TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

ĐỒ ÁN TỐT NGHIỆP

Giải pháp phân tích mã độc dựa trên MITRE ATT&CK framework và kỹ thuật học sâu

NGUYỄN THÚY HẰNG
hang.nt183523@sis.hust.edu.vn

Ngành Khoa học máy tính
Chuyên ngành Khoa học máy tính

Giảng viên hướng dẫn: PGS.TS. Nguyễn Khanh Văn _____

Chữ kí GVHD

Khoa: Khoa học máy tính

Trường: Công nghệ thông tin và Truyền thông

HÀ NỘI, 08/2022

LỜI CẢM ƠN

Nếu một năm có bốn mùa xuân, hạ, thu, đông thì Bách Khoa có bốn mùa là mùa học, mùa thi, mùa đồ án và mùa để cảm ơn. Trong dòng đầu tiên của trang lời cảm ơn này, mình gửi tới bạn Nguyễn Đức Kiên, lời cảm ơn từ tận đáy lòng. Cảm ơn Kiên đã luôn ở bên mình từ những ngày mà ý tưởng đồ án còn mới chớm nở, cảm ơn Kiên đã luôn hướng dẫn, giúp đỡ và động viên mình vào những thời điểm bế tắc nhất của đồ án. Có những ngày mình khóc và mệt mỏi vì đồ án, cảm ơn Kiên đã kiên nhẫn an ủi và dỗ dành mình.

Lời tiếp theo, em xin được trân trọng gửi tới thầy Nguyễn Khanh Văn, người đã tận tình hướng dẫn để em có thể hoàn thành đồ án một cách tốt nhất. Trong thời điểm mà em vẫn còn thấy mông lung với chính đề tài của mình, em cảm ơn thầy vì đã cho em một câu" thầy nghĩ là đề tài khả thi." Câu nói đó đã trở thành động lực to lớn để em có thể tiếp tục theo đuổi và phát triển đề tài của mình.

Em cũng xin cảm ơn thầy Đặng Tuấn Linh và các thành viên trong Lab, nhờ sự dạy bảo của thầy trong hơn 2 năm qua, nhờ những góp ý của thầy và các bạn mà đồ án của em có thể trở nên hoàn thiện hơn.

Con xin được cảm ơn mẹ vì luôn ủng hộ con theo đuổi ước mơ của mình. Cảm ơn gia đình đã trở thành một điểm tựa vững chắc để con có thể bước thật nhanh và thật xa trên con đường nghiên cứu khoa học.

Cảm ơn bạn Phương Anh, không vì gì cả và vì rất nhiều điều không thể kể tên.

Trong những dòng cuối cùng, tôi gửi lời cảm ơn tới chính bản thân mình vì đã luôn kiên cường trước những khó khăn của cuộc sống, kiên trì theo đuổi ước mơ và kiên định với lựa chọn của bản thân. Dù cho là quá khứ hay tương lai, là chông gai hay là mật ngọt, vẫn luôn cảm ơn cuộc đời vì đã cho tôi được sống, để làm việc và cống hiến, để cho đi mà không cần nhận lại điều gì.

TÓM TẮT NỘI DUNG ĐỒ ÁN

Với thiệt hại lên đến hàng tỉ đô một năm do mã độc gây ra, việc phòng tránh và ngăn chặn mã độc đã trở thành một nhiệm vụ quan trọng trong ngành an ninh thông tin. Để hoàn thành được nhiệm vụ đó thì ta cần hiểu rõ về mã độc cũng như suy nghĩ của kẻ tấn công khi tạo ra các mẫu mã độc đó. Năm 2015, tập đoàn MITRE (Hoa Kỳ) đã đề xuất MITRE ATT&CK framework, là một chuẩn kiến thức chung về các chiến thuật và kỹ thuật mà các nhóm tấn công sử dụng. MITRE ATT&CK framework có ý nghĩa đặc biệt quan trọng với các doanh nghiệp trong việc xác định trước các lỗ hổng trong hệ thống phòng thủ mạng của mình, thay vì tìm hiểu về chúng một cách khó khăn sau mỗi cuộc tấn công. Tuy vậy, trong thực tế, với lượng mã độc lên đến hơn 450000 mẫu mới mỗi ngày thì tốc độ phân tích mã độc từ một đến hai ngày của những nhà nghiên cứu mã độc hiện tại là không đủ nhanh để bắt kịp với sự tinh vi của mã độc, cũng như tốc độ thiệt hại do mã độc gây ra.

Chính vì vậy, đồ án này đề xuất một giải pháp phân tích mã độc dựa trên MITRE ATT&CK framework và kỹ thuật học sâu. Giải pháp gồm có 4 bước bao gồm: (i)thu thập và lưu trữ dữ liệu, (ii) tiền xử lý và vector hóa dữ liệu, (iii)huấn luyện mô hình học máy, (iv)kiểm tra và đánh giá kết quả. Điểm mới trong giải pháp được để xuất trong đồ án này nằm ở hai bước: (i)thu thập và lưu trữ dữ liệu, (ii) tiền xử lý và vector hóa dữ liệu. Trong bước thu thập và lưu trữ dữ liệu, tác giả đã tự thu thập một bộ dữ liệu mới với thông tin của 21000 mẫu mã độc. Trong bước tiền xử lý và vector hóa dữ liệu, dựa trên nền tảng kỹ thuật BoW và đề xuất một phương pháp mới là MISMAC (MIST multiSet based on MITRE ATT&CK) với ý tưởng sắp xếp các phần tử trong multiset của BoW theo các nhóm kỹ thuật trong MITRE ATT&CK framework. Trong bước huấn luyện mô hình học máy, mô hình học máy được tác giả sử dụng là một mạng MLP với các tham số tự đề xuất. Bài toán phân tích mã độc dựa trên MITRE ATT&CK framework là một bài toán phân loại đa nhãn với nhãn đầu ra là các nhóm kỹ thuật của MITRE ATT&CK Framework. Các kết quả và đánh giá được thực hiện độc lập trên bộ dữ liệu tác giả thu thập được và hiện tại chưa có các nghiên cứu tương tự.

Kết quả của mô hình do tác giả đề xuất, đạt độ chính xác tính theo F-score là 96,7% trên tập huấn luyện và 80,5% trên tập kiểm tra. Bên cạnh đó, tác đã thử nghiệm huấn luyện mô hình học máy trong hai trường hợp với bộ dữ liệu có xử lý bằng phương pháp MISMAC và không xử lý bằng phương pháp MISMAC. Độ chính xác của mô hình trong cả hai trường hợp là tương đương nhau. Tuy nhiên khi sử dụng MISMAC thì quá trình huấn luyện mô hình nhanh đạt tới điểm tối ưu hơn.

ABSTRACT

With billions of dollars per year in damage caused by malware, malware prevention and containment have become an important task in the information security industry. To accomplish that task, we need to understand the malicious code as well as the attacker's mindset when creating those malicious code samples. In 2015, the MITRE Corporation(USA) proposed the MITRE ATT&CK framework, which is a globally-accessible knowledge base of adversary tactics and techniques based on real-world observations. The MITRE ATT&CK framework is extremely important for businesses to identify vulnerabilities in their network defenses in advance, instead of learning about them the hard way after each attack. However, in reality, with the amount of malware up to more than 450,000 new samples per day, the one- to two-day malware analysis speed of the current malware researchers is not fast enough to keep up with the sophisticated intelligence. vi of the malware, as well as the rate of damage caused by the malware.

Therefore, this project proposes a malware analysis solution based on MITRE ATT&CK framework and deep learning techniques. The solution consists of 4 steps including (i) data collection and storage, (ii) data preprocessing and vectorization, (iii) machine learning model training, (iv) testing and evaluation. result. The novelty of the proposed solution in this project lies in two steps: (i) data collection and storage, (ii) data preprocessing and vectorization. In the data collection and storage step, the author himself collected a new data set with information of 21000 malicious code samples. In the data preprocessing and vectorization step, the author used the BoW technique and proposed a new method - MISMAC (MIST multiSet based on MITRE ATTCK) with the idea of arranging the elements in the BoW multiset based on the MITRE ATT&CK framework. In the training step of the machine learning model, the machine learning model used by the author is an MLP network with self-recommended parameters. The results and assessments were performed independently on the author's collected dataset; according to the author's knowledge, there are currently no similar researches.

The result of the model proposed by the author, the accuracy of F-score is 96.7% on the training set and 80.5% on the test set. In addition, the author tested the machine learning model training in two cases with the data set with the MISMAC method and the non-MISMAC method. The accuracy of the model in both cases is similar, but when using MISMAC, the model training process quickly reaches the optimal point.

MỤC LỤC

CHƯƠNG 1. GIỚI THIỆU ĐỀ TÀI 1

1.1 Đặt vấn đề 1

1.2 Các giải pháp hiện tại và hạn chế 2

1.3 Mục tiêu và định hướng giải pháp 3

1.4 Đóng góp của đồ án 5

1.5 Bố cục đồ án 5

CHƯƠNG 2. NỀN TẢNG LÝ THUYẾT 7

2.1 MITRE ATT&CK 7

2.1.1 ma trận MITRE ATT&CK 7

2.2 Cấu trúc MIST 15

2.3 Các kỹ thuật vector hóa 17

2.3.1 BoW 17

2.3.2 Multi-hot Encoding 17

2.4 Mạng Multi-Layer Perceptron 18

2.4.1 Kiến trúc cơ bản 18

CHƯƠNG 3. PHƯƠNG PHÁP ĐỀ XUẤT 20

3.1 Mô hình đề xuất tổng quát 20

3.2 Thu thập - lưu trữ dữ liệu 21

3.2.1 Thu thập dữ liệu 21

3.2.2 Lưu trữ dữ liệu 24

3.2.3 Xóa dữ liệu lỗi và Xác nhận dữ liệu đã chính xác 29

3.3 Tiền xử lý - vector hóa dữ liệu 29

3.3.1 Thiết kế kiến trúc MIST 29

3.3.2 Xây dựng multiset của hành vi sử dụng MISMAC 34

3.3.3 Vector hóa MITRE ATT&CK matrix của mã độc 36
3.4 Huấn luyện mô hình học máy..... 39
3.4.1 Đề xuất mô hình học máy 39
3.4.2 Huấn luyện mô hình học máy 39
3.5 Kiểm tra - đánh giá kết quả..... 40
3.5.1 Kiểm tra..... 40
3.5.2 Đánh giá kết quả 41
CHƯƠNG 4. ĐÁNH GIÁ THỰC NGHIỆM..... 43
4.1 Kết quả xây dựng bộ dữ liệu 43
4.1.1 Kích thước bộ dữ liệu 43
4.1.2 Cấu trúc bộ dữ liệu..... 43
4.2 Kết quả xây dựng *MISMAC_multiset* bằng phương pháp MISMAC 46
4.3 Kết quả của mô hình học máy đề xuất..... 46
4.4 Kết quả khi có áp dụng phương pháp MISMAC và không áp dụng phương pháp MISMAC 48
CHƯƠNG 5. KẾT LUẬN 50
5.1 Kết luận..... 50
5.2 Hướng phát triển trong tương lai 51
TÀI LIỆU THAM KHẢO..... 53

DANH MỤC HÌNH VẼ

Hình 1.1 Các bước cơ bản của phương pháp tiếp cận bài toán được đề xuất. 4

Hình 2.1 MITRE ATT&CK Matrix for Enterprise [13] 8
Hình 2.2 Các thành phần của MITRE ATT&CK matrix [13] 8
Hình 2.3 Mô tả sơ đồ của một lệnh MIST [11] 15
Hình 2.4 Ví dụ biểu diễn MIST của một thao tác "load dll" [11] 16
Hình 2.5 Ví dụ Mô hình BoW đối với một câu 17
Hình 2.6 Ví dụ về kỹ thuật Multi-hot Encoding với một tập gồm 5 nhãn 18
Hình 2.7 Kiến trúc cơ bản của một Perceptron 18
Hình 2.8 Kiến trúc cơ bản của mạng Neural nhân tạo 19

Hình 3.1 Tổng quát quy trình hoạt động của giải pháp được đề xuất . . . 20
Hình 3.2 Một phần danh mục lệnh gọi hệ thống trong Joesandbox của báo cáo mã 555555 22
Hình 3.3 Một phần chi tiết lệnh gọi hệ thống trong Joesandbox của báo cáo mã 555555 23
Hình 3.4 Ma trận MITRE ATT&CK của báo cáo mã 555555 trong Joesandbox 23
Hình 3.5 Hộp thoại của Ô kỹ thuật "Scripting" - chiến thuật "Defense Evasion"
báo cáo mã 555555 trong Joesandbox [18] 24
Hình 3.6 Ví dụ về cách lưu trữ MITRE ATT&CK matrix 27
Hình 3.7 Ví dụ cách tham chiếu tên kỹ thuật MITRE ATT&CK đến hành vi tương ứng 34
Hình 3.8 Sơ đồ cách xây dựng *MISMAC_multiset* 35
Hình 3.9 Sơ đồ cách chuyển nội dung file xxx-mitre.json thành *mitre_techniques_vector* 37
Hình 3.10 Ví dụ về cách tạo *mitre_techniques_vector* 38
Hình 3.11 Kiến trúc cơ bản của mạng Neural nhân tạo đề xuất 39
Hình 3.12 Minh họa cách một kết quả biểu diễn trên MITRE ATT&CK® Navigator [20] 40

Hình 4.1 Biểu đồ tỉ lệ định dạng file 44
Hình 4.2 Biểu đồ cột của 20 nhãn kỹ thuật nhiều mẫu nhất 44
Hình 4.3 Mô hình học máy có kiến trúc phù hợp với bộ dữ liệu 48

DANH MỤC BẢNG BIỂU

Bảng 2.1 Danh sách 14 chiến thuật của phiên bản MITRE ATT&CK v11.3 năm 2022 9
Bảng 2.2 Danh sách 1-38/191 kỹ thuật của MITRE ATT&CK v11.3 năm 2022 10
Bảng 2.3 Danh sách 39-64/191 kỹ thuật của MITRE ATT&CK v11.3 năm 2022 11
Bảng 2.4 Danh sách 65-100/191 kỹ thuật của MITRE ATT&CK v11.3 năm 2022 12
Bảng 2.5 Danh sách 101-139/191 kỹ thuật của MITRE ATT&CK v11.3 năm 2022 13
Bảng 2.6 Danh sách 139-178/191 kỹ thuật của MITRE ATT&CK v11.3 năm 2022 14
Bảng 2.7 Danh sách 179-191/191 kỹ thuật của MITRE ATT&CK v11.3 năm 2022 15

Bảng 3.1 Danh sách mã hóa Category và Operation của thiết kế MIST . 33
Bảng 3.2 Các tham số của quá trình huấn luyện 40

Bảng 4.1 Số liệu về định dạng file trong bộ dữ liệu 43
Bảng 4.2 15/191 nhãn kỹ thuật có số lượng nhiều nhất 45
Bảng 4.3 Bảng chiến thuật của 15 kỹ thuật nằm trong nhiều mẫu nhất . 45
Bảng 4.4 Số lượng đối số và số phần tử trong *MISMAC_multiset* 46
Bảng 4.5 Danh sách các nhãn được lựa chọn để huấn luyện và kiểm tra . 47
Bảng 4.6 Kết quả của bốn tập trọng số theo các hàm đánh giá 48
Bảng 4.7 Kết quả của các tập trọng số trong hai trường hợp MISMAC và non-MISMAC 49

DANH MỤC THUẬT NGỮ VÀ TỪ VIẾT TẮT

Thuật ngữ	**Ý nghĩa**
BoW	Phương pháp túi từ (phương pháp Bag-of-words)
Malware	Phần mềm độc hại(Malicious Software)
MISMAC	tập hợp MIST dựa trên MITRE ATT&CK(MIST multiSet based on MITRE ATT&CK)
MIST	Cấu trúc biểu diễn mã độc dựa trên phân tích hành vi (Malware Instruction Set for Behavior-Based Analysis)
MLP	Mô hình Perceptron nhiều lớp (Multi-layer Perceptron)
XML	Ngôn ngữ đánh dấu mở rộng (eXtensible Markup Language)

CHƯƠNG 1. GIỚI THIỆU ĐỀ TÀI

Chương này giới thiệu những vấn đề thực tế dẫn tới lý do chọn đề tài, tổng quan về bài toán tự động phân tích mã độc thành ma trận MITRE ATT&CK cùng một số cách tiếp cận hiện nay. Sau đó đưa ra phạm vi và mục tiêu của đồ án, định hướng giải pháp và bố cục trình bày của đồ án.

1.1 Đặt vấn đề

Mã độc hay còn gọi là phần mềm độc hại (malware/malicious software) là một khái niệm chung dùng để chỉ các phần mềm mà khi thực thi sẽ gây tổn hại tới tài nguyên của hệ thống hoặc chiếm đoạt một phần/toàn bộ quyền điều khiển hệ thống. Các hành vi thường thấy ở mã độc đều là các hành vi bất hợp pháp và gây thiệt hại nghiêm trọng như phá hủy dữ liệu, phần cứng; nghe trộm hoạt động của người dùng trên các thiết bị vào ra (Keylogging); đánh cắp thông tin (Spyware); mã hóa dữ liệu để tống tiền (Ransomware); đánh cắp tài nguyên tính toán (Coinminer); tạo cửa hậu (Backdoor) để kẻ tấn công xâm nhập và điều khiển thiết bị; ...

Theo the AV-TEST Institute, mỗi ngày có hơn 450000, mã độc mới được phát hiện và 99% trong số đó chỉ được nhìn thấy một lần trước khi được sửa đổi và sử dụng lại [1]. Và theo báo cáo của Cybersecurity Ventures, chỉ tính riêng năm 2021, thiệt hại do mã độc gây ra là 20 tỉ đô và con số này sẽ là 42 tỉ đô trong năm 2024 [2]. Dưới tốc độ tấn công và thiệt hại tăng nhanh do mã độc gây ra, khi mà mỗi con mã độc lại có một đặc điểm riêng, những chuyên gia bảo mật gặp rất nhiều khó khăn trong việc nghiên cứu và phân tích mã độc. Bên cạnh đó, các công ty, doanh nghiệp cũng gặp nhiều khó khăn trong việc bảo vệ hệ thống khỏi các cuộc tấn công.

Để bắt kịp với các cuộc tấn công không ngừng như vậy, năm 2015, tập đoàn MITRE - The Mitre Corporation đã đề xuất mô hình MITRE ATT&CK (Adversarial Tactics Techniques & Common Knowledge). Theo định nghĩa của tập đoàn MITRE, MITRE ATT&CK là "một cơ sở kiến thức có thể truy cập toàn cầu về các chiến thuật và kỹ thuật của đối thủ dựa trên những quan sát trong thế giới thực" [3]. Dưới góc nhìn của các chuyên gia an ninh mạng, MITRE ATT&CK là một tập hợp có thể biểu diễn dưới dạng một ma trận 3 chiều, trong đó có chứa tất cả các chiến thuật, kỹ thuật và quy trình mà những kẻ tấn công mạng thường sử dụng. Bằng cách phân loại các cuộc tấn công thành các phân đoạn cụ thể, các chuyên gia phân tích mã độc sẽ dễ dàng hơn trong việc xem các mẫu, tìm ra kẻ đã phát động các cuộc tấn công khác nhau và theo dõi cách một phần mềm độc hại đã phát triển theo thời gian. MITRE ATT&CK có ý nghĩa quan trọng với các doanh nghiệp trong việc xác

định trước các lỗ hổng trong hệ thống phòng thủ mạng của mình, thay vì tìm hiểu về chúng một cách khó khăn sau mỗi cuộc tấn công.

Để có thể phân tích một mã độc ra MITRE ATT&CK, đầu tiên, các chuyên gia phân tích mã độc cần đưa mã độc vào các công cụ phân tích như IDA[4], Binary Ninja[5], sandbox... để chuyển mã độc thành các dạng như tập hành vi(behaviors), mã Assembly,... .Một hành vi của mã độc (behavior) là một thao tác mà mã độc yêu cầu hệ thống hay hệ điều hành thực hiện, ví dụ như mở file, ghi file, kết nối với địa chỉ IP, ... Một mã độc có thể có từ hàng trăm tới hàng nghìn hành vi và chỉ một vài hành vi trong số đó là hành vi độc hại (malicious behavior). Sau khi chuyển mã độc từ mã máy về các dạng dễ đọc hơn như tập hành vi hay mã Assembly , các chuyên gia cần phải đọc kỹ từng hành vi mà mã độc đã thực hiện, từng chi tiết trong mã Assembly của mã độc, từ đó phát hiện ra các hành vi độc hại và ghi lại vào ma trận MITRE ATT&CK của mã độc đó. Công việc này mất từ 1 đến 2 ngày đối với một mã độc đơn giản và khoảng 1 đến 2 tuần đối với các mã độc phức tạp. Như vậy là chưa đủ nhanh với độ tinh vi của mã độc hiện tại, tốc độ mà mã độc mới được phát hiện trên không gian mạng, cũng như tốc độ thiệt hại do mã độc gây ra. Từ những vấn đề đó, một bài toán quan trọng được đặt ra là làm thế nào để phân tích MITRE ATT&CK của mã độc một cách nhanh chóng và chính xác.

