Book Title: XXXXXXXXXXXXXXXXXXXXXXXXXXXX

Editors

December 29, 2007

Contents

Chapter 1

Message Authentication in Surveillance Networks

Raymond Sbrusch, T. Andrew Yang

1.1 Abstract

Wireless sensor networks simplify the collection and analysis of data from multiple locations. Target tracking and perimeter intrusion detection applications benefit from the ad-hoc deployment and self-organization capabilities of wireless sensor networks. However, sensor networks deployed in hostile environments must be fortified against attacks by adversaries. This chapter will examine the constraints that make wireless sensor network surveillance challenging and evaluate algorithms that provide reasonable security to maintain origin integrity and data integrity for wireless sensor networks deployed for surveillance and target tracking.

Target tracking techniques may focus on intruders penetrating a border in a two- dimensional space or may try to provide comprehensive coverage of a detection region. These

techniques commonly impose a trade-off between accuracy of the tracking algorithm and efficient use of sensor energy, computation power, and communication resources. The naive surveillance strategy delivers high accuracy by enabling the sensing mechanism on every node in the network and forcing each node to send alerts directly to the base station. This strategy reduces network longevity and increases contention for the radio channel. Advanced strategies organize nodes into clusters with child nodes detecting an intrusion and intermediary nodes relaying messages between children and the base station. This chapter will first explain how methods, including Scalable Tracking Using Networked Sensors (STUN) [1], Low Energy Adaptive Clustering Hierarchy (LEACH) [2], and Optimized Communication and Organization (OCO) [3], balance network longevity and detection accuracy.

Secure wireless sensor network deployments remain elusive because of resource constraints; however, operation of sensor networks in hostile locations requires secure, authenticated communication. Conventional security mechanisms in use on the Internet are usually not applicable to wireless sensor networks because of the limited resources available in the sensor nodes, such as limited processor speed, smaller memory size, and limited communication channels and speed. The second goal of this chapter is to identify threats against target tracking wireless sensor networks and to evaluate characteristics of authentication protocols that can be employed to mitigate those risks. The survey will track the evolution of sensor network authentication from widely-adopted standards including SPINS [4], and TinySec [5], to more complex strategies.

1.2 Target Tracking with Wireless Sensor Networks

Detection and tracking of objects moving through a surveilled region remains in the forefront of wireless sensor network research. In wireless sensor networks deployed for tracking objects, the effectiveness of organization and communication methods depend on many factors, including the node deployment tactic, the layout of the surveyed landscape, and the path of the intruder. Careful selection of appropriate communication protocols can assure longer life for the network. This communication strategy must identify how nodes send alerts, which nodes need to receive alerts, the role of nodes in monitoring, and how the network handles changes such as destruction or movement of a node. This section identifies the unique challenges of target tracking and surveys organization, communication, and tracking algorithms.

1.2.1 Challenges of Target Tracking

Target tracking wireless sensor networks face a unique set of challenges when compared to sensor networks deployed for other applications such as building automation or habitat monitoring. Threats against the network increase with the hostility of the surveilled area and range from simple node destruction to sophisticated network-based exploits. The sensor network must be able to monitor intrusions and securely deliver alerts even if a large percentage of the nodes have been compromised.

Resource Constraints

Wireless sensor nodes may range in size from a few millimeters to the size of a conventional computer. Regardless of size, sensor nodes share common constraints. Wireless sensor nodes are characteristically constrained in terms of computational capabilities, communication bandwidth, energy, and storage, when compared to conventional Internet-connected computing devices. Resource constraints must be calculated into any wireless sensor network design.

Sensor network packets are commonly 32 bytes or less [6]. Conventional security mechanisms that add 16 bytes of overhead are inappropriate for use in wireless sensor networks since they will quickly drain the sensor power source. On some platforms, each bit transmitted consumes about as much power as executing 800-1000 instructions.

Besides the requirement of conserving energy, a wireless sensor network deployed for target tracking must provide a robust communication channel, real-time alerting, resistance to tampering and stealthiness. This paradox forces network designers to exploit ways to improve efficiency while maintaining a high level of accuracy by, for example, incorporating redundancy, clustering, etc. into the network.

Physical Threats

Deployment of a sensor network in a safe environment can be carefully planned and controlled. Safer environments benefit from more deliberate node distribution. For example, in the Twenty-nine Palms experiment [7], researchers at U.C. Berkeley deployed wireless sensor nodes to track the path of tanks along a predetermined path. A small number of sensors were carefully dropped from an unmanned aerial vehicle at points near the tank path. The sensors detected the presence of the tanks using a magnetometer and sent alerts over a 916.5 MHz radio signal to the base station. This application detected the arrival of the tank and calculated its speed and direction as it moved along the path.

In contrast, the challenges commence as soon as sensor nodes are deployed in a hostile environment, where sensor nodes are subject to threats. Furthermore, sensor nodes may be dispersed from an aerial vehicle that itself could become the target of attack. Aerial dispersion results in a highly random coverage pattern with coverage gaps in some areas and excessive redundancy in others. Once sensors are deployed in an adversary's domain, there are few chances to modify the coverage patterns or refresh dying nodes. In a sensor network, nodes typically remain immobile throughout their lifetime. However, external forces such as explosions or earthquakes can unpredictably relocate nodes.

1.2.2 Target Tracking Methods

Target tracking strategies can be evaluated on how they organize to form a network, how they sense intrusions, how they communicate results to the base station, and how they reorganize to handle exhausted or damaged nodes.

Single-Hop Communication

The simplest organization and communication strategy, known as Direct Communication (DC) [2], requires each node to transmit alerts directly to the base station. This single-hop strategy suffers from rapid depletion of resources. In DC, each node monitors the environment with its sensing module and transmits alerts directly to the base station over a wireless channel. While this delivers the most accurate detection of intrusions from any attack vector, DC faces a number of setbacks.

Each sensor node must have a radio transmitter powerful enough to directly reach the base station. This limits the effective range of the network and forces the base station to allocate enough radio channels for communication with all sensors. Nodes maintain an activated sensor module throughout their lifetime. This module continuously draws power from the node battery. Since nodes may have overlapping sensor coverage areas, redundant alerts may be sent to the base station, again unnecessarily depleting the battery. While theoretically effective at tracking targets, DC is inefficient and impractical for use in real-world applications.

Hierarchy Trees

Organizing nodes into hierarchical network topologies or clusters can increase the coverage area and take advantage of redundancy to improve efficiency. In a tree-structured network organization, senor nodes occupy graph vertices and graph edges signify direct communication links between nodes. Scalable Tracking Using Networked Sensors (STUN) [1], based on a hierarchy tree, aims to track a large number of objects as they move through the surveilled region. Leaf nodes at the bottom of the STUN tree function as sensing nodes. STUN organizes sensing nodes into a linear graph; however, adjacency of nodes in the graph does not necessarily imply physical nearness. Nodes at intermediate levels relay messages from sensing nodes to the base station at the root of the tree. Intermediary nodes store information about the presence of detected objects. When leaf nodes send detection messages toward the base station, the intermediate nodes compare the alerts to the information they already recorded and drop the messages if they are redundant. Pruning redundant alerts lowers communication costs. While hierarchy trees are an improvement over Direct Communication, sensor networks gain more efficiency and accuracy by accounting for the physical proximity of the sensor nodes after they are dispersed.

