Chapter 3

Secrecy amplification

The uncertainty about the identity of direct neighbours prior to the actual deployment naturally results in such a property of the key distribution schemes that most of the nodes in the network should be able to establish a shared key (either directly or with support from other nodes). This alone is a serious challenge for memory- and battery-limited nodes. At the same time, this property implies one of the main problems for maintaining a secure network in presence of an adversary with the ability to compromise link keys (e.g., via node capture or eavesdropping). If the extracted secrets can be commonly used in any part of the network, various Sybil and replication attacks can be mounted (e.g., to join an existing cluster with a captured node clone). Moreover, even when the compromised node is detected and its keys are revoked, the revocation list must be maintained for the whole network (e.g., if the revocation list is maintained in a completely decentralized way then ultimately every node must store a copy of the list). A common way and good practice to introduce localized secrets is to not use pre-distributed keys for ordinary communication, but only to secure the key exchange of fresh keys, which are locally generated only by nodes involved in the exchange. If the usage of the pre-distributed keys is allowed only for a certain time (or treated with more scrutiny later), such practice can limit the impact of the node capture as the localized keys have no validity outside the area of their generation. An attacker is forced to mount his attack only during a limited time interval and it is thus reasonable to expect that the number of compromised nodes will be lower. When such localized secrets are established with the support of other (potentially compromised) nodes, the secrecy of the resulting key can be violated. To tackle this threat, secrecy amplification protocols were proposed.

This chapter starts with discussion of the principles, properties and performance results of several secrecy amplification protocols including those proposed by us for a partially compromised network resulting from plaintext key distribution known as Key Infection [3]. We will provide detailed comparison is based on simulations resulting from an S3 simulator (see Section 1.2 for description). Later we discuss the applicability of secrecy amplification protocols to different types of partially compromised networks resulting from a node capture in probabilistic key pre-distribution. The differences between characteristics of compromise (so-called compromise patterns) are described, and the impact of these differences on the success of secrecy amplification protocols is examined. To cover additional compromise patterns, we

turn from manual design of the protocols and focus on an automatic generation of such protocols. We are using a combination of evolutionary algorithms that generate candidate solutions and a network simulator that evaluates them. Such an approach enables us to find personalized protocols that work well for a particular key distribution method and against a given attacker and his tactics, avoiding unnecessary messages and thus significantly reducing the communication overhead and making secrecy amplification protocols more practical. An automated approach helps us to find the new protocol with a better fraction of secured links than all published. Finally, we propose a novel principle of secrecy amplification protocols design. This design exhibits linear instead of exponential increase of protocol messages with increasing network density. An automated approach was used to design new protocols with a comparable fraction of secured links to the original (message expensive) approach.

Analysis of the Pull protocol is based on our paper [16], but is significantly rewritten to incorporate information from additional experiments and to better fit with the remaining chapter text. Initial results for automatic generation of secrecy amplification protocols were published in [57] and is accepted for publication as a chapter in book [50].

3.1 Related work

Secrecy amplification protocols (also known as privacy amplification protocols) were introduced by [3] for weaker attacker model together with the plaintext key exchange as a lightweight key establishment method (so-called Key Infection) for wireless sensor networks. This approach does not require any pre-distributed keys. Nodes are simply distributed over the target deployment plane and perform discovery of neighbours by radio. Every node then generates a separate random key for its each neighbour and sends it *un-encrypted* (as no keys are pre-shared) over a radio link. This key is then used as a basic link key to encrypt subsequent communication. This scheme has minimal memory requirements, requires no pre-distribution operations in a trusted environment and every node can essentially establish a key with any other nodes. Perfect node capture resilience is achieved as any two nodes share different keys. If no attacker with eavesdropping capability is present during such plaintext key exchange then all keys will be secure. On the other side, if an attacker is able to eavesdrop all plaintext exchanges then the scheme will be completely insecure as all keys will be compromised. A modified attacker model based on the assumption that the attacker has limited material/financial resources is used instead. Basic assumption is that not all links are eavesdropped.

The first assumption of the model is the similarity of devices used as network nodes and eavesdropping nodes, especially radio sensitivity. We will reason about this assumption a bit: The higher sensitivity of the radio implies higher energy consumption and eavesdropping nodes will require stronger battery sources implying an increased attack cost. Additionally, the relatively small radio range of legal nodes enables frequent radio channel reuse. An eavesdropping node with a highly sensitive antenna will receive signals from several parallel transmissions on the same channel, rendering the received cumulated signal unreadable. The assumption of similar radio sensitivity is therefore reasonable.

3.1.1 Plaintext key exchange – whispering

Communication between two neighbours might be performed with full radio transmission power (we will call this transmission as *maximal screaming mode*). This mode of transmission is suboptimal both from energy-efficiency and security points of view. Current sensor platforms allow us to control the transmission power to some degree to save on node battery, and this feature can be used to facilitate plaintext key exchange which can be eavesdropped only in a limited area.

We will use the term *whispering* for this message transmission mode between two nodes that is performed with the minimal transmission power necessary to communicate. If the sending node is using lower power than this minimal value, then the receiving node is not able to receive messages successfully with its own antenna. The minimal power strength can be obtained from the following process: node starts sending a hello message with the minimal possible power. If no response is received within a defined time-frame, then power is repeatedly increased by small steps until the particular neighbour can hear the transmission and responds.

This minimal transmission power is used later to exchange the link key in plaintext. An attacker will compromise a link key when he is able to record this key exchange transmission. If the eavesdropping device has the same quality of receiver as legal nodes (antenna, signal amplification) then the eavesdropping device must be positioned at equal or smaller distance to the sending node from the receiving node to eavesdrop transmission (if signal propagation is an ideal sphere). This assumption will be used for simulations of secrecy amplification protocols.

As the plaintext key exchange takes place immediately after network deployment, the attacker's eavesdropping devices must be present from the very beginning, actually placed in the deployment field *before* the deployment of the network. If the exact deployment field is not known in advance, the attacker must cover larger areas by its eavesdropping nodes than the owner of the network. The ratio between legal nodes and attacker's eavesdropping nodes will be then unbalanced toward a higher number of legal nodes. This forms the second assumption of the model.

In real deployment, several hello messages should be sent with the same transmission power, as wireless signal propagation might vary, transmission with minimum possible power is desirable, and several lost messages during the key exchange (short messages themselves) can be tolerated. Also, the key exchange can be actually in the hello message. If multiple messages are used, then a different random key should be used for each message to limit the time frame when a particular key value is in the air, possibly vulnerable to eavesdropping. The receiving node will respond with a hash of the key from the received message to confirm the exchange of the key.

In the Key Infection approach, a weakened attacker model is necessary for the first stage (plaintext key exchange) and in some cases also during secrecy amplification. The attacker then reverts to the mode where all transmissions are eavesdropped by an attacker. The length of this interval, together with the resilience of the used exchange and subsequent secrecy amplification, determines the cost for an attacker to successfully attack the network.

3.1.2 Basics of secrecy amplification protocols

A secrecy amplification protocol is an additional scheme executed by the nodes in the network after the basic link key establishment, plaintext key exchange in case of the Key Infection. Fresh new secrets are generated locally and distributed using existing links with associated security state (secure/compromised). As a result, new link keys are constructed. These are different from original pre-shared or exchanged secrets. Especially in WSNs, secrets usable only locally should be preferred due to the possibility of various Sybil-like attacks. Moreover, some links can be secured, even when the original link was compromised.

