I. J. Communications, Network and System Sciences, 2009, 1, 1-89
Published Online February 2009 in SciRes (http://www.SciRP.org/journal/ijcns/).

Five Basic Types of Insider DoS Attacks of Code Dissemination in Wireless Sensor Networks

Yu ZHANG[1, 3], Xing She ZHOU[1], Yi Ming JI[2], Yee Wei LAW[3], Marimuthu PALANISWAMI[3]
[1]*School of Computer Science, Northwestern Polytechnical University, Xi'an, Shaanxi, China*
[2]*College of Engineering and Computer Science, Australian National University, Canberra, ACT, Australia*
[3]*Department of Electrical and Electronic Engineering, the University of Melbourne, Parkville, VIC, Australia*
Email: [1]{*zhangyu, zhouxs*}*@nwpu.edu.cn,* [2]*yiming@rsise.anu.edu.au,* [3]{*y.law, swami*}*@ee.unimelb.edu.au*
Received October 27, 2008; revised December 8, 2008; accepted December 28, 2008

Abstract

Code dissemination is one of the important services of wireless sensor networks (WSNs). Securing the process of code dissemination is essential in some certain WSNs applications, state-of-the-art secure code dissemination protocols for WSNs aim for the efficient source authentication and integrity verification of code image, however, due to the resource constrains of WSNs and the epidemic behavior of the code dissemination system, existing secure code dissemination protocols are vulnerable to Denial of Service (DoS) attacks when sensor nodes can be compromised (insider DoS attacks). In this paper, we identify five different basic types of DoS attacks exploiting the epidemic propagation strategies used by Deluge. They are (1) Higher-version Advertisement attack, (2) False Request attack, (3) Larger-numbered Page attack, (4) Lower-version Adv attack, and (5) Same-version Adv attack. Simulation shows these susceptibilities caused by above insider DoS attacks. Some simple models are also proposed which promote understanding the problem of insider DoS attacks and attempt to quantify the severity of these attacks in the course of code dissemination in WSNs.

Keywords: Sensor Networks, Code Dissemination, Deluge, Security, DoS Attacks

1. Introduction

Wireless sensor networks (WSNs) now can provide many services with a large number of resource-constrained nodes. One important service is *code dissemination* which can disseminate new code images into all sensor nodes that need them over the wireless link. In order to guarantee flexibility, efficiency and reliability of code propagation, a number of code dissemination protocols (MOAP [1], Deluge [2], MNP [3] and Infuse [4], Sprinkler [5], Aqueduct [6], and Freshet [7], etc.) have been developed. However, none of them consider the communication security of WSNs.

Recently, some research works (Sluice [8], SecureDeluge [9] and Deng-tree [10]) have attempted to provide efficient authentication of code dissemination. These approaches, unfortunately, are vulnerable to Denial of Service (DoS) attacks because they do not take the authentication of the control packets (in Deluge, they are named as Advertisement (Adv) and Request (Req)) into consideration.

The contribution of this paper is that we identify five different basic kinds of DoS attacks made by malicious nodes exploiting control packets. First is Higher-version Adv attack, second one is False Req attack, the third is Larger-numbered Page attack, the fourth is Lower-version Adv attack and finally is Same-version Adv attack. We also present the degree of damage made by each attack through quantitative analysis.

The paper is organized as follows. Section 2 reviews the related work on security and DoS attacks in WSNs. Section 3 gives an overview of Deluge and describes some vulnerability of the epidemic propagation strategies. Section 4 introduces five basic types of insider DoS attacks against Deluge and proposes the system models. Section 5 evaluates the performance of Deluge under different forms of the DoS attacks and discusses the simulation results. Section 6 concludes the paper.

2. Related Work

A variety of protocols have been proposed to support code dissemination in wireless sensor networks. MOAP [1] developed by Stathopoulos *et al*. extended XNP [11] and employed a publish/subscribe scheme to propagate

software update over a multi-hop network. Deluge [2], which is distributed with TinyOS [12], shared many ideas with MOAP, including the use of unicast NACKs and broadcast of the code. It spreads the code using spatial multiplexing. MNP [3], which was implemented in the Michigan State University, introduced a sender-selection algorithm which limits the total number of senders in one neighborhood to mitigate the hidden terminal effect.

Recent researches have developed some protocols to provide secure reprogramming services by extending Deluge with authentication and integrity mechanisms. Sluice [8], SecureDeluge [9] and Deng-tree [10] leveraged the similar solutions which based on digital signature and cryptographic hash function to guarantee the security of code images. they are distinguished through structure, granularity and strength of hashing [13].

Furthermore, some other protocols which are focused on security of communication in wireless sensor networks have been proposed. They can be classified into two types: *asymmetric* and *symmetric* mechanism. μTELSA [14] is the representative of the former one. It provided broadcast authentication via symmetric primitives only, and introduced asymmetry with delayed key disclosure and one-way function key chains. The typical protocols of symmetric mechanism are q-composite key pre-distribution and random pairwise key schemes [15] proposed by Chan *et al*. They used pairwise keys to establish a secure communication infrastructure of wireless sensor networks and attempt to mitigate the threat of compromised nodes. Moreover, routing security is another important issue which is needed to pay attention to. Karlof and Wagner analyzed the security of all the major sensor network routing protocols and described crippling attacks against all of them and presented countermeasures [16].

