# 형태보존암호화를 이용한 랜섬웨어 방지 및 스테가노그래피 보안 강화기술

Ransomware Prevention and Steganography Security Enhancement Technology Using Format Preserving Encryption


**저자**
**(Authors)**

임지환, 나관우, 우재민, 서화정
Ji-hwan Lim, Gwan-Woo Na, Jae-Min Woo, Hwa-joeng Seo



**출처**
**(Source)**

한국정보통신학회논문지 22(5), 2018.5, 805-811 (7 pages)
Journal of the Korea Institute of Information and Communication Engineering 22(5), 2018.5, 805-811 (7 pages)

**발행처**
**(Publisher)**

한국정보통신학회
The Korea Institute of Information and Communication Engineering

**URL**

http://www.dbpia.co.kr/Article/NODE07447949

**APA Style**

임지환, 나관우, 우재민, 서화정 (2018). 형태보존암호화를 이용한 랜섬웨어 방지 및 스테가노그래피 보안 강화기술. 한국정보통신학회논문지, 22(5), 805-811.

JKIICE Journal of the Korea Institute of Information and Communication Engineering

한국정보통신학회논문지 Vol. 22, No. 5: 805~811, May 2018

# 형태보존암호화를 이용한 랜섬웨어 방지 및 스테가노그래피 보안 강화기술

임지환[1] · 나관우[1] · 우재민[1] · 서화정[1*]

# Ransomware Prevention and Steganography Security Enhancement Technology Using Format Preserving Encryption

**Ji-hwan Lim[1] · Gwan-Woo Na[1] · Jae-Min Woo[1] · Hwa-joeng Seo[1*]**

[1*]Department of IT Engineering, Hansung University, Seoul 02876, Korea

## 요 약

형태 보존 암호는 암호화하고자 하는 목적 정보에 대한 변형 없이 형태를 유지한 상태로 암호화하는 기법으로써 최근에 국가보안기술연구소에 의해 제안되었다. 본 논문에서는 형태 보존 암호를 활용하여 기존의 사이버 보안 관련 문제를 해결하는 방안을 제안하고자 한다. 먼저 랜섬웨어 공격을 효과적으로 방어하기 위해 시그니처 및 확장자를 형태보존암호로 암호화하는 방안을 제시한다. 해당 기법은 최소한의 정보를 암호화함으로써 랜섬웨어에 대한 노출을 최소화할 수 있다. 두 번째로 스테가노그래피와 같이 비밀 정보를 숨기는 기술상에서도 해당 정보의 양을 최소화함으로써 공격에 대비할 수 있는 방안을 제시한다. 마지막으로 형태보존암호와 경량암호에서 암호화에 따른 동작 속도를 비교하고, 형태보존암호를 최적화하였을 때, 그에 따른 성능 향상까지 비교하고자 한다.

## ABSTRACT

Recently, Format-Preserving-Encryption (FEA) was suggested by the National Security Research institute (NSR) as an encryption method while maintaining the format without a distortion to the intended information to be encrypted. In this paper, we propose a scheme to solve conventional cyber security problems by using FEA scheme. First, we present the method to encrypt signatures and extensions with FEA in order to effectively defend against Ransomeware attacks. This technique can mitigate the exposure to the Ransomeware by encrypting the minimum information. Second, in order to reduce the secret information for Steganography, we introduce a new way to minimize the secret information with FEA. Finally, we compare the operation speed by encryption with FEA and Lightweight Encryption Algorithm (LEA), furthermore when we optimize FEA we want to compare with the performance improvement accompanying with it.

**키워드** : 랜섬웨어, 스테가노그래피, 형태보존암호, 암호화

Keywords : Ransomware, Steganography, Format-Preserving-Encryption, Encryption

**Received** 2 January 2018, **Revised** 15 January 2018, **Accepted** 15 April 2018

*** Corresponding Author** Hwa-jeong Seo(E-mail:hwajeong@hansung.ac.kr, Tel:+82-2-760-8033)
Department of IT Engineering, Hansung University, Seoul 02876, Korea

Open Access **http://dx.doi.org/10.6109/jkiice.2018.22.5.805** pISSN:2234-4772

# Ⅰ. 서 론

사이버 범죄에서 사용자의 데이터를 안전하게 보호하기 위한 많은 연구들이 진행되고 있다. 하지만, 모든 정보를 보호하고, 사전에 예방하는 것은 불가능에 가깝다. 그 이유는 단지 보안 프로그램의 유무뿐만 아니라 본인의 실수와 보안인식 결여로 인해서도 정보 노출이 발생할 수 있기 때문이다. 따라서 정보보안 기법, 기술, 그리고 프로그램 등에 의존해서 정보를 보호하기보다는 본인이 직접 자신의 정보를 보호해야 한다.

