Hybrid Multi-user Broadcast Authentication for Wireless Sensor Networks

Sungjune Yoon∗ Tomoyuki Asano† Masafumi Kusakawa† Hyunrok Lee∗ Kwangjo Kim∗

Abstract— In wireless sensor networks (WSNs), broadcast authentication allows only legitimate users (or senders) to disseminate messages into the networks. µTESLA [14] is the first broadcast authentication scheme for WSNs. It allows only few users, mainly several base stations, to broadcast messages, but in reality there are many scenarios that require a large number of dynamic users (mobile sinks or users) [17]. Until now, only few schemes have dealt with multiple broadcasting users based on public key cryptosystem [16, 17]. However, these schemes require heavy computation and communication overhead on sensor nodes. To minimize the overhead, we propose a hybrid multi-user broadcast authentication scheme for WSNs.

Keywords: Wireless Sensor Network, Broadcast Authentication

1 Introduction

Wireless sensor networks (WSNs) have been attracting a lot of interest as one of the core technique for the upcoming ubiquitous age. A WSN is a wireless ad-hoc network which consists of thousands of tiny resource-constrained devices which gather environmental data with several powerful and secure base stations. Although WSNs are kind of ad-hoc networks, there are several differences which make security protocols originally designed for ad-hoc networks impractical for WSNs; for example, more constrained resources and large scale deployments [4].

Broadcasting is an efficient communication method for disseminating messages (queries, commands, etc.) into WSNs. Although sensor nodes have very limited resources, each sensor node has to authenticate these messages before processing them since these types of messages are of importance. Therefore, broadcast authentication (BA) is one of the most important security mechanisms. Most of the previous BA schemes used a secret key cryptosystem (SKC) as a basic security building block to minimize the energy consumption in sensor nodes and allowed only few fixed users to broadcast messages [3, 5, 10, 12, 14, 19].

However, some applications need a mechanism by which a large number of users can broadcast messages into WSNs [17]. To support multiple users, some SKC-based schemes [1, 11] and public key cryptosystem (PKC)-based schemes [17, 16] have recently proposed, but the former is impractical when users are dynamically added or deleted and the latter requires much more communication and computation overhead than the former.

To support an amount of dynamic users with low communication and computation overhead, we propose a hybrid multi-user broadcast authentication scheme by adopting public key concept into µTESLA [14].

The remaining parts are organized as follows: In Section 2, we briefly review the previous work. In Section 3, we provide some preliminaries. In Section 4, we describe our design goal, assumptions and then explain our proposed scheme. In Section 5, we analyze the efficiency and security of our scheme and then compare it with other ones. Finally, we make a conclusion in Section 6.

2 Previous work

Earlier studies mainly focused themselves on SKC-based BA. µTESLA is well known scheme to provide source authentication and message integrity by utilizing a one-way hash chain (OHC) [8] and loosely-coupled time synchronization between a sender and receivers. µTESLA is an efficient broadcast authentication mechanism, but has limited scalability due to its unicast-based parameter distribution to add new receivers.

Multi-level µTESLA [10] and L-TESLA [5] were designed to enhance the scalability of µTESLA. However, these schemes do not support a large number of broadcast users since the broadcast parameters of all the users (we explain this at next section) must be stored into all sensor nodes which usually have a small storage. T-TESLA [11] supports a lot of users by utilizing Merkle hash tree [13], but all users must be predetermined before the deployment of sensor nodes.

Due to the recent advances in sensor nodes, PKC has become a good solution for providing security services
even in a tiny sensor node [6, 15, 18]. The main advantages are to construct simple protocol and to authenticate a message immediately. Based on these advantages, Ren et al. [17, 16] have proposed several multi-user broadcast authentication (MBA) schemes which focused on reducing the number of PKC operations and increasing the number of users. In order to authenticate a message in their approaches, however, each sensor node has to verify the signature of the message individually; thus, all the approaches could more easily exhaust the scarce energy of sensor nodes than SKC-based ones.

