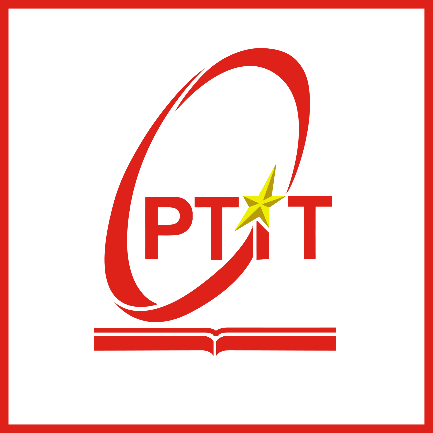
**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG KHOA AN TOÀN THÔNG TIN**

**-----------------o-0-o-----------------**



**BÁO CÁO BÀI TẬP LỚN**

**Môn học: Phần tích mã độc**

**Đề tài: Phân tích tĩnh malware Trojan Zeus Banking**

|  |  |  |
| --- | --- | --- |
| **Giảng viên hướng dẫn** | **:** | Đinh Trường Duy |

|  |  |  |
| --- | --- | --- |
| **Nhóm lớp** | **:** | 01 |
| **Nhóm bài tập lớn** | **:** | 02 |
| **Danh sách thành viên** | | |
| Nguyễn Khắc Tuyên | **:** | B21DCAT217 |
| Bùi Thanh Hiếu | **:** | B21DCCN085 |
| Phí Đức Tuân | **:** | B21DCAT202 |
| Nguyễn Khắc Hưng | **:** | B21DCAT099 |

Hà Nội-2024

MỤC LỤC

[**I. CƠ SỞ LÝ THUYẾT** 4](#_Toc180191782)

[**1.** **Tổng quan về mã độc** 4](#_Toc180191783)

[**2. Phân tích tĩnh** 5](#_Toc180191784)

[**3. Một số công cụ** 12](#_Toc180191785)

[**II. CÀI ĐẶT VÀ CẤU HÌNH** 14](#_Toc180191786)

[**1.** **Một số công cụ** 14](#_Toc180191787)

[**2.** **Cấu hình cài đặt** 14](#_Toc180191788)

[**III. TROJAN ZEUS BANKING** 14](#_Toc180191789)

[**IV. PHÂN TÍCH - THỰC NGHIỆM** 15](#_Toc180191790)

[**1. Xác định loại của tệp** 15](#_Toc180191791)

[**2. Phân tích mã Hash** 16](#_Toc180191792)

[**3. Phân tích mẫu đang có sử dụng công cụ kết hợp nhiều Antivirus (AV)** 17](#_Toc180191793)

[**4. Tách chuỗi từ chương trình** 21](#_Toc180191794)

[**5. Xác định các cách che giấu của tập tin** 22](#_Toc180191795)

[**6. Nhận biết mã độc có bị nén hay mã hóa hay không** 24](#_Toc180191796)

[**7. Tập tin PE** 24](#_Toc180191797)

[**8.** **Phân loại** 33](#_Toc180191798)

[**V. TỔNG KẾT** 34](#_Toc180191799)

[**TÀI LIỆU THAM KHẢO** 35](#_Toc180191806)

**Mục lục hình ảnh**

[***Hình 1.1: Cơ chế tiêm mã vào tiến trình và thực thi*** 9](#_Toc180191503)

[***Hình 1.2: Mô tả các phần của tệp thực thi (PE Sections)*** 11](#_Toc180191504)

[***Hình 1.3: Mô tả các trường trong phần của tệp thực thi (PE Section Fields)*** 11](#_Toc180191505)

[***Hình 1.4: Phân tích chi tiết tệp PE bằng PEStudio*** 11](#_Toc180191506)

[***Hình 4.1: File nghi ngờ*** 15](#_Toc180191507)

[***Hình 4.2: Sử dụng PEStudio để phân tích tệp đáng ngờ*** 16](#_Toc180191508)

[***Hình 4.3: Phân tích dấu vết và hàm băm của tệp PE bằng PEStudio*** 17](#_Toc180191509)

[***Hình 4.4: Phân tích hàm băm và thuộc tính tệp bằng HashMyFiles*** 18](#_Toc180191510)

[***Hình 4.5: Kết quả phân tích tệp độc hại trên VirusTotal*** 19](#_Toc180191511)

[***Hình 4.6: Kết quả phân tích tệp độc hại trên VirusTotal*** 19](#_Toc180191512)

[***Hình 4.12: Kết quả phân tích chuỗi tĩnh và giải mã từ FLOSS*** 23](#_Toc180191513)

[***Hình 4.13: Kết quả phân tích chuỗi tĩnh và giải mã từ FLOSS*** 23](#_Toc180191514)

[***Hình 4.14: Phân tích thông tin chi tiết tệp thực thi bằng Exeinfo PE*** 23](#_Toc180191515)

[***Hình 4.15: Phân tích các phần của tệp thực thi bằng PEStudio*** 24](#_Toc180191516)

[***Hình 4.16: Phân tích thư viện và hàm nhập của tệp thực thi trên PEStudio*** 25](#_Toc180191517)

[***Hình 4.17: Phân tích thư viện và hàm nhập của tệp thực thi trên PEStudio*** 27](#_Toc180191518)

[***Hình 4.18: Phân tích chi tiết tiêu đề DOS và các phần của tệp thực thi trên PEView*** 29](#_Toc180191519)

[***Hình 4.20: Phân tích kỹ thuật né tránh và phát hiện máy ảo bằng CAPA*** 30](#_Toc180191520)

[***Hình 4.21: Phân tích mã máy: GetTickCount và AllowSetForegroundWindow*** 32](#_Toc180191521)

[***Hình 4.22: Biểu đồ dòng mã: GetTickCount và AllowSetForegroundWindow*** 32](#_Toc180191522)

[***Hình 4.23: Biểu đồ dòng mã: GetCapture trong mã độc*** 33](#_Toc180191523)

[***Hình 4.24: Biểu đồ dòng mã: GetClipboardData trong mã độc*** 33](#_Toc180191524)

[***Hình 4.25: Biểu đồ dòng mã: GetDriveTypeA trong mã độc*** 34](#_Toc180191525)

[***Hình 4.26: Kết quả phát hiện mã độc Trojan và Backdoor trên VirusTotal*** 34](#_Toc180191526)

[***Hình 4.27: Kết quả phát hiện mã độc Trojan và Backdoor trên VirusTotal*** 35](#_Toc180191527)

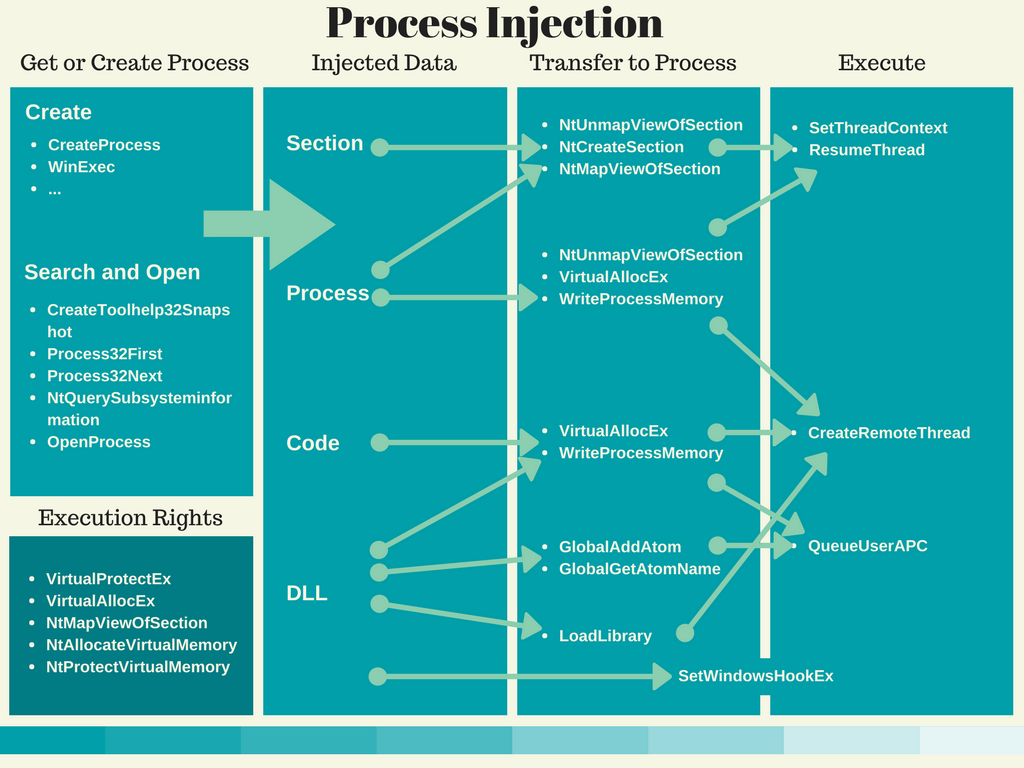
[***Hình 4.28: Kết quả phát hiện mã độc Trojan và Backdoor trên VirusTotal*** 35](#_Toc180191528)

1. **CƠ SỞ LÝ THUYẾT**
2. **Tổng quan về mã độc**
3. **Mã độc là gì?**

* Mã độc (Malware) là phần mềm độc hại được thiết kế để xâm nhập vào hệ thống máy tính mà không có sự đồng ý của người dùng. Mục đích của mã độc có thể là gây hại cho hệ thống, đánh cắp dữ liệu cá nhân hoặc làm gián đoạn hoạt động của máy tính.
* Mã độc có thể tồn tại dưới nhiều hình thức, như tập tin thực thi, script hoặc mã nguồn. Kẻ tấn công sử dụng mã độc để thu thập thông tin nhạy cảm, theo dõi hành vi người dùng hoặc chiếm quyền điều khiển máy tính.
* Thông thường, mã độc xâm nhập qua email lừa đảo, trang web độc hại hoặc thiết bị lưu trữ USB. **Giới thiệu về trojan horses**
* **Trojan Horses**:
  + Đây là loại mã độc ngụy trang dưới vỏ bọc phần mềm vô hại và thậm chí có thể hoạt động.
  + Như một chương trình hợp pháp trong hệ thống trước khi thực hiện các hành vi độc hại được chỉ định
  + Các đoạn mã của Trojan được “che giấu” trong các phần mềm máy tính bình thường để bí mật xâm nhập vào máy tính.
  + Tới thời điểm thuận lợi, chúng sẽ đánh cắp thông tin cá nhân và chiếm quyền điều khiển máy tính…

