An Algorithmic Framework for Robust Access Control in Wireless Sensor Networks

Zinaida Benenson* Felix C. Gärtner Dogan Kesdogan

Department of Computer Science
RWTH Aachen University of Technology
D-52056 Aachen, Germany

Abstract

If the data collected within a sensor network is valuable or should be kept confidential then security measures should protect the access to this data. We first determine security issues in the context of access control *in sensor networks especially focusing on the problem of* node capture*, i.e., the possibility that an attacker can completely take over some of the sensor nodes. We then introduce the notion of* t-robust sensor networks *which can withstand capture of up to* t *nodes and consider three basic security concepts for such networks: (1)* t*-robust storage, a mechanism to securely store data within a set of sensors such that capture of any* t *sensors does not reveal that data to the adversary; (2)* n*-authentication which ensures that authentication is achieved with every uncompromised sensor in the broadcast range of a client (*n *denotes the number of nodes in that broadcast range); and (3)* n*-authorization, an authorization primitive with similar properties like* n*-authentication. We present a generic* t*-robust protocol for implementing access control using these primitives.*

1 Introduction

Wireless Sensor Networks (WSNs) are networks of tiny sensing devices which are spread over a large geographic area and can be used to collect and process environmental data like temperature, humidity, light conditions, seismic activities, images of the environment etc. This data can be used to detect certain events and to trigger activities. Some of applications are habitat monitoring, precision agriculture, wildfire detection, building and perimeter security.

With the increasing ubiquity of WSNs, environmental data will be available almost everywhere in our environment. We believe that in the future the current temperature, humidity, etc. at a particular location will be available on demand from a surrounding WSN. Of course, accessing this data will in general not be for free since deployment of WSNs induces some costs. This means that the deployment agencies of some of these services will make them available only to certain people, usually people which pay for receiving the service. In this case, a WSN must be able to distinguish legitimate users from illegitimate users, resulting in the problem of access control. Access control problem also arises inherently in such applications as building and perimeter security.

Access control is an old problem from classical computer science but has not received much attention in the context of WSNs. This is unfortunate since WSNs define an environment which naturally calls for security solutions but — due to the resource-constraints with respect to computational and battery power for example — also defines an environment in which security solutions are extremely hard to implement. As WSNs usually cannot be protected against node capture, security solutions in this domain cannot rely on single sensors. This makes security in sensor networks a challenging area. Most of the current protocols for any sensor network operations (routing, query processing, data dissemination and storage, access control) are highly susceptible to node capture [11].

Contributions. The contributions of this paper are threefold:

1. We give an overview over the security problems arising with access control in WSNs. We introduce the notion of *t-robust sensor networks* which can withstand capture of up to t nodes and discuss their security issues.
2. We dissect the access control problem for sensor networks into the separate subproblems of authentication and authorization and precisely describe versions

*Zinaida Benenson was supported by Deutsche Forschungsgemeinschaft as part of the Graduiertenkolleg "Software for mobile communication systems" at RWTH Aachen University of Technology.

of these problems which take node capture into account. We call these primitives n-authentication and n-authorization. The idea of these primitives is that authentication and authorization can take advantage of the sensor redundancy which is inherent in WSNs, namely that it is often sufficient to reach any subset of the n sensors assumed to be in one's communication range. We also describe the notion of t-robust storage which is similar to the notion of secret sharing from the area of cryptographic protocols.

3. We then precisely describe a framework for how to build access control solutions out of n-authentication and n-authorization protocols. These access control solutions again exploit the inherent redundancy of WSNs to achieve their goal.

To our knowledge, the framework presented in this paper is the first approach to build access control for WSNs which can fully withstand node capture of up to t nodes (t-*robust* access control). We argue that t-robustness should be taken into account in other areas of WSN security. Efficient implementation of the necessary primitives (e.g., n-authentication) is the focus of concurrent and forthcoming work [2].

Outline. We give a brief overview over the security issues in the context of protecting sensor network data (Section 2). We then discuss access control in sensor networks and give our adversary model (Section 3). In Section 4 we define t-robust access control and give a generic implementation in Section 5. We discuss our approach in the context of related work in Section 6 and conclude in Section 7.

