EWD – Security part

Cryptography Essentials

Objectives

- Gain understanding of three main ingredients of most security protocols & products
  - **Symmetric encryption**
  - **Public-key cryptography**
  - **Cryptographic hash functions**

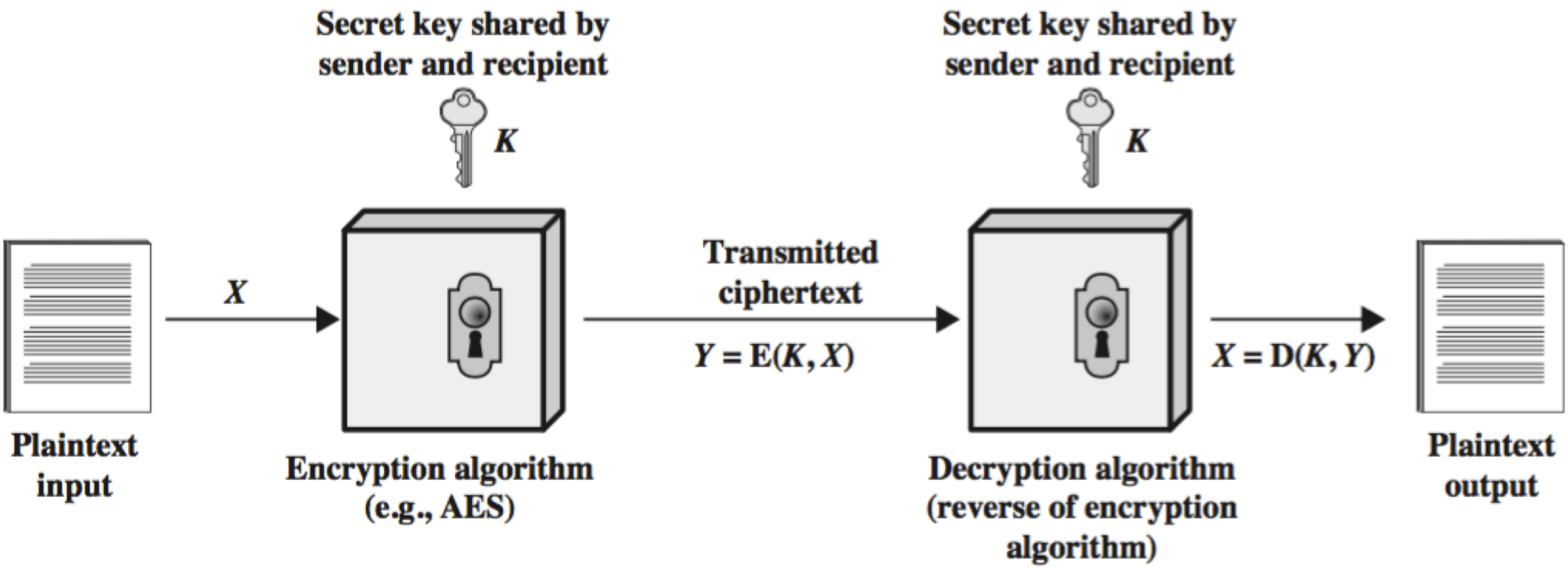
- ... and their application to
  - **Data confidentiality**
  - **Data integrity**
  - **Authentication**

Encryption

Some jargon

Cryptography:	Science of “secret writing”
Plaintext:	Original message
Ciphertext:	Transformed message
Encryption:	plaintext -> ciphertext process
Decryption:	ciphertext -> plaintext process
Cipher:	“Secret method of writing” (i.e. algorithm)
Key:	Some critical information used by the cipher, known only to sender and/or receiver
Cryptanalysis:	Attempting to discover plaintext or key or both

Symmetric Encryption



- Sender and receiver use <u>same</u> key (shared secret)
- Fast
- But how to share secret keys?
  - "chicken-and-egg" problem

