PROVIDING HOLISTIC SECURITY IN SENSOR NETWORKS

S. Olariu[1], A.Wadaa[1], L. Wilson[1], Q. Xu[1], M. Eltoweissy[2] and K. Jones[3]
[1]*Department of Computer Science, Old Dominion University, Norfolk, VA 23529, U.S.A.;*
[2]*Department of Computer Science, Virginia Tech, Falls Church, VA 22403, U.S.A.,*
[3]*NASA Langley Research Center, Hampton, VA 23681, U.S.A.*

Abstract: We propose a new paradigm for the secure operation of wireless sensor networks. The network model assumed in this paper consists of tiny, energy-constrained, commodity sensors massively deployed alongside with one or more sink nodes that provide the interface to the outside world. The sensors in the network are initially anonymous and unaware of their location. Our main contribution is a scalable and energy-efficient solution where security is based upon using parameterized frequency hopping and cryptographic keys in a unified framework to provide differential security services for wireless sensor networks.

Key words: wireless sensor networks, energy-efficient protocols, holistic security.

1. INTRODUCTION

Recent advances in nano-technology made it technologically feasible and economically viable to develop low-power, battery-operated devices that integrate general-purpose computing with multiple sensing and wireless communications capabilities [5,12,15,18]. It is expected that these small devices, referred to as *sensors*, will be mass-produced making production costs negligible. Individual sensors have a non-renewable power supply and, once deployed, must work unattended. For most applications, we envision a massive random deployment of sensors, numbering in the thousands or tens of thousands. Aggregating sensors into sophisticated computation and communication infrastructures, called *wireless sensor networks* (WSN), will have a significant impact on a wide array of applications including military, scientific, industrial, health, and domestic [1,8].

A WSN is only as good as the information it produces. In this respect, perhaps the most important concern is *information* security since WSNs will constitute a mission critical component requiring commensurate security protection [1,3,4,10,16]. Recently, the problem of securing ad-hoc networks has received a great deal of well-deserved attention in the literature [7,9,11,17,19]. However, since WSNs are significantly different in their characteristics from ad-hoc networks, security solutions designed specifically for the former do not apply to the latter [14]. Quite recently, a number of solutions for securing WSNs have been proposed in the literature [5,10,13,15,16]. Somewhat surprisingly, none of these solutions addresses the problem of jamming. Furthermore, all assume sensors with unique identities.

We take a novel look at the problem of securing WSNs and show that by a suitable enhancement, the classic frequency hopping strategy provides a lightweight and robust mechanism for securing WSNs. A significant advantage of our solution is that it is readily applicable to networks having anonymous nodes that are unaware of location. Also, our solution supports a *differential security service* that can be dynamically configured to accommodate changing application and network state.

2. THE SYSTEM MODEL

We assume a class of WSNs consisting of a *sink* and a large number of individual *sensors* randomly deployed within the transmission range of the sink. The sensors operate subject to five fundamental constraints. First, sensors are *anonymous*, that is, they do not have either fabrication-time or run-time identities. Second, each sensor has a modest non-renewable power budget. Third, the sensors are in sleep mode most of the time, waking up at random points in time for short intervals under the control of a watchdog timer. Fourth, upon deployment the nodes must work unattended as human intervention is both impractical and undesirable. Fifth, the sensors have a modest transmission range, perhaps a few meters with the ability to send and receive a wide range of frequencies. The range constraint implies that outbound messages sent by a sensor can reach only the sensors in its proximity, typically a small fraction of the sensors in the entire network. As a consequence, the WSN must be multi-hop and only a limited number of the sensors count the sink among their one-hop neighbors. For reasons of scalability, it is assumed that no sensor knows the topology of the network.

The WSN interfaces to the outside world through a sink. The sink has a full range of computational capabilities, can send long-range directional broadcasts to all sensors on a wide range of frequencies, can receive messages from nearby sensors, and has a steady power supply. The sink may or may not be collocated with the WSN, may be stationary or may be mobile (e.g. a LEO satellite, an airplane, or helicopter over-flying the deployment area). In this paper we assume a sink collocated with the WSN. We realize that such a sink is a single point of failure. This can be mitigated by having the role of the sink performed by a collection of devices.