2 Security Issues in WSNs

We consider a WSN with a large number of *sensors* distributed over a sensor field (see Figure 1). The WSN continuously monitors the environment and collects data about it which is stored and exchanged between the sensors. For network management we assume that there exist a small number of *base stations* which are dedicated machines that serve as access points for network administrators, for key management and other security-related tasks. Their number is small compared with the size of the sensor field. This means that they cannot be used as data access points for other users of the WSN.

A *user* (or *client*) who would like to read the data of the WSN approaches the network with a small wireless mobile device (laptop, PDA, mobile phone, etc.). Some users are *authorized* to access the data of the WSN, e.g., they have subscribed to a "WSN data service" and have paid a fee to gain access. Other (unauthorized) users may not have done this and should be prevented from gaining access to the data.

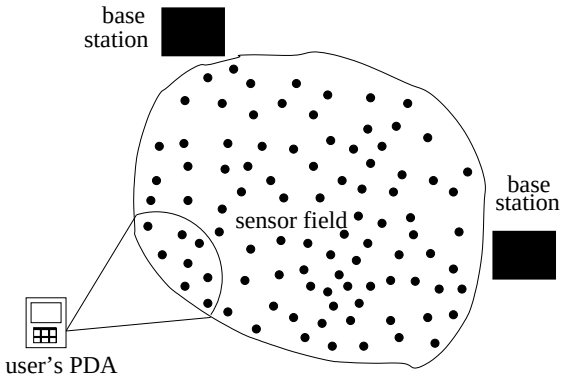

Figure 1. Sensor network architecture

Usually, there are more than one sensor in the communication range of the user. We assume the expected number of sensors which are in communication range of the user to be n.

2.1 Security Impediments in WSNs

Sensor nodes are cheap devices with several limitations which are unfavorable if security issues are at stake. Firstly, they have *limited memory and computational power*. These limitations make operations such operations as modular multiplication prohibitively slow. Secondly, sensors have *limited battery life*. Battery life determines the life of a particular sensor and of the whole sensor network. The most power consuming operation is wireless communication. And finally, the sensor *hardware is not reliable* and *not tamper proof*. Moreover, sensor networks operate unattended in open, sometimes hostile environments. This makes node capture in sensor networks especially easy.

In this paper, we pay special attention to the problem of *node capture*. As stressed in [11], defense against node capture is one of the most important and difficult problems in designing fault-tolerant and secure protocols for sensor networks since such protocols cannot rely on a single sensor for critical operations.

2.2 Inside vs. Outside Security

The data collected within a sensor network should be kept confidential to unauthorized users and so it is necessary to control the access to this data in convenient ways. There are several ways to attack the communication of a sensor network. To separate concerns we propose to distinguish *inside security* and *outside security* for WSNs.

Inside security. Inside security refers to secure communication between the sensors and secure communication be-

tween the sensors and the base stations (if there are any). In this case, base stations are usually considered to be trusted and to have a similar authority as network administrators in classical networks. Among the security problems evolving in WSNs, inside security has been studied most extensively [3, 5, 12, 20].

Outside security. In case outside users, i.e., the "subscribers" to WSN services are allowed to query a sensor network, outside security is needed in addition to the inside security. Outside security means secure communication between the WSN (sensors and base stations) and the outside users. A legitimate user can send data requests to the WSN. Usually this means that the user sends the request to some sensor or a set of sensors in her neighborhood and — if the request is legitimate — receives a valid response. However, a user should not be able to participate in arbitrary communication with the WSN, for example to hear all messages exchanged in the network. Therefore, separate means are needed for inside and outside security.

2.3 Security Goals in WSNs

We adapt the three classical security goals of *confidentiality*, *integrity* and *availability* to WSNs.

In the context of WSNs, *confidentiality* means that only authorized entities should be able to access the data of a WSN. This property can be violated by eavesdropping on an insecure wireless link. Additionally, since sensors are not assumed to be tamper proof, such data can be obtained by "taking over" a sensor, i.e., opening it and reading out its data and all data passing through that sensor.

Integrity means that the data collected and processed by a WSN cannot be changed in an unauthorized way. So if an authorized user receives data from his WSN subscription, this data is correct and valid. Integrity of the WSN data can be violated again by gaining control over some nodes and changing their internal state, or by adding adversarial sensor nodes which feed the WSN with fake data.

Finally, *availability* means that the WSN should always be able to answer any authorized request in due time. Violation of availability results in denial of service. This can be achieved by destroying sensors or by running some denial of service attack on a WSN. Some of these attacks consist of battery draining, jamming and deliberate collisions by sensor nodes which are controlled by an adversary.

