

**1.1.**

**H**

**Ọ**

**C VI**

**Ệ**

**N CÔNG NGH**

**Ệ**

**BƯU CHÍNH VI**

**Ễ**

**N THÔNG**

**KHOA AN TOÀN THÔNG TIN**

H

ọ

c ph

ầ

n:

**Ki**

**ể**

**m th**

**ử**

**xâm nh**

**ậ**

**p**

**BÁO CÁO**

**CHUYÊN Đ**

**Ề**

**TH**

**Ự**

**C HÀNH**

**TÌM HI**

**Ể**

**U BÀI TH**

**Ự**

**C HÀNH**

**PTIT**

**-**

**Static**

**-**

**Danabot**

**Gi**

**ả**

**ng viên:**

Đinh Trư

ờ

ng Duy

**ớ**

**p:**

0

2

1

.

Ngô Anh Ng

ọ

–

2.

Nguy

ễ

ệ

–

3

.

Ph

ạ

m Xuân Hinh

–

4.

Chu Th

ả

o Mai

–

5.

Nguy

ễ

n Đ

ứ

c Trung

–

**Bài thực hành: Phân tích tĩnh mã độc Danabot trên Linux**

***ptit-static-danabot***

# 1.1.1. Nội dung và hướng dẫn bài thực hành

### 1.1.1.1. Mục đích

Bài lab này giúp sinh viên hiểu và thực hành các nhiệm vụ trong quy trình phân tích tĩnh.

Cụ thể, sinh viên sẽ phân tích mã độc Danabot trên nền tảng Linux với các nhiệm vụ bao gồm phân tích các đặc tính của file, phân tích mã hash, phân tích string, kiểm tra đóng gói và ẩn giấu, kiểm tra các import. Qua các nhiệm vụ, sinh viên sẽ tiếp cận và hiểu rõ về quy trình phân tích tĩnh đồng thời đưa ra xác định và dự đoán về hành vi của mã độc Danabot.

### 1.1.1.2. Yêu cầu đối với sinh viên

Có kiến thức cơ bản về hệ điều hành Linux, biết quy trình phân tích tĩnh, biết sử dụng công cụ Detect It Easy và Ghidra.

### 1.1.1.3. Nội dung thực hành

**Tìm hiểu về phân tích tĩnh:**

Phân tích tĩnh không yêu cầu thực thi mã độc và thường bao gồm các bước sau:

* Phân tích thuộc tính tập tin: Xác định tên tập tin, kích thước, dấu thời gian và loại tập tin.
* Phân tích băm: Tạo ra giá trị băm để so sánh và xác định sự trùng lặp với các mẫu mã độc đã biết.
* Phân tích chuỗi: Tìm kiếm các chuỗi văn bản có thể tiết lộ thông tin về chức năng của mã độc.
* Kiểm tra đóng gói và mã hóa: Phát hiện các kỹ thuật đóng gói hoặc mã hóa nhằm ẩn mã độc khỏi các công cụ phân tích.
* Giải gói: Nếu mã độc được đóng gói, cần thực hiện giải gói để có thể phân tích mã nguồn.
* Phân loại malware: Dựa vào các đặc điểm đã phân tích để xác định họ malware mà mã độc thuộc về.

**Giới thiệu mã độc Danabot:**

Danabot là một trojan ngân hàng được phát hiện thực tế vào năm 2018. Kể từ lần đầu tiên xuất hiện, Danabot đã nhận được sự yêu thích cao của tội phạm mạng và trở thành mối đe dọa tích cực ở nhiều khu vực trên thế giới. Mã độc này được viết bằng ngôn ngữ lập trình Delphi chứa một số tính năng chống phân tích, cũng như các mô-đun đánh cắp thông tin và điều khiển từ xa được cập nhật thường xuyên, làm tăng thêm sự đe dọa của nó đối với các mục tiêu.

