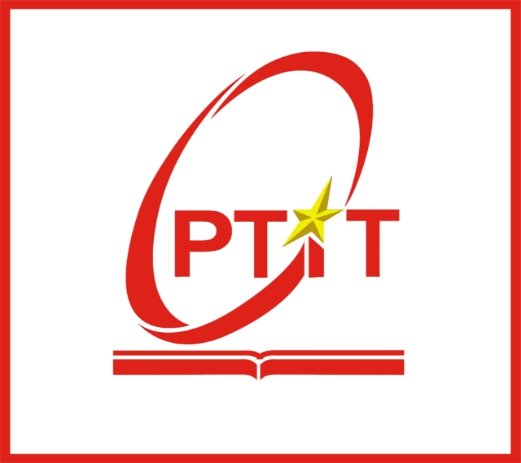


**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

**KHOA AN TOÀN THÔNG TIN**



**ptit-static-analys-linux**

Giảng viên: Đinh Trường Duy

1. **Bài thực hành phân tích tĩnh trên Linux**
   1. **Nội dung và hướng dẫn bài thực hành**
      1. ***Mục đích***

* Giúp sinh viên tìm hiểu mã, dữ liệu và các thành phần cấu trúc của phần mềm độc hại, đóng vai trò là tiền đề quan trọng để phân tích chi tiết hơn, sâu hơn.
  + 1. ***Yêu cầu với sinh viên***
* Có kiến thức cơ bản về hệ điều hành linux
  + 1. ***Nội dung thực hành***
* Khởi động bài lab
* Vào terminal gõ :

labtainer ptit-static-analys-linux

*( chú ý : sinh viên sử dụng mã sinh viên của mình để nhập thông tin email người thực hiện bài lab khi có yêu cầu để sử dụng khi chấm điểm)*

Sau khi khởi động xong một terminal ảo sẽ xuất hiện. Trên terminal thực hiện kiểm tra các file tệp có trong máy ảo

* Nhiệm vụ 1: Xác định thông tin ban đầu về mẫu phần mềm độc hại . Sử dụng công cụ file utility và hexdump để kiểm tra định dạng file của file mã độc có tên financials-xls.exe được đặt trong thư mục ubuntu trên máy ảo
* Để hoàn thành nhiệm vụ này sinh viên cần phải trích ra được định dạng bằng file utility và phần mô tả hexdump của file financials-xls.exe bằng cách sử dụng lệnh hexdump -C < tên tệp độc hại > | more
* Tại kết quả sau khi chạy lệnh hexdump sinh viên chỉ ra giá trị cho biết định dạng của file độc hại

*Để kiểm tra kết quả sinh viên thực hiện kiểm tra bằng checkwork tại mục file\_type*

* Nhiệm vụ 2 : Xác định mã nhận dạng, dấu vân tay duy nhất cho mẫu mã độc. Mã nhận dạng thường có dạng hàm băm mật mã - MD5, SHA1 hoặc SHA256.Lấy dấu vân tay được sử dụng cho nhiều mục đích, bao gồm:
* Xác định và theo dõi các mẫu phần mềm độc hại
* Quét toàn bộ hệ thống để tìm sự hiện diện của phần mềm độc hại giống hệt nhau
* Chia sẻ với các bên liên quan dưới dạng IoC hoặc như một phần của báo cáo thông tin về mối đe dọa
* Để hoàn thành nhiệm vụ này sinh viên cần sử dụng md5sum và sha256sum để kiểm tra hàm băm của financials-xls.exe

*Để kiểm tra kết quả sinh viên thực hiện bằng lệnh checkwork tại mục fingerprinting*

* Nhiệm vụ 3 :Kiểm tra hàm băm tệp được tạo ở bước trước đối với các máy quét phần mềm độc hại trực tuyến VirusTotal, một công cụ quét phần mềm độc hại trực tuyến, cộng tác với nhiều nhà cung cấp phần mềm chống vi-rút khác nhau, cho phép tìm kiếm hàm băm của tệp. IMPHASH, viết tắt của "Import Hash", là hàm băm mật mã được tính từ các chức năng nhập của tệp Windows Portable Executable (PE). Thuật toán của nó hoạt động bằng cách trước tiên chuyển đổi tất cả các tên hàm đã nhập thành chữ thường. Theo đó, tên DLL và tên hàm được hợp nhất với nhau và sắp xếp theo thứ tự bảng chữ cái. Cuối cùng, hàm băm MD5 được tạo từ chuỗi kết quả.

