

CK09: Machine Learning in Android Malware Evasion

NT522.O21.ATCL
GVHD: Phan Thế Duy
Nhóm: 2



Team members

- Trần Tấn Hải - 21522036
- Bùi Nguyên Phúc - 21522469
- Huỳnh Anh Nguyễn - 21522388



Nội dung

- **Giới thiệu**
- Phương pháp thực hiện
- Thực nghiệm và đánh giá
- Kết luận và hướng phát triển



Giới thiệu

- Sự phổ biến của các thiết bị android.
- Sự phổ biến của việc ứng dụng machine learning vào việc phát hiện mã độc.
- Mục tiêu của chúng ta là tạo ra mẫu mã độc có khả năng tránh né trình phát hiện mã độc dựa trên machine learning.
- Ref: X. Chen et al., "Android HIV: A Study of Repackaging Malware for Evading Machine-Learning Detection," in IEEE Transactions on Information Forensics and Security, vol. 15, pp. 987-1001, 2020, doi: 10.1109/TIFS.2019.2932228.

Giới thiệu



Giới thiệu

machine learning in malware d... x +

scholar.google.com.vn/scholar?q=machine+learning+in+malware+detection&hl=vi&as_sdt=0&as_vis=1&oi=scholar

Google Scholar machine learning in malware detection

Bài viết Khoảng 117.000 kết quả (0,12 giây) Hồ sơ của tôi Thư viện cá nhân

Mọi lúc
Từ 2024
Từ 2023
Từ 2020
Phạm vi tùy chọn...

Sắp xếp theo mức độ liên quan
Sắp xếp theo ngày

Mọi loại
Bài viết đánh giá

☐ bao gồm bằng sáng chế
☐ bao gồm trích dẫn
☒ Tao thống báo

The curious case of machine learning in malware detection [PDF] arxiv.org
S Saad, W Brigguglio, H Elmiligi - arXiv preprint arXiv:1905.07573, 2019 - arxiv.org
... the success of **malware** detectors powered by **machine learning** in the wild... **malware detection**. Finally, we outline potential research directions in **machine learning** for **malware detection**...
☆ Lưu Trích dẫn Trích dẫn 46 bài viết Bài viết có liên quan Tất cả 8 phiên bản

A survey on machine learning-based malware detection in executable files
J Singh, J Singh - Journal of Systems Architecture, 2021 - Elsevier
... **malware** development. In this paper detailed study of **malware detection** techniques using **machine learning** ... In addition, this paper discusses various challenges for developing **malware** ...
☆ Lưu Trích dẫn Trích dẫn 185 bài viết Bài viết có liên quan

Opem: A static-dynamic approach for machine-learning-based malware detection [PDF] santosgrueiro.com
I Santos, J Devesa, E Brezo, J Nieves - ... joint conference CISIS'12 ..., 2013 - Springer
... Given this background, we present here OPEM, the first **machine-learning**based **malware detector** that employs a set of features composed of both static and dynamic features. The ...
☆ Lưu Trích dẫn Trích dẫn 241 bài viết Bài viết có liên quan Tất cả 3 phiên bản

A review of android malware detection approaches based on machine learning [PDF] ieee.org
K Liu, S Xu, G Xu, M Zhang, D Sun, H Liu - IEEE access, 2020 - ieeeexplore.ieee.org
... of **detection** effectiveness. ... **malware detection** based on **machine learning**. This review will help academics gain a full picture of Android **malware detection** based on **machine learning**. It ...
☆ Lưu Trích dẫn Trích dẫn 268 bài viết Bài viết có liên quan Tất cả 4 phiên bản

Analysis of machine learning techniques used in behavior-based malware detection [PDF] sgu.ac.id
I Firdausi, A Erwin, AS Nugroho - 2010 second international ..., 2010 - ieeeexplore.ieee.org
https://link.springer.com/chapter/10.1007/978-3-642-33018-6_28 ce, automated behavior-based **malware detection** using **machine learning**



Nội dung

- Giới thiệu
- **Phương pháp thực hiện**
- Thực nghiệm và đánh giá
- Kết luận và hướng phát triển

