Journal of Institute of Control, Robotics and Systems (2016) 22(9):766-772
http://dx.doi.org/10.5302/J.ICROS.2016.16.0058 ISSN:1976-5622 eISSN:2233-4335

IoT 시스템을 위한 시간 동기화 방식 기반 SEED 알고리즘

One Time Password-Based SEED Algorithm for IoT Systems

이 성 원, 박 승 민, 심 귀 보*
(Sung-Won Lee[1], Seung-Min Park[1], and Kwee-Bo Sim[1,*])
[1]Departure of Electrical and Electronics Engineering, Chung-Ang University

Abstract: Recent advances in networking and computers, especially internet of things (IoT) technologies, have improved the quality of home life and industrial sites. However, the security vulnerability of IoT technologies causes life-threatening issues and information leakage concerns. Studies regarding security algorithms are being conducted. In this paper, we proposed SEED algorithms based on one time passwords (OTPs). The specified server sent time data to the client every 10 seconds. The client changed the security key using time data and generated a ciphertext by combining the changed security key and the matrix. We applied the SEED algorithms with enhanced security to Linux-based embedded boards and android smart phones, then conducted a door lock control experiment (door lock & unlock). In this process, the power consumed for decryption was measured. The power consumption of the OTP-based algorithm was measured as 0.405-0.465W. The OTP-based algorithm didn't show any difference from the existing SEED algorithms, but showed a better performance than the existing algorithms.

Keywords: one time password, SEED algorithm, security key, information security, IoT

I. 서론

네트워크 통신의 발달로 사물인터넷(Internet of Things)은 산업단지나 홈 네트워크 등 다양한 분야에 빠르게 보급화 되어가고 있다[1,2]. 하지만 사물인터넷은 네트워크를 활용한 시스템으로 해킹에 대한 취약점을 가지고 있다. 해킹은 네트워크를 통해 상대방의 시스템에 접근하여 악영향을 끼치는 행위로 데이터를 위조, 복조 시켜 통신을 방해하거나 바이러스를 심어 정보를 유출시키는 등의 피해를 입힐 수 있다[3]. 또한 최근 개발되고 있는 무인 자동차도 해킹에 의해 제어될 수 있어 데이터 보안의 중요성이 강조되고 있다[4-6].

보안 알고리즘으로는 대칭형 알고리즘과 비대칭형 알고리즘으로 구분된다. 비대칭형 알고리즘은 대표적으로 RSA (Rivest, Shamir, Adleman) [7], DSA (Digital Signature Algorithm) [8]가 있다. 비대칭형 알고리즘은 암호화 키와 복호화 키가 동일하지 않아 복잡한 산술연산을 통해 복호화를 수행하며 대칭형 알고리즘에 비해 보안성이 좋은 반면 암호화와 복호화 하는 시간이 많이 소요된다. 이러한 이유로 인증서나 전자서명에 주로 사용된다. 대칭형 알고리즘은 대표적으로 SEED [9], LEA (Lightweight Encryption Algorithm) [10], HIGHT (High security and light weigHT) [11] 등이 있다. 대칭형 알고리즘은 송신 측과 수신 측의 비밀 키가 같아 암호화, 복호화 속도가 빠르다는 장점을 가지고 있다. 이중 SEED 알고리즘은 국내 한국인터넷진흥원(KISA)에서 개발하였으며 ISO/IEC 국제표준에 제정되었다.

SEED 알고리즘은 s-box 테이블과 비밀 키를 사용하여 암호화, 복호화를 수행한다. 이때 사용되는 비밀 키는 외부에 노출되면 속수무책으로 정보가 유출되는 문제가 있다. 또한 알고리즘의 보안성을 위해 주기적으로 비밀 키는 변경해 주어야 하는 문제점이 있다. 이러한 문제점으로 인해 SEED 알고리즘의 취약점이나 결함을 찾아 개선하는 등 다양한 연구가 진행되고 있다.

본 논문에서는 위에 언급된 문제점을 보완하기 위해 비밀 키를 이용하여 기존 알고리즘을 다음과 같이 개선하였다. SEED 알고리즘은 송신 측과 수신 측의 동기화를 위한 비밀 키와 주기적으로 변경하기 위한 보조 비밀 키 그리고 행렬 비밀 키로 구성하였다. 동기화를 위한 비밀 키는 서버에 신규 클라이언트가 접속하게 되면 서버와 접속되어 있는 모든 클라이언트에게 브로드캐스팅으로 특정 데이터를 전송한다. 이때 데이터를 수신 받은 클라이언트는 동기화를 위한 비밀 키로 변경한다. 보조 비밀 키는 서버의 시간을 기준으로 현재 날짜와 시간을 데이터화하여 10초마다 주기적으로 클라이언트에게 전송하였다. 데이터를 수신 받은 클라이언트는 시간 데이터를 이용해 서버와 동일한 보조 비밀 키로 변경한다. 또한 행렬 비밀 키는 암호문을 다시 암호화하기 위한 비밀 키이다. 이러한 알고리즘의 실효성을 확인하기 위해 개선된 SEED 알고리즘을 도어록 시스템에 적용하여 암호화, 복호화를 검증하였으며 기존의 SEED 알고리즘과 소비 전력을 비교하였다[12].