Trong những năm trở lại đây, mạng noron nhân tạo đã trở thành một xu hướng công nghệ mới hàng đầu. Multi Layer Perceptron (MLP) là thuật ngữ dùng để chỉ chung các kiến trúc mạng noron nhân tạo kết nối đầy đủ có nhiều hơn 1 lớp ẩn. MLP thường được sử dụng trong các bài toán có tập dữ liệu phức tạp và phân bố phi tuyết tính. Bằng cách sử dụng kiến trúc MLP, các bài toán như: Phát hiện phần mềm độc hại Android dựa trên quyền với MLP [6], Phân loại phần mềm độc hại bằng thuật toán học máy [7] , Phát hiện phần mềm độc hại bằng học máy [8], ... đã được giải quyết tốt. Điều này thể hiện rằng, bằng cách sử dụng các mô hình học máy, ta có thể trích xuất các thông tin quan trọng của mã độc thông qua mã nguồn hay cách hoạt động. Từ đó, có thể thấy được tính khả thi của việc ứng dụng học máy nói chung và kiến trúc MLP nói riêng trong các phân tích liên quan đến mã độc.

Trong đồ án này , tôi sẽ đề xuất một giải pháp phân tích mã độc dựa trên MITRE ATT&CK framework và kỹ thuật học sâu, mục tiêu và phạm vi của đề tài sẽ được trình bày trong mục 1.3.

1.2 Các giải pháp hiện tại và hạn chế

Theo như tìm hiểu của tôi, hiện nay trên thế giới đã có một vài công trình nghiên cứu có nội dung gần tương tự với đề tài mà tôi đề xuất như phân tích hành vi phần

mềm độc hại với ATT&CK Framework[9], phân loại chiến thuật và kỹ thuật trong thông tin mối đe dọa an ninh mạng [10], ...Trong các nghiên cứu này, tôi nhận thấy rằng có một số phương pháp xử lý dữ liệu thường được sử dụng trong các dạng bài toán phân tích mã độc theo hành vi là chuyển tập hành vi sang dạng MIST (Malware Instruction Set for Behavior-Based Analysis) , sau đó sử dụng phương pháp N-grams để vector hóa tập MIST.

MIST là một phương pháp biểu diễn cho hành vi được giám sát của mã độc được phát triển từ năm 2010 [11]. Việc chuyển tập hành vi sang dạng MIST là một cách hiệu quả để mã hóa và định danh hành vi. Sau khi áp dụng MIST, tập hành vi sẽ trở lên ngắn gọn hơn, có dung lượng nhỏ hơn và dễ để sử dụng hơn trong quá trình vector hóa. Tuy nhiên, do đề xuất về MIST đã có từ năm 2010 nên nhiều hành vi mới hiện nay vẫn chưa được cập nhật trong MIST. Do đó, hiện nay đã xuất hiện sự khác nhau giữa cấu trúc MIST của nghiên cứu gốc và cấu trúc MIST của các công cụ hỗ trợ khác.

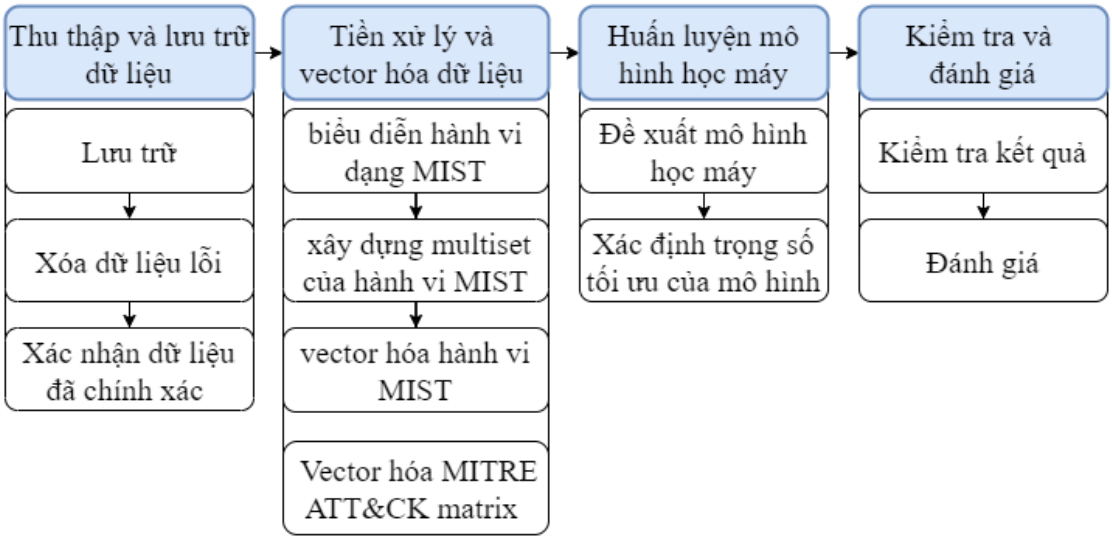
Một cụm N-grams là 1 dãy con gồm n phần tử liên tiếp nhau của 1 dãy các phần tử cho trước. Việc sử dụng N-grams để vector hóa tập MIST là một phương pháp phù hợp để biểu diễn dữ liệu tuần tự. Các nghiên cứu mã độc sử dụng N-grams đã xuất hiện trong các nghiên cứu từ năm 2011 [12] và vẫn được sử dụng rộng rãi cho tới nay. Ưu điểm của N-grams là có thể biểu diễn được sự liên kết về thứ tự hành vi của mã độc theo thời gian. Tuy vậy, đó cũng là điểm yếu của thuật toán này nếu như có mẫu mã độc cố tình thêm vào các hành vi gây nhiễu.

Trong trường hợp N>=2, việc thêm vào các hành vi gây nhiễu không chỉ làm giảm hiệu quả của phương pháp N-grams, làm mất nhiều tài nguyên tính toán, mà nó còn có thể khiến mã độc "qua mặt" cả bước vector hóa tập MIST. Điều này dẫn tới việc mô hình học máy đưa ra những kết quả không chính xác về mã độc. Đáng lo ngại hơn đó là với sự tinh vi của mã độc hiện nay thì việc thêm vào các hành vi gây nhiễu là điều quá dễ dàng với những kẻ tấn công.

1.3 Mục tiêu và định hướng giải pháp

Như đã trình bày trong mục 1.1, nhằm giải quyết bài toán đặt ra, mục tiêu của đồ án là xây dựng một phương pháp tiếp cận để giải quyết bài toán ánh xạ hành vi của mã độc sang ma trận MITRE ATT&CK sử dụng kiến trúc MLP. Trong bài toán này, loại mã độc mà tôi lựa chọn để làm mẫu là mã độc trên hệ điều hành Windows. Do những hạn chế của các phương pháp được để cập ở phần 1.2, đồ án này sẽ tự xây dựng một cấu trúc MIST tham khảo nền tảng của cấu trúc MIST được đề xuất năm 2010 [11], đề xuất một phương pháp mới để vector hóa tập hành vi MIST dựa trên MITRE ATT&CK và nền tảng phương pháp BoW(Bag-of-words). Kiến trúc

MLP được sử dụng là một kiến trúc tự đề xuất, sẽ được trình bày chi tiết ở chương ba. Phương pháp được đề xuất gồm có 4 bước: thu thập và lưu trữ dữ liệu, tiền xử lý và vector hóa dữ liệu, huấn luyện mô hình học máy, kiểm tra và đánh giá kết quả.



Hình 1.1: Các bước cơ bản của phương pháp tiếp cận bài toán được đề xuất.

Trong bước thu thập và lưu trữ giữ liệu, dữ liệu được thu thập từ trang web Joesandbox.com bằng công cụ chính là thư viện selenium của ngôn ngữ lập trình Python. Joesandbox(Joe Sandbox Cloud Basic v35.0.0 Citrine) là một sản phẩm phân tích phần mềm độc hại của công ty Joe Security LLC, trên đó có lưu trữ những bản báo cáo phân tích chuyên sâu về mã độc. Trong quá trình thu thập và lưu trữ, một số lỗi khách quan hoặc chủ quan có thể xảy ra khiến một vài mẫu dữ liệu không được đầy đủ thông tin. Do đó, tôi thực hiện nhiệm vụ xóa dữ liệu lỗi và xác nhận bộ dữ liệu đã chính xác.

Trong bước tiền xử lý và vector hóa dữ liệu, ba công việc chính cần thực hiện là (i)chuyển tập hành vi của mã độc sang dạng MIST, (ii)xây dựng multiset của hành vi và (iii)vector hóa MITRE ATT&CK matrix của mã độc. Công việc tiền xử lý được thực hiện xuyên suốt bước hai với các nhiệm vụ như xóa các thông tin trùng lặp và loại bỏ các lỗi được phát hiện trong quá trình vector hóa. Trong bước biểu diễn hành vi của mã độc dưới dạng MIST, tôi tự xây dựng cấu trúc MIST mới theo dữ liệu thu thập được, hành vi trong tập hành vi định dạng json sẽ được biểu diễn dưới dạng cấu trúc MIST và lưu lại dưới định dạng txt. Tiếp đó là bước xây dựng multiset của hành vi theo phương pháp BoW. Thông qua chữ ký, một hoặc nhiều hành vi MIST sẽ được nối với một kỹ thuật trong MITRE ATT&CK. Khi xây dựng multiset, các hành vi có cùng kỹ thuật sẽ được sắp xếp cạnh nhau thành một nhóm. Cách xây dựng multiset trên được tôi đặt tên là MISMAC (MIST multiSet based on MITRE ATT&CK). Hành vi của mã đọc sẽ được vector hóa dựa trên tần suất

xuất hiện trong multiset. Ở bước vector hóa MITRE ATT&CK matrix của mã độc, MITRE ATT&CK matrix của mã độc sẽ được chuyển thành dạng vector để biểu diễn nhãn của mã độc bằng phương pháp Multi-hot Encoding. Phương pháp sắp xếp, vector hóa và lưu trữ kết quả sẽ được trình bày chi tiết ở chương 3.

Trong bước huấn luyện mô hình học máy, tôi huấn luyện một mô hình với kiến trúc MLP tự đề xuất. Dữ liệu được chia làm 2 bộ huấn luyện và kiểm tra với tỉ lệ 7:3. Kết quả thu được là trọng số tối ưu của mô hình học máy và kết quả này sẽ được sử dụng trong bước kiểm tra và đánh giá kết quả.

Trong bước kiểm tra và đánh giá kết quả, tôi thực hiện hai nhiệm vụ so sánh kết quả giữa các model đề xuất và so sánh kết quả của hai trường hợp: (i) sử dụng phương pháp MISMAC và (ii) không sử dụng phương pháp MISMAC.

Bài toán phân tích hành vi của mã độc dựa trên MITRE ATT&CK Framework sử dụng kiến trúc MLP là một bài toán học máy phân loại đa nhãn với dữ liệu đầu vào là hành vi đã vector hóa của mã độc và mục tiêu cần dự đoán là nhãn kỹ thuật trong MITRE ATT&CK của mã độc.

1.4 Đóng góp của đồ án

Đồ án này có 4 đóng góp chính như sau:

1. Thiết kế một giải pháp hoàn thiện cho bài toán phân tích mã độc dựa trên MITRE ATT&CK framework và kỹ thuật học sâu.
2. Thu thập bộ dữ liệu gồm 21000 mẫu mã độc có thể sử dụng trong nhiều bài toán như phân tích mã độc, phân loại mã độc.
3. Đồ án đề xuất một phương pháp mới có tên MISMAC(MIST multiSet based on MITRE ATT&CK) với ý tưởng dựa trên nền tảng phương pháp BoW, sắp xếp phần tử là hành vi MIST của mã độc trong multiset của BoW theo các nhóm kỹ thuật trong MITRE ATT&CK.
4. Đề xuất mô hình học máy để huấn luyện cho bộ dữ liệu đã được xử lý trong bài toán.

1.5 Bố cục đồ án

Dựa vào các vấn đề tồn tại và hướng tiếp cận được đề xuất để giải quyết bài toán ở phần trước, các phần tiếp theo của đồ án được tổ chức như sau.

Chương 2 giới thiệu về các cơ sở lý thuyết được sử dụng trong hai nhiệm vụ tiền xử lý - vector hóa dữ liệu và huấn luyện mô hình học máy. Chương này sẽ bao gồm 3 phần chính. Phần đầu sẽ trình bày cấu trúc của ma trận MITRE ATT&CK, phần tiếp theo trình bày về cấu trúc MIST và cách chuyển tập hành vi của mã độc sang

dạng MIST, phần tiếp nữa trình bày về các phương pháp vector hóa dữ liệu và phần cuối cùng trình bày các kiến thức cơ bản của mạng MLP.

Chương 3 trình bày chi tiết về mô hình đề xuất tổng quát trong phần 3.1 và 4 bước của hướng tiếp cận đề xuất bao gồm: (i) thu thập - lưu trữ dữ liệu trong phần 3.2, (ii) tiền xử lý - vector hóa dữ liệu trong phần 3.3, (iii) huấn luyện mô hình học máy trong phần 3.4, (iv) kiểm tra - đánh giá kết quả trong phần 3.5. Trong đó, phương pháp MISMAC xây dựng $MISMAC\_multiset$ trong phần tiền xử lý - vector hóa dữ liệu là điểm tạo ra sự khác biệt giữa hướng tiếp cận được đề xuất và các hướng tiếp cận đã có hiện tại, kiến trúc MLP là nền tảng cho mô hình học máy được đề xuất trong phần huấn luyện mô hình học máy.

Chương 4 trình bày về các kết quả thực nghiệm từ phương pháp đề xuất ở chương 3. Chương này gồm ba phần: (i)Kết quả xây dựng bộ dữ liệu, (ii) Kết quả xây dựng *MISMAC_multiset* bằng phương pháp MISMAC, (iii) So sánh kết quả của một số mô hình học máy tự đề xuất và (iv) So sánh kết quả khi có áp dụng phương pháp MISMAC và không áp dụng phương pháp MISMAC.

Chương 5 tổng kết các kết quả đạt được cũng như trình bày các hạn chế còn tồn tại, đồng thời định hướng phát triển trong tương lai.

CHƯƠNG 2. NỀN TẢNG LÝ THUYẾT

Ở chương một, đồ án đã mô tả các khó khăn và thách thức và một số hướng tiếp cận với bài toán phân tích mã độc trong đồ án này. Chương này trình bày những thuật ngữ cơ bản, những kĩ thuật cơ bản được sử dụng để giải quyết bài toán. Chương này sẽ giúp người đọc có một cái nhìn tổng thể về các kĩ thuật được sử dụng, là tiền đề để cho các kỹ thuật và mô hình đề xuất được trình bày ở chương tiếp theo.

2.1 MITRE ATT&CK

2.1.1 ma trận MITRE ATT&CK

MITRE ATT&CK là "một cơ sở kiến thức có thể truy cập toàn cầu về các chiến thuật và kỹ thuật của đối thủ dựa trên những quan sát trong thế giới thực" [3]. Dưới góc nhìn của các chuyên gia an ninh mạng, MITRE ATT&CK là một tập hợp có thể biểu diễn dưới dạng một ma trận 3 chiều, trong đó có chứa tất cả các chiến thuật (Tactics), kỹ thuật (Techniques) và chi tiết kỹ thuật (Sub-Techniques) mà những kẻ tấn công thường sử dụng.

Chiến thuật là các những hành động hoặc bước cụ thể mà kẻ tấn công thực hiện để hoàn thành chiến lược của mình. Đó là mục đích để kẻ tấn công thực hiện một hành vi. Ví dụ như với chiến thuật mã TA0001 - "Initial Access": Kẻ tấn công thực hiện hành vi lừa đảo qua các trang web giả mạo nhằm chiếm được "Quyền truy cập đầu".

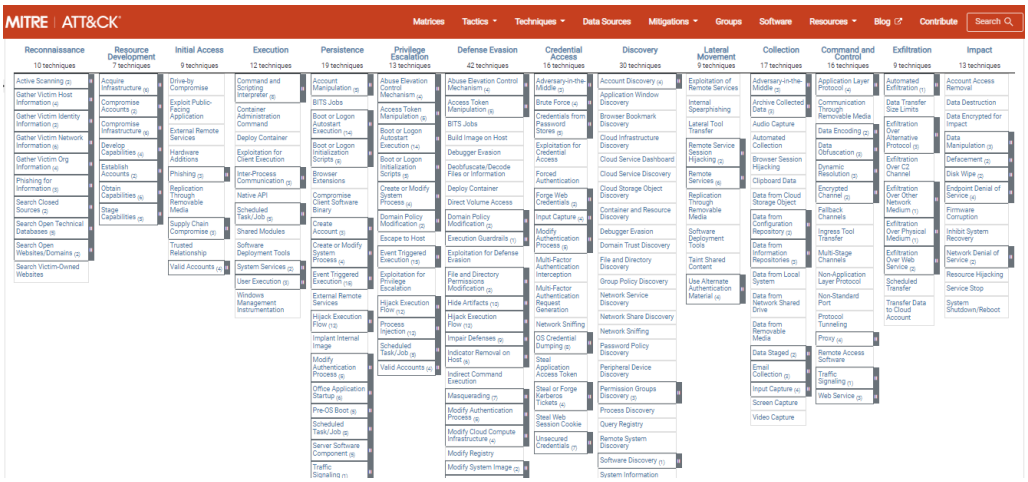
Kỹ thuật (Techniques) là cách mà kẻ tấn công hành động để đạt được mục tiêu của phần chiến thuật. Ví dụ như với kỹ thuật mã T1110 - "Bruce Force": Kẻ tấn công thực hiện kỹ thuật "Bruce Force" nhằm đạt được mục đích là có thông tin đăng nhập của chiến thuật TA0006 - "Credential Access".

Chi tiết kỹ thuật, còn được gọi là phụ lục kỹ thuật, là phân loại chi tiết hơn của kỹ thuật. Ví dụ như với kỹ thuật mã T1110.001 - "Brute Force: Password Guessing" : Kẻ tấn công thực hiện hành động "đoán mật khẩu" trong kỹ thuật Bruce Force nhằm nhằm đạt được mục đích là chiến thuật TA0006 - "Thông tin đăng nhập".

Enterprise Matrix, tên đầy đủ là MITRE ATT&CK Matrix for Enterprise, là biểu diễn của MITRE ATT&CK dưới dạng ma trận ba chiều. Trong đó, chiều đầu tiên là hướng chiến thuật có 14 giá trị bao gồm 14 chiến thuật, chiều thứ hai là chiều kỹ thuật bao gồm 191 kỹ thuật khác nhau và chiều thứ 3 là chiều chi tiết kỹ thuật của 191 kỹ thuật ở chiều thứ hai. Trong Enterprise Matrix, mỗi chiến thuật là duy nhất, không trùng lặp và có nhiều kỹ thuật. Chiến thuật $id_{tactics}$ có $N\_techniques_{id_{tactics}}$

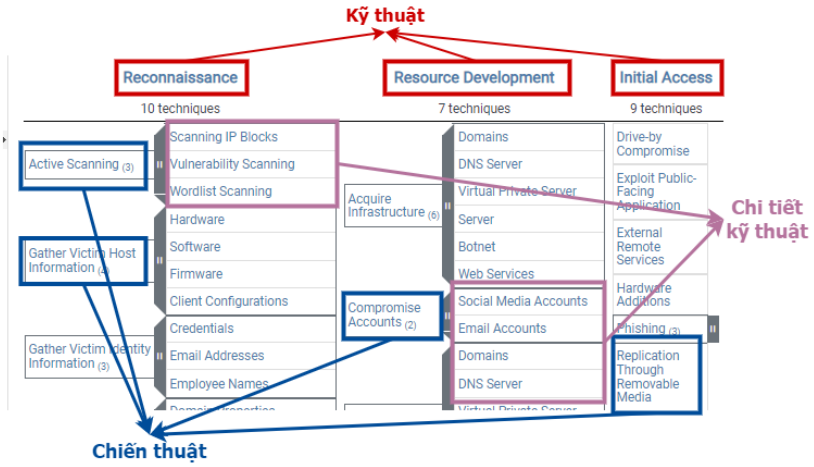
kỹ thuật ($N\_techniques_{id_{tactics}}$ >= 1). Mỗi kỹ thuật có thể có hoặc không nhiều chi tiết kỹ thuật và có thể thuộc về nhiều hơn một chiến thuật. Kỹ thuật $id_{technis}$ có $N\_subtechniques_{id_{techniques}}$ chi tiết kỹ thuật ($N\_subtechniques_{id_{techniques}}$ >= 0). Trong phiên bản ATT&CK v11.3, Website v3.6.4, Enterprise Matrix bao gồm 14 chiến thuật, 191 kỹ thuật và 385 chi tiết kỹ thuật khác nhau. Nếu tính cả các kỹ thuật bị trùng lặp trong các chiến thuật khác nhau thì Enterprise Matrix ATT&CK v11.3 có 222 kỹ thuật.