What is commonly unknown to the network nodes is the identity of links that are actually compromised. Still, we can execute the amplification protocol as the second layer of defence, even when the link between A and B is secure against the attacker (but we do not know that). If we create a new link key as $K'_{AB} = H(K_{AB}, K)$, where K_{AB} is the original link key, K is a fresh key exchanged during amplification protocol and H is a cryptographically strong one-way function, we will obtain a secure link if either the original link is already secure or K can be securely transported to both A and B over some existing path. Such process poses a significant communication overhead as the number of such paths is significant, but may also significantly improve the overall network security.

Eventually, more iterations of the amplification protocol can be performed. The security of link keys can be further improved as links newly secured in the previous iteration can help to secure a new link in the next iteration.

There is no difference between passive and active attackers for the secrecy amplification protocol with respect to the number of secured links. An active attacker controlling a node is equivalent to a passive one that has compromised all links to the node, thus intercepting all passing messages. A denial-of-service attack can be mounted if intermediate nodes propagate incorrect values, but will be detected after the construction of a new link key, because two non-compromised nodes will not be able to establish a functional key. By gradually removing the keys used in the construction, they can spot the node or path which contributed the defective key and ignore it for later protocol runs. The inverse attack must be considered as well as two compromised nodes may blame a legal node for providing an incorrect key. A link jammed by an adversary is equivalent to a missing connection rendering path unusable for secrecy amplification.

Secrecy amplification protocols can be categorized based on:

Number of distinct paths used to send parts of the final key – if more than one path is used then the protocol performs so-called *multi-path amplification*. An attacker must eavesdrop all paths to compromise the new key value. If two nodes A and B exchange a new key directly in one piece, then only one path is used. Note that multiple virtual paths can be constructed over one physical path [56].

Number of involved intermediate nodes per single path – basic key exchange between A and B requires no intermediate node. If at least one intermediate node is used then the protocol performs so-called *multi-hop amplification*. The path is compromised if an attacker is able to eavesdrop at least one link on the path.

Notation	Description
$\|$	concatenation operator
A, B	identification of nodes for which link key is strengthened during secrecy amplification
C_i	identification of intermediate node(s) used during secrecy amplification
N_C	identification of central node during group-oriented secrecy amplification protocols
N_P	identification of node with special role during group-oriented secrecy amplification protocols
N_{d1_d2}	distance relative identification of a node with distance $d1$ from N_C and $d2$ from N_P
k_{xy}	key exchanged in plaintext from node x to node y
K_{xy}	pairwise key shared between nodes x and y
K'_{xy}	new pairwise key shared between nodes x and y after secrecy amplification
$H(.)$	application of cryptographical strong one-way hash function H
$E_{K_{xy}}(.)$	symmetric encryption function using key K_{xy}

Table 3.1: Notation used for secrecy amplification protocols. Function H should be realized as HMAC construction [6] instead of simple hash function to prevent extension attacks.

3.1.3 The mutual whispering protocol

The simplest secrecy amplification protocol is based on a combination of keys exchanged between two nodes. Mutual whispering secrecy amplification constructs the new key between A and B simply as $K'_{AB} = H(k_{AB}, k_{BA}, K_{AB})$, where k_{AB} is the key exchanged (whispered) from A to B, k_{BA} from B to A, and K_{AB} is an already existing shared key for the link between A and B if such key exists, otherwise only keys k_{AB} and k_{BA} are combined. This protocol was not explicitly mentioned in [3], but was actually used for simulations there (based on provided source code). Mutual whispering is a one-hop two-path protocol – no intermediate node is used and keys from two paths are combined (from A to B and from B to A – these paths overlap).

$$A \rightarrow B : (A, B, k_{AB})$$
$$B \rightarrow A : (B, A, k_{BA})$$
$$A, B \text{ compute } K'_{AB} = H(k_{AB}|k_{BA}|K_{AB})$$

Table 3.2: Message diagram for mutual whispering protocol.

3.1.4 The Push protocol

The multi-hop (two-hop) and multi-path (number of neighbours reachable from both A and B) secrecy amplification protocol was described in [3]. Node A generates q different random values (key parts) and sends each one along a different path over an intermediate node(s) C_i to node B, encrypted with an existing link key(s). All values combined together with the existing key shared between A and B are used to create the new key value. If at least one path is not compromised, the resulting key will be secure. Simulations in [3] for attacker/legal nodes ratio up to 5% are presented, showing that the plaintext key exchange followed by the Push protocol is suitable within this attacker model.

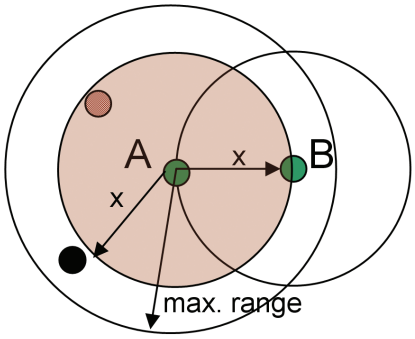

Figure 3.1: Minimal transmission power used during whispering between (green) nodes A and B. Attacker's node positioned inside the inner circle (red node) is able to eavesdrop the transmission between A and B whereas node positioned outside (black node) is not.

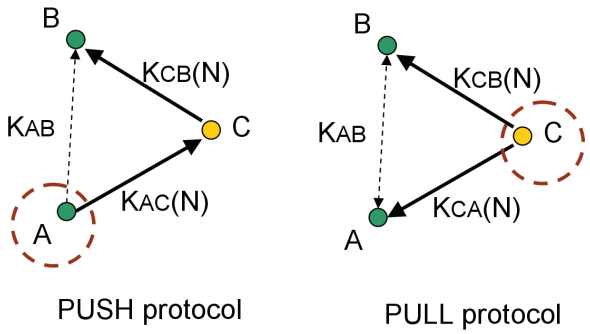

Figure 3.2: Graphic representation of the simplest version of Push and Pull amplification protocols with only one intermediate node C. Red dashed circle highlights the node generating a fresh new random secret N. K_{xy} is the existing directional key between nodes x and y.

$$A \rightarrow C_i : E_{K_{AC_i}}(A, B, N_i)$$
$$C_i \rightarrow B : E_{K_{C_iB}}(A, B, N_i)$$
$$A, B \text{ compute } K'_{AB} = H(K_{AB}|N_i)$$

Table 3.3: Message diagram for two-hop version of the Push protocol.

3.1.5 The Commodity protocol

A variant of initial key exchange mixed with the Push protocol (will be denoted as Commodity) without explicit secrecy amplification is presented in [29]. Node A sends the same key k_i to nodes B and C_i in plaintext using whispering. Then k_i is used to secure distribution of initial key material $E_{k_i}(A, B, K_{i2})$ between (A, B), $E_{k_i}(C_i, A, K_{i3})$ between (C_i, A) and $E_{k_i}(C_i, B, K_{i3})$ between (C_i, B). The final key shared between (A, B) is constructed as $K_{AB} = H(K_{i3}, H(K_{i2}, k_i))$. Formal security proof of the proposed scheme is presented in the paper. The fraction of secured links will be lower than for the Push protocol as the transmission of initial exchange is performed with a higher transmission power (maximum of transmissions required to reach both B and C from A) and therefore is more likely to be compromised. We exclude the Commodity protocol from more detailed analysis, as it is only a variant of the Push protocol, does not provide secrecy amplification as separate and the fraction of secure links will be lower than for the Push protocol alone.