II. 관련연구

사물인터넷의 데이터 보안이 중요시되고 있다. 하지만 알고리즘의 보안성이 점점 낮아짐에 따라 알고리즘을 개선

* Corresponding Author
Manuscript received March 28, 2016 / revised May 31, 2016 / accepted July 15, 2016
이성원, 박승민, 심귀보: 중앙대학교 전자전기공학부
(sungwon8912@cau.ac.kr/sminpark@cau.ac.kr/kbsim@cau.ac.kr)
※ 본 논문은 한국연구재단 중견연구자지원사업(No. 2012-000872)에서 지원하여 연구하였으며 연구비 지원에 감사드립니다.

하는 연구가 진행되고 있다.

장태민 외 1인은 SEED 알고리즘의 속도 향상을 위해 F 함수를 개선하였다[13]. F 함수를 통해 출력된 암호문 C'과 D'을 서로 교차시켜 새로운 암호문을 생성하는 라운드 구조를 설계하였다. 또한 라운드 키를 생성하는 과정에 필요한 s-box은 4개의 테이블 중 s1 box, s2 box 2개를 사용하여 데이터 처리 속도를 빠르게 하였으며 하드웨어 오버헤드를 줄일 수 있도록 하였다. 이러한 알고리즘을 통해 소비전력의 최소화와 암호화, 복호화의 처리 속도를 상당히 높일 수 있다. 하지만 테이블의 개수를 줄여 암호문을 생성하는 과정에서 기존의 SEED 알고리즘보다 보안성이 떨어질 수 있다.

김태원 외 1인은 Cho가 제안한 Masked SEED 알고리즘의 취약점 찾아내 분석하였다[14]. Masked SEED 알고리즘은 마스크 값에 의존하여 연산의 차이가 발생하는 취약점을 가지고 있다. 전력 파형을 측정하여 차이를 확인하였으며 마스크 값을 복원하였다. 전력 파형의 차이는 시각적으로 분석하였고 수집된 파형을 실시간으로 복원하여 평문 데이터와 키를 추측하였다. 이를 이용해 1차 전력분석 공격을 수행하였다. 또한 1차 전력분석을 통해 마스크 정보를 복원하여 비밀 키를 찾아내어 취약점을 검증하였다. 개선된 SEED 알고리즘을 분석하여 문제점을 파악하였으며 개선되어야할 부분을 확인할 수 있어 본 논문에 참고가 되었다.

권태웅 외 2인은 GEZEL을 이용하여 SEED 알고리즘과 ARIA 알고리즘의 설계 방법을 제안하였다[15]. GEZEL을 이용해 SEED 알고리즘의 데이터 패스를 만들고 한 클록에 동작이 가능하도록 하였으며 각 연산 과정 사이에 레지스터를 두어 파이프 라이닝 방식으로 동작되도록 하였으며 기존의 설계과 다른 설계 방법을 제안하여 데이터 속도를 향상시키도록 하였다. GEZEL을 이용한 알고리즘 설계로 인해 데이터 처리의 속도가 향상 되도록 구현하였다. 사물인터넷의 저전력 향상을 위한 방법으로 도움이 될 것이다.

Pirpilidis 외 2인은 하드웨어를 최소화하는 방법을 제안하였다[16]. 제안된 기술은 파이프라인 기술을 암호화 라운드에 적용하였다. 기존의 알고리즘에 사용되는 F 함수와 G 함수의 구조를 개선하여 효율성을 검증하였다. 이 또한 속도 향상과 저전력에 효율성이 증가시켜 사물인터넷에 적용할 수 있을 것이다.

관련 연구에서는 소비 전력을 줄이고, 속도를 향상시키는 연구와 취약점에 대한 평가를 이루는 연구를 몇 가지 살펴보았다. 이 외에도 SEED 알고리즘의 적합성, 성능 개선 등 다양한 연구가 진행되고 있다[17-24].

III. SEED 알고리즘

표 1은 SEED 알고리즘에 기본 구조에 사용되는 연산기호들을 나타내는 표이다.

그림 1은 SEED 알고리즘의 전체 구조이다. SEED 알고리즘은 128비트 암호화 키와 복호화 키를 이용하여 임의의 길이를 갖는 평문을 128bit의 블록 단위로 처리하는 블록암호 알고리즘이다. 평문 메시지를 각각 L0와 R0에 64bit씩 블록단위로 나누어 Addition, XOR, bit-rotation으로 연산을 하며 총 16라운드를 거쳐 암호화한다. SEED 알고리즘은

표 1. SEED 알고리즘 연산 표기.

Table 1. The logic symbol of SEED Algorithm.

