

BÁO CÁO TỔNG KẾT ĐỒ ÁN MÔN HỌC

Môn học: **Cơ chế hoạt động của mã độc**

Tên chủ đề: **Malware mutation**

Mã nhóm: 1 Mã đề tài: S36

Lớp: **NT230.N21.ANTN**

1. THÔNG TIN THÀNH VIÊN NHÓM:

(Sinh viên liệt kê tất cả các thành viên trong nhóm)

STT	Họ và tên	MSSV	Email
1	Nguyễn Bùi Kim Ngân	20520648	20520648@gm.uit.edu.vn
2	Nguyễn Bình Thục Trâm	20520815	20520815@gm.uit.edu.vn
3	Võ Anh Kiệt	20520605	20520605@gm.uit.edu.vn

2. TÓM TẮT NỘI DUNG THỰC HIỆN:¹

A. Chủ đề nghiên cứu trong lĩnh vực Cơ chế hoạt động của mã độc:

- ☐ Phát hiện lỗ hổng bảo mật phần mềm
- ☐ Khai thác lỗ hổng bảo mật phần mềm
- ☐ Sửa lỗi bảo mật phần mềm tự động
- ☐ Lập trình an toàn
- ☒ Khác: Tạo các mẫu mã độc đối kháng

B. Tên bài báo tham khảo chính:

AIMED-RL: Exploring Adversarial Malware Examples with Reinforcement Learning

C. Dịch tên Tiếng Việt cho bài báo:

AIMED-RL: Phát triển các mẫu mã độc đối kháng bằng phương pháp học tăng cường

D. Tóm tắt nội dung chính:

Các mô hình học máy đang được ứng dụng rộng rãi trong việc phân loại phần mềm nhằm phát hiện mã độc. Mặc dù có độ chính xác cao nhưng những mô hình này vẫn có nhược điểm là có thể bị qua mặt bằng cách biến đổi input một cách

¹ Ghi nội dung tương ứng theo mô tả

tin cậy. Mặc dù những biến đổi đó bảo toàn ngữ nghĩa nhưng có thể khiến file bị hư hỏng. Do đó, nhiều hướng tiếp cận đã nghiên cứu về việc tạo các mẫu mã độc đối kháng nhằm khiến các mô hình học máy phân loại mã độc sai, đồng thời vẫn đảm bảo chức năng của file gốc. Nhóm tác giả giới thiệu AIMED-RL, ứng dụng học tăng cường để tự động sửa đổi mã độc thông minh nhằm trốn tránh phát hiện. Theo đó, hướng tiếp cận này có thể tạo các mẫu đối kháng khiến mô hình học máy phân loại sai file mã độc (cụ thể trong bài là PE file) mà không gây ảnh hưởng tới chức năng file. Nhóm tác giả thực hiện bằng cách dùng Distributional Double Deep Q-Network agent để chèn nhiễu (perturbation) vào file gốc. Qua đó, so sánh kết quả nghiệm thu với các nghiệm thu trước cũng ứng dụng học tăng cường đồng thời giảm được số trình tự chuyển đổi cần thiết.

E. Tóm tắt các kỹ thuật chính được mô tả sử dụng trong bài báo:

Bài báo sử dụng học tăng cường để tạo mẫu đối kháng, một số khái niệm quan trọng:

- State: input cho agent, gồm 2 thành phần là các đặc điểm đặc trưng của PE file và các đặc điểm không phụ thuộc cấu trúc
- Agent: đối tượng training, cố gắng đưa ra action tối ưu để tối đa reward nhận được
- Action: mỗi lượt sẽ chọn perturbation chèn vào PE file mà không ảnh hưởng đến chức năng của file
- Episode: gồm đủ 5 turn tức 5 lần chèn perturbation liên tiếp
- Reward: phần thưởng cho mỗi turn, dựa trên output của classifier (malicious or benign) và yếu tố khác, cụ thể là detection (= 0 nếu bị phát hiện và = 10 nếu evasion thành công), similarity (so sánh tỷ lệ giữa kích thước file đã sửa đổi Smod với kích thước file gốc Sorig với Sbest = 40%), distance (thể hiện số turn đã thực hiện qua, min ở turn 1 và max ở turn 5). Công thức tính tổng điểm:

$$R = R_{det} * \omega_{det} + R_{sim} * \omega_{sim} + R_{dis} * \omega_{dis}$$

- ω là trọng số cho mỗi thành phần điểm
- Penalty: nhận phạt bị trừ bớt điểm khi agent chèn trùng perturbation với các turn trước
- Action space: là số loại nhiễu được chọn để chèn, gồm 10 loại: overlay append, imports append, section rename, section add, section append, upx pack, upx unpack, remove signature, remove debug, break optional header checksum.

