relying on costly, large SCM platforms for job security and payments. Decentralized, blockchain-based SCMs address these issues by enabling secure collaboration, empowering small retailers with enhanced privacy, transparency, and traceability of products, while reducing costs through a pay-per-service model.

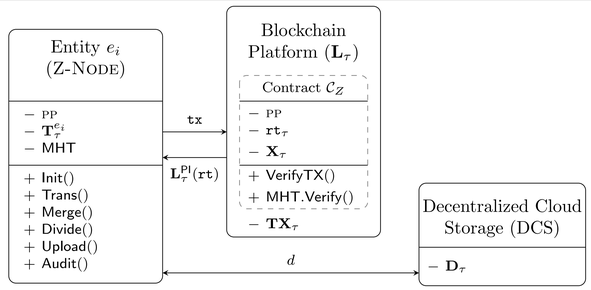
Our Primary Contributions

1. **Zupply Framework with Enhanced Security**: We designed an *anonymous authentication token* scheme using *zero-knowledge* proofs on *public blockchains*.
2. **Off-chain Authenticated Storage**: Zupply separates data storage from the blockchain while maintaining a concealed link to authentication tokens.
3. **Optimized Implementation**: We optimized proof size and introduced protocol for efficient Merkle tree updates. It is validated through implementations in C++ and Solidity.

Symbol	Description
τ	Time
$e_i, \mathbf{E}_\tau$	The i-th entity, the set of all es at τ
$T_i, \mathbf{T}_\tau$	The i-th authn. token, the set of all Ts at τ
$\mathtt{cm}_i, \mathbf{C}_\tau$	Commitment to the i-th authn. token, the set of all cms at τ
$\mathtt{eol}_i, \mathbf{X}_\tau$	End-of-life of the i-th authn. token, the set of all eols at τ
$\mathtt{tx}_i, \mathbf{TX}_\tau$	The i-th transaction, the set of all txs at τ
MHT, $\mathtt{rt}_\tau$	Merkle hash tree (MHT), the root of MHT at τ
$d_n, \mathbf{D}_\tau$	The n-th data record, the set of all ds at τ
$\mathrm{CID}_n, \mathbf{CID}_\tau$	CID of the n-th data record, the set of all CIDs at τ
$\mathtt{Tag}_i$	Data ownership transfer tag associated with T_i
$\mathbf{L}_\tau$	The shared ledger (i.e., blockchain platform) at τ
$\mathcal{H}$	Collision-resistant hash function
$\mathcal{E}_{\mathsf{sym}}$, $\mathcal{S}_{\mathsf{sig}}$	Symmetric encryption algorithm, digital signing algorithm

Zupply Framework

The Zupply framework Π Consists of a tuple of polynomial-time algorithms and protocols:
Π = (Setup, Init, Trans, Merge, Divide, Upload, VerifyTX, Audit; OT-Protocol, MHT-Protocol).



1. **Entities**: Data uploaders or auditors in the supply chain, form the entity set in Zupply and run the Zupply Node (Z-NODE)
2. **Blockchain Platform (BP)**: The Zupply smart contract ($\mathcal{C}_Z$) operates on a smart contract-enabled, permissionless blockchain to create and transfer of anonymous authentication tokens.
3. **Decentralized Cloud Storage (DCS)**: InterPlanetary File System (IPFS). Data is addressed by content identifiers (CID).

Adversary Model

1. $\mathcal{A}$, at one point, may have possessed an anonymous authentication token transferred from an authenticated entity. Currently, all such tokens are considered expired.
2. $\mathcal{A}$ has access to $\mathbf{TX}_\tau$, $\mathbf{E}_\tau$, $\mathbf{C}_\tau$, $\mathbf{X}_\tau$, MHT and $\mathbf{D}_\tau$.
3. *Authentication token commitment attribution oracle*: $\mathcal{A}$ can query $\mathcal{O}^{\mathsf{Attr}}: \mathbf{C}_\tau \mapsto \mathbf{E}_\tau$, such that, $\mathcal{O}^{\mathsf{Attr}}(\mathtt{cm}_i) = e_j$, where it assigns any $\mathtt{cm}_i$ to the entity e_j that created it. However, the $\mathtt{cm}_i$ can be owned by a different entity.
4. *Authentication token commitment classification oracle*: $\mathcal{A}$ can query $\mathcal{O}^{\mathsf{Class}}: \mathbf{C}_\tau \mapsto \{\mathsf{Init}, \mathsf{Trans}, \mathsf{Merge}, \mathsf{Div}\}$.
5. *Framework algorithms*: $\mathcal{A}$ is capable of executing all the algorithms in Π, or their modified version, except Setup.

