MODULE 2
BASIC SECURITY SERVICES

In present day scenario security of the system is the sole priority of any organisation. The main aim of any organisation is to protect their data from attackers. In cryptography, attacks are of two types such as Passive attacks and Active attacks.

- A useful means of classifying security attacks is in terms of passive attacks and activeattacks.
- A passive attack attempts to learn or make use of information from the system butdoes not affect system resources.
- An active attack attempts to alter system resources or affect their operation

Passive Attacks: Passive attacks are in the nature of eavesdropping on, or monitoring of, transmissions. The goal of the opponent is to obtain information that is being transmitted. Passive attacks are very difficult to detect, because they do not involve any alterationof the data. However, it is feasible to prevent the success of these attacks, usually bymeans of encryption

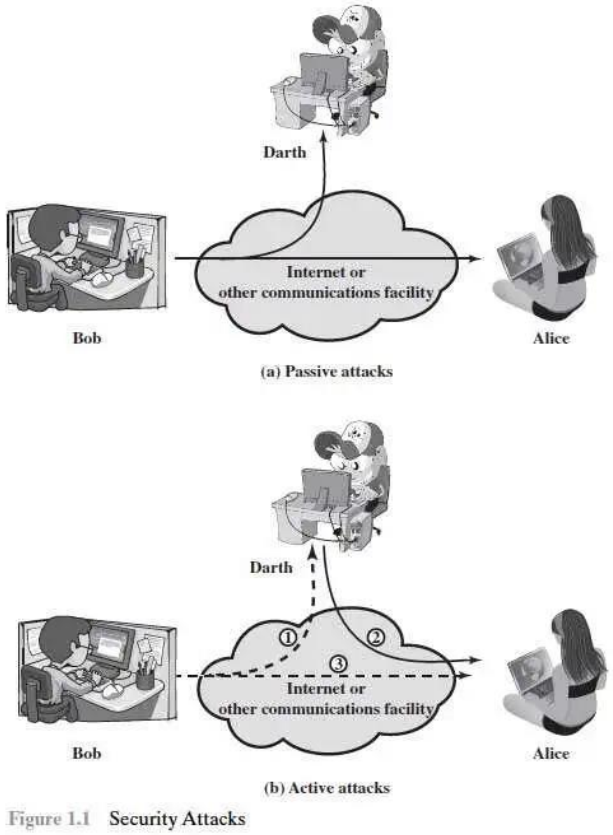


Figure 1.1 Security Attacks

Active Attack: Active attacks are a type of cybersecurity attack in which an attacker attempts to alter, destroy, or disrupt the normal operation of a system or network. Active attacks involve the attacker taking direct action against the target system or network, and can be more dangerous than passive attacks.

SECURITY SERVICES:

The classification of security services are as follows:

- Confidentiality: Ensures that the information in a computer system and transmitted information are accessible only for reading by authorized parties.

- Integrity: Ensures that only authorized parties are able to modify computer system assets and transmitted information. Modification includes writing, changing status, deleting, creating and delaying or replaying of transmitted messages.

- Non repudiation: Requires that neither the sender nor the receiver of a message be able to deny the transmission.

- Authentication: Ensures that the origin of a message or electronic document is correctly identified, with an assurance that the identity is not false.

- Availability: Requires that computer system assets be available to authorized parties when needed.

1) **Data Confidentiality:** An attacker should not be able to read the transmitted data or extract data in case of encrypted data. In short, confidentiality is the protection of transmitted data from passive attacks. One of the best ways to protect data confidentiality is encryption. encryption is a process that uses an algorithm to turn data into an unreadable format. Only authorised people can decrypt the data and read it.

CLASSICAL ENCRYPTION TECHNIQUES: There are two basic building blocks of all encryption techniques: substitution and transposition.

SUBSTITUTION TECHNIQUES: A substitution technique is one in which the letters of plaintext are replaced by other letters or by numbers or symbols. If the plaintext is viewed as a sequence of bits, then substitution involves replacing plaintext bit patterns with cipher text bit patterns.

- **Caesar cipher (or) shift cipher :**
  The earliest known use of a substitution cipher and the simplest was by Julius Caesar. The Caesar cipher involves replacing each letter of the alphabet with the letter standing 3 places further down the alphabet.

  e.g: plain text : pay more money
  Cipher text: SDB PRUH PRQHB

  Note that the alphabet is wrapped around, so that letter following “z‟ is “a‟.