F. Môi trường thực nghiệm của bài báo:

- Các công cụ hỗ trợ sẵn có:
 - + Cockoo sandbox: kiểm tra chức năng file
- Đối tượng nghiên cứu (chương trình phần mềm dùng để kiểm tra tính khả thi của phương pháp/tập dữ liệu – nếu có)
 - + VirusTotal: 4,187 PE file để training và 200 file khác để kiểm nghiệm
 - + Target model: LGBM
- Tiêu chí đánh giá tính hiệu quả của phương pháp
 - + Evasion rate

G. Kết quả thực nghiệm của bài báo:

Nhóm tác giả thực nghiệm trên 10 agents và được training trên từng các chiến lược khác nhau (có hoặc không có Penalty, đặt trọng số điểm khác nhau, số episode là 1000 và 1500) từ đó đưa ra tỉ lệ né tránh trung bình của 10 agent trên từng chiến lược và tỉ lệ của agent tốt nhất (43.15%). Từ kết quả thì thấy rằng chiến lược Standard (trọng số mỗi thành phần điểm là 0.33), có Penalty và episode là 1500 cho kết quả tốt nhất.

AIMED-RL cũng được so sánh với 3 nghiên cứu khác cùng chủ đề và ứng dụng học tăng cường, các tiêu chí so sánh gồm space (số loại nhiễu), cách tính Reward, số nhiễu chèn trong 1 Episode, target model, functionality test và evasion rate (số liệu của agent tốt nhất). So với 3 bài trước, hướng nghiên cứu này bổ sung Reward similarity, giảm số nhiễu chèn và có tỉ lệ né tránh vượt trội so với 2 bài (1 bài có tỉ lệ là 46.56%).

Ưu:

- + Đạt được kết quả về tỉ lệ né tránh thành công cao
- + Giảm đáng kể số nhiễu chèn, do nhóm tác giả cho rằng thứ tự chèn phù hợp hơn là số lượng, và vẫn đạt kết quả khả quan

Nhược:

- + Nhóm tác giả chưa trình bày cụ thể flow hoạt động (ví dụ bước functionality test được thực hiện khi nào và sẽ làm gì tiếp theo nếu file bị hỏng)

H. Công việc/tính năng/kỹ thuật mà nhóm thực hiện lập trình và triển khai cho demo:

<liệt kê các công việc mà nhóm thực hiện cho đề tài dựa trên phân tích phương pháp/hệ thống được sử dụng trong bài báo đã tham khảo>

Bước 1: thực hiện quá trình xử lý dữ liệu từ file PE để trích xuất ra dữ liệu ở file CSV bao với mỗi file thì có 10 feature và 1 label

Bước 2: thực hiện tạo target model dựa trên dataset vừa trích xuất

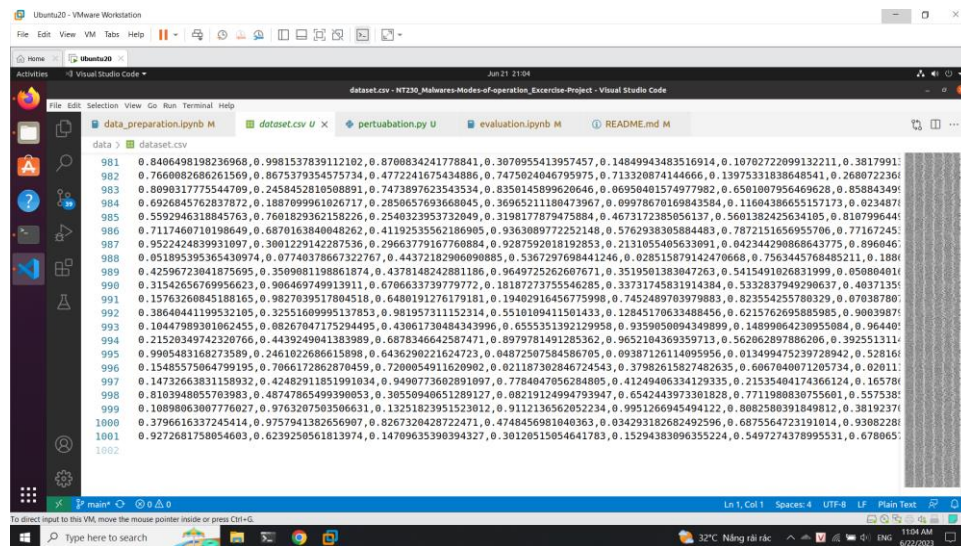
Bước 3: Thực hiện quá trình tạo nhiễu và thực hiện đánh giá kết quả từ việc tạo nhiễu

