A Distributed Authentication Scheme for a Wireless Sensing System

Kevin Bauer and *Hyunyoung Lee*
Department of Computer Science, University of Denver, Denver, Colorado, USA
{kbauer, hlee}@cs.du.edu

ABSTRACT

Wireless sensor technologies are becoming more efficient and smaller in size and as a result, they are becoming more widely used for a variety of applications. However, due to their limited computational, energy, and storage resources, these devices can only perform relatively simple tasks. Furthermore, implementing strong security in sensor networks has often been disregarded because most common cryptographic schemes are too expensive. Nonetheless, sensor networks may often relay critical data, thus, security must be a high priority.

We propose a novel distributed authentication scheme that is efficient and robust using the well-known concepts of "secret sharing" cryptography and group "consensus." We prove that our scheme is well-suited to the resource-deprived nature of networked sensing systems and demonstrate that it requires a minimal message and bit complexity.

1 INTRODUCTION

Wireless sensor networks are becoming increasingly popular for a variety of applications such as monitoring temperatures in forests to detect forest fires or monitoring the movement of an enemy on a battlefield. These sensor nodes typically consist of a primitive CPU, a small quantity of main memory, a flash memory card providing a small amount of secondary storage, a battery and a low power radio transmitter to provide communication capabilities. For example, consider the Berkeley Mica mote [4]: a typical configuration of this sensing device consists of an 8-bit Atmel ATMEGA103 CPU operating at 4 MHz, 4 KB of RAM, 512 KB of flash memory, a 916 MHz radio transmitter providing up to 40 Kbps of bandwidth and a range of a several meters, and two AA batteries. Clearly, these sensing devices must operate under severe resource constraints, minimizing energy consumption and computational and storage demands is of great importance in order to maximize the lifetime (i.e., battery life) of these devices.

Recently, many researchers and developers of sensor node technologies have focused much attention on increasing the efficiency of sensor nodes, specifically with respect to energy consumption. As a result, these devices have become smaller, more powerful, and more efficient. However, as with most systems, security is rarely a top consideration. Such is the case with sensor networks. Given the severe resource limitations, providing a high degree of security in terms of data privacy and authentication is a difficult task and been a low priority. As a result, these networks are vulnerable to a variety of attacks.

Generally speaking, we define data privacy as the degree to which a communication channel between two entities in a wireless network is free of eavesdropping or message corruption. Traditionally, encryption schemes such as RSA or El Gamal have been employed to guarantee a higher level of data privacy. Using these public-key cryptographic schemes, only a party with a valid secret key can decrypt the messages being sent across the message channel and ascertain the contents of the data being transmitted. Also, we generally define authentication as the degree to which one communicating party can ensure the valid identity of another party in the network. Typical authentication methods include the use of a symmetric cryptographic system or the use of digital certificates used by a trusted certification authority. In a traditional wired network, the security problems of data privacy and authentication have been greatly reduced thanks to the use of such tools.

Related Work. In [4], Karloff and Wagner focus specifically on the issue of secure routing in wireless sensor networks. They introduce two classes of attacks on this type of network, sinkholes and HELLO floods and provide a series of countermeasures to protect against such attacks. Many researchers have proposed employing relatively advanced public-key cryptographic protocols to provide a higher degree of data privacy and node authentication. Perrig et al. [5] have developed a set of security protocols for sensor networks that provide authenticated and confidential communications and authenticated broadcast using simple symmetric encryption schemes. This scheme relies upon inflexible sym-

metric cryptography that is perhaps vulnerable to attack by a more computationally sophisticated mobile device such as a laptop computer.

In wireless sensor nodes, due to the severe limitations of computational and energy resources, it is not practical to employ an expensive public-key cryptographic scheme. In addition, even symmetric encryption schemes are not ideal for the simple reason that if the key is recovered through some process of analyzing encrypted broadcast messages, the entire network could be compromised by a malicious attacker using the key to impersonate nodes, disrupt network communication, corrupt the network's data, or perform other malicious activities.

