NIST 표준 형태 보존 암호에 대한 딥러닝 기반의 신경망 구별자

김덕영[1], 김현지[2], 장경배[2], 윤세영[3],서화정[4]
[1]한성대학교 융합보안학과 석사과정
[2]한성대학교 정보컴퓨터공학과 박사과정
[3]한성대학교 IT융합공학부 학부생
[4]한성대학교 융합보안학과 교수
dudejrdl123@gmail.com, khj1594012@gmail.com, starj1023@gmail.com,
sebbang99@gmail.com, hwajeong84@gmail.com

Deep Learning-Based Neural Distinguisher for NIST Standard Format-Preserving Encryption

Duk-young Kim[1], Hyun-Ji Kim[2], Kyung-Bae Jang[2], Se-Young Yoon[3],
Hwa-jeong Seo[4]
[1,4]Dept. of Convergence Security, Han-sung University
[2]Dept. of Information Computer Engineering, Han-sung University
[3]Dept. of IT Convergence Engineering, Han-sung University

요 약

차분 분석은 암호 분석기법 중 하나이며, 차분 공격을 위해 랜덤 데이터들로부터 차분 특성 (입/출력 차분)을 만족하는 데이터를 구별해 내는 것을 구별자 공격이라 한다. Neural distinguisher는 구별자에 딥러닝을 적용한 것이다. 본 논문에서는 NIST 표준 형태보존암호인 FF1, FF3-1을 위한 단일 차분을 사용한 최초의 신경 구별자를 제안하였다. FF1은 차분으로 $0F$를 사용할 때, 숫자 및 소문자 도메인에서 차분 데이터 구별에 성공하였다 (정확도는 각각 0.85 및 0.52). FF3-1에서는 08을 사용할 때, 숫자 및 소문자 도메인에서 차분 데이터 구별에 성공하였다 (정확도는 각각 0.98 및 0.55).

1. 서론

차분 분석이란 암호 분석기법 중 하나로 입력 차분에 따른 출력 차분을 분석하여 키를 유추할 수 있다면, 암호 알고리즘이 안전하지 않게 설계되었음을 의미한다. 이때, 차분 공격을 위해 무작위 데이터들로부터 차분 특성 (입력 및 출력 차분)을 만족하는 데이터를 구별해낼 수 있다면, 차분 분석에 필요한 공격 복잡도가 줄어든다. 이와 같이 데이터를 구별해내는 공격 기법을 구별자 공격이라 한다.

최근 딥러닝 기술이 발달하면서 구별자 공격에도 딥러닝 기술들이 활발히 적용되고 있다. 하지만 아직까지 NIST 표준 형태 보존 암호인 FF1, FF3-1에 대한 딥러닝 기반의 구별자에 관한 연구는 수행되지 않았다. 본 논문에서는 NIST 표준 형태 보존 암호 (FF1, FF3-1)에 대한 딥러닝 기반의 신경망 구별자를 최초로 제안하였다.

2. 관련 연구

2.1 NIST 표준 형태 보존 암호[1]

개인정보보호법 시행 등으로 인해 데이터베이스 (Database, DB) 암호화의 중요성이 커졌으며, 특히 주민등록번호, 신용카드번호 등의 암호화가 주요 이슈로 대두된다. DB 암호화에 기존 암호기술을 적용할 경우 데이터의 타입이 변하거나 길이가 증가하여 DB 스키마 변경이 필요하지만 형태보존암호를 사용 할 경우 데이터의 타입과 길이를 보존하는 암호화 방식이므로 DB 스키마의 변경 없이 암호화를 적용할 수 있다. 이처럼 형태 보존 암호는 일반 블록암호와 달리 암호문의 평문의 형태가 그대로 유지되도록 보장하는 암호화 방식이다. 예를 들어, 신용카드의 16자리 10진수 번호 중에서 일부 6자리를 암호화해야 한다고 할 때, 만약 128비트 블록 사이즈의 블록 암호 AES를 이용해서 암호화한다면 암호문의 길이도 128비트가 될 것이다. 평문인 6자리 10진수는 10^6 ~ 2^{19}이므로 약 20비트 밖에 안 되는데, 암호문은 이

