Ad Hoc Networks xxx (2012) xxx–xxx

Contents lists available at SciVerse ScienceDirect

Ad Hoc Networks

journal homepage: www.elsevier.com/locate/adhoc

A secure broadcasting scheme to provide availability, reliability and authentication for wireless sensor networks

E. Ayday *, F. Fekri

School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332, USA

ARTICLE INFO

Article history:
Received 26 December 2010
Accepted 29 March 2012
Available online xxxx

Keywords:
Broadcast authentication
Security
Reliability
Sensor networks

ABSTRACT

Reliability and security of broadcasting is critical in Wireless Sensor Networks (WSNs). Since reliability and security compete for the same resources, we are interested in jointly solving for error control coding (to achieve reliability) and integrity for a broadcast scenario. We assume Byzantine attacks in which the adversary can compromise nodes and then drop (or modify) the legitimate packets or inject its own packets. For reliable and efficient multihop broadcasting, it is critical to reduce the energy consumption and latency. To prevent the adversary from consuming the scarce network resources by injecting bogus packets, each receiver node should make sure that packets it receives are authentic and it filters out malicious packets immediately. We build our authentication scheme, on top of a reliable and energy efficient broadcasting protocol called Collaborative Rateless Broadcast (CRBcast) to improve efficiency and reliability. On contrary to the previous schemes, our scheme is resilient with respect to Byzantine adversary as well as routing and flooding attacks and protocol exploits. Moreover, we compared our scheme with the previously proposed broadcast authentication schemes and showed that our scheme outperforms them in terms of efficiency and data availability. This is a crucial improvement over the previous schemes that ensure availability by flooding, introducing very large communication overhead and latency.

1. Introduction

Wireless Sensor Networks (WSNs) are expected to play key roles in many applications, such as managing energy plants, logistics and inventory, battlefields, and medical monitoring to name a few [1]. A sensor is a low cost and low power device with limited computational power and memory that is equipped with sensing and radio transmission units. A typical sensor network may include hundreds to several thousands of sensor nodes. WSNs are often infrastructureless and may be deployed randomly. Therefore, the security of WSNs poses new challenges because of the node constraints and networking features.

Reliable and time critical broadcasting is needed in many important applications in WSNs. Examples are updating the software in all the network nodes after their deployment, reactive routing protocols (where route query packets are forwarded to all nodes in the network), and broadcasting streaming media. In addition to reliability, a broadcasting protocol must be energy efficient especially for battery-powered wireless networks. Therefore, finding a reliable and energy-efficient broadcasting scheme for multihop WSNs is of great interest.

The security of broadcast data transfer becomes very important especially for the networks that are deployed in hostile areas in which there is high probability of node compromise and adversary initiates serious attacks against the network by compromising a few nodes of the network. In this paper, we consider Byzantine attackers (insider adversarial nodes with the same authority as any other

* Corresponding author. Tel.: +1 404 518 7037; fax: +1 404 894 8363.
E-mail addresses: eayday@gatech.edu (E. Ayday), fekri@ece.gatech.edu (F. Fekri).

1570-8705/$ - see front matter © 2012 Published by Elsevier B.V.
http://dx.doi.org/10.1016/j.adhoc.2012.03.010

legitimate node). Hence, the typical attacks by the adversary can be listed as below:

- *Data Drop:* An insider malicious node drops legitimate reports on the forwarding path toward the sink.
- *Bogus Packet Injection* and *Packet Modification:* The adversary injects bogus packets or modifies the contents of legitimate reports.

Cryptographic services required to prevent these attacks are *data availability* and *data authenticity*, respectively.

In this paper, we reintroduce the *Authenticated Collaborative Rateless Broadcast* (AuCRB) of [2] for WSNs with extended simulations and thread analysis. We propose an efficient multihop broadcast authentication scheme for WSNs. The proposed scheme provides data authenticity and availability with low communication and computation overheads. This scheme employs node collaboration to provide data authenticity and rateless information delivery mechanism [3,4] to provide data availability. We build our authentication scheme, on top of a reliable and energy efficient broadcasting protocol called Collaborative Rateless Broadcast (CRBcast) [5] to improve efficiency and reliability. CRBcast [5] is a reliable and energy efficient node-to-network broadcasting protocol for multihop wireless networks. The motivation for using CRBcast is that it is shown to be 72% more efficient than flooding in terms of energy efficiency [5]. As opposed to previously proposed schemes, we integrated both authentication and broadcast in the designation of the AuCRB. The advantages of this approach are significant improvements in efficiency and data availability. We consider the case that a large amount of packets (of order 1000 or more) have to be broadcasted in a multihop WSN while requiring robustness against Byzantine attacks, reliability, energy-efficiency and low latency.

The organization of the paper is as follows. In the rest of this section, we summarize the related work in broadcast authentication and present the notation used throughout the paper. In Section 2, we briefly review the mathematical and cryptographic primitives used in this paper. The detailed description of AuCRB is provided in Section 3. We analyze the security of our scheme against possible attacks in Section 4. The performance analysis of the proposed scheme and comparison to related work are studied in Section 5. Finally, the concluding remarks are provided in Section 6.

1.1. Related work

Based on the main cryptographic primitives employed, we can classify previously proposed broadcast authentication schemes into three groups based on the main cryptographic primitive employed: (1) message authentication code ($\mathbb{MAC}$), (2) signature amortization, and (3) one-time signature.

Protocols in the first group are TESLA [6], its simplified version for resource limited networks μTESLA [7], and the enhancements of μTESLA such as [8]. These schemes provide broadcast authentication by using $\mathbb{MAC}$s and require time synchronization between the nodes and the sink. This requirement is an implementation hurdle for multihop broadcasting in densely deployed networks. Even if the time synchronization is established between the nodes, the nodes can easily become asynchronous in time. Moreover, since the disclosure of the authentication information is done in delayed time intervals, packets have to be buffered at the receivers. Hence, there is a delayed authentication problem which causes this type of schemes to be vulnerable to flooding (or denial or service) attack. A recently proposed scheme [9] introduced a mechanism called "message specific puzzle" to mitigate such flooding attacks. However, this technique works under the assumption of a computationally powerful source node which is not suitable for node-to-network broadcasting in resource constrained WSNs. Another shortcoming of μTESLA is the difficulty of establishing the initial trust between the nodes and the sink.

Schemes in the second group employ signature amortization. One of the first protocols in this group is SAIDA [10]. This protocol is not robust against false packet injection and packet modification attacks since the receivers may not be able to decode for the signature in case of a single packet modification or noisy packet injection. The designers of SAIDA have proposed using Reed Solomon codes [11] to handle the packet modification attack. However, this kind of coding is too complex for wireless devices with low-power processors. A similar approach is proposed in [12], which is too complex for multihop broadcasting in WSNs. In [13], authors improved SAIDA by using Merkle hash tree to detect packet injections and modifications. However, since the hash tree is generated using plaintext packets, an adversary can easily capture all the packets, modify them and generate a legitimate hash tree using the modified packets.

One-time signature based schemes BiBa [14], an improvement of BiBa, called HORS [15], and an extension of HORS [16] are among the schemes in the third group. The major drawback of using one-time signature based schemes is that the public key has to be frequently updated to maintain security. This requirement significantly adds to the communication overhead of the protocol. Moreover, broadcast authentication schemes based on one-time signatures are not suitable for designing node-to-network multihop broadcast protocols. Broadcast communications of any node has to be handled by the sink as an intermediary.

A recently proposed scheme [17] introduces a simple method to secure the deluge network programming. In [17], authors create a hash chain to authenticate the packets. In Section 5, we compared the efficiency of AuCRB with this scheme and showed that our scheme has considerable advantages over this recent approach.

To sum up, we claim that neither of the previous schemes consider data availability in an efficient fashion. Although all these schemes are focused on efficient authentication, the communication efficiency and the data availability are largely ignored. Hence, in this work, we include both data availability and authenticity to design an efficient scheme. Moreover, latency has never been considered by the previous works when defining the data availability. However, we define data availability based on the latency, which is a crucial requirement.

