FKM: A Fingerprint-based Key Management Protocol for SoC-based Sensor Networks
Xiaoguang Niu1,4, Yanmin Zhu2,3, Li Cui1, Lionel M. Ni2
1 Institute of Computing Technology, Chinese Academy of Sciences
2 Hong Kong University of Science and Technology
3 Imperial College of London
4 Graduate School of Chinese Academy of Sciences
{niuxiaoguang, lcui}@ict.ac.cn, yzhu@doc.ic.ac.uk, ni@cse.ust.hk

Abstract—Recently, System-on-Chip (SoC) technology has been adopted to design smaller, lower-power and cheaper tamper-resistant sensor nodes. In these nodes, we find that there exists a lifetime-secure memory fraction which stores the anterior part of the application executable binary code, namely “fingerprint”. We propose a key management protocol based on this secure fingerprint—FKM. In this protocol, any pair of nodes can build a secret key by combining two raw key elements randomly selected by both nodes from their fingerprints respectively. To further strengthen the security, we also present two multi-dimension grid key reinforcement schemes. To the best of our knowledge, this paper is the first attempt at the use of application executable binary code itself to develop a key management protocol. A thorough analysis shows that FKM supports higher security and superior operational properties while consuming less memory resource compared to the existing key establishment schemes.

Keywords—security, wireless sensor network, fingerprint, key management scheme.

I. INTRODUCTION

Wireless sensor networks (WSNs) are quickly gaining popularity due to the fact that they are potentially low-cost solutions to a variety of real-world challenges. Security is essential for those mission-critical applications which may be deployed in adverse and unattended environments. As a result, one great challenge is how to establish secret keys between communicating nodes. This has been the issue of key management in WSNs.

Many schemes have been proposed for key management in WSNs, most of which are based on symmetric keys on account of resource constraints of sensor nodes. The basic idea is that a small number of secret keys or key sources are predistributed to nodes before their deployment. Nearly all existing schemes assume that a node can easily be compromised and all the information in that node can consequently be obtained by an adversary after being captured. For this reason, most of these schemes exhibit a threshold behavior [13]: The system can be considered secure when the number of compromised nodes is less than the threshold. However, these schemes fail to effectively defend against active attacks such as Sybil attack, replication attack and Denial of Service (DoS) attack even if the number of compromised nodes is less than the threshold [1]. What’s more, after capturing a node, an adversary can pay a moderate cost to capture the nodes around the captured one based on techniques such as analyzing received signal strength indicator (RSSI) [2]. Once the number of captured nodes has exceeded the threshold which is only a small fraction of a WSN generally consisting of hundreds of thousands of nodes, the adversary can break down the secure design and launch any attack. Therefore, it is insufficient to merely exploit the difficulty of capturing threshold number of nodes to achieve secure communications [13]. To guarantee the security of WSNs, one of the feasible solutions is to accomplish two critical tasks: to improve the security of the node itself and to develop a corresponding secure key management protocol.

In recent years, SoC technology has been developed to design smaller, lower-power and cheaper tamper-resistant sensor nodes [3, 4]. In these sensor nodes, we find that there exists a lifetime-secure memory fraction which stores the anterior part of the application executable binary code which we term fingerprint. The fingerprint can be overwritten but never be captured by the attackers (see Section II). All sensor nodes in a network hold the same fingerprint. Leveraging the existence of secure fingerprint in each node, we propose FKM, an efficient fingerprint-based key management protocol. In this scheme, we employ a concept of multi-dimension grid key space, in which each coordinate stands for a secret key for node-to-node communication. Any pair of nodes can build a secret key by combining two raw key elements which are randomly selected by both nodes from their fingerprints respectively. The key renewal and probability-based authentication properties of FKM ensure that it can defeat any known attacks with a high probability.

The major contributions in this paper consist of the following aspects:

- We identify the existence of lifetime-secure fingerprint in SoC-based sensor nodes, and exploit it as raw key source. To the best of our knowledge, this is the first attempt at using application executable binary code itself for developing a key management protocol.
- We develop a key management protocol based on the secure fingerprint to enhance the time and cost penalty imposed to the attacker by a huge key space.
- Thorough security and efficiency analyses show that FKM supports higher security while consuming less resource compared with the existing key predistribution schemes.

