Message Integrity and Message Authentication

Dr. B C Dhara

Department of Information Technology

Jadavpur University

Objectives

- Define message integrity & authentication
- Define criteria for a cryptographic hash function
- Define the Random Oracle Model and its role in evaluating the security of cryptographic hash functions
- Distinguish between an MDC and a MAC

MESSAGE INTEGRITY

- *The cryptography systems that we have studied so far provide secrecy, or confidentiality, but not integrity*

- *However, there are occasions where we may not even need secrecy but instead must have integrity*
  - Person ‘X’ write a will to distribute his property
    - Need not to be encrypted, no change is allowed, i.e., integrity is important

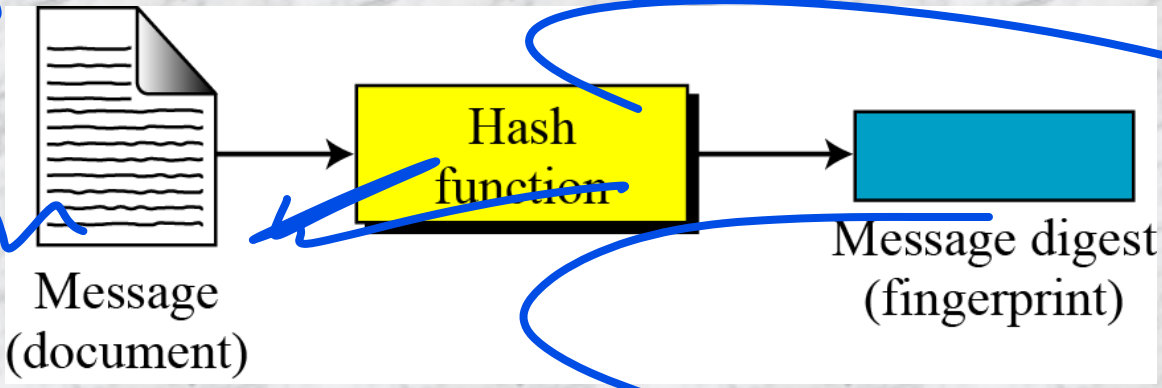
Document and Fingerprint

- *One way to preserve the integrity of a document is through the use of a fingerprint*
- *If Alice needs to be sure that the contents of her document will not be changed, she can put her fingerprint at the bottom of the document*

non – digital

Message and Message Digest

- *The electronic equivalent of the document and fingerprint pair is the message and digest pair*
  - *message will passes through a cryptographic hash function and creates a compressed message which is called as message digest*

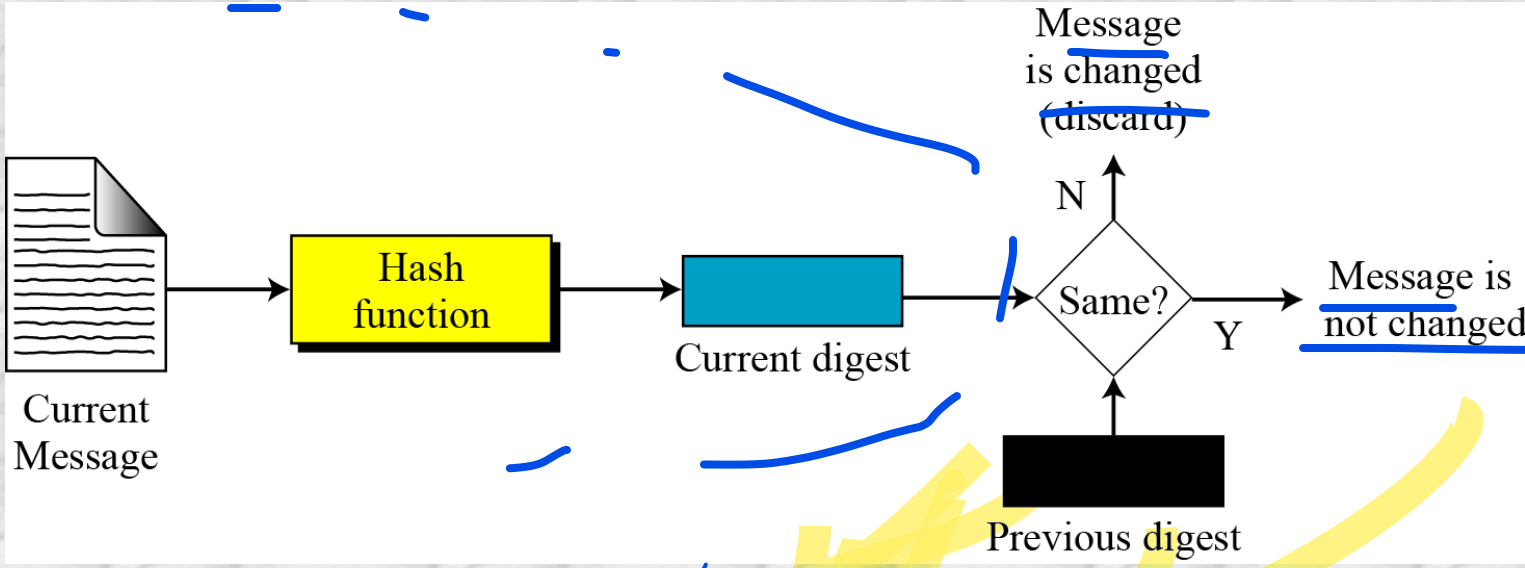


Difference: (message, digest) and (document, fingerprint)

- *The two pairs (document / fingerprint) and (message / message digest) are similar, with some differences*
- *The document and fingerprint are physically linked together*
- *The message and message digest can be unlinked and can sent separately*

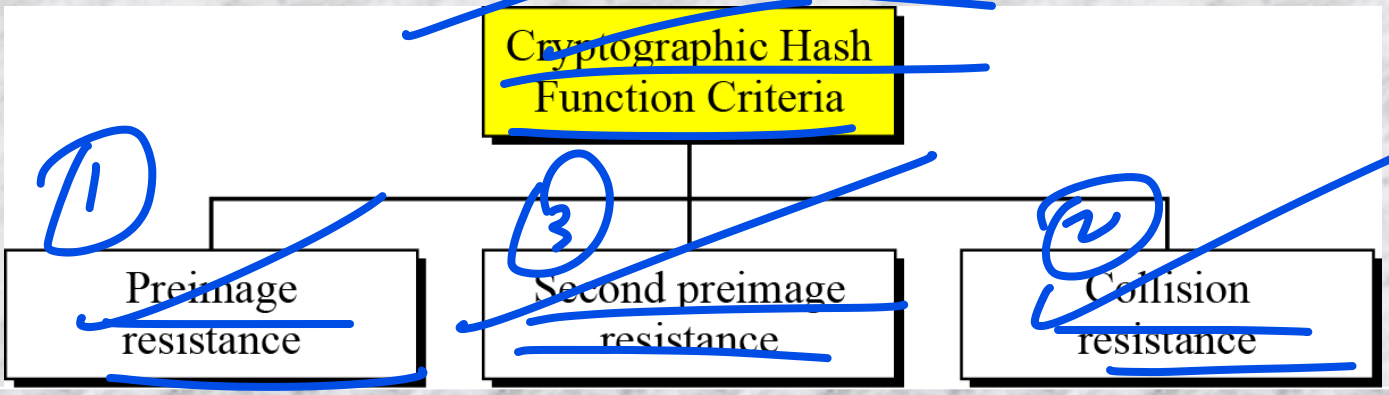
The message digest needs to be safe from change.