  For each plaintext letter p, substitute the cipher text letter c such that

  $C = E(p) = (p+3) \bmod 26$

A shift may be any amount, so that general Caesar algorithm is

C = E (p) = (p+k) mod 26

Where k takes on a value in the range 1 to 25.
The decryption algorithm is simply P = D(C) = (C-k) mod 26

- **Monoalphabetic Substitution Cipher:** Instead of shifting alphabets by fixed amount as in Caesar cipher, any random permutation is assigned to the alphabets. This type of encryption is called monoalphabetic substitution cipher. For example, A is replaced by Q, B by D, C by T etc. then it will be comparatively stronger than Caesar cipher. The number of alternative keys possible now becomes 26!. Thus, Brute Force attack is impractical in this case. However, another attack is possible. Human languages are redundant i.e. certain characters are used more frequently than others. This fact can be exploited. In English 'e' is the most common letter followed by 't', 'r', 'n', 'o', 'a' etc. Letters like 'q', 'x', 'j' are less frequently used. Moreover, digrams like 'th' and trigrams like 'the' are also more frequent. These can be used to guess the plaintext if the plaintext is in uncompressed English language. The most common two letter combinations are called as digrams. e.g. th, in, er, re and an. The most common three letter combinations are called as trigrams. e.g. the, ing, and, and ion

Polyalphabetic Substitution ciphers: A polyalphabetic cipher is a substitution, using multiple substitution alphabets.

- ## Vigenere Cipher

Vigenere Cipher is a method of encrypting alphabetic text. It uses a simple form of **polyalphabetic substitution**. A polyalphabetic cipher is any cipher based on substitution, using multiple substitution alphabets. The encryption of the original text is done using the***Vigenère square or Vigenère table***.

The table consists of the alphabets written out 26 times in different rows, each alphabet shifted cyclically to the left compared to the previous alphabet, corresponding to the 26 possible.Caesar Ciphers

At different points in the encryption process, the cipher uses a different alphabet from one of the rows.

The alphabet used at each point depends on a repeating keyword.

Example:
Input : Plaintext : GEEKSFORGEEKS

Keyword : AYUSHNon repudiation:

Output : Ciphertext : GCYCZFMLYLEIM

For generating key, the given keyword is repeated in a circular manner until it matches the length of

the plain text. The keyword "AYUSH" generates the key "AYUSHAYUSHAYU" The plain text is then encrypted using the process explained below.

- **One-time pad cipher**

  One-time pad cipher is a type of Vignere cipher which includes the following features −

  - It is an unbreakable cipher.
  - The key is exactly same as the length of message which is encrypted.
  - The key is made up of random symbols.
  - As the name suggests, key is used one time only and never used again for any other message to be encrypted.
  - Due to this, encrypted message will be vulnerable to attack for a cryptanalyst. The key used for a one-time pad cipher is called **pad**, as it is printed on pads of paper.

Why is it Unbreakable?

The key is unbreakable owing to the following features −

- The key is as long as the given message.
- The key is truly random and specially auto-generated.
- Each key should be used once and destroyed by both sender and receiver.

- **Playfair cipher:** The best known multiple letter encryption cipher is the playfair, which treats digrams in the plaintext as single units and translates these units into cipher text digrams. The playfair algorithm is based on the use of 5x5 matrix of letters constructed using a keyword. Let the keyword be **"monarchy".** The matrix is constructed by filling in the letters of the keyword (minus duplicates) from left to right and from top to bottom, and then filling in the remainder of the matrix with the remaining letters in alphabetical order. The letter "i" and "j" count as one letter.

M	O	N	A	R
C	H	Y	B	D
E	F	G	I/J	K
L	P	Q	S	T
U	V	W	X	Z

Plaintext is encrypted two letters at a time According to the following rules:

- Repeating plaintext letters that would fall in the same pair are separated with a Filler letter such as "x".

- Plaintext letters that fall in the same row of the matrix are each replaced by the letter to the right, with the first element of the row following the last.
- Plaintext letters that fall in the same column are replaced by the letter beneath, with the top element of the column following the last.
- Otherwise, each plaintext letter is replaced by the letter that lies in its own row And the column occupied by the other plaintext letter.
- Plaintext = meet me at the school house Splitting two letters as a unit => me et me at th es ch o x ol ho us ex
- Corresponding cipher text => CL KL CL RS PD IL HY AV MP HF XL IU .

- **Hill cipher :** The Hill cipher is a polygraphic substitution cipher based on linear algebra. developed by the mathematician Lester S. Hill. It was the first polygraphic cipher in which it was practical to operate on more than three symbols at once.

