2023 암호분석경진대회

5번 문제

1. 서론

5번 문제는 문제 설명과 4장의 사진들을 이용해 주어진 데이터들에 은닉된 데이터들을 찾아내는 문제이다.

2. 이론 배경

이미지 파일 구조

해당 문제에는 png, bmp등의 이미지 파일이 등장한다. 이러한 이미 파일은 각자 독특한 구조를 가지고 있다. 문제의 [2.bmp]는 JPEG의 파일 구조를 설명하고 있다. 이미지 파일은 실제 픽셀에 관한 정보를 담은 데이터 부분과, 이미지 파일의 종류 및 픽셀 표현 방식 들을 포함 header 등 여러 구획으로 구성되어 있다. 이미지의 각 구획들은 이미지 파일의 시작을 가리키는 SOI(FF D8), 이미지의 끝을 가리키는 EOI(FF D9), 실제 이미지 데이터의 시작인 SOS(FF DA)등의 시그니처로 구분된다.

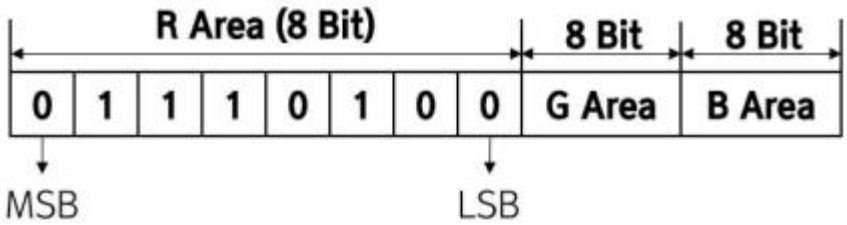


그림 1 24비트 픽셀 구조

이미지 파일의 실제 데이터 부분은 픽셀표현의 연속으로 구성되어 있다. PNG 파일의 경우, RGB값에 각 8비트씩 할당되어 총 24비트로 1픽셀을 표현한다. (투명도를 조절하는 alpha 값이 추가될 수도 있다) 색 하나의 8비트 중 가장 상위 비트를 MSB, Most Significant Bit라 부르고, 가장 하위 비트를 LSB, Least Significant Bit라고 부른다.