Checking Integrity



Cryptographic Hash Function Criteria

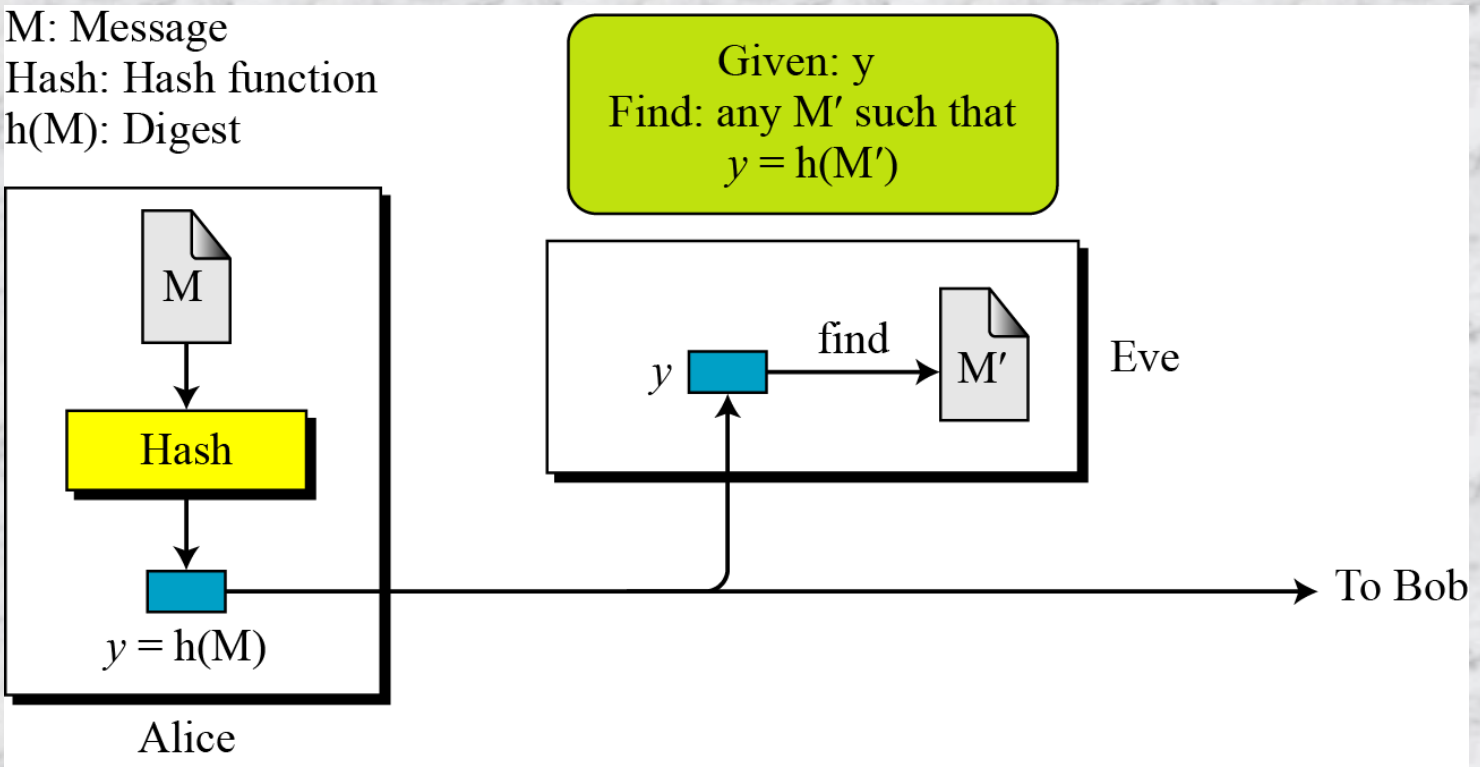
- *A cryptographic hash function must satisfy three criteria:*
  - *preimage resistance,*
  - *second preimage resistance, and*
  - *collision resistance*



