ISSN:0975-9646

A Secured Multicasting Key and Data Exchange By Using Extended Chebhysev Map

Jyothsna Ivaturi
M.Tech Student,
Dept of cse,AIET

K.Ravindra
Asst. Professor,
Dept of CSE, AIET

Y.Ramesh Kumar
Asst. Professor,
Dept of CSE, AIET

Abstract: The Chebyshev map is a typical chaotic map which has been widely investigated for cryptography However, researchers argue that almost all the encryption algorithms based on some chaotic maps are not as secure as they are announced. In this paper, the limitations of some multicast key agreement schemes are discussed. To eliminate those false and remain novelty, we consider how the chaotic maps can be improved to fulfill the cryptographic demands. To achieve this, we extend the domain of Chebyshev maps from real number set, to finite field. Based on this extension, this paper proposes a novel multicast key exchange algorithm. By detailed experiments and analysis, we show that this algorithm is secure and efficient. As far as we know, such a multicast key exchange algorithm has not been reported. The results given here, to the best of our knowledge, are novel.

Keywords: key exchange, encryption, decryption, cryptography.

INTRODUCTION

Multicast key management, which is much different from unicast key management, is one of the most attractive area of cryptography. For an uncast application, the Diffie-Hellman key exchange protocol can be employed to establish a KEK (Key Encryption Key) between two entities. Then use this KEK to dispatch or update a session key. In contrast, the situation is much more complicated for a multicast application. A multicast application must dynamically handle multi-entities. For example, in a dynamic multicast group, the membership is changeable all the time due to frequently users' addition and eviction. Therefore, the key materials will probably be revealed if no security policies are adopted. For instance, if the key is not updated after the membership change, a new comer is able to read the contents before his coming, or a evictor is capable of reading the content after his leaving. In this case, multicast key management scheme should provide forward secrecy and backward secrecy for security reasons in some special applications, e.g. Pay-Per-View. In the past two decades, researchers have proposed many multicast key management schemes. These schemes can be categorized into three different types: centralized, decentralized and distributed. A centralized group key management scheme involves a Key Server (KS) to generate and distribute shared key to all group members via a secure channel. A decentralized key management divides the whole group into smaller subgroups. Each subgroup is controlled by a single or several KS. A Distributed scheme allows each member to take part in a group key generation collaboratively. Each of the three schemes has its own advantages and disadvantages. Centralized scheme is the simplest one but has the risk of single-point-failure. Decentralized scheme adds some communication complexity between two members within different subgroups.

Distributed scheme is somehow more complex than the other two, but it doesn't involve KS. This feature is very useful in the case of no one can play the role of KS, e.g. a sensor Ad-hoc network application. In this paper, we deeply analyze the multicast key management scheme proposed in and figure out its fatal drawbacks. To improve the results obtained in , a new distributed multicast key exchange algorithm is proposed. By theoretical analysis, this algorithm, based on the extended Chebyshev map has the same security level as Diffie-Hellman or Group Diffie-Hellman key exchange protocol.

Already a no key exchage alogarithms are exist. some of them we included here a multicast key exchange algorithm based on the Chebyshev map. The recursive definition of Chebyshev map is given by:

$$T0(x) = 1\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(1)$$

$$T1(x) = x\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(2)$$

$$Tn(x) = 2xTn\text{-}1(x) - Tn\text{-}2(x)\ldots\ldots..(3)$$

where $x \in [-1;\ 1]$. The Chebyshev map has a semi-group property: For any positive integer number $r;\ s$ and real number $x \in [-1;\ 1]$, this equation is always true.

$$Trs(x) = Tr(Ts(x) = Ts(Tr(x))\ldots\ldots\ldots\ (4)$$

Algorithms existed for key exchange is

***1**: **Group Key Initiation**.*

Input : *n; x; ri; ki*

Output : The initial group session key *k*.

