Advanced Cryption Using HASH-Cipheration

Hari Venkateswaran¹, Denesh Kumar Sundar²
¹,²Department Of ECE, St. Joseph’s College Of Engineering, Chennai

Abstract— In this paper, ADVANCED CRYPTOGRAPHY USING HASH AND RANDOM NUMBER GENERATION has been proposed. It is aimed at providing a greater security for the hard disk access. Even though hard disks are more secure, still there is an urge to steal data from hard disks. Hence to authenticate the access, we can use the proposed system. A HASH CODE is generated using SHA or MD5. The generated hash code is then concatenated with the random number generated using the random number generator algorithm. Then the concatenated code is used as a key to the TRIPLE DES system for encryption of hard disk access and the same key is given to the authenticated user. Then in the process of accessing the hard disk, the user must provide the hash number code he/she has got from system. Then pattern matching is done through the PATTERN MATCHING ALGORITHM (PMA) between the entered hash code and the previously used key for the process of encryption. If both the pattern match, the same key is provided for DECRYPTION using the same TRIPLE DES algorithm and the user is granted access.

Keywords— brute force attack prevention, hard disk access security, hash function encryption, pattern matching algorithm, quadrant colour algorithm, triple DES algorithm

I. INTRODUCTION

The Advanced encryption has been proposed for a secure hard disk access. In this mechanism, the encryption is performed in a new fashion. A hash code concatenated with a random number is used as a key for encryption and decryption thereby providing a secure access.

II. PROPOSED SYSTEM

The System consists of a Hash code generating function using the hash algorithms SHA or MD5. Three algorithms are proposed to generate a random number. The hash code is concatenated with the random number from any of the three algorithms and the result is a concatenated hash code of the form "###I***********". This is the key used for encryption using the TRIPLE DES algorithm. The same hash number is provided to the authenticated user. When the user again wants to access the hard disk he/she enters the hash number he received from the system. Then pattern matching is done with the code entered and the key used.

If both the patterns match, same key is used for DECRYPTION using TRIPLE DES and the user is granted access for the hard disk.

A. TRIPLE DES Encryption And Decryption

The Data Encryption Standard (DES) was developed by an IBM team around 1974 and adopted as a national standard in 1977. Triple DES is the minor variation of this standard. It is three times slower than regular DES but can be billions of times more secure if used properly. Triple DES enjoys wider use than DES because DES is so easy to break with rapidly advancing technology. In 1998 the Electronic Frontier Foundation, using a specially developed computer the DES Cracker, managed to break DES in less than 3 days. No sane security expert would consider using DES to protect data. So TRIPLE DES came as answer for all the shortcomings of DES. It has the advantage of proven reliability and longer key length that eliminates many of the shortcut attacks that can be used to reduce the amount of time it takes to break DES. Triple DES is simply another mode of DES operation. It takes three 64-bit keys, for an overall key length of 192 bits. In Private Encryption, we simply type in the entire 192 -bit (24 char) key rather than entering each of the three keys individually. The Triple DES DLL then breaks the user provided key into three sub keys padding the keys if necessary so they are each 64 bits long. The procedure for encryption is exactly the same as regular DES, but it is repeated three times. Hence the name Triple DES. Hence the data is encrypted with the first key, decrypted with the second key and finally encrypted again with the third key. Consequently, Triple DES runs three times slower than the normal DES, but is more secure if used properly. The procedure for encryption is same as the procedure of decryption but the process is executed in the reverse order. Although the input key 64 bits long, the actual key used by DES is only 56 bits in length. The least significant (right-most) bit in each byte is a parity bit, and should be set so that there are always odd number of 1’s in every byte. These parity bits are ignored and seven most significant bits of each byte is used, thereby resulting in a key length of 56 bits. This means the effective key strength for Triple DES is 168 bits.
B. **SECURE HASH Algorithm (SHA)**

The **SECURE HASH ALGORITHM** is one of the most secure implementations to generate a hash code. Since a hash code is not dependent on a definite pattern. It is a random code generated by random processes and a final compression function.