Data integrity and availability are important aspects of outside security and have been studied, e.g., in [13] and [18]. On the other hand, *access control*, the heart of solutions to confidentiality, has not received much attention yet.

3 Access Control Issues in WSNs

In this section we discuss the notion of access control and its different phases of authentication and authorization. We then present our adversary model and discuss its implications to access control solutions.

3.1 Authentication and Authorization

A method of access control should provide access to legitimate users and deny access to illegitimate ones. There are two main issues in this process. *Authentication* means establishing a relation between the user and some identity. *Authorization* means establishing a relation between a user and a set of privileges (access rights or allowed operations). An *identity* is the individuality property of a user which ideally cannot be forged or copied. In practice, identities are implemented by items which users know (passwords), possess (secret keys or security tokens) or properties which they have (biometrics).

In authentication, a user sends his name (e.g., IP address) and proof of his identity to a sensor and the sensor should be able to decide whether or not the identity is valid and in fact belongs to the user of that name. This can be done, e.g., by verifying a digitally signed certificate.

In authorization, a users sends his name together with the requested access operations (e.g., read, write) to a sensor and the sensor should be able to decide whether or not this user is allowed to perform this operation. This is usually implemented by access control lists.

Finally, in access control, a user sends his name, identity, and the requested operation to the sensor. The sensor first authenticates the user, i.e., it checks the validity of the name and identity. Upon successful authentication, the sensors authorizes the user, i.e., it checks the access permissions of that user. If both checks succeed, the user is granted access to the data.

In practice, authentication, authorization and data access are combined into one single operation. A request is sent, the access control mechanism checks legitimacy (authentication and authorization), and sends a response back to the user (which may be the data requested or a message "access denied").

3.2 Adversary Model

We assume the existence of an adversary which wants to fool the system into violating one of its security goals. The adversary is successful if he can manage that WSN data is revealed to illegitimate users (loss of confidentiality), legitimate users receive corrupted data (loss of integrity), or legitimate users are denied access to the data (loss of availability).

We assume the adversary can access the WSN in ways which are open to regular users too, i.e., the adversary can send requests to sensors in his communication range. Within his communication range, the adversary can also eavesdrop on all communications within the sensor network and between the WSN and legitimate users (i.e., we assume a *local* adversary, not a global one which can eavesdrop on the entire WSN at all times).

Furthermore, the adversary cannot fully "impersonate" a legitimate user, i.e., he cannot aquire full proof of that user's identity. In practice this means that the adversary cannot "guess" the secret key (or the password) of a legitimate user. However, any solution to access control needs to be careful that the identity of a legitimate user cannot be inferred from protocol messages (i.e., when a secret key is sent in clear text over a communication link). We assume that the adversary cannot read encrypted messages unless he has the necessary keys.

We also assume that the adversary can take *full control* over some set of sensors. Full control means that he can read all data stored within that sensor, he can observe all messages passing through that sensor, and he can use the sensor for sending messages into the sensor network. However, we assume that at most t out of the n sensors in his communication range can be corrupted by the adversary.

We assume that the adversary cannot take over a base station. This is necessary in our model (Section 2) where base stations are trusted entities responsible for network and trust management. This is one of standard assumptions on the sensor network architecture [4,12,20]. There, a takeover of a base station would mean a complete breakdown in the inside security. One of approaches to avoid trusted base stations is random key predistribution [3, 5].

Apart from taking over at most t out of n sensors, the adversary could use jamming to prevent other users or sensor nodes to communicate with sensors within the adversary's communication range. Solutions to this denial-of-service attack are out of scope of this paper.

Moreover, in the real world, the adversary could be able to impersonate the WSN to the user. Therefore, sensor nodes must be able to authenticate themselves to the user. We leave mutual authentication to the future work. For the present, we assume that the adversary cannot impersonate the WSN.

3.3 Implications of Adversary Model: t-robust WSNs

Considering the rather strong adversary model above, it is a challenge to implement the security goals from Section 2. In this model, a WSN needs to be able to withstand capture of up to t nodes. We call such sensor networks t-robust. All critical operations in such networks need to be t-robust, including inside security, data aggregation and authentication of the WSN to its users [1]. However, in this paper we focus on impact of t-robustness on access control to the WSN services. There are several implications.