2017년 5월에 대한민국을 포함해 전 세계 150여 개국의 20만대에 달하는 컴퓨터를 감염시키면서 매우 강력한 전염성을 보여줬던 워너크라이 (WannaCry) 랜섬웨어는 사이버 공간에서 심각한 위협을 보여주었다. 모든 랜섬웨어가 인터넷에 접속만 해도 감염되는 것은 아니지만, 워너크라이는 윈도우즈 운영체제의 특성을 이용하여 전파력을 극대화시킨 예시이다. 워너크라이 뿐 아니라 웹을 통해서 범죄를 일으키는 경우 또한 한국에서 2015년 이후 급격히 증가하기 시작했다. 또 랜섬웨어 공격은 전 세계적으로 2015년 34만 건에서 2016년 46만 건으로 가파르게 증가하고 있으며 이러한 공격 기법은 특히 보안이 취약한 사이트, 가짜 이메일, 그리고 광고 등을 통해 비밀리에 들어와 사용자의 클릭으로 바이러스를 전파시키게 된다. 이렇게 수많은 사이버 범죄의 위협 속에서 기업뿐만 아니라 개개인들이 자신의 정보를 안전히 지켜내는 것은 매우 중요한 요소이다. 이와 더불어 형태보존암호화를 스테가노그래피에 적용하는 방안에 대해 확인해 보도록 한다. 스테가노그래피의 보안성을 높이기 위해 실제 이미지 파일에 삽입되는 정보 양의 변형 없이 암호화하는 방안에 대해 확인해 보도록 한다. 본 논문에서는 랜섬웨어의 위협을 형태보존암호를 사용하여 예방하는 방법과 중요 정보 암호화에 특화 되어있는 스테가노그래피 암호화 기법을 형태보존암호를 사용하여 보다 효과적으로 암호화하는 방법을 소개하도록 한다.

본 논문의 구성은 다음과 같다. 2장 본론에서 관련 연구 동향에 대해 먼저 살펴보고, 제안기법을 설명하겠다. 그리고 3장에서 제안 기법에 대한 성능 평가를 수행하고, 마지막 4장에서 본 논문의 결론을 맺도록 한다.

# Ⅱ. 본 론

## 2.1. 관련 연구 동향

본론에서 제안하고자 하는 형태보존암호를 활용한 랜섬웨어 방지 기법, 스테가노그래피 보안 기술 강화 설명에 앞서 관련 연구 동향에 대해 살펴보도록 한다.

### 2.1.1. 형태보존암호화 알고리듬

국내에서 독자적으로 개발한 Format-Preserving Encryption Algorithm (FEA) 암호기술은 국가보안기술연구소에 의해 개발되었다. 이는 NIST에서 개발해 표준으로 등록된 FPE의 Feistel 구조와 다른 트위커블 (Tweakable) 가변길이 블록암호의 특징을 갖는다 [1]. NSR에서는 국내에서 사용되는 민감한 개인정보들이 대체로 짧은 길이를 갖는다는 것에 착안하여 FEA가 기존 AES (Advanced Encryption Standard)와 달리 원본 데이터에 대해서만 암호화하기 때문에 암호화 솔루션 구축비용을 낮추고 성능은 높일 수 있는 암호화가 가능하다고 한다 [2]. 형태보존암호가 실용화된다면 암호화에 있어서 기존 데이터베이스의 스키마 변경이 필요 없다는 큰 장점을 가진다. 이와 더불어 형태보존암호에서 평문의 길이가 짧다면, 라운드 수를 증가시키는 방법을 적용하여 큰 비용을 발생시키지 않으며 메시지 복구 공격에 충분한 안전성을 확보하는 기법 등 다양한 방법에 대한 연구가 진행 중이다 [3].

