JNNCE Journal of Engineering & Management, Volume 6, No.1, Jan – June 2022 ISSN 2582-0079
This paper is presented at National Workshop on Cryptology NWC 2021, held at JNN College of Engineering, Shimoga during 18-20 Nov 2021

Available online @ https://jjem.jnnce.ac.in
https:www.doi.org/10.37314/JJEM.2022.060115
Indexed in International Scientific Indexing (ISI)
Impact factor: 1.395 for 2021-22
Published on: 31July 2022

Cryptographic Key Management Using Elliptic Curve X448 for Multiuser Environment

Mohan Naik R[1], Sowmya T K[2]

[1,2]Department of Electronics and Communication Engineering, SDM Institute of Technology,

m.naik5785@gmail.com, sowmyatk@sdmit.in

Abstract

Key management is very essential part in the cryptographic field where the key must be maintained very secure so that the problem of maintaining of the key is important. The proposed method elaborates the much enhanced and actual key management system along with improved speed and less storage of the key. The policy of managing the key is mandatory for producing a key and required to take care to the storage as well as processing of it in the cloud scenario where security of the data is up most important thing when number cloud users are more in the network to use several applications. Making use of several algorithms by considering Diffie Hellman by combing with the Elliptic Curve will provide most effective in managing of the keys with much lesser key size. In this paper key exchange and shared key generation with Curve 448 is considered the practical results shows that effective key generation and exchange of the shared key. It obtains a 224-bit security level which delivers increased security over Curve 25519, and which has a 255-bit prime number. The proposed elliptic curves which are quicker and simpler to implement than other prime-order curves which are applicable for most of the applications. Hamburg chose the Solinas trinomial prime base $p=2^{448}-2^{224}-1$ which provides faster Karatsuba multiplication.

Keywords: *Cloud computing, Elliptic-curve-Diffie-Hellman, Curve X448.*

1. Introduction

Several applications based on cloud computing are used which is more connected to several concerns in late decades. So, it is crucial to promise the protection of the stored data into the cloud. Several methods were planned to protect the data as well as to provide the security of the redistributed information in the storage. The key management not only provides secrecy of the keys obtained but also provides much faster key generation with much improved security. The implementations considered in other curve have been for fields of size around 2^{256} making in security comparison with other existing algorithms [1]. The proposed curve used a design of an 448-bit field which provides effective and faster in key generation compared to curve 25519 which can also consider for generating and securing the cryptographic keys [2]. Since other existing algorithm for key generation and exchange are required to modify to make them more accurate in terms of speed and size of the keys used in the algorithm. In this paper, employs the key management methods to several ECDH shared key generation is discussed and executed. The results suggesting that curve 448 is faster in generating of the shared key which can be later used for any group messages.

In X448 which considers the use a Montgomery curve which uses 448bit (56 byte) prime number of $P=2^{448}–2^{224}–1$. It has better security compared to Curve 25519, and which has a 255-bit prime number ($P=2^{255}–19$). A Montgomery curve ($v^2=u^3+486662u^2+u$) with scalar

multiplication. In X448, we use 56-byte string values, rather than 32-byte values for X25519. The key size for reliable Discrete Logarithmic Problem based Diffie-Hellman has popular over time, so it is also positioned a heavier managing load in terms of speed and storage issues [3]. The attractiveness of Elliptic Curve Cryptography is considered to provide essential security with far reduced size in the keys thereby reduction in commutation overhead. In mobiles and other cloud computing tools that manages with respect to storage and accessing power is to be considered as compulsory for securing the any transmission of information as increases the exposure since the susceptibility to the opponent due to publicly accessibility of all these above also increases so it is the primarily important to detect and reduce the complexity in generating and processing of the keys.

