HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

NGUYỄN VIỆT DŨNG

PHÁT HIỆN SỚM MÃ ĐỘC IOT BOTNET TRÊN CÁC THIẾT BỊ IOT

CHUYÊN NGÀNH: HỆ THỐNG THÔNG TIN

MÃ SỐ: 8.48.01.04

TÓM TẮT LUẬN VĂN THẠC SỸ

NGƯỜI HƯỚNG DẪN KHOA HỌC: PGS TS PHẠM VĂN CƯỜNG

HÀ NỘI – 2021

Luận văn được hoàn thành tại:

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

Người hướng dẫn khoa học: PGS TS Phạm Văn Cường

Phản biện 1: TS Hoàng Xuân Dậu

Phản biện 2: PGS TS Nguyễn Hà Nam

Luận văn sẽ được bảo vệ trước Hội đồng chấm luận văn thạc sĩ tại Học viện Công nghệ Bưu chính Viễn thông

Vào lúc: 08 giờ 30 ngày 28 tháng 8 năm 2021

Có thể tìm hiểu luận văn tại:

- Thư viện của Học viện Công nghệ Bưu chính Viễn thông.

MỞ ĐẦU

1. Tính cấp thiết của đề tài

Với sự gia tăng không ngừng về số lượng của các thiết bị IoT, Cisco dự đoán số lượng thiết bị IoT kết nối vào Internet sẽ là 50 tỉ thiết bị vào năm 2020. Cùng với thực trạng số lượng tăng lên nhanh chóng theo từng năm, những vấn đề bảo mật cho các thiết bị IoT đã được các nhà nghiên cứu chỉ ra trong những năm gần đây. Với đặc điểm bị hạn chế về tài nguyên, các thiết bị IoT dân dụng thường có mức độ bảo mật thấp hoặc không tồn tại cơ chế bảo mật cơ bản. Do thiếu các biện pháp bảo mật phù hợp, các thiết bị IoT dân dụng đã trở thành mục tiêu tấn công phổ biến, tạo nên mạng lưới IoT Botnet quy mô lớn và được sử dụng vào các cuộc tấn công mạng DDoS. Tiêu biểu, năm 2016 những kẻ tấn công đã xâm nhập thành công 600.000 thiết bị IoT dân sự, tạo ra mạng lưới IoT Botnet và thực hiện các cuộc tấn công DDoS với lưu lượng mạng đạt mức 1.1 Tbps.

Các giải pháp phát hiện mã độc IoT Botnet hiện nay đang tập trung vào việc phân tích các chuỗi dữ liệu hành vi đầy đủ (như luồng mạng, chuỗi lời gọi hệ thống,…) của mã độc để phát hiện các dấu hiệu độc hại tác động tới hệ thống thông tin. Tuy nhiên, chuỗi dữ liệu hành vi đầy đủ chỉ được sinh ra khi mã độc đã thực hiện thành công các hành vi độc hại tác động tới hệ thống. Điều này dẫn tới hạn chế khả năng giảm thiểu tác động của mã độc IoT Botnet đối với hệ thống thông tin. Để giải quyết vấn đề này, một số nhà nghiên cứu đã đưa ra ý tưởng phát hiện mã độc trước khi nó kịp thực hiện đầy đủ hành vi của mình (phát hiện sớm) dựa trên các mô hình học máy, một hướng nghiên cứu khả thi và hiệu quả để nâng cao khả năng phòng chống mã độc.

Với mục đích đưa những tiến bộ công nghệ và ứng dụng nghiên cứu vào phục vụ thực tế cuộc sống, luận văn với đề tài "*Phát hiện sớm mã độc IoT Botnet trên các thiết bị IoT*" sẽ tiến hành nghiên cứu, thực nghiệm và đưa ra mô hình học máy phù hợp để phát hiện sớm IoT Botnet.

2. Mục tiêu nghiên cứu

Nghiên cứu phương pháp phát hiện sớm mã độc IoT Botnet dựa trên công nghệ học máy và kết hợp nhiều nguồn dữ liệu hành vi của mã độc.

3. Đối tượng và phạm vi nghiên cứu

- Đối tượng nghiên cứu của đề tài: Luận văn tập trung vào nghiên cứu bài toán phát hiện mã độc IoT Botnet và được thực hiện trên các thiết bị IoT.

- Phạm vi nghiên cứu của đề tài: Trong đó, phạm vi nghiên cứu ở đây chỉ tập trung vào phân loại, phát hiện dòng mã độc **Botnet** và các thiết bị IoT ở đây tập trung vào dòng thiết bị **IoT dân dụng có tài nguyên hạn chế** nhưng rất phổ biến như Router, IP Camera, Smart TV Box, Smart phone,…

4. Nội dung và phương pháp nghiên cứu

- Nội dung nghiên cứu: Để đạt được mục tiêu đã đề ra ở trên, luận văn sẽ tập trung nghiên cứu, phân tích, đánh giá các nội dung sau:

+ Nghiên cứu, tìm hiểu đặc điểm, sự phát triển mã độc IoT Botnet và bài toán phát hiện mã độc IoT Botnet.

+ Khảo sát, phân tích, đánh giá các phương pháp phát hiện mã độc IoT Botnet và các phương pháp phát hiện sớm mã độc dựa trên công nghệ học máy.

+ Ứng dụng, thử nghiệm một mô hình học máy phù hợp để phát hiện sớm IoT Botnet.

+ Phân tích và đánh giá mô hình lựa chọn trên một tập dữ liệu đã có sẵn.

- Phương pháp nghiên cứu

Phương pháp nghiên cứu lý thuyết kết hợp với nghiên cứu thử nghiệm.

+ Nghiên cứu lý thuyết: Luận văn thực hiện nghiên cứu, khảo sát, tổng hợp, đánh giá các công trình nghiên cứu liên quan ở trong và ngoài nước để phân tích những vấn đề chưa giải quyết, những vấn đề còn tồn tại trong các phương pháp phát hiện mã độc IoT Botnet

+ Nghiên cứu thực nghiệm: Tiến hành thu thập, phân tích, tiền xử lý tập dữ liệu sử dụng cho quá trình thực nghiệm, ứng dụng vào mô hình học máy để đánh giá độ hiệu quả của mô hình, so sánh kết quả với các nghiên cứu đã có để nâng cao hiệu quả phát hiện mã độc IoT Botnet

5. Bố cục luận văn

Luận văn gồm phần mở đầu, 3 chương, phần kết luận và kiến nghị, tài liệu tham khảo với 66 trang, trong đó có 27 hình vẽ, 10 bảng và 50 tài liệu tham khảo. Cụ thể:

Trình bày tính cấp thiết và cấu trúc của luận án

Chương 1: Tổng quan về phát hiện sớm mã độc trên các thiết bị iot

Chương 2: Xây dựng mô hình học máy phát hiện sớm mã độc iot botnet

Chương 3: Thực nghiệm và đánh giá

Kết luận và kiến nghị

Danh mục tài liệu tham khảo

CHƯƠNG 1: TỔNG QUAN VỀ PHÁT HIỆN SỚM MÃ ĐỘC TRÊN CÁC THIẾT BỊ IOT

1.1. Tổng quan về các thiết bị IoT và IoT Botnet

Cụm từ IoT (Internet of Things - Vạn vật kết nối Internet) lần đầu được sử dụng bởi Kevin Ashton - nhà khoa học đã sáng lập ra Trung tâm Auto-ID tại Viện công nghệ Massachusets (MIT - Massachusetts Institute of Technology) vào năm 1999. Theo định nghĩa của Kevin Ashton, "*Internet of Things*" là "*tập hợp các thiết bị cảm biến và bộ điều khiển nhúng được kết nối thông qua môi trường mạng (có dây và không dây)*". Với định nghĩa của Kevin Ashton, thuật ngữ thiết bị "*IoT*" được sử dụng để chỉ các thiết bị cảm biến và bộ điều khiển nhúng điện tử.

Liên minh Viễn thông thế giới (ITU – International Telecommunication Union) cũng đã đưa ra khái niêm về "IoT", khái niệm này đã góp phần giúp làm sáng tỏ hơn về IoT. Theo ITU thì: "*Internet of Things là một cơ sở hạ tầng toàn cầu trong xã hội thông tin, nó cho phép các dịch vụ thông minh hoạt động bằng cách kết nối các vật thể bao gồm cả vật lý và ảo dựa trên các công nghệ thông tin truyền thông phù hợp hiện có và đang phát triển"*. Với khái niệm được nêu ở trên, IoT có thể được nhìn nhận trong một viễn cảnh rộng như là một tầm nhìn với những hàm ý về công nghệ và xã hội. Thông qua việc khai thác nhận dạng, thu thập,

xử lý dữ liệu và khả năng truyền thông, IoT tận dụng mọi thứ để hỗ trợ cho các loại ứng dụng, trong khi duy trì sự riêng tư cần thiết.

Theo khái niệm của ITU, vạn vật (Things) là đối tượng của thế giới thực (vật chất tồn tại) hoặc của thế giới thông tin (thực thể ảo), có thể được xác định và tích hợp vào mạng thông tin, truyền thông hiện có và đang phát triển. Vạn vật (Things) được định nghĩa phân thành hai loại chính:

- Physical things: Vật thể tồn tại trong thế giới vật chất và có khả năng cảm nhận, hoạt động tương tác trở lại môi trường và kết nối với các thực thể khác. Các thiết bị đại diện cho loại này có thể kể đến như: cảm biến nhiệt độ, các rô-bốt công nghiệp, phần cứng thiết bị nhúng gia dụng.

- Virtual things: Những thực thể ảo (không cảm nhận vật lý được) tồn tại trong thế giới thông tin và có khả năng lưu trữ, xử lý và truy cập dữ liệu. Một số ví dụ về Virtual things có thể kể đến như: nội dung đa phương tiện, phần mềm ứng dụng và các đại diện dịch vụ của các vật thể vật lý (tài khoản ảo).

Theo đó, ITU cũng đưa ra khái niệm thiết bị IoT là "*các thiết bị có khả năng kết nối và có thể cảm nhận thay đổi của môi trường, tương tác qua cơ cấu truyền động, thu thập, lưu trữ và xử lý dữ liệu*". Từ khái niệm thiết bị IoT này cho thấy rằng trong môi trường IoT, có rất nhiều loại thiết bị IoT khác nhau như: điện thoại di động thông minh, máy tính cá nhân, đồng hồ thông minh, smart TV, máy in, máy quét, IP Camera, thiết bị định tuyến, thiết bị gia dụng thông minh có kết nối Internet,... Các thiết bị IoT có mặt ở mọi nơi, hầu hết các ngành nghề, các mặt của đời sống con người như y tế, quản lý dây truyền sản xuất, quản lý năng lượng, hệ thống giao thông thông minh… Ngoài các tiện ích đem lại và sự có mặt trong nhiều mặt của cuộc sống, ngành nghề thì các thiết bị IoT cũng được dự báo sẽ đóng góp lớn vào nền kinh tế toàn cầu. Theo báo cáo của công ty IoT Analytics (nhà cung cấp hàng đầu về tìm hiểu thị trường cho IoT có trụ sở ở Đức) thì giá trị kinh tế toàn cầu do IoT mang lại sẽ từ 2.700 tỷ USD cho đến 6.700 tỷ USD trước năm 2025.

Để có thể đưa ra giải pháp phát hiện mã độc botnet phù hợp cho thiết bị mà luận văn này hướng tới, luận văn đi sâu làm rõ những đặc điểm của thiết bị IoT cỡ nhỏ bao gồm:

- *Môi trường hoạt động ít chịu sự điều khiển trực tiếp của con người*: Các thiết bị IoT có tính di động và tự hành cao theo các kịch bản hoạt động được cài đặt sẵn, hiếm khi cần sự điều khiển trực tiếp của con người (ví dụ: Thiết bị định tuyến, IP Camera, rô-bốt hút bụi trong gia đình,…).

- *Tính đa nền tảng phần cứng và phần mềm*: Khác với các thiết bị điện tử truyền thống như máy tính đa phần sử dụng vi xử lý kiến trúc i386 thì các thiết bị IoT cỡ nhỏ thường sử dụng nhiều loại các kiến trúc vi xử lý tiêu thụ năng lượng thấp như: MIPS, ARM, PowerPC, SPARC, MIPSEL,...

- *Tài nguyên phần cứng hạn chế*: Các thiết bị IoT cỡ nhỏ thường được trang bị các phần cứng hạn chế tài nguyên như dung lượng bộ nhớ ít, năng lực tính toán nhỏ, năng lượng pin dự trữ cho thời gian hoạt động ngắn.

- *Trạng thái động:* Trạng thái của các thiết bị IoT thay đổi linh hoạt, ví dụ như lúc hoạt động và ngủ chờ, lúc kết nối và ngắt kết nối... phụ thuộc vào hoàn cảnh của các thiết bị gồm vị trí, chức năng và tốc độ di chuyển.

- *Khả năng kết nối đa kênh:* Các thiết bị IoT có khả năng kết nối với các thiết bị và hạ tầng truyền dẫn theo nhiều giao thức khác nhau như Wifi, Bluetooth, Zigbee, Z-wave, LoRa, Lifi,...

