Proceedings of the 5th WSEAS International Conference on Applied Computer Science, Hangzhou, China, April 16-18, 2006 (pp970-974)

Multi-Seed Key Distribution Scheme With Test System

XIE YUMIN, SHI FENG , MUHAMMAD KAMRAN
Dept. of Computer Science & Engineering
Beijing Institute of Technology
Beijing , 100081
China

Abstract: - The key problem of securing multicast is to generate, distribute and update Session Encryption Key(SEK). After analyzing PE scheme and its shortage, a group key distribution scheme utilizing a polynomial expansion with multi-seed(M-PE)is proposed. Its operation is demonstrated by using multi-seed, the group member is partitioned to many subgroups. While updating the SEK, computation is needed only in one of subgroups, the other of them will use the computation history to update their SEK. A prototype test system is designed in order to validate our new scheme especially for its computable complexity. In the system, a kind of technology of producing a strict prime number is introduced as well as a table is designed in order to reduce the computation complexity.

Key-Words: - M-PE; Multi-seed; Computable test; Strict prime number;SEK

1 Introduction

Multicast has a far-ranging application area such as network conference, online games, and cooperative work. The key management for multicast is a core problem which includes how to generate, distribute and update Session Encryption Key(SEK) when a group member has some change while the session is running. SEK is known by all members and it is used in operations like encryption/decryption, authentication, which can satisfy the requirement of secrecy, membership authentication and integrality[1]. Comparison to the single broadcasting, multicast has many unique features including forward confidentiality, backward confidentiality and collusion decryption.

Protection against unwanted access, computation complexity, communication costs and storage requirement must be considered when we solve the aforementioned problem. After analyzing a polynomial expansion(PE) scheme and its shortage, this paper proposes a new scheme called polynomial expansion with multi-seed(M-PE). Details of scheme and its four considering pre-conditions be discussed by comparing with related schemes

2 Key distribution scheme based on PE and its shortage

Now a brief review of scheme PE is given with details expressed in[2]. When a group member leaves a running session, the session must renew its SEK in order to satisfy the request of forward confidentiality. Thus GC(group controller)constructs polynomial

$$P(x) = SEK_{new} + \prod_{i=1}^{N-1} [x - h(KEK_i, \mu)] \quad (1)$$

we obtain (2) after expanding (1) and eventually found(3)

$$P(x) = x^{N-1} + a_{N-2}x^{N-2} + a_{N-3}x^{N-3} + \ldots + a_1 x + a_0 \quad (2)$$

$$P(h(KEK_i, \mu) = SEK_{new} \quad (3)$$

So after GC broadcasting the following message

$$\{\mu, a_{N-2}, a_{N-3}, \ldots, a_1, a_0\} \quad (4)$$

The remainder of group members can get new SEK through expression (3). PE doesn't include the procedure of encryption/decryption that usual method does.

In multicast message (4), we use following expression to compute a_0

$$a_0 = SEK_{new} + \prod_{i=1}^{N-1} h(KEK_i, \mu) \quad (5)$$

where μ is random seed and KEK is key encryption key.

There are two shortcomings of PE. One is that when computing message (4) the following detailed expression can be obtained from expression (1)

$$P(x) = x^N + (h_1 + h_2 + \ldots + h_n)x^{N-1} + (h_1h_2 + h_1h_3 + \ldots + h_{n-1}h_n)x^{N-2} + \ldots + h_1h_2..h_n + SEK \quad (6)$$

So the following numbers of multiplication must be computed in order to construct message (4)

$$C_n^n(n-1) + C_n^{n-1}(n-2) + \ldots + C_n^2 \quad (7)$$

This is a NP complete problem and when n is greater than 30, PE is unable to provide useful results in practice. According to above analysis we can conclude that the conclusion of [2] suggests that at the permission of communication complexity PE can support at most 900 group members is not correct.The other shortcoming of PE is that it requires a prime number which is less than and adjacent to 2^B-1. This is a hard problem when B is a high digit number such as B=128 binary bits.

PE-LKH is an improvement scheme of PE which reduces communication complexity through combination with LKH. But it needs renewing key on balance bintree and this needs a complex storage. When renewing SEK on the tree it must update a lot of KEK at the same time, this is a tanglesome procedure because when a KEK is updated there must a communication be taken place between GC and a group member.

The scheme of this paper adopts grouping method and group member is partitioned to many subgroups. Every subgroup holds a random seed and thus when there is a member change in the group, only subgroup which includes the changing member has to update its subgroup seed, the others need not to be updated. Within acceptable computation and communication complexity this scheme can support about 21000 members. Lt is suitable for most practical application. In order to validate the computation feasibility a test system is designed.