Public-key Cryptography

- Major limitations of Symmetric Encryption:
  - Key distribution problem
  - Not suitable for authentication: receiver can forge message & claim it came from sender

- Addressed by Public-key Cryptography

- Public-key methods based on sender and receiver using <u>different</u> keys

Public-key Cryptography

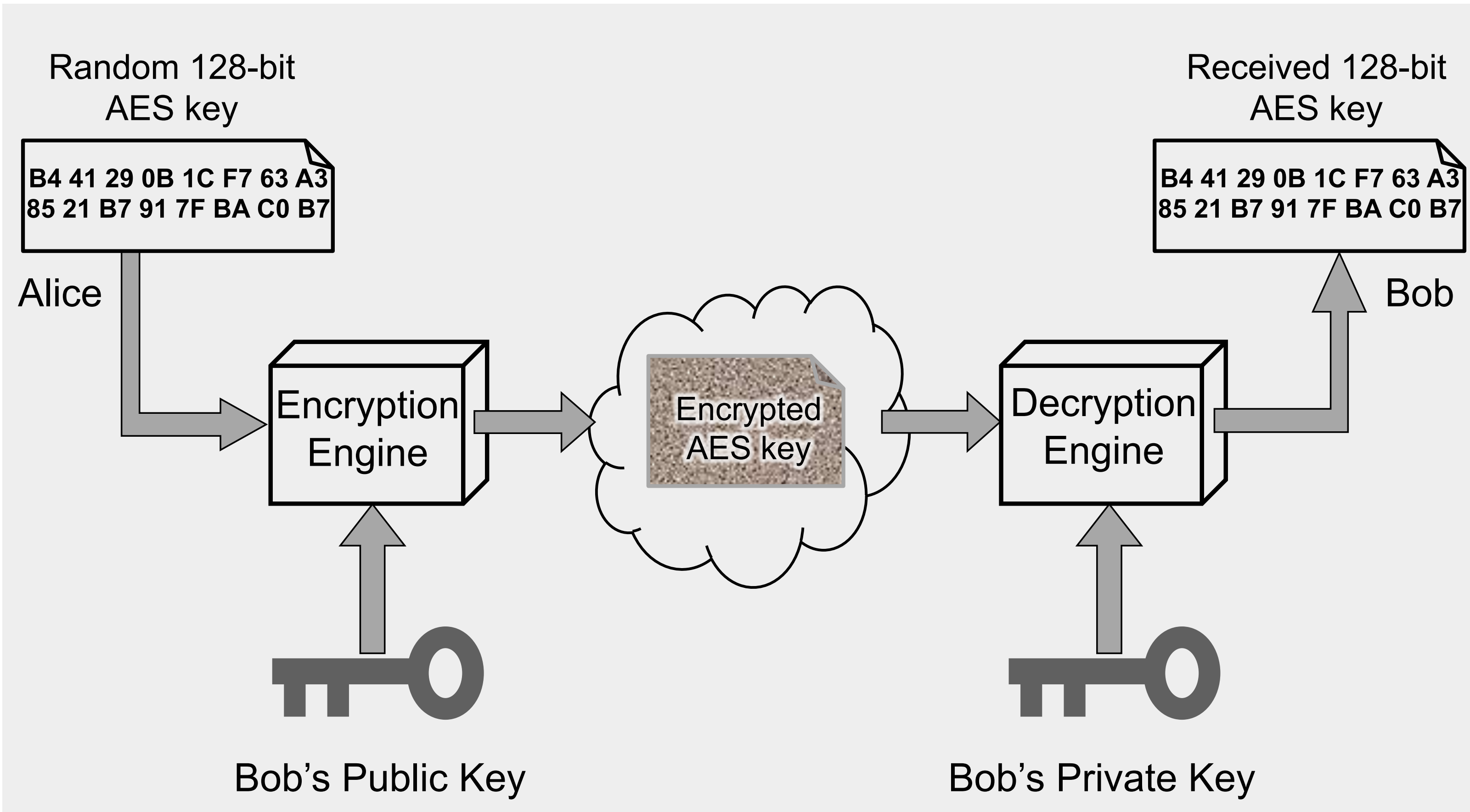
- Each party has two keys:
  - a **public key**, known potentially to anybody, used to **encrypt messages**, and **verify signatures**
  - a **private key**, known only to its owner, used to **decrypt messages**, and **create signatures**

