International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459 (Online), An ISO 9001:2008 Certified Journal, Volume 3, Special Issue 1, January 2013)

International Conference on Information Systems and Computing (ICISC-2013), INDIA.

Reliable And Efficient Distributed Code Dissemination In Wireless Sensor Networks

P.Suganya[1], G.Murugaboopathi[2]

[1]*P.G Student, Network Engineering, Veltech MultiTech Dr. Rangarajan Dr. Sakunthala Engineering College, Avadi, Chennai, Tamil Nadu.*

[2]*Assistant Professor, Veltech MultiTech Dr.Rangarajan Dr.Sakunthala Engineering College, Avadi, Chennai, Tamil Nadu.*

Email: mpsugi@gmail.com

Abstract

In Wireless Sensor Networks ,Code Dissemination is a process of Propagating new code images or commands in the network. Since the WSN are mostly deployed in the military and hostile environments secure code dissemination is needed. Mostly code dissemination protocols are based on two approaches. First based on the centralized approach in which only the base station has the authority to initiate code dissemination. Second based on the distributed manner which allows multiple authorized network users to simultaneously and directly update code images on different nodes without involving the base station. Motivated by this consideration, we develop a secure and distributed code dissemination protocol named DiCode. A salient feature of DiCode is its ability to resist denial-of-service attacks which have severe consequences on network availability. DiCode in a network of resource limited sensor nodes, which shows the efficiency of our protocol signature verification on the propagation delay of code dissemination in multihop networks, performance can be improved by using more powerful sensor node. To verify the efficiency of the proposed approach in practice, we also implement the proposed mechanism in a network of resource-constrained sensor nodes.

***Keywords*-- Denial Of Service, Code dissemination, Sensor nodes, Multihop networks.**

I. INTRODUCTION

A wireless sensor network is expected to consist of a potentially large number of low-cost, low-power, and multifunctional sensor nodes that communicate over short distances through wireless links. Due to the potential to provide fine-grained sensing and actuation at a reasonable cost, wireless sensor networks are considered ideal candidates for a wide range of applications, such as industry monitoring and military operations.Sometimes it is desirable necessary to reprogram sensor nodes. The wireless links after they are deployed, due to, for example, the need of removing bugs and adding new functionalities. The process of propagating a new code image to the nodes in a network is referred to as code dissemination. A few code dissemination protocols have been developed recently to propagate new code images using the ad-hoc wireless network formed by the sensor nodes. In particular, Deluge [3] an epidemic protocol [8] for efficient advertisement of meta data and spatial multiplexing for efficient propagation of code images.

Deluge is generally accepted as the state of the art for code dissemination in wireless sensor networks and has been included in recent TinyOS distributions [4]. In hostile environments where theremay bemalicious attacks against wireless sensor networks, code dissemination faces threats from both external attackers and potentially compromised nodes.

For example, the adversary may attempt to modify or replace the real code image being propagated to sensor nodes, introducing malicious code into the sensor network. As another example, the adversary may inject bogus code dissemination packets and force normal sensor nodes to verify and/or forward them, thus exhausting their limited battery power.

However, all these code dissemination protocols ([2]–[4], are based on the centralized approach which assumes the existence of a base station and only the base station has the authority to reprogram sensor nodes. As shown in Fig. 1(a), when the base station wants to disseminate a new code image, it broadcasts the signed code image and each sensor node only accepts code images signed by it.

Unfortunately, there are WSNs having no base station at all. For example networks of a military WSN in a battlefield to monitor enemy activity, a WSN deployed along an international border to monitor weapons smuggling or human trafficking, and a WSN situated in a remote area of a national park monitoring illegal activities (e.g., firearm discharge, illicit crop cultivation). Having a base station in these WSNs introduces a single point of failure and a very attractive attack target. Obviously, the centralized approach is not applicable to such WSNs.

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459 (Online), An ISO 9001:2008 Certified Journal, Volume 3, Special Issue 1, January 2013)
International Conference on Information Systems and Computing (ICISC-2013), INDIA.

Also, the centralized approach is inefficient, weakly scalable and vulnerable to some potential attacks along the long communication path.

