Journal of Computer Science 7 (3): 409-415, 2011
ISSN 1549-3636

Topological Decoupled Group Key Management for Cellular Networks

Jorge Estudillo Ramirez, Saul Eduardo Pomares Hernandez,
Gustavo Rodriguez Gomez and Jose Roberto Perez Cruz
Department of Computer Science,
National Institute of Astrophysics, Optics and Electronics,
Luis Enrique Erro # 1, 72840, Sta. Maria Tonantzintla, Puebla, Mexico

Abstract: Problem statement: The continuous increasing capacity of the cellular networks motivates the development of multiparty applications, such as interactive mobile TV and mobile social networks. For these environments, security group services are required. A practical way to provide security services is by using cryptographic methods. However, the key management needed for these methods, which considers a dynamic group membership, introduces a high communication and storage overheads. **Approach:** In this study we propose an efficient group key management scheme suitable for cellular networks. **Results:** Our scheme reduces the number of keys to be transmitted and to be stored at a mobile host in the presence of membership changes. The scheme is based on a two tier structure to organize the cells in areas and the mobile hosts in clusters within an area. The main objective of the two tier structure is to dissociate, in an advantageous manner, the mobile hosts' distribution from the topological network. **Conclusion:** Our approach offers security services to a large number of mobile hosts by using lower cryptographic resources, thus providing us a more efficient key updating process.

Key words: Group key management, key updating, secure group communication, cellular networks, Session Key (SK), Group Key Management (GKM), Logical Key Hierarchy (LKH), hexagonal shape, Multimedia Group Communication (MGC), Base Station (BS)

INTRODUCTION

The advances in cellular networks along with the developments on multicast communication motivate the deployment of several multiparty multimedia applications on mobile environments. Examples of such applications are interactive mobile TV and mobile social networks (Tjondronegoro *et al*., 2006; Pietilainen *et al*., 2009; Gaol and Widjaja, 2008). One common aspect in these applications is the requirement of efficient security group communications services.

A practical way to provide group security services is by using cryptography methods, where keys are shared among the group of members. For dynamic groups, the membership frequently changes, introducing the need of to update the shared keys. When a new member joins the group, it is necessary to prevent such member from accessing the previously transmitted data (backward secrecy). On the other hand, when a member leaves the group, such member must be disabled from continuing to access the new data transmitted (forward secrecy). The process of updating keys is called rekeying and it is handled by the group key management.

The key management for dynamic groups introduces high communication and storage overheads. Although there are several group key management schemes in the literature (Hardjono *et al*., 2003; Rafaeli and Hutchison, 2003; Eskicioglu, 2003), they are not suitable for cellular networks since these networks are characterized by a limited storage and processing capabilities at the mobile devices, in addition to presenting a limited bandwidth on wireless channels. This is the reason that motivates the design of better group key management schemes for such environments.

In this study we propose an efficient group key management scheme suitable for cellular networks. Our scheme reduces the number of keys to be updated. At the communication channel, we reduce the number of keys to be transmitted; and at a mobile host we reduce the number of keys to be stored and the number of ciphering operations.

The scheme is based on a two tier structure in order to dissociate the mobile hosts' distribution from the topological network. The two tier structure logically organizes the entities of the system in the

Corresponding Author: Jorge Estudillo Ramirez, Computer Science Department, National Institute of Astrophysics, Optics and Electronics (INAOE), Luis Enrique Erro # 1, 72840, Sta. Maria Tonantzintla,Puebla, Mexico

following way. In the first tier, contiguous cells are organized in entities called areas. In the second tier, the mobile hosts within an area are organized in logical entities called clusters. A cluster intersects one or more cells in an area. At each cluster, an individual key hierarchy is used, making transparent the mobile host cell distribution. Our scheme allows us to offer security services to a large number of mobile hosts by transmitting a reduced set of keys in the rekeying process, due to the way we use the clusters and areas. This attribute is the core of our scheme.

Related work: The main function of a Group Key Management (GKM) is to update a set of keys each time the group membership changes during a work session (Hardjono *et al*., 2003). This process is called rekeying. The rekeying performance is commonly evaluated using the following parameters: communication cost, measured by the number of exchanged messages during a rekeying operation; storage cost, measured by the number of keys stored by the group entities; and computational cost, measured by the number of encryption/decryption operations performed to obtain the updated keys.