1. **Phân tích tĩnh**
   1. **Phân tích tĩnh là gì?**

* Phân tích tĩnh là kỹ thuật phân tích tệp nghi ngờ (mẫu) mà không thực thi nó.
* Phân tích tĩnh được chia làm 2 loại:
  + Phân tích tĩnh cơ bản.
  + Phân tích tĩnh nâng cao.
  1. **Vai trò của phân tích tĩnh**
* Cung cấp thông tin ban đầu mà không cần thực thi tệp: Phân tích tĩnh có thể được thực hiện mà không cần thực thi tệp. Điều này cho phép thu thập thông tin ban đầu về tệp mà không kích hoạt mã độc tiềm ẩn.
* Xác định hành vi tiềm ẩn: Phân tích tĩnh có thể tiết lộ những gợi ý về hành vi tiềm ẩn của tệp như kết nối mạng, truy cập hệ thống tập tin, v.v. Điều này có thể giúp xác định liệu tệp có đáng nghi ngờ hay không.
* Xác định cách thức hoạt động: Phân tích tĩnh có thể tiết lộ cách thức hoạt động của tệp thông qua việc khám phá cấu trúc, luồng điều khiển và các hàm.
* Hỗ trợ phân tích động sau đó: Thông tin thu thập được từ phân tích tĩnh có thể hỗ trợ phân tích động sau đó khi thực thi tệp trong môi trường cô lập
* Phân loại tệp: Phân tích tĩnh có thể cung cấp đủ thông tin để phân loại nhanh tệp là bening hoặc đáng ngờ.
  1. **Một số công cụ phân tích tĩnh**
* Capa: Capa tập trung vào việc xác định các đặc điểm quan trọng của mã độc, phát hiện chữ ký.
* Floss: Cũng giống như Capa, công cụ Floss cũng tìm chữ ký trong tệp thực thi, ngoài ra nó còn tìm chuỗi.
* Cutter: Công cụ có khả năng thực hiện debugging cho các tệp thực thi
* PEStudio: Công cụ này cung cấp thông tin chi tiết về các thành phần quan trọng trong tệp thực thi: module, ký sự số bản quyền, thông tin nhúng giúp kiểm tra tệp PE.
* CFF Explorer: Công cụ này cung cấp thông tin một cách chi tiết về các cấu trúc của tệp thực thi, bao gồm module, bảng phân đoạn.
  1. **Quy trình phân tích tĩnh**
     1. **Xác định loại của tệp**
* Phần mở rộng tệp (File extension) là một phần của tên tệp được gắn liền với tên tệp để chỉ định định dạng hoặc loại tệp, vd: .exe, .dll, .sys, docx, xlsx...
  + Phần mở rộng tệp giúp hệ điều hành và các ứng dụng phần mềm nhận biết loại tệp và sử dụng chương trình mở tương ứng để xem hoặc xử lý tệp.
  + Các kẻ tấn công có thể sử dụng các chiêu trò khác nhau để che dấu tệp của họ bằng cách sửa đổi phần mở rộng tệp và thay đổi giao diện của tệp để lừa người dùng vào thực thi nó.
* Chữ ký tệp (File Signature) có thể được sử dụng để xác định loại tệp, thay cho phần mở rộng tệp.
  + Chữ ký tệp là trình tự byte duy nhất được viết vào phần đầu tệp.
  + Các tệp khác nhau có chữ ký khác nhau có thể được sử dụng để xác định loại tệp.
* Timestamp: Phần mềm đóng gói chứa thông tin xác định thời điểm tệp được được biên dịch; kiểm tra trường này có thể cho bạn biết khi phần mềm độc hại được tạo lần đầu.
* Các công cụ sử dụng để xác định loại của tệp:
  + Sử dụng phương pháp thủ công để xác định loại tệp là tìm kiếm chữ ký tệp bằng cách mở nó trong một trình chỉnh sửa hex.
  + Trình chỉnh hex là một công cụ cho phép một nhà điều tra kiểm tra từng byte của tệp; hầu hết trình chỉnh hex cung cấp nhiều chức năng giúp trong việc phân tích tệp.
  + Sử dụng phương pháp xác định bằng công cụ
    1. **Phân tích mã Hash**
* Quá trình tạo các giá trị băm cho các tệp nghi ngờ dựa trên nội dung của chúng, cũng giống như tạo vân tay (Fingerprinting) cho mã độc.
* Các thuật toán băm thường được sử dụng như **MD5**, **SHA1** hoặc **SHA256**.
* Các công cụ có thể sử dụng:
  + Linux: Md5sum, Sha256sum, Sha1sum …
  + Windows: HashMyFiles, FsumFrontEnd, Jacksum …
    1. **Tách chuỗi từ chương trình mã động**
* **Sử dụng phần mềm để lấy Strings từ các chương trình có thể là mã độc**
  + Để trích xuất chuỗi từ một tập tin nhị phân nghi ngờ, bạn có thể sử dụng tiện ích strings trên hệ thống Linux.
  + Đối với các mẫu mã độc hại, cũng sử dụng chuỗi Unicode (2 byte mỗi ký tự).
  + Trên Windows, PEStudio là một công cụ tiện ích hiển thị cả chuỗi ASCII và chuỗi Unicode.
  + PEStudio là một công cụ phân tích PE tuyệt vời để thực hiện đánh giá ban đầu về phần mềm độc hại từ một tập tin nhị phân nghi ngờ và được thiết kế để thu thập các mẫu thông tin hữu ích từ một tập tin thực thi PE.
* **Lấy những chuỗi đã bị che dấu bằng Floss**
  + Trong hầu hết các trường hợp, tác giả phần mềm độc hại sử dụng các kỹ thuật che dấu chuỗi đơn giản để tránh bị phát hiện.
  + Trong những trường hợp như vậy, những chuỗi đã được che giấu sẽ không xuất hiện trong tiện ích strings và các công cụ trích xuất chuỗi khác.
  + FireEye Labs Obfuscated String Solver (FLOSS) là một công cụ được thiết kế để tự động xác định và trích xuất chuỗi đã được làm mờ từ phần mềm độc hại.
* **Xác định cách che giấu của tập tin**
  + Dù việc trích xuất chuỗi là một kỹ thuật xuất sắc để thu thập thông tin có giá trị, thường tác giả phần mềm độc hại sẽ che giấu hoặc bảo vệ tập tin nhị phân của họ.
  + Điều này được sử dụng bởi tác giả phần mềm độc hại để bảo vệ cấu trúc bên trong của phần mềm độc hại khỏi các nhà nghiên cứu bảo mật, các chuyên gia phân tích phần mềm độc hại và kỹ sư đảo ngược mã.
  + Các kỹ thuật làm mờ này làm cho việc phát hiện/phân tích tập tin trở nên khó khăn; việc trích xuất chuỗi từ tập tin nhị phân như vậy dẫn đến rất ít chuỗi, và hầu hết các chuỗi được che giấu.
* **Packers and Cryptors**
  + Packer là một chương trình lấy tập tin thực thi làm đầu vào và sử dụng nén để để che giấu nội dung của tập tin thực thi.
  + Một Cryptor tương tự như một Packer, nhưng thay vì sử dụng nén, nó sử dụng mã hóa để giấu nội dung của tập tin thực thi, và nội dung đã được mã hóa được lưu trữ trong tập tin thực thi mới
* **Phát hiện các tệp tin ẩn giấu bằng Exeinfo PE** 
  + Hầu hết các tập tin thực thi hợp pháp không che giấu nội dung, nhưng một số tập tin thực thi có thể thực hiện điều này để ngăn người khác kiểm tra mã nguồn của họ.
  + Khi bạn gặp một mẫu mã đã được đóng gói, có khả năng cao nó sẽ là độc hại.
  + Để phát hiện các chương trình đóng gói trên Windows, bạn có thể sử dụng một công cụ miễn phí như Exeinfo PE ,nó có giao diện người dùng dễ sử dụng, để phát hiện các trình biên dịch, chương trình đóng gói hoặc mã hóa được sử dụng để xây dựng chương trình.
    1. **Nhận biết mã độc có bị nén hay mã hóa không?**
* Các công cụ nhận dạng gói (packers) đã biết sẽ được xác định bằng các nhận dạng gói như Detect it Easy, PEiD, và các công cụ tương tự.
* Tuy nhiên, phần mềm độc hại thường được gói bởi các gói tùy chỉnh, được tạo ra đặc biệt để tránh các chương trình antivirus.
* Những gói này không thể được xác định bằng những công cụ này.
* Có một số nguyên tắc tổng quát có thể chỉ ra liệu một tập tin có được gói hay không, như:
  + Có độ entropy cao
  + Các hàm nhập điển hình của packed files hay nhiễm độc



##### Hình 1.1: Cơ chế tiêm mã vào tiến trình và thực thi

- Có dòng rác, không có nghĩa trong tệp nhị phân

**3.4.7 Tệp tin PE**

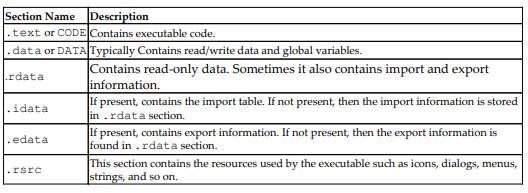
- Các tệp thực thi Windows (như .exe, .dll, .sys, .ocx và .drv) phải tuân thủ theo định dạng PE (Portable Executable).

- Tệp PE là một loạt các cấu trúc và thành phần con chứa thông tin cần thiết cho hệ điều hành để tải nó vào bộ nhớ.

- Công cụ:

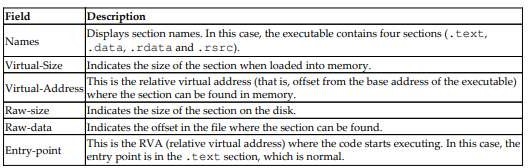
* CFF Explorer: <http://www.ntcore.com/exsuite.php>
* PEInternals:<http://www.andreybazhan.com/pe-internals.html>
* PPEE(puppy): [https://www.mzrst.com](https://www.mzrst.com/)
* Thường, phần mềm độc hại tương tác với tệp, registry, mạng, và các thành phần khác.
* Để thực hiện các tương tác như vậy, phần mềm độc hại thường phụ thuộc vào các hàm được hệ điều hành cung cấp.
* Windows xuất ra hầu hết các hàm của nó, gọi là Application Programming Interfaces (API), cần thiết cho các tương tác này trong các tệp Dynamic Link Library (DLL).
* Các tệp thực thi nhập và gọi các hàm này thường từ các DLL khác nhau cung cấp các chức năng khác nhau.
* Các hàm mà một tệp thực thi nhập từ các tệp khác (chủ yếu là DLL) được gọi là các hàm nhập (hàm imports).
  + Kiểm tra các imports có thể:
  + Kiểm tra Exports
* Nội dung của tệp PE được chia thành các Section.
* Các section này đại diện cho code hoặc data và chúng có các thuộc tính trong bộ nhớ như read/ write .

Trong quá trình biên dịch của tệp thực thi, các trình biên dịch thêm các tên section liên tục. Bảng sau đây cho biết một số section thông thường trong một tệp PE:

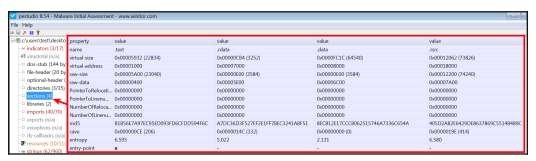


##### Hình 1.2: Mô tả các phần của tệp thực thi (PE Sections)

* Các tên section này chủ yếu dành cho con người và không được hệ điều hành sử dụng, điều này có nghĩa rằng có khả năng một kẻ tấn công hoặc phần mềm làm mờ thông tin có thể tạo ra các section với tên khác nhau.
* Thông tin về các section này (như tên phần, vị trí của phần và các đặc điểm của nó) có sẵn trong bảng phần trong PE header (Portable Executable). Kiểm tra bảng phần sẽ cung cấp thông tin về phần và các đặc điểm của nó.
* Khi bạn tải một tệp thực thi trong pestudio và nhấp vào mục "sections," nó sẽ hiển thị thông tin về các phần được trích xuất từ bảng phần và các thuộc tính của chúng.



##### Hình 1.3: Mô tả các trường trong phần của tệp thực thi (PE Section Fields)



##### Hình 1.4: Phân tích chi tiết tệp PE bằng PEStudio

- Kiểm tra PE Resources: Các tài nguyên cần thiết bởi tệp thực thi như biểu tượng, menu, hộp thoại và chuỗi được lưu trữ trong phần tài nguyên (.rsrc) của tệp thực thi. Thường, các kẻ tấn công lưu trữ thông tin như các tệp nhị phân bổ sung, tài liệu mạo danh và dữ liệu cấu hình trong phần tài nguyên, vì vậy kiểm tra tài nguyên có thể tiết lộ thông tin quý giá về một tệp thực thi.

**3.4.8 Phân loại mã độc**

* Phân tích mẫu mã độc có thể cho biết mẫu đó có thuộc họ mã độc (family), hoặc có tính chất giống với những mẫu đã phân tích trước đó hay không. Cụ thể:
  + Phân tích mẫu mã độc có thể xác định nó thuộc gia đình malware nào, chẳng hạn như ransomware, trojan, backdoor, ...
  + So sánh mẫu đó với các mẫu đã biết để xác định nó có đặc trưng giống với mẫu đã biết trước đó hay không, ví dụ so sánh mã, hành vi, kỹ thuật lây lan, ...
  + Nếu so sánh có kết quả gần giống thì có thể kết luận đó là một mẫu thuộc cùng gia đình (family) hoặc biến thể của một mã đã biết

Mặc dù việc tạo băm mã hóa (MD5/SHA1/SHA256) là một kỹ thuật tốt để phát hiện các mẫu giống nhau, nhưng nó không giúp xác định các mẫu tương tự. Rất nhiều lần, tác giả phần mềm độc hại thay đổi những khía cạnh rất nhỏ của phần mềm độc hại, điều này thay đổi hoàn toàn giá trị băm.

