Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.28, NO.6, Dec. 2018 ISSN 2288-2715(Online)
https://doi.org/10.13089/JKIISC.2018.28.6.1309

스마트 IoT 디바이스를 위한 경량 암호기반 종단간 메시지 보안 프로토콜*

김 희 정,[1†] 김 정 녀[2‡]
[1]과학기술연합대학원대학교, [2]한국전자통신연구원

A Study of End-to-End Message Security Protocol Based on Lightweight Ciphers for Smart IoT Devices*

Hee-jeong Kim,[1†] Jeong Nyeo Kim[2‡]
[1]University of Science and Technology,
[2]Electronics and Telecommunications Research Institute

요 약

IoT 시장은 전 세계에서 지속적으로 성장하고 있지만, IoT 환경에서 증가하는 보안 위협에 대한 고려는 여전히 미흡한 실정이다. 특히 자원이 한정적인 IoT 디바이스에 기존 IP 보안 기술을 적용하기 어렵기 때문에, 이기종 사물 네트워크 간 통신 과정에서 발생할 수 있는 정보 변조·유출과 같은 보안 위협에 대응하기 위한 디바이스 종단간 신뢰 통신 보안 대책이 필요하다. 이에 본 논문에서는 저성능의 IoT 디바이스 간 통신에서 보안성은 늘리며 보안 오버헤드는 줄일 수 있는 경량 암호기반 종단간 메시지 보안 프로토콜을 제안한다. 실행 시간 성능 시뮬레이션을 통해 제안 프로토콜이 기존의 AES 기반의 프로토콜을 적용한 경우보다 성능이 더 우수함을 검증하였다.

ABSTRACT

Although the IoT market is steadily growing, there is still a lack of consideration for increasing security threats in the IoT environment. In particular, it is difficult to apply existing IP security technology to resource-constrained devices. Therefore, there is a demand for reliable end-to-end communication security measures to cope with security threats such as information tampering and leakage that may occur during communication between heterogeneous networks do. In this paper, we propose an end-to-end message security protocol based on lightweight cipher that increases security and lowers security overhead in resource-constrained IoT device communication. Through simulation of processing time, we verified that the proposed protocol has better performance than the existing AES-based protocol.

Keywords: Lightweight cipher, MQTT, End-to-End Message Security Protocol

I. 서 론

IoT(Internet of Things)는 인터넷에 연결된 사람, 디바이스, 공간, 데이터 등과 같은 모든 Things로부터 생성 및 수집된 정보를 공유하고 활용하여 서비스를 제공하는 초연결 인터넷을 말한다[1]. IoT에서는 센서와 임베디드 컴퓨팅 기술을 이용해 다양한 데이터를 수집하고 가공하여 사람들에게

Received(07. 31. 2018), Modified(10. 18. 2018) Accepted(10. 18. 2018)
* 이 논문은 2018년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임 (No.2018-0-00231, (IoT 2세부) IoT 인프라 공격 확산 방어를 위한 상황 적응형 보안 자율제어 기술개발)
† 주저자, aimcan1@gmail.com
‡ 교신저자, jnkim@etri.re.kr(Corresponding author)

편리한 서비스를 지속적으로 제공할 수 있다. IoT는 사람들이 미처 인식하지 못한 정보를 인식할 수 있도록 도움을 줄 수 있고 반복적이거나 비효율적인 작업을 줄여줄 수 있다. 이로 인해 IoT는 지속적으로 성장하고 있으며[2] IoT의 발전은 IT 산업 분야뿐 아니라 사람과 사물을 활용하는 모든 산업에 영향을 미칠 것이라는 전망이다[3].

IoT에 대한 관심이 증가함에 따라 보안의 중요성 역시 대두되고 있다. 미국의 정보 기술 연구 및 자문 회사인 가트너는 기업의 IoT 보안 지출 규모가 꾸준히 증가하여 2021년에는 그 규모가 31억 달러에 달할 것이라고 보고했다[4].모든 사물들이 인터넷에 연결되어 통신한다는 것은 공격자의 공격 범위가 넓어지고 침투 가능성이 높다는 의미이다. IoT 보안 위협 사례 중, 공격자가 스마트미터에 의해 수집된 데이터들이 평문으로 전송된다는 점을 악용하여 가정에서의 소비패턴을 분석하여 개인정보를 유출하는 사례도 있다[5]. 이처럼 비 암호화 통신은 주요 정보가 쉽게 탈취되며, 단순 정보 유출 피해뿐 아니라 유출된 정보를 이용한 2차 피해까지도 야기할 수 있으므로 이를 예방하기 위해 종단간 신뢰 통신 보안 대책이 필요하다.

