International Journal on Recent and Innovation Trends in Computing and Communication ISSN: 2321-8169
Volume: 3 Issue: 10 5951 - 5952

Generic key Generation Algorithm using Weighted Graphs

Vigneet Sompura
Computer Engineering
Ahmedabad,India
vigneetsompura@gmail.com

Prajakta Karandikar
Computer Engineering
Ahmedabad,India
prajaktaiskarandikar@gmail.com

Dhvanish Patel
Computer Engineering
Ahmedabad,India
dhvanish17@gmail.com

***Abstract*—** Data is of all sorts, ranging from all pertaining entertainment to national defense. There is a highly pressing need in today's "online" world to maintain confidentiality of data. The cost of revelation of highly sensitized data far exceeds the cost to establish and maintain security measures. Disclosure of confidential data leads to cataclysmic losses to the victim enterprise. Not just a company's finances, even national security could be threatened by lax security. In order to keep our information discreet, we need to be on top of things by implementation of stronger encryption systems. On referring several encryption algorithms it was found that most encryption algorithms had a common trait that they used a single fixed key for entire data encryption. Modifying this peculiarity, our key generation algorithm strengthens any encryption algorithm by generating multiple keys.

Keywords- *Key Generation, Weighted Graph, Shortest Path, Unique Subkeys*

*****

I. INTRODUCTION

Nowadays, methods of data storage and data communication are evolving and are moving towards digitization increasing the vulnerability of data secrecy. Hence continuously increasing the need for higher security. Computer processors are getting faster which increases the efficiency of brute force attacks. We have smart intruders raising the need for smarter security features. One way to increase the security would be to employ higher data structures for generation of keys used for encryption. This would facilitate higher key complexity making it much harder to crack.

A conventional algorithm contains a key in string format which is used to encrypt multiple characters or blocks of plain text using an algorithm known as the encryption algorithm. If an intruder has the knowledge of the key, this compromises all the messages that are transmitted using that algorithm. An alternative to this would be to use different unique keys for each character or block of each message. To store such a large sequence of keys requires a lot of memory. This could be obviated by using our proposed key generation algorithm. The algorithm uses graphs to generate multiple subkeys for different characters or blocks of plaintext which eliminates the need of a large memory.

II. PROPOSED ALGORITHM

A weighted graph of N nodes is agreed upon by both the sender and receiver. To increase the complexity of the graph (and in turn the key), a higher value of N can be used. Values of two variables K1 and K2 are pre-decided. A counter is initialized once to zero at start of transmission of data. N1 is value of K which is calculated once for each new character/block. Counter is incremented with each new character for a character stream, or for each new block for block sequence. Value of counter for each new character/block is stored in N2.

Generation of sub-keys occurs in the following manner:

i. Find value of K such that K=(K1+K2)mod N
ii. Find the shortest path sequence between node (N1) and node (N2) according to the weights depicted in the graph.
iii. S1, S2, S3 are the node numbers in sequence on this shortest path.
iv. This sequence S: S1, S2, S3… is to be converted to a binary sequence B: B1, B2, B3…
v. If S1 is greater in value than S2, then B1 takes value: 1(One). If S2 is lesser in value than S3, then B2 takes value: 0(Zero).
vi. Similarly calculate the values of the B-sequence.

