Volume 2, Issue 12, December 2012 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

An Enhanced Secure Route Formation Using Secure Key Management Mechanism for MANETS (SCRFKM)

R. Naveen Kumar *
Department of Informatics, Kakatiya University.

Bapuji V
Department of Informatics, Kakatiya University.

Dr. A. Govardhan,
Department of CSE, JNTU (H).

Prof. S.S.V.N. Sarma
Department of CSE, Vaagdevi College of Engineering

Abstract— To enhance security in distributed networks, such as ad hoc networks, it is important to evaluate the trustworthiness of participating entities since trust is the major driving force for collaboration especially in route formation from source to destination. Wireless broadcast is an effective approach to find route information among number of nodes. To provide secure access to routing table in wireless route formation, symmetric key-based encryption is used to ensure that only nodes who own the valid keys can decrypt the route information. Regarding various broadcasts, an efficient key management to distribute and change keys is in great demand for access control in route establishment. In this paper, we propose an efficient key management scheme (namely SCRFKM to handle key distribution with regarding to complex route formation possibilities and node activities. SCRFKM is designed to have the following advantages. First, it supports all route request activities in MANETs. Second, in SCRFKM, a node only needs to hold one set of keys for all requested route formation operations, instead of separate sets of keys for each request. Third, SCRFKM identifies the minimum set of keys that must be changed to ensure routing security and minimize the rekey cost. Our simulations show that SCRFKM can save about 25% of communication overhead in the broadcast channel and about 30% of decryption cost for each node, compared with logical key hierarchy based approaches.

Keywords —Wireless broadcast, key management, access control, key hierarchy, key distribution, secure group communication, SCRFKM (Secure Route Formation Key Management)

I. INTRODUCTION

The rapid advent of wireless technology helped in the increasing popularity of mobile devices and there is a tremendous increase in interest of wireless routing mechanisms in both industrial and academic communities in recent years. Among various approaches, broadcasting allows a very efficient usage of the scarce wireless bandwidth, because it allows simultaneous access by an arbitrary number of mobile nodes [1]. Wireless data broadcast services have been available as commercial products for many years. A wireless data broadcast system can thus be assumed to consist of three components as depicted in Figure 1.
(1) The broadcast server; (2) the mobile devices; and (3) the communication mechanism. The server broadcasts route formation requests on air. A mobile node that receives the broadcast information filters the nodes request according to node's queries and privileges. The specialty of the broadcast system is that (a) the routing server determines the schedule to broadcast all route requests on air and (b) the mobile nodes listen to the broadcast channel but only retrieve data (filter data out) based on nodes' queries. The communication mechanism consists of wireless broadcast channels and uplink channels. Broadcast channel is the main mechanism for data dissemination. Route request data is broadcasted at regular intervals of time so that nodes can recover lost or missed route requests. The uplink channels, which have limited bandwidth, are reserved for occasional uses to dynamically route change requests.

In broadcast services, the basic data unit is **data item**, such as a Route Request Packet. Data items are grouped into **requests** and a node specifies which request he would like to transmit or receive. Typical request could be Route Request, Route Reply, Route Error, etc. For simplicity, we assume that each request covers a set of data items, and routes are exclusively formed successfully. A node may request for one or more routes. The set of requests is called the node's **subscription**. Nodes can subscribe via Internet or uplink channels to specify the routes that they are interested in route formation. Earlier studies on wireless route request broadcast services have mainly focused on performance issues such as reducing routing information access latency and conserving battery power of mobile nodes.
Unfortunately, the critical security requirements of this type of broadcast routing service have not yet been addressed, i.e. broadcast service providers need to ensure backward and forward secrecy [2], [3] with respect to membership dynamics. In the wireless broadcast environment, any node can monitor the broadcast channel and record the broadcast data of route formation. If the routing information data is not encrypted, the content is open to the public and anyone can access the routing information data and thus may lead to adversaries' modification of routing information. In addition, a node may only subscribe to a few routes. If routing information data in other routing tables are not encrypted, the node can obtain

data beyond the subscription privilege. Hence, access control should be enforced via encrypting routing data in a proper way so that only subscribing nodes can access the broadcast routing information data, and subscribing nodes can only access the data to which they subscribe.

