2020 암호분석경진대회 답안제출

2020. 08. 31

참가자 1	**성 명**	이창원
	소속	서울시립대학교 수학과
	휴대폰	
	E-mail	

참가자 2	**성 명**	장호빈
	소속	서울시립대학교 수학과
	휴대폰	
	E-mail	

참가자 3	**성 명**	김인성
	소속	서울시립대학교 수학과
	휴대폰	
	E-mail	

4번 문제 답안

① 파일 복호화 단계 정답 :

정보보호개론 중간고사

1. ECDSA 파라미터 secp256r1으로 두 개의 메시지(M_1, M_2)에 대응하는 ECDSA (with sha256) 서명쌍 두 개(S_1, S_2)를 다음과 같이 생성하였다.

【 secp256r1 파라미터 】

p	0xFFFFFFFF 00000001 00000000 00000000 00000000 FFFFFFFF FFFFFFFF FFFFFFFF
a	-3
b	0x5AC635D8 AA3A93E7 B3EBBD55 769886BC 651D06B0 CC53B0F6 3BCE3C3E 27D2604B
G_x	0x6B17D1F2 E12C4247 F8BCE6E5 63A440F2 77037D81 2DEB33A0 F4A13945 D898C296
G_y	0x4FE342E2 FE1A7F9B 8EE7EB4A 7C0F9E16 2BCE3357 6B315ECE CBB64068 37BF51F5
n	0xFFFFFFFF 00000000 FFFFFFFF FFFFFFFF BCE6FAAD A7179E84 F3B9CAC2 FC632551

* 파라미터 정보 : 유한체 F_p 상에서 정의된 타원곡선 $y^2 = x^3 + ax + b$의 생성원 $G(G_x, G_y)$와 생성된 그룹의 차수 n

M_1	cryptography
M_2	security
SHA256(M_1)	E06554818E902B4BA339F066967C0000DA3FCDA4FD7EB4EF89C124FA78BDA419
SHA256(M_2)	5D2D3CEB7ABE552344276D47D36A8175B7AEB250A9BF0BF00E850CD23ECF2E43
S_1 (r_1, s_1)	r_1 : F42A3ADB78BF22D9AA571FDFB0C93B415C8B50719C25B23F6F77DC299C01F2D7 s_1 : 90A7F16EAFE3DB7923A07A6E8CF68F688100ABE3730A5416B37FA4BFBA7F5E22
S_2 (r_2, s_2)	r_2 : F42A3ADB78BF22D9AA571FDFB0C93B415C8B50719C25B23F6F77DC299C01F2D7 s_2 : 865BF221DD856B991EB6EF7EF1E0D263215FC8D7F2283A22C1F927ABDCC35B5B

ECDSA 서명 검증에 사용되는 공개키 ($Q(Q_x, Q_y) = dG$, d: 개인(서명)키)가 다음과 같을 때, 개인(서명)키 d를 찾으시오.

Q_x	F44CD6277CED3F9CC2F29144CDBCFDC40F1BF556707ED8190E838D711A12EC03
Q_y	68B6FAF59BEC47A11D98800B4BE578CA4399B34EF94D6B602F5186D41B7430C9

② 복호화된 파일 내의 시험문제인 공개키 암호(전자서명) 해독 단계 정답 :

0xbea4fd03c804ea0160a096f4c6b438d54ab78458e124e8f68d42cd7010807a1b

4번 문제 답안

① 파일 복호화 단계 풀이 :

AES는 이론상 안전한 암호화 알고리즘으로 알려져 있다. 하지만 암호화 과정에서 발생하는 물리적인 부가적인 정보 (전자파, 소비 전력, 알고리즘 수행 시간) 들의 노출을 이용하여 비밀 정보를 알아내는 방법이 부채널 분석이다.

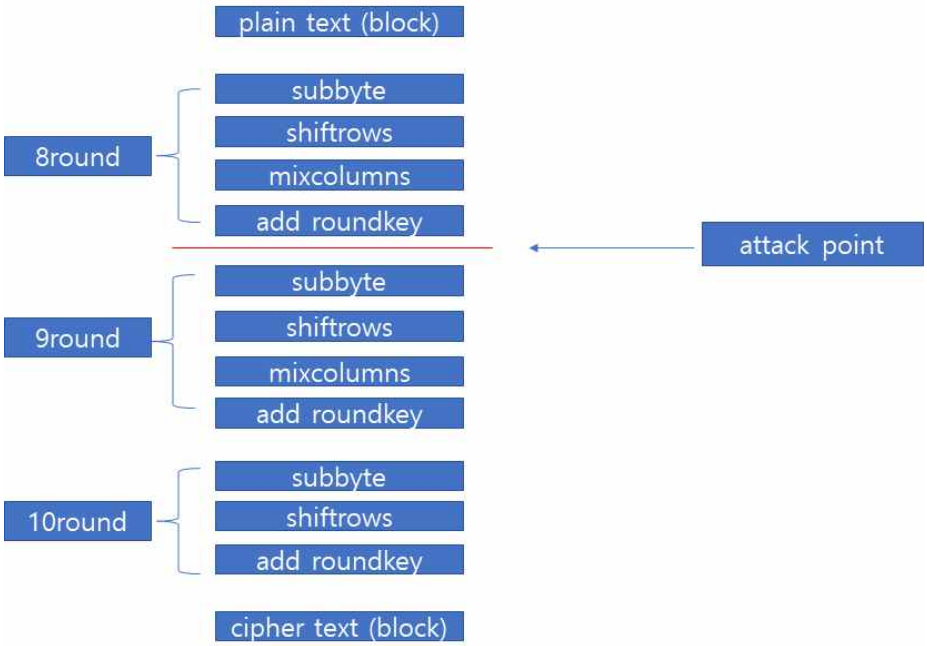
부채널 분석 중 전력분석이란, 암호화 연산 도중 발생한 소비 전력을 이용하여 암호키를 찾는 부채널 분석 기법으로 효율적인 방법으로 알려져 있다. 그 중 상관전력분석 (CPA, Correlation Power Analysis)은 전력소비량과 암호화 알고리즘 중간 계산 결과의 상관도를 이용하는 것이다. CPA에 이용되는 전력소비 모델은 해밍웨이트 모델 (Hamming Weight Model)을 활용하고, 상관관계는 피어슨 상관계수 (Pearson Correlation)을 활용한다. 암호화 알고리즘 중간 계산 결과와 전력소비량의 상관도가 높을 경우 상관계수의 크기가 높게 나타나고 이는 추측한 키와 실제 알고리즘에 쓰인 키가 일치함을 의미한다. 상관계수 식은 다음과 같다.

