SEVENTH FRAMEWORK PROGRAMME

Information & Communication Technologies
Trustworthy ICT

NETWORK OF EXCELLENCE

A European Network of Excellence in Managing Threats and Vulnerabilities in the Future Internet: *Europe for the World* [†]

Deliverable D6.1: Report on the State of the Art of Security in Sensor Networks

Abstract: This deliverable presents a review of the state of the art in security for wireless sensor networks.

Contractual Date of Delivery	August 2011
Actual Date of Delivery	August 2011
Deliverable Dissemination Level	Public
Editor	Andreas Larsson
Contributors	All *SysSec* partners

The *SysSec* consortium consists of:

FORTH-ICS	Coordinator	Greece
Politecnico Di Milano	Principal Contractor	Italy
Vrije Universiteit Amsterdam	Principal Contractor	The Netherlands
Institut Eurécom	Principal Contractor	France
IPP-BAS	Principal Contractor	Bulgaria
Technical University of Vienna	Principal Contractor	Austria
Chalmers University	Principal Contractor	Sweden
TUBITAK-UEKAE	Principal Contractor	Turkey

[†] The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 257007.

Contents

1 Introduction

1.1 Sensor Networks

A wireless sensor network is a network of small computers, sensor nodes, that can gather information via its sensors, do computations and communicate wirelessly with other sensor nodes. In general a wireless sensor network is an ad hoc network in which the nodes organize themselves without any preexisting infrastructure. Nodes could be deployed randomly, e.g., by being thrown out from a helicopter over an area that is to be monitored. Once in the area, the nodes that survived the deployment procedure communicate with the other nodes that happened to end up in its vicinity, and they establish an infrastructure.

There are many application areas for sensor networks. The possibilities span areas as civil security, health care, agriculture, research, environmental, commercial and military applications [53, 6]. There are many parameters in these areas that a sensor network can monitor, e.g., disaster areas, restricted areas, wildlife, crowds, manufacturing machinery, structural integrity, earthquakes, agriculture, traffic, pollution or even heart rates.

The sensor nodes in a sensor network are often small and quite cheap. They can therefore be used in great numbers over a large area. This can provide fault tolerance, in which the system can withstand loss of sensor nodes without losing coverage of the monitored area or losing functionality of the network. In addition, compared to more centralized long range sensors, such a sensor network can give a high number of more precise local readings over large areas. The areas monitored can be chosen according to needs and can change over time [5]. The possibility of rapid deployment can be of high value for many areas like medical, civil security and military. One example is rapid monitoring of disaster areas.

Sensor nodes, in contrast to computers in general ad hoc networks, are often very limited in computing power and memory capacity. As an example, the popular MICAz sensor node has a 16 MHz processor and only 4 kB of RAM memory and 128 kB of program memory [3]. These limitations restricts the algorithms that feasibly can be used.

Furthermore, the nodes typically run on battery power and communication is usually the most expensive activity of a sensor node. A MICAz node in receive mode uses around 20 mA [1], which would empty 1000 mAh batteries in just 50 hours. The corresponding lifetime for an idle node that does not communicate or sense could be several years. Thus, it is important in many sensor networks to be conservative in communication.

A sensor network often consists of a large number of nodes. Furthermore, nodes eventually run out of batteries and new nodes are deployed to maintain the network. Therefore, even if the nodes are immobile, the network topology changes over time. Thus, algorithms both have to scale well [137] and need to cope with topology changes.

1.2 Security Requirements

Security is critical for many applications of sensor networks. Some concrete examples of applications obviously needing security include border protection, trespassing and burglar alarm systems, surveillance systems, systems dealing with industrial secrets, and law enforcement and military applications in general. However, just as for other kinds of networks and systems, security is important for a much wider set of applications. There are gains in attacking many different kind of systems for different purposes.

An entity that wants to attack the network are called an *adversary*. The adversary can be a human being controlling things manually, but to be able to make a large impact over a large area of the network, she might deploy sensor nodes of her own that are under her control.

Confidentiality and privacy is needed for sensitive, classified or proprietary information, e.g., medical data, sensitive information in civil security, industrial secrets or military information. It is important to be able to withstand attacks that aim to degrade the functionality of the network. Any kind of application can come under attack from someone that wants to disturb the network. For some applications it is critical to keep as much functionality as possible during an attack. Applications, e.g., that monitor restricted areas might have active adversaries that have an interest in making the sensor network report erroneous information and the sensor network plays a critical role in maintaining security and/or safety of the facility.

Sensor networks are deployed in areas that are to be monitored. This usually implies that they are physically available to an adversary. Furthermore, to deploy large number of nodes, they need to be inexpensive. Tamper-proof nodes are therefore often out of the question. The limitations in computing power, memory and battery makes many traditional security algorithms inappropriate for use in sensor networks [122]. This also limits the cryptography possibilities, especially for public key cryptography. Sensor networks often have very different traffic patterns than other networks. Information usually flows between the sensor nodes and a base station, or between nodes close to each other. Another possibility is that someone with a smart device can query the network dynamically. Thus, it temporarily takes the base station role at some place in the network topology to collect data after which it leaves the network. In any case the traffic does not flow between any pair of nodes in general. In addition, information is often aggregated on the way to decrease the total amount of needed traffic. The wireless medium makes it easy for an adversary to eavesdrop on the traffic, to jam communication or to inject messages into the network. This combination of circumstances that holds for many sensor networks opens up a set of security issues that needs to be addressed. It also means that security protocols that are used in other networks, e.g., the Internet, are often unsuitable for the sensor network setting.

The physical access to nodes, the environment and the open communication medium makes security for sensor networks especially tricky. There are many ways an adversary can use compromised nodes to attack the network [31]. The adversary could place her own sensor nodes in the area to disturb or infiltrate the network. The adversary can capture and reprogram nodes that are part of the network. A much stronger node, e.g., a laptop, can be used to infiltrate and attack the network either as a new node or to replace a captured node after extracting secret information, like cryptographic keys. Malicious nodes like this inside the network, *compromised nodes*, are a challenge to deal with and are an important area for research. Compromised nodes can do a lot of damage to the network. They can use and share encrypted information, they can report erroneous information and they can degrade routing in the network. They can behave in arbitrary ways and break protocols that are not resilient to misbehavior. If countermeasures against misbehaving nodes are taken, they can report innocent nodes as misbehaving.

Security is rarely something that can be added on top of insecure systems to be able to withstand attacks. Security needs to be part of most protocols and algorithms in the system. Otherwise the adversary can chose to direct the attention to the unsecured parts. Therefore, it is important to have secure algorithms for all the basic services that are needed in sensor networks.

This is just a short introduction. In the following chapters we are going to look at attacks towards sensor networks in general and look at cryptography, key management, authentication, localization, clock synchronization, clustering, routing, aggregation and self-stabilization in more detail. More information on other security challenges can be found in [32], [151], [136] and [111].

2 Attacks

2.1 The Adversary

An adversary is an entity that attempts to break the security of a system. The purpose may be to extract secret information, to gain unauthorized access to the network or to cause harm to the network. In this chapter we give a brief overview of the adversary and different general attacks against sensor networks. Additional details can be found in [118].

We can distinguish between a *passive* and an *active* adversary:

- A passive adversary only monitors the communication link and listens to every piece of information that passes through. The adversary uses this information offline to try to break confidentiality to gain unauthorized information.
- An active adversary can use all the techniques available to a passive adversary. She can also interfere with the operations of the network by tampering with nodes, sending messages, causing collisions, jamming communications and performing other active attacks. This has the potential to cause much greater harm to the network as it may in turn cause other changes to the network. Here integrity and availability can be attacked in addition to confidentiality.

We can also distinguish between a mote-class adversary and a laptop-class adversary:

- A mote-class adversary has access to one or a few nodes with capabilities similar to the nodes that are deployed in the network.
- Laptop-Class Adversary: This type of adversary has access to a much more powerful device than the sensor nodes, e.g., a laptop. This allows for a larger set of attack techniques.

Finally, we can distinguish between an insider and an outsider adversary:

- An insider adversary is able to compromise or capture nodes of the network or insert new nodes of her own into the network. Once this is done she can attack the network using these nodes.
- An outsider adversary has no such access to nodes inside the network.

2.2 Physical Layer Attacks

2.2.1 Jamming

Jamming is a physical layer attack in which the adversary transmits signals over the wireless medium to prevent other nodes from communicating because of the signal to noise ratio being to low [21].

2.2.2 Tampering

The adversary gains physical access to the nodes where they are deployed. This allows for extracting information, e.g., cryptographic keys, or even reprogramming them and redeploying them. Such reprogrammed *compromised* nodes can be used in insider attacks [34].

2.2.3 Sensor Manipulation

The sensing hardware itself might also be spoofed or attacked. Possibilities range from distant manipulations, e.g., by laser pointers, to local manipulations, e.g., chemical sprays.

2.3 Data-link Layer Attacks

2.3.1 Collisions

In collision attacks the adversary sends messages that collides with specific messages, instead of constantly jamming the medium. The adversary figures out when a message is being sent, either from knowing details about the protocols the sensor nodes are running or simply by listening to the communication medium to hear transmissions that are being started. Then, at the same time as this message is being sent, she sends a message of her own, causing a collision preventing other nodes from receiving the message.

2.3.2 Exhaustion

The batteries of sensor nodes can be exhausted if the network faces continuous collisions and back-off in MAC protocols, potentially resulting in degradation of availability.

2.4 Network Layer Attacks

2.4.1 Selective Forwarding

A malicious nodes can refuse to forward some or all messages that is supposed to be forwarded by it to other nodes. This can break many protocols or result in delays and bandwidth degradation in the network.

2.4.2 Sinkhole

In a sink hole attack a compromised node sends out incorrect routing information to erroneously convince other nodes that it is a good node to route through to, e.g., towards a base station [79]. This allows for larger impact for selective forwarding attacks or to tamper with forwarded messages.

2.4.3 Sybil Attacks

A Sybil attack is when a malicious node creates its own multiple identities and presents them to other nodes in the network [42, 106, 154]. This can give the malicious node a larger influence in many different protocols, e.g., with voting or redundancy, than it would have just using its own identity.

2.4.4 Hello Flood

A laptop-class adversary broadcasts messages with powerful signals reaching a large portion of the network. Being regarded as a neighbor of many nodes it can gain undue influence, especially in routing protocols [79].

3 Cryptography, Key Management and Authentication

A set of different attempts to implement secure communication specifically for wireless sensor networks appears in the literature. Solutions such as TinySec [78], SenSec [88], MiniSec [97], and TinyECC [90] are all designed to run under TinyOS [2], a widely used operating system for sensor nodes. ContikiSec [18] presents a system designed for the Contiki operating system [49].

