# Protecting End-Device from Replay Attack on LoRaWAN

Woo-Jin Sung*, Hyeong-Geun Ahn*, Jong-Beom Kim*, Seong-Gon Choi*

*Information & Communication Engineering, ChungBuk National University, Cheongju-si Chungcheongbuk-do, Korea

sungwoojin@cbnu.ac.kr, gudrhf@cbnu.ac.kr, dragonslash@cbnu.ac.kr, choisg@cbnu.ac.kr

***Abstract*— In this paper, we propose a method to protect end-device by using RSSI and Hand-Shaking technique using Proprietary Message. One of the frequently used attacks in LoRaWAN is replay attack. It is so easy to sniff packets in a wireless network environment. If an attacker intrudes a service provided by LoRaWAN, the usage pattern of the end-device may be exposed, or a replay attack may cause a problem in connection with the user. To prevent replay attack, LoRaWAN standard uses user identification method by using the value known as DevNonce, but this is not a complete countermeasure. In order to complement these vulnerabilities, we propose a method to protect users by using the physical characteristics of a network called RSSI and a new technique called Proprietary Hand-Shaking.**

***Keywords*— DevNonce, Internet of Things, LoRaWAN, Network Security, Proprietary Hand-Shaking, Received Signal Strength Indicator, Replay Attack**

## I. INTRODUCTION

Recently, Internet of Things (IoT) is one of the most popular technologies, and the market is growing gradually and technology is likely to develop. IoT is a technology in which many objects with communication functions are interconnected through a network. Through the network, connected objects can exchange and analyze, learn, and refine the collected data to provide users with useful information and convenient functions [1].

In this paper, we analyzed LoRaWAN technology among various network technologies supporting IoT. LoRaWAN is a wireless communication technology currently emerging in the IoT market that has advantages such as low power, long and wide transmission range, and long-life [2].

However, since LoRaWAN is a wireless communication protocol, data transmitted and received are easily exposed to an attacker [3].

Meanwhile, LoRaWAN provides join procedures that allow end-devices to participate securely in the network and be served. However, on the study associated with join procedures, it is founded a security vulnerability for this process [4]. In the LoRa join procedure, a join request packet contains a random value (DevNonce) to prevent a replay attack. The network server compares the DevNonce of the join request packet previously used by the device with the DevNonce value of the new join request. If a join request is made with a DevNonce of the same value previously used, it determines as a replay attack. Otherwise, it determines as a normal join request. However, since DevNonce is generated as a random value, the previously used DevNonce value can be selected for the normal join request, which may cause the network server to misidentify it as a replay attack.

In order to solve these problems, this paper proposes a new method to protect end-device of the user by using RSSI and Proprietary Hand-Shaking.

The rest of the paper is organized as follows. Section II introduces LoRaWAN technology and research related to vulnerability attack. In Section III, we propose a method to protect user's end-device by using RSSI and newly proposed Proprietary Hand-Shaking method. Section IV explains how to distinguish between attackers and users using the actually measured RSSI values and proposed new methods. Finally, in Section V, we discuss the conclusion.

## II. RELATED WORK

In the LoRaWAN network, the end-device needs to perform a connection activating process in order to communicate with the network server [5]. There are two methods to activate the end-device: Over-the-air activation (OTAA) and Activation-By-Personalization (ABP). The OTAA method activates the end-device by exchanging join request packets and join accept packets between the end-device and the network server.

The vulnerability of replay attack in OTAA join procedure was analyzed [4]. In some cases, an attacker makes a join request for malicious purposes, by using the packets, which have same values by duplicating join request contains AppEUI, DevEUI, and DevNonce values used by the existing user. In other words, if the server receives a replay attack, the server that is using the policy of case B (devices using the same DevNonce are permanently excluded from the network) introduced in [4] may limit the connection of existing normal user's devices. Therefore, it is concluded that using case A (waiting for a valid DevNonce to wait for a join request) instead of case B, but to use a sequential number as DevNonce is more desirable. This prevents the case that DevNonce accidentally matches of existing one, so it can prevent the normal user’s join request packet from being misidentified as a replay attack packet. However, this sequential number method has a problem that an attacker can easily predict DevNonce.