**Giới thiệu công cụ:**

* Detect It Easy (DIE) là một chương trình đa nền tảng để xác định loại tệp và biết được tệp có bị mã hóa hay không với kiến trúc chữ ký hoàn toàn mở (MS-DOS, PE, ELF,MACH,..). Ngoài ra DIE ở chế độ nâng cao còn có thể cho biết nhiểu thông tin của tệp (loại packet, mã hash, entropy, hex, string, import,…). DIE có 3 phiên bản: basic version (die), lite version (diel) và console version ("diec"). Ở bài thực hành này sẽ sử dụng die và diec.
* Ghidra là một công cụ phân tích mã nguồn mở được phát triển bởi Cơ quan Tình báo Trung ương Hoa Kỳ (NSA). Được công bố công khai vào tháng 3 năm 2019, Ghidra cung cấp các tính năng mạnh mẽ để phân tích mã nhị phân và mã nguồn.

**Khởi động bài lab:**

* Vào terminal, gõ:

*labtainer ptit-static-danabot*

*(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 hai terminal ảo sẽ xuất hiện, một cái là ***die*** (với công cụ chính là Detect It Easy), một cái là ***ghidra.***

**Các nhiệm vụ:**

* **Task 1: Xác định loại tệp**.
  + Khởi động die chế độ GUI tại terminal ***die***: *die danabot\_sample.exe &*
  + Trên terminal ***die*** xem thông tin cơ bản của file: *diec -S Info danabot\_sample.exe*
  + Gõ lệnh quét loại của tệp và kiểm tra mã hóa (lệnh này tương ứng với chọn all types và chọn Scan tại giao diện): *diec -a danabot\_sample.exe*

➔ File mã độc trên thuộc loại mã độc nào và có kiểu mã hóa nào không?

- **Task 2: Phân tích mã hash**.

* Tại terminal ***die*** tính giá trị hash của file mã độc: diec -S Hash danabot\_sample.exe
* Tại giao diện chọn chế độ Advanced và chọn Hash để có thể xem thêm giá trị hash của các thành phần khác trong mã độc
* Phân tích mẫu với virustotal.com với 1 trong 2 cách
  + Cách 1: Sử dụng mã hash thu được bên trên vào phần tìm kiếm trên virustotal.com. Hoặc tại giao diện die chọn mục VirusTotal và sử dụng link tại cảnh báo hiện ở terminal.

• Tại trang virustotal trên trình duyệt. Ở phần Detection, sinh viên bôi đen chuột và copy tất cả các phát hiện và dán lưu vào 1 file mới <detecion> trên terminal.

* + Cách 2: Đăng ký tài khoản tại trang virultotal.com và lấy apikey. Tại giao diện die chọn Option -> Online tools -> Nhập apikey -> Ok. Sau đó sinh viên chọn mục VirusTotal tại giao diện die.
    - Chọn mục sẽ cho kết quả quét file với các phát hiện và chọn nút Save để lưu vào file <detecion>.
    - Sinh viên có thể chọn mục Website trên giao diện VirusTotal(on die) để lấy link tại cảnh báo hiện ở terminal

o Sau khi lưu các phát hiện vào file, kiểm tra có bao nhiêu phát hiện dự đoán mã độc là Danabot *grep “Dana” <detecion>*

➔ Có bao nhiêu phát hiện đây là mã độc Danabot?

- **Task 3: Phân tích string.**

* Tại terminal die, sinh viên tìm các chuỗi có trong chương trình: *strings -n 6 danabot\_sample.exe*
* Tại giao diện die, sinh viên chọn Strings để có thể sử dụng 1 số fillter và các kiểu định dạng để dễ tìm kiếm các chuỗi đáng ngờ

➔ Kết quả sau khi quét như thế nào (các chuỗi có thể đọc rõ như nào, số lượng các chuỗi khó đọc, địa chỉ ip tìm thấy và so sánh virustotal có địa chỉ ip đó không, qua đó rút ra kết luận gì) ?