Preimage Resistance

Preimage Attack

Given: y = h(M) **Find: M′ such that y = h(M′)**

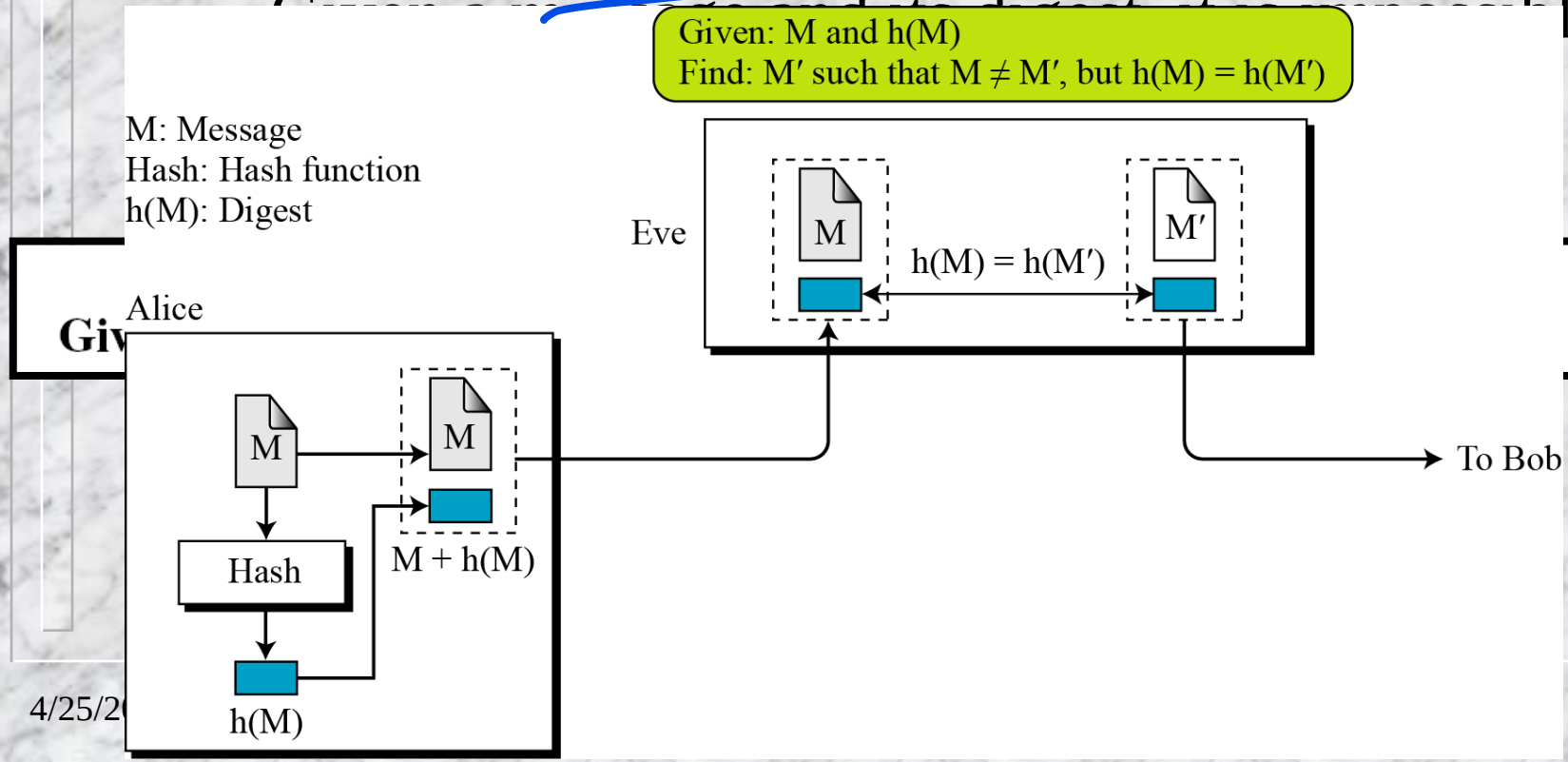


Preimage Resistance (contd…)

- Can we use a conventional lossless compression method such as *StuffIt* as a cryptographic hash function?
  - No, since original message can be retrieved from compressed image by decoding technique
- Can we use a checksum function as a cryptographic hash function?
  - No, since many messages can be constructed for which we have same checksum value

Second Preimage Resistance

- Second preimage resistance ensures that a message cannot be forged easily
  - Given a message and its digest, it is impossible



Collision Resistance

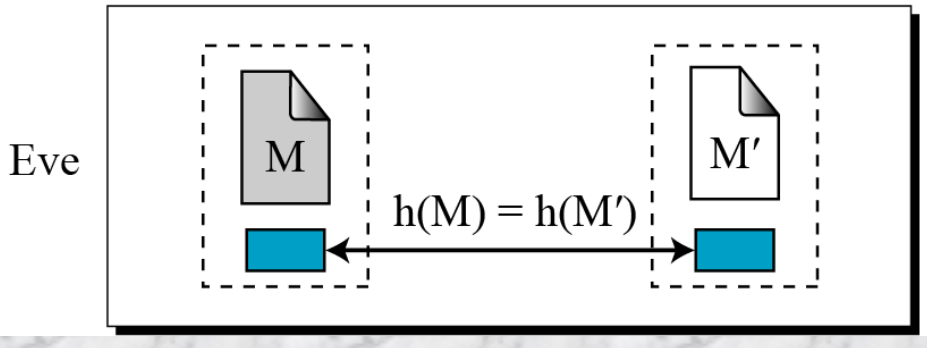
- We cannot find two messages that hash to same digest

Collision Attack

Given: none **Find: M′ ≠ M such that h(M) = h(M′)**

M: Message
Hash: Hash function
h(M): Digest

Find: M and M′ such that M ≠ M′, but h(M) = h(M′)



This attacker is easier than previous two attacks

RANDOM ORACLE MODEL

- The Random Oracle Model is a mathematical model for a hash function
- It behaves as
  - For a new message of any length it creates a fixed length digest (which is random string of 0s and 1s)
    - Also, records the (message, digest) pair
  - For a message, checks it exists or not; if yes, simple return the digest
  - Digest for a new message chosen independently from all previous digests
    - i.e., no formula or algorithm is used to compute digest

Example: RANDOM ORACLE MODEL

- Assume an oracle with a table and a fair coin. The table has two columns

Table 11.1 *Oracle table after issuing the first three digests*

Message	*Message Digest*
4523AB1352CDEF45126	13AB
723BAE38F2AB3457AC	02CA
AB45CD1048765412AAAB6662BE	A38B

- The message AB1234CD8765BDAD is given for digest calculation. The oracle checks its table

Example: RANDOM ORACLE MODEL (contd…)

Table 11.2 *Oracle table after issuing the fourth digest*

Message	*Message Digest*
4523AB1352CDEF45126	13AB
723BAE38F2AB3457AC	02CA
AB1234CD8765BDAD	DCB1
AB45CD1048765412AAAB6662BE	A38B

b. The message 4523AB1352CDEF45126 is given for digest calculation. The oracle checks its table and finds that there is a digest for this message in the table (first row). The oracle simply gives the corresponding digest (13AB).

Example: RANDOM ORACLE MODEL (contd…)

The oracle cannot use a formula or algorithm to create the digest

For example, if formula h(M) = M mod n

Suppose oracle has h(M1) and h(M2)

If message is M3 = M1 + M2, then h(M3)= [h(M1) + h(M2)] mod n

$$h(M_3) = (M_1 + M_2) \bmod n = M_1 \bmod n + M_2 \bmod n = [h(M_1) + h(M_2)] \bmod n$$

This violates the third requirement that each digest must be randomly chosen based on the message given to the oracle.

Message-digest relation

- Assume that the messages in a hash function are ***m*** bits long and the digests are only ***n*** (< ***m***) bits long
- Possible number of digests is $2^n = 16$, and the Possible number of messages is $2^m = 64$.
- At least one digest corresponds to 2^{m-n} messages

Birthday problems

- P1: what is the minimum number, k, of students in a classroom such that it is likely that at least one student has a predefined birthday with probability ≥ ½ ?
  - Let N possible birthdays
  - No one has predefined birthday, probability is $(1-1/N)^k$
  - at least one has predefined birthday, probability is $1- (1-1/N)^k$

- To analyze the Random Oracle Model, result of birthday problems are important
  - $1-x \approx e^{-x}$, for small x
  - $1- (1-1/N)^k \geq ½$

  ➔ $(1-1/N)^k \leq ½$

  ➔ $e^{-k/N} \leq ½$

  ➔ $e^{k/N} \geq 2$

  ➔ $k \geq \ln 2 * N$

Birthday problems (contd…)

- P2: what is the minimum number, k, of students in a classroom such that at least one student has same birthday as the student selected by the teacher with probability ≥ ½ ?

