# A Collusion Attack on Pairwise Key Predistribution Schemes for Distributed Sensor Networks

Tyler Moore
*Computer Laboratory, University of Cambridge, United Kingdom*
*Tyler.Moore@cl.cam.ac.uk*

## Abstract

*Key predistribution schemes are a favoured solution for establishing secure communication in sensor networks. Often viewed as the safest way to bootstrap trust, the main drawback is seen to be the large storage overhead imposed on resource-constrained devices. In this paper, we argue that predistribution schemes can actually be quite insecure: pre-loading global secrets onto exposed devices strengthens the incentive for attackers to compromise nodes. Furthermore, lack of coordination between nodes arising from localised communication helps attackers hide misbehaviour. We consider one scheme in particular—Chan* et al.*'s random pairwise key predistribution [3]—and demonstrate an attack where colluding nodes reuse selected pairwise keys to create many false identities. We find that a small, colluding minority can hijack a majority of node communication channels. Finally, we consider countermeasures, from improved detection to scrapping predistribution altogether.*

## 1 Introduction

Researchers have proposed several schemes for establishing secure communication in sensor networks. Many proposals assign symmetric keys to nodes prior to deployment. This strategy is chosen to satisfy two assumptions: first, that the network topology cannot be established in advance, and second, that nodes are deployed in hostile environments monitored by the adversary. Eschenauer and Gligor have devised a scheme where nodes are assigned a random subset of keys from a large key space [7]. Any two nodes sharing a common key can establish communication.

Unfortunately, such key pools are vulnerable to a colluding minority of attacker-controlled nodes. One problem is that several nodes possess the same keys; another is that any node can make use of them. Simply combining the keys obtained from a handful of nodes greatly increases the attacker's chances of sharing keys with (and therefore eavesdropping on) other nodes.

To alleviate these shortcomings, Chan *et al.* propose to randomly predistribute *pairwise* keys [3]: unique secrets shared between just two nodes throughout the network. This solves the key-harvesting attack above at the expense of increased storage overhead. The authors also claim that pairwise key assignment enables mutual authentication at no added expense: nodes are assured of each other's identities by possessing the appropriate pairwise key.

In this paper, we describe an attack that undermines these properties. While predistributing pairwise keys does protect confidentiality, it still loads nodes with a large number of globally-applicable secrets. By eliminating the eavesdropping attack, the pairwise scheme makes another type of malicious behaviour more attractive. A collusive attacker can now share its pairwise keys between compromised nodes, enabling each to present multiple 'authenticated' identities to neighbouring nodes while escaping detection. We show that such an attacker can establish enough forged communication channels to outnumber legitimate ones using a relatively small group of colluding nodes. Pairwise or not, preloading global secrets significantly strengthens an attacker's incentive to corrupt more nodes.

First, we describe this attack in greater detail; next we analyse its impact. One result is that a small, colluding minority (less than 5% of the network) can control half of its neighbours' outbound communication channels. Finally, we discuss the implications for countermeasures.

## 2 Key-swapping collusion attack

### 2.1 System model

A summary of notation used is given in Table 1.

For a sensor network of maximum size $n$, each node is assigned a pairwise key to another node with probability $p$. This probability is chosen so that the resulting graph is connected with very high probability. The keys are stored along with the corresponding node identifier. Typically, $p$ ranges from $0.2$ to $0.4$, while $n$ ranges from several hundred

| Symbol | Meaning |
|---|---|
| $n$ | network size |
| $n'$ | expected number of neighbour nodes in radio range |
| $p$ | probability of two nodes sharing a pairwise key |
| $k_{cd}$ | pairwise key shared between nodes $c$ and $d$ |
| $q$ | number of attacker-controlled nodes |
| $A$ | set of attacker-controlled nodes |
| $N(d)$ | set of neighbours of node $d$ |
| $U(d)$ | set of usable pairwise keys for node d |

**Table 1. Notation.**

to several thousand. Therefore, each node stores a random set of $n * p$ pairwise keys and IDs rather than $n - 1$ keys required for full pairwise key predistribution [3].