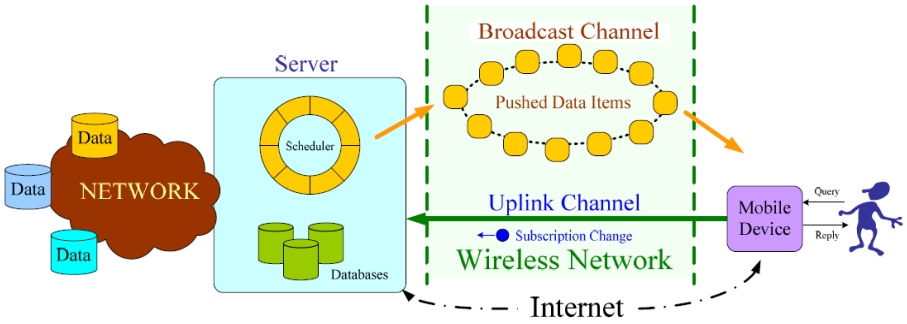


Figure 1: Broadcasting mechanism for route formation form

Symmetric key based encryption is a good choice for ensuring secure data dissemination and access in routing information to be accessed. The broadcast routing data can be encrypted so that only those nodes who own valid keys can decrypt them. Thus, the encryption keys can be used as an effective means for access control in wireless route information broadcast. For example, each node has one unique key to encrypt the route information. The key is issued to the node that is authorized to receive and decrypt the data items. If a node subscribes to multiple route requests, it needs an encryption key for each request. Since a node only has keys for his route request, he cannot decrypt broadcast data and rekey messages designated to other nodes. At the same time, a routing data item can be decrypted by an arbitrary number of nodes who subscribe to it. This allows many nodes to receive the data at the same time and addresses the scalability problem, or request lost or missed keys Two categories of key management schemes in the literature may be applied in broadcast services: (1) logic key hierarchy (LKH) based techniques [2]–[9] proposed for multicast services; and (2) broadcast encryption techniques [10]–[16] in current broadcast services. They normally require nodes to possess decryption boxes to receive the subscribed route information, and the broadcast services can only provide to nodes a few routes, each of which includes a fixed set of routing information. Nodes cannot select individual route within a set of routes. If a node wants to change his route, the node needs to request another decryption box that can decrypt the subscribed route information. This straightforward approach used in LKH is inefficient for nodes subscribing to many routes. If nodes could use the same set of keys for multiple routes, there would be fewer requirements for nodes to handle keys. Furthermore, when a node changes its route request, we argue that it is unnecessary to change keys for the routes to which the node is still using, as long as security can be ensured. In this way, rekey cost can be reduced and fewer nodes will be affected. Therefore, we propose a new key management scheme, namely Secure Route formation Key Management (**SCRFKM**), based on two important observations: (1) nodes who subscribe to multiple routes can be captured by a shared key tree, and (2) old keys can be reused to save rekey cost without compromising security. **SCRFKM** has two components: **shared key tree** and **shared key management**.

II. Related Work

A. Logical Key Hierarchy

Secure key management for wireless broadcast is closely related to secure group key management in networking [4]. Logical key hierarchy (LKH) is proposed in [2], [5] that uses a key tree (depicted in Figure 2) for each group of nodes who subscribe the same route. The root (top node) of the tree is the data encryption key (DEK) of the route. Each leaf (bottom node) in the tree represents an individual key (IDK) of a node that is only shared between the system and the node. Other keys in the tree, namely key distribution keys (KDKs), are used to encrypt new DEKs and KDKs. A node only knows the keys along the path from the leaf of the node to the root of the key tree.

When a node joins or leaves the group, the server needs to change and broadcast the corresponding new keys, and this operation is called *rekey*, and the broadcast message of new keys is called *rekey message*. In our system, data and rekey messages are broadcast in the same broadcast channel to the nodes.

B. Broadcast Encryption

There are some other key management schemes in the literature for multicast and broadcast services for routing establishment.[10] used arbitrarily revealed key sequences to do scalable multicast key management without any overhead on joins/leaves.[11] proposed two schemes that insert an index head into packets for decryption. However, both of them require pre-planned subscription, which contradicts the fact that in pervasive computing and air data access, a node may change subscriptions at any moment. In addition [11] only supports a limited combination of programs.[13] proposed a scheme to yield maximal resilience against arbitrary coalitions of non-privileged nodes. Zero-message scheme [14], [15] does not require the broadcast server to disseminate any message in order to generate a common key. Naor *et al.* [16] proposed a stateless scheme to facilitate group members to obtain updated session keys even if they miss

some previous key distribution messages. Although this scheme is more efficient than LKH in rekey operations, it mainly handles revocation when a node stops subscription. It does not efficiently support joins, which are crucial in our system. Finally, [24], [27] proposed self-healing approaches for group members to recover the session keys by combining information from previous key distribution information. Compared with LKH-based approaches, key management schemes in broadcast route information encryption are less flexible regarding possible subscriptions. Conforming to the current practice described in RFC2627 [2], we select binary trees to present our scheme. Note that our scheme does not require binary trees and can be applied in trees of other degrees.

