Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.30, NO.1, Feb. 2020 ISSN 2288-2715(Online)
https://doi.org/10.13089/JKIISC.2020.30.1.1

마스킹 화이트 박스 AES에 대한 새로운 고차 차분 계산 분석 기법*

이 예 찬,[1†] 진 성 현,[1] 김 한 빛,[1] 김 희 석,[2] 홍 석 희[3‡]
[1,3]고려대학교 정보보호학과(대학원생, 교수), [2]고려대학교 사이버보안전공(교수)

New Higher-Order Differential Computation Analysis on Masked White-Box AES*

Yechan Lee,[1†] Sunghyun Jin,[1] Hanbit Kim,[1] HeeSeok Kim,[2] Seokhie Hong[3‡]
[1,3]Graduate School of Information Security and Institute of Cyber Security & Privacy (ICSP), Korea University(Graduate student, Professor)
[2]Department of Cyber Security, College of Science and Technology, Korea University(Professor)

요 약

화이트 박스 암호에 대한 부채널 분석 맥락의 공격인 차분 계산 분석(Differential computation analysis, DCA) 공격이 제안됨에 따라, 이에 대응하기 위해 Lee 등의 대응기법과 같이 테이블 인코딩 기반 마스킹 화이트 박스 암호가 제안되었다. 마스킹 화이트 박스 암호에 대한 기존 고차 DCA는 테이블 인코딩 기반의 마스킹 구현 구조를 고려하지 못하여 Lee 등이 제안한 대응 기술에는 적용이 불가능하였다. 본 논문에서는 테이블 인코딩 기반 마스킹 구현에도 적용할 수 있는 새로운 고차 DCA 기법을 제안하고, Lee 등이 제안한 마스킹 화이트 박스 암호의 비밀키 정보를 실제로 찾음으로써 그 유효성을 증명하였다.

ABSTRACT

As differential computation analysis attack(DCA) which is context of side-channel analysis on white-box cryptography is proposed, masking white-box cryptography based on table encoding has been proposed by Lee et al. to counter DCA. Existing higher-order DCA for the masked white box cryptography did not consider the masking implementation structure based on table encoding, so it is impossible to apply this attack on the countermeasure suggested by Lee et al. In this paper, we propose a new higher-order DCA method that can be applied to the implementation of masking based on table encoding, and prove its effectiveness by finding secret key information of masking white-box cryptography suggested by Lee et al. in practice.

Keywords: White-Box Cryptography, Differential Computation Analysis, Masked White-Box AES, 2nd Order DPA

Received(08. 30. 2019), Modified(10. 31. 2019), Accepted(11. 24. 2019)

* 본 연구는 고려대 암호기술 특화연구센터(UD170109ED)를 통한 방위사업청과 국방과학연구소의 연구비 지원으로 수행되었습니다.

† 주저자, younggo@korea.ac.kr
‡ 교신저자, shhong@korea.ac.kr(Corresponding author)

I. 서 론

최근 스마트 네트워크에서 제공되는 DRM, 클라우드 등의 서비스들은 서비스의 사용자 본인이 접근할 수 없는 보안 영역을 요구한다. 화이트 박스 환경은 사용자가 자신의 디바이스를 공격하는 가정으로, 암호 연산의 중간 단계에 접근하여 모든 정보를 습득할 수 있음을 의미한다. Chow 등은 이러한 가정에 안전한 암호 설계를 위해, 연산 구조를 감추는 입출력 인코딩으로 내부 정보가 보호된 테이블을 생성하여, 해당 테이블을 참조하는 연산만으로 암호 연산을 수행하는 화이트 박스 암호를 제안하였다[2,6]. 이러한 구조는 BGE 공격[7] 등과 같은 대수적 분석을 통해 테이블의 비선형 연산을 복원하려는 시도와 더불어 Dinur의 아핀 인코딩을 복원하는 효율적인 알고리즘의 공격 대상이 되기도 하였다[8]. 통상적으로 화이트 박스 암호 분석은 아핀 복원 공격의 복잡도를 낮추는 방법에 집중되어 있었으므로, Baek의 구현[9]처럼 아핀 인코딩의 복잡도를 높이는 대응기술이 제안되기도 하였다. Chow의 구현을 기반으로 하는 대부분의 화이트 박스 암호를 CEJO framework로 분류하며, 이들 대부분은 테이블 설계의 세부적인 구조를 감춘 채 공격 복잡도를 높이는 방향으로 제안되었다.

Kocher와 Brier가 각각 제안한 DPA(Differential Power Analysis[1])와 CPA(Correlation Power Analysis [5]) 공격 기법은 블랙박스 암호의 중간 연산 정보를 활용하여 실질적으로 암호에 강력한 위협을 가하는 공격 기법이다. Bos는 이러한 부채널 분석 공격을 화이트 박스 암호에 적용하는 방법을 제안하였다. 차분 계산 분석(DCA, Differential Computational Analysis[3]) 공격은 암호 연산 테이블의 인코딩 구조를 파악하지 않고도 화이트 박스 암호를 무력화하는 공격 기법으로, 화이트 박스 부채널 분석의 초석이 되었다. 기존의 부채널 분석은 그레이 박스 환경으로 일컬으며, 공격자가 획득한 전력 소비량 등의 정보를 모델링하여 암호 연산과의 상관성을 분석한다. 화이트 박스 환경에서 수행하는 DCA는 이 점을 보다 구체적으로 적용하여 인코딩이 포함된 테이블에 대한 통계적 분석을 가능하게 하였다. Bos의 DCA 공격은 부채널 분석을 기반으로 하기 때문에 선형 인코딩의 영향을 받지 않는다. 따라서 기존의 구현은 DCA에 대한 화이트 박스 암호 안전성을 높일 수 없다.

DCA의 공격 대상인, 실제 암호의 연산값과 화이트 박스 정보 사이의 연관성을 제거하기 위해 암호 연산 중간값에 난수를 덧씌우는 마스킹 대응기술을 통해 부채널 분석 안전성을 제공하여야 한다. Lee 등은 CEJO framework를 기반으로 화이트 박스 환경에 적합하도록 부채널 대응기술을 적용한 마스킹 화이트 박스 AES를 제안하였다[4]. 그는 난수 발생기를 사용할 수 없는 화이트 박스 환경에 적합하도록, 테이블 설계에 마스킹 과정을 삽입하여 난수 정보가 직접적으로 드러나지 않도록 하였다. 마스킹이 적용된 테이블 참조 연산 정보는 DCA 공격자가 예측한 연산 중간값과의 연관성이 떨어지게 되었고, Lee의 구현은 화이트 박스 환경에서도 안전하게 마스킹을 적용할 수 있었다.

본 논문에서는 Lee등이 제안한 마스킹 화이트 박스 암호에 대한 새로운 고차 차분 계산 분석을 제안한다. 고차 부채널 분석은 획득한 정보의 두 개 이상의 시점을 조합하여 본래 1차 정보로는 드러나지 않던 정보를 분석할 수 있도록 하는 공격이다. 이를 위해 공격자는 반드시 조합 대상이 되는 정보의 시점을 면밀히 파악할 수 있어야 한다. 올바른 시점 분석을 통해 조합된 정보를 2차 trace로 일컬으며, 이는 1차 부채널 분석에 대해 비밀 정보를 드러내게 된다. 본 논문은 Lee의 구현으로부터 수집한 화이트 박스 정보를 통해 테이블 참조 연산의 시점을 면밀히 파악하고, 구체적으로 분석된 테이블 출력을 조합하여 생성한 2차 trace를 분석해 마스킹 화이트 박스 AES를 무력화하는 공격 과정을 서술한다.

실제로 Bogdanov 등은 DCA 공격에 대해 화이트 박스 암호가 취하게 될 마스킹 대응기술을 예측하였고, 이에 대한 고차 DCA 공격이 수행될 방향성에 대해 이론적으로 제시하였다[11]. 그러나 그들은 인코딩이 적용되지 않은 테이블 외부의 마스크만을 고려하였고, 화이트 박스 환경에 안전한 난수 생성기를 사용하여 마스킹과 셔플링을 구현하여야 한다는 조건을 제안하였다. 따라서 해당 기법은 이와 같은 방식의 마스킹을 사용하지 않는 Lee의 테이블 인코딩 기반 마스킹 구현으로부터 비밀 정보를 드러내진 못했다. 구현의 기반이 되는 CEJO framework는 선형, 혹은 아핀의 인코딩과 비선형 인코딩으로 구성된다. Lee의 주장에 따라, 이들 중 비선형 인코딩에 의해 테이블 출력이 가지는 연산 정보의 특성이 달라지게 된다[4]. 따라서 본 논문에서는 고차 분석 기법의 성능을 향상하여 실질적으로 모든 비밀 정보를

드러내는 방안을 제시한다. 이를 위하여 비선형 인코딩을 약화하는 조합 방법을 적용하여 중간 정보의 비트 특성을 강하게 드러내는 방법을 제안한다.