1.2. Tổng quan mã độc IoT Botnet

Mặc dù có rất nhiều loại mã độc tấn công, lây nhiễm các thiết IoT, nhưng xu hướng mã độc botnet được xem là phổ biến nhất, gây hậu quả lớn nhất đối với các thiết bị IoT. Các hoạt động của mã độc IoT botnet gần đây cho thấy tội phạm mạng đang chuyển hướng lợi dụng thiết bị IoT để thực hiện các cuộc tấn công mạng với số lượng lớn thiết bị với băng thông cực lớn gây gián đoạn mạng Internet. Hiện nay, số lượng mã độc được ra đời với mục tiêu lây nhiễm, tấn công các thiết bị IoT cỡ nhỏ ngày càng tăng, cụ thể là theo báo cáo của hãng Kaspersky thì số lượng mã độc trên thiết bị IoT năm 2018 đã tăng gấp hơn 37 lần so với năm 2016,

Một cách tổng quan, mã độc (tiếng anh là Malicious Software/Code) là một chương trình hoặc một đoạn mã được bí mật chèn vào hệ thống người dùng nhằm gây hại hệ thống máy tính, hệ thống mạng, thông tin dữ liệu, … như tính tin cậy, tính bảo mật, tính toàn vẹn hoặc tính sẵn sàng.

Trong các loại mã độc xuất hiện trên thiết bị IoT cỡ nhỏ, phổ biến nhất là mã độc IoT Botnet (theo thống kê của Kaspersky). Tác giả Pamela và cộng sự đã đưa ra khái niệm IoT Botnet là "*một mạng lưới các thiết bị IoT cỡ nhỏ bị xâm nhập và lây nhiễm mã độc phục vụ xây dựng Botnet*". Theo khái niệm này kết hợp với phạm vi nghiên cứu của luận văn, mã độc IoT Botnet có thể hiểu là loại mã độc cho phép xây dựng mạng lưới Botnet dựa trên các thiết bị IoT cỡ nhỏ.

Thông qua việc khảo sát các loại mã độc IoT Botnet đã xuất hiện đến hiện nay, luân văn tổng quát lại mã độc IoT Botnet chứa hầu hết 2 thành phần cơ bản và 4 thành phần hỗ trợ gồm:

- Mã độc botnet thực hiện tấn công DDoS khi nhận lệnh;

- Máy chủ C&C để điều khiển mã độc botnet; Bộ Scanner để dò quét các thiết bị IoT mới có thể bị khai thác;

- Máy chủ Reporting có chức năng thu thập các dữ liệu dò quét của mã độc botnet hoặc bộ Scanner; Bộ Loaders được sử dụng để đăng nhập vào các thiết bị IoT có thể bị khai thác, và ra chỉ thị cho chúng tải về các tập tin thực thi mã độc có kiến trúc phù hợp;

- Máy chủ phân phối mã độc xác định vị trí lưu trữ mà độc sẽ được tải về bởi các thiết bị IoT đã bị lây nhiễm.

Trên thực tế một số mạng Botnet có thể sẽ có thêm hoặc giảm bớt một số thành phần cấu trúc, xong các thành phần đó vẫn thực hiện đầy đủ các chức năng được luận văn trình bày bên trên.

Với những thành phần đó, cơ chế hoạt động chung cho hầu hết mã độc IoT botnet như sau

Bước 1: Mã độc dò quét dải địa chỉ IP ngẫu nhiên thông qua TCP cổng 23/2323 để tìm kiếm các thiết bị IoT có lỗ hổng bảo mật để xâm nhập, lây nhiễm mở rộng mạng lưới máy tính botnet.

Bước 2: Sau khi dò quét được thiết bị có khả năng xâm nhập và đã thu thập được thông tin để xác thực và leo thang đặc quyền trên thiết bị thì mã độc sẽ gửi những thông tin đặc trưng của thiết bị về máy chủ Report thông qua các cổng dịch vụ khác

Bước 3: Mã độc nhận lệnh từ C&C để kiểm tra thông tin đặc tả của thiết bị như địa chỉ IP, kiến trúc phần cứng (MIPS, ARM, PowerPC, …)

Bước 4: Sau khi máy chủ C&C tiếp nhận thông tin đặc tả về thiết bị thì sẽ ra lệnh cho máy chủ Loader lựa chọn tập tin thực thi mã độc phù hợp.

Bước 5: Máy chủ Loader gửi tới thiết bị muốn xâm nhập tập tin mã độc phù hợp. Ngay sau khi tập tin mã độc được tải về và thực thi trên thiết bị thì mã độc sẽ xóa tập tin thực thi và chỉ chạy trong bộ nhớ RAM để tránh bị phát hiện, đồng thời mã độc sẽ tắt các dịch vụ cho phép truy cập từ xa như Telnet, SSH, vô hiệu hóa các chức năng của tường lửa.

Bước 6 và 7: Thông qua C&C kẻ tấn công có thể ra lệnh cho mã độc thực hiện các tấn công từ chối dịch vụ phân tán bằng nhiều kỹ thuật như UDP flood, SYN flood, GRE IP flood… tới một mục tiêu cụ thể.

Cũng giống như các loại mã độc truyền thống, sự tiến hóa của mã độc IoT Botnet cũng gia tăng theo sự thay đổi của công nghệ điện toán, các phương thức trao đổi dữ liệu Internet... Sự phức tạp và khả năng phá hoại của mã độc đã tăng lên kể từ khi công nghệ IoT ra đời với hàng ngàn, hàng tỷ thiết bị IoT được kết nối tới mạng Internet. Qua khảo sát cho thấy, các loại mã độc IoT Botnet vẫn đang được phát triển, tiến hoá thường xuyên với cấu trúc tinh vi và khả năng bảo vệ phức tạp hơn. Tuy nhiên, phần khảo sát các loại mã độc vẫn chưa đầy đủ bởi tội phạm mạng thường xuyên sửa đổi và cập nhật các loại mã độc đã biết để tạo ra các loại mã độc mới, khai thác nhiều loại thiết bị IoT hơn.

1.3. Các nghiên cứu liên quan trong phát hiện sớm mã độc

Các phương pháp phát hiện mã độc IoT botnet có thể chia thành 2 hướng tiếp cận chính như: (1) các phương pháp dựa trên chữ ký (signature-based) và (2) các phương pháp dựa trên hành vi (behavior-based). Các phương pháp dựa trên chữ ký sử dụng một chuỗi các byte có thể là mã băm, được trích xuất từ mã độc IoT botnet đã biết như một chữ ký đại diện đặc trưng duy nhất cho mỗi tập tin nhận dạng mã độc. Phương pháp dựa trên hành vi yêu cầu thực thi các tập tin trong một môi trường có giám sát và các hành vi sẽ được ghi nhận và kiểm tra làm căn cứ để phát hiện mã độc.

Ngày nay với xu hướng mã độc trên thiết bị IoT đặc biệt mà mã độc IoT botnet đang tăng trưởng không ngừng cả về số lượng và biến thể, kéo theo đó các dữ liệu về chữ ký và hành vi mã độc cũng tăng với số lượng lớn khiến việc xử lý, phân tích được thực hiện bằng con người trở nên rất khó khăn. Nhằm khắc phục các vấn đề trên, hiện nay các nhà nghiên cứu tiếp cận các phương pháp phát hiện mã độc dựa trên học máy. Phương pháp dựa trên học máy không sử dụng chữ ký hay hành vi mã độc cụ thể mà nó sử dụng các đặc trưng và các đặc trưng này được xem là thành phần lõi của phát hiện dựa trên học máy. Thông qua việc khảo sát có thể thấy, tất cả các hướng tiếp cận trên được nhóm thành 2 phương pháp chính là phân tích tĩnh (static) và phân tích động (dynamic).

1.3.1. Phân tích tĩnh

Phân tích tĩnh là phương pháp phân tích nội dung của mã nguồn mà không cần thực thi các tệp tin để phát hiện các hành vi nghi vấn. Phương pháp phân tích tĩnh cho phép chi tiết hóa toàn bộ luồng điều khiển (Control Flow Graph – CFG) và luồng dữ liệu (Data Flow Graph – DFG) thông qua các công cụ dịch ngược mã nguồn như IDA Pro, BinDiff… để phát hiện mã độc bằng phân tích đặc trưng như mã thực thi (Opcode), lời gọi hàm hệ thống (API calls) hay các chuỗi ký tự có nghĩa trong mã nguồn (Printable Strings Information – PSI). Phương pháp này cho phép phân tích chi tiết các tệp tin và đưa ra các khả năng kích hoạt của mã độc

Với những nghiên cứu được khảo sát, phân tích tĩnh đem lại kết quả khả quan trong bảo mật IoT nói chung và phát hiện mã độc IoT nói riêng. Tuy nhiên, phân tích tĩnh vẫn tồn tại nhiều hạn chế cho việc phân

tích, phát hiện mã độc IoT Botnet như: Khó áp dụng đối với mã độc sử dụng kỹ thuật gây rối (obfuscation) hoặc đóng gói (pack) phức tạp do hạn chế của công cụ Unpack và Debug; Khó thu thập mẫu mã độc do mã độc chỉ lưu trữ trên RAM thiết bị, và biến mất khi khởi động lại thiết bị; Kết quả dịch ngược có thể không chính xác do các tùy chọn biên dịch khác nhau của công cụ dịch ngược đối với các nền tảng CPU đa dạng của các thiết bị IoT. Vì vậy, với bài toán phát hiện mã độc IoT Botnet trên các thiết bị IoT hiện nay, hướng tiếp cận dựa trên phân tích tĩnh trở nên khó thực hiện.

1.3.2. Phân tích động

Phân tích động là phương pháp giám sát các hành vi trong khi các tập tin đó đang chạy, từ đó phát hiện có hay không các hành vi độc hại, bất thường. Môi trường thực thi các tập tin thường là một môi trường mô phỏng (như sandbox) hoặc các thiết bị IoT thực tế (như cài đặt các tác tử). Những thông tin được thu thập như các hành vi mức hệ thống (syscall, giá trị thanh ghi, dữ liệu bộ nhớ), các hành vi mức mạng (dữ liệu luồng mạng pcap). Phân tích động sẽ loại bỏ được các kỹ thuật gây rối mã nguồn, không dịch ngược được mã nguồn tệp tin thường gặp trong phân tích động. Tuy nhiên, khó khăn khi thực hiện phân tích động là việc xây dựng môi trường cho phép mã độc bộc lộ hoàn toàn các hành vi và có khả năng giám sát đầy đủ các hành vi đó.

Việc sử dụng phân tích động để phát hiện IoT Botnet có thể được phân loại theo hai phương pháp chính là phát hiện xâm nhập dựa trên dữ liệu mạng (Network-based Intrusion Detection System – NIDS) và phát hiện xâm nhập dựa trên dữ liệu máy chủ (Host-based Intrusion Detection System – HIDS).

Qua việc khảo sát các nghiên cứu trên về việc phát hiện mã độc Botnet sử dụng phương pháp phân tích động cho thấy, phần lớn các nhà nghiên cứu chú tâm vào việc phát hiện mã độc dựa trên luồng mạng NIDS. Tuy nhiên hướng tiếp cận này chỉ phát hiện được IoT botnet khi thiết bị đã bị lây nhiễm và bắt đầu truyền thông đến các máy chủ lệnh và điều khiển, các bot khác hoặc chúng thực hiện tấn công. Hình thức hiện mã độc dựa trên dữ liệu máy chủ HIDS có thể khắc phục được nhược điểm này. Tuy nhiên các thiết bị IoT khác với các thiết bị điện toán truyền thống vì chúng hạn chế về tài nguyên xử lý cũng như năng lượng. Hơn nữa với sự phát triển thần tốc về số lượng cũng như sự đa dạng về chức năng khiến cho các thiết bị IoT trở nên bất đồng nhất cả về kiến trúc phần cứng, giao tiếp truyền thông cũng như trạng thái hoạt động. Do đó, hướng tiếp cận phát hiện mã độc chỉ bằng HIDS là khó khăn và chưa đầy đủ.