In this paper, we will address the issue of authentication in wireless sensor networks. Our approach differs from other proposed solutions to this problem in that our authentication scheme is completely distributed and does not rely upon expensive cryptographic schemes. Rather, we employ the concepts of "secret sharing" and "group consensus" to provide a scheme that is highly fault tolerant and efficient in terms of computational and message complexity.

The remainder of this paper is organized as follows: In Section 2 we provide a detailed explanation of our assumptions about the system model, based on which we develop our authentication scheme. In Section 3 we describe our distributed authentication algorithms in detail and Section 4 provides an analysis of these algorithms. Finally, we conclude in Section 5.

2 SYSTEM MODEL

In our authentication scheme, we assume that there is a set of N sensor nodes. We define a sensor node as a primitive computing device capable of sending and receiving broadcast messages within a limited transmission radius. In typical sensor networks, communication can be conducted through a fixed infrastructure using a static base station. This base station is analogous to a wireless base station in an 802.11 wireless local area network (WLAN) in that all communication with a wireless device must be provided through this base station. In addition, many sensor networks can support an infrastructure-less ad hoc networking environment where all communication is accomplished through multi-hop routing from sensor to sensor until the destination node is reached. It is also common that these two environments are combined to form a hybrid networking model that allows for centralized communication with a base station and also direct inter-node communication. In our scheme, we assume the following system model:

1. Sensor nodes can be either mobile or static (non-mobile).

2. Our network model relies upon the use of a base station to provide wireless connectivity and inter-node communication (i.e., we do not assume ad hoc communication).

3. The wireless sensor system is partitioned into a subset of $|N|$ nodes, where N denotes the set of sensor nodes. Each subgroup has exactly one base station which is within the broadcast radius of all members of this subgroup.

4. We assume that each subgroup's membership is static, that is to say that the composition of nodes in any arbitrary subgroup is fixed at the time of network construction and it does not add new nodes or delete old ones.

5. Our model relies upon an entity called a Processing Center (PC) which is responsible for coordinating the distribution of secret keys to each node $n \in N$. This Processing Center can be regarded as a completely authenticated and trusted party, similar perhaps to a digital certificate provider.

6. We define a "target" node to be any node in the network (including the base station) that must be authenticated. At any time, any node (typically the base station) that detects particular activities of another node can initiate the authentication procedure against the target node.

Figure 1 illustrates the described model. Using this system model, we have developed a novel authentication scheme that is capable of detecting imposter nodes, i.e., nodes that have forged their identity to join the network. In the next section, we will explain our proposed scheme in greater detail.

We abstractly model a wireless sensor network S by a tuple $\langle N, T, M \rangle$, where N is the set of sensor nodes in the network, $T \in N$ is the "target" node in S, and M is the set of message channels between an arbitrary sensor and the base station. Let $N \backslash \{T\}$ be the set of "trustee" nodes. Our model assumes that there are at most k corrupted (Byzantine faulty) sensor nodes that exist in the network.

We further assume that each node in N has a unique identification (ID) number (i.e., the ID is a unique natural number) that is assigned by the PC at the time when the network is constructed. Let t_i denote the ID of node i. During the time of ID assignment, node t_i becomes aware of its predecessor node t_{i-1} and its successor node t_{i+1} and a logical ring is formed. We assume that if the sensor nodes in this system are mobile, their movement is confined to an area such that they never are out of range of the base station and their neighbor nodes in the

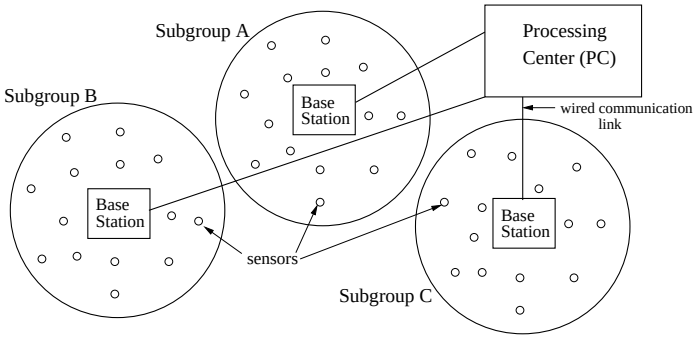

Figure 1: A sensing system model

logical ring structure. Therefore, the logical ring is never broken.