Phương pháp thực hiện

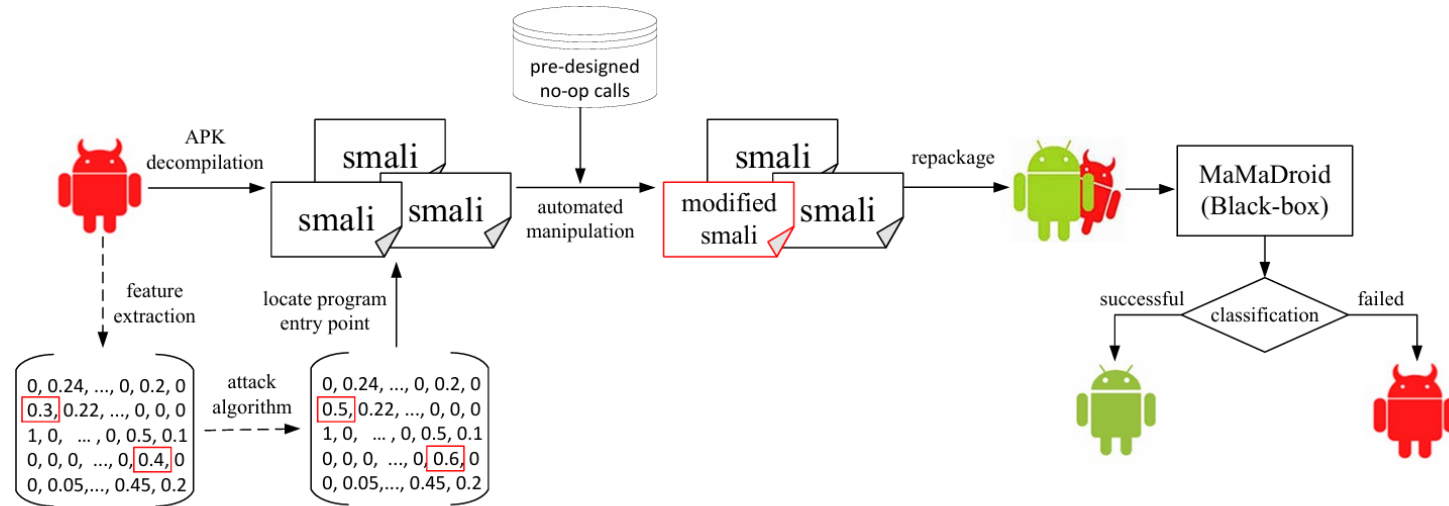


Fig. 3. The attack process: the dashed lines show the process of our attack algorithm, and the solid lines illustrate our APK manipulation procedure.



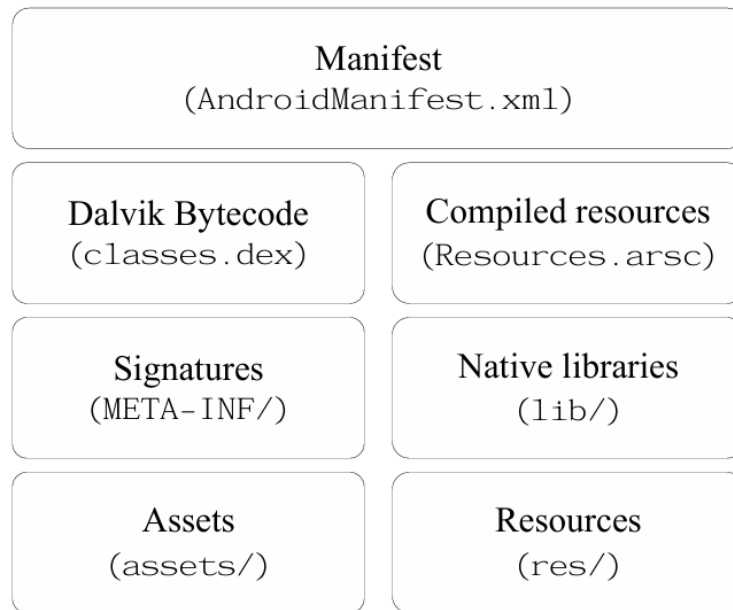
Phương pháp thực hiện

- Xây dựng mô hình tấn công (attack model)
- Xây dựng mô hình phát hiện mã độc (detection model)
- Thử nghiệm xem đầu ra của mô hình tấn công có tránh bị phát hiện từ mô hình phát hiện không



Cấu trúc của file APK

- Ứng dụng Android được đóng gói và phân phối dưới dạng file APK
- File APK là 1 dạng file jar
- Đặc trưng thường được trích xuất từ Manifest và Dalvik Bytecode
- Manifest: chứa thông tin về permissions
- Dalvik Bytecode: chứa Source Code





Xây dựng mô hình phát hiện mã độc

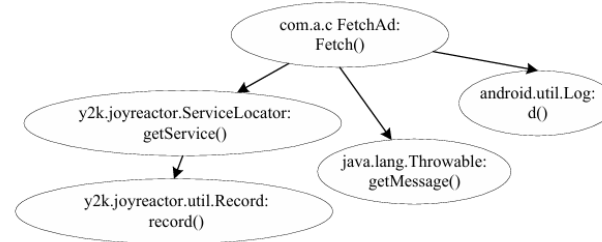
- MaMaDroid: Phát hiện mã độc bằng cách xây dựng Markov Chains of Behavior Models
- MaMaDroid tận dụng các mô hình RF, KNN, SVM để xây dựng trình phát hiện mã độc (binary classification)

Xây dựng MaMaDroid

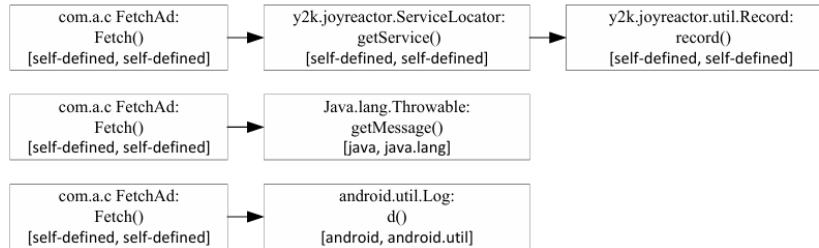
```
package com.a.c;
import android.util.Log;
import android.content.Context;
import y2k.joyreactor.ServiceLocator;
import y2k.joyreactor.util.Record;

public class FetchAd {
    public static boolean Fetch(Context para){
        try {
            ServiceLocator.getService(true).record(para);
            return true;}
        catch (Exception para) {
            Log.d("u tty", para.getMessage()); }
        return false;
    }
}
```