However, though many secure protocols have been proposed, few of them could mitigate DoS attacks in sensor networks. Strictly speaking, although we usually use the term to refer to an adversary's attempt to disrupt, subvert, or destroy a network, a denial of service attack is any event that diminishes or eliminates a network's capacity to perform its expected function [17,22]. One typical DoS attack is that a captured node broadcast malicious messages to other nodes, resulting in a large amount of extra transmission, storage or computation overhead. Because wireless sensor networks are more resource-constrained compared with traditional networks, they are much easier to be destroyed by DoS attacks.

3. Problem of the Epidemic Propagation of Deluge

In this section we will first give a detail description of Deluge and then we point out some security vulnerabilities of the epidemic propagation strategies in Deluge.

Supported by the National Natural Science Foundation of China under Grant No.60573161, the Australian Research Council Research Network on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), and the DEST International Science and Linkage Grant.

3.1. Overview of Deluge

Deluge is an epidemic protocol used for code dissemination which can guarantee large data objects to be disseminated quickly and reliably over a multi-hop wireless sensor network. It employs advertise-request-code handshaking protocol [18] to set up a reliable bi-directional link before transferring data and reduce the transmission of redundant data throughout the network. In Deluge, the binary image is divided into fixed-size pages, each page can be transferred within 48 same-size packets. This data representation supports software update based on page differences and makes use of spatial multiplexing to allow parallel transfers of code image. Deluge also borrows some ideas from Trickle [19] which uses suppression mechanism and dynamic adjustment of advertisement rate to achieve *density-aware capability* and energy efficiency.

Trickle uses a "polite gossip policy", where sensor nodes periodically broadcast a code summary to local neighbors but stay quiet if they have recently heard a summary identical to theirs. It divides time into a series of rounds and in each round nodes can decide whether or not to broadcast its own Adv. Deluge uses $\tau_{m,i}$ to denote the duration of round i. And $\tau_{m,i}$ is bounded by τ_l and τ_h. In each round, a node could broadcast its Adv at r_i which is picked up randomly in the range $[\tau_{m,i}/2, \tau_{m,i}]$. An Adv has a code summary φ which contains two integers $\{\upsilon, \gamma\}$, where υ is the version number and γ is the largest numbered page available for transfer. If there is a new code image injected into WSN by base station, a node S, which has received the image, broadcasts an Adv with a summary $\varphi\{\upsilon_s, \gamma_s\}$. When a node R hears this Adv, it first compares its own image version υ_R with υ_s. If $\upsilon_s > \upsilon_R$, node R will update its version to υ_s. Then if $\gamma_s > \gamma_R$, node R will send a Req to node S to request smallest numbered page it needed. S broadcasts the requested page after it received the Req. Through this process new images are propagated to all nodes page by page.

Density-aware capability of Deluge makes effect on Adv and Req propagation, where redundant advertisement and request messages are suppressed to minimize contention. In each round a node will not broadcast its own Adv unless the number of Adv with same version it heard is less than a predefined threshold k which is adjusted according to the density of the network. Moreover, if a node has heard Req packets for the page it needs before transmitting Req, it will stop requesting its own Req. Similarly, if a node hears request for the pages with smaller than that of the page it is currently transmitting, the node suppresses transmitting of the subsequent code packets.

If the network is consistent, Deluge will decreases its advertisement rate, i.e. it set $\tau_{m,i}$ to $2\tau_{m,i}$ in each round, but will be not larger than τ_h. Otherwise, it set $\tau_{m,i}$ to the minimal value τ_l. Deluge changes dynamically the rate of

advertisements to allow quick dissemination when needed and save resources when new code updates propagation is not needed.

3.2. Security Vulnerabilities of the Epidemic Propagation Strategies in Deluge

Epidemic propagation strategies allow rapid dissemination of information through purely local interactions in large scale, dynamic and not all the time coherent environments. In an epidemic protocol such as Deluge [2], a new code initiated from a source is rebroadcasted by neighboring nodes and extends outward, hop by hop, until the entire network is reached. The epidemic behavior provides high resilience to random process and network failures in the free attack scenario.

However, the power, communication, computation and storage capabilities of each sensor node are extremely limited and wireless sensor networks could be deployed in hostile and unattended environments for long periods of time. Each sensor node is insecure, which means it is trivially easy to retrieve program code, static data, and even dynamic program memory from nodes [20]. Moreover, the most of current code dissemination protocol like Deluge have not been designed with security in mind. Consequently, code dissemination may be possible to face threats from compromised nodes. If a node is captured, the attacker can gain control of that node and even gain complete control of an entire deployed network due to the epidemic nature of protocols like Deluge. For example, an adversary may use suppression and dynamic adjustments of the broadcast rate mechanisms of Deluge to prevent the propagation of code updates, waste network resources, introduce unnecessary latency or disrupt the normal operation of code dissemination.

Although many secure code dissemination protocols have been proposed recently, most of them aim for providing authentication and integrity of code updates in sensor networks to ensure malicious code are not disseminated or installed. However, none of them provide the effect schemes to prevent insider attackers from exploiting epidemic and suppression mechanisms of Deluge to launch DoS attacks. Although our future goal is to provide Insider-DoS-Resistant code dissemination scheme, we first should discover and analyze the susceptibilities of Deluge caused by insider DoS attacks.