The remainder of this paper is organized as follows. Section II describes the foundation of fingerprint in sensor nodes. Section III presents our proposed FKM in details. The performance of FKM is evaluated in Section IV. Finally, we discuss related works in Section V, and make a conclusion in Section VI.

II. WHAT IS FINGERPRINT?

This section introduces the fingerprint in SoC-based tamper-resistant sensor node and explains why it is secure enough to be used as the foundation for FKM.

1 In the following of this paper, sensor nodes refer to the SoC-based lifetime-secure nodes designed with tamper-resistant technologies introduced in Section II.
As shown in Fig. 1, the fingerprint in a SoC-based tamper-resistant sensor node is the anterior part of the application executable binary code. All nodes in a WSN which are installed with the same application hold an identical fingerprint which can be considered secure since its content can be kept confidential for the lifetime of the WSN. This can be proved by analyzing its security against two possible kinds of attacks: physical and side-channel attacks, and logical attacks [4].

First, we introduce the practical tamper-resistant sensor node which is lifetime-secure against physical and side-channel attacks. Many low-cost countermeasures (e.g., restricted program counter, randomized clock signal, top-layer sensor meshes, robust low-frequency sensor, bus encryption, randomized multithreading, test circuitry abolition, etc.) have been proposed to effectively defend against a range of attacks (e.g., most of the non-invasive attacks such as glitch attacks, power analysis attacks, timing attacks, fault injection attacks, memory read-out attacks, and electromagnetic analysis attacks; the light-weighted invasive attacks such as manual micro-probing, data and address bus line observation, tampering involving electronbeam testers, etc.) [4, 5], which satisfy the general security requirements of most WSNs. Besides these, zeroization mechanism, which allows the nodes to zeroize all secrets whenever any unauthorized attempt at physical access is detected, is a cost-effective protection against sophisticated attacks [4]. It is particularly feasible in the large scale WSNs for their enough redundancy of information [6]. The redundancy of WSNs can be utilized to lower the detection threshold of the zeroization mechanism which will significantly reduce its cost and further enhance the time and cost penalty imposed to the attacker. Recently, SoC technology is increasingly being used to design practical secure sensor nodes [4]. On one hand, SoC-based nodes can defeat the non-invasive attacks inherently because of its design features [3]; on the other hand, it can be integrated with various countermeasures according to different level security requirements under the constraints of stringent design turn-around time and low cost. To sum up, SoC-based nodes are of adequate lifetime-security against physical and side-channel attacks.

Second, we discuss fingerprint’s security against logical attacks. For the constraint of low-cost, sensor nodes should be able to download new software to support new or updated applications. While this certainly increases the flexibility of WSNs, it poses a new security challenge in terms of the increased probability of logical attacks. An attacker can download a malicious executable binary code to a captured node, which reads the content of the memory and transmits it to the attacker by radio [6]. By this means, an attacker can compromise most of the content of nodes just by launching such logical attacks. Fortunately, the malicious code itself needs to occupy some memory space whose original content cannot be obtained by that malicious code, and the address checking circuit and substitution box circuit in a node guarantee that only the executable binary code downloaded to the continuous memory space which starts from a fixed address can be loaded and executed [5]. The program counter restriction circuit and bus monitoring hardware can be used to restrict the illegal jump statement and distinguish between the legal and illegal accesses to the sensitive information in the node [4]. Therefore, the anterior fraction of the application executable binary code—fingerprint overwritten by the minimum-sized malicious code can be considered secure against logical attacks, which is illustrated in Fig. 1.

In order to ensure that the attacker cannot compromise any part of the fingerprint in a WSN by exploiting the obtained fingerprint from another network, it is necessary that different WSNs possess different fingerprints. To achieve this, the design of the SoC-based tamper-resistant node adopts cryptographic confusion technology by integrating lightweight non-linear and non-correlated substitution box circuits [7]. After deliberately obfuscating the binary code by utilizing the confusion technology and programming the encrypted binary code into each node, the fingerprints in different WSNs are nearly irrelevent and sufficiently distinct even if the original binary codes in these systems are very similar [7]. When nodes start up, the substitution box circuit will decrypt the confused binary code and execute the original program. The fingerprint independence guarantees that the fingerprint is only applicable for its own WSN and the compromise of a fingerprint must be performed in its own WSN, hence, lifetime-security is sufficient for it.

