On Secure Key Management in Mobile Ad Hoc Networks

Dahai Xu Jeffrey Dwoskin Jianwei Huang Mung Chiang Ruby Lee

***Abstract*—It is widely believed that although being more complex, a probabilistic key predistribution scheme is much more resilient against node capture than a deterministic one in lightweight wireless ad hoc networks. Backed up by the surprisingly large successful attack probabilities computed in this paper, we show that the probabilistic approaches have only limited performance advantages over deterministic approaches. We first consider a static network scenario as originally considered in the seminal paper by Eschenauer and Gligor [1], where any node capture happens after the establishment of all pairwise links, and show that the deterministic approach can achieve a performance as good as the probabilistic one. Whereas in a mobile network, the probabilistic key management as described in [1] can lead to a successful attack probability (SAP) of one order of magnitude larger than the one in a static network due to node fabrication attacks. Finally, we propose two low-cost secure-architecture-based techniques to improve the security against such attacks. Our new architectures, specifically targeted at the sensor-node platform, protect long-term keys using a root of trust embedded in the hardware System-on-a-Chip (SoC). This prevents an adversary from extracting these protected long-term keys from a captured node to fabricate new nodes. The extensive simulation results show that the proposed architecture can significantly decrease the SAP and increase the security level of key management for mobile ad hoc networks.**

I. Introduction

A. Motivation

Lightweight ad hoc networks typically consist of nodes that are highly distributed with very limited computation and energy resources. Examples include portable mobile devices and tiny low-cost sensors used for environment surveillance and emergency response. Providing secure communication over this kind of network is a challenging task. Various key management schemes have been proposed trying to provide a highly secure communication environment in lightweight ad hoc networks against various malicious attacks. Among key management schemes, *symmetric* key predistribution schemes (e.g., [1]–[5]) are more suitable to the light weight ad hoc network than *asymmetric* public-key schemes (e.g., [6], [7]), because the former schemes require less resource (e.g., battery, memory, computation power) and there is no need for a trusted third party for authorization.

There are two main approaches among the symmetric key predistribution schemes: *probabilistic* (e.g., [1], [5], [8]–[10]) and *deterministic* (e.g., [11]–[14]). In a probabilistic approach, the keys in each node's key ring are randomly chosen from a large key pool. In a deterministic approach, the key ring is chosen deterministically. In general, probabilistic approaches end up with a large key pool, a larger key ring per node, and poorer network connectivity than the deterministic approaches.[1] On the other hand, a typical deterministic algorithm preloads each node with a single common key and reaches connectivity of 100%. More related references can be found in the recent survey [15].

It is often believed that a typical probabilistic scheme is much more resilient against node capture than a typical deterministic approach [1], [8], [9], [16], thus making probabilistic schemes popular despite their clear disadvantage on many other metrics when compared with deterministic approaches. *In this paper, we show that the probabilistic approaches have only limited performance advantages over deterministic approaches*. Our performance measurement is the *Successful Attack Probability* (SAP). In particular, we consider an attack on a pairwise link between two authorized nodes to be successful if a compromised node can intercept and decipher the information transmitted through that link.

Parts of this paper were presented at IEEE ISIT and GLOBECOM 2007. This work was supported by DARPA and NSF Cybertrust CNS-0430487 and CNS-0636808.

D. Xu is with AT&T Labs - Research, 180 Park Ave, Building 103, Florham Park, NJ 07932, (email: dahaixu@research.att.com).

J. Dwoskin, M. Chiang and R. Lee are with Department of Electrical Engineering, Princeton University, NJ 08544, USA, (email: {jdwoskin, chiangm, rblee}@princeton.edu).

J. Huang is with Department of Information Engineering, Chinese University of Hong Kong, Hong Kong, (email: jwhuang@ie.cuhk.edu.hk).

[1]For example, the probabilistic scheme in [1] requires preloading each node with 83 keys out of a key pool size of $10,000$, and achieves a local direct connectivity of 50%.

B. Summary of our study between representative probabilistic and deterministic schemes

The probabilistic scheme was first proposed in the seminal and widely cited paper by Eschenauer and Gligor [1], and we call the corresponding scheme the *EG scheme*. It consists of three phases: *key distribution*, *shared key discovery* and *path-key establishment*. In the key distribution phase, each node is loaded with k keys randomly chosen from a large key pool of size m, where $k \ll m$. The shared key discovery is the procedure of establishing a pairwise link between two neighbor nodes if they share one or more key(s). Finally, in the path-key establishment phase, a pairwise link is established between any two neighbor nodes who do not share any key but can establish a path between them through one or more relay nodes. In this case, a path-key is sent from one node to its neighbor through the relay(s), and then a link is established similarly to the shared-key discovery phase.

A representative deterministic scheme uses only a single common key, and each node is preloaded with the same initial key. After the deployment, each pair of neighbor nodes exchange messages encrypted by the common initial key to derive a unique (or even random) key for all later communications between them.

Throughout the paper, we will compare the performance of probabilistic and deterministic key management schemes based on the EG scheme [1] and single common key scheme. *We will show that the probabilistic scheme is not significantly better than the deterministic scheme measured in terms of SAP.* We will also discuss in the Appendix how other probabilistic schemes based on the EG scheme do not change our conclusion. Since single common key is one of the simplest deterministic schemes, any further improvement over it (e.g. [14]) will only enhance our conclusion.

We will consider two network scenarios: *static network* and *mobile network*. In a static network, all pairwise links have been established before an adversary captures any node. This is the case previously considered in [1]. This could happen, for example, if all nodes are deployed almost at the same time and remain stationary after deployment. In contrast, in a mobile network, an adversary can capture a node before all pairwise links have been established. This is true for a network where nodes are constantly on the move and need to establish new links. This includes, for example, a sensor network of buoys floating freely on the ocean to gather environmental data [17], or a network consisting of sensors moving around in an unknown environment to form reasonable coverage [18].

In a static network, the single common key deterministic scheme can achieve almost perfect resiliency against node capture (i.e. SAP $\approx$ 0). This is because the initial common key can be deleted permanently from all nodes after the establishment of all pairwise keys (as in [11]). Since all pairwise keys are randomly generated and known only to the corresponding two neighbor nodes, they cannot be deduced by a captured node even if the common initial key is disclosed. In the EG scheme, however, the SAP equals k/m with only one captured node where each neighbor node pair uses one of the shared keys to encrypt the communication. It is possible to reduce the SAP to almost zero as in the single common key case if two neighbor nodes also generate a random key for future communication. In short, the deterministic scheme can achieve performance as good as the probabilistic approach in a static network, but with much lower complexity.

In a mobile network, the single common key deterministic scheme could lead to an SAP as high as 100% if the common initial key is obtained by an adversary before any link is established. However, we show that the EG algorithm is also quite vulnerable in this case, and may lead to a value of SAP one order of magnitude larger than the one in the static network case (e.g., as high as 60%), especially when the adversary can fully utilize the keys obtained from several compromised nodes. The intuition for the surprising result in this case is as follows. In the static network, there is only one way to attack a link successfully, i.e., knowing the key with which the communications on that link is encrypted. In a mobile network, however, a compromise node can also attack a link by acting as a relay during the path-key establishment phase. By intercepting the key information that is being relayed, a compromised node can figure out the key which the two authorized nodes will use for future mutual communication. This new *man-in-the-middle* attack opportunity can significantly increase the value of SAP for a probabilistic approach, since nodes frequently use a relay for link establishment.

After re-examining the performance difference between probabilistic and deterministic key predistribution schemes, we propose two secure-hardware-based techniques, specifically targeted to the sensor-node platform, that protect long-term keys for both deterministic and probabilistic key management schemes for mobile networks. This ensures that protected secrets cannot be extracted from a captured node. This is the first step towards building a comprehensive low-cost secure-hardware design for sensor nodes.

