International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

CONTRIBUTORY BROADCAST ENCRYPTION PLAN WITH SHORT CIPHER TEXTS

KV RAGHAVENDER[1], CH.JYOTHI[2], B.KALYANI DEVI[3], ANABATHULA UMESH KUMAR[4], B SAI TEJA[5] BONABOYANA MANASWINI[6]

[1] Associate Professor, Department of CSE, Malla Reddy Engineering College (Autonomous), Hyderabad, India

[2] Assistant Professor, Department of CSE, Malla Reddy Engineering College (Autonomous), Hyderabad, India

[3] Assistant Professor, Department of CSE, Malla Reddy Engineering College (Autonomous), Hyderabad, India

[4,5,6] UG Student, Department of CSE, Malla Reddy Engineering College (Autonomous), Hyderabad, India

Abstract: Customary telecast encryption (TE) plans permit a sender to safely show to any subset of individuals yet require a trusted gathering to disseminate unscrambling keys. Bunch key understanding (BKU) conventions empower a gathering of individuals to arrange a typical encryption key by means of open systems so that lone the gathering individuals can decode the ciphertexts encoded under the common encryption key, yet a sender can't reject a specific part from unscrambling the ciphertexts. In this paper, we connect these two thoughts with a half and half primitive alluded to as contributory show encryption (ConBE). In this new primitive, a gathering of individuals arrange a typical open encryption key while every part holds an unscrambling key. A sender seeing people in general gathering encryption key can confine the unscrambling to a subset of individuals from his decision. Tailing this model, we propose a ConBE plan with short ciphertexts. The plan is ended up being completely plot safe under the choice n-Bilinear Diffie-Hellman Exponentiation (BDHE) supposition in the standard model. Of autonomous interest, we introduce another BE plan that is aggregately. The aggregatability property is appeared to be valuable to build propelled conventions.

Keywords: Broadcast Encryption, Group Key Agreement, Contributory Broadcast Encryption, Provable Security.

I. INTRODUCTION

With the increase in technology advancement in communication technologies, there is an increasing demand of versatile cryptographic primitives to protect group communications and computation platforms. These new platforms include instant-messaging tools, collaborative computing, mobile ad hoc networks and social networks. These new applications call for cryptographic primitives allowing asunder to securely encrypting to any subset of the users of the services without relying on a fully trusted dealer. Broadcast encryption (BE) is a well-studied primitive intended for secure group-

oriented communications. It allows a sender to securely broadcast to any subset of the group members. Nevertheless, a BE system heavily relies on a fully trusted key server who generates secret decryption keys for the members and can read all the communications to any members. Group key agreement (GKA) is another well-understood cryptographic primitive to secure group-oriented communications. A conventional GKA allows a group of members to establish a common secret key via open networks. However, whenever a sender wants to send a message to a group, he must first join the group and run a GKA protocol to share a secret key with the intended members more recently, and to overcome this limitation, with the introduction of asymmetric GKA, in which only a common group public key is negotiated and each group member holds a different decryption key. However, neither conventional symmetric GKA nor the newly introduced asymmetric GKA allow the sender to unilaterally exclude any particular member from reading the plaintext. Hence, it is essential to find more flexible cryptographic primitives allowing dynamic broadcasts without a fully trusted dealer. This paper investigates a close variation of the above mentioned problem of one-round group key agreement protocols and focuses on "how to establish a confidential channel from scratch for multiple parties in one round". We provide a short overview of some new ideas to solve this variation. Asymmetric GKA Observe that a major goal of GKAs for most applications is to establish a confidential broadcast channel among the group. We investigate the potentiality to establish this channel in an asymmetric manner in the sense that the group members merely negotiate a common encryption key (accessible to attackers) but hold respective secret decryption keys. We introduce a new class of GKA protocols which we name asymmetric group key agreements (ASGKAs), in contrast to the conventional GKAs. A trivial solution is for each member to publish a public key and withhold the respective secret key, so that the final ciphertext is built as a concatenation of the underlying individual ones. However, this trivial solution is highly inefficient: the ciphertext increases linearly with the group size; furthermore, the sender has to keep all the public keys of the group members and separately encrypt for each member.We are interested in nontrivial solutions that do not suffer from these limitations. Group key agreement (GKA) is another well-understood cryptographic primitive to secure group-oriented communications. A conventional GKA allows a group of members to establish a common secret key via open networks. However, whenever a sender wants to send a message to a group, he must first join the group and run a GKA protocol to share a secret key with the intended members. More recently introduced asymmetric GKA in which only a common group public key is negotiated and each group member holds a different decryption key. However, neither conventional symmetric GKA nor the newly Introduced asymmetric GKA allow the

sender to unilaterally exclude any particular member from reading the plaintext1. Hence, it is essential to find more flexible cryptographic primitives allowing dynamic broadcasts without a fully trusted dealer.

