CHAPTER 5
NEW ENCRYPTION SCHEME USING FINITE STATE MACHINE AND GENERATING FUNCTION

5.1 INTRODUCTION

Cryptography is the science of writing in secret code and is ancient art. But modern cryptography is heavily based on mathematical theory and computer science practice. Cryptographic algorithms are designed around computational hardness assumptions, making such algorithms hard to break in practice by any adversary. It is theoretically possible to break such a system but it is infeasible to do so, by any known practical means. These schemes are therefore termed computationally secure. Theoretical advances and faster computing technology require these solutions to be continually adapted. There exist information theoretically secure schemes that provably cannot be broken even with unlimited computing power. But these schemes are more difficult to implement than the best theoretically breakable but computationally secure mechanisms.

In the present chapter a new cryptographic scheme using generating functions for protection of digital signals security services is proposed. The efficacy of the proposed method is analyzed, which ensures improved cryptographic protection in digital signals. The proposed schemes solve cryptographic goals such as non-repudiation, authenticity, integrity problems etc in many security aware digital applications.

Literature [32] shows that any polynomial of finite order can be a secret key hence encryption schemes are proposed using generating function as secret key.

5.2 DEVELOPMENT OF CIPHER USING GENERATING FUNCTION

Generating function for a given recurrence relation of order n is given by

\[ a_0 + a_1 z + a_2 z^2 + a_3 z^3 + a_4 z^4 + \cdots \]
For example Fibonacci sequence 0 1 1 2 3 5 8 13 ……….

Generating function is $z + z^2 + 2z^3 + 3z^4 + \ldots$.

If $f(x)$ and $g(x)$ are two polynomials of degree $n-1$ in single variable $x$ i.e.,

$$f(x) = a_0 + a_1 x + \ldots + a_{n-1} x^{n-1} \quad \text{and} \quad g(x) = b_0 + b_1 x + \ldots + b_{n-1} x^{n-1}$$

Then define a new polynomial $h(x) = f(x) * g(x) = c_0 + c_1 x + \ldots + c_{n-1} x^{n-1}$

Where $c_i = a_0 b_i + a_1 b_{n-i} + \ldots + a_{i} b_0 + a_{i+1} b_{n-i-1} + \ldots + a_{n-1} b_{i+1}$.

The inverse of $f(x)$ (say $F(x)$) is also a polynomial of degree $n-1$ in single variable $x$, if it satisfies the property $f * F = 1$.

It is observed that not every polynomial has an inverse, but it is easy to determine $F$ if inverse of $f$ exists.\[32\]

Let $f(x)$ be the secret key polynomial (generating function up to some predefined value) and $g(x)$ be message polynomial in which each coefficient of $x$ is message in some predefined order.

### 5.3 PROPOSED ALGORITHM

#### 5.3.1 Encryption

Step 1

Let $P$ be the plain text array of order $1 \times n, n > 0$.

Consider the elements in $P$ as coefficients of $(n-1)$th degree polynomial in single variable 'x'.

Step 2

Define Moore machine through public channel. Send the secret key in binary form to the receiver. Send secret key polynomial of degree $(n-1)$ which is selected from the generating function.

Step 3

Define cipher text at each state.
Cipher text at q(i+1)th state equal to the cipher text at q(i) th state *( the polynomial ) output at q(i+1)th state . ]

Step 4

Send this cipher text to the receiver.

5.3.2 Decryption

Step 1

On receiving finite state machine, secret key and the secret polynomial and the cipher text calculate the original message using inverse of the secret polynomial.

5.4 Efficacy of the proposed algorithm

Mathematical work

Algorithm proposed is application of generating function defined on some recurrence relation. Number of rounds depends on the secret key. It is very difficult to break the cipher text without proper key, secret polynomial and the chosen finite state machine. The value ‘n’ is always finite.

Strength of the key

Select lengthy secret key to avoid the brute force attack.

Rounds

Number of rounds depends on secret key used and the chosen finite state machine. It is very difficult to guess the number of rounds and the recurrence matrix without an apt secret key.

5.5 Security analysis

To extract the original information is very difficult due to the chosen finite state machine. Brute force attack on key is also difficult due to the increase in key size.
<table>
<thead>
<tr>
<th>S.No</th>
<th>Name of the attack</th>
<th>Possibility of the attack</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Cipher text attack</td>
<td>Very difficult</td>
<td>Due to generating function and the chosen finite state machine.</td>
</tr>
<tr>
<td>2</td>
<td>Known plain text attack</td>
<td>Difficult</td>
<td>Due to the chosen finite state machine and the operation '*'.</td>
</tr>
<tr>
<td>3</td>
<td>Chosen plain text attack</td>
<td>Difficult</td>
<td>Due to the chosen finite state machine and the operation '*'.</td>
</tr>
<tr>
<td>4</td>
<td>Adaptive chosen plain text attack</td>
<td>Difficult</td>
<td>Due to the chosen finite state machine and the operation '*'.</td>
</tr>
<tr>
<td>5</td>
<td>Chosen cipher text attack</td>
<td>Very difficult</td>
<td>Due to generating function and the chosen finite state machine.</td>
</tr>
<tr>
<td>6</td>
<td>Adaptive chosen cipher text attack</td>
<td>Very difficult</td>
<td>Due to generating function and the chosen finite state machine.</td>
</tr>
</tbody>
</table>

Table 5.1 security analysis

5.6 **Application**

Let $P= (1 \ 2)$

Recurrence relation is Fibonacci sequence $0 \ 1 \ 1 \ 2 \ 3 \ 5 \ldots \ldots$

Select secret key as 21 (10101) and polynomial is $(1+x)$.

Send the polynomial, key in binary to the receiver and finite state machine which calculates the residue mod4.

For easy computations in each state, the cipher text is calculated with respect to consider mod 61

Cipher text at $q(i+1)$th state equal to the cipher text at $q(i)$th state * output at $q(i+1)$th state.

Then the cipher texts at various states are
<table>
<thead>
<tr>
<th>S.No</th>
<th>Input</th>
<th>Previous state</th>
<th>Present state</th>
<th>Out put</th>
<th>Cipher text</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>q₀</td>
<td>q₁</td>
<td>1</td>
<td>(3 5)</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>q₁</td>
<td>q₂</td>
<td>2</td>
<td>(21 34)</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>q₂</td>
<td>q₁</td>
<td>1</td>
<td>(55 18)</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>q₁</td>
<td>q₂</td>
<td>2</td>
<td>(42 11)</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>q₂</td>
<td>q₁</td>
<td>1</td>
<td>(53 3)</td>
</tr>
</tbody>
</table>

**Table 5.2 Example**

Then the cipher text is (53 3)

### 5.7 Conclusions

Algorithm proposed, is based on finite state machine and the polynomial selected from the generating function. Secrecy is maintained at four levels, the secret key, the chosen finite state machine, operation defined and the polynomial selected from the generating function. The obtained cipher text becomes quite difficult to break or to extract the original information even if the algorithm is known. .