3.1 Security Properties

Security properties that should be provided by a secure network layer for wireless sensor networks are briefly described below. After that, individual paper contributions are discussed.

3.1.1 Confidentiality

Confidentiality is a basic property of any secure communication system. Confidentiality guarantees that information is kept secret from unauthorized parties. The typical way to achieve confidentiality is by using symmetric key cryptography for encrypting the information with a shared secret key. Symmetric key algorithms are often divided into stream ciphers and block ciphers. In the case of block ciphers, a mode of operation is needed to achieve semantic security (see below).

3.1.2 Semantic Security

Semantic security guarantees that a passive adversary cannot extract partial information about the plaintext by observing the ciphertext [97]. Block ciphers do not hide data patterns since identical plaintext blocks are encrypted into identical ciphertext blocks. Thus, a special mode of operation and an initialization vector (IV) are often used and are needed to provide some randomization. Initialization vectors have the same length as the block and are typically added in clear to the ciphertext.

3.1.3 Integrity

Integrity guarantees that the packet has not been modified during the transmission. It is typically achieved by including a message integrity code (MIC) or a checksum in each packet. The MIC is computed by calling a cryptographic hash function. By comparing the current MIC with the one stated in the packet, malicious altering or accidental transmission errors can be detected. Checksums are designed to detect only accidental transmission errors.

3.1.4 Authenticity

Data authenticity guarantees that legitimate parties should be able to detect when a message is sent by unauthorized parties and reject it. One common way to achieve authenticity is by including a message authentication code (MAC) in each packet. The MAC of a packet is computed using a shared secret key, which could be the same key used to encrypt the plaintext. In such a scheme, anyone that knows this shared secret key can issue a MAC for a message. In contrast, public key authentication algorithms can provide authentication for which anyone that knows the public key can authenticate that a message is from the one entity holding the corresponding private key.

3.2 Symmetric Key Cryptography

In recent years, the increased need of security in wireless sensor networks has prompted research efforts to develop and provide security modules for these platforms. These efforts go from simple stream ciphers to public key cryptography architectures.

SPINS [112], presented in 2002, is the first security architecture designed for wireless sensor networks. It is optimized for resource-constrained environments and it is composed of two secure building blocks: SNEP and Tesla. SPINS offers data confidentiality, two-party data authentication, and data freshness. However, SNEP was unfortunately neither fully specified nor fully implemented [78].

In 2004, TinySec [78] was presented as the first fully implemented link layer security suite for wireless sensor networks. It is written in the nesC language and is incorporated in the official TinyOS release. TinySec provides confidentiality, message authentication, integrity, and semantic security. The default block cipher in TinySec is Skipjack, and the selected mode of operation is CBCCS. Skipjack has an 80-bit key length, which is expected to make the cipher insecure in the near future [76]. In order to generate a MAC, it uses Cipher Block Chaining Message Authentication Code (CBC-MAC), which has security deficiencies [50]. It provides semantic security with an 8-byte initialization vector, but adds only a 2-byte counter overhead per packet. TinySec adds less than 10% energy, latency, and bandwidth overhead.

SenSec [88] is another cryptographic layer, presented in 2005. It is inspired by TinySec, and also provides confidentiality, access control, integrity, and semantic security. It uses a variant of Skipjack as the block cipher, called Skipjack-X. In addition, SenSec provides a resilient keying mechanism.

MiniSec [97] is a secure sensor network communication architecture designed to run under TinyOS. It offers confidentiality, authentication, and replay protection. MiniSec has two operating modes, one tailored for single-source communications, and the other tailored for multi-source broadcast communication. The authors of MiniSec chose Skipjack as the block cipher, but they do not evaluate other block ciphers as part of their design. The mode of operation selected in MiniSec is the OCB shared key encryption mechanism, which simultaneously provides authenticity and confidentiality.

TinyECC [90] is a configurable library for elliptic curve cryptography operations for sensor nodes. It was released in 2008 and targets TinyOS. Compared with the other attempts to implement public key cryptography in wireless sensor networks, TinyECC provides a set of optimization switches that allow it to be configured with different resource consumption levels. In TinyECC, the energy consumption of the cryptographic operations is on the order of millijoules, whereas using symmetric key cryptography is on the order of microjoules [25].

3.3 Key Management

No cryptographic algorithms can of course be used without having the nodes share keys in some way, regardless if it is secret keys for symmetric cryptography or public keys for public key cryptography. There are many different approaches to share keys in a secure manner.

3.3.1 Key Predistribution

In key predistribution solutions the nodes are being loaded with keys before deployment and with these keys the nodes can setup communications and possibly generate new keys. In regards to the risk of having nodes compromised, more sophisticated solutions are needed than merely having one master key shared by all nodes. However, considering the other end of the spectrum, it is not generally feasible for all pair of nodes to share a unique key. That takes up far too much storage space.

In [52], Eschenauer and Gligor present a random predistribution scheme that starts out by drawing a number of keys randomly for each node before deployment from a pool of keys. After deployment nodes discover what keys they share with neighboring nodes. They can then set up secure communications using those shared keys. With properly set parameters the chance of a node sharing at least one key with a certain neighbor is high.

To increase the resiliency against compromised nodes in the network Chan et al. propose in [23] a method in which it is not enough to just share one pregenerated key but a certain number of pregenerated keys. Other methods set a threshold on the number of compromised nodes that can be tolerated. These include [94], [45] and [36].

Various methods, e.g., [92], [70], [43], [142] and [147], aim to reduce the overhead of key predistribution by taking into account roughly which areas different nodes will be deployed in and predistribute keys accordingly to reduce the number of needed keys for nodes to keep track of.

The SecLEACH protocol in [107] adapts the idea in [52] to set up secure communications for the changing clusters generated by the cluster algorithm in [62].

The previous methods were all probabilistic in the sense that there were no guarantees that a certain pair of nodes would share keys with each other. Chan and Perrig presents a method in [22] in which nodes deterministically set up $\sqrt(n)$ different keys per node using other nodes as trusted intermediaries. Here n is the size of the network and each key is a pairwise key shared by only two nodes.

3.3.2 Other Mechanisms

Zhu et al. reason in [152] that in many systems it takes a longer time for an adversary to compromise nodes than for nodes to set up keys between themselves. They use a global predistributed key together with unique node identifiers to set up pairwise keys with direct neighbors and to set up a cluster key for a cluster of nodes. This predistributed key is erased to limit the effect of compromised nodes. They also present an efficient way for the base station to share pairwise keys with each node in the network and discuss how to update a global network key in case of node compromise.

Anderson et al. present a technique in [9] in which keys are generated and transmitted in clear text. Assuming that an eavesdropping adversary cannot eavesdrop everywhere at once, not all keys will be known to an adversary. Nodes then take help from other nodes to reinforce the security of keys so that a key that might be known by the adversary gets updated to a key that is not known even if the adversary listens in on the update messages. In [35], Cvrcek and Svenda verify results from [9] and introduces a variant of the key reinforcement scheme. Miller and Vaidya also exchange keys in the clear in [104], but use multiple channels to make it hard for an eavesdropping adversary to get hold of more than a few of the keys that are being broadcast in its vicinity.

In [108], Oliveira et al. set up keys in clustered heterogeneous networks between nodes and their cluster heads. They use a hybrid approach by partly using predistributed keys and partly setting up new keys between nodes.

More details on key management in wireless sensor networks can be found in the survey by Camtepe and Yener in [16] and the review by Zhang and Varadharajan [150].

3.4 Authentication

Authentication is a keystone for secure protocols. Public key based authentication schemes are very powerful, but may be too expensive for sensor networks.

The SNEP protocol in [112], the LEAP protocol in [152], the TinySec protocol in [78] and an AES-based protocol in [145] provide node to node authentication without resorting to public key cryptography. In [30], an algorithm is presented that is specifically aimed for clustered ZigBee networks.

3.4.1 Broadcast Authentication

Broadcasting is important for many sensor network services. Thus there is a need for authenticating broadcasts in an efficient manner.

In [112], Perrig et al. also introduce the μTESLA algorithm for authenticating broadcasts. The basic idea is as follows. A chain of keys is used in which a new key in the chain is created by using a one-way hash function on the current last key in the chain. Time is divided into timeslots and the keys are assigned for the different timeslots in reverse order. The creator of the keys can in one timeslot send a message with a MAC calculated by a key in the chain. In a later timeslot it can reveal the earlier key in the chain, which only the creator of the key chain can do. In that way it authenticates that it sent the message. The starting key of the chain needs to be distributed and authenticated separately, which requires predistribution. In [91], Liu and Ning reduce the setup requirements and increase the robustness of μTESLA. In [93], Liu and Ning introduce multi-level key chains to allow for better scaling. Liu et al. add revocation possibilities to μTESLA in [96] and using basic μTESLA as a building block allows for better scaling with reduced storage needs and better resiliency against denial of service attacks. Luk et al. present in [98] the RPT protocol, based on μTESLA, that is specially suited for authenticating broadcasts that are sent at regular times.

They also present the LEA protocol that is aimed for broadcasts with low entropy. They discuss different properties of broadcast authentication and what protocols to use depending on the underlying needs of a system.

3.4.2 User Authentication

Separate from the authentication problem, where nodes authenticate themselves to each other, is the user authentication problem, where a user of the network is being authenticated by the nodes in the network. Just as for node to node authentication, different methods are based on tools like public key cryptography, symmetric key cryptography and one way hash functions. The user that is being authenticated can often be assumed to be much more powerful in terms of processing power, memory, storage, etc.

For node to node authentication in a static network there might be no need for any node to be able to authenticate any other node. In contrast, for user authentication, it might be required for any node in the network to be able to authenticate any user. Additional challenges arise when there is a need for privacy for the users. For details on this topic, we refer the reader to [12], [75], [139], [130], [81] and [131].

4 Localization

4.1 The Importance of Localization

Localization is the service providing information about where sensor nodes are located. This is needed to identify where different events happened, both by knowing the location of the nodes sensing the event and, using multiple cooperating sensors, where the event itself took place. Geographic location information is also needed for other services like geographic routing, geographic information querying, geographic key distribution, location-based authentication and checking geographic network coverage. It is also useful if the nodes themselves need to be found, e.g., for repairs or battery changes, or to find resources tagged by sensor nodes.

4.2 Localization Techniques

The easiest method to localize sensor nodes is to use Global Positioning System (GPS). However, this can be unfeasible due to several reasons: (1) it makes the nodes more expensive, (2) it drains batteries much quicker, and (3) it makes the nodes larger. Also, GPS does not work properly in all environments such as indoors, between tall buildings, etc.

