ISSN: 2278 – 1323
International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)
Volume 2, Issue 2, February 2013

Secure Secret Key (SSK) Generation Algorithm for Multimedia Multicast Networks

[1]Sathya S, [2]Ajay Kumar
[1,2]M.Tech IT (Networking)
School of Information Technology & Engineering, VIT University,
Vellore 632014, Tamilnadu, India

Abstract:
In the multimedia applications messages like audio and video messages are broadcasted through groups. To provide secrecy of those keys which are used in this communication, the methods like key generation and the key distribution among these groups should be done in a secure way. The problems due to this are mutual authentication with the exchange of a session key. In most of the Key Management schemes, different types of group users obtain a new distributed multicast group key which is issued for encrypting and decrypting multimedia data for every session update. Among the various works on key distribution, The RSA method focuses on making the key generation operation speed and also reduces the complexity. The main advantage of this algorithm is that only one message is generated per rekeying operation and only one key is stored in each user's memory. Key distribution is an important issue in multimedia networks. The Key Distribution algorithm and at the same time, increases the number of keys to be stored by GC (Group Center) and group members.

A new GCD (Greatest Common Divisor) and LCM based Key Distribution Protocol which focuses on two dimensions is also used to decrease the complexity by increasing the number of multiplication operations and for this the divide and conquer approach for multiplication is used. The second dimension aims at reducing the amount of information stored in the Group Center and group members while performing the update operation in the key content.

Index Terms—: **Multicast communication, Key distribution, GCD, LCM, Computation complexity, Karatsuba multiplication, Euclidian algorithm**

I. INTRODUCTION

In cryptography, key distribution is a part of cryptosystem intended to reduce the risks inherent in exchanging keys. Keys are used as part of encryption and decryption authentication functions to lock and unlock messages. While a particular encryption *algorithm* is largely used and well known, the keys used to make each encryption should be unique. Those keys should be kept secure and private. But there will be problems occurring while in exchanging of those keys. If you send an encrypted message to a friend, your friend will need a key to decrypt the message. The process of getting that key to your friend may be compromised. The methods for exchanging keys in secure ways over open networks like the Internet are described. The exchanging of keys provides secure communication between two authorized parties based on the basic principles of key management.

Multimedia services, such as pay-per-view, videoconferences, some sporting events, audio and video broadcasting are based upon multicast communication where multimedia messages are sent to a group of members. In such a scenario, groups can be either opened or closed with regard to senders. In a closed group, only registered members can send messages to this closed group. In contrast, data from any sender is forwarded to the group members in open groups. Groups can be classified into static and dynamic groups. In static groups, membership of the group is predetermined and does not change during the communication. In dynamic groups, membership can change during multicast communication. Therefore, in a dynamic group communication, members either join or depart from the service at any time. When a new member joins into the service, it is the responsibility of the Group Center (GC) to disallow new members from having access to previous data. This provides backward secrecy in a secure multimedia communication. Similarly, when an existing group member leaves from any group, he/she should not have further access to data. This achieves forward secrecy. In order to provide forward and backward secrecy the keys are frequently updated whenever a member joins/leaves the multicast service. Furthermore, if a device lacks storage capabilities, it may be impossible within the receiving device to implement a group key management protocol based on a key tree structure. Hence the amount of information to be stored to find the updated key by GC and group members should also be minimized. GC also takes care of the job of distributing the Secret key and Group key to the group members. In this paper, we propose a Key Distribution

Manuscript received Feb 28, 2013.

Ajay Kumar ,M.Tech IT Networking ,VIT University.
Sathya S M.Tech IT Networking ,VIT University.

algorithm that reduces the computational complexity and at the same time, it decreases the number of keys to be stored by GC and group members.

II. RELATED WORK

A. LCM of Fractions

The Least Common Multiple (LCM) of two integers **a** and **b**, is the smallest positive integer that is a multiple of both **a** and **b**. Generally LCM is used for adding fractions where denominators are not same.

