Unit-6 –Message Authentication Codes

Message authentication is a mechanism or service used to verify the integrity of a message. Message authentication assures that data received are exactly as sent by (i.e., contain no modification, insertion, deletion, or replay) and that the purported identity of the sender is valid.

Need for Message Authentication

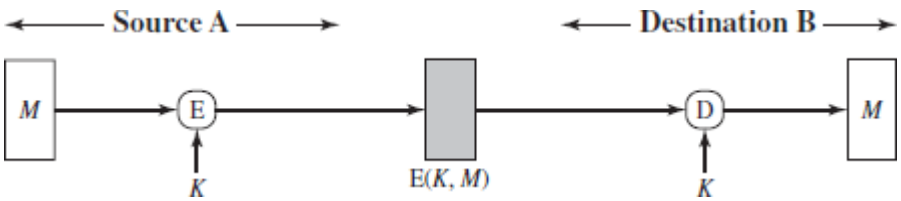
- Following attacks are possible which are the reason why authentication is needed:
  1. **Disclosure:** Release of message contents to any person not knowing the secret key.
  2. **Traffic analysis:** Discovery of the pattern of traffic between parties. Traffic analysis reveals information like the frequency and length of messages between parties and the communicating parties could be determined.
  3. **Masquerade:** Impersonating other person and sending messages.
  4. **Content modification:** Changes are made to the contents of a message. Changes may include insertion, deletion, transposition, and modification.
  5. **Sequence modification:** Sequence of messages between parties is modified. This attack may include insertion, deletion, and reordering.
  6. **Timing modification:** Delay or replay of messages.
  7. **Source repudiation:** Denial of transmission of message by source.
  8. **Destination repudiation:** Denial of receipt of message by destination.
- Message authentication verifies that received messages come from the alleged source and have not been altered.
- Message authentication may also verify sequencing and timeliness.

Authentication Techniques

- Following techniques are used for authentication:
  - **Hash function:** Hash function maps a message of any length into a fixed-lengthhash value, which serves as the authenticator.
  - **Message encryption:** The ciphertext of the entire message serves as its authenticator.
  - **Message authentication code (MAC):** A MAC is a function of the message and a secretkey that produces a fixed-length value that serves as the authenticator.
- Authentication using message encryption is explained below:

Message Encryption

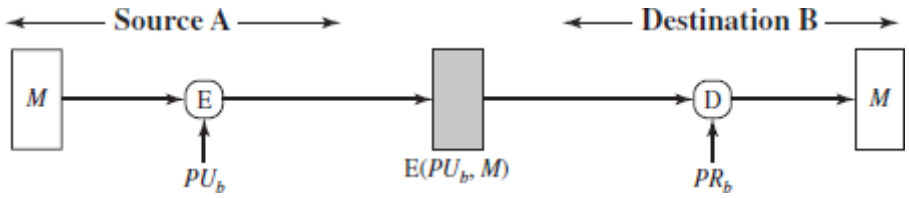
- **Symmetric Encryption:** A message **M** transmitted from source A to destination B is encrypted using a secret key K shared by A and B.



Confidentiality and authentication with symmetric encryption

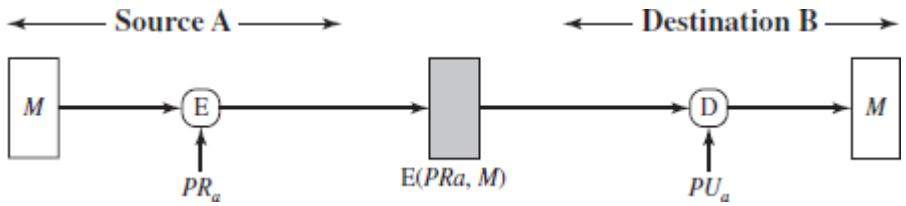
- No other party knows the key, and hence **confidentiality** is provided as no other party can recover the plaintext of the message without the knowledge of key.
- The message must have come from A because A is the only other party that possesses *K* and therefore the only other party which can construct cipher text that can be decrypted with *K*. Thus, authentication is provided.
- Furthermore, if *M* is recovered, B knows that none of the bits of *M* have been altered, because an opponent that does not know *K* would not know how to alter bits in the cipher text to produce desired changes in the plaintext. Thus, **data integrity** is also provided.

