

Taxonomy of Security Attacks in Sensor Networks and Countermeasures

Tanya Roosta

Shiuhpyng Shieh

Shankar Sastry

Abstract— Ad-hoc sensor networks have become common over the past few years and the domain of their application is increasing widely. However, the security of these networks poses a great challenge due to the fact that they consist of tiny wireless devices which have limited hardware and energy resources. In addition, these networks are generally deployed and then left unattended. These facts coupled together make it impractical to directly apply the traditional security mechanisms to the sensor network paradigm. Therefore, there is a need to analyze and better understand the security requirements of sensor networks. This paper provides a comprehensive taxonomy of security attacks on sensor networks, and gives solutions for each set of attacks. More importantly, it points out the research directions which need to be investigated in the future.

I. INTRODUCTION

Ad hoc networks are infrastructure-less, possibly multi-hop wireless networks where every node can be either a host or a router, forwarding packets to other nodes in the network. Sensor networks are becoming widely integrated into the critical physical as well as personal infrastructures. The vision for the future is to integrate sensors into critical infrastructure, such as Supervisory Control And Data Acquisition systems (SCADA). Some current applications of sensor networks are: providing health care for the elderly, surveillance, emergency disaster relief, detection of chemical or preventing biological threats, and battlefield intelligence gathering.

A sensor network consists of anything from a handful to thousands of tiny wireless devices with sensors. One very popular type of nodes are the motes developed primarily at U.C. Berkeley and Intel, Figure 1. Motes are low cost, small wireless devices with very constrained resources. An example of a sensor mote is the mica2dot. A typical configuration of a mote has a 4MHz, 8-bit processor, with 128KB of instruction memory, 4KB of RAM, and 512KB of external flash memory. The radio has a frequency of 433 MHz and 38.4 Kbps. Given the limited resources of these sensor nodes, in terms of both hardware and energy, it is a key technical challenge to design secure services. Earlier research on sensor networks

This work was supported in part by the Team for Research in Ubiquitous Secure Technology at UC Berkeley (TRUST), the National Science Council (NSC), the Industrial Technology Research Institute (ITRI), and the Taiwan Information Security Center at NCTU (TWISC@NCTU). S. P. Shieh, is with the University of California, Berkeley, CA 94720 USA on leave from the Department of Computer Science and Information Engineering, National Chiao Tung University, Hsinchu, Taiwan (e-mail: ssp@cs.nctu.edu.tw). Tanya Roosta is with the University of California, Berkeley, CA 94720 USA (e-mail:roosta@EECS.Berkeley.EDU). Shankar Sastry is Professor of Electrical Engineering at University of California, Berkeley, CA 94720 (e-mail:sastry@eecs.berkeley.edu)

has focused on developing extremely optimized protocols at different layers of networking stack, as well as a specialized operating system called TinyOs [10]. However, the majority of these protocols have not been designed with security and privacy in mind resulting in substantial performance degradation if there is a security breach. Security can not be designed as a separate module to be added on top of these protocols. Rather, security has to be integrated in the design of every component of the sensor network.

Security in sensor networks has a number of challenges, some of which are: wireless communication among the nodes, lack of pre-existing infrastructure, dynamic topology changes, and resource constraints in terms of memory, energy, and low communication bandwidth.

The goal of this paper is to provide a taxonomy of attacks on sensor network and outline possible solutions for each attack. To the best of our knowledge there has not been a comprehensive taxonomy of attacks for sensor networks. Denial of service attacks have been discussed in [32], however, the authors do not cover a comprehensive list of possible security attacks on sensor networks. In this paper we attempt to give such a comprehensive taxonomy. The main contributions of our work are: 1) to describe the possible attacks on the software (section III), 2) to discuss attacks on data aggregation and its consequences on two important applications in sensor networks (section IX), 3) to show the effects of time synchronization attacks in sensor networks (section X). In addition, the paper is meant to give directions for future research in the area of sensor network security.

The rest of the paper is organized as follows. In Section II we discuss the threat model, trust model, and security objectives in sensor networks. In Section III, physical attacks on the sensor nodes and attacks on TinyOS are covered. Attacks on the communication stack are explained in Section IV. Traffic analysis attacks are described in Section V followed by attack on key management protocols in Section VI. Sybil attack is covered in Section VII. Attack on reputation schemes are explained in Section VIII. The attacks on in-network data aggregation are covered in Section IX. Finally, we explain the attacks on time synchronization protocols and the effect on the higher level application in Section X.



Fig. 1. Mica mote family

II. PROBLEM STATEMENT

A. Threat Model

Attacks on sensor networks can be put into different general categories, as outlined below [17]:

- *A mote-class attacker vs. a laptop-class attacker*: A mote-class attacker has access to a few motes with the same capabilities as other motes in the network. A laptop-class attacker has access to more powerful devices, such as laptops. This will give the adversary an advantage over the sensor network since it can launch more serious attacks.
- *An insider attacker vs. an outsider attacker*: An outsider attacker has no special access to the sensor network, such as passive eavesdropping, but an insider attacker has access to the encryption keys or other code used by the network. For example, an insider attacker could be a compromised node which is a legitimate part of the sensor network .
- *Passive vs. active attacker*: A passive attacker is only interested in collecting sensitive data from the sensor network, which compromises the privacy and confidentiality requirement. In contrast, the active attack goal is to disrupt the function of the networks and degrade the performance. For example, the attacker might inject faulty data into the network by pretending to be a legitimate node.

