Enabling Untraceable Anonymous Payments in the Lelantus Protocol

Aram Jivanyan[1] and Sarang Noether[2]

[1] Zcoin
aram@zcoin.io
https://zcoin.io
[2] Monero Research Lab
sarang.noether@protonmail.com
https://getmonero.org

Abstract. We provide an update to the Lelantus protocol to enable untraceable direct anonymous payments.

Keywords: Lelantus, One-out-of-Many Proofs, Double-blinded commitments

1 Introduction

One of the major issues of the Lelantus protocol [3] is one of traceability in direct anonymous transfers. In such scenarios, the sender can observe transactions and detect when a coin (represented as a commitment) is later spent by the receiver. As this presents a notable security problem, one workaround requires the recipient to spend a newly-received coin and mint a new coin to itself to remove this tracing. Although this approach mitigates the traceability issue, it requires an additional transaction, increasing the space required on the blockchain, overall verification time, and delay before the coin can be used. This approach may also introduce timing data that an observer could use to form heuristics about spending behavior and reduce practical anonymity.

We describe a protocol upgrade intended to mitigate this issue in a more efficient and secure way.

2 Background

In order to support this protocol upgrade we have to modify how Lelantus coin serial numbers are generated. Next we heavily utilize cryptographic proofs of representation as described below.

2.1 Coin Serial Numbers and Spending Keys

In Lelantus, coins are represented by double-blinded commitments of the form $C = g^S h_1^V h_2^R$ in a prime-order group G. Here S is the coin's *serial number* generated by hashing a unique public key $Q = g^q$, V is the value of the coin, and R is a Pedersen blinder. During a spend transaction, the coin's public key Q is revealed to compute the serial number S, which is published as part of the transaction; further, the spender signs the entire transaction with q to prove ownership of the coin. The public key Q enables all network participants to both compute the serial number (which is required for verification of the spend transaction) and verify the transaction signature. This prevents previously-known burning attacks where a malicious interloper who controls network messages can intercept an honest spend, mint a coin with the same serial number, and spend it [5]; in our construction, the attacker would be unable to sign with q, and the burn attempt would be rejected by the network. Note that in the original Zerocoin protocol, this situation does not arise since the protocol does not support direct anonymous transfers [4].

A tracing problem arises in the case of direct payments, where a Diffie-Hellman exchange is used to generate Q. This exchange ensures that the recipient, and not the sender, can recover the private key q corresponding to the coin. However, because the sender knows the public key Q, it can observe transactions until it sees this public key later revealed in another transaction. At this point, the sender knows the coin has been spent. Note that in the original Zerocoin protocol, this situation does not arise since the protocol does not support direct anonymous transfers [4].

We propose to modify this scheme and redefine the serial number. Using the same commitment format, we consider g^S to be the serial number, where the spender must now sign the transaction with S directly. During a spend transaction, g^S is publicly revealed and used to verify the transaction signature.

It is important to mention that enabling efficient batch verification using this construction is still possible.

2.2 Proof of Knowledge of a Representation

Consider a group G of prime order p and n generators $g_1, g_2, \ldots, g_n$. A representation of the element z against the generators $g_1, g_2, \ldots, g_n$ is the vector $(s_1, s_2, \ldots, s_n)$ such that $z = g_1^{s_1} g_2^{s_2} \cdots g_n^{s_n}$. Below we describe a protocol for proof of representation for arbitrary values n, referred to as a generalized Schnorr proof of knowledge.

Formally, such a proof of representation is a zero-knowledge argument for the following relation:

$$R = \{g_1, g_2, \ldots, g_n \in G, y \quad ; \quad s_1, s_2, \ldots, s_n \in Z_p \quad | \quad y = g_1^{s_1} g_2^{s_2} \cdots g_n^{s_n}\}$$

The protocol is depicted in Figure 1. Verifying the completeness of the protocol is straightforward. It can be converted into a non-interactive protocol that is secure and special honest-verifier zero-knowledge in the random oracle model using the Fiat-Shamir heuristic [1].