Q1 Please cite this article in press as: E. Ayday, F. Fekri, A secure broadcasting scheme to provide availability, reliability and authentication for wireless sensor networks, Ad Hoc Netw. (2012), http://dx.doi.org/10.1016/j.adhoc.2012.03.010

1.2. Contributions of this paper

The main contributions of our scheme are summarized in the following:

1. AuCRB is designed based on a broadcast protocol using rateless coding. Hence, it benefits from the same low computation and communication overheads.
2. Nodes individually authenticate each received packet instead of waiting for several packets to perform authentication. Therefore, the receivers can immediately filter out bogus packets and save energy.
3. Rateless coding intrinsically provides data availability by the loss recovery of the coding mechanism.
4. Authentication information transmitted by the source can be used to detect malicious nodes in the network.
5. The scheme ensures availability of data with very low latency in the presence of the malicious nodes (as long as the network is connected). This is a substantial improvement upon similar schemes that ensure availability by flooding but with very large communication overhead and latency.

1.3. Notations

In order to facilitate future references, frequently used notations are listed below.

N	total number of nodes in the network
p	probability of forwarding in *phase I* of AuCRB
t	number of data packets to be sent from the source node
T	number of encoded packets generated after rateless encoding
ℓ	number of partitions in *phase II* of AuCRB
P_i	the ith encoded packet during *phase I* of AuCRB
Q_i	the ith encoded packet during *phase II* of AuCRB
G_i	the ith partition during *phase II* of AuCRB

2. Technical background in context

2.1. Bloom filter

A Bloom filter is a simple space-efficient randomized data structure for representing a set in order to support membership queries [18]. A Bloom filter for representing a set U of T elements is described by an array of m bits, initially all set to 0. It employs k independent hash functions $H_1, \ldots, H_k$ with range $\{1, \ldots, m\}$. For every element $x \in U$, the bits $H_1(x), \ldots, H_k(x)$ in the array are set to 1. A location can be set to 1 multiple times, but only the first change has an effect. To check if y belongs to U, we check whether all $H_1(y), \ldots, H_k(y)$ are set to 1. If not, y definitely does not belong to U. Otherwise, we assume $y \in U$ although this may be wrong with some probability. Hence, a Bloom filter may yield a *false positive* where it suggests that y is in U even though it is not.

The probability of false positive is an important parameter in a Bloom filter. After all elements of U are hashed into the filter, the probability that a specific bit is 0 is

$$\left(1-\frac{1}{m}\right)^{kT} \approx e^{-kT/m}. \quad (1)$$

Hence, the probability of false positive is

$$\tilde{p} = \left(1-\left(1-\frac{1}{m}\right)^{kT}\right)^{k} \approx (1-e^{-kT/m})^{k}. \quad (2)$$

As an example, assume the number of elements in U is $T = 100$. If $k = 5$ and the desired probability of false positive is $\tilde{p} = 0.01$, then the length of the filter must be $m = 985$ bits. If we decrease the probability of false positive to $\tilde{p} = 0.001$ and increase the number of hash functions to $k = 10$, the size of the filter becomes $m = 1438$ bits.

2.2. Elliptic Curve DSA (ECDSA)

ECDSA is a variant of the Digital Signature Algorithm (DSA) which operates on elliptic curve groups. The Elliptic curve variant provides smaller key sizes than DSA for the same security level. On the other hand, the execution time is roughly the same and the signature size is exactly the same. Once the sender and the receiver are agreed upon the elliptic curve parameters, the sender signs the message using its private key d_A and receiver verifies the signature by using the public key of the sender Q_A.

3. AuCRB description

The proposed AuCRB has three phases to decrease the number of transmissions and the total latency. In *phase 0*, the original packets are first encoded using a rateless coding mechanism [4,3]. Authentication information is then generated by the source from the encoded packets. This authentication information is then broadcasted to the network via flooding. In *phase I*, the encoded packets are broadcasted using a simple and scalable probabilistic relaying scheme referred to as PBcast. In PBcast, a node rebroadcasts the packets that it has received for the first time to its neighbors (two nodes are called neighbors if they are within the communication range of each other) with some probability $p \leqslant 1$. In *phase II*, the nodes which received sufficient number of packets to decode for the original data during *phase I* help their neighbors which still need packets to retrieve for the original data.

In this work, we also take the latency into account and define the data availability based on the latency. Hence, throughout our studies, we consider a Medium Access Control (MAC) protocol in AuCRB as in [19] to compute the latency of the network and to obtain more realistic results. In the following subsections, we explain the three phases of AuCRB in detail. All three phases are simply illustrated in Fig. 1.[1]

3.1. Phase 0

Phase 0 consists of two steps: (1) generating the report and encoding the data packets at the source, and (2)

[1] In Fig. 1, BO and HO are the outputs of the Bloom filter and the cryptographic hash function, respectively. The source node generates BO and HO to generate the authentication information, as will be discussed later in Section 3.1.

Q1 Please cite this article in press as: E. Ayday, F. Fekri, A secure broadcasting scheme to provide availability, reliability and authentication for wireless sensor networks, Ad Hoc Netw. (2012), http://dx.doi.org/10.1016/j.adhoc.2012.03.010

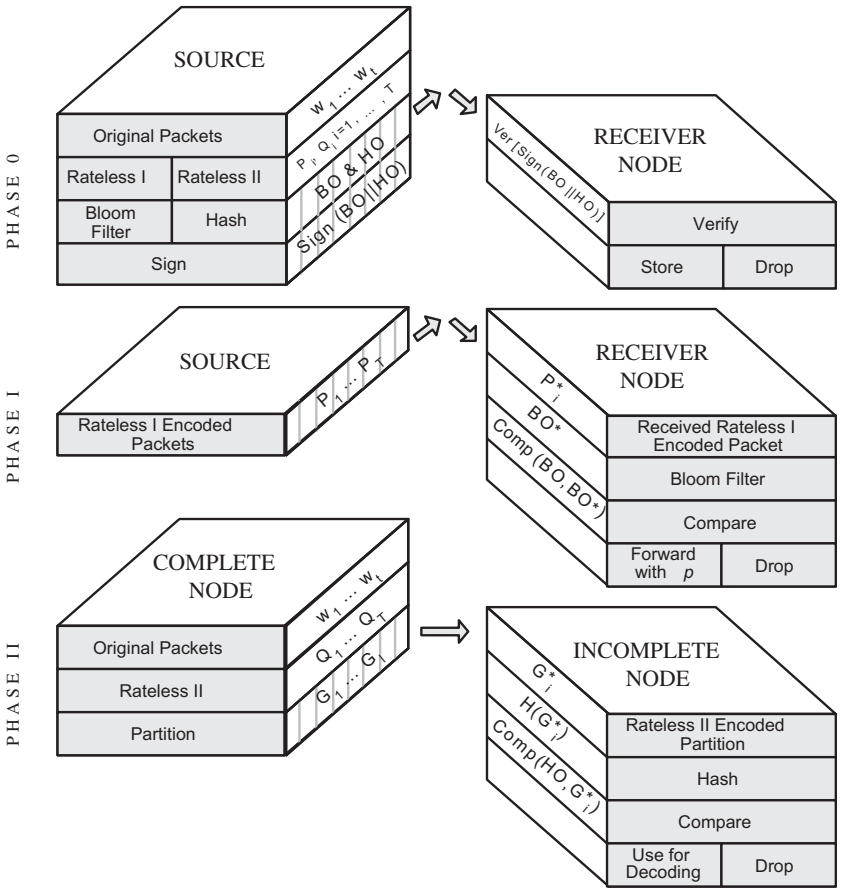


Fig. 1. Authenticated Collaborative Rateless Broadcast.

generating and delivering the authentication information. In AuCRB, using two instances of rateless coding with different parameters, the source node generates two sets of encoded packets. Sets of encoded packets generated by *RatelessI* and *RatelessII* are used in *phase I* and *phase II*, respectively. In the first instance, the linear coefficients of *RatelessI* are randomly driven from an optimized distribution [4]. The linear coefficients employed in *RatelessII* are generated using a pseudorandom function based on an optimized distribution that is known to all nodes. We assume that all nodes have access to the same pseudorandom function and employ the same seed[2] to generate random coefficients. Hence, using *RatelessII*, all nodes generate the same set of coefficients. In the following, we provide the details of *phase 0*.