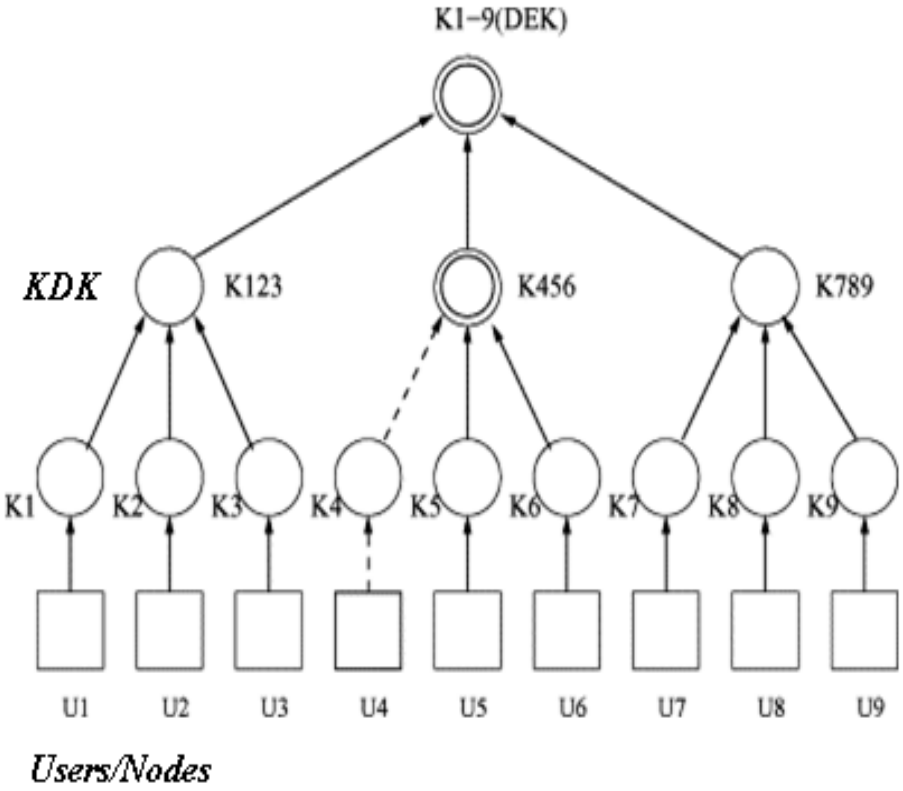


Figure 2: Logical Key Hierarchy

III. Shared Key Structure

a. Rekey Operations

In this study, we consider node activities of joining/leaving/shifting among trees, instead of joining/quitting/changing among route formation. Consider the example in Figure 3, where a node u_s shifts from tr_4 to tr_6.

When u_s was in tr_4, u_s subscribed g_1 and g_2. After he shifts to tr_6, he subscribes g_1, g_2 and g_3. Hence, the shift in fact means the node adds g_3 into his current subscription. To issue new keys upon a node event, the main task is to identify the keys that need to be changed. We use two types of paths in the key forest to represent the to-be-changed keys. When a node leaves a tree, we say, a leave path is formed, which consists of keys that the node will no longer use.

When a node joins a tree, we say, an enroll path is formed, which consists of keys that the node will use in the future. Similarly, when a node shifts from one tree to another, a leave path and an enroll path are formed. In SCRFKM, a complete path starts from the leaf node and ends at the multiple DEKs of the subscribed routes that share the tree. For example, in Figure 4, when us shifts from tr4 to tr6, the leave path consists of kn_L and kr_4, and the enroll path consists of kn_J, kr_6, kg_1, kg_2 and kg_3. Note that in this example, kg_1 and kg_2 are the keys that node already has and still needs in the future. Hence, kg_1 and kg_2 are not in the leave path, although us leaves tr_4. To broadcast new keys, the server should first compose rekey packets.

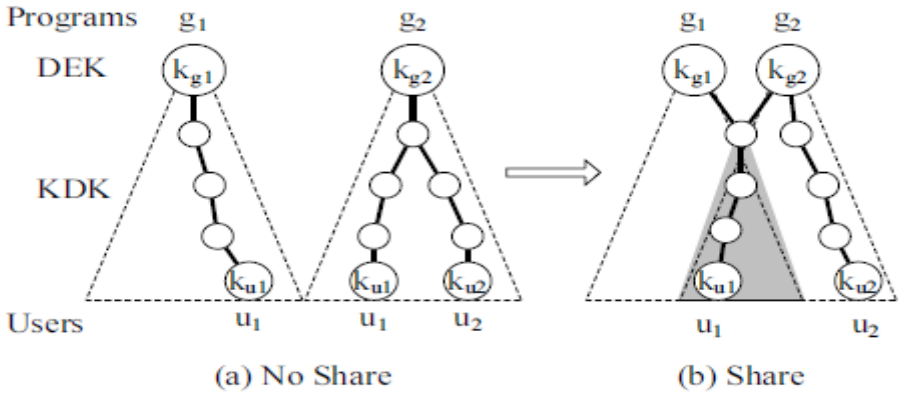


Figure 3: Shared Key Tree

Algorithm of SCRFKM in Broadcast Server	5: end if
1: if a join or shift event happens then	6: if a leave or shift event happens then
2: according to Theorem of Critical Key, find all critical keys in the tree the node wants to join or shift to;	7: change all keys in the leave path, and broadcast corresponding rekey messages;
3: select the best enroll path that has the minimum number of critical keys;	8: end if
4: change all critical keys in the best enroll path, and broadcast corresponding rekey messages;	

