SCIENCE CHINA
Information Sciences

• **Supplementary File** •

Area-based mobile multicast group key management scheme for secure mobile cooperative sensing

Jie Cui, Hong Zhong*, Weiya Luo & Jing Zhang

School of Computer Science and Technology Anhui University 230601

Appendix A Basic knowledge

Assuming a pseudo-random sequence generator G where time output length is twice the input length that is G:$\{0,1\}^n \rightarrow \{0,1\}^{2n}$, the attacker cannot distinguish the output of G from a real random string of equal length. By setting $M = G(S)$, we can know that the length of M is twice that of S. Let $G_L(S)$ be the left half of M and $G_R(S)$ be the right half of M.

In our model, a special Security Coprocessor (SC) is embedded in the hardware of the mobile device. SC can prevent any outside access to its internal memory space. It is resistant to security vulnerabilities in that it can preserve and calculate cryptographic keys securely. It has a pseudo-random number generator algorithm G, a special encryption algorithm E, and three input ports. Data and information about the current state is input through the first port, while the encryption algorithm seed is input through the second and third ports. Assuming that the first port receives data M, the second port receives k, and the third port receives r, SC will output the data $L = E_r(G_L(D_k(M)))$ or $L = E_r(G_R(D_k(M)))$ according to the state information, where $D_A(B)$ means B is encrypted by A and means decrypting B through A. The second port and the third port both receive seed k, and they will output the data $L = E_k(G_L(D_k(M)))$ or $L = E_k(G_R(D_k(M)))$ according to the state information.

Appendix B Reference framework

Our model is constructed in two layers, and the first layer is the domain layer, which is the main wired component subject to the Domain Key Distributor (DKD). DKD is equipped to organize the initial distribution of keys and authorize the mobile users. The second layer is in charge of sequential Area Key Distributors (AKD).

The DKD manages every AKD, and each of these AKDs manages its cluster in a point-to-point pattern in our model. Upon logging onto the DKD, users request the DKD to allocate lk, a key that is used privately and through a safe channel, and they transmit these private messages to the AKD in the users location, and the messages are saved in a list that contains key distribution exclusive to the user under the management of AKD. Each AKD receives the service data through DKD. Subsequently, these AKDs send these data clusters to the corresponding users. This diminishes the load in the network. When users shift from one cluster area to another, a move-notify command will be transmitted to the corresponding AKDs, which will then independently renew the keys that fall into their cluster, ensuring forward and backward secrecy.

Appendix C Performance analysis

1. **Function analysis**

   The scheme of our project contains a DKD that is accountable for the cellphone users registration process and the generation of their private keys. In addition, DKD is accountable for the multicast of all the data in service to every AKD. AKD manages the update and renewal of users keys, which are then distributed in its situated group, so that the 1-affect-n effects are enhanced and transmission overhead in the main network is reduced. Only key sk that is used in a long term is shared with every AKD by DKD, which transmits all the data in service coded by sk to all the AKDs. The AKD further allocates them, which are coded into respective group key to the users who subscribe. We utilize decentralized framework to prevent single point of error so that even if one or more AKDs fail, the rest of them that do not associated with those failed AKDs can still maintain normal operation. Each of AKDs manages its

* Corresponding author (email: zhongh@ahu.edu.cn)

corresponding UPKDL in an independent manner. When cellphone user handoffs from one cluster to another cluster, the target AKDv will receive the corresponding UPKDLi rows from original AKDi, so AKDv can be informed of the private keys and subscribed service ids of cellphone users and then authenticate them. The generic comparison of function analysis between the proposed scheme and other schemes is summed up in **Table C1**.