Lanigan et al. [9] suggested an idea to combine a digital signature and an OHC to efficiently broadcast a bulk of messages for program updates or reprogramming, but their scheme does not support a large number of network users. Benenson [2] proposed a PKC-based user authentication scheme which could be used to establish a bunch of secure channels between a user and his neighbor sensor nodes, but did not provide any specific method to broadcast users’ messages into WSNs.

3 Preliminaries

3.1 $\mu$TESLA

It provides an authentication method for broadcasted messages from a trusted and powerful base station (BS) to a large number of resource-constrained sensor nodes. Before the deployment of sensor nodes, BS generates a chain of keys through repeatedly applying a one-way hash function $H$ on a random number $k_Q$ as follows: $k_i = H(k_{i+1})$ for $i=Q-1, \ldots, 0$. Then, BS splits up the expected lifetime, $T_{LF}$, of a WSN into uniform time intervals, $T_{INT} = \frac{T_{LF}}{n}$, and associates each key in the key chain with one time interval. The broadcast parameters of BS (the first key in the chain, $k_0$, the broadcast start time, $T_{START}$, the time interval, $T_{INT}$, and the key discloser time schedule, $z$, which is an integer value $z > 1$) are distributed into sensor nodes.

After the deployment, BS uses $k_i$ to compute the message authentication code (MAC) of broadcasted messages in $i$ time interval. In time interval $i + z$, BS reveals key $k_i$. Upon receiving $k_i$, sensor nodes first verify the key by comparing $k_0$ and $H^i(k_1)\dagger$ and then they can verify the messages arrived at the time interval $i$. See [14] for the details.

3.2 $(n, t)$-threshold scheme

An $(n, t)$-threshold scheme implies that any cooperative attacks do not break the security of the scheme without more than $t$ entities out of $n$ cooperate the attacks. Usually $n$ represents all the entities in a network. In our paper, however, a network is further divided into several clusters, each cluster consists of $n$ entities (i.e., $n$ sensor nodes), and $(n, t)$-threshold assures that a network is secure unless $t$ sensor nodes are compromised out of $n$ sensor nodes in a cluster.

\dagger Repeatedly hashing $k_i$ $i$ times.

4 Our proposed scheme

4.1 Design goal

Our goal is to design an efficient MBA scheme to minimize the communication, computation, and memory overhead. It must satisfy the following requirements:

- A large number of dynamic users: Apart from BS, there are a lot of mobile users who could be dynamically added and deleted, so an MBA scheme must support this characteristic.

- Broadcast efficiency: Although sensor nodes are getting improved, their energy is always limited; thus, the scheme must have as low communication and computation overhead as possible.

- Security: Without compromising at least $t$ sensor nodes in a cluster, an adversary could not broadcast forged messages into WSNs and the damage from DoS attacks must be confined within a small part of the network.

4.2 Assumption and Notations

In this subsection, we describe our assumption of the network architecture and its entities and notations used in the remaining section.

Usually WSN consists of hundreds to thousands of resource-constrained sensor nodes, and one trusted and powerful base station (BS). We further divide the network into $N$ by $N$ grid cells and each cell, which represents a cluster, $C_i$ $(1 \leq i \leq N \times N)$, has 4 edges which connect exactly two cells. $\eta^m_i$ represents $m$th sensor node in $C_i$. Additionally, we define two types of sensor nodes; master node of $C_i$ and gateway node connecting $C_i$ and $C_j$ denoted by $M_i$ and $G_{ij}$ respectively.