B. Trust Model

In sensor networks, there are one or more base stations, such as PCs, which are sinks and aggregation points for the information gathered by the nodes. Base stations are the interface between the sensor network and the users. Since base stations are often connected to a larger and less resource-constrained network, it is generally assumed that a base station is trustworthy so long as it is available. Besides the base stations, there are no trust requirements on the sensor nodes since they are vulnerable to physical capture and other attacks.

C. Security Objectives

Security objectives in sensor networks are very similar to security requirements for embedded systems and are summarized below:

- **Data Confidentiality**: In many applications, sensor networks gather sensitive data, for example in health care

or military applications. Data confidentiality ensures that this data is protected and will not leak outside of the sensor network and be used by unauthorized parties. This task could be accomplished using cryptography.

- **Data Authentication**: This requirement allows the receiver to verify that the data was really sent by the node it claims to be coming from. This is accomplished using a Message Authentication Code (MAC) on the communicated data.
- **Data Integrity**: This ensures that the data has not been altered or modified by unauthorized users while in transit.
- **Data Freshness and Availability**: Given that sensor networks are used to monitor time-sensitive events, it is important to ensure that the data provided by the network is fresh and available at all times. This means that an adversary can not replay old messages in the future.
- **Graceful Degradation**: This requirement ensures that the designed mechanisms are resilient to node compromise, and the performance of the networks *degrades gracefully* when a small portion of the nodes are compromised.

III. ATTACKS ON THE MOTE

Sensor networks are self-organizing networks which, once deployed, are expected to run autonomously and without human attendance. Sensor nodes are vulnerable to physical tampering and capture. The physical tampering in term facilitates attacks on the software running on the motes. We will discuss each of these attacks in more detail in the following sub-sections.

A. Physical Tampering

Current sensor hardware does not provide any resistance to physical tampering. If an adversary captures a mote, he can easily extract the cryptographic primitives as well as exploit the shortcomings of the software implementation.

In the realm of sensor networks, the physical attacks can be divided into two types:

- *Invasive Attacks*: This type of attack consists of reverse engineering followed by probing techniques that require access to the chip level components of the device. As a result, the attacker has an unlimited access to the information stored on the chip, and can cause substantial damage to the system.
- *Non-invasive Attacks*: In this type of attack the embedded device is not opened and physically tampered with. An example of this type of attack is the side-channel attack. A side channel attack refers to any attack that is based on the information gathered from the physical implementation of a cryptosystem, in contrast to a vulnerability in the algorithms. For example, the attacker may analyze the power consumption, the timing of the software operation execution, or the frequency of the Electro Magnetic (EM) waves.

Both types of attacks map directly to the sensor network domain. Invasive attack is possible through the physical capture of a sensor node. As of yet, there is no solution available to make the sensor nodes resistant to physical tampering; the sensor nodes' micro-controllers lack any kind of hardware-based memory protection. Traditionally in embedded systems cryptoprocessors, which are physically secure processors, have been extensively used to provide some level of physical tamper resistance. Even though there are known attacks on cryptoprocessors, they do provide a first line of defense against physical tampering. Therefore, there is a need to develop optimized cryptoprocessors that fit the low-cost, low-energy requirements of sensor networks.

Non-invasive attacks, such as side-channel attacks, are also possible in sensor networks. For example, a recent study has shown that a side-channel attack on Message Authentication Codes (MAC), using simple Power Analysis as well as Differential Power Analysis, is possible in sensor networks [24]. Their results suggests that several key bits can be extracted through the power analysis attack. This leads to the conclusion that protecting block ciphers against side channel attacks is not adequate. The future research has to explore possible security measures for Message Authentication Codes as well.

Other side-channel attacks which have not been explored in the context of sensor networks are timing attacks and frequency-based attacks. The timing attack involves algorithms which have non-constant execution time and this can potentially leak secret information. Non-constant execution time can be caused by conditional branching and various optimization techniques. As we will outline in the next subsection, the operating system running on the sensor nodes is event-driven and extremely optimized in terms of memory consumption. This suggests that the timing side-channel attack is possible. A solution to this attack is to use constant execution time software. However, it is not clear if this is easily achievable in sensor networks. Therefore, searching for countermeasures for the timing attack in sensor networks is an important area for future research. In addition to timing attack, frequency-based attacks to extract secret keys of symmetric cryptographic algorithms needs to be further explored in the context of sensor networks. It has been shown in ?? that the frequency-based attack can be carried out on small devices, such as PDA. However, no experiment has been carried out to verify whether this attack is plausible in sensor network context.

Another problem that can arise in sensor network is attacks on the block cipher ¹. TinySec [16] which is the primary encryption mechanism in sensor networks uses block cipher. Attacks on the block cipher are usually accomplished through linear or differential crypto-analysis. As we mentioned earlier, this attack is possible in sensor networks by using power analysis techniques. In addition, if the block cipher is used as a hash function, attacking the block cipher will result in breaking the hash function.

¹In cryptography, block cipher refers to a symmetric key cipher that operates on fixed length groups of bits.

Given the discussion in this section, it is obvious that there needs to be more research done to prevent these attack. Some of the countermeasures for side-channel attacks used in traditional and embedded systems are:

- power consumption randomization
- randomization of the execution of the instruction set
- randomization of the usage of register memory
- CPU clock randomization
- using fake instructions
- using bit splitting

Future research should look into each of these solutions to determine their applicability to the sensor node platforms. Future sensor node hardware has to be designed so as to support the security required by the software.

B. Software Attacks

Software-based attacks are concerned with modifying the code, and exploiting known vulnerabilities. A well-known example of this type of attack is the *buffer overflow* attack. Buffer overflow attack refers to the scenario where a process attempts to store data beyond the boundaries of a fixed length buffer. This results in the extra data overwriting the adjacent memory locations.