There is also research on how to encrypt DevNonce [6]. Before the end device makes a join request, it uses the

AppKey or session key as a token and generates an encrypted field by XOR operation with DevNonce. The server side also has the same token, which can be used to interpret the encrypted field sent by the end device and finally can get the DevNonce value. This will prevent the attacker from deducing the DevNonce value during the join procedure.

Meanwhile, several types of research have been conducted to detect network attacks using Received Signal Strength Indicator (RSSI) in wireless LAN environment. The RSSI is a measure of the strength of a signal coming from the receive antenna and can be expressed mathematically [7]. This is a value that can vary according to the position of the transmitter and the conditions of the surrounding environment when the position of the receiving antenna is fixed, and it is difficult to forge arbitrarily [8].

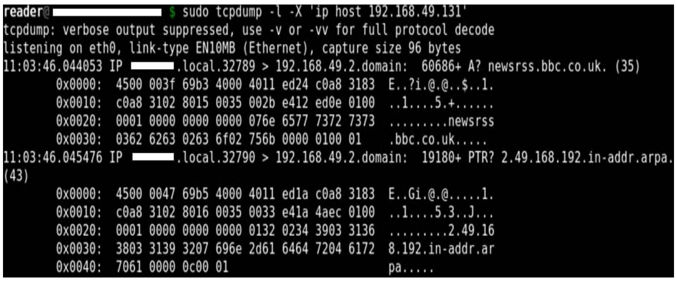
There are several studies that use RSSI to detect spoofing attempts of MAC addresses [9], [10]. Especially in case of [10], RSSI is measured and its physical characteristic is observed when MAC addresses and IP addresses of two packets received in consecutive times are the same. According to the result of this paper, we have found a physical feature that the standard deviation of received RSSI is different according to the distance. Compared with the RSSI of the packets received in the past, the RSSI of the packet sent by the attacker rapidly changes its physical characteristics within a continuous time. Therefore, it can be understood that the attacker has attempted to spoof the MAC address.

## III. Attack Analysis & Protecting Method

In this section, we explain how attackers do sniffing and spoofing from outside the network, and analyze attack scenario that performs replay attack by sniffing join requests which sent by a specific node. Then, we will introduce countermeasures against such attacks.

### A. Sniffing and Spoofing

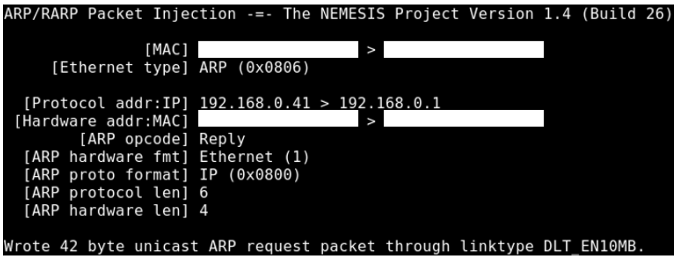
Sniffing is the process of looking at other people's packets. And, faking a MAC address or an IP address that packet possessed is called as spoofing. Both sniffing and spoofing can be easily conducted by an attacker.



```
reader@[redacted] $ sudo tcpdump -l -X 'ip host 192.168.49.131'
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes
11:03:46.044053 IP [redacted].local.32789 > 192.168.49.2.domain:  60686+ A? newsrss.bbc.co.uk. (35)
        0x0000:  4500 003f 69b3 4000 4011 ed24 c0a8 3183  E..?i.@.@..$..1.
        0x0010:  c0a8 3102 8015 0035 002b e412 ed0e 0100  ..1....5.+......
        0x0020:  0001 0000 0000 0000 076e 6577 7372 7373  .........newsrss
        0x0030:  0362 6263 0263 6f02 756b 0000 0100 01    .bbc.co.uk.....
11:03:46.045476 IP [redacted].local.32790 > 192.168.49.2.domain:  19180+ PTR? 2.49.168.192.in-addr.arpa.
(43)
        0x0000:  4500 0047 69b5 4000 4011 ed1a c0a8 3183  E..Gi.@.@.....1.
        0x0010:  c0a8 3102 8016 0035 0033 e41a 4aec 0100  ..1....5.3..J...
        0x0020:  0001 0000 0000 0000 0132 0234 3903 3136  .........2.49.16
        0x0030:  3803 3139 3207 696e 2d61 6464 7204 6172  8.192.in-addr.ar
        0x0040:  7061 0000 0c00 01                        pa.....
```

