SCID: Secured Code Image Dissemination in Wireless Sensor Networks
Vikram Shiroi, Aruna Kumar Joshi2
Associate Professor1, Assistant Professor2
Department of CSE
SKSVMACET-Laxmishwar, Karnataka, India

Abstract:
Wireless sensor networks (WSNs) are used in important domains (e.g., health care, military, critical infrastructure) wherever it’s necessary that the nodes be reprogrammed with a replacement or changed code image without removing them from the deployed place. Different protocols are developed for the dissemination of code pictures between sensors in multi-hop WSNs, where these sensing nodes may have different levels of link quality. However, the code dissemination method in these protocols is difficult for (someone) to do something to the nodes with poor link quality. This results associate magnified number of retransmissions and code dissemination time. Moreover, in many of the techniques, the code dissemination method is not secure and may be eavesdropped or discontinuous by a malicious wireless sensing element node within the transmission range. This research paper shows a direct approach, Secured Image Dissemination in Wireless Sensor Networks (SID), to boost the present code dissemination protocol mistreatment the accessible resources within the sensors. Specifically, our approach adapts to the variable link conditions via dynamic packet filler to cut back the quantity of retransmissions and overall code dissemination time. Our approach additionally provides confidentiality and integrity to the code dissemination method by utilizing energy-efficient coding and authentication mechanisms with RC4 and also the CBC-MAC. We have evaluated SID in an exceedingly network of real sensors and the results show that adjusting the packet size as a operate of link quality reduces the retransmitted data by 93% (ninety three) and also the image UTC by 35% (thirty five) in comparison to the present code dissemination protocols. The trade-offs between dependableness, security overhead, and overall UTC for SID is also mentioned.

Index Terms: Wireless Sensor Networks, Secure Code Dissemination, Link Quality Indicator (LQI)

I. INTRODUCTION

Sensors are deployed in various locations such as underwater, volcano regions and underground, making them hard to recover or replace for different reasons like upgrade, repair, changing the parts of sensor nodes. As a result of this numerous preparation of wireless sensor networks (WSNs) and lots of functionalities provided by them, reprogramming the deployed detector nodes through wireless links becomes a necessary and fascinating task. For instance, if software running on the node needs related update as a result of a security patch or extra practicality, it would be necessary to interchange the present code on the node with the new updated code. This method of diffusive code over wireless links is named code dissemination. Moreover, under hostile conditions, an attacker might try to modify or get access to disseminated code. The attacker might additionally attack the dissemination method itself by injecting malicious code dissemination packets to exhaust the restricted energy of the sensor nodes. Therefore, it’s imperative that the dissemination process be secured. In addition to security, code dissemination protocols should address the variation in link quality between nodes. The nodes with poor link quality would hinder other nodes within the network from continuing forward within the code dissemination method. Moreover, such nodes with poor links would yield accrued retransmissions which might additional degrade the security of the system as an intruder would have multiple opportunities to intercept the disseminated packets. In this paper, we tend to introduce a Secure and Link-Quality cognizant Image Distribution (SID) technique for WSNs that aims at reducing the number of retransmitted code dissemination packets. SID also provides energy economical security services, confidentiality using the RC4 [1], [2] cryptography algorithmic rule and integrity using the CBC-MAC provided by the CC2420 [3] transceiver module of the sensors (e.g., MICAz and TelosB). The results from our experiments victimization real sensors show that code dissemination using SID is a lot of economical than alternative existing code dissemination protocols, in terms of code dissemination time and total variety of retransmissions. The experimental results show that SID reduces the retransmission bytes by 93% and also the image UTC by 35% compared with the prevailing code dissemination protocols below poor link quality conditions the contributions of work for this research are shown in 3 directions:

1) We tend to create and implement a secure code dissemination protocol - SID;

2) We tend to contemplate link quality between nodes throughout the dissemination process; and

3) We tend to dynamically adapt the packet size supported link quality. As a result, our research work outperforms previously planned techniques the research work as follows: existing code
dissemination protocols for WSNs is mentioned in Section II. To inspire our work, associate degree analysis of the issues encountered by the existing code dissemination protocol is provided in Section III. Section IV discusses the new proposed method, SID, and explains its implementation intimately. Evaluation of SID and comparison of SID with the existing code dissemination protocols are shown in Section V. Section VI summarizes the advantages of SID over existing code dissemination protocols and concludes the paper.