Chúng ta có thể tìm thấy IMPHASH trong Detailstab kết quả VirusTotal.

* Để hoàn thành nhiệm vụ này sinh viên cần sử dụng mô-đun Python [pefile](https://pypi.org/project/pefile/" \t "_blank) sau để tính toán thành công IMPHASH của tệp financials-xls.exe bằng cách tạo file imphash\_calc.py .

import sys

import pefile

import peutils

pe\_file = sys.argv[1]

pe = pefile.PE(pe\_file)

imphash = pe.get\_imphash()

print(imphash)

*Sau khi hoàn thành nhiệm vụ sinh viên kiểm tra kết quả tại mục imphash\_python*

* Nhiệm vụ 4: Fuzzy Hashing (SSDEEP), còn được gọi là băm từng phần được kích hoạt theo ngữ cảnh (CTPH), là một kỹ thuật băm được thiết kế để tính toán giá trị băm biểu thị sự giống nhau về nội dung giữa hai tệp. Kỹ thuật này chia tệp thành các khối nhỏ hơn, có kích thước cố định và tính toán hàm băm cho mỗi khối. Các giá trị băm thu được sau đó được hợp nhất để tạo ra hàm băm mờ cuối cùng.
* Để hoàn thành nhiệm vụ này sinh viên cần sử dụng ssdeep để kiểm tra hàm băm ssdeep của financials-xls.exe

Các đối số dòng lệnh -pb của SSDEEP có thể được sử dụng để bắt đầu chế độ khớp trong SSDEEP.

$ ssdeep -pb \*

* -p biểu thị chế độ Khá khớp và -b được sử dụng để chỉ hiển thị tên tệp, không có đường dẫn đầy đủ.
* Từ kết quả có được sinh viên rút ra được kết luận gì về các tệp?

*Sau khi hoàn thành nhiệm vụ sinh viên kiểm tra kết quả bằng checkwork tại mục fuzzy\_hashing.*

* Nhiệm vụ 5: Section hashing, (băm các phần PE) là một kỹ thuật mạnh mẽ cho phép các nhà phân tích xác định các phần của tệp (PE) đã được sửa đổi. Bằng cách áp dụng section hashing, các nhà phân tích bảo mật có thể xác định các phần của tệp PE đã bị giả mạo hoặc thay đổi. Điều này có thể giúp xác định các mẫu phần mềm độc hại tương tự, ngay cả khi chúng đã được sửa đổi một chút để tránh các phương pháp phát hiện dựa trên chữ ký truyền thống. Các công cụ như pefile trong Python có thể được sử dụng để thực hiện các công việc section hashing.
* Để hoàn thành nhiệm vụ này sinh viên sử dụng mô-đun pefile để truy cập và băm dữ liệu trong các phần riêng lẻ của tệp PE bằng việc chỉnh sửa file python section\_hashing.py có sẵn để hiển thị được các giá trị băm md5 và sha256

*Sau khi hoàn thành nhiệm vụ sinh viên kiểm tra bằng checkwork tại mục section\_hashing*

* Nhiệm vụ 6: Mục tiêu của nhiệm vụ này là trích xuất các chuỗi (ASCII & Unicode) từ tệp nhị phân. Các chuỗi có thể cung cấp manh mối và thông tin chi tiết có giá trị về chức năng của phần mềm độc hại. Đôi khi, chúng tôi có thể phát hiện các chuỗi nhúng duy nhất trong mẫu phần mềm độc hại, chẳng hạn như:
* Tên tệp được nhúng
* Địa chỉ IP hoặc tên miền
* Đường dẫn hoặc khóa đăng ký
* Các hàm API của Windows
* Đối số dòng lệnh
* Thông tin duy nhất có thể gợi ý về một tác nhân đe dọa cụ thể

