# LSTM 학습 모델을 이용한 권한 & 서드파티 라이브러리 기반 안드로이드 악성코드 탐지

박건우*, 타메르 아부하메드*, 양대헌*, 이경희**

*인하대학교 컴퓨터공학과, **수원대학교 전기공학과

# Android Malware Detection Using Permission & Third-party Library Based LSTM model

Geon-Woo Park*, Tamer Abuhmed*,
Dae-Hun Nyang*, Kyung-Hee Lee**

*Computer Science, Inha University.

**Electrical Engineering, The University of Suwon.

**요 약**

본 논문에서는 정적 분석 도구를 사용하여 24554개의 일반 어플리케이션과 24488개의 악성코드로부터 598개의 AndroidManifest권한과 55개의 라이브러리 목록을 추출하고 이 특징들을 기반으로 LSTM 악성코드 탐지 모델을 구성하였다. 권한 특징만을 사용하여 모델을 구성할 경우 96.06%의 탐지율을 보였지만 권한 특징과 라이브러리를 같이 사용할 경우 97.67%의 탐지 결과를 보였다.

**ABSTRACT**

In this paper, we extract the 598 AndroidManifest privileges and 55 library lists from 24554 general applications and 24488 malicious codes using the static analysis tool, and constructed the LSTM malware detection model based on these features. Our result show that we are able to achieve 96.06% detection accuracy using privileges but to achieve 97.67% detection accuracy using both feature.

## I. 서론

2018년 국내의 스마트폰 보급률은 94%로 세계 최고 수준이라고 할 수 있다. IDC에서 조사한 2018년 스마트폰 OS 점유율은 안드로이드가 85.1%로 가장 높다. 통신 기술의 발전에 따라 스마트폰은 금융, 상거래, 메신저 등 민감한 데이터를 다루는 서비스가 증가하고 있으며 이런 데이터들을 노리는 악성코드 또한 매년 증가하고 있다. Mcafee의 2019년 보고서에 따르면 2018년 1분기에 180만 개 이상의 새로운 악성코드가 발견되었으며 2018년 2분기에는 230만 개가 발견되었고, 해마다 꾸준히 백만 개 이상의 새로운 악성코드가 발견되고 있다.

본 논문에서는 이러한 악성코드 증가 추세에 대응하여 권한 사용 여부와 라이브러리 사용 여부를 이용한 LSTM 머신러닝 악성코드 탐지 모델을 제시하고자 한다.

## II. 관련연구

### 2.1 LibScout [1]

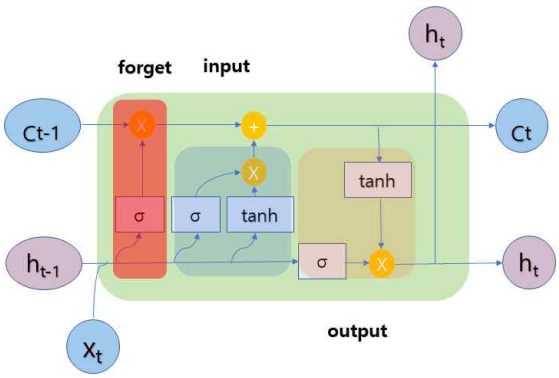
apk 파일에서 권한 & 라이브러리 정보를 추출하기 위하여 LibScout을 사용하였다.

LibScout은 정적분석을 통해 어플리케이션의 서드파티(third-party) 라이브러리를 탐색하는 도구이다. LibScout은 라이브러리 이름, 라이브러리 버전, 사용된 권한과 패키지 구조 등을 분석한다. 실제 라이브러리의 jar 혹은 aar 파일을 대상으로 DEX 바이트코드를 통해 패키지 구조를 학습하여 머클트리(Merkle Tree)를 구성하고 이를 대상 어플리케이션의 패키지 구조를 통해 만든 머클트리를 비교하여 라이브러리 사용을 판별한다. 패키지, 클래스, 모듈의 구조 정보만을 이용하기 때문에 Proguard와 같은 안드로이드 renaming 난독화 도구에 저항성을 갖는다.

### 2.2 LSTM

RNN으로 학습시킬 때는 신경망을 거쳐 갈수록 과거의 정보가 없어져 파라미터 학습을 방해하는 Vanishing Gradient Problem이 발생할 수 있다. LSTM은 이러한 문제를 3개의 게이트 구조(입력, 출력, 망각)를 이용하여 보완한 구조이다. 그림 1은 LSTM 신경망의 내부 구조이다. [2]

그림 1




### 2.2 Bi-LSTM

앞서 설명한 단방향 LSTM은 과거 데이터의 망각 정도를 조절하기 때문에 Vanishing Gradient Problem을 다소 해결하였다. 하지만 단방향으로 학습을 진행할 경우 데이터의 길이가 길고 층이 깊으면 직진패턴을 기반으로 수렴하는 문제가 발생한다. Bi-LSTM은 두 개의 신경망이 순방향 역방향으로 짝을 이루어 학습을 시키므로 이 문제를 개선하였다. 그림 2은 Bi-LSTM의 구조이다.

