Security Analysis of Network Slicing-based Mission-Critical Services with Formal Verification Tool

KyeongA Kang, Seungbin Lee, Jiyoon Kim

Gyeongsang National University, Jinju, Republic of Korea

Received: October 17, 2024; Revised: November 18, 2024; Accepted: December 20, 2024; Published: December 29, 2024

Abstract

Mission Critical Service is a service requiring high reliability and low latency, utilized in various fields such as automotive safety systems, disaster safety communications, and medical services. In particular, group communication is essential in disaster scenarios, where the secure delivery of information and user reliability are critical factors. This paper focuses on the user authentication and service authorization procedures for Mission Critical X (MCPTT, MCVideo, MCData), the core components of MCS. The OpenID Connect 1.0 protocol is employed as a key framework for authentication and identity verification, while the TLS protocol ensures the confidentiality and integrity of data. The application procedures of these protocols are elaborated, and the security of the TLS (Transport Layer Security) protocol used in Mission Critical X within the Mission Critical Service environment is analyzed using the formal verification tool BAN-Logic.

Keywords: Mission Critical Services, Network Slicing, Formal Verification.

1 Introduction

MCS (Mission Critical Service) refers to fields requiring high reliability, such as automotive safety systems, air traffic control, and medical services. These services must ensure that connections are not interrupted or that latency does not increase significantly. MCS requires high system availability, with latency typically below 1ms, and emphasizes reliability in terms of coverage. MCX stands for Mission Critical X, where X represents MCPTT, MCVideo, or MCData. The MCPTT service supports group communication among multiple users, where each user can request permission to speak through designated communication mechanisms [1]. MCVideo supports video communication (group calls) among multiple users, where each user can obtain speaking rights through coordinated mechanisms [2]. MCData supports one-to-one communication as well as communication among multiple users [3]. MCX (MCPTT, MCVideo, MCData) is deployed in infrastructure-based solutions for areas lacking connectivity, such as disaster zones or remote locations like forests, where communication is essential [4].

In disaster safety-related tasks, missions are mostly performed in groups, making it essential to share commands and mission-related information through group communication in disaster situations [5]. In emergency situations such as disasters, the information transmitted via MCS can directly impact users' lives and property, making security critically important. This paper examines the security of OpenID Connect and TLS (Transport Layer Security) used in MCS environments using formal verification tools.

2 Related Research

2.1 OAuth 2.0

The OAuth 2.0 authorization framework enables third-party applications to obtain limited access to HTTP services. It either facilitates interaction between the resource and the HTTP service to allow authorization on behalf of the resource owner or enables third-party applications to obtain access rights independently [6]. Key use cases of OAuth 2.0 include social login, granting access permissions, and controlling API access for third-party applications.

2.2 OpenID

OpenID Connect 1.0 is an identity layer built on top of OAuth 2.0. OpenID Connect enables clients to substantiate the identity of the end user based on the authentication performed by an authorization server and to get basic profile information about the end user in an interoperable manner using REST-like APIs. OpenID Connect 1.0 defines features of OpenID Connect, built on OAuth 2.0, including authentication and the use of claims to convey information about the end user [7].

2.3 TLS

The main goal of the TLS protocol is to ensure confidentiality and data integrity between two communicating applications. This protocol enables client applications and server applications to communicate in a way that prevents message, forgery eavesdropping, or tampering [8].

3 Main Discussion

3.1 MCX Security

The security architecture of the MC system provides security functions between MC clients, between MC clients and the MC domain, and between MC domains. Before accessing most MC services, user authentication and authorization to the MC domain are required.

