**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**

VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

**AN TOÀN THÔNG TIN**

---------------o0o---------------

****

**Báo cáo đồ án môn học**

**Ứng dụng học máy vào bài toán nhận diện mã độc JavaScript**

**SV: Trần Đức Sơn 20163571**

**GVHD: Nguyễn Đức Toàn**

**HÀ NỘI, 4/2020**

Mục lục

[Đặt vấn đề 2](file:///H:\DoAnTotNghiep\DAMH\docs\Doan_baocao_RandomForests.docx#_Toc38806399)

[Phân tích hành vi JavaScript malware 3](file:///H:\DoAnTotNghiep\DAMH\docs\Doan_baocao_RandomForests.docx#_Toc38806400)

[Các hành vi cụ thể 3](file:///H:\DoAnTotNghiep\DAMH\docs\Doan_baocao_RandomForests.docx#_Toc38806401)

[Lý thuyết cơ bản 7](file:///H:\DoAnTotNghiep\DAMH\docs\Doan_baocao_RandomForests.docx#_Toc38806402)

[1. Xử lý dữ liệu 7](file:///H:\DoAnTotNghiep\DAMH\docs\Doan_baocao_RandomForests.docx#_Toc38806403)

[2. Thuật toán Machine Learning 7](file:///H:\DoAnTotNghiep\DAMH\docs\Doan_baocao_RandomForests.docx#_Toc38806404)

[Giải pháp đề xuất 11](file:///H:\DoAnTotNghiep\DAMH\docs\Doan_baocao_RandomForests.docx#_Toc38806405)

[Công nghệ sử dụng 11](file:///H:\DoAnTotNghiep\DAMH\docs\Doan_baocao_RandomForests.docx#_Toc38806406)

[Triển khai 14](file:///H:\DoAnTotNghiep\DAMH\docs\Doan_baocao_RandomForests.docx#_Toc38806407)

[Kết luận 22](file:///H:\DoAnTotNghiep\DAMH\docs\Doan_baocao_RandomForests.docx#_Toc38806408)

[Tài liệu tham khảo 22](file:///H:\DoAnTotNghiep\DAMH\docs\Doan_baocao_RandomForests.docx#_Toc38806409)

# Đặt vấn đề

Công nghệ web hiện nay đang chiếm vị trí quan trọng trong tổng thể ngành công nghệ thông tin. Ở bất cứ đâu con người đều có thể truy cập vào web thông qua các ứng dụng web, làm mọi thứ từ truy cập thông tin đến giải trí, học tập và làm việc. Công cụ đóng vai trò quan trọng, giúp chúng ta tương tác với Web và Internet là Trình duyệt (browser). Ở bất cứ hệ điều hành nào, ta đều bắt gặp ít nhất một browser được cài sẵn giúp người dùng thuận tisện khi muốn truy cập vào website.

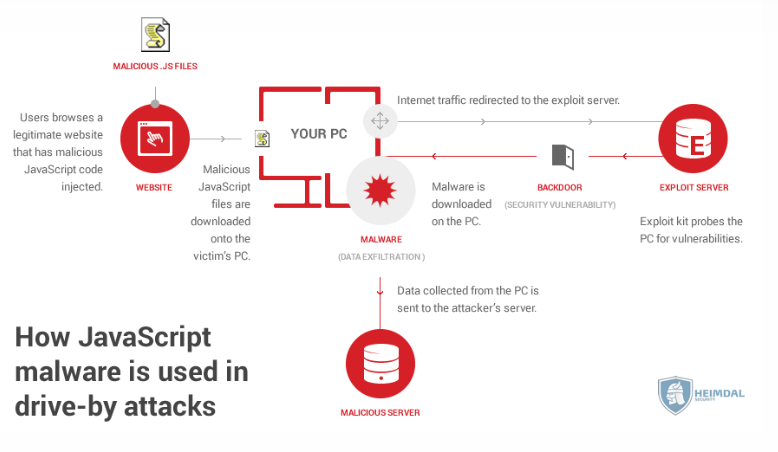
Tuy nhiên, song song với sự phát triển, đi lên của trình duyệt internet thì những hình thức tấn công vào máy tính người dùng thông qua nền tảng này cũng được phát triển, mở rộng và đang là mối nguy lớn. Chủ yếu, nó được lợi dụng bằng cách cài đặt một số những mã nguồn độc hại bằng JavaScript nhằm thực thi chuỗi lệnh từ đó thực hiện hành vi độc hại mà kẻ tấn công đã lên kịch bản trước.