마지막으로, 실제 구현된 마스킹 화이트 박스 암호에 대해 해당 공격을 수행한 결과를 보일 것이다. 공격 대상은 GitHub에 게시된 Chow, Lee의 구현이며, 분석 도구 또한 GitHub에 공개된 도구를 공격 수행에 적절하도록 다소 개선하여 사용하였다. 이 과정을 통해, 본 논문에서 제안하는 공격 기법이 보편적인 환경에서 수행 가능함을 확인할 수 있다.

II. 관련 연구

2.1 CEJO framework

화이트 박스 환경에서 안전하게 동작하는 암호를 구현하기 위해, Chow 등은 CEJO framework로 명명되는 암호 설계 기법을 제안하였다. 대표적인 표준 블록 암호인 DES, AES에 대해 모든 암호화 연산을 테이블 참조만으로 수행하는 구현 기법을 제시하였다[2]. 기존 AES는 T-table이라고 불리는, 연산 속도의 최적화를 위해 대부분의 라운드 연산을 단일 테이블 참조만으로 수행하는 구현 기법이 존재한다. Chow의 구현은 해당 최적화 테이블에 이전 라운드의 AddRoundKey를 포함하고 입력과 출력에 각각 랜덤한 인코딩 연산을 덧붙여 공격자가 테이블 연산의 중간 정보를 예측할 수 없도록 하였다. 더불어 암호화 키를 다루는 연산을 해당 테이블 연산 내에 포함하는 방식으로 가장 민감한 정보가 연산 과정에 직접 드러나지 않도록 하였다. 이러한 구현은 암호 알고리즘에서 수행되는 모든 연산에 대해, 모든 입력을 수용할 수 있는 테이블이 각각 설계되어야 한다. 따라서 CEJO framework는 기존 암호에 비해 큰 사전 연산 테이블 공간을 요구하였다.

Chow가 Type II 라고 명명한, AES의 라운드 연산을 담당하는 테이블은 Fig 1과 같은 형태로 구성된다. Type II 테이블 내에 포함된 $T^{r}_{i,j}$은 8비트의 입출력을 가지며 AES 라운드 연산 중, 한 라운드 이전의 라운드 키와의 AddRoundKey 연산과 해당 라운드에서의 SubByte 연산을 수행한다. 이후로 위치하는 T_{y_i}는 8비트의 입력과 32비트의 출력을 가지며, 8비트 입력에 대한 MixColumn 연산을 열 단위로 수행한다. NL은 4비트 입출력의 비선형

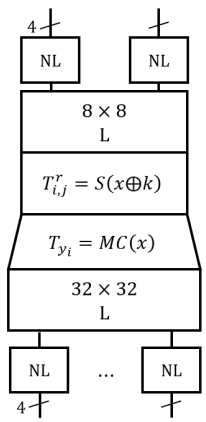


Fig. 1. Type II table on Chow`s implementation[2]

인코딩 연산을, 8×8과 32×32의 두 종류의 L은 각각 8비트와 32비트의 입출력을 가지는 선형 인코딩 연산을 수행한다. 이 구현에서 알고리즘의 구조가 공개되지 않는 연산이 바로 인코딩 연산에 해당하는 NL과 L이며, 각각 블록 암호의 기본 구조인 혼돈과 확산의 기능을 수행한다. 테이블 입출력에 추가한 인코딩으로 인해, 화이트 박스 공격자는 테이블 연산의 입출력 정보를 얻더라도 실제로 암호 알고리즘이 수행한 라운드 함수의 입출력 정보를 얻을 수 없다.

2.2 차분 계산 분석

부채널 분석 공격자들은 일반적으로 공개된 암호 알고리즘의 구조와 입력 평문, 출력 암호문 외의 부가적인 정보를 통해 통계적으로 비밀 정보를 추측할 수 있다. DPA[1]는 공격자가 수집한 부가 정보와 연산 중간값 사이의 관계성을 통해 알려진 암호 알고리즘의 비밀 키 값을 효율적으로 추측하게 하였다. AES에 대한 DPA 공격은 8비트 단위로 0라운드 마스터 키를 추측할 수 있으며, 통계적으로 연산 된 부채널 정보는 올바른 키 추측에서만 그 특성을 드러낸다. 이와 같은 공격 방식은 본래 기존의 경량 암호 장비, 경량 스마트 통신 장비 등에서 위험성을 드러낸 바 있다.

화이트 박스 암호 환경에서도 그레이 박스와 유사한, 하지만 훨씬 정확한 형태의 중간 정보를 획득할 수 있다[3]. 소프트웨어 프로그래머는 자신이 개발한 프로그램이 정상적으로 작동하는지, 메모리 누수 혹은 주소 충돌 등의 현상이 일어나지 않는지 점검하기 위해 바이너리 분석 수집 도구를 사용한다. 바이

너리 수집 도구는 특정 소프트웨어가 실행되는 동안 접근하는 모든 주소, 주소에서 연산되는 데이터, 연산을 수행하는 연산자 등의 모든 인스턴스 정보를 바이너리 형태로 수집할 수 있으며, 그 종류로는 Valgrind와 Intel 사에서 배포한 PIN이 있다. 이와 같은 도구들이 수집하는 소프트웨어 정보는 부채널 분석에 사용하기에 매우 적합한 요소와 형태를 지니게 된다.

Fig 2는 바이너리 수집 도구 Valgrind를 통해 Chow가 구현한 화이트 박스 AES 연산으로부터 주소 정보를 수집하여 시각화한 것이다. 가로축은 암호의 연산 시간, 세로축은 메모리 주소값의 최하위 1바이트 값을 나타내며, 노이즈 없이 깔끔한 형태의 연산 패턴이 드러남을 확인할 수 있다. 화이트 박스의 정보는 특성상 분석자가 이해하기에 수월한 형태로 이루어져 있다. Valgrind는 프로그램의 연산 수행 정보를 연산자, 메모리의 주소와 데이터 등으로 구분지어 수집하므로, 이 정보들을 분류하여 각각 필요한 공격 프로세스에 어렵지 않게 적용할 수 있다.

Bos 등은 수집한 바이너리의 메모리 데이터 정보를 통해 수행하는 DCA(Differential Computation Analysis, 차분 계산 분석)를 제안하였다[3]. Valgrind 등의 바이너리 분석 도구를 통해 수집한 암호 알고리즘 실행 정보는 그레이 박스 환경보다 월등히 정밀한, 중간 연산 정보의 실제 값을 드러낸다. 따라서 이를 통한 통계적 분석에는 부채널 분석에서 가정하는 전력 소비 모델이 요구되지 않으며, 중간 정보의 단일 비트 단위의 세부적인 분석이 가능하다. 정밀한 부채널 공격은 선형과 비선형의 인코딩을 지나더라도 잔류하게 되는 특정 비트의 연산 정보를 충분히 활용하여 화이트 박스 암호의 비밀 정보를 추측할 수 있게 하였다. 이를 실제 실험을 통해 보인 과정은 4장에서 포함하였다.

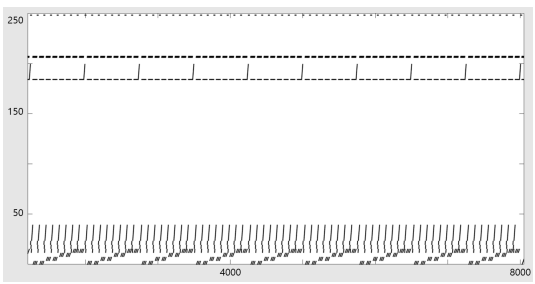


Fig. 2. Binary address trace on Chow`s implemenation

DCA 공격은 그레이 박스 부채널 분석 공격과 동일한 논리로, 올바른 키 추측을 통해 계산한 중간값과 수집한 바이너리 메모리 데이터 간의 상관계수가 높게 나타나는 시점을 도출할 수 있다. 이는 화이트 박스 암호에서 사용하는 테이블 크기를 최적화하기 위해 선택한 4비트 비선형 인코딩의 성질에 의해 발생하는 현상이다. DCA 분석은 테이블의 구조를 자세히 파악하고 있지 않더라도 화이트 박스 암호에서 테이블 연산 내부에 감춰진 비밀 정보를 복원해내는 것이 가능하다.

2.3 마스킹이 적용된 화이트 박스 AES

DCA 공격에 대응하기 위해, 암호 설계자는 부채널 분석 공격에 대해 적용하는 마스킹 대응기술을 고려하여야 하였다. 마스킹은 암호 연산 내부의 중간 정보에 마스크 난수를 덧씌워 공격자가 획득한 부가 정보가 통상적인 암호 알고리즘의 중간 정보와 연관을 가지지 않도록 하는 대응기술이다. 해당 연산을 위해 일반적인 그레이 박스 환경에서는 마스크 난수를 생성하여 중간 연산에 적용하고, 난수가 적용된 상태에서도 연산이 올바르게 수행될 수 있도록 사전 연산 테이블을 생성하여 기존의 Sbox를 대체한다.