2.1. Acquiring location awareness

We refer to training as the task of endowing individual sensors with *coarse-grain* location awareness in the area of deployment. Training has to be contrasted with localization whose stated goal is to provide sensors with *accurate* location information [1]. The goal of training is to establish clusters and to organize the WSN for node-to-sink multi-hop communications. For simplicity, we assume that the sink is centrally placed, although this is not necessary. Training imposes a *coordinate system* onto the WSN in such a way that each sensor belongs to exactly one *sector*. Referring to Figure 1, the coordinate system divides the WSN into equiangular wedges. In turn, these wedges are divided into sectors by means of *coronas* centered at the sink and whose radii are determined to optimize the transmission efficiency of sensors-to-sink transmission [14]. It is very important to note that sectors are implementing, at no additional cost, the concept of clustering. Indeed, the nodes in the same sector have the same coordinates (i.e. the same corona and the same wedge) uniquely defining the cluster to which they belong. This clustering mechanism is consistent with the anonymity of sensors and is extremely light-weight. Each node knows the identity of the cluster to which it belongs but has no information about the other nodes in the same cluster.

Figure 1: *A trained WSN*

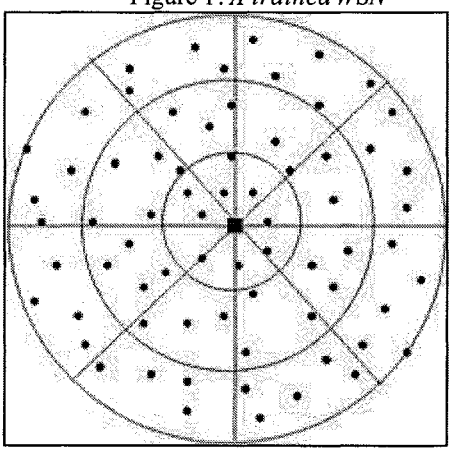
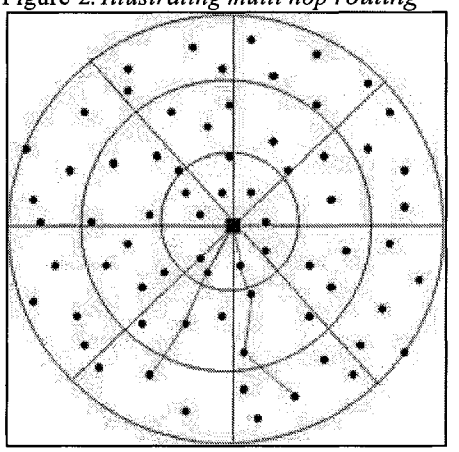


Figure 2: *Illustrating multi-hop routing*



The intuition for establishing coronas and wedges is simple and will be summarized below. As shown in Figure 1, at the end of training each sensor has acquired two coordinates: the identity of the corona in which it lies, as well as the identity of the wedge to which it belongs. The locus of all the sensors that have the same coordinates is a *cluster*. For the details of the training process we refer to [14].

2.2. Routing in a trained WSN

Recall that WSN communications are multi-hop from node to sink. Thus, in order for the sensing information to be conveyed to the sink node routing is necessary. Our cluster structure allows a very simple routing scheme in that the information is routed within one wedge along a virtual path joining the outermost sector to the sink one sector per hop, as illustrated in Figure 2.

3. NETWORK SECURITY FUNDAMENTALS

In this section we define the parameters supported by our proposed solution to securing WSNs. We begin our discussion of these parameters by briefly reviewing some fundamentals of network security. We then go on to describe the principles underlying our application of these fundamentals. Encryption and preventing access to the physical layer are two basic techniques for securing computer nodes connected via a network.

3.1. Encryption

It is both intuitively clear and confirmed by recent work that due to the modest energy budget and limited on-board storage capabilities at the individual sensors, public-key cryptography is not an option for securing WSNs. Indeed, increasing the ratio of the total number of bits transmitted to the effective data bits (a result of encryption) increases the total number of bits transmitted and, thus, the energy consumed. Key management is also a problem for use of encryption in WSNs. How are keys generated and disseminated? How can keys be changed in a reasonable time? Humans are not available at each sensor, distribution and modification of keys are difficult, and the sensor (and thus, embedded keys) is physically at risk. Perrig *et al.* [10] describe techniques for reducing the resource requirements. While their reduction techniques are creative and substantial, they still consume nearly 50% of memory, computation, and transmission resources available in their target mote.

