Cryptanalysis and improvement of a dynamic and secure key management model for hierarchical heterogeneous sensor networks

Xi-Jun Lin *and Lin Sun †

October 26, 2013

Abstract: In 2012, Alagheband and Aref presented a dynamic and secure key management model for hierarchical heterogeneous sensor networks. They proposed a signcryption algorithm which is the main building block in their key management model. They proved the algorithm is as strong as the elliptical curve discrete logarithm problem. In this work, we study the security of their signcryption algorithm. It is regretful that we found their algorithm is insecure. The adversary can impersonate the base station by sending forged messages to the cluster leaders after capturing the signcrypted messages. Hence, the key management model proposed by them is insecure. Then, we propose an improved signcryption algorithm to fix this weakness.

Key words: Cryptanalysis, Cryptographic Authentication, Network Security

1 Introduction

Wireless sensor networks (WSNs) have the ability to permanently monitor, control and respond to events and phenomena in a specified environment by numerous sensor devices. The structure of WSNs can be divided into two categories: homogeneous and heterogeneous.

Compared with heterogeneous WSNs, the flat architecture of homogeneous WSNs restricts network scalability and performance. In hierarchical heterogeneous WSNs (HHWSNs), two or more kinds of sensors are defined and all the SNs are also separated into a number of clusters and a cluster leader (CL) is assigned to every cluster, and there exists a WSN administrator, usually a base station (BS).

The structure of the HHWSN emphasises the importance of security in the BS-CL and

*X.J.Lin is with the Department of Computer Sciences and Technology, Ocean University of China. Qingdao 266100, P.R.China. email: {linxj77@163.com, 826061296@qq.com, wzgwzq@ouc.edu.cn }

†L. Sun is with the College of Liberal Arts, Qingdao University. Qingdao 266071, P.R.China. email: sunlin9@126.com

CL-CL links. HHWSN requires a method to communicate securely between the BS and the CLs prior to the SNs registration.

The first crucial function to achieve security in HHWSN is key management because the SNs and the CLs need common keys to protect the communication by employing cryptography algorithms. In wireless sensor networks, the key management protocols are classified into three categories: symmetric, asymmetric and hybrid models [8].

Symmetric schemes are studied to load keys in the SNs prior to the deployment phase, which suffer from some problems [2, 3, 7]. In recent years, elliptical curve cryptography (ECC) and identity-based cryptography (IBC) are employed in asymmetric schemes, which are more flexible than the symmetric schemes. However, These schemes is very heavyweight for sensor networks.

In HHWSNs, hybrid schemes are designed for different kinds of nodes. As the computational cost of the CLs is more than that of the SNs, the CLs usually have more obligations such as data aggregation, control and cluster leading. With the recent progress in ECC, applying public key cryptography in WSNs becomes more practical [4–6].

In 2012, Alagheband and Aref proposed a dynamic and secure key management model for hierarchical heterogeneous sensor networks [1]. A signcryption algorithm was suggested as the main building block in their key management framework for CL-CL and BS-CL links. They proved that this algorithm is as strong as the elliptical curve discrete logarithm problem (ECDLP).

In this paper, we attack this algorithm to point out that an adversary can impersonate the BS by sending forged messages to CLs. The key management model is insecure since the signcryption algorithm which is the main building block of the model is insecure. Then, we propose an improved signcryption algorithm to fix this weakness.

2 Review of Alagheband and Aref's Signcryption Algorithm

In this section, we review Alagheband and Aref's signcryption algorithm as follows. The parameters used in this signcryption algorithm is given in Table 1.

Table 1: List of notations

Parameter	Definition
G	base point of elliptical over E with order n
n	order of point G, where n is a prime, $n \times G = \mathcal{O}$ and $n > 2^{160}$ (The symbol '$\times$' denotes elliptical curve point multiplication.)
$\mathcal{O}$	point of E at infinity
P_{cl_i}, U_{cl_i}	CL_i's private key and public key
P_{bs}, U_{bs}	BS's secret key and public key
$E_k(\cdot)$, $D_k(\cdot)$	lightweight symmetric encryption/decryption algorithm with key k
$meta$	a public and fixed message
K_N	network key [128bit] (only for registration)
K_{SN_i}	SN key
K_{cl}	cluster key
Sgn	signcryption algorithm
$t.s.$	timestamp
H	a lightweight and secure one-way hash function

The BS generates public-private keypairs based on the ECDLP. These keys are assigned to all the nodes in asymmetric key management, or just CLs in the hybrid key management schemes. The BS performs the following terms for key generation:

- Choose P_i, a random number, as a private key, $P_i \in [1, q-1]$, where q is a large strong prime and the order of the elliptical curve, and $1 \leq i \leq N$ (N is the number of CLs).
- Compute $U_i = P_i G$ as a public key.
- Embed (P_i, U_i) securely in every CL_i after deployment and save the same in the database of the BS.