# Phân tích hành vi JavaScript malware

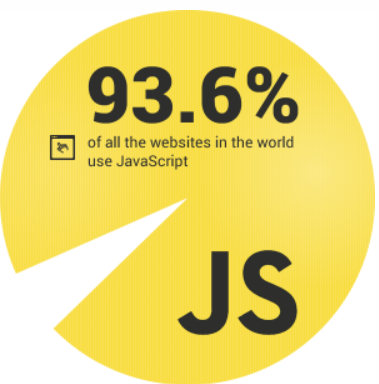
## Giới thiệu chung

JavaScript là đoạn mã cho phép người tạo ra website chạy khi một người dùng truy cập vào trang web.

Những người tạo ra những website này có thể là những nhà phát triển bình thường hoặc cũng có thể là những kẻ hacker mũ đen có mục đích xấu, thao túng code javascript và khiến nó chạy những chức năng độc hại



Hiện nay, JavaScript được sử dụng bởi 93.6% trên tổng số trang web hiện nay (theo số liệu của Heimdal Security). Các cuộc tấn công lợi dụng javascript nhắm vào dữ liệu người dùng hoặc lợi dụng máy tính người dùng thực hiện những hành vi trái phép



Mô hình tấn công cơ bản có thể là: Chèn các đoạn mã độc hại vào những trang web hợp lệ, sử dụng chúng để chuyển hướng đến các trang web độc hại. Người dùng là các nạn nhân không biết chuyện gì đang xảy ra do quá trình này là tự động. Tại trang web độc hại, kẻ tấn công có thể khai thác mọi thứ từ người dùng từ tài khoản, thông tin thẻ tín dụng, hoặc lợi dụng máy tính nạn nhân như một botnet để tấn công các máy tính khác. Các cuộc tấn công kiểu này có thể diễn ra trên hàng triệu máy tính khác nhau trong cùng một thời điểm, do vậy qui mô và thiệt hại của hình thức tấn công này vô cùng lớn.

## Các hành vi cụ thể

Có 8 con đường chính lây lan javascript malware trong một cuộc tấn công mạng

* Mã độc javascript được chèn vào trong các website hợp pháp – thường được sử dụng để chuyển hướng người dùng tới một trang web chứa nội dung độc hại hoặc để kích hoạt, khai thác các server khởi động cuộc tấn công lây lan mã độc

Một lỗ hổng XSS (Cross-site scripting) trong Yahoo Mail đã ảnh hưởng đến hơn 300 triệu tài khoản email trên toàn cầu được vá, người phát hiện ra lỗi này được thưởng 10000 đô. Lỗ hổng cho phép mã JavaScript độc hại được nhúng trong thông báo email được định dạng đặc biệt. Mã được tự động chạy khi email được xem. JavaScript có thể được sử dụng để thỏa hiệp với tài khoản, thay đổi cài đặt của nó, hoặc gửi mail tới người khác mà không có sự đồng ý của chủ tài khoản (Nguồn InfoSecurity Magazine)

* Ẩn Iframes: Load các đoạn mã độc javascript từ một trang web thỏa hiệp, mã độc sẽ cố gắng thực thi code trên trình duyệt nạn nhân và lây lan vào máy tính
* Mã độc JavaScript lây lan trong các mạng quảng cáo trực tuyến: xuất hiện trong các banner quảng cáo onliene và cũng thường âm thầm chuyến huowgs người dùng tới địa chỉ trang web mã độc
* Drive-by downloads: được sử dụng ddeere lây lan những file javascript chạy mã độc
* Loại mã độc javascript được đóng gói: loại này chạy thông qua một chương trình trên hệ điều hành windows và có thể kích hoạt lây lan bên ngoài trình duyệt