- Like previous one,

First student is selected by the professor and then there are k-1 students to full fill the selected value.

Hence, probability is $P= 1- (1-1/N)^{k-1}$

Therefore, $k \geq \ln 2 *N+1$

Birthday problems (contd…)

P3: what is the minimum number, k, of students in a classroom such that it likely that at least two students have same birthday with probability ≥ ½ ?

It can be easily proved that probability $P = 1-e^{-k^2/2N}$ and $k \geq (2 * \ln 2)^{1/2} * N^{1/2}$

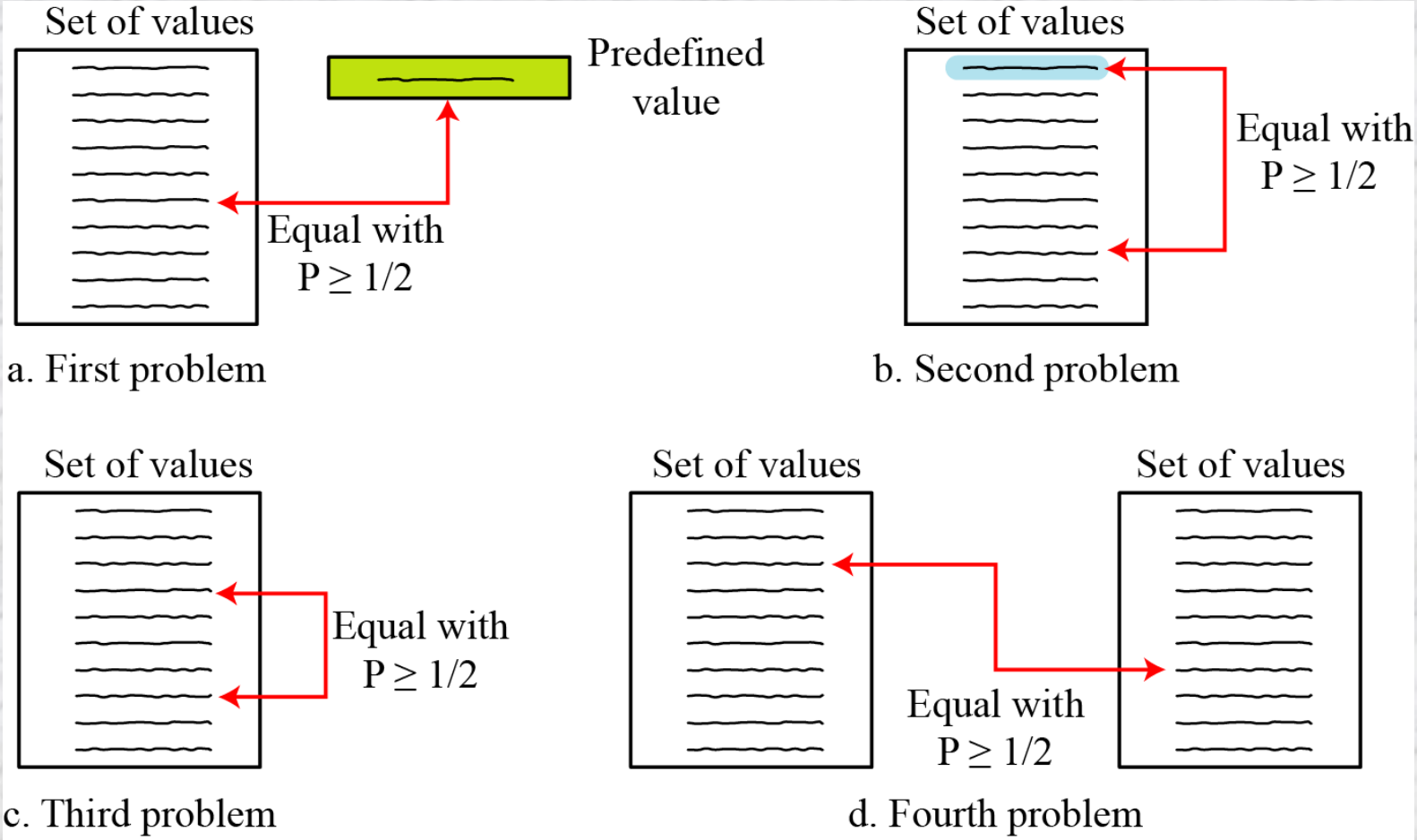
$k = \sqrt{N}$

Birthday problems (contd…)

P4: we have two classes, each with k students. What is the minimum value of k such that it likely that at least one student from first class has same birthday as one student from second class with probability ≥ ½ ?

It can be easily proved that probability $P = 1-e^{-k^2/N}$ and $k \geq (\ln 2)^{1/2} * N^{1/2}$

Birthday problems (contd…)