Note that nodes have a limited communication radius; each node $d$ can reach a set of nodes $N(d)$ within $d$'s radio range. Nodes are geographically positioned randomly across a space so that each can contact $n'$ other nodes on average. Typical values for $n'$ range from 40 to 60.

Upon deployment, nodes broadcast their identifiers to neighbours, who examine the ID to determine whether they share a pairwise key. For example, a pairwise key $k_{cd}$ is added to $U(c)$, the set of usable pairwise keys for $c$, if one of node $c$'s neighbours $d \in N(c)$ holds $k_{cd}$.

## 2.2 Threat model

Sensor networks are often deployed in hostile environments, yet nodes cannot afford expensive tamper-resistant hardware. Therefore, a motivated attacker can compromise (via physical or remote exploitation) a set of nodes $A$, $q = |A|$, obtaining their pairwise secret keys and controlling outbound communications. We also assume nodes can collude by sharing their keys with other attacker nodes.

Traditionally, the threat from node compromise is measured by its impact on confidentiality—whether secret keys shared between uncompromised nodes can be obtained. In contrast, we have expanded the threat model to include the impact on integrity and availability by considering the proportion of attacker-controlled communication channels. Notions of confidentiality are moot if an attacker commands the vast majority of transmission paths.

## 2.3 Attack description

We define a novel attack that exploits the combination of pre-loaded keys and localised interaction of sensor nodes. Consider two nodes controlled by the attacker, $a, b \in A$. If $a$ tells $b$ its secrets, then $b$ can masquerade as $a$ to all of $b$'s neighbours that $a$ shares pairwise keys with, and vice versa. The keys from each subsequently obtained node can be reused by the other attacker-controlled nodes, cascading the impact of node compromise.

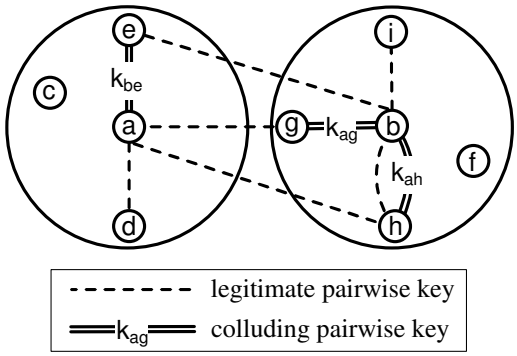


**Figure 1. Example key-swapping collusion attack between attacker nodes $a$ and $b$.**

The attack is similar to a Sybil attack [4] in that single nodes present multiple identities; however, these identities are not randomly generated but instead are reused according to available pairwise keys. The attack also shares characteristics of node replication attacks, where copies of a node are inserted into a network. However, the *key-swapping collusion attack* is unique in that attacker-controlled nodes pretend to be different nodes to different neighbours.

Consider the example sensor network in Figure 1. Node $a$'s neighbours are $c$, $d$ and $e$, while $a$ shares pairwise keys with $d$, $g$ and $h$. Node $b$'s neighbours are $f$, $g$, $h$ and $i$, while $b$ shares keys with $e$, $h$ and $i$. Thus, $a$ can legitimately communicate directly with only node $d$, while node $b$ can communicate legitimately with nodes $h$ and $i$. But if $a$ and $b$ collude to share each other's secrets, then $a$ can communicate with $e$ by pretending to be $b$, and $b$ can pretend to be $a$ when communicating with $g$ and $h$. Note that even though node $b$ can already communicate with $h$, colluding with $a$ enables $b$ to present multiple identities to $h$. Attacker nodes can communicate beyond radio range using the network's existing routing mechanism or an out-of-band channel.