보다 6배 이상 큰 128비트 길이를 갖는다. 따라서 평문 대신 암호문을 저장하기 위한 추가적인 저장 용량은 필요하지 않게 되기 때문에 형태 보존 암호는 기존의 데이터베이스 시스템에서 이용하고 싶을 때 시간과 비용 면에서 효율적이다. 형태 보존 암호 중 NIST 표준으로 지정된 암호는 FF1, FF3가 있다. FF1와 FF3는 각각 10 라운드와 8라운드로 구성되며 블록 크기와 키 크기는 각각 32비트와 128비트이다. 또한 Feistel 구조로 설계되었으며, 내부 라운드 함수로 암호화 함수 (예: AES)를 사용하며, 해당 암호화 알고리즘은 변경 될 수 있다. 위의 두 암호는 유사한 점도 있지만, FF1은 FF3보다 더 높은 라운드를 가짐으로써 상대적으로 더 안전하며 ,FF3는 FF1에 비하여 데이터 처리량이 더 높다는 이점이 있다.

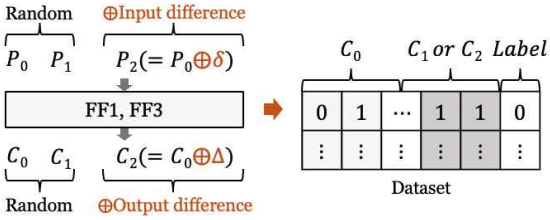
2.2 딥러닝 기반의 신경망 구별자

딥러닝 기술은 데이터에 대한 확률적 예측을 수행하기 적합하다. 딥러닝 기반의 신경망 구별자는 이러한 특성을 활용하여 기존의 구별자 공격에 적용한 것이며, 여러 연구들이 진행되고 있다[2,3].

3. NIST 표준 형태보존암호 FF1, FF3에 대한 신경망 구별자

3.1 데이터 셋

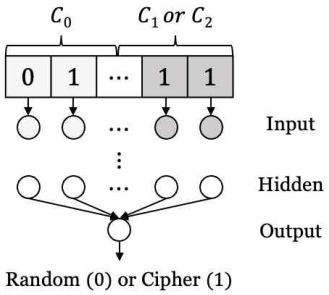
(그림 1)은 신경망 구별자의 데이터셋을 생성하는 과정이다. 먼저, 임의의 랜덤 평문 P_0, P_1을 생성 후 δ (입력 차분)을 만족하는 평문 쌍을 생성하기 위해 P_0에 입력차분을 XOR 하여 평문 P_2을 구한다. 그 후, 각 평문 P_0, P_1, P_2을 암호화하여 암호문 C_0, C_1, C_2을 구한다. 이때 C_0와 C_1은 차분 관계가 아닌 랜덤 평문을 암호화한 결과이므로, 두 값을 연접한 결과를 0으로 라벨링한다. 반면 C_0와 C_2는 입력 차분을 만족하는 평문의 암호문으로 특정 확률로 출력 차분을 만족하는 암호 데이터이다. 따라서, C_0와 C_2를 연접한 값은 1로 라벨링 한다. 암호화 과정에서 사용되는 평문 및 암호문은 숫자 (0 ~ 9) 또는 소문자 (a ~ z) 도메인에서 선택되고, 실제 데이터 셋에는 C_0, C_1, C_2의 비트 값이 저장된다.



(그림 1) 데이터 셋 생성 과정

3.2 모델 구성

NIST 표준 형태보존 암호 FF1, FF3에 대한 신경망 구별자의 전체적인 구조는 (그림 2)와 같다. 랜덤 또는 차분 암호문 쌍의 각 비트는 입력 레이어의 각 뉴런에 할당된다. 이후, 히든 레이어를 거치고 출력 레이어에서 sigmoid 활성화 함수를 거쳐 0 ~ 1 사이의 값을 얻어낸 뒤 해당 값과 실제 정답 (0 또는 1)의 손실을 계산한다. 이러한 과정을 통해 입력 데이터에 대해 올바른 구별이 가능하도록 학습을 진행하면 FF1, FF3에 대한 신경망 구별자로서 동작할 수 있게 된다.