II. EXISTING AND PROPOSED SYSTEMS

A. Existing System Group key agreement (GKA) is another well-understood cryptographic primitive to secure group-oriented communications. A conventional GKA allows a group of members to establish a common secret key via open networks. However, whenever a sender wants to send a message to a group, he must first join the group and run a GKA protocol to share a secret key with the intended members. More recently, and to overcome this limitation, Wu et al. introduced asymmetric GKA, in which only a common group public key is negotiated and each group member holds a different decryption key. However, neither conventional symmetric GKA nor the newly introduced asymmetric GKA allow the sender to unilaterally exclude any particular member from reading the plaintext. Hence, it is essential to find more flexible cryptographic primitives allowing dynamic broadcasts without a fully trusted dealer.

Disadvantages of Existing System:

- Need a fully trusted third party to set up the system.
- Existing GKA protocols cannot handle sender/ member changes efficiently.

B. Proposed System

We present the Contributory Broadcast Encryption (ConBE) primitive, which is a hybrid of GKA and BE. This full paper provides complete security proofs, illustrates the necessity of the aggregatability of the underlying BE building block and shows the practicality of our ConBE scheme with experiments. First, we model the ConBE primitive and formalize its security definitions. ConBE incorporates the underlying ideas of GKA and BE. A group of members interact via open networks to negotiate a public encryption key while each member holds a different secret decryption key. Using the public encryption key, anyone can encrypt any message to any subset of the group members and only the intended receivers can decrypt. We formalize collusion resistance by defining an attacker who can fully control all the members outside the intended receivers but cannot extract useful information from the ciphertext. Second, we present the notion of aggregately broadcast encryption (AggBE). Coarsely speaking, a BE scheme is aggregately if its secure instances can be aggregated into a new secure instance of the BE scheme. Specifically, only the aggregated decryption keys of the same user are valid decryption keys corresponding to the aggregated public keys of the underlying BE instances. Finally, we construct an efficient ConBE scheme with our AggBE scheme as a building block. The ConBE construction is proven to be semi-adaptively secure under the decision BDHE assumption in the standard model.

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

Advantages of Proposed System:

 We construct a concrete AggBE scheme tightly proven to be fully collusion-resistant under the decision BDHE assumption.

 The proposed AggBE scheme offers efficient encryption/decryption and short ciphertexts.

 Only one round is required to establish the public group encryption key and set up the ConBE system.

III. SYSTEM ARCHITECTURE

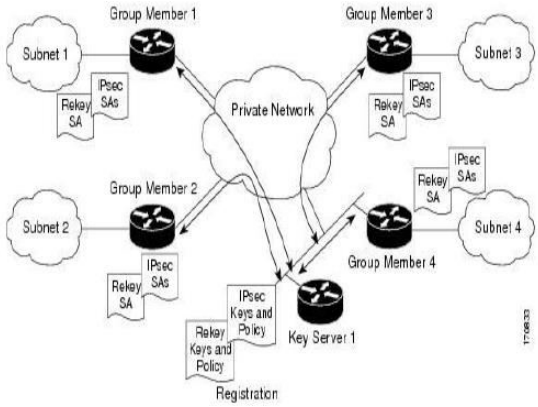

Fig.1. System Architecture. At the high-level, two main methods of this group encryption service are **Encrypt (set, m) c:** where set is a set of participant identifiers to which message m is to be encrypted. This method returns the corresponding ciphertext c **Decrypt (c) (m or error status):** where c is the ciphertext and m is the resulting decryption. If decryption fails, an appropriate error code is returned. Depending on the implementation, ciphertext c may have certain structure, such as include the identity of the sender, the key encapsulation block, the encryption of the message under the encapsulated key, the signature block, etc.

In addition to these two main methods, other methods can be exposed to the application, such as AddUserCertificate and RemoveUserCertificate. It may also be convenient to allow the application to use named groups instead of sets in Encrypt (group, m); if this method is provided it needs to be accompanied with the following group management methods: NewGroup, AddMember, and RemoveMember.