Hình 2.1 là hình ảnh tổng quát của Enterprise Matrix, với dòng đỏ trên cùng là thanh công cụ, phần bên dưới là ma trận rút gọn gồm 14 cột tương ứng với 14 kỹ thuật và các ô trong mỗi cột tương ứng với các kỹ thuật trong mỗi chiến thuật.



Hình 2.1: MITRE ATT&CK Matrix for Enterprise [13]

Hình 2.2 là hình ảnh chi tiết về một phần của Enterprise Matrix. Các vị trí tương ứng với các ô màu đỏ là các chiến thuật, các vị trí tương ứng với các ô màu xanh là các kỹ thuật, các vị trí tương ứng với các ô màu tím là các chi tiết kỹ thuật.



Hình 2.2: Các thành phần của MITRE ATT&CK matrix [13]

$id_{tactics}$	Tên chiến thuật	Mô tả	$N\_techniques\ id_{tactics}$
TA0043	Reconnaissance	Thu thập thông tin để lập kế hoạch hoạt động trong tương lai như: chi tiết về tổ chức của đơn vị bị tấn công, cơ sở hạ tầng, hoặc nhân viên/ người sử dụng.	10
TA0042	Resource Development	Tạo, mua hoặc xâm phạm/ đánh cắp tài nguyên có thể được sử dụng để tấn công mục tiêu trong tương lai như: cơ sở hạ tầng, tài khoản hoặc quyền hạn.	7
TA0001	Initial Access	Xâm nhập vào mạng lưới của đơn vị bị tấn công và đạt được những bị trí ban đầu trong mạng lưới.	9
TA0002	Execution	Thực thi mã độc trên hệ thống.	12
TA0003	Persistence	Giữ quyền truy cập vào hệ thống khi hệ thống khởi động lại, thay đổi thông tin đăng nhập hay xảy ra bất cứ sự cố nào làm gián đoạn quyền truy cập đã chiếm được.	19
TA0004	Privilege Escalation	Giành quyền cấp cao hơn trong hệ thống như SYSTEM/root level, quản trị viên, tài khoản người dùng có các quyền đặc biệt.	13
TA0005	Defense Evasion	Tránh bị phát hiện trong suốt quá trình tấn công.	42
TA0006	Credential Access	Lấy cắp các thông tin xác thực như tên đăng nhập và mật khẩu.	16
TA0007	Discovery	Có được kiến thức về hệ thống và mạng nội bộ của nạn nhân.	30
TA0008	Lateral Movement	Xâm nhập và điều khiển các hệ thống từ xa bằng những kỹ thuật che dấu tốt hơn.	9
TA0009	Collection	Thu thập các thông tin như ổ đĩa, trình duyệt, tệp âm thanh, video, email, ...	17
TA0011	Command and Control	Liên lạc với các hệ thống bị xâm nhập để kiểm soát chúng.	16
TA0010	Exfiltration	Lấy cắp dữ liệu, sau đó nén và xóa hoặc mã hóa dữ liệu.	9
TA0040	Impact	Thao túng, làm gián đoạn hoặc phá hủy hệ thống và dữ liệu của nạn nhân.	13

Bảng 2.1: Danh sách 14 chiến thuật của phiên bản MITRE ATT&CK v11.3 năm 2022

ID	$id_{technis}$	Tên kỹ thuật	Chiến thuật
0	T1595	Active Scanning	Reconnaissance
1	T1592	Gather Victim Host Information	Reconnaissance
2	T1589	Gather Victim Identity Information	Reconnaissance
3	T1590	Gather Victim Network Information	Reconnaissance
4	T1591	Gather Victim Org Information	Reconnaissance
5	T1598	Phishing for Information	Reconnaissance
6	T1597	Search Closed Sources	Reconnaissance
7	T1596	Search Open Technical Databases	Reconnaissance
8	T1593	Search Open Websites/Domains	Reconnaissance
9	T1594	Search Victim-Owned Websites	Reconnaissance
10	T1583	Acquire Infrastructure	Resource Development
11	T1586	Compromise Accounts	Resource Development
12	T1584	Compromise Infrastructure	Resource Development
13	T1587	Develop Capabilities	Resource Development
14	T1585	Establish Accounts	Resource Development
15	T1588	Obtain Capabilities	Resource Development
16	T1608	Stage Capabilities	Resource Development
17	T1189	Drive-by Compromise	Initial Access
18	T1190	Exploit Public-Facing Application	Initial Access
19	T1133	External Remote Services	Initial Access, Persistence
20	T1200	Hardware Additions	Initial Access
21	T1566	Phishing	Initial Access
22	T1091	Replication Through Removable Media	Initial Access, Lateral Movement
23	T1195	Supply Chain Compromise	Initial Access
24	T1199	Trusted Relationship	Initial Access
25	T1078	Valid Accounts	Initial Access, Persistence, Privilege Escalation, Defense Evasion
26	T1059	Command and Scripting Interpreter	Execution
27	T1609	Container Administration Command	Execution
28	T1610	Deploy Container	Execution, Defense Evasion
29	T1203	Exploitation for Client Execution	Execution
30	T1559	Inter-Process Communication	Execution
31	T1106	Native API	Execution
32	T1053	Scheduled Task/Job	Execution, Persistence, Privilege Escalation
33	T1129	Shared Modules	Execution
34	T1072	Software Deployment Tools	Execution
35	T1569	System Services	Execution
36	T1204	User Execution	Execution
37	T1047	Windows Management Instrumentation	Execution

Bảng 2.2: Danh sách 1-38/191 kỹ thuật của MITRE ATT&CK v11.3 năm 2022

ID	$id_{technis}$	Tên kỹ thuật	Chiến thuật
38	T1098	Account Manipulation	Persistence
39	T1197	BITS Jobs	Persistence, Defense Evasion
40	T1547	Boot or Logon Autostart Execution	Persistence, Privilege Escalation
41	T1037	Boot or Logon Initialization Scripts	Persistence, Privilege Escalation
42	T1176	Browser Extensions	Persistence
43	T1554	Compromise Client Software Binary	Persistence
44	T1136	Create Account	Persistence
45	T1543	Create or Modify System Process	Persistence, Privilege Escalation
46	T1546	Event Triggered Execution	Persistence, Privilege Escalation
47	T1574	Hijack Execution Flow	Persistence, Privilege Escalation, Defense Evasion
48	T1525	Implant Internal Image	Persistence
49	T1556	Modify Authentication Process	Persistence, Defense Evasion
50	T1137	Office Application Startup	Persistence
51	T1542	Pre-OS Boot	Persistence, Defense Evasion
52	T1505	Server Software Component	Persistence
53	T1205	Traffic Signaling	Persistence, Command and Control
54	T1548	Abuse Elevation Control Mechanism	Privilege Escalation, Defense Evasion
55	T1134	Access Token Manipulation	Privilege Escalation, Defense Evasion
56	T1484	Domain Policy Modification	Privilege Escalation, Defense Evasion
57	T1611	Escape to Host	Privilege Escalation
58	T1068	Exploitation for Privilege Escalation	Privilege Escalation
59	T1055	Process Injection	Privilege Escalation, Defense Evasion
60	T1612	Build Image on Host	Defense Evasion
61	T1622	Debugger Evasion	Defense Evasion, Discovery
62	T1140	Deobfuscate/Decode Files or Information	Defense Evasion
63	T1006	Direct Volume Access	Defense Evasion

Bảng 2.3: Danh sách 39-64/191 kỹ thuật của MITRE ATT&CK v11.3 năm 2022

ID	$id_{technis}$	Tên kỹ thuật	Chiến thuật
64	T1480	Execution Guardrails	Defense Evasion
65	T1211	Exploitation for Defense Evasion	Defense Evasion
66	T1222	File and Directory Permissions Modification	Defense Evasion
67	T1564	Hide Artifacts	Defense Evasion
68	T1562	Impair Defenses	Defense Evasion
69	T1070	Indicator Removal on Host	Defense Evasion
70	T1202	Indirect Command Execution	Defense Evasion
71	T1036	Masquerading	Defense Evasion
72	T1578	Modify Cloud Compute Infrastructure	Defense Evasion
73	T1112	Modify Registry	Defense Evasion
74	T1601	Modify System Image	Defense Evasion
75	T1599	Network Boundary Bridging	Defense Evasion
76	T1027	Obfuscated Files or Information	Defense Evasion
77	T1647	Plist File Modification	Defense Evasion
78	T1620	Reflective Code Loading	Defense Evasion
79	T1207	Rogue Domain Controller	Defense Evasion
80	T1014	Rootkit	Defense Evasion
81	T1553	Subvert Trust Controls	Defense Evasion
82	T1218	System Binary Proxy Execution	Defense Evasion
83	T1216	System Script Proxy Execution	Defense Evasion
84	T1221	Template Injection	Defense Evasion
85	T1127	Trusted Developer Utilities Proxy Execution	Defense Evasion
86	T1535	Unused/Unsupported Cloud Regions	Defense Evasion
87	T1550	Use Alternate Authentication Material	Defense Evasion, Lateral Movement
88	T1497	Virtualization/Sandbox Evasion	Defense Evasion, Discovery
89	T1600	Weaken Encryption	Defense Evasion
90	T1220	XSL Script Processing	Defense Evasion
91	T1557	Adversary-in-the-Middle	Credential Access, Collection
92	T1110	Brute Force	Credential Access
93	T1555	Credentials from Password Stores	Credential Access
94	T1212	Exploitation for Credential Access	Credential Access
95	T1187	Forced Authentication	Credential Access
96	T1606	Forge Web Credentials	Credential Access
97	T1056	Input Capture	Credential Access, Collection
98	T1111	Multi-Factor Authentication Interception	Credential Access
99	T1621	Multi-Factor Authentication Request Generation	Credential Access

Bảng 2.4: Danh sách 65-100/191 kỹ thuật của MITRE ATT&CK v11.3 năm 2022

ID	$id_{technis}$	Tên kỹ thuật	Chiến thuật
100	T1040	Network Sniffing	Credential Access, Discovery
101	T1003	OS Credential Dumping	Credential Access
102	T1528	Steal Application Access Token	Credential Access
103	T1558	Steal or Forge Kerberos Tickets	Credential Access
104	T1539	Steal Web Session Cookie	Credential Access
105	T1552	Unsecured Credentials	Credential Access
106	T1087	Account Discovery	Discovery
107	T1010	Application Window Discovery	Discovery
108	T1217	Browser Bookmark Discovery	Discovery
109	T1580	Cloud Infrastructure Discovery	Discovery
110	T1538	Cloud Service Dashboard	Discovery
111	T1526	Cloud Service Discovery	Discovery
112	T1619	Cloud Storage Object Discovery	Discovery
113	T1613	Container and Resource Discovery	Discovery
114	T1482	Domain Trust Discovery	Discovery
115	T1083	File and Directory Discovery	Discovery
116	T1615	Group Policy Discovery	Discovery
117	T1046	Network Service Discovery	Discovery
118	T1135	Network Share Discovery	Discovery
119	T1201	Password Policy Discovery	Discovery
120	T1120	Peripheral Device Discovery	Discovery
121	T1069	Permission Groups Discovery	Discovery
122	T1057	Process Discovery	Discovery
123	T1012	Query Registry	Discovery
124	T1018	Remote System Discovery	Discovery
125	T1518	Software Discovery	Discovery
126	T1082	System Information Discovery	Discovery
127	T1614	System Location Discovery	Discovery
128	T1016	System Network Configuration Discovery	Discovery
129	T1049	System Network Connections Discovery	Discovery
130	T1033	System Owner/User Discovery	Discovery
131	T1007	System Service Discovery	Discovery
132	T1124	System Time Discovery	Discovery
133	T1210	Exploitation of Remote Services	Lateral Movement
134	T1534	Internal Spearphishing	Lateral Movement
135	T1570	Lateral Tool Transfer	Lateral Movement
136	T1563	Remote Service Session Hijacking	Lateral Movement
137	T1021	Remote Services	Lateral Movement
138	T1080	Taint Shared Content	Lateral Movement

Bảng 2.5: Danh sách 101-139/191 kỹ thuật của MITRE ATT&CK v11.3 năm 2022

ID	$id_{technis}$	Tên kỹ thuật	Chiến thuật
139	T1560	Archive Collected Data	Collection
140	T1123	Audio Capture	Collection
141	T1119	Automated Collection	Collection
142	T1185	Browser Session Hijacking	Collection
143	T1115	Clipboard Data	Collection
144	T1530	Data from Cloud Storage Object	Collection
145	T1602	Data from Configuration Repository	Collection
146	T1213	Data from Information Repositories	Collection
147	T1005	Data from Local System	Collection
148	T1039	Data from Network Shared Drive	Collection
149	T1025	Data from Removable Media	Collection
150	T1074	Data Staged	Collection
151	T1114	Email Collection	Collection
152	T1113	Screen Capture	Collection
153	T1125	Video Capture	Collection
154	T1071	Application Layer Protocol	Command and Control
155	T1092	Communication Through Removable Media	Command and Control
156	T1132	Data Encoding	Command and Control
157	T1001	Data Obfuscation	Command and Control
158	T1568	Dynamic Resolution	Command and Control
159	T1573	Encrypted Channel	Command and Control
160	T1008	Fallback Channels	Command and Control
161	T1105	Ingress Tool Transfer	Command and Control
162	T1104	Multi-Stage Channels	Command and Control
163	T1095	Non-Application Layer Protocol	Command and Control
164	T1571	Non-Standard Port	Command and Control
165	T1572	Protocol Tunneling	Command and Control
166	T1090	Proxy	Command and Control
167	T1219	Remote Access Software	Command and Control
168	T1102	Web Service	Command and Control
169	T1020	Automated Exfiltration	Exfiltration
170	T1030	Data Transfer Size Limits	Exfiltration
171	T1048	Exfiltration Over Alternative Protocol	Exfiltration
172	T1041	Exfiltration Over C2 Channel	Exfiltration
173	T1011	Exfiltration Over Other Network Medium	Exfiltration
174	T1052	Exfiltration Over Physical Medium	Exfiltration
175	T1567	Exfiltration Over Web Service	Exfiltration
176	T1029	Scheduled Transfer	Exfiltration
177	T1537	Transfer Data to Cloud Account	Exfiltration

Bảng 2.6: Danh sách 139-178/191 kỹ thuật của MITRE ATT&CK v11.3 năm 2022

.

.

ID	$id_{technis}$	Tên kỹ thuật	Chiến thuật
178	T1531	Account Access Removal	Impact
179	T1485	Data Destruction	Impact
180	T1486	Data Encrypted for Impact	Impact
181	T1565	Data Manipulation	Impact
182	T1491	Defacement	Impact
183	T1561	Disk Wipe	Impact
184	T1499	Endpoint Denial of Service	Impact
185	T1495	Firmware Corruption	Impact
186	T1490	Inhibit System Recovery	Impact
187	T1498	Network Denial of Service	Impact
188	T1496	Resource Hijacking	Impact
189	T1489	Service Stop	Impact
190	T1529	System Shutdown/Reboot	Impact

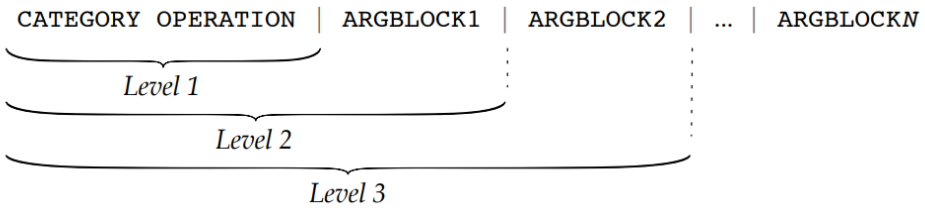
Bảng 2.7: Danh sách 179-191/191 kỹ thuật của MITRE ATT&CK v11.3 năm 2022

Từ bảng 2 đến bảng 7 là danh sách 191 kỹ thuật của MITRE ATT&CK v11.3 năm 2022, trong đó, các kỹ thuật có ID được đánh số từ 0 đến 190, giá trị $id_{technis}$ và Tên kỹ thuật là hai giá trị "ID" và "Name" trên danh sách do tập đoàn MITRE công bố. Cột chiến thuật cho biết kỹ thuật tương ứng thuộc về chiến thuật nào.

2.2 Cấu trúc MIST

MIST có tên đầy đủ là "The Malware Instruction Set" , là một phương pháp biểu diễn mã độc được đề xuất vào năm 2010 với mục tiêu: "tối ưu hóa cho bài toán phân tích hành vi một cách hiệu suất và hiệu quả sử dụng khai phá dữ liệu và các kỹ thuật học máy"[11]. Lấy cảm hứng từ các bộ lệnh được sử dụng trong thiết kế bộ xử lý, mỗi lệnh ở định dạng này sẽ được biểu diễn dưới dạng các số định danh ngắn. Một hành vi của mã độc khi biểu diễn dưới dạng MIST bao gồm hai phần: lệnh gọi hệ thống và đối số.

Hình 2.3 cho thấy cấu trúc cơ bản của một lệnh MIST. Phần lệnh gọi hệ thống bao gồm 2 trường là $CATEGORY$ và $OPERATION$. Trường $CATEGORY$ mã hóa danh mục lệnh gọi hệ thống và trường $OPERATION$ phản ánh một lệnh gọi hệ thống cụ thể. Các đối số được biểu diễn trong trường $ARGBLOCK$.



Hình 2.3: Mô tả sơ đồ của một lệnh MIST [11]

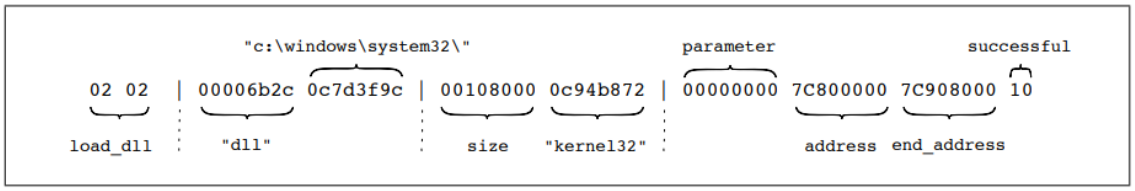
Ví dụ với $CATEGORY$ = "03" VÀ $OPERATION$ = "05", "03" tương ứng với "hệ thống tập tin và "05" tương ứng với lệnh di chuyển tệp, "03 05" tương ứng với lệnh di chuyển tệp tin.

Các cấp độ sau của lệnh chứa các khối đối số khác nhau, trong đó tính cụ thể của các khối tăng từ trái sang phải. Ý tưởng chính cơ bản của việc sắp xếp này là di chuyển các phần tử gây nhiễu ví dụ như tên tệp tạm thời, đến cuối một lệnh còn các phần tử quan trọng như tên thư mục và mutex, được sắp xếp ở đầu.

Hình 2.4 mô tả một ví dụ biểu diễn MIST của một thao tác "load dll". Hình 2.4(a) là hành vi của mã độc biểu diễn dưới dạng XML, đây là kết quả của quá trình theo dõi hành vi của một mã độc trong CWSandbox. Hình 2.4(b) là biểu diễn dạng MIST tương ứng của hình 2.4(a).

```
<load_dll filename="C:\WINDOWS\system32\kernel32.dll" successful="1"
 address="#7C800000" end_address="#7C908000" size="1081344"
 filename_hash="c88d57cc99f75cd928b47b6e444231f26670138f"/>
```

(a) CWSandbox representation



(b) MIST representation

Hình 2.4: Ví dụ biểu diễn MIST của một thao tác "load dll" [11]

"02 02" tương ứng với lệnh gọi "load dll. Để thực hiện chuyển đổi không phân biệt chữ hoa chữ thường, trước tiên, tất cả các giá trị thuộc tính được chuyển đổi thành chữ thường. Những thuộc tính có chứa các ký tự không thuộc biểu biễn thập lục phân sẽ được đưa vào một hàm băm nhanh ví dụ như hàm băm standard ELF. Kết quả được lưu trữ trong một bảng tra cứu và sử dụng trong biểu diễn MIST dưới dạng số thập lục phân.

$ARGBLOCK1$ có giá trị bằng "00006b2c 0c7d3f9c" là kết quả băm của định dạng và đường dẫn tập tin. $ARGBLOCK2$ lưu trữ kích thước tệp và tên tệp của thư viện. $ARGBLOCK$ sẽ tăng dần đối với sự quan trọng giảm dần của các tham số. Với mỗi lệnh gọi hệ thống khác nhau thì cách sắp xếp dữ liệu trong $ARGBLOCK$ cũng sẽ khác nhau.