Method:

1. Each member sends *ri* to the KS.
2. KS computes *Tri* (*x*)*; i* = 1*;* 2*; · · · ; n*, encrypts all the results by pair-wise key *ki* separately and pushes back to all members.
3. KS pushes random numbers *ri; i* = 1*;* 2*; · · · ; n* back to all members by multicast.
4. After each member received corresponding *{ri}* and the encrypted message, they are able to do decryption using pairwise key independently. 5. Everyone can compute the group session key according to the semi-group properties

This algorithm is designed for setting up a group session and establishing an initial key shared among all members. It works as a register protocol. Any user intending to join in a secure multicast group must run this algorithm to obtain key materials, which is involved in algorithm 2 and 3. A good secure multicast key management scheme must take the forward secrecy and backward secrecy into consideration: The session key have to be updated right away whenever the group membership change is detected. Algorithm 2 and 3 are designed for dealing with such cases: user's addition and eviction

2. Member Addition

Input : *n; x; ri; ki*

Output : The updated group session key *k*.

Method :

1. The new joint member *un*+1 sends *rn*+1 to the KS.
2. KS computes and encrypts Trn+1(x) using pair-wise key kn+1, then feeds back to un+1.
3. KS pushes all random numbers ri; i = 1; 2; · · · n+1 back to all members by multicast.
4. After received all {ri}; i = 1; 2; · · · n+1, every member can calculate the new session key according to the semi-group property. The algorithm 3 is so simple that only one message is involved: The KS multicasts the random number of the leaving user (assume ui). Each user re computes the new group key after received this message.

Member Eviction:

Input : n; x; ri; ki

Output : The updated group session key k.

Method :

1. KS multicasts the random numbers *ri* of *ui* to all members
2. After received *{ri}*, everyone is capable of computing the new session key. The "theoretical correctness" of the above three algorithms can be found in and excluded here. To make this paper more briefly, the details of another proposal named "JGKM" based on Jacobian Elliptic Rational Map are also omitted here, but can be found in . However, it is very pertinent to mention that, the "JGKM" is nearly the same as the Chebyshev-based algorithm. LIMITATION OF THE ABOVE ALGORITHMS although the above algorithm is very novel, as the author announced, efficient and secured, it has a fatal drawback, which causes it basically impractical. The basis of the above algorithm is semi-group property, which is always true for Chebyshev map theoretically. However, we must notice that, on one hand, Chebyshev map is defined over real numbers and sensitive to initial conditions. On the other hand, computer can only do approximate other than precise computation. Therefore, computer cannot do "*real*" chaotic computation since it is a common sense that a × b × c b × c × a if *a; b; c* are real numbers. For example, if the parameters of the Chebyshev map are set to be: *r* = 68*; s* = 96 and *x* = 0*:*39, it is easy to verify that *Tr*(*x*) =−0*:*513634, *Ts*(*x*) = 0*:*723788, *Tr*(*Ts*(*x*)) = 0*:*0528869,*Ts*(*Tr*(*x*)) = 0*:*0524104, *Trs*(*x*) = 0*:*0523997. Obviously, *Tr*(*Ts*(*x*))*Ts*(*Tr*(*x*))*Trs*(*x*). This is inconsistent with the theoretical result. By some advanced programming skills, this error can be wiped off. However, it will cost too much time and space. The same problem also lies in the proposal . Besides, we must argue that the author announced of the security of above algorithms is grounded on two "assumptions

1)Secrecy of *x*: *x* is a secret seed; no one knows the value of *x* except KS.

2) One-way property: it is very easy to compute *Tr*(*x*) if *x* and *r* are known. But it is nearly impossible to compute *x* according to *Tr*(*x*) and *r*. Unfortunately, the second assumption is not correct. The explanation is: The Chebyshev polynomial has another equivalent definition

Tr(x) = cos(r arc cos(x)) ……………(5)

Therefore, theoretically, *x* can be obtained easily by:

x = cos(arccos(T)/r) ……………….(6)

During the whole process of key updating, *Tri* (*x*) is encrypted by pre-shared pair-wise key *ki* and can be uniquely decrypted by *up*, any outside adversary hardly can resolve *x*. However, *x* is resolvable for any member in current group according to (6), which implies all of the above three algorithms do not provide forward and backward secrecy. Now that *x* is resolvable, these algorithms are not secure any more since the security is completely based on the secrecy of *x*. Due to the similar reason, the algorithms based on the Jacobean Elliptic Rational Map is not secure neither. The recursive Jacobian Elliptic Rational Map with modulus *k* is defined as:

Rn+1(w; k) =2wRn(w; k)/1 − k2(1 − w2)(1 − R2n(w; k))−Rn1(w; k)…………..(7)

where w ∈ [−1; 1]; k ∈ [0; 1] and R0(w; k) = 1;R1(w; k) =w. Obviously, the Chebyshev map is a particular case the Jacobian Elliptic Rational Map when k = 0. Another equivalent definition of equation (7) is

Rn(w;k)= cn(n.cn1(w; k); k) ……………………(8)

where cn(w; k) is the inverse function of the elliptic integral with modulus k, that is

w =∫ 1cn (w;k)dv √(1 − v2)[(1 − k2) + k2v2] …………(9)

After investigated the Jacobian Elliptic Rational Map's semi-group property, The author of [2], [3] proposed a series multicast key managements named "JGKM" which is very similar to the algorithms mentioned in section II. As we claimed, it is not as secure as it announced. Again, this is mainly because an inside adversary is able to resolve the secret seed *w* according to equation 9.

2. METHODOLOGY

The above algorithms are unsuccessful due to two reasons: first, the Chebyshev map (or Jacobian Elliptic Rational Map) is defined over real number area, which results in not all members can compute the same key. Second, the secret *x* (or *w*) can be resolved. In this paper, we proposed a new multicast key exchange algorithm based on the extended Chebyshev map. In other words, the Chebyshev map in this paper is defined over a finite field. This is not a new idea, by surfing the Internet, several other references can be found in the literature However, those articles do not answer the efficiency of the algorithms which use Chebyshev map over finite filed. In this paper, we give a deep insight of the Chebyshev polynomial defined over a finite field. The performance is also analyzed the security of the algorithms 1- 3 are based on the secrecy of *x*. Thus, those algorithms are vulnerable since *x* is easily to be solved The most significant difference of the new algorithm is its security grounds on the open problem – discrete logarithm, which will be analyzed as

Definition:

Let T(N) : {0;1;N − 1} → {0;1 ;N −1}
is a self-mapping, the extended Chebyshev *Tn*(*x*) is defined as:
T0(x) = 1 mod N
T1(x) = x mod N
Tn(x) = 2xTn-1(x) − Tn-2(x) mod N …………..(10)
Where x ∈ {0; 1; 2; · · ·N − 1} and N is a prime. In some literature, it is reported that N does not have to be a prime; it can also be a product of two large primes. However, from security reasons, N should be a large prime and N + 1 should have a large prime factor. Especially, if we define the equation over GF(N), and force x to be the primitive root modulo N, then the recursive Chebyshev map can be transform to non-recursive form:
Tn(x) = c1qn1 + c2qn 2 mod N ………………(11)

Where q1 and q2 are the roots of equation and c1, c2 satisfy c1 + c2 ≡ 1, c1 ≡ c2 mod N.

A. Semi-Group Property:

The Chebyshev map over finite filed still remains the semi group property.
Trs(x) mod N=Tr(Ts(x) mod N) modN= Ts(Tr(x) mod N) mod N(13)
For example1, let x = 13;N = 41; r = 4; s = 5, It is easy to verify that T4(13) = 38, T5(13) = 29; T20(13) = 40 and T5(38) = 40; T4(29)= 40. Obviously, T4(T5(13)) =T5(T4(13)) = T20(13).

B. Periodicity of Extended Chebyshev Map

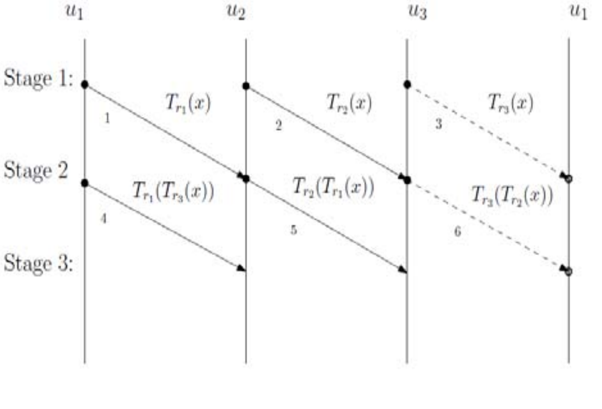
Due to the domain of the Chebyshev has been changed , the new map does not have chaotic properties, instead, it is periodic. For example, if the parameters are set to be:
N =41; x = 12, the results of each iteration are listed as below:
T0(x) = 1 T1(x) = 12 T2(x) = 0 T3(x) = 29 T4(x) = 40 T5(x) = 29 T6(x) = 0 T7(x) = 12
T8(x) = 1 T9(x) = 12 .
The periodicity of the sequence is 8. Generally, the periodicity of the extended Chebyshev map satisfies the following theorem [8]:
Theorem: Let N be an odd prime and x ∈ Z such that 0 ≤ x < N. Let t be the preiod of the sequence Ti(x) mod N for i = 0; 1; 2; Let _2 − 2x_ + 1 have roots q1,q2.
Then (i). t|N − 1 if the roots are in GF(N), otherwise,(ii). t|N + 1 when the roots are in GF(N2). From the above theorem and experimental results, we know that the sequence of the extended Chebyshev map is indeed periodic. Any sequence with short periodicity mustn't be applied in cryptography. However, by strictly choosing parameters N and x, the periodicity can be forced as long as it is required, e.g. 2^256 or even larger