2. Related Work

The various works has been undergone in connection with Elliptic curve Diffie Hellman key exchange protocol for generating of keys and soon after utilized for securing the information or any other applications. The figure.1 represents the Diffie Hellman Key Exchange algorithm in detail, and this will be elaborated in the case of untrusted cloud network where we use to process and store the data The algorithm provides the secret key which in turn further utilized for the encryption. Both the cloud nodes agree on two values G and n after that uses the random values x and y and generates A= G^x mod n and B= G^y mod n after that A and B are mutually exchanged to each other which in turn generate the k1 and k2 which are same secret keys generated and utilized further. In this algorithm the technique accepts two members where they do not have any sort of information about the key and each other to jointly determine a key over a vulnerable channel as shown in Figure 1.

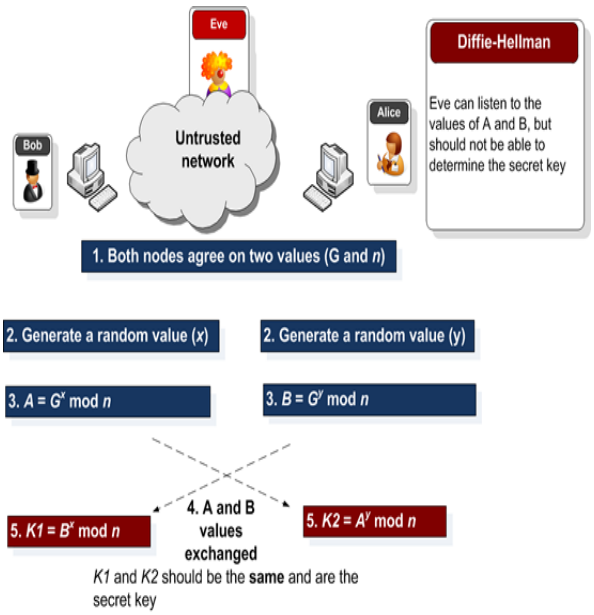

Figure1: Representation of Diffie Hellman Key Exchange Algorithm

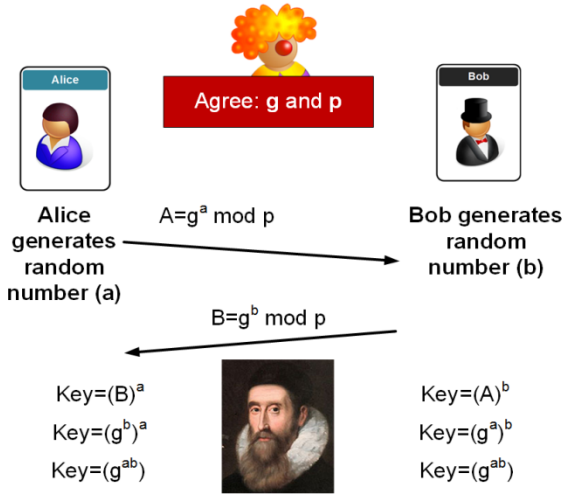

Figure 2: Representation of Diffie Hellman Key Exchange Algorithm to generate secret key k.

The figure 2 elaborates the secret key generation using random numbers. The Cloud Key Management Interoperability Protocol (CKMIP) establishes up a single comprehensive agreement for the use of interaction connecting to various servers and users of these assets. In most of the cloud applications where the data is to be process in the untrusted network so that the requirement of secrecy in data is mandatory.

Ivan Damgrd et.al.,[4] describes about the managing of the keys on the server as well as machine side which is having the condition in which the server has be work and process only when it is required to do so. Atulbhai Patel et.al., [5] represents cloud computing security which are considered by utilizing the key management includes every one of the delicacies of the procedure to deal with keys whenever it is essential. Ching-Nung Yang et.al.,[6] describes securing the data without losing or disturbing the content in that the main event primarily uses the concept of Elliptic Curve Diffie-Hellman algorithm (ECDH) [8][9][10]. The benefits of unique Montgomery primes is generally in carry proliferation on full-radix operations but primes larger than about 2^{256} favor vectorized growth on ARM and decreased-radix proliferation on x86-64.

Curve25519 [7] proposes and examines the curve 25519 the function suitable for a wide variety of applications by considering the ECDH algorithm and proves that curve 25519 has faster in generating required keys and high security DH computations.