스테가노그래피 - Bit 변조

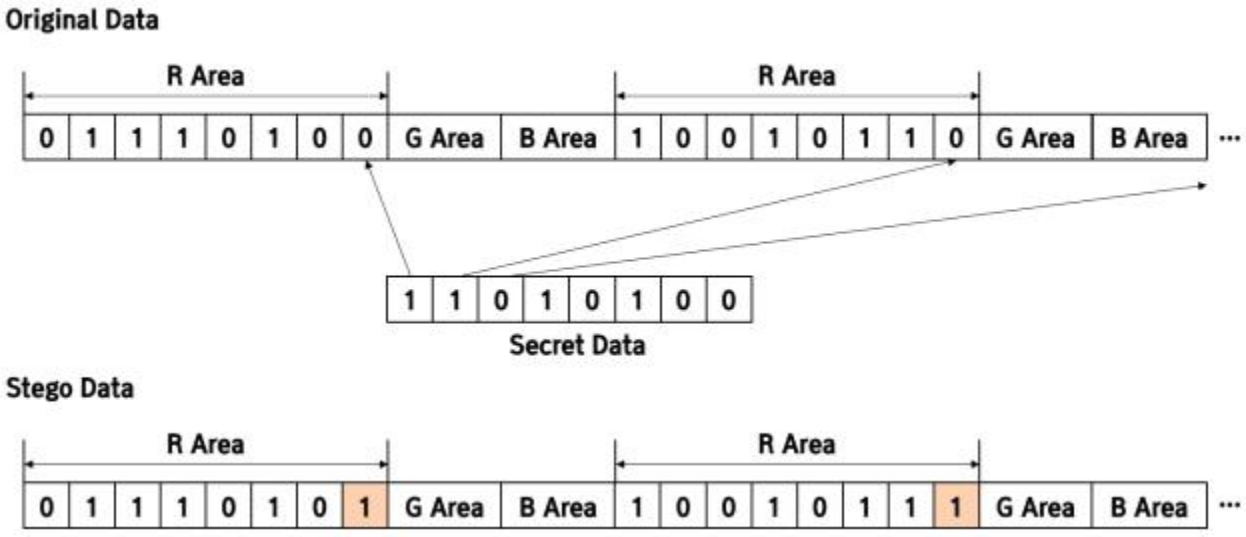


그림 2 Bit 변조 데이터 은닉 방법

이미지에 정보를 은닉하는 간단한 방법으로 LSB가 존재한다. Jpg, png 파일들은 8비트의 RGBA값을 이용해 픽셀 하나를 표기한다. 이 8비트는 상위 비트가 변하면 전체값이 크게 변하지만, 하위 비트가 변할 경우 전체값에 비해 변화량이 미미해 그 차이를 알아차리기 쉽지 않다. LSB 방식은 이 8비트 중 최하위 비트(LSB, Least Significant Bit)를 우리가 원하는 데이터로 변조하여 이미지에 정보를 숨기는 방식이다. 이 방식은 육안으로 보

았을 때 데이터가 숨겨져 있다는 것을 파악하기 어렵다. 해당 문제에는 LSB뿐 아니라 MSB (최상위 비트)를 변조해 데이터를 은닉하는 방식도 포함되어 있다. 이 경우 육안으로 이미지의 변화가 확연히 식별 가능하다.

그림 3 LSB와 MSB의 변화

위와 같이 LSB가 변화할 경우, 색 차이를 알아볼 수 없지만 MSB가 변화할 경우 확연한 색의 차이를 느낄 수 있다.

스테가노그래피 - 파일 내부 삽입

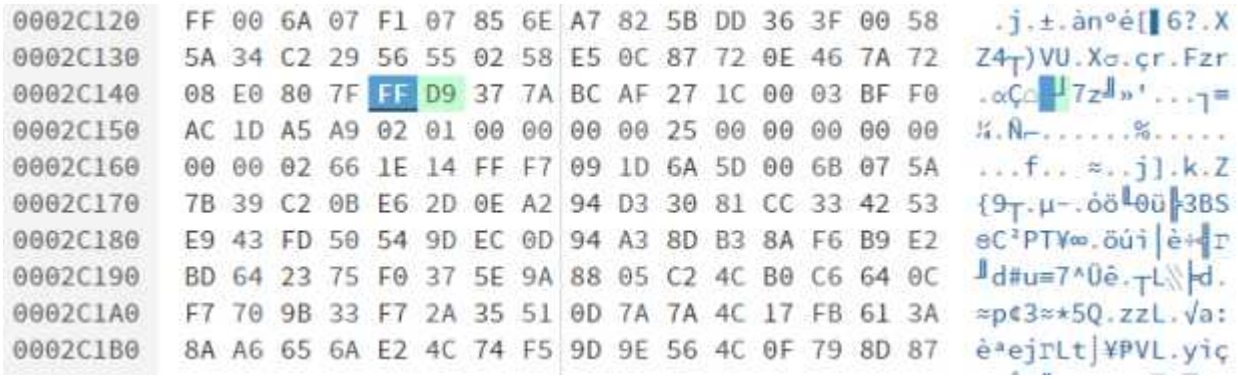


그림 4 PNG의 푸터 시그니처 뒤에 데이터가 삽입된 모습

Bit 변조 외에 다른 방법으로는 파일 내부에 직접 데이터를 삽입하는 방법이 있다. 이미지 파일의 구조에 따라 데이터 중 읽지 않는 부분이 생긴다. 가령, png 파일의 경우 EOI 시그니처인 FF D9가 등장한 이후의 데이터는 이미지 표현에 사용되지 않는다. 마찬가지로 예약된 구획인 APPn부터 SOS까지의 헤더 영역도 데이터가 바뀌어도 이미지 표현에 지장이 가지 않는다. 이런 구획에 우리가 숨기고 싶은 데이터를 은닉할 수 있다.