4. Five Basic Tipes of Insider DoS Attacks and System Models

In this section we first describe five different bisic types of insider DoS attacks agaist efficient code dissemiantion mechanisms used by Deluge. Then we proposed simple models for these attacks made by malicious nodes using control packets.

4.1. Insider DoS Attacks on Deluge

Attack 1: Higher-version Adv

This is an insider attack which aims to increase energy comsuption and prevent normal nodes from receiving new code images. We assume the current version of image in it is υ, at this moment a malicious node broadcast an Adv with higher version υ' ($\upsilon' > \upsilon$), the network will be inconsistent and all the neighbor nodes will update their version to υ'. Meanwhile, they will adjust $\tau_{m,i}$ to the minimal value τ_l according to the Deluge's rules. Therefore the adversary could exploit the dynamic adjustments of the advertisement rate mechanisms to enforce legitimate nodes to transmit more frequently and eventually result in energy waste of network. Furthermore, if at the same time the base station has injected a new image with version υ'' lower than υ', the nodes, whose image version have updated to υ', could not get the new image with version υ''. In this case, the code dissemination will be failed.

Attack 2: False Request

This is another insider attack which targets to introduce unnecessary communication overhead or disrupt the normal code dissemination. According to the Deluge's rules, if a node S sends a Req to node R to request a specific page, R will broadcast a large number of code packets to S as response. Besides this case, if the page requested has a smaller page number than other neighbor nodes', Deluge will give this Req higher priority. Therefore the adversary could leverage these weaknesses of the rules to send bogus Req to trigger unnecessary transmission of code packets or send Req modified which included a smaller page number in order to suppress other Req from normal nodes.

Attack 3: Larger-numbered Page

This is an insider attack which is to target sensor node's energy consumption and looks similar to Attack1 mentioned above. We assume a node R has successfully received all packets in page0 to page γ'. According to the Deluge's rules, when this node R hears an Adv sent by node S which contains higher page γ ($\gamma > \gamma'$), R will send a Req packet to S to request new page γ'+1. This Larger-numbered Page of Adv from S could cause inconsistencies among neighboring nodes. And these nodes including R will adjust $\tau_{m,i}$ to the minimal value $\tau_{l..}$ Therefore a malicious node can take advantage of this principle to enforce nodes to broadcast Adv more frequently and eventually waste sensor nodes' energy through inducing them disseminating meaningless Req and producing a lot of Adv messages.

Attack 4: Lower-version Adv

This is another insider attack which is similar to Higher-version attack except that in this attack adversarial nodes could broadcast Adv with lower version than its neighboring nodes'. Therefore this attack induces a larger number of extra communications overhead

Table 1. Five kinds of DoS attacks exploiting the epidemic propagation strategies used by Deluge.

Attack Class	Resource Consumption	Prevention of code propagation or introduction of unnecessary latency
Adv-based	Attack 1, Attack 3, Attack 4	Attack1, Attack 5
Req-based	Attack 2	Attack 2

which lead to resource consumption because many normal nodes which have already the new object profile [2] would send their Adv and code packets to malicious node.

Attack 5: Same-version Adv
This is an attack which aims to introduce unnecessary latency. According to the Deluge's rules, a node will broadcast its Adv with summary φ only if less than a threshold k advertisements with summary $\varphi'\{\upsilon'=\upsilon, \gamma'=\gamma\}$ have been received. Therefore, a compromised node could exploit suppression mechanisms of Deluge to reduce the Adv transmission of legitimate nodes through sending multiple bogus advertisements with φ'. As a result, the code updates would not be disseminated efficiently and rapidly.

These five basic kinds of DoS attacks mentioned above can be divided into two classes in terms of attack methods.

Furthermore, the insider DoS attacks against Deluge can be more complex and effective through combining the different basic kinds of DoS attacks. For example, a malicious node can broadcast Higher-version Adv while send Req to neighbor nodes. This hybrid attack could cause much greater damage to the propagation of code updates in WSNs.

4.2. System Models

In this subsection, we set up some simple system models for four different kinds of attacks described above. Because the impact of the Same-version Adv attack is obvious and easy to understand, we shall not present this model here. Section 5 will give more performance results obtained by simulation.

Attack 1: Higher-version Adv
Firstly we analyze the propagation time of a bogus Adv from one comprised node to all sensor nodes in a WSN through epidemic mechanisms like Deluge. This duration $\boldsymbol{T}_{\text{bAdv,h}}$ for a node h hops away from the first malicious node of the bogus Adv is $\boldsymbol{T}_{\text{bAdv, h}} = h \cdot \boldsymbol{T}_{\text{Adv}}$ where $\boldsymbol{T}_{\text{Adv}}$ is the time used by the nodes in advertising their bogus Adv. To calculate $\boldsymbol{T}_{\text{Adv}}$, we need to find the expected number of transmission required for a successful transmission of a packet. Let $\boldsymbol{P}_{\text{one-hop}}$ be the probability of a successful transmission of a packet over a single hop. Assuming that the retransmission of a packet is independent, the probability that the number of transmissions of a packet $\boldsymbol{N}_{\text{tpkt}}$ equals k is

