Secure data Discovery and dissemination based on short-length public key scheme in Wireless sensor networks

Yan Wang
East China Normal University
No.3663,Zhongshanbei Road
Putuo District,Shanghai,China
boliangzai@foxmail.com

Daojing he
East China Normal University
No.3663,Zhongshanbei Road
Putuo District,Shanghai,China
djhe@sei.ecnu.edu.cn

ABSTRACT

In Wireless Sensor Networks, WSNs, dissemination is typically used to query nodes, send commands, and reconfigure the network. In fact, WSNs are always deployed in harsh and open environment, it is important to judge messages as if they are from trusted source. Authentication can be satisfied by Public Key Cryptography (PKC). But PKC operations are expensive in terms of bandwidth, computing and storage consumption. To reduce costs, we propose a lightweight and Dos-resilient broadcast authentication mechanism using short-length public/private keys and presents its application in securing Drip protocol, an open source message dissemination protocol. We compare our mechanism with traditional 160-bit ECC public-key schemes, and show that our scheme can achieve a significant improvement on energy consumptions. Besides, we believe that our mechanism is able to fulfill the broadcast authentication requirements of WSNs.

CCS Concepts

•**Computer systems organization** → **WSNs;** • **Cryptography** → *Public key;*

Keywords

Public key; message dissemination, Dos resilience

1. INTRODUCTION

Wireless sensor networks are being used in a wide variety of applications, such as military sensing and tracking, industrial control. Sensor networks are composed of one or more base stations and a number of sensor nodes. Obviously, wireless industrial control sensor networks are broadcast networks and channels are shared among all users in the network.

ASIA CCS '17 Abu Dhabi, UAE

DOI: 10.475/123_4

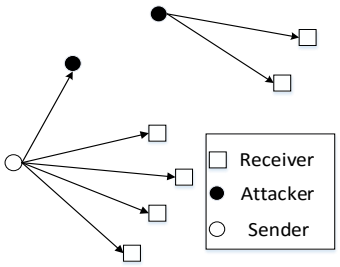

Figure 1: Attack scenario

As shown in Fig.1, when deployed in hostile environments, sensor networks are susceptible to a variety of attacks. For example, adversaries are able to easily intercept the messages between base stations and sensor nodes, impersonate the base stations to deceive sensor nodes.

Many countermeasures such as PKC, have been developed to prevent adversaries from impersonating base stations. But these Countermeasures make sensor networks an easy target of Dos attacks. We provide an example to illustrate this problem.

First, as shown in Fig.1, sensors are measuring temperature, the control center may want sensors to measure humidity of the atmosphere instead. In this case, control center need broadcast values to reconfigure the whole network. In mentioned network, communications are based on IEEE 802.11.4 standard. It is open and adversaries can broadcast a large number of bogus messages, exhausting resources of sensor nodes.

In order to secure message broadcast in industry control sensor networks. This paper has the following major contributions:

1 We first investigate the requirements for a secure broadcast mechanism in industry control sensor networks, and then show some security weaknesses and efficiency problems of the existing broadcast authentication mechanisms.

2 We propse a secure, lightweight, robust, Dos-resilient authentication mechanism. We make it an secure extentsion of Drip and refine our protocol to reduce bandwidth, computing, energy consumption.

3 We implement our protocol on telosb platform. And experimental results in charts demonstrate high efficiency and high performance.

Organization The organization of our paper is as follows: Section II, we discuss the related work, and Section III describes the security requirements and vulnerability of industrial control sensor networks. In Section IV, We present our scheme in detail. Section V analyses the security and efficiency properties of our protocol. Section VI describes the implementation and experimental results of the proposed protocol via real sensor platforms. Finally, Section VII concludes this paper.