As mentioned earlier, sensor networks have limited resources in terms of energy and memory (RAM and flash memory). Consequently, a new OS called TinyOS has been specifically designed for these networks. TinyOS is a low-power, event-driven OS which consists of code that can be reused. TinyOS is a set of components that can be wired together as needed by the application, as shown in Figure 2 [12]. The implementation language of TinyOS is NesC, which is a component based language with event-based execution model [10]. The current implementation of TinyOS does not provide any memory access control, meaning there is no function to control which users/processes access which resources on the system, and what type of execution rights they have. In TinyOS the assumption is largely that a single application or user controls the system.

The solution to the access control in traditional Operating System has been to authenticate the processes, and then mediate their access to different system resources. Another example is to use *Protection Ring* method. A protection ring consists of a number of hardware-enforced levels, for example $0, \dots, m$, of privilege within the architecture of a computer CPU. These ring levels are arranged in a hierarchy starting from the most trusted process (usually at level 0) to the least trusted process (usually at level m). The hardware that uses protection ring greatly restricts the ways in which one ring passes control to another ring.

In addition, the hardware enforces restrictions on the types of memory access that can be performed across rings. In order to effectively implement ring architecture, there needs to be a close cooperation between hardware and software. A possible future direction is to design the hardware platform of the sensor nodes such that it supports the ring architecture.

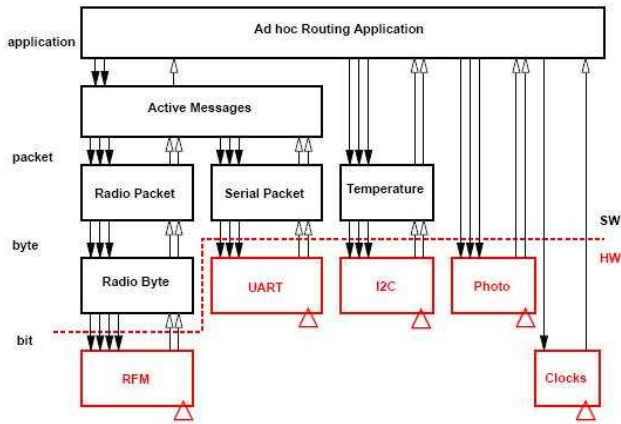


Fig. 2. TinyOS component hierarchy

A recent work in [28] uses the concept of drawing a *red line*, which refers to having a boundary between the trusted and un-trusted code. Their solution, called Un-trusted Extension for TinyOS (UTOS), uses a concept similar to sandboxing. It provides an environment in which un-trusted, and possibly malicious, code could be run without affecting the kernel. UTOS creates the sandbox by using *extensions* which are the interface between the un-trusted code and the TinyOS components. The architecture of UTOS is shown in Figure 3.

In addition to the above problem, i.e. the non-existence of kernel and user separation, TinyOS uses the concept of Active Messaging (AM). AM is an environment that facilitates message-based communication in distributed computer systems. Each AM message consists of the name of a user-level handler on the target node that needs to be invoked as well as the data that needs to be passed on [11]. This approach enables the implementation of a TCP/IP like network stack on the nodes that fits the hardware limitations of the sensor nodes.

Another vulnerability with the current implementation of TinyOS is that it is possible to open a port to a remote sensor node using the USB port and a PC. The *serial forwarder*, which is one of the most fundamental components of TinyOS software, is called to open a port to a node. There is no security check to authenticate the user who is attempting to open the port. This could lead to an attack on the software where the adversary opens a port to the nodes and uploads software, or downloads information from the nodes.

Some of the solutions that should be considered to secure the TinyOS software and protect the software from being exploited by malicious users are:

- Defining rigorous trust boundaries for different components and users
- Software authentication and validation. For example, a new line of research in sensor network has started looking into the problem of *remote software-based attestation* [29].

- Using restricted environment such as Java Virtual Machine. The JVM is already available in TinyOS software [18] and can be used to restrict the access of unauthorized users to the kernel.
- Hardware attestation. For example, Trusted Computing Group and Next Generation Secure Computing Base provide this type of attestation [25]. A similar model could be used in sensor networks.
- Dynamic run-time encryption decryption for software: this is similar to the encryption/decryption of the data except that the code running on the device is encrypted. This will prevent a malicious user from exploiting the software.

IV. ATTACKS ON THE NETWORK COMMUNICATION STACK

As explained in [32], the attacks on the communication layer of the network can be divided up into the following categories:

- Physical layer
- Link layer
- Network and Routing layer
- Transport layer

In the following subsections, we explain the attacks on each layer in more detail.

A. Physical Layer

Sensor nodes use Radio Frequency (RF) to communicate wirelessly among each other. One of the important attacks on the wireless communication is *jamming*. Jamming is the interference with the RF used by the nodes in a network. The adversary can use a small number of nodes scattered around in the network to disrupt the communication in the entire network.

Common defenses against the jamming attack is using some form of the spread spectrum communication. Examples of this are frequency hopping and code spreading. Another solution to the jamming attack has been proposed in [33]. The authors suggest a mechanism by which a jammed region can be mapped by the surrounding nodes. The goal of the protocol is to cope with the jamming attack, and isolate the jammed region from the rest of the network.

