Singapore Management University
Institutional Knowledge at Singapore Management University

Research Collection School Of Information Systems
School of Information Systems

11-2011

Applying Time-Bound Hierarchical Key Assignment in Wireless Sensor Networks

Wentao ZHU
Chinese Academy of Sciences

Robert H. DENG
Singapore Management University, robertdeng@smu.edu.sg

Jianying ZHOU
Institute of InfoComm Research, Singapore

Feng BAO
Institute of InfoComm Research, Singapore

DOI: https://doi.org/10.1007/978-3-642-25243-3_25

Citation

ZHU, Wentao; DENG, Robert H.; ZHOU, Jianying; and BAO, Feng. Applying Time-Bound Hierarchical Key Assignment in Wireless Sensor Networks. (2011). *Information and Communications Security: 13th International Conference, ICICS 2011, Beijing, China, November 23-26: Proceedings*. 7043, 306-318. Research Collection School Of Information Systems.
Available at: https://ink.library.smu.edu.sg/sis_research/1420

Applying Time-Bound Hierarchical Key Assignment in Wireless Sensor Networks

Wen Tao Zhu[1], Robert H. Deng[2], Jianying Zhou[3], and Feng Bao[3]

[1] State Key Lab of Information Security,
Graduate University of Chinese Academy of Sciences,
19A Yuquan Road, Beijing 100049, China
wtzhu@ieee.org

[2] School of Information Systems,
Singapore Management University,
80 Stamford Road, Singapore 178902
robertdeng@smu.edu.sg

[3] Cryptography and Security Department,
Institute for Infocomm Research,
1 Fusionopolis Way, Singapore 138632
{jyzhou,baofeng}@i2r.a-star.edu.sg

Abstract. Access privileges in distributed systems can be effectively organized as a partial-order hierarchy that consists of distinct security classes, and are often designated with certain temporal restrictions. The time-bound hierarchical key assignment problem is to assign distinct cryptographic keys to distinct security classes according to their privileges so that users from a higher class can use their class key to derive the keys of lower classes, and these keys are time-variant with respect to sequentially allocated temporal units called time slots. In this paper, we explore applications of time-bound hierarchical key assignment in a wireless sensor network environment where there are a number of resource-constrained low-cost sensor nodes. We show time-bound hierarchical key assignment is a promising technique for addressing multiple aspects of sensor network security, such as data privacy protection and impact containment under node compromise. We also present the technical challenges and indicate future research directions.

1 Introduction

1.1 Hierarchical Key Assignment for Distributed Systems

With the rapid growth and pervasive deployment of information systems, sharing resources among multiple users over an open environment has become widespread. Access control on user permissions is an important issue in any system that manages distributed resources. In this paper, we consider a multilevel security scenario, where users and data of an information system are organized into a security hierarchy composed of m disjoint classes. A *hierarchical* key assignment (KA) is to assign a distinct cryptographic key to each class so that users attached

S. Qing et al. (Eds.): ICICS 2011, LNCS 7043, pp. 306–318, 2011.

to any "base" class can also derive the keys of "lower" classes. As confidential data are classified into such security classes, they can be protected with respective encryption keys using a symmetric cipher, where the decryption operation asks a user for the same encryption key so as to recover the data.

For ease of presentation, we have the classes partially ordered according to a binary relation "$\preceq$". They form a partial-order hierarchy $(C, \preceq)$, where $C_j \prec C_i$ means the clearance or security level of class C_j is lower than that of C_i, and $C_j \preceq C_i$ allows for the additional case of $j = i$. The hierarchical KA problem is to assign a key K_ℓ to each class C_ℓ, so that a user attached to her base class C_i can use the issued K_i to derive any K_j (thus to recover the data in C_j), if and only if $C_j \preceq C_i$. The hierarchy can be mapped to a directed acyclic graph, where each class corresponds to a vertex. A class may have multiple immediate ancestors. For example in Fig. 1, vertex C_7 has two immediate ancestors C_2 and C_4. If there is a top-level class with no ancestor, and each of the rest classes has exactly one immediate ancestor, the hierarchy representation then reduces to a rooted tree [1].

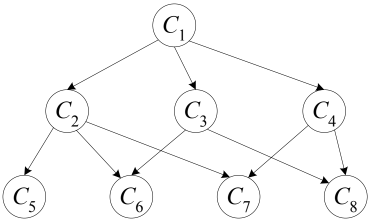

Fig. 1. A partial-order hierarchy $(C, \preceq)$ of $m = 8$ security classes. One class may have multiple immediate ancestors (e.g., $C_7 \prec C_2$ and $C_7 \prec C_4$). Although there is a top-level class C_1, this graph is not a rooted tree.

