A Study on Android Malware Generation using Generative Adversarial Networks

Đoàn Minh Trung[1,2]

[1] University of Information Technology
HCMC, Vietnam

[2] Vietnam National University
HCMC, Vietnam

What ?

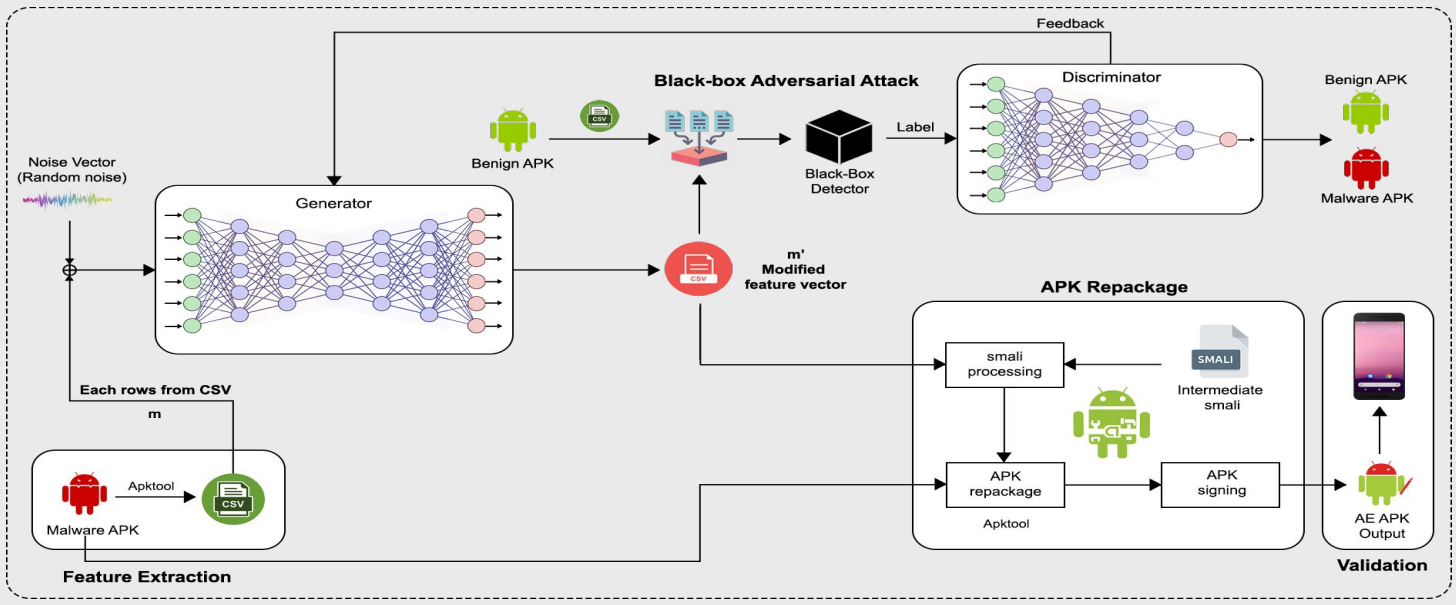
We introduce a framework to automatically generate adversarial examples (AE) without robust detection evasive ability, in which we have:

- Automatically generate adversarial examples (AE) using a generative adversarial network (GAN).
- Modify AE by adding small perturbation only on differentiable features.
- Rebuild AE into new malicious APKs, and validate them by running on the Android Emulator.

Why ?

- Machine learning (ML) algorithms have recently led to many vital breakthroughs in malware detection, they are still particularly vulnerable to AE attacks.
- Recent research has focused on enhancing the effectiveness of GAN-generated AEs to deceive malware classifiers but often overlooks preserving their original functionality. Furthermore, there is a dearth of empirical evidence demonstrating the similarity in behavior between repackaged AEs and the original malware APKs.

Overview



Description

1. Android malware generation system

- Generate AEs using GAN based on 4 static features such as permissions, intent actions, services, categories.
- Modify AEs by adding small perturbations only on differentiable features.
- Evaluate the proximity of the perturbed AEs to the decision boundary of the target classifier.

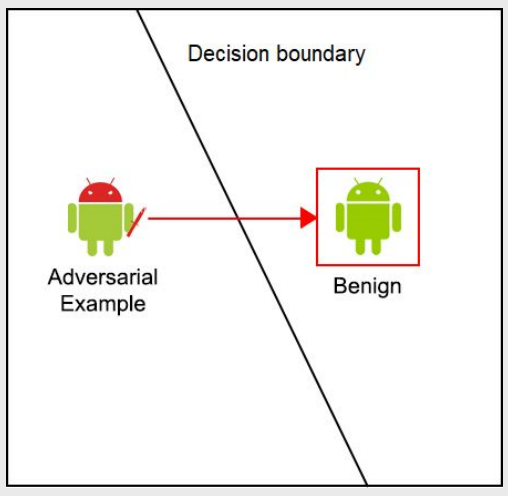


Figure 1.The illustration of AEs.

2. Black-box adversarial attack

- We demonstrate our attack on different state-of-the-art ML classifiers like Support Vector Machines (SVM), Decision Tree (DT), Random Forests (RF), Logistic Regression (LR), eXtreme Gradient Boosting (XgBoost), Convolutional Neural Networks (CNN).
- GAN-generated AEs achieve high attack success rate on those classifiers.

3. Original Functionality Validation

- Rebuild AEs into new malicious APKs, and validate them by running them on the Android emulator.
- We collect and compare the execution log files that include a pseudo-random stream of user events such as touches, clicks, or gestures to ensure they still keep the original functionality as the original malicious APKs.

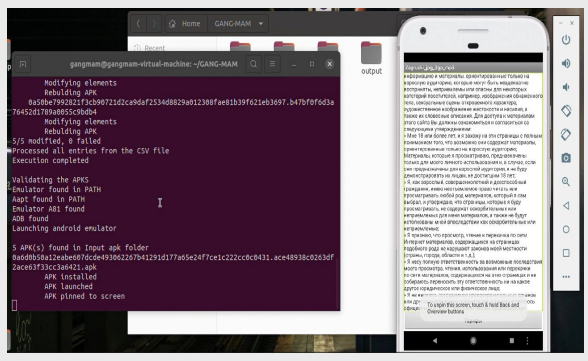


Figure 2. The functionality testing process..

Table 1. the execution details of 10 Android APKs (original and AE).

	APK names	Input	Output	Line Diff	% Diff
1	datiduobao1.17.apk	159	159	3	1.89
2	This-Is-The-Police-2-mod.apk	142	142	1	0.7
3	55you.com_1.0.2.apk	139	139	1	0.72
4	output.165436670.txt	140	140	1	0.71
5	55you.com_sboy2.0.apk	145	145	1	0.69
6	VirusShare_344fa122bf4...	141	141	3	2.13
7	com.youloft.calendar.1905201710.apk	144	141	5	3.47
8	output.158164165.txt	141	139	4	2.84
9	jyhj.apk	54	56	5	9.26
10	xingg.apk	145	145	3	2.07

Đoàn Minh Trung– Trường Đại học Công nghệ thông tin
TEL : 0946456123 Email : trungdm.17@grad.uit.edu.vn