Summary of Solutions

Table 11.3 *Summarized solutions to four birthday problems*

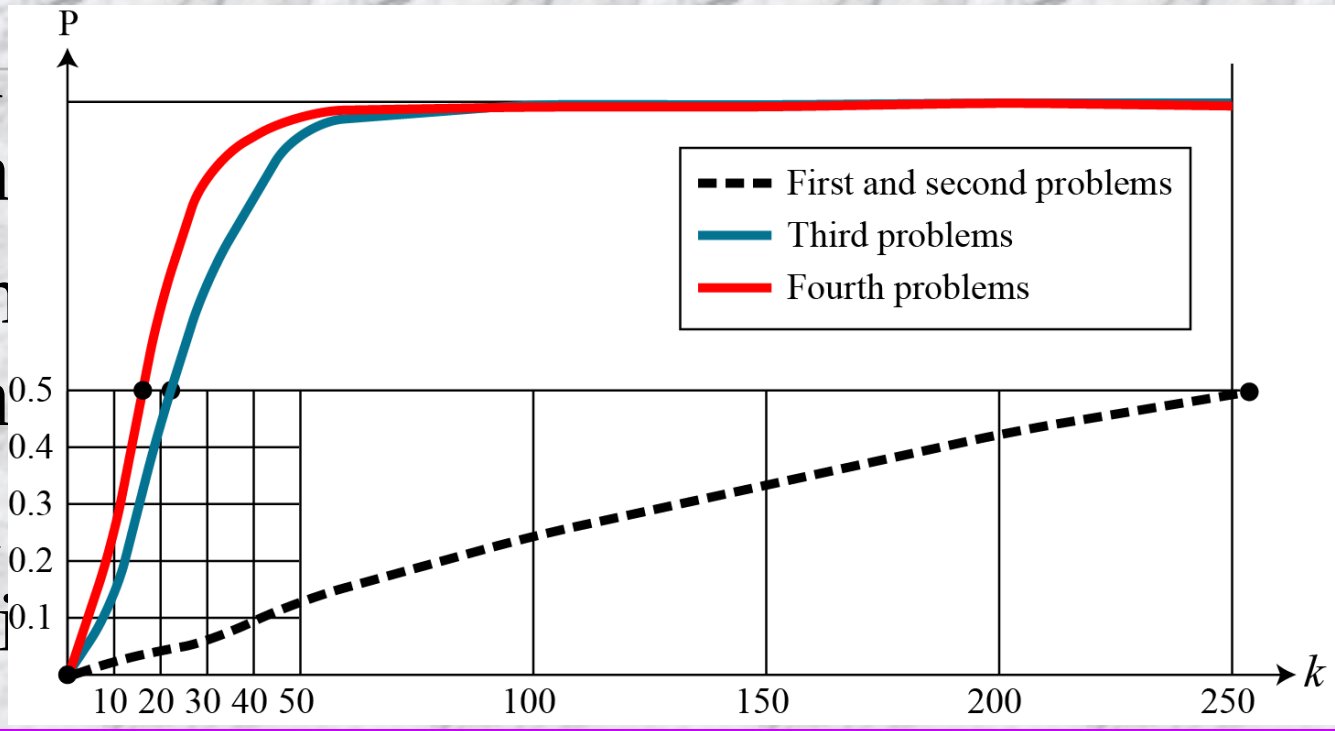
Problem	*Probability*	*General value for k*	*Value of k with P = 1/2*	*Number of students (N = 365)*
1	$P \approx 1 - e^{-k/N}$	$k \approx \ln[1/(1-P)] \times N$	$k \approx 0.69 \times N$	253
2	$P \approx 1 - e^{-(k-1)/N}$	$k \approx \ln[1/(1-P)] \times N + 1$	$k \approx 0.69 \times N + 1$	254
3	$P \approx 1 - e^{k(k-1)/2N}$	$k \approx \{2 \ln [1/(1-P)]\}^{1/2} \times N^{1/2}$	$k \approx 1.18 \times N^{1/2}$	23
4	$P \approx 1 - e^{-k^2/2N}$	$k \approx \{\ln [1/(1-P)]\}^{1/2} \times N^{1/2}$	$k \approx 0.83 \times N^{1/2}$	16

The third problem is known as birthday paradox

The result shows that is there are just 23 students in a classroom, it is likely (with $P \geq ½$) that two students have same birthday (ignoring year of birth)

Discussion on birthday problems

- It can … to the preim…
- Secon… preim…
- Third… collis…



It is obtained that value of k in first and second problems are proportional to N

In third and fourth problems it is proportional to $N^{1/2}$

It is much more difficult to lunch preimage attack and second preimage attack compare to the collision attack

Attacks on Random Oracle Model

- Random Oracle model creates n bit digests
- The oracle selects one value $\in$ [0 2^n-1] randomly for the given message
  - Does not imply that selection is exhaustive, i.e., some values may never be selected whereas some other values are selected several times
- Hash function is public and value of n also public

Preimage attack

- In preimage attack
  - An adversary intercepts a digest ‘D’ and try to find any message M1 such that h(M1)=D

Algorithm 11.1 *Preimage attack+*

```
Preimage_Attack (D)
{                                          // consider a list of k messages
   for (i = 1 to k)
   {
        create (M [i])
        T ← h(M [i])                       // T is a temporary digest
        if (T = D) return M [i]
   }                                       // the probability to be at least 50 percent successful,
   return failure                          // k should be k ≈ 0.69*N i.e., ≈ 0.69*2^n
}
```

The difficulty of a preimage attack is proportional to 2^n.

Preimage attack (contd…)

A cryptographic hash function uses a digest of 64 bits. How many digests does Eve need to create to find the original message with the probability more than 0.5?

Solution

The number of digests to be created is $k \approx 0.69 \times 2^n \approx 0.69 \times 2^{64}$. This is a large number. Even if Eve can create 2^{30} (almost one billion) messages per second, it takes 0.69×2^{34} seconds or more than 500 years. This means that a message digest of size 64 bits is secure with respect to preimage attack, but, as we will see shortly, is not secured to collision attack.

Second preimage attack

- Intercepts digest ‘D’ with corresponding message M, i.e. D=h(M) and try to find another message M1 so that D=h(M1)

Algorithm 11.2 *Second preimage attack*

```
Second_Preimage_Attack (D, M)
{
   for (i = 1 to k −1)                    // consider a list of k-1 messages
   {
        create (M [i])
        T ← h (M [i])                     // T is a temporary digest
        if (T = D) return M [i]
   }
   return failure                         // the probability to be at least 50 percent successful,
}                                         // k should be k ≈ 0.69*N+1 i.e., ≈ 0.69*2^n +1
```

The difficulty of a preimage attack is proportional to 2^n.

Collision Attack

- Adversary needs to find two messages M1 and M2 such that h(M)=h(M1)

Algorithm 11.3 *Collision attack*

```
Collision_Attack
{
   for (i = 1 to k )
   {                                   // consider a list of k messages
        create (M[i])
        D[i] ← h (M[i])                // D [i] is a list of created digests
        for (j = 1 to i − 1)
        {
           if (D[i] = D[j]) return (M[i] and M[j])
        }                              // the probability to be at least 50 percent successful,
   }                                   // k should be k ≈ 1.18*N^½  i.e., ≈ 1.18*2^(n/2)
   return failure
```

The difficulty of a collision attack is proportional to $2^{n/2}$.

Collision Attack (contd…)

A cryptographic hash function uses a digest of 64 bits. How many digests does Eve need to create to find two messages with the same digest with the probability more than 0.5?

Solution

The number of digests to be created is $k \approx 1.18 \times 2^{n/2} \approx 1.18 \times 2^{32}$. If Eve can test 2^{20} (almost one million) messages per second, it takes 1.18×2^{12} seconds, or less than two hours. This means that a message digest of size 64 bits is not secure against the collision attack.

