HUSTLE:
Deploying A Secured Wireless Sensor Network by Light Communication with Smartphone

Nguyen Doan Minh Giang

Faculty of Environment and Information Studies

Keio University

5322 Endo Fujisawa Kanagawa 252-8520 JAPAN

Submitted in partial fulfillment of the requirements for the degree of Bachelor

Advisors:

Professor Hideyuki Tokuda
Professor Jun Murai
Associate Professor Hiroyuki Kusumoto
Professor Osamu Nakamura
Associate Professor Kazunori Takashio
Assistant Professor Rodney D. Van Meter III
Associate Professor Keisuke Uehara
Associate Professor Jin Mitsugi
Lecturer Jin Nakazawa
Professor Keiji Takeda

Abstract of Bachelor's Thesis

HUSTLE: Deploying A Secured Wireless Sensor Network by Light Communication with Smartphone

Until now, deploying a secured Wireless Sensor Network (WSN) at home environment still requires users to have certain skills like WSN configuration, security configuration. At the same time, WSN has become popular and WSN has a lot of potential applications at home environment such as home security, remote health services, smart home and children supervising applications.

In this thesis, we propose a method called HUSTLE. HUSTLE provides regular users with an easier and faster method to deploy a secured WSN. HUSTLE connects sink node and new sensor nodes to exchange one-time security key and identifier between the nodes. HUSTLE will be used to set-up those new sensor nodes into the WSN via two interactions, which are (1) setting up new sensor nodes one by one by touching a smartphone, and (2) setting up new sensor nodes simultaneously by directing a smartphone's flashlight to all of them.

To examine the feasibility and effectiveness of HUSTLE, we implemented a quantitative study based on a series of experiments. The result shows that our HUSTLE can help users set-up WSN six times faster than existing method which mainly rely on selecting true sensor nodes and inputting security code manually. In addition, we requested participants to evaluate HUSTLE by referring to criteria including: "Simple to use", "Tiring to use", "Useful", "Easy to learn", "Easy to fix in case of errors". The result shows that HUSTLE can provide users to setup a secured WSN more simply, faster.

Giang Doan Minh Nguyen

Faculty of Environment and Information Studies, Keio University

卒業論文要旨 2012 年度 (平成 24 年度)

HUSTLE：光通信を用いた, セキュアな WSN 設定手法

近年，ワイヤレスセンサネットワーク (WSN) が普及し，様々な環境で利用されるようになった．例えば，WSN を用いた火山のモニタリングや地震のモニタリングを行う研究がある．またそれらの情報は天気予報などに応用されている．また，近い将来私たちの家庭環境においても警備アプリケーションや子供の見守りシステムなどが普及すると考えられる．

それらのシステムやアプリケーションを構築するためにはセキュアな WSN が必要となる．しかし，既存のセキュアな WSN を構築するための手法は WSN に関する知識や経験が必要なので，エンドユーザには困難である．

そこで，本論文では HUSTLE と言う手法を提案する．HUSTLE でエンドユーザでも簡単にセキュアな WSN を構築するすることができる．ユーザはスマートフォンを用いたインタラクションを行うことで WSN の構築をセキュアに行う．HUSTLE では以下の 2 つのインタラクションを提案する．スマートフォンの LED 点滅を照射する手法と，センサノードの LED の点滅をスマートフォンの照度センサで受信する手法である．どちらの手法もセキュアな光通信を用いるため，セキュアな WSN を構築することが可能である．

定量的評価の結果，HUSTLE を用いると既存手法の約 6 倍早く WSN を設定することが可能であった．ユーザビリティ評価では，ユーザは HUSTLE の方がより簡単で使いやすいと回答した．以上により，HUSTLE を用いることでセキュアな WSN を構築することが可能であると言える．

グエン・ゾアン・ミン・ザン

環境情報学部，慶應義塾大学

Contents

List of Tables

List of Figures

Chapter 1

Introduction

This chapter describes the background of our research and the overview of the problem.

1.1 Background

Nowadays with advances in hardware and wireless network technologies, we can make multi-functional tiny sensor devices through low-cost, low-power consumption methods. By using hundreds of sensor devices with several types of sensor nodes, we can make a Wireless Sensor Network (WSN). We can use WSN for collecting, processing and analyzing data in a large area. And we can make a lot of applications with it such as environment observation and surveillance on remote health services. The cost of sensor node is depreciating, thus we can apply WSN at home environment in near future.

On the other hand, to use WSN at home, we need setup it at first. This is not only at first time, but also need to change error sensor nodes, reinstall some sensor nodes for change struct of network such as change target or change application. And it is better if users can make a WSN by themselves. Therefore, for home environment usage, WSN needs to meet the following requirements:

- Easy deployment and modification: Because it is used by regular users who do not have skills about WSN, thus a simple method is required to setup WSN, modify required sensor nodes.

- Fast setup and modify: A lot of applications have hundred sensor nodes. An example can be found in environment observation or security monitoring applications, in which users have to put sensor nodes at some where, then it requires a fast method to setup number of sensor nodes.

- Maintaing security of WSN: For home environment usage such as remote health services or security application, privacy of users is an important requirement. For this, data communication inside WSN needs to be protected.

- Low cost: For home with regular users, the cost of WSN also become importance, with cheaper WSN we can make it more popular. Therefore we need make a WSN cheap as cheap possible.

In daily task, to ensure privacy of the user, we always use lock, password, security key. For example, we can set password for laptop, Wi-Fi or Smartphone to protect it. We will input password from physical keyboard, virtual keyboard or biometric devices such as camera, microphone and fingerprint. But sensor nodes must be tiny, cheap and low energy consumption, thus we have to limit hardware of it. We have to avoid unnecessary hardware, therefore it is impossible to directly input password, security key to sensor nodes.

Nowadays, users have to select true sensor nodes and input correlative security code to a manage computer for adding this sensor node with securely. This task takes a lot of time to perform. We also can use near communication like NFC, RFID to transfer data automatically to sensor nodes. But the problem is cost of sensor node, it makes sensor nodes more expensive and wasteful, especially when we only use it once at the setup step. Therefore we need a setup method, can help end-users to make a secured Wireless Sensor Network easy and simply with low cost.

1.2 Challenges and Research Goals

We have to keep the cost of sensor nodes as cheap as possible, the size of sensor nodes as small as possible. We also need to apply it at home environment for regular users, need to help them make a WSN with a hundred sensor nodes. Therefore we propose some requirements for setup method as follows:

1. Applying with common hardware of sensor nodes to ensure cost of sensor nodes

2. Providing a friendly, intuitive interaction

3. Deploying numbers of sensor nodes as fast as possible

4. Providing a way to ensure privacy of WSN

In research environment such as laboratory and office, we have a rich equipment with some types of sensors and hardware. But in home environment, WSN is used by regular users, thus sensor nodes need to be cheap and small. And thus, our first goal is to propose a faster setup method with common sensor nodes.

Second problem of sensor nodes is their low performance CPU and limited power source. Because of this, setup method need to be as simple as possible. Third problem problem is about number of sensor nodes, with a hundred sensor nodes case, it takes a lot of time to setup with each sensor node method. Therefore, a method which can setup several sensor nodes at once is very useful. We can reduce the time to setup a larger WSN in geometric progression.

Last problem is data privacy, then we need a security key for encrypting and decrypting transfer data. But in the setup step, we do not have any security key, therefore we require a safe communication method without encrypt/decrypt data for exchanging security key.

We present a method called HUSTLE, which enables users to deploy sensor network based on light communication among smartphone and sensor nodes. HUSTLE is developed to use common hardware of sensor nodes (LED and light sensor) while providing a friendly and intuitive interaction to setup, reducing time to setup by multiple sensor nodes at once. HUSTLE also provides a way to maintain security of WSN.

1.3 Structure of Thesis

This thesis is organized as following. In chapter 2, we discuss background of deploying WSN problems, some security protocol for WSN and some communication techniques and

limited of each. We describe some related works in chapter 3. In chapter 4, we propose HUSTLE and explain how it works, and also explain light communication specifically, which is used in HUSTLE. Then we discuss the design and implementation of HUSTLE in chapter 5. Chapter 6 discusses about evaluation of HUSTLE, purpose, methodology, comparison targets and results. Finally, in chapter 7, we conclude this thesis and discuss about our future works.

Chapter 2

Problems of Deploying Secured Wireless Sensor Network

In this chapter, we introduce background of WSN and setup problems in order to deploy a secured WSN. Firstly, we describe the background of WSN and nowadays Smartphone. In the next section, we describe about some secure protocol, which can be used in a secured WSN. We introduce some applications of WSN and requirements of it in next section. In section 4, we also describe about the principle of setup a secured WSN and difficult point when applied at home environment. Finally, we discuss some communication techniques.

2.1 Background

In this section, we describe about hardware and properties of Sensor node and Smartphone.

2.1.1 Sensor Nodes

In recent years, we have some advance in wireless technologies, sensor technologies and microprocessing technologies. We can make a smaller and cheaper wireless communication part, sensor component and micro processing. And we combined them to make wireless sensor node devices, which is tiny and light. Which also can measure environment condition, communicate with others sensor nodes, and exchange data with a computer. Examples are mote [1], Sunspot [2] and upart [3] (Figure 2.1). With a lot of sensor nodes, we can make some useful applications in home environment such as Smart home, Secure home, Remote Medication. But we also have a challenge to apply it at home, regular users need a method to setup WSN, configure new sensor nodes and modify sensor nodes friendly, easy, and fast. Unlike computer or Smartphone, sensor node do not have friendly input method, this makes it difficult for users to setup sensor nodes and deploy a secured WSN.

Table 2.1 show that LED is common hardware of sensor nodes. We also have light sensor in some type of sensor nodes, another can use light sensor by only addition a sensor board. In addition, light sensor is used in a lot of applications such as environment monitoring, security system. Then we also can say that light sensor is a common hardware of sensor nodes.

2.1.2 Smartphones

Nowadays, Smartphone became a important device for everybody. We use it not only to contact with others but also to manage life style such as managing bank account, navigation. In the near feature, when WSN becomes popular and is used at home, Smartphone is also

Table 2.1: Sensor Node's hardware status

Type	LED	light sensor	accelerometer
Mote	O[a]	A[b]	A
CSIRO Fleck	A	O	A
Tmote	A	O	A
EYES	A	O	O
Sunspot	O	O	O
CPart	O	A	A
uPart	O	O	O
BTnode	O	A	A
PowWow	O	A	A
Preon32	O	A	A
Mulle	O	A	A
Waspmote	O	A	A
Zolertia	O	A	O
iSense	O	A	A

[a]Build in
[b]Need an addition sensor board

(a) Mote
58mm x 32mm x 7mm
runs with two AA batteries

(b) SunSpot
63.5mm x 38.1mm x 12.7mm
runs with LI-ION prismatic battery

(c) μpart
20mm x 17mm x 7mm
runs with one CR1632 battery.

Figure 2.1: Sensor node types

very useful to manage WSN such as monitoring status, adding new sensor node or modifying participating sensor node.

Start with Smartphone's flashlight, we made a survey about modern Smartphone's flashlight. To do this, we used GSMArena website [4] to collect information about all Smartphone and make Table 2.2. We can identify that, the number of Smartphone is being increased and the percent of Smartphone with flashlight also begin to increase. From 49.8% (2010) to 60.7% in 2012 year. We can say that flashlight is a common hardware of modern Smartphone.

Table 2.2: Smartphone's flashlight status

Time (Year)	All	Smartphone with flashlight	Percentage
In 2010	223	252	49.8%
In 2011	445	252	56.6%
In 2012	438	266	60.7%

2.2 Secure protocol in WSN

Simplicio et. al. [5] showed that there are some secure schemes for encrypting-decrypting data in WSN.

2.2.1 Network-wide keys scheme

Network-wide keys scheme is the most straightforward key distribution. In this scheme, we have a single master key, which is loaded into all sensor nodes. With this, we have a high level of efficiency and flexibility, requiring minimal memory for the storage of keys, and do not depend on the size of the network. With master key, this scheme allows adding any number of sensor nodes after the initial deployment with. Furthermore, because all of sensor nodes have the same master key, this scheme provides perfect key connectivition.

There are more papers discussing about this scheme: *Broadcast Session Key Negotiation*

Protocol (BROSK) [6], *Symmetric-Key Key Establishment (SKKE)* [7], *Loop-Based Key Management Scheme (LBKMS)* [8] and *Lightweight Key Management System (LKMS)* [9].

Despite the numerous advantages, the Network-Wide-Key approach has serious security vulnerabilities: an attacker can capture of a single node, this would disclose the common key, compromising all the nodes in the network and their communications.

2.2.2 The full pairwise scheme

In previous scheme, we only used a single key for the communication between all sensor nodes. This can provide perfect simplest key connection but it has some problems. When the master key is out, the security of the network will be destroyed. Therefore Full Pairwise scheme is proposed. This scheme adopts the extreme opposite approach. In this case, each of the N nodes in the network receives $N-1$ pairwise keys to communicate with every other node. This scheme provides higher security level, providing features such as node-to-node authentication and perfect resilience, which thwarts node replication attacks.

However the main drawback of this scheme is the great memory overhead it introduces, because each node have to store many keys and many key in there may never be used.

2.2.3 Probabilistic approaches

Some paper [10] [11] are proposing probabilistic approaches. In there, each node receives a group of keys, which is called *key chain*, whose size is much lower than the size of the network itself. This provides a good key connectivity and also avoid both the memory overhead in the *Full Pairwise* scheme and the security risk of a single master key. In this approach, we have three distinct and sequential phases:

1. Key pre-distribution: the Key Distribution Center chooses each sensor node's *key chain* from a large pool of keys P, These chains are then loaded into the sensor nodes prior to

deployment. We usually receive a unique ID of each key in the pool, for identification.

2. Share-key discovery: After deployment, the sensor nodes try to discover who their neighbors are and which keys they have in common. When two nodes established a shared key, we say that there is a *direct link* between them.

3. Path-key establishment: Because the key management scheme employed can't provide perfect key-connectivity with a small size *key chain*, some neighboring nodes may not have keys in common. When nodes A and B need to establish a secure communication, they have to find an intermediary node C, that shares a common key with both A and B. Node C can then act as a mediator for the messages exchanged between A and B or, in order to avoid this extra communication overhead, C can create and distribute a new key to be used by A and B. Then A and B shares the new key and establish direct-link.

2.2.4 Others schemes

We also have some others schemes such as *Matrix-based schemes* [12], *Polynomial-based schemes* [13], and *Combinatorial designs* [14].

All of mentioned schemes have a same feature, we need an initial information in new sensor nodes. In *Network-wide keys scheme*, this is only a master key. With *The full pairwise scheme* it is a key table of all sensor nodes of the network. In *Probabilistic approaches*, it is a list of *key chain* and in *Matrix-based schemes* it is a matrix to calculate pairwise key.

2.3 Applications of WSN and requirements

There are some applications of WSN in home environment:

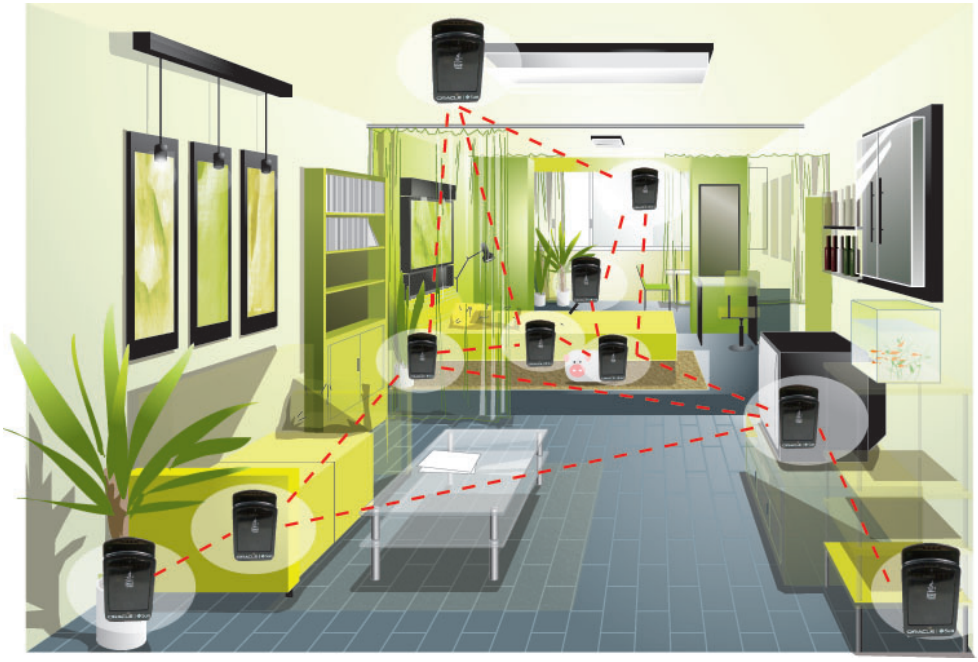


Figure 2.2: WSN application: Smarthome system

- Smart-home is also called home automation. It is the automation of the home, housework or home-activity. In smart-home everything are controllable not only from Smartphone but from itself. With WSN, every sensor node can communicate with sink node or other sensor nodes over wireless communication then we can put sensor nodes at anywhere at home. We can use it to collect data, control other devices, and make a smart-home (Figure 2.2).

- Home security also is a application in range of smart-home. With the properties "can put sensor node anywhere" of WSN, it is very useful to monitor a large area.