3. Mathematical Prerequisites

a. Elliptic Curves

An elliptic curve is described across a field K and describes points in K^2 the Cartesian product of *K* with itself. If the field has characteristic different then the curve can be defined as a plane algebraic curve which, after a true change of variables, consists of solutions (*x*,*y*) to:

$$E: y^2 = x^3 + ax + b \quad (1)$$

for some coefficients a and b in K. The curve is expected to be non-singular, which means that the curve has no cusps or self-intersections.

There are enormous literatures about elliptic curves. Elliptic Curves is an performing element that can be described as lightly looping lines in the (x,y) plane. Elliptic Curves can be given by using Weierstrass equation[14].

$$E: y^2 + u_1xy + u_3y = x^3 + u_2x^2 + u_4x + u_6 \quad (2)$$

The points on an elliptic curve over an arbitrary field K can be defined as the set of solutions (x,y) of equation (1) [13], The addition operation of the group is based on the geometric properties of elliptic curve. Point at infinity O is additive identity. Some basic facts of Elliptic Curves are discussed below. The curve coefficients in Edward is of the form

$$Ed: y^2 + x^2 = 1 + dx^2y^2 \quad (3)$$

In Equation (3) Ed and its twist together have 4·prime order, and because of that the order of the curve is fewer than p, which is primary advantage as compared to any other techniques.

b. Discrete logarithm problem

In mathematics, for given real numbers p and q the logarithm $\log_q p$ is a number y such that $q^y = p$. Analogously in any group G, power q^k can be considered for all integers k. The discrete logarithm $\log_q p$ is an integer k such that $q^k = p$[11][12].

c. Elliptic Curve Discrete Logarithm Problem (ECDLP)

The security of Elliptic Curve Cryptography lies on the idea that the Elliptic Curve discrete logarithm problem to be much harder.

In most of the cloud computing applications the required security lies on the selection of the type of the Elliptic Curve for efficient utilization for generation of keys which in turn used for cloud data encryption.

For the decryption same shared key used for the successful data retrieval that has stored in

the cloud.

Curve 25519 based shared key generation discussed in [2] which gives clear indication of generating the keys based on the curve 25529 which makes faster computation due to its structure which considers scalar multiplication. The paper [2] also shows the results based on the curve 25519 and effectively utilization of the curve for generation of the shared key and proved the security aspects required for the cloud data encryption and decryption of the data based on these obtained keys. The paper [2] also elaborates the implementation of the curve which is compared to several existing algorithms and justifies the results with respect to different keys size and with several sample results. The nature of this result obtained is based on faster computation in x-coordinate point operation.

The comparison of paper [2] with proposed algorithm provides the computation with selected curves for implementation in different environment based on the utilization of the curve. The size of the keys generated in ECDH is having the size of 64 KB and it is high as compared to generated key in paper [2]. The proposed algorithm based on Curve X 448 gives much lesser memory of the generated shared key as compared to normal ECDH key size.

Table No. 1 symbolizes the implementation time for shared key generation. In X 448 uses 56-byte string values as compared to 32-byte value of the curve 25519. The memory of the generated key for comparison between Curve 25519 and Curve X448 is represented in Table No. 3.

Table 1 :The Implementation time (in sec) for shared key generation in paper [2] and Curve X448

Number of Samples taken	Shared key generation time in paper [2]	Curve X 448
Sample 1	0.049	0.032
Sample 2	0.051	0.022
Sample 3	0.048	0.012
Sample 4	0.049	0.012

Table 2: The comparison of shared key memory in Kilo Bytes used between paper [2], Curve X 448 with ECDH

Parameters	Paper [2]	Curve X 448	ECDH
For secret keys	32	56	64
For public keys	32	56	64

Table 3: The comparison of size of shared key generated memory in Kilo Bytes in paper [2] and Curve X 448 .