* Phân loại mã độc sử dụng Fuzzy Hashing:

Fuzzy hashing là một phương pháp để so sánh sự tương đồng giữa các tập tin.

Kỹ thuật này so sánh một nhị phân nghi ngờ với các mẫu trong kho lưu trữ để xác định các mẫu tương tự là xác định các mẫu thuộc cùng gia đình malware hoặc cùng nhóm tác nhân

* Phân loại mã độc sử dụng Import Hash (imphash):

Phân tích mã độc sử dụng các hàm API (hay còn gọi là imphash) là một kỹ thuật có thể được sử dụng để xác định mẫu liên quan và mẫu được sử dụng bởi các nhóm tác nhân đe dọa.

Đặc điểm này dựa trên việc tính toán giá trị băm dựa trên tên hàm/ hàm nhập (API) và thứ tự của chúng trong tập tin thực thi. Nếu các tập tin được biên dịch từ cùng một nguồn và theo cùng một cách, những tập tin đó sẽ có giá trị imphash giống nhau.

1. **Một số công cụ**
   1. **PEstudio**

* PEstudio là một công cụ phân tích tĩnh mã độc được thiết kế để kiểm tra các tệp PE (Portable Executable). Công cụ này giúp phát hiện các dấu hiệu đáng ngờ mà không cần thực thi mã, bảo đảm an toàn khi phân tích phần mềm độc hại.

**Các tính năng chính:**

* Phân tích các thành phần của tệp PE như: header, sections, imported functions, exported functions, resources, và signatures.
* Cung cấp thông tin về chứng chỉ (nếu có) của tệp tin.
* Phân tích các APIs mà tệp PE có thể sử dụng để kiểm tra khả năng chứa mã độc.
* Tương thích với VirusTotal để kiểm tra tệp tin có bị phát hiện là mã độc hay không.
  1. **FLOSS (FireEye Labs Obfuscated String Solver)**
* FLOSS là công cụ được sử dụng để giải mã các chuỗi ký tự (string) bị ẩn hoặc mã hóa trong phần mềm mã độc. Nó được phát triển bởi FireEye Labs nhằm mục tiêu phát hiện các chuỗi đã bị làm xáo trộn hoặc ẩn trong mã.

**Các tính năng chính:**

* Tự động phát hiện và giải mã các chuỗi ký tự đã bị mã hóa trong mã thực thi.
* Kết hợp với các công cụ phân tích tĩnh và động để cung cấp thêm thông tin chi tiết về mã độc.
* FLOSS có thể giúp phát hiện tên miền, URL, key đăng nhập ẩn, và nhiều thông tin khác mà kẻ tấn công cố ý giấu.
  1. **PEview**
* PEview là một công cụ phân tích tĩnh khác dành cho việc kiểm tra cấu trúc các tệp PE (Portable Executable). PEview cung cấp một giao diện đơn giản để người dùng có thể xem từng thành phần chi tiết trong cấu trúc PE của tệp tin.

**Các tính năng chính:**

* Xem các section header, import/export tables, và dữ liệu của tệp PE một cách chi tiết.
* Kiểm tra các cấu trúc nội tại của tệp PE mà không cần thực thi mã.
* Phân tích các trường thông tin quan trọng như thời gian biên dịch, tham chiếu đến thư viện DLL, và điểm nhập chương trình.
  1. **Capa**
* Capa là công cụ phân tích mã độc do FireEye phát triển, giúp xác định các chức năng của phần mềm mã độc dựa trên cách chúng được viết và vận hành.
* Nó thực hiện phân tích tĩnh và tìm kiếm các mẫu mã (patterns) để nhận diện các hành vi hoặc chức năng đặc biệt.

**Các tính năng chính:**

* Phân tích các chức năng mã độc như kết nối mạng, mã hóa, tương tác với hệ thống tệp, hoặc thao tác bộ nhớ.
* Cung cấp thông tin chi tiết về chức năng của mã độc mà không cần phải thực thi nó.
* Dựa trên quy tắc (rules) để phát hiện các hành vi khác nhau của phần mềm độc hại.

1. **CÀI ĐẶT VÀ CẤU HÌNH**
2. **Một số công cụ**

* **PEStudio**: <https://pestudio.en.lo4d.com/download/mirror-ls1>
* **Cutter**: <https://cutter.re/>
* **Capa**: <https://github.com/mandiant/capa/releases>
* **CFF Explorer**: <https://ntcore.com/?page_id=388>

1. **Cấu hình cài đặt**

* Sau khi tải về PEStudio, Cutter, và CFF Explorer, tiến hành giải nén các tệp đã tải xuống và chạy trực tiếp các công cụ từ các tệp này mà không cần cài đặt thêm.
* Đối với Cmder, sau khi giải nén, bạn có thể chạy trực tiếp công cụ này. Sau đó, bắt đầu cấu hình Capa và Floss bằng cách cài đặt chúng vào môi trường làm việc thông qua các đường dẫn tương ứng.

1. **TROJAN ZEUS BANKING**

* Virus **trojan Zeus** được tạo ra lần đầu tiên vào năm 2007, khi tin tặc ở Đông Âu sử dụng nó để nhắm vào Bộ Giao thông Vận tải Hoa Kỳ.
* Mặc dù khó có thể nói chắc chắn ai đã tạo ra nó, nhưng cuộc tấn công thực sự bùng nổ khi mã độc của nó bị công khai vào năm 2011.
* Kể từ đó, nó đã tạo ra hàng chục biến thể khiến các chuyên gia bảo mật internet và cơ quan thực thi pháp luật bận rộn.
* Zeus Banking Trojan hay còn gọi là ZueS hoặc Zbot, là một gói phần mềm mãi độc dạng Trojan, nó phổ biến và đáng ngại trong lĩnh vực tài chính trên internet.
* Nó được thiết kế để đánh cắp thông tin đăng nhập của người dùng, cho phép kẻ tấn công truy cập và lừa đảo trong quá trình giao dịch tài khoản trực tuyến.
* **Link tải**:

**TrojanBankingMalware**: [https://github.com/ytisf/theZoo/tree/master/malware/Binarie s/ZeusBankingVersion\_26Nov2013](https://github.com/ytisf/theZoo/tree/master/malware/Binarie%20s/ZeusBankingVersion_26Nov2013)

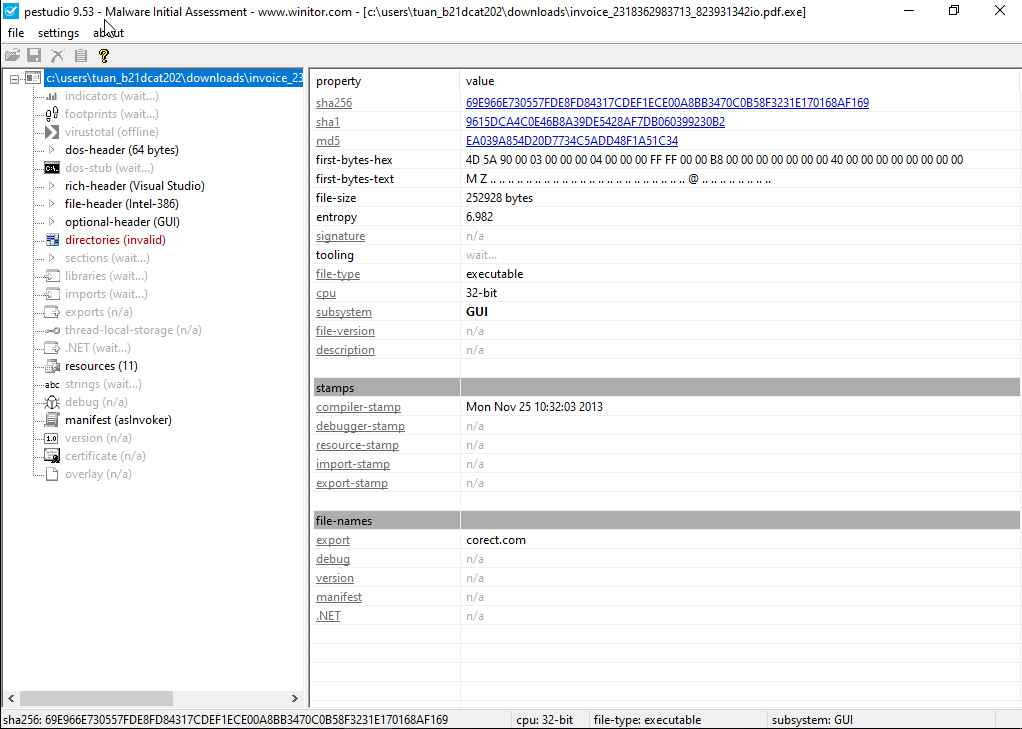
1. **PHÂN TÍCH - THỰC NGHIỆM**
2. **Xác định loại của tệp**

* Phát hiện một tệp đáng ngờ trong máy với đuôi file là .pdf



##### Hình 4.1: File nghi ngờ

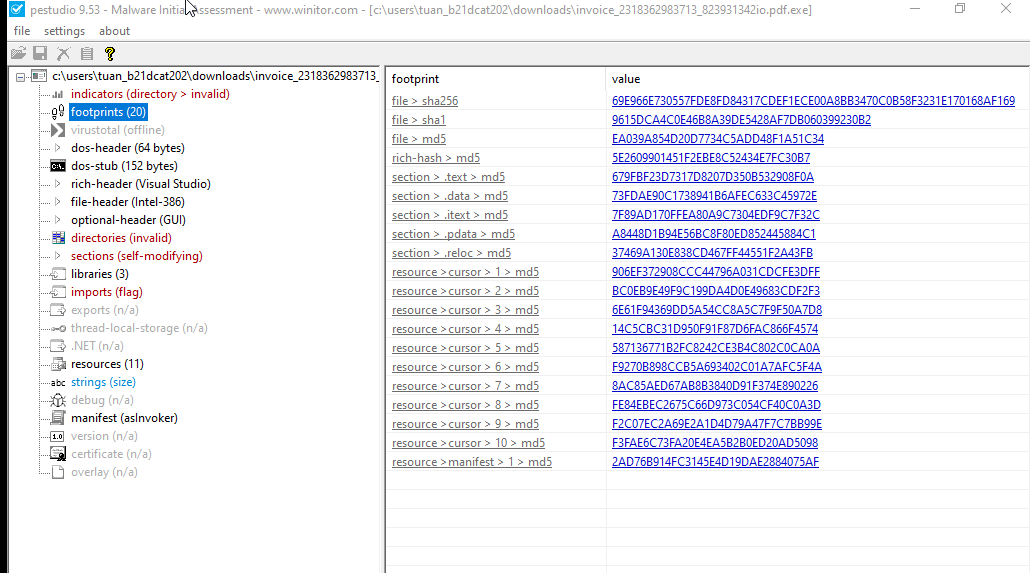
* Sử dụng PE studio để xem kĩ file nghi ngờ