$$\hat{\rho}_{W,H}(R) = \frac{N\sum W_i H_{i,R} - \sum W_i \sum H_{i,R}}{\sqrt{N\sum W_i^2 - (\sum W_i)^2}\ \sqrt{N\sum H_{i,R}^2 - (\sum H_{i,R})^2}}$$

N: 평문 개수 (블럭), W_i : i번째 전력 소비량, $H_{i,R}$: $M_i \oplus R$의 Hamming Weight,
M_i : random data words, R : reference state, $\hat{\rho}_{W,H}(R)$: W,H와 R에 대한 상관계수

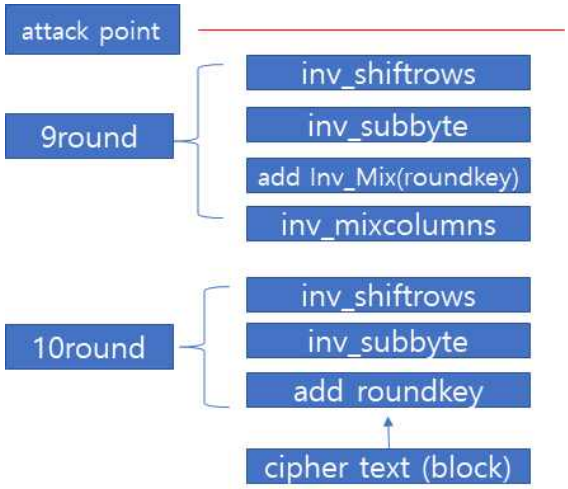
주어진 exam.hwp.encrypted 파일은 exam.hwp 파일을 AES128/CBC/PKCS#7으로 암호화를 진행한 것이고 PowerConsumption.csv는 AES128 10라운드 중 7,8라운드 연산에 해당하는 전력소비량이다. 전력소비량의 파형은 각 20,000 포인트로 구성되어 있으며 exam.hwp.encrypted 파일은 1025*16 bytes 이다. 또한 마지막 라운드의 라운드 키가 $\begin{pmatrix} 22 & E9 & * & * \\ E5 & * & * & 9B \\ * & * & 5B & * \\ * & 4C & 2C & * \end{pmatrix}$으로 주어져있다.

exam.hwp.encrypted 파일 크기가 1025*16 bytes 이므로 암호문 블록이 총 1025개이다. 또한 7,8 라운드 연산의 전력소비량이 주어져있으므로 attack point를 8라운드와 9라운드 사이로 잡자.



4번 문제 답안

exam.hwp.encrypted 파일은 unicode로 구성되어 있고 16bytes 씩 한 블록임을 이용하여 1025개의 블록에 PowerConsumption.csv를 활용하여 CPA를 진행하자. AES128 복호화 과정 각 라운드 내에서 라운드 키에 inverse mixcolumns를 취하면 inverse mixcolumns 와 add roundkey의 순서를 바꿀 수 있다. 이는 행렬 곱 성질에 의해 Inv_Mix(C⊕K)=Inv_Mix(C)⊕Inv_Mix(K) 임을 이용하였다.
(Inv_Mix : inverse mixcolumns, C : cipher block, K : roundkey)
또한 inverse subbyte와 inverse shiftrows는 블록 내의 각 바이트 별로 진행하는 것이기에 순서를 변경하여도 무관하다. 이를 이용하여 복호화 과정을 나타내면 다음과 같다. (inv_ : inverse)



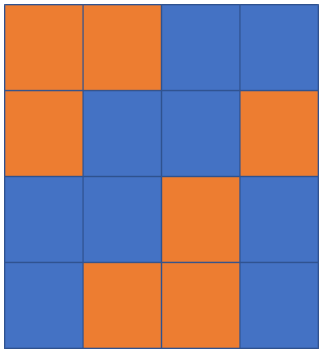
그리고 주어진 10라운드의 라운드키 (마지막 라운드의 라운드키)를 1바이트 기준으로 hex값 블록으로 표시하면 다음과 같다.

