Implementation and Evaluation of Mutual Authentication Protocol for Artificial Pancreas System

Hoseok Kwon[1], Youngsin Park[2], Jiyoon Kim[3], Ilsun You[2]

[1]Financial Information Security, Kookmin University, Seoul, Republic of Korea

[2]Information Security and Cryptography, Kookmin University, Seoul, Republic of Korea

[3]School of Computer Science, Gyeonsang National University, Seoul, Republic of Korea

Received: January 10, 2024; Revised: March 05, 2024; Accepted: April 10, 2024; Published: April 25, 2024

Abstract

Bluetooth Low Energy (BLE) is an international standard protocol widely used in IoT (Internet of Things) environments due to its low power consumption. However, when BLE is used in an Artificial Pancreas System (APS), it faces the threat of Man-In-The-Middle (MITM) attacks. These attacks can lead to eavesdropping on sensitive medical information or manipulation of insulin injection commands, potentially endangering the patient's life. In this paper, we introduce a protocol that mitigates the vulnerability of BLE to MITM attacks and enhances resilience to sudden situations that may occur in APS environments. Finally, we evaluate the implementation suitability by deploying this protocol in a test environment.

Keywords: APS, BLE, IoT, Security Protocol.

1 Introduction

The Artificial Pancreas System (APS) is a closed-loop medical system that consists of three major components: a Continuous Glucose Monitor (CGM), an Insulin Pump (IP), and a Controller. The CGM monitors blood glucose levels and transmits the measured data to the Controller. Based on the data displayed on the controller, patients can send a command to the IP to inject the appropriate amount of insulin into the body [1]. CGM and IP, collectively referred to as Implanted Medical Devices (IMDs), operate by being inserted into or attached to the patient’s body. IMDs perform sensitive operations close to the patient’s body but are constrained by battery life, processor performance, and other factors [2].

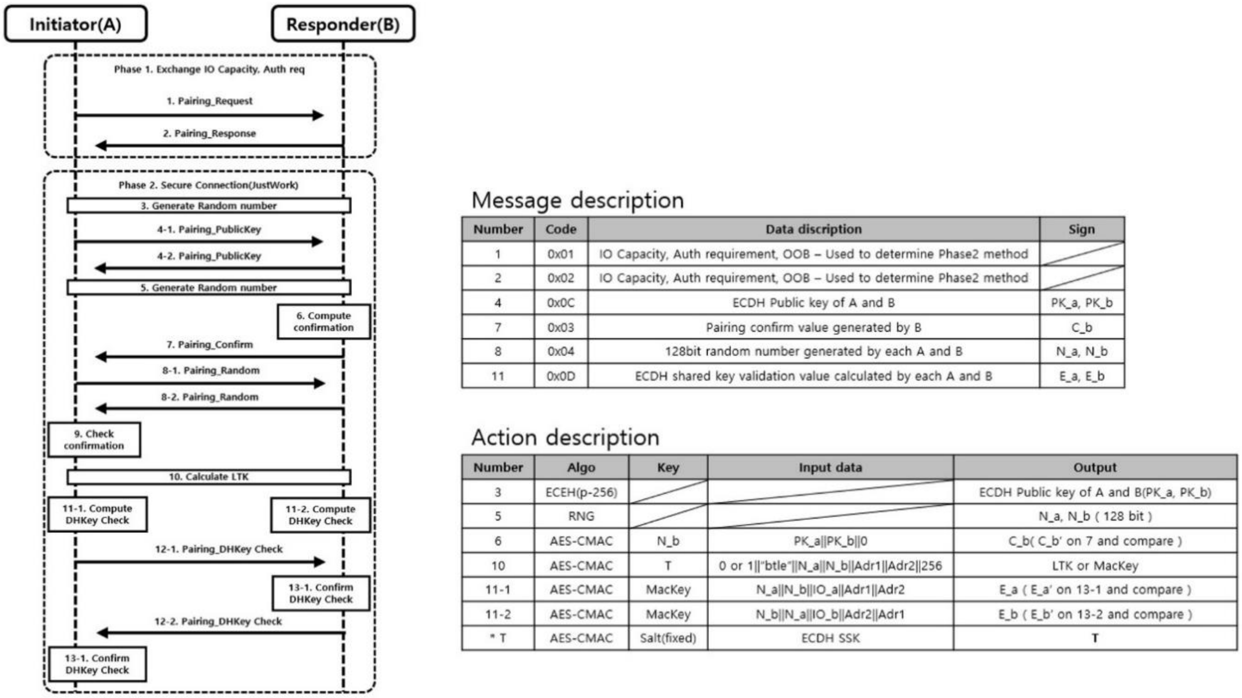
Bluetooth Low Energy (BLE) is a communication protocol designed for use on the Internet of Things (IoT) environment. BLE is also used in commercial APS products, which utilize IMDs that broadly fall under the IoT category. However, BLEs pairing protocols do not support device authentication for devices without input or output mechanisms.

