Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.29, NO.3, Jun. 2019 ISSN 2288-2715(Online)
https://doi.org/10.13089/JKIISC.2019.29.3.491

비프로파일링 기반 전력 분석의 성능 향상을 위한 오토인코더 기반 잡음 제거 기술*

권 동 근,† 진 성 현, 김 희 석,‡ 홍 석 희
고려대학교

Improving Non-Profiled Side-Channel Analysis Using Auto-Encoder Based Noise Reduction Preprocessing*

Donggeun Kwon,† Sunghyun Jin, HeeSeok Kim,‡ Seokhie Hong
Korea University

요 약

최근 보안 디바이스의 물리적 취약성을 찾을 수 있는 부채널 분석 분야에서 딥러닝을 활용한 연구가 활발히 진행되고 있다. 하지만, 최신 딥러닝 기반 부채널 분석 기술 연구는 템플릿 공격 등과 같은 프로파일링 기반 부채널 분석 환경에서 파형을 옳게 분류하기 위한 연구에 집중되어 있다. 본 논문에서는 이전 연구들과 다르게 딥러닝을 신호 전처리 기법으로 활용하여 차분 전력 분석, 상관 전력 분석 등과 같은 논프로파일링 기반 부채널 분석의 성능을 고도화 할 수 있는 방법을 제안한다. 제안기법은 오토인코더를 부채널 분석 환경에 적합하게 변경하여 부채널 정보의 노이즈를 제거하는 전처리 기법으로, 기존 노이즈 제거 오토인코더는 임의로 추가한 노이즈에 대한 학습을 하였다면 제안하는 기법은 노이즈가 제거된 라벨을 사용하여 실제 데이터의 노이즈를 학습한다. 제안기법은 논프로파일링 환경에서 수행 가능한 전처리 기법이며 하나의 뉴런 네트워크의 학습만을 통해 수행할 수 있다. 본 논문에서는 실험을 통해 제안기법의 노이즈 제거 성능을 입증하였으며, 주성분분석 및 선형판별분석과 같은 기존 전처리 기법들과 비교하여 우수하다는 것을 보인다.

ABSTRACT

In side-channel analysis, which exploit physical leakage from a cryptographic device, deep learning based attack has been significantly interested in recent years. However, most of the state-of-the-art methods have been focused on classifying side-channel information in a profiled scenario where attackers can obtain label of training data. In this paper, we propose a new method based on deep learning to improve non-profiling side-channel attack such as Differential Power Analysis and Correlation Power Analysis. The proposed method is a signal preprocessing technique that reduces the noise in a trace by modifying Auto-Encoder framework to the context of side-channel analysis. Previous work on Denoising Auto-Encoder was trained through randomly added noise by an attacker. In this paper, the proposed model trains Auto-Encoder through the noise from real data using the noise-reduced-label. Also, the proposed method permits to perform non-profiled attack by training only a single neural network. We validate the performance of the noise reduction of the proposed method on real traces collected from ChipWhisperer board. We demonstrate that the proposed method outperforms classic preprocessing methods such as Principal Component Analysis and Linear Discriminant Analysis.

Keywords: Side-Channel Analysis, Non-Profiled Attack, Deep Learning, Auto-Encoder, Preprocessing

Received(03. 11. 2019), Modified(05. 02. 2019), Accepted(05. 02. 2019)

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

† 주저자, donggeun.kwon@gmail.com
‡ 교신저자, 80khs@korea.ac.kr(Corresponding author)

I. 서 론

사물 인터넷 기술의 보편화와 함께 물리적 장비가 노출된 환경에서의 보안이 중요해지면서, 암호의 수학적 안전성뿐만 아니라 물리적인 공격에 대한 안전성 또한 필요하게 되었다. 물리적 공격 중 하나인 부채널 분석(Side-Channel Analysis)이란 AES(Advanced Encryption Standard)와 같이 이론적으로 안전성이 검증된 암호알고리즘이 실제 장비에서 동작할 때 누출되는 부채널 정보를 이용하여 암호 시스템을 분석하는 기법이다. 부채널 정보로는 장비가 소모하는 전력, 전자기파, 연산 시간 등이 있다.

부채널 분석은 공격 환경에 따라 프로파일링 공격(Profiling Attack)과 논프로파일링 공격(Non-Profiling Attack)으로 나누어진다. 이중 프로파일링 공격은 고정된 비밀키가 있는 공격 대상의 타겟 장비와 동일한 장비인 프로파일링 장비를 가지고 있을 때 수행 가능한 공격으로 템플릿 공격(Template Attack)[1], 스토캐스틱 공격(Stochastic Attack)[2] 등이 있다. 공격자는 프로파일링 장비를 이용하여 장비에 대한 취약점을 특징화시킨 후, 수집한 정보를 이용하여 파형을 분석한다. 이와 반대로 논프로파일링 공격은 프로파일링 장비가 없는 환경에서의 부채널 분석 기법으로, 타겟 장비만을 이용하여 부채널 정보를 수집하고 이를 이용하여 통계적인 기법을 통해 비밀정보를 분석한다. 따라서 사전에 계산된 템플릿 없이 타겟 장비로부터 수집한 부채널 정보만을 이용하여 비밀키를 복구해야 하며, 이러한 공격으로는 차분 전력 분석(Differential Power Analysis, DPA)[3], 상관 전력 분석(Correlation Power Analysis, CPA)[4] 등이 있다.