In $C_i$, there are $n$ nodes; one $M_i$, $8t - 4 C_{ij}$s in order to prevent the network from node capture attacks, and $n - 8t + 3 n^0$ s. The combination of $m$ and $i$ (or $ij$) is unique in WSN. $M_i$ manages $C_i$ and has its own $\mu$TESLA parameters (time interval ($T_{INT}$), initial broadcast start time ($T_{START}$), initial commitment ($HK^0_i$), the first key in an OHC), and key discloser schedule, $(z^i > 1)$ to locally broadcast messages into $C_i$. Every $G_{ij}^m$ and $\eta^m_i$ share a secret with $M_i$ (and $M_j$ in case of $G_{ij}^m$) and have the $\mu$TESLA parameters of their $M_i$ (and $M_j$ in case of $G_{ij}^m$). All nodes have the public key of BS ($PK_{BS}$). Even though $\eta^m_i$ is a resource-constrained device, infrequent public key computation tasks (more specifically, signature verifications) do not affect its lifetime [15].

$G_{ij}^m$ is a special sensor node which is at the edge between $C_i$ and $C_j$; thus, it belongs to these two clusters and relays a message between them.

$M_i$ is a strongly secured sensor node which equips a tamper-resilient device. Since the number of $M_i$ is relatively small, the total cost of embedding tamper-resistant devices into all $M_i$s is not so much. $M_i$ has a certificate revocation list (CRL) issued by BS. $M_i$ periodically releases its $\mu$TESLA keys according to $z^i$. 


Table 1: Notation

<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$K_{AB}$</td>
<td>Shared secret key between $A$ and $B$</td>
</tr>
<tr>
<td>$T_c^A$</td>
<td>Timestamp of current time generated by $A$</td>
</tr>
<tr>
<td>$D$</td>
<td>Upper bound of broadcast delay</td>
</tr>
<tr>
<td>$Sign_{SK_A}(msg)$</td>
<td>Signature of $msg$ signed by $SK_A$</td>
</tr>
<tr>
<td>$MAC_{K_{AB}}(msg)$</td>
<td>Message authentication code of $msg$ using $K_{AB}$</td>
</tr>
<tr>
<td>$UMB$</td>
<td>Message buffer</td>
</tr>
<tr>
<td>$</td>
<td>A</td>
</tr>
</tbody>
</table>

At $r + z^i - 1$ time interval, every $\eta_i^m$ and $G_{ij}^m$ have $HK_{T_{c-1}}^i$. Thus, these nodes can use the key to verify $HK_{T_c^i}$ received at $r + z^i$ time interval.

BS is assumed to manage the entire WSN. In our scheme, BS just issues certificates to users and broadcasts CRL only to $M_i$. We assume there is a secure broadcast mechanism between the BS and $M_i$ by which each $M_i$ can maintain an up-to-date CRL. To reduce the communication and memory overhead caused by the CRL, it contains only the ID of revoked users. Therefore, revoking a user’s certificate has the same meaning that of revoking a user in our scheme. Ren et al. [16] provide a good method to reduce these overhead.

User ($U$) is a person who wants to broadcast a message ($msg$) into WSN and has already received a certificate ($Cert_U$) which consists of the ID of $U$ ($ID_U$), the expiration time of the certificate ($T_{exp}$), the public key of $U$ ($PK_U$), and a signature ($Sign_{SK_U}(ID_U,T_{exp},PK_U)$), and the secret/public key pair ($SK_U$ and $PK_U$) from BS. Signing is not much difficult from the $U$’s point of view. We assume that $U$ can access at least $t$ $\eta_i^m$ in the communication range of $U$ ($RU$) before broadcasting $msg$. If not, $U$ must move to other areas. Table 1 shows the other notations used in remaining sections.

4.3 Hybrid Multi-user Broadcast Authentication (H-MBA)

Step 1. Before broadcasting $msg$, $U$ generates $Sign_{SK_U}(msg,T_c^U)$ and then broadcasts the following packet, $P_1$, into $RU$.

$$P_1 = < msg, T_c^U, Sign_{SK_U}(msg,T_c^U), Cert_U >$$

Step 2. On receiving $P_1$, each $\eta_i^m$ first checks whether $T_c^i + D$ is smaller than its current time or not. If so, it further verifies $Cert_U$ and $Sign_{SK_U}(msg, T_c^i)$. If the verification succeeded, it sends the following packet, $P_2$, to $M_i$.

$$P_2 = < ID_{\eta_i^m}, ID_U, msg, T_c^i >$$

$$MAC_{K_{M_i,\eta_i^m}}(ID_{\eta_i^m}, ID_U, msg, T_c^i) >$$

Algorithm 1 Local broadcast for users

Assuming that $M_i$ receives $P_2$ or $P_5$ at $T_c^{M_i}$, $M_i$ will do the following

If $T_c^U + D < T_c^{M_i}$
Discard $P_2$ (or $P_5$) then exit.