Robust storage of data: If the confidential sensor data is *produced and stored* on a sensor within the communication range of the adversary, then nothing can be done to prevent disclosure of that data if the adversary can take over such a sensor. This means that data must be stored in such a way that t sensors together cannot construct "real" data or cause a legitimate user to receive "forged" data.

Robust authentication of users: Care must be taken that a final access to sensor data does not depend on the authorization operation of a single sensor (or generally less than $t + 1$ sensor). If the adversary takes over such a sensor, the sensor may pretend to successfully authenticate an illegitimate user, resulting in unintended disclosure of data. Or the sensor may fail to authenticate a legitimate user, resulting in loss of availability. In general, this requires that $n > 2t$, i.e., the number of uncompromised sensors should "dominate" corrupted sensors.

Robust authorization: Similar to authentication, authorization may not depend on the decision of t or less sensors.

4 Robust Access Control Primitives

In this section we present primitives which address the implications of the adversary model discussed in the previous section.

4.1 t-robust Storage

Data is stored in a t-robust way if no t sensors in the communication range of the adversary can construct confidential data or cause a legitimate user to receive corrupted data. More precisely, every sensor V_i has a share d_i of the jointly stored sensor date D. If necessary, a sensor V_i may send its share to some entity. That entity will be able to derive D only if at least $t + 1$ shares are received. We assume that the entity can distinguish between a correct share and random data sent to it by some corrupted sensor.

There are two basic ways to implement t-robustness. *Application specific sharing* is possible if the user needs aggregated data, e.g., average temperature in some region must be found, where the number of measured values should be more than t. However, this cannot be assumed for general types of queries.

Generic sharing can be implemented using an (n, t)-secret sharing scheme such as [15] where a special entity called the *dealer* shares the data between n sensors such

[1] For example, the assumption about locality of the adversary implies t-robust inside security, including routing, as otherwise an adversary could be able to take over entire network after capturing t nodes.

that at least $t+1$ sensors are needed to recover the information. If the dealer is compromised, the data cannot be secure. Therefore in our threat model this method is not appropriate if the data is stored in the communication range of the adversary. However, if the data is stored at a sensor or set of sensors which are out of the adversary's communication range and therefore, according to our assumption, cannot be compromised, secret sharing easily can be applied using one of the remote sensors as the dealer.

4.2 n-Authentication

Menezes et al. [10, p. 386] define the term *entity authentication* as "...the process whereby one party is assured [...] of the identity of a second party involved in a protocol...". In our approach, the first party is distributed and consists of the nodes of the sensor network which are in the communication range of the second party (the client). We let P denote the client ("prover") and we let $\mathcal{V} = \{V_1, \ldots . V_n\}$ denote the set of sensors in the communication range of P ("verifiers"). The verifiers are requested by the prover to establish a correct relation between a particular identity and the prover.

There can be multiple provers having the same identity, e.g., Alice's PDA, her workstation or her mobile phone can all be associated with the identity of Alice. We assume that a prover has at most one identity. We denote the set of all identities by $\mathcal{I}$.

We now define the properties of *simple authentication* protocols. These properties are defined with respect to the two primitive operations of authentication: (1) *authenticate*(V, I) is invoked by the prover P whenever P would like to be authenticated by V using identity $I \in \mathcal{I}$; (2) *associate*(P, I) is invoked by the verifier whenever it has established the relation between P and some identity I. Intuitively, an authentication protocol is correct if the identity associated to P by V is the "real" identity of P. If P is dishonest or claims to have a fake identity this is indicated by a special value $\perp$ which is supposed to be distinct from any value in $\mathcal{I}$. Authentication is *successful* if V invokes *associate*(P, I) with some $I \neq \perp$.

We now introduce the notion of n-authentication, a robust version of simple authentication. To be robust against failures, this new form of authentication succeeds even if the user cannot authenticate with a subset of sensors out of a set of n sensors.

To distinguish the primitive operations of simple authentication from those of n-authentication we denote the latter ones with n-*associate*(P, I) and n-*authenticate*$(\mathcal{V}, I)$. Note that n-authenticate refers to the entire set of verifiers while n-associate just refers to a single prover.

