# Module: 03

Hashes, Message Digests and Digital Certificates

Lecture : 1

**Motivation:**

Encryption is most commonly used for secrecy; we usually encrypt something so that its contents or even its existence are unknown to all but a privileged audience. In some cases, however, integrity is a more important concern than secrecy. For example, in a document retrieval system containing legal records, it may be important to know that the copy retrieved is exactly what was stored. Likewise, in a secure communications system, the need for the correct transmission of messages may override secrecy concerns. Let us look at how encryption provides integrity.

In most files, the elements or components of the file are not bound together in any way. That is, each byte or bit or character is independent of every other one in the file. This lack of binding means that changing one value affects the integrity of the file, but that one change can easily go undetected.

**Syllabus:**

| Lecture no | Content | Duration (Hr) | Self-Study (Hrs) |
|---|---|---|---|
| 1 | Cryptographic Hash Function, Properties of secure hash function | 1 | 1 |
| 2 | MAC, HMAC, CMAC | 1 | 1 |
| 3 | Digital Signatures and Certificates | 1 | 1 |
| 4 | MD5, SHA-1 | 1 | 2 |

**Learning Objective:**
**Learner should be able to gain knowledge about**

- Concepts of cryptographic Hash functions
- Applications of Hash functions
- Secure Hash Algorithms
- Message Authentications Codes
- Digital Signatures and Certificates

**Theoretical Background:**

Hash functions are extremely useful and appear in almost all information security applications. At the heart of a hashing is a mathematical function that operates on two fixed-size blocks of data to create a hash code. This hash function forms the part of the hashing algorithm. Hashing algorithm involves rounds of above hash function like a block cipher. Each round takes an input of a fixed size, typically a combination of the most recent message block and the output of the last round. Since, the hash value of first message block becomes an input to the second hash operation, output of which alters the result of the third operation, and so on. This effect, known as an avalanche effect of hashing.

Avalanche effect results in substantially different hash values for two messages that differ by even a single bit of data.

There are two direct applications of hash function based on its cryptographic properties.

One is the Password Storage (Hash functions provide protection to password storage)
Other is Data Integrity Check (Data integrity check is a most common application of the hash functions. It is used to generate the checksums on data files. This application provides assurance to the user about correctness of the data)

**Key Definitions:**

**Hash function:** A hash function is a mathematical function that converts a numerical input value into another compressed numerical value. The input to the hash function is of arbitrary length but output is always of fixed length

**Message Digest:** Values returned by a hash function are called Message Digest or simply Hash Values

**Course Content:**
Cryptographic Hash Functions

What we would like to do is somehow put a seal or shield around the file so that we can detect when the seal has been broken and thus know that something has been changed. This notion is similar to the use of wax seals on letters in medieval days; if the wax was broken, the recipient would know that someone had broken the seal and read the message inside. In the same way, cryptography can be used to seal a file, encasing it so that any change becomes apparent. One technique for providing the seal is to compute a cryptographic function, sometimes called a **hash** or **checksum** or **message digest** of the file.

The hash function has special characteristics. For instance, some encryptions depend on a function that is easy to understand but difficult to compute. For a simple example, consider the cube function, $y = x^3$. It is relatively easy to compute $x^3$ by hand, with pencil and paper, or with a calculator. But the inverse function, is much more difficult to compute. And the function $y = x^2$ has no inverse function since there are two possibilities for and -x. Functions like these, which are much easier to compute than their inverses, are called **one-way functions**.

A one-way function can be useful in an encryption algorithm. The function must depend on all bits of the file being sealed, so any change to even a single bit will alter the checksum result. The checksum value is stored with the file. Then, each time the file is accessed or used, the checksum is recomputed. If the computed checksum matches the stored value, it is likely that the file has not been changed.

A cryptographic function, such as the DES or AES, is especially appropriate for sealing values, since an outsider will not know the key and thus will not be able to modify the stored value to match with data being modified. For low-threat applications, algorithms even simpler than DES or AES can be used. In block encryption schemes, **chaining** means linking each block to the previous block's value (and therefore to all previous blocks), for example, by using an exclusive OR to combine the encrypted previous block with the encryption of the current one. A file's cryptographic checksum could be the last block of the chained encryption of a file since that block will depend on all other blocks.

The most widely used cryptographic hash functions are **MD4, MD5** (where MD stands for Message Digest), and **SHA/SHS** (Secure Hash Algorithm or Standard). The MD4/5 algorithms were invented by Ron Rivest and RSA Laboratories. MD5 is an improved version of MD4. Both condense a message of any size to a 128-bit digest. SHA/SHS is similar to both MD4 and MD5; it produces a 160-bit digest.

**The typical features of Hash functions are:**

- **Fixed Length Output (Hash Value)**
- Hash function coverts data of arbitrary length to a fixed length. This process is often referred to as **hashing the data**.

- In general, the hash is much smaller than the input data, hence hash functions are sometimes called **compression functions**.
- Since a hash is a smaller representation of a larger data, it is also referred to as a **digest**.
- Hash function with n bit output is referred to as an **n-bit hash function**. Popular hash functions generate values between 160 and 512 bits.