There are two basic categories of localization algorithms. The first one is based on so called infrastructure-based techniques in which there are some entities called beacons, possibly a subset of the sensor nodes themselves, that are equipped with GPS or know their location by some other means. With the help of these beacons the location of the regular nodes in the network can be calculated. The second category include autonomous techniques in which no such infrastructure or special hardware are available. Another characteristic is if a protocol is range-dependent or range-independent, i.e., whether there is a need to calculate distances between nodes.

The usual way to measure the location of a node is to collect data from nodes in the neighborhood and use this information to calculate the node's location. The information needed include distances and/or angles to other nodes together with their respective locations. Distances can be calculated using signal strength or receive time of signals. Finally, the location can be calculated using techniques like triangulation, trilateration or multilateration.

4.3 Attacks Against Localization

Attacks include beacon nodes reporting false locations in beacon messages, ordinary nodes reporting false locations for location verification techniques, misrepresenting distances, e.g., by sending with a different transmission power in signal strength base techniques or using delay attacks (see Section 5.3) to misrepresent signal propagation times. Impersonation, wormhole attacks and Sybil attacks can also be used to fool nodes to calculate incorrect locations [8].

4.4 Secure Localization

The SeRLoc protocol in [85] is a range-independent protocol in which the nodes of the network are divided into two sets. One set of nodes have omnidirectional antennas and the other set, the locators, are equipped with directional antennas. The locators send out different beacons in different directions that contain the position of the locator and the broadcasting angle of the antenna. The normal nodes use these beacons to calculate their position. As the non-locator nodes do not participate actively in the protocol, the locators or their messages would be the points that adversaries are most interested in manipulating to attack the protocol. The nodes and locators share a symmetric key that is used to encrypt the location information. The beacons are authenticated by a one-way hash chain. The protocol defends against a set of compromised nodes and wormhole attacks.

In [148], Zeng et al. improve the Monte Carlo based localization technique for mobile sensor networks described in [68]. They add authentication, filter out inconsistent values and add a new sampling method to be used in case of detected attacks.

Chen et al. present three localization techniques in [28], that use detection mechanisms to detect and disregard nodes with malicious behavior. The detection mechanisms look for nodes that send multiple messages when they should only send one, pair of nodes that claim to be further away from each other than possible given that both were heard by the same node, and nodes that do not act consistently with other nodes. Furthermore, nodes that act consistently with already detected misbehaving nodes are also deemed misbehaving. In [27], Chen et al. present a wormhole localization algorithm based on distance inconsistencies and inconsistencies where nodes receive their own messages or the same message multiple times. The algorithm can not deal with packet loss though, but is further refined in [26] where packet loss is taken care of.

Iqbal and Murshed use trilateration in [73] on all possible subsets of size three of the neighboring beacon nodes to find out the area that data from most triplets produce. Thus, many malicious beacons need to collude to sway the result of a node as long as fair number of honest beacons are in range of that node. Simulations compare the algorithm favorably with the EARMMSE algorithm in [95].

Algorithms that use received signal strength to calculate distances for use in localization calculations are vulnerable to attacks that tamper with received signal strength, e.g., by placing absorbing or reflecting materials in the area. In [89], Li suggests that algorithms should instead be implemented using signal strength differences to be resilient against such attacks.

In [74], Jadliwala et al. investigate under which conditions location errors can be bounded in a setting with captured beacon nodes. They show a lower bound on the number of captured nodes and describe a class of algorithms that can bound the location error. They also present and evaluate three algorithms that are in this class.

In [103], Mi et al. present a technique for secure localization (together with location-based key distribution) in networks that are manually deployed with a GPS equipped master node. They defend against wormhole attacks, restrict impact of insider nodes and propose using motion sensors as a backup if the GPS module becomes unusable, possibly due to an attack.

In [141], Wozniak et al. investigate the robustness using least median squares in a multi-hop distance vector technique and present modifications that need to be made in order to withstand attacks.

Above we have described major recent results, but more details can be found in the surveys [8], [14], [126] and [149] that exclusively look at the topic of secure localization.

5 Clock Synchronization

5.1 The Importance of Clock Synchronization

Many wireless sensor network applications and protocols need a shared view of time. Examples include localization schemes, pinpointing and tracking events, scheduling of a shared radio medium, e.g., using Time Division Multiple Access (TDMA), detecting duplicate events. For some applications the precision needs to be very high. Therefore, clock synchronization protocols are crucial for wireless sensor networks. Broadly speaking, existing clock synchronization protocols for more general networks are too expensive for sensor networks because of the nature of the hardware and the limited resources that sensor nodes have.

5.2 Clock Synchronization Techniques

Elson et al. present the reference broadcast synchronization technique in [51], in which beacon nodes are broadcast wirelessly. Due to the wireless medium different recipients will receive the beacon at more or less the same time, thus having a common event to relate to. All recipients of the beacon sample the clock when they receive it, and by comparing their clock samples they can approximate offsets between their respective clocks.

Another technique for approximate clock offsets is the round-trip synchronization technique used by Ganeriwal et al. in the TPSN protocol described in [59]. A message is sent from node A to node B and another message back from B to A. By sampling the clocks at send and receive of the two messages, the clock offset can be approximated, given that the delays for the two messages are close to equal. The delay can also be approximated from this information, given that the clock rates are approximately equal. This can be useful, especially when a long delay can be a sign of an attack.

A third technique that Maroti et al. use in the FTSP protocol in [102] is to have a clock source and then using a hierarchy to flood the time from the source outwards, with nodes synchronizing their time to the closest node higher up in the hierarchy that they received the time from.

The clocks of the nodes can be synchronized using the approximations of clock offsets gained by the above techniques. Elson et al. [51] use linear regression to deal with differences in clock rates. Their basic algorithm synchronizes a cluster. Overlapping clusters with shared gateway nodes can be used to convert timestamps among clusters. Karp et al. [80] input clock samples for beacon receive times into an iterative algorithm, based on resistance networks, to converge to an estimated global time. Römer et al. [116] give an overview of methods that use samples from other nodes to approximate their clocks. They present phase-locked looping (PLL) as an alternative to linear regression and present methods for estimating lower and upper bounds of neighbors' clocks.

5.3 Attacks Against Clock Synchronization

One threat from insider nodes is that they can send out incorrect timestamps used at various points in many of the common clock synchronization techniques. Another threat is that the malicious nodes in some cases can be placed, possibly due to deliberate manipulation of protocol, in important positions in hierarchies used in global synchronization techniques.

A different threat is the so called delay attack (also known as the pulse-delay attack) described in [57]. An adversary can receive (at least part of) a message, jam the medium for a set of nodes before they receive the entire message, and then replay the message slightly later. This requires no inside nodes in the network or any cryptographic keys, but the jamming must happen at a precise moment in time. This attack can also be performed, without the time requirement for the jamming, by two collaborating insider nodes. The first jams the network in a small area and the second, outside this area, receives the message normally. Then the second node sends out the jammed beacon at a later time or forwards it to the first node to send out at a later time.

Additional details on attacks against clock synchronization in wireless sensor networks can be found in [101].

5.4 Secure Clock Synchronization Techniques

Song et al. present in [124] ways to detect bad timestamp values from insider nodes using a roundtrip synchronization approach and two methods to filter out such outlier values. The first uses the generalized extreme student deviate algorithm and the other uses a time transformation technique to filter out timestamps that have too large offset values.

Sun et al. present in [127] two related schemes to withstand attacks from insider nodes. One divides nodes into levels depending on their distance to a clock source node by comparing pairwise clock differences in a chain between the nodes and the source. The other uses a diffusion scheme that allows for any pair of nodes to compare clock differences with each other. The authors also show how to use several source nodes for this second scheme. The first scheme is more efficient and provides better precision, whereas the second provides better coverage. The algorithms are vulnerable to delay attacks though.

Sun et al. present in [128] a two-phase algorithm where one phase uses a roundtrip synchronization technique to give a basic pairwise synchronization between nodes. They present a way to both timestamp and add authentication to messages on the fly while transmitting to be able to timestamp as close as possible to the actual transmission. Phase two adapts the μTesla solution from [112] to get local broadcast authentication (which needs the loose synchronization from phase one) and achieves global synchronization. Key chains of rapidly expiring keys defend against delay attacks.

Sanchez synchronizes nodes both pairwise and, in a clustered network, clusterwise in [119], using the round-trip synchronization technique. They take duty cycling into account so that nodes can be sleeping between synchronization rounds and their technique defends against some nodes in the network being compromised.

Ganeriwal et al. present a family of clock synchronization algorithms in [57] and [58]. They are based on the roundtrip synchronization technique in [59] and filter out over-delayed message exchanges to fend against delay attacks and compromised nodes. They present both single and multi-hop pairwise synchronization techniques as well as group synchronization techniques, where some can deal with insider attacks from compromised nodes and some can not. Byzantine agreement is used to get a group synchronization algorithm that withstands insider attacks in the group synchronization.

Hoepman et al. present in [65] a secure clock synchronization algorithm with a randomized clock sampling algorithm at the core. The algorithm is resilient against both delay attacks and attacks from insider nodes. Moreover, the algorithm is self-stabilizing. The clock sampling allows a combination of getting the precision of the reference broadcast technique were many nodes have common points with the ability of the roundtrip synchronization technique to detect spurious delays.

Hu et al. [66] consider under-water sensor networks where nodes communicate using acoustic means and may be following streaming water. Nodes deployed at different depths move at different speeds. In this setting the propagation delay is variable and far from negligible and must be taken into account. They propose a method that synchronizes clocks vertically, between nodes at different depths. They consider insider attacks from compromised nodes and use various statistical methods to detect and defend against such attacks.

Li et al. build up a hierarchy under a base station in [87] and use overhearing to get verification that nodes do not send out incorrect data. Hu et al. use an FTSP style flooding protocol in [69] and use a system of predicting future clock values to detect attacks from insider nodes. Roosta et al. propose in [117] a set of attack countermeasures for the FTSP protocol and present results from their testbed implementation. Chen and Leneutre propose a method using one-way hash chains in [29] to ensure authenticity and integrity of synchronization beacons. Rasmussen et al. show in [114] methods to protect against attacks towards localization and clock synchronization protocols with the help of external navigation stations. Farrugia and Simon use a cross-network spanning tree in which the clock values propagate for global clock synchronization in [54]. They use passive overhearing to let some nodes synchronize without the need of active participation. They defend against replay and worm-hole attacks. Du et al. discuss in [47] how to take advantage of high-end nodes with GPS to improve efficiency for secure clock synchronization. Secure clock synchronization in wireless sensor networks is also discussed in [15].