Upon obtaining information critical to the entire network, a source node generates the t packets $w_1, \ldots, w_t$, and then it constructs the encoded packets $P_1, \ldots, P_T$ via *RatelessI* as illustrated in Fig. 2, where $T = t\gamma$ and $\gamma > 1$. The encoding process can be summarized as follows:

1. Pick a generator polynomial $\Omega(x) = \sum_{1 \leqslant i \leqslant t} \Omega_i x^i$ where $\Omega_i \in [0,1]$ for $i = 1, \ldots, t$ and $\Omega(1) = 1$.
2. Generate an instance, say z, of a random variable Z with distribution Ω.
3. Pick z distinct packets at random from the input t data packets.
4. XOR the selected packets and declare as the encoded packet.

The desirable feature of rateless coding is that the reception of any $t\gamma$ (γ is around 1.05 for $t = 1000$[3]) encoded packets suffices to decode for the original message. Although one of the advantages of our scheme is to provide reliability in the presence of lossy packets using its built in rateless coding mechanism, we considered lossless links between any two neighboring nodes for the simplicity of discussion and simulations, and hence, we set $T = t\gamma$.

The source generates authentication information partially using a Bloom filter. The Bloom filter takes the encoded packets $P_1, \ldots, P_T$ as inputs and employs k independent hash functions $H_1, \ldots, H_k$. The output of the Bloom filter, an array M of bit length m, forms a piece of the authentication information. Bloom filter process for the packet P_1 is illustrated in Fig. 3. Another piece of the

[2] The seed is updated after every broadcast session.

[3] It can be shown that when the decoder receives $1000(1 + \zeta_{1000})$ packets, where ζ_{1000} is a positive number very close to zero, it can successfully decode all 1000 input packets with high probability [4].

Q1 Please cite this article in press as: E. Ayday, F. Fekri, A secure broadcasting scheme to provide availability, reliability and authentication for wireless sensor networks, Ad Hoc Netw. (2012), http://dx.doi.org/10.1016/j.adhoc.2012.03.010

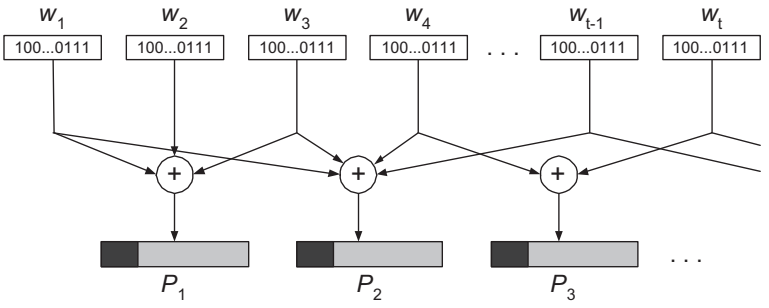


Fig. 2. Illustration of encoding of rateless codes.

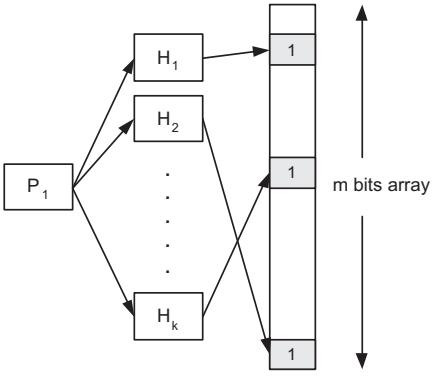


Fig. 3. Process of setting up the Bloom filter for the packet P_1.

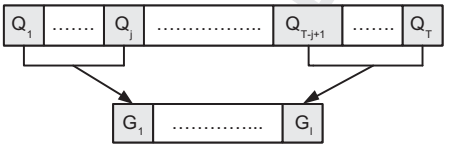


Fig. 4. Partitioning packets in *phase II* of AuCRB.

authentication information belongs to *phase II*. In *phase II*, the encoded packets are generated from the original data known to the source. Moreover, all complete nodes generate the same set of encoded packets using *RatelessII*. Therefore, the source generates authentication information for *phase II* as well. Let $Q_1, \ldots, Q_T$ be the encoded packets generated using *RatelessII*.[4] These packets are partitioned into ℓgroups $G_1, \ldots, G_\ell$ as illustrated in Fig. 4. Assuming that $j = T/\ell$ is an integer, the ℓgroups are related to the encoded packets as follows.

$$G_i = [Q_{1+(i-1)j}, \ldots, Q_{ij}], \quad \forall i = 1, \ldots, \ell. \tag{3}$$

Next, the source compiles the authentication information required for both phases as

$$A = ID \| M \| H(G_1) \| \cdots \| H(G_\ell), \tag{4}$$

where $H(\cdot)$ is a cryptographically secure hash function, M is the output of the Bloom filter, and ID is the identity of the source node.

To prevent an adversary from modifying the authentication information, the source signs A using an efficient signature scheme $Sign(\cdot)$ enhanced for use in resource constrained wireless networks [20]. For this purpose, we use the Elliptic Curve DSA (ECDSA) [21] algorithm illustrated in Algorithm 1 and obtain the signature as $Sign(A) = [r,s]$. Eventually, the source broadcasts the authentication information $(A, Sign(A), Ver)$ to entire network. Here, *Ver* is the description of the signature verification algorithm. Every node receiving the authentication information, first verifies its integrity as illustrated in Algorithm 2 (refer to [21] for details). If it is verified (the output of Algorithm 2 is *true*), the node then broadcasts it with probability one to its neighbors. We note that nodes do not forward any data packets before receiving a valid authentication information from the source (they can store the received packets in their memory).

Algorithm 1. Signature generation

INPUT: Authentication information to be signed, A and private key of the source d_A
OUTPUT: Signature pair $[r,s]$
1 Calculate the hash of the authentication information $e = H(A)$.
2 Select an integer k randomly from $[1, n-1]$.
3 Calculate $(x_1, y_1) = kG$ (where G is a base point of prime order on the elliptic curve) and set $r = x_1 (mod\ n)$ ($r \neq 0$, if $r = 0 \Rightarrow$ repeat 2nd step).
4 Calculate $s = k^{-1}(g + d_A r)(mod\ n)$, where g is the L_n leftmost bits of e ($s \neq 0$, if $s = 0 \Rightarrow$ repeat 2nd step).
5 Signature pair is $[r,s]$.

For simplicity, we assume that every node has access to the algorithm for verifying the integrity of the authentication information. In other words, every node has the necessary information to execute *Ver* (i.e., every node has the public key of the other nodes in the network). It may be argued that this assumption limits the scalability of the network. However, there are efficient techniques to deliver the public key of the source to the other nodes just when it is necessary. We list the most common techniques to deliver the public key of the source as follows:

[4] Generation of *RatelessII* encoded packets is similar to the process described before for *RatelessI*. However, the coefficients employed in *RatelessII* are generated using a pseudorandom function based on an optimized distribution.

- Public key of the source is delivered to other nodes by a central authority at the beginning of the communication session.
- Each node is initially installed with a certificate signed by the central authority. Hence, when the source sends its public key and public key certificate attached to the authentication information, each receiver node verifies the public key of the source by just storing the public key of the central authority.
- Identity-Based Cryptography (IBC) or Hierarchical Identity-Based Cryptography (HIBC) schemes are used as proposed in [22–24] in which the verification algorithm is obtained from the ID of the source node generating the signature.

We do not analyze these methods in detail as it is not the main focus of this paper.