Token Ownership Transfer

$\mathsf{Trans}\ (\mathrm{PP}, T_{i_1}, \mathsf{PKsig}_{i_2}) \to (\mathtt{tx}_{\mathsf{Trans}}, \tilde{T}_{i_2}, \mathtt{Tag})$

Protocol: Token Ownership Transfer

Entity **A**		Entity **B**
$T_1^{\mathrm{A}} = ((q_1^{\mathrm{A}}, \mathrm{PKsig}_1^{\mathrm{A}}, \rho_1^{\mathrm{A}}), \mathrm{SKsig}_1^{\mathrm{A}})$	"Req PKsig_2" →	
	← $\mathrm{PKsig}_2^{\mathrm{B}}$	$(\mathrm{PKsig}_2^{\mathrm{B}}, \mathrm{SKsig}_2^{\mathrm{B}}) \leftarrow B^{\mathcal{K}_{\mathsf{sig}}}(\mathrm{PP}_{\mathsf{sig}})$
$A^{\mathsf{Trans}}(\mathrm{PP}, T_1^{\mathrm{A}}, \mathrm{PKsig}_2^{\mathrm{B}})$ ↓ $(\mathtt{tx}_{\mathsf{Trans}_1}^{A}, \tilde{T}_2^{\mathrm{B}}, \mathtt{Tag}_1)$ Sends $\mathtt{tx}_{\mathsf{Trans}_1}^{A}$ to $\mathcal{C}_Z$	$\tilde{T}_2^{\mathrm{B}}; \mathtt{Tag}_1$ →	$T_2^{\mathrm{B}} \leftarrow (\tilde{T}_2^{\mathrm{B}}, \mathrm{SKsig}_2^{\mathrm{B}})$ $(\mathrm{PKsig}_3^{\mathrm{B}}, \mathrm{SKsig}_3^{\mathrm{B}}) \leftarrow B^{\mathcal{K}_{\mathsf{sig}}}(\mathrm{PP}_{\mathsf{sig}})$ $B^{\mathsf{Trans}}(\mathrm{PP}, T_2^{\mathrm{B}}, \mathrm{PKsig}_3^{\mathrm{B}})$ ↓ $(\mathtt{tx}_{\mathsf{Trans}_2}^{B}, \tilde{T}_3^{\mathrm{B}}, \mathtt{Tag}_2)$ Sends $\mathtt{tx}_{\mathsf{Trans}_2}^{B}$ to $\mathcal{C}_Z$

Entity A cannot discern if, when and to whom T_3^B is transferred. Also, entities not involved cannot recognize a link between the corresponding authentication token commitments, i.e., $\mathtt{cm}_1^A$, $\mathtt{cm}_2^B$, $\mathtt{cm}_3^B$.

Merkle Tree Root Update

Storing $2^L - 1$ nodes of An L-layer MHT on the blockchain is expensive. In $\mathcal{C}_Z$, it is enough to maintain $\mathtt{rt}_\tau$. Therefore, MHT populating follows a specific mechanism to let $\mathcal{C}_Z$ verify the consistency between the current $\mathtt{rt}_\tau$, the new cm, and the new $\mathtt{rt}^{\mathsf{new}}$. Such that, MHT leaves are initialized with their ind $\in [2^{L-1}]$.

$\mathsf{MHT.Verify}(\mathtt{rt}_\tau, \mathtt{rt}^{\mathsf{new}}, \mathtt{cm}, \mathsf{ind}, \mathsf{path}_{\mathsf{ind}}) \to b \in \{0, 1\}$

1: **if not** VerifyPath($\mathtt{rt}_\tau$, , $\mathsf{path}_{\mathsf{ind}}$, ind) **then**
2: **return** 0
3: **if not** VerifyPath($\mathtt{rt}^{\mathsf{new}}$, $\mathsf{path}_{\mathsf{ind}}$, cm) **then**
4: **return** 0
5: **return** 1