The contributions of this paper are:

- *SAP analysis*: An analysis of various security at-

tacks on the secure key management in mobile lightweight ad hoc networks, with a focus on the probabilistic approach (i.e., EG scheme).
- *Sensor-mode Secret-Protecting (SP) architecture*: Two new secure architectures that defend against node fabrication attacks for sensors with very limited or moderate capabilities, and enhance the security of mobile sensor network key management.
- *SAP reduction*: Extensive simulation results showing how node fabrication attacks increase the SAP and how our new architecture reduces SAP to an insignificant level.

The rest of the paper is organized as follows. In Section II, we calculate the values of SAP in both static and mobile networks, with a focus on the probabilistic approach (i.e., EG scheme). In Section III, we propose processor architecture based techniques for securing secret keys and critical software on a node. In Section IV we analyze the security of the proposed architecture under several specific attacks. In Section V, we validate the analytical results from Section II and the security improvement of the proposed architecture with simulations based on a C++ simulator. We conclude in Section VI.

II. Fragility Analysis for Probabilistic Key Management

In this section, we first review the results in [1], where the successful attack probability (SAP) is calculated for a static network. We then consider a mobile network, and show how the value of SAP is significantly larger. We only consider the attacks on the *pairwise* link between two authorized nodes that are within each other's communication range. The SAP will be even higher if A and B are far away and can only be connected with a multi-hop path, since a successful attack on any hop will jeopardize the confidentiality of the whole communication.

The establishment of a link requires two neighbor nodes, A and B, to be able to encrypt the communication over such a link using a common key. This could be achieved in two ways:

(i) A and B share a key within their preloaded key rings, thus can establish the link directly.

(ii) A and B do not share a key initially, and need to exchange additional information through one or more relay nodes, with whom the pairwise links have already been established. For example, A can randomly choose an unused key from its key-ring and send it to B through the relay node(s). Then A and B can use this key to encrypt the pairwise key between them.

TABLE I
Summary of Notation

Notations	Meaning
$A \leftrightarrow B$	A and B establish a pairwise link between them
$A \leftrightarrow \mathbb{C}^h \leftrightarrow B$	A and B communicate through one node in $\mathbb{C}^h$
$A \otimes B$	The link between A and B is successfully attacked
$A\sharp B$	A and B share at least one key
$(A\sharp B) \lhd C$	C has all the keys (≥ 1) shared by A and B
$(A\sharp B) \lhd \mathbb{C}^h$	At least one node of $\mathbb{C}^h$ has all the keys (≥ 1) shared by A and B
$(A, B)\sharp\mathbb{C}^h$	At least one node in $\mathbb{C}^h$ shares at least one key with A and at least one key with B
$(A, B)\sharp\mathbb{C}^h_r$	Exactly r nodes out of $\mathbb{C}^h$, each of which shares at least one key with A and at least one key with B

In either case, SAP of the link between A and B is defined as

$$SAP \triangleq P(A \otimes B | A \leftrightarrow B), \tag{1}$$

where $A \otimes B$ denotes the event that the link between A and B is successfully attacked, and $A \leftrightarrow B$ denotes the event that A and B establish a link between them. Since a link can only be attacked if it has been established, we have (2) and (3) below.

$$P(A \otimes B \cap A \leftrightarrow B) = P(A \otimes B) \tag{2}$$

$$SAP = \frac{P(A \otimes B)}{P(A \leftrightarrow B)} \tag{3}$$

All the notation used in this section is defined in Table I to enable a cleaner presentation of later derivations. A, B and C denote three generic nodes, and $\mathbb{C}^h$ denotes a set of h nodes. Each node is preloaded with a key-ring of k randomly chosen keys out of a key pool of size m.

A. SAP for a static network

If a compromised node wants to attack an established link, it needs to know the key that is used to encrypt the link. Therefore a compromised node can successfully attack an existing link with probability k/m, as stated in [1].

B. SAP for a mobile network

In a mobile network, a compromised node C can attack the link between A and B in three ways:

(i) If A and B share a key initially and establish the link directly, then C needs to know the key chosen by A and B to encrypt the link.

(ii) If A and B do not share a key initially and use C as a relay, then C can get the desired information while relaying the information between A and B. A

first communicates with C via encrypted messages protected by shared key K_{ac}. C decrypts this with K_{ac} giving it access to the plaintext message, and encrypts this with K_{cb}, a key it shares with node B, then sends the re-encrypted message to B. This sets C up as a man-in-the-middle eavesdropper between A and B, since C can see the plaintext of all messages going from A to B.

(iii) If A and B do not share a key and do not choose C within the relay path, C can still attack the communication between A and B by either eavesdropping on the links along the relay path or attacking the eventual pairwise link established between A and B, if it has any of the keys used for these links.

Overall, the value of SAP depends on the number of compromised nodes and authorized nodes within both A and B's communication range, as well as how A and B choose the relay nodes. To simplify the analysis, we only consider cases (i) and (ii), and further assume only one node relay in case (ii). In the simulation in Section V, we calculate SAP for all three cases.

It will be useful to know the probability of sharing at least one key between any two nodes in the network. Denote δ_m^k as the probability that any two nodes A and B do *not* share any key, then

$$\delta_m^k \triangleq P\left(\overline{A\sharp B}\right) = \binom{m-k}{k}\Big/\binom{m}{k}, \tag{4}$$

where $A\sharp B$ denotes A and B share at least one key. The value of δ_m^k can be either accurately calculated as $\prod_{i=0}^{k-1}(m-k-i)/(m-i)$, or approximated using Stirling's approximation for $n!$ as in [1], i.e.,

$$\delta_m^k = \frac{\binom{m-k}{k}}{\binom{m}{k}} \approx \frac{(1-\frac{k}{m})^{2(m-k+0.5)}}{(1-\frac{2k}{m})^{m-2k+0.5}}. \tag{5}$$

Then the probability of A and B sharing at least one key is

$$P(A\sharp B) = 1 - \delta_m^k. \tag{6}$$

For example, if $k = 83, m = 10000$, $P(A\sharp B) \approx 50\%$.

Next we derive the value of SAP based on the number of authorized users and compromised users within both A and B's communication range. We start with the simplest case, where there is only one compromised node available. We then consider the case where there are h compromised nodes. Finally, we consider the case with h compromised nodes and g authorized nodes.

1) Scenario I: only one compromised node C is within both A and B's communication range: Depending on whether A and B share a key initially, they may establish the pairwise link with or without the relay of C. The probability of successfully establishing the link is (7) and the probability of attacking the link is (8).

$$P(A \leftrightarrow B) = P(A\sharp B) + P((A\sharp C \cap B\sharp C) \cap \overline{A\sharp B}), \tag{7}$$

$$P(A \otimes B) \geq P((A\sharp B) \lhd C) + P((A\sharp C \cap B\sharp C) \cap \overline{A\sharp B}). \tag{8}$$

Here $((A\sharp B) \lhd C)$ means that A and B share at least one key, and all the shared keys between A and B are within the key-ring of node C. Since we ignore the case where C only knows a subset of the shared keys between A and B, where C still has a chance to successfully attack the link between A and B, we have an inequality in (8) instead of an equality.