The relationship between a block of plaintext and its ciphertext is expressed by

Encryption: $C = K P \text{ mo26}$

Decryption: $P = K^{-1}C \text{ mod } 26$

$$c_1 = p_1 k_{11} + p_2 k_{21} + \ldots + p_m k_{m1} \bmod 26$$
$$c_2 = p_1 k_{12} + p_2 k_{22} + \ldots + p_m k_{m2} \bmod 26$$
$$\ldots$$
$$\ldots$$
$$c_m = p_1 k_{1m} + p_2 k_{2m} + \ldots + p_m k_{mm} \bmod 26$$

Hill Cipher source link

- https://www.youtube.com/watch?v=sYdq6-kquKc
- https://www.educative.io/edpresso/what-is-the-hill-cipher

Transposition Techniques:

Transposition techniques, cipher text is obtained by changing position of letters in the plaintext, that is performing permutation on the plaintext letters. For decryption the reverse permutation is performed.

Rail fence technique:

In **rail fence** technique, the plaintext is written down as a sequence of diagonals and then read off as a sequence of rows.

Example: To encipher the message "meet me after the toga party= with a rail fence of depth 2, we write the following:

Plaintext: meet me after the toga party

M		E		M		A		T		R		H		T		G		P		R		Y
	E		T		E		F		E		T		E		O		A		A		T	

Cipher text : mematrhtgpryetefeteoaat

A more complex scheme is to write the plaintext in a rectangle, row by row, and read column by column in the order provided by the key to obtain the cipher text. The order of reading columns is the key to the algorithm.

Example:

```
Key:          4 3 1 2 5 6 7
Plaintext:    a t t a c k p
              o s t p o n e
              d u n t i l t
              w o a m x y z
Ciphertext:   TTNAAPTMTSUOAODWCOIXKNLYPETZ
```

In the above example, the key is 4312567. To encrypt, start with the column that is labelled 1, in this case column 3. Write down all the letters in that column. Proceed to column 4, which is labelled 2, then column 2, then column 1, then columns 5, 6, and 7.

2. **Data Integrity:** Make sure that the message received was exactly the message the sender sent.Integrity revolves around ensuring that information remains unaltered and free from tampering. Cryptographic techniques, like hash functions, play a crucial role in guaranteeing data integrity by offering a mechanism to detect any unauthorized modifications to the data.

Cryptographic Hash

INTRODUCTION

Definition: A hash function is a deterministic function that maps an input element from a larger (possibly infinite) set to an output element in a much smaller set.

The input element is mapped to a ***hash value.***

For example, in a district-level database of residents of that district, an individual's record may be mapped to one of 26 hash buckets.

Each hash bucket is labelled by a distinct alphabet corresponding to the first alphabet of a person's name.

Given a person's name (the input), the output or hash value is simply the first letter of that name (Fig. 7.1).

Hashes are often used to speed up insertion, deletion, and querying of databases.

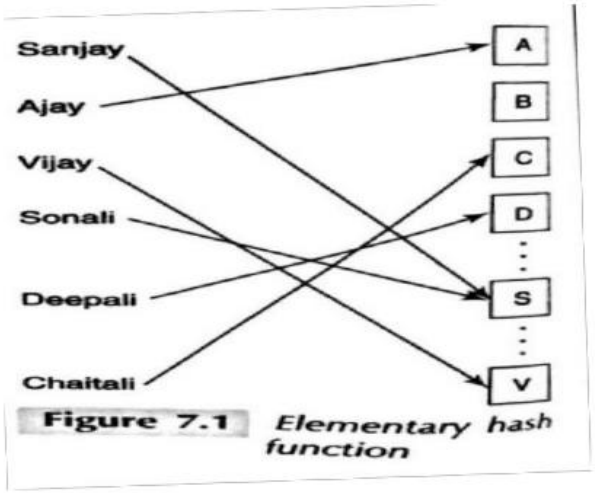


Figure 7.1 *Elementary hash function*

In the example above, two names beginning with the same alphabet map to the same hash bucket and result in a collision.

Proppertics:

A cryptographic hash function, ***h(x),*** maps a binary string of arbitrary length to a fixed length binary string.

