KIPS Tr. Software and Data Eng.
Vol.4, No.4 pp.169~176 pISSN: 2287-5905

http://dx.doi.org/10.3745/KTSDE.2015.4.4.169

Malicious Application Determination Using the System Call Event

SeokMin Yun† · YouJeong Ham†† · GeunShik Han††† · HyungWoo Lee††††

ABSTRACT

Recently smartphone market is rapidly growing and application market has also grown significantly. Mobile applications have been provided in various forms, such as education, game, SNS, weather and news. And It is distributed through a variety of distribution channels. Malicious applications deployed with malicious objectives are growing as well as applications that can be useful in everyday life well. In this study, Events from a malicious application that is provided by the normal application deployment and Android MalGenome Project through the open market were extracted and analyzed. And using the results, We create a model to determine whether the application is malicious. Finally, model was evaluated using a variety of statistical method.

Keywords : Malicious Applications, Machine Learning, Event Extraction, Logistic Regression

시스템 콜 이벤트 분석을 활용한 악성 애플리케이션 판별

윤 석 민† · 함 유 정†† · 한 근 식††† · 이 형 우††††

요 약

최근 스마트폰 시장의 빠른 성장과 함께, 애플리케이션 시장 또한 크게 성장하고 있다. 애플리케이션은 날씨, 뉴스와 같은 정보검색을 비롯하여 교육, 게임, SNS 등 다양한 형태로 제공되고 있으며 다양한 유통경로를 통해 배포되고 있다. 이에 따라 일상에서 유용하게 사용할 수 있는 애플리케이션뿐만 아니라 악의적 목적을 가진 악성 애플리케이션의 배포 역시 급증하고 있다. 본 연구에서는 오픈마켓을 통해 배포되고 있는 정상 애플리케이션 및 Android MalGenome Project에서 제공하는 악성 애플리케이션의 이벤트를 추출, 분석하여 임의의 애플리케이션의 악성 여부를 판별하는 모형을 작성하고, 여러 가지 지표를 통해 모형을 평가하였다.

키워드 : 악성 애플리케이션, 기계 학습, 이벤트 추출, 로지스틱 회귀분석

1. 서 론

최근 스마트폰 시장이 빠르게 성장하고 있다. 2009년 5%에 이르던 전 세계 스마트폰 보급률이 2013년에는 22%로, 불과 4년 사이에 4배 이상 폭발적으로 증가하였다. 우리나라는 현재, 전 세계 스마트폰 보급률 2위(73%) 국가로 이는 국내 인구의 4명 중 3명이 스마트폰을 사용하고 있는 셈이다. 스마트폰 사용자가 증가함에 따라 날씨, 뉴스와 같은 정보검색을 비롯하여 교육, 게임, SNS 등 다양한 형태의 애플리케이션이 개발되고 오픈마켓을 통해 배포되고 있다. 하지만 누구나 개발하여 배포할 수 있는 오픈마켓의 특성상 일상에서 유용하게 사용할 수 있는 애플리케이션뿐만 아니라 악의적 목적을 가진 악성 애플리케이션의 배포 역시 급증하고 있다. 대부분의 악성 애플리케이션은 오픈마켓 또는 인터넷, 블랙마켓 등 여러 유통경로를 통해 정상 어애플리케이션인 것처럼 위장하여 악성코드를 삽입한 형태로 배포되고 있으며[1] 이러한 악성코드는 사용자의 기기정보, 위치정보 외에도 단말 내에 저장된 전화번호부, SMS, 통화기록 등의 개인정보뿐만 아니라 저장된 공인인증서 등 금융정보를 외부 서버로 유출시키는 공격을 시도하기도 한다[2]. 이러한 악성 애플리케이션들의 확산으로 인한 피해를 줄이기 위해서는 보다 정확한 방법으로 정상 애플리케이션과 악성 애플리케이션을 판별할 수 있는 메커니즘이 개발되어야 한다.

본 연구에서는 안드로이드 기반의 스마트폰 애플리케이션에서 발생하는 이벤트의 패턴을 분석하여 정상 애플리케이션과 악성 애플리케이션을 판별하고자 한다. 분석 방법으로는 회귀분석, 의사결정나무, 신경망 학습을 적용하였으며 각 모형의 결과를 비교하였다.

※ 이 논문은 2013년도 정부(미래창조과학부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. NRF-2013R1A2A2A01068256).
† 준 회 원 : 한신대학교 컴퓨터공학과 석사
†† 비 회 원 : 한신대학교 컴퓨터공학과 석사
††† 정 회 원 : 한신대학교 컴퓨터공학부 교수
†††† 종신회원 : 한신대학교 컴퓨터공학부 교수
Manuscript Received : October 15, 2014
First Revision : February 27, 2015
Accepted : February 27, 2015
* Corresponding Author : Yun SeokMin(the_alchemist@hs.ac.kr)

2. 관련 연구

모바일 보안 분야에서 악성 애플리케이션을 탐지, 분류하는 연구는 꾸준히 진행되고 있으며 관련 연구로는 Crowdroid 기법, 리패키징 앱 탐지 기법, 악성 패밀리 기반 악성 앱 탐지 기법, 유사도 기반 악성 애플리케이션 판별 등이 있다. 본 절에서는 여러 가지 분석 기법 중 Strace를 이용하여 데이터를 수집하는 Crowdroid 기법과 유사도 기반 악성 애플리케이션 판별 기법을 간략하게 정리하였다.