Clustering Methods

Clustering methods capitalize on ability to detect node proximity when forming the hierarchy tree. Like a hierarchy tree, clustering algorithms prune redundant messages as they are sent to the base station. They also conserve energy by assuring low cost radio communication between sensor nodes and intermediate nodes. The Low Energy Adaptive Clustering Hierarchy (LEACH) [2] illustrates how knowledge of actual node proximity can improve efficiency of hierarchy tree based organizations. During the LEACH setup phase, nodes elect themselves to be local cluster heads and broadcast messages to their neighbors advertising their status. The cluster heads act as intermediate nodes and relay messages between leaf nodes in a neighborhood and the base station. To qualify for the role of cluster head, a node must have enough power to relay messages to the base station. Sensor nodes analyze cluster head advertisements to determine the cost of wireless communication with the cluster head. They choose which cluster they want to join by selecting the one which requires the least radio transmission power. The role of cluster head consumes more energy than the role of sensor node, thus a LEACH network periodically repeats the setup process, electing new nodes to the cluster head position.

LEACH distributes the cost of serving as cluster head among all nodes in the network, thus increasing the lifetime of the network. Like STUN, LEACH still lacks knowledge of the surveilled terrain. Since any node could theoretically elect itself as the cluster head, LEACH requires that all nodes initially have enough radio power to communicate directly with the base, even if they are on the perimeter of the surveilled region. Another potential consequence of the random election process is that some areas of the target zone may not be covered by a cluster head. The election process only evaluates the strength of the node's radio, not the coverage patterns of neighboring nodes.

Image Processing Techniques

Image processing techniques have been shown to be a more efficient and accurate target tracking method than conventional graph-based methods [3, 8]. Image processing techniques map physical node location on a grid representing the coverage region. The efficiency is achieved in part by activating the tracking components of perimeter nodes and placing interior nodes in an energy preserving sleep state until the perimeter is breached. The Optimized Communication and Organization (OCO) method [3, 8], for example, efficiently secures the perimeter of the detection region and reorganizes the network when a node is damaged or lost. OCO efficiently maintains a secure perimeter by segmenting a network lifetime into four phases.

When nodes are initially deployed, the sensor network enters a "Position Collection" phase. The base station broadcasts a message to all nodes and requests that they report their ID and position. The base's neighbors acknowledge the request, flag themselves as recognized, and broadcast a request for the ID and position of all their neighbors. The process repeats until all nodes are accounted for. The base station maps each node's unique ID and location onto a map of the surveilled region.

The next phase, "Processing", identifies the minimal set of sensor nodes required to completely cover the detection region. It achieves this by finding redundant nodes whose sensing radius overlaps with the sensing radius of other nodes in the network. The identification of non-redundant nodes is accomplished in three steps. The base station initializes an empty list and adds itself to the list. It then analyzes the map to identify sensor nodes with non-overlapping coverage. Any node that does not overlap with another is added to the list. The base station then identifies areas of the graph that are not covered by a sensor node and assigns a nearby node responsibility for that region. All nodes not on the list after this step are considered redundant.

Identification of perimeter nodes (aka border nodes) and construction of the hierarchy tree also occur in the processing phase. The base station analyzes each pixel in the coverage map to determine if it belongs to a node. Pixels in the map that are covered by a node are assigned a value of 1. Regions of the graph that are not covered by a sensor node are assigned a value of 0. When the base station locates a pixel p with value 1, it checks the value of neighboring pixels. If any neighbor has a value of 0, then the node encompassing pixel p is considered a border node.

Once border nodes are defined, the base station must find the shortest path from itself to each of the active nodes, including border and intermediate nodes. Border nodes can route alerts to the base station by relaying them through multiple interior forwarding nodes. After the path is found, the base station broadcasts a message activating the sensor and the radio modules on the border nodes. The base station instructs redundant nodes to enter a sleeping state where their radio and sensor modules are temporarily inactive. Forwarding nodes deactivate their sensor module, yet keep their radio receiver on.

Once the network topology is defined, the network enters the "Target Tracking"phase. Intruders are assumed to enter the surveilled area from outside the perimeter. When a node detects an intruder, it sends a report to the base station via the forwarding nodes. It will continue to send periodic alerts to the base station while the intruder remains within its sensing radius. The neighbor notification strategy depends on the capabilities of the sensor. The sensing node will also alert its neighbors of the intrusion. Nodes capable of tracking multiple objects will only send two alerts to its neighbors; one when an intruder enters the coverage area and another when the intruder exits. A node that can only track one object at a time will periodically broadcast alerts to its neighbors while an intruder is present. When the neighbors receive an alert, they activate their sensor modules and try to determine if the intruding object has entered their sensing radius.

The "Maintenance"phase of OCO manages reconfiguration of the network when a node is destroyed, moved, or depleted of power. To detect damaged nodes, status messages are periodically exchanged between parents and children. When a node fails to send the status message on time, its peer assumes that it has been damaged. The node that was expecting to receive the status message will notify the base that its peer has vanished. When nodes move, a repositioned node will broadcast an alert stating that it has been moved. Any nodes receiving this alert will forward the message to the base. When a node detects that its power level has fallen below a predefined threshold, it sends a notification toward the base. When the base station receives notification of damage, depleted power, or movement of a node, it will trigger a local reorganization algorithm that reorganizes the affected area in order to cover the gap left by the affected node(s). The reorganization algorithm may activate one or more of the redundant nodes.

OCO was evaluated in a network simulator against DC and LEACH. The simulation was run under various scenarios, respectively with no intruders, one intruder, and multiple intruder objects. As exhibited in the simulation experiments, OCO has several strengths. When no intruders are present, an OCO network will outlast a LEACH network. An OCO network with multiple intruders will reach a constant rate of energy dissipation regardless of the number of intruders. The simulation shows that OCO is nearly as accurate as DC in detecting intruders penetrating the border. OCO surpasses other algorithms in terms of least cost per detected object.

Rotating Guard Duty

Many wireless sensor network target tracking approaches assume that an intruder will enter the detection region from the perimeter. These techniques conserve energy by allowing interior nodes to sleep until an intrusion occurs, thus increasing the longevity of the network. However, this strategy leaves the interior of the network vulnerable to attacks that bypass the perimeter, such as aerial or insider attacks. Rababaah and Shirkhodaie [9] propose a clustering and tracking technique that provides the efficiency comparable to image processing techniques while maintaining sensing throughout the detection region. This method, called Guard Duty Alarming Technique (GDAT), is inspired by military practice of rotating guard duty. While most soldiers sleep, one soldier is ordered to remain on guard duty for an assigned period. In GDAT, one sensing node is "on guard"actively sensing while its other cluster members sleep.

GDAT requires two types of nodes: head nodes and sensing nodes. Head nodes are provisioned with both strong radios to communicate with the base station and with global positioning receivers to append location information to intrusion alerts. Sensing nodes are equipped with intrusion detection devices and only require sufficient power to communicate with the head nodes.

Following network deployment, both head nodes and sensing nodes are awake. Head nodes broadcast advertisements requesting that sensing nodes adopt them as their cluster head. Sensing nodes accept the first offer they receive, record the ID of the head node, and send an acceptance message to the head node. Each head node can accept 32 sensing nodes into its cluster.

The tracking phase begins once clustering is complete. The cluster head assigns one sensing node to "guard duty"for a small interval and instructs the other sensing nodes to sleep. The sensing node can reside anywhere within the cluster. When the sensing node detects an intrusion, it alerts the head node, which appends GPS coordinates to the alert and forwards it to the base station. Following the intrusion, the sensing node will remain on duty until its shift is complete. At that point, the head node will assign another sensing node to guard duty.

