https://doi.org/10.7840/kics.2025.50.6.978

CAN BUS 통신을 위한 경량 암호화 알고리즘 비교

인 지 원◆, 최 승 식°

Comparison of Lightweight Encryption Algorithms for CAN BUS Communication

Ji-won In◆, Seung-sik Choi°

요 약

4차 산업혁명의 핵심인 자율주행자동차에서 사용되는 CAN BUS는 호스트 컴퓨터 없이 컨트롤러나 장치들이 서로 통신하기 위해 설계된 표준 통신 규격으로 현재 암호화 과정 없이 사용된다. 따라서 차량 내부에 허가 없이 접속되는 사용자가 있거나 혹은 차량 외부 네트워크을 통해 침입하는 공격자가 있을 경우에는 정보 보호에 취약해 질 수 있다. 본 논문에서는 CAN BUS를 사용하는 차량 내부 네트워크에 사용할 수 있는 암호화 알고리즘 AES, ARIA, HIGHT, PRESENTS를 구현하여 암호화/복호화 시간과 단위시간에 처리할 수 있는 데이터 수를 분석하여 알고리즘의 성능을 비교하였다. 또한 AES 암호화/복호화 알고리즘을 활용해 64Bits 단위의 AES-Light 암호화/복호화 알고리즘을 제안하고 이의 성능을 기존 알고리즘과 비교 분석하였다.

Key Words : CAN BUS, AES, ARIA, HIGHT, PRESENTS

ABSTRACT

The CAN BUS used in autonomous vehicles, which are the core of the 4th industrial revolution, is a standard communication specification designed for controllers and devices to communicate with each other without a host computer. However, since CAN BUS is currently used without an encryption process, there may be users who access the system without permission from the outside, and it may be vulnerable to attacks from the outside. In this paper, we implement encryption algorithms that can be used in CAN BUS, such as AES, ARIA, HIGHT and PERSENTS. And compare them by measuring the encryption time, decryption time, and the number of data that can be processed per second. In addition, we propose a 64-Bit AES-Light encryption/decryption algorithm using the AES encryption/decryption algorithm and evaluate its performance.

Ⅰ. 서 론

자율주행 자동차에서의 보안은 프라이버시와 기밀성 보호 이외에도 보안 전 영역에 걸쳐 암호화가 활용된다. 암호화는 비밀키 기반, 공개키 기반, ID 기반 서명을 사용하는 방법, 인증서 없는 서명 기반 방법들이 활용되었다. 자율주행 자동차 통신 중 차량 내부 통신 기술로는 CAN BUS와 LIN(Local Internet Network)이 있다. CAN BUS는 자동차 내부의 서로 다른 전자장치 간의 통신을 위해 개발되었다.[1] 비동기 직렬 통신 방식인 UART가 자동차 무게를 증가시키고 원가를 상승시킨다는 단점을 보완하기 위해 개발 되었고 1:1 방식으로 통신하여 데이터의 양이 증가하면 배선 수를 늘려야하는 UART와 달리 CAN BUS는 주로 ECU에 연결되어

◆ First Author : Incheon National University Department of Computer Science & Engineering, dlswl0618@inu.ac.kr, 정회원
° Corresponding Author : Incheon National University Department of Computer Science & Engineering, sschoi@inu.ac.kr, 종신회원
논문번호 : 202410-260-B-RN, Received October 27, 2024; Revised December 27, 2024; Accepted December 30, 2024

마이크로센서의 요청에 따라 CAN BUS와 연결된 센서에 데이터를 페러럴 통신으로 진행하다는 특징이 있다[2]. 자동차 환경과 같은 심각한 잡음 환경에 적합하도록 강한 에러 검출 및 에러 보정 기능을 가지고 있다[3].