1.2 Time-Bound Hierarchical Key Assignment

In many applications such as electronic archive subscription, there is a temporal restriction so that a user is attached to her base class for only a limited period of time (typically the subscription period) consisting of a consecutive set of time units. Let the time dimension be discretized into even units (i.e., time slots or intervals) $t = 0, 1, \cdots, z$. Here the maximum index z should not be considered as a limitation of the access control policy, because the system lifetime can be arbitrarily large. For example, if each unit represents a minute, $z = 5.256 \times 10^6$ denotes 10 years. The *time-bound* hierarchical KA is to have the K_i of class C_i further mapped to a volatile key $k_{i,t}$, i.e., let the data categorized into class C_i at time t be encrypted with $k_{i,t} (1 \leq i \leq m, 0 \leq t \leq z)$. By specifying the following, we say a *class key* is instantiated with a series of *session keys*:

- The static K_i for C_i is only used for generating session keys $\{k_{i,t}\}$ as well as deriving the time-invariant class key K_j of any lower class $C_j \prec C_i$, but not used directly for data encryption.

– Only the time-variant $k_{i,t}$ is employed by the aforementioned symmetric cipher for actual data protection with respect to security class C_i, from session to session indexed by time t.

A typical application of time-bound hierarchical KA is the pay TV broadcasting, where a service provider organizes the channels into several subscription packages for users' choices. For example in Fig. 1 there are four independent TV channels C_5, C_6, C_7, and C_8, and subscription to package C_3 allows for the access to two of them (C_6 and C_8), while subscription to package C_1 allows for all. In such applications, a trusted central authority (CA) manages the key assignment. Upon registration, a user authorized to her base class C_i ($1 \leq i \leq m$) for period of time $[t_1 \cdots t_2]$ ($0 \leq t_1 \leq t_2 \leq z$) is assigned by the CA a *private primitive* denoted as $I(i, t_1, t_2)$. She should only be able to derive from $I(i, t_1, t_2)$ the session keys $\{k_{j,t}\}$ satisfying $C_j \preceq C_i$ and $t_1 \leq t \leq t_2$, thus only authorized to access the data stored in C_j at time t. The session key derivation is constrained by both the security hierarchy $(C, \preceq)$ and the time bounds (t_1 and t_2), and the derived $k_{j,t}$ should equal the instance of the class key K_j at time t. The CA is active only at user registration. After the issuance of the private primitive, no private channel exists between the CA and the user, i.e., the user should derive $k_{j,t}$ from only $I(i, t_1, t_2)$ and certain static public information, but with no interaction with the CA or any other user. Interested readers can refer to [2] for a comprehensive overview of time-bound hierarchical KA.

This work complements [2] by exploring prospect applications of time-bound hierarchical KA in wireless sensor networks (WSNs), though in the literature little work has been done to address this topic. The motivation stems from the following observation. Historically, the hierarchical KA technique was introduced to implement multilevel access control [3], which is concerned with the protection of classified data and their aggregation, dissemination, update control, etc., and thus can be connected to emerging distributed data acquisition systems like WSNs. We discuss certain application scenarios in WSNs, and show how time-bound hierarchical KA can be utilized with two case studies. The first leverages the hierarchical property for privacy protection, while the second leverages the time-bound property for enhanced security. We also present the technical challenges and indicate possible future research directions.

2 Applying Time-Bound Hierarchical Key Assignment in WSNs: General Considerations

A WSN consists of a number of sensor nodes, and is an efficient approach to delivering data from the real world to the digital world. Sensor nodes have stringent resource constraints in terms of communication, computation, storage, and energy. These limitations along with possibly harsh deployment scenarios lead to many critical security and privacy issues.

We envision time-bound hierarchical KA to be a promising technique to meet many security and privacy requirements in emerging wireless networks including

WSNs. Indeed, in some cases the clearance designated to (hence cryptographic information like $I(i, t_1, t_2)$ entitled to) a sensor node can be predetermined according to contextual information [4] after network deployment. The next section presents such a scenario. For most WSN applications, however, it is unlikely to obtain the context apriori; there may be no way to predetermine the nodes' clearances. A walkaround is to preload all sensor nodes with private primitives concerning the top-level class C_{top} (if any). After deployment, each node gathers the contextual information and decides its clearance C_i in the partial-order hierarchy $(C, \preceq)$; if $C_i \prec C_{top}$, the node "downgrades" its private primitive. This is expected for the majority of the nodes as few (sometimes only the base station) would remain in C_{top}, but to be done within a short time interval. Here we follow the assumption in [5] of introductory security at the early deployment stage: sensor nodes are manufactured to sustain possible break-in attacks at least for a short interval (say several seconds) when captured, and the time necessary for an adversary to compromise a sensor node is larger than the time needed for nodes to complete the key derivation. An example in the literature that can be accommodated by this framework is the location-based compromise-tolerant security mechanism for wireless sensor networks [6], the autonomous implementation of which preloads each node with the network master secret κ, from which the so called location-based key can be derived.

