HASHDEEP:

Hashdeep is like a digital detective for file integrity. It's a command-line tool that calculates and verifies hash values (like checksums or fingerprints) for files and directories. This helps ensure the integrity of files, detecting any changes, corruption, or tampering. Hashdeep is commonly used for data verification, digital forensics, and security assessments.

To apply hash to file:

```
hashdeep -c md5,sha1,sha256,tiger filename
```

To make hashset file :

```
hashdeep -c md5,sha1,sha256 -r /home/shachi/myfiles > hashset1.txt
```

To audit the hashset file :

```
hashdeep -a -r -k hashset1.txt /home/shachi/myfiles
```

To get where it failed use the command with -v option :

```
hashdeep -v -a -r -k hashset1.txt /home/shachi/myfiles
```

To add a new file :

```
touch /home/shachi/myfiles/newfile.txt
```

To move a file :

```
mv /home/shachi/myfiles/example.txt /tmp
```

To rename a file :

```
mv /home/shachi/myfiles/shachi.txt /home/shachi/myfiles/shachi.bak
```

RECONNAISSANCE TOOLS :

Reconnaissance tools are like information gatherers for cybersecurity or network professionals. They help collect data about a target, which can be used for security assessments, network troubleshooting, or understanding the online presence of a domain or host. These tools are often the first step in security testing and vulnerability assessment.

Commands:

1. **whois**:
    - Whois is like a digital phonebook. It provides information about domain names, IP addresses, and their owners. You can find out details like the domain's registration information and the contact details of the domain owner.
2. **traceroute**:
    - Traceroute is like a map for network traffic. It shows the path that data packets take from your computer to a destination, revealing the network hops in between. It's useful for diagnosing network issues and assessing the route your data takes.
3. **dig** (Domain Information Groper):
    - Dig is like a data miner for DNS (Domain Name System). It helps you query DNS servers to obtain information about domain names, IP addresses, and DNS records. It's useful for troubleshooting DNS-related issues.
4. **nslookup** (Name Server Lookup):
    - Nslookup is like asking questions about DNS. It helps you interactively query DNS servers to obtain information about hostnames, IP addresses, and DNS records. It's useful for DNS troubleshooting and testing.
5. **nikto**:
    - Nikto is like a web server scanner. It's used to identify potential security issues in web servers by scanning for known vulnerabilities and misconfigurations. It's often used in web application security testing.
6. **dmitry**:

- Dmitry is like a Swiss Army knife for gathering information about a target. It can perform tasks such as banner grabbing, WHOIS queries, port scanning, and more, making it a versatile tool for reconnaissance and profiling of hosts and networks.

TCP DUMP:

TCPdump is like a surveillance camera for your network. It's a command-line tool that captures and analyzes network traffic. You can use it to monitor and understand the data flowing in and out of your computer or network, helping you troubleshoot network issues, investigate security incidents, and more.

Commands:

1. **tcpdump -n tcp**:
    - This command tells TCPdump to capture and display TCP packets. It shows you information about TCP traffic, such as source and destination IP addresses and ports.
2. **tcpdump -n src 192.168.0.243**:
    - With this command, you're telling TCPdump to capture packets where the source IP address is specifically 192.168.0.243. It's like saying, "Show me only the traffic coming from this specific place."
3. **tcpdump -n port 80**:
    - This command captures packets that are using port 80, which is often associated with web traffic (HTTP). It's like filtering for web-related data.
4. **tcpdump udp and src port 53**:
    - This command captures UDP packets where the source port is 53. Port 53 is typically associated with DNS traffic. So, it's like saying, "Show me DNS traffic from the source."
5. **tcpdump -n portrange 1-80**:
    - This command captures packets where the port number falls within the range of 1 to 80. It's a way to focus on a specific range of ports, which can help you analyze various types of network traffic within that range.

NMAP :

Nmap is like a detective tool for computer networks. It's used to explore and scan networks to find out what devices are there, what services they offer, and if there are any potential vulnerabilities. It's like creating a map of a city to see what buildings are there and what they do.