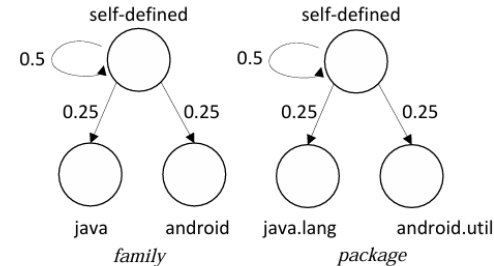
(a) source code



(b) call graph generated from (a)

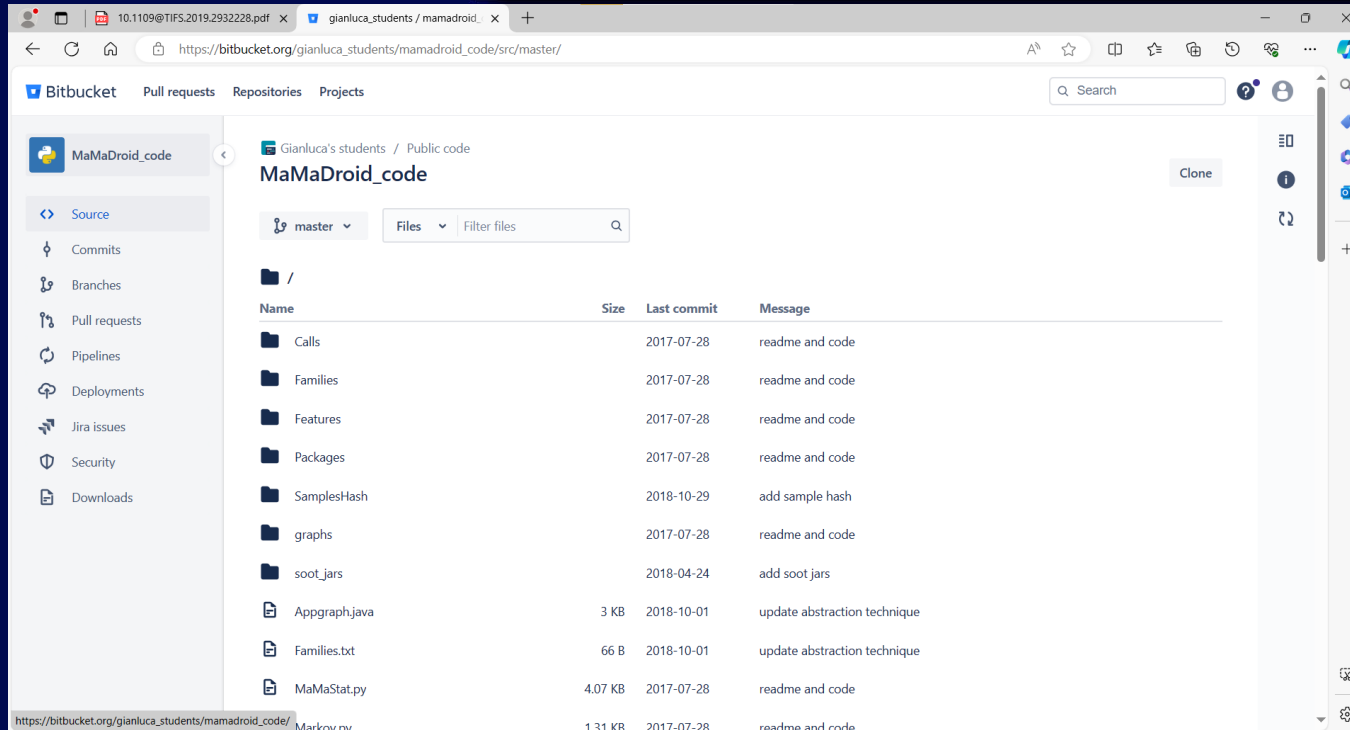


(c) call sequence extracted from (b), with corresponding family/package abstraction in square brackets



(d) Markov chains generated from (c), in family and package mode.

Xây dựng MaMaDroid



The screenshot shows a Bitbucket web interface for a repository named 'MaMaDroid_code' under the user 'Gianluca's students'. The left sidebar contains navigation links: Source, Commits, Branches, Pull requests, Pipelines, Deployments, Jira issues, Security, and Downloads. The main content area displays the 'master' branch with a table of files and folders. The table has columns for Name, Size, Last commit, and Message. The files listed are: Calls, Families, Features, Packages, SamplesHash, graphs, soot_jars, Appgraph.java, Families.txt, MaMaStat.py, and Markov.nv. The last commit for all files is dated 2017-07-28, except for SamplesHash which is dated 2018-10-29.

