Addressing Network-Level Threats in Zcash Using the Nym Network

Introduction

Purpose of this Document

Zcash is a privacy-enhanced decentralized cryptocurrency that prioritizes privacy in transactions (Sasson et al., 2014). Launched in 2016, Zcash was created to provide users with the option of conducting fully *shielded* transactions, where both the sender, recipient, and transaction amount are hidden from public view. This unique feature sets Zcash apart from other cryptocurrencies like Bitcoin, which offer pseudonymous transactions but lack the privacy features of Zcash's shielded transactions.

The privacy offered by Zcash is twofold: transaction confidentiality and user anonymity. In fully shielded transactions, all critical details such as the sender's address, the recipient's address, and the transaction amount are completely hidden from anyone observing the blockchain. This ensures that outsiders cannot trace the flow of funds or link transactions to specific individuals.

This level of privacy is particularly beneficial for users concerned about financial privacy, protecting their transactions from being exposed to prying eyes, whether they be individuals, organizations, or even government entities. By keeping transaction details confidential, users can protect themselves from potential threats such as identity theft, financial profiling, and targeted attacks.

At the core of Zcash's privacy model is its innovative use of zero-knowledge proofs, specifically zk-SNARKs (Zero-Knowledge Succinct Non-Interactive Arguments of Knowledge). These cryptographic proofs enable users to prove transaction validity without revealing sensitive information. In the context of Zcash, zk-SNARKs allow users to shield their transactions, ensuring confidentiality while still maintaining the integrity and security of the blockchain.

The use of zk-SNARKs in Zcash involves complex cryptographic operations that allow one party to prove to another that a statement is true without revealing any information beyond the validity of the statement itself. For Zcash transactions, this means that the network can verify that a transaction is legitimate and the sender has sufficient funds, all without exposing any details about the sender, recipient, or amount involved.

By offering these privacy features, Zcash empowers users with greater control over their financial information, enhancing their ability to conduct private and secure transactions in an increasingly transparent digital world.

However, as extensive research and analysis have shown [Kappos, 2018; Tramer, 2020; Biryukov, 2021], network-level metadata of data packets sent and received over the network during transactions can inadvertently leak crucial information even when the blockchain itself uses shielded transactions. While cryptographic techniques such as zero-knowledge proofs (zk-SNARKs) or encryption are designed to ensure transaction confidentiality on the blockchain, vulnerabilities at the network layer can compromise user privacy. By analyzing patterns in network traffic, adversaries can potentially infer sensitive details about transactions, such as the identities of the parties involved or the nature of the transactions. As we discuss further in this document, the leakage at the network layer undermines the robust privacy features offered by cryptographic protocols, highlighting the importance of addressing network-level vulnerabilities to safeguard user anonymity and confidentiality. In particular, we outline how the Nym network can effectively mitigate these vulnerabilities and enhance user privacy at the network level.

Overview of Zcash

Network Entities

Full Nodes

Full nodes are essential components of the Zcash network infrastructure, serving as the backbone of the decentralized blockchain. They are complete implementations of the Zcash protocol that store and maintain a full copy of the entire blockchain, including all transaction history from the genesis block to the most recent block. Full nodes play a critical role in ensuring the integrity, security, and consensus of the Zcash network. Full nodes play a crucial role in achieving and maintaining network consensus within the Zcash ecosystem. By independently validating transactions and blocks according to the established consensus rules, full nodes help ensure that all network participants agree on the state of the blockchain. In the event of a disagreement or attempted violation of the consensus rules, full nodes serve as guardians of network integrity, rejecting invalid transactions and blocks to preserve the security and trustworthiness of the network.
The key functionalities of full nodes can be summarised as follows:

1. **Transaction Validation:** Full nodes independently validate incoming transactions to ensure they adhere to consensus rules and protocol specifications. This includes verifying digital signatures, checking for double-spending attempts, and confirming that transactions meet the criteria for inclusion in a block.
2. **Block Validation:** Full nodes validate new blocks propagated by miners to ensure they contain valid transactions and adhere to the network's consensus rules. This validation process helps maintain the security and integrity of the blockchain by preventing the inclusion of invalid or fraudulent transactions.
3. **Propagation of Transactions and Blocks:** Full nodes participate in the propagation of valid transactions and blocks throughout the network, ensuring that all nodes have a consistent view of the blockchain. By relaying transactions and blocks to their peers, full nodes help maintain network synchronization and facilitate efficient communication between network participants.
4. **Storage of Blockchain Data:** Full nodes store a complete copy of the Zcash blockchain locally, including all transaction history and block headers. This data is essential for performing transaction and block validation, as well as providing historical information to network participants querying the node for specific data.

Mining Nodes

Miners are specialized nodes within the Zcash network responsible for creating new blocks and securing the blockchain through the process of mining. Miners play a vital role in the decentralized consensus mechanism of Proof of Work (PoW), which is used by Zcash to achieve agreement on the state of the blockchain. Miner nodes play a critical role in achieving consensus within the Zcash network by collectively determining the order and validity of transactions. Through the process of mining, miners compete to create new blocks and extend the blockchain, with the longest valid chain being considered the authoritative version of the blockchain. This decentralized consensus mechanism ensures that all network participants agree on the state of the blockchain and prevents double-spending and other fraudulent activities.

The key functionalities of mining nodes can be summarised as follows:

1. **Block Creation:** Miner nodes collect transactions from the network and bundle them into candidate blocks. These candidate blocks contain a set of transactions waiting to be confirmed and added to the blockchain. Miners compete to find a valid solution to a cryptographic puzzle, known as the proof of work, which allows them to create a new block and add it to the blockchain.
2. **Proof of Work Computation:** To create a new block, miner nodes must solve a computationally intensive mathematical puzzle that requires significant computational power and energy expenditure. This process, known as proof of work mining, involves repeatedly hashing the block header with different nonce values until a solution is found that meets the network's difficulty target.
3. **Transaction Inclusion:** Once a miner successfully solves the proof of work puzzle and creates a new block, they broadcast the block to the rest of the network for validation

and inclusion in the blockchain. The transactions included in the block are considered confirmed and become part of the immutable transaction history of the Zcash blockchain.

