Key Foisting and Key Stealing Attacks in Sensor Networks

Peng Wang
University of California, Riverside
wangpe@cs.ucr.edu

Chinya Ravishankar
University of California, Riverside
ravi@cs.ucr.edu

ABSTRACT

We show how to establish cryptographic keys in sensor networks where neither PKI nor a trusted third party exists. We use a "web-of-trust" model, establishing "path" keys using pairwise trust relationships between intermediaries sharing preloaded keys. We first show how to defeat current schemes with *key foisting*, a devastating novel attack not described in the literature. Foisting compromises 90% of the path keys, when only 10% of the sensors in the network are seized. We then present a two-way path-key establishment scheme, and a highest random weight based path-key establishment scheme to deal with key foisting, using mGKE as an illustrative example. Our schemes reduce the probability of successful key foisting to nearly zero even when 20% sensors are seized. Its overhead is affordable, and its resilience is excellent. We also discuss key foisting in general distributed systems.

1. INTRODUCTION

Sensors are widely deployed in traffic control, supply-chain management, theft prevention, patient monitoring, personal security, driver's licenses, passports, and so on. They are now present in nearly every device we encounter in our daily life. Sensors systems are evolving from networks of special-purpose devices deployed on demand, to becoming ubiquitous in our environment.

This trend is exemplified by the Internet of Things (IoT), a universal network of devices. The IoT is seen [2, 24] as a self-configuring global network infrastructure based on interoperable protocols, comprising physical and virtual nodes with identities, attributes, and intelligent interfaces, seamlessly integrated into a network. Its nodes will participate in business, information and social processes, interacting with themselves and the environment, and influencing the real world through actions, with or without human intervention.

1.1 Heterogeneity Complicates Security

Security is a big concern in such networks, but heterogeneity complicates the establishment of cryptographic keys. In large sensor networks, including IoT, sensors will have widely different configurations and hardware and software capabilities. They will also belong to different administrative domains, each with its own policies and protocols. Heterogeneity means that no single set of policies or protocols will work for all sensors.

Public-Key Infrastructures (PKIs) [1] can be an effective solution, but not all nodes in such a network may support public key protocols, or even subscribe to PKIs. For similar reasons, it is unlikely that any single third party will be sufficiently trusted to mediate symmetric pairwise key establishment between all nodes.

1.1.1 Sensor Groups and Webs of Trust

We note that it is natural to organize such large networks as groups, mirroring their structural, communication and trust relationships in the real world. Nodes in each organizational unit (floor, building, factory, vehicle, etc.) form a natural group for administrative purposes. Nodes in one group are more likely to communicate more with each other, and trust each other more than they might trust nodes from a different unit.

The last point is especially important, when neither PKI nor a globally trusted authority is available. Under these constraints, we are forced to exploit the trust relationships between nodes that naturally arise in a group-based organization. Such trust can be formalized as shared keys within and across groups at configuration time. This serves as the foundation for the subsequent dynamic establishment of trust (keys) between nodes.

A similar "web of trust" model is seen in PGP [22], and widely used for decentralized public-key discovery. Users maintain validated user-public key associations in the form of personal "key rings". if user Alice needs Carol's public key, and Bob is able to forward a key

he can cryptographically certify as Carol's, Alice can accept this key if she trusts Bob. When Alice only has partial trust in a set of users, she can accept a key if it is certified by a threshold number of users.

1.1.2 Assumptions and Threats

Our work does not use public keys, but does use pair-wise trust relationships. We will first present key foisting, a new attack that easily compromises web-of-trust models, whether they use public keys or symmetric keys. We will then describe a two-way key establishment protocol that addresses this attack.

We will show how to establish dynamic symmetric pairwise cryptographic keys in the scenario where trusted authorities or PKI may be available to some nodes, but not to all. Mutual trust must now be realized through pairwise shared symmetric keys between sensors. However, sensors lack enough memory to store all $O(n)$ pairwise keys for all other sensors in the system. Communication patterns are unknown in advance, so not all pairs of communicating sensors can share preloaded keys.

We will assume that we are dealing with wireless sensor networks (WSNs), since they are inherently insecure [6, 7, 9, 14, 19, 20]. However, our work applies equally to wired sensor networks. In Sec .9, we also discuss pairwise key establishment in general distributed networks.

1.1.3 Sensor Groups, Associations, and Agents

Current group-based schemes [4, 10, 11, 16, 17, 19] try to establish shared keys without trusted third parties or PKI, but we will see that they have serious flaws. Typically, a WSN with n nodes is organized into g groups with γ nodes each [19]. Each node pair within a group U is preloaded with a unique key. In Fig. 1a, each of $\{a_1, a_2, a_3, a_4\}$ holds a key for its neighbors in U. Also, $t > 1$ node pairs across each group pair (U, V) share preloaded keys. $(a_4, b_3), (a_2, b_1)$ are such pairs in Fig. 1a. Fig. 2 shows a 3-level group hierarchy. For a 2-level hierarchy with t agents across each group pair, each node holds only $\gamma - 1$ preloaded intra-group keys, and $t(g-1)/\gamma$ inter-group keys [19]. If $g = \gamma = \sqrt{n}$, each node holds only $O(\sqrt{n}) + O(t)$ keys, instead of the $O(n)$ pairwise keys required in a naive model.

Sensor pairs, such as (a_1, a_2) and (a_4, b_3) in Fig. 1a, that share preloaded keys are called *associated*. A sensor $s_i \in U$ sharing a key with a sensor $s_j \in V$ is an *agent* in U for V. Sensors not associated will establish *path keys* using agents as intermediaries. Fig. 1b shows typical path-key establishment in current schemes. a_3 establishes a path key with b_2, by forwarding it via agents a_4 and b_3. Hops $\langle a_3, a_4\rangle, \langle a_4, b_3\rangle$, and $\langle b_3, b_2\rangle$ forward encrypted messages, possibly over multiple radio hops. Decryption and re-encryption occurs at a_4 and b_3.

Preloaded keys are resilient, since they are compromised only when one of their owners is. However, compromising an intermediary compromises a path key.

1.1.4 General Distributed System

In mGKE, it is guaranteed that there are at most three cryptographic hops between two nodes not associated (Fig. 1a), and cryptographic paths between them are pre-defined. However, in a general distributed system, cryptographic paths between two nodes may not be pre-defined. There can be arbitrary number of cryptographic paths between two nodes, and each path can contain arbitrary number of cryptographic hops.

1.2 Key Stealing and Foisting Attacks

By seizing a sensor, the adversary gains both its preloaded keys, as well as all path keys it mediates. Current schemes [4, 5, 17, 19] recognize such attacks, which we call *key stealing* (**KS**). As in Fig. 3, **KS** permits eavesdropping and false data injection. Typically, **KS** allows the adversary to compromise 30% of the path keys by seizing about 10% of its sensors [17, 19].

In this paper, we introduce *key foisting* (**KF**), a novel attack which can compromise 90% of the path keys by seizing only 10% of the sensors. **KF** is devastating because end-to-end authentication is impossible during path-key establishment. Since there is no trusted third party, path key establishment must rely on on trusted intermediaries, who can only authenticate on a hop-by-hop basis. *End-to-end authentication requires end-to-end keys, but the very purpose of path-key establishment is to set up such keys between the end points.*

In key foisting, the adversary uses seized intermediaries to send fabricated path key establishment messages to fool other sensors to accept path keys generated by him. Such attacks are devastating and hard to detect. In current schemes, such as mGKE [19], PIKE [4], GP [17], only about 1% of the communication channels are secured via preloaded keys. The rest are secured by path keys. The adversary can compromise 90% of path keys by seizing a mere 10% of sensors. In effect, no content transmitted in the network is secure.

1.3 Our Contributions

We propose a new two-way path-key establishment scheme to address **KF**. This scheme allows recipients to verify the correctness of path keys, while in earlier schemes, recipients have no choice but to assume that the path keys are correct. We apply the two-way scheme in mGKE and the results show that the resilience against **KF** is greatly improved. We also propose a highest random weight (HRW) based path key establishment scheme. The resilience of our HRW based scheme is comparable to our two-way scheme while its communication overhead is much smaller.

We make the following contributions:

- We describe key foisting, and show that it is devastating against schemes such as [4, 17, 19].
- We present a novel two-way (2W) scheme that defeats key foisting attacks. This scheme is compatible with a variety of key management schemes.
- We present an HRW based scheme to deal with key foisting, with less communication overhead than the 2W scheme.
- We apply our 2W scheme and HRW based scheme to mGKE [19] as an example, and present a rigorous analysis of resilience. Similar analysis is possible for other schemes.
- We show how to perform key foisting in general distributed networks. Our 2W scheme and HRW based scheme can also be used to deal with key foisting in the general model.

Group-based key management schemes have the following advantages. First, sensors in the same group are more likely to communicate with each other. By giving sensors in the same group preloaded pairwise keys, we increase resilience to attacks and reduce overhead, since path keys between neighbors now requires local communication, unlike [5, 12]. Second, the two-way scheme can be combined with multipath reinforcement to resist hybrid attacks **KF-KS** (Section 7.1). Unlike [12] and [5], it is easy to find multiple disjoint key establishment paths between any two sensors in group-based schemes.

In this paper, related work appears in Sec. 2, and Sec. 3 presents an overview of key distribution, path key establishment, **KS** and **KF** attacks. Sec. 4 presents our two-way scheme and its use in mGKE. Sec. 5 presents our HRW based path key establishment scheme. Sec. 6 analyzes one- and two-way path key schemes. Sec. 7 analyzes replay attacks and hybrid **KF-KS** attacks against our two-way scheme. Sec. 8 analyzes the resilience of HRW based key establishment scheme against attacks. Sec. 9 analyzes key foisting in the general model. Sec. 10 concludes the paper.

2. RELATED WORK

Group-based key management [4, 19, 17] operates in three phases. In the predistribution phase, selected sensors pairs are assigned shared keys before deployment. In the key-setup phase, sensors are deployed, and discover neighbors and associations. In the path-key establishment phase, unassociated sensors establish path keys as needed, using agents.

In [12], each sensor s_i randomly selects an m-subset S_i of a key pool K. Sensors s_i, s_j can use any key from $S_i \cap S_j$ as their shared key. If $S_i \cap S_j = \emptyset$, they can establish path keys via intermediaries. In the q-composition scheme [5], two sensors may set up a key if they share at least q preloaded keys. [5] generates an ID pool and a pairwise key pool for IDs. A sensor randomly selects an ID from the ID pool, and is preloaded with a key matching its ID from the key pool.

Threshold-based key predistribution is proposed in [11] and [16]. Blom's key space scheme [3] is improved in [11] using multiple key spaces. The polynomial-based key-predistribution scheme is expanded in [16] using a polynomial pool instead of a single polynomial. This scheme uses a logical grid in which all sensors on a row or columns share a key. Sensors on different rows or columns establish path keys via agents. Another grid-based scheme appears in [4]. GP [17] uses a grid, placing sensors on each row or column into the same group. Among the schemes for intra-group key predistribution, using unique pairwise keys achieves the best resilience.

In the scheme of [19], sensors in the same group share preloaded pairwise keys, and path keys established via agents are very robust. Intra-group keys have perfect resilience against key stealing. KeEs [7] guarantees backward and forward key security for key compromise attacks, but fails catastrophically for node compromises.