Security Properties:

 Confidentiality: Communicated data is protected from non-members.

 Sender authentication and non-repudiation: Participants can authenticate message senders.

 Membership dynamism: It is possible to form groups and to add/remove participants.

 Perfect Forward Security: Compromise of long term keys of a member does not compromise earlier communication of that member.

 Group Forward and Backward Secrecy: Secrecy of new communication from revoked members, and old communication from new members.

A. Modules Description

 Network Environment Setup Module

 Certificate Authority Module

 Key Broadcast Module

 Group Key management

Network Environment Setup Module: In the first module, we create the network environment setup with nodes, certificate authority as shown in Fig.1. Network environment is set up with nodes connected with all and using socket programming in java.

Certificate Authority Module: In this module, each receiver has a public/secret key pair. The public key is certified by a certificate authority, but the secret key is kept only by the receiver. A remote sender can retrieve the receiver's public key from the certificate authority and validate the authenticity of the public key by checking its certificate, which implies that no direct communication from the receivers to the sender is necessary. Then, the sender can send secret messages to any chosen subset of the receivers.

Key Broadcast Module: In this module formally define the model of group key agreement based broadcast encryption. The definition incorporates the up-to-date definitions of group key agreement and public-key broadcast encryption. Since the core of key management is to securely distribute a session key to the intended receivers, it is sufficient to define the system as a session key encapsulation mechanism. Then, the sender can simultaneously encrypt any message under the session key, and only the intended receivers can decrypt. The new paradigm seems to require a trusted third party as its counterpart in traditional broadcast encryption systems. A closer look shows there is a difference. In a traditional broadcast encryption system, the third party has to be fully trusted, that is, the third party knows the secret keys of all group members and can read any transmission to any subgroup of the members. This kind of fully trusted third party is hard to implement in open networks. In contrast, the third party in our key management model is only partially trusted. In other words, the third party only knows and certifies the public key of each member. This kind of partially trusted third party has been implemented and is known as public key infrastructure (PKI) in open networks.

Group Key Management: The new key management paradigm ostensibly requires a sender to know the keys of the receivers, which may need communications from the receivers to the sender as in traditional group key agreement protocols. However, some subtleties must be pointed out here. In traditional group key agreement protocols, the sender has to simultaneously stay online with the receivers and direct communications from the receivers to the sender are needed. This is difficult for a remote sender. On the contrary, in our key management paradigm, the sender only needs to obtain the receivers' public keys from a third party, and no direct communication from the receivers to the sender is required, which is implementable with exactly the existing PKIs in open networks. Hence, this is feasible for a remote sender. In our scheme, it is almost free of cost for a sender to exclude a group member by deleting the public key of the member from the public key chain or, similarly, to enroll a user as a new member by inserting that user's public key into the proper position of the public key chain of the receivers. After the deletion/addition of certain member, a new logical public-key ring naturally forms. Hence, a trivial way to enable this change is to run the protocol independently with the new key ring. If the

sender would like to include a new member, the sender just needs to retrieve the public key of this user and insert it into the public key chain of the current receiver set. By repeatedly invoking the member addition operation, a sender can merge two receiver sets into a single group. Similarly, by repeatedly invoking the member deletion operation, a sender can partition one receiver set into two groups. Both merging and partitioning can be done efficiently. In this module shows the deletion of member from the receiver group. Then, the sender and the remaining receivers need to apply this change to their subsequent encryption and decryption procedures.

IV. PERFORMANCE ANALYSIS

A. Theoretical Analysis

We first examine the online complexity that is critical for the practicality of a ConBE scheme. When evaluating the performance, we use the widely adopted metrics for regular BE schemes. In these metrics, the costs of simple operations (e.g., read the indices of receivers and perform some simple quantification of group elements associated to these indices) and communication (e.g., the binary representation of the receivers' set) are not taken into consideration. After the CBSetup procedure, a sender needs to retrieve and store the group public key PK consisting of n elements in G and n elements in GT. Moreover, for encryption, the sender needs only two exponentiations and the ciphertext merely contains two elements in G. This is about n times more efficient than the trivial solution. At the receiver's side, in addition to the description of the bilinear pair which may be shared by many other security applications, a receiver needs to store n elements in G for decryption. For decryption, a receiver needs to compute two single-base bilinear pairings (or one double base bilinear pairing). The online costs on the sides of both the sender and the receivers are really low. We next discuss the complexity of the CBSetup procedure to set up a ConBE system. The overhead incurred by this procedure is O (n2). This procedure needs to be run only once and this can be done offline before the online transmission of secret session keys. For instance, in the social networks example, a number of friends exchange their CBSetup transcripts and establish a ConBE system to secure their subsequent sharing of private picture/videos. Since ConBE allows revoking members, the members do not need to reassemble for a new run of the CBSetup procedure until some new friends join. From our personal experience, the group lifetime usually lasts from weeks to months. These observations imply that our protocol is practical in the real world. Furthermore, if the initial group is too large, an efficient trade-off can be employed to balance the online and offline costs. Suppose that n is a cube, i.e., $n = n^3_1$, and the initial group has n members. We divide the full group into n^2_1 subgroups, each of which has n_1 members. By applying our basic ConBE to each subgroup, we obtain a ConBE scheme with O (n^2_1)-size transcripts per member during the offline stage of group key establishment; a sender needs to