Likewise, the BS has two keys U_{bs} and P_{bs} ($U_{bs} = P_{bs}G$). P_{bs} is the secret key of the BS that the CLs and the SNs do not know indefinitely. U_{bs} is embedded in the CLs to execute the signcryption algorithm. CL_i verifies the authenticity of the BS with the aid of U_{bs}. The BS knows the public key of all the CLs.

When the BS wants to send authenticated messages to CL_i with the signcryption algorithm, they can perform the following procedures.

- The BS picks a random number r_i and performs the following steps:

- Compute $R = r_i G = (r_1, r_2)$
- Compute $K = r_i U_{cl_i} = (k, l)$
- Compute $C = E_k(m)$
- Compute $h = H(C \| r_1)$
- Compute $e = H(hr_i)G$
- Compute $s' = P_{bs} - H(hr_i) \pmod{q}$ and $s = s' + h$.

Then, the BS sends (C, R, s, e) to CL_i.

- After receiving (C, R, s, e), CL_i computes as follows.
  - Compute $K = P_{cl_i} R = (k, l)$
  - Compute $m = D_k(C)$
  - Compute $h = H(C \| r_1)$
  - Compute $s' = s - h$

  Then, If $s'G + e = U_{bs}$, the BS is authenticated and CL_i accepts m as plaintext and k as common key.

Furthermore, Alagheband and Aref proved that the cluster nodes of the HHWSN are untraceable and the unsigncryption of their sent signcrypted message is infeasible even though the CL's private key is corrupted. The adversary opts for two cluster leaders $\{CL_0, CL_1\}$ where they sent several signcrypted messages. Depending on a randomly chosen bit $b \in \{0, 1\}$, the adversary is given CL_b from the set $\{CL_0, CL_1\}$. The signcryption scheme is untraceable unless it can truly guess b.

In SNs registration phase, after the WSN deployment, the SNs should find the nearest CL for registration in its cluster. One SN_i will be enroled in the nearest CL through the following steps:

- Every SN_i sends $\alpha = ID_{SN_i}$ and $\beta = H_{K_N}(ID_{SN_i})$ to the nearest CL by means of a keyed one-way hash function (H).

- The CL verifies whether $H_{K_N}(\alpha)$ is equal to β. If it is true, the process goes to the next step; otherwise, the CL rejects the message and alerts the BS.
- The CL computes $Sgn(ID_{SN_i}, t.s.)$ with its private key and sends it to the BS.
- As soon as the unsigncryption and versification phases are completed, the BS responds to the CL by $Sgn(ID_{SN_i}, K_{SN_i}, t.s.)$.
- The CL saves the ID_{SN_i} and K_{SN_i} after verification, and the SN_i becomes a legitimate member of the CL's cluster.
- The CL uses a lightweight symmetric encryption algorithm to generate ciphertext $\gamma = E_{K_{SN_i}}(meta \| K_{cl})$.
- The SN_i uses the lightweight symmetric decryption algorithm to compute $D_{K_{SN_i}}(\gamma)$, where the secret key K_{SN_i} has been embedded in SN_i during a pre-deployment phase. SN_i verifies provided the first part of $D_{K_{SN_i}}(\gamma)$ is equal to 'meta'. If it is true, the SN_i generates K'_N from K_N with a lightweight one-way hash function ($K' = H(K_N)$).

There is periodic authentication mechanism between SNs and CLs in every cluster. Every CL regularly authenticate the SNs which have been registered in its cluster. This mechanism is as follows.

- CL periodically sends query $\{t.s., ID_{SN_i}, H_{K_{SN_i}}(ID_{SN_i}, t.s.)\}$ to SN_i.
- SN_i checks the truth of the query and sends $\{H_{K_{SN_i}}(K'_N, t.s.'), ID_{SN_i}, t.s.'\}$

CL checks the truth of the query sent by SN_i. If it is correct, SN_i is a legitimate SN in CL's cluster; otherwise, CL revokes SN_i immediately.

Alagheband and Aref proved that their model can against node capture attack, i.e. even if a sensor node SN_t is compromised and its secret keys (K'_N, K_{SN_t}, K_{cl}) is revealed, the SNs registration mechanism and periodic authentication mechanism can prevent the adversary from penetrating the WSN.