하지만 기존 인터넷 환경의 종단간 통신 보안 기술은 배터리, 메모리, CPU 등의 자원 제약 조건이 많은 IoT 에 곧바로 적용되기 어렵다[6]. IoT에서는 불필요한 기능들로 인해 낭비되는 전력 및 메모리를 최소화해야한다. 그렇기 때문에 경량화가 주요 이슈가 되며 이와 관련하여 IoT 환경을 위한 경량 실시간 OS(Operating System)와 CoAP (Constrained Application Protocol)과 MQTT(Message Queuing Telemetry Transport) 같은 IoT용 메시지 전송 프로토콜이 필요하다. MQTT는 UDP 기반의 CoAP과는 달리 TCP 기반의 메시지 전송 프로토콜이며 QoS(Quality of Service) 레벨을 이용해서 애플리케이션 계층에서도 신뢰성을 제공한다. 그리고 HTTP 메시지 헤더의 크기가 최소 26 Bytes인 것에 비해 MQTT는 최소 2 Bytes 크기의 메시지 전송이 가능할 만큼 메시지 헤더의 크기가 작고 간단하여 IoT 환경에 적합하다. 하지만 MQTT는 표준화된 보안 정책 가이드라인이 없기 때문에 별도의 보안 기술이 필요하다. 비표준에서는 OpenSSL의 사용을 권고하고 있지만 자원이 극히 한정된 디바이스에서 사용하기에는 무리가 있다. 또한 저성능 IoT 디바이스는 제한된 컴퓨팅 기능 때문에 기존 암호로는 서비스가 실행되기 어렵고 실행이 되더라도 정상적으로 서비스를 제공받기 어렵다. 그러므로 저성능 디바이스에서 본래 기능에 대한 성능 저하를 최소화하면서 암호를 적용하기 위해 경량 암호가 필요하다.

따라서 본 논문에서는 IoT 디바이스의 성능적 제한을 고려하여 MQTT에 보안성을 제공하는 경량 암호기반 종단간 메시지 보안 프로토콜을 제안한다. 저성능 IoT 디바이스에 적합한 OS로 개발된 Zephyr를 기반으로 구축한 IoT 개발 환경에서 제안 프로토콜을 구현하였다.

본 논문은 제1장의 서론을 포함하여 총 4개의 장으로 구성되었다. 제2장에서는 IoT 응용 프로토콜, IoT 네트워크 보안 프로토콜, 그리고 경량 암호에 대해 소개한다. 제3장에서는 경량 암호기반 종단간 메시지 보안 프로토콜에 대해 제안하고 프로토콜 성능을 평가한다. 마지막으로 제4장은 본 논문의 결론으로, 논문의 전반적인 내용을 정리하고 향후 연구 방향에 대해 기술한다.

II. 관련 연구

2.1 통신 프로토콜

2.1.1 TLS

TLS(Transport Layer Security)[7]는 TCP 프로토콜에 보안성을 제공해주는 보안 프로토콜이다. 웹 서버와 웹 브라우저 간의 암호화 통신을 위하여 Transport Layer와 Application Layer 사이에서 동작하는 프로토콜이다[8]. TLS의 주요기능은 크게 인증, 기밀성, 무결성 제공이며 클라이언트와 서버 간의 인증 및 암호화 통신을 위해 사용된다. TLS는 암호화 스위트(Cipher suite)을 이용하는데, 암호화 스위트는 통신 과정에서 사용할 프로토콜, 키교환 방식, 인증서 검증, 대칭키 블록 암호화 방식, 블록 암호 운용방식, 메시지 인증에 대한 정보를 가진다[9]. 암호화 스위트는 TLS 핸드쉐이크를 통해서 합의되기 때문에 디바이스에 미리 필요한 보안 모듈이 적재되어 있어야한다. 그러나 초소형 센서 디바이스와 같은 IoT 디바이스는 자원이 매우 한정적이기 때문에 기존의 TLS 프로토콜을 그대로 적용하기에는 어려움이 있다.

2.1.2 DTLS

DTLS(Datagram Transport Layer Security)[10]는 TLS를 UDP에 적용하기위해 제안되었으며 UDP 프로토콜에 보안성을 제공해주는 보안 프로토콜이다. 따라서 UDP 기반의 애플리케이션에서 DTLS를 사용하여 도청, 간섭, 메시지 변조 등 네트워크상에서 발생할 수 있는 위협에 대비할 수 있다. IETF 표준화 그룹에서는 안전한 IoT 서비스를 위해 경량화된 DTLS 프로토콜을 제안하고 있다[11]. DTLS는 IoT 표준 메시지 전송 프로토콜인 CoAP에 적용되어 쓰이고 있다.