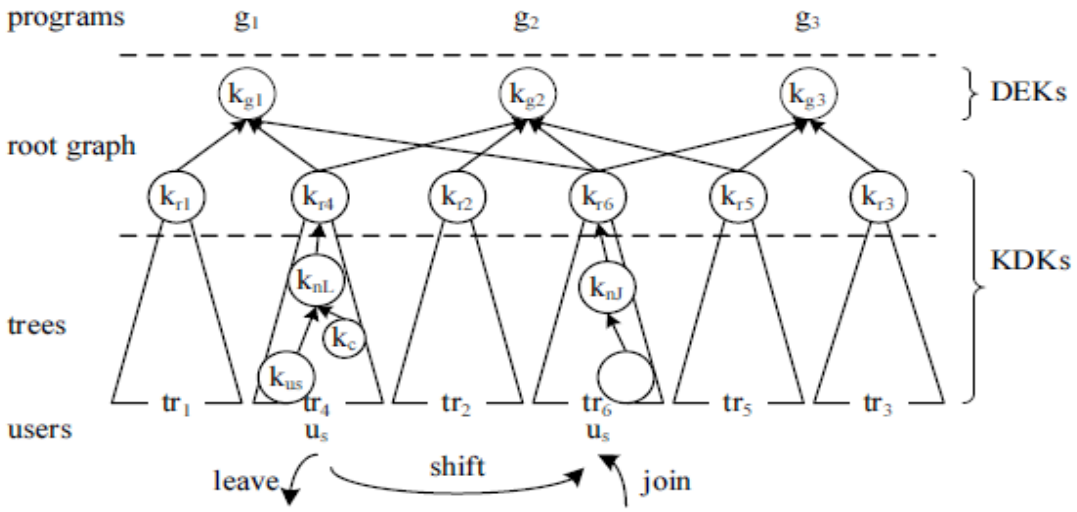


Figure 4: Key Forest

b. Security Analysis

To ensure multicast or broadcast security, group key management should satisfy four security properties [2], [3]: non group confidentiality, collusion freedom, future confidentiality (forward secrecy), and past confidentiality (backward secrecy).

In the following, we discuss how SCRFKM satisfies these properties.

Non-group confidentiality: passive adversaries should not have access to any group key. Because keys are encrypted when being broadcast, passive adversaries can not decrypt any key without knowing decryption key.

Collusion freedom: by sharing group keys, multiple present nodes can not derive any group key that they are not holding. When multiple nodes collude, they may try to share their keys to derived unknown group keys. The sharing can be represented by a sub graph of the paths belonging to the colluding nodes. However, in SCRFKM, a node does not know any key not on this path. Hence, colluding nodes do not know any key outside the sub graph that represents the collusion.

Future confidentiality (forward secrecy): a leaving node should not have access to any group key after leaving his present group. According to Algorithm, SCRFKM changes all keys in the leave path, because the leaving node holds these keys. Hence, the leaving node will not have the new keys after the node leaves his group.

Past confidentiality (backward secrecy): a joining node added at time t should not have access to any keys used to encrypt data before t. According to Algorithm, SCRFKM changes all critical keys in the enroll path when a node joins. It has been basically proved that the joining node can only derive past group keys from critical keys. Hence, changing critical keys and reusing non-critical keys prevent the joining node from obtaining past group keys.

In this study we analyze and evaluate the performance of SCRFKM at the server side and the client side respectively. We define the following parameters for the analysis.

- The service provides m programs.
- n users subscribe the service.
- n users are distributed to e trees.
- Each tree is shared by d programs on average.
- The leave rate is λ_l: the number of users unsubscribing to the service per unit time.
- The join rate is λ_j: the number of users subscribing to the service per unit time.
- The shift rate is λ_s: the number of users changing subscriptions per unit time.
- The total event rate $\Delta = \lambda_l + \lambda_j + \lambda_s$.

IV. Performance Evaluation

Server Side Analysis: Since a server is generally abundant in energy and memory, its computation capacity becomes the main factor that affects the performance of the whole system. If the processing time for each event is large, this would delay user's request. We measured the management cost of a server by two metrics: (a) the total number of keys to be managed, and (b) the number of keys to be inspected and updated per rekey event.

Client Side Analysis: At the client side, three main performance metrics need to be measured: *average rekey message size per event*, *average number of decryption per event per user*, and *maximum number of keys to be stored*, that can well capture the overhead of SCRFKM on resource limited mobile devices in terms of communication, storage, power consumption and computation. For example, if we decide a particular encryption algorithm, we know the length of a key, the time to compute a key, and the energy consumption to execute the algorithm. Consequently, we can obtain metrics, such as the communication overhead in the rekey messages, etc.

Simulation: The analysis gives estimates on major performance metrics. We notice that some factors could bring more insightful results. For example, users may not be evenly distributed in trees due to the fact that some programs may be more popular than other programs. Also, users may be more or less likely to stay in current subscriptions than to frequently change their subscriptions. Hence, in the following, we use simulation to examine the impacts of these user behaviors. In the simulation, we compare SCRFKM with two other representative schemes: SKT, LKH to illustrate how *shared key* improves the performance of key management in wireless broadcast services. SKT is the approach by Y. Sun and K. R. Liu where only shared key tree is applied. LKH does not adopt shared key tree. If shared key tree is adopted, key management is based on the key forest as illustrated in Figure 8; otherwise, a key tree is created for each program and a user is assigned to all trees corresponding to the programs he subscribes.

Settings: We assume that the server provides 50 programs. In the experiments carried out the key forest consists of 300 trees when shared key tree is adopted. Each tree represents a different option of subscriptions. Another assumption is that there are 10000 users (on average) subscribing to the services. The root graph in key forest was automatically generated according to subscriptions. Each program had a different depth in root graph depending on the number of subscriptions that include the program. In the worst case where a program is included in all subscriptions, the depth for this program in root graph is 9. Most programs had a depth of 8 or less, because the probability that a program is selected by more than 256 subscriptions is low. At the same time, the tree depth is around 5, since 10000 users are randomly distributed to 300 trees. In each experiment, two equivalent key graphs are compared for shared key schemes and non-shared key schemes. A random key forest is generated and users are assigned to leaf nodes of the key forest. Then, for non-shared key schemes (LKH), the users are assigned to different key trees according to their subscriptions in shared key schemes. Any user event in shared key schemes was also mapped into the equivalent event in non-shared key schemes.