3 Cryptanalysis of Alagheband and Aref's Key Management Model

In this section, we firstly prove that Alagheband and Aref's signcryption algorithm is insecure, i.e. an adversary can send a forged message to CLs by impersonating the BS. Then, we propose the node capture attack by employing this forgery to Alagheband and Aref's key management model, i.e. if a sensor node is compromised and its secret keys is revealed, the adversary can penetrate the WSN.

3.1 Cryptanalysis of Alagheband and Aref's Signcryption Algorithm

The adversary can send a forged message m' to CL_i by impersonating the BS after he captures (C, R, s, e) between the BS and CL_i. The forgery process is as follows.

- Compute $h = H(C\|r_1)$, where r_1 is the x-coordinate of R, and then compute $s' = s - h$.
- Pick two random numbers r, t and perform as follows.
  - Compute $R_1 = rG = (r'_1, r'_2)$
  - Compute $K_1 = rU_{cl_i} = (k', l')$
  - Compute $C_1 = E_{k'}(m')$
  - Compute $h_1 = H(C_1\|r'_1)$
  - Compute $e_1 = e - tG$
  - Compute $s'_1 = s' + t \pmod q$ and $s_1 = s'_1 + h_1$.
- Send (C_1, R_1, s_1, e_1) to CL_i.

The correctness of the above forgery can be proved as follows.

After receiving (C_1, R_1, s_1, e_1), CL_i computes and checks it with the following steps.

- Compute $K_1 = P_{cl_i}R_1 = rP_{cl_i}G = rU_{cl_i} = (k', l')$.
- Decrypt C_1 by employing $D(\cdot)$ with k' as key, i.e. $m' = D_{k'}(C_1)$.

- Compute $h_1 = H(C_1 \| r'_1)$ and $s'_1 = s_1 - h_1$.

It is clear that

$$\begin{aligned} s'_1 G + e_1 &= (s' + t)G + (e - tG) \\ &= s'G + e \\ &= U_{bs} \end{aligned}$$

From above we can see that the adversary can impersonate the BS by sending forged messages to the CLs.

3.2 Node Capture Attack

If there is a sensor node (only one is enough) which is compromised, the adversary can penetrate the WSN with the above forgery method. After compromising the sensor node SN_t and revealing its secret keys (K'_N, K_{SN_t}, K_{cl}), the adversary performs as follow.

In SNs registration phase, the adversary can cheat the CLs into communicating with an illegitimate sensor node by impersonating the BS. The adversary performs as follows.

- When SN_i sends $\alpha = ID_{SN_i}$ and $\beta = H_{K_N}(ID_{SN_i})$ to its nearest cluster for registration, the adversary captures these messages, and then impersonates SN_i by sending α and β to another cluster leader CL_j.

- CL_j verifies $H_{K_N}(\alpha) = \beta$ and computes $Sgn(ID_{SN_i}, t.s.)$ with its private key, then sends it to the BS.

- The adversary impersonates the BS and sends $Sgn(ID_{SN_i}, K'_{SN_i}, t.s.)$ to CL_j since the adversary knows $\alpha = ID_{SN_i}$ and t.s. can be guessed easily, where K'_{SN_i} is picked by the adversary as a forged SN_i's key.

- CL_j accepts $Sgn(ID_{SN_i}, K'_{SN_i}, t.s.)$ since the adversary has the ability to generate a valid signcryption message, and CL_j saves ID_{SN_i} and K'_{SN_i} after verification. In this step, CL_j accepts SN_i as a legitimate sensor node in its cluster.

From above we can see that CL_j accepts the adversary as a legitimate sensor node(i.e. SN_i) and the shared key is K'_{SN_i}.

In the periodic authentication mechanism, the adversary can reply CL_j's query as follows.

- CL_j periodically sends query $\{t.s., ID_{SN_i}, H_{K'_{SN_i}}(ID_{SN_i}, t.s.)\}$ to the adversary who impersonates SN_i.
- The adversary replies $\{H_{K'_{SN_i}}(K'_N, t.s.'), ID_{SN_i}, t.s.'\}$ since the adversary knows the shared key K'_{SN_i} and the revealed secret key K'_N.

According to the above node capture attack, we conclude that the key management model is insecure if the signcryption algorithm is insecure.

4 Improvement on Alagheband and Aref's Signcryption Algorithm

Our improved signcryption algorithm is based on Alagheband and Aref's algorithm. The notations are the same with their algorithm.

When the BS wants to send authenticated messages to CL_i with the improved signcryption algorithm, they can perform the following procedures.

- The BS picks a random number r_i and performs the following steps:
  - Compute $R = r_i G = (r_1, r_2)$
  - Compute $K = r_i U_{cl_i} = (k, l)$
  - Compute $C = E_k(m)$
  - Compute $s = P_{bs} H(C \| r_1 \| r_2) + r_i \pmod q$.