The properties of *h* are as follows:

1. **One-way property.** Given a hash value, ***y*** (belonging to the range of the hash function), it is computationally infeasible to find an input x such that ***b(x) = y***
2. ***Weak collision resistance.*** Given an input value x1, it is computationally infeasible to find another input value x2 such that ***h(x1) = h(x2)***
3. **Strong collision *resistance.*** It is computationally infeasible to find two input values x1 and no x2 such that h(x1)=h(x2)
4. ***Confusion + diffusion.*** If a single bit in the input string is flipped, then each bit of the hash value is flipped with probability roughly equal to 0.5.

CONSTRUCTION

Generic cryptographic hash:

The input to a cryptographic hash function is often a message or document.

To accommodate inputs of arbitrary length, most hash functions (including the commonly used MD-5 and SHA-1) use iterative construction as shown in Fig. 7.5.

C is a compression box.

It accepts two binary strings of lengths **b and w** and produces an output string of **length w.**

Here, **b is the block size and w is the width of the digest**.

During the first iteration, it accepts a pre-defined initialization vector (IV), while the top input is the first block of the message.

In subsequent iterations, the ***"partial hash output" is fed back*** as the second input to the C-box.

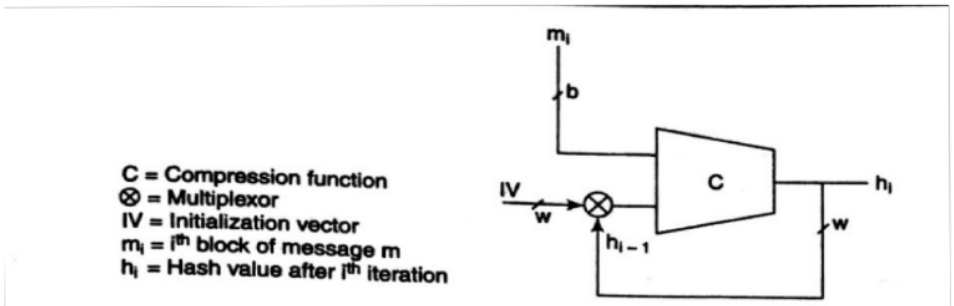
The top input is derived from successive blocks of the message.

This is repeated until all the blocks of the message have been processed.

The above operation is summarized below:

$h_1 = C (IV, m_1)$ for first block of message

$h_i = C (h_{i-1}.m_i)$ for all subsequent blocks of the message



7.5 *Iterative construction of cryptographic hash*

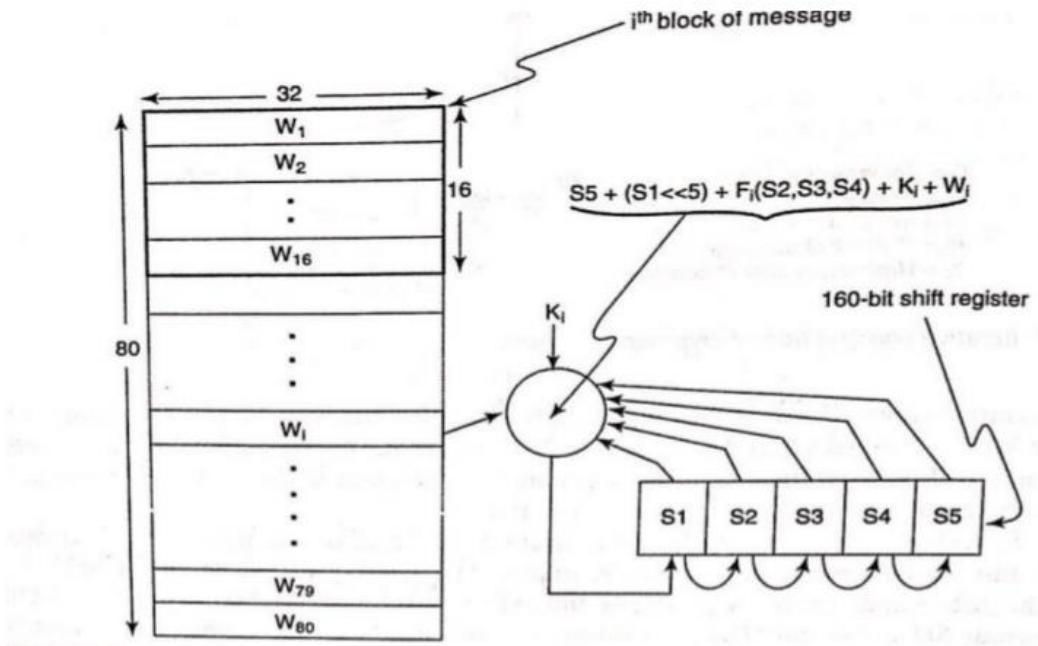
The above iterative construction of the cryptographic hash function is a simplified version of that proposed by **Merkle and Damgard.**

It has the property that if the compression function is collision-resultant, then the resulting hash function is also collision-resultant.