It is denoted by LCM (**a, b**). The least common multiple of two numbers **a** and **b** can be obtained by finding the prime factorization of each

$$a = p_1^{a1} \dots p_n^{an}$$
$$b = p_1^{b1} \dots p_n^{bn}$$

Where the p_is are all prime factors of a and b, and if p_i does not occur in one factorization, then the corresponding exponent is taken as 0. The least common multiple is then given by

$$\mathrm{LCM}(a, b) = \prod_{i=1}^{n} p_i^{\max(a_i, b_i)}.$$

For example, consider LCM(12,30)

$$12 = 2^2 \cdot 3^1 \cdot 5^0$$
$$30 = 2^1 \cdot 3^1 \cdot 5^1,$$

So

$$\mathrm{LCM}(12, 30) = 2^2 \cdot 3^1 \cdot 5^1 = 60.$$

In terms of GCD

$$\mathrm{LCM}(a, b) = \frac{a\,b}{\mathrm{GCD}(a, b)},$$

B. GREATEST COMMON DIVISOR

In mathematics, the greatest common divisor (gcd), also known as the greatest common factor (gcf), or highest common factor (hcf), of two or more non-zero integers, is the largest positive integer that divides the numbers without a remainder.

FOR EXAMPLE, THE GCD OF 8 AND 12 IS 4.

The greatest common divisor of two integers is denoted by ***a*** and ***b*** as gcd (*a*, *b*).

C. Calculation

1) Using prime factorizations

Greatest common divisors can in principle be computed by determining the prime factorizations of the two numbers and comparing factors, as in the following example:
To compute gcd (18, 84), we find the prime factorizations $18 = 2 \cdot 3^2$ and $84 = 2^2 \cdot 3 \cdot 7$ and notice that the "overlap" of the two expressions is $2 \cdot 3$; so gcd (18, 84) = 6.

D. LOWEST COMMON DENOMINATOR

The lowest common denominator or least common denominator (abbreviated LCD) is the least common multiple of the denominators of a set of vulgar fractions. It is the smallest positive integer that is a multiple of the denominators.

The least common denominator of two or more non-zero denominators is actually the smallest whole number that is divisible by each of the denominators.
To find the least common denominator using this method, factor each of the denominators into primes. Then for each different prime number in all of the factorizations, do the following-

1. Count the number of times each prime number appears in each of the factorizations.
2. For each prime number, take the largest of these counts.
3. Write down that prime number as many times as you counted for it in step #2.
4. The least common denominator is the product of all the prime numbers written down.

Example: **1/5**, **1/6** and **1/15**.

Factor into primes

- Prime factorization of **5** is **5** (**5** is a prime number)
- Prime factorization of **6** is **2 x 3**
- Prime factorization of **15** is **3 x 5**
  // Notice that the different primes are 2, 3 and 5.

1. Now, we do **Step #1** – **Count** the number of times **each** prime number appears in **each** of the factorizations…
   - The count of primes in **5** is **one 5**
   - The count of primes in **6** is **one 2** and **one 3**

- The count of primes in **15** is **one 3** and **one 5**

2. **Step #2** – For **each** prime number; take the **largest** of these counts. So we have…
   - The largest count of **2s** is **one**
   - The largest count of **3s** is **one**
   - The largest count of **5s** is **one**
3. **Step #3** – Since we now know the count of each prime number, you simply – write down that prime number as **many times as you counted** for it in step #2.Here are the numbers… 2, 3, 5
4. **Step #4** – The least common denominator is the product of all the prime numbers written down.
   2 x 3 x 5 = 30
   // Therefore, the least common denominator of **1/5, 1/6** and **1/15** is **30**

3. Euclidean Algorithm

Start with any two elements 'u and v' of a Euclidean Domain

- If v=0, then gcd (u, v) =u.
- Otherwise take the remainder when u is divided by u (mod v), and find Gcd (u, u mod v)
- Repeat this until the remainder is 0.

u (mod v) = W_1

v (mod W_1) =W_2

.

.

.

