CHAPTER 4

DEMONSTRATION OF EXISTING SYSTEMS

4.1 PCA ALGORITHM

The Principal Component Analysis Algorithm is used for the recognition of a human face using eigen vectors. Image M can be represented by a matrix using its pixel values. The vector which is used to represent these values is said to be a eigen values of an image. Sub vectors from these basis vectors to compare stored and acquired new images. Based on these vectors the principal components of the given image can be found. The acquired new image is then compared with the existing stored images to achieve the perfect match.

Figure 4.1 illustrates the block diagram of PCA.

![Figure 4.1 Block diagram of PCA](image-url)
4.2 SWAPPING ALGORITHM

Swapping algorithm is the method that is used to encrypt the message easily.

![Block diagram of swapping algorithm](image)

**Figure 4.2 Block diagram of swapping algorithm**
Figure 4.2 shows the block diagram of swapping algorithm.

Features of swapping algorithm are:

- It uses 32 bit key
- Stream buffer encoding
- ANSI coding
- Uses single primitive: XOR
- Data is divided into two halves

Swapping algorithm steps:

Step 1: Start.
Step 2: Get message.
Step 3: Swap it,

\[
L = \text{len}(\text{msg});
\]

\[
\text{Swap}(\text{int})
\]

\[
\text{While}(i\leq 0)
\]

\[
\{
\]

\[
L_0 \& R_0
\]

for \( i = 1 \) to \( L \) do

\[
R_i = L_{i-1} \text{ XOR } P_i;
\]

\[
L_i = F[R_i] \text{ XOR } R_{i-1};
\]

\[
L = \text{len}(\text{msg})/2;
\]

\[
\text{Swap}(L);
\]

\[
}\]

where \( F[a,b] = ((S_1,a \text{ XOR} S_2,b) \text{ XOR} S_3,c) \text{ XOR} S_4,d \)

\[
S[a,b] = F[a,b]/2;
\]

Step 4: Get encrypted message.

Step 5: Stop
The algorithm provides reliable security. Since the key is generated by the system and the user, it gives higher security than dictionary password. It cannot be easily extracted by the hackers. After that the carrier signal is swapped with the generated key. This function has many combinations of AND, OR gates to collapse the text. This technique provides a strong encryption to a given text.

4.3 BLOWFISH ALGORITHM

Blowfish is a keyed, symmetric block cipher, designed in 1993 by Bruce Schneier and includes in a large number of cipher suites and encryption products. Blowfish provides a good encryption rate in software and no effective cryptanalysis of it has been found to date. However, the Advanced Encryption Standard now receives greater attention.

Schneier (1993) has designed Blowfish as a general-purpose algorithm, intended as an alternative to the aging DES and free of the problems and constraints associated with other encryption algorithms. When Blowfish was released, many other designs were proprietary, encumbered by patents or were commercial/government secrets. Schneier (1993) has stated that Blowfish is unpatented and will continue to remain so in all countries. It can be freely used by anyone.

The notable features of the design include key-dependent S-boxes and a highly complex key schedule. Figure 4.3 shows S-boxes manipulation of Blowfish algorithm.
This part is the inbuilt process for doing the encryption. It

• uses two primitives: addition & XOR

• divided data into two 32-bit halves $L_0$&$R_0$

for $i = 1$ to 16 do

$R_i = L_{i-1}$ XOR $P_i$;

$L_i = F[R_i]$ XOR $R_{i-1}$;

$L_{17} = R_{16}$ XOR $P_{18};$

$R_{17} = L_{16}$ XOR $i_{17};$

where $F[a,b,c,d] = ((S_1,a$ XOR $S_2,b)$ XOR $S_3,c)$ XOR $S_4,d$
4.4 SHA-256 ALGORITHM

The speciality of the blowfish algorithm is its expectation of its key and the complexity of the S-block routines.

SHA- is a group of hash functions denoted by its bits. It was developed by the U.S National Security Agency (NSA). It is considered as a U.S Federal Information Processing standard by NIST.

SHA-hashes each text character into a specific value. This mapping process can be done by the computerised program. Computation of the hash value is required for each character of given text data. 224,256,384,512 bits SHA have been developed yet.

The security provided by a hashing algorithm is entirely dependent upon its ability to produce a unique value for any specific set of data. When a hash function produces the same hash value for two different sets of data, a collision is said to occur. Collision raises the possibility for an attacker to computationally craft sets of data which provide access to information secured by the hashed values of pass codes or to alter computer data files in a fashion that would not change the resulting hash value and would thereby escape detection. A strong hash function is one that is resistant to such computational attacks, while a weak hash function is one where a computational approach to producing collisions is believed to be possible. A broken hash function is one where a computational method for producing collisions is known to exist.
Hashing algorithm steps are:

Step 1: Get original message.
Step 2: Get password to encrypt.
Step 3: Create message digest using original message.
Step 4: Generate hash value to each character of original message using key.
Step 5: Get output.

4.5 MD5 ALGORITHM

MD5 means Message Digest version 5. It has been developed by Ronald Rivest (1992). It accepts variable length message and converts it into a fixed length output of 128 bits. A given input can be broken into small parts of 512 bit messages that are sixteen 32 bit blocks. Then the message is padded to get fixed length message which is divisible by 512.

The padding mechanism follows two steps detailed below:

i) Single bit value 1 is added to the end of the message.

ii) Later several zeros are added until it reaches its fixed length word up to 64 bit but lesser than the multiple of 512. The rest of the bits are filled with the original message.

MD5 message digest works with 128 bits which can be divided into 4 X 32 bit words. So that four different words can be formed. That is represented by A,B,C,D as given below.

\[ F(B, C, D) = (B \land C) \lor (\neg B \land D) \]

\[ G(B, C, D) = (B \land D) \lor (C \land \neg D) \]

\[ H(B, C, D) = B \oplus C \oplus D \]
\[ I(B,C,D) = C \oplus (B \lor \neg D) \]

\( \oplus, \land, \lor, \neg \) denote the XOR, AND, OR and NOT operations respectively.

Figure 4.4 Block diagram of MD5

The message is fragmented into 512 bit blocks. Each block maintains its own status whenever message is switched over from one block to another. F is a non-linear function which allows the performance of different operations on the message. Probably addition and XOR functions are used to scramble the message.