2006

A Survey of Security Issues In Wireless Sensor Networks

Yong Wang
University of Nebraska-Lincoln

Garhan Attebury
University of Nebraska-Lincoln, gattebury2@unl.edu

Byrav Ramamurthy
University of Nebraska-Lincoln, bramamurthy2@unl.edu

2ND QUARTER 2006, VOLUME 8, NO. 2

IEEE COMMUNICATIONS SURVEYS

The Electronic Magazine of Original Peer-Reviewed Survey Articles

www.comsoc.org/pubs/surveys

A SURVEY OF SECURITY ISSUES IN WIRELESS SENSOR NETWORKS

YONG WANG, GARHAN ATTEBURY, AND BYRAV RAMAMURTHY
UNIVERSITY OF NEBRASKA-LINCOLN

ABSTRACT

Wireless Sensor Networks (WSNs) are used in many applications in military, ecological, and health-related areas. These applications often include the monitoring of sensitive information such as enemy movement on the battlefield or the location of personnel in a building. Security is therefore important in WSNs. However, WSNs suffer from many constraints, including low computation capability, small memory, limited energy resources, susceptibility to physical capture, and the use of insecure wireless communication channels. These constraints make security in WSNs a challenge. In this article we present a survey of security issues in WSNs. First we outline the constraints, security requirements, and attacks with their corresponding countermeasures in WSNs. We then present a holistic view of security issues. These issues are classified into five categories: cryptography, key management, secure routing, secure data aggregation, and intrusion detection. Along the way we highlight the advantages and disadvantages of various WSN security protocols and further compare and evaluate these protocols based on each of these five categories. We also point out the open research issues in each subarea and conclude with possible future research directions on security in WSNs.

Advances in wireless communication and electronics have enabled the development of low-cost, low-power, multifunctional sensor nodes. These tiny sensor nodes, consisting of sensing, data processing, and communication components, make it possible to deploy Wireless Sensor Networks (WSNs), which represent a significant improvement over traditional wired sensor networks. WSNs can greatly simplify system design and operation, as the environment being monitored does not require the communication or energy infrastructure associated with wired networks [1].

WSNs are expected to be solutions to many applications, such as detecting and tracking the passage of troops and tanks on a battlefield, monitoring environmental pollutants, measuring traffic flows on roads, and tracking the location of personnel in a building. Many sensor networks have mission-critical tasks and thus require that security be considered [2, 3]. Improper use of information or using forged information may cause unwanted information leakage and provide inaccurate results.

While some aspects of WSNs are similar to traditional wireless ad hoc networks, important distinctions exist which greatly affect how security is achieved. The differences between sensor networks and ad hoc networks are [4]:

- The number of sensor nodes in a sensor network can be several orders of magnitude higher than the nodes in an ad hoc network.
- Sensor nodes are densely deployed.
- Sensor nodes are prone to failures due to harsh environments and energy constraints.
- The topology of a sensor network changes very frequently due to failures or mobility.
- Sensor nodes are limited in computation, memory, and power resources.
- Sensor nodes may not have global identification.

These differences greatly affect how secure data-transfer schemes are implemented in WSNs. For example, the use of radio transmission, along with the constraints of small size, low cost, and limited energy, make WSNs more susceptible to denial-of-service attacks [5]. Advanced anti-jamming techniques such as frequency-hopping spread spectrum and physical tamper-proofing of nodes are generally impossible in a sensor network due to the requirements of greater design complexity and higher energy consumption [5]. Furthermore, the limited energy and processing power of nodes makes the use of public key cryptography nearly impossible. While the results from recent studies show that public key cryptography might be feasible in sensor networks [6, 7], it remains for the most part infeasible in WSNs. Instead, most security schemes make use of symmetric key cryptography. One thing required in either case is the use of keys for secure communication. Managing key distribution is not unique to WSNs, but again constraints such as small memory capacity make centralized keying techniques impossible. Straight pairwise key sharing between every two nodes in a network does not scale to large networks with tens of thousands of nodes, as the storage requirements are too high. A security scheme in WSNs must provide efficient key distribution while maintaining the ability for communication between all relevant nodes.

In addition to key distribution, secure routing protocols must be considered. These protocols are concerned with how a node sends messages to other nodes or a base station. A key challenge is that of authenticated broadcast. Existing authenticated broadcast methods often rely on public key cryptography and include high computational overhead making them infeasible in WSNs. Secure routing protocols proposed for use in WSNs, such as SPINS [8], must consider these factors. Additionally, the constraint on energy in WSNs leads to the desire for data aggregation. This aggregation of sensor data needs to be secure in order to ensure information integrity and confidentiality [9, 10]. While this is achievable through cryptography, an aggregation scheme must take into account the constraints in WSNs and the unique characteristics of the cryptography and routing schemes. It is also desirable for secure data aggregation protocols to be flexible, allowing lower levels of security for less important data, thus saving energy, and allowing higher levels of security for more sensitive data, thus consuming more energy.

As with any network, awareness of compromised nodes and attacks is desirable. Many security schemes provide assurance that data remain intact and communication unaffected as long as fewer than *t* nodes are compromised [11]. The ability of a node or base station to detect when other nodes are compromised enables them to take action, either ignoring the compromised data or reconfiguring the network to eliminate the threat.

The remainder of this article discusses the above areas in more detail and considers how they are all required to form a complete WSN security scheme. A few existing surveys on security issues in ad hoc networks can be found in [12–14]; however, only small sections of these surveys focus on WSNs. A recent survey article on security issues in mobile ad hoc networks also included an overview of security issues in WSNs [15]. However, the article did not discuss cryptography and intrusion detection issues. Further, it included only a small portion of the available literature on security in WSNs.

The rest of this article is organized as follows. Background information on WSNs is presented, followed by a discussion of attacks in the different network layers of sensor networks. Then we focus on the selection of cryptography in WSNs, key management, secure routing schemes, secure data aggregation, and intrusion detection systems. We discuss future research directions on security in WSNs and then conclude the article.

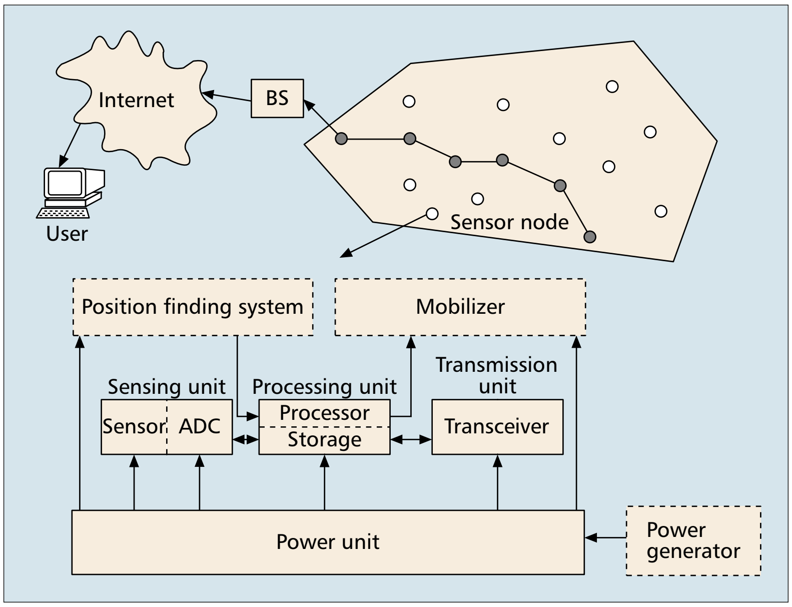

Figure 1. *The components of a sensor node (Source: [4]).*

Background

Communication Architecture

A WSN is usually composed of hundreds or thousands of sensor nodes. These sensor nodes are often densely deployed in a sensor field and have the capability to collect data and route data back to a base station (BS). A sensor consists of four basic parts: a sensing unit, a processing unit, a transceiver unit, and a power unit [4]. It may also have additional application-dependent components such as a location finding system, power generator, and mobilizer (Fig. 1). Sensing units are usually composed of two subunits: sensors and analog-to-digital converters (ADCs). The ADCs convert the analog signals produced by the sensors to digital signals based on the observed phenomenon. The processing unit, which is generally associated with a small storage unit, manages the procedures that make the sensor node collaborate with the other nodes. A transceiver unit connects the node to the network. One of the most important units is the power unit. A power unit may be finite (e.g., a single battery) or may be supported by power scavenging devices (e.g., solar cells). Most of the sensor network routing techniques and sensing tasks require knowledge of location, which is provided by a location finding system. Finally, a mobilizer may sometimes be needed to move the sensor node, depending on the application.

The protocol stack used in sensor nodes contains physical, data link, network, transport, and application layers defined as follows [4]:

- Physical layer: responsible for frequency selection, carrier frequency generation, signal deflection, modulation, and data encryption
- Data link layer: responsible for the multiplexing of data streams, data frame detection, medium access, and error control; as well as ensuring reliable point-to-point and point-to-multipoint connections
- Network layer: responsible for specifying the assignment of addresses and how packets are forwarded
- Transport layer: responsible for specifying how the reliable transport of packets will take place
- Application layer: responsible for specifying how the data are requested and provided for both individual sensor nodes and interactions with the end user

	Berkeley mote [16]					EYES [17]	Medusa MK-2 [18]	Imote[19]
	WeC	rene2	rene2	dot	mica			
Month/Year	09/99	10/00	06/01	08/01	02/02	03/02	09/02	01/03
CPU	AT90LS8535		ATmega163		ATmega103[1]	MSP 430F149	40MHz ARM THUMB	ARM core 12MHz
Prog. memory	8KB		16KB		128KB	60KB	1MB	512KB
RAM	0.5KB		1KB		4KB	2KB	136KB	64KB
Radio	RFM TR1000 916MHz				RFM TR1001 868.35MHz		RFM TR1000 916MHz	BT 2.4 GHz
Rate	10 kb/s				10/40 kb/s	115 kb/s	115 kb/s	100kb/s

[1] Later versions are an ATmega128 running in 103 mode.

Table 1. *Variety of real-life sensor nodes.*

Constraints in WSNs

Individual sensor nodes in a WSN are inherently resource constrained. They have limited processing capability, storage capacity, and communication bandwidth. Each of these limitations is due in part to the two greatest constraints — limited energy and physical size. Table 1 shows several currently available sensor node platforms. The design of security services in WSNs must consider the hardware constraints of the sensor nodes:

- *Energy*: energy consumption in sensor nodes can be categorized into three parts:
  –Energy for the sensor transducer
  –Energy for communication among sensor nodes
  –Energy for microprocessor computation

The study in [20, 21] found that each bit transmitted in WSNs consumes about as much power as executing 800–1000 instructions. Thus, communication is more costly than computation in WSNs. Any message expansion caused by security mechanisms comes at a significant cost. Further, higher security levels in WSNs usually correspond to more energy consumption for cryptographic functions. Thus, WSNs can be divided into different security levels, depending on energy cost [22, 23].

- *Computation*: the embedded processors in sensor nodes are generally not as powerful as those in nodes of a wired or ad hoc network. As such, complex cryptographic algorithms cannot be used in WSNs.
- *Memory*: memory in a sensor node usually includes flash memory and RAM. Flash memory is used for storing downloaded application code and RAM is used for storing application programs, sensor data, and intermediate computations. There is usually not enough space to run complicated algorithms after loading OS and application code. In the SmartDust project, for example, TinyOS consumes about 3500 bytes of instruction memory, leaving only 4500 bytes for security and applications [20, 21]. This makes it impractical to use the majority of current security algorithms [8]. With an Intel Mote, the situation is slightly improved, but still far from meeting the requirements of many algorithms.
- *Transmission range*: the communication range of sensor nodes is limited both technically and by the need to conserve energy. The actual range achieved from a given transmission signal strength is dependent on various environmental factors such as weather and terrain.

Security Requirements

The goal of security services in WSNs is to protect the information and resources from attacks and misbehavior. The security requirements in WSNs include:

- Availability, which ensures that the desired network services are available even in the presence of denial-of-service attacks
- Authorization, which ensures that only authorized sensors can be involved in providing information to network services
- Authentication, which ensures that the communication from one node to another node is genuine, that is, a malicious node cannot masquerade as a trusted network node
- Confidentiality, which ensures that a given message cannot be understood by anyone other than the desired recipients
- Integrity, which ensures that a message sent from one node to another is not modified by malicious intermediate nodes
- Nonrepudiation, which denotes that a node cannot deny sending a message it has previously sent
- Freshness, which implies that the data is recent and ensures that no adversary can replay old messages

Moreover, as new sensors are deployed and old sensors fail, we suggest that forward and backward secrecy should also be considered:

- Forward secrecy: a sensor should not be able to read any future messages after it leaves the network.
- Backward secrecy: a joining sensor should not be able to read any previously transmitted message.

The security services in WSNs are usually centered around cryptography. However, due to the constraints in WSNs, many already existing secure algorithms are not practical for use. We discuss this problem in the section "Cryptography in WSNs" below.

Threat Model

In WSNs, it is usually assumed that an attacker may know the security mechanisms that are deployed in a sensor network; they may be able to compromise a node or even physically capture a node. Due to the high cost of deploying tamper-resistant sensor nodes, most WSN nodes are viewed as non-tamper-resistant. Further, once a node is compromised, the attacker is capable of stealing the key materials contained within that node.

Base stations in WSNs are usually regarded as trustworthy. Most research studies focus on secure routing between sensors and the base station. Deng *et al.* considered strategies against threats which can lead to the failure of the base station [24].

Attacks in sensor networks can be classified into the following categories:

- Outsider versus insider attacks: outside attacks are defined as attacks from nodes which do not belong to a WSN; insider attacks occur when legitimate nodes of a WSN behave in unintended or unauthorized ways.
- Passive versus active attacks: passive attacks include eavesdropping on or monitoring packets exchanged within a WSN; active attacks involve some modifications of the data steam or the creation of a false stream.
- Mote-class versus laptop-class attacks: in mote-class attacks, an adversary attacks a WSN by using a few nodes with similar capabilities to the network nodes; in laptop-class attacks, an adversary can use more powerful devices (e.g., a laptop) to attack a WSN. These devices have greater transmission range, processing power, and energy reserves than the network nodes.

Evaluation

We suggest using the following metrics to evaluate whether a security scheme is appropriate in WSNs.

- Security: a security scheme has to meet the requirements discussed above.
- Resiliency: in case a few nodes are compromised, a security scheme should still protect against the attacks.
- Energy efficiency: a security scheme must be energy efficient so as to maximize node and network lifetime.
- Flexibility: key management needs to be flexible so as to allow for different network deployment methods, such as random node scattering and predetermined node placement.
- Scalability: a security scheme should be able to scale without compromising the security requirements.
- Fault-tolerance: a security scheme should continue to provide security services in the presence of faults such as failed nodes.
- Self-healing: sensors may fail or run out of energy. The remaining sensors may need to be reorganized to maintain a set level of security.
- Assurance: assurance is the ability to disseminate different information at different levels to end-users [25]. A security scheme should offer choices with regard to desired reliability, latency, and so on.