딥러닝(Deep Learning)은 하드웨어의 발전과 학습 알고리즘의 개선으로 인해 여러 분야에서 향상된 결과를 보여주면서, 최근에는 부채널 분석 분야에서도 분석 절차 단순화 등을 위하여 딥러닝을 이용한 연구가 활발히 진행되고 있다. 초기 연구는 전력 모델 특징화(Leakage Characterization, Power Model Characterization)를 시도하는 회귀 문제에 관한 연구도 있었지만[5], 이후 분류 문제를 해결하는 방법으로써 딥러닝을 이용하는 연구가 주로 이루어졌다. 이는 프로파일링 공격 시나리오를 가정하여 공격자는 프로파일링 장비를 통해 뉴런 네트워크를 학습시킨 후, 타겟 장비로부터 획득한 파형의 중간값을 분류하는 방법으로 딥러닝을 이용한다. Maghrebi et al[6]의 연구 결과를 통하여 딥러닝 기반의 프로파일링 공격은 마스킹 대응기법 적용 여부와 관계없이 분석이 가능하다는 것이 알려졌다. 이후 Cagli et al[7]의 연구를 통해 CNN(Convolutional Neural Network)을 이용한다면 하이딩 대응기법이 적용된 구현에 대해서도 시점 정렬과 같은 전처리 과정 수행 없이 딥러닝 기반의 부채널 분석만을 통해 비밀키를 복원할 수 있다는 것이 밝혀졌다. 최근 Timon[8]이 제안한 방법으로 딥러닝을 구별자(Distinguisher)로 이용하는 분석 기법은 데이터 라벨 분류에 따라 뉴런 네트워크의 학습 결과가 손실(Loss), 정답률(Accuracy), 가중치(Weight)의 측면에서 다르게 나타난다는 사실을 이용하여 비밀정보를 추정한다. 해당 연구를 통해 논프로파일링 공격 시나리오에서도 딥러닝 기반의 부채널 분석이 수행 가능하다는 것을 보여주었다.

하지만 프로파일링 기반의 딥러닝을 이용하는 방법은 학습 데이터에 대한 올바른 라벨을 알고 있어야 뉴런 네트워크를 학습할 수 있다는 지도 학습의 한계점에서 의해, 학습 데이터와 그에 대한 라벨을 획득할 수 있는 프로파일링 환경에서의 연구로 제한되어 진행되고 있다. 따라서 파형에 대한 올바른 중간값을 얻기 힘든 논프로파일링 공격 환경에서는 수행이 힘들다는 단점이 존재한다. Timon[8]의 분석 기법은 논프로파일링 기반이지만 비밀키를 추측하여 중간값을 계산하여 위와 유사하게 수행하는 기법으로, 결국 데이터의 라벨로 중간값이 필요하며 추측한 키만큼의 뉴런 네트워크 모델 학습이 요구된다.

본 논문에서는 논프로파일링 공격 환경에서도 수행 가능한 딥러닝 기반의 노이즈 제거 전처리 기법을 최초로 제안한다. 제안기법은 기존에 알려진 노이즈 제거 오토인코더를 부채널 분석 환경에 적합하게끔 실제 파형을 학습 데이터로, 노이즈 제거 전처리한 파형을 라벨로 사용함으로써 노이즈 제거 오토인코더의 문제점을 극복함과 동시에 실제 노이즈에 대한 제거 효과를 기대할 수 있다. 제안기법을 이용할 경우, 이전 연구들과는 다르게 파형의 중간값을 모르더라도 단일 뉴런 네트워크에 대한 학습을 통해서 논프로파일링 공격을 수행할 수 있다. 또한, 기존 연구인 딥러닝 전력 분석은 딥러닝의 학습 경향을 구별자로 이용하는 새로운 방법을 제시하였지만, 기존 분석 기법과 성능을 비교 및 기존 기법 적용이 힘들다. 제안기법은 딥러닝 기반의 부채널 분석의 장점 중 하나인 전처리 과정을 편리하게 수행할 수 있다는 점을 유지

하면서, CPA와 같은 기존 부채널 분석 기법을 함께 이용할 수 있다는 장점이 있다. 실험을 통해 본 논문에서 제안한 기법이 노이즈 제거에 유의미한 성능을 가진다는 것을 보이며, 기존 제안된 주성분분석(Principal Component Analysis, PCA)과 선형판별분석(Linear Discriminant Analysis, LDA)과 같은 전처리 기법들과도 비교하여 좋은 성능을 낸다는 것을 보인다.

본 논문의 구성은 다음과 같다. 2장에서는 제안기법을 이해하기 위한 배경지식인 논프로파일링 공격, 딥러닝과 오토인코더 및 노이즈 제거 오토인코더에 대해 소개한다. 3장에서 오토인코더를 이용한 부채널 파형 노이즈 제거 전처리 기법을 새롭게 제안하며, 4장에서 실험을 통해 기존에 알려진 전처리 기법들과 제안기법 사이의 성능을 비교한다. 마지막 5장에서는 결론과 향후 연구 방향을 제시하면서 마친다.

II. 관련 연구

2.1 논프로파일링 공격(Non-Profiled Attack)

논프로파일링 공격이란 프로파일링 장비없이 타겟 장비로부터만 전력 파형을 수집 가능한 환경에서의 부채널 분석 기법이며, 대표적으로 차분 전력 분석과 상관 전력 분석이 있다. 그중에서 2004년에 Brier et al[4]이 제안한 상관 전력 분석은 암호 연산을 수행하는 장비가 발생하는 소비 전력(P)과 연산하는 중간값($data$) 사이의 상관관계를 이용하여 분석하는 통계적인 기법이다. 일반적으로 전력 소비 모델은 다음 수식 (1)과 같이 정의한다[9]. 여기서 δ은 고정된 상수 오프셋(offset), $HW(\bullet)$는 해밍 웨이트(Hamming Weight) 함수, $Noise$는 $N(0, \sigma^2)$의 정규 분포를 따르는 랜덤 노이즈를 의미한다.

$$P = \delta + HW(data) + Noise \tag{1}$$

수식 (1)의 의미는 장비가 소모하는 전력은 연산되는 데이터의 해밍웨이트와 상관관계가 있다는 것이다. 상관 전력 분석은 먼저 장비가 비밀정보와 관련된 연산을 하고 있을 때 소모하는 전력을 측정하고, 해당 연산의 결과인 중간값을 추측하여 두 값 사이의 상관계수(Correlation Coefficient)를 구한다. 이때 가장 큰 상관계수의 추측 중간값이 올바른 중간값임을 유추할 수 있고, 중간값과 평문으로 비밀키를 복원할 수 있다. 이때 두 값 사이의 상관계수는 수식 (2)와 같다.

$$\rho(X, Y) = \frac{Cov[X, Y]}{\sqrt{Var[X]\ Var[Y]}} = \frac{\sum(X_i - \overline{X})(Y_i - \overline{Y})}{\sqrt{\sum(X_i - \overline{X})^2 \sum(Y_i - \overline{Y})^2}} \tag{2}$$

2.2 딥러닝(Deep Learning)

딥러닝이란 뉴런 네트워크를 이용하여 임의의 함수를 근사하는 기계학습의 한 분야로 이미지, 자연어, 음성 처리 등 여러 분야에서 사용되고 있다[10]. 딥러닝에서 학습은 뉴런 네트워크를 어떠한 함수로 근사하기 위해 파라미터를 수정하는 과정이며, 이에 사용되는 학습 데이터에 따라 데이터의 라벨(Label)이 주어진 경우에는 지도 학습(Supervised Learning), 그렇지 않은 경우는 비지도 학습(Unsupervised Learning)으로 구분한다.