- Wearable health monitoring is a application of WSN. With properties of WSN is tiny, number of devices, we can make wearable health monitoring system(Figure 2.3). The system allow an individual to closely monitor changes in patient vital signs and provide feedback to help maintain an optimal health status. If integrated in to a tele medical

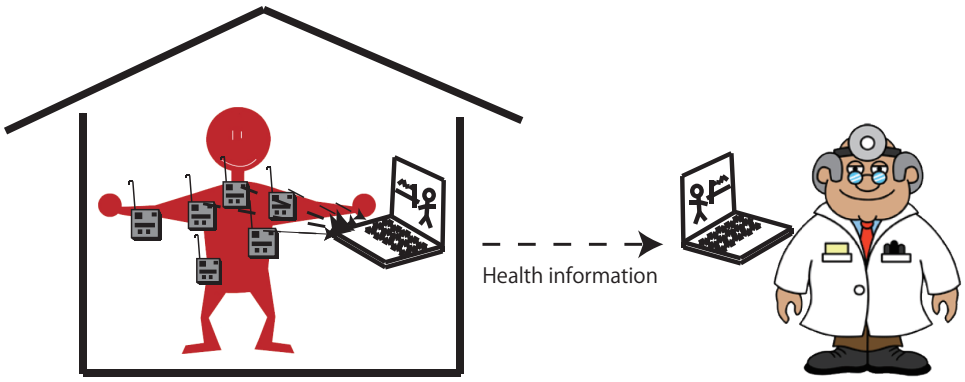


Figure 2.3: WSN application: Wearable health monitoring system

system, these systems can even alert medical personnel when life-threatening changes occur. In addition, patients can benefit from continuos long-term monitoring as a part of a diagnostic procedure. It can achieve optimal maintenance of a chronic condition, or can be supervised during recovery from an acute event or surgical procedure. Long-term health monitoring can capture the diurnal and circadian variation in physiological signals. These variations, for example, are a very good recovery indicator in cardiac patients after myocardial infarction. In addition, long-term monitoring can confirm adherence to treatment guidelines (e.g., regular cardiovascular exercise) or help monitor effects of drug therapy. Other patients can also benefit from these systems; for example, the monitors can be used during physical rehabilitation after hip or knee surgeries, stroke rehabilitation, or brain trauma rehabilitation.

- Children supervise is another application in range of WSN (Figure 2.4). Like with health monitoring, we can use WSN for monitoring health of children like raising, alarm in case of any danger or tracking position of children.

All of home environment applications have similar requirements. Firstly, because it is applied at home environment where there is a lot of personal information, then we need data

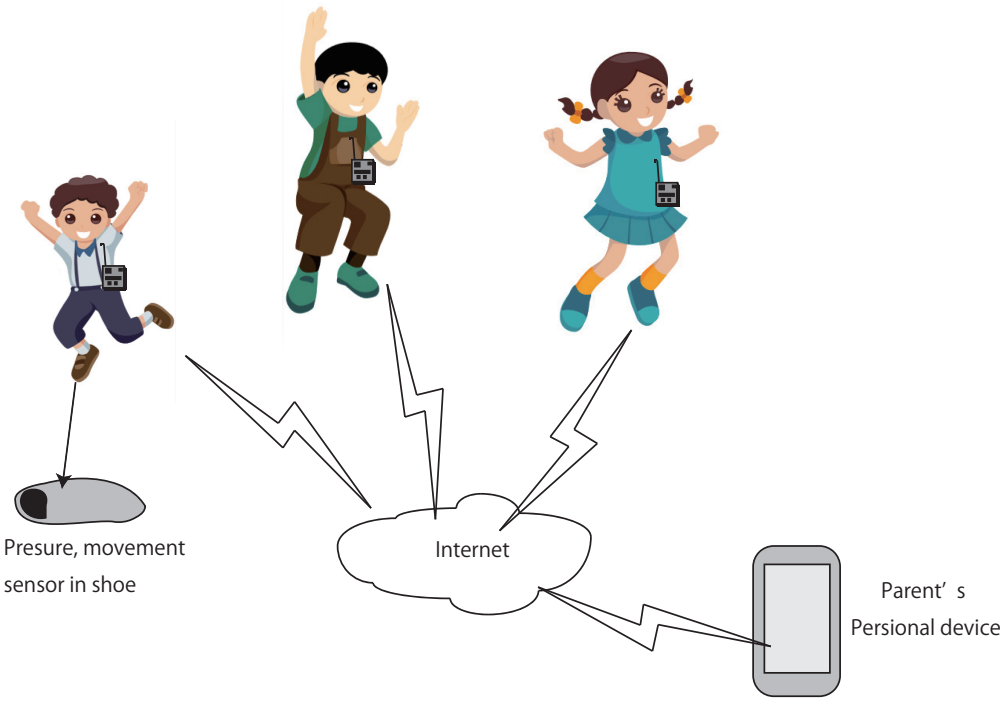


Figure 2.4: WSN application: Children supervise system

privacy, for example in security application or health monitoring application. Secondly, in the home environment, they need as simple as possible WSN, to setup, manage and use. Lastly, for regular users, the cost of the system is also important, and we need a cheap WSN for making it more popular.

2.4 How to deploy a secured WSN

In previous section, we described about some applications in the home environment and requirements of it. We also described some secure protocols in range of WSN. In this section, we discuss about WSN management, how to deploy a secured WSN and modify it after that.

Firstly, with data privacy requirement, we must utilize secure protocol for WSN, this is described in [5] [6] [7] [8] [9] [10] [11] [12] [13] [14]. We use it for encrypt and decrypt data exchanged between devices to ensure that all of the data is secured. But we have a problem

when we configure new sensor nodes, how to transfer WSN information (identify, secure protocol, routing) and setup it securely. In research environment, we can pre-install all of sensor nodes by connecting them with a computer and deploy. But in home environment this is difficult to use. For this reason, instead of transferring all of data, we only need to pre-install sensor nodes with a private security key and input this security key to sink node by one device such as computer, then sink node uses this key to encrypt initial data and send it to new sensor nodes over wireless communication.

But, we still have a problem with private security key, especially when inputing the key of each sensor node. For security reason, all of key have to be different form each other. Security key are randomly generated with a lot of special characters and it's difficult to input by keyboard. Especially when there are a lot of sensor nodes. On the other hand, sensor nodes uses battery and sometime them is used in door, therefore we need to change battery of them or replace disabled sensor nodes with new sensor nodes then we need re-setup it more one time. Therefore we need a method that can help users configure new sensor nodes to make a secured WSN by themselves, more simply, fast and maintain secure of WSN.

In the next, for reinstalling a sensor node simply, it is reseted and deployed with new information similar new sensor nodes case.

2.5 Communication techniques in current ICT

As same as mentioned in section 2.4, we have a problem with input method of security key, and to solve this we can use near communication like NFC, RFID, Bluetooth or Infrared. Which can be used to send data in small range securely without encrypt data.

Infrared [15]

The infrared light has a short range ($<$ 30 meters). Finally, the infrared technology is relatively old, consequently the infrared transceivers are cheap compared to the other

technologies. The infrared transmissions should also be intense in order to not be confused with other light sources like television, window, light bulbs, etc. Considering the risk of light confusion, this communication protocol has not been retained.

Bluetooth [16]

Bluetooth is a short range communication technology (< 10 meters), but this range can easily be extended with a power booster. This communication protocol is considered as low power consumption and is not according to our requirements. It requires pair step, this also need some code then it makes some complexes to use. And it usually is not used in WSN.

NFC&RFID [17]

Near field communication (NFC) is a new technique, which is a set of standards for Smartphones and similar devices to establish radio communication with each other by touching them together or bringing them into close proximity, usually no more than a few centimeters. Present and anticipated applications include contactless transactions, data exchange, and simplified setup of more complex communications such as Wi-Fi. Communication is also possible between an NFC device and an unpowered NFC chip, called a "tag". But it still do not become popular, only have some devices support it.

2.6 Summary

In this chapter, we discuss about background of WSN, Smartphone and some applications of WSN. We also discuss about some requirements of each application and problem when deploying a secured WSN. On top of that, we discuss some communication techniques which can be used to solve inputing security key problems.

Chapter 3

Related Work

In the previous chapter, we described some communication techniques, but these are some problems when apply at sensor node, small devices. With this reason we have some researches about near communication techniques depend on properties and hardware of sensor nodes, which are described in the first section of this chapter. In the next section, we also describe some deploying methods for both secured WSN and unsecured WSN.

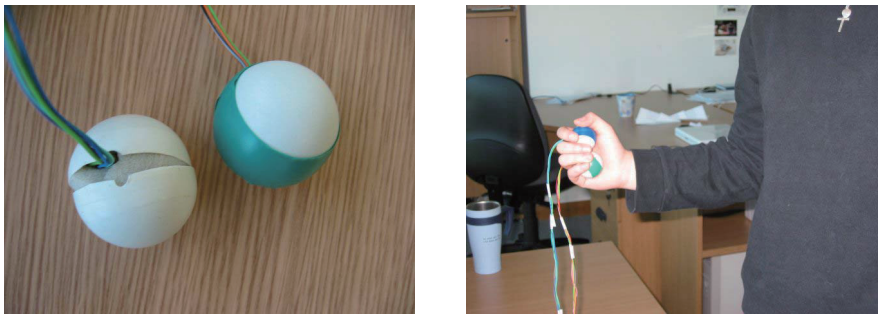


Figure 3.1: Shake Well Before Use: devices with accelerometers

3.1 Near communication

In this section, we discuss communication between devices which can be applied with only common hardwares of sensor nodes. We have two methods that use object movement for exchanging data. Shake Well Before Use [18] is an example for authenticating device pairings with shaking patterns. In fact, with this method, we can not send data between two devices, but this can help us make a shared key between two devices by only shaking it at the same time like Figure 3.1. Then we have same accelerometer data from two devices and can make a shared key from it. We can use the key for encrypting and decrypting data, which is sent over some communication techniques such as wireless communication. However, the problem of this method is speed and feedback. It only can make 7 bits shared key per second. This will take a lot of time to make a longer shared key such as 64bit security key. Another problem is about feedback step, with shaking action, users find it difficult to know the status of devices and it make difficult to know when making step finishes and when shared key is created successfully or not.

Vib-Connect [19] is also one type of near communication with only common hardwares, accelerometer for receiving data. On the other hand, according to the previous method, we can send data with this method. We use the smartphone's vibrate motor to send the vibrate pattern to other smartphones or laptops. With an accelerometer at receiver, we can

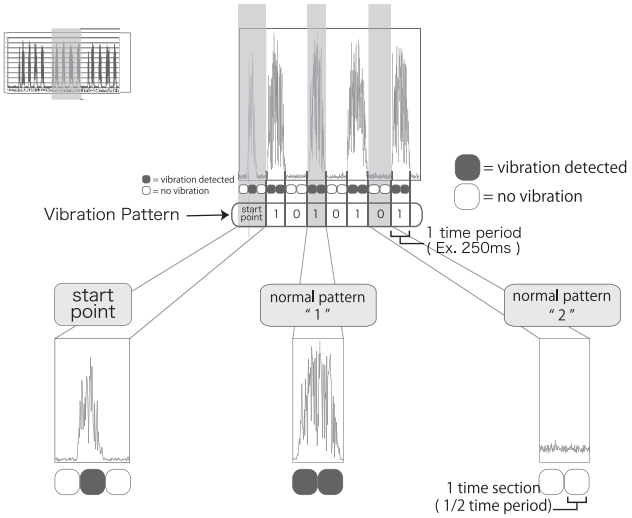


Figure 3.2: Vib-Connect: Vibration Patterns

sense the change of vibration and decode vibrate pattern like Figure 3.2. However, there is a problem with this method, which needs a quiet environment and have a low speed at about 10 bits data per second.

3.2 WSN deploying methods

In this section we discuss some WSN deploying methods. Basing on technique that is applied, WSN deploying methods can be classified into three groups: 1)Pre-Install new sensor nodes, 2)Exchanging static information, 3)Exchanging dynamic information.

3.2.1 Pre-Install new sensor nodes

In this method, WSN identification and other information are deployed to new sensor nodes. After that, a predefined identifier is used to decide on the communication of new sensor nodes to an exiting WSN. Only sensor node with same WSN information will be a member of the same WSN.

Luster [20] is an example of this method. Inspire of the problem of this method is about configuring process, this method requires to have certain skills, need to do with a computer

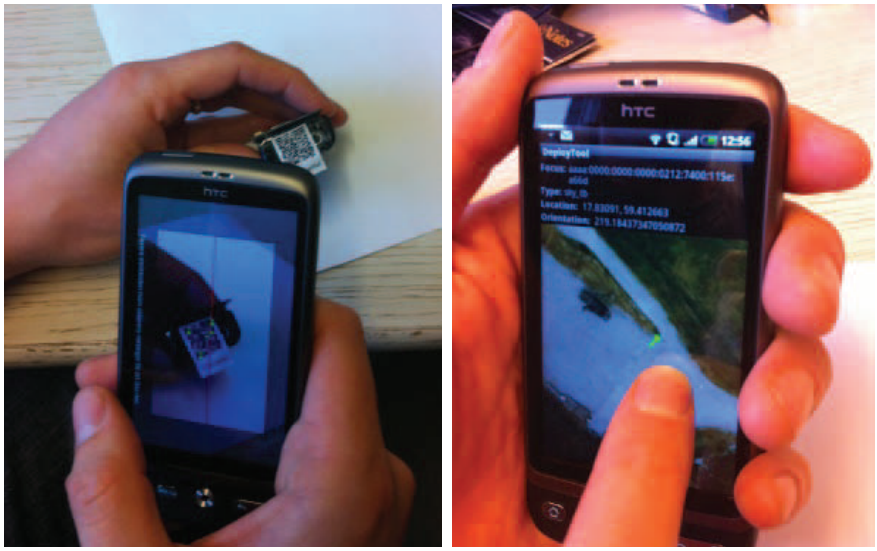


Figure 3.3: Snap: Node identification

with limited mobility. Besides, it is also difficult to reconfigure it again for use in a different WSN. Thus, this method cannot meet our research's requirements.

3.2.2 Exchanging static information

This method is used to make an unsecured WSN. Identification of each sensor node is put on it such as QR-code, barcode. The identification is used to determine what sensor nodes will be added.

Snap [21] is a research in range of this type. Camera of smartphone is used to read QR-Code of each sensor node, so we can identify exactly sensor node which should be added (Figure 3.3). However, there is not any safe data from QR-Code to use as security key, so data could not be sent safely in adding process. As the result, this method cannot meet security requirement.

Another research can be found in Some Assembly Required [22]. By using a sensor network kit called Home Energy Tutor and installing the application, users only have to place various sensor nodes and scan barcodes on the sensor and in a printed catalog (Figure 3.4). This

Figure 3.4: Sensor association in the Home Energy Tutor.

creates an association between sensor node and a room or appliance. However, the problem is that with only barcodes there is no safe data to use as a security key, especially when the device is placed outdoors. Therefore, setting up data could not be sent safely. At one-time only one sensor node is added, consequently this method can not meet setting up time requirement.

3.2.3 Exchanging dynamic information

In this method, data is exchanged between sensor node and other devices such as identification, security key. Therefore, this can be used to make a secured WSN. In oder to exchange data securely, near communication is used such as RFID, NFC, infrared communication or light communication.

The first research in this method is Reliable set-up of medical body-sensor networks [23]. IrDA (Infrared Data Association) is used to transfer unique code to identify the exact sensor node for adding to WSN. However it may cause some problems about price of sensor nodes, these problems make WSN more expensive. With IrDA, sensor node requires having correlative hardware which is used only one-time in the installation step. This makes the sensor node more expensive and wasteful, especially with the existence of cheap sensor nodes. This method also needs a special device, which is used as a connector of WSN.

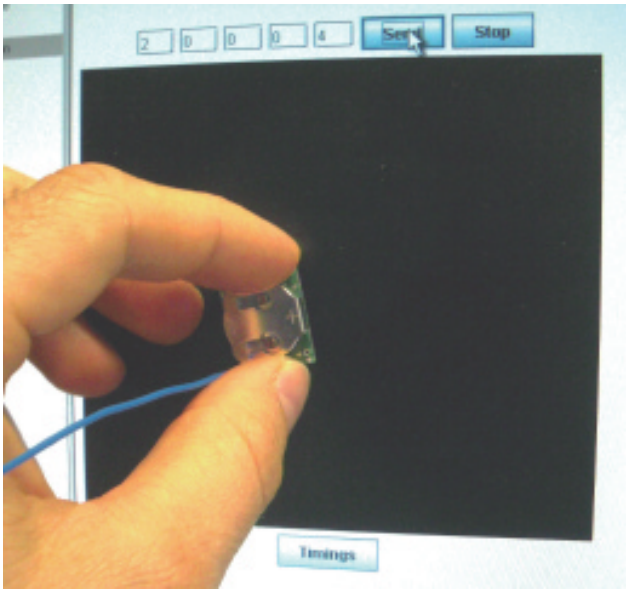


Figure 3.5: uPart configuration using a PC monitor

The uPart experience [24] is also one research in range of this method. The light sensor is used as a receiving data channel to configure the uPart sensor node (Figure 3.5). In fact, it is only sensor reading cycle, compression values and what values should be transferred, and same with previous related research there is no safe data to use as security key, then we cannot keep the security of WSN.

3.3 Summary

To meet the requirements of deploying a WSN with low cost, we have some researches about near communication with common hardwares such as [18] and [19] with accelerometer. There are some methods for deploying a WSN become simpler and faster: Pre-Install new sensor nodes (Luster [20]), Exchanging static information (Snap [21], Some assembly required [22]), Exchanging dynamic information (Reliable set-up of medical body-sensor networks [23], [24]). However some researches cannot make a secured WSN [21] [22] [24], [23] need some special hardwares and others [20] do not provide regular users with a friendly interaction. Finally

they do not meet all of the requirements in our research.

Chapter 4

HUSTLE

This chapter introduces HUSTLE. Firstly we discuss about advantages and disadvantages of communication techniques, and explain why we use light communication in HUSTLE. Secondly, we explain the concept of HUSTLE with hardware and software requirements. Thirdly, we describe data flow and protocol of HUSTLE. In the next, we discuss about architecture of HUSTLE. Finally, we propose a method to encode and decode light pattern in home environment with specific properties of common smartphon and, sensor node with indoor lighting conditions.

4.1 Approach to a near communication technique

In the Background chapter, we shown that LED and light sensor are common hardware of sensor nodes. We have also known that smartphones with camera flashlight are increasing, and flashlight also becomes common hardware of Smartphone. With flashlight, LED, and light sensor we can implement light communication.