Table 1.4 Cases in Key Management

Case	Major subscriptions	Major events
Case I	Multiple	Join and Leave
Case II	Single	Join and Leave
Case III	Multiple	Shift
Case IV	Single	Shift

Four test cases are generated for evaluation based on major subscriptions and major events (as summarized in Table 1.4). In Case I and Case III, 80% of the users subscribe to multiple programs and the other 20% only subscribe to one of the programs. In Case II and Case IV, 20% of the users subscribe to multiple programs and the other 80% subscribe to only one program. Furthermore, in Case I and Case II, the major events are joins and leaves; while in Case III and Case IV, the major events are shifts. In the simulations, we vary the rates for the major events while keeping the other rates at 1.

Average Rekey Message Size Per Event: The performance of the key management schemes is evaluated in terms of average rekey message size, by fixing $\lambda_s = 1$ and varying λ_l (x-axis) from 1 to 9 as shown in Figure 10(a) and (b).

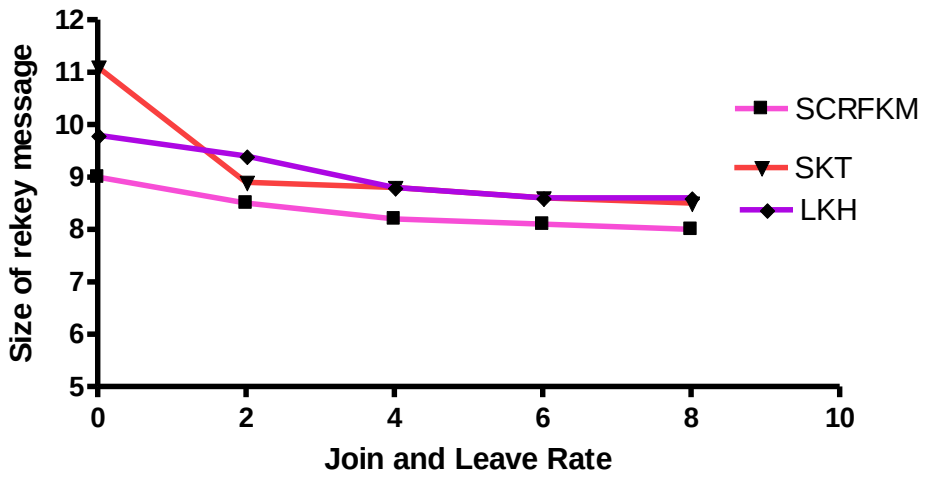
SCRFKM and SKT that adopt shared keys significantly outperforms LKH, Because the major user events are join and leave, a user only needs to join or leave one tree when he subscribes or unsubscribes to multiple programs. In contrast, LKH that does not have shared keys requires a user needs to join or leave multiple trees for multiple programs. Furthermore, SCRFKM is better than SKT, because SCRFKM allows the server to change fewer keys. Then, we evaluate performance of the key management schemes by fixing $\lambda_l = 1$ and varying λ_s (x-axis) from 1 to 9 as shown in Figure 10(c) and (d). Obviously, when the shift rate grows, the performance of SKT turns worse because of the extra overhead that is introduced in managing shared keys when a user quits or adds some but not all of his subscribed programs.

By comparing CASE III and CASE IV (corresponding to the subscriptions of multiple programs and single program, respectively), we found that the extra overhead of shift is higher in CASE III. Figure 5 also shows that SCRFKM and SKT are more sensitive to these types of major user events than LKH. The average rekey message size in SCRFKM and STK grows as the shift rate increases and shrinks as the join/leave rate increases. On the contrary, the average rekey message size in LKH remains almost flat regardless of the rate of user events.

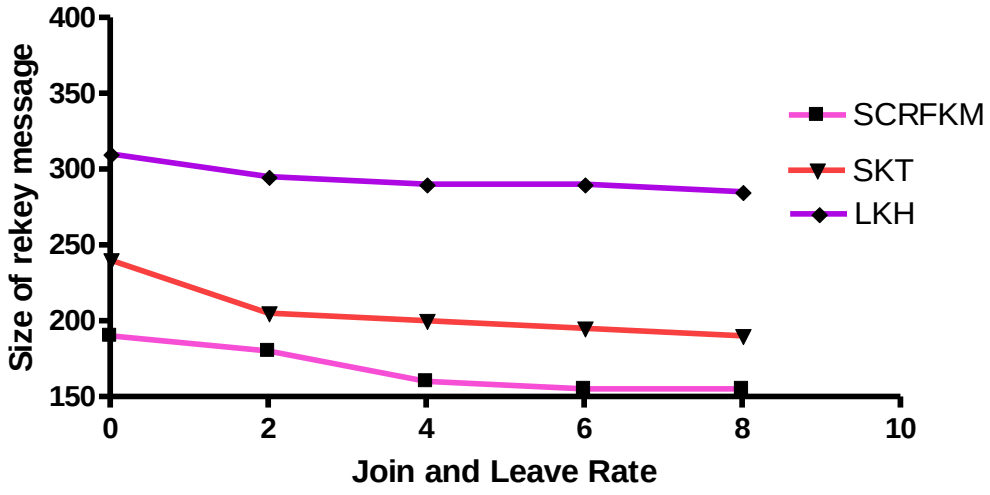
Based on our experimental results, by adopting only shared key tree, SKT reduces the rekey message size to around 60% to 90% of LKH. This result also validates our claim that many keys in the enroll path do not need to be changed. In fact, over all the experiments, only around 23% keys in the enroll path need to be changed.