Practical time-bound hierarchical KA schemes typically adopt a decoupled structure [2] that can be formulated as $k_{i,t} = H(K_i, w_t)$, where H is a one-way hash function and w_t is the *instance secret* enabling time constraints. The above "downgrade" may then be done by deriving an appropriate base class key K_i from K_{top} and immediately erasing K_{top} for the sake of security. For example, in [7] an energy-efficient level-based hierarchical system for secure routing is proposed, where context-aware sensor nodes are self-organized into 4 levels after deployment. Although therein the self-organized hierarchy is for secure routing, a similar approach can be employed to constitute a multilevel security paradigm for hierarchical and session-oriented WSN applications like secure data aggregation, where time-bound hierarchical KA can be applied. The technique may also help with role-based and/or subscription-based applications.

Applying time-bound hierarchical KA to sensor networks is of particular interest due to the fact that, although there has been extensive research on cryptographic key management in WSNs [8–10], little work has been done to address such a particular topic. Some research efforts such as [4] took into consideration the hierarchical KA property, but the proposed scheme only considers time-invariant cryptographic keys, and thus does not represent a full fledged access control solution. Note that sensor nodes may employ ciphers with relatively short keys, and thus even simply updating the encryption keys periodically shall lead to much improved security. Other research such as [11] claimed a dynamic key derivation, but the paradigm is event-driven (by active revocation, in contrast to scheduled, spontaneous, and non-interactive key expiration), and thus is far from the perception (and benefits) of time-bound KA. Moreover, in [11] the active rekeying by the CA is based on the specious assumption that there exists a secure broadcast channel from the CA to all non-compromised sensor nodes, which actually drives the work [11] into a contradictive chicken-and-egg situation. Therefore, we envision that integrating time-bound hierarchical KA with certain WSN applications shall be an interesting and valuable research area. As a possible direction, one of the metrics that are appropriate for evaluating a security scheme for WSNs is assurance [8]; it is an ability to disseminate different information at different assurance levels to the end user. This is similar to the multilevel security paradigm, and time-bound hierarchical KA is a prospective approach.

Nevertheless, many pragmatic issues need to be considered. For example, usually sensor nodes are not made tamper-resistant due to cost concerns, but practical time-bound hierarchical KA schemes require tamper-resistance to thwart *collusion attacks* [2]. More research efforts are needed to address the problem. Of course, these KA schemes can be adopted once technical development has made it possible for massive production of tamper-resistant sensor nodes; actually, such tamper-resistance is already assumed in recent publications like [4, 12]. Even for the current generation of WSNs, it is possible that the network is heterogeneous and composed of a mix of sensor nodes with different capabilities, some of which are tamper-resistant. It would be interesting to apply time-bound hierarchical KA to such a heterogeneous WSN while minimizing the impacts by possible collusion attacks. Last, even if tamper-resistant sensor nodes are not available currently, it is still feasible to employ time-bound hierarchical KA for sensor networks. Next we present such a case study.

3 Case Study One: Protecting Data Privacy in Body Sensor Networks

Many WSN applications such as health-care and automotive ones need access control to sensed data; otherwise, attackers may easily jeopardize user privacy (e.g., in medical solutions, or in vehicular and urban sensing networks). We take the former for a case study, as medical solutions are considered as one of the two application fields where WSN security and privacy are of most importance [9] (with the other being military solutions like battlefield surveillance). In hospitals (or at home), future e-Health systems known as body sensor networks (BSNs) will consist of low-power on-body wireless sensors attached to mobile users that interact with an ubiquitous computing environment to monitor the health and well-being of patients [13]. Whilst the sensors for e-Health are a reality today, the configuration and management of the multiple sensors and software components still require considerable technical computing expertise [13]. Since physiological data collected from BSNs are legally required to be kept private, any implementation should take the trouble to protect patients' privacy [9, 14]. Traditional access control policies may be difficult to implement as sensor nodes have limited capabilities for the evaluation of complex access control rules.

A BSN is an attended network composed of on-body sensors, which are hardly subject to physical capture. Unlike a military WSN, the environment for a BSN is far from adversarial, and the major concern lies in privacy rather than security against attacks. Assume it is legally required that the sensed physiological data of each patient (referred to as a *record* hereinafter) be encrypted with a distinct cryptographic key, and a patient's record should only be accessed by his or her attending doctor and the doctor's direct or indirect superiors. The policy can be enforced by encrypting the record of any patient before it is transmitted to a central database, and this can be done by associating each patient with an identifier, and associating his or her sensors with a corresponding encryption key. If a patient once discharged from the hospital is hospitalized again, he or she should be regarded as a new case and associated with a new identifier (and thus a new key), as he or she may neither have the same illness nor be treated by the same attending doctor.