Several solutions for group key management have been proposed. The broadly used technique to organize keys for a GKM is the Logical Key Hierarchy (LKH) (Wu *et al*., 2009; Xu *et al*., 2008) since it allows the reduction of the communication and storage overhead. The LKH organizes the keys in balanced trees. The set of keys in a hierarchy is called Key Encryption Keys (KEKs). The root key is used as the Session Key (SK). The set of keys of the leaves are used as individual keys for the members. A member knows and stores the set of KEKs in the path from its individual key to the session key. The costs using LKH are O(d log n) for communication, O(log n) for storage at a group member and $O(\frac{dn-1}{d-1})$ at a group manager (Rafaeli and Hutchison, 2003).

A way to increase the efficiency of the LKH is using derivation techniques (Jen-Chiun *et al*., 2009; Gu *et al*., 2009) which enable to the group of members to derive new keys instead of being ciphered and transmitted by the key server. The derivation techniques create new keys from already existing keys. With this kind of technique, the communication cost may be reduced to even O(log n).

There are some approaches based on LKH designed for cellular networks (Um and Delp, 2008; Sun *et al*., 2004; Wang *et al*., 2006; Bruschi and Rosti, 2002). All these approaches are based on the network topology. This means that their key structures reflect the physical distribution of the entities. The main advantage of associating the key structure to the network topology is that the transmission of messages is bounded to a small region. However, strongly coupled key structure to the physical topology has as a main disadvantage: a significant increase in the communication and storage overhead, especially when the tracking of mobile devices is needed. Tracking refers to the task of determining the current location of a mobile host in the system. The tracking results in the relocation of the mobile host from one key structure to another when it changes from communication service point.

MATERIALS AND METHODS

System model: In this study we consider that a distributed Multimedia Group Communication (MGC) runs on a cellular network which consists of two kinds of entities: Base Station (BS) and Mobile Host (MH). A BS has the necessary infrastructure to support and to communicate with mobile hosts. The BS communicates with mobile hosts through wireless communication channels. The geographic area covered by a BS is called cell. An MH is an entity that undergoes BSs while retaining its network connection. At any time, an MH is assumed to be served by at most one BS which is called its local BS. An MH can communicate with other MHs and BSs only through its local BS. We have two communication levels: inter-base and intra-base. The inter-base communication is provided by a static network, which is formed by wired channel connecting BSs. The intra-base communications is provided by a wireless channel that links MHs to BSs. A representation of such a cellular network is depicted by Fig. 1. Bandwidth on wireless channels is a scarce resource compared to that of wired channel. We assume both inter-base and intra-base communication levels are reliable, with an arbitrary but finite amount of time to deliver messages. The mobility of an MH among BSs is managed by a handoff procedure at a communication level (Li *et al*., 2008).

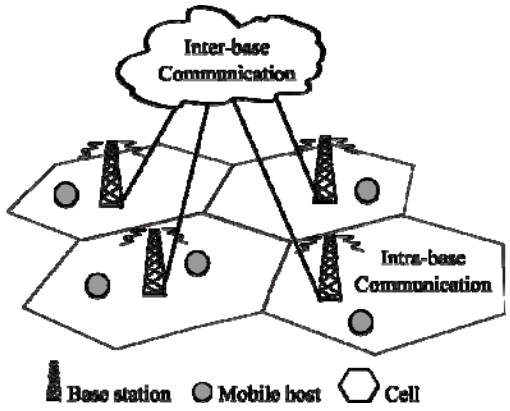

Fig. 1: Example of cellular network

Generally, the cellular networks are modeled as hexagonal structures for design purposes (Alhunaity, 2006). Each hexagonal shape represents a cell, as it is depicted on Fig. 1. Generally, all existent theoretical analysis for this kind of network is based on such representation. In this study we use the hexagonal representation to explain our proposed scheme.

RESULTS

Topological Decoupled Key Management Scheme for Cellular Network (TDKMS-CN): In this section we introduce the proposed group key management scheme suitable for cellular network. The scheme is based on a two tier structure to organize the cells in areas and the mobile hosts in clusters within an area. This reduces the number of keys to be transmitted and to be stored at a mobile host in the presence of frequent membership changes. Based on this structure, we build the key architecture. Then, we present the rekeying procedures for join and leave events that compose the scheme.