Average Number of Decryption per Event per User: Power consumption and computation cost are two primary concerns for mobile users. We use the average number of decryptions to measure these costs. Similar to the experiments in the previous section, we vary the rates of major events to observe their impacts on decryption overhead. Figure 11 shows that the scheme SCRFKM is better than SKT and LKH. The number of decryption in LKH, and SCRFKM is around 80% of SKT. In all schemes, the number of decryptions drops as the join/leave rate increases and rises as the shift rate increases.

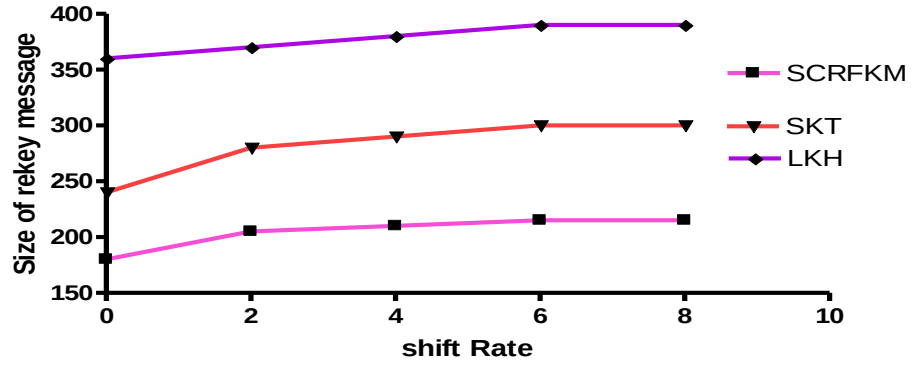
(a) Case I



(b) Case II



(C)Case III



(d) Case IV

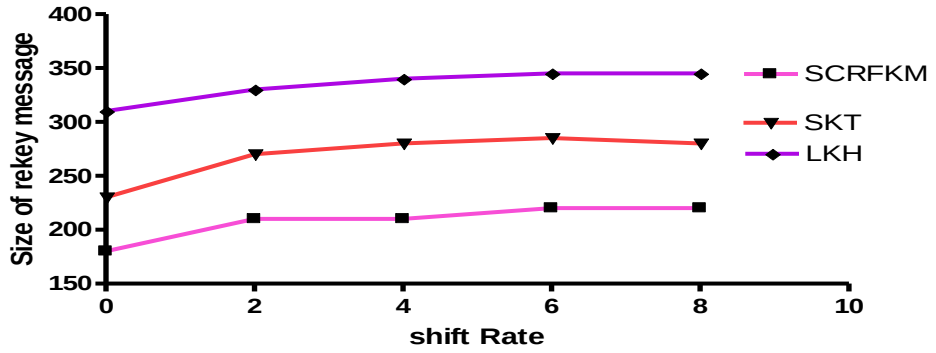
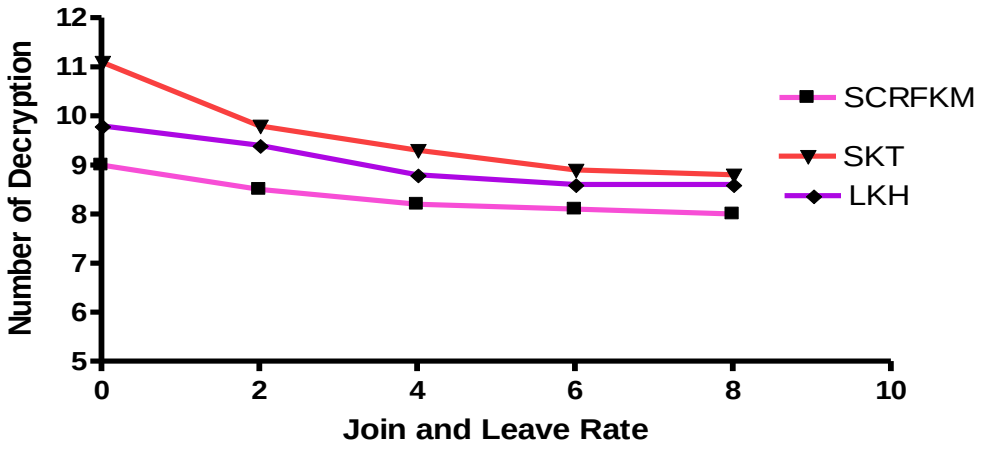