$$P(N_{tpkt} = k) = (1 - P_{one-hop})^{k-1} P_{one-hop}$$

The expected number of transmissions for a given packet is

$$E[N_{tpkt}] = \sum_{k=1}^{\infty} k(1 - P_{one-hop})^{k-1} P_{one-hop}$$

$\boldsymbol{T}_{\text{Adv}}$ could be approximated as follows:

$$T_{Adv} = E[N_{tpkt}](\frac{\tau_l}{2} + T_{MAC} + T_{tpkt} + T_{ppkt})$$

where $\boldsymbol{T}_{\text{MAC}}$ is MAC delay for a sing packet. $\boldsymbol{T}_{\text{tpkt}}$ is the transmission time for a single packet. $\boldsymbol{T}_{\text{ppkt}}$ is the processing time required by a node after receiving the packet. Then the total communication overhead caused by a bogus Adv during $\boldsymbol{T}_{\text{bAdv, h}}$ is

$$Cost_{high-version} = Cost_{tpkt} S_n \frac{C_b^{k-1}}{C_b^k} \sum_{i=1}^{S_n} I_i$$

$$I_i = \begin{cases} 0 & 0 < t - \frac{\tau_l}{2} h_i \le \frac{\tau_l}{2} \\ 1 & \frac{\tau_l}{2} < t - \frac{\tau_l}{2} h_i \le 2\tau_l \\ n & \tau_l 2^{n-2} + \sum_{j=1}^{n-1} \tau_l 2^{j-1} < t - \frac{\tau_l}{2} h_i \le \min(\tau_l 2^{n-1} + \sum_{j=1}^{n} \tau_l 2^{j-1}, T), n = \left\lfloor \log_2(\frac{\tau_h}{\tau_l}) \right\rfloor \\ \left\lfloor \log_2(\frac{\tau_h}{\tau_l}) \right\rfloor + \left\lfloor \frac{t - h_i - T}{\tau_h} \right\rfloor & t - \frac{\tau_l}{2} h_i > T, T = \tau_l(2^{n+1} - 1) + \frac{\tau_h}{2} \end{cases}$$

where $\boldsymbol{Cost}_{\text{tpkt}}$ is the communication overhead a node broadcasts an Adv. b is the average amount of neighbors for each node (it is decided by network density and communication range of a sensor node). $\boldsymbol{S}_{\text{n}}$ is the number of sensor nodes in the network. I_i is the number of rounds within Deluge during the $\boldsymbol{T}_{\text{bAdv, h}}$. h_i is the number of hops for $node_i$ away from the first malicious node. $\boldsymbol{T}$ is a up bound of the time, when this threshold is exceeded, $\tau_{m,i}$ will be set to 60 seconds [2].

Attack 2: False Request
In Deluge we know that one Req packet sent by a malicious node to a neighbor node which has received packets requested will cause 48 code packets as response.

Meanwhile all neighbor nodes of this malicious node will adjust $\tau_{m,i}$ to the minimal value $\tau_{l.}$. Therefore, the latency introduced by this kind of Attack2, $\boldsymbol{T}_{\text{Latency,}}$ consists of the following components:

$$T_{Latency} = T_{Req} + T_{tGiveUp} + T_{Code}$$

where $\boldsymbol{T}_{\text{Req}}$ is the time used for requesting the code packets. $\boldsymbol{T}_{\text{tGiveUp}}$ is the time which is due to the condition when a node exceeds its limit of λ requests, it must transmit to MAITAIN and wait for another advertisement before making additional requests [2]. $\boldsymbol{T}_{\text{Code}}$ is the time required to send 48 code packets.

$\boldsymbol{T}_{\text{Req}}$ could be calculated as follows based on system model above:

$$T_{Req} = E[N_{tpkt}]E[N_{reqs}](E[t_r] + T_{MAC} + T_{tpkt} + T_{ppkt})$$

where $\boldsymbol{E}[N_{\text{reqs}}]$ is the expected number of requests a node makes to complete a given page and $\boldsymbol{E}[t_{\text{r}}]$ is the expected time between two requests.

$\boldsymbol{T}_{\text{Code}}$ could be calculated as follows:

$$T_{Code} = 48 \times E[N_{tpkt}](T_{MAC} + T_{tpkt} + T_{ppkt})$$

The communication overhead caused by a false Req during $\boldsymbol{T}_{\text{Latency}}$ is

$$Cost_{false-req} = 48 \times E[N_{tpkt}]Cost_{tpkt}$$

Attack 3: Larger-numbered Page

In this attack, the malicious node continuously broadcasts Adv ($\upsilon=\upsilon'$, $\gamma > \gamma'$) containing high page number. Therefore neighboring nodes always keep MAINTAIN state with high rate of advertisements. The communication overhead caused by this kind of attack is

$$Cost_{high-pageid} = Cost_{tpkt}(E[N_{reqs}] + S_{one-hope}\frac{C_b^{k-1}}{C_b^k}\frac{t}{\tau_l})$$

where $\boldsymbol{t}$ is attack time caused by a malicious node. $\boldsymbol{S}_{\text{one-hop}}$ is the number of contending nodes of neighboring nodes of the malicious node in one-hop range.