Theo nguồn Naked Security Blog của Sophos: “Khi một file .js được lưu trong ổ đĩa, hệ điều hành windows sẽ chạy nó bằng phần mềm mặc định ngoài trình duyệt, gọi là WSH (Windows Script Host). Một chương trình hje thống chuẩn gọi là WScript.exe (một commandline scripts không có giao diện đồ họa) sẽ load đoạn script, chuyển nó cho WSH và chạy như một chương trình thực thi thông thường”

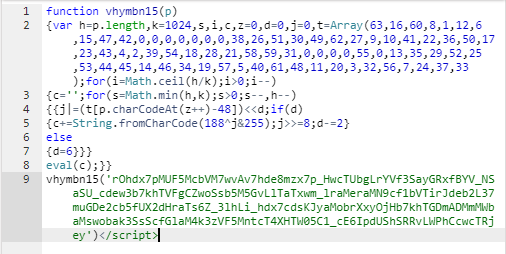
* Duyệt các add-ons và plugins: chúng có thể lây nhiễm hoặc load những nội dung mã độc từ nguồn bên ngoài
* Giả dạng những thông điệp popup của phần mềm: Có thể giả mạo sao cho giống y như thật

1. **Chèn thuộc tính ẩn vào document**

Những mẫu JavaScript loại này được cài đặt trong những trang web không được xác thực, không an toàn hay sử dụng giao thức HTTP để giao tiếp. Nó có cấu trúc như một đoạn mã JavaScript thông thường nhưng thay vì thực hiện theo mục đích của lập trình viên website thì nó lại mang những hành động xấu. Cụ thể là chèn trực tiếp vào document HTML một thẻ <iframe> với attribute display:none hoặc cài đặt attribute.width, attribute.height tương đối nhỏ, làm cho người dùng không nhận ra.

Mục đích của hành vi này là đánh lừa người dùng click vào đối tượng, hoặc tương tác với đối tượng mà không chủ ý. Dẫn đến sinh lợi cho attacker (thông quan chạy quảng cáo) hoặc thực hiện những hành vi nguy hiểm hơn (chuyển hướng đến trang web độc hại hay tải mã độc về máy tính người dùng), …

Ví dụ một JavaScript malware chèn thẻ iframe ẩn vào document được lấy mẫu trên VirusTotal:



Hình minh họa: 1 Một mẫu JavaScript chèn thẻ iframe ẩn

Đoạn code sau khi giải mã (decode) là một câu lệnh document.write để ghi một thẻ iframe ẩn vào trong file HTML:



Hình minh họa: 2 Mã nguồn mẫu này sau khi giải mã

1. **Gọi hàm ẩn đối tượng trong document**

Những hàm ẩn này cũng có tác dụng tương tự như set lại thuộc tính ẩn. Tuy nhiên, do khả năng tương tác với mọi đối tượng của document nên có thể sử dụng để tương tác với tất cả các đối tượng trong document. Từ đó dễ dàng đặt lại thuộc tính cho đối tượng hơn.

1. **Thực hiện các câu lệnh jQuery**

Những câu lệnh JQuery có tính tương tác với các đối tượng trong document HTML, tác động đến nội dung file HTML, cấu trúc DOM, có thể lựa chọn những đối tượng trong DOM, trong CSS, hỗ trợ hiệu ứng, hoạt cảnh trong trang web, các phương thức HTML, chuyển đổi JSON, gọi hàm AJAX, các plug-ins, …

Những đoạn mã độc giả mạo những câu lệnh jQuery có tác động trực tiếp đến trang web mà người dùng truy cập đến, thường được inject vào các framework phổ biến để tạo trang web hiện nay như Wordpress, Joomla. Dưới đây là minh họa mã javascript JQuery trong các website Joomla và Wordpress