22	E9		
E5			9B
		5B	
	4C	2C	

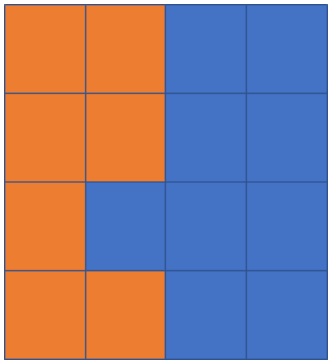
암호문 1025개의 각 블록에 대해 위 과정의 attack point까지 도달하기 위해 복호화 과정을 진행한다. 복호화 과정 중 암호문 블록의 바이트 값을 알 수 있는 부분을 주황색, 알 수 없는 부분을 파란색으로 표시하기로 하자.

(1) 10라운드의 addroundkey 단계 후
(2) 10라운드의 inverse subbyte, inverse shiftrows 단계 후
(3) 9라운드의 inverse mixcolumns 단계 후

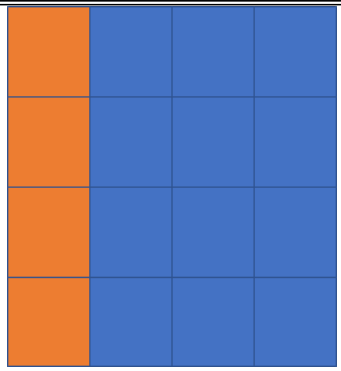
4번 문제 답안



(1)



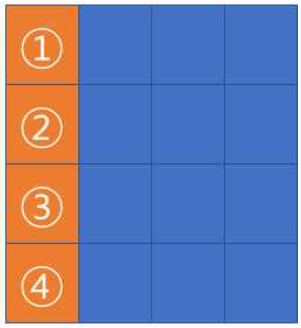
(2)



(3)

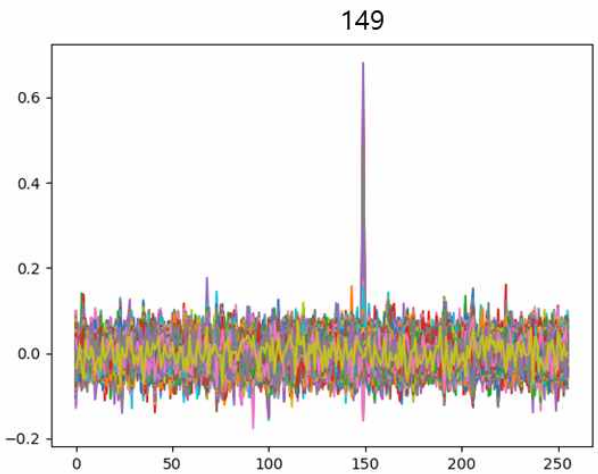
이제 9라운드의 라운드 키에 inverse mixcolumns를 취한 값으로 add round 단계를 계산하여야 하지만 9라운드의 라운드 키를 모르므로 주어진 전력소비량과 각 블록의 바이트 별로 0x00~0xFF을 더한 값으로 CPA를 이용하여 Inv_Mix(K) 값을 구한다. (K: 9라운드의 라운드 키) (전수조사량 : 4×2^8)

주황색으로 표기된 부분이 암호문 블록 복호화 과정 중 현재 알고 있는 값이므로 9라운드 키의 1~4번째 바이트에 해당하는 값부터 CPA를 진행하자.



Inv_Mix(9round key)

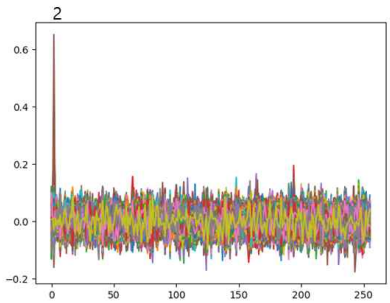
①에 해당하는 Inv_Mix(K) 값을 구하기 위해 1025개의 블록에 대해 ①위치의 값(바이트)들에 한해 0x00~0xFF 값을 Xor 한 후 inverse subbyte, inverse shiftrows를 진행한다. 그리하여 나온 256가지의 1025개의 값과 PowerConsumption.csv를 이용하여 CPA를 진행한 그래프는 다음과 같다.

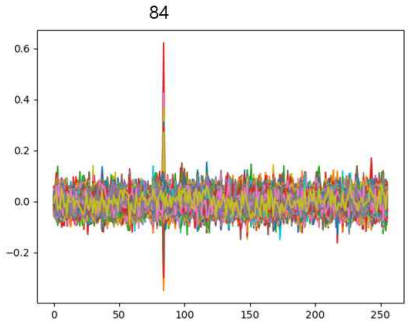


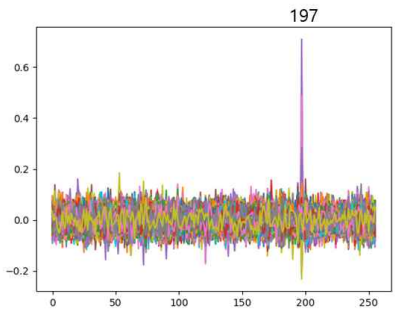
0x00~0xFF 중 위 과정을 진행한 것과 상관도가 가장 높은 값이 149 = 0x95 이다.

4번 문제 답안

마찬가지 방법으로 ②,③,④에 해당하는 Inv_Mix(K) 값을 구하기 위해 블록 1025개의 ②,③,④위치의 값(바이트)와 0x00~0xFF 값을 XOR 후 inverse subbyte를 진행하여 주어진 전력소비량과 CPA를 진행하면 다음과 같다.