LIN(Local Internet Network)의 경우 CAN BUS의 전체 개발비용이 많이 필요하다는 단점을 보완하기 위해 개발 되었다. LIN은 Single Wire를 이용해 20Kbit/s로 데이터 통신이 이루어지기 때문에 개발비용이 매우 저렴하다는 장점을 가지고 있다. Master/Slave의 통신 방식을 적용하여 하나의 Master와 하나 이상(최대 16개)의 Slave로 구성 되어있다[4].

그러나 고속의 데이터 전송과 유연한 통신을 요구하는 자율주행자동차의 환경에서는 저속 인 LIN과 달리 최대 1Mbps까지 지원하며 고속 데이터 전송이 가능하고 이중 마스터 구조를 지원하여 네트워크 내 모든 노드가 데이터를 송수신할 수 있는 CAN BUS 통신이 더 많이 사용된다.

자동차 컨트롤유닛 프로세서 간 통신과 엘리베이터와 같은 산업기기에 많이 사용되는 CAN BUS는 현재 암호화 과정 없이 사용되어지고 있다. 따라서 데이터가 브로드캐스트될 때 데이터 프레임의 기밀성과 무결성, 인증을 보장하지 않아 허가 받지 않은 외부 접속자가 발생할 수 있다[5]. 더 나아가면 외부에 있는 공격자가 쉽게 접속하여 데이터 및 내부 정보를 도청하거나 추가적인 공격이 가능하다는 문제가 제기된다. 따라서 CAN BUS 통신의 보안을 위해 암호화가 필요하다.

본 논문에서는 CAN BUS 통신에서 사용할 수 있는 암호화 알고리즘 AES, ARIA, HIGHT, PRESENT를 구현하여 암호화 시간과 복호화 시간, 일정 단위시간 동안 실행할 수 있는 데이터 수를 비교분석한다. 그리고 AES 방식을 활용하여 블록크기가 64Bit인 암호화/복호화 알고리즘을 구현하고 이의 성능을 평가한다.

Ⅱ. 관련 연구 정보

2.1 CAN BUS 통신

CAN BUS 통신은 최대 1Mbps까지 사용할 수 있으며 Multi Node 구성인 자동차나 엘리베이터 등에서 많이 사용된다. Address Bit 수에 따라 11 Bit 식별자를 가진 CAN Bus 2.0A와 29 Bit 식별자를 가진 CAN 2.0B 방식이 있다[6]. 이때 데이터 필드 크기는 최대 64 Bit이다. CAN BUS 통신은 프레임이라고 부르는 패킷으로 데이터를 전송한다. 프레임은 하나의 메세지를 이루는 필드 또는 Bit들의 집합을 말하며 다음과 같은 분할 구역으로 구성되어 있다. CAN FD는 2012년에 출시

그림 1. CAN BUS 패킷 구조
Fig. 1. CAN Bus Packet Structure

되어 데이터 전송량이 증가된 프로토콜이다. CAN 2.0B와 동일한 프레임 형식을 사용하고 새로운 제어필드를 추가하여 CAN 2.0 장치와의 하위 호환성을 유지한다. 이때 데이터 필드 크기는 최대 512 Bit이다[7].

CAN은 2가지의 메시지 프레임 포맷을 지원한다. 그림 1의 Identifier에서 11 비트의 ID를 갖는 표준 포맷(CAN 2.0A)과 29 비트의 ID를 갖는 확장된 포맷(CAN 2.0B)으로 구분된다[8]. 자동차의 ECU의 CAN 메시지들은 고유의 ID값을 가지고 있어 동시에 메시지를 CAN BUS에 전송하려는 경우 가장 낮은 ID값을 가진 최우선 노드가 자동으로 버스에 접근하게 된다[9].