Assigning a different privileges of reprogramming sensor nodes is especially important in large scale WSNs owned by an owner and used by different users from both public and private sectors. Distributed service protocol in WSNs (e.g., decentralized sensing) as shown in Fig. 1(b) is a research field that is getting increasingly more attention. In this paper, we further extend this scheme in three important aspects. Firstly, we consider denial-of-service (DoS) attacks on code dissemination, which have severe consequences on network availability, as well as propose and implement two approaches to defeat DoS attacks. Secondly, the proposed code dissemination protocol is based on a secure and efficient proxy signature by warrant (PSW) technique using cipher Puzzle constructor which makes it stronger. Thirdly, we consider how to avoid reprogramming conflict and support dynamic participation.

A secure distributed code dissemination protocol should satisfy the following requirements similar to the centralized approaches:

(1) *Freshness:* An earlier version of a program image can- not be installed over the program with the same or greater version number, ensuring a node always installs the newest version of a program image.

(2) *Integrity of Code Images:* The source of a program image must be verified by a sensor node prior to installation, ensuring that only a trusted source can install a program. In addition, it must be possible to ensure that a program has not been altered during its transmission.

(3) *Node Compromise Tolerance:* A compromised node must be prevented from causing an uncompromised node to violate the above security requirements.

(4) *DoS Attacks Resistance:* A practical code dissemination mechanism should maintain service availability even in the presence of DoS attacks Other than meeting the above requirements, a distributed code dissemination protocol should also have the follow- ing properties.

(5) *Supporting Different User Privileges:* To ensure smooth functioning for a WSN, the level of each user privilege should be limited by the network owner area during his/her subscription period.

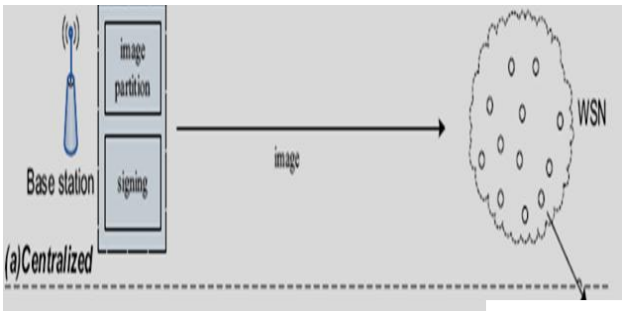

Fig 1(a) Centralized Approach

(6) *Partial Reprogram Capability:* To prevent sensor nodes from being totally controlled by network users, the special modules (e.g., authentication module for each new program image) on each node cannot be overwritten by anyone except the network owner.

(7) *Avoiding Reprogramming Conflicts:* Multiple users may be allowed to reprogram a node, which may result in reprogramming conflicts. Some mechanisms should be provided to avoid such conflicts.

(8) *User Traceability:* In most application scenarios, traceability is highly desir- able, particularly for code dissemination, where it is used for collecting the users' activities for some purposes.

(9) *Scalability:* Firstly, the protocol needs to be efficient even in a large-scale WSN with thousands of sensor nodes, and secondly, the protocol should be able to support a large number of users.

(10) *Dynamic Participation:* New nodes can be supplemented whenever they are needed, and the scheme should allow dynamic addition of new users.

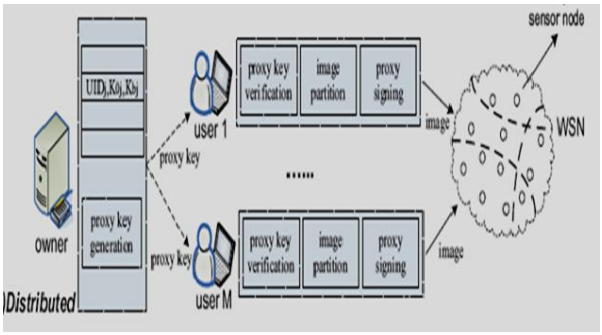

Fig1(b) Distributed Approach

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459 (Online), An ISO 9001:2008 Certified Journal, Volume 3, Special Issue 1, January 2013)
International Conference on Information Systems and Computing (ICISC-2013), INDIA.