B. Link Layer

Link layer protocol provides means for neighboring nodes to access the shared wireless channel, for example Carrier Sense Multiple Access (CSMA). Examples of attack on the link layer protocol are:

- Causing collision with packets in transmission
- Exhaustion of the node's battery due to repeated retransmission
- Unfairness in using the wireless channel among neighboring nodes

A number of solutions have been suggested for detecting these attacks, such as using collision detection techniques,

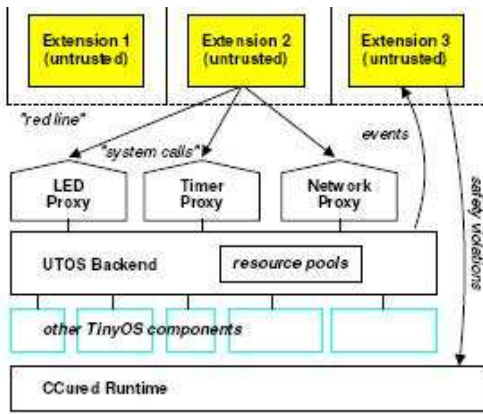


Fig. 3. UTOS architecture [28]

modifying the MAC code so as to limit the rate of requests, and using smaller frames for each packet [32].

C. Network and Routing Layer

The goal of this layer is to provide reliable end-to-end transmission. As mentioned earlier, all the nodes in the sensor network act as routers. This introduces a new complexity dimension to the design of routing protocols for sensor networks. The routing protocols have to be energy and memory efficient but at the same time they have to be robust to security attacks and node failures. There have been many power-efficient routing protocols proposed for sensor networks. However, most of them suffer from different security vulnerabilities as discussed by authors in [17]. Here we briefly mention a few of the attacks on the routing protocols. The complete list can be found in [17]:

- Black holes: This attack is launched against distance vector routing protocols. A compromised node advertises a zero or a very low cost to its neighbors. As a result, a large number of packets get routed toward this node.
- Wormhole attack: in this attack the adversary node tunnels the messages to another part of the network through a low latency link, and then replays them. This attack is particularly challenging to deal with since the adversary does not need to compromise any nodes and can use laptops or other wireless devices to send the packets on a low latency channel. In [34] the authors propose using *packet leashes*. Packet leashes are additional information added to the packet whose purpose is to restrict the maximum distance the packet can travel in a given amount of time. Another solution has been proposed in [27] where a graph theoretic framework for modeling wormhole links is given. The authors derive necessary and sufficient conditions, based on the graph theoretic framework, for detecting and defending against wormhole

attacks. However, the technical details of the solution are beyond this paper.

- Spoofed, altered, replayed packets: This attack targets the routing information used by nodes. As a result, it could lead to creating routing loops, or increase the end to end delay.
- Selective forwarding: in this attack the compromised node only forwards a fraction of the packets it receives and drops the rest. Denial-of-Message attack on broadcast in sensor networks is an example of the selective forwarding.
- Sinkhole attack: in this attack the adversary tries to attract most of the traffic toward the compromised nodes.
- Acknowledgement spoofing: The goal of the adversary in this attack is to spoof a bad link or a dead node using the link layer acknowledgement for the packets it overhears for those nodes.

D. Transport Layer

The transport layer is used for managing the end-to-end connections for different applications in the network. Transport layer protocols are usually simplified to fit the requirements of sensor networks, such as energy-efficiency. STCP [14] is an example of a generic transport layer protocol specifically designed for sensor networks.

Flooding and desynchronization are two types of attack targeted at the transport layer protocols. The goal of the flooding attack is to exhaust the memory of a node through sending many connection establishment requests. In the desynchronization attack the adversary forges packets to one or both ends of a connection using different sequence number on the packets. This will cause the end points of the connection to request retransmission of the 'perceived' missed packets. Authentication and using client puzzles are two possible solutions to guard against these attacks [32]. The question that needs to be answered is whether these solutions can be implemented in sensor networks, and what modifications need to be made to make these schemes plausible in the realm of sensor networks.

V. TRAFFIC ANALYSIS ATTACKS

Given the nature of the sensor network, the traffic through the network has a specific pattern, i.e. it is a many-to-one pattern or many-to-a-few. Most of the nodes in the network send their observations back to the base station, as shown in Figure 4. This gives an adversary an extra dimension of vulnerability to exploit.

An adversary is able to gather a lot of information on the topology of the networks as well as the location of the base station and other strategic nodes by observing the traffic volume and pattern. For example, the adversary might observe the traffic and deduce the nodes that are on the vertex cut-set. Then he can attack and compromise those nodes which will result in breaking the network into two disconnected

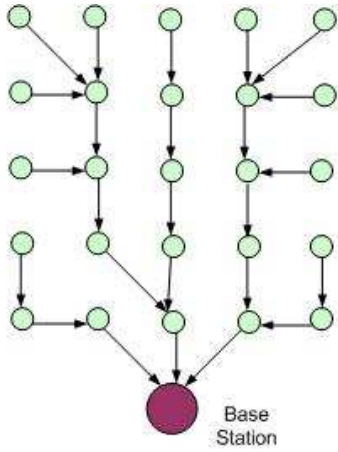


Fig. 4. An example of traffic pattern in sensor networks.

components, and any path from one component to the other has to go through the compromised nodes. Alternatively, the attacker might launch a Denial of Service attack against the nodes on the vertex cut-set in order to drain their energy. As a result, the overall lifetime of the network is decreased ². There are two ways in which traffic analysis attacks can be performed by an adversary:

- In the first attack, the adversary observes the packet-sending rate of the nodes that are close to it, and then moves towards nodes that have a higher packet sending rate.
- In the second attack, the adversary observes the time between consecutive packet-sendings among neighboring nodes. Then he tries to follow the path of the packet that is being forwarded until it reaches the base station.