화이트 박스 암호 환경에서 이를 실현하기 위해, 설계자는 이와 같은 대응기술이 화이트 박스 환경에서도 안전하게 적용되도록 구현 기법을 설계하여야 한다. 암호 알고리즘 연산 중간에 난수 정보를 덧씌우기 위해서는 모든 연산에 대해 난수 값을 생성하고

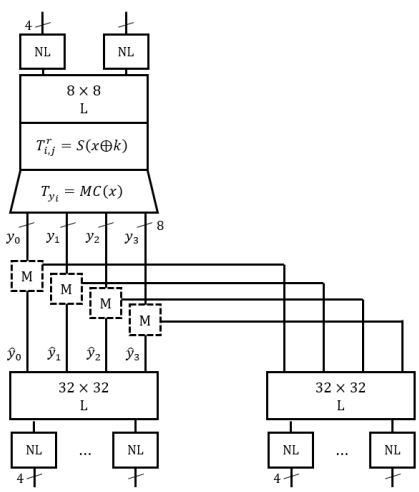


Fig. 3. Type II-M table of Lee`s Masked white-box AES [4]

계산하여야 한다. 그러나 화이트 박스 환경에서는 난수 생성기조차 그 실행 과정이 전부 드러난다. 더불어 화이트 박스 암호의 안전성은 암호화 연산을 알고리즘 내부에서 직접 수행하지 않고 사전에 생성된 테이블을 통해 처리하는 것으로 보장받는다. 따라서 연산 과정에서 새로운 테이블을 생성하는 연산을 수행한다면 기존 테이블의 중간 정보까지 노출될 가능성이 있으므로, 이 모든 과정을 화이트 박스 암호의 맥락과 동일하게 구현할 방법이 요구되었다.

Lee 등은 해당 마스킹 난수 정보를 테이블 설계에 포함하는 구현 방법을 고안하였다[4]. Chow의 화이트 박스 AES에서 가장 민감한 정보를 포함하고 있는 Type II 테이블에 마스킹을 포함하기 위해, Lee는 논문에서 다음과 같은 방법을 사용하였다. Fig 3의 Type II-M 테이블은 Chow의 Type II 테이블 중간에 M함수를 추가하여 구성된다. T_{y_i}의 32비트 출력에 적용되는 M함수는 각각 8비트의 입출력을 가지며, 독립적으로 랜덤한 마스크를 생성하여 T_{y_i} 연산 결과값에 XOR하는 연산과 랜덤 마스크 자체를 출력하는 연산을 수행한다. 이로 인해 Type II-M 테이블의 출력은 기존의 마스킹 대응기술과 마찬가지로 공격자가 예측하는 AES의 라운드 연산 결과에 무관한 난수로 구성된다. 이후 XOR 테이블만으로 마스크를 벗겨내기 위해, 생성한 마스크 난수 또한 테이블 출력으로 내보내야 한다. 이때, 마스킹이 적용된 중간값과 마스크 난수 모두 Chow의 제안과 같은 방식으로 선형과 비선형의 인코딩을 적용한다. 결과적으로 Type II-M 테이블은 8비트 입력에 64비트의 출력으로 구성되므로, 기존 Type II 테이블의 2배에 해당하는 공간을 요구하였다.

생성된 테이블에 대한 참조 연산만으로 암호화를 수행하는 화이트 박스 암호의 특성상 테이블 내부에 포함된 마스크 난수는 화이트 박스 공격자에게 분석되지 않을 것이다. 더불어 구현된 테이블의 출력 정보는 기존 부채널 대응 기법인 마스킹이 적용된 중간값이므로, Lee의 제안은 기존 부채널 분석 공격인 DCA에 안전하다고 할 수 있었다.

III. 제안하는 고차 차분 계산 분석

본 장에서는 화이트 박스 암호에 대해 수행하는 DCA에 고차 부채널 분석 공격 기법을 적용한 고차 DCA를 제안한다. 고차 부채널 분석이란 마스킹 대응기술이 적용된 암호에 대해, 획득한 부채널 정보의 특정 시점을 조합하는 과정을 추가하여 가려진 정보를 드러내는 분석 기법을 말한다. 이를 위해 공격자는 조합 대상이 되는 연산 시점을 세밀하게 파악할 수 있어야 하고, 이는 그레이 박스 환경에서 매우 고도의 기술을 요구한다. 부채널 분석 공격자들은 공격 대상이 되는 시점인 POI(Point of Interest)를 찾아내는 다양한 기술과 방법을 제시하였고, 실제로 2차 DPA, CPA는 1차 마스킹이 적용된 블록 암호에 대해 유효한 공격을 수행할 수 있었다. 따라서 본 장에서는 이들과 동일한 맥락의 2차 DCA 분석 기법을 제안하여 1차 마스킹이 적용된 Lee의 구현에 대한 공격이 수행됨을 보였다.

화이트 박스 환경의 특성상 고차 분석 기법이 더욱 강력하게 적용됨을 확인할 수 있다. 조합하여야 하는 공격 시점을 파악하는 과정은 그레이 박스에 비해 매우 직관적이며, 공격 성능을 높이기 위한 전후처리의 방식이나 효과가 월등히 높게 나타남을 확인할 수 있다. 공격 시점을 파악할 수 있는 단순 계산 분석(SCA, Simple Computation Analysis) 기법은 3.1절에서 소개하였으며, 파악한 시점의 연산 정보를 조합하여 공격을 수행하는 고차 분석 기법을 3.2절에서, 공격 성능을 향상하는 후처리 기법은 3.3절에서 소개하였다.

3.1 단순 계산 분석

본 절에서는 화이트 박스 공격자가 POI를 얼마나 자세하게 얻어낼 수 있는지에 대해 소개한다. 고차 부채널 분석 공격의 성능과 성공률에 가장 큰 영향을 미치는 POI는 주로 비밀값과 연관된 연산이 수행되는 시점, 특정 마스킹 값 혹은 중간 정보가 충돌하는 시점 등을 일컫는다. Lee의 알고리즘에서의 POI에 해당하는 시점은 Type II-M 테이블의 출력 데이터가 이에 해당하게 되고, 화이트 박스 환경에서는 SCA만으로도 이를 구체적으로 파악할 수 있었다.

2.2절에서 설명한 대로, 바이너리 분석 도구는 소프트웨어가 실행되는 동안 다루는 모든 데이터와 해당 데이터의 위치 정보인 바이너리 주소 정보를 수집할 수 있다. 화이트 박스 부채널 분석 공격자는 해당 바이너리 주소 정보를 이미지화하여 특정 시점에서 수행되는 연산의 종류를 정확히 파악할 수 있다. 바이너리 분석 도구를 통해 수집하는 연산 부채널 정보는 바이너리 trace로 명명하며, Fig 4는 Github에

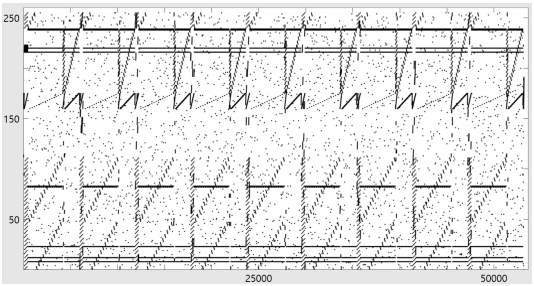


Fig. 4. Binary address trace of Lee`s case 1

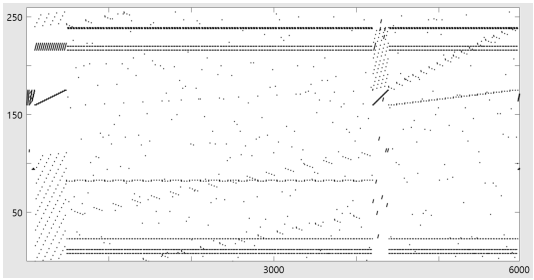


Fig. 5. Binary address trace on first round operation of Lee`s case 1

공개된 Lee의 마스킹 화이트 박스 AES에 대해 수집한 주소 바이너리 trace를 Fig 2와 같은 방법으로 시각화한 것이다.

Fig 4를 통해 보여지듯이, 화이트 박스 AES 연산으로 추측할 수 있는 9개의 서로 같은 패턴이 확연하게 드러난다. Chow의 구현에는 Type II 테이블의 연산 이후, MixColumn 연산을 수행하기 위해 인코딩된 중간값에 XOR 연산을 수행하게 해주는 XOR 테이블이 사용된다. Chow는 XOR 테이블을 Type IV 테이블로 명명하였으며, 한 번의 테이블 참조로 4비트 데이터 두 개의 XOR을 수행한다. 8비트 입력에 4비트 출력을 가지는 Type IV 테이블은 한 라운드에서만 192회 수행되므로 화이트 박스 AES에서 가장 많은 테이블 참조 연산을 차지한다. 그리고 MixColumn 연산을 수행하지 않는 최종 라운드는 이 연산을 포함하고 있지 않다. 따라서 공격자는 해당 9개의 패턴이 최종 라운드를 제외한 9개의 라운드 연산임을 추측할 수 있다. 주소 바이너리 trace는 연산을 수행하는 데이터의 주소값을 가진다. 따라서 특정 위치의 데이터를 반복적으로 갱신할 경우, 혹은 배열의 형태로 된 데이터를 다루는 연산을 수행할 경우 그 패턴이 명확하게 드러나게 된다. 화이트 박스 암호 구현의 특성상 테이블 참조 연산만으로 암호화를 수행해야 하므로, 그로 인해 발생하는 주소 바이너리 trace의 패턴을 지우는 일은 매우 어렵다.