Table C1 Comparison of function analysis

Evaluation criteria	BS	IR	FEDRP	GKMF	KELLIL	SMGKM	OURs
Decentralized framework	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Membership changes	No	No	No	No	No	No	Yes
1-affect-n	Yes	Yes	Yes	Yes	Yes	No	No
AKD to AKD communication link	No	No	No	No	No	Yes	Yes
Support multiple group services	No	No	No	No	No	Yes	Yes
Single point of failure	Yes	Yes	Yes	Yes	Yes	No	No
DKD scalability	No	No	No	No	No	Yes	Yes
Use of list to manage mobility	No	No	Yes	Yes	Yes	Yes	Yes)
Authentication at move	0.No	No	No	Yes	No	Yes	Yes

2. **Security analysis**

• **Prevent key replacement**: key replacement means that attacker fakes rekeying messages, so as to replace the group key generated by group control center. In our scheme, the group controller does not distribute new group key in the cluster but broadcast rekeying messages so that users can calculate the updated group key by themselves. Therefore, attacker cannot fake group key in our scheme. As for faking the rekeying messages, AKD encrypts the rekeying messages through its private key and users can verify the legitimacy of the messages. Thus attacker cannot fake the rekeying messages due to ignorance about AKDs private key.

• **Prevent key interception**: the key interception means that attacker intercepts the rekeying messages distributed by group control center and cracks the ciphertext, and thereby they can gain the new group key. In our scheme, there are no updated group key in rekeying messages only LKH nodes and a random number. Even if the attacker cracks those data, they will not be able to get the updated group key.

• **Prevent replay attack**: replay attack means that attacker uses the previous rekeying messages sent by the group control center, and then redistributes those messages to the cluster in order to use the old group key again in the cluster. In our scheme, the rekeying messages sent by AKD are encrypted through the current group key, therefore, users have to decrypt ciphertext by the current group key. Replay attack distributes the messages encrypted by the old group key and user cannot decrypt them.

• **Prevent LKH node attack**: the LKH node attack means that attackers frequently join and leave the cluster to gain all the nodes value of LKH, and then crack the system. In our scheme, although node LABELs of LKH stay the same all the time, what users receive is $LABEL'$ encrypted by users short-term private key. Every user has different $LABEL'$. Every time a user joins the cluster, AKD will distribute a new short-term private key to him, as a result, user gets different $LABEL'$s even for the same node every time the node joins the group. And our scheme adopts SC to guarantee the security of calculation, so that attackers cannot get the LABELs of nodes.

• **Forward security and backward security**: the forward security means when a user leaves the group, in order to prevent the future group message being got by the leaving user, the affected keys need to be updated. The backward security means when a user joins the group, the affected keys should also to be updated to stop the joining user getting old messages sent before him joining the group. In our scheme, when a user join or leave the cluster or change membership, all the public keys that the user have would be updated, so that the proposed scheme satisfies the forward security and backward security.

Table C2 Comparison of communication overhead

Reference framework	Communication overhead of Cluster i	Communication overhead of Cluster v	Total communication overhead
IR	$Sd\mathrm{O}(\log_d(M))$+2S+1	$Sd\mathrm{O}(\log_d(M))$+2S+2	$2Sd\mathrm{O}(\log_d(M))$+4S+3
GKMF	$Sd\mathrm{O}(\log_d(M))$+3	S+4	$Sd\mathrm{O}(\log_d(M))$+S+7
SMGKM	$Sd\mathrm{O}(\log_d(M))$+2	S+3	$Sd\mathrm{O}(\log_d(M))$+S+5
OURs	2S+2	3S+1	5S+3

3. **Communication overhead**

The transmission overhead of communication contained in our model is summarized in **Table C2**. It is compared with other correlative models. Communication overhead is defined as the total overhead for rekeying and signal-loading. Suppose that the number of services affected is S when a mobile user moves from one cluster to another. In order to manage the users for the corresponding schemes for comparison, multiple LKHs are constructed in our project, in which the degree of LKH is d, and the subscribing users for each service are managed by a LKH independently. The node at the root of LKH is TEK (traffic encryption keys), and leaf nodes are private keys subject

to the users. The rest of the nodes of LKH are KEK (key encryption keys). The keys of every user range from the root to the leaf part at which nodes are situated.