### 2.1.2. 랜섬웨어 방지 기법

랜섬웨어는 데이터를 암호화하고 이를 인질로 금전을 요구하는 악성 프로그램이다. 랜섬웨어의 종류로는 크립트, 케르베르, 세이지, 록키 등의 유명한 랜섬웨어들이 있다. 주요 동작원리는 특정한 확장자 (중요 파일)를 검색한 후 이를 암호화하는 방법이다. 현재는 랜섬웨어에 감염 시 이에 대한 신속한 인지를 위하여 네트워크 트래픽 내에서 랜섬웨어 감염 시 발생하는 고유한 신호를 탐지하는 방법이 많이 활용되고 있는데, 더 나아가 랜섬웨어 탐지 패턴 자동화 모델에 대한 연구도 있다 [4]. 대응방안으로 랜섬웨어 침입이 불가능한 보안 저장 공간을 생성해 백업 솔루션 구체화에 대한 연구가 있다 [5]. 다양한 연구들이 이뤄지고 있지만, 현재까지는 악성코드에 대한 완벽한 해결방안은 없다. 따라서 많은 보안 전문가들은 이런 문제를 해결하기 위해 끊임

없이 연구해야 할 것이다. 본 논문에서는 형태보존암호를 활용하여 랜섬웨어를 예방하는 방법을 소개한다.

2.1.3. 스테가노그래피 보안강화기술

스테가노그래피는 사진 음악 동영상 등의 일반적인 파일 안에 데이터를 숨기는 기술이며 은밀한 의사소통 그리고 일급 비밀문서의 수송 등에 활용되고 있다. 하지만 다량의 데이터는 숨기기 힘들다. 스테가노그래피 기법에서 가장 기본적인 방법인 LSB (Least Significant Bit)기법은 최하위 비트에 정보를 삽입하는 방법이다 [6]. LSB에 데이터를 삽입하는 기법 이외에 다른 발전된 방식으로는 주파수 영역 (Frequency Domain)에 데이터를 은닉하는 방법이 있다. DFT, DCT, 또는 DWT 같은 알고리즘을 이용하여 커버 객체를 주파수 변환한 후, 비밀 데이터를 삽입하는 방법이다 [7]. 또한, 키 교환 시 발생할 수 있는 위협에서 비밀 정보를 안전하게 지키기 위해, QR코드 이미지를 사용하는 스테가노그래피 기법을 이용하여 키 교환 프로토콜을 제안하는 기법 등 다양한 방법이 있다 [8].

### 2.2. 랜섬웨어 예방을 위한 확장자/시그니처 암호화

랜섬웨어의 동작 원리 중 하나는 특정한 확장자를 탐색하여 이를 암호화하는 방식이다. 그 이유는 모든 파일을 암호화하게 될 경우 시스템 동작이 불가능하기 때문이다. 따라서 이를 예방하기 위해 확장자를 암호화하여 파일의 속성을 숨긴다면 악성코드의 파일 탐지로부터 1차적으로는 벗어날 수 있다. 하지만 해당 기법 역시 파일이 가지는 고유 시그니처 정보가 변경되지 않아 여전히 해킹 위험이 남아있다. 표. 1을 보면 각각의 파일은 파일마다 특정한 시그니처가 있는데, 핵심은 확장자와 시그니처를 동시에 형태보존암호를 사용하여 암호화해야 안전하다는 것이다. 확장자/시그니처 암호화에 형태보존암호를 제안한 이유는 [확장자명||시그니처]를 형태보존암호화 시키면 평문과 동일한 길이로 암호화가 가능해서 다른 암호화에 비해 파일 암·복호화, 관리 등이 수월하고 공격자가 파일의 암호화를 확인하기 어렵다는 특징이 있다. 암호화에 따른 속도 비교는 4장에서 하도록 하겠다. 본 논문의 2.2.1.과 2.2.2.에서는 제안하고자 하는 방법에 대한 2가지 시스템 동작 메커니즘을 설명하겠다.

