情報保護學會論文誌
第19卷 第2號, 2009. 4

AES의 연관키 렉탱글 공격에 대한 안전성 분석*

김 종 성,[1†] 홍 석 희,[1] 이 창 훈[2‡]
[1]고려대학교 정보보호연구원, [2]한신대학교 컴퓨터공학부

Security Analysis of AES for Related-Key Rectangle Attacks*

Jongsung Kim,[1†] Seokhie Hong,[1] Changhoon Lee[2‡]
[1]CIST, Korea University, [2]School of Computer Engineering, Hanshin University

요 약

본 논문에서는 미 연방 표준 블록 암호 AES에 대한 기존의 9 라운드 연관키 렉탱글 공격을 10 라운드로 향상시킨다. 256개의 연관키를 사용하는 12 라운드 AES-192의 첫 10 라운드는 2^{124}의 데이터 복잡도와 2^{183}의 시간 복잡도로 공격되며, 64개의 연관키를 사용하는 AES-192의 첫 10 라운드는 2^{122}의 데이터 복잡도와 $2^{183.6}$의 시간 복잡도로 공격된다. 본 논문의 공격은 AES-192에 대한 기존 공격 중 최상의 공격이다.

ABSTRACT

In this paper we improve previous related-key rectangle attacks on AES from 9 rounds to 10 rounds: Our attacks break the first 10 rounds of 12-round AES-192 with 256 related keys, a data complexity of $2^{124}$and a time complexity of 2^{183}, and also break the first 10 rounds of 12-round AES-192 with 64 related keys, a data complexity of 2^{122} and a time complexity of $2^{183.6}$. Our attacks are the best knoown attacks on AES-192.

Keywords: Security analysis, Related-key rectangle Attack, AES

I. 서 론

AES(Advanced Encryption Standard)는 미 연방 표준 128 비트 블록 암호로 128, 192 또는 256 비트의 키를 사용하며, 각각의 키 길이에 따라 10, 12 또는 14 라운드를 사용한다[1]. AES의 한 라운드는 비 선형층 SubBytes 함수를 적용한 후, 선형층 ShiftRows, MixColumns, AddRoundKey 함수를 차례로 적용한다. 또한, 첫 번째 라운드 전에 AddRoundKey 함수를 적용하고, 마지막 라운드에서는 MixColumns 함수를 생략한다. 본 논문에서는 192 비트 키를 사용하는 AES의 안전성 분석에 초점을 마춘다 (이하 192 비트 키를 사용하는 AES를 AES-192로 표기한다).

본 논문에서는 AES-192에 대한 기존의 9 라운드 연관키 렉탱글 공격[2]을 10 라운드로 향상시킨다. 본 논문의 10라운드 AES-192 공격 결과는 다음과 같다: 256개의 연관키를 사용하는 첫 10 라운드 축소 AES-192는 2^{124}의 데이터 복잡도와 2^{183}의 시간 복잡도로, 64개의 연관키를 사용하는 첫 10 라운드 축소 AES-192는 2^{122}의 데이터 복잡도와 $2^{183.6}$의 시간 복잡도로 공격된다. 본 논문의 공격은 AES-192에 대한 기존 공격 중 최상의 공격이다. AES에 대한 본 논문의 결과와 기존 공격 결과는 [표 1]과 같다.

본 논문의 구성은 다음과 같다. 2장에서는 AES-192 알고리즘과 연관키 렉탱글 공격 방법을 간략히 다룬다. 3장에서 256개의 연관키를 갖는 10 라운드 축소 AES-192 공격을 소개하고, 4장에서 64개의 연관키를 갖는 10 라운드 축소 AES-192 공격을 소개한다. 끝으로, 5장은 본 논문의 결론 부분이다.

접수일(2008년 11월 11일), 게재확정일(2009년 3월 23일)
* 이 연구에 참여한 연구자의 일부는 '2단계 BK21사업'의 지원비를 받았음.
† 주저자, joshep@cist.korea.ac.kr
‡ 교신저자, chlee@hs.ac.kr

〔표 1〕 AES-192에 대한 공격 결과

블록암호	공격 유형	공격 라운드	연관키 수	데이터 복잡도	공격 복잡도
AES-192	불능 차분	7	1	2^{92} CP	2^{162}〔3〕
	포화	7	1	2^{32} CP	2^{184}〔4〕
	연관키 (불능) 차분	7	2	2^{111} RK-CP	2^{116}〔5〕
		7	32	2^{56} CP	2^{94}〔6〕
		8	2	2^{88} RK-CP	2^{183}〔5〕
		8	32	$2^{68.5}$ CP	2^{184}〔6〕
	연관키 렉탱글[1]	8	2	2^{94} RK-CP	2^{120}〔7〕
		9	256	2^{86} RK-CP	2^{125}〔8〕
		10	256	2^{124} RK-CP	2^{183} (본논문)
		10	64	2^{122} RK-CP	$2^{183.6}$(본논문)
	연관키 차분-선형	7	2	2^{22} RK-CP	2^{187}〔3〕
		8	2	2^{118} RK-CP	2^{165}〔3〕

CP: 선택 평문, ACPC: 능동적 선택 평문과 암호문, RK: 연관키.
Time: Encryption units.

II. AES-192 알고리즘의 소개

AES-192는 12 라운드로 구성된 128 비트 블록 암호로 192 비트 키를 사용한다〔1〕. AES-192의 암호화 과정 상의 128 비트 상태 값은 〔그림 1〕과 같이 16개 바이트로 이루어진 4X4 행렬로 나타낼 수 있다. AES-192의 한 라운드는 SubBytes(SB), ShiftRows (SR), MixColumns(MC), AddRoundKey(ARK) 함수를 차례대로 사용하며, 첫 번째 라운드 전에 ARK 함수를 적용하고(화이트닝 키), 마지막 라운드에서 MC 함수를 생략한다. 각 함수는 다음과 같이 동작한다.

- BS 함수는 각각의 바이트에 동일한 비선형 S-박스를 적용한다.
- SR 함수는 행 0,1,2,3을 왼쪽으로 각각 0,1,2,3 바이트 순환 이동 시킨다.
- MC 함수는 선형 변환으로 4 바이트로 구성된 각 열을 변환시키는 4X4 행렬로 GF(2^8) 위에서 연산한다.
- ARK 함수는 키와 상태값의 비트별 합 연산을 수행한다.