3.1.6 The Pull protocol

A variant of the Push protocol called Pull protocol is presented in our work [16]. The initial key exchange is same as for the Push protocol, but node C_i generates fresh secrets which

$A \rightarrow B : (A, B, C_i, k_i)$
$A \rightarrow C_i : (A, B, C_i, k_i)$
$A \rightarrow B : E_{k_i}(A, B, K_{i2})$
$C_i \rightarrow A : E_{k_i}(C_i, A, K_{i3})$
$C_i \rightarrow B : E_{k_i}(C_i, B, K_{i3})$
A, B compute $K_{AB} = H(K_{i3}, H(K_{i2}, k_i))$

Table 3.4: Message diagram for two-hop version of the Commodity protocol.

are used to improve the secrecy of the key shared between nodes A and B instead of node A as in the Push protocol. The basic idea is that the area where eavesdropping nodes must be positioned to successfully compromise the link key is smaller than for the Push protocol. The resulting fraction of compromised keys is then lower as an attacker has a smaller chance to place eavesdropping nodes properly.

$C_i \rightarrow A : E_{K_{C_iA}}(A, B, N_i)$
$C_i \rightarrow B : E_{K_{C_iB}}(A, B, N_i)$
A, B compute $K'_{AB} = H(K_{AB}|N_i)$

Table 3.5: Message diagram for two-hop version of the Pull protocol.

3.2 Analysis of secrecy amplification protocols

Previous work [3] and [29] dedicated little attention to the behaviour of proposed protocols for different network densities, networks with a higher number of eavesdropping nodes or repeated iterations of amplification. Simulations were performed only for small sizes of the network. Work [29] provided no analysis of the fraction of secured links at all.

We focused on the development of an optimized simulator capable of simulating networks up to hundreds thousands of nodes with variable deployment field size, network density, number of eavesdropping nodes and repetition of amplification process. The simulator aims to simulate such networks in reasonable time to provide more detailed performance estimation, used also for our Pull protocol. See Section 1.2 for simulator details.

The initial results of our simulator for the Push protocol had the same dynamics, but the absolute numbers were different from the original results presented in [3] quite substantially – up to 50-100% of the original results. We asked authors for their original simulator and inspected the source code. We found that their implementation is correct, but they use a mesh with very small resolution to position the nodes and the total number of nodes used was also quite low. When we increased the implicit number of nodes in their simulator, the results varied in tens of percent from the results presented in the original paper. The results presented in this section come from our simulator.

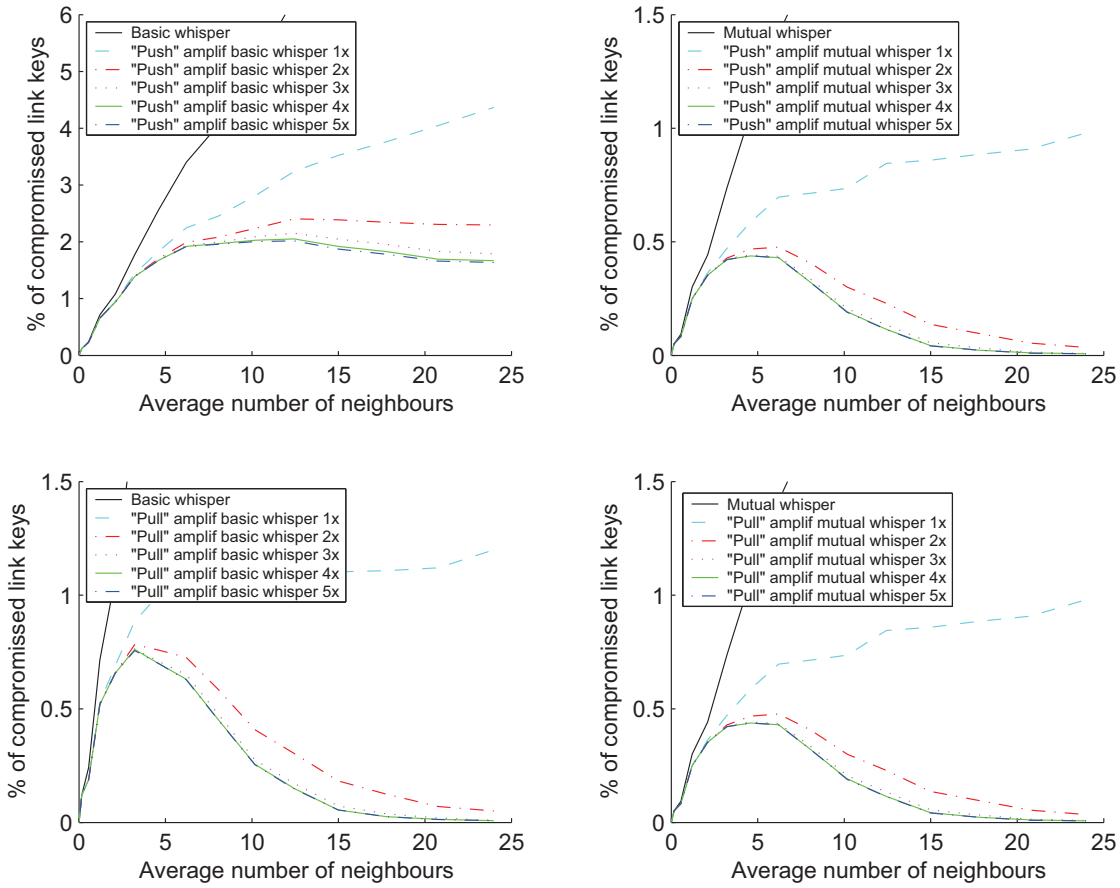

Figure 3.3: Fractions of compromised link keys with 1% of eavesdropping nodes for multiple iteration of the protocols. The Push protocol applied over basic whispering (upper left) and over mutual whispering (upper right) is shown. The Pull protocol applied over basic whispering (lower left) and over mutual whispering (lower right) is shown.

3.2.1 Network settings and simulation setup

Legal and attacker nodes are randomly distributed over a pre-defined area. The neighbour discovery phase is performed for each legal node based on its transmission range. Attacker nodes act just as passive communication eavesdroppers – they represent a passive adversary, but they immediately share all information eavesdropped by any of them so that they can instantly combine key values sent over different paths.

We used network sizes between 10^4 to 10^5 of legal nodes deployed over a square plane with side equal to 25 distance units. Actual variation of average number of neighbours was achieved by changing the transmission range for legal nodes (values between 0.1 and 0.5 were used).