W_{n-1} (mod W_n) = 0

Then gcd (u, v) = W_n

Usually the Euclidean algorithm is written down just as a chain of divisions with remainder:

For $W_{n+1} < W < W_{n-1}$

u= v*q_1 + W_1

v=W_1*q_2 + W_2

W_1= W_2*q_3 + W_3

.

.

.

W_{n-1} = W_n*q_{n+2} + 0

And so gcd (u, v) = W_n

LCM (u, v) = u*($\frac{v}{\gcd(u,v)}$)

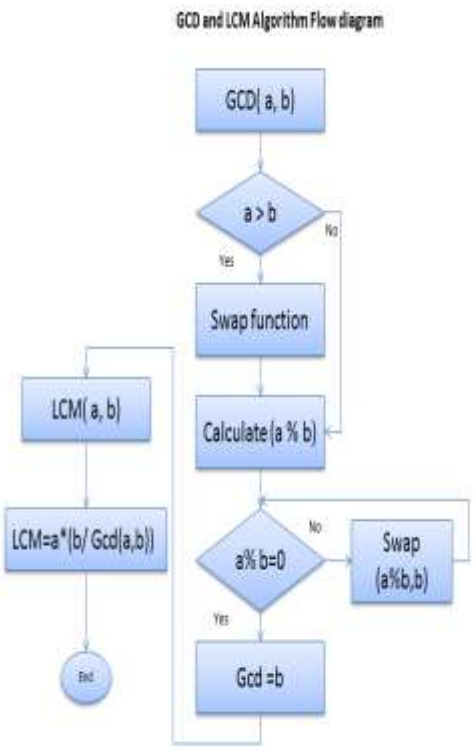


Figure I: Flow Diagram of generation of Keys using algorithm.

The flowchart is designed in such a way the values taken for calculating GCD will be taken randomly. Namely a and b. A condition is checked that the values a and b taken should be equal to zero. If not the values are swapped using the swap function and generated GCD values are taken to find the LCM. so finally the output will be two windows asking to enter a and b values and to generate GCD and LCM values. By this process the keys are generated more secure with the help of LCM so that an attacker cannot easily break the key.

Below flow diagram shows password encryption method using GCD&LCM algorithm. This is an implementation of this algorithm that how it can be used in net banking sector.

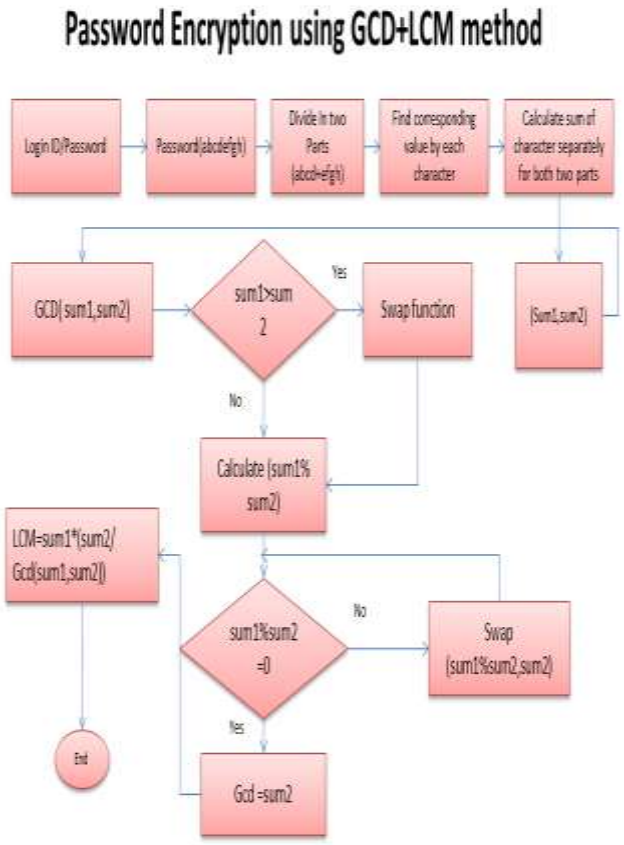


Figure II: Flow diagram of Password encryption using GCD & LCM algorithm.