We note that although miner nodes and full nodes are not the same type of nodes in the Zcash network, some nodes may perform both functions simultaneously, acting as both full nodes and miner nodes.

Light Clients (Light Wallets)

Light clients (or light wallets) interact with the Zcash network through a simplified protocol that allows them to query remote servers, known as lightwalletd servers, for specific blockchain data. These servers maintain a copy of the blockchain and provide lightweight clients with the information they need on demand.

By offloading the burden of storing and processing blockchain data to remote servers, light clients can operate efficiently on devices with constrained resources, such as smartphones or tablets. Without the need to download and verify the entire blockchain, light clients can synchronize with the network more quickly, enabling users to access up-to-date transaction data in a timely manner.

The light client runs on the user's device, which can be a mobile phone, tablet, or personal computer. The light client handles the creation and signing of transactions locally. The private keys are stored on the user's device and never leave it, ensuring that the user retains full control over their private keys.

Throughout this document, we will use the terms "light clients" and "light wallets" interchangeably.

Lightwalletd

lightwalletd is a service within the Zcash ecosystem specifically designed to support light clients, particularly mobile-friendly light wallets such as Zashi, Zingo!, Ywallet, or Nighthawk. It allows the light wallets to interact with the Zcash network without needing to maintain a full copy of the blockchain, which streamlines the interaction between light wallets and the Zcash network and enhances efficiency for users with limited storage and computational resources. *lightwalletd* is designed to run on a server, providing an interface for multiple light wallets to interact with the Zcash network, without the need to maintain a full copy of the blockchain. It acts as a middle layer between the Zcash full nodes and the light clients. The interaction between light wallets and *lightwalletd* follows a client-server model. Key features of lightwalletd include:

1. **Block Processing:** lightwalletd regularly fetches new blocks from the Zcash network, processing them into a compact format suitable for light clients while retaining all necessary transaction information.
2. **Data Storage:** The service stores relevant blockchain data, including compact blocks and transaction details, ensuring quick and efficient retrieval when requested by light clients.

3. **Transaction Broadcasting:** While lightwalletd does not handle transaction signing or manage private keys, it plays a crucial role in broadcasting signed transactions submitted by light wallets to the Zcash network.

In summary, lightwalletd acts as a bridge between light wallets and the Zcash network, providing essential blockchain data and facilitating secure transaction broadcasting while optimizing resource utilization for users on mobile and other resource-constrained devices.

LightClient <> Lightwalletd Interactions

In this section, we outline a breakdown of how lightwalletd operates, explaining its interactions with light clients and full nodes in the context of sending, receiving, and looking up transactions.

1. Sending a Transaction

When a user wants to send a transaction using a light client connected to lightwalletd, the process involves several steps. First, the light client constructs the transaction locally by specifying the inputs (sources of funds), outputs (recipients), and the amount to be transferred. For shielded transactions, zk-SNARK proofs are generated to ensure the transaction is valid without revealing sensitive details. Next, the transaction is signed with the user's private key on the light client. Once signed, the transaction is sent to lightwalletd via gPRC, which broadcasts it to the Zcash network. Acting as a relay, lightwalletd ensures the transaction reaches full nodes and miners for inclusion in a block. Once the transaction is broadcasted, it propagates through the network until a miner includes it in a block.

2. Receiving a Transaction

The light client periodically queries lightwalletd for new blocks and transactions using a "compact block" protocol using gRPC, which efficiently reduces the amount of data the client needs to download. Lightwalletd provides these compact blocks, containing enough information for the light client to scan for relevant transactions without needing to download full blocks. The light client scans the compact blocks for transactions destined for the user's address, checking both transparent and shielded addresses. For shielded addresses, the client uses its view key to detect and decrypt relevant transactions from the compact blocks. Once a relevant transaction is detected and validated, the light client updates the user's balance to reflect the received funds and may display transaction details such as the amount received and the sender's address (if it's a transparent transaction).

3. Looking Up a Transaction

To look up a specific transaction, the light client sends a query to lightwalletd requesting its details. This query might include the transaction ID or other identifying information. lightwalletd then searches its database, which is synchronized with the blockchain, for the requested transaction and returns the transaction details to the light client. For transparent transactions, this includes information like inputs, outputs, and transaction amounts. For shielded transactions, the light client may need to decrypt the details using its private keys to access the full transaction information.

Lightwalletd <> Full Node Interactions

1. Synchronization with Full Node

lightwalletd does not interact directly with the Zcash network in the same way a full node does. Instead, it relies on a Zcash full node to access the blockchain. Here is how this interaction typically works:

1. Connection Setup:
    - lightwalletd establishes a connection to one or more Zcash full nodes. The connection is established using standard RPC (Remote Procedure Call) supported by the Zcash full node. This connection allows lightwalletd to request block data and receive updates about new blocks as they are mined.
2. Requesting Blocks:
    - Initial Sync: When lightwalletd first starts, it needs to synchronize with the blockchain. It requests blocks from the full node starting from the genesis block or the last known block if it has been previously synchronized.
    - Ongoing Sync: After the initial synchronization, lightwalletd continues to request new blocks as they are added to the blockchain. This can be done by polling the full node or by subscribing to notifications if the full node supports such a feature.
3. Block Fetching:
    - The full node provides the requested blocks to lightwalletd. These blocks contain all the transaction data, including both transparent and shielded transactions.

