

Theory of Blockchain



Session 2:

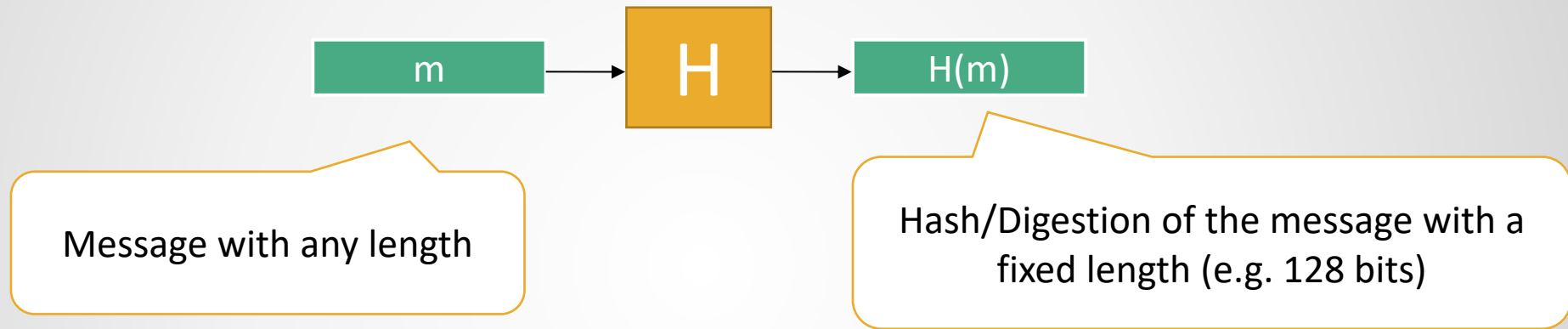
Symmetric Cryptography

Module 3 – Hash Functions and
their Applications

Classification of Ciphers

Forget about this classification for a while. Hash functions are not encryption algorithms. They are keyless, but can sometimes be constructed by using block ciphers.

Hash Function



Examples: SHA-1 ,MD5 ...

Hash functions are keyless. A hash function takes any message with any length and turns it into a fixed-length output. This conversion is fast, but calculating the reverse, is computationally hard.

Examples

MD4("The quick brown fox jumps over the lazy **d**og")
= 1bee69a46ba811185c194762abaeae90

MD4("The quick brown fox jumps over the lazy **c**og")
= b86e130ce7028da59e672d56ad0113df