From the above analysis we can conclude that the SoC-based tamper-resistant sensor nodes in a network always hold the same fingerprint which is the only secure part that survives not only the physical and side-channel attacks but also the logical attacks throughout the limited lifetime of a sensor node.

III. FINGERPRINT-BASED KEY MANAGEMENT SCHEME

A. Fingerprint-based Raw Key Source

The time-effective security of the key material in sensor nodes is sufficient because of the limited lifetime of WSNs. That is to say, a node can be considered secure if its key material cannot be retrieved by adversaries within a limited period. Before deployment, all the nodes in a network are preloaded with an identical application program. As shown in section II, the fingerprint, part of application program in a node, is lifetime-secure enough to be utilized as the raw key source. In FKM, We partition the fingerprint into \( n \) non-overlapping key elements with a specific length, each of which has a unique identifier \( S_i \) where \( i \in \{0, 1, \ldots, n-1\} \). The size of the fingerprint \( n \) is decided by the length of the executable binary code of a minimum-sized malicious program which depends on the hardware of the node. Thus, the length of the fingerprint is fixed for a certain kind of node. After extensive analysis on typical hardware platforms for nodes and on various TinyOS applications, we assume the minimum size of a malicious executable binary code is 1600 bytes. We set the length of a key element to 64 bits. Thus, there are 200 key elements in the fingerprint.

B. Basic Key Establishment

In FKM, two nodes can build a link session key with the key elements from their fingerprints. First, each node contributes a

![Fig. 1 Fingerprint exists in the memory space in a sensor node](image)
key element, and exchanges the identifier of the selected one. Next, both nodes apply a sequential one-way hash operation \( H \) (here we select RC5 which has been adopted in the implementation of many key management schemes [13]) on two selected key elements to compute a session key, which guarantees that attackers can hardly obtain the key elements just by conversely operating on their identifiers and the corresponding session key.

For an n-dimensional grid, each node can access an \( n \times n \) square grid key space, where each element corresponds to a symmetric key. For example, element \((i, j)\) in this grid corresponds to the key \( K_{ij} \) which is computed by hash operation \( H \) on the \( i \)th and \( j \)th key elements in the fingerprint,

\[
K_{ij} = H(S_i, S_j)
\]

(1)

where X-coordinate \( i \) and Y-coordinate \( j \) are selected by the source and destination side nodes of a secure link, respectively.

In the rest of the paper, the identifier of a link session key refers to its coordinate in the square grid key space.

A link session key is negotiated by both sides of a link with sequential one-way operation, which ensures that an attacker cannot dominate the link session key establishment since it is only one side of the link. We will further study this in Section IV.

C. Key Reinforcement

We propose two techniques to strengthen the security of communication links: multiple square-grids key reinforcement and path key reinforcement, which effectively increase the size of key space by using multi-dimension grid.

**Multiple Square-grids Key Reinforcement** One way to extend the key space is to use multiple square-grids. For each square-grid, each location in that grid is associated with a sub-key which can be computed similar to the basic scheme. When two nodes need to communicate, they exchange their selected part-coordinates in all grids (i.e. source node of the link proposes the X-coordinates and the expected key grid dimensions and destination node proposes the Y-coordinates and the acknowledged key grid dimensions), which is demonstrated by Fig.2 (a). After computing all the sub-keys, both nodes apply the hash operation \( H \) on these sub-keys to compute a session key. For \( i \) square-grids key reinforcement, the key space is \( n^{2i} \).