The potential of multipath reinforcement [5] is not realized by current schemes. They require disjoint cryptographic paths to be found on-demand, an expensive task. A cryptographic path may include many agents in [5], multiplying the chances of compromise. [15] guarantees paths with at most one agent, but requires flooding, which is too expensive. Fault localization is the focus in [25]. Other schemes [6, 20] try to mitigate the impact of false data injection attacks on in-network aggregation.

3. ATTACK MODEL

We illustrate key predistribution and path key establishment using mGKE [19] as an example. mGKE divides a sensor network into groups. All sensor pairs within each group share pairwise keys (i.e., are *associated*). For any two groups G_1, G_2, mGKE guarantees that at least one $s_i \in G_1$ and $s_j \in G_2$ share preloaded keys. Such sensors in different groups but sharing preloaded keys are called *agents*.

Fig. 1a shows two groups, each containing four sensors. Real sensor networks may have hundreds of groups. All sensors within each group share preloaded pairwise keys. In Fig. 1a, (a_2, b_1) and (a_4, b_3) are the agent pairs between these groups.

In Fig. 1b, sensors a_3 and b_2 establish a path key via agents a_4 and b_3. Let $K_{a_3b_2}$ be the key between nodes a_3 and b_2, and $\langle M \| K_{a_ib_i} \rangle$ denote the message M encrypted with $K_{a_ib_i}$. To establish a path key with b_2, sensor a_3 picks a random value $K_{a_3b_2}$, and proceeds as follows (headers omitted for simplicity).

1. $a_3 \rightarrow a_4 : \langle (K_{a_3b_2}, a_3, b_2, G_v) \| K_{a_3a_4} \rangle$
2. $a_4 \rightarrow b_3 : \langle (K_{a_3b_2}, a_3, G_u, b_2) \| K_{a_4b_3} \rangle$
3. $b_3 \rightarrow b_2 : \langle (K_{a_3b_2}, a_3, G_u) \| K_{b_3b_2} \rangle$

Message (1), encrypted by a_3, may be relayed by several nodes before a_4 receives and decrypts it. Thus, $\langle a_3, a_4\rangle$, $\langle a_4, b_3\rangle$, $\langle b_3, b_2\rangle$ are not radio hops, but "cryptographic" hops between "active" encryption and decryption sites. A series of cryptographically active nodes mediating path keys, such as a_3, a_4, b_3, b_2 in Fig. 1b, is a *keypath*. The path key $K_{a_3b_2}$ is known to the end points a_3, b_2, but also to the agents a_4, b_3 that mediate the key. The adversary can get the key by seizing a_4 or b_3.

We assume the Yao-Dolev model [8, 14]. The adversary may record all traffic, but wishes to remain undetected. Preloaded keys have perfect resilience [19], so we focus on threats to path keys. Cryptanalysis yields individual keys, but can be mitigated, as in [7]. We assume *node seizures*, a greater threat. Seizure yields all keys in a node, including path keys it mediates, and permits insider attacks [14], such as the following.

In *key stealing* attacks, seized agents steal path keys they mediate [4, 5, 17, 19]. In Fig. 4a, agent a_3 is seized, and steals the path keys it mediates. a_2 is also seized, and can steal keys if used as agent. Keys mediated by a_1 are safe until it is seized. *Data injection* is a different attack, but also well-recognized [6, 20].

Redundancy can mitigate key stealing. A group pair G_u, G_v may have t agent pairs, each defining a keypath (Fig. 1a). A keypath is seized iff an agent within it is seized. Two sensors $s_i \in G_u$ and $s_j \in G_v$ can select any one of these t keypaths for key establishment, with probability $\frac{1}{t}$. An adversary who seizes c keypaths can seize this keypath with probability $\frac{c}{t}$. He succeeds with high probability only for high c, giving some protection against stealing.

3.1 Key Foisting: A Serious New Attack

Unfortunately, the literature has not recognized that fraudulent path keys can be forced on victims by faked path-key establishment requests from seized agents. Such *key foisting* (**KF**) may be seen as an impersonation-and-key-injection attack. Injection has been studied for fake data [6, 20], but not path keys. Superficially similar, Sybil attacks [9, 18] overwhelm reputation systems with fake identities.

The **KF** attack works as follows. Let $K_{s_x s_y}$ denote the preloaded key shared by some two nodes s_x and s_y.

1. Seize $s_a \in G_u$. Let s_a be an agent in G_u for groups $G_{v_1}, \ldots G_{v_k}$. Identify the pairs (s_i, s_j), $s_i \in G_u, s_j \in G_{v_l}, 1 \leq l \leq k$ served by s_a.
2. Target such a pair (s_i, s_j). Fabricate a key $K^*_{s_i s_j}$.
3. Fabricate a message that s_j wishes to establish key $K^*_{s_i s_j}$ with s_i, encrypt message with $K_{s_a s_i}$, and send to s_i. Now, s_i is tricked into accepting $K^*_{s_i s_j}$.
4. Let s_j belong to G_{v_l}, and let s_a be associated with agent $s_b \in G_{v_l}$. Fabricate a message claiming s_i wishes to establish key $K^*_{s_i s_j}$ with s_j. Encrypt this message with $K_{s_a s_b}$, and send it to s_b.
5. s_b accepts s_a's message, decrypts and re-encrypts it with $K_{s_b s_j}$, and forwards it to s_j, who is tricked into believing that the request originated with s_i.
6. s_i and s_j have been fooled into using $K^*_{s_j s_j}$.

Foisting defeats agent redundancy. *Seizing a single agent s_a suffices to foist fake path keys on all sensor pairs across all groups s_a serves.* In Fig. 4b, agent a_2 is seized, and sends fake path-key establishment requests to all sensor pairs it serves. This attack succeeds because authentication is hop-by-hop, not end-to-end.

Key foisting is feasible whenever path keys are established [4, 17], not just in group-based methods. Our analysis shows that current schemes are all vulnerable.

3.1.1 Foisting When Public Keys are Used

KF can be addressed if PKI is available, since sensors can verify public-key certificates. However, as we have noted, this assumption is not realistic in very large and heterogeneous networks. **KF** is possible in public cryptography based key establishment schemes like PGP, when no certification authority is available. Let Alice and Bob have public keys P_A and P_B, and Carol have secret key S_C. Alice and Bob both trust Carol. They both know Carol's public key, but not each other's public key. The adversary compromises the secure channel between Alice and Bob as follows:

1. Seize Carol and her secret key S_C.
2. Fabricate two public keys P^*_A and P^*_B.
3. Claim that Alice wishes to establish a secure channel with Bob, and send the message $\langle (P^*_A) \| S_C \rangle$ to Bob. Now, Bob is tricked into accepting P^*_A since it is signed by S_C.
4. Claim that Bob wishes to establish a secure channel with Alice, and send $\langle (P^*_B) \| S_C \rangle$ to Alice, who accepts P^*_B, since it is signed with S_C.

The stage is now set for a man-in-the-middle attack.

3.2 Setting Realistic Goals

The following theorem helps clarify our goals.

THEOREM 1. *Key stealing cannot be prevented if path keys are established using hop-by-hop intermediaries.*

Proof: If intermediaries $s_{i_1}, s_{i_2}, \ldots, s_{i_r}$ help establish path key K, this key is known to each of them. The adversary can steal K by seizing any of these nodes. □

Key stealing cannot be prevented if PKI or a trusted authority is not available. We will present a scheme that prevents **KF** and strongly resists **KS**.

4. TWO-WAY PATH-KEY ESTABLISHMENT

We first briefly introduce mGKE[19], then present the two-way path-key establishment and its use in mGKE.

4.1 mGKE

mGKE preloads a unique key into each pair of sensors in the same group, so its intra-group resilience is perfect. In addition, t sensors pairs from $G_u \times G_v$ are preloaded with unique pairwise keys. Other sensors pairs use these agents to establish path keys (Fig. 1a). Each group contains n_s sensors, and there are n_g groups in the network. Agents in G_u for G_v are selected using the formula

$$F_{uv}(i) = (t(v-1)+i) \mod n_s, \quad (1)$$

where $F_{uv}(i)$ is ID of the i^{th} agent in G_u for G_v, t is the number of agents between groups. mGKE minimizes memory overhead at the sensors, since agent IDs are found via Eqn. (1), rather than stored. Let $A_{uv} = \{a^1_{uv}, \ldots a^t_{uv}\}$ denote this set of agents in G_u for G_v.

4.2 Two-Way Key Establishment (2W)

We propose two-way key establishment (2W) to deal with key foisting. All schemes to date have used one-way key establishment (1W).

In the 1W scheme, let $s_j \in G_v$ receive the key-establishment request $\langle (K_{s_i s_j}, G_u, s_i) \| K_{s_j, a^1_{vu}} \rangle$ from an agent $a^1_{vu} \in G_v$, which is associated with $a^1_{uv} \in G_u$. It is now impossible for s_j to know whether $K_{s_i s_j}$ is legitimate or was faked by a compromised a^1_{uv} or a^1_{vu}.

In contrast, in our 2W scheme, s_i creates and sends a *forward half* $K^{\rightarrow}_{s_i s_j}$ of the path key to s_j, which responds with a *reverse half* $K^{\leftarrow}_{s_i s_j}$ via a disjoint path. s_i and s_j compute the path key as $K_{s_i s_j} = K^{\rightarrow}_{s_i s_j} \oplus K^{\leftarrow}_{s_i s_j}$. They can both trust $K_{s_i s_j}$ since each generated a part of it.

4.2.1 Forward Phase of 2W Key Establishment:

$s_i \in G_u$ finds the agent set A_{uv} for G_v via Eqn. (1), and randomly selects an agent a^x_{uv}. Next, s_i encrypts a random $K^{\rightarrow}_{s_i s_j}$ with $K_{s_i, a^x_{uv}}$, sending it to a^x_{uv} in message $\langle (K^{\rightarrow}_{s_i s_j}, G_v, s_i, s_j) \| K_{s_i, a^x_{uv}} \rangle$. a^x_{uv} recovers $K^{\rightarrow}_{s_i s_j}$ and sends it encrypted to a^x_{vu} as $\langle K^{\rightarrow}_{s_i s_j}, G_u, s_i, s_j \| K_{a^x_{uv}, a^x_{vu}} \rangle$. s_j recovers $K^{\rightarrow}_{s_i s_j}$, and begins the reverse phase.

4.2.2 Reverse Phase of 2W Key Establishment:

As shown in Fig. 1a, all keypaths between two sensors are disjoint. To ensure disjoint keypaths, s_j drops the agents used in the forward phase, and picks an agent a^y_{vu} from among the remaining $t-1$ agents in A_{vu}. s_j now picks a random values $K^{\leftarrow}_{s_i s_j}$ representing its half of the path key. s_j sends this half to s_i, exactly mirroring s_i's actions in the forward phase, but using the agents a^y_{vu} instead. The agent forwards $K^{\leftarrow}_{s_i s_j}$ to its peer agent in G_u, who forwards it to s_i. At the end of the reverse phase, s_i and s_j both have $K^{\rightarrow}_{s_i s_j}$ and $K^{\leftarrow}_{s_i s_j}$ and generate the path key $K_{s_i s_j} = K^{\rightarrow}_{s_i s_j} \oplus K^{\leftarrow}_{s_i s_j}$. We require $t \geq 2$.