Name	Size	Last commit	Message
Calls		2017-07-28	readme and code
Families		2017-07-28	readme and code
Features		2017-07-28	readme and code
Packages		2017-07-28	readme and code
SamplesHash		2018-10-29	add sample hash
graphs		2017-07-28	readme and code
soot_jars		2018-04-24	add soot jars
Appgraph.java	3 KB	2018-10-01	update abstraction technique
Families.txt	66 B	2018-10-01	update abstraction technique
MaMaStat.py	4.07 KB	2017-07-28	readme and code
Markov.nv	1.31 KB	2017-07-28	readme and code

Link: [Bitbucket](https://bitbucket.org/gianluca_students/mamadroid_code/src/master/)



Xây dựng mô hình phát hiện mã độc

- Drebin: trích xuất đặc trưng từ file manifest và disassembled dexcode bằng quét tuyến tính (linear sweep) qua các file đó.
- Drebin: các đặc trưng được trích xuất có giá trị 0 hoặc 1
- Drebin: sử dụng SVM để xây dựng mô hình phát hiện mã độc (binary classification)

Xây dựng mô hình phát hiện mã độc

```
.method private addSuspiciousApiFeature()V
    .locals 1
    const-string v0, "phone"
    .line 17
    invoke-virtual {p0, v0},
        La/test/com/myapp/MainActivity;->
        getSystemService(Ljava/lang/String;)
        Ljava/lang/Object;
    move-result-object v0
    check-cast v0,
        Landroid/telephony/TelephonyManager;
    return-void
.end method
```

Xây dựng mô hình phát hiện mã độc

The screenshot shows the GitHub repository page for 'annamalai-nr/drebin'. The repository is public and has 46 forks and 100 stars. The main content area displays the repository's structure, including a 'src' directory, '.gitignore', and 'README.md'. The 'README.md' file is expanded, showing the repository's purpose and dependencies. The 'About' section on the right provides a brief description of the project and lists related topics.

Repository: **annamalai-nr / drebin** (Public)

Navigation: [Code](#) [Issues](#) [Pull requests](#) [Actions](#) [Projects](#) [Security](#) [Insights](#)

Branches: **master** (1 Branch) 0 Tags

Search:

Commits: **40 Commits**

Files:

- src** Update HoldoutClassification.py 7 years ago
- .gitignore** Change the feature builder from using DictVectorizer into Tf... 7 years ago
- README.md** Added explanations for '--model' & '--numfeatforexp' argu... 7 years ago

README

What does this repository contain?

This repo contains a python implementation of Arp, Daniel, et al. "DREBIN: Effective and Explainable De

What package/platform dependencies do I need to have to run the code?

The code is developed and tested using python 2.7 on Ubuntu 16.04 PC.
The following packages need to be installed to run the code:

1. sklearn (==0.18.1)
2. pebble
3. glob

About

Drebin - NDSS 2014 Re-implementation

Topics: [machine-learning](#) [malware-analysis](#) [malware-research](#) [androguard](#) [android-malware](#) [malware-detection](#) [android-malware-detection](#) [drebin](#)

Readme

Activity

100 stars

8 watching

46 forks

Report repository

Releases

No releases published

Packages

No packages published

Contributors 5

Link: <https://github.com/annamalai-nr/drebin>



So sánh MaMaDroid và Drebin

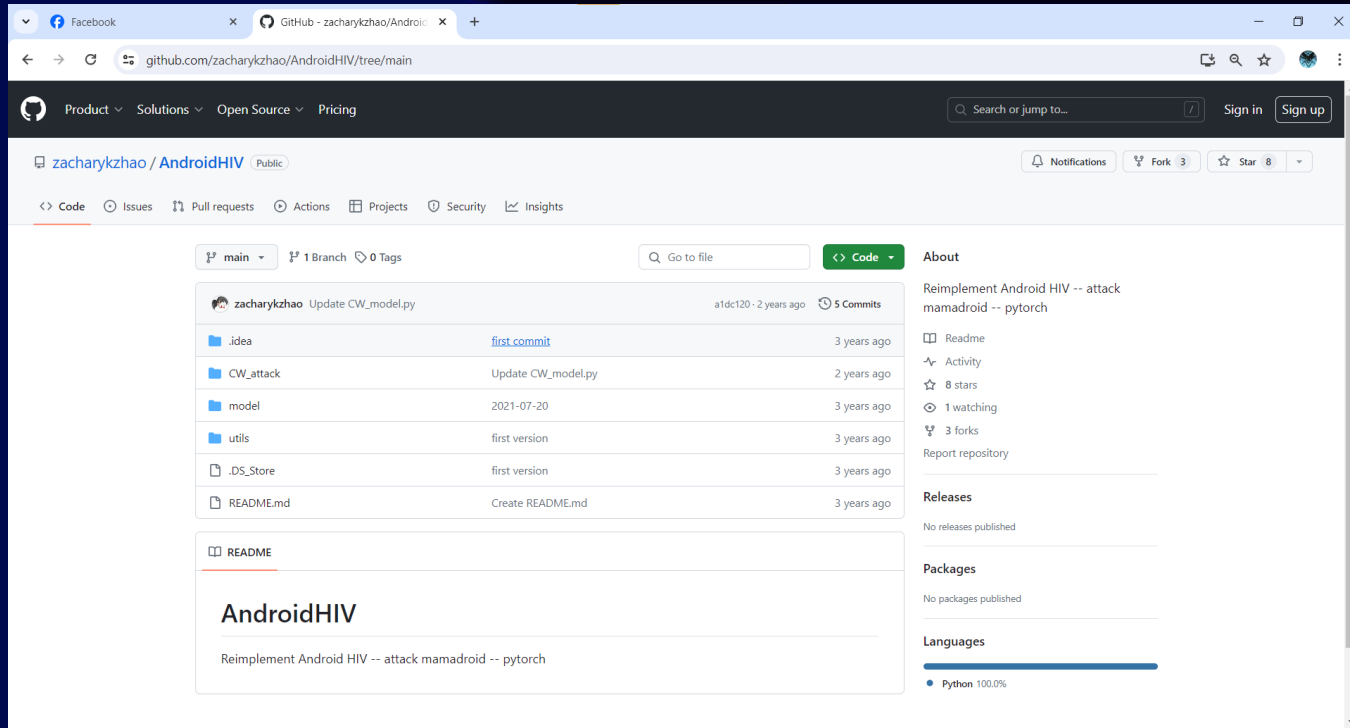
Tên mô hình	MaMaDroid	Drebin
Cách trích xuất đặc trưng	Sử dụng Markov Chain	Sử dụng linear sweep
Giá trị của đặc trưng được trích xuất	Từ 0 đến 1	0 hoặc 1
Đặc điểm	Có 2 chế độ hoạt động: family (nhẹ) và package (nặng)	Nhẹ
Thuật toán để phát hiện mã độc	RF, KNN, SVM	SVM
Tỉ lệ phát hiện mã độc	96%	97%