**Figure 1.** Packet Sniffing

Figure 1 shows packet sniffing performed on Linux Ubuntu Kernel version 2.6.20. Monitoring the packet can be performed by simple commands. And packet sniffing of LoRa packet is shown at [6].



```
ARP/RARP Packet Injection -=- The NEMESIS Project Version 1.4 (Build 26)

               [MAC] [redacted] > [redacted]
     [Ethernet type] ARP (0x0806)

  [Protocol addr:IP] 192.168.0.41 > 192.168.0.1
 [Hardware addr:MAC] [redacted] > [redacted]
        [ARP opcode] Reply
  [ARP hardware fmt] Ethernet (1)
  [ARP proto format] IP (0x0800)
  [ARP protocol len] 6
  [ARP hardware len] 4

Wrote 42 byte unicast ARP request packet through linktype DLT_EN10MB.
```

**Figure 2.** ARP Poisoning for Packet Spoofing

Figure 2 shows the procedure of generating spoofed packets which have an intention of ARP Poisoning attack by using spoofing tools. In this way, packets exchanged between the two clients in communication can be forced to pass via attacker by manipulating the ARP Cache of both sides temporarily. The replay attack introduced in the next clause is similar to these sniffing and spoofing methods.

### B. Analysis of Replay Attack

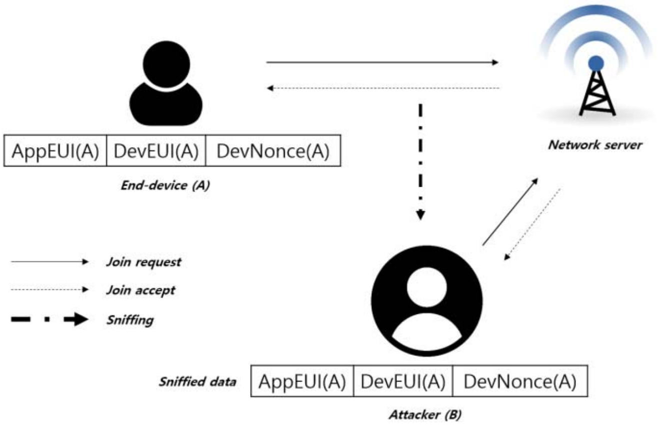


**Figure 3.** Replay Attack

Figure 2 shows a situation that attacker performing a replay attack on a network server by using information which obtained by sniffing packets of end-devices. First, it is assumed that the attacker has the same module as the LoRaWAN module used by the user. It is also assumed that the AppEUI, DevEUI, and DevNonce of the device can be copied and attacked an attacker.

LoRa is basically used in a wireless network environment. When a general user sends a join request to the server, the packet will contain the user's AppEUI, DevEUI, and DevNonce information. Attackers can sniff it and try a replay attack that makes a join request to the server by using the packet, which contains the same information. When the server receives such a request, it will also compare it with the previously stored information. The server will determine that the same DevNonce is used on the same device and will process it according to the policies defined in advance by each manufacturer. Generally, when the same DevNonce comes in, that is, when a user's device receives a replay attack, the established connection of user's device will disconnect.

Network Server also waits until the user makes a join request with a valid DevNonce again [2].

### *C. Proprietary Hand-Shaking*

A policy to block devices using the same DevNonce has proved problematic in [1]. It is desirable to select a policy that waits for a new Request with a valid DevNonce, since a normal user's device may be disabled. However, this method alone cannot distinguish which Request packet is sent by the attacker. In fact, it is almost impossible to identify an attacker by only a join request packet. However, we will use the physical characteristics of network known as RSS to detect that an existing user has been attacked, and suggest ways to prevent the user from being disconnected and protect the user.

