Secure Code Dissemination Algorithm Based on Hash Digest and Layered Strategy in WSNs

1Guoping Zhang and 2Mande Xie
1Faculty of Informatics and Electronics, Zhejiang Sci-Tech University, Hangzhou, 310018, Zhejiang, China
2College of Computer Science and Information Engineering, Zhejiang Gongshang University, Hangzhou, 310018, Zhejiang, China

Abstract: In Wireless Sensor Networks (WSNs), code dissemination faces threats from both external attackers and potentially compromised nodes. Security thus becomes a critical requirement for code dissemination protocols. In this study, we propose a secure code dissemination algorithm based on the hash digest and the layered strategy. Compared to the existing algorithms, our main contributions are two-folds: (1) All data packets can be roughly and immediately verified by their hash digest. All image blocks can be exactly verified by their hash value. By the double independent verification, the code image can be securely updated. Compared to directly transmitting all hash value, the message traffic and delay are reduced (2) The hash chain is transmitted layer by layer. The below hash chain is request only if the corresponding image block is failed to verify in the above layer. Therefore, without loss of security, the message traffic is greatly reduced.

Key words: Wireless sensor networks, code dissemination, hash digest, code image

INTRODUCTION

Wireless Sensor Networks (WSNs) have drawn the attention of the research community in the last few years. Lots of new applications have been deployed. Up to now, most deployed applications measure scalar physical phenomena such as temperature, pressure, humidity, or carbon dioxide concentration. Program Image (PI) updates have become very necessary in WSNs because they may be required for bug fixing or to provide new functionalities after WSNs have been deployed. However, if WSNs are large scale or deployed in the harsh environment where individual sensor nodes, once deployed, are practically inaccessible, it is impossible to manually reprogram all nodes. Because the online code dissemination can remotely reprogram the nodes via wireless communication, it becomes a promising technique. In general, online dissemination protocol not only should be able to fulfill code update functionality, but also they need to be reliable and robust against different adverse network conditions, efficient in terms of speed, scalable in terms of both network size and code size propagated, etc. In addition, many applications about WSNs are often deployed in the hostile environments where there may be malicious attacks against WSNs, code dissemination faces threats from both external attackers and potentially compromised nodes. Security thus becomes another critical requirement for network protocols. Due to the sheer number of sensor nodes and the broadcast nature of wireless links, it is often desirable for a base station to broadcast commands and data to the network. The authenticity of such commands and data is critical for the correct operation of sensor networks. If convinced to accept forged or modified commands and data, normal sensor nodes are force to verify and/or forward them. As a result, their limited battery power is exhausted and the intended purposes of the network cannot be fulfilled. In this study, we propose a secure code dissemination algorithm based on hash digest and layered strategy for WSNs. Compared to the existing algorithms, our algorithm reduces the message traffic and delay are reduced without loss of security.

RELATED WORKS

Hui and Culler (2004) is the earliest and most sophisticated code dissemination protocol. It is currently distributed as part of TinyOS and has been the defacto