**Properties of Hash functions:**

**Pre-Image Resistance**

- This property means that it should be computationally hard to reverse a hash function.
- In other words, if a hash function h produced a hash value z, then it should be a difficult process to find any input value x that hashes to z.
- This property protects against an attacker who only has a hash value and is trying to find the input.

**Second Pre-Image Resistance**

- This property means given an input and its hash, it should be hard to find a different input with the same hash.
- In other words, if a hash function h for an input x produces hash value h(x), then it should be difficult to find any other input value y such that h(y) = h(x).
- This property of hash function protects against an attacker who has an input value and its hash, and wants to substitute different value as legitimate value in place of original input value.

**Collision Resistance**

- This property means it should be hard to find two different inputs of any length that result in the same hash. This property is also referred to as collision free hash function.
- In other words, for a hash function h, it is hard to find any two different inputs x and y such that h(x) = h(y).

## Message Integrity and Authentication

The Cryptography systems that discussed so far provide *Secrecy* or *Confidentiality,* but not *Integrity*. However, there are occasions that may not even need confidentiality but instead have *Integrity*

**Example:**

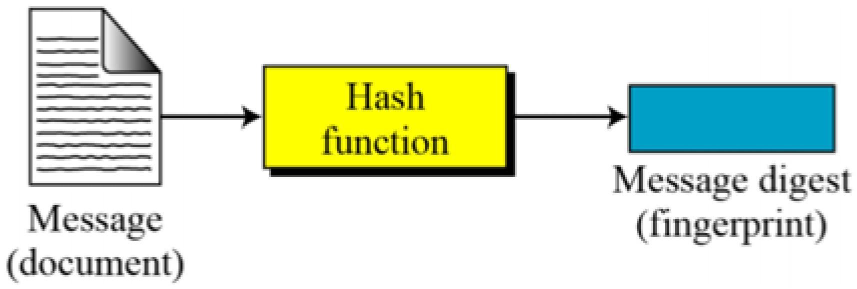
Alice may write a will to distribute his estate upon his death. The will does not need to be encrypted because anyone can examine the will after he died. However, the integrity of the will needs to be preserved so that the contents of the will is unchanged. On way to preserve the integrity of a document is through the use of a fingerprint. If Alice needs to ensure the content of her document will not changed, she can put her fingerprint at the bottom of the document. Eve cannot modify the contents of the document or create a false document because she cannot forge Alice's fingerprint.

Q. How to ensure the document not been changed?

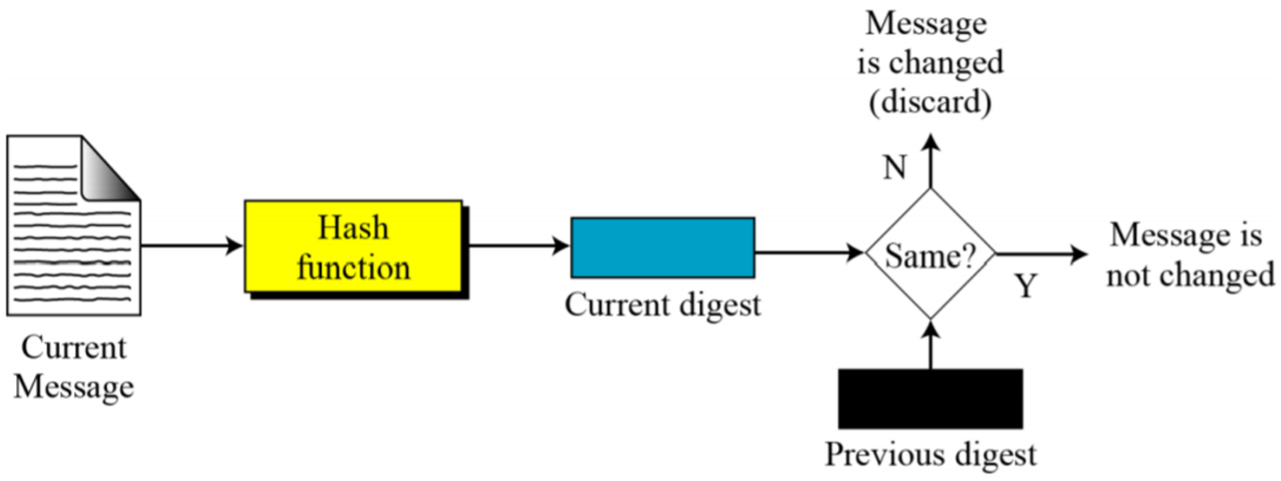
A. Compare Alice's fingerprint on the document with Alice's fingerprint on file

The electronic equivalent of the document & fingerprint pair is the ***Message*** & ***Digest*** **pair**

A message is passed through an algorithm called a ***Cryptographic Hash Function*** to preserve the integrity. The function creates a compressed image of the message that can be used like a fingerprint.



- Checking Integrity



## Message Authentication Code:

1. Alice uses a hash function to create MAC from the concatenation of the key and the message, h(K|M).
2. She send the message and the MAC to Bob over the channel.
3. Bob separates the message from the MAC.
4. He them makes a new MAC from the concatenation of the message and the secret key.
5. Bob then compares the newly created MAC with the one received.
6. If the two MAC match, the message is authentic and has not modified by any intruder.