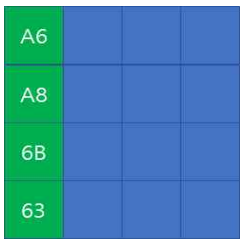




(②: 2=0x02, ③: 84=0x54, ④: 197=0xC5)

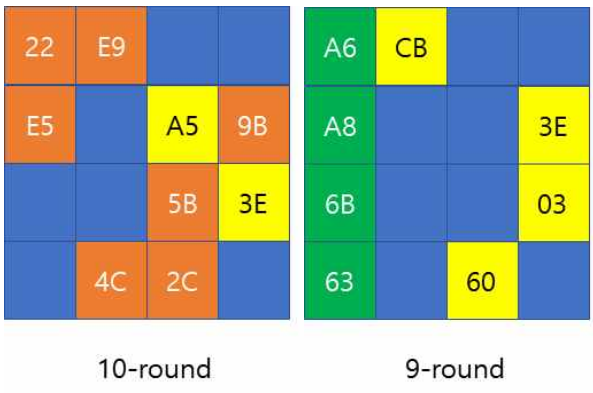
그러므로 CPA를 진행한 결과, ①,②,③,④ 값은 위와 같이 구한 값들이다. Inv_Mix(K)=K2 <=> K=Mix(K2) 임을 이용하면 실제 9라운드의 라운드키의 ①,②,③,④ 위치의 값은 위의 구한 결과에 Mixcolumns를 취한 값이므로 다음과 같다. (K: 9라운드의 라운드 키, K2: Inv_Mix(K)의 결과)

$$\begin{pmatrix} 2&3&1&1 \\ 1&2&3&1 \\ 1&1&2&3 \\ 3&1&1&2 \end{pmatrix} \begin{pmatrix} 95 \\ 02 \\ 54 \\ C5 \end{pmatrix} = \begin{pmatrix} A6 \\ A8 \\ 6B \\ 63 \end{pmatrix}$$



(9라운드 라운드키)

그러므로 9라운드의 라운드 키의 일부를 다음과 같이 구할 수 있고, 주어진 10라운드 키와 구한 9라운드 키를 이용하여 key-scheduling 과정을 이용해 추가적으로 10라운드, 9라운드 라운드 키에 대해 다음과 같이 구할 수 있다. (노란색 부분이 key-scheduling으로 구한 것)



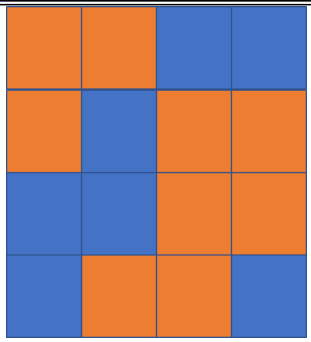
구한 10라운드의 라운드 키를 이용하여 암호문 1025개 블록에 대해 다시 위의 과정을 반복하자.

(1) 10라운드의 라운드키를 이용한 addroundkey 단계 후

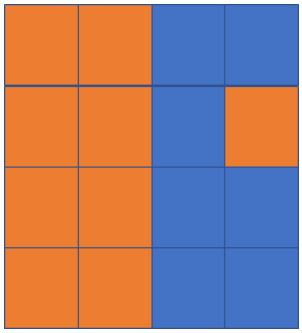
(2) 10라운드의 inverse subbyte, inverse shiftrows 단계 후

(3) 9라운드의 inverse mixcolumns 단계 후

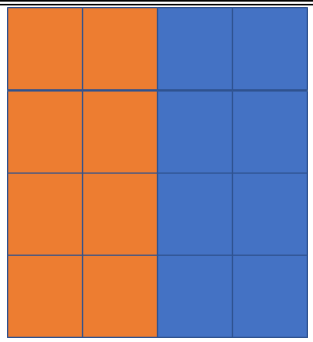
4번 문제 답안



(1)

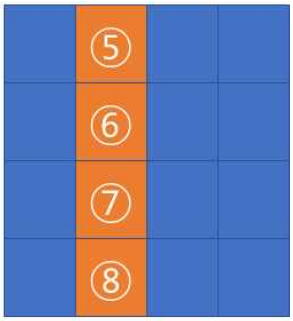


(2)

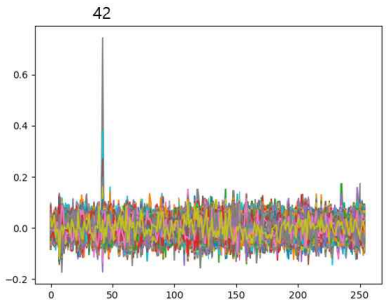


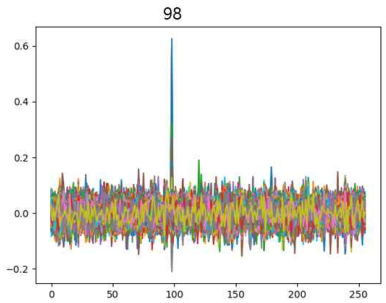
(3)

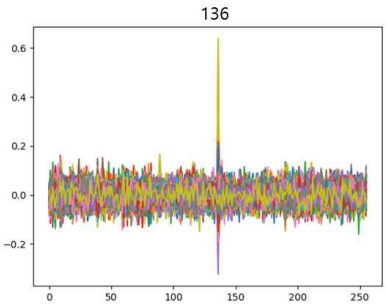
위와 마찬가지로 9라운드 라운드 키에 inverse mixcolumns를 취한 값을 이용하여 add round 단계를 계산하여야 하지만 9라운드 키의 5~8번 째 바이트 값을 모르므로 아래 그림의 ⑤,⑥,⑦,⑧에 해당하는 값 1025개에 대하여 각 1바이트 별로 0x00~0xFF 값을 더한 후 inverse subbyte, inverse shiftrows 단계를 진행한 결과와 전력소비량을 이용하여 CPA를 진행한다.