(그림 2) 신경망 구별자의 구조

<표 1>은 FF1, FF3에 대한 신경망 구별자의 하이퍼파라미터를 나타낸 것이다. Epoch은 FF1이 FF3 보다 더 많은 라운드를 가지기 때문에 각각 20과 15로 설정하였다. 레이어는 Dense 레이어를 사용하였다. 또한, 신경망 구별자는 차분을 갖는 데이터와 랜덤 데이터를 구별해야하므로 이진 분류를 수행한다. 따라서 손실함수로는 “binary_crossentropy”를 사용한다. 최적화 함수는 일반적으로 성능이 우수한 Adam을 사용하였다. 이때, 최적화 함수의 학습률 (Learning rate)를 조절하여 더욱 정교한 학습을 하기 위해 학습률을 0.001에서 시작하여 0.0001까지 감소하도록 하였다.

<표 1> Hyperparameters of the proposed neura distinguisher for FF1, FF3

Format-Preserving Encryption	FF1	FF3
Epoch	20	15
Hidden layers	5 hidden layers with 64 units	4 hidden layers with 128 units
parameters	173,956	74,497
Batch size	32	
Activation	ReLu (Hidden). Sigmoid (Output)	
Optimizer (Learning rate)	Adam(lr = 0.0001~0.001)	
Loss function	binary_crossentropy	

4. 실험 및 성능평가

본 실험은 Ubuntu 20.04.5 LTS와 Tesla T4 (GPU) 12GB RAM를 지원하는 클라우드 컴퓨팅 플랫폼 Google colab에서 수행되었다. 프로그래밍 환경으로는 tensorflow 2.12.0 및 Python 3.9.16를 사용하였다.

4.1 FF1의 실험결과

<표 2>는 FF1 입력 차분에 따른 정확도를 보여준다. 숫자 도메인에서 최대 10라운드까지 데이터를 구분할 수 있으며 $0F$ 차분에서 0.85의 가장 높은 정확도 및 신뢰도 (Reliability=Ts - 0.5 (이진 분류의 랜덤 확률))을 달성하였다. 소문자 도메인에서는 평문과 암호문의 경우의 수가 증가하므로 최대 2라운드까지 데이터를 구분할 수 있으며 03, 09, $0E$, $0F$에 대해 0.52의 정확도로 숫자 도메인에 비해 낮지만 구별자로서의 유효한 정확도를 달성하였다. 그러나 다른 차분을 사용할 경우 $0F$에 비해 두 도메인에서 상대적으로 낮은 정확도 및 신뢰도가 도출된다. 본 실험을 통해 FF1에서는 $0F$가 숫자와 소문자에서 공통적으로 가장 좋은 차분임을 알 수 있다.

<표 2> Result of FF1 according to input difference (Tr, Val, Ts : Accuracy, Rel : Reliability).

	Number (10-round)				Lowercase (2-round)			
	Tr	*Val*	*Ts*	*Rel*	*Tr*	*Val*	*Ts*	*Rel*
01	0.73	0.74	0.73	0.23	0.50	0.50	0.50	0.00
02	0.74	0.75	0.74	0.24	0.51	0.51	0.51	0.01
03	0.71	0.71	0.71	0.21	0.52	0.51	0.52	0.02
04	0.75	0.75	0.75	0.25	0.51	0.51	0.51	0.01
05	0.75	0.75	0.75	0.25	0.51	0.51	0.51	0.01
06	0.75	0.75	0.75	0.25	0.51	0.51	0.51	0.01
07	0.75	0.75	0.75	0.25	0.51	0.51	0.51	0.01
08	0.80	0.80	0.80	0.30	0.51	0.50	0.51	0.01
09	0.84	0.83	0.84	0.34	0.52	0.52	0.52	0.02
0A	0.84	0.84	0.84	0.34	0.50	0.50	0.50	0.00
0B	0.82	0.82	0.82	0.32	0.51	0.51	0.51	0.01
0C	0.85	0.84	0.85	0.35	0.5	0.5	0.5	0.00
0D	0.78	0.78	0.78	0.28	0.51	0.51	0.51	0.01
0E	0.81	0.81	0.81	0.31	0.52	0.52	0.52	0.02
0F	0.85	0.85	0.85	0.35	0.52	0.52	0.52	0.02