Attacks in Sensor Networks

WSNs are vulnerable to various types of attacks. According to the security requirements in WSNs, these attacks can be categorized as [3]:

- Attacks on secrecy and authentication: standard cryptographic techniques can protect the secrecy and authenticity of communication channels from outsider attacks such as eavesdropping, packet replay attacks, and modification or spoofing of packets.
- Attacks on network availability: attacks on availability are often referred to as denial-of-service (DoS) attacks. DoS attacks may target any layer of a sensor network.
- Stealthy attacks against service integrity: in a stealthy attack, the goal of the attacker is to make the network accept a false data value. For example, an attacker compromises a sensor node and injects a false data value through that sensor node.

In these attacks, keeping the sensor network available for its intended use is essential. DoS attacks against WSNs may permit real-world damage to the health and safety of people [5]. In this section, we focus only on DoS attacks and their countermeasures in sensor networks. We discuss attacks on secrecy and authentication in the section "Secure Routing Protocols," and discuss stealthy attacks and countermeasures in the section "Intrusion Detection" below.

The DoS attack usually refers to an adversary's attempt to disrupt, subvert, or destroy a network. However, a DoS attack can be any event that diminishes or eliminates a network's capacity to perform its expected function [5]. Sensor networks are usually divided into layers, and this layered architecture makes WSNs vulnerable to DoS attacks, as DoS attacks may occur in any layer of a sensor network.

Previous discussions on DoS attacks in WSNs can be found in [3, 5, 26, 27]. The remainder of this section summarizes the possible DoS attacks and countermeasures in each layer of a sensor network.

Physical Layer

The physical layer is responsible for frequency selection, carrier frequency generation, signal detection, modulation, and data encryption [4]. As with any radio-based medium, there exists the possibility of jamming in WSNs. In addition, nodes in WSNs may be deployed in hostile or insecure environments where an attacker has easy physical access. These two vulnerabilities are explored in this subsection.

Jamming — Jamming is a type of attack which interferes with the radio frequencies that a network's nodes are using [3, 5]. A jamming source may either be powerful enough to disrupt the entire network or less powerful and only able to disrupt a smaller portion of the network. Even with lesser-powered jamming sources, such as a small compromised subset of the network's sensor nodes, an adversary has the potential to disrupt the entire network provided the jamming sources are randomly distributed in the network.

Typical defenses against jamming involve variations of spread-spectrum communication such as frequency hopping and code spreading [5]. Frequency-hopping spread spectrum (FHSS) is a method of transmitting signals by rapidly switching a carrier among many frequency channels using a pseudo random sequence known to both transmitter and receiver. Without being able to follow the frequency selection sequence, an attacker is unable to jam the frequency being used at a given moment in time. However, as the range of possible frequencies is limited, an attacker may instead jam a wide section of the frequency band.

Code spreading is another technique used to defend against jamming attacks and is common in mobile networks. However, this technique requires greater design complexity and energy, thus restricting its use in WSNs. In general, to maintain low cost and low power requirements, sensor devices are limited to single-frequency use and are therefore highly susceptible to jamming attacks.

Tampering — Another physical layer attack is tampering [5]. Given physical access to a node, an attacker can extract sensitive information such as cryptographic keys or other data on the node. The node may also be altered or replaced to create a compromised node which the attacker controls. One defense to this attack involves tamper-proofing the node's physical package [5]. However, it is usually assumed that the sensor nodes are not tamper-proofed in WSNs due to the additional cost. This indicates that a security scheme must consider the situation in which sensor nodes are compromised.

LINK LAYER

The data link layer is responsible for the multiplexing of data streams, data frame detection, medium access, and error control [4]. It ensures reliable point-to-point and point-to-multipoint connections in a communication network. Attacks at the link layer include purposely introduced collisions, resource exhaustion, and unfairness. This subsection looks at each of these three link-layer attack categories [5].

Collisions — A collision occurs when two nodes attempt to transmit on the same frequency simultaneously [5]. When packets collide, a change will likely occur in the data portion, causing a checksum mismatch at the receiving end. The packet will then be discarded as invalid. An adversary may strategically cause collisions in specific packets such as ACK control messages. A possible result of such collisions is the costly exponential back-off in certain media access control (MAC) protocols.

A typical defense against collisions is the use of error-correcting codes [5]. Most codes work best with low levels of collisions, such as those caused by environmental or probabilistic errors. However, these codes also add additional processing and communication overhead. It is reasonable to assume that an attacker will always be able to corrupt more than what can be corrected. While it is possible to detect these malicious collisions, no complete defenses against them are known at this time.

Exhaustion — Repeated collisions can also be used by an attacker to cause resource exhaustion [5]. For example, a naive link-layer implementation may continuously attempt to retransmit the corrupted packets. Unless these hopeless retransmissions are discovered or prevented, the energy reserves of the transmitting node and those surrounding it will be quickly depleted.

A possible solution is to apply rate limits to the MAC admission control such that the network can ignore excessive requests, thus preventing the energy drain caused by repeated transmissions [5]. A second technique is to use time-division multiplexing where each node is allotted a time slot in which it can transmit [5]. This eliminates the need of arbitration for each frame and can solve the indefinite postponement problem in a back-off algorithm. However, it is still susceptible to collisions.

Unfairness — Unfairness can be considered a weak form of a DoS attack [5]. An attacker may cause unfairness in a network by intermittently using the above link-layer attacks. Instead of preventing access to a service outright, an attacker can degrade it in order to gain an advantage such as causing other nodes in a real-time MAC protocol to miss their transmission deadline. The use of small frames lessens the effect of such attacks by reducing the amount of time an attacker can capture the communication channel. However, this technique often reduces efficiency and is susceptible to further unfairness, for example, when an attacker is trying to retransmit quickly instead of randomly delaying.

NETWORK AND ROUTING LAYER

The network and routing layer of sensor networks is usually designed according to the following principles [4]:

- Power efficiency is an important consideration.
- Sensor networks are mostly data-centric.
- An ideal sensor network has attribute-based addressing and location awareness.

The attacks in the network and the routing layer include the following.

Spoofed, Altered, or Replayed Routing Information — The most direct attack against a routing protocol in any network is to target the routing information itself while it is being exchanged between nodes. An attacker may spoof, alter, or replay routing information in order to disrupt traffic in the network [26]. These disruptions include the creation of routing loops, attracting or repelling network traffic from select nodes, extending and shortening source routes, generating fake error messages, partitioning the network, and increasing end-to-end latency.

A countermeasure against spoofing and alteration is to append a message authentication code (MAC) after the message. By adding a MAC to the message, the receivers can verify whether the messages have been spoofed or altered. To defend against replayed information, counters or timestamps can be included in the messages [8].

Selective Forwarding — A significant assumption made in multihop networks is that all nodes in the network will accurately forward received messages. An attacker may create malicious nodes which selectively forward only certain messages and simply drop others [26]. A specific form of this attack is the black hole attack in which a node drops all messages it receives. One defense against selective forwarding attacks is using multiple paths to send data [26]. A second defense is to detect the malicious node or assume it has failed and seek an alternative route.

Sinkhole — In a sinkhole attack, an attacker makes a compromised node look more attractive to surrounding nodes by forging routing information [5, 26]. The end result is that surrounding nodes will choose the compromised node as the next node to route their data through. This type of attack makes selective forwarding very simple, as all traffic from a large area in the network will flow through the adversary's node.

Sybil — The Sybil attack is a case where one node presents more than one identity to the network [3, 26, 27]. Protocols and algorithms which are easily affected include fault-tolerant schemes, distributed storage, and network-topology maintenance. For example, a distributed storage scheme may rely on there being three replicas of the same data to achieve a given level of redundancy. If a compromised node pretends to be two of the three nodes, the algorithms used may conclude that redundancy has been achieved while in reality it has not.

Wormholes — A wormhole is a low-latency link between two portions of the network over which an attacker replays network messages [26]. This link may be established either by a single node forwarding messages between two adjacent but otherwise non-neighboring nodes or by a pair of nodes in different parts of the network communicating with each other. The latter case is closely related to the sinkhole attack, as an attacking node near the base station can provide a one-hop link to that base station via the other attacking node in a distant part of the network. Hu *et al.* presented a novel and general mechanism called packet leashes for detecting and defending against wormhole attacks [28]. Two types of leashes were introduced: geographic leashes and temporal leashes. The proposed mechanisms can also be used in WSNs.

Network	Attacks	Defense
Physical	Jamming Tampering	Spread-spectrum, priority messages, lower duty cycle, region mapping, mode change Tamper-proofing, hiding
Link	Collision Exhaustion Unfairness	Error-correcting code Rate limitation Small frames
Network and routing	Spoofed, altered or replayed routing information Selective forwarding Sinkhole Sybil Wormholes Hello flood attacks Acknowledgment spoofing	Egress filtering, authentication, monitoring Redundancy, probing Authentication, monitoring, redundancy Authentication, probing Authentication, packet leashes by using geographic and temporal information Authentication, verify the bidirectional link Authentication
Transport	Flooding Desynchronization	Client puzzles Authentication

Table 2. *Sensor network layers and denial-of-service defenses.*

Hello Flood Attacks — Many protocols which use HELLO packets make the naive assumption that receiving such a packet means the sender is within radio range and is therefore a neighbor. An attacker may use a high-powered transmitter to trick a large area of nodes into believing they are neighbors of that transmitting node [26]. If the attacker falsely broadcasts a superior route to the base station, all of these nodes will attempt transmission to the attacking node, despite many being out of radio range in reality.

Acknowledgment Spoofing — Routing algorithms used in sensor networks sometimes require Acknowledgments to be used. An attacking node can spoof the Acknowledgments of overheard packets destined for neighboring nodes in order to provide false information to those neighboring nodes [26]. An example of such false information is claiming that a node is alive when in fact it is dead.

Transport Layer

The transport layer is responsible for managing end-to-end connections [4]. Two possible attacks in this layer, flooding and desynchronization, are discussed in this subsection.

Flooding — Whenever a protocol is required to maintain state at either end of a connection it becomes vulnerable to memory exhaustion through flooding [5]. An attacker may repeatedly make new connection requests until the resources required by each connection are exhausted or reach a maximum limit. In either case, further legitimate requests will be ignored. One proposed solution to this problem is to require that each connecting client demonstrate its commitment to the connection by solving a puzzle [5]. The idea is that a connecting client will not needlessly waste its resources creating unnecessary connections. Given that an attacker does not likely have infinite resources, it will be impossible for him/her to create new connections fast enough to cause resource starvation on the serving node. While these puzzles do include processing overhead, this technique is more desirable than excessive communication.

Desynchronization — Desynchronization refers to the disruption of an existing connection [5]. An attacker may, for example, repeatedly spoof messages to an end host, causing that host to request the retransmission of missed frames. If timed correctly, an attacker may degrade or even prevent the ability of the end hosts to successfully exchange data, thus causing them to instead waste energy by attempting to recover from errors which never really existed.

A possible solution to this type of attack is to require authentication of all packets communicated between hosts [5]. Provided that the authentication method is itself secure, an attacker will be unable to send the spoofed messages to the end hosts.

Table 2 shows the possible DoS attacks and countermeasures in WSNs.

In the following sections we discuss cryptography, key management protocols, secure routing protocols, secure data aggregation, and intrusion detection for WSNs. For the remainder of this article, we use the following notation:

- A, B are principals such as communicating nodes.
- ID_A denotes the sensor identifier of node A.
- N_A is a nonce generated by A (a nonce is an unpredictable bit string, usually used to achieve freshness).
- K_{AB} denotes the secret pairwise key shared between A and B.
- M_K is the encryption of message M with key K
- $MAC(K,M)$ denotes the computation of the message authentication code of message M with key K
- $A \rightarrow B$ denotes A unicasts a message to B
- $A \rightarrow *$ denotes A broadcasts a message to its neighbors

Cryptography in WSNs

Selecting the most appropriate cryptographic method is vital in WSNs because all security services are ensured by cryptography. Cryptographic methods used in WSNs should meet the constraints of sensor nodes and be evaluated by code size, data size, processing time, and power consumption. In this section, we focus on the selection of cryptography in WSNs. Public key cryptography, discussed first, is followed by symmetric key cryptography.

Public Key Cryptography in WSNs

Many researchers believe that the code size, data size, processing time, and power consumption make it undesirable for public key algorithm techniques, such as the Diffie–Hellman key agreement protocol [29] or RSA signatures [30], to be

Algorithm	Operation time (s)
ECC secp160r1	0.81s
ECC secp224r1	2.19s
RSA-1024 public-key $e = 2^{16} + 1$	0.43s
RSA-1024 private key w. CRT[1]	10.99
RSA-2048 public-key $e = 2^{16} + 1$	1.94s
RSA-2048 private-key w. CRT[1]	83.26
[1] Chinese Remainder Theory	

Table 3. *Public key cryptography: average ECC and RSA execution times (Source: [6]).*

employed in WSNs.

Public key algorithms such as RSA are computationally intensive and usually execute thousands or even millions of multiplication instructions to perform a single security operation. Further, a microprocessor's public key algorithm efficiency is primarily determined by the number of clock cycles required to perform a multiply instruction [31]. Brown *et al.* found that public key algorithms such as RSA usually require on the order of tens of seconds and up to minutes to perform encryption and decryption operations in constrained wireless devices which exposes a vulnerability to DoS attacks [32]. On the other hand, Carman *et al.* found that it usually takes a microprocessor thousands of nano-joules to do a simple multiply function with a 128 bit result [31]. In contrast, symmetric key cryptography algorithms and hash functions consume much less computational energy than public key algorithms. For example, the encryption of a 1024-bit block consumes approximately 42 mJ on the MC68328 DragonBall processor using RSA, while the estimated energy consumption for a 128 bit AES block is a much lower at 0.104 mJ [31].

Recent studies have shown that it is feasible to apply public key cryptography to sensor networks by using the right selection of algorithms and associated parameters, optimization, and low-power techniques [6, 7, 33]. The investigated public key algorithms include Rabin's Scheme [34], NtruEncrypt [35], RSA [30], and Elliptic Curve Cryptography (ECC) [36, 37]. Most studies in literature focus on RSA and ECC algorithms. The attraction of ECC is that it appears to offer equal security for a far smaller key size, thereby reducing processing and communication overhead. For example, RSA with 1024 bit keys (RSA-1024) provides a currently accepted level of security for many applications and is equivalent in strength to ECC with 160 bit keys (ECC-160) [38]. To protect data beyond the year 2010, RSA Security recommends RSA-2048 as the new minimum key size which is equivalent to ECC with 224 bit keys (ECC-224) [39]. Table 3 summarizes the execution time of ECC and RSA implementations on an Atmel ATmega128 processor (used by Mica2 mote) [6]. The execution time is measured on average for a point multiplication in ECC and a modular exponential operation in RSA. ECC secp160r1 and secp224r1 are two standardized elliptic curves defined in [40]. As shown in Table 3, by using the small integer $e = 2^{16} + 1$ as the public key, RSA public key operation is slightly faster than ECC point multiplication. However, ECC point multiplication outperforms RSA private key operation by an order of magnitude. The RSA private key operation, which is too slow, limits its use in a sensor node. ECC has no such issues since both the public key operation and private key operation use the same point multiplication operations.