AES 암호화(CTR Mode)

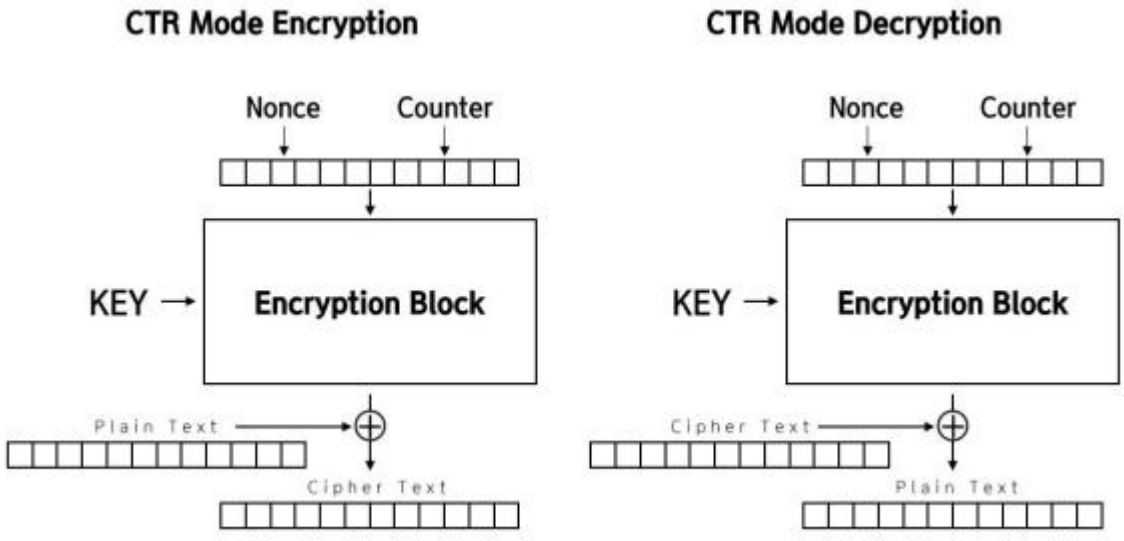


그림 5 AES CTR Mode의 암호화 복호화 방식

AES는 블럭 암호화 알고리즘의 한 종류이다. AES는 암호 블럭(Key Stream)의 생성 방식에 따라 여러가지 Mode로 나뉘는데, 해당 문제에서는 CTR Mode의 암호화 방식이 등장한다. CTR Mode는 plain text를 바로 암호화하지 않고, 사전에 지정된 Nonce 값과 암호화 할 때 마다 1씩 증가하는 Counter을 이용해 암호 블럭을 생성한뒤, 이를 plain text와 xor 연산하여 cipher text를 만들어낸다. 암호화와 복호화가 완전히 같은 구조로, 복호화할 때는 cipher text를 똑같은 암호 블럭과 xor 연산하면 plain text를 얻을 수 있다.