그림 2

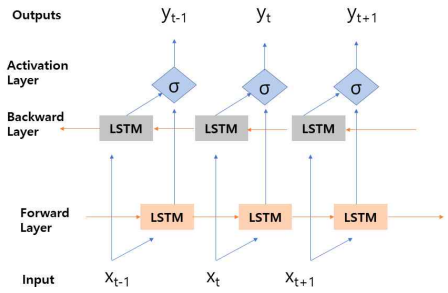



표 1. 24554개의 일반 어플리케이션과 24488개의 악성코드 중 라이브러리를 사용하는 비율

| | 일반 | 악성코드 |
|---|---|---|
| firebase | 6017 | 618 |
| Facebook | 22313 | 12965 |
| Amazon | 273 | 2 |
| Retrofit | 352 | 0 |
| crashlytics-core | 397 | 0 |
| support-annotations | 9948 | 230 |
| AndroidAsync | 67 | 8 |

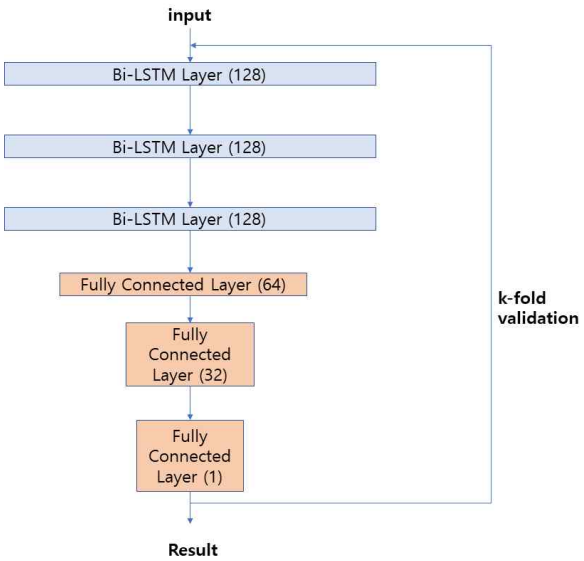
## III. 악성코드의 라이브러리 사용 경향

표 1은 주요 라이브러리의 사용 정보이다. 악성코드에서는 평균 1.3%로 매우 낮은 라이브러리 사용률을 보인다. 악성코드는 목적이 분명하기에 다양한 라이브러리의 지원을 요구하지 않는 것으로 보인다. 반면에 안드로이드 라이브러리, 특히 서드파티 라이브러리의 사용은 생산성을 크게 증가시키기 때문에 정상 안드로이드 개발자는 높은 비중으로 이용한다. 이러한 특징을 이용하여 악성코드의 탐지율을 높이고자 한다.

## IV.실험

LibScout을 통해 추출된 정보는 Json 형태로 저장된다. 이 Json 파일로부터 권한과 라이브러리 사용 목록을 확인하여 벡터 형태로 입력 데이터를 만들고 csv형태로 저장한다 ($input = (perm_1, perm_2, ..., Lib_1, Lib_2, ...)$). 각 벡터가 나타내는 값은 0 또는 1이다. 구성된 특징을 이용하여 양방향 LSTM 신경망 3계층을 구성하였으며 뒤이어 fully connected 신경망을 3계층을 추가로 구성하였다. k-fold validation을 통해 교차검증을 수행하였고 권한만 사용했을 경우 96.06%, 라이브러리와 권한을 같이 사용했을 경우 97.67%의 정확도를 보였다.

그림 3




## V. 결론

라이브러리 탐색 도구를 활용하여 악성코드의 라이브러리 사용률이 대단히 저조함을 보였다. 이 사실에 근거하여 라이브러리 사용 여부를 부차적인 특징으로 이용하여 권한 기반의 악성코드 탐지에 효율을 소폭 높였다.

본 연구에서는 라이브러리의 탐지율이 다소 낮아 큰 효과를 얻지 못하였다. 향후 연구로 라이브러리 탐지율 증가와 함께 더 높은 정확도를 목표로 연구하려고 한다.


## [Acknowledgements]

이 논문은 2016년도정부(과학기술정보통신부)의 재원으로 한국연구재단-글로벌연구실사업의 지원을 받아 수행된 연구임(NRF-2016K1A1A2912757).


## [참고문헌]


[1] Michael Backes, et al. “Reliable Third-Party Library Detection in Android and its Security Applications.” 2016 ACM SIGSAC Conference on Computer and Communications Secufity (CCS)

[2] 고상준, 윤호영, 신동명 “양방향 LSTM을 활용한 전력수요 데이터 예측 기법 연구” .2018년 6월 한국소프트웨어감정평가학회 논문집, 14(1), 33-40.

[3] 강성은, 응웬부렁, 정수환 (2018.6). “머신러닝을 이용한 권한 기반 안드로이드 악성코드 탐지” 정보보호학회논문지, 28(3), 617-623