Wander *et al.* investigated the energy cost of authentication and key exchange based on RSA and ECC cryptography on an Atmel ATmega128 processor [7]. The result is shown in Table 4. The ECC-based signature is generated and verified using the Elliptic Curve Digital Signature Algorithm (ECDSA) [41]. The key exchange protocol is a simplified version of the SSL handshake, which involves two parties: a client initiating the communication and a server responding to the initiation [42]. The WSN is assumed to be administered by a central point with each sensor having a certificate signed by the central point's private key using a RSA or ECC signature. In the handshake process, the two parties verify each other's certificate and negotiate the session key to be used in the communication. As Table 4 shows, in comparison with RSA cryptography at the same security level, ECDSA signatures are significantly cheaper than RSA signatures and ECDSA verifications are within reasonable range of RSA verifications. Further, the ECC-based key exchange protocol outperforms the RSA-based key exchange protocol at the server side, and there is almost no difference in the energy cost for these two key exchange protocols at the client side. In addition, the relative performance advantage of ECC over RSA increases as the key size increases in terms of the execution time and energy cost. Tables 3 and 4 indicate that ECC is more appropriate than RSA for use in sensor networks.

The implementation of RSA and ECC cryptography on Mica2 motes further proved that a public key based protocol is viable for WSNs. Two modules, TinyPK [43], based on RSA, and TinyECC [44], based on ECC, have been designed and implemented on Mica2 motes using the TinyOS development environment. Similar work was also conducted by Malan *et al.* on ECC cryptography using a Mica2 mote [45]. In their work, ECC was used to distribute a single symmetric key for the link-layer encryption provided by the TinySec module [46].

While public key cryptography may be possible in sensor nodes, the public key operations are still expensive. The assumptions in [33, 45] may not be satisfied in some applications. For example, the work in [33, 45] concentrated on the public key operations only, assuming the private key operations will be performed by a base station or a third party. By selecting appropriate parameters, for example, using the small integer $e = 2^{16} + 1$ as the public key, the public key operation time can be extremely fast while the private key operation time does not change. The limitation of private key operation

Algorithm	Signature		Key Exchange	
	Sign	Verify	Client	Server
RSA-1024	304	11.9	15.4	304
ECDSA-160	22.82	45.09	22.3	22.3
RSA-2048	2302.7	53.7	57.2	2302.7
ECDSA-224	61.54	121.98	60.4	60.4

Table 4. *Public key cryptography: average energy costs of digital signature and key exchange computations [mJ]. (Source: [7]).*

Algorithm	Operation time (ms)
Skipjack (C) [47]	0.38ms
RC5 (C, assembly) [48]	0.26ms

■ Table 5. *Symmetric key cryptography: average RC5 and skipjack execution times (Source: [46]).*

Algorithm	Energy
SHA-1 (C) [49]	5.9 mJ/byte
AES-128 Enc/Dec (assembly) [50]	1.62/2.49 mJ/byte

■ Table 6. *Symmetric key cryptography: average energy numbers for AES and SHA-1. (Source: [7]).*

occurring only at a base station makes many security services using public key algorithms not available under these schemes. Such services include peer-to-peer authentication and secure data aggregation.

In contrast, Tables 5 and 6 show the execution time and energy cost of two symmetric cryptography protocols on an Atmel ATmega128 processor. In Table 5, the execution time was measured on a 64 bit block using a 80 bit key. From the table we can see that symmetric key cryptography is faster and consumes less energy as compared to public key cryptography. In the next section we focus on symmetric key cryptography.

Symmetric Key Cryptography in WSNs

The constraints on computation and power consumption in sensor nodes limit the application of public key cryptography in WSNs. Thus, most research studies focus on symmetric key cryptography in sensor networks.

Five popular encryption schemes, RC4 [51], RC5 [48], IDEA [51], SHA-1 [49], and MD5 [51, 52], were evaluated on six different microprocessors ranging in word size from 8 bit (Atmel AVR) to 16 bit (Mitsubishi M16C) to 32 bit widths (StrongARM, XScale) in [53]. The execution time and code memory size were measured for each algorithm and platform. The experiments indicated uniform cryptographic cost for each encryption class and each architecture class. The impact of caches was negligible while Instruction Set Architecture (ISA) support was limited to specific effects on certain algorithms. Moreover, hashing algorithms (MD5 and SHA-11) incurred almost an order of a magnitude higher overhead than encryption algorithms (RC4, RC5, and IDEA).

In [54], Law *et al.* evaluated two symmetric key algorithms: RC5 and TEA [55]. They further evaluated six block ciphers, including RC5 and RC6 [56], Rijndael [50], MISTY1 [57], KASUMI [58], and Camellia [59] on IAR Systems' MSP430F149 in [60]. The benchmark parameters were code, data memory, and CPU cycles. The evaluation results showed that Rijndael is suitable for high-security and energy-efficiency requirements while MISTY1 is suitable for good storage and energy efficiency. The evaluation results in [60] disagreed with the work in [8] in which RC5 was selected as the encryption/decryption scheme, and with the work in [22] in which RC6 was selected. The work in [60] provides a good resource for deciding which symmetric algorithm should be adopted in sensor networks.

The performance of symmetric key cryptography is mainly decided by the following factors:

- Embedded data bus width: many encryption algorithms prefer 32 bit word arithmetic, but most embedded processors usually use 8 or 16 bit wide data bus.
- Instruction set: the Instruction Set Architecture (ISA) has specific effects on certain algorithms. For example, most embedded processors do not support the variable-bit rotation instruction like ROL (rotate bits left) of the Intel architecture, which greatly improves the performance of RC5.

Due to the constraints in sensor nodes, symmetric key cryptography is preferred in a WSN.

Open Research Issues

Selecting the appropriate cryptography method for sensor nodes is fundamental to providing security services in WSNs. However, the decision depends on the computation and communication capability of the sensor nodes. Open research issues range from cryptography algorithms to hardware design as described below:

- Recent studies on public key cryptography have demonstrated that public key operations may be practical in sensor networks. However, private key operations are still too expensive in terms of computation and energy cost to accomplish in a sensor node. The application of private key operations to sensor nodes needs to be studied further.
- Symmetric key cryptography is superior to public key cryptography in terms of speed and low energy cost. However, the key distribution schemes based on symmetric key cryptography are not perfect. Efficient and flexible key distribution schemes need to be designed.
- It is also likely that more powerful motes will need to be designed in order to support the increasing requirements for computation and communication in sensor nodes.

Key Management Protocols

Key management is a core mechanism to ensure the security of network services and applications in WSNs. The goal of key management is to establish required keys between sensor nodes which must exchange data. Further, a key management scheme should also support node addition and revocation while working in undefined deployment environments. Due to the constraints on sensor nodes, key management schemes in WSNs have many differences with the schemes in ad hoc networks.

As shown above, public key cryptography suffers from limitations in WSNs. Thus, most proposed key management schemes are based on symmetric key cryptography. Further, a straight pairwise private key sharing scheme between every pair of nodes is also impractical in WSNs. A pairwise private key sharing scheme requires predistribution and storage of $n - 1$ keys in each node, where n is the number of nodes in a sensor network. Due to the large amount of memory required, pairwise schemes are not viable when the network size is large. Moreover, most key pairs would be unusable since direct communication is possible only among neighboring nodes. This scheme is also not flexible for node addition and revocation. In this section, we discuss key management protocols in WSNs. Another investigation of key management mechanisms for WSNs can be found in [61].

Figure 2 shows a taxonomy of key management protocols in WSNs. According to the network structure, the protocols

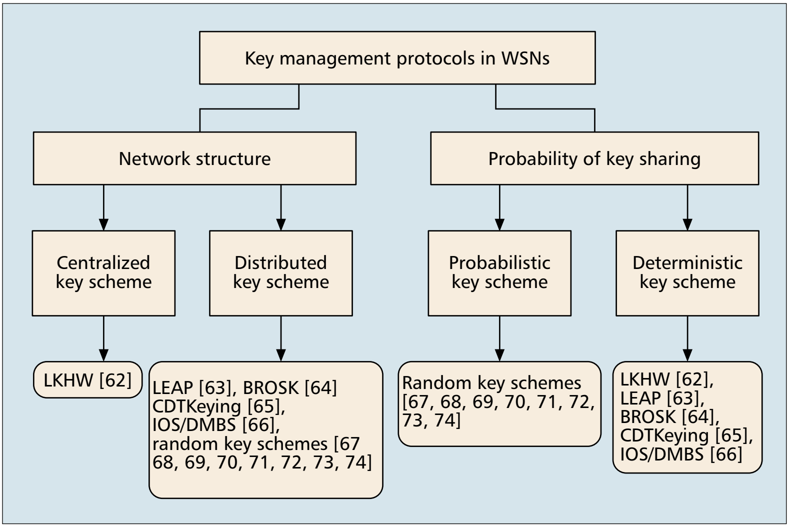
Key management protocols in WSNs
Network structure
Probability of key sharing
Centralized key scheme
Distributed key scheme
Probabilistic key scheme
Deterministic key scheme
LKHW [62]
LEAP [63], BROSK [64] CDTKeying [65], IOS/DMBS [66], random key schemes [67 68, 69, 70, 71, 72, 73, 74]
Random key schemes [67, 68, 69, 70, 71, 72, 73, 74]
LKHW [62], LEAP [63], BROSK [64], CDTKeying [65], IOS/DMBS [66]

Figure 2. *Key management protocols in WSNs: a taxonomy.*

can be divided into centralized key schemes and distributed key schemes. According to the probability of key sharing between a pair of sensor nodes, the protocols can be divided into probabilistic key schemes and deterministic key schemes. In this section, we present a detailed overview of the main key management protocols in WSNs. We start with key management protocols based on network structure.

Network Structure Based Key Management Protocols

The underlying network structure plays a significant role in the operation of key management protocols. According to the structure, the protocols can be divided into two categories: centralized key schemes and distributed key schemes.

Centralized Key Management Schemes — In a centralized key scheme, there is only one entity, often called a key distribution center (KDC), that controls the generation, regeneration, and distribution of keys. The only proposed centralized key management scheme for WSNs in the current literature is the LKHW scheme, which is based on the Logical Key Hierarchy (LKH) [62]. In this scheme, the base station is treated as a KDC and all keys are logically distributed in a tree rooted at the base station.

The central controller does not have to rely on any auxiliary entity to perform access control and key distribution. However, with only one managing entity, the central server is a single point of failure. The entire network and its security will be affected if there is a problem with the controller. During the time when the controller is not working, the network becomes vulnerable as keys are not generated, regenerated, and distributed. Furthermore, the network may become too large to be managed by a single entity, thus affecting scalability.

Distributed Key Management Schemes — In the distributed key management approaches, different controllers are used to manage key generation, regeneration, and distribution, thus minimizing the risk of failure and allowing for better scalability. In this approach, more entities are allowed to fail before the whole network is affected.

Most proposed key management schemes are distributed schemes. These schemes also fall into deterministic and probabilistic categories, which are discussed in detail in the following subsection.

Key Management Protocols Based on the Probability of Key Sharing

In the remainder of this section, we present the key management protocols based on the probability of key sharing between a pair of sensor nodes. We first discuss deterministic approaches and then discuss probabilistic approaches.

Deterministic Approaches — Zhu *et al.* have proposed a key management protocol, Localized Encryption and Authentication Protocol (LEAP), for sensor networks in [63]. LEAP supports the establishment of four types of keys for each sensor node:
- An individual key shared with the base station (predistributed)
- A group key that is shared by all the nodes in the network (predistributed)
- Pairwise keys shared with immediate neighboring nodes
- A cluster key shared with multiple neighboring nodes

The pairwise keys shared with immediate neighboring nodes are used to protect peer-to-peer communication while the cluster key is used for local broadcast. The pairwise keys can be set up as follows: in the key predistribution stage, each sensor node is loaded with an initial key K_I and each node A generates a master key $K_A = f_{K_I}(A)$, where f is a pseudorandom function. Then, in the neighbor discovery stage, A broadcasts a HELLO message and expects an Acknowledgment from neighboring nodes, e.g., node B:

$$A \to {*}: A$$
$$B \to A: B, MAC(K_B, A|B)$$

Node A computes its pairwise key with B, $K_{AB} = f_{K_B}(A)$. Node B knows A, K_B and can also compute K_{AB} in the same way. Then, K_{AB} serves as their pairwise key.

Cluster key establishment follows the pairwise key establishment phase. Suppose node A wants to establish a cluster key with all its immediate neighbors $B_1, B_2, \ldots, B_m$. Node A first generates a random key K_c^A, then encrypts this key with the pairwise key shared with each neighbor, and finally transmits the encrypted key to each neighbor B_i where $1 \le m$:

$$A \to B_i: (K_c^A)K_{ABi}$$

LEAP uses unicast for key exchange. Notice that most of the proposed security protocols were based on point-to-point handshaking procedures to negotiate session keys. Lai *et al.* have proposed a BROadcast Session Key (BROSK) negotiation protocol [64]. BROSK assumes a master key is shared by all nodes in the network. To establish a session key K with its neighbors, such as node B, a sensor node A broadcasts a key negotiation message:

$$A \to {*}: ID_A|N_A, MAC(K, ID_A|N_A)$$
$$B \to {*}: ID_B|N_B, MAC(K, ID_B|N_B)$$

A and B will receive the broadcast message. They can verify the message using the master key K and both A and B can calculate the shared session key:

$$K_{AB} = MAC(K, N_A|N_B)$$

BROSK therefore establishes pairwise session keys between every two neighboring nodes. It is both scalable and energy efficient.

Camtepe and Yener have proposed a deterministic key distribution schemes for WSNs using Combinatorial Design Theory [65]. The combinatorial design theory based pairwise key predistribution (CDTKeying) scheme is based on block design techniques in combinatorial design theory. It employs symmetric and generalized quadrangle design techniques. The scheme uses a finite projective plane of order n (for prime power n) to generate a symmetric design with parameters $n^2 + n + 1, n + 1, 1$. The design supports $n^2 + n + 1$ nodes, and uses a key-pool of size $n^2 + n + 1$. It generates $n^2 + n + 1$ key chains of size $n + 1$ where every pair of key chains has exactly one key in common, and every key appears in exactly $n + 1$ key chains. After the deployment, every pair of nodes finds exactly one common key. Thus, the probability of key sharing among a pair of sensor nodes is 1. The disadvantage of this solution is that the parameter n has to be a prime power, thus indicating that not all network sizes can be supported for a fixed key-chain size.

Lee and Stinson have proposed two combinatorial design theory based deterministic schemes: the ID-based one-way function scheme (IOS) and the deterministic multiple space Blom's scheme (DMBS) [66]. They further discussed the use of combinatorial set systems in the design of deterministic key predistribution schemes for WSNs in [67].