3.2. Preventing access to the physical layer

Frequency hopping can provide this service to wireless sensor networks. Given that techniques are known to discover a hopping sequence by monitoring transmissions, security can only be provided if the design modifies the hopping sequence in less time than is required to discover the sequence. Parameters in the specification of frequency hopping determine the time required to discover the sequence:

Hopping Set: The set of frequencies available for hopping,

Dwell Time : The time interval per hop, and

Hopping Pattern: The sequence in which frequencies are visited.

A dynamic combination of these parameters can improve security at little expense of memory, computation and power. As frequency hopping requires events to happen simultaneously for both senders and receivers, all must maintain a synchronized clock.

3.3 Resistance to physical tampering

Along with other workers [2,4], we assume that the form factor and low cost of individual sensors makes for minimal tamper resistance and protection. Indeed, rudimentary tamper protection can be obtained by blanking out memory if the sensor is pried open. However, the protection offered by this solution is far for adequate. One of our contributions is to propose a lightweight solution to the tampering problem that does not rely on sophisticated hardware.

3.4 Guiding principles of holistic security in WSN

We view this paper as an initial contribution towards developing a holistic solution for securing sensor networks. Our solution provides security not only for the various individual layers of the system but also for the entire system in an integrated fashion. We now propose a set of four *guiding principles* for addressing the problem of securing WSNs. A solution in the context of these principles supports a *differential security service* that can be dynamically configured to cope with changing network state, for example, a detected state change in security risks or energy content in the network. Reconfiguration of dynamic security service can potentially minimize the energy cost of security over the network lifetime.

The four guiding principles for securing WSNs are:

i. **Security of a network is determined by the security over all layers.** For example, provisioning confidentiality, two-party authentication, and data freshness typically addresses security of the link layer. We note that securing the link layer confers the layers above some security; however, it does not address security problems in the physical layer below, most notably jamming. In general, an insecure physical layer renders the entire network insecure, even if the layers above are secure. This is especially true in the WSN environment since basic wireless communication is inherently not secure.

ii. **In a massively distributed network, security measures should be amenable to dynamic reconfiguration and decentralized management.** Given the basic characteristics of WSNs, a security solution must work without prior knowledge of the network configuration after deployment. Also, the security solution should work with minimal or no involvement of a central node to communicate globally (or regionally) shared information.

iii. **In a given network, at any given time, the cost incurred due to the security measures should not exceed the cost assessed due to the security risks at that time.** The WSN experiences risks of different magnitudes at different times, especially in view of the long-lived nature of the network. In principle, security services should adapt to changes in assessed security risk. This suggests that a cost model for both security provisioning and risk be an integral part of the security model.

iv. **If the physical security of nodes in a network is not guaranteed, the security measures must be resilient to physical tampering with nodes in the field of operation.** For example, a WSN deployed in a battlefield should exhibit graceful degradation if some sensors are captured.

3.5. Related work

Recently, the problem of securing ad-hoc networks has received a great deal of well-deserved attention in the literature [7,9,11,17,19]. However, since WSN are different in their characteristics from ad-hoc networks security solutions designed specifically for the former do not apply to the latter. Quite recently, a number of solutions for securing WSNs have been proposed in the literature [5,10,13,15,16]. We now examine some of these solutions from the viewpoint of the guiding principles proposed above. Due

to stringent page limitations, only a brief overview will be offered here. For the full details, we refer the reader to the journal version of this work.

4. OUR SOLUTION

We propose a solution for securing WSNs that adheres as closely as possible to the guiding principles stated earlier. The proposed solution uses parameterized frequency hopping and cryptographic keys in a unified framework to provide differential security services for the network. Parameters are used to define a *configuration space* for the security service. In general, different configurations of the security service are characterized by the energy cost assessed and the amount of security afforded.

It is to be noted that frequency hopping in radio communications is not a new idea and has been explored before [4,6]. Conventional frequency hopping mechanisms have been used as a means of implementing frequency diversity and interference averaging in a non-hostile context. Typically these mechanisms offered little cryptographic value.

