Blockchain

Nepal

Cryptographic Hash Functions

- Hash Functions **$H:\{0,1\}^* \rightarrow \{0,1\}^n$**
- Notation
  - X $\rightarrow$ (input) message
  - Y = H(X) $\rightarrow$ hash value, message digest, fingerprint
- Applications
  - As a fingerprint ; compact representation image of the message
  - Making digital signatures efficient; signing the hash of message instead of complete message
  - …
- Examples
  - MD5 (completely broken; don’t use anymore)
  - SHA family (1,2,3)

Properties of Hash Functions (Desired)

- **Easy to Compute** : given x ➔ computing H(x) is easy

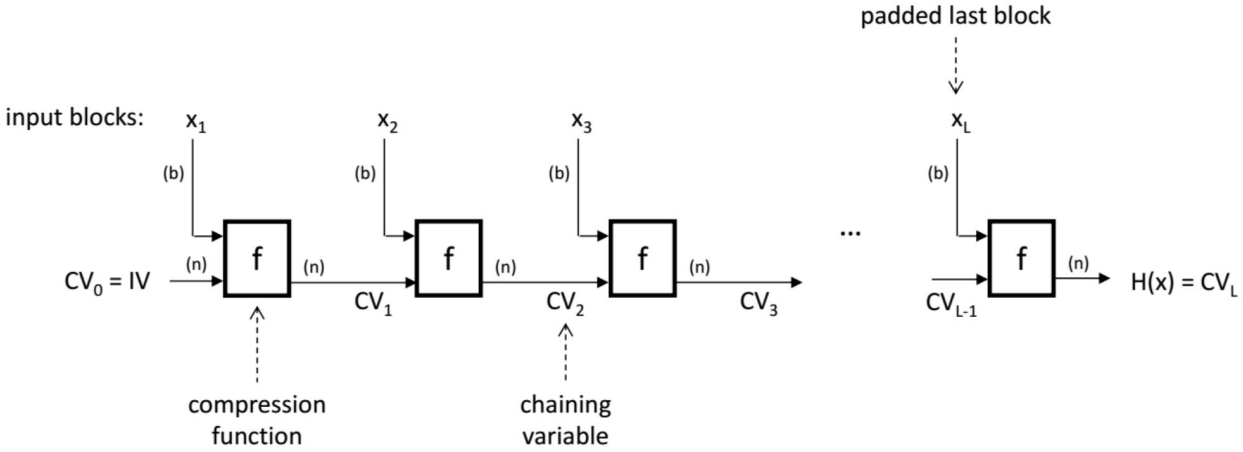
- **One-way Property** (Preimage Resistance)
  - Given y, it is computationally infeasible to find x such that y = H(x)

- **Weak Collision Resistance** (2nd Preimage Resistance)
  - Given x, it is computationally infeasible to find another x’ such that H(x’) = H(x)

- **Strong Collision Resistance** (Collision Resistance)
  - It is computationally infeasible to find two distinct x & x’ such that H(x) = H(x’)

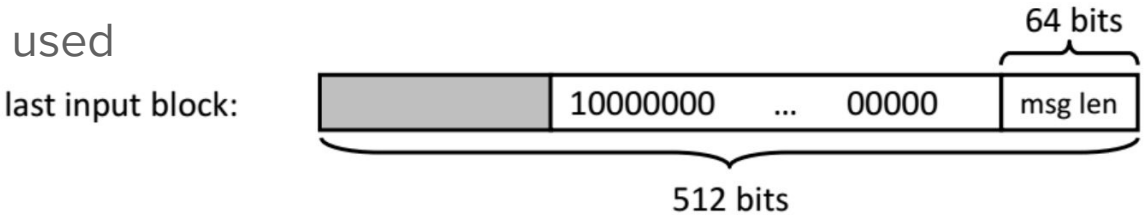
Iterative Hash Functions

- Divide input into blocks (x_1, x_2, ...x_n)
- Pad the last block if necessary
- Process the blocks as follows:



Secure Hash Algorithm 1 (SHA-1)

- Input Block size (b) : 512 bits
- Output Block size (n) : 160 bits
- Padding is always used



- CV_0
  - A = **67 45 23 01**
  - B = **EF CD AB 89**
  - C = **98 BA DC FE**
  - D = **10 32 54 76**
  - E = **C3 D2 E1 F0**

- https://en.wikipedia.org/wiki/SHA-1

SHA - 2

Family of Hash functions published by NIST in 2001

- **SHA-256 & SHA-512**
  - Identical structure, different word sizes: 32 bits & 64 bits
- **SHA-224 & SHA-384**
  - Truncated version of SHA-256 & SHA-512; different CV_0
- **SHA-512/224 & SHA-512/256**
  - Truncated version of SHA-512; CV_0 generated as specified in FIPS 180-4 Standard

- Number in names specify the output size
  Eg. SHA-256 produces 256 bit output

- https://en.wikipedia.org/wiki/SHA-2

SHA-3

- Based on Keccak algorithm (winner of NIST hash function competition, 2012)

  Parameters

  - Input block size (bits) : 1152 1088 832 576
  - Output block size (bits) : 224 256 384 512
  - Internal state (bits) : 1600

- Based on so-called “sponge construction”
  - Message blocks are XORed into subset of internal state which is permuted as a whole

- As an alternative to SHA-2, not to replace SHA-2

  https://en.wikipedia.org/wiki/SHA-3

Hash Functions in Action

```
// message to hash
String message = "A quick brown fox jumps over the lazy dog";

// select the hash function, eg. SHA1, SHA-224, SHA-256, SHA-512
MessageDigest digest = MessageDigest.getInstance("SHA-512");

// compute the hash
byte[] encodedHash = digest.digest(message.getBytes());

// base 64 encoding
String encodedHashString = new String(Base64.encode(encodedHash));
System.out.println("Hash:"+encodedHashString);
```

SHA-3 example (Lib: BouncyCastle)

```
// message to hash
String input = "A quick brown fox jumps over the lazy dog";

// sha3 hash digest
SHA3.DigestSHA3 digestSHA3 = new SHA3.Digest512();
byte[] digest = digestSHA3.digest(input.getBytes());

// hash encoding
System.out.println("SHA3-512 = " + new String(Base64.encode(digest)));
System.out.println("SHA3-512 = " + Hex.toHexString(digest));
```

Projects

1. Secure QR - code
2. End-to-end encrypted messenger
3. Decentralized Photo Storage

Secure QR Code

Android/iOS Application that

- Scans QR Code
  ➜ decrypts the message
  ➜ displays message

- Inputs message
  ➜ encrypts message
  ➜ displays QR code
  ➜ shares

- Personalize QR code with a logo

End-to-end encrypted messenger

Android/iOS Application

- Text/Data messaging
- Uses web sockets for transmission
- REST API Server as message dispatcher

End-to-end encryption (later)

- Data encrypted/decrypted by the application, Server just forwards the message

Decentralized Photo Storage

- User can upload/download Photos
- Photo timeline/gallery
- Web/Android frontend
- REST API server backend

- Decentralized blockchain Storage and digital certificate of ownership (later)

Questions?