- **Task 4: Kiểm tra mã độc packed và obfuscated.**

* Tính giá trị entropy của chương trình:

*diec -S Entropy danabot\_sample.exe*

* Tại giao diện die, sinh viên chọn Entropy để có thể xem rõ giá trị entropy của từng section
* Nhận xét giá trị entropy? Tại sao section .text có entropy cao? Tại sao ở nhiệm vụ 1 không phát hiện mã hóa nào nhưng ở đây die lại dự đoán tỉ lệ packed cao ?

o Phân tích obfuscated strings:

*floss -s danabot\_sample.exe --no-static-strings* Kết quả ra 1 số stackstring nhưng khó có thể đọc.

* Rút ra kết luận về việc section .text có chứa các stackstring và việc sử dụng nó để làm mờ, che giấu mã độc để thực hiện các hành vi đáng ngờ khác

- **Task 5: Phân tích tập tin PE.**

* Tại giao diện die, chọn Import để xem các import từ các dll có trong chương trình
* Chọn Save các import từ KERNEL32.dll vào file *<import>* và thực hiện hiển thị các import đó trong terminal: *cat <import>*
* Từ các import được tìm thấy kết hợp với mục Behavior trên virustotal.com, chỉ ra các import đáng ngờ có thể khớp với MITRE ATT&CK Tactics and Techniques trên virustotal.
* Nhận xét về khả năng của chương trình? Khả năng này có giống với Trojan không?

- **Task 6: Phân tích mã nguồn** o Phân tích mã nguồn là 1 nhiệm vụ khó khăn, đặc biệt với mã độc có khả năng che giấu là 1 đặc điểm chính của Trojan. Ở đây sinh viên chỉ cần thực hiện tìm ra đoạn mã có chứa nơi bị che giấu.

* Tại terminal ***ghidra***, sinh viên khởi động ./ghidra và thực hiện phân tích mã nguồn của file danabot\_sample.exe
* Để tìm nơi bị che giấu, sinh viên chọn vào section có entropy cao, và dùng chức năng Function Call Tree có trong ghidra. Để chắc chắn hãy kiểm tra xem fuction call tree nhận được có node gốc từ entry hoặc main (là các hàm khởi tạo thường thấy của chương trình)
* Sinh viên chọn lấy 1 function cách xa entry nhất trong function call tree nhận được và xem mã C

➔ Nhận xét về function nhận được (Số lượng giá trị khởi tạo, các vòng lặp và phép tính, số lượng hàm được gọi) ? Đưa ra dự đoán về khả năng của function này ?

o Tìm địa chỉ đầu và địa chỉ cuối của function có dạng 0x004d\_\_ và nhập vào terminal die với câu lệnh sau để hiển thị mã nguồn gốc của function đó. *objdump -D --section=< section\_name> --startaddress=<start\_address> --stop-address=<stop\_address> danabot\_sample.exe*

* **Task 7: Phân tích khả năng chính (Nâng cao)** o Thông qua các phân tích trên, chúng ta chỉ biết được 1 số thông tin và có thể rút ra đây là mã độc Trojan còn các khả năng chính của nó đã được ẩn giấu. Để phân tích rõ xem hoạt động của nó, tại terminal

đã có file danabot\_sample.dll là 1 file dll được tạo ra từ hoạt động ẩn giấu của chương trình chính, phân tích file dll này sẽ biết được nhiều hơn về khả năng của danabot.

* + Dùng lệnh strings tìm ra link virustotal.com được giấu trong o Scan nó trên giao diện die và chọn Import để xem các import được sử dụng.
  + So sánh các import và Behavior trên trang virustotal tìm được **Kết thúc bài lab:**
* Trên terminal đầu tiên sử dụng câu lệnh sau để kết thúc bài lab: *stoplab ptit-static-danabot*
* 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-danabot*