Let us calculate each term in (7) and (8). We know the value of $P(A\sharp B)$ from (6). Also,

$$\begin{aligned} &P(A\sharp C \cap B\sharp C|\overline{A\sharp B}) \\ &= 1 - P(\overline{A\sharp C}) - P(\overline{B\sharp C}) + P(\overline{A\sharp C} \cap \overline{B\sharp C}|\overline{A\sharp B}) && (9)\\ &= 1 - 2\delta_m^k + \binom{m-2k}{k}\Big/\binom{m}{k} && (10)\\ &= 1 - 2\delta_m^k + \binom{m-k}{k}\Big/\binom{m}{k} \cdot \binom{m-2k}{k}\Big/\binom{m-k}{k} && (11)\\ &= 1 - 2\delta_m^k + \delta_m^k \cdot \delta_{m-k}^k && (12) \end{aligned}$$

Define

$$\phi_m^k \triangleq P(A\sharp C \cap B\sharp C|\overline{A\sharp B}), \tag{13}$$

we then have

$$\begin{aligned} &P(A\sharp C \cap B\sharp C \cap \overline{A\sharp B}) \\ &= P(\overline{A\sharp B}) \cdot P(A\sharp C \cap B\sharp C|\overline{A\sharp B}) \\ &= \delta_m^k \phi_m^k. && (14) \end{aligned}$$

Thus from (6), (7) and (14)

$$P(A \leftrightarrow B) = 1 - \delta_m^k + \delta_m^k \phi_m^k, \tag{15}$$

Meanwhile,

$$\begin{aligned} &P((A\sharp B) \lhd C) \\ &= \sum_{i=1}^{k} \left(\binom{k}{i} \cdot \left(\frac{\binom{m-k}{k-i}}{\binom{m}{k}} \right) \cdot \left(\frac{\binom{m-i}{k-i}}{\binom{m}{k}} \right) \right) && (16)\\ &\geq \binom{k}{1} \cdot \left(\frac{\binom{m-k}{k-1}}{\binom{m}{k}} \right) \cdot \left(\frac{\binom{m-1}{k-1}}{\binom{m}{k}} \right) && (17)\\ &= k \left(\frac{k}{m-2k+1} \cdot \frac{\binom{m-k}{k}}{\binom{m}{k}} \right) \cdot \left(\frac{\frac{(m-1)!}{(k-1)!(m-k)!}}{\frac{m!}{k!(m-k)!}} \right) && (18)\\ &= \frac{\delta_m^k k^3}{m(m-2k+1)}, && (19) \end{aligned}$$

whereas in (17), for simplicity we ignore the event that A, B and C share more than one key. Define

$$\gamma_m^k \triangleq P((A\sharp B) \lhd C),$$

we then have

$$SAP = \frac{P(A \otimes B)}{P(A \leftrightarrow B)} \geq \frac{\gamma_m^k + \delta_m^k \phi_m^k}{1 - \delta_m^k + \delta_m^k \phi_m^k}. \tag{20}$$

2) Scenario II: h compromised nodes are within both A and B's communication range: We use $\mathbb{C}^h$ to denote the set of h compromised nodes. Since

$$\begin{aligned} &P((A,B)\sharp\mathbb{C}^h \cap \overline{A\sharp B}) \\ =&P(\overline{A\sharp B}) \cdot P((A,B)\sharp\mathbb{C}^h|\overline{A\sharp B}) && (21) \\ =&P(\overline{A\sharp B}) \cdot (1 - (1 - P(A\sharp C \cap B\sharp C|\overline{A\sharp B}))^h) && (22) \\ =&\delta_m^k \cdot (1 - (1 - \phi_m^k)^h), && (23) \end{aligned}$$

then using a similar argument as in Scenario I, we have

$$\begin{aligned} SAP \geq& \frac{P((A\sharp B) \lhd \mathbb{C}^h) + P((A,B)\sharp\mathbb{C}^h \cap \overline{A\sharp B})}{P(A\sharp B) + P((A,B)\sharp\mathbb{C}^h \cap \overline{A\sharp B})} && (24) \\ \geq& \frac{1 - (1 - \gamma_m^k)^h + \delta_m^k \cdot (1 - (1 - \phi_m^k)^h)}{1 - \delta_m^k + \delta_m^k \cdot (1 - (1 - \phi_m^k)^h).} && (25) \end{aligned}$$

3) Scenario III: h compromised nodes and g authorized nodes are within both A and B's communication range: In this case, if A and B do not share any key initially and need to communicate through a relay, a successful attack can happen if one compromised node is chosen as the relay. Assuming there are a total of a *qualified* relays (i.e, nodes who can establish pairwise links with both A and B), b out of which are compromised nodes. Denote μ_a^b as the probability of A and B picking a compromised node as the relay, which can have different values depending on the specific attack models (details in the next subsection).

The probability of having r useable relays out of all h compromised nodes when A and B do not share keys is

$$\begin{aligned} &P((A,B)\sharp\mathbb{C}_r^h|\overline{A\sharp B}) \\ &= \binom{h}{r}\left(P(A\sharp C \cap B\sharp C|\overline{A\sharp B})\right)^r \left(1 - P(A\sharp C \cap B\sharp C|\overline{A\sharp B})\right)^{h-r} && (26) \\ &= \binom{h}{r}(\phi_m^k)^r(1 - \phi_m^k)^{h-r}. && (27) \end{aligned}$$

Similarly, the probability of having w useable relays out of all g authorized nodes when A and B do not share keys is

$$P((A,B)\sharp\mathbb{C}_w^g|\overline{A\sharp B}) = \binom{g}{w}(\phi_m^k)^w(1 - \phi_m^k)^{g-w}. \tag{28}$$

Then the probability of sending a message through a compromised node given the existence of h compromised nodes, g authorized nodes and A, and B do not share any key is

$$\begin{aligned} &P(A \leftrightarrow \mathbb{C}^h \leftrightarrow B|\overline{A\sharp B}) \\ &= \sum_{r=1}^{h}\sum_{w=0}^{g} \mu_{r+w}^r (P((A,B)\sharp\mathbb{C}_r^h|\overline{A\sharp B}) \cdot P((A,B)\sharp\mathbb{C}_w^g|\overline{A\sharp B})) && (29) \\ &= \sum_{r=1}^{h}\sum_{w=0}^{g} \mu_{r+w}^r \left(\binom{h}{r}\binom{g}{w}\left((\phi_m^k)^{r+w}(1 - \phi_m^k)^{h+g-(r+w)}\right)\right). && (30) \end{aligned}$$

Since

$$P(A \leftrightarrow \mathbb{C}^h \leftrightarrow B \cap \overline{A\sharp B}) = P(A\sharp B) \cdot P(A \leftrightarrow \mathbb{C}^h \leftrightarrow B|\overline{A\sharp B}), \tag{31}$$

we have the following lower bound on SAP

$$\begin{aligned} SAP =& \frac{P(A \otimes B)}{P(A \leftrightarrow B)} && (32) \\ \geq& \frac{P((A\sharp B) \lhd \mathbb{C}^h) + P(A \leftrightarrow \mathbb{C}^h \leftrightarrow B \cap \overline{A\sharp B})}{P(A\sharp B) + P(A \leftrightarrow \mathbb{C}^{h+g} \leftrightarrow B \cap \overline{A\sharp B})} && (33) \\ =& \frac{1 - (1 - \gamma_m^k)^h + \delta_m^k \cdot \left(\sum_{r=1}^h \sum_{w=0}^g \mu_{r+w}^r \left(\binom{h}{r}\binom{g}{w}\left((\phi_m^k)^{r+w}(1 - \phi_m^k)^{h+g-(r+w)}\right)\right)\right)}{1 - \delta_m^k + \delta_m^k \cdot \left(1 - (1 - \phi_m^k)^{h+g}\right)}. && (34) \end{aligned}$$

4) Numerical results: Table II shows the SAP for different values of h and g based on the previous analysis. The key-ring size is $k = 83$, with a key pool size of $m = 10000$.