2.2 IoT 응용 프로토콜

2.2.1 MQTT

MQTT는 OASIS에 의해 2013년에 IoT 표준 메시지 전송 프로토콜로 지정되었고 2016년도에 ISO 표준(ISO/IEC PRF 20922)으로 채택된 TCP 기반의 경량 메시지 전송 프로토콜이다[12]. MQTT는 IoT 통신을 위한 신뢰성을 제공하는 프로토콜로 주목받고 있으며 다양한 IoT 플랫폼 및 서비스에서 지원되고 있다. 대표적인 사례로 Facebook이 Messenger 애플리케이션에 MQTT를 도입하여 수 초 걸리던 종단간 메시지 전송 속도를 수백 밀리초로 단축해 서비스한 사례가 있다[13].

MQTT[14]는 Publish/Subscribe 구조로 되어 있으며 비동기화 방식을 지원한다. MQTT는 Publisher와 Subscriber 사이에서 메시지를 중개해주는 Broker가 존재한다. 메시지는 일종의 메시지 통신 채널인 Topic을 이용하여 전달되며 QoS 레벨을 설정하여 메시지 전송 신뢰성을 보장 받을 수 있다.

다만 MQTT 프로토콜은 IoT 환경에서 단순 포맷의 메시지를 전송할 뿐 메지시 통신상의 기밀성을 제공하지 못하여 보안에 취약하다. 또한 보안 정책 가이드라인도 없어 프로토콜 상에서 메시지가 플레인 텍스트로 전달되므로 이를 위한 보안 방안이 필요하다. 이를 위한 대책으로 프로토콜 상의 메시지를 암호화에서 전달하는 것이 논의되고 있다.

2.2.1.1 Topic

Topic[14]은 "/" 문자로 구분된 문자열의 계층 구조로 되어있다. "Home/Room1/Temperature"와 같이 최상위 계층인 "Home"을 시작으로 하위 계층들을 차례대로 작성해줄 수 있으며 최상위 계층에는 "/" 문자를 제외한다. 동시에 여러 계층에 대한 정보를 Publish/Subscribe 하기 위한 방법으로 와일드카드 "+"와 "#"을 지원한다. 와일드카드 "+"는 단일 계층, "#"은 계층의 나머지를 대신한다.

2.2.1.2 QoS

QoS[14]는 이기종 간의 네트워크 통신과정에서 신뢰성 있는 메시지 전달을 위해 제공되는 방법으로 3가지 옵션이 지원된다. QoS 0는 메시지가 최대 한 번만 전달된다. 따라서 가장 빠른 속도로 메시지를 전달할 수 있지만 통신 과정 중 메시지가 올바르게 수신되었는지에 대한 여부를 확인 할 수 없다. QoS 1은 메시지가 최소 한 번은 전달되지만 동일한 메시지가 중복해서 전달 될 수 있으므로 주의해야한다. QoS 2는 단 한번만 메시지가 전달될 수 있도록 보장한다.

2.2.1.3 TLS/SSL 적용

MQTT는 새롭게 개발된 보안 형식은 아니며, 암호화되지 않은 TCP를 사용한다. TCP를 사용하는 이유는 TLS/SSL 인터넷 보안을 사용할 수 있기 때문이다. 그러나 TLS/SSL은 필요한 핸드셰이크와 증가되는 패킷 오버헤드로 인해 경량화된 클라이언트에게 리소스가 집약적으로 사용된다. IoT환경에서는 배터리 등 에너지가 매우 높은 우선순위를 가지고 있고 보안이 훨씬 덜 중요한 네트워크의 경우 패킷 페이로드 암호화만으로 충분할 수 있다. 그러므로 저전력/경량형을 요구하는 IoT 기기인 경우에는 성능이 좋은 패킷 페이로드 암호화만을 사용한다.

2.2.2 CoAP

CoAP[15]은 IETF(Internet Engineering Task Force)에 의해 2014년에 표준으로 제정된 UDP 기반의 경량 REST 프로토콜이다. 스마트 에너지, 빌딩 자동화와 같은 M2M(Machine-to-Machine) 애플리케이션을 위해 설계되었으며 저전력/저사양의 IoT 환경에 적합한 경량 메시지 전송 프로토콜이다.

