Cryptography

Section 1

Understanding Zero Knowledge Proofs

Zero-knowledge proofs (ZKPs) are a cryptographic mechanism that allows a prover to convince a verifier of the truth of a statement without revealing any information about the statement itself. This concept is fundamental to ensuring privacy and security in various applications, from blockchain technology to secure communication systems.

Key Components of ZKPs

1. Privacy: The verifier learns nothing about the statement beyond its truth or falsity.
2. Completeness: An honest prover can convince an honest verifier of a true statement.
3. Soundness: A dishonest prover cannot convince an honest verifier of a false statement.
4. Interactivity: ZKPs can be interactive, requiring multiple exchanges between prover and verifier, or non-interactive, involving a single proof.

Applying the Concept of ZKP using SHA-256 Algorithm

SHA-256

SHA-256 (Secure Hash Algorithm 256) is a widely used cryptographic hash function that plays a crucial role in maintaining data security and integrity.SHA-256 is a member of the SHA-2 family of cryptographic hash functions, specifically designed to produce a 256-bit (32-byte) hash value 3. It was introduced in 2001 as a successor to the SHA-1 algorithm, which had become vulnerable to certain types of attacks.

* Padding: The input message is padded to ensure its length is a multiple of 512 bits.
* Initialization: Eight 32-bit registers are initialized with default values.
* Compression: The padded message is divided into 512-bit blocks.
* Processing: Each block goes through 64 rounds of operations, with the output of each round becoming the input for the next.
* Output: The final hash value is generated after processing all blocks.

Applications of SHA-256

SHA-256 finds widespread use in various security applications and protocols:

* Digital signatures: Ensures the authenticity and integrity of documents.
* Password hashing: Used by websites to securely store user passwords.
* SSL/TLS: Contributes to secure connections in web communications.
* Blockchain technology: Utilized in cryptocurrencies like Bitcoin and Ethereum.

Understanding ZKP using Zokrates

ZoKrates is a domain-specific language (DSL) and toolkit for creating and verifying zero-knowledge proofs on Ethereum. It aims to bridge the gap between complex cryptographic concepts and practical blockchain development.

* **High-Level Abstraction:** ZoKrates provides a high-level language for specifying computations, allowing developers to focus on the logic of their application rather than the intricacies of zero-knowledge proofs.
* **Compiler and Proof Generator**: It includes a compiler that translates the high-level specifications into provable constraint systems, and generates proofs attesting to the correctness of computations.
* **On-Chain Verification:** ZoKrates supports exporting verification smart contracts, which can be deployed on Ethereum to verify the proofs generated off-chain.
* **Ethereum Focus:** While designed for Ethereum, ZoKrates is compatible with other blockchain systems.

The idea behind this is that the generation key must take infinite time by brute force but the verification must take linear time this way transactions could be verified faster on ethereum.

Implementation process

First we take a string which is in the format of encoding of UTF-8 inorder to make the input and output uniform we make it hexadecimal so that it will be easy to compare the input string and output strings.

In the high level function we pass 4 private parameters and 2 public parameters where the private parameters must not be known to the verifier and the public parameters are the hashes of the 256 SHA function.

### **1. Private Parameters:**

* **Private inputs** (a, b, c, and d) are **hidden** from the verifier. These inputs are not revealed publicly and are kept secret by the prover.
* These private inputs serve as the **preimage** in the SHA256 hash computation. In this case, the values for a, b, c, and d are combined into a 512-bit input to compute the hash.
* The private parameters are critical when proving you know certain data without revealing it. In this case, you are proving that you know the original data (a, b, c, d) that produces the public hash.

### **2. Public Parameters:**

* **Public inputs** (ho, h1) are **visible** to both the prover and the verifier. These parameters are the **expected result** of the hash computation and will be part of the proof that gets verified.
* In this case, ho and h1 are two 256-bit outputs (combined to form a 512-bit SHA256 hash) that the prover wants to prove match the hash of the private inputs.
* The public parameters are used to verify the correctness of the proof. The verifier only needs to know these values and does not need to know the private inputs.