TABLE II
Successful attack probability (SAP) for different numbers of authorized nodes (g) and compromised nodes (h). We assume there are a total of a qualified relays, b out of which are compromised nodes. μ_a^b is the probability of picking a compromised node as the relay. The key pool size $m = 10000$, the preloaded key-ring size $k = 83$, and the original SAP estimation is hk/m.

h	$g = 0$	$g = 10$		$g = 20$		hk/m
		$\mu_a^b = b/a$	$\mu_a^b = 1$	$\mu_a^b = b/a$	$\mu_a^b = 1$	
1	20.4%	4.7%	13.0%	2.7%	12.8%	0.8%
2	31.1%	8.8%	22.7%	5.1%	22.4%	1.7%
3	37.6%	12.3%	30.0%	7.4%	29.7%	2.5%
4	41.9%	15.3%	35.5%	9.5%	35.2%	3.3%
5	44.8%	18.0%	39.7%	11.4%	39.5%	4.2%
6	46.9%	20.4%	42.9%	13.2%	42.7%	5.0%
7	48.5%	22.5%	45.4%	15.0%	45.2%	5.8%
8	49.7%	24.4%	47.3%	16.6%	47.2%	6.6%
9	50.6%	26.2%	48.8%	18.1%	48.7%	7.5%

Several observations are in order. When the probability of picking a compromised node as the relay $\mu_a^b = b/a$, the SAP increases with h (the number of compromised nodes) under fixed g (the number of authorized nodes). When $\mu_a^b = 1$, the general trend is similar, but the SAP is not very sensitive to g between the cases of $g = 10$ and $g = 20$, since A and B will always choose a compromised node as relay if possible. Comparing with the value of SAP estimated in [1], which is approximated as hk/m, the SAP in Table II is much larger. For example, with $\mu_a^b = b/a$, $h = 9$ and $g = 20$, we have a SAP of 18.1%, as opposed to $hk/m = 7.5\%$. The value of SAP increases further when $\mu_a^b = 1$.

The value of μ_a^b depends heavily on the attack model used by the compromised nodes. We define two attack models, *honest attack* and *smart attack*. In an honest attack, the relays nodes are randomly chosen and $\mu_a^b = b/a$. In a smart attack, however, the compromised nodes will improve the value of μ_a^b by various methods. In a *smart attack with incentive*, the compromised nodes provide incentives for nodes A and B to choose one of them as a relay. If the choice of relay is determined by a shortest path routing protocol, the compromised nodes can announce distance metrics of the links connected to them smaller than the actual values. If the choice of relay is based on energy efficiency, the compromised nodes can pretend to be very energy efficient. In most cases, the incentives provided by the compromised nodes can make the value of μ_a^b very close to 1. In *a smart attack with virtual node fabrication*, each compromised node is able to collect the keys from all other compromised nodes, and can then fabricate up to $\binom{hk}{k}$ nodes with distinct key rings. The number will be very large if $h \geq 2$. For example, when two nodes are captured with non-overlapping key rings, then

$$\binom{2k}{k} = \frac{(2k)!}{k!} \approx \frac{\sqrt{2\pi}(2k)^{2k+0.5}e^{-2k}}{(\sqrt{2\pi}(k)^{k+0.5}e^{-k})^2} = \frac{2^{2k+0.5}}{\sqrt{2\pi k}}, \tag{35}$$

which is around 5.8×10^{48} if $k = 83$. As a result, the value of μ_a^b will be closer to 1 with the increase of the number of fabricated nodes.

III. Secret-Protecting Processor Architecture

The analysis in the previous section is based on the assumption that an adversary can obtain the long-term key information from the captured nodes. In particular, we assume an adversary with physical access to the device, so software protections are easily bypassed. The keys are accessible to an adversary if the existing software is exploited or if the software is replaced entirely with malicious code. He might also read the keys directly from a flash memory chip or other permanent storage when the device is offline.

We propose a solution that protects secrets by storing them inside the System on a Chip (SoC). The chip includes the processor core and main memory, and it is quite expensive for an adversary to remove the packaging and directly probe the registers and memory. The SoC chip can further implement physical tamper-resistance mechanisms that will clear the secrets when probing attempts are detected and also cut power to the chip, erasing any intermediate data based on those secrets. Therefore the assumption of protected on-chip secrets is valid for a large class of attacks.

Our solution is to provide a Secret Protecting (SP) architecture which minimizes the trusted computing base (TCB) of hardware and software that has to be fully correct, verified and trusted. Our TCB comprises some SP hardware features (described below) and a small Trusted Software Module (TSM) that does the key management.

We first present Reduced Sensor-mode SP, suitable for the simplest sensors. We then extend the solution for slightly more capable sensors. Our work is inspired by the SP architecture proposed for general-purpose microprocessors [19], [20], but stripped to the bare minimum for sensors with very constrained computing and storage resources.

A. Reduced Hardware Architecture

The simplest version of our architecture, Reduced Sensor-mode SP, is shown in Fig. 1. It only requires one new register — the Device Key — and a bit to indicate protected mode. Additionally, a Trusted Software Module (TSM) is stored in the on-chip instruction EEPROM, and the long-term keys for the probabilistic key management scheme, provided by a central authority, are stored in the on-chip data EEPROM. Also, a portion of the main memory of the node is reserved for the TSM Scratchpad Memory.

The main concept is that the TSM is the only software module that can use the Device Key and the protected long-term keys. Since the TSM code is stored within the trusted SoC chip in ROM, it cannot be changed by other software — whether by a malevolent application or a compromised operating system. Similarly, the long-term keys never leave the SoC chip. Also, any intermediate data (which may leak key bits) generated during TSM execution is placed in the TSM scratchpad memory, and also never leaves the SoC chip. We will discuss how this prevents node fabrication attacks in Section IV.

The TSM code is stored on-chip in a segment of the existing instruction EEPROM along with other system software for the node. Similarly, the long-term keys from the authority are stored in a TSM segment of the data EEPROM. The keys are encrypted with the device key or with another encryption key derived from it by the TSM.

The device key is the SP master key and is protected by the processor hardware; it can only be used by the

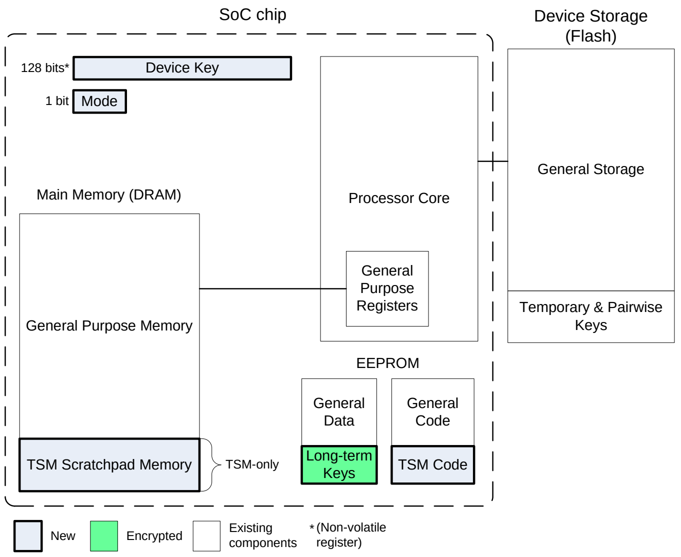


Fig. 1. Reduced Sensor-mode SP

TSM running in protected mode and can never be read by any other software.

When the unprotected software wants to make use of protected keys, it calls the TSM. The TSM functions access the protected keys, perform the requested operation and return the results, never revealing the protected keys themselves to the unprotected software. Each TSM function starts with a *Begin_TSM* instruction, which disables interrupts, sets the protected mode bit, and enters protected mode for the next instruction. *Begin_TSM* is only valid for code executing from the instruction-EEPROM; any code executed from main memory or off-chip storage cannot enter protected mode at all. The end of the TSM code is indicated by the *End_TSM* instruction which clears the mode bit and re-enables interrupts. Table III shows the set of instructions used only by the TSM and for initialization, in the Sensor-mode SP architectures.

The TSM Scratchpad Memory is a section of main memory reserved for the exclusive use of the TSM. It is addressed separately from the regular on-chip memory and accessed only with special *Secure_Load* and *Secure_Store* instructions (see Table III). These new instructions are available only to the TSM, making it safe for storing sensitive intermediate data in the TSM scratchpad memory. The TSM can also use this extra space to spill general registers, to decrypt and store keys, and to encrypt data for storage in regular unprotected memory.