MD4 ("a") = bde52cb31de33e46245e05fbdbd6fb24

MD4 ("abc") = a448017aaf21d8525fc10ae87aa6729d

MD4 ("message digest") = d9130a8164549fe818874806e1c7014b

MD4 ("abcdefghijklmnopqrstuvwxyz")
= d79e1c308aa5bbcddea8ed63df412da9

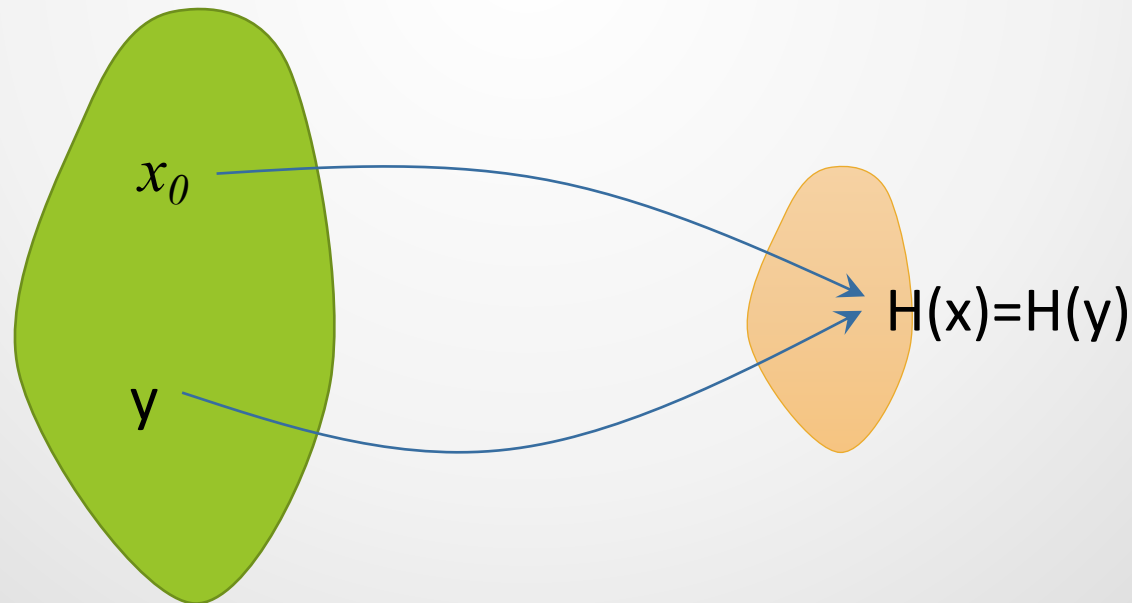
Properties of a Good Hash Function

1. H can be applied to a block of data of any size.
2. H produces a fixed-length output.
3. $H(x)$ is relatively easy to compute for any given x .
4. For any given code h , it is computationally infeasible to find x such that $H(x)=h$.
(A hash function with this property is referred to as **one-way** or **preimage resistant**)

Properties of a Good Hash Function

5. For any given block x_0 , it is computationally infeasible to find $y \neq x_0$ with $H(y) = H(x_0)$.

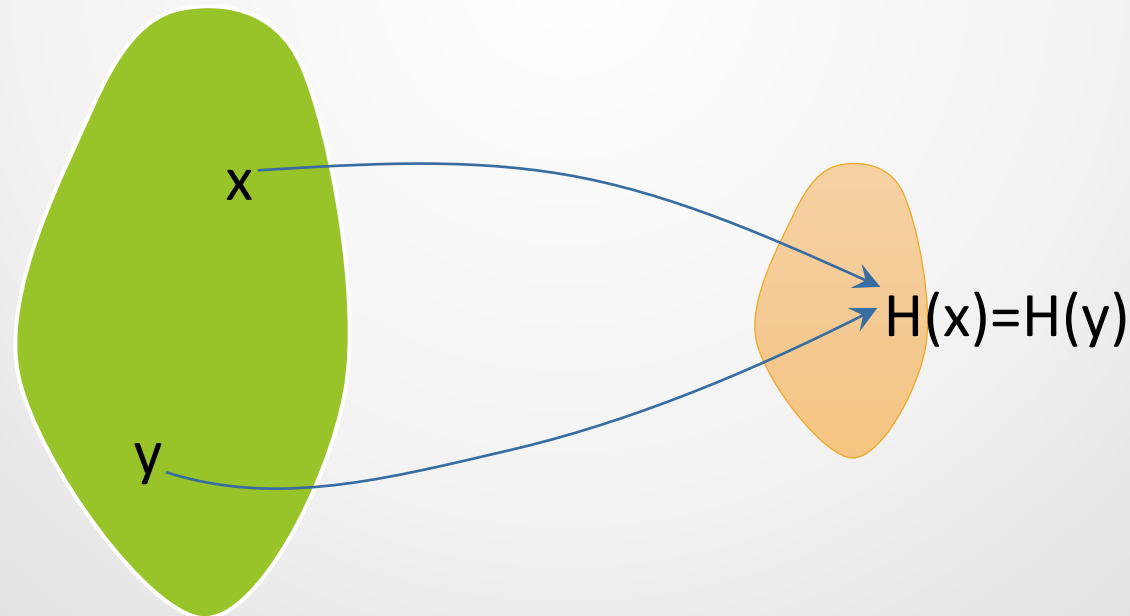
(A hash function with this property is called **second preimage resistant**. This is sometimes referred to as **weak collision resistant**)



Properties of a Good Hash Function

6. It is computationally infeasible to find any pair (x, y) such that $H(x)=H(y)$.

(A hash function with this property is referred to as **collision resistant**. This is sometimes referred to as **strong collision resistant**)



Hash Function Applications

1- OS

2- Message Authentication Codes

3- Blockchain

4- Hash Chains

.

.

.