Symbol	Value
$\boxplus$	addition in modular 2^{32}
$\boxminus$	subtraction in modular 2^{32}
$\oplus$	bitwise exclusive OR
&	bitwise AND
$\ll$ n	left circular rotation by n bits
$\gg$ n	right circular rotation by n bits
$\|$	concatenation

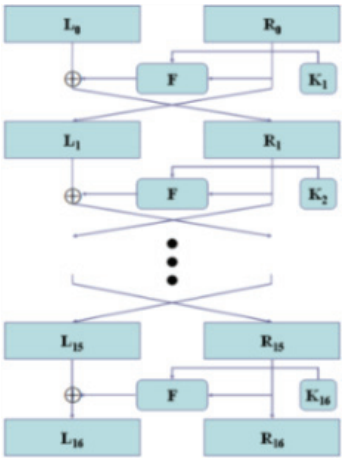

그림 1. SEED 알고리즘의 전체 구조.

Fig. 1. The overall structure of the SEED algorithm.

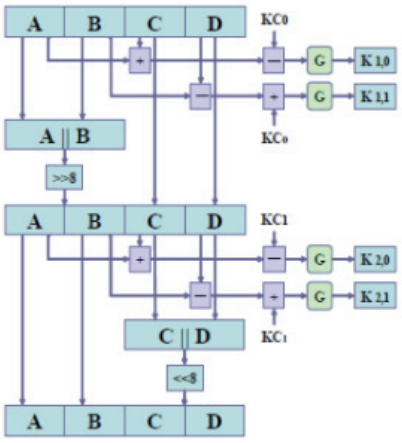

그림 2. 라운드 키 생성과정.

Fig. 2. Round key generation process.

Feistel [25] 구조로 되어 있는 F 함수와 G 함수를 이용해 암호문을 생성한다. F 함수에서는 비밀 키 값이 대입되고 F 함수 안에 있는 G 함수를 통해 암호화한다. 같은 방법으로 총 16번의 라운드를 거쳐 암호화를 이루고 마지막으로 L16과 R16에 대입된다.

그림 2는 라운드 키가 생성되는 과정이다. KC_i는 비밀 키에 의해 생성된 라운드 키이며 기존 SEED 알고리즘에서 비밀 키가 지속적으로 변경되지 않아 개선된 SEED 알고리

즘에서는 시간 주기로 비밀 키를 변경되도록 하여 암호화에 사용되는 라운드 키 KC_i 를 지속적으로 갱신되도록 하였다. 또한 암호화 과정중 행렬 암호화를 추가하여 암호문의 복잡도를 향상시켰다.

IV. 시간동기화 방식 기반 SEED 알고리즘

제안된 SEED 알고리즘은 새로운 비밀 키로 변경하기 위한 비밀 키, 서버와 클라이언트를 동기화하기 위한 비밀 키, 행렬 비밀 키 등 3 가지의 비밀 키를 가지고 있다. 새로운 비밀 키는 모듈라와 XOR 연산을 통해 비밀 키 값을 변경하였다. 암호화 시 변경된 비밀 키를 사용해 기존 SEED 알고리즘과 동일한 방법으로 암호화하였다. 출력된 암호문은 복잡도를 높이기 위해 행렬식으로 다시 암호화하였다. 비밀 키 값이 노출되더라도 다음에 전송될 데이터를 예측할 수 없도록 하였으며 일정한 시간이 지나면 비밀 키 값이 사라져 복호화를 할 수 없도록 하였다.

1. 비밀 키 변경

새로운 비밀 키를 변경하기 위해 서버의 시간을 데이터화하였다. 그림 3은 시간 데이터가 연산되는 과정을 보여준다. tK는 시간을 나타내는 함수로 사용하였다. $tK0$는 날짜, $tK1$은 시간, $tK2$는 분, $tK3$는 초 단위로 구성하였다. 시간 데이터 $tK0$, $tK1$, $tK2$와 $tK3$를 모듈라 연산을 하여 $tKi \boxplus tK3$와 같이 연산하였다. 시간 데이터의 동일한 값이 출력되는 것을 방지하기 위해 $tK3$와 $tK0$, $tK1$, $tK2$을 조합하였다. tK_3은 $(tK1 \boxplus tK2) \oplus tK3$와 같이 연산하여 비밀 키를 생성하는데 조합하였다.

128바이트의 암호화 데이터를 64바이트씩 나누어 비밀 키를 변경하였다. 그림 4는 $K0 \sim K3$ 자리와 $K4 \sim K7$ 자리의 비밀 키를 변경하기 위한 라운드이다. 그림 3에서 연산된 시간 함수 데이터와 현재 비밀 키를 연산하여 새로운 비밀 키를 변경하였다. 각 비밀 키 값은 규칙적이지 않도록 비밀 키마다 연산과정을 다르게 하여 비밀 키를 생성하였다. K'은 연산된 비밀 키를 나타내었다. 각 자리의 비밀 키는 식 (1)과 같이 연산하였다.

$$\begin{aligned} K0' &= (tK3 \oplus K2') \boxplus K3 \\ K1' &= (tK2 \oplus K3') \boxplus K2 \\ K2' &= (tK1 \oplus K2) \boxplus K1 \\ K3' &= (tK0 \oplus K3) \boxplus K0 \end{aligned} \tag{1}$$

$K4 \sim K7$의 자리의 비밀 키 또한 동일한 방법으로 비밀 키를 변경하였다.