1.4. Mô tả bài toán

Với sự phát triển nhanh chóng của các thiết bị IoT trên thế giới cả về số lượng lẫn chức năng, môi trường hoạt động. Do đó, các loại mã độc IoT Botnet cũng được tiến hoá thích ứng với môi trường của nạn nhân và khó bị phát hiện và phân tích hơn giúp duy trì hoạt động. Nhiều loại mã độc IoT Botnet gần đây đã được thiết kế để tránh bị phát hiện bởi các giải pháp bảo mật truyền thống hiện có hạn như phần mềm phát hiện và xử lý mã độc (Anti-virus), hệ thống phát hiện và xử lý xâm nhập mạng (IDS/IPS), các bộ lọc gói tin bằng tường lửa thông thường (Firewall). Các giải pháp bảo mật truyền thống này chỉ thực sự được phát hiện được mạng lưới IoT Botnet khi chúng đã ở trong giai đoạn thực thi tấn công từ chối dịch vụ phân tán (DDoS) và gây ra hậu quả được thấy rõ. Hiện nay cũng đã có nhiều phương pháp để phát hiện, chống lại các cuộc tấn công DDoS như vậy. Tuy nhiên, việc phát hiện và chống lại mạng Botnet khi các cuộc tấn công đã diễn ra là khó khăn vì khi đó mạng Botnet đã tích trữ một lượng lớn các bot cho mình. Minh chứng là có thời điểm có tới 400.000 thiết bị IoT bị lây nhiễm mã độc Mirai và vụ tấn công DDoS đã được ghi nhận với quy mô lớn nhất được thực hiện bởi mã độc Mirai có lưu lượng lên đến 1.2 Tbps vào năm 2016. Với số lượng lớn bot

như vậy kèm theo lưu lượng tấn công lớn sẽ dễ dàng đánh bại mọi hệ thống phần cứng hiện nay. Hơn thế nữa, giai đoạn rà quét và xâm nhập chiếm dụng quyền quản trị thiết bị mục tiêu (Bước 1 và 2) của mạng lưới IoT Botnet diễn ra trong thời gian dài. Nếu như có thể phát hiện các thiết bị IoT cỡ nhỏ bị lây nhiễm trước khi chúng thực thi lệnh tấn công từ chối dịch vụ (Bước 6 và 7) thì sẽ hạn chế được hậu quả phá hoại của mã độc IoT Botnet.

Về cơ bản, các phương pháp phát hiện mã độc đều dựa trên hai phương pháp chính là phân tích tĩnh và động như đã trình bày ở trên. Với những nội dung khảo sát và đánh giá được luận văn trình bày bên trên, luận văn lựa chọn sử dụng phương pháp phân tích động để có thể phát hiện sớm mã độc IoT Botnet. Phương pháp phát hiện được luận văn lựa chọn sẽ kết hợp được ưu điểm của các phương pháp phát hiện trước đây và hướng tới mục tiêu có thể phát hiện sớm mã độc IoT Botnet ngay từ giai đoạn đầu.

Kết luận chương

Trong chương 1, luận văn đã chỉ ra tổng quan về các thiết bị IoT cỡ nhỏ, đặc điểm mã độc IoT Botnet lây nhiễm trên loại thiết bị này và các giải pháp phát hiện mã độc IoT Botnet đã được công bố. Từ đó, nội dung chương đưa ra những vấn đề còn tồn tại và đề xuất phương án giải quyết các vấn đề này. Nội dung chi tiết phương pháp giải quyết vấn đề được nêu ra tại Chương 2 của luận văn.

CHƯƠNG 2: XÂY DỰNG MÔ HÌNH HỌC MÁY PHÁT HIỆN SỚM MÃ ĐỘC IOT BOTNET

Những nghiên cứu liên quan về vấn đề phát hiện mã độc IoT Botnet đa phần sử dụng các dữ liệu được thu thập theo thời gian gọi là dữ liệu chuỗi thời gian. Với số lượng lớn các thiết bị IoT thì lượng dữ liệu thu thập được là rất lớn và có thể gây khó khăn trong việc xây dựng các bộ phân loại. Phương pháp phát hiện mã độc IoT thường được xây dựng bằng cách xây dựng một bộ phân loại đơn lẻ trên tập dữ liệu thu thập đầy đủ theo chuỗi thời gian. Các phương pháp xây dựng bộ phân loại khác nhau thường có những ưu, nhược điểm khác nhau, nhưng điểm chung của các mô hình này chính là khi mã độc đã thực hiện toàn bộ hành vi trong hệ thống thì mới có thể phát hiện được, trong thực tế có thể gây hậu quả nghiêm trọng cho hệ thống nếu không cảnh báo trước nguy cơ. Để đáp ứng được yêu cầu phân loại chính xác trong việc sử dụng các bộ phân loại học máy, mô hình học máy cộng tác đã được luận văn đưa vào sử dụng để tăng hiệu suất dự đoán của mô hình. Mô hình học máy cộng tác đưa ra dự đoán từ việc sử dụng các bộ phân loại con và tổng hợp dự đoán từ những bộ phân loại này để đưa ra quyết định. Từ những điểm đã nêu trên, trong phần này luận văn sẽ xây dựng mô hình học máy phát hiện sớm mã độc IoT Botnet sử dụng các bộ dữ liệu có đặc trưng về thời gian của thiết bị, thực hiện các phương pháp chọn lọc và chuẩn hóa dữ liệu, áp dụng mô hình học máy cộng tác với các bộ phân loại con thích hợp cho bài toán phát hiện sớm mã độc IoT Botnet.

2.1. Tổng quan mô hình học máy cộng tác

Học máy cộng tác là quá trình sử dụng các bộ phân loại và kết hợp kết quả dự đoán của các bộ phân loại này để tạo nên một mô hình đưa ra quyết định phức tạp nhưng cải thiện hiệu năng dự đoán hơn so với các bộ phân loại con. Phương pháp kết hợp có thể linh hoạt dựa trên đặc trưng, hoặc kết quả phân loại.

Dựa vào phương thức kết hợp dữ liệu có thể chia các mô hình học cộng tác thành 3 nhóm chính:

- Hợp nhất sớm: là phương pháp hợp nhất các dữ liệu đầu vào bằng cách tạo ra một tập dữ liệu đại diện cho các tập dữ liệu con đơn lẻ. Tập dữ liệu đại diện này được sinh ra bằng cách nối các đặc trưng của

các tập dữ liệu con vào với nhau để tạo thành tập dữ liệu đại diện có chứa tất cả các đặc trưng của các tập dữ liệu con. Sau khi đã có được tập đại diện thì mô hình phân loại sử dụng một thuật toán học máy duy nhất để thực hiện quá trình phân loại dữ liệu đại diện.

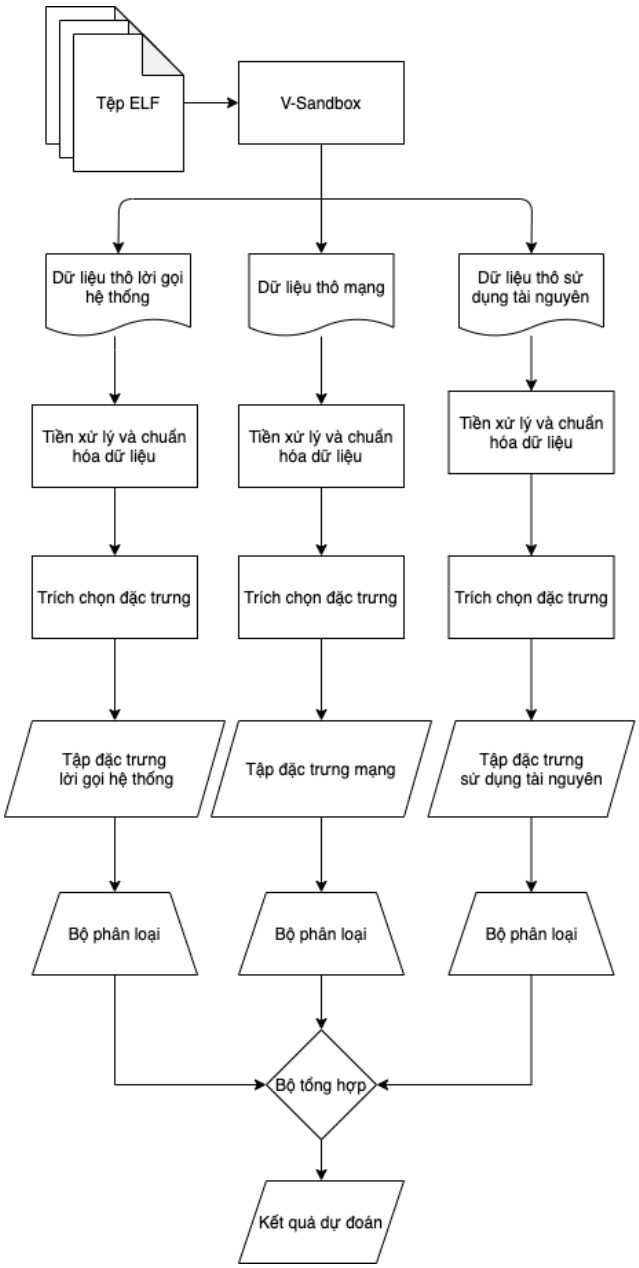
- Hợp nhất muộn: là phương pháp cho phép các tập hợp các kết quả phân loại của các bộ học máy phân loại (Classifier) đơn lẻ thông qua hàm hợp nhất (Fusion). Mỗi một bộ dữ liệu đặc trưng đầu vào sẽ được huấn luyện và phân loại dựa trên các thuật toán học máy riêng biệt. Kết quả phân loại sẽ được tổng hợp để đưa ra quyết định cuối cùng.

- Hợp nhất trung gian (Intermediate fusion): là cách hợp nhất các đặc trưng qua việc sử dụng các lớp ẩn (Hidden layer). Các đặc trưng đầu vào sẽ được đưa vào các thuật toán học máy có lớp ẩn để tìm ra các đặc trưng có liên quan tới mục tiêu phân loại. Kết quả phân loại từng bộ dữ liêu riêng lẻ này sẽ được đưa qua lớp hợp nhất và quyết định cuối cùng cũng sử dụng một lớp ẩn để tổng hợp kết quả.

Các mô hình học máy cộng tác kể trên đều có ưu điểm và nhược điểm, bằng việc nghiên cứu lý thuyết và qua quá trình thực nghiệm, luận văn đã lựa chọn sử dụng phương pháp hợp nhất muộn cho bài toán phát hiện sớm mã độc IoT Botnet và đạt được hiệu quả khả quan.

2.2. Mô hình ứng dụng

Trong phần này, luận văn sẽ ứng dụng mô hình học máy cộng tác phát hiện sớm IoT Botnet. Kiến trúc tổng quát của mô hình được biểu diễn cụ thể trong hình 2.1.



Hình 2.1. Kiến trúc mô hình ứng dụng

Tổng quát của mô hình có 5 thành phần chính để trích xuất và xử lý dữ liệu giúp đưa ra quyết định:

- Bộ phận thu thập dữ liệu;

- Bộ phận tiền xử lý và chuẩn hóa dữ liệu;

- Bộ phận trích chọn đặc trưng;

- Các bộ phát hiện dựa trên thuật toán học máy khác nhau;

- Bộ tổng hợp kết quả phát hiện.

Sự khác biệt để giải quyết vấn đề phát hiện mã độc IoT Botnet trong mô hình thử nhiệm so với các mô hình truyền thống là mô hình thử nghiệm sử dụng môi trường sandbox (trong trường hợp này là V-Sandbox) để lấy một phần nhỏ lượng dữ liệu đặc trưng cho các hành vi đầu tiên của tệp đầu vào đang được xử lý để thực hiện phân tích và phát hiện mã độc thay vì phải đợi mã độc thực hiện đầy đủ hành vi để thu thập và xử lý với toàn bộ dữ liệu.

Tổng quan quy trình xử lý và đưa ra quyết định của hệ thống được mô tả như sau. Các dữ liệu hành vi (bao gồm luồng mạng, lời gọi hệ thống và sử dụng tài nguyên của thiết bị) của tệp ELF đầu vào được thu thập trong môi trường sandbox. Các dữ liệu thu thập được này sẽ được tiền xử lý, chuẩn hóa và lựa chọn các đặc trưng phù hợp để đưa vào các mô hình bộ phân loại sử dụng thuật toán học máy. Những dự đoán của các mô hình học máy đơn lẻ này sau đó sẽ được đưa vào bộ tổng hợp để đưa ra kết luận phân loại cuối cùng.

Các thành phần và hoạt động cụ thể của chúng sẽ được trình bày trong phần tiếp theo và quá trình thực hiện cụ thể sẽ được trình bày trong chương 3 qua phần thực nghiệm và kết quả.

2.2.1. Bộ phận thu thập dữ liệu

Qua tìm hiểu, luận văn đã tìm được một số môi trường được xây dựng cho việc thu thập dữ liệu hành vi trong bài toán phát hiện mã độc trên các thiết bị IoT cỡ nhỏ như IOTBOX, Cuckoo, REMNUX, Limon và V-Sandbox. Mỗi một loại sandbox đều có nhưng ưu nhược điểm khác nhau và cụ thể về sự khác biệt trong việc hỗ trợ thu thập dữ liệu của các loại sandbox được mô tả trong Bảng dưới.

Bảng 2.1. Các tính năng của các mô trường Sandbox

Tính năng / **Sandbox**	**Khả năng hỗ trợ**			**Thu thập dữ liệu**		
	Đa kiến trúc CPU	**C&C Server**	**Thư viện liên kết động**	**Luồng mạng**	**Lời gọi hệ thống**	**Chiếm dụng tài nguyên**
IOTBOX	Có	Không	Không	Có	Không	Không
Cuckoo	Có	Không	Không	Có	Không đầy đủ	Không
REMNUX	Không	Không	Không	Không	Có	Không
Limon	Không	Không	Không	Có	Có	Không
V-Sandbox	**Có**	**Có**	**Có**	**Có**	**Có**	**Có**

Qua việc so sánh các tính năng của các môi trường Sandbox trong bảng trên, dễ dàng nhận thấy V-Sandbox có đầy đủ tính năng hơn so với các môi trường sandbox khác. Kiến trúc tổng quan của môi trường V-Sandbox được tác giả minh họa như trong Hình 2.5. Với đầu vào là các tệp định dạng ELF (được sử dụng phổ biến trong Linux), môi trường V-Sandbox tự động tạo môi trường phù hợp cho phép tệp này thực thi và giám sát các hành vi tương tác với hệ điều hành.