In the Background chapter, we also described some near communication techniques such as Bluetooth, Infrared, NFC and RFID. In Related Works chapter, we also described two method: *Shake Well Before Use: Authentication Based on Accelerometer Data [18]* and *Vib-Connect: A Device Collaboration Interface Using Vibration [19].*

We analyze advantages and disadvantages of all methods in Table 4.1. As mentioned in the first chapter (Introduction chapter), four requirements that HUSTLE has to meet are: (1) it can be applied with common hardware of sensor nodes to ensure cost of sensor nodes, (2) it provides a friendly, intuitive interaction, (3) it can set-up numbers sensor nodes as fast as possible and (4) it provides a way to ensure privacy of WSN. We examined exiting methods and techniques base on these requirements in Table 4.2

However Infrared, Bluetooth, NFC, Shake, Vib-connect can not meet our research requirements. Meanwhile light communication can meet all of our research requirements, therefore we explore the use of light communication in HUSTLE.

4.2 Hardware and Software requirements

In HUSTLE, we need some hardware like Figure 4.1. We have smartphone with an accessory, sensor node, sink node and a computer. To implement HUSTLE, we need some requirements for hardware and WSN. There are:

Table 4.1: Advantages and disadvantages of communication techniques and methods

Technique	Advantages	Disadvantages
Infrared	Cheap. Invisible. Secure in short range.	Not a common hardware among sensor nodes and personal devices
Bluetooth	Invisible. Secure in short range.	Need pair step. Not a common hardware of sensor nodes
NFC, RFID	Invisible. Secure in short range.	Not a common hardware among sensor nodes among personal devices
Shake	Invisible, secure	Very low speed, hard to idenfify whenever it is successful or not
Vib-Connect	Invisible, Secure in short range	Low speed, need a quiet environment
Light communication	Work perfect with indoor condition, secure in short range	Low speed

Table 4.2: Communication techniques, methods and our research requirements

Technique	(1) Common hardware	(2) Friendly	(3) Faster	(4) Privacy
Infrared	X	O	O	O
Bluetooth	X	X (need pair)	X(one time only one)	O
NFC, RFID	X	O	O	O
Shake	O	X(difficult to feedback)	X(very low speed)	O
Vib-Connect	O	X(need quite environment)	X (Low speed)	O
Light communication	O	O	O(multi-sensor nodes)	O

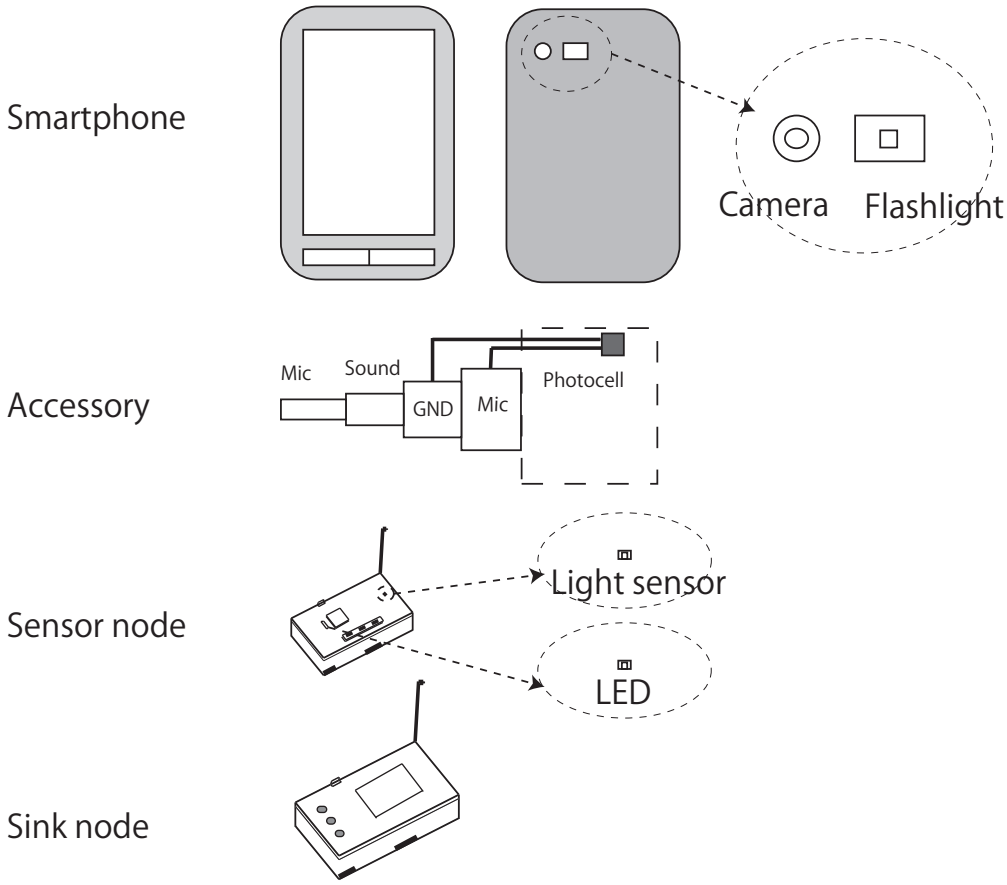


Figure 4.1: HUSTLE: Hardware

- With Smartphone, we need a camera's flashlight, which can controlled by programming and also can process signal from microphone line.

- With sensor node, we need an LED or a light sensor. We can control LED by programming with fast frequency, about some hundred times per second. Light sensor also needs to be get brightness value by programming with fast frequency, about some thousand times per second.

- With WSN, we need a secured WSN with secure protocol. It also has an unique ID to distinct with another. About routing protocol, sink node can transfer a message to all of sensor nodes, also broadcast a message to new sensor nodes.

4.3 HUSTLE's Concept

In HUSTLE, when sensor nodes are started, it activates HUSTLE in order to add sensor nodes to WSN. After HUSTLE is finished, the main application will be run. HUSTLE is discussed in below.

With a smartphone like mentioned requirements, we can use a flashlight to blink light patterns. If the sensor node has a light sensor, it can use the sensor to receive the light pattern. Otherwise, in case it has an LED, we use the LED to send data to the Smartphone. We can connect the Smartphone with an accessory like Figure 4.2 that can receive light patterns from sensor nodes and we also use screen of it to show the status of this process. With this, we can provide two setup interactions like Figure 4.2 and Figure 4.3. When users want to add some new sensor nodes, the first user must log-in to sever with sever address and security key, after that users can process like Figure 4.2 or Figure 4.3 according hardware of sensor nodes, have LED or light sensor (in case it have both LED and light sensor, light sensor will be used).

- Have an LED case (Figure 4.2): In this case, user turn on sensor nodes, touch it with accessory at Smartphone. After that, sensor node uses LED to send one-time security key to Smartphone automatically. With this, we can encrypt-decrypt setup data, which is sent over wireless communication later such as WSN information, secure protocol information or something else.

- Have a light sensor case (Figure 4.3): In this case, user turn on all of new sensor nodes, put it on a table example, direct Smartphone's flashlight to it and push send button. After that, flashlight is used to send a one-time security key to all of sensor nodes automatically. We use the key for encrypting-decrypting data like previous case.

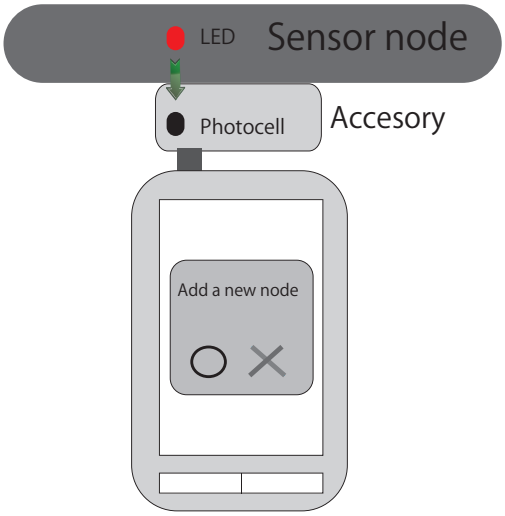


Figure 4.2: Touch Interaction: Touching an Accessory of Smartphone

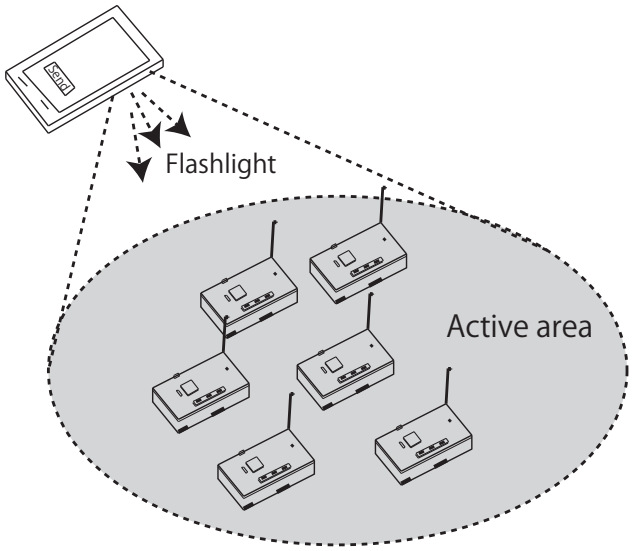


Figure 4.3: Blink Interaction: Directing Smartphone's flashlight to all new sensor nodes

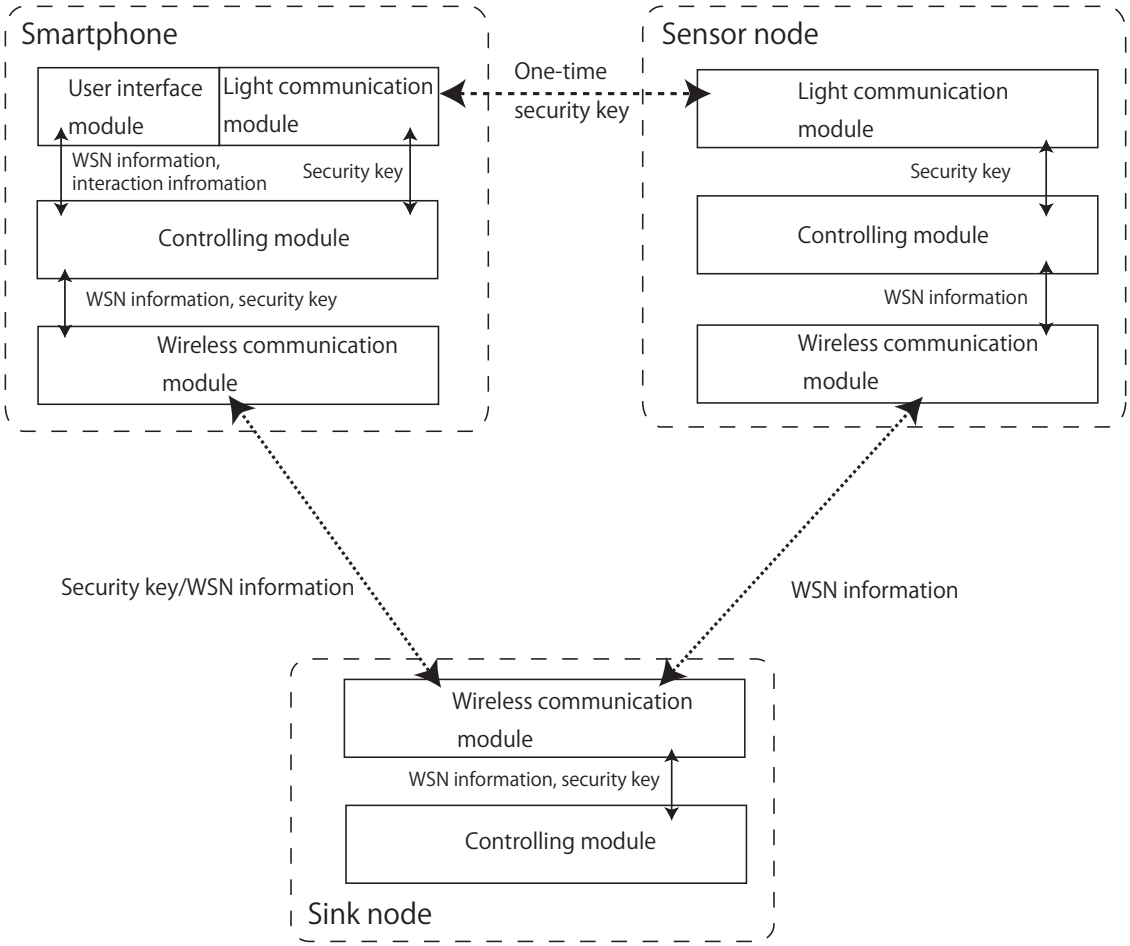


Figure 4.4: Architecture of HUSTLE

4.4 Architecture of HUSTLE

The architecture of HUSTLE is shown in this section. We have user interface module, wireless communication module, controlling module and light communication module. The connection of each module and each part Sensor Node, Smartphone and Sink Node is shown in Figure 4.4. Smartphone and Sensor Node is connected over light communication module, which is used to exchange one-time security key and identifier. Exchanging data between Smartphone and Sink Node such as one-time security key, added sensor nodes information is exchanged over wireless communication, and this data is encrypted by fixed security key

of user. With this reason we can ensure that data is exchanged safely. Between Sink Node and Sensor Node, in adding process data such as network information, secure protocol information is exchanged by wireless communication and is encrypted by using one-time security key which is exchanged by light communication module beforehand. When sensor nodes are added to network, the data is encrypted by secure protocol of network.

4.5 HUSTLE Protocol

In this section, we present about adding new sensor nodes protocol. Depending on hardware of sensor nodes we divide it in to two cases, LED case and light sensor case.

4.5.1 New sensor node with LED

As shown in Figure 4.5, in HUSTLE, the first step (1) is when users turn on the sensor node, then deployed HUSTLE program will be activated. At same time, we touch LED part of the sensor node with the accessory of the smartphone. After that, in (2.1), we use LED of the sensor node to send identification and one-time security key to the smartphone. The smartphone, when receiving data from step (2.1) also sends the information to the sink node over wireless communication in step (2.2). In step (2.1) because flashlight is only active in a small area and with it's properties is isotropic, then we can send data safely in the privacy area such as at home or in a room. In (2.2), exchanging data is encrypted by user's security key, then we also can send data safely in step (2).

With the received security key in step (2.2), the sink node encrypts setting-up data (WSN information, routing information and secure information, etc.) in step (3).

In step (4), the sink node broadcasts encrypted setting-up data, and to enlarge the broadcast area, all of added sensor nodes also broadcast this encrypted setting-up data (Figure 4.6). In this, a message from the sink node to others added sensor nodes is encrypted by

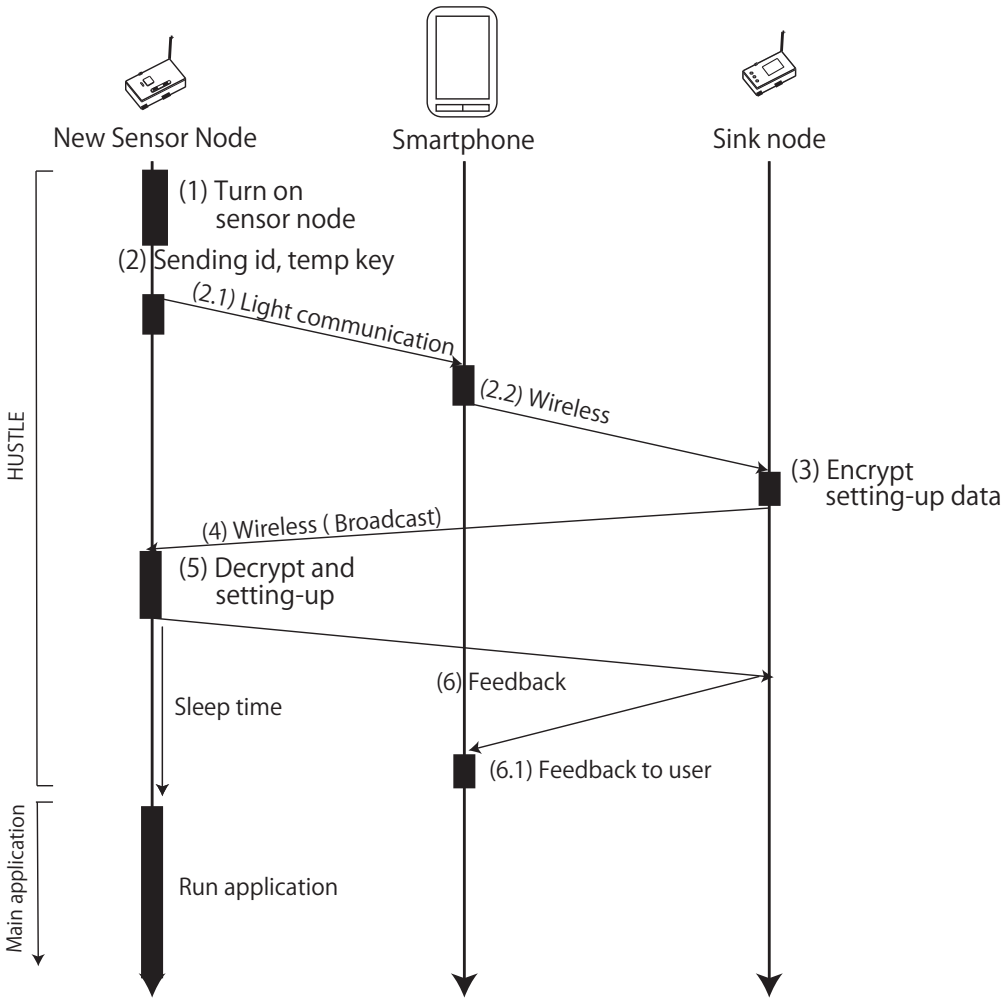


Figure 4.5: HUSTLE Protocol: Add a new sensor node with LED

secure protocol, message broadcast from the sink node, added sensor nodes to new sensor nodes is encrypted by the security key. At the new sensor nodes, received the security key in step (2.1) is used to decrypt and setting-up with the decrypted data. Therefore, we can maintain securely of WSN. After this step, new sensor node becomes a part of WSN.