딥러닝을 통해 근사하고자 하는 함수 $f^*(\bullet)$를 나타내기 위한 뉴런 네트워크가 $f(\bullet)$이고, θ가 해당 뉴런 네트워크를 구성하는 파라미터일 때, $f(X;\theta)$는 입력 X에 대한 뉴런 네트워크의 출력을 나타낸다. 이때 출력 $f(X;\theta)$에 대한 값이 라벨 $Y(= f^*(X))$와 유사하도록 파라미터를 조정하는 과정을 학습이라고 부른다. 학습은 손실(Loss, 또는 비용, Cost)을 최소화하는 과정과 동일하며, 뉴런 네트워크의 손실은 데이터를 뉴런 네트워크에 입력하여 얻어낸 출력과 라벨 사이의 차이를 의미하며, 수식 (3)과 같다.

$$Loss(X, Y) = L(f(X;\theta), Y) \tag{3}$$

$$\theta^* = \arg\min_{\theta} L(f(X;\theta), Y) \tag{4}$$

수식 (3)에서 함수 $L(\bullet)$는 손실 함수(Loss Function, 또는 비용 함수, Cost Function)를 의미하며, 크로스 엔트로피(Cross Entropy), 평균 제곱 오차(Mean Squared Error) 등이 있다. 뉴런 네트워크는 학습 과정을 통하여 수식 (4)를 만족하는 뉴런 네트워크의 파라미터 θ^*를 얻게 된다. 파라미터를 변경하는 학습은 SGD(Stochastic Gradie

nt Descent) 알고리즘을 기반으로 제안된 RMSprop, Adam 등의 여러 최적화기(optimizer)를 이용하며, 여러 번의 파라미터 조정 과정을 통해 학습이 이루어진다[10].

뉴런 네트워크는 여러 함수들로 구성되며 이러한 함수를 층(layer)라고 부른다. 뉴런 네트워크의 층은 데이터의 입력에 해당하는 입력층(input layer), 네트워크의 출력에 해당하는 출력층(output layer)과 그 외 나머지 은닉층(hidden layer)으로 크게 3가지로 구분된다. 입력층과 출력층은 각각 입력 데이터와 출력 데이터의 차원 크기와 일치하는 뉴런의 수를 가지며 학습 데이터에 따라 고정된다. 은닉층의 경우에는 층의 개수, 종류, 구성하는 뉴런의 수 등의 공격자가 설정해야 하는 파라미터가 존재하며, 이처럼 학습을 통해 조정되는 파라미터가 아닌 파라미터를 하이퍼 파라미터라고 부른다. 공격자는 하이퍼 파라미터들을 조절하여 뉴런 네트워크가 원하는 함수를 근사하게끔 만들 수 있으며, 특히 은닉층의 구조에 따라 MLP(Multi-Layer Perceptron), CNN, RNN(Recurrent Neural Network) 등으로 구분한다.

2.3 다층 퍼셉트론(Multi-Layer Perceptron)

MLP는 기본적인 뉴런 네트워크 모델이며 DNN(Deep Neural Network), ANN(Artificial Neural Network) 등 여러 명칭으로 불린다. MLP에서 뉴런 네트워크의 각 은닉층은 선형 함수와 비선형 함수로 구성된다. 선형 함수 $\lambda(\bullet)$를 사용하는 층을 Fully- connected layer(FC)라고 하며 입력 X에 대해 가중치(W)와 바이어스(b)와의 연산 $WX+b$를 수행한다. 활성 함수(Activation Function)라고 부르는 비선형 함수 $\sigma(\bullet)$를 사용하는 층을 Activation layer라고 하며 Sigmoid, ReLU(Rectified Linear Unit), SELU(Scaled Exponential Linear Unit) 등의 함수를 사용한다. 주로 두 종류의 층을 합친 함수($\sigma \circ \lambda(\bullet)$)를 은닉층이라고 부른다. Universal approximation theorem에 의해 은닉층을 가지는 뉴런 네트워크는 임의의 연속 함수를 근사할 수 있다는 것이 알려졌으며[11], 딥러닝 기반의 부채널 분석에서는 MLP와 같은 뉴런 네트워크를 이용하여 파형을 입력으로 중간값(또는 해밍 웨이트)을 출력하는 함수를 근사하는 연구가 진행되고 있다.

2.4 오토인코더(Auto-Encoder)

오토인코더란 딥러닝 알고리즘 중 대표적인 비지도 학습으로 뉴런 네트워크의 출력을 입력과 유사하도록 학습한다. 즉 출력 $f(X;\theta)$가 라벨이 아닌 입력 X가 되도록 가중치를 조정한다. 오토인코더는 데이터를 압축하거나, 뉴런 네트워크의 사전 학습, 데이터의 노이즈 제거 등의 목적을 위해 사용된다.

초기의 오토인코더는 뉴런 네트워크의 가중치 초기화를 위한 사전 학습(Pre-training)의 용도로 사용되었다. 뉴런 네트워크의 각 레이어의 가중치가 입력 데이터를 가장 잘 표현할 수 있도록 오토인코더를 이용해 사전 학습하여 가중치를 초기화한 이후, Fine-tuning을 통하여 전체적인 가중치를 조정하는 방식으로 네트워크를 학습하였다. 사전 학습이 각 레이어에 대한 가중치 학습이라면, Fine-tuning은 모든 레이어들에 대한 가중치 학습으로 전체 네트워크가 잘 동작할 수 있도록 학습한다. 이러한 가중치 초기화 방식을 Stacked Auto-Encoder(SAE) 라고 부른다. Maghrebi et al[6]에 의해 부채널 분석에서도 사용된 연구 결과가 존재하며, 해당 논문의 실험 결과에 따르면 MLP보다 좋은 성능을 내는 경우가 확인되었다.