We propose that when the server receives a join request, it should store an RSSI in addition to the corresponding DevNonce. So, if AppEUI and DevEUI information match the existing one each time a new request comes in, compare the DevNonce and RSSI values. If the request comes in with the same DevNonce, the server compares the RSSI value additionally.

First, it is necessary to divide the structure of the attack situation into two types as follows.

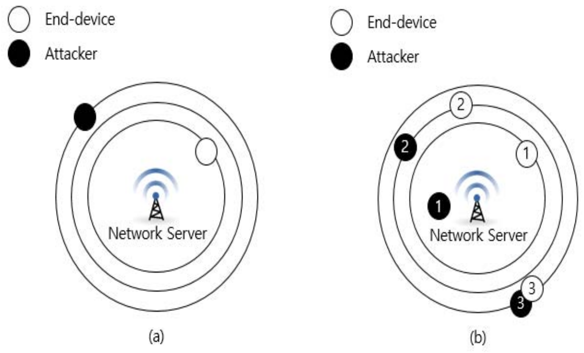


**Figure 4.** (a) Nodes in fixed position, (b) Nodes in moving position with time

#### *a) Nodes in fixed position*

In Figure 4 (a), we assume that end-device and attacker are in fixed position, and the distance from the server is different respectively. The distance from the network server to the circle is assumed 1km, 1.5km, 2km in turn. If the stored RSSI and the newly received RSSI are the same, it is possible to determine that the existing user is a normal user. Therefore, the server needs to allow access again. However, if the RSSI is different, it implies that the sender of the packet is different from the existing user so that the network server can determine that replay attack has been received.

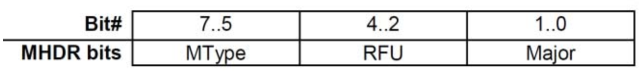
However, the limit of this method is that the server can only determine by using RSSI that the existing user is a normal user, and cannot determine the fact that the attacker is an attacker because the attacker imitates the EUI values as the normal user when attacking. In other words, the identity of the attacker cannot be identified.

#### *b) Nodes in moving position*

By extending the conditions of a), Assume that the location of the end-device and the attacker can change continuously. Figure 3 demonstrates that the location of the end-device and the attacker varying with time. The number of the circles means time t. At time 1, it represents the end-device and the attacker, which are being in different directions and at different distances. At time 2, it represents end-devices and attackers at similar distances but in different directions. At time 3, end-device and attacker have similar directions and similar distances. Since the situation at time 1 can be distinguished only by RSSI, it is processed in the manner described in a). In the situation at time 2, the distances from the network server are similar, so it cannot be distinguished only by RSSI. At time 3, the end-device and the attacker are in the similar direction, and at the similar distance like at time 2, they also cannot be distinguished by RSSI alone.

In this case, since the RSSI value varies from situation to situation, it is difficult to discriminate whether it is a user's join request or an attacker's replay attack with only RSSI.

Receiving a replay attack will also disconnect existing users. Therefore, there is a need to distinguish the attacker from the user while preventing the replay attack from disconnecting the existing user. To solve this problem, we propose a hand-shaking method using proprietary message.

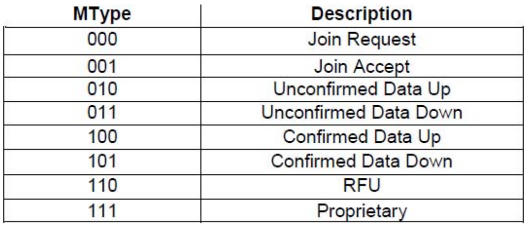


| Bit# | 7..5 | 4..2 | 1..0 |
|---|---|---|---|
| MHDR bits | MType | RFU | Major |

**Figure 5.** MAC Header of LoRa Message

Figure 5 shows the MAC header structure of LoRa. Proprietary message is not a general operation of end-device and server, but it is a message type that can be used by promising each other in advance. The possible values for the MType field are as below.