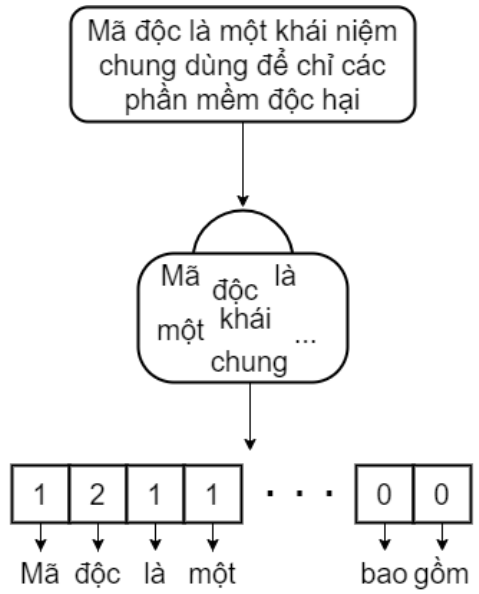
Từ ví dụ trên,biểu diễn dưới dạng MIST cho thấy sự phù hợp với các kỹ thuật khai phá dữ liệu và học máy hơn là biểu diễn XML truyền thống. Việc mã hóa nhỏ

gọn giúp đảm bảo khả năng so sánh thích hợp giữa tất cả các lệnh 'load dll' và thứ tự của tất cả các thuộc tính trong biểu diễn MIST, điều này có thể giúp ích trong nhiệm vụ nâng cao chất lượng phân tích.

2.3 Các kỹ thuật vector hóa

2.3.1 BoW

BoW(Bag-of-words), còn gọi là mô hình túi từ, là một mô hình diễn dữ liệu thường được sử dụng trong các bài toán xử lý ngôn ngữ tự nhiên và truy vấn thông tin. Khi cho một đoạn văn bản, BoW thể hiện văn bản đó dưới dạng một túi(multiset) chứa các từ(words). Multiset này có thể chuyển thành một vectơ có độ dài cố định là số lượng từ để biểu diễn văn bản. Khi biểu diễn một câu, BoW sẽ sử dụng tần suất xuất hiện các chữ trong câu để vector hóa câu đó. Ví dụ về việc áp dụng mô hình Bow đối với một câu được mô tả trong hình dưới dây.

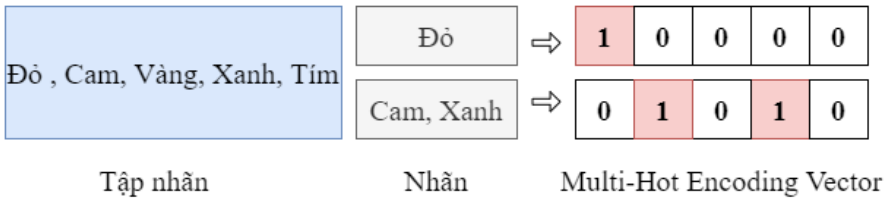


Hình 2.5: Ví dụ Mô hình BoW đối với một câu

2.3.2 Multi-hot Encoding

Multi-hot Encoding là một cách biến đổi dữ liệu từ dạng hạng mục sang dạng số. Kỹ thuật này biến đổi các giá trị thành một chuỗi bit chỉ chứa giá trị "1" hoặc "0". Giá trị "1" được gọi là một giá trị "hot" và giá trị "0" được gọi là một giá trị "cold". Vị trí có giá trị bằng "1" chính là vị trí của nhãn cần biểu diễn trong từ điển. Khác với kỹ thuật One-hot Encoding, trong One-hot Encoding, chỉ có duy nhất một vị trí có giá trị bằng "1", các vị trí còn lại đều bằng "0" thì trong Multi-hot Encoding sẽ có một hoặc nhiều hơn một vị trí có giá trị bằng "1". Multi-hot Encoding là một kỹ thuật phù hợp để biểu diễn nhãn của các đối tượng trong bài toán phân loại đa nhãn. Hình 2.4 là ví dụ về việc áp dụng kỹ thuật Multi-hot Encoding với một tập gồm 5 nhãn màu theo thứ tự là là "Đỏ", "Cam", "Vàng", "Xanh", "Tím". Nếu một

vật thể có một nhãn là nhãn màu đỏ thì biểu diễn nhãn của vật thể đó sẽ là: "1 0 0 0 0 ". Nếu một vật thể có hai nhãn là nhãn màu cam và xanh thì biểu diễn nhãn của vật thể đó sẽ là: " 0 1 0 1 0 ".



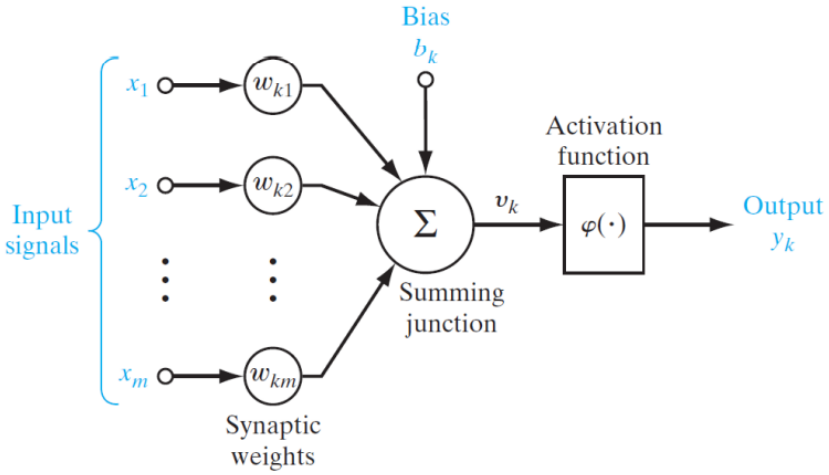
Hình 2.6: Ví dụ về kỹ thuật Multi-hot Encoding với một tập gồm 5 nhãn

2.4 Mạng Multi-Layer Perceptron

Với sự phát triển không ngừng nghỉ, ngày nay, học máy đã đạt được những tiến bộ vượt bậc và có nhiều ứng dụng trong các bài toán thực tế. MLP (Multi-Layer Perceptron) là một trong những kiến trúc học máy đầu tiên được đề xuất.

2.4.1 Kiến trúc cơ bản

Mạng Neural nhân tạo [14] là một mô hình xử lý thông tin được lấy cảm hứng từ cách các hệ thống Neural thần kinh hoạt động, mô phỏng cách não bộ xử lý thông tin. Một mạng Neural bao gồm rất nhiều các Neural đơn lẻ liên kết với nhau. Một nơron như vậy còn được gọi là một perceptron, có cấu trúc tương tự như các Neural sinh học [15].

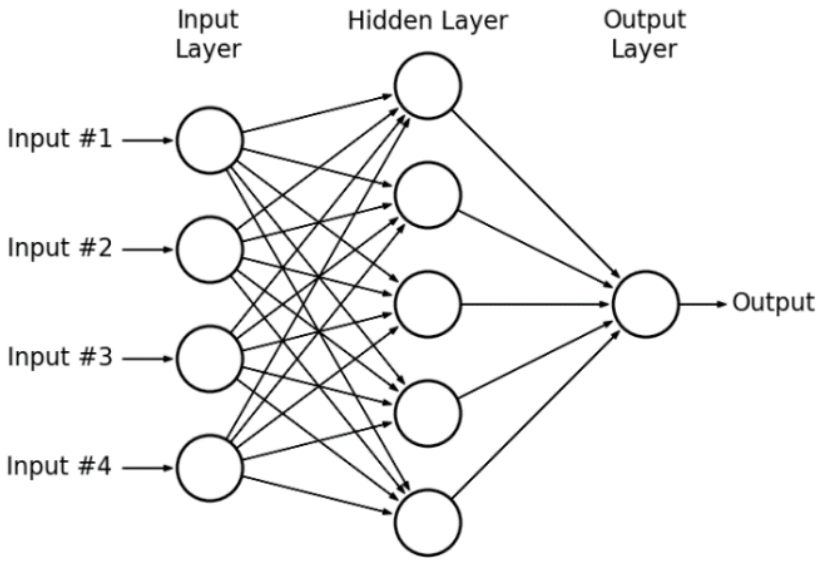


Hình 2.7: Kiến trúc cơ bản của một Perceptron

Hình 2.7 mô tả cấu trúc một Perceptron. Một Perceptron nhận dữ liệu đầu vào dưới dạng các vector, sau khi được đưa qua một phép biến đổi tuyến tính, và qua một hàm kích hoạt phi tuyến ta thu được kết quả đầu ra. Mỗi đầu vào của mạng X_i sẽ được nhân với một trọng số tương ứng W_i sau đó cộng tất cả lại với nhau. Cuối cùng ta đưa kết quả qua một hàm kích hoạt phi tuyến để thu được kết quả đầu

ra của Neural.Các hàm (i) hàm sigmoid, (ii) hàm tanh, (iii) hàm ReLU... là những hàm kích hoạt phổ biến nhất được sử dụng.

Một mạng Neural là tổng hợp của rất nhiều các perceptron liên kết với nhau qua các cạnh, việc lan truyền kết quả giữa các perceptron cũng giống như việc lan truyền tín hiệu điện hóa giữa các nơron thần kinh. Các Neural nhân tạo được sắp xếp thành các tầng liên tiếp nhau. Một mạng Neural nhân tạo thường bao gồm ba thành phần: (i) Tầng đầu vào (input layer), (ii) Các tầng ẩn (hidden layer), (iii) Tầng đầu ra (output layer). Khi đếm số tầng (layer)(thường được gọi là L) của một mạng Neural ta sẽ không tính tầng đầu vào, có nghĩa là số tầng của mạng sẽ bằng số tầng ẩn cộng thêm 1. Kiến trúc cơ bản của môt mạng Neural được mô tả như trong Hình 2.8



Hình 2.8: Kiến trúc cơ bản của mạng Neural nhân tạo

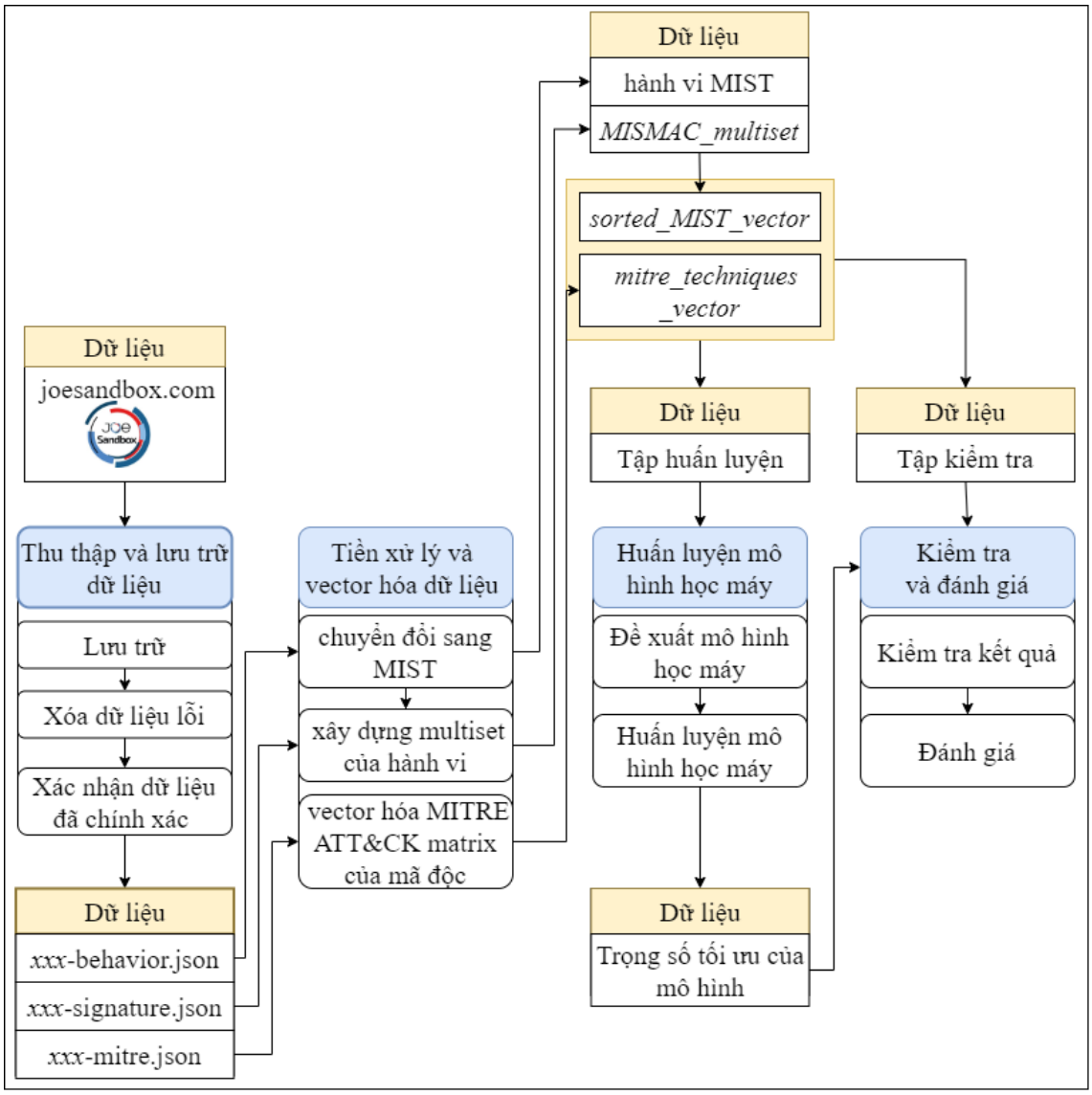
Việc huấn luyện mạng nơ-ron, chính là việc tìm ra bộ tham số w để tối thiểu hóa hàm mất mát, việc này sẽ được thực hiện thông qua quá trình lan truyền ngược (Backpropagation).

CHƯƠNG 3. PHƯƠNG PHÁP ĐỀ XUẤT

Trong chương hai, các kiến thức cơ sở sử dụng trong bài toán phân tích mã độc dựa trên MITRE ATT&CK framework và kỹ thuật học sâu đã được trình bày. Từ những kiến thức đó, chương này sẽ trình bày chi tiết về giải pháp tôi đưa ra để giải quyết bài toán trên. Đây sẽ là cơ sở để thực hiện các đánh giá, cũng như là nền tảng để tiếp tục phát triển các giải pháp khác trong tương lai.

3.1 Mô hình đề xuất tổng quát

Hình 3.1 mô tả tổng quát quy trình hoạt động của giải pháp được đề xuất. Giải pháp bao gồm 4 bước: (i) thu thập - lưu trữ dữ liệu, (ii) tiền xử lý - vector hóa dữ liệu, (iii) huấn luyện mô hình học máy và (iv) kiểm tra - đánh giá kết quả.



Hình 3.1: Tổng quát quy trình hoạt động của giải pháp được đề xuất

Trong bước thu thập và lưu trữ giữ liệu, dữ liệu được thu thập từ trang web Joesandbox.com[16] bằng công cụ chính là thư viện selenium của ngôn ngữ lập trình Python[17]. Trong mỗi báo cáo này, ba nội dung được tôi chú ý đến là hành vi của mã độc, ma trận MITRE ATT&CK và chữ ký. Hành vi của mã độc được lưu vào các file *xxx*-behavior.json, ma trận MITRE ATT&CK được lưu vào *xxx*-mitre.json, chữ ký được lưu vào *xxx*-signature.json. File chữ ký là một file có chứa thông tin kết nối giữa hành vi của mã độc và ma trận MITRE ATT&CK. *xxx* là tên của mã độc, cách lưu trữ tên mã độc đề xuất sẽ được trình bày chi tiết hơn trong mục 3.2.

Trong bước tiền xử lý và vector hóa dữ liệu, dữ liệu về hành vi trong các file *xxx*-behavior.json được chuyển đổi sang dạng MIST bằng một cấu trúc MIST tự thiết kế. Kết quả của công việc này là tập hành vi mã độc dạng MIST, trong đó có chứa các hành vi MIST. Mỗi hành vi MIST là đại diện cho một hành vi của mã độc trong file *xxx*-behavior.json. Bằng kết quả truy vấn trong thông tin trong hai file *xxx*-signature.json và *xxx*-mitre.json, các hành vi MIST sẽ được sắp xếp vào các mục kỹ thuật MITRE ATT&CK tương ứng. Sau khi sắp xếp hành vi MIST vào các mục kỹ thuật MITRE ATT&CK tương ứng, các hành vi trùng lặp sẽ bị lọc ra, kết quả của bước này là một *MISMAC_multiset*. File hành vi MIST sẽ được chuyển sang dạng vector dựa vào tần xuất của từng hành vi MIST trong *MISMAC_multiset*, kết quả thu được là *sorted_mist_vector*. Trong bước vector hóa MITRE ATT&CK của mã độc, sử dụng phương pháp Multi-hot Encoding, mỗi file *xxx*-mitre.json sẽ được chuyển thành một *mitre_techniques_vector* gồm 191 phần tử.

Dữ liệu được chia làm 2 bộ huấn luyện và kiểm tra với tỉ lệ 7:3 và sử dụng trong hai bước là (i)Huấn luyện mô hình học máy và (ii)Kiểm tra và đánh giá. Trong bước huấn luyện mô hình học máy, mô hình học máy được sử dụng là những kiến trúc MLP tự đề xuất với tầng đầu vào là *sorted_MIST_vector* và tầng đầu ra là *mitre_techniques_vector*. Kết quả thu được là trọng số tối ưu của mô hình học máy và kết quả này sẽ được sử dụng trong bước kiểm tra và đánh giá kết quả.

3.2 Thu thập - lưu trữ dữ liệu

3.2.1 Thu thập dữ liệu

Để thu thập dữ liệu từ một trang web, đầu tiên, ta cần biết đường dẫn của trang có chứa thông tin mình cần tìm. Joesandbox.com cung cấp API tìm kiếm đường dẫn có cấu trúc như sau:

"https://www.joesandbox.com/analysis/<*Mã báo cáo*>/0/html?dowload=1"

Trong đó, <*Mã báo cáo*> là một số có 6 chữ số, mỗi chữ số có giá trị từ 0 đến 9. Một báo cáo mã độc có thể có nhiều thông tin như hành vi của mã độc, chữ ký,

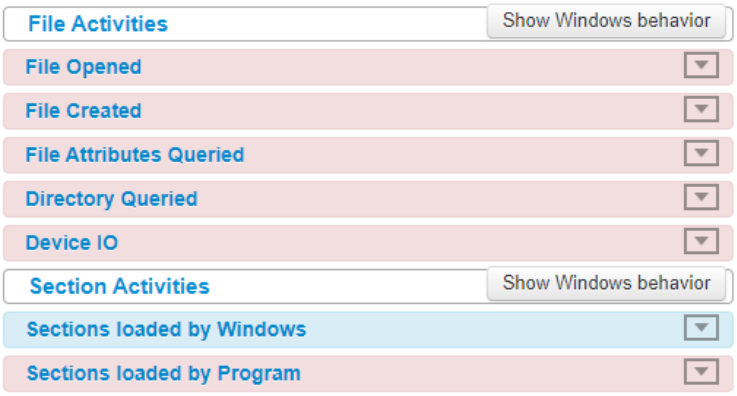
ma trận MITRE ATT&CK, cây thực thi, ảnh chụp màn hình quá trình thực thi mã độc, phân loại mã độc, các địa chỉ IP mã độc kết nối, ...Trong các thông tin trên, ba thông tin được sử dụng trong bài toán này là hành vi của mã độc, chữ ký, ma trận MITRE ATT&CK.

a, Hành vi của mã độc

Tập hành vi mã độc là kết quả của quá trình chạy mã độc trong sandbox. Mỗi sandbox sẽ có một cấu trúc file kết quả khác nhau. Trong Joesandbox.com[16], thông tin về hành vi của mã độc là kết quả của sandbox cùng tên.

Hình 3.2 và 3.3 là hình ảnh của một phần trong mục thông tin hành vi trong Joesandbox. Mục thông tin hành vi trong Joesandbox có thể có nhiều danh mục lệnh gọi hệ thống như: "Files Activities", "Sections Activities", "Registry Activities". Trong các danh mục lệnh gọi hệ thống có các chi tiết lệnh gọi hệ thống như "File Opened", "File Created"," Directory Queried" trong "Files Activities" hay , "Sections loaded by Windows","Sections loaded by Program" trong "Section Activities". Mỗi chi tiết lệnh gọi hệ thống bao gồm thêm các đối số. Ví dụ như: lệnh "File Opened" trong mục "File Activities" có các đối số:

- *File_path* = "C:/Users/user/Desktop/"
- *Access* = "read data or list directory | synchronize"
- *Options* = "directory file | synchronous io non alert | open for backup ident"
- *Content overwritten* = "false"
- *Completion* = "success or wait"
- *Count* = "1"
- *Source Address* = "13F8B6628"
- *Symbol* = "FindFirstFileW"



Hình 3.2: Một phần danh mục lệnh gọi hệ thống trong Joesandbox của báo cáo mã 555555 [18]

File Opened

File Path	Access	Options	Content overwritten	Completion	Count	Source Address	Symbol
C:	read attributes | synchronize	no options	true	success or wait	1	7FEEF9B683B	unknown
C:\Users\user\Desktop\	read data or list directory | synchronize	directory file | synchronous io non alert | open for backup ident	false	success or wait	1	13F8B6628	FindFirstFileW

Hình 3.3: Một phần chi tiết lệnh gọi hệ thống trong Joesandbox của báo cáo mã 555555 [18]

b, Ma trận MITRE ATT&CK

Ma trận MITRE ATT&CK trong Joesandbox,là một phiên bản rút gọn của MITRE ATT&CK Enterprise Matrix. Ma trận MITRE ATT&CK trong Joesandbox loại bỏ một số phần kỹ thuật không bị nghi ngờ để ma trận hiển thị ra có số kỹ thuật trong mỗi chiến thuật là bằng nhau.