Corresponding Author: Mande Xie, College of Computer Science and Information Engineering, Zhejiang Gongshang University, Hangzhou, 310018, Zhejiang, China
standard network reprogramming for WSNs. Deluge lays down some design principles widely used by the latter protocol. However, Deluge does not take account of any security issues and is vulnerable to lots of attacks. Hence, based on the framework of Deluge, lots of secure network reprogramming algorithms are proposed. The lecture (Lamigan et al., 2005; Dutta et al., 2006; Deng et al., 2006; Xie, 2011) propose some hash chain-based schemes. Sluice (Lamigan et al., 2006) integrates signature and cryptographic hash functions to provide efficient authentication for network reprogramming. Sluice built the hash chain in page level and only perform authentication when an entire page is received. As a result, it cannot authenticate a packet immediately after the packet is received. Different to Sluice, Dutta et al. (2006) build a hash chain in packet level to authenticate a packet immediately after it is received. However, this approach requires that the packets are received in order and in fact, it is normal that a packet reached the target node out of order in WSNs. Deng et al. (2006) build a hash chain in page level and then build a Merkle hash tree for each page. This approach can authenticate a packet immediately after it is received and also allow that the packet is received out of order. But, it introduces an excessive traffic for the Merkle hash trees. According to the Deny of Service (DoS) attacks on Deluge, Park et al. (2007) propose two schemes in the way of providing the recovery method for verification processes at packet loss, using supplementary hashing: Redundant hash scheme and page digest scheme. Dong et al. (2008) present two filtering techniques, a group-based filter and a key chain-based filter, to handle DoS attacks against signature verification. Tan et al. (2013) firstly integrate confidentiality and DoS-attack-resistance in a multi-hop code dissemination protocol. At the same time, they propose countermeasures against both types of DoS attacks based on requests. Hyun et al. (2008) propose DoS-resistant network reprogramming system named Seluge. Seluge can provide immediate authentication of each packet upon receipt, without disrupting the efficient propagation mechanisms used by Deluge. According to the vulnerability of reboot and erase command, Liu et al. (2009) propose a hash chain-based scheme to provide the security of image management.

In the latest lectures, some algorithms based on network coding are proposed (Firooz and Roy, 2013; Salkuyeh et al., 2013; Li et al., 2013). Firooz and Roy (2013) analyze the time needed to diffuse information throughout a network when network coding is implemented at all nodes. According to V2V (vehicle-to-vehicle) and V2I (vehicle-to-infrastructure) data transfer, Salkuyeh et al. (2013) propose an efficient method of message dissemination using rateless coding. They claim by employing rateless coding at road side units and using vehicles as data carriers, messages can be propagated efficiently. Li et al. (2013) propose a trajectory-based network coding (TBNC) method to disseminate data which is used in mobile wireless sensor network (MWSN). However, these algorithms based on network coding do not consider the security performance. Zeng et al. (2012) propose a novel code dissemination scheme, which integrates immediately authentication into Fountain codes. Their analysis shows that proposed scheme can provide code image confidentiality, bogus code image protection, DoS protection and out-of-order delivery-tolerant property. Almost of all code dissemination protocols are based on the centralized approach in which only the base station has the authority to initiate code dissemination. According to this situation, He et al. (2012a; b) develop a secure and distributed code dissemination protocol named DiCode and SDRP to disseminate code images in a distributed manner which allows multiple authorized network users to simultaneously and directly update code images on different nodes without involving the base station.

**PREPROCESSING FOR CODE DISSEMINATION**

For the ease of presentation, all symbols used in this study are firstly introduced in Table 1.

The Fig. 1 shows the basic principle of the preprocessing. The most important idea of the

<table>
<thead>
<tr>
<th>Table 1: All symbols used in this study</th>
</tr>
</thead>
<tbody>
<tr>
<td>H0: The length of the hash value</td>
</tr>
<tr>
<td>ML: The maximum payload size</td>
</tr>
<tr>
<td>H: The hash value of the image block i</td>
</tr>
<tr>
<td>HNS: The hash value of the next element in the hash chain</td>
</tr>
<tr>
<td>HD: The hash digest of the data packet i</td>
</tr>
<tr>
<td>HDS: The length of the hash digest</td>
</tr>
<tr>
<td>[B]: The length of the image block</td>
</tr>
<tr>
<td>[sub_B]: The size of the image sub-block i</td>
</tr>
<tr>
<td>FHC: The hash value of the first element in the hash chain</td>
</tr>
<tr>
<td>n: The number of the image block</td>
</tr>
<tr>
<td>sub_n: The number of the sub-block in an image block</td>
</tr>
<tr>
<td>max_sub_n: The maximum number of the sub-block in an image block</td>
</tr>
<tr>
<td>PI: The size of program image</td>
</tr>
<tr>
<td>[Sg]: The size of digest signature</td>
</tr>
<tr>
<td>[Header]: The size of the header information</td>
</tr>
<tr>
<td>GetDigest: Get the hash digest from a hash value</td>
</tr>
<tr>
<td>BuildGenChain: Build the general chain data structure</td>
</tr>
<tr>
<td>InsertToChain: Insert the hash value into a node and the size of a node is no more than ML</td>
</tr>
<tr>
<td>InsertToHDNode: Insert the hash digest into a node and the size of a node is no more than ML</td>
</tr>
<tr>
<td>InsertToLinkNode: Insert the FMC into a node and the size of a node is no more than ML</td>
</tr>
<tr>
<td>BuildHashChain: Build the hash chain based on the general chain</td>
</tr>
</tbody>
</table>
The preprocessing in the layer 1 is computed as Eq. 1:

$$\text{max\_n} = \left\lfloor \frac{\text{ML}}{\text{FHC}} \right\rfloor + \left\lfloor \frac{\text{ML} - |\text{SigO}| - |\text{Header}|}{\text{FHC}} \right\rfloor - 2 \quad (1)$$

where, the first item and the second item in the right side indicate the number of the hash value contained in the second node and the first node in the hash chain, respectively. The item 2 in the right indicates the space occupied by HN in the first and second node.

The maximum number of the data packet is computed as Eq. 2:

$$p = \left\lfloor \frac{\text{PI}}{\text{ML}} \right\rfloor \quad (2)$$

The number of the image block is computed as Eq. 3.

If the...
Preprocessing algorithm()
{
1) max_n = \left\lceil \frac{\text{ML}}{|H()|} \right\rceil + \left\lceil \frac{\text{ML}-|\text{Sig()}|-|\text{Header}|}{|H()|} \right\rceil - 2

2) p = \frac{\text{PL}}{\text{ML}}

3) if (\text{PL} < \text{max}_n \times \text{ML})

4) n = \frac{\text{PL}}{\text{ML}}

5) else
6) n = \text{max}_n
7) Divide the program image into n image blocks;
8) BuildGenChain();
9) Compute H_i = H(B_i);
10) InsertToNode(); // insert \textit{H}_n node
11) Devide the program image into p data packet;
12) Compute H_d = GetH(1DHD(p));
13) InsertToIDNode(); // insert \textit{H}_d node.
14) j = 0;
15) While (j<n)
16) FHC = SubBlockProcess (B_i);
17) InsertToFHCNode;
18) BuildHashChain();
}

which indicates the size of each image block in the layer 1 is more than the maximum payload. Otherwise, n = max_n, which indicates the layer 2 is not required:

\[
\text{n} = \begin{cases} \frac{\text{PL}}{\text{ML}} & \text{if } \text{PL} > \text{max}_n \times \text{ML} \\ \text{max}_n & \text{otherwise} \end{cases}
\]  \hspace{1cm} (3)

The line 7-13 divide the PL and generate the HNodes and HDNodese. The line 14-17 iteratively divide the image block layer by layer. The SubBlockProcess is a subroutine which recursively divides the image block and its work flow is shown in the Fig. 3. The BuildHashChain subroutine builds the hash chain based on the general chain.

If |B_i|<ML, the further division is no t required and the subroutine is terminal. In Fig. 3, the line 3-7 compute some important parameters and the computation method is similar to the Fig. 2. The line 8-17 is also similar to the Fig. 2. The only difference is there are no HDNode in Fig. 3.

Assume that we employ the current generation of sensor platforms that use IEEE 802.15.4 compliant radios, such as MicaZ and TelosB. We also use the 64-bit truncation of SHA-1 as the hash function. It provides sufficient pre-image resistance and has been used previously (Dutta et al., 2006). For digital signature, we use ECDSA over the 160-bit elliptic curve secp160 k1, which is defined in (CertiCom Research, 2000). The maximum payload size in the IEEE 802.15.4 standard is 102 byte and the size of hash digest is 8 bits and header information occupies 8 bytes. Hence, the max_n = 102/8=(102-40-8)-2 - 14. If the PL is 20 kB, p = 20*1024/102 = 201. Because 20*1024+14*102, n = max_n = 14 and the size of image block is |B_i|=20*1024/14 = 1463. The number of nodes in the hash chain in the layer 1 is 2+2+2 = 6. In the SubBlockProcess routine:

\[
\text{max}_n = 2^3 = 22
\]

\[
\text{sub}_n = \frac{1463}{102} = 15
\]