This paper analyzes the BLE pairing process and describes Man-In-The-Middle (MITM) attacks that can occur during BLE pairing. Furthermore, we introduce a proposed protocol specialized for the APS environment that overcomes the vulnerabilities of the BLE protocol. Finally, we implement the proposed protocol in a testing environment to compare and evaluate it against the BLE protocol.

DOI:https://doi.org/10.56801/rebicte.v10i.186

2 Background

BLE is a short-range communication standard protocol established by the Bluetooth Special Interest Group [3]. Figure 1 illustrates the pairing process that performs BLE key exchange and integrity verification. The pairing process consists of two phases. In Phase 1, Authentication Requirement and IO capability are exchanged, and based on exchanged capacities the method to be used in Phase 2 are determined. IO capability, which refers to the existence of input/output mechanism on IoT devices, classifies BLE device to five categories: NoInputNoOutput, KeyboardOnly, DisplayOnly, DisplayYesNo, and KeyboardDisplay. CGM and IP are associated to NoInputNoOutput, and the controller to Keyboard Display. Under these conditions, the method used in Phase 2 is the Just Work method.


Initiator(A)
Responder(B)
Phase 1. Exchange IO Capacity, Auth req
1. Pairing_Request
2. Pairing_Response
Phase 2. Secure Connection(JustWork)
3. Generate Random number
4-1. Pairing_PublicKey
4-2. Pairing_PublicKey
5. Generate Random number
6. Compute confirmation
7. Pairing_Confirm
8-1. Pairing_Random
8-2. Pairing_Random
9. Check confirmation
10. Calculate LTK
11-1. Compute DHKey Check
11-2. Compute DHKey Check
12-1. Pairing_DHKey Check
13-1. Confirm DHKey Check
12-2. Pairing_DHKey Check
13-1. Confirm DHKey Check

Message description

Number	Code	Data discription	Sign
1	0x01	IO Capacity, Auth requirement, OOB – Used to determine Phase2 method	
2	0x02	IO Capacity, Auth requirement, OOB – Used to determine Phase2 method	
4	0x0C	ECDH Public key of A and B	PK_a, PK_b
7	0x03	Pairing confirm value generated by B	C_b
8	0x04	128bit random number generated by each A and B	N_a, N_b
11	0x0D	ECDH shared key validation value calculated by each A and B	E_a, E_b