Fig. 5 shows 2W path key establishment in mGKE. To establish a path key with b_2, sensor a_3 picks a random value $K^{\rightarrow}_{a_3 b_2}$, and proceeds as follows (message headers are omitted for simplicity).

1. $a_3 \rightarrow a_4 : \langle (K^{\rightarrow}_{a_3 b_2}, a_3, b_2, G_v) \| K_{a_3 a_4} \rangle$
2. $a_4 \rightarrow b_3 : \langle (K^{\rightarrow}_{a_3 b_2}, a_3, G_u, b_2) \| K_{a_4 b_3} \rangle$
3. $b_3 \rightarrow b_2 : \langle (K^{\rightarrow}_{a_3 b_2}, a_3, G_u) \| K_{b_3 b_2} \rangle$
4. $b_2 \rightarrow b_1 : \langle (K^{\leftarrow}_{a_3 b_2}, a_3, b_2, G_u) \| K_{b_1 b_2} \rangle$
5. $b_1 \rightarrow a_2 : \langle (K^{\leftarrow}_{a_3 b_2}, b_2, G_v, a_3) \| K_{a_2 b_1} \rangle$
6. $a_2 \rightarrow a_3 : \langle (K^{\leftarrow}_{a_3 b_2}, b_2, G_v) \| K_{a_2 a_3} \rangle$

4.3 k-Path Reinforcement (k-PR)

A great strength of a scheme like mGKE is its support for multipath reinforcement. mGKE with our 2W scheme defeats foisting. However, as Theorem 1 shows, key stealing is always possible. Using k-path reinforcement [5] also adds resilience against key stealing. We will find k-path reinforcement useful in increasing resilience to a mixed attack described in Section 7.

In k-path reinforcement [5], a key is cryptographically divided into shares, and sent along k node-disjoint paths to the destination, where it is reconstituted from the shares. The adversary must compromise all k of these paths to steal the key.

Using k-PR in mGKE is much more efficient than in other schemes. Randomized methods like RKP [5] only make probabilistic guarantees about network connectivity, without assuring that node degrees are at least k. Nodes of lower degree cannot use k-path key reinforcement. Even when k disjoint paths exist, they are expensive to find.

In contrast, k-PR works well in mGKE, where keypaths are all agent-disjoint (Fig. 1a), so it suffices to pick any k keypaths. In mGKE, keypaths have two or fewer agents, but paths in [5] may have any number of them. Agents encrypt/decrypt messages, so k-PR security drops as the number of agents per path grows in [5]. Finally, mGKE initiators can find agents from Eqn. (1), and send path key messages via standard routing, but initiators in [5] must themselves discover paths and select agents from them. Even worse, [15] uses broadcasting and flooding to find agents. Group-based schemes like mGKE have several desirable properties:

- $k \leq t$ agent-disjoint keypaths exist between any two nodes, as t exist between any two groups.
- $s_i \in G_u$ gets k agent-disjoint keypaths to $s_j \in G_v$ just by selecting k agents from the t agents in A_{uv}.
- Intra-group communication overhead is far smaller than inter-group communication overhead [19].

In Sec. 7.2, we show how to use aggregation with k-PR to greatly reduce the communication overhead.

5. HRW BASED PATH KEY ESTABLISHMENT

In this section, we first briefly introduce highest random weight (HRW), then present HRW based multipath key reinforcement ((k, I)-**HPR**).

5.1 Highest Random Weight

The Highest Random Weight algorithm was introduced in [23] to achieve distributed consensus on object-server mappings. It uses a hash function $h : \{0,1\}^* \rightarrow \mathbb{Z}_p$ as follows. Given an object name o_i and N servers $\mathbf{S} = \{\alpha_1, \ldots, \alpha_N\}$, HRW first computes $h(o_i \| \alpha_1), \ldots, h(o_i \| \alpha_N)$ where $\|$ means concatenation, and selects $n \leq N$ servers $\alpha_{i_1}, \ldots, \alpha_{i_n} \subseteq \mathbf{S}$ having the highest hash values to serve the object.

HRW ensures that each server set is selected to serve a given object with the same probability. Each object is always mapped to the same server set, and this mapping can be computed locally by each client. Most importantly, HRW minimizes disruption in the event of server failure. If a single server is to be chosen, we set $n = 1$.

5.2 Highest Random Weight based path key establishment

We state how to use HRW in path key establishment. Let the $s_i \in G_u$ denote the initiator and $s_j \in G_v$ denote the recipient. Let $\{(a^1_{uv}, a^1_{vu}), \ldots, (a^t_{uv}, a^t_{vu})\}$ denote the t agent pairs between G_u and G_v. The k agent pairs used to establish a path key between s_i and s_j are selected as follows. s_i first calculates the hash value $h(a^l_{uv} || a^l_{vu})$ for each agent pair, and select the k pairs with the highest hash values to establish the path key. Let $\{(a^{(1)}_{uv}, a^{(1)}_{vu}), ..., (a^{(k)}_{uv}, a^{(k)}_{vu})\}$ denote the k agent pairs selected. s_i generates the k shares of the path key $K_{s_i s_j}$, and uses k-PR to send the k shares to s_j.

s_j uses HRW to verify $K_{s_i s_j}$. s_j calculates the hash value for each agent pair participated in the path key establishment, and checks whether the k agent pairs are those with highest hash values. If so, s_j accepts $K_{s_i s_j}$, otherwise rejects it.

HRW based path key establishment can improve the resilience against **KF**. Unless the adversary seizes the k agent pairs with the highest random values, $K_{s_i s_j}$ is safe since s_j will reject $K_{s_i s_j}$ not sent by these k agent pairs. Meanwhile, its communication overhead is just half of that of 2W k-PR.

5.3 Fault Tolerant

Agents may not always be available due to physical damage or power outage. We use (k, I) secret sharing scheme in path key establishment to mitigate the affection of faulty agents. In (k, I) secret sharing, a secret is divided into I pieces, and can be recovered by any k in I pieces. In path key establishment, I agent pairs with the highest hash values are selected to establish the path key for s_i and s_j. s_i divides the path key using (k, I) secret sharing, and sends each piece via an agent pair to s_j. In the verification phase, s_i accepts the key if the path key is sent by k agent pairs in the top I agent pairs with highest hash values.

6. ANALYSIS OF RESILIENCE

We now show how devastating the **KF** attack is by analyzing the resilience of mGKE with 1W against **KF**. When a node is seized, all its keys are lost. Since it may have mediated path keys, keys for unseized nodes may be affected. The *resilience* of a path key scheme is hence judged [19] by *the rate at which keys (or keypaths) between unseized sensors are lost, as sensors are seized.*

Let mGKE (1W k-PR) denote mGKE using 1W path key establishment and k-path key reinforcement, and mGKE (2W k-PR) denote mGKE using 2W path key establishment and k-path key reinforcement. In mGKE (2W k-PR), k keypaths are used by both the initiator and the recipient, so that $2k$ keypaths are used in all for path key establishment. In k-path reinforcement [5], a path key is divided into k shares. The initiator sends the k shares via k agent-disjoint keypaths. To steal the path key, the adversary must now seize an agent in each path. k-path key reinforcement improves resilience at the cost of communication overhead.

DEFINITION 1. *A set of keypaths $\{p_1 \ldots, p_k\}$ used in k-path reinforcement is a k-keypath.*

To compare the resilience against **KF** and **KS**, we first analyze the resilience of mGKE (1W k-PR) against both attacks. We will show that the resilience against **KF** is very much poorer than the resilience against **KS**.

Three cases arise when $K_{s_i s_j}$ is a path key for $s_i \in G_u, s_j \in G_v$:

1. neither s_i nor s_j is an agent for pair (G_u, G_v),
2. one of s_i or s_j is an agent for (G_u, G_v), or
3. both s_i and s_j are agents for (G_u, G_v).

Let $\mathbf{K_{s_i s_j}}$ denote the event that $K_{s_i s_j}$ is a path key and let $\mathbf{K^q_{ij}}$ denote that $K_{s_i s_j}$ is a path key matching case q above. Let $\mathbf{c^{\langle ij \rangle}}$ be the event that c nodes are seized in all, but neither s_i nor s_j is. Let $\mathbf{b}$ denote that b of $2t$ agents for (G_u, G_v) are seized. Let $\widehat{\mathbf{K}}_{\mathbf{s_i s_j}}$ denote that the key $K_{s_i s_j}$ between s_i and s_j is stolen. Let $\mathbf{b^{\langle ij \rangle}}$ denote that b agents are seized, but not s_i or s_j. We are interested in determining

$$\Pr\left[\widehat{\mathbf{K}}_{\mathbf{s_i s_j}} \mid \mathbf{c^{\langle ij \rangle}} \wedge \mathbf{K_{s_i s_j}}\right] = \sum_{q=1}^{3} \sum_{b=1}^{2t} \left(P_{\widehat{K}|b} P_{b|c} P_q \right) \quad (2)$$

where $P_{\widehat{K}|b} = \Pr[\widehat{\mathbf{K}}_{\mathbf{s_i s_j}} \mid \mathbf{b} \wedge \mathbf{c^{\langle ij \rangle}} \wedge \mathbf{K^q_{ij}}]$, $P_q = \Pr[\mathbf{K^q_{ij}}]$, and $P_{b|c} = \Pr[\mathbf{b} \mid \mathbf{c^{\langle ij \rangle}} \wedge \mathbf{K^q_{ij}}]$.

Each group pair has t agent pairs, and $n_s^2 - t$ path keys. Of these, $(n_s - t)^2$ path keys are of type-1, $2t(n_s - t)$ of type-2, and $t(t-1)$ of type-3. Clearly, $\Pr[\mathbf{K^1_{s_i s_j}}] = \frac{(n_s-t)^2}{n_s^2-t}$, $\Pr[\mathbf{K^2_{s_i s_j}}] = \frac{2t(n_s-t)}{n_s^2-t}$, and $\Pr[\mathbf{K^3_{s_i s_j}}] = \frac{t(t-1)}{n_s^2-t}$. For simplicity, we only analyze the resilience of type-1 path keys. The resilience of type-2 and type-3 path keys can be easily obtained using the same method.