Upon leaving a cluster, a mobile user's keys may be adjusted in order to assure forward secrecy and these keys are formerly shared by this user and the rest of the users. If a d-degree tree can hold M users, its depth will be given by $\log_d(M)$ and shared by both the aforementioned user and the rest of the users. Consequently, the rekeying transmission cost of a LKH can be described as $d\mathrm{O}(\log_d(M))$. However, in the case of S services, the overhead of rekeying is $Sd\mathrm{O}(\log_d(M))$. Likewise, if a mobile user leaves the group, we ought to adjust the keys as well. Suppose that each time the group operator sends a renewed TEK, it will be firstly coded by the previous TEK, and then it will be coded by KEK accordingly. It is commonly known that the KEKs of the LKH affected by the transmission are not requested to be renewed. Even though the user who joins discovers the KEKs that remain the same, he or she still cannot acquire the previous TEK. Thus, backward secrecy is assured. This is because only those users affected TEKs will be renewed. The rekeying transmission cost of LKHs for S multicast services within a target cluster is expressed as S.

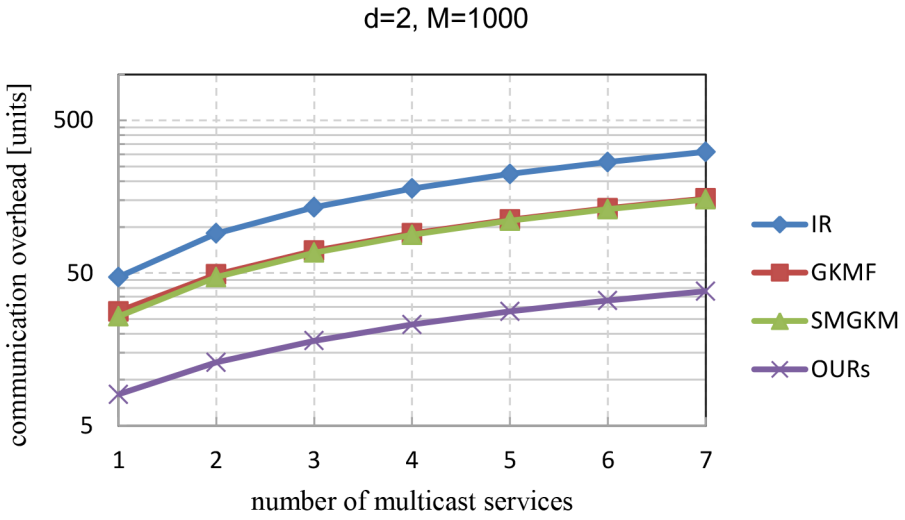

Figure C1 The communication overhead with varying the number of multicast services.

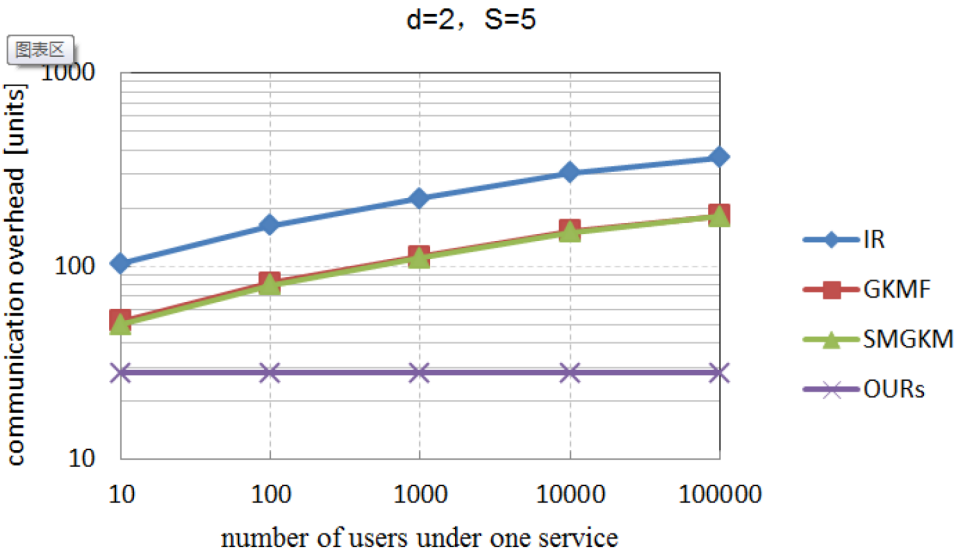

Figure C2 The communication overhead with varying the number of users.