2.2 기존 암호화 알고리즘

2.2.1 AES

AES(Academy Encryption Standard)는 미국 표준연구소에서 2001년도에 제정된 SPN 구조의 암호 방식이다[10]. 1977년 제정된 DES 알고리즘을 대체하여 암호화 복호화 과정에서 같은 KEY를 사용하는 대칭키 알고리즘이다. KEY는 128 Bit, 192 Bit, 256 Bit로 확장이 가능하며 널리 사용되는 미국 정보표준 암호화 알고리즘이다[11]. 암호화를 위해 평문과 키를 입력으로 받아 암호문을 생성한다. 복호화 과정은 부분적으로 다른 데이터 경로에서 발생하는 반복을 반전시킨다.

AES 알고리즘에서 평문은 4*4 바이트 행렬로 표현된다. 중간 암호 결과를 'state' 라고 하는데, 초기 라운드 키 추가가 수행된 후 128비트에 대해 10개의 라운드를 구현하여 상태를 변환한다. 최종 라운드를 제외한 각 라운드 함수에는 하나의 단일 바이트 기반 대체 단계(SubBytes), 행 단위 순열 단계(ShiftRows), 열 단위 혼합 단계(MixColumns), 라운드 키 추가 단계(AddRoundKey)의 4가지 변환이 포함된다. 최종 라운드는 다른 라운드들과 다르게 SubBytes, ShiftRows, AddRoundKey의 단계로 구성되며 MixColumns 연산은 포함되지 않는다[12].

2.2.2 ARIA

ARIA(Academy Research Institute Agency)는 2004년 한국산업규격 KS 기준으로 제정된 SPN 구조의 대칭키 알고리즘으로 국내 기술로 개발되었다[10]. 블

록 크기는 128 Bit이며, 128 Bit, 192 Bit, 256 Bit의 확장키를 사용할 수 있다. 입력된 키를 128비트씩 나누어 KL, KR라 하고 KR의 부족한 비트는 0으로 패딩한다[13].

암호화와 복호화를 수행하는 라운드 함수와 키 스케줄러로 구성되어 있다. 키의 크기에 따라 가변 라운드(12, 14, 16)가 적용된다. 내부 함수는 세단계로 라운드 키 삽입, 치환계층, 확산계층으로 이루어져있다. 라운드 키 삽입은 키 스케줄에서 생성된 라운드 키와 라운드 함수 입력간의 XOR 연산으로 이루어져있다. 라운드를 사용한 3 라운드 256비트 Feistel 구조로 4개의 마스터 키(MK)로 128비트 초기 값 4개를 생성한다[14].

2.2.3 HIGHT

HIGHT(HIGH security and light weigHT)는 2005년 KISA와 고려대학교가 공동으로 개발한 64 Bit ARX 구조의 블록 암호 알고리즘이다[10]. 알고리즘의 전체 구조는 일반화된 Feistel 변형 구조로 이루어져 있으며, 64 Bit의 평문과 128 Bit 키로부터 생성된 8개의 8 Bit 화이트닝 키와 128개의 8 Bit 서브키를 입력으로 사용한다. 총 32 라운드를 거쳐 64 Bit 암호문을 출력한다. 64비트 블록암호로 128비트 암호키를 활용해 메시지를 64비트 블록 단위로 암호화와 복호화 하는 알고리즘으로 데이터의 기밀성과 같은 기능을 제공하기 위해 사용될 수 있다[15].

2.2.4 PRESENT

AES 기반의 SPN 구조인 경량 블록 암호 알고리즘이다[10]. 블록 사이즈는 64 Bit, 키 사이즈는 80 Byte와 128 Bit 두 종류를 가지고 있다. 주로 64 Bit의 평문과 80 Bit의 키를 사용한다. PRESENT 알고리즘은 4 Bit의 S-Box와 배타적 논리합 및 비트 시프트로 연산되는 32개의 라운드로 구성되어 있다[8]. AES 와 비교했을 때 암호화 등급은 조금 낮지만 AES 보다 2.5배 작은 하드웨어 설계가 가능할 정도로 면적과 소비전력을 개선하여 효율성을 높인 알고리즘 이다[16].