For example, if node \( A \) needs to establish a 3 square-grids key with node \( B \), firstly both nodes need exchange their selected part-coordinates (we suppose node \( A \) and node \( B \) propose \([X_1, X_2, X_3] \) and \([Y_1, Y_2, Y_3] \) as their part-coordinates respectively). Then each node can compute 3 square-grids key as follows:

\[
\text{Key}_{AB} = H(S_{X_1}, S_{Y'}, S_{Y''}) = H(H(H(H(S_{X_1}, S_{Y_1}), H(S_{X_2}, S_{Y_2})), H(S_{X_3}, S_{Y_3})), H(S_{X_1}, S_{Y_2}))
\]

(2)

**Path Key Reinforcement** We can also utilize path keys to achieve higher dimension key grids. When communicating with each other, two nodes first set up a route path and establish a path key simultaneously. Each node on the route path contributes to the generation of the path key. Supposing that node \( A \) needs to send message to node \( B \), the path created during initial route-setup is \( A, N_1, \ldots, N_i, B \). Each node selects an element of fingerprint \( S_{Ni} \) and exchanges its index with other nodes on the path, which is demonstrated by Fig.2 (b). When a node receives all the selected element indexes, it will apply the hash operation \( H \) on these elements to compute the \( i+1 \) hops path key:

\[
\text{Key}_{AB} = H(S_{X_1}, S_{Y'}, S_{Y''}) = H(H(H(H(S_{X_1}, S_{Y_1}), H(S_{X_2}, S_{Y_2})), H(S_{X_3}, S_{Y_3}) \ldots S_{Y_2}), S_{Y_3}) \ldots S_{Y_n})
\]

(3)

which can be used to secure all the link communications on that path. For \( i \)-hop path key reinforcement, the key space is \( n^{2i} \). Each node on a path participates in the generation of path key, which makes it harder for the attacker to launch attacks via captured keys. It provides end-to-end validity authentication and thus can be applied to critical communications such as the compromise information reporting during key revocation process.

D. Key Maintenance

1) Revocation

Whenever the compromise of the session keys is detected during the key renewal process, it is essential to revoke these keys from each node in the network. First, the finder node of the compromise builds a path to the sink node and establishes a path key. The finder node then encrypts the identifiers of the compromised keys with the established path key and delivers them to the sink node. Second, the sink node broadcasts a revocation message containing a signed list of identifiers for these compromised keys. To sign the list of key identifiers, the sink node generates a signature key with uTesla authentication [8]. Last, after obtaining the signature key, each node verifies the signature of the signed list of key identifiers, records these identifiers in the invalid key list, and removes them. After key revocation, some link session keys may disappear, and the affected nodes need to restart the key establishment process. In the later key establishment, node checks the invalid key list first to guarantee that the established session key has not been compromised.

2) Renewal

It is important for two communicating nodes to update their link session key periodically, which can shorten the durable attacking time for a compromised link. In FKM, the nodes can access the whole key space, which enables them to renew session keys. First, one node on the communicating link informs another node about the invalidation of the link session key. Then both nodes will remove the expired key from their session key lists and record it in their invalid key lists. After expired-key removal, the affected nodes restart the key establishment phase. Once a node detects that its correspondent has failed to encrypt packets with the renewed session key, it will induce that its correspondent and the session keys ever used for that link have been compromised, and then launches a revocation of the compromised link session keys. Key renewal is entirely localized which does not involve any network-wide broadcast message.

E. Discussion

FKM exploits the fingerprint as the raw key source. Besides the basic scheme, two reinforcement schemes are also designed to further enhance the time and cost penalty imposed to the attacker by a much larger multi-dimension grid key space. However, if a sophisticated and well funded adversary could, in a way, read the content of the memory which holds the finger-
print, then the security of the whole network would be compromised. Thus the security of the system asymptotically is equal to the secrecy of the fingerprint. To mitigate the dependence upon the secrecy of the fingerprint, FKM can easily exploit the clustering characteristic of large WSNs to combine the fingerprint and the inner-cluster private material to build session keys. The private material in each cluster can be a 64-bit key element selected randomly and broadcasted securely throughout the cluster with FKM session key by the clusterhead node during the setup of the clustered WSNs. Two nodes A and B in the same cluster can apply hash operation $H$ on their common private material and the computed link key to get the improved session key:

$$\text{ImpKey}_{AB} = H(\text{Key}_{AB}, PM_{Cluster})$$

where $\text{Key}_{AB}$ is the link key computed by Eqs.(1) and (2), $PM_{Cluster}$ is the private material of the cluster which $A$ and $B$ belong to. For the establishment of improved session path key, each node on the path can compute it with its private materials and the path key built by Eq.(3). There may be several improved session keys for a path since it falls into several clusters. The transition between two neighbored improved session keys on a path is performed by the gateway node which owns the private materials of two neighbored clusters. The improved session key achieves better security by mitigating the dependence on the secrecy of fingerprint while only costing additional 64-bit memory space and one hash operation compared with FKM.

IV. PERFORMANCE ANALYSIS

In this section, we analyze the performance of FKM by comparing its security and efficiency with several existing representative schemes for key establishment.

A. Evaluation Metrics

- **Resilience against passive attacks** The fraction of total network communications that can be compromised by a capture of $x$ nodes.
- **Resilience against active attacks** The probability that the adversary can insert additional hostile nodes into the network after obtaining some secret credentials.
- **Efficiency** The storage overhead, computation overhead and communication load for each sensor node.

B. Security Analysis

We evaluate FKM in terms of its resilience against node capture and analyze its security compared with three representative schemes: the random key predistribution scheme [9], multi-matrix based key management scheme [10] and deterministic key grids scheme [11]. In the simulations, the network consists of 10,000 nodes with an average density of 50 neighbors per node. In random key pre-distribution scheme, the size of link session key ring in each node is 200, and the size of key pool is 100,000. In multi-matrix based scheme, we select 2 distinct key spaces from 7 key spaces for each node (i.e., $\omega = 7, \tau = 2$). In FKM, the number of key elements in the fingerprint $n$ is 200, the grid dimensions $i$ of the basic scheme, 3-hop path scheme and 2-square grids scheme are 2, 3, and 4 respectively.

From section II we infer that attackers can hardly obtain the fingerprint in a tamper-resistant node within the lifetime of WSN. After capturing a node, an attacker can only compromise the link session keys in use within its neighborhood by eavesdropping and analyzing the ciphertext or launching logical attacks to the private memory of that node. The number of compromised link session key from each captured node $\beta$ equals to the node degree of that captured node. While for other key management schemes, an attacker can compromise all the secret credentials in a captured node even if it is tamper-resistant.

1) Resilience Against Passive Attacks

The security of fingerprint and larger key space contribute FKM favorable performance against passive attacks. Provided that the number of key elements in fingerprint is $n$, the dimension of grid key space in FKM is $i$ and the number of compromised session keys obtained from each captured node is $\beta$, the probability that any existing link $C_{ij}$ between two uncompromised nodes is compromised when $x$ nodes have been captured is:

$$\Pr(C_{ij} \text{ is compromised } | x) = 1 - (1 - \beta h')^x$$

which also denotes the fraction of links that an adversary can compromise by using the information retrieved from $x$ captured nodes. Fig. 3 clearly demonstrates that the FKM’s resilience against passive attacks rises as the decline in the node degree.

Fig. 4 shows the security of various schemes against passive attacks. In deterministic one-grid scheme and multi-matrix based scheme, the adversary will break down the secure design after capturing more than the threshold number of nodes. Both FKM and random key scheme achieve graceful performance degradation to an increasing number of compromised nodes. Specifically, in the basic FKM, the adversary needs to capture 100 more nodes than the random key scheme to compromise half the links. In two FKM reinforcement schemes, the adversary can only compromise less than 1% of the links after capturing 400 nodes.

**Renewal Property** After capturing some link session keys, an attacker should collect the ID information of all the session keys being used throughout the network in order to launch attacks to these links using captured keys. For most of existing schemes [9-13], an attacker needs to do collection only once. However, because of dynamic key renewal in FKM, the attacker needs to collect key ID information periodically in order to launch durable DoS attacks (e.g., packet injection attack, sink/ black hole attack), which may aggravate the time and cost penalty on attackers, shorten the successfully attacking duration.