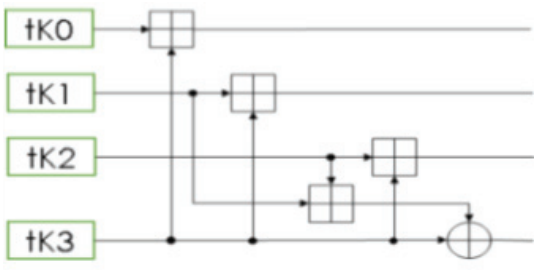

그림 3. 비밀 키 변경을 위한 시간 함수.
Fig. 3. Time function for change the key.

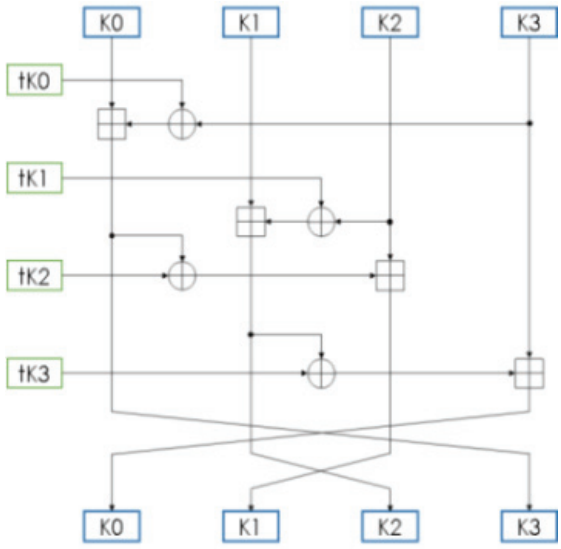

그림 4. 비밀 키 $K0 \sim K7$ 변경.
Fig. 4. The change of security key from K0 to K7.

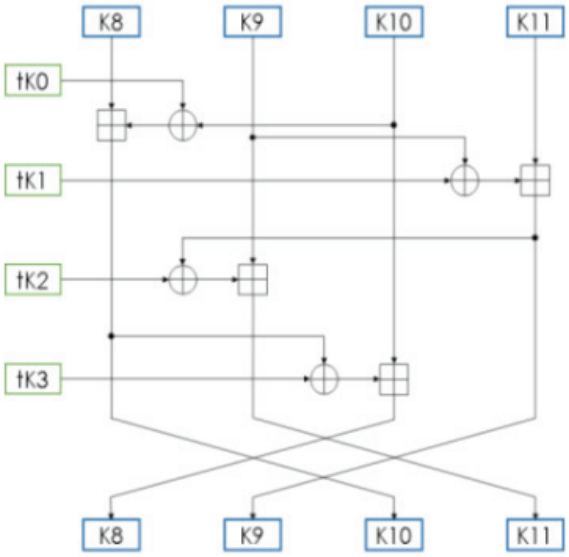

그림 5. 비밀 키 $K8 \sim K15$ 변경.
Fig. 5. The change of security key from K8 to K15.

그림 5는 $K8 \sim K11$ 자리와 $K12 \sim K15$ 자리의 비밀 키를 연산하기 위한 라운드이다. 나머지 64바이트 8자리의 비밀 키 또한 비슷한 방법으로 식 (2)와 같이 연산하였다.

$$\begin{aligned} K8' &= (tK3 \oplus K10') \boxplus K10 \\ K9' &= (tK1 \oplus K9) \boxplus K11 \\ K10' &= (tK0 \oplus K10) \boxplus K8 \\ K11' &= (tK2 \oplus K11) \boxplus K9 \end{aligned} \tag{2}$$

2. 행렬 암호화

행렬 암호화는 1byte 단위로 임의의 숫자 4자리를 구성된 암호화 보조 비밀 키와 복호화 보조 비밀 키로 구성하였다. 비밀 키가 노출되어도 행렬 보조 비밀 키가 노출되지 않을 시 복호화 할 수 없도록 하였다. 행렬 암호화는 비밀 키를 통해 출력된 암호문 16자리를 4자리씩 나누어 행렬

암호화를 수행한다. 복호화 또한 16자리의 암호문을 4자리씩 나누어 보조 비밀 키로 행렬 복호화를 수행한 후 비밀 키로 복호화를 수행하여 평문 데이터로 변환한다. 식 (3)에 C는 16자리의 암호문이고 M은 행렬 암호화에 사용되는 보조 비밀 키이다. 행렬 보조 비밀 키 M과 암호문 C16자리를 $C0$~$C3$, $C4$~$C7$, $C8$~$C11$, $C12$~$C14$ 4자리씩 나누어 행렬 암호화하였다. 암호문은 식 (3)과 같이 2x2 행렬로 암호화하여 $C'i$에 대입하여 16자리 암호문을 생성한다.