Probabilistic Approaches — Most proposed key management schemes in WSNs are probabilistic and distributed schemes.

Eschenauer and Gligor introduced a key predistribution scheme for sensor networks which relies on probabilistic key sharing among the nodes of a random graph in [68]. This scheme consists of three phases: key predistribution, shared-key discovery, and path key establishment. In the key predistribution phase, each sensor is equipped with a key ring held in the memory. The key ring consists of k keys which are randomly drawn from a large pool of P keys. The association information of the key identifiers in the key ring and sensor identifier is also stored at the base station. Further, the authors assumed that each sensor shares a pairwise key with the base station. In the shared key discovery phase, each sensor discovers its neighbors within the wireless communication range with which it shares keys. Two methods to accomplish this are suggested in [68]. The simplest method is for each node to broadcast a list of identifiers of the keys in their key ring in plain text, thus allowing neighboring nodes to check whether they share a key. However, an adversary may observe the key-sharing patterns among sensors in this way. The second method uses the challenge–response technique to hide key-sharing patterns among nodes from an adversary. For every K_i on a key ring, each node could broadcast a list α, $E_{K_i}(\alpha)$, $i = 1, \ldots, k$ where α is a challenge. The decryption of $E_{K_i}(\alpha)$ with the proper key by a recipient would reveal the challenge and establish a shared key with the broadcasting node. This method requires that the challenge α be well known in the sensor network, thus allowing the recipient with the proper key to discover the challenge.

Finally, in the path-key establishment phase, a path-key is assigned for those sensor nodes within wireless communication range and not sharing a key, but connected by two or more links at the end of the second phase. If a node is compromised, the base station can send a message to all other sensors to revoke the compromised node's key ring. Rekeying follows the same procedure as revocation. The messages from the base station are signed by the pairwise key shared by the base station and sensor nodes, and thus it is ensured that no adversary can forge a base station. If a node is compromised, the attacker has a probability of approximately k/P to be able to successfully attack any link. Because $k << P$, it only affects a small number of sensor nodes.

Inspired by the work in [68], which we call the basic random key scheme in the following section, additional random key predistribution schemes have been proposed in [69–74].

In the basic random key scheme, any two neighboring nodes need to find a single common key from their key rings in order to establish a secure link in the key setup phase. However, Chan *et al.* observed that increasing the amount of key overlap in the key ring can increase the resilience of the network against node capture [69]. Thus, they proposed a q-composite keying scheme, in which it is required that at least q common keys be shared in the key setup phase in order to build a secure link between any two neighboring nodes. Further, a key update phase was introduced to enhance the basic random key scheme. Suppose A has a secure link to B after the key setup phase and the secure key k is from the key pool P. Because k may be residing in the key-ring memory of some other nodes in the network, the security of the link between A and B is jeopardized if any of those nodes are captured. Thus, it is better to update the communication key between A and B instead of using a key in the key pool. To address this problem, they presented a multipath key reinforcement for the key update. Assume there are j disjoint paths between A and B. A generates j random values $v_1, v_2, \ldots, v_j$ and then routes each random value along a different path to B. When B has received all j keys, the new link key can be computed by both A and B as follows:

$$k' = k \oplus v_1 \oplus v_2 \oplus \ldots \oplus v_j$$

The adversary has to eavesdrop on all j paths if he/she wants to reconstruct the communication key. This security enhancement comes at the cost of more communication overhead needed to find multiple disjoint paths. Further, Chan *et al.* also developed a random-pairwise keys scheme for node-to-node authentication.

Blundo *et al.* presented a polynomial-based key predistribution protocol for group key predistribution [75] which can also be adapted to sensor networks. The key setup server randomly generates a bivariate t-degree polynomial $f(x, y) = \Sigma^t_{i,j=0} a_{ij}x^iy^j$ over a finite field F_q where q is a prime number that is large enough to accommodate a cryptographic key such that it has the property of $f(x, y) = f(y, x)$. For each sensor i, the setup server computes a polynomial share of $f(x, y)$, that is, $f(i, y)$. For any two sensor nodes i and j, node i can compute the common key $f(i, j)$ by evaluating $f(i, y)$ at point j, and node j can compute the same key $f(j, i) = f(i, j)$ by evaluating $f(j, y)$ at point i. In this approach, each sensor node i needs to store a t-degree polynomial $f(i, x)$, which occupies $(t + 1) \log q$ storage space. This scheme is unconditionally secure and t-collusion resistant. However, the storage cost for a polynomial share is exponential in terms of the group size, making it prohibitive in sensor networks.

Inspired by the work of [68, 69, 75], Liu and Ning proposed a polynomial pool-based key predistribution scheme in [70], which also includes three phases: setup, direct key establishment, and path key establishment. In the setup phase, the setup server randomly generates a set F of bivariate t-degree polynomials over the finite field F_q. For each sensor node, the setup server picks a subset of polynomials $F_i \subseteq F$ and assigns the polynomial shares of these polynomials to node i. In the direct key establishment stage, the sensor nodes find a shared polynomial with other sensor nodes and then establish a pairwise key using the polynomial-based key predistribution scheme discussed in [75]. The path key establishment phase is similar to that in the basic random key scheme. Further, the

	Protocol	Ref.	Theory	Master key	Pairwise key	Path key	Cluster key	Scala-bility	Resiliency	Process. load	Comm. load	Storage load
I	All pairwise	—	n/a	n/a	Yes	No	No	Low	Low	Low	Low	High
	LEAP	[63]	n/a	Yes	Yes	Yes	Yes	Good	Low	Low	Low	Low
	BROSK	[64]	n/a	Yes	Yes	No	No	Good	Low	Low	Low	Low
	LKHW	[62]	LKH	Yes	Yes	No	Yes	Limited	Low	Low	Low	Low
	CDTKeying	[65]	Combinatorial	n/a	Yes	No	No	Good	Good	Medium	Medium	High
	IOS & DMBS	[66]	Combinatorial	n/a	Yes	No	No	Good	Good	Medium	Medium	High
II	Basic random key	[68]	Random graph	n/a	Yes	Yes	No	Good	Good	Medium	Medium	High
	q-composite	[69]	Random graph	n/a	Yes	Yes	No	Good	Good	Medium	Medium	High
	Polynomial based	[70]	t-degree polynomial & random graph	n/a	Yes	Yes	No	Good	Good	Medium	Medium	High
	Blom based	[72]	Blom's method & random graph	n/a	Yes	Yes	No	Good	Good	Medium	Medium	High
	Deployment knowledge based	[73]	Random graph	n/a	Yes	Yes	No	Good	Good	Medium	Medium	Medium
	Cluster key grouping	[74]	Random graph	n/a	Yes	Yes	No	Good	Good	Medium	Medium	High
	Location based	[77]	Random graph	n/a	Yes	Yes	No	Good	Good	Medium	Medium	Medium

Note: Category I denotes deterministic approaches and category II denotes probabilistic approaches. Master key is the key shared by all the nodes in the network. Pairwise key is the key shared between two neighboring nodes. Path key denotes the key shared between any two nodes which need exchange data but does not share a pairwise key. Cluster key denotes the common key shared by all cluster members.

■ Table 7. *Classification and comparison of key management protocols in WSNs.*

proposed framework allows for the study of multiple instantiations of possible pairwise key establishment schemes. Two of the possible instantiations, the key predistribution scheme based on random subset assignment and the grid-based key predistribution scheme, are also presented and analyzed in the article.

Similar to [70], Du *et al.* presented another pairwise key predistribution scheme in [72] which uses Blom's method [76]. The key difference between [70] and [72] is that the scheme in [70] is based on a set of bivariate t-degree polynomials while Du's scheme is based on Blom's method. The proposed scheme allows any pair of nodes in a network to be able to find a pairwise secret key. As long as no more than λ nodes are compromised, the network is perfectly secure (which is called the λ-secure property). To use Blom's method, during the predeployment phase, the base station first constructs a $(\lambda + 1) \times N$ matrix G over a finite field $GF(q)$ where N is the size of the network and G is considered to be public information. Then the base station creates a random $(\lambda + 1) \times (\lambda + 1)$ symmetric matrix D over $GF(q)$, and computes a $N \times (\lambda + 1)$ matrix $A = (D \cdot G)^T$ where $(D \cdot G)^T$ is the transpose of $D \cdot G$. Matrix D needs to be kept secret, and should not be disclosed to adversaries. It is easy to verify that $A \cdot G$ is a symmetric matrix.

$$A \cdot G = (D \cdot G)^T \cdot G = G^T \cdot D^T \cdot G = G^T \cdot D \cdot G = (A \cdot G)^T$$

Thus, we know that $K_{ij} = K_{ji}$. The idea is to use K_{ij} (or K_{ji}) as the pairwise key between nodes i and j. To carry out the above computation, in the predistribution phase, for any sensor $k = 1, \ldots, N$:

- Store the k^{th} row of matrix A at node k
- Store the k^{th} column of matrix G at node k

Therefore, when nodes i and j need to find the pairwise key between them, they first exchange their columns of G, and then respectively compute K_{ij} and K_{ji} using their private rows of A.

In the proposed scheme in [72], each sensor node is loaded with G and τ distinct D matrices drawn from a large pool of ω symmetric matrices $D_1, \ldots, D_\omega$ of size $(\lambda + 1) \times (\lambda + 1)$. For each D_i, calculate the matrix $A_i = (D_i \cdot G)^T$ and store the j^{th} row of A_i at this node. After deployment, each node needs to discover whether it shares any space with neighbors. If they find out that they have a common space, the nodes can follow Blom's method to build a pairwise key. The scheme in [72] is scalable and flexible. Moreover, it is substantially more resilient against node capture as compared to [70].

Hwang *et al.* extended the basic random key scheme and proposed a cluster key grouping scheme [74]. They further analyzed the tradeoffs involved between energy, memory, and security robustness.

Notice that location information helps to avoid unnecessary key assignments and thus improve the performance of sensor networks, for example, connectivity, memory usage, and network resilience against node capture. Taking this into account, two random key predistribution schemes were proposed in [73, 77]. Although the presented schemes show improved performance, the deployment information (e.g., location) is required when sensors are deployed.

The abovementioned schemes are classified and compared in Table 7.

Open Research Issues

Although some key management protocols have been proposed for sensor networks, the design of key management protocols is still largely open to research. Open research issues include the following:

- The proposed key management protocols discussed in this section employ different strategies on the trade-off between memory, processing and communication overhead. These protocols could be improved and new key management protocols need to be designed.
- All key management protocols discussed in literature so far are based on symmetric key cryptography. Recent progress in public key cryptography has shown that public key cryptography may be suitable for sensor networks. Key management schemes based on public key cryptography need to be designed.
- Current proposed key management schemes assume that the base station is trustworthy. However, there may be situations (e.g., in the battlefield) where the security of a base station needs to be considered. New schemes need to be designed considering the security of base stations.

Secure Routing Protocols

Many routing protocols have been specifically designed for WSNs. These routing protocols can be divided into three categories according to the network structure: flat-based routing, hierarchical-based routing, and location-based routing [78]. In flat-based routing, all nodes are typically assigned equal roles or functionality. In hierarchical-based routing, nodes play different roles in the network. In location-based routing, sensor node positions are used to route data in the network. Although many sensor network routing protocols have been proposed in literature, few of them have been designed with security as a goal. Lacking security services in the routing protocols, WSNs are vulnerable to many kinds of attacks.

Most network layer attacks against sensor networks fall into one of the categories described above, namely:

- Spoofed, altered, or replayed routing information
- Selective forwarding
- Sinkhole
- Sybil
- Wormholes
- Hello flood attacks
- Acknowledgment spoofing

These attacks may be applied to compromise the routing protocols in a sensor network. For example, directed diffusion is a flat-based routing algorithm for drawing information from a sensor network [79]. In directed diffusion, sensors measure events and create gradients of information in their respective neighboring nodes. The base station requests data by broadcasting interest which describes a task to be conducted by the network. The interest is diffused through the network hop by hop, and broadcasted by each node to its neighbors. As the interest is propagated throughout the network, gradients are setup to draw data satisfying the query towards the requesting node. Each sensor that receives the interest sets up a gradient toward the sensor nodes from which it received the interest. This process continues until gradients are setup from the sources back to the base station. Interests initially specify a low rate of data flow, but once a base station starts receiving events it will reinforce one or more neighboring nodes in order to request higher data rate events. This process proceeds recursively until it reaches the nodes generating events, causing them to generate events at a higher data rate. Paths may also be negatively reinforced. Directed diffusion is vulnerable to many kinds of attacks if authentication is not included in the protocol [26]. For example, it is easy for an adversary to add himself/herself onto the path taken by a flow of events as described in the following:

- The adversary can influence the path by spoofing positive reinforcements. After receiving and rebroadcasting an interest, an adversary could strongly reinforce the nodes to which the interest was sent while spoofing high-rate, low-latency events to the nodes from which the interest was received.
- The adversary can replay the interests intercepted from a legitimate base station and list himself/herself as a base station. All events satisfying the interest will then be sent to both the adversary and the legitimate base station.

By using the attacks above, the adversary can add himself/herself onto the path and thus gain full control of the flow. The adversary can eavesdrop, modify, and selectively forward packets of his/her choosing. He/she can drop all forwarded packets and act as a sinkhole. Further, a laptop-class adversary can exert great influence on the topology by using a wormhole attack. The adversary creates a tunnel between a node located near a base station and a node located close to where events are likely to be generated. By spoofing positive or negative reinforcements, the adversary can push data flows away from the base station and towards the nodes selected by the adversary.

Hierarchical and location based routing protocols not incorporating security services are also vulnerable to many attacks [26]. For example, location-based routing protocols such as Geographic and Energy Aware Routing (GEAR) [80] require location information to be exchanged between neighbors. However, location information can be misrepresented. Regardless of the adversary's actual location, he/she may advertise false position data to place himself/herself on the path of a known flow. Once on that path, the adversary can mount selective forwarding and Sybil attacks in the data flows. Simulations in [81] found that such attacks have great influence on the overall ratio of successfully delivered messages in the network.

Secure routing in ad hoc networks is similar to that in sensor networks and has been well studied in the literature [14]. However, the defense mechanisms developed for ad hoc networks cannot be directly applied to sensor networks because of the differences between sensor and ad hoc networks discussed earlier.

Ideally, a secure routing protocol should guarantee the integrity, authentication, and availability of messages in the presence of adversaries of arbitrary power. In the presence of only outsider adversaries, it is conceivable to achieve these idealized goals. However, in the presence of compromised nodes or insider adversaries, especially those with laptop class capabilities, it is most likely that some if not all of these goals are not fully attainable. In this situation, the best we can hope for is graceful degradation instead of a complete compromise of the network. To achieve the above goal requires that a routing protocol degrades no faster than a rate approximately proportional to the ratio of compromised nodes to total nodes in the network [26].