Commands:

1. **sudo wireshark** (To open Wireshark):
    - Wireshark is like an X-ray machine for network traffic. It lets you see what's going on in your network by capturing and analyzing the data packets.
2. **nmap -sS tsec.edu** (TCP SYN scan):
    - This is like knocking on doors to see if anyone is home. It sends TCP SYN packets to check which ports on the target system are open.
3. **nmap -sT tsec.edu** (TCP connect scan):
    - It's like ringing the doorbell to see if someone answers. Nmap establishes a full TCP connection to each port on the target to check if it's open.
4. **nmap -sU tsec.edu** (UDP scans):
    - UDP scans are like shouting questions to see if anyone responds. Nmap sends UDP packets to target ports to see if they respond.
5. **nmap -sN tsec.edu** (TCP NULL):
    - This is like trying to sneak in quietly. Nmap sends TCP packets with no flags set to probe for open ports.
6. **nmap -sF tsec.edu** (FIN scans):
    - It's like sending a goodbye message and waiting for a response. Nmap sends TCP FIN packets to see if ports are open.
7. **nmap -sX tsec.edu** (Xmas scans):
    - Xmas scans are like sending a flashy, mysterious package. Nmap sends packets with specific flags set to check for open ports.
8. **nmap -sA tsec.edu** (TCP ACK scan):

- It's like asking questions without expecting answers. Nmap sends TCP ACK packets to see how the target system responds.

9. **nmap -sO tsec.edu** (IP protocol scan):
    - This is like asking about the language someone speaks. Nmap probes for the protocols supported by the target.
10. **nmap -sV tsec.edu** (Version detection):
    - It's like checking a book's cover to see what's inside. Nmap tries to identify the version and type of services running on open ports.

sudo wireshark (To open wireshark)

nmap -sS tsec.edu (TCP SYN scan)

nmap -sT tsec.edu (TCP connect scan)

nmap -sU tsec.edu (UDP scans)

nmap -sN tsec.edu (TCP NULL)

nmap -sF tsec.edu (FIN scans)

nmap -sX tsec.edu (Xmas scans)

nmap -sA tsec.edu (TCP ACK scan)

nmap -sO tsec.edu (IP protocol scan)

nmap -sV tsec.edu (Version detection)

STUNFXAOV

DOS :

DOS (Denial of Service):

DOS is like a traffic jam for computer systems. It's when someone or something floods a website or network with too much traffic, causing it to slow down or even crash. The goal is to make a service unavailable to its intended users.

Types of DOS:

1. **Flood Attacks**: These send tons of data or requests to overwhelm the target. It's like too many cars trying to use a narrow road, causing a traffic jam. Two common types are:
    - **UDP Flood**: Overloads the target with UDP data.
    - **TCP Flood**: Overwhelms the target with TCP connection requests.
2. **Resource Exhaustion**: Attacks that use up all of a system's resources, like memory or CPU power. It's like using up all the gas in a car, so it can't run.
3. **Application Layer Attacks**: Target specific services or applications, like a website or email server, by sending malicious requests. It's like someone making endless, bogus reservations at a restaurant to fill up all the tables.
4. **Distributed Denial of Service (DDOS)**: Many computers (a botnet) attack a target together. It's like a massive group of people trying to enter a small store, making it impossible for others to get in.

DOS attacks can disrupt websites, online services, and networks, causing frustration and financial losses. Protecting against them is crucial for online security.

Syn flood :

```
hping3 -c 15000 -d 120 -S -w 64 -p 80 --flood --rand-source 192.168.1.159
```

ICMP flood:

```
hping3 -1 --flood -a 192.168.103 192.168.1.255
```

Command 1 (SYN Flood):

- The first command uses **hping3** to perform a SYN flood attack.

- It sends 15,000 packets (**-c 15000**) with 120 bytes of data (**-d 120**), using the TCP SYN flag (**-S**) and a TCP window size of 64 (**-w 64**) to the target's port 80 (**-p 80**).
- The **--flood** option sends packets as fast as possible, and **--rand-source** uses random source IP addresses.
- The target IP address is 192.168.1.159.