- If the message contains regular language, then the legitimacy of the message can be determined.
- But if the message contains arbitrary data like binary object file, digitized X-ray, then alteration in the message cannot be determined by simply looking at the messages.
- In that case, plaintext must have some structure like some message based function (one example is checksum) or add TCP header if TCP/IP is being used.
- **Public-Key Encryption:** The source (A) uses the public key $\mathbf{PU_b}$ of the destination (B) to encrypt *M*. Because only B has the corresponding private key $\mathbf{PR_b}$, only B can decrypt the message. But this scheme provides **confidentiality** but not authentication because any opponent could also use B's public key to encrypt a message, claiming to be A.



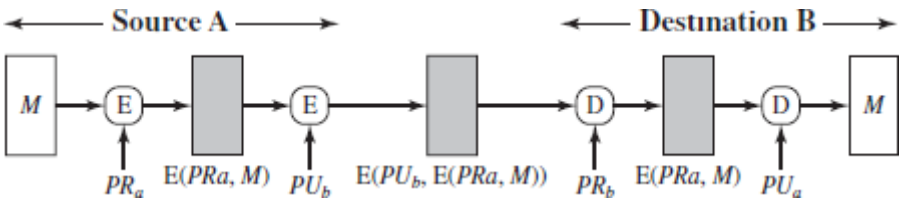
Confidentiality using public key encryption

- To provide authentication, A uses its private key to encrypt the message, and B uses A's public key to decrypt it. The message must have come from A because A is the only party that possesses $\mathbf{PR_a}$. Anyone with $\mathbf{PU_a}$ can decrypt the message. This scheme also provides digital signature because only A could have constructed the cipher text by encrypting it with $\mathbf{PR_a}$.



Authentication using public key encryption

- If both authentication and confidentiality is needed, then message is encrypted using both $\mathbf{PU_a}$ and **PRa.by** using its private key to encrypt. Note that this scheme does not provide confidentiality.



Confidentiality and Authentication using public key encryption

- This scheme also requires some structure in plaintext if it contains arbitrary data.

Message Authentication Code /Cryptographic Checksum

- Cryptographic checksum or MAC is a function of the message and a secret key that produces a fixed-length value that serves as the authenticator.

$$\text{MAC} = \text{MAC}(K, M)$$

where

M = input message
C = MAC function
K = shared secret key
MAC = message authentication code

- A MAC function is similar to encryption. One difference is that the MAC algorithm need not be reversible, as in the case of decryption.
- A MAC function is generally a many-to-one function.

Application of MAC

- Three situations in which a message authentication code is used are:
  1. **Many applications need to broadcast message to a number of destinations.**
     - ✓ Examples are notification to users that the network is now unavailable or an alarm signal in a military control center.
     - ✓ Instead of decrypting message at every node it is cheaper and more reliable to have only one destination responsible for monitoring authenticity.
     - ✓ The message is broadcasted in plaintext with an associated message authentication code. The responsible system has the secret key and performs authentication.
     - ✓ If a violation occurs, the other destination systems are alerted by a general alarm.
  2. **One side in the communication has a heavy load and cannot afford the time to decrypt all incoming messages.**
     - ✓ Authentication is carried out on a selective basis. Messages are chosen at random for checking.
  3. **Authentication of a computer program in plaintext.**
     - ✓ The computer program can be executed without having to decrypt it every time.
     - ✓ However, if a message authentication code were attached to the program, it could be checked whenever assurance is required about the integrity of the program.