Fig 5는 Fig 4의 유사한 9개 패턴 중 가장 앞에 보이는 패턴, 즉, 1라운드 연산 부분을 확대한 것이다. 라운드 연산 내부에서도 특정한 패턴들이 반복되는 것을 확인할 수 있으며, 고차 부채널 분석 공격자는 이를 통해 공격 대상을 더욱 좁혀 부분 파형을 획득할 수 있다. 본 논문에서는 대상 알고리즘의 설계를 알고 있으므로 테이블 연산을 보다 구체적으로 파악할 수 있었으나, 정확한 구현 원리가 파악되지 않는다면 1라운드 전체의 정보를 사용하여 같은 방법의 전처리를 수행할 수 있다.

Lee의 구현에서는 Type II-M 테이블 연산 이후 해당 테이블 정보를 조합하는 Type IV-II 테이블 연산을 수행한다. 이후 Chow의 화이트 박스 구현과 마찬가지로, 다음 라운드의 선형 입력 디코딩과 현 라운드의 선형 출력 인코딩을 맞추기 위한 Type III 인코딩 변환 테이블 연산을 수행하고, 최종적으로 Type III의 테이블 출력을 조합하는 Type IV-III XOR 테이블 연산을 수행한다. 이때, Type II-M 출력에 대한 XOR 연산은 32비트의 라운드 연산 결과에 대한 XOR에 더해 32비트의 마스크 값과의 XOR 연산을 추가로 수행해야 하므로 Type III 테이블 출력보다 더 많은 수의 Type II-M 테이블 참조와 XOR 연산을 요구한다. Fig 5는 두 구간의 연산이 명백하게 구분되는 길이 차이를 보이므로, 중앙의 가장 큰 범위를 차지하는 연산을 Type IV-II, 오른쪽에 존재하는 두 번째로 큰 범위를 차지하는 연산은 Type IV-III으로 특정 지을 수 있다. 또한, 그림의 가장 왼쪽에 존재하는 짧은 패턴과 두 Type IV 연산 사이에 존재하는 짧은 패턴을 각각 Type II-M, Type III 테이블 연산으로 특정지어, 매우 정교하게 연산 시점을 특정할 수 있다.

[11]에서는 테이블 참조 연산 과정에서 테이블 주소를 셔플하는 방식의 대응 기술을 추가로 제안하였다. 이는 테이블 참조 연산 외부에서 난수를 생성하여 참조 연산에 사용하여야 한다는 전제 조건이 요구되므로, 화이트 박스 환경에서 이와 같은 구현을 사용하는 구체적인 방법을 제시하지 않는 한 직접적인 적용이 매우 어려울 것이다. 이와 유사한 맥락으로, 단순히 테이블 참조 연산의 순서를 바꾸어 공격자가 정확한 테이블 출력 위치를 분류하지 못하도록 하는 대응 기법이 제안될 수 있다. 그러나 이 경우에도

Type II-M 테이블의 출력이 각각 다음 연산인 Type IV-II 테이블 참조 연산의 입력으로 사용되므로, 완전히 테이블 출력을 섞어 공격 복잡도를 비약적으로 높인다고 볼 수 없다. 예를 들어, AES 상태의 첫 바이트와 세 번째 바이트를 입력으로 취하는 Type II-M 테이블 참조 연산 두 번의 정보를 셔플하였다고 가정할 때, 해당 두 바이트의 연산은 이후의 AES 연산인 ShiftRows나 MixColumns의 최종 XOR 연산에서 서로 관여하지 않는다. 따라서 두 테이블의 출력은 각각 서로 다른 Type IV-II 테이블의 입력으로 사용될 것이다. 이를 통해 공격자가 셔플된 테이블 참조 출력을 재분류할 수 있으므로, 이와 같은 방식의 구현은 기존의 공격에 대해 어떠한 추가적인 안전성도 보장할 수 없다.

바이너리 주소 trace의 특정 패턴으로부터, 해당 연산 패턴이 정확히 화이트 박스 암호의 어떤 테이블 참조 연산에 해당하는지 밝혀내는 간단한 방법이 있다. 본래 Chow가 화이트 박스 암호를 제안할 때 언급했던 외부 인코딩(External encoding) 과정은 암호 알고리즘의 모듈화 관점에서 상당한 리소스 소모를 요구하므로, 실제 화이트 박스 상용 환경에서는 이를 배제한 채 암호를 구현한다. 따라서 [2]에서 제안된 Type I의 테이블은 실제로 사용되지 않고, 암호에 입력된 평문은 어떠한 인코딩 과정도 거치지 않은 채 Type II 테이블의 입력으로 사용된다. 따라서 공격자는 바이너리 데이터 trace 정보에 대해 입력 평문 정보와의 상관 계수를 구하여 해당 입력 평문 값이 직접적으로 연산에 사용되는 시점인 Type II 테이블 참조 연산의 위치를 특정할 수 있다.

다음은 GitHub에 공개된 Lee의 Masked White-Box AES의 데이터 trace와 입력 평문값의 상관계수를 계산한 결과를 통해, 주소 바이너리 trace로부터 평문 정보가 드러나는 지점의 패턴을 파악할 수 있음을 보인다. Fig 6은 Fig 5에서 파악한 Type II-M의 패턴 중 첫 라운드에 해당하는 연산부에서 평문 상관계수가 1을 나타내는 지점을 확대한 것이다. 그림의 가로축은 Lee의 구현이 연산을 수행하는 시간 정보에 해당하며, 세로 축은 바이너리 주소 값의 최하위 바이트 정보와 평문 및 바이너리 데이터 정보간의 상관계수 값을 나타내고 있다. 이 때, 상관계수는 최대 1의 값을 가지므로, 이를 주소 바이트와 시각적으로 비교할 수 있도록 255배의 스케일링을 거친 후 두 trace를 겹쳐 나타내었다. 상관계수 계산은 그레이 박스 부채널 분석에서의 피어슨 상관계수 연산을 적용하였다. 모든 평문의 정보가 해당 연산부에서 드러나는 것을 확인할 수 있으므로, 이 지점이 입력된 평문이 최초로 테이블 참조 연산에 사용되는 1라운드의 Type II-M 연산이라는 것을 단일 포인트 단위로 확인할 수 있다.

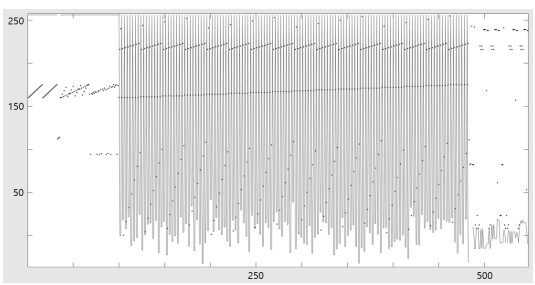


Fig. 6. Plaintext CPA on Lee`s case 1

Fig 7은 마찬가지로 GitHub에 공개된 Chow의 White-Box AES 연산에 대한 첫 번째와 세 번째 평문 바이트의 CPA 결과이다. 앞서 기술한 대로 해당 구현은 테이블 참조 연산의 순서가 Lee의 구현과 다르게 구성되어 있고, 패턴 전 영역에 걸쳐 평문 CPA의 상관계수가 높게 나타나는 구간이 분포되어 있음을 확인할 수 있다. 평문에 대한 상관계수가 높게 나타나는 지점의 연산을 Type II 테이블 참조라고 가정할 때, 해당 구현이 MixColumn 연산에 관여하는 4개의 평문 바이트 단위로 Type II 테이블

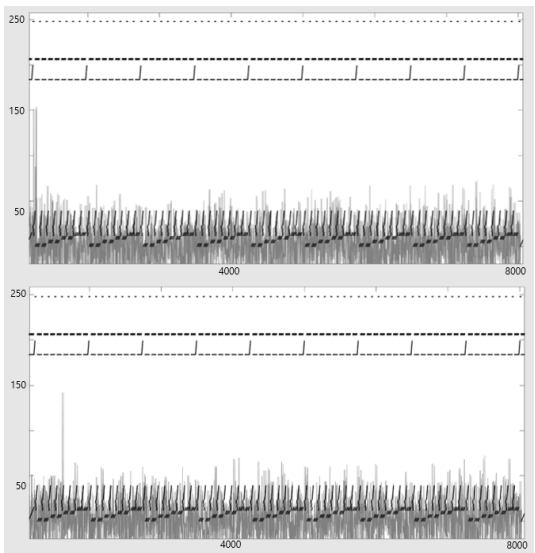


Fig. 7. Binary address trace and plaintext CPA on Chow`s implementation (Upper : First byte, Below : Third byte)

연산과 이에 대한 XOR 연산, Type III의 연산과 이들에 대한 XOR 연산을 순차적으로 수행함을 추측할 수 있다.

