Volume 2, Issue 10, October 2012 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

A Security of Wireless Sensor Networks and Analysis on Efficient Broadcast Authentication

M.Rameshkumar
Assistant Professor, Department Of Computer Science and Engineering.
Vel Tech Multi Tech Dr.Rangarajan Dr.Sakunthala Engineering College, Chennai, Tamil Nadu, India.

Dr.C.Suresh Gnana Dhass
Professor & Head of IT Dept,
Park College of Engineering and Technology, Kaniyur, Coimbatore, Tamil Nadu, India.

ABSTRACT: A broadcast authentication mechanism is important in wireless sensor networks, assuring receivers of a packet's validity. To provide authentication, some researchers utilize one way key chains and delayed disclosure of keys; however, such an approach requires time synchronization and delayed authentication. Another technique uses one-time signature schemes. Unfortunately, such schemes suffer from large key sizes and limited number of uses per key. To cope with these problems, we propose an efficient, one-time signature-based broadcast authentication scheme for wireless sensor networks that reduces storage usage and includes a re-keying mechanism.

Keywords: **Authentication, re-keying mechanism, Merkle hash trees.**

I. INTRODUCTION

A wireless sensor network (WSN) can cheaply monitor an environment for diverse industries, such as healthcare, military, or home. A WSN typically consists of several base stations and thousands of sensor nodes, which are resource limited devices with low processing, energy, and storage capabilities. Distributing data through wireless communication is also bandwidth limited. Broadcast authentication is a basic and important security mechanism in a WSN because broadcast is a natural communication method in a wireless environment. A message authentication code (MAC) is an authentication tag derived by applying an authentication scheme and a secret key to message. MAC is an efficient symmetric cryptographic primitive for two-party authentication. Therefore, to achieve authenticated broadcasts, it is necessary to establish an asymmetric mechanism in which only the sender consign messages, and the receivers can only verify messages.

A. *Requirements*

In addition to the asymmetric mechanism that is needed for broadcast authentication, designing an efficient broadcast authentication scheme for wireless sensor networks still faces many challenges.

1. Robust to packet loss. The wireless communication Environment is not reliable; therefore, the scheme should be able to cope with the loss of packets during transmission.

2. Short authentication latency. Many WSN applications are real time applications, e.g. the tracking of enemy movements on a battlefield. To authenticate real time data, the maximum number of additional packets that need to be received before a packet can be authenticated should be small.

3. Individual authentication. The receiver should verify the Received packets individually without depending on other Packets; otherwise, the failure to verify a packet prevents the verification of subsequent packets.

4. Low computation cost. Receivers have limited computation power. Thus, they should only perform a small number of operations to verify a packet.

5. Low communication overhead. Because a WSN is restricted in bandwidth, the number of bytes per packet used for authentication should be small.

B.Related Work

Previously proposed broadcast authentication schemes roughly divide into two categories by the cryptographic primitives they use. The first type is a signature amortization scheme that utilizes asymmetric primitives and distributes the cost of the signature over a block of packets. The second type is a MAC-based scheme which uses symmetric primitives to design an elaborate scheme that achieves asymmetry.

As a consequence, receivers cannot authenticate each packet individually and require larger storage space. Because of this buffering problem, we will not use a signature amortization scheme. Instead, we focus on using an efficient symmetric cryptographic primitive to achieve asymmetry. Per rig et al. proposed a very efficient time based stream authentication scheme, called TESLA [19], and also provided a tiny version for WSN, called μ TESLA [18]. They use pure symmetric primitives to achieve asymmetric property by one way key chain and delayed disclosure. However, μ TESLA has some constraints including time synchronization of the whole network, inefficient unicast of the initial trust, and delayed authentication. We detail our proposed scheme in section 3. In section 4, we provide security analysis of our scheme and also compare it against other authentication schemes for WSNs. Finally, we conclude our findings in section 5.