And:

\[
\text{|Sub IB_i|} = \frac{1463}{15} = 98
\]

Because |Sub IB_i|<102, the one time iteration is enough and the layer 3 is not required.
DATA INTEGRALITY CHECKING AND CODE RETRANSMISSION

When the Base Station (BS) wants to update the code image running in the sensor nodes, it firstly broadcasts the digital signature packet. Then all of nodes in the hash chain of the layer 1 are transmitted in sequence. The integrity of the data packet including the first node in the hash chain is guaranteed by the digital signature. Upon receiving a packet data including the nodes in the hash chain, sensor nodes can immediately verify its integrity because its hash value has been received in the previous data packet.

After the sensor nodes successfully receive the hash chain, the data packets can be roughly and immediately verified when they arrived out of order. The steps of data integrity checking are as follows:

- Receive and verify the digital signature
- Receive and verify the hash chain
- Upon receiving a data packet, the sensor nodes compute its hash value and extract its hash digest. Compare it with the hash digest received in the hash chain and set the bit vector where each bit in the vector corresponds to a data packet of code image. If the hash digest is successfully verified, the corresponding bit in the bit vector is set 1. Otherwise, the corresponding bit in the bit vector is set 0
- If all data packets in an image block are received and some corresponding bits are 0, sensor nodes request BS to retransmit the corresponding data packet
- If all data packets in an image block are received and all the corresponding bits are 1, the hash value of the image block is computed. Compare it with the hash value received in the hash chain. If the verification of the hash value passes, the image block is successfully received. Otherwise, request BS to send the hash chain of the image block in below layer and retransmit all data packets in the image block. The integrity of the first node in the hash chain of the image block in next layer is ensured by the FHC inserted in the above hash chain.

In a brief, the hash digest provides the rough and immediate verification means and the hash value of the image block provide the exact verification means. By the double independent verification, the code image can be securely updated.

CONCLUSION

Code dissemination faces threats from both external attackers and potentially compromised nodes. Security thus becomes a critical requirement for code dissemination protocols. In this study, we propose a secure code dissemination algorithm based on the hash digest and the layered strategy. In this algorithm, code image is divided into a series of data packets which is directly transmitted in the wireless channel. Then, the hash digests of these data packets are extracted from their hash value and are embedded in the hash chain in the layer 1. All data packets can be roughly and immediately verified by their hash digest. In additional, the code image is divided into some image blocks. Their hash value are computed and embedded in the hash chain in the layer 1. Each image block can be exactly verified by their hash value. If the image block has been successfully verified or several data packets lied in this image block are failed to verify, the hash chain of this image block in the layer 2 is requested. The below hash chain is required only if the corresponding image block is failed to verify in the above layer. Therefore, without loss of security, the message traffic is greatly reduced by the hash digest and the layered strategy.

The code dissemination algorithms based on the network coding are very popular and efficient in latest year. They can reduce the energy consumption and improve the robustness of code dissemination in noisy environment. However, in our algorithm, we do not employ any networking coding method. In fact, integrating the secure frame with network coding is still open problem. Hence, in the further, proposing a secure code dissemination algorithm with networking coding is our important work. In addition, we will further improve the efficiency of secure code dissemination.

ACKNOWLEDGMENTS

This study was partially supported Grant No. 2011C14024 from Science and Technology Department of Zhejiang Province Program and Grant No. Z201122764 from Educational Committee of Zhejiang Province and Grant No. SB1105001-E from Experimental Teaching Demonstration Center of Zhejiang Province and Grant No. 2010R50041 from Key Innovation Team of Science and Technology Department of Zhejiang Province.

REFERENCES