이렇게 보안성과 빠른 속도를 갖추고 있는 많은 경량 암호 알고리즘들이 증가하고 있지만, 전송속도와 더불어 데이터의 변형이 없이 암호화 및 복호화가 더 빠르게 진행되는 경량 암호 알고리즘이 필요로 하고 있다.

Ⅲ. CAN BUS를 위한 AES-Light

AES-Light 알고리즘은 AES 암호화 알고리즘과 같이 라운드별로 SubBytes, ShiftRows, MixColumns, AddRoundKey의 단계를 거쳐 평문 데이터를 암호화된 데이터로 변환한다. JAVA 언어를 사용하여 구현된 AES-Light 알고리즘은 효율적인 메모리 사용과 빠른 처리 속도를 통해 다양한 환경에서 사용이 용이하고, 특히 제한된 자원과 낮은 전력 소비가 중요한 환경에서 AES-Light는 효과적인 암호화 솔루션을 제공한다.

AES-Light 알고리즘은 블록 암호화 알고리즘인 AES 알고리즘을 64비트 블록 크기에 맞게 변형한 것이다. AES는 일반적으로 128비트 블록을 처리하지만, AES-Light 알고리즘은 더 작은 블록 크기를 필요로 하는 환경에서 최적화된 알고리즘으로 경량 암호화 응용 프로그램이나 자원이 제한된 환경에 적합하다.

AES-Light 알고리즘은 그림 2의 라운드함수에 제시된 형태로 평문을 고정된 크기의 블록으로 나누어 암호화 한다. 각 블록은 총 11라운드에 걸쳐 변환되며, 각 라운드는 SubBytes, ShiftRows, MixColumns, AddRoundKey 단계로 구성되면서 입력 블록 크기와 일부 연산을 64비트에 맞게 조정하였다. AES-Light 알고리즘은 8바이트(64비트) 블록으로 데이터를 처리하며, 암호학적 보안을 위해 필수적인 확산(diffusion) 속성을 보장하기 위해 고정된 변환 행렬을 사용한다.

AES-Light 알고리즘의 SubBytes 단계에서는 AES와 유사하게 미리 정의된 S-Box를 사용한다. S-Box는 입력 바이트를 출력 바이트로 치환하여 데이터의 비선형성을 높이고 암호화 강도를 향상시킨다. 각 입력 바이트는

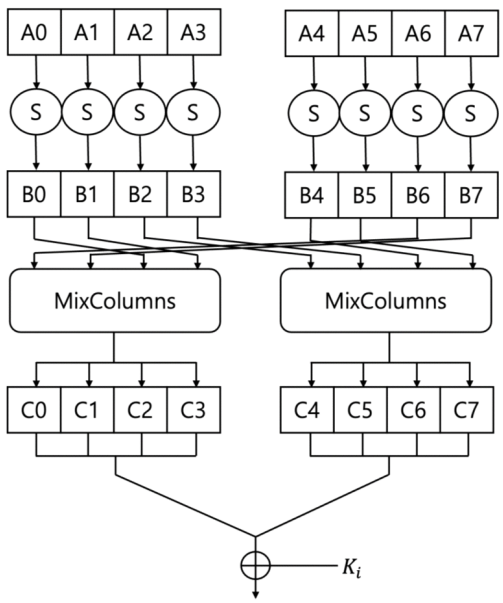

그림 2. 블록암호 AES-Light의 라운드 함수
Fig. 2. Round function of block cipher AES-Light

S-Box를 통해 변환되며 이는 암호화의 첫번째 단계에서 중요한 역할을 한다.

ShiftRows 단계에서는 64비트 블록을 2x4 행렬로 보고, 각 행의 바이트를 왼쪽으로 이동시킨다. 이 연산은 데이터의 행 간 혼합을 통해 암호화 강도를 높이는 역할을 한다. 원래 데이터 블록의 각 비트가 암호화된 결과물에 전반적으로 영향을 미치도록 한다. 그림 3과 같이 Shift Rows를 하기 전 초기 라운드 구조를 시작으로 짝수 번째 라운드에서 그림 4와 같이 짝수번에 있는 위치끼리 위치를 바꾸게 된다. 이 후 라운드가 홀수인 경우 그림 5와 같이 홀수번에 있는 위치끼리 위치를 바꾸면서 Shift Rows 단계를 수행한다.