6.1 mGKE (1W k-PR) Key-Stealing Resilience

With Eqn. (2) in mind, the probability of event $\mathbf{b}$ with c seized sensors, excluding s_i and s_j is found as follows. For type-1 path keys, we can seize b of $2t$ agents in $\binom{2t}{b}$ ways, and seize $c-b$ sensors from n sensors, except for s_i, s_j and $2t$ agents, in $\binom{n-2-2t}{c-b}$ ways. Hence,

$$\Pr\left[\mathbf{b} \mid \mathbf{c^{\langle ij \rangle}} \wedge \mathbf{K^1_{s_i s_j}}\right] = \frac{\binom{n-2-2t}{c-b}\binom{2t}{b}}{\binom{n}{c}} \tag{3}$$

For type-1 path keys, if $b \leq k-1$, the adversary can steal no keypaths. If $b \geq 2t-1$, all keypaths are stolen. Define the ranges $R_1 = [0, k-1]$, $R_2 = [k, 2(t-1)]$, and $R_3 = [2t-1, 2t]$. Now,

$$\Pr\left[\widehat{\mathbf{K}}_{\mathbf{s_i s_j}} \mid \mathbf{b^{\langle ij \rangle}} \wedge \mathbf{K^1_{s_i s_j}}\right] = \begin{cases} 0 & b \in R_1 \\ g_1 & b \in R_2 \\ 1 & b \in R_3 \end{cases} \tag{4}$$

$$g_1 = \sum_{l=\lceil \frac{b}{2} \rceil}^{\min(t,b)} \frac{\binom{t}{b-l}\binom{t-b+l}{2l-b}\binom{l}{k}}{\binom{2t}{b}\binom{t}{k}} \cdot 2^{2l-b} \tag{5}$$

In Eqn. (5), $\binom{l}{k}/\binom{t}{k}$ is the probability that the k-keypath used by s_i and s_j is seized, when l keypaths are seized. $\frac{\binom{t}{b-l}\binom{t-b+l}{2l-b}}{\binom{2t}{b}} \times 2^{2l-b}$ is the probability that l keypaths are seized, when b agents are seized.

Eqns. (3–5) give the probabilities Eqn. (2) needs for type-1 path keys. Analysis for type-2 and type-3 keys is similar. Our analysis matches simulation, and resilience is excellent (Fig. 6). We simulate a sensor network with 10000 nodes under the mGKE (1W k-PR) scheme. Let c denote the number seized sensors, N_c denote the total number of path keys of $n-c$ unseized sensors and N_c^f denote the number of compromised path key when c sensors are seized. We use the ratio $\frac{N_c^f}{N_c}$ as the probability of successful key stealing for various c. All simulations in this paper were conducted in this manner. With even 20% sensors seized, the chances that a given pathkey is stolen are under 5% for 3-PR, and under 1% for 5-PR. We see that k-path key reinforcement is very effective in dealing with **KS**. k-PR enhances the resilience to **KS** for methods other than mGKE as well [19, 4, 17]. However, we will now show that no 1W scheme can resist **KF**, despite the use of k-PR.

6.2 mGKE (1W k-PR) Key-Foisting Resilience

We now present the first-ever analysis of foisting. We show that all 1W schemes [19, 4, 17] perform poorly against **KF**.

For type-1 path keys,

$$\Pr\left[\widehat{\mathbf{K}}_{\mathbf{s_i s_j}} \mid \mathbf{b^{\langle ij \rangle}} \wedge \mathbf{K^1_{s_i s_j}}\right] = \begin{cases} 0 & 0 \leq b \leq k-1 \\ g_2 & k \leq b \leq 2t \end{cases} \tag{6}$$

$$g_2 = \sum_{l=\max(\lceil \frac{b}{2} \rceil, k)}^{\min(b,t)} 2^{2l-b} \frac{\binom{t}{b-l}\binom{t-b+l}{2l-b}}{\binom{2t}{b}} \tag{7}$$

In Eqns. (6, 7), all keypaths are secure when $b \leq k-1$. Otherwise, we can choose l agent pairs from $2t$ agents in $2^{2l-b}\binom{t}{b-l}\binom{t-b+l}{2l-b}/\binom{2t}{b}$ ways. Compared with Eqn. (4), $\frac{\binom{l}{k}}{\binom{t}{k}}$ is missing because all path keys can be foisted with any k keypaths seized.

We now have the probabilities needed in Eqn. (2). Eqns. (6) yield $P_{\widehat{K}|b}$, and Eqns. (3), (4) yield P_q and $P_{b|c}$. Fig. 7 shows the probability that a given path key in mGKE (1W) has in fact been foisted, as per Eqns. (6–7) and simulation, for $k = 1, 10, 20$. Our analysis matches simulations perfectly. Comparing the resilience shown in Fig. 6 and Fig. 7, it is clear that **KF** is much more devastating than **KS** and simply using multipath reinforcement cannot improve the resilience much.

An analysis of PIKE, GP to stealing is given in [19], but no analysis for foisting has appeared. Fig. 8 shows our simulation results of foisting resilience for PIKE-2D and GP (unique pairwise keys), with $n_g = n_s = 100$, $t = 10$. PIKE's good showing is meaningless, given its poor resilience to stealing [19]. GP performs the worst, as its groups share too many agents.

6.3 mGKE (2W 1-PR) Resilience to Foisting

We analyze mGKE (2W 1-PR) performance, based on mGKE (1W). The following simple lemma is useful.

LEMMA 1. *A keypath between G_u and G_v is compromised either in both directions, or not at all.*

Proof: A keypath is compromised iff one or more agents in it are. Agents are indifferent to message direction. ■

We next show that mGKE (2W 1-PR) is immune to foisting.

THEOREM 2. *mGKE (2W k-PR) is immune to key foisting if $k > 0$, no matter how many nodes are seized.*

Proof: We assume that the adversary has knowledge of data local to any node *if and only if* he has seized the node. We yield him the maximum advantage, setting $k = 1$. Now, let him seize all sensors in G_u and G_v except s_i, s_j.

Assume the adversary foists a key $K^*_{s_i s_j}$ on s_i and s_j, so that neither s_i nor s_j functioned as initiator. By

the 2W algorithm (Sec. 4.2), s_i must have received a share f_i^* from the adversary, generated a random g_i, and computed a key locally as $K^i_{s_is_j} = f_i^* \oplus g_i$. Similarly, s_j must have computed $K^j_{s_is_j} = f_j^* \oplus g_j$, using the locally generated random value g_j. Since the adversary foisted the key $K^*_{s_is_j}$ successfully, $K^i_{s_is_j} = K^j_{s_is_j} = K^*_{s_is_j}$.

Since the adversary knows $K^*_{s_is_j}$, f_i^* and f_j^*, he can compute $g_i = K^*_{s_is_j} \oplus f_i^*$ and $g_j = K^*_{s_is_j} \oplus f_j^*$. However, g_i and g_j were randomly generated local values, which he can access only if he controls both s_i and s_j. This contradicts our assumption that he controls neither. ■

To make mGKE immune to **KF**, it suffices to use 2W path key establishment. Multipath key reinforcement is not required to guard against foisting. Other schemes, such as PIKE and GP can also adopt the 2W path key establishment to guard against **KF**.

7. REPLAYS, KEY FOISTING, AND MAN-IN-THE-MIDDLE ATTACKS

Current 1W schemes do not guarantee message freshness, and are vulnerable to replays. Let a path key $K_{s_is_j}$ established at time t_1 over k keypaths be compromised at time $t_2 > t_1$. In a 1W scheme, recording the inter-group path-key establishment messages at time t_1 allows the adversary to replay them at time $t_3 \geq t_2$, and foist $K_{s_is_j}$ on s_j.

THEOREM 3. *mGKE (2W) is immune to replays.*

Proof: Exactly as for Theorem 2. ■

Keys cannot be directly foisted in 2W schemes since the adversary cannot control the key half generated by receiver s_j. **KS** attacks remain viable (see Theorem 1), as is the following hybrid attack, when a very large number of nodes are compromised. mGKE (2W k-PR) continues to show excellent resilience.

7.1 Hybrid (KF-KS) Attacks

The adversary can combine **KF** with **KS** to compromise security, by creating separate keys with a pair of sensors and interposing himself in between. He must control enough agents in each group to control all key paths with high probability. We will show that **KF-KS** is no worse for mGKE (2W k-PR) than simple **KS**.

In Fig. 9, the adversary has seized agents $a_2, a_3 \in G_u$, $b_1 \in G_v$, and attacks $s_i \in G_u$ and $s_j \in G_v$ as follows.

1. Fabricate a forward half $K^{*\rightarrow}_{s_js_i}$. Fabricate a message that s_j wishes to establish key with s_i. Encrypt message with $K_{a_3s_i}$, and send to s_i. Now, s_i is tricked into accepting $K^{*\rightarrow}_{s_js_i}$.
2. Fabricate another forward half $K^{*\rightarrow}_{s_is_j}$. Fabricate a message that s_i wishes to establish key with s_j. Encrypt message with $K_{a_3b_3}$, and send to b_3.
3. b_3 will forward the forward half to s_j. Now, s_j is tricked into accepting $K^{*\rightarrow}_{s_is_j}$.
4. s_i generates its reverse half $K^{\leftarrow}_{s_js_i}$ and sends it to s_2 via agent a_2. If the adversary has seized a_2, he can steal this reverse half, and also suppress the message. He now computes the path key $K^{*\rightarrow}_{s_js_i} \oplus K^{\leftarrow}_{s_js_i}$, which is used by s_i as the path key for s_j.
5. s_j generates its reverse half: $K^{\leftarrow}_{s_is_j}$ and sends it to s_i via agent b_1. If the adversary has seized b_1, he can steal this reverse half and also suppress the message. He now computes the path key $K^{*\rightarrow}_{s_is_j} \oplus K^{\leftarrow}_{s_is_j}$, which is used by s_j as the path key for s_i.

The adversary can now mount a Man-in-the-Middle attack between s_i and s_j:

1. Send a false message $\langle (M_1) \| K^{*\rightarrow}_{s_is_j} \oplus K^{\leftarrow}_{s_is_j} \rangle$ to s_j. M_1 is encrypted with a key that s_j accepts, so s_j will accept this message.
2. When s_j responds to s_i with $\langle (M_2) \| K^{*\rightarrow}_{s_is_j} \oplus K^{\leftarrow}_{s_is_j} \rangle$, seize and suppress this message. Now decrypt the message, tamper with it, encrypt it using s_i's key $\langle (M_2') \| K^{*\rightarrow}_{s_js_i} \oplus K^{\leftarrow}_{s_js_i} \rangle$, and send the message to s_i. s_i will also accept the message.

We will now show that **KF-KS** *is no more effective against mGKE (2W k-PR) than a simple **KS** attack.*

In mGKE (2W k-PR), k keypaths are used by both the initiator and the recipient, so that $2k$ keypaths are used in all. With l keypaths seized, the chances that all shares of r_i and r_j are stolen are $\left[\frac{\binom{l-k}{k}}{\binom{t-k}{k}}\right]^2$. Let $\widehat{\mathbf{M}}$ denote success of a **KF-KS** attack. For type-1 keys,

$$\Pr\left[\widehat{\mathbf{M}} \mid \mathbf{b}^{\langle \mathbf{ij} \rangle} \wedge \mathbf{K^1_{s_is_j}}\right] = \begin{cases} 0 & b \in [0, 2k-1] \\ g_3 & b \in [2k, 2t] \end{cases}, \quad (8)$$

$$g_3 = \sum_{l=\max(2k, \lceil \frac{b}{2} \rceil)}^{\min(t,b)} 2^{2l-b} \frac{\binom{t}{l}\binom{l}{2l-b}}{\binom{2t}{b}} \left[\frac{\binom{l-k}{k}}{\binom{t-k}{k}}\right]^2. \quad (9)$$

Analysis of type-2 and 3 keys is similar.