II. LITERATURE SURVEY

The Deluge Data Dissemination Protocol contains epidemic behavior. That means code is propagated because nodes can overhear from their neighbors when a new code is being broadcasted. Specifically, each node advertises the most recent version of the data object it has available. Nodes that realize that they are running an older version, request the newest one. Sensors receiving requests broadcast the desired data. Nodes that get updated, advertise the newly received data in order to help data propagation in the remaining nodes. Deluge is density-aware in a sense that redundant messages are suppressed so as to increase the efficiency of the protocol. In addition to that, if a node has not received its data after making a particular number of requests, it tries finding another sender. Moreover, Deluge changes dynamically the rate of advertisements to allow quick dissemination when required while saving resources when new code propagation is not needed. What is more, Deluge makes use of pipelining to support parallel transfers of data images. To achieve pipelining Deluge divides the data object into fixed-size pages. In many cases, an update may require only some minor changes from the previous older code. Therefore, nodes would need only the pages which have changed. A version number is being used to distinguish the different updates. Because nodes need to know which pages have changed so as to update them, an object profile is being advertised to describe a certain object. This object profile represents how old a page is. A node receiving an advertisement uses the object profile to determine which pages need to be upgraded and requests these pages. The Secure Network Programming Protocol [6] uses public private key encryption to sign the advertisement packets of the code image and SHA-1 for computing the hash of every packet. During this work, the computed hash of 1 future packet is embedded into the payload of the previous packet in sequence and the hash of the primary packet is embedded into the advertisement packet which is signed. So, a receiver node, upon authenticating the advertisement packet, verifies the integrity of successive received packet like a shot and saves or discards the packet supported the computed. However, hash within the case of out-of-order delivery, the cache receiver must The packet and watch for the previous packet in sequence to verify the cached packets. This could be utilized by the attacker to inject fake packets and expend the cache memory of the receiver node leading to a denial-of-service attack. Seluge [7] is basically a secure version of Deluge and so, it preserves the page-by-page propagation technique of Deluge. Almost like [6], a hash of every packet in an exceedingly page computed is Victimization SHA-1 and this worth is embedded within the Correspondingly sequenced packet of the previous page. So, once all the packets in an exceedingly page ar received, the receiver has all the hash values for the packets within the next page. Seluge conjointly uses the Elliptic Curve Digital Signature algorithmic rule (ECDSA) to sign the initial code image publicity. However, the SHA-1 and therefore the ECDSA algorithmic rule used for intense processes [8]-[10]. Another recent work on the code dissemination process in WSNs, DiCode [11], provides a secure and distributed code dissemination protocol. As opposed to the aforementioned centralized approaches [4],[6],[7], the key difference in this technique is the usage of distributed control for code dissemination. However, the security features used in this approach are similar to that of Seluge. The experimental results also indicate that DiCode has longer propagation delays when compared to Deluge and Seluge. Although [11] proposes a distributed code dissemination protocol, it does not address any of the inter-node link quality issues. Nonetheless, as discussed in [12] with experimental results, link quality prediction is important for better system provisioning and resource management. Moreover, the experiments conducted on IEEE 802.15.4 Zigbee radios with the CC2420 chipset illustrate that the Link Quality Indicator (LQI) is linear with respect to the instantaneous SNR. Finally, it is shown that LQI can be used as an effective parameter to predict instantaneous link quality between nodes [12]. Therefore, in this paper, we propose SID, which provides secure and link-quality aware code image dissemination for WSNs.

III. EXISTING SYSTEM AND MOTIVATION

A common drawback in several wireless networks is variable and poor link quality of the channel between the communication nodes [12]. Throughout communication, a reduction in the link quality increases the number of retransmissions between the nodes to transfer useful information. The retransmissions have a major impact on WSNs that consists of resource-constrained device nodes. The code dissemination protocol in TinyOS, Deluge [4], uses a hard and fast packet size to transmit the code image between the wireless device nodes and is at risk of enlarged packet loss below poor link quality conditions. For an equivalent code image size, due to the increased packet loss rate, the code dissemination protocol in TinyOS [4] additional retransmission bytes to pass around the code image as analyzed in Section V. This, successively can increase the whole code dissemination time between the nodes as orders to research this, we have a tendency to conduct an easy preliminary experiment to calculate the packet retransmission rate for various packet sizes with totally different link quality conditions between a combine of MICAz motes. The link quality of the channel is calculable victimization the LQI worth within the MICAz molecule. In the experimental setup, the primary molecule transmits fifty packets and the range of retransmissions is calculated. The experiment is recurrent ten times, with an equivalent packet size and link quality condition. The common retransmission per fifty packets is calculated and also the average Packet Retransmission magnitude relation (p) is calculated with the common retransmission worth victimization Equation 1,

\[ \psi = \frac{\phi}{t} \]  