II. PRELIMINARIES

In this section, we first outline the system architecture. We then review some cryptographic primitives used for authentication and a one-time signature called HORS, which is thus far the fastest one-time signature scheme for signing and verifying. Our scheme can be viewed as an improvement of HORS.

A. System Architecture

A WSN may contain one or more base stations and hundreds or thousands of sensor nodes. A sensor node can broadcast messages by first uni casting the message to the base station, which then broadcasts the messages on the sensor node's behalf. In addition, messages transmitted in a sensor network may reach the destination directly or may be forwarded by some intermediate nodes; however, we do not distinguish between them in our scheme. Furthermore, we assume a base station shares apiaries secret key with each sensor node, allowing each sensor node to securely receive the base station's public key.

B. Cryptographic Primitives

In this section, we introduce various cryptographic primitives' and schemes for authentication.

1) Message Authentication Code

A message authentication code (MAC) is a symmetric cryptographic mechanism that takes as input a *k*-bit secret key and message, and outputs an *l*-bit authentication tag. To exchange authentic messages, a sender and receiver must share the same secret key. Using the secret key, the sender computes the message's authentication tag (or MAC) and appends it to the message. To verify the authenticity of a message, the receiver computes the message's MAC with the secret key and compares into the original MAC appended with the message.

For any message, a secure MAC function prevents an attacker without prior knowledge of the secret key from computing the correct MAC. A MAC achieves authenticity for point-to-point communications because a receiver knows that a message with the correct MAC must have been generated either by itself or by the sender.

2) Collision Resilient Hash Function

A collision-resilient hash function H [10] is a function that maps an arbitrary length message M to a fixed length message digest MD and exhibits the following properties. (1) The description of H is publicly known and does not require any secret information for its operation. (2) Given x, it is easy to compute $H(x)$. (3) Given y, in the range of H, it is computationally infeasible to find an x such that $H(x) = y$. (4) It is computationally infeasible to find two distinct messages (M, M') that hash to the same result $H(M) = H(M')$.

The collision-resilient hash function is very efficient. It only costs few microseconds to compute on a Pentium □800 Hz PC, which is a thousand times cheaper than asymmetric primitives.

3) Message Authentication Code

A Merkle hash tree [7], [8], [9] is a mechanism for calculating a message digest over a group of data items. Figure 2-2 depicts a Merkle hash tree. It is constructed from a binary tree by using the hash of each data item as a leaf in the tree. Each internal node is then computed by taking the hash of the concatenation of its two children. Let H be a collision resilient hash function. Then, $parent = H(child\ left \mid child\ right)$.

A Merkle hash tree can reduce the authentication overhead needed for a large group of data items. For example, a sender signs the root of the tree instead of individual data items. The receiver can then verify the authenticity of every data item by reconstructing the tree and comparing the computed hash value of the tree, which we call *tree hash*, with the authenticated root value. To reconstruct the tree, the receiver needs all of the data items. An alternative is for the receiver to verify a data item individually by computing the tree hash using the data item and its authentication path.

C. One-time Signatures

First proposed by Lam port [6] and Rabin [13], one-time signature schemes are efficient signature schemes based on one way functions. These signature schemes differ from public key signature schemes by the number of messages each can sign.

Using a single key pair, the former can only sign several messages, while the latter can sign an unlimited number of messages. This is due to the disclosure of the private key of a onetime signature scheme shortly after signing a few messages.

Public key signature schemes never disclose a private key. Despite this limitation, one-time signature schemes are advantageous because of their speed. Since they are based on one way functions, their computation cost is quite low when compared with that of asymmetric primitives. In general, one-time signature schemes use the secret key as input to a sequence of one-way functions which generate a number of intermediate results that eventually lead to the public key.

The one-way property of the function implies that it is infeasible to compute the secret key, or any intermediate result, from the public key. The private key is a self-authenticating value. Motivated by the application of signatures to stream and broadcast authentication, Perrig proposed a one-time signature called BiBa [2], which features fast verification and a short signature. The disadvantage of BiBa is a longer signing time and larger public key size than previous schemes.