3.2 생존 비트 조합 공격

본 절에서 설명하게 될 생존 비트 조합 공격은 맥락상 Bogdanov의 고차 DCA와 같다[11]. 2.3절에서 소개한 Type II-M 테이블은 마스킹이 적용된 T_{y_i}의 연산 결과값 32비트의 정보와 마스크 난수 32비트의 정보를 인코딩이 적용된 상태로 출력한다. Bos가 소개한 DCA 공격에서 서술하였듯, 인코딩이 적용된 테이블 출력은 단일 비트 단위로 기존 AES의 라운드 연산 결과와 관련된 정보를 일부 포함하게 된다[10]. 그리고 이 특성은 Type II-M 테이블의 마스크 출력 정보에서도 마찬가지로 나타난다. 따라서 공격자는 단일의 Type II-M 테이블 출력 64비트 중 마스크에 해당하는 32비트 정보를 마스킹이 적용된 T_{y_i} 연산 결과값에 XOR하는 조합 방식으로 기존의 정보를 복원할 수 있다.

3.1의 가정에 따라, 공격자는 Type II-M 테이블의 연산 위치를 정확하게 파악할 수 있다. 해당 위치의 데이터 바이너리 trace를 획득한 공격자는 단순히 해당 비트들을 XOR하는 과정만으로 2차 trace를 생성할 수 있다. T_{y_i}의 결과값에 마스킹이 적용된 테이블 출력을 T_m, 마스크 값에 대한 테이블 출력을 m이라 할 때 2차 trace는 다음 식 (1)을 통해 형성될 수 있다.

$$SecT = T_m \oplus m^T \tag{1}$$

이 때, m^T는 32비트의 마스크 출력 행벡터 m을 열벡터의 형태로 전치하는 연산을 의미한다. 따라서 행벡터 T_m과 열벡터 m의 XOR 연산은 T_m의 길이와 같은 행, m의 길이와 같은 열을 가지는 $SubT$ 행렬로 결과값을 출력한다. 형성된 2차 trace $SubT$를 다시 행벡터의 형태로 나열하여 일반 DCA와 같은 분석 기법을 적용할 수 있다. 실험적으로 이와 같은 공격을 수행할 경우 16개의 키 바이트 중 절반의 값이 올바른 키로 나타남을 확인할 수 있다. 이는 DCA 공격의 근거가 되는 4×4 비선형 인코딩의 비트 특성 보존 정도에 의해 설명된다. [4]에서 설명한대로, 연산 중간값의 단일 비트에 잔류하는, 인코딩 전 암호 연산 정보의 성질을 통해 추측 중간값과의 상관 계수에 분포차가 발생할 수 있다. 본 논문에서는 해당 분포차를 비트 특성이라 표현하고, 이 비트 특성을 보존하여 공격 대상이 되는 특정 테이블 출력 비트를 생존 비트로 표현하였다.

4×4 비선형 인코딩은 블록 암호에서 주로 사용하는 4비트의 비선형 치환 테이블 연산으로 생각할 수 있으며, 암호 분석가들은 이에 대해 차분 분석 공격(DC, Differential Cryptanalysis) 혹은 선형 분석 공격(LC, Linear Cryptanalysis) 수행에 있어 해당 테이블의 단일 비트별 바이너리 함수의 특성을 고려한다[12]. 화이트 박스 암호에서 사용하는 비선형 인코딩 또한 이와 같은 방식으로 접근할 수 있으며, 암호학적으로 안전성이 떨어지는 테이블을 인코딩 테이블로 선택할 경우 비트의 특성이 상대적으로 강하게 드러나게 된다. 이러한 취약 비트를 다시 부채널 분석 공격에서 주로 사용하는 상관계수 분석을 통해 통계적으로 정렬할 경우, 특징이 강하게 남아있는 비트에 대해 중간 정보를 추측할 수 있다.

마스크 난수가 드러나는 테이블 출력에서도 이 특성을 동일하게 활용할 수 있으므로, 약한 특성의 바이너리 함수를 포함하는 비선형 테이블의 경우 마스크 난수와 마스킹이 적용된 암호화 중간값의 정보를

Algorithm 1. 2nd Order DCA

INPUT 256 Plaintext $P[0...15]$, 256 Binary trace T

OUTPUT Key k

1. $T_{begin} \leftarrow \max_{\arg}(PtCorr(P[0]))$,
2. $T_{end} \leftarrow \max_{\arg}(PtCorr(P[1]))$
3. $SubL \leftarrow T_{end} - T_{begin}$
4. **for** $i = 0...15$ **do**
5. $T_{begin} \leftarrow \max_{\arg}(PtCorr(P[i]))$
6. $T_{end} \leftarrow T_{begin} + SubL$
7. $SecT \leftarrow T[T_{begin} : T_{end}]^{\oplus 2}$
8. $Mid \leftarrow AES_{MixColumn}(P[i], gkey)$
9. $k[i] \leftarrow \max_{\arg}(DCA(Mid, SecT))$
10. **end for**
11. **return** k

Fig. 8. Algorithm of Second-order DCA

드러내게 된다. 또한, 각 중간 정보에 적용된 인코딩은 매 시행마다 특정 위치의 비트에 대해 고정된 값으로 적용되므로, 같은 위치의 비트가 인코딩 이전의 정보를 매번 강하게 보존하고 있을 확률이 높다. 따라서 단순히 그레이 박스와 같은 2차 상관계수 분석만으로도 일부의 키 바이트를 복원해낼 수 있다.

생존 비트 조합 공격에 대한 전체 공격은 알고리즘 1의 과정에 따라 수행할 수 있다. 매 공격 시행에 대해, 입력되는 i번째 평문 바이트 $P[i]$를 통해 계산되는 T_{begin}과 T_{end}는 각각 3.1절에서 정교하게 분류한, 바이너리 trace에서 공격 대상 시점의 시작점과 끝점을 의미한다. $PtCorr$과 DCA는 각각 평문과 AES 연산 중간값에 대한 바이트와 비트 단위의 상관계수를 계산하며 $\max_{\arg}$는 상관계수가 최대가 되는 시점의 위수를 출력한다. 7번 과정에서의 $T[T_{begin}:T_{end}]^{\oplus 2}$ 연산은 하나의 비트 벡터를 교차하여 XOR함을 의미한다($\boldsymbol{a}\oplus\boldsymbol{a}^T$). 이는 식 (1)에서 설명한 2차 trace 계산을 Type II-M 테이블 출력 전체에 대해 수행하여 $SubT$를 생성하는 과정을 의미하며, Type II-M 테이블 출력 64비트 중 T_m과 m의 위치를 정확히 구분해낼 수 없음을 가정하여 설정되었다. 7번 과정의 결과로 나타나는 $SecT$는 본래 $SubL\times SubL$의 행렬이 되나, 실제로 연산에 사용할 때는 이 행렬의 모든 요소를 포함하는 하나의 벡터로 간주한다. 8번 과정은 AES 알고리즘의 1라운드 연산에서 MixColumn 라운드 함수의 출력 중간값을 추측하였다.

그러나 4장에서 확인할 수 있듯, 이러한 공격을 통해 찾아낼 수 있는 키 바이트는 전체의 절반에 불과하고, 탐색된 절반의 옳은 키조차 노이즈 혹은 고스트 픽(ghost peak[5])과 완전히 구분되는 정도의 상관 계수를 나타내지 않으므로, 실제 키에 대한 올바른 복원으로 볼 수 없다. 따라서 다음의 4비트 비선형 조합 과정을 통해 전체 16바이트의 올바른 키를 모두 설득력 있는 상관계수 범위에서 찾아내는 과정을 보이며 해당 공격의 실효성을 보였다.

3.3 4비트 비선형 조합

생존 비트 조합 공격은 화이트 박스 구현에 포함되는 4×4 비선형 인코딩 이후에도 보존되는 비트의 특성에 의해 발생한다. 따라서 공격자는 비선형 인코딩이 적용된 테이블 출력에 추가 조작을 가해 이 특성을 강하게 드러낼 수 있다. 여기서 말하는 비트의 특성이란 비선형 인코딩 이후에도 특정 비트가 AES 연산 중간값의 성질과 유사한 경향을 나타내는 정도를 의미하며, Masked White-Box AES 논문에서 이를 수식적으로 보였듯 4×4 비선형 인코딩은 이 특성을 충분히 제거하지 못한다. 또한, 이 특성은 다른 4×4 비선형 인코딩을 추가 조합하는 과정을 통해 더욱 강하게 드러낼 수 있다.