MixColumns 단계에서 고정된 2x4 행렬을 사용하여 열 간의 복잡한 연산이 수행된다. 각 행을 XOR 연산하는데, 이 연산은 행렬과의 결합을 통해 데이터를 더욱 복잡하게 만들어 열 간의 데이터 의존성을 높이고, 데이터의 확산성을 강화한다. 이러한 XOR 연산은 암호화 과정에서 중복 패턴을 제거하고 다양한 키와 평문에 대해 더욱 안전한 암호화를 제공한다.

AddRoundKey 단계에서 현재 상태와 라운드 키를 XOR 연산하여 결합한다. 이 단계는 알고리즘의 보안을 높이는 핵심 단계로, 매 라운드 마다 새로운 키가 생성되고 사용되어 보안성을 강화한다. 키 스케줄링을 통해 생성된 라운트 키와 현재 블록 상태를 XOR 연산하여, 각 라운드마다 데이터가 안전하게 보호 되도록 한다.

0	1	2	3
4	5	6	7

그림 3. Shiht Rows 단계 초기 라운드 구조
Fig. 3. Shiht Rows Initial round Structure

6	1	4	3
2	5	0	7

그림 4. Shiht Rows 단계 라운드 0 구조
Fig. 4. Shiht Rows round 0 Structure

6	7	4	5
2	3	0	1

그림 5. Shiht Rows 단계 라운드 1 구조
Fig. 5. Shiht Rows round 1 Structure

Ⅳ. 구현 및 성능

AES-Light 알고리즘은 64비트 크기의 평문과 키를 사용하는 암호화 알고리즘이다. 제한된 자원을 가진 환경에서의 사용을 염두에 두고 설계되었다. 성능 평가를 위해 타 알고리즘과 비교 실험을 진행하였다. 실험에 소요된 총 시간을 측정한 결과 평균적으로 0.04975ms, 복호화에 소요된 총 시간은 평균적으로 0.058167ms가 소요되었다.

이 결과는 AES-Light 알고리즘이 암호화 및 복호화 모두에게 빠른 처리속도를 자랑하는 것으로 보여진다. 그림 6과 같이 AES-Light알고리즘이 PRESENT 알고리즘 보다 약 12배 빠른 속도를 기록하였으며, 복호화는 그림 7을 참고하면 AES와 비교했을 때 약 12배 이상의 성능 향상을 보였다. 이는 특히 실시간 데이터 처리가 중요한 CAN BUS에서 큰 이점으로 적용할 수 있다. 위와 같은 암호화 시간과 복호화 시간 결과로 일정 단위 시간에 실행할 수 있는 데이터 수를 계산해본다면 암호화 과정의 경우 그림 8과 같이 표현될 수 있다. 암호화에 걸린 시간을 일정 시간 단위로 변환하여 각 처리 시간의 역수로 계산한 방식이다. AES-Light 알고리즘이 다른 알고리즘에 비해 월등하게 높은 암호화 처리량을 보이며 단위 시간당 처리 가능한 데이터 수가 가장 많다. 복호화 과정의 경우 그림 9와 같이 표현될 수 있다. 복호화에 걸린 시간을 일정 시간 단위로 변환하여 각 처리 시간의 역수로 계산한 방식으로, AES-Light가 복호화 성능이 가장 뛰어나며 일정 시간 단위 당 처리할