Từ nhưng ưu điểm và tính năng nêu trên, luận văn nhận thấy môi trường V-Sandbox rất phù hợp cho quá trình xây dựng môi trường mô phỏng các thiết bị IoT để thu dữ liệu hành vi nhằm giải quyết bài toán phát hiện sớm mã độc.

2.2.2. Bộ phận tiền xử lý và chuẩn hóa dữ liệu

Trong bài toán phát hiện mã độc IoT Botnet, các hành vi thường được những nhà nghiên cứu sử dụng để nhận biết dấu hiệu của chúng thường là: các lời gọi hệ thống, các gói tin trao đổi trong mạng và các

thay đổi về tài nguyên của hệ thống. Do đó, luận văn sử dụng ba loại hành vi phổ biến kể trên để thu thập dữ liệu phục vụ cho đánh giá khả năng phát hiện mã độc IoT Botnet của mô hình ứng dụng.

Như đã đề cập trong mô tả tổng quan về mô hình ứng dụng, dữ liệu đưa vào mô hình học máy để phát hiện chỉ là các hành vi đầu tiên của tệp thực thi, không phải là dữ liệu toàn bộ hành vi mã độc cần thực hiện. Vì vậy, cần xác định lượng dữ liệu cần thiết là bao nhiêu để có khả năng phát hiện chính xác mã độc IoT Botnet. Để trả lời câu hỏi này, luận văn đã tiến hành khảo sát, so sánh sự khác nhau trong dữ liệu thu thập được của mã độc IoT Botnet và tệp lành tính. Sau khi thu được dữ liệu tương ứng với lượng dữ liệu được xác định kể trên, luận văn tiến hành trích xuất đặc trưng để thu được bộ dữ liệu vector làm đầu vào cho các thuật toán học máy. Quá trình xử lý riêng biệt của mỗi loại dữ liệu được trình bày chi tiết sau đây.

- Dữ liệu lời gọi hệ thống

Từ quá trình thống kê dữ liệu có trong bộ dữ liệu lời gọi hệ thống, luận văn nhận thấy số lượng lời gọi hệ thống của các tệp thực thi dữ liệu lành tính thường ít hơn số lượng lời gọi hệ thống của các tệp thực thi có chứa mã độc, các tệp lành tính thường chỉ có khoảng dưới 100 lời gọi hệ thống mỗi lần thực thi trong khi con số này đối với các tệp có chứa mã độc là trên 300. Do vậy luận văn chọn ngưỡng 300 lời gọi hệ thống làm ngưỡng ngắt giám sát cho V-Sandbox và sử dụng 300 lời gọi hệ thống đầu tiên này làm dữ liệu cho các bộ học máy huấn luyện và kiểm thử.

Để trích xuất đặc trưng từ dữ liệu lời gọi hệ thống, luận văn sử dụng đồ thị lời gọi hệ thống (System Call Graph) được đề xuất bởi tác giả Lê Hải Việt và các cộng sự. Tuy nhiên, dữ liệu đồ thị không thể được huấn luyện trực tiếp bằng các bộ phân loại thông dụng. Do đó, dữ liệu này phải được chuyển về dưới dạng véc-tơ. Quá trình này được thực hiện thông qua công cụ Graph2vec. Graph2vec là một kỹ thuật học không giám sát để chuyển đổi một đồ thị thành dạng véc-tơ số dựa trên ý tưởng hướng tiếp cận của thuật toán nhúng văn bản Doc2vec. Theo đó, Graph2vec học cách biểu diễn các đồ thị bằng cách xem toàn bộ một đồ thị như một văn bản và các đồ thị con như các từ tạo nên văn bản đó. Kết quả thu được là một không gian véc-tơ nhúng mà các đồ thị có cấu trúc giống nhau thì có véc-tơ đặc trưng gần nhau. Đặc điểm này giúp cho các thuật toán học máy có thể hoạt động hiệu quả hơn trong việc phân loại đồ thị.

- Dữ liệu luồng mạng

Quá trình thống kê dữ liệu có trong bộ dữ liệu luồng mạng cũng cho thấy số lượng gói tin sử dụng để giao tiếp giữa các tệp lành tính cũng ít hơn các tệp có chứa mã độc. Điều này có thể lý giải bằng lý thuyết của IoT Botnet vì chung thường xuyên phải kết nối và nhận lệnh từ máy chủ C&C, dẫn đến lưu lượng truy cập cao hơn bình thường. Kết quả thống kê cho thấy hầu hết các tệp cho số lượng gói tin được sử dụng nằm ở ngưỡng 50 gói tin. Do đó, quá trình phát hiện sớm sử dụng ngưỡng 50 gói tin đầu tiên để ngắt giám sát và tiến hành phát hiện mã độc.

Dữ liệu luồng mạng sau khi thu được từ môi trường V-Sandbox là các tệp tin định dạng PCAP. Luận văn sử dụng công cụ CICFlowMeter để trích xuất đặc trưng dữ liệu từ các file PCAP này. Tuy nhiên, một tập tin PCAP chứa nhiều luồng mạng (flow), vì vậy để sinh được 1 véc-tơ duy nhất đại diện cho tập tin PCAP, luận văn tiến hành kết hợp thông tin của các luồng có trong tệp tin PCAP bằng cách thống kê các đại lượng tổng và giá trị lớn nhất. Bộ đặc trưng bao gồm 48 đặc trưng thống kê trên và 1 đặc trưng số lượng luồng mạng trong tập tin PCAP đó. Như vậy, có tất cả 49 đặc trưng được sử dụng.

- Dữ liệu sử dụng tài nguyên hệ thống

Thống kê bộ dữ liệu cho thấy các tệp thực thi mã độc thường yêu cầu sử dụng tài nguyên nhiều hơn so với các tệp thực thi lành tính và biểu hiện rõ nhất ở 20 trạng thái tài nguyên hệ thống đầu tiên. Do đó, luận văn chọn ngưỡng 20 trạng thái đầu tiên làm ngưỡng phát hiện sớm.

Dữ liệu sử dụng tài nguyên thiết bị bao gồm chuỗi nhiều trạng thái hiệu năng liên tiếp của hệ thống. Để có thể mô tả lại dữ liệu này, luận văn sử dụng phương pháp thống kê để định nghĩa các đặc trưng được sử dụng. Theo đó thông tin liên quan đến CPU, bộ nhớ của các trạng thái sẽ được tính toán các đại lượng trung bình cộng, độ lệch chuẩn, số lớn nhất, số nhỏ nhất. Với 20 đặc trưng thu được từ V-Sandbox, luận văn tiến hành thống kê và thu được 80 đặc trưng theo các 4 đại lượng trên.

2.2.3. Bộ phận trích chọn đặc trưng

Sau khi trích xuất đặc trưng và véc-tơ hóa, dữ liệu đã có thể đưa vào các bộ phân loại để huấn luyện/ kiểm thử. Tuy nhiên để nâng cao độ hiệu quả của mô hình, luận văn tiến hành trích chọn đặc trưng để nâng cao độ chính xác, độ hiệu quả cho mô hình.

Trích chọn đặc trưng giúp nâng cao độ chính xác, độ hiệu quả cho mô hình vì 2 lý do chính:

- Trích chọn đặc trưng giúp lược bỏ các đặc trưng thừa, không đóng góp nhiều vào quá trình phân loại của mô hình, cũng như các đặc trưng gây nhiễu, gây ảnh hưởng đến hiệu quả của bộ phân loại. Do đó, trích chọn đặc trưng vừa giúp nâng cao độ chính xác cho mô hình, vừa giảm thiểu tình trạng quá khớp (overfitting). - Trích chọn đặc trưng làm giảm chiều véc-tơ đặc trưng đầu vào, qua đó đẩy nhanh tốc độ tính toán, giúp mô hình hội tụ nhanh hơn.

Có nhiều phương pháp trích chọn đặc trưng, luận văn đã sử dụng phương pháp đơn giản Information gain và áp dụng lên 2 loại đặc trưng là dữ liệu luồng mạng và đặc trưng dữ liệu hiệu năng hệ thống.

Information Gain là đại lượng đo độ hỗn tạp hay hỗn loạn của đặc trưng. Trong cây quyết định, information gain đươc sử dụng làm tiêu chí phân chia một node. Một thuộc tính có information gain hay độ giảm entropy lớn sẽ được chọn để chia node, bởi vì tính bất định của thông tin được giảm xuống nhiều nhất. Trong bài toán trích chọn đặc trưng, information gain được sử dụng để đo độ liên quan của đặc trưng A đối với lớp C. Giá trị của thông tin chung (mutual information) giữa thuộc tính A và lớp C càng cao thì độ liên quan giữa chúng càng lớn.

$$I(C, A) = H(C) - H(C|A),$$

Trong đó $H(C) = -\sum p(C)\log p(C)$, entropy của lớp C, và $H(C|A)$ là entropy của lớp C với điều kiện thuộc tính A, $H(C|A) = \sum p(C|A).\log p(C|A)$.

Luận văn tiến hành tính thông tin chung giữa tất cả các đặc trưng và nhãn và thu được kết quả sau:

- Đối với đặc trưng mạng:

Kết quả lựa chọn đặc trưng được trình bày ở bảng 3. Một số đặc trưng cho giá trị thông tin chung tính được bằng 0 là do những đặc trưng này chỉ có 1 giá trị duy nhất trong cả bộ dữ liệu huấn luyện. Do đó, những đặc trưng này chắc chắn sẽ bị loại bỏ, vì không đóng góp gì vào quá trình phân loại của các thuật toán học máy. Thông qua thực nghiệm, luận văn đã loại bỏ 35 đặc trưng và trích chọn được 14 đặc trưng có độ ảnh hưởng lớn nhất là đầu vào cho các bộ phân loại ở bước tiếp theo.

Bảng 2.2. Các đặc trưng mạng được sử dụng

STT	Đặc trưng
1	Sum Flow Duration
2	Max Flow Duration
3	Sum Fwd Header Len
4	Sum TotLen Bwd Pkts
5	Sum Bwd Header Len
6	Max Fwd Header Len
7	Sum TotLen Fwd Pkts
8	Max Tot Fwd Pkts
9	Max Bwd Header Len
10	Sum Tot Fwd Pkts
11	Sum Tot Bwd Pkts
12	Sum Init Fwd Win Byts
13	Sum SYN Flag Cnt
14	Max Init Fwd Win Byts

- Đối với đặc trưng hiệu năng hệ thống:

Tương tự như trên, vẫn có một số đặc trưng chỉ có 1 giá trị duy nhất trong cả bộ dữ liệu huấn luyện và chắc chắn sẽ bị loại bỏ. Thông qua thực nghiệm, luận văn đã loại bỏ 62 đặc trưng và chọn 18 đặc trưng có độ ảnh hưởng lớn nhất là đầu vào cho các bộ phân loại ở bước tiếp theo.

Bảng 2.3. Các đặc trưng hiệu năng hệ thống được sử dụng

STT	Đặc trưng
1	swap_cache_min
2	swap_cache_mean
3	swap_cache_max
4	mem_used_max

5	mem_free_min
6	mem_buffers_mean
7	mem_used_mean
8	mem_free_mean
9	mem_buffers_max
10	mem_buffers_min
11	mem_used_min
12	mem_free_max
13	swap_cache_std
14	cpu_%_id_mean
15	cpu_%_us_std
16	num_total_sleeping_mean
17	cpu_%_sy_std
18	num_total_sleeping_std

2.2.4. Bộ tổng hợp dự đoán

Quá trình kết hợp kết quả phát hiện các bộ phân loại học máy khác nhau cần đến sự tham gia của một hàm tổng hợp. Trong học máy, các hàm tổng hợp kết quả phổ biến được sử dụng như Voting, Stacking, Bagging và Boosting. Luận văn lựa chọn phương pháp bầu chọn (Voting) để áp dụng cho bài toán phát hiện sớm của mình dựa trên kết quả thực nghiệm. Kết quả cuối cùng của một dự đoán được thực hiện bởi đa số "phiếu bầu" theo hai chiến lược khác nhau là biểu quyết cứng (hard voting) và biểu quyết mềm (soft voting).