Hình minh họa: 3 Một mẫu giả dạng Jquery

Thường thì địa chỉ của những loại virus này thường có dạng:



Hình minh họa: 4 Đường dẫn trong biến base ở mẫu trên trỏ đến link mã độc JavaScript

Một đặc điểm nữa của những loại giả mạo Jquery là thường sẽ không sử dụng kèm kỹ thuật Obfuscated giống như những loại chèn thuộc tính ẩn hay chuyển hướng do mục đích là tối đa giả mạo câu lệnh JQuery và đánh lừa những chuyên viên phân tích mã độc.

1. **Chuyển hướng trang web hiện tại**

Một mã độc JavaScript chuyển hướng được chèn vào một website với mục đích chuyển hướng người dùng tới một website khác. Nó thường được tạo ra bởi các attacker nhằm đáp ứng một nội dung quảng cáo. Tuy nhiên, một vài mã độc chuyển hướng kiểu này có tác động nhiều hơn tới máy tính người dùng. Nó exploit (khai thác) một lỗ hổng trong máy tính của người dùng đang truy cập vào website, thông qua đoạn mã để cài những loại mã độc mà không được bảo vệ ở phía người dùng, hoặc lợi dụng để cài đặt mã độc cùng loại chuyển hướng mà rất khó có thể loại bỏ trực tiếp khỏi máy người dùng.

Các phương thức tấn công của loại này:

* Chèn vào file trong trang web người dùng: ví dụ trong WordPress, nó được chèn vào các theme file. Thường được tìm thấy trong các theme header
* Chèn vào trang hoặc post
* Chèn vào các Widgets
* Chèn vào các file .htaccess dùng để cấu hình máy chủ web Apache, hoặc dùng quy định đường dẫn tới server mà tập tin của trang web được kết nối tới
* Ad networks: các mạng quảng cáo được chấp nhận trong mạng, kẻ tấn công có thể lợi dụng nó để giả mạo hoặc chèn vào mạng một loại quảng cáo độc hại do hệ thống không có (hoặc ít) cơ chế để kiểm tra và quét.

1. **Mã hóa file mã nguồn js hoặc sử dụng kỹ thuật Obfuscated javascript**

Với mục đích ẩn giấu mã nguồn, không cho những người không có kỹ năng về phân tích mã nguồn có thể đọc được, kỹ thuật obfuscated ra đời. Tuy nhiên, vì đặc tính che dấu mã nguồn này nên đa số các javascript malware lợi dụng để che dấu mã nguồn thật đằng sau một chuỗi mã hóa, khiến cho những người làm về phân tích mã nguồn không thể trực tiếp tìm ra được đoạn code mã độc

Các dạng Obfuscator trong javascript:

* **Randomization Obfuscation**: Ở dạng này, một số các element được chèn vào (ví dụ comment, những câu lệnh thực thi không liên quan, ..) mà không làm thay đổi ngữ nghĩa của đoạn mã để nhằm mục đích tạo nên sự ngẫu nhiên của các chữ cái trong mã nguồn được sử dụng, tính ngẫu nhiên trong comment và sử dụng các biến, đặt tên ngẫu nhiên
* **Data Obfuscation**: Ý tưởng chính là chuyển giá trị biến hoặc giá trị hằng thành kết quả có thể tính toán được của một hoặc nhiều biến khác. Có 2 kỹ thuật chính được sử dụng trong mục này là string splitting và keyword substitution. String splitting là chuyển đổi chuỗi thành kết quả của tổng nhiều chuỗi con. Keyword substitution là ánh xạ một biến đến một keyword từ đó thực thi bằng cách gọi biến đó như khi gọi đến keyword.
* **Encoding Obfuscation**: Có 3 cách để encode mã gốc: một là chuyển code thành ký tự ASCII, Unicode hoặc chuỗi hexadecimal được đã được escape. Phương thức thứ 2 là sử dụng hàm encode được định nghĩa, kẻ tấn công sử dụng hàm encoding này để tạo ra obfuscated code và khi thực thi thì sẽ gọi hàm decode trước, giải mã code sau đó mới triển khai. Ngoài ra, một vài chuẩn mã hóa và giải mã có thể được lợi dụng trong việc mã hóa code Javascript: ví dụ Jscript.Encode của Microsoft hỗ trợ mã hóa code JavaScript.
* **Logic Obfuscation**: Loại này lợi dụng luồng thực thi của JavaScript để thay đổi cấu trúc logic mà không ảnh hưởng đến ngữ nghĩa của code. Có 2 cách để thực hiện dạng này là chèn thêm một vài câu lệnh vớ vẩn hoặc không theo logic, không phụ thuộc vào hàm,. .. Một cách khác là thêm hoặc thay đổi một vài cấu trúc if…else, switch..case, for, while, …