**Table. 1** Signature information for specific file [9]

| Extension | Signature | Offset | Explanation |
|---|---|---|---|
| AVI | 52 49 46 46 | 0 | Video File |
| BMP | 42 4D | 0 | Image File |
| GIF | 47 49 46 38 | 0 | Image File |
| PDF | 25 50 44 46 | 0 | PDF File |

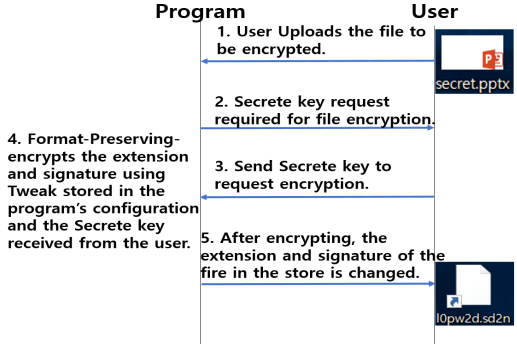


**Fig. 1** Encrypt the file to protect (version 1)

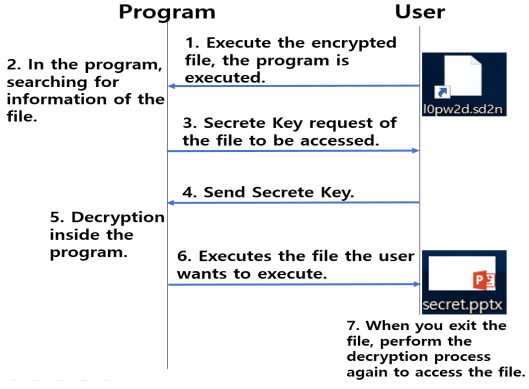


**Fig. 2** Decrypt the file to access (version 1)

2.2.1. 프로그램을 통한 시스템 동작 메커니즘

본 절에서는 앞선 2.2를 적용한 프로그램을 통한 시스템 동작 메커니즘을 제안하겠다. 우선 보호하고자 하는 파일을 프로그램을 통해 암호화하는 과정 그림. 1를 먼저 살펴보겠다. 사용자가 암호화하고자 하는 파일을 프로그램에 업로드 한다(1). 프로그램에서 해당 파일의 암호화에 필요한 Secret Key를 요구하면(2) 사용자는 Secret Key를 보내준다(3). 프로그램에서 사용자가 보내준 Secret Key와 프로그램의 설정에 저장되어 있는

Tweak을 사용하여 파일의 확장자/시그니처를 형태보존암호화 시키고(4) 암호화 요청했던 파일을 암호화된 파일로 변경한다(5). 다음으로 암호화된 파일을 복호화하는 과정 그림. 2에 대해 살펴보겠다. 사용자가 암호화했던 파일을 실행시키면 프로그램이 실행되어(1) 프로그램에서는 해당 파일에 대한 정보를 찾는다(2). 그리고 사용자가 접근하고자 하는 파일에 대해 암호화할 때 설정했던 Secret Key를 요청한다(3). 사용자가 Secret Key를 입력하면(4) 프로그램 내부에서 자체적으로 암호화했던 확장자/시그니처를 복호화 하여(5) 본래 사용자가 실행하고자 했던 파일을 실행시켜준다(6). 파일을 종료하면 다시 접근하기 위해서 위 복호화 과정을 다시 수행해야 한다(7).