6 Clustering

6.1 The Importance of Clustering

Clustering nodes together into groups is an important low level service for wireless sensor networks. Sensor networks, like other ad-hoc networks, need to organize themselves after deployment. Clustering sets up a structure, e.g., for forming backbones, for routing in general, for aggregating data from many nodes to reduce the amount of data that needs to be sent through the network, for building hierarchies that allow for scaling and for nodes to take turns doing energy-intensive tasks.

6.2 Clustering Techniques

One way of clustering nodes in a network is to have nodes associating themselves with one or more cluster heads. In the (k,r)-clustering problem, each node in the network should have at least k cluster heads within r communication hops away. This might not be possible for all nodes if the number of nodes within r hops is smaller than k. In such cases a best effort approach can be taken for getting as close to k cluster heads as possible. Assuming that the network allows k cluster heads for each node, the set of cluster heads forms a (k,r)-dominating set in the network. If the cluster heads need to have k cluster heads as well, it forms a *total* (k,r)-dominating set, in contrast to an ordinary (k,r)-dominating set in which this is only required for nodes not in the set. The clustering should be achieved with as few cluster heads as possible. Finding the global minimum number of cluster heads is in general NP complete, so algorithms usually provide an approximation instead. Many algorithms are limited to providing (1,1)-clustering and some provide (1,r)-clustering, (k,1)-clustering or other subsets of (k,r)-clustering.

Some clustering algorithms provide a number of cluster heads but do not make sure that a certain node has a number of cluster heads within some certain radius, but instead use random approaches to get a good statistical coverage.

Another way of providing clusters is for nodes to assign themselves to different clusters without any nodes being assigned as cluster heads. Often these clusters are based on cliques, sets of nodes that forms a complete graph.

A general overview of clustering in wireless sensor networks can be found in [4] by Abbasi and Younis. A survey on clustering wireless ad-hoc networks in general can be found in [33].

6.3 Attacks against clustering algorithms

As for other services, an adversary can disturb protocols from the outside, e.g., by jamming the network, causing collisions, inserting false messages and replaying possibly altered messages. Apart from defending against such outside attacks, it is important to take attacks by malicious insider nodes into account.

By not following protocol, malicious nodes can make sure to be cluster heads whenever they want in protocols where nodes declare that they are cluster heads with a certain probability. Thus they can gain an undue influence in the network and from there have a better platform to launch attacks against other protocols that is running on top of the clustering service. Instead of assigning cluster heads, other algorithms form clusters of nodes by agreeing upon group membership. For such algorithms, a malicious node can send conflicting information to other nodes so that they cannot agree on which nodes are part of which groups. For multi-hop clustering a malicious node can forward false information on which nodes are cluster heads and which are not.

6.4 Secure Clustering Algorithms

In [129], Sun et al. present a secure clustering algorithm that divides the network into disjoint cliques, sets of nodes that all can communicate directly with each other and where each node belongs to exactly one clique (possibly by itself). No cluster heads are assigned. The algorithm takes compromised nodes into account. The use of signed messages allows for nodes to be able to prove misbehavior of malicious nodes to be able to remove them from consideration.

The SLEACH algorithm that Wang et al. present in [135] is based on the LEACH algorithm in [62]. Time is divided into rounds and in each round nodes become cluster heads with a certain probability. To make sure that no node can become cluster head too often, or for outsider nodes to be able to join the protocol, extensive key exchanges are done with a base station.

Banerjee et al. present in [11] the GS-LEACH protocol. It is another secured version of the LEACH protocol. It is based on key distribution that is done in grids with nodes within the grids taking turns being cluster heads.

Wang and Cho in [134] look at secure clustering from a secure election point of view and present a scheme based on signal strength to defend against attacks that try to split an agreement of election results.

6.5 Self-stabilizing Clustering Algorithms

There is a multitude of existing clustering algorithms for ad-hoc networks of which a number are self-stabilizing. Johnen and Nguyen present a self-stabilizing (1,1)-clustering algorithm that converges fast in [77]. Dolev and Tzachar tackle a lot of organizational problems in a self-stabilizing manner in [41]. As part of this work they present a self-stabilizing (1,r)-clustering algorithm. Caron et al. present a self-stabilizing (1,r)-clustering in [17] that takes weighted graphs into account. Larsson and Tsigas present a self-stabilizing (k,r)-clustering algorithm in [83] and [84].

7 Routing

7.1 The Importance of Routing

Unless the user of the network moves around in the area the network is deployed in and collects data directly from the nodes, information needs to be sent through the network. Therefore the nodes need to solve the routing problem, i.e., how to forward messages through the network when a message needs to travel from some node to another. At times there is only a need for information to flow between each sensor node and the base station. Therefore some algorithms only take care of routing to and from a base station.

7.2 Attacks Against Routing Protocols

We present an overview of different attacks that can be used to interfere with routing protocols below. Many of the attack techniques are being used against many other types of protocols, but some, like sinkhole attacks, are specifically aimed against routing protocols. For further details we refer the reader to [79].

7.2.1 Wormhole Attacks

The idea of the wormhole attack is to tunnel messages via a low latency link between two compromised nodes and replay them in different parts of the network. This can disrupt routing protocols as other nodes will get an incorrect view of the network topology. If one of the compromised nodes is close to the base station, the other compromised node can launch a sinkhole attack (see description below).

7.2.2 Sybil Attacks

By presenting multiple identities to the other nodes of the network a node can increase its chances of being included in many communication paths in the network. Other nodes will not realize that these identities in fact belongs to one physical node.

7.2.3 Clone Attacks

This attack is a relative of the Sybil attack where a node acts using multiple existing identities. Keys or other credentials from different captured nodes are being used by different compromised nodes in many different places in the network to maximize the possible damage. By being located in different regions no legitimate nodes can directly hear different traffic sources using the same credentials. Therefore, by having many compromised nodes presenting themselves as many legitimate nodes each, they can gain a large influence in the network.

7.2.4 Selective Forwarding

A simple form of the selective forwarding attack is for a compromised node to act like a “black hole” by refusing to forward any messages. However, in many protocols this results in the node being regarded as dead and thereafter being excluded from consideration. A more effective attack can be to forward certain messages and drop others to disturb the routing protocol itself or another protocol running on top of the routing protocol.

7.2.5 Hello Flood Attacks

Many protocols, including routing protocols, exchange some form of so called hello messages, where they present themselves to their neighbors. A laptop-class-adversary, generating a much more powerful signals than the normal nodes, can convince many nodes that the laptop is their neighbor and use this fact to get into a position were many nodes include the laptop in their routes.

7.2.6 Sinkhole Attacks

Sinkhole attacks are performed by a compromised node by making itself an attractive choice for routing. The goal of this attack is to direct a lot of traffic to a particular area of the network. This position can be used to launch other attacks such as selective forwarding attacks.

7.2.7 Routing Loop Attacks

The idea behind the routing loop attacks is to create loops in how messages are being routed. The result is that a message are being constantly forwarded around in this loop, draining batteries of nodes involved in the loop and preventing the message from reaching its destination.

7.2.8 Using False Information

A compromised node can send out false information about its battery levels, its distance to a base station or its location or other metrics that are used to decide how to route. This can make it seem more attractive from other nodes' point of view than it really is, resulting in that the compromised node becomes part of many routing paths after which it can launch selective forwarding or other attacks.

7.2.9 Base Station Impersonation

In routing algorithms where the goal is to forward messages towards the base station, a simple attack against an unsecured routing protocol can be claiming to be a base station. In protocols that have many possible base stations it might also be possible for a node to insert itself into the lists of available base station without impersonating any existing base stations.

7.3 Secure Routing Algorithms

Lee and Choi present the SeRINS algorithm in [86] that uses multiple paths to be resilient against attacks by compromised nodes. The algorithm defends against both selective forwarding attacks and injection of false routing data.

The SHEER algorithm by Ibriq and Mahgoub is presented in [72]. It sets up a hierarchy and uses probabilistic transmissions with the aim to preserve energy. It adapts to changes of battery in the network. It does not cope with malicious insider nodes.

Yin and Madria present their SecRout, also known as ESecRout, in [143] which is extended in [144] with more experiments and analysis. It is an algorithm for routing query results from nodes towards the sink. They aim to stop message tampering and selective forwarding attacks by using blacklisting.

Du et al. present the TTSR routing algorithm in [46] that, in a heterogeneous setting, takes advantage of high performance nodes scattered throughout the network together with more limited nodes. It defends against spoofed routing information and selective forwarding, sink-hole, wormhole and hello flood attacks.

The SeRWA algorithm in [99] uses wormhole detection to find routes in the presence of wormhole attacks. It is based on overhearing together with authentication of messages to detect when a node that is supposed to forward a message drops it or tampers with it. Such detected malicious nodes can be excluded and routed around.

In [38], Deng et al. present the hierarchical multiple path routing algorithm INSENS. Here the nodes send their neighbor information to the base station, that in turn chooses the multiple paths for routing. Kumar and Jena use the same basic mechanism for their SCMRP algorithm in [82], but they build up a clustered hierarchy to be more energy efficient. The base station is responsible for the cluster formation process.

For more details on secure hierarchical routing, we point the reader to the survey in [121].

7.3.1 Geographic Protocols

These protocols assume that the nodes know their locations and use the geographical location knowledge to decide what routes messages should be forwarded along.

In [48], Du et al. present the SCR algorithm, together with a key management scheme. The geographic coordinate system is divided into a grid, or cells. They choose redundant paths for sending a message and forward messages by choosing cells rather than individual nodes. They defend against attacks such as sink hole, Sybil, wormhole, selective forwarding, hello flood and clone attacks.

Wood et al. present a family of secure routing protocols in [140] with varying levels of security and varying amounts of state that needs to be stored and kept up to date. The weakest provides probabilistic defenses but does not need to keep any state. And stronger ones provides more security guarantees but requires to keep more state information.

The ATSR geographic routing algorithm is presented in [61] and uses a distributed trust model to defend against attacks. It detects and excludes nodes that do not forward messages correctly or that do not execute the trust protocol correctly. It also takes remaining battery levels into account when making routing decisions to prolong the network lifetime.

8 Aggregation

8.1 The Importance of Aggregation

Often information from the sensor nodes in the network is gathered at a base station (or by some other entity querying the network). The battery constraints of many wireless sensor networks make it very important to limit communications. Instead of having every sensor reading being sent from every node all the way to the base station data aggregation can be used to produce reports from data gathered by many nodes.