MD-5 and SHA-1 are the best known examples. MD-5 is a 128-bit hash, while SHA-1 is a 160-bit hash.

2 Case Study: SHA-1

SHA-1 uses the iterative hash construction of Fig. 7.5.



gure 7.6 *Computation of SHA-1*

```
initialize the shift register, S1 S2 S3 S4 S5
for each block of the (message + pad + length field) {
        create the 80-word array [using Eq. (7.2)]
        for i = 1 to 80 {
                temp ← S5 + (S1 << 5) + Fi(S2, S3, S4) + Ki + Wi
                S5 ← S4
                S4 ← S3
                S3 ← S2 >> 2
                S2 ← S1
                S1 ← temp
        }
}
```

$$
\begin{aligned}
F_i(S2, S3, S4) &= (S2 \wedge S3) \vee (\neg S2 \wedge S4), && 1 \le i \le 20 \\
F_i(S2, S3, S4) &= S2 \oplus S3 \oplus S4, && 21 \le i \le 40 \\
F_i(S2, S3, S4) &= (S2 \wedge S3) \vee (S2 \wedge S4) \vee (S3 \wedge S4), && 41 \le i \le 60 \\
F_i(S2, S3, S4) &= S2 \oplus S3 \oplus S4 && 61 \le i \le 80
\end{aligned}
$$

- The message is split into blocks of ***size 512 bits***.
- The length of the message, expressed in binary as a 64 bit number, is appended to the message.
- Between the end of the message and the length field, a pad is inserted so that the length of the **(message + pad + 64)** is a ***multiple of 512,*** the block size.
- The pad has the form: 1 followed by the required number of 0's.

Array Initialization

- Each block is split into 16 words, each 32 bits wide.
- These **16 words** populate the first 16 positions, W1, W2 ……W16, of an array of **80 words.**
- The remaining **64 words** are obtained from :

$$W_i = W_{i-3} \oplus W_{i-8} \oplus W_{i-14} \oplus W_{i-16} \quad 16 < i \leq 80$$

- This array of words is shown in Fig. 7.6.

Hash Computation in SHA 1

- A 160-bit shift register is used to compute the intermediate hash values (Fig. 7.6).
- It is initialized to a fixed pre-determined value at the start of the hash computation.
- We use the notation S1, S2, S3, S4, and S5 to denote the five 32 -bit words making up the shift register.
- The bits of the shift register are then mangled together with each of the words of the array in turn.
- The mangling is achieved using a combination of the following Boolean operations: **+, v, ~, ^, XOR ROTATE.**

APPLICATIONS

Hash-based MAC:

MAC

➢ MAC is used as a message integrity check as well as to provide message authentication.

➢ It makes use of a common shared secret, k, between two communicating parties.

➢ The hash-based MAC that we now introduce is an alternative to the CBC-MAC.

➢ The cryptographic hash applied on a message creates a digest or digital fingerprint of that message.Non repudiation:

➢ Suppose that a sender and receiver share a secret, k.

➢ If the message and secret are concatenated and a hash taken on this string, then the hash value becomes a fingerprint of the combination of the message, m and the secret, k.

➢ MAC = h (m|| k)

➢ The MAC is much more than just a checksum on a message.

➢ It is computed by the sender, appended to the message, and sent across to the receiver.

On receipt of the message + MAC, the receiver performs the computation using the common secret and the received message.

➢ It checks to see whether the MAC computed by it matches the received MAC.

➢ A change of even a single bit in the message or MAC will result in a mismatch between the computed MAC and the received MAC.

➢ In the event of a match, the receiver concludes the following:

➢ (a) The sender of the message is the same entity it shares the secret with — thus the MAC provides source authentication.

➢ (b) The message has not been corrupted or tampered with in transit — thus the MAC provides verification of message integrity.

➢ Drawbacks:

➢ An attacker might obtain one or more message—MAC pairs in an attempt to determine the MAC secret.

➢ First, if the hash function is one-way, then it is not feasible for an attacker to deduce the input to the hash function that generated the MAC and thus recover the secret.

➢ If the hash function is collision-resistant, then it is virtually impossible for an attacker to suitably modify a message so that the modified message and the original both map to the same MAC value.