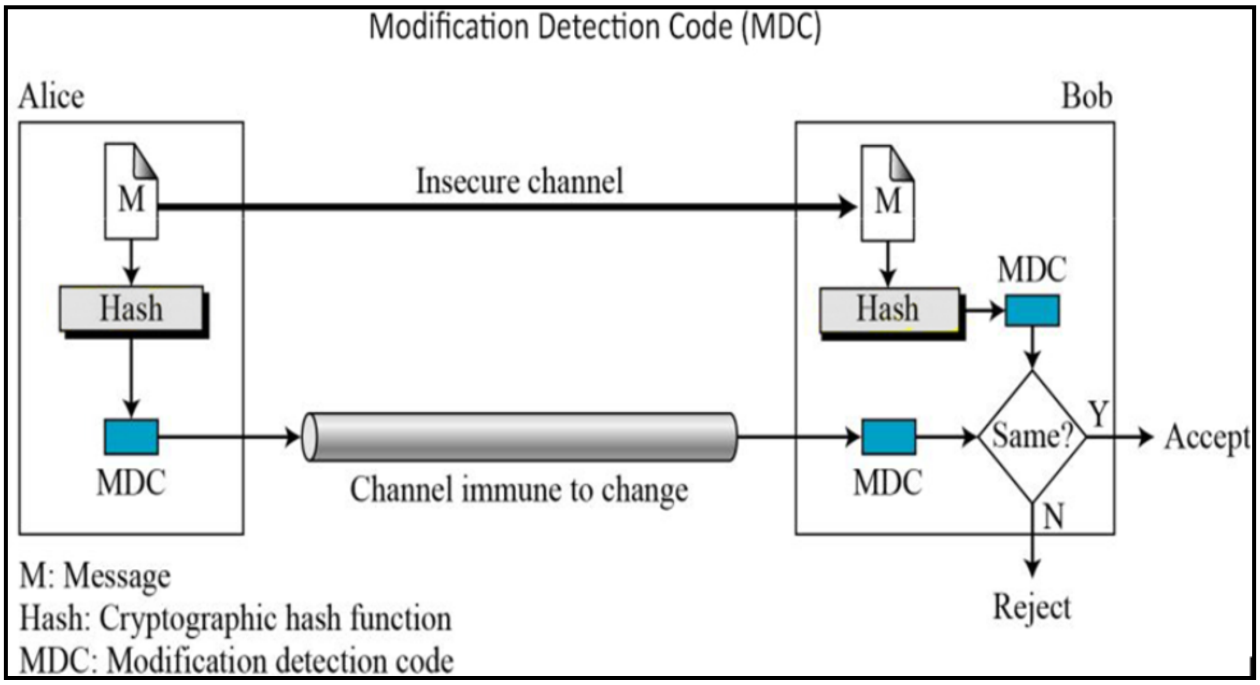
## Message Detection Code (MDC)

The **Message Digest** guarantees the integrity of a message that not been changed. However, message digest does not authenticate the sender of the message. To provide message authentication, sender needs to provide proof that he/she sending the message & not an imposter. The digest created by a cryptographic hash function is called a **Modification Detection Code (MDC)** to detect any modification in the message. **For message authentication, we need a Message Authentication Code (MAC).**

- A modification detection code (MDC) is a message digest that can prove the integrity of the message: that message has not been changed.
- If Alice needs to send a message to Bob and be sure that the message will not change during transmission, Alice can create a message digest, MDC, and send both the message and the MDC to Bob.
- Bob can create a new MDC from the message and compare the received MDC and the new MDC. If they are the same, the message has not been changed.

- A modification detection code (MDC) is a message digest that can prove the integrity of the message: that message has not been changed.

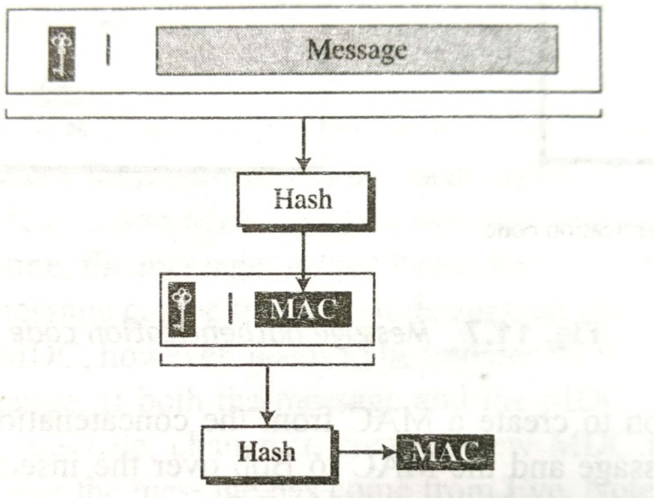
    - If Alice needs to send a message to Bob and be sure that the message will not change during transmission, Alice can create a message digest, MDC, and send both the message and the MDC to Bob.
    - Bob can create a new MDC from the message and compare the received MDC and the new MDC. If they are the same, the message has not been changed.