- Complements rather than replaces symmetric cryptography
  - As it's much slower, it can't be used for large scale data encryption
  - Instead used for exchanging keys for symmetric ciphers
  - Also used for signing message digests (hashes) for authentication – more on this later

Symmetric Key Exchange using Public-key Crypto

- Alice wishes to communicate with Bob using a shared secret key
  - First, she generates a random key (e.g. 128 bit key for AES-128)
  - Then she uses Bob's public key to encrypt this shared secret key and she sends it to Bob
  - Bob opens the message with his private key and retrieves the shared secret key
  - Alice & Bob then use the shared secret key to communicate with each other

Symmetric Key Exchange using Public-key Crypto



Once exchanged, Alice & Bob then use the shared secret key to communicate using symmetric encryption

Limitations of Public-key Cryptography

1. Processing speed
   - Calculations required for public-key algorithms (mainly multiplications) much slower than those of conventional algorithms (permutations & XORs)
   - Thus public-key methods not suitable for general-purpose encryption/decryption
   - Instead often just use public-key method to exchange session (secret) key at beginning of session & use session key thereafter

Limitations of Public-key Cryptography

2. Authenticity of public keys (MITM attack)
   - Bob’s public key is in the public domain and only Bob has the corresponding private key
   - What happens though if an eavesdropper (Eve) generates another key pair and advertises the public key produced as belonging to Bob?
   - People then may send messages to Bob using the wrong public key, for which Eve has the corresponding private key.

$\Rightarrow$ *Need to be able to **trust** that a public key belongs to whom it is reputed to belong.*

Cryptographic strength & cryptanalysis

Kerckhoff’s principle

- Security should depend on the secrecy of the **key**, not the secrecy of the algorithm

- Attempts to keep algorithms secret are usually ineffective (they leak out)
- … and counterproductive as review by the wider crypto community allows weaknesses to be found early on, before deployment.

Cryptanalysis

- Cryptanalysis is the process of trying to find the plaintext or key
- Two main approaches
  – Brute Force
    - try all possible keys
  – Exploit weaknesses in the algorithm or lack of randomness in the key

Cryptanalysis: Brute Force Attack

- Try all possible keys until code is broken
- On average, need to try half of all possible keys
- Infeasible if key length is sufficiently long

Key size (bits)	No. of keys	Time required at 1 encryption per *μs*	Time required at 10^6 encryptions per *μs*
32	4.3×10^9	36 minutes	2 milliseconds
56	7.2×10^{16}	1142 years	10 hours
128	3.4×10^{38}	5.4×10^{24} years	5.4×10^{18} years
168	3.7×10^{50}	5.9×10^{36} years	5.9×10^{30} years

Age of universe: ~ 10^{10} years

Note: DES has a 56 bit key; AES key has 128+ bits

Data Integrity

Data Integrity

- Integrity refers to assurance of non-alteration
- Many systems and components have checksums or cyclic redundancy checks that are designed to detect *accidental* errors, etc.
  - For example, a credit card number contains a digit that is used to verify the others
- But such schemes are not sufficient to prevent *deliberate* modifications

