Abstract
The most commonly used cryptosystem is DES. DES created a revolution in the field of cryptography when it was first announced. However, the advancement in the cryptanalysis techniques has rendered this technique as a weak algorithm. The original DES used a 64 bit key and followed the standard Feistel cipher methodology in which the message was divided into two parts and operations of permutation and substitution were performed on it. The key schedule of the standard DES and its improved versions is easily obtainable because the permutations are very simple. In this paper, we have introduced a better and improved DES in which we will use a 112 bit key instead and also divide a 96 bit message block into 3 sub-blocks and use 2 functions f1 and f2 in each round. Each function uses different key (K1 and K2 respectively). For the key schedule, we will perform SHA 2 hash function on the initial key and also successive rounds. Each hash value is 512 bit and we will partition it into 2 keys that can be used for four successive rounds of the 3 block DES. The time required for cracking the algorithm will increase. This makes the key schedule much stronger. Thus, the security is further enhanced. Even dividing the message block into 3 blocks instead of 2 and applying 2 different functions f1 and f2 at each round with two different round sub-keys will improve the complexity further. We will also perform cryptanalysis of our algorithm to test its strength and will try to make some changes after analysis to further improve security.

1. INTRODUCTION
The main base for our improved algorithm is the Data Encryption Standard. We referred to some white papers on the original algorithm. DES (Data Encryption Standard) was the first encryption standard to be recommended by NIST (National Institute of Standards and Technology). It was developed by an IBM team around 1974 and adopted as a national standard in 1997. The algorithm processes with an initial permutation, sixteen rounds block cipher and a final permutation. DES application is very popular in commercial, military, and other domains in the last decades. There are variants like 3DES, AES by enhancing DES function. A major weakness of DES is its key schedule. Thus, as a solution, we will be using SHA 2 to make the key more secure against cryptanalysis.

In cryptography, SHA-2 is a set of cryptographic hash functions (SHA-224, SHA-256, SHA-384, SHA-512) designed by the National Security Agency(NSA) and published in 2001 by the NIST as a U.S. Federal Information Processing Standard. SHA stands for Secure Hash Algorithm. SHA-2 includes a significant number of changes from its predecessor, SHA-1. SHA-2 consists of a set of four hash functions with digests that are 224, 256, 384 or 512 bits.

In 2005, security flaws were identified in SHA-1, namely that a mathematical weakness might exist, indicating that a stronger hash function would be desirable. Although SHA-2 bears some similarity to the SHA-1 algorithm, these attacks have not been successfully extended to SHA-2.

SHA 512 is a very strong algorithm used to compute a 512 bit hash value. We will be using it for our key schedule in the modified DES rounds.

The enhanced DES algorithm is one in which the message block is divided into 3 sub-blocks and the Feistel technique is slightly modified and applied to it. The complexity of the algorithm is increased and hence its security. Thus, we have combined several improvements and have changed the key schedule to make a new algorithm altogether.

2. Reasons to propose a new algorithm
The DES algorithm is insecure primarily due to the following reasons:\[3]:

- Small key length: 56 bits
- Susceptible key schedule: The user password directly becomes the key for the first round.

By doubling the key length, changing the internal operation and fortifying the key schedule using the SHA-512 algorithm for initial key generation, the proposed algorithm removes several of DES’s known vulnerabilities. Splitting the 512-bit hash key into two parts also ensures that consecutive round keys are not interdependent. This splitting action essentially means that double DES is applied in the guise of single DES.

The standard DES however, became the base for all future robust algorithms. Although DES created a revolution in cryptography, it is now considered insecure in many applications. Let us take a look at the internal working of our new improved algorithm [1]. We will then focus on the strengths of the algorithm.

3. Internal Working of existing standard DES

DES[1] (and most of the other major symmetric ciphers) is based on a cipher known as the Feistel block cipher. It consists of a number of rounds where each round contains bit shuffling, non-linear substitutions (S-boxes) and exclusive OR operations. Most symmetric encryption schemes today are based on this structure (known as a feistel network).

As with most encryption schemes, DES expects two inputs: the plaintext to be encrypted and the secret key. DES is a symmetric, 64 bit block cipher as it uses the same key for both encryption and decryption and only operates on 64 bit blocks of data at a time. The key size used is 56 bits, however a 64 bit key is actually input. The least significant bit of each byte is either used for parity.