Algorithm 2. Signature verification

INPUT: $[r,s]$, public key of the source Q_A and authentication information A
OUTPUT: *true* or *false*
1 Verify that $[r,s]$ are integers in the interval $[1, n-1]$.
2 Calculate hash for received authentication information $e = H(A)$.
3 Calculate $\upsilon = s^{-1}(mod\ n)$.
4 Calculate $u_1 = e\upsilon(mod\ n)$ and $u_2 = r\upsilon(mod\ n)$.
5 Calculate $(x_1,y_1) = u_1G + u_2Q_A$.
6 $x_1 = r(mod\ n) \Rightarrow$ *true*, otherwise *false*.

3.2. Phase I

In this phase, *RatelessI* encoded packets generated during *phase 0* are broadcasted to the network. When a node receives a packet encoded with *RatelessI*, it initially verifies the authenticity of the packet using the Bloom filter. Every packet authenticated by a node is forwarded with probability p to its neighbors. If a packet cannot pass the authentication test, it is dropped and the forwarding node is detected by its legitimate neighbors. Hence, a legitimate node can take action against its malicious neighbors to increase the data availability of the network. This will be explained in Section 4.

During *phase I*, some nodes receive sufficient number of packets to decode for the original data. These nodes are called the "complete nodes". Complete nodes decode the received packets to obtain the original data $w_1, \ldots, w_t$ as illustrated in Fig. 5. We summarize the decoding process as follows:

1. Wait until $t\gamma$ packets are received.
2. Construct the Tanner graph between the encoded and the data packets.
3. Use Message Passing Decoding [25,26] to recover data packets.

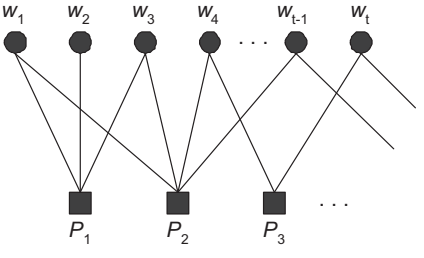


Fig. 5. Illustration of decoding of rateless codes.

3.3. Phase II

Nodes that cannot receive sufficient number of packets to decode for the original data during *phase I* are called the "incomplete nodes". In *phase II*, incomplete nodes try to receive the sufficient number of packets from their one-hop complete neighbors for successful decoding. Complete nodes advertise their completeness to their neighbors by broadcasting advertisement ADV messages. An incomplete node receiving ADV, responds with a request message REQ that includes the number of requested packets and the ID of the complete node. This ADV–REQ exchange between a complete node C and its incomplete neighbors is illustrated in Fig. 6a and b. A complete node, after receiving all the REQ messages from its neighbors, generates encoded packets $Q_1, \ldots, Q_T$ from the original data using *RatelessII*. A complete node partitions these packets into groups $G_1, \ldots, G_\ell$ of almost equal sizes. This operation is demonstrated in Fig. 4. As we explained in Section 3.1, the partitioning technique enhances the generation of authentication information. Complete nodes, instead of the sending the encoded packets $Q_1, \ldots, Q_T$ one by one, send groups $G_1, \ldots, G_\ell$ to their incomplete neighbors. The number of transmitted packets from a complete node is equal to the maximum number of requested packets from its incomplete neighbors. Let c be the maximum number of packets requested from a complete node. This node broadcasts $G_1, \ldots, G_s$, where $s = \lceil (c/j) \rceil$. After several iterations of the above process, all the incomplete nodes in the network recover the source data.

Since an incomplete node receives its requested packets from a specific complete node, it is not required to verify the authenticity of packets individually. Hence, an incomplete node authenticates the packets received from a complete neighbor as a group. In other words, using the authentication information A, an incomplete node verifies the authenticity of the received packet groups. If the authentication fails, all packets are discarded and the complete node is detected as a malicious node. In this case, the incomplete node must wait until an ADV message from a legitimate complete neighbor is arrived.

4. Threat analysis

In this section, we analyze the security of AuCRB in terms of data authenticity and node impersonation. We note that the proposed scheme is not vulnerable to the routing attacks via colluding Byzantine nodes since it does not use any fixed routing (nodes broadcast the packets to their neighbors). In the proposed scheme, every node can

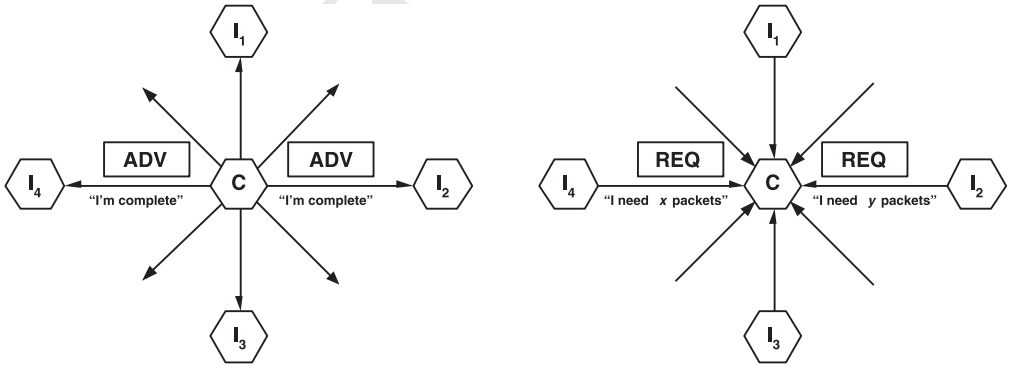
Q1 Please cite this article in press as: E. Ayday, F. Fekri, A secure broadcasting scheme to provide availability, reliability and authentication for wireless sensor networks, Ad Hoc Netw. (2012), http://dx.doi.org/10.1016/j.adhoc.2012.03.010

potentially be a broadcasting source. Providing the broadcast ability to every node allows an adversary to send its own data to the network by just compromising a single node. This is a common drawback of all such schemes that allow every node in the network to broadcast. A solution is requiring a message to be generated by the collaboration of several local nodes using a secret sharing scheme [27]. This is not within the scope of our paper.

Assuming the legitimacy of the source node, the adversary cannot modify the report at its generation time. Moreover, adversary cannot cheat receivers by modifying the message since authentication information is provided and digitally signed by the source node. A bogus packet injected during *phase I* is filtered out with a high probability after one hop travel. The filtering strength of the *phase I* depends on the false positive probability of the Bloom filter. The network designer can arbitrarily decrease this probability to the expense of increasing communication overhead as explained in Section 2.1. We note that the false positive probability can be significantly reduced by using recently proposed techniques such as [28]. Similarly, in *phase II*, an incomplete node uses the signed authentication information to authenticate the group of packets. Hence, it is not possible for an adversary to inject malicious packets without being detected. Therefore, when an adversary either attempts to inject bogus packets to the network or to drop legitimate packets in either of the two phases of the protocol, its major impact is increased latency. This increase in latency due to adversary can be attributed to two factors. (1) The number of legitimate nodes who supply packets to the network reduces (hence, the packets may travel longer hops), and (2) The waiting time to access the channel by the legitimate nodes increases. In other words, malicious nodes may compete via the MAC protocol to access the channel and inject packets or remain silent.

To combat attacks at the MAC layer, we use the fact that a malicious node sending bogus packets are detected by its legitimate neighbors who keep a list of the sensor nodes that they detected as malicious. Due to the MAC protocol [19], a sensor node gains access to the channel after exchanging RTS (request to send) and CTS (clear to send) messages with its neighbors. This RTS–CTS exchange is illustrated in Fig. 7a and b. Hence, when a legitimate node receives a bogus packet from one of its neighbors, the receiver will no longer accept any packets from that specific node. In other words, when it hears an RTS from the detected node, it will not respond with a CTS. As a result, a legitimate node can get the channel instead of the malicious one. We note that in our scheme every node is able to detect malicious nodes individually without using expensive and vulnerable voting systems.