Organizational model: The entities of the system are organized in a two tier structure. This arrangement dissociates the mobile hosts' distribution from the topology of the network.

In the first tier, we divide the entire wireless cellular network into small group of cells called area. Each area is composed of seven contiguous cells, as it is depicted in Fig. 2.

In each area we have a Group of Base Stations (BSG) and a Group of Mobile Hosts (MHG). We define an Area Key Controller (AKS) on each area. The AKS generates the Session Key (SK) and distribute it to its entire area.

In the second tier, the AKS subdivides the Mobile Host Group (MHG) located at an area into some clusters, as it is depicted in Fig. 3. Each base station in the BSG is assigned as a Cluster Controller (CC) which is responsible for the group key management for its cluster in the area. With this, we still affect only a small number of cells as it is done by topology matching approaches, but we reduce the number of rekeying messages and the number of rekeying processes triggered during a session since no tracking is needed for a mobile host within an area.

Key architecture: In the TDKMS-CN, in the first tier, the keys and mobile hosts in an area are organized in a key forest. The key forest is composed of different key trees, each one associated with a cluster (Fig. 4). Each key tree is maintained by a base station. The tree is composed of the Key Encryption Keys (KEKs) and the Cluster Key (CK). The KEKs are used as auxiliary keys for the rekeying operations. The CK key is used to securely communicate the Area Key (AK), which is used to distribute in a secure manner the Session Key (SK).

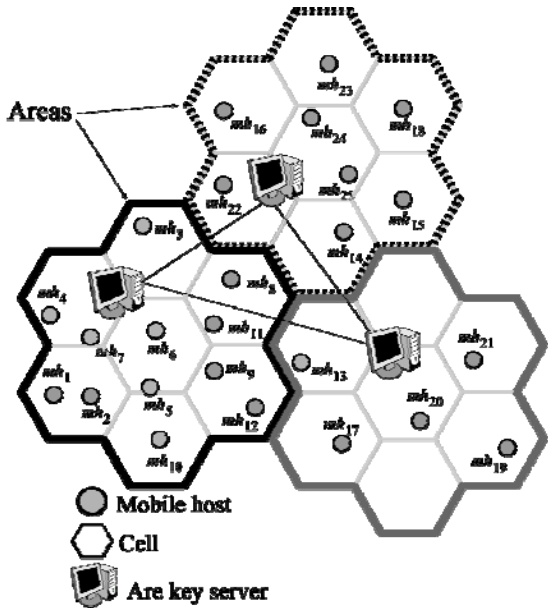

Fig. 2: Key management arrangement

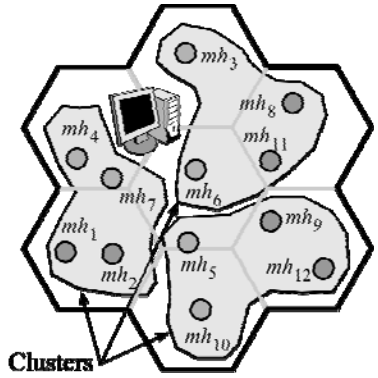

Fig. 3: Example of cluster organization

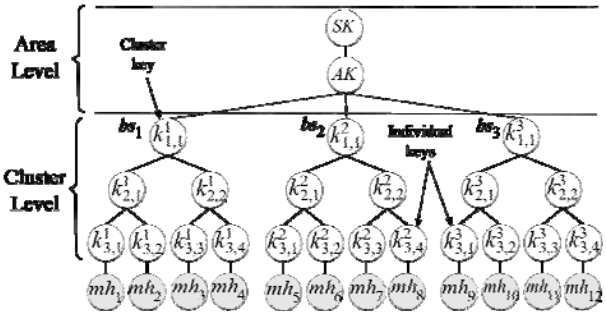

Fig. 4: Key Management Architecture

Table 1: Nomenclature

$x \rightarrow y$	x sends a unicast message to y
$x \Rightarrow y$	x sends a multicast message to y
$\{x\}_y$	A key x encrypted with the key y
x'	A new version of the key x
{id(x)}	A message with the index of the key x

In the second tier, all the roots of the key forest are logically connected to the common AK, which is maintained by the AKS.