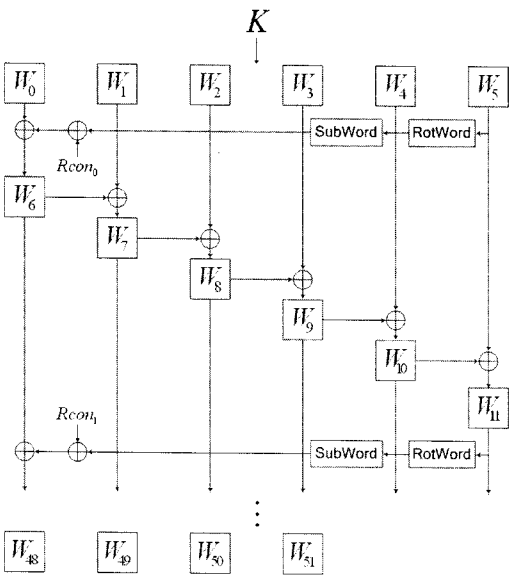
0	4	8	12
1	5	9	13
2	6	10	14
3	7	11	15

〔그림 1〕 AES의 128비트 상태값

AES-192의 키스케줄은 6개의 32-비트 워드 ($W_0, W_1, W_2, W_3, W_4, W_5$)를 입력하여 $4 \cdot 13 = 52$개의 32-비트 워드를 생성하는 과정이다. 생성된 키의 32-비트 워드를 W_i라고 한다면, 화이트닝 키 $K^w = (W_0 \| W_1 \| W_2 \| W_3)$이고 첫번째 라운드 키 $K^0 = (W_4 \|$

1) 본 논문의 AES-192에 대한 연관키 렉탱글 공격은 FSE 2007에 발표된 〔9〕의 공격 알고리즘보다 효율적인 공격 알고리즘을 포함한다. 〔9〕의 결과는 본 논문에서 이용한 것과 같이 최악의 경우 고려 시 ([9]에서 제시한 공격 결과는 평균의 경우를 고려하였음), 256개의 연관키를 사용하는 10 라운드 AES-192 공격은 2^{126}의 데이터 복잡도와 2^{183}의 시간 복잡도를 가지며, 64개의 연관키를 사용하는 10 라운드 AES-192 공격은 2^{125}의 데이터 복잡도와 2^{184}의 시간 복잡도를 가진다.

$W_5 \| W_6 \| W_7$)이다. 이러한 순서로 12 라운드의 AddRoundkey() 변환에 사용된다. 이 때 두 개의 변환 RotWord()와 SubWord()가 사용된다. RotWord()는 4-바이트 워드 입력값을 좌측으로 1-바이트 순환 이동시키는 변환이고, SubWord()는 입력받은 4-바이트 워드 입력값의 각 바이트에 S-box를 적용하는 변환이다. 다음은 AES-192 키 스케줄을 나타낸 것이다 ([그림 2] 참조). $Rcon_i = RC_i \| 00 \| 00 \| 00$, $RC_i = (02)^{i-1}$.



[그림 2] AES-192의 키스케줄

```
for (i=6; i<51; i++){
    if (i≡0 mod6)

W_i = W_{i-6} ⊕_{Word}(RotWord(W_{i-1})) ⊕ Rcon[i/6];
    else W_i = W_{i-1} ⊕ W_{i-6};
}
```

III. 연관키 렉탱글 공격

연관키 렉탱글 공격[2,10]의 아이디어는 하나의 낮은 확률을 가지는 긴 연관키 차분 특성 대신에 높은 확률을 가지는 두 개의 짧은 연관키 차분 특성을 이용하는 것이다. $\{0,1\}^k$와 $\{0,1\}^n$을 각각 비밀키 공간과 평문/암호문 공간이라 할 때, 블록 암호 $E: \{0,1\}^n \times \{0,1\}^k \rightarrow \{0,1\}^n$를 다음과 같이 두 개의 sub-cipher의 연결로 생각한다: $E = E^1 \circ E^0$. 이 때, E^0, E^1에 대한 연관키 차분 특성의 확률을 다음과 같이 정의하자.

$$\hat{p} = \sqrt{\sum_{\beta} (\Pr_{X,K}[E_K^0(X) \oplus E_{K \oplus \Delta K_{ab}}^0(X \oplus \alpha) = \beta])^2},$$
$$\hat{q} = \sqrt{\sum_{\gamma} (\Pr_{X,K}[E_K^1(X) \oplus E_{K \oplus \Delta K_{ac}}^1(X \oplus \gamma) = \delta])^2}.$$

그러면, $\hat{p} \cdot \hat{q} > 2^{-n/2}$이면 연관키 렉탱글 공격을 적용할 수 있다. 연관키 렉탱글 distinguisher는 다음과 같이 구성된다.

① N개의 평문쌍 $(P_a, P_b = P_a \oplus \alpha)$를 임의로 선택한다. 이 평문쌍의 집합을 S로 표기한다. 평문 P_a와 키 K_a에 대응하는 암호문 C_a와 평문 P_b와 키 K_b에 대응하는 암호문 C_b를 얻는다 $(K_a \oplus K_b = \Delta K_{ab})$.

② N개의 평문쌍 $(P_c, P_d = P_c \oplus \alpha)$를 임의로 선택한다. 이 평문쌍의 집합을 T로 표기한다. 평문 P_c와 키 K_c에 대응하는 암호문 C_c와 평문P_d와 키 K_d에 대응하는 암호문 C_d를 얻는다 $(K_c \oplus K_d = \Delta K_{ab}, K_a \oplus K_c = K_b \oplus K_d = \Delta K_{ac})$.

③ 대응하는 암호문쌍이 다음을 만족하는 2개의 평문쌍 $(P_1, P_2) \in S$와 $(P_3, P_4) \in T$를 찾는다. $C_a \oplus C_c = C_b \oplus C_d = \delta$.

S에 속하는 N개의 평문쌍과 T에 속하는 N개의 평문쌍을 이용하여 N^2개의 quartet을 구성할 수 있으므로, ③의 δ를 만족시키는 올바른 quartet의 기댓값은 $N^2 \cdot 2^{-n} \cdot \hat{p}^2 \cdot \hat{q}^2$이다. random cipher의 경우, 이러한 quartet의 기댓값은 $N^2 \cdot 2^{-2n}$이 된다. 따라서, $\hat{p} \cdot \hat{q} > 2^{-n/2}$이면 이러한 차분 특성을 이용하여 연관키 렉탱글 공격이 가능하다.

IV. 256개의 연관키를 사용하는 10 라운드 AES-192에 대한 연관키 렉탱글 공격

본 장에서는 10 라운드 AES-192를 $E = E^f \circ E^1 \circ E^0 \circ E^b$으로 표기한다. 여기에서 E^b는 화이트닝 단계부터 첫 번째 라운드의 MixColumn()까지의 암호화 과정을 의미하고, E^0은 첫 번째 라운드의 AddRoundKey()를 포함한 라운드 1~4의 암호화 과정을 의미한다. 또한, E^1과 E^f는 각각 라운드 5~8

의 암호화 과정과 마지막 라운드의 암호화 과정을 의미한다. 본 공격에 E^0는 [그림 3]과 같은 연관키 차분 특성을 사용하고, E^1은 [그림 4]와 같은 연관키 차분 특성을 사용한다. 이러한 연관키 차분을 이용하여 $E^1 \circ E^0$에 대한 연관키 렉탱글 distinguisher를 구성하고, 이를 이용하여 E^b와 E^f에 사용된 부분 라운드 키를 복구한다. 본 공격에서는 다음과 같은 표기를 사용한다.