2.1 Crowdroid 시스템[3]

Crowdroid 시스템은 행동 기반의 악성 프로그램 탐지 프레임워크로 데이터 획득 단계, 데이터 처리 단계, 악성 분석 및 탐지 단계로 구성되어있다.

1) 데이터 획득 단계에서는 N명의 모바일 사용자가 Crowdsourcing 애플리케이션을 설치하고 이를 이용하여 각각의 단말에서 실행된 시스템 콜 이벤트 로그를 수집한다.
2) 데이터 처리 단계에서는 Perl 기반 데이터 분석 스크립트를 이용하여 N명의 사용자에 대한 벡터 정보를 생성한다.
3) 악성 분석 및 탐지 단계에서는 N개의 벡터 정보에 대해 K-means 알고리즘을 이용하여 클러스터링 한 뒤, 그 결과를 DB에 저장하고 정상 행위 데이터와 비정상 행위 데이터에 대한 구별과정을 수행한다.

Crowdroid 시스템은 비교적 간단한 유사도 분석과정을 통해 특정 애플리케이션의 악성 여부를 판단하는 과정을 수행할 수 있다. 그러나 구체적으로 어떤 시스템 콜 이벤트가 추출되었는지에 대한 내용 없이 정상 및 악성 애플리케이션 실행 시 발생하는 시스템 콜 이벤트의 유사도만을 가지고 악성 여부를 판단하고 있다는 문제점을 가지고 있다.

2.2 유사도 기반 악성 애플리케이션 판별[4]

유사도 기반 악성 애플리케이션 판별은 Jaccard Coefficient 기법[5]을 기반으로 안드로이드 애플리케이션에서 수집한 이벤트의 유사도에 적용시킬 수 있도록 계산을 변형한 방법으로 수행 과정은 다음과 같다.

1) 준비된 정상 또는 악성 애플리케이션 실행 후 Strace를 이용하여 시스템 콜 이벤트를 추출하고 이벤트별 발생빈도를 계산한다.
2) 임의의 애플리케이션을 실행하여 이벤트를 추출하고 이벤트별 발생빈도를 계산한다.
3) 정상 및 악성 애플리케이션 실행 시 이벤트 발생 빈도값 EV_i^{App}과 임의의 애플리케이션 실행 시 발생한 이벤트 빈도값 EV_j^{AppRdm}에 대하여

$$\frac{\min(EV_i^{App}, EV_j^{AppRdm})}{\max(EV_i^{App}, EV_j^{AppRdm})} \quad (1)$$

값을 계산한다.

4) 1)부터 3)까지의 과정을 임의의 애플리케이션 실행 시 발생한 모든 이벤트에 대해 반복적으로 수행하여 계산하면 각 이벤트마다 유사도값이 나타나게 된다. 이를 바탕으로 다시 이벤트 전체에 대한 평균 유사도값을 구한다.
5) 위와 같은 단계를 거쳐 애플리케이션의 이벤트 셋과 임의의 애플리케이션 이벤트의 유사도를 나타내는 수식은 다음과 같다.

$$S = \frac{\left| \sum_{i=1}^{n} EV_i^{App} \right|}{\left| EV_j^{AppRdm} \right|} \quad (2)$$

위 식에서 이벤트 유사도는 0에서 1 사이의 값이 된다.

유사도 기반 악성 애플리케이션 판별 기법은 정상 및 악성 애플리케이션 그룹의 이벤트별 평균 발생빈도를 이용하기 때문에 사용자가 임의의 애플리케이션을 설치하였을 때, 해당 애플리케이션의 악성 여부를 판별할 수 있는 기준이 된다. 그러나 모든 이벤트에 대해 유사도를 작성하여 비교하기 때문에 악성 여부 판별에 영향을 주지 못하는 이벤트 또한 유사도 분석에 포함될 수 있고, 이는 판별능력의 저하를 유발할 수 있다.

3. 분석모형 설계

3.1 모형설계 환경

본 연구에서는 R Ver.3.1.0과 RStudio Ver.0.98을 개발환경으로, 모형에 관한 패키지는 의사결정나무 관련 패키지인 rpart와 신경망학습 지원 패키지인 RSNNS(R Stuttgart Neural Network Simulator)를 사용하였다. R은 통계 계산과 그래픽을 위한 프로그래밍 언어로, 행렬 계산을 위한 도구로서도 사용될 수 있으며 GNU Octave나 MATLAB에 견줄만한 성능을 보여준다.

3.2 데이터 구성

본 연구에 활용한 데이터는 안드로이드 플랫폼을 기반으로 하는 애플리케이션을 대상으로 이벤트를 추출한 이벤트 발생빈도 데이터이다. 현재 안드로이드 마켓에서 사용자 이용도가 높은 124개의 애플리케이션을 정상 애플리케이션 분석대상으로 선정하였고, Android MalGenome Project[6]에서 배포하고 있는 1260개의 악성 애플리케이션 리스트 중 123개의 애플리케이션을 악성 애플리케이션 분석대상으로 선정하여 이벤트를 추출하였다. 애플리케이션에서 이벤트 추출은 리눅스 기반 시스템 콜 추출도구인 Strace를 사용하였다.