2.2.2. 서버와 연결을 통한 시스템 동작 메커니즘

본 절에서 제안하는 서버와 연결하는 시스템 동작 메커니즘은 2.2.1과는 조금 다른 성격을 지닌다. 2.2.1.의 경우에는 해당 프로그램이 설치된 컴퓨터에서 만 사용이 가능하지만 2.2.2.의 경우는 공간의 제약이 없이 암호화 복호화가 가능한 메커니즘이다. 사용자가 파일을 암호화하는 그림. 3 과정에 대해 먼저 살펴보겠다. 사용자가 파일을 암호화하기 위해서 우선 서버에 먼저 ID, PW를 입력하여 로그인 한다(1). 로그인 후에 서버에 접속이 되면 사용자가 암호화하고자 하는 파일을 서버에 업로드 한다(2). 서버는 사용자의 ID, PW를 Secret Key로 하고, 프로그램이 파일의 확장자/시그니처를 암호화하도록 Secret key를 전송한다(3). 프로그램은 전송받은 Secret Key와 프로그램 내부에 저장되어있는 Tweak을 형태보존암호의 라운드 함수 과정을 거쳐 Round Key, Round Tweak을 생성한다. 단, 라운드 함수 과정은 파일을 처음 암호화하거나, 사용자 PW가 바뀐 경우에만 수행한다(4). 사용자가 업로드 한 파일을 암호화된 파일로 변경한다(5). 다음으로 암호화된 파일을 복호화 하는 그림. 4 과정에 대해 살펴보겠다. 사용자가 암호화했던 파일을 클릭하면(1) 서버는 ID, PW를 요청한다(2). 사용자가 올바른 ID, PW를 입력하면(3) 클릭한 해당 파일에 대한 정보를 서버에서 검색한다(4). 서버에서 해당 파일을 찾으면 프로그램으로 사용자의 Secret Key를 전송하고 확장자/시그니처 복호화를 요청한다(5). 프로그램에서는 자체적으로 파일을 복호화 하여(6) 서버로 전달한다(7). 사용자는 서버를 통해서 파일에 접근할 수 있다(8). 위와 마찬가지로 파일의 종료 시 복호화 과정을 다시 수행하여 파일에 접근할 수 있다(9). 그림. 2과 그림. 4 과정의 복호화 과정은 Secret Key를 전달받아 프로그램 내부에서 직접 복호화 하는 방법에 대해 설명했지만, 복호화하지 않고 파일에 접근하는 방법도 있다. 이 방법을 간단히 설명하면, 프로그램은 이미 해당 파일에 대한 정보를 갖고 있기 때문에 지정된 포맷에 따라 해당 파일에 대한 정보를 읽어 복호화를 대신하는 메커니즘이다.