- $K_a^w, K_b^w, K_c^w, K_d^w$: K_a, K_b, K_c, K_d으로부터 생성된 화이트닝 키.
- $K_a^i, K_b^i, K_c^i, K_d^i$: K_a, K_b, K_c, K_d으로부터 생성된 라운드 i의 라운드 키.
- P_a, P_b, P_c, P_d: K_a, K_b, K_c, K_d으로 암호화 되는 평문.
- $I_a^i, I_b^i, I_c^i, I_d^i$: K_a, K_b, K_c, K_d으로 P_a, P_b, P_c, P_d를 암호화할 때, 라운드 i의 입력값.
- ΔK_{ab}^i: $K_a^i \oplus K_b^i$ 또는 $K_c^i \oplus K_d^i$.
- ΔK_{ac}^i: $K_a^i \oplus K_c^i$ 또는 $K_b^i \oplus K_d^i$.
- ΔI_{ab}^i: $I_a^i \oplus I_b^i$ 또는 $I_c^i \oplus I_d^i$.
- ΔI_{ac}^i: $I_a^i \oplus I_c^i$ 또는 $I_b^i \oplus I_d^i$.
- x: 고정된 0이 아닌 차분 값.
- y, z: 입력 차분 x에 대한 S-box의 출력 차분.
- $*$: 알려지지 않은 바이트 변수.

4.1 8 라운드 연관키 렉탱글 distinguisher[2)]

[그림 3]과 [그림 4]의 연관키 차분 특성은 AES-192의 키스케줄에서 차분의 확산 효과가 적은 특성을 이용하여 구성되었다. 즉, $HW_b(X)$가 X의 바이트 단위 해밍 무게를 나타낼 때, $HW_b(\Delta K_{ab}^0) = HW_b(\Delta K_{ac}^5) = 2$, $HW_b(\Delta K_{ab}^1) = HW_b(\Delta K_{ac}^6) = 0$, $HW_b(\Delta K_{ab}^2) = HW_b(\Delta K_{ac}^7) = 1$을 만족하는 라운드 3의 라운드 키 차분 $\Delta K_{ab}^0 \| \Delta K_{ab}^1 \| \Delta K_{ab}^2$와 $\Delta K_{ac}^5 \| K_{ac}^6 \| \Delta K_{ac}^7$을 구성할 수 있는 특성을 이용한다. 이러한 작은 해밍 무게를 가지는 키 차분을 이용하여 $\Delta I_{ab}^1 = \Delta I_{ac}^6 = 0$인 연관키 차분 특성을 구성하면, $HW(\Delta I_{ab}^3) = HW(\Delta I_{ac}^8) = 1$을 만족한다. 그리고 1~2 라운드를 확장한 연관키 차분 특성을 구성할 수 있다. 이렇게 구성한 4 라운드 연관키 차분 특성을 8 라운드 연관키 렉탱글 distinguisher에 적용하기 위하여 다음을 가정한다.

- 가정 1. 키 quartet (K_a, K_b, K_c, K_d)은 다음을 만족한다.

$$K_a \oplus K_b = K_c \oplus K_d = \Delta K_{ab},$$
$$K_a \oplus K_c = K_b \oplus K_d = \Delta K_{ac}.$$

- 가정 2. 평문 quartet (P_a, P_b, P_c, P_d)은 다음을 만족한다.

$$P_a \oplus P_b,\ P_c \oplus P_d \in \Delta P_{ab}.$$

- 가정 3.

$$E_{K_a}^b(P_a) \oplus E_{K_b}^b(P_b) = E_{K_c}^b(P_c) \oplus E_{K_d}^b(P_d) = \Delta K_{ab}^0$$

E^0의 연관키 차분 특성에 의해서 $I_a^5 \oplus I_b^5 = I_c^5 \oplus I_d^5$일 확률 $\hat{p} = 2^{39}$이다. $(2^{-32} \cdot 2^{-7})^2 \cdot (2^7 - 2) \cdot 2^{32} + (2^{-32} \cdot 2^{-6})^2 \cdot 2^{32} \approx 2^{-39}$또한, [그림 2-6]에서 가리키는 것과 같이, $|\Delta I_{ac}^5| = 2^{64}$의 조건 하에서 $I_a^6 \oplus I_c^6 = 0$일 확률은 2^{-64}이고, $E_{K_a}^1(I_a^5) \oplus E_{K_b}^1(I_b^5), E_{K_c}^1(I_c^5) \oplus E_{K_d}^1(I_d^5) \in \Delta I_{ac}^9$일 확률은 $2^{-128}(\hat{p} \cdot \hat{q})^2 = 2^{-231}$이다.

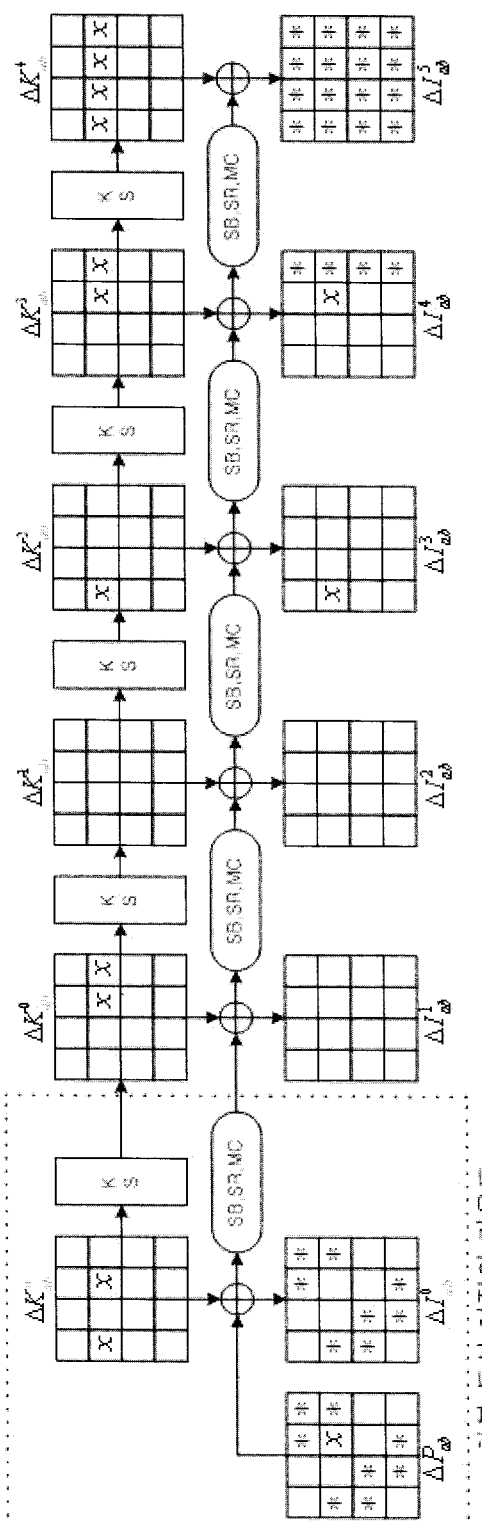
random cipher에 대해서는, ΔI_{ac}^9의 원소가 $(2^7 - 1)$개 이므로, 동일한 조건을 만족하는 quartet이 존재할 확률은 $(2^{-128} \cdot (2^7 - 1))^2 \approx 2^{-242}$이다. ΔI_{ac}^9의 첫 번째 열 B는 다음과 같다. 여기서 MC는 MixColumns()을 의미하고, SB는 SubBytes()를 의미한다.