In the next step, to feedback to users about setup process, just added sensor nodes sends a feedback message to the sink node and the sink node forwards this message to the smartphone in step (6). And step (6.1), the smartphone's screen is used for feedback to users about deploying process like what sensor nodes were just added. Then main application is initiated.

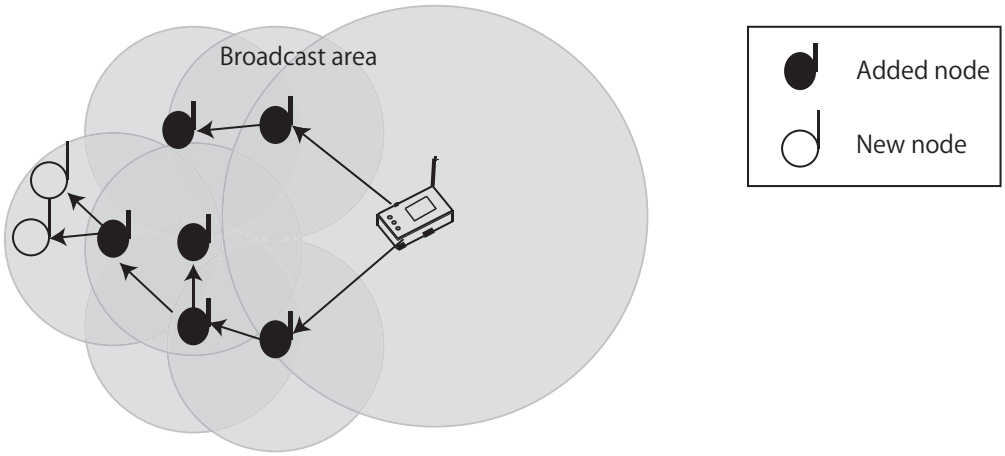


Figure 4.6: HUSTLE Protocol: Broadcast setting-up message to new sensor nodes

4.5.2 New sensor nodes with light sensor

Protocol schema of this case is shown in Figure 4.7. In HUSTLE, in step (1) user turns on all of new sensor nodes manually. After finishing step (1), the user uses the smartphone with HUSTLE program at step (2). In step (2), after users pushes "Send" button, there are two small processes (2.1) sending WSN identification and one-time security key by light communication and (2.2) sending this security key by wireless communication. In (2.1), we use the smartphone's flashlight to send WSN identification and security key to new sensor nodes that should be added to. When (2.1) finished, new sensor nodes save the identification and the security key in step (2.1.a), the smartphone also sends the security key to the sink node over the Internet (Wi-Fi connection or Mobile connection) in step (2.2).

With the received security key in step (2.2), the sink node encrypts setting-up data (WSN information, routing information and secure information, etc.) in step (3).

In step (4), the sink node broadcasts encrypted setting-up data, and to enlarge the broadcast area, all of added sensor nodes also broadcast this encrypted setting-up data (Figure 4.6). In this, a message from the sink node to others added sensor nodes is encrypted by

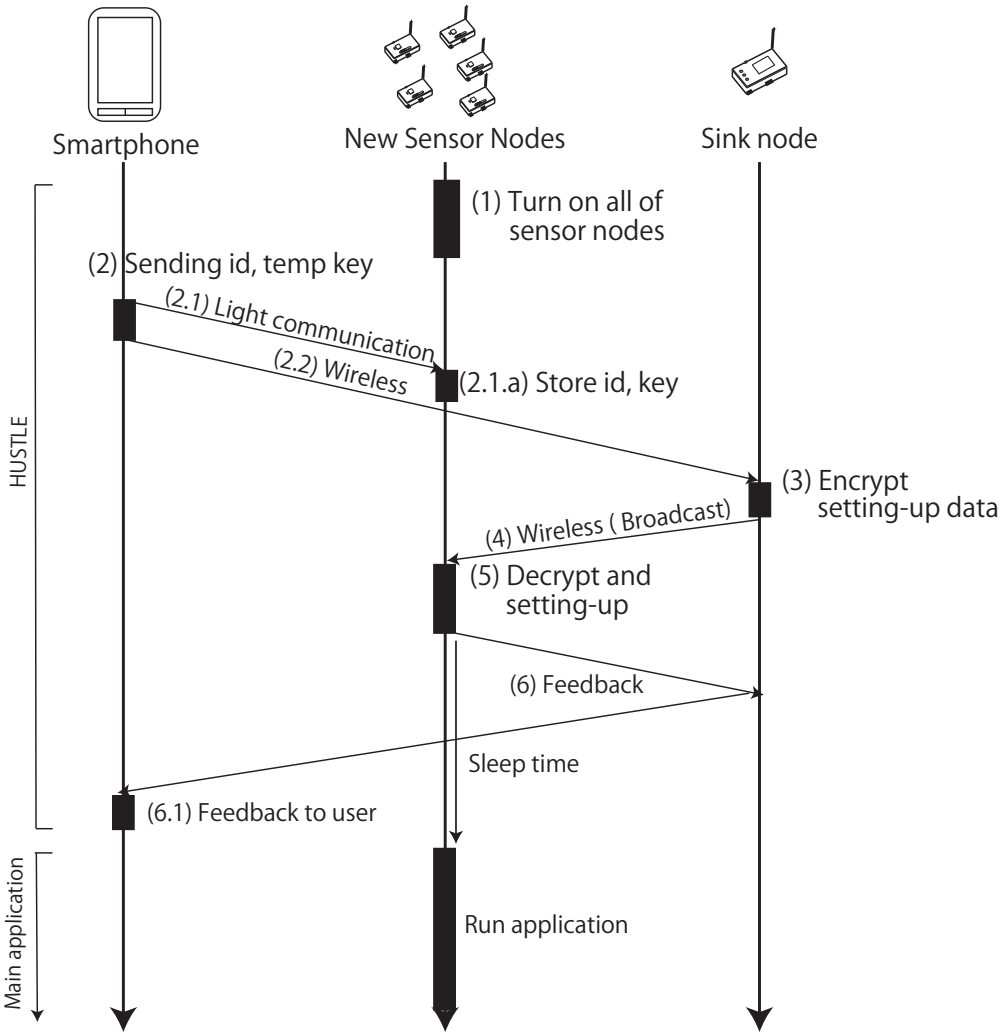
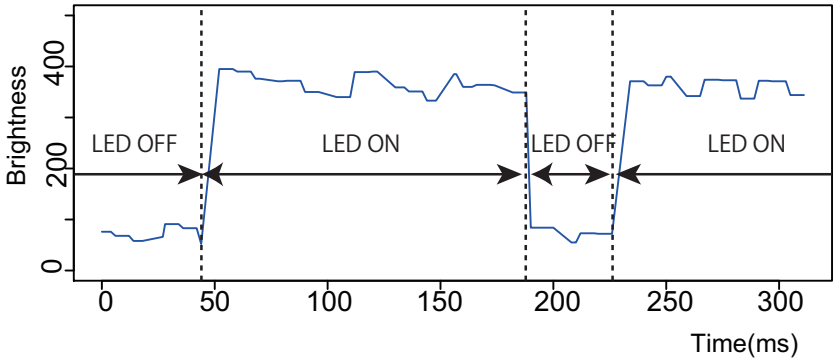


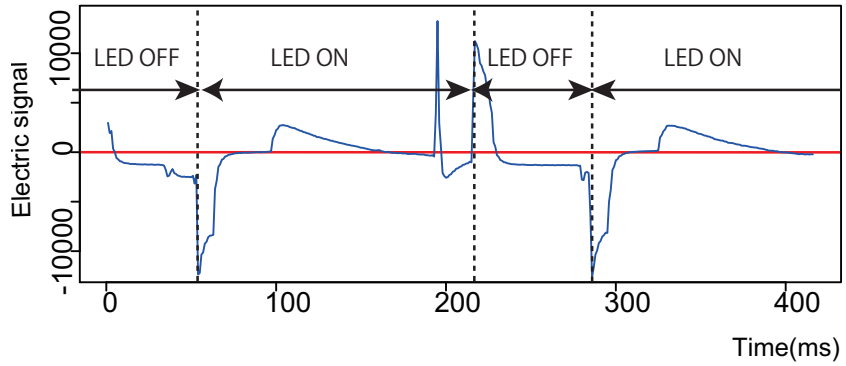
Figure 4.7: HUSTLE Protocol: Add new sensor nodes with light sensor

secure protocol, broadcasted message from the sink node, added sensor nodes to new sensor nodes is encrypted by the security key. At the new sensor nodes, it can use received security key in step (2.1) to decrypt and setting-up with the decrypted data. Therefore, we can maintain security of WSN. After this step, new sensor nodes become a part of WSN.

Like previous case, in order to feedback to user about setting-up process, just added sensor nodes send a feedback message to the sink node and it forwards this message to the smartphone in step (6). And step (6.1), the smartphone screen is used to feedback to user



Signal from light sensor



Signal from microphone line by photocell

Figure 4.8: Light communication: Signal from light sensor and photocell

about setting-up process, what sensor nodes just added (by sensor node ID, type of sensor node). Then main application is initiated.

4.6 Light communication in HUSTLE

In this section, we describe about principles of light communication between Smartphone and Sensor Node. We have two types of light communication like Figure 4.2 and 4.3. For receiving light signal, we have different hardware, light sensor of sensor node or photocell of accessory. With light sensor of sensor node, we can get light signal directly by reading

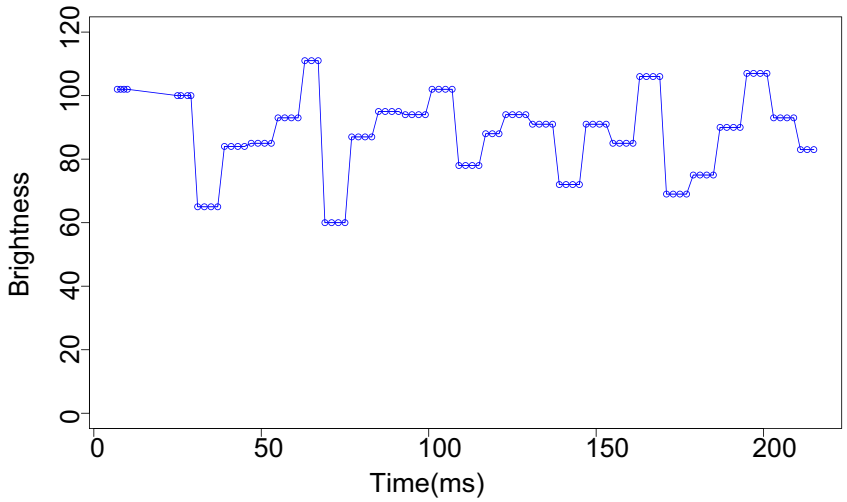


Figure 4.9: Light communication: Environment light

status of light sensor but with photocell of accessory (Figure 4.10), we only get the change of light signal, similar to derivative of light signal (the difference of two type signal is shown in Figure 4.8). Therefore we also present two ways to decode with light sensor and photocell.

4.6.1 Light communication by Smartphone's flashlight and light sensor

When we direct flashlight to sensor node, sensor node's light sensor can sense the change of flashlight. We can detect whenever flashlight-blink is started or finished and calculate length of blink period. Unlike *LED and photocell* case (Figure 4.2), in this case besides light signal from flashlight, we also have light signal from in-door light such as fluorescent lighting or incandescent lamp. On the other hand, in home environment all of electric goods use alternating current (AC) power, then light signal form in-door light is also changed with very fast periodically. As a result, we have strong noise from environment's like Figure 4.9. In addition, we have a big change of brightness when changing environment such as form fluorescent lighting to incandescent lamp, or move to another room. With this reason,

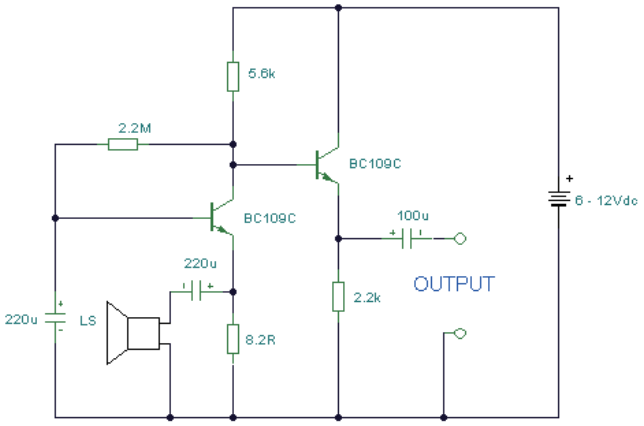


Figure 4.10: Microphone circuit

it is difficult to detect light pattern and need a method to reduce the noise and calculate threshold dynamically. To solve noise problem, we propose a method to calculate threshold dynamically, which will be described in *Decoding Light Pattern* part.

4.6.2 Light communication by LED and Photocell

We have a simple circuit of microphone in Figure 4.10. We sense sound of environment by measuring the change of resistance of microphone. When the sound changes, the resistance of microphone is also changed correlatively. Therefore we can get sound signal from OUPUT gate.

Photocell also has same properties with microphone, its resistance is changed correlatively with change of brightness like Figure 4.11. Therefore we can connect photocell with microphone line like Figure 4.1 and measure the change of brightness by signal from microphone line. From this signal we can detect start and end of LED blink like Figure 4.12 and calculate length of blink period.

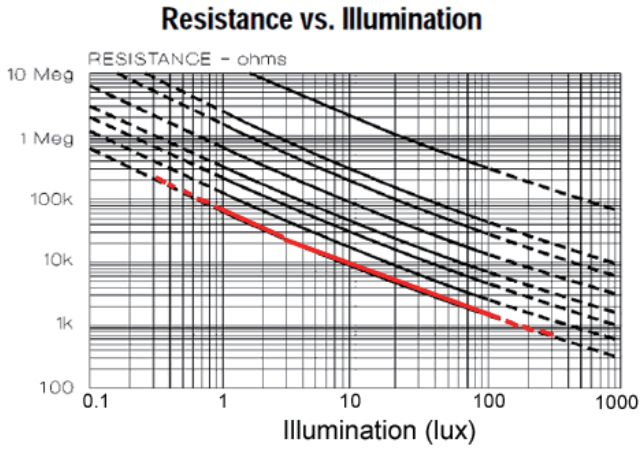


Figure 4.11: Photocell resistance and brightness.

4.6.3 Encoding Light Pattern

As mentioned, we need to send data with LED and Smartphone's flashlight, therefore we must encode data to light pattern. And for decode process, we need to know when data is sent therefore we add START flag at begin of data and FINISH flag at end of data. With this data, we represent data by getting binary of data and insert light pattern according 0 bit or 1 bit.

About light pattern, because we only can change state of flashlight form ON to OFF or OFF to ON, we represent START flag, FINISH flag, bit 0, bit 1 by only ON or OFF time length. In addition, we tested performance of Smartphone's flashlight by turning ON and turning OFF flashlight in a period time and measure this change by sensor node's light sensor. The result of flashlight is shown at Figure 4.13 and Figure 4.14. In Figure 4.13, we show the range of receiving period when sending a period. In Figure 4.14 we show *error range* of receiving period with sending period, in there *error range* is calculated by $Er = (Tr_{max} - Tr_{min})/T$ (with T is sent period time length, Tr_{max} is max time length of receiving period, Tr_{min} is min time length of receiving period).

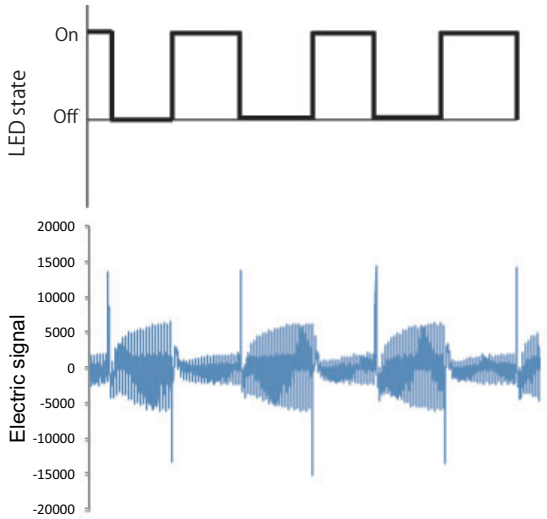


Figure 4.12: Brightness state and microphone signal.

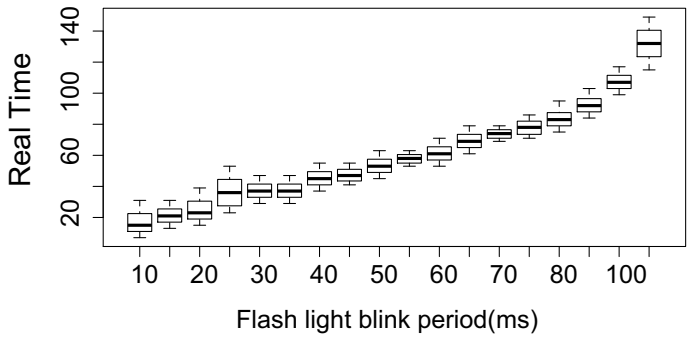


Figure 4.13: Flashlight performance: Sending period and receiving period

From this result, we know that flashlight performance is not good and it is difficult to send and receive too short period. Example can find at $T = 10ms$, we received time length from $7ms$ to $32ms$ and *error range* is 240%. In addition, we also difficult to distinguish one period form a list of period like 20ms from [10,20,30,40]ms. With this reason, instead we make light pattern based on state of light ON or OFF (that mean we have base time period T, turn on light in $n * N$ represent n bit 1 and turn off light in $n * N$ represent n bit 0), we make light pattern base on time length of blink-light like Figure 4.15. In there, white period

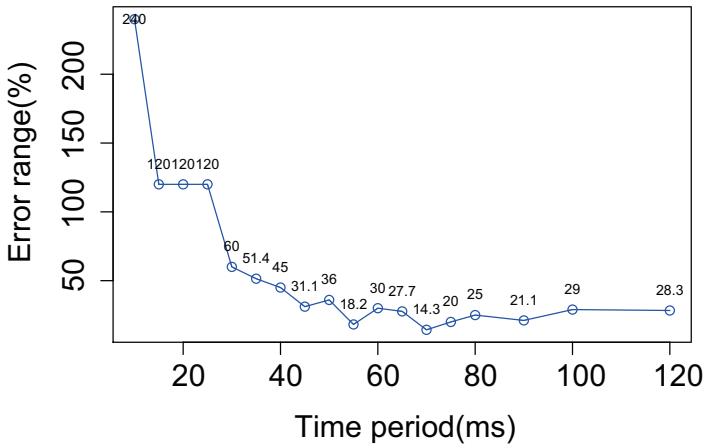


Figure 4.14: Flashlight performance: Error range $= (max - min)/T$

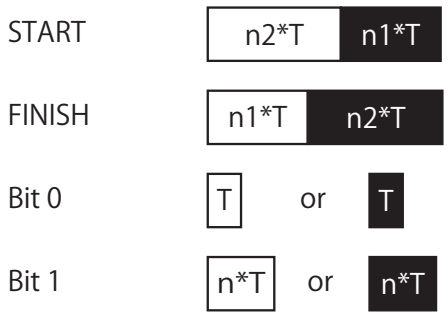


Figure 4.15: Light communication: Light pattern

represent turn ON state, and black period is turn OFF state.