Number of Samples taken	Paper [2]	Curve X 448
Sample 1	4634	4434
Sample 2	4517	4436
Sample 2	4575	4516

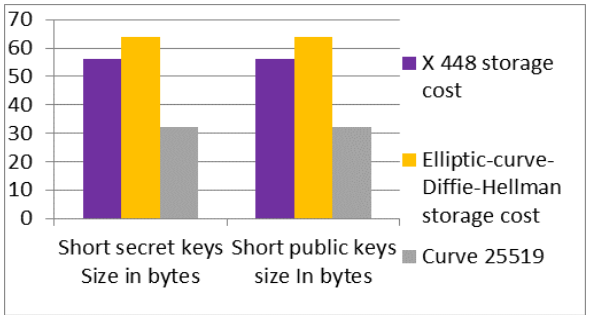

Figure 3: comparison of Storage cost for ECDH, Curve 25519 and Curve X 448

The implementation time in ECDH is high as compared to Curve 25519 which provides the 160-bit security level. The Curve 25519 has good security level with much more implementation speed. The Curve X448 has 224-bit security with reduced processing speed which may causes system considering for cloud applications are much suitable for this. Some of the cloud application where memory

is essential parameter since some of the platform it is based on the payment for each usage of the memory. In X448 it uses Montgomery curve ($v^2=u^3+486662u^2+u$) with scalar multiplication.

4. Experimental Results

The experimental results are performed in Intel i5 processor 8GB ram in Python 3.8.2 the results based on the implementation time and memory of the generated shared key is illustrated in Table No. 1 and Table No. 3. Here considering implementation parameters, for example, storage memory and implementation time. ECC provides the much-reduced key size with required security [13][14]. From Figure No. 3 representing Storage cost for comparison between ECDH and Curve 25519. In Table 1 shows The Implementation time (in sec) for shared key generation in paper [2] and Curve X448 the data shows implementation time of the Curve X448 which is much lesser which is highly recommended in most of the cloud data encryption and processing of the data where speed and storage memory is at most important. Figure No. 3 shows comparison of Storage cost between ECDH, Curve 25519 and Curve X 448. Figure No. 4 shows Comparison of implementation time in sec vs various other Algorithm with Curve X448 which clearly indicating that even with the higher length of key in Curve X448 it provides the faster shared key generation and reduced memory size.

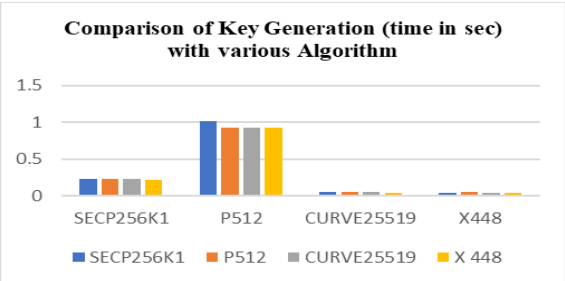

Figure No. 4 - Comparison of implementation time in sec vs various other Algorithm with Curve X448.

Sample 1

```
User 1 private key:
b'5dce2f82a5662f5d8648257b01e1ec8d235e4616c401564573cb76eeaad966ce'

User 2 private key:
b'8f45744a360002d4d3f53f36f0e2e180e60cae2a706bcbe2da97f2ff2df7987f'

User 1 public key:
b'c389c3ac000457c3a60d2ec2b26dc286c28c2dc384c38ec2b61d5844439c39cc38fc3a5c3b75fc390c2b07d56
503d58'

User 2 public key:
b'c29e75c288c38a0f0dc28a0dc293c2bf30c29dc3bb4c4010c38f6a6c5a45c39ec397c3866c0ec283c3b61936c
2a01d'

User 1 shared key:
b'c3ac3aa70c3a2c3824155754142c3aec3bbc2b276c396c390c383c285094cc396394fc3905a09c28dc380c2b6c
38b0e49'

User 2 shared key:
b'c3a2c3aa70c3a2c3824155754142c3aec3bbc2b276c396c390c383c285094cc396394fc3905a09c28dc380c2b6
c38b0e49'

Executed in 0.044 sec(s)

Memory: 4424 kilobyte(s)
```