Việc sử dụng kết hợp các phương pháp được nêu có thể gây nên khó khăn cho các nhà phân tích, điều này là một cách lý tưởng để kẻ tấn công thực hiện hành vi độc hại đồng thời cũng che dấu bản thân gọi sự phát hiện của hệ thống hoặc chuyên viên về an ninh an toàn.

Ví dụ về một dạng Obfuscator:



Hình minh họa: 5 Mã nguồn phức tạp sau khi đã được Obfuscator

1. **Theo dõi hành vi và vị trí con trỏ chuột trên trình duyệt**

Hành vi: Đoạn code JS tìm kiếm vị trí hiện tại của con trỏ chuột người dùng, sau đó tạo một vùng hoặc một thẻ tại ví trí đó, thẻ đó sẽ được tạo lại mỗi khi người dùng di chuyển con trỏ chuột

1. **Con đường lây lan**

Có 7 con đường lây lan chính cho Javascript malware

* Bị inject và các trang web hợp pháp
* Thông qua thẻ iframe ẩn
* Inject thông qua mạng lưới quảng cáo online
* Thông qua việc download bởi các drive độc hại: thường sử dụng file JavaScript để chạy malware infection
* Malicious JavaScript attachments: chạy thông qua chương trình windows và có thể kích hoạt lây lan bên ngoài trình duyệt
* Kích hoạt tải xuống thông qua việc inject một JavaScript độc hại khác
* Add-ons và plugins trên trình duyệt
* Giả mạo thông điệp pop-up của phần mềm

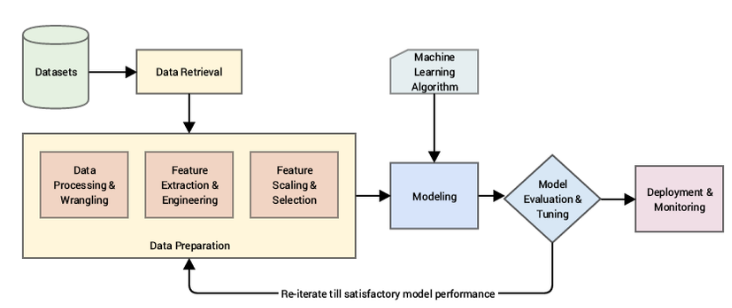
# Lý thuyết cơ bản

## Xử lý dữ liệu

Vai trò của xử lý dữ liệu:

Bất kỳ bài toán học máy nào đều phải trải qua các bước từ dữ liệu thô thu được, sử dụng các kỹ thuật để sắp xếp, xử lý, thiết kế các đặc trưng (feature) và thuộc tính có ý nghĩa từ dữ liệu này. Sau đó mới sử dụng các mô hình thống kê và học máy để xây dựng và giải quyết các yêu cầu đặt ra.