Following are the resulting sets of usable pairwise keys when $a$ and $b$ act independently and collude:

| | Independence | Collusion |
|---|---|---|
| $U(a)$ | $\{k_{ad}\}$ | $\{k_{ad}, k_{be}\}$ |
| $U(b)$ | $\{k_{bh}, k_{bi}\}$ | $\{k_{bh}, k_{bi}, k_{ag}, k_{ah}\}$ |

As more nodes are compromised, overlap between node communication ranges must be taken into consideration. Overlap becomes unavoidable as the number of attacker-controlled nodes $q$ approaches $\frac{n}{n'}$. Note that two colluding nodes gain nothing by both pretending to be the same node to a common neighbour. In Figure 2, nodes $c$ and $a$ can both masquerade as $b$ to node $e$, but only one of them should do so. Also, node $c$ achieves nothing by pretending to be $a$ to $d$, since $d$ already shares a pairwise key with $a$.

The collusion attack works due to a devastating combination of globally-applicable secrets and locally-

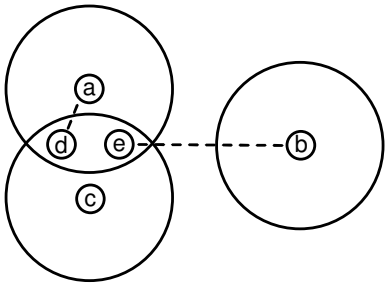


**Figure 2. Overlap between attacker nodes $a$ and $c$.**

communicating nodes. Pairwise keys can be used throughout the network, yet ordinary sensors can only communicate with the small fraction of nodes within radio range. An attacker can readily exploit this lack of coordination between nodes. The smaller the ratio $\frac{n'}{n}$ between average node neighbourhood and the overall network size, the greater is the level of uncoordination between nodes. Likewise, as the average fraction $p$ of pairwise keys stored by each node increases, each compromised node offers more potentially usable pairwise keys to the attacking node.

We have noted that the confidentiality of node communications is not affected by increasing node capture. However, the integrity and availability of node interaction is certainly threatened. For instance, attacker-controlled nodes increase their chances of partitioning the network or counteracting redundant routing whenever ordinary nodes believe they are dealing with many nodes instead of one.

Clearly, authenticating nodes based upon possession of a particular pairwise key is inadequate. Therefore, any distributed node revocation scheme where votes are authenticated by the possession of pairwise keys (as proposed in [3]) is undermined if nodes share their secrets.

# 3 Analysis

We simulated a sensor network comprised of nodes uniformly distributed over a plane, setting $n = 1000$, $n' = 60$, $p = .25$ and varied $q$, averaging results from 20 rounds.

To quantify the collusion attack's impact, we focus on the pairwise secret keys stored by each sensor node. These keys are required to establish secure communication as well as provide authentication; as such, they are prized by an attacker. We first compare the number of usable pairwise keys available when the attacker-controlled nodes act independently versus when they collude. Second, we compare the number of pairwise keys available to the attacker relative to the number of legitimate usable keys available to the attacker's neighbours. This measure quantifies the level of network penetration achieved by the attacker.

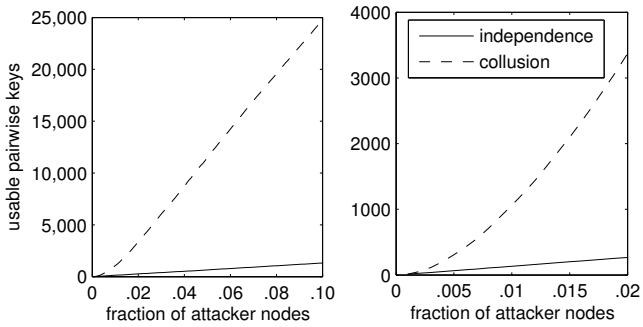


**Figure 3. Usable pairwise keys available to attacker-controlled nodes.**

## 3.1 Increased usable pairwise keys

Figure 3 compares the number of usable pairwise keys available to attacker-controlled nodes, $\sum_{a \in A} |U(a)|$ for increasing $q$. A pairwise key is usable if it is shared between nodes in communication range and it is not already in use within this range. Colluding attackers can access each other's pairwise keys; thus it is no surprise that the figure indicates many more usable pairwise keys available to attacker-controlled nodes when they share secrets.