2. Processing into Compact Block Format

Once lightwalletd receives the blocks from the full node, it processes them into a more lightweight format suitable for light clients. This is what happens during this processing:

1. Filtering Transactions:
    - lightwalletd filters the transactions within each block to create a compact representation. This involves including only the necessary metadata and identifiers, excluding the full transaction details.
2. Creating Compact Blocks:
    - Transparent Transactions: The compact block includes the transaction ID and the indices of the inputs and outputs for transparent transactions.
    - Shielded Transactions: The compact block includes the commitment, nullifiers, and other cryptographic elements required to identify and validate the transactions without revealing sensitive details.
3. Serialization:
    - The compact block is serialized into a format that can be efficiently transmitted to and processed by light clients. This typically involves compressing the data and indexing it to allow quick lookups and scans.

3. Providing Compact Blocks to Light Clients

Once the blocks are processed into the compact format, lightwalletd provides them to connected light clients:

1. Serving Requests:

    - Light clients request compact blocks from lightwalletd to stay updated with the latest blockchain data. These requests can include parameters like the block range they are interested in, which helps in resuming from where they left off.
2. Data Transmission:
    - lightwalletd transmits the compact blocks to the light clients over the network. This data is smaller in size compared to full blocks, making it efficient for light clients to download and process.
3. Incremental Updates:
    - As new blocks are mined and added to the blockchain, lightwalletd continuously fetches these blocks from the full node, processes them into a compact format, and makes them available to light clients. This ensures that light clients can stay in sync with the network without needing the full blockchain data.

Light Clients <> Full Nodes

Light wallets have the option to engage directly with Zcash full nodes for transaction-related activities such as sending, receiving, or querying transactions. This interaction begins with the light wallet establishing connections with one or more full nodes on the Zcash network. Once connected, the light wallet can send transactions directly to the full nodes for broadcasting across the network and monitor incoming transactions by querying the connected full nodes for updates on relevant addresses. Similarly, to retrieve specific transaction details, the light wallet forwards queries directly to the connected full nodes, typically based on transaction IDs. Direct interaction with full nodes promotes decentralization by bypassing the need for intermediary servers like lightwalletd. Thus have direct control over their transactions and data, without relying on third-party servers. Direct communication with full nodes may also result in lower latency for transaction broadcasting and retrieval. However, maintaining direct connections with multiple full nodes can be resource-intensive for light wallets, requiring sufficient network bandwidth and computational resources. When a light client wants to synchronize with the blockchain to obtain the latest transaction data, it requests blocks from the connected full nodes, which respond by providing the requested full blocks. Fetching full blocks can be resource-intensive, especially for light clients running on devices with limited storage and bandwidth. Implementing direct interaction entails handling various network protocols and managing connections, which can add complexity to light wallet software. Finally, direct communication exposes certain metadata to full nodes, potentially compromising user privacy, especially if malicious entities operate nodes.

Types of Transactions

Zcash offers three types of transactions: *shielded*, *unshielded*, and *partially shielded*. Unshielded transactions, often referred to as transparent transactions, closely resemble transactions found in Bitcoin. These transactions utilize transparent addresses (*t-addresses*) and do not rely on zero-knowledge proofs for privacy. In essence, unshielded transactions are fully visible on the blockchain, exposing details such as **sender** and **recipient addresses**, **transaction amounts**, and accompanying data like **transaction fees**. While transparent transactions provide a familiar experience for users accustomed to traditional cryptocurrencies, they lack the enhanced privacy features offered by shielded transactions.

Shielded transactions, also known as z-transactions, provide the highest level of privacy by fully concealing transaction details. These transactions are stored in the **shielded pool**, a special segment of the Zcash blockchain that holds all shielded transactions. Thanks to the use of cryptographic techniques such as zk-SNARKs (Zero-Knowledge Succinct Non-Interactive Arguments of Knowledge) and encryption, the shielded pool obfuscates critical transaction details, including sender and recipient addresses, transaction amounts, and memo fields. By leveraging zk-SNARK proofs, shielded transactions validate transactions without revealing sensitive information. This ensures that user privacy is safeguarded while maintaining the integrity and security of the blockchain.

Partially shielded transactions represent a hybrid approach, allowing users to move funds between transparent (t-addresses) and shielded (z-addresses) addresses. These transactions can be further categorised into t-to-z (shielding) and z-to-t (deshielding) transactions, each serving unique privacy objectives. When funds are moved into the shielded pool (through t-to-z transactions), they are converted into shielded notes, which are encrypted and hidden from public view. Conversely, when funds are moved out of the shielded pool (through z-to-t transactions), only the necessary transaction details are revealed to maintain transparency where required. Below, we detail each type of transaction.

Unshielded Transactions (Transparent Transactions)

Unshielded transactions in Zcash are similar to transactions in Bitcoin. They use transparent addresses (**t-addresses**) and do not employ zero-knowledge proofs for privacy.

How Unshielded Transactions Work

1. **Addresses:** Transparent addresses in Zcash start with a "t" (e.g., t1abcd...). These addresses are publicly visible on the blockchain.
2. **Transaction Details:** The details of unshielded transactions are fully visible on the blockchain, including:
    - The sender's address
    - The recipient's address
    - The transaction amount
    - Any accompanying data (e.g., transaction fees)
3. **Transaction Creation:**
    - The user creates a transaction specifying the amount to send from their t-address to the recipient's t-address.
    - The transaction includes inputs (the sources of funds, which are previous transactions outputs) and outputs (the destination addresses and amounts).

4. **Signing:** The transaction is signed with the sender's private key to authorize the spending of their funds.
5. **Broadcasting:** The signed transaction is broadcast to the Zcash network.
6. **Validation:**
    - Full nodes validate the transaction to ensure it adheres to consensus rules (e.g., the sender has sufficient funds).
    - Once validated, the transaction is included in a block and added to the blockchain.