The above paradigm fits into the familiar problem of multilevel security, and at first glance may be tackled with any hierarchical KA scheme. It may seem comparable to the access control in a corporation (or government department), where data are usually classified into only a few classes, say "unclassified" $\prec$ "confidential" $\prec$ "secret" $\prec$ "top-secret". The corporation example is relatively simple, as there are merely four classes, and all the data of the same clearance are protected with the same class key. In the considered BSN, however, the record of each case should be protected with a *distinct* encryption key, and a large hospital may accommodate thousands of patients. Instead of how to guard the BSN against attacks, the real challenge stems from how to efficiently organize the encryption keys to ensure privacy concerning a large and continuously growing number of cases. Derivatives of the exponentiation-based Akl-Taylor KA scheme [3], typically involving a 1-affects-n problem, may not apply to a BSN, since there are an overwhelming and growing number of classes, and thus the exponentiations may be extremely difficult to manage. On the other hand, modern schemes based on a reference table will involve heavy cost for public storage, and the maintenance of a voluminous reference table will be error-prone, as health practitioners may have little technical computing expertise [13]. We refer readers interested in exponentiation-based and reference-table-based KA schemes to [2] for technical details. Herein, a preferable solution is expected to be self-configuring and self-managing with little or no user input. That is, an effective system that works out of the box is desired.

Without loss of generality, we assume the following partial-order hierarchy in the e-Health system. Patients monitored by the BSN are directly taken care of (i.e., treated) by attending doctors, who are mostly interns and sometimes residents. An intern is supervised by a resident, who is in turn supervised by a physician. At the top of the medical hierarchy is the senior physician who supervises the physicians. Regarding the access rights to a patient's record, the hierarchy is "intern" $\prec$ "resident" $\prec$ "physician" $\prec$ "senior physician". There may be quite a few senior physicians in the hospital, in charge of different (i.e., non-overlapping) departments respectively. As usually one patient is attended by only one doctor (an intern or a resident), and any doctor (an intern, a resident, or a physician) is only directly supervised by one superior (a resident, a physician, or a senior physician, respectively), a favorable KA solution to the medical hierarchy could be based on a tree-like structure (recall Section 1.1), where Sandhu's KA scheme [1] can be employed.

In this scenario, the physiological data collected by sensors attached to a patient are encrypted and then sent to the central database, involving the partial-order hierarchy illustrated in Fig. 2. Assume the senior physician in a medical department is assigned a key K_{sp}, and there is a pseudo-random function F for key derivation, which maps a secret key k of a specified length and a binary string x of arbitrary length to $F_k(x)$ of the same length with k. Then the senior physician can derive the key of a subordinate physician by $K_p = F_{K_{sp}}(ID_p)$, where ID_p is the identifier of the physician. Note that in the KA scheme, only K_{sp} can be chosen randomly (known as "information-theoretic"), while other keys like K_p are derived with F (known as "computational"). Similarly, a physician can derive the key of a subordinate resident by $K_r = F_{K_p}(ID_r)$, where ID_r is the identifier of the resident. Next, a resident can derive the access key of a subordinate intern by $K_i = F_{K_r}(ID_i)$ where ID_i is the identifier of the intern, and the encryption key of the sensors attached to a patient attended by the resident (if any) according to $K_{c(r)} = F_{K_r}(ID_c)$ where ID_c is the case identifier of the patient. Last, an intern can derive the encryption key of the sensors attached to an attended patient with the case identifier ID_c following $K_{c(i)} = F_{K_i}(ID_c)$.

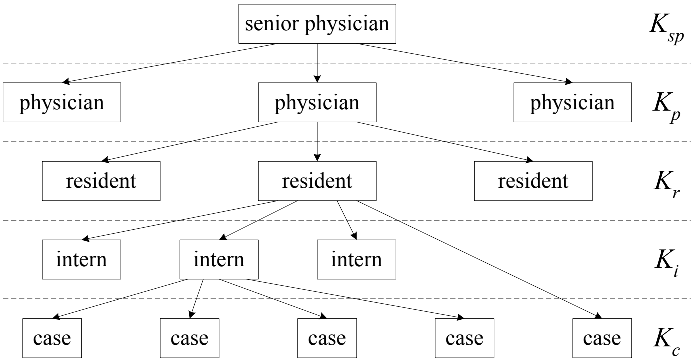

Fig. 2. The proposed key assignment for the medical hierarchy in a certain hospital department, where physiological data collected from a body sensor network (BSN) are encrypted before being sent to a central database