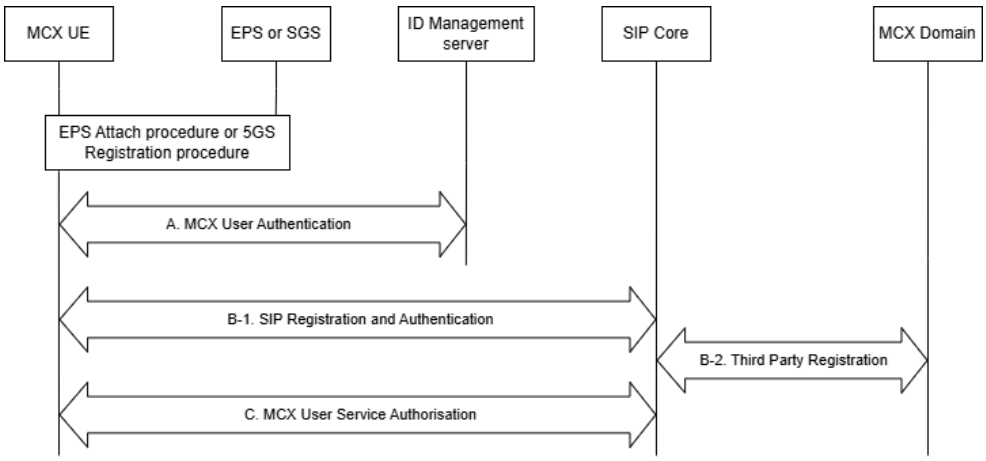

Figure 1. MCX Authentication and Authorization Procedures

Figure 1 is a flowchart illustrating the authentication and authorization of the MCX service. When the UE is powered on, the MCX UE performs EPS UE authentication defined in TS 33.401 [9] or 5GS UE authentication defined in TS 33.501 [10], depending on the system. The MCX UE performs the following steps to complete user authentication, user authorization, MCX service registration, and identity binding between

the signaling layer identity and the MC service ID.

- A: MCX User Authentication

- B: SIP Registration and Authentication

- C: MCX User Service Authorization

This paper aims to conduct a security analysis of the authentication process in stages A and C.

3.2 MCX User Authentication

MCX user authentication, MCX user service authorization, OpenID Connect 1.0, and the OpenID Connect profile for MCX form the foundation of the identity management architecture. The HTTP connection between the identity management client and the identity management server must be protected using HTTPS. User authentication procedure in step A of Figure 1 is detailed in three sub-steps, which are part of the MCX user authentication framework.

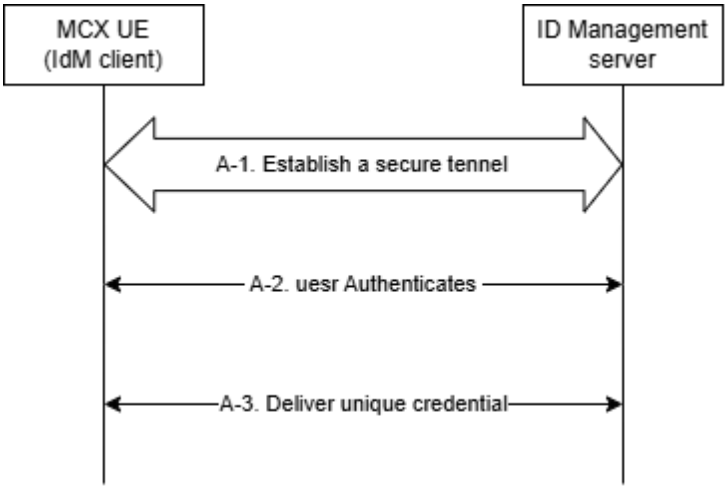

Figure 2. MCX User Authentication Framework

- A-1: A secure tunnel is established between the MCX UE and the Identity Management (IdM) server, which is then used in subsequent steps.

- A-2: The user authentication process is performed.

- A-3: The credentials that uniquely identify the MCX user are transmitted to the IdM client.

After completing step A-3, MCX client uses credentials obtained in step A-3 to perform MCX user service authorization according to step C in Figure 1. The framework supporting steps A-2 and A-3 is implemented using OpenID Connect 1.0 [6],[7],[11].