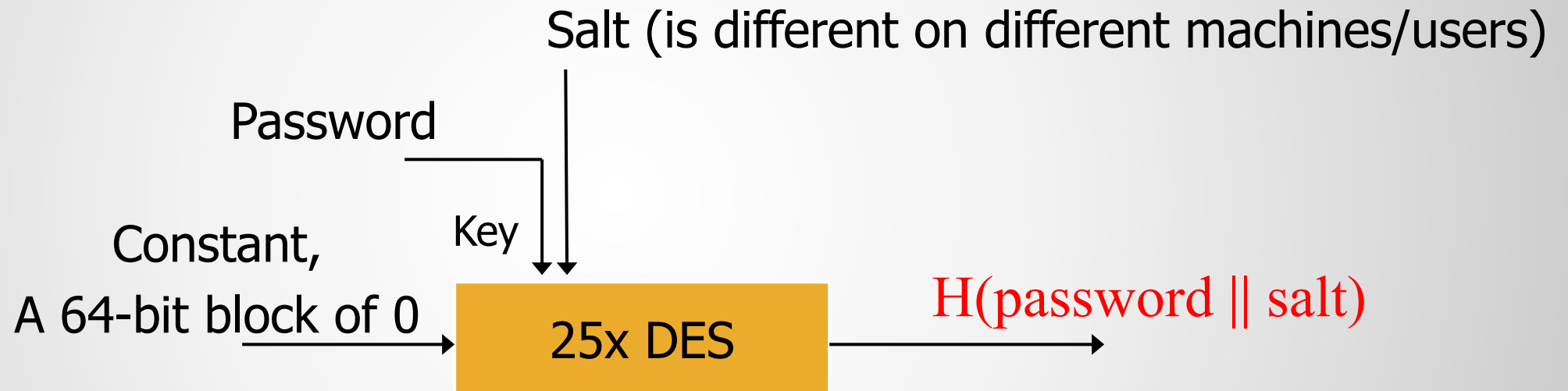
Hash Application in OS

- User password stored as $H(\text{password})$
- When user enters password
 - System computes $H(\text{password})$
 - Compares with entry in password file
- No passwords stored on disk as plain-text

Example: **Unix/Linux password system**

- Hash function is 25xDES
 - 25 rounds of DES-variant encryptions
- Any user can try “dictionary attack”
- **“Salt”** makes dictionary attack harder

Unix Password Storage System



When password is set, salt is chosen randomly. 12-bit salt makes dictionary attacks more costly (by a factor of 2^{12}).

/etc/shadow on Centos 7

File Edit View Search Terminal Help

```
polkitd:!!:16842:::::::
sssd:!!:16842:::::::
abrt:!!:16842:::::::
tss:!!:16842:::::::
unbound:!!:16842:::::::
usbmuxd:!!:16842:::::::
colord:!!:16842:::::::
amandabackup:!!:16842:::::::
saslauth:!!:16842:::::::
libstoragemgmt:!!:16842:::::::
geoclue:!!:16842:::::::
rpc:!!:16842:0:99999:7:::
setroubleshoot:!!:16842:::::::
rtkit:!!:16842:::::::
qemu:!!:16842:::::::
ntp:!!:16842:::::::
rpcuser:!!:16842:::::::
nfsnobody:!!:16842:::::::
radvd:!!:16842:::::::
chrony:!!:16842:::::::
pulse:!!:16842:::::::
gdm:!!:16842:::::::
gnome-initial-setup:!!:16842:::::::
avahi:!!:16842:::::::
postfix:!!:16842:::::::
sshd:!!:16842:::::::
topdump:!!:16842:::::::
mms:$6$da9xBn7q7ChREZDt$4popy3T8GqG3c75p9Lm/fsm0bUcW30UeIk7Wdzg1w8mARQdiA8611p0WfD3iu5snlTatucTz1yLaES
KBKbY4J1:16842:0:99999:7:::
[root@localhost ~]#
```

Recent Linux Distributions

- Salt is chosen for each user ID upon creation of that user.
- Salt is not limited to 12 bits anymore.
- Hash function is not DES-based anymore. SHA-512 and MD5 are more dominant.