##### Hình 4.2: Sử dụng PEStudio để phân tích tệp đáng ngờ

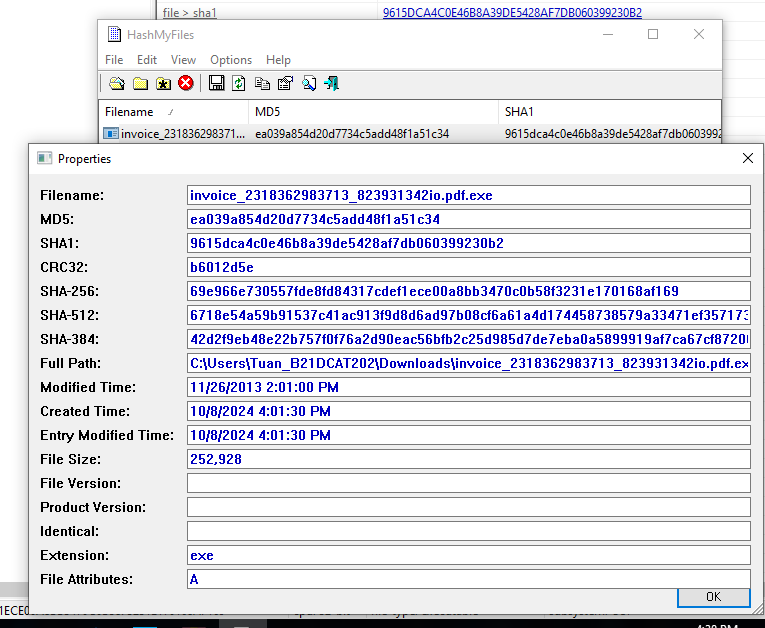
* Khi sử dụng PEStudio để phân tích một tệp đáng ngờ ta thấy tên file là **invoice\_2318362983713\_823931342io.pdf.exe** . Như có thể thấy định dạng của file là exe nhưng nó đang cố giả dạng thành file pdf để đánh lừa người dùng.
* Loại tệp hiển thị là Portable Executable 32-bit, cho thấy đây là tệp được phát triển nhằm mục tiêu tấn công hệ điều hành Windows.
* Trong phần "Stamps", chúng ta có thể xác định rằng tệp này lần đầu tiên được biên dịch vào ngày Thứ Hai, 25/11/2013 lúc 10:32:03

1. **Phân tích mã Hash**
2. **Kiểm Tra mã HASH thu được của Pestudio**



##### Hình 4.3: Phân tích dấu vết và hàm băm của tệp PE bằng PEStudio

1. **Sử dụng HashMyFiles để tính toán mã Hash của File gốc**



##### Hình 4.4: Phân tích hàm băm và thuộc tính tệp bằng HashMyFiles

1. **So sánh - Kiểm Tra tính toàn vẹn của file mã độc**

* Khi tiến hành phân tích mã độc bằng PEStudio, mã băm thu được là:

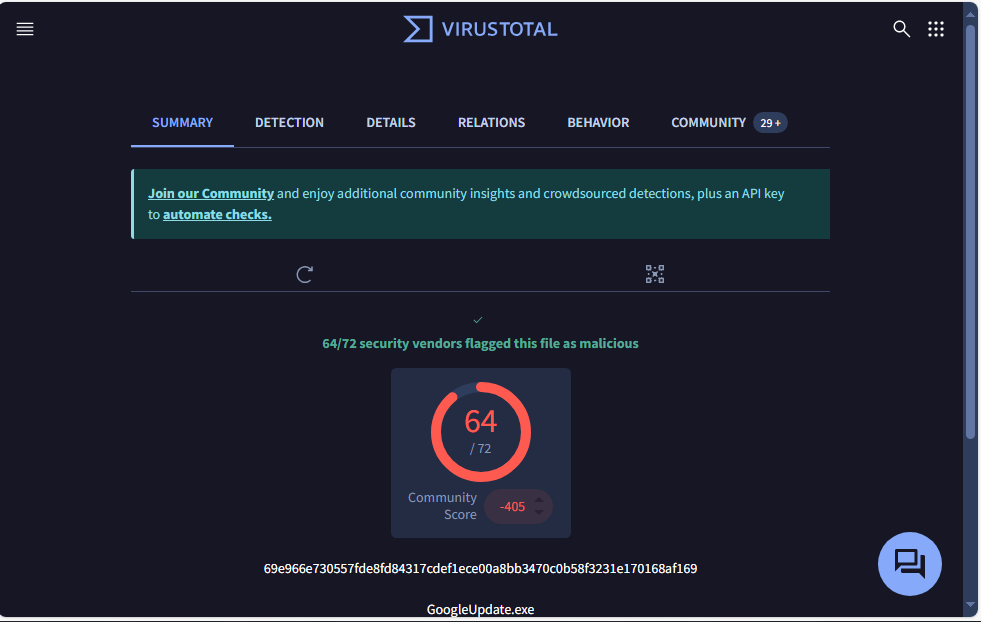
**69E966E730557FDE8FD84317CDEF1ECE00A8BB3470C0B58F3231E170168AF169**

* Trong khi đó, khi sử dụng HashMyFiles, mã băm của tệp cũng được tính toán là:

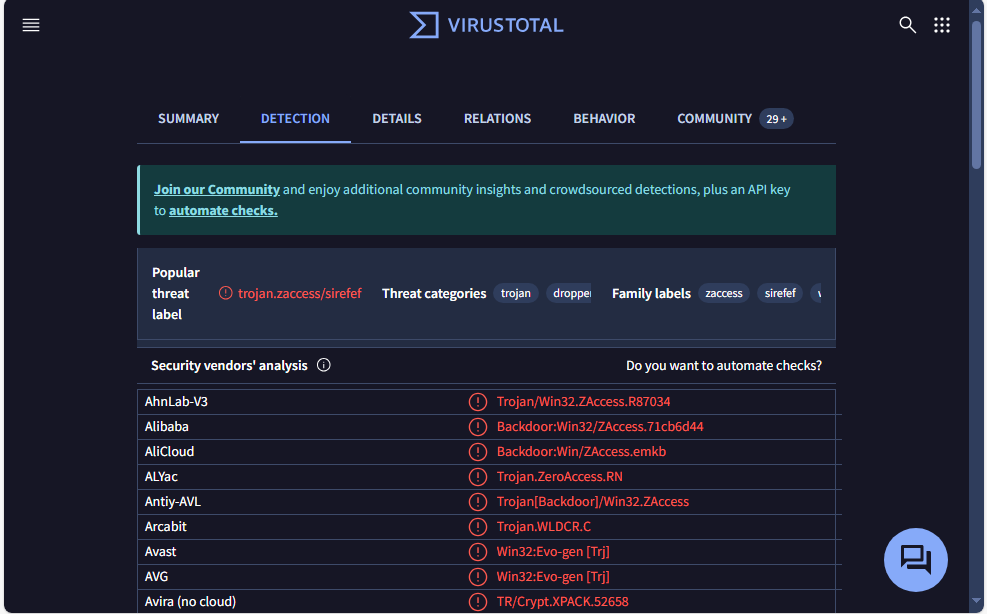
**69e966e730557fde8fd84317cdef1ece00a8bb3470c0b58f3231e170168af169**

* Nhìn chung, hai mã hash này hoàn toàn trùng khớp, điều này chứng tỏ rằng tệp không bị thay đổi kể từ khi bắt đầu quá trình tính toán hash. Sự toàn vẹn của tệp vẫn được đảm bảo.
* Trong trường hợp này, việc mã hash không thay đổi cho thấy không có sự khác biệt giữa các tệp trong quá trình phân tích.

1. **Phân tích mẫu đang có sử dụng công cụ kết hợp nhiều Antivirus (AV)**

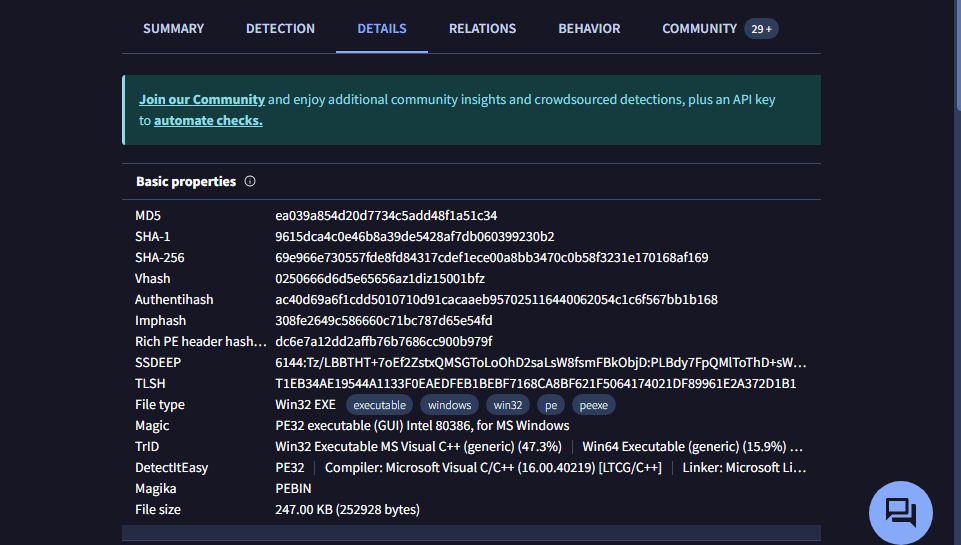


##### Hình 4.5: Kết quả phân tích tệp độc hại trên VirusTotal

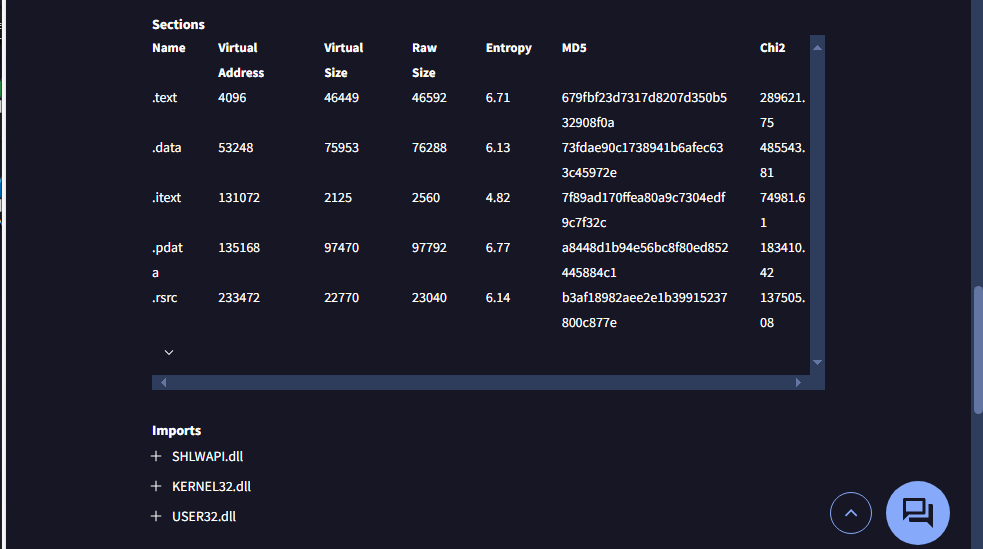


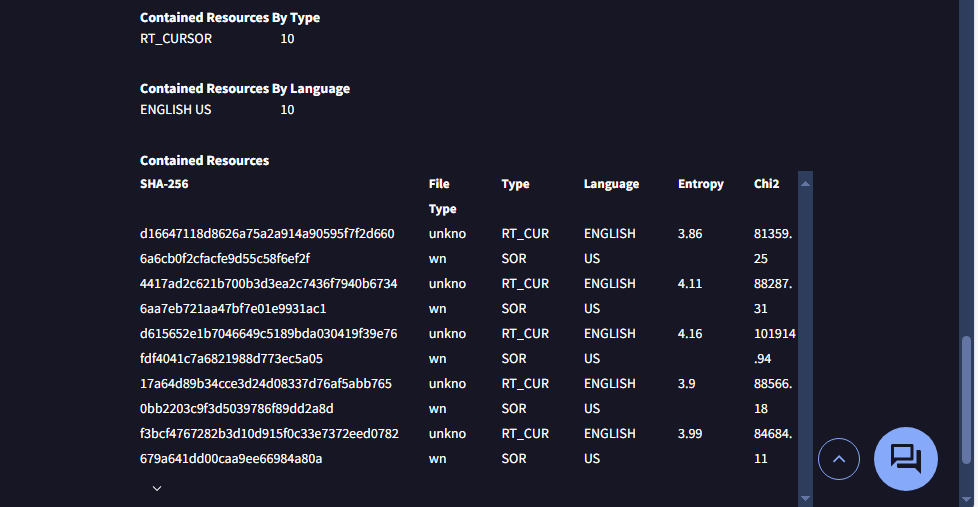
##### Hình 4.6: Kết quả phân tích tệp độc hại trên VirusTotal