- Biểu quyết cứng: Trong biểu quyết cứng (còn được gọi là biểu quyết đa số), mỗi bộ phân loại riêng lẻ đưa ra dự đoán dữ liệu đầu vào thuộc nhãn nào và nhãn chiếm đa số dự đoán sẽ chiến thắng.

$$\hat{y} = mode\{C_1(x), C_2(x), \dots, C_m(x)\}$$

Trong đó: $C_i(x)$ là kết quả dự đoán (nhãn) của bộ phân lớp thứ j

- Biểu quyết mềm: Trong biểu quyết mềm, mỗi bộ phân loại riêng lẻ đưa ra xác suất dự đoán dữ liệu đầu vào thuộc về từng nhãn tương ứng. Các giá trị dự đoán được đánh trọng số theo mức độ quan trọng của bộ phân loại và được tổng hợp lại. Sau đó, nhãn có tổng xác suất có trọng số lớn nhất sẽ được chọn.

$$\hat{y} = arg \max_i \sum_{j=1}^{m} w_j p_{ij}$$

Trong đó: w_i là trọng số của kết quả dự đoán (giá trị p) thuộc bộ phân lớp thứ j.

Trong phần thực nghiệm đánh giá được trình bày tại chương 3, luận văn sử dụng biểu quyết mềm vì các bộ phân loại đơn lẻ được thực nghiệm đều có đầu ra là xác suất dự đoán. Ngoài ra biểu quyết mềm xem xét xác suất, mức độ chắc chắn của từng bộ phân loại đơn lẻ rồi mới quyết định nên sẽ đem lại kết quả chính xác hơn so với với việc chỉ xem xét nhãn dự đoán của từng bộ phân loại như biểu quyết cứng.

Kết luận chương

Trong chương này, luận văn đã ứng dụng mô hình tổng quan cho bài toán phát hiện sớm mã độc IoT Botnet, mô tả cụ thể về chức năng, kỹ thuật sử dụng và mục đích của các thành phần chính. Mô hình sử dụng môi trường sandbox để thu thập 3 loại dữ liệu đặc trưng dựa trên những nền tảng nghiên cứu của các bên nghiên cứu liên quan đến chủ đề được đề cập. Những dữ liệu này sau đó được phân tích và xử lý phù hợp để đưa vào các bộ phân loại. Mô hình cộng tác được ứng dụng để giải quyết bài toán phát hiện với sự kết hợp để tăng hiệu năng cho các bộ phân loại đơn lẻ. Kết quả triển khai thực nghiệm mô hình với tập dữ liệu thực tế và đánh giá so sánh với các nghiên cứu đã công bố được trình bày trong chương tiếp theo của luận văn.

CHƯƠNG 3: THỰC NGHIỆM VÀ ĐÁNH GIÁ

Trong chương này sẽ trình bày về tập dữ liệu sử dụng cho quá trình thực nghiệm, các bước thực thi cụ thể từ mô hình tổng quan để đưa ra kết quả từ tập dữ liệu đầu vào. Sau khi thu được kết quả thực nghiệm sẽ thực hiện đánh giá độ hiệu quả của mô hình, các so sánh kết quả với các nghiên cứu đã có và từ đó đề ra phương hướng nghiên cứu phát triển sau này cho bài toán.

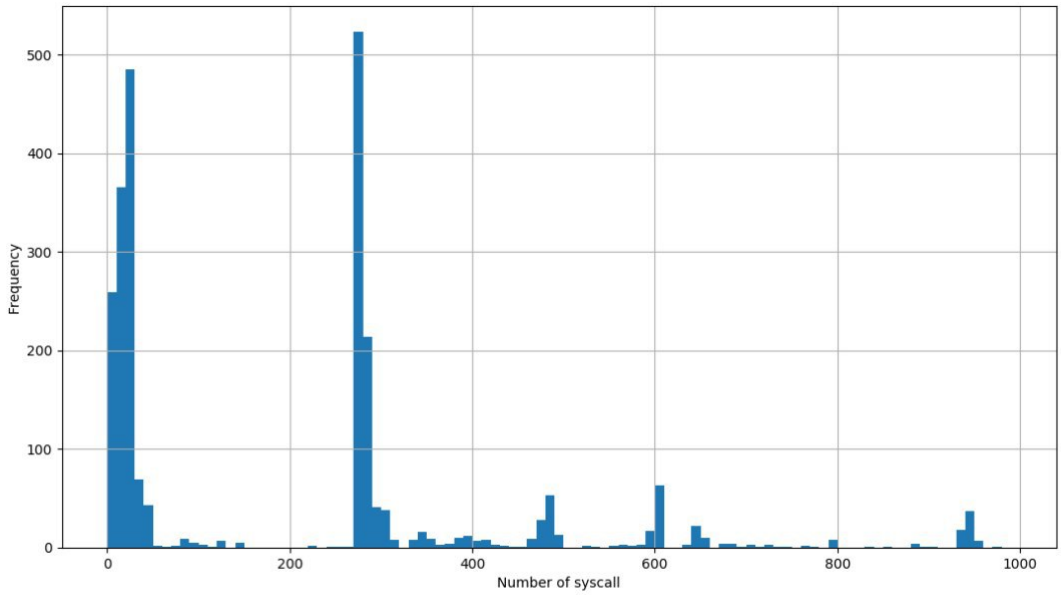
3.1. Bộ dữ liệu

Để đánh giá kết quả hoạt động của mô hình ứng dụng, luận văn sử dụng tập dữ chứa 8911 mẫu bao gồm 5023 IoT Botnet và 3888 mẫu lành tính đã được thu thập và sử dụng cho thực nghiệm. Chia tập dữ liệu theo tỉ lệ 70% bộ dữ liệu dùng để huấn luyện các mô hình và 30% bộ dữ liệu để kiểm nghiệm độ hiệu quả. Mô tả về các mẫu trong tập dữ liệu được mô tả trong bảng 3.1

Bảng 3.1. Mô tả bộ dữ liệu

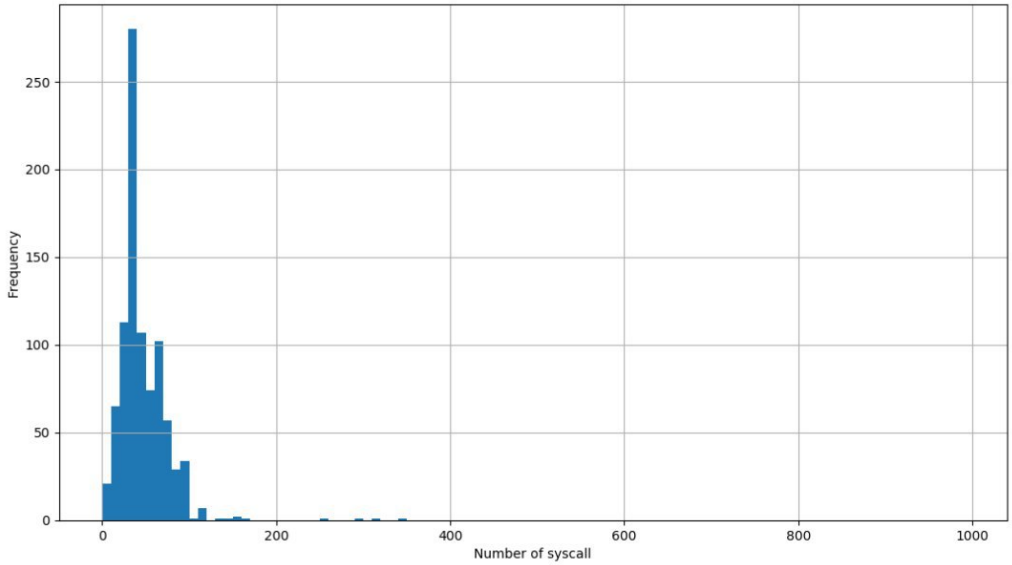
Loại	**Số lượng**
Mã độc Bashlite	2786
Mã độc Mirai	1510
Mã độc IoT Botnet khác (MrBlack, Spike, Dofloo,…)	727
Tệp lành tính	3888

Luận văn tiến hành phân tích, thống kê chi tiết và vẽ biểu đồ thống kê của các dữ liệu mã độc cũng như lành tính trong bộ dữ liệu để tạo cơ sở cho việc xác định ngưỡng phát hiện sớm như đã trình bày ở chương 2: 300 lời gọi hệ thống, 20 trạng thái tài nguyên hệ thống và 50 gói tin luồng mạng đầu tiên. Các kết quả thống kê như hình dưới.



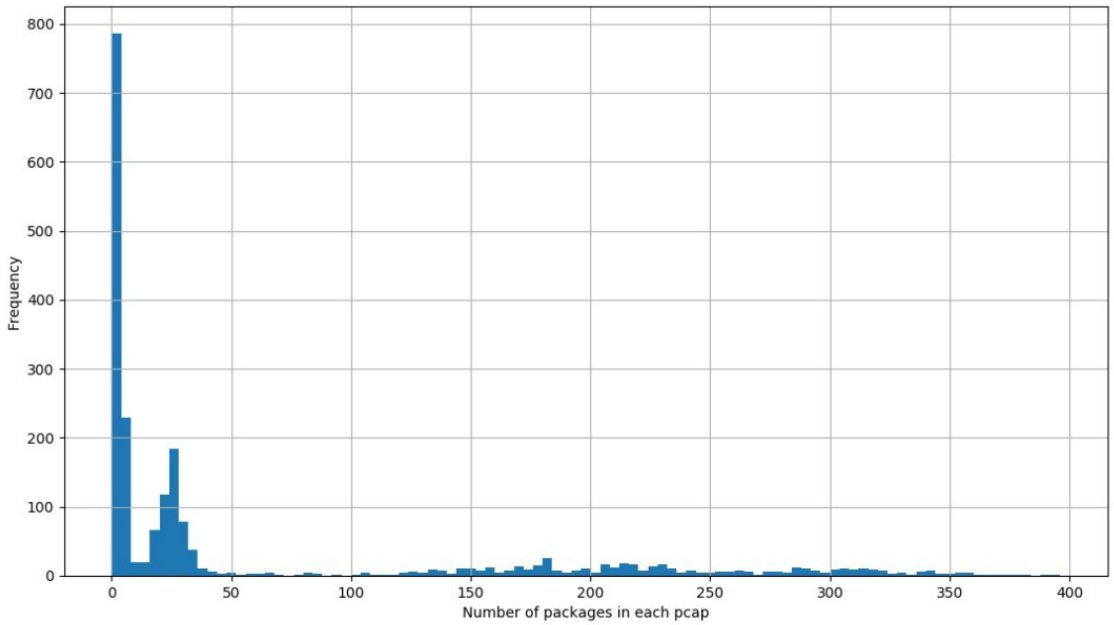
Hình 3.1. Thống kê số lượng lời gọi hệ thống của mã độc IoT Botnet

Tiến hành phân tích số lượng lời gọi hệ thống của các mẫu trong Dataset trong hình 3.1 và 3.2, luận văn nhận thấy các mã độc IoT Botnet chủ yếu thường thể hiện hành vi hoạt động dưới ngưỡng 300 lời gọi hệ thống, trong khi số lượng lời gọi hệ thống của các tệp lành tính thể hiện ở hình 3.5 tập trung chủ yếu dưới 100 lời gọi hệ thống.

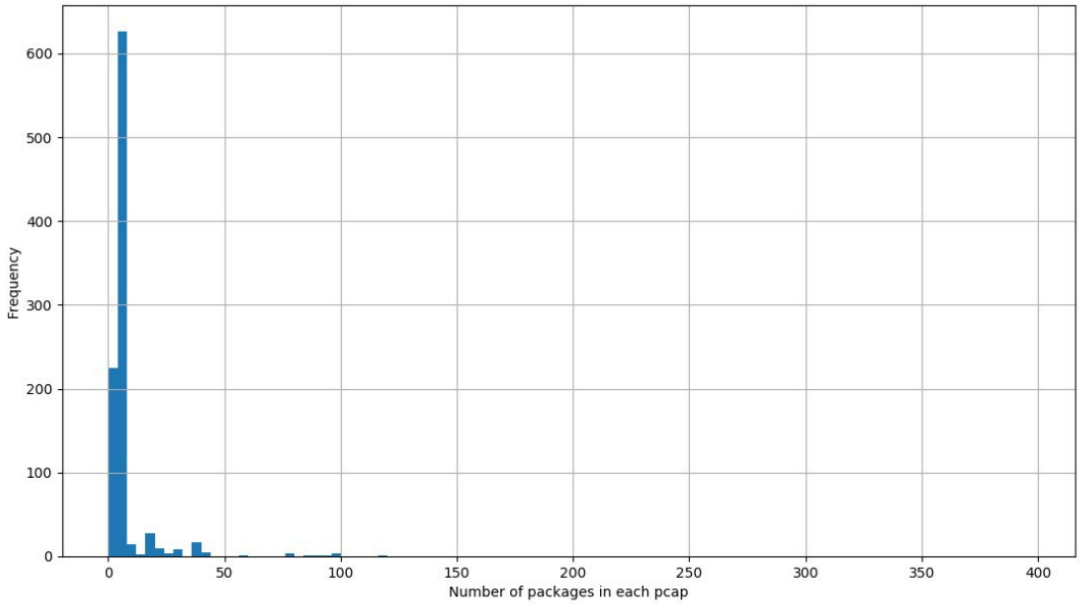


Hình 3.2. Thống kê số lượng lời gọi hệ thống của tệp lành tính

Hình 3.3 cho thấy lượng gói tin giao tiếp trong luồng mạng của IoT Botnet tập trung chủ yếu dưới 50 gói tin và có một số lượng mẫu IoT Botnet có thể lên đến 400 gói tin giao tiếp luồng mạng, trong khi đó lượng gói tin của các tệp lành tính thể hiện trong hình 3.4 gần như chỉ dưới ngưỡng 50 gói tin.