Table 1. Components of Confusion Matrix

Confusion Matrix		Predict	
		Positive'(P')	Negative'(N')
Target	Positive(P)	True Positive(TP)	False Negative(FN)
	Negative(N)	False Positive(FP)	True Negative(TN)

Table 2. Terminology and Derivations from a Confusion Matrix[9]

Performance Indicator	Calculations
True Positive Rate(TPR, Sensitivity)	TP/P = TP/(TP+FN)
True Negative Rate(TNR, Specificity)	TN/N = TN/(FP+TN)
Positive Predictive Value(PPV)	TP/(TP+FP)
Negative Predictive Value(NPV)	TN/(TN+FN)
False Positive Rate(FPR)	FP/N = FP/(FP+TN)
False Discovery Rate(FDR)	FP/(FP+TP) = 1-PPV
False Negative Rate(FNR)	FN/(FN+TP)
Accuracy(ACC)	(TP+TN)/(P+N)

Strace는 안드로이드 애플리케이션이 실행될 경우 발생한 모든 시스템 콜 이벤트들을 추적하고 이를 수집하여 출력 파일로 생성하는 기능을 제공한다[7]. 전체 247개의 애플리케이션을 대상으로 분석을 진행하였으며, 전체 데이터 중 70%는 분석기법별 모형설계를 위한 학습(Train)용 데이터로 사용하였고, 나머지 30%는 작성된 모형에 대입하여 모형으로 인한 예측 결과와 실제값을 비교하기 위한 모형평가(Test)용 데이터로 사용하였다.

3.3 분석변수 선정

Strace를 이용한 이벤트 추출 시 발생한 이벤트의 종류는 llseek, dup, dup2, SYS_285, exit, futex, ioctl, syscall_983042, setpriority, chmod 등 총 98종이며 발생한 모든 이벤트의 종류를 독립변수로 사용하였다. 분석대상 선정 시 Type 변수를 지정하여 정상 애플리케이션의 경우 'Norm(Normal)'값을, 악성 애플리케이션의 경우 'Mal(Malicious)'값을 갖는 변수를 생성하고 이를 종속변수로 사용하였다.

자료의 특성상 불필요한 변수들이 많기 때문에 요약통계량과 상관분석, 산점도 등을 통해 변수들에 대한 탐색적 자료분석을 시행하였고, 이를 통해 회귀분석, 의사결정나무, 신경망 등의 데이터 분석모형 작성에 부적합한 변수를 제거하였다. 특정 애플리케이션에서만 발생한 이벤트의 경우 이벤트 발생빈도가 대부분이 0으로 구성되어있어, 분석에 악영향을 미치는 것으로 판단된다. 따라서 발생빈도가 0인 값이 95% 이상 차지하는 이벤트 변수는 분석대상에서 제외하였고, 이로 인해 dup2, SYS_285, SYS_297을 비롯하여 총 21개의 변수가 분석대상에서 제외되었다.

3.4 모형평가 기준

모형은 크게 의사결정나무, 로지스틱 회귀분석, 신경망 학습모형(오류역전파 알고리즘)으로 나누어 작성하였으며 이를 평가하기 위한 기준으로는 혼동행렬과 ROC Curve를 이용하였다.

1) 혼동행렬

혼동행렬(Confusion Matrix)은 기계학습 분야에서 분할표(Contingency Table) 또는 오차행렬(Error Matrix)로도 잘 알려져 있다[8]. 혼동행렬은 실제값과 모형을 통한 예측값 간의 관계를 나타내는 행렬인데 이는 모형의 성능을 평가하는 지표로 사용된다.

Table 1에서 True의 의미는 실제값과 예측값이 일치하는 경우를, False는 실제값과 예측값이 일치하지 않는 경우를 나타낸다. 혼동행렬의 성능을 측정하는 방법은 실제 P를 P'로 예측한 비율을 나타내는 True Positive Rate(TPR, Sensitivity), 실제 N을 N'로 예측한 비율을 나타내는 True Negative Rate(TNR, Specificity), 값을 정확히 예측한 비율을 나타내는 Accuracy(ACC) 등의 방법들이 있으며 이러한 지표를 계산하는 공식은 Table 2에 기술하였다.

Table 2에서 보는 바와 같이 혼동행렬을 이용하여 모형 성능을 측정하는 다양한 방법이 있으며, 그중 일반적으로 가장 많이 사용되는 성능 지표는 민감도(Sensitivity, TPR)와 특이도(Specificity, TNR)이다. 본 연구에서는 민감도와 특이도를 이용하여 ROC Curve를 작성하고 이를 통해 각 모형을 평가하였다.

2) ROC Curve

ROC(Receiver Operating Characteristic) Curve는 민감도

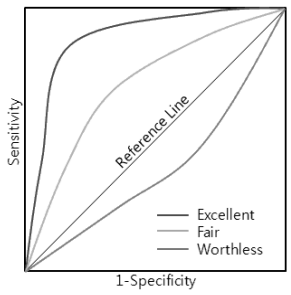
```
> summary(fit)
Call:
rpart(formula = Type ~ ., data = appDataTree, method = "class")
  n= 247

         CP nsplit rel error    xerror       xstd
1 0.77235772      0 1.0000000 1.1219512 0.06344526
2 0.02032520      1 0.2276423 0.2601626 0.04290823
3 0.01626016      3 0.1869919 0.2764228 0.04402260
4 0.01000000      5 0.1544715 0.2764228 0.04402260

Variable importance
         futex syscall_983042          ioctl      mprotect     setpriority         write          exit       SYS_294       SYS_295
            19            15             13            12              12            11             1             1             1
         pread         prctl        SYS_286         clone  syscall_983045      SYS_281      SYS_283     ftruncate          stat64
             1             1              1             1               1            1            1             1             1
       fcntl64        access    sigprocmask
             1             1              1
```

Fig. 2. Summary of Decision Tree

와 특이도를 비교하여 나타내는 그래프로, 모형의 성능을 평가하는 척도로 사용된다[10]. ROC Curve는 그래프 아래 면적을 의미하는 AUC(Area Under the ROC Curve)를 기준으로 모형의 성능을 평가하는데, 면적에 따른 모형성능 지표는 다음 Fig. 1과 같다.