3.2절에서 탐지한 T_m과 m에 대한 2차 trace는 바이너리 trace의 특성상 연산 중간값과 같은 값을 가진다. 해당 값은 실제 AES 라운드 연산에 선형과 비선형의 인코딩을 포함하는 Type II-M 테이블 출력값이다. 선형 인코딩은 상관계수에 영향을 미치지 않으므로, 공격자는 해당 출력값에 적용된 비선형 인코딩의 특성만을 고려해도 충분하다. 공격자는 3.2절에서 설명한 $SecT$를 계산하는 과정 중간에 추가로 임의의 4비트 비선형 테이블을 조합하는 연산을 추가한다. 그로 인해 본래 Type II-M 테이블 출력값에서 드러나던 일부 비트 특성을 보다 강하게 나타낼 수 있고, 이로 인해 공격자는 정확한 비밀 값 복원이 가능하게 된다. 위 과정에 따라 $NSecT$는 다음 식의 형태로 계산할 수 있다.

$$NSecT = Non4(T_m)\oplus m^T \tag{2}$$

식 (2)에서 $Non4$는 테이블 출력값에 추가로 조합하는 4비트 비선형 테이블 연산을 의미한다. 다시 말해, Non_4의 연산은 입력 T_m에 대한 임의의 4비트 단위 치환 테이블로 이해할 수 있다. 공격은 하나의 $NSecT$에 대한 공격만으로 완전히 성공하지 않으므로, 다수의 랜덤 비선형 치환 테이블에 대해 이와 같은 과정을 반복 수행하는 생존 비트 조합 공격은 알고리즘 2를 통해 수행될 수 있다.

알고리즘 2는 적절한 임계점을 설정하여 그보다 높은 키 상관계수가 발생할 때까지 공격을 반복 수행한다. 9번 과정의 $RNon4$는 랜덤한 4비트 비선형 연산을 의미하며, 단순히 이전 시행에서의 Non_4 테이블 인덱스를 셔플하는 연산으로 이해할 수 있다. 이 과정을 거친 공격 대상 포인트에 대해 3.2절에서 설명한 비선형 인코딩 조합을 수행하여 2차 trace를 계산한다. 일반적으로 랜덤한 4비트의 비선형 인코딩은 총 16!개의 경우의 수를 가질 수 있다. 그러나 4장의 실험 결과에 따라, 모든 경우를 탐색하거나

특정 대푯값을 설정하지 않고, 단지 원하는 결과가 나타날 때까지 랜덤한 비선형 연산을 조합하더라도 효과적인 공격을 수행할 수 있다. 더불어 13번의 조건문을 볼 때, 최대의 상관계수를 가지는 키를 옳은 키로 선택하는 것을 확인할 수 있는데, 이는 언제나 올바른 키에서만 특정 임계점을 넘는 상관계수가 발생하기 때문에 가능하다.

T_m에 대한 4비트 비선형 인코딩 조합은 본래 Type II-M 테이블의 출력 인코딩과 결합하여 새로운 비트 특성을 나타낼 수 있다. 실험적으로 이와 같은 비선형 조합 공격은 마스킹이 적용된 화이트 박스 AES에 대한 비트 조합 공격의 상관계수를 가시적으로 높일 수 있었고, 기존에 탐색하지 못했던 키 바이트 또한 높은 신뢰도로 탐색할 수 있었다. 이는 3.2절에서 설명한 4비트 비선형 테이블의 바이너리 함수의 특성과 연관되어 있다. 테이블 출력에 적용된 새로운 4비트 테이블로 인해 기존 인코딩이 전혀 다른 테이블 인코딩으로 변형되며, 해당 테이블이 기존

Algorithm 2. 2nd Order DCA (composite)

INPUT 256 Plaintext $P[0...15]$,
256 Binary trace T
OUTPUT Key k

$pk \leftarrow 0$
$T_{begin} \leftarrow \max_{\arg}(PtCorr(P[0]))$
$T_{end} \leftarrow \max_{\arg}(PtCorr(P[1]))$
$SubL \leftarrow T_{end} - T_{begin}$
while $pk[:] > Threshold$ **do**
  for $i = 0...15$ **do**
    $T_{begin} \leftarrow \max_{\arg}(PtCorr(P[i]))$
    $T_{end} \leftarrow T_{begin} + SubL$
    $SecT \leftarrow RNon4(T[T_{begin} : T_{end}])^{\oplus 2}$
    $Mid \leftarrow AES_{MixColumn}(P[i], gkey)$
    $k_{guess} \leftarrow \max_{\arg}(DCA(Mid, SecT))$
    $pk_{guess} \leftarrow \max(DCA(Mid, SecT))$
    if $pk[i] < pk_{guess}$ **do**
      $k[i] \leftarrow k_{guess}$
      $pk[i] \leftarrow pk_{guess}$
    end if
  end for
end while
return k

Fig. 9. Algorithm of composite Second-order DCA

보다 취약하게 정보를 드러내도록 유도할 수 있다. 따라서 기존에 드러난 키에 대한 상관계수를 높이는 효과를 가져올 수 있으며, 기존에 드러나지 않던 키 바이트에 대한 합리적인 복원 또한 가능하게 한다. 이 때, 사용한 Threshold는 4장의 실험을 통해 결정할 수 있으며, pk_{guess}는 각 시행에서 최대의 peak을 가지는 추측 비밀키에 대한 상관계수를 의미한다. pk는 16개의 인자를 가진 배열로, 현재 추측키로 사용하고 있는 키의 상관계수를 저장하며, pk_{guess}가 해당 값을 넘을 경우 키를 갱신하기 위해 사용된다.

해당 공격의 정확도를 설명하기 위해 4비트 비선형 조합 이후 검출 가능한 높은 상관계수가 올바른 키 추측에 의한 값임을 확인할 수 있어야 한다. 반대로 말해, 틀린 키 추측으로부터 [5]에서 설명하는 ghost peak이 발생하지 않음을 증명할 수 있어야 한다. 그러나 [3]에서 주장한 것과 같이, 인코딩을 거친 테이블 출력 중간값으로부터 올바르지 않은 상관계수는 발생하지 않고, 4비트 비선형 인코딩의 종류를 바꾸는 과정과 같은 효과를 낼 뿐이다. 따라서 4비트 비선형 조합으로 인해 DCA 공격과 다른 효과를 내는 ghost peak은 발생하지 않으므로, 랜덤한 4비트 비선형 조합의 모든 시행에서 가장 높은 peak을 나타내는 추측 키를 옳은 키로 선택할 수 있다. 이와 관련된 실험 결과는 4장에서 더욱 자세히 보이도록 한다.

IV. 실험 결과

본 논문에서 제안한 공격 기법의 실험 환경은 다음과 같이 설정하였다. 먼저 Valgrind 바이너리 분석 도구를 통한 바이너리 trace 수집 방식과 공격 대상 알고리즘인 Chow의 White-Box AES, Lee의 Masked White-Box AES Case 1의 구현은 모두 GitHub에 공개된 오픈 소스와 실행 파일을 사용하였다. 암호 연산과 바이너리 trace 수집은 Ubuntu 64bits Linux에서 수행하였으며, 화이트 박스 환경의 특성상 모든 바이트가 16진수 00부터 FF까지 순서대로 구성된 평문 256개만을 사용하여 바이너리 trace를 수집하였다.

먼저 단순한 환경에서도 DCA 수행이 가능함을 보이기 위해 Chow의 White-Box AES에 대해 일반적인 DCA 공격을 수행하였다. 그 결과는 Fig 10

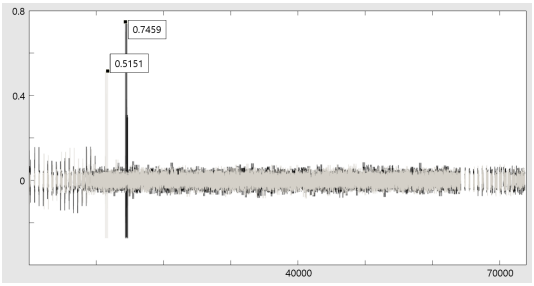


Fig. 10. Correlation peak of DCA attack on Chow`s implemenation

에서 확인할 수 있다.

Chow의 초기 구현에 대한 DCA는 Fig 10에 나타나는 것처럼 상당히 강력한 효과를 보인다. 가로축은 위와 동일하게 연산 시간 정보를 나타내며, 세로 축이 의미하는 상관계수는 옳은 키에서 0.5부터 0.7의 수준으로 분포하며, 틀린 키에 대한 상관계수는 최대 0.3 정도로 나타난다. 따라서 옳은 키와 틀린 키를 설득력 있게 구분할 수 있음이 확인되었다. DCA는 우리의 실험 환경과 공격 수행 방식으로 충분히 수행 가능한 공격이며, 이 공격 과정에 테이블의 세부적인 구조를 파악하거나 연산의 종류, 비선형 인코딩의 복원을 고려하는 등의 암호 분석적 기법은 전혀 포함되어 있지 않다. 단지 화이트 박스 환경의 기본 가정인, 연산 중간 정보의 정확한 값만을 가지고 단일 비트 단위로 부채널 분석에 사용하는 통계적 분석을 적용하였다.