If an attacker intends to launch a durable DoS attack on a specific link, it should hold all the session keys through the duration of the attack $t$. Provided that the key renewal interval is $k$, the number of renewal interval $r$ within duration $t$ is $\lfloor t/k \rfloor$. Given that the number of compromised session keys obtained from each captured node is $\beta$, the number of the compromised session key at the beginning of the DoS attack and at time $t$ are $\mu$ and $\mu_0$ respectively and the number of newly captured nodes within every renewal interval is $m$, the probability that the attacker launch a DoS attack of duration $t$ on a compromised link $C$ is:

$$\prod_{j=1}^{r} \Pr(C \text{ is compromised in the } j\text{th interval } | \mu_0)$$

$$= \prod_{j=1}^{r} \left[ \frac{\mu + (l - 1) \times m \times \beta}{n} \right] \times \left[ \frac{\mu + (l \times m \times \beta \times i) / k}{n} \right]^{m/2}$$

which clearly shows that FKM’s ability of resilience against durable DoS attack rises as the decline in the key renewal interval $k$. Fig. 5 shows the resilience of the basic FKM against durable QoS attacks with 500 compromised nodes. Most of the durations of durable DoS attacks are no more than 3 intervals.

2) Resilience Against Active Attacks

The passive attacks are only interested in collecting the sensitive data from the WSN, while the active attack goal is to dis-
The probability of the forged link being established is:

$$
\Pr(C \text{ is established } | \mu) = \frac{\mu_{\text{max}}}{n^{c_i}} \sum_{k=\mu_{\text{max}}}^{n^{c_i}} \frac{k \cdot \gamma^\mu}{i_c^2} n^{i_c}
$$

Here, $\gamma$ denotes a constant coefficient. Eq. (7) indicates that this probability has a negative correlation with the dimension of grid key space $i$ and an attacker can improve it by compromising more session keys. However, an attacker generally needs to compromise more than $n^c$ link session keys in order to increase $\mu_{\text{max}}$ by one, which makes the attacker harder to establish a forged session key. For the basic FKM, we suppose that attacker $A$ intends to launch an active attack to node $B$. First, $A$ needs to establish a session key with $B$. However, $A$ which only proposes the $X$-coordinate of the session key cannot ascertain what the session key is until $B$ proposes the $Y$-coordinate of the session key. Therefore, $B$ can verify $A$ with a high probability. What’s more, $B$ can forbid $A$ to use the same $X$-coordinate during the key renewal phase and revoke the compromised session keys. These measures enhance the resilience against active attacks.

Fig. 6 shows the security of various schemes under active compromise including Sybil attack, replication attack, and masquerade-based wormhole attack. In the graphs (Fig.6(a) and Fig.6(b)), the random key scheme has no resistance against either Sybil attack or replication attack. The multi-matrix based key management scheme can resist against Sybil attack but has no resistance against replication attack when the number of captured nodes is less than a threshold. Under both kinds of active attacks, FKM achieves graceful performance degradation to an increasing number of captured nodes. For the basic FKM, the attackers need to capture a great number of nodes in order to collect enough session keys located in a certain $X$-coordinate to launch active attacks successfully. For the FKM reinforcement schemes, the increasing dimension grid key space makes the attacker more difficult to collect enough proportion of keys located in a certain $X$-coordinate to launch active attacks.

FKM’s authentication property enables it to be well suited for combining with TrueLink to resist the masquerade-based wormhole attacks effectively [11]: the attacker $M$ needs to establish a forged link (link $C_{i,j}$ and link $C_{M,j}$) with each end-node of the wormhole (node $i$ and node $j$) simultaneously to build a tunnel link $W_{i,j}$. Given $\mu$ session keys are compromised, the probability of masquerade-based wormhole being established is:

$$
\Pr(W_{i,j} \text{ is established } | \mu) = \Pr(C_{i,j} \text{ is established } | \mu) \times \Pr(C_{M,j} \text{ is established } | \mu) = \frac{1}{n^{c-i_c} \gamma^\mu i_c^2}
$$

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the WCNC 2009 proceedings.
Eqs. (7) and (8) show that launching a wormhole attack becomes even much harder than launching one-forged-link active attacks. Fig. 6(c) demonstrates the security of various schemes against masquerade-based wormhole attacks.