HMAC

➢ There are other ways of computing the hash MAC other than this method using HMAC .

➢ Another possibility is to use key itself as the Initialization Vector (IV) instead of concatenating it with the message.

➢ Bellare, Canetti, and Krawczyk proposed the HMAC and showed that their scheme is re against a number of subtle attacks on the simple hash-based MAC.

➢ Figure 7.7 shows how an HMAC is computed given a key and a message. The key is padded with O's (if necessary) to form a 64-byte string denoted K' and XORed with a constant (denoted IPAD).

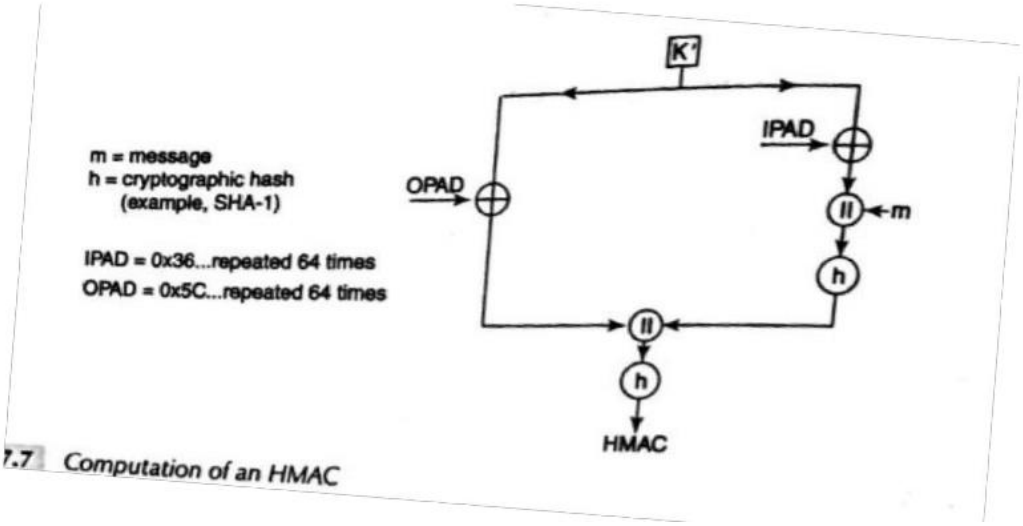
➢ It is then concatenated with the message and a hash is performed on the result.

➢ K' is also XORed with another constant (denoted OPAD) after which it is prepended to the output of the first hash.

➢ Once again hash is then computed to yield the HMAC.

➢ As shown in Fig. 7.7, HMAC performs an extra hash computation but provides greatly enhanced security.

3) Non



7.7 Computation of an HMAC

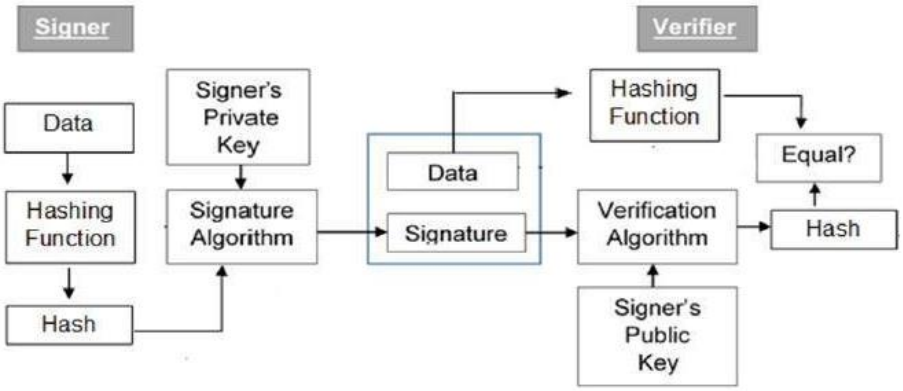
repudiation: Non-repudiation is the assurance that someone cannot deny the validity of something. Non-repudiation is a legal concept that's widely used in information security and refers to a service, which provides proof of the origin and integrity of data. In other words, non-repudiation makes it very difficult to successfully deny who/where a message comes from, as well as the authenticity and integrity of that message. Nonrepudiation is achieved through cryptographic technique like digital signatures.

Digital Signature: 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.

Model of Digital Signature As mentioned earlier, the digital signature scheme is based on public key cryptography. The model of digital signature scheme is depicted in the following illustration −

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.**

It should be noticed that instead of signing data directly by signing algorithm, usually a hash of data is created. Since the hash of data is a unique representation of data, it is sufficient to sign the hash in place of data. The most important reason of using hash instead of data directly for signing is efficiency of the scheme.