다음은 Lee의 Masked White-Box AES에 대해 수행한, 3.2절에서 설명한 생존 비트 조합 DCA 공격 결과이다. Fig 11에서 회색의 상관계수는 옳은 키를 탐색한 바이트 중 가장 높은 상관계수를 나타낸 것이며, 검정 상관계수는 틀린 키를 선택한 바이트에 대한 상관계수를 나타낸 것이다. 3.2절에서 언급한 바와 같이, 생존 비트 조합 공격의 결과, 옳은 키의 상관 계수가 틀린 키의 상관 계수와 비교하여 높은 값을 나타내지 못했으며, 결과적으로 전체 16바이트 중 8바이트의 옳은 키만을 탐색해낼 수 있었다. 옳은 키의 경우에도 상관계수가 0.4의 수준에 그쳤고, 틀린 키의 상관계수는 Chow의 구현에 대한 DCA와 마찬가지로 0.3의 수준을 보였다. 그러나 올바른 키를 추측한 바이트 중 일부는 상대적으로 틀린 키 바이트에 비해 큰 차이를 나타내는 경우도 존재하여 공격의 성공 가능성을 보였다. 따라서 이 분석 기법의 성능을 향상 시킬 수 있는 3.3절의 후처리 기법을 사용한다면 모든 바이트의 키를 올바르게 찾아낼 수 있을 것으로 여겨졌다.

3.3절의 설명대로 랜덤한 4비트 비선형 테이블을 마스킹 된 연산 중간값과 마스크 값 양쪽에 모두 적용한 공격 결과이다. 해당 공격은 임의의 임계 상관계수인 0.5를 넘을 때까지 공격을 반복 수행한 것으로, Fig 12와 같은 결과를 얻을 수 있었다. 회색 상관계수는 4비트 비선형 조합을 적용하기 이전의 옳은 키에 대한 상관계수이며, 검정 그래프는 동일 바이트의 중간값에 4비트 비선형 조합을 적용하여 높아진 상관계수를 나타내고 있다. Fig 12에서 나타나는 바와 같이, 옳은 키에 대한 상관계수가 0.36에서 0.81까지 높은 폭으로 상승한 것을 확인할 수 있으며, 이러한 현상은 옳은 키에서만 두드러지게 나타난다는 것을 확인할 수 있었다. 4비트 비선형 테이블 조합을 수행할 때는 이전 시행에서의 최대 상관계수보다 높은 값에 대해서만 키와 상관계수를 갱신하도록 하였다. 두 공격에 대한 결과를 Table 1에 정리하여 확연히 비교할 수 있도록 하였다. Table 1의 Max peak과 Noise peak은 각각 올바른 키와 틀린 키 중에서 나타난 최대의 상관계수를 의미하며, Ratio는 이 두 값을 통해 계산한 SNR(Signal to Noise Ratio, 신호대 잡음비율)이다.

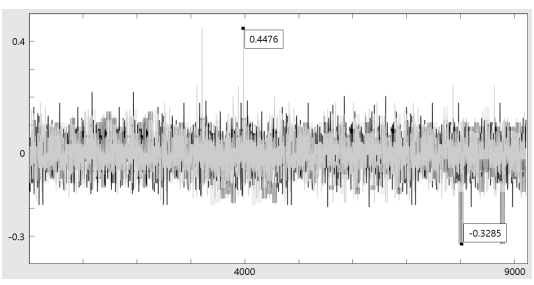


Fig. 11. Second-order DCA on Lee`s case 1

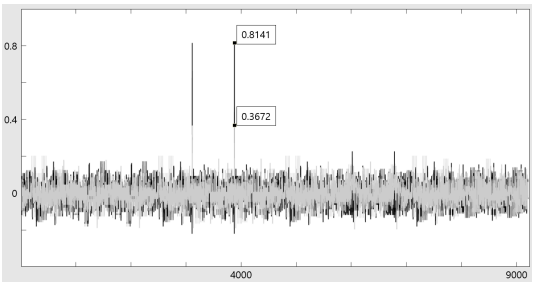


Fig. 12. 4-bit nonlinear composit second-order DCA on Lee`s case 1

Table 1. Compare performance between two attacks

	Max peak	Noise peak	Ratio
Alg 1	0.4476	0.3285	1.3626
Alg 2	0.8141	0.3285	2.4782

4비트 비선형 조합 공격의 성능은 곧 랜덤한 비선형 테이블의 조합이 얼마나 빠르게 옳은 키에 대한 상관 계수를 올릴 것인지에 의존할 것이다. 해당 공격에 대한 온전한 설득력을 갖추기 위해선 다음의 두 가지가 실험적으로 보여져야 한다.

첫째, 통계적으로 해당 공격이 항상 올바른 키를 복원할 수 있음이 보여져야 한다. 이 과정에서는 올바른 키를 복원하기까지 사용되는 $RNon_4$의 요구 개수에 대한 통계적인 산출이 수반되어야 한다. 실제로, 공격의 설득력을 위해 임의의 비선형 인코딩을 거치는 Lee의 구현과 같은 Type II-M 테이블 다수에 대해 다수의 4비트 비선형 조합 공격을 수행하여, 총 100개의 공격 표본을 획득하였다. 이 때 100개의 공격 표본이란 각 실험 시행마다 참조 테이블에 적용된 비선형 인코딩의 종류가 항상 다르고, 공격에 적용된 비선형 연산이 항상 16!의 경우 중 독립적인 랜덤으로 선택됨을 의미한다. 공격의 성능과 복잡도를 명료하게 나타내기 위해 새로운 $RNon_4$를 생성하는 공격은 전체 16바이트의 AES 마스터 키가 모두 올바르게 복원되는 시점을 공격 성공으로 간주하였고, 공격 수행 성공까지 사용된 $RNon_4$의 개수를 공격 성공률과 복잡도를 나타내는 지표로 생각하여 해당 값의 평균과 분산을 통해 나타내었다. Fig 13의 그래프를 통해 이 결과를 시각적으로 확인할 수 있다. 그래프의 가로 축의 100번에 시행에 대해 세로 축의 사용된 $RNon_4$ 개수를 시각적으로 확인할 수 있으며, 두 개의 가로선을 통해 평균과 표준편차를 나타내었다.

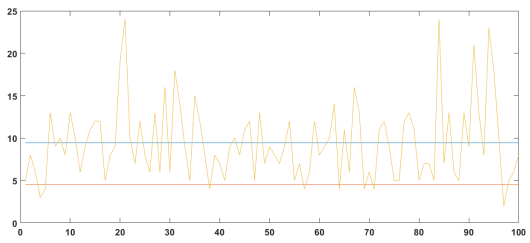


Fig. 13. Attack count of Alg. 2(Avg : 9.45 / StDev : 4.53)

둘째, 공격자는 실제로 올바른 키가 복원된 시점을 판단할 수 없으므로, 올바른 키로 판단할 수 있는, 설득력 있는 상관계수 임계점을 결정할 수 있어야 한다. 이 과정을 통해 3.3절의 Algirithm 2에서 사용할 Threshold를 결정할 수 있다. Fig 11과 Fig 12에 나타나는 실험을 통해 Lee의 구현이 0.33을 넘지 않는 최대 노이즈 상관계수를 가짐을 알 수 있다. 그러나 암호마다 서로 다른 비선형 인코딩을 사용하는 화이트 박스 암호 특성상, 노이즈 상관계수의 임계치를 수식적으로 계산할 수 있는 어떠한 방법도 존재하지 않는다. 따라서 본 논문에서는 이에 대한 설득력 있는 상관계수의 범위를 설정하기 위해, 상관계수 임계치를 0.41부터 0.50까지 설정하며 Fig 13에서 수행한 것과 같은 100개의 표본 공격을 반복 수행하였다. 해당 실험 결과는 Fig 12에 나타나 있으며, 상단은 각 상관계수 임계치에 대한 평균, 하단은 분산의 변화곡선을 나타내고 있다. 두 값 모두 가로 축의 원점 값을 단순히 실제 키를 찾아내기만 하는 Fig 13의 평균과 분산 값으로 설정하였다. Fig 14에서 나타나는 것처럼, 임계치를 높게 설정함에 따라 요구되는 $RNon_4$의 개수가 점차 증가하는 것을 확인할 수 있다. 또한 총 1000번의 모든 실험에 대해 틀린 키가 더 높은 상관계수를 가지게 되는 경우는 나타나지 않았으므로, 각 임계치에 대한 오류 발생율을 계산하지 않았다.

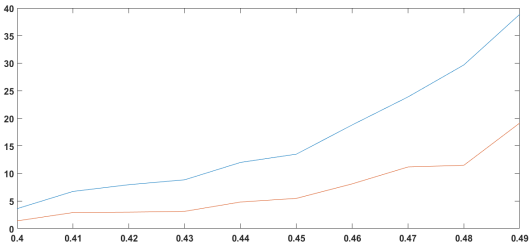


Fig. 14. Attack count of Alg.2 with Thresholds (Blue : avg / Orange : stdev)

V. 결 론

화이트 박스 암호에 대한 공격은 현대 정보 교환 체계에서 다루어지는 중요한 보안적 가치를 위협한다. 화이트 박스 암호의 안전성은 모듈화의 관점에서 설득력 있는 안전성을 제공할 수 없음이 공공연하게

알려져 있고, 이를 위해 암호 설계자들은 다양한 방식의 테이블 설계나 중간값 조합 방식을 제시함과 동시에, 공격자가 수행할 가능성이 있는 선형, 비선형의 인코딩 복원 공격을 예측하고 있다.

