# Assignment: Cryptography Analysis and Implementation

**Registration Number: 20BRS1237**

**Name: Kodhai U**

**Objective:** The objective of this assignment is to analyze cryptographic algorithms and implement them in a practical scenario.

**Analysis:** Choose three cryptographic algorithms (one symmetric, one asymmetric, and one hash function) and write a detailed analysis of each. Include the following points in your

**Data Encryption Standard:**
A symmetric key technique called the Data Encryption Standard (DES) works with data blocks of a set length, usually 64 bits. Here is a high-level explanation of how DES functions:

**Step 1:** Create a 56-bit encryption key . Using a key generation algorithm, the key is created using a user-provided passphrase. The key needs to be protected and should only be given to people who have permission.

**Step 2:** Each block of the plaintext message consists of 64 bits. The plaintext block undergoes an initial permutation before encryption begins. In this permutation, the bits are rearranged in accordance with a predetermined table.
Discuss the key strengths and advantages of the algorithm.
Identify any known vulnerabilities or weaknesses.
Provide real-world examples of where the algorithm is commonly used.
**Step 3:** A Feistel structure, a particular design for cryptographic algorithms, is utilised by DES. The plaintext block is split in half for each cycle of encryption, and the identical set of operations are carried out on both sides.
**Step 4:** 16 distinct 48-bit subkeys are created from the 56-bit encryption key. Using a key schedule technique, these subkeys are created in advance from the original key.
**Step 5:** The following are the stages that make up each round of encryption:
a. predetermined expansion table is used to increase the 32-bit right half of the plaintext block to 48 bits.
b. Key Mixing: The subkey for the current round is XORed with the extended right half.
c. Substitution: Eight 6-bit segments make up the XOR operation's output. Eight 4-bit segments are produced by substituting each segment with one from a specified S-box table
d. Permutation: Next, an existing permutation table is used to rearrange the eight 4-bit segments.
e. Combining: The left half of the plaintext block is XORed with the processed right half.

f. Swap: The left and right halves of the plaintext block are switched when each round is finished.
**Step 6:** After the final round of encryption, the ciphertext block is subjected to a permutation. Inverting the first permutation, this one returns the bits to their original order.
**Step 7:** The encrypted ciphertext is contained in the last permuted block. The same procedures are followed in reverse order, using the same subkeys, to decode a block of ciphertext using DES. The decryption algorithm processes the inverse operations of the encryption method on the ciphertext block, yielding the original plaintext block.

**Elliptic Curve Cryptography:**
A public-key encryption technique called elliptic curve cryptography (ECC) is based on the theory of elliptic curves over finite fields. Compared to other public-key algorithms like RSA, ECC offers good security with comparatively reduced key sizes. Here is a high-level explanation of how ECC functions:
Generating a Key:
Curve choice: A appropriate elliptic curve is first selected from a list of available curves. The mathematical characteristics and security of the ECC system are determined by the curve parameters, such as the prime modulus and coefficients.
2. The creation of a public-private key pair is a step in the ECC process. The private key is a number that is created at random from a range and is normally selected uniformly at random. Using point multiplication on the elliptic curve, the public key is created by subtracting the private key.
Encryption: 1. Message Representation: To encrypt a message, a numerical representation that represents the message as a point on the elliptic curve is commonly used. Elliptic Curve Integrated Encryption technique (ECIES), for example, is one encoding technique that is used in this conversion process.
2. Key Agreement (Optional): Key agreement techniques like Diffie-Hellman key exchange can employ ECC. Both sides create their own private-public key pairs, trade public keys, and then use those keys to calculate a shared secret in this scenario.
3. Key Derivation (Optional): If a common secret is discovered via key agreement, a key derivation function can be employed to create a symmetric encryption key that is used to encrypt the actual message.
4. Encryption: Using a symmetric encryption method like AES, the message and the generated encryption key are combined during the encryption process.
1. Key Recovery: The recipient must have the corresponding private key linked to the public key used for encryption in order to decipher the ciphertext.
2. Point Multiplication: Using their private key and the elliptic curve, the recipient multiplies the received ciphertext. The shared secret or the numerical representation of the original message point is produced by this process.

3. Decoding the message: To get the original plaintext message, the shared secret, which stands for the message's numerical representation, is decoded in accordance with the established encoding method.
Elliptic Curve Cryptography has a number of benefits over other public-key techniques, including robust security, effective computing, and reduced key sizes. Due to these features, it is a good fit for devices with limited resources, such as mobile and Internet of Things (IoT) devices, where computational effectiveness and low power consumption are essential.

**Message Digest Algorithm 5:**
512-bit blocks are used to split up the input message. Padding is inserted to a message if it is not a multiple of 512 bits so that it may be processed in blocks. Four 32-bit words, A, B, C, and D, are used by MD5 as internal state variables. These are started with certain constants.

Processing Blocks: Various operations are carried out on the internal state variables throughout each cycle of processing a 512-bit block. 16 32-bit words, referred to as M[0] through M[15], which are utilised in the rounds make up the block. The letters F, G, H, and I stand for the four primary round functions of MD5. These operations include modular addition, circular shifts, and bitwise logical operations like AND, OR, and XOR. The final values of the internal state variables A, B, C, and D are concatenated to produce the 128-bit message digest, which is frequently shown as a string of 32 hexadecimal numbers, once all the blocks have been processed. To provide a fixed-size hash value that represents the input message is MD5's primary goal. It is ideal for the hash function to provide a distinct digest for each distinct input, making it appropriate for confirming data integrity and spotting modifications or tampering.
It's crucial to remember that MD5 is regarded as cryptographically flawed and unsafe for a number of applications, especially those involving password hashing and digital signatures. It is inappropriate for security-sensitive applications because to flaws including collision attacks and the capacity to produce the same hash value for various inputs. For cryptography applications, more secure options like SHA-256 (Secure Hash Algorithm 256-bit) are advised.