A secure routing protocol depends on an appropriate key management scheme in a WSN, which has been discussed earlier. Before a routing protocol starts, sensor nodes should have been loaded with proper keys (e.g., the key for confidentiality, authentication, etc.). One of the fundamental security services in sensor networks is broadcast authentication, which enables the base station to broadcast authenticated data to the entire sensor network. In this section, we first discuss the broadcast authentication problem and then review several

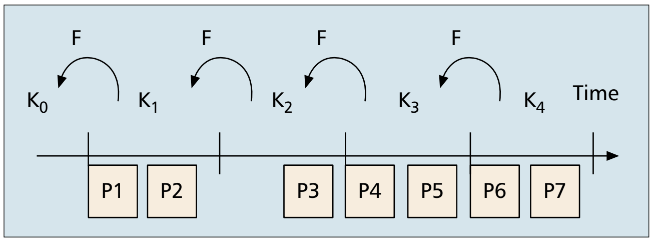

Figure 3. *Using a time-released key chain for source authentication (Source: [8]).*

secure routing schemes.

BROADCAST AUTHENTICATION

Previous proposals for authenticated broadcast are impractical in WSNs for the following reasons:

- Most proposals rely on public key cryptography for the authentication. However, public key cryptography is impractical for WSNs;
- Even one-time signature schemes that are based on symmetric key cryptography have too much overhead.

µ*TESLA* [10] and its extensions [82, 83] have been proposed to provide broadcast authentication for sensor networks.

µ*TESLA* is an authenticated broadcast protocol which was proposed by Perrig *et al.* for the SPINS protocol [8]. µ*TESLA* introduces asymmetry through a delayed disclosure of symmetric keys resulting in an efficient broadcast authentication scheme. µ*TESLA* requires that the base station and nodes be loosely time synchronized, and that each node knows an upper bound on the maximum synchronization error.

To send an authenticated packet, the base station simply computes a MAC on the packet with a key that is secret at that point in time. When a node gets a packet, it can verify that the corresponding MAC key was not yet disclosed by the base station. Since a receiving node is assured that the MAC key is known only by the base station, the receiving node is assured that no adversary could have altered the packet in transit. The node stores the packet in a buffer. At the time of key disclosure, the base station broadcasts the verification key to all receivers. When a node receives the disclosed key, it can easily verify the correctness of the key. If the key is correct, the node can now use it to authenticate the packet stored in its buffer.

Each MAC key is a key from the key chain, generated by a public one-way function F. To generate the one-way key chain, the sender chooses the last key K_n from the chain, and repeatedly applies F to compute all other keys: $K_i = F(K_{i+1})$.

Figure 3 shows an example of µ*TESLA*. The receiver node is loosely time-synchronized and knows K_0 in an authenticated way. Packets P_1 and P_2 sent in interval 1 contain a MAC with key K_1. Packet P_3 has a MAC using key K_2. If P_4, P_5, and P_6 are all lost, as well as the packet that disclosed key K_1, the receiver cannot authenticate P_1, P_2, and P_3. In interval 4 the base station broadcasts key K_2, which the nodes authenticate by verifying $K_0 = F(F(K_2))$, and hence also know $K_1 = F(K_2)$, so that they can authenticate packets P_1, P_2 with K_1, and P_3 with K_2.

SPINS limits the broadcasting capability to only the base station. If a node wants to broadcast authenticated data, the node has to broadcast the data through the base station. The data is first sent to the base station in an authenticated way. It is then broadcasted by the base station.

To bootstrap a new receiver, µ*TESLA* depends on a point-to-point authentication mechanism in which a receiver sends a request message to the base station and the base station replies with a message containing all the necessary parameters. Notice that µ*TESLA* requires that the base station unicast initial parameters to individual sensor nodes, thus incurring a long delay to boot up a large-scale sensor network. Liu and Ning proposed a multilevel key chain scheme for broadcast authentication to overcome this deficiency in [82, 83].

The basic idea in [82, 83] is to predetermine and broadcast the initial parameters required by µ*TESLA* instead of using unicast-based message transmission. The simplest way is to predistribute the µ*TESLA* parameters with a master key during the initialization of the sensor nodes. As a result, all sensor nodes have the key chain commitments and other necessary parameters once they are initialized, and are ready to use µ*TESLA* as long as the starting time has passed. Furthermore, Liu and Ning introduced a multilevel key chain scheme, in which the higher-level key chains are used to authenticate the commitments of lower-level ones. However, the multilevel key chain scheme suffers from possible DoS attacks during the commitment distribution stage. Further, none of the µ*TESLA* or multilevel key chain schemes is scalable in terms of the number of senders. In [84], a practical broadcast authentication protocol was proposed to support a potentially large number of broadcast senders using µ*TESLA* as a building block.

µ*TESLA* provides broadcast authentication for base stations but is not suitable for local broadcast authentication. This is because µ*TESLA* does not provide immediate authentication. For every received packet, a node has to wait for one µ*TESLA* interval to receive the MAC key used in computing the MAC for the packet. As a result, if µ*TESLA* is used for local broadcast authentication, a message traversing l hops will take at least l µ*TESLA* intervals to arrive at the destination. In addition, a sensor node has to buffer all the unverified packets. Both the latency and the storage requirements limit the scheme for authenticating infrequent messages broadcast by the base station. Zhu *et al.* proposed a one-way key chain scheme for one-hop broadcast authentication in LEAP [63]. In this scheme, every node generates a one-way key chain of certain length and then transmits the commitment (i.e., the first key) of the key chain to each neighbor, encrypted with their pairwise shared key. Whenever a node has a message to send, it attaches to the message the next authenticated key in the key chain. The authenticated keys are disclosed in an order that is reverse to their generation. A receiving neighbor can verify the message based on the commitment or an authenticated key it received from the sending node more recently.

SECURE ROUTING

The goal of a secure routing protocol is to ensure the integrity, authentication, and availability of messages. The proposed secure routing protocols for WSNs in the literature are based on symmetric key cryptography, except the work in [85], which is based on public key cryptography.

SPINS is a suite of security protocols optimized for sensor networks [8]. SPINS includes two building blocks: SNEP and µ*TESLA*. SNEP provides data confidentiality, two-party data authentication, and data freshness for peer-to-peer communication (node to base station). µ*TESLA* provides authenticated broadcast as discussed before. We discuss SNEP in this subsection.

SPINS assumes that each node is predistributed with a master key K which is shared with the base station at creation time. All other keys, including a key K_{encr} for encryption, a key K_{mac} for MAC generation, and a key K_{rand} for random number generation, are derived from the master key using a strong one-way function. SPINS uses RC5 for confidentiality.

	Reference	Routing	Confidentiality	P2P Authentication	Broadcast authentication	Integrity	Scalability
SNEP	[8]	Flat	Yes	Yes	No	Yes	Good
LKHW	[62]	Flat	Yes	No	No	No	Limited
µ*TESLA*	[8]	Flat/Hierarchy	No	No	Yes	Yes	Medium
Multi level key chains	[82]	Flat/Hierarchy	No	No	Yes	Yes	Good
LEAP	[63]	Hierarchy	Yes	Yes	Yes	Yes	Medium

Table 8. *Comparison of secure routing protocols.*

If A wants to send a message to base station B, the complete message that A sends to B is

$$A \rightarrow B : D_{\langle K_{encr,C} \rangle}, MAC(K_{mac}, C|D)_{\langle K_{encr,C} \rangle}$$

while D is the transmitted data and C is a shared counter between the sender and the receiver for the block cipher in counter mode. The counter C is incremented after each message is sent and received in both the sender and receiver sides. SNEP also provides a counter exchange protocol to synchronize the counter value in both sides.

SNEP offers the following properties: semantic security, data authentication, replay protection, weak freshness, and low communication overhead. SPINS identifies two types of freshness: weak freshness and strong freshness. Weak freshness provides partial message ordering and carries no delay information while strong freshness provides a total order on a request–response pair and allows for delay estimation.

- Semantic security: The counter value is incremented after each message and thus the same message is encrypted differently each time.
- Data authentication: A receiver can be assured that the message originated from the claimed sender if the MAC verifies correctly.
- Replay protection: The counter value in the MAC prevents replaying old message.
- Weak freshness: The counter also maintains a message ordering in the receiver side and yields weak freshness. SNEP provides weak data freshness only because there is no absolute assurance to node A that a message was created by node B in response to an event in node A.
- Low communication overhead: The counter state is kept at each end point and does not need to be sent in each message.

The directed diffusion routing protocol was proposed by Intanagonwiwat *et al.* without considering security issues [79]. Pietro *et al.* proposed an extension of the directed diffusion protocol which provides secure multicasting in [62]. The extended scheme, Logical Key Hierarchy for WSNs (LKHW), provides robustness in routing and security and supports both backward and forward secrecy for sensor join and leave operations. However, it does not provide data authentication.

Inspired by the work on public key cryptography [6, 7, 33, 43], Du *et al.* investigated the public key authentication problem [85]. The use of public key cryptography eases many problems in secure routing, for example, authentication and integrity. However, before a node A uses the public key from another node B, A must verify that the public key is actually B's (i.e., A must authenticate B's public key); otherwise, man-in-the-middle attacks are possible. In general networks, public key authentication involves a signature verification on a certificate signed by a trusted third party Certificate Authority (CA) [86]. However, the signature verification operations are still too expensive for sensor nodes, as depicted in Table 3 and 4. Du *et al.* proposed an efficient alternative that uses only a one-way hash function for the public key authentication. The proposed scheme can be divided into two stages. In the predistribution stage, a Merkle tree R is constructed with each leaf L_i corresponding to a sensor node (more information on Merkle trees is given below). Let pk_i represent node i's public key, V be an internal tree node, and V_{left} and V_{right} be V's two children. The value of an internal tree node is denoted by ϕ. The Merkle tree can then be constructed as follows:

$$\phi(L_i) = h(id_i, pk_i), \text{ for } i = 1, \ldots, N$$

$$\phi(V) = h(\phi(V_{left}) \,||\, \phi(V_{right}))$$

where "||" represents the concatenation of two strings and h is a one-way hash function such as MD5 or SHA-1. Let R be the root of the tree. Each sensor node v needs to store the root value $\phi(R)$ and the sibling node values $\lambda_1, \ldots, \lambda_H$ along the path from v to R. If node A wants to authenticate B's public key, B sends its public key pk, along with the value of $\lambda_1, \ldots, \lambda_H$ to node A. Then, A can use the same procedure to reconstruct the Merkle tree R' and calculate the root value $\phi(R')$. A will trust B to be authentic if $\phi(R') = \phi(R)$. A sensor node only needs $H + 1$ storage units for the extra hash values. Based on this scheme, Du *et al.* further extended the idea to reduce the height of the Merkle tree in order to improve the communication overhead of the scheme. The proposed scheme is more efficient than signature verification on certificates. However, the scheme requires that some hash values be distributed in a predistribution stage. This results in some scalability issues when new sensors are added to an existing WSN.

The above discussion is summarized in Table 8.

Open Research Issues

The development of secure routing protocols is challenging because sensor nodes are prone to failures and the topology of a sensor network changes frequently due to node failures and possible mobility. Key open research issues include the following:

- The proposed secure routing protocols for WSNs focus on static sensor networks only, ignoring mobility. Secure routing protocols for mobile sensor networks need to be investigated.
- Current broadcast authentication schemes such as µ*TESLA* and its extensions require the sensor network to be loosely time-synchronized. This requirement is often hard to meet and new techniques that do not require

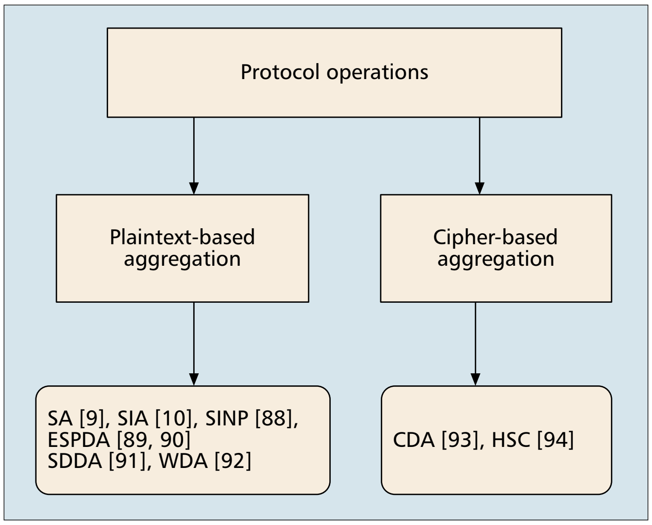
Protocol operations
Plaintext-based aggregation
Cipher-based aggregation
SA [9], SIA [10], SINP [88], ESPDA [89, 90] SDDA [91], WDA [92]
CDA [93], HSC [94]

Figure 4. *Secure data aggregation in WSNs: a taxonomy.*

time synchronization are desirable.

- New schemes with higher scalability and efficiency need to be developed for the authenticated broadcast protocols. The recent progress on public key cryptography may facilitate the design of authenticated broadcast protocols.
- Quality of Service (QoS) in WSNs needs to be evaluated with the addition of secure routing services.

Secure Data Aggregation

Data communication constitutes an important share of the total energy consumption of the sensor network. The simulation in [8] shows that data transmission accounts for 71 percent of the energy cost of computation and communication for the SNEP protocol. Thus, data aggregation can greatly help conserve the scarce energy resources by eliminating redundant data.

Data aggregation (fusion) protocols aim at eliminating redundant data transmitted across the network and are essential for energy-constrained WSNs. Traditional data aggregation techniques include simple types of queries such as SUM, COUNT, AVERAGE, and MIN/MAX. Some researchers also extend data aggregation to median, the most frequent (consensus) data values, a histogram of the data distribution, and range queries [87]. Data aggregation can be divided into two stages: detection and data fusion.

In a WSN, there are usually certain nodes, called aggregators, helping to aggregate information requested by queries. When an aggregator node is compromised, it is easy for the adversary to inject false data into sensor networks. Thus, the aggregators are vulnerable to attack. Another possible attack is to compromise a sensor node and inject forged data through a sensor node. Without authentication, the attackers can fool the aggregators into reporting false data to the base station. Secure data aggregation requires authentication, confidentiality, and integrity. Moreover, secure data aggregation also requires the cooperation of sensor nodes to identify the compromised sensors.

However, requirements for confidentiality and data aggregation are at odds with each other. Confidentiality requires the data to be transmitted in encrypted text while data aggregation is usually based on plain text. A straightforward method is to invoke end-to-end encryption and decryption before evoking data aggregation. However, the tradeoff is that the end-to-end encryption and decryption operations consume more energy, which is of great concern in WSNs. An alternative way is to provide data aggregation on concealed data, which requires a particular class of encryption transformation. However, this method usually lowers the security level.

Figure 4 shows a taxonomy of secure data aggregation protocols in WSNs. According to the protocol operation, secure data aggregation can be classified into two categories: plaintext based and cipher based. This section reviews the techniques for secure data aggregation.