Hình 3.4 là hình ảnh ma trận MITRE ATT&CK của báo cáo mã 555555 trong Joesandbox. Trong đó, dòng đầu tiên là các chiến thuật trong MITRE ATT&CK. Trong các dòng còn lại, mỗi ô là một kỹ thuật trong MITRE ATT&CK. Các ô màu trắng là các ô không có hành vi tương ứng với kỹ thuật, các ô màu hồng các các ô có hành vi tương ứng với kỹ thuật. Trong các ô màu hồng có các chỉ số cùng các màu: xanh lục, cam nhạt, cam đậm và đỏ, tương ứng với mức độ nguy hiểm tăng dần là chưa xác định, nghi ngờ, độc hại và đặc biệt độc hại của các hành vi. Chỉ số trong mỗi ô màu thể hiện số lượng hành vi tương ứng với mức độ nguy hiểm đó.

Mitre Att&ck Matrix

Initial Access	Execution	Persistence	Privilege Escalation	Defense Evasion	Credential Access	Discovery	Lateral Movement	Collection	Exfiltration	Command and Control
Valid Accounts	Scripting 4 2	Path Interception	Path Interception	Scripting 4 2	OS Credential Dumping	File and Directory Discovery 1	Remote Services	Data from Local System	Exfiltration Over Other Network Medium	Encrypted Channel 1
Default Accounts	Exploitation for Client Execution 3	Boot or Logon Initialization Scripts	Boot or Logon Initialization Scripts	Obfuscated Files or Information 1	LSASS Memory	System Information Discovery 2	Remote Desktop Protocol	Data from Removable Media	Exfiltration Over Bluetooth	Non-Application Layer Protocol 2
Domain Accounts	At (Linux)	Logon Script (Windows)	Logon Script (Windows)	Obfuscated Files or Information 1	Security Account Manager	Query Registry	SMB/Windows Admin Shares	Data from Network Shared Drive	Automated Exfiltration	Application Layer Protocol 1 1 3
Local Accounts	At (Windows)	Logon Script (Mac)	Logon Script (Mac)	Binary Padding	NTDS	System Network Configuration Discovery	Distributed Component Object Model	Input Capture	Scheduled Transfer	Ingress Tool Transfer 1

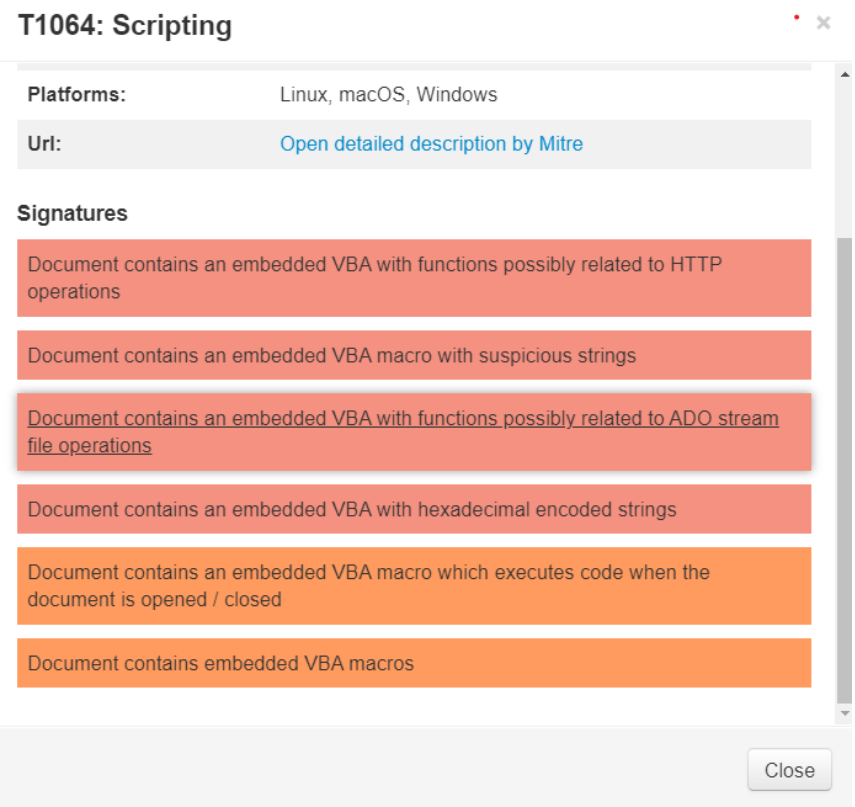
Hình 3.4: Ma trận MITRE ATT&CK của báo cáo mã 555555 trong Joesandbox [18]

Ví dụ như ô kỹ thuật "Application Layer Protocol" của chiến thuật "Command and Control" có ô cam nhạt số 1, một ô cam đậm số 1 và ô xanh lục số 3 thì có nghĩa là trong các hành vi của mã độc, có ba hành vi chưa xác định, một hành vi

nghi ngờ và một hành vi độc hại thuộc về kỹ thuật "Application Layer Protocol" của chiến thuật "Command and Control". Ô kỹ thuật "Scripting" của chiến thuật "Defense Evasion" có ô đỏ số 4 và ô cam nhạt số 2, thì có nghĩa là trong các hành vi của mã độc, có bốn hành vi đặc biệt độc hại và hai hành vi nghi ngờ thuộc về kỹ thuật "Scripting" của chiến thuật "Defense Evasion".

c, Chữ ký

Khi lựa chọn xem chi tiết nội dung các ô có hành vi tương ứng với kỹ thuật (các ô màu hồng). Trang web sẽ trả về kết quả là một hộp thoại như tương tự như hình 3.5. Các ô trong phần "Signatures" chứa các đường link trỏ đến phần hành vi của mã độc tương ứng trong cùng báo cáo.



Hình 3.5: Hộp thoại của Ô kỹ thuật "Scripting" - chiến thuật "Defense Evasion" báo cáo mã 555555 trong Joesandbox [18]

3.2.2 Lưu trữ dữ liệu

Do báo cáo trên trang Joesandbox.com được viết dưới dạng html và javascript, nên để có thể lưu trữ các thông tin cần thiết , thông tin cần được lọc và sử dụng một cấu trúc phù hợp để lưu trữ. Dữ liệu khi được lưu về máy thuộc dạng html, sau đó tôi sử dụng thư viện BeautifulSoup của ngôn ngữ lập trình python để lọc và tách dữ liệu, dữ liệu được lưu dưới dạng file json bằng thư viện ujson của ngôn ngữ lập trình python.

a, Cấu trúc của tên mẫu - *xxx*

xxx là biến của tên mã độc. Trong đồ án này, có cấu trúc :

mal<*phần trăm là mã độc*>.<*một hoặc nhiều phân loại mã độc*>.
<*hệ điều hành*><*định dạng file*>-<*tên mã độc dạng hash SHA256*>

Ví dụ với hai tên:

(1) mal52.evad.winEXE-f5b53de3ec96bf609794143acffb7960a90ecff7b4085e2b09230f9a007cbbc4

(2) mal100.troj.expl.evad.winXLS-b81dede24878d2f5b336b366da8dc830651982b1999409a0fbbcbf92ee95bcbd

Tên (1) có nghĩa là mã độc có 52% khả năng là malware, thuộc loại Evader, là mã độc trên Win, có định dạng file là EXE(file thực thi), tên dạng SHA256 là "f5b53de3ec96bf609794143acffb7960a90ecff7b4085e2b09230f9a007cbbc4"

Tên (2) có nghĩa là mã độc có 100% khả năng là malware, thuộc ba loại là Trojan, Exploiter và Evader, là mã độc trên Win, có định dạng file là XLS (file Excel), tên dạng SHA256 là "b81dede24878d2f5b336b366da8dc830651982b1999409a0fbbcbf92ee95bcbd"

b, Cấu trúc lưu trữ các file *xxx*-behavior.json

Định dạng của file *xxx*-behavior.json có dạng như sau:

{ { <*ID của chương trình chạy hành vi*> :{
<*Tên tập hợp hành vi 1*> : {
"title": <*Tên*>
"activies": {
<*Tên hành vi 1*>: {
"items": <*Các dấu hiệu của hành vi*>,
"table":{ [<*Giá trị thứ nhất*>,..., <*Giá trị thứ N*>], ... }
<*Tên hành vi 2*>:{ ... } }
<*Tên tập hợp hành vi 2*>: ...
...
<*Tên tập hợp hành vi* $N_{behaviors}$>: ...
}
} }

<ID của chương trình chạy hành vi> là id gắn với chương trình thực thi các hành vi. "Tên tập hợp hành" là tên bao gồm browser, com, crypto, ..., tất cả hành vi được liệt kê đầy đủ ở bảng 3.1. Bên trong đó title là tên tương ứng với tập đó, activites bao gồm các hành, bên trong đó có "items" liệt kê các hành vi và "table" là tên của các hành vi tương ứng.

c, Cấu trúc lưu trữ các file *xxx*-mitre.json

Cấu trúc lưu trữ của một file *xxx*-mitre.json có dạng như sau:

{

{ *<Tên kỹ thuật/chi tiết kỹ thuật$_1$>* :

{

"tactics_type" : *<Tên chiến thuật>*,

"signature_len" : SL_1,

"signatures": [*<Dấu hiệu thứ 1>*, ... *<Dấu hiệu thứ SL_1>*]

}

},

. . . ,

{ *<Tên kỹ thuật/chi tiết kỹ thuật$_{Nmitre}$>* :

{

"tactics_type" : *<Tên chiến thuật>*,

"signature_len" : SL_{Nmitre},

"signatures": [*<Dấu hiệu thứ 1>*, ... *<Dấu hiệu thứ SL_{Nmitre}>*]

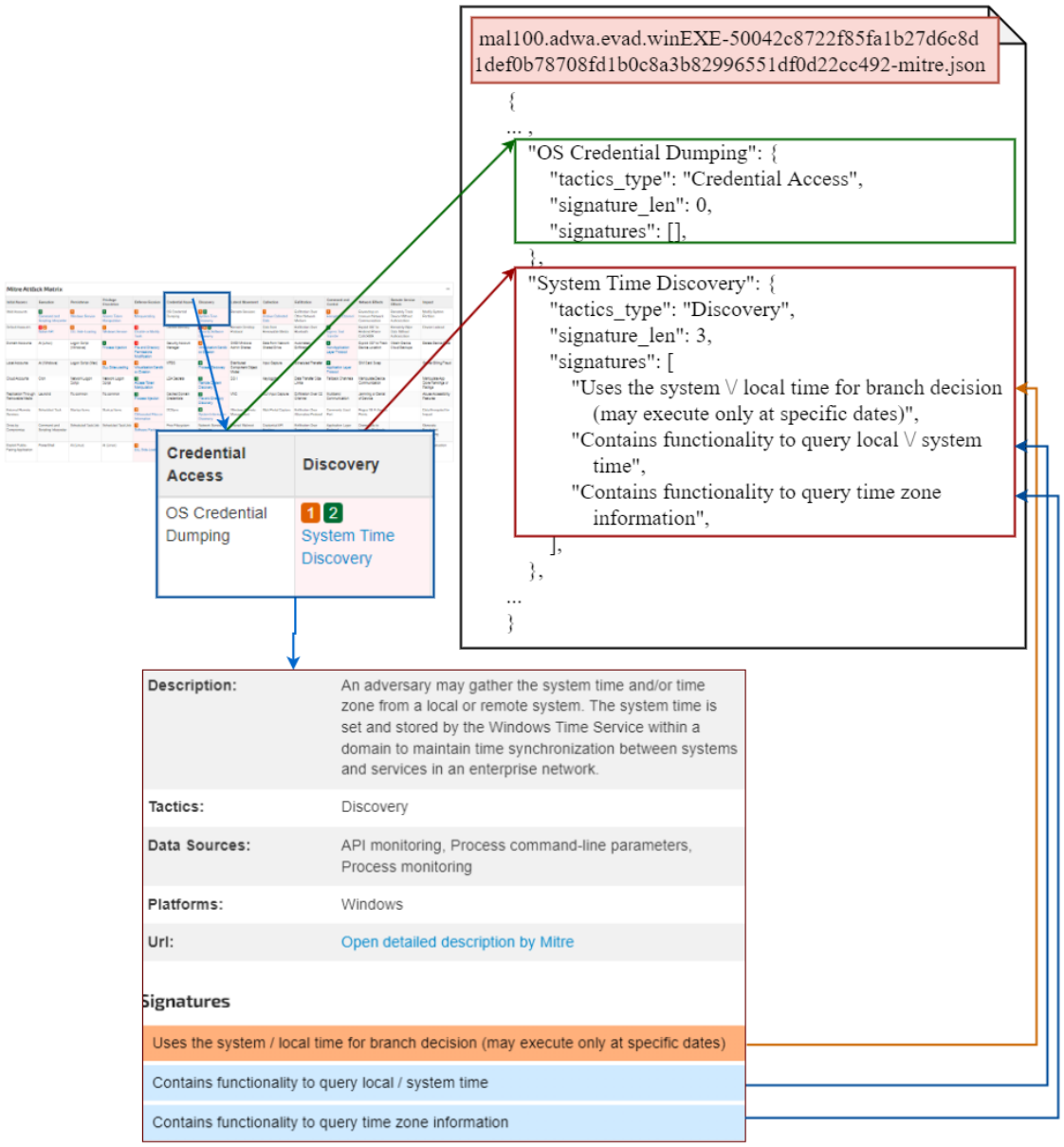
}

}

}

Trong đó: <Tên kỹ thuật/chi tiết kỹ thuật> là một trong 191 tên kỹ thuật hoặc một trong 385 chi tiết kỹ thuật của MITRE ATT&CK. <Tên chiến thuật> là một trong 14 chiến thuật của MITRE ATT&CK. N_{mitre} là số kỹ thuật trong ma trận MITRE ATT&CK của mã độc đó hiển thị. $SL_{N_{mitre}}$ là số lượng chữ ký kết nối với hành vi độc hại tương ứng của kỹ thuật đó ($SL_{N_{mitre}} \geq 0$). Trong trường hợp $SL_{N_{mitre}}$ = 0, kỹ thuật thứ N_{mitre} không có hành vi độc hại tương ứng. Hình 3.6 mô tả cách một ô kỹ thuật trong MITRE ATT&CK của một file mã độc

chuyển thành một đoạn thông tin dưới dạng jsoN, lấy thông tin của mẫu mã độc "50042c8722f85fa1b27d6c8d1def0b78708fd1b0c8a3b82996551df0d22cc492" làm ví dụ.



Hình 3.6: Ví dụ về cách lưu trữ MITRE ATT&CK matrix

d, Cấu trúc lưu trữ các file *xxx*-signature.json

Cấu trúc của một file lưu trữ *xxx*-signature.json có dạng như sau:

{

{ <*Tên chữ ký 1*> :

{

"signature_header" : <*Tên hiển thị*>,

"behaviour_id" : *ID hành vi*,

"pid_md5": *id và md5 của tiến trình*

"data_section": *Tên hành vi*

"data_activity": *Tên tập hành vi*

}

},

. . . ,

{ <*Tên chữ ký* $Signature_N$> :

{

"signature_header" : <*Tên hành vi*>,

"behavior_id" : *ID hành vi*,

"pid_md5": *id và md5 của tiến trình*

"data_section": *Tên hành vi*

"data_activity": *Tên tập hành vi*

}

}

}

"signature_header" là dấu hiệu này thuộc vào tập dấu hiệu mẹ nào, "behavior_id" là ID để định danh dấu hiệu, giúp biết hành vi nào trong file *xxx*-behavior.json nối với dấu hiệu này, "pid_md5" là md5 của ID tiến trình đang chạy sinh ra tập hành vi này, data_section là tên hành vi tương ứng với dấu hiệu này, data_activity là tập hành vi mà hành vi của dấu hiệu này thuộc về.

3.2.3 Xóa dữ liệu lỗi và Xác nhận dữ liệu đã chính xác

Trong quá trình lưu trữ, một số lỗi có thể xảy ra như lỗi đường truyền, lỗi bộ nhớ, file bị xóa bởi các trình diệt virus. Điều đó khiến một số mẫu dữ liệu bị hỏng và gây lỗi trong các phần xử lý dữ liệu tiếp theo. Do đó, tôi thực hiện công việc xóa dữ liệu lỗi để loại bỏ các mẫu dữ liệu lưu trữ gặp sai sót.

Như đã đề xuất ở phần lưu trữ dữ liệu, dữ liệu lưu trữ của một mẫu mã độc bao gồm 3 file là *xxx*-behavior.json, *xxx*-mitre.json và *xxx*-signature.json. Nếu như quá trình lưu trữ gặp sai sót thì mẫu mã độc có thể mất từ một đến hai file trong số ba file thông tin. Các mẫu dữ liệu như vậy cần được tìm thấy và xóa các file còn lại khỏi bộ dữ liệu.

Sau khi đã thực hiện bước xóa các dữ liệu lỗi, dữ liệu được kiểm tra lại một lần nữa bằng cách kiểm tra xem, mỗi mẫu mã độc có đủ 3 file không. Nếu như không có mẫu mã độc nào bị thiếu file thì bước Xóa dữ liệu lỗi và Xác nhận dữ liệu đã chính xác được coi là hoàn thành.

3.3 Tiền xử lý - vector hóa dữ liệu

3.3.1 Thiết kế kiến trúc MIST

Do kiến trúc MIST trong bài báo gốc [11] được đề xuất từ năm 2010 và là kiến trúc được thiết kế cho kết quả đầu ra của CWSandbox, nhiều thành phần trong cấu trúc MIST được đề xuất trong bài báo gốc đã không còn tương thích với các kết quả đầu ra của các sandbox hiện tại. Do đó, trong đồ án này, tôi thiết kế một cấu trúc MIST mới phù hợp với bộ dữ liệu đã thu thập. Thiết kế này vẫn dựa trên tư tưởng xây dựng của cấu trúc MIST trong bài báo gốc.

Để có thể biểu diễn các thông tin về đoạn ký tự thì chúng ta sử dụng hàm băm để đưa về giá trị số học. Nếu như hai đoạn văn bản giống nhau thì sẽ có cùng một giá trị. Trong bài báo gốc, tác giả sử dụng hàm băm ELFHash để có thể biểu diễn các đoạn ký tự thành số tương ứng. Nhưng việc sử dụng ELFHash không tối ưu và độ dài kết quả hàm băm chỉ có độ dài là 32bit. Thế nên đề tài sử dụng thuật toán xxhash3 thay thế cho ELFHash. Ưu điểm của xxhash so với ELFHash bao gồm:

1. xxhash3 cho ra kết quả có độ dài là 64bit lớn hơn so với kết quả thuật toán ELFHash chỉ có độ dài 32bit.
2. xxhash3 sử dụng phần cứng để tối ưu các hoạt động tính toán nên kết quả nhanh hơn và sử lý được nhiều dữ liệu hơn so với ELFHash.

Đối với các thông tin về đường dẫn thì đầu tiên ta tách ra thành từng phần bao gồm:

1. thư mục mẹ
2. tên file
3. đuôi file
4. toàn bộ đường dẫn

Ví dụ với toàn bộ đường dẫn: `D:\Daihocbk\crawldata\malware1.exe` thì thư mục mẹ là `D:\Daihocbk\crawldata\\`, tên file là malware1, đuôi file là exe.

$$\underbrace{\text{D:\Daihocbk\crawldata\}}_{\text{thư mục mẹ}}\underbrace{\text{malware1}}_{\text{tên file}}.\underbrace{\text{exe}}_{\text{đuôi file}}$$

sau đó sử dụng hàm băm xxhash3 cho từng phần trong đường dẫn đó. Chúng ta chia thế này do mã độc thông thường sử dụng tên file khác nhau, nhưng sẽ lưu vào các nơi thông dụng như %TEMP% hoặc là sử dụng kỹ thuật để duy trì trên máy thông qua registry run. Việc biết được thư mục mẹ mà mã độc thực thi hành vi giúp cho có thể thấy được tương đồng giữa các hành vi của cùng một phần trong mitre.

Đối với các thông tin về giá trị như khả năng đọc, viết, thực thi của file hoặc tương tự thì đề tài đưa về hệ nhị phân để biểu diễn từng khả năng, ví dụ khả năng đoc thôi là 1 khả năng đọc và viết là 11, khả năng đọc, viết và thực thì là 111, khả năng đọc và thực thi là 101. Còn lại với biểu diễn số thì đưa về giá trị tương ứng của nó.