We get the resilience by using $P_{\widehat{M}|b}$ in Eqn. (8) to replace $P_{\widehat{K}|b}$ in Equation 2. Fig. 10 shows mGKE (2W k-PR)'s excellent resilience to **KF-KS** attacks, which succeed less than 12% of the time even with $k = 1$ and 20% of the sensors seized. mGKE (2W k-PR) has nearly perfect resilience against **KF-KS** even with $k \geq 3$ and 20% of sensors seized. It outperforms the original multipath reinforcement significantly (Fig. 7, 8). **KF-KS** is no more effective than **KS** (Fig. 6). Fig. 11 shows the resilience of mGKE (2W k-PR) against **KF-KS** in different settings of t and k.

7.2 Aggregation to Reduce k-PR Overhead

As shown in Fig. 10, the resilience of mGKE (2W 1-PR) against **KF-KS** is excellent. However, we can use k-PR to improve the resilience if a higher standard of

security is required, say, that **KF-KS** succeed less than 5% of the time with 20% sensors seized.

Unfortunately, the default implementation of k-PR incurs k times the communication overhead of 1-PR. We now present k-PR with Aggregation, which makes k-PR efficient in mGKE. We can drop the **KF-KS** 's success rate below 1% in mGKE, at little or no additional cost, by choosing $k \geq 3$.

In k-PR with aggregation, instead of sending each share from G_u to G_v as a separate inter-group message, we collect all shares at one agent in G_u, and send them to G_v in one inter-group message (Fig. 12b).

$s_i \in G_u$ finds the agent set A_{uv} for G_v via Eqn. (1) and selects k agents $\{a_{uv}^1, a_{uv}^2, ..., a_{uv}^k\}$ at random. Next, s_i generates k random shares $[K_{s_is_j}^{\rightarrow}\backslash 1], \ldots, [K_{s_is_j}^{\rightarrow}\backslash k]$ so the "forward" half is $K_{s_is_j}^{\rightarrow} = [K_{s_is_j}^{\rightarrow}\backslash 1] \oplus \ldots \oplus [K_{s_is_j}^{\rightarrow}\backslash k]$. Let message $\langle [K_{s_is_j}^{\rightarrow}\backslash p] \| K_{s_i,a_{uv}^p} \rangle$ be denoted by L_{up}, and $\langle [K_{s_is_j}^{\rightarrow}\backslash p] \| K_{a_{uv}^p a_{vu}^p} \rangle$ be denoted by I_p. s_i encrypts and sends all shares to the first agent a_{uv}^1 in a single message.

$$s_i \rightarrow a_{uv}^1 : \langle (L_{u1}, L_{u2}, \ldots, L_{uk},\, G_v, s_i, s_j,\\ a_{uv}^2, a_{uv}^3, \ldots, a_{uv}^k) \,\|K_{s_i,a_{uv}^1} \rangle$$

a_{uv}^1 decrypts this, extracts $[K_{s_is_j}^{\rightarrow}\backslash 1]$, re-encrypting it to get I_1. It now sends the following message to a_{uv}^2

$$a_{uv}^1 \rightarrow a_{uv}^2 : \langle (I_1, L_{u2}, \ldots, L_{uk},\, G_v, s_i, s_j,\\ a_{vu}^1, a_{uv}^3, \ldots, a_{uv}^k) \| K_{a_{uv}^1 a_{uv}^2} \rangle$$

Each agent a_{uv}^p repeats this process, forwarding it to the next agent in the list as an intra-group message. The last agent a_{uv}^k sends all shares to its peer a_{vu}^k in G_v:

$$a_{uv}^k \rightarrow a_{vu}^k : \langle (I_1, I_2, \ldots, I_k,\, G_u, s_i, s_j,\\ a_{vu}^1, a_{vu}^2, \ldots, a_{vu}^{k-1}) \| K_{a_{uv}^k a_{vu}^k} \rangle$$

Agent a_{vu}^k now extracts $[K_{s_is_j}^{\rightarrow}\backslash k]$, re-encrypts it with $K_{a_{vu}^k s_j}$ to get L_{vk}, and sends $L_{vk}, I_1, I_2, \ldots, I_{k-1}$ to a_{vu}^{k-1}, mirroring the forwarding in G_u. Such forwarding continues within G_v, until the message:

$$a_{vu}^1 \rightarrow s_j : \langle (L_{v1}, L_{v2}, \ldots, L_{vk},\, G_u, s_i, s_j,\\ a_{vu}^2, a_{vu}^3, \ldots, a_{vu}^k) \,\|K_{a_{vu}^1 s_j} \rangle.$$

s_j recovers all shares, computes $K_{s_is_j}^{\rightarrow}$, and begins the reverse phase which mirrors the forward phase.

7.2.1 Energy Savings of k-PR with Aggregation

Consider a $D \times D$ region with n_g groups. A group has extent $d \times d$, for $d = \frac{D}{\sqrt{n_g}}$. The average distance between two random points in a unit square [13] is about 0.52, so the average distance between two sensors in the network is $\delta \approx 0.52D$, and $\delta_{uu} \approx 0.52d$ if they are both in G_u. Let δ_{uv} be the average distance between a sensor in G_u and one in G_v. Let p_{uu} be the probability that two randomly chosen sensors are in the same group, and p_{uv} be the probability that they are not. Clearly, $0.52D = p_{uu}0.52d + p_{uv}\delta_{uv}$. Using $p_{uu} = \frac{1}{n_g}$, $p_{uv} = \frac{n_g - 1}{n_g}$, we obtain

$$\delta_{uv} = \frac{0.52n_g^{\frac{3}{2}} - 0.52}{n_g - 1} d \tag{10}$$

Let the expected number of hops to transmit a message a unit distance be γ. The expected hop count between two sensors is $h_u = \frac{\delta_{uu}}{\gamma}$ if they are in the same group and $h_{uv} = \frac{\delta_{uv}}{\gamma}$ otherwise. Let $e(L)$ be the expected energy consumed per hop to send and receive a packet of length L. A length-L message from $s_i \in G_u$ to $s_j \in G_v$, involves two intra-group messages and one inter-group message, and consumes energy $e(L)(2h_u + h_{uv})$. The expected energy consumed in establishing a path key between s_i and s_j in the 2W and 2WA schemes is

$$E_{2W}(G_u, G_v) = e(L)2k(2h_u + h_{uv})$$
$$E_{2WA}(G_u, G_v) = e(kL)(4kh_u + 2h_{uv}).$$

Substituting for h_u, h_{uv}, δ_u, δ_{uv}, and d, we have

$$\frac{E_{2WA}(G_u, G_v)}{E_{2W}(G_u, G_v)} = \frac{e(kL)(2kn_g + n_g^{\frac{3}{2}} - 2k - 1)}{e(L)(2kn_g + kn_g^{\frac{3}{2}} - 3k)} \tag{11}$$

Each message in the 2W scheme contains a share, two sensor IDs and a group id, so that $L \leq 30$ bytes, and $kL \leq 300$ bytes even for $k = 10$. We know $\frac{e(L)}{e(kL)} \approx 1$ when $kL \leq 300$ bytes [21]. If $n_g = 100$,

$$\frac{E_{2WA}(G_u, G_v)}{E_{2W}(G_u, G_v)} = \frac{(198k + 999)}{1197k} \tag{12}$$

The E_{2WA}/E_{2W} ratio is 0.58 for $k = 2$, and 0.33 for $k = 5$. Aggregation is clearly effective. Surprisingly, 2WA can be more efficient than the very insecure 1W schemes. Since $E_{2W}/E_{1W} = 2$, the E_{2WA}/E_{1W} ratio is 1.16 for $k = 2$, and 0.66 for $k = 5$. Our 2WA method provides both strong security and efficiency at the same time.

8. RESILIENCE OF HPR

In this section, we give the formal analysis of the resilience of (k, I)-**HPR** against **KF**. **KF-KS** does not work here since there is no reverse phase in (k, I)-**HPR**.

8.1 Resilience against KF

We use mGKE as the key predistribution scheme, and use (k, I)-**HPR** as the path key establishment scheme. We analyze the probability that a path key K_{ij} of two sensors $s_i \in G_u$, $s_j \in G_v$ is compromised by **KF** attack when c sensors are captured and f sensors are faulty in the network.

We first derive the probability that $K_{s_is_j}$ is compromised in (k, I)-**HPR**. For simplicity, we only consider type-1 path keys. Let $\widehat{\mathbf{K}}_{\mathbf{s_is_j}}$ be the event that $K_{s_is_j}$ is compromised, $\mathbf{c}^{\langle \mathbf{ij} \rangle}$ be the event that c nodes are seized

in all, but neither s_i nor s_j is. Let $\mathbf{l_c}$ be the event that l_c agent pairs are captured between G_u and G_v. Let $\mathbf{b}$ be the event that b in $2t$ agents are captured between G_u and G_v. The probability that $K_{s_i s_j}$ is compromised when c sensors are captured is:

$$\Pr[\widehat{\mathbf{K}}_{\mathbf{s_i s_j}}|\mathbf{c}^{\langle \mathbf{ij}\rangle}] = \sum_{l_c} \Pr[\widehat{\mathbf{K}}_{\mathbf{s_i s_j}}|\mathbf{l_c}] \cdot \Pr[\mathbf{l_c}|\mathbf{c}^{\langle \mathbf{ij}\rangle}] \quad (13)$$

$$= \sum_{l_c}\sum_{b} \Pr[\widehat{\mathbf{K}}_{\mathbf{s_i s_j}}|\mathbf{l_c}] \cdot \Pr[\mathbf{l_c}|\mathbf{b}] \cdot \Pr[\mathbf{b}|\mathbf{c}^{\langle \mathbf{ij}\rangle}] \quad (14)$$

We derive each term in Formula 14. When (k, I)-**HPR** is used, the probability that K_{ij} is compromised when l_c agent pairs are seized between G_u and G_v is

$$\Pr[\widehat{\mathbf{K}}_{\mathbf{s_i s_j}}|\mathbf{l_c}] = \begin{cases} 0 & 0 \le l_c < k \\ \frac{\sum_{i=k}^{\min(l_c, I)} \binom{I}{i}}{\binom{t}{l_c}} & k \le l_c < t - I + k \\ 1 & t - I + k \le l_c \le t \end{cases} \quad (15)$$

For the second term in Formula 14, we have

$$\Pr[\mathbf{l_c}|\mathbf{b}] = \frac{\binom{t}{b-l_c}\binom{t-b+l}{2l_c-b}}{\binom{2t}{b}} \times 2^{2l_c-b} \quad \lceil \frac{b}{2} \rceil \le l_c \le \min(b, t) \quad (16)$$

The third term in Formula 14, we have

$$\Pr[\mathbf{b} \mid \mathbf{c}^{\langle \mathbf{ij}\rangle}] = \frac{\binom{n-2-2t}{c-b}\binom{2t}{b}}{\binom{n}{c}} \quad (17)$$

Fig. 13 shows the resilience of (k, I)-**HPR** against **KF**. From Fig. 13a and Fig 13b, we can see that the resilience is reduced by increasing I as we expect. In Fig. 13a, 13c and 13d, we set I to half of t. We can see that by increasing t and k, the resilience is also improved. We can see the resilience of (k, I)-**HPR** is comparable to 2W k-PR, and the communication overhead of (k, I)-**HPR** is smaller than 2W k-PR. **KF** can only compromise less than 1% path keys by carefully selecting proper t, k and I when 20% sensors are seized.