(1)

Where, average Packet Retransmission Ratio that the range of retransmitted packets and is that the total range of packets transmitted, together with the retransmitted packets:
In Equation 3, \( \sigma \) is the minimum integer value, starting from zero, that satisfies the equation and \( n \) is the number of packets per page. Figure 1(b) shows the number of retransmitted bytes per page in the code dissemination protocol for various packet sizes in different LQI ranges. The figure shows that for an increase in packet size up to 55 bytes, the retransmitted bytes per page remains zero for the good and best LQI ranges. However, when the packet size exceeds 55 bytes, the retransmissions increase gradually for the good LQI range, while it remains zero for all packet sizes in the best LQI range. Since transmitting a page with minimum retransmissions reduces the code dissemination time, to achieve better code dissemination under fluctuating link quality conditions, we propose a LQI based adaptive packet size code dissemination protocol. This protocol samples the LQI of all the receivers and determines the optimal packet size for disseminating a page. The detail of the scheme is explained in Section IV.

**IV. PROPOSED METHOD**

SID technique, that builds the code dissemination protocol of TinyOS, Deluge [4], was enforced and tested in a very network of MICAz sensor motes. SID has two specific options. First, is that the dynamic, adaptive packet size estimation that determines associate best packet size for transmitting page using the LQI values of the request messages. Second, it provides security for code dissemination with energy efficient, stream cipher RC4 encryption [2] and using the CBC-MAC. The small prints of the implementations are explained below.

### A. Dynamic Adaptive Packet Size Algorithm

The experiments in Section III illustrate that different LQI ranges have different optimal packet sizes. So, using an adaptive packet size in the code dissemination process is instrumental to reduce the per page dissemination time. Hence, from the experiment in Section III, the optimal data packet size values for different LQI ranges have been determined and are shown in Table I.

<table>
<thead>
<tr>
<th>LQI Optimal</th>
<th>LQI Value Range</th>
<th>Payload Size (Bytes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Poor LQI</td>
<td>&lt; 90</td>
<td>5</td>
</tr>
<tr>
<td>Average LQI</td>
<td>99 - 105</td>
<td>20</td>
</tr>
<tr>
<td>Good LQI</td>
<td>105 - 104</td>
<td>40</td>
</tr>
<tr>
<td>Best LQI</td>
<td>105 - 110</td>
<td>80</td>
</tr>
</tbody>
</table>

Note that the data packet size in the experiments in Section III illustrate that different LQI ranges have different optimal packet sizes. So, using an adaptive packet size in the code dissemination process is instrumental to reduce the per page dissemination time. Hence, from the experiment in Section III, the optimal data packet size values for different LQI ranges have been determined and are shown in Table I. Note that the data packet size in receiver and transmitter nodes are shown in Algorithm 1 and Algorithm 2 respectively.
Algorithm 1 Receiver Node

for each received data message do
if DataPktSize \neq CurrPktSize then
modify parameters()
end if
end for

function modify parameters()
\delta \leftarrow DataPktSize/CurrPktSize
CurrPktSize \leftarrow DataPktSize
NewBitvectorSize \leftarrow CurrPktSize/\delta
NewBitvector \leftarrow Bitvector \delta
end function

Algorithm 2 Transmitter Node

for each request message received do
lqi \leftarrow LQI(RcvdPkt)
map (src_id, lqi) lqi_map \leftarrow (SrcId, lqi)
end for

for first data message transmission do
if retransmission then
compute p(lqi_map)
DataPktSize \leftarrow DataPktSize(p)
modify parameters()
else if not retransmission then
compute p(lqi_map)
MA \leftarrow p*sf + (1 - sf)*MA
DataPktSize \leftarrow DataPktSize(MA)
modify parameters()
end if
end for