* Như ta có thể thấy các nhà cung cấp đều thông báo file có hại

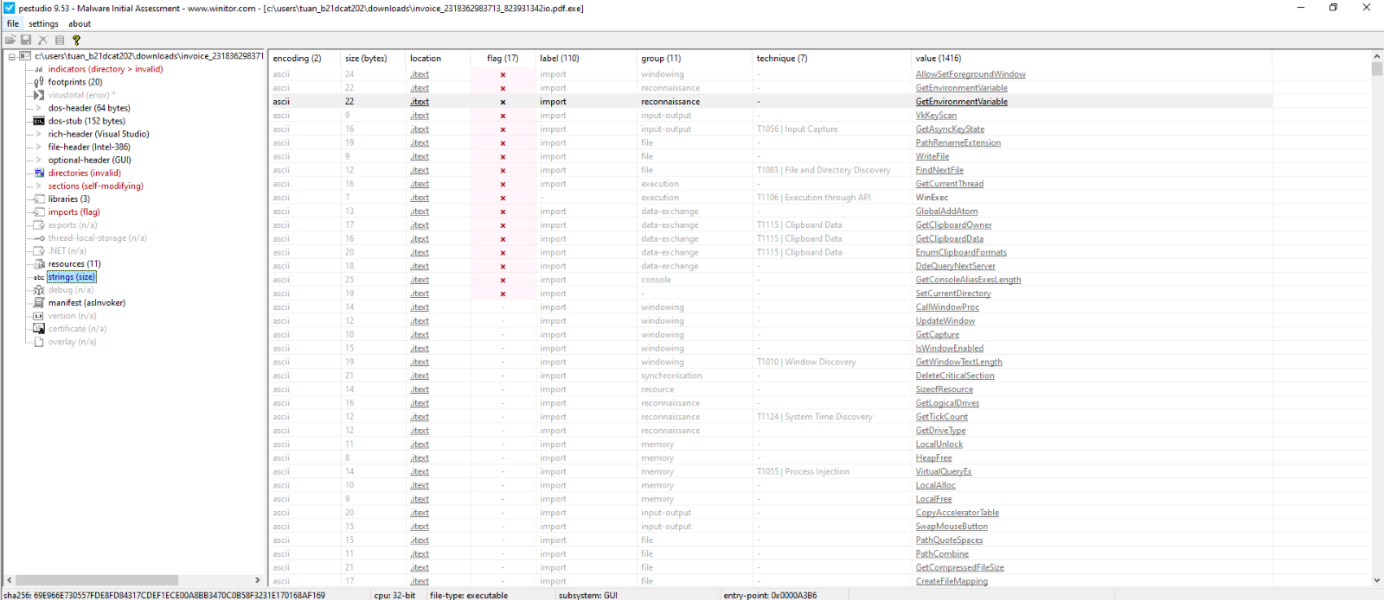
  
Hình 4.7: Thông tin chi tiết về tệp thực thi từ VirusTotal

  
Hình 4.8: Thông tin chi tiết về tệp thực thi từ VirusTotal

  
Hình 4.9: Thông tin về các phần và hàm nhập của tệp PE trên VirusTotal

  
Hình 4.10: Phân tích tài nguyên chứa trong tệp thực thi trên VirusTotal

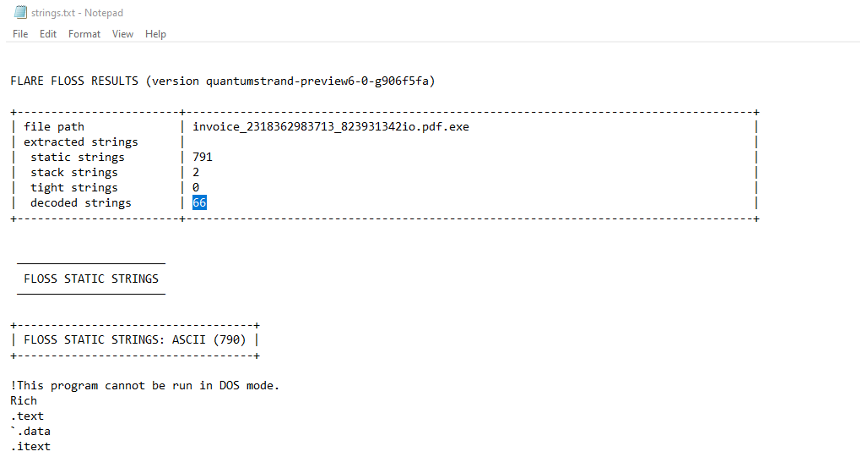
1. **Tách chuỗi từ chương trình**
2. **Sử dụng Pestudio để tách chuỗi từ chương trình**

  
Hình 4.11: Danh sách hàm nhập và kỹ thuật tấn công mã độc trong PEStudio

* Khi sử dụng PEStudio để hiển thị cả chuỗi ASCII và Unicode, có thể thấy tổng cộng hơn 1416 chuỗi trong tệp tin. Tuy nhiên, do giới hạn của PEStudio, chỉ có 1416 chuỗi đầu tiên được hiển thị.
* Trong PEStudio, các chuỗi được đánh dấu bằng cờ “x” thường được coi là có khả năng liên quan đến hành vi độc hại của phần mềm mã độc. Những chuỗi này có thể bao gồm tên file, đường dẫn, lệnh hệ thống, hoặc các yếu tố khác thường được mã độc sử dụng trong quá trình hoạt động của mình

1. **Lấy chuỗi bị che dấu bằng floss**

* Sẽ có những chuỗi mà mã độc không muốn ta phát hiện ra nên đã che giấu đi và để lấy được những chuỗi được che giấu đó ta sử dụng FLOSS.

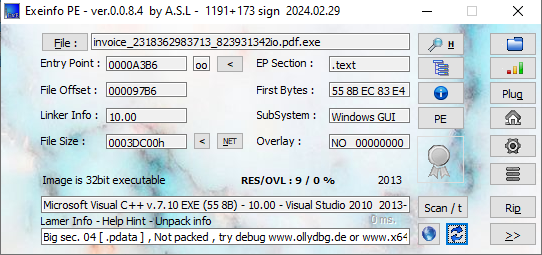


##### Hình 4.12: Kết quả phân tích chuỗi tĩnh và giải mã từ FLOSS



##### Hình 4.13: Kết quả phân tích chuỗi tĩnh và giải mã từ FLOSS

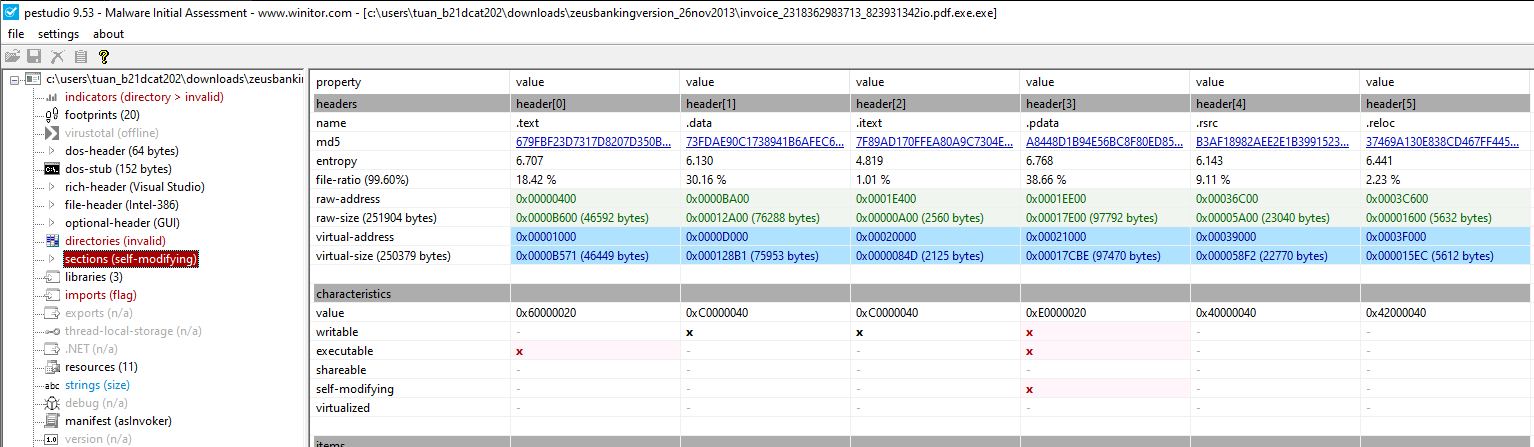
1. **Xác định các cách che giấu của tập tin**



##### Hình 4.14: Phân tích thông tin chi tiết tệp thực thi bằng Exeinfo PE

* Tên file: invoice\_231836298713\_823931342io.pdf.exe
* Đáng chú ý là tên file kết thúc bằng ".pdf.exe", có thể là một nỗ lực ngụy trang file thực thi dưới dạng tài liệu PDF.
* Entry Point: 0000A3B6 (Đây là địa chỉ bắt đầu thực thi của chương trình)
* EP Section: .text (Section chứa entry point là .text, thường chứa mã thực thi chính)
* File Offset: 00009786 (Vị trí bắt đầu của entry point trong file)
* File Size: 0003C00h (245,760 bytes) (Kích thước tổng thể của file)
* Subsystem: Windows GUI (File này là ứng dụng đồ họa Windows, không phải console)
* Linker Info: 10.00 (Phiên bản linker được sử dụng.)
* Image type: 32bit executable (File là thực thi 32-bit.)
* Compiler: Microsoft Visual C++ v7.10 EXE (File được biên dịch bằng Visual Studio 2010)
* RES/OVL: 9 / 0 % (Có 9 tài nguyên trong file, chiếm 0% kích thước file)
* Packer info: Not packed (File không bị nén hoặc đóng gói)
* Debug info: Có thông tin debug, có thể được sử dụng để phân tích sâu hơn.
* Các yếu tố đáng ngờ:
  + Tên file kết thúc bằng ".pdf.exe"
  + Sự hiện diện của thông tin debug trong một file invoice
  + Kích thước file tương đối lớn cho một file hóa đơn đơn giản

## **Nhận biết mã độc có bị nén hay mã hóa hay không**

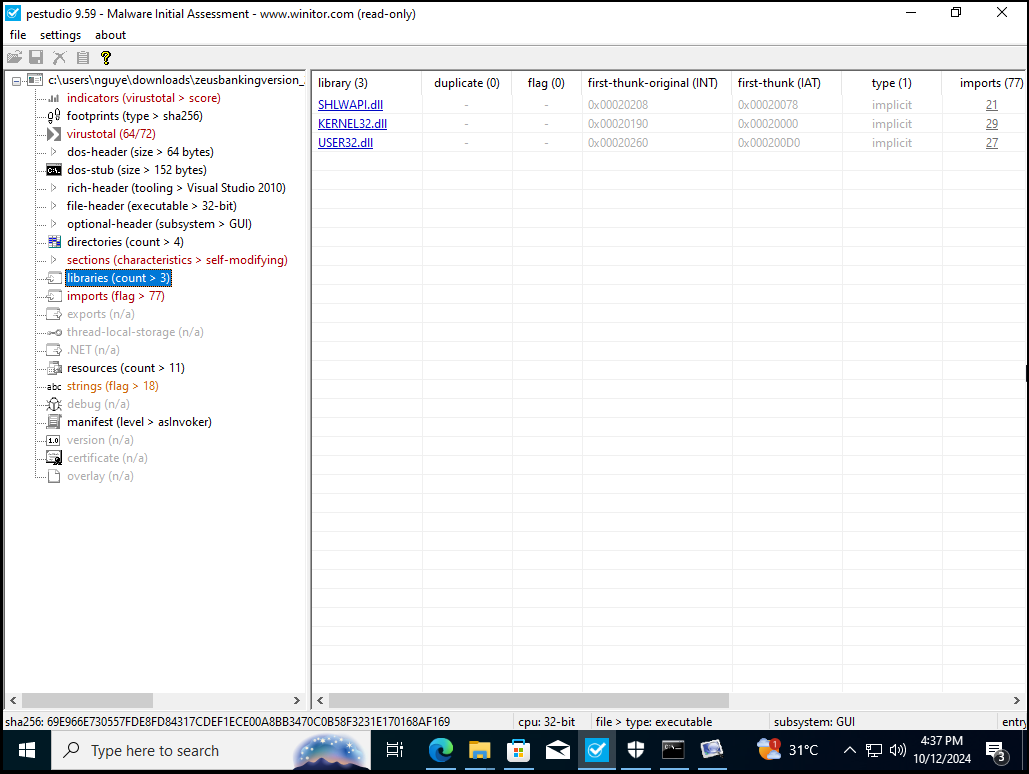


##### Hình 4.15: Phân tích các phần của tệp thực thi bằng PEStudio

- Ta kiểm tra kích thước của raw-size và virtual-size. Nếu kích thước chênh lệch quá lớn thì có khả năng nó đăng được đóng gói hoặc mã hóa.