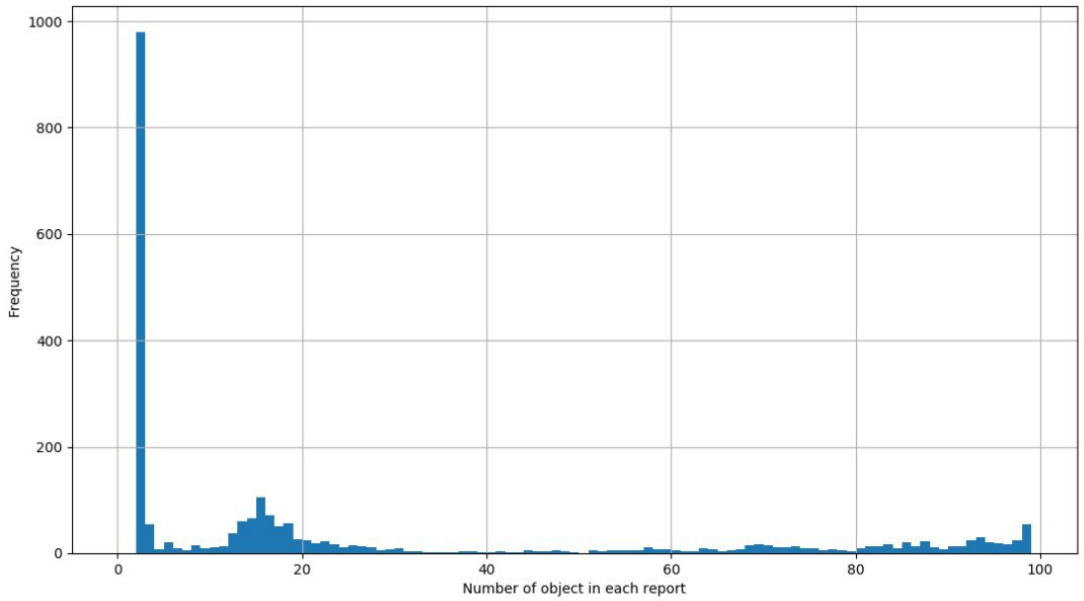


Hình 3.3. Thống kê số lượng gói tin mạng của IoT Botnet

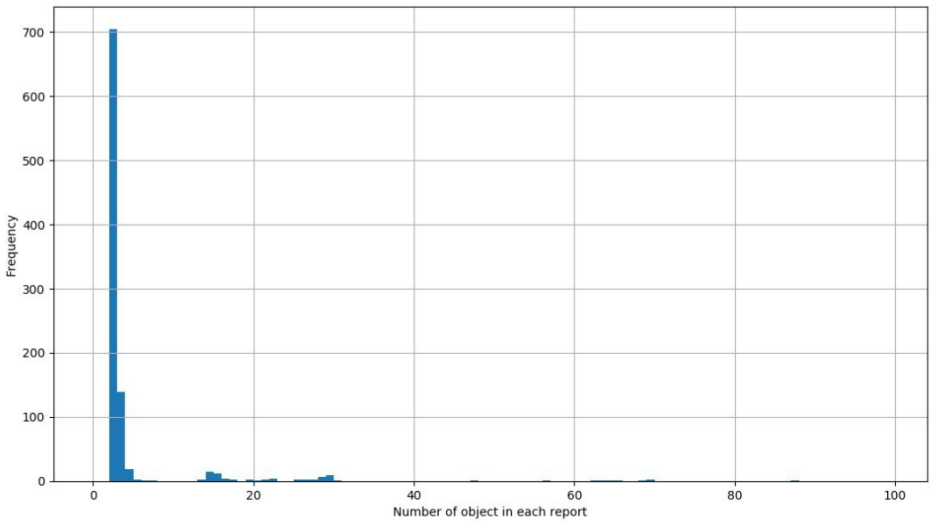


Hình 3.4. Thống kê số lượng gói tin mạng của tệp lành tính

Về thông tin yêu cầu sử dụng tài nguyên của hệ thống, hình 3.5 cho thấy các mẫu IoT Botnet thường có lượng yêu cầu sử dụng tài nguyên hệ thống cao, có thể lên đến 100 yêu cầu, nhưng chủ yếu lượng lớn vẫn tập trung ở ngưỡng 20 yêu cầu sử dụng tài nguyên. Hình 3.6 thể hiện các tệp lành tính thường không yêu cầu sử dụng tài nguyên nhiều như các mẫu IoT Botnet, các mẫu này đều có dưới 20 yêu cầu sử dụng tài nguyên hệ thống.



Hình 3.5. Thống kê yêu cầu chiếm dụng tài nguyên hệ thống của IoT Botnet



Hình 3.6. Thống kê yêu cầu chiếm dụng tài nguyên hệ thống của tệp lành tính

Tập dữ liệu sau khi được đưa vào bộ phận tiền xử lý và chuẩn hóa sẽ có các đặc trưng được biểu diễn như trong hình 3.7 – 3.9.

name	Num of flow	Sum Flow Duratio	Max Flow Duratic	Sum Tot Fwd Pkts	Max Tot Fwd Pkts	Sum Tot Bwd Pkts	Max Tot Bwd Pkts	Sum TotLen Fwd I	Max TotLen Fwd I	Sum TotLen Bwd	Max TotLen Bwd
06916bd06712bc6652ee5478250 8da93_1593707152	6	3642	1498	6	1	6	1	270	46	270	46
ca90ad8fefe28c724c2736984d6e95ec_1590180585	2	1844	1508	2	1	2	1	90	46	90	46
017662f941ebe50c7e0d3fd9a826144a_1589905104	2	2254	1644	2	1	2	1	90	46	90	46
9a983419ddc7ab33507b41e3685f33bf_1590173095	2	2017	1704	2	1	2	1	90	46	90	46
888ccce705a9450caef48e6cf7901781_1590141182	0	0	0	0	0	0	0	0	0	0	0
740cea9b9bdbb761c49270e72f4b77db_1589981287	6	7683	1806	6	1	6	1	270	46	270	46
f1085b90bdd41b23bb18e9a457e3a266_1593698468	2	1839	990	2	1	2	1	90	46	90	46
ade13c6d0182c53febc9b8486296e483_1590143440	0	0	0	0	0	0	0	0	0	0	0
7535ff0d990b6cfc8d20dc06f3a7201a_1593511352	2	1724	1478	2	1	2	1	90	46	90	46
e4c17224fd574e62ad0e8ae2a8306a26_1590216459	2	1876	1716	2	1	2	1	90	46	90	46
c5ce05ebee11ea7c06dc3e4bdd853931_1590172542	2	2345	1955	2	1	2	1	90	46	90	46
8ca7ec7b22b8bc8fc436672a62c1f7e9_1593691094	2	1755	1526	2	1	2	1	90	46	90	46
bf07a7a21d7ffa99bb8ab4035db5cdf6_1590162833	2	11074	9801	2	1	2	1	90	46	90	46
a553e97d88f7b7934276856f1a528716_1590132757	2	1727	1443	2	1	2	1	90	46	90	46
91d9701615b7b179a1b7eb666a490374_1590158787	2	10959	10624	2	1	2	1	90	46	139	95
318c43200fc8c71c24ca592e2eda9c9c_1589946963	2	2633	1357	2	1	2	1	90	46	90	46
9ab79e3cf393ae7506653cf2e485268a_1593688609	2	1488	1252	2	1	2	1	90	46	90	46
462f4c4aebb2a55141264a4a52d54144_1593685252	2	3258	2349	2	1	2	1	90	46	90	46
0056d6b61f45430102109b1e241dbdc8_1589902117	2	1907	1577	2	1	2	1	90	46	90	46
783692b89d729bd25d268e063028a93b_1593656486	0	0	0	0	0	0	0	0	0	0	0
456c81820c4bbbd795d238c28a7b5709_1589974130	2	12730	10172	2	1	2	1	90	46	90	46
9484c862cb0e7915fd2a1c0e36d7a917_1590163775	0	0	0	0	0	0	0	0	0	0	0

Hình 3.7. Bộ đặc trưng dữ liệu luồng mạng chuẩn hóa theo mô tả đặc trưng của bộ dữ liệu mạng CSE-CIC-IDS2018

name	num_total	num_total	num_total	num_total	num_total	num_total	num_total	num_total	num_total	num_total	num_total	num_total	num_total	num_total	num_total	num_total	cpu_%_us	cpu_%_us	cpu_%_us	cpu_%_us
06916bd06712bc6652ee547825	2.444444	1.342561	5	1	82.88889	1.099944	86	82	0	0	0	0	0	0	0	0	26.18889	1.099439	27.9	24.3
ca90ad8fefe28c724c2736984d6	2.5	1.5	4	1	56.5	0.5	57	56	0.5	0.5	1	0	0	0	0	0	59.65	0.15	59.8	59.5
017662f941ebe50c7e0d3fd9a8	2.5	1.5	4	1	56.5	0.5	57	56	0	0	0	0	0	0	0	0	59.2	0.1	59.3	59.1
9a983419ddc7ab33507b41e368	3	2.160247	6	1	56.66667	0.942809	58	56	0	0	0	0	0	0	0	0	59.06667	0.20548	59.3	58.8
888ccce705a9450caef48e6cf79	3	2	5	1	50	0	50	50	0.5	0.5	1	0	0	0	0	0	59.2	0.1	59.3	59.1
740cea9b9bdbb761c49270e72f	2.533333	1.203698	5	1	52.93333	1.236482	54	50	0	0	0	0	0	0	0	0	59.96	0.54626	60.7	59
f1085b90bdd41b23bb18e9a457	2.5	1.5	4	1	50.5	0.5	51	50	0.5	0.5	1	0	0	0	0	0	58.4	0.1	58.5	58.3
ade13c6d0182c53febc9b84862	3	2	5	1	51.5	1.5	53	50	0	0	0	0	0	0	0	0	59.4	0.1	59.5	59.3
7535ff0d990b6cfc8d20dc06f3a	2.666667	0.942809	4	2	57	1.414214	59	56	0	0	0	0	0	0	0	0	60	0.244949	60.3	59.7
e4c17224fd574e62ad0e8ae2a8	3.5	0.5	4	3	56	1	57	55	0.5	0.5	1	0	0	0	0	0	59.75	0.15	59.9	59.6
c5ce05ebee11ea7c06dc3e4bdd	2	1	3	1	50.5	0.5	51	50	0.5	0.5	1	0	0	0	0	0	59.05	0.05	59.1	59
8ca7ec7b22b8bc8fc436672a62	2.75	0.829156	4	2	80.75	0.829156	82	80	0	0	0	0	0	0	0	0	24.675	0.258602	25	24.3
bf07a7a21d7ffa99bb8ab4035d	2	1	3	1	56.5	0.5	57	56	0	0	0	0	0	0	0	0	59.7	0.1	59.8	59.6
a553e97d88f7b7934276856f1a	2.5	0.5	3	2	56	1	57	55	0	0	0	0	0	0	0	0	59.7	0.1	59.8	59.6
91d9701615b7b179a1b7eb666	4	1	5	3	50.5	1.5	52	49	0	0	0	0	0	0	0	0	58.15	0.05	58.2	58.1
318c43200fc8c71c24ca592e2e	2.5	0.5	3	2	56.5	0.5	57	56	0	0	0	0	0	0	0	0	59.9	0.1	60	59.8
9ab79e3cf393ae7506653cf2e4	2.75	1.47902	5	1	57	1.224745	59	56	0	0	0	0	0	0	0	0	60.125	0.192029	60.4	59.9
462f4c4aebb2a55141264a4a52	2.666667	1.699673	5	1	50.66667	0.942809	52	50	0	0	0	0	0	0	0	0	58.06667	0.094281	58.2	58
0056d6b61f45430102109b1e24	3	2	5	1	57	1	58	56	0	0	0	0	0	0	0	0	59.85	0.15	60	59.7
783692b89d729bd25d268e063(	2	1	3	1	51	1	52	50	0.5	0.5	1	0	0	0	0	0	58.55	0.05	58.6	58.5
456c81820c4bbbd795d238c28a	2	1	3	1	51.5	1.5	53	50	0	0	0	0	0	0	0	0	58.8	0.1	58.9	58.7
9484c862cb0e7915fd2a1c0e36	3	2	5	1	57.5	1.5	59	56	0	0	0	0	0	0	0	0	52.75	0.05	52.8	52.7
a4d8e93df4f3b9aeff6c61593aa	2.5	1.5	4	1	50	0	50	50	0	0	0	0	0	0	0	0	58.8	0.1	58.9	58.7
52ee76e2bb81ea0dbac6ca2278	3	2	5	1	56.5	0.5	57	56	0.5	0.5	1	0	0	0	0	0	59.95	0.15	60.1	59.8
f308903a85b05334d5aefb7a64	2.5	1.5	4	1	50.5	0.5	51	50	0.5	0.5	1	0	0	0	0	0	58.75	0.05	58.8	58.7