$$B = \left\{ \begin{matrix} \mathrm{MC}(j,0,0,0) \mid j = \mathrm{SB}(i) \oplus \mathrm{SB}(x \oplus i), \\ i = 0,1,2,\cdots,255 \end{matrix} \right\}$$

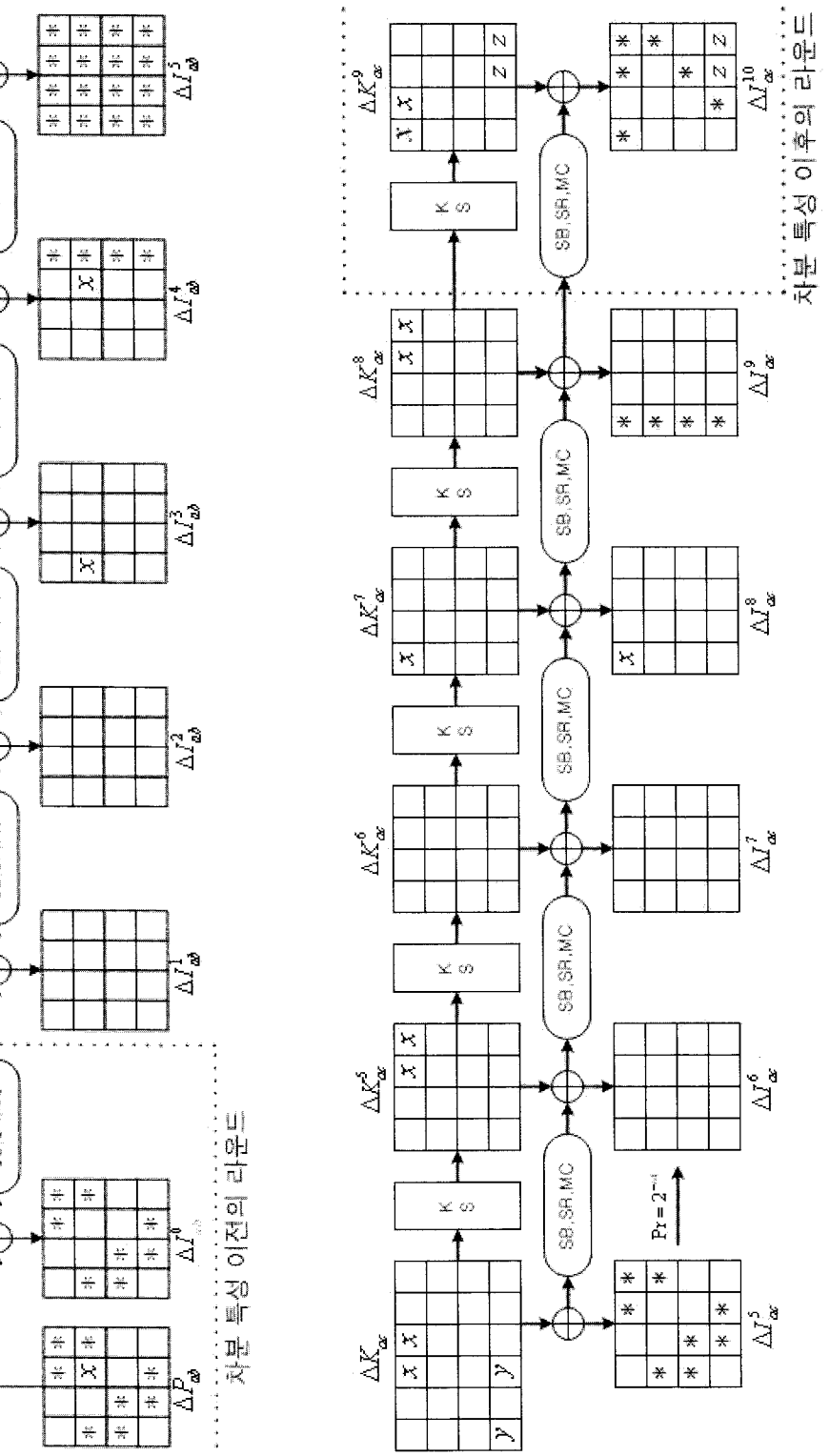
4.2 256개 연관키를 사용하는 10 라운드 AES-192에 대한 키 복구 공격

[그림 4]에서 나타내는 키 차분을 생성하기 위해서, ΔK_{ac}^3의 세 번째 열에서 8-비트 차분 x는 S-box를 통과한 후 y가 되어야 한다. 주어진 8-비트 차분 x에 대해 127개의 y가 가능하므로, $K_b = K_a \oplus \Delta K_{ab}$, $\widetilde{K_c} = K \oplus \Delta \widetilde{K_{ac}}$, $\widetilde{K_d} = K_a \oplus \Delta K_{ab} \oplus \Delta \widetilde{K_{ac}}$을 만족시킬 수 있는 가능한 모든 키 quartet $(K_a, K_b, \widetilde{K_c}, \widetilde{K_d})$을 이용한다. 즉, ΔK_{ab}는 [그림 3]의 ΔK_{ab}^w와 ΔK_{ab}^0의 처음 두 열로 고정되고, $\Delta \widetilde{K_{ac}}$는 입력 차분 x에 대해 S-box를 통과하여 나올 수 있는 127개의 출력 차분 y'에 의해 127개의 가능한 후보를 가진다. 그러므로 256개의 연관키를 이용하여 127개의 키 quartet $(K_a, K_b, \widetilde{K_c}, \widetilde{K_d})$을 구성

2) 본 소절에서 다루는 AES-192에 대한 8 라운드 렉탱글 distinguisher는 [9]에서 소개되었다.



〔그림 3〕 E^0의 연관키 차분 특성



〔그림 4〕 E^1의 연관키 차분 특성

하고 각각의 키 quartet에 대하여 10 라운드 AES-192에 대한 렉탱글 공격을 시도한다. 이 과정을 $y'=y$가 되는 키 quartet을 찾을 때까지 반복한다. 즉, 다음을 만족한다.

$$\Delta\widetilde{K}_{ac}=\Delta K_{ac},\ (K_a, K_b, \widetilde{K}_c, \widetilde{K}_d)=(K_a, K_b, K_c, K_d).$$

본 공격을 통하여 화이트닝 키 quartet $(K_a^w, K_b^w, K_c^w, K_d^w)$의 1, 2, 6, 7, 8, 11, 12, 13번째 바이트와 9 라운드 키 quartet $(K_a^9, K_b^9, K_c^9, K_d^9)$의 0, 7, 8, 10, 12, 13번째 바이트를 복구한다. 이 바이트는 [그림 3]과 [그림 4]의 ΔP_{ab}와 ΔI_{ac}^{10}에서 $*$로 표시된 부분을 의미한다. 그러므로 공격에서 2^{64}의 화이트닝 키 quartet 추측과 2^{72}의 라운드 9의 라운드 키 quartet의 추측이 필요하다. d, e, f가 알려지지 않은 8-비트 값일 때, ΔK_{ab}^9의 0, 7, 8, 10, 12, 13번째 바이트의 차분은 각각 d, 0, d, e, 0, f이고, ΔK_{ac}^9의 0, 7, 8, 10, 12, 13번째 바이트의 차분은 각각 a, 0, 0, 0, 0, 0이다. 그러므로 차분 y를 가지는 후보를 추측하기 위해서 총 2^{143}의 키 추측이 필요하다. 공격 알고리즘은 다음과 같다.