- Nhưng ở đây ta thấy độ chênh lệch không quá lớn nên không có đóng gói hoặc mã hóa.

## **7. Tập tin PE**



##### Hình 4.16: Phân tích thư viện và hàm nhập của tệp thực thi trên PEStudio

- Tại đây ta thấy chương trình có sử dụng các thư viện động: ta có thể đưa ra một sổ giả thuyết liên quan đến việc mã độc có thể thực viễn những hành vi độc hại nào ảnh hưởng đến nạn nhân cũng như cách thức ẩn mình của mã độc

● KERNEL32.DLL: Chứa các hàm tiện ích nhẹ cho giao diện người dùng và các hoạt động liên quan đến chuỗi, đường dẫn, đăng ký và URL.

● USER32.DLL: Chứa các hàm cốt lõi của hệ điều hành Windows, như quản lý bộ nhớ, xử lý, tài nguyên, đồng bộ hóa, xử lý ngoại lệ và nhập xuất. Thư viện này là một phần không thể thiếu của hầu hết các ứng dụng Windows và nếu nó bị hỏng hoặc thiếu, máy tính có thể gặp phải một số lỗi nghiêm trọng.

● SHLWAPI.DLL: Là thư viện chứa các hàm liên quan đến giao diện người dùng của Windows. Như tạo và quản lý cửa sổ, menu, thanh cuộn, nút, hộp thoại bàn phím, chuột và các thông báo. Thư viện này là một phần không thể thiếu của hầu hết các ứng dụng Windows và nếu nó bị hỏng hoặc thiếu, máy tính có thể gặp phải các lỗi khi hiển thị hoặc tương tác với các ứng dụng. - Đối với việc Zeus Banking Trojan có thể gây ra những hành vi độc hại:

● Zeus banking trojan có thể sử dụng các hàm SHLWAPI.dll để thao tác với các đường dẫn, đăng ký và URL, ví dụ như tạo ra các đường dẫn giả mạo, thay đổi các giá trị đăng ký hoặc kết nối với các máy chủ điều khiển từ xa.

● Zeus banking trojan có thể sử dụng các hàm KERNEL32.dll để thực hiện các hoạt động cốt lõi của hệ điều hành, ví dụ như để cấp phát bộ nhớ, tạo ra các tiến trình con, ghi đè các hàm hệ thống hoặc gọi các hàm API khác.

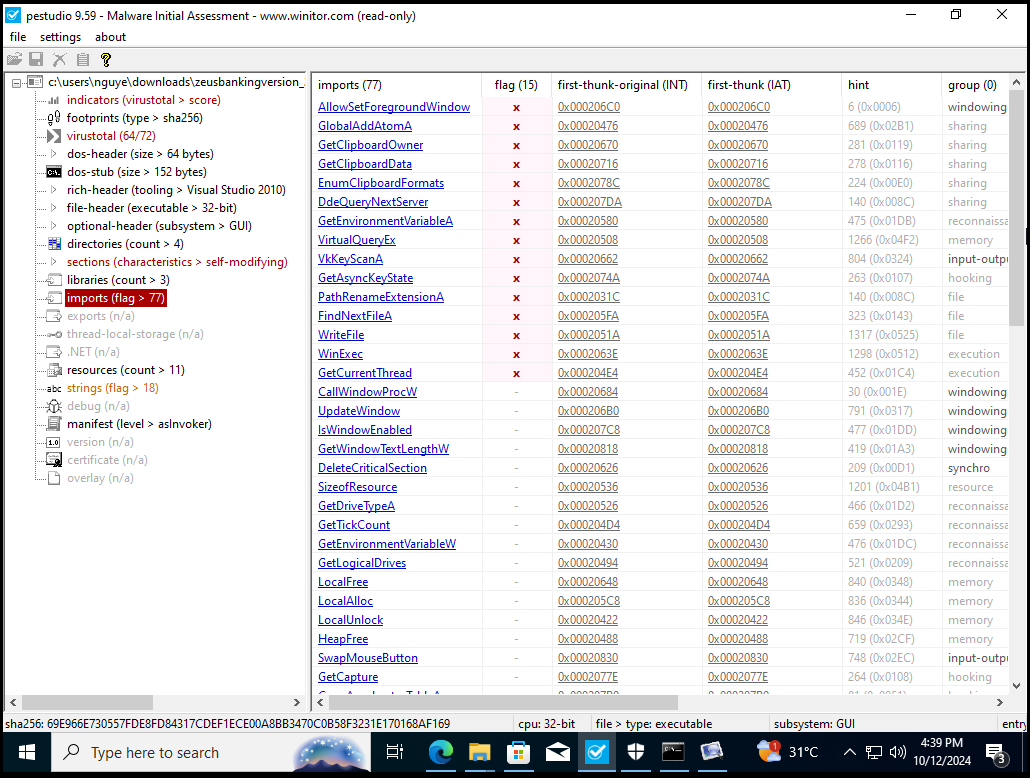
● Zeus banking trojan có thể sử dụng các hàm USER32.dll để thao tác với giao diện người dùng, ví dụ như theo dõi các sự kiện bàn phím, chuột, cửa sổ, hàn mình hoặc hộp thoại, hoặc để hiển thị các thông báo giả mạo để thu thập các thông tin nhạy cảm từ người dùng.

● Zeus banking trojan có thể sử dụng các hàm SHLWAPI.dll để ẩn mình trong các đường dẫn, đăng ký hoặc URL, ví dụ như đổi tên, mã hóa hoặc che dấu các tệp của nó, hoặc để sử dụng các tên mình động hoặc các địa chỉ IP ngẫu nhiên.

● Zeus banking trojan có thể sử dụng các hàm KERNEL32.dll để ẩn mình khỏi các hoạt động cốt lõi của hệ điều hành, ví dụ như đê chèn mã của nó vào các tiến trình hợp lệ, sửa đổi các bảng nhập xuất, hoặc tạo ra các kỹ thuật rootkit.

● Zeus banking trojan có thể sử dụng các hàm USER32.dll để ẩn mình trong giao diện người dùng, ví dụ như để giả lập các ứng dụng, trang web hoặc thông báo hợp lệ hoặc để ẩn các cửa sổ, menu hoặc hộp thoại của nó.

- Kiểm tra các imports có thể



##### Hình 4.17: Phân tích thư viện và hàm nhập của tệp thực thi trên PEStudio

- Ta có thể chú ý đến một số imports quan trọng: các imports này thường liên quan tới việc truy cập tệp tin, bộ nhớ, thông tin cửa sổ, ghi lại thao tác của người dùng, thực hiện ẩn mình, mở cửa hậu…

- Những hàm mà con mã độc này sử dụng để thực hiện ẩn mình hoặc có các hành vi gây cản trở quá trình phát hiện bởi các công cụ phát hiện mã độc:

● KERNEL32.SetCurrentDirectoryA để thay đổi thư mục làm việc hiện tại của tiến trình mã độc sang một thư mục khác, thường là thư mục của một ứng dụng hợp pháp. Điều này giúp mã độc tránh bị phát hiện bởi các công cụ kiểm tra thư mục làm việc của các tiến trình đang chạy.

● KERNEL32.VirtualQuery để lấy thông tin về các trang bộ nhớ được sử dụng bởi tiến trình mã độc, sau đó sử dụng hàm KERNEL32.ConvertDefaultLocale để mã hóa các trang bộ nhớ đó bằng một thuật toán XOR. Điều này giúp mã độc tránh bị phát hiện bởi các công cụ kiểm tra bộ nhớ của các tiến trình đang chạy.

● KERNEL32.CreateIoCompletionPort để tạo một cổng hoàn thành I/O, sau đó sử dụng hàm KERNEL32.OpenFileMappingA để mở một ánh xạ tệp được tạo bởi một tiến trình khác. Điều này giúp mã độc tránh bị phát hiện bởi các công cụ kiểm tra các tệp được mở bởi các tiến trình đang chạy.

● KERNEL32.DeleteFileA để xóa tệp thực thi của mã độc sau khi nó được chạy. (Điều này giúp mã độc tránh bị phát hiện bởi các công cụ kiểm tra các tệp có nghi ngờ trên đĩa cứng)

● SHLWAPI.PathMakeSystemFolderW để tạo một thư mục hệ thống ẩn, sau đó sử dụng hàm SHLWAPI.PathAddExtensionA để thêm một phần mở rộng tệp giả vào tên của thư mục đó. Điều này giúp mã độc lừa nạn nhân vào việc mở thư mục đó như một tệp thực thi, từ đó kích hoạt mã độc.

● USER32.GetShellWindow để lấy cửa sổ của shell Windows, sau đó sử dụng hàm USER32.GetPropW để lấy các thuộc tính của cửa sổ đó. Điều này giúp mã độc lấy được các thông tin về phiên làm việc của nạn nhân, như tên người dùng, tên máy tính, phiên bản hệ điều hành, v.v.

● USER32.SetDlgItemTextW để thay đổi nội dung của các hộp thoại trên các trang web của các ngân hàng hoặc các dịch vụ thanh toán trực tuyến.

● USER32.GetMonitorInfoW để lấy thông tin về kích thước và độ phân giải của màn hình, sau đó sử dụng hàm USER32.GetUpdateRgn để lấy vùng cập nhật của một cửa sổ. Điều này giúp mã độc chụp ảnh màn hình của nạn nhân, từ đó thu thập thêm các thông tin về hoạt động của nạn nhân trên internet.

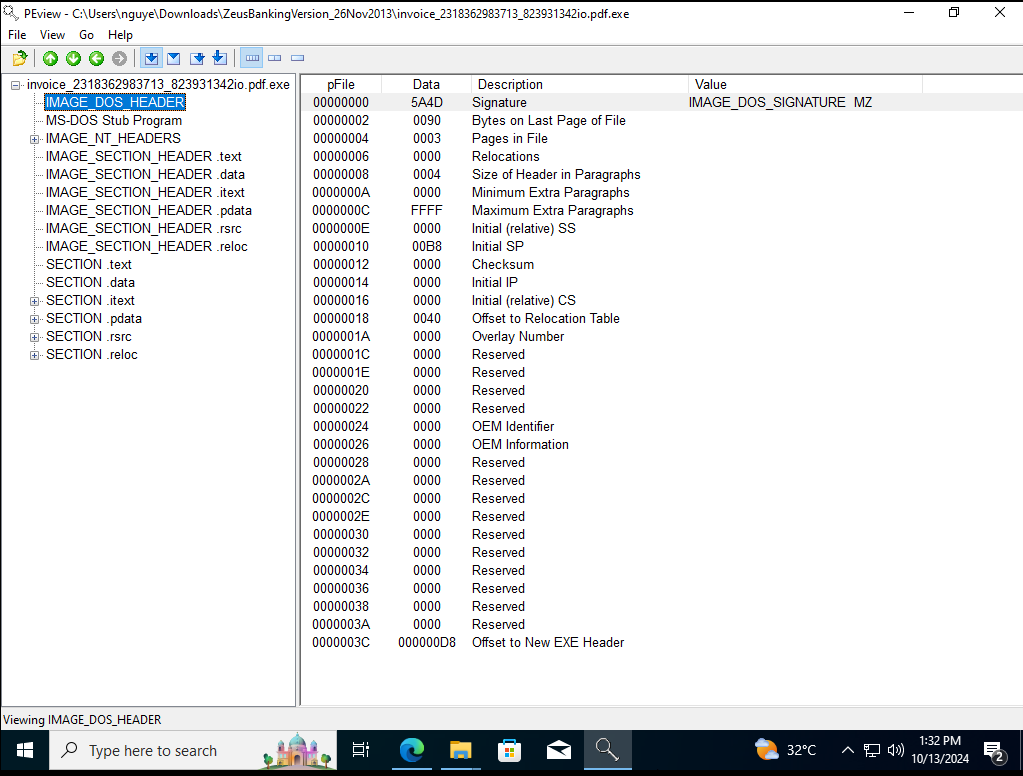
● KERNEL32.OpenFileMappingA: Hàm này cho phép mã độc mở một ánh xạ tệp được tạo bởi một tiến trình khác, có thể là một cửa hậu đã được cài đặt trước đó. Hàm này giúp mã độc truy cập vào các tệp hoặc bộ nhớ được chia sẻ bởi cửa hậu, từ đó có thể gửi dữ liệu ra ngoài hoặc nhận lệnh từ kẻ tấn công.