Acting alone, each newly compromised node creates an additional $n' * p$ usable pairwise keys on average. Therefore, the total number of usable pairwise keys grows linearly in the number of attacker-controlled nodes, or $q * n' * p$. Ignoring overlap, the number of usable pairwise keys under collusion grows with the square of the number of attacker-controlled nodes, or $q^2 * n' * p$. This is because each newly-compromised node can be used to communicate with $n' * p$ of each compromised node's neighbours.

Taking node overlap into account does significantly slow the growth of attacker-controlled pairwise keys. The second graph in Figure 3 shows usable keys for initially compromised nodes. One can see that the number initially grows quadratically before quickly slowing to linear growth; note the coefficient is much larger than in the independent case.

Figure 4 shows the rate of change of the number of usable pairwise keys; as $q$ grows larger, each colluding node possesses an average of $n * p$ usable pairwise keys. Thus the average total number of pairwise keys for $q$ compromised nodes is $q * n * p$. Note that $n$ is typically much larger than $n'$ and that $n'$ stays constant even as $n$ grows. For our simulations, $\frac{n}{n'} = 16.67$. Therefore, even at its limiting growth rate, colluding attackers obtain $\frac{n}{n'}$ times as many usable pairwise keys as when acting alone.

The following table summarises results for this measure:

| Independence | Collusion (initial) | Collusion (limiting) |
|---|---|---|
| $q * n' * p$ | $q^2 * n' * p$ | $q * n * p$ |

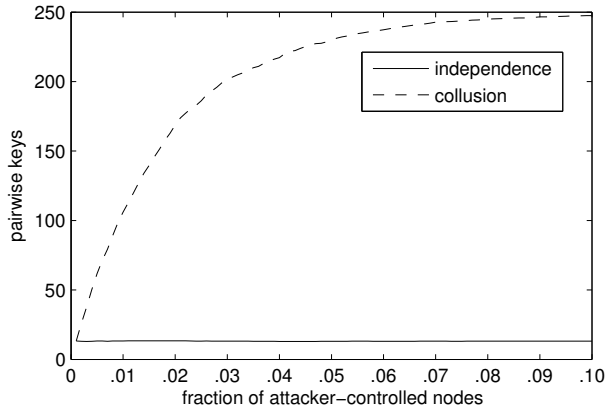


**Figure 4. Average number of usable pairwise keys per attacker node.**

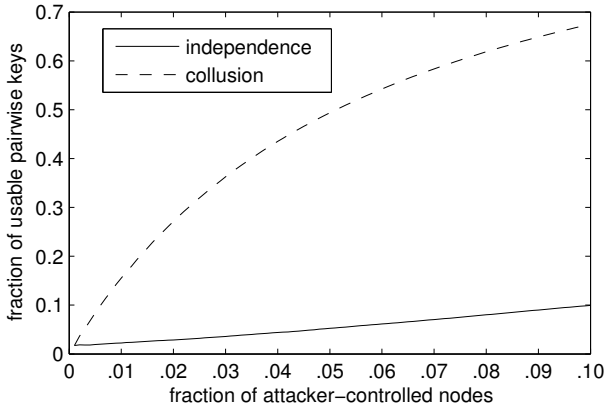


**Figure 5. $I(A)$, fraction of communication channels controlled by attacker.**

### 3.2 Quantifying attacker penetration

But what is the overall impact of a collusion attack on the sensor network? We have devised a telling measure:

$$I(A) = \frac{\sum_{a \in A} |U(a)|}{\sum_{a \in A} \sum_{b \in N(a)} |U(b)|}$$

$I(A)$ compares the number of usable pairwise keys available to an attacker to the number of usable pairwise keys for all of the attacker-controlled nodes' neighbours. In effect, this reveals the degree to which an attacker's geographical neighbours are enveloped by masquerading attacker nodes.