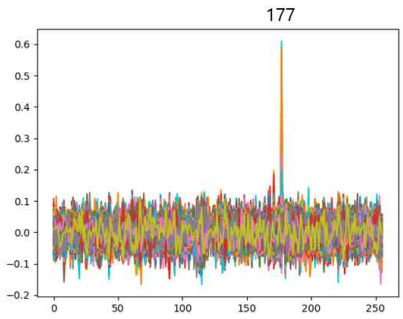


Inv_Mix(9round key)





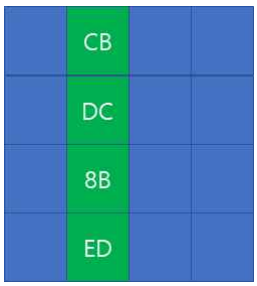




⑤: 42=0x2A, ⑥: 98=0x62,
⑦: 136=0x88, ⑧: 177=0xB1

그러므로 CPA를 진행한 결과, ⑤,⑥,⑦,⑧ 값은 위와 같이 구한 값들이다. Inv_Mix(K)=K2 <=> K=Mix(K2) 임을 이용하면 실제 9라운드의 라운드키의 ⑤,⑥,⑦,⑧ 위치의 값은 위 결과에 Mixcolumns를 취한 값이므로 다음과 같다. (K: 9라운드의 라운드 키, K2: Inv_Mix(K)의 결과, hex 값으로 계산하였다.)

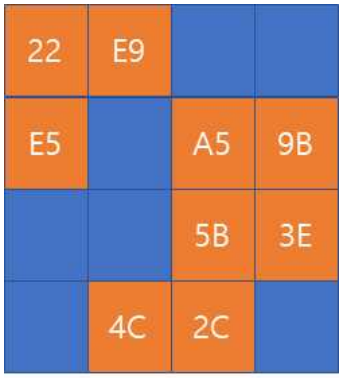
$$\begin{pmatrix} 2&3&1&1 \\ 1&2&3&1 \\ 1&1&2&3 \\ 3&1&1&2 \end{pmatrix} \begin{pmatrix} 2A \\ 62 \\ 88 \\ B1 \end{pmatrix} = \begin{pmatrix} CB \\ DC \\ 8B \\ ED \end{pmatrix}$$



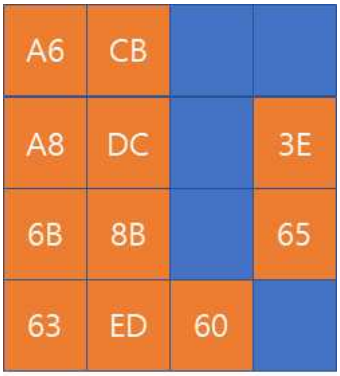
(9라운드 라운드키)

4번 문제 답안

현재까지 구한 10라운드, 9라운드 라운드 키를 종합하면 다음과 같다.

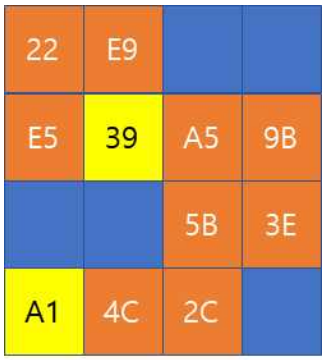


10round key

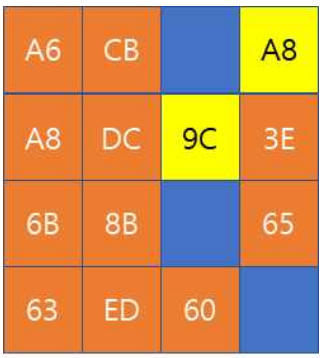


9round key

이를 이용하여 key-scheduling 과정을 활용하여 추가적으로 다음과 같이 키를 얻을 수 있다.
(노란색 부분이 key-scheduling을 이용하여 구한 것)



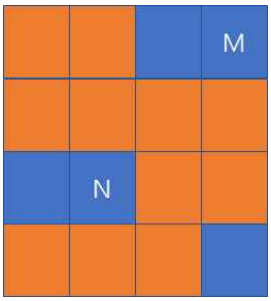
10round key



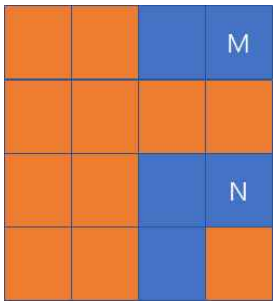
9round key

위와 같이 구한 10라운드, 9라운드 라운드 키를 활용하여 추가적으로 CPA를 적용하기 위해서는 다음과 같이 전수 조사량이 2^{16}가 됨을 알 수 있다.

(1) 10라운드 add roundkey 단계 후



(2) inverse subbyte, inverse shiftrows 단계 후



4번 문제 답안

기존의 방식으로 inverse mixcolumns를 먼저 진행하고 9라운드의 add roundkey 단계를 진행하기 위해서는 (2)을 진행 후 column 하나를 통째로 알아야한다. 그러므로 최소한의 전수조사를 위해서는 10라운드의 라운드 키의 N,M 위치에 해당하는 값을 알아야 하므로 2^{16}개의 경우에 대해 CPA를 적용해야한다.