4.2.1 데이터 수집 단계

① 2^{52}의 structure $S_a^1, S_a^2, \cdots, S_a^{2^{72}}$를 선택한다. 각각의 structure는 0, 3, 4, 5, 9, 10, 14, 15번째 바이트를 고정한 2^{64}개의 평문으로 이루어진다. 각 structure의 평문에 대해 K_a를 이용하여 대응하는 암호문을 얻는다 (단계 ①은 2^{116}의 선택 평문 query가 필요하다).

② 128-비트 M이 9번째 바이트만이 a이고 나머지 바이트는 0일 때, 각 $S_a^1, S_a^2, \cdots, S_a^{2^{72}}$의 평문들에 M을 XOR하여 2^{52}의 structure $S_b^1, S_b^2, \cdots, S_b^{2^{72}}$를 계산한다. 각 structure의 평문에 대해 K_b를 이용하여 대응하는 암호문을 얻는다. 여기서, $K_b=K_a\oplus\Delta K_{ab}$이다 (단계 ②도 단계 ①과 유사하게 2^{116}의 선택 평문 query가 필요하다).

③ 차분 y에 대한 후보를 추측하고 $\Delta\widetilde{K}_{ac}$를 계산한다. 각각의 차분 $\Delta\widetilde{K}_{ac}$에 대해 다음을 수행한다.

㉠ 2^{52}의 structure $S_c^1, S_c^2, \cdots, S_c^{2^{72}}$를 선택한다. 각각의 structure는 0, 3, 4, 5, 9, 10, 14, 15번째 바이트를 고정한 2^{64}개의 평문으로 이루어진다. 각 structure의 평문에 대해 $\widetilde{K}_c$를 이용하여 대응하는 암호문을 얻는다. 여기서, $\widetilde{K}_c=K_a\oplus\Delta\widetilde{K}_{ac}$이다 (각각의 $\Delta\widetilde{K}_{ac}$의 추측에 대하여, 단계 ③-㉠은 2^{116}의 선택 평문 query가 필요하다).

㉡ 각 $S_c^1, S_c^2, \cdots, S_c^{2^{72}}$의 평문들에 M을 XOR하여 2^{54}의 structure $S_d^1, S_d^2, \cdots, S_d^{2^{72}}$를 계산한다. $\widetilde{K}_d=K\oplus\Delta K_{ab}\oplus\Delta\widetilde{K}_{ac}$일 때, 각 structure의 평문에 대해 $\widetilde{K}_d$를 이용하여 대응하는 암호문을 얻는다 (각각의 $\Delta\widetilde{K}_{ac}$의 추측에 대하여, 단계 ③-㉡는 2^{116}의 선택 평문 query가 필요하다).

4.2.2 E^b와 암호문 quartet 분석 단계

④ 화이트닝 키 k_a^w의 1, 2, 6, 7, 8, 11, 12, 13번째 바이트를 추측한다. 그리고 $k_b^w=k_a^w\oplus\Delta k_{ab}^w$, $k_c^w=k_a^w\oplus\Delta\tilde{k}_{ac}^w$, $k_d^w=k_a^w\oplus\Delta k_{ab}^w\oplus\Delta\tilde{k}_{ac}^w$를 계산한다. 여기서 Δk_a^w와 $\Delta\tilde{k}_{ac}^w$는 각각 (그림 2-6)의 ΔK_w와 $\Delta\widetilde{K}_{ac}^w$의 1, 2, 6, 7, 8, 11, 12, 13번째 바이트가 고정된 차분을 의미한다. 각각의 화이트닝 키 quartet $(K_a^w, K_b^w, K_c^w, K_d^w)$에 대하여 다음을 수행한다.

㉠ i는 $1, 2, \cdots, 2^{52}$이고 l_0는 $1, 2, \cdots, 2^{64}$일 때, S_a^i에 포함되는 각각의 평문 P_a^{i,l_0}에 대하여 k_a^w를 이용하여 E^b를 부분 암호화한다. 이렇게 암호화 된 값을 x_a^{i,l_0}라 표기한다. $x_a^{i,l_0}\oplus\Delta K_{ab}^{0,R}$에 대해 k_b^w를 이용하여 E^b를 부분 복호화한다 ($\Delta K_{ab}^{0,R}$은 ΔK_{ab}^0의 오른쪽 절반을 나타낸다). 그리고 이 값에 대응하는 S_b^i에 속하는 평문을 찾는다. 이를 P_b^{i,l_0}로 표기한다. 또한, P_a^{i,l_0}와 P_b^{i,l_0}에 대응하는 암호문을 각각 C_a^{i,l_0}와 C_b^{i,l_0}로 표기한다 (각각의 64-비트 k_a^w의 추측에 대해, 단계 ④-㉠에서는 약 $2^{64+1}\cdot(8/16)\cdot(1/10)=2^{60.7}$ 암호화 연산이 필요하다. 여기서 n 암호화 연산은 n번의 10 라운드 AES-192의 암호화 연산을 의미한다. 이 단계는 단계 ③의 $\Delta\widetilde{K}_{ac}$과 무관하기 때문에 단계 ③을 반복할 때마다 계산하지 않는다).

㉡ j는 $1, 2, \cdots, 2^{52}$이고 l_1는 $1, 2, \cdots, 2^{64}$일 때, S_c^j에 포함되는 각각의 평문 P_c^{j,l_1}에 대하여 k_c^w를 이용하

여 E^b를 부분 암호화한다. 이렇게 암호화 된 값을 x_c^{j,l_1}라 표기한다. $x_c^{j,l_1} \oplus \Delta K_{ab}^{0,R}$에 대해 k_d^w를 이용하여 E^b를 부분 복호화한다. 그리고 이 값에 대응하는 S_d^j에 속하는 평문을 찾는다. 이를 P_d^{j,l_1}로 표기한다. 또한, P_c^{j,l_1}와 P_d^{j,l_1}에 대응하는 암호문을 각각 C_c^{j,l_1}와 C_c^{j,l_1}로 표기한다 (각각의 71-비트 $(k_a^w, \Delta\widetilde{K_{ac}})$의 추측에 대해, 단계 ④-㉡는 약 $2^{64+1} \cdot (8/16) \cdot (1/10) = 2^{60.7}$ 암호화 연산이 필요하다).