Another possible attack is that a malicious node may send bogus packets by impersonating another legitimate node. As a result of this attack, the impersonated legitimate node will be marked as malicious by its neighbors. Even though it is not the main contribution of our paper, AuCRB can prevent the node impersonation attack by assuming that nodes have pairwise keys with their neighbors by using a key pre-distribution scheme such as the one in [29]. Due to the initial key pre-distribution, all nodes in the network know their one-hop neighbors, and hence, a node does not communicate with a node that is not its one-hop neighbor (and, hence, a malicious node cannot impersonate a legitimate node that is more than two-hop from itself). Moreover, as a result of the MAC protocol [19], one and two-hop neighbors of a node (that has access to the channel) know which node is occupying the channel due to the RTS–CTS message exchange as illustrated in Fig. 7a and b. Thus, if a malicious node impersonates one of its one or two-hop neighbors and sends RTS to the neighbors of the impersonated node, the impersonated node will learn about this due to either RTS (if it is a one-hop neighbor of the malicious node) or CTS (if it is a two-hop neighbor of the malicious node). As soon as a legitimate node realizes that it is being impersonated by a malicious node, it sends warning messages to its neighbors encrypted by its pairwise keys. Hence, the impersonation attack is easily detected with a cost of a few more transmissions. This is illustrated in Fig. 8. Here, node E is the legitimate node with neighbors 1–5. When the malicious node $\widehat{E}$ impersonates E and sends RTS (to nodes 3, 5, 6 and 7), E will also learn that a node occupying the channel is using its ID as soon as nodes 3 or 5 sends back a CTS to $\widehat{E}$. Hence, E will send encrypted warning messages to its

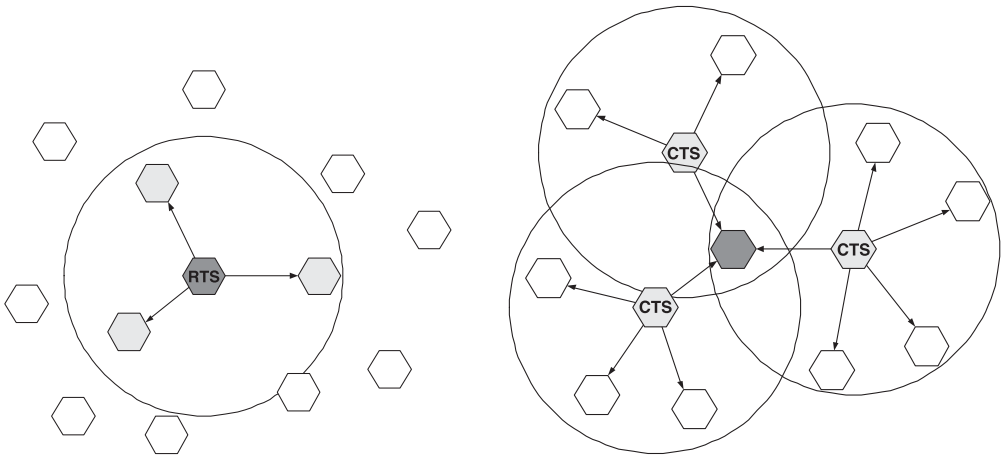


(a) Advertisement (ADV) message from the complete node C to its incomplete neighbors.

(b) Request (REQ) messages from the incomplete nodes to their complete neighbor C.

Fig. 6. Exchange of ADV and REQ messages between the complete node C and its incomplete neighbors.

Q1 Please cite this article in press as: E. Ayday, F. Fekri, A secure broadcasting scheme to provide availability, reliability and authentication for wireless sensor networks, Ad Hoc Netw. (2012), http://dx.doi.org/10.1016/j.adhoc.2012.03.010



(a) Request to send (RTS) message from a sensor node to its neighbors to gain access to the channel.

(b) Clear to send (CTS) messages from the neighbors of the node that want to gain access to the channel.

Fig. 7. RTS–CTS message exchange between the sensor nodes.

neighbors. As nodes 3 and 5 get these warnings, they remove E from their *detected adversary* list. We note that since nodes 6 and 7 are not in one-hop neighborhood on E, they are not affected from this attack (they do not communicate with a node that is not their one-hop neighbor).

5. Performance analysis and comparison to related work

We obtained our results considering the characteristics of MICA2DOT with 4 MHz 8-bit processor using Chipcon CC100 antenna. We assume the nodes are uniformly and randomly deployed in the field. In addition, they form a connected stationary network. We further assume the existence of an underlying MAC protocol for the channel access as in [19]. As a result of the MAC protocol, a sensor node gains access to the channel after exchanging RTS–CTS messages with its neighbors as illustrated in Fig. 7a and b. Further, due to this RTS–CTS exchange, one and two-hop neighbors of a node (that has access to the channel) know which node is occupying the channel, and hence, they remain silent throughout that communication session (for details refer to [19]). Finally, as we discussed before, we consider lossless links between any two neighboring nodes for the simplicity of discussion and simulations.

We compare our scheme with a recently proposed broadcast authentication scheme in [17]. The rational for choosing Secure Deluge in [17] for comparison is that all previous schemes on broadcast authentication either have vulnerabilities to certain attacks or they are impractical for implementation as we discussed in Section 1.1. Even though Secure Deluge is also vulnerable to flooding attack, its implementation is easy and it does not require too much extra overhead. Besides, all broadcasting schemes, which guarantee availability, use the flooding technique to transmit the data packets. Hence, they have more or less similar overhead as in [17]. Throughout this section, we assume that the fraction of compromised nodes in the entire network is C and the network is connected unless otherwise stated.

5.1. Data availability

As opposed to previous works, we define the data availability based on the latency. In other words, for 100% availability, all nodes in the network need to become complete in a definite time.

Using computer simulations (via MATLAB), we have studied data availability in ensemble of 50 networks. In our simulations, we have assumed $N = 1000$ nodes, $T = 1000$ packets, the payload size of each packet is 64 byte, the transmission range $r = 0.2$ units, size of the deployment field is 2×2[5] unit square, and the average degree of a node is 30 for connectivity. We note that there is no motive to select these simulation parameters. We simulated the proposed scheme with different simulation parameters (for different network models) and obtained similar results.

As we mentioned before, for a reliable and efficient multihop broadcasting, it is critical to reduce the energy consumption and latency. Thus, we consider both the energy consumption of the network (due to the number of packet transmissions) and the latency versus the forwarding probability p in *phase I*.[6] Therefore, we created the energy consumption-latency metric as

$$metric = \frac{N_{tx}(i)}{min(N_{tx})} \times \frac{latency(i)}{min(latency)}, \quad (5)$$

where $N_{tx}(i)$ is the number of transmissions per node for $p = i$ (where $0 \leqslant i \leqslant 1$), $min(N_{tx})$ is the minimum number of transmissions per node among all p values, $latency(i)$ is the total time required for all nodes to become complete for $p = i$, and $min(latency)$ is the minimum latency value ob-

[5] Length measurements are normalized to the unit of measurement.

[6] Varying forwarding probability p determines the efficiency of the scheme.

Q1 Please cite this article in press as: E. Ayday, F. Fekri, A secure broadcasting scheme to provide availability, reliability and authentication for wireless sensor networks, Ad Hoc Netw. (2012), http://dx.doi.org/10.1016/j.adhoc.2012.03.010

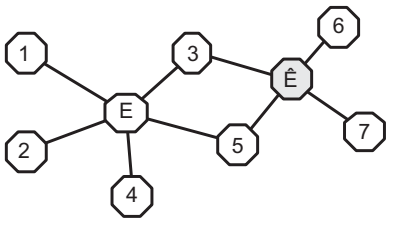


Fig. 8. Node impersonation attack.