A protocol solves n-authentication if it satisfies the following properties:

- (Termination) If P invokes n-*authenticate*$(\mathcal{V}, I)$ then eventually all honest $V_i \in \mathcal{V}$ invoke n-*associate*(P, I_i) for some $I_i \in \mathcal{I}$ or $I_i = \perp$.
- (Validity) An honest verifier V_i invokes n-*associate*(P, I) only if P in fact has identity $I \in \mathcal{I}$.
- (Agreement) If honest verifier V_i invokes n-*associate*(P, I') and honest verifier V_j invokes n-*associate*(P, I'') then $I' = I''$.

If we assume that at most t verifiers fail, then n-authentication ensures that the remaining (at least $n-t$) verifiers eventually successfully authenticate an honest prover and that they agree on his identity. If a prover is dishonest or claims to have a fake identity then all honest verifiers will return $\perp$ so that the prover is not authenticated.

4.3 n-Authorization

Authorization means establishment of authority, i.e., knowledge that an agent is enabled to perform some action. Again, we denote the client by P, the set of sensors $\mathcal{V} = \{V_1, \ldots, V_n\}$ and we use $\mathcal{A}$ to denote a set of possible actions which may be performed by P on the sensor data.

In standard security protocols, authorization is usually performed between a single prover and a single verifier. As discussed above, we need a more robust notion of authorization. We specify n-*authorization* using two primitives: (1) n-*authorize*$(\mathcal{V}, A)$ is invoked by P whenever he wishes to be authorized by a set of sensors $\mathcal{V}$ to perform action $A \in \mathcal{A}$ on the sensor data; (2) n-*grant*(P, A) is invoked by a verifier V_i whenever he has established the legitimacy of the relation between P and the action A. Intuitively, V_i should only invoke n-*grant* with action A if P in fact has the right to perform A. If P is dishonest and wants to perform an action to which it is not entitled, then an honest verifier invokes n-*grant* with a special value $\perp$ which is distinct from any action in $\mathcal{A}$.

More precisely, a protocol solves n-authorization if it satisfies the following properties:

- (Termination) If P invokes n-*authorize*$(\mathcal{V}, A)$ then eventually all honest $V_i \in \mathcal{V}$ invoke n-*grant*(P, A_i) for some $A_i \in \mathcal{A}$ or $A_i = \perp$.
- (Validity) An honest verifier V_i invokes n-*grant*(P, A) only if P in fact has the right to perform action A.
- (Agreement) If honest verifier V_i invokes n-*grant*(P, A') and honest verifier V_j invokes n-*grant*(P, A'') then $A' = A''$.

Similarly to n-authentication, n-authorization ensures that all honest verifiers will agree on the outcome of the authorization request.

4.4 Robust Access Control

We now finally turn our attention to access control. Access control is about granting access to resources. This should only be done if the requesting entity can prove that it is authentic and authorized to have access. For simplicity, we assume that there is only one action A which can be performed on the sensor network. For example, A may mean "read the current temperature". We assume that the sensor network stores some data D which we call the *correct* data.

We define access control using two primitives, both of which are invoked by the prover: (1) *request*(I, A) is invoked by the prover, who claims to have identity I, whenever he wishes to perform action A (e.g., read the current temperature); (2) *response*(D') happens at the prover when he receives the answer to his request. Intuitively, D' should be equal to D whenever the prover is authentic and authorized to perform A.

A *robust access control* protocol satisfies the following conditions:

- (Integrity) If an honest prover with identity I previously invoked *request*(I, A) and may perform action A and if he receives *response*(D') then the data D' is correct data, i.e., $D' = D$.
- (Availability) If an honest prover with identity I may perform action A and invokes *request*(I, A) then eventually he will receive *response*(D') for some data D'.
- (Confidentiality) The adversary learns nothing about the sensor data.

From the above definition follows that access control requires authentication and authorization. To be robust against the assumed adversary, the response to a request by a legitimate user must be authenticated and authorized by at least $t + 1$ sensors. We will use n-authentication, n-authorization, and t-robust storage to implement robust access control in the following section.

5 A Generic Access Control Protocol

Assume we have solutions to t-robust storage, n-authentication, and n-authorization. Then we can implement robust access control in the following way:

The sensor data D is stored in a t-robust way. Whenever a client wishes to read the data, it invokes n-authentication using his identity. Immediately following this, the client invokes n-authorization. A sensor V_i which invokes n-associate *and* n-grant for the same client sends his share back to the client in encrypted form. The client waits for $t + 1$ correct shares, reconstructs D which is taken as the value of the response.

