Holonym: Private Proofs on Identity for Blockchains and Beyond

Nanak Nihal Khalsa
nanak@holonym.id

Shady El Damaty
shady@holonym.id

November 21, 2023

High-Level Reason For Building V2

Holonym V1 addressed the problem of identity on public blockchains, enabling private Sybil resistance and Know-Your-Customer (KYC). Holonym V2 improves it and extends the use cases beyond identity verification to be closer to the "dream" of applied ZK privacy: private proofs about practically any existing data. Despite Holonym V1's relatively niche success, its privacy gaurantees and ease of use were found to be lower than initially imagined. Privacy and user experience UX in Holonym V1 were a tradeoff, and a surprisingly high fraction of users chose the latter. Holonym v2 removes this tradeoff, arguably surpassing "web2" ease with far stronger notions of privacy than even possible during in Holonym V1. Holonym V1 was also limited to ID verification use cases with *new* issuers redoing KYC and using SNARK-friendly cryptographic primitives. Meanwhile, Holonym V2 is able to efficiently support common *already-existing* credentials using non-SNARK-friendly cryptography such as RSA signatures and SHA256. Highly practical examples only enabled by V2 architecture include Aadhaar, Apple Wallet, biometric passports, emails, signed PDFs, SSL certificates, TLS sessions attested by a proxy, and signed blockchain Merkle Patricia Trie (MPT) roots. This entails that facts about numerous existing credentials, web data, or blockchain data can be proven without revealing the prover's identity. Furthermore, in V2, privacy is no longer expensive, issuers incur no economic risk, and economic incentives aid in distributing the zkEscrow regulatory compliance network.

Technical Contributions

VOLE-Based ZK, with noninteractivity and (optimistic) succinctness In blockchains, zero-knowledge succinct nontinteractive arguments of knowledge (zkSNARKs) have reached high popularity because of their

succinctness and noninteractivity enable the posting of compressed valid L2 state transitions to L1s. However, for identity-related use cases that neither require compression nor public verifiability, succinctness and non-interactivity can be sacrificed for speed. This enables the exploration of other ZK arguments and ZKPs that are not zkSNARKs. Particularly, we choose VOLE-based ZK, and a publicly verifiable variant called VOLE in the Head (VitH)[1]. For blockchain use cases, we optimistically "compress" its results into signed blockchain transactions, retaining blockchain-friendly properties while giving well-over 100x speedup to existing SNARKs. This enables proofs over much desired but beyond-reach legacy credentials that are typically too slow and memory-intensive to enable private computations on consumer hardware.

Optimal Linear Codes for VitH While building the first general-purpose VitH library to our knowledge, we encountered an asymptotic bottleneck in practice: while the VitH paper [1] suggests a 2x communication overhead relative to standard VOLE-based ZK, the computational overhead is far higher due to conversion to a subspace VOLE which requires a large matrix multiplication of $O(n^{2.8073})$. Silent VOLE requires matrix multiplication but is employs linear codes with either sparse generators or sparse parity check matrices. These codes are not optimal for subspace VOLE because subspace VOLE requires that the verifier multiply by a matrix $\mathbf{T}_c$ containing generator and the prover multiply by its inverse $\mathbf{T}_c^{-1}$. Sparse matrices do not typically have sparse inverses, so we present two methods for constructing efficient linear codes, the latter of which results in extremely fast $O(n)$ matrix multiplication.

Removal of need for new anonymity sets Time-based anonymity sets hamper user experience and privacy. Users do not typically wait between verifying their ID and proving, leaving the privacy provided by Holonym V1 less than intended. Holonym V2 leverages existing credentials, much more convenient and anonymous than redoing KYC, while removing the need for end users to wait for an anonymity set to increase.

Composability with Existing Ecosystems ZKPs are useless if they not, well, used...so our library compatible with R1CS domain-specific languages (DSLs) such as Keelung, Circom, and ZoKrates, and supporting the bn254 prime. Support for other prime fields, especially the secp256k1 field and binary fields is on short-medium term roadmap in order to support Ethereum state proofs.

Timelock Puzzles for Additional Privacy Holonym V1 introduced zkEscrow, a threshold decryption network for compliance which has not yet

been released. In V2, we exploit a simplistic version of timelock puzzles [2] akin to proof-of-work [3] to add an additional economic layer of security to disincentivize even a corrupt majority of the nodes of the network from decrypting user data.

Technical Details

1 ZK from VOLE

Vector Oblivious Linear Evaluation involves many iterations of Oblivious Linear Evaluation (OLE) with the same global "MAC key," Δ. In each OLE, the verifier receives the same global key Δ and a value q. These represent a linear function defined by Δ and q. The prover receives a point (u, v) on the line defined by the verifier's function.