3.3 OpenID Connect Protocol

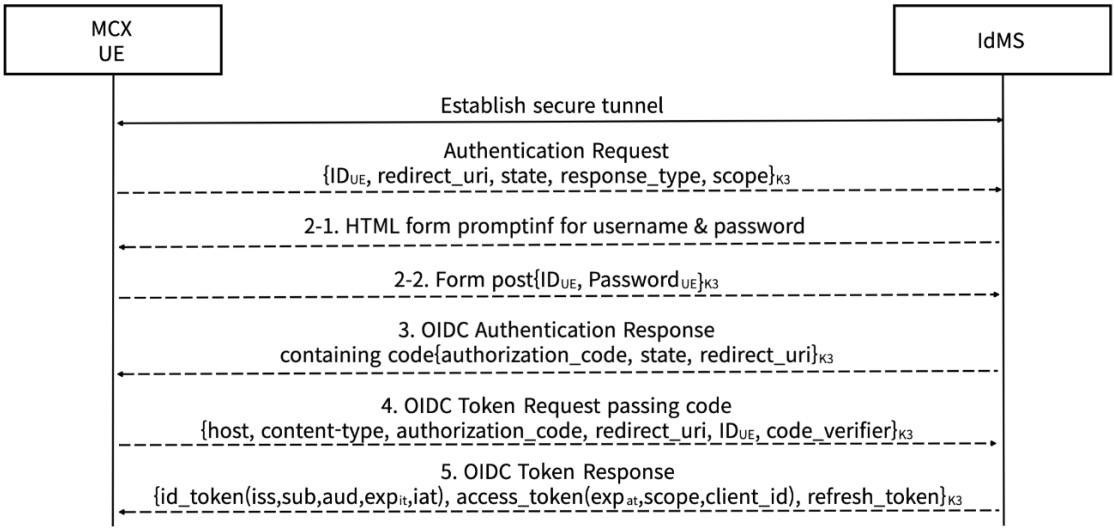

Figure 3. OpenID Connect (OIDC) Flow Diagram Supporting MCX User Authentication

Figure 3 explains MCX user authentication framework using the OpenID Connect protocol. Specifically, it describes the step in which the MCX user authenticates with the Identity Management (IdM) server and transfers the credential set that uniquely identifies the MC service ID to the UE. In this process, the essential authentication methods, such as username and password, are used, and a secure communication channel is established between the MCX UE and the Identity Management server (IdMS).

- Step 1: The UE sends an OpenID Connect authentication request to the IdMS.

- Step 2-1: The IdMS sends an HTML form to the UE requesting the username and password.

- Step 2-2: The UE sends username and password provided by user to the IdMS.

- Step 3: The IdMS sends the OIDS authentication response, which includes the authorization code, to the UE.

- Step 4: The UE sends the authorization code to the IdMS and submits the OIDC token request.

- Step 5: The IdMS sends the OIDC token response to the UE, including the ID token and access token.

The tokens uniquely identify the users of the MCX service. The UE uses the ID token to personalize the MCX client for the MCX user. The access token is used by the UE to transmit the MCX user's identity to the MCX server.

3.4 OpenID Connect Protocol

MCX user service authentication is the process of verifying whether the MCX user has the authority to access a specific MCX service. The MCX client presents the access token for the MCX service from the UE to each service of interest (e.g., MCX server configuration management, key management, group management, etc.) to access the MCX service. If the access token is valid, the user is granted access to the service. All HTTP traffic between the UE and the HTTP proxy, and all HTTP traffic between the UE and the KMS (Key Management Server), are protected using HTTPS.

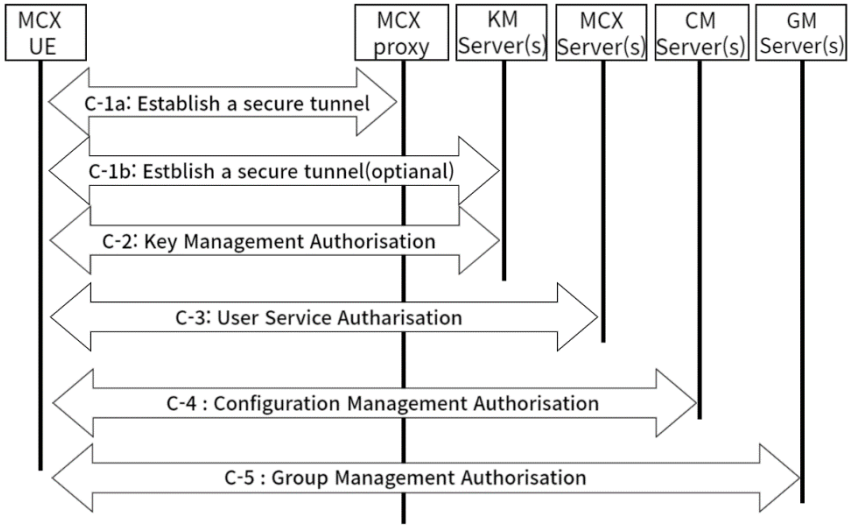

Figure 4. MCX User Service Authorization Flow Diagram

Figure 4 is a flowchart that details the MCX user service authorization in step C of Figure 1. In step C, the user authorization procedure is explained in detail through five sub-steps, which make up the MCX user service authorization process.