CoAP은 서버/클라이언트 구조로 되어 있다. 기

본으로 일대일 '요청/보고' 방식의 인터랙티브 모델 제공하지만 IPv6 환경에서는 멀티캐스트를 지원할 수 있다. HTTP 및 RESTful 웹과 상호 운용이 가능하도록 설계되어 있기 때문에 기존의 인터넷에 적합하다. 네 가지로 정의된 메시지 타입인 '확인형(Confirmable)', '비확인형(Non-confirmable)', '승인(Acknowledgement)', '리셋(reset)' 에 따라 메시지 전송에 대한 신뢰성 제공 여부가 결정된다. 신뢰성은 메시지 요청 과정에서 메시지의 ID를 포함한 확인형 메시지를 전송하여 동일한 메시지 ID를 포함한 응답으로 승인 메시지를 받는 과정으로 제공된다. 비신뢰성은 메시지 요청 과정에서 비확인형 메시지를 전송하고 응답을 받지 않는 과정으로 제공된다.

2.3 경량암호

2.3.1 SPECK

SPECK[16]은 Beaulieu 등이 제안하였으며 미국 NSA(National Security Agency)를 통해 2013년도에 발표되었다. Feistel 구조의 경량 블록 암호이며 ARX(Addition, Rotation, XOR)로 구성된 연산을 수행하여 암/복호화 한다. 소프트웨어 동작에 초점을 맞춰 설계되었으며, Microcontroller 플랫폼에 최적화된 암호이다. NSA에 의하면 SPECK이 가진 주요 강점으로 유연성과 성능을 제시하고 있다. SPECK은 다양한 블록/키 길이 (48/96, 64/96, 64/128, 96/96, 96/144, 128/128, 128/192, 128/256)을 지원하기 때문에, 다양한 환경에서 사용할 수 있다.

2.3.2 ECC (타원 곡선 암호)

ECC (타원 곡선 암호) 알고리즘은 이산대수에서 사용하는 유한체의 곱셉군을 타원곡선군으로 대치한 암호체계로써, 다른 암호체계에 비하여 짧은 키 사이즈로 대등한 안전도를 가지는 장점을 가지고 있다. ECC 알고리즘을 이용한 암호 시스템은 난수와 결합한 공개키를 각 단말에 공유하여 공격자가 유추할 수 없는 비밀 키로 동기화하고 암호화하는 순서로 진행하며, 이같은 암호 시스템을 구현하기 위해서는 키 분배 알고리즘과 메시지 암호 알고리즘이 구성된다. ECC 알고리즘의 키 분배 방식은 ECDH (Elliptic Curve Diffie-Hellman) 알고리즘이 대표적이다. 메시지 암호화는 비밀 키 계산 이후 단말이 메시지와 비밀 키를 연산하여 서버 송신 과정과 서버가 비밀 키를 이용하여 암호화 된 메시지를 연산하는 과정으로 진행된다[17]. 최근 들어, 성능 오버헤드 때문에 키를 미리 공유하는 방식으로 사용했던 PSK기반의 대칭키 암호의 보안 문제를 해결하기 위하여 IoT 환경에서 많이 사용되고 있다.

III. 제안 프로토콜

3.1 제안 프로토콜 구조

제안 프로토콜 구조는 Fig. 1.과 같으며, Publisher가 암호화된 메시지와 함께 Topic을 PUBLISH 하면, 메시지를 전달받은 Broker는 해당 토픽을 구독하는 Subscriber에게 메시지를 전달하는 구조이다.

Publisher와 Subscriber는 특정 토픽을 기준으로 메시지 PUBLISH/SUBSCRIBE 여부에 따라 구분된다. Publisher는 메시지를 경량 암호(lightweight cipher)로 암호화하여 전달하고 Subscriber는 전달 받은 메시지를 복호화하여 평문 메시지를 획득하므로 기밀성이 보장된다. 그리고 QoS 레벨을 이용해서 신뢰성 있는 서비스를 제공할 수 있다.

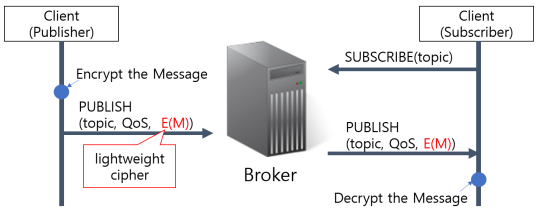

Fig. 1. A structure of end-to-end message security protocol

3.2 제안 프로토콜 메시지 구조

프로토콜 메시지 전체 구조는 Fig. 2.와 같이 2 바이트의 고정 헤더와 옵션에 따른 가변 헤더 그리고 암호화 된 페이로드로 구성 되어 있다.