AUC	모형성능
0.90-1.00	excellent
0.80-0.90	good
0.70-0.80	fair
0.60-0.70	poor
0.50-0.60	fail

Fig. 1. AUC and Model Performance Evaluation[11]

양 모서리를 지나는 Reference Line(AUC=0.5)에 가까울수록 종속변수에 대한 변별력이 떨어진다고 판단하며 1에 가까울수록 증가한다고 판단한다. 일반적으로 AUC가 0.9 이상이면 매우 좋은 예측치를 갖는 모형, 0.8 이상이면 좋은 예측치를 갖는 모형, 0.7 이상이면 타당한 수준의 모형이라고 판단한다.

3.5 모형설계

모형설계 단계에서는 앞에서 수집한 데이터를 대상으로 의사결정나무, 로지스틱 회귀분석, 신경망 학습모형을 작성하였다.

1) 의사결정나무

의사결정나무(Decision Tree)는 의사결정규칙을 나무 구조로 도식화하여 분류와 예측을 수행하는 분석 방법으로, 일반적으로 사용되는 알고리즘은 CART, C4.5&C5.0, CHAID 등이 있다. 본 연구에서는 가장 설명력 있는 변수에 대하여 최초로 분리가 일어나는 특징을 가지는 CART 알고리즘[12]을 이용하였다.

R에서 의사결정나무의 CART 알고리즘은 rpart 패키지로 제공된다. rpart 패키지를 이용하면 의사결정나무 작성과정, 사용된 변수의 중요도 등을 확인해볼 수 있다. 우선 앞 절에서 일부 변수를 제외하고 남은 78개 변수를 대상으로 의사결정나무를 작성하면 Fig. 2와 같은 요약 결과를 얻을 수 있다.

의사결정나무 요약 결과 futex, syscall_983042, ioctl, mprotect, setpriority, write 변수가 다른 변수들에 비해 중요도가 높게 나타났으며 이 변수들을 대상으로 의사결정나무를 작성하면 Fig. 3과 같은 결과를 얻을 수 있다.

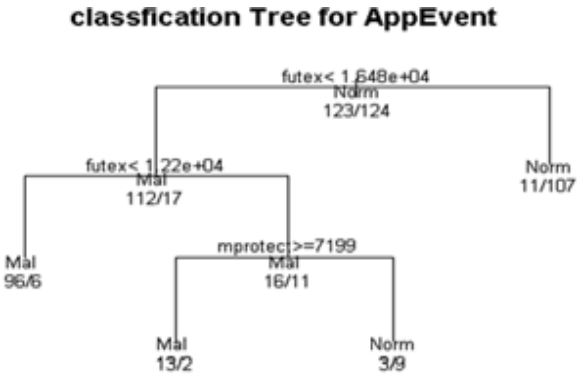


Fig. 3. Result of Decision Tree

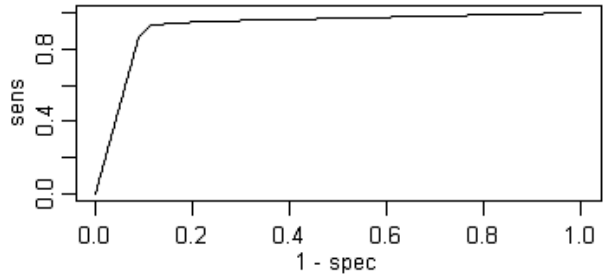
Fig. 3을 살펴보면 실제 의사결정나무에 사용된 변수는 futex와 mprotect이며, 먼저 futex 이벤트의 발생빈도 약 16,480회를 기점으로 정상집단과 악성집단이 1차적으로 분류되는 것을 확인할 수 있다. 또 전체 247개 애플리케이션 중 1차 분류를 통해 정상집단으로 분류된 것은 118개로 그중 90.7%인 107개가 실제 집단과 일치하였으며 악성집단으로 분류된 129개 애플리케이션 중 86.8%에 해당하는 112개가 실제 집단과 일치하였다. 악성으로 분류된 집단은 약 12,200회를 기점으로 한 번 더 분류하게 되는데, futex 이벤트 발생빈도 12,200회 미만인 집단의 경우 102개 애플리케이션 중 실제 집단과 일치하는 애플리케이션은 96개로, 94.1%의 높은 정확도를 보였다. futex 이벤트가 12,200회 이상 발생한 집단에서는 마지막으로 mprotect 이벤트 발생빈도 7,199회를 기준으로 정상과 악성집단을 분류한다. 위 내용을 통해 의사결정나무에서 애플리케이션의 정상/악성을 구분하는 데 있어 futex 이벤트가 가장 설명력이 높음을 확인할 수 있었다. Fig. 3의 의사결정나무 작성 결과를 종합하여 정리하면 다음 Table 3과 같은 혼동행렬(Confusion Matrix)을 얻을 수 있다.