8.2 Aggregation Techniques

There are several different aggregation techniques. One family of methods forms a tree rooted in the base station and has parents aggregate data from themselves and their children. Another family has cluster heads appointed by running a clustering algorithm (see Chapter 6) and has these cluster heads take the role as special aggregator nodes. Aggregation schemes can also be classified as single aggregator or multiple aggregator schemes. In the former, aggregation happens once for each piece of data and the report is transferred to the base station. In the latter, aggregation happens multiple times on the way.

More details on general aggregation in wireless sensor networks can be found in [55].

8.3 Secure Aggregation Algorithms

Hu and Evans introduce in [67] an aggregation method that is resilient against malicious outsider nodes in the network and against a single compromised key.

Deng et al. present in [37] methods both for nodes to authenticate themselves towards an aggregator, and for an aggregator to authenticate itself toward nodes it aggregates data for.

There are various methods, [44], [113], [153], [132] and [146], with the common denominator that an aggregator needs some form of certificate from the node it aggregates for.

Data injection attacks are done by compromised insider nodes that inject false data to skew the aggregated value [133, 24]. Algorithms for which the largest possible influence done by data injection attacks is proportional to the number of compromised nodes are said to achieve *optimal security*. The algorithms in [24], [56] and [100] all achieve optimal security. However the amount of communication required for a single node might be $O(\log n)$ and they require two round-trip communication rounds between the base station and the nodes of the network. Miyaji and Omote present in [105] an algorithm that achieves optimal security with an $O(1)$ communication load per node and only one round-trip communication round by assuming a weaker adversary model in which the adversary cannot compromise keys of both a node and its parent node.

8.3.1 Aggregating Encrypted Data

Some aggregation algorithms use homomorphic encryption techniques. Such techniques aggregate encrypted data without the need of decryption. In this way data from one node can be kept secret from other nodes, but still be aggregated. Let D be a decrypting function and E the corresponding encrypting function. The cryptographic algorithm is additively homomorphic if $D(E(a) + E(b)) = a + b$, for any a and b. In the same way it is multiplicatively homomorphic if $D(E(a) \cdot E(b)) = a \cdot b$.

Castelluccia et al. present in [19] how to use an additive homomorphic encryption scheme to let nodes keep their data private while still being able to efficiently calculate functions over the data from different nodes. They support calculating sums, mean variances and standard deviations. Parent nodes in a tree can aggregate encrypted data from their children without any decryption. Moreover, the method defends against outside tampering of any data with an authentication scheme. However, there is no prevention to avoid bad values from a node inside the network.

In [71], Huang et al. present a single aggregator scheme for keeping sensor data private. It provides an encryption method that lets an aggregator evaluate if two of its children provide the same data without revealing the value itself.

In [110], Ozdemir and Xiao present an algorithm that allows for aggregation of data encrypted with different encryption keys in different regions.

Bahi et al. achieve homomorphic encryption using elliptic cryptography in [10].

Other algorithms involving homomorphic encryption include [20], [138], [115] and [13].

8.3.2 Further Reading on Secure Aggregation

More details on secure data aggregation for wireless sensor networks can be found in the surveys [120], [125], [7], and [109].

9 Self-stabilization

Self-stabilizing algorithms [39, 40, 123] cope with the occurrence of transient faults in an elegant way. Starting from an arbitrary state, self-stabilizing algorithms let a system stabilize to and stay in a consistent state as long as the algorithms' assumptions hold for a sufficiently long period.

There are many reasons why a system could end up in an inconsistent state of some kind. Assumptions that algorithms rely on could temporarily be invalid. Memory content could be changed by radiation or other elements of harsh environments. Messages could temporarily get lost to a much higher degree than anticipated. Topology changes happens when nodes eventually run out of memory, if they get physically destroyed in harsh environments or when new nodes are added to the network to maintain coverage. Such topology changes could break assumptions and lead to temporary inconsistencies. It is often not feasible to manually reconfigure large ad-hoc networks to recover from events like this. Self-stabilization is therefore often a desirable property of algorithms for ad-hoc networks and especially for sensor networks [63].

In the sensor network setting assumptions about the system could eventually be violated when an adversary, far more powerful than the limited sensor nodes, starts disturbing the sensor network. An example is a temporary denial of service attack that disturbs communications to a level where assumptions about message throughput are violated. It can be hard to anticipate all possible states the network could end up in after an attack. Large numbers of nodes could get compromised and send incorrect information, nodes could be physically attacked in different ways or the adversary might jam the communication medium. Self-stabilization makes sure that the network can recover from any state as long as assumptions hold once again, e.g., after the adversary has been chased away or more nodes have been added to the network.

As an example, the secure and self-stabilizing clock synchronization algorithm presented in [64] and [65] assumes that there is an upper bound on the fraction of sent messages from each node that are being lost due to malicious collisions or attacks. The underlying assumption is that an adversary wishes to remain undetected and therefore does not jam or produce collisions for all messages of a node. In a situation where this bound assumption does not hold, e.g., if the adversary attacks more messages than that, the algorithm cannot guarantee to deliver the specified level of service. In this case it cannot guarantee to share a complete set of timestamps between neighboring nodes with high probability within a certain time span. When message delivery assumptions once again hold, e.g., after the adversary is detected and chased off, the algorithm can, due to the self-stabilizing property, quickly recover and deliver the promised level of service.

The self-stabilizing (k, r)-clustering algorithm in [84] sets up, for each node in the network, k cluster heads within r hops. By using different communication paths to different cluster heads, both the level of fault tolerance [60] and security towards malicious insider nodes [38] can be increased. The clustering algorithm in [84] assumes a static network topology. However, wireless sensor networks seldom have truly static topologies in the long run, even if they are static during their normal modes of operation. Intermittent topology changes can happen due to, e.g., nodes that break or run out of battery, new nodes that are deployed to replace lost nodes, or nodes that are moved or destroyed by a malicious adversary or by the environment.

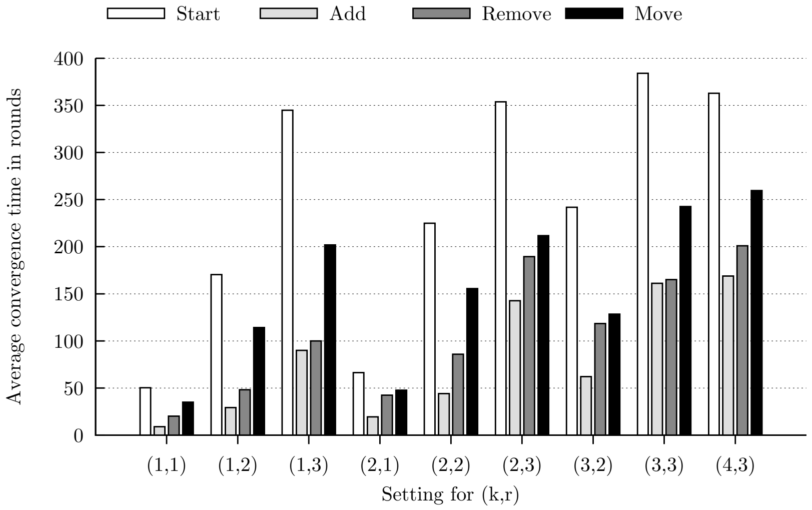

Figure 9.1: Self-stabilization time of the clustering algorithm in [84]. Convergence times from a fresh start, after 5% node additions, after 5% node removes and after 5% node relocations.

The clustering algorithm, however, is self-stabilizing. Thus, it is able to stabilize from any state and can therefore, of course, stabilize after network topology changes. Furthermore, in this case, convergence from small changes to the network topology is typically faster than convergence from any general state. In Figure 9.1 we can see convergence times for the algorithm for different parameters of k and r. The “Start” bars show the convergence times for newly started networks. The others are convergences after small changes to initially converged networks. The topology changes are 5% added nodes (“Add”), 5% removed nodes (“Remove”) and 5% relocated nodes (“Move”). Thus a self-stabilizing algorithm that assumes a static network topology can nevertheless cope with intermittent changes to the topology.

10
Conclusions

In this document we presented a survey of security in wireless sensor networks. We presented an overview of security needs and obstacles in sensor networks. As in most networks, security in wireless sensor networks is of high importance. Resource limitations combined with exposure to the environment and physical access by adversaries raise the need for new security solutions for these kinds of networks.

Furthermore, we described general attacks that are problematic for sensor networks. There are many different services that are needed for sensor network applications as well as for higher-level services. We went into details about cryptography, key management, authentication, localization, clock synchronization, clustering, routing and aggregation. We explained the importance of these services and gave an overview of the state-of-the-art of secure algorithms for the services. Finally, we presented a view on the role self-stabilization can have in secure systems for wireless sensor networks.

Bibliography

[1] MICAz data sheet. www.openautomation.net/uploadsproductos/micaz_datasheet.pdf.

[2] TinyOS. http://www.tinyos.net.

[3] ATmega128(L). http://www.atmel.com/dyn/resources/prod_documents/doc2467.pdf, 2009.

[4] A. A. Abbasi and M. Younis. A survey on clustering algorithms for wireless sensor networks. *Comput. Commun.*, 30(14-15):2826–2841, 2007.

[5] J. Agre and L. Clare. An integrated architecture for cooperative sensing networks. *Computer*, 33(5):106 –108, may. 2000.

[6] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A survey on sensor networks. *Communications Magazine, IEEE*, 40(8):102 – 114, aug. 2002.

[7] H. Alzaid, E. Foo, and J. G. Nieto. Secure data aggregation in wireless sensor network: a survey. In *Proceedings of the sixth Australasian conference on Information security - Volume 81*, AISC '08, pages 93–105, Darlinghurst, Australia, Australia, 2008. Australian Computer Society, Inc.

[8] W. Ammar, A. ElDawy, and M. Youssef. Secure localization in wireless sensor networks: A survey. *CoRR*, abs/1004.3164, 2010.

[9] R. Anderson, H. Chan, and A. Perrig. Key infection: Smart trust for smart dust. *Proceedings of the 13:th IEEE International Conference on Network Protocols*, 0:206–215, 2004.

[10] J. M. Bahi, C. Guyeux, and A. Makhoul. Efficient and robust secure aggregation of encrypted data in sensor networks. In *Proceedings of the 2010 Fourth International Conference on Sensor Technologies and Applications*, SENSORCOMM '10, pages 472–477, Washington, DC, USA, 2010. IEEE Computer Society.

[11] P. Banerjee, D. Jacobson, and S. Lahiri. Security and performance analysis of a secure clustering protocol for sensor networks. In *Network Computing and Applications, 2007. NCA 2007. Sixth IEEE International Symposium on*, pages 145 –152, july 2007.

[12] Z. Benenson, N. Gedicke, and O. Raivio. Realizing robust user authentication in sensor networks. In *Proceedings of Workshop on Real-World Wireless Sensor Networks (REALWSN'05)*, 2005.