Bảng sau đây là các category và operation tương ứng với số của nó:

Category	Category Name	Operation Name	Operation
1	Hành vi file	DeviceIO	1
		DirectoryQueried	2
		FileAttributesQueried	3
		FileCopied	4
		FileCreated	5
		FileDeleted	6
		FileMoved	7
		FileOpened	8
		FileOtherOp	9
		FileRead	10
		FileWritten	11
		VolumeInformationQueried	12
		DriveTypeQueried	13
		FolderQueried	14

2	Hành vi registry	KeyCreated	1
		KeyDeleted	2
		KeyEnumerated	3
		KeyMonitored	4
		KeyOpened	5
		KeyValueCreated	6
		KeyValueDeleted	7
		KeyValueEnumerated	8
		KeyValueModified	9
		KeyValueQueried	10
3	Hành vi process	ProcessCreated	1
		ProcessQueried	2
		ProcessSet	3
		ProcessSuspended	4
		ProcessTerminated	5
		ShellExecuted	6
4	Hành vi mutex	MutexCreated	1
5	Hành vi memory	MemAccess	1
		MemAlloc	2
		MemProtect	3
		MemRead	4
		MemWritten	5
		MemStats	6
6	Hành vi token	AdjustToken	1
7	Hành vi thread	ThreadAPCQueued	1
		ThreadCreated	2
		ThreadDelayed	3
		ThreadGot	4
		ThreadInformationSet	5
		ThreadResumed	6
		ThreadSet	7
		ThreadSuspended	8
		ThreadTerminated	9
8	Hành vi browser	Cdocumentwrite	1
		InetConnect	2
		InetOpenRequest	3
		InetReadFile	4

		InetWriteFile	5
		Jsscriptcompile	6
9	Powershell	Eventlog	1
10	Hành vi crypto	CertificateCreated	1
		DataDecrypted	2
		DataEncrypted	3
		GenKey	4
		KeyImported	5
		KeyExported	6
11	Hành vi debug	DebugSet	1
12	Hành vi driver	DeviceCreated	1
		DriverUnloaded	2
		DriverLoaded	3
13	Hành vi user	DesktopCreated	1
		ForegroundWindowGot	2
		InputBlocked	3
		InputSent	4
		KeyStateQueried	5
		WindowDestroyed	6
		MessagePosted	7
		MessageThreadPosted	8
		WindowCreated	9
		WindowEnumerated	10
		WindowFound	11
		WindowPlacementGot	12
		WindowPlacementSet	13
		WindowShown	14
		WindowsHookSet	15
14	Hành vi system	CommandLineQueried	1
		ComputerNameQueried	2
		CpuIDQueried	3
		HardErrorRaised	4
		KeyboardLayoutListQueried	5
		KeyboardLayoutQueried	6
		LanguageOrLocaleQueried	7
		ServiceCreated	8
		ServiceDeleted	9

		ServiceOpened	10
		ServiceStarted	11
		SystemParameter	12
		SystemPowerStateSet	13
		SystemQueried	14
		SystemSet	15
		SystemShutdown	16
		UserNameQueried	17
		VersionQueried	18
15	Hành vi timing	PerformanceQueried	1
		TicksQueried	2
		TimeQueried	3
		TimerSet	4
		UserTimerSet	5
16	Hành vi sysmon	Eventlog	1
17	Hành vi string	StringCompared	1
18	Hành vi security	ObjectSet	1
19	Hành vi section	ModuleHandleQueried	1
		ProcAddressQueried	2
		ResourceQueried	3
		SectionLoadedW	4
		SectionLoadedP	5
20	Hành vi port	ConsoleWritten	1
		PortRequestWaitReplay	2
21	Hành vi network	Adapterinfoquery	1
		Addrinfoquery	2
		Httpopenrequest	3
		Internetconnect	4
		Networkconnect	5
		Networkbind	6
		Networkresourceconnected	7
		Tcporudptablequeried	8
22	Hành vi java	Javaactivities	1

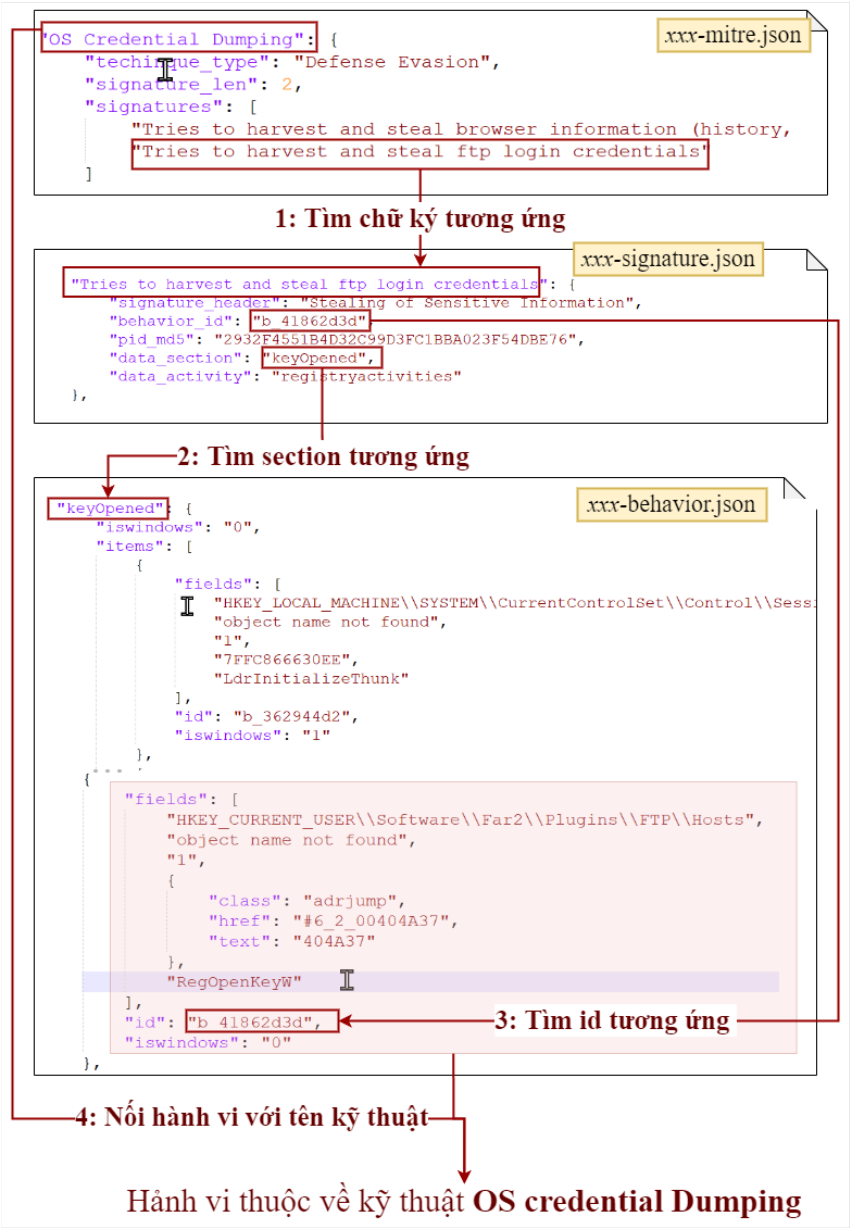
Bảng 3.1: Danh sách mã hóa Category và Operation của thiết kế MIST

3.3.2 Xây dựng multiset của hành vi sử dụng MISMAC

Ý tưởng của MISMAC là sắp xếp các hành vi MIST thành các nhóm thuộc về cùng kỹ thuật của MITRE ATT&CK. MISMAC bao gồm hai bước:(i) Tìm tên kỹ thuật của hành vi và (ii) Xây dựng *MISMAC_multiset*.

a, Tìm tên kỹ thuật MITRE ATT&CK của hành vi

Ba file là*xxx*-mitre.json *xxx*-signature.json và *xxx*-behavior.json sẽ được dùng để tham chiếu dữ liệu. Bắt đầu từ file *xxx*-mitre.json, mỗi signature sẽ tương ứng với một hành vi, ta lấy signature và tìm trong file *xxx*-signature.json tương ứng. Trong file *xxx*-signature.json sẽ có hai thông tin là "behavior_id" và "data_section", hai thông tin đó sẽ được tìm trong file *xxx*-behavior.json tương ứng ở vị trí <*tên hành vi*> là giá trị của "data_section" và "id" bằng với "behavior_id".

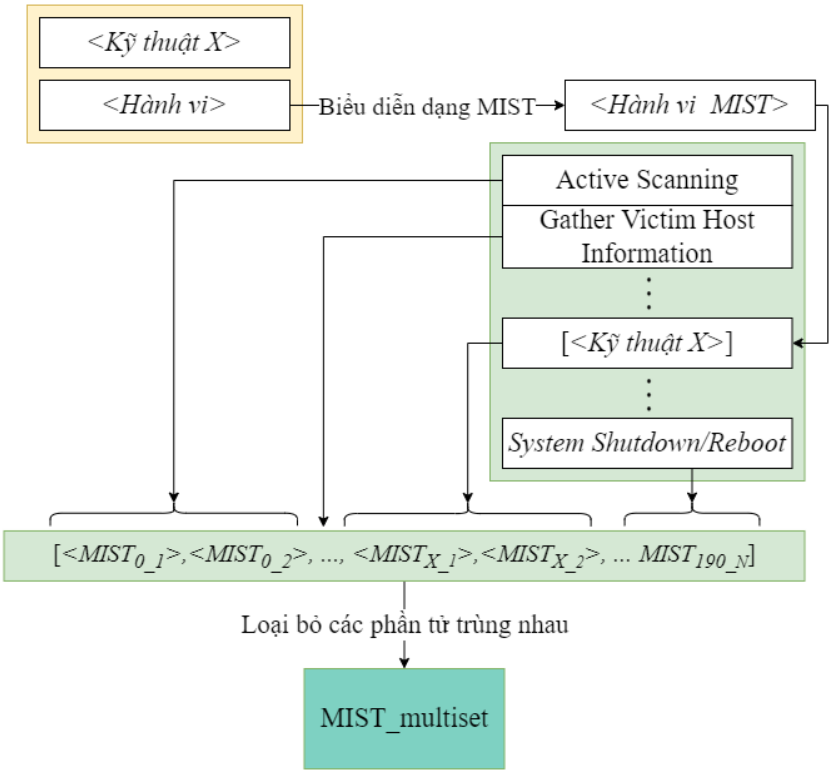


Hình 3.7: Ví dụ cách tham chiếu tên kỹ thuật MITRE ATT&CK đến hành vi tương ứng

Hình 3.7 lấy một phần thông tin của mẫu mã độc "4e7dbf896967ae2b896187625e c0e112e26a797c6f5a42e44fb55cf633cdfea0" làm ví dụ để mô tả cách tham chiếu kỹ thuật đến các hành vi tương ứng. Đầu tiên ở file *xxx*-mitre.json, kỹ thuật "OS Credential Dumping" có hai signature, có nghĩa là có hai hành vi tương ứng. Xét hành vi "Tries to harvest and steal ftp login credentials", tìm kiếm "Tries to harvest and steal ftp login credentials", ta thấy một chữ kí có "behavior_id" là "b_41862d3d" và "data_sections" là "keyOpened". Tìm hai thông tin này trong file *xxx*-behavior.json, ta sẽ tìm được hành vi tương ứng ứng vớ chữ kỹ "Tries to harvest and steal ftp login credentials" trong kỹ thuật "OS Credential Dumping".

b, Xây dựng *MISMAC_multiset*

Sau khi đã có được hành vi và tên kỹ thuật của hành vi đó, ta có thể chia hành vi về các nhóm có kỹ thuật giống nhau để xây dựng *MISMAC_multiset*. Hình 3.8 mô tả cách xây dựng một *MISMAC_multiset* sau khi đã tìm được kỹ thuật tương ứng của tất cả các hành vi. Đầu tiên, hành vi được biểu diễn ở dạng MIST để trở nên ngắn gọn và dễ dàng trong việc lưu trữ hơn. Hành vi MIST sẽ được xếp vào nhóm hành vi có cùng kỹ thuật trong cùng danh sách. Sau đó, danh sách chứa các hành vi được chuyển thành một mảng một chiều với mỗi phần tử là một hành vi MIST. Lúc này, mảng một chiều chứa các hành vi MIST vẫn còn các phần tử trùng nhau. Các phần tử đó sẽ bị loại bỏ để mảng một chiều sẽ trở thành một *MISMAC_multiset* có ý nghĩa tương tự như "Bag" trong phương pháp BoW.



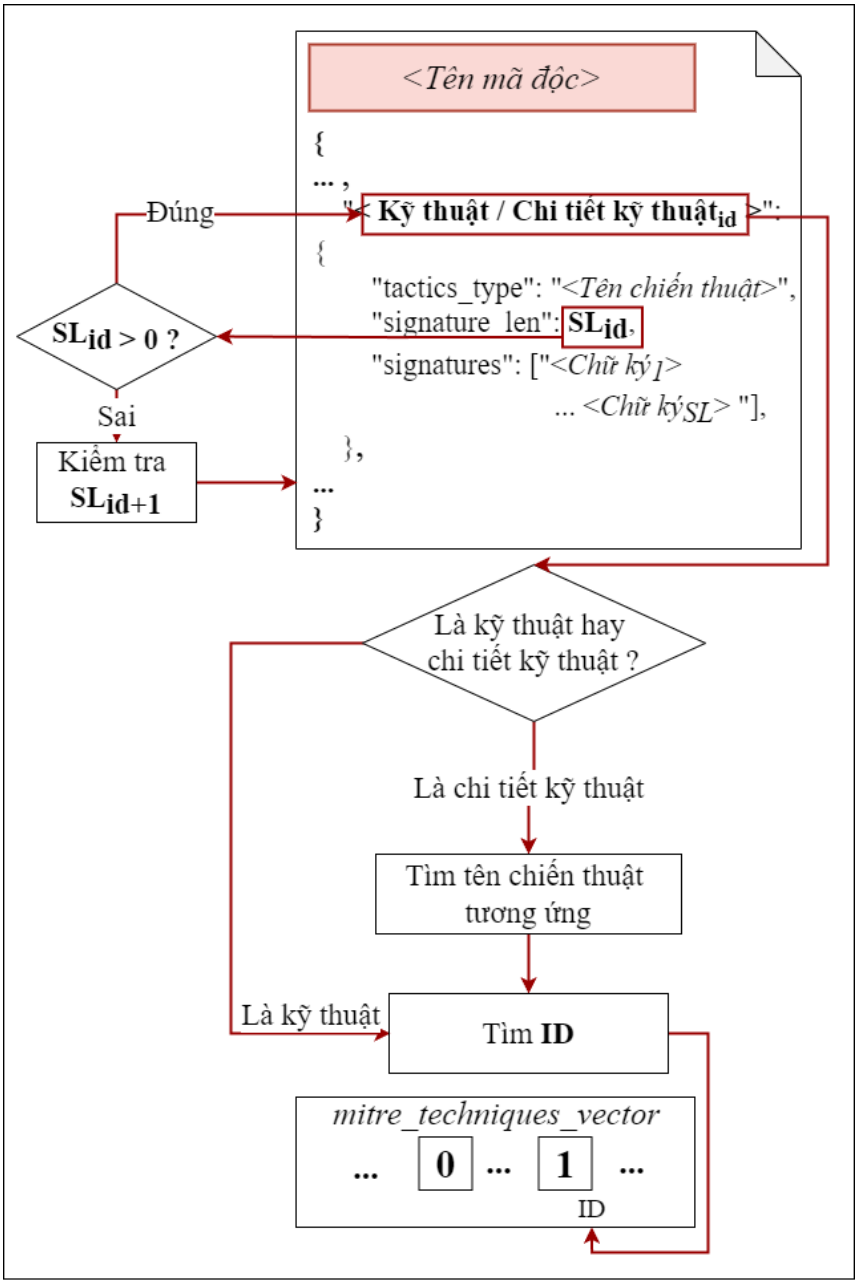
Hình 3.8: Sơ đồ cách xây dựng *MISMAC_multiset*

Theo như thiết kế kiến trúc MIST, một hành vi MIST có thể có nhiều hơn một đối số, việc chọn số đối số của hành vi MIST để sử dụng trong BoW là rất quan trọng vì nó liên quan đến kích thước của tập dữ liệu. Với sự đa dạng của hành vi mã độc, nếu như việc chọn số đối số không phù hợp sẽ dẫn đến việc *sorted_MIST_vector* trở lên quá lớn với phần cứng, khiến quá trình huấn luyện mô hình học máy bị gián đoạn.Trong quá trình biểu diễn hành vi sang dạng MIST của hình 3.8, vì các đối số càng ở sau thì sẽ càng ít quan trọng hơn nên số lượng đối số được đề xuất sử dụng của MIST là từ 1 đến 3, tương ứng với Level 2, Level 3 và Level 4 trong hình 2.3. Số lượng đối số được chọn không nên bằng 0 vì khi đó, sẽ có quá ít thông tin được lưu lại cho quá trình học máy. Với mỗi số lượng đối số được chọn ta sẽ xây dựng được một *MISMAC_multiset* khác nhau.

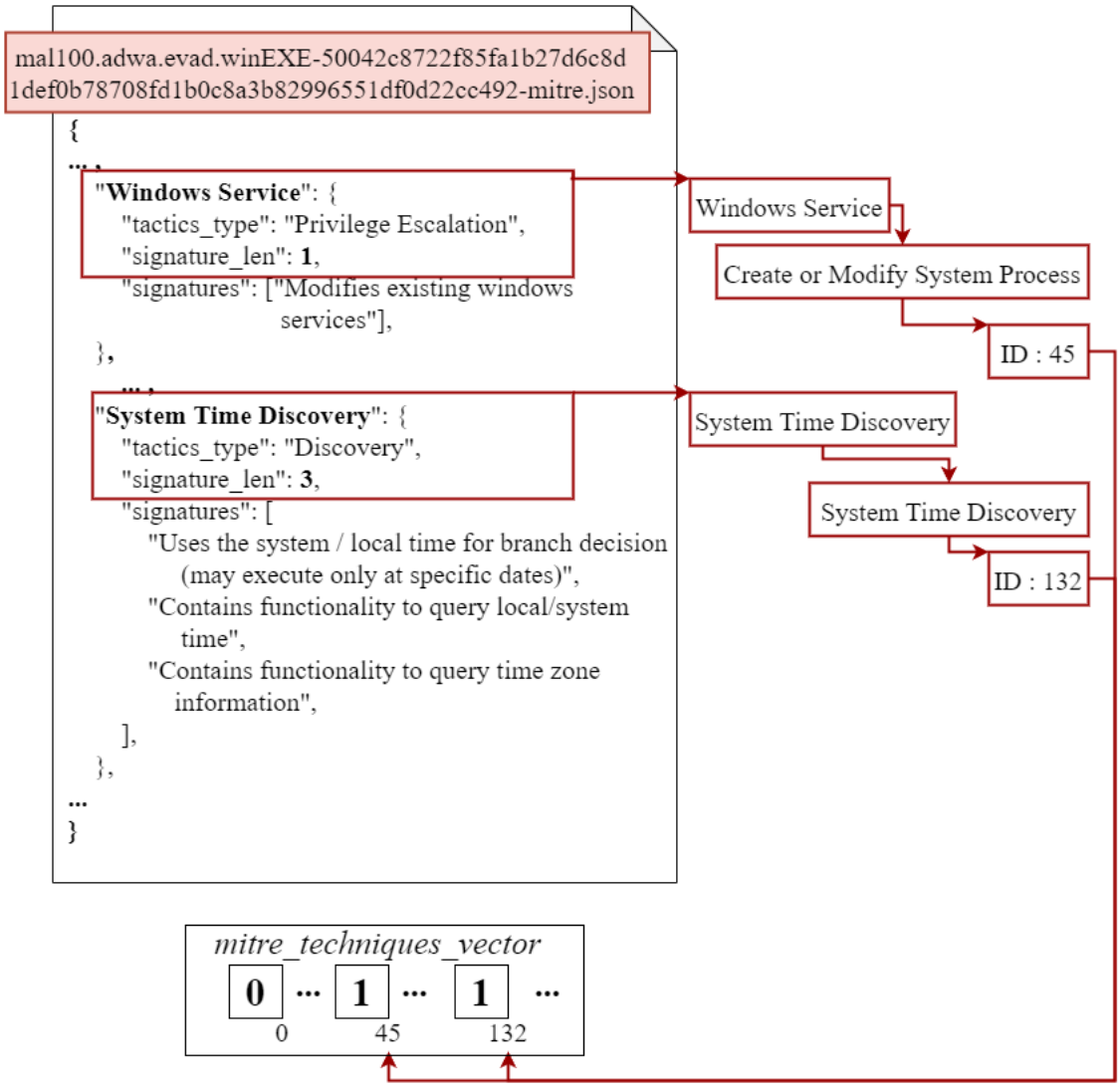
3.3.3 Vector hóa MITRE ATT&CK matrix của mã độc

Dựa vào cấu trúc lưu trữ các file xxx-mitre.json, các thông tin có thể trích xuất ra là tên kỹ thuật/chi tiết kỹ thuật, số lượng chữ ký của kỹ thuật/chi tiết kỹ thuật đó. MITRE ATTCK có 191 kỹ thuật và 385 chi tiết kỹ thuật, nếu như sử dụng tất cả kỹ thuật và chi tiết kỹ thuật thì tổng số lượng nhãn của bài toán sẽ là 576 nhãn. Tuy vậy, nếu như sử dụng cả 576 nhãn thì mô hình có thể gặp vấn đề dư thừa về nhãn và sẽ có nhiều nhãn có nội dung gần như tương tự nhau, do đó tôi quyết định chọn 191 kỹ thuật làm nhãn đại diện cho bài toán. Các nhãn chi tiết kỹ thuật sẽ được chuyển thành các nhãn kỹ thuật tương ứng. Tổng số nhãn đầu ra của bài toán theo đề xuất này là 191 nhãn. 191 nhãn này sẽ được lưu trong *mitre_techniques_vector* với 191 phần tử. Nếu nhãn số id có số lượng hành vi độc hại tương ứng là SL_{id} với $SL_{id} \geq 1$ thì $mitre\_techniques\_vector_{id}$ sẽ có giá trị bằng "1", có nghĩa là mã độc có sử dụng kỹ thuật có ID là id. Trong trường hợp ngược lại, Nếu SL_{id} với $SL_{id} = 0$ thì $mitre\_techniques\_vector_{id}$ sẽ có giá trị bằng "0", có nghĩa là mã độc không sử dụng kỹ thuật có ID là id. Giá trị id của mã độc có thể được tra cứu trong các bảng: bảng 2, bảng 3, bảng 4, bảng 5, bảng 6, bảng 7 trong chương hai. Hình 3.9 là sơ đồ hoạt động của thuật toán chuyển xxx-mitre.json thành *mitre_techniques_vector*.