Action description

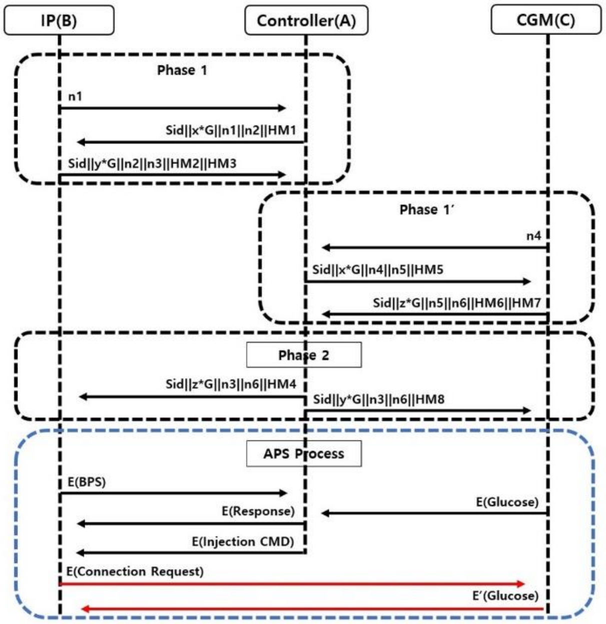
Number	Algo	Key	Input data	Output
3	ECEH(p-256)			ECDH Public key of A and B(PK_a, PK_b)
5	RNG			N_a, N_b (128 bit)
6	AES-CMAC	N_b	PK_a||PK_b||0	C_b(C_b' on 7 and compare)
10	AES-CMAC	T	0 or 1||"btle"||N_a||N_b||Adr1||Adr2||256	LTK or MacKey
11-1	AES-CMAC	MacKey	N_a||N_b||IO_a||Adr1||Adr2	E_a (E_a' on 13-1 and compare)
11-2	AES-CMAC	MacKey	N_b||N_a||IO_b||Adr2||Adr1	E_b (E_b' on 13-2 and compare)
* T	AES-CMAC	Salt(fixed)	ECDH SSK	T

Figure 1. BLE protocol diagram.

BLE Just Work performs security protocols in the pairing process as illustrated in Figure 1, but it is vulnerable to man-in-the-middle attacks. This vulnerability arises from the use of ECDH(Elliptic Curve Diffie-Hellman) key exchange without additional authentication method [2]. For this reason, the standard document also classifies Just Work method as Unauthenticated. To safely communicate between two node mutual authentication are crucial [4]. The protocol proposed in the following chapter overcomes these vulnerabilities and adds techniques to deal with emergency situations that may occur in the APS environment. Thereby providing a more resilient and stable APS service to the patients.

3 Proposed protocol

In the proposed protocol, denoted as Figure 2, the entities are assigned as follows. A represents the controller, B and C each stands for the IP and CGM. The figure 2 encompasses both the pairing procedure and the communication process. In the proposed protocol, an additional step of authenticating the peer using a pre-exchanged secret value Pwd has been added. In Phase 1 of the pairing process, HM1 and HM3 are HMAC[5] values using this Pwd as the key. Through mutual authentication, this process verifies the integrity of the pairing messages and authenticates the peer. This allows IP and CGM to trust the controller. Subsequently, in Phase 2, the trusted controller is used as a proxy to exchange parameters for the security channel between IP and CGM.



- Pre-shared secret
Pwd(Between IP and Con)
Pwd′(Between CGM and Con)

n1, n2, n3, n4, n5, n6 : Random number
Sid : Session ID

- Session Keys
SK_IP : HMAC(x*y*G, n2||n3)
SK_CGM : HMAC(x′*z*G, n5||n6)
SK_EM : HMAC(y*z*G, n3||n6)

- Phase1(HMAC)
HM1 : HMAC-SHA256(Pwd, Sid||x*G||n1||n2)
HM2 : HMAC-SHA256(SK_IP, Sid||y*G||n2||n3)
HM3 : HMAC-SHA256(Pwd, Sid||y*G||n2||n3||HM2)
HM5 : HMAC-SHA256(Pwd′, Sid||z*G||n5||n6)
HM6 : HMAC-SHA256(SK_CGM, Sid||z*G||n5||n6)
HM7 : HMAC-SHA256(Pwd′, Sid||z*G||n5||n6||HM6)