하지만 10라운드, 9라운드의 라운드 키의 key-scheduling 관계를 이용하면 2^{16}가지의 경우에 대해 10라운드 키가 결정됨을 알 수 있고, hwp 파일의 file header signature가 'D0 CF 11 E0 A1 B1 1A E1' 임을 활용하면 CPA를 적용하지 않고 2^{16}가지의 경우에 대해 암호문의 첫 블록만 복호화를 진행하여 복호화한 평문의 첫 8바이트가 위와 동일한 결과(hwp 파일의 file header signature)를 얻은 경우가 나옴을 이용하자.

22	E9	x	x⊕A8
E5	39	A5	9B
y	y⊕8B	5B	3E
A1	4C	2C	inv_sub (y⊕6B) ⊕2C

10round key

이와 같이 이전까지 구한 10라운드, 9라운드의 라운드 키를 활용하여 10라운드의 라운드 키를 위와 같이 x,y에 대해 남은 값을 모두 나타낼 수 있다. (x,y)의 경우의 수는 총 2^{16}가지이며 한 블록만을 복호화하는 것은 빠르게 진행할 수 있다. (inv_sub : inverse subbyte)

AES128/CBC/PKCS#7, IV[16]={0x00,} 임을 이용하여 복호화하면 x=0x36, y=0x2C 임을 알 수 있고, 최종 10라운드 키와 암호화에 사용된 마스터키는 다음과 같다.

22	E9	36	9E
E5	39	A5	9B
2C	A7	5B	3E
A1	4C	2C	3A

10round key

84	06	2E	47
17	0A	E2	F3
FF	86	3D	5D
21	45	01	8E

master key

이를 이용하여 exam.hwp.encrypted 파일을 복호화를 진행하면 '**정보보호개론 중간고사**'를 얻을 수 있다.

또한, 상관전력분석에 사용한 전수조사 범위는 $2^8 \times 8 = 2^{11}$, 대칭키 전수조사 범위는 $2^8 \times 2^8 = 2^{16}$이다.

4번 문제 답안

그리고 “(전력분석에 사용한 키 전수조사 범위) * 1025 * 20,000 + (대칭키 암호키 전수조사 범위)”를 최소화하기 위해 기존에 주어진 10라운드 키 (마지막 라운드 키)를 이용한 키 스케쥴링으로 9라운드 키의 5번째 바이트 ‘0xCB’와 CPA를 위 ①,②,③,④ 바이트에 대해 진행한 이후를 고려해보자. 아래의 값은 모두 헥사값이다.

1) ⑤,⑥,⑦,⑧ 부분에 해당하는 바이트에 대해서 CPA를 모두 실행하지 않았을 때

앞선 방법으로는 ⑤,⑥,⑦,⑧ 부분에 해당하는 Inverse MixColumns(K)를 이용하여 CPA를 실행하여 9라운드 라운드키(K)의 5~8번째 바이트 값을 구하였다. 이 부분 모두를 다음과 같이 변수로 처리하여 대칭키 암호키 전수조사 범위를 확장하자.

	x		
	y		
	z		
	w		

Inv_Mix(K) (K : 9라운드 키)

$$\begin{pmatrix}2&3&1&1\\1&2&3&1\\1&1&2&3\\3&1&1&2\end{pmatrix}\begin{pmatrix}x\\y\\z\\w\end{pmatrix}=\begin{pmatrix}CB\\*\\*\\*\end{pmatrix}\Rightarrow CB=2x\oplus 3y\oplus z\oplus w\ on\ GF(2^8)$$

$$\Rightarrow w=CB\oplus 2x\oplus 3y\oplus z\ on\ GF(2^8)$$

그러면, 위 와 같이 변수 3개로 줄일 수 있고, 추가적인 2개의 변수 s, t를 이용하여, 9,10라운드 키에 대해 키 스케쥴링을 이용하여 대칭키 암호키 전수조사를 하면 다음과 같다. (invsub : inverse subbyte)

22	E9	s	$g(s,x,y,z)$
E5	$M_1(x,y,z)$ ⊕ E5	A5	9B
t	$t\oplus M_2(x,y,z)$	5B	3E
$M_3(x,y,z)$ ⊕ 4C	4C	2C	$f(t)\oplus$ 2C

10라운드 키

A6	CB	$s\oplus$E9	$h(x,y,z)$
A8	$M_1(x,y,z)$	$M_1(x,y,z)$ ⊕E5 ⊕A5	3E
6B	$M_2(x,y,z)$	$t\oplus M_2(x,y,z)$ ⊕ 5B	03
63	$M_3(x,y,z)$	60	$f(t)$

9라운드 키

$$\begin{pmatrix}2&3&1&1\\1&2&3&1\\1&1&2&3\\3&1&1&2\end{pmatrix}\begin{pmatrix}x\\y\\z\\w(x,y,z)\end{pmatrix}=\begin{pmatrix}CB\\M_1(x,y,z)\\M_2(x,y,z)\\M_3(x,y,z)\end{pmatrix},\quad f(t)=\mathrm{invsub}(t\oplus 6B),$$

$$h(x,y,z)=\mathrm{invsub}(M_3(x,y,z)\oplus 4C\oplus 63),\quad g(s,x,y,z)=s\oplus h(x,y,z)$$

이를 통해 hwp 파일의 file header signature를 이용하여 대칭키 암호키 전수조사를 시행하면 마스터키 복구 및 파일 복호화가 가능하다.