We represent 0 bit is a blink-light with time length is T in both ON or OFF state. 1 bit is a blink-light with time length is $n * T$. With START flag, we need to start it with turn ON state light, therefore we make START pattern which is turn ON in $n1 * T$ then turn OFF in $n2 * T$. With FINISH flag, because number of data bits is a even number, we represent FINISH float with turn ON in $n2 * T$ then turn OFF in $n1 * T$. The example of proposed light pattern is shown in Figure 4.16 below.

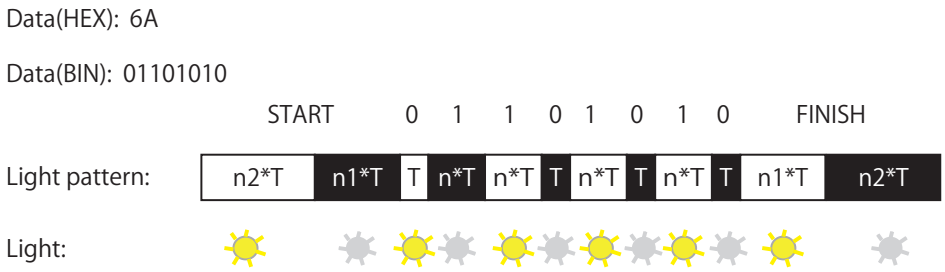


Figure 4.16: Light communication: Encoding example

4.6.4 Decoding Light Pattern

In this part, we present about decode light signal to data by use light sensor of sensor node and photocell of accessory. We divide the decode process in to 3 steps, *(1)read light signal*, *(2)detect light period time from light signal*, *(3)get light pattern from light period time and get data*.

Decoding by light sensor of sensor node

In step *(1)read light signal*, it is very simple. We get signal by reading state of light sensor and storing light value in a array for each fixed length window (Figure 5.3).

In step *(2)detect light period time from light signal*, most important work is how to calculate threshold for detect HIGHT and LOW brightness. Because we only turn ON or turn OFF flashlight and getting data cycle is about some millisecond, then the signal always changes immediately from LOW to HIGHT or HIGHT to LOW when we change state of flashlight.

Like Figure 4.9 we have a lot of noise signal, but different from signal of flashlight, the change of noise is lower with more frequency. Therefore we can use these properties to calculate threshold dynamically. We propose a method base on the change of light signal in two continuos signals (Figure 4.18).

At first, we collect signal in each window with fixed size (Figure 5.3), we collect frequency of changed value Δ between two signal S_{n-1} and S_n ($\Delta = |S_{n-1} - Sn|$). After that, we find

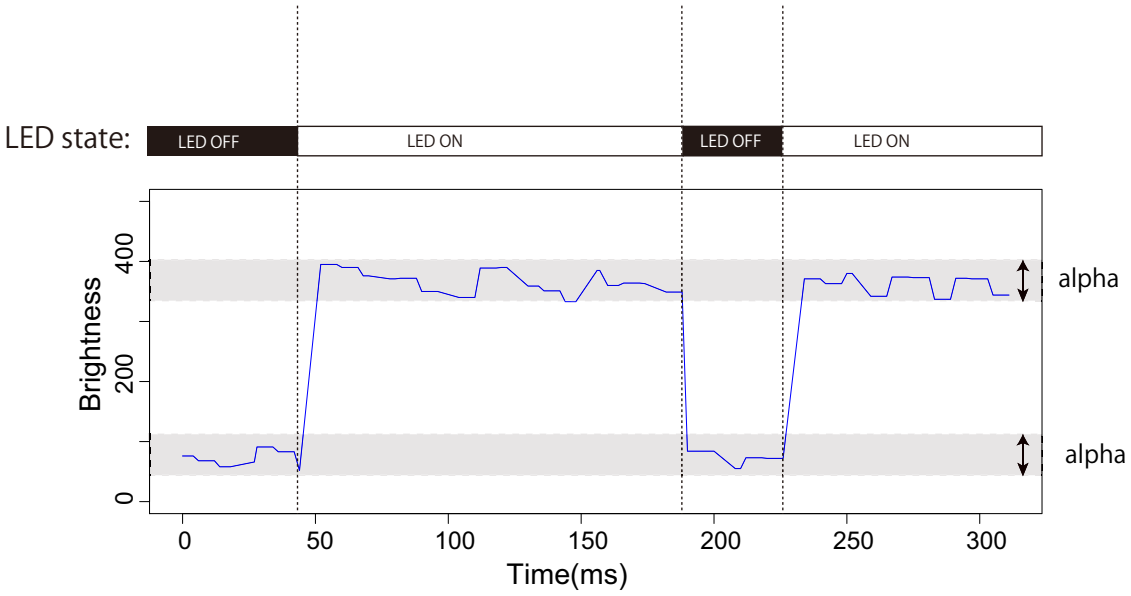


Figure 4.17: Decoding with light sensor: a window

longest empty segment (called LES) whose frequency is zero from frequency array. This segment starts from LES_{begin} to LES_{end} (Image of this process is shown in Figure 4.18). We also find the minimum signal S_{min} in this window and calculate threshold for this window by below equation:

$$Th = S_{min} + \frac{LES_{begin} + LES_{end}}{2} \tag{4.1}$$

And now, we prove it work. From properties of noise signal and blink signal, noise signal has low change and blink signal has high change of brightness. Therefore we can expect that LES can split Δ array in to two parts, noise's Δ and blink's Δ. In addition, from Figure 5.3 we can image that value of brightness is changed in $[S_{min}, S_{min} + \alpha]$ or $[S_{max} - \alpha, S_{max}]$. Then we have:

$$\begin{cases} LES_{end} & \gg & \alpha & \text{because } \alpha \text{ from noise} \\ LES_{end} - LES_{begin} & \gg & \alpha & \\ LES_{begin} & > & \alpha & \text{because } \alpha \text{ from noise} \\ LES_{end} & < & S_{max} - S_{min} & \end{cases} \tag{4.2}$$

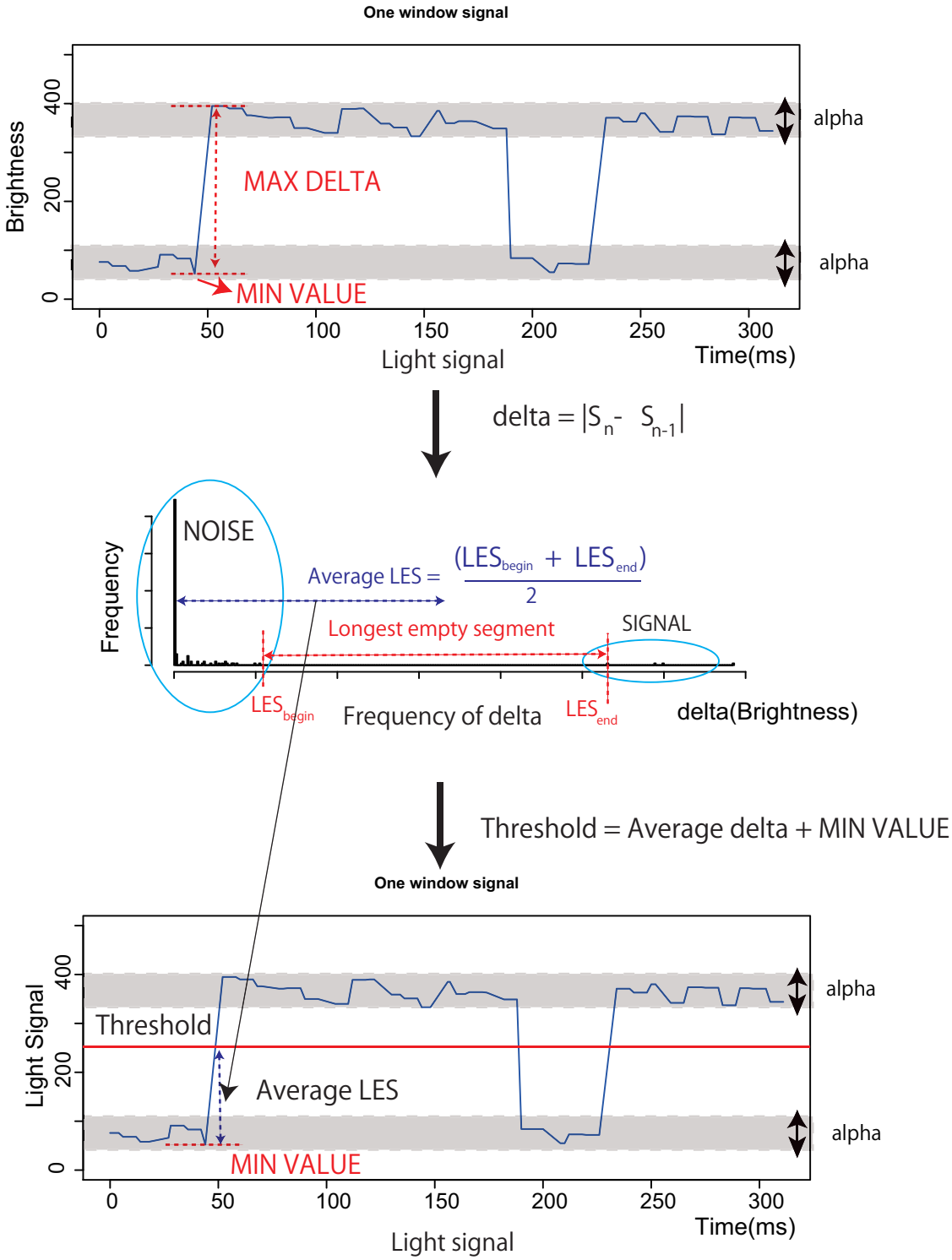


Figure 4.18: Dynamic threshold

Because $LES_{begin} > \alpha$ and $LES_{end} > LES_{begin}$ then we have

$$\begin{aligned} S_{min} + \frac{LES_{begin}+LES_{end}}{2} &> S_{min} + \frac{LES_{begin}+LES_{begin}}{2} \\ &> S_{min} + \frac{\alpha+\alpha}{2} \\ &> S_{min} + \alpha \end{aligned} \tag{4.3}$$

Same with this, we also have $LES_{end} < S_{max} - S_{min}$ (then we can write $S_{min} + LES_{end} < S_{max}$) and $LES_{end} - LES_{begin} \gg \alpha$ (then we can write $LES_{end} - LES_{begin} > 2\alpha$)

$$\begin{aligned} S_{min} + \frac{LES_{begin}+LES_{end}}{2} &< S_{min} + \frac{LES_{end}-2\alpha+LES_{end}}{2} \\ &< S_{min} + LES_{end} - \alpha \\ &< S_{max} - \alpha \end{aligned} \tag{4.4}$$

Therefore, we have

$$[S_{min}, S_{min} + \alpha] < Th < [S_{max} - \alpha, S_{max}] \tag{4.5}$$

and we can use Th to determine low or high signal.

After we calculated threshold value, we compare each value of light signal and threshold to detect the beginning and the ending of light pattern. The beginning of current light pattern is defined by previous light value S_{n-1} lower threshold Th and now light value S_n higher or equal threshold Th. In other cases, the ending of previous light pattern is defined by previous light value S_{n-1} is higher or equal threshold Th and now light value S_n is lower threshold Th. With this, we can detect length time of each turn ON, OFF period and detect light pattern.

In the next step *(3)get light pattern from light period time and get data*, we use detected light period length time to detect light pattern. This process relies on structure of light pattern, which is described in previous part *Encoding Light Pattern*. The first, we set status is *CHECKING-START-FLAG*. We compare now period length and previous period length with base time T to check "whether it is START pattern or not". In case we detected START pattern, the status is switched to *GETTING-DATA-AND-CHECK-FINISH-FLAG*.

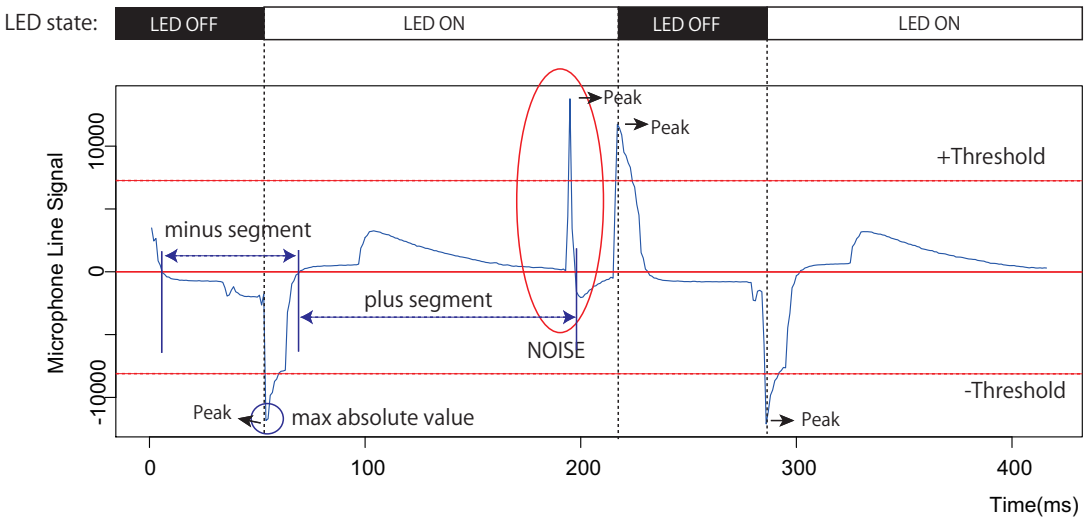


Figure 4.19: Decoding with photocell: a window

In *GETTING-DATA-AND-CHECK-FINISH-FLAG* state, we check the current period and previous period with base time T to check "whether it is FINISH pattern or not?". If it is not, we check the current period length to get 0 or 1 bit. In case FINISH pattern is detected, we finish receiving process , get array of bit and reset status to *CHECKING-START-FLAG*.

Decoding with photocell of accessory with Smartphone

Similar with light sensor case, in photocell case we also have 3 steps to decode light pattern. There are *(1)read light signal*, *(2)detect light period time from light signal*, *(3)get light pattern from light period time and get data*. Step(1) is similar with previous case. Unlike light sensor case in step (2), we have different ways to detect the start of blinking light period. In Figure 5.4 we show data from photocell when we change state of LED. We have the change of signal correlative with LED, then we can detect the end and begin of a light period by get maximum absolute value of light signal in a minus segment or plus segment. If the max absolute value is higher than threshold, it is the beginning of new light period and the ending of previous light period. We also have a similar way to calculate Threshold value in each window, which

is described in light sensor case. We also have a NOISE which is marked by red circle, and we can recognize that this noise have the same sign with next peak, therefore we can avoid this type of noise by checking sign of the current peak with previous peak. If it has the same sign, we save this peak as the end of light period instead of previous peak. In other cases, if it doesn't have the same sign, we process a light period which is the end in previous peak and set the begin of new peak is previous.

In step (3) to get light pattern from light period time and decode to get data, we do it the same way with light sensor case.

4.7 Summary

In this section, we summarize this chapter. We explain HUSTLE by showing hardware requirements for it, and how it works. In HUSTLE, we assumes sensor node has light sensor or LED, we use light sensor to receive one-time security key from the smartphone's flashlight or use LED to send data to the smartphone by using a accessory. With this key we can encrypt and decrypt setup data and send it safely. We also describe encode and decode light pattern with distinct properties of home environment and LED, flashlight which is used in HUSTLE. With this, HUSTLE can successfully fulfill all requirements in our research.

Chapter 5

Implementation of HUSTLE

In this chapter we describe the implementation of HUSTLE. Firstly, we discuss about hardware and software in HUSTLE. In the next section, we describe the implementation of each module. Lastly, we discuss interactions in HUSTLE and summary this chapter.

Figure 5.1: Implementation: Hardware

5.1 Hardware and Software

To implement HUSTLE, we used Samsung Google Nexus S Smartphone, SunSpot Java Developer Kit that is shown in Figure 5.1. We used a photocell CDS 11mm MI11516 and a 3.5mm microphone jack as an accessory. The full specification of Nexus S and Sunspot Java Developer Kit is shown in Table 5.1 and Table 5.2.

With Nexus S Smartphone, we have a touch screen and a controllable flashlight. Nexus S is programmed by the Java language with Android SDK. In this time, we used Android SDK 4.1 to program it. Sunspot Java Developer Kit consist SunSpot sensor node and Base Station that is used as Sink Node. SunSPOT Java Developer Kit is also programmed by Java language with Sun SPOT SDK.

We use SunSPOT sensor nodes to make a secured WSN with multi-hop routing protocol and Network-wide keys scheme [5] for secure protocol.