The above scheme is depicted in Fig. 2. Each doctor in the hierarchy is assigned a smart device, where his or her access key (typically computed from the superior's key) is embedded. The key derivation can be easily implemented with the device which only takes as input certain identifiers. Once a patient is hospitalized, he or she is attached with sensors preloaded with the corresponding encryption key K_c (either $K_{c(r)}$ or $K_{c(i)}$). Importantly, the addition of a new case, hence the addition of a new K_c (or other keys like K_i), does not affect any existent keys. This is an advantage over the Akl-Taylor style hierarchical KA [2], where adding a new class tends to be painful. When a doctor needs to access his or her patient's record, he or she inputs the patient's case identifier ID_c (e.g., via RFID) and the device derives the access key K_c; if the doctor is not the patient's attending doctor, additional identifier(s) of his or her corresponding subordinate(s) can be input manually or automatically.

This paradigm can be improved to have better privacy protection by instantiating all the class keys with time-bound session keys. It can be done by employing the afore-mentioned structure $k_{i,t} = H(K_i, w_t)$, where the time-variant instance secret w_t may be computed from dual hash chains [2]. For a doctor, the access period $t_1 \leq t \leq t_2$ for $k_{i,t}$ can be constrained according to his or her tenure (i.e., term of service in the hospital). For a sensor, irrespective of how long the attaching patient may be hospitalized, the time bounds can simply be specified according to the sensor's lifetime estimated from its battery sustainment. In a BSN, the smart device of a doctor or the sensors attached to a patient do not necessarily need to be tamper-resistant. This is due to the attended (also legally protected) nature of BSN, and health practitioners may not have the commercial incentive to tamper with the time-bound hierarchical KA system. Last, note the one-key-per-record policy already contributes significantly to security and privacy, while the adopted KA system by no means hinders healthcare workers from on-the-spot checks of an patient in case of emergency.

4 Case Study Two: Enhancing Multicast Security

Key management is an important aspect of WSN security [8–10]. It is recently understood that key management schemes offering group or multicast abilities are much more compatible with industry trends; based on the new tendency in IEEE 802.15.4b and the ZigBee Enhanced standard, it is envisioned that a purely random or pairwise key management scheme would be economically unviable [10]. Unfortunately, existent multicast key management schemes (also called rekey schemes), be they stateful (represented by [15]) or stateless (represented by [16]), are shown [17] to be seriously challenged by the threats in a WSN: an outside active adversary who compromises a single node could obtain not only the current multicast key but also some or all past keys, as well as future keys if detection and revocation are not promptly taken. Particularly, detection of node compromise is a nontrivial task. Therefore, even the compromise of a single node at an arbitrary time may jeopardize the *entire* multicast communication. This is highly counterintuitive: the impact of just a single node compromise may not be even worse, as it already reaches the worst case. In-depth analysis on rekey security is beyond the scope of this paper but is referred to [15–17].

We aim at improving multicast security based on purely time-bound KA, i.e., the security setting considered in Section I-B reduces to only one universal class comprising all sensor nodes. Hence, instead of following traditional and mainstream rekey schemes [15, 16] driven by group membership changes, we secure multicast traffic with time-dependent session keys, thus featuring scheduled, spontaneous, and non-interactive key expiration. In a dynamic WSN where new nodes are added while old nodes perish, different nodes have different life cycles. Consider a certain node entitled to the multicast session key $k(t)$ (simplified from the previous form of $k_{i,t}$, as the partial-order hierarchy is absent) for $t \in [t_1 \cdots t_2] \subset [0 \cdots z]$, where the lower and upper time bounds can be determined based on its scheduled deployment time (right before t_1) and the estimated end of battery life (t_2 at the most). The node is only preloaded with the private primitive $I(t_1, t_2)$. As a result, even if it is later captured, the attacker cannot gain more session keys beyond the pre-specified time scope $[t_1 \cdots t_2]$. Furthermore, if the adopted algorithm has the nice feature of memory deallocation to timely erase used cryptographic materials, the attacker can barely reveal secret keys between t_1 and the time she compromises the node.