Ascon's S-box

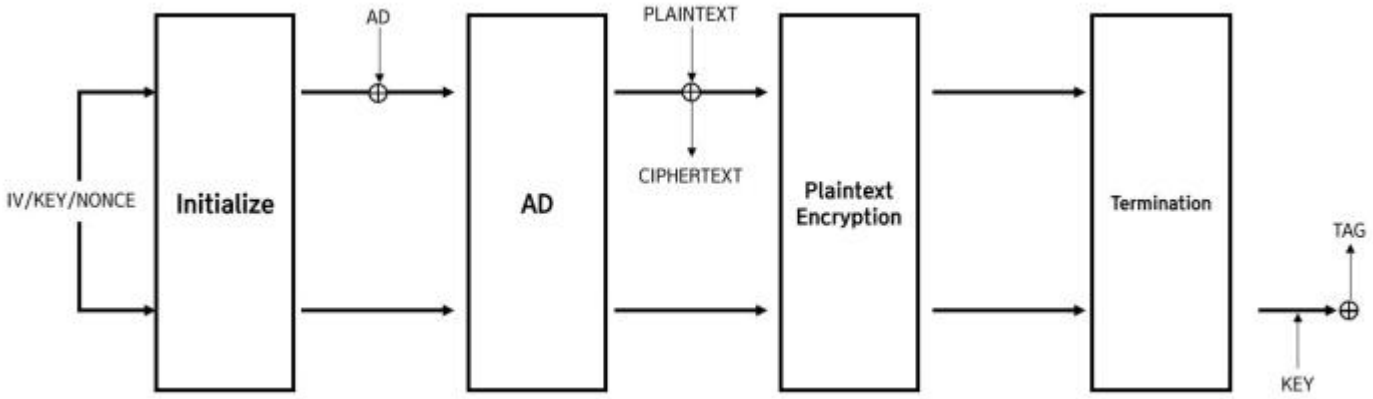


그림 6 Ascon's S-Box의 개략적인 암호화 과정

초기에 주어지는 [1.png] 파일에는 Ascon's S-Box라는 연산이 기입되어 있다. Ascon's S-Box는 암호화 방식의 한 종류로, 암호화에 key, nonce, tag(AD, Associated Data)가 사용되며, 암호화 과정은 [1.png]의 안내를 따르면 된다. 이 암호화 과정은 duplex sponge 구조를 가지게 되는데, 도식화하면 위의 그림과 같아진다. 암호화는 크게 4가지 과정 초기화, AD 가공, plaintext 암호화, 종결로 이루어지는데, 각 과정은 연산이 여러 번 반복해서 이루어질 수도(multi-round) 있다. 평문을 모두 암호화 한 후에는 복호화 과정에서 확인하는 용도로 쓰이는 태그를 도출해낸다.

Ascon's S-Box는 128비트에서 Ascon-128, Ascon-128a 두 가지 암호화 모드를 가진다. 각 암호화 모드는 AD 가공과 plaintext 암호화의 반복 횟수를 결정한다. (128의 경우 6, 128a의 경우 8)

3. 풀이 방법

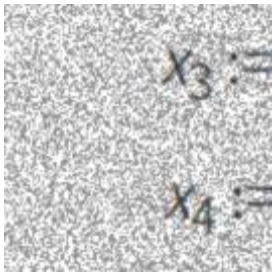
은닉되지 않은 데이터 분석

숨겨진 데이터를 찾아내기 이전, 최초로 주어진 4장의 사진에 표면적으로 기입된 데이터를 분석한다. [1.png]에는 Ascon's S-box라는 해싱 함수의 매커니즘이 설명되어 있다. [2.bmp]는 이미지 파일의 구조를 설명하고 있다. 파일 구조내에 EOI, SOI와 같은 고유 마커들이 존재하고, 해당 마커들은 HEX파일에서 FFD9과 같은 고유의 순열로 표기되어 있다는 사실을 알려준다. [3.bmp]는 대표적인 스테가노그래피 방식을 표현하고 있다. 이미지 파일의 R부분에 한개의 비트씩 데이터를 은닉하는 LSB라는 기법을 기술하고 있다. [4.png]에는 제주공항의 E와 J 사이에 핀 포인트된 항공사진이 주어져 있다.

첫번째 은닉 데이터

[3.bmp]와 문제 설명에 언급된 R영역 LSB를 4개 사진에 대해 모두 조사해 보았다. [1.png]에서 노란 메모지에 [PASSWORD IS #2023$CRYPT@NALY$IS2023#]이라 적힌 사진 파일을 찾아낼 수 있었다.

두번째 은닉 데이터



문제 설명에 언급된 MSB가 변경된 경우가 있는지 4가지 사진에 대하여 조사해 보았다. [1.png]의 사진이 육안으로 보기에도 비정상적인 회색 글리치가 나타난다는 사실을 확인하였다. 따라서, 해당 사진의 alpha값의 MSB만을 추출해 7z파일을 습득하였다. 해당 압축파일은 암호로 잠겨 있었는데, 첫번째 은닉 데이터에서 얻은 암호를 사용하여 압축 해제할 수 있었다. 해당 압축 파일에는 [partition.vhd]라는 가상 디스크 파일과 [P.txt]라는 txt 파일이 존재했다.

세번째 은닉 데이터

주어진 사진들을 HEX파일로 열어 조사해 보는 과정에서 [2.bmp] 파일이 화질에 비해 비정상적으로 용량이 크고, 가장 첫 바이트들이 BMP의 파일 시그니처인 42 4D로 시작하지 않고 jpg의 파일 시그니처(SOI)인 FF D8 FF E0으로 시작한다는 사실을 알아내었다. 따라서, 해당 파일에 은닉된 데이터가 있다고 판단, 조사하는 과정에서 jpg 파일의 EOI인 FF D9 이후에 7z 파일의 파일 시그니처인 37 7A BC AF 27 1C가 등장한다는 것을 확인하였다. 따라서 [2.bmp]의 FF D9 이후 부분만을 추출해내어 7z 압축 파일을 습득하였다.