5.2 Modules Implementation

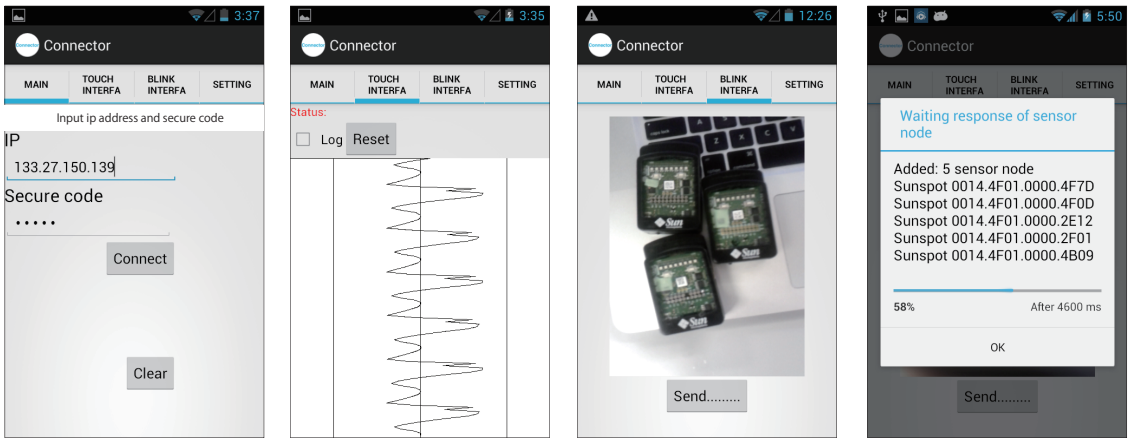
This section describes implementation of each module in HUSTLE: user interface module, wireless communication module, controlling module, and light communication module.

Table 5.1: Specific hardware of SunSpot Sensor Node

Processing	180 MHz 32 bit ARM920T core - 512K RAM - 4M Flash 2.4 GHz IEEE 802.15.4 radio with integrated antenna AT91 timer chip
Sensor Board	2G/6G three-axis accelerometer Temperature sensor Light sensor 8 tri-color LEDs 6 analog inputs 2 momentary switches 5 general purpose I/O pins and 4 high current output pins
Battery	3.7V rechargeable 750 mAh lithium-ion battery 30 uA deep sleep mode Automatic battery management provided by the software

Table 5.2: Specific hardware of Samsung Galaxy Nexus S

Processing	Samsung Exynos 3110 1 GHz ARM Cortex A8 based CPU core with a PowerVR SGX 540 GPU 512MB of RAM - 16GB of NAND memory
Screen	4.0 inch touch screen
Data input	3-axis gyroscope Accelerometer Ambient light sensor Capacitive touch-sensitive buttons Digital compass Microphone Multi-touch capacitive touchscreen Proximity sensor Push buttons
Camera	5.0 megapixel rear camera with LED flash VGA front camera



(a) Login step

(b) Receive light data (Touch interface)

(c) Send data with flashlight (Blink interface)

(d) Feedback

Figure 5.2: User interfaces

5.2.1 User interface module

This module plays a role in interacting with users such as how to start sending or receiving data and how to feedback to users. The interfaces are shown in Picture 5.2. We have four screens: (a) logging screen, (b) receiving light data for touch interface, (c) sending data with flashlight for blink interface and (d) feedback screen. To begin with, users must to login with their accounts. After that, to add a new sensor node, users can use (b) or (c) (Picture 5.2) which depends on hardware of sensor node. Lastly, in order to feedback to users, we show a list of added sensor node's information such as its type and address like Picture 5.2(d). Picture (b) shows interface in case of adding a new sensor node with accessory. We show light signals and also show status of reading data process. In Picture (c), we show interface of adding processing by using the flashlight of smartphone. We also show capture of camera, which can help users know what sensor nodes will be added.

5.2.2 Wireless communication module

This module is used to send data to others by using radio communication such as secure protocol data, routing data, or identification. This module receives commands from controlling module consist which and where data must be sent. In case address of destination is undefined, this module broadcasts the message to all sensor nodes.

5.2.3 Controlling module

This module determines which data will be sent when having an event such as new light data and new radio data. Rules table of each device is shown in Table 5.3 5.4 5.5 5.6.

5.2.4 Light communication module

This module is the most important one in our research, so we describe it more specific. We split this module into 2 parts: encoding and decoding light pattern.

Encoding Light Pattern

This part is very simple, it only get string bits of data and send the light pattern according bit value. The Java code of this part is shown in Listing5.1.

Listing 5.1: Encoding light pattern

```
1  private void send_light_pattern(String data){
2      //send START pattern, turn on LED in n1*T and turn off in n2*T
3      set_led_time("ON",n1*T);
4      set_led_time("OFF",n2*T);
5      //send light pattern of each bit
6      for(int i=0;i<data.length;i++){
7          if(data.charAt(i)=='0'){
8              if(i%2==0) set_led_time("ON",T);
9              else set_led_time("OFF",T);
10         }else if(data.charAt(i)=='1'){
11             if(i%2==0) set_led_time("ON",n*T);
12             else set_led_time("OFF",n*T);
13         }
14     }
```

Table 5.3: Rule table of Smartphone

Event	Action
New light data	Send received key to Sink node
New added node information	Show it in screen
Push start button	Make a random security key and send it by flashlight

Table 5.4: Rule table of Sensor node with light sensor

Event	Action
New light data	Save received key
New information from sink node	Decrypt by saved key and setup with this data, send hello message to sink node

Table 5.5: Rule table of Sensor node with LED

Event	Action
Started	Make and save a random security key and send it by LED
New information from sink node	Decrypt by saved key and setup with this data, send hello message to sink node

Table 5.6: Rule table of Sink node

Event	Action
New security key message	Encrypt setup data with this key and broadcast it without secure protocol
New hello message	Send this node information to Smartphone

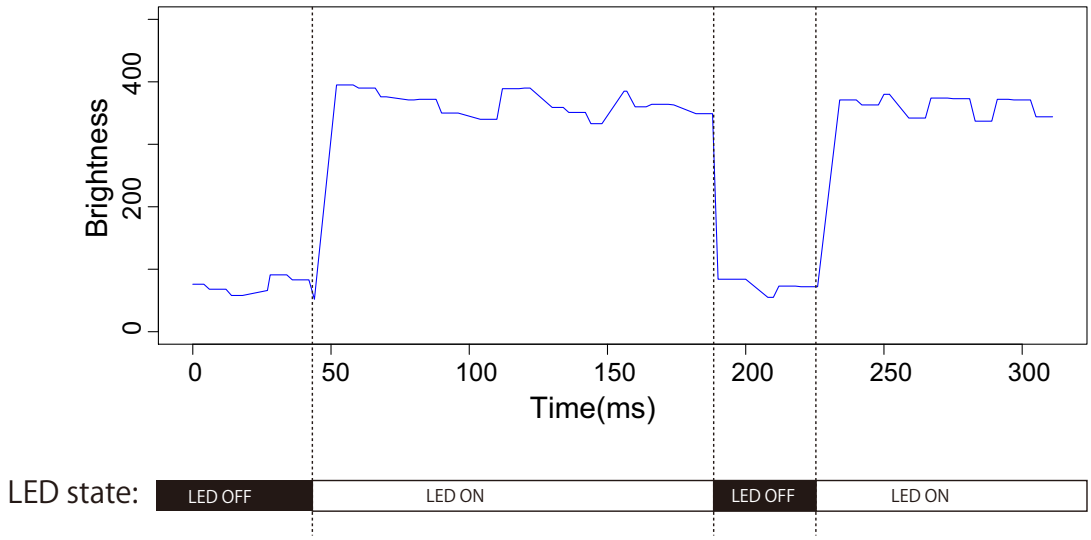


Figure 5.3: Decoding light pattern with light sensor: a window

```
15      //send FINISH pattern, turn on LED in n2*T and turn off in n1*T
16      set_led_time("ON",n2*T);
17      set_led_time("OFF",n1*T);
18  }
```

Decoding Light Pattern: light sensor of sensor node

As mentioned in the previous chapter (HUSLTE chapter), we have 3 steps: (1) getting light signals,(2) detecting period length time, (3) decoding period length time to light pattern and getting data. At first, we get light signals by using the light sensor at step (1) based on a window with size W_LENGTH as Listing 5.2 and Figure 5.3.

Listing 5.2: Light sensor case: read sensor data

```
1  private void read_light_sensor_data(){
2      int data_part = 0;
3      while(true){
4          for(int i=0;i<W_LENGTH;i++){
5              light_data[data_part][i]= ligtSensor.getValue();
6              time_data[data_part][i] = System.currentTimeMillis();
7              Utils.sleep(SLEEP_TIME);
8              new ProcessDataThread(light_data[data_part],time_data[data_part]);
9          }
```

```
10            data_part = (data_part==0?1:0);
11        }
12    }
```

In new thread, we calculate threshold then using this value to determine when the light period is starts or finishes. The source code of this task is shown in Listing 5.3 and 5.4.

Listing 5.3: Light sensor case: Dynamic threshold function

```
1  public int calc_threshold(short[] data,int length){
2      short max_delta =0;
3      short min_light = MAX_SHORT;
4      //find max value of delta and light signal
5      for(int i=1;i<length;i++) {
6          if(max_delta<Math.abs(data[i]-data[i-1])) max= Math.abs(data[i]-data[i-1]);
7          if(max_light<data[i]) max_l=data[i];
8      }
9      //make new frequency array, which store frequency of each delta value
10     byte[] frequency = new byte[max_delta+1];
11     for(int i=0;i<max_delta+1;i++) frequency[i]=0;
12     //count delta value and add to frequency array
13     for(int i=1;i<length;i++){
14         frequency[Math.abs(data[i]-data[i-1])]++;
15     }
16     //find longest empty segment
17     short begin=0,end=0;
18     short les_begin=0;//start of longest empty segment
19     short les_end=0;//end of longest empty segment
20     for(short i=1;i<max_delta-1;i++){
21         if(frequency[i-1]!=0&&frequen[i]==0){//start of empth segment
22             begin = i;
23         }else if(frequen[i]==0&&frequen[i+1]!=0){//end of empth segment
24             end = i;
25             //check is it longest empth segment??
26             if(end-begin > les_end - les_begin){
27                 les_end = end;
28                 les_begin = begin;
29             }
30         }
31     }
32     //return threshold value
33     return min_light+ ((0.5*les_begin+0.5*les_end));
34 }
```

Listing 5.4: Light sensor case: Detect light period function

```
static boolean stt = false;//status of light period, TRUE is ON period, FALSE is OFF period
static long pre_time = 0;//time of previous light signal
static long start_time = 0;//start time of one period
public void detect_light_pattern(int now, long now_time) throws IOException{
    if(now>=threshold &&stt==false){//end of OFF period, start of ON period
        stt=true;//set it is ON period
        count = (int)((now_time+pre_time)/2 - start_time);//calculate time length of previous OFF period
        start_time = (now_time+pre_time)/2;//save begin time of ON period
        new_period(count);//process with new period
    }else if(now < threshold && stt == true){//end of ON period, start of OFF period
        stt=false;//set it is OFF period
        count = (int)((now_time+preVT)/2 - start_time);//calculate time length of previous ON period
        count*=-1;//set it is minus value to distinguish with OFF period
        start_time = (now_time+pre_time)/2;//save begin time of OFF period
        new_period(count);//process with new period
    }
    //update pre_time
    pre_time = now_time;
}
```

After that, with light period information (start time and finish time) we can check light pattern such as START, FINISH ,0 bit or 1 bit; and we also get data from light pattern. This task is shown in Listing 5.5.

Listing 5.5: Light sensor case: Decoding light period function

```

private void new_period(byte len){
    if((pre_len >= n1*T-delta && pre_len <= n1*T+delta)
                    &&(len >= n2*T-delta && len <= n2*T+delta)){//check START pattern
        add("START");
        status=true;
    }else if(status==true&&(pre_len >= n2*T-delta && pre_len <= n2*T+delta)
                                &&(len >= n1*T-delta && len <= n1*T+delta)){//check FINISH pattern
        add("FINISH");
        status=false;
    }else if(status==true){//check bit daat
        if(len>0&&len<T){//it is 0 bit
            add("0");
        }else if(len>T&&len<n*T){//it is 1 bit
            add("1");
        }
    }
    pre_len=len;
}
```

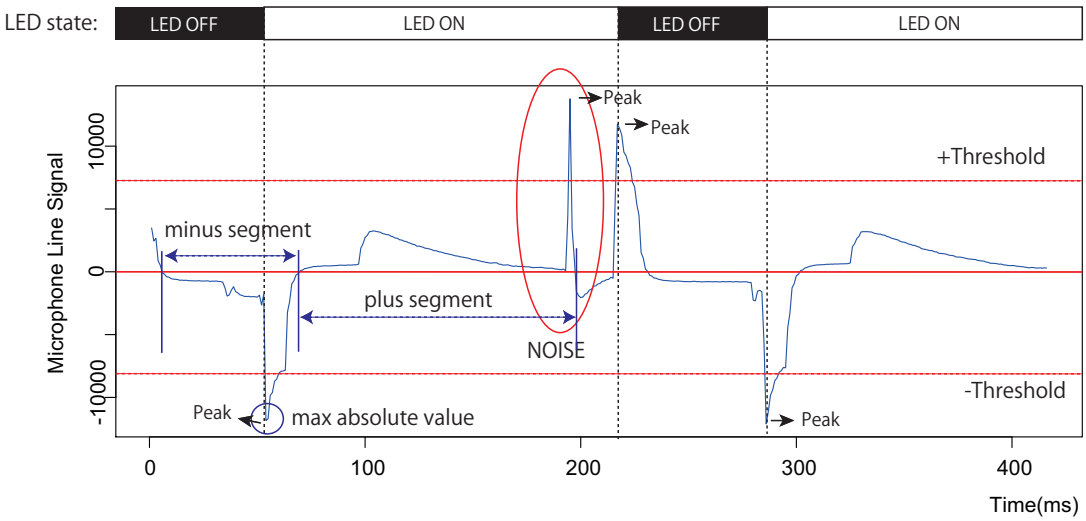


Figure 5.4: Decoding light pattern with photocell: a window

Decoding Light Pattern: photocell of accessory with Smartphone

As said in HUSTLE chapter, in this case, we have only a difference on detecting light period step. We must detect peaks and check this peak whether it is true peak or noise peak. Source code for this step is shown in Listing 5.6

Listing 5.6: Photocell case: Detect light period function

```
1  static max_value   = 0;
2  static max_value_time  = 0;
3  public void detect_light_pattern(int length, int[] data, int[] data_time){
4      for(int i=1;i<length;i++){
5          if(Math.abs(data[i]) > Math.abs(max_value){//update max value of segment
6                  max_value   = data[i];
7                  max_value_time = data_time[i];
8          }
9          if(data[i-1]*data[i]<=0){//start new minus or plus segment
10                 if(Math.abs(max_value) > THRESHOLD){
11                         new_peak(max_value, max_value_time);
12                 }
13                 max_value =0;//reset
14         }
15     }
16 }
17 static int pre_peak   =0;
18 static int pre_peak_time =0;
```

```
19 static int tmp_peak   =0;
20 static int tmp_peak_time =0;
21 public void new_peak(int value, long time){
22         if(value*tmp_peak<0){//true peak
23                 new period(tmp_peak_time-pre_peak_time);
24                 //update new peak information
25                 pre_peak   = tmp_peak;
26                 pre_peak_time  = tmp_peak_time;
27                 tmp_peak   = value;
28                 tmp_peak_time  = time;
29         }else{//noise peak
30                 tmp_peak   = value;
31                 tmp_peak_time  = time;
32         }
33 }
```

5.3 Interactions

As mentioned in the previous chapter, HUSTLE provides two interactions for adding new sensor nodes, which is shown in Figure 5.5. In touch interaction, users only have to turn on the new sensor node, touch it with the accessory of smartphone like Figure 5.5A. In difference with touch interaction, in blink interaction (Figure 5.5B) users can add multi-sensor nodes at the same time in one-shot. Users only have to turn on all new sensor nodes and direct flashlight of smartphone to them, which is used to send information.

5.4 Summary

This section summarizes this chapter. Firstly, we described that we implement HUSTLE on Samsung Galaxy Nexus S Smartphone, SunSpot Java Development Kits. We also described implementation of each module such as controlling module, wireless communication module, light communication module and user interface module. At last, we showed the implementation of light communication module, encoding and decoding part with Java code, which is used in Smartphone and Sensor Node.

A: Touch interaction

B: Blink flashlight interaction

Figure 5.5: Interactions in HUSTLE

Chapter 6

Evaluation of HUSTLE

In this chapter, we present the evaluation of HUSTLE. Firstly, we explain the purpose of the evaluation. Secondly, methodology of evaluation, comparison targets and evaluation items are explained in detail. Next, we show the result of the evaluation in terms of evaluation items for each comparison target. Finally, we summarize this chapter.

6.1 Purposes of the Evaluation

The ultimate goal of HUSTLE is to reduce time to deploy a secured WSN. Therefore, we need to check the ability of HUSTLE when making a new secured WSN or when adding new sensor nodes and comparing with base method. On the other hand, the accuracy of adding process also becomes important, otherwise users need a lot of times to add new sensor nodes. We also check the accuracy of HUSTLE when adding new sensor nodes.

The other goal of HUSTLE is to provide a friendly and intuitive interaction for deploying a secured WSN. Thus, throughout the evaluation about the ability of HUSTLE, we also take a questionnaire about HUSTLE and base method.

Lastly, speed and accuracy of light communication also affect to add new sensor nodes and make a WSN, so we also evaluate light communication in some environments, indoor with different light conditions.

6.2 Evaluation Methodology

We describe the evaluate environments in the first subsection. In others, we also describe comparison target and evaluation items in next subsection.