Attack 4: Lower-version Adv

In that attack a malicious node pretend having an object with old version. As a result, this node will require the transfer of all pages in the code image. The communication overhead caused by Lower-version Adv attack is given by

$$Cost_{lower-version} = Cost_{tpkt}E[N_{tpkt}](S_{one-hope}\frac{C_b^{k-1}}{C_b^k}\frac{t}{\tau_l} + E[N_{reqs}] + E[N_{tGiveUp}] + 48 \times N_{page})$$

where $\boldsymbol{t}$ is attack time caused by a malicious node. $\boldsymbol{N}$page is the number of pages in code updates.

5. Evaluation

We have provided system models of basic insider DoS attacks against Deluge in Section 4. In this subsection, we compare normal Deluge's performance to that of attacking in different cases through using TOSSIM [21] which is a bit-level node simulator designed specifically for the TinyOS platform.

We use two performance metrics in our evaluation: *Communication overhead* and *update completion time*. The communication overhead is measured as the total number of packets transmitted by all the sensor nodes during a code dissemination, which is related to energy consumption. The update completion time is the time required to finish disseminating a code image to all the sensor nodes in the network.

In the simulation, we set the parameters of Deluge as following. For the maintenance service, we set τ_l =2 seconds, τ_h =60 seconds, and k =1. For requests, we set τ_r =0.5, λ=2, and ω=8. And for each set of results, we perform the simulation 10 times using the same topology and then take an average over them. The results of these simulations are presented in the following subsections

Attack 1: Higher-version Adv

In this kind of DoS attack, we repeat the simulation with a 10×10 grid topology network with adjacent nodes spaced 15 feet apart (see Figure 1). The black node denotes a malicious node. And the grey node represents the first node which can receive packets from the base station.

Figure 2 shows the communication overhead of different conditions which are measured as the total number of Adv broadcasted by all sensor nodes in each test case. For the case of free attack, Deluge takes about 10.4 seconds for all nodes in the network to update their image version to υ'. And in this duration the total number of Adv under free attack is transferred only 30% less than the condition under High-version Adv attack.

Figure 3 shows the number of each packet type sent by all sensor nodes during disseminating a code image in the two different conditions respectively.

Among all the simulations, the average number of Adv packets sent under Higher-version Adv attack is 7.5 times more than that of the free attack condition, while the average number of code packets sent under Attack1 is reduced by approximately 90%.

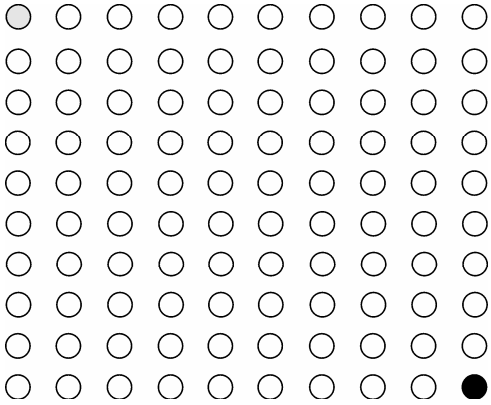

Figure 1. Topology of WSN.

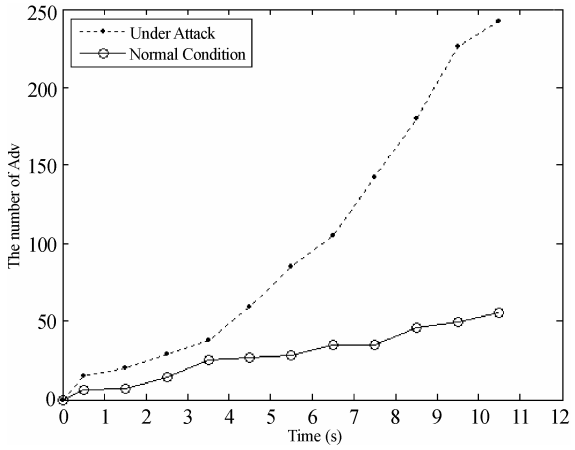

Figure 2. The number of Adv of free attack and Attack1.

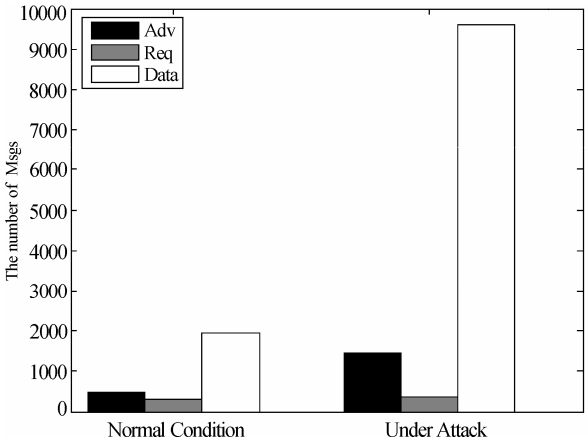

Figure 4. Communication overhead of free attack and Attack2.