Important assumptions effect the success of GDAT. Most significantly, GDAT assumes that head nodes and sensors will be normally distributed across a detection region following a probability distribution function. Factors such as rugged terrain, battle, or natural forces can adversely influence the distribution of head nodes and sensing nodes. In comparison to image processing techniques, GDAT leaves the perimeter porous. Even though it delivers higher accuracy throughout the network, it may not detect an intruder crossing the perimeter as quickly as an image processing technique. The authors indicate that they will include image processing techniques in future GDAT research.

1.3 Message Authentication in WSN for Surveillance

Security risks in wireless sensor networks include threats to the confidentiality, integrity, and availability of the system. The adoption of efficient algorithms to mitigate these risks has not kept pace with the rate of miniaturization.

1.3.1 Security Goals

Security assessments of any application focus on the five fundamental tenets of information security: confidentiality, origin integrity, data integrity, non-repudiation, and availability. The interpretation of these principles varies depending on the necessities of the application. The definitions used in this subsection are derived from [10].

Confidentiality means the concealment of information from unauthorized entities. Mechanisms used to achieve confidentiality include access control mechanisms and cryptography. In a robust wireless sensor network, the information contained in a message holds a lower priority than assuring that the message is delivered intact from an authentic sender. Thus, confidentiality alone can not sufficiently provide security. For example, child nodes commonly send "HELLO"messages to parents to inform them that they are still active. Concealing this message is less important than assuring that it originated from a legitimate node and not an impostor who had perhaps compromised or destroyed that node.

Semantic security implies a stronger guarantee of confidentiality. Semantic security requires that repeated encryption of a message M will yield unique ciphertext each round. This limits the ability of an eavesdropper to interpret the plaintext even after observing multiple encryptions of the same message. Use of initialization vectors (IVs) seeded with counters or non-repeating nonces provide semantic security.

Origin integrity, also known as authentication, refers to the trustworthiness of the source of data. It means that the receiver of a message can trust that the sender of the message is truthfully who it claims. An intruder should not be able to send a fabricated message and have it treated as a legitimate message from a trusted peer. Data integrity means that the user of the data can trust that the content of the information has not been changed in any way by an unauthorized intruder or improperly modified by an authorized user.

Non-repudiation means that the sender of a message should not be able to falsely deny later that he ever sent that message. In the pre-digital world, non-repudiation was achieved with a simple hand-written signature. In cryptography, it implies that authentication and data integrity can be certified with a high level of assurance and it can not later be refuted.

Non-repudiation is a critical security service and must be guaranteed in applications that involve financial and business transactions, where accountability of actions is critical to ensure success of the applications. Digital signatures provide non-repudiation.

Availability implies that an authorized user should be able to use the information or resource as required. In a wireless sensor network, the wireless communication link must remain available for the network to sustain operations.

Origin authentication and message authentication overshadow other security goals because of their influence on the reliability of the system and its output.

1.3.2 Challenges

The lack of efficient authenticated messaging exposes all layers of the sensor network protocol stack to potential compromise. Without link-layer authentication, an attacker may inject unauthorized packets into the network. This may be used to introduce collisions and force legitimate nodes into an infinite waiting state [11]. Network layer attacks such as those against routing protocols are of particular interest. Attacks against routing protocols give the attacker the ability to cause routing loops, delay messages, or selectively drop messages [12]. In wireless sensor networks deployed for tracking targets, the notifications about the location of the target are valuable messages. Without authentication, the attacker can perpetrate attacks such as manipulating routes and dropping intruder notifications, spoofing intruder notifications to create a diversion, or forcing the entire network into a continual state of reorganization.

While many wireless sensor network authentication schemes have been conceived, none of them is a panacea. Algorithms for unicast message authentication, for example, do not meet the requirements for authenticating broadcast messages. Similarly, algorithms that use time slots to create asymmetry may not be suitable for systems with real-time constraints. Results in [13] show that the best solution tailors the authentication mechanism to the characteristics of the network.

In wireless sensor networks, the need for integrity surpasses all other security goals. Without data integrity and authentication, there is no way to construct a highly available and trustworthy network. The following section describes common attacks against wireless sensor networks and demonstrates how integrity mitigates the threat.

Attacks Against Sensor Networks

Wood and Stankovic provide a comprehensive survey of attacks against wireless sensor networks and describe strategies that have been used to reduce the threat [11]. The discussions of attacks against sensor networks in the rest of this subsection are mainly derived from their work. Attacks are possible at each layer of the network stack. To defend against these attacks, sensor designers must build protective measures into both the physical design of the sensors and the design of the communications protocols. Security mechanisms used on the Internet are not easily adaptable to sensor networks because of the limited resources of the sensors and the ad-hoc nature of the networks.

The sensors are physically susceptible to tampering. If sensors are distributed in an unprotected area, an attacker could collect the sensors, analyze the electronics, and steal cryptographic keys. This complicates the process of bootstrapping newly deployed sensors with cryptographic keying material. To protect against this, sensors should be tamper-proof or they should self-destruct when compromised.

Sensors are also susceptible to jamming attacks against wireless radio frequencies. While it is most efficient to program sensors to communicate on one specific wireless frequency, an attacker could just as easily broadcast a more powerful signal on the same frequency and introduce noise into the communications channel. This can be alleviated by using spread spectrum, but spread spectrum will reduce battery life. Nodes could also try to detect when their channel is being jammed and sleep until the jamming device itself runs out of power, resulting in a temporary, self-induced denial of service (DoS).

Link layer protocols face similarly challenging threats. Attackers can introduce collisions that force communicating nodes to retransmit. Following a collision, a node must back-off and listen for retransmissions before attempting to resend. The attacker must continually introduce collisions until the victim runs out of power. While error-detecting mechanisms suffice for common transmission errors, they do not work for maliciously generated errors. Collisions maliciously introduced near the end of a frame can quickly exhaust the resources of the legitimate node. Researchers have proposed a link layer security architecture that addresses this problem [5].

Network layer attacks take advantage of the ad-hoc characteristic of wireless sensor networks. Any node in the network may route traffic from one node to another. A simple attack is to compromise a node in the network and force it to randomly drop messages that pass through it.

Internet style attacks have their analogue in wireless sensor networks. Misdirection attacks, such as the Internet smurf attack, work in sensor networks. The attacker can send multiple messages to broadcast addresses with a source address forged to the intended victim's address. The victim, overwhelmed with responses, will temporarily loose the ability to communicate and eventually run out of power. Filtering legitimate messages from attacks requires a hierarchy not present in many wireless sensor network routing protocols.

Some routing protocols assign a cost to potential next hops. Nodes send their messages through next hop with the lowest cost. An attacker can manipulate this strategy to convince neighbors that it has the lowest cost path to every desirable destination. One way to accomplish this is with HELLO floods [12]. To perpetrate a HELLO flood, the attacker repeatedly broadcasts HELLO messages at higher power than everyone else. Hello floods can be used by an attacker to persuade the network that it is a legitimate neighbor and a reliable next hop. An attacker can also corrupt shared routing tables by spoofing, manipulating, or replaying routing messages.

In a Sybil attack, the adversary sends a large number of messages with fabricated sources. The source can be the spoofed identity of other legitimate nodes or a totally fictitious identity. This is analogous to the hotspotter attack plaguing unauthenticated 802.11 networks. It results in legitimate nodes routing messages to virtual nodes hosted by the attacker.