1) Sub subsections

Reyzin and Reyzin proposed a new one-time signature scheme called HORS [3], which improves upon the BiBa scheme by decreasing the time needed to sign and verify messages while also reducing the key and signature sizes, making HORS the fastest one-time signature scheme available thus far. The security offbeat depends upon the random-oracle model, while the security of HORS relies on the assumption of the existence of one-way functions and the *subset-resilience.*

III. PROPOSED SCHEME

We propose a computationally lightweight one-time signature scheme that allows sensor nodes to authenticate broadcast messages from a base station in a wireless sensor network.

A. Signature Scheme

Compared to HORS, our scheme consumes less storage and communication overhead at the expense of a higher computation cost. This is a worthwhile tradeoff because storage is a more precious resource than computation power in a sensor node, especially since our scheme only uses a few additional hash computations.

In the remainder of this section, we first explain the basic idea of our scheme, followed by our generalized scheme. Finally, we propose a re-keying mechanism for our scheme.

1) The Basic Idea

First, the signer must generate the key pair. The key pair includes the *private key*, which consists of *t* random numbers, and the *public key*, which consists of *t* hash values of these *t* random numbers. For convenience, we call these random numbers *private balls* and their hash values *public balls*.

A verifier can efficiently authenticate the private balls based on the public balls but cannot feasibly compute a valid private ball given a public ball. HORS uses the one-way hash function *F* as commitment scheme. Given a set of private balls, the public balls $pi = F(ri)$. The verifier can easily authenticate *ru*by verifying $pi = F(ri)$.

As previously mentioned, the public key is composed of *t* public balls. To reduce the size of the public key, we can either reduce the ball size or the number of balls. Because the length of public ball is related to the security strength of the hash function, we cannot reduce the ball size.

A change in the public key generation also effects signature generation and verification. The signer generates the signature as before, by picking *k* private balls out of *t* private balls. However, the signer must affix additional public balls to the signature. The additional public balls correspond to the authentication path of each picked ball. Using the authentication path, the verifier can, validate each picked ball by reconstructing the path from picked ball to the root of the Merkle hash tree.

One security flaw occurs when an attacker replaces a disclosed private ball *i* with private ball *j*. We cannot distinguish between two private balls in the same tree. To resolve this problem, we use the uniqueness of each leaf's authentication path. For each private ball, we concatenate the public balls along its authentication path. Then, we obtain the identity of the private ball by applying the hash function to this concatenated value.

2) The Generalized Scheme

We generalize our scheme by first constructing many small Merkle hash trees of height *h* that hold *2h* private balls. The public key contains the root nodes of all the Merkle hash trees, and hence reduces the key size by a factor of *2h*. However, to

Authenticate each private ball, the signer adds the authentication path of each private ball, which requires *h* verification nodes.

Thus, the signature size increases by a factor of *h*. By constructing many Merkle trees, we can lower the overall storage requirement. As public key size decreases, the signature size increases. If we only build one Merkle tree, the combined size of the public key and signature would not be minimal. Therefore, we need to find an optimal balance between the public key size and signature size.

2.1) Key Generation

A key pair consists of a private key and a public key. The private key is composed of *t l*-bit random numbers created by pseudorandom generator, and the public key is derived from these random numbers. First, we input *t* random numbers into the one way hash function and output *t* hash values. Then, we separate *t* hash values into *d* groups so that there are *t/d* values in each group.

Finally, we use these *t/d* values as the leaves of the binary tree and compute each intermediate node as the hash of the concatenation of the two child values. Thus, we attain *d* Merkle trees, whose roots comprise our public key. We note that the original public key of HORS is *t* hash values generated from *t* random numbers while our public key is *d* Merkle tree roots.

3-5. A sender first transmits ball *s0* and its authentication path *{v1, m23, m47}*. It then sends *s1* and its authentication path *{v0, m23, m47}*. The sender should send nodes *m23* and *m47* once since they are duplicates. Moreover, a receiver can compute *v0 = H(s0)*.