Mô hình xử lý chung cho các bài toán học máy dựa theo mô hình tiêu chuẩn công nghiệp CRISP-DM



Hình minh họa: 6 Mô hình cơ bản về quá trình làm một project học máy

Nếu chỉ giữ nguyên dữ liệu thô mà đưa luôn vào mô hình xử lý thì hiệu quả đạt được của mô hình sẽ không cao.

Với mỗi vấn đề cụ thể luôn có những đặc trưng quan trọng quyết định đến hiệu suất hệ thống. Feature Engineering luôn chiếm tới 70% thời gian trong quá trình xây dựng mô hình.

Xử lý dữ liệu hay cụ thể là trích chọn đặc trưng là quá trình biến đổi dữ liệu thành các đặc trưng đóng vai trò là đầu vào cho các mô hình học máy. Các đặc trưng xử lý tốt sẽ nâng cao hiệu suất mô hình. Tùy vào những dự án khác nhau mà những đặc trưng được lựa chọn cũng sẽ khác nhau.

Đối với bài toán JavaScript malware, đặc trưng được trích xuất dựa vào các hành động thực thi của mã nguồn, những đặc điểm thu được khi phân tích tĩnh mã nguồn và phân tích động hành vi thông qua quá trình giả lập và thực thi

## Mạng neural xử lý từ

# Giải pháp đề xuất

***Xây dựng mạng Neural CNN xử lý cú pháp được trích xuất và chuẩn hóa từ mã nguồn JavaScript***

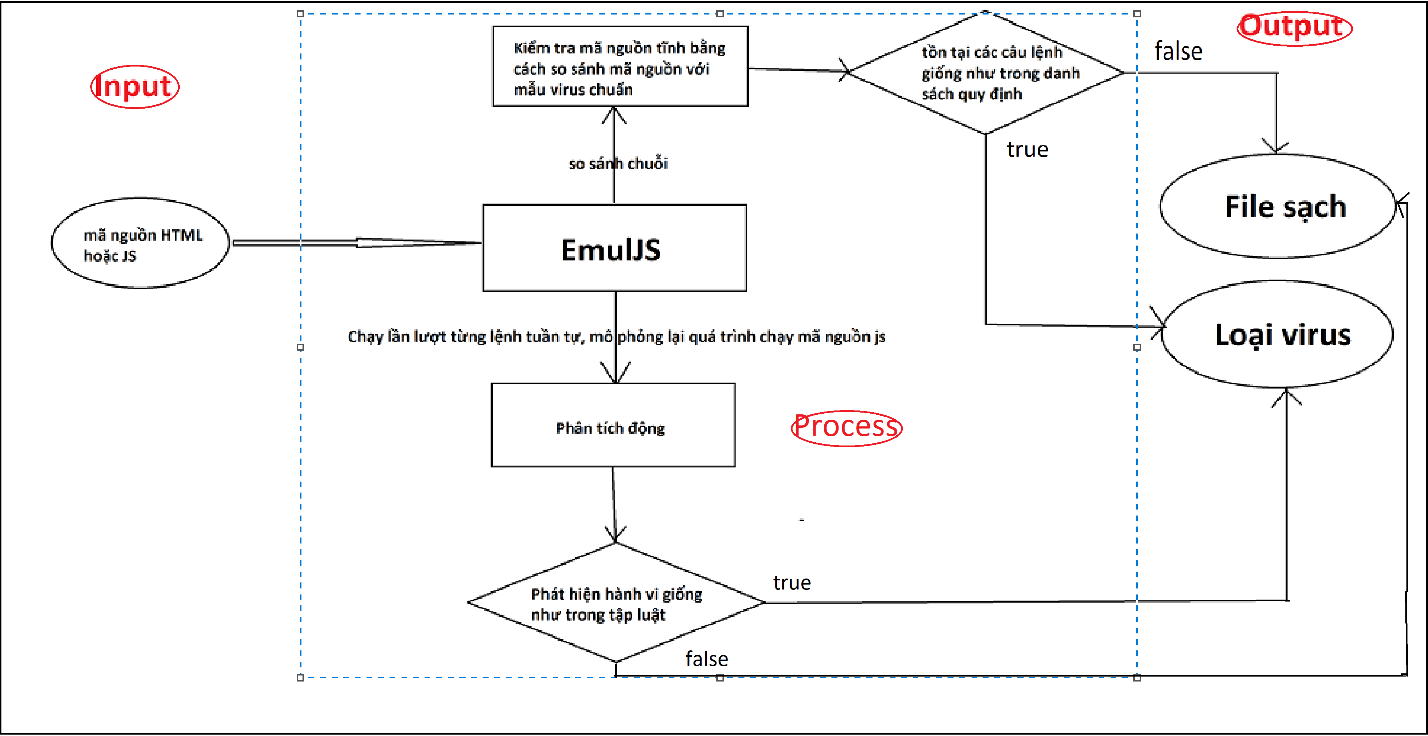
# Công nghệ sử dụng

1. **Giới thiệu về Engine EmulJS**

Engine EmulJS hiện đang được triển khai để nhận diện mẫu virus trên VirusTotal của Bkav. Công cụ này sử dụng kết hợp 2 phương pháp phát hiện code tĩnh và phân tích code động để tìm hành vi độc hại, từ đó sẽ trả về kết quả loại virus tùy theo hành vi phân tích được