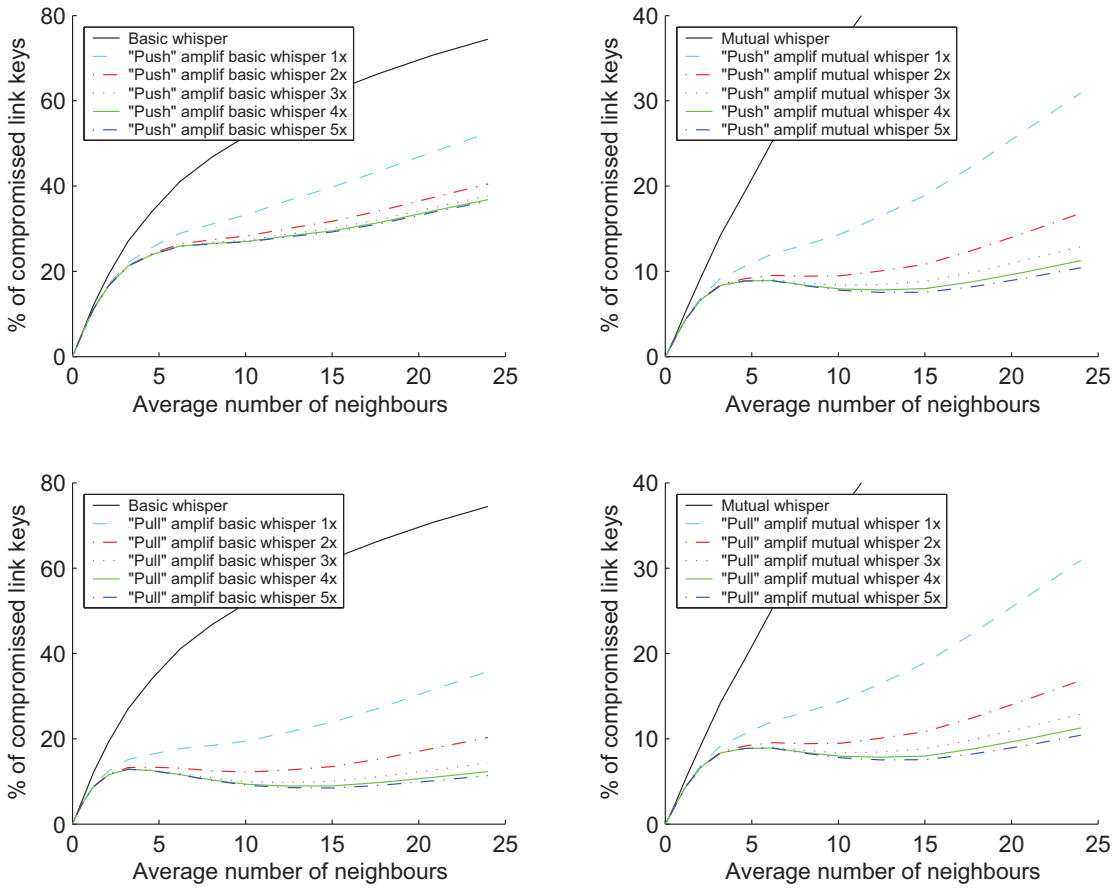

Figure 3.4: Fractions of compromised link keys with 20% of eavesdropping nodes. Results are presented for Push, Pull and mutual whispering protocols, executed after the initial key exchange (basic whisper) or after the initial key exchange with the mutual whisper amplification applied first. The Push protocol applied over basic whispering (upper left) and over mutual whispering (upper right) is shown. The Pull protocol applied over basic whispering (lower left) and over mutual whispering (lower right) is shown.

The simulations were performed with an increasing density of the networks and the resulting graphs are averaged results from at least five distinct simulation runs (note that the high number of nodes in the network and a large deployment plane provides reasonable independence of simulation results from a particular placement of the nodes – therefore five to ten simulations with different nodes layout are sufficient to provide a reasonable average).

3.2.2 Discussion of simulation results

The set of graphs presented in Figures 3.3, and 3.4provides simulation results for all inspected protocols and some selected combinations. Dependency of the fraction of secured links on network density and number of eavesdropping nodes are inspected. Performance of maximum screaming and whispering alone is shown and serves as a baseline (number of secure links if no secrecy amplification protocol is executed). The second set of results is generated from

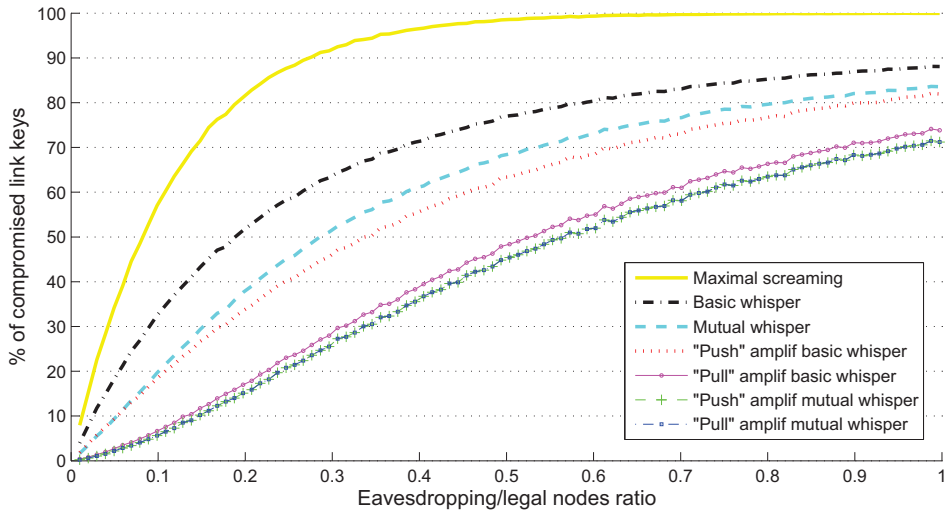

Figure 3.5: Fraction of compromised keys with increasing fraction of eavesdropping nodes. Network density for legal nodes is 8 neighbours on average.

execution of mutual whispering with both Push and Pull protocols over network. Finally, the execution of Push and Pull protocols over network with already performed mutual whispering is examined. Figures 3.5 and 3.2.2 show the behaviour of the protocols with the increasing number of eavesdropping nodes for two network densities with eight and fifteen neighbours respectively on average.

The first set of graphs (Figure 3.3) shows results from a network of 10^4 nodes with 1% of eavesdropping nodes (i.e., there are 100 eavesdropping nodes).

The amplification results are naturally getting worse with an increasing number of eavesdropping nodes in the network as more links are compromised during the initial plaintext key exchange. The second set of graphs (Figure 3.4) shows results for the same settings (10^4 of legal nodes), but with a significantly increased number of eavesdropping nodes at 20% (i.e., there are 2000 eavesdropping nodes).

Based on simulation results for these two scenarios, following findings were observed:

Repetition of secrecy amplification iterations significantly increases the total number of secure links, especially for the lower rates of eavesdropping nodes. The results presented in [3] were done for only one iteration of the amplification protocol. As new links are secured in the first iteration, following iterations have a better starting position than the first one (more secure links). Simulations showed that the increase of secure links can be very significant, decreasing the number of compromised links effectively to zero for scenarios with a low attacker presence and a dense enough network (e.g., 15 neighbours on average). Significant improvement by repetition is possible also for a strong attacker presence (20%).

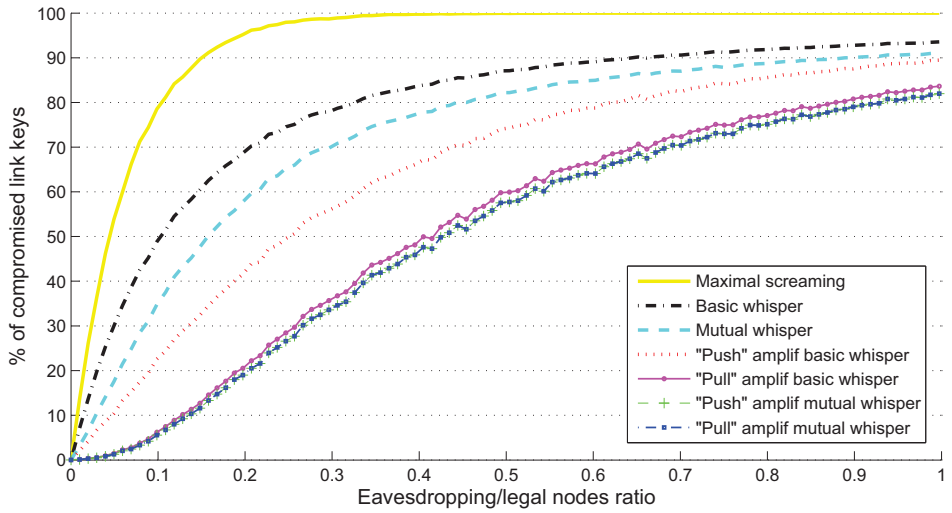

Figure 3.6: Fraction of compromised keys with increasing fraction of eavesdropping nodes. Network density for legal nodes is 15.2 neighbours on average.