4) Availability: Availability guarantees that systems, applications and data are available to users when they need them. The most common attack that impacts availability is denial-of-service in which the attacker interrupts access to information, system, devices or other network resources.

Denial of Service (DoS) is a cyber-attack on an individual Computer or Website with the intent to deny services to intended users. Their purpose is to disrupt an organization's network operations by denying access to its users. Denial of service is typically accomplished by flooding the targeted machine or resource with surplus requests in an attempt to overload systems and prevent some or all legitimate requests from being fulfilled. For example, if a bank website can handle 10 people a second by clicking the Login button, an attacker only has to send 10 fake requests per second to make it so no legitimate users can log in. DoS attacks exploit various weaknesses in computer network technologies. They may target servers, network routers, or network communication links. They can cause computers and routers to crash and links to bog down. The most famous DoS technique is the Ping of Death. The Ping of Death attack works by generating and sending special network messages that cause problems for systems that

receive them. In the early days of the Web, this attack could cause unprotected Internet servers to crash quickly.

Other basic types of DoS attacks involve.

- Flooding a network with useless activity so that genuine traffic cannot get through.
- Remotely overloading a system's CPU so that valid requests cannot be processed.
- Changing permissions or breaking authorization logic to prevent users from logging into a system. One common example involves triggering a rapid series of false login attempts that lockout accounts from being able to log in.
- Deleting or interfering with specific critical applications or services to prevent their normal operation (even if the system and network overall are functional).

Another variant of the DoS is the Smurf attack. This involves emails with automatic responses. If someone emails hundreds of email messages with a fake return email address to hundreds of people in an organization with an autoresponder on in their email, the initially sent messages can become thousands sent to the fake email address. If that fake email address belongs to someone, this can overwhelm that person's account. DoS attacks can cause the following problems:

- Ineffective services
- Inaccessible services
- Interruption of network traffic
- Connection interference

How Do DoS Attacks Work?

DoS attacks typically exploit vulnerabilities in a target's network or computer systems. Attackers can use a variety of methods to generate overwhelming traffic or requests, including:

1. **Flooding the target with a massive amount of data**
2. **Sending repeated requests to a specific part of the system**
3. **Exploiting software vulnerabilities to crash the system**

Prevention Given that Denial of Service (DoS) attacks are becoming more frequent, it is a good time to review the basics and how we can fight back.

- **Cloud Mitigation Provider– Cloud mitigation providers are experts at providing DDoS mitigation from the cloud. This means they have built out massive amounts of network bandwidth and DDoS mitigation capacity at multiple sites around the Internet that can take in any type of network traffic, whether you use multiple ISP's, your own data**

center, or any number of cloud providers. They can scrub the traffic for you and only send "clean" traffic to your data center.

- **Firewall– This is the simplest and least effective method. Python scripts are often written to filter out malicious traffic, or existing firewalls can be utilized by enterprises to block such traffic.**
- **Internet Service Provider (ISP)– Some enterprises use their ISP to provide DDoS mitigation. These ISPs have more bandwidth than an enterprise would, which can help with large volumetric attacks.**

Features to help mitigate these attacks:

Network Segmentation: egmenting the network can help prevent a DoS attack from spreading throughout the entire network. This limits the impact of an attack and helps to isolate the affected systems.
Implement Firewalls: Firewalls can help prevent DoS attacks by blocking traffic from known malicious IP addresses or by limiting the amount of traffic allowed from a single source.
Use Intrusion Detection and Prevention Systems: Intrusion Detection and Prevention Systems (IDS/IPS) can help to detect and block DoS attacks by analyzing network traffic and blocking malicious traffic.
Limit Bandwidth: Implementing bandwidth limitations on incoming traffic can help prevent a DoS attack from overwhelming the network or server.
Implement Content Delivery Network (CDN): CDN can help to distribute traffic and reduce the impact of a DoS attack by distributing the load across multiple servers.
Use Anti-Malware Software:Anti-malware software can help to detect and prevent malware from being used in a DoS attack, such as botnets.
Perform Regular Network Scans:Regular network scans can help identify vulnerabilities and misconfigurations that can be exploited in a DoS attack. Patching these vulnerabilities can prevent a DoS attack from being successful.