Rekeying processes: Below we introduce the procedures to update the key management architecture for the rekeying process from join and leave events. The procedures use the nomenclature given in Table 1. As a form to improve the generation of key we implement the derivation techniques described next.

Key derivation technique: We introduce the shared derivation key technique over key tree presented in the work of Lin *et al*. (2009). It improves the performance of rekeying operations by allowing members to derive new keys by themselves. The improvement is achieved by the use of a key derivation function f (.) in the server and in the group members. With this function, new keys are derived based on old keys, which are called the derivation keys.

We assume that the key structure has a tree arrangement and a node can be added or deleted at a time. A key tree or key subtree whose root node is $x_{i,j}$ is denoted as $y_{i,j}$ and the key $k_{i,j}$ stored in the node $x_{i,j}$ is also called the key of $y_{i,j}$.

Assume that x_{h,p_h}, the root of subtree y_{h,p_h}, is the last internal key node on the adding path. Adding a new node in a key tree implies two scenarios: when x_{h,p_h} is not full and when x_{h,p_h} is full.

In the first scenario, the new external key node $x'_{h+1,p_{h+1}}$ will be placed under it. In the second scenario, a new internal key node $x'_{h+1,p_{h+1}}$ is created to replace the place of the old child $x_{h+1,s_{h+1}}$. Then $x_{h+1,s_{h+1}}$ becomes $x'_{h+2,s_{h+2}}$ and is placed as a child of the new internal key node $x'_{h+1,p_{h+1}}$. The new external key node $x'_{h+2,p_{h+2}}$ will be placed under the new internal key node too. For both scenarios the new key of the node x'_{i,p_i}, $1 \le i \le h$, will be computed as follows:

$$k'_{i,p_i} = f(k_{i,p_i})$$

where, k_{i,p_i} is the derivation key. For the new internal key node created, the new key is computed as:

$$k'_{h+1,p_{h+1}} = f(k'_{h+2,s_{h+2}} \oplus k_g)$$

where $k'_{h+2,s_{h+2}} = k_{h+1,s_{h+1}}$ is the derivation key and $k_g = k_{1,1}$ is known as the salt value.

Again, assume that x_{h,p_h}, the root of subtree y_{h,p_h}, is the last internal key node on the deleted path. Eliminating a node in a key tree implies two scenarios; when x_{h,p_h} has at least two children and when x_{h,p_h} has less than two children.

In the first scenario, x_{h,p_h} becomes x'_{h,p_h}. In the second scenario, x_{h,p_h} will be replaced by the root key node of the remaining child subtree $y_{h+1,s_{h+1}}$ and becomes x'_{h,p_h}. For both scenarios the new key of the node x'_{i,p_i}, $1 \le i \le h$ or $1 \le i \le h-1$ will be computed as follows:

$$k'_{i,p_i} = f(k_{i+1,s_{i+1}} \oplus k_{i,p_i})$$

where, $k_{i+1,s_{i+1}}$ is the derivation key and the old key k_{i,p_i} is used as a salt value.

Join rekeying: When a mobile host joins the group it is assigned by the AKS to a cluster. The AKS generates the new SK and AK keys and sends them to its area ciphered with the previous area key:

$$AKS \Rightarrow MHG: \{AK', SK'\}_{AK}$$

The base station controlling the cluster, where the mobile host joins the group, performs the join procedure to rekey the cluster key structure. As an example, let mh_1 in Fig. 4 be a mobile host joining the group via the cluster controlled by the bs_1. The controller will update its key structure. The following keys are updated: $k^1_{3,2} = k^1_{2,1}$, $k'^1_{2,1} = f(k^1_{3,2} \oplus k^1_{1,1})$ and $k'^1_{1,1} = f(k^1_{1,1})$. The CC sends the next messages:

$$bs_1 \rightarrow mh_1 : \{k'^1_{2,1}, k'^1_{1,1}, AK', SK'\}_{k^1_{3,1}}$$

$$bs_1 \Rightarrow cluster_1, cluster_2, cluster_3 : \{id(k^1_{2,1}), id(k^1_{1,1})\}$$

The first message is unicasted to the joining member unable to derive the keys while the second

message is multicasted to the MHG where the old mobile hosts can derive the keys.