Initialization of a new device takes place at the authority's depot. First it must generate a new random device key. Long-term keys and other secrets are encrypted with it are then stored along with the TSM code on the on-chip EEPROM. Next it uses the *DeviceKey_Set* instruction to store the device key. Finally, any other unprotected software and data can be copied to the flash storage.

Any time the Device Key register is set (or cleared), the processor will automatically clear the TSM scratchpad memory, wiping any intermediate data that was protected by the old key. If in protected mode at the time, the mode bit is also cleared along with the general purpose registers. Similarly, the processor will clear the device key upon writing to either the instruction or data EEPROM; this in turn clears the other intermediate data.

B. *Expanded Sensor-mode SP Architecture*

The Reduced Sensor-mode SP architecture is ideal for the smallest sensor nodes which use minimal software and have very limited resources. In slightly larger lightweight sensor nodes, the software will be more complex. The additional applications that run on this sensor combined with the TSM and long-term keys will be too large to store on-chip. This greater flexibility in the sensor also requires additional support for security. Hence, we propose the Expanded Sensor-mode SP architecture shown in Fig. 2.

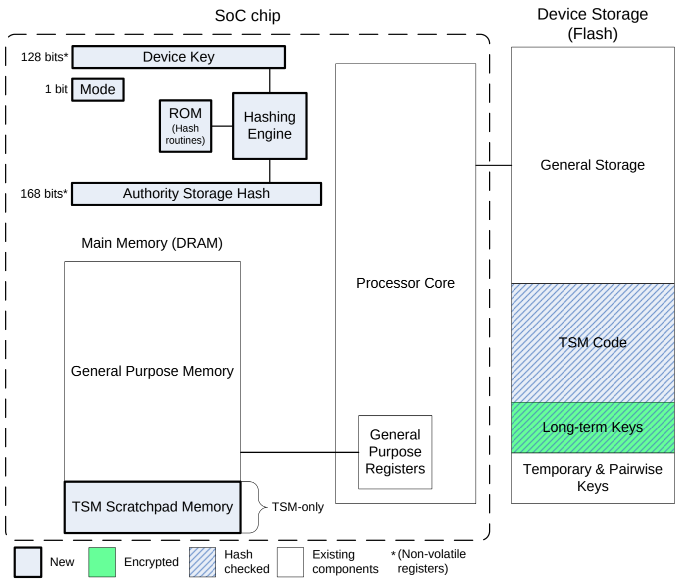


Fig. 2. Expanded Sensor-mode SP

The TSM code and encrypted long-term keys are moved to the off-chip device storage. This makes them susceptible to modification by other software or through physical attacks. Therefore we must verify their integrity before they can be used. To do this, we add a new register — the Authority Storage Hash (ASH), a hardware hashing engine (implementing SHA-1, MD5, or

TABLE III
New Sensor-mode SP Instructions

Instruction	**Description**
Begin_TSM	Begins execution of the TSM
End_TSM	Ends execution of the TSM
Secure_Store	Secure store from processor to TSM scratchpad memory. (TSM only)
Secure_Load	Secure load from TSM scratchpad memory to processor. (TSM only)
DeviceKey_Read	Read the Device Key. (TSM only)
DeviceKey_Set	Sets the Device Key register. First clears the TSM scratchpad memory.
ASH_Set	Sets the ASH register. First clears the device key and TSM scratchpad memory.

another cryptographic hash function), a small ROM, and an additional initialization instruction.

The ASH register contains a hash over the entire memory region of the TSM code and long-term keys. It is set by the authority during initialization and is rechecked by the processor each time the TSM is called. The checking code is stored in the on-chip ROM and is fixed and therefore safe from modification; it uses the hardware hashing engine to compute the hash over the TSM code and the encrypted keys. When *Begin_TSM* is called, the processor disables interrupts and jumps to the TSM-checking routine. If the hash check succeeds, the protected mode bit is set, and execution jumps to the newly-verified TSM code. If the check fails, an exception is triggered. The *ASH_Set* instruction sets the ASH register, first clearing the device key to ensure that the TSM can't be replaced and still access the protected keys.

IV. Security and Economics Analysis of SP Architecture Based Solution

A. Attacks on Protected Keys

Our new Sensor-mode SP architectures safeguard a sensor node's long-term keys, preventing extraction by an adversary in the event of node capture. The keys are always stored in encrypted form in permanent storage in either on-chip EEPROM or off-chip storage. The adversary cannot obtain the device key needed to decrypt them. The device key never leaves the SP processor or its protected software environment. Therefore, rather than access the keys directly, regular software must call TSM functions which perform operations with the keys on its behalf. Thus, software can use the keys in any way permitted by the TSM, but can never extract the keys themselves, even under physical attacks.

1) Node Fabrication Attacks: Without SP protection, an adversary maximizes his SAP by cloning multiple copies of compromised nodes and combining their long-term keys. This increases his ability to observe link establishment and the likelihood of being used as a relay. With SP protection, he cannot create any clones and is limited to using only the keys originally stored on the captured node.

2) Node Capture Attacks: Node capture attacks use long-term keys in the node to observe pairwise links between other nodes in the network. With SP, an adversary can no longer extract the keys. However, he can still change unprotected software which calls the TSM. A simple TSM might provide functions like *Encrypt(key, data*) and *Decrypt(key, data)*. The adversary can use the keys through this TSM interface to observe or attack pairwise links without ever seeing the actual keys. While we do not prevent node capture attacks outright, such attacks are limited since the adversary can only observe links within the communication range of the compromised node. We show in Section V that this severely limits the SAP, which is constrained by the number of captured nodes.

B. Attacks on Changing the TSM or the Device Key

The security of the long-term keys relies on the correctness and proper design of the authority's TSM. As part of the trusted computing base of the system, this software must not leak secrets it has access to. This includes any intermediate data written to general purpose memory, placed in off-chip storage, or left in general registers when it exits. The TSM runs with interrupts disabled, so no other software will have an opportunity to observe its registers or modify its code or data while it is executing. If the TSM ever exits abnormally due to an exception, the processor clears the general registers before ending protected mode. Any other sensitive data will be in the TSM scratchpad memory which other software cannot access.

In order to circumvent the access control provided by the authority's TSM, the attacker might try to replace it with his own TSM or modify the existing TSM. In Reduced Sensor-mode, the TSM and long-term keys are stored in on-chip EEPROM where they cannot be modified without clearing the device key. In Expanded Sensor-mode, the attacker could modify or replace the TSM code in off-chip storage. The hash checking routine will detect any such modifications made to the TSM before execution. We assume that the data in off-chip storage cannot be modified through a physical attack during execution. If this is not the case, the TSM and keys should first be copied to general purpose memory on-chip before being verified, where they will be safe from physical attacks.

Finally, if the attacker tries to modify the ASH register to match the new TSM code, the device key will be cleared, irrevocably cutting off his access to all of the keys that were encrypted with that device key. Clearing or setting the device key also clears the TSM scratchpad memory, so any intermediate data stored there that might have leaked secrets is also unavailable to the new TSM.

C. Economics Analysis

When considering low-cost sensors, any new hardware must be designed for high volume in order to keep down fabrication costs. Accordingly, Sensor-mode SP provides basic security primitives and a hardware root of trust using a design that is easily integrated into the SoC of standard embedded processors. It therefore supports a wide range of software protection mechanisms with only a slight increase in chip area.

Our hardware also provides physical security. SP prevents attacks by an adversary with physical control over a captured node who tries to modify the code or data in storage while the device is in operation or offline. The physical integrity of the SoC itself is sufficient to prevent adversaries from probing the SP registers inside the chip, without requiring more costly tamper-proofing mechanisms in most cases.

V. Simulation Results

To verify our probability computations in Section II and demonstrate the improvement of security performance of the proposed architecture in Section III, we evaluate the SAP of the probabilistic and deterministic key predistribution scheme (the EG scheme) through a simulator written in C++.