● SHLWAPI.SHLockShared và SHLWAPI.SHFreeShared: Hai hàm này cho phép mã độc khóa và giải phóng một đối tượng được chia sẻ bởi nhiều tiến trình. Hàm này giúp mã độc đồng bộ hóa việc gửi và nhận dữ liệu qua cửa hậu, tránh xung đột hoặc mất mát dữ liệu.

● USER32.GetShellWindow: Hàm này cho phép mã độc lấy cửa sổ của shell Windows, là một cửa sổ ẩn chứa các icon và thanh tác vụ. Hàm này giúp mã độc tạo ra một cửa sổ con bên trong cửa sổ shell, từ đó có thể gửi và nhận dữ liệu qua mạng mà không bị phát hiện bởi người dùng hoặc các công cụ bảo mật.

● KERNEL32.PathRelativePathToW: dùng để tạo một đường dẫn tương đối từ một tệp hoặc đường thư mục khác, mã độc có thể sử dụng hàm này để xác định vị trí của các tệp hoặc thư mục quan trọng trên hệ thống nạn nhân như tệp cấu hình, tệp nhật ký, hoặc tệp chứa thông tin tài khoản ngân hàng, sau đó mã độc có thể đọc, ghi, hoặc xóa các tệp hoặc thư mục này để thực hiện các hành vi độc hại.

● KERNEL32.OpenFileMappingA: dùng để mở một đối tượng ánh xạ tệp haojcw không có tên cho 1 tệp chỉ định, Zeus Banking Trojan co thể sử dụng hàm này để tạo 1 vùng nhớ chia sẽ giữa các tiến trình, hoặc giữa các máy tính thông qua mạng, điều này giúp mã độc truyền dữ liệu vè cho kẻ tấn công hoặc lây nhiễm sang các máy tính khác.



##### Hình 4.18: Phân tích chi tiết tiêu đề DOS và các phần của tệp thực thi trên PEView

+ .text: Phần ".text" chứa mã máy (machine code) của chương trình, đây là phần của tệp thực thi chứa các lệnh và mã nhị phân được thực thi bởi máy tính.

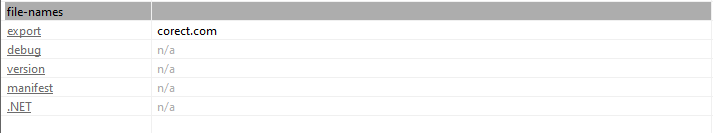
+ .data: Phần ".data" chứa dữ liệu tĩnh (static data) của chương trình, bao gồm biến toàn cục và các dữ liệu được khai báo trong mã nguồn.

+ .itext: Phần ".itext" (instruction text) chứa mã máy của chương trình nhưng được chỉ đọc (read-only), nghĩa là nó không thể bị thay đổi trong quá trình thực thi.

+ .pdata: Phần ".pdata" (procedure data) chứa các thông tin liên quan đến các hàm và quy trình trong chương trình. Nó bao gồm các bản ghi (records) về các hàm, bao gồm địa chỉ bắt đầu và kết thúc của hàm, thông tin liên quan đến xử lý ngoại lệ (exception handling), và các thông tin khác để hỗ trợ quá trình gỡ lỗi và xử lý ngoại lệ.

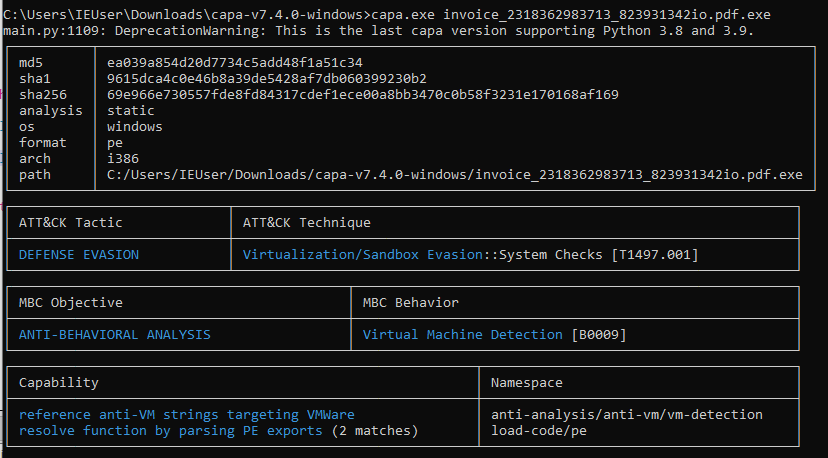
+ .rsrc: Phần ".rsrc" (resource) chứa các tài nguyên không phải mã máy, chẳng hạn như hình ảnh, biểu đồ, âm thanh, văn bản, biểu mẫu giao diện người dùng, và các tài liệu khác.

+ .reloc: Phần ".reloc" (relocation) chứa thông tin về việc tương đối (relocation) của mã máy.



Hình 4.19: Thông tin tên tệp xuất và các thành phần khác của tệp thực thi

Có thể đây là nơi dữ liệu bị đánh cắp gửi đến.



##### Hình 4.20: Phân tích kỹ thuật né tránh và phát hiện máy ảo bằng CAPA

* **Defense Evasion** là một **tactic** trong MITRE ATT&CK, mô tả các kỹ thuật mà mã độc hoặc kẻ tấn công sử dụng để **tránh bị phát hiện** bởi các hệ thống phòng thủ. Mục tiêu của **Defense Evasion** là che giấu hoạt động của mã độc, làm khó khăn cho các phần mềm bảo mật như antivirus, IDS/IPS, hoặc các hệ thống giám sát mạng phát hiện ra sự hiện diện của nó.
* **Virtualization/Sandbox Evasion::System Checks (T1497.001)** là một **technique** cụ thể thuộc tactic **Defense Evasion**, tập trung vào việc phát hiện xem môi trường thực thi của mã độc có phải là **ảo hóa** hoặc **sandbox** hay không.

#### Các hành vi điển hình trong System Checks:

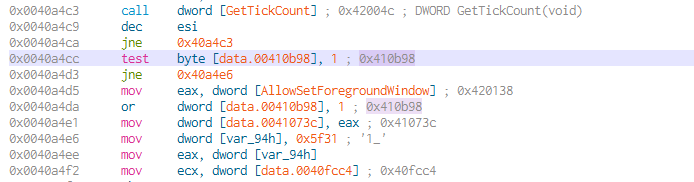
* Kiểm tra các chỉ số hệ thống (system indicators) như tên máy, tên người dùng, phần cứng, và driver để phát hiện xem mã độc đang chạy trên máy ảo (VM) hoặc trong môi trường sandbox.
* Nếu mã độc phát hiện rằng nó đang chạy trong môi trường giả lập, nó có thể dừng hoạt động, tự xóa hoặc thực hiện các hành động khác để tránh bị phát hiện.

**ANTI-BEHAVIORAL ANALYSIS** là một mục tiêu (**Objective**) trong **MBC**, mô tả các kỹ thuật mà mã độc sử dụng để **tránh bị phát hiện** hoặc **phân tích** bởi các công cụ và kỹ thuật phân tích hành vi. Phân tích hành vi (behavioral analysis) thường được thực hiện bởi các công cụ giám sát mã độc trong khi nó đang hoạt động. Mã độc sẽ tìm cách phá vỡ hoặc né tránh các phân tích này bằng cách:

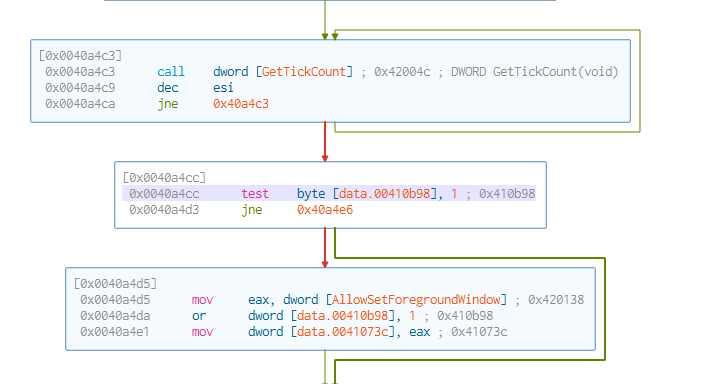
* Phát hiện các môi trường phân tích, như các máy ảo, sandbox, hoặc các công cụ ghi nhận hành vi.
* Thay đổi hành vi của chính nó, vô hiệu hóa hoặc thực hiện hành động khác khi phát hiện các môi trường này.

**Virtual Machine Detection (B0009)** là một **technique** cụ thể thuộc **ANTI-BEHAVIORAL ANALYSIS**, tập trung vào việc mã độc cố gắng phát hiện xem nó có đang chạy trong một **máy ảo** hay không.

Sử dụng Cutter sẽ hiển thị cho chúng ta các đoạn mã , giúp ta biết được mã độc này đang muốn làm gì.

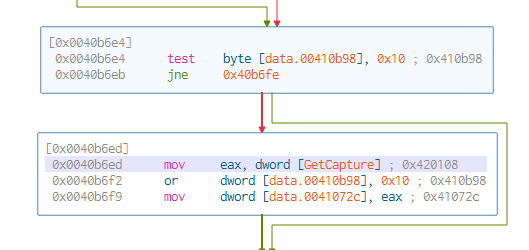


##### Hình 4.21: Phân tích mã máy: GetTickCount và AllowSetForegroundWindow



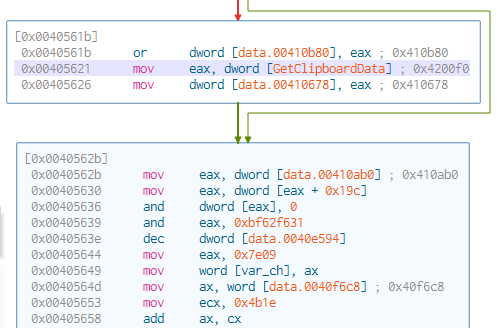
##### Hình 4.22: Biểu đồ dòng mã: GetTickCount và AllowSetForegroundWindow

* Hàm GetTickCount() kiểm tra xem máy Windows đã chạy được bao lâu. Đây có thể là hàm giúp cho mã độc phát hiện được ra môi trường có phải là môi trường ảo hay không và lẩn tránh.
* Đây cũng có thể là để mã độc vào chế độ ngủ đông, tránh gây việc theo dõi trong quá trình phân tích động.
* Sau đó một thời gian sẽ tự khởi động chạy hàm AllowSetForegroundWindow cho mã độc quyền chạy mà không bị giới hạn bởi các quy tắc an toàn trong hệ điều hành, vượt qua bảo mật hệ thống.