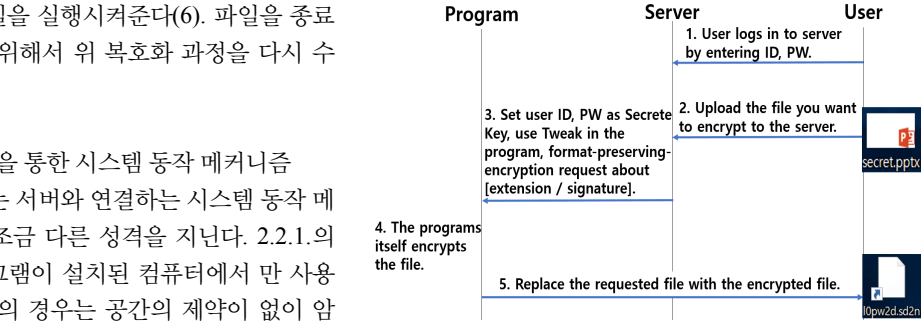


**Fig. 3** Encrypt the file to protect (version 2)

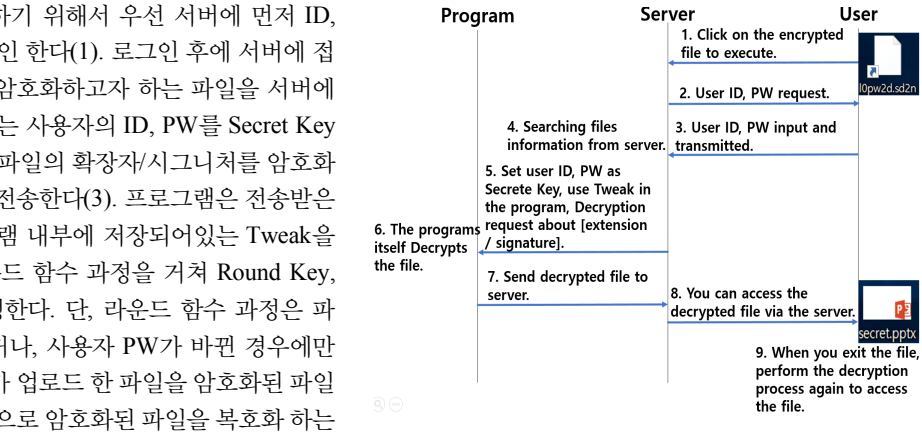


**Fig. 4** Decrypt the file to access (version 2)

### 2.3. 스테가노그래피 보안강화기술

본 절에서는 형태보존암호를 활용하여 기존 스테가노그래피 정보 삽입의 문제점을 보완할 수 있는 방법을 제안한다. 스테가노그래피 암호화 기법은 많은 정보를 숨기게 될 경우 원본에 대한 변형이 크게 일어나 해커가 이를 눈치 채고 탐지가 보다 쉽게 일어난다는 문제

점이 있다. 기존 블록 암호의 가장 큰 문제점은 암호화 후에 스테가노그래피에 들어가는 데이터 크기이다. 형태보존암호는 암호화 전후에 데이터의 크기가 변하지 않지만, 다른 블록 암호는 16byte 단위로 암호화함으로 대부분 경우에 데이터의 크기가 늘어난다. 즉, 스테가노그래피에 들어가는 정보가 많아지게 되고 정보의 변형이 더 커지게 된다. 이는 정보 변형이 민감한 스테가노그래피 보안기법에서 비효율적이다. 또한, 형태보존암호의 이점으로 정보 변형 크기가 같은 수준에서 암호화할 데이터를 더 많이 삽입할 수 있다. 성능평가의 2.25MB 비트맵 방식의 이미지에서 각 픽셀의 LSB에 데이터를 삽입하는 예시를 통해 결과를 확인할 수 있다.

### 2.4. Look Up Table을 사용한 형태보존암호 최적화

형태보존암호 TTA 표준 [1]에 기술되어있는 SBL, DL 함수는 Look Up Table을 활용하여 최적화 가능하다. DL의 M 박스의 반복되는 구조로 인하여 SBL과 DL을 사전 테이블을 활용하여 한 번에 저장할 수 있다. 이는 형태보존암호 핵심 연산인 SBL, DL 함수가 필요한 Key Scheduling 함수와 F 함수에서 사용된다. 8 bit 인풋인 X1~X8은 0~255의 값을 가짐으로 256가지의 SBL, DL 통과 후 반환 값을 미리 저장한다. X1~X8이 LUT를 통과한 후에 순차적으로 Y1부터 Y8까지 로테이션 과정을 거치고 XOR 연산을 하면 최종 결과는 최적화 전과 같고, 연산 양은 크게 줄어든다. 표. 2는 이를 C 언어 코드로 나타낸 모습이다.

**Table. 2** Optimization code for SBL and DL using Look Up Table (2KB)

```
void OPT_SBL_DL(u8* in1){
        int i=0; u64 Sum=0;
        Sum = LUT[in1[0]] ^
        ((LUT[in1[1]] << 8) ^ (LUT[in1[1]] >> 56) ) ^
        ((LUT[in1[2]] << 16) ^ (LUT[in1[2]] >> 48) ) ^
        ((LUT[in1[3]] << 24) ^ (LUT[in1[3]] >> 40) ) ^
        ((LUT[in1[4]] << 32) ^ (LUT[in1[4]] >> 32) ) ^
        ((LUT[in1[5]] << 40) ^ (LUT[in1[5]] >> 24) ) ^
        ((LUT[in1[6]] << 48) ^ (LUT[in1[6]] >> 16) ) ^
        ((LUT[in1[7]] << 56) ^ (LUT[in1[7]] >> 8) );
            for(i=0; i<8; i++) { in1[i] = 0;}
        NumToBits(Sum, in1); }
```

## Ⅲ. 성능 평가

본 장에서는 128bit의 키 길이를 갖는 LEA와 128bit의 키 길이를 갖고, 제2형 TBC를 사용하는 FEA를 비교하여 파일 암호화와 스테가노그래피 암호화에 효율성을 비교하고자 한다. 비교에 앞서 테스트 환경은 표 3과 같다.