5. privacy: Data privacy, also called information privacy, is an aspect of data protection that addresses the proper storage, access, retention, immutability and security of sensitive data.
Data privacy is typically associated with the proper handling of personal data or personally identifiable information (PII), such as names, addresses, Social Security numbers and credit card numbers. However, the idea also extends to other valuable or confidential data, including financial data, intellectual property and personal health information.

Key management is crucial to the security of any cryptosystem.

What is key management?

The scope of key management is perhaps best described as the secure administration of cryptographic keys. This is a deliberately broad definition, because key management involves a wide range of quite disparate processes, all of which must come together coherently if cryptographic keys are to be securely managed.

The key lifecycle

The key lifecycle, which identifies the various processes concerning cryptographic keys throughout their lifetime. The main phases of the key lifecycle are depicted in Figure 10.1.

Key generation concerns the creation of keys.

Key establishment is the process of making sure keys reach the end points where they will be used. This is arguably the most difficult phase of the key lifecycle to implement.

Key storage deals with the safekeeping of keys. It may also be important to conduct key backup so keys can be recovered in the event of loss of a key and, ultimately, key archival.

Key usage is about how keys are used. As part of this discussion, we will consider key change.

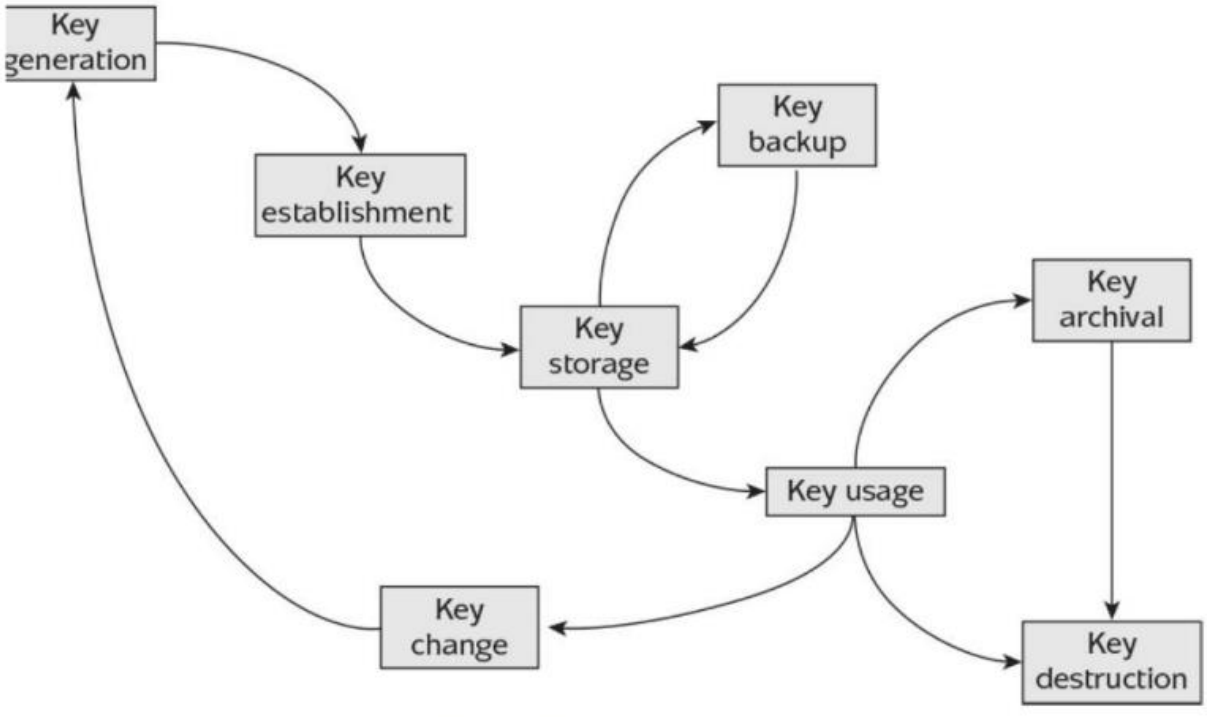


Figure 10.1: The key lifecycle.

Fundamental key management requirements

There are two fundamental key management requirements which apply throughout the various phases of the key lifecycle:

Secrecy of keys: Throughout the key lifecycle, secret keys (in other words, symmetric keys and private keys) must remain secret from all parties except those authorised to know them
Assurance of purpose of key: Throughout the key lifecycle, those parties relying on a key must Have assurance of purpose of the key. In other words, someone in possession of a key should be Confident they can use this key for the purpose they believe it to be for.