8.2 Resilience against Faulty Sensors

Now we begin to analyze the affection of faulty sensors to the path key establishment. We try to derive the probability that s_i and s_j cannot establish a path key when f sensors are faulty in the network.

Faulty sensors can affect path key establishment since although we loose the threshold for agent pair selection from top k agent pairs to any k agent pairs in top I agent pairs, it is still possible that more than $I - k + 1$ agent pairs are faulty in the top I agent pairs.

Let $\overline{\mathbf{K_{ij}}}$ denote the event that s_i and s_j cannot establish path keys because of faulty sensors. Let $\mathbf{f}^{\langle \mathbf{ij}\rangle}$ be the event that f nodes are faulty in all, but neither s_i nor s_j is. Let $\mathbf{l_f}$ be the event that l_f agent pairs are faulty between G_u and G_v. Let $\mathbf{b_f}$ be the event that b_f in $2t$ agents are faulty between G_u and G_v. We have

$$\Pr\left[\overline{\mathbf{K_{ij}}}|\right] = \sum_{l_f} \Pr\left[\overline{\mathbf{K_{ij}}}|\mathbf{l_f}\right] \cdot \Pr[\mathbf{l_f}|\mathbf{f}^{\langle \mathbf{ij}\rangle}] \quad (18)$$

$$= \sum_{l_f}\sum_{b_f} \Pr\left[\overline{\mathbf{K_{ij}}}|\mathbf{l_f}\right] \Pr[\mathbf{l_f}|\mathbf{b_f}] \Pr[\mathbf{b_f}|\mathbf{f}^{\langle \mathbf{ij}\rangle}] \quad (19)$$

The first term in Formula 19 is

$$\Pr\left[\overline{\mathbf{K_{ij}}}|\mathbf{l_f}\right] = \begin{cases} 0 & 0 \le l_f < I - k + 1 \\ \frac{\sum_{i=I-k+1}^{\min(l_f, I)} \binom{I}{i}}{\binom{t}{l_f}} & I - k + 1 \le l_c \le t - k \\ 1 & t - k + 1 \le l_c \le t \end{cases} \quad (20)$$

The second term in Formula 19 is

$$\Pr[\mathbf{l_f}|\mathbf{b_f}] = 2^{2l_f - b_f} \frac{\binom{t}{l_f}\binom{l_f}{b_f - l_f}}{\binom{2t}{b_f}} \quad \lceil \frac{b_f}{2} \rceil \le l_f \le \min(b_f, t) \quad (21)$$

The third term in Formula 19 is

$$\Pr[\mathbf{b_f}|\mathbf{f}^{\langle \mathbf{ij}\rangle}] = \frac{\binom{n-2-2t}{f-b}\binom{2t}{b}}{\binom{n}{f}} \quad (22)$$

Fig. 14 shows the resilience of (k, I)-**HPR** against faulty sensors. From Fig. 14a and Fig 14b, we can see that the resilience is improved by increasing I. In Fig. 14a, 14c and 14d, we set I to half of t. We can see that by increasing t and k, the resilience is also improved. when 20% sensors are faulty, only less than 1% path key establishments are affected by carefully selecting proper t, k and I. From these figures, we can see that with I increasing, our scheme has better resilience against faulty sensors. However, larger I means worse resilience against **KF**, which is not what we want.

9. KEY FOISTING IN THE GENERAL MODEL

We describe how to perform **KF** in general distributed systems. In a distributed system containing n nodes $\{s_1, s_2, \ldots, s_n\}$, the n nodes form a network, and can send messages to other nodes. There is also an adversary with the ability of eavesdropping messages transmitted between any two nodes. We assume that no central authority is available, but some node pairs have pre-defined cryptographic channels to protect communication security, and other node pairs can only use the pre-defined cryptographic channels to establish new cryptographic channels. Without lose of generality, we assume that these cryptographic channels are protected by *secrets* used in message encryption and decryption. We do not restrict secrets must be symmetric keys here.

If node s_i and s_j have a pre-defined cryptographic channel, they share a secret $K_{s_i s_j}$ only known by them used in message encryption and decryption.

Nodes not having preloaded secrets may need to establish *path secrets* for secure communication. In a system without central authority, these nodes can use existing cryptographic channels to establish path secrets. One difference between general distributed systems and mGKE based sensor networks is that cryptographic paths between two nodes are not pre-defined in general distributed systems. There can be arbitrary number of cryptographic paths between two nodes, and each path can contain arbitrary number of hops. We assume that s_i and s_j know all cryptographic paths to each other, and agree on these paths used in path secret establishment. How to make the agreement is not our focus in this paper.

We first describe one-way (1W) secret establishment scheme used in the general model. Say two nodes s_i, s_j want to establish a secret $K_{s_i s_j}$ for secure communication, and they share an existing cryptographic path $\{s_i \to s_{i_1} \to \ldots, \to s_{i_m} \to s_j\}$. This path can be used to establish the path secret $K_{s_i s_j}$, where (s_i, s_{i_1}) share a preloaded secret $K_{s_i s_{i_1}}$, $\{(s_{i_l}, s_{i_{l+1}})\}_{l=1:m-1}$ share preloaded secrets $\{K_{s_{i_l} s_{i_{l+1}}}\}_{l=1:m-1}$, and (s_{i_m}, s_j) share preloaded secret $K_{s_{i_m} s_j}$. K_{ij} is established as follows:

1. $s_i \to s_{i_1} : \langle (K_{s_i s_j}, s_i, s_{i_2}, s_{i_3}, \ldots, s_j) \| K_{s_i s_{i_1}} \rangle$
2. $s_{i_l} \to s_{i_{l+1}} : \langle (K_{s_i s_j}, s_i, s_{i_{l+2}, \ldots, s_j}) \| K_{s_{i_l} s_{i_{l+1}}} \rangle$
3. $s_{i_m} \to s_j : \langle (K_{s_i s_j}, s_i) \| K_{s_{i_m} s_j} \rangle$

The problem of this scheme is that if one node on the path is compromised, then $K_{s_i s_j}$ is compromised by key stealing attack. The adversary can also use **KF** to force s_i s_j to accept a fabricated $K_{s_i s_j}$.

9.1 Key Foisting

KF works in the general model as follows.

1. Seize s_a, and find all cryptographic paths containing s_a.
2. Target such a pair (s_i, s_j), where s_i, s_a are on a cryptographic path, and s_j, s_a are on another path. Fabricate a secret $K^*_{s_i s_j}$.
3. Fabricate a message that s_j wishes to establish secret $K^*_{s_i s_j}$ with s_i, encrypt the message, and send to s_i via the cryptographic path connecting s_a and s_i. Now, s_i is tricked into accepting $K^*_{s_i s_j}$.
4. Fabricate a message claiming s_i wishes to establish secret $K^*_{s_i s_j}$ with s_j via the cryptographic path connecting s_a and s_j. Encrypt this message, and send it to s_j. s_j is tricked into believing that the request originated with s_i.
5. s_i and s_j have been fooled into using $K^*_{s_j s_j}$.

9.2 Two-Way Key Establishment

Two-way secret establishment (2W) can also be used to deal with **KF** in the general model, and is described as follows.

9.2.1 Forward Phase of 2W Key Establishment:

s_i selects a cryptographic path $\{s_i \to s_{i_1} \to \ldots, \to s_{i_m} \to s_j\}$. Next, s_i encrypts a random $K^{\to}_{s_i s_j}$ with $K_{s_i s_{i_1}}$, sending it to s_{i_1} in message

$$\langle (K^{\to}_{s_i s_j}, s_i, s_{i_1}, \ldots, s_{i_m} s_j) \| K_{s_i s_{i_1}} \rangle.$$

Here secret $K_{s_i s_{i_1}}$ is used in encryption.

s_{i_1} recovers $K^{\to}_{s_i s_j}$ and sends it encrypted to s_{i_2} as

$$\langle K^{\to}_{s_i s_j}, G_u, s_i, s_{i_2}, \ldots, s_{i_m}, s_j \| K_{s_{i_1} s_{i_2}} \rangle.$$

At last, s_j recovers $K^{\to}_{s_i s_j}$ from s_{i_m}, and begins the reverse phase.

9.2.2 Reverse Phase of 2W Key Establishment:

In the reverse phase, s_j selects another cryptographic path $\{s_j \to s_{j_1} \to \ldots, \to s_{j_{m'}} \to s_i\}$ to s_i. s_j now picks a random values $K^{\leftarrow}_{s_i s_j}$ representing its half of the path secret. s_j sends this half to s_i, exactly mirroring s_i's actions in the forward phase, but using the nodes on the new path instead.

At the end of the reverse phase, s_i and s_j both have $K^{\to}_{s_i s_j}$ and $K^{\leftarrow}_{s_i s_j}$ and generate the path secret $K_{s_i s_j} = K^{\to}_{s_i s_j} \oplus K^{\leftarrow}_{s_i s_j}$.

k-Path Reinforcement (k-PR) can be used to improve security in the general model.

9.3 HRW based Key Establishment

We state how to use HRW in the general model. s_i first calculates the hash value $h(s_{i_1} || s_{i_2} || \ldots s_{i_m})$ for each cryptographic path between s_i and s_j, and select the k paths with the highest hash values to establish the path secret. The input to the hash function is the concatenation of all node IDs on the cryptographic path. The k paths with the highest random weights are selected. s_i generates the k shares of the path secret $K_{s_i s_j}$, and uses k-PR to send the k shares to s_j via the k paths selected. Each share is send to s_j via a different path.

s_j uses HRW to verify $K_{s_i s_j}$. s_j calculates the hash value for each cryptographic path participated in the path secret establishment, and checks whether the k paths are those with highest hash values. If so, s_j accepts $K_{s_i s_j}$, otherwise rejects it.

(k, I)-**HPR** can also be used here to improve the resilience against faulty nodes.

9.4 Security Analysis

9.4.1 Two-Way Scheme

KF can be perfectly addressed by two-way scheme. We first prove the following theorem.

THEOREM 4. *2W k-PR path key establishment scheme is immune to key foisting if $k > 0$, no matter how many nodes are seized, in the general model.*

Proof: Exactly as for Theorem 2. ■

9.4.2 HRW based Key Establishment

KF attack can be efficiently addressed by $(k, I)-$**HPR**. We analyze the resilience of $(k, I)-$**HPR** against **KF**. We analyze the probability that a path secret $K_{s_i s_j}$ of two nodes s_i s_j is compromised by **KF** attack when l_c in t cryptographic paths are compromised by the adversary. We have

$$\Pr[\widehat{\mathbf{K}}_{\mathbf{s_i s_j}}|\mathbf{l_c}] = \begin{cases} 0 & 0 \le l_c < k \\ \frac{\sum_{i=k}^{\min(l_c, I)} \binom{I}{i}}{\binom{t}{l_c}} & k \le l_c < t - I + k \\ 1 & t - I + k \le l_c \le t \end{cases} \quad (23)$$

In Formula. 23, when the adversary compromises fewer than k cryptographic paths, he cannot compromise $K_{s_i s_j}$. When the adversary compromises $k \le l_c < t - I + k$ cryptographic paths, he has the probability $\frac{\sum_{i=k}^{\min(l_c, I)} \binom{I}{i}}{\binom{t}{l_c}}$ to have s_i accept his $K_{s_i s_j}$, and he has the same probability to have s_j accepts the secret. When the adversary compromises more than $t-I+k$ paths, he will have both s_i, s_j accepts his $K_{s_i s_j}$ with probability 1.