II. Related Work

Several Recent works have attempted to provide secure code dissemination for wireless sensor networksThe design, development, and evaluation of an efficient, secure, robust, and DoS-resistant code dissemination system named *Seluge* for wireless sensor networks. Seluge is an extension to Deluge [6], an open source code dissemination system included in TinyOS [4]. It inherits the efficiency and robustness properties from Deluge, and at the same time provides security protections for code dissemination, including the integrity protection of code images and resistance to the following three classes of DoS attacks: (1) DoS attacks against signature packets; (2) DoS attacks against code dissemination packets; and (3) DoS attacks against maintenance packets. To the best of our knowledge, these are all the DoS attacks that manipulate code dissemination protocols.

The key contribution of Seluge is a novel way to organize the packets used to distribute new code images. By carefully arranging code dissemination data items and their hash images in packets, Seluge provides *immediate authentication* of each packet upon receipt, without disrupting the efficient propagation mechanisms used by Deluge. Thus, it can defeat the DoS attacks exploiting authentication delays.[6].

Deluge [6] is an open source code dissemination system for wireless sensor networks running TinyOS [4]. Deluge uses a page-by-page dissemination strategy. A code image is divided into fixed-size pages, and each page is further split into same-size packets. The pages for a code image is delivered in a sequential order. After completely receiving a page, a node advertises the availability of the newly received page, and may transmit the corresponding packets upon request. On the other hand, a receiver requests a page only after having received all packets in previous pages. It uses an epidemic protocol for efficient advertisement of code meta data [5]. Each node periodically advertises the version of its code image and the number of pages it has for that version. For energy efficiency, the advertisement rate is dynamically adjusted: If a node discovers its own advertisement is different from those received from others, it increases its advertisement rate. Otherwise, it decreases the rate. As a result, Deluge can achieve rapid propagation during dissemination of a new code image, but consumes little resource in steady state.

III. Proposed Approach

Before giving the detailed description of DiCode, we first discuss what kind of cryptographic techniques are suitable for distributed code dissemination.

A. Cryptographic Techniques for Distributed Code Dissemination

Fig. 1(b) shows that a distributed code dissemination protocol consists of three kinds of participants, the network owner, authorized network users and all sensor nodes. After the users register to the owner, they can enter to the WSN and then have pre-defined privileges to simultaneously and directly reprogram the sensor nodes without involving the owner. A naive solution is to pre-quip each node with multiple public key/reprogramming privilege pairs, each of which corresponds to one authorized user. This scheme allows a user to sign a program image with his/her private key such that each node can verify if the program image originates from an authorized user. However, resource constraints on sensor nodes often make it undesirable to implement such an expensive algorithm. For example, in RSA-1024 public key cryptosystem, the length of each public key is more than 1026 bits.

Assuming that the length of reprogramming privilege is 48 bytes, the length of each public key/reprogramming privilege pair is more than 176 bytes. This means that not too many public key/reprogramming privilege pairs can be stored in a node. In this case, not too many users can be supported. Moreover, it is clear that the network owner has no ability to pre-define the reprogramming privileges of new users who will join after the WSN deployment. Once a new user registers to the network owner, the owner needs to sign a new public key/ reprogramming privilege pair and then broadcasts it to all nodes. A more suitable approach is for each authorized user to send a new program image to the nodes through a standard group signature technique. A group signature scheme allows one member of the group to sign a message such that any verifier can verify that the message originated from a group member. Thus, only the group public key is pre-loaded onto each sensor node. Unfortunately, a group signature algorithm does not allow the network owner to specify different reprogramming privileges for different users.

B. DiCode Overview

In this paper, PSW is introduced into the design of DiCode. This technique involves two kinds of participants, an original signer and proxy signers. The original signer gives the proxy signer a warrant, which specifies the identity of the proxy signer, the identity of the original signer, the range of messages to sign, the expiration time of the delegation of signing power, etc. The proxy signer generates proxy signatures only with the proxy signature key given by the original signer.