Sample 2

```
User 1 private key:
b'08373160de1f9fa24e3c2eefee99163cb3b28959550fada6a14a618e64c11185'

User 2 private key:
b'20cbcea11850f4d41f87c9a56ddc213cc4a0da67a62f38896cdb360cc566c488'

User 1 public key:
b'5d74c2a81f6d5cc2a4c2b560c2b25b00c3a854c3b169c39048c39ac395c2986bc2b4c281c3a92962661ec388
c2ad78'

User 2 public key:
b'c3adc28c523cc38221c28bc3bbc3abc2b4c29b5ac2a6c298c38974c387c3adc38b3e4813c29f71c28e2ec2b8
0350702967'

User 1 shared key:
b'c28475c2ba4e283e52c38e11c3aec28cc3b74f42c3be6dc2801ac384c2a4c28f1fc2a6c3927d2a3e69c2b7
20c3af55'

User 2 shared key:
b'c28475c2ba4e283e52c38e11c3aec28cc3b74f42c3be6dc2801ac384c2a4c28f1fc2a6c3927d2a3e69c2b
720c3af55'

Executed in 0.034 sec(s)

Memory: 4524 kilobyte(s)
```

Sample 3

```
User 1 private key:
b'7f5adb04c5b68ba78c92c8d4a1792d4688d07eb23e78030615eef110aaf90505'

User 2 private key:
b'09c3dc24597f3c2c30eea6301feb005e60b6990a9aa544bd5f429a01d25cf12a'

User 1 public key:
b'c3860fc2b6c3a643c2957bc398c39dc388c283687a165d1d61c2bd55c2a163517a4768c3931ec2b84437c3bd3
9'

User 2 public key:
b'c2860103c384c397c3b4c296c3a4c2987ac2b9c299c2bc06c3a6c2b5656ec282c385c28903c3be0dc28ac3bbc
3a6c2a50c77c2b86d'

User 1 shared key:
b'57030015c2bcc2bfc2acc38371c3a07ac39371c38dc392c29d70c2be6bc38f565dc297c2a5c3a8c3a8325e720b
5611'

User 2 shared key:
b'57030015c2bcc2bfc2acc38371c3a07ac39371c38dc392c29d70c2be6bc38f565dc297c2a5c3a8c3a8325e720b
5611'

Executed in 0.053 sec(s)

Memory: 4436 kilobyte(s)
```

Figure 5: Representation of execution time and Memory of shared key generation using curve 25519.

Sample 1

```
User 1 private key:
b'2a9e1f15f57144a7946cb2d4ba61d9e226dadb8d28b90f88dda052df81a0e0edfa9de700fdc433980c1
c5b45f4816c25b2288c623a6bd1e8'

User 2 private key:
b'90b09d874f1262e6ca80ab6034105d961e40a28a371bf7820c1b18d83a936fa1372887cb42955419db8
c478b2edccc505508a393ef2a734b'

User 1 public key:
b'566a4e0c6a5bc2bcc386c39c366fc39dc387767dc382c392c2a61cc2bdc29ec2b0273ac38dc2a9c2b17
40fc2946ec2bec28cc3bf59c28a2a43c39308c2aec2ae56527ac2aec3a114c2aec3b2c3b02ac28420c285
c294'

User 2 public key:
b'37291b6d5bc38a2bc3bd5fc388c29b40c2ba18690159c2a40a47c285c390c2807ec3b8c281c3aa00c29
528c3aec296c291c3917314c38c653c154fc3b9131dc3b973c2b7c293c29a13c2b71dc3b6c3914cc3ad'

User 1 shared key:
b'22c2ad27c3b5c28665c3884dc3a7c38dc390c28ec3956ec3bb24c2b071c2b67b2433c3aec3bec2a52ec
29bc3bf5641470ac2a06323354a54c2be70c295c3a56b4838c2abc2abc290c292c29ac2814ec3ae0a586c
'

User 2 shared key:
b'22c2ad27c3b5c28665c3884dc3a7c38dc390c28ec3956ec3bb24c2b071c2b67b2433c3aec3bec2a52ec
29bc3bf5641470ac2a06323354a54c2be70c295c3a56b4838c2abc2abc290c292c29ac2814ec3ae0a586c
'

Executed in 0.053 sec(s)

Memory: 4280 kilobyte(s)
```