Command 2 (ICMP Flood):

- The second command uses **hping3** for an ICMP flood attack.
- It continuously sends ICMP echo requests (ping) to IP address 192.168.1.255 using the **-1** option and flooding mode with **-a** specifying the source IP address as 192.168.103.
- This attack aims to flood a network with ICMP traffic, potentially disrupting network operations.

IPTABLES FIREWALL :

Firewall: A firewall is like a security guard for your computer or network. It's a barrier that checks and controls incoming and outgoing data traffic based on a set of rules. Firewalls protect your system from unauthorized access, cyberattacks, and harmful data. They act as a filter, allowing safe traffic to pass and blocking the bad stuff.

Types of Firewalls:

1. **Packet Filtering Firewall**: These check individual data packets to see if they meet specific criteria (like source and destination addresses) and decide whether to allow or block them.
2. **Stateful Firewall**: These are smarter than packet filters. They remember the state of active connections and make decisions based on the context of the traffic.
3. **Proxy Firewall**: These act as intermediaries between your computer and the internet. They receive and forward requests, hiding your system's details from potential threats.
4. **Application Layer Firewall (Proxy Firewall)**: They focus on specific applications or services (like web browsing). They inspect the data content of the traffic to ensure it's safe.

IPtables: IPtables is a powerful tool on Linux systems that helps in setting up and managing firewall rules. Think of it as a toolbox for creating rules to allow or block network traffic. It's like giving instructions to the firewall about who's allowed to enter your system and who's not. IPtables uses these rules to protect your computer from online threats and secure your network. It's like a bouncer for your server or computer, deciding who's on the guest list and who's not.

`sudo iptables -L` (List current filter rules)

`sudo iptables -A INPUT -p tcp --dport ssh -j ACCEPT`

`sudo iptables -I INPUT 1 -i lo -j ACCEPT`

1. **sudo iptables -L**:
    - This command is used to list the current iptables rules. It displays the existing rules for filtering network packets.

2. **sudo iptables -A INPUT -p tcp --dport ssh -j ACCEPT**:
    - This command appends a rule to the INPUT chain of the iptables firewall. It allows incoming TCP traffic on the SSH (Secure Shell) port (default is port 22) and accepts it.
    - **-A INPUT**: Appends the rule to the INPUT chain.
    - **-p tcp**: Specifies that the rule applies to TCP packets.
    - **--dport ssh**: Specifies the destination port (SSH port).
    - **-j ACCEPT**: Indicates that the action to take when a packet matches the rule is to accept it.
3. **sudo iptables -I INPUT 1 -i lo -j ACCEPT**:
    - This command inserts a rule at the top (position 1) of the INPUT chain, allowing all traffic on the loopback interface (lo).
    - **-I INPUT 1**: Inserts the rule at position 1 of the INPUT chain.
    - **-i lo**: Specifies that the rule applies to the loopback interface, which is used for local communication.
    - **-j ACCEPT**: Indicates that the action to take when a packet matches the rule is to accept it

GPG :

GPG is the OpenPGP part of the GNU Privacy Guard (GnuPG). It is a tool to provide digital encryption and signing services using the OpenPGP standard. gpg features complete key management and all the bells and whistles you would expect from a full OpenPGP implementation.

Step 1: Generate private key and public key pairs for sender and receiver using command

gpg --gen-key (repeat for sender and receiver)

Step 2: Create a file containing sender's and receiver public key which then can be sent to other users.

gpg --export -a username>filename

Step 3: Similarly create file containing sender's private key.

gpg --export-secret-key -a username>filename (for sender)

Step 4: You can create a fingerprint of key using the command

gpg --fingerprint receiver's_email (for receiver)

Step 7: Sender can sign the public key of receiver using command

gpg --sign-key receiver_email

Step 8: Encrypt the data to send. (create a file beforehand to be encrypted)

gpg --encrypt -r receiver_email name_of_file (only encrypt, .gpg file created)

cat > newfile.txt (To create a file)

Step 9: Decrypt the file

gpg -o myfiledecrypted -d myfile.txt.gpg

Playfair, Vigenere cipher:

How vigenere cipher works?(with eg)

The Vigenere Cipher is a method of encrypting and decrypting messages. It is a polyalphabetic substitution cipher, which means that it uses multiple cipher alphabets to encrypt the plaintext.