A. Comparison of Probabilistic and Deterministic Key Predistribution

To compare probabilistic and deterministic key predistribution schemes, we consider a unit disk network model, as shown in Fig. 3. A total of g authorized nodes (denoted by symbol A) are uniformly distributed in the unit disk. All the compromised nodes (including any virtually fabricated nodes) are placed at the center of the unit disk and denoted with symbol C. All nodes are assumed to have the same transmission range equal to the radius of the disk. This means an adversary can eavesdrop on any communication in the unit disk through the compromised nodes as long as it has the right key(s). Two neighbor nodes will setup a pairwise link directly if they share one or more keys. Otherwise, they will try to find a relay path through one or more nodes to exchange additional key information, so that they can set up pairwise link between them. When there is more than one qualified relay node available, the authorized nodes will choose a relay randomly in the case of $\mu_a^b = b/a$ (i.e, honest attack or finite virtual node fabrication), or search for a shortest relay path in an attack with incentive.[2] Any two nodes that are not neighbors cannot establish pairwise links among themselves. The main reason of using the above unit disk network model is to derive a uniform and fair metric (i.e., SAP) among various approaches where failing to attack is only due to the lacking of appropriate keys rather than the limitation of transmission range.

The SAP is calculated as the fraction of the links (among all the pairwise links) that can be eavesdropped by the compromised nodes. As we explained in Sec. I, a basic deterministic scheme like single common key either enables almost zero SAP in a static network, or leads to 100% SAP for the unit disk model in a mobile network. Hence, our focus here is to determine the SAP for the probabilistic key predistribution scheme (i.e., the EG scheme). All the simulation results are averaged over 10 sets of random seeds which affect the distribution of the authorized nodes within the unit disk, the key ring preloaded to each node and the choices in case of multiple qualified relays.

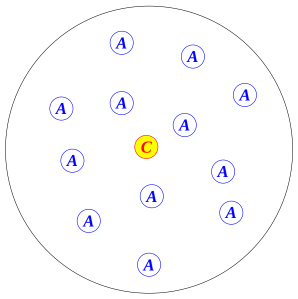
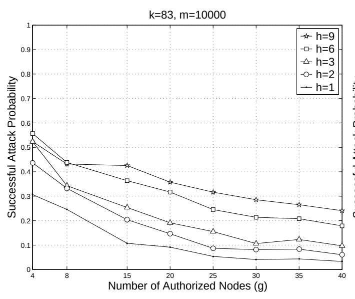


Fig. 3. Unit disk network model with unit radius. The authorized nodes are uniformly distributed in the unit disk and denoted by symbol A. The compromised and virtually fabricated nodes are placed in the center of the network and denoted by symbol C. All nodes have the same communication range equal to the disk radius.

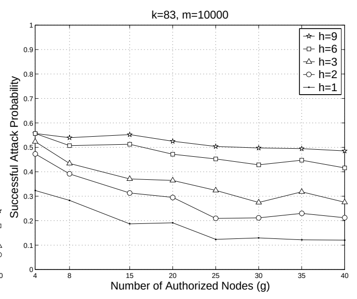
Figs. 4(a) to 4(d) illustrate the values of SAP under different assumptions on the number of compromised nodes (h), number of authorized nodes (g) and different attack models (honest attack, smart attack with incentive, or smart attack with fabrication). Unless otherwise specified, each node is preloaded with a key ring consisting of $k = 83$ keys that are randomly chosen from a key pool of size $m = 10,000$.

Fig. 4(a) shows the SAP for various values of h and g under the *honest attack*. For a fixed value of h, the SAP decreases when the density of authorized nodes

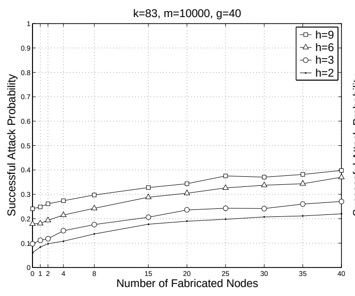
[2]In the simulation, the smart attack with incentive is approximated as setting the cost of the links adjacent to the compromised nodes as 0.9999 instead of as 1 unit (hop) for other authorized nodes.



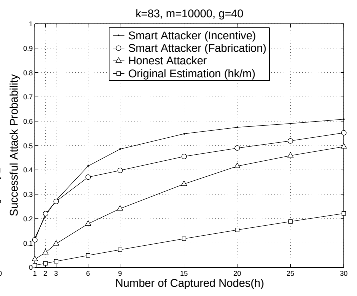
(a) Honest attack



(b) Smart attack (incentive)



(c) Smart attack (node fabrication)



(d) Different attack models

Fig. 4. Successful Attack Probability with various numbers of captured nodes (h) and authorized nodes (g)

increases. This is because in a denser network, there are more qualified relay nodes available between any two neighbor nodes, thus the probability of choosing a compromised node as the relay is smaller under honest attack. For a fixed number of authorized nodes g, a higher value of h increases the probability of picking a compromised node as the relay, thus leads to a higher value of SAP. In a network where there are 9 compromised nodes and 15 authorized nodes, the SAP could be as high as 42%.

Fig. 4(b) shows the SAP for various values of h and g under the *smart attack with incentive*. In this case, two neighbor nodes without a common key will have a high chance to pick a compromised node as relay if it is qualified. There is a high probability of finding a qualified relay node among the compromised nodes when h is large, in which case the SAP is insensitive to the number of authorized nodes g. Similarly as in Fig. 4(a), a higher value of h also leads to a higher value of SAP. In a network with 40 authorized nodes and 9 compromised nodes, the SAP would be around 50%.

Fig. 4(c) shows the SAP for the smart attack of various numbers of compromised nodes and different total numbers of virtually fabricated nodes. The total number of authorized nodes is kept at 40. The node fabrication is achieved as follows. All the keys collected from the h compromised nodes will constitute a *compromised key pool*. Then each fabricated node will be loaded with $k = 83$ keys randomly chosen from the compromised key pool. A larger number of fabricated nodes increases the chance of such a node being chosen as a relay node, thus increasing SAP. A larger value of h leads to a larger compromised key pool, which again increases the chance of a fabricated node serving as a qualified relay.

Fig. 4(d) shows the SAP under different numbers of captured nodes, for different kinds of attacks as well as the estimation based on the result in [1][3]. The number of authorized nodes is fixed at 40. It is clear that the results in [1] significantly underestimate the SAP in mobile networks. With a large enough number of compromised nodes, the SAP can easily reach an unacceptably high value of 50% with all attack models.

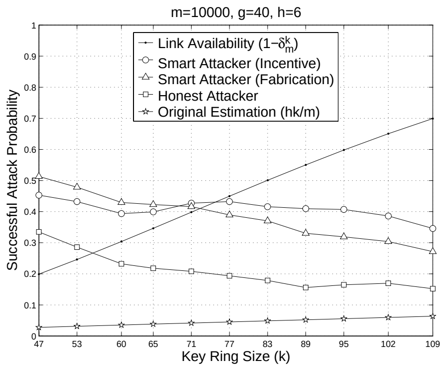


Fig. 5. Successful attack probability for different key ring size (k), for different attack models as well as the estimation in [1].

Fig. 5 shows the SAP under different sizes of the preloaded key ring, k, for different attack models as well as the estimation based on the results in [1]. We also plot the link connectivity (i.e., the probability that two neighbor nodes share at least one key, $1 - \delta_m^k$) under different values of k. With the increase of k, the link connectivity increases, as well as the SAP estimation based on the analysis in [1], which is linear in k. On the other hand, the SAPs for all three attack models actually decrease with an increasing k, due to less need of going through a relay to establish a pairwise link. However, they are still much higher than the original estimation of SAP in [1] and the nodes need more memory to store so many keys.