The aforementioned processing center (PC) will be responsible for the distribution of shares of the secret to the set of participating sensor nodes. The PC is responsible for the following:

- At the time of network construction (i.e., when the group of sensor nodes are physically installed in their environment), the PC must provide each sensor with a unique ID number and information about its predecessor and successor nodes.
- Also, the PC maintains a secret for each node in the group. This secret is some arbitrarily agreed-upon data that is known only by the PC and the node corresponding to the secret. No node is aware of any other node's secret and we assume the PC to be trustworthy and secure.
- When the distribution of shares of a node's secret is needed (e.g., when a new target node is chosen and the group must verify its authenticity), the PC computes the shares and distributes them to the set of participating sensor nodes.

3 PROPOSED SCHEME

First, we briefly explain *secret sharing* and *group consensus* to give some background for our authentication scheme. In order to deliver a high degree of sensor authentication, we will use the cryptographic concept of "secret sharing" in which some secret data is shared among a group of "trustee" processes. First developed by Shamir [6] and Blakley [2], cryptographic protocols using secret sharing have the following properties:

1. The secret data is partitioned into a set of distinct shares and distributed to the set of trustees by special process called a "dealer."
2. When necessary, the trustee group can reconstruct the original secret by combining their shares.

In our scheme, the reconstructed secret must be compared to the original secret possessed by the dealer to determine the authenticity of the target node.

In addition, our scheme relies upon the concept of group "consensus." Informally, this is a procedure for getting a group of processes to agree upon a value. In the asynchronous communication model, Fischer, Lynch, and Paterson have proven that it is impossible to achieve group consensus if only one process in the distributed system can fail [3]. Therefore, as one solution to this impossibility result, researchers have proposed adding randomization into consensus algorithms to provide a higher degree of fault tolerance and reliability [1]. A "COIN-FLIP" operation can be incorporated into a consensus algorithm to provide randomized inputs, allowing the consensus algorithm to overcome the impossibility result demonstrated by Fischer, Lynch, and Paterson and achieve group consensus in an asynchronous environment with a high probability. We will incorporate the concepts of secret sharing and group consensus in our distributed authentication scheme.

We discuss now the specifics of our authentication scheme. To initialize our algorithm, a target node must be chosen. Any node in the network (including the base station) can choose another node in the network to be authenticated. Once the target T is chosen, T must notify the PC that it has been chosen and signal the PC to distribute $|N|-1$ shares of its secret to the remaining nodes. The PC sends shares of node T's secret to the remaining group of nodes, whom receive the share and store it locally. This is presented in Algorithm 1.

Once the system has been initialized, the chosen target must be authenticated (see Algorithms 2 and 3). To begin the authentication procedure, we must select a special group that is a subset of the non-target nodes. We will call this group "challengers." When the target node needs to be authenticated, every node u ($u \in N \backslash \{T\}$) se-

Algorithm 1 Initialization Step

T: target node to be authenticated
Distribute_Shares$_{PC}$:
/* Send shares of T's secret to trustees */
  receive $\langle T, |N| \rangle$ from target node T
  /* Query its local table for T's secret */
  secret := SECRET(T)
  forall $t_i \in N \backslash \{T\}$ /* simplified secret sharing */
    /* send i-th share to node t_i */
    send $\langle$ secret$_i$ $\rangle$ to node t_i

Receive_Share$_{t_i}$:
/* $t_i \in N \backslash \{T\}$ receives its share of secret from PC */
  receive $\langle$ secret$_i$ $\rangle$
  share := secret$_i$ /* store share locally */

lects a node v, which is the successor of u, as a challenger with a certain probability p.

If node v is elected into the challenger group by node u, then u will send a broadcast message and v will learn that it has been chosen to act as a challenger. When a node learns that v has been chosen to be a challenger, it adds v to its local set of challengers (which is a local data structure) and sends its share to node v. Once node v receives all of the shares from the group of non-target nodes and the secret s from the target node, it reconstructs the secret s' using the shares and compares the reconstruction to the secret revealed by the target node. If they are equal, the target node is authentic and node v broadcasts a "true" message; else, the target cannot be authenticated (i.e., it is an imposter node) and v broadcasts a "false" message. When a node receives all $|N| - 1$ decisions from the challenger set, it decides by majority rule the authenticity of the target node.