Hiện tại em đang chỉnh sửa công cụ này để thay vì trả về kết quả virus như bình thường thì sẽ chỉ có tác dụng chạy động virus để thu thập những hành vi đặc trưng của virus từ đó lấy được dữ liệu output làm đầu vào cho mô hình học máy

Luồng xử lý của EmulJS:



Có 2 hướng đi cho chỉnh sửa công cụ này

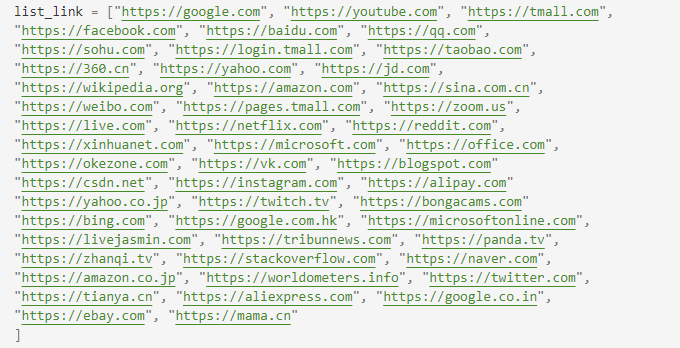
* Trả về những nội dung làm đầu vào cho học máy
* Xử lý và chuẩn hóa mã nguồn dạng text để làm đầu vào cho các mạng nơ ron.

Mã nguồn EmulJS được code lại, chỉ dữ lại luồng xử lý code chính, bỏ đi các luật phát hiện virus JavaScript, bỏ đi phần tính toán khi chạy động mã nguồn

1. **Giới thiệu về Tool Crawl mẫu HTML sạch**

Công cụ được viết bằng ngôn ngữ Python và sử dụng các thư viện làm việc với trang web và hỗ trợ crawl dữ liệu về như: urllib (download các trang web theo định dạng HTML), BeautifulSoup, time, os

Nguồn link ban đầu bao gồm:



Trong quá trình duyệt trang web, nếu thấy link Website khác với danh sách hoặc không nằm trong những link đã download thì sẽ thêm vào list để download tiếp. Tạo một list lưu danh sách những trang web đã crawl dữ liệu, danh sách sẽ tự động refresh lại về rỗng khi kích thước lớn hơn 2000 link. Có thể điều chỉnh kích thước nếu thực tế không hiệu quả.