Reasonable number of repetitions within tested scenarios is between two and four as additional repetitions do not provide significant increase. The Figures 3.3 and 3.4 show results for up to five iterations of amplification protocols, where only the last two iterations provide a small improvement.

Success of amplification fluctuates with network density. The actual number of links secured by the secrecy amplification protocol as well as the relative factor of improvement (rate between number of secure links before and after amplification) depends heavily on the network density and may fluctuates.

This behaviour is caused by interplay between two factors – a) amplification protocols generally work better with a higher density of the network; b) the attacker will initially compromise a higher fraction of links in dense networks (regardless of the number of eavesdropping nodes). The baseline number of compromised links after the plaintext key exchange (no amplification applied yet) increases quickly with the increasing density of network.

For very sparse networks (up to five nodes in transmission range on average), protocols requiring intermediate nodes often do not find such intermediates between two original nodes and only factor b) applies and so the fraction of compromised links is increasing. With a moderate density of network, amplification protocols have enough intermediates and factor a) is dominating over b), causing a decrease in compromised links. For very dense networks (more than 20 neighbours on average), factor b) will eventually overweight factor a) gain, if enough eavesdropping nodes are present. For a low attacker presence (e.g. 1% of eavesdropping nodes), factor b) never overweight a). For scenarios with a significant attacker presence (e.g., 20%), increasing network density after some threshold does not help to decrease the fraction of compromised links. Such situation is best seen in the graph of the Pull protocol with basic whispering on Figures 3.3 and 3.4.

The Pull protocol outperforms the Push protocol when whispering is used. One iteration of the Pull protocol provides significantly more secure links than one iteration of the Push protocol (factor of two for sparse networks up to factor of four for dense networks). Multiple iterations work significantly better with the Pull protocol than for the Push protocol, especially for 1% eavesdropping nodes. The threshold point of network density where the number of compromised links starts to decrease is also lower (around five for the Pull protocol and around eight for the Push protocol).

Combination of Mutual whispering with other amplification protocol might increase the number of secured links. Combination of the Push protocol with mutual whispering provides exactly the same fraction of secured keys as the Pull protocol alone. The reason comes from the geographical distribution of areas where eavesdropping nodes must be positioned to successfully compromise the link key during amplification. Intersection of compromise areas (area where the attacker eavesdrops a key if positioned inside) for mutual whispering and the Push protocol is exactly same as for the Pull protocol alone. Consequently, amplification success is same as well. This behaviour introduces the idea of efficient composition (stacking) of several secrecy amplification protocols – we will discuss this issue later in Section 3.5.

Mutual whispering is better than Push/Pull protocols alone for sparse networks. Mutual whispering requires no intermediate node and therefore is not impacted by the possibility of unreachable intermediates, which might happen frequently for sparse networks. The combination of a protocol without intermediates (e.g., mutual whispering) with a protocol that uses intermediates (e.g., the Pull protocol) should be generally preferred.

3.2.3 Transmission overhead

The number of messages necessary to execute Push and Pull protocols is same. Also the probability that new key K'_{AB} can be established (not necessarily a secure key) using mediator C_i remains the same as both protocols use at least one intermediate node. Intermediate node C_i is unusable for amplification when no path from A to B over C_i exists. Specifically for the two-hop version of Push and Pull protocols (one intermediate node), C_i cannot be used if C_i is not neighbour of A and B at the same time. This situation can occur with the same probability for both Push and Pull protocols as node identities for the Push protocol can be viewed as permutation of the Pull protocol. Similar situation holds for multi-hop versions of Push and Pull protocols.

Further improvement is possible for the two-hop Pull protocol if the eavesdropping node density is assumed to remain same from the initial plaintext key exchange between neighbours to the amplification phase as well. Messages exchanged during the amplification phase then do not need to be encrypted, as each legal node transmits messages with exactly the same strength as for the initial key exchange. Intermediate C_i can simply transmit the value N_i using one transmission with strength equal to the stronger value used for A or B, preserving the same compromise ratio. If the attacker is able to receive a stronger signal, the new key will be compromised anyway as the attacker already has one of the underlying link key k_{C_iA}

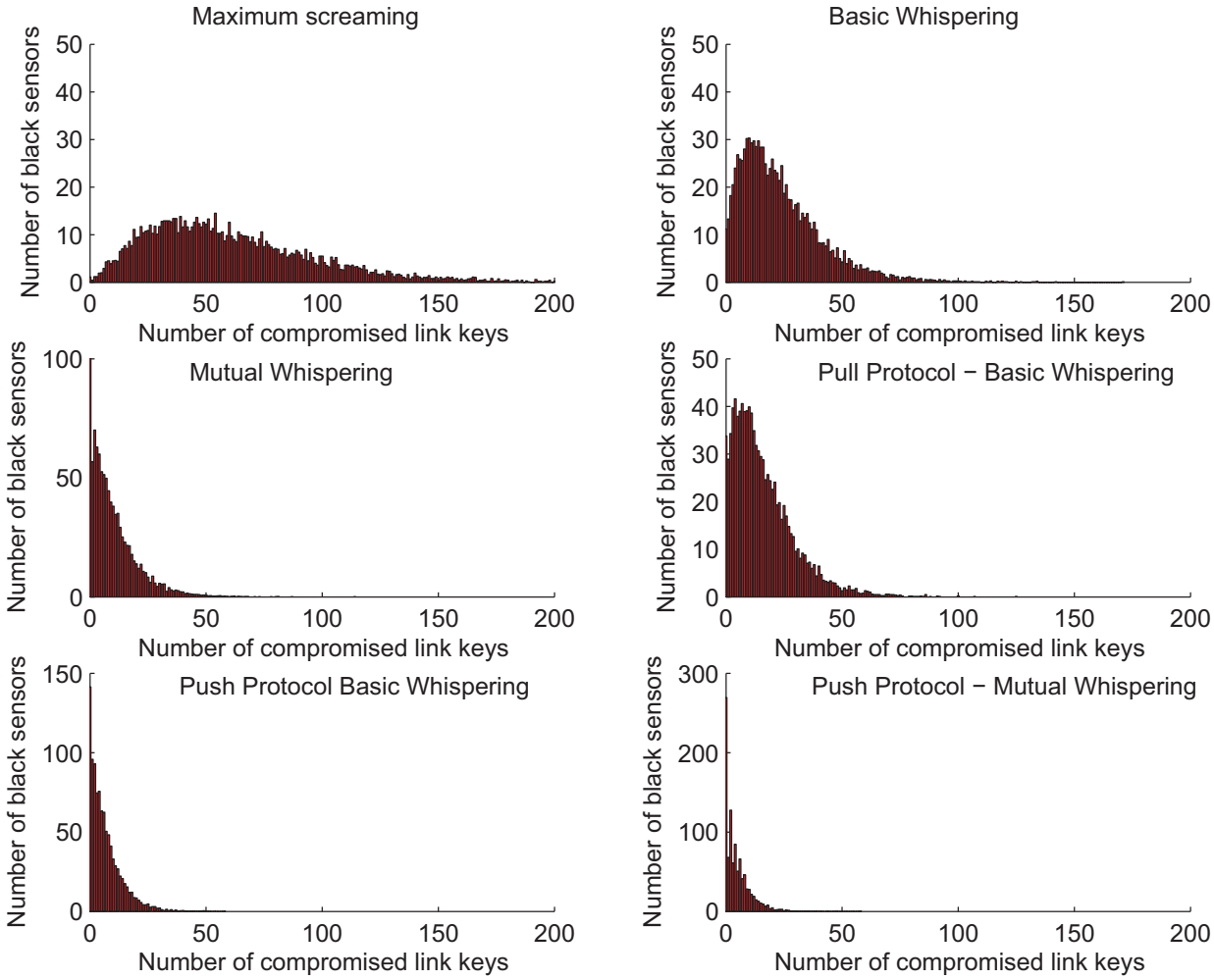

Figure 3.7: Distribution of eavesdropping node success rates.