The section 3 of paper gives description of the scheme, while section 4 we discuss about the scheme. Description of test system is elaborated in section 5 with its conclusion. At the end we summarize the whole paper.

3 Key distribution scheme of M-PE

3.1 Generating of KEK

Generate a kek for every member is obligation for GC. Our test system uses a reformative BLUM scheme to create a random number which is used to transmit kek. The reformative point is, in our procedure factorization of BLUM number isn't published anymore, so a BLUM number can be used by many members for numberless times. This reformation is based on trust of member to GC. Details of BLUM scheme can be found in[3]. The generating procedure steps of kek be listed as follows.

1).GC consult with the member to generate a random R through reformative BLUM scheme.

2).GC takes a random number which be described in the test system as member kek and than GC computes $h(kek, k_u) \oplus KEK$,R $\oplus kek$, transmits them to member.(k_u is member password, h is one way function and it's definition is given in part 4, KEK is subgroup kek)

3)R and k_u are known to member, so the member can get KEK and kek .

3.2 Key distribution using M-PE

We consider Fig.1 as the system logistic topological structure.

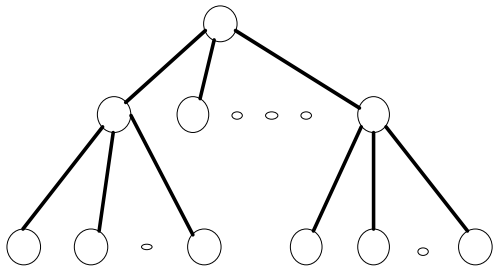

Fig.1 Logic tree structure for M-PE

In this figure, all leaf nodes those have a common parent share a random seed μ . Through the SEK renewal deal procedure we can find multi-seed is useful for performance improvement. The reason is that when renewing SEK we can keep most members to update their SEK through history computation instead iterative computation must be operated each time. This keeps the computation less complex and at the same time much reduction in system communication is obtained.

Non-leaf node of the tree storage message can be expressed as follows

$$\{ (KEK_{11}, \mu_{11}, ch_{11}),$$
$$(KEK_{12}, \mu_{12}, ch_{12}), \ldots$$
$$(KEK_{1m}, \mu_{1m}, ch_{1m}),$$
$$(KEK_{111}, KEK_{112}, ...KEK_{11m}),$$
$$(KEK_{121}, KEK_{122}, ...KEK_{12m}), \ldots$$
$$(KEK_{1m1}, KEK_{1m2}, ...KEK_{1mm})\}$$

We use $\{M_1, M_2, ..., M_m, E_1, E_2, ...E_m\}$ to express it in a simplified form.

In above set ch is computation history and its meanings will be explained later on.

Suppose a KEK_{1hl} (Member) leaves system:

GC: $\forall(M_i)$

If $i \neq h$ than compute

$$a_{1i}^{new} = [SEK_{new} + ch_{1i}] \oplus h(KEK_{1i}, \mu_{1i}) \qquad (8)$$

If $i = h$ than select a new random seed μ_{1i} and the left members re-compute (2)

expression, specially a_{1i}^{new} is as (9),

$$a_{1i}^{new} = [SEK_{new} + \prod_{j=1}^{s} h(KEK_{1ij}, \mu_{1i})] \oplus h(KEK_{1i}, \mu_{1i}) \quad (9)$$

update $ch_{1i} = \prod_{j=1}^{N_i} h(KEK_{1ij}, \mu_{1i})$, Compose message (8) to multicast

$$\{a_{11}^{new}, a_{12}^{new}, ...a_{1v}^{new}, ..a_{1n}^{new}, u_{1i}, a_{1i}^{s-1}, a_{1i}^{s-2}, ...a_{1i}^{1}, a_{1i}^{new}\}(v \neq i) \quad (10)$$

In (10) n=subgroup number-1. s is the number which is hold by h group.

U: $\forall(KEK_{1yz})$

If $y \neq h$ than U get a_{1y}^{new} from the message according to (8), U has known $ch_{1y}, KEK_{1y}, \mu_{1y}$, so SEK_{new} can be computed out.

If $y = h$ than U updates μ_{1y}, through (9) to obtain a_{1y}^{0}, according to (3) compute SEK_{new}, and updates $ch_{1y} = a_{1y}^{0} - SEK_{new}$.

Suppose a member is added to system:

GC: Performing the procedure as in section 3.1, GC and member exchanges kek and the subgroup which member will join is decided. Similarly with member leaving procedure GC generates and multicasts message. If a subgroup member arrives lts capacity, a new logistic subgroup be created in order to accept new members.

U: If a subgroup has no change, its member will get new SEK through (8). Otherwise it gets new SEK from (9) and (3), computes computation history through (5) and stores it.