- C-1a: If not already performed, a secure HTTP tunnel is established between the MCX UE and the MCX proxy server using HTTPS. Subsequent HTTP messages are transmitted through this tunnel.

- C-1b: If required by the MCX system, a secure HTTP tunnel is established between the MCX KM client and the KMS using HTTPS. If supported, subsequent HTTP messages between the MCX KM client and the KMS will use this tunnel instead of the tunnel established in step C-1a.

- C-2: The KM client of the MCX UE presents the access token to the KMS via HTTP. Based on the provided MC service ID, the KMS authorizes the user for the key management service and responds to the client with specific key information for the ID. If necessary, this step may be repeated to authorize the user for additional KM services (MCPTT, MCVideo, MCData).

- C-3: The MCX client of the UE presents the access token to the MCX server via SIP. If necessary, this step may be repeated to authorize the user for additional MCX services (MCPTT, MCVideo, MCData).

- C-4: The CM client of the UE presents the access token in the Get MCX user profile request via HTTP. If the token is valid, the CM server authorizes the user for the configuration management service and, upon completion of the dl step, provides the user's profile to the CM client. If necessary, this step may be repeated to get the user profile for additional services (MCPTT, MCVideo, MCData).

- C-5: The GM client of the UE presents the access token in the Get group configuration request via HTTP. If the token is valid, GMS authorizes the user for the group management service and, upon completion of this step, sends the user's group policy information and group key information to the GM client. If necessary, this step may be repeated to authorize the user for additional group services (MCPTT, MCVideo, MCData).

Figure 5. TLS Protocol Flow Diagram

Figure 5 illustrates the process of establishing a secure communication channel in OIDC and the TLS protocol used in forming a secure channel in the MCX user service authorization flow.

- Step 1: The UE sends the ClientHello message to the Proxy.

- Step 2: The Proxy sends the ServerHello, EncryptedExtension, and Finished messages to the UE.

- Step 3: The UE sends the Finished message to the Proxy.

4 Verification

4.1 BAN-Logic

This paper conducts formal verification of TLS used in MCX through BAN-Logic. BAN-Logic is a formal verification tool that verifies the security of security protocols through its unique symbols and rules [12]. Tables 1 and 2 show the unique symbols and rules of BAN-Logic, respectively.

Table 1. BAN-Logic Unique Symbols

Symbols	Meaning
P believes X (P $\mid\equiv$X)	P believes X.
P sees X (P $\lhd$ X)	P receives a message containing X.
P said X (P $\mid\sim$ X)	P sent a message M containing X at some point in the past.
P controls X (P $\mid\Rightarrow$ X)	P controls X.
#(X)	X is fresh.
$P \overset{K}{\leftrightarrow} Q$	K has the shared secret key between P and Q.
$\overset{K}{\rightarrow} P$	K is the public key of P.
$P \overset{K}{\rightarrow} Q$	K is the shared secret between P and Q.
$\{X\}_K$	X is encrypted with the key K.
$<X>_K$	X is combined with the secret K.

Table 2. BAN-Logic Rules

Symbols	Meaning
Message Meaning Rule (MM)	$\frac{P \text{ believes } P \overset{K}{\leftrightarrow} Q, P \text{ sees } \{X\}_K}{P \text{ believes } Q \text{ said } X}$ $\frac{P \text{ believes } P \overset{K}{\leftrightarrow} Q, P \text{ sees } < X >_K}{P \text{ believes } Q \text{ said } X}$ $\frac{P \text{ believes } P \overset{K}{\leftrightarrow} Q, P \text{ sees } \{X\}_{K^{-1}}}{P \text{ believes } Q \text{ said } X}$
Nonce Verification Rule (NV)	$\frac{P \text{ believes } \#(X), P \text{ believes } Q \text{ said } X}{P \text{ believes } Q \text{ believe } X}$
Jurisdiction Rule (JR)	$\frac{P \text{ believes } Q \text{ controls } X, \quad P \text{ believes } Q \text{ believes } X}{P \text{ believes } X}$
Freshness Rule (FR)	$\frac{P \text{ believes } \#(X)}{P \text{ believes } \#(X, Y)}$
Decomposition Rule (DCO)	$\frac{P \text{ sees } (X, Y)}{P \text{ sees } X}$
Belief Conjunction Rule (BC)	$\frac{P \text{ believes } X, \ P \text{ believes } Y}{P \text{ believes } (X, Y)}$ $\frac{P \text{ believes } Q \text{ believes } (X, Y)}{P \text{ believes } Q \text{ believes } X}$ $\frac{P \text{ believes } Q \text{ said } (X, Y)}{P \text{ believes } Q \text{ said } X}$
Diffie Hellman Rule (DH)	$\frac{P \text{ believes } Q \text{ said } \overset{g^Y}{\rightarrow} Q, \quad P \text{ believes } \overset{g^X}{\rightarrow} P}{P \text{ believs } P \overset{g^{XY}}{\leftrightarrow} Q}$