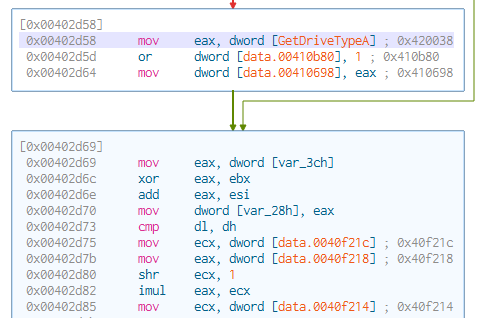
##### Hình 4.23: Biểu đồ dòng mã: GetCapture trong mã độc

* GetCapture có thể sử dụng để lấy cắp hình ảnh trên màn hình máy nhiễm độc, có khả năng để xem thông tin cá nhân hoặc thông tin đăng nhập của người dùng



##### Hình 4.24: Biểu đồ dòng mã: GetClipboardData trong mã độc

* GetClipBoardData có thể sử dụng để ăn cắp các thông tin liên quan đến việc copy trong quá trình người dùng sử dụng, nếu người dùng copy mật khẩu tài khoản thì mọi thông tin sẽ bị lấy cắp

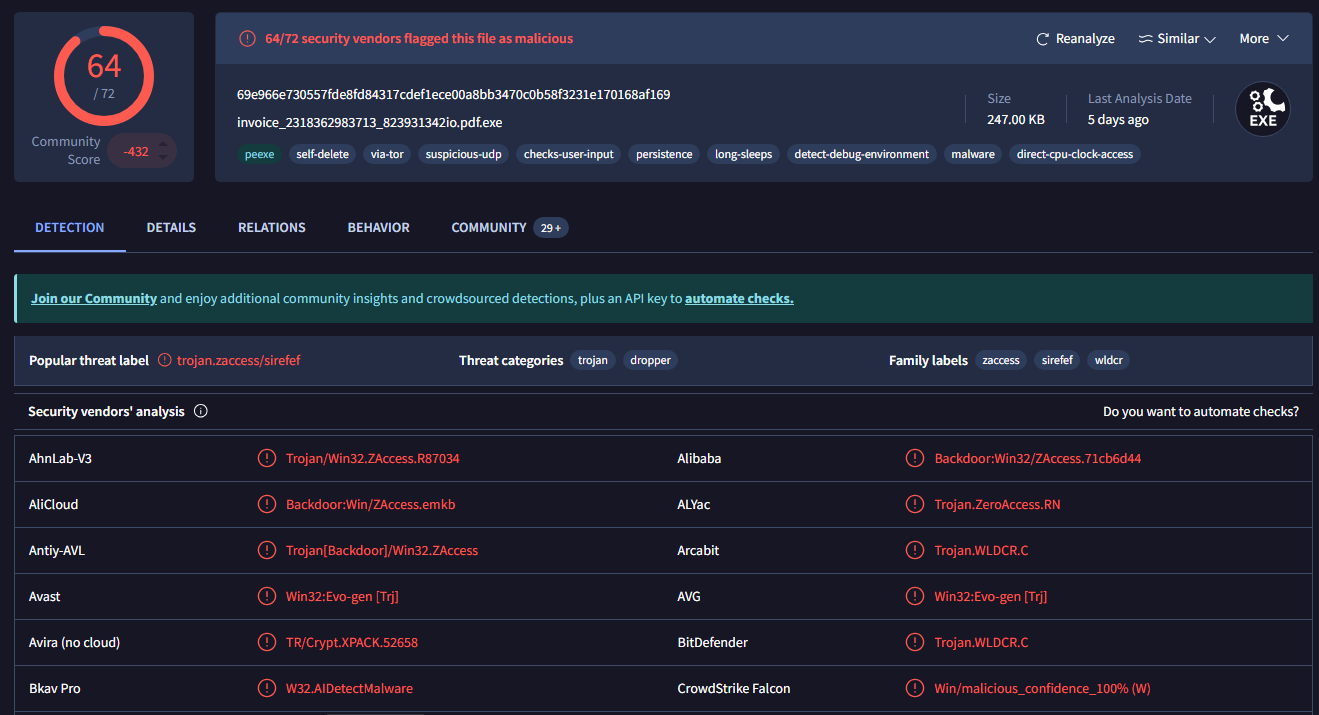


##### Hình 4.25: Biểu đồ dòng mã: GetDriveTypeA trong mã độc

* GetDriveType có thể dùng để phân biệt loại của drive trên máy tính bị nhiếm mã độc, khả năng nếu đấy là USB drive thì mã độc sẽ thực thi lnehj WriteFile để lây nhiễm qua USB rồi qua các máy khác qua con USB này

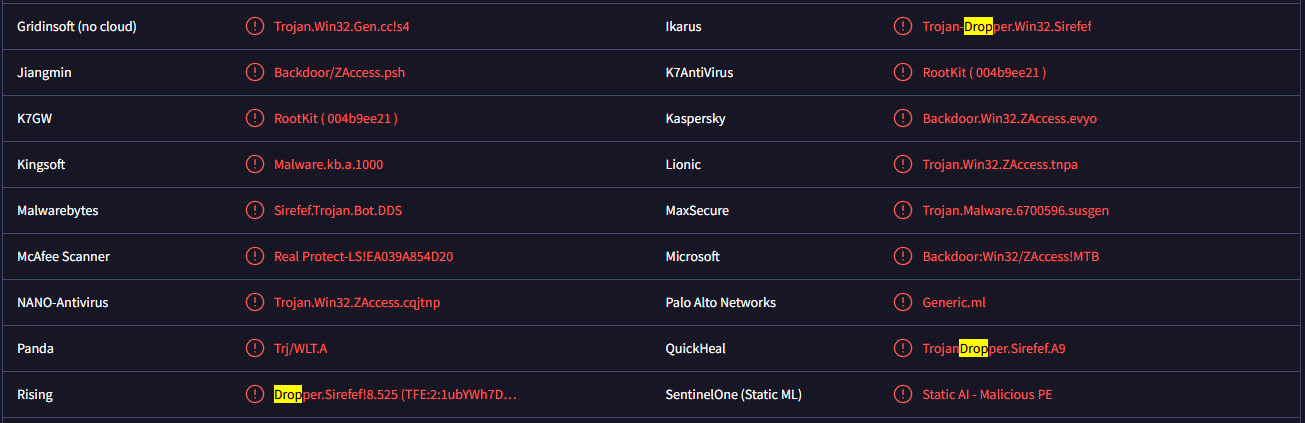
## **Phân loại**

* Phân loại dựa trên chữ ký: dựa vào virustotal, so sánh mã Hash hoặc các đoạn mã đặc trưng của tập tin nghi ngờ với cơ sở dữ liệu chứa chữ ký của các loại mã độc đã biết.



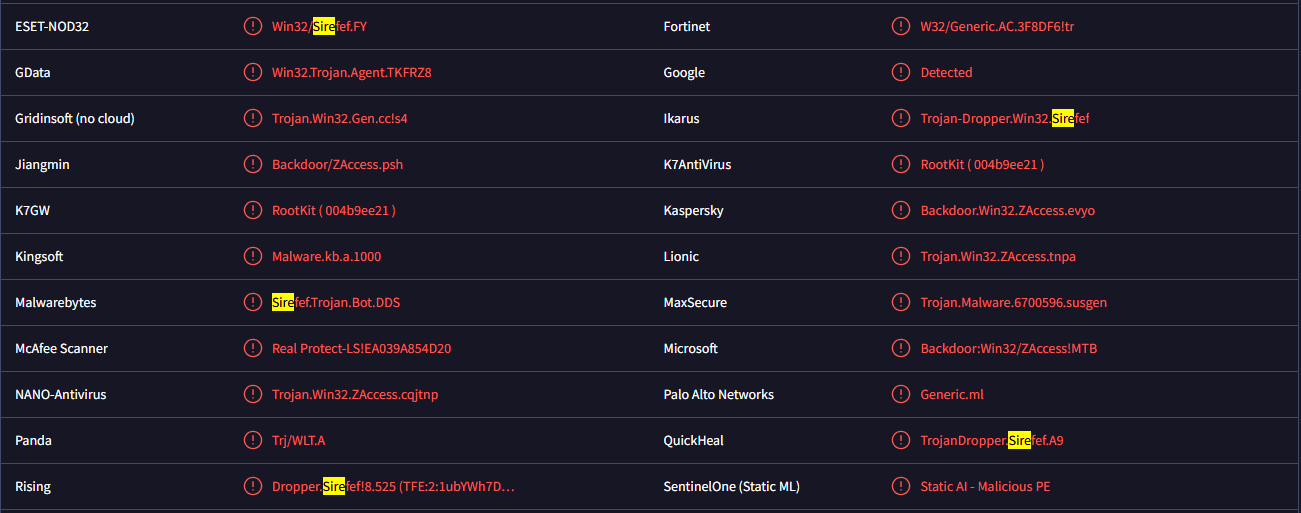
##### Hình 4.26: Kết quả phát hiện mã độc Trojan và Backdoor trên VirusTotal

=> Nếu có thể quét được, ta có thể biết được mã độc này thuộc loại nào, họ nào, thực hiện các hành vi độc hại nào trên máy tính của nạn nhân



##### Hình 4.27: Kết quả phát hiện mã độc Trojan và Backdoor trên VirusTotal

* Zeus banking trojan thực hiện các kỹ thuật như rootkit, polymorphism, obfusaction …, nó co thể tự xóa hoặc tự sao chép để trốn tránh các công cụ bảo mật, ngoài ra một số cái tên khác của mã độc này có thể kể đến:



##### Hình 4.28: Kết quả phát hiện mã độc Trojan và Backdoor trên VirusTotal

1. **TỔNG KẾT**

* Trong quá trình nghiên cứu và phân tích mã độc, đặc biệt là với trojan Zeus Banking, chúng ta đã làm rõ được cơ chế hoạt động cũng như cách thức mà loại mã độc này che giấu và thực hiện hành vi tấn công.
* Qua việc sử dụng các công cụ phân tích tĩnh như PEStudio, Cutter, và Floss, chúng tôi có thể phát hiện được các chuỗi mã độc ẩn, các hàm nhập API quan trọng, và cách thức mà trojan này thực hiện các thao tác như đánh cắp thông tin đăng nhập, ẩn mình và qua mặt các công cụ bảo mật.
* Zeus Banking Trojan là một dạng mã độc nguy hiểm, không chỉ nhắm đến các cá nhân mà còn tấn công vào các tổ chức tài chính lớn, gây thiệt hại đáng kể. Điều này đòi hỏi các biện pháp phòng ngừa mạnh mẽ hơn từ phía người dùng cũng như các tổ chức.
* Việc phát hiện và phân tích sớm mã độc đóng vai trò quan trọng trong việc ngăn chặn thiệt hại mà nó có thể gây ra.
* Báo cáo này cung cấp cái nhìn tổng quan về quy trình phân tích mã độc từ khâu thu thập thông tin, phân tích tĩnh, đến việc thử nghiệm thực tế với các công cụ hỗ trợ hiện đại.
* Hy vọng rằng, những kỹ thuật và công cụ được trình bày sẽ là nguồn tham khảo hữu ích cho các nhà nghiên cứu bảo mật trong việc đối phó với mã độc nói chung và các biến thể của trojan Zeus Banking nói riêng.

# TÀI LIỆU THAM KHẢO

[1] VirusTotal, "VirusTotal: Free Online Virus, Malware, and URL Scanner," [Online]. Available: https://www.virustotal.com. [Accessed: Oct. 18, 2024].

[2] M. Ochsenmeier, "PEStudio - Malware Analyzer," 2024. [Online]. Available: https://winitor.com. [Accessed: Oct. 18, 2024].

[3] Kaspersky Labs, "The Evolution of Banking Malware: A Look at Zeus Trojan," [Online]. Available: https://www.kaspersky.com/reports/zeus-trojan. [Accessed: Oct. 18, 2024].

[4] PGS.TS Lê H. Lập và Đ. T. Hồng, "Nghiên cứu kỹ thuật nhận dạng và phân tích mã độc," Luận văn, Học viện Công nghệ Bưu chính Viễn thông, 2016. [Online]. Available: http://dlib.ptit.edu.vn/handle/HVCNBCVT/2933. [Accessed: Oct. 18, 2024].

[5] M. Ochsenmeier, "PEStudio - Malware Analyzer," 2024. [Online]. Available: https://winitor.com. [Accessed: Oct. 18, 2024].