해당 압축 파일에서 [enc_3.bmp], [enc_4.png], [enc_5.png], [README_README.txt] 총 4가지의 파일을 얻어내었다. 세 개의 이미지 파일은 읽어낼 수 없는 상태였으며, txt 파일에는 파일들이 AES-256 bit; CTR Mode로 암호화 되어 있다고 적혀있었다. [enc_3.bmp], [enc_4.png]가 초기에 주어진 [3.bmp], [4.png]와 파일 크기와 확장자가 동일하여 해당 파일의 암호화되기 전 원본이 동일 하다고 판단하였다.

AES-256 bit의 CTR Mode는 알 수 없는 Encryption Block과 plain text를 xor 연산하여 cipher text를 도출한다. 따라서, 암호화에 사용된 key와 nonce 등을 모르더라도 plain text와 cipher text를 완벽하게 보유하고 있다면 이 둘을 xor 연산하여 Encryption Block을 얻어낼 수 있다. 우리는 [enc_4.png]의 plain text와 cipher text를 모두 보유하고 있으므로, 여기서 Encryption Block을 얻어 [enc_5.png]를 decrypt할 수 있다. (enc_3의 경우 enc_5보다 파일 크기가 더 작아 완벽하게 복호화 하는데 부족하다) 위 방법을 통해 [enc_5.png]를 복호화 하였고, Python코드를 얻을 수 있었다.

```
# pip install scrypt
import scrypt
salt = b"contest2023"
data = scrypt.hash(password=P,salt=salt,N=65536,r=8,p=1,buflen=48)
KEY = data[:16]
NONCE = data[16:32]
AD = data[32:]
```

그림 9 enc_5.png

얻어낸 코드는 몇가지 값을 입력해 data라는 배열을 얻고, 거기서 KEY, NONCE, AD에 해당하는 세가지 정보를 정리하는 코드이다. Scrypt 복호화에 필요한 데이터 중 [P]를 제외하면 모두 주어진 상태였다.

그림 10 P.txt의 좌표들

두번째 은닉 데이터의 [P.txt]는 좌표로 추정되는 숫자 4개가 적혀있는 파일이었다. 해당하는 4개의 좌표는 각각 인천국제공항, 제주국제공항, 간사이 국제공항, JFK국제공항의 좌표였다. 각 좌표들이 ||로 이어져 있는 것을 보아, 각 좌표들이 pinpoint한 메시지를 일렬로 이어붙인 것이 위쪽 python 코드의 P에 해당되는 password일 것이라 추측했다. 따라서 INCHEONJEJUKANSAI31L-13R이라는 P 값을 얻어냈고, 해당 값으로 코드를 실행시켰다.

KEY: AF 5E 3E 44 AE 74 54 D1 C5 BC 8F 5F 6F AF 76 C6
NONCE: B4 D3 FF 12 57 AF 53 55 9A EF ED 1A 3F 1B AE 3B
AD: 90 18 97 06 DC EC D6 60 89 83 F3 3B 29 37 A7 1F

이를 통해 위와 같은 hex KEY, NONCE, AD 값을 얻어내었다.

네번째 은닉 데이터

[partition.vhd]의 파일이 읽어낼 수 없는 상태였다. 해당 파일의 0x410C00주소부터 알 수 없는 데이터 더미가 있는 것을 확인하였다. 따라서 해당 데이터 더미 부분만을 추출하여 파일로 변환해두었다. (이 파일을 [enc_blueprint]라고 칭한다) [enc_blueprint]를 HEX파일로 열어본 결과 데이터가 읽어낼 수 없는 형태였다. 또한, 파일이 어떻게 암호화되어있는지 별다른 언급이 없었기에, 아직 사용하지 않은 Ascon으로 암호화되어있다고 판단하였다.

어떤 모드로 암호화 되어있는지 알 수 없었기에, Ascon-128과 Ascon-128a를 모두 실행해 보았고, Ascon-128로 복호화 한 결과, 파일의 헤더가 25 50 44 46으로 시작하는 PDF파일이라는 것을 알 수 있었다. 따라서 [enc_blueprint]를 Ascon-128로 복호화 하였고, pdf 파일을 얻을 수 있었다.

4. 결론

그림 11 복호화 한 PDF 파일

왼쪽과 같은 도면이 그려진 PDF 파일을 얻어내었다. 따라서 4가지 은닉된 데이터와, 최종 도면까지 모두 찾아내는데 성공하였다.