Bước 4: Thực hiện tạo agent từ những kết quả thu được từ 3 bước trên

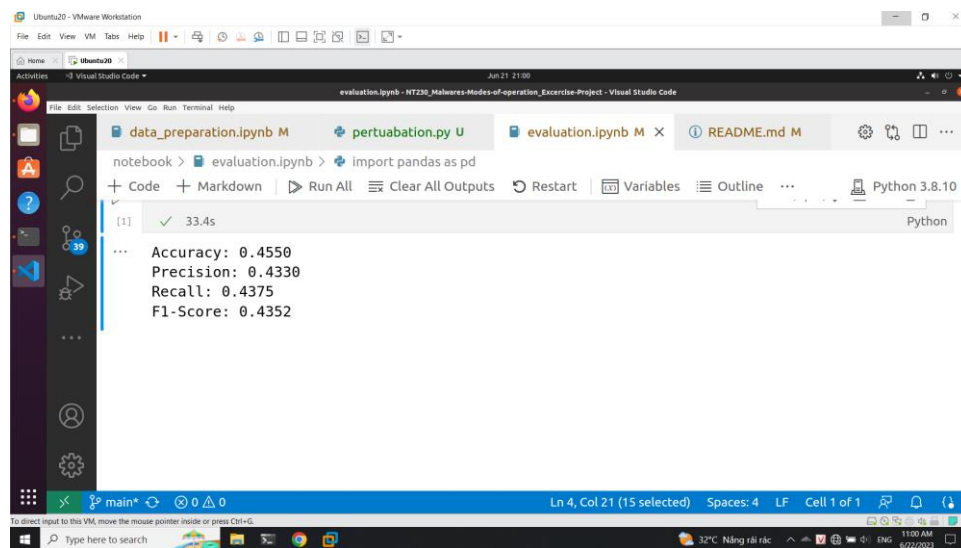
Bước 5: Thực hiện tạo mã độc tự động từ agent và kiểm tra kết quả mã độc thuần chuẩn và sau khi được đột biến từ agent

<liệt kê các công việc đã thực hiện+ tóm tắt kết quả của công việc này>

Bước 1: Đã thực hiện tạo được file CSV của 1000 file PE với 10 feature và 1 label



Bước 2: Đã thực hiện được việc tạo target model và xuất kết quả



Bước 3: Đã thực hiện được quá trình tạo nhiễu và xuất kết quả

```

223
224
225 # Usage example
226 #file_path = "/original/putty_upx_packed.exe"
227 #file_path = "/original/putty.exe"
228 file_path = "/original/x0r.exe"
229 apply_random_perturbation(file_path)

kiet@ubuntu:~/Downloads/NT230_Malwares-Modes-of-operation_Exercise-Project/utis$ python3 perturbation.py
Ultimate Packer for eXecutables
Copyright (C) 1996 - 2023
UPX 4.0.2 Markus Oberhumer, Laszlo Molnar & John Reiser Jan 30th 2023

File size      Ratio      Format      Name
-----
upx: ./original/x0r.exe: FileAlreadyExistsException: ./original/x0r_upx_packed.exe: File exists

Packed 0 files.
Perturbation upx_pack applied to file ./original/x0r.exe
Output file: ./original/x0r_upx_packed.exe
kiet@ubuntu:~/Downloads/NT230_Malwares-Modes-of-operation_Exercise-Project/utis$
    
```

Bước 4 và Bước 5 do còn gặp khó khăn trong việc tạo agent nên chưa thể thực hiện

I. Các khó khăn, thách thức hiện tại khi thực hiện:

- Quá trình đầu có gặp khó khăn khi đọc các thuật ngữ về học tăng cường
- Khó khăn trong quá trình triển khai thuật toán bằng python về học tăng cường