By manipulating routing information, the attacker can shape the flow of traffic. The attacker could simply create a black hole by luring messages toward itself and dropping the messages. The attacker can also selectively delay messages that it is asked to forward. In a wormhole attack, the adversary tunnels messages destined for one part of the network through a path under its control. This can be used to assist in eavesdropping, message replay, or to disconnect a segment of the network.

These threats justify the need for origin and data integrity of even the simplest HELLO messages. Public key cryptography has been used on the Internet to authenticate route messages, but this has been difficult to achieve in resource constrained sensor networks.

Transport-layer protocols provide end-to-end connectivity between nodes. Protocols that use sequencing to improve reliability are, however, susceptible to DoS attacks. The classic TCP SYN flood applies to sensor networks. An adversary can flood the victim with synchronization requests and limit the ability for other nodes to communicate with the victim. One solution is to limit the number of synchronization requests accepted, but this limits both adversaries and allies. Client puzzles require the client to make a commitment to the server before it is allowed to initiate a conversation. When the client initiates a connection, the server will respond with a puzzle that the client must solve. The client must complete the puzzle and send the answer to the server before the server will accept a full connection. This solution is only satisfactory if the adversaries have similar computational constraints as the allies.

Attacks Against Authentication in Target Tracking Networks

Target tracking wireless sensor networks are susceptible to many of the attacks described above. The following informal examples corroborate the need for authentication in target tracking wireless sensor networks.

Many hierarchy tree and clustering techniques instruct redundant nodes to sleep so that network longevity is preserved. The base station or cluster head will assign the nodes a schedule to wake up and wait for further instructions. During their sleep state, the redundant nodes will deactivate their sensor component and RF transmitter. They will also temporarily turn off the radio receiver. An attacker can overwrite the wake-up schedule to force the redundant nodes to re-enable their radios more frequently and to keep the radio on for a longer duration. This attack against message integrity can drain power on the redundant nodes and reduce the life span of the network.

In another scenario, the base station or cluster heads assign roles to the sensing nodes in the network. An attacker could send spoofed messages during the network organization phase telling border nodes that they are redundant and should sleep. In this case, the border nodes, whose sensors must be active to maintain a secure perimeter, would disable their sensors. This attack against origin integrity would effectively disable any perimeter intrusion detection.

Some network organization and target tracking techniques include a network maintenance and repair phase to manage damaged nodes or nodes that are running out of power. A node will send an SOS message to the base station or cluster head to call attention to the problem. The base station will initiate a local reorganization algorithm by forcing the network into the maintenance state. This reprocessing can consume a significant amount of energy since it typically requires sleeping nodes to wake up and wait for new role assignments. An attacker could fabricate resynchronization messages appearing to come from the base station simply to reduce the lifetime of the sensor network. A craftier attacker, though, would spoof the message from the base station forcing reprocessing so that the attacker could have a node it compromised take over a forwarding role. This attack against origin authentication may allow the attacker to drain network resources and intercept or inject messages sent to the forwarding node.

Forgery of messages from border nodes can have an equally damaging effect. For example, imagine a sensor network monitoring a large corporate campus. When the sensor network detects an intruder crossing the perimeter, the company security guard must immediately visit the location to investigate the intrusion. An impostor border node could send an alert on one side of the campus as a diversion while real intruders break into the campus from the other side.

These scenarios describe simplistic attacks against wireless sensor networks used for tracking targets, yet they justify the need for authentication.

1.3.3 Survey of Message Authentication Protocols

Significant advances have been made to integrate origin integrity and data integrity in to wireless sensor networks. However, target tracking networks face unique challenges that require a solution tailored to the characteristics of the network. Many of the proposals in this chapter combine techniques for origin integrity and message integrity with other security goals, such as confidentiality or replay protection.

There are a number of requirements that guide the design of authentication protocols. Authentication protocols must be resistant to node compromise. The protocol must have low computation overhead for both the sender and the recipient of a message. The protocol must also require low communication overhead. Messages supporting the authentication protocol must function in an unreliable network. The protocol must also support immediate authentication upon receipt of the message.

Unicast Authentication

Unicast authentication refers to authenticating a message sent from one sender to one receiver. It is a much simpler goal than broadcast authentication. Typically, message authentication codes (MAC) are used to ensure message authentication, especially in a resource constrained system such as sensor networks. Static symmetric key cryptography can be implemented for unicast authentication via MACs, if one can assure that nodes will never be compromised. Symmetric key cryptography is faster and uses less storage than asymmetric algorithms. In a perfect environment, keys could be coded into the node at the factory. However, sensor networks will be deployed in hostile areas. An enemy could potentially capture a deployed sensor and reverse engineer the cryptographic mechanisms to uncover the key.

Broadcast Authentication

Broadcast authentication is a more challenging goal, which ensures that all receivers of the broadcast message be able to verify the origin integrity of the received message. When using MACs to ensure broadcast authentication, a symmetric key is shared between the sender and all potential receivers of the broadcast message. The unique challenge for broadcast authentication involves the management of that shared key. Broadcasting the shared key is not acceptable, because once the key is broadcast, everyone knows it and anyone could use it to generate a MAC for the message. Public key cryptography solves the problem of securely sharing a key for conventional Internet computing systems. However, public key cryptosystems consume far too much storage, computation, and bandwidth resources to be applied in wireless sensor networks.

SPINS

In [4], Perrig, etc., propose a suite of security protocols called SPINS that improve data confidentiality, authentication, integrity, and message freshness. Two protocols are proposed in the paper: SNEP and μTESLA. SNEP provides confidentiality, unicast authentication, and data freshness. μTESLA offers a solution for authenticated broadcast messaging in sensor networks. These two protocols specifically address self-organizing wireless sensor networks with a multihop routing topology. The network is composed of wireless sensor nodes and base stations. The base stations connect the nodes to an external network and have more computing capacity than the nodes. The paper targets three communication paths: Node to base station, base station to a single node, and base station to all nodes. It does not offer a solution for node-originated broadcast messages.

SNEP

As part of SPINS, SNEP packages data authentication, protection against replay, and weak data freshness into one cryptographic protocol. SNEP requires communicating end points to maintain a shared counter. This counter is incremented after each communication block and thus does not need to be sent along with the message. This minimizes cryptographic communication overhead to 8 bytes per message. A message authentication code (MAC) is appended to the ciphertext to provide origin authentication and message integrity.

The counter is used both in the encryption process and in calculation of the MAC. Use of the counter in the encryption process provides semantic security, which means that repeated encryption of a message M will yield unique ciphertext each round. This limits the ability of an eavesdropper to interpret the plaintext even after observing multiple encryptions of the same message. Use of the counter in calculation of the MAC provides protection against replay attacks. The counter also helps enforce weak message freshness. Message freshness is the level of assurance that a message to a node A was created by node B in response to a request from node A. The authors employ a nonce to enforce strong message freshness. Node A generates the nonce and sends it in the request to node B. Node B implicitly returns the nonce to node A by encoding it into the MAC.

μTESLA

While SNEP mitigates threats against unicast messaging, secure broadcasting messages in a wireless sensor network remains a complex problem. Authenticated broadcast messaging is typically implemented with asymmetric keys. This is too computationally expensive and requires too much storage for sensor nodes. Symmetric keying is risky because once the key is disclosed, an impostor could use it to spoof messages from the legitimate sender. As part of SPINS, μTESLA manages this problem by using a chain of symmetric keys that are periodically rotated. This mechanism works best with broadcast messages sent by a base station. An intermediary is needed for a node to broadcast a message.