Xây dựng mô hình tấn công

- C&W: thuật toán tạo mẫu hình ảnh đối kháng
- C&W attack: huấn luyện một substitute model để tạo một giá trị xấp xỉ bằng AdaGrad
- C&W attack: né trình phát hiện bằng cách tối ưu các giá trị của đặc trưng để từ đó làm nhiễu vector đặc trưng (bằng cách gia thay đổi số lượng gọi API từ các callers và callees)

Building attack model



Facebook x GitHub - zacharyzhao/AndroidHIV x +

github.com/zacharyzhao/AndroidHIV/tree/main

Product Solutions Open Source Pricing

Search or jump to... Sign in Sign up

zacharyzhao / AndroidHIV Public

Notifications Fork 3 Star 8

<> Code Issues Pull requests Actions Projects Security Insights

main 1 Branch 0 Tags

Go to file <> Code

zacharyzhao Update CW_model.py a1dc120 · 2 years ago 5 Commits

.idea	first commit	3 years ago
CW_attack	Update CW_model.py	2 years ago
model	2021-07-20	3 years ago
utils	first version	3 years ago
.DS_Store	first version	3 years ago
README.md	Create README.md	3 years ago

README

AndroidHIV

Reimplement Android HIV -- attack mamadroid -- pytorch

About

Reimplement Android HIV -- attack mamadroid -- pytorch

Readme Activity 8 stars 1 watching 3 forks Report repository

Releases

No releases published

Packages

No packages published

Languages

Python 100.0%

Link: <https://github.com/zacharyzhao/AndroidHIV/tree/main>



Xây dựng mô hình tấn công

- JSMA: thuật toán tạo mẫu hình ảnh đối kháng
- JSMA attack: thay đổi 1 giá trị nào đó của 1 đặc trưng nào đó và kiểm tra xem độ ảnh hưởng của sự thay đổi đó đối với vector đặc trưng. Ta sẽ chọn thay đổi giá trị có độ ảnh hưởng cao nhất và lặp lại cho tới khi né được trình phát hiện hoặc khi chạm tới giới hạn mà ta đã đặt

Phương pháp thực hiện

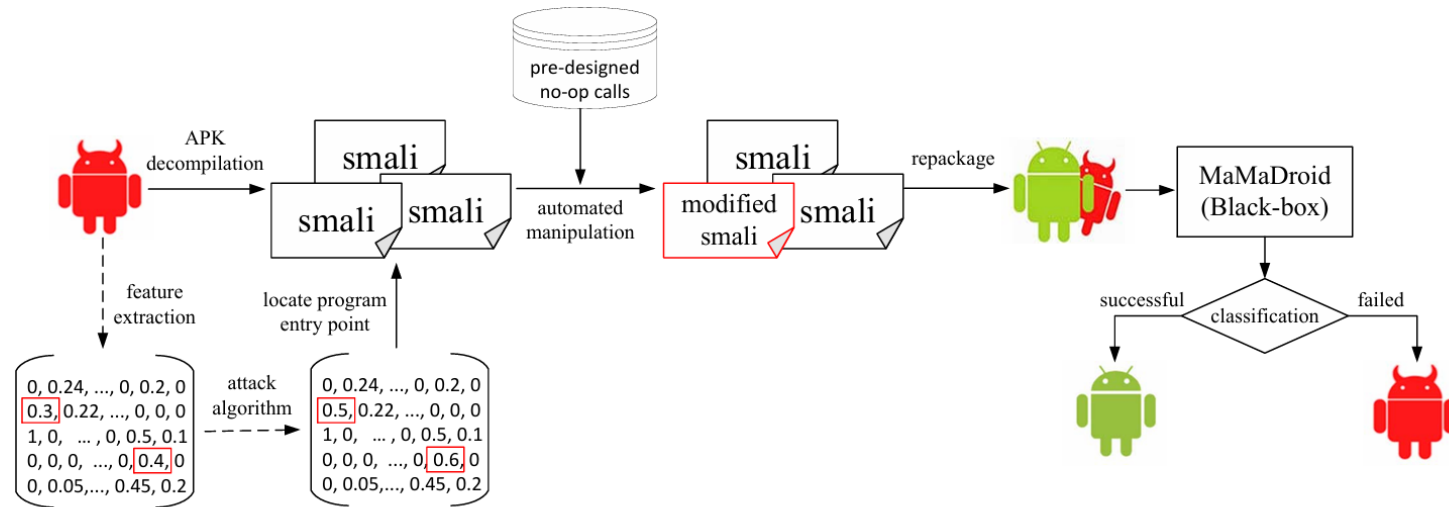


Fig. 3. The attack process: the dashed lines show the process of our attack algorithm, and the solid lines illustrate our APK manipulation procedure.