다음으로 오토인코더는 데이터의 차원을 압축하는 방법으로 사용할 수 있다. 차원 압축을 목적으로 사용되는 오토인코더는 기본적으로 입력 데이터의 차원을 압축하는 인코더(Encoder) 부분과 인코더를 통하여 압축된 데이터를 원본의 입력 데이터로 다시 재구성하는 디코더(Decoder) 부분으로 이루어져 있다. Fig.1.은 오토인코더의 기본적인 네트워크 구조를 나타낸 그림이다.

그림 Fig.1.에서 $x_1, x_2, \ldots, x_n$는 오토인코더의 입

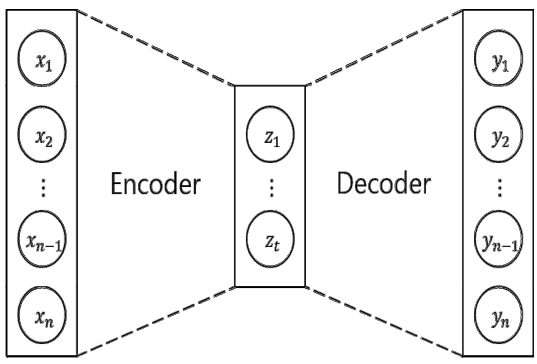

Fig. 1. Basic Architecture of Auto-Encoder

력 데이터, $z_1, z_2, \ldots, z_t$는 코드(Code)라고 부르는 오토인코더의 인코더를 이용하여 압축한 데이터이며, $y_1, y_2, \cdots, y_n$는 뉴런 네트워크의 출력 데이터이다. 입력과 Code 사이의 은닉층들로 구성된 뉴런 네트워크를 인코더라고 부르며, Code와 출력 사이의 은닉층들로 구성되는 뉴런 네트워크는 디코더라고 부른다. 인코더와 디코더가 각각 한 개의 은닉층으로 구성되는 오토인코더의 연산은 수식 (5,6)으로 나타낼 수 있다. 일반적으로 Code의 차원인 t는 입력 데이터의 차원 n보다 작은 값을 사용하며 이러한 오토인코더를 Undercomplete Auto-Encoder라고도 부른다. 반대로 t가 n보다 큰 오토인코더는 Overcomplete Auto-Encoder라고 부른다.

$$z_i = \sigma(\sum_{j=1}^{n} Weight_{(j,i)}^{enc} x_j + bias_i^{enc}) \tag{5}$$

$$y_i = \sum_{j=1}^{t} Weight_{(j,i)}^{dec} z_j + bias_i^{dec} \tag{6}$$

$$\theta^* = \arg\min_{\theta} L(g(f(X;\theta^{enc});\theta^{dec}),\ Y) \tag{7}$$

오토인코더는 학습을 통하여 입력 데이터(X)와 출력 데이터(Y)의 차이가 가장 작아지는 파라미터(Weight, Bias)를 탐색하게 된다. 다시말해 오토인코더의 인코더 함수가 $f(\bullet)$이며, 디코더가 $g(\bullet)$일 때, 수식 (7)를 만족하는 파라미터를 학습하게 된다. 만약 학습이 성공적으로 수행되어 각각의 입력에 대해 출력이 동일하다면, 즉 $X = Y$를 만족한다면 $X = g(f(X)) = g(Z)$가 성립한다. 이것은 오토인코더를 통한 압축 데이터(Z, Code)는 입력 데이터보다 작은 차원의 데이터이면서 어떠한 함수 $g(\bullet)$를 통해 원래 데이터로 복원 가능하다는 의미이다. 따라서 오토인코더를 이용하여 원본 데이터(X)의 특징을 모두 가지고 있으면서도 낮은 차원의 데이터(Z)를 획득할 수 있다. 디코더의 활성 함수는 선형 함수이면서 $L(\bullet)$은 MSE를 사용할 경우 주성분분석과 동일한 부분공간(Subspace)을 생성(Span)한다고 알려져 있다[10].

2.5 노이즈 제거 오토인코더(Denoising Auto-Encoder)

오토인코더를 기반으로 하여 노이즈 제거 효과를 줄 수 있는 뉴런 네트워크가 Vincent et al[12]에 의해 제안되었다. 노이즈 제거 오토인코더(DAE)는 오토인코더와 네트워크의 구조는 동일하지만, 학습에 사용되는 데이터에서 차이가 존재한다. 간략하게 요약하면, 입력 X를 그대로 사용하는 오토인코더와 다르게 DAE의 경우에는 학습 데이터에 노이즈를 추가하여 노이즈가 들어간 데이터 $\tilde{X}$를 뉴런 네트워크의 입력으로 사용한다. Fig.2.는 DAE의 기본적인 구조를 나타낸 것이다.

원본 데이터 X가 주어졌을 때, Fig.2.와 같이 DAE에서는 데이터 X에 노이즈를 추가하여 새로운 노이즈 데이터(Noise Trace $\tilde{X}$)를 생성하여 뉴런 네트워크의 입력으로 사용한다. 해당 데이터 $\tilde{X}$를 뉴런 네트워크의 입력으로 하여 얻어낸 출력 데이터(Output Y)와 노이즈를 추가하기 전의 원본 데이터(X)를 라벨로 사용하여 손실을 계산하고, 손실을 최소화하도록 가중치를 학습한다. 수식 (8)은 DAE의 손실을 나타낸다. 다시 말해 DAE는 공격자가 임의로 추가한 노이즈를 제거하여 원본 데이터를 복원할 수 있도록 뉴런 네트워크를 학습한다. 이러한 오토인코더에 대한 학습 전략을 통하여 노이즈를 제거하는 뉴런 네트워크를 만들 수 있다.