$$\begin{aligned} C'0 &= (M0^{*} C0) + (M1^{*} C2) \\ C'1 &= (M0^{*} C1) + (M1^{*} C3) \\ C'2 &= (M2^{*} C0) + (M1^{*} C2) \\ C'3 &= (M2^{*} C1) + (M1^{*} C3) \end{aligned} \tag{3}$$

V. 실험 및 결과

1. 복호화 분석

서버와 클라이언트가 서로 접속된 상태에서 새로운 클라이언트를 생성하여 서버에 접속하였다. 새로운 클라이언트는 해킹을 가장한 클라이언트로 암, 복호화를 확인하는 절차를 위해 생성하였다. 현재 사용되고 있는 동일한 비밀 키를 한번만 전달해주었고 서버에 접속하여 주기적으로 데이터를 전송하여 복호화를 시도하였다. 새로운 클라이언트는 기존 클라이언트의 암호화 데이터의 복호화가 이루어지는 과정을 확인하였다. 서버와 기존 클라이언트는 10초가 지난 후 시간 데이터에 의해 비밀 키가 갱신된다. 그 후 새로운 클라이언트의 데이터를 복호화하지 못하는 과정을 확인하였다. 비밀 키가 노출되더라도 해당 비밀 키는 10초만 유효하게 된다. 10초가 지난 후 서버는 시간 데이터를 클라이언트들에게 전송하며 클라이언트는 전송받은 시간 데이터를 서버와 동일한 연산을 통해 비밀 키 갱신 및 동기화가 이루어진다. 공격자는 새로운 비밀 키를 알아야하며 비밀 키에 의해 생성된 암호문은 행렬암호화로 다시 암호화하기 때문에 비밀 키가 노출되더라도 암호문을 해독하기 어렵게 된다.

그림 6은 서버와 클라이언트로 구성할 라즈베리파이 2개와 클라이언트 스마트폰 1개로 구성하여 위와 같은 방법으로 실험하였다. 그림에 Client 1은 해킹을 가장한 클라이언트, Client 2는 사용자 스마트폰이다. 평문 데이터는 0 ~ f로 send decrypt에 0 ~ f 가 출력되어야 복호화가 되는 것을 확인할 수 있다. 그림에 Client1과 Client2가 복호화되는 것을 확인할 수 있다. 서버는 10초가 지난 후 Client2에게 시간 데이터를 전송하여 연산을 통해 비밀 키를 갱신한다. 이때 변경된 비밀 키 pbUser_Key__ “eb 5c ... 61 60”를 통해 확인할 수 있다. 다시 동일한 방법으로 데이터를 전송한다. 사용자의 스마트폰 Clinet2는 복호화를 되었고 Clinet1은 복호화 되지 않아 이전 비밀 키는 유효하지 않음을 확인할 수 있다.

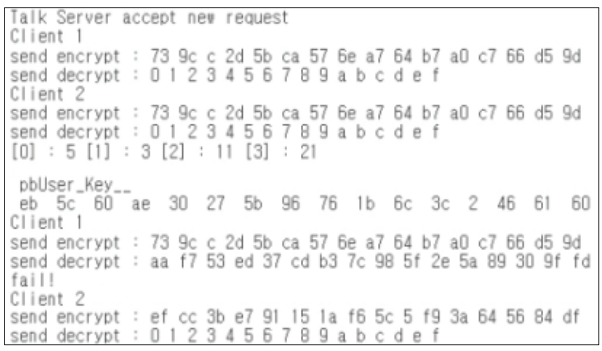

그림 6. 모의 해킹 실험.

Fig. 6. Penetration testing.

2. 소비 전력 분석

32비트 RAM 1G, ARM 계열의 CPU가 탑재되어 있고 리눅스 계열의 라즈비안이 설치되어 있는 라즈베리 파이2를 이용하여 암호화, 복호화를 총 50번의 거쳐 소비 전력을 측정하였다. 전력을 측정하기 위해 오실로 스코프를 이용하여 라즈베리 파이의 복호화 할 때의 소비전력과 평균 대기 소비전압을 측정하였다. 측정 장비는 Oscilloscope : GWINSTEK사의 GDS-1102A-U와 Oscilloscope Probe : Texas 사의 p6100 그리고 1K 저항을 사용하였다. 측정 방법은 라즈베리파이의 전원 전압 5V 사이에 1K 저항을 연결하고 스코프로 저항의 양단을 측정하였다. 라즈베리파이에서 복호화가 이루어질 때 측정된 전압을 $P = VI$ 를 이용하여 전력 사용량을 구하였다.

그림 7은 기존 SEED 알고리즘의 복호화 소비 전력이다. 기존 알고리즘과 개선된 알고리즘의 복호화 시 소비 전력을 비교하였다. 기존 SEED 알고리즘은 전압 60[mV]~80[mV]가 소비되는 것을 확인하였으며 개선된 알고리즘과 비교했을 때 전력 소비량의 차이가 크지 않은 것을 확인하였다. 소비 전력을 계산하면 약 0.465[W]가 소비되는 것을 확인하였고 소비 전력이 줄어들지는 않았지만 암호문이 더욱 복잡해져 암호문을 예측하기 어려울 것이다.