Partially Shielded Transactions

Partially shielded transactions in Zcash involve moving funds between transparent addresses (t-addresses) and shielded addresses (z-addresses). These transactions can be categorised into two types:

t-to-z Transactions: Moving funds from a transparent address to a shielded address.
z-to-t Transactions: Moving funds from a shielded address to a transparent address.

t-to-z Transactions (Shielding)

The flow of the shielded transactions can be summarised as follows:
1. **Sender's Transparent Address:** The sender uses a t-address to initiate the transaction.
2. **Recipient's Shielded Address:** The recipient uses a z-address to receive the funds.
3. **Transaction Details:**
    - The input comes from the sender's t-address. The input value (e.g., 10 ZEC) is used to create a new shielded note. This note includes a commitment to the value and the recipient's z-address.
    - The output is to the recipient's z-address.
    - The transaction amount and sender's t-address are publicly visible, but the recipient's z-address and received amount are encrypted and hidden on the blockchain.
4. **Shielding Process:**
    - The user creates a transaction that specifies moving funds from their t-address to a z-address.
    - A zk-SNARK proof is generated to ensure the correctness of the shielded part of the transaction.
    - The transaction is signed with the sender's private key and broadcast to the network.
5. **Validation:**
    - Full nodes validate the transparent input and ensure the zk-SNARK proof is correct.
    - Upon validation, the value is effectively moved into the shielded pool by recording the new shielded note in the blockchain. The transparent input is consumed and no longer available.

In this type of transaction, the **shielded pool** is affected as follows: it increases by the value of the newly created shielded note. While the transaction details, such as the sender's t-address and the amount, are publicly visible, the specifics of the shielded note, including the recipient's z-address and the amount, remain hidden, thus ensuring the recipient's privacy.

z-to-t Transactions (Deshielding)

The flow of the deshielding transactions can be summarised as follows:
1. **Sender's Shielded Address:** The sender uses a z-address to initiate the transaction.
2. **Recipient's Transparent Address:** The recipient uses a t-address to receive the funds.
3. **Transaction Details:**
    - The input comes from the sender's z-address.
    - The output is to the recipient's t-address.
    - The sender's z-address and the amount being deshielded are hidden, but the recipient's t-address and received amount are publicly visible.
4. **Deshielding Process:**
    - The user creates a transaction specifying the movement of funds from their z-address to a t-address.
    - A zk-SNARK proof is generated to ensure the correctness of the shielded part of the transaction.
    - The transaction is signed with the sender's private key and broadcast to the network.
5. **Validation:**
    - Full nodes validate the zk-SNARK proof and the transparent output.
    - Upon validation, the value is effectively moved out of the shielded pool by marking the note as spent and recording the new transparent output on the blockchain.

In this type of transaction, the **shielded pool** is affected as follows: it decreases by the value of the spent shielded note. While the transaction details, such as the recipient's t-address and the amount, are publicly visible, the specifics of the shielded note, including the sender's z-address and the amount, remain hidden, thereby ensuring the privacy of the sender.

Shielded Transactions

Shielded transactions in Zcash leverage zk-SNARKs and encryption to provide enhanced privacy, ensuring that transaction details (including sender and recipient addresses, transaction amounts, and memo fields) are hidden from public view. Here's a detailed breakdown of how shielded transactions work and what exactly is obfuscated.

How Shielded Transactions Work

1. **Addresses**: Shielded addresses (z-address) in Zcash start with "zs" (e.g., zs1abcd...). These addresses can send and receive shielded transactions. When a user creates a new z-address (shielded address) in Zcash, a corresponding spending key is generated. The viewing key, which can be derived from the spending key, allows the owner to view details of incoming shielded transactions without the ability to spend the funds. This viewing key can be shared with trusted parties for monitoring purposes, such as auditing or transaction tracking. When a z-address is shared, it allows others to send funds securely and privately to the owner, but only

the owner, who holds the spending key, can actually spend the funds. The viewing key can be used separately to monitor incoming transactions without compromising the security of the funds.

2. **Transaction Creation:**
   - **Inputs and Outputs**: A shielded transaction involves shielded inputs (existing shielded balances) and shielded outputs (the destination addresses for the shielded funds).
   - **Zero-Knowledge Proofs**: At the core of a shielded transaction is the zk-SNARK proof. This proof allows the sender to prove that the transaction is valid without revealing the transaction details. The proof ensures that:
     - The sender has enough funds to cover the transaction.
     - The transaction has been constructed correctly according to Zcash's rules.

     The zk-SNARK proof is constructed using a proving key. This proof is compact and can be quickly verified by nodes in the network. It ensures that no sensitive information about the transaction (such as the sender's address, the recipient's address, or the amount) is revealed.
   - **Signing:** The transaction is signed with the sender's private key, which authorizes the spending of the shielded funds.

3. **Broadcasting the Transaction**
   - Encryption: The transaction, including the zk-SNARK proof, is encrypted and broadcast to the Zcash network.

4. **Validation:** Full nodes on the network validate the zk-SNARK proof without needing to see the actual transaction details. If the proof is valid, the transaction is added to the blockchain.

What is obfuscated in Shielded Transactions

Shielded transactions are designed to hide the following details:

1. *Sender's Address*
The sender's shielded address (z-address) is hidden in the transaction. The zk-SNARK proof confirms that the sender has the right to spend the funds without revealing the sender's identity.
2. *Recipient's Address*
The recipient's shielded address is also hidden. The transaction does not expose which address is receiving the funds, ensuring the recipient's privacy.
3. *Transaction Amount*
The amount being sent in the transaction is concealed. This prevents observers from knowing how much money is being transferred, protecting the financial privacy of both the sender and the recipient.
4. *Transaction Memo Field*
Shielded transactions can include a memo field, which allows users to attach additional information to the transaction. This memo is encrypted and only visible to the recipient.