Modification Detection Code (MDC)
Alice
Bob
M
Insecure channel
M
Hash
Hash
MDC
MDC
Channel immune to change
MDC
Same?
Y
Accept
N
Reject
M: Message
Hash: Cryptographic hash function
MDC: Modification detection code

**Nested MACs**

It Used to improve the security of MAC, **nested MACs** are used. In nested MAC the hashing is done in two steps:

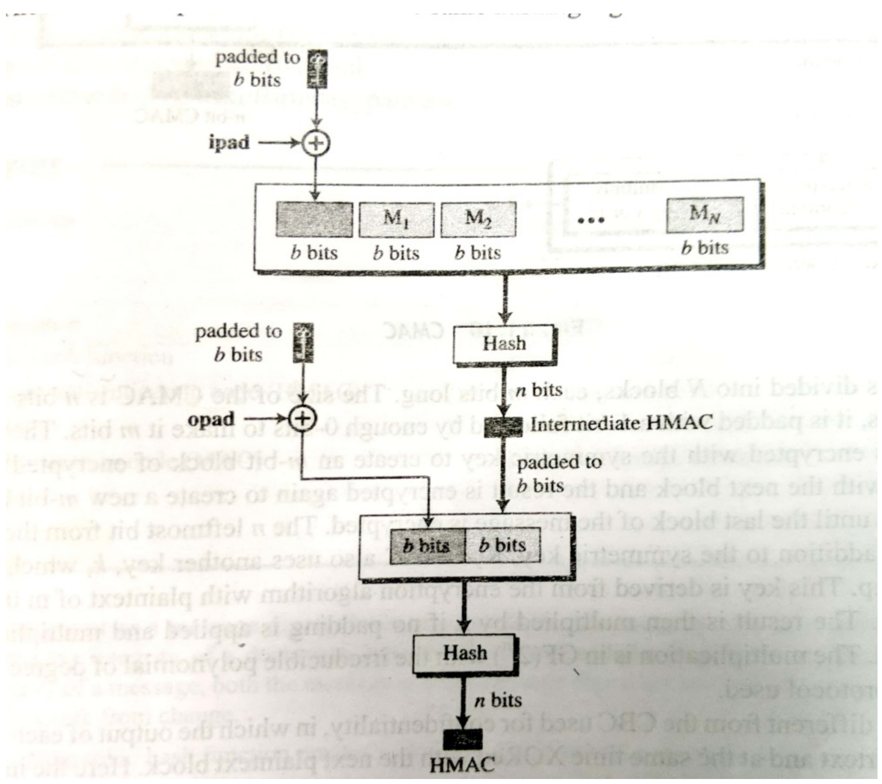
1. The key is concatenated with the message and is hashed to create an intermediate digest.
2. The key is concatenated with the intermediate digest to create the final digest



Lecture : 2

## Hashed Message Authentication Code (HMAC)

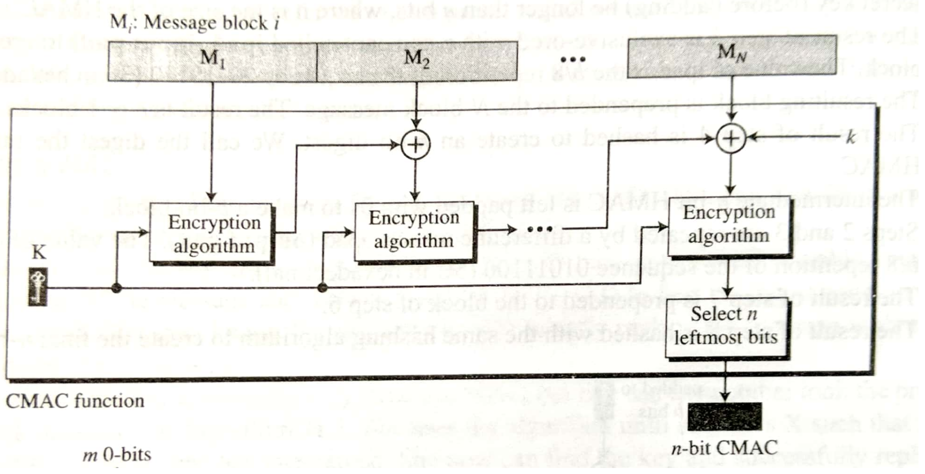
- It is the standard version of nested MAC. The implementation of HMAC is much more complex than the simplified nested MAC.
- **HMAC – Process**



1. The message is divided into N blocks, each of b bits.
2. The secret key is left padded with 0's to create a b-bit key.
3. The result of step 2 is XORed with a constant called **ipad** (input pad) to create a b-bit block. The value of ipad is 36 in hexadecimal.
4. The resulting block is prepended to the N-block message. The result is N+1 blocks.
5. The result of step 4 is hashed to create an n-bit digest. The digest is called the intermediate HMAC.
6. The intermediate n-bit HMAC is padded with 0's to make a b-bit block.
7. Steps 2 and 3 are repeated by a different constant **opad** (output pad). The value of opad is 5C in hexadecimal.
8. The result of step 7 is prepended to the block of step 6.
9. The result of step 8 is hashed with the same hashing algorithm to create the final n-bit HMAC.

