Summary on Zerocash

Zerocash was the first widespread application of zk-SNARKs, a novel form of zero-knowledge cryptography. The strong privacy guarantee of Zcash is derived from the fact that shielded transactions in Zcash can be fully encrypted on the blockchain, yet still be verified as valid under the network's consensus rules by using zk-SNARK proofs.

1. Zero-Knowledge Proof

Types of Zero-Knowledge Proofs

Interactive Zero-Knowledge Proofs
In this model, the prover and verifier engage in a series of interactions. The prover sends a statement, and the verifier challenges the prover with questions to confirm the truth of the statement. This process continues until the verifier is convinced of the statement's validity. An example is the "magical cave" scenario, where a prover demonstrates knowledge of a secret without revealing it.
Non-Interactive Zero-Knowledge Proofs (NIZKs)
NIZKs eliminate the need for interaction between the prover and verifier. The prover generates a proof that can be independently verified by the verifier. This is particularly useful in blockchain applications, where proofs need to be posted on a public ledger without direct interaction

Differents protocols

Graph Isomorphism and Graph 3-Colorability

- **Graph Isomorphism**: A protocol is presented where the prover demonstrates knowledge of an isomorphism between two graphs without revealing the isomorphism itself. This is achieved through an interactive proof system that ensures the verifier cannot distinguish between a true and false statement without additional information
- **Graph 3-Colorability**: The document discusses a zero-knowledge proof for the NP-complete problem of determining whether a graph can be colored with three colors such that no two adjacent vertices share the same color. This proof system is significant because it forms the basis for proving that all languages in NP have zero-knowledge proofs.

Universal Zero-Knowledge Proofs for NP Languages

Building upon the previous discussion, the paper demonstrates that any language in NP has a zero-knowledge proof. This is established by showing that a zero-knowledge proof exists for an NP-complete problem, and since all NP problems can be reduced to NP-complete problems, they too have zero-knowledge proofs.

Fiat-Shamir Identification Protocol

The Fiat-Shamir protocol is introduced as a method for entity authentication using zero-knowledge proofs. In this protocol, the prover demonstrates knowledge of a secret without revealing the secret itself, thereby authenticating their identity. The paper also discusses the transformation of interactive proofs into non-interactive proofs using the Fiat-Shamir heuristic, which is crucial for practical implementations of digital signatures.

2. Formal Definition of a Decentralized Anonymous Payment (DAP) System

A DAP allows direct, private payments that hide the origin, destination, and amount. It relies on two types of transactions: "Mint" (creation of coins) and "Pour" (transfer of coins). The coins are defined by:

- cm(c): cryptographic commitment of the coin
- v(c): coin value
- sn(c): unique serial number
- apk(c): public key of the owner's address

These elements are used in zero-knowledge proofs (zk-SNARKs) to ensure transaction validity without revealing sensitive information.

DAP Algorithm Construction

The DAP scheme is defined by six algorithms:

1. **Setup**: Initial setup to generate public parameters (proof/verification keys, etc.).
2. **CreateAddress**: Generates a new protected address.
3. **Mint**: Creates new coins.
4. **Pour**: Spends old coins to create new ones.
5. **VerifyTransaction**: Verifies the proofs of "Pour" and "Mint" transactions.
6. **Receive**: Detects and records received coins.

The pseudocode is detailed in Section IV of the document.

zk-SNARKs for "Pour" Transactions

In a "Pour" transaction, zk-SNARKs are used to verify the transaction without revealing sensitive details, ensuring that the user proves ownership of the old coins, their existence in the Merkle tree, correct computation of serial numbers, validity of new coin commitments, and balance preservation (incoming value equals outgoing value). The proof involves witnesses such as full data of old and new coins, secret keys of the addresses, and Merkle paths proving the coins' existence. Additionally, the depth of the Merkle tree is hard-coded, which limits the number of coins that can be processed.