Sau khi cho chạy lệnh này bằng python, mã nguồn HTML của các trang web được duyệt sẽ được lưu vào một thư mục làm đầu vào cho Tool Heuristic FS để trích xuất thông tin cần thiết ra file text

1. **Giới thiệu về công cụ Heuristic FS**

Công cụ sử dụng dll được tạo ra từ ***emulJS*** để quét offline một tập mẫu JavaScript đầu vào. Có thể tùy chỉnh đầu ra khi chạy công cụ bằng cách sửa mã nguồn của dll EmulJS

1. **Giới thiệu về nguồn lấy mẫu JavaScript và công cụ xử lý liên quan**

Nguồn lấy mẫu: VirusTotal

* VirusTotal là một dịch vụ miễn phí phân tích tập tin và URL nghi ngờ, nó có chứa một kho dữ liệu bao gồm rất nhiều mẫu JavaScript malware. Mỗi engine dùng để phân tích được hỗ trợ bởi cộng đồng các công cụ phân tích và diệt mã độc nổi tiếng trên thế giới. Mỗi một công ty cung cấp dịch vụ như vậy đều được cấp một tài khoản có thể truy cập và tải về kho tập mẫu của VirusTotal chứa nội dung Virus và một file log chứa thông tin các engine phát hiện.
* Các trang web được truy cập nhiều

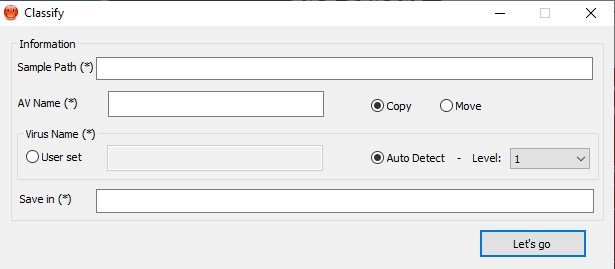
Cấu trúc thư mục chứa mẫu và log trả ra từ virus total

Nội dung log trả ra

Quy trình thu thập và phân loại mẫu:

* Sử dụng công cụ FileZila Client có cung cấp tài khoản, mật khẩu và địa chỉ host để lấy tập mẫu. Tập mẫu thu được sẽ bao gồm tổng hợp của file html + javascript + (có thể có) VBScript.
* Ở đây chỉ xét mã javascript do vậy sẽ loại bỏ đi những file chỉ thuần HTML và các file chứa mã VBScript. Để làm được điều này ta sử dụng công cụ Find VirusTotal Sample có chứa năng phân loại các mẫu Virus theo các từ khóa: ví dụ Microsoft thì sẽ phân loại theo tên virus được nhận diện dựa theo engine của Microsoft trên VirusTotal.

Giao diện của Tools:



Kết quả trả về là tập mẫu đã được phân loại và file log chứa thông tin về số lượng các loại virus liệt kê theo tên của Microsoft.

* Tiếp theo sẽ sử dụng Tool: ***scanfolder.py*** để lọc ra những loại virus nào là Javascript, VB và html sau đó chỉ lấy Javascript làm mẫu.
* Tên của Virus được lấy về thường sẽ đặt theo mã hash sha256 của file virus.

Nguồn lấy mẫu sạch Javascript:

* Sử dụng công cụ ***crawler\_ver1.py*** để download các file html với đường link cho trước để làm mẫu. Những file HTML này sau đó sẽ đưa vào engine EmulJS để lọc lấy code JavaScript, loại bỏ các đoạn mã HTML dư thừa
* Để chuẩn hóa, sử dụng công cụ ***find\_sample.py*** để tìm kiếm trong một đường dẫn tên file có mã sha256 được chỉ định.

# Triển khai

## Sử dụng mạng Neural: CNN

Mô tả cấu trúc mạng:

Các thành phần gồm có:

Đầu vào

Đầu ra

Các layer ẩn

Ý nghĩa các tham số

Kết quả đạt được

Đánh giá

# Kết luận

# Tài liệu tham khảo

* <https://heimdalsecurity.com/blog/javascript-malware-explained>
* Data Mining with Decision Trees – Theory and Applications 2nd Edition – Lior Rokach – Oded Maimon
* Heimdal Sercurity: heimdalsecurity.com/blog/javascript-malware-explained