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

do O (n^2_1) encryption operations of the basic ConBE scheme, which produces O (n^2_1)-size ciphertexts. Consequently, we obtain a semi-adaptive ConBE scheme with O ($n^{2/3}$) complexity. This is comparable to up-to-date public-key BE systems whose complexity is O ($n^{1/2}$).

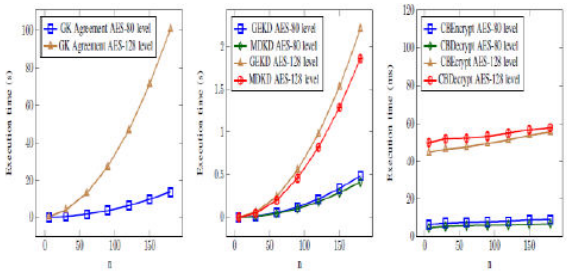

Fig.2. Execution time of Group Key Agreement, Group Encryption Key Derivation, Member Decryption Key Derivation, CB Encrypt, and CBDecrypt for AES-80 and AES-128 levels. B. Experimental Analysis

In this section we present experimental results on our ConBE scheme. The experiments were run on a PC with Intel Core i7-2600 CPU at 3.4GHz, using the C programming language. The cryptographic operations were implemented using the Pairing-Based Cryptography library2. Following the NIST-2012 key size recommendation3, we realized our protocol for a moderate AES-80 level and a more usual AES-128 level, corresponding to the security level of an ideal symmetric cipher with 80-bit and 128-bit secret keys, respectively. We used Type A pairings constructed on the curve y2 = x3 + x with embedding degree 2. Accordingly, in the first case for AES-80 level, G has 512-bit elements of a 160-bit prime order and GT has 1024-bit/128-byte elements; and in the second case for AES-128 level, G has 1536-bit elements of a 256-bit prime order and GT has 3072-bit/386-byte elements, respectively. We performed experiments on the offline procedures including Group Key Agreement, Group Encryption Key Derivation and Member Decryption Key Derivation, and the online procedures including CBEncrypt and CBDecrypt for different group sizes n = 6, 30, 60, 90, 120, 150, 180. The values for CBEncrypt and CBDecrypt consider the worst case, i.e., |S| = 1. Also, we did not optimize the underlying pairing-related parameters or operations, e.g., by choosing a large prime characteristic of the base field and the prime order p with most bits 0 (or 1), and by accelerating multi-base exponentiations/multi-base pairings. Hence, the practical performance of our protocol can be better than the illustrated experimental results. In Fig.2, the security level of our protocol is measured by the secret key size of AES (assumed to be an ideal symmetric cipher), i.e., AES with a truncated 80-bit key and AES with a standard 128-bit key. The leftmost graph in the figure illustrates the group key agreement time for different group sizes and different security levels. The execution time grows almost quadratically with the group size, and also grows with the security level. This is consistent with our theoretical analysis, because the pairings and the exponentiations dominate the computation costs. To achieve a moderate 128-bit

security, the execution time is about 3 minutes for a group of 180 users. This is realistic as the GKA procedure only needs to be run once and then one can broadcast to any subset of the users, without re-running the protocol or any extra revocation sub protocol. The central graph in Fig.2 shows the time to extract the group encryption key and the decryption key for different group sizes and different security levels. Similarly to the group key agreement time, the key extraction time also grows with the security level and the group size. However, even in the worst case, only about 3 seconds are required, which is affordable in practice.