THE NEW MULTICAST KEY EXCHANGE ALGORITHM:

The goal of the multicast key exchange algorithm can be expressed as follows: By exchanging messages over un trusted network, multi-entities are able to compute the secret share key independently. During the entire process, no one is responsible for the key generation or distribution. Instead, all members play an important role in this interaction. The main advantage of this kind of protocols (e.g. Diffie-Hellman protocol) is that they remarkably reduces the cost of key distribution. Ke Qin et al were the first researchers who attempt to design multicast key exchange protocol using Chebyshev map or Jacobian Elliptic Rational Map. But unfortunately, those proposals, as presented in [2], [3], was not correct and secure. However, the main concept introduced in this literature is the ground-breaking attempt to apply chaotic map to multicast key exchange protocol design.

Algorithm. 4: Novel Multicast Key Exchange Algorithm:

Input : n: the number of current group members.
{ri}: A set of "private " integer number, which can be treat as the "private" key of member ui.
N: A public large safe-prime number.
x: A "public" random integer number.
Output : The group session key k.
Method :
Stage 1:
(1) The first member computes Tr1 (x) and sends it to the second member.
(2) The second member computes Tr2 (x) and sends it to the third one.
(3) Repeat until the last member computes Trn(x) and sends it to the first member.
Stage2:
(1) The first member computes Tr1 (Trn(x)) and sends it to the second member.
(2) The second member computes Tr2 (Tr1 (x)) and sends it to the next.
(3) Repeat until the last member computes Trn(Trn1(x)) and sends it to the first member.
Stage i:
(1) The first member computesTr1(Trn(Trni+2(x)))and sends it to the second member.
(2) The second member computes Tr2(Tr1 Trni+3(x))) and sends it to the next.
(3) Repeat until the last member computes Trn(Trn1 (· · · Trn□i+1(x))) and sends it to the first member. By n − 1 stages message exchange, any member, e.g. the ith member compute the group session key by:
Tri (Tri1 (T1(Tn(Tn1(· · · Ti+1(x)))))) which is equal toTr1r2___rn(x)



3. ADVANCED ENCRYPTION STANDARD ALGORITHM (AES):

The Advanced Encryption Standard (AES) is a specification for the encryption of electronic data established by the U.S. National Institute of Standards and Technology (NIST) in 2001[8]

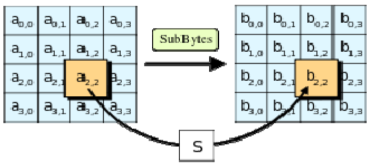
AES is a block cipher, but it does not use a Feistel structure. The block size of AES is 128-bit, but the key size may differ as 128, 192, or 256 bits [9].

Substitution: This method substitutes each byte of the block in the order of S-box. It provides an invertible transformation of blocks during encryption, with the reverse during decryption.

Shifting Rows: This operation performs left circular shifts of rows 1, 2, and 3 by 1, 2 and 3,

Mix Columns: This method multiplies each column of the input block with a matrix. The multiplication operation is just like matrix multiplication, except that it uses a Finite Field to multiply two elements and performs an XOR operation instead of addition.

Add Rounded Keys: This operation just applies an XOR operation to each byte of the input block and the current weight (key) matrix.