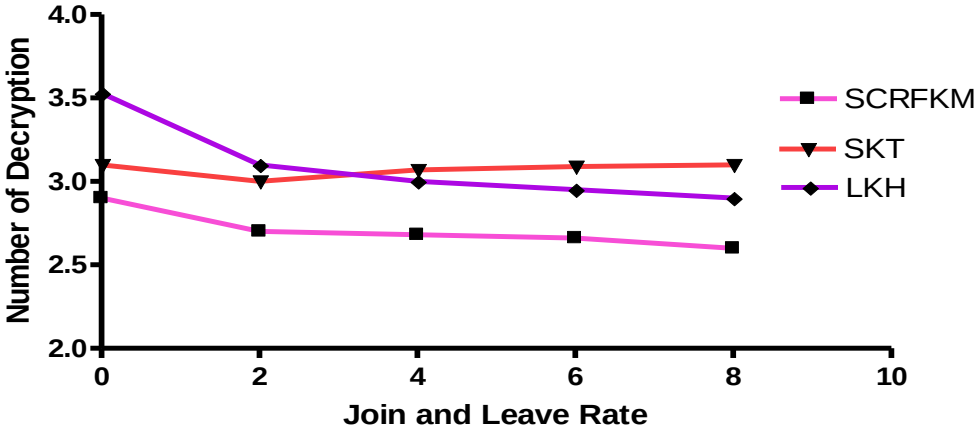
Figure 5 Average rekey message size per event

However, the adoption of shared key has negative impacts on reducing user's decryption cost. In Figure 6, LKH is better than SKT, especially when the shift rate is high in comparison with SCRFKM. This can be explained as follows. In shared key schemes, when a user shifts from a tree to another, some users will be affected by the changed keys in both the leave and the enroll paths if they subscribe to the programs that the user keeps subscribing to. As previously discussed, shared key schemes introduce extra rekey cost because they change keys in two paths. On the contrary, in non-shared key schemes, no key needs to be changed for the programs the user keeps subscribing to. Hence, those users who are affected by keys in two paths in shared key schemes only need to decrypt keys in the leave path in non-shared key schemes. As a consequence, the decryption cost per user is less in non-shared key schemes that that in shared key schemes. Figure 11 also shows that the user subscription pattern has a great impact on the average number of decryptions. The average number of decryptions in Case II and Case IV (where only 20% of users subscribe multiple programs) is around 55% of that in Case III and Case IV (where 80% of users subscribe multiple programs). Obviously, if a user subscribes to more programs, it is more likely that he will be affected by other user activities.

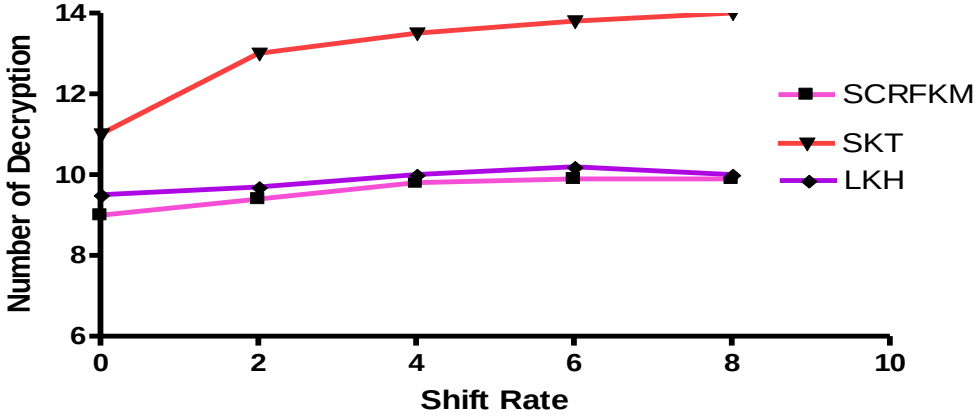
(a) Case I



(b) Case II



(c)Case III



(d)Case 4

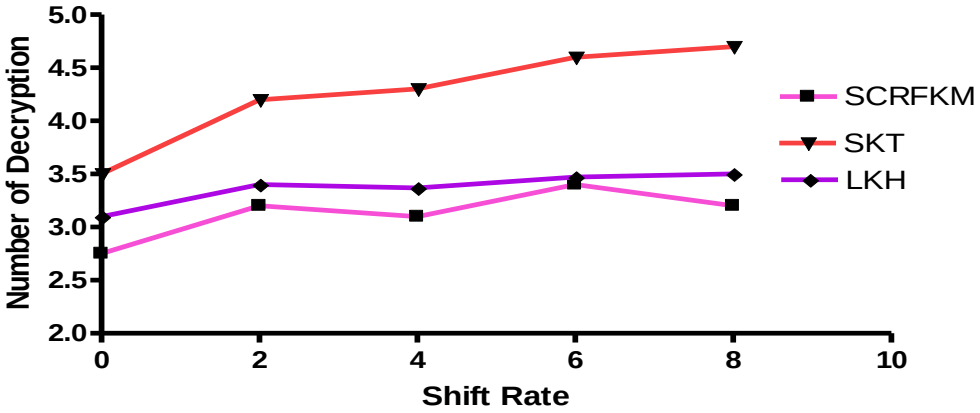


Figure 6 Average number of decryption per event per user

V. Conclusion & Future Scope

In this work, we investigated the issues of key management in support of *secure* wireless broadcast services for route formation. We proposed SCRFKM as a scalable, efficient and secure key management approach in the broadcast system. We used the key forest to exploit the overlapping nature between nodes and programs in broadcast services. SCRFKM let multiple programs share a single tree so that the nodes subscribing these programs can hold fewer keys. In addition, we proposed a novel shared key management approach to further reduce rekey cost by identifying the minimum set of keys that must be changed to ensure broadcast security. This approach is also applicable to other LKH-based approaches

to reduce the rekey cost as in SCRFKM. Our simulation showed that SCRFKM can save about 30% of communication overhead in the broadcast channel and about 35% of decryption cost for each node, compared with the traditional LKH approach. Further deep investigations have to be explored in case of Rekey Operations