To accomplish base station originated broadcasts, the base station first randomly selects a key, K_n. It then successively applies a one-way hash function to K_n, generating K_{n-1} through K_0. Keys are rotated based upon an design specific interval. When the base station broadcasts a message, it appends a MAC to the message that was generated with key K_i. Any recipients that are new to the network can not authenticate the message until the next interval, when the base station broadcasts the new key, K_{i+1}. The recipient will calculate K_i by applying the one-way hash function to K_{i+1}. Existing recipients can immediately authenticate new messages by verifying that $K_i = F\ (K_{i+1})$. Note that this use of key rotation based on time intervals requires the base station and the nodes to be loosely time synchronized.

In SPINS, wireless sensor nodes can not send authenticated broadcast messages without the assistance of the base station. Two strategies are proposed. First, the node can send the message to the base station and allow the base station to broadcast it. Second, the node can broadcast the message and let the base station manage distribution of keys.

SPINS calls upon an RC5 stream cipher in counter (CTR) mode for encryption. RC5 was selected because it has a small code footprint, it is efficient, it does not rely on multiplication, and it does not require storage of large tables. They selected a set of libraries from the open source OpenSSL project and tuned the code to improve performance for their Atmel processor. They implemented it as a macro because the costs of calling the RC5 routing as a function were too high. RC5 in counter mode uses the same function for encryption and decryption. For message authentication, the authors selected the cipher CBC-MAC with a standard MAC construction. They use the same algorithm to generate pseudo-random numbers.

SPINS addresses the security requirements for a specific hierarchy of sensor network. This system provides confidentiality for unicast messages and message authentication for both unicast and broadcast messages. The system is based on well-tuned, standard cryptographic algorithms and can scale well to other applications. Most importantly, utilization of energy, storage, and bandwidth were minimized. However, this model does not efficiently handle node-originated broadcast authentication and it does not scale to all sensor network topologies.

μTESLA Successors

μTESLA requires the recipient to wait for an interval of time before it can discern the key to authenticate a previously received message. This is not satisfactory for sensor networks with real-time constraints like target tracking networks. Some alarm messages may occur irregularly, but need to be authenticated immediately. If keys are sent at sparse intervals, the receivers must expend computation cycles working through the key chain to find the key for the interval in which the message was received. One solution is to reduce the key disclosure delay. Another is to publish multiple keys in a single message. A third solution is to use a hash tree in place of a one way hash chain. The height of a tree with N keys is $d = log_2$ *(N)*. Only d values must be sent with each message to verify that it is authentic. With these solutions, while communication costs are reduced, they remain too high for sensor networks.

In [13], the authors propose combining hash chains with trees. The leaf nodes of a tree can be used to compute the hash chain appended to the previous leaf. There are two advantages to this strategy. First, messages can be authenticated using the one-way hash of the last known key as with standard μTESLA. Second, if no messages were sent in a long period of time, the recipient can use the hash tree leaves as a shortcut through the hash chain.

Another modification of μTESLA efficiently authenticates messages sent at regular and predictable times (RPT) [13]. This protocol is designed for circumstances where the message contents are known well in advance, but some procedure requires that the message be sent regularly at some pre-specified time. For example, a base station can broadcast the same time synchronization message every day at noon. With standard μTESLA, the message could not be authenticated by a new member of the sensor network until the following day.

RPT functions by breaking time into intervals of a specific duration. A one way key chain is created with one key assigned to each interval. The sender calculates δ, the sum of the maximum propagation delay and the maximum time synchronization error. The sender first broadcasts only the MAC of the message. The sender then waits δ, and sends the message and the key used to generate the MAC. The receiver verifies that the key is still fresh, and then verifies that the MAC of the message matches the MAC originally broadcast by the sender.

The second protocol introduced by Perrig and Luk, Low Entropy Authentication (LEA) [13], is based on one-time signatures. To create a one-time signature, a directed acyclic graph is created with a one-way hash function. The initial node of the graph represents the private key. It is used to sign broadcast messages. The private key is generated with a pseudo-random function. Two edges connect this private key to two vertices: the one-way hash of the key and a checksum of the key. These two vertices are again repeatedly hashed so that the length of the chain is sufficient to sign the message. A message of x bits requires a chain of $2x$ values long. At the end of the chain, the hash of the key and the hash of the checksum are concatenated to form the public key. While this protocol efficiently generates and verifies signatures, it does not scale well for signatures of long messages. The Merkle-Winternitz signature is similar to this model, but it splits off multiple signature chains and checksum chains from the private key. This lays the foundation for LEA.

One-time signatures such as Merkle-Winternitz are effective for messages with low entropy. The length of the signature is dependent upon the length of the message to be signed. This method is preferred for signatures of short messages. The challenge with one-time signatures is that there must be a unique, authentic public key for each message. This challenge can be overcome, however, by distributing the public keys far in advance of the signed message. The sender could potentially send a set of n keys to sign the next n messages. Perrig and Luk suggest using their own RPT protocol for this purpose. The problem with this approach is that the receiver would have to store n keys until they are actually used.

TinySec

The TinySec [5] approach to securing wireless sensor networks provides message confidentiality and integrity in a way that enables simple integration into sensor network applications. Security, performance, and usability can be enabled in software with compile-time flag in a TinyOS makefile. This ease of integration into application development increases the likelihood of deployment of secure wireless sensor networks.

TinySec aims to satisfy three security goals: origin integrity, message integrity, and message confidentiality. It achieves these goals while limiting the impact on computation, memory usage, and bandwidth. To increase acceptance of TinySec by developers, TinySec is simple to include in a standard TinyOS implementation.

Security mechanisms must be easy to integrate into an application; otherwise they will not see widespread deployment. TinySec simplifies integration with a design focused on transparency and portability. TinySec achieves transparency by assuring that there is consistency between standard network APIs and network APIs that enable security. Legacy applications should not need to be significantly modified to support security mechanisms. Implementation of TinySec as a link layer module makes this transparency possible. The TinyOS radio stack was modified so that radio events will be sent to the TinySec module. In support of portability, TinySec aims to run on a variety of platforms that support TinyOS.

Developers can customize the TinySec implementation to tune the level of security around their application's security requirements. There are two modes to select from: TinySec with origin and message authentication only (TinySec-Auth) and TinySec with authentication and encryption (TinySec-AE). TinySec always provides message authentication.

Authentication and message integrity are implemented in TinySec with message authentication codes (MACs). The sender and receiver share a secret key that is used to cryptographically sign messages before their delivery and to validate messages upon receipt. If the message is tampered with or damaged during transit, the validation of the signature by the receiver will fail. The TinySec MAC is founded on the cipher-block- chaining MAC. Since CBC-MAC is not secure for variably sized messages, the authors XOR the encryption of the message length with the first plaintext block. The resultant MAC is 4 bytes. This MAC replaces the CRC field at the end of a TinyOS packet.

Use of a 4-byte MAC is an unconventional approach. MACs usually range from 8 to 16 bytes in length. The TinySec authors claim that a 4-byte MAC is sufficient for wireless sensor networks. This MAC length gives adversaries a 1 in 232 chance of successfully forging a MAC for a particular message. To succeed in this forgery, the adversary must send messages to the recipient. The limited wireless data transfer rate on wireless sensor networks only allows for 40 forgery attempts per second. It would take over 20 months to send all 232 possible MAC combinations. The recipient sensor would likely run out of power before this attack could complete. For this reason, nodes should alert the base station when the rate of MAC failures exceeds a predefined threshold.