or k_{C_iB}. One message exchange is thus spared, resulting in a 50% decrease in the number of total messages in case of the two-hop Pull protocol. Note that this optimization cannot be used for multi-hop version of the Pull protocol with more than one intermediate.

3.2.4 Compromise success for eavesdropping nodes

The set of graphs on Figure 3.7 shows the number of eavesdropping nodes, which were able to compromise a particular number of link keys for different secrecy amplification protocols. The results are generated from networks of 10^5 legal nodes with 1000 (1%) eavesdropping nodes. Every variation of the protocols we have been studying is provided in a separate graph.

The first graph covers the situation when keys are sent in clear with the maximum transmission power. The success rate of compromised keys corresponds to a Poisson distribution. Basic whispering shifts the mean value strongly towards a low number of compromised link keys per eavesdropping node, but still almost all eavesdropping nodes are usually responsible for the compromise of several link keys.

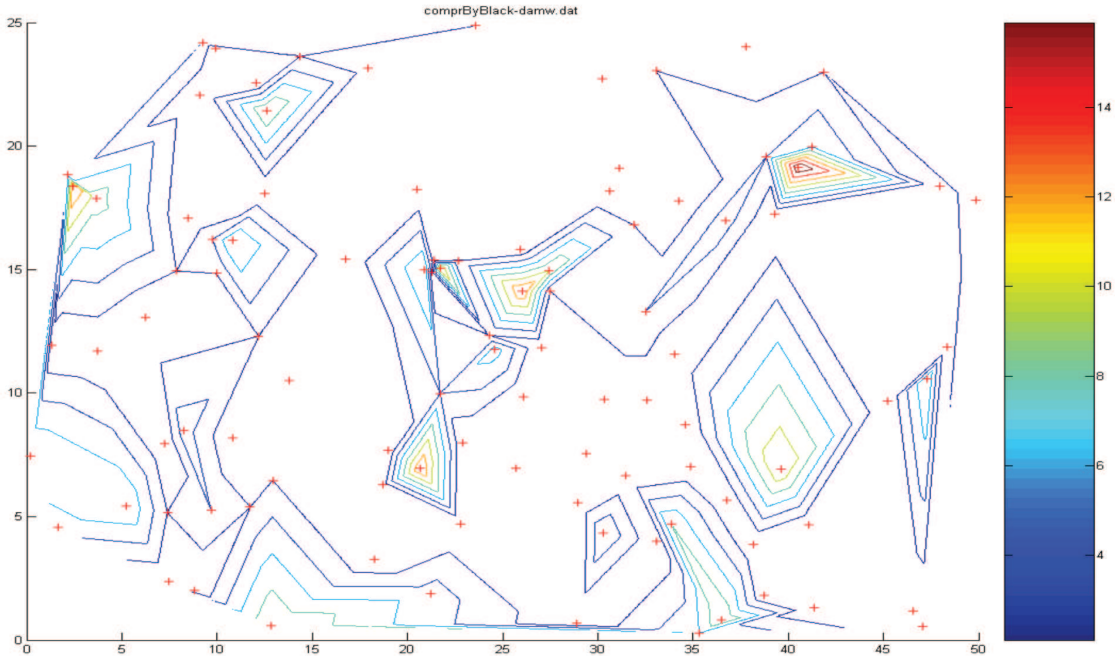

Figure 3.8: Example of non-uniform distribution of the link key compromise for the Pull protocol. Two-dimensional deployment plane is displayed with number of compromised link keys as third axis. Red crosses show position of the eavesdropping nodes. Legal nodes are not shown for the clarity reasons.

Secrecy amplification protocols have a notable impact. A large fraction of eavesdropping nodes is not able to compromise a single key, and the number of really successful eavesdropping nodes (nodes responsible for compromise of a large number of link keys) is rapidly decreasing with the value of compromised link keys. The last graph shows the amplification protocol combined with mutual whispering (Push and Pull protocols have the same results for such combination) where about 300 out of 1000 eavesdropping nodes are not able to eavesdrop a single key. The uneven distribution of compromised links also indicates that there might be large areas in the network without compromised link keys. Visualization of eavesdropping nodes layout together with the number of compromised links on Figure 3.8 provide an example of such non-uniform compromise for the Pull protocol. Such non-uniformity can be exploited to further increase the number of secure links and provides a background for the inspection of different compromise patterns as we later discuss in Section 3.4.

3.3 Key Infection analysis conclusions

One of the goals of the work was to verify simulations from [3] and to provide detailed information about protocols behaviour for the various network parameters. We believe that the results we introduced confirm very good resistance of amplification protocols against a local adversary. The presented results provided an insight in the published protocols and introduced a new secrecy amplification protocol with better faction of secured links.

The most important findings can be summarized as follows. The secrecy amplification generally works better with denser networks, but one cannot improve the ratio of secure keys with density over certain threshold when a certain density and attacker nodes fraction threshold was reached. Multiple iterations of secrecy amplification protocols provide a significant increase in the fraction of secure links with about three repetitions being reasonable. The combination of protocols is possible and a proper combination improves the number of secured links. The combination of the Push protocol with mutual whispering provides same results as the Pull protocol alone for network densities except for sparse networks. Sparse networks should combine mutual whispering with multi-paths/hops protocols to prevent situation with missing intermediate nodes.

Comparing Push, Push and Commodity protocols, Commodity requires the shortest period of the weakened attacker model (transmission of only one key k), but k can be intercepted from a larger distance than keys in Push and Pull protocols. An additional problem exploitable by an attacker is such that an intermediate node C_i knows the value of key between A and B. The Push protocol results in a significantly lower number of compromised link keys than the Pull protocol, especially for denser networks (more than 15 neighbour average).

Secrecy amplification protocols can make a network almost completely secure, when 1% of eavesdropping nodes is assumed. Even when 20% of eavesdropping nodes are present and each legal node has two eavesdropping nodes in transmission range on average, there are still 90% of link keys secure.

Secrecy amplification protocols were originally introduced for the Key Infection plaintext key exchange, but can be used also for a partially compromised network resulting from a node capture for the probabilistic pre-distribution and other partially compromised networks. This idea is later described in Section 3.5.