**TABLE 1.** MAC MESSAGE TYPES OF LORA



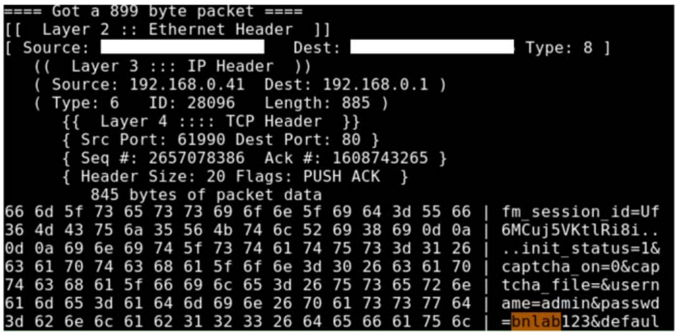
| MType | Description |
|---|---|
| 000 | Join Request |
| 001 | Join Accept |
| 010 | Unconfirmed Data Up |
| 011 | Unconfirmed Data Down |
| 100 | Confirmed Data Up |
| 101 | Confirmed Data Down |
| 110 | RFU |
| 111 | Proprietary |

As shown in Table 1, proprietary is available with the MType set to 111. In case of b, we propose the following method. First, in the case of a user and an attacker, they make a join request while changing their locations, and then server compares the RSSI. In this case, since the position of the user is continuously changed, there is a possibility that the user's join request is misunderstood as the replay attack only by the method described in a). In other words, since it is impossible to judge only by RSSI, the server sends a proprietary message instead of join accept for the join request. On the end-device side, it responds to the proprietary message of network server with the same proprietary message. If the hand-shaking process is defined in advance by the server and the user, it is a special operation between the server and the user. Therefore, the attacker cannot respond to the hand-shaking request because the attacker does not know the defined rule.

In addition, the proprietary message is encrypted so that even if an attacker monitors the message exchange pattern through sniffing, the attacker cannot forge the contents. This allows the server to communicate only with the user's end-device. However, it is necessary to attempt to distinguish users and attackers preferentially through RSSI, because there is a possibility of wasting network resources by executing the process of exchanging proprietary messages for all join requests.

## IV. Node Distinguish Method Based on Experimented RSSI & Proprietary Hand-Shaking

In this section, we will apply the countermeasures against the replay attack that previously described, based on the RSSI data actually experimented.



```
==== Got a 899 byte packet ====
[[  Layer 2 :: Ethernet Header  ]]
[ Source: [redacted]   Dest: [redacted] Type: 8 ]
   ((  Layer 3 ::: IP Header  ))
   ( Source: 192.168.0.41  Dest: 192.168.0.1 )
   ( Type: 6   ID: 28096   Length: 885 )
      {{  Layer 4 :::: TCP Header  }}
      { Src Port: 61990 Dest Port: 80 }
      { Seq #: 2657078386  Ack #: 1608743265 }
      { Header Size: 20 Flags: PUSH ACK  }
         845 bytes of packet data
66 6d 5f 73 65 73 73 69 6f 6e 5f 69 64 3d 55 66 | fm_session_id=Uf
36 4d 43 75 6a 35 56 4b 74 6c 52 69 38 69 0d 0a | 6MCuj5VKtlRi8i..
0d 0a 69 6e 69 74 5f 73 74 61 74 75 73 3d 31 26 | ..init_status=1&
63 61 70 74 63 68 61 5f 6f 6e 3d 30 26 63 61 70 | captcha_on=0&cap
74 63 68 61 5f 66 69 6c 65 3d 26 75 73 65 72 6e | tcha_file=&usern
61 6d 65 3d 61 64 6d 69 6e 26 70 61 73 73 77 64 | ame=admin&passwd
3d 62 6e 6c 61 62 31 32 33 26 64 65 66 61 75 6c | =bnlab123&defaul
```

**Figure 6.** Sniffed Packet by Attacker's Spoofing

Figure 6 shows the attacker using a sniffing and spoofing scheme as described in Section 3, A, in which an attacker obtained a password that someone on the same network used to access the Gateway. Since personal information is easily leaked by an attacker, it is necessary to prevent an attacker from attempting to attack and protect the user.

As a countermeasure against replay attack, a similar attack technique, we propose to use RSSI and Proprietary Hand-Shaking method together.

Figure 7 shows the performance of LoRa measured in an actual urban environment. This experiment measured the RSSI vs. distance under the condition of Spreading Factor = 7 in Non-Line-of-Sight (NLoS) environment [11].