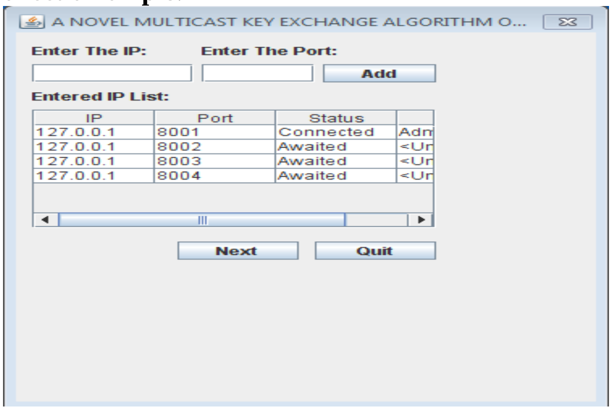


$a_{0,0}$ $a_{0,1}$ $a_{0,2}$ $a_{0,3}$
$a_{1,0}$ $a_{1,1}$ $a_{1,2}$ $a_{1,3}$
$a_{2,0}$ $a_{2,1}$ $a_{2,2}$ $a_{2,3}$
$a_{3,0}$ $a_{3,1}$ $a_{3,2}$ $a_{3,3}$
SubBytes
$b_{0,0}$ $b_{0,1}$ $b_{0,2}$ $b_{0,3}$
$b_{1,0}$ $b_{1,1}$ $b_{1,2}$ $b_{1,3}$
$b_{2,0}$ $b_{2,1}$ $b_{2,2}$ $b_{2,3}$
$b_{3,0}$ $b_{3,1}$ $b_{3,2}$ $b_{3,3}$
S

Figure2: the Sub-Bytes step, one of four stages in a round of AES

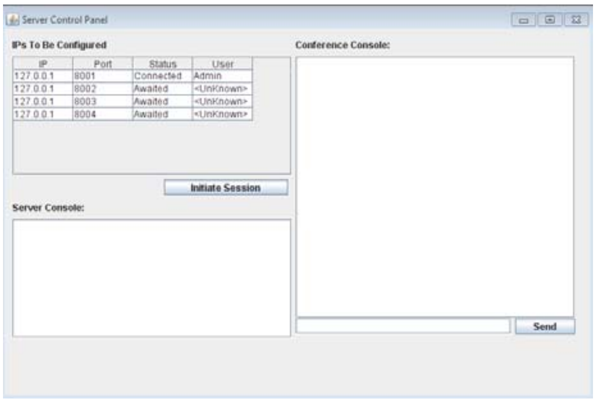
4. DATA ANALYSIS

Collection of ip's:



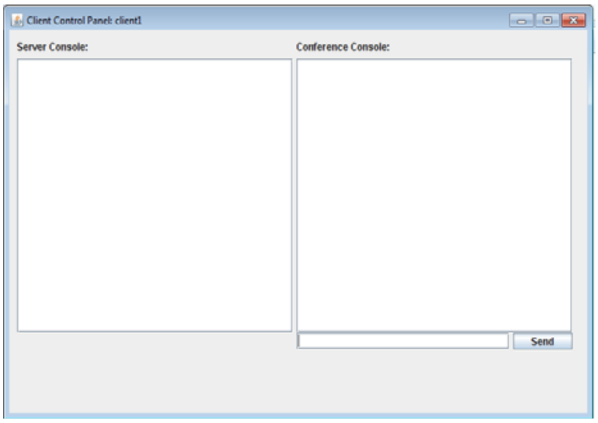
IP	Port	Status	Adm
127.0.0.1	8001	Connected	Adm
127.0.0.1	8002	Awaited	<Ur
127.0.0.1	8003	Awaited	<Ur
127.0.0.1	8004	Awaited	<Ur

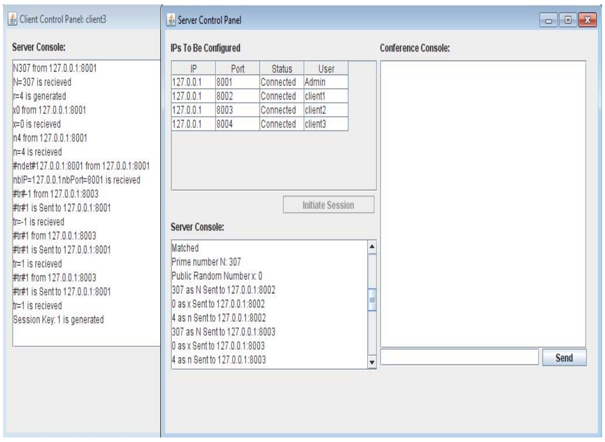
Server control window:



IP	Port	Status	User
127.0.0.1	8001	Connected	Admin
127.0.0.1	8002	Awaited	<UnKnown>
127.0.0.1	8003	Awaited	<UnKnown>
127.0.0.1	8004	Awaited	<UnKnown>

After client initialization:



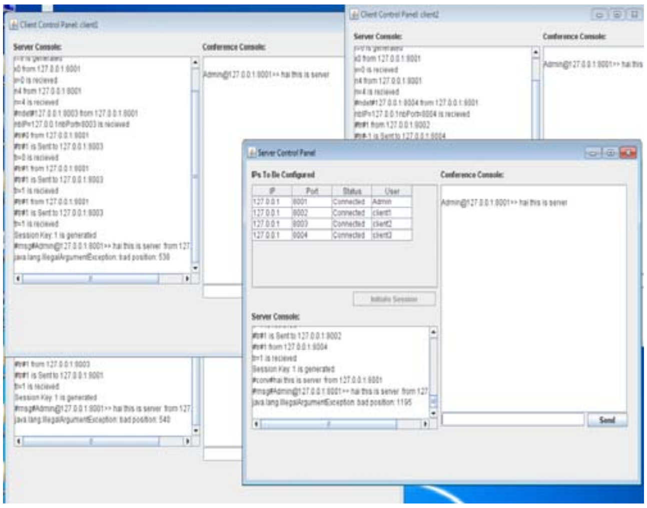


```
N307 from 127.0.0.1:8001
N=307 is recieved
r=4 is generated
x0 from 127.0.0.1:8001
x=0 is recieved
n4 from 127.0.0.1:8001
n=4 is recieved
#ndet#127.0.0.1:8001 from 127.0.0.1:8001
nbIP=127.0.0.1nbPort=8001 is recieved
#tr#-1 from 127.0.0.1:8003
#tr#1 is Sent to 127.0.0.1:8001
tr=-1 is recieved
#tr#1 from 127.0.0.1:8003
#tr#1 is Sent to 127.0.0.1:8001
tr=1 is recieved
#tr#1 from 127.0.0.1:8003
#tr#1 is Sent to 127.0.0.1:8001
tr=1 is recieved
Session Key: 1 is generated
```

IP	Port	Status	User
127.0.0.1	8001	Connected	Admin
127.0.0.1	8002	Connected	client1
127.0.0.1	8003	Connected	client2
127.0.0.1	8004	Connected	client3

```
Matched
Prime number N: 307
Public Random Number x: 0
307 as N Sent to 127.0.0.1:8002
0 as x Sent to 127.0.0.1:8002
4 as n Sent to 127.0.0.1:8002
307 as N Sent to 127.0.0.1:8003
0 as x Sent to 127.0.0.1:8003
4 as n Sent to 127.0.0.1:8003
```

Key exchange and data communication between server and clients:



```
x0 from 127.0.0.1:8001
x=0 is recieved
n4 from 127.0.0.1:8001
n=4 is recieved
#ndet#127.0.0.1:8003 from 127.0.0.1:8001
nbIP=127.0.0.1nbPort=8003 is recieved
#tr#0 from 127.0.0.1:8001
#tr#1 is Sent to 127.0.0.1:8003
tr=0 is recieved
#tr#1 from 127.0.0.1:8001
#tr#1 is Sent to 127.0.0.1:8003
tr=1 is recieved
#tr#1 from 127.0.0.1:8001
#tr#1 is Sent to 127.0.0.1:8003
tr=1 is recieved
Session Key: 1 is generated
#msg#Admin@127.0.0.1:8001>> hai this is server from 127
java.lang.IllegalArgumentException: bad position: 538
```

```
Admin@127.0.0.1:8001>> hai this is server
```

```
x0 from 127.0.0.1:8001
x=0 is recieved
n4 from 127.0.0.1:8001
n=4 is recieved
#ndet#127.0.0.1:8004 from 127.0.0.1:8001
nbIP=127.0.0.1nbPort=8004 is recieved
#tr#1 from 127.0.0.1:8002
#tr#-1 is Sent to 127.0.0.1:8004
```

```
Admin@127.0.0.1:8001>> hai this
```