In our scheme, LKH is used to assist users in calculating the group key. In addition to the brother node of the moving user node, all the rest of users in the cluster can update the group key through a same rekeying message sent by AKD. The brother node of the moving user node needs a unicast from AKD. All the calculation of group key is through SC in the client and what client requires is a certain extent computing power. Consider the worst case that there is always a user in a brother leaf node of a moving user node, which means that AKD will unicast a rekeying message to the brother node every time. In order to evaluate signaling load overhead when users handoff between two clusters, we compare the communication overheads as shown in **Table C2**.

As shown in **Figure C1** and **Figure C2**, our scheme has efficient communication overhead as the increasing number of multicast services or users. Our scheme has clear advantages especially when the number of users increases. The depth of LKH increases with the increasing number of users. However, users calculate the new group key by themselves in our scheme, the number of rekeying messages distributed by AKD is not directly correlated with the depth of LKH. Therefore, the communication overhead in our scheme remains the same. This only increases the computational overhead of mobile clients, and clients will only increase a unit of computational cost as the exponential increasing number of users. Therefore, our scheme has a good scalability and is suitable for a large group of users. However, there are still some defect in our scheme, which is that only one move-notify can be dealt with every time. When multiple users simultaneously make a move-notify or change subscribed services, AKD has to deal with them in accordance with the request priority sequence, which will cause the delay to timely respond to users' requests. Therefore, our scheme prefers the environment with relatively low network churns.

Appendix D Signal flow

When a cellphone user U_i handoffs from the cluster i to the target cluster v and discovers a low signal strength (P_i) from AKD_i and a high signal strength (P_v) from AKD_v ($P_i << P_v$), U_i sends a $\{MOVE\_NOTIFY\}_{k_i}$ message to AKD_i. After receiving the notify message, AKD_i verifies the legitimacy of U_i through his long-term private key and then sends U_i row of $UPKDL_i$ to AKD_v through a secure communication link. Therefore, AKD_v could know the private key of U_i as well as subscribed multicast service ID. After a cellphone user U_i arriving at the target cluster v, he/she will also send a message $\{MOVE\_NOTIFY\}_{lki}$ to AKD_v. If AKD_v authenticates U_i , U_i can be allowed to join in the cluster v. Then AKD_v will update the $UPKDL_i$ and multicast rekeying information. In the meantime, if AKD_i finds u_i having left cluster, $UPKDL_i$ and affected keys in the cluster would be updated. The overall signal flow is summarized in **Figure D1**.

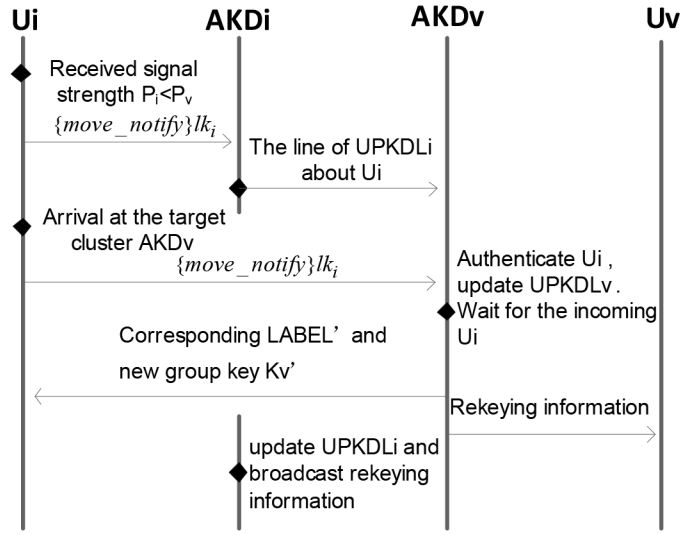

Figure D1 Signal flow for our scheme.