㉢ 암호문의 1, 2, 3, 4, 5, 6, 9, 14번째 바이트의 순서대로 $C_a^{j,l_0} \| C_b^{j,l_0}$를 해쉬 테이블에 저장한다. 그리고 모든 i, j, l_0, l_1에 대해 $(C_a^{j,l_0} \| C_b^{j,l_0}) \oplus \oplus (C_c^{j,l_1} \| C_d^{j,l_1}) \in \Delta I_{ac}^{10}(1) \| \Delta I_{ac}^{10}(2)$인지 검사한다. 여기서 y_i가 입력 차분 x에 의해 발생할 수 있는 출력 차분 중 하나일 때, $\Delta I_{ac}^{10}(1) = ((*,0,0,0), (x,0,0,*), (y_1,0,*,y_2), (y_3,*,0,y_2))$이 고, $\Delta I_{ac}^{10}(2) = ((*,0,0,0), (x,0,0,*), (y_4,0,*,y_2), (y_5,*,0,y_2))$이다. 여기서, $\Delta I_{ac}^{10}(1)$와 $\Delta I_{ac}^{10}(2)$는 모두 ΔI_{ac}^{10}의 후보가 될 수 있다. 검사를 통과하는 암호문 quartet $(C_a^{j,l_0}, C_b^{j,l_0}, C_c^{j,l_1}, C_d^{j,l_1})$을 해쉬 테이블에 유지하여 단계 5에서 이용한다. $\Delta I_{ac}^{10}(1)$은 2^{128}개의 후보에서 2^{53}개의 값이 검사를 통과하고, $\Delta I_{ac}^{10}(2)$는 2^{128}개의 후보에서 2^{46}개의 값이 검사를 통과한다. 그러므로 검사를 통과하는 암호문 quartet의 수의 기댓값은 $2^{(52+64)\cdot 2} \cdot 2^{-128+53} \cdot 2^{-128+46} = 2^{75}$이다 (각각의 71-비트 $(k_a^w, \Delta\widetilde{K_{ac}})$의 추측에 대해, 단계 ④-㉢에서는 2^{118} 메모리 접근이 필요하다. 이때 필요한 시간은 2^{111} 암호화 연산과 동일하다.

4.2.3 E^f의 분석 단계

⑤ 라운드 9에서 라운드 키의 12번째 바이트인 8-비트 값 $k_a^{9,v}$를 추측하고, $k_b^{9,v} = k_c^{9,v} = k_d^{9,v} = k_a^{9,v}$으로 정한다. 8-비트 라운드 키 quartet $(k_a^{9,v}, k_b^{9,v}, k_c^{9,v}, k_d^{9,v})$에 대해 다음을 수행한다.

㉠ 남아 있는 모든 암호문 quartet $(C_a^{j,l_0}, C_b^{j,l_0}, C_c^{j,l_1}, C_d^{j,l_1})$에 대해, C_a^{j,l_0}와 C_c^{j,l_1}을 각각 $k_a^{9,v}$와 $k_c^{9,v}$로 E^f를 부분 복호화한다. 이 때, 부분 복호화한 값의 차분이 x가 아니면 이에 대응하는 quartet을 버린다. 이 과정은 대략 7-비트 필터링에 해당하므로 이 단계를 지나면 2^{68}개의 암호문 quartet이 남는다 (부분 복호화는 $(C_a^{j,l_0}, C_c^{j,l_1})$의 12 바이트를 이용하여 정렬한 후, 남은 암호문 quartet에 대하여 수행하거나, 사전 계산 테이블을 이용할 수 있기 때문에 단계 ⑤-㉠은 이전 단계에 비해 상대적으로 적은 시간 복잡도를 필요로 한다).

㉡ 남은 암호문 quartet에 대해, C_b^{j,l_0}와 C_d^{j,l_1}을 각각 $k_b^{9,v}$와 $k_d^{9,v}$로 E^f를 부분 복호화 한다. 단계 ⑤-㉠과 유사하게 부분 복호화 한 값의 차분이 x가 아닌 경우, 이에 대응하는 암호문 quartet을 버린다. 그러므로 이 단계를 지나면 2^{61}개의 암호문 quartet이 남게 된다 (단계 ⑤-㉡ 역시 상대적으로 적은 시간 복잡도를 필요로 한다).

⑥ 라운드 9에서 라운드 키의 8번째 바이트인 8-비트 값 $k_a^{9,u}$를 추측하고, $k_c^{9,u} = k_a^{9,u}$으로 정한다. 8-비트 라운드 키쌍 $(k_a^{9,u}, k_c^{9,u})$에 대해 다음을 수행한다.

㉠ 남아있는 모든 암호문 quartet $(C_a^{j,l_0}, C_b^{j,l_0}, C_c^{j,l_1}, C_d^{j,l_1})$에 대해, C_a^{j,l_0}와 C_c^{j,l_1}를 각각 $k_a^{9,u}$와 $k_c^{9,u}$로 E^f를 부분 복호화한다. 이때, 부분 복호화한 값의 차분이 x가 아닌 경우, 이에 대응하는 암호문 quartet을 버린다. 이 단계를 지나면 2^{54}개의 암호문 quartet이 남는다.

㉡ 8-비트 값 d를 추측하고, 8-비트 라운드 키쌍 $(k_b^{9,u} = k_a^{9,u} \oplus d, k_d^{9,u} = k_c^{9,u} \oplus d)$를 계산한다. 그리고 8-비트 라운드 키의 쌍 $(k_b^{9,u}, k_d^{9,u})$에 대해 다음을 수행한다.

ⓐ 남아있는 모든 암호문 quartet $(C_a^{j,l_0}, C_b^{j,l_0}, C_c^{j,l_1}, C_d^{j,l_1})$에 대해, C_b^{j,l_0}와 C_d^{j,l_1}를 각각 $k_b^{9,u}$와 $k_d^{9,u}$를 이용하여 E^f를 부분 복호화한다. 이때, 부분 복호화한 값의 차분이 x가 아닌 경우, 이에 대응하는 암호문 quartet을 버린다. 이 단계를 지나면 2^{47}개의 암호문 quartet이 남는다 (단계 ⑤와 유사하게 단계 ⑥은 효율적으로 수행된다).

⑦ 9 라운드 키의 0, 7, 10, 13번째 바이트인 32-비트 값 $k_a^{9,t}$를 추측하고, $k_c^{9,t} = k_a^{9,t} \oplus (x,0,0,0)$을 계산한다. 8-비트 라운드 키의 쌍 $(k_a^{9,u}, k_c^{9,u})$에 대해 다음을 수행한다.

㉠ 남아있는 모든 암호문 quartet ($C_a^{i,l_0}, C_b^{i,l_0}, C_c^{i,l_1}, C_d^{i,l_1}$)에 대해, C_a^{i,l_0}와 C_c^{i,l_1}를 각각 $k_a^{9,t}$와 $k_c^{9,t}$로 E^f를 부분 복호화한다. 이때, 부분 복호화한 값의 차분이 B에 속하지 않으면, 이에 대응하는 암호문 quartet을 버린다. B는 총 2^{32}개의 값에서 2^7-1개로 이루어지므로, 이 단계를 지나면 약 15-비트 필터링 이 되어 2^{22}개의 암호문 quartet이 남는다 (각각의 127-비트 값 $(k_a^{9,t}, d, k_a^{9,u}, k_a^{9,v}, k_a^w, \Delta\widetilde{K}_{ac})$의 추측에 대해, 단계 ⑦-㉠은 $2^{47+1}\cdot(4/16)\cdot(1/10)=2^{42.7}$ 암호화 연산을 필요로 한다).