Alternate Collision Attack

- In this attack, an adversary needs to create two messages: one real and other one is bogus such that they have same digest
- Attacker considered two original messages M and M’ and then these messages are modified without changing the meaning
- From the messages: k different variants are generated
  - M: M_1, M_2, …, M_k and M’: M'_1, M'_2, …, M'_k

Alternate Collision Attack (contd…)

Algorithm 11.4 *Alternate collision attack*

```
Alternate_Collision_Attack (M [k], M′[k])
{
   for (i = 1 to k )
   {
      D[i] ← h (M[i])
      D′[i] ← h (M′[i])
      if (D [i] = D′[j]) return (M[i], M′[j])
   }
   return failure
}
```

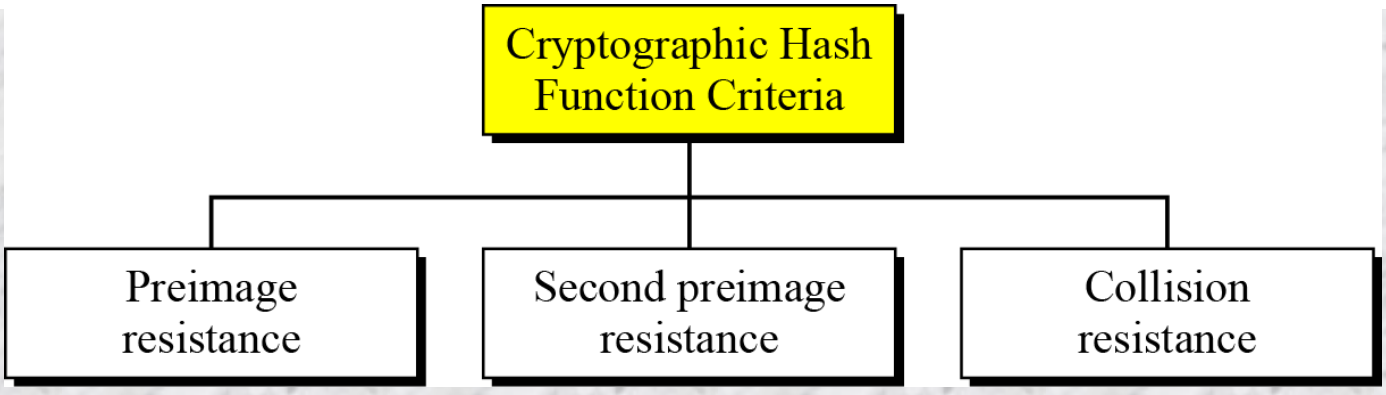
// the probability to be at least 50 percent successful,
// k should be $k \approx 0.83*N^{1/2}$ i.e., $\approx 0.83*2^{n/2}$

The difficulty of an alternative collision attack is proportional to $2^{n/2}$.

Summary of Attacks

Table 11.4 *Levels of difficulties for each type of attack*

Attack	*Value of k with P=1/2*	*Order*
Preimage	$k \approx 0.69 \times 2^n$	2^n
Second preimage	$k \approx 0.69 \times 2^n + 1$	2^n
Collision	$k \approx 1.18 \times 2^{n/2}$	$2^{n/2}$
Alternate collision	$k \approx 0.83 \times 2^{n/2}$	$2^{n/2}$



Level of difficulty for collision attack is much less than preimage and second preimage attacks

if hash algorithm is collision resistance, no worry about the first two attacks

Example

Originally hash functions with a 64-bit digest were believed to be immune to collision attacks

But with the increase in the processing speed, today everyone agrees that these hash functions are no longer secure

Eve needs only $2^{64/2} = 2^{32}$ tests to launch an attack with probability 1/2 or more

Assume she can perform 2^{20} (one million) tests per second. She can launch an attack in $2^{32}/2^{20} = 2^{12}$ seconds (almost an hour).

Example

MD5 (see Chapter 12), which was one of the standard hash functions for a long time, creates digests of 128 bits

To launch a collision attack, the adversary needs to test 2^{64} ($2^{128}/2$) tests in the collision algorithm

Even if the adversary can perform 2^{30} (more than one billion) tests in a second, it takes 2^{34} seconds (more than 500 years) to launch an attack

This type of attack is based on the Random Oracle Model. It has been proved that MD5 can be attacked on less than 2^{64} tests because of the structure of the algorithm

Example

SHA-1 (see Chapter 12), a standard hash function developed by NIST, creates digests of 160 bits

To launch a collision attack, the adversary needs to test $2^{160/2} = 2^{80}$ tests in the collision algorithm

Even if the adversary can perform 2^{30} (more than one billion) tests in a second, it takes 2^{50} seconds (more than ten thousand years) to launch an attack

However, researchers have discovered some features of the function that allow it to be attacked in less time than calculated above

Example

The new hash function, that is likely to become NIST standard, is SHA-512 (see Chapter 12), which has a 512-bit digest

This function is definitely resistant to collision attacks based on the Random Oracle Model. It needs $2^{512/2} = 2^{256}$ tests to find a collision with the probability of 1/2.

Attacks on the Structure

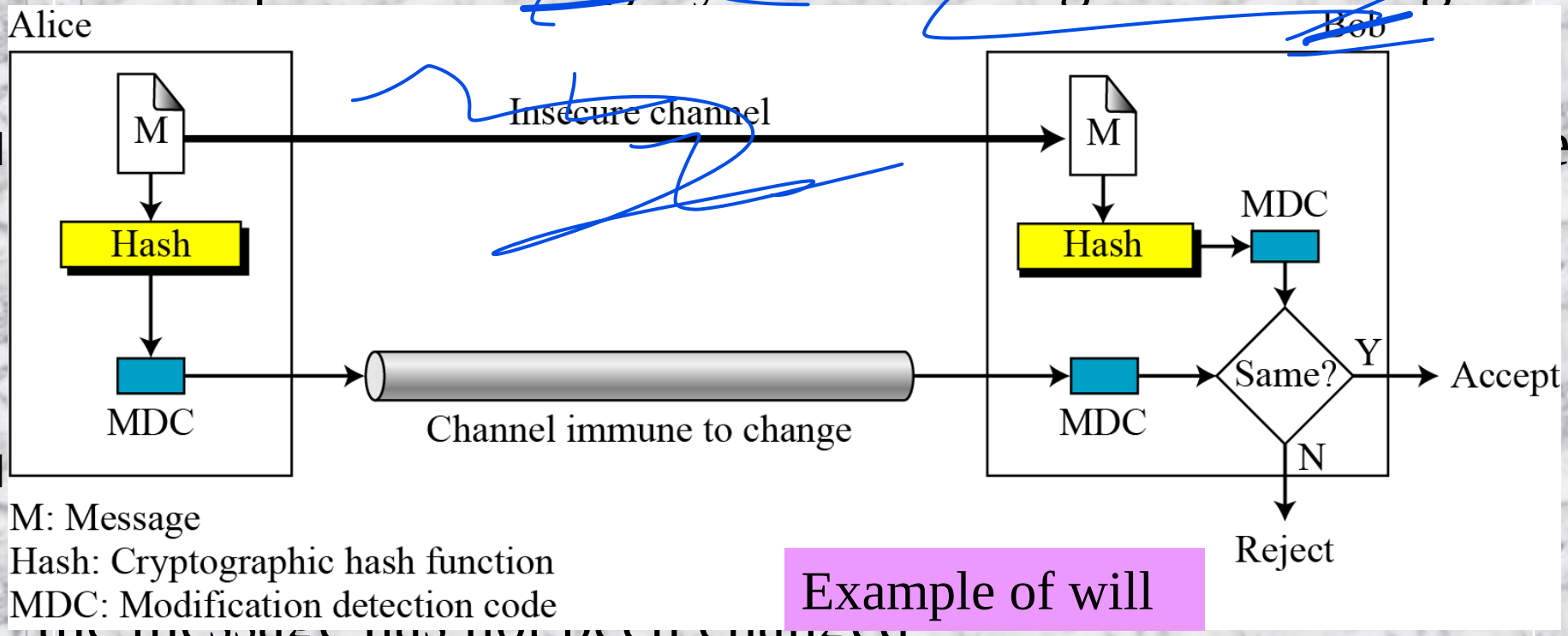
- *The adversary may have other tools to attack hash function. One of these tools, for example, is the meet-in-the-middle attack that we discussed for double DES*