So sánh hai mô hình tấn công

Tên mô hình	C&W attack	JSMA attack
Đối tượng tấn công	MaMaDroid	MaMaDroid, Drebin
Cách tấn công	Làm thay đổi giá trị của đặc trưng bằng cách tính toán độ xấp xỉ và tối ưu giá trị	Làm thay đổi giá trị của đặc trưng bằng cách tìm ra đặc trưng nào ảnh hưởng nhất đến kết quả phân lớp
Evasion Rate	100%	100%
Average Distortion	50	3.5



Nội dung

- Giới thiệu
- Phương pháp thực hiện
- **Thực nghiệm và đánh giá**
- Kết luận và hướng phát triển

Thực nghiệm và đánh giá

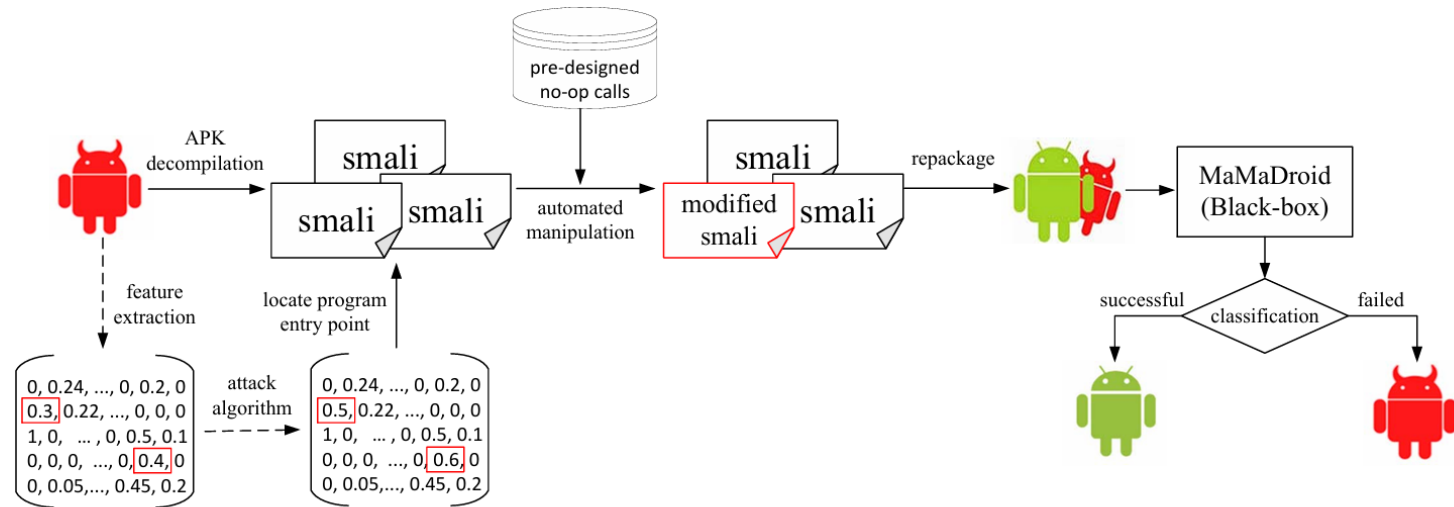


Fig. 3. The attack process: the dashed lines show the process of our attack algorithm, and the solid lines illustrate our APK manipulation procedure.



Môi trường

- Jupyter notebook - visual studio code
- CPU: 11th Gen Intel(R) Core(TM) i5-1135G7 @ 2.40GHz
- RAM: 8GB
- Windows 10 Home Single Language



Dataset

Name	drebin215dataset5560malware9476benign
Number of features	215
Number of instances	15036
Class	Benign, Malicious
Number of benign files	9476
Number of malicious files	5560