3. TỰ ĐÁNH GIÁ MỨC ĐỘ HOÀN THÀNH SO VỚI KẾ HOẠCH THỰC HIỆN:

75%

4. NHẬT KÝ PHÂN CÔNG NHIỆM VỤ:

STT	Công việc	Phân công nhiệm vụ
1	Nghiên cứu và đọc hiểu thuật toán	Nguyễn Bình Thực Trâm Nguyễn Bùi Kim Ngân Võ Anh Kiệt
2	Hiểu luồng dữ liệu, các thành phần và nhiệm vụ của chúng	Nguyễn Bùi Kim Ngân
3	Nghiên cứu các thí nghiệm được trình bày nhằm giải quyết các câu hỏi nghiên cứu được đề ra	Nguyễn Bình Thực Trâm Nguyễn Bùi Kim Ngân Võ Anh Kiệt
4	Triển khai source code, demo	Võ Anh Kiệt Nguyễn Bình Thực Trâm
5		



BÁO CÁO TỔNG KẾT CHI TIẾT

Phần bên dưới của báo cáo này là tài liệu báo cáo tổng kết - chi tiết của nhóm thực hiện cho đề tài này.

Qui định: Mô tả các bước thực hiện/ Phương pháp thực hiện/Nội dung tìm hiểu (Ảnh chụp màn hình, số liệu thống kê trong bảng biểu, có giải thích)

A. Phương pháp thực hiện

<Trình bày kiến trúc, thành phần của hệ thống trong bài báo>

- State: input cho agent, gồm 2 thành phần là các đặc điểm đặc trưng của PE file và các đặc điểm không phụ thuộc cấu trúc
- Agent: đối tượng training, cố gắng đưa ra action tối ưu để tối đa reward nhận được
- Action: mỗi lượt sẽ chọn perturbation chèn vào PE file mà không ảnh hưởng đến chức năng của file
- Episode: gồm đủ 5 turn tức 5 lần chèn perturbation liên tiếp
- Reward: phần thưởng cho mỗi turn, dựa trên output của classifier (malicious or benign) và yếu tố khác, cụ thể là detection (= 0 nếu bị phát hiện và = 10 nếu evasion thành công), similarity (so sánh tỷ lệ giữa kích thước file đã sửa đổi Smod với kích thước file gốc Sorig với Sbest = 40%), distance (thể hiện số turn đã thực hiện qua, min ở turn 1 và max ở turn 5). Công thức tính tổng điểm:

$$R = R_{det} * \omega_{det} + R_{sim} * \omega_{sim} + R_{dis} * \omega_{dis}$$

- ω là trọng số cho mỗi thành phần điểm
- Penalty: nhận phạt bị trừ bớt điểm khi agent chèn trùng perturbation với các turn trước
- Action space: là số loại nhiễu được chọn để chèn, gồm 10 loại: overlay append, imports append, section rename, section add, section append, upx pack, upx unpack, remove signature, remove debug, break optional header checksum.

<Trình bày kiến trúc, thành phần đã thực hiện (nội dung mà nhóm đã thực hiện)>

- State: Thực hiện xử lý file PE lấy 10 feature và 1 label
- Action: Thực hiện chèn nhiễu bằng function trong python
- Episode: Thực hiện chèn 10 lần chèn perturbation
- Evaluation: Thực hiện đánh giá target model

B. Chi tiết cài đặt, hiện thực

<cách cài đặt, lập trình trên máy tính, cấu hình máy tính sử dụng, chuẩn bị dữ liệu, v.v>

Cấu hình máy:

- CPU: Intel Core i5 8250U (2 nhân 2 luồng)
- Ram: 4GB DDR3L
- HDD: 80GB
- Card mạng: NAT
- OS: Ubuntu 20.04.5

Ngôn ngữ lập trình: python phiên bản 3.8.10

Các thư viện hỗ trợ:

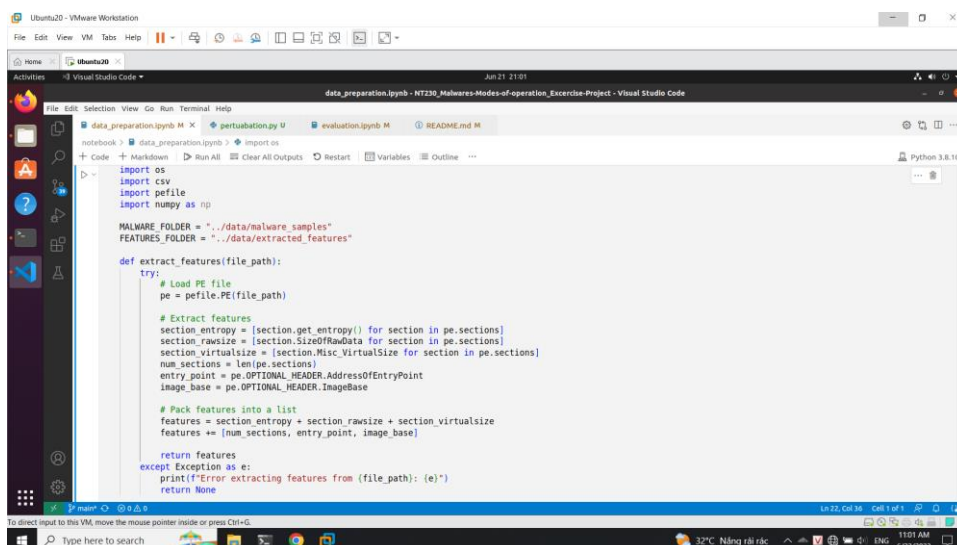
- numpy==1.19.5
- pandas==1.2.1
- scikit-learn==0.24.1
- tensorflow==2.4.0
- matplotlib==3.3.4
- seaborn==0.11.1
- pefile
- csv

Nguồn malware: <https://github.com/ytisf/theZoo>

Nguồn file dataset: <https://github.com/iosifache/DikeDataset>

Lập trình:

Bước 1: Thực code để xử lý dữ liệu từ file PE nhằm lấy 10 feature và 1 label



```

import os
import csv
import pefile
import numpy as np

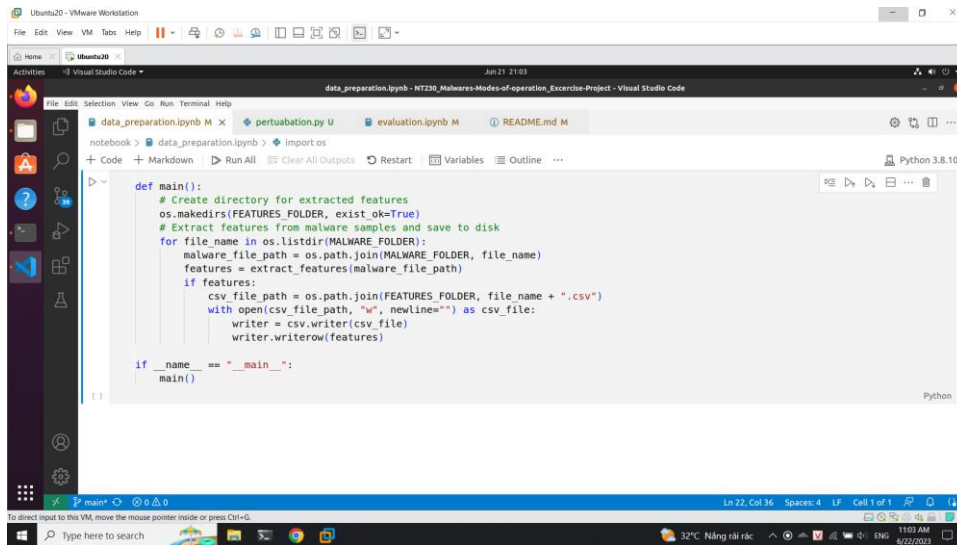
MALWARE_FOLDER = "../data/malware samples"
FEATURES_FOLDER = "../data/extracted_features"

def extract_features(file_path):
    try:
        # Load PE file
        pe = pefile.PE(file_path)

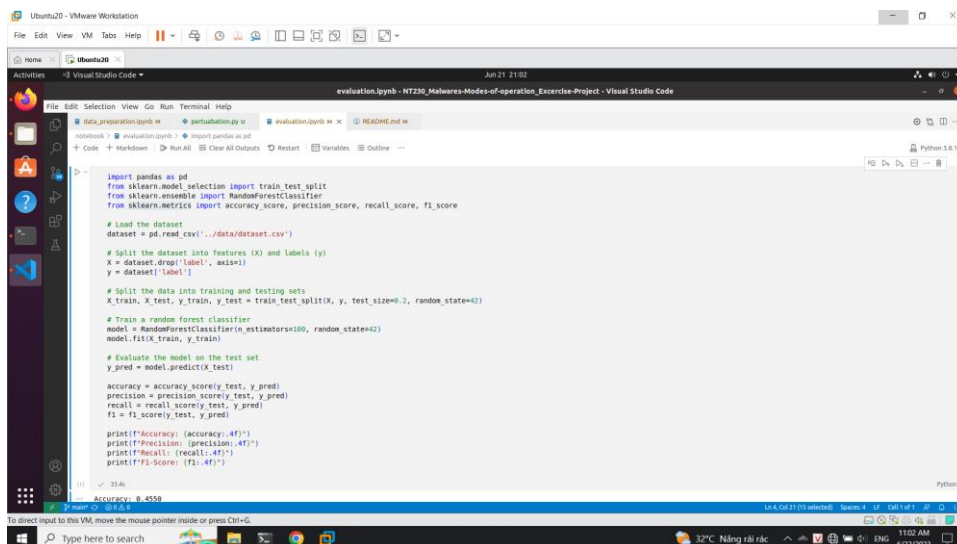
        # Extract features
        section_entropy = [section.get_entropy() for section in pe.sections]
        section_rawsize = [section.SizeOfRawData for section in pe.sections]
        section_virtualsize = [section.Misc_VirtualSize for section in pe.sections]
        num_sections = len(pe.sections)
        entry_point = pe.OPTIONAL_HEADER.AddressOfEntryPoint
        image_base = pe.OPTIONAL_HEADER.ImageBase

        # Pack features into a list
        features = section_entropy + section_rawsize + section_virtualsize
        features += [num_sections, entry_point, image_base]

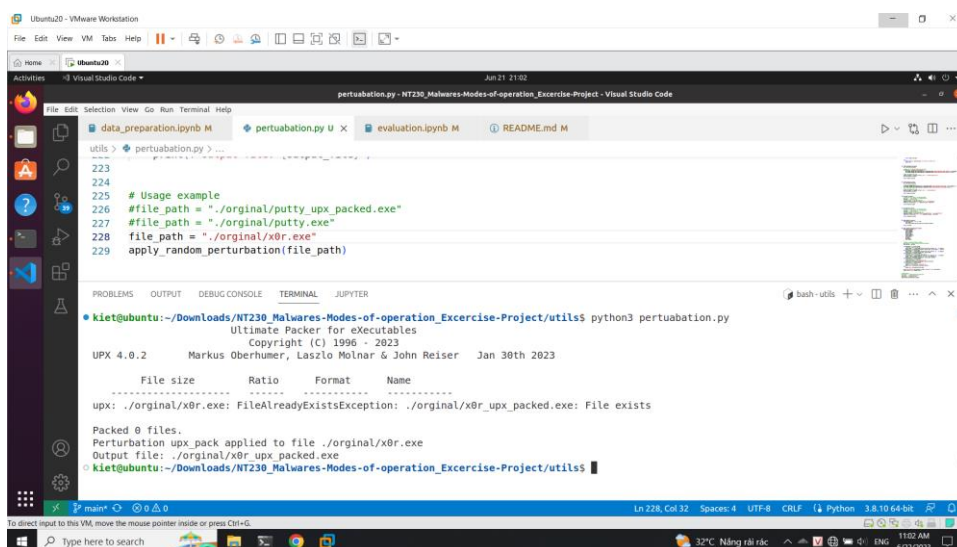
        return features
    except Exception as e:
        print(f"Error extracting features from {file_path}: {e}")
        return None
    
```

Bước 2: Thực hiện việc tạo ra target model



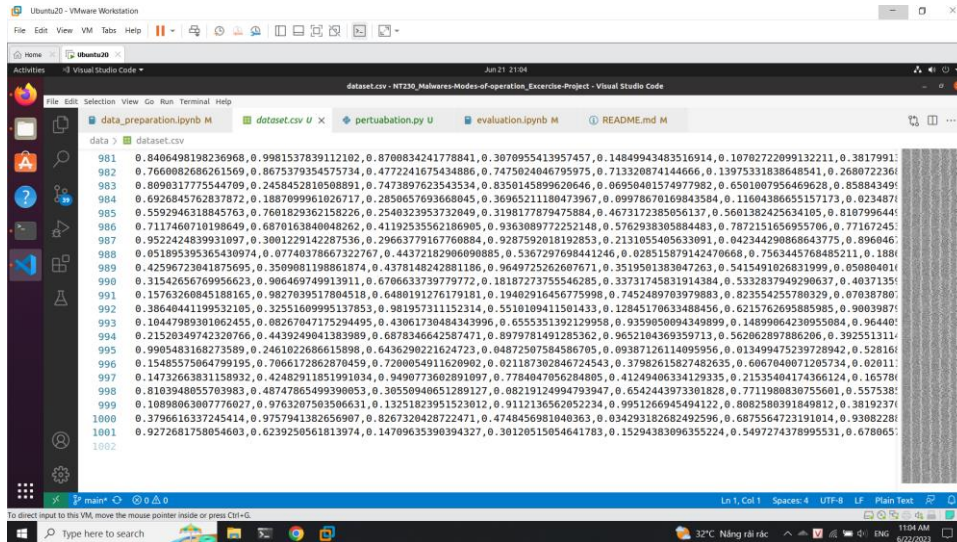
Bước 3: Thực hiện quá trình tạo nhiễu và xuất kết quả