Table 3. Confusion Matrix of Decision Tree

Decision Tree		Predict	
		Norm	Mal
Target	Norm	116	14
	Mal	8	109

Performance Indicator	Calculations
Sensitivity	**89.2%**
Specificity	**93.2%**
Positive Predictive Value	93.5%
Negative Predictive Value	88.6%
False Positive Rate	6.8%
False Discovery Rate	6.5%
False Negative Rate	10.8%
Accuracy	**91.1%**

의사결정나무 수행 결과, 민감도(Sensitivity)는 89.2%로 나타났으며, 특이도(Specificity)는 그보다 조금 높은 93.2%로 나타났다. 전체 분류 정확도(Accuracy)는 91.1%로 좋은 성능을 보였다. 다음 Fig. 4는 의사결정나무의 ROC Curve를 나타낸 것이다.



Area under the curve: 0.9218

Fig. 4. ROC Curve for Decision Tree

위의 Fig. 4에서 의사결정나무의 AUC는 0.9218로, 의사결정나무가 애플리케이션의 정상/악성 여부를 판별하는 데 있어 좋은 모형임을 확인할 수 있었다.

2) 로지스틱 회귀분석

로지스틱 회귀분석(Logistic Regression)은 잘 알려진 통계분류모형 중 하나로, 분석하고자 하는 대상들이 두 집단 혹은 그 이상의 집단으로 나누어진 경우, 개별 관측값들이 어느 집단으로 분류될 수 있는가를 분석하고 이를 예측하는 데 사용되는 대표적인 알고리즘이다[13]. 앞 절에서 나타난 변수 중요도에서 상위 6개 변수를 대상으로 로지스틱 회귀분석을 수행한 결과는 다음 Fig. 5와 같다.

```
> summary(res2)

Call:
glm(formula = Type ~ futex + syscall_983042 + ioctl + setpriority +
    mprotect + write, family = "binomial", data = trainData)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-3.1445  -0.5757   0.0000   0.3135   2.6121

Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)    -3.531e+00  5.996e-01  -5.889 3.87e-09 ***
futex           3.996e-05  1.155e-05   3.460 0.000541 ***
syscall_983042  1.109e-03  4.009e-04   2.767 0.005662 **
ioctl          -2.561e-05  1.226e-05  -2.089 0.036674 *
setpriority    -9.563e-04  3.017e-03  -0.317 0.751296
mprotect       -2.979e-04  2.014e-04  -1.479 0.139052
write           3.138e-04  2.284e-04   1.374 0.169407
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 246.20  on 177  degrees of freedom
Residual deviance: 112.87  on 171  degrees of freedom
AIC: 126.87
```

Fig. 5. Summary of Logistic Regression

위의 로지스틱 회귀분석 결과를 보면 futex, syscall_983042 변수의 p-value가 각각 0.000541, 0.005662로 모형에 매우 유의한 영향을 주는 것을 알 수 있다. 반면 setpriority 변수의 p-value는 0.752로 모형에 전혀 유의미한 결과를 주지 못하는 것으로 나타났다. 모형의 개선을 위하여 setpriority를 비롯하여 p-value가 0.1 이상인 write, mprotect를 분석대상에서 제외하고 로지스틱 회귀분석을 수행한 결과는 다음 Fig. 6와 같다.

```
> summary(res2)

Call:
glm(formula = Type ~ futex + syscall_983042 + ioctl,
    family = "binomial", data = trainData)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.86838  -0.57693   0.00009   0.32449   2.62572

Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)    -3.563e+00  5.639e-01  -6.318 2.66e-10 ***
futex           4.059e-05  1.113e-05   3.646 0.000266 ***
syscall_983042  6.856e-04  1.439e-04   4.764 1.90e-06 ***
ioctl          -2.366e-05  1.198e-05  -1.975 0.048302 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 246.20  on 177  degrees of freedom
Residual deviance: 117.63  on 174  degrees of freedom
AIC: 125.63

Number of Fisher Scoring iterations: 7
```

Fig. 6. Summary of Improved Logistic Regression

학습용 데이터를 기반으로 작성된 모형을 모형평가 데이터에 적용시킨 결과는 아래 Fig. 7과 같이 나타난다. futex, syscall_983042, iotcl는 Strace를 이용하여 추출한 각 이벤트의 빈도수를 나타내며 my.res는 로지스틱 회귀분석 모형을 통해 예측된 결과이다. Type은 정상 애플리케이션과 악성 애플리케이션을 분류하는 실제값을 나타낸다.

	Name	futex	sc_983042	ioctl	my.res	Type
1	talkingben	56723	7590	18130	Norm	Norm
2	hereyo4	18018	4905	15030	Norm	Norm
3	galaxypiano	67711	11509	20851	Norm	Norm
4	girlbg	32605	9019	20294	Norm	Norm
5	gatalk	35580	10064	16693	Norm	Norm
6	audiko2	133763	34136	52796	Norm	Norm
7	sail	49225	8499	14888	Norm	Norm
8	simsimi	22546	6584	5795	Norm	Norm
9	interpark_app_ticket	27126	20101	17090	Norm	Norm
10	freemusicplayer	9387	2678	2297	Mal	Norm
11	freedrama1	1092	181	587	Mal	Norm
12	gorealra	76511	4867	9568	Norm	Norm
13	imbc	45100	6210	7846	Norm	Norm
14	isadari	18023	2412	10650	Mal	Norm
15	kwansang	17049	4397	12456	Mal	Norm
60	wtofall	107533	40847	46150	Norm	Norm
61	bakerystory	243278	6090	24458	Norm	Norm
62	omokmaster	187129	6687	129892	Norm	Norm
63	as_danti598	8828	2420	6005	Mal	Mal
64	as_mood	9529	2468	4098	Mal	Mal
65	as_Racing	38	5	37	Mal	Mal
66	bb_mfzone	2515	591	1402	Mal	Mal
67	bb_rootcallblocker	2931	750	806	Mal	Mal
68	gd_Dizz	13619	3119	8897	Mal	Mal
69	ddl_myapn	5513	1494	5632	Mal	Mal