The rightmost graph in Fig.2 illustrates the online session key encryption/decryption time. It can be seen that the time is almost constant for different group sizes, which is consistent with the theoretical analysis. Both the session key encryption and decryption take less than 10ms for a 80-bit security level, and less than 80ms for a 128-bit security level. After the system is set up, the session key transmission is really efficient, which is user-friendly and definitely makes our ConBE scheme practical.We also performed experiments on cost tradeoff between set-up and online encryption. For n = 180 and AES-128 level, the execution times for Group Key Agreement, Group Encryption Key Derivation, Member Decryption Key Derivation, CBEncrypt and CBDecrypt are 101s, 2.20s, 1.86s, 55.3ms, and 57.6ms, respectively. However, using the trade-off described in the previous section, specifically taking subgroups of 6 users, the times become 410ms, 2.05ms, 1.63ms, 1.33s, and 57.6ms. The set-up efficiency was significantly improved, at the cost of a

1.33s encryption time, to be compared to a 55.3ms encryption time without tradeoff.

V. CONCLUSION

In this paper, we formalized the ConBE primitive. In ConBE, anybody can send mystery messages to any subset of the gathering individuals, and the framework does not require a trusted key server. Neither the change of the sender nor the dynamic decision of the planned beneficiaries requires additional rounds to arrange bunch encryption/ unscrambling keys. Taking after the ConBE model, we instantiated and productive ConBE plan that is secure in the standard model. As a flexible cryptographic primitive, our novel ConBE idea opens another boulevard to set up secure telecast stations and can be relied upon to secure various developing circulated calculation applications.

VI. REFERENCES

[1] Qianhong Wu, Member, IEEE, Bo Qin, Lei Zhang, Member, IEEE, Josep Domingo-Ferrer, Fellow, IEEE Oriol Farr`as, and Jes´us A. Manj´on, “Contributory Broadcast Encryption with Efficient Encryption and Short Ciphertexts”, IEEE Transactions On Computers, Vol. Xxx, No. Xxx, Xxx 2015. [2] A. Fiat and M. Naor, “Broadcast Encryption,” in Proc. Crypto 1993, 1993, vol. LNCS 773, Lecture Notes in Computer Science, pp. 480-491. [3] I. Ingemarsson, D.T. Tang and C.K. Wong, “A Conference Key Distribution System,” IEEE Transactions on Information Theory, vol.

28, no. 5, pp. 714-720, 1982. [4] Q. Wu, Y. Mu, W. Susilo, B. Qin and J. Domingo-Ferrer, "Asymmetric Group Key Agreement," in Proc. Eurocrypt 2009, 2009, vol. LNCS 5479, Lecture Notes in Computer Science, pp. 153-170. [5] http://en.wikipedia.org/wiki/PRISM %28surveillance program%29, 2014. [6] Q. Wu, B. Qin, L. Zhang, J. Domingo-Ferrer and O. Farr`as, "Bridging Broadcast Encryption and Group Key Agreement," in Proc. Asiacrypt 2011, 2011, vol. LNCS 7073, Lecture Notes in Computer Science, pp. 143-160. [7] D. H. Phan, D. Pointcheval and M. Strefler, "Decentralized Dynamic Broadcast Encryption," in Proc. SCN 2012, 2011, vol. LNCS 7485, Lecture Notes in Computer Science, pp. 166-183. [8] M. Steiner, G. Tsudik and M. Waidner, "Key Agreement in Dynamic Peer Groups," IEEE Transactions on Parallel and Distributed Systems, vol. 11, no. 8, pp. 769-780, 2000. [9] A. Sherman and D. McGrew, "Key Establishment in Large Dynamic Groups Using One-way Function Trees," IEEE Transactions on Software Engineering, vol. 29, no. 5, pp. 444-458, 2003. [10] Y. Kim, A. Perrig and G. Tsudik, "Tree-Based Group Key Agreement," ACM Transactions on Information System Security, vol. 7, no. 1, pp. 60-96, 2004. [11] Y. Mao, Y. Sun, M. Wu and K.J.R. Liu, "JET: Dynamic Join-Exit-Tree Amortization and Scheduling for Contributory Key Management," IEEE/ACM Transactions on Networking, vol. 14, no. 5, pp. 1128-1140, 2006.
[12] C. Boyd and J.M. Gonz´alez-Nieto, "Round-Optimal Contributory Conference Key Agreement," in Proc. PKC 2003, 2003, vol. LNCS 2567, Lecture Notes in Computer Science, pp. 161-174.