TinySec offers both basic confidentiality and semantic security, preventing adversaries from gaining partial knowledge of the plaintext by observing repeated encryption of the same plaintext message. Semantic security requires use of initialization vectors (IVs), which increase diversity in the plaintext. This is especially important in wireless sensor networks where there is limited message entropy.

MiniSec

The research team that introduced SPINS and μTesla propose a wireless sensor network security architecture called MiniSec [16] that improves efficiency in delivering authentication and confidentiality to wireless sensor networks. MiniSec secures both unicast and broadcast communication. MiniSec's most significant contribution rests in its use of the same block cipher to provide confidentiality and authentication. MiniSec also introduces a novel semantic security strategy that only requires transmission of a fragment of the initialization vector (IV). Third, MiniSec utilizes increasingly available mote memory to defend against replay attacks.

When evaluating their security claims, the authors assume that the radio channel is insecure. An attacker, with potentially unlimited computational and energy resources, can intercept, modify, and replay any radio communications.

MiniSec clearly advances the WSN security space. TinySec minimized memory and energy use by omitting replay protection and using a single network shared key. ZigBee maintains a high level of security by sending an 8-byte IV. The authors claim that MiniSec consumes 1/3 the memory of TinySec.

MiniSec employs a block cipher mode known as Offset Code Book (OCB) [17]. OCB provides authentication and encryption in one pass. OCB takes in the message, the key, and a nonrepeating nonce, and concurrently generates the ciphertext and a tag used for authentication. The nonce provides semantic security, assuring that any two identical plaintext messages encrypted with the same key yield different ciphertext. The tag functions as a message authentication code (MAC). An advantage of OCB is that it does not cause ciphertext expansion; it produces ciphertext the same length as the plaintext. In comparison, TinySec and ZigBee require two block cipher passes; one for confidentiality and another for authentication.

Bloom filters and loose time synchronization help defend against replay attacks. Bloom filters allow nodes to efficiently store a fingerprint of received messages into an array and quickly query the array to determine if the message has already been seen. Bloom filters guarantee that replayed messages will be detected. However, there is a possibility that new messages may be identified as replayed messages.

The underlying block cipher used in the MiniSec implementation is Skipjack, with a block size of 64 bits. OCB requires the nonce to be the same length as the block size, so the counter is also 64 bits. This monotonically increasing counter guarantees semantic security. The tag, used for authentication, is recommended to be 32 bits. Symmetric keys are set to 80-bits. MiniSec uses a counter resynchronization protocol similar to the one used in SNEP.

MiniSec is composed of two schemes, MiniSec-U for unicast messaging, and MiniSec-B for broadcast messaging. The two schemes differ in the way they handle counters for replay protection. Minisec-U requires each receiver to maintain a counter for each sender. MiniSec-B uses Bloom filters. Shared counters were also used in SNEP, part of the SPINS suite.

MiniSec-U reduces the cost of transmitting counters. It conserves radio energy by only sending the last few bits (the LB value) of the 64-bit counter. The LB value is selected based on the potential for dropped packets. A low LB value can be used in environments with less potential for interference, thus reducing the communication overhead. The protocol requires two keys and two counters for each pair of communicating nodes, one per direction.

Counters can not be used in broadcast communication because of the complexity of keeping the counters synchronized among multiple nodes. OCB provides confidentiality and authentication for broadcast communication, but a novel approach is required to defend against replay attacks. One proposal is to use a sliding window. Time is split into a series of finite epochs. The length of the epoch is selected based on the estimated network latency. Each epoch is assigned a unique ID, which is used as the nonce. When an encrypted message is received, it is deciphered with both the current epoch ID and the previous epoch ID. If neither ID yields successful decipherment, the packet is considered a replay.

A problem with this approach is that a packet sent early in an epoch can be replayed throughout the epoch. Bloom filters detect replayed messages within an epoch. Each receiving node maintains two Bloom filters; one for the current epoch and one for the previous epoch. When a packet is received, the receiver first validates the packet by performing OCB decryption. If this succeeds, a test is performed to determine if the packet was replayed. The receiver first queries the Bloom filter for the packet. If it is found, the packet is considered a replay. If it is not found, it is considered new and added to the Bloom filter. This strategy will detect all replay attacks within the epoch.

MiniSec provides a high level of security with less overhead than its predecessors. Implementation details such as packet length, key length, and the authentication tag influence the level of security. While TinyOS uses only a 2-byte CRC, MiniSec uses a 4-byte tag. The authors recommend using a 32-bit tag because it was referenced in a paper on banking application security. Use of cryptographic keys replaces the functionality of the TinyOS group ID, thus this field is dropped. Like TinySec, MiniSec adds a 2-byte source address. Overall, MiniSec adds three bytes to the standard TinyOS packet.

MiniSec achieves notable efficiency while maintaining a high level of security. It reduces the transmission overhead of TinySec by two bytes. MiniSec reduces computational overhead by employing OCB. This allows for processing the ciphertext and authentication tag in one pass.

SecureSense

SecureSense [18] dynamically enables a sensor node to modify its security controls based upon observations about the external environment and requirements from the application. The approach differs from the conventional static security model where the sensor must apply the highest level of security to all messages at all times. Dynamic variation of security controls allows sensors to preserve precious power resources when the threat level is low.

Not all sensor network applications require strong adherence to the goals of information security: confidentiality, integrity, and availability. For example, in a common reference grid application sensors are deployed as a local version of GPS. A receiver uses position reports provided by the sensors to determine its location. There are multiple, redundant sensors, so availability is not a high priority. The sensors only provide their location, so confidentiality is not critical. Protection against spoofing, however, is important. In a target tracking application, however, all three security goals are required at some point in the sensor network lifetime. SecureSense allows provisioning enough security to balance threats and vulnerabilities.

SecureSense identifies five security concerns, each requiring different mitigation strategies: confidentiality, integrity, access control, semantic security, and message replay protection. These concerns can be appropriately controlled at the TinyOS link layer. Other security threats, such as RF jamming or network denial of service attacks, should be resolved at a different layer of the network stack.

The SecureSense design was guided by constraints in small-footprint systems like TinyOS. The core TinyOS operating system only occupies 400 bytes of data and code memory. In such compact systems, the energy consumption for radio communication is three orders of magnitude higher than the amount of energy required for computation. This drives SecureSense to maximize code reuse, require little or no attention from the user, and to be flexible enough that security controls may vary from one message to the next. It achieves these goals by replacing the 8-bit Active Message field in the TinyOS packet header with a SecureSense Security Composition ID (SCID).

SecureSense acts as a runtime security service in the TinyOS radio stack. This TinyOS stack includes five layers: application, active message, radio packet, radio byte, and RF module. The radio byte component sends and receives bits one-by-one over the RF module. The radio packet module spools incoming bytes to recompose them into packets. The TinyOS active message component is analogous to TCP or UDP ports in TCP/IP. It is responsible for tagging packets destined for specific application level components. SecureSense inserts a security broker between the radio packet and the radio byte components.

The security broker calls upon service modules in a service library. These services interface with cryptographic functions to fulfill requirements identified in the SCID. The first four bits in the SCID identify the security goals: confidentiality, integrity, semantic security, and replay protection. The final two bits identify cipher strength. Two other bits are currently unused. Depending on the value of the SCID, other fields in the TinyOS packet header may or may not be necessary. For example, the 1-byte group ID has been removed since group membership will implicitly be provided by the group key. The data field can range from 0 to 29 bytes. Where TinySec always requires the data field to be padded to 29 bytes, SecureSense only pad the field if encryption is required. This reduces transmission costs for small packets that don't require confidentiality. If only error detection is required, SecureSense fills the last two bytes with a CRC. If message integrity and authentication are required, the CRC is replaced by a message authentication code (MAC).