Considering the implementation constraints for low-cost sensor nodes, an affordable time-bound multicast KA scheme should be as cost-efficient as possible. In existent schemes [2], an instance secret w_t based on an algebraic tool known as the Lucas sequence seems competent for a time-bound session key; it favorably impedes collusion attacks even in the absence of tamper-resistant hardware. However, the Lucas sequence computation is prohibitive for sensor nodes of low processing profile. On the other hand, utilizing the computation-efficient technique of the afore-mentioned dual hash chains to generate multicast session keys requires nodes be protected by tamper-resistant casing; otherwise, the compromise of a node with $I(t_1, t_2)$ and another node with $I(t_3, t_4)$ is equivalent to the compromise of every session key $k(t)$ for $t_1 \leq t \leq t_4$, where t_2 can be far less than t_3. The tamper-resistant prerequisite is cost-expensive or even unrealistic for the current generation of sensor nodes. (We notice, however, dual hash chains are exactly adopted in [12].) In a nutshell, a sensible solution that neither incurs heavy computational overhead nor asks for tamper-resistant protection is needed. Our scheme proposed below is a variant of Briscoe's MARKS [18].

Assume there is a one-way hash function H whose output is of size $2|k(t)|$. For example, if the standard cipher AES-128 is employed for encrypting the multicast traffic, a good choice for H is SHA-256. For brevity, assume the maximum time index is $z = 2^h - 1$, where h is an integer. Then we can build a virtual binary "tree of computational secrets" of height h employing $H(\cdot) = H_L(\cdot) \| H_R(\cdot)$, where H_L and H_R are the left and right halves of H, respectively (thus $|H_L| = |H_R| = |k(t)|$). The 2^h leaf nodes of the binary tree are associated with the $z + 1$ session keys for securing the multicast. As depicted in Fig. 3, the CA randomly selects from a key space the seed secret s_1 for the root node, and applies H for z times, to each secret s_i ($1 \leq i \leq z$) in the tree respectively. Hence every secret s_i for $2 \leq i \leq 2z + 1$ is computational. Then the CA assigns each leaf node secret to a session key according to $k(t) = s_{t+z+1}$ for $0 \leq t \leq z$.

Upon registration, a sensor node entitled to time period $[t_1 \cdots t_2]$ is preloaded with the private primitive $I(t_1, t_2)$ of size $\mathcal{O}(\log z)$. The private primitive consists of all and only the secrets closest to the root node in the tree that exactly enable computation of the authorized range of session keys. In [18], Briscoe presented a very simple but efficient algorithm for identifying such a minimum set of secrets. For example in Fig. 3, a node entitled to $k(t)$ for $1 \leq t \leq 8$ is assigned with

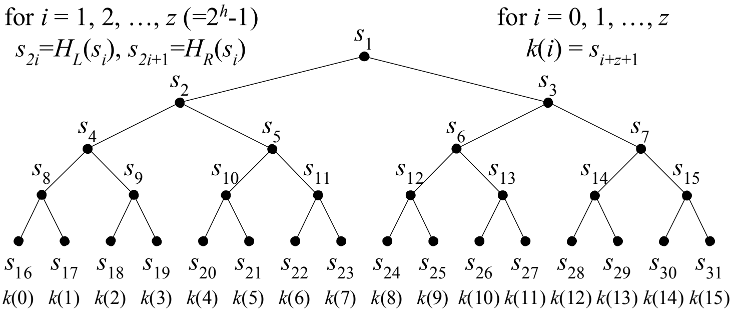
for $i = 1, 2, \ldots, z$ (=2^h-1)
$s_{2i}=H_L(s_i)$, $s_{2i+1}=H_R(s_i)$
for $i = 0, 1, \ldots, z$
$k(i) = s_{i+z+1}$
s_1
s_2
s_3
s_4
s_5
s_6
s_7
s_8
s_9
s_{10}
s_{11}
s_{12}
s_{13}
s_{14}
s_{15}
s_{16} s_{17} s_{18} s_{19} s_{20} s_{21} s_{22} s_{23} s_{24} s_{25} s_{26} s_{27} s_{28} s_{29} s_{30} s_{31}
$k(0)$ $k(1)$ $k(2)$ $k(3)$ $k(4)$ $k(5)$ $k(6)$ $k(7)$ $k(8)$ $k(9)$ $k(10)$ $k(11)$ $k(12)$ $k(13)$ $k(14)$ $k(15)$

Fig. 3. A "tree of computational secrets" of height $h = 4$ generated by a one-way hash function H covering a system lifetime starting from 0 and ending at $z = 2^h - 1 = 15$. H_L and H_R are the left and right halves of H, respectively.

$I(1, 8) = \{s_{17}, s_9, s_5, s_{24}\}$. The leaf node secrets s_{17} and s_{24} are directly mapped to $k(1)$ and $k(8)$, respectively. By applying H to the seed secret s_9, the sensor node can derive $k(2)$ and $k(3)$. By applying H three times, the sensor node can derive $k(4)$ to $k(7)$ from s_5. A sensor node only needs to derive a session key $k(t)$ when necessary (usually right before time t). Implementation details for the multicast client (herein the sensor node) like smart memory deallocation and storage/processing tradeoff can be found in [18].