2. RELATED WORK

2.1 Existing Work on Broadcast Authentication

Providing security measures for smart grids encounters many challenges. The primary one is the limited computing, communication and storage capabilities of receivers. A message authentication code (MAC) is an authentication tag derived by applying an efficient symmetric cryptographic primitive for two-party authentication, but it is not suitable for broadcast authentication. Because the sender and its receivers share the same secret key, any one of the receivers can impersonate the sender and forge messages to other receivers. That is, both sender and receivers can authenticate messages. As to asymmetric mechanism, only the sender can sign messages while the receivers can only verify messages, is more suitable for broadcast authentication. More exactly, the sender signs each packet individually using digital signature technique and each receiver verifies the signature before processing the packet. The signature is vulnerable to Dos attacks. That is, the adversary may flood a large number of illegal signature messages to the receivers to exhaust their resources and render them less capable of serving legitimate users.

To provide authentication, some researchers proposed TESLA and its various extensions to authenticate broadcast packets in a network. It employs symmetric cryptography primitives with delayed key disclosure to authenticate the claimed sender of a broadcast packet, i.e., the key used to authenticate a message is disclosed in the next message. However, these methods requires time synchronization for the whole network, which leads to more complicated measures to secure synchronization. In addition, the receivers cannot authenticate packets immediately, but must wait until the respective keys are disclosed, resulting in excessive verification latency and a requirement to store unauthenticated packets. More importantly, they are vulnerable to a DoS attack due to the authentication delay. Through a simple flooding attack, each receiver has to buffer all forged messages claimed to be from the sender until it receives the disclosed key. Another technique uses one-time signature schemes [10]. Unfortunately, such schemes suffer from large key sizes and a limited number of uses per key.

Recently, a message broadcast authentication protocol [11] has computed a hash function of the last message, and then, attached this hash to the previous message. The packet with the hash appended is then itself hashed and this hash value is attached to the previous packet. This process is continued until the first packet is reached. The first packet is signed with the private key of the sender. Each receiver verifies the first packet with the senderâĂŹs public key, then recursively authenticates each packet using the hashed value from the predecessor packet until all message update has been successfully received. Another approach has been adopted by some researchers [12], [13] who use a Merkle tree instead of recursive hashing for packet authentication. However, in most cases, whether they are recursive or a variant such as a Merkle tree, all are based on the use of a digital signature to sign the first packet. There are three problems associated with these approaches. Firstly, these approaches require the number and content of data packets to be known in advance of the message dissemination process, since they construct a hash chain (or a hash tree) which starts from the last data packet and works back to the first. This requirement limits the expansion of these approaches to some message dissemination applications where often all data items to be broadcast cannot be available beforehand. Secondly, hash chain approach must depend on good communication channel. Once the channel conditions get worse, data packet canâĂŹt be carried out in accordance with the established order receiving, it is unlikely to validate data integrity. Thirdly, the hash tree method adds additional overhead due to the transmission of part of a hash tree for each data item. Energy is an extremely scarce resource on any device while radio communication consumes the most amount of energy.

2.2 Review on Message Dissemination

Drip is one of the most popular message dissemination protocol in sensor networks. In Drip, each message is represented as a 3-tuple (key, seqno, data), where key uniquely identifies the variable to be updated, data denotes the disseminated data item (e.g., parameter, command or query), and seqno indicates if the data item is old or new (the larger the seqno, the newer the data). In the Drip implementation, key and seqno are 2 bytes and 4 bytes long, respectively. Drip disseminates each data item with a separate instance of Trickle algorithm [14]. Once new data are injected to a receiver by the sender, they will be disseminated by Trickle quickly.

Although we have used Drip for our reference implementation, our mechanism can be extended to other popular broadcast protocols, with the assumptions listed in Section III.C. In the rest of this paper, unless otherwise specified, we assume that Drip is used as the message dissemination protocol

3. PROBLEM DEFINITION

3.1 Network Model

We consider a broadcast group involving one sender (S) and a group of receivers (R_i). Each message is delivered from S to each R through lossy and insecure PLC network, as illustrated in Fig. 4. The intermediate receivers in the network only forward the packets and do not provide any security measure (such as integrity and authenticity checks). These receivers may be malicious to drop or modify S's packets or even inject fake packets.