6.2.1 Evaluation Environment

This part shows the evaluation environment. The specific of hardware and software is shown in Table 6.1. We implemented HUSTLE on SunSpot Java Development Kits with the latest version of SunSpot SDK,yellow-101117-1 version. SunSpot sensor node has ARM920T CPU with 512KB RAM and 4M Flash. SunSpot is also equipped 2.4 GHz IEEE 802.15.4 radio communication, light sensor, 8 tri-color LED. We used Samsung Galaxy Nexus S Smartphone with Android SDK version 4.1. Nexus S Smartphone has 1GHz ARM Cortex A8 CPU with

512 MB of RAM, 16GB of NAND memory. Nexus S also has a camera with an LED flashlight and a touch screen.

Table 6.1: Evaluation environment

Operating system	Mac OSX 10.7
SunSpot SDK	Yellow-101117-1
Android SDK	4.1
Platform	SunSpot Java Development Kits
CPU	ARM920T
Memory	512KB
Radio	2.4 GHz IEEE 802.15.4
Equipment	Light sensor, 8 tri-color LED
Platform	Samsung Galaxy Nexus S
CPU	1GHz ARM Cortex A8
Memory	512MB
Equipment	LED flashlight, touch screen

In our evaluation,we used 10 SunSpot sensor nodes as new sensor nodes with one base-station of SunSpot Java Development Kits as sink node. The sink node is connected to a personal computer through a USB connection.

All sensor nodes are installed with sensor node version of HUSTLE program. Sink node and smartphone are also installed with HUSTLE program according the sink node version and the smartphone version.

All evaluations is performed at indoor with some light condition like Picture 6.1.

6.2.2 Comparison Targets

This section explains comparison targets of the evaluation, base method based on input security key manually and HUSTLE.

Base method: In this method to add a new sensor node, users have to turn on it, input identification and hexadecimal security key of the new sensor node to sink node. After that,

Figure 6.1: Evaluation: light conditions

by using the security key, sink node can encrypt the setup data and send it to the sensor node safely. Lastly, sensor node decrypts received data, set-up with decrypted data and sensor node is a part of WSN.

HUSTLE: In HUSTLE, we provide friendly interactions, faster method to deploy and maintain security of WSN with low cost. Firstly, users use their smartphone with HUSTLE application and login to them WSN management by IP address and security key. After that, to add new sensor nodes, users only have to turn on it and use the smartphone to set-up new sensor nodes. Depending on hardware of sensor node, we have difference way to set-up it. In case new sensor nodes has a light sensor, simply turn on all of it, put it on the table and direct flashlight of Smartphone into it, press "Send" button and all of new sensor nodes is added automatically. Others case, new sensor nodes have an LED, users turn on each sensor node, touch it with the accessory of smartphone, and the new sensor node will be

added automatically.

6.2.3 Evaluation Items

In this section, we explain the evaluation items. Firstly, we evaluate light communication speed, accuracy with some value of time unit in-door with common light condition, it is fluorescent lamp. Secondly, we evaluate length of time to deploy a secured WSN, its accuracy when adding new sensor nodes and taking questionnaire.

- Speed and accuracy of light communication: Light communication is used to exchange security key between sensor nodes and smartphone. Thus its speed and accuracy affect to performance of adding new node process. We can avoid error with higher accuracy light communication and make adding method become more effective. As the result, we can make a WSN faster. So as to measure speed and accuracy of light communication, we use it to send a fixed size of random data and measure length of time with some values of time unit and n (for simple, we set n_1 and n_2 is $2n$ and $3n$). We evaluate in common light condition of the indoor environment, fluorescent lamp. Lastly, we can use evaluated results to select the best parameters of light communication such as time unit and n.

- Adding new sensor nodes accuracy: Adding accuracy is an important item. We can reduce errors of sensor nodes when deploying a WSN at a higher rate accuracy. In addition, we can make set-up method faster, one of the goals of HUSTLE. We try to add a new sensor node with touch interaction in 100 times with the fluorescent lamp condition, because touch interaction does not depend on light condition. We also evaluate it with some indoor conditions (Figure 6.1) by adding from 1 new sensor node case to 10 sensor nodes case at one-time, each case is looped in 10 times with blink interaction.

- Time length to make a secured WSN: One of the goals of this research is to make a secured WSN faster. Thus, we ask 20 participants to evaluate HUSTLE and base method. We measure the time length of each participant when setting-up a new WSN with 10 sensor nodes in the fluorescent lamp environment.

- Evaluation of users: We also need the evaluation of participants because the setup method is used by end-users. Therefore, after the participants finished deploying the WSN, we also asked participants to fill out a questionnaire survey about each method with following questions: Was it simple to deploy? Was it useful? Was it tiring to deploy? Was it simple to learn? Was it easy to fix in case of errors? (Max score of each question is 4).

6.3 Evaluation Results

In this section, we present the results of the evaluation. At first, we show the succinct results, then we discuss the results of four evaluation items in details.

6.3.1 Succinctly results

We show the succinct result of evaluation in Table 6.2. From this result, we can know that HUSTLE with blink interaction is 6.5 times faster than base method and touch interaction is 3 times faster than base method when deploying a new WSN with 10 sensor nodes. All of the participants evaluated that HUSTLE is better than base method, simplier, more useful, less tiring, easy to learn and easy to fix errors.

6.3.2 Speed and accuracy of light communication

We evaluated speed and accuracy of light communication by sending 8bytes data in in-door environments. We used both light communication case, (sensor node's LED - Smartphone's

Table 6.2: Succinctly result of evaluation

		Base method	HUSTLE.1 [a]	HUSTLE.2 [b]
Adding accuracy	Fluorescent lamp		80.8%[c]	73.64%
	Incandescent lamp			71.09%
	Low sunshine			31.82%
	Low brightness			78.55%
Make a WSN	Time length	596.5s	201.1s	93.3s
	Simply	2.15	3.30	3.80
	Useful	2.40	3.50	3.55
	Tire[d]	3.55	1.60	1.45
	Learnable	2.85	3.45	3.85
	Fix errors	2.30	3.25	3.10

[a]Touch interaction
[b]Blink interaction
[c]Without resend in light communication
[d]Smaller is better

accessory) and (Smartphone's flashlight - sensor node's light sensor) with some different parameters (time unit, n, n_1 and n_2). We tested them in a common indoor environment which is the fluorescent lamp condition.

The results are shown in Figure 6.2 and Figure 6.3. We can see that, the accuracy increases with lower time unit.

In common case fluorescent lamp, with time unit is 10μs, the accuracy changed form 50% to 80% with only changing N from 4 to 5. The reason can be seen from Figure 4.13 in Chapter HUSTLE 4. Because we do not have stability of flashlight with very short light period. For instance: 10μs the range of received period is from 7μs to 32μs, with 40μs, the range is form 30μs to 55μs. This makes a lot of wrong patterns when sending the number of light patterns. In case of increasing N to 5, we have the range of 50μs is from 45μs to 63μs, then we can decrease the number of wrong patterns and get higher accuracy. Because fluorescent lamp is common light condition in home environment, we configure light communication bases on

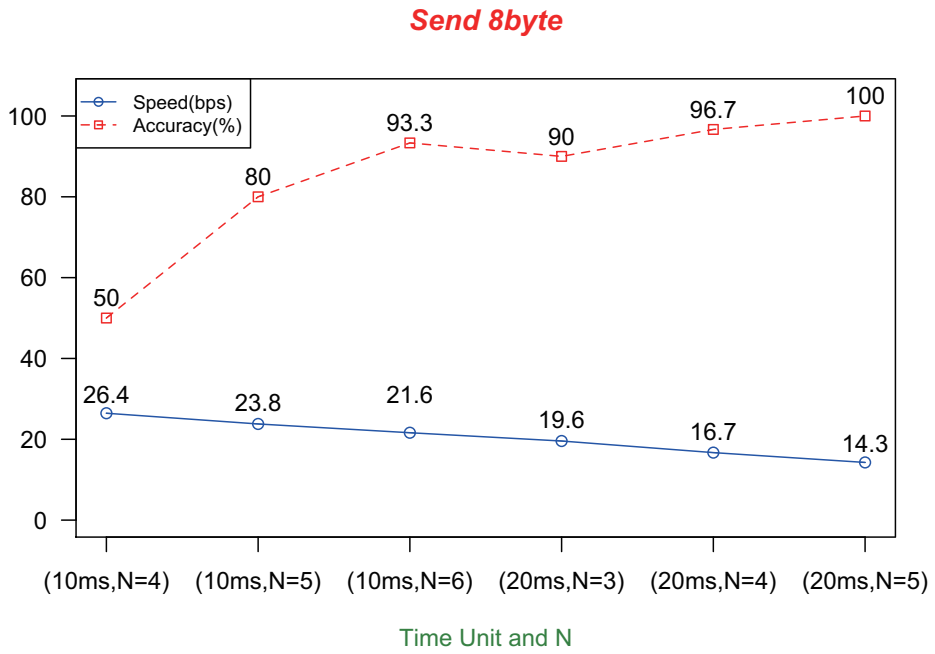


Figure 6.2: Speed and accuracy with smartphone's flashlight: fluorescent lamp

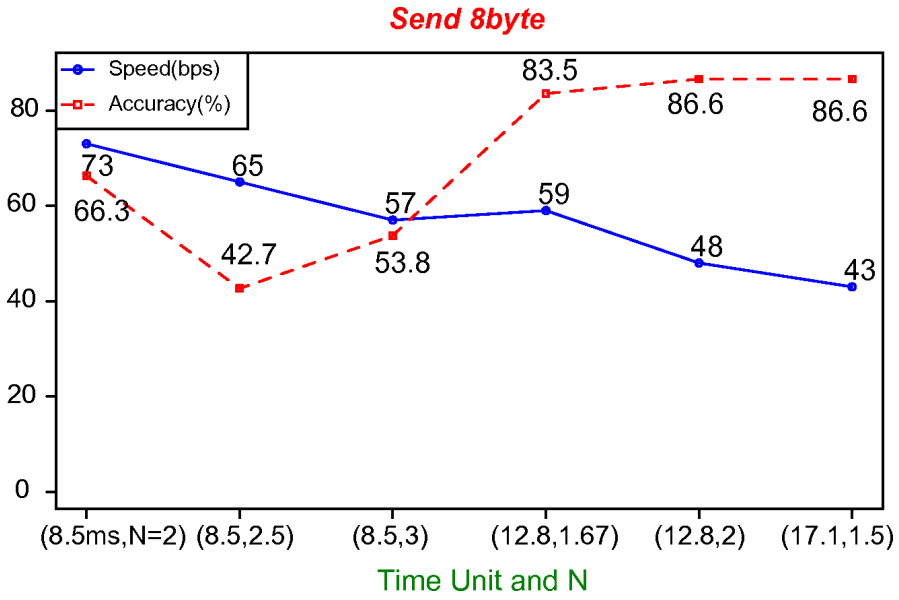


Figure 6.3: Speed and accuracy with sensor node's LED

fluorescent lamp condition. From Figure 6.2, we can chose time unit is 10ms and the value of n is 6. With this value, light communication have balance between speed (21.6bps) and accuracy (93.3%).

In sensor node's LED case (Figure 6.3), with same reasons we have very low accuracy with short time unit, we have a problem when detecting some continue blink periods such as (ON-OFF-ON is detected to ON-ON-ON). Therefore, we chose time unit is 12.8ms and N is 1.57, with this value, we have high speed with high accuracy.

6.3.3 Accuracy when adding new sensor nodes

We measure this item by adding new sensor nodes in some cases with both types of interactions, touch and blink flashlight.

Touch Interaction

Table 6.3: Accuracy when adding new nodes with touch interaction

Required times	1	2	2
Number	32	10	0
Percentage	76.2%	23.8%	0%

In case touch interaction is used, we show the result at Table 6.3. When we try to add new sensor node by touching Smartphone-accessory, we can add it immediately without any errors in 76.2% case. In comparison with other cases, it takes only two times using light communication with one error of light communication and one success of light communication.

Sometimes we need two times to add a new sensor nodes because of the error of light communication. For speeding up, we make a light pattern with shorter period blink light, so we receive some strange periods which difficult to determine 0 or 1 bit. On the other hand,

we have different accuracy when compare light communication. We think that the reason of this problem is differ from among sensor nodes. In speed and accuracy evaluation of light communication, we evaluated it with only one sensor node, but in adding new node accuracy evaluation we used some different sensor nodes.

Table 6.4: Adding new nodes with blink flashlight interaction: fluorescent lamp

Case/Added	0	1	2	3	4	5	6	7	8	9	10	Avg
1 Node	1	9										0.9
2 Nodes	0	2	8									1.8
3 Nodes	0	0	1	9								2.9
4 Nodes	1	0	0	1	8							3.5
5 Nodes	0	0	0	0	2	8						4.8
6 Nodes	0	0	0	0	1	3	6					5.5
7 Nodes	1	0	0	0	0	3	4	2				5.3
8 Nodes	0	0	0	1	3	2	3	1	0			5
9 Nodes	2	0	0	0	2	2	2	2	0	0		4.4
10 Nodes	0	0	0	0	0	3	1	5	1	0	0	6.4

Table 6.5: Adding new nodes with blink flashlight interaction: incandescent lamp

Case/Added	0	1	2	3	4	5	6	7	8	9	10	Avg
1 Node	1	10										1
2 Nodes	0	1	9									1.9
3 Nodes	0	1	3	6								2.5
4 Nodes	0	0	0	2	8							3.8
5 Nodes	0	0	0	0	5	5						4.5
6 Nodes	0	0	0	0	1	7	2					5.1
7 Nodes	0	0	0	0	1	4	5	0				5.4
8 Nodes	0	0	0	0	0	2	3	0	0			4.9
9 Nodes	1	0	1	2	2	2	1	0	0	0		3.7
10 Nodes	0	0	0	0	0	3	3	5	0	0	0	6.9

Table 6.6: Adding new nodes with blink flashlight interaction: low sunshine

Case/Added	0	1	2	3	4	5	6	7	8	9	10	Avg
1 Node	3	7										0.7
2 Nodes	0	4	6									1.6
3 Nodes	0	4	4	2								1.8
4 Nodes	2	0	2	6	0							2.2
5 Nodes	1	1	4	2	2	0						2.3
6 Nodes	0	1	4	4	1	0	0					2.5
7 Nodes	2	2	2	2	2	0	0	0				2
8 Nodes	3	1	4	3	0	0	0	0	0			1.6
9 Nodes	3	2	2	3	0	0	0	0	0	0		1.5
10 Nodes	3	3	2	2	0	0	0	0	0	0	0	1.3

Table 6.7: Adding new nodes with blink flashlight interaction: low brightness

Case/Added	0	1	2	3	4	5	6	7	8	9	10	Avg
1 Node	1	9										0.9
2 Nodes	0	1	9									1.9
3 Nodes	0	0	0	10								3
4 Nodes	0	0	0	1	2							3.9
5 Nodes	1	1	0	1	2	5						3.7
6 Nodes	2	0	1	0	0	4	3					4
7 Nodes	0	0	0	0	1	3	4	4				6.1
8 Nodes	0	0	0	0	0	1	1	6	2			6.9
9 Nodes	1	0	0	0	1	3	3	0	2	0		5.3
10 Nodes	0	0	0	0	0	0	1	4	3	1	0	7.9

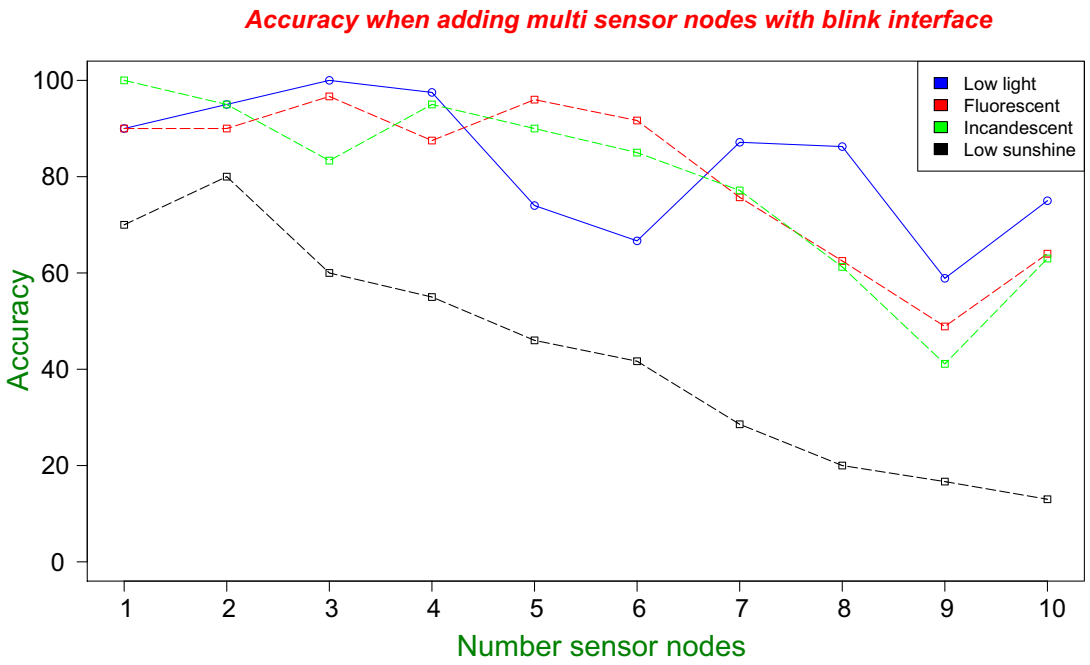


Figure 6.4: Adding percentage with blink interaction

Blink Interaction

Table 6.4 6.5 6.6 6.7 and Figure 6.4 show the results in case blink interactions is used. We tested it by adding from 1 new sensor node case to 10 sensor nodes case at one-time, each case is looped in 10 times.

When comparing with low brightness case and incandescent lamp case, we can see that the average number of added sensor nodes in each case is decreases if we change from low brightness environment to incandescent. This make light sensor difficult to distinct light from the flashlight and light from the environment. Especially in a sunshine environment (Table 6.6 and Figure 6.4), the accuracy is increases at a high rate with more sensor nodes. In comparison with others condition such as low light, fluorescent, or incandescent, we received very low accuracy in low sunshine environment case.