Leave rekeying: When a mobile host leaves the group or it is expelled by the AKS, it must be unsubscribed from the cluster it belongs. The AKS generates the new SK and AK keys and send them to its cluster controllers via a secure channel:

$$AKS \rightarrow bs_i : \{AK', SK'\}$$

The controller of a cluster with removed mobile hosts performs the rekeying process. As an example, according to Fig. 4 let mh_1 be the mobile host leaving the group. The controller (bs_1) deletes the key node corresponding to the leaving mobile and performs the rekeying process for the deleted node as it was described above for the key derivation technique. First, bs_1 generates $k'^{1}_{2,1} = k^{1}_{3,2}$ and $k'^{1}_{1,1} = f(k^{1}_{1,1} \oplus k^{1}_{2,2})$; then, bs_1 sends the following messages:

$$bs_1 \Rightarrow MHG: \{k'^{1}_{1,1}\}_{k^{1}_{2,1}}$$

$$bs_1 \Rightarrow MHG: \{id(k^{1}_{2,1}, k^{1}_{3,2}), id(k^{1}_{1,1}, k^{1}_{2,2})\}$$

The first message is sent to mobile hosts unable to derive the keys and the second one is sent to mobile hosts able to derive such keys.

Finally, each cluster controller transmits the AK and SK keys ciphered with its cluster key:

$$bs_i \Rightarrow MHG: \{AK', SK'\}_{k^{i}_{1,1}}$$

Handoff: In our approach we perform the rekeying process only for a mobile host switching from one area to another instead for mobile host switching from one cell to another. This results in a reduction on the rekeying cost for intra area mobility. When a mobile hosts arrives to a different area it performs a rekeying process similar to that performed for join rekeying. The difference here is that the session key is not updated; whereas, when a mobile host exits from an area it performs a rekeying process similar to the leaving process but without updating the session key.

DISCUSSION

Storage, communication and computational costs: In our scheme we use key tree structures of d degree to organize the KEKs (KEK-tree) involved in a cluster. Indeed the KEK-tree uses two additional keys located above the root of the KEK-tree to store the AK and the SK keys. The KEK-trees are maintained as balanced as possible.

We assume that on each cluster we have a n_l mobile hosts. Then the number of key nodes in a KEK-tree is given by $O((dn_l-1)/(d-1))$. As the CC is responsible for maintaining the KEK-tree, it needs to store $O((dn_l-1)/(d-1))$ KEKs and the two additional keys (AK and SK).

The mobile host needs to store the path of KEKs from its leaf to the root of the KEK-tree and the two additional keys. Therefore, the mobile host stores $O(\log_d n_l)$ KEKs and the two additional keys.

The key tree must be updated when a join or a leave event occurs. In both cases, a path of the KEK-tree is compromised. Then, $O(\log_d n_l)$ KEKs and the two additional keys must be updated. However, the transmission cost and the computational cost of the key updating is different in each case.

For the join event, as the new member cannot generate the keys, the CC sends to it a unicast message with the set of updated keys of the KEK-tree and the new AK and SK keys ciphered with the individual key of the new member. Then, the CC needs to cipher $O(\log_d n_l)$ keys and to send $O(\log_d n_l)$ keys to the new member. After that, the new member perform $O(\log_d n_l)$ decipher operations to get the set of keys. For the remaining mobile hosts, the CC sends a multicast message with the updated keys IDs of the KEK-tree. Thus, the CC sends $O(\log_d n_l)$ IDs. The new AK and SK keys are multicasted by using the previous AK key. Then, the number of transmitted keys and the number of cipher/decipher operations are O(1).

For the leave event, when a mobile host leaves the system, the CC multicasts $O((d-1)\log_d n_l)$ messages with KEKs of the affected path to the users which cannot derive them. This means that the CC needs to perform $O((d-1)\log_d n_l)$ cipher operations. Meanwhile, a mobile host needs to perform an average of $\frac{d-1}{dn_l}\sum_{i=0}^{h-2}\frac{n_l}{d^i}$ decipher operations, where $h = \log_d n_l$. The CC also multicasts a message with the key IDs of the affected path to the mobile hosts able to derive the keys. Thus, the CC sends $O((d-1)\log_d n_l - 1)$ IDs. The new AK and SK keys are multicasted with the root of the KEK-tree. Then, the number of transmitted messages is O(1). Each mobile host needs to perform two additional decipher operations in order to obtain the AK and SK keys.