[13] R. Bista, K.-J. Jo, and J.-W. Chang. A new approach to secure aggregation of private data in wireless sensor networks. In *Proceedings of the 2009 Eighth IEEE International Conference on Dependable, Autonomic and Secure Computing*, DASC '09, pages 394–399, Washington, DC, USA, 2009. IEEE Computer Society.

[14] A. Boukerche, H. Oliveira, E. Nakamura, and A. Loureiro. Secure localization algorithms for wireless sensor networks. *Communications Magazine, IEEE*, 46(4):96 –101, April 2008.

[15] A. Boukerche and D. Turgut. Secure time synchronization protocols for wireless sensor networks. *Wireless Communications, IEEE*, 14(5):64 –69, october 2007.

[16] S. A. Camtepe and B. Yener. Key distribution mechanisms for wireless sensor networks: a survey. Technical report, Department of Computer Science, Rensselaer Polytechnic Institute, Troy, NY, 2005.

[17] E. Caron, A. K. Datta, B. Depardon, and L. L. Larmore. A self-stabilizing k-clustering algorithm using an arbitrary metric. In *Euro-Par*, pages 602–614, 2009.

[18] L. Casado and P. Tsigas. Contikisec: A secure network layer for wireless sensor networks under the contiki operating system. In *Proceedings of the 14th Nordic Conference on Secure IT Systems: Identity and Privacy in the Internet Age*, NordSec '09, pages 133–147, Berlin, Heidelberg, 2009. Springer-Verlag.

[19] C. Castelluccia, A. C.-F. Chan, E. Mykletun, and G. Tsudik. Efficient and provably secure aggregation of encrypted data in wireless sensor networks. *ACM Trans. Sen. Netw.*, 5:20:1–20:36, June 2009.

[20] C. Castelluccia, E. Mykletun, and G. Tsudik. Efficient aggregation of encrypted data in wireless sensor networks. In *Mobile and Ubiquitous Systems: Networking and Services, 2005. MobiQuitous 2005. The Second Annual International Conference on*, pages 109 – 117, july 2005.

[21] M. Çakiroğlu and A. T. Özcerit. Jamming detection mechanisms for wireless sensor networks. In *Proceedings of the 3rd international conference on Scalable information systems*, InfoScale '08, pages 4:1–4:8, ICST, Brussels, Belgium, Belgium, 2008. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

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

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

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

[25] C.-C. Chang, S. Muftic, and D. Nagel. Measurement of energy costs of security in wireless sensor nodes. In *ICCCN 2007: Proceedings of 16th International Conference on Computer Communications and Networks*, pages 95–102, aug. 2007.

[26] H. Chen, W. Lou, X. Sun, and Z. Wang. A secure localization approach against wormhole attacks using distance consistency. *EURASIP J. Wirel. Commun. Netw.*, 2010:8:1–8:11, April 2010.

[27] H. Chen, W. Lou, and Z. Wang. Conflicting-set-based wormhole attack resistant localization in wireless sensor networks. In *Proceedings of the 6th International Conference on Ubiquitous Intelligence and Computing*, UIC '09, pages 296–309, Berlin, Heidelberg, 2009. Springer-Verlag.

[28] H. Chen, W. Lou, and Z. Wang. A novel secure localization approach in wireless sensor networks. *EURASIP J. Wirel. Commun. Netw.*, 2010:12:1–12:12, February 2010.

[29] L. Chen and J. Leneutre. Toward secure and scalable time synchronization in ad hoc networks. *Comput. Commun.*, 30:2453–2467, September 2007.

[30] W. Chen, X. Zhang, D. Tian, and Z. Fu. An identity-based authentication protocol for clustered zigbee network. In *Proceedings of the Advanced intelligent computing theories and applications, and 6th international conference on Intelligent computing*, ICIC'10, pages 503–510, Berlin, Heidelberg, 2010. Springer-Verlag.

[31] X. Chen, K. Makki, K. Yen, and N. Pissinou. Node compromise modeling and its applications in sensor networks. In *Computers and Communications, 2007. ISCC 2007. 12th IEEE Symposium on*, pages 575 –582, jul. 2007.

[32] X. Chen, K. Makki, K. Yen, and N. Pissinou. Sensor network security: a survey. *Communications Surveys Tutorials, IEEE*, 11(2):52 –73, 2009.

[33] Y. P. Chen, A. L. Liestman, and J. Liu. *Clustering Algorithms for Ad Hoc Wireless Networks*, volume 2, chapter 7, pages 154–164. Nova Science Publishers, 2004.

[34] M. Conti, R. Di Pietro, L. V. Mancini, and A. Mei. Emergent properties: detection of the node-capture attack in mobile wireless sensor networks. In *Proceedings of the first ACM conference on Wireless network security*, WiSec '08, pages 214–219, New York, NY, USA, 2008. ACM.

[35] D. Cvrcek and P. Svenda. Smart dust security – key infection revisited. *Electron. Notes Theor. Comput. Sci.*, 157:11–25, May 2006.

[36] F. Delgosha and F. Fekri. Threshold key-establishment in distributed sensor networks using a multivariate scheme. In *Proceedings of the 25th IEEE International Conference on Computer Communications (INFOCOM 2006)*, Barcelona, Spain, April 2006.

[37] J. Deng, R. Han, and S. Mishra. Security support for in-network processing in wireless sensor networks. In *Proceedings of the 1st ACM workshop on Security of ad hoc and sensor networks*, SASN '03, pages 83–93, New York, NY, USA, 2003. ACM.

[38] J. Deng, R. Han, and S. Mishra. Insens: Intrusion-tolerant routing for wireless sensor networks. *Comput. Commun.*, 29:216–230, January 2006.

[39] E. W. Dijkstra. Self-stabilizing systems in spite of distributed control. *Communications of the ACM*, 17(11):643–644, 1974.

[40] S. Dolev. *Self-Stabilization*. MIT Press, March 2000.

[41] S. Dolev and N. Tzachar. Empire of colonies: Self-stabilizing and self-organizing distributed algorithm. *Theor. Comput. Sci.*, 410(6-7):514–532, 2009.

[42] J. R. Douceur. The sybil attack. In *1st International Workshop on Peer-to-Peer Systems*, IPTPS'02, pages 251–260, 2002.

[43] W. Du, J. Deng, Y. Han, S. Chen, and P. Varshney. A key management scheme for wireless sensor networks using deployment knowledge. In *23:rd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2004)*, volume 1. IEEE, 2004.

[44] W. Du, J. Deng, Y. S. Han, and P. Varshney. A witness-based approach for data fusion assurance in wireless sensor networks. In *In Proceedings of the IEEE Global Telecommunications Conference*, pages 1435–1439, 2003.

[45] 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.*, 8:228–258, May 2005.

[46] X. Du, M. Guizani, Y. Xiao, and H.-H. Chen. Two tier secure routing protocol for heterogeneous sensor networks. *Wireless Communications, IEEE Transactions on*, 6(9):3395 –3401, september 2007.

[47] X. Du, M. Guizani, Y. Xiao, and H.-H. Chen. Secure and efficient time synchronization in heterogeneous sensor networks. *Vehicular Technology, IEEE Transactions on*, 57(4):2387 –2394, july 2008.

[48] X. Du, Y. Xiao, H.-H. Chen, and Q. Wu. Secure cell relay routing protocol for sensor networks: Research articles. *Wirel. Commun. Mob. Comput.*, 6:375–391, May 2006.

[49] A. Dunkels, B. Gronvall, and T. Voigt. Contiki - a lightweight and flexible operating system for tiny networked sensors. In *Proceedings of the 29th Annual IEEE International Conference on Local Computer Networks*, LCN '04, pages 455–462, Washington, DC, USA, 2004. IEEE Computer Society.

[50] M. Dworkin. *NIST Special Publication 800-38B, Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication*. National Institute of Standards and Technology, Computer Security Division, 2005.

[51] J. Elson, L. Girod, and D. Estrin. Fine-grained network time synchronization using reference broadcasts. *Operating Systems Review (ACM SIGOPS)*, 36(SI):147–163, 2002.

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

[53] D. Estrin, R. Govindan, J. Heidemann, and S. Kumar. Next century challenges: scalable coordination in sensor networks. In *MobiCom '99: Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, pages 263–270, New York, NY, USA, 1999. ACM.

[54] E. Farrugia and R. Simon. An efficient and secure protocol for sensor network time synchronization. *Journal of Systems and Software*, 79(2):147–162, 2006.

[55] E. Fasolo, M. Rossi, J. Widmer, and M. Zorzi. In-network aggregation techniques for wireless sensor networks: a survey. *Wireless Communications, IEEE*, 14(2):70–87, April 2007.

[56] K. B. Frikken and J. A. Dougherty, IV. An efficient integrity-preserving scheme for hierarchical sensor aggregation. In *Proceedings of the first ACM conference on Wireless network security*, WiSec '08, pages 68–76, New York, NY, USA, 2008. ACM.

[57] S. Ganeriwal, S. Capkun, C.-C. Han, and M. B. Srivastava. Secure time synchronization service for sensor networks. In *Proceedings of the 4th ACM workshop on Wireless security (WiSe'05)*, pages 97–106, NYC, NY, USA, 2005. ACM Press.

[58] S. Ganeriwal, S. Capkun, and M. B. Srivastava. Secure time synchronization in sensor networks. *ACM Transactions on Information and Systems Security*, 2008.

[59] S. Ganeriwal, R. Kumar, and M. B. Srivastava. Timing-sync protocol for sensor networks. In *SenSys '03: Proceedings of the 1st international conference on Embedded networked sensor systems*, pages 138–149, NYC, NY, USA, 2003. ACM Press.

[60] D. Ganesan, R. Govindan, S. Shenker, and D. Estrin. Highly-resilient, energy-efficient multipath routing in wireless sensor networks. *SIGMOBILE Mob. Comput. Commun. Rev.*, 5:11–25, October 2001.

[61] M. García-Otero, T. Zahariadis, F. Álvarez, H. C. Leligou, A. Población-Hernández, P. Karkazis, and F. J. Casajús-Quirós. Secure geographic routing in ad hoc and wireless sensor networks. *EURASIP J. Wirel. Commun. Netw.*, 2010:10:1–10:12, January 2010.

[62] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks. In *Proceedings of the 33rd Hawaii International Conference on System Sciences*, HICSS '00, pages 8020–, Washington, DC, USA, 2000. IEEE Computer Society.

[63] T. Herman. Models of self-stabilization and sensor networks. In S. R. Das and S. K. Das, editors, *Distributed Computing - IWDC 2003*, volume 2918 of *Lecture Notes in Computer Science*, pages 836–836. Springer Berlin / Heidelberg, 2004.