Windows is no different!

- Hash of the passwords are stored in SAM file C:\windows\system32\config\sam

C:\WINDOWS\system32\cmd.exe

Microsoft Windows [Version 5.2.3790]
(C) Copyright 1985-2003 Microsoft Corp.

C:\Documents and Settings\Administrator>whoami
w2k3r2\administrator

C:\Documents and Settings\Administrator>Z:

Z:\>cd shared\tools

Z:\shared\tools>gsecdump.exe s
Administrator(current):500:aad3b435b51404eeaad3b435b51404ee:237599e85cf684a6785a
12acd2e24e5c:::
ADMINISTRATOR(current):1000:bae1b21f93c933c1f61c8a77c3b36172:3016281a6ad77d83b11c3b1611
d3b50b:::
db2admin(current):1030:3ae6ccce2a2a253f93e28745b8bf4ba6:35ccba9168b1d5ca6093b4b7
d56c619b:::
foobar(current):1031:87dceb9223be0e08fd8e74c8ceb3053a:33d807d89b36acdf2fab42a361
de0b91:::
Guest(current-disabled):501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73
c59d7e0c089c0:::
inquis(current):1024:0ac9a586623764e16591bb5472a3ad4a:89f411f435a93044e2e8aa4ced
fe0fba:::
IUSR_INQUIS-6J3CW98R(current):1000:4280a842d7bb7cbb3fcfbdba1ab0536b:43f8f8f69bf1
a36f4d979319ba562751:::
IWAM_INQUIS-6J3CW98R(current):1001:f3038e9ee720b8a6151ff516d245e5fc:3c6e3eeaab52
815a419aab14b020950e:::
mark(current):1032:2fd83b6f1038526a36a7a25cb8d074c0:792a16bb0d28da9be5289918aa40
e80e:::
postgres(current):1023:3ae6ccce2a2a253f93e28745b8bf4ba6:35ccba9168b1d5ca6093b4b7
d56c619b:::

Username:ID:LM hash:NT hash::

Old LM-hash vs the new NTLM hash

LM hash is used only for backward compatibility. It's been broken too.

LM hash is computed as follows:

1. The password is padded with NULL bytes to reach 14 characters.
2. The password is converted to all uppercase.
3. The password is split into two 7-byte (56-bit) keys.
4. Each key is used to encrypt a fixed string (that is "KGS!@#\$\$%") by using DES.
5. The two results from step 4 are concatenated and stored as the LM hash.

No Salt in Windows hashes

NT hash is simply a hash.

1. The password is hashed by using the MD4 algorithm and stored.

Message Authentication Code (MAC)