10. CONCLUSION

We have described key foisting, a new attack on sensor systems that has not so far been recognized in the literature, and showed how current schemes fail catastrophically against it. We then presented two-way key establishment and HRW based key establishment which are practical in mGKE, and confer excellent resilience against foisting and related attacks, including man-in-the-middle attacks. We provided a detailed analysis of these attacks, and verified the accuracy of our analysis with detailed simulations. Our analysis and simulations confirm that mGKE (2W) has excellent resilience against both key stealing and foisting attacks. The two-way scheme and (k, I)-**HPR** have very low overhead compared even with the insecure one-way scheme. Our future work will include reducing the overheads even further, and implementing these schemes on real sensor networks.

11. REFERENCES

[1] C. Adams and S. Lloyd. *Understanding PKI: Concepts, Standards, and Deployment Considerations*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2nd edition, 2002.

[2] L. Atzori, A. Iera, and G. Morabito. The internet of things: A survey. *Comput. Netw.*, 54:2787–2805, October 2010.

[3] R. Blom. An optimal class of symmetric key generation systems. In *Proc. of the EUROCRYPT 84 workshop on Advances in cryptology: theory and application of cryptographic techniques*, pages 335–338, New York, NY, USA, 1985. Springer-Verlag New York, Inc.

[4] H. Chan and A.Perrig. Pike: Peer intermediaries for key establishment in sensor networks. In *Proceedings of IEEE Infocom*, pages 524–535, 2005.

[5] H. Chan, A. Perrig, and D. Song. Random key predistribution schemes for sensor networks. In *SP '03: Proceedings of the 2003 IEEE Symposium on Security and Privacy*, page 197, Washington, DC, USA, 2003. IEEE Computer Society.

[6] H. Chan, A. Perrig, and D. Song. Secure hierarchical in-network aggregation in sensor networks. In *CCS '06: Proceedings of the 13th ACM conference on Computer and communications security*, pages 278–287, New York, NY, USA, 2006. ACM.

[7] R. Di Pietro, L. Mancini, and S. Jajodia. Providing secrecy in key management protocols for large wireless sensors networks. *Ad Hoc Networks*, 1(4):455–468, 2003.

[8] D. Dolev and A. C. Yao. On the security of public key protocols. *Foundations of Computer Science, Annual IEEE Symposium on Foundations of Computer Science*, 0:350–357, 1981.

[9] J. R. Douceur. The sybil attack. In *1st International Workshop on Peer-to-Peer Systems*, March 2002.

[10] W. Du, J. Deng, Y. S. Han, P. K. Varshney, J. Katz, and A. Khalili. A pairwise key predistribution scheme for wireless sensor networks. *ACM Trans. Inf. Syst. Secur.*, 8(2):228–258, 2005.

[11] W. Du, J. Deng, Y. S. Han, P. K. Varshney, J. Katz, and A. Khalili. A pairwise key predistribution scheme for wireless sensor networks. *ACM Trans. Inf. Syst. Secur.*, 8(2):228–258, 2005.

[12] L. Eschenauer and V. D. Gligor. A key-management scheme for distributed sensor networks. In *CCS '02: Proceedings of the 9th ACM conference on Computer and communications security*, pages 41–47, New York, NY, USA, 2002. ACM.

[13] B. Ghosh. Random distances within a rectangle, and between two rectangles. *Bulletin of the*

Calcutta Mathematical Society, 43:17–24, 1951.

[14] C. Karlof and D. Wagner. Secure routing in wireless sensor networks: Attacks and countermeasures. In *First IEEE International Workshop on Sensor Network Protocols and Applications*, pages 113–127, 2002.

[15] G. Li, H. Ling, and T. Znati. Path key establishment using multiple secured paths in wireless sensor networks. In *CoNEXT '05: Proceedings of the 2005 ACM conference on Emerging network experiment and technology*, pages 43–49, New York, NY, USA, 2005. ACM.

[16] D. Liu and P. Ning. Establishing pairwise keys in distributed sensor networks. In *CCS '03: Proceedings of the 10th ACM conference on Computer and communications security*, pages 52–61, New York, NY, USA, 2003. ACM.

[17] D. Liu, P. Ning, and W. Du. Group-based key predistribution for wireless sensor networks. *ACM Trans. Sen. Netw.*, 4(2):1–30, 2008.

[18] J. Newsome, E. Shi, D. Song, and A. Perrig. The sybil attack in sensor networks: analysis & defenses. In *IPSN '04: Proceedings of the 3rd international symposium on Information processing in sensor networks*, pages 259–268, New York, NY, USA, 2004. ACM.

[19] J. Ni, L. Zhou, , and C. V. Ravishankar. Dealing with random and selective attacks in wireless sensor systems. *ACM Transactions on Sensor Networks*, 6(2), February 2010.

[20] B. Przydatek, D. Song, and A. Perrig. Sia: secure information aggregation in sensor networks. In *SenSys '03: Proceedings of the 1st international conference on Embedded networked sensor systems*, pages 255–265, New York, NY, USA, 2003. ACM.

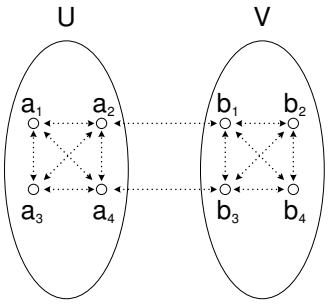
[21] Y. Sankarasubramaniam, I. F. Akyildiz, and S. W. McLaughlin. Energy efficiency based packet size optimization in wireless sensor networks. *Sensor Network Protocols and Applications, 2003. Proceedings of the First IEEE. 2003 IEEE International Workshop on*, pages 1–8, June 2003.

[22] B. Schneier. Applied cryptography: Protocols, algorithms, and source code in c. Wiley; 2nd endition, 1995.

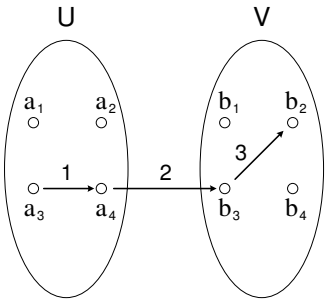
[23] D. G. Thaler and C. V. Ravishankar. Using name-based mappings to increase hit rates. *IEEE/ACM Trans. Netw.*, 6(1):1–14, Feb. 1998.

[24] O. Vermesan, M. Harrison, H. Vogt, K. Kalaboukas, M. Tomasella, K. Wouters, S. Gusmeroli, and S. Haller. Internet of things—strategic research roadmap. Technical report, European Commission - Information Society and Media DG, 2009.

[25] S. Zhu, S. Setia, and S. Jajodia. LEAP+: Efficient security mechanisms for large-scale distributed sensor networks. *ACM Transactions on Sensor Networks (TOSN)*, 2(4):528, 2006.



(a) Group based sensor network



(b) Path key establishment

Figure 1: Path key establishment. Dotted lines (Fig. 1a) show associations, and solid lines (Fig. 1b) show path key establishment messages. a_3 initiates path key establishment with b_2, via intermediaries a_4 and b_3.

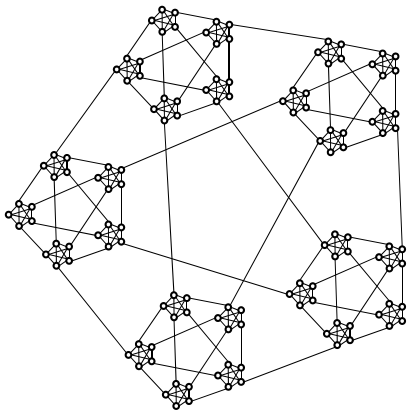
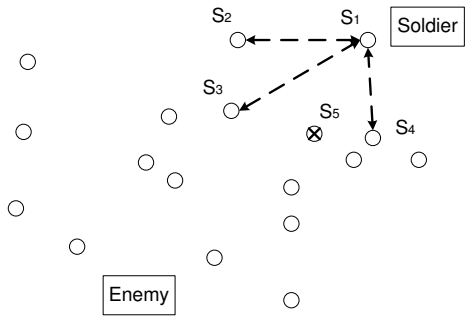
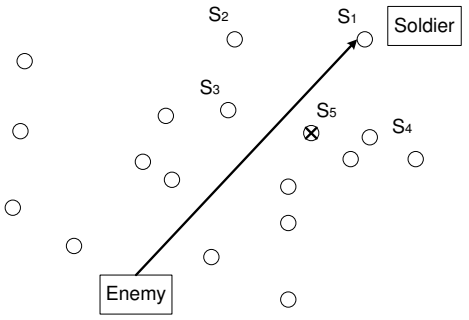


Figure 2: Hierarchy of sensor groups

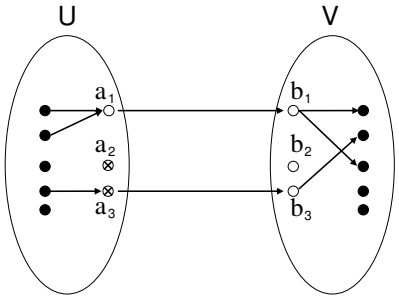


(a) Compromising path keys

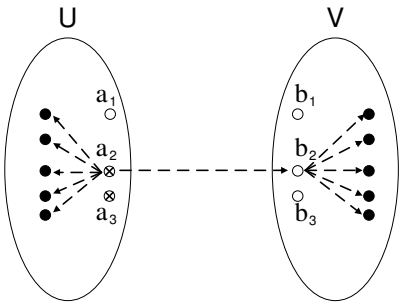


(b) Injecting false data

Figure 3: False data injection. Fig. 3a shows that sensor s_1 shares path keys with s_2, s_3, s_4. These have been mediated by s_5. The adversary seizes s_5, and gains these path keys. In Fig. 3b, he feeds false data to s_1 by impersonating s_2, s_3 and s_4.

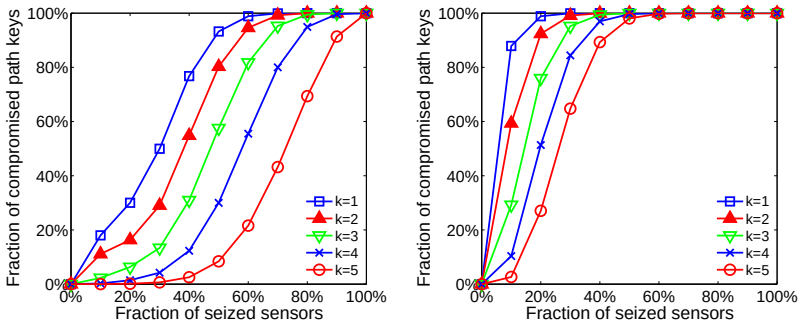


(a) Key Stealing

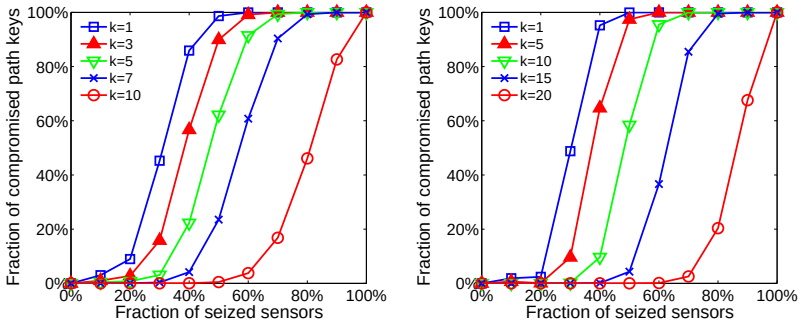


(b) Key Foisting

Figure 4: Key stealing and foisting. ⊗: seized agents.