"Remaining Length"는 가변 헤더 및 페이로드의 데이터를 포함한 전체 메시지의 크기를 계산하기 위해서 사용된다. "Remaining Length"의 최대

	7	6	5	4	3	2	1	0
Byte 1	Message Type: PUBLISH				DUP	QoS Level		RETAIN
Byte 2	Remaining Length (1-4 bytes)							
Byte 3 ... Byte n	Variable Length Header: Topic Name, Packet Identifier							
Byte n+1 ... Byte m	Encrypted(Variable Length Message Payload)							

Fig. 2. A structure of proposed protocol message

크기가 4 Bytes이기 때문에 전송 가능한 최대 메시지 크기는 $(2^7)^4$인 256 MB이다. "Message Type"은 제안 프로토콜의 메시지를 정의하기 위해 사용된다. "Message Type"에 따라 프로토콜의 고정 헤더, 가변 헤더 그리고 페이로드의 내용이 달라질 수 있다. "Message Type"에 대한 상세한 내용은 Table 1.과 같으며 "Message Type"을 "PUBLISH"로 설정하면 Publisher와 Broker에서 메시지를 발행 할 수 있고 메시지는 페이로드에 암호화되어 전달된다. 제안 프로토콜은 "Message Type"을 "PUBLISH"로 설정하고 페이로드를 암호화하여, 메시지를 발행 할 때 암호화된 메시지를 전달하여 통신 과정에서 기밀성을 제공한다. 암호화하여 전달 할 수 있는 페이로드의 크기는 $'256_{MB} - variable\ Length\ Header's\ size'$ 이다.

Table 1. A definition of Message Type

Name	Val	Description
Reserved	0	Reserved
CONNECT	1	Request to connect
CONNACK	2	Connect acknowledgement
PUBLISH	3	Publish message
PUBACK	4	Publish acknowledgement
PUBREC	5	Publish received
PUBREL	6	Publish release
PUBCOMP	7	Publish complete
SUBSCRIBE	8	Subscribe request
SUBACK	9	Subscribe acknowledgement
UNSUBSCRIBE	10	Unsubscribe request
UNSUBACK	11	Unsubscribe acknowledgement
PINGREQ	12	PING request
PINGRESP	13	PING response
DISCONNECT	14	Disconnect
Reserved	15	Reserved

3.3 제안 프로토콜 상세 내용

3.3.1 프로토콜 제안

과금 부과 등을 포함하는 IoT 서비스에서는 신뢰성이 중요한 이슈이며, TCP 기반의 제안 프로토콜은 애플리케이션 계층에서 3단계의 QoS를 제공하기 때문에 서비스에서 요구하는 수준의 신뢰성을 제공할 수 있다. 제안 프로토콜은 Fig. 3.과 같이 동작한다.

Publisher와 Subscriber 사이에 대칭키가 미리 공유되어 있다고 가정한다. Broker는 1883 포트를 통해 메시지를 중개한다. Fig. 3.의 경우 QoS 레벨이 '0'으로 설정되어있지만 서비스 요구사항에 맞게 레벨을 지정할 수 있다.

1, 2단계: 초기 세션을 맺는 과정으로, Subscriber와 Publisher는 각각 Broker에게 *CONNECT* 메시지를 전송하여 연결을 요청한다. 그리고 해당 메시지를 받은 Broker는 *CONNACK* 메시지로 응답하여 연결을 맺는다. 해당 세션은 client(Publisher와 Subscriber)가 세션을 중단할 때까지 유효하다.

3, 4단계: Publisher와 Subscriber는 *PINGREQ*와 *PINGRESP* 메시지를 이용해서 Broker와의 연결 상태를 확인한다.

5-1, 5-2단계: Publisher는 발행할 메시지를 사전에 공유된 대칭키로 암호화한다.

6단계: Subscriber는 자신이 구독하고자 하는 Topic을 보내 Publisher가 해당 Topic으로 발행하는 메시지를 수신하고자 요청한다.

7단계: Publisher는 서비스에 따라 QoS 레벨 옵

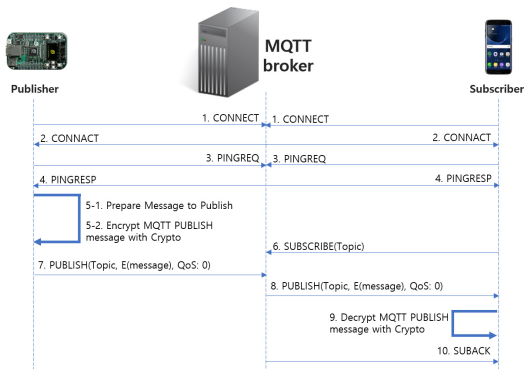

Fig. 3. An end-to-end message security protocol based lightweight cipher

션을 지정하고 Topic과 암호화된 메시지를 발행한다.