Plain-text Based Secure Data Aggregation

Hu and Evans proposed a secure aggregation (SA) protocol for WSNs that is resilient to both intruder devices and single device key compromises [9]. However, the protocol may be vulnerable if a parent and a child node in the hierarchy are compromised.

Przydatek *et al.* proposed a secure information aggregation (SIA) framework for sensor networks [10]. The framework consists of three node categories: a home server, base station(s), and sensor nodes. A base station is a resources-enhanced node which is used as intermediary between the home server and the sensor nodes, and is also the candidate to perform the aggregation task. SIA assumes that each sensor has a unique identifier and shares a separate secret cryptographic key with both the home server and the aggregator. The keys enable message authentication and encryption if data confidentiality is required. Moreover, it further assumes that the home server and base station can use a mechanism, such as μ*TESLA*, to broadcast authentic messages. The proposed solution consists of three parts: computation of the result, committing to the collected data, and reporting the aggregation result while proving the correctness of the result.

In the first part, the aggregator collects the data from sensors and locally computes the aggregation result. The aggregator can verify the authenticity of each sensor reading.

In the second part, the aggregator commits to the collected data. The commitment to the input data ensures that the aggregator uses the data provided by the sensors, and that the statement to be verified by the home server about the correctness of computed results is meaningful. One efficient way of committing to the data is a Merkle hash-tree construction. In this construction, all the data collected from the sensors is placed at the leaves of the tree. The aggregator then computes a binary hash tree starting from the leaf nodes. Each internal node in the hash tree is computed as the hash value of the concatenation of its two child nodes. The root of the tree is called the commitment of the collected data. As the hash function in use is collision resistant, once the aggregator commits to the collected values it cannot change any of them. Figure 5 shows an example of a Merkle hash tree.

In the third part, the aggregator and the home server engage in a protocol in which the aggregator communicates the aggregation result and the commitment to the server while proving to the server that the reported results are correct using interactive proof protocols. Moreover, the authors also presented efficient protocols for secure computation of the median and average of the measurements, for the estimation of the network size, and for finding the minimum and maximum sensor reading.

Deng *et al.* proposed a collection of mechanisms for securing in-network processing (SINP) for WSNs [88]. Security mechanisms were proposed to address the downstream requirement that sensor nodes authenticate commands disseminated from parent aggregators and the upstream requirement that aggregators authenticate data produced by sensors before aggregating that data. In the downstream stage, two techniques are involved: one-way functions and μ*TESLA*. The

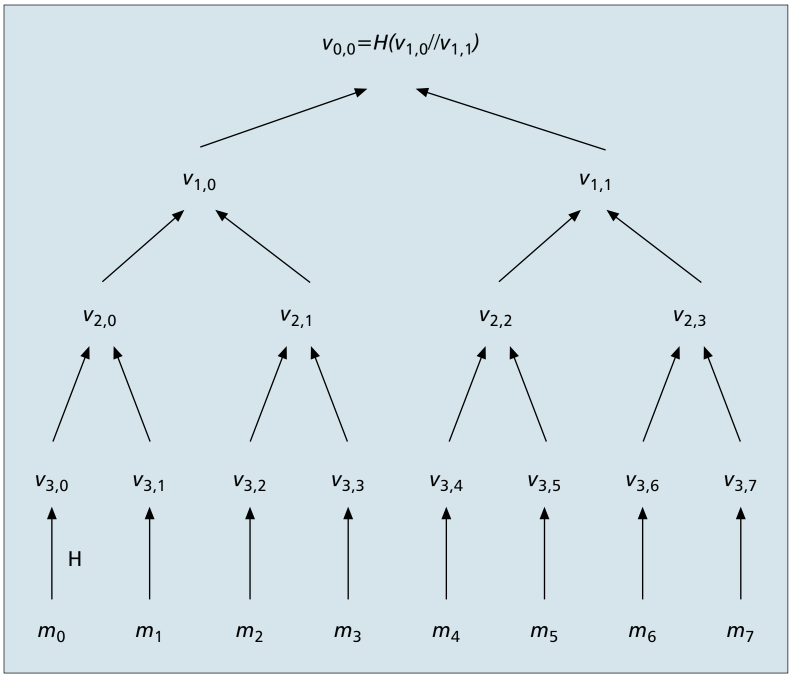

Figure 5. *Merkle hash tree used to commit to a set of values. The aggregator constructs the Merkle hash tree over the sensor measurement* $m_0, \ldots, m_7$. *To construct the Merkle hash tree, the aggregator first hashes the measurements with a cryptographic hash function, e.g.,* $v_{3,0} = H(m_0)$, *assuming that the size of the hash is smaller than the size of the data. Then, each internal value of the Merkle hash tree is derived from its two child nodes:* $vi,j = H(v_{i+1,2j} \,||\, v_{i+1,2j+1})$. *The Merkle hash tree is a commitment to all the leaf nodes. Once the aggregator commits to the collected values, it cannot change any of the collected data. A verifier can authenticate any value by verifying that the leaf value is used to derive the root node given the authentic root node* $v_{0,0}$. *For example, to authenticate the measurement* m_5, *the aggregator sends* m_5 *along with* $v_{3,4}$, $v_{2,3}$, $v_{1,0}$, *and* m_5 *is authentic if the following equality holds:* $v_{0,0} = H(v_{1,0} \,||\, H(H(v_{3,4} \,||\, H(m_5)) \,||\, v_{2,3}))$. *(Source:[10]).*

upstream stage requires that a pairwise key be shared between an aggregator and its sensor nodes.

Çam *et al.* proposed an energy-efficient secure pattern-based data aggregation (ESPDA) protocol for wireless sensor networks in [89, 90]. ESPDA is applicable for hierarchy-based sensor networks. In ESPDA, a cluster head first requests sensor nodes to send the corresponding pattern code for the sensed data. If multiple sensor nodes send the same pattern code to the cluster head, only one of them is permitted to send the data to the cluster head. ESPDA is secure because it does not require encrypted data to be decrypted by cluster-heads in order to perform data aggregation.

Further, the authors introduced a secure differential data aggregation (SDDA) scheme based on pattern codes [91]. SDDA prevents redundant data transmission from sensor nodes by implementing the following schemes: SDDA transmits differential data rather than raw data, SDDA performs data aggregation on pattern codes representing the main characteristics of sensed data, and SDDA employs a sleep protocol to coordinate the activation of sensing units in such a way that only one of the sensor nodes capable of sensing the data is activated at a given time. In the SDDA data transmission scheme, the raw data from sensor nodes is compared to reference data with the difference data being transmitted. The reference data is obtained by taking the average of previously transmitted data.

Du *et al.* proposed a witness-based data aggregation (WDA) scheme for WSNs to assure the validation of the data sent from data fusion nodes to the base station [92]. In order to prove the validity of the fusion result, the fusion node has to provide proofs from several witnesses. A witness is one who also conducts data fusion like a data fusion node, but does not forward its result to the base station. Instead, each witness computes the message authentication code (MAC) of the result and then provides it to the data fusion node who must forward the proofs to the base station.

Wagner studied secure data aggregation in sensor networks and proposed a mathematical framework for formally evaluating their security [93]. In [11, 94], the authors proposed two data fusion schemes for the filtering of injected false data in sensor networks, which will be introduced below.

Cipher-Based Secure Data Aggregation

Two cipher-based secure data aggregation schemes were proposed in [95, 96], works which are based on a particular encryption transformation called a privacy homomorphism (PH). This is an encryption transformation that allows direct computation on encrypted data. Let Q and R denote two rings, and let + denote addition and × denote multiplication on both. Let K be the key space. We denote an encryption transformation E: $K \times Q \rightarrow R$ and the corresponding decryption transformation $D : K \times R \rightarrow Q$. Given $a, b \in Q$, and $k \in K$, we term

$$a + b = D_k(E_k(a) + E_k(b))$$

additively homomorphic and

$$a \times b = D_k(E_k(a) \times E_k(b))$$

multiplicative homomorphic [12].

The proposed scheme in [95], Concealed Data Aggregation (CDA), is based on the PH proposed in [97], although the study in [98] has shown that the proposed PH in [97] is unsecure against chosen plain text attacks for some parameter settings. In [95] the authors claimed that, for the WSN data aggregation scenario, the security level is still adequate and the proposed PH method in [97] can be employed for encryption. CDA can be used to calculate SUM and AVERAGE in a hierarchical WSN. To calculate AVERAGE, an aggregator needs to know the number of sensor nodes n.

Castelluccia *et al.* proposed a simple and provable secure additively homomorphic stream cipher (HSC) that allows for the efficient aggregation of encrypted data [96]. The new cipher uses modular addition and is therefore very well suited for CPU-constrained devices such as those in WSNs. The aggregation based on this cipher can be used to efficiently compute statistical values such as the mean, variance, and standard deviation of sensed data while achieving significant bandwidth gain.

Open Research Issues

Data aggregation is essential for WSNs and security is absolutely necessary to defend against compromised sensor nodes. Open research issues include the following:

- Several secure data aggregation protocols have been proposed; however, no comparisons have been conducted on these protocols. Further evaluation and comparison are

desirable to learn the performance of these protocols. The performance matrices might include security, processing overhead, communication overhead, energy consumption, and data compression rate.
- New data aggregation protocols need to be developed to address higher scalability and higher reliability against aggregator and sensor node cheating.

INTRUSION DETECTION

The security mechanisms implemented in secure routing protocols and secure data aggregation protocols are configured ahead of time in order to inhibit an attacker from breaking the security of the network. These security mechanisms alone cannot ensure perfect security of a WSN. Since sensor nodes can be compromised, it is easy for an adversary to inject false data into a WSN through the compromised nodes. Authentication and data encryption are not enough for ensuring data security. Another approach to protect WSNs involves mechanisms for detecting and reacting to intrusions.

An Intrusion Detection System (IDS) monitors a host or network for suspicious activity patterns outside normal and expected behavior [5]. It is based on the assumption that there exists a noticeable difference in the behaviors of an intruder and a legitimate user in the network such that an IDS can match those preprogrammed or possibly learned rules. Based on the analysis model used for analyzing the audit data to detect intrusions, IDSs in ad hoc networks are classified into rule-based and anomaly-based systems. The rule-based intrusion detection systems are used to detect known patterns of intrusions (e.g., [99] and [100]) while anomaly-based systems are used to detect new or unknown intrusions (e.g., [101] and [102]). A rule-based IDS has a low false-alarm rate when compared to an anomaly-based system, and an anomaly-based IDS has a high intrusion-detection rate in comparison to a rule-based system.

However, WSNs are generally application-specific and lack basic information on topology, normal usage, expected communication patterns, and so forth. It is impractical to preinstall some fixed patterns in sensors before they are deployed. Moreover, due to constraints in sensors, to learn and detect these parameters after deployment is both time and energy consuming. Thus, existing IDSs in ad hoc networks may not be adapted to WSNs. The research on intrusion detection in WSNs is still preliminary. Current research focuses on how to detect and eliminate injected false information. Note that compromised nodes can always inject false information into a sensor network. Thus, cooperation among sensors, especially neighboring nodes, is necessary to decide the validity of a report. In this section, we discuss the intrusion detection techniques in WSNs.

INTRUSION DETECTION IN WSNS

Zhu *et al.* proposed an interleaved hop-by-hop authentication (IHOP) scheme in [11]. IHOP guarantees that the base station will detect any injected false data packets when no more than a certain number t of nodes are compromised. The sensor network is organized in a cluster-based hierarchy. Each cluster head builds a route to the base station and each intermediate node has an upper associate node and a lower associate node that is $t + 1$ hops away. IHOP depends on the use of some sharing keys:
- Every node shares a master secret key with the base station.
- Each node knows its one-hop neighbors and has established a pairwise key with each of them.
- A node can establish a pairwise key with another node that is multiple hops away if needed.

Further, IHOP also assumes that the base station has a mechanism to authenticate broadcast messages (e.g., μ*TESLA)*.

A cluster head collects information from its members and sends a report to the base station only when at least $t + 1$ sensors observe the same result. Meanwhile, a cluster head also collects the message authentication codes (MACs) from detecting nodes. Each detecting node sends two MACs to the cluster head: a MAC using the key shared with the base station, referred to as the individual MAC, and a MAC using the key shared with its upper associate nodes, referred to as the pairwise MAC. The cluster head then compresses the $t + 1$ individual MACs by XORing them to reduces the size of a report. However, the pairwise MACs are not compressed for transmission. If they were, a node relaying the message would not be able to extract the pairwise MACs of interest to it. Thus, a legitimate report includes $t + 1$ pairwise MACs and a compressed MAC for the base station. When an intermediate node receives a report, it verifies the MAC of its lower associate node. If it fails, the report is eliminated. Otherwise, it removes the MAC, generates a new MAC using its upper associate node pairwise key, and appends it to the report.

IHOP ensures that the base station can detect false data packets when no more than t nodes are compromised. However, the article does not show how to select the parameter t for a sensor network.

Ye *et al.* proposed a statistical en-route filtering (SEF) mechanism that can detect and drop false data [94]. SEF uses a similar key assignment scheme as the basic random key scheme presented in [68]. There is a global key pool and each sensor is pre-installed with a partition selected from the pool. When a stimulus occurs in the fields, the sensors detecting this event elect one of the nodes as the center of stimulus (CoS), a node which collects and summarizes the detection results from all detecting nodes and produces a synthesized report on behalf of the group. The CoS generates the report and broadcasts the report to all detecting nodes. If a detecting node agrees with the report, it generates a MAC using a key in its partition and sends the MAC to the CoS. The CoS reports the stimulus to the base station only if it receives adequate MACs. A legitimate report carries multiple MACs and a single compromised node cannot fake all MACs. When an en-route node receives the report, it verifies the correctness of the MACs probabilistically and drops those with invalid MACs immediately. Finally, if a report reaches the base station, the base station checks all the MACs and filters out any remaining false reports that escaped the en-route filtering. When a stimulus appears, multiple nodes that detect it collaborate to process the signal and elect the CoS based on the sensing signal strength. The node with the strongest signal stands out as the CoS. To reduce the communication overhead, SEF further uses a Bloom filter [103] to reduce MAC sizes. SEF is designed to protect against injected false information and cannot defend against selective forwarding attacks.

Deng *et al.* proposed an intrusion-tolerant routing in wireless sensor networks (INSENS) in [104] and they further evaluated its performance in [105]. INSENS is a proactive routing protocol. The sensors collect local topology information and send this information back to the base station. The base station generates a forwarding table based on the collected information and sends the routing table to the corresponding sensors. The base station is the central control point for calculating the routing table, which relieves the computation load of individual sensors. Protecting against intrusions focuses on three attacks: DoS-type attacks, routing attacks, and select forwarding attacks. To protect against DoStype attacks, only the base station is allowed to broadcast to the entire network and individual sensors can only send unicast messages. INSENS requires some broadcast authentication scheme such as μ*TESLA*. Although a compromised node may still alter a valid message and broadcast that message to its neighbors, the damage is restricted to only nearby nodes and the downstream nodes. To protect against routing attacks which propagate erroneous control packets, a symmetric key is chosen for confidentiality and authentication. Further, to protect against select forwarding attacks, data are sent to base stations along two separate paths which are calculated by the base stations in the route discovery step. However, INSENS is built on a table based routing protocol, and as such depends on the base stations to collect all needed topology information to calculate the forwarding table for each individual sensor. Thus, INSENS is not scalable in large sensor networks.