The Vigenere Cipher operates by using a keyword, typically a word or phrase, as the basis for encryption. The keyword is repeated to match the length of the plaintext message. Each letter in the keyword is then used to determine the shift value for the corresponding letter in the plaintext.

To encrypt a message, each letter of the plaintext is shifted by the corresponding letter in the keyword. Here the plaintext is divided into group of n characters each where n is length of key used for encryption . For Eg:-

Plaintext :- hello world

Key:- vig

Plain text will be divided as follows and mapped with key given accordingly :- (in group of 3)

Hel low orl d

Vig vig vig v

Then (p+c)%26 formula is applied and letter corresponding to it is encoded character

Where ,

p:- number corresponding to character in plain text starting from zero,

c:-number corresponding to character present in key starting from zero

So applying above text we get cipher text as: cmrgwcjzry

Explain in brief how Kasiki test is used to break vigenere cipher?

The Kasiski test is a cryptanalytic method for breaking the Vigenere cipher. It works by looking for repeated sequences of characters in the ciphertext. If a repeated sequence of characters is found, the distance between the occurrences of the sequence is likely to be a multiple of the length of the keyword.

For example, suppose the ciphertext contains the sequence "ABCABC". If the keyword is 3 characters long, then the distance between the occurrences of the sequence will be 3, 6, 9, 12, etc. If the keyword is 4 characters long, then the distance between the occurrences of the sequence will be 4, 8, 12, 16, etc.

By finding the distances between repeated sequences in the ciphertext, the cryptanalyst can narrow down the possible values of the keyword length. Once the keyword length is known, the cryptanalyst can then use other methods to break the cipher.

Here is an example of how the Kasiski test can be used to break a Vigenere cipher:

Ciphertext: ABCDEFGH HIJKLMNOPQRSTUVWXYZ

The cryptanalyst first finds a repeated sequence in the ciphertext. In this case, the sequence "ABC" is repeated twice. The distance between the occurrences of the sequence is 12.

Since the distance between the occurrences of the sequence is a multiple of 3, the cryptanalyst knows that the keyword length is 3. The cryptanalyst can then use other methods to break the cipher, such as frequency analysis.

The Kasiski test is a simple but effective method for breaking the Vigenère cipher. It is not foolproof, but it can be used to break ciphers that have been encrypted with short keywords.

How Playfair Cipher works ? (with eg)

The Playfair cipher is a symmetric encryption technique that employs a 5x5 square matrix (keyed matrix) of letters to encrypt and decrypt messages. It was invented by Charles Wheatstone in 1854 and later popularized by Lord Playfair. The main idea behind the Playfair cipher is to encrypt pairs of letters (digraphs) from the plaintext into ciphertext using the following rules for encryption :

1) Before encrypting plain text if two consecutive letters in plaintext are same then insert bogus character ‘x’ in between them

2) If both characters in pair are in same row replace them by immediate right character from same row

3) if two characters in pair appear in same column replace them with immediate bottom character

4) If above two cases are not satisfied replace them by character in same row but in column of other character

Eg:- Suppose we want to encrypt the message "HELLO" using the keyword "KEYWORD" (without repeating letters, and 'J' is combined with 'I').

Keyed Matrix Setup:

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

The plaintext "HELLO" is divided into digraphs: "HE" and "LLO".

Encryption Rules:

"HE": H and E are in the same row, so we replace H with the letter to its right (E) and E with the letter to its right (F).

"LLO": L and O form a rectangle, so we replace L with the letter at the opposite corner of the rectangle (M) and O with the letter at the opposite corner of the rectangle (N).

Ciphertext: The encrypted message is "FE MN NM."

To decrypt the ciphertext, the recipient would use the same keyed matrix and apply the decryption rules in reverse.

How cryptanalysis on playfair cipher can be done ?

Cryptanalysis on the Playfair cipher involves attempting to break the encryption without knowing the key or the plaintext-ciphertext pair. Most common way is using frequency analysis

Cryptanalysts can perform frequency analysis on the ciphertext to identify patterns in the letter distribution. In English text, certain letters appear more frequently than others (e.g., 'E' is the most common letter). By analyzing the frequency of letters in the ciphertext, they can make educated guesses about which letters might correspond to common letters in the English language.