To show that this protocol satisfies robust access control, we consider the three properties Integrity, Availability, and Confidentiality in turn.

Availability and Integrity: Assume the client is honest, authentic, may perform the read action, and sends a request. The properties of n-authentication and n-authorization ensure that all correct sensors will eventually invoke n-associate and n-grant. Since we assume $n > 2t$, this means that $n - t \geq t + 1$ correct sensors will send their share to the client. So the client is able to reconstruct some data. This shows Availability. Since the client is able to select the shares from the correct sensors, he is able to reconstruct the correct data D. This shows Integrity.

Confidentiality: We need to show that the adversary cannot learn anything about the sensor data. If the client is controlled by the adversary and therefore, does not have the claimed identity or is not allowed to read the sensor data, then no correct sensor will successfully invoke n-associate or n-grant. Otherwise, the Validity property of n-authentication or n-authorization, respectively, would be violated. If the adversary additionally controls t sensors, at most t shares will be sent back to the client. From the properties of t-robust storage this is not sufficient to reconstruct D. Eavesdropping on the n-authorization protocol is of no help to the adversary. Similarly, eavesdropping on the correct sensors which send their share to the (legitimate) clients is also of no use since that communication is encrypted.

6 Discussion and Related Work

There are many questions arising from the previous work, some of which we now discuss.

The approach seems to be rather theoretical. What about real protocols? Is there any evidence of practicality? The final goal of the research described herein is to develop practical access control protocols for WSNs. In this paper we present the first step, a detailed and formal separation of concerns. This makes it easier to develop protocols which are correct and additionally clarifies the inherent trade-offs and structures of the problem. Based on this framework, we are developing several protocols for t-robust access control using symmetric key cryptography and inexpensive asymmetric key cryptography [9]. We refer the reader to [2] for a first example.

What about the costs of the protocols in terms of computation and communication? They seem to be very high? One objection against security protocols in WSNs is that, regarding the computation complexity, asymmetric cryptography is infeasible for sensor nodes. However, for some cases, such as modular cubing [17] or elliptic curve

cryptography [9], these restrictions do not necessarily apply.

The asymptotic communication complexity of the protocols we have developed (see [2] for an example) can be made linear in the order of n and t by exploiting the broadcast nature of the communication. This may still be considered too much since communication is by far the most energy-consuming task of sensor nodes.

How adequate is the adversary model in Section 3.2 and how necessary are the restrictions on the adversary in the protocols? The assumptions we make about node capture are in general *worst case* assumptions which need to be justified in practice using a concrete technology. The currently available technologies (such as MICA MOTEs [7]) are not tamper proof which justifies the assumption that the adversary *can* take full control over a set of sensors, which of course in practice may incur considerable costs. Stronger restrictions on the adversary (e.g., assuming that some part of the sensor memory is physically protected in tamper-proof hardware) will make it possible to develop more efficient protocols. If you assume that the adversary is constrained by a certain maximal effort (like costs to break a sensor) it is possible to adapt the system parameters t and n to reflect this. The framework does not have to be changed.

What about the assumptions on n and t? What numbers would you expect in practice? Values for n and t depend on the concrete application environment of the WSN. Considering the typical range of a sensor's radio link, values of n could range between 5 and 100. We expect the value of t to be much smaller than n, usually in the range between 1 and 5.

What is the related work in the area of WSN security? To our knowledge, algorithms for access control in sensor networks which can withstand malicious node capture have not been considered so far. However, there are several algorithms which tolerate node capture in other areas, including routing [4], data aggregation [13] and key management [3].

There is also rich literature on robust security implementations in wireless ad hoc networks, including authentication. In this context, authentication of single nodes to other members of the ad hoc network is considered, which is an aspect of inside security and so unrelated to access control for outsiders (users) which we consider here. There are solutions using threshold cryptography [8, 19], hierarchical network architecture and public key cryptography [16] or symmetric mechanisms [1]. However, none of these solutions is really applicable to sensor networks due to extensive use of resource hungry cryptography or unsuitable methods for bootstrapping trust (e.g., using physical contact of devices) [1].