Wang *et al.* proposed a scheme to detect whether a node is faulty or malicious with the collaboration of neighbor nodes [106]. In the proposed scheme, when a node suspects that one of its neighbors is faulty, it sends out messages to request opinions on the behavior of this suspected node from other neighbors of the suspect. After collecting the results, the node analyzes the results to diagnose whether the suspect has a fault. The authors formalized the problem as how to construct a dominating tree to cover all the neighbors of the suspect and further proposed two tree-based propagation collection protocols to construct a dominating tree and collect information via the tree structure.

Open Research Issues

Intrusion detection in WSNs is still largely open to research. Key research issues include the following:

- Due to the constraints in WSNs, intrusion detection has many aspects that are not of concern in other network types. The problem of intrusion detection needs to be well defined in WSNs.
- The proposed IDS protocols in literature focus on filtering injected false information only [11, 94, 104]. These protocols need to be improved so as to address scalability issues.

Security in WSNs: Future Directions

WSNs are promising solutions for many applications and security is often a key concern. Although research efforts have been made with regard to cryptography, key management, secure routing, secure data aggregation, and intrusion detection in WSNs, there are still some challenges to be addressed. First, the selection of the appropriate cryptographic methods depends on the processing capability of sensor nodes, indicating that there is no unified solution for all sensor networks. Instead, the security mechanisms are highly application-specific. Second, sensors are characterized by the constraints on energy, computation capability, memory, and communication bandwidth. The design of security services in WSNs must satisfy these constraints. Third, most of the current protocols assume that the sensor nodes and the base station are stationary. However, there may be situations, such as battlefield environments, where the base station and possibly the sensors need to be mobile. The mobility of sensor nodes has a great influence on sensor network topology and thus raises many issues about secure routing protocols. In particular, we identify some of the future directions in the study of security issues in WSNs as follows.

- Exploit the availability of private key operations on sensor nodes: Recent studies on public key cryptography show that public key operations may be practical in sensor nodes. However, private key operations are still too expensive to accomplish in a sensor node. As public key cryptography can greatly ease the design of security in WSNs, improving the efficiency of private key operations on sensor nodes is highly desirable.
- Secure routing protocols for mobile sensor networks: The mobility of sensor nodes has a great influence on sensor network topology and thus on the routing protocols. Mobility can be at the base station, sensor nodes, or both. Current protocols assume the sensor network is stationary. New secure routing protocols for mobile sensor networks need to be developed.
- Continuous stream security in WSNs: Current work on security in sensor networks focuses on discrete events such as temperature and humidity. Continuous stream events such as video and images are not discussed. Video and image sensors for WSNs might not be widely available now, but will likely be in the future. Substantial differences in authentication and encryption exist between discrete events and continuous events, indicating that there will be distinctions between continuous stream security and the current protocols in WSNs.
- QoS and security: Performance is generally degraded with the addition of security services in WSNs. Current studies on security in WSNs focus on individual topics such as key management, secure routing, secure data aggregation, and intrusion detection. QoS and security services need to be evaluated together in WSNs.

Summary

As WSNs grow in capability and are used more frequently, the need for security in them becomes more apparent. However, the nature of nodes in WSNs gives rise to constraints such as limited energy, processing capability, and storage capacity. These constraints make WSNs very different from traditional ad hoc wireless networks. As such, special protocols and techniques have been developed for use in WSNs.

While existing surveys [12–15] discuss security in wireless networks, none focus specifically on security in WSNs and the constraints unique to them. In this article, we have surveyed the security issues in WSNs starting with the attacks and countermeasures in each network layer followed by the issues and solutions in cryptography, key management, secure routing, secure data aggregation, and, finally, intrusion detection. While the discussed security services certainly add more computation, communication, and storage overhead in WSNs, and thus consume more energy, they are highly desirable and often required in real-world applications.

Acknowledgments

This work is partially supported by NSF Grant no. CCR-0311577. The authors thank the anonymous reviewers for their valuable comments on this manuscript.

References

[1] D. Estrin *et al.*, "Instrumenting the World with Wireless Sensor Networks," *Proc. Int'l. Conf. Acoustics, Speech and Signal Processing*, Salt Lake City, UT, May 2001.

[2] H. Chan and A. Perrig, "Security and Privacy in Sensor Net-

works," *IEEE Comp. Mag.*, Oct. 2003, pp. 103–05.
[3] E. Shi and A. Perrig, "Designing Secure Sensor Networks," *Wireless Commun. Mag.,* vol. 11, no. 6, Dec. 2004 pp. 38–43.
[4] I. F. Akyildiz *et al.*, "A Survey on Sensor Setworks," *IEEE Commun. Mag.*, vol. 40, no. 8, Aug. 2002, pp. 102–114.
[6] N. Gura *et al.*, "Comparing Elliptic Curve Cryptography and RSA on 8-bit CPUs," CHES '04: Proc. Wksp. Cryptographic Hardware and Embedded Systems, Aug. 2004.
[7] A. S. Wander *et al.*, "Energy Analysis of Public-Key Cryptography for Wireless Sensor Networks," *PerCom '05: Proc. 3rd IEEE Int'l. Conf. Pervasive Computing and Commun.,* Mar. 2005.
[8] A. Perrig *et al.*, "SPINS: Security Protocols for Sensor Networks," *Wireless Networks*, vol. 8, no. 5, Sept. 2002, pp. 521–34.
[9] L. Hu and D. Evans, "Secure Aggregation for Wireless Networks," *Wksp. Security and Assurance in Ad Hoc Networks*, 2003.
[10] B. Przydatek, D. Song, and A. Perrig, "SIA: Secure Information Aggregation in Sensor Networks," *SenSys '03: Proc. 1st Int'l. Conf. Embedded Networked Sensor Systems*, New York: ACM Press, 2003, pp. 255–65.
[11] S. Zhu *et al.*, "An Interleaved Hop-by-Hop Authentication Scheme for Filtering of Injected False Data in Sensor Networks," *Proc. IEEE Symp. Security and Privacy*, Oakland, CA, May 2004, pp. 259–71.
[12] R. L. Rivest, L. Adleman, and M. L. Dertouzos, "On Data Banks and Privacy Homomorphisms," in *Foundations of Secure Computation*, New York: Academic, 1978, pp. 169–79.
[13] F. Stajano and R. J. Anderson, "The Resurrecting Duckling: Security Issues for Ad-hoc Wireless Networks," Proc. 7th Int'l. Wksp. Security Protocols, London: Springer-Verlag, 2000, pp. 172–94.
[14] Y.-C. Hu and A. Perrig, "A Survey of Secure Wireless Ad Hoc Routing," *IEEE Security & Privacy Special Issue: Making Wireless Work*, vol. 2, no. 3, May/June 2004, pp. 28–39.
[15] D. Djenouri, L. Khelladi, and N. Badache, "A Survey on Security Issues in Mobile Ad Hoc and Sensor Networks," *IEEE Commun. Surveys and Tutorials*, vol. 7, no. 4, 2005.
[16] P. Levis and D. Culler, "Mate: A Tiny Virtual Machine for Sensor Networks," *ASPLOS-X: Proc. 10th Int'l. Conf. Architectural Support for Programming Languages and Operating Systems*, New York: ACM Press, 2002, pp. 85–95.
[17] EYES project, Mar. 2002–Feb. 2005, http://www.eyes.eu.org
[18] A. Savvides and M. B. Srivastava, "A Distributed Computation Platform for Wireless Embedded Sensing," *ICCD '02: Proc. 2002 IEEE Int'l. Conf. Computer Design: VLSI in Computers and Processors*, Washington, DC: IEEE Computer Society, 2002, p. 220.
[19] R. Kling, "Intel Research Mote," in *Network Embedded Systems Technology*, Winter 2003 Retreat, Jan. 15–17. 2003.
[20] J. Hill *et al.*, "System Architecture Directions for Networked Sensors," *ASPLOSIX: Proc. 9th Int'l. Conf. Architectural Support for Programming Languages and Operating Systems*, New York: ACM Press, 2000, pp. 93–104.
[21] J. Hill *et al.*, "System Architecture Directions for Networked Sensors," *SIGOPS Oper. Syst. Rev.*, vol. 34, no. 5, 2000, pp. 93–104.
[22] S. Slijepcevic *et al.*, "On Communication Security in Wireless Ad-Hoc Sensor Networks," *Proc. 11th IEEE Int'l. Wksp. Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE'02)*, 2002, pp. 139–44.
[23] L. Yuan and G. Qu, "Design Space Exploration for Energy-Efficient Secure Sensor Network," *IEEE Int'l. Conf. Application-Specific Systems, Architectures, and Processors* (*ASAP '02)*, July 2002, pp. 88–100.
[24] J. Deng, R. Han, and S. Mishra, "Enhancing Base Station Security in Wireless Sensor Networks," Department of Computer Science, University of Colorado, Tech. Report CU-CS-951-03, 2003.
[25] B. Deb, S. Bhatnagar, and B. Nath, "Information Assurance in Sensor Networks," *Proc. 2nd ACM Int'l. Conf. Wireless Sensor Networks and Applications* (*WSNA '03)*, New York: ACM Press, 2003, pp. 160–68.
[26] C. Karlof and D. Wagner, "Secure Routing in Wireless Sensor Networks: Attacks and Countermeasures," *Proc. First IEEE Int'l. Wksp. Sensor Network Protocols and Applications*, May 2003, pp. 113–27.
[27] J. Newsome *et al.*, "The Sybil Attack in Sensor Networks: Analysis and Defenses," *IPSN '04: Proc. IEEE Int'l. Conf. Info. Processing in Sensor Networks*, Apr. 2004.
[28] Y.-C. Hu, A. Perrig, and D. B. Johnson, "Packet Leashes: A Defense Against Wormhole Attacks in Wireless Networks," *Proc. IEEE INFOCOM 2003*, Apr. 2003.
[29] W. Diffie and M. E. Hellman, "New Directions in Cryptography," *IEEE Trans. Info. Theory*, vol. 22, no. 6, Nov. 1976, pp. 644–54.
[30] R. L. Rivest, A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems," *Commun. ACM*, vol. 26, no. 1, 1983, pp. 96–99.
[31] D. W. Carman, P. S. Kruus, and B. J. Matt, "Constraints and Approaches for Distributed Sensor Network Security," NAI Labs, Tech. Report 00-010, 2000.
[32] M. Brown *et al.*, "PGP in Constrained Wireless Devices," *Proc. 9th USENIX Security Symp.*, Aug. 2000.
[33] G. Gaubatz, J.-P. Kaps, and B. Sunar, "Public Key Cryptography in Sensor Networks-Revisited," *ESAS '04: 1st European Wksp. Security in Ad-Hoc and Sensor Networks*, 2004.
[34] M. O. Rabin, "Digitalized Signatures and Public-Key Functions as Intractable as Factorization," Tech. Rep., Cambridge, MA, 1979.
[35] J. Hoffstein, J. Pipher, and J. H. Silverman, "Ntru: A Ring-Based Public Key Cryptosystem," *ANTS-III: Proc. 3rd Int'l. Symp. Algorithmic Number Theory*, London: Springer-Verlag, 1998, pp. 267–88.
[36] V. S. Miller, "Use of Elliptic Curves in Cryptography," Lecture notes in computer sciences; 218 on *Advances in Cryptology-CRYPTO 85*, New York: Springer-Verlag, 1986, pp. 417–26.
[37] N. Koblitz, "Elliptic Curve Cryptosystems," *Mathematics of Computation*, vol. 48, 1987, pp. 203–09.
[38] Elliptic Curve Cryptography, SECG Std. SEC1, 2000, available at www.secg.org/collateral/sec1.pdf
[39] B. Kaliski, "TWIRL and RSA Key Size," RSA Laboratories," Tech. Note, May 2003.
[40] Recommended Elliptic Curve Domain Parameters, SECG Std. SEC2, 2000, available at www.secg.org/collateral/sec2.pdf
[41] D. Hankerson, A. Menezes, and S. Vanstone, *Guide to Elliptic Curve Cryptography*, New York: Springer-Verlag, 2004.
[42] A. Freier, P. Karlton, and P. Kocher, "The SSL Protocol, Version 3.0." http://home.netscape.com/eng/ssl3/
[43] R. Watro *et al.*, "TinyPK: Securing Sensor Networks with Public Key Technology," *SASN '04: Proc. 2nd ACM Wksp. Security of Ad Hoc and Sensor Networks*, New York: ACM Press, 2004, pp. 59–64.
[44] A. Liu and P. Ning, "TinyECC: Elliptic Curve Cryptography for sensor networks (version 0.1)," Sept. 2005, available at http://discovery.csc.ncsu.edu/software/TinyECC/
[45] D. J. Malan, M. Welsh, and M. D. Smith, "A Public-Key Infrastructure for Key Distribution in TinyOS based on Elliptic Curve Cryptography," *Proc. 1st IEEE Int'l. Conf. Sensor and Ad Hoc Communications and Networks*, Santa Clara, CA, Oct. 2004.
[46] C. Karlof, N. Sastry, and D. Wagner, "Tinysec: A Link-Layer Security Architecture for Wireless Sensor Networks," *SenSys '04: Proc. 2nd Int'l. Conf. Embedded Networked Sensor Systems*, New York: ACM Press, 2004, pp. 162–75.
[47] U.S. National Institute of Standards and Technology (NIST), "SKIPJACK and KEA Algorithm Specifications," Federal Information Processing Standards Publication 185 (FIPS PUB 185), June 1998.
[48] R. L. Rivest, "The RC5 Encryption Algorithm," *Fast Software Encryption*, B. Preneel (Ed.), Springer, 1995, pp. 86–96.
[49] D. Eastlake III and P. Jones, "US Secure Hash Algorithm 1 (SHA1)," RFC 3174 (Informational), Sept. 2001.
[50] J. Daemen and V. Rijmen, "AES Proposal: Rijndael," *Proc. 1st AES Conf.*, Aug. 1998.
[51] A. J. Menezes, S. A. Vanstone, and P. C. V. Oorschot, *Handbook of Applied Cryptography*, Boca Raton, FL: CRC Press, 1996.
[52] R. L. Rivest, "The MD5 Message-Digest Algorithm," RFC 1321, Apr. 1992.
[53] P. Ganesan *et al.*, "Analyzing and Modeling Encryption Over-

head for Sensor Network Nodes," *WSNA '03: Proc. 2nd ACM Int'l. Conf. Wireless Sensor Networks and Applications*, New York: ACM Press, 2003, pp. 151–59.