## Cipher-based Message Authentication Code (CMAC)

- The method is same as cipher block chaining (CBC) mode.
- The idea is to create one block of MAC from N-bits of plaintext using a symmetric key cipher N times.

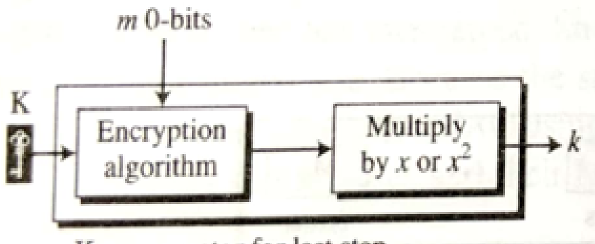


## CMAC - Process

- The message is divided into N blocks, each m bits long.
- The size of the CMAC is n bits.
- If the last block is not m bits, it is padded with a 1 bit followed by enough 0 bits to make it m bits.
- The first block of the message is encrypted with the symmetric key to create a new m bit block of encrypted data.
- This block is XORed with the next block and the result is encrypted again to create a new m bit block.
- The process continues until the last block and the result is encrypted.
- The n leftmost bit from the last block is the CMAC.
- In addition to symmetric key K, CMAC also uses another key k, which is applied only at the last step.

### Key Generation in CMAC

- The key is derived from the encryption algorithm with plaintext of m 0 bits using the cipher key, K.
- The result is then multiplied by x if no padding is applied and multiplied by if padding is applied.



Key-generator for last step

Lecture : 3

## Digital Signatures

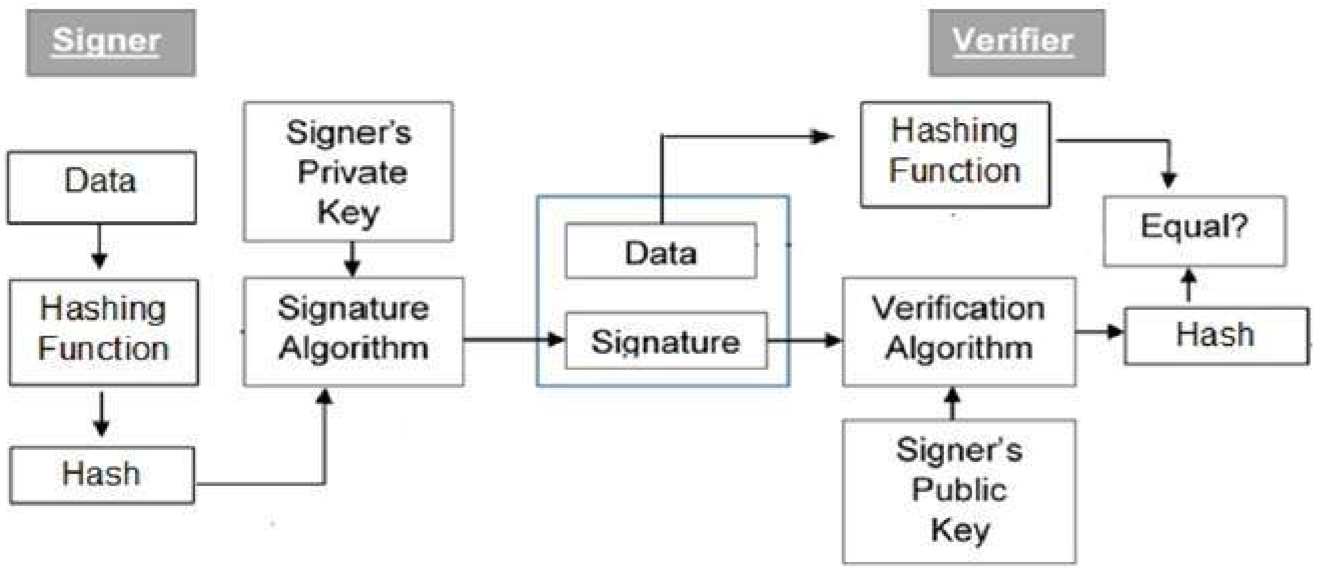
Digital signatures are the public-key primitives of message authentication. In the physical world, it is common to use handwritten signatures on handwritten or typed messages. They are used to bind signatory to the message.

Similarly, a digital signature is a technique that binds a person/entity to the digital data. This binding can be independently verified by receiver as well as any third party.

Digital signature is a cryptographic value that is calculated from the data and a secret key known only by the signer.

In real world, the receiver of message needs assurance that the message belongs to the sender and he should not be able to repudiate the origination of that message. This requirement is very crucial in business applications, since likelihood of a dispute over exchanged data is very high.

As mentioned earlier, the digital signature scheme is based on public key cryptography. The model of digital signature scheme.



The following points explain the entire process in detail:

- Each person adopting this scheme has a public-private key pair.
- Generally, the key pairs used for encryption/decryption and signing/verifying are different. The private key used for signing is referred to as the signature key and the public key as the verification key.
- Signer feeds data to the hash function and generates hash of data.