While SecureSense embraces dynamism, it requires installation of the components of the security service prior to deployment. Thus, the range of security options depends on the constraints of the hardware platform. Efficient, reusable code improves configuration options. SecureSense utilizes the RC5 block cipher to enable this efficiency. In RC5, key size, block size, and number of rounds can be defined at runtime depending on the security strength outlined in the SCID. The authors selected the RC5 implementation from TinySec since it is more efficient than the SPINS or the default C implementation.

The evaluation shows that SecureSense can conserve power by balancing security capabilities with the threat environment. This evaluation focuses specifically on energy required for communication. The actuators will consume a consistent amount of energy regardless of security configurations. It omits evaluation of energy consumed by computation. When the SCID is set to provide confidentiality, integrity, semantic security, and replay protection, SecureSense consumes an equivalent amount of communication energy to TinySec. When none of the bits of the SCID are set, SecureSense actually consumes less communication energy than TinyOS. This seems to occur because SecureSense can truncate the CRC field.

AMSecure

AMSecure [19] describes implementation of hardware accelerated cryptography in TinyOS. The hardware module is wired between the Active Message (AM) module and the radio. This strategy appropriates bytes from the message payload to identify the type of cryptography employed.

The four cryptographic modes specified in IEEE 802.15.4 [20] are offered: no cryptography, authentication-only with CBC-MAC, encryption-only with CTR mode, and authenticated encryption with CCM (CTR with CBC-MAC). The system consumes between one and nine bytes of the 29-byte TinyOS payload, depending on the cryptography utilized. All cryptographic operations are based on AES. In order to preserve backward-compatibility with legacy TinyOS messaging, a compile-time flag is set to distinguish standard TinyOS Active Messages from AMSecure messages.

The strategy was evaluated based on processing and communication overhead. By using a hardware cryptographic module, AMSecure message processing time is kept to a low, predictable level. The message overhead resulting from addition of MAC decreases as payload length increases. With a standard 29-byte payload, addition of authentication and encryption result in 20 percent message overhead. This drops to approximately 8 percent with a 100 byte payload. Processing overhead predictably increases with message payload. When receiving an AMSecure message, the AMSecure module must strip the header and MAC, perform cryptographic operations, and signal the payload to the higher layers. For a 90-byte payload, the receive processing time increases from about 500 microseconds for a standard TinyOS message to 1750 microseconds for an authenticated, encrypted AMSecure message. When transmitting a message, the AMSecure module must shift the original payload to make room for AMSecure headers. This increases processing time by a similarly predictable amount.

Interleaved Authentication

When analyzing security of wireless sensor network protocols, designers assume that nodes will be compromised. The compromise could be as simple as physical destruction of a node or sophisticated enough to allow an attacker to manipulate messages originating from the compromised node. Compromising an active node gives the attacker access to keying material, thus the attacker has the ability to calculate legitimate message authentication codes. These reputedly authentic messages can misguide the base station, divert intrusion alerts, or deplete network resources. Zhu, etc. present an interleaved hop-by-hop authentication method that can detect such false data injection attacks [21]. The method defines a threshold t for the minimum number of compromised nodes for an attack to succeed, and requires $t + 1$ nodes to send an authenticated reports of an event.

The scheme guarantees that if no more than t nodes are compromised, falsely injected data can be detected and dropped. The value t is a design parameter that can be adjusted based on the threat of node compromise. Since compromised nodes may collude in an attack against the network, at least $t + 1$ nodes must agree on an alert before it can be trusted.

Zhu's proposal assumes that the network is organized into clusters, with a subset of nodes acting as cluster heads. There are $t + 1$ nodes in each cluster, including the cluster head. The cluster may reside multiple hops from the base station. Nodes within the network can send unicast messages up and down the tree, and broadcast messages to their neighbors. Nodes share a master key with the base station and have the ability to establish pairwise keys with most of their one-hop neighbors.

The scheme relies on an association of nodes that are $t + 1$ hops apart on the path to the base station. The node closest to the base station is referred to as the upper associated node; the lower peer is referred to as the lower associated node. Upper associated nodes validate the message authentication code (MAC) appended to messages from their lower associated peers. Each alert may carry as many as $t + 1$ MACs as it travels from leaf nodes to the base station. As it travels toward the base station, a validation failure on any of the MACs will cause the message to be dropped. As long as the number of compromised nodes remains below the value t, the system can detect false data injection.

Five unique phases comprise the hop-by-hop authentication technique, including initialization and deployment, association discovery, report endorsement, en-route filtering, and base-station verification.

During node initialization and deployment, the key server loads each node with a unique id and a unique key that node shares with the base station. The node derives an authentication key from the encryption key it shares with the base station. The key server then can use one of many key establishment algorithms to initiate network key distribution. This enables nodes to establish pairwise keys with their neighbors.

The association discovery phase allows nodes to discover their associated peers both on the path downward from the base station and in reverse. The base station kicks off the process by broadcasting a hello message. Each node that receives the broadcast checks for the id of the node $t + 1$ hops up the tree, replaces that id with its own, and rebroadcasts the modified hello. This provides an upper bound of $t + 1$ node ids attached to the hello message. A receiving node records the id of the node $t + 1$ hops up the tree as its upper associated node. Note that nodes less than $t + 1$ hops from the base station do not have an upper associated node. When the hello message reaches a cluster, the cluster head assigns its leaves to upper associated nodes. The hello message can be authenticated with a broadcast authentication scheme such as μTESLA.

After the cluster notifies its leafs of their peers, it sends an acknowledgment back to the base station. The lower associated nodes authenticate the acknowledgment with the pairwise key they share with their upper associated node. Along with the MAC, the acknowledgment includes the node ids of the cluster head and the leaf nodes. As the acknowledgment is returned up the tree toward the base station, upper associated nodes learn the node id of their lower associated node. They replace the id of their lower associated node with their own id and forward the acknowledgment back up the tree. In cases where upper nodes have branches to multiple clusters, they record cluster ids and nodes in a table.

When nodes witness an event, they send a report to the base station. This hop-by-hop proposal requires $t + 1$ nodes to witness an event and endorse a report of the event. As the endorsement moves upstream, the ids of the nodes that witnessed the events and authentication codes will be appended to it. Each node computes two MACs for the event. The individual MAC is computed using the node's key with the base station. The pairwise MAC is computed using the node's pairwise key with its upper associated node. If the cluster head can authenticate a report from all its leaf nodes, it compresses the individual MACs by XORing its individual MAC with the individual MACs from the leaf nodes. The pairwise MACs are not compressed.

Upper level nodes that receive this report must authenticate it using their pairwise key with their lower associated node. If authentication succeeds, the node will extract the MAC from its lower associated node and append its own. If authentication fails, the message will be dropped. This in-route filtering assures that as long as no more than t nodes are compromised, then falsely injected data will be dropped.

Once the alert reaches the base station, the base station performs its own verification. It extracts the event data and node ids from the report. It then computes its own compressed MAC on the event data using its keys shared with the nodes in the node list. If the MAC matches the compressed MAC in the report, the alert is considered valid.