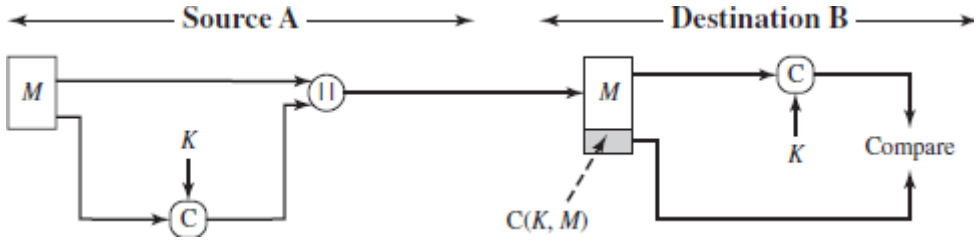
Basic Uses of MAC

- A MAC is an authentication technique involves the use of a secret key to generate a small fixed-size block of data, known as a **cryptographic checksum** or MAC. The MAC is then appended to the message.
- Here, sender and receiver share a secret key.
- When A has to send a message to B, it calculates the MAC as a function of the message and the key:

MAC = MAC (*K*, *M*)

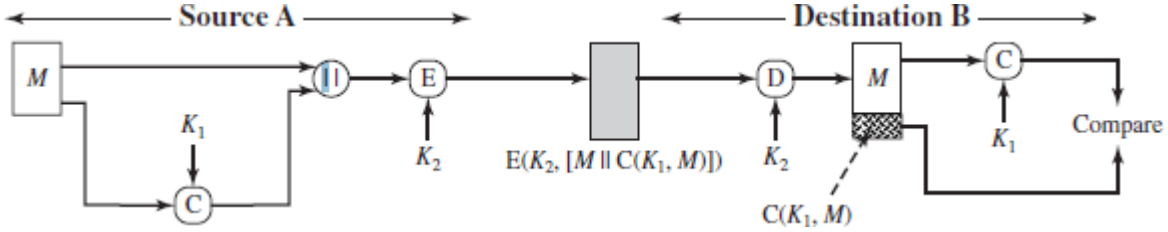
Where M is plaintext
C is the MAC function
K is the secret key and
MAC is the message authentication code.

- The message plus MAC are transmitted to the intended recipient.
- The recipient performs the same calculation on the received message, using the same secret key, to generate a new MAC. The received MAC is compared to the calculated MAC.



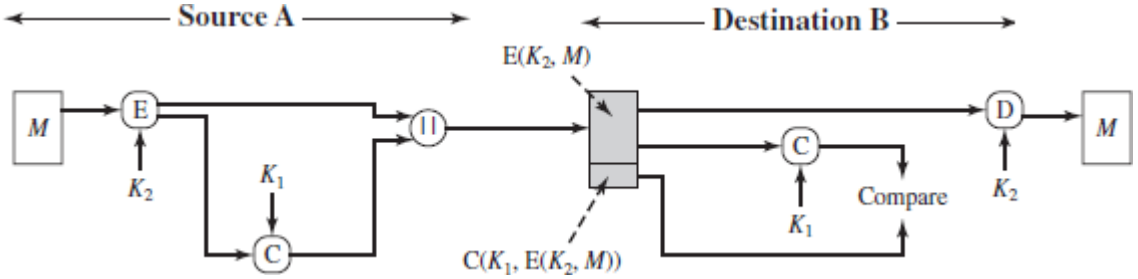
Authentication using MAC, no confidentiality

- Since only the receiver and the sender know the secret key, and if the received MAC matches the calculated MAC, then
  - The receiver is assured that the message has not been altered. If an attacker alters the message but does not alter the MAC, then the receiver's calculation of the MAC will differ from the received MAC.
  - The receiver is assured that the message is from the alleged sender. Because no one else knows the secret key.
- Confidentiality can be provided by performing message encryption either after or before the MAC algorithm.
- In both these cases, two separate keys are needed, each of which is shared by the sender and the receiver.
- MAC can be calculated with the message as input and then concatenated to the message. The entire block is then encrypted.



Authentication and confidentiality using MAC

- It is preferable to tie the authentication directly to the plaintext, hence the above method is typically preferred.
- Alternately, the message is encrypted first. Then the MAC is calculated using the resulting cipher text and is concatenated to the cipher text.



Authentication and confidentiality using MAC

Requirements for Message Authentication Codes

- A MAC, also known as a cryptographic checksum, is generated by a function C of the form

$$T = MAC(K, M)$$

Where
M is a variable-length message,
K is a secret key shared only by sender and receiver, and
MAC (K, M) is the fixed-length authenticator also called a **tag**.