Sample 2

```
User 1 private key:
b'6c60d376729e93973f0cc9379859e05dc28ff41b7c2c683c8aae7fd33a9847988ff4351a0fa08ba6424
54436270dea2f19018160464a4c25'

User 2 private key:
b'356d692dc3364d2648d0d0b25f1ffdec4bdc4576a865528eb12516ec6f1bfe7ace56ec66c9136ac1a3e
86af1120491bb6479972f809355b2'

User 1 public key:
b'c28f0dc391c285c39a79147ec2b65fc2ab1759c288c3842ac2a205c29e4e65c2812f315114c2a57b02c
3ab23c3a2257169c2b17cc29c5ec288c295c2bd14c3a6c3bdc39bc280c3b035c3a5c3bdc2a2584613c2a9
'

User 2 public key:
b'c29b296054c39ac2b4c3afc3852469c3a5c3bd60c3a9036a453f51c29a49c286c3ad1ac2ae46c28bc28
4676614c2b7c2954e75134b6e46c297414a09c2b6c38f455bc38b4ac2bf22c28dc387c2afc3ba7b'

User 1 shared key:
b'501459c3b3c297c29d50c3ab5cc3ae1a4dc2970ac3b458c2b5c2953d1a2ec2b9c292c3ab506fc3b4261
17bc2b97a26c281c285c3adc3b3c39538c2ac0b3817285ac3acc3abc383c3bcc297c28cc3a143c3b046c2
93'

User 2 shared key:
b'501459c3b3c297c29d50c3ab5cc3ae1a4dc2970ac3b458c2b5c2953d1a2ec2b9c292c3ab506fc3b4261
17bc2b97a26c281c285c3adc3b3c39538c2ac0b3817285ac3acc3abc383c3bcc297c28cc3a143c3b046c2
93'

Executed in 0.051 sec(s)

Memory: 4316 kilobyte(s)
```

Sample 3

```
User 1 private key:
b'3750427e2514f40c087beb758112184350e949d156f575ffe594076a31ef319828fa1c53eb82db875b2
0db421f1c2454452871cd87cbf126'

User 2 private key:
b'ec4035e0ce446da3f9b4423c83ad66561303d19ba06ac2a306b68c4982e9d5262e7c19deb6a80fff587
f139c77a001274658b9c5fb331d4a'

User 1 public key:
b'c2a2c387c3b41307c39a5630735c2ac2a169c3bdc287c38f3d4a34c3a97c646c083859c3b8c3bac284c
280c38d2e7d5443c288c280552e364dc3905a002a09c3885ec3aa4e5d54c380c3801c3f'

User 2 public key:
b'4b2637c28fc29b3bc292c2b7c283440727c2a1c29ec2b8c29ac3bbc28b327bc384c396c2833ec2ab7bc
292357bc2bf14621d08c2bac3b83e5ec2a7c382c3bf5b35645dc2955dc28d1bc29e6bc2a8c3a7343e65'

User 1 shared key:
b'14c2a6c3903268c3b1c29bc38fc295c3a50f6435c380c2ad025307c2a6c2adc3b7c2b374c38965c28e3
06cc281c38cc28b62c28427c2992c4444c39bc2890e5e1d4d0bc2890c1b52c2bec39f2a08c298c295c39a
'

User 2 shared key:
b'14c2a6c3903268c3b1c29bc38fc295c3a50f6435c380c2ad025307c2a6c2adc3b7c2b374c38965c28e3
06cc281c38cc28b62c28427c2992c4444c39bc2890e5e1d4d0bc2890c1b52c2bec39f2a08c298c295c39a

Executed in 0.043 sec(s)

Memory: 4344 kilobyte(s)
```

Figure 6: Representation of execution time and Memory of shared key generation using Curve X448.