㉡ 두 개의 8-비트 값 e, f를 추측하고, $(k_b^{9,t}=k_a^{9,t}\oplus(d,0,e,f), k_d^{9,t}=k_a^{9,t}\oplus(d\oplus x,0,e,f))$를 계산한다. 여기서 d는 단계 ⑥-㉡에서 추측한 8-비트 값이다. 32-비트 라운드 키의 쌍 $(k_b^{9,t}, k_d^{9,t})$에 대해 다음을 수행한다.

ⓐ 남아있는 모든 암호문 quartet ($C_a^{i,l_0}, C_b^{i,l_0}, C_c^{i,l_1}, C_d^{i,l_1}$)에 대해, C_b^{i,l_0}와 C_d^{i,l_1}를 각각 $k_b^{9,t}$와 $k_b^{9,t}$를 이용하여 E^f를 부분 복호화한다. 이때, 부분 복호화한 값의 차분이 B에 속하지 않으면, 이에 대응하는 암호문 quartet을 버린다. 이 과정은 25-비트 필터링에 해당하므로, 이 단계를 지나면 각각의 잘못된 키의 추측에 대해 2^{-3}개의 암호문 quartet이 남는다 (각각의 143-비트 값 $(e, f, k_a^{9,t}, d, k_a^{9,u}, k_a^{9,v}, k_a^w, \Delta\widetilde{K}_{ac})$의 추측에 대해, 단계 ⑦-㉡-ⓐ은 $2^{22+1}\cdot(4/16)\cdot(1/10)=2^{17.7}$ 암호화 연산을 필요로 한다).

⑧ 남아있는 모든 암호문 quartet ($C_a^{i,l_0}, C_b^{i,l_0}, C_c^{i,l_1}, C_d^{i,l_1}$)을, C_a^{i,l_0}와 C_c^{i,l_1}의 11번째 바이트의 차분에 따라 분류한다. 가장 많은 수를 가지는 그룹을 제외한 모든 암호문 quartet을 버린다. 이 과정은 7-비트 필터링에 해당하므로 각각의 잘못된 키의 추측에 대해, 이 과정을 거친 후 남게 되는 암호문 quartet의 수는 2^{-10}이다.

4.2.4 올바른 키 찾기 단계

⑨ 테이블에 하나 이상의 암호문 quartet이 남아있다면,[3] 이때 추측한 키를 옳은 키 후보로 결정한다. 남은 키 비트는 두 쌍의 평문·암호문을 이용한 전수조사를 통해 찾는다. 만약, 이 과정을 통과하는 키 후보가 존재하면 이 키를 실제 비밀키로 결정한다.

단계 ①, ②, ③에서 약 2^{124}개의 선택 평문이 필요하므로, 공격에 필요한 데이터 복잡도는 2^{124} 연관키 선택 평문이 필요하다. 단계 ④는 2^{71}번 수행되므로 단계 ④-㉡의 시간 복잡도는 $2^{60.7+71}=2^{131.7}$ 암호화 연산이다. 그리고 단계 ④-㉡는 $2^{111+71}=2^{182}$의 암호화 연산이 필요하다. 위에서 언급한 바와 같이 단계 ⑤, ⑥, ⑧의 시간 복잡도는 상대적으로 적다.

단계 ⑦의 시간 복잡도는 추측하는 라운드 키의 개수에 따라 반복하는 횟수에 의존한다. 단계 ⑦-㉠과 ⑦-㉡는 각각 2^{127}회, 2^{143}회 수행되므로, $2^{169.7}$과 $2^{160.7}$의 암호화 연산이 필요하다. 그러나 이 단계의 연산은 분할 정복 방법을 통해 더 효율적으로 계산이 가능하다. 단계 ⑦-㉠에서는 남아 있는 암호문 quartet의 4 바이트 중 2 바이트를 먼저 복호화하고, 이 차분이 B에 속하지 않는 quartet을 버리고 나머지 2개의 바이트에 대해 각각의 바이트에 대해서 동일한 검사를 수행한다. 이러한 방법을 단계 ⑦-㉡에도 적용할 수 있다. 이를 통해 단계 ⑦-㉠과 단계 ⑦-㉡의 시간 복잡도를 각각 $2^{132.7}$과 $2^{143.7}$의 암호화 연산으로 줄일 수 있다.

이 공격의 성공 확률을 계산하기 위해서 포와송 분포를 이용한다. 잘못 추측된 키에 대해서 남겨지는 암호문 quartet의 기댓값은 2^{-10}이므로, 포와송 분포에 의해 각각의 잘못된 추측된 키에 대해서 하나 이상의 암호문 quartet이 남겨질 확률은 2^{-10}이다.

$X\sim Poi(\lambda=2^{-10})$, $P_X(X\geq 1)\approx 2^{-10}$.

그러므로 단계 ⑨에서 검사할 잘못 추측된 키 quartet의 개수의 기댓값은 $2^{143-10}=2^{133}$이고, 단계 ⑨의 시간 복잡도는 약 2^{182} 암호화 연산이다.

올바른 키 quartet에 대해서 남겨지는 올바른 암호문 quartet의 기댓값은 8 라운드 렉탱글 distinguisher에 의해 $2=2^{232}\cdot 2^{-231}$이다. 그러므로 올바른 키 quartet에 대해서 1개 이상의 암호문 quartet이 남겨질 확률은 포와송 분포에 의해 0.86이다, $Y\sim Poi(\lambda=2)$, $\Pr_Y(Y\geq 1)\approx 0.86$. 그러므로 본 공격은 2^{124}의 연관키 선택 평문을 이용하여 2^{183}의 암호화 연산을 통하여 성공 확률 0.86으로 올바른 키를 복구할 수 있다.

3) 본 논문에서는 threshold를 1 개의 암호문 quartet으로 사용함으로써 [9]의 결과를 향상시켰다 ([9]에서는 threshold를 16 개의 암호문 quartet으로 사용하였다).

4.3 AES-192에 대한 64개의 연관키를 이용하는 연관키 렉탱글 공격

본 공격에 사용하는 연관키를 더욱 정교하게 선택하면, 공격에 사용되는 연관키의 개수를 256개에서 64개로 줄일 수 있다. 다음과 같은 64개의 연관키를 이용한다.[4)]

- o 16개의 K_a의 후보 $K_a^i (i=0,1,\cdots,15)$는 3, 11번째 바이트를 제외한 모든 바이트가 동일하고, 3, 11번째 바이트는 동일한 값으로 각각 $s_0, s_1, \cdots, s_{15}$이다. 여기서 s_i는 쌍마다 다른 값을 가진다.
- o 16개의 K_b의 후보 $K_b^i (i=0,1,\cdots,15)$는 $K_a^i \oplus K_b^i$의 1, 9번째 바이트는 x를 이고, 이외의 바이트는 0을 만족한다.
- o 16개의 K_c의 후보 $K_c^j (j=0,1,\cdots,15)$는 3, 11번째 바이트를 제외한 모든 바이트가 동일하고, 3, 11번째 바이트는 동일한 값으로 각각 $t_0, t_1, \cdots, t_{15}$이다. 여기서 t_j는 쌍마다 다른 값을 가진다. 그리고 $K_c^j \oplus K_d^j$의 8, 12번째 바이트는 x이다.
- o 16개의 K_d의 후보 $K_d^j (j=0,1,\cdots,15)$는 $K_c^j \oplus K_d^j$의 1, 9번째 바이트는 x이고, 이외의 바이트는 0을 만족한다.