**Table. 3** Test Environment (PC, Program)

| | |
|---|---|
| Processor | Intel Core i5-6200U CPU 2.3 GHz |
| Memory | 8GB (DDR3L) |
| Environment | Visual Studio C++ 2010 Express x86 tools |
| Optimization | O2 |
| FEA | Functions for TBC operation type 2 |
| | Implementation based on TTA-Standard and Optimization SBL, DL (C Language) |
| LEA | 128bit Block Cipher LEA |
| | Using standard reference (C Language) |

LEA와 FEA로 8MB의 GIF 파일을 암호화했을 때 데이터베이스 포맷을 유지한다는 전제 하에 FEA는 [확장자||시그니처]의 7byte만 암호화 하면 되지만, LEA는 [확장자||시그니처]가 15byte 이하라면 데이터베이스 포맷을 유지하기 위해 8MB를 전부 암호화해야 한다. 그에 따른 성능 테스트는 표. 4와 같다. 정리하면 LEA는 8MB 파일을 암·복호화 하는데 13.143sec 시간이 소요되고, FEA는 7B를 암·복호화 하는데 $2.625*10^{-5}$sec 시간이 소요됨으로, FEA로 암·복호화 했을 때 500,000 배 이상의 속도가 나온다. 랜섬웨어 대상인 파일 대부분의 [확장자||시그니처]가 15byte 이하인 점을 고려하면 이는 매우 안전하고 효율적이다.

**Table. 4** Comparison of LEA (8MB encrypt and decrypt) and FEA (7B encrypt and decrypt)

| Method | LEA | FEA |
|---|---|---|
| Timing | 13.143s | $2.625\times10^{-5}$s |

**Table. 5** Comparison of LEA and FEA 2.25MB bitmap image file encryption and decryption

| Method | LEA | FEA |
|---|---|---|
| Timing | 0.18s | 0.4s |

표. 5는 2.25MB (1024x768) 비트맵 방식의 이미지 파일에 모든 픽셀의 24bit LSB에 데이터를 삽입하고 암·복호화 했을 때 FEA는 0.4sec, LEA는 0.18sec 정도의 시간이 소요된다. 15byte 이하의 데이터를 삽입할 때, LEA는 데이터 포맷의 변형으로 안전성 문제가 일어날 수 있다.

표. 6은 구현한 기본 FEA 레퍼런스와 Look Up Table을 사용한 FEA(LUT)와의 연산 속도를 비교한 것이다. FEA는 F 함수에서 주요 연산이 이루어지는데, F 함수의 SBL과 DL을 Look Up Table을 사용하면 6배 속도가 향상되는 모습을 보여준다. 표 7은 FEA(LUT)와 LEA의 테스트 케이스 10만 개에 대한 16byte 암·복호화 속도를 비교한 표이다. LEA와 비교를 위해 FEA는 8byte*2 암호화를 수행하였다. 결과적으로 LEA보다 2.6배 속도가 향상된 모습을 볼 수 있다.

**Table. 6** FEA performance improvement in 8byte encryption and decryption using Look Up Table in 100,000 Test Cases

| Method | FEA | FEA(LUT) |
|---|---|---|
| Timing | 3s | 0.5s |

**Table. 7** Comparison of LEA and FEA(LUT) for 16b encryption and decryption in 100,000 Test Cases

| Method | LEA | FEA(LUT) |
|---|---|---|
| Timing | 2.6s | 1s |

## Ⅳ. 결 론

본 논문에서는 형태보존암호를 활용하여 랜섬웨어 예방, 스테가노그래피 보안기술 향상 기법을 제안하였다. 제안한 방법은 암·복호화 후 데이터의 포맷이 변하지 않는다는 장점으로 기존 블록 암호 대비 사용 편리성, 속도, 안전성 측면에서 장점을 가진다. 현재 형태보존암호화에 대해 여러 가지 연구가 이루어지고 있는데, 앞으로 기존 데이터베이스 시스템에 맞게 활용될 것으로 기대된다. 본 논문에서 테스트한 LEA는 한국암호포럼에 기재되어있는 표준 레퍼런스 코드를 사용하였고, FEA는 C 언어로 직접 구현하여 테스트하였다. 아직 FEA를 어셈블리까지 최적화하진 못하여 Clock Cycle이 다소 느릴 수 있지만, 이를 감안하더라도 LUT를 사용함으로 LEA보다 더 빠른 연산을 수행할 수 있음을 보였다. 향후 FEA에 대해 어셈블리 단계에서 최적화하여 성능을 개선하고자 한다. 최종적으로 본 논문에서 제안한 기법에 대한 추가적인 수정 및 보완을 거친 후에 서버를 구축하고 FEA를 활용하여 암호화하는 테스트를 수행하고자 한다.