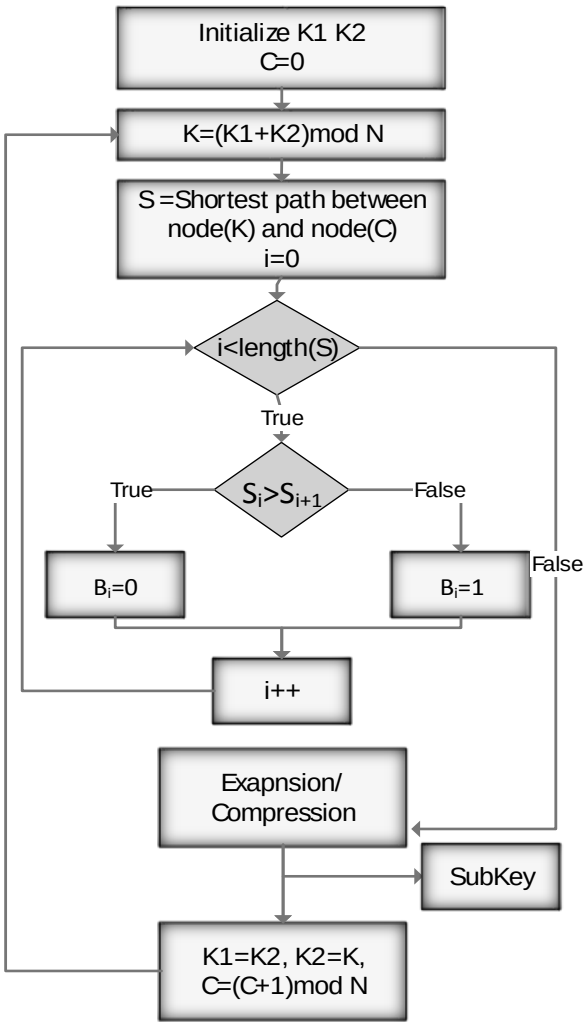

Figure 1 Flowchart for proposed key generation algorithm

While finding the shortest path, if there is a conflict at any point, then select the path for which the node in the sequence has a higher node value. For example, If there are two different shortest paths are 6-3-1-7 and 6-1-4-7 then select 6-3-1-7.

The binary sequence must be expanded or compressed to make the length of the binary sequence B equal to the length of the key as required by the selected encryption algorithm.

Expansion Function:

If the binary sequence falls short of the required key length, repeat the B-sequence until the required key length is obtained. For Example, If B-sequence is 10010 and required length is 8 bits, then on using expansion function, key obtained is 10010100.

Compression Function:

If the binary sequence is longer than the required key length suppose L, then select the first L bits of the B-sequence as the required key. For Example, if B-sequence is 1011101101 and required key length is 8 bits, then required key is 10111011.

Increment counter for the next character/ block in the following manner

C=(C+1) mod N

Find next key value

K1=K2

K2=K

Repeat the procedure till entire data is transmitted.

III. EXAMPLE

The graph with 10 nodes 0 to 9is given as below in Fig 2.
K1= 1 and K2= 3 and required subkey length is 8.

i. Set the Counter C to zero.

ii. Find K=(K1+K2) mod N (N is the number of nodes in the graph)

K= (1+3) mod 10
K= 4

iii. Find the shortest path sequence S between node (N1) and node (N2) where N1 has the value of K and N2 has the value of counter C.

Shortest path between node (4) and node (0) is 4-5-1-0
So S=4510

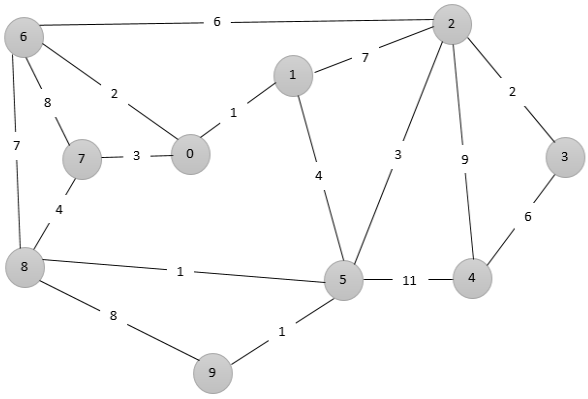

Figure 2. Weighted Graph for Example

iv. Binary Key Sequence B:

S0<S1 so B0=1
S1>S2 so B1=0
S2>S3 so B2=0
So Binary sequence B=100

v. Expansion/Compression Function:
In this case, Length (B) is lesser than required length so we have to repeat the binary sequence until the subkey of required length is obtained.

Subkey=10010010

vi. Increment Counter/Find next Key Value
K1=3
K2=4
C= (0+1) mod 10 = 1

Repeat step ii and vi to generate the next subkey.

IV. ADVANTAGES

- This key generation algorithm has universality in terms of adaption to the given encryption algorithm as the size of generated subkey can be varied using the expansion/compression function
- With this algorithm, there is an option to generate and store values of variable K to an array and use it to generate subkeys simultaneously using parallel processors and speed up the process of encryption or decryption.
- There is a unique subkey for each character of the message. If a letter occurs more than one times in a message, then for each occurrence a different subkey is generated.
- The level of security can be changed by changing the number of nodes in the graph.
- The weight assigned to each edge can be any number, so the number of possible graphs is very high.

ACKNOWLEDGMENT

Our thanks to Prof. Maitrik K. Shah who has helped us understanding the subject and gave helpful insights for improvement of this algorithm.

REFERENCES

[1] William Stallings , Cryptography and Network Security, 1998
[2] Amrita Sahu, Yogesh Bahendwar, Swati Verma, Prateek Verma, Proposed Method of Cryptographic Key Generation for securing Digital Image, 2012.
[3] Jean-Paul Tremblay and Paul G. Sorenson, An introduction to Data Structures with Applicatio, 1991.
[4] Atul Kahate, Cryptography and Network Security, 2003.