8단계: Broker는 해당 Topic을 구독하고자 요청했던 Subscriber에게 Publisher로부터 받은 메시지를 전달한다. 상세한 과정은 아래의 과정 8-1단계부터 8-4단계까지와 같다.

8-1단계: Broker는 Publisher로부터 받은 메시지를 큐(queue)에 저장한다.

8-2단계: *PUBLISH* 메시지에 설정되어있는 QoS 레벨에 따라 Broker는 Publisher에게 해당하는 *ACK*로 응답한다.

8-3단계: 구독을 요청한 Subscriber에게 큐에 저장해둔 메시지를 QoS 레벨과 함께 *PUBLISH* 메시지로 전송한다. Subscriber는 QoS 레벨에 따라 해당하는 *ACK*로 응답한다.

8-4단계: Broker는 retain 필드 값(true of false)에 따라 큐에 저장해둔 메시지를 삭제하거나 유지한다.

9단계: Subscriber는 사전에 공유된 대칭키로 전달받은 암호화된 메시지를 복호화하여 원하는 메시지를 획득한다. 이로써 Publisher/Subscriber 간의 메시지는 암호화 되어 전송되므로 기밀성을 보장받아 스니핑과 같은 위협을 대비할 수 있다.

3.4 제안 프로토콜 성능평가 및 비교

본 논문에서는 제안 프로토콜 성능을 평가하기 위한 실험 환경은 Fig. 4.와 같다. ARM Cortex-M3 기반의 QEMU 플랫폼에서 Zephyr v1.10.0을 기반으로 실험 환경을 구축했다. Zephyr가 운영되는 플랫폼에서 메시지를 PUBLISH 하고 Mosquitto Broker가 Mosquitto Client에게 발행된 메시지를 전달해주는 구조이다. 실행에 사용된 메시지 길이는 25바이트, 블록 크기는 128비트 키 길이는 256비트를 기준으로 하였다.

성능 시뮬레이션은 크게 세 가지의 Security Mode로 나누어 수행하였다. 세 가지 경우는 다음과 같다.

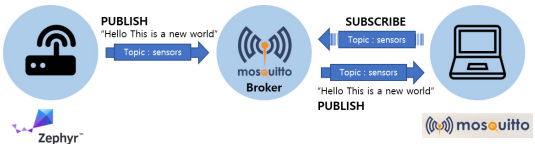

Fig. 4. An experimental environment of end-to-end message security protocol based on lightweight cipher

① No Cipher: 평문으로 통신

② AES: AES 암호를 적용하여 암호화 통신

③ SPECK(제안기법): SPECK 암호를 적용하여 암호화 통신

- SPECK 암호 블록 크기에 따라 암호화 통신

시뮬레이션을 통한 성능 측정 방법은 다음과 같다. 먼저 초기 시간에 대한 값으로 Subscriber에서 수신한 첫 번째 메시지의 복호화 처리가 끝난 지점의 Clock을 측정했다. 그리고 201 번째 수신한 메시지의 복호화 처리가 끝난 시점의 Clock 측정값과 첫 번째 메시지의 복호화 처리가 끝난 시점의 Clock 측정값의 차이를 구하였다(Fig. 5.의 'B-A'). 이와 같은 방법으로 메시지 통신 과정에서 암/복호화를 400회, 600회, 800회, 1000회 반복 수행하여 각 세트에 따른 실행 시간을 측정하였다. 또한 측정한 데이터를 바탕으로 1회 암/복호화 평균 실행 시간을 도출하였다. No Cypher 모드일 경우에도 암호화/복호화과정이 생략되므로 그 과정을 생략한 A시점부터 B시점까지가 된다.

Fig. 6.에서는 각각의 Security Mode에 대한 암/복호화 실행 시간을 보여주고 있다. 'No Cipher'의 경우 1회 실행 시간이 약 68.77 ms로 가장 빨랐다. SPECK은 블록 크기가 32/64/128 bits일 경우 각각 평균적으로 약 81.65/78.12/77.22 ms로 블록 크기가 커질수록 메시지 암/복호화 처리 시간이 빨랐다. 그리고 키 길이 128bits를 기준으로 AES와 SPECK을 비교했을 때는 AES_128은 약