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459 (Online), An ISO 9001:2008 Certified Journal, Volume 3, Special Issue 1, January 2013)
International Conference on Information Systems and Computing (ICISC-2013), INDIA.

Verifiers validate proxy signatures only with the public key of the original signer and pay attention to the legality of the warrant. The detailed information about applying the PSW technique into DiCode is as follows[8].

The network owner plays the role of original signer while the network users play the role of Proxy signers . Through registration. The users obtain one or more proxy keys from the network owner before they enter into the Wireless network.

version num (1)	targeted node identities set	code image size (2)	root of the Merkle hash tree

Fig. 2. An example of the format of the message *m* of DiCode. The byte size of each field is indicated below the label.

IV. Some Techniques For Resisting Dos- Attacks

Most secure centralized schemes of DiCode uses a digital signature technique for the authentication of the program image. This signature is vulnerable to DoS attacks. Since the adversary may flood a large number of illegal signature messages to the nodes to exhaust their resources and render them less capable of serving legitimate users. In order to prevent the attacks, message specific puzzle and the improved message specific puzzle are described here, which can complement the basic protocol of DiCode. Assuming the entire network is partitioned into a set of regions. Each sensor node belongs to exactly one region and a region may contain multiple nodes. The size of a region is application specific and sufficiently small to support the distributed reprogramming resolution. For reducing communication costs we choose to elect region (cluster) heads. In that each region has a head sensor node which is responsible for generating a puzzle and then distributing it to all other nodes of the region.

A. Message Specific Puzzle Approach

To prevent the attacks, we adopt message specific puzzle (also called client puzzles) of into DiCode [8]. The idea is summarized as follows. In the system initialization phase,the network owner sets a threshold for user reprogramming rate. When there is no evidence of such an attack, each sensor node processes the signature messages normally, that is, indiscriminately. On the other hand, once a node finds the rate of incoming signature message is more than the threshold, it believes that it is under a DoS attack and only performs verification on signature messages selectively. In particular, the node attaches a unique puzzle into the beacon messages which is periodically broadcasted to declare service existence, and requires the solution of the puzzle to be attached in each signature message. The node commits resources to process a signature message only when the solution is correct. In general, solving a puzzle requires a brute-force search in the solution space, while solution verification is very fast.

Additionally, puzzles are deployed in conjunction with conventional time-outs on node resources. Thus, in order to create an interruption in service, an adversary must have abundant resources to be able to promptly compute a large enough number of puzzle solutions in line with his sending rate of illegal signature messages. In contrast, although puzzles slightly increase legitimate users' computational load, they are still able to obtain reprogramming services regardless the existence of the attack. To incorporate this method into our protocol, we add a MESSAGE SPECIFIC PUZZLE flag in the beacon messages. If a node, say Sv, is not under attack, it sets the MESSAGE SPECIFIC PUZZLE flag to "No". It indicates to the users that no puzzles are being distributed. And our basic protocol is executed normally. If Sv is under attack, it sets the MESSAGE SPECIFIC PUZZLE flag to "Yes", and adds a puzzle (i.e., a timestamp Tv, a random number a and an integer l) into the beacon messages.

A valid solution Li is such a value that after applying the hash function h() to (m_mw_y_k_Tv_a_Li), the first l bits of the resulting image are all "0", as illustrated in Fig. 2. The parameter l determines the strength of the puzzle.

Before transmitting the signature message {m,mw, y, k}, a user first tries to solve the puzzle by finding the puzzle solution Li. Subsequently, the user sends the final signature message {m,mw, y, k, Tv, a, Li} to Sv. Obviously, the puzzle solution in every signature message can be efficiently verified by Sv via a hash function operation and comparison. Only if this verification is successful, Sv performs expensive verification on the signature message {m,mw, y, k}.

B. An Improved Message Specific Puzzle Approach

To provide a more effective message specific puzzle approach, we can make use of the one-way hash chain [8].

However, this requires more changes to the basic protocol of DiCode. For brevity, we just present the parts that need to be changed.

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459 (Online), An ISO 9001:2008 Certified Journal, Volume 3, Special Issue 1, January 2013)
International Conference on Information Systems and Computing (ICISC-2013), INDIA.