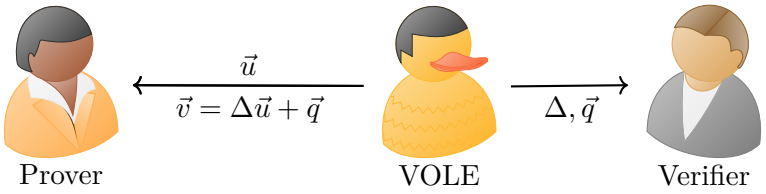

Figure 1: VOLE functionality, as a Duck

This forms a two-party commitment to a vector of $\vec{u}$ of u values, where the prover can open an entry u_i by sending (u_i, v_i). The verifier can check this point lies on their line. The commitment is information-theoretically binding since another (u_i, v_i) point on the verifier's line cannot be forged without knowing the line. It is also information-theoretically hiding because the verifier's view of a commited value reveals nothing about the prover's commited value.

Importantly, these commitments are linearly homomorphic so the outputs of linear gates (addition and constant multiplication) can be computed on the commited values directly by the verifier. Multiplication gates are more difficult, though a number of efficient solutions such as Mac'N'Cheese [4], Quicksilver [6], Line-point ZK [7], and and Wolverine [5] have been proposed. We utilize Quicksilver.

VOLE-based ZK results in fast prover times, proving millions of multiplications per second on relatively standard hardware [6]. However, the proof size grows linearly with the number of multiplication gates and size of the witness.

2 VOLE in the Head

VOLE commitments can only be verifiably opened to a single private-coin verifier, who must keep its randomness secret. Public verifiability is important for blockchain ecosystems. Hence, we use VOLE in the head, a recent work that enables conversion to a public-coin and thus amenable to a Fiat-Shamir transform. VOLE in the head works via a modified SoftSpokenVOLE [**?**] which produces the a VOLE correlation where the verifier's choices do not affect the prover's VOLE output - thus the prover can compute it's view of the commitment before the verifier's challenge.

In this, there is a small set of seeds $S = [s_0...s_n]$, and a function $PRG : S \mapsto \mathbb{F}_p^\ell$ to expand these seeds to a vector $\mathbb{F}_p$ elements of length ℓ. The Verifier can later choose an index Δ, where he receives all s except s_Δ. The parties can compute a VOLE without each other (note the verifier can compute q without known s_Δ because this term becomes 0)

- $\vec{q} = \sum_{x \in S_\Delta} PRG(s_x)(\Delta - x)$
- $\vec{u} = \sum_{x \in S_\Delta} PRG(s_x)$
- $\vec{v} = -\sum_{x \in S_\Delta} PRG(s_x) * x$

Since S is small, Δ has few choices, so this is insecure by itself. To combat this limitation, one can do many VOLEs and stack them as columns into matrices that satisfy a VOLE correlation: $\mathbf{Q} = \mathbf{U}\mathbf{diag}(\vec{\Delta}) + \mathbf{V}$ ($diag(\vec{\Delta})$ refers to a matrix with all zeroes except the values of Δ along its diagonal). Then, one can convert this to a subspace VOLE, where for some linear code's generator $textbfG_c$, the VOLE is of $\mathbf{UG}_c$, instead of $\mathbf{U}$:

$$\mathbf{Q} = \mathbf{U}\mathbf{G}_c\mathbf{diag}(\vec{\Delta}) + \mathbf{V}$$

Note that each row of $\mathbf{UG}_c$ is a codeword; i.e. it is in the subspace defined by $\mathbf{G}_c$. This means that if we can get the prover to open a commitment to a whole row instead of just a single value, the prover cannot easily cheat – even if she can forge an invalid opening of an item in the row by guessing Δ, she will be caught as long as she cannot make a valid codeword. For a code with minimum distance d, she would need to guess at least d values of Δ to open to a fake row that is also a codeword. This results in roughly $|S|^d$ bits of security.

To convert subspace VOLE to VitH,

1. Treat the $\mathbf{U}$ matrix as $\begin{bmatrix}\mathbf{U1}\\ \mathbf{R}\end{bmatrix}$
2. Treat the $\mathbf{V}$ matrix as the concatenation of two matrices, $\begin{bmatrix}\mathbf{V1}\\ \mathbf{V2}\end{bmatrix}$

3. Now, verifier sends a $\Delta' \in \mathbb{F}_p$
4. Prover send $\mathbf{S}$ claimed to equal $\mathbf{R} + \mathbf{U1} * \Delta'$

A regular VOLE but the prover can't cheat! $\mathbf{S}$ in the code's subspace $\iff$ the Prover is honest

3 Optimal linear codes for subspace VOLE

To create subspace VOLE, the parties must agree on a large square matrix, $\mathbf{T}_c$, that contains $\mathbf{G}_c$ as its first rows and is invertible. The prover multiplies some values by $\mathbf{T}_c$ to create subspace VOLE and the verifier at some point must checks the prover's honesty by multiplying some values with $\mathbf{G}_c$. The exact values are irrelevant to this discussion - the important part is that matrix multiplication is asymptotically bad. Thus, we looked for ways to make linear codes that

- after appending linearly independent rows to make it square, have sparse inverses (fast prover)
- are sparse (fast verifier)
- have high minimum distance (security exponential in minimum distance)

One such general-purpose method if only the prover needs to be efficient is:

1. Start with the transpose of a k x n generator of distance-d code. Call the transpose's generator $\mathbf{G}'$
2. Append with $n-k$ linear independent columns to $\mathbf{G}'$ so it is invertible
3. Multiply $\mathbf{G}'^{-1}$ by elementary row operation matrices $[\mathbf{E}_1, \mathbf{E}_2, ...\mathbf{E}_n]$ that act only on the first n rows until reaching a satisfactorily sparse matrix $\mathbf{T}$
4. Let $\mathbf{E} = \prod_{i=1}^{r} \mathbf{E}_i$. It can be shown$\mathbf{T}^{-1} = \mathbf{E}^{-1}\mathbf{G}'$.
5. Let $\mathbf{G}_e = \mathbf{T}^{-1}$ transpose
6. Let $\mathbf{G}_c =$ first k rows of $\mathbf{G}$

This produces a sparse $\mathbf{T}$ whose transpose is the inverse of a generator $\mathbf{G}_c$ for a distance-d code, with extra rows to that let it be invertible.

If we desire an efficient prover and verifier, but are okay with the code having relatively low rate (this is likely good trade for the most use cases

as), we can use a Repeat-Multiple-Accumulate (RMA) code [8]. These have sufficiently high minimum distance and long enough block sizes [8]. One can see they not only are sparse but also can see informally that they have a sparse inverse – it is possible to construct an efficient algorithm to invert the operations performed by these matrices.

4 Circuits for legacy credentials

Existing credentials exist in primarily three places:

1. **Signed certificates** such as the growing trends of mobile drivers licenses and ISO-14443-compliant government IDs, along with the well-adopted Aadhaar within India. This may also include X509 certificates, .pkpass files, and other forms of cryptographically signed certificates of data. These have substantial utility but the RSA and SHA256 involved has been impossible for standard proving systems on average user hardware. Developing tools for practical proofs on such data is the primary key Holonym V2.

2. **In emails** Often, certificates and receipts of important actions such as login attempts, legal document signatures, financial transfers, course completions, or whisteblower evidence, are sent via email and thus signed with DKIM. Yet these require RSA, SHA256, and complex regex to make use of in a ZKP and thus the initial demonstration of such use cases[9] [10] still struggle with slow prover time while providing the benefits of signed emails.

3. **The state tries of blockchains** store commitments to all blockchain user data. A user can prove anything about the blockchain state, e.g., asset balances, SBT ownership, or anything else stored on the blockchain by proving this, but Keccak256 tries have been orders of magnitude too computationally expensive within standard proving systems' client-side ZK proofs. Proving these rapidly on consumer devices is a short-medium term goal of Holonym V2.

Conclusion

Holonym V2 enables new use cases due to its adoption of modern ZK proof systems and addition of faster linear codes. These use cases include legacy credentials which are inherently more convenient and far more private than redoing KYC. They enable private proofs of emails that can be computed quickly client-side, and they make private proofs of Ethereum state possible.

References

[1] Carsten Baum, Lennart Braun, Cyprien Delpech de Saint Guilhem, Michael Klooß, Emmanuela Orsini, Lawrence Roy, and Peter Scholl. *Publicly verifiable zero-knowledge and post-quantum signatures from VOLE-in-the-Head.* In CRYPTO. Springer, 2023.

[2] Ronald Rivest, Adi Shamir, and David Wagner. *Timelock puzzles and timed-release crypto.* Technical report, Cambridge, MA, USA (1996)

[3] Nakamoto, Satoshi. "Bitcoin: A peer-to-peer electronic cash system." Available at SSRN 3440802 (2008).

[4] Baum, C., Malozemoff, A. J., Rosen, M. B., & Scholl, P. *Mac'n'Cheese Mac n Cheese: Zero-Knowledge Proofs for Boolean and Arithmetic Circuits with Nested Disjunctions.*Advances in Cryptology–CRYPTO 2021: 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16–20, 2021, Proceedings, Part IV 41. Springer International Publishing, 2021.

[5] Weng, Chenkai, Kang Yang, Jonathan Katz, and Xiao Wang. *Wolverine: fast, scalable, and communication-efficient zero-knowledge proofs for boolean and arithmetic circuits.* In 2021 IEEE Symposium on Security and Privacy (SP) (pp. 1074-1091). IEEE.

[6] Yang, K., Sarkar, P., Weng, C., & Wang, X. *Quicksilver: Efficient and affordable zero-knowledge proofs for circuits and polynomials over any field.* Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security (pp. 2986-3001).

[7] Dittmer, Samuel, Yuval Ishai, and Rafail Ostrovsky. *Line-point zero knowledge and its applications.* Cryptology ePrint Archive (2020).

[8] Kliewer, Jörg, Kamil S. Zigangirov, and Daniel J. Costello Jr. *New results on the minimum distance of repeat multiple accumulate codes.* Proc. 45th Annual Allerton Conf. Commun., Control, and Computing. 2007.

[9] Divide-By-0, saleel, & curryrasul. Proof of email Retrieved from https://github.com/zkemail (2022).

[10] asoong, richardliang, bweick, 0xSachinK & justinkchen . Proof of email Retrieved from https://github.com/zkp2p/zk-p2p (2023).