In the transmitting node, if it is a first round of data transmission for a page, the moving average (MA) is calculated as a function of the average LQI (p) and scaling factor (sf) as the following:

\[ MA = p \times sf + (1 - sf) \times MA \]  \hspace{1cm} \text{(4)}

In Equation 4, the scaling factor (sf) is chosen such that the current LQI sample is given more weight than the previous LQI samples and any temporary fluctuation in the current LQI sample does not immediately affect the packet size estimation. Hence, the scaling factor (sf) is given a value of 0.4 based on our experiments. Note, for values less than 0.4 the moving average took longer to converge to an optimal LQI estimate, whereas, for values greater than 0.4 there are increased fluctuations in the LQI estimate. The data message packet size (Data Pkt Size) is determined by the estimated moving average, according to the data mapping in Table I. In the case of retransmission, only the nodes with poor LQI will request more packets. So to reduce packet losses, the DataPktSize during retransmission is determined according to the current average LQI p estimate instead of the moving average.

\[ \delta = \frac{\text{New packet size}}{\text{Current packet size}} \]  \hspace{1cm} \text{(5)}

After determining the new packet size, the multiplication factor p is calculated using Equation 5. If the new packet size is greater than the current packet size, then _ number of current packets will be combined together into a single packet and transmitted. If the new packet size is less than the current packet size, then the current packet will be split into number of packets and transmitted. In SID, the current packet can be divided into smaller packets or, two or more current packets can be combined into a single packet. To facilitate this, the payload size value for different LQI ranges in Table I are chosen such that the bigger packet size values are a multiple of all other smaller packet size values. After determining the new packet size, the transmitter communicates it to the receiver using the data packet header. So, there is a one byte overhead in the data packet in our scheme when compared to the code dissemination protocol in TinyOS, Deluge [4]. On receiving the first data packet of a page, the receiver will check for a change in packet size. If the new packet size value is greater or less than the current packet size, the receiver computes the _ value and populates the new bit-vector, which consists of the packet identifiers of packets to be transmitted, based upon the packets requested in the old bit-vector and _ value. After updating the new bit-vector, it checks whether the packet number of the received packet is requested by this node. If it was requested by this node, it stores the data, otherwise, it discards it.

B. Secure Dissemination.

The three different types of messages involved in the code dissemination process are advertisement messages, request messages and data messages. Securing each of these message types protects against different types of attacks. For instance, securing only the data messages and not providing integrity for the request messages or advertisement messages will enable an attacker to perform a denial-of-service attack by flooding the nodes with request messages or advertisement messages, respectively. In SID, we ensure an increased level of security by encrypting all three types of messages using RC4 encryption and provide integrity using the CBC-MAC. The CBC-MAC integrity check is fast as it is provided by the CC2420 hardware module. We use a 64-bit key for CBC-MAC which adds only 8 bytes of overhead to the overall payload. Whereas, RC4 encryption uses a 128-bit key and being a stream cipher technique, it performs an in-place encryption and does not add any data header overhead.

V. PERFORMANCE ANALYSIS

In this section, we evaluate the performance of SID (with and without the security) under varying link quality conditions and compare it to the existing code dissemination protocols for TinyOS - Deluge [4] and Seluge [7]. Two different experiments were performed in order to analyze and expose the benefits of SID: Experiment 1: An experimental setup consisting of two MICAZ sensor motes is used. A binary image of 35,276 bytes is transferred between the nodes using the default code dissemination protocol (Deluge) [4], SID without security and SID with security. The three techniques mentioned above are compared with respect to the average number of retransmitted bytes per page and the average transmission time per page. This experiment is performed for different link quality conditions. Figures 2(a) and 2(b) illustrate the results obtained from the first experiment. As seen in Figure 2(a), for the best LQI range, the default code dissemination protocol [4], SID with security and
SID without security do not exhibit any retransmissions. Similarly, for the good LQI range, only a small amount of retransmissions are encountered by all three techniques. As the link quality decreases, Deluge experiences increased retransmissions as compared to SID with and without security. Furthermore, at very poor link quality conditions, Deluge exhibits a steep increase in the number of bytes retransmitted, whereas SID with and without security maintains an almost steady number of retransmitted bytes. Moreover, under such low link quality cases,