3.4 Compromise patterns of key distribution

Different key distribution schemes behave differently when the network is under attack targeted to disturb a link key security. The impact on link key security differs based on the attack strategy used. In case of node capture, all links to captured node are compromised. If some probabilistic pre-distribution scheme like [21, 10, 19] is used then some additional links between non-compromised nodes become compromised as well. An eavesdropping of the exchanged key in the Key Infection approach [3] does not compromises nodes directly, but compromises links in reach of eavesdropper radio instead. The characteristics of a particular compromise pattern may significantly influence the success rate of the secrecy amplification executed later. We will focus on two types of network compromise patterns – Random compromise pattern and highly correlated Key Infection pattern.

Bibliography

[1] Crossbow Technology, Inc. *http://www.xbow.com/ [Last Access: 2008-08-31].*

[2] Smart dust project website. *http://robotics.eecs.berkeley.edu/∼pister/SmartDust/ [Last Access: 2008-08-31].*

[3] Ross Anderson, Haowen Chan, and Adrian Perrig. Key infection: Smart trust for smart dust. In *Proceedings of the Network Protocols (ICNP'04), 12th IEEE International Conference, Washington, DC, USA*, 2004.

[4] Tuomas Aura, Pekka Nikander, and Jussipekka Leiwo. Denial of service in sensor networks. *IEEE Computer, Issue 10*, pages 54–62, 2002.

[5] Wolfgang Banzhaf, Peter Nordin, Robert E. Keller, and Frank D. Francone. *Genetic Programming – An Introduction.* Morgan Kaufmann Publishers, San Francisco, CA, 1998.

[6] Mihir Bellare, Ran Canetti, and Hugo Krawczyk. Keying hash functions for message authentication. pages 1–15. Springer-Verlag, 1996.

[7] Peter J. Bentley. *Evolutionary Design by Computers.* Morgan Kaufmann Publishers, San Francisco, CA, 1999.

[8] Rolf Blom. An optimal class of symmetric key generation systems. *EUROCRYPT '84, LNCS 209*, pages 335–338, 1984.

[9] Shaobin Cai, Xiaozong Yang, and Jing Zhao. Mission-guided key management for ad hoc sensor network. *PWC 2004, LNCS 3260*, page 230237, 2004.

[10] Haowen Chan, Adrian Perrig, and Dawn Song. Random key predistribution schemes for sensor networks. In *Proceedings of the 2003 IEEE Symposium on Security and Privacy (SP'03), Washington, DC, USA*, pages 197–214. IEEE Computer Society, 2003.

[11] Haowen Chan, Adrian Perrig, and Dawn Song. *Key distribution techniques for sensor networks.* Kluwer Academic Publishers, Norwell, MA, USA, 2004. ISBN 1-4020-7883-8.

[12] Siu-Ping Chan, Radha Poovendran, and Ming-Ting Sun. A key management scheme in distributed sensor networks using attack probabilities. *GLOBECOM 2005, St. Louis, USA*, 2005.

[13] Elizabeth M. Royer Charles E. Perkins. Ad hoc on-demand distance vector routing. In *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications*, pages 90–100, 1999.

[14] David Chaum, Amos Fiat, and Moni Naor. Untraceable electronic cash. In *CRYPTO '88: Proceedings on Advances in cryptology*, pages 319–327, New York, NY, USA, 1990. Springer-Verlag New York, Inc. ISBN 0-387-97196-3.

[15] Hyun-Jin Choi. Security protocol design by composition. In *University of Cambridge, technical report UCAM-CL-TR-657, GB*, 2006.

[16] Dan Cvrček and Petr Švenda. Smart dust security - key infection revisited. *Security and Trust Management 2005, Italy, ENTCS*, pages 10–23, 2005.

[17] Josh Broch David B. Johnson, David A. Maltz. Dsr: The dynamic source routing protocol for multi-hop wireless ad hoc networks. In Charles E. Perkins, editor, *Ad Hoc Networking*, pages 139–172. Addison-Wesley, 2001.

[18] Jing Deng, Carl Hartung, Richard Han, and Shivakant Mishra. A practical study of transitory master key establishment for wireless sensor networks. In *SECURECOMM '05: Proceedings of the First International Conference on Security and Privacy for Emerging Areas in Communications Networks*, pages 289–302, Washington, DC, USA, 2005. IEEE Computer Society. ISBN 0-7695-2369-2.

[19] Wenliang Du, Jing Deng, Yunghsiang S. Han, and Pramod K. Varshney. A pairwise key pre-distribution for wireless sensor networks. In *Proceedings of the 10th ACM Conference on Computer and Communications Security (CCS'03), Washington, DC, USA*, pages 42–51, 2003.

[20] Wenliang Du, Jing Deng, Yunghsiang S. Han, and Pramod K. Varshney. A key management scheme for wireless sensor networks using deployment knowledge. *IEEE INFOCOM 2004, Hong Kong*, 2004.

[21] Laurent Eschenauer and Virgil D. Gligor. A key-management scheme for distributed sensor networks. In *Proceedings of the 9th ACM Conference on Computer and Communications Security (CCS'02), Washington, DC, USA*, pages 41–47, 2002.

[22] Konstantinos P. Ferentinos and Theodore A. Tsiligiridis. Adaptive design optimization of wireless sensor networks using genetic algorithms. *Computer Networks: The International Journal of Computer and Telecommunications Networking*, 51(4):1031–1051, 2007.

[23] Prahlad Fogla and Wenke Lee. Evading network anomaly detection systems: formal reasoning and practical techniques. In *CCS '06: Proceedings of the 13th ACM conference on Computer and communications security*, pages 59–68, New York, NY, USA, 2006. ACM. ISBN 1-59593-518-5.

[24] Huirong Fu, Satoshi Kawamura, Ming Zhang, and Liren Zhang. Replication attack on random key pre-distribution schemes for wireless sensor networks. *IEEE Information Assurance Workshop, West Point, USA*, 2005.

[25] Ashish Gehani and Surendar Chandra. Past: Probabilistic authentication of sensor timestamps. In *ACSAC '06: Proceedings of the 22nd Annual Computer Security Applications Conference on Annual Computer Security Applications Conference*, pages 439–448, Washington, DC, USA, 2006. IEEE Computer Society. ISBN 0-7695-2716-7.

[26] Dijiang Huang, Manish Mehta, Deep Medhi, and Lein Harn. Location-aware key management scheme for wireless sensor networks. *SASN'04, Washington, DC, USA*, pages 29–42, 2004.

[27] Joengmin Hwang and Yongdae Kim. Revisiting random key pre-distribution schemes for wireless sensor networks. In *Proceedings of the ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN'04), Washington DC, USA*, pages 43–52, 2004.

[28] Rajarshi Das James P. Crutchfeld, Melanie Mitchell. The evolutionary design of collective computation in cellular automata. In *Evolutionary Dynamics, Exploring the Interplay of Selection, Neutrality, Accident, and Function*. New York: Oxford University Press, 2002.

[29] Yong Ho Kim, Mu Hyun Kim, Dong Hoon Lee, and Changwook Kim. A key management scheme for commodity sensor networks. *ADHOC-NOW 2005, LNCS 3738*, pages 113–126, 2005.

[30] J. R. Koza, F. H. Bennett III., D. Andre, and M. A. Keane. *Genetic Programming III: Darwinian Invention and Problem Solving*. Morgan Kaufmann Publishers, San Francisco, CA, 1999.

[31] Jan Krhovják, Petr Švenda, and Vashek Matyáš. The sources of randomness in mobile devices. In *Proceeding of the 12th Nordic Workshop on Secure IT System*, pages 73–84. Reykjavik University, October 2007.