Figure 5 measures $I(A)$ for increasing $q$. It demonstrates that a small collection of colluding nodes can overtake a large fraction of its neighbours' communications. Without colluding, attacker-controlled nodes gain no added influence over their neighbours: compromising 2% of the network yields control over just 2% of their neighbours' pairwise keys. With colluding, attacker-controlled nodes quickly gain considerable influence over their neighbours: compromising 2% of the network yields control over 27% of their neighbours' pairwise keys; compromising 5% enables power over approximately half of the valid communication channels.

Therefore, a colluding minority, while not capable of eavesdropping, can nonetheless swallow much of the network's interactions. A distributed voting scheme can be undermined by a 5% colluding minority since half of the votes are cast by the attacker. Any application that requires honest interaction with the majority of node neighbours is susceptible. Unfortunately, most sensor network applications do, from routing to data aggregation.

## 4 Discussion

### 4.1 Storage requirements

We have not considered whether colluding nodes can store and transmit all applicable pairwise keys. From Figure 4, each attacker-controlled node receives at most $n * p$ usable pairwise keys from collusion in addition to $n * p$ predistributed pairwise keys; thus, these nodes need to store up to $2n * p$ keys. Given the severe memory constraints placed on sensor nodes, storing extra keys could prove too onerous.

However, an observation about the effects of overlap on available keys reveals an alternative requiring no hardware modification to meet added storage demands. Each pairwise key can only be used once: when two nodes collude, the only usable keys for one node are those that could not be used by the other. Pairwise keys could help two nodes only when overlap exists in the communication range. As discussed in Section 2, only one node may use such keys.

A resourceful attacker can exploit this fact by adding another step to the collusion. After a node shares a pairwise key with another attacker-controlled node, it can simply delete the key, replacing it with any keys provided by the other node. Thus key-sharing becomes key-swapping. In the end, each attacker-controlled node still stores $n * p$ keys, but now each key can be used to communicate with neighbours, albeit using the identities of colluding nodes.

### 4.2 The cost of predistributing secrets

The problem with assigning general-purpose keys to nodes prior to deployment is that such secrets often prove more useful to attackers than to ordinary nodes. A node should store keys for its neighbours; holding any more, as advocated by predistribution schemes, only serves a collusive attacker. Until now, the only costs attributed to pre-

distribution have been storage-related. We have shown how predistributing secrets also raises security costs.

In fact, any scheme that preassigns global secrets to locally communicating nodes is at risk to similar attacks. Instead of predistribuing pairwise keys directly as in Chan *et al.*'s approach, Du *et al.* [6] and Liu and Ning [8] randomly predistribute capabilities for computing pairwise keys. Nodes may collude just the same, spoofing keys from shared seeds. More recently, PIKE [2] reduces node storage requirements to $O(\sqrt{n})$ pairwise keys, yet remains susceptible to a collusive minority of key-swapping nodes.

### 4.3 Countermeasures

There are two approaches for countering collusion attacks: either reducing the utility of compromised nodes to attackers or detecting the reuse of pairwise keys. The former can be limiting, while the latter is often quite expensive.

One option is for nodes to discard unused keys after an initialisation phase, but this means new nodes can no longer join the system once initialisation is complete. Another is to reduce the number of pre-loaded keys. This can be achieved in many ways, though each technique introduces its own limitations. Some have proposed just pre-loading keys that are geographically close [9, 5]. However, it is not always reasonable to assume the existence of topological knowledge prior to deployment, especially in mobile applications. A more radical approach is key infection, which scraps predistribution altogether in favor of simply transmitting keys in the clear [1]. Key infection schemes are therefore not susceptible to the attack presented here, though they are more vulnerable when a powerful adversary is present.

