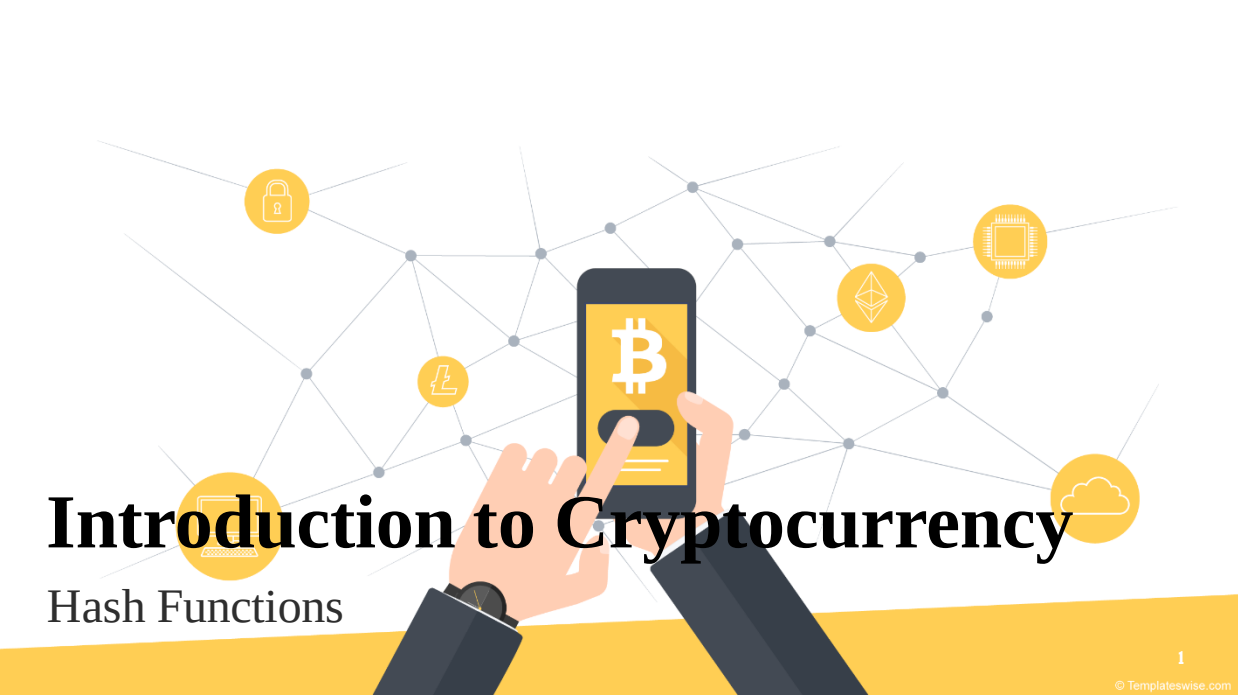


Introduction to Cryptocurrency

Hash Functions

MESSAGE AUTHENTICATION

- Goal: having received a message one would like to make sure that the message has not been altered on the way (data integrity)
  - Produce a short sequence of bits that depends on the message and on a secret key
  - To authenticate the message, the partner will compute the same bit pattern, assuming he shares the same secret key
- This does not necessarily include encrypting or signing the message
  - The message can be sent in plain, with the authenticator appended
  - This is not a digital signature: the receiver can produce the same MAC
  - One may encrypt the authenticator with his private key to produce a digital signature
  - One may encrypt both the message and the authenticator

AUTHENTICATION FUNCTIONS

- Possible attacks on message authentication:
  - Content modification
  - Sequence modification – modifications to a sequence of messages, including insertion, deletion, reordering
  - Timing modification – delay or replay messages

- Three types of authentication functions exist
  - Message encryption – the ciphertext serves as authenticator
  - Message authentication code (MAC) – a public function of the message and a secret key producing a fixed-length value to serve as authenticator
    - This does not provide a digital signature because A and B share the same key
  - Hash function – a public function mapping an arbitrary length message into a fixed- length hash value to serve as authenticator
    - This does not provide a digital signature because there is no key

MESSAGE ENCRYPTION AS AUTHENTICATION

- **Main idea**: the message must have come from **A** because the ciphertext can be decrypted using his (secret or public) key.

- Also, none of the bits in the message have been altered because an opponent does not know how to manipulate the bits of the ciphertext to induce meaningful changes to the plaintext.

- **Conclusion**: encryption (either symmetric or public-key) provides authentication as well as confidentiality.