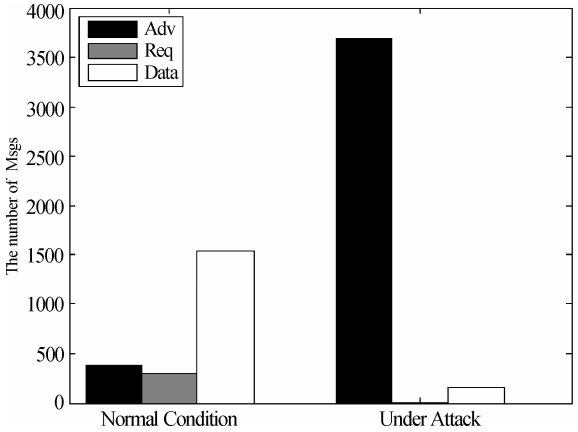

Figure 3. Communication overhead of free attack and Attack1.

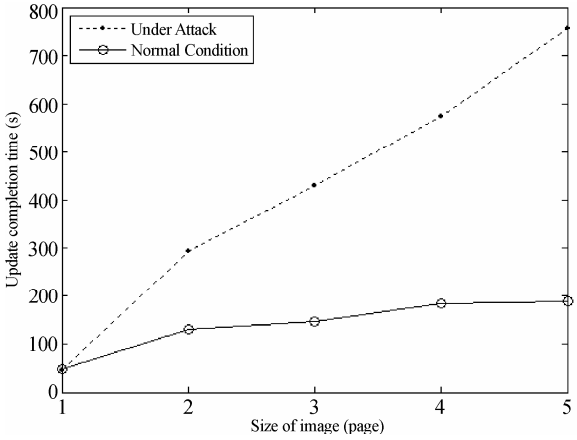

Figure 5. Update completion time of free attack and Attack2.

As we explained earlier, the main reason for this performance difference is that the malicious node enforces legitimate nodes to transmit more Adv packets by using the dynamic adjustments of the advertisement rate mechanisms. When the bogus Adv contains a higher image version than the image injected by base station, infected nodes are prevented from requesting the new image from base station. Therefore the number of Req and code packets decrease dramatically.

Attack 2: False Request

In this simulation with a 10×10 grid topology network with neighboring nodes spaced 15 feet apart. We let a malicious node always propagate Req for the first page of a new image.

Figure 4 shows the number of each packet type sent by all nodes during disseminating a code update in the two cases respectively.

From Figure 4 we can see that under free attack condition, the control packets (Adv and Req) occupy about 18.18% of total messages, while code packets take the left 81.82%. This results fits fairly well that of Deluge [2]. In contrast, under the False Request attack condition, the number of code packets increases 3.9 times more than that of the normal condition) due to bogus Req sent by the adversary.

Figure 5 shows the update completion time of the whole network for different size of images in two conditions. Under the False Request attack, when five pages of the code image have been disseminated, it takes 4 times longer than that of the free attack. From Figure 5 we can also see that when there is only one page in an image, this attack could not block reprogramming because the malicious node always propagates the Req for first page.

The additional latency introduced by Attack2 is due to suppressing other Req from normal nodes when the adversary sends Req modified which included a smaller page number.

These simulation results demonstrate that the False Request attack introduces a large amount of unnecessary transmission of code packets which would cause energy consumption of resource-constrained sensor nodes.

Attack 3: Larger-numbered Page

We simulate this attack in a 10×10 grid topology network with nodes spaced 15 feet apart and set the attack time to be 60 seconds. During this period, the malicious node continuously broadcasts Adv ($\upsilon=\upsilon'$, $\gamma>\gamma'$) to its neighboring nodes.

Figure 6 shows the number of each packet type sent by all sensor nodes during disseminating a code image in the two different conditions respectively.

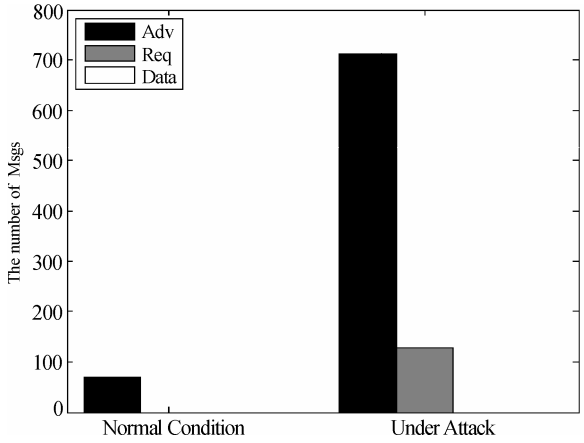

Figure 6. Communication overhead of free attack and Attack3.

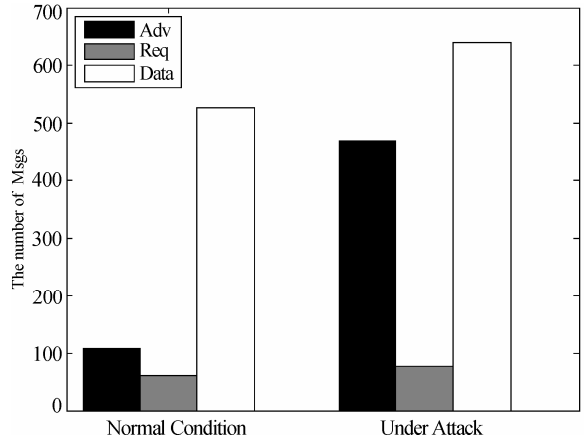

Figure 7. Communication overhead of free attack and Attack4.

