
DEPARTMENT OF CSE (IOT & CS Including BCT)

Hash Functions

A **hash function** is a mathematical function that converts an input (or "message") of any size into a fixed-size output, called a **hash value** or **hash code**. Hash functions are widely used in various fields of cryptography and computer science due to their ability to quickly produce a unique representation (digest) of data, which is useful for ensuring data integrity, indexing data, and more.

Key Properties of Cryptographic Hash Functions:

1. **Deterministic**: For a given input, the hash function will always produce the same output.
2. **Fixed Size Output**: Regardless of the size of the input data, the output hash value will always have a fixed length (e.g., SHA-256 always produces a 256-bit hash).
3. **Preimage Resistance**: It is computationally infeasible to reverse the hash function (i.e., given a hash value, it's hard to find the original input).
4. **Collision Resistance**: It is computationally infeasible to find two different inputs that produce the same hash value.
5. **Avalanche Effect**: A small change in the input should result in a drastically different hash value.
6. **Efficiency**: Hash functions should be fast to compute for any input.

Uses of Hash Functions:

- **Data Integrity**: Hash values are often used to verify that data has not been altered. For example, when you download a file, the provider may give a hash of the file, and you can compute the hash yourself to ensure the file hasn't been tampered with.
- **Digital Signatures**: In public-key cryptography, hash functions are used to generate a digest of the data, which is then signed using a private key.
- **Password Storage**: Hashing passwords before storing them ensures that even if the database is compromised, the passwords remain protected.
- **Cryptographic Protocols**: Hash functions are used in various cryptographic protocols like HMAC (Hash-based Message Authentication Code) and blockchain technologies.

Secure Hash Algorithm (SHA)

The **Secure Hash Algorithm (SHA)** is a family of cryptographic hash functions published by the **National Institute of Standards and Technology (NIST)**. SHA functions are widely used for data integrity and security purposes. The SHA family includes multiple versions, each designed to provide stronger security and better performance for various applications.

SHA Family Overview

The **SHA** family consists of several algorithms, which differ in their output lengths, security levels, and efficiency. The most commonly used members of the SHA family include SHA-1, SHA-2, and SHA-3. Below is a brief overview of these algorithms:

1. SHA-1 (Secure Hash Algorithm 1)

- **Output Length**: 160 bits (20 bytes)
- **Security**: SHA-1 was widely used in the past but is now considered insecure due to vulnerabilities to collision attacks. In 2005, researchers discovered that SHA-1 could be broken with enough

computational effort, making it unsuitable for security-sensitive applications today.
- **Usage**: Although it's still supported in some legacy systems, it's no longer recommended for cryptographic security. It was used in digital signatures, certificates, and cryptographic protocols like SSL/TLS.

2. SHA-2 (Secure Hash Algorithm 2)

SHA-2 is a family of hash functions that includes several variants, with different output lengths:

- **SHA-224**: Produces a 224-bit (28-byte) hash value.
- **SHA-256**: Produces a 256-bit (32-byte) hash value.
- **SHA-384**: Produces a 384-bit (48-byte) hash value.
- **SHA-512**: Produces a 512-bit (64-byte) hash value.
- **SHA-512/224**: Produces a 224-bit (28-byte) hash value.
- **SHA-512/256**: Produces a 256-bit (32-byte) hash value.
- **Security**: SHA-2 is currently one of the most secure and widely used cryptographic hash families. It is considered to be highly resistant to collision and preimage attacks. As of now, it is recommended for most cryptographic applications, including digital certificates, signatures, and blockchain.
- **Performance**: SHA-2 is more secure than SHA-1 but is slower due to the larger bit length and more complex computations.
- **Usage**: SHA-2 is used in many security protocols, including SSL/TLS, code signing, file integrity checks, and cryptocurrency (e.g., Bitcoin uses SHA-256 for mining and transactions).

3. SHA-3 (Secure Hash Algorithm 3)

- **Output Length**: SHA-3 provides output lengths similar to SHA-2 (224, 256, 384, and 512 bits).
- **Security**: SHA-3 is based on the **Keccak** algorithm and was developed as an alternative to SHA-2. It has a different internal

structure and is designed to be resistant to some types of attacks that could potentially affect SHA-2.

- **Performance**: SHA-3 is slower than SHA-2 but offers higher security for certain applications. It can also be more efficient in hardware implementations.
- **Usage**: SHA-3 is increasingly being adopted for specific use cases and is considered a future-proof alternative to SHA-2. It is suitable for use in security protocols and applications requiring higher security assurances.

Comparison Between SHA-1, SHA-2, and SHA-3

Algorithm	Output Length	Security Level	Use Cases	Vulnerabilities
SHA-1	160 bits	Weak (broken)	Legacy systems, digital signatures	Vulnerable to collision attacks
SHA-2	224, 256, 384, 512 bits	Strong (secure)	SSL/TLS, digital certificates, blockchain	Secure (no practical collisions found)
SHA-3	224, 256, 384, 512 bits	Strong (future-proof)	Cryptographic applications needing high security	Secure (different internal structure from SHA-2)

How SHA-2 Works (SHA-256 Example)

The SHA-256 algorithm, which produces a 256-bit hash, works through a series of steps that transform the input message into a hash value:

1. **Message Padding**: The input message is padded so that its length is a multiple of 512 bits. Padding includes adding a 1 bit, followed by enough zeros to make the length fit.
2. **Message Parsing**: The padded message is divided into blocks of 512 bits, which are processed sequentially.

3. **Initial Hash Values**: SHA-256 starts with a set of eight 32-bit words (initial hash values) that are defined in the specification.
4. **Processing Blocks**: Each 512-bit block is processed using a series of logical functions and bitwise operations (e.g., AND, OR, XOR, NOT), along with a set of constants. The hash values are updated in each round.
5. **Final Hash**: After processing all the blocks, the final hash value is produced, which is a 256-bit string.

Applications of SHA Algorithms

1. **Data Integrity**: SHA is used to verify that data has not been altered. For example, file checksums often use SHA-256 to ensure the file's integrity.
2. **Digital Signatures**: SHA-2 and SHA-3 are frequently used to hash messages or documents before signing with a private key. The hash value ensures that the signature is tied to the exact content.
3. **Blockchain**: SHA-256 is integral to cryptocurrencies like Bitcoin, where it’s used in the process of mining and for securing transaction blocks.
4. **Password Hashing**: Cryptographic hashes like SHA-256 are used (though with additional techniques like salting) for securely storing passwords.