As assumed in Section 3, we have divided the area within 1 km from the Network Server into case 1, the area within 1.5 km from 1 km to case 2, and the area within 2 km from 1.5 km to case 3. Therefore, the situation of time 1, time 2, and time 3 are categorized as case 1, case 2, and case 3, respectively.

Now we can determine which case needs to perform our proposed method.

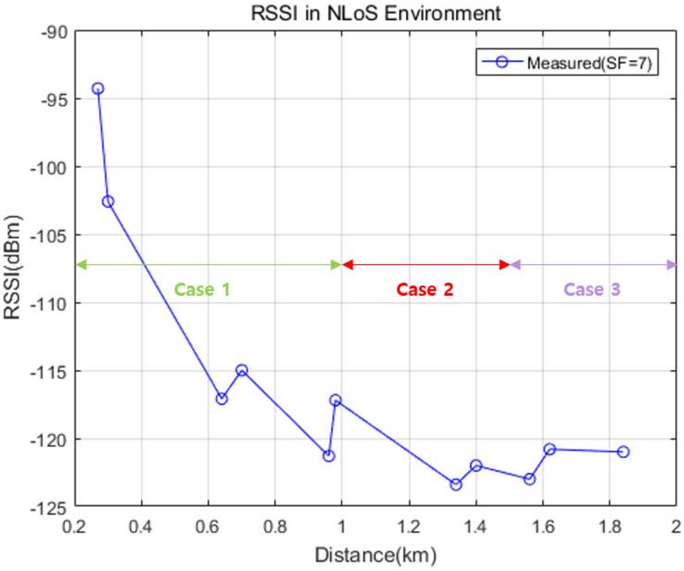


**Figure 7.** RSSI in NLoS Environment(SF = 7)

### A. *Case 1*

In case 1, users and attackers are relatively close to Network Server. In Fig. 5, since the RSSI of case 1 has a large variation range depending on the distance, it is possible to distinguish by using the RSSI used in the join request even if the distance between the end-device and the attacker is small. Therefore, it is not necessary to perform the Proprietary Hand-Shaking process in this case.

### B. *Case 2*

In case 2, the user and the attacker are further away from the Network Server. Especially at time 2, the distance between the end-device and the Attacker's server is the same. Therefore, there is a possibility that RSSI is measured very similarly, and a Proprietary Hand-Shaking process is required in this case.

### C. *Case 3*

In case 3, the user and the attacker are very distant from the Network Server. In particular, in the case of time 3, the end-device and the attacker are located at similar distances in the similar direction. In this case, it is difficult to distinguish the end-device and the attacker with only the RSSI. So it also needs to perform the procedure of Proprietary Hand-Shaking.

In addition, case 3 has a higher RSSI than some of the range in case 2. Overall, the graph is decreasing exponentially. Theoretically, the RSSI value appears to be a neatly decreasing exponential function graph, but this experiment has been done in the real world, so there is a part that is different from the idealistic expectation that the RSSI will decrease as the distance increases (1.3km to 1.4km and 1.6km to 1.8km).

Therefore, it can be said that this result indicates that authentication process such as Proprietary Hand-Shaking is necessary because it is uncertain to distinguish nodes only by RSSI when the distance becomes long.

## V. Conclusions

In this paper, we analyzed the replay attack in the LoRaWAN environment, which can be appeared when the join procedure to the Network Server is performing by end-device, and we proposed countermeasures using RSSI and Proprietary Hand-Shaking. We assumed a specific situation that the attacker is in a fixed position and a changing position, and performs replay attack. Then, we described how to distinguish the user's end-device from the attacker on that situation.

We have emphasized the fact that identifying the nodes only with RSSI has a limit, so the additional Proprietary Hand-Shaking process is required by using the RSSI measurements experiment, which implemented on LoRa environment in the real world.

By applying our proposed methods, it is expected that when users received replay attack, it can be possible to protect the user while maintaining existing connection. However, the proposition of this paper must be verified in practice, and we will leave these tasks as a future work.