이러한 관계를 가지는 64개의 연관키를 이용하여 256개의 연관키 quartet $(K_a^i, K_b^i, K_c^j, K_d^j)$을 생성할 수 있다. 그러면 이 중에 하나는 $K_a^i \oplus K_b^i = K_c^j \oplus K_d^j = \Delta K_{ab}$와 $K_a^i \oplus K_c^j = K_b^i \oplus K_d^j = \Delta K_{ac}$를 만족한다.

위의 조건을 만족하는 64개의 연관키를 이용하면 각각의 키에 대해 2^{116}개의 선택 평문을 이용하므로 2^{122}의 연관키 선택 평문이 필요하다. 또한, 단계 ④-㉢에서 256개의 키 quartet에 대해 반복해야 하므로 2^{183} 암호화 연산이 필요하고, 단계 ⑨에서 2^{182}의 암호화 연산이 필요하므로 $2^{183}+2^{182}=2^{183.6}$의 암호화 연산으로 키를 복구할 수 있다.

V. 결 론

본 논문에서는 연관키 렉탱글 공격 기법이 AES-192의 총 12 라운드 중 10 라운드에 적용될 수 있음을 보였다. 256개의 연관키를 사용하는 첫 10 라운드 축소 AES-192는 2^{124}의 데이터 복잡도와 2^{183}의 시간 복잡도로, 64개의 연관키를 사용하는 첫 10 라운드 축소 AES-192는 2^{122}의 데이터 복잡도와 $2^{183.6}$의 시간 복잡도로 공격된다. 본 논문의 공격은 AES-192에 대한 기존 공격 중 최상의 공격이다.

4) 연관키의 개수를 줄이는 방법은 [2,9]에서 소개되었다.

참 고 문 헌

[1] J. Daemen and V. Rijmen, "AES proposal : Rijndael," http://csrc.nist.gov/encryption/aes/round2/conf3/aes3paper.html

[2] E. Biham, O. Dunkelman, and N. Keller, "Related-key boomerang and rectangle attacks," EUROCRYPT'05, LNCS 3494, pp. 507-525, 2005.

[3] W. Zhang, W. Wu, and D. Feng, "New results on impossible differential cryptanalysis of reduced AES," ICISC'07, LNCS 4817, pp. 239-250, 2007.

[4] S. Lucks, "Attacking seven rounds of Rijndael under 192-bit and 256-bit keys," Proceedings of AES 3, NIST, 2000.

[5] G. Jakimoski and Y. Desmedt, "Related-key differential cryptanalysis of 192-bit key AES variants," SAC'03, LNCS 3006, pp. 208-221, 2004.

[6] E. Biham, O. Dunkelman, and N. Keller, "Related-key impossible differential attacks on AES-192," CT-RSA'06, LNCS 3860, pp. 21-31, 2006.

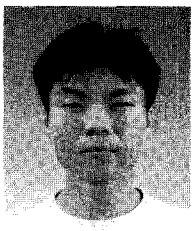
[7] S. Hong, J. Kim, S. Lee, and B. Preneel, "Related-key rectangle attacks on reduced versions of SHACAL-1 and AES-192," FSE 2005, LNCS 3557, pp. 368-383, 2005.

[8] A. Biryukov, "The boomerang attack on 5 and 6-round AES," AES, LNCS 3373, pp. 11-16, 2005.

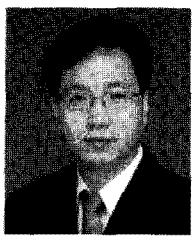
[9] J. Kim, S. Hong, and B. Preneel, "Related-key rectangle attacks on reduced AES-192 and AES-256," FSE 2007, LNCS 4593, pp. 225-241, 2007.

[10] J. Kim, G. Kim, S. Hong, S. Lee, and D. Hong, "The related-key rectangle attack - application to SHACAL-1," ACISP 2004, LNCS 3108, pp. 123-136, 2004.
[11] E. Biham, "New types of cryptanalytic attack using related keys," Journal of Cryptology, Vol. 7, No. 4, pp. 156-171, Dec. 1994.
[12] E. Biham and N. Keller, "Cryptanalysis of reduced variants of Rijndael," http://csrc.nist.gov/encryption/aes/round2/conf3/aes3paper.html
[13] E. Biham and A. Shamir, "Differential cryptanalysis of DES-like cryptosystems," Advances in Cryptology - CRYPTO'90, LNCS 537, pp. 2-21, 1990.
[14] J.H. Cheon, M.J. Kim, K. Kim, J.Y. Lee, and S.W. Kang, "Improved impossible differential cryptanalysis of Rijndael and Crypton," ICISC 2001, LNCS 2288, pp. 121-143, 2002.
[15] N. Ferguson, J. Kelsey, S. Lucks, B. Schneier, M. Stay, D. Wagner, and D. Whiting, "Improved cryptanalysis of Rijndael," FSE 2000, LNCS 1978, pp. 213-230, 2001.

〈著者紹介〉



김 종 성 (Jongsung Kim) 정회원
2000년 8월: 고려대학교 수학과 학사
2002년 8월: 고려대학교 수학과 석사
2006년 11월: K.U.Leuven, ESAT/SCD-COSIC 박사
2007년 2월: 고려대학교 정보보호대학원 박사
2007년 3월~현재: 고려대학교 정보경영공학전문대학원 연구교수
〈관심분야〉 대칭키 암호의 분석 및 설계, 멀티미디어/유비쿼터스 정보보호, 부채널 공격



홍 석 희 (Seokhie Hong) 종신회원
1995년 2월: 고려대학교 수학과 학사
1997년 2월: 고려대학교 수학과 석사
2001년 2월: 고려대학교 수학과 박사
1999년 8월~2004년 2월: (주) 시큐리티 테크놀로지스 선임연구원
2003년 2월~2004년 2월: 고려대학교 정보보호기술연구센터 선임연구원
2004년 4월~2005년 2월: K.U.Leuven, ESAT/SCD-COSIC 박사후연구원
2005년 3월~2008년 8월: 고려대학교 정보보호대학원 조교수
2008년 9월~현재: 고려대학교 정보경영공학전문대학원 부교수
〈관심분야〉 대칭키 암호의 분석 및 설계, 컴퓨터 포렌식

이 창 훈 (Changhoon Lee) 정회원
2001년 2월: 한양대학교 수학과 학사
2003년 2월: 고려대학교 정보보호대학원 석사
2008년 2월: 고려대학교 정보경영공학전문대학원 박사
2008년 4월~2009년 2월: 고려대학교 정보경영공학전문대학원 연구교수
2009년 3월~현재: 한신대학교 컴퓨터공학부 전임강사
〈관심분야〉 정보보호, 암호 분석 및 설계, 멀티미디어 보안, 게임