Shift cipher and Mono-alphabetic Substitution Cipher:

What is Shift Cipher ? (with eg)

Shift Cipher is one of the earliest and the simplest cryptosystems. A given plaintext is encrypted into a ciphertext by shifting each letter of the given plaintext by n positions.An example of encrypting the plaintext by shifing each letter by 3 places.

Plaintext: shift cipher is simple

Ciphertext: vkliwflskhulvvlpsoh

x be the position number of a letter from the alphabet

n be the an integer which 0<=n<=25, it is the key for encryption and decryption of shift cipher cryptosystem

The encryption process is (the x here represents a letter from plaintext):

x+n (mod 26)

The decryption process is (the x here represents a letter from ciphertext):

x-n (mod 26)

How and why shift cipher can broken using Brute force attack ?

A brute force attack is a method of breaking an encryption by trying every possible key until the plaintext is revealed. In the case of a shift cipher, there are only 26 possible keys, so a brute force attack is relatively easy to implement.

To break a shift cipher using a brute force attack, the attacker would simply encrypt the ciphertext with each of the 26 possible keys. Once the plaintext is revealed, the attacker would know which key was the correct one.

For example, if the ciphertext is "FEQJP" and the shift is 3, then the plaintext would be "hello". The attacker would simply encrypt the ciphertext with each of the 26 possible keys until they get "hello" as the plaintext.

What is monoalphabetic cipher ?(with eg)

A monoalphabetic cipher is a type of substitution cipher in which each letter of the plaintext is mapped to a single letter of the ciphertext. This means that there is a one-to-one correspondence between the letters of the plaintext and the letters of the ciphertext.

The Caesar cipher is a simple monoalphabetic cipher that shifts each letter of the plaintext by a fixed number of positions. For example, if the shift is 3, then the letter A would be encrypted as D, B would be encrypted as E, and so on.

Here is an example of how the Caesar cipher would work:

Plaintext: HELLO

Key: 3

Ciphertext: KILO

To decrypt the ciphertext, simply shift each letter back by the same number of positions. In this case, you would shift each letter back 3 positions.

Ciphertext: KILO Key: 3 Plaintext: HELLO

Can monoalphabetic cipher broken using brute force attack ? Why ?

No, monoalphabetic cipher cannot be broken using brute force attack because of its large keyspace. The keyspace of a monoalphabetic cipher is the set of all possible keys that can be used to encrypt a message. In the case of a monoalphabetic cipher, the key is simply a permutation of the alphabet. This means that there are 26! possible keys for a monoalphabetic cipher with a 26-letter alphabet.

Frequency analysis can be used to break monoalphabetic cipher

How can it be broken using frequency analysis attack ?

Frequency analysis is a cryptanalysis technique that exploits the fact that certain letters are more common than others in a language. For example, the letter "E" is the most common letter in the English language, followed by "T" and "A". This means that if you know the frequency of letters in the plaintext, you can use frequency analysis to guess the key of the ciphertext.

To break a monoalphabetic cipher using frequency analysis, you would first need to create a frequency table of the letters in the ciphertext. This table would show how often each letter appears in the ciphertext. Once you have created the frequency table, you would then need to compare it to the frequency table of the plaintext.

If the two frequency tables are similar, then you can use the frequency table of the plaintext to guess the key of the ciphertext. For example, if the letter "E" appears most often in the ciphertext, then you can guess that the letter "E" in the plaintext is mapped to the letter "E" in the ciphertext.This procedure can be repeated and then ciphertext can be converted to plain text .

RSA :

Explain the steps of RSA key generation

Steps involved in RSA key generation are as follows:-

1) Generate two large prime numbers, p and q.

2)Calculate the modulus n as n = p * q.

3)Calculate the totient ϕ(n) as ϕ(n) = (p - 1)(q - 1).

4)Choose an integer e such that 1 < e < ϕ(n) and gcd(ϕ(n), e) = 1.