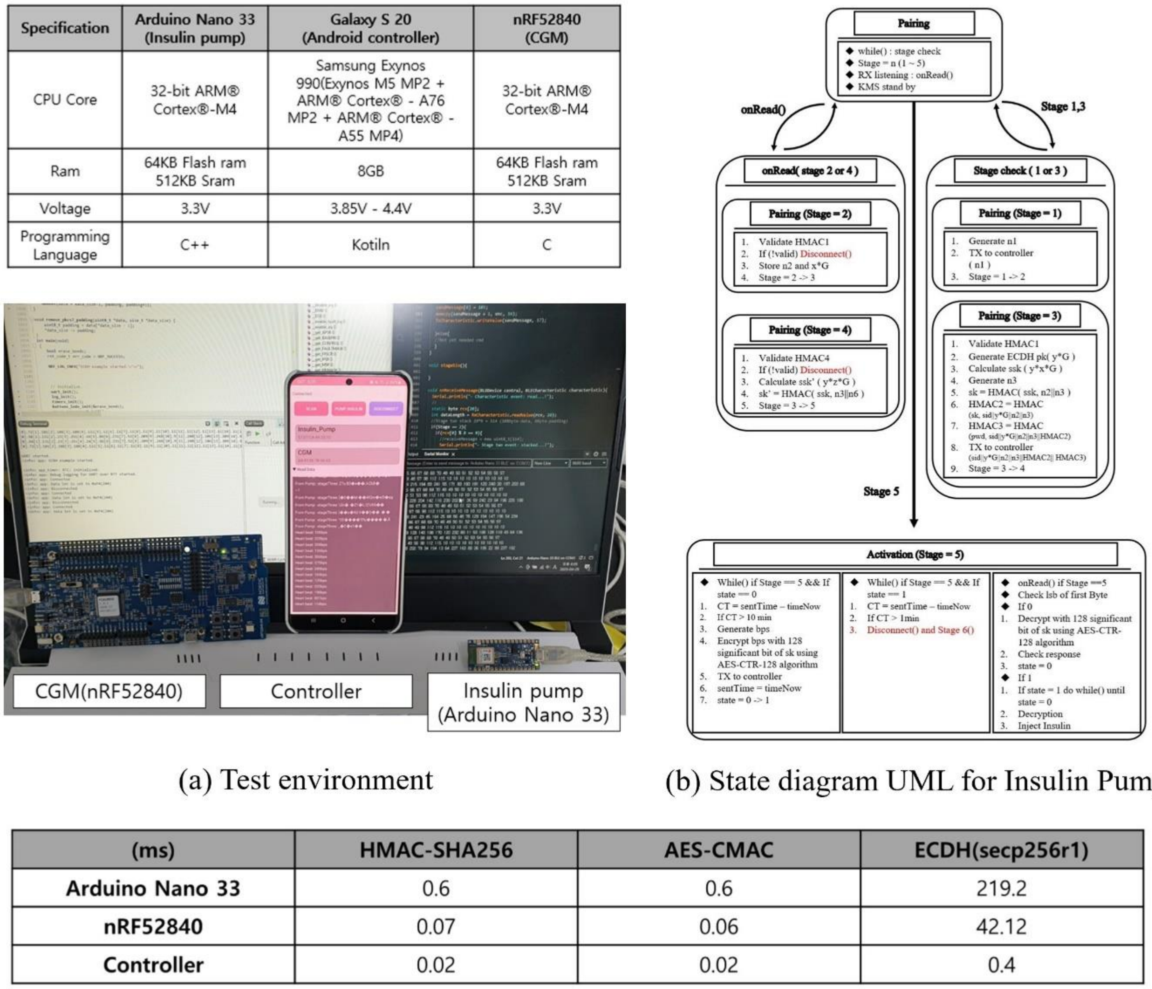
- Phase2(HMAC)
HM4 : HMAC-SHA256(SK_IP, Sid||z*G||n3||n6)
HM8 : HMAC-SHA256(SK_CGM, Sid||y*G||n3||n6)

Figure 2. Proposed protocol diagram.

Through the pairing process of the proposed protocol IP, CGM, and the controller have established a mutual security channel. This enables secure communication between IMDs via the secure channel between IP and CGM even in situations where the controller becomes disabled. In the absence of the controller, CGM directly transmits the blood glucose levels to IP, and IP interprets the glucose levels and manages the patient's insulin dosage [6].