Fig. 7. Result of Logistic Regression

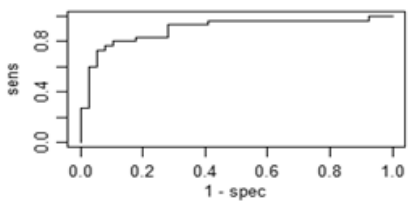
모형평가에 사용된 데이터는 전체 데이터의 약 30%에 해당하는 69개로, 이 중 정상 애플리케이션의 수는 39개, 악성 애플리케이션의 수는 30개로 구성되어있다. 이 데이터를 앞 절에서와 같은 방법으로 혼동행렬을 작성하면 다음 Table 4와 같은 결과를 얻을 수 있다.

Table 4. Confusion Matrix of Logistic Regression

Logistic Regression		Predict	
		Norm	Mal
Target	Norm	37	2
	Mal	8	22

Performance Indicator	Calculations
Sensitivity	**94.9%**
Specificity	**73.3%**
Positive Predictive Value	82.2%
Negative Predictive Value	91.7%
False Positive Rate	26.7%
False Discovery Rate	17.8%
False Negative Rate	5.1%
Accuracy	**85.5%**

로지스틱 회귀분석 결과 민감도(Sensitivity)는 94.9%로 나타났으며, 높은 민감도에 비해 특이도(Specificity)는 상대적으로 낮은 73.3%로 나타났다. 전체 분류 정확도는 85.5%로 의사결정나무에 비해 저조한 성능을 보였다. 다음 Fig. 8은 로지스틱 회귀모형의 ROC Curve를 그린 것이다.



Area under the curve: 0.9

Fig. 8. ROC Curve for Logistic Regression Models

Fig. 8에서 로지스틱 회귀분석 모형의 AUC는 0.9로, 의사결정나무에 비해 다소 낮은 성능을 보이지만, 그럼에도 불구하고 상당히 좋은 판별능력을 보임을 확인할 수 있다.

3.6 신경망 학습

신경망(Neural Network) 학습이란, 인간의 뇌의 구조와 뇌에서 수행되는 정보처리 방식을 모방함으로써 인간이 지능적으로 처리하는 복잡한 정보처리 능력을 기계를 통해 실현하고자 하는 연구로 현재 신호처리, 제어, 패턴인식, 음성·문자인식, 예측 등의 분야에서 널리 응용되고 있다. 본 연구에서는 표본수의 한계로 인해, 여러 가지 신경망 분석기법 중 보다 안정적인 결과를 얻을 수 있는 오류역전파(EBP, Error Back Propagation) 알고리즘을 이용하였다. 전체 데이터 중 30%를 모형평가용 데이터로 나누고 학습률 0.1, 최대 반복수행횟수(Iteration)는 100회로 설정한 뒤 작업을 수행하였다. 학습 알고리즘을 수행하여 얻은 예측 결과값을 출력해보면 Fig. 9과 같은 결과가 나타난다.

Fig. 9에서 name 변수는 해당 애플리케이션의 이름을 나타내며, Mal과 Norm 변수는 해당 애플리케이션이 속한 그룹을 예측한 확률값을 보여준다. 즉, Mal의 값이 0.5 이상이면 해당 애플리케이션은 악성으로 판단하며 Norm의 값이 0.5 이상이면 해당 애플리케이션을 정상 애플리케이션으로 판단한다.

name	Mal	Norm
as_danti572	0.959984601	0.04114276
dkf1_stetris	0.655847907	0.35593832
2048	0.120318726	0.88078672
concubinewar	0.002321868	0.99722838
bb_danti416	0.951649129	0.05066647
interpark_app_ticket	0.320879310	0.67486256
as_danti285	0.987972021	0.01146221
ddl_calculator	0.503559709	0.51864606
stickerbooth	0.172248736	0.85799509
hairclipper	0.299885005	0.71594441

Fig. 9. Neural Network Predictions

학습 결과 전체를 혼동행렬로 정리하면 Table 5, Table 6과 같이 표현할 수 있다. Table 5는 학습용 데이터를 이용하여 모형을 학습시켰을 때 나타난 결과를 정리한 혼동행렬이며, Table 6은 학습이 완료된 모형을 평가용 데이터에 적용시켰을 때의 혼동행렬이다.