  Then, the BS sends (C, R, s) to CL_i.
- After receiving (C, R, s), CL_i computes as follows.
  - Compute $K = P_{cl_i} R = (k, l)$

– Compute $m = D_k(C)$

Then, If $sG = H(C\|r_1\|r_2)U_{bs} + R$, the BS is authenticated and CL_i accepts m as plaintext and k as common key.

Theorem 1 *The improved signcryption algorithm is secure if the ECDLP and the elliptical curve Diffie-Hellman problem (ECDHP) are at least a computationally infeasible problem and H is a random oracle.*

Proof: In the improved signcryption algorithm, the secret key k is the x-coordinate of point K. Without knowing r_i and P_{cl_i}, the adversary cannot compute $K = r_i U_{cl_i} = P_{cl_i} R$ because of the difficulty of ECDLP and ECDHP. In another hand, without knowing P_{bs} and r_i, the adversary cannot forge a legal authenticated message because of the difficulty of ECDLP. □

Theorem 2 *With the improved signcryption algorithm, the cluster nodes are untraceable and the unsigncryption of their sent signcrypted message is infeasible even though the CL's private key is corrupted.*

Proof: The adversary can eavesdrop on the transcript of the executed protocol between CL_b and either BS or CL_k. Although it knows $\{CL_0, R_0, s_0\}$, $\{CL_1, R_1, s_1\}$ and $\{P_{cl_0}, P_{cl_1}\}$, since the private key of CL is not needed in the verification of the signcrypted message (i.e. the equation $sG = H(C\|r_1\|r_2)U_{bs} + R$), it can only distinguish b with flip coin probability and the adversary's success is negligible. □

With the improved signcryption algorithm, in SNs registration phase, the adversary cannot generate the right forged signature $Sgn(ID_{SN_i}, K'_{SN_i}, t.s.)$ in step 4 after receiving $Sgn(ID_{SN_i}, t.s.)$ from CL_j in step 3 since the improved algorithm is secure. Then he cannot cheat CL_j into receiving K'_{SN_i} as a legal key in step 5. Finally, The node capture attack launched by the adversary who wants to penetrate the WSN by impersonating the sensor node SN_i will fail.

We conclude that with the improved algorithm it is not possible for an adversary to impersonate a BS and in the following trick a CL to accept illegal SN.

5 Conclusion

In this paper, we point out that Alagheband and Aref's signcryption algorithm is insecure, i.e., an adversary can impersonate the BS by sending forged messages to CLs. The key management model proposed by them is insecure since the signcryption algorithm which is the main building block of the model is insecure. Then, we propose an improved signcryption algorithm to fix this weakness.

References

[1] Alagheband, M.R., and Aref, M.R.: 'Dynamic and secure key management model for hierarchical heterogeneous sensor networks', IET Information Security, 2012, 6, (4), pp. 271-280

[2] Camtepe,S.A., and Yener, B.: 'Combinatorial design of key distribution mechanisms for wireless sensor networks', J. ACM/IEEE Trans. Netw., 2007, 15, (2), pp. 346-358

[3] Eschenauer,L., and Gligor,V.D.: 'A key management scheme for distributed sensor networks', Ninth ACM Conf. on Computer and Communication Security, November 2002, pp. 41-47

[4] Gura, N., Patel, A., Wander, A., Eberle, H., Shantz, S.C.: 'Comparing elliptic curve cryptography and RSA on 8-bit CPUsc'. The Sixth Int. Workshop on Cryptographic Hardware and Embedded Systems, Boston, Massachusetts

[5] Malan, D.J., Welsh, M., Smith, M.D.: 'A public-key infrastructure for key distribution in TinyOs based on elliptic curve cryptography'. First Annual IEEE Communications Society Conf. on Sensor and ad hoc Communications and Networks, IEEE SECON, L.B, 2004

[6] Oliveira a, L.B., Aranha, D.F., Gouvea, C.P.L., et al.: 'TinyPBC: Pairings for authenticated identity-based non-interactive key distribution in sensor networks', Comput. Commun. J. Elsevier, 2011, 34, (3), pp. 485-493

[7] Yu, Z., Guan, Y.: 'A robust group-based key management scheme for wireless sensor networks'. IEEE Wireless Communications and Networking Conf. (WCNC), New Orleans, LA, USA, 2005, pp. 137

[8] Zhang, J., Varadharajan, V.: 'Wireless sensor network key management survey and taxonomy', J. Netw. Comput. Appl., 2010, 33, pp. 63-75