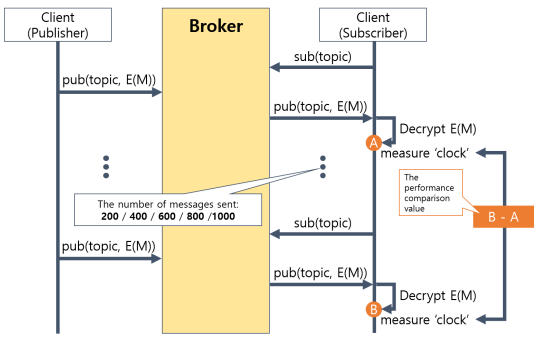

Fig. 5. A simulation of end-to-end message security protocol based on lightweight cipher

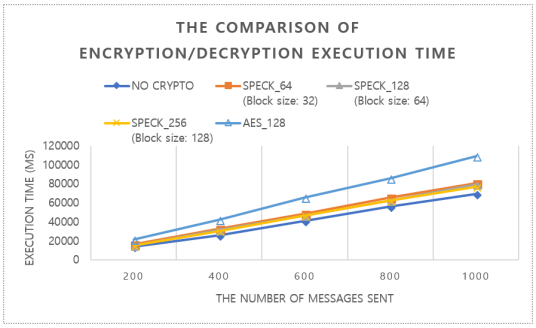

Fig. 6. A comparison of protocol performance

107.97 ms, SPECK_128은 약 78.12 ms로 측정되었다. 따라서 SPECK_128이 AES_128보다 처리 속도 측면에서의 성능이 대략 28% 더 우수함을 검증하였다. Table 2.에서는 AES와 SPECK의 메시지 전송 횟수에 따른 성능 부하를 보여주고 있다.

$$\text{성능부하} = \frac{(Encrypt - No\ Crypto)}{No\ Crypto} \times 100(\%)$$

암호로 인한 성능 부하를 분석한 결과 평균적으로 AES는 57.15%, SPECK은 13.78%이다. SPECK 기반 제안 프로토콜의 경우 성능 부하가 AES를 적용한 경우보다 작고 큰 비중을 차지하지 않아, 강한 보안을 요구하지 않는 서비스에 적용하여 메시지를 보호할 수 있다.

Table 2. A comparison of performance load (%)

mode / messages	AES_128	SPECK_128
200	51.58	6.40
400	64.68	23.42
600	58.46	12.53
800	53.50	11.58
1000	57.54	14.95

IV. 결 론

본 논문에서는 경량형 IoT 디바이스에 알맞은 안전한 MQTT 통신을 위해 경량 암호기반 종단간 메시지 보안 프로토콜을 제안하였다. Publisher와 Subscriber 그리고 그 사이에 Broker를 두어 애플리케이션 계층에서 종단간 메시지 보안을 제공하기 때문에 TLS의 세션 오버헤드로 인해 원활한 서비스 제공이 어려운 저성능 IoT 디바이스에서도 안전한 통신이 가능하도록 하였다. 또한 제안 프로토콜을 적용한 실행 시간 성능 시뮬레이션을 통해 그 성능이 기존 AES 기반의 프로토콜을 적용한 경우보다 성능이 더 우수함을 검증하였다. 또한, 본 논문에서 제안한 종단간 메시지 보안 프로토콜은 상시 전기가 들어오지 않거나 또는 성능상의 제약이 많은 IoT 기기의 경우에 배터리, CPU 성능 등의 제약사항을 고려하여 기존 TLS/SSL 프로토콜을 쓸 수 없으므로 패킷 페이로드만 암호화하여 사용할 수 있도록 하는 장점을 가진다.

본 제안 프로토콜의 안전성 분석은 우선 MQTT 프로토콜의 안전성을 증명함으로 어느 정도 증명이 될 것으로 본다. 본래 MQTT는 IBM에서 유선 장비와의 위성통신을 위해 개발한 프로토콜로 기본적으로 신뢰성과 저전력을 특징으로 하므로 IoT 네트워크에 적용하기에 적합하다. MQTT 표준은 개방형 표준을 위한 단체인 OASIA에 의해 채택되었으며 버전 3.1.1로 발표되어, 오픈 소스 스택과 컨설팅을 제공하는 많은 상업 회사와 이클립스(Eclipse) 커뮤니티 내의 지원을 받고 있다. 현재 본 제안 프로토콜에 핵심으로 구현된 MQTT 프로토콜은 표준 규격에 의해 구현된 프로토콜이므로 해당 규격에 의해 그 안전성은 검증이 되었으며, 이에 패킷 페이로드만 SPECK으로 암호화하여 플레인 텍스트로 전송되는 것을 막아 위협을 예방하는 것이므로 본 제안 프로토콜 또한 어느 정도 안전성은 검증 된 것으로 보여진다. 다만 본 제안 프로토콜을 정형적 방법론에 의해 안전성 분석을 하는 것은 Future Work으로 하여 추후 발견되는 보안 취약점들이 있을 수 있으니 설계 단계에서 보안 프로토콜의 안전성을 검증하는 정형적 방법론을 선정하고, 정형적 검증 방법론의 실효성을 보이기 위한 Casper 및 FDR 도구를 이용하여 본 제안 프로토콜의 안전성을 분석할 예정이다.