Table 5. Confusion Matrix of Error Back Propagation Model (Train Data)

Error Back Propagation		Predict	
		Norm	Mal
Target	Norm	84	4
	Mal	4	80

Performance Indicator	Calculations
Sensitivity	95.5%
Specificity	95.2%
Accuracy	95.3%

학습이 완료된 모형의 혼동행렬을 보면 민감도 95.5%, 특이도 95.2%, 예측 정확도 95.3%로 매우 높은 정확도를 보였다. 반면 학습이 완료된 모형에 대하여 평가용 데이터를 이용하여 모형의 성능을 평가한 결과, 민감도 94.3%, 특이도 92.5%, 정확도 93.3%로 학습이 완료된 모형에 미치지는 못하지만 의사결정나무나 로지스틱 회귀에 비해 전반적으로 높고 안정적인 판별능력을 보이는 것으로 나타났다.

Table 6. Confusion Matrix of Error Back Propagation Model (Test Data)

Error Back Propagation		Predict	
		Norm	Mal
Target	Norm	33	2
	Mal	3	37

Performance Indicator	Calculations
Sensitivity	**94.3%**
Specificity	**92.5%**
Positive Predictive Value	91.7%
Negative Predictive Value	94.9%
False Positive Rate	7.5%
False Discovery Rate	8.3%
False Negative Rate	5.7%
Accuracy	**93.3%**

다음 Fig. 10은 오류역전파 알고리즘을 이용하여 학습시킨 모형과, 이를 모형검증용 데이터에 적용시켰을 때의 결과를 그래프로 표현한 것이다. Fig. 10에서 좌측 상단 그래프는 학습반복횟수에 따른 오차제곱합을 나타낸다. 검정색은 모형설계에 사용된 학습용 데이터를 나타내며, 빨간색은 모형검증에 사용된 데이터를 나타낸다. 그래프를 보면 두 그래프가 X축(Iteration)값이 30 이후부터 차이가 드러나기 시작하는데 이는 반복학습으로 인해 해당 모형이 모형설계 데이터에 다소 과적합(Over-fitting)되었음을 보여준다. 하단의 두 그래프는 ROC Curve를 나타내는데 좌측 하단 그래프는 학습용 데이터를 통해 설계된 모형의 ROC Curve이며 우측 하단의 그래프는 모형평가용 데이터를 사용하여 나타낸 ROC Curve이다. 평가용 데이터를 기준으로 한 모형의 AUC는 0.9729로, 분류성능이 매우 좋은 모형임을 확인할 수 있다.

4. 결 론

본 연구에서는 오픈마켓을 통해 배포되고 있는 애플리케이션의 시스템 콜 이벤트를 추출하고, 이를 대상으로 예측모형을 작성함으로써 임의의 애플리케이션에 대한 악성 여부를 판별하고자 하였다. 연구 방법으로는 애플리케이션에서 추출된 시스템 콜 이벤트 빈도를 대상으로 로지스틱 회귀분석, 의사결정나무(CART 알고리즘), 신경망 학습(오류역전파 알고리즘) 기법을 적용하고, 모형의 성능을 나타내는 AUC값을 기준으로 모형의 성능을 비교하였다. 각 모형별 AUC값을 살펴보면 의사결정나무는 0.9218, 로지스틱 회귀모형은 0.9, 신경망 학습모형이 0.9729로 나타났다. 즉 모형의 성능은 신경망 학습모형, 의사결정나무, 로지스틱 회귀모형순으로 임의의 애플리케이션에 대한 악성여부 판별능력은 신경망학습을 이용하는 것이 가장 적절함을 확인 할 수 있었다.

의사결정나무 작성 결과, futex 이벤트의 발생빈도가 애플리케이션의 정상과 악성을 분류하는 데 있어 높은 설명력을 보임을 확인할 수 있었다. futex 이벤트는 리소스에 접근하는 태스크의 우선순위를 관리해줌으로써 간단한 코드를 수행하면서 발생하는 오버헤드를 줄여주기 때문에 애플리케이션의 반응속도를 향상시켜주는 중요한 역할을 하는 이벤트이다. 이러한 futex 이벤트의 빈도수 차이는 애플리케이션의 배포목적에 따라 차이를 보이고 있는데, 이를 통해 상업적 목적으로 배포되는 일반적인 애플리케이션의 경우, 애플리케이션의 성능을 좌우하는 태스크의 우선순위 관리가 필

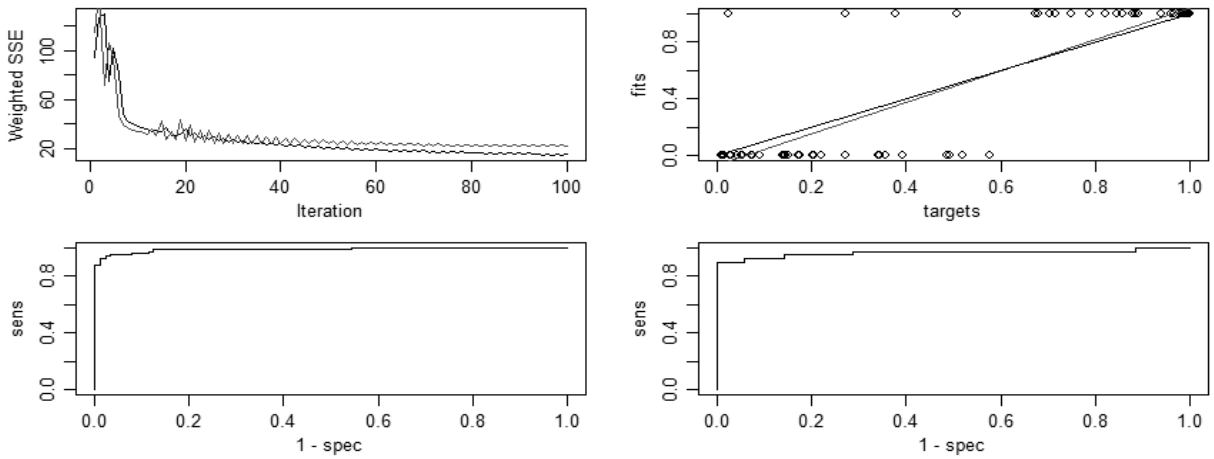


Fig 10. Error Back Propagation Model Results Performed

수적인 요소이지만, 개인정보 및 금융정보의 유출 등 악의적 목적을 띄고 배포되는 애플리케이션의 경우 그렇지 않다는 사실을 추측해볼 수 있다.