B. Security Improvement with SP architecture

In this subsection, we show the security performance of the proposed SP architecture for lightweight ad hoc networks. We focus on the evaluation of the basic probabilistic key predistribution approach (the EG scheme)

[3]When the network is static, an adversary captures h nodes, then its successful attack probability on a link is $1-(1-\frac{k}{m})^h \approx \frac{hk}{m}$.

since the deterministic approach (e.g., single common key) is a special case of the probabilistic approach. In addition, many advanced versions of probabilistic key predistribution (e.g., [8], [21]) are also vulnerable to node capture attacks and can benefit from the proposed architecture.

We have run the simulation for a 10×10 grid network, and all nodes are assumed to have the same (1 unit) transmission range. A total of 400 nodes are randomly placed in the network. Network-wide SAP is calculated as the fraction of links that can be intercepted by the compromised nodes among all the pairwise links established among the authorized nodes. All simulation results are averaged over 10 sets of random seeds that affect the distributions of the location of each node, the key rings preloaded to nodes, and the relay choices.

We consider several possible attack models depending on whether SP architecture is used. If every node is equipped with the Sensor-mode SP architecture, the adversary can only launch a *node capture attack*, where the adversary utilizes the captured nodes themselves to intercept pairwise-key establishment. Without the SP architecture, the adversary can further launch *node fabrication attacks* where he can turn the captured nodes into super-nodes by loading each of them with all of the keys from all captured nodes. Each super-node can mimic multiple nodes. A straightforward method to achieve this is to let each super-node stay at its original location but announce the existence of all the captured nodes. The adversary can even make more copies of the super-nodes and deploy them into the network to eavesdrop additional communication. We note that it is difficult to detect the duplication of nodes within the network, since it requires knowledge of the location of each node (possibly using Global Positioning System) and non-trivial communication and memory overhead [22].

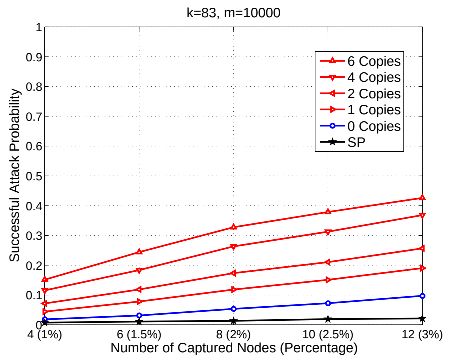


Fig. 6. Network-wide successful attack probability under different numbers of captured nodes for different attack models

Fig. 6 shows the network-wide SAP under different numbers of captured nodes, for different kinds of attacks. "SP" means launching only the node capture attack with the SP architecture. "0 copies" means changing captured nodes into super-nodes (i.e., node fabrication attack) due to the lack of the SP architecture. "*x* copies" means making *x* extra copies of these super-nodes. Note that, without SP, the effect of node capture can be serious. When only 3% of the nodes are captured, the SAP for the network will be 9.7% even with "0 copies", and becomes 42.6% if the adversary makes 6 copies of the captured nodes to cover more area. Whereas the SAP for the nodes with SP is only 2.1% — a reduction by roughly an order of magnitude. Therefore, SP provides significant benefits in terms of alleviating node fabrication attacks.

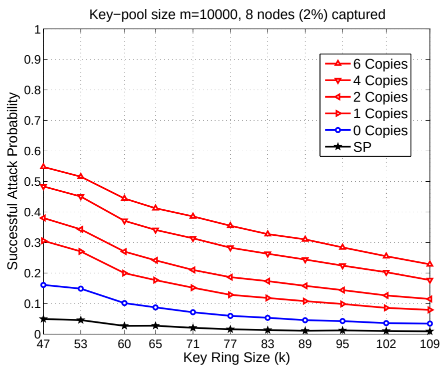


Fig. 7. Network-wide successful attack probability for different key ring size (k) with different attack models

Fig. 7 shows the network-wide SAP under different sizes of the preloaded key ring, *k*, for different attack models, assuming 2% of nodes have been captured. An increasing value of *k* has two effects on the network. First, the link connectivity increases; this reduces the probability of two neighboring nodes establishing a pairwise link through a relay node, and thus can improve the network security. Second, each node captured by the adversary contains more keys, which will increase the chance of intercepting the communications on other pairwise links. This is detrimental to the network security. Fig. 7 shows that the advantage of the first effect dominates and the overall SAP decreases with an increasing value of *k*. Notice that the SP architecture offers significant advantages over the other schemes for all values of *k*.

Finally, the single common key scheme also benefits from SP since the adversary, without the ability to learn the common key, can only eavesdrop on the information exchanged within the communication range of the captured nodes.

TABLE IV
Comparison of probabilistic and deterministic key predistribution

Metric	Probabilistic	Deterministic
Key Storage	$O(k)$	$O(1)$
Communication Overhead	$O(k)$	$O(1)$
Computation Overhead	$O(\log k)$ {Search}	$O(1)$
Node Density Required	High	Low
Link Connectivity	50%, $\left(1 - \frac{\binom{m-k}{k}}{\binom{m}{k}} \middle\| \begin{smallmatrix} k=83 \\ m=10000 \end{smallmatrix}\right)$	100%
Resiliency (Static Network)	High	High
Resiliency (Mobile Network)	Low	Low
Resiliency (SP-Enhanced)	High	High

VI. Concluding Remarks

In this paper, we discuss key management in lightweight mobile ad hoc networks. Backed up by the large successful attack probabilities computed in this paper, we show that the probabilistic key predistribution schemes are in fact quite vulnerable to node captures in many practical cases. Considering the large key pool and key ring sizes, complex key predistribution, low network connectivity, and complex pairwise link establishments, the advantage of the probabilistic approach over the deterministic approach is not as much as people have believed. A partial list of the comparisons between the two approaches (basic versions) is presented in Table IV. We also generalize the re-examination to other probabilistic key predistribution schemes, including the q-composite key scheme, the multiple disjoint path key reinforcement scheme, and the scheme based on partial deployment information in the Appendix. All of these schemes are vulnerable to node capture in a mobile network. A selective node capture will further weaken the performance of a probabilistic approach.

Finally, we propose two low-cost hardware-based architectures to enhance the security of key management schemes against the attack of sensor node fabrication for a lightweight mobile ad hoc network, which can benefit both probabilistic and deterministic key management.

APPENDIX: Implications to Related Work

A. Reinforcements on the Basic EG Scheme

Many probabilistic schemes based on the EG scheme have been proposed, e.g., [5], [8], [9], [16], [23]. We show that many of them are also very vulnerable to node capture in the mobile networks, or that the proposed improvements in those schemes can benefit deterministic approaches as well.

In the q-composite key scheme [8], two nodes can only establish a pairwise link between them if they share at least q keys initially (i.e., within the preloaded key rings.). The real key for encrypting the communication is a hash result of all q keys. Though this approach can reduce SAP in a static network, it is almost as fragile as the EG scheme in a mobile network, following similar analysis as in Section II. In addition, to keep the link connectivity comparable to the EG scheme, the key ring size has to be substantially increased, which means fewer node captures are needed to disclose a sufficiently large key space to the adversary.

The concept of multiple disjoint path key reinforcement was proposed in [8] and [16]. A node will send partial keys to its neighbor through several disjoint paths. The counterpart then regenerates the original key after receiving all these partial keys. A compromised node can only regenerates the key if it intercepts all partial keys. However, this approach requires each node to maintain a global network topology to calculate disjoint paths, and the ability to do source routing. In addition, if virtual node fabrication is possible, there is still a high probability that every path passes through a compromised node. Finally, the approach is also fragile to the attack of deliberate modification by a compromised node along the path, so that the neighbor node cannot successfully regenerate the key.

Some schemes utilize the "partial" deployment information to increase the resilience against node capture (e.g., [9], [23]). In particular, two nodes have a higher probability of sharing keys if they are supposed to be deployed in a group (e.g., in the same geographic area), or have a lower probability if they will be deployed in different groups. Therefore, a node captured in a particular group will have little effect on the security of nodes in other groups. However, the same technique can also be used to enhance the security of a deterministic scheme. For example, a different common key can be assigned to the nodes deployed in the same group, and a node is then equipped with all the keys of the groups it belongs to. Thus capturing one node will not have much adverse effect on the nodes in other groups. Therefore, such improvements do not change the nature of our comparisons of deterministic and probabilistic key management.