향후 연구에서는 제안한 프로토콜에 대해 IoT 서비스 목적에 따라 사용을 고려해 볼 수 있도록 다양한 경량 암호들을 적용하여 프로토콜 성능을 비교하고, 암호 및 보안 성능 개선을 위한 연구를 진행하고자 한다. 또한 본 제안 프로토콜의 안전성을 증명할 계획이다.

References

[1] MSIP, "The internet of things (IoT) information protection roadmap," Ministry of Science ICT and Future Planning, Oct. 2014.
[2] Rob van der Meulen, "Gartner says

8.4 billion connected 'Things' will be in use in 2017, up 31 percent from 2016," Gartner, Feb. 2017.
[3] Brussels Center/KBA Europe, "Application and development prospects of 'Internet of Things (IoT)' in industry sector," BMI, May. 2017.
[4] Chris Middleton, "Gartner: IoT security spend hitting $1.5 billion - but strategy poor," Internet of Business, Mar. 2018.
[5] Namseoul University, "In the 'Internet of Things', a research of actual cases and analysis of factors infringing privacy," Personal Information Protection Commission, Dec. 2015.
[6] Zhi-Kai Zhang, Michael Cheng Yi Cho, Chia-Wei Wang, Chia-Wei Hsu, Chong-Kuan Chen, Shiuhpyng Shieh, and IEEE Fellow, "IoT security: ongoing challenges and research opportunities," 2014 IEEE 7th International Conference on IEEE, pp. 230-234, Nov. 2014.
[7] T. Dierks and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2," NWG RFC 5246, Aug. 2008.
[8] TLS protocol stack, https://commons.wikimedia.org/wiki/File:TLS_protocol_stack.jpg
[9] About cipher suite of TLS, https://rsec.kr/?p=455
[10] E. Rescorla and N. Modadugu, "Datagram Transport Layer Security Version 1.2," IETF RFC 6347, Jan. 2012.
[11] Heer, T., Garcia-Morchon, O., Hummen, R., Keoh, S. L., Kumar, S. S., and Wehrle, K., "Security Challenges in the IP-based Internet of Things," Wireless Personal Communications, vol. 61, no. 3, pp. 527-542, Dec. 2011.
[12] ISO/IEC, "Information technology -- Message Queuing Telemetry Transport (MQTT) v3.1.1," ISO/IEC 20922:2016, Jun. 2016.
[13] Facebook messanger using MQTT, https://d2.naver.com/helloworld/1846
[14] Andrew Banks and Rahul Gupta, "MQTT Version 3.1.1 Plus Errata 01," OASIS Standard Incorporating Approved Errata 01, Dec. 2015.
[15] Zach Shelby, Klaus Hartke, and Carsten Bormann, "The Constrained Application Protocol (CoAP)," IETF RFC 7252, Jun. 2014.
[16] Beaulieu, R., Treatman-Clark, S., Shors, D., Weeks, B., Smith, J., and Wingers, L., "The SIMON and SPECK lightweight block ciphers," Design Automation Conference (DAC), 2015 52nd ACM/EDAC/IEEE, pp. 1-6, Jun. 2015.
[17] K. Maletsky, "RSA vs ECC Comparison for Embedded Systems," Security ICs White Paper, 2015.

〈저자소개〉

김 희 정 (Hee-jeong Kim) 학생회원
2015년 8월: 금오공과대학교 컴퓨터공학과 졸업
2016년 9월~현재: 과학기술연합대학원대학교 ICT(정보보호공학)과 석사과정
〈관심분야〉 IoT 보안, 시스템·네트워크 보안

김 정 녀 (Jeong-nyeo Kim) 종신회원
1987년 2월: 전남대학교 전산통계학과 졸업
2000년 2월: 충남대학교 컴퓨터공학과 석사
2004년 2월: 충남대학교 컴퓨터공학과 박사
1988년~현재: 한국전자통신연구원 정보보호연구본부 책임연구원
1996년: OSF/RI 공동연구 파견(미국)
2005년: Univ. of California, Irvine Post-Doc.
2015년~현재: 과학기술연합대학원대학교(UST) ICT(정보보호공학)과 교수
〈관심분야〉 IoT 보안, 모바일 보안, 시스템·네트워크 보안, 보안 OS 등