5. Conclusion

Since the size of the encrypting key as well as maintaining the key in the cloud storage is most important aspect in the cloud computing because the sensitive data and important information has been shared among the cloud networks. The RSA key management is considering a key size of 1024 bits which makes heavier processing as well as communication cost. ECDH provides much reduced key size with necessary security involving it but recent time the shared generated by using ECHD has increased its storage cost due to its size and structure, so ECDH curve 25519 affords required much quicker shared key creation with smaller storage memory cost. Curve Ed448-Goldilocks is some 17% faster for key generation comparing for ECDH. The shared key obtained from the Curve X448 is providing 224 bits security level with faster shared key generation with much reduced storage cost. So, the cloud storage network where speed and memory are the fundamental aspect the Curve X448 provides solution to it.

References

1. Daniel Bernstein, Peter Birkner, Marc Joye, Tanja Lange, and Christiane Peters. Twisted edwards curves. Progress in Cryptology–AFRICACRYPT 2008, pages 389–405.

2. R. Mohan Naik, S.V.Sathyanarayana and T.K. Sowmya, Key Management Using Elliptic Curve Diffie Hellman Curve 25519, 2020 Third International Conference on Multimedia Processing, Communication & Information Technology (MPCIT), 2020, pp. 33-39,
doi: 10.1109/MPCIT51588.2020.9350454.

3. Yong Yong Wang, Byrav Ramamurthy, Xukai Zou, The Performance of Elliptic Curve Based Group Diffie-Hellman Protocols for Secure Group Communication over Ad Hoc Networks,.IEEE International Conference on Communications ICC '06, Vol.5, 2006, pp. 2243 – 2248.

4. Damgard I., Jakobsen T.P., Nielsen J.B., Pagter, Secure Key Management in the Cloud. In: Stam M. (eds) Cryptography and Coding". IMACC 2013. Lecture Notes in Computer Science, Vol. 8308. Springer, Berlin, Heidelberg. J.I.2013

5. Dr. Atulbhai Patel and Kalpit Soni, Cloud Computing Security using Federated Key Management, International Journal Of Engineering And Computer Science, Vol.3, No. 2, 2014, pp. 3978-3981.

6. Yiling Wang. Key Management for Secure GroupApplications in Wireless Networks, Ph.D thesis, Faculty of Information Technology Monash University.2008.

7. Bernstein, Daniel J. Curve25519: new Diffie-Hellman speed records. In International Workshop on Public Key Cryptography, Springer, Berlin, Heidelberg, 2006, pp. 207-228.

8. S. Singh and V. Kumar, Secured user's authentication and private data storage- access scheme in cloud computing using Elliptic curve cryptography, 2nd International Conference on Computing for Sustainable Global Development (INDIACom), New Delhi, 2015,pp. 791-795.

9. Pandi Vijayakumar and Ramu Naresh and Lazarus Jegatha Deborah and SK Hafizul Islam, An Efficient Group Key Agreement Protocol for Secure P2P Communication, Security and Communication Networks, Vol.9, 2016, pp:3952–3965. DOI: 10.1002/sec.1578.

10. Sandro Rafaeli, David Hutchison, September, A Survey of Key Management for Secure Group Communication, ACM Computing Surveys, Vol. 35, No. 3, 2003, pp. 309–329.

11. Steiner, M., Tsudik, G., and Waidner, M. Key Agreement in Dynamic Peer Groups, IEEE Transactions on Parallel and DistributedSystems, Vol. 11, Issue 8, 2000, pp.769-780.

12. Hilyati Hanina Zazali, Wan Ainun Mior Othman, Key Exchange in Elliptic Curve Cryptography Based on the Decomposition Problem, Sains Malaysiana, Vol. 41, Issue 7, 2012, pp.907–910.

13. Koblitz ,N. A Course In Number Theory And Cryptography. New York: Springer-Verlag,1994.

14. Shalini I S, Mohan Naik R, and Dr.S V Sathyanarayana, A Comparative Analysis of Secret Sharing Schemes with Special Reference to Group Communication Applications, IEEE International Conference on Emerging Research in Electronics and computor sceince and Technology (ICERECT-2015). Vol 1, Issue 2, Dec 2015.