## Acknowledgment

This work was supported by the National Research Foundat ion of Korea(NRF) grant funded by the Korea government(M SIP) (No. NRF-2015R1A2A2A03004152). This work was supported by "Human Resources Program in Energy Technology" of the Korea Institute of Energy Technology Evaluation and Planning (KETEP), granted financial resource from the Ministry of Trade, Industry & Energy, Republic of Korea. (No. 20164030201330).

*corresponding author is S.G. Choi (choisg@cbnu.ac.kr)

## References

[1] Hussain Fakhruddin, LoRa & LPWAN: Technology Growth Prospects, Opportunities for 2017 and Beyond, 2017. [Online]. Available: http://teks.co.in/site/blog/lora-iot-growth-prospects-opportunities-for-2017-and-beyond/

[2] Lizhe Wang, Rajiv Ranjan, "Processing Distributed Internet of Things Data in Clouds", *IEEE Cloud Computing*, 2015.

[3] R. Miller, LoRa Security - Building a Secure LoRa Solution, *MWRLabs,* 2017. [Online].Available: https://labs.mwrinfosecurity.com /assets/BlogFiles/mwri-LoRa-security guide-1.2-2016-03-22.pdf.

[4] Stefano Tomasin, Simone Zulian, Lorenzo Vangelista, "Security Analysis of LoRaWAN Join Procedure for the Internet of Things Networks", 2017 IEEE Wireless Communications and Networking Conference Workshops (WCNCW), 2017.

[5] N. Sornin, M. Luis, T. Eirich, T. Kramp, O.Hersent, "LoRaWAN Specification", 2016.

[6] SeungJae Na, DongYeop Hwang, WoonSeob Shin, Ki-Hyung Kim, "Scenario and Countermeasure for Replay Attack Using Join Request Messages in LoRaWAN", 2017 International Conference on Information Networking (ICOIN), 2017.

[7] Jan Blumenthal, Ralf Grossmann, Frank Golatowski, Dirk Timmermann, "Weighted Centroid Localization in Zigbee-based Sensor Networks", 2007 IEEE International Symposium on Intelligent Signal Processing, 2007.

[8] Sudip Misra, Ashim Ghosh, A. P. Sagar P., Mohammad S. Obaidat, "Detection of Identity-Based Attacks in Wireless Sensor Networks Using Signalprints", Green Computing and Communications (GreenCom), 2010 IEEE/ACM Int'l Conference on & Int'l Conference on Cyber, Physical and Social Computing (CPSCom), 2010.

[9] Y. Sheng, K. Tan, G. Chen, D. Kotz, A. Campbell, "Detecting 802.11 MAC Layer Spoofing using Received Signal Strength", IEEE INFOCOM 2008 - The 27th Conference on Computer Communications, 2008.

[10] Jaegwan Yu, Eunsoo Kim, Hyoungshick Kim, Junho Huh, "A framework for detecting MAC and IP spoofing attacks with network characteristics", 2016 International Conference on Software Security and Assurance (ICSSA), 2016.

[11] Dong Hee Yi, Suk Chan Kim, "Analysis of Computer-Simulated and Field Experimental Results of LoRa Considering Path Loss under LoS and NLoS", The Journal of Korean Institute of Communications and Information Sciences, 2017.

**Woo-Jin Sung** is currently a B.S. & M.S. candidate in School of Electrical & Computer Engineering, Chungbuk National University, Korea in 2017. His research interest is network security.

**Hyeong-Geun Ahn** is currently a B.S. & M.S. candidate in School of Electrical & Computer Engineering, Chungbuk National University, Korea in 2017. His research interest is embedded programming.

**Jong-Beom Kim** received B.S. degree in the College of Electrical & Computer Engineering, Chungbuk National University, Korea in 2017. He is currently a M.S. candidate in School of Electrical & Computer Engineering, Chungbuk National University. His research interest is network programming.

**Seong-Gon Choi** received B.S. degree in Electronics Engineering from Kyeongbuk National University in 1990, and M.S. and Ph.D. degree from KAIST, Korea in 1999 and 2004, respectively. He is currently a professor in the College of Electrical & Computer Engineering, Chungbuk National University. His research interests include smart grid, IoT, mobile communication, high-speed network architecture and protocol.