4.2 Verification Results

(1) Idealization

$(I1)\ \text{Proxy} \rightarrow \text{UE} : \{\text{random}_{\text{UE}}, \text{random}_{\text{P}}, g^y, \text{FTS}\}_{\text{PSK}}$
$(I2)\ \text{UE} \rightarrow \text{Proxy} : \{\text{random}_{\text{P}}, \text{FTS}\}_{\text{FTS}}$

In the idealization, (I1) represents the Proxy server sending the ServerHello message to the UE, encrypting the message containing the UE's nonce, Proxy server's nonce, gy, and the FTS key with the pre-shared key. (I2) represents the UE sending the Finished message to the Proxy server, encrypting the message containing the Proxy server's nonce and the FTS key with the pre-shared key.

(2) Assumption

$(A1)\ \text{UE} \mid \equiv \text{UE} \overset{\text{PSK}}{\longleftrightarrow} \text{Proxy}$
$(A2)\ \text{UE} \mid \equiv \#(\text{random}_{\text{UE}})$
$(A3)\ \text{UE} \mid \equiv \overset{g^x}{\rightarrow} \text{UE}$
$(A4)\ \text{Proxy} \mid \equiv \text{UE} \overset{\text{FTS}}{\longleftrightarrow} \text{Proxy}$
$(A5)\ \text{Proxy} \mid \equiv \#(\text{random}_{\text{P}})$

In the assumption process, it is assumed that the UE and Proxy each share a pre-shared key and trust that their respective nonces are fresh. It can be assumed that the UE believes gx is publicly available.

(3) Goal

$(G1)\ \text{UE} \mid \equiv \text{Proxy} \mid \equiv \text{FTS}$

(G2) UE | ≡ FTS
(G3) Proxy | ≡ UE | ≡ FTS

In the goal process, indirect trust is required because the UE and Proxy server must have a verification procedure for the FTS key. Additionally, the UE requires direct trust to verify the key's integrity in FTS.

(4) Derivation

(D1) $UE \lhd \{random_{UE}, random_P, g^y, FTS\}_{PSK}$ by (I1)
(D2) $UE \mid \equiv Proxy \mid\sim (random_{UE}, random_P, g^y, FTS)$ by (D1), (A1), MM
(D3) $UE \mid \equiv Proxy \mid \equiv (random_{UE}, random_P, g^y, FTS)$ by (D2), (A2), FR, NV
(D4) $UE \mid \equiv Proxy \mid \equiv FTS$ by (D3), BC
(D5) $UE \mid \equiv g^{xy}$ by (D3), (A3), DH
(D6) $UE \mid \equiv FTS$ where $FTS = HKDF(g^{xy}, random_{UE}, random_P)$ by (D3), (A4), BC

(D7) $Proxy \lhd \{random_P, FTS\}_{FTS}$ by (I2)
(D8) $Proxy \mid \equiv UE \mid\sim (random_P, FTS)$ by (D7), (A4), MM
(D9) $Proxy \mid \equiv UE \mid \equiv (random_P, FTS)$ by (D7), (A5), FR, NV
(D10) $Proxy \mid \equiv UE \mid \equiv FTS$ by (D9), BC

The goal of this verification is to require both direct and indirect trust in the UE's FTS key and indirect trust in the Proxy server's FTS key. The content of (G1) can be derived in the (D4) process, (G2) can be derived in the (D6) process, and (G3) can be derived in the (D10) process. In the process of deriving this, no additional hypotheses are found, and it can be seen that all the goals can be derived. In conclusion, the derived results above indicate that the keys can be sufficiently trusted, and the end-to-end communication can be considered secure.