$$\begin{array}{lcl}
\text{Prover}(g_1, g_2, \ldots, g_n, y, (s_1, s_2, \ldots, s_n)) & & \text{Verifier}(g_1, g_2, \ldots, g_n, y) \\
\text{Computes} & & \\
\quad x_1, x_2, \ldots, x_n, \leftarrow_R \mathbb{Z}_p & \xrightarrow{u} & \\
\quad u = g_1^{x_1} g_2^{x_2} \ldots g_n^{x_n} \in G & & \\
 & \xleftarrow{x \leftarrow \{0,1\}^{\lambda}} & \\
\quad t_1 = x_1 - x \cdot s_1 \in \mathbb{Z}_p & & \\
\quad t_2 = x_2 - x \cdot s_2 \in \mathbb{Z}_p & & \\
\quad \quad \vdots & & \\
\quad t_n = x_n - x \cdot s_n & \xrightarrow{s_1, s_2, \ldots, s_n} & \text{Accepts if and only if} \\
 & & u = y^x g_1^{t_1} g_2^{t_2} \cdots g_n^{t_n}
\end{array}$$

Fig. 1. Generalized Schnorr proof of knowledge

3 Direct Anonymous Payments

In order to ensure proper authentication for a spend of the coin $C = g^S h_1^V h_2^R$, the coin owner should exclusively possess the secret values S, V, and R. A spend transaction reveals the serial number g^S and provides a valid signature using S. Our goal is to develop a protocol that ensures that only the recipient can recover both S and g^S, which ensures that the sender cannot watch for a known serial number to trace later transactions.

In Figure 2 below we describe such a protocol. Note that we assume a recipient generates a new address for each such transaction, and that the sender is assumed to transmit the secret values y and V to the recipient. This can be done either through a secure side channel, or by encrypting them with a recipient public key and including them in the transaction.

Notice also the requirement for the recipient to include a Schnorr proof of representation of the point Q with respect to generators g and h_2; without this, it is not possible to verify that the resulting coin commitment is constructed using fixed public generators with no discrete logarithm relationship.

Sender		**Recipient**
		Samples $x, r \leftarrow_R \mathbb{Z}_p$
		Computes new public address $Q = g^x h_2^r$
		Generates a Schnorr proof P on the point Q
	$\xleftarrow{Q, P}$	
Verifies P		
Samples $y \leftarrow_R \mathbb{Z}_p$		
Creates the output coin $C = Q^y h_1^V (= g^{xy} h_1^V h_2^{ry})$		
Generates a range proof with respect to the generators Q and h_1		
C, Q, P are published to the blockchain	$\xrightarrow{y, V}$	
		Reveals the coin serial number g^{xy}
		Generates the Spend proof for the coin
		Signs the transaction with the private key xy

Fig. 2. Minting new coin for dedicated recipient

4 Generating a Balance Proof

This construction requires modification of the transaction balance proof generation process described in [3]. Assume a transaction spends N_{old} input coins and outputs N_{new} coins $C_{O,1}, \ldots, C_{O,N_{new}}$. Each output coin $C_{O,i}$ is associated with the address Q_i, which is published on the blockchain with the transaction. The balance proof generation steps are the following:

1. The sender takes all output coins, the net output value V_{OUT}, the transaction fee f and the common challenge value x used for Σ-proofs constructions, and computes the following element:

$$\begin{aligned}\mathbf{A} :&= (C_{O,1} \cdot \ldots \cdot C_{O,N_{new}})^{x^m} \cdot h_1^{(V_{OUT}+f)x^m} = \\ &= Q_1^{y_1 x^m} \ldots Q_{N_{new}}^{y_{N_{new}} x^m} h_1^{(V_{OUT}+V_{O,1}+\ldots+V_{O,N_{new}}+f)x^m}\end{aligned}$$