$$Loss_{DAE} = L(g(f(\tilde{X};\theta)), X) \tag{8}$$

DAE에서 입력 데이터에 노이즈를 추가하는 방법으로는 크게 2가지로 데이터에 가우시안 노이즈를

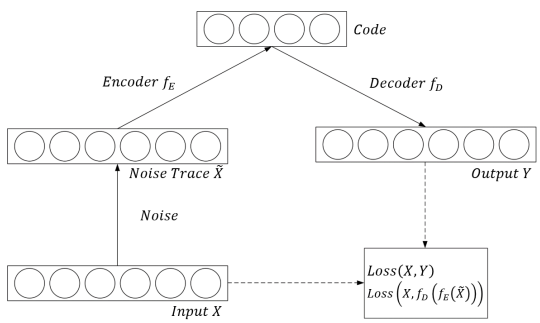

Fig. 2. Basic Architecture of Denoising Auto-Encoder

추가하거나, 데이터의 몇몇 원소들을 0으로 만드는 방법이 있다. 랜덤 노이즈를 추가함으로써 근접한 범위에 있는 데이터를 동일한 데이터로 판단할 수 있게 끔 하며, 몇몇 원소들을 0으로 만듦으로써 데이터의 특정 부분에 집중하여 학습하는 것이 아닌 데이터 전체에 대해 학습을 할 수 있도록 한다. 이러한 방법을 통하여 노이즈가 제거된 데이터를 출력하도록 뉴런 네트워크를 설계할 수 있다.

III. Side-Channel Preprocess based on Deep Learning

3.1 Side-Channel Auto-Encoder

이미지 처리 분야와의 차이점으로 인해 앞서 설명한 DAE를 부채널 분석 환경에서 사용하기에는 2가지 단점이 존재한다. 첫 번째로 만약 수집한 파형에 노이즈를 추가하여 학습 데이터로 사용할 경우, 이미 수집 파형에 존재하는 노이즈에 새로운 노이즈를 추가적으로 더하게 되는 것이며, 이로인해 클래스 구분이 더욱 힘들어져 학습에 실패할 수 있다. 전력 모델을 앞서 소개한 내용과 같이 수식 (1)이라고 가정할 때, 수식 (9)는 입력 데이터를 전력 파형으로 하였을 때의 DAE의 손실을 계산한 것이다.

$$\begin{aligned} Loss_{DAE} &= L(g(f(\widetilde{X};\theta)), X) \\ &= L(g(f(\delta + HW(data) + Noise + Noise')) \\ &\quad , \delta + HW(data) + Noise) \end{aligned} \tag{9}$$

여기서 노이즈인 $Noise$의 값이 작은 경우에는 해당 파형에 대해서 노이즈 제거의 필요성이 크지 않다. 반대로 $Noise$의 값이 큰 경우, 새로운 노이즈인 $Noise'$를 추가하여 계산된 학습 데이터 $\delta + HW(data) + Noise + Noise'$에서 노이즈의 비중이 이전보다 더욱 커져 학습의 난이도가 증가하게 된다 이때 $Noise'$를 너무 작게 설정하여 학습하게 된다면, 오토인코더가 작은 노이즈에 대해서만 학습을 하게 되어 노이즈 제거의 효과는 줄어들게 된다.

또한 부채널 분석 환경에서는 공격의 대상이 되는 연산은 어떤 특정 시점에서만 수행된다. 따라서 랜덤한 시점을 0으로 초기화하여 학습 데이터를 생성할 경우, 공격의 타겟이 되는 정보가 제외된 학습 데이터가 생길 수 있으므로 해당 노이즈 추가 방법은 부채널 분석 환경에 적합하지 않다.

따라서 이러한 환경의 차이로 인해 DAE를 쉽게 적용하기 힘들다. 본 논문에서는 이러한 문제점들을 해결하기 위해서 기존에 제안된 구조를 변경하여 설계한 새로운 오토인코더 모델을 제안한다.

Fig.3.은 본 논문에서 제안하는 뉴런 네트워크의 기본 구조이다. 제안 모델은 오토인코더의 기본 구조를 따라가지만, DAE와 다르게 입력 데이터를 그대로 학습 데이터로 사용한다. 또한, 전처리 기법을 이용하여 처리한 데이터를 라벨로 사용함으로써 뉴런 네트워크가 실제 노이즈를 제거한 데이터를 출력할 수 있도록 네트워크의 가중치를 학습한다. Fig.3.과 같이, 뉴런 네트워크의 입력으로 입력 데이터(Input X)를 사용하고 그에 대한 출력 데이터(Output Y)와 전처리된 데이터(Denoise Trace $\hat{X}$)와 손실을 계산하여 가중치를 학습한다. 이와 같은 제안한 오토인코더의 손실을 수식으로 나타내면 수식 (10)과 같다.

$$\begin{aligned} Loss_{SCAE} &= L(g(f(X;\theta)), \hat{X}) \\ &= L(g(f(\delta + HW(data) + Noise)) \\ &\quad , \delta + HW(data)) \end{aligned} \tag{10}$$

기존 DAE에서는 새롭게 추가한 노이즈를 제거하도록 뉴런 네트워크를 학습하였지만, 제안한 기법에서는 수집한 전력 파형에 존재하는 노이즈를 제거하도록 학습한다. DAE 손실(9)과 다르게 원본 파형 $\delta + HW(data) + Noise$를 입력으로 오토인코더를 통해 얻은 $g(f(\delta + HW(data) + Noise))$와 노이즈를 제거한 데이터 $\delta + HW(data)$ 사이의 차이를 계산한다. 이러한 전처리 과정 이후에는 잘 알려진 차분 전

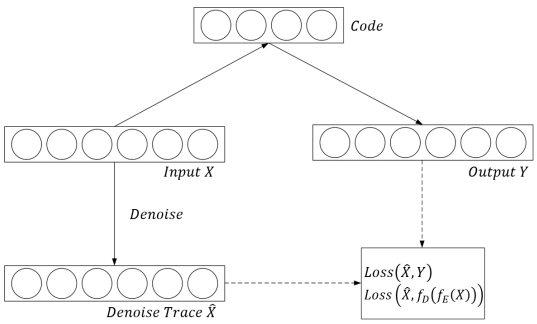

Fig. 3. Basic Architecture of Proposal Auto-Encoder

력 분석 또는 상관 전력 분석과 같은 기존 부채널 분석 기법을 이용하여 비밀정보를 복원할 수 있다.