We must guarantee that the recently authenticated target's secret is refreshed, since the secret has just been received by the set of challenger nodes. The Receive_Request_Secret_Message module must be executed after a positive authentication is complete to refresh the target's secret. The target node simply chooses a new secret and sends it to the PC through a secure message channel, utilizing a symmetric encryption scheme whose secret key is unique to the PC and the target node. This authentication scheme can be repeated for a request to authenticate an arbitrary sensor in the network. In order to understand why creating a secure message channel for refreshing an authenticated node's secret is necessary, consider the following scenario: Node a has just been authenticated by an execution of our algorithm. In order for this same node to be authenticated again at a later time, it is necessary to ensure that no nodes other than a and the PC have any knowledge of a's secret. Therefore, a must create a new secret and share it with the PC. However, without a secure message channel, it is very easy for a malicious node within communication range to intercept a's insecure message containing its secret data and use this information to undermine the authentication process in a future execution of the algorithm. Also, with a's secret, any node in the network could easily impersonate node a! So, it is crucial to establish a secure message channel between the node a and the PC using the most inexpensive means possible, such as a symmetric cipher algorithm, for example, RC4 or similar.

Algorithm 2 Authentication – Part I

Select_Challenger$_{u \in N \backslash \{T\}}$:
/* Select node v = successor of u as a */
/* challenger with probability p by flipping coin: */
/* Pr(HEAD) = p and Pr(TAIL) = $1 - p$ */
  if $v \neq T$ and FLIP_COIN() = HEAD then
  /* a non-target node is chosen as a challenger */
    broadcast "v is a challenger" message
    send $\langle$ share, u $\rangle$ to node v
    challengers := challengers $\cup \{v\}$
  /* otherwise do nothing */

Receive_Challenger_Message$_{u \in N \backslash \{T\}}$:
/* Upon u receiving "v is a challenger" message */
/* or other node's share for the first time */
  if $u = v$ then
  /* u learns that it's been chosen as a challenger */
    forall $t_i \in N \backslash \{T\}$
      receive $\langle$ share$_i$, t_i $\rangle$
      retrievedShares[i] := share$_i$
    /* reconstruct secret from received shares */
    secret := FIND_SECRET(retrievedShares)
    send "request secret" message to target T
    receive $\langle$ Tsecret $\rangle$ from T
    if Tsecret = secret then
    /* T is authenticated */
      broadcast "true" message
    else /* Tsecret is incorrect */
    /* T fails to be authenticated */
      broadcast "false" message
  else /* $u \neq v$ */
    send $\langle$ share, u $\rangle$ to node v
    challengers := challengers $\cup \{v\}$

4 ANALYSIS

We assume the total number of nodes $|N|$ is big enough, $|N| - 1 > 2k$, where k denotes the number of unfaithful nodes. Then we need at least $2k + 1$ challengers to

Algorithm 3 Authentication – Part II

```
Receive_Request_Secret_Message_T:
/* Upon the target T receiving "request secret"      */
/* message from v                                    */
   send its secret to v
   send new secret to PC /* need to refresh secret */

Decide_Authenticity_u:
/* Upon u receiving "true" or "false" message        */
   true_counter integer initially 0
   forall v ∈ challengers
      receive ⟨ vote[v], v ⟩
      if vote[v] = "true" then true_counter++
   if true_counter > ½(|N| − 1) then
      decide "T is authentic"
   else
      decide "T is not authentic"

Refresh_Secret_PC:
/* Upon the PC receiving a new secret from T         */
   receive ⟨ Tsecret ⟩ from T
   /* Update its local table entry for T's secret */
   SECRET(T) := Tsecret
   forall t_i ∈ N\{T} /* simplified secret sharing */
      /* send i-th share to node t_i */
      send ⟨ secret_i ⟩ to node t_i

/* Every node t_i ∈ N\{T} starts by the module       */
/* Receive_Share_{t_i}:                              */
```

guarantee that the authentication result is correctly determined by majority rule. We do not want to have a single challenger because then the challenger might be untruthful and tell a lie about the authentication result. We also do not want to have too many challengers because then the scheme will have higher message complexity since every non-target node needs to send its share of secret to every challenger.