그림 8은 개선된 알고리즘으로 복호화 시 사용된 전압이다. 실험은 라즈베리 파이의 공급 전원에 저항을 추가하여 오실로스코프로 저항의 전압 변화를 측정하였다. 측정 결과 라즈베리 파이의 대기 소비전력은 약 174[mW]로 확인하였고 라즈베리 파이에서 개선된 알고리즘에서 복호화를 진행할 때 마다 그림 8처럼 전압이 상승하는 구간을 확인하였

그림 7. SEED 알고리즘의 복호화 출력 전압.

Fig. 7. Decryption output voltage signal of SEED algorithm.

그림 8. 제안한 OTP 기반 SEED알고리즘의 복호화 출력 전압.

Fig. 8. Decryption output voltage signal of proposed algorithm.

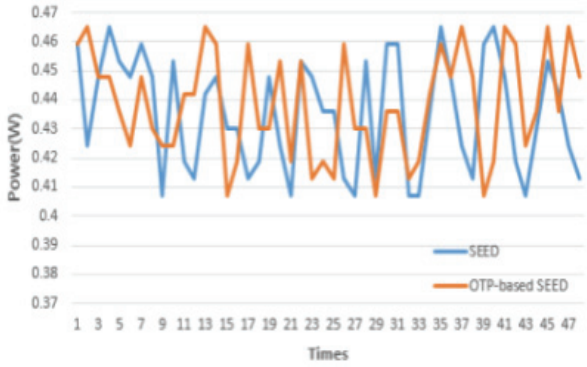

그림 9. 기존 알고리즘과 개선된 알고리즘 소비 전력 비교.

Fig. 9. Power Comparison of the improved algorithms of existing algorithms.

다. 이때 Peak 치를 기준으로 60~80[mV] 정도가 나왔으며 옴의 법칙으로 계산할 때 93[mA]가 나왔다. 이때 라즈베리 파이 정격전압 5[V]와 전류를 곱하여 소비 전력을 구하였다. 결과적으로 복호화를 할 때마다 평균 약 0.465[W]의 전력이 소비되는 것을 확인하였다.

그림 9는 SEED 알고리즘과 제안한 시간동기화방식 기반 SEED 알고리즘의 소비전력을 나타낸 그래프이다. x 축은 복호화를 시도한 횟수이며 y 축은 소비되는 전력을 나타내었다.

3. 실험과정

서버는 신규 클라이언트가 접속하면 서버에 접속 되어 있는 모든 클라이언트에게 브로드 캐스팅하여 특정 데이터를 전송하도록 하였다. 이때 특정 데이터가 수신되면 클라이언트는 동기화하기 위한 비밀 키로 변경하여 서버와 비밀 키 값을 동기화하였다. 또한 일정한 시간 내에 복호화되지 않으면 암호문을 해독할 수 없도록 하였다.

그림 10은 암 · 복호화를 이루어지는 과정을 확인하기 위한 애플리케이션이다. 클라이언트는 스마트폰을 사용하여 암호화된 데이터를 라즈베리 파이 서버로 전송하도록 하였다. 서버는 라즈베리 파이와 도어록을 구성하여 복호화가

그림 10. 암 · 복호화 애플리케이션.

Fig. 10. Application to verify encryption and decryption.

그림 11. 도어록 제어를 위한 암호화 데이터 검증.

Fig. 11. Verification of the encrypted data for the door lock control.

이루어지면 도어록이 개폐되도록 하였다.

서버와 연결된 스마트폰은 Door Open/Close 버튼을 통해 암호문을 전송하였다. 그림 8은 애플리케이션의 암호화, 복호화 이루어지는 데이터를 보여준다. 암호화된 평문 데이터는 $\{0X00,\ 0X01,\ 0X02 \cdots\ 0X14,\ 0X15\}$로 선언된 byte 변수 16자리를 전송하였다.

그림 11은 Encryption은 암호화 과정을 보여준다. Encryption에 enc Data는 시간 함수와 현재 비밀 키의 조합으로 이루어진 암호문이며 Mat Data는 행렬식으로 암호문을 암호화한 데이터를 보여준다.

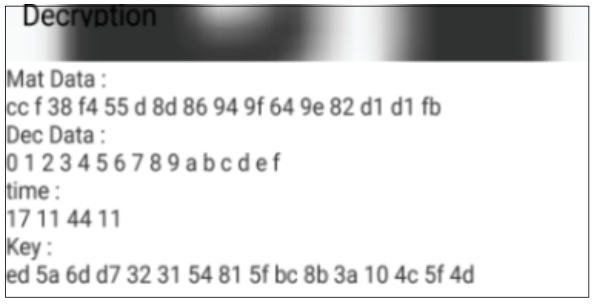

그림 12. 도어록 제어를 위한 복호화 데이터 검증.

Fig. 12. Verification of the decoded data for the door lock control.