C. Efficiency Analysis

Unlike the existing schemes whose memory is required to store both security credentials and session keys [6], the only additional memory required for facilitating FKM in a node is the storage of session keys which is directly proportional to the number of its correspondents. Thus, the memory cost for FKM is on the order of $O(1)$ which is much less than that of other key management schemes whose memory overhead is generally a function of the network size $n$ [9-13]. Moreover, all the nodes in a network hold the same fingerprint, which ensures that any two nodes are able to establish link session keys or path keys regardless of the node’s mobility, density or distribution.

To achieve a desirable level of security, each node needs to pay out relatively high computation and communication costs which are mainly due to the periodic session key renewal. During a period of time $t$, each node needs to compute $(\text{d}_{\text{link}} \times \lfloor t/k \rfloor)$ session keys, each of which carries out $i$ hash operations (see Eqs. (1), (2) and (3)), where $\text{d}_{\text{link}}$ is the node link degree in the communication spanning tree of the WSN which is much less than the node degree $d$ of the WSN (referred to the graph theory, $E(d_{\text{link}}) = 2$, $\text{Max}(d_{\text{link}}) = d$, $\text{Min}(d_{\text{link}}) = 1$). $k$ is the key renewal interval and $l$ is the dimensions of the grid key space. Therefore, at each key renewal period, the average and upper-bound computation overheads of each node are $2\times i$, and $d\times l$ hash operations respectively. During a period of time $t$, each node will incur $(\text{d}_{\text{link}} \times \lfloor t/k \rfloor \times l)$ messages, where $l$ is the length of the path key (both basic FKM and multiple square-grids link key reinforcement schemes can be considered as the specific path key reinforcement schemes with $l=1$). Therefore, at each key renewal period, the average and upper-bound communication costs of each node are $2\times l$, and $d\times l$ messages respectively.

V. RELATED WORKS

Recently proposed key predistribution schemes can be divided into two major categories: random key predistribution schemes and deterministic key predistribution schemes.

Many random key predistribution schemes have been proposed. Eschenauer and Gligor rely on probabilistic key sharing among nodes within WSN [9]. An enhancement utilizing multiple keys increases the security of key setup such that an attacker has to compromise much more nodes to achieve a high probability of compromising any given communication [12]. However, this kind of schemes only provide probabilistic security in the sense that the compromised nodes may expose other nodes’ keys with some possibility and cannot authenticate the availability of a node in a purely decentralized fashion.

The deterministic key predistribution schemes [10,11,13] usually employ complex mathematical methods or combinatorial design. Du et al. [10] use $r$-degree bivariate key matrices, in which two nodes can directly communicate only if they can identify at least one matrix in common. Camtepe and Yener [13] employ block design techniques in combinatorial design theory to decide how many and which keys to assign to each node. The grid-based idea proposed recently [11] arranges the secret keys based on logical multi-dimension grid to achieve minimum memory cost while guaranteeing secure communications. However for most of these schemes the probability of compromising the entire network will dramatically increase when the number of compromised nodes exceeds a certain threshold.

VI. CONCLUSIONS

In this paper, we have presented FKM, a novel key predistribution scheme for those SoC-based WSNs in which all the nodes share an identical secure memory fraction (we name it fingerprint) whose content cannot be accessed by the attackers throughout the lifetime of the network. This scheme utilizes the fingerprint to guarantee that any two nodes can directly compute and agree pairwise link session keys periodically with pretty low memory cost. It can support any size of network since the corresponding resource overhead does not rise with network size. Compared to the existing schemes, FKM is substantially more resilient against node capture and different kinds of attacks by trading off some node computation and communication cost.

ACKNOWLEDGMENTS

This work was partially supported by the State Key Basic Research Program of China “973 Project” (Grant No.2006CB303000), the National High-Tech Research and Development Program of China “863 Project” (Grant No.2007AA01229A), and the National Natural Science Foundation of China “NSFC” (Grant No.60533110).

REFERENCES