Privacy Implications

Obfuscating the above-mentioned details of a transaction allows shielded transactions to provide the following privacy properties:

- *Transaction Graph Privacy:* The links between senders and recipients are hidden, making it difficult to trace the flow of funds.
- *Amount Privacy:* The amounts being transacted are kept confidential, preventing outsiders from gaining insights into users' financial activities.
- *Enhanced Security:* By hiding transaction details, shielded transactions protect users from targeted attacks and financial profiling.

Quick Overview of the zk-SNARKS

In the context of Zcash, generating a zk-SNARK (Zero-Knowledge Succinct Non-interactive Argument of Knowledge) for a transaction ensures several key aspects of the transaction's correctness without revealing any sensitive information.

Key Aspects of Correctness in zk-SNARKs

1. Balance Preservation: The zk-SNARK must demonstrate that the total value of the inputs equals the total value of the outputs, ensuring no new money is created out of thin air (except for miner fees, which are allowed to be deducted).
2. Ownership: The zk-SNARK must confirm that the payer has the right to spend the input notes. This means that the zk-SNARK shows that the sender knows the secret key corresponding to the notes being spent without revealing the key itself.
3. Validity of the Coins: The zk-SNARK must show that the input coins being spent are valid and not previously spent. This involves verifying the note's serial number against a set of previously spent notes. This prevents double-spending.
4. Correctness of Encryption: The zk-SNARK must ensure that the encryption of the output notes is correct. This involves verifying that the notes are encrypted with the correct recipient addresses and amounts.

A Simplified Overview of zk-SNARK Construction

This section provides a simplified overview of how zk-SNARKs are constructed, breaking down the complex process into easy-to-understand steps.

1. Input and Output Notes:
    - Input Notes: Represented by commitments that hide the actual value and owner of the notes being spent.
    - Output Notes: Similarly, represented by commitments that will be received by the recipients.

2. Commitments and Nullifiers:

- Commitments: Bind the values and recipients' addresses in a cryptographic commitment that hides these values.
- Nullifiers: Unique identifiers for each note that ensure a note cannot be spent more than once. Nullifiers are derived from the note and the owner's secret key.

3. Proof Components:
    - Knowledge of Values: Proves knowledge of the values being transferred without revealing the values themselves.
    - Non-reusability: Ensures the input notes have not been previously spent by checking the nullifiers.
    - Correctness of Commitments: Ensures that the commitments to the input and output notes are correctly formed.
    - Balance Check: Proves that the total value of inputs matches the total value of outputs, allowing for a fee.

Threat model and potential attacks

In this section, we outline the threat model and potential attacks against Zcash. While many of these threats are common to other blockchain systems, such as attacks on the blockchain (double-spending attempts, 51% attacks, selfish-ming attacks, and so on) and network-level attacks such as (Denial of Service attacks, Sybil attacks, and eclipse attacks), our focus here is on privacy-specific attacks. Robust protocol development can mitigate many of these general attacks. However, our primary concern are the privacy attacks, which aim to undermine the anonymity and confidentiality of Zcash users. By exploring these privacy threats, we highlight the importance of continuously enhancing privacy-preserving features in Zcash. While network-level attacks are powerful, @@ [ ].

Types of Adversaries

In the context of Zcash, various types of adversaries can be modelled to understand potential risks and vulnerabilities. These adversaries can be categorized based on their capabilities and scope of action:

1. **Active Adversaries**: These adversaries can actively interfere with the network. They might corrupt and gain full access to entities within the network, such as nodes or servers. Active adversaries can introduce malicious nodes to disrupt the network's integrity and operations, and compromise security.
2. **Passive Adversaries**: Passive adversaries primarily observe and record publicly accessible information within the network without directly interfering with the operations. They can monitor the blockchain and network traffic data, including transaction details and timing information, to gather insights and infer user behaviour. Passive adversaries rely on data correlation and traffic analysis, highlighting the need for strong encryption and traffic obfuscation techniques.
3. **Combined Adversaries**: These adversaries utilize both active and passive methods to achieve their objectives. By combining observational techniques with active interference, these adversaries can execute more sophisticated attacks, such as de-anonymizing users by correlating network traffic patterns with blockchain data.

Adversaries can also be categorized by their scope of observation and control:

1. **Local Adversaries**: Local adversaries have limited scope, typically restricted to a specific area or network segment. Examples include ISPs or telecom providers that can record network traffic they route. Their observations are confined to the data passing through their infrastructure, making their reach limited but still potentially impactful on user privacy. Local adversaries emphasize the importance of securing individual network segments and using encrypted communications.
2. **Global Adversaries:** These adversaries have extensive capabilities to monitor the entire network. A global passive adversary can observe all network traffic, providing them with a comprehensive view of network operations and user interactions. This might

include powerful entities like state-sponsored actors or large surveillance organisations that can monitor multiple points in the network simultaneously. Global adversaries underscore the necessity for end-to-end encryption, metadata obfuscation and decentralized network structures to minimize the risk of broad surveillance.

Privacy Attacks

Blockchain Analytics and Transaction Linking

We begin by discussing blockchain monitoring attacks to highlight that, despite robust cryptographic protocols, blockchain analytics can still compromise user anonymity. When combined with metadata leakage on the wire, described in the next section, the impact on user privacy can be profound.

Transparent transactions (t-to-t)

Transparent transactions in Zcash can be directly analyzed on the blockchain. Because the attacker can see both the sender and the recipient, as well as the transaction amount, they can easily trace the movement of funds and the relationships between the involved parties. By analyzing the inputs and outputs of transactions, an attacker can link multiple addresses together, gaining detailed insights into the financial activities and spending patterns of users.