The possible solutions to the traffic analysis attack are to use randomness and multiple paths in routing, using probabilistic routing ³, and the introduction of fake messages in the network. Probabilistic geographic routing scheme (PGR) has been explored by the authors in [30], where the next hop is chosen based on the link quality and residual energy of a subset of the neighbors of a node. It has been shown through simulations that PGR is energy efficient and performs well in terms of throughput.

A cautionary note on using fake messages is in order. Fake messages will introduce additional overhead in terms of energy-consumption and in-network traffic. In order for the fake messages to be effective in preventing the adversary from learning any information, they have to look like real messages. Therefore, we can not use any optimization on these fake messages.

²Lifetime of the networks is generally defined as the time it takes for the network to become partitioned.

³In deterministic routing, the next hop is selected based on a known rule, such as shortest path. In contrast, in probabilistic routing, the next hop is chosen at random from among a number of candidate nodes. Therefore, the next hop can not be determined ahead of time.

VI. KEY MANAGEMENT PROTOCOLS

Nodes in a sensor network use either pre-distributed keys or use some form of keying material to generate the keys dynamically. The cryptographic keys are either group-wise or pair-wise. In addition, each node has to discover its neighbors with which it shares a secret key. If two nodes do not share a secret key directly, then they have to find a path that connect the two of them in a secure fashion. The goal of the key management protocol is to pre-distribute cryptographic keys among the nodes prior to the deployment, revoke keys if nodes leave the network, and assign new keys to the nodes joining the network or when some of the keys expire. In sensor networks the key management protocols fall into one of the following categories:

- Deterministic: In this case the processes that generate the key pools and the key chains are deterministic.
- Probabilistic: The key chains are selected randomly from a given key pool and distributed among the nodes.
- Hybrid: This approach uses a combination of the probabilistic and deterministic solutions to increase resilience and scalability.

Examples of the key management protocols can be found in [5], [7], [19], [35]. The problem with some of the probabilistic approaches such as [7] is that the scheme is not resilient to physical capture attack. Since nodes share pair-wise keys, capturing a small number of nodes is enough to break the protocol.

Most of the key management protocols, however, are not resilient to an attacker who observes the nodes during the 'discovery' process of the shared keys. The attacker can use the information he has obtained to attack some of the nodes, and break the key management protocol. This has been shown through simulation in [26]. A possible solution is to use public key cryptography. However, it is generally believed that implementing a public key protocol on the sensor nodes with their limited power and memory resources is not feasible. It remains to be seen if the security gain from having a public key cryptography algorithm outweighs the energy consumed in the extra overhead.

VII. SYBIL ATTACK

Sybil attack refers to the scenario when a malicious node pretends to have multiple identities. For example, the malicious node can claim false identities (*fabricated identities*), or impersonate other legitimate nodes in the network (*stolen identities*).

As the authors point out in [25], the Sybil attack can affect a number of different protocols:

- Distributed Storage
- Routing Protocols
- Data Aggregation (used in query protocols)
- Voting (used in many trust schemes)
- Fair Resource Allocation
- Misbehavior Detection

The proposed solutions to Sybil attack include: 1) radio resource testing which relies on the assumption that each physical device has only one radio, 2) random key pre-distribution which associates the identity of the node to the keys assigned to it and validate the keys to see if the node is really who it claims to be, 3) registration of the node identities at a central base station, 4) position verification which makes the assumption that the sensor network topology is static.

Each of these solutions has its own drawbacks. For example, there is no guarantee that every physical device is going to have only one radio. In fact some of the MAC protocols rely on each node having more than one radio. The key pre-distribution is challenging as mentioned in the previous section. The problem with the last proposition is that there is no guarantee that the network topology is static, and the nodes do not change their location. Many sensor network deployment require mobile nodes. Therefore, this solution is likely to fail in the case of dynamic topologies.

VIII. ATTACKS ON REPUTATION-ASSIGNMENT SCHEMES

Reputation or recommendation systems have proven useful as a self-policing mechanism to address the threat of compromised entities. They operate by identifying selfish nodes, and isolating them from the network. They help the users, i.e., the nodes in the network, to decide which nodes they can trust and communicate with. Centralized reputations systems were popularized by the internet. An example of this system is Ebay's rating system.

Decentralized methods were then created for use in ad hoc networks [15]. The *CORE* reputation system [21] and the *CONFIDANT* protocol [1] use a watchdog module at each node to monitor the forwarding rate of the neighbors of the node. If the node does not forward the message, its reputation is decreases, and this information is propagated throughout the network. Each node also uses the second hand information from other nodes to find the overall reputation of a node. Over time the bad behaving nodes are less trusted and will not be used in forming reliable paths for routing purposes. These two protocols differ in how they use the second hand information, how to punish bad behavior and how to instill trust for the node which misbehave temporarily. A formal model for trust in dynamic networks is introduced by Carbone et. al [2]. The most relevant work for sensor networks is presented in [8]. The authors suggest a high level reputation system frame work for sensor networks. However, they only suggest a *watchdog* mechanism to determine the reputation of each node. The purpose of the watchdog mechanism is to monitor the neighbors of a node, and determine if any of the nodes deviate from their expected behavior. The authors in [8] state that there is no unified way to design the watchdog mechanism, i.e. the mechanism to assign reputation to each node has to be context dependent and varies based on the application at hand.