\begin{align*}
\text{SID without security do not exhibit any retransmissions. Similarly, for the good LQI range, only a small amount of retransmissions are encountered by all three techniques. As the link quality decreases, Deluge experiences increased retransmissions as compared to SID with and without security. Furthermore, at very poor link quality conditions, Deluge exhibits a steep increase in the number of bytes retransmitted, whereas SID with and without security maintains an almost steady number of retransmitted bytes. Moreover, under such low link quality cases,} \\
\text{Deluge has 23 bytes and SID has 20 bytes. Again, due to the security overhead in each packet, SID with security takes slightly longer to transfer a page than the other two schemes. When the link quality further deteriorates and reaches the poor link quality condition, Deluge suffers a large number of retransmissions and takes a very long time to transfer a page. Whereas, it can be seen that SID with and without security transfers a page in a relatively short time. For performance reasons, Deluge keeps the radio switched on during code dissemination [4] and any decrease in the code dissemination time will reduce the energy consumed by all the nodes in a network. So, reduction in the per page transmission time by SID, in turn reduces the energy consumption of the nodes and proves that SID is more energy efficient than Deluge. Furthermore, Seluge [7], consumes more time to do the integrity check using the SHA-1 algorithm. SHA-1 takes around 15 milliseconds to compute the hash of a packet and the inter-packet transmission time is increased from 2 to 17 ms in order to accommodate the computation time [7], which will increase the per page transmission time. Using the performance results from Seluge [7], a comparison of per page transmission time is shown in Figure 2(b). SID with security outperforms both Deluge and Seluge under poor link quality conditions.}
\end{align*}

**Experiment 2:** The second experiment consisted of three levels of sensor nodes with two MICAz sensor motes at each level, where each level is separated by a distance of one hop. While the previous experiments show the benefits of SID under various link quality conditions, the primary motive of this experiment was to prove the effectiveness of SID over the default code dissemination protocol [4] under very good link quality conditions as well. The time taken by the node in the last level to receive the entire binary image (35,276 bytes) is evaluated for both the default code dissemination protocol [4] and SID. The results obtained from the second experiment are presented in Table II. The total time taken for the binary image transfer from source node to intermediate nodes and from source node to sink node is evaluated. All the nodes in the experiment are maintained under very good link quality conditions such that the retransmissions are largely reduced. This experiment reiterates that SID performs better than Deluge in terms of total transmission time for the entire binary image even under good link quality conditions. This also illustrates that SID makes better use of the channel when the link quality between nodes is good, whereas, Deluge does not make optimum use of the channel under good link quality conditions.

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Node</th>
<th>Total Image Transfer Time (Sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deluge</td>
<td>intermediate sink</td>
<td>147.493</td>
</tr>
<tr>
<td></td>
<td>sink</td>
<td>296.380</td>
</tr>
<tr>
<td>SID</td>
<td>Intermediate sink</td>
<td>96.574</td>
</tr>
<tr>
<td></td>
<td>sink</td>
<td>198.218</td>
</tr>
</tbody>
</table>

**VI. CONCLUSION**

Wireless sensor nodes are always deployed in groups and the link quality between different pairs of nodes is not the same in
the deployed area. Also, the code is disseminated hop-by-hop in a multi-hop WSN, where the delay in disseminating the code between the nodes with poor link quality hinders the overall code dissemination time. Our experimental results with real sensors show that using a fixed packet size in the current code dissemination protocols will increase the retransmissions between the nodes with poor link quality and hinder the overall code dissemination time. Our code dissemination protocol SID, samples the link quality of the channel using LQI as a metric and determines the optimal packet size before transmitting a page. The performance analysis of SID in the experiments show that by dynamically adapting the packet size, retransmission bytes are reduced by 93% and the per page transmission time is reduced by 35% for poor LQI values. Results also show that SID, during the good link quality condition, exploits the channel and disseminates the code faster when compared to the other code dissemination protocols. SID with security uses the simple energy efficient stream cipher encryption algorithm RC4 [2] and a hardware based hashing function CBC-MAC for providing secure code dissemination. Since SID uses a dynamic adaptive packet size technique, the overhead of providing secure dissemination is reduced when compared to Seluge [7]. Our experimental results also show that SID with security outperforms the TinyOS code dissemination protocols Deluge [4] and Seluge [7] under the poor LQI ranges. In our future work, we will improve the performance of SID in the average LQI ranges, where it performs similar to the default TinyOS code dissemination protocol. Also, calculating the average LQI value at both the transmitting and receiving ends will yield a better link quality estimate in a bidirectional channel.

VII. REFERENCES