4 Test

To evaluate the feasibility of the proposed protocol, we have established an APS test environment and implemented it. The test environment is depicted in Figure 3 (a). The proposed protocol was implemented, and computational overhead were tested in the nRF52840[7], Arduino nano 33[8], and Galaxy S20, each acts as CGS, IP and Controller in APS system. However, the primary objective of this protocol proposal is to replace the BLE layer. At the current stage, the protocol was implemented in a cross-layer format for ease of implementation, operating once again at the application layer after establishing a BLE connection. Figure 3 (b) is a state diagram of the proposed protocol for an insulin pump. It includes specifications for the pairing process, as described earlier, and continuous normal operation, represented by "Activation (Stage = 5)". In this stage, three main actions are performed. The most periodic action is measuring the heart rate (bps - beats per second) every 10 minutes and transmitting it to the controller. After this action, the IP must receive a response within 1 minute for controller’s state confirmation. To achieve this, the time of the last bps transmission is saved, and the state is set to "state = 1" indicating that the bps was sent but no response was received. The time spent in this state is then measured, and if no response is received within 1 minute, it is assumed that the controller is not operating properly. In such a case, the connection is terminated, and a direct connection to the CGM is established. The final action is executing the insulin injection command sent by the controller. However, if the state is still "state = 1" indicating that the response to the bps transmission has not been received, the insulin is injected only after receiving the response. Table 1 below presents the computation times for ECDH[9], AES-CMAC[10], and HMAC-SHA256 when performed in the controller, IP, and CGM environments.



Specification	Arduino Nano 33 (Insulin pump)	Galaxy S 20 (Android controller)	nRF52840 (CGM)
CPU Core	32-bit ARM® Cortex®-M4	Samsung Exynos 990(Exynos M5 MP2 + ARM® Cortex® - A76 MP2 + ARM® Cortex® - A55 MP4)	32-bit ARM® Cortex®-M4
Ram	64KB Flash ram 512KB Sram	8GB	64KB Flash ram 512KB Sram
Voltage	3.3V	3.85V - 4.4V	3.3V
Programming Language	C++	Kotiln	C

CGM(nRF52840)
Controller
Insulin pump (Arduino Nano 33)

(a) Test environment

Pairing
◆ while() : stage check
◆ Stage = n (1 ~ 5)
◆ RX listening : onRead()
◆ KMS stand by
onRead()
Stage 1,3
onRead(stage 2 or 4)
Pairing (Stage = 2)
1. Validate HMAC1
2. If (!valid) Disconnect()
3. Store n2 and x*G
4. Stage = 2 -> 3
Pairing (Stage = 4)
1. Validate HMAC4
2. If (!valid) Disconnect()
3. Calculate ssk' (y*z*G)
4. sk' = HMAC(ssk, n3||n6)
5. Stage = 3 -> 5
Stage check (1 or 3)
Pairing (Stage = 1)
1. Generate n1
2. TX to controller (n1)
3. Stage = 1 -> 2
Pairing (Stage = 3)
1. Validate HMAC1
2. Generate ECDH pk(y*G)
3. Calculate ssk (y*x*G)
4. Generate n3
5. sk = HMAC(ssk, n2||n3)
6. HMAC2 = HMAC (sk, sid||y*G||n2||n3)
7. HMAC3 = HMAC (pwd, sid||y*G||n2||n3||HMAC2)
8. TX to controller (sid||y*G||n2||n3||HMAC2|| HMAC3)
9. Stage = 3 -> 4
Stage 5
Activation (Stage = 5)
◆ While() if Stage == 5 && If state == 0
1. CT = sentTime – timeNow
2. If CT > 10 min
3. Generate bps
4. Encrypt bps with 128 significant bit of sk using AES-CTR-128 algorithm
5. TX to controller
6. sentTime = timeNow
7. state = 0 -> 1
◆ While() if Stage == 5 && If state == 1
1. CT = sentTime – timeNow
2. If CT > 1min
3. Disconnect() and Stage 6()
◆ onRead() if Stage ==5
◆ Check lsb of first Byte
◆ If 0
1. Decrypt with 128 significant bit of sk using AES-CTR-128 algorithm
2. Check response
3. state = 0
◆ If 1
1. If state = 1 do while() until state = 0
2. Decryption
3. Inject Insulin

(b) State diagram UML for Insulin Pump

(ms)	HMAC-SHA256	AES-CMAC	ECDH(secp256r1)
Arduino Nano 33	0.6	0.6	219.2
nRF52840	0.07	0.06	42.12
Controller	0.02	0.02	0.4