Comparison: We compare our scheme with the SGKM scheme presented by Um and Delp (2008). We use the results given by Um *et al*. and the result obtained in the previous analysis of cost. We compare the number of

keys and the number of secrets that are transmitted under both schemes. We denoted by S the size of a secret and by K the size of a key. In Table 2-5 n_l refers to the number of mobile hosts involved in a key hierarchy and $h = \log_d n_l$ refers to the height of the key tree. In the case of SGKMS, a key hierarchy contains the keys used for a mobile host in a cell. In our scheme, a key hierarchy represents the keys used by a mobile host in a cluster, which is composed by more than one cell.

In Table 2, we refer to a Server as the key server in the SGKMS and to the cluster controller in TDKMS-CN.

From Table 3 we can observe that the communication cost for join and leave events in both schemes are similar. The main difference between SGKMS and TDKMS-CN is that in a base case, when a mobile host undergoes seven contiguous cells (area), with TDKMS-CN the mobile host does not perform any rekeying, while with SGKMS a mobile host needs to update keys each time it changes from one cell to another.

We note that, by using our approach, if the organization arrangement of area-cluster is replicated in the whole system, eventually the number of triggered rekeying processes will be substantially decreased since the rekeying process is triggered at an area level instead of at a cell level.

Table 2: Storage cost

Scheme	SGKMS	TDKMS-CN
Mobile	$O(\log_d n_l)$	$O(\log_d n_l)$
Server	$O\left(\frac{dn_l}{d-1}\right)$	$O\left(\frac{dn_l}{d-1}\right)$

Table 3: Communication cost by single membership event

Scheme	SGKMS	TDKMS-CN
Join	$O((\log_d n_l)S)$	$O((\log_d n_l)K)$
Leave	$O((d \log_d n_l)S)$	$O(((d-1) \log_d n_l)K)$

Table 4: Computational cost on TDKMS-CN

Scheme	Server	Requesting Member	Non-requesting member
Join	$O(\log_d n_l)$	$O(\log_d n_l)$	$O(1)$
Leave	$O((d-1)\log_d n_l)$	0	$\Theta\left(\frac{d-1}{dn_l}\sum_{i=0}^{h-2}\frac{n_l}{d^i}\right)$

Table 5: Computational cost on SGKMS

Scheme	Server	Requesting Member	Non-requesting member
Join	$O(2 \log_d n_l)$	$O(\log_d n_l)$	$\Theta\left(\frac{1}{n_l}\sum_{i=0}^{h-2}\frac{n_l}{d^i}\right)$
Leave	$O(d \log_d n_l)$	0	$\Theta\left(\frac{1}{n_l}\sum_{i=0}^{h-2}\frac{n_l}{d^i}\right)$

In Tables 4 and 5 we estimate the computational cost as the number of cipher/decipher operations performed by the entities of the system. We observe a difference in computational cost at the server level and at non-requesting members. Such difference is the improvement of the key derivation technique. Note that the computational cost for each operation depends on the algorithm implemented for encryption on each scheme.

In both schemes there is a computational cost for evaluating both the key derivation function and the polynomial function. In both cases, this cost depends on the type of function implemented and on the algorithms used to evaluate the function.

In addition, in the SGKMS an additional computational cost emerges from the need of processing the construction of the polynomials used to generate the keys from the shares. There is a 1-1 correspondence between the number of polynomials constructed by the server and the number of encryptions performed by the server. Also, there is 1-1 correspondence between the number of polynomials constructed by each member and the number of decryptions performed by each member.

CONCLUSION

In this study we proposed the Topological Decoupled Group Key Management Scheme for Cellular Networks (TDKMS-CN). Our scheme is efficient in terms of the overhead generated at the rekeying process. At the communication channel, we reduce the number of keys to be transmitted and at a mobile host, we reduce the number of keys to be stored. The scheme is based on an original two tier structure that dissociates the mobile hosts' distribution from the topological network. The dissociation is mainly achieved through the use of clusters. On one hand, a cluster hides the mobile host cell distribution. This attribute allows the scheme to offer security services to a large number of mobile hosts by using a reduced set of keys. On the other hand, a cluster hides the mobility of the hosts resulting in a reduction in the amount of triggered rekeying processes produced by tracking mobility events.