How does this approach differ from work in the areas of fault-tolerance and traditional security? The notion of t-robustness has similarities to work in the area of fault-tolerant and secure data replication. For example, Rabin [14] adapts secret sharing techniques from the area of cryptography to make information available and keep it secure even if up to a certain fraction of nodes behave arbitrarily. As another example, Herlihy and Tygar [6] modify a fault-tolerant replication protocol with secret sharing to maintain the confidentiality of data in a similar setting. However, both papers do not focus on access control.

7 Conclusions

We introduced t-robust security for sensor networks, discussed security issues in this context and presented a framework for organizing t-robust access control to sensor network data.

References

[1] D. Balfanz, D. K. Smetters, P. Stewart, and H. C. Wong. Talking to strangers: Authentication in ad-hoc wireless networks. In *Proceedings of Network and Distributed System Security Symposium 2002 (NDSS'02)*, San Diego, CA, February 2002.

[2] Z. Benenson, F. C. Gärtner, and D. Kesdogan. User authentication in sensor networks (extended abstract). In *Informatik 2004, Workshop on Sensor Networks*, September 2004.

[3] H. Chan, A. Perrig, and D. Song. Random key predistribution schemes for sensor networks. In *IEEE Symposium on Security and Privacy*, pages 197–213, May 2003.

[4] J. Deng, R. Han, and S. Mishra. A performance evaluation of intrusion-tolerant routing in wireless sensor networks. In *2nd IEEE International Workshop on Information Processing in Sensor Networks (IPSN 2003)*, April 2003.

[5] L. Eschenauer and V. D. Gligor. A key-management scheme for distributed sensor networks. In *Proceedings of the 9th ACM conference on Computer and communications security*, pages 41–47. ACM Press, 2002.

[6] M. P. Herlihy and J. D. Tygar. How to make replicated data secure. In C. Pomerance, editor, *Advances in Cryptology—CRYPTO '87*, volume 293 of *Lecture Notes in Computer Science*, pages 379–391. Springer-Verlag, 1988, 16–20 Aug. 1987.

[7] J. L. Hill and D. E. Culler. Mica: A wireless platform for deeply embedded networks. *IEEE Micro*, 22(6):12–24, 2002.

[8] J. Kong, P. Zerfos, H. Luo, S. Lu, and L. Zhang. Providing robust and ubiquitous security support for mobile ad hoc networks. In *Proceedings of the Ninth International Conference on Network Protocols (ICNP'01)*, page 251. IEEE Computer Society, 2001.

[9] D. J. Malan, M. Welsh, and M. D. Smith. A public-key infrastructure for key distribution in tinyos based on elliptic curve cryptography. In *First IEEE International Confer-*

ence on Sensor and Ad Hoc Communications and Networks, Santa Clara, California, October 2004.

[10] A. J. Menezes, P. C. von Oorschot, and S. A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, Boca Raton, FL, 1997.

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

[12] A. Perrig, R. Szewczyk, V. Wen, D. Culler, and J. D. Tygar. Spins: security protocols for sensor netowrks. In *Proceedings of the 7th annual international conference on Mobile computing and networking*, pages 189–199. ACM Press, 2001.

[13] B. Przydatek, D. Song, and A. Perrig. SIA: Secure information aggregation in sensor networks. In *ACM SenSys 2003*, Nov 2003.

[14] M. O. Rabin. Effi cient dispersal of information for security, load balancing, and fault tolerance. *Journal of the ACM*, 36(2):335–348, 1989.

[15] A. Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, 1979.

[16] L. Venkatraman and D. Agrawal. A novel authentication scheme for ad hoc networks. In *IEEE Wireless Communications and Networking Conference (WCNC 2000)*, volume 3, pages 1268–1273, 2000.

[17] R. Watro, D. Kong, S. Cuti, C. Gardiner, C. Lynn, and P. Kruus. Tinypk: securing sensor networks with public key technology. In *Proceedings of the 2nd ACM workshop on Security of ad hoc and sensor networks*, pages 59–64. ACM Press, 2004.

[18] A. D. Wood and J. A. Stankovic. Denial of service in sensor networks. *Computer*, 35(10):54–62, 2002.

[19] L. Zhou and Z. Haas. Securing ad hoc networks. *IEEE Network Magazine*, Nov./Dec. 1999.

[20] S. Zhu, S. Setia, and S. Jajodia. Leap: effi cient security mechanisms for large-scale distributed sensor networks. In *Proceedings of the 10th ACM conference on Computer and communication security*, pages 62–72. ACM Press, 2003.