Hình 3.10 lấy ví dụ nội dung file xxx-mitre.json của mẫu mã độc "50042c8722f8 5fa1b27d6c8d 1def0b78708fd1b0c8a3b82996551df0d22cc492" để mô tả cách mà thông tin trong file xxx-mitre.json chuyển thành một *mitre_techniques_vector*. "Windows Service" là một chi tiết kỹ thuật của kỹ thuật "Create or Modify System Process" có ID bằng 45. Với mã độc được xét, signature_len của chi tiết kỹ thuật "Windows Service" có giá trị bằng 1, do đó, $mitre\_techniques\_vector_{45} = 1$. "System Time Discovery" là một kỹ thuật có ID bằng 132. signature_len của kỹ thuật "System Time Discovery" có giá trị bằng 3. Do đó, $mitre\_techniques\_vector_{132} = 1$.



Hình 3.9: Sơ đồ cách chuyển nội dung file xxx-mitre.json thành *mitre_techniques_vector*



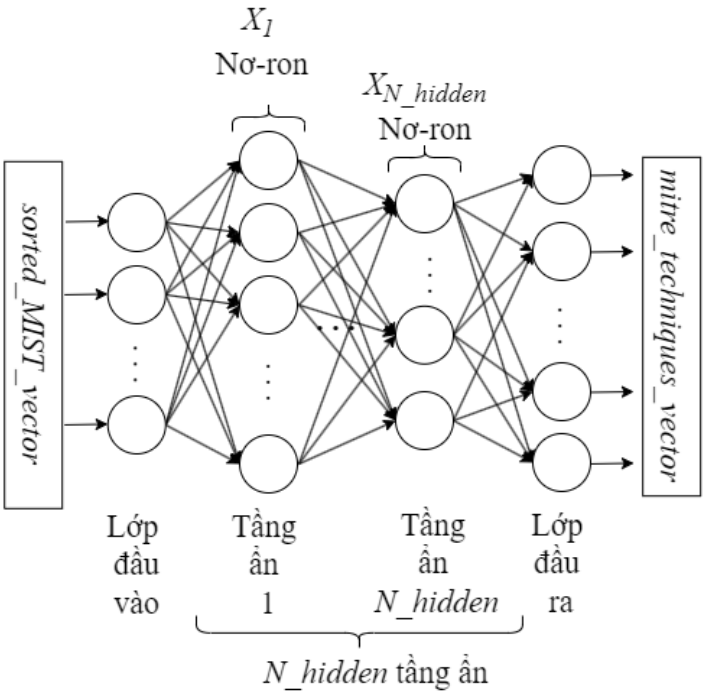
Hình 3.10: Ví dụ về cách tạo *mitre_techniques_vector*

3.4 Huấn luyện mô hình học máy

Một mô hình học máy phù hợp sẽ giúp bài toán có một kết quả tốt. Nếu như mô hình học máy quá lớn hay quá nhỏ so với bộ dữ liệu thì có thể dẫn tới việc mô hình học không hiệu quả và cho kết quả không chính xác.

3.4.1 Đề xuất mô hình học máy

Dựa vào kết quả của bước hai, mô hình học máy sẽ có lớp đầu vào với số nơ-ron bằng với số phần tử trong *MISMAC_multiset* và lớp đầu ra có số nơ-ron bằng với số phần tử của *mitre_techniques_vector*. Hình 3.11 là kiến trúc tổng quan của mô hình MLP với kết quả của bước hai.



Hình 3.11: Kiến trúc cơ bản của mạng Neural nhân tạo đề xuất

Với một mô hình MLP thì các tham số ảnh hưởng đến mô hình gồm có:

1. Số tầng ẩn : *N_hidden* $(1 \leq N\_hidden \leq 3)$
2. Số nơ-ron của mỗi tầng ẩn : X_{N_hidden}($X_{N\_hidden} \in [64, 128, 256, 512, 1024]$)

Để có thể tìm ra một mô hình phù hợp với bộ dữ liệu,các tham số sẽ được thử nghiệm để tìm ra tham số phù hợp nhất.

3.4.2 Huấn luyện mô hình học máy

Với các siêu tham số của mô hình huấn luyện, sau khi thử nghiệm với nhiều bộ siêu tham số khác nhau, bộ tham số cho ra kết quả tốt nhất được trình bày trong chương bốn. Trong quá trình huấn luyện, hàm tối ưu được sử dụng là Adam[19], kích thước lô là 64, số vòng lặp là 200. Để có thể chọn ra mô hình tốt nhất,tập huấn

luyện được chia làm 2 phần là train và validation với tỉ lệ 7:3. 30% mẫu được chọn ra từ tập huấn luyện để sử dụng làm tập đánh giá (Validation set). Trong quá trình huấn luyện, các trọng số của mô hình sẽ được lưu lại khi kết thúc một vòng lặp, bộ tham số với kết quả cao nhất trên tập đánh giá sẽ được sử dụng là kết quả huấn luyện cuối cùng để sử dụng trong bước đánh giá kết quả, điều này giúp mô hình tránh việc học quá khớp.

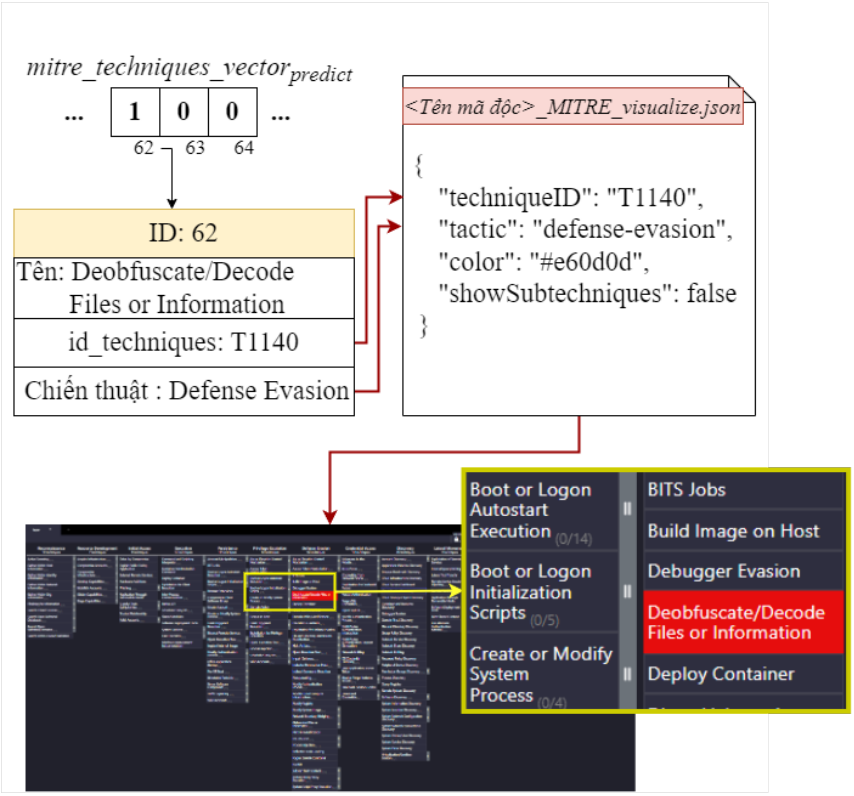
Thuật toán tối ưu	Adam
Kích thước lô	64
Số vòng lặp	200
Tốc độ học	0.001
Train:Validation	7:3

Bảng 3.2: Các tham số của quá trình huấn luyện

3.5 Kiểm tra - đánh giá kết quả

3.5.1 Kiểm tra

Trong bước kiểm tra, tôi xây dựng một mã nguồn để có thể chuyển kết quả dự đoán của mô hình học máy sang dạng một file json. File json này có thể mở trên trang MITRE ATT&CK® Navigator được cung cấp bởi The MITRE Corporation để có thể theo dõi trực quan kết quả dự đoán.Hình 3.12 minh họa cách một kết quả biểu diễn trên MITRE ATT&CK® Navigator.



Hình 3.12: Minh họa cách một kết quả biểu diễn trên MITRE ATT&CK® Navigator [20]

Trong đó, *mitre_techniques_vector*$_{predict}$ là kết quả dự đoán của mô hình học máy.Vị trí có id = 62 của *mitre_techniques_vector*$_{predict}$ có giá trị bằng 1 có nghĩa là mô hình dự đoán rằng mã độc có sử dụng kỹ thuật có ID=62. Tra cứu thông tin về kỹ thuật có ID = 62 trong bảng chương hai, ta có các thông tin: $id_{techniques}$="T1140", chiến thuật của kỹ thuật này là Defense Evasion. Các thông tin này sẽ được thay thế vào các vị trí trong mẫu có sẵn. Trong mẫu minh họa, "color" có giá trị là e60d0d tương ứng với màu đó, "showSubtechniques" bằng false tương ứng với việc không hiển thị ra các chi tiết kỹ thuật.

Trong trường hợp một kỹ thuật thuộc về nhiều hơn một chiến thuật thì tất cả các ô kỹ thuật thuộc về nhiều chiến thuật đó đều được biểu diễn.

3.5.2 Đánh giá kết quả

Để biết kết quả của một mô hình tốt đến mức nào, ta cần đánh giá kết quả đó trên tập kiểm tra. Hàm đánh giá được đề xuất trong bước này là Micro-averaging bao gồm Micro-average precision, Micro-average recall và F1-score được tính bằng công thức sau:

$$micro - average\ precision = \frac{\sum_{c=1}^{C} TP_c}{\sum_{c=1}^{C} (TP_c + FP_c)}$$

$$micro - average\ recall = \frac{\sum_{c=1}^{C} TP_c}{\sum_{c=1}^{C} (TP_c + FN_c)}$$

$$F - score = \frac{5}{\frac{1}{micro-average\ precision} + \frac{4}{micro-average\ recall}}$$

Trong đó:

Positive để chỉ các nhãn có giá trị bằng "1"

Negative để chỉ các nhãn có giá trị bằng "0"

TP là True Positive (Các nhãn "1" được dự đoán là "1").

FP là False Positive (Các nhãn "1" được dự đoán là "0")

FN là False Negative (Các nhãn "0" được dự đoán là "1")

C là tổng số nhãn ở lớp đầu ra của mô hình

micro-average precision thể hiện tỉ lệ số nhãn "1" được dự đoán đúng trên tổng số nhãn có giá trị "1" bao gồm cả các nhãn dự đoán đúng và các nhãn dự đoán sai. Giá trị của *micro-average precision* nằm trong khoảng (0,1] , *micro-*

average precision càng lớn có nghĩa là độ chính xác của các nhãn đúng tìm được càng cao, tuy nhiên mô hình có thể bị nhận nhầm các nhãn "0" thành "1".

micro-average recall thể hiện tỉ lệ số nhãn "1" được dự đoán đúng trên tổng số nhãn được dự đoán đúng (Các nhãn "1" được mô hình dự đoán là "1" và Các nhãn "0" được mô hình dự đoán là "0"). Giá trị của *micro-average recall* nằm trong khoảng (0,1] , *micro-average recall* càng lớn có nghĩa là độ chính xác của các nhãn đúng tìm được càng cao, tuy nhiên mô hình có thể bị nhận nhầm các nhãn "1" thành "0".

Cả *micro-average precision* và *micro-average recall* đều thể hiện rằng mô hình dự đoán càng chính xác khi 2 giá trị này càng cao. Tuy nhiên, trong bài toán thực tế, nếu ta điều chỉnh model để tăng *micro-average recall* quá mức có thể dẫn đến *micro-average precision* giảm và ngược lại, cố điều chỉnh model để tăng *micro-average precision* có thể làm giảm *micro-average recall*. Đối với bài toán phân tích mã độc dựa trên MITRE ATT&CK, việc nhận nhầm một kỹ thuật từ không thành có không nguy hiểm bằng việc nhận nhầm một kỹ thuật từ có thành không, do đó, trong trường hợp này, *micro-average recall* sẽ quan trọng hơn *micro-average precision*. Để cân bằng được hai kết quả đánh giá *micro-average precision* và *micro-average recall*, ta tính F-score với tỉ lệ:

$$\textit{micro-average precision} : \textit{micro-average recall} = 1 : 4$$

CHƯƠNG 4. ĐÁNH GIÁ THỰC NGHIỆM

4.1 Kết quả xây dựng bộ dữ liệu

4.1.1 Kích thước bộ dữ liệu

Với giải pháp thu thập và lưu trữ được đề xuất, bộ dữ liệu được xây dựng gồm 21000 mẫu, các file xxx-mitre.json, xxx-signature.json và xxx-behavior.json, mỗi loại có 21000 file. Toàn bộ dữ liệu có 63000 file, tổng dung lượng là 95 Gigabyte.
Một file xxx-mitre.json có dung lượng trung bình là 12.5197 Kilobytes.
Một file xxx-signature.json có dung lượng trung bình là 5.0449 Kilobytes.
Một file xxx-behavior.json có dung lượng trung bình là 4691.6601 Kilobytes.

4.1.2 Cấu trúc bộ dữ liệu

a, Hệ điều hành

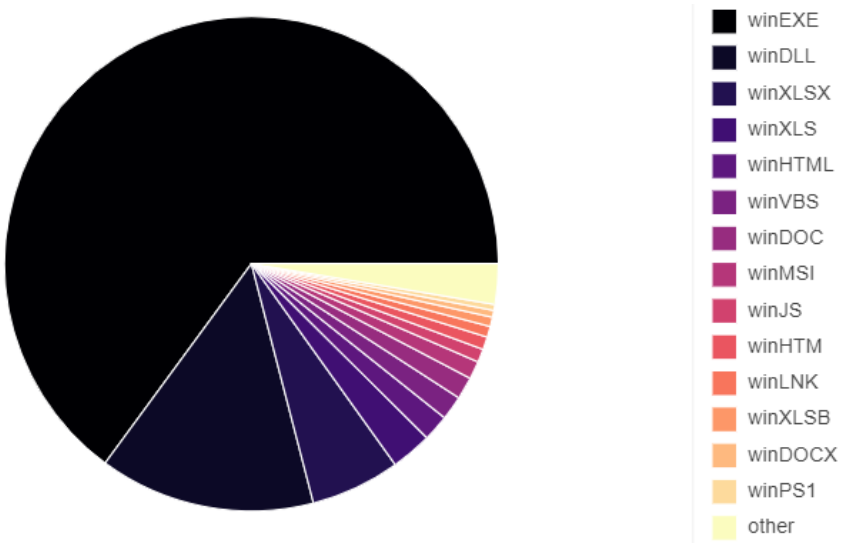
Trong bộ dữ liệu được xây dựng, 100% mẫu là mã độc chạy trên hệ điều hành Windows.

b, Định dạng file

Hình 4.1 là biểu đồ tỉ lệ về định dạng file của các mẫu mã độc trong bộ dữ liệu. Bảng là bảng số liệu chi tiết của các định dạng file. Mã độc có định dạng EXE chiếm tỉ lệ nhiều nhất với 65,5%

STT	Định dạng file	Số lượng	Tỉ lệ
1	EXE	13756	65.5%
2	DLL	2995	14.3%
3	XLSX	1241	5.9%
4	XLS	567	2.7%
5	HTML	365	1.7%
6	VBS	355	1.7%
7	DOC	308	1.5%
8	MSI	248	1.2%
9	JS	184	0.9%
10	HTM	177	0.8%
11	LNK	150	0.7%
12	XLSB	130	0.6%
13	DOCX	95	0.5%
14	Khác	429	2.0%
	Tổng cộng	21000	100%

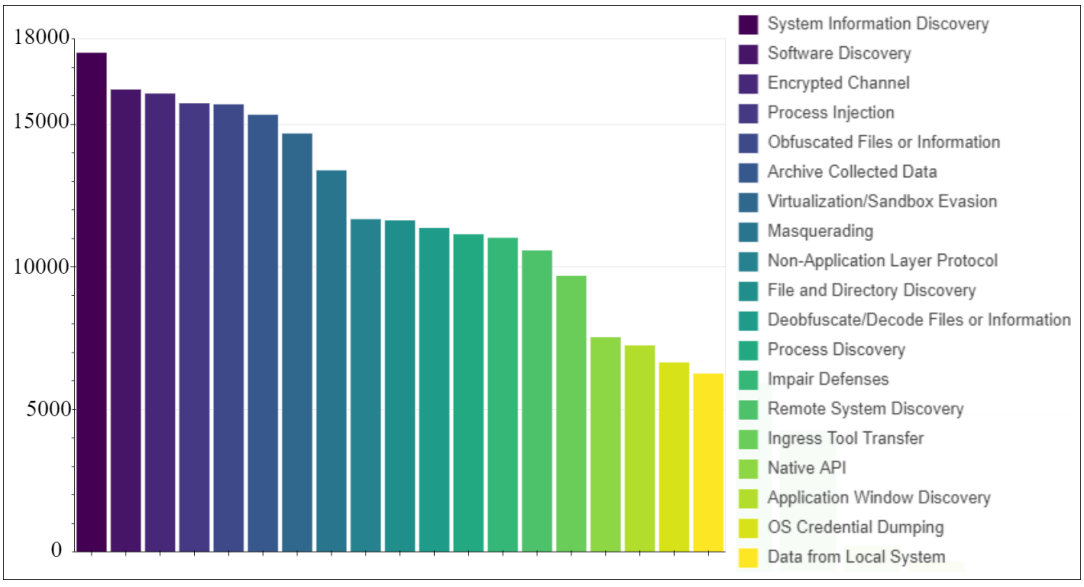
Bảng 4.1: Số liệu về định dạng file trong bộ dữ liệu



Hình 4.1: Biểu đồ tỉ lệ định dạng file

c, Nhãn kỹ thuật

Theo mô hình đề xuất, các nhãn kỹ thuật/chi tiết kỹ thuật của mã độc sẽ được quy đổi về 191 nhãn kỹ thuật. Bài toàn phân tích mã độc dựa trên MITRE ATT&CK, là bài toán phân loại đa nhãn, mỗi mẫu mã độc sẽ có nhiều hơn một nhãn. Trong bộ dữ liệu 21000 mẫu mà tôi đã xây dựng, trung bình mỗi mẫu mã độc có 17 nhãn kỹ thuật. Có 80 trên 191 kỹ thuật xuất hiện trong bộ dữ liệu, 111 kỹ thuật không xuất hiện trong bộ dữ liệu. Trong đó kỹ thuật có trong nhiều mẫu nhất là System Information Discovery xuất hiện trong 17521 mẫu. Có ba kỹ thuật chỉ xuất hiện trong mẫu một lần là Remote Services, Software Deployment Tools và Brute Force. Hình 4.2 là biểu đồ của 20 nhãn kỹ thuật MITRE ATT&CK có số lượng nhiều nhất trong bộ dữ liệu. Bảng 4.2 là bảng số liệu chi tiết của hình 4.2 với 15 nhãn đầu tiên.