- Hash value and signature key are then fed to the signature algorithm which produces the digital signature on given hash. Signature is appended to the data and then both are sent to the verifier.
- Verifier feeds the digital signature and the verification key into the verification algorithm. The verification algorithm gives some value as output.
- Verifier also runs same hash function on received data to generate hash value.
- For verification, this hash value and output of verification algorithm are compared. Based on the comparison result, verifier decides whether the digital signature is valid.
- Since digital signature is created by 'private' key of signer and no one else can have this key; the signer cannot repudiate signing the data in future.

## Importance of Digital Signature

Out of all cryptographic primitives, the digital signature using public key cryptography is considered as very important and useful tool to achieve information security.

Apart from ability to provide non-repudiation of message, the digital signature also provides message authentication and data integrity. Let us briefly see how this is achieved by the digital signature

- **Message authentication** − When the verifier validates the digital signature using public key of a sender, he is assured that signature has been created only by sender who possess the corresponding secret private key and no one else.
- **Data Integrity** − In case an attacker has access to the data and modifies it, the digital signature verification at receiver end fails. The hash of modified data and the output provided by the verification algorithm will not match. Hence, receiver can safely deny the message assuming that data integrity has been breached.
- **Non-repudiation** − Since it is assumed that only the signer has the knowledge of the signature key, he can only create unique signature on a given data. Thus the receiver can present data and the digital signature to a third party as evidence if any dispute arises in the future.

By adding public-key encryption to digital signature scheme, we can create a cryptosystem that can provide the four essential elements of security namely − Privacy, Authentication, Integrity, and Non-repudiation.

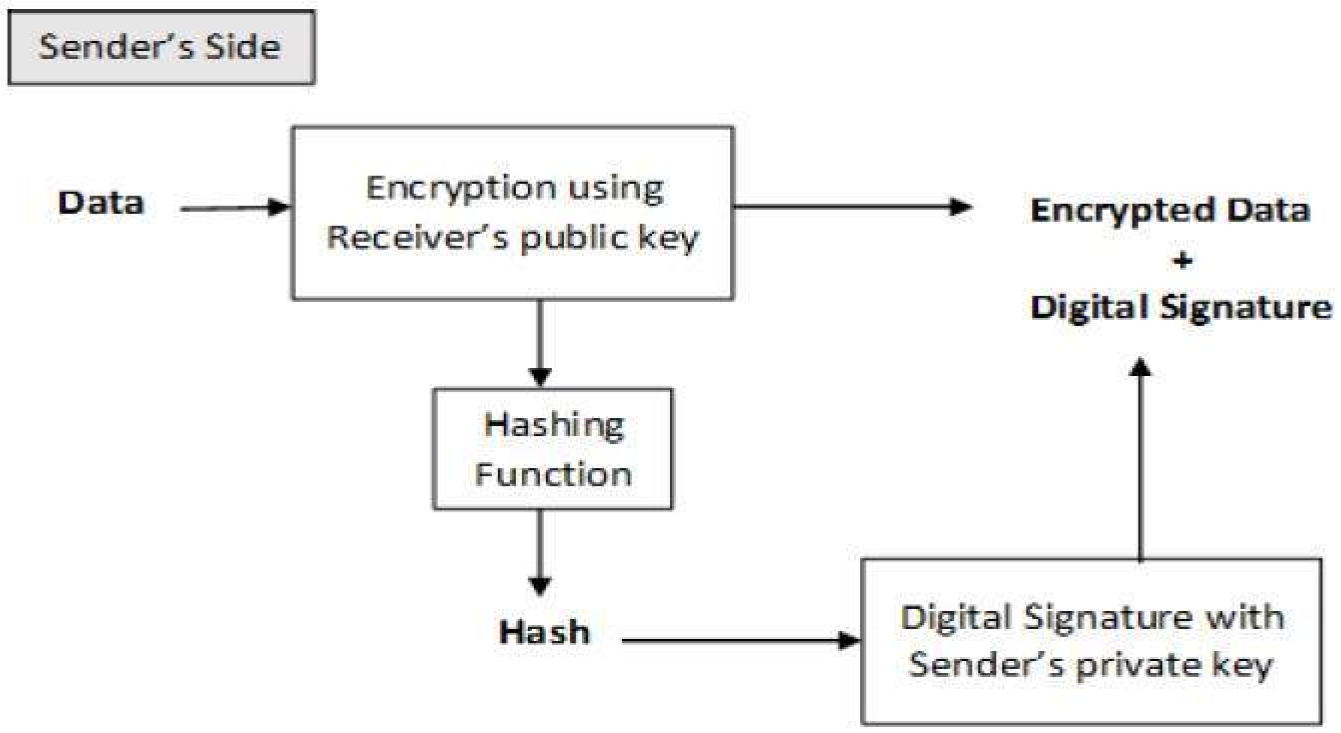
## Encryption with Digital Signature

In many digital communications, it is desirable to exchange an encrypted messages than plaintext to achieve confidentiality. In public key encryption scheme, a public (encryption) key of sender is available in open domain, and hence anyone can spoof his identity and send any encrypted message to the receiver.

This makes it essential for users employing PKC for encryption to seek digital signatures along with encrypted data to be assured of message authentication and non-repudiation.

This can archived by combining digital signatures with encryption scheme. Let us briefly discuss how to achieve this requirement. There are **two possibilities, sign-then-encrypt** and **encrypt-then-sign**.