Hình 3.8. Bộ đặc trưng dữ liệu sử dụng tài nguyên chuẩn hóa theo đầu ra của V-Sandbox

name	#0	#1	#2	#3	#4	#5	#6	#7	#8	#9	#10	#11	#12	#13	#14	#15	#16	#17	#18	#19
ca422e82a753cc77fd39359c	0.417392	-0.03792	0.614784	-0.10077	0.274377	0.03604	0.775788	0.163963	-0.07387	0.201733	0.231688	-0.02996	-0.19174	0.452063	0.362744	-0.33363	0.206744	0.406058	0.407741	0.028207
75b425d4c17e38c482adbb9	-0.14718	0.299331	0.463711	-0.30361	0.04424	0.075229	-0.02438	0.200255	-0.38953	-0.12514	0.010705	0.209225	-0.42184	-0.03995	0.353882	-0.07386	-0.0258	0.256283	-0.17373	-0.20575
f31857095e18b0ac8a7a92d	-0.00176	-0.00063	-0.00314	-0.00252	-0.00297	-0.00145	0.000528	0.003748	-0.00043	-0.00285	-0.00064	-0.00112	-0.00237	-0.00289	-0.0005	0.003219	-0.00168	0.00141	-0.00225	0.000605
b10365c2ed158f42e0d3042	0.022033	-0.01266	0.446192	-0.4163	0.635272	0.437435	0.385394	0.080333	-0.56905	-0.00684	0.172779	-0.27382	-0.06186	-0.28334	0.176178	0.029936	-0.16532	0.089836	-0.05862	-0.15821
245c9cc38b412e8f44d3b92f	-0.0962	0.433907	1.011162	-0.06523	-0.13495	0.241593	-0.15654	0.367647	-0.10447	-0.03408	0.226747	0.08468	-0.53062	-0.46443	0.395556	-0.30085	-0.12144	0.122074	0.04191	0.204901
d5728c795024c3b222e47fa	-0.28907	0.154219	0.415879	-0.30724	0.029628	0.362739	-0.0734	-0.29989	-0.47756	-0.025	0.09332	0.12423	-0.1446	0.112502	0.43597	0.005839	-0.31836	0.18118	0.051933	-0.04126
5d9dbea9e5282a888efc534	-0.01318	0.353012	0.607745	-0.17214	-0.56273	-0.56886	-0.05968	-0.27012	-0.08558	-0.42347	-0.41621	0.161724	-0.23135	-0.18187	0.657714	-0.20444	-0.12053	-0.03243	-0.14843	-0.54647
b71664df5f5e98b55bb9456	0.081088	1.337151	0.361997	-1.40365	0.075783	0.212316	-0.36495	0.543099	-0.41076	-0.08299	-0.18544	-0.45737	0.458539	0.219077	1.182909	0.083035	-0.07689	0.784917	0.143437	-0.58269
d03fb405ac5bbdd7e9408b2	-0.25357	0.193825	0.375461	-0.30972	-0.01456	0.353501	-0.08779	-0.28084	-0.41776	0.012673	0.140692	0.159382	-0.20381	0.097105	0.406815	-0.01846	-0.28706	0.191409	-0.01246	-0.069
9ad8153341fd7ec5ceff4e66	0.225042	0.519073	0.769218	0.32508	0.162861	0.437172	0.027757	-0.11979	-0.50686	0.203949	-0.11293	0.18567	-0.37332	0.106478	0.635276	-0.24322	-0.16033	0.328741	-0.22825	-0.16233
f673d8017239bc44665dd34	0.255746	0.237762	0.613032	-0.1865	0.587081	0.205936	0.183963	-0.28986	-0.09969	0.029589	-0.03962	0.271528	-0.05857	-0.12939	0.369428	-0.27796	-0.04009	0.460018	-0.35985	0.224411
ea9eb02384e7c682959baa2	-0.14334	0.351151	0.474507	-0.35802	0.055298	0.110998	-0.06229	0.12844	-0.42126	-0.10658	0.048058	0.204316	-0.42517	-0.07929	0.349108	-0.14733	-0.02902	0.300404	-0.16249	-0.19764
9607542fb3a1c70d5687378	0.034741	0.028888	0.526335	-0.21137	-0.0243	-0.23027	0.145494	0.162151	0.007353	0.249399	0.347631	0.261706	-0.28469	-0.05473	0.662704	-0.15386	-0.17961	0.205242	0.06738	-0.25302
5b7b5ff335d4886f2dec647a	-0.10835	0.090265	0.854913	-0.68474	0.007111	-0.02896	-0.19689	-0.05032	-0.20575	-0.58587	-0.21166	-0.01845	-0.04184	-0.00576	0.800238	-0.12428	-0.17744	0.230784	-0.10066	-0.30897
38f4b61c190969bd54a7b47	0.03743	0.233955	0.697677	-0.29646	-0.33348	0.125354	-0.34608	0.20325	-0.37535	-0.24818	-0.36015	0.169096	-0.13553	-0.06391	0.58914	0.104786	-0.29931	0.634703	-0.33271	-0.40689
e5727dec18a8c103a5738f2	-0.04429	0.022743	0.416712	-0.46799	0.625582	0.416693	0.347489	0.111736	-0.54602	0.00772	0.178264	-0.34786	-0.08633	-0.26232	0.171659	0.012985	-0.18913	0.057902	-0.043	-0.10319
c208f7d28574298f31a60c53	0.278494	0.091782	0.434803	-0.22934	-0.12764	-0.5359	0.756806	0.038701	-0.33608	0.017152	0.32207	-0.08773	-0.29326	-0.07098	0.349737	-0.13453	0.015086	0.42394	0.206366	-0.28545
74d9494e23ca1a69e0128a2	0.329144	0.2971	0.207542	-0.51156	0.212242	0.089371	0.061098	0.337489	-0.24993	0.063206	0.365313	0.169767	-0.28018	0.006353	0.665	0.092555	-0.10749	0.052168	0.158571	-0.08997
0c59360fca2ad71ff3ae71dc	0.042912	0.302755	0.973488	-0.20227	-0.02839	0.154709	-0.36625	0.14501	-0.06943	-0.28075	-0.19212	-0.03649	-0.34802	0.461677	0.786784	-0.36884	-0.06095	0.320535	0.030281	0.003158
e673d98a5b9267b2b71a521	-0.26521	0.017313	0.630079	-0.35302	0.179435	0.290078	-0.24522	0.321574	-0.39175	0.048162	-0.22297	0.322941	-0.01256	0.094246	0.859197	-0.29977	-0.12879	0.228327	0.33009	-0.02469
698221d41cc41273e535a15	-0.17145	0.351307	0.398695	-0.3529	0.081661	0.092789	-0.07096	0.195571	-0.44406	-0.12535	-0.0356	0.158482	-0.36695	-0.08849	0.371836	-0.09209	-0.09042	0.290106	-0.11213	-0.18462
6991b61d6b8ac639530e432	-0.3309	0.406627	0.854847	-0.34289	-0.21119	0.175671	-0.18708	-0.03287	-0.10974	-0.43092	-0.34616	0.048189	-0.15165	-0.1064	0.990737	-0.01803	-0.11722	0.277402	-0.0959	-0.54681
04ca6eb03cd704266f5518b	-0.00161	-0.00353	-0.00072	0.001656	0.000171	-0.00065	-0.00378	0.001249	-0.00025	-0.00066	-0.00094	-0.00011	0.002951	0.002875	0.003075	0.002832	-0.0006	0.002958	0.003743	-0.00051
d4e146cac920b23578c1d16	-0.21588	0.210603	0.494665	-0.29457	0.155683	0.207889	0.054948	0.048882	-0.46316	-0.10894	0.014314	0.142726	-0.28739	-0.19347	0.254745	-0.08465	-0.06045	0.141387	0.037188	-0.09083
6f77993cbe5a2f762d458d53	0.289063	0.305564	0.268342	-0.04281	0.349053	0.092081	0.381969	0.263415	-0.19793	0.24906	-0.16273	0.947429	-0.28368	0.269315	0.61912	-0.10736	0.192682	0.37383	-0.38285	0.04861
147af70b815093d9247e22f6	0.014795	0.553409	0.16702	-0.30323	0.415719	-0.04275	0.178464	-0.04967	-0.09156	0.086259	-0.01112	-0.13627	0.266683	-0.17136	0.696277	-0.06352	0.046174	0.185606	-0.24118	-0.12011

Hình 3.9. Bộ đặc trưng dữ liệu lời gọi hệ thống trích xuất các đặc trưng từ đồ thị SCG thành vector đặc trưng

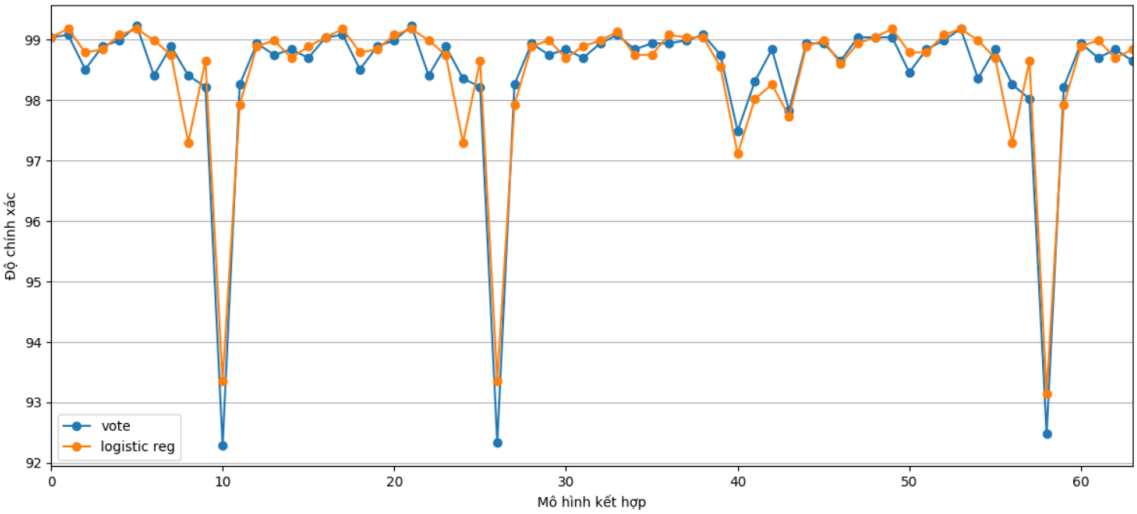
3.2. Môi trường triển khai thực nghiệm

Quá trình triển khai thực nghiệm để đánh giá mô hình đề xuất được tiến hành trên máy chủ với thông số kỹ thuật cơ bản gồm: Bộ vi xử lý AMD Ryzen 3.6 GHz, ổ cứng lưu trữ HDD dung lượng 1TB, bộ nhớ DDR3 32 GB. Luận văn đã tìm hiểu và sử dụng các thuật toán học máy thường được sử dụng hiện nay cho bài toán phát hiện mã độc như Bagging, ADABoost, Random Forest, GradientBoosting và các hàm hợp nhất khác nhau như biểu quyết hay hồi quy tuyến tính để chọn được phương pháp tối ưu nhất. Sử dụng các bộ phân loại hợp nhất tăng khả năng tổng hợp kết quả do các bộ phân loại này cũng xây dựng trên các thành phần học yếu hơn để đưa ra kết quả. Các thuật toán học máy đơn lẻ thử nghiệm Bagging, Random Forest, ADABoost, GradientBoosting,… được cài đặt thông qua ngôn ngữ Python với thư viện Scikit-learn (Sklearn).

3.3. Kết quả thực nghiệm

Luận văn đã sử dụng các bộ phân loại tổng hợp như Bagging, Random Forest, ADABoost, GradientBoosting tạo được tổ hợp 64 cách kết hợp 4 thuật toán này và hàm hợp nhất bầu chọn, hồi quy tuyến tính để đánh giá hiệu quả của mô hình học máy cộng tác. Từ các cách kết hợp các bộ phân loại phổ biến cùng với sử dụng 2 phương pháp hợp nhất thu được 128 kết quả thử nghiệm đánh giá độ chính xác của việc phát hiện mã độc IoT Botnet cho mô hình ứng dụng được mô tả trong Hình 3.10 và Bảng 3.2.

Dựa vào kết quả thu được mô tả trong hình, ta có thể nhận thấy hàm hợp nhất bầu chọn có hiệu suất tốt tương đương với hàm hồi quy tuyến tính và bộ ba thuật toán phân loại học máy đơn lẻ sử dụng random forest trên 3 tập dữ liệu thuộc tính khác nhau cho kết quả tốt nhất với độ chính xác ACC = 99.23% sử dụng phương pháp bầu chọn. Các mô hình học máy đã được luận văn điều chỉnh các tham số trên toàn bộ dữ liệu và tiến hành đánh giá mô hình ứng dụng dựa trên các chỉ số thu được. Kết quả đánh giá mô hình ứng dụng trên bộ dữ liệu được trình bày trong bảng 3.2.