REFERENCES

[1] J. Xu, D. Lee, Q. Hu, and W.-C. Lee, "Data broadcast," in *Handbook of Wireless Networks and Mobile Computing*, I. Stojmenovic, Ed. New York: John Wiley and Sons, 2002, pp. 243–265.
[2] D. Wallner, E. Harder, and R. Agee, "Key management for multicast:issues and architectures," *IETF RFC 2627*, 1999.
[3] J. Snoeyink, S. Suri, and G. Varghese, "A lower bound for multicast key distribution," in *IEEE infocom*, vol. 1, 2001, pp. 422–431.
[4] S. Mittra, "Iolus: a framework for scalable secure multicasting," in *ACM SIGCOMM*, vol. 277-288, 1997.
[5] C. K. Wong, M. Gouda, and S. S. Lam, "Secure group communications using key graphs," in *ACM SIGCOMM*, 1998, pp. 68–79.
[6] Y. Kim, A. Perrig, and G. Tsudik, "Simple and fault-tolerant key agreement for dynamic collaborative groups," in *ACM CCS*, 2000, pp.235–244.
[7] S. Setia, S. Koussih, S. Jajodia, and E. Harder, "Kronos: a scalable group re-keying approach for secure multicast," in *IEEE Symposium on Security and Privacy*, 2000, pp. 215–228.
[8] Y. R. Yang, X. S. Li, X. B. Zhang, and S. S. Lam, "Reliable group rekeying: a performance analysis," in *ACM SIGCOMM*, 2001, pp. 27–38.
[9] M. Onen and R. Molva, "Reliable group rekeying with a customer perspective," in *IEEE GLOBECOM*, vol. 4, 2004, pp. 2072–2076.
[10] B. Briscoe, "Marks: zero side effect multicast key management using arbitrarily revealed key sequences," in *NGC*, 1999, pp. 301–320.
[11] A.Wool, "Key management for encrypted broadcast," *ACM Transactions on Information and System Security*, vol. 3, no. 2, pp. 107–134, 2000.
[12] M. Just, E. Kranakis, D. Krizanc, and P. v. Oorschot, "On key distribution
via true broadcasting," in *ACM CCS*, 1994, pp. 81–88.
[13] M. Luby and J. Staddon, "Combinatorial bounds for broadcast encryption," in *Advances in Cryptology, Eurocrypt*, 1998, pp. 512–526.
[14] A. Fiat and M. Naor, "Broadcast encryption," in *Advances in Cryptology, CRYPTO*, 1994, pp. 480–491.
[15] C. Blundo and A. Cresti, "Space requirements for broadcast encryption," in *Advances in Cryptology, Eurocrypt*, 1994, pp. 471–486.
[16] D. Naor, M. Naor, and J. B. Lotspiech, "Revocation and tracing schemes for stateless receivers," in *Advances in Cryptology, CRYPTO*, 2001, pp. 41–62.
[17] *Basic Interoperable Scrambling System*,
http://www.ebu.ch/CMSimages/en/tec doc t3292 tcm6-10493.pdf,2002.
[18] "North american mpeg-2 information," http://www.coolstf.com/mpeg/.
[19] "Irdeto access," http://www.irdeto.com/index.html, 2006.
[20] Markus G. Kuhn, "Analysis of the nagravision video scrambling method," University of Cambridge, Tech. Rep., 1998.
[21] "Viaccess," http://www.viaccess.com/en/, 2006.
[22] "Nds videoguard: Security, flexibility, and growth,"
http://www.ndsworld.com/conditional access/conditional access.html , 2006.
[23] R. Canetti, J. Garay, G. Itkis, D. Micciancio, M. Naor, and B. Pinkas, "Multicast security: a taxonomy and some efficient constructions," in *IEEE Infocom*, vol. 2, 1999, pp. 708–716.
[24] A. Perrig, D. Song, and D. Tygar, "Elk, a new protocol for efficient large group key distribution," in *IEEE Symposium on Security and Privacy*, 2001, pp. 247–262.
[25] C. K. Wong and S. S. Lam, "Keystone: a group key management service," in *International Conference on Telecommunications*, 2000.
[26] M. Moyer, J. Rao, and P. Rohatgi, "Maintaining balanced key trees for secure multicast," *raft-irtf-smug-key-tree-balance-00.txt*, 1999.
[27] J. Staddon, S. Miner, M. Franklin, D. Balfanz, M. Malkin, and D. Dean, "Self-healing key distribution with revocation," in *IEEE Symposium on Security and Privacy*, 2002, pp. 241–257.
[28] Y. Sun and K. R. Liu, "Scalable hierarchical access control in secure group communications," in *IEEE Infocom*, 2004.