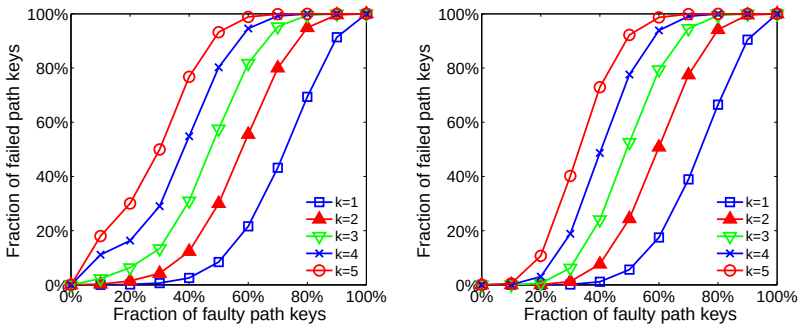


(a) $t = 10$ $I = 5$ (b) $t = 10$ $I = 10$

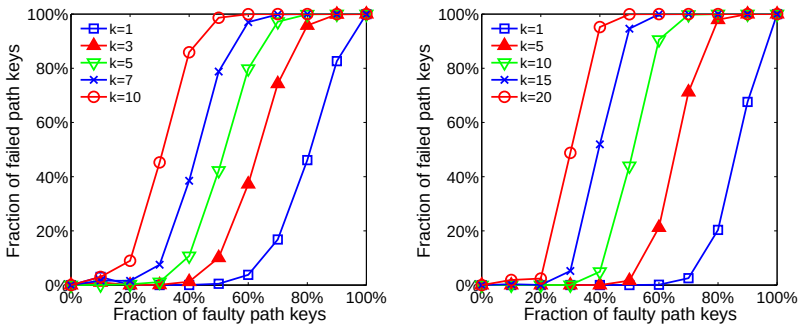


(c) $t = 20$ $I = 10$ (d) $t = 40$ $I = 20$

Figure 13: Resilience against **KF**



(a) $t = 10$ $I = 5$ (b) $t = 10$ $I = 10$



(c) $t = 20$ $I = 10$ (d) $t = 40$ $I = 20$

Figure 14: Fraction of sensors cannot establish path keys due to faulty sensors

Notation	Meaning
n_s	number of sensors per group
n_g	number of groups
G_u	the uth group
$K_{s_i s_j}$	the pairwise key between s_i and s_j
t	number of agent pairs between groups
a_{uv}^i	the i-th agent in G_u for G_v
A_{uv}	the set of agents between G_u, G_v
$\langle M \| K_{s_i s_j} \rangle$	message M encrypted with key $K_{s_i s_j}$
$[K \backslash p]$	p-th share of a secret K
2W / 1W	Two-way/one-way path key establishment
k-PR	k-path key reinforcement
$K_{s_i s_j}^{\rightarrow}$	a half of the path key sent from s_i to s_j
$K_{s_i s_j}^{\leftarrow}$	a half of the path key sent from s_j to s_i

Table 1: Our Notation

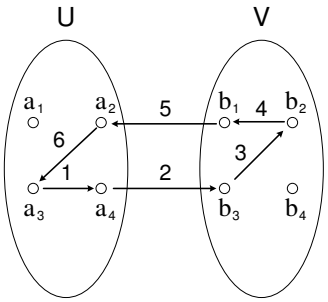


Figure 5: 2W path key establishment in mGKE

$\mathbf{K_{s_i s_j}}$	$K_{s_i s_j}$ is path key between s_i, s_j
$\mathbf{K_{ij}^q}$	$K_{s_i s_j}$ is type-q path key, $q = 1, 2, 3$
$\mathbf{c}^{\langle \mathbf{ij} \rangle}$	c nodes are seized in the network as a whole
$\mathbf{b}$	b of the $2t$ agents between (G_u, G_v) seized
$\mathbf{b}^{\langle \mathbf{ij} \rangle}$	same as $\mathbf{b}$, without s_i or s_j being seized.
$\widehat{\mathbf{K}}_{\mathbf{s_i s_j}}$	path key $K_{s_i s_j}$ between s_i, s_j is stolen

Table 2: Notation

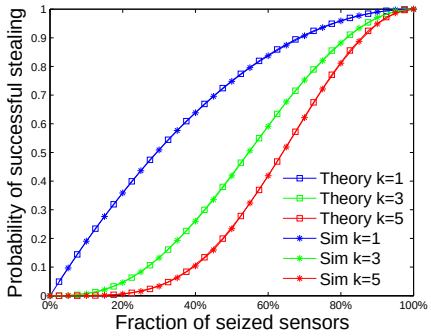


Figure 6: mGKE (1W k-PR) key stealing resilience ($t = 10$, $n_s = n_g = 100$). Theory matches simulation perfectly.

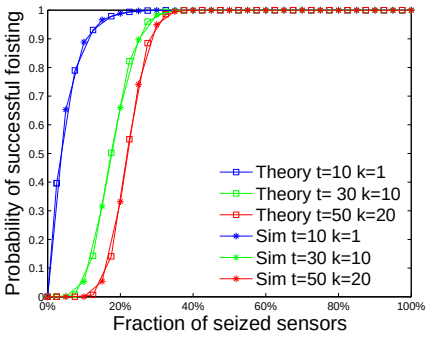


Figure 7: mGKE (1W k-PR) resilience to key foisting ($n_s = n_g = 100$). Theory matches simulation perfectly.

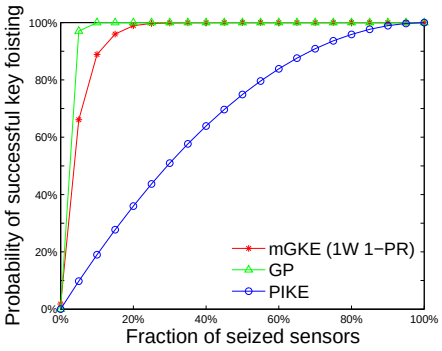


Figure 8: Key-foisting resilience: mGKE (2W 1-PR), mGKE (1W 1-PR), PIKE, GP.

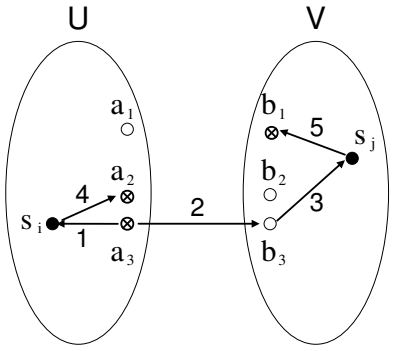


Figure 9: **KF-KS** attack in mGKE (2W 1-PR)

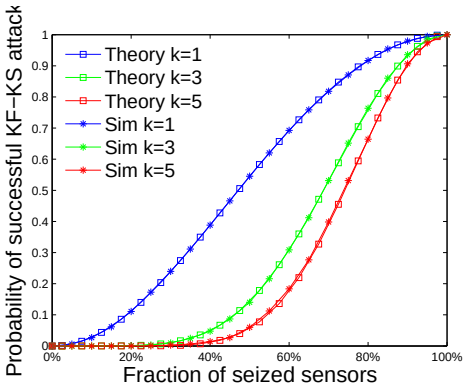
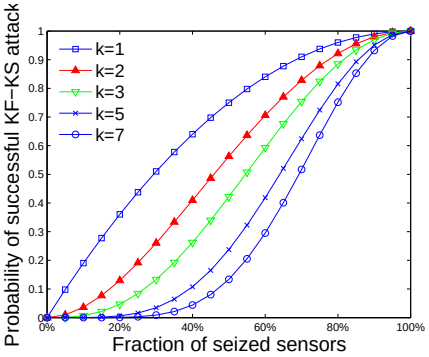
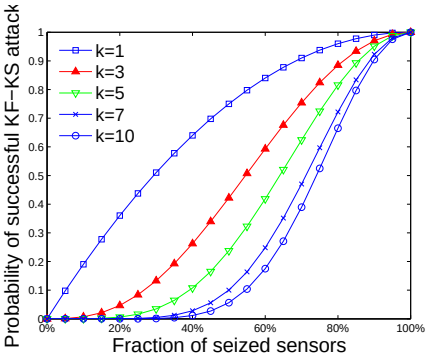


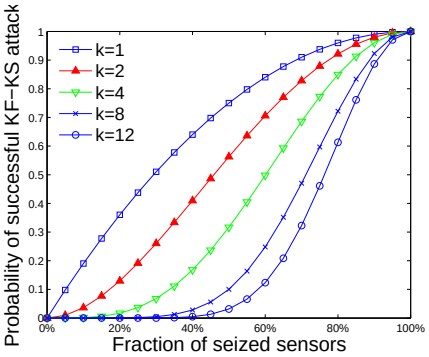
Figure 10: mGKE (2W k-PR) resilience to man-in-the-middle attack, theory vs. simulation ($t = 10$, $n_s = n_g = 100$)



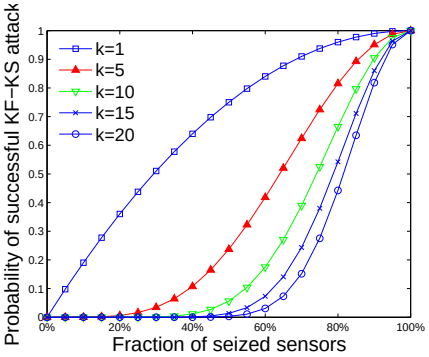
(a) $t = 15$



(b) $t = 20$

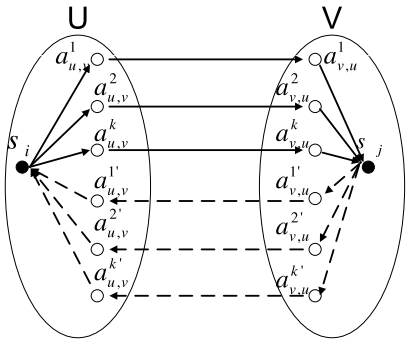


(c) t=25

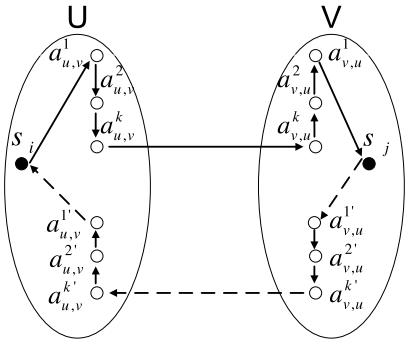


(d) $t = 40$

Figure 11: The resilience of mGKE (2W k-PR) against **KF-KS** Attacks ($t = 15, 20, 25, 40$, $n_s = 100$, $n_g = 100$)



(a) Naive scheme.



(b) Message aggregation (2WA).

Figure 12: 2W path-key establishment