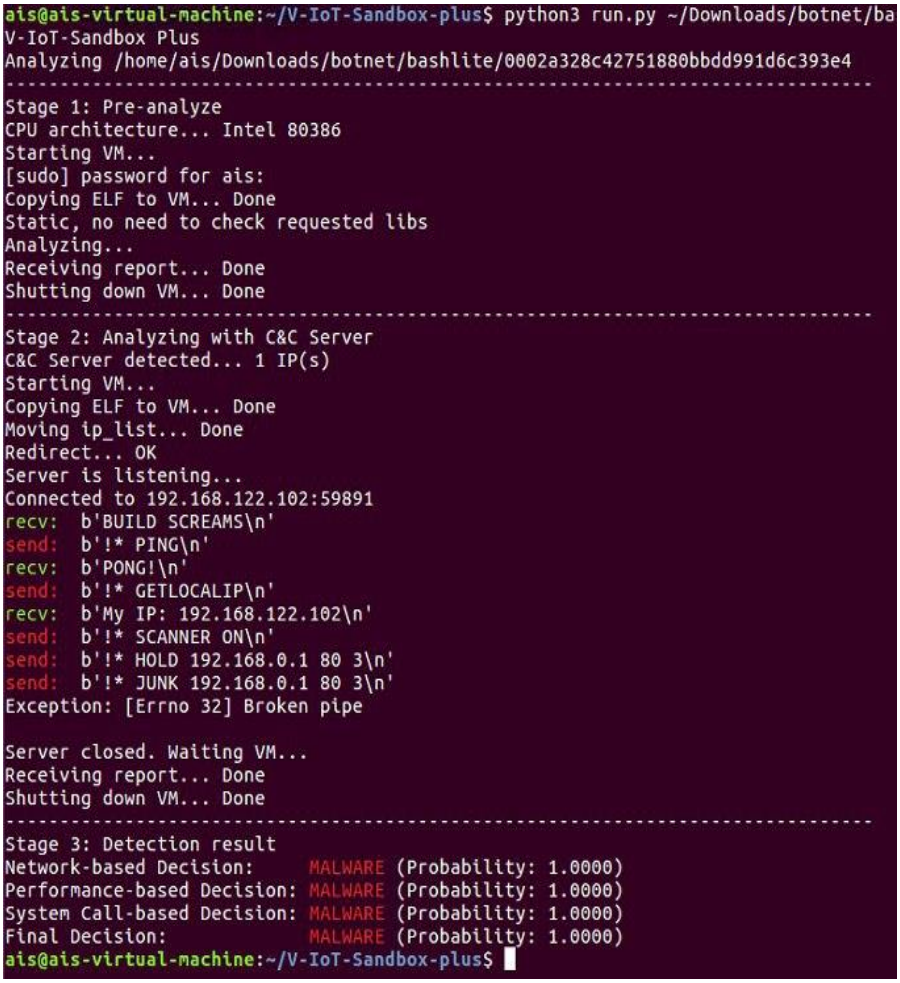
Hình 3.10. Kết quả đánh giá các phương án kết hợp thuật toán học máy

Bảng 3.2. Độ chính xác các bộ phân loại học máy đơn lẻ huấn luyện trên bộ dữ liệu và kết quả tổng hợp dự đoán

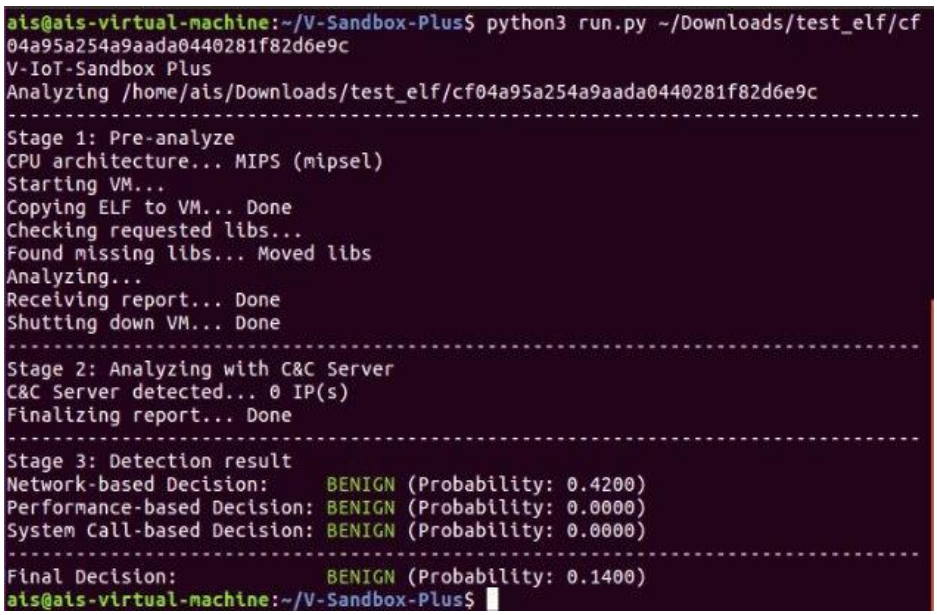
STT	Thuật toán	Độ chính xác phân loại dữ liệu của mô hình (ACC - %)				
		Dữ liệu luồng mạng	Dữ liệu sử dụng tài nguyên thiết bị	Dữ liệu lời gọi hệ thống	Bầu chọn	Hồi quy tuyến tính
1.	Bagging + Bagging + Bagging	89.97	98.84	97.2	99.04	99.04
2.	Bagging + Bagging + Random Forest	89.97	98.84	97.78	99.08	99.18
3.	Bagging + Bagging + AdaBoost	89.97	98.84	97.2	98.51	98.79
4.	……..					
5.	Random Forest + Random Forest + Bagging	90.07	99.08	97.2	98.99	99.08
6.	**Random Forest + Random Forest + Random Forest**	**90.07**	**99.08**	**97.78**	**99.23**	**99.18**
7.	Random Forest + Random Forest + AdaBoost	90.07	99.08	97.2	98.41	98.99
8.	Random Forest + Random Forest + Gradient Tree Boosting	90.07	99.08	97.69	98.89	98.75

STT	Thuật toán	Độ chính xác phân loại dữ liệu của mô hình (ACC - %)				
		Dữ liệu luồng mạng	Dữ liệu sử dụng tài nguyên thiết bị	Dữ liệu lời gọi hệ thống	Bầu chọn	Hồi quy tuyến tính
9.	Random Forest + AdaBoost + Bagging	90.07	98.65	97.2	98.36	97.3
10.	……					

Mô hình sau khi được huấn luyện xong sẽ được đưa vào vận hành thử trong hệ thống; luận văn lựa chọn thời gian thu thập dữ liệu theo thời gian thực là 03 giây (đảm bảo để có thể thu được 300 lời gọi hệ thống, 20 trạng thái tài nguyên thiết bị và 50 gói tin luồng mạng như đã phân tích) cho quá trình thực thi tệp đầu vào và đưa ra kết quả dự đoán phân loại tệp tin. Kết quả chạy thực tế được minh họa như hình 3.11 và 3.12:



Hình 3.11. Quá trình phân tích tệp chứa mã độc



Hình 3.12. Quá trình phân tích tệp lành tính

3.4. Đánh giá kết quả thực nghiệm

Với kết quả được nêu ra ở phần trên có thể đánh giá mô hình ứng dụng có khả năng phát hiện chính xác với *ACC = 99.23%*. Kết quả thực nghiệm này cho thấy hiệu quả của việc sử dụng mô hình học máy cộng tác cho 3 loại dữ liệu hành vi phổ biến trong phát hiện IoT Botnet. Sử dụng Information Gain để trích chọn đặc trưng, cách kết hợp các bộ phân lớp sử dụng thuật toán học máy với nhau trong một mô hình cộng tác đã góp phần làm tăng hiệu quả phát hiện của mô hình. Mô hình kết hợp đã cho kết quả phát hiện vượt trội hơn các mô hình học máy đơn lẻ. Khả năng phát hiện sớm của mô hình cũng được thể hiện ở đặc điểm chỉ lấy một phần nhỏ lượng dữ liệu đặc trưng cho các hành vi đầu tiên của tệp đầu vào đang được xử lý để thực hiện phân tích và phát hiện mã độc thay vì phải đợi mã độc thực hiện đầy đủ hành vi để thu thập và xử lý với toàn bộ dữ liệu.

Kết luận chương

Trong chương này, luận văn đã trình bày kết quả thực nghiệm triển khai mô hình ứng dụng được mô tả trong chương 2 với bộ dữ liệu và nền tảng phần cứng, phần mềm đi kèm. Kết quả thử nghiệm đã cho thấy hiệu quả vượt trội của mô hình ứng dụng và khả năng ứng dụng vào giải quyết bài toán phát hiện mã độc IoT Botnet trong thực tế.

KẾT LUẬN VÀ KIẾN NGHỊ

Cuộc cách mạng 4.0 và sự bùng nổ phát triển về công nghệ, khoa học kỹ thuật những năm vừa qua đã mang lại rất nhiều thay đổi lớn với cuộc sống nhân loại. Trong cuộc phát triển đó, xu hướng Vạn vật kết nối IoT nổi một cách mạnh mẽ và trở thành một phần không thể thiếu trong không gian số của mỗi cá nhân, tổ chức. Các thiết bị IoT đã, đang và sẽ tiếp tục được sử dụng phổ biến tại các tổ chức, doanh nghiệp ở nhiều quốc gia trên thế giới, trong đó có cả Việt Nam. Số lượng thiết bị IoT ngày càng tăng, tỉ lệ thuận với đó là số lượng mã độc, các cuộc tấn công khai thác đem lại thách thức rất lớn đối với việc bảo đảm an ninh, an toàn thông tin. Song song với việc phát triển và mở rộng rất nhanh về số lượng thì các nhà phát triển thiết bị IoT lại không có sự quan tâm đến vấn đề bảo mật khiến những thiết bị này trở thành mục tiêu dễ dàng cho các hành vi tấn công, khai thác. Do đó việc nghiên cứu, phát triển các hình thức bảo vệ các thiết bị IoT là hoàn toàn cần thiết, góp phần đảm bảo an ninh, an toàn thông tin trên môi trường mạng.

Trong phạm vi nghiên cứu về phân tích, phát hiện mã độc Botnet trên các thiết bị IoT, luận văn đã tiến hành tìm hiểu các phương pháp phân tích, phát hiện mã độc Botnet trên thiết bị IoT phổ biến hiện nay và sau đó xây dựng, ứng dụng thực nghiệm mô hình học máy cộng tác trong phân tích, phát hiện mã độc IoT Botnet. Cụ thể, luận văn đã đạt được một số kết quả như sau:

- Nghiên cứu, lựa chọn, ứng dụng và xây dựng thử nghiệm thành công mô hình học máy cộng tác trong phân tích và phát hiện mã độc IoT Botnet.

- Thử nghiệm phát hiện mã độc với các mô hình học máy đơn lẻ và so sánh đánh giá với mô hình học máy đã xây dựng. Kết quả cho thấy hiệu suất phát hiện được cải thiện vượt trội so với việc sử dụng các bộ học máy đơn lẻ.

- Đem lại khả năng ứng dụng trong thực tế khi mô hình cho ra kết quả phát hiện trong thời gian ngắn và chỉ yêu cầu lượng dữ liệu đầu vào nhỏ ngay khi mã độc bắt đầu thực hiện hành vi. Do đó giảm thiểu được hậu quả của mã độc gây ra với thiết bị và hệ thống thông tin.

Kết quả của luận văn góp phần bổ sung vào các nghiên cứu phát hiện mã độc IoT Botnet dựa trên phương pháp phân tích động và tiềm năng ứng dụng cao. Một số nội dung nghiên cứu trong luận văn cũng được chấp nhận công bố trên Kỷ yếu hội nghị quốc tế lần thứ 3 về Điện tử, truyền thông và khoa học máy tính (ICECCE 2021) với bài báo *"Adversarial Attack and Defense on Graph-based IoT Botnet Detection Approach"*.

Tuy nhiên, luận văn vẫn còn một số hạn chế, vướng mắc ở phần xử lý sandbox vì không gian tài nguyên yêu cầu lớn, thời gian khởi động chậm. Ngoài ra khi chạy V-Sandbox để thu thập hành vi dữ liệu của một số mẫu thì có thể xảy ra việc thực hiện vòng lặp để thu thập thêm dữ liệu cho mỗi lần chạy khiến cho thời gian xử lý 1 mẫu lên đến 3 phút.

Dựa trên kết quả nghiên cứu, luận văn đưa ra một số kiến nghị cho các hướng phát triển trong tương lai như sau:

- Tiếp tục nghiên cứu, thử nghiệm và cải thiện phương pháp kết hợp các đặc trưng và bộ phân loại nhằm cải thiện độ chính xác và thời gian xử lý của phương pháp phát hiện sớm mã độc IoT Botnet.

- Nghiên cứu, cải tiến để rút ngắn thời gian hoạt động của V-Sandbox và lượng tài nguyên yêu cầu để làm cho mô hình phát hiện sớm IoT Botnet hoạt động hiệu quả hơn.