Furthermore, by randomizing the selection of challengers, there is very little chance of unfaithful nodes successfully influencing the formation of the challenger set. The probability p should be large enough so that, upon receiving the votes of the members of the challenger group, each node can decide correctly (by majority rule) whether the target is authenticated or not, even when there are k unfaithful nodes (all of which might be included in the challenger group). The following lemma addresses the fairness of the selecting procedure of challengers.

Lemma 1 *Algorithm 2 selects challenger nodes fairly.*

Proof. Recall that each node $t_0, t_1, \ldots, t_{|N|-1} \in N$ is logically arranged in a circular ring structure. For any $t_i \in N\backslash\{T\}$, there exists a probability p that t_i will select its successor in the ring as a challenger, and each node has exactly one predecessor and one successor. Therefore, each node has an equal probability p of being selected as a challenger. ■

The next lemma gives the formula to compute the precise value of p as a function of the numbers of non-target nodes and unfaithful nodes. The given probability ensures that there are enough challengers to reach a majority rule consensus, which can ensure the correctness of the authentication result.

Lemma 2 *If Algorithm 2 selects challengers with a probability p,*

$$p = \frac{1}{a} \cdot \lfloor \log_2 a + 1 \rfloor, \text{ with } a = \frac{|N| - 1}{2k + 1} \geq 1,$$

then a majority rule consensus can be reached based on the values of the challenger group.

Proof. The value of a is the ratio of the number of non-target nodes over the minimum number of nodes to reach a consensus by majority rule. Therefore, the inverse of this ratio, $\frac{1}{a}$, gives the lower-bound of p, i.e., every node selects a challenger with a probability at least $\frac{2k+1}{|N|-1}$ to have the minimum required number of challengers. Thus, the probability p is lower-bounded by $\frac{1}{a}$. Furthermore, $\log_2 a < a$ for $a \geq 1$, thus the probability p is a *slowly* decaying function of a. This is to ensure that there are a good number of challengers chosen, sufficient to reach a majority rule consensus. ■

We computed the probability p as a function of the number $(|N| - 1)$ of non-target nodes using the formula given in Lemma 2 and plotted in Figure 2. Figure 2 (a) is the probability values for each value of $|N|-1$ starting from the smallest possible value (for the given k) up to 20. This is to show concretely how the probabilities vary with an increasing number of non-target nodes. Figure 2 (b) is to show the general trend of logarithmic decay of the probability values over a greater range of non-target nodes.

Let b_{secret} denote the size of a secret and b_{share} the size of each share of a secret, both in bits. The sizes b_{secret} and b_{share} depend on the particular secret sharing scheme that is used in the implementation. Let b_{elect} denote the size (in bits) of the message indicating that a node has been selected for the challenger group and b_{req} the size (also in bits) of the message requesting the target's share. Both b_{elect} and b_{req} can be very small constants.

Theorem 3 *The authentication scheme presented in Algorithms 1, 2 and 3 correctly authenticates a*

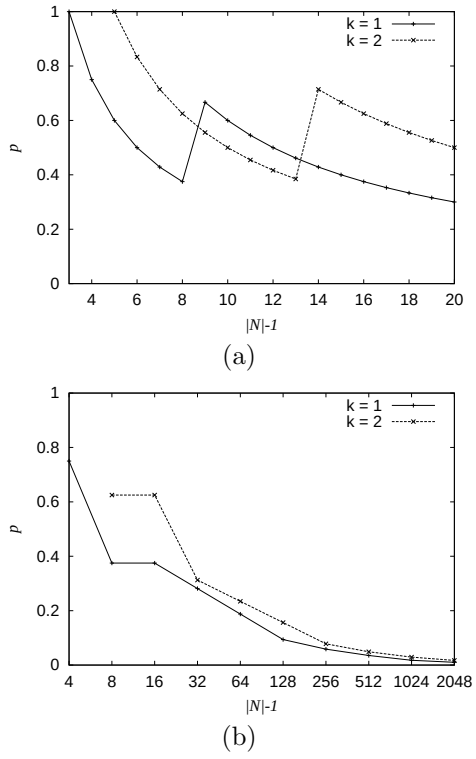

Figure 2: Probability p as a function of the number of nodes

target node in a wireless sensing system using $O(|N|\log_2|N|)$ *secure messages and* $O(\log_2|N|)$ *broadcast messages, and* $\Theta(b_{share}|N|\log_2|N| + (b_{secret} + b_{elect} + b_{req})\log_2|N| + b_{secret})$ *bits in total.*