Security Properties

1. $\mathcal{A}$ cannot link a d_n, which $\mathcal{A}$ has not created, to its corresponding $\mathtt{cm}_i$ (***Data Anonymity***),
2. $\mathcal{A}$ cannot decide whether an entity e_i is the owner of a $\mathtt{cm}_j$ which is created by any $e_k \in \mathbf{E}_\tau$ (***Token Anonymity***),
3. $\mathcal{A}$ cannot decide whether $\mathtt{cm}_i$ has been transferred to $\mathtt{cm}_j$ for any i and j where $i < j$ (***Token Unlinkability***),
4. Let $\mathcal{A}$ does not own T_i, $\mathcal{A}$ cannot (1) transfer, merge or divide T_i, (2) use T_i to authenticate d^*, (3) alter an already existing $d_n \in \mathbf{D}_\tau$ (***Authenticity***),
5. $\mathcal{A}$ cannot transfer, merge, or divide tokens that are already transferred, merged or divided (***Token Undeniability***).

Implementation

Comparison of zkSNARK NP statements for $L = 20$. $\nu = n(x_{\mathtt{x}})$ is the number of public inputs, N is the number of constraints, $|\mathsf{vk}_{\mathtt{x}}|$ is the size of the verification key, and $|\mathsf{pk}_{\mathtt{x}}|$ is the size of the proving key.

x	ν	$\lvert\mathsf{vk}_{\mathtt{x}}\rvert$ (B)	N	$\lvert\mathsf{pk}_{\mathtt{x}}\rvert$ (MB)
Complexity	-	$O(\nu)$	$O(L)$	$O(N)$
Auth	2	640	588,248	182.5
Trans	6	896	642,876	200
Merge	8	1024	1,258,337	393.8
Div	8	1024	670,091	210

Transaction costs for $\mathcal{C}_Z$ and the baseline model in Gas and USD, based on ETH's value of USD 2,030.01 and gas price of 32 Gwei at the time of writing.

Transaction	Zupply		Baseline model	
	Gas	Cost	Gas	Cost
Deployment	3,088,611	$200.63	902,355	$58.62
$\mathtt{tx}_{\mathsf{Init}}$	133,415	$8.67	96,166	$6.25
$\mathtt{tx}_{\mathsf{Trans}}$	448,013	$29.10	29,649	$1.93
$\mathtt{tx}_{\mathsf{Merge}}$	455,534	$29.59	92,382	$6.00
$\mathtt{tx}_{\mathsf{Div}}$	518,701	$33.69	168,740	$10.96

Related Works

Framework	Security Properties						Costs				
							Storage				
	Anonymity	AATOT*	Unlinkability	Integrity	Decentralization	No Central Server	On-chain	Entity Software	Transfer (on-chain)	Audit (off-chain)	Scalability
Mesh [2]	◖	N/A	○	●	◖	○	●	○	N/A	N/A	●
DECOUPLES [3]	●	N/A	●	●	◖	●	●	○	○	N/A	○
zkLedger [4]	◖	N/A	●	●	●	●	●	○	◖	●	●
Zupply	●	●	●	●	●	●	○	◖	◖	○	○
Baseline model	○	N/A	○	●	●	●	○	○	○	○	○

*Anonymous Authentication Token Ownership Transfer

Conclusions

Zupply offers a trustless, decentralized solution for managing off-chain DAG data in supply chains, using zero-knowledge proofs on Ethereum. Our C++ and Solidity implementation remains efficient despite high gas costs. Future work will address the need for trusted setup and lack of post-quantum security in the zkSNARK protocol.

References

[1] Mohammadtaghi Badakhshan and Guang Gong. "Zupply: Anonymously Maintained Decentralized DAG Data Record Over Public Blockchains". In: TechRxiv (June 2024)
[2] Riham AlTawy and Guang Gong. "Mesh: A Supply Chain Solution with Locally Private Blockchain Transactions". In: Proceedings on Privacy Enhancing Technologies 3 (2019), pp. 149–169.
[3] Mourad El Maouchi, Oğuzhan Ersoy, and Zekeriya Erkin. "DECOUPLES: A Decentralized, Unlinkable and Privacy-Preserving Traceability System for the Supply Chain". In: Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing. SAC '19.
[4] Neha Narula, Willy Vasquez, and Madars Virza. "zkLedger: Privacy-Preserving Auditing for Distributed Ledgers". In: 15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18).