However, the crypto system based on sign-then-encrypt can be exploited by receiver to spoof identity of sender and sent that data to third party. Hence, this method is not preferred. The process of encrypt-then-sign is more reliable and widely adopted.


Sender's Side
Data
Encryption using Receiver's public key
Encrypted Data + Digital Signature
Hashing Function
Hash
Digital Signature with Sender's private key

The receiver after receiving the encrypted data and signature on it, first verifies the signature using sender's public key. After ensuring the validity of the signature, he then retrieves the data through decryption using his private key.

Lecture : 4

## Digital Certificates

Digital Certificates are a standard of security for establishing an encrypted link between a server and a client. Generally, between a mail server or a webserver, which protects data in transitions by encrypting them. A Digital Certificate is also a Digital ID or a passport which is issued by a Third Party Authority which verifies the identity of the server's owner and not claiming a false identity.

## Components of a Digital Certificate

All these following components can be found in the certificate details:

- **Serial Number** − Used to uniquely identify the certificate.
- **Subject** − The person, or entity identified.
- **Signature Algorithm** − The algorithm used to create the signature.
- **Signature** − The actual signature to verify that it came from the issuer.
- **Issuer** − The entity that verified the information and issued the certificate.
- **Valid-From** − The date the certificate is first valid from.
- **Valid-To** − The expiration date.
- **Key-Usage** − Purpose of the public key (For example: encipherment, signature, certificate signing...).
- **Public Key** − The public key.
- **Thumbprint Algorithm** − The algorithm used to hash the public key certificate.
- **Thumbprint** − The hash itself, used as an abbreviated form of the public key certificate.

## Levels of Validations

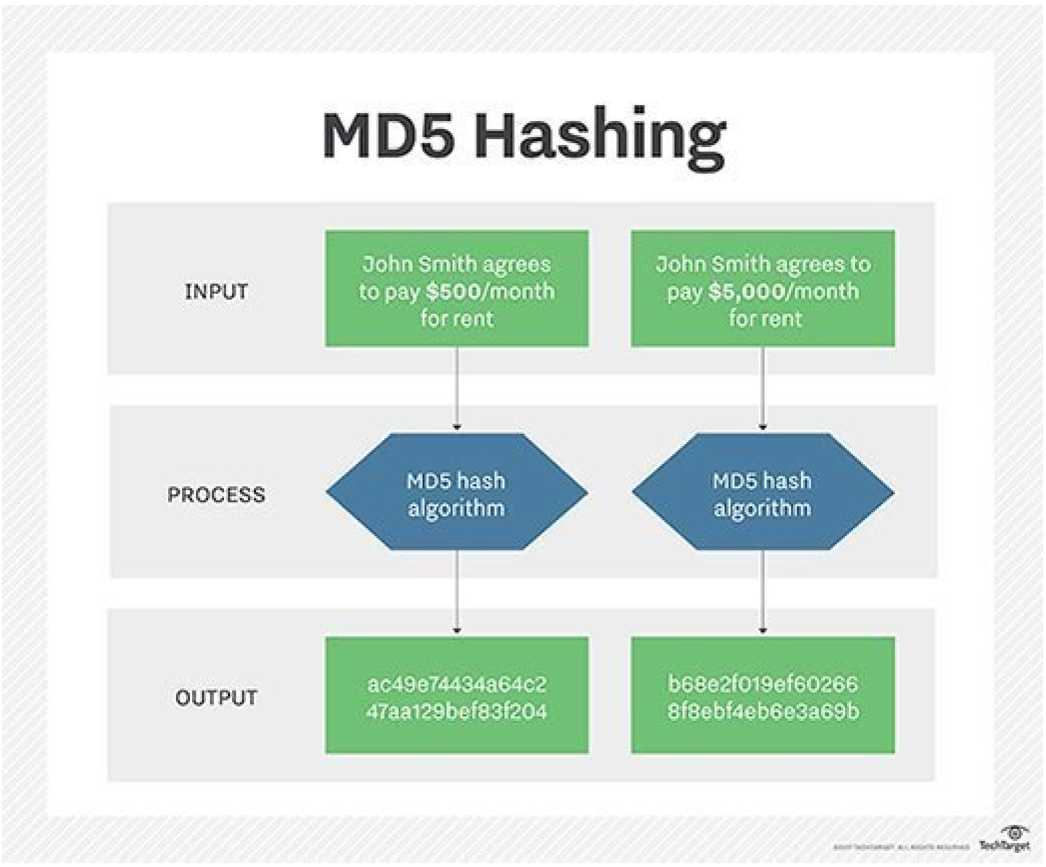
- **Domain Validation SSL Certificate** − It validates the domain that is registered by a system administrators and they have administrator rights to approve the certificate request, this validation generally is done by email request or by a DNS record.
- **Organization Validated SSL Certificates** − It validates the domain ownership and also the business information like the Official Name, City, Country, etc. This validation is done by email or DNS record entering and the certificate authority would also need some genuine documents to verify the Identity.
- **Extended Validation SSL Certificates** − It validates domain ownership and organization information, plus the legal existence of the organization. It also validates that the organization is aware of the SSL certificate request and approves it. The validation requires documentation to certify the company identity plus a set of additional steps and checks. The Extended Validation SSL Certificates are generally identified with a green address bar in the browser containing the company name.