In Figure 6, the average number of Adv packets sent under Large-numbered Page attack is 7.8 times more than that of the free attack condition, while the average number of Req packets sent under Attack3 increase approximately to 150 packets. Because $\tau_{m,i}$ is minimized, the rate of advertisements increases, the total number of Adv is much greater compared with that of free attack condition.

These simulations confirm that Large-numbered Page attack eventually waste sensor nodes' energy through inducing a number of meaningless Req and Adv packets.

Attack 4: Lower-version Adv

The simulations were performed in a 10×10 grid topology network with nodes spaced 15 feet apart. Figure 7 shows the number of each packet type sent by all sensor nodes during disseminating a code image in the two different conditions respectively.

In Figure 7, the average number of code packets under Lower-version Adv attack is about 17% more than that of the case under free attack. Meanwhile the average number of Adv packets under Attack4 increase approximately to 500 packets which are about 2 times more than that of the free attack.

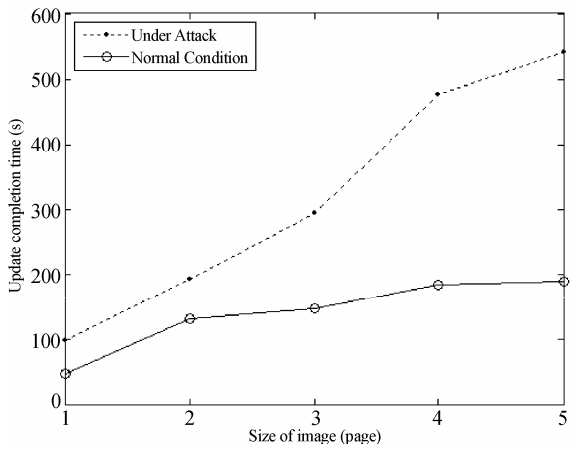

Figure 8. Update completion time of free attack and Attack4.

Figure 8 shows the update completion time of the whole network for different size of images in two conditions.

From Figure 8 we can see that when the network is under attack the larger code image is, the longer the delay would be compared with normal condition. Under the Lower-version Adv attack, when five pages of the code image have been disseminated, it takes 2.6 times longer than that of the free attack.

From the simulation results we know the main reason for this performance is that this attack can cause the delay of code dissemination while extra resource consumption is introduced.

Attack 5: Same-version Adv

In this simulation with a 10×10 grid topology network with neighboring nodes spaced 15 feet apart. Figure 9 shows the update completion time of the whole network for different size of images in two conditions.

Under the Same-version Adv attack, when five pages of the code image have been disseminated, it takes about 42% longer than that of the free attack.

The additional latency introduced by Attack2 is due to a compromised node could exploit suppression mechanisms of Deluge to reduce the Adv transmission of legitimate nodes.

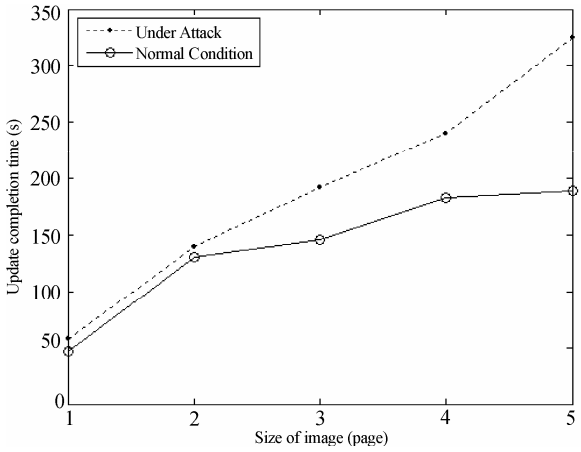

Figure 9. Update completion time of free attack and Attack5.

6. Conclusions and Future Work

In this paper we identify five different basic types of insider DoS attacks exploiting the epidemic propagation strategies used by Deluge. They are Higher-version Advertisement attack, False Request attack, Larger-numbered Page attack, Lower-version Adv attack, and Same-version Adv attack. We also proposed the simple system models for these DoS attacks to try to find out the impact of those attacks on Deluge. Despite the fact that Deluge is an efficient protocol for code propagation in WSNs, it is susceptible to different kind of attacks. To understand more deeply about them, we simulate these five basic types of insider DoS attacks by using TOSSIM and report the detailed statistical results.

There are still many issues that need further investigation to make reprogramming highly available although some recent works have attempted to provide DoS-Resistant code dissemination in WSNs [23,24]. Y. Zhang, et al. proposed a public-key scheme called "combined public key" to secure Advertisement and Request packets. The ignorable problem with the approach is the resource requirement [24]. Seluge is the latest work on secure code dissemination and is a solution that seamlessly integrates the security mechanism and original deluge. Unfortunately, although the current version of Seluge adopts the cluster key approach to provide authentication of ADV and SNACK packets [23], it cannot uniquely identify senders. As a result, a compromised node can still pretend to be its neighbors using their cluster keys to launch DoS attacks. In our future work, we will investigate techniques to detect insider DoS attacks exploiting the Deluge epidemic and suppression mechanisms. Finally, we plan to provide Insider-DoS-Resistant code dissemination scheme.