Distributed reputation systems face a great challenge since the

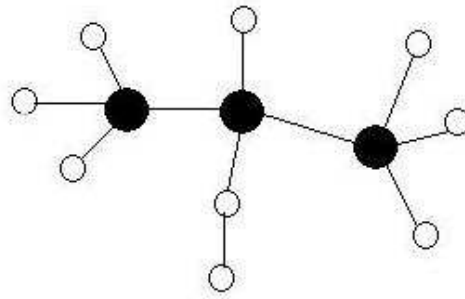


Fig. 5. The white nodes are sensor nodes, and the black circles are the aggregate nodes.

main assumption behind reputation systems in general is that the information available on an entity of the network is correct. However, there are a number of problems that arise in the reputation-based systems. The following is a short list of the security issues [22]:

- Ballot stuffing
- Bad-mouthing
- Reputation transitivity: if a node can not directly observe another node and assign a reputation, should it trust its neighbors and take their reputation for the unobserved node into account, and how should it combine this second-hand information with its own first-hand information.
- The Sybil attack
- Measuring the performance of the network: how should one assign a reputation value to a node so as to get the desired outcome.

In addition to these difficulties, there are other problems that can arise within the reputation system framework. For example, as mentioned earlier, in [1] the authors propose using a watchdog mechanism that determines if nodes are forwarding their packets properly. Since the medium of transmission is wireless, all the nodes in the neighborhood of a transmitting node can hear the communication. This *overhearing* property is used by the watchdog mechanism to determine misbehavior. However, in a number of other applications in sensor network, we can not utilize the overhearing property. For example, if the network is used to monitor an event, such as a moving object, then the watchdog mechanism has to be designed in a way to take into account the physical correlation among neighboring nodes' signal strengths (observations) to detect misbehavior. Currently, there are no available analytical models for signal strength correlation based on distance in sensor networks.

Given the security problems facing reputation systems along with the lack of analytical models for designing watchdog mechanisms, the design of a reliable reputation system for sensor networks has proven to be a very challenging problem. Future research needs to focus on designing reputation-assignment mechanisms that are resilient to most of the above security attacks. At the same time these mechanisms have to

be simple enough to be implemented on the limited memory of a sensor node.

IX. ATTACKS ON IN-NETWORK PROCESSING

In-network processing, also called data aggregation, is a key feature of sensor networks. Given the limited resources of the sensor nodes, it is nearly impossible for all the nodes to send back their data to the base station. In addition, the rate of packet collision will significantly increase if all the nodes report their observed data. Therefore, some of the intermediate nodes fuse the data from their neighborhood, and send the aggregated data back to the base station. It is possible to have multiple levels of hierarchy among nodes, and in each level one node fuses the data from the nodes at the level directly below it. An example of the aggregation process is shown in Figure 5.

As one can immediately see, if a few nodes are compromised, they can inject faulty data into the network. This will result in a corrupted aggregate. For example, assume that the nodes are monitoring the temperature of the environment, and the fusion process is simply to take the average of all the sensors' readings. Now if one node is compromised and gives a high or a low reading, it will affect the average and pull in one direction. Other, more fundamental, applications that use data aggregation are multi-object tracking [23] and *directed diffusion* routing [13]. Here we will give an example of how a hierarchical multi-object tracking can be affected by attacks on the aggregation. In hierarchical tracking, there are hierarchies of nodes. At each level, a leader is selected and the nodes in its neighborhood send their observations to the leader. The leader uses an averaging scheme to fuse the observations. The leader nodes, then, send their fused observations to the base station. Base station forms the object's track bases on the received aggregated data. However, if a fraction of the nodes are compromised and send faulty observations to the leader nodes, the fused observation will be skewed. As a result, the track formed by the base station is either wrong, or in some cases non-existent in reality. Figure 6 shows a scenario were a fraction of the nodes are compromised. The compromised nodes send a constant faulty observation. As a result the tracks formed by the base station, shown in blue, are severely different from the real track of the object, shown in red.

There have been a few solutions suggested in the literature to secure the in-network processing [3], [4], [31]. Wagner [31] suggests using statistical properties, such as median, to reduce the effect of attacks on the aggregation process. Authors in [4] propose a solution based on forming secure hierarchies of node clusters. Cryptographic keys are used at each level of the hierarchy to establish secure communication among the nodes in a cluster. The solution proposed in [3] also relies on cryptographic key establishment to ensure security of the fused data.

One possibility to secure in-network data aggregation is to use a reputation system. A node's data is only considered

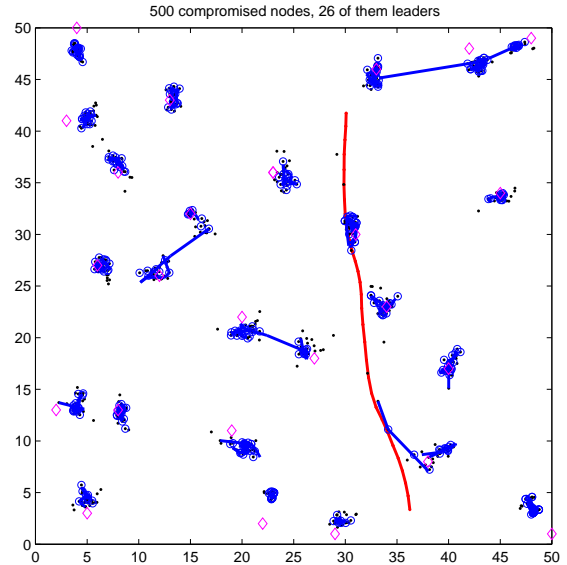


Fig. 6. The blue lines are the tracks formed at the base station. The red line is the actual track of the moving object.

if the reputation is high enough and discarded otherwise. However, using this solution requires having a robust and attack-resistant reputation system. A second possibility is to use robust statistical methods for estimation that are not as prone to errors as averaging.