We consider a class of applications where

1) each generated message is unknown to S until it is ready to send;

model.pdf

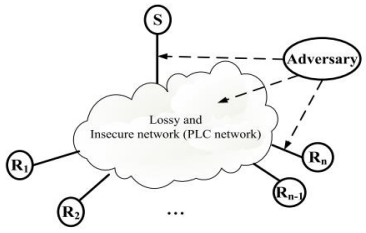

Fig. 4. Network model.

Figure 2: Network Model

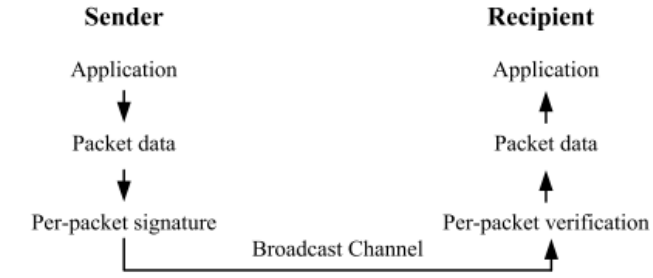

Fig. 5. Data flow of broadcast authentication.

Figure 3: str

2) S (resp. R) signs (resp. verifies) the message once it appears;

3) the sending rate at S is dynamic. The data flow of the broadcast authentication protocol is shown in Fig. 5

3.2 Requirement

In addition to the asymmetric mechanism that is needed for broadcast authentication, an efficient and secure broadcast authentication scheme for smart grids should still satisfy the following requirements:

1 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.

2 Robust to packet loss: The smart grid communication environment is not reliable; therefore, the scheme should be able to cope with the loss of packets during trans- mission.

3 Short authentication latency: Many PLC applications are real time applications, e.g. sending the control informa- tion to the customers. 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.

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

5 Receiver compromise tolerance: The protocol should be resilient to receiver compromise attack no matter how many receivers have been compromised, as long as the subset of non-compromised receivers can still form a connected graph with the trusted source.

6 Low communication overhead: Because a PLC network often is restricted in bandwidth, the number of bytes per packet used for authentication should be small.

7 DoS attacks resistance: The functions of the PLC network should not be disrupted by DoS attacks.

8 Freshness: A receiver should be able to differentiate whether an incoming message is the newest version.

9 Scalability: The protocol should be efficient even for large-scale smart grids with thousands of receivers.

10 Low storage requirement: Since the storage space of receivers is limited, some data for authentication like key material and signatures stored in memory cannot be too large.

Ideally, we would like a scheme that is able to recover from any loss of packets, has no authentication latency, can individually authenticate packets and ensure data confidentiality, has negligible overhead, and has a low computation cost. In practice, such a perfect scheme is difficult to achieve, and a compromise needs to be found between these requirements.

3.3 Assumption

Our protocol makes the following assumption.

- The sender cannot be compromised, and is trusted. In Drip, the sender is the origin of all legitimate message updates. The sender has unlimited computational power compared with receiver.
- The receiver can perform a limited number of asymmetric cryptographic operations such as signature verification in TinyECC [15], but they cannot afford to perform many such operations due to their energy limitations.

4. THE PROPOSED PROTOCOL

Before giving the detailed description of the proposed protocol, we first provide an overview of our protocol.