MESSAGE AUTHENTICATION

- *The digest created by a cryptographic hash does not authenticate the sender of the message*
- *To provide message authentication, Alice needs to provide proof that it is Alice sending the message and not an impostor*
- *Digest is normally called a modification detection code (MDC)*
- *What we need for message authentication is a message authentication code (MAC).*

Modification Detection Code (MDC)

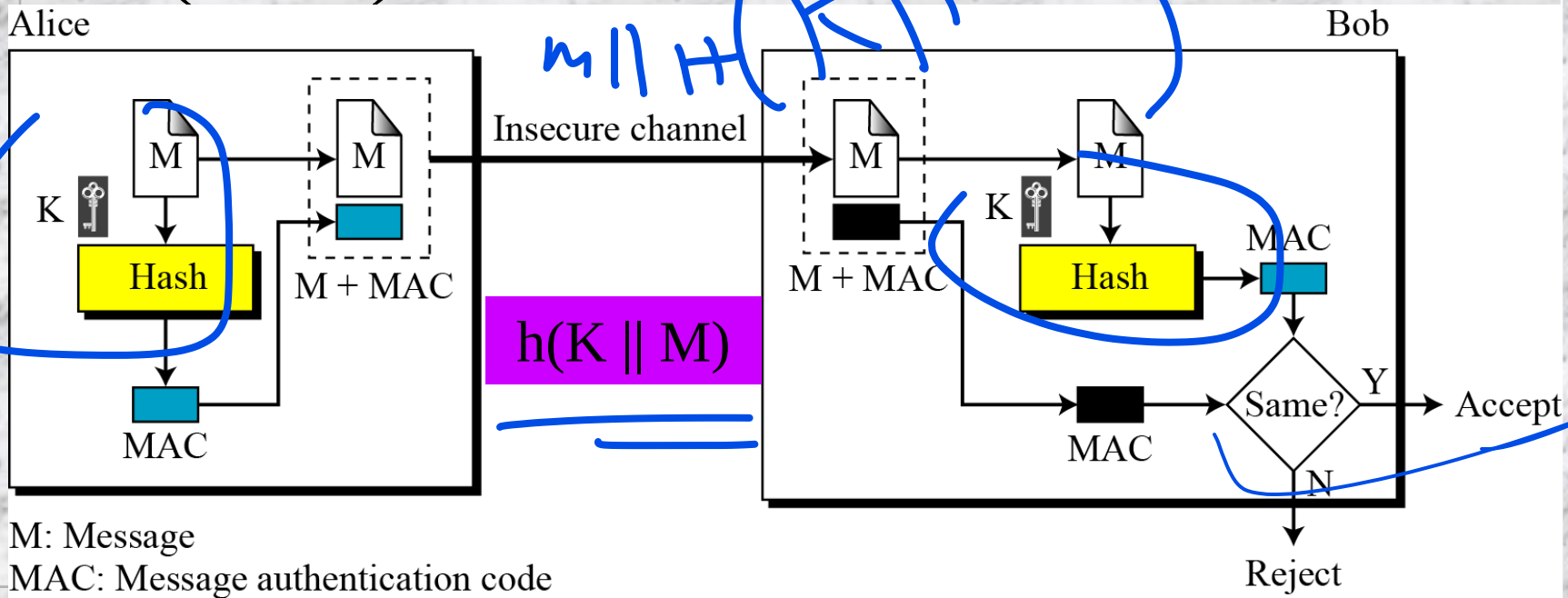
- A modification detection code (MDC) is a message digest that can prove the integrity of the message: that message has



Example of will

Message authentication code

- To ensure the integrity and origin authentication of a message, we need to change the modification detection code (MDC) to message authentication code (MAC)



M: Message
MAC: Message authentication code
K: A shared secret key

Message authentication code (contd…)

- Both the message and digest can send through an insecure channel
- Attacker can see this, but cannot forge a new message to replace it since attacker does not have the secret key
- Still MAC is insecure