4.1. The basic problems

In the context of our proposed solution, secure communications between a given sender and receiver, can be defined in terms of three basic problems, as described below.

4.1.1. How do we endow sensors with resistance to tampering?

The most obvious tamper resistance strategies are hardware-based and involve special hardware circuits within the sensor to protect sensitive data, special coatings or tamper seals [2]. However, hardware solutions to the tampering problems require extra circuitry that increases the cost and hardware complexity of sensors. Worse yet, the additional hardware is very likely to consume valuable energy, already in short supply. Also special coatings and seals may offer protection against some but, certainly, not all tampering attempts. Indeed, it is assumed that a sufficiently capable adversary can extract confidential information, thus compromising the sensor node. Thus, not surprisingly, tamper resistance or tamper protection is not found in present-day sensors [2,4]. Since WSNs must function unattended, the potential for tamper attacks is significant.

It is worth noting that while *pre-deployment* tamper detection may be worthwhile, *post-deployment* tamper detection is of little use in WSN since,

in the vast majority of applications, inspecting individual sensors is not an option. Our solution to endow individual sensors with tamper resistance does not rely on additional or more sophistic ated hardware.

In order to set the stage for discussing our solution, we note that the tampering *threat model* assumes that the adversary is (1) either trying to force open an individual sensor node or (2) is physically removing sensors from the deployment area. We guard against the first threat by blanking out the memory. We guard against the second threat by relying on local data that the sensor can collect, thus establishing a signature of its neighborhood. To be more specific, immediately after deployment each sensor transmits, during its wake time, on a specified sets of frequencies, using a frequency hopping sequence established prior to deployment. This allows individual sensors to collect an array of signal strengths from the sensors in their locale. It is important to recall that sensors do not have identities and that, consequently, the array of signal strengths is the only data available to the sensor node. This array, establishes, in the obvious way, a signature of the neighborhood of the node. For this reason the array will be referred to as the node's *signature array* (SA, for short). If the node is subject to removal from the area of deployment, it will notice that the radio signals it is receiving (if any) do not correspond with those in its SA. Once this condition is detected, the sensor blanks out memory thus thwarting the tampering attempt. Note also that tampering attempts that involve the removal of several sensors at the same time do not work either. This is because some node in the set of removed nodes is guaranteed to notice changes in its SA and can alert the others.

4.1.2. How does a sensor authenticate a sensor in its neighborhood?

Node *authentication* problem is one of the key problems in securing wireless networks. Our solution to node authentication relies on the signature array discussed above. Specifically, neighboring nodes exchange SA information that creating a matrix of signatures. When a node wishes to communicate with a neighbor, it establishes a trusted connection by identifying itself with its own SA. Upon receiving the SA the target node only needs verify that the corresponding signature is valid. The scheme can be made even more secure by storing several previous instance of the matrix of SAs. With this, the authentication dialogue can ask, for example, for the "second to last SA".

4.1.3. How do anonymous senders and receivers establish a trusted communication path?

In our solution communication uses a frequency hopping mechanism, as we explain shortly. The *cryptographic measure* we employ is a *randomization process* defined on the frequency hopping mechanism and driven by a secret shared by the path from sender to receiver. Within our paradigm, we propose a solution to the trust establishment problem that scales in the number of nodes, and addresses security of an entire *path*, as opposed to *hop or link* security.

4.1.4. How do a sender and a receiver synchronize?

For communication to occur, both the sender and receiver must be in sync. We propose a lightweight synchronization solution scalable in the number of nodes that enables a sender and a receiver sharing a common secret to synchronize.

4.1.5. How do a synchronized sender and receiver communicate securely?

We propose an integrated cryptographic key and parameterized frequency-hopping security solution that is scalable in the number of nodes. In Subsection 4.2 we leverage frequency hopping to provide significant security for wireless sensor networks. In Subsection 4.3 we extend our solution to further enhance this security for WSN that has been trained as described in Section 2.

4.2. Parameterized frequency hopping

We assume that the sender and the receiver are mutually trusting, and are synchronized. Our solution works as follows. We assume that time is ruled into epochs. For a given sender, s, and receiver, r, at time epoch, t, s transmits (and r receives) following a *hopping pattern* across a set of frequencies, called the *hopping set* for t. We assume that, for each epoch, the hopping set is drawn from a designated frequency space (band) that provides the set of all possible frequencies that can be used, e.g. ISM band.