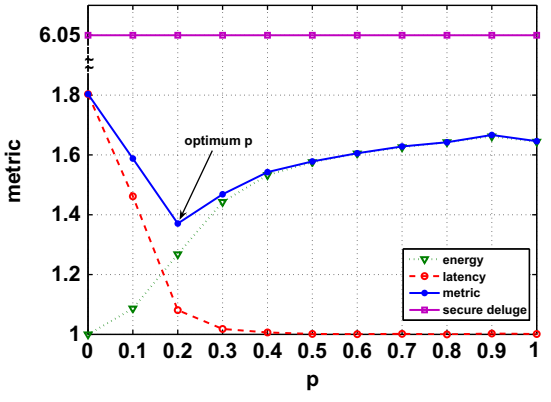


Fig. 9. Energy consumption-latency metric versus p (without any adversary).

tained from all p values. We assume that the transmission time for one packet is equivalent to one time-unit.

In Fig. 9, we illustrate variance of the energy consumption-latency metric with p for 100% data availability. In the figure, energy stands for $N_{tx}(i)/min(N_{tx})$ and latency stands for $latency(i)/min(latency)$. Hence, we illustrate the optimal value for p at which the scheme works the most efficiently in terms of both the energy consumption and the latency when there is no adversary ($C = 0$). We further observed that energy consumption increases and latency decreases with increasing p as expected. We simulated the same network for [17] and obtained the above metric as 6.0679. This is more than 4 times larger than our scheme with optimum p and more than 3 times larger than our worst case p. Hence, we conclude that our scheme outperforms all previous schemes that use flooding technique as in [17] in terms of the energy consumption-latency metric.

We also evaluated and compared the performance of both our scheme and [17] in adversarial environments where malicious nodes either drop or modify legitimate packets. We assume that during *phase I* of AuCRB, a malicious node forwards a packet with probability p_{forw} and modifies or injects a bogus packet with probability p_{mod}. In Fig. 10, we illustrate the impact of adversary to the energy consumption-latency metric for different values of p_{forw} and p_{mod} during a single broadcast session when the fraction of compromised nodes $C = 10\%$ (we obtained similar results for different C values). In AuCRB, since the malicious nodes that modify the legitimate packets or inject bogus packets are detected, we observed that the adversary gives the most serious damage when it modifies

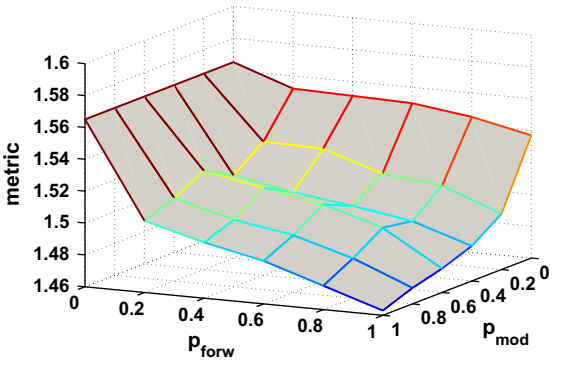


Fig. 10. Impact of adversary for different p_{forw} and p_{mod} values when $C = 10\%$.

or injects bogus packets only during *phase II* (when $p_{forw} = p_{mod} = 0$). This result was expected because when the adversary gets the channel in *phase I*, it can only send a single packet. If it sends a malicious packet, it will be immediately detected by its neighbors. Therefore, its legitimate neighbors can take actions against that malicious node as explained in Section 4. Hence, in our simulations, malicious nodes only drop the packets during *phase I* and modify packets or inject bogus ones during *phase II* to reflect the worst case scenario. In Fig. 11, we evaluated the energy consumption-latency metric for different p and C values for both AuCRB and [17]. We observed that the optimum p value of AuCRB does not change with increasing C. Moreover, AuCRB performs considerably better than [17] as more nodes are compromised in the network.

We also evaluated the performance of AuCRB for a smaller network to evaluate the scalability of the proposed scheme. In Fig. 12, we illustrate the energy consumption-latency metric of AuCRB in a network with average node degree of 10 (we decreased the number of nodes from $N = 1000$ to $N = 400$ and kept the other simulation settings exactly the same as described before). We observed that the optimum p value shifted to $p = 0.4$ as we decreased the scale of the network. However, the optimum p value is still independent of the value of C. Hence, we conclude that the optimum p value can be easily set based on the average node degree of the network, independent of the fraction of

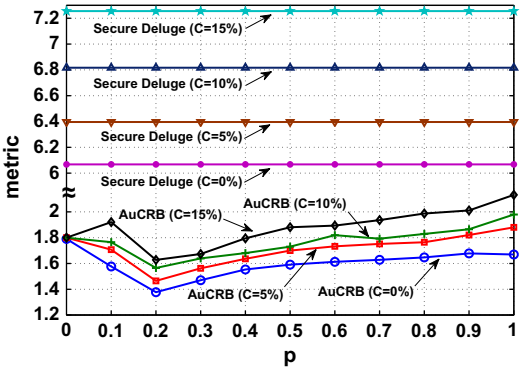


Fig. 11. Energy consumption-latency metric versus p for different C values when $N = 1000$.

Q1 Please cite this article in press as: E. Ayday, F. Fekri, A secure broadcasting scheme to provide availability, reliability and authentication for wireless sensor networks, Ad Hoc Netw. (2012), http://dx.doi.org/10.1016/j.adhoc.2012.03.010

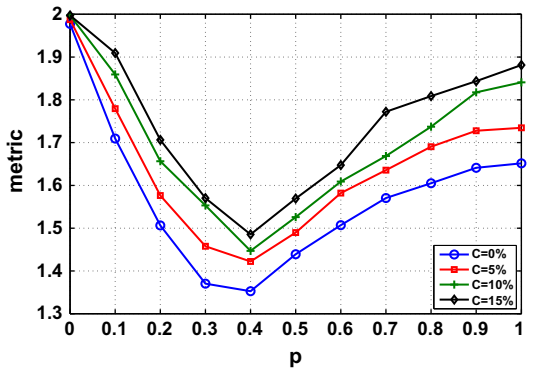


Fig. 12. Energy consumption-latency metric versus p for different C values when $N = 400$.

the compromised nodes. We note that AuCRB considerably outperforms [17] with these settings as well.

Finally, we compared the data availability provided by AuCRB and [17]. When there are no malicious nodes in the network, we observed that all nodes become complete in about 7100 time units. Thus, we take this time as a reference and obtain the availability at 7100 time units for different fractions of malicious nodes in Fig. 13. In this figure, we normalized all latency values by 7100 and compared AuCRB and Secure Deluge in terms of total latency with different fractions of adversarial nodes. The numerical results obtained for our scheme and [17] are summarized in Table 1. The probability of availability at time 7100 and time needed to attain 100% availability are given for different fractions of malicious nodes. We conclude that the scheme in [17] requires twice as much time as our scheme to provide 100% availability.

5.2. Overhead analysis

In this section, we study the computation and communication overheads of AuCRB and compare the proposed scheme with [17]. Our study reveals that the proposed scheme when compared with all other schemes that use the flooding technique is superior in terms of communication overhead and has a negligible disadvantage in computational overhead due to decoding. Thus, all schemes designed based on flooding are much more expensive than AuCRB in terms of the overall communication and computation overhead.

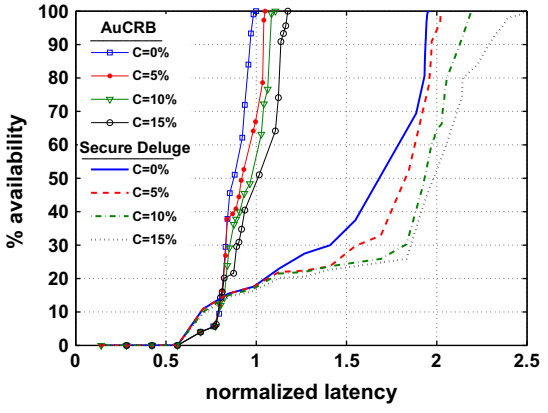


Fig. 13. Data availability versus latency for different fractions of malicious nodes.

5.2.1. Computation overhead

We analyzed the computation overhead of AuCRB for each phase separately.