4 Security and efficiency of M-PE

Similar to PE, in M-PE a kind of function called one way function h is used frequently. This function is defined as z=h(x,y) In [4] considering z and y as known parameters, it is difficult to compute x. M-PE is based on PE and thus its security is decided by this kind of one way function. More details is founded in [2].

Theorem 1: If PE holds the property of satisfying forward confidentiality, backward confidentiality and collusion decryption, M-PE holds the same property comparably.

Proof: When there is a member change in a subgroup, new SEK will be transmitted through a new seed updates. According to PE the security property is realized. For most subgroups new SEK will be got through computation history(CH). Following expressions be used for this procedure.

$$a_{1i}^{new} = [SEK_{new} + \prod_{j=1}^{s} h(KEK_{1ij}, \mu_{1i})] \oplus h(KEK_{1i}, \mu_{1i}) \quad (11)$$

$$ch_{1i} = \prod_{j=1}^{s} h(KEK_{1ij}, \mu_{1i}) \quad (12)$$

$$a_{1i}^{new} = [SEK_{new} + ch_{1i}] \oplus h(KEK_{1i}, \mu_{1i}) \quad (13)$$

CH is created by GC when there is a member change in a subgroup, and be multicast to all members. From (11) we can see that only the changing-subgroup members (except for removed member)can compute out $SEK_{new} + \prod_{j=1}^{s} h(KEK_{1ij}, \mu_{1i})$, removed member can not get new SEK because of seed changing. So removed member can't get CH . The other subgroup members can only get new SEK and unable to get CH from (13). Thus changing-subgroup through new seed and unchanged subgroups through CH update their SEK safely.

For collusion decryption we can conclude that the removed member can't get new SEK because of seed updating. If it gives some message about the subgroup to other subgroup members, this kind of message only is the subgroup KEK. From (9) we say this message is useless because of seed updating.

Now we give some discussion about complexity of M-PE. We select a three layer logistic structure to represent a M-PE system structure. Suppose the child node number is limited to 25 for any middle node of the structure. The order of those nodes is from up to bottom and from left to right. The capacity of system is 21900 nodes.

Comparison of communication cost of M-PE with other related scheme is given in table 1. Among these schemes kek, seed and SEK are all 128 bits binary digits. The message with IP format is 1500 bytes long. The communication costs of SKDC, LHK, PE are N-1, (d-1) $\log_d N$, $\left\lceil \frac{N}{90} \right\rceil$ [2] respectively. According to PE, the communication cost of M-PE is $\left\lceil \frac{N_S + S}{90} \right\rceil$, where S is the middle node number of the logic structure. In our example $N_S = 25$.

In scheme LKH balance tree is used in order to get a low communication cost. If we adopt balanced bintree then the communication cost will be reduced to table 2. But it will increase storage cost and PE-LKH adopts this idea.

Tab.1 Communication costs Comparison

n	*100*	500	*1000*	*5000*	*9000*	*10000*
SKDC	99	499	999	4999	8999	9999
LKH	99	134	149	184	196	198
PE	2	6	12	56	100	112
M-PE	1	1	1	3	5	5

Tab.2 Communication costs of LKH with balance bintree

n	*100*	*500*	*1000*	*5000*	*9000*	*10000*
LKH	7	9	10	13	14	14

If we take 25 members as a group under PE scheme and take its computation complexity as 1, than the computation complexity of M-PE is 1 and it is a constant, no matter how many members are there in the system.

Adopting three layer balanced tree with d degrees of each node under M-PE, GC need to storage $N+3\times N/d$ related messages which include a kek for a member, a seed, a CH, and a kek for a subgroup. Its storage complexity is $O(N)$.

5 Computable test and conclusion

According to [2], 900 is the largest number of system members under feasible communication cost when SEK is a 128 bit long digital number. Under this precondition, we will get a (900-25)*25=21875 members that system can support under M-PE in which the subgroup size is 25. With this changing seed subgroup there are total 21900 members that system of M-PE can support.

Large prime number generation. At present, generate a long bit prime number is a difficult problem in mathematics. Many encryption schemes adopt an approximate prime number or a probability prime number. In our system using prime number is to reduce computation complexity such as the expression (xy)%m=((x%m)(y%m))%m. If a non-prime number m is used, and at the same time x and y just satisfy xy=m, than there will be an error in the system. Our method is described below.

1).Select a random odd number r which is closed to 2^B-1.

2).Test that r has no factor in [2,2000], if it has then jump to 1).[5]

3).Select a million random numbers in (2000,r), test if r can be divided exactly by each of them. If one of them is a factor of r then jump to 1).

4).Continuous run 3) for 100000 times and there is no jump to 1) then jump to 5), else jump to 1).

5).Record r for further use.