Using simple heuristics, such as the change heuristic, attackers can group addresses likely controlled by the same user based on transaction patterns and behaviour. This clustering can reveal how funds are distributed among different addresses owned by a single individual or entity. By mapping the transaction network, attackers can identify key addresses that act as hubs or intermediaries, exposing relationships between different users and entities.

Transaction graph analysis further allows attackers to trace the entire transaction history of an address. By examining the flow of funds through the network, attackers can reconstruct a user's financial history, identifying all transactions involving their addresses. This analysis can also uncover hidden relationships between addresses and users. For example, if two addresses frequently transact with each other, they are likely controlled by the same user or closely associated users.

These privacy threats are inherent to transparent transactions, whether in Bitcoin or Zcash and can significantly compromise user anonymity and confidentiality.

Partially Shielded Transactions

Most users do not fully utilize the privacy features of Zcash [CITE KAPPOS and METRICS]. Instead, they often move funds between a transparent and a shielded address (t-to-z and z-to-t transactions). These partially shielded transactions reveal either the sender's or the recipient's address and the transaction amount. By monitoring such transactions, an adversary can link

shielded and transparent addresses, gradually de-anonymizing users. While the cryptographic techniques underlying Zcash are robust and provide strong privacy guarantees in theory, the actual privacy users experience in practice heavily depends on how they use these features.
In a t-to-z transaction, a user transfers funds from a transparent address (t-address) to a shielded address (z-address). Although the z-address remains hidden, the transfer itself and the amount are visible on the blockchain. The attacker can track when and how much funds are moved into the shielded pool, which can be correlated with future transactions. In a z-to-t transaction, funds are moved from a shielded address (z-address) to a transparent address (t-address). The withdrawal from the shielded pool is observable, even though the z-address remains hidden. When funds are moved quickly from a transparent address to a shielded address and then back to a transparent address, it reduces the effectiveness of the privacy mechanism. This can link the depositor and the recipient, reducing the effective anonymity provided by the shielded pool. If a user frequently performs t-to-z transactions at similar times or shortly before corresponding z-to-t transactions, these patterns can reveal user behaviour and link transactions. Moreover, by monitoring the amounts transferred to the shielded pool, attackers can correlate large or unique amounts to subsequent z-to-t transactions. Transactions involving very distinct values can be traced more easily. If the same unique amount is moved in and out of the shielded pool, it can be easily correlated, thereby compromising anonymity.

In May 2022, a significant update to Zcash was released, introducing the highly anticipated auto-shielding feature
(https://electriccoin.co/blog/halo-arc-for-zcash-proposed-for-release-later-this-year/). However, without a robust network-level privacy layer accompanying it, this advancement alone will not prevent adversaries from fingerprinting Zcash users.
Once the adversary obtains a user's t-address, either through direct means or by extracting it from a unified address, the adversary could initiate a transaction to the identified t-address and monitor the user's wallet for the subsequent broadcasting of the t-to-z auto-shielding transaction. By repeatedly initiating transactions and observing the corresponding auto-shielding broadcasts, the adversary can gather increasingly granular IP measurements over time. Thus, while the introduction of auto-shielding represents a significant advancement in Zcash's privacy features, it underscores the critical importance of implementing robust network-level privacy.

Network Level Attacks

Over the past decade, extensive documentation has highlighted the privacy threats posed by network metadata exposure, particularly IP addresses, within the Bitcoin broadcast protocol, specifically concerning full nodes [Biryukov et al., 2014, Heilman et al., 2015]. Those attacks thus are inherent to the Zcash network. By monitoring the propagation of transactions and blocks among nodes, an attacker can deduce the network's topology. Nodes that frequently send transactions in quick succession or exhibit similar behavioural patterns can be clustered together and inferred to be under the control of the same entity even when using Tor [Biryukov et al., 2014]. Additionally, attackers can exploit anomalies in transaction propagation, such as delays or unusual patterns, to infer connections between nodes. For instance, if a transaction consistently follows a specific path through the network, the nodes along this path are likely linked. Finally, by correlating the timing and sequence of transaction arrivals across multiple nodes, an attacker can trace the path of a transaction back to its origin.

Network-level attacks are a significant threat not only to the network as a whole but also to individual users, as they can expose personal transaction details and undermine the intended privacy protections of the Zcash protocol. Since those attacks exploit the metadata on the wire they do not require direct access to the user's device. Although the connection between wallets, lightwalletd and full nodes is encrypted and authenticated, side-channel attacks exploiting network metadata can reveal information about the wallet's behaviour. The attackers can monitor the network traffic remotely, or use compromised nodes or internet infrastructure (e.g., ISP or telco). By analysing metadata information like timing, volume, frequency or traffic patterns, the adversary can link transaction history with the user's IP address, compromising the anonymity of users. Each interaction of a wallet with the Zcash network, including fetching compact blocks, submitting a transaction, or fetching a memo field, creates characteristic patterns.

Volume analysis involves examining the size of data packets exchanged within the Zcash network to infer the type of transactions being conducted. For instance, shielded transactions in Zcash typically result in larger or more variable packet sizes compared to transparent transactions (due to the use of zk-SNARKs). By analyzing packet sizes, attackers can gain insights into transaction types. Furthermore, attackers can leverage *traffic pattern analysis*, employing machine learning and statistical techniques to recognize patterns in network traffic that correlate with specific transaction types within the Zcash network. This analysis enhances the confidence level in identifying transaction characteristics.

Additionally, attackers can observe the precise *timing of data packet transmissions* and receptions within the Zcash network. By correlating these timings with known transaction patterns, attackers can infer the initiation and completion times of transactions (Tramer et al., 2020). Moreover, by aligning these timings with transactions recorded on the blockchain, attackers can link individual transactions and addresses, whether transparent or shielded, to users' IP addresses.