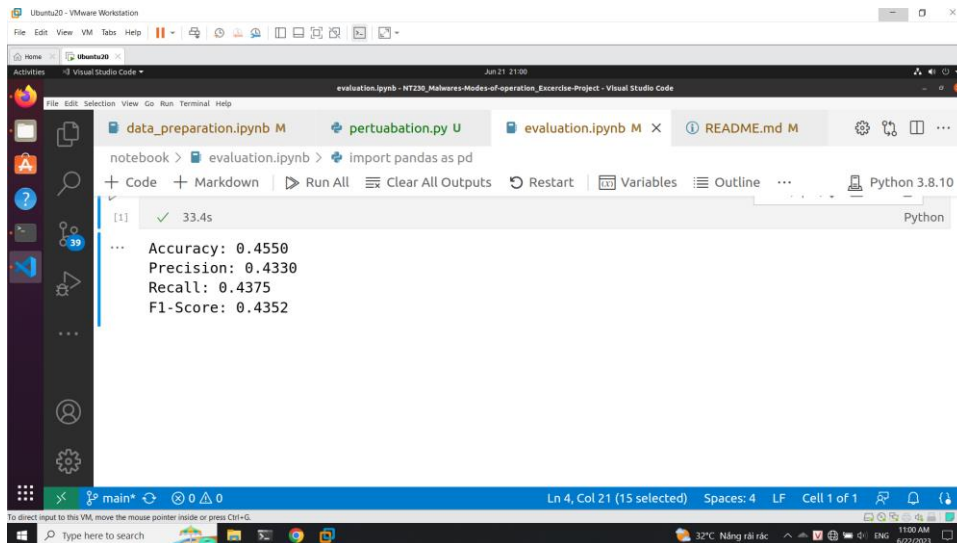
C. Kết quả thực nghiệm

<mô tả hình ảnh về thực nghiệm, bảng biểu số liệu thống kê từ thực nghiệm, nhận xét về kết quả thu được.>

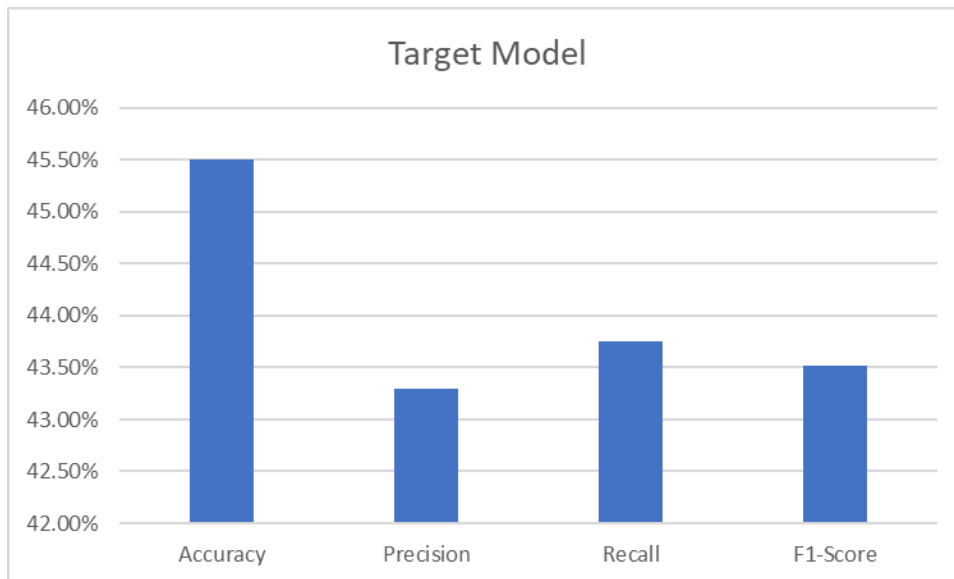
Kết quả bước 1: Phân tích và xử lý 1000 mẫu file PE