It is not assuredly to say that the number which is got from above procedure is a prime number. But it isn't important whether r is a prime or not presently. Because r has passed the above test procedure, then if we run 3) of the procedure in any single later on, we will get a million random numbers in (2000,r) for each of them will be at almost 100% probability that it is not a factor of r. If there are some numbers which can not satisfy this property, we would run 3) once again to get another numbers in order to get a million numbers. In a session period a million numbers is enough for system to use. If we take these million numbers and r as a universal set, then r is a strict prime number on the set.

Complexity Reduction Methodology. (1)Use formula

(xy%m=((x%m)(y%m))%m and

(x+y)%m=((x%m)+(y%m))%m

adequately and this can avoids multiplication in the system. (2)Use

x%y=(x%(y<<m))>>m

to simplify modular arithmetic. The following is an example for demonstration.

Because when we use

(xy) %m = ((x%m)(y%m))%m

there may be a 2*B long bit number mod B long bit number operation. In order to simplify this kind of operation, we generate a B size long table. The table records the result of number $2^B, 2^{B+1},\ldots, 2^{2B}$ mod r respectively. Thus when we compute (b1*b2*…bn)%r(bi is a B bit long number), use the following procedure.

1).For(i=1 to n-1)

For bi and bi+1, take every bit of bi which is noted down as 2^x and multiply by every bit of bi+1 which is noted down as 2^y, than 2^{x+y} will be the result of each pair of them. So there will get at most a 512 numbers after we merge the same number result. Among them we get the result great than 2^B-1 and replace it with its corresponding result in table that we have created initially. And then we add up these 512 numbers as p, if p great than 2^B-1 then we take every high bit of p which is greater than 2^B-1, find its corresponding result in table and add the result to

the remainder low bit part of p. Repeat this step until p is less than 2^B and than store p to bi+1.

2).Compute bn%r and this result is the end result of (b1*b2*…bn)%r.

The above procedure is efficient in generating prime. According to the law of testing prime number, when a number has no factor in [2,2000], then the probability of it is a prime is 80%[5]. Although a running span is at least 7 hours to finish a test, but our tests show that if a number through 3 minutes test, then the number will be get across the whole test span.

If a number passed the test, there is no need to establish a million size of table because this makes storage cost very high. Instead we only need to generate a random number x and then test $h(x,\mu)$ and r is prime to each other. If it is not, the only thing that we do is to repeat this step until we get it.

Through test we can conclude that although we get a big prime number will take a long time, but as long as we get it, then it can be used as a constant in the future.

6 Conclusion

With the internet widespread application, multicast became a more and more important research domain. In this paper we propose a key distribution scheme called M-PE. The core idea of M-PE is that, through computation history concept when renew SEK the computation can be controlled on a local part. Thus the un-computable problem of PE is conquered with communication and storage cost keep in a low level. It is an important feature that our scheme is checked by our test system, thus it has a stronger persuasion than some other schemes described in [2]and [4].

Until now there are many key distribution schemes have been proposed or have been used in practical systems. Such as Clique[6], GKMP[7], Iolus[8], and so on. Comparing with PE or M-PE, these schemes must include encryption and decryption process with in their scheme description. PE and M-PE show us that this kind of process can be canceled. Moreover M-PE is a useful one compare with PE according to its computability.

References:

[1] XU Mingwei, DONG Xiaohu, XU Luo. A Survey of research on key management for multicast. *Journal of Software*, Vol.15, No.1, 2004, pp.141-150.(in Chinese)

[2] ZHU Wentao, XIONG Jiping, LI Jinsheng, et al. A Study of the key distribution in secure multicast. *Journal of Software*, Vol.14, No.12, 2003 pp. 2052-2059.(in Chinese)

[3] MAO W. *Modern cryptography : theory and practice*. Beijing. Publishing house of electronics industry. 2004.

[4] TRAPPE W, SONG J, POOVENDRAN R, et al. Key distribution for secure multimedia multicasts via data embedding. *In: Proceedings of the Acoustics, Speech, and Signal Processing*. Salt Lake City, 2001. pp.1449-1452.

[5]WANG Ying. A fast way to find a big prime number in RSA scheme. *Journal of human university of science and engineering.* Vol.26, No5, 2005, pp.14-16 (in Chinese).

[6]Setiner M, Taudik G, Waidnet M. Cliques: A new approach to group key agreement. *Technical Report ,RZ, 2984, IBM research.*. 1997

[7]Harney H, Muckenhirn C. Group key management protocol(GKMP) specification. RFC2093. 1997.

[8]Mittra S. Iolus: A framework for scalable secure multicasting. *In: ACM SIGCOMM Computer Communication Review.* Vol.27, No.4,1997, pp.277-288.