Lecture :5

MD5

The MD5 (message-digest algorithm) hashing algorithm is a one-way cryptographic function that accepts a message of any length as input and returns as output a fixed-length digest value to be used for authenticating the original message.

The MD5 hash function was originally designed for use as a secure cryptographic hash algorithm for authenticating digital signatures. But MD5 has been deprecated for uses other than as a noncryptographic checksum to verify data integrity and detect unintentional data corruption.

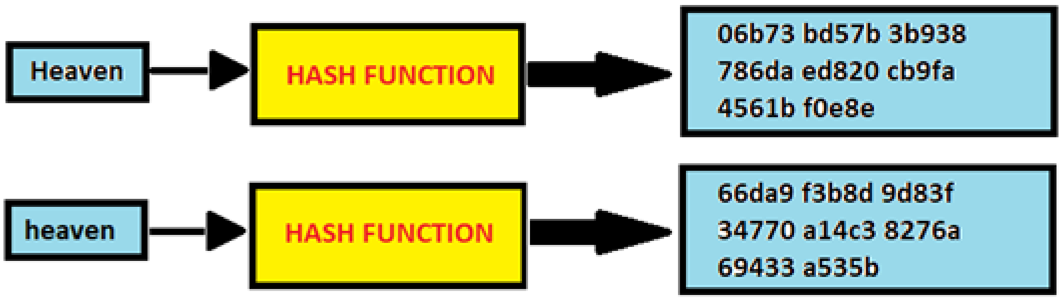

MD5 Hashing
INPUT
John Smith agrees to pay $500/month for rent
John Smith agrees to pay $5,000/month for rent
PROCESS
MD5 hash algorithm
MD5 hash algorithm
OUTPUT
ac49e74434a64c2 47aa129bef83f204
b68e2f019ef60266 8f8ebf4eb6e3a69b
TechTarget

**Steps**

1. Pad message so its length is 448 mod 512 (Message is 64 bit less than the multiple of 512)
2. Append a 64-bit length value to message (1 is appended followed by many zeros)
3. Initialise 4-word (128-bit) MD buffer (A,B,C,D) each of 32bits
4. Process message in 16-word (512-bit) blocks:
    - using **4 rounds** of 16 bit operations on message block & buffer
    - add output to buffer input to form new buffer value
5. Output hash value is the final buffer value

## SHA-1

SHA stands for secure hashing algorithm. SHA is a modified version of MD5 and used for hashing data and **certificates**. A hashing algorithm shortens the input data into a smaller form that cannot be understood by using bitwise operations, modular additions, and compression functions. You may be wondering, can hashing be cracked or decrypted? Hashing is similar to **encryption**, the only difference between hashing and encryption is that hashing is one-way, meaning once the data is hashed, the resulting hash digest cannot be cracked, unless a brute force attack is used. See the image below for the working of SHA algorithm. SHA works in such a way even if a single character of the message changed, then it will generate a different hash. For example, hashing of two similar, but different messages i.e., Heaven and heaven is different. However, there is only a difference of a capital and small letter.



**Secure Hash Algorithm 1 (SHA-1).**
Secure Hash Algorithm 1 (SHA-1). Developed by the U.S. government in the 1990s, SHA-1 used techniques like those of MD5 in the design of message-digest algorithms. But SHA-1 generated more secure 160-bit values when compared to MD5's 128-bit hash value lengths. Despite this, SHA-1 had some weaknesses and did not prove to be the ultimate algorithmic methodology for encryption, either. Security concerns began to emerge, prompting companies like Microsoft to discontinue support for SHA-1 in its software.

## References:

1. Behrouz Forouzan, “Cryptography & Network Security”

2. Charles P. Pfleeger,” Security in Computing “, Pearson Education

3.https://www.tutorialspoint.com/cryptography/cryptography_digital_signatures.htm#:~:text=Digital%

20signatures%20are%20the%20public,entity%20to%20the%20digital%20data.

4. https://www.tutorialspoint.com/internet_security/internet_security_certificates.htm

**Short Answer Questions:**

1. What is a Hash function?
2. Explain MAC & HMAC
3. Explain Message Digest in detail
4. Explain Digital Signature in detail
5. Explain the difference between Digital Signature and Digital Certificate
6. When a hash function is used to provide message authentication, the hash function value is referred to as
   a) Message Field
   b) Message Digest

c) Message Score
d) Message Leap

7. What is the full-form of CMAC?
   a) Code-based MAC
   b) Cipher-based MAC
   c) Construct-based MAC
   d) Collective-based MAC
8. What are the applications of Hash functions?

**Long Answer Questions:**

1. Explain Hash function, its properties and applications
2. How does Hash functions achieve Message Integrity & Authentication?
3. Explain MDC and MAC in detail
4. Explain MAC, Nested MAC and HMAC in detail
5. Explain CMAC
6. What is Digital Signature? How does it achieve Message Integrity, Confidentiality and Non-repudiation
7. What are Digital Certificates? Explain the process in detail.
8. What is the difference between Digital Signature & Certificates?

---------------------------------------------***-------------------------------------------------