Frequency analysis involves monitoring the rate of data packet transmission and reception within the Zcash network. Attackers performing frequency analysis aim to deduce user behaviour patterns. Utilizing various monitoring tools, attackers capture and analyze network traffic within the Zcash network, enabling observation of data packet flows between different nodes, including transactions, block propagation, and other network activities. Frequent spikes or peaks in network traffic often indicate heightened transaction volumes within the Zcash network. Furthermore, patterns in network activity offer additional insights into user behaviour within the Zcash network. For example, consistent spikes in network traffic during specific hours may suggest regular transaction patterns or trading activities by particular users or entities. Analyzing these patterns enables attackers to infer user behaviour effectively.

Shielded Transactions Tracing

Advanced network monitoring tools such as Deep Packet Inspection (DPI) [Cite] have the capability to identify encrypted traffic types. While they cannot decrypt the content of shielded transactions, they can recognize patterns specific to Zcash traffic associated with shielded transactions, enabling adversaries to identify users engaging in such activities. For instance, if an eavesdropper observes a wallet consistently sending substantial amounts of encrypted traffic to Zcash nodes, they may infer the user's active participation in shielded transactions. This monitoring can be conducted by global network observers or local Internet Service Providers (ISPs). By detecting a surge in encrypted traffic from a user's IP address concurrently with the appearance of a new shielded transaction on the Zcash network, the attacker can deduce that the user initiated that transaction. When a user improperly utilizes the shielded pool, such as by quickly transferring funds between transparent (t) and shielded (z) addresses, an attacker can link the user's IP address to their on-chain addresses. As we discussed in the previous section, if a user improperly utilizes the shielded pool, such as by quickly transferring funds between transparent and shielded addresses, the attacker can associate specific IP addresses with particular on-chain activities and addresses.

Compromised lightwalletd server - Threat to Light Clients

Transaction and Address Linkability

Lightwalletd—the server to which the wallet connects—can learn the transaction graph of all shielded and unshielded transactions between its users. This is because clients broadcast transactions through lightwalletd, hence lightwalletd learns where each transaction originates. When a light client sends a signed transaction to a lightwalletd, the lightwalletd can observe the timing, source IP address, and transaction details. If a light client frequently sends transactions through the same lightwalletd server, it can become easier for that server to identify patterns. Similarly, when a wallet checks for received transactions, the wallet will indicate exactly which transactions it wants the ciphertext for. This allows lightwalletd to discern the destination of each transaction. By combining this information, lightwalletd can map out who sends and receives transactions among its users, thereby compromising their privacy.

IP Address Tracking

Every time a light client requests compact blocks or broadcasts transactions, the lightwalletd can log the client's IP address. Over time, this can be used to build a profile of the light client's activity. An adversary operating a lightwalletd can use IP addresses to track and identify users, potentially compromising their privacy. This is particularly concerning if the client uses the same network or IP address for repeated interactions.

Block Request Patterns

The pattern in which a light client requests blocks can also leak information. For example, if a light client regularly queries specific ranges of blocks, it might reveal information about the timing and frequency of the user's transactions. lightwalletd server can analyze the requested block ranges and infer the client's activity periods. This can be used to narrow down the time windows during which the client is active, potentially correlating this information with other network activities. Repeated requests for updates on specific blocks and transactions associated with a particular address indicate user association, allowing to link the user with an address. Moreover, the timing of update requests can enhance correlation accuracy. For example, if a light client regularly requests updates shortly after a transaction is sent to one of its addresses, the lightwalletd can infer that this address belongs to the light client.

Transaction Broadcast Timing

The time at which a transaction is broadcast can be used to infer additional information. If a light client frequently sends transactions at certain times of the day, a lightwalletd server can use this timing information to create a usage pattern. Predictable timing can allow a lightwalletd to correlate transactions with specific users, especially if combined with other metadata such as IP addresses and block request patterns.

The attacks described below in the context of lightwalletd also apply to direct client-to-full-node connections, if such connections exist.

Addressing Threats with the Nym Mix Network

Overview of the Nym Mix Network and its Features

The Nym Mix Network is an advanced privacy infrastructure designed to enhance online anonymity by obfuscating network traffic patterns. It provides a robust solution for protecting users against surveillance and metadata analysis, both from local and global attackers, ensuring that their online activities remain private. The Nym Mix Network relies on a distributed network of nodes operated by various participants. To encourage the operation of these nodes and ensure network robustness, Nym incorporates an incentive mechanism using its native cryptocurrency token (NYM). Operators are rewarded for their contribution to the network, fostering a self-sustaining ecosystem.

At the core of the Nym Mix Network is its mixnet architecture, which routes internet traffic through a series of mix nodes. Each mix node reorders and re-encrypts messages, making it exceedingly difficult for adversaries to trace the origin, destination, or content of the traffic. This process of mixing disrupts any direct correlation between incoming and outgoing packets, providing strong anonymity guarantees. To further hinder the correlation of packets and communicating parties, each packet in the Nym mixnet traverses a different, independent path (in contrast to Tor circuits). The layered topology of the Nym mixnet ensures scalability as the user base grows.

Nym employs layered encryption to ensure that each layer of the network only has access to specific information. The initial message is encrypted by the sender, with each layer being decrypted only by the corresponding mix node. This multilayer encryption ensures that no single node can access both the sender and recipient information. One of the primary goals of the

Nym Mix Network is to protect against metadata leakage. Metadata, such as timing and frequency, are hidden by shuffling the order of the packets at each mix hop. Notably, Nym deviates from the traditional approach of batching packets in groups, which often results in high latency overhead, and instead employs a mixing technique that relies on delaying each packet individually by a random amount of time specified by the sender of the packet. This prevents traffic analysis attacks that rely on the timing and frequency of the packets. To further hinder traffic analysis, the Nym mixnet generates cover traffic to further disrupt unique patterns of communication.

Nym offers network-layer privacy and is designed to be compatible with existing internet protocols and applications. It can integrate with various services, including email, messaging, and cryptocurrency networks, providing enhanced privacy without requiring significant changes to those applications.