[54] Y. W. Law *et al.*, "Assessing Security- Critical Energy-Efficient Sensor Networks," *Proc. 18th IFIP TC11 Int'l. Conf. Info. Security, Security, and Privacy in the Age of Uncertainty (SEC)*, Athens, Greece, May 2003, pp. 459–63.

[55] D. J. Wheeler and R. M. Needham, "TEA, A Tiny Encryption Algorithm," *Proc. Fast Software Encryption: 2nd Int'l. Wksp.*, in *Lecture Notes in Computer Science* (series), B. Preneel (Ed.), vol. 1008, 1994.

[56] R. L. Rivest *et al.*, "The RC6 Block Cipher," submitted to NIST as a candidate for the AES.

[57] M. Matsui, "New Block Encryption Algorithm Misty," *Proc. 4th Int'l. Wksp. Fast Software Encryption*, in LNCS, E. Biham (Ed.), vol. 1267, London: Springer-Verlag, 1997, pp. 54–68.

[58] ETSI/SAGE, "Specification of the 3GPP Confidentiality and Integrity Algorithms Document 2: KASUMI Specification," Dec. 1999.

[59] K. Aoki *et al.*, "Specification of Camellia: A 128-Bit Block Cipher, specification version 2.0," Nippon Telegraph and Telephone Corporation and Mitsubishi Electric Corporation, 2001.

[60] Y. W. Law, J. M. Doumen, and P. H. Hartel, "Benchmarking Block Ciphers for Wireless Sensor Networks (Extended Abstract)," *1st IEEE Int'l. Conf. Mobile Ad-hoc and Sensor Systems*, IEEE Computer Society Press, Oct. 2004.

[61] S. A. Camtepe and B. Yener, "Key Distribution Mechanisms for Wireless Sensor Networks: A Survey," Computer Science Department at RPI, Tech. Rep. TR-05-07, 2005.

[62] R. D. Pietro *et al.*, "LKHW: A Directed Diffusion-Based Secure Multicast Scheme for Wireless Sensor Networks," *ICPPW '03: Proc. 32nd Int'l. Conf. Parallel Processing Wksps.*, IEEE Computer Society Press, 2003, pp. 397–406.

[63] S. Zhu, S. Setia, and S. Jajodia, "LEAP: Efficient Security Mechanisms for Large-Scale Distributed Sensor Networks," *CCS '03: Proc. 10th ACM Conf. Comp. and Commun. Security*, New York: ACM Press, 2003, pp. 62–72.

[64] B. Lai, S. Kim, and I. Verbauwhede, "Scalable Session Key Construction Protocols for Wireless Sensor Networks," *IEEE Wksp. Large Scale Real Time and Embedded Systems*, 2002.

[65] S. A. Cametepe and B. Yener, "Combinatorial Design of Key Distribution Mechanisms for Wireless Sensor Networks," *Proc. 9th European Symp. Research Computer Security*, 2004.

[66] J. Lee and D. R. Stinson, "Deterministic Key Predistribution Schemes for Distributed Sensor Networks," *Proc. Selected Areas Cryptography*, 2004, pp. 294–307.

[67] J. Lee and D. R. Stinson, "A Combinatorial Approach to Key Predistribution for Distributed Sensor Networks," *Proc. IEEE Wireless Commun. and Net. Conf.*, 2005.

[68] L. Eschenauer and V. D. Gligor, "A Key-Management Scheme for Distributed Sensor Networks," *CCS '02: Proc. 9th ACM Conf. Comp. and Commun. Security*, New York: ACM Press, 2002, pp. 41–47.

[69] H. Chan, A. Perrig, and D. Song, "Random Key Predistribution Schemes for Sensor Networks," *Proc. IEEE Symp. Security and Privacy*, May 2003.

[70] D. Liu and P. Ning, "Establishing Pairwise KeysDistributed Sensor Networks," *CCS '03: Proc. 10th ACM Conf. Comp. and Commun. Security*, New York: ACM Press, 2003, pp. 52–61.

[71] R. D. Pietro, L. V. Mancini, and A. Mei, "Random Key-Assignment for Secure Wireless Sensor Networks," *SASN '03: Proc. 1st ACM Wksp. Security of Ad Hoc and Sensor Networks*, New York: ACM Press, 2003, pp. 62–71.

[72] W. Du *et al.*, "A Pairwise Key Predistribution Scheme for Wireless Sensor Networks," *CCS '03: Proc. 10th ACM Conf. Comp. and Communications Security*, New York: ACM Press, 2003, pp. 42–51.

[73] W. Du *et al.*, "A Key Management Scheme for Wireless Sensor Networks using Deployment Knowledge," *Proc. IEEE INFOCOM*, Hong Kong, 2004, pp. 586–97.

[74] D. D. Hwang, B. Lai, and I. Verbauwhede, "Energy-Memory-Security Trade-OffsDistributed Sensor Networks," *Proc. 3rd Int'l. Conf. Ad-Hoc Networks and Wireless*, July 2004, pp. 70–81.

[75] C. Blundo *et al.*, "Perfectly-Secure Key Distribution for Dynamic Conferences," *CRYPTO '92: Proc. 12th Annual Int'l. Cryptology Conf. Advances in Cryptology*, London: Springer-Verlag, 1993, pp. 471–86.

[76] R. Blom, "An Optimal Class of Symmetric Key Generation Systems," *Proc. EUROCRYPT '84 Wksp., Advances in Cryptology*, New York: Springer-Verlag, 1985, pp. 335–38.

[77] D. Liu and P. Ning, "Location-Based Pairwise Key Establishments for Static Sensor Networks," *Proc. ACM Wksp. Security Ad Hoc and Sensor Networks*, Oct. 2003.

[78] J. N. Al-Karaki and A. E. Kamal, "Routing Techniques in Wireless Sensor Networks: A Survey," *IEEE Wireless Commun.*, vol. 11, no. 6, Dec. 2004, pp. 6–28.

[79] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks," *MobiCom '00: Proc. 6th Annual Int'l. Conf. Mobile Computing and Networking*, New York: ACM Press, 2000, pp. 56–67.

[80] Y. Yu, R. Govindan, and D. Estrin, "Geographical and Energy Aware Routing: A Recursive Data Dissemination Protocol for Wireless Sensor Networks," UCLA Computer Science Department, Tech. Rep. UCLA/CSD-TR-01-0023, May 2001.

[81] T. Leinmüller *et al.*, "Influence of Falsified Position Data on Geographic Ad-Hoc Routing," *2nd European Wksp. Security and Privacy Ad Hoc and Sensor Networks (ESAS 2005)*, LNCS, July 2005.

[82] D. Liu and P. Ning, "Efficient Distribution of Key Chain Commitments for Broadcast AuthenticationDistributed Sensor Networks," *Proc. 10th Annual Network and Distributed System Security Symp.*, San Diego, CA, Feb. 2003, pp. 263–76.

[83] D. Liu and P. Ning, "Multilevel mTESLA: Broadcast Authentication for Distributed Sensor Networks," *Trans. Embedded Computing Sys.*, vol. 3, no. 4, 2004, pp. 800–36.

[84] D. Liu *et al.*, "Practical Broadcast AuthenticationSensor networks," *MobiQuitous '05: Proc. 2nd Annual Int'l. Conf. Mobile and Ubiquitous Systems: Networking and Services*, July 2005, pp. 118–29.

[85] W. Du, R. Wang, and P. Ning, "An Efficient Scheme for Authenticating Public KeysSensor Networks," *MobiHoc '05: Proc. 6th ACM Int'l. Symp. Mobile Ad Hoc Net. and Comp.*, New York: ACM Press, 2005, pp. 58–67.

[86] Public-Key Infrastructure (X.509) (pkix), available: http://www.ietf.org/html.charters/pkix-charter.html

[87] N. Shrivastava *et al.*, "Medians and Beyond: New Aggregation Techniques For Sensor Networks," *SenSys '04: Proc. 2nd Int'l. Conf. Embedded Networked Sensor Systems*, New York: ACM Press, 2004, pp. 239–49.

[88] J. Deng, R. Han, and S. Mishra, "Security Support for in-Network Processing Wireless Sensor Networks," *SASN '03: Proc. 1st ACM Wksp. Security of ad Hoc and Sensor Networks*, New York: ACM Press, 2003, pp. 83–93.

[89] H. Çam, D. Muthuavinashiappan, and P. Nair, "ESPDA: Energy Efficient and Secure Pattern-Based Data Aggregation for Wireless Sensor Networks," *Proc. IEEE Sensors*, Toronto, Canada, Oct. 2003, pp. 732–36.

[90] H. Çam, D. Muthuavinashiappan, and P. Nair, "Energy-Efficient Security Protocol for Wireless Sensor Networks," *Proc. IEEE VTC Conf.*, Orlando, FL, Oct. 2003, pp. 2981–84.

[91] H. Çam *et al.*, *Sensor Network Operations*, Wiley, 2004, ch. "Secure Differential Data Aggregation for Wireless Sensor Networks".

[92] W. Du *et al.*, "A Witness-Based Approach for Data Fusion Assurance wireless Sensor Networks," *GLOBECOM '03: Proc. IEEE Global Telecommun. Conf.*, San Francisco, CA, Dec. 2003, pp. 1435–39.

[93] D. Wagner, "Resilient Aggregation Sensor Networks," *SASN '04: Proc. 2nd ACM Wksp. Security of Ad Hoc and Sensor Networks*, New York: ACM Press, 2004, pp. 78–87.

[94] F. Ye *et al.*, "Statistical En-Route Filtering of Injected False Datasensor Networks," *Proc. IEEE INFOCOM*, Hong Kong, 2004.

[95] J. Girao, D. Westhoff, and M. Schneider, "CDA: Concealed Data Aggregation for Reverse Multicast Traffic wireless Sensor Networks," *ICC '05: Proc. IEEE Int'l. Conf. Commun.*, Seoul, Korea, May 2005.

[96] C. Castelluccia, E. Mykletun, and G. Tsudik, "Efficient Aggre-

gation of Encrypted Data Wireless Sensor Network," *Proc. ACM/IEEE Mobiquitous*, San Diego, CA, July 2005.
[97] J. Domingo-Ferrer, "A Provably Secure Additive and Multiplicative Privacy Homomorphism," *Lecture Notes Comp. Sci.*, vol. 2433, 2002, pp. 471–83.
[98] D. Wagner, "Cryptanalysis of an Algebraic Privacy Homomorphism," *ISC '03: Proc. 6th Info. Security Conf.*, Bristol, U.K., Oct. 2003.
[99] S. Marti *et al.*, "Mitigating Routing Misbehavior Mobile Ad Hoc Networks," *MobiCom '00: Proc. 6th Annual Int'l. Conf. Mobile Comp. and Net.*, New York: ACM Press, 2000, pp. 255–65.
[100] Y. Zhang, W. Lee, and Y. Huang, "Intrusion Detection Techniques for Mobile Wireless Networks," *Wireless Networks*, vol. 9, no. 5, 2003, pp. 545–56.
[101] Y. Huang *et al.*, "Cross-Feature Analysis For detecting Ad-Hoc Routing Anomalies," *ICDCS '03: Proc. 23rd Int'l. Conf. Distributed Computing Systems*, Providence, RI, May 2003.
[102] Y. Huang and W. Lee, "Attack Analysis and Detection for Ad Hoc Routing Protocols," *RAIS '04: Proc. 7th Int'l. Symp. Recent Advances Intrusion Detection*, Sophia Antipolis, France, Sept. 2004.
[103] B. H. Bloom, "Space/Time Trade-Offshash Coding with Allowable Errors," *Commun. ACM*, vol. 13, no. 7, 1970, pp. 422–26.
[104] J. Deng, R. Han, and S. Mishra, "INSENS: Intrusion-Tolerant Routing Wireless Sensor Networks," Department of Computer Science, University of Colorado, Tech. Report CU CS-939-02, Nov. 2002.
[105] J. Deng, R. Han, and S. Mishra, "A Performance Evaluation of Intrusion-Tolerant Routing Wireless Sensor Networks," *IPSN '03: Proc. IEEE 2nd Int'l. Wksp. Information Processing Sensor Networks*, Palo Alto, CA, 2003, pp. 349–64.
[106] G. Wang *et al.*, "On supporting Distributed Collaboration Sensor Networks," *Proc. MILCOM*, 2003.

BIOGRAPHIES

YONG WANG (ywang@cse.unl.edu) is a Ph.D. candidate in the Department of Computer Science and Engineering (CSE) at the University of Nebraska-Lincoln. He received B.S. and M.S.E degrees from Wuhan University, China in 1995 and 1998, respectively. Before joining the CSE department, he has worked as a senior telecom engineer at ZTE Corp. and UTStarcom Inc., in China. His research focuses on secure group communication and security for wireless ad hoc and sensor networks.

GARHAN ATTEBURY (attebury@cse.unl.edu) is an M.S. degree candidate in the Department of Computer Science and Engineering at the University of Nebraska-Lincoln. He received his B.S. degree from Eastern Oregon University in 2004. His research efforts are in the areas of computer network security and computational chemistry.

BYRAV RAMAMURTHY (byrav@cse.unl.edu) received his B.Tech. degree in computer science and engineering from the Indian Institute of Technology, Madras, India in 1993. He received M.S. and Ph.D. degrees in computer science from University of California (UC), Davis in 1995 and 1998, respectively. Since August 1998 he has been a faculty member in the Department of Computer Science and Engineering at the University of Nebraska-Lincoln (UNL), where he is currently an associate professor. He is the co-director of the UNL Academic Program Priority Initiative in the areas of Simulation & Computing Engineering (SCE) and Information Technology & Telecommunications (ITT). He is the founding co-director of the Advanced Networking and Distributed Experimental Systems (ANDES) Laboratory at UNL. He served as Feature Editor on Theses for *Optical Network Magazine*. He served as a guest co-editor for a special issue of *IEEE Network* on Optical Communication Networks. He served as a member of the technical program committees for the IEEE INFOCOM, IEEE GLOBECOM, Opticomm/Broadnets, ICC, and ICCCN conferences. He is author of the book *Design of Optical WDM Networks — LAN, MAN and WAN Architectures* and a co-author of *Secure Group Communications over Data Networks*, published by Kluwer Academic Publishers/Springer in 2000 and 2004, respectively. From 2001–2003, he served as the founding secretary of the IEEE ComSoc Optical Networking Technical Committee (ONTC), for which he currently serves as the Secretary and Online Content Chair. He serves as the TPC Co-Chair for the IEEE ICC 2006 Optical Systems and Networks Symposium and the Broadnets 2006 Optical Symposium. He was a recipient of the Indian National Talent Search scholarship and was a fellow of the Professors for the Future program at UC Davis. He is a recipient of the UNL Research Council Grant-in-Aid award (1999), the College of Engineering and Technology Faculty Research Award (2000), and the UNL CSE Dept. Students Choice Award for the Best Graduate Professor (2002–2003). His research areas include optical networks, wireless/sensor networks, network security, distributed computing, and telecommunications. His research is supported by the U.S. National Science Foundation, Agilent Tech., and OPNET Inc.