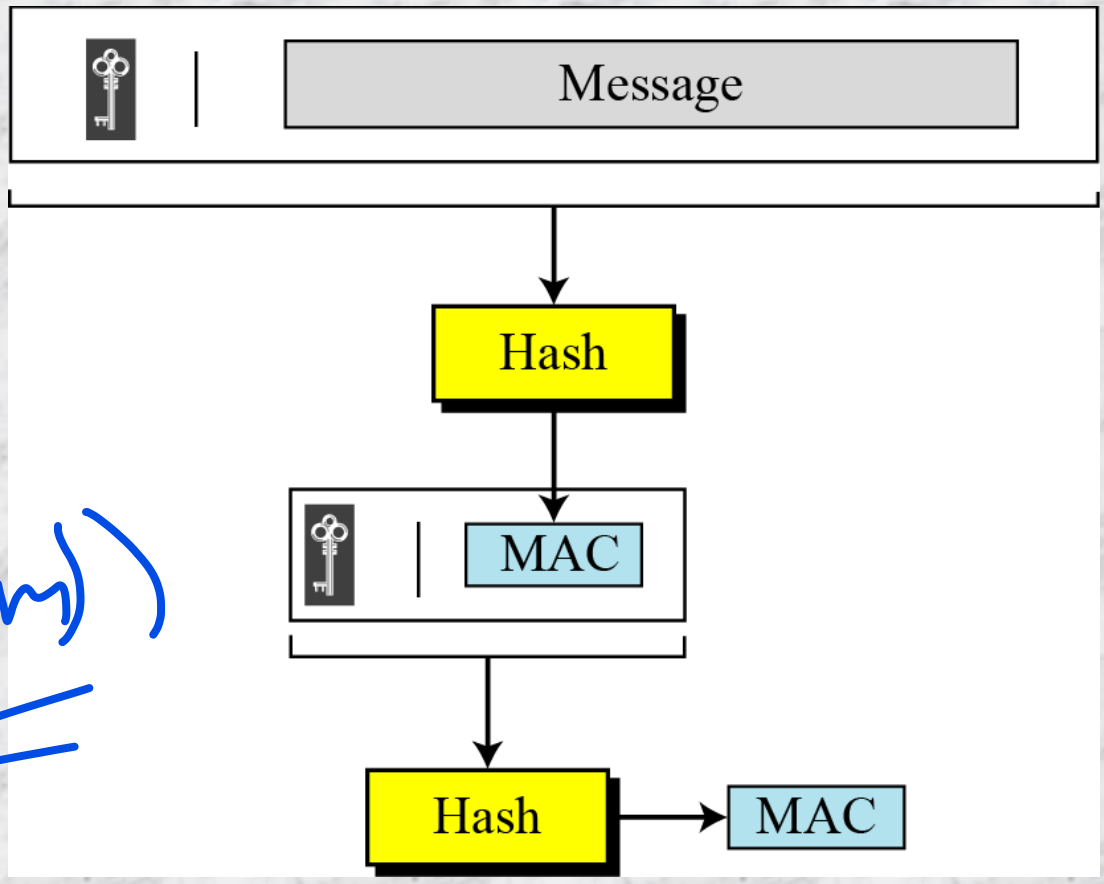
Security of MAC

- Attacker can intercepted M and h(K | M)
- If key size allows exhaustive search, then key can be identified and hence a new forged message can be used.
- Given some pairs of message and MAC, attacker can manipulate to come up with a new message and its MAC

The security of a MAC depends on the security of the underlying hash algorithm.

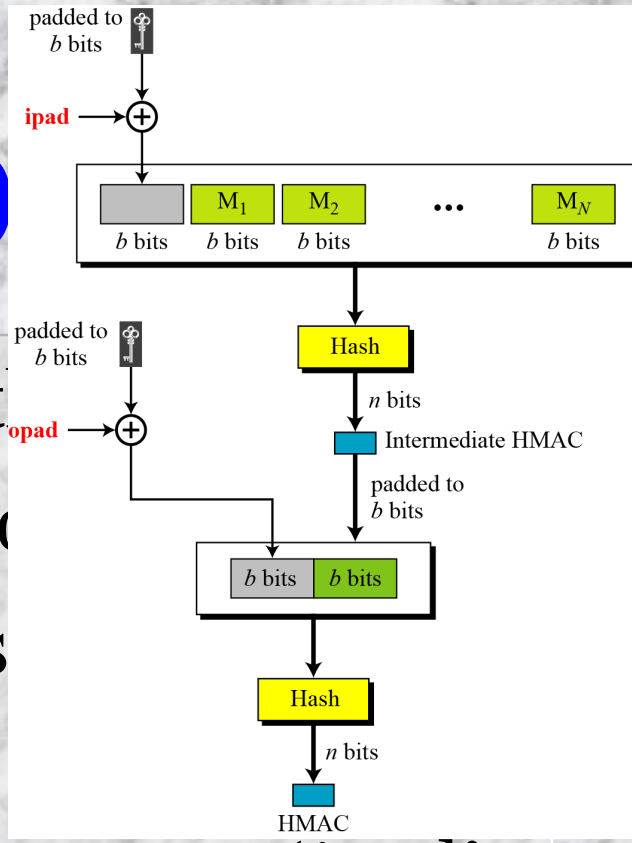
Nested MAC

- To improve the security, *Nested MAC* is designed



$H(K \| H(M))$

Hashed MAC (HMAC)

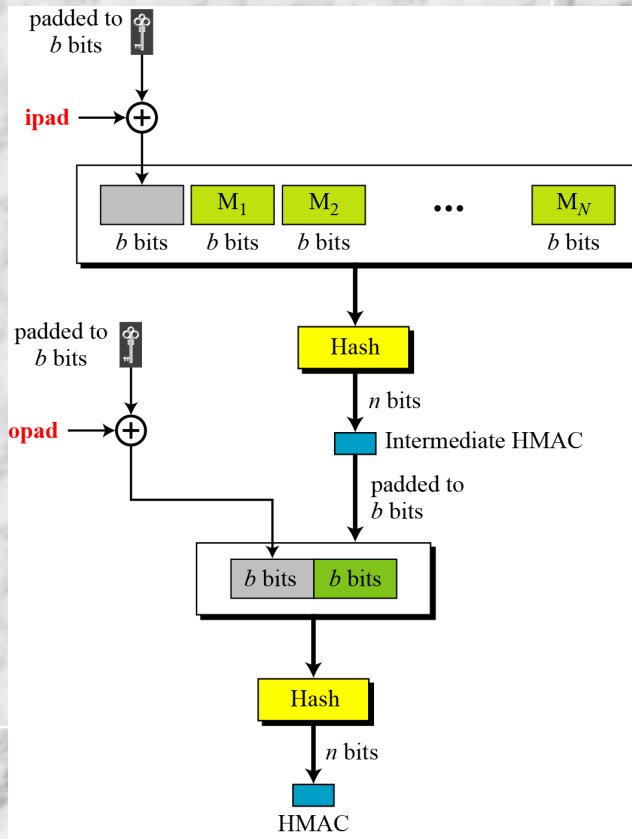


- Design of HMAC more complex t

1. The message is divided into N bl
2. Secret key is left padded with 0's key
3. Modified key is XOR-ed with a constant 'ipad'
4. The resulting block is prepended to the N blocks and results N+1 blocks
5. n-bits digest, called intermediate HMAC, is computed for above data

Hashed MAC (HMAC) (contd…)

- Secret key is left padded with 0's to create a b-bit key
- Modified key is XOR-ed with a constant 'opad'
- The result is prepended to intermediate HMAC
- Again, compute hash value, which is final digest



Cipher based MAC (CMAC)