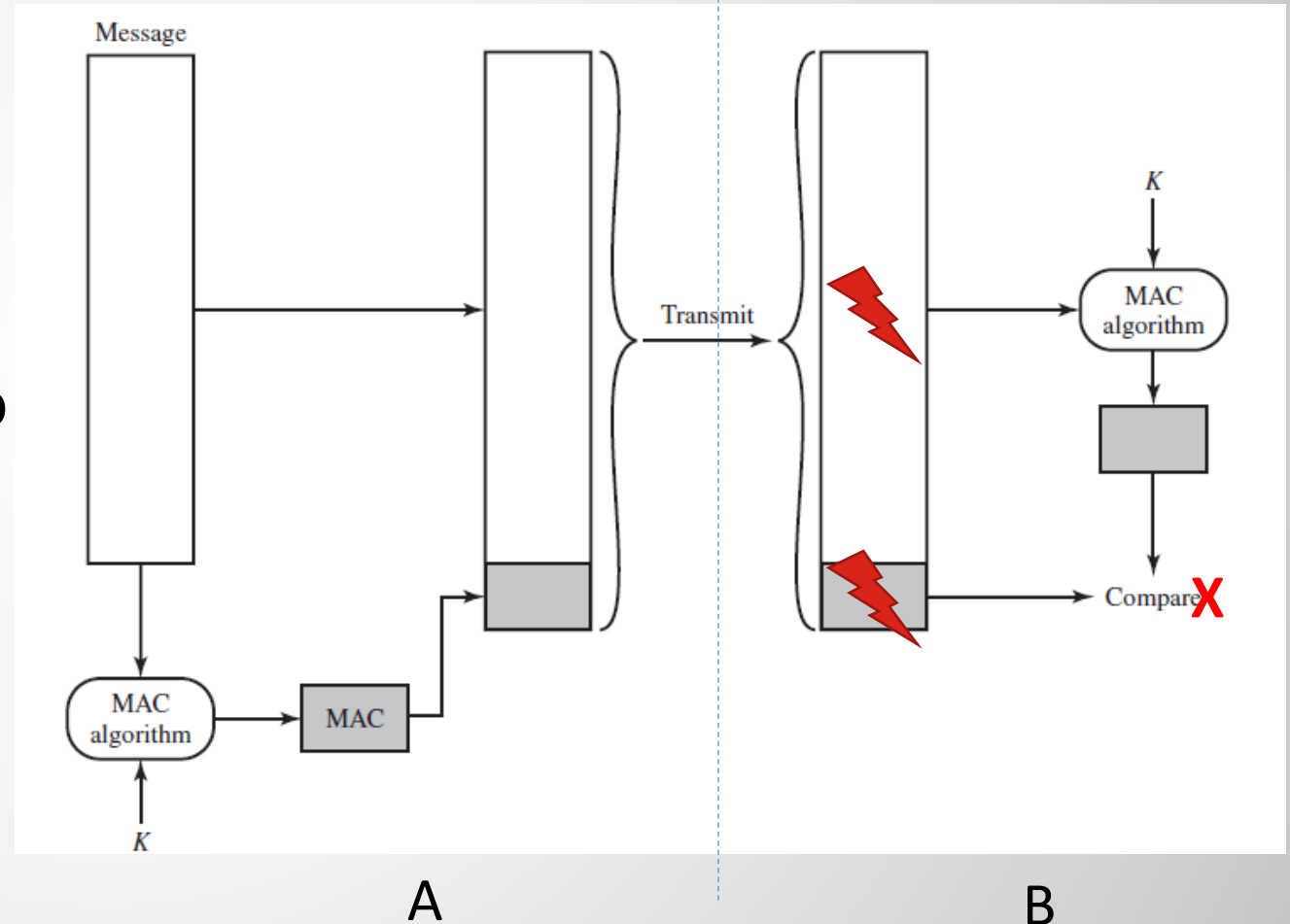
- MAC provides Authentication + Integrity services.
 - But not Confidentiality
- It is sometimes called Message Integrity Code (MIC) to differentiate it from Medium Access Control (MAC).

Message Authentication Code (MAC)

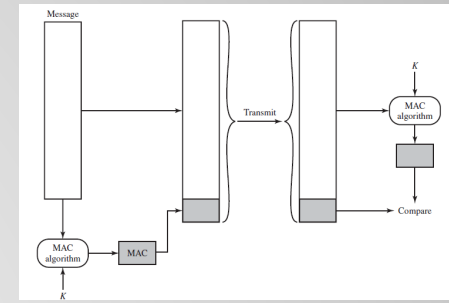
This technique assumes that two communicating parties, say A and B, share a common secret key K_{AB} .

When A has a message to send to B, it calculates the message authentication code as a function of the message and the key:

$$MAC_M = F(K_{AB}, M)$$



Points



- Since the key “ K_{AB} ” is only shared between A and B, if the MAC verifies, the receiver knows it the message must have come from the other party (This means message authentication).
- Verification of MAC implies that the message has not been tampered with. Because if either the message or the MAC part is touched, the receiver will know. Since no one (except A and B) knows the key, it is impossible to change both the message and the MAC so that they match at the receiver.

What is Inside the MAC Box?