In addition, we also have a problem when adding too many sensor nodes. For example,

in Table 6.4 with 10 sensor nodes case, 30% test can only add 5 nodes, and 50% test can only add 7 nodes. This is because of the limitation of the range of smartphone's flashlight. With more sensor nodes, we need a larger space. In order to enlarge the range of flashlight we have to take smartphone farther, so it also makes the light signal more weak (signal strength is calculated from: $I = \frac{C}{d^2}$) and harder to handle. Another reason is with many sensor nodes, sometimes we cannot send and receive a packet in case of a lot of packets are sent in a short time. We try to solve it by setting random waiting time in each sensor node, but this problem still occurs, such as in 9 sensor nodes case.

Another reason of low accuracy is usage of users. For best result, flashlight needs to be directed to all sensor nodes with as strong as light possible, but each user has each different usage, then sensor nodes received different light signals strength in every time.

From all results, we can conclude that 4 sensor nodes is the best case. We can direct flashlight to all 4 sensor nodes with strong signals.

6.3.4 Time length to make a secured WSN

For evaluating usability of HUSTLE, we asked 10 participants to evaluate the performance of HUSTLE, and compared with base method. With base method, participants have to select true sensor nodes from the list and input security key of this sensor node. We measure the time length when each user sets-up a WSN with 10 sensor nodes in common indoor condition, fluorescent lamp.

Firstly, we can see the big difference of time length in base method. The reason is difference among participants, someone can input security code faster and feel familiar with hexadecimal.

From Figure 6.5, we can see HUSTLE with blink interaction is about 6.5 times faster than base method (93.3s and 596.2s). The reason is with HUSTLE users only have to use

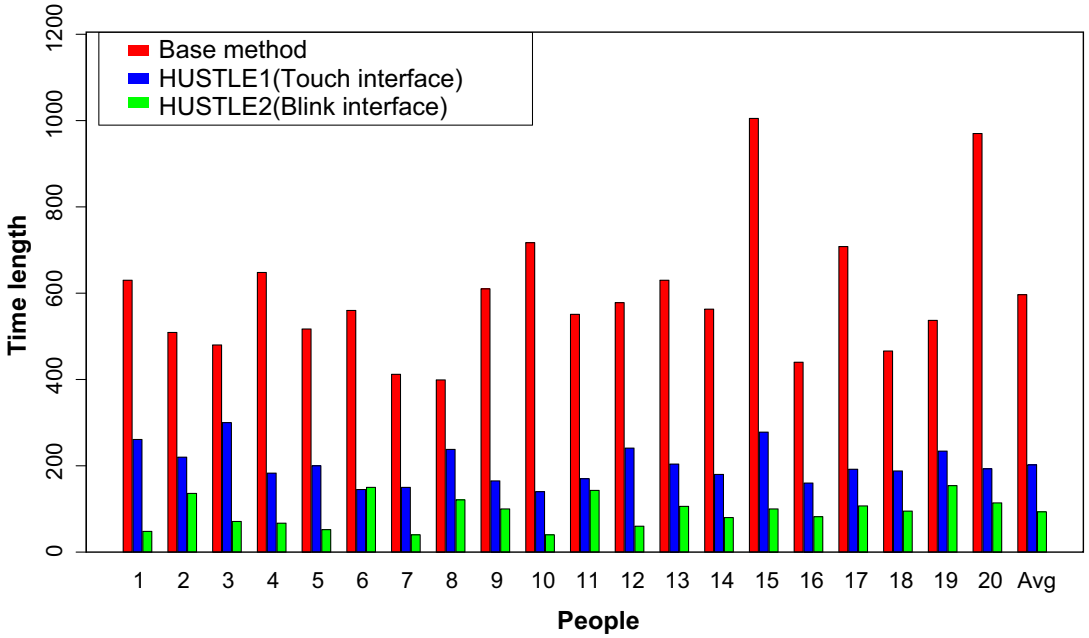


Figure 6.5: Setting-up time length

Smartphone to shine to all sensor nodes, this is very simple to perform.

To add a new sensor node, participants have to select true sensor node and input the security key of each sensor node. This task takes lots of time to perform. In touch interaction, we only add one sensor node at once, so HUSTLE with touch interaction is slower than blink flashlight interaction. However it still 3 times faster than base method (202.1s and 596.5s).

6.3.5 Evaluation of users

We also asked participants to fill out a questionnaire survey about each method with following questions: Was it simple to deploy? Was it useful? Was it tiring to deploy (lower is better)? Was it simple to learn? Was it simple to fix in case of errors? (Max score is 4). The results are shown in Table 6.8.

In Table 6.8, all participants evaluated good results about HUSTLE. In HUSTLE with

Table 6.8: Average question score

	Simply	Useful	Tiring	Learnable	Fixes errors
Base method	2.15	2.40	3.25	2.85	2.30
HUSTLE(Touch Interaction)	3.30	3.50	1.60	3.45	3.25
HUSTLE(Blink Interaction)	3.80	3.55	1.45	3.85	3.10

blink interaction, we can make WSN through simple interaction, so it has (3.30/4 and 3.80/4) point with Simply and (3.50/4 and 3.55/4) with Useful. In HUSTLE with blink interaction, one-time security key is sent automatically to all sensor nodes and also can check received data by checking checksum byte. Therefore it avoids wrong security key problem which usually happens with base method. Thus, HUSTLE with blink interaction got (1.60/4 and 1.45/4) point with Tire instead of 3.25/4 of base method (With Tiring, lower score is better).

In Learnable and Fix errors question, almost participants evaluated HUSTLE with blink interaction well with (3.45/4 and 3.85/4) point of Learnable and (3.25/4 and 3.10/4) of Fixes error. Therefore we can say that HUSTLE can be easily applied with current WSN.

Almost participants commented that HUSTLE can make a WSN easier, faster. On the other hand, some people also said that HUST with blink interaction is better than touch interaction: they need less time and less interaction. As the result, HUSTLE with touch interaction received slower score in all questions (Table 6.8).

6.4 Summary

In this section we summarize this chapter. First we introduced the purpose of evaluation: checking performance of HUSTLE and its usability. We discussed about the methodology of evaluation in the next section, about the environment and compare targets. Next, we described evaluation items, which is taken from the purposes of evaluation. They are: speed and accuracy of light communication, accuracy of adding new nodes, time length for deploying a secured WSN and evaluation of users. Finally, we have shown the results of evaluation

in each evaluation item. From this results, we can see that HUSTLE is faster, simpler than base method. And with this result, we can say that HUSTLE can be applied with current WSN.

Chapter 7

Conclusions and Future Works

7.1 Conclusions

In this thesis, we propose HUSTLE method to help users make a secured WSN faster and more simply. HUSTLE is used to solve deploying problem, which occurs when applying WSN at home for regular users. Compared with base method and another method, instead of special hardware for communication and identification, HUSTLE only use common hardware of sensor nodes and Smartphone such as sensor node's light sensor, sensor node's LED, smartphone's flashlight and a simple accessory of smartphone.

We use LED and light sensor to implement light communication between smartphone and sensor node. On the other hand, because light from Smartphone flashlight and sensor node's LED are only active on small area, we can send data safely in home, a private environment. Therefore we can provide simple interactions for exchanging one-time security key between smartphone and sensor nodes by light communication such as touch new sensor node with smartphone's accessory and setup multi new sensor nodes at same time with blink flashlight interaction. With this key, we can encrypt setup data which is sent over wireless communication and decrypt data for setup process. With this order, users can add new sensor nodes securely more simply, therefore making a secured WSN by themselves.

We implemented HUSTLE with SunSPOT sensor node and Google Nexus S smartphone.

We also evaluate HUSTLE by comparing with base method which selects exactly new sensor nodes and input secure code of new sensor node method. The results of evaluation (Table 6.2) shows that HUSTLE is simpler and faster than base method about six times. This also shows the effectiveness of using HUSTLE instead of base method. In HUSTLE, users only have to touch or direct light to new new sensor nodes, therefore receiving high score with "Simple to use", "Useful", "Easy to learn" item ([3.30, 3.50, 3.45] and [3.80, 3.55, 3.85] instead of [2.10, 2.40, 2.90] in base method). In addition feedback information is shown directly on Smartphone screen and users find it easier to solve occurred problem when adding new sensor nodes.

7.2 Future Works

We discuss the future work of HUSTLE in this section.

The first, we implemented HUSTLE in SunSPOT sensor nodes, and however this is only one type of sensor nodes, and we have a lot of other kinds of sensor nodes, which are used nowadays. With this reason, we want to provide HUSTLE not only for SunSPOT, but also for other types of sensor nodes such as mote, Tmote, iSense (Table 2.1). And in some application we need some type of sensor nodes such as secure application, therefore in the future we want to apply HUSTLE with a secured WSN with some type of sensor nodes.

The second, through the evaluation, we confirmed that light communication is still too slow, and it take a long time to send a longer security key with lower accuracy. With this reason we also need to improve light communication speed also accuracy.

The last but not the least, from evaluation of some participants we learn some problems with touch interfaces because it needs a accessory. And only one sensor node can be added at once, then it take more time to make a WSN. Therefore in future, we consider using smartphone's camera to read signal from sensor node's LED instead of a accessory. We can

add multi new sensor node at once without any external hardware.

Acknowledgments

Foremost, I would like to express my great appreciate to my supervisor, Professor Hideyuki Tokuda, Professor Hideyuki Tokuda who gave me advice, guidance and encouragement from the beginning when I has just joined to Hide.Tokuda Laboratory.

I would like to thank Professor Jun Murai, Osamu Nakamura and Kenji Takeda, Associate Professor Keisuke Uehara, Hiroyuki Kusumoto, Jin Mitsugi, Rodney D. Van Meter, and Kazuki Takashio, Assistant Professors Massaki Sato, Noriyuki Shigechika and Jin Nakazawa as well.

I am extremely thankful for Dr.Takuro Yonezawa, who has always supported and guided me when I has just joined to CPSF research group of Hide.Tokuda Laboratory. He also helped me analyze the meaning of my research, clarify my research goal and guided me to write technique papers.

I would like to thank my advisor, Mr.Takuya Takimoto, who has always taken care of me since I joined Hide.Tokuda Laboratory. He gave me a lot of advice and support when I am not sure about my research motivations. I would like to thank Mr.Tomotaka Ito and Mr.Ogawa Masaki for a listening and giving me many useful discussion points and advices.

I would like to thank members of CPSF research group in Hide.Tokuda Laboratory for their great friendship and support.

I would like to thank Mr.Tomotaka Ito, Mr.Ogawa Masaki, Ms.Vu Le Thao Chi, Ms.Nguyen Thi Ngoc Diep, my friend Ms.Vu Thi Thai Ha, Ms.Luu Thanh Huong, for their comments

for the thesis structure and my English writings.

I send my special thank to my experiment supporters: Takuya Takimoto, Yuuki Nishiyama, Teruaki Ishiguro, Hiroki Shoji, Yutaro Kyono, Nguyen Anh Tien, Do Trung Kien, Nguyen Tien Thanh, Tran Duc Thang Nguyen Thanh Tung, Nguyen Trung Duc, Tran Ngoc Anh and Dinh Hoang Long.

And last but not least, I am heartily grateful to my family. This is the first time I have left my family to start a new life in Japan, many things happened, success and failure, sadness and happiness, tears and smiles. But they still and always are there, giving me strength to follow all goals of my life.

January 22, 2013

Nguyen Doan Minh Giang

Publications

Poster/Demonstration

- Giang Doan Minh Nguyen, Takuya Takimoto, Takuro Yonezawa, Jin Nakazawa, Kazunori Takashio, and Hideyuki Tokuda. 2012. LiDSN: a method to deploy wireless sensor networks securely based on light communication. In Proceedings of the 2012 ACM Conference on Ubiquitous Computing (UbiComp '12). ACM, Pittsburgh, PA, USA, 539-540.
- SFC OPEN RESEARCH FORUM 2012.

Domestic Conference

- Giang Doan Minh Nguyen, Takuya Takimoto, Takuro Yonezawa, Jin Nakazawa, Kazunori Takashio, and Hideyuki Tokuda. HUSTLE: Deploying A Secure Wireless Sensor Network by Light Communication with Smartphone. IPSJ Ubiquitous Computing System 2012, Tokyo, 11 月 2012
- グエン・ゾアン・ミン・ザン，瀧本 拓哉，米澤 拓郎，中澤　仁，高汐 一紀，徳田 英幸．LiDSN：光通信を用いた，セキュアで WSN を構築する手法．情報処理学会 ユビキタスコンピューティング研究会 論文誌，東北大学，宮城県仙台市．7 月 2012.

Award

- Best Paper Award, IPSJ Ubiquitous Computing System (11/2012).

Bibliography

[1] Crossbow, "Mote." `http://www.xbow.com/Products/productdetails.aspx?sid=156`, Nov 2009.

[2] S. Microsystems, "SunSPOT." `http://www.sunspotworld.com/`, Dec 2012.

[3] M. Beigl, C. Decker, A. Krohn, T. Riedel, and T. Zimmer, "μparts: Low cost sensor networks at scale," *Ubicomp: The Seventh International Conference on Ubiquitous Computing*, 2005.

[4] L. Arena Com, "Gsmarena.com - gsm phone reviews, news, opinions, votes, manuals and more...." `http://www.gsmarena.com/`, Dec 2012.

[5] M. A. Simplício, Jr., P. S. L. M. Barreto, C. B. Margi, and T. C. M. B. Carvalho, "A survey on key management mechanisms for distributed wireless sensor networks," *Comput. Netw.*, vol. 54, pp. 2591–2612, October 2010.

[6] B. Lai, S. Kim, and I. Verbauwhede, "Scalable session key construction protocol for wireless sensor networks," 2002.

[7] Z. Alliance, *Zigbee specification document 053474r06*. ZigBee Alliance, 2004.

[8] Y. Zeng, B. Zhao, J. Su, X. Yan, and Z. Shao, "A loop-based key management scheme for wireless sensor networks," in *Emerging Directions in Embedded and Ubiquitous Computing* (M. Denko, C.-s. Shih, K.-C. Li, S.-L. Tsao, Q.-A. Zeng, S. Park, Y.-B. Ko, S.-H.

Hung, and J. Park, eds.), vol. 4809 of *Lecture Notes in Computer Science*, pp. 103–114, Springer Berlin Heidelberg, 2007.

[9] B. Dutertre, S. Cheung, and J. Levy, "Lightweight key management in wireless sensor networks by leveraging initial trust, sdl," 2010.

[10] L. Eschenauer and V. D. Gligor, "A key-management scheme for distributed sensor networks," in *Proceedings of the 9th ACM conference on Computer and communications security*, CCS '02, (New York, NY, USA), pp. 41–47, ACM, 2002.

[11] D. Hwang, B.-C. Lai, and I. Verbauwhede, "Energy-memory-security tradeoffs in distributed sensor networks," in *Ad-Hoc, Mobile, and Wireless Networks* (I. Nikolaidis, M. Barbeau, and E. Kranakis, eds.), vol. 3158 of *Lecture Notes in Computer Science*, pp. 70–81, Springer Berlin Heidelberg, 2004.

[12] R. Blom, "An optimal class of symmetric key generation systems," in *Advances in Cryptology* (T. Beth, N. Cot, and I. Ingemarsson, eds.), vol. 209 of *Lecture Notes in Computer Science*, pp. 335–338, Springer Berlin Heidelberg, 1985.

[13] W. Du, J. Deng, Y. S. Han, and P. K. Varshney, "A pairwise key pre-distribution scheme for wireless sensor networks," in *Proceedings of the 10th ACM conference on Computer and communications security*, CCS '03, (New York, NY, USA), pp. 42–51, ACM, 2003.

[14] J. Lee and D. Stinson, "Deterministic key predistribution schemes for distributed sensor networks," in *Selected Areas in Cryptography* (H. Handschuh and M. Hasan, eds.), vol. 3357 of *Lecture Notes in Computer Science*, pp. 294–307, Springer Berlin Heidelberg, 2005.

[15] Wikipedia, "Infrared communication." http://en.wikipedia.org/wiki/Infrared_communication#Communications, Dec 2012.

[16] Wikipedia, "Bluetooth." http://en.wikipedia.org/wiki/Bluetooth, Dec 2012.

[17] Wikipedia, "Near field communication." http://en.wikipedia.org/wiki/Near_field_communication, Dec 2012.

[18] R. Mayrhofer and H. Gellersen, "Shake well before use: authentication based on accelerometer data," in *Proceedings of the 5th international conference on Pervasive computing*, PERVASIVE'07, (Berlin, Heidelberg), pp. 144–161, Springer-Verlag, 2007.

[19] T. Yonezawa, H. Nakahara, and H. Tokuda, "Vib-connect: A device collaboration interface using vibration," in *Embedded and Real-Time Computing Systems and Applications (RTCSA), 2011 IEEE 17th International Conference*, pp. 121–125, 2011.

[20] L. Selavo, A. Wood, Q. Cao, T. Sookoor, H. Liu, A. Srinivasan, Y. Wu, W. Kang, J. Stankovic, D. Young, and J. Porter, "Luster: wireless sensor network for environmental research," in *Proceedings of the 5th international conference on Embedded networked sensor systems*, SenSys '07, (New York, NY, USA), pp. 103–116, ACM, 2007.

[21] S. Duquennoy, N. Wirström, and A. Dunkels, "Demo: Snap: rapid sensornet deployment with a sensornet appstore," in *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems*, SenSys '11, (New York, NY, USA), pp. 405–406, ACM, 2011.

[22] C. Beckmann, S. Consolvo, A. LaMarca, N. Davies, and E. a. Mynatt, "Some assembly required: Supporting end-user sensor installation in domestic ubiquitous computing environments.," *UbiComp 2004: Ubiquitous Computing*, pp. 107–124, 2004.

[23] H. Baldus, K. Klabunde, and G. Musch, "Reliable set-up of medical body-sensor networks," in *EWSN 2004 : European workshop on wireless sensor networks*, EWSN 2004, pp. 353 – 363[, Springer, 2004.

[24] M. Beigl, A. Krohn, T. Riedel, T. Zimmer, C. Decker, and M. Isomura, "The upart experience: The upart experience," in *Proceedings of the 5th international conference on Information processing in sensor networks*, IPSN '06, (New York, NY, USA), pp. 366–373, ACM, 2006.