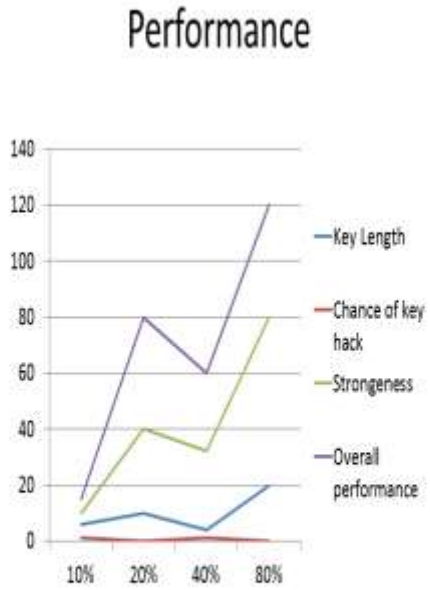
Table I: Assumption values corresponding to all character.

Integer value corresponding to each character

A=0	a=26	0=52	-=78
B=1	b=27	1=53	?=79
C=2	c=28	2=54	/=80
D=3	d=29	3=55	>=81
E=4	e=30	4=56	<=82
F=5	f=31	5=57	,=83
G=6	g=32	6=58	.=84
H=7	h=33	7=59	~=85
I=8	i=34	8=60	`=86
J=9	j=35	9=61	
K=10	k=36	!=62	
L=11	l=37	@=63	
M=12	m=38	#=64	
N=13	n=39	$=65	
O=14	o=40	%=66	
P=15	p=41	^=67	
Q=16	q=42	&=68	
R=17	r=43	*=69	
S=18	s=44	(=70	
T=19	t=45	)=71	
U=20	u=46	_=72	
V=21	v=47	+=73	
W=22	w=48	|=74	
X=23	x=49	==75	
Y=24	y=50	==76	
Z=25	z=51	_=77	

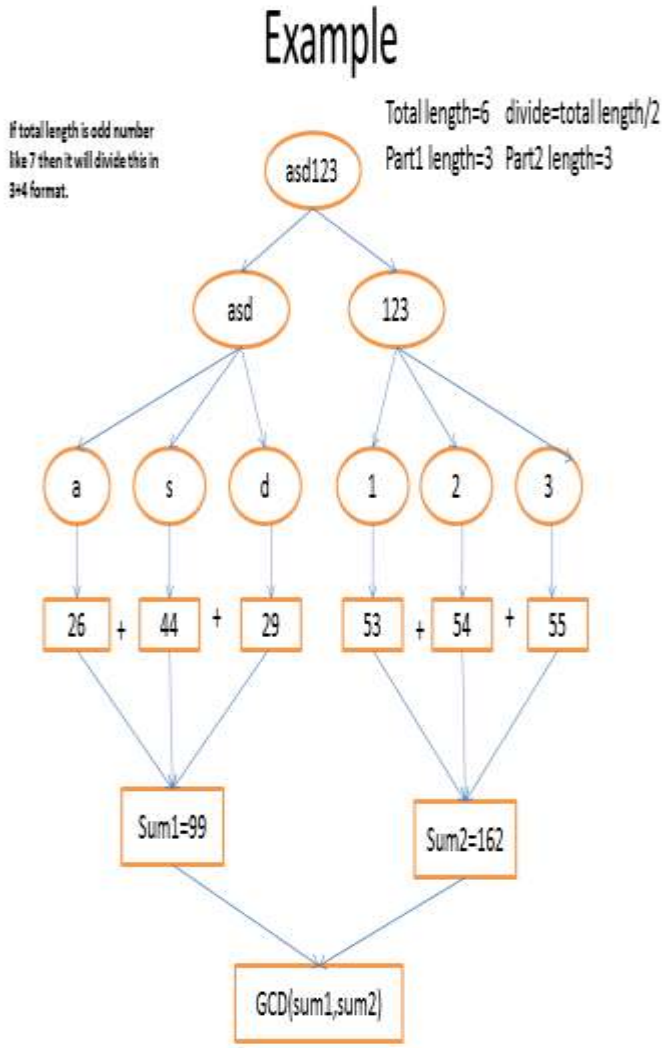
Below values contain diagram shows how this algorithm assumes values corresponding to all character. This is fix for all the time.