X. ATTACKS ON TIME SYNCHRONIZATION PROTOCOLS

Time synchronization protocols provide a mechanism for synchronizing the local clocks of the nodes in a sensor network. There are several time synchronization protocols for the internet, such as Network Time Protocol (NTP). However, given the non-determinism in transmissions in sensor networks, NTP cannot be directly used in wireless sensor networks. Time synchronization implementations have been developed specifically for sensor networks. Three of the most prominent are Reference Broadcast Synchronization (RBS) [6] Timing-sync Protocol for Sensor Networks (TPSN) [9], and Flooding Time Synchronization Protocol (FTSP) [20]. However, none of these protocols were designed with security as one their goals. An adversary can easily attack any of these time synchronization protocols by physically capturing a fraction of the nodes and have them inject faulty time synchronization message updates. In effect, the nodes in the entire network will be out-of-sync with each other.

Time-synchronization attacks have great effects on a set of sensor network applications and services since they heavily rely on accurate time synchronization to perform their respective functions. To illustrate the effects of corrupted time synchronization, we describe the effect on estimating the state based on sensor readings from a sensor network. The state estimation is the foundation of any tracking algorithm. We give a simple example using the Kalman filter, which is used extensively in tracking. The Kalman filter estimates the

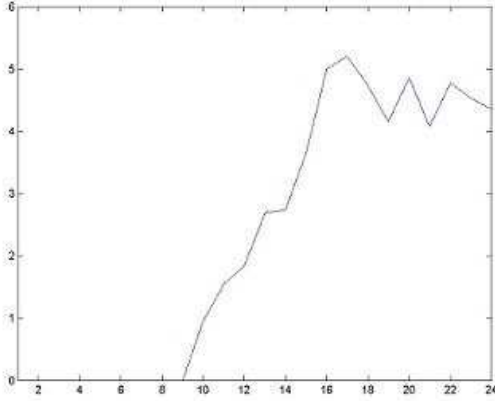


Fig. 7. The y axis shows the norm of the difference between the results from the Kalman filter before and after de-synchronization. The x axis is the time of the corresponding observation.

state of a discrete-time controlled process that governed by a linear stochastic difference equation.

$$x_k = Ax_{k-1} + Bu_{k-1} + w_{k-1} \quad x \in \mathbb{R}^n \quad (1)$$

given the measurement $z_k \in \mathbb{R}^m$, where

$$z_k = Hx_k + v_k \quad (2)$$

The random variables w and v represent process and measurement noise and are assumed to be independent random variables with normal distribution,

$$p(w) \sim N(0, Q) \quad p(v) \sim N(0, R)$$

The Kalman filter estimates the state at every time step. We simulated the movement of an object using equation, where the state is position and velocity of the object in two dimensions. We then used the Kalman filter to estimate the position and velocity of the object before and after modifying the time of some of the position observations, as might occur in an attack on the time synchronization in the sensor network. We simulated on moving object in our experiment. The norm of the error is shown in Figure 7, and we began the de-synchronization at time 10. The y axis shows the norm of the difference between the results from the Kalman filter before and after de-synchronization. The x axis is the time of the corresponding observation.

As this example shows, an accurate time synchronization protocol is essential to guaranteeing correct performance of different algorithms in sensor network. Future research needs to assess the effect of security attacks on different applications in sensor networks by running experiments on real testbeds. In addition, designing secure time synchronization protocols for sensor networks is of great importance.

XI. CONCLUSION

Sensor networks are a promising technology with many important applications, such as environment monitoring, health care, surveillance. They are the way of the future, and it is envisioned that the sensor networks will be used in critical infrastructure.

The nodes that comprise these networks have very limited resources in terms of memory and power. The reason for these constraints is that the driving force behind sensor network's success is the small dimension, which facilitates non-intrusive deployment, and the cheap price of the hardware.

The protocols designed for sensor networks have to be simple and efficient both memory-wise and energy-wise to accommodate the resource constraints of the sensor nodes. However, given the unattended nature of sensor networks, they are vulnerable to a number of security attacks which could substantially degrade the performance of the network.

The goal of this paper is to give a comprehensive taxonomy of the security attacks on sensor networks and their effect on the performance of the network. We gave some of the possible solutions that should be considered. In addition, we gave future directions for extending research in the area of sensor network security.

REFERENCES

- [1] S. Buchegger and J. L. Boudec. Performance analysis of the confidant protocol: Cooperation of nodes - fairness in dynamic ad-hoc networks. *IEEE/ACM Symposium on Mobile Ad Hoc Networking and Computing*, June 2002.
- [2] M. Carbone, M. Nielsen, and V. Sassone. A formal model for trust in dynamic networks. *IEEE International Conference on Software Engineering and Formal Methods*, 2003.
- [3] J. Deng, R. Han, and S. Mishra. Security support for in-network processing in wireless sensor networks. *First ACM Workshop on the Security of Ad Hoc and Sensor Networks (SASN)*, 2003.
- [4] T. Dimitriou and D. Foteinakis. Secure and efficient in-network processing for sensor networks. *Workshop on Broadband Advanced Sensor Networks(BroadNets)*, October 2004.
- [5] W. Du, J. Deng, Y. Han, and P. Varshney. A pairwise key pre-distribution scheme for wireless sensor networks. *In 10th ACM Conference on Computer and Communications Security (CCS03)*, October 2003.
- [6] J. Elson and D. Estrin. Fine-grained network time synchronization using reference broadcast. *The fifth symposium on Operating Systems Design and Implementation (OSDI)*, December 2002.
- [7] L. Eschenauer and V. Gligor. A key-management scheme for distributed sensor networks. *In Conference on Computer and Communications Security*, November 2002.
- [8] S. Ganeriwal and M. B. Srivastava. Reputation-based framework for high integrity sensor networks. *ACM Security for Ad-hoc and Sensor Networks*, 2004.