- The tag is appended to the message at the source at a time when the message is assumed or known to be correct. The receiver authenticates that message by re-computing the tag.
- When an entire message is encrypted for confidentiality, using either symmetric or asymmetric encryption, the security of the scheme generally depends on the bit length of the key.
- Barring some weakness in the algorithm, the opponent must resort to a brute-force attack using all possible keys.
- On average, such an attack will require 2^{k-1} attempts for a k-bit key. In particular, for a cipher text only attack, the opponent, given cipher text C, performs $P_i = D(K_i, C)$ for all possible key values K_i until a P_i is produced that matches the form of acceptable plaintext.
- Then the MAC function should satisfy the following requirements.
  1. If an opponent observes M and $MAC(K, M)$, it should be computationally infeasible for the opponent to construct a message M' such that $MAC(K, M) = MAC(K, M')$.
  2. $MAC(K, M)$ should be uniformly distributed in the sense that for randomly chosen messages, M and M', the probability that $MAC(K, M) = MAC(K, M')$ is 2^{-n}, where n is the number of bits in the tag.
  3. Let M' be equal to some known transformation on M. That is, $M' = f(M)$. For example, f may involve inverting one or more specific bits. In that case, $\Pr[MAC(K, M) = MAC(K, M')] = 2^{-n}$

Security of MACs/ Attacks on MACs

We group attacks on MACs into two categories: brute-force attacks and cryptanalysis.

Brute-Force Attacks

- A brute-force attack on a MAC requires more known message-MAC pairs than a brute-force attack on a hash function.
- There are two types of possible attack:
  - attack the key space
  - attack the MAC value

1. Attacking the key space

- ✓ If an attacker can determine the MAC key, then it is possible to generate a valid MAC value for any input.
- ✓ Suppose the key size is **k** bits and that the attacker has one known text–tag (MAC) pair.
- ✓ The attacker can then compute the **n-bit** tag on the known text for all possible keys.
- ✓ At least one key will produce the correct MAC value for the message. Till now, the level of effort is $\mathbf{2^k}$.
- ✓ However, the MAC is a many-to-one mapping, so there may be other keys that produce the correct value.
- ✓ Thus, if more than one key is found to produce the correct value, additional text–tag pairs must be tested.
- ✓ The level of effort becomes less with each additional text–MAC pair and after 2 or 3 levels, a single key is obtained.

2. Attacking the MAC value

- ✓ The attacker will try to generate a valid MAC for a given message or to find a message that matches a given MAC value.
- ✓ Here the level of effort is that of $\mathbf{2^n}$.
- ✓ This attack cannot be conducted off line without further input; the attacker will require chosen text–tag pairs or knowledge of the key.

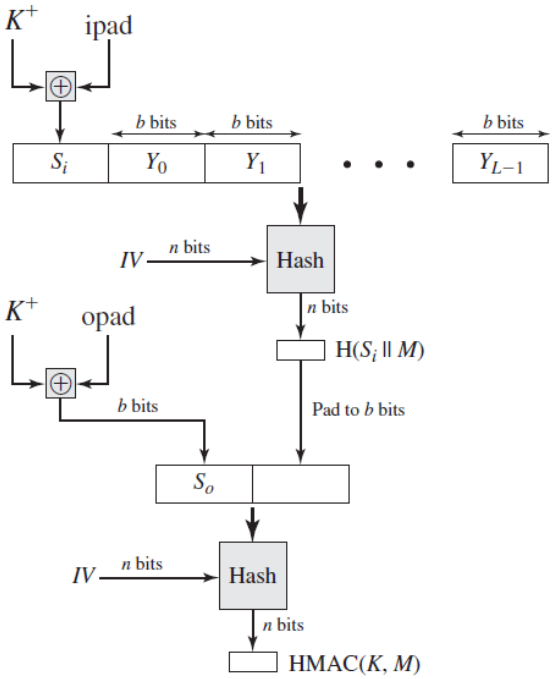
Cryptanalysis

- Cryptanalytic attacks on MAC algorithms try to exploit some property of the algorithm to perform some attack other than an exhaustive search.