[32] Jan Krhovják, Petr Švenda, Vashek Matyáš, and Ludek Smolík. The sources of randomness in smartphones with Symbian OS. In *Security and Protection of Information 2007*, pages 87–98. University of Defence, May 2007.

[33] Jong-Keun Lee, Min-Woo Lee, Jang-Se Lee, Sung-Do Chi, and Syng-Yup Ohn. Automated cyber-attack scenario generation using the symbolic simulation. In *AIS 2004*, pages 380–389, 2004.

[34] Donggang Liu and Peng 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 Press. ISBN 1-58113-738-9.

[35] Donggang Liu and Peng Ning. Location-based pairwise key establishments for static sensor networks. *1st ACM Workshop Security of Ad Hoc and Sensor Networks Fairfax, Virginia*, pages 72–82, 2003.

[36] Frederic Massicotte, Francois Gagnon, Yvan Labiche, Lionel Briand, and Mathieu Couture. Automatic evaluation of intrusion detection systems. In *ACSAC '06: Proceedings of the 22nd Annual Computer Security Applications Conference on Annual Computer Security Applications Conference*, pages 361–370, Washington, DC, USA, 2006. IEEE Computer Society. ISBN 0-7695-2716-7.

[37] Catherine Meadows. Formal methods for cryptographic protocol analysis: emerging issues and trends. In *IEEE Journal on Selected Areas in Communications, Volume 21, Issue 1*, pages 44–54, 2003.

[38] Ralph C. Merkle. Secure communications over insecure channels. *Communications of the ACM*, 21(4):294–299, 1978.

[39] Julian Miller and Peter Thomson. Cartesian Genetic Programming. In *Proc. of the 3rd European Conference on Genetic Programming EuroGP2000*, LNCS 1802, pages 121–132. Springer-Verlag, 2000.

[40] Tyler Moore. A collusion attack on pairwise key predistribution schemes for distributed sensor networks. In *Proceedings of the Fourth Annual IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOMW'06), Washington, DC, USA*, 2005.

[41] Tyler Moore. Cooperative attack and defense in distributed networks. In *University of Cambridge, Technical report UCAM-CL-TR-718*. University fo Cambridge, 2008.

[42] Roger M. Needham and Michael D. Schroeder. Using encryption for authentication in large networks of computers. *Communications of the ACM, vol. 21, issue 12*, pages 993–999, 1978.

[43] James Newsome, Elaine Shi, Dawn Song, and Adrian Perrig. The sybil attack in sensor networks: Analysis & defenses. In *Proceedings of the third international symposium on Information processing in sensor networks (IPSN'04), Berkeley, California, USA*, pages 259–268, 2004.

[44] Stephan Olariu and Ivan Stojmenovič. Design guidelines for maximizing lifetime and avoiding energy holes in sensor networks with uniform distribution and uniform reporting. In *Proceedings of the 25th IEEE Conference on Computer Communications (INFOCOM'06)*, 2006.

[45] Bryan Parno, Adrian Perrig, and Virgil Gligor. Distributed detection of node replication attacks in sensor networks. In *Proceedings of the 2005 IEEE Symposium on Security and Privacy (SP'05), Washington, DC, USA*, pages 49–63, 2005. ISBN 0-7695-2339-0.

[46] Adrian Perrig, Robert Szewczyk, J.D. Tygar, Victor Wen, and David E. Culler. Spins: Security protocols for sensor networks. *Wireless Networks 8/2002, Kluwer Academic Publishers*, pages 521–534, 2002.

[47] Roberto Di Pietro, Luigi V. Mancini, and Alessandro Mei. Random key-assignment for secure wireless sensor networks. *1st ACM Workshop Security of Ad Hoc and Sensor Networks Fairfax, Virginia*, pages 62–71, 2003.

[48] Shai Rubin, Somesh Jha, and Barton P. Miller. Automatic generation and analysis of nids attacks. In *ACSAC '04: Proceedings of the 20th Annual Computer Security Applications Conference*, pages 28–38, Washington, DC, USA, 2004. IEEE Computer Society. ISBN 0-7695-2252-1.

[49] Joel L. Schiff. *Cellular Automata: A Discrete View of the World*. Wiley & Sons, Ltd, 2008. 0-470-16879-X.

[50] LukáŠ Sekanina, Zdeněk Vašíček, Richard Ružička, Michal Bidlo, Jiří Jaroš, and Petr Švenda. *Evolucni hardware – in final preparation*. Academia, Praha, Czech Republic, 2009.

[51] Oleg Sheyner, Joshua Haines, Somesh Jha, Richard Lippmann, and Jeannette M. Wing. Automated generation and analysis of attack graphs. In *SP '02: Proceedings of the 2002 IEEE Symposium on Security and Privacy*, page 273, Washington, DC, USA, 2002. IEEE Computer Society. ISBN 0-7695-1543-6.

[52] Dawn Xiaodong Song, Sergey Berezin, and Adrian Perrig. Athena: A novel approach to efficient automatic security protocol analysis. *Journal of Computer Security*, 9(1/2):47–74, 2001.

[53] Piotr Szczechowiak, Leonardo B. Oliveira, Michael Scott, Martin Collier, and Ricardo Dahab. Nanoecc: Testing the limits of elliptic curve cryptography in sensor networks. In *LNCS 4913*, pages 305–320, 2008.

[54] Adrian Thompson. *Hardware Evolution: Automatic design of electronic circuits in reconfigurable hardware by artificial evolution*. Distinguished dissertation series. Springer-Verlag, 1998. ISBN 3-540-76253-1.

[55] Jim Torresen, W. Jorgen Bakke, and Luk Sekanina. Recognizing speed limit sign numbers by evolvable hardware. *Lecture Notes in Computer Science*, 2004(3242):682–691, 2004.

[56] Harald Vogt. Exploring message authentication in sensor networks. *ESAS 2004, LNCS 3313*, pages 19–30, 2005.

[57] Petr Švenda. Automatic construction of secrecy amplification protocols. *3th Workshop Mathematical and engineering methods in computer science, MEMICS 2007*, 2007.

[58] Petr Švenda and Václav Matyáš. Authenticated key exchange with group support for wireless sensor networks. *The 3rd Wireless and Sensor Network Security Workshop, IEEE Computer Society Press. Los Alamitos, CA*, pages 21–26, 2007. ISBN 1-4244-1455-5.

[59] Petr Švenda and Václav Matyáš. *From Problem to Solution: Wireless Sensor Networks Security (chapter in book)*. Nova Science Publishers, New York, USA, 2008. ISBN 978-1-60456-458-0.

[60] Petr Švenda and Martin Osovský. A forward onion encryption scheme for wireless sensor networks. *MEMICS 2005*, pages 38–44, 2005.

[61] Ronald Watro, Derrick Kong, Sue-fen Cuti, Charles Gardiner, Charles Lynn, and Peter Kruus. TinyPK: Securing sensor networks with public key technology. In *Proceedings of the 2nd ACM workshop on Security of ad hoc and sensor networks (SASN'04), Washington, DC, USA*, pages 59–64, 2004.

[62] Eiko Yoneki and Jean Bacon. A survey of wireless sensor network technologies: research trends and middleware's role. *Technical Report, UCAM 646, Cambridge*, 2005.

[63] Sencun Zhu, Sanjeev Setia, and Sushil Jajodia. Leap: efficient security mechanisms for large-scale distributed sensor networks. In *CCS '03: Proceedings of the 10th ACM conference on Computer and communications security*, pages 62–72, New York, NY, USA, 2003. ACM. ISBN 1-58113-738-9.