3.2 평균 파형을 이용한 라벨링

전력 모델이 수식 (1)이라고 하면, 충분히 많은 파형의 평균은 노이즈가 제거된 $\delta + HW(data)$와 유사해진다. 따라서 같은 중간값을 가지는 파형들의 시점별 평균 파형은 노이즈가 제거된 파형이라고 가정할 수 있다. 이러한 방법을 통해 노이즈가 제거된 파형을 중간값마다 1개씩 얻을 수 있다. 중간값별 1개씩 노이즈가 제거된 파형을 얻은 후, 각 학습 데이터(파형)에 대한 라벨을 동일한 중간값에 대한 노이즈가 제거된 데이터(평균 파형)로 결정할 수 있다.

Fig.4.처럼, 같은 중간값을 가지는 파형들에 대해 평균 파형을 계산하여, 해당 중간값을 가지는 파형에 대한 라벨로 사용한다. 예를 들어 오토인코더의 입력 데이터인 어떤 파형의 중간값이 1인 경우, 1을 중간값으로 가지는 파형들의 평균 파형을 해당 데이터의 라벨로 설정한다. 각 중간값에 일치하는 평균파형을 라벨로 설정하여 해당 뉴런 네트워크가 파형에 있는 실제 노이즈를 제거할 수 있도록 할 수 있다.

주성분분석, 선형판별분석 또는 SSA(Singular Spectrum Analysis)와 같은 기존에 제안된 전처리 기법을 사용하여 각 파형에 대한 라벨을 설정하여 제안기법을 수행할 수 있다. 하지만, 기존 연구 결과들을 참고할 때 평균 파형을 최적의 파형으로 가정하는 것은 합리적이며[9], 다른 전처리 기법들과 다르게 어떠한 파라미터도 없이 공격자가 쉽게 얻을 수 있는 파형이다. 또한, 이러한 방법은 선형판별분석과 유사하게 학습 결과가 데이터의 라벨에 의존되도록 유도할 수 있다.

평균 파형을 이용하는 제안기법은 중간값을 알지 못하거나 추측하지 않더라도 수행할 수 있다. 예를 들어, AES와 같은 블록 암호알고리즘은 비밀키가 고정되어 있을 경우에 동일한 평문에 대해 동일한 중간값을 출력하므로, 비밀키를 모르는 때 중간값을 정확하게 알아내지는 못하더라도 같은 평문을 가지는 파형들의 집합은 같은 중간값을 가지는 파형들의 집합이라는 것을 알 수 있다. 따라서 평문으로 분류한 파형에 대한 평균 파형과 올바른 중간값을 이용하여 분류한 파형에 대한 평균 파형은 동일하다는 것을 알 수 있으며, 그러므로 제안기법을 수행함에 있어 중간값은 필요하지 않다. 앞서 명시한 단계에서 중간값이 아닌 평문으로 대체하여도 제안기법을 수행할 수 있다.

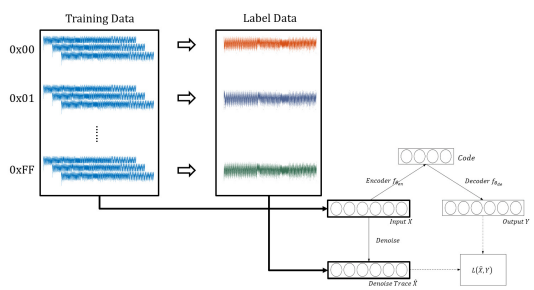

Fig. 4. Labeling of Proposed Auto-Encoder

IV. 실 험

본 장에서는 실험을 통하여 제안기법의 노이즈 제거 성능을 검증하며, 기존에 제안된 다른 전처리 기법들과 성능을 비교한다.

4.1 실험 환경

본 논문의 실험에 사용된 장비의 경우, CPU는 Intel Core i7-8700K, GPU는 NVIDIA GeForce GTX 1080 8GB를 사용하였고, 딥러닝 네트워크 구현을 위해서 TensorFlow "1.10.0"[13]과 Keras "2.1.6-tf"[14]를 이용하였다.

노이즈 제거 성능을 검증하기 위해서 부채널 대응 기법이 적용되지 않은 AES-128 구현에 대한 전력 파형을 사용하였다. ChipWhisperer-Lite를 이용하여 동작 주파수 7.37MHz의 Atmel XMEGA-128(8-bit 프로세서) 보드에서 임의의 평문에 대한 AES 알고리즘 1 Round 연산을 수행할 때, 샘플링 레이트(Sampling rate) 29.538 MS/s으로 클락당 샘플 수(points per cycle) 4로 총 800포인트의 전력 파형 10,000개를 수집하여 실험을 수행하였다. 수집한 전력 파형 9,000개는 학습 데이터(Training Set), 나머지는 검증 데이터(Validation set)로 사용하였다.

4.2 실험 방법 및 하이퍼 파라미터

제안기법의 노이즈 제거의 성능을 검증하기 위하여 전처리 기법에 따른 파형의 신호 대 잡음비(Signal-to-Noise Ratio, SNR)을 계산하여 비교하였다. 파형은 4.1에서 설명한 칩위스퍼에서 수집한 A