1) System Initialization: In addition to the activity described involved in the system initialization phase of the basic protocol, the network owner generates M oneway key chains. A one way key chain is based on a public cryptographic hash function h(), which is easy to compute but computationally hard to invert. A key chain with length b is generated by applying h() on the initial selected element Kbj repeatedly b times. The last output of applying h() b times, is called the committed value of the key chain. In DiCode, the length (i.e., b) of every key chain must be exactly the number of times the respective user Uj is allowed to reprogram the sensor nodes. For user Uj, the ith element in the hash chain is denoted as Kij. Then we have Kij = h(K(i+1)j), 0≤i≤b. Here a key chain is not only used to limit the number of reprogramming times, but also generate a message specific puzzle to mitigate DoS attacks against signature messages.

2) User Pre-processing: The user pre-processing phase of the basic protocol, user Uj does b − i hash operations on the value Kbj , and obtains Kij , and then uses it to generate a puzzle., here we consider the signature message of version i code image, denoted a {m,mw, y, k}. The signature message {m,mw, y, k} and the puzzle key Kij constitute a message specific puzzle. A valid solution Li is such a value that after applying the hash function h() to (m_mw_y_k_Kij_Li), the first l bits of the resulting image are all "0", as illustrated in Fig. 3. The parameter l determines the strength of the puzzle. Before transmitting the signature message, user Uj first tries to solve the puzzle by finding the puzzle solution Li. Subsequently, user Uj sends the finial signature message {m,mw, y, k,Kij, Li} to the nodes.

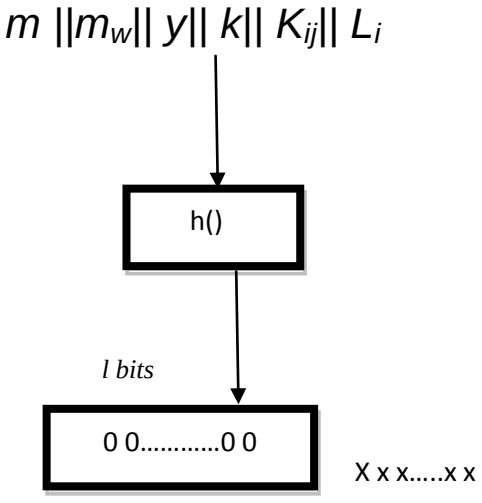

Fig 3. An improved Message Specific Puzzle

3) Sensor Verification: The sensor verification phase of the basic protocol, if the warrant mw and the message m are valid, the sensor node obtains the identity UIDj of user Uj from the warrant mw. According to UIDj , the node can extract the corresponding puzzle key K(i−1)j from its memory. Thus the node can verify whether the received puzzle key Kij is valid by comparing whether h(Kij) equals K(i−1)j . At the same time, the node needs to confirm that Kij has not been used along with a valid signature message before. Only if these two verifications are successful, the node replaces K(i−1)j with Kij . If the puzzle solution is valid, the node goes to the next step. Therefore, without first solving some message specific puzzles with a fresh puzzle key, the adversary cannot force a node to verify signatures in forged messages. Also, in the above procedure, since the node replaces K(i−1)j with Kij for each successful verification and does just one hash operation for comparison, an adversary cannot claim a puzzle key close to the end of key chain and fool the node to perform a large number of unnecessary hash operations, causing a DoS attack. Compared to the message puzzle approach presented in previous existing protocols , this improved approach can more effectively legitimate user may be incomparable in computation power. This is because that although it takes the same effort for both an authorized user and adversary to solve a puzzle, the authorized user has a clear advantage over the adversary due to the prior knowledge of the puzzle keys. The authorized user has enough time to solve a puzzle off-line before disseminating a new code image. In contrast, the adversary has a very tight time limit of solving the puzzle; it cannot begin to solve the puzzle until it has intercepted the puzzle key when the user transmits the signature massage, but has to finish solving the puzzle before the puzzle key becomes invalid when the signature message reaches the targeted sensor nodes.So this approach able to support more number of users.