5)Calculate the private exponent d as d = e^(-1) mod ϕ(n).

6)The public key is the pair (n, e), and the private key is the pair (n, d).

Eg:-

Let p = 7 and q = 11. Then, n = p * q = 77. The totient ϕ(n) is (p - 1)(q - 1) = 6 * 10 = 60. An integer e that satisfies the condition 1 < e < ϕ(n) and gcd(ϕ(n), e) = 1 is e = 5. The private exponent d is calculated as d = e^(-1) mod ϕ(n) = 3.

The public key is the pair (n, e) = (77, 5). The private key is the pair (n, d) = (77, 3).

In this example, the public key is (77, 5), and the private key is (77, 3). These keys can be used to encrypt and decrypt messages.

To encrypt a message, the sender would use the public key to encrypt the message. The receiver would then use the private key to decrypt the message.

For example, to encrypt the message "hello" using the public key (77, 5), the sender would first convert the message to a number. In this case, the number would be 104. The sender would then encrypt the number using the public key. The encrypted number would be 525.

The receiver would then use the private key (77, 3) to decrypt the number. The decrypted number would be 104. The receiver would then convert the number back to the message, which would be "hello".

Explain the steps of digital signature verification and generation process .

steps in generating and verifying a digital signature :

1. Generate the public and private keys.

The public and private keys are generated using the RSA key generation algorithm. The public key is made public, while the private key is kept secret.

2. Create a hash of the message.

The message to be signed is hashed using a cryptographic hash function. The hash function produces a fixed-length value, called the hash digest, that uniquely represents the message.

3. Encrypt the hash digest with the private key.

The hash digest is encrypted with the private key. The encrypted hash digest is the digital signature.

4. Send the message and the digital signature to the recipient.

The sender sends the message and the digital signature to the recipient.

5. Verify the digital signature.

The recipient decrypts the digital signature with the public key. The decrypted hash digest is compared to the hash of the message. If the two hashes match, then the signature is valid. overview of the steps involved in generating and verifying a digital signature:

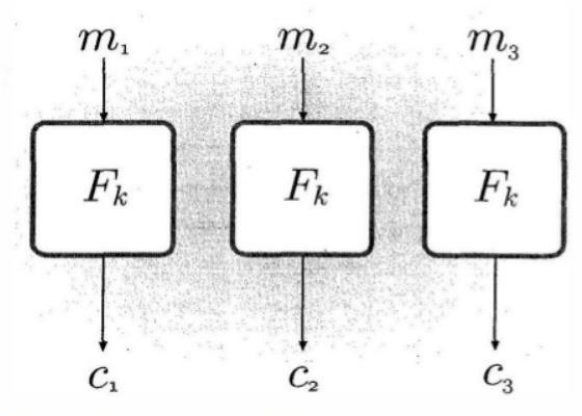
Generation

Generate the public and private keys. Create a hash of the message. Encrypt the hash digest with the private key.

Verification

Decrypt the digital signature with the public key. Compare the decrypted hash digest to the hash of the message.

Block cipher modes are techniques used in cryptography to specify how a block cipher, like AES (Advanced Encryption Standard) or DES (Data Encryption Standard), should be applied to encrypt or decrypt data of varying sizes, beyond just single blocks of data.



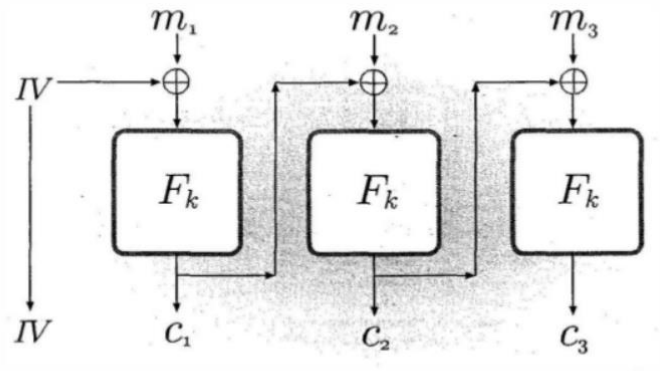
Electronic Code Book(ECB) mode

1. Electronic Codebook (ECB):

● Description: In ECB mode, data is divided into fixed-size blocks (typically 64 or 128 bits) and each block is encrypted independently using the same encryption key.