It has two parts

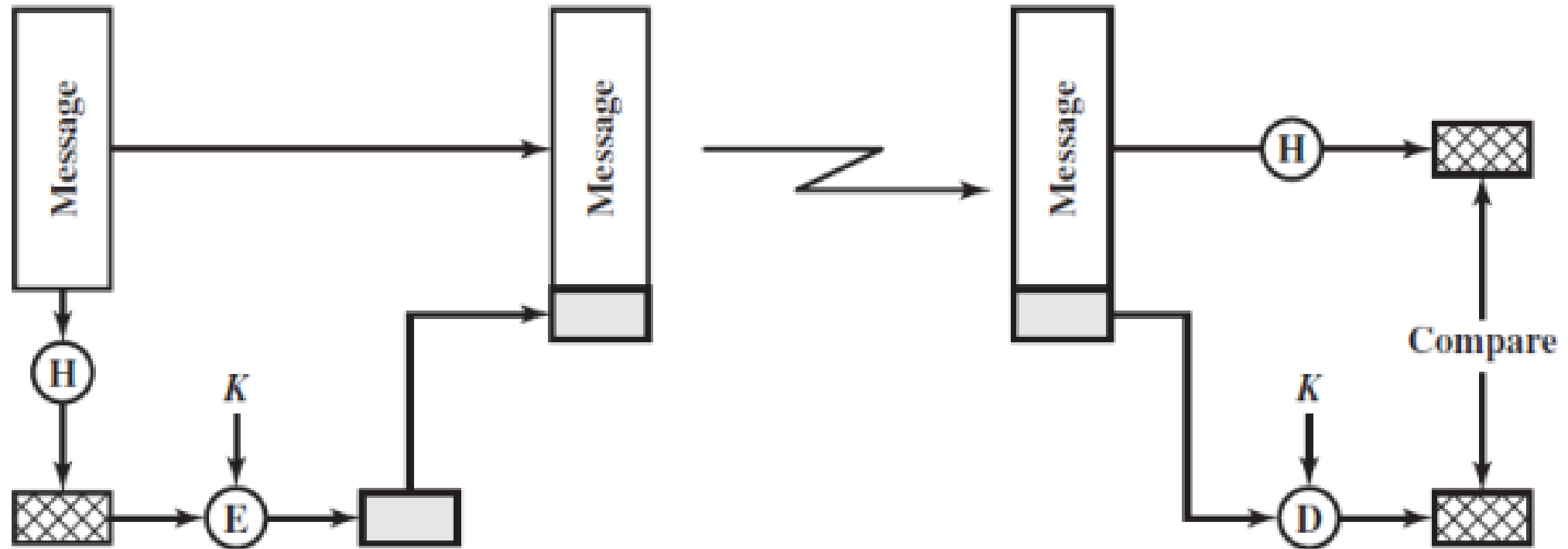
- 1- Message Digestion (i.e. hash)
- 2- Combination of the digestion with the secret key

Because applying the key on the whole message is costly.

It is possible to create a MAC that everyone can verify.

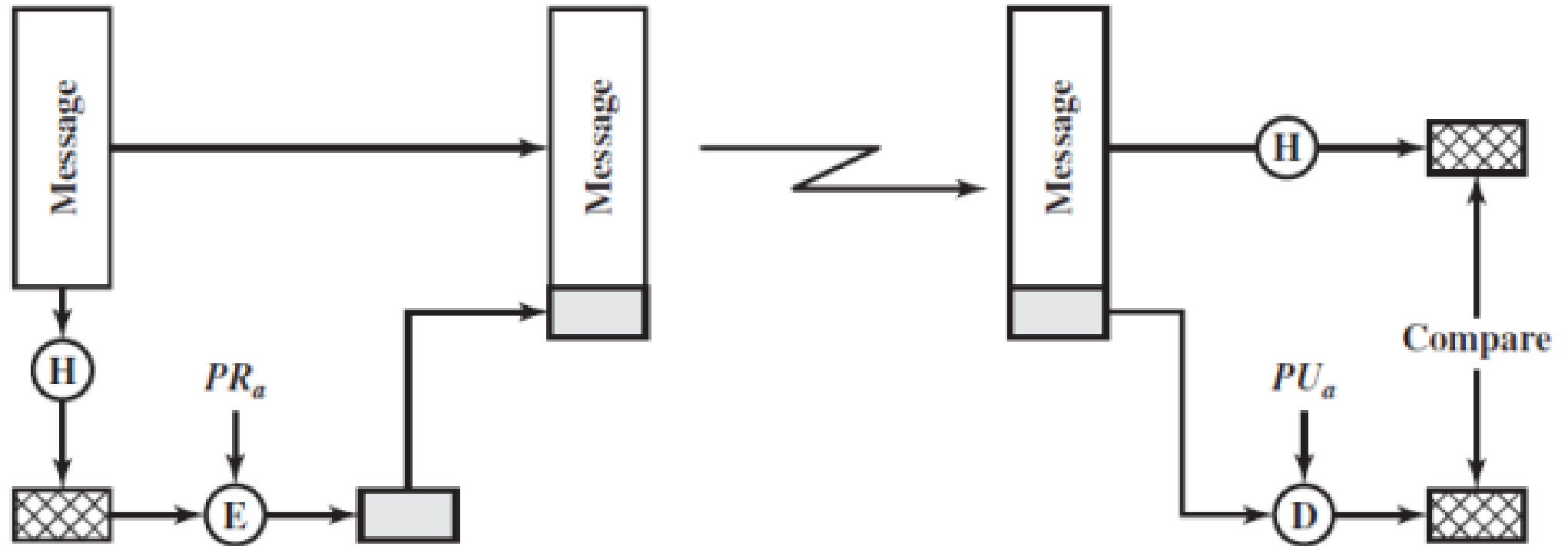
But if everyone has the key, security becomes meaningless. So the key for creation of MAC and verification of MAC must be different.