Classifier

Classifier algorithm	Support Vector Machine (SVM)
Kernel	Linear
C	1



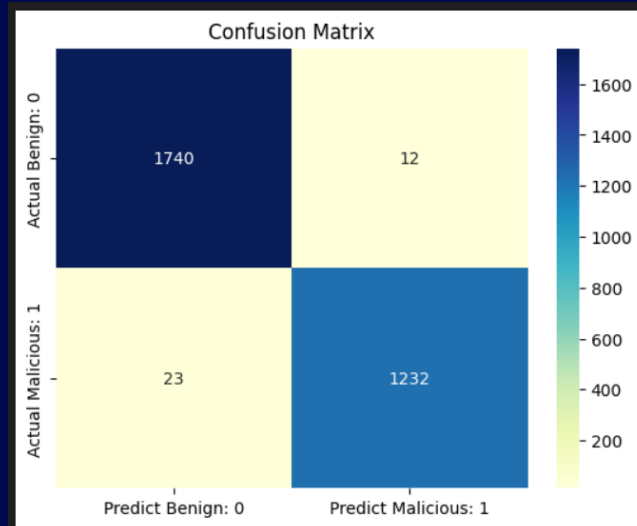
Attack model

Attack algorithm	JSMA
Classifier	SVM
theta	0.1
gamma	1
batch_size	1

Kết quả của classifier

	From paper	From our experiment
Accuracy	97%	98%

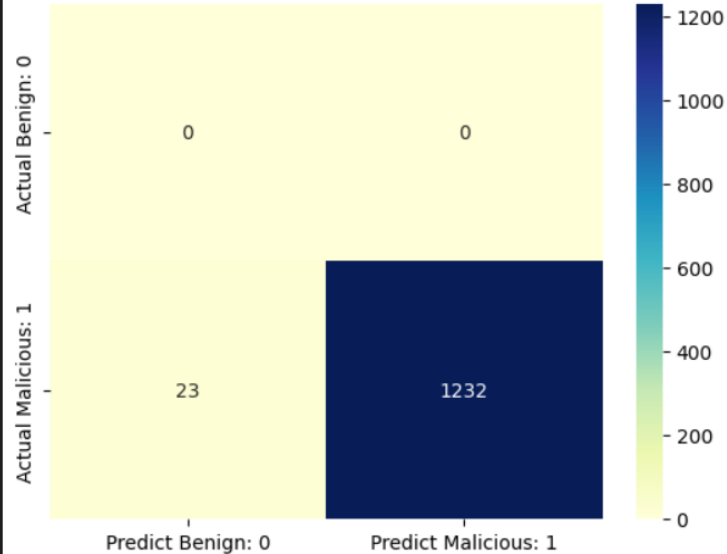
- Our experiment



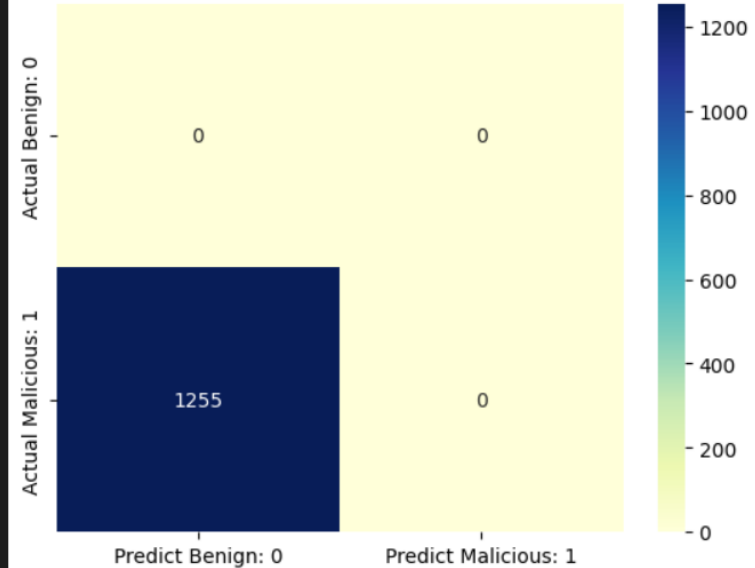
Kết quả của classifier

- Our experiment

Confusion Matrix Before JSMA Attack

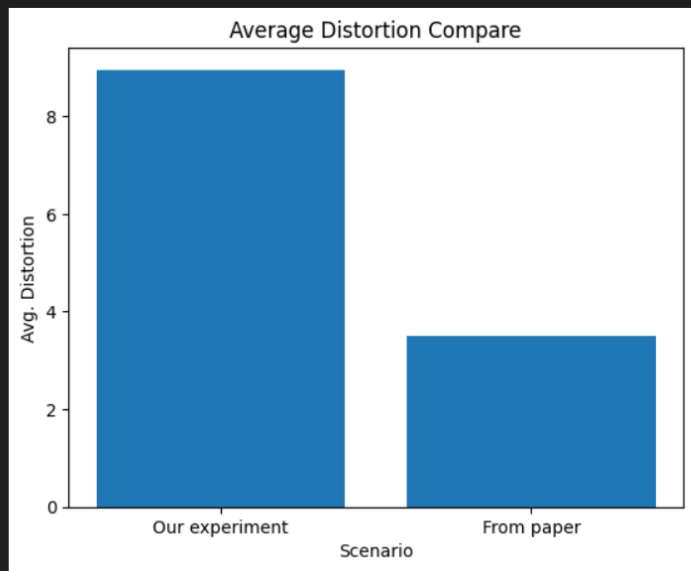


Confusion Matrix After JSMA Attack

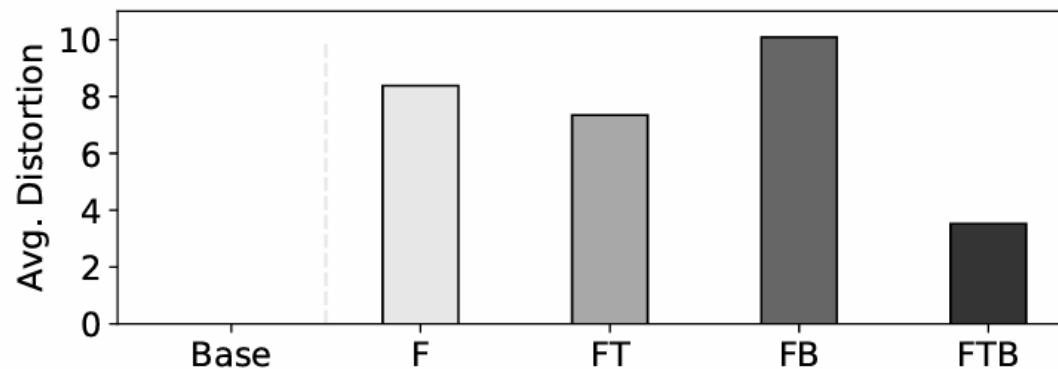


Kết quả của attack model

- Our experiment

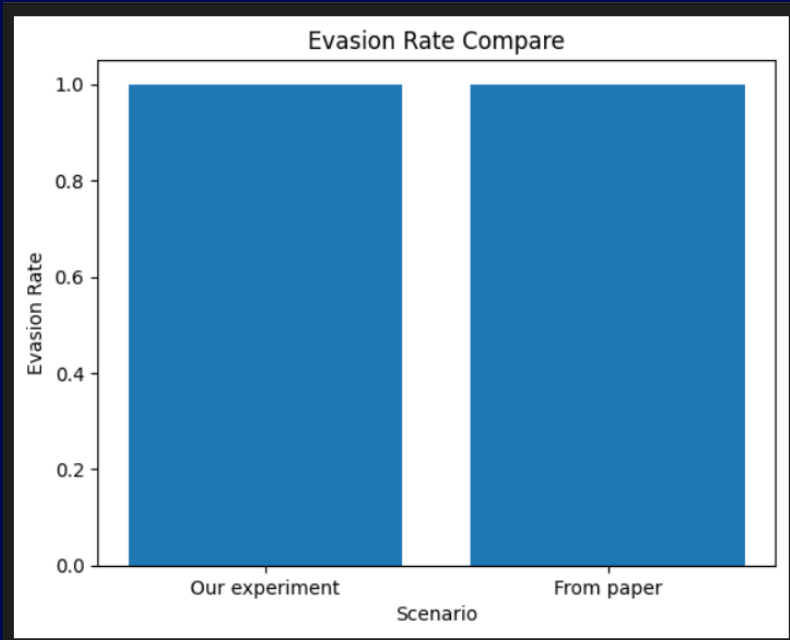


- From paper

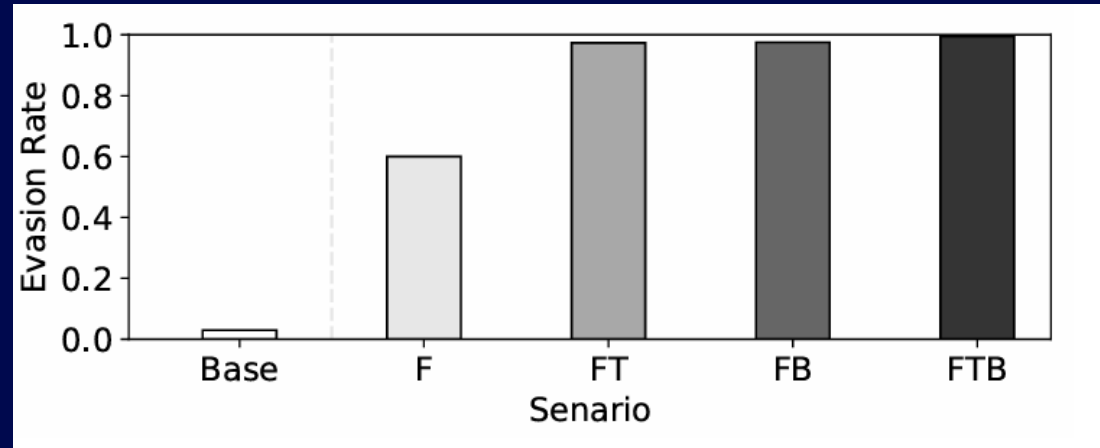


Kết quả của attack model

- Our experiment



- From paper





Nội dung

- Giới thiệu
- Phương pháp thực hiện
- Thực nghiệm và đánh giá
- **Kết luận và hướng phát triển**



Kết luận

- Tạo các mẫu mã độc đối kháng vẫn rất có hiệu quả đối với các trình phát hiện dựa trên machine learning
- JSMA attack tạo ra các mẫu mã độc đối kháng rất nhanh
- Nên sử dụng JSMA attack để tạo ra thêm data để huấn luyện cho mô hình machine learning từ đó cải thiện điểm yếu trước các mẫu mã độc đối kháng



Hướng phát triển

- Tạo ra ứng dụng web có khả năng tạo ra mẫu mã độc đối kháng có khả năng né các trình phát hiện mã độc dựa trên machine learning một cách tự động (chỉ cần bỏ file APK vào web là sẽ cho tải về mẫu mã độc đã được chỉnh sửa)



THANK YOU FOR LISTENING TO OUR
PRESENTATION