4.1 Overview of Our Protocol

Elliptic curve cryptography is an outstanding approach to public-key cryptography in terms of strength. Traditionally, the cost of public key operations may dominate the cost of transmitting packets in sensor network. Since broadcast authentication requires sensors to conduct a large number of PKC operations, using PKC in the traditional way is still impractical. It is desirable if we can significantly reduce the cost of PKC operations by optimizing the broadcast authentication protocols. We propose ShortPK approach, i.e. short-length public key, for example, if we reduce an ECC public key from 160bits to 80 bits, the computation cost for signature is reduced to roughly one eighth and the length of signatures is reduced to half. While short-length public key is easy to break, we need a more efficient way to distribute short-length public keys. With enough memory, we can load public keys into sensor′s memories before sensor deployment. But the public keys must be encrypted, otherwise if one sensor is compromised, the public keys will be compromised. We can use AES for the symmetric-key encryption, therefore nobody, including sensor nodes, knows public keys before the arrival of decryption key included in broadcast packet.

Meanwhile, we denote encryption key as Ki. HoweverïijŇit is important that sensor nodes are able to verify Ki in each of the broadcast packet. If the adversary randomly chooses a key Ki′ and a public key PKi , encrypts PKi into (PKi′)Ki′ using Ki′, both PKi′ and Ki′ are invalid. When receiving packet, sensors do not verify Ki′, then will get a gibberish PKi′. They will use PK′ as the public key to conduct signature verification. Even though the verification is not successful, sensors spend a lot of energy verifying the signature.

Therefore, sensors need to verify whether the received Ki is from the basestation or not. This can be achieved by one-way hash chain.

As illustrated in Fig. 6, the sender randomly chooses a number Kb, and then generates a one-way hash chain. A hash chain is based upon a public function H that is easy to compute, but computationally difficult to invert - thus, the term, one-way hash chain. A hash chain of length b is generated by repeatedly applying the hash function H to the last element, say, Kb, in the chain, to generate a sequence of hashes such that Kj = H(Kj+1). The head of the chain, K0, serves as a commitment to the entire hash chain. The hash chain is generated in the order Kb,KbâĹŠ1,....,K1,K0 and revealed in the reverse order. Here b is the maximum number of message disseminations permitted in the lifetime of the WSN. The keys K0,K1,. ..,Kb are referred to as puzzle keys, and Kj is used for the j-th disseminated messages, where j> 0.

But adversaries can intercept the communication channel between base station and sensor nodes. The encryption key broadcasted in open channel is exposed to adversaries. By compromising sensor nodes, the cipher text of PKi is also obtained by adversary. Based on mentioned information, adversary is able to get public key by decrypting cipher text stored on sensor nodes using intercepted key. After getting Public key, adversary is able to forge a packet containing malicious dataâĂŹ, that is (data′),Ki ,signature.

Signature is produced using decrypted public key.

Obviously, it take adversary time to listen to communication in channel and forge a malicious packet. We must prolong the time for adversary to produce an invalid packet. Here we can use message specific puzzle, make the packet hard to produce. And without solving a message specific puzzle, adversary cannot send a malicious packet.

4.2 System Initialization Phase

In this stage, the sender sets up an ECC by deriving a private key SK and public parameters PK,Q, p, q,H(.) by performing the following operations. It selects an elliptic curve E over GF(p), where p is a big prime number. Here Q denotes the base point of E while q is also a big prime number and represents the order of Q. It then picks the private key SKâĹĹGF(q) and generates the public key PK = SKÂůQ. As an illustrative example, for 160-bit ECC, both PK and Q are 320 bits long, and both p and q are 160 bits long.

Using a node of one-way hash chain as security key, strong encryption is achieved through the use of a 128-bit AES algorithm. The cipher text is loaded on every sensor nodes, thus adversary is not able to squeeze any key information out of cipher text. On the basis of AES algorithm, we can use small-sized public key and small-sized signature to keep our protocol cost-effective.

The secret key is included in broadcast packet.

If the public keys are not encrypted, adversaries can immediately get all the public keys, and will have much longer enough time to find the corresponding private keys.

In this stage, sender transforms public keys into ciphertext, (PKi)Ki. The encryption keys, denoted as K1„ „KN, are the elements of hash chain in Fig.1. Ki is the i-th key in the one-way key chain.