5 Conclusion

This paper conducted formal security verification of the TLS protocol, which is used for authentication in MCX. No additional hypotheses exist in the protocol verification process, and all goals can be derived. Therefore, the result of the verification shows that it is a secure protocol for end-to-end communication security. Future research will analyze and improve whether the OpenID Connect protocol, used for MCS authentication, can meet the requirements in mobile communication environments where handovers frequently occur.

Funding Details

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. RS-2023-00210767).

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

Kyeonga Kang: She is currently an undergraduate student at Gyeongsang National University. Her current research interests are network security and formal security analysis.

Seungbin Lee: He is currently an undergraduate student at Gyeongsang National University. His current research interests are 5G/6G security, internet of things, and formal security analysis.

Jiyoon Kim: Received the B.S., M.S. and Ph.D. degrees in information security from Soonchunhyang University, Asan, Republic of Korea, in 2017, 2019, and 2022, respectively. He was a Senior Researcher at Kookmin University, Seoul, Republic of Korea. He is currently working as an Assistant Professor with the Department of Computer Science and Engineering, Gyeongsang National University, Jinju, Republic of Korea. His current research interests include 6G security, internet of things, formal security analysis, protocol design and optimization.

References

[1] 3GPP, “Mission Critical Push To Talk (MCPTT); Stage 1,” 3GPP TS 22.179 version 19.3.0, Sept. 2024.
[2] 3GPP, “Functional architecture and information flows for mission critical video; Stage 2,” 3GPP TS 23.281 version 19.4.0, Sept. 2024.
[3] 3GPP, “Functional model and information flows for Mission Critical Data,” 3GPP TS 23.282 version 19.4.0, Sept. 2024..
[4] S. Kuklinski, K. Szczypiorski, and P. Chemouil, “UAV support for mission critical services,” Energies, 15(15):5681, Aug. 2022.
[5] J. Kim, Y. Kim, Y. Song, and S. Choi, “Group Communication Enabler Technologies and a Verification Methodology for Mission Critical Service,” The Journal of Korean Institute of Communications and Information Sciences, 42(6):1285-1296, June 2017.
[6] D. Hardt, “The OAuth 2.0 Authorization Framework,” IETF RFC 6749, Oct. 2012. https://datatracker.ietf.org/doc/html/rfc6749 [Online; accessed on November 20, 2024]
[7] N. Sakimura, J. Bradley, M. Jones, B. de Medeiros, and C. Mortimore, “OpenID Connect Core 1.0 incorporating errata set 1,” OpenID Foundation. https://openid.net/specs/openid-connect-core-1_0-errata1.html [Online; accessed on November 20, 2024]
[8] E. Rescorla, “The Transport Layer Security (TLS) Protocol Version 1.3,” IETF RFC 8446, Aug. 2018. https://datatracker.ietf.org/doc/html/rfc8446 [Online; accessed on November 20, 2024]
[9] 3GPP, “3GPP System Architecture Evolution (SAE); Security architecture,” 3GPP TS 33.401 version 18.2.0, Sept. 2024.
[10] 3GPP, “Security architecture and procedures for 5G System,” 3GPP TS 33.501 version 19.0.0, Sept. 2024.
[11] M. Jones and D. Hardt, “The OAuth 2.0 Authorization Framework: Bearer Token Usage,” IETF RFC 6750, Oct. 2012. https://datatracker.ietf.org/doc/html/rfc6750 [Online; accessed on November 20, 2024]
[12] M. Burrows, M. Abadi, and R. Needham, “A Logic of Authentication,” ACM Transactions on Computer Systems, 8(1), 18–36, Feb. 1990.

Author’s Biography

KyeongA Kang: She is currently an undergraduate student at Gyeongsang National University. Her current research interests are network security and formal security analysis.

Seungbin Lee: He is currently an undergraduate student at Gyeongsang National University. His current research interests are 5G/6G security, internet of things, and formal security analysis.

Jiyoon Kim: He received the B.S., M.S. and Ph.D. degrees in information security from Soonchunhyang University, Asan, Republic of Korea, in 2017, 2019, and 2022, respectively. He was a Senior Researcher at Kookmin University, Seoul, Republic of Korea. He is currently working as an Assistant Professor with the Department of Computer Science and Engineering, Gyeongsang National University, Jinju, Republic of Korea. His current research interests include 6G security, internet of things, formal security analysis, protocol design and optimization.