The key idea is that the shared secret between s, and r, is used to drive a randomization of the frequency-hopping process. Specifically, the shared secret enables the *epoch length*, the hopping pattern, and the size and membership of the hopping set for each epoch to be changed according to random number sequences. Let both s, and r be in sync at epoch t_i. Seeded

by the shared secret, a random number generation scheme is used, in both s and r to generate the successive epoch lengths, hopping sets, and hopping patterns, for the epochs $t_i, t_{i+1}, t_{i+2}, \ldots$. To an observer, however, successive epoch lengths, hopping sets, and hopping patterns appear as the product of an unknown random process. It should be noted that our solution makes it feasible to graft an encryption scheme on top of the frequency-hoping scheme described above.

In our solution, the cost incurred by the network is a function of the configuration of the security parameters; each of the epoch length, frequency set, and frequency pattern can be dynamically configured. This gives rise to a differential security service that potentially incurs differential cost. On one hand, a constant epoch length, hopping pattern, and hopping set correspond to minimal security and minimal cost incurred. However, randomized epoch length, hopping pattern, and hopping set correspond to maximum security provided and cost incurred.

Synchronization is an important concern in any frequency-hopping scheme. In our solution we propose a scalable and lightweight synchronization scheme, to enable arbitrary pairs of sender and receiver nodes that can exchange messages directly to synchronize. This scheme is detailed next.

4.3. How do a sender and receiver synchronize?

The main goal of this subsection is to spell out the details of a scalable synchronization protocol that underlies our new security paradigm for wireless sensor networks. Our protocol achieves synchronization in a probabilistic sense. The natural way for nodes to synchronize is by following the master clock running at the sink node. Thus, the sink node here is the sender, and the node that wants to synchronize is the receiver.

We assume that the sink dwells τ micro-seconds on each frequency in the hopping sequence. It is clear that determining the epoch and the position of the sink in the hopping sequence corresponding to the epoch is sufficient for synchronization.

For the purpose of showing how synchronization is effected, assume that time is ruled into epochs $t_1, t_2, \ldots, t_n$. For every $i, (1 \le i \le n)$, we let l_i stand for $\left\lfloor \frac{t_i}{\tau} \right\rfloor$; thus, epoch t_i involves a hopping sequence of length l_i. Further, with epoch t_i $(1 \le i \le n)$, we associate a set of n_i frequencies and a corresponding hopping sequence $\lambda_1, \lambda_2, \ldots, \lambda_{l_i}$. We can think of the epoch t_i $(1 \le i \le n)$, as being partitioned into l_i slots $s_1, s_2, \ldots, s_{l_i}$ such that in slot $j, (1 \le j \le l_i)$, the sink is visiting frequency λ_j.

We assume that, just prior to deployment, the sensors are synchronized. However, due to natural clock drift re-synchronization is necessary. Our synchronization protocol is predicated on the assumption that clock drift is bounded, as we are about to explain. Specifically, assume that whenever a sensor node wakes up during its *local* time epoch t_i the master clock is in one of the time epochs t_{i-1}, t_i, or t_{i+1}. The sensor node knows the *last* frequencies $\lambda_{l_{i-1}}, \lambda_{l_i}, \lambda_{l_{i+1}}$ on which the sink will dwell in the time epochs t_{i-1}, t_i, and t_{i+1}. Its strategy, therefore, is to tune in, cyclically, to these frequencies, spending $\frac{\tau}{3}$ time units on each of them. It is clear that, eventually, the sensor node meets the sink node on one of theses frequencies. Assume, without loss of generality, that the node meets the sink on frequency λ_{i+1} in some (unknown) slot s of one of the epochs t_{i-1}, t_i, or t_{i+1}. To verify the synchronization, the node will attempt to meet the sink in slots $s+1$, $s+2$ and $s+3$ according to its own frequency hopping for epoch t_{i+1}. If a match is found, the node declares itself synchronized. Otherwise, the node will return to scanning frequencies $\lambda_{l_{i-1}}, \lambda_{l_i}, \lambda_{l_{i+1}}$ as discussed above.