[64] J.-H. Hoepman, A. Larsson, E. M. Schiller, and P. Tsigas. Secure and self-stabilizing clock synchronization in sensor networks. In *Prooceedings of the 9th International Symposium on Self Stabilization, Safety, And Security of Distributed Systems (SSS 2007)*, volume 4838 of *Lecture Notes in Computer Science*, pages 340 – 356. Springer-Verlag, 2007.

[65] J.-H. Hoepman, A. Larsson, E. M. Schiller, and P. Tsigas. Secure and self-stabilizing clock synchronization in sensor networks. *Theoretical Computer Science*, In Press, Corrected Proof, 2010.

[66] F. Hu, S. Wilson, and Y. Xiao. Correlation-based security in time synchronization of sensor networks. In *WCNC'08*, pages 2525–2530, 2008.

[67] L. Hu and D. Evans. Secure aggregation for wireless networks. In *Proceedings of the 2003 Symposium on Applications and the Internet Workshops (SAINT'03 Workshops)*, SAINT-W '03, pages 384–, Washington, DC, USA, 2003. IEEE Computer Society.

[68] L. Hu and D. Evans. Localization for mobile sensor networks. In *Proceedings of the 10th annual international conference on Mobile computing and networking*, MobiCom '04, pages 45–57, New York, NY, USA, 2004. ACM.

[69] X. Hu, T. Park, and K. Shin. Attack-tolerant time-synchronization in wireless sensor networks. In *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, pages 41 –45, april 2008.

[70] D. Huang, M. Mehta, D. Medhi, and L. Harn. Location-aware key management scheme for wireless sensor networks. In *Proceedings of the 2nd ACM workshop on Security of ad hoc and sensor networks*, SASN '04, pages 29–42, New York, NY, USA, 2004. ACM.

[71] S.-I. Huang, S. Shieh, and J. D. Tygar. Secure encrypted-data aggregation for wireless sensor networks. *Wirel. Netw.*, 16:915–927, May 2010.

[72] J. Ibriq and I. Mahgoub. A secure hierarchical routing protocol for wireless sensor networks. In *Communication systems, 2006. ICCS 2006. 10th IEEE Singapore International Conference on*, pages 1 –6, oct. 2006.

[73] A. Iqbal and M. M. Murshed. Attack-resistant sensor localization under realistic wireless signal fading. In *Proceedings of the 2010 IEEE Wireless Communications and Networking Conference, WCNC 2010*, pages 1–6, Sydney, Australia, April 2010. IEEE.

[74] M. Jadliwala, S. Zhong, S. Upadhyaya, C. Qiao, and J.-P. Hubaux. Secure distance-based localization in the presence of cheating beacon nodes. *Mobile Computing, IEEE Transactions on*, 9(6):810 –823, june 2010.

[75] C. Jiang, B. Li, and H. Xu. An efficient scheme for user authentication in wireless sensor networks. In *Proceedings of the 21st International Conference on Advanced Information Networking and Applications Workshops - Volume 01*, AINAW '07, pages 438–442, Washington, DC, USA, 2007. IEEE Computer Society.

[76] D. Jinwala, D. Patel, and K. Dasgupta. Optimizing the block cipher and modes of operations overhead at the link layer security framework in the wireless sensor networks. In *Proceedings of the 4th International Conference on Information Systems Security*, ICISS '08, pages 258–272, Berlin, Heidelberg, 2008. Springer-Verlag.

[77] C. Johnen and L. H. Nguyen. Robust self-stabilizing weight-based clustering algorithm. *Theor. Comput. Sci.*, 410(6-7):581–594, 2009.

[78] C. Karlof, N. Sastry, and D. Wagner. Tinysec: a link layer security architecture for wireless sensor networks. In *Proceedings of the 2nd international conference on Embedded networked sensor systems*, SenSys '04, pages 162–175, New York, NY, USA, 2004. ACM.

[79] 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, 2003.

[80] R. M. Karp, J. Elson, C. H. Papadimitriou, and S. Shenker. Global synchronization in sensornets. In *Proceedings of the 6th Latin American Symposium on Theoretical Informatics, LATIN04*, volume 2976 of *LNCS*, pages 609–624, Buenos Aires, Argentina, 2004. Springer.

[81] L. Ko. A novel dynamic user authentication scheme for wireless sensor networks. In *Proceedings of IEEE International Symposium on Wireless Communication Systems (ISWCS '08).*, Reykjavik, Iceland, October 2008.

[82] S. Kumar and S. Jena. Scmrp: Secure cluster based multipath routing protocol for wireless sensor networks. In *Wireless Communication and Sensor Networks (WCSN), 2010 Sixth International Conference on*, pages 1 –6, dec. 2010.

[83] A. Larsson and P. Tsigas. Self-stabilizing (k,r)-clustering in wireless ad-hoc networks with multiple paths. In *OPODIS'10, 14th International Conference On Principles Of Distributed Systems*, Tozeur, Tunisia, December 2010.

[84] A. Larsson and P. Tsigas. A self-stabilizing (k,r)-clustering algorithm with multiple paths for wireless ad-hoc networks. In *Proceedings of the 31st International Conference on Distributed Computing Systems (ICDCS 2011)*, Minneapolis, Minnesota, USA, June 2011.

[85] L. Lazos and R. Poovendran. Serloc: Robust localization for wireless sensor networks. *TOSN*, 1(1):73–100, 2005.

[86] S.-B. Lee and Y.-H. Choi. A secure alternate path routing in sensor networks. *Comput. Commun.*, 30:153–165, December 2006.

[87] H. Li, Y. Zheng, M. Wen, and K. Chen. A secure time synchronization protocol for sensor network. In *Proceedings of the 2007 international conference on Emerging technologies in knowledge discovery and data mining*, PAKDD'07, pages 515–526, Berlin, Heidelberg, 2007. Springer-Verlag.

[88] T. Li, H. Wu, X. Wang, and F. Bao. SenSec: Sensor security framework for TinyOS. Technical report, Institute for Infocomm Research, Singapore, 2005.

[89] X. Li. Designing localization algorithms robust to signal strength attacks. In *Proceedings of the Seventh Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks, SECON 2010*, pages 1–3, Boston, Massachusetts, USA, June 2010. IEEE.

[90] A. Liu and P. Ning. TinyECC: A configurable library for elliptic curve cryptography in wireless sensor networks. In *Proceedings of the 7th international conference on Information processing in sensor networks*, IPSN '08, pages 245–256, Washington, DC, USA, 2008. IEEE Computer Society.

[91] D. Liu and P. Ning. Efficient distribution of key chain commitments for broadcast authentication in distributed sensor networks. In *Proceedings of the Network and Distributed System Security Symposium (NDSS 2003)*, San Diego, California, USA, 2003. The Internet Society.

[92] D. Liu and P. Ning. Location-based pairwise key establishments for static sensor networks. In *Proceedings of the 1st ACM workshop on Security of ad hoc and sensor networks*, SASN '03, pages 72–82, New York, NY, USA, 2003. ACM.

[93] D. Liu and P. Ning. Multilevel μtesla: Broadcast authentication for distributed sensor networks. *ACM Trans. Embed. Comput. Syst.*, 3:800–836, November 2004.

[94] D. Liu, P. Ning, and R. Li. Establishing pairwise keys in distributed sensor networks. volume 8, pages 41–77, New York, NY, USA, February 2005. ACM.

[95] D. Liu, P. Ning, A. Liu, C. Wang, and W. K. Du. Attack-resistant location estimation in wireless sensor networks. *ACM Trans. Inf. Syst. Secur.*, 11:22:1–22:39, July 2008.

[96] D. Liu, P. Ning, S. Zhu, and S. Jajodia. Practical broadcast authentication in sensor networks. In *Mobile and Ubiquitous Systems: Networking and Services, 2005. MobiQuitous 2005. The Second Annual International Conference on*, pages 118 – 129, july 2005.

[97] M. Luk, G. Mezzour, A. Perrig, and V. Gligor. MiniSec: A secure sensor network communication architecture. In *Proceedings of the 6th International Conference on Information Processing in Sensor Networks*, IPSN '07, pages 479–488. ACM Press, 2007.

[98] M. Luk, A. Perrig, and B. Whillock. Seven cardinal properties of sensor network broadcast authentication. In *Proceedings of the fourth ACM workshop on Security of ad hoc and sensor networks*, SASN '06, pages 147–156, New York, NY, USA, 2006. ACM.

[99] S. Madria and J. Yin. Serwa: A secure routing protocol against wormhole attacks in sensor networks. *Ad Hoc Netw.*, 7:1051–1063, August 2009.

[100] M. Manulis and J. Schwenk. Security model and framework for information aggregation in sensor networks. *ACM Trans. Sen. Netw.*, 5:13:1–13:28, April 2009.

[101] M. Manzo, T. Roosta, and S. Sastry. Time synchronization attacks in sensor networks. In *Proceedings of the 3rd ACM workshop on Security of ad hoc and sensor networks (SASN'05)*, pages 107–116, NYC, NY, USA, 2005. ACM Press.

[102] M. Maroti, B. Kusy, G. Simon, and A. Ledeczi. The flooding time synchronization protocol. In *Proceedings of the 2nd international conference on Embedded networked sensor systems (SenSys'04)*, pages 39–49, NYC, NY, USA, 2004. ACM Press.

[103] Q. Mi, J. A. Stankovic, and R. Stoleru. Secure walking gps: a secure localization and key distribution scheme for wireless sensor networks. In *Proceedings of the Third ACM Conference on Wireless Network Security, WISEC 2010*, pages 163–168, Hoboken, New Jersey, USA, March 2010. ACM.

[104] M. J. Miller and N. H. Vaidya. Leveraging channel diversity for key establishment in wireless sensor networks. In *Proceedings of the 25th IEEE International Conference on Computer Communications (INFOCOM 2006)*, pages 1–12, Barcelona, Spain, April 2006.

[105] A. Miyaji and K. Omote. Efficient and optimally secure in-network aggregation in wireless sensor networks. In *Proceedings of the 11th international conference on Information security applications*, WISA'10, pages 135–149, Berlin, Heidelberg, 2011. Springer-Verlag.

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

[107] L. Oliveira, H. Wong, M. Bern, R. Dahab, and A. Loureiro. Secleach - a random key distribution solution for securing clustered sensor networks. In *Network Computing and Applications, 2006. NCA 2006. Fifth IEEE International Symposium on*, pages 145 –154, july 2006.