Kết quả bước 2: Tạo ra target model và kiểm tra số liệu



Trực quan hoá



Kết quả bước 3: Thực hiện tạo nhiễu và kiểm tra kết quả

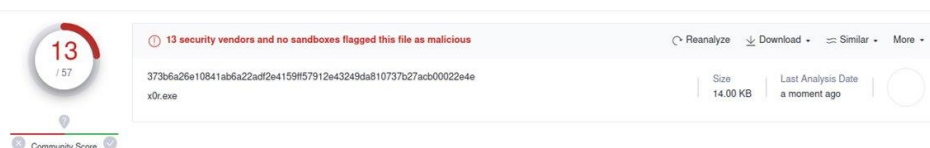


Kiểm tra qua virustotal

Trước khi tạo nhiễu



Sau khi tạo nhiễu



Kết quả trên các phương pháp tạo nhiễu

Method	VirusTotal Score
original	59/71
overlay_append	58/71

imports_append	59/68
section_rename	59/68
section_add	56/71
section_append	54/71
remove_signature	63/71
remove_debug	63/71
upx_pack	50/70
upx_unpack	59/68
break_header	13/57

Tạo nhiễu qua 5 bước

Step 1	Step 2	Step 3	Step 4	Step 5
upx_pack	overlay_append	imports_append	section_append	break_header
50/70	49/71	49/72	45/71	9/59
remove_debug	overlay_append	imports_append	section_rename	break_header
63/71	58/71	58/71	58/71	14/59
imports_append	section_append	remove_signature	upx_pack	break_header
59/68	57/71	57/71	error	error
imports_append	overlay_append	remove_signature	upx_pack	break_header

59/68	58/71	58/71	50/71	12/58
-------	-------	-------	-------	-------

Nhận xét kết quả thực nghiệm:

Thành công trong xử lý dataset

Thực hiện tạo target model nhưng các chỉ số đầu ra vẫn còn thấp (vd: accuracy chỉ ở mức 45.5%)

Thực hiện tạo nhiều manual thì đạt trường hợp 9/59 điểm trên virusTotal nhưng vẫn chưa thể đưa mức điểm về 0 (không bị phát hiện)

D. Hướng phát triển

<Nêu hướng phát triển tiềm năng của đề tài này trong tương lai. Nhận xét về tính ứng dụng của đề tài>.

- Đề tài này hiện đang ứng dụng trên file PE do đó có thể hướng tới thử nghiệm trên các loại file khác như pdf, word

- Đề tài có tính ứng dụng cao trong việc tạo ra mã độc advanced

Sinh viên báo cáo các nội dung mà nhóm đã thực hiện, có thể là 1 phần hoặc toàn bộ nội dung của bài báo. Nếu nội dung thực hiện có khác biệt với bài báo (như cấu hình, tập dữ liệu, kết quả,...), sinh viên cần chỉ rõ thêm khác biệt đó và nguyên nhân.

Sinh viên đọc kỹ yêu cầu trình bày bên dưới trang này

YÊU CẦU CHUNG

- Sinh viên tìm hiểu và thực hiện bài tập theo yêu cầu, hướng dẫn.
- Nộp báo cáo kết quả chi tiết những việc (**Report**) bạn đã thực hiện, quan sát thấy và kèm ảnh chụp màn hình kết quả (nếu có); giải thích cho quan sát (nếu có).
- Sinh viên báo cáo kết quả thực hiện và nộp bài.

Báo cáo:

- File **.PDF**. Tập trung vào nội dung, không mô tả lý thuyết.
- Đặt tên theo định dạng: [Mã lớp]-Project_Final_NhomX_Madetai. (trong đó X và Madetai là mã số thứ tự nhóm và Mã đề tài trong danh sách đăng ký nhóm đồ án).
Ví dụ: [NT521.N11.ANTT]-Project_Final_Nhom03_CK01.
- Nếu báo cáo có nhiều file, nén tất cả file vào file .ZIP với cùng tên file báo cáo.
- Nộp file báo cáo trên theo thời gian đã thống nhất tại courses.uit.edu.vn.

Đánh giá:

- Hoàn thành tốt yêu cầu được giao.
- Có nội dung mở rộng, ứng dụng.

Bài sao chép, trễ, ... sẽ được xử lý tùy mức độ vi phạm.

HẾT