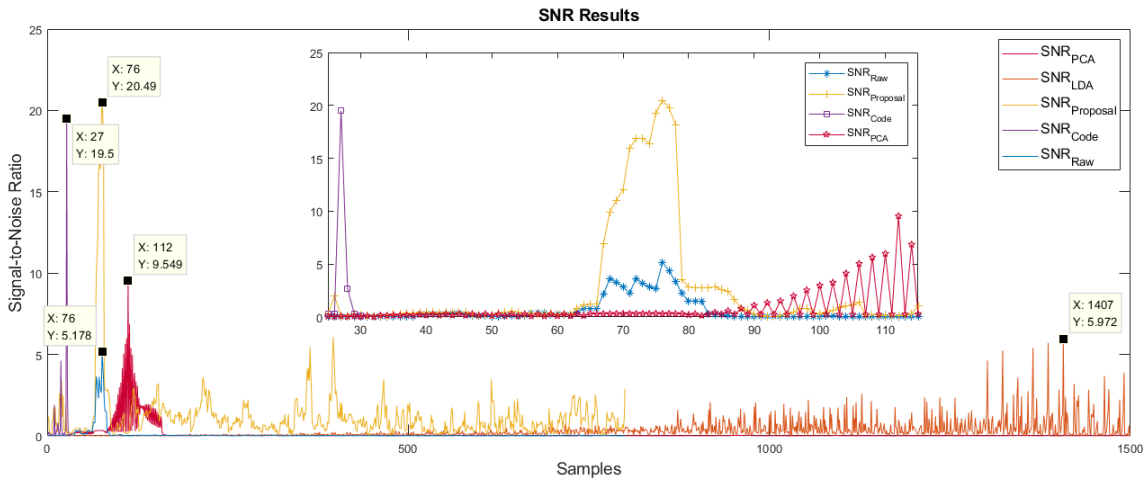

Fig. 5. Results of Signal-to-Noise Ratio

ES 파형를 사용하여 실험을 진행하였다. 실험의 SNR은 [15]을 참고하였으며 데이터는 1라운드 Subbytes 출력의 해밍 웨이트 값을 사용하였다.

제안기법에 사용된 오토인코더의 하이퍼 파라미터의 경우에는 입력과 출력의 크기는 파형의 크기와 동일한 800이며, 은닉층는 1층으로 은닉층의 노드는 30, 은닉층의 활성함수는 ReLU, 출력층의 활성함수는 선형 함수를 사용하였다. 가중치는 He normal initialization[16] 방법으로, 바이어스는 0으로 초기화하였고, 초기 학습률(learning rate)은 0.001로 Adam optimizer를 사용하여 네트워크를 학습하였다. 최종 결과는 검증 데이터에 대한 손실이 최소일 때의 파라미터를 사용하였다.

4.3 실험 결과

Fig.5.는 전처리 기법에 따른 SNR 결과를 그린 것이다. Fig.5.에서 초록색 그래프인 SNR_{Raw}는 전처리 기법을 수행하지 않은 파형에 대한 SNR을 계산한 결과이다. 해당 SNR에서 최대값은 76포인트에서 5.1782이다. 파란색 그래프의 SNR_{PCA}는 기존 제안된 전처리 기법인 주성분분석을 이용하여 step size 1, window size 24, principal components 2로 전처리를 수행한 파형에 대한 SNR이다. 해당 파라미터는 실험을 통해 SNR이 최대가 되는 값을 탐색하여 사용하였으며, 최대 SNR은 112포인트에서 9.5489이다. 빨간색 그래프 SNR_{LDA}는 선형판별분석을 이용하여 주성분분석과 유사한 방법을 통해 전처리를 수행하였으며, step size 1, window size 23, principal components 21로 전처리를 수행한 파형에 대한 SNR 결과이다. 최대 SNR은 1407포인트에서 5.9725로 SNR_{PCA}보다 낮은 수치이지만 파형의 수가 충분히 많아진다면 더 높아질 것으로 예상된다.

노란색 그래프 SNR_{Proposal}은 제안기법을 이용하여 전처리를 수행한 파형에 대한 SNR 결과이다. 전체적으로 SNR이 높은 것을 볼 수 있으며, 특히 76번째 시점에서 20.4902로 다른 전처리 기법들의 최대값과 비교하여 가장 높은 결과을 보여준다. 또한, 76번째 시점은 원본 파형에서 SNR이 가장 높은 지점으로 제안기법에서의 최대 SNR 시점과 일치한다는 것을 알 수 있다. 이와 다르게 다른 전처리 기법의 결과인 SNR_{PCA}와 SNR_{LDA}의 경우 최대 피크의 시점이 principal components를 고려하여 계산하더라도 다르다는 것을 확인할 수 있다.

제안기법은 오토인코더의 구조 중 하나로 차원 압축의 성능 또한 가진다는 것을 보이기 위하여

Table 1. Comparison of Signal-to-Noise Ratio Results

Trace	Maximum SNR
SNR_{Raw}	5.1782
SNR_{PCA}	9.5489
SNR_{LDA}	5.9725
SNR_{Proposal}	20.4902
SNR_{Code}	19.4983

$SNR_{Proposal}$에 사용한 모델과 동일한 오토인코더의 인코더를 이용하여 얻은 압축 데이터에 대한 SNR 결과인 보라색 그래프 SNR_{Code} 또한 비교하였다. 최대 SNR은 $SNR_{Proposal}$보다는 낮지만 19.4983으로 주성분분석과 선형판별분석과 비교하여 우수한 성능을 보여준다. 이는 주성분분석과 선형판별분석과 같은 일반적인 파형 압축 기법을 전체 파형에 사용할 경우에는 기대하기 힘든 결과이다.

실험을 통해 제안기법의 노이즈 제거 성능을 입증할 수 있으며, 또한 기존에 잘 알려진 기법들을 이용하는 것보다 분석 성능이 향상되었음을 알 수 있다.

Fig.6.은 SNR이 가장 높은 76포인트 시점에 대해 각 중간값의 해밍웨이트에 따라 파형 샘플들을 이용하여 정규분포를 피팅한 결과이다. 원본 파형에서는 각 분포 간에 겹치는 부분이 크고, 해밍웨이트 7과 8의 평균이 유사하다. 하지만 제안기법을 통해 획득한 파형에서는 두 분포의 평균이 차이가 크다는 것을 확인할 수 있다.