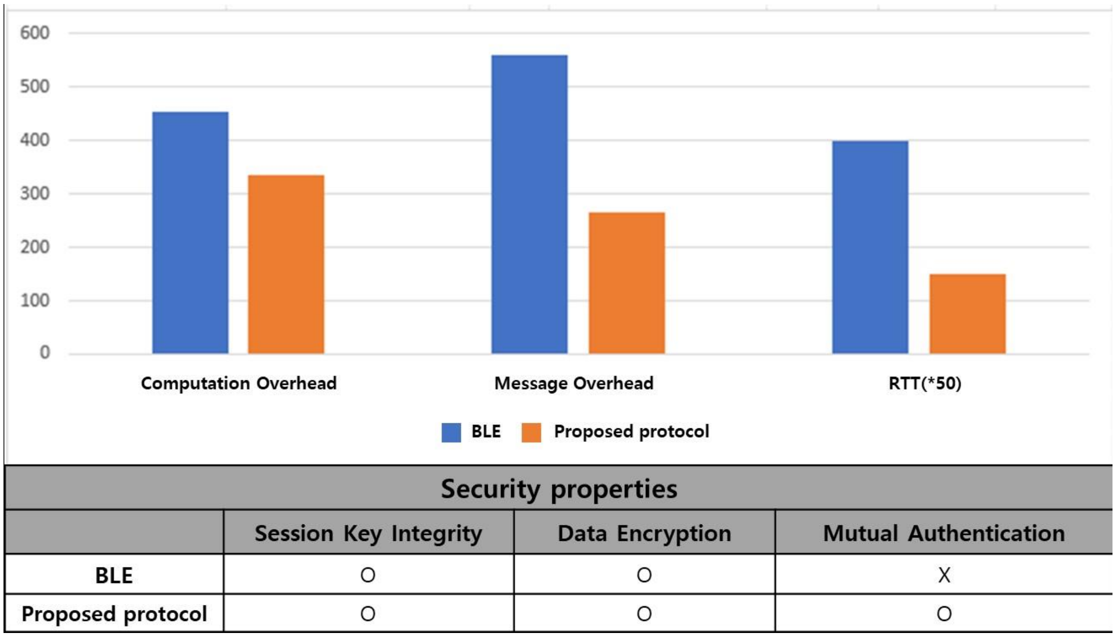
(c) Computation overhead

Figure 3. Experiment.

As shown in Figure 3 (c), the computation time in IMD is 100 to 500 times greater compared to the controller, with ECDH computation serving as the benchmark for this significant difference. This indicates that the protocol significantly influences performance, particularly in the IMD rather than the controller. Thus, for the assessment of implementation suitability, protocol comparison experiments were conducted using the Arduino Nano 33 board.

To ensure a fair comparison between the two protocols, it is necessary to standardize the conditions. BLE operates on a one-to-one pairing basis, whereas the proposed protocol establishes practical security channels between one controller and two IMDs. To mimic the structure of the proposed protocol using BLE, after the controller establishes a BLE connection with the IMDs, the IP must establish another BLE connection with the CGM, resulting in a total of two BLE pairings from the IP's perspective.

Figure 4 presents a comparison of overhead and security properties between BLE and the proposed protocol. Both BLE and the proposed protocol ensure the integrity of the session key and data encryption. However, the proposed protocol additionally provides mutual authentication as a security property. Nonetheless, as evident from the experimental results, the computational overhead of the proposed protocol is lower than that of BLE. These experimental results were obtained by simulating an arbitrary pairing process using a combination of cryptographic algorithms used in the protocol. In addition to computational overhead, the proposed protocol also has an advantage over BLE in terms of message overhead, which refers to the size of the messages exchanged during the pairing process, as well as Round Trip Time (RTT).