4.2 FF3의 실험결과

<표 3>는 FF3 입력 차분에 따른 정확도를 보여준다. 숫자 도메인에서 최대 8라운드까지 데이터를 구분할 수 있으며 08 차분에서 0.98의 가장 높은 정확도 및 신뢰도를 달성하였다. 소문자 도메인에서는 앞의 FF1과 마찬가지로 평문과 암호문의 경우의 수가 증가하므로 최대 2라운드까지 데이터를 구분할 수 있으며 08, $0E$에 대해 0.55의 정확도로 숫자 도메인에 비해 낮지만 구별자로서의 유효한 정확도를 달성하

였다. 그러나 다른 차분을 사용할 경우 08에 비해 두 도메인에서 상대적으로 낮은 정확도 및 신뢰도가 도출된다. 본 실험을 통해 FF3에서는 08이 숫자와 소문자에서 공통적으로 가장 좋은 차분임을 알 수 있다.

<표 3> Result of FF3 according to input difference (Tr, Val, Ts : Accuracy, Rel : Reliability).

	Number (8-round)				Lowercase (2-round)			
	Tr	*Val*	*Ts*	*Rel*	*Tr*	*Val*	*Ts*	*Rel*
01	0.62	0.62	0.62	0.12	0.54	0.54	0.54	0.04
02	0.82	0.82	0.82	0.32	0.55	0.54	0.54	0.04
03	0.78	0.76	0.77	0.27	0.52	0.51	0.51	0.01
04	0.76	0.75	0.75	0.25	0.52	0.52	0.51	0.01
05	0.77	0.75	0.74	0.24	0.53	0.53	0.53	0.03
06	0.75	0.74	0.75	0.25	0.52	0.51	0.52	0.02
07	0.75	0.73	0.74	0.24	0.53	0.52	0.52	0.02
08	0.98	0.97	0.97	0.47	0.55	0.55	0.55	0.05
09	0.96	0.94	0.94	0.44	0.54	0.54	0.54	0.04
0A	0.96	0.95	0.95	0.45	0.53	0.53	0.53	0.03
0B	0.97	0.96	0.96	0.46	0.53	0.52	0.52	0.02
0C	0.97	0.95	0.95	0.45	0.53	0.53	0.53	0.03
0D	0.96	0.96	0.96	0.46	0.53	0.52	0.51	0.01
0E	0.96	0.96	0.96	0.46	0.54	0.54	0.55	0.05
0F	0.96	0.93	0.94	0.44	0.52	0.52	0.52	0.02

4. 결 론

본 논문에서는 NIST 표준 형태보존암호인 FF1, FF3-1을 위한 단일 차분을 사용한 최초의 신경 구별자를 제안하였다. FF1은 차분으로 $0F$를 사용할 때, 숫자 및 소문자 도메인에서 차분 데이터 구별에 성공하였다 (정확도는 각각 0.85 및 0.52). FF3-1에서는 08을 사용할 때, 숫자 및 소문자 도메인에서 차분 데이터 구별에 성공하였다 (정확도는 각각 0.98 및 0.55). 향후에는 형태 보존 암호에 양자 인공지능을 적용하여 양자 컴퓨터 상에서 동작 가능한 신경망 구별자를 구현할 예정이다.

5. Acknowledgment

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No.2022-0-00627, Development of Lightweight BIoT technology for Highly Constrained Devices, 50%) and this work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No.2021-0-00540, Development of Fast Design and Implementation of Cryptographic Algorithms based on GPU/ASIC, 50%).

참고문헌

[1] Dworkin, Morris. "Recommendation for block cipher modes of operation: methods for format-preserving encryption." NIST Special Publication 800 (2016): 38G.

[2] Gohr, Aron. "Improving attacks on round-reduced speck32/64 using deep learning," Annual International Cryptology Conference. Springer, Cham, 2019.

[3] Baksi, Anubhab. "Machine learning-assisted differential distinguishers for lightweight ciphers," Classical and Physical Security of Symmetric Key Cryptographic Algorithms. Springer, Singapore, 141-162, 2022.