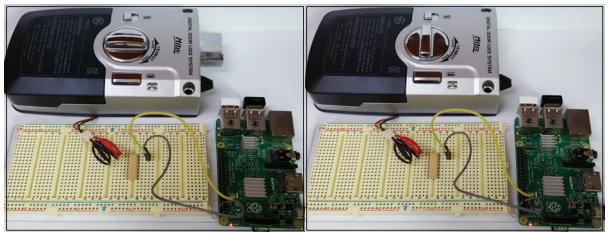

그림 13. 암 · 복호화가 이루어져 도어록이 개폐되는 모습.
Fig. 13. How to open and close the door from the encryption and decryption.

그림 12는 Decryption은 복호화 과정을 보여준다. Decryption에 Mat Data는 행렬로 암호화된 데이터를 역행렬을 취해 복호화하였다. 복호화 된 암호문은 처음 암호화한 enc Data와 동일한 암호문이 출력되는 것을 확인할 수 있다. Dnc Data는 enc Data를 복호화하여 출력된 평문 데이터이다. time은 서버로부터 수신 받은 시간이며 Key는 수신 받은 시간 데이터를 이용해 변경된 비밀 키이다.

그림 13은 도어록의 암 · 복호화가 이루어져 도어가 개폐되는 모습을 보여준다. 스마트폰과 라즈베리 파이는 TCP/IP 프로토콜을 이용하여 통신하였다. 도어록은 잠금장치를 on/off 하는 스위치와 라즈베리 파이 GPIO 포트 사이에 포토커플러를 연결하여 GPIO 포트에서 지속적으로 신호가 출력되는 것을 방지하기 위해 사용하였다. 또한 발광부는 라즈베리 파이의 GPIO 포트와 연결하였고 수광부는 도어록 스위치와 연결하여 도어록을 제어하였다.

VI. 결론

본 논문에서 SEED 알고리즘을 개선하여 도어록에 적용하였다. 비밀 키는 서버에서 클라이언트에게 서버 시간을 10초마다 전송하여 비밀 키가 변경 되도록 하였고 정해진 시간 내에 복호화하지 않으면 비밀 키 값이 유효하지 않게 하여 복호화를 할 수 없도록 하였다. 또한 새로운 클라이언트가 접속되면 서버와 연결되어 있는 모든 클라이언트에게 특정 데이터를 전송하여 비밀 키를 동기화하였다.

본 논문에서 제안한 알고리즘은 도어록을 통해 암호화, 복호화가 이루어지는 과정을 확인하였다. 비밀 키를 시간 주기로 변경시켜 보안성을 증가시켰음에도 알고리즘의 암 · 복화에 사용되는 전력 소비량이 기존 알고리즘과 큰 차이를 없음을 확인하였다. 본 논문에서 제안한 알고리즘은 사물인터넷 보안 시스템 및 보안 시스템에 활용이 가능할 것이다.

REFERENCES

[1] J. H. Kang, H. J. Kim, and M. S Jun, “Market and Technical Trends of internet of things,” *The Korea Contents Association*, vol. 13, pp. 14-17. Mar. 2015.

[2] W. S. Jeong, S. H. Kim, and K. S. Min, “An analysis of the economic effects for the IoT industry,” *Korean Society for Internet Information*, vol. 14, pp. 119-128, Oct. 2013.

[3] H. S. Ryu and J. Kwak, “Analysis of security threats and security requirements in smart home,” *Korean Society for Internet Information*, pp. 113-114, Oct. 2014.

[4] W. J. Chang and Y. T. Shin, “A study on the network and security for the internet of things,” *Korean Institue Of Information Technology*, pp. 19-21, Jun. 2015.

[5] B. I. Jang and C. S. Kim “A study on the security technology for the internet of things,” *Journal of Security Engineering*, vol. 11, no. 5, pp. 429-438, Oct. 2014.

[6] H. N. Chin, S. C. Park, and W. T. Choi, “Analysis of network security technology for Internet of Things (IoT),” *Korean Society for Internet Information*, pp. 353-354. Oct. 2014.

[7] R. L. Rivest, A. Shamir, and L. Adleman, “A method for obtaining digital signatures and public-key cryptosystems,” *Communications of the Association for Computing Machinery*, vol. 21, pp. 120-126, Feb. 1978.

[8] C. Kerry and P. Gallagher, “Digital Signature Standard (DSS),” NIST, MD 20899-8900, Jul. 2013.

[9] K. S. Chun, Y. J. Lee, J. Y. Kim, H. R. Park, and H. S. Ju, “128-bit block cipher SEED,” TTAS.KO-12.0004/R1. Dec. 2005.

[10] J. H. Park, D. J. Hong, D. C. Kim, D. S. Kwon, and H. R. Park, “128-Bit block cipher LEA,” TTAK.KO-12.0223. Dec. 2013.

[11] K. S. Chun, S. J. Lee, Y. J. Yeom, H. R. Park, H. Kim, J. H. Back, and J. H. Kim, “64-bit block cipher HIGHT,” TTAK.KO-120040/R1. Dec. 2008.