End if

If $ID_U$ is in the CRL or the MAC is not authentic
Discard $P_2$ (or $P_5$) then exit.

End if

If $(ID_{\eta_i^m}, ID_U, msg, T_c^i)$ is not in UMB
Store $(ID_{\eta_i^m}, ID_U, msg, T_c^i)$ into UMB
Else if
Discard $P_2$ (or $P_5$) then exit.

End if

If $t = \text{the number of entries containing} (ID_U, msg, T_c^i)$

$$r = \frac{T_c^M - T_{STR}}{T_{INT}}$$

$$P_3 = (ID_U, msg, T_c^U, MAC_{HK_i}(ID_U, msg, T_c^U))$$

broadcast $P_3$

End if

Step 3. On receiving $P_2$ or $P_5$ (see Step 6), $M_i$ invokes the Algorithm 1.

Step 4. When $(T_c^M, T_{STR}) \mod T_{INT} = \text{equal to 0}$, $M_i$ sets $r$ to $\frac{T_c^M - T_{STR}}{T_{INT}}$ – $z^i$ and broadcasts the following packet, $P_4$.

$$P_4 = < HK_i^1 >$$

Step 5. On receiving $P_4$, each $\eta_i^m$ (and $G_{ij}^m$) in $C_i$ verifies $P_4$ and then does $P_5$. If one of the verifications is failed, just drops $P_5$. If the verifications are succeeded, it forwards $< ID_U, msg, T_c^i >$ to its message processing unit which actually handles the received message.

Step 6. Each $G_{ij}^m$ which successfully verified $P_5$ in $C_i$ checks whether $P_5$ is already received from $M_j$ by searching its $UMB$. If not, $G_{ij}^m$ appends $< ID_U, msg, T_c^i >$ to the $UMB$, calculates the MAC of $< ID_{G_{ij}^m}, ID_U, msg, T_c^i >$ using $K_{M_i,G_{ij}^m}$, and then sends the following packet, $P_6$, to $M_j$.

$$P_6 = < ID_{G_{ij}^m}, ID_U, msg, T_c^i >$$

$$MAC_{K_{M_i,G_{ij}^m}}(ID_{G_{ij}^m}, ID_U, msg, T_c^i) >$$

Step 7. Repeat Steps 3 to 6 until the $msg$ is broadcasted into the entire network.

5 Analysis

5.1 Communication overhead

The communication overhead heavily depends on the expected number of compromised nodes, i.e., the threshold value $t$, since it decides the number of $G_{ij}^m$ in $C_i$ and the number of $\eta_i^m$ in $RU$. Additionally, it also relies
upon the length of routing path between $G_{ij}^m$ (or $\eta_i^m$ in $R_U$) and $M_i$. For the sake of simplicity, we assume that all the nodes in $C_i$ are uniformly deployed, $M_i$ is at the center of $C_i$ and $G_{ij}^m$ is at the edge of $C_i$, and a message from $G_{ij}^m$ or $\eta_i^m$ to $M_i$ is relayed by all the intermediaries between them. Thus, the average routing length (ARL) from $G_{ij}^m$ to $M_i$ is $\frac{\sqrt{2n}}{2}$. Figure 1 shows an example of $C_i$ where $n = 25$ and $t = 2$.

![Figure 1: Sensor nodes in $C_i$](image)

In addition, we also assume that the $msg$ is relayed by a half of $G_{ij}^m$ from one cluster to other clusters in average. Therefore, the average communication overhead per a node in $C_i$ can be computed by the following equation.

$$\frac{(4t-2) \times (|P_3| - |msg|) \times ARL + n \times (|P_3| - |msg|)}{n}$$

We will show some quantitative evaluations of the communication overhead comparing with other schemes in Section 5.6.