Once a plain-text message is received to be encrypted, it is arranged into 64 bit blocks required for input. If the number of bits in the message is not evenly divisible by 64, then the last block will be padded. The processing of the plaintext proceeds in three phases as follows:

1. Initial permutation (IP) rearranging the bits to form the "permuted input".

2. Followed by 16 iterations of the same function (substitution and permutation). The output of the last iteration consists of 64 bits which is a function of the plaintext and key. The left and right halves are swapped to produce the pre-output.

3. Finally, the pre-output is passed through a permutation (IP−1) which is simply the inverse of the initial permutation (IP). The output of IP−1 is the 64-bit cipher text.

4. Proposed Algorithm

4.1 Pre-encryption in SVKT algorithm

Initially the user enters a password which will be used as the input to our key generator. The password will be 16 Unicode characters in length. Then we divide the password into 8 blocks as shown and perform SHA-512 operations on it to generate a new 512 bit hash key.

SHA 512 is a very strong hash function and has not been broken so far. It is a one way hash and takes in a string of any length and returns a hash key of 512 bits. It uses 8 chaining variables A to H. These chaining variables are initialized in the beginning to 32 bit values. The hash key produced at the end of 64 rounds is unique for every message. Also, this algorithm exhibits the avalanche effect i.e. with a small change in the message there is a considerably large change in the final hash value. However, because of its complex internal operations, it has not yet been successfully attacked by cryptanalysts.
One iteration in a SHA-2 family compression function. The blue components perform the following operations:

\[
    Ch(x, y, z) = (x \land y) \oplus (\neg x \land z)
\]
\[
    Maj(x, y, z) = (x \land y) \oplus (x \land z) \oplus (y \land z)
\]
\[
    \Sigma_0(x) = S^{28}(x) \oplus S^{34}(x) \oplus S^{39}(x)
\]
\[
    \Sigma_1(x) = S^{14}(x) \oplus S^{18}(x) \oplus S^{41}(x)
\]
\[
    \sigma_0(x) = S^1(x) \oplus S^8(x) \oplus R^7(x)
\]
\[
    \sigma_1(x) = S^{19}(x) \oplus S^{61}(x) \oplus R^6(x)
\]

**SHA-512 message schedule:**

\[
    W_j = M_j^{[i]} \quad \text{for } j = 0, 1, \ldots, 15, \text{ and } \forall x \quad j = 16 \text{ to } 79
\]

\[
    W_j \leftarrow \sigma_1(W_{j-2}) + W_{j-7} + \sigma_0(W_{j-1})
\]

These are the initial default values assigned and \(H_0\) to \(H_8\) are assigned to the variables \(A\) to \(H\) respectively.

The bitwise rotation uses different constants for SHA-512. The given numbers are for SHA-256. The red \(\oplus\) is an addition modulo \(2^{32}\).

\(\gg\gg\) represents rotate right and the number corresponds to the number of bits rotated.

The 512 bit hash key produced will be divided into 4 parts of 128 bits each.

### 3.3 SVKT Key Modification

This hash key is divided into two 64-bit keys \(k_1\) and \(k_2\). These keys will be passed through the DES S-boxes to generate 96 bit keys that are used as round keys. \(k_1\) will be used for round 1 and \(k_2\) will be used for round 2. This eliminates a DES weakness of interdependence of round keys. After every 4 rounds, \(k_1\) & \(k_2\) are modified using the SHA-512 algorithm. We will run SH-512 2 times in order to generate all the round keys for 8 rounds (2 sub-keys at each round).

### 4.2 Internal working of SVKT rounds

Instead of the standard 56 bit DES, we will use a 112 bit key. Also, our modified DES algorithm will be able to encrypt 96 bits of data at a time as against 64 bit in the earlier versions. Instead of the standard feistel cipher technique, we will make a few additional changes as follows:
Steps:

1) The 96 bit block will be divided into 3 blocks of 32 bits each (A, B, C).
2) We will use the 8 S3 S-boxes (S1 to S8). These S-boxes are resistant to linear and differential cryptanalysis.
3) In each round, the generated 128 bit key is divided into two parts (k1 and k2) and the appropriate function is performed.
4) The same function is used with two different 56 bit keys (reduced from 64 bit) is used in each round.
5) The operations are as follows:

Encryption:
\[ A(i) = B(i-1) \]
\[ B(i) = C(i-1) \oplus f(A(i-1), K(2,i)) \]
\[ C(i) = A(i-1) \oplus f(B(i-1), K(1,i)) \]

Decryption:
\[ A(i-1) = C(i) \oplus f(A(i), K(2,i)) \]
\[ B(i-1) = A(i) \]
\[ C(i-1) = B(i) \oplus f(A(i), K(1,i)) \]

\[ f1(B(i-1), K(1,i)) = P(S1(D1),\ldots,S8(D8)) \]
\[ f2(B(i-1), K(2,i)) = P(S1(D9),\ldots,S8(D16)) \]

We will perform the SHA-512 algorithm at the end of every four rounds as a part of our key schedule. We will perform 8 rounds of modified DES Feistel technique. We will be using S3 S-boxes instead of standard DES S-boxes.

5. Cryptool Testing

The tool used for performing cryptographic analysis is ‘Cryptool’.

Cryptool is used to analyse the following factors of the text:

1) Entropy

Entropy is the planned disorder given to the set of cipher-text in a message. It might be made up of several things that are mainly randomness, mathematical disfiguration and obfuscation. Entropy is the measure of uncertainty associated with a random variable. In terms of Cryptography, entropy must be supplied by the cipher for injection into the plaintext of a message so as to neutralize the amount of structure that is present in the unsecure plaintext.
message. The common notion of entropy is the notion of Shannon entropy. The Shannon entropy $H(x)$ of a value $x$ that occurs with probability $P_r[x]$:

$$H(x) = -\log_2(P_r[x])$$

you can interpret this as the uncertainty about $x$ knowing only the distribution according to which $x$ is chosen. For the case of bit strings the important message is, the entropy not always equals the bit length of $x$. I.e. if you want to seed a pseudorandom generator, it is important to choose a seed with high entropy. If you use the actual time as seed the entropy of the seed is very low, as everybody can easily guess parts of the seed (year, day, perhaps even hour and minute).

2) Histogram

A histogram is a graphical representation showing a visual impression of the distribution of data. It is an estimate of the probability distribution of a continuous variable and was first introduced by Karl Pearson. A histogram consists of tabular frequencies, shown as lines erected over discrete intervals (bins), with an area equal to the frequency of the observations in the interval. The height of a line is also equal to the frequency density of the interval, i.e., the frequency divided by the width of the interval. A histogram may also be normalized displaying relative frequencies. The categories are usually specified as consecutive, non-overlapping intervals of a variable. The categories (intervals) must be adjacent, and often are chosen to be of the same size.

3) N gram

What is N gram? An n-gram model is a type of probabilistic language model for predicting the next item in such a sequence in the form of a $n$-order Markov model. n-gram models are now widely used in probability, communication theory, computational linguistics (for instance, statistical natural language processing), computational biology (for instance, biological sequence analysis), and data compression. The two core advantages of n-gram models (and algorithms that use them) are relative simplicity and the ability to scale up – by simply increasing $n$ a model can be used to store more context with a well-understood space–time tradeoff, enabling small experiments to scale up very efficiently.
N-gram

Digram

<table>
<thead>
<tr>
<th>No.</th>
<th>Char.</th>
<th>Frequency</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>b1</td>
<td>1.1587</td>
<td>10</td>
</tr>
<tr>
<td>2</td>
<td>a0</td>
<td>1.0429</td>
<td>9</td>
</tr>
<tr>
<td>3</td>
<td>08</td>
<td>0.9327</td>
<td>8</td>
</tr>
<tr>
<td>4</td>
<td>4b</td>
<td>0.9327</td>
<td>8</td>
</tr>
<tr>
<td>5</td>
<td>9b</td>
<td>0.9327</td>
<td>8</td>
</tr>
<tr>
<td>6</td>
<td>ef</td>
<td>0.9327</td>
<td>8</td>
</tr>
<tr>
<td>7</td>
<td>5b</td>
<td>0.8111</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td>se</td>
<td>0.8111</td>
<td>7</td>
</tr>
<tr>
<td>9</td>
<td>88</td>
<td>0.8111</td>
<td>7</td>
</tr>
<tr>
<td>10</td>
<td>ee</td>
<td>0.8111</td>
<td>7</td>
</tr>
<tr>
<td>11</td>
<td>c0</td>
<td>0.8111</td>
<td>7</td>
</tr>
<tr>
<td>12</td>
<td>09</td>
<td>0.6952</td>
<td>6</td>
</tr>
<tr>
<td>13</td>
<td>19</td>
<td>0.6952</td>
<td>6</td>
</tr>
<tr>
<td>14</td>
<td>1b</td>
<td>0.6952</td>
<td>6</td>
</tr>
<tr>
<td>15</td>
<td>1d</td>
<td>0.6952</td>
<td>6</td>
</tr>
<tr>
<td>16</td>
<td>22</td>
<td>0.6952</td>
<td>6</td>
</tr>
</tbody>
</table>