Making a MAC - Method 1



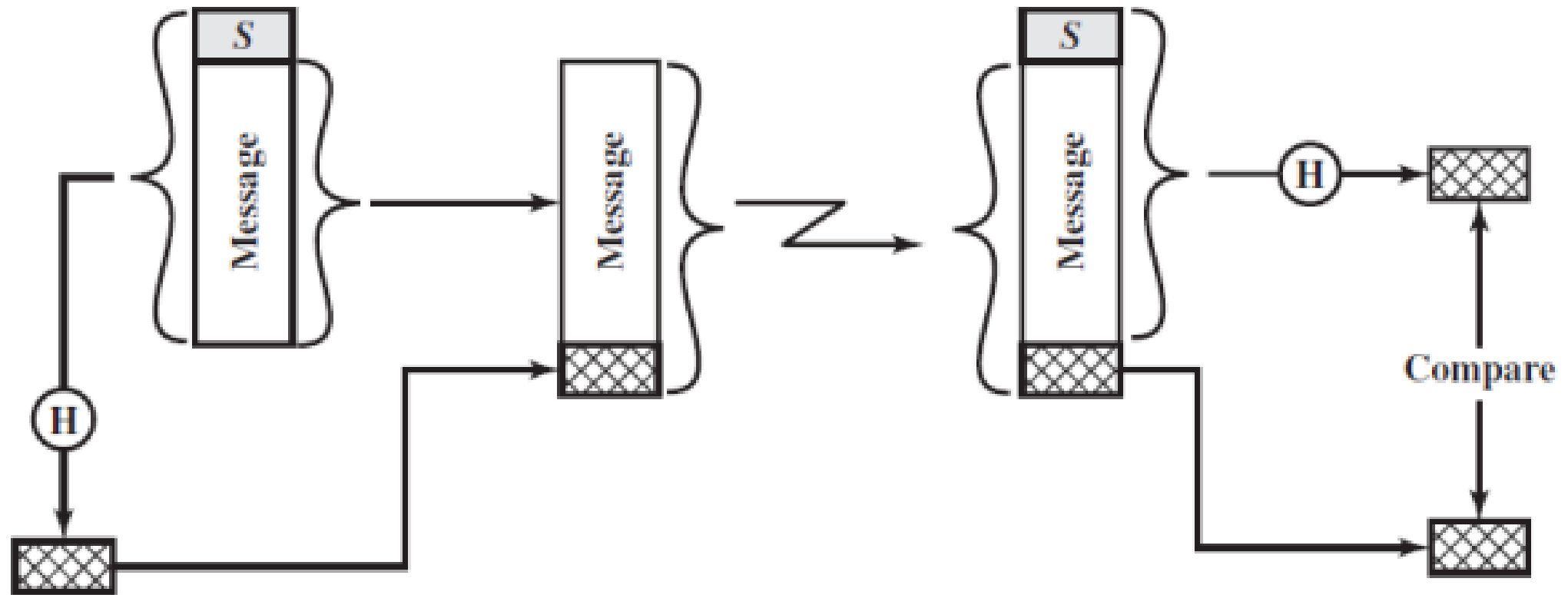
(a) Using conventional encryption

Making a MAC - Method 2 (Dig. Sig.)