Phase 0: The computation overhead introduced by the source at *phase 0* is mainly due to signature generation, Bloom filter construction, generation of authentication information for *phase II* and rateless encoding. We note that the computation cost due to rateless encoding and generation of the authentication information for *phase II* is negligible when compared to the other two. Since the Bloom filter takes T input packets and calculates k hash values for every packet, its computation cost is kT hash operations. Thus, assuming that time complexities of a single hash and a single signature calculation are τ_h and τ_s, respectively, the total computational cost at the source is

$$\mathcal{C}_s^0 = (kT)\tau_h + \tau_s. \tag{6}$$

At *phase 0* each receiver node verifies the signed authentication information formed by the source. Hence, if the time complexity for the signature verification is τ_v, the computation cost for the receiver nodes is

$$\mathcal{C}_r^0 = \tau_v. \tag{7}$$

Phase I: At *phase I*, the only computational complexity is due to the Bloom filter verification. If we denote the average number of different packets received by a node at *phase I* as P, then the average computational cost at each node is

$$\mathcal{C}_r^1 = (kP)\tau_h. \tag{8}$$

Phase II: At this phase the main computations are the verification of the partitions, *RatelessII* encoding at the complete nodes and rateless decoding at the nodes that receive sufficient number of packets. However, we note that the cost due to the rateless decoding is dominant. We denote the time complexity to decode T packets as τ_d. Then, the average computational cost per node at phase *II* is

$$\mathcal{C}_r^2 = \tau_d. \tag{9}$$

Thus, the total average computational complexity per source, $\mathcal{C}_s$, and a receiver node, $\mathcal{C}_r$, for the three phases are obtained as

$$\mathcal{C}_s = (kT)\tau_h + \tau_s \tag{10}$$

and

$$\mathcal{C}_r = \tau_v + (kP)\tau_h + \tau_d, \tag{11}$$

respectively.

Table 1
Availability versus latency for AuCRB and Secure Deluge.

C (%)	% Availability @ 7100	100% availability @
AuCRB		
0	100	7100
5	67	7450
10	63	7850
15	47	8340
Secure Deluge		
0	18	13,900
5	17	14,430
10	16	15,560
15	14	17,710

Table 2
Communication and computation overheads per receiver node in AuCRB.

Overhead	*Phase 0* (mJ)	*Phase I* (J)	*Phase II* (mJ)	Total (J)
Computation	45.9	0.46	40.4	0.551
Communication	120.160	4.003	950.4	5.44

We numerically obtained the computational complexity by conducting our calculations for MICA2DOT with 4 MHz 8-bit processor using Chipcon CC100 antenna at both optimum forwarding probability $p = 0.2$ and $p = 1$ (the worst case in AuCRB in terms of energy consumption when $N = 1000$). We used 160-bit elliptic curve signature (ECC-160) and SHA1 for the hash. ECC-160 signature generation and verification consume 22.82 mJ and 45.09 mJ energy, respectively [30]. From [31], we calculated the energy consumption for the hash operations. Moreover, using [32,33], we obtained energy consumed to decode T packets by counting the number of XOR operations, memory reads and memory writes.

As a result, we obtained the average energy consumption (due to computations) for the source as 1.4 J. Further, the average energy consumption (due to computation at each phase) is illustrated in Table 2 for every receiver node. Furthermore, the comparison of our scheme with the Secure Deluge in terms of computation overhead is shown in Fig. 14. Since AuCRB uses rateless codes and provides immediate packet authentication, the computational overhead of our scheme is higher than the Secure Deluge. However, as we will show in Section 5.2.2, this disadvantage has a negligible effect on the overall energy consumption when both the computation and communication overheads are considered.

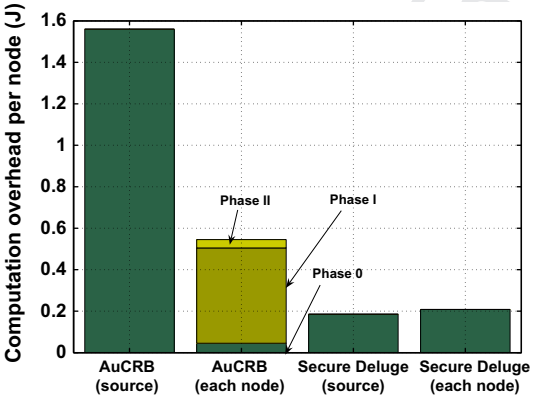


Fig. 14. Comparison of energy consumption due to computation overhead.

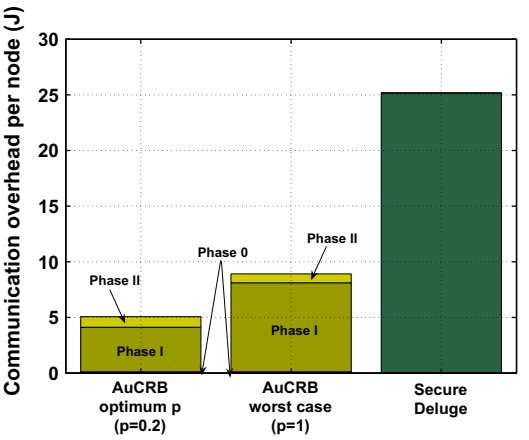


Fig. 15. Comparison of energy consumption due to communication overhead.

5.2.2. Communication overhead

In order to transmit and receive one byte data using Chipcon CC100 antenna, 59.2 μJ and 28.6 μJ is consumed respectively. To be consistent with [17], we used packets with 64 byte payload in our simulations. We calculated the average number of packets sent and received by a node for both optimum forwarding probability $p = 0.2$ and $p = 1$ (the worst case scenario when $N = 1000$) and obtained the average energy consumption per node for different phases of the protocol. The results are summarized in Table 2. Further, we compared the communication overhead of AuCRB for both $p = 0.2$ and $p = 1$ with the Secure Deluge scheme in Fig. 15.

Finally, considering both the computation and communication overheads, we calculated the total energy

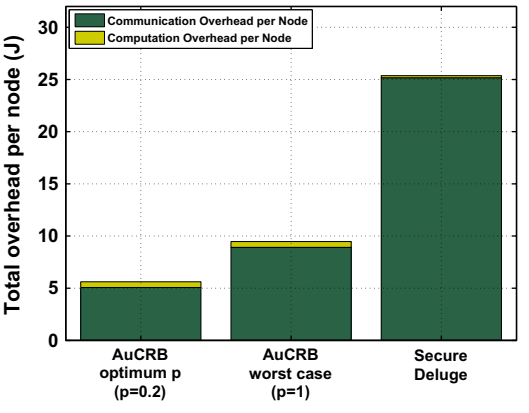


Fig. 16. Comparison of energy consumption due to total overhead.

Q1 Please cite this article in press as: E. Ayday, F. Fekri, A secure broadcasting scheme to provide availability, reliability and authentication for wireless sensor networks, Ad Hoc Netw. (2012), http://dx.doi.org/10.1016/j.adhoc.2012.03.010

consumption per sensor node for AuCRB for $p = 0.2$ and $p = 1$ as well as for [17]. The results are shown in Fig. 16. Hence, we conclude that even in the worst case scenario (when $p = 1$) our scheme outperforms the Secure Deluge in terms of energy consumption. We can make the similar conclusion against all the other schemes which use flooding to ensure availability in broadcasting.

6. Conclusions

This paper was concerned with availability, reliability and authentication for broadcasting in wireless sensor networks, where adversary may compromise nodes, then drops or modifies packets, injects bogus packets or mounts routing attacks. We proposed a secure node-to-network multihop broadcasting scheme by simultaneously considering the above requirements. We build our authentication scheme, on top of a reliable and energy efficient broadcasting protocol to improve efficiency and reliability. We showed superiority of our scheme to meet the requirements with reduced energy consumption and latency. Furthermore, our proposed scheme, due to its built in coding mechanism, can be employed when the packets are lost due to reasons other than the Byzantine attacks.

Acknowledgement

This material is based upon work supported partially by the Army Research Office (ARO) under Grant 49586CI.