Figure 4: Digram for cipher text (SVKT)

Trigram

<table>
<thead>
<tr>
<th>No.</th>
<th>Char.</th>
<th>Frequency</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>477</td>
<td>0.5480</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>56d</td>
<td>0.5480</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>55e</td>
<td>0.5480</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>56e</td>
<td>0.5480</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>01b</td>
<td>0.5480</td>
<td>3</td>
</tr>
<tr>
<td>6</td>
<td>c07e</td>
<td>0.5480</td>
<td>3</td>
</tr>
<tr>
<td>7</td>
<td>007</td>
<td>0.2320</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>065</td>
<td>0.2320</td>
<td>2</td>
</tr>
<tr>
<td>9</td>
<td>089</td>
<td>0.2320</td>
<td>2</td>
</tr>
<tr>
<td>10</td>
<td>08b</td>
<td>0.2320</td>
<td>2</td>
</tr>
<tr>
<td>11</td>
<td>16b</td>
<td>0.2320</td>
<td>2</td>
</tr>
<tr>
<td>12</td>
<td>1ab</td>
<td>0.2320</td>
<td>2</td>
</tr>
<tr>
<td>13</td>
<td>1af</td>
<td>0.2320</td>
<td>2</td>
</tr>
<tr>
<td>14</td>
<td>1b2</td>
<td>0.2320</td>
<td>2</td>
</tr>
<tr>
<td>15</td>
<td>22b</td>
<td>0.2320</td>
<td>2</td>
</tr>
<tr>
<td>16</td>
<td>22e</td>
<td>0.2320</td>
<td>2</td>
</tr>
</tbody>
</table>

Figure 5: Trigram values for cipher text (SVKT)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>DES</th>
<th>3DES</th>
<th>AES</th>
<th>SVKT (8 round)</th>
<th>SVKT (16 round)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Entropy</td>
<td>3.98</td>
<td>3.98</td>
<td>3.98</td>
<td>3.98</td>
<td>3.98</td>
</tr>
<tr>
<td>Histogram range</td>
<td>39-65</td>
<td>38-64</td>
<td>46-70</td>
<td>39-65</td>
<td>43-67</td>
</tr>
<tr>
<td>Digram range</td>
<td>7-12</td>
<td>6-9</td>
<td>7-8</td>
<td>6-10</td>
<td>7-10</td>
</tr>
<tr>
<td>Trigram range</td>
<td>2-3</td>
<td>2-3</td>
<td>2-3</td>
<td>2-3</td>
<td>2-3</td>
</tr>
</tbody>
</table>

Table 8: Comparison

Analysis:

As we can see, the entropy is the same as that of DES. Also the graph of histogram is more or less the same. However we can see that the di-gram of SVKT Block Cipher is better than that of DES.

Conclusion

The SVKT algorithm certainly seems to be a stronger and robust encryption technique. This is evident from the cryptanalysis that we have carried out. It combines the various strengths of the DES algorithm with the advantages of the Hashing technique which is used for key generation and provides a better alternative to several existing algorithms including the DES itself.

The proposed algorithm can replace DES in multiple applications, especially the ones requiring additional security. It can be used in innumerable areas of information security and networking. Today, several new types of attacks have made existing algorithms weak and vulnerable. SHA 512 is a very strong hash function and will indeed make it very difficult to crack. Thus, this algorithm can be used for securing important information that is accessed over the internet. It can also be used on a small scale such as securing confidential information within a company or organization.

References