IP	Port	Status	User
127.0.0.1	8001	Connected	Admin
127.0.0.1	8002	Connected	client1
127.0.0.1	8003	Connected	client2
127.0.0.1	8004	Connected	client3

```
#tr#1 is Sent to 127.0.0.1:8002
#tr#1 from 127.0.0.1:8004
tr=1 is recieved
Session Key: 1 is generated
#conv#hai this is server from 127.0.0.1:8001
#msg#Admin@127.0.0.1:8001>> hai this is server from 127
java.lang.IllegalArgumentException: bad position: 1195
```

```
Admin@127.0.0.1:8001>> hai this is server
```

```
#tr#1 from 127.0.0.1:8003
#tr#1 is Sent to 127.0.0.1:8001
tr=1 is recieved
Session Key: 1 is generated
#msg#Admin@127.0.0.1:8001>> hai this is server from 127
java.lang.IllegalArgumentException: bad position: 540
```

5. CONCLUSION

In this paper we have concentrated on the field of multicast key exchange, which is a attractive sub-field of cryptography. We deeply analyze the multicast key management schemes proposed in [2], [3], and consequently figure out the fatal limitations. That is, due to the authors' inappropriate assumptions, the three algorithms are not practical at all. However, enlightened by those literatures, we propose another algorithm based on the extended Chebyshev polynomial to achieve multicast key exchange. Correctness and security analysis indicate that this new algorithm is reasonable and practical. The efficiency of the algorithm is also analyzed, while in others' articles ,the efficiency of the algorithms which based on Chebyshev map is not answered. As far as we know, such a multicast key exchange algorithm has not been reported, and the results given here are novel.

REFERENCES

[1] L. Kocarev and Z. Tasev, "Public-key encryption based on chebyshev maps," in *Proceedings of the 2003 International Symposium on Circuits and Systems*, vol. 3, pp. 28–31, 2003

[2] K. Qin, "Research on ip multicast rekey algorithm," Master's thesis (InChinese), University of Electronic Science and Technology of China,Chengdu, 2006.

[3] K. Qin, M. Zhou, and N. Liu, "A novel group key management based on jacobian elliptic chebyshev rational map," *Lecture Notes in Computer Science*, vol. 4672, pp. 287–295, 2007.

[4] P. Bergamo, P. D'Arco, A. de Santis, and L. Kocarev, "Security of publickey cryptosystems based on chebyshev polynomials," *IEEE Transaction on Circuits and Systems-I*, vol. 52, no. 7, pp. 1382–1393, 2005.

[5] T. Hardjono, L. R. Dondeti, and R. Perlman, *Multicast and Group Security*. Norwood, MA, USA: Artech House, Inc., 2003.

[6] M. Burmester and Y. Desmedt, "A secure and efficient conference key distribution system," *Lecture Notes in Computer Science*, vol. 950, pp.275–286, 1995. [7] M. Steiner, G. Tsudik and M. Waidner. "Diffie-Hellman key distribution extended to group communication," in *Proceedings of the 3rd ACM conference on Computer and communications security, New Delhi, India*. pp. 31-37, 1996

[8] G. J. Fee and M. B. Monagan, "Cryptography using chebyshev polynomial," in *2004 Maple Summer Workshop. Available at: http://oldweb.cecm.sfu.ca/CAG/papers/Cheb.pdf*, Burnaby, Canada, 2004.

[9] L. Aceto and D. Trigiante, "Periodic solutions of chebyshev polynomials with respect to the discrete variable," *Journal of Difference Equations and Applications*, vol. 8, no. 2, pp. 195–199, 2002.

[10] M. Hunziker, A. Machiavelo, and J. Park, "Chebyshev polynomials over finite field and reversibility of _ -automata on square grids," *Theoretical Computer Science*, no. 320, pp. 465–483, 2004.

[11] M. Hunziker, A. Machiavelo and J. Park, "Chebyshev Polynomials over Finite Field and Reversibility of automata on Square Grids," *Theoretical Computer Science*, vol. 320, pp. 465-483, 2004.

[12] D.Wang and X.Wei, "A New Key Exchange Scheme Based on Extended Chebyshev Polynomials", *The 4th WSEAS International Conference on Applied Mathematics and Computer Science, Rio de Janeiro, Brazil*,pp.1-5