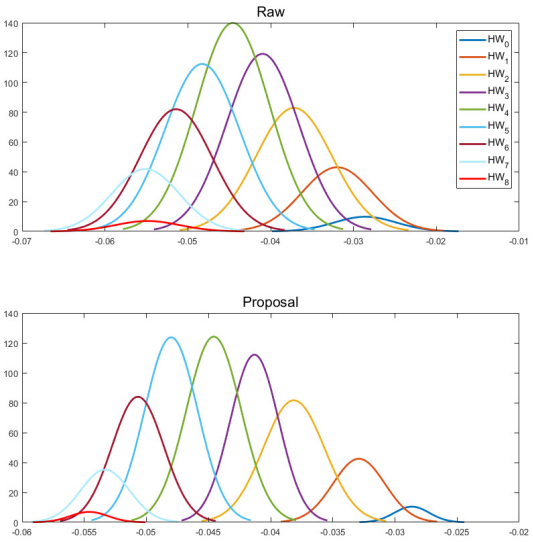

Fig. 6. Normal Distribution of Traces

V. 결 론

딥러닝을 이용한 부채널 분석에 관한 연구가 주목을 받는 이유 중 하나는 부채널 대응기법 적용 여부와 상관없이 기존 전력 분석에서 요구되었던 노이즈 제거(Noise Reduction), 파형 정렬(Trace Alignment) 등의 전처리 과정을 수행하지 않아도 분석할 수 있기 때문이다. 하지만 전처리 과정과 분석 단계를 한 번에 수행하는 End-to-end 공격은 파형에 대한 중간값을 알아야 학습을 할 수 있으므로 프로파일링 공격 환경에서만 수행하거나 추측한 수만큼의 뉴런 네트워크의 학습이 필요하다는 한계점이 존재하였다. 본 논문의 제안기법은 전처리 단계와 분석 단계를 분리하여 수행함으로써 프로파일링 공격 환경뿐만이 아닌 논프로파일링 환경에서도 딥러닝을 이용한 부채널 분석을 수행할 수 있으며, 더 나아가 기존 부채널 분석 성능 향상에도 기여할 수 있다. 제안기법은 전처리 기법으로 기존 부채널 분석 기법과는 독립적으로 수행할 수 있기에, 기존 분석 기법들을 결합하여 적용하여 기존 부채널 분석 기법의 성능을 더욱 향상할 수 있을 것으로 기대된다. 또한, 본 논문에서는 논프로파일링 공격 환경에서의 부채널 분석에 초점을 맞추어서 연구를 수행하였지만, 해당 기법을 통하여 프로파일링 공격의 성능 또한 향상할 수 있을 것으로 예상한다. 하지만 본 논문의 제안기법으로는 마스킹 대응기법이 적용된 구현에 대해서는 분석이 불가하지만, 이는 추후 연구를 통해 해결 가능할 것이다.

References

[1] S. Chari, J.R. Rao, and P. Rohatgi, “Template Attacks,” Cryptographic Hardware and Embedded Systems-CHES 2002, LNCS 2523, pp. 13 - 28, Aug. 2002.

[2] W. Schindler and K. Lemke and C. Paar, “A stochastic model for differential side channel cryptanalysis,” Cryptographic Hardware and Embedded Systems-CHES 2005, LNCS 3659, pp. 30 - 46, Aug. 2005.

[3] P. Kocher, J. Jaffe, and B. Jun, “Differential power analysis,” Advances in Cryptology, CRYPTO ‘99, LNCS 1666, pp. 388-397, Aug. 1999.

[4] E. Brier, C. Clavier, and F. Olivier, “Correlation power analysis with a leakage model,” Cryptographic Hardware and Embedded Systems-CHES 2004, LNCS 3156, pp. 16-29, Aug. 2004.

[5] S. Yang and Y. Zhou and J. Liu and D. Chen, "Back Propagation Neural Network Based Leakage Characterization for Practical Security Analysis of Cryptographic Implementations," International Conference on Information Security and Cryptology-ICISC 2011, LNCS 7259, pp. 169-185, Nov. 2011.

[6] H. Maghrebi and T. Portigliatti and E. Prouff, "Breaking cryptographic implementations using deep learning techniques," International Conference on Security, Privacy, and Applied Cryptography Engineering-SPACE 2016, LNCS 10076, pp. 3-26, Nov. 2016.

[7] E. Cagli and C. Dumas and E. Prouff, "Convolutional Neural Networks with Data Augmentation Against Jitter-Based Countermeasures," Cryptographic Hardware and Embedded Systems-CHES 2017, LNCS 10529, pp. 45-68, Aug. 2017.

[8] B. Timon, "Non-Profiled Deep Learning-based Side-Channel attacks with Sensitivity Analysis," IACR Transactions on Cryptographic Hardware and Embedded Systems-TCHES '19, Vol. 2019, no. 2, pp. 107-131, Feb. 2019.

[9] E. Prouff and M. Ricain and R. Bevan "Statistical Analysis of Second Order Differential Power Analysis," IEEE Transactions on computers, Vol.58, No. 6, pp. 799-811, Jun. 2009.

[10] I. Goodfellow and Y. Bengio and A. Courville, Deep Learning: Adaptive Computation and Machine Learning series, MIT Press, Nov. 2016.

[11] K. Hornik, "Approximation capabilities of multilayer feedforward networks," Neural Networks, vol. 4, pp. 251-257, Jan. 1990.

[12] P. Vincent and H. Larochelle and Y. Bengio and P.A. Manzagol "Extracting and composing robust features with denoising autoencoders," Proceedings of the 25th international conference on Machine learning-ICML'08, pp. 1096-1103, Jul. 2008.

[13] M. Abadi et al, "TensorFlow: Large-scale machine learning on heterogeneous systems," 12th USENIX Symposium on Operating Systems Design and Implementation-OSDI'16, pp. 265-283, Aug. 2016.

[14] Keras: The Python Deep Learning library, "Keras," https://keras.io, 2015.

[15] S. Mangard and E. Oswald and T. Popp, Power Analysis Attacks: Revealing the Secrets of Smart Cards, Springer US, Mar. 2007.

[16] K. He and X. Zhang and S. Ren and J. Sun, "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification," Proceedings of the 2015 IEEE International Conference on Computer Vision-ICCV'15, pp. 1026-1034, Dec, 2015.

〈저자소개〉

권 동 근 (Donggeun Kwon) 학생회원
2018년 2월: 고려대학교 수학과 학사
2018년 3월~현재: 고려대학교 정보보호학과 석사과정
〈관심분야〉 부채널 공격, 딥러닝, 머신러닝 기반 암호분석

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, 머신러닝 기반 암호분석

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