- The way to measure the resistance of a MAC algorithm to cryptanalysis is to compare its strength to the effort required for a brute-force attack.
- An ideal MAC algorithm will require a cryptanalytic effort greater than or equal to the brute-force effort.

MACs Based On Hash Functions: HMAC

- AMAC derived from hash function is called HMAC.
- The reason for developing HMAC were that hash functions incur less overhead than encryption and the code of hash functions is easily and freely available.
- The overall operation of HMAC is shown below:



1. Append zeros to the left end of *K* to create a **b-bit** string **K⁺.**
2. XOR **K⁺** with **ipad** to produce the *b*-bit block ***S*i**. Value of ipad is 36 in hexadecimal.
3. Append *M* to ***S*i**.
4. Apply H to the stream generated in the above step.
5. XOR **K⁺** with **opad** to produce the b-bit block ***S*o**. Value of opad is 5C in hexadecimal.
6. Append the hash result H from step 4 to***S*o**.
7. Apply H to the stream generated in the above step and output the result.

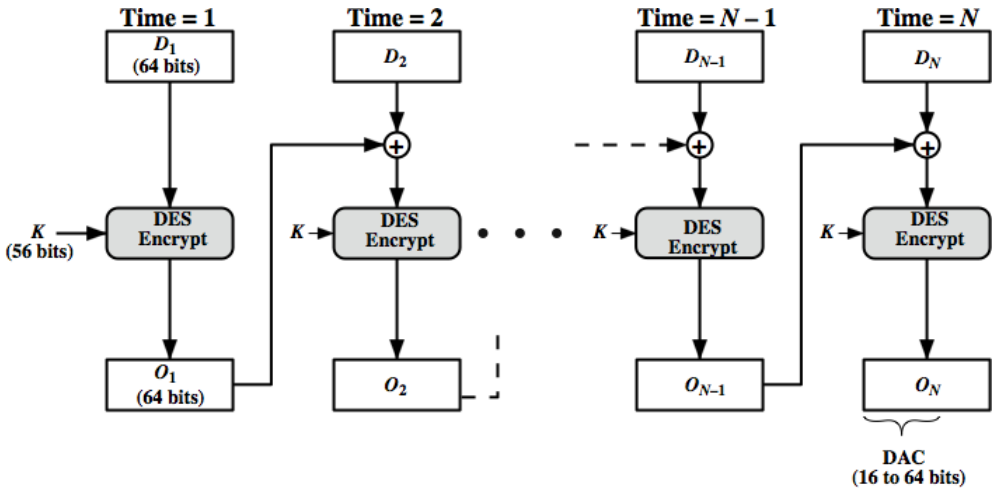
- XOR with ipad results in flipping one-half of the bits of *K.*
- Similarly, the XOR with opad results in flipping one-half of the bits of*K*, but a different set of bits.

Security of HMAC

- The security of any HMAC function is based on the cryptographic strength of the underlying hash function.
- The security of a MAC function is expressed in terms of the probability of successful forgery with a given amount of time spent by the forger and a given number of message-MAC pairs created with the same key.
- The probability of successful attack on HMAC is equivalent to one of the following attacks on the embedded hash function:
  - The attacker is able to compute an output of the compression function even with an IV that is random, secret, and unknown to the attacker.
  - The attacker finds collisions in the hash function even when the IV is random and secret.

MACS Based on Block Ciphers: Data Authentication Algorithm (DAA)

- The **Data Authentication Algorithm** (DAA), based on DES, has been one of the most widely used MACs for a number of years.
- Security weaknesses in this algorithm have been discovered, and it is being replaced by newer and stronger algorithms.
- The algorithm can be defined as using the cipher block chaining (CBC) mode of operation of DES with an initialization vector of zero.
- The data to be authenticated are grouped into contiguous 64-bit blocks: D_1, D_2, …, D_N.
- If necessary, the final block is padded on the right with zeroes to form a full 64-bit block.
- Using the DES encryption algorithm E and a secret key K, a data authentication code (DAC) is calculated as follows:



$$O_1 = E(K, D)$$
$$O_2 = E(K, [D_2 \oplus O_1)$$
$$O_3 = E(K, [D_3 \oplus O_2)$$
.
.
.
$$O_N = E(K, [D_N \oplus O_{N-1})$$