ENCRYPTION AS AUTHENTICATOR

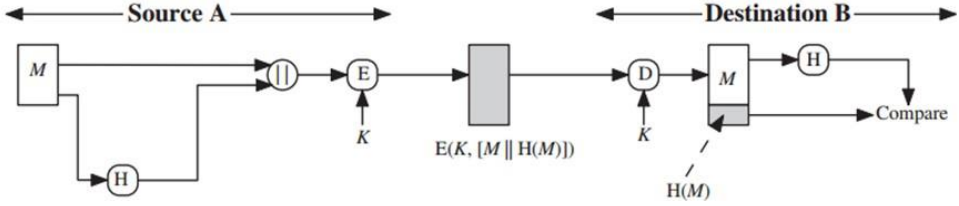
- Some careful considerations are needed here:
  - How does **B** recognize a meaningful message from an arbitrary sequence of bits?
    - He can apply the decryption key to **any** sequence of bits he receives.
  - This is not necessarily easy task if the message is some sort of binary file.
  - Immediate idea of attack: send arbitrary bit sequences to disrupt the receiver – he will try to figure out the meaning of that bit sequence

- Defense against this type of attack: add to the message a certain structure such as an error-correcting code (e.g., check-sum bits) and then encrypt the whole file.
  - **B** will detect illegitimate messages because they will not have the required structure.

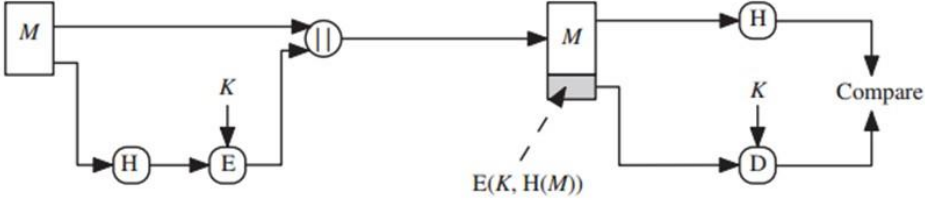
HASH FUNCTIONS

- A fixed-length hash value h is generated by a function H that takes as input a message of arbitrary length: $h = H(M)$
  - **A** sends M and $H(M)$
  - **B** authenticates the message by computing $H(M)$ and checking the match
- Requirements for a hash function:
  - H can be applied to a message of any size
  - H produces fixed-length output
  - It is easy to compute $H(M)$
  - Preimage resistance property: for a given h, it is computationally infeasible to find M such that $H(M) = h$.
  - Second-preimage resistance property: for a given M, it is computationally infeasible to find $M' \neq M$ such that $H(M') = H(M)$.
  - Collision-resistance property: it is computationally infeasible to find M, M' with $H(M) = H(M')$

APPLICATIONS OF HASH FUNCTIONS

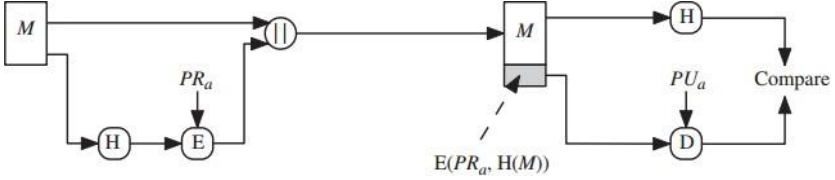


(a) Classical encryption of message + hash

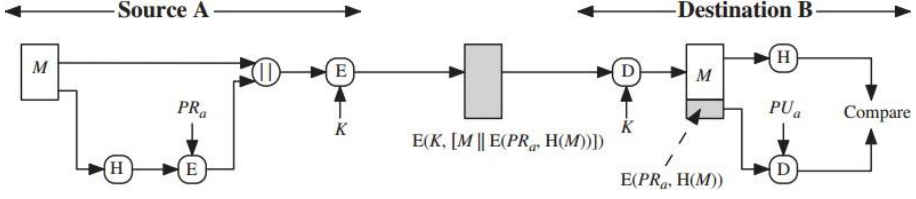


(b) Only the hash value is encrypted

APPLICATIONS OF HASH FUNCTIONS

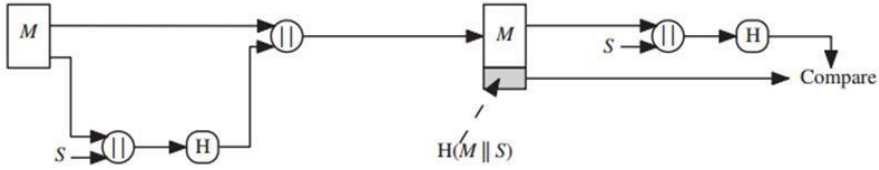


(c) As in (b) but with private key (provides digital signature)