DCA 공격의 등장으로 그레이 박스 공격이 화이트 박스 암호에 치명적인 위협으로 다가왔다. 그레이 박스 환경의 공격은 본래 공격자가 고성능의 분석 장비를 보유한 상태에서만 수행 가능한 공격으로 여겨졌으나, 화이트 박스 환경에서 이와 동일한 맥락의 공격을 수행하는 것은 단순히 PC 한 대만으로도 가능하다. 더불어 기존 부채널 분석에 익숙하지 않은 공격자라도 어렵지 않게 이를 실현할 수 있다. 고도의 암호 분석적 관점에서의 공격도, 암호학의 기본 원리인 Kerckhoffs의 원리를 위반하며 구현한 반공개적 알고리즘으로도 단순한 그레이 박스 맥락의 통계적 분석을 막아낼 수 없었다. 따라서 화이트 박스 암호 구현자들은 그레이 박스 환경과 같은 대응기술을 적용하는 선택이 불가피하였다.

테이블 인코딩을 통한 화이트 박스 암호 연산은 기존 그레이 박스 환경에서의 통계적 분석을 충분히 견딜 수 없었다. 이를 토대로 고안된 그레이 박스 공격의 대응기술과 같은 구현은 마찬가지로 그레이 박스 환경의 고차 분석 공격에 의해 비밀 정보를 드러내게 되었다. 본래 그레이 박스 대응기술은 한시도 중간 정보가 드러나지 않도록 암호를 설계하며, 난수 정보조차 최대한 독립적으로 드러나지 않도록 면밀한 구현 검증 과정을 수행하여야 한다. 따라서 고차 DCA 공격에 대응하기 위해서는 조합되는 대상인 마스크의 정보가 테이블 출력으로 드러나지 않도록 하거나 같은 그레이 박스 고차 대응기술 기법을 적용하여야 한다. 더불어 본 논문에서 제안한 공격 기법에 약간의 공격 복잡도만을 추가하여 수행할 수 있는 다중 조합 공격에도 안전하도록 테이블 연산과 중간값 조합 연산을 고려하여야 한다.

화이트 박스 암호에 대해 그레이 박스 맥락의 통계적 공격을 수행하는 것이 심각한 위협이라는 것은 이미 공공연한 사실이다. 그러나 그레이 박스 공격자들은 이미 화이트 박스에서 고려하는 공격 수준보다 훨씬 고차원적인 공격을 수행할 수 있으며, 다양한 환경에서 이를 적용하여 비밀 정보를 복원하는 연구를 진행해왔다. 화이트 박스 암호 구현자들은 일반적인 경우보다 그레이 박스 분석에 더욱 취약할 수 있는 화이트 박스 환경을 고려하여 기존 암호 분석 기법에 더불어 그레이 박스 분석 기법에 대해 필수적으로 민감하게 반응하여 대처할 수 있어야 한다.

References

[1] Kocher, P., Jaffe, J., Jun, B.: "Differential Power Analysis," In: Wiener, M. (ed.) CRYPTO 1999. Aug. 1999, LNCS, vol. 1666, pp. 388 - 397. Springer, Heidelberg (1999)

[2] Chow, S., Eisen, P., Johnson, H., van Oorschot, P.C.: "White-box cryptography and an AES implementation," In: Nyberg, K., Heys, H.M. (eds.) SAC 2002. Feb. 2003, LNCS, vol. 2595, pp. 250 - 270. Springer, Heidelberg (2003) (to appear)

[3] Bos, J.W., Hubain, C., Michiels, W., Teuwen, P.: Differential computation analysis: hiding your white-box designs is not enough. In: Gierlichs, B., Poschmann, A.Y. (eds.) CHES 2016. Aug. 2016, LNCS, vol. 9813, pp. 215 - 236. Springer, Heidelberg (2016). doi: 10.1007/978-3-662-53140-2_11

[4] S. Lee, T. Kim, and Y. Kang, "A Masked White-Box Cryptographic Implementation for Protecting Against Differential Computation Analysis", in IEEE transaction on information forensics and security, vol. 13, no. 10, 2018, pp. 2602-2615. Apr. 2018.

[5] Brier E., Clavier C., Olivier F. (2004) "Correlation Power Analysis with a Leakage Model." In: Joye M., Quisquater JJ. (eds) Cryptographic Hardware and Embedded Systems - CHES 2004. Lecture Notes in Computer Science, vol 3156. pp. 16-29, Springer, Berlin, Heidelberg

[6] Chow S., Eisen P., Johnson H., van Oorschot P.C. (2003) "A White-Box DES Implementation for DRM Applications." In: Feigenbaum J. (eds) Digital Rights Management. DRM 2002. Lecture Notes

in Computer Science, vol 2696. Springer, Berlin, Heidelberg, pp. 1-15.

[7] Billet O., Gilbert H., Ech-Chatbi C. (2004) "Cryptanalysis of a White Box AES Implementation." In: Handschuh H., Hasan M.A. (eds) Selected Areas in Cryptography. SAC 2004. Lecture Notes in Computer Science, vol 3357. Springer, Berlin, Heidelberg, pp. 227-240.

[8] Dinur I. (2018) "An Improved Affine Equivalence Algorithm for Random Permutations." In: Nielsen J., Rijmen V. (eds) Advances in Cryptology - EUROCRYPT 2018. EUROCRYPT 2018. Lecture Notes in Computer Science, vol 10820. Springer, Cham. pp. 413-442.

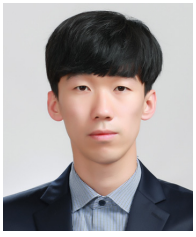
[9] C. H. Baek, J. H. Cheon and H. Hong, "White-box AES implementation revisited," in Journal of Communications and Networks, vol. 18, no. 3, pp. 273-287, June 2016.

[10] Sasdrich P., Moradi A., Güneysu T. (2016) White-Box Cryptography in the Gray Box. In: Peyrin T. (eds) Fast Software Encryption. FSE 2016. Lecture Notes in Computer Science, vol 9783. Springer, Berlin, Heidelberg, pp. 185-203.

[11] Bogdanov A., Rivain M., Vejre P.S., Wang J. (2019) Higher-Order DCA against Standard Side-Channel Countermeasures. In: Polian I., Stöttinger M. (eds) Constructive Side-Channel Analysis and Secure Design. COSADE 2019. Lecture Notes in Computer Science, vol 11421. Springer, Cham. pp. 118-141.

[12] Saarinen MJ.O. (2012) Cryptographic Analysis of All 4 × 4-Bit S-Boxes. In: Miri A., Vaudenay S. (eds) Selected Areas in Cryptography. SAC 2011. Lecture Notes in Computer Science, vol 7118. Springer, Berlin, Heidelberg. pp. 118-133.

〈저자소개〉



이 예 찬 (Yechan Lee) 학생회원
2017년 2월: 고려대학교 정보수학과 학사
2018년 3월~현재: 고려대학교 정보보호학과 석사과정
〈관심분야〉 화이트박스 암호, 부채널 공격

진 성 현 (Sunghyun Jin) 학생회원
2015년 2월: 서울시립대학교 수학과 학사
2017년 2월: 고려대학교 정보보호학과 석사
2017년 3월~현재: 고려대학교 정보보호학과 박사과정
〈관심분야〉 부채널 공격

김 한 빛 (Hanbit Kim) 학생회원
2014년 2월: 고려대학교 신소재공학과 학사
2016년 2월: 고려대학교 정보보호학과 석사
2016년 3월~현재: 고려대학교 정보보호학과 박사과정
〈관심분야〉 부채널 공격, 부채널 대응기법, 암호시스템 안전성 분석 및 고속구현

김 희 석 (Heeseok Kim) 정회원
2006년: 연세대학교 수학과 학사
2008년: 고려대학교 정보보호대학원 석사
2011년: 고려대학교 정보보호대학원 박사
2011년 9월~2012년 12월: Bristor University 박사후 연구원
2013년~2016년 8월: 한국과학기술정보연구원(KISTI) 선임연구원
2015년~2016년 8월: 과학기술연합대학원대학교(UST) 조교수
2016년 9월~현재: 고려대학교 과학기술대학 사이버보안전공 조교수
〈관심분야〉 부채널 공격, 암호시스템 안전성 분석 및 고속구현, 암호칩 설계 기술, 보안관제, 네트워크 보안

홍 석 희 (Seokhie Hong) 종신회원
1995년: 고려대학교 수학과 학사
1997년: 고려대학교 수학과 석사
2001년: 고려대학교 수학과 박사
1999년 8월~2004년 2월: ㈜시큐리티 테크놀로지 선임연구원
2003년 3월~2004년 2월: 고려대학교 정보보호기술연구센터 선임연구원
2004년 4월~2005년 2월: K.U. Leuven ESAT/SCD-COSIC 박사후 연구원
2005년 3월~2013년 8월: 고려대학교 정보보호대학원 부교수
2013년 9월~현재: 고려대학교 정보보호대학원 정교수
〈관심분야〉 대칭키 및 공개키 암호 알고리즘, 부채널 공격 및 대응기법, 디지털 포렌식