Thus, if a sender selects a direct neighbor of a disclosed private ball, then no additional nodes are required to be sent. In general, if a node at height *e* is a common parent, then all nodes higher than *e* need not be resent. Therefore, selected private balls that are close together can reduce transmission overhead. We note that the upper bound of the sum of the authentication paths is *min(r*k*h, the whole tree).*

2.2) Authenticating Broadcast Messages

When a receiver obtains a broadcast message, it must ensure that the message originated from an authentic sender by verifying the signature of message *m* with the following procedure. Since the sender will periodically re-key, the receiver must first decide which of the sender's public keys should be used to verify the message's signature. Second, the receiver checks which sequence period of that public key the sequence number falls into. Third, it computes the hash value *h = H(m)*. Fourth, the receiver separates the hash value h into k pieces and regards these pieces as integers317 *(i1, i2,…,ik)* between *0* and *t-1*, with each integer as an index to a private ball *(r1, r2,…, rt)*. Next, the receiver checks the identities of the balls by uniqueness of authentication path as discussed at the end of section 3.1.1. Finally, the receiver verifies each private ball by computing the tree hash of the private ball with its authentication path and checking whether this tree hash is equivalent to the public key. A complete match assures the receiver that the private ball belongs to the authenticated sender.

B.Re-keying Mechanism

Because a single key pair can only sign *r* messages, a sender should use a new key pair when signing more than *r* messages.

Therefore, we offer a re-keying scheme as a solution. If one key pair can sign *r* messages, we set the duration of a sequence period to *r*. The sequence numbers of the first public key ranges from *0* to *r-1*, the sequence numbers of the second public key ranges from *r* to *2r-2*, and so on, as depicted in A receiver must first check which sequence period a received message belongs to. Since a base station shares a pair-wise secret key with each sensor node, the base station unicast the first public key to each sensor node through the authenticated channel. For efficiency, the base station may distribute the next public key by authenticated broadcast using the old private key.

C. Comparison

We compare our proposed scheme with μ TESLA [18], an efficient broadcast authentication protocol for WSNs. Our scheme has four main advantages over μ TESLA, which we list below.

1. No time synchronization requirement. In μ TESLA, the sender and receivers utilize time synchronization and delayed disclosure to achieve asymmetry needed for broadcast authentication, while our scheme uses public private key pairs. Thus, our scheme has no time synchronization requirement, which may be impractical in large WSN.

2) No buffering needed by receiver. M TESLA requires receivers to buffer a received packet until the corresponding MAC key is disclosed. Receivers need not buffer received messages in our scheme.

*3) Individual authentication of message***.** Receivers can individually authenticate a received packet without waiting for another packet. In μ TESLA, receivers must wait for the packet containing the disclosed MAC key.

*4) Instant authentication of message***.** With our scheme, receivers can instantly authenticate a received packet. In contrast, μ TESLA requires a receiver to wait one or more time periods for the packet containing the disclosed MAC key.

We also compare our scheme against two other efficient one-time signature schemes, BiBa [2] and HORS [3]. Testing at the same security level with system parameters of *t=1024, k=16, r=10,h=5*, our proposed scheme outperformed BiBa and HORS in three areas: computation, communication, and storage overhead.

.

IV. CONCLUSION

In this paper, we propose an efficient broadcast authentication scheme for wireless sensor networks that utilizes a one-time signature scheme and re-keying mechanism. Our scheme exhibits many nice properties including individual authentication, instant authentication, robustness to packet loss, and low overhead in computation, communication and storage. We improve upon the HORS one-time signature scheme, reducing the large key storage requirement of HORS by using Merkle hash trees in generating the key pair. Despite increasing computation and communication overhead, we significantly decrease the storage space usage. We also devise a simple but efficient re-keying mechanism for our scheme.

REFERENCES

[1] I.F.Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A Survey on Sensor Networks," IEEE Comm. Magazine, Vol. 40, No.8, pp. 102–114, Aug. 2002.