One may be concerned with the computation load for the CA to generate the entire tree. Actually, the proposed scheme differs slightly from MARKS in the way the tree is generated. Instead of employing any one-to-one (typically the rotary) function as in [18], we choose a "size-doubling" one-way hash function $H(\cdot) = H_L(\cdot) \| H_R(\cdot)$, where no covert channel (as concerned in [18]) should arise. This facilitates the tree generation, which can be efficiently implemented with a standard programming language. For example, even if each time unit represents only one minute (hence the sensor nodes only need loose time synchronization) and the network lifetime is as long as 16 years, it is enough to specify $h = 23$. We tested the scheme with C on an ordinary Lenovo ThinkPad T61 laptop powered by the free download edition of Mandriva Linux 2009.1 i586. In our experiment, assuming each $k(t), 0 \leq t \leq z$ is 128-bit, generation of the full binary tree invoked $z = 2^h - 1 = 8,388,607$ times of the one-way hash function SHA-256, but the overall running time turned out to be only 10.1 seconds.

Compared with mainstream multicast key management schemes [15, 16], adopting a time-bound approach has the distinct benefit of no interaction between a user and the CA (recall Section 1.2), completely avoiding rekey communication overhead. This also implies no dependence on a reliable multicast channel for rekeying, as well as intrinsic immunity to eavesdropping. In the context of a WSN, the proposed scheme involving only efficient processing but no tamper-resistant requirement lends itself to improved multicast security, as the capture at time t_c of a node preloaded with $I(t_1, t_2)$ may not affect multicast communications either before t_c or after t_2. Besides memory deallocation for timely erasing used secrets, if upon detection of low power the sensor nodes have the intelligence to erase unused secrets, additional security against key exposure can be achieved. In a nutshell, even if compromise of low-cost sensor nodes may be unavoidable, time-bound multicast KA translates into impact containment. The philosophy is not to eliminate attacks (which is unrealistic), but to minimize the breakage.

Finally, the above scheme can be extended to a more general case, where a sensor node is authorized to an *arbitrary* set of time slots. Let T be a random combination of any t non-overlapping periods of time: $\mathsf{T} = \cup_{i=1}^{\mathsf{t}}[t_{i_1} \cdots t_{i_2}]$, where $t_{i_1} \leq t_{i_2}$ for $1 \leq i \leq \mathsf{t}$, and $t_{i_2} < t_{(i+1)_1}$ for $1 \leq i \leq \mathsf{t} - 1$. (The afore-discussed $I(t_1, t_2)$ is just a special case where $\mathsf{t} = 1$, i.e., T is consecutive.) Although as a whole T appears intermittent, a sensor node still only needs to be preloaded with the private primitive $I(\mathsf{T})$ in order to access exactly any $k(t)$ for $t \in \mathsf{T} \subseteq [0 \cdots z]$. Such a scenario regarding an arbitrarily specified set of time slots caters to certain WSN applications. For example, sensor nodes may be deployed for monitoring the tourism traffic and/or ecological environment at a local scenery spot, but not in off-season. That is, to reduce the maintenance expense, the nodes are scheduled for hibernation in off-season, and the monitoring is only in operation during peak season. Another example is school campus monitoring, which is unnecessary in summer and winter vacations. These applications necessitate time-based KA concerning an arbitrary (i.e., intermittent) set of time slots.

The data structure in Fig. 3 can still accommodate the generation of session keys even with respect to an arbitrary set of time slots T. For example, assume $\mathsf{t} = 3$ and $\mathsf{T} = \{2, 3, 8 \cdots 11, 14\}$. A sensor node only needs to be preloaded with $I(\mathsf{T}) = \{s_9, s_6, s_{30}\}$ in Fig. 3 to derive $k(2)$, $k(3)$, $k(8)$ to $k(11)$, and $k(14)$. This bears much similarity to the broadcast encryption problem following the Subset-Cover framework, particularly, the Complete Subtree Method [16], though both the scenario (revocation of stateless receivers for digital content protection) and the meaning of the employed data structure (a tree of information-theoretic keys shared by a large number of potential receivers) there are completely different. Following the results in [16], the size of the private primitive $I(\mathsf{T})$ assigned to a sensor node is increased from $\mathcal{O}(\log z)$ to $\mathcal{O}(r \log \frac{z}{r})$, where $r = |[0 \cdots z] \setminus \mathsf{T}| = z + 1 - |\mathsf{T}|$. In case t in T is small, $|I(\mathsf{T})|$ simply reduces to $\mathcal{O}(\mathsf{t} \log z)$. Therefore, adopting the tree depicted in Fig. 3 is promising for multicast KA in sensor networks, even if the upper application has an unusual demand for intermittent monitoring.