전체 분석모형 중에서는 신경망 학습 알고리즘이 전반적으로 가장 안정적인 성능을 보였는데, 이는 모형 작성에 가장 많은 변수를 사용하며 은닉층(hidden layer)을 통해 각 노드 간의 가중치를 조정하는 신경망 학습 알고리즘의 특성이 잘 반영되었기 때문인 것으로 보인다.

그러나 본 연구는 애플리케이션의 카테고리별 특성 차이로 발생하는 오탐과 끊임없이 변화하는 악성 애플리케이션의 다양한 공격기법에 대한 한계를 가진다. 이는 애플리케이션의 카테고리별 분석이 필요함을 의미하며 다수의 사용자로부터 지속적으로 데이터를 수집한다면 이러한 부분을 개선할 수 있을 것으로 보인다.

References

[1] Y. J. Ham, H. W. Lee, "Normal and Malicious Application Pattern Analysis using System Call Event on Android Mobile Devices for Similarity Extraction," *Journal of Internet Computing and Services(JICS)*, Vol.16, No.8, pp.125-139, 2013.

[2] Wajeb, Abdulrahman Mirza, "Software Vulnerabilities, Banking Threats, Botnets and Malware Self-Protection Technologies," *International Journal of Computer Science Issues(IJCSI)*, Vol.8, No.1, pp.236-241, 2011.

[3] I. Burguera, U. Zurutuza, and S. Nadjm-Tehrani, "Crowdroid: Behavoir-Based Malware Detection System for Android," *Proceeding of the 1st ACM workshop on security and privacy in smartphones and mobile devices (SPSM'11)*, ACM, Vol.1, pp.15-26, 2011.

[4] Y. J. Ham, "Malicious Application Event Discrimination and Diagnosis Mechanism on SmartPhone," Master's Thesis, Hanshin University, 2014.

[5] Suphakit Niwattanakul, Jatsada Singthongchai, Ekkachai Naenudorn, and Supachanun Wanapu, "Using of Jaccard Coefficient for Keywards Similarity," *Proceedings of the International MultiConference of Engineering and Computer Scientists 2013 (IMECS 2013)*, Vol.1, pp.13-15, 2013.

[6] Yajin Zhou, Xuxian Jiang, Android Malgenome Project, [Internet], http://www.malgenomeproject.org/

[7] Strace, trace system calls and signals, [Internet], http://linux.die.net/man/1/strace/

[8] Stehman, Stephen V, "Selecting and interpreting measures of thematic classification accuracy," *Remote Sensing of Environment,* Vol.62, No.1, pp.77-89, 1997.

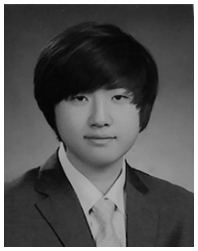
[9] Fawcelt, Tom, "An Introduction to ROC Analysis," *Pattern Recognition Letters*, Vol.27, No.8, pp.861-874, 2006.

[10] Swets, John A, "Signal detection theory and ROC analysis in psychology and diagnostics," collected papers, Lawrence Erlbaum Associates, Mahwah, NJ, 1996.

[11] C. H. Jun, "Data mining Techniques," Hannarae, 2012.

[12] Breiman, L., Friedman, J., Stone, C. J., and Olshen, R. A., "Classification and regression trees," CRC press, 1984.

[13] Christopher M. Bishop, "Pattern Recognition and Machine Learning," Springer, pp.205, 2006.



윤 석 민

e-mail : the_alchemist@hs.ac.kr
2012년 한신대학교 컴퓨터공학부(학사)
2015년 한신대학교 컴퓨터공학과 석사
관심분야 : 데이터분석, 데이터마이닝, 빅데이터, R

함 유 정

e-mail : you86400@hanmail.net
2012년 한신대학교 정보통신학과(학사)
2014년 한신대학교 컴퓨터공학과 석사
관심분야 : 정보보호, 스마트폰 보안, Smart Fuzzing.

한 근 식

e-mail : gshan@hs.ac.kr
1993년 Oklahoma State University(박사)
1994년~현 재 한신대학교 컴퓨터공학부 정교수
관심분야 : Pattern Recognition, Big data analysis

이 형 우

e-mail : hwlee@hs.ac.kr
1994년 고려대학교 컴퓨터학과(학사)
1996년 고려대학교 컴퓨터학과(석사)
1999년 고려대학교 컴퓨터학과(박사)
1999년~2003년 백석대학교 정보통신학부 조교수
2003년~현 재 한신대학교 컴퓨터공학부 부교수, 정교수
관심분야 : 정보보호, 스마트폰 보안, 컴퓨터 포렌식스, Fuzzing