We note that further work is needed in order to evaluate the performance of the proposed scheme under real conditions of membership and mobility management events. Our attention is focused in this direction and we expect to have some interesting contributions shortly.

REFERENCES

Alhunaity, M.F., 2006. Comparative study between various topological models of base stations in cellular mobile radio communication. Am. J. Applied Sci., 3: 1711-1714. DOI: 10.3844/ajassp.2006.1711.1714

Bruschi, D. and E. Rosti, 2002. Secure multicast in wireless networks of mobile hosts: Protocols and issues. Mobile Networks Appl., 7: 503-511. DOI: 10.1023/A:1020781305639

Eskicioglu, A.M., 2003. Multimedia security in group communications: Recent progress in key management, Authentication, and Watermarking. Multimedia Syst., 9: 239-248. DOI: 10.1007/s00530-003-0095-2

Gaol, F.L. and B. Widjaja, 2008. Framework of regression-based graph matrix analysis approach in multi-relational social network problem. J. Math. Stat., 4: 51-57. DOI: 10.3844/jmssp.2008.51.57

Gu, Q., P. Liu, W.C. Lee and C.H. Chu, 2009. KTR: An efficient key management scheme for secure data access control in wireless broadcast services. IEEE Trans. Dependable Secure Comp., 6: 188-201. DOI: 10.1109/TDSC.2008.12

Hardjono, T., Dondeti and R. Lakshminath, 2003. Multicast and Group Security. 1st Edn., Artech House, USA., ISBN: 1580533426, pp: 334.

Jen-Chiun, L., H. Kuo-Hsuan, L. Feipei and L. Hung-Chang, 2009. Secure and efficient group key management with shared key derivation. Comp. Standards Interfaces, 31: 192-208. DOI: 10.1016/j.csi.2007.11.005

Li, J., N.B. Shroff and E.K. Chon, 2008. Channel carrying: A novel handoff scheme for mobile cellular networks. IEEE/ACM Trans. Network., 7: 38-50. DOI: 10.1109/90.759316

Pietilainen, A.K., E. Oliver, J. LeBrun, G. Varghese and C. Diot, 2009. Mobiclique: Middleware for mobile social networking. Proceedings of the 2nd ACM Workshop on Online Social Networks, (WOSN'09), New York, USA., pp: 49-54. DOI: 10.1145/1592665.1592678

Rafaeli, S. and D. Hutchison, 2003. A survey of key management for secure group communication. ACM Comp. Surveys, 35: 309-329. DOI: 10.1145/937503.937506

Sun, Y., W. Trappe and K.J. Ray, 2004. A scalable multicast key management scheme for heterogeneous wireless networks. IEEE/ACM Trans. Network., 12: 653-666. DOI: 10.1109/TNET.2004.833129

Tjondronegoro, D.W., L. Wang and A. Joly, 2006. Delivering a fully interactive mobile TV. Int. J. Web Inform. Syst., 2: 197-211. DOI: 10.1108/17440080780000300

Um, H. and E.J. Delp, 2008. A secure group key management scheme for wireless cellular systems. Int. J. Network Security, 6: 40-52.

Wang, Y., D. Damodaran and P.D. Lee, 2006. Efficient group key management in wireless networks. Proceeding of the 3rd International Conference on Information Technology: New Generations, Apr. 10-12, Las Vegas, NV., pp: 432-439. DOI: 10.1109/ITNG.2006.58

Wu, B., J. Wu and Y. Dong, 2009. An efficient group key management scheme for mobile ad hoc networks. Int. J. Security Networks, 4: 125-134. DOI: 10.1504/IJSN.2009.023431

Xu, J., F. Zhou, Z. Li and M. Yang, 2008. Hierarchical data processing model and complete tree key management mechanism. Proceedings of the 9th International Conference for Young Computer Scientists, Nov. 18-22, Hunan, pp: 1606-1612. DOI: 10.1109/ICYCS.2008.252