[12] S. W. Lee, J. H. Yu, and K. B. Sim, “Real-time streaming and remote control for the smart door-lock system based on internet of things,” *Journal of Korean Institute of Intelligent Systems*, vol. 25, pp. 565-570, Dec. 2015.

[13] T. M. Chang and M. S. Kang, “Design of SEED cipher processor based on modified F function,” *Journal of Security Engineering*, vol. 10, pp. 503-512. Aug. 2013.

[14] T. W. Kim and N. S. Chang, “Analysis on vulnerability of masked SEED algorithm,” *Journal of The Korea Institue of Information Security & Cryptology*, vol. 25, pp. 739-747. Aug. 2015.

[15] T. W. Kwon, H. M. Kim, and S. K. Hong, “SEED and ARIA algorithm design methods using GEZEL,” *Journal of Korea Institute of Information Security & Cryptology*, vol. 24, pp. 15-29. Feb. 2014.

[16] F. Pirpilidis, P. Kitsos, and A. Kakarountas, “A compact design of SEED block ciper,” *Embedded Computing (MECO), 2015 4th Mediterranean Conference on*, pp. 119-123. Jun. 2015.

[17] H. B. Song and G. Y. Cho, “A study on the constitution of S box and G function in SEED-type

cipher," *The Korean Institue of Communications and Information Sciences*," pp. 592-597. Jan. 1992.

[18] Y. Kim, C. H. Jung, Y. S. Jang, S. G. Lee, and S. G. Lee, "On the SEED validation system," *Korea Institute of Information Security & Cryptology*, vol. 13, no. 1, pp. 69-85, Feb. 2003.

[19] I. S. Ahn, T. S. Choi, E. B. Choi, S. H. Lim, and S. C. Sakong, "The proposal and chop design of the expanded SEED cipher algorithm," *The Institute of Electronics and Information Engineers*, vol. 40, pp. 30-41, Mar. 2003.

[20] K. W. Lee and S. Y. Ohm, "A study on pipeline chip of SEED block cipher algorithm," *Korean Institute of Information Scientists and Engineers*, vol. 28, pp. 43-45, Oct. 2001.

[21] I. S. Ahn, "A study on the cipher algorithm for the communication system," *The Institute of Electronics and Information Engineers*, vol. 43, pp. 16-21. Jun. 2006.

[22] S. Y. Ohm, K. W. Lee, and S. H. Park, "A pipelined design of the block cipher algorithm SEED," *Korean Institute of Information Scientists and Engineers*, vol. 30, pp. 149-159. Apr. 2003.

[23] J. H. Lim, J. M. Kang, S. M. Cho, H. O. Kim, and D. K. Kim, "High-speed implementation of SEED algorithm on graphics processing units using CUDA library," *Korean Institute of Information Scientists and Engineers*, vol. 37, pp. 417-421. Nov. 2010.

[24] S. K. Choi, S. Y. Kim, D. H. Shin, B. R. Lee, and Y. H. Lee, "Improvement of security cryptography algorithm in transport layer," *The Korea Contents Association*, vol. 3, pp. 107-111. May 2005.

[25] H. Feistel, "Cryptography and computer privacy," *Scientific American*, vol. 228, pp. 15-23, May 1973.

이 성 원

2015년 서경대학교 전자공학과(공학사). 2015년~현재 중앙대학교 대학원 전자전기공학과 석사 과정. 관심분야는 사물인터넷(IoT), 센서 네트워크, 임베디드, 보안 알고리즘 등.

박 승 민

2010년 중앙대학교 전자전기공학부(공학사). 2010년~현재 중앙대학교 대학원 전자전기공학과 석 · 박사 통합과정 수료, 관심분야는 기계학습, 패턴인식, 뇌-컴퓨터 인터페이스, 의도인식 등.

심 귀 보

1984년 중앙대학교 전자공학과(공학사). 1986년 중앙대학교 전자공학과(공학석사). 1990년 The University of Tokyo 전자공학과(공학박사). 1991년~현재 중앙대학교 전자전기공학부 교수. 2002년~현재 중앙대학교 중소기업산학협력센터 센터장. 2006년~2007년 한국지능시스템학회 회장. 2007년~2013년 (사)한국산학연협회 서울지역협회 회장. 2009년~2010년 중앙대학교 중앙도서관장 및 박물관장. 2011년~현재 중앙대학교 스마트지능로봇연구센터 센터장. 관심분야는 인공생명, 뇌-컴퓨터 인터페이스, 의도 인식, 감성인식, 유비쿼터스 지능형로봇, 지능시스템, 컴퓨테이셔널 인텔리전스, 지능형 홈 및 홈 네트워크, 유비쿼터스 컴퓨팅 및 센서 네트워크, 소프트 컴퓨팅(신경망, 퍼지, 진화연산), 다개체 및 자율분산로봇시스템, 인공 면역시스템, 지능형 감시 시스템, 사물인터넷(IoT), 빅데이터 등. ICROS Fellow.