이 경우, 전력분석에 사용한 키 전수조사 범위는 $4\times 2^8=2^{10}$, 대칭키 암호키 전수조사 범위는 $(2^8)^5=2^{40}$ 이다.

4번 문제 답안

마찬가지로 ⑤,⑥,⑦,⑧ 부분에 해당하는 CPA를 1,2,3번 시행했을 때를 살펴보면 다음과 같다.

2) ⑤,⑥,⑦,⑧ 중 ⑤ 부분에 해당하는 바이트에 대해서만 CPA를 실행했을 때

1)과 마찬가지 방법으로 ⑤ 부분에 해당하는 바이트만 CPA를 이용해 계산하고, ⑥,⑦,⑧ 부분에 해당하는 바이트는 CPA를 실행하지 않고, 변수로 처리했을 때 다음과 같다. 앞선 결과를 이용하여, ⑤ 부분에 해당하는 바이트만 CPA를 이용하여 구하였을 때 '0x2A' 이고, 2*0x2A=0x54 이다.

	2A		
	x		
	y		
	z		

$$\begin{pmatrix}2&3&1&1\\1&2&3&1\\1&1&2&3\\3&1&1&2\end{pmatrix}\begin{pmatrix}2A\\x\\y\\z\end{pmatrix}=\begin{pmatrix}CB\\*\\*\\*\end{pmatrix}\Rightarrow CB=54\oplus 3x\oplus y\oplus z\ on\ GF(2^8)$$

$$\Rightarrow z=CB\oplus 54\oplus 3x\oplus y\ on\ GF(2^8)$$

Inv_Mix(K) (K : 9라운드 키)

이와 같이 Inverse MixColumns(K) 의 5~8번째 바이트에 해당하는 부분을 변수 2개에 대해 처리할 수 있고, (K : 9라운드 키) 위와 동일하게 9,10라운드 키에 대해 변수 s, t를 추가하여 키 스케쥴링을 적용하면 다음과 같다.

22	E9	s	$g(s,x,y)$
E5	$M_1(x,y)$ ⊕ E5	A5	9B
t	$t \oplus M_2(x,y)$	5B	3E
$M_3(x,y)$ ⊕ 4C	4C	2C	$f(t)\oplus$ 2C

10라운드 키

A6	CB	$s\oplus$E9	$h(x,y)$
A8	$M_1(x,y)$	$M_1(x,y)$ ⊕E5 ⊕A5	3E
6B	$M_2(x,y)$	$t \oplus M_2(x,y)$ ⊕ 5B	03
63	$M_3(x,y)$	60	$f(t)$

9라운드 키

$$\begin{pmatrix}2&3&1&1\\1&2&3&1\\1&1&2&3\\3&1&1&2\end{pmatrix}\begin{pmatrix}2A\\x\\y\\z(x,y)\end{pmatrix}=\begin{pmatrix}CB\\M_1(x,y)\\M_2(x,y)\\M_3(x,y)\end{pmatrix},\quad f(t)=\mathrm{invsub}(t\oplus 6B),$$

$$h(x,y)=\mathrm{invsub}(M_3(x,y)\oplus 4C\oplus 63),\quad g(s,x,y)=s\oplus h(x,y)$$

이를 이용하여, 대칭키 암호키 전수조사를 시행하여 위와 같은 방법으로 마스터키 복구 및 파일 복호화가 가능하다. 이 경우, 전력분석에 사용한 전수조사 범위는 5×2^8, 대칭키 암호키 전수조사 범위는 $(2^8)^4=2^{32}$ 이다.

4번 문제 답안

마찬가지 방법으로 ⑤,⑥,⑦,⑧ 중 ⑤,⑥ 부분에 해당하는 바이트만 CPA를 이용해 계산하였을 때, 변수 3개에 대해 대칭키 암호키 전수조사를 시행할 수 있고, ⑤,⑥,⑦,⑧ 중 ⑤,⑥,⑦ 부분에 해당하는 바이트만 CPA를 이용해 계산했을 때, 변수 2개에 대해 대칭키 암호키 전수조사를 시행할 수 있다.

주어진 식 "(전력분석에 사용한 키 전수조사 범위) * 1025 * 20,000 + (대칭키 암호키 전수조사 범위)"를 최솟값으로 갖는 경우는 " 2) ⑤,⑥,⑦,⑧ 중 ⑤ 부분에 해당하는 바이트에 대해서만 CPA를 실행했을 때" 에 해당하는 경우로써, 총 5바이트에 대해 CPA를 이용하여 계산하고, 변수 4개를 이용하여 대칭키 암호키 전수조사를 시행했을 때이다. 전력분석 전수조사 범위와 대칭키 전수조사 범위를 변경했을 때 주어진 식에 대입한 결과는 아래 표와 같다.

그러므로 "(전력분석에 사용한 키 전수조사 범위) * 1025 * 20,000 + (대칭키 암호키 전수조사 범위)"를 최솟값으로 갖는 경우는 아래 표와 같이, 전력분석에 사용한 키 전수조사 범위가 5×2^8, 대칭키 암호키 전수조사 범위가 $(2^8)^4 = 2^{32}$ 일 때 이다.