## ACKNOWLEDGEMENT

This research was financially supported by Hansung University. This research of Jihwan Lim was supported by the MSIT(Ministry of Science and ICT), Korea, under the ITRC(Information Technology Research Center) support program (2014-1-00743) supervised by the IITP(Institute for Information & communications Technology Promotion)


## REFERENCES


[1] Telecommunications Technology Association. TTAK.KO-12.0275. Format-Preserving Encryption Algorithm FEA [Internet]. Available: https://tta.or.kr/include/Download.jsp?filename=choan%2F%5B2015-203%5D_%C7%FC%C5%C2+%BA%B8%C1%B8+%BE%CF%C8%A3+FEA.hwp.

[2] Digital Dailey. New encryption technology came out for personal information protection [Internet]. Available: http://www.ddaily.co.kr/news/article.html?no=119354.

[3] S. Y. Jeong, D. W. Hong, and C. H. Seo, “Secure Format-Preserving Encryption for Message Recovery Attack,” *Journal of Korean Institute of Information Scientists and Engineers*, vol. 44, no. 8, pp. 860-869, Aug. 2017.

[4] H. K. Lee, J. H. Seong, Y. C. Kim, J. B. Kim, and G. Y. Gim, “The Automation Model of Ransomware Analysis and Detection Pattern,” *Journal of the Korea Institute of Information and Communication Engineering*, vol. 21, no. 8, pp. 1581-1588, Aug. 2017.

[5] Y. K. Kim, D. G. Ham, Y. H. Joo, and K. H. Lee, “Analysis and Countermeasures for the Ransomware Cryptolocker,” in *Proceeding of the 2016 Spring Conference of the Korea Information Processing Society*, Seoul, vol. 23, no. 1, pp. 293~293, Apr. 2016.

[6] D. K. Andrew, "Steganalysis of Embedding in Two

Least-Significant Bits," *IEEE Transactions on Information Forensics and Security*. vol. 2, no. 1, pp. 46-54, Feb. 2007.

[7] B. K. Moon, D. G. Ryoo, M. S. Ko, K. W. Eom, and M. S. Jun, "An Implementation of Database Security Using Steganography in the Web," *Journal of The Korea Institute of Information Security and Cryptology*, vol. 15, no. 2, pp. 3-11, Apr. 2005.

[8] G. J. Lee, E. J. Yoon, and K. Y. Yoo, "A Key Exchange Protocol based on the Steganography with the QR code," *Journal of the Institute of Electronics Engineers of Korea*, vol. 50, no. 6, pp. 173-179, Jun. 2013.

[9] File Signature Database. File Signature [Internet]. Available: https://filesignatures.net/.

**임지환(Ji-Hwan Lim)**

2014년 3월~현재: 한성대학교 IT 융합공학부 학부과정(3학년)
※관심분야 : 사이버보안, 암호화 구현, IoT, 블록체인

**나관우(Gwan-Woo Na)**

2017년 3월~현재: 한성대학교 IT 융합공학부 학부과정(2학년)
※관심분야 : 암호화 구현, 네트워크 보안

**우재민(Jae-Min Woo)**

2014년 3월~현재: 한성대학교 IT 융합공학부 학부과정(3학년)
※관심분야 : 모의해킹, 데이터 통신 보안

**서화정(Hwa-jeong Seo)**

2010년 2월: 부산대학교 컴퓨터공학과 학사 졸업
2012년 2월: 부산대학교 컴퓨터공학과 석사 졸업
2012년 3월~2016년 1월: 부산대학교 컴퓨터공학과 박사 졸업
2016년 1월~2017년 3월: 싱가포르 과학기술청
2017년 4월~현재: 한성대학교 IT 융합공학부 조교수
※관심분야 : 정보보호, 암호화 구현, IoT