[108] L. B. Oliveira, H. C. Wong, A. A. F. Loureiro, and R. Dahab. On the design of secure protocols for hierarchical sensor networks. *Int. J. Secur. Netw.*, 2(3/4):216–227, 2007.

[109] S. Ozdemir and Y. Xiao. Secure data aggregation in wireless sensor networks: A comprehensive overview. *Computer Networks*, 53(12):2022 – 2037, 2009.

[110] S. Ozdemir and Y. Xiao. Integrity protecting hierarchical concealed data aggregation for wireless sensor networks. *Computer Networks*, 55(8):1735 – 1746, 2011.

[111] A. Perrig, J. Stankovic, and D. Wagner. Security in wireless sensor networks. *Commun. ACM*, 47(6):53–57, 2004.

[112] A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen, and D. E. Culler. Spins: security protocols for sensor networks. *Wirel. Netw.*, 8(5):521–534, September 2002.

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

[114] K. B. Rasmussen, S. Capkun, and M. Cagalj. Secnav: secure broadcast localization and time synchronization in wireless networks. In *Proceedings of the 13th annual ACM international conference on Mobile computing and networking*, MobiCom '07, pages 310–313, New York, NY, USA, 2007. ACM.

[115] I. Rodhe and C. Rohner. n-LDA: n-Layers data aggregation in sensor networks. In *Distributed Computing Systems Workshops, 2008. ICDCS '08. 28th International Conference on*, pages 400 –405, june 2008.

[116] K. Römer, P. Blum, and L. Meier. Time synchronization and calibration in wireless sensor networks. In *Handbook of Sensor Networks: Algorithms and Architectures*, pages 199–237. John Wiley and Sons, September 2005.

[117] T. Roosta, W.-C. Liao, W.-C. Teng, and S. Sastry. Testbed implementation of a secure flooding time synchronization protocol. In *Wireless Communications and Networking Conference, 2008. WCNC 2008. IEEE*, pages 3157 –3162, 31 2008-april 3 2008.

[118] T. Roosta, S. Shieh, and S. Sastry. Taxonomy of security attacks in sensor networks and countermeasures. In *The First IEEE International Conference on System Integration and Reliability Improvements*, pages 13–15, Hanoi, 2006.

[119] D. Sanchez. Secure, accurate and precise time synchronization for wireless sensor networks. In *Proceedings of the 3rd ACM workshop on QoS and security for wireless and mobile networks*, Q2SWinet '07, pages 105–112, New York, NY, USA, 2007. ACM.

[120] Y. Sang, H. Shen, Y. Inoguchi, Y. Tan, and N. Xiong. Secure data aggregation in wireless sensor networks: A survey. *Parallel and Distributed Computing Applications and Technologies, International Conference on*, 0:315–320, December 2006.

[121] S. Sharma and S. K. Jena. A survey on secure hierarchical routing protocols in wireless sensor networks. In *Proceedings of the 2011 International Conference on Communication, Computing &#38; Security*, ICCCS '11, pages 146–151, New York, NY, USA, 2011. ACM.

[122] E. Shi and A. Perrig. Designing secure sensor networks. *Wireless Communications, IEEE*, 11(6):38 – 43, dec. 2004.

[123] Z. Shi and P. K. Srimani. Self-stabilizing distributed systems & sensor networks. In *Handbook on Theoretical and Algorithmic Aspects of Sensor, Ad-Hoc Wireless, and Peer-to-Peer Networks*, chapter 23, pages 393–402. Auerbach Publications, 2005.

[124] H. Song, S. Zhu, and G. Cao. Attack-resilient time synchronization for wireless sensor networks. *Ad Hoc Networks*, 5(1):112–125, 2007.

[125] A. Sorniotti, L. Gomez, K. Wrona, and L. Odorico. Secure and trusted in-network data processing in wireless sensor networks: a survey. *Journal of Information Assurance and Security*, 2(3), September 2007.

[126] A. Srinivasan and J. Wu. *A Survey on Secure Localization in Wireless Sensor Networks*. 2007.

[127] K. Sun, P. Ning, and C. Wang. Secure and resilient clock synchronization in wireless sensor networks. *IEEE Journal on Selected Areas in Communications*, 24(2):395–408, Feb. 2006.

[128] K. Sun, P. Ning, and C. Wang. Tinysersync: secure and resilient time synchronization in wireless sensor networks. In *Proceedings of the 13th ACM conference on Computer and communications security*, CCS '06, pages 264–277, New York, NY, USA, 2006. ACM.

[129] K. Sun, P. Peng, P. Ning, and C. Wang. Secure distributed cluster formation in wireless sensor networks. In *ACSAC '06: Proceedings of the 22nd Annual Computer Security Applications Conference on Annual Computer Security Applications Conference*, pages 131–140, Washington, DC, USA, 2006. IEEE Computer Society.

[130] H. Tseng, R. Jan, and W. Yang. An improved dynamic user authentication scheme for wireless sensor networks. In *IEEE Global Telecommunications Conference (GLOBECOM'07)*, pages 986–990. IEEE, 2007.

[131] B. Vaidya, J. J. Rodrigues, and J. H. Park. User authentication schemes with pseudonymity for ubiquitous sensor network in ngn. *Int. J. Commun. Syst.*, 23:1201–1222, September 2010.

[132] H. Vogt. Exploring message authentication in sensor networks. In *Proceedings of European Workshop on Security of Ad Hoc and Sensor Networks (ESAS)*. Springer-Verlag, 2004.

[133] D. Wagner. Resilient aggregation in sensor networks. In *Proceedings of the 2nd ACM workshop on Security of ad hoc and sensor networks*, SASN '04, pages 78–87, New York, NY, USA, 2004. ACM.

[134] G. Wang and G. Cho. Secure cluster head sensor elections using signal strength estimation and ordered transmissions. *Sensors*, 9(6):4709–4727, 2009.

[135] X. Wang, L. Yang, and K. Chen. Sleach: Secure low-energy adaptive clustering hierarchy protocol for wireless sensor networks. *Wuhan University Journal of Natural Sciences*, 10(1):127–131, 2005. Cited By (since 1996): 3.

[136] Y. Wang, G. Attebury, and B. Ramamurthy. A survey of security issues in wireless sensor networks. *Communications Surveys Tutorials, IEEE*, 8(2):2 –23, 2006.

[137] R. Wattenhofer. Sensor networks: Distributed algorithms reloaded - or revolutions. In *13th Colloquium on Structural Information and Communication Complexity (SIROCCO), United Kingdom*, pages 24–28, 2006.

[138] D. Westhoff, J. Girao, and M. Acharya. Concealed data aggregation for reverse multicast traffic in sensor networks: Encryption, key distribution, and routing adaptation. *Mobile Computing, IEEE Transactions on*, 5(10):1417 –1431, oct. 2006.

[139] K. H. M. Wong, Y. Zheng, J. Cao, and S. Wang. A dynamic user authentication scheme for wireless sensor networks. In *Proceedings of the IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing -Vol 1 (SUTC'06) - Volume 01*, pages 244–251, Washington, DC, USA, 2006. IEEE Computer Society.

[140] A. D. Wood, L. Fang, J. A. Stankovic, and T. He. Sigf: a family of configurable, secure routing protocols for wireless sensor networks. In *Proceedings of the fourth ACM workshop on Security of ad hoc and sensor networks*, SASN '06, pages 35–48, New York, NY, USA, 2006. ACM.

[141] S. Wozniak, T. Gerlach, and G. Schaefer. Optimization-based secure multi-hop localization in wireless ad hoc networks. In N. Luttenberger and H. Peters, editors, *17th GI/ITG Conference on Communication in Distributed Systems (KiVS 2011)*, volume 17 of *OpenAccess Series in Informatics (OASIcs)*, pages 182–187, Dagstuhl, Germany, 2011. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[142] H. Yang, F. Ye, Y. Yuan, S. Lu, and W. Arbaugh. Toward resilient security in wireless sensor networks. In *Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*, MobiHoc '05, pages 34–45, New York, NY, USA, 2005. ACM.

[143] J. Yin and S. Madria. Secrout: A secure routing protocol for sensor networks. In *Proceedings of the 20th International Conference on Advanced Information Networking and Applications - Volume 01*, AINA '06, pages 393–398, Washington, DC, USA, 2006. IEEE Computer Society.

[144] J. Yin and S. K. Madria. Esecrout: An energy efficient secure routing for sensor networks. *Int. J. Distrib. Sen. Netw.*, 4(2):67–82, 2008.

[145] S. M. Youssef, A. B. Mohamed, and M. A. Mikhail. An enhanced security architecture for wireless sensor network. In *Proceedings of the 8th WSEAS international conference on Data networks, communications, computers*, pages 216–224, Stevens Point, Wisconsin, USA, 2009. World Scientific and Engineering Academy and Society (WSEAS).

[146] Z. Yu and Y. Guan. A dynamic en-route scheme for filtering false data injection in wireless sensor networks. In *Proceedings of the 3rd international conference on Embedded networked sensor systems*, SenSys '05, pages 294–295, New York, NY, USA, 2005. ACM.

[147] Z. Yu and Y. Guan. A key management scheme using deployment knowledge for wireless sensor networks. *IEEE Trans. Parallel Distrib. Syst.*, 19:1411–1425, October 2008.

[148] Y. Zeng, J. Cao, J. Hong, S. Zhang, and L. Xie. Secmcl: A secure monte carlo localization algorithm for mobile sensor networks. In *Mobile Adhoc and Sensor Systems, 2009. MASS '09. IEEE 6th International Conference on*, pages 1054 –1059, oct. 2009.

[149] Y. Zeng, J. Cao, J. Hong, S. Zhang, and L. Xie. Secure localization and location verification in wireless sensor networks: a survey. *The Journal of Supercomputing*, pages 1–17, 2010. 10.1007/s11227-010-0501-4.

[150] J. Zhang and V. Varadharajan. Review: Wireless sensor network key management survey and taxonomy. *J. Netw. Comput. Appl.*, 33:63–75, March 2010.

[151] Y. Zhou, Y. Fang, and Y. Zhang. Securing wireless sensor networks: a survey. *Communications Surveys Tutorials, IEEE*, 10(3):6 –28, 2008.

[152] S. Zhu, S. Setia, and S. 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, NYC, NY, USA, 2003. ACM Press.

[153] S. Zhu, S. Setia, S. Jajodia, and P. Ning. An interleaved hop-by-hop authentication scheme for filtering false data injection in sensor networks. In *IEEE Symposium on Security and Privacy (IEEE-SP'04)*, 2004.

[154] W. Znaidi, M. Minier, and J.-P. Babau. An ontology for attacks in wireless sensor networks. Research Report RR-6704, INRIA, 2008.