The committed value of this hash chain (i.e., K0) and ciphertext of public keys((PKi)Ki) are preloaded in each receiver of the network before deployment, together with the public parameters PK, Q, p, q,H(.) of the sender. Any of the reported secure key pre-distribution schemes (e.g., [16], [17]) may be used for this purpose. The hash chain is called the version chain as its elements correspond to update versions of the message. The j-th element of this hash chain stored in the sensor receiver is called j-th version key.

4.3 Pakcet Pre-processing Phase

Our solution is keyed message-specific puzzles based on one-way key chains (or briefly, message-specific puzzles). Intuitively, to prevent an attacker from precomputing puzzle solutions to forged messages, we add in such a puzzle a previously undisclosed key in the one-way key chain. As a result, an attacker cannot precompute a puzzle solution until such a key is released by the sender. Upon receiving such a packet, any node can easily verify the puzzle solution. However, we develop the puzzle system in such a way that it will take a substantial amount of time to solve such a puzzle. As a result, even if the key Ki is released in a broadcast packet, an attacker cannot immediately solve the puzzle for a forged packet, and thus cannot immediately launch Dos attacks.

Now let us describe the details of message specific puzzles. As in the strawnman approach, we assume the sender has generated a one-way key chain consisting of K0, K1„ „Kn, and distributed K0 to all potential receivers.

puzzle.pdf

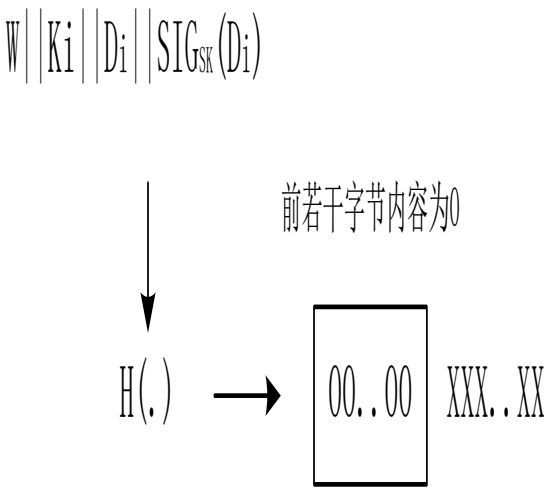

Figure 4: specific puzzle

Given the i-th message Mi, the sender first generates the broadcast authenticator, i.e., signaturei, and Ki then construct the puzzle, which we call the i-th message specific puzzle. For the sake of presentation, we dente the solution to this puzzle as Pi. As shown int Figure.2

A valid solution Pi to the i-th message-specific puzzle, where 1<= i <=n, must satisfy the following condition: After applying the hash function Fp to the i-th message-specific puzzle and its solution, we get an image where the first l bits are all âĂIJ0âĂİ. That is, F(i | Mi | signature | Ki | P) = 0000 xxxxxx where âĂIJxxx..âĂİ represents any bit pattern. The parameter l is called the strength of the puzzle. Because of the one-way property of the hash function Fq, one has to search through the space of possible solutions to solve the puzzle. In other words, given Mi signature Ki for each candidate solution W, the sender (or an attacker) has to verify if the first l bits of Fp(Mi, signature, Ki p) are all âĂIJ0âĂİ. Finally, the sender then broadcasts the packet with the payload i | Mi | signature | Ki | P .

4.4 packet verification phase

upon receiving a broadcast packet, each receiver first verifies the puzzle solution, If the puzzle solution is invalid, the receiver will simply drop this packet.

Then receiver verifies the puzzle key K1 included in the packet, i.e., whether the hash result of K1 equals to K0 or not.

If the puzzle key k1 is authentic, we can use it to decrypt the ciphertext stored in receiver nodes. After that, signature included in packet will be verified by decrypted public keys.

Then, receiver accept the data included in packet.

5. CONCLUSIONS

6. ACKNOWLEDGMENTS

APPENDIX