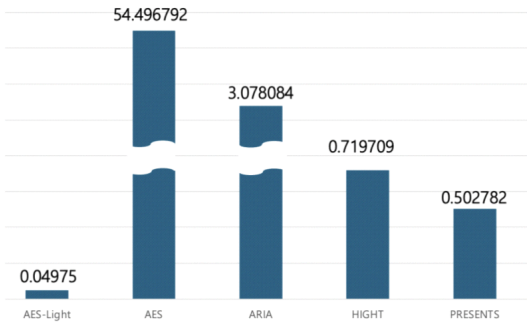

그림 6. 암호화 시간 비교 그래프
Fig. 6. Encryption time Comparison graph

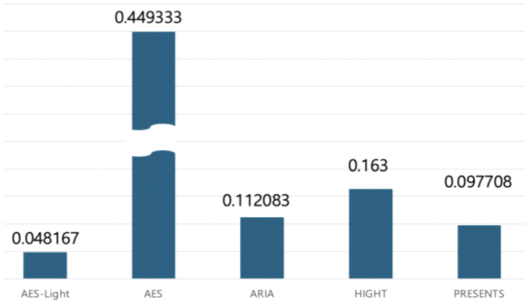

그림 7. 복호화 시간 비교 그래프
Fig. 7. Decryption time Comparison graph

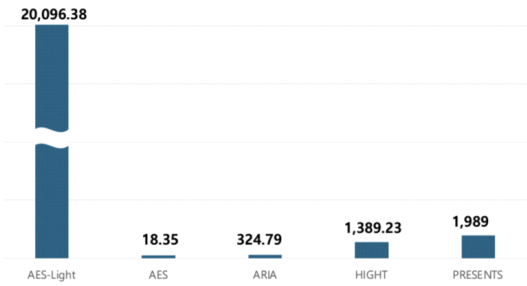

그림 8. 암호화 데이터 수 (단위시간: 초)
Fig. 8. Number of encrypted data (Unit Time: Seconds)

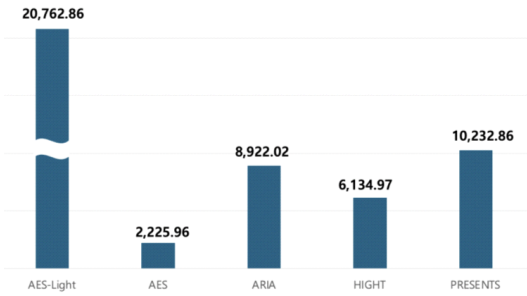

그림 9. 복호화 데이터 수 (단위시간: 초)
Fig. 9. Number of decrypted data (Unit Time: Seconds)

수 있는 복호화 데이터 수가 가장 많았다. AES-Light 알고리즘의 빠른 처리 속도와 연산 복잡도의 높이는 제한된 메모리와 처리 능력을 가진 기기 및 시스템 환경에서 비교적 유리하게 적용된다.

V. 결 론

본 연구에서는 128 Bit부터 선택할 수 있는 AES 알고리즘을 64 Bit로 경량화한 AES-Light 알고리즘을 제안하고 이를 기존의 다양한 경량 암호화 알고리즘과 성능 비교를 통해 평가하였다. 실험 결과, 제안된 AES-Light 알고리즘은 모든 테스트에서 가장 빠른 암호화 및 복호화 속도를 기록하였으며, 이는 연산 자원이 제한된 환경에서의 적용 가능성을 확인할 수 있다. AES-Light 알고리즘의 성능 개선은 실시간 암호화가 필요한 시스템에서 효과적으로 활용될 수 있음을 시사한다. 예를 들어, 무선 센서 네트워크, CAN BUS 같은 실시간 데이터 전송이 중요한 환경에서는 빠른 암호화와 복호화가 필요로 하고 있고 AES-Light 알고리즘이 이를 충족하는데에 있어 적합하다.

References

[1] G. H. Kim, D. S. Sin, and S. H. Kim, "Horizontal control system using CAN communication," in *Proc. 2013 KICS Fall Conf.*, Seoul, Korea, Oct. 2013.