2. Takes the elements $z_{V1}, \ldots, z_{V_{N_{old}}}, z_{R1}, \ldots, z_{R_{N_{old}}}$ and $\{Comm(0, \rho_k^t, \tau_k^t + \gamma_k^t)\}_{k=0}^{m-1}$ from the corresponding Σ-proof transcripts (for more details we refer to the original paper [3]) and computes the element:

$$\begin{aligned}\mathbf{B} :&= Comm(0; z_{V1} + \ldots + z_{VN_{old}}, z_{R1} + \ldots + z_{RN_{old}}) \cdot \prod_{t=1}^{N_{old}} \Big(\prod_{k=0}^{m-1} (h_2^{\gamma_k^t} \cdot Comm(0; \rho_k^t, \tau_k^t))^{x^k} \Big) \\ &= h_1^{(V_{I1}+\cdots V_{Iold})x^m} h_2^{\sum_{t=1}^{old}(R_{It} \cdot x^m + \sum_{k=0}^{m-1} \gamma_k^t \cdot x^k)}\end{aligned}$$

If the transaction balance holds, the h_1 exponents in A and B will cancel:

$$\frac{\mathbf{A}}{\mathbf{B}} = Q_1^{y_1} Q_2^{y_2} \ldots Q_{N_{new}}^{y_{N_{new}}} h_2^Y \qquad (1)$$

where

$$Y = -\sum_{t=1}^{old}\Big(R_{It} \cdot x^m + \sum_{k=0}^{m-1} \gamma_k^t \cdot x^k\Big)$$

Now observe that to provide a balance proof, it is sufficient for the transaction owner to prove knowledge of the exponent values $y_1, y_2, \ldots, y_{new}$ and Y in Equation 1 with respect to the generators $Q_1, Q_2, \ldots, Q_{new}$ and h_2. Given that these generators are public, the sender can provide a proof of representation of the value $\frac{\mathbf{A}}{\mathbf{B}}$ with respect to the generators $Q_1, \ldots, Q_{new}, h_2$, which then can be publicly verified by all network participants.

5 Performance

This protocol change produces tradeoffs in storage and computational efficiency compared to [3].

1. This construction requires each new minted coin to be explicitly associated with the recipient address Q along with its proof of representation with respect to the generators g and h_1. Hence each minted coin will be composed of the commitment C, the address Q and corresponding proof of representation P. This requires an additional group element and two additional scalars per mint.
2. The payment system implies that the coin private data is communicated with the recipient either through a secure side channel or directly on the blockchain. In the latter case, the sender encrypts all private coin data with the recipient's public key and includes this in the transaction. In our new construction, this private data is the elements y and V; in [3] it is y, V, and R. The reduction in space depends on the method used to encrypt these values to the recipient.
3. Balance proofs (which are composed of a single proof of representation) will require an additional $N_{new} - 1$ group elements. For a transaction with a single output, there is no additional overhead.
4. Range proofs in this construction are produced using standard Pedersen commitments, instead of the double-blinded commitments in the original protocol. When using Bulletproofs [2], this is a reduction in storage of a single group element.

We note that batch verification of spend proofs is still possible with our updated construction.

6 Further Work

The construction presented here mitigates the tracing problem present in the original Lelantus transaction protocol. However, it requires a recipient to present fresh address data to senders for each transaction. It is desirable for a recipient to publish a single address such that senders can derive ephemeral one-time addresses as needed. However, securely designing such a construction that is resistant to tracing remains an open task.

Finally, this construction is informal and lacks a security analysis.

References

1. Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proceedings of the 1st ACM Conference on Computer and Communications Security*, CCS '93, pages 62–73, New York, NY, USA, 1993. ACM.
2. B. Bünz, J. Bootle, D. Boneh, A. Poelstra, P. Wuille, and G. Maxwell. Bulletproofs: Short proofs for confidential transactions and more. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 315–334, May 2018.
3. Aram Jivanyan. Lelantus: Towards confidentiality and anonymity of blockchain transactions from standard assumptions. Cryptology ePrint Archive, Report 2019/373, 2019. `https://eprint.iacr.org/2019/373`.

4. I. Miers, C. Garman, M. Green, and A. D. Rubin. Zerocoin: Anonymous distributed e-cash from Bitcoin. In *2013 IEEE Symposium on Security and Privacy*, pages 397–411, May 2013.
5. T. Ruffing, S. A. Thyagarajan, V. Ronge, and D. Schroder. Burning Zerocoins for fun and for profit - a cryptographic denial-of-spending attack on the Zerocoin protocol. In *2018 Crypto Valley Conference on Blockchain Technology (CVCBT)*, pages 116–119, June 2018.