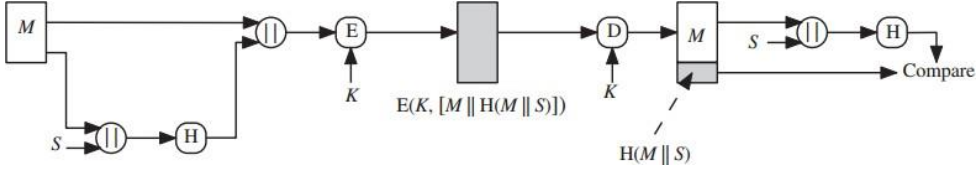


(d) Hash is encrypted with an asymmetric system, then a second encryption is applied

APPLICATIONS OF HASH FUNCTIONS

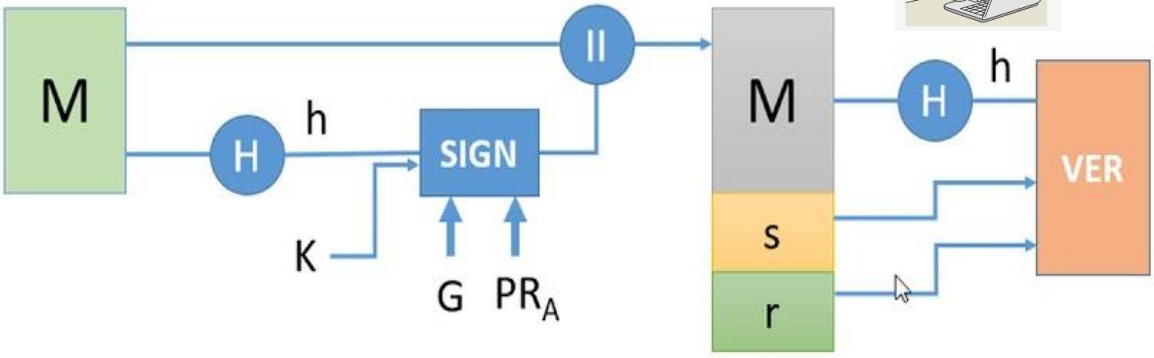


(e) No encryption here but the hash is applied to a message where a secret text *S* has been appended



(f) As in (e), but with encryption

DIGITAL SIGNATURE SCHEME



Signature = {s, r}

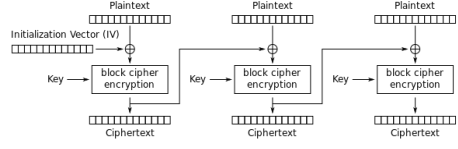
A FEW SIMPLE HASH FUNCTIONS

- Bit-by-bit XOR of plaintext blocks: $h = D_1 \oplus D_2 \oplus \cdots \oplus D_N$
  - Provides a parity check for each bit position
  - Not very effective with text files: most significant bit always 0
  - Attack: to send blocks $X_1, X_2, \ldots, X_{N-1}$ choose:
    $$X_N = X_1 \oplus X_2 \oplus \cdots \oplus X_{N-1} \oplus h$$
  - Does not satisfy the preimage resistance condition: "computationally infeasible to find M such that $H(M) = h$, for a given h "

- Another example: rotated XOR – before each addition the hash value is rotated to the left with 1 bit
  - Better than the previous hash on text files
  - Similar attack

A FEW SIMPLE HASH FUNCTIONS

- Another method: cipher block chaining technique without a secret key
  - Divide message into blocks $D_1, D_2, \ldots, D_N$ and use them as keys in the encryption method (e.g., DES)
  - H_0 = some initial value, $H_i = E_{D_i}(H_{i-1})$
  - $H = H_N$
  - This can be attacked with the birthday attack if the key is short (as in DES)



Cipher Block Chaining (CBC) mode encryption

- Birthday paradox: Given at least 23 people, the probability of having two people with the same birthday is more then 0.5
- **General case**: Given two sets X, Y each having k elements from the set $\{1,2,\ldots,N\}$, how large should k be so that the probability that X and Y have a common element is more than 0.5?
  - Answer: k should be larger than $\sqrt{N}$
  - If $N = 2^m$, take $k = 2^{m/2}$

BIRTHDAY ATTACK

- Suppose a hash value on 64 bits is used (as the one based on DES)
  - In principle this is secure: given M, to find a message M' with $H(M) = H(M')$, one has to generate in average 2^{63} messages M'.
- A different much more effective attack is possible:
  - **A** is prepared to sign the document by appending its hash value (on m bits) and then encrypting the hash code with its private key
  - **E** (i.e. attacker) will generate $2^{m/2}$ variations of the message M and computes the hash values for all of them.
  - **E** also generates $2^{m/2}$ variations of the message M' that she would really like to have **A** authenticating and computes the hash values for all of them
  - Birthday paradox: the probability that the two sets of hash values have one common element is more than 0.5 - she finds $M \neq M'$ such that $H(M) = H(M')$
  - **E** will offer M to **A** for hashing and then signing and will send instead M' with the signature A has produced
  - **E** breaks the protocol although she does not know **A**'s private key with a level of effort for the hash based on DES: 2^{33}

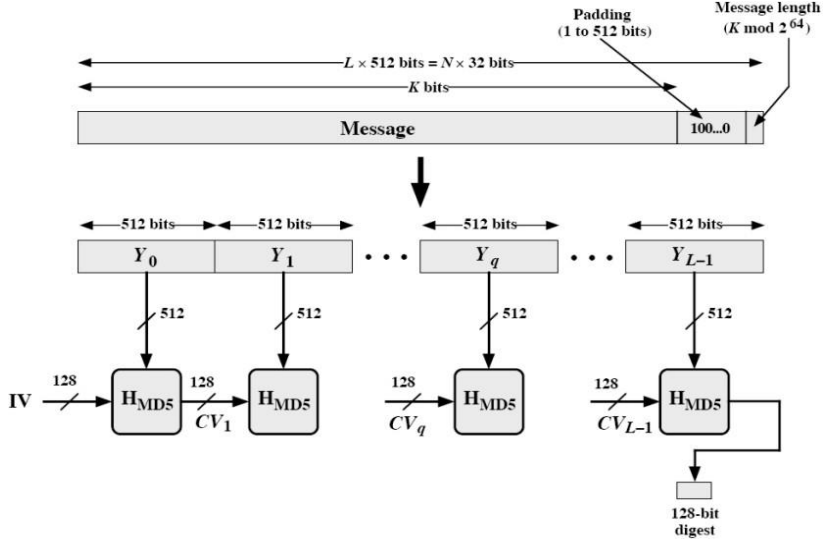
POPULAR HASH ALGORITHMS:

- MD5 (Message Digest 5)
- SHA1 (Secure Hash Algorithm 1)
- SHA2 family: **SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224, SHA-512/256**
- SHA3 (Secure Hash Algorithm 3)

MD5

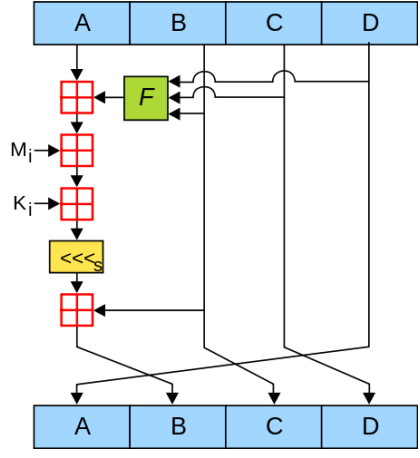
- Most popular hash algorithm until recently: concerns for its security were raised and is replaced by SHA-1, SHA-2, and SHA-3
- Developed by Ron Rivest at MIT in 1991.
- For a message of arbitrary length, it produces an output of 128 bits
  - Processes the input in blocks of 512 bits
- Idea:
  - Start by padding the message to a length of 448 bits modulo 512 – padding is always added even if the message is of required length
    - The length of the message is added on the last 64 bits so that altogether the length is a multiple of 512 bits
  - Several rounds, each round takes a block of 512 bits from the message and mixes it thoroughly with a 128 bit buffer that was the result of the previous round
  - The last content of the buffer is the hash value.

MERKLE-DAMGÅRD STRUCTURE OF MD5



MD5 OPERATIONS

- MD5 consists of 64 of these operations, grouped in four rounds of 16 operations.
- F is a nonlinear function; one function is used in each round.
- M_i denotes a 32-bit block of the message input, and K_i denotes a 32-bit constant, different for each operation.
- $\lll_s$ denotes a left bit rotation by s places; s varies for each operation.
- $\boxplus$ denotes addition modulo 2^{32}.



SECURITY ISSUES OF MD5

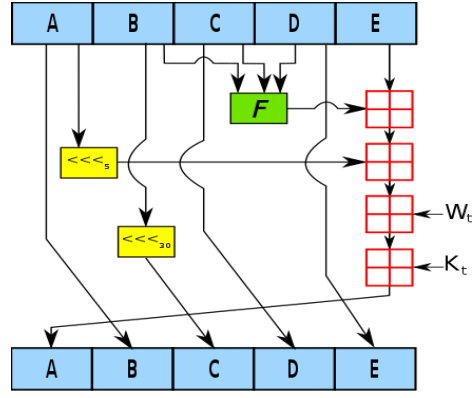
- In 1996 a flaw was found in the design of MD5.
  - While it was not deemed a fatal weakness at the time, cryptographers began recommending the use of other algorithms.
- In 2004 it was shown that MD5 is not collision-resistant.
  - As such, MD5 is not suitable for applications like SSL certificates or digital signatures that rely on this property for digital security.
- In December 2008, a group of researchers used this technique to fake SSL certificate validity.
- In 2012, the Flame malware exploited the weaknesses in MD5 to forge a Windows code-signing certificate.
  - majority of targets in Iran (more than 65%)
- Best known attack: 2013 attack by Xie Tao, Fanbao Liu, and Dengguo Feng breaks MD5 collision resistance in 2^{18} time.
  - This attack runs in less than a second on a regular computer!

SHA-1

- Developed by NSA and adopted by NIST in FIPS 180-1 (1993)
- Part of a family of 3 hashes: SHA-0, SHA-1, SHA-2
  - SHA-1 most widely used
- Design based on MD4 (previous version of MD5)
- Takes as input any message of length up to 2^{64} bits and gives a 160-bit message digest
- Microsoft, Google, Apple and Mozilla have all announced that their respective browsers will stop accepting SHA-1 SSL certificates by 2017.
- On February 23, 2017 CWI Amsterdam and Google announced they had performed a collision attack against SHA-1, publishing two dissimilar PDF files which produce the same SHA-1 hash.

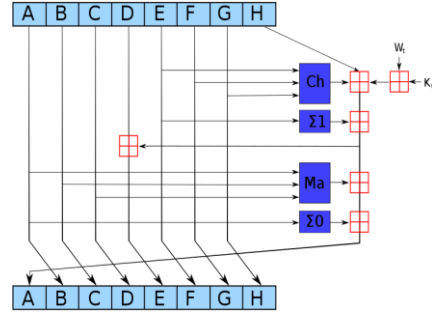
SHA-1 OPERATION

- Structure very similar to MD4 and MD5.
  - Secret design criteria
- Stronger than MD5 because of longer message digest
- Slower than MD5 because of more rounds



SHA-2

- SHA-2 similar to SHA-1, but with different input-output length.
- The algorithms are collectively known as SHA-2, named after their digest lengths: SHA-256, SHA-384, and SHA-512.
- There is no known attack against SHA-2.

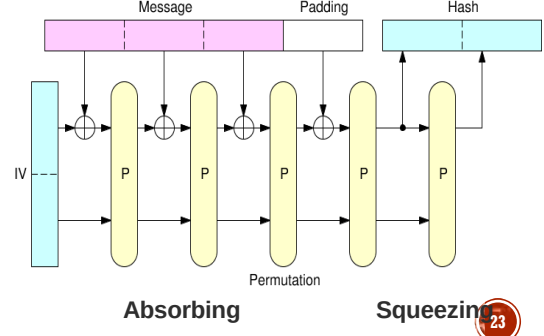


Ch(E, F, G) = (E ∧F) ⊕ (¬ E ∧G)
Ma (A, B, C) = (A ∧B) ⊕(A ∧C) ⊕ (B ∧C)
Σ0 (A) = (A ⋙ 2) ⊕ (A ⋙ 13) ⊕ (A ⋙ 22)
Σ1 (E) = (E ⋙ 6) ⊕ (E ⋙ 11) ⊕ (E ⋙ 25)

SHA-3

- SHA-3 is the latest member of the Secure Hash Algorithm family of standards, released by NIST on 2015 as FIPS 202.
- In 2006 NIST started to organize the NIST hash function competition to create a new hash standard, SHA-3.
  - On October 2, 2012, Keccak was selected as the winner of the competition.

Sponge construction of SHA-3:



RIPEMD-160

- RIPEMD (RACE Integrity Primitives Evaluation Message Digest) is a family of cryptographic hash functions developed in Katholieke Universiteit Leuven, and first published in 1996.

- RIPEMD-160 is an improved, 160-bit version of the original RIPEMD, and the most common version in the family.

- RIPEMD-160 was designed in the open academic community, in contrast to the NSA- designed SHA-1 and SHA-2 algorithms.

- There is no known attack against RIPEMD-160.