There are various versions of SHA, the popular being **SHA-1, SHA-256 and SHA-512**. In SHA-1 and SHA-256, begins by converting the message to a unique form of the message that is a multiple of 512 bits in length, without loss of information about its exact original length in bits, as follows: append a 1 to the message. Then add as many zeroes as necessary to reach the target length, which is the next possible length that is 64 bits less than a whole multiple of 512 bits. Finally, as a 64-bit binary number, append the original length of the message in bits. Starting from the 160 - bit block value (in hexadecimal) **67452301 EFCDBA89 98BADCFE 10325476 C3D2E1F0** as input for the processing of the first 512-bit block of the modified message, each message block do the following Encipher the starting value using the 80 sub keys for the current message block. Add each of the 32 - bit pieces of the cipher text result to the starting value, modulo $2^{32}$, of course, and use that result as the starting value for handling the next message block. The starting value created at the end of handling the last block is the hash value, which is 160 bits long.

1) **Description of SHA-512**:

SHA-512 is very similar to SHA-256, but not in the way that SHA-256 is similar to SHA1 both operate on 32-bit words, although the former operates on a block of eight of them, and the latter operates on a block of five of them.

On the other hand, SHA - 512 operates on eight 64-bit words, but the procedure it applies to them closely resembles that of SHA-256. From each block, considered as 16 64 bit words, 80 64-bit words are produced, the first 16 being the block itself, and the remaining words being the sum modulo $2^64$ of the following quantities:

a) the word 16 words ago
b) the word 7 words ago
c) the XOR of the word 2 words ago rotated right 19 places
   *that word rotated right 61 places
   *that word shifted right 6 places
d) the XOR of
   *the word 15 words ago rotated right 1 places
   *that word rotated right 8 places
   *that word shifted right 7 places.
One round of the part of SHA-512 that looks like a round of a block cipher is performed for each of these 80 words. For the first block, the initial input values to SHA-512 are:
6A09E66713BCC908 BB67AE8584CAA73B
3C6EF372FE94F82B A54FF53A5F1D36FG1
510E527FADE682D1 9B05688C2B3E6C1F
1F83D9ABFB41BD6B 5BE0CD19137E2179

C. Random Number Generation

In general, random number is something whose logic is not known to the outsiders who try to attack the logic through the loop holes present in the code. In this proposed system, we use a random number logic so as make the key used in TRIPLE DES more secure and less prone to BRUTE FORCE ATTACK. All cryptanalysts try to find the key used and then they try to find the data.

In the above set of random numbers, we can't find a particular logic of how it has been derived. In general, a statistical algorithm is used to generate these random numbers. An example of one such algorithm is Trek's random number generator algorithm. Random number table is created.

1) What Is A Random Table?

A random number table is a listing of random numbers. Stat Trek's Random Number Generator produces a listing of random numbers, based on the following User specifications:

- The quantity of random numbers desired.
- The max & min values of random numbers in Table 1
- Whether or not duplicate random numbers are permitted.

In our proposed system, we use three algorithms to generate random numbers. The logic of each algorithm is different from one another. The reason is attackers can’t find which algorithm has been used at the present time. All the three algorithms will not work in an ordered fashion.

<table>
<thead>
<tr>
<th>100 RANDOM NUMBERS</th>
</tr>
</thead>
<tbody>
<tr>
<td>92078 17049 15577 46552 53625</td>
</tr>
<tr>
<td>58966 42944 91010 20254 05963</td>
</tr>
<tr>
<td>75392 40548 45080 84601 39480</td>
</tr>
<tr>
<td>48284 74988 68320 35208 61102</td>
</tr>
<tr>
<td>09777 10236 62574 79001 03164</td>
</tr>
<tr>
<td>57234 95023 55098 01432 35871</td>
</tr>
<tr>
<td>85669 36276 57897 46148 81396</td>
</tr>
<tr>
<td>49093 92887 61506 82205 93550</td>
</tr>
<tr>
<td>86073 08504 41212 69647 84342</td>
</tr>
<tr>
<td>87142 05300 50825 40807 97418</td>
</tr>
<tr>
<td>01027 63643 75797 12777 28394</td>
</tr>
<tr>
<td>08100 68983 59775 60843 57638</td>
</tr>
<tr>
<td>67252 38412 51489 65374 97159</td>
</tr>
<tr>
<td>48689 47216 78192 58706 64047</td>
</tr>
<tr>
<td>48025 96091 51893 11045 07032</td>
</tr>
<tr>
<td>72188 50161 89682 71120 53366</td>
</tr>
<tr>
<td>80069 99959 66847 66183 41616</td>
</tr>
<tr>
<td>41875 94214 10640 34803 88873</td>
</tr>
<tr>
<td>26662 86737 33071 67511 90750</td>
</tr>
<tr>
<td>67915 62979 51230 86478 80733</td>
</tr>
</tbody>
</table>

2) How these Algorithms Work?