### 5.2 Computation overhead

Since $R_U$ is usually small, $\text{Sign}_{SK_{BS}}(ID_U, T_{exp}, PK_U)$ and $\text{Sign}_{SK_{ij}}(msg, T_{ij}^U)$ are verified by only few $\eta_i^m$. The other nodes only need to compute one hash value and one MAC value (see Step 5), and additional $t + 1$ MAC computations in each $M_i$ and one MAC computation in each $G_{ij}^m$.

For the sake of simplicity, we assume that all the nodes in $C_i$ receive exactly one $P_3$ and there are $2t - 1$ $G_{ij}^m$ at each edge of $C_i$ (so, $8t - 4$ $G_{ij}^m$ in a cluster). We do not include the energy consumption of CRL and UMB searching operations which are usually negligible when their size is small and the energy consumption of a message transmission since it could be easily computed by the above communication overhead.

The average computation overhead per a sensor node in a cluster which directly receives $P_3$ can be computed by the following equation ($O_{PKC}$, $O_{MAC}$, and $N_{RU}$ represent a signature verification overhead, a MAC computation overhead\(^2\), and the number of $\eta_i^m$ in $R_U$ respectively).

$$\frac{(2 \times O_{PKC} + O_{MAC}) \times N_{RU} + (9t + 2n - 3) \times O_{MAC}}{n}$$

\(^2\)To simplify the equation, we assume that one hash computation requires the equivalent computation of one MAC.

We will show some quantitative evaluations of the computation overhead comparing with other schemes in Section 5.6.

### 5.3 Memory overhead

Each $\eta_i^m$ should store $PK_{BS}$, the $\mu$TESLA parameters of $M_i$, and a shared secret key with $M_i$. $G_{ij}^m$ should store $PK_{BS}$, two $\mu$TESLA parameters, two secret keys (one with $M_i$ and another with $M_j$), and some additional memory for its UMB whose size is decided by the expected number of users’ messages which consist of $ID_U$, $T_{ij}^U$, and $msg$ during one time interval. The messages in UMB are periodically removed according to $D$. Figure 2 shows the maximum number of users’ messages which should be accepted by $G_{ij}^m$ during one time interval (we assume that $|PK_{BS}|$ is 20 bytes, a shared secret key is 16 bytes and a set of $\mu$TESLA parameters is 26 bytes).

![Figure 2: Acceptable number of messages](image)

### 5.4 The number of dynamic users

The number of users supported by H-MBA is decided by $|ID_U|$; thus, the memory limitation of sensor nodes does not affect the number of users. Moreover, dynamic user addition and deletion are very easy. User addition is simply done by issuing certificates from $BS$ and user deletion is done by broadcasting the $ID$ of deleted users to $M_i$ from $BS$ [16]. It means that user addition does not consume the energy of sensor nodes and the deletion requires little energy consumption in the intermediary nodes between $M_i$ and $BS$ to relay the $ID$ of deleted users.

### 5.5 Security

The security strength of H-MBA is decided by underlying MAC function and PKC used for generating a signature as well as $t$. For example, if we use HMAC-SHA1 and ECC-160 as its building blocks, it provides 80 bits level of security. The threshold, $t$, must be carefully chosen according to the possibility of node capture attacks.

- Resilient against node capture attacks: Since the message of a user can not be broadcasted until $M_i$ receives at least $t$ authentic MACs from $t$ different sensor nodes (shared secret keys between $M_i$ and $\eta_i^m$ are distinct keys). Therefore,
an attacker has to compromise more than $t$ sensor nodes or a $M_i$ to flood a forged message into the whole WSN.