We note that even if the sensor node declares itself synchronized with the sink, there is a slight chance that, in fact, it is not. The fact that the node has not synchronized will be discovered fast and the node will attempt to synchronize again. There are ways in which we can make the above synchronization protocol deterministic. For example, the hopping sequence can be designed in such a way that the last frequency in each epoch is unique and it is not used elsewhere in the epoch. However, this entails less flexibility in the design of the hopping sequence and constitutes, in fact, an instance of a *differential security service* where the level of security is tailored to suit the application or the power budget available.

5. CONCLUDING REMARKS

We have presented a new solution to the problem of securing WSNs. Specifically, we showed that a suitable enhancement to the classic frequency hopping strategy would provide a lightweight and scalable mechanism for securing WSNs. Our solution supports a *differential security service* that can be dynamically configured to accommodate changing application and network system state. We view this paper, in part, as an initial contribution towards developing a holistic solution for securing WSNs. Research is underway towards a solution that provides security not only for the various individual layers of the system but also for the entire system in an integrated fashion.

REFERENCES

1. I. F. Akyildiz, W. Su, Y. Sankarasubramanian, and E. Cayirci, Wireless sensor networks: A survey, *Computer Networks*, 38(4), 2002, 393-422.
2. R. Anderson and M. Kuhn, Tamper resistance – a cautionary note, *Proc. 2nd Usenix Workshop on Electronic Commerce,* Berkely, CA, 1996, 1-11.
3. P. Bahl, W. Russell, Y.-M. Wang, A. Balachandran, G. M. Voelker, and A. Miu, PAWNs: satisfying the need for ubiquitous secure connectivity and location services, *IEEE Wireless Communications*, 9(1), 2002, 40-48.
4. D. W. Carman, P. S. Kruus, and B. J. Matt, Constraints and approaches for distributed sensor network security, Tech. Rep. #00-010, NAI Labs, 2000.
5. Cross Bow Technologies, http://www.xbow.com/
6. A. Ephremides, J. Wieselthier and D. Baker, A design concept for reliable mobile radio networks with frequency hopping signaling, *Proceedings of the IEEE*, 75(1), 1987, 56-73.
7. J.-P. Hubaux, L. Buttyan, and S. Capkun, The quest for security in mobile ad-hoc networks, *Proc. MobiHoc,* Long Beach, October 2001, 146-155.
8. J. M. Kahn, R. H. Katz and K. S. J. Pister, Mobile networking for Smart Dust, *Proc. MOBICOM'99,* Seattle, WA, August 1999.
9. S. Marti, T. J. Giuli, K. Lai, and M. Baker, Mitigating routing misbehavior in mobile ad-hoc networks, *Proc. MOBICOM,* Boston, MA, August 2000.
10. A. Perrig, R. Szewczyk, V. Wen, D. Culler and J. D. Tygar, SPINS: Security protocols for sensor networks, *Proc. MOBICOM,* Rome, Italy, August 2001, 189-199.
11. A. Pfitzmann, B. Pfitzmann and M. Waidner, Trusting mobile user devices and security modules, *IEEE Computer*, 30(2), 1997, 61-68.
12. P. Saffo, Sensors, the next wave of innovation, *Communications of the ACM*, 40(2), 1997, 93-97.
13. TinySec http://www.cs.berkeley.edu/~nks/tinysec/
14. A. Wadaa, S. Olariu, L. Wilson, K. Jones, and Q. Xu, On training wireless sensor networks, *MONET,* February 2005, to appear.
15. B. Warneke, M. Last, B. Leibowitz, and K. Pister, SmartDust: communicating with a cubic-millimeter computer, *IEEE Computer*, 34(1), 2001, 44-51.
16. A. D. Wood and J. A. Stankovic, Denial of service in sensor networks, *IEEE Computer*, 35(4), 2002, 54-62.
17. Y. Zhang and W. Lee, Intrusion detection in ad-hoc networks, *Proc. MOBICOM,* Boston, MA, August 2000.
18. V. V. Zhirnov and D. J. C. Herr, New frontiers: self-assembly and nano-electronics, *IEEE Computer*, 34(1), 2001, 34-43.
19. L. Zhou and Z.J. Haas, Securing ad-hoc networks, *IEEE Network*, 13(6), 1999, 24-30.