Bên cạnh đó giải pháp phân tích chuỗi khác được gọi là FLOSS. FLOSS, viết tắt của "FireEye Labs Obfuscated String Solver", là một công cụ được phát triển bởi nhóm FLARE của FireEye để tự động giải mã các chuỗi trong phần mềm độc hại. Nó được thiết kế để bổ sung thay cho việc sử dụng các công cụ chuỗi truyền thống, như lệnh chuỗi trong các hệ thống dựa trên Unix, có thể bỏ sót các chuỗi bị xáo trộn thường được phần mềm độc hại sử dụng để tránh bị phát hiện.

* Để hoàn thành nhiệm vụ này sinh viên cần thực hiện lệnh strings hiển thị các chuỗi cho mẫu mã độc financials-xls.exe được đặt trong thư mục ubuntu bằng cách sử dụng lệnh

Strings -n -15 <tên tệp độc hại> .

Đồng thời cần sử dụng công cụ floss để hiển thị các chuỗi cho mẫu mã độc financials-xls.exe được đặt trong thư mục ubuntu

* Từ kết quả các chuỗi thu được kết hợp với kết quả từ nhiệm vụ 1 sinh viên trả lời câu hỏi liệu tệp độc hại financials-xls.exe đã bị pack hay chưa. Nếu có chỉ ra chuỗi chứng minh cho kết luận này.

*Sau khi hoàn thành nhiệm vụ sinh viên kiểm tra kết quả tại mục string\_check*

* Nhiệm vụ 7 : Trong phân tích tĩnh, có thể phát hiện tệp độc hại đã được nén hoặc làm xáo trộn bằng kỹ thuật được gọi là đóng gói. Đóng gói phục vụ một số mục đích:
* Nó làm xáo trộn mã, khiến việc phân biệt cấu trúc hoặc chức năng của nó trở nên khó khăn hơn.
* Nó làm giảm kích thước của tệp thực thi, giúp truyền nhanh hơn hoặc ít bị chú ý hơn.
* Nó làm cản trở các nỗ lực kỹ thuật đảo ngược truyền thống.

Điều này có thể làm giảm khả năng phân tích chuỗi vì các tham chiếu đến chuỗi thường bị che khuất hoặc bị loại bỏ. Kết quả là, tệp phần mềm độc hại trở nên khó phân tích hơn vì không thể quan sát trực tiếp mã gốc.

Một trình đóng gói phổ biến được sử dụng là Ultimate Packer for Executables (UPX).

* Để hoàn thành nhiệm vụ này sinh viên cần sử dụng công cụ upx để unpack tệp độc hại với lệnh upx -d -o <tên tệp lưu trữ> <tên tệp độc hại> sau đó sử dụng lệnh strings để trích xuất lại các chuỗi từ tệp sau khi giải nén
* Sinh viên rút ra được kết luận gì dựa trên kết quả thu được chuỗi trích xuất ra được thu ở bước 6 và sau unpack từ tệp độc hại có tên financials-xls.exe được đặt trong thư mục ubuntu .

*Sau khi hoàn thành nhiệm vụ sinh viên kiểm tra tại mục unpacking\_upx*

* Để thúc bài lab :
* Trên terminal đầu tiên sử dụng lệnh sau để kết thúc bài lab : stoplab ptit-static-anlys-linux
* Khi bài lab kết thúc, một tệp zip lưu kết quả được tạo và lưu vào một vị trí được hiển thị bên dưới stoplab.
* Khởi động lại bài lab:
* Trong quá trình làm bài sinh viên cần thực hiện lại bài lab, dùng câu lệnh:

*labtainer –r ptit-static-analys-linux*