Security properties			
	Session Key Integrity	Data Encryption	Mutual Authentication
BLE	O	O	X
Proposed protocol	O	O	O

Figure 4. Comparison of security properties and protocol overhead

5 Conclusion

Lack of authentication provides no integrity [11]. When configuring APS using BLE, it is vulnerable to man-in-the-middle attacks due to lack of authentication and not being able to verify the integrity of messages that are exchanged. This not only risks the theft of patient information but also can be used to threaten the patient's life with malicious commands. Therefore, the proposed protocol addresses these concerns by achieving more significant security requirements with 35.7% less computational overhead compared to BLE in the same scenario. Additionally, it is designed to handle emergency situations that may arise during the actual operation of APS. By implementing this protocol in a real environment, practical experiments were conducted to compare the actual computational overhead rather than simply comparing algorithmic computation counts. Through these experiments, it was confirmed that the proposed protocol performs the pairing process faster while satisfying more security properties.

Funding Details

This work was supported by Institute of Information \& communications Technology Planning \&

Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2020-0-00952, Development of 5G Edge Security Technology for Ensuring 5G+ Service Stability and Availability, 100%).

Disclosure Statement

The author declares that (s)he has no relevant or material financial interests that relate to the research described in this paper.

Notes on Contributors

Hoseok Kwon: Graduated Information Security, Cryptology, and Mathematics from Kookmin University, South Korea. And currently undergraduate course in Financial Information Security, Kookmin University. His research interests are network security and machine learning.

Youngsin Park: Graduated in Information Security, Cryptology, and Mathematics from Kookmin University, South Korea. Her research interests are 5G/6G security and machine learning.

Jiyoon Kim: Received the M.S. and Ph.D. degrees in information security engineering from Soonchunhyang University, Asan, South Korea. His current research interests include mobile Internet security, 5G security, and formal security analysis.

Ilsun You: Received the M.S. and Ph.D. degrees in computer science from Dankook University, Seoul, South Korea, in 1997 and 2002, respectively, and the second Ph.D. degree from Kyushu University, Japan, in 2012. From 1997 to 2004, he was with the THIN multimedia Inc., Internet Security Co., Ltd., and Hanjo Engineering Co., Ltd. as a Research Engineer. He is currently a Full Professor at the department of Information Security, Cryptography and Mathematics, Kookmin University. he has focused on 4/5G security, security for wireless networks & mobile internet, IoT security and so forth while publishing more than 180 papers in these areas.

References

[1] Astillo, P. V., Jeong, J., Chien, W. C., Kim, B., Jang, J., & You, I. (2021). SMDAps: A specification-based misbehavior detection system for implantable devices in artificial pancreas system. Journal of Internet Technology, 22(1), 1-11.

[2] Duguma, D. G., You, I., Gebremariam, Y. E., & Kim, J. (2021). Can Formal Security Verification Really Be Optional? Scrutinizing the Security of IMD Authentication Protocols. Sensors, 21(24), 8383.

[3] Bluetooth SIG. Bluetooth core specification v5.4. Technical report, 2023.

[4] Liu, P., Liu, B., Sun, Y., Zhao, B., & You, I. (2018). Mitigating DoS attacks against pseudonymous authentication through puzzle-based co-authentication in 5G-VANET. IEEE Access, 6, 20795-20806.

[5] National Institute of Standards and Technology (NIST), FIPS 198-1 The Keyed-Hash Message Authentication Code (HMAC), 2008

[6] Jiyoon Kim, Jongmin Oh, Daehyeon Son, Hoseok Kwon, Philip Virgil Astillo, and Ilsun You. Apsec1. 0: innovative security protocol design with formal security analysis for the artificial pancreas system. Sensors, 23(12):5501, 2023..

[7] Nordic semiconductor. nRF52840 specification v1.8. Product specification, 2023.

[8] Arduino. Arduino Nano 33 BLE. Product specification, 2024.

[9] National Institute of Standards and Technology (NIST), SP800 56A Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography, 2013

[10] Tetsu Iwata, Junhyuk Song, Jicheol Lee, and Radha Poovendran. The AES-CMAC Algorithm. RFC 4493, June 2006.

[11] Dave, G., Choudhary, G., Sihag, V., You, I., & Choo, K. K. R. (2022). Cyber security challenges in aviation communication, navigation, and surveillance. Computers & Security, 112, 102516