Here below diagram shows the performance of this algorithm.



Graph I: Performance of algorithm password length Vs. Strongness & overall performance

Below diagram shows how all characters are separating and assigning integer values corresponding to all characters as shown in Table1.



III. CONCLUSION

This algorithm proposes a new solution to reduce the computation and storage complexity while providing secure multimedia multicast through effective key management techniques. Using this method we can implement multiserver and multiclient communication with less time complexity. The proposed algorithm has two dimensional focuses—minimal computation complexity and minimal storage complexity. When the key size is small (key size=8 digits), the computation time decreases by 0.2 ms and when the key size increases (16 or 32 digits) the computation time decreases by 0.15–0.18 ms for updating a single key from any level of the clustered tree by using Karatsuba fast multiplication. With regard to the storage complexity, the amounts of keys stored by GC and group members are reduced substantially by employing the cluster tree approach. Further extensions to this work are to devise techniques for reducing the communication complexity which is the number of keys to be sent from GC to the group members'

area in order to recover the updated keying information and to reduce rekeying cost for batch join and batch leave operations.

IV.REFERENCES

1) Ng W. Hock Desmond, M. Howarth, Z. Sun, H. Cruickshank, Dynamic balanced key tree management for secure multicast communications, IEEE Transactions on Computers 56(5) (2007) 590–605.

2) Yun Zhou, Xiaoyan Zhu, Yuguang Fang, and MABS: "multicast authentication based on.batch signature, IEEE Transactions on Mobile Computing" 9 (2010) 982–993

3) Xianjin Fang, Longshu Li, On karatsuba "multiplication algorithm", in: Proc. 7th Int. Data,Privacy, and E-Commerce Symp., Washington, DC, USA, 2007, pp. 274–276.

4) Xukai Zou , Byrav Ramamurthy and Spyros S. Magliveras "A GCD attack resistant CRTHACS for secure group communications" Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'04) 0-7695-2108-8/04 © 2004 IEEE.

5) Zhibin Zhou and Dijiang Huang "an Optimal Key Distribution Scheme for Secure Multicast Group Communication" 978-1-4244-5837-0/10 ©2010 IEEE.

6) Jingwei Hu, Wei Guo, Jizeng Wei, Yisong Chang, Dazhi Sun A Novel Architecture for Fast RSA Key Generation Based on RNS "978-0-7695-4575-2/11 © 2011 IEEE DOI 10.1109/PAAP.2011.75".

7) Ajeet P Singh, Swapnil M Potey, Ferdous A Barbhuiya and Sukumar Nandi "A scalable and Secure Key Distribution Mechanism for Multicast Networks" 978-0-7695-4723-7/12 © 2012 IEEE DOI 10.1109/ICACC.2012.78.

First Author: Sathya S

I have completed my B.E in computer science and Engineering Degree from RANIPETTAI ENGINEERING COLLEGE, Vellore, India in 2011. Currently I am pursuing M.Tech In information Technology (Networking) at VIT University, Vellore, Tamil Nadu, India. My Major Research areas are Network Security, Theory of computation, Logic Design, wireless adhoc network, Operating System, Data Structure and Algorithm, RDBMS. I have Completed Successfully Research Project Entitled Channel Based Non Cryptographic Authentication in Wireless Networks.

Second Author: Ajay Kumar

I have completed my B.E in computer science and Engineering Degree from LNCT, Indore, and Madhya Pradesh, India in 2011. Currently I am pursuing M.Tech In information Technology (Networking) at VIT University, Vellore, Tamil Nadu, India. My Major Research areas are Network Security, Theory of computation, Compiler Construction, wireless adhoc network, Operating System, Data Structure and Algorithm, RDBMS. I have Completed Successfully Research Project Entitled rainbow table to crack password using md5 hashing algorithm. Currently I am doing my final project on Development of the Universal Automation Tester Framework in VMware.