Proof. The correctness follows from Lemmas 1 and 2. To find the message complexity and bit complexity, let c denote the number of challengers.

In Algorithm 1, to distribute shares, the PC sends $|N|-1$ secure messages, which requires total $b_{share}(|N|-1)$ bits to be transmitted.

In Algorithm 2, there are $c = O(\log_2|N|)$ challengers chosen, each of which is informed via a broadcast message. This will result in $b_{elect}c$ bits total. And each of the c challengers will receive $|N|-2$ secure messages of the shares from every node except the target and itself, therefore, total $b_{share}c(|N|-2)$ bits are sent. After receiving enough shares to compute the secret, each challenger requests the target T it's secret, which is done using $2c$ secure messages and $b_{req}c + b_{secret}c$ total bits. Then each challenger broadcasts its opinion.

In Algorithm 3, the target replies to the challengers' requests by sending its secret to them using c secure messages (which we have already counted in the $2c$ messages in the previous paragraph). Then T must refresh its secret by sending one secure message to the PC of size b_{secret}.

Therefore, in total, there are $|N|(c+1) = O(|N|\log_2|N|)$ secure messages, $c + c = O(\log_2|N|)$ broadcast messages, and $b_{share}(|N|-1) + b_{elect}c + b_{share}c(|N|-2) + b_{req}c + b_{secret}c + b_{secret} = \Theta(b_{share}|N|\log_2|N| + (b_{secret} + b_{elect} + b_{req})\log_2|N| + b_{secret})$ bits needed to complete the authentication. ■

With Shamir's secret sharing scheme, $b_{share} = \Theta(b_{secret})$; thus, in that case, we can simplify the bit complexity in Theorem 3 to $\Theta(b_{secret}|N|\log_2|N|)$.

5 CONCLUSIONS

We have introduced a unique distributed authentication scheme for wireless sensor networks, that is based on the concepts of "secret sharing" and "group consensus." We introduced the notion of "challengers" as a dynamically formed group of sensor nodes that will authenticate a particular target node. Our algorithm chooses nodes to be part of this challenger group using a randomized approach to ensure that there is very little chance of unfaithful (or corrupt) nodes successfully influencing the authentication process. Finally, we proved that our scheme works well in the resource-deprived environment of wireless networked sensing systems. Our scheme requires $O(|N|\log_2|N|)$ secure messages, $O(\log_2|N|)$ broadcast messages, and $\Theta(b_{secret}|N|\log_2|N|)$ bits in total, where b_{secret} denotes the size of a secret and when Shamir's secret sharing scheme is used in the implementation.

Acknowledgments. The research by H.L. was supported by the University of Denver PROF grant 88197. The research by K.B. was supported by the University of Denver PINS grant, Spring 2005.

REFERENCES

[1] J. Aspnes. Randomized protocols for asynchronous consensus. *Distributed Computing*, 16:165–175, 2003.

[2] G.R. Blakley. Safeguarding cryptographic keys. *Proc. of the National Computer Conference*, 48:313–317, 1979.

[3] M.J. Fischer, N.A. Lynch, and M.S. Paterson. Impossibility of distributed consensus with one faulty process. *Journal of the ACM*, 32(2):374–382, 1985.

[4] C. Karlof and D. Wagner. Secure routing in wireless sensor networks: Attacks and countermeasures. *Elsevier's AdHoc Networks Journal, Special Issue on Sensor Network Applications and Protocols*, 1(2–3):293–315, September 2003.

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

[6] A. Shamir. How to share a secret. *Communications of the ACM*, 22:612–613, 1979.