5 Conclusion

In this paper we addressed the problem of time-bound hierarchical key assignment, and explored possible applications of the technique in the context of wireless sensor network security. Due to cost considerations, some existent time-bound hierarchical KA schemes may not be readily applicable to the current generation of sensor nodes, which may neither afford heavy computations nor have tamper-resistant casing. More research efforts are needed to develop efficient and practical KA solutions. Nevertheless, the technique is still feasible and preferable in a few typical applications. For example, time-bound hierarchical KA is valuable for application scenarios where the contextual information is available for sensor nodes. The technique can be adopted in future e-Health systems for protecting patients' privacy. A related topic, post-deployment clearance evaluation based on context acquisition, may be a prospective direction for future research. For another example, sensor network key management schemes that offer multicast abilities are more compatible with industry trends, and time-bound KA is an approach to improving multicast security with breakage alleviation. Even if the group oriented upper application requires intermittent monitoring, the technique is still promising for impact containment under node compromise.

Acknowledgement. This work was supported by the Singapore A$*$STAR project SEDS-0721330047 and by the National Natural Science Foundation of China under Grant 60970138.

References

1. Sandhu, R.S.: Cryptographic implementation of a tree hierarchy for access control. Information Processing Letters 27, 95–98 (1988)
2. Zhu, W.T., Deng, R.H., Zhou, J., Bao, F.: Time-bound hierarchical key assignment: An overview. IEICE Transactions on Information and Systems E93-D, 1044–1052 (2010)
3. Akl, S.G., Taylor, P.D.: Cryptographic solution to a problem of access control in a hierarchy. ACM Transactions on Computer Systems 1, 239–248 (1983)
4. Shehab, M., Bertino, E., Ghafoor, A.: Efficient hierarchical key generation and key diffusion for sensor networks. In: Proc. 2nd Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks (SECON 2005), pp. 76–84 (2005)
5. Zhu, S., Setia, S., Jajodia, S.: LEAP+: Efficient security mechanisms for large-scale distributed sensor networks. ACM Transactions on Sensor Networks 2, 500–528 (2006)
6. Zhang, Y., Liu, W., Lou, W., Fang, Y.: Location-based compromise-tolerant security mechanisms for wireless sensor networks. IEEE Journal on Selected Areas in Communications 24, 247–260 (2006)
7. Tubaishat, M., Yin, J., Panja, B., Madria, S.: A secure hierarchical model for sensor network. ACM SIGMOD Record 33, 7–13 (2004)
8. Chen, X., Makki, K., Yen, K., Pissinou, N.: Sensor network security: A survey. IEEE Communications Surveys & Tutorials 11, 52–73 (2009)
9. Li, Z., Gong, G.: A survey on security in wireless sensor networks. Technical Report, University of Waterloo, CACR 2008-20 (October 2008)
10. Lee, J.C., Leung, V., Wong, K.H., Cao, J., Chan, H.: Key management issues in wireless sensor networks: Current proposals and future developments. IEEE Wireless Communications 14, 76–84 (2007)
11. Sorniotti, A., Molva, R., Gomez, L.: Efficient access control for wireless sensor data. In: Proc. 19th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC 2008), pp. 1–5 (2008)

12. Jiang, Y., Lin, C., Shi, M., Shen, X.: Self-healing group key distribution with time-limited node revocation for wireless sensor networks. Ad Hoc Networks 5, 14–23 (2007)
13. Lupu, E., Dulay, N., Sloman, M., Sventek, J., Heeps, S., Strowes, S., Twidle, K., Keoh, S.-L., Schaeffer-Filho, A.: AMUSE: autonomic management of ubiquitous e-Health systems. Concurrency and Computation: Practice and Experience 20, 277–295 (2008)
14. Leavitt, N.: Researchers fight to keep implanted medical devices safe from hackers. Computer 43, 11–14 (2010)
15. Wong, C.K., Gouda, M., Lam, S.S.: Secure group communications using key graphs. IEEE/ACM Transactions on Networking 8, 16–30 (2000)
16. Naor, D., Naor, M., Lotspiech, J.: Revocation and Tracing Schemes for Stateless Receivers. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 41–62. Springer, Heidelberg (2001)
17. Xu, S.: On the security of group communication schemes. Journal of Computer Security 15, 129–169 (2007)
18. Briscoe, B.: MARKS: Zero Side Effect Multicast Key Management Using Arbitrarily Revealed Key Sequences. In: Rizzo, L., Fdida, S. (eds.) NGC 1999. LNCS, vol. 1736, pp. 301–320. Springer, Heidelberg (1999)