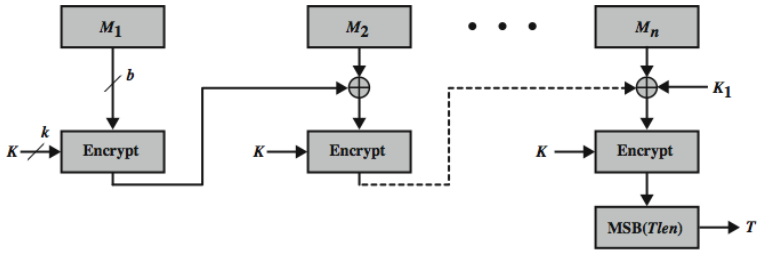
- The DAC consists of either the entire block O_N or the leftmost M bits of the block, with $16 \leq M \leq 64$.

MACS Based on Block Ciphers: Cipher-Based Message Authentication Code (CMAC)

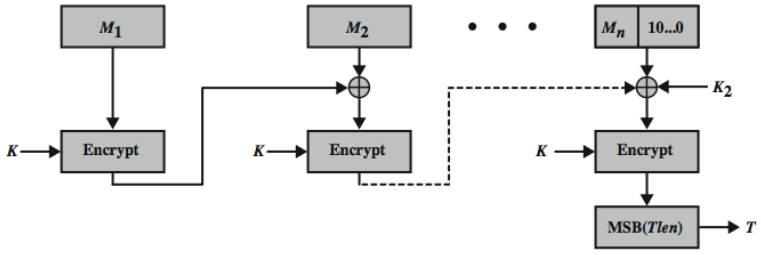
- As was mentioned, DAA has been widely adopted in government and industry.
- MAC is secure under a reasonable set of security criteria, with the following restriction.
  - Only messages of one fixed length of **mn** bits are processed, where **n** is the cipher block size and **m** is a fixed positive integer.
- Black and Rogaway demonstrated that this limitation could be overcome using three keys:
- one key of length **K** to be used at each step of the cipher block chaining and two keys of length **n**, where **k** is the key length and **n** is the cipher block length.
- This proposed construction was refined by Iwata and Kurosawa so that the two n-bit keys could be derived from the encryption key.
- This refinement, adopted by NIST, is the **Cipher-based Message Authentication Code** (CMAC) mode of operation for use with AES and triple DES.

operation of CMAC

- The message is divided into **n** blocks (M1, M2, …, Mn).
- The algorithm makes use of a k-bit encryption key **K** and an n-bit constant, $\mathbf{K_1}$.
- For AES, the key size is 128, 192, or 256 bits;
- for triple DES, the key size is 112 or 168 bits.
- CMAC is calculated as follows:



(a) Message length is integer multiple of block size



(b) Message length is not integer multiple of block size

$C_1 = E(K, M_1)$
$C_2 = E(K, [M_2 \oplus C_1)$
$C_3 = E(K, [M_3 \oplus C_2)$
.
.
.
$C_n = E(K, [M_n \oplus C_{n-1} \oplus K_1)$
$T = MSB_{Tlen}(C_n)$
Where
$T =$ message authetication code, also referred to as the tag
$Tlen =$ bit length of T
$MSB_s(X) =$ the s leftmost bits of the bit string$X$

- If the message is not an integer multiple of the cipher block length, then the final block is padded to the right with a 1 and as many 0s as necessary.
- The CMAC operation then proceeds as before, except that a different n-bit key K_2 is used instead of K_1.
- The two n-bit keys are derived from the k-bit encryption key as follows.
  $L = E(K, 0^n)$
  $K_1 = L \cdot x$
  $K_2 = L \cdot x^2 = (L \cdot x) \cdot x$

where multiplication (●) is done in the finite field GF(2^n) and **x** and $\mathbf{x^2}$ are first- and second-order polynomials that are elements of GF(2^n).