[2] G. Avoine, P. Junod, and P. Oechslin, "Time-Memory 3797, pp. 183–196, Springer-Verlag, 2005.

[3] A. Durresi, V. Paruchuri, S. Iyengar, and R. Kannan, "Optimized Broadcast Protocol for Sensor Networks," IEEE Trans. Computers,vol. 54, no. 8, pp. 1013–1024, Aug. 2005.

[4] S. Ganeriwal, S. Capkun, C. Han, and M. Srivastava, "Secure Time Synchronization Service for Sensor Networks," Proc. ACM Workshop on Wireless Security (WiSe), pp. 97–106, 2005

[5] Y. Hu, M. Jakobson, and A. Perrig, "Efficient Constructions for One-way Hash Chains," Proc. ACNS 05, LNCS 3531, pp. 423–441,Springer-Verlag,2005

[6] Intel IMote2 Overview, http://www.intel.com/research/ downloads/imote_overview.pdf, 2005. Commercialized by Crossbow, INC. http://www.xbow.com/.

[7] M. Karaata and M. Gouda, "A Stabilizing Deactivation/Reactivation Protocol," IEEE Trans. Computers, vol.56, no. 7, pp. 881–888, Jul. 2007.

[8] Q. Li and D. Rus, "Global Clock Synchronization in Sensor Networks," IEEE Trans. Computers, vol. 55, no. 2, pp. 214–226, Feb. 2006.

[9] D. Liu, P. Ning, "Efficient Distribution of Key Chain Commitments for Broadcast Authentication in Distributed Sensor Networks," Proc. ISOC Network and Distributed System Security Symposium (NDSS), pp. 263–276, Feb. 2003.

[10] D. Liu and P. Ning, "Multi-Level μTESLA: Broadcast Authentication for Distributed Sensor Networks," ACM Trans. Embedded Computing Systems, Vol. 3, No. 4, pp. 800–836, Nov. 2004.

[11] M. Luk, A. Perrig, and B. Willock, "Seven Cardinal Properties of Sensor Network Broadcast Authentication," Proc. ACM Workshop on Security of Ad

[12] K. Ren, W. Lou, and Y. Zhang, "Multi-user broadcast authentication in Wireless sensor networks," in *Proc. SECON*, San Diego, CA, Jun. 2007,pp. 223–232.

[13] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey On sensor networks," *IEEE Commun. Mag.*, vol. 40, no. 8, pp. 102–114, Aug. 2002.

[14] I. Akyildiz and I. Kasimoglu, "Wireless sensor and actor networks: Research Challenges," *Ad Hoc Netw.*, vol. 2, no. 4, pp. 351–367, Oct. 2004.

[15] K. Ren and W. Lou, *Communication Security in Wireless Sensor Networks*. Saarbrücken, Germany: VDM Verlag, 2008.

[16] K. Ren, W. Lou, and Y. Zhang, "LEDS: Providing location-aware endto- End data security in wireless sensor networks," *IEEE Trans. Mobile Comput.*, vol. 7, no. 5, pp. 585–598, May 2008.

[17] Q. Wang, K. Ren, W. Lou, and Y. Zhang, "Dependable and secure Sensor data storage with dynamic integrity assurance," in *Proc. IEEE INFOCOM*, 2009, to be published.

[18] S. Yu, K. Ren, and W. Lou, "FDAC: Toward fine-grained distributed data Access control in wireless sensor networks," in *Proc. IEEE INFOCOM*, 2009, to be published.

[19] R. Zhang, Y. Zhang, and K. Ren, "DP2AC: Distributed privacy-preserving Access control in sensor networks," in *Proc. IEEE INFOCOM*, 2009, to be Published.

[20] C. Lu, G. Xing, O. Chipara, C. Fok, and S. Bhattacharya, "A spatiotemporal Query service for mobile users in sensor networks," in *Proc. ICDCS*, Washington, DC, Jun. 2005, pp. 381–390.