7. Acknowledgment

We would like to thank the anonymous reviewers for their valuable comments.

8. References

[1] T. Stathopoulos, J. Heidemann, and D. Estrin, "A remote code update mechanism for wireless sensor networks," Technical Report, UCLA, Los Angeles, CA, USA, 2003.

[2] J. W. Hui and D. Culler, "The dynamic behavior of a data dissemination protocol for network programming at scale," in ACM International Conference on Embedded Networked Sensor Systems, pp. 81–94, November 2004.

[3] S. S. Kulkarni and L. Wang, "MNP: Multihop network reprogramming service for sensor networks," in International Conference on Distributed Computing Systems (ICDCS'05), June 2005.

[4] S. S. Kulkarni and M. Arumugam, "INFUSE: A TDMA based data dissemination protocol for sensor networks," Technical Report, Michigan State University, East Lansing, MI, USA, 2004.

[5] V. Naik, *et al*., "Sprinkler: A reliable and energy efficient data dissemination service for wireless embedded devices," 26th IEEE Real-Time System Symposium, December 2005.

[6] L. A. Phillips, "Aqueduct: Robust and efficient code propagation in heterogeneous wireless sensor networks," Master's thesis, University of Colorado at Boulder, 2005.

[7] M. D. Krasniewski, R. K. Panta, S. Bagchi, C. L. Yang, and W. J. Chappell, "Energy-efficient on-demand reprogramming of large-scale sensor networks," ACM Transactions on Sensor Networks, 4(1): pp. 1–38, 2008.

[8] P. E. Lanigan, R. Gandhi, and P. Narasimhan, "Sluice: Secure dissemination of code updates in sensor networks," in IEEE International Conference on Distributed Computing Systems, Lisbon, Portugal, July 2006.

[9] P. K. Dutta, J. W. Hui, D. C. Chu, and D. E. Culler, "Securing the Deluge network programming system," in the Fifth International Conference on Information Processing in Sensor Networks (IPSN'06), 2006.

[10] J. Deng, R. Han, and S. Mishra, "Secure code distribution in dynamically programmable wireless sensor networks," in Proceedings of the Fifth International Conference on Information Processing in Sensor Networks (IPSN'06), April 2006.

[11] Crossbow Tech Inc., Mote In-Network Programming User Reference, http://www.tinyos.net/tinyos-1.x/doc/Xnp.pdf, 2003.

[12] TinyOS: An open-source OS for the networked sensor regime, http://www.tinyos.net/.

[13] P. E. Lanigan, P. Narasimhan, R. Gandhi, "Tradeoffs in configuring secure data dissemination in sensor network: An empirical outlook: [CMU-CyLab-07-006]," CyLab, Carnegie Mellon University, PA, 2007.

[14] A. Perring, R. Szewczyk, J. D. Tygar, V. Wem, and D. E. Culler, "SPINS: Security protocols for sensor networks," Wirelss Networks, 8(5): pp. 521–534, 2002.

[15] H. Chan, A. Perrig, and D. Song, "Random key predistribution schemes for sensor networks," in IEEE Symposium on Research in Security and Privacy, 2003.

[16] C. Karlof and D. Wagner, "Secure routing in wireless sensor networks: Attacks and countermeasures," Sensor Network Protocols and Applications (SNPA 03), May 2003.

[17] A. D. Wood and J. A. Stankovic, "Denial of service in sensor networks," IEEE Computer, pp. 48–56, October 2002.

[18] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy efficient communication protocol for wireless microsensor networks," in the 33rd Hawaii International Conference on System Sciences, 2000.

[19] P. Levis, N. Patel, D. Culler, and S. Shenker, "Trickle: A self-regulating algorithm for code propagation and maintenance in wireless sensor networks," NSDI 2004, pp. 15–28, 2004.

[20] J. Deng, R. Han, and S. Mishra, "Practical study of transitory master key establishment for wireless sensor

networks," in 1st IEEE/CreateNet Conference on Security and Privacy in Communication Networks (SecureComm 2005), Athens, Greece, pp. 289−299, September 2005.

[21] P. Levis, N. Lee, M. Welsh, and D. Culler, "TOSSIM: Accurate and scalable simulation of entire tinyos applications," in Proceedings of the First ACM Conference on Embedded Networked Sensor Systems (SenSys 2003), ACM Press, November 2003.

[22] D. R. Raymond and S. F. Midkiff, "Denial-of-service in wireless sensor networks: Attacks and defenses," IEEE Pervasive Computing, 2008.

[23] S. Hyun, P. Ning, A. Liu, and W. L. Du, "Seluge: Secure and DoS-resistant code dissemination in wireless sensor networks," In Proceedings of the Seventh International Conference on Information Processing in Sensor Networks (IPSN'08), April 2008.

[24] Y. Zhang, X. S. Zhou, Y. M. Ji, Z. Y. Fang, and L. F. Wang, "Secure and DoS-resistant network reprogramming in sensor networks based on CPK," 4th IEEE International Conference on Wireless Communications, Networking and Mobile Computing, 2008.