- **Resilient against DoS attacks**: An adversary can forge any of H-MBA packets, $P_1$ to $P_3$, to launch DoS attack. We assume that an adversary could try to flood these foraged messages whereabouts in a cluster but not move to other clusters. Broadcasting forged $P_1$, an adversary could deprive the energy of sensor nodes in his communication range, but the message does not reach the other sensor nodes. Since the user's communication range is usually small, the attack affects only a small part of the network. Broadcasting forged $P_2$ to $P_5$, an adversary could launch a type of buffer overflow attacks [10], but the damage is confined within the cluster where the attack occurs since forged $P_2$ and $P_5$ are filtered by $M_i$ and forged $P_3$ and $P_4$ are filtered by $\eta^m$ and $G^m_i$. Therefore, these forged messages are not flooded into the other clusters.

### 5.6 Comparison

From now on, we compare the communication and computation overhead of our H-MBA with two simple PKC-based MBAs and $\mu$TESLA.

- **Certificate-based MBA (C-MBA) [17]**: $U$ has $\text{Cert}_U$ and all $\text{msgs}$ are signed by $\text{SK}_U$. A broadcasted message is as follows:

  $$<\text{msg}, T^U_c, \text{Sign}_{\text{SK}_U}(\text{ID}_U, \text{msg}, T^U_c), \text{Cert}_U >$$

- **Certificate-less MBA (Cl-MBA)**: All public keys of legitimate users are stored in each sensor node. A broadcasted message is as follows:

  $$<\text{ID}_U, \text{msg}, T^U_c, \text{Sign}_{\text{SK}_U}(\text{ID}_U, \text{msg}, T^U_c) >$$

- **$\mu$TESLA** [14]: Similar with Cl-MBA, all $\mu$TESLA parameters of users should be stored in each sensor node. A broadcasted message is as follows:

  $$<\text{ID}_U, \text{msg}, \text{MAC}_{\text{SK}_V}(\text{ID}_U, \text{msg}) >$$

For the comparison, we use ECDSA-160 for generating a signature and a certificate and HMAC-SHA1 [7] for generating a MAC. We assume that the $|\text{ID}_U|$ and $|T^U_c|$ are 2 bytes respectively. According to Wunder et al.[18], $|\text{Cert}_U|$ is at least 86 bytes. Additionally, we also assume that a MAC is 20 bytes and a signature is 40 bytes. Therefore, the communication overhead of C-MBA is 128 bytes per a message, 44 bytes in Cl-MBA, and 22 bytes in $\mu$TESLA. The communication overheads of these three schemes are all fixed, but in H-MBA the overhead is fluctuated according to the threshold value $t$. Figure 3 shows the overhead comparison according to $n$.

In C-MBA, each $\eta^m$ must verify two signatures, $\text{Sign}_{\text{SK}_U}(\text{ID}_U, \text{msg}, T^U_c)$ and $\text{Sign}_{\text{SK}_M}(\text{ID}_U, T_{\text{exp}})$, $PK_U$, but Cl-MBA requires just one signature verification and one MAC verification is needed in $\mu$TESLA.

In H-MBA, the computation overhead is decided by $t$ and $n$. According to [18], SHA-1 computation and ECDSA-160 signature verification consume 0.0059 $mJ$ and 45.09 $mJ$ in the MICA2DOT sensor node. We use HMAC-SHA1 which requires two hash operations to compute a MAC and ECDSA-160 as our underlying security primitives for estimating the computation overhead. Figure 4 shows the comparison of the computation overhead.

---

$^3$ We assume that $N_{R_U}$ is equal to $t$
6 Conclusion

In this work, we have proposed an efficient MBA scheme, namely H-MBA, adopting PKC into uTESLA. H-MBA is a hybrid between SKC-based schemes and PKC-based schemes. H-MBA supports a large number of users because the number of users does not affect the memory size of sensor nodes and dynamic addition/deletion of users by using a signature which is difficult to achieve in SKC-based MBA schemes. Furthermore, H-MBA significantly reduces the communication and computation overhead compared with PKC-based MBA schemes by allowing the signature verification only to few sensor nodes in the communication range of users.

References