The execution of each algorithm depends on the hash code that is generated from the hash code generator. We check the initial digit or alphabet of the hash code. The algorithm 1 is called if it satisfies the given condition provided to it. If none of the conditions are satisfied, then the TRIPLE DES algorithm will not work and so encryption cannot be done.
In general a **PSEUDO RANDOM NUMBER GENERATOR** is the one which is more secure and reliable. The sequence is not truly random in that it is completely based on a set of initial values. In general careful mathematical analysis is required numbers that are sufficiently "random" to suit the intended use. In the proposed paper one of the algorithm to be used is the **MODULO RANDA TWISTER (MRT)**.

This is a theory based on colossal period of $2^{19937} - 1$ iterations which is $(>4.3 \times 10^{6001})$. We also have cryptographically secure **PRNG.** We use **BLOCK CIPHERS AND STREAM CIPHERS** here. We also have **NON-UNIFORM GENERATORS which** are based on general mathematical Distributions like the **POISSON DISTRIBUTION AND PROBABILITY DISTRIBUTIONS.**
D. Pattern Matching Algorithm (PMA)

In general, PATTERN MATCHING is said to be a process of making a match of the input patterns. There are four ways of pattern matching EXACT PATTERN MATCHING, GREP, REPATTERN MATCHING, KNUTH-MORRIS-PRATT. In our proposed system, pattern matching is used to compare the input streams that is, hash code that was used as a key for the encryption process and the code entered by the authenticated user of the system. If the two streams hold an exact match between each other, then the same input is used for the DECRYPTION using TRIPLE DES. A pattern can be dependent on the case used.

A simple code for a simple pattern match is as follows:

```java
public static int search(String pattern, String text)
{
    int M = pattern.length(); int N = text.length();
    for (int i = 0; i < N - M; i++)
    {
        int j;
        for (j = 0; j < M; j++)
            if (text.charAt(i+j) != pattern.charAt(j)) break;
        if (j == M) return i;
    }
    return -1;
}
```

Thus in our proposed system, pattern matching is used to provide more & more security and reliability as only authenticated users can give the exact pattern and they can only access the hard disk and they are granted access for it.

III. QUADRANT COLOR ALGORITHM

There may be a case where even the authenticated users can provide the wrong code or they may forget code. This process is not done intentionally. And because of this reason, users may be unable to access the system. This causes many disturbances at most times.

If the user types the wrong code for third time, then the hard disk gets locked. This action must be surely avoided. For this we use the QUADRANT COLOR ALGORITHM (QCA). The above algorithm is based on a pattern drawn by the authenticated user before he has an access to the system. The pattern drawn by the user is stored by the processor and it requires the pattern from the user when the code is entered wrongly for the third time.

We have a database of saved patterns from the authenticated users of the system. One user can have a pattern to his name. He must enter it correctly to gain the hash code again and then access the system. In this algorithm, we have a rectangular window with a black color externally & black contain a mixture of many colors. The rectangular windows is divided into 4 quadrants each having its co-ordinates. The user draws random lines to his interest and that particular pattern is saved by the system. When the exact co-ordinates match the user is again given the hash code again so that he can access the hard disk again. The user must be careful in drawing the pattern as a slight mismatch would cause the entire view of the image to change. Two Different patterns can be like this are shown in fig.4.

Fig.4. Two Different Patterns for QCA

And the user must be careful in drawing these patterns. We can see in these two patterns that they have a slight difference in the pixel positions of the brown colour. If the drawn matches the pattern in the database then user is an authenticated user and he can access the hard disk.
IV. CONCLUSION

Thus this is the proposal of our paper and our system be used in all business level paradigms and to provide a tight security for hard disks.

ACKNOWLEDGMENT

We wish to thank Dr.B. Babu Manoharan, M.A., M.B.A., Ph.D., Managing Director, St.Joseph's College of Engineering, for his constant support and encouragement to us in completing this paper. We are also grateful to Ms. Victoria Janjice, HOD, ECE, St.Joseph's College of Engineering. We also thank Mr.S. Hareesh, Programmer Analyst Trainee, Cognizant Technology Solutions India Private Limited, for his valuable support. We are thankful to our parents and friends for their support through all odds.

REFERENCES