● Strengths: Simplicity and parallelism, making it efficient for hardware implementations.

● Weaknesses: Identical plaintext blocks result in identical ciphertext blocks, which can leak information and make it vulnerable to known-plaintext attacks. Not suitable for secure encryption of structured data.



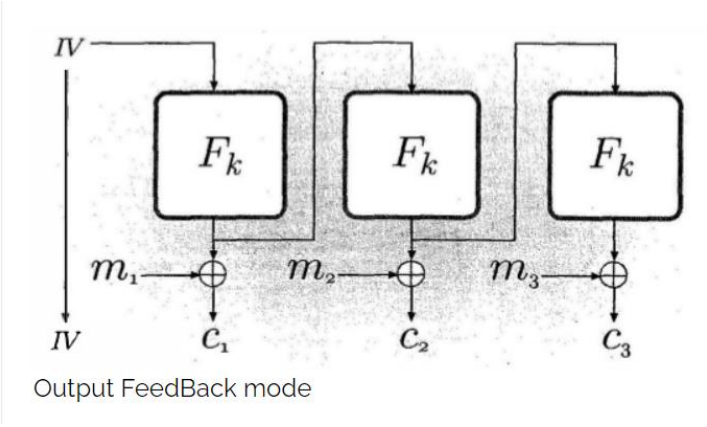
Cipher Block Chaining(CBC) mode

2. Cipher Block Chaining (CBC):

● Description: CBC mode adds an extra layer of security by XORing each plaintext block with the ciphertext of the previous block before encryption. It requires an Initialization Vector (IV) to start the process.

● Strengths: Provides confidentiality and data integrity, even when encrypting structured data like images and documents. Resistant to pattern analysis.

● Weaknesses: Not suitable for parallel processing due to the sequential nature of XOR operations. Requires proper handling of the IV to prevent security vulnerabilities.



Output FeedBack mode

5. Output Feedback (OFB):

● Description: OFB mode also turns a block cipher into a stream cipher. It generates a keystream independently of the plaintext, which is then XORed with the plaintext to produce the ciphertext.

● Strengths: Suitable for scenarios where data integrity is not the primary concern. Can be used for secure communication when error propagation is acceptable.

● Weaknesses: May not provide data integrity, as errors in ciphertext blocks won't affect the decryption of subsequent blocks. Not ideal for applications where data integrity and authentication are crucial.

FOR VIVA :

Aspect	SHA	MD5
Security	More secure, with recommended versions like SHA-256	No longer considered secure due to vulnerabilities
Length	Produces longer hashes, e.g., SHA-256 results in a 256-bit hash	Produces shorter 128-bit hashes
Collisions	Less likely to produce hash collisions	More susceptible to hash collisions
Usage	Widely used in modern security protocols and applications	Mainly used for non-cryptographic purposes, such as checksums

Kerberos: It's an authentication protocol for secure identity verification, commonly used in networks to prevent unauthorized access.

1. **Identity Verification**: Kerberos provides a robust method for verifying the identity of users or services, preventing unauthorized access to network resources.
2. **Single Sign-On**: It enables single sign-on (SSO), allowing users to access multiple services with a single authentication, improving user convenience and security.

Public Key Infrastructure (PKI): It's a system for managing digital certificates and keys to secure online communications and verify identities.

1. **Digital Certificates**: PKI uses digital certificates to authenticate and verify the identity of users and devices in online communications.
2. **Data Encryption**: It employs public and private keys to encrypt data, ensuring confidentiality, data integrity, and secure communication over the internet.

Diffie-Hellman: It's a method for two parties to exchange secret keys over a public channel securely, used to establish secure connections in various applications.

1. **Secure Key Exchange**: Diffie-Hellman facilitates secure key exchange between two parties, ensuring confidential communication even over insecure channels.
2. **Cryptographic Security**: It relies on the difficulty of solving the discrete logarithm problem, making it computationally infeasible for eavesdroppers to derive shared secret keys.