References

[1] T. Arampatzis, J. Lygeros, S. Manesis, A survey of applications of wireless sensors and wireless sensor networks, in: Proceeding of IEEE International Symposium on Intelligent Control, vol. 1, 2005, pp. 719–724.
[2] E. Ayday, F. Delgosha, F. Fekri, AuCRB: an efficient mechanism to provide availability, reliability and authentication for multihop broadcasting in wireless networks, in: Proceedings of IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks – SECON'08, 2008, pp. 460–468.
[3] A. Shokrollahi, Raptor codes, IEEE Transactions on Information Theory 52 (6) (2006) 2551–2567.
[4] M. Luby, LT codes, in: Proceedings of IEEE Symposium on Foundations of Computer Science, 2002, pp. 271–280.
[5] N. Rahnavard, B.N. Vellambi, F. Fekri, CRBcast: a reliable and energy-efficient broadcast scheme for wireless sensor networks using rateless codes, IEEE Transactions on Wireless Communications 7 (12) (2008) 5390–5400.
[6] A. Perrig, R. Canettiz, J.D. Tygary, D. Song, Efficient authentication and signing of multicast streams over lossy channels, in: Proceeding of IEEE Symposium on Security and Privacy, 2000, pp. 56–73.
[7] A. Perrig et al., SPINS: security protocols for sensor networks, Wireless Networks 8 (5) (2002) 521–534.
[8] D. Liu, P. Ning, Efficient distribution of key chain commitments for broadcast authentication in distributed sensor networks, in: Proceedings of the 10th Annual Network and Distributed System Security Symposium.
[9] P. Ning, A. Liu, W. Du, Mitigating dos attacks against broadcast authentication in wireless sensor networks, ACM Transactions on Sensor Networks, 4.
[10] J.M. Park, E.K.P. Chong, H.J. Siegel, Efficient multicast stream authentication using erasure codes, ACM Transactions on Information and System Security 6 (2) (2003) 258–285.
[11] I.S. Reed, G. Solomon, Polynomial codes over certain finite fields, SIAM Journal on Applied Mathematics 8 (1960) 300–304.
[12] A. Lysyanskaya, R. Tamassia, N. Triandopoulos, Multicast authentication in fully adversarial networks, in: Proceeding of IEEE Symposium on Security and Privacy, 2004, pp. 241–255.
[13] C. Karlof, N. Sastry, Y. Li, A. Perrig, J. Tygar, Distillation codes and applications to DoS resistant multicast authentication, in: Proceedings of the 11th Network and Distributed Systems Security Symposium – NDSS'04.
[14] A. Perrig, The BiBa one-time signature and broadcast authentication protocol, in: Proceedings of the Eight ACM Conference on Computer and Communications Security CCS, 2001, pp. 28–37.
[15] L. Reyzin, N. Reyzin, Better than BiBa: Short one-time signatures with fast signing and verifying, in: Proceedings of Australasian Conference on Information Security and Privacy – ACISP'02 2384, 2002, pp. 144–153.
[16] S.M. Chang, S. Shieh, W.W. Lin, C.M. Hsieh, An efficient broadcast authentication scheme in wireless sensor networks, in: Proceedings of ACM Symposium on Information, Computer and Communications Security – ASIACCS'06.
[17] P.K. Dutta, J.W. Hui, D.C. Chu, D.E. Culler, Securing the deluge network programming system, in: Proceedings of the Fifth International Conference on Information Processing in Sensor Networks, 2006, pp. 326–333.
[18] B.H. Bloom, Space/time trade-offs in hash coding with allowable errors, Communications of the ACM 13 (7) (1970) 422–426.
[19] W. Ye, J. Heidemann, D. Estrin, An energy-efficient MAC protocol for wireless sensor networks, in: Proceedings of IEEE Conference on Computer Communications – INFOCOM'02.
[20] G. Gaubatz, E.O. Jens-Peter Kaps, B. Sunar, State of the art in ultra-low power public key cryptography for wireless sensor networks, in: Proceedings of IEEE International Conference on Pervasive Computing and Communications – PerCom'05, 2005, pp. 146–150.
[21] D.B. Johnson, A.J. Menezes, Elliptic curve DSA (ECSDA): an enhanced DSA, in: Proceedings of the 7th Conference on USENIX Security Symposium, 1998, pp. 13–13.
[22] F. Hess, Efficient identity based signature schemes based on pairings, in: Revised Papers from the 9th Annual International Workshop on Selected Areas in Cryptography – SAC'02, 2002, pp. 310–324.
[23] S. Cui, P. Duan, C. Chan, An efficient identity-based signature scheme with batch verifications, in: Proceedings of the 1st International Conference on Scalable Information Systems InfoScale'06, 2006, p. 22.
[24] D. Boneh, X. Boyen, E. Goh, Hierarchical identity based encryption with constant size ciphertext, in: Proceedings of International Conference on the Theory and Applications of Cryptographic Techniques – Eurocrypt'05, 2005, pp. 440–456.
[25] T.J. Richardson, R.L. Urbanke, The capacity of low-density parity check codes under message-passing decoding, IEEE Transactions on Information Theory 47 (2001) 599–618.
[26] G. Forney Jr., On iterative decoding and the two-way algorithm, Proceedings of International Symposium on Turbo Codes and Related Topics (1997) 12–15.
[27] A.J. Menezes, P.C. van Oorschot, S.A. Vanstone, Handbood of Applied Cryptography, CRC Press, NY, 1997.
[28] F. Hao, M. Kodialam, T.V. Lakshman, Building high accuracy bloom filters using partitioned hashing, in: Proceedings of ACM International Conference on Measurement and Modeling of Computer Systems, 2007, pp. 277–288.
[29] F. Delgosha, F. Fekri, Threshold key-establishment in distributed sensor networks using a multivariate scheme, in: Proceedings of IEEE Conference on Computer Communications – INFOCOM'06CD-ROM.
[30] A.S. Wander, et al., Energy analysis of public-key cryptography for wireless sensor networks, in: Proceedings of IEEE International Conference on Pervasive Computing and Communications – PerCom'05, 2005, pp. 324–328.
[31] N. Potlapally, S. Ravi, A. Raghunathan, N. Jha, Analyzing the energy consumption of security protocols, in: Proceedings of International Symposium of Low Power Electronics and Design.
[32] Technical Report, Report of MBMS FEC Status in SA4, Technical Report TSGS28(05)0246, Technical Specification Group Services and System Aspects, 2005.
[33] S. Park, J.W. Kim, K.-Y. Shin, D. Kim, A nano operating system for wireless sensor networks, in: Proceedings of Advanced Communication Technology – ICACT'06.

Erman Ayday received his B.Sc. degree in electrical and electronics engineering from Middle East Technical University, Ankara, Turkey, in 2005. He received his M.S. degree in electrical and computer engineering from School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, in 2007. He is currently a Research Assistant in the Information Processing, Communications and Security Research Laboratory and pursuing his Ph.D. degree at the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA. His current research interests include wireless network security, game theory for wireless networks, trust and reputation management, and recommender systems. Erman is the recipient of 2010 Outstanding Research Award from the Center of Signal and Image Processing (CSIP) at Georgia Tech.

Faramarz Fekri received Ph.D. degree from the Georgia Institute of Technology in 2000. Since 2000, He has been with the faculty of the School of Electrical and Computer Engineering at the Georgia Institute of Technology where he currently holds a full Professor position. He serves on the editorial board of the IEEE Transactions on Communications, and on the Technical Program Committees of several IEEE conferences.

His current research interests are in the area of communications and signal processing, in particular coding and information theory, information processing for wireless and sensor networks, and communication security. He received the National Science Foundation CAREER Award (2001), and Southern Center for Electrical Engineering Education (SCEEE) Research Initiation Award (2003), Outstanding Young faculty Award of the School of ECE (2006). He is a Senior Member of the IEEE.

Q1 Please cite this article in press as: E. Ayday, F. Fekri, A secure broadcasting scheme to provide availability, reliability and authentication for wireless sensor networks, Ad Hoc Netw. (2012), http://dx.doi.org/10.1016/j.adhoc.2012.03.010