V. ANALYSIS

The operations of Dicode has been discussed. By the protocol, we can achieve secure and distributed code dissemination. However, some important issues need further discussion.

A. Avoiding Reprogramming Conflicts

When multiple users are allowed to reprogram a node, some approaches should be employed as follows. In the user pre- processing phase, each network user, say Uj , needs to add the chosen TP to every updated program image, where TP denotes the time duration that a node will be occupied for reprogramming.

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459 (Online), An ISO 9001:2008 Certified Journal, Volume 3, Special Issue 1, January 2013)
International Conference on Information Systems and Computing (ICISC-2013), INDIA.

Our protocol allows a user to execute the reprogramming of nodes individually, without waiting for all nodes to be available.

B. Dynamic Participation

1) *New User Joining Phase:* This phase is invoked when a user, U_{j+1}, hopes to obtain code dissemination privilege after the network is deployed.

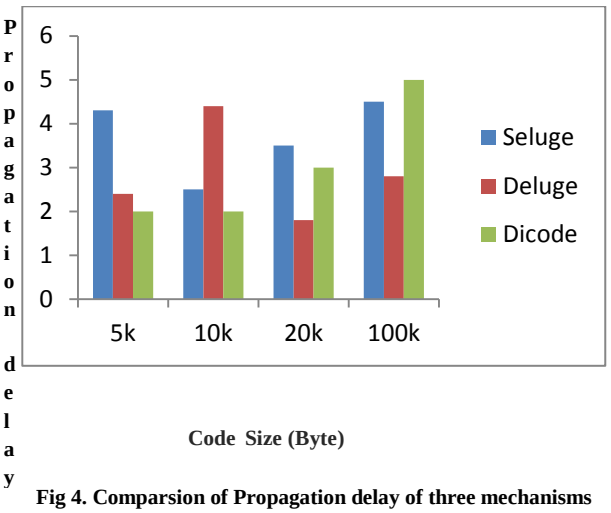

Fig 4. Comparsion of Propagation delay of three mechanisms

VI. CONCLUSION

In this project, a distributed and DoS-resistant code dissemination protocol named DiCode has been proposed. Besides analyzing the security of DiCode, this project has also reported the evaluation results of DiCode in a network of resource limited sensor nodes, which shows the efficiency of our protocol signature verification on the propagation delay of code dissemination in multihop networks, and in multi-hop, performance can be improved by using more powerful sensor node (e.g., Imote2 has 416 MHz processor). Upon receiving the signature packet, each sensor node can start to check the validity of the signature packet. At the same time, it transmits the packet to next-hop node. Considering the security benefits that DiCode provides, the cost is acceptable.

REFERENCES

[1] D. He, J. Bu, S. Chan, C. Chen, and M. Yin, "Privacy-preserving universal authentication protocol for wireless communications," IEEE Trans. Wireless Communication., vol. 10, no. 2, pp. 431–436, 2011.

[2] H. Tan, D. Ostry, J. Zic, and S. Jha, "A confidential and DoS resistant multi-hop code dissemination protocol for wireless sensor networks," in WiSec'09, Zurich, Switzerland, Mar.2009, pp. 245–252

[3] Satya Venkatesh Kadali, O.Srinivasa Rao, Dr MHM Krishna Prasad, "A Routing-Driven Public-Key Cryptosystem Based Key Management Scheme for a Sensor Network"

[4] Qijun Gu, "Efficient Code Diversification for Network Reprogramming in Sensor Networks

[5] Kun Sun An Liu, Roger Xu Peng Ning, Douglas Maughan, "Securing Network Access in Wireless Sensor Networks"

[6] A. Juels and J. Brainard, "Client puzzles: a cryptographic countermeasure against connection depletion attacks," in Proc. NDSS'99.

[7] R. Fonseca, P. Dutta, P. Levis, and I. Stoica, "Quanto: tracking energy in networked embedded systems," in Proc. USENIX OSDI'08, pp. 323– 328.

[8] Daojing and Sammy, "DiCode: DoS Resistant and Distributed Code Dissemination in Wireless Sensor Networks" VOL. 11, NO. 5, 2012.