Hình 4.2: Biểu đồ cột của 20 nhãn kỹ thuật nhiều mẫu nhất

STT	Tên nhãn kỹ thuật	Số lượng
1	System Information Discovery	17521
2	Software Discovery	16230
3	Encrypted Channel	16087
4	Process Injection	15747
5	Obfuscated Files or Information	15710
6	Archive Collected Data	15342
7	Virtualization/Sandbox Evasion	14686
8	Masquerading	13393
9	Non-Application Layer Protocol	11680
10	File and Directory Discovery	11637
11	Deobfuscate/Decode Files or Information	11373
12	Process Discovery	11150
13	Impair Defenses	11028
14	Remote System Discovery	10587
15	Ingress Tool Transfer	9696

Bảng 4.2: 15/191 nhãn kỹ thuật có số lượng nhiều nhất

Bảng 4.3 mô tả 15 nhãn có số lượng nhiều nhất trên bảng 4.2 thuộc về 6 nhóm chiến thuật là (i) Privilege Escalation, (ii) Defense Evation, (iii) Discovery, (iv) Collection, (v) Command and Control.

<table>
<tr><th>Chiến thuật</th><th>Privilege Escalation</th><th>Defende Evasion</th><th>Discovery</th><th>Collection</th><th>Command and Control</th></tr>
<tr><td rowspan="7">Kỹ thuật</td><td>Process Injection</td><td>Deobfucate/ Decode Files or Information</td><td>File and Directory Discovery</td><td>Archive Collected Data</td><td>Encrypted Channel</td></tr>
<tr><td rowspan="6"></td><td>Impair Defenses</td><td>Process Discovery</td><td rowspan="6"></td><td>Non-Application Layer Protocol</td></tr>
<tr><td>Masquerading</td><td>Remote System Discovery</td><td rowspan="2">Ingress Tool Transfer</td></tr>
<tr><td>Obfucated Files or Infomation</td><td>Software Discovery</td></tr>
<tr><td>Process Injection</td><td>System Infomation Discovery</td><td rowspan="3"></td></tr>
<tr><td>Virtualization/ Sandbox Evasion</td><td>Virtualization/ Sandbox Evasion</td></tr>
<tr><td>Tổng</td><td>15757</td><td>81937</td><td>81810</td><td>15342</td><td>37463</td></tr>
</table>

Bảng 4.3: Bảng chiến thuật của 15 kỹ thuật nằm trong nhiều mẫu nhất

4.2 Kết quả xây dựng *MISMAC_multiset* bằng phương pháp MISMAC

Vì trong kiến trúc MIST, có thành phần là các đối số nên việc chọn đối số sẽ ảnh hưởng đến kích thước của *MISMAC_multiset*. Bảng là bảng về Số lượng đối số và Số phần tử trong *MISMAC_multiset*. Trong đó, khi chọn số lượng đối số là 1 thì trong *MISMAC_multiset* có 23616 hành vi, khi chọn số lượng đối số là 2 thì trong *MISMAC_multiset* có 441898 hành vi, khi chọn số lượng đối số là 3 thì trong *MISMAC_multiset* có 3773460 hành vi.

Số lượng đối số	Số phần tử trong *MISMAC_multiset*
1	23616
2	441898
3	3773460

Bảng 4.4: Số lượng đối số và số phần tử trong *MISMAC_multiset*

4.3 Kết quả của mô hình học máy đề xuất

Vì bộ dữ liệu được xây dựng có sự mất cân bằng trong nhãn và chưa đủ lớn cho 191 nhãn kỹ thuật nên ở bước huấn luyện mô hình, tôi chọn mười kỹ thuật cho lớp đầu ra của mô hình. Mười kỹ thuật được chọn được liệt kê trong bảng

Do giới hạn tính toán của phần cứng mà tôi sử dụng là 12Gb Ram nên khi huấn luyện mô hình, *MISMAC_multiset* được sử dụng là *MISMAC_multiset* có số lượng đối số bằng 1. Nếu như quá trình huấn luyện sử dụng *MISMAC_multiset* có số lượng đối số bằng 2 hoặc 3 thì trong quá trình đó, phần cứng sẽ hết dung lượng để tính toán và quá trình huấn luyện sẽ bị gián đoạn và chấm dứt.

Trong quá trình thử nghiệm, mô hình học máy tốt nhất tôi đào tạo được có kiến trúc, cách huấn luyện và kết quả như sau:

Tầng đầu vào: 23616 nơ-ron

Số tầng ẩn: *N_hidden* = 2

Tầng ẩn một: X_1 = 256 nơ-ron

Tầng ẩn hai: X_2 = 128 nơ-ron

Tầng đầu ra: 10 nơ-ron

Thuật toán tối ưu: Adam

Kích thước lô : 64

Số lần lặp : 200

Số tham số : 6080138

ID	Kỹ thuật	Mô tả
59	Process Injection	Đưa mã vào các quy trình để trốn tránh các biện pháp bảo vệ dựa trên quy trình cũng như có thể nâng cao các đặc quyền
68	Impair Defense	Sửa đổi một cách ác ý các thành phần của môi trường nạn nhân để cản trở hoặc vô hiệu hóa các cơ chế phòng thủ
88	Virtualization/Sandbox Evasion	Sử dụng nhiều phương tiện khác nhau để phát hiện và tránh các môi trường phân tích và ảo hóa
101	OS Credential Dumping	chuyển thông tin xác thực để lấy thông tin đăng nhập tài khoản và thông tin xác thực, thường ở dạng băm hoặc mật khẩu văn bản rõ ràng, từ hệ điều hành và phần mềm
105	Unsecured Credentials	tìm kiếm các hệ thống bị xâm nhập để tìm và lấy thông tin xác thực được lưu trữ không an toàn
122	Process Discovery	lấy thông tin về các quy trình đang chạy trên hệ thống
125	Software Discovery	lấy danh sách phần mềm và các phiên bản phần mềm được cài đặt trên hệ thống hoặc trong môi trường đám mây
126	System Information Discovery	lấy thông tin chi tiết về hệ điều hành và phần cứng, bao gồm phiên bản, các bản vá lỗi, hotfix, gói dịch vụ và kiến trúc
147	Data from Local System	tìm kiếm các nguồn hệ thống cục bộ, chẳng hạn như hệ thống tệp và tệp cấu hình hoặc cơ sở dữ liệu cục bộ, để tìm các tệp quan tâm và dữ liệu nhạy cảm trước khi Exfiltration
151	Email Collection	nhắm mục tiêu email của người dùng để thu thập thông tin nhạy cảm như bí mật thương mại hoặc thông tin cá nhân

Bảng 4.5: Danh sách các nhãn được lựa chọn để huấn luyện và kiểm tra

Mô hình học máy được huấn luyện bốn là và sau đó chọn ra những tập trọng số tối ưu nhất trong mỗi lần huấn luyện. Bốn tập trọng số này có tên lần lượt là:

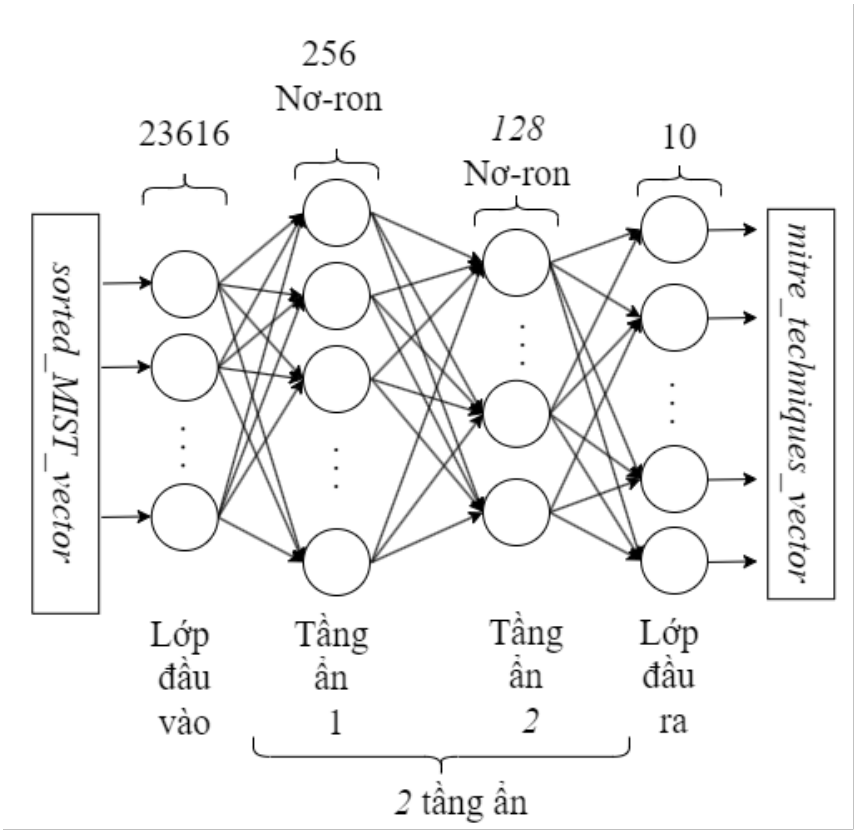
param_6080138_bz_64_loss_0.00802 (A)

param_6080138_bz_64_loss_0.01189 (B)

param_6080138_bz_64_loss_0.01567 (C)

param_6080138_bz_64_loss_0.02000 (D)

Kết quả của các mô hình được thể hiện trong bảng 4.6



Hình 4.3: Mô hình học máy có kiến trúc phù hợp với bộ dữ liệu

Tập trọng số	Tập huấn luyện			Tập kiểm tra		
	micro_average _precision	micro_average _recall	F_score	micro_average _precision	micro_average _recall	F_score
(A)	0.99368	0.95996	0.96652	0.95364	0.75294	0.78603
(B)	0.99116	0.96109	0.96696	0.94548	0.76080	0.79173
(C)	0.99043	0.96268	0.96811	0.93790	0.75631	0.78678
(D)	0.97878	0.96466	0.96745	0.93753	0.77783	0.80527

Bảng 4.6: Kết quả của bốn tập trọng số theo các hàm đánh giá

Qua bốn lần huấn luyện, F_score ở tập huấn luyện của cả bốn lần huấn luyện đều xấp xỉ 96,6%, trong đó, cao nhất là tập trọng số (C) với 96,8%. Với tập kiểm tra,tập trọng số có F_score cao nhất (D) với 80,5%.

4.4 Kết quả khi có áp dụng phương pháp MISMAC và không áp dụng phương pháp MISMAC

Để tìm hiểu sự khác nhau của độ chính xác của mô hình khi áp dụng phương pháp MISMAC và không áp dụng phương pháp MISMAC, tôi xây dựng một multiset không áp dụng phương pháp MISMAC (non-MISMAC). *non-MISMAC_multiset* là multiset không áp dụng phương pháp MISMAC được xây dựng dựa trên *MISMAC_multiset* bằng các hoán đổi ngẫu nhiên các phần tử trong *MISMAC_multiset* nhằm phá vỡ cấu trúc sắp xếp theo nhóm kỹ thuật MITRE ATT&CK của *MISMAC_multiset*.

Để tránh sự ngẫu nhiên trong quá trình huấn luyện mô hình, tôi huấn luyện mô hình có kiến trúc giống trong phần 2.3 ba lần với *non-MISMAC_multiset*. Ba tập trọng số tối ưu của ba lần huấn luyện này có tên lần lượt là:

non-MISMAC_param_6080138_bz_64_loss_0.00292 (NA)
non-MISMAC_param_6080138_bz_64_loss_0.00497 (NB)
non-MISMAC_param_6080138_bz_64_loss_0.00930 (NC)

Bảng 4.7 là kết quả so sánh chi tiết của 3 tập trọng số tối ưu nhất của cả hai trường hợp đối với tập kiểm tra. Trong đó, các cột micro_average_precision, micro_average _recall và F_score là các cột đánh giá, cột Eopchs là cột để ghi lại số lần lặp để đạt được tập trọng số tương ứng. Tập trọng số có kết quả được đánh giá tốt nhất ở trường hợp MISMAC là tập trọng số (D) với F_score = 0.80527 và tập trọng số này đạt được ở lần lặp thứ 14/200. Trong trường hợp non-MISMAC, tập trọng số (NA) là tập trọng số tối ưu nhất với F_score = 0.80108 và đạt được ở lần lặp thứ 75/200. Từ kết quả ta có thể thấy độ chính xác ở hai trường hợp là tương đương nhau nhưng số lần lặp để đạt được trọng số tối ưu ở trường hợp MISMAC ít hơn sơ với trường hợp non-MISMAC.

MISMAC					non-MISMAC				
Tên	micro_average _precision	micro_average _recall	F_score	Epoch	Tên	micro_average _precision	micro_average _recall	F_score	Epoch
(B)	0.94548	0.76080	0.79173	22	(NA)	0.94492	0.77171	0.80108	75
(C)	0.93790	0.75631	0.78678	26	(NB)	0.95027	0.76209	0.79352	48
(D)	0.93753	0.77783	0.80527	14	(NC)	0.94962	0.76579	0.79663	55

Bảng 4.7: Kết quả của các tập trọng số trong hai trường hợp MISMAC và non-MISMAC

CHƯƠNG 5. KẾT LUẬN

5.1 Kết luận

Trong đồ án này, tôi đã trình bày một giải pháp để phân tích mã độc dựa trên MITRE ATT&CK Framework và các kỹ thuật học sâu. Giải pháp bao gồm bốn bước là (i)thu thập và lưu trữ dữ liệu, (ii) tiền xử lý và vector hóa dữ liệu, (iii)huấn luyện mô hình học máy, (iv)kiểm tra và đánh giá kết quả.

Dữ liệu cho bài toán được tôi tự xây dựng với thông tin của 21000 mẫu mã độc thực thi trên hệ điều hành Windows có dung lượng 95 Gigabytes, có xuất hiện 80 trên 191 kỹ thuật của MITRE ATT&CK Framework. Trung bình mỗi mẫu mã độc sử dụng 17 kỹ thuật trong 191 kỹ thuật của MITRE ATT&CK Framework. Đây là một bộ dữ liệu lớn có chứa nhiều thông tin và có thể sử dụng cho nhiều bài toán khác nhau như phát hiện mã độc, phân tích mã độc, phân loại mã độc.

Để có thể phân tích hành vi của mã độc một cách hiệu quả, đồ án sử dụng cấu trúc MIST để biểu diễn dữ liệu sang một dạng ngắn gọn hơn. Cấu trúc MIST dã được tôi thiết kế lại để phù hợp với cấu trúc của bộ dữ liệu đã xây dựng. Bên cạnh đó, đồ án đề xuất một phương pháp mới là MISMAC với ý tưởng sắp xếp các phần tử (các hành vi MIST) trong multiset của BoW theo các nhóm kỹ thuật tương ứng trong MITRE ATT&CK Framework.

Với bộ dữ liệu đã được vector hóa, tác giả sử dụng một mô hình MLP với kiến trúc tự đề xuất. Do sự thiếu đa dạng về mẫu và nhãn kỹ thuật trong bộ dữ liệu, thay vì thực hiện bài toán phân loại đa nhãn với 191 nhãn là 191 kỹ thuật của MITRE ATT&CK, tôi thực hiện bài toán phân loại đa nhãn với 10 nhãn trong số đó. Độ chính xác 96,7% trên tập huấn luyện và 80,5% trên tập kiểm tra cho thấy tính khả thi của giải pháp với bài toán được nêu ra. Qua kết quả của bốn lần huấn luyện trên cùng một mô hình đề xuất, có thể thấy được sự ổn định và chính xác về kết quả của mô hình MLP được đề xuất đối với bài toán này.

Bên cạnh đó, với kết quả thực nghiệm trong hai trường hợp MISMAC và non-MISMAC, tuy rằng độ chính xác trong hai trường hợp là tương đương nhau nhưng ta có thể thấy sự hội tụ nhanh hơn trong quá trình huấn luyện dữ liệu ở phương pháp MISMAC.

Từ các kết quả trên ta có thể thấy giả pháp được đề xuất là khả thi và có thể cho kết quả tốt với bài toán phân tích mã độc dựa trên MITRE ATTCK Framework. Trong trường hợp pháp triển phải pháp với nhiều nhãn hơn, có thể ta sẽ cần thay đổi một mô hình mới để phù hợp với bộ dữ liệu.

5.2 Hướng phát triển trong tương lai

Nhờ vào việc phát triển hoàn thiện một quy trình giải quyết bài toán học máy từ bước thu thập dữ liệu đến xử lý dữ liệu, huấn luyện mô hình học máy và ứng dụng kết quả, trong tương lai, tôi sẽ tiếp tục phát triển giải pháp này với hai nhiệm vụ: tự động hóa quy trình huấn luyện và tăng chất lượng của mô hình học máy. Việc tự động hóa các bước trong giải pháp không chỉ giúp giải pháp trở lên hoàn thiện hơn, có tính thực tiễn cao hơn mà còn hỗ trợ nhiệm vụ huấn luyện mô hình học máy trong tác vụ học tăng cường.

Với kết quả đã đạt được ở đồ án này, tôi sẽ tiếp tục triển khai giải pháp ở một phạm vi rộng hơn. Đầu tiên là tăng tính sự đa dạng và chính xác trong bộ dữ liệu, không chỉ có mã độc của hệ điều hành Windows, mã độc chạy trên các hệ điều hành khác như Linux hay Android cũng sẽ được thu thập. Sau khi phát triển bộ dữ liệu thì mô hình học máy có thể huấn luyện với nhiều nhãn hơn và kết quả sẽ trở nên có ý nghĩa hơn đối với các phân tích mã độc.

TÀI LIỆU THAM KHẢO

[1] AVTEST - The Independent IT-Security Institute, *Malware(HTTP)*. **url**: https://www.av-test.org/en/statistics/malware/ (**urlseen** 16/07/2022).

[2] C. Ventures, *Global ransomware damage costs predicted to exceed $265 billion by 2031*. **url**: https://cybersecurityventures.com/global-ransomware-damage-costs-predicted-to-reach-250-billion-usd-by-2031/ (**urlseen** 16/07/2022).

[3] T. M. Corporation, *Att&ck*. **url**: https://attack.mitre.org/ (**urlseen** 16/07/2022).

[4] Hex-Rays, *Ida pro*. **url**: https://hex-rays.com/ida-pro/ (**urlseen** 16/07/2022).

[5] V. 35, *Binary ninja*. **url**: https://binary.ninja/ (**urlseen** 16/07/2022).

[6] I. A. D. A. Utku **and** M. A. Akcayol, "Permission based android malware detection with multilayer perceptron," *Signal Processing and Communications Applications Conference, , 2018, pp. 1-4*, 2018. **url**: https://doi.org/10.1109/SIU.2018.8404302.

[7] A. V. C. N. Udayakumar V. J. Saglani **and** T. Subbulakshmi, "Malware classification using machine learning algorithms," *International Conference on Trends in Electronics and Informatics (ICOEI), 2018, pp. 1-9*, 2018. **url**: https://doi.org/10.1109/ICOEI.2018.8553780.

[8] A. S. S. S. S. M. Rathore H., "Malware detection using machine learning and deep learning," *Big Data Analytics. BDA 2018. Lecture Notes in Computer Science(), vol 11297*, 2018. **url**: https://doi.org/10.1007/978-3-030-04780-1_28.

[9] L. Qiyou, "Malware behavior analysis with att ck framework," phdthesis, National Taiwan University, 2020.

[10] B. T. L. L. Zhongkun Yu JunFeng Wang, "Tactics and techniques classification in cyber threat intelligence," *The Computer Journal, 2022;, bxac048*, 2022. **url**: https://doi.org/10.1093/comjnl/bxac048.

[11] P. Trinius, C. Willems, T. Holz **and** K. Rieck, "A malware instruction set for behavior-based analysis," **in***Sicherheit* 2010.

[12] P. W. C. H. T. Rieck Konrad Trinius, "Automatic analysis of malware behavior using machine learning," *Journal of Computer Security. 19. 639-668*, 2011. **url**: https://doi.org/10.3233/JCS-2010-0410.

[13] T. M. Corporation, *Enterprise techniques*. **url**: https://attack.mitre.org/matrices/enterprise/ (**urlseen** 16/07/2022).

[14] Y. B. Y. LeCun **and** G. Hinton, "Deep learning," *Nature, vol. 521, no. 7553, pp. 436–444*, 2015. **url**: https://www.nature.com/articles/nature14539.

[15] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain," *Psychol. Rev., vol. 65, no. 6, pp. 386–408, 1958*, 2016. **url**: http://doi.org/10.1037/h0042519.

[16] Joe Security LLC, *Automated malware analysis - joe sandbox cloud basic, urldate = 2022-08-03, url = https://www.joesandbox.com*.

[17] SeleniumHQ, *Selenium*. **url**: hhttps://github.com/SeleniumHQ/selenium (**urlseen** 03/08/2022).

[18] Joe Security LLC, *Windows analysis report payment.xls*. **url**: https://www.joesandbox.com/analysis/555555/0/html (**urlseen** 03/08/2022).

[19] P. Kingma **and** J. Ba, "Adam: A method for stochastic optimization," *ArXiv14126980 Cs, Jan. 2017*, 2017. **url**: http://arxiv.org/abs/1412.6980.

[20] T. M. Corporation, *Mitre attck® navigator*. **url**: https://mitre-attack.github.io/attack-navigator/ (**urlseen** 01/08/2022).

PHỤ LỤC