Nym node operators are rewarded for their contribution to the network, fostering a self-sustaining ecosystem. Also, delegators play a crucial role in maintaining the network's quality of service. Delegators are individuals or entities that stake their tokens to support and delegate responsibility to specific node operators. They contribute to the network by selecting trustworthy and efficient node operators, thereby ensuring the overall integrity and reliability of the Nym Mix Network. Delegators earn rewards based on the performance of the nodes they have delegated to. These rewards are typically a portion of the incentives earned by the nodes for their contribution to the network. By receiving rewards, delegators are incentivized to carefully select reliable node operators, as the performance of these nodes directly impacts the rewards they earn. This approach acts as a reputation systems that track the behaviour and performance of node operators. Nodes that engage in malicious activities or provide poor quality of service get their reputation decreased, reducing their chances of being selected into the active set of mixnet nodes. The Nym network eventually plans to prevent DDoS and other

How the Nym Network Enhances Privacy and Security in Zcash Transactions

Nym mix network can provide robust network-level privacy to users and entities within the Zcash network. A mixnet was originally seen as one way to mitigate these attacks on user-level privacy (Sasson et al., 2014) and it has been shown users are confused about their privacy on the network-level using ZCash (Halpn et al., 2021) The use of a mixnet has been concretely proposed to increase the security of the mixnet (Kappos and Piotrowska, 2019), but until the advent of the Nym network there was not a usable mixnet.

Wallet Integration

Integrating Zcash light wallets with the Nym network allows for safeguarding Zcash users. Instead of directly communicating with the lightwalletd servers, the communication would be routed through the Nym mix network. This means that instead of exposing their IP address and communicating directly with a centralized server, the user's transaction data is relayed through a series of mix nodes within the Nym network. These mix nodes shuffle and obfuscate the transaction data, making it very difficult for adversaries to trace the origin or destination of the transaction, thereby preserving user anonymity. The Nym mixnet ensures that transaction metadata, including IP addresses, volume, timing or frequency are concealed from external observers.

In detail, when a newly created transaction or a request to fetch transaction data is sent to the lightwalletd server, the visible IP address corresponds to a Nym exit gateway, which is unrelated to the originating wallet. This prevents the server from exploiting network metadata to infer whether the requests originate from the same wallet or different ones. And in the case of a wallet sending a request related to the same on-chain address, the lightwalletd server still cannot associate the address with a network IP. Subsequently, the requested data or confirmations are routed back to the user via Single-Use Reply Blocks, ensuring that the lightwalletd server remains unaware of the user's IP address throughout the interaction.

The process of "mixing," which involves the cryptographic transformation and randomized reordering of packets at each hop, adds an additional layer of privacy and security to the communication. Even with visibility over all communications in the network, network adversaries are unable to track packets as they are relayed through the network. To fully link a wallet to its corresponding lightwalletd server, an adversary would need to control all five intermediaries involved in the packet exchange: the entry gateway, the exit gateway, and all three mix nodes along the packet's path. However, of these five intermediaries, the client algorithmically selects three mix nodes per packet, while the entry and exit gateways are chosen by clients based on their own criteria. This distributed selection process further strengthens the network's resilience against potential attacks and preserves user privacy.

Research Questions:

1. **What is the size of the current ZCash shielded (z-z) transactions?**

The answer to this question affects how many Sphinx packets the shielded transaction must be broken into. Larger packets will have a larger cost in latency and cryptographic operations, and this must be measured.

2. **What is latency tolerance, how much would a typical ZCash user accept in terms of achieving network-level privacy?**

The answer to this question determines how much latency is possible for various operations 1) sending transactions over the mixnet 2) scanning the blockchain for transactions and receiving transactions and 3) looking up a transaction. Obviously this is a subjective question but some surveys or in-person user interviews would be ideal.

3. How anonymous does the average ZCash user want to be?

The answer to this question determines the mixing delay parameters, and so impact latency. Ideally, it should be within the acceptable range of 2. In general, the higher the delay parameter the more anonymous one can be. Also, users typically think in terms of “anonymity sets” but measurements of mixnets are typically done in terms of entropy.

Bibliography:

Biryukov, A., Khovratovich, D., & Pustogarov, I. (2014). Deanonymisation of clients in Bitcoin P2P network. In *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security* (pp. 15-29).

Biryukov, A., & Tikhomirov, S. (2019). Deanonymization and linkability of cryptocurrency transactions based on network analysis. In *2019 IEEE European symposium on security and privacy (EuroS&P)* (pp. 172-184).

Halpin, H. (2021). Holistic privacy and usability of a cryptocurrency wallet. In *Proceedings of Usable Security* (USEC 2021) .

Heilman, E., Kendler, A., Zohar, A., & Goldberg, S. (2015). Eclipse attacks on {Bitcoin’s peer-to-peer network. In *24th USENIX security symposium (USENIX security 15)* (pp. 129-144).

Kappos, G., Yousaf, H., Maller, M., & Meiklejohn, S. (2018). An empirical analysis of anonymity in Zcash. In *27th USENIX Security Symposium (USENIX Security 18)* (pp. 463-477).

Kappos, G., & Piotrowska, A. M. (2019). Extending the anonymity of zcash. *arXiv preprint arXiv:1902.07337*.

Sasson, E. B., Chiesa, A., Garman, C., Green, M., Miers, I., Tromer, E., & Virza, M. (2014, May). Zerocash: Decentralized anonymous payments from bitcoin. In *2014 IEEE symposium on security and privacy* (pp. 459-474).

Tramèr, F., Boneh, D., & Paterson, K. (2020). Remote Side-Channel attacks on anonymous transactions. In *29th USENIX security symposium (USENIX security 20*) (pp. 2739-2756).