[2] T. W. Kang and S. S. Lee, "Multiple UART communication using CAN bus," *J. Korean Inst. Electr. Electron. Eng.*, vol. 24, no. 4, pp. 1184-1187, Dec. 2020.

[3] Y. J. Wu and J. G. Chung, "Efficient controller area network data compression for automobile application," *Front. Inf. Technol. Electron. Eng.*, vol. 16, no. 1, pp. 70-78, Jan. 2015.

[4] *Introduction to the LIN Bus Protocol(2024)*, Retrieved Jul. 2024, from https://buly.kr/6tbx4Q7

[5] D. H. Lee, G. H. Jang, and S. S. Lee, "Design of an authenticated encryption architecture and hardware engine with improved reliability and security on the CAN-FD protocol," *J. Korean Inst. Electr. Electron. Eng.*, vol. 27, no. 2, pp. 204-212, Jun. 2023.

[6] B. J. Hong, I. C. Han, D. W. Jang, and N. Y. Lee, "Empirical study on the extended CAN bus communication with security algorithms," *J. KICS*, vol. 43, no. 9, Jul. 2018.

[7] J. B. Lee and S. S. Lee, "Implementation and verification of automotive CAN-FD controller," *J. Inst. Korean Electr. Electron. Eng.*, vol. 21, no. 3, pp. 240-243, Sep. 2017.

[8] S. K. Lee, J. Y. Lee, D. H. Kim, K. J. Choi, and J. I. Jung, "CAN communication system using CAN Protocol," in *Proc. 25th Spring Conf. Korean Inf. Process. Soc.*, vol. 13, no.

1, Korea, May 2006.

[9] O. Avatefipour and H. Malik, “State-of-the-art survey on in-vehicle network communication (CAN-bus) security and vulnerabilities,” arXiv preprint *arXiv:1802.01725*, 2018. (https://doi.org/10.48550/arXiv.1802.01725)

[10] *Lightweight Cryptography*(2021), Retrieved May 2024, from https://buly.kr/Alkcea5

[11] D. Selent, “Advanced encryption standard,” *Rivier Academic J.*, vol. 6, no. 2, pp. 1-14, 2010.

[12] S. G. Kim, G. H. Kim, and G. Y. Cho, “Development of stream cipher using the AES,” *J. KICS*, vol. 38C, no. 11, Nov. 2013.

[13] T. W. Kwon, H. M. Kim, and S. H. Hong, “SEED and ARIA algorithm design methods using GEZEL,” *J. Korea Inst. Inf. Secur. Cryptol.*, vol. 24, no. 1, Feb. 2014.

[14] H. R. Yoo, S. J. Lee, and Y. D. Son, “Hardware design and implementation of block encryption algorithm ARIA for high throughput,” *J. Inst. Korean Electr. Electron. Eng.*, vol. 22, no. 1, pp. 104-109, Mar. 2018.

[15] D. Hong, et al., “HIGHT block encryption algorithm specification and detailed specification,” *CHES 2006*, LNCS 4249, Springer Verlag, 2006.

[16] A. Bogdanov, L. R. Knudsen, G. Leander, C. Paar, A. Poschmann, M. J. B. Robshaw, Y. Seurin, and C. Vikkelsoe, “PRESENT: An ultra-lightweight block cipher,” *Cryptographic Hardware and Embedded Syst.*, pp. 450-466, 2007.

인 지 원 (Ji-won In)

2021년 3월~2025년 2월 : 인천대학교 컴퓨터공학부 졸업

<관심분야> 통신, 네트워크, V2X, 보안

최 승 식 (Seung-sik Choi)

1988년 : 연세대학교 전자공학과 졸업

1990년 : KAIST 대학원 전기 및 전자공학과 석사

2002년 : KAIST 대학원 전기 및 전자공학과 박사

1990년~2004년 : KT 연구센터 선임 연구원

2004년~현재 : 인천대학교 컴퓨터공학과 교수

<관심분야> 사물인터넷, 무선 MAC, 인터넷 프로토콜