Since wireless sensor network are susceptible to damage, the Zhu proposal includes maintenance techniques. One strategy proposes piggybacking association discovery messages on base station beacons such as those sent in TinyOS. Nodes accept the first beacon they receive as their parent node. Since these beacons are sent every epoch, it is possible for nodes to change parents every epoch. While this strategy is satisfactory for dynamic networks, it is costly for networks that do no change frequently. A less costly base station initiated strategy has the parent change only if the node determines that any of the nodes in the beacon from its parent have changed. Repair can also be initiated locally when nodes detect failure of a neighbor. The proposed technique requires nodes to use GPS or similar technology to determine the physical location of their neighbors. When a node detects that its parent has failed, it will send a REPAIR message to the first node counterclockwise from the edge between itself and its deceased parent. It will then exchange messages with this node to learn the ids of the upstream nodes and rebuild any broken node associations.

Zhu's proposal provides a higher level of security than simple pairwise authentication between neighboring nodes. The base station can trust that reports from leaf nodes are authentic based on the MAC computed with pairwise key between itself and the leaf node. Intermediary nodes can authenticate reports based on pairwise keys with their lower associated nodes. As long as $t + 1$ nodes agree on an event, the system will detect a falsely injected report sent by t nodes or less. This high level of security requires a high level of node redundancy.

1.4 Open Issues

This survey has tracked the evolution of two disparate fields in sensor network research: target tracking applications and authentication protocols. While both areas of research strive for efficiency, the need for high security challenges the quest to eliminate redundancy. Target tracking methods such as GDAT and OCO improve network longevity by forcing the majority of nodes to sleep. These techniques demonstrate demonstrable gains in efficiency while maintaining a high level of accuracy in tracking intruders. Sensor network authentication protocols have similarly striven for efficiency by reducing computation and communication costs. Classic proposals such as TinySec and SPINS can integrate easily with any target tracking method. However, as Zhu demontrated, simple peer-to-peer authentication is not sufficient. Future sensor network research must focus on weaving the security of interleaved authentication protocols with the efficiency of tracking mechanisms such as OCO.

References

[1] Kung, H. T., and Vlah, D. 2003. Efficient Location Tracking Using Sensor Networks. In Proc. of 2003 IEEE Wireless Communications and Networking Conference (WCNC), March 2003.

[2] Heinzelman, W. R., Chandrakasan, A., and Balakrishnan, H. 2000. Energy-Efficient Communication Protocol for Wireless Microsensor Networks. In Proceedings of the 33rd Hawaii international Conference on System Sciences-Volume 8 - Volume 8 (January 04 - 07, 2000). HICSS. IEEE Computer Society, Washington, DC, 8020.

[3] Tran, S. P. and Yang, T. A. 2006. OCO: Optimized Communication and Organization for Target Tracking in Wireless Sensor Networks. In Proceedings of the IEEE international Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing -Vol 1 (Sutc'06) - Volume 01 (June 05 - 07, 2006). SUTC. IEEE Computer Society, Washington, DC, 428-435.

[4] Perrig, A., Szewczyk, R., Tygar, J. D., Wen, V., and Culler, D. E. 2002. SPINS: Security Protocols for Sensor Networks. Wireless Networks. 8, 5 (Sep. 2002), 521 534.

[5] Karlof, C., Sastry, N., and Wagner, D. 2004. TinySec: A link layer security architecture for wireless sensor networks. In Proceedings of the 2nd international Conference on Embedded Networked Sensor Systems (Baltimore, MD, USA, November 03 - 05, 2004). SenSys '04. ACM Press, New York, NY, 162-175.

[6] Hill, J. Szewczyk, R. Woo, A. Hollar, S. Culler, D. and Pister, K. System Architecture Directions for Networked Sensors. In Proceedings of ACM ASPLOS IX, pages 93-104,November 2000.

[7] Pister, Kris. 29 Palms Fixed/Mobile Experiment: Tracking vehicles with a UAV delivered sensor network. http://robotics.eecs.berkeley.edu/ pister/29Palms0103/.

[8] Yang, T.A., Tran, S.P., Cao, D. and Nguyen, T.A. 2006. OCO: An Efficient Method for Tracking Objects in Wireless Sensor Networks. Working draft.

[9] Rababaah, H., and Shirkhodaie, A. 2007. Guard Duty Alarming Technique (GDAT): A Novel Scheduling Approach for Target-tracking in Large-scale Distributed Sensor Networks. IEEE International Conference on System of Systems Engineering, 2007. SoSE'07. Volume , Issue , 16-18 April 2007 Page(s):1-6.

[10] Bishop, M. 2003. Computer Security: Art and Science, Addison-Wesley Professional, New York, NY, 2003.

[11] Wood, A. D. and Stankovic, J. A. 2002. Denial of Service in Sensor Networks. Computer 35, 10 (Oct. 2002), 54-62.

[12] Wood, A. D., Fang, L., Stankovic, J. A., and He, T. 2006. 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 (Alexandria, Virginia, USA, October 30 - 30, 2006). SASN '06. ACM Press, New York, NY, 35-48.

[13] Luk, M., Perrig, A., and Whillock, B. 2006. Seven cardinal properties of sensor network broadcast authentication. In Proceedings of the Fourth ACM Workshop on Security of Ad Hoc and Sensor Networks (Alexandria, Virginia, USA, October 30 - 30, 2006). SASN '06. ACM Press, New York, NY, 147-156.

[14] Yang, T. A. and Nguyen, T. A. 2006. Network security development process: a framework for teaching network security courses. J. Comput. Small Coll. 21, 4 (Apr. 2006), 203-209.

[15] Romer, K., and Mattern, F. The Design Space of Wireless Sensor Networks, IEEE Wireless Communications, pp. 54-61, December 2004.

[16] Luk, M., Mezzour, G., Perrig, A., and Gligor, V. 2007. MiniSec: a secure sensor network communication architecture. In Proceedings of the 6th international Conference on information Processing in Sensor Networks (Cambridge, Massachusetts, USA, April 25 - 27, 2007). IPSN '07. ACM, New York, NY, 479-488.

[17] Rogaway, P., Bellare, M., Black, J., and Krovetz, T. 2001. OCB: a block-cipher mode of operation for efficient authenticated encryption. In Proceedings of the 8th ACM Conference on Computer and Communications Security (Philadelphia, PA, USA, November 05 - 08, 2001). P. Samarati, Ed. CCS '01. ACM, New York, NY, 196-205. DOI= http://doi.acm.org/10.1145/501983.502011.

[18] Qi Xue and Aura Ganz. Runtime security composition for sensor networks (SecureSense). In IEEE Vehicular Technology Conference (VTC Fall 2003), October 2003.

[19] Wood, A. D. and Stankovic, J. A. 2006. AMSecure: secure link-layer communication in TinyOS for IEEE 802.15.4-based wireless sensor networks. In Proceedings of the 4th international Conference on Embedded Networked Sensor Systems (Boulder, Colorado, USA, October 31 - November 03, 2006). SenSys '06. ACM, New York, NY, 395-396.

[20] IEEE 802.15.4-2003. Wireless MAC and PHY Specifications for Low Rate Wireless Personal Area Networks (LR-WPANs), 2003.

[21] Zhu, S., S. Setia, S. Jajodia, and P. Ning, An Interleaved Hop-by-Hop Authentication Scheme for Filtering False Data Injection in Sensor Networks, IEEE Symposium on

Security and Privacy, Oakland, CA, May 9-12, 2004, pp. 260-272.