- [9] S. Ganeriwawal, R. Kumar, and M. Srivastava. Timing-sync protocol for sensor networks. *The first ACM Conference on Embedded Networked Sensor Systems (SenSys)*, November 2003.
- [10] D. Gay, P. Levis, and D. Culler. Software design patterns for tinyos. *Proceedings of the ACM SIGPLAN/SIGBED Conference on Languages, Compilers, and Tools for Embedded Systems (LCTES'05)*, June 2005.
- [11] J. Hill, P. Bounadonna, and D. Culler. Active message communication for tiny network sensors. *UC Berkeley Technical Report, Berkeley*, January 2001.
- [12] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister. System architecture directions for network sensors. *ASPLOS 2000*, November 2000.
- [13] C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed diffusion: A scalable and robust communication paradigm for sensor networks. *In Proceedings of the Sixth Annual International Conference on Mobile Computing and Networking (MobiCOM '00)*, August 2000.
- [14] Y. Iyer, S. Gandham, and S. Venkatesan. A generic transport layer protocol for sensor networks. *Proceedings of 14th IEEE International Conference on Computer Communications and Networks*, October 2005.
- [15] a. J. L. B. J. Mundinger. Analysis of a robust reputation system for self-organized networks. *University of Cambridge, Statistical Laboratory Research Report*, January 2004.
- [16] C. Karlof, N. Sastry, and D. Wagner. Tinysec: A link layer security architecture for wireless sensor networks. *Proceedings of the Second ACM Conference on Embedded Networked Sensor Systems*, November 2004.
- [17] C. Karlof and D. Wagner. Secure routing in sensor networks: Attacks and countermeasures. *Ad Hoc Networks, vol 1, issues 2–3 (Special Issue on Sensor Network Applications and Protocols)*, pp. 293-315, Elsevier, September 2003.
- [18] P. Levis and D. Culler. Mate : a virtual machine for tiny networked sensors. *ASPLOS*, October 2002.
- [19] D. Liu and P. Ning. Establishing pairwise keys in distributed sensor networks. *In Computer Communication Society*, October 2003.
- [20] M. Maroti, B. Kusy, G. Simon, and A. Ledeczi. The flooding synchronization protocol. *Proc. Of the Second ACM Conference on Embedded Networked Sensor Systems*, November 2004.
- [21] P. Michiardi and R. Molva. Core: A collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks. *Conference on Communications and Multimedia Security*, September 2002.
- [22] S. Moloney and P. Ginzboorg. Security for interactions in pervasive networks: Applicability of recommendation systems. *1st European Workshop on Security in Ad-Hoc and Sensor Networks*, 2004.
- [23] S. Oh, S. Russell, and S. Sastry. Markov chain monte carlo data association for general multiple-target tracking problems. *IEEE International Conference on Decision and Control*, December 2004.
- [24] K. Okeya and T. Iwata. Side channel attacks on message authentication codes. *2nd European Workshop on Security and Privacy in Ad hoc and Sensor Networks*, July 2005.
- [25] A. Perrig, J. Newsome, E. Shi, and D. Song. The sybil attack in sensor networks: Analysis and defenses. *Third International Symposium on Information Processing in Sensor Networks*, 2004.
- [26] R. Pietro, L. Mancin, and A. Mci. Efficient and resilient key discovery based on pseudo random key pre-deployment. *International Parallel and Distributed Processing Symposium*, April 2004.
- [27] R. Poovendran and L. Lazos. A graph theoretic framework for preventing the wormhole attack in wireless ad hoc networks. *in ACM Journal on Wireless Networks*, 2005.
- [28] J. Regehr, N. Coopriider, W. Archer, and E. Eide. Memory safety and untrusted extensions for tinyos. *In submission*, April 2006.
- [29] M. Shaneck, K. Mahadevan, V. Kher, and Y. Kim. Remote software-based attestation for wireless sensors. *In Proceedings of the 2nd European Workshop on Security and Privacy in Ad Hoc and Sensor Networks*, July 2005.
- [30] S. S. Tanya Roosta. Probabilistic geographic routing in ad hoc and sensor networks. *In proc. of International Workshop on Wireless Ad-hoc Networks (IWWAN)*, May 2005.
- [31] D. Wagner. Resilient aggregation in sensor networks. *ACM Workshop on Security of Ad Hoc and Sensor Networks*, October 2004.
- [32] A. Wood and J. Stankovic. Denial of service in sensor networks. *IEEE Computer pages 5462*, Oct. 2002.
- [33] A. Wood, J. Stankovic, and S. H. Son. Jam: A jammed-area mapping service for sensor networks. *In Real-Time Systems Symposium*, 2003.
- [34] H. Yih-Chun and D. Johnson. Wormhole attacks in wireless networks. *IEEE Journal on Selected Areas in Communications (JSAC)*.
- [35] S. Zhu, S. Setia, and S. Jajodia. Leap: Efficient security mechanisms for large-scale distributed sensor networks. *In 10th ACM Conference on Computer and Communications Security*, October 2003.