If a sensor network is deployed with a uniform density, then nodes can detect whether they might be under attack by tracking how many connected neighbours they have. Nodes have $n'$ neighbours on average, but an attacked one may have up to an additional $q * p$ faked neighbours. However, determining which of these neighbours are lying can be difficult, especially if colluding attackers do not reuse the same keys on overlapping node neighbourhoods.

One way to identify misbehaving nodes is to require them to transmit their locations. While this too can be faked, key reuse can be detected if nodes recursively ask their neighbours for the location of their transaction partners. Because even a faked location must be within communication range of a duped node, an attacker has no choice but to transmit multiple locations for a single identity. In [11], Parno *et al.* propose a similar detection scheme for node replication attacks on sensor networks.

Such detection schemes do have drawbacks, though. Requiring nodes to transmit their location helps an attacker target new nodes for compromise. It is also quite costly to do so: even Parno *et al.*'s 'efficient' technique requires a further $O(\sqrt{n})$ storage per node and $O(n\sqrt{n})$ messages.

Strategies for defending against Sybil attacks given in [10] do not apply since nodes present multiple identities by reusing legitimate identifiers and pairwise keys.

## 5 Conclusions

We have presented a novel collusion attack on the class of pairwise key predistribution schemes and demonstrated its devastation of secure communications on a sensor network. In doing so, we have devised a measure quantifying the level of attacker penetration. This could prove to be an important first step for comparing the suitability of different key establishment schemes—a crucial task given the proliferation of approaches.

Essentially, we have questioned the wisdom of assigning global secrets to locally-communicating nodes. Doing so strengthens the attacker's incentive to compromise more nodes as well as increase the potency of subsequent attacks. Perhaps the next step is to find ways to effectively pair limited secrets that correspond to localised interactions.

## References


[1] R. Anderson, H. Chan and A. Perrig. Key Infection: Smart Trust for Smart Dust. In *IEEE International Conference on Network Protocols*, 2004.

[2] H. Chan and A. Perrig. PIKE: Peer Intermediaries for Key Establishment in Sensor Networks. In *IEEE INFOCOM*, 2005.

[3] H. Chan, A. Perrig and D. Song. Random Key Predistribution Schemes for Sensor Networks. In *IEEE Symposium on Security and Privacy (S&P)*, 2003.

[4] J.R. Douceur. The Sybil Attack. In *International Workshop on Peer-to-Peer Systems*, 2001.

[5] W. Du, J. Deng, Y. Han, S. Chen and P. Varshney. A Key Management Scheme for Wireless Sensor Networks Using Deployment Knowledge. In *IEEE INFOCOM*, 2004.

[6] W. Du, J. Deng, Y. Han and P. Varshney. A Pairwise Key Pre-distribution Scheme for Wireless Sensor Networks. In *ACM Conference on Computer and Communications Security (CCS)*, 2003.

[7] L. Eschenauer and V. Gligor. A Key-Management Scheme for Distributed Sensor Networks. In *ACM CCS*, 2002.

[8] D. Liu and P. Ning. Establishing pairwise keys in distributed sensor networks. In *ACM CCS*, 2003.

[9] D. Liu and P. Ning. Location-Based Pairwise Key Establishments for Static Sensor Networks. In *ACM Workshop on Security of Ad Hoc and Sensor Networks*, 2003.

[10] J. Newsome, E. Shi, D. Song and A. Perrig. The Sybil Attack in Sensor Networks: Analysis & Defenses. In *Symposium on Information Processing in Sensor Networks*, 2004.

[11] B. Parno, A. Perrig and V. Gligor. Distributed Detection of Node Replication Attacks in Sensor Networks. In *IEEE S&P*, 2005.