Cryptographic Hash Functions

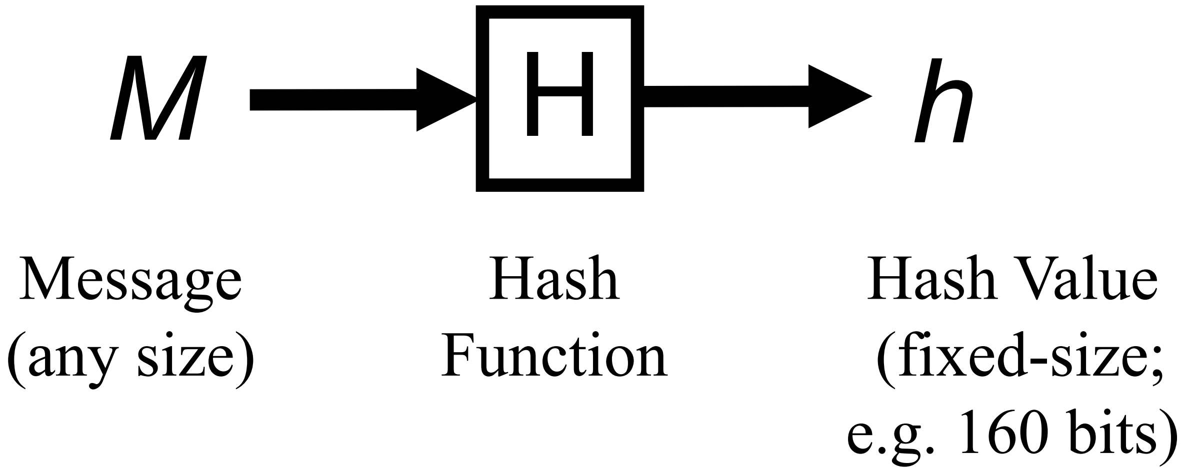
- Used to provide integrity of a message
- Purpose is to produce a fixed-size *hash-value:*