CPA 시행 횟수 (구한 byte 수)	대칭키 전수조사에 필요한 변수 개수	전력분석 전수조사 범위 / 대칭키 전수조사 범위	결과 (주어진 식)
4	5	$4\times 2^8 / (2^8)^5$	$1{,}120{,}503{,}627{,}776 \approx 2^{40.03}$
5	**4**	$5\times 2^8 / (2^8)^4$	**$30{,}534{,}967{,}296 \approx 2^{34.83}$**
6	3	$6\times 2^8 / (2^8)^3$	$31{,}504{,}777{,}216 \approx 2^{34.87}$
7	2	$7\times 2^8 / (2^8)^2$	$36{,}736{,}065{,}536 \approx 2^{35.10}$
8	2	$8\times 2^8 / (2^8)^2$	$41{,}984{,}065{,}536 \approx 2^{35.29}$

4번 문제 답안

② 복호화된 파일 내의 시험문제인 공개키 암호(전자서명) 해독 단계 풀이 :

위 문제 ('정보보호개론 중간고사')는 ECDSA 서명을 활용한 개인키를 찾는 문제이다. 주어진 ECDSA의 서명 과정을 살펴보면 다음과 같다.

1. 해쉬 함수를 이용하여 메시지 M에 대한 해쉬값 H(M)을 계산한다. (해쉬 함수 : SHA256)
2. Z는 H(M)의 이진값에서 왼쪽으로부터 n의 비트 길이만큼 자른 값이다.
(위 문제에서 n이 256bit, 주어진 메시지의 해쉬값은 SHA256을 이용하여 만들었으므로 H(M)은 256비트이다.
그러므로 Z=H(M) 이다.)
3. 1~(n-1) 값 중 random한 난수 k를 고른다.
4. $(x_1, y_1) = kG \pmod n$을 계산한다.
5. $r = x_1 \pmod n$을 계산한다. (r=0 (mod n) 이면 3번으로 돌아가 새로운 k를 고른다.)
6. 개인키 d에 대해 $s = k^{-1}(Z + rd) \pmod n$을 계산한다. (s=0 (mod n) 이면 3번으로 돌아가 새로운 k를 고른다.)
7. 메시지 M에 대한 서명은 (r,s) 이다.

문제에 주어진 r_1, r_2에 대해 $r_1 = r_2$ 이고 EC (Elliptic Curve)에서 EC 위의 점 P에 대해 P=(x,y), -P=(x,-y) 임을 활용하면 다음과 같다. (k_1, k_2 : $S_1(r_1, s_1)$, $S_2(r_2, s_2)$ 서명 과정에 쓰인 난수)

$r_1 = x_1 = (k_1 G)(x) \pmod n, r_2 = x_2 = (k_2 G)(x) \pmod n$ 이므로 주어진 G에 대해 k_1, k_2번 연산 시 x좌표가 같으므로 modulo n에 대해 $k_1 = k_2$ or $k_1 = -k_2$ 이다. ($(k_i G)(x)$: $k_i G$의 x좌표, $i = 1, 2$)

① $k_1 = k_2$ 일 때

1. $r_1 = r_2 = r$, $k_1 = k_2 = k$라 하자.
2. $s_1 = k^{-1}(h(M_1) + rd) \pmod n$, $s_2 = k^{-1}(h(M_2) + rd) \pmod n$ 을 만족한다.
(s_1, s_2 : 주어진 서명, $h(M_1), h(M_2)$: 주어진 메시지 M_1, M_2의 $SHA256(M_1), SHA256(M_2)$ 값)
3. $k = (s_1)^{-1}(h(M_1) + rd) = (s_2)^{-1}(h(M_2) + rd) \pmod n$ 이므로 아래의 식을 얻는다.

$$(h(M_1) + rd)(s_2) = (h(M_2) + rd)(s_1) \pmod n$$
$$\Rightarrow (h(M_1)s_2 - h(M_2)s_1) = rd(s_1 - s_2) \pmod n$$
$$\Rightarrow (h(M_1)s_2 - h(M_2)s_1) r^{-1}(s_1 - s_2)^{-1} = d \pmod n$$

4. 그러므로 문제에 주어진 서명값과 메시지의 해쉬값을 이용하여 개인키 d를 계산할 수 있다.
5. d=0x85cd61964a0aaecdf53e19eb254cb1c07fd7ee85b4a7474fea7645cebf2432d4 이다.
이를 주어진 공개키 Q를 이용하여 검증 시 Q=dG를 만족하지 않는다.

4번 문제 답안

② $k_1 = -k_2$ 일 때

1. $r_1 = r_2 = r,\ k_1 = k, k_2 = -k$라 하자.

2. $s_1 = k^{-1}(h(M_1)+rd) \pmod n,\ s_2 = (-k)^{-1}(h(M_2)+rd) \pmod n$ 을 만족한다.

3. $k = (s_1)^{-1}(h(M_1)+rd) = (-s_2)^{-1}(h(M_2)+rd) \pmod n$ 이므로 아래의 식을 얻는다.

$$(h(M_1)+rd)(-s_2) = (h(M_2)+rd)(s_1) \pmod n$$
$$\Rightarrow (-h(M_1)s_2 - h(M_2)s_1) = rd(s_1+s_2) \pmod n$$
$$\Rightarrow (-h(M_1)s_2 - h(M_2)s_1)r^{-1}(s_1+s_2)^{-1} = d \pmod n$$

4. 문제에 주어진 서명값과 메시지의 해쉬값을 이용하여 개인키 d를 계산할 수 있다.

5. d=0xbea4fd03c804ea0160a096f4c6b438d54ab78458e124e8f68d42cd7010807a1b 이다.
이를 주어진 공개키 Q를 이용하여 검증시 Q=dG를 만족한다.

그러므로 개인키 **d=0xbea4fd03c804ea0160a096f4c6b438d54ab78458e124e8f68d42cd7010807a1b** 이다.