(b) Using public-key encryption

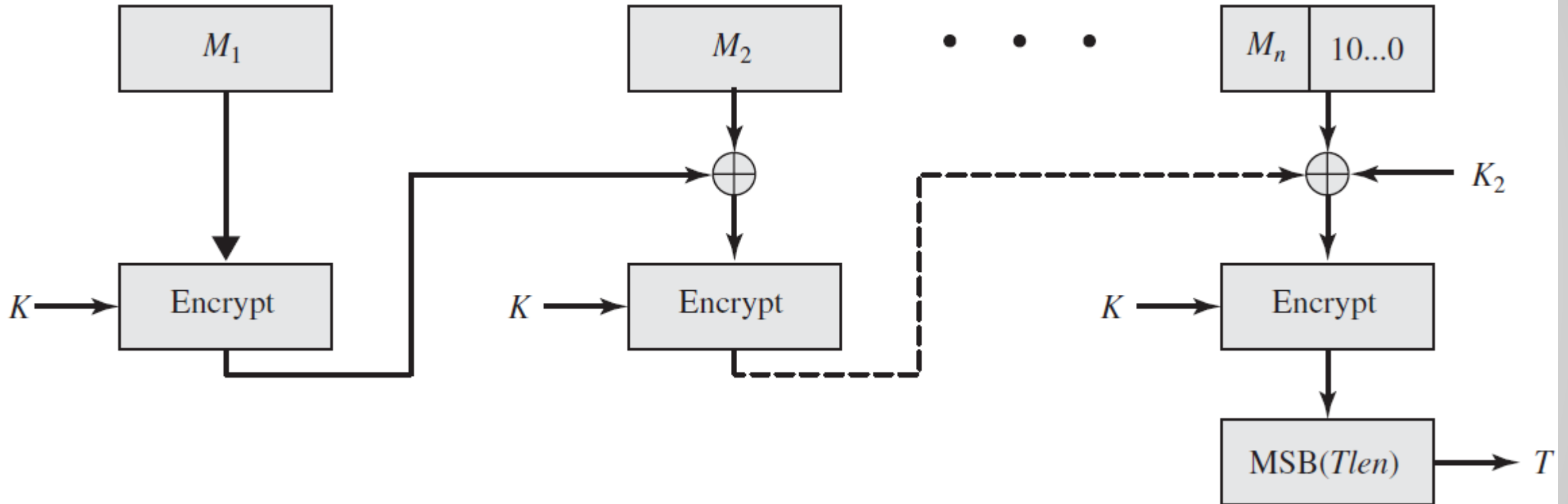
Making a MAC - Method 3



(c) Using secret value

Secret value can be the key

How to make a MAC using a block cipher



$M_1 \dots M_{n-1}$ are plain text pieces. K and K_2 are encryption and MAC keys, respectively. $\text{MSB}(Tlen)$ takes a desired length from output (from the most significant bit).

Famous Hash Functions

- MD4 , MD5

MD4 was proposed by Ronald Rivest in 1990. The digest length is 128 bits. It was broken later but it influenced the design of MD5 (RFC1321), SHA-1 and RIPEMD hashing algorithms.

- SHA (1), SHA-2 ... SHA-3

Bitcoin uses SHA-256 (SHA-2), and Ethereum uses Keccak-256 (the base of SHA-3).

SHA was developed by the National Institute of Standards and Technology (NIST) and published as a federal information processing standard (FIPS 180) in 1993. SHA used MDx designs.

What Comes Next ...

- We learned about hash functions and their applications.
- We learned how Message Authentication Codes are generated.
- In the next video, we introduce some asymmetric encryption algorithms and use the subjects you learned here to create digital signatures.

See you in the next video ...