A black and white border

Description automatically generated-

A red circle with a yellow star in the middle

Description automatically generated

**TÌM HIỂU BÀI THỰC HÀNH**

**ptit-yara**

**Giảng viên:** Đỗ Xuân Chợ

**Họ và tên sinh viên:** Nguyễn Quốc Vượng

**Mã sinh viên:** B21DCAT227

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

**KHOA AN TOÀN THÔNG TIN**

**Mục lục**

[**1. Nội dung 3**](#_Toc180934168)

[**1.1. Mục đích 3**](#_Toc180934169)

[**1.2. Tìm hiểu lí thuyết 3**](#_Toc180934170)

[**1.3. Nội dung thực hành 4**](#_Toc180934171)

[**2. Checkwork: 15**](#_Toc180934172)

# Nội dung

## 1.1. Mục đích

* Giúp sinh viên hiểu về khái niệm cách sử dụng công cụ yara để phát hiện mã độc.

## 1.2. Tìm hiểu lí thuyết

- YARA (Yet Another Recursive Acronym) được phát triển bởi Victor Alvarez của VirusTotal, là một công cụ giúp xác định và phân loại phần mềm độc hại bằng cách sử dụng các mẫu hoặc quy tắc xác định. Các quy tắc này có thể dựa trên nhiều yếu tố, bao gồm chữ ký nhị phân, chuỗi văn bản, và các đặc điểm khác của tệp tin. Mục tiêu của YARA là giúp các nhà nghiên cứu xác định các mẫu và hành vi của phần mềm độc hại một cách dễ dàng và hiệu quả.

YARA đã trở thành một công cụ không thể thiếu trong kho vũ khí của các nhà nghiên cứu bảo mật và chuyên gia phân tích phần mềm độc hại. Trong bài viết này, chúng ta sẽ tìm hiểu chi tiết về công cụ này, cách thức hoạt động của nó và cung cấp một ví dụ cụ thể về cách viết và sử dụng quy tắc YARA.

- Cách thức hoạt động của YARA:  
YARA hoạt động dựa trên các quy tắc được viết bằng một ngôn ngữ đặc tả riêng. Mỗi quy tắc YARA bao gồm ba phần chính:

1. **Meta:** Thông tin mô tả về quy tắc, chẳng hạn như tên tác giả, ngày tạo, và mục đích của quy tắc.
2. **Strings:** Các chuỗi hoặc mẫu cần tìm trong tệp tin. Đây có thể là các chuỗi văn bản, chuỗi nhị phân, hoặc các chuỗi có dạng hex.
3. **Condition:** Điều kiện để xác định xem tệp tin có khớp với quy tắc hay không. Điều kiện này thường dựa trên sự xuất hiện của các chuỗi trong phần Strings.

- Cấu trúc file yara:

* Tiêu đề (Header): Bắt đầu bằng từ khóa "rule" và tên đặt cho luật. Ví dụ: "rule MyRule".
* Metadata: Thông tin bổ sung về luật như tác giả, mô tả, ngày tạo, ...
* Strings: Đây là phần quan trọng nhất của một file luật YARA. Nó chứa các mẫu chuỗi (string patterns) mà YARA sẽ tìm kiếm trong các file hoặc quá trình đích. Mỗi mẫu chuỗi bắt đầu bằng từ khóa "strings" và có thể chứa các biểu thức chính quy (regular expressions).
* Condition: Điều kiện xảy ra khi một mẫu chuỗi được phát hiện. Điều kiện được đặt trong dấu ngoặc nhọn và có thể sử dụng các toán tử logic như AND (&&), OR (||), NOT (!) để kết hợp nhiều mẫu chuỗi.
* Ví dụ về 1 luật của yara:

rule MyRule{

meta:

author = "John Doe"

description = "Detects a specific malware variant"

created\_date = "2022-01-01

strings:

$my\_string = "malware\_sample"

condition:

$my\_string

}

## 1.3. Nội dung thực hành

* Khởi động bài lab:
* Vào terminal, gõ:

*imodule https://github.com/mei3011/ptit-yara*

Labtainer ptit-yara

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

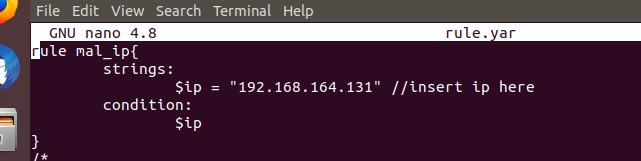
A screenshot of a computer

Description automatically generated

* Sau khi khởi động xong một terminal ảo sẽ xuất hiện, đại diện cho máy **yara**.
* Sinh viên làm các nhiệm vụ dưới đây để hoàn thành bài lab:
* **Nhiệm vụ 1: Đọc file “rule.yar” để xem các luật có sẵn**
* Trên terminal **yara** sử dụng lệnh “nano rule.yar” để xem các luật có sẵn.

nano rule.yar

* Yêu cầu sinh viên đọc hiểu các luật trong file rule.yar

1. 
2. Luật này kiểm tra xem file có chứa chuỗi IP cụ thể hay không. Nếu trong file xuất hiện chuỗi IP nà
3. y, luật sẽ khớp.
4. A computer screen shot of white text

   Description automatically generated\

Luật này khớp với bất kỳ file nào có chứa chuỗi "hello" ở định dạng ASCII (chuỗi văn bản mặc định không có ký tự đặc biệt)

1. A computer screen shot of white text

   Description automatically generated  
   Luật này khớp với bất kỳ file nào có chứa chuỗi "hello" ở định dạng Unicode. Từ khóa wide cho biết chuỗi đang kiểm tra có dạng mã hóa Unicode (UTF-16).
2. A computer screen shot of a code

   Description automatically generated  
   Luật này khớp với bất kỳ file nào chứa chuỗi hex cụ thể như { 0123456789ABCDEF }, { 0123456789abcdef } hoặc { 01 23 45 67 89 ab cd ef }.
3. A computer screen shot of a code

   Description automatically generated  
   Luật này kiểm tra các chuỗi hex với các ký tự đại diện ??, nghĩa là nó cho phép sự không xác định tại các vị trí này. Ví dụ, { 012345????ABCDEF } sẽ khớp với các byte bắt đầu bằng 012345 và kết thúc bằng ABCDEF, không quan trọng hai byte ở giữa là gì.
4. A computer code with white text

   Description automatically generated  
   Luật này khớp với bất kỳ file nào có chứa chuỗi "MZ" tại offset (vị trí) 0. Chuỗi "MZ" thường là dấu hiệu cho thấy file là định dạng PE (Portable Executable), như các file .exe
5. A computer screen shot of white text

   Description automatically generated  
   Luật này kiểm tra xem chuỗi "PE" có xuất hiện trong khoảng từ offset 32 đến 100 của file hay không. Điều này thường được sử dụng để phát hiện định dạng PE.
6. A computer screen shot of a program

   Description automatically generated  
   Luật này kiểm tra xem chuỗi "PE" có xuất hiện trong phạm vi 0x200 byte (512 byte) kể từ vị trí đầu tiên của chuỗi "MZ" tại offset 0. Đây là kiểm tra tương đối, nhằm xác định định dạng PE.
7. A black background with white text

   Description automatically generated  
   Luật này xác định xem một file có phải là file PE không. Nó kiểm tra xem chuỗi "MZ" có nằm tại offset 0 hay không và xác nhận rằng giá trị ở offset 0x3C là 0x4550 (hex cho "PE"). Đây là cách xác định một file .exe hợp lệ.
8. A computer screen shot of white text

   Description automatically generated  
   Luật này khớp với bất kỳ file nào chứa ít nhất 5 lần chuỗi "hello". Sử dụng cú pháp #hello để đếm số lần xuất hiện của chuỗi.
9. A computer screen with white text

   Description automatically generated  
   Luật này khớp với bất kỳ file nào chứa chuỗi "hello" và đồng thời cũng là file PE (được xác định bởi luật IsPeFile).
10. A computer screen shot of white text

    Description automatically generated  
    Luật này được định nghĩa là một **luật toàn cục**. Nếu bật lên, nó sẽ áp dụng giới hạn kích thước file là 2MB cho tất cả các luật. Nghĩa là các luật sẽ chỉ khớp với những file có kích thước nhỏ hơn 2MB.

* **Nhiệm vụ 2: Phát hiện IP độc hại trong file “example.txt” với yara**
* Sau khi xem qua các luật, mở file example.txt sẽ thấy có 1 địa chỉ ip, sinh viên cần thêm địa chỉ ip này vào luật trong file rule.yar.

A close up of a text

Description automatically generated

* Dùng yara để phát hiện