B. Selective Node Capture

So far we have only considered *random node capture*, i.e., the adversary randomly captures nodes to collect sufficient keys to attack the whole network. Another new attack mode, called *selective node capture*, can further weaken the security levels of the probabilistic approaches. As originally proposed in [5], the adversary can listen to the information exchanges when each node tries to identify the keys to be shared with its neighbors.

By identifying the key indices in each node and physically locating any node, the adversary can selectively capture nodes with the least overlap in keys. Compared with random node capture, fewer selective node captures are needed to disclose a certain number of unique keys. Although several methods have been proposed to reduce the communication overhead and avoid unnecessary key index disclosure, none of them can completely preclude selective node capture.

One class of key discovery methods is "key indices notification". A basic approach is that each node announces all of the key indices, with a message size of $O(k)$, as in [1]. Zhu et al. [16] proposed a refined approach that uses a publicly known pseudo-random key index generation function. In the key predistribution phase, the authority's server chooses a random seed for each node, which calculates a set of k outputs (i.e. key indices) using the key index generation function. The node then uses the random seed as its ID, and announces this ID to its neighbors in the key discovery phase. Each of its neighbors can figure out the key indices stored in the node from its ID, since they all have the same key index generation function. As a result, the communication overhead is only of order $O(1)$. However, since the adversary can also figure out all the key index information from the announcements, the scheme is still fragile to selective node capture.

A more complicated challenge-response-like key discovery technique was proposed in [1], [21], [23], where a node can determine whether its neighbor has a particular key if and only if it also has the key. In this scheme, a node announces k challenges separately encrypted by its own k keys. A neighbor node then tries to decrypt the k challenges with its own k keys. Only after the successful decryption of a challenge, can the neighbor node figure out the key from the node who announced the challenge. The process involves k^2 decryption operations and $O(k)$ message exchanges between any two neighbor nodes. To further reduce the computation and communication overhead, Pietro et al. [5] designed a verification function $\Phi(ID||key)$ using a one way hash function, like SHA-1, where $||$ is the concatenation operation. The function $\Phi(ID||key)$ returns "true" for any argument (i.e. $ID||key$) with a (pseudo-random) probability of k/m. In the key distribution phase, each node uses $\Phi(ID||key)$ to test each key k_i from the key pool with its ID, and uses that key only if the result is "true". Therefore, around k keys out of an m sized key pool will be stored on each node. In the key discovery phase, the node just needs to announce its ID, and a neighbor node will execute $\Phi(ID||k_i)$ k times (i.e., with all its own k keys and the ID it hears) to discover the shared key(s). In short, the procedure involves only k hashing operations and $O(1)$ message exchange between any two neighbor nodes. However, these challenge-response-like methods are still subject to selective node capture. The major difference here is that the selective node capture is only more meaningful than random node capture in sequential node capture mode. That is, in each step, the adversary identifies and captures the node with the fewest keys existing in the compromised key pool. While in "key indices notification" methods, the adversary can identify a set of nodes with the least key overlap and capture them concurrently.

References

[1] L. Eschenauer and V. D. Gligor, "A key-management scheme for distributed sensor networks," in *CCS'02: ACM conference on Computer and communications security*, New York, NY, 2002, pp. 41–47.

[2] R. Blom, "An optimal class of symmetric key generation system," in *Advanced in Cryptology - Eurocrypt'84, LNCS*, vol. 209, 1984, pp. 335–338.

[3] T. Matsumoto and H. Imai, "On the key predistribution systems: A practical solution to the key distribution problem," in *Advances in Sryptology - Crypto'87,*, 1987.

[4] C. S. R. Murthy and B. S. Manoj, *Ad Hoc Wireless Networks: Architectures and Protocols*. Prentice Hall Communications Engineering and Emerging Technologies Series, 2004.

[5] R. D. Pietro, L. V. Mancini, and A. Mei, "Random key-assignment for secure wireless sensor networks," in *SASN'03: ACM workshop on Security of ad hoc and sensor networks*, New York, NY, 2003, pp. 62–71.

[6] J. Kohl and B. Neuman, "The kerberos network authentication service (v5)," RFC1510, Sept. 1993.

[7] M. Bellare, A. Desai, D. Pointcheval, and P. Rogaway, "Relations among notions of security for public-key encryption schemes," in *Advances in Cryptology - CRYPTO'98: 18th Annual International Cryptology Conference*, Santa Barbara, CA, August 1998.

[8] H. Chan, A. Perrig, and D. Song, "Random key predistribution schemes for sensor networks," in *IEEE Symposium on Security and Privacy*, 2003.

[9] W. Du et al., "A key management scheme for wireless sensor networks using deployment knowledge," in *INFOCOM'04, Hong Kong*, Mar. 2004.

[10] J. Hwang and Y. Kim, "Revisiting random key pre-distribution schemes for wireless sensor networks," *ACM workshop on Security of ad hoc and sensor networks*, pp. 43–52, 2004.

[11] S. Zhu, S. Setia, and S. Jajodia, "LEAP: efficient security mechanisms for large-scale distributed sensor networks," in *CCS'03: ACM conference on Computer and communications security*, New York, NY, 2003, pp. 62–72.

[12] J. Lee and D. Stinson, "Deterministic key predistribution schemes for distributed sensor networks," *Selected Areas in Cryptography*, 2004.

[13] S. Camtepe and B. Yener, "Combinatorial design of key distribution mechanisms for wireless sensor networks," *European Symposium On Research in Computer Security (ESORICS'04)*, 2004.

[14] S. Zhu, S. Setia, S. Jajodia, and P. Ning, "An interleaved hop-by-hop authentication scheme for filtering of injected false data in sensor networks," *Security and Privacy, 2004. Proceedings. 2004 IEEE Symposium on*, pp. 259–271, 2004.

[15] S. A. Çamtepe and B. Yener, “Key distribution mechanisms for wireless sensor networks: a survey,” Rensselaer Polytechnic Institute, Computer Science Department, Tech. Rep. TR-05-07, Mar. 2005, available at http://www.cs.rpi.edu/research/pdf/05-07.pdf.

[16] S. Zhu, S. Xu, S. Setia, and S. Jajodia, “Establishing pairwise keys for secure communication in ad hoc networks: A probabilistic approach,” in *ICNP’03*, 2003.

[17] S. Seys and B. Preneel, “The wandering nodes: Key management for lower-power mobile ad hoc netowrks,” in *IEEE International Conference on Distributed Computing Systems Workshops, ICDCSW’05*, 2005.

[18] A. Howard, M. J. Mataric, and G. S. Sukhatme, “Mobile sensor network deployment using potential fields: A distributed, scalable solution to the area coverage problem,” in *Distributed Autonomous Robotic Systems*, 2002.

[19] R. Lee et al., “Architecture for protecting critical secrets in microprocessors,” in *International Symposium on Computer Architecture (ISCA 2005)*, June 2005, pp. 2–13.

[20] J. Dwoskin and R. Lee, “Hardware-rooted trust for secure key management and transient trust,” in *CCS’07: ACM conference on Computer and communications security*, 2007.

[21] W. Du, J. Deng, Y. S. Han, and P. K. Varshney, “A pairwise key pre-distribution scheme for wireless sensor networks,” in *CCS’03: ACM conference on Computer and communications security*, New York, NY, 2003, pp. 42–51.

[22] B. Parno, A. Perrig, and V. Gligor, “Distributed detection of node replication attacks in sensor networks,” in *Proceedings of the 2005 IEEE Symposium on Security and Privacy*, 2005.

[23] D. Liu and P. Ning, “Location-based pairwise key establishment for static sensor networks,” in *1st ACM Workshop on Security of Ad Hoc and Sensor Networks*, 2003.