$$h = H(M)$$

where h is the hash value
H is the hash function
M is the message

- Any change in M, however small, should produce a different h-value

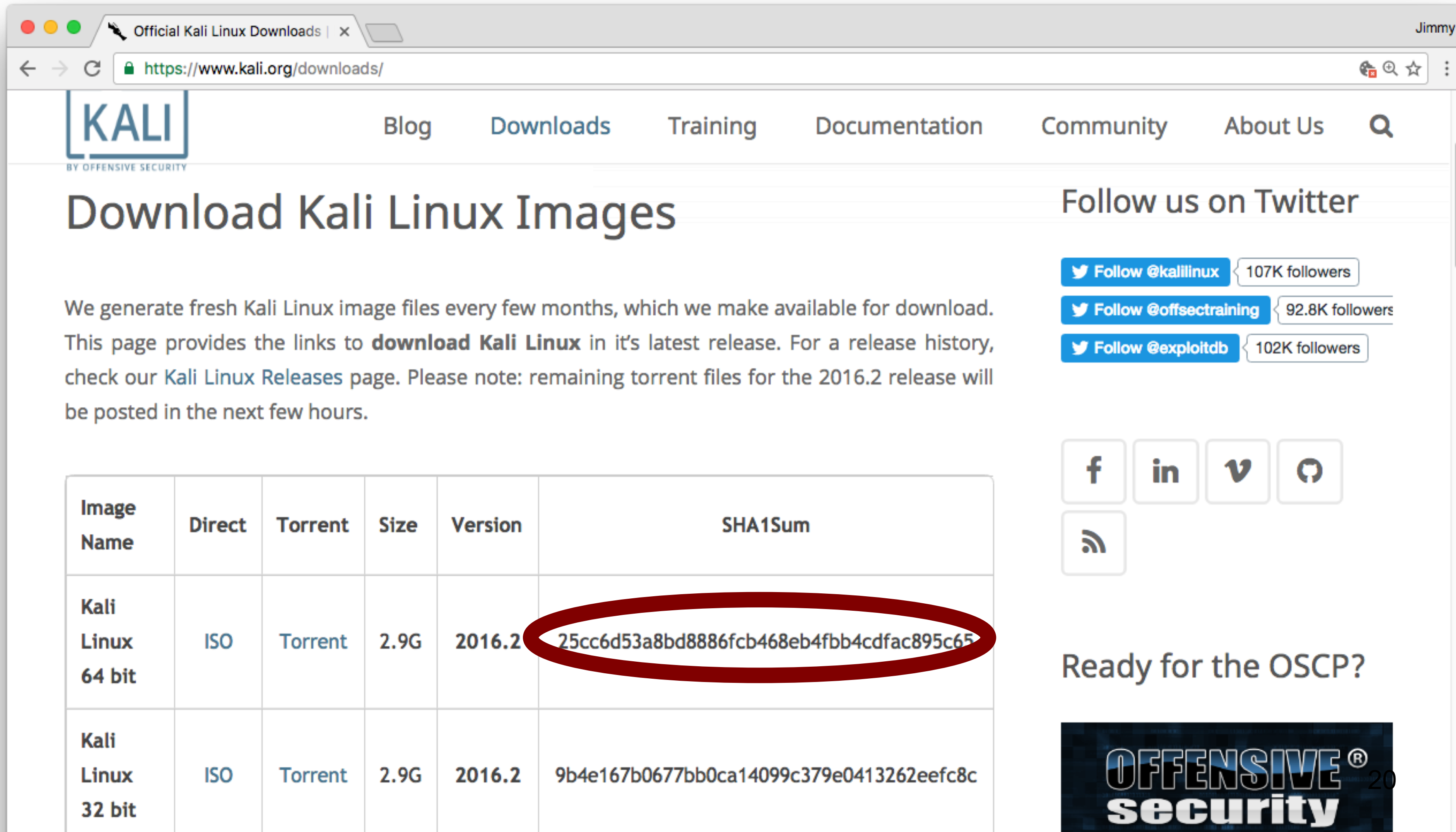
Cryptographic Hash Functions



- Note that a hash function is a many-to-one function. Potentially many messages can have the same hash, but finding these should be very difficult

Applications of Hash Functions

- As cryptographic checksum
  - e.g. to verify software downloads

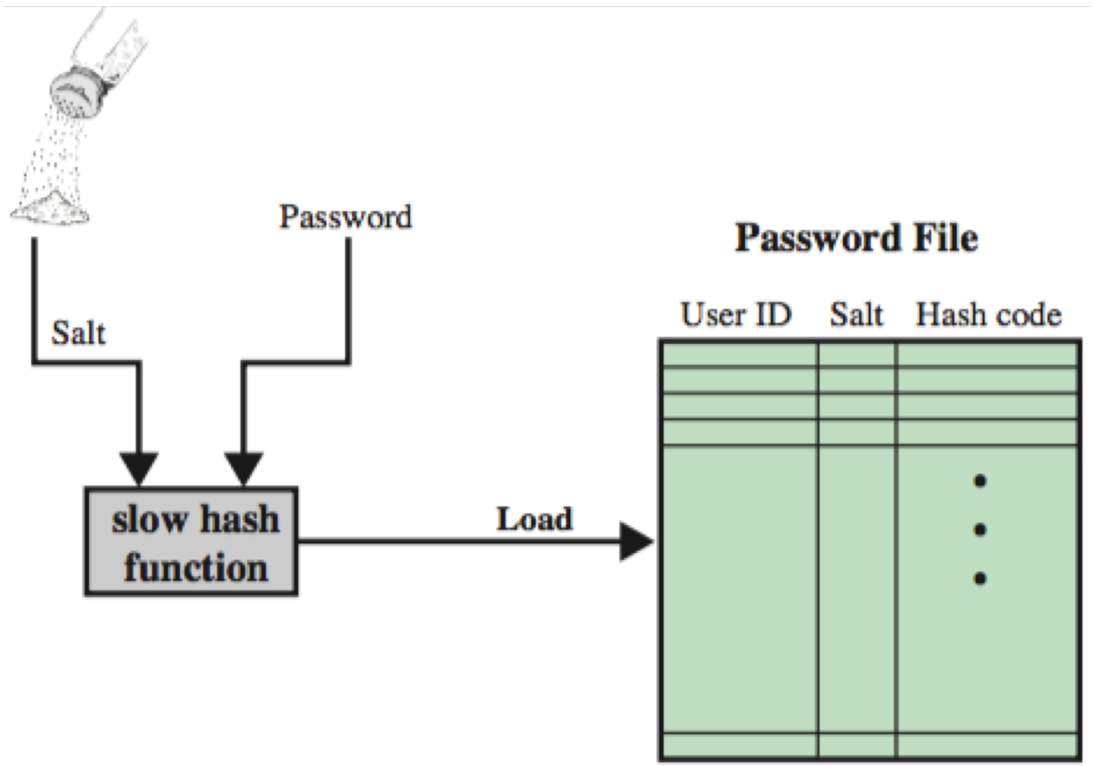


Applications of Hash Functions

- Authentication
  - It usually makes more sense to sign the hash of a message (with a private key) than to sign the original message
  - This is done with digital certificates and many other authentication schemes

Applications of Hash Functions

- Password storage
  - Store only the hash of password (+ *salt*)



Cryptanalysis: Breaking hash functions

- Strength depends on the length, n, in bits of the hash value

- Brute force attacks require time proportional to:
  - one-way property: 2^n
  - weak collisions property: 2^n
  - strong collisions property: $2^{n/2}$
    - This means the ability to find ***any*** two messages that hash to the same value:

Main Hash Algorithms

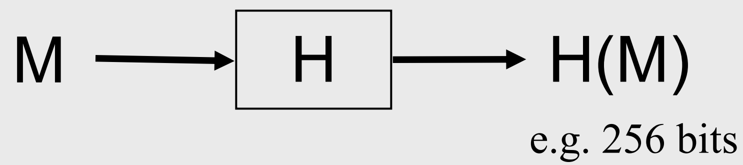
- MD5
  - Produces 128-bit hash value (i.e. 64-bit security)
  - Collisions found (2004)
  - No longer recommended for use
- SHA-1
  - Produces 160-bit hash value (80-bit security)
  - Collisions found (2017)
  - No longer recommended for use
- SHA-2
  - Set of 4 hash functions with different size outputs
  - SHA-224, SHA-256, SHA-384, SHA-512
  - Considered safe to use
    - (though new SHA-3 has been established due to concerns over structural similarities with SHA-1)

Authentication

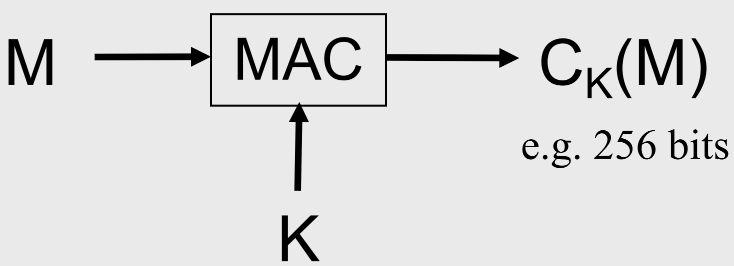
Message Authentication Code (MAC)

- Very similar to Hash Function
- Difference is the use of a **key**

Hash Function:

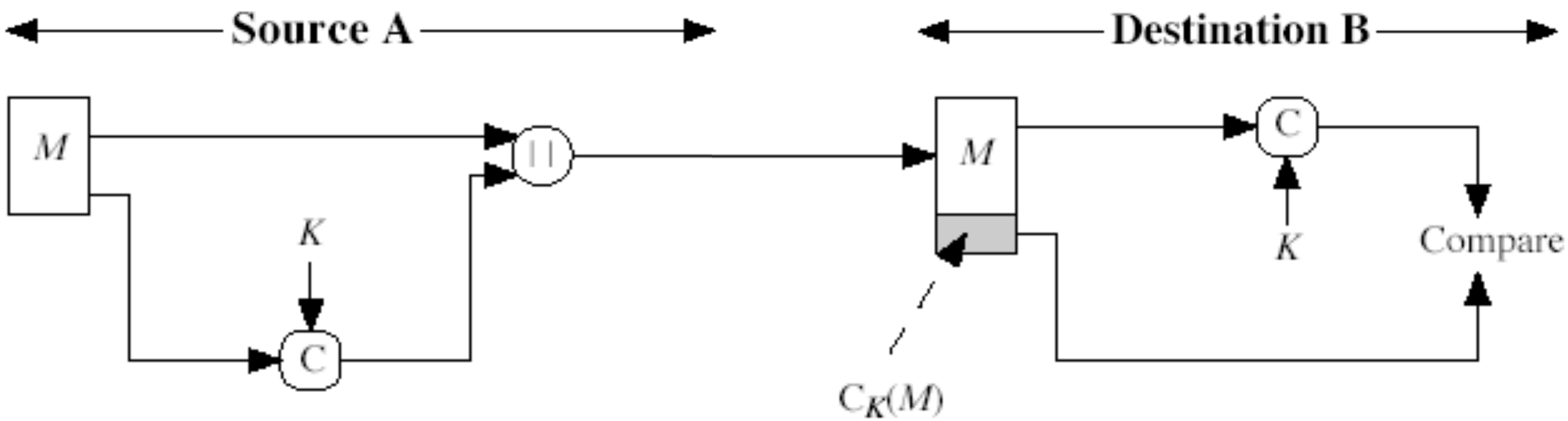


MAC:



Basic use of MAC for authentication

- Sender and recevier need to have shared secret



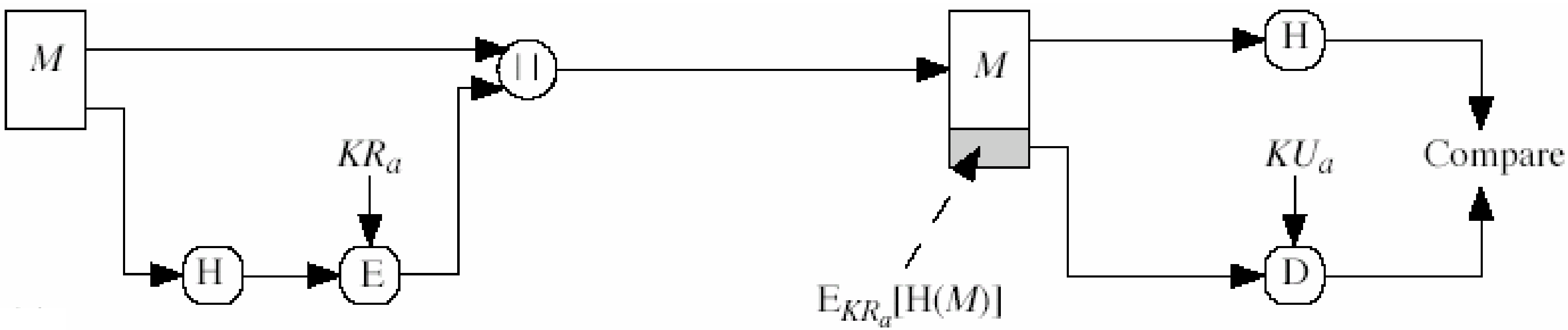
<u>Note:</u> The symbol with two vertical bars **||** means *concatenate*; i.e. join inputs together

Digital Signatures: signing the hash

- Digital signature created by adding a small authentication block to a message
- Often done by taking the hash of the message and **encrypt the hash** with the **sender's private key**
- The result is a very compact signature (relative to message size)
- And is just as secure as encrypting the entire message with the sender's private key
  - assuming that a secure hash function is used

Typical Use of Hash Function with Digital Signature

- Just sign the hash
  - much more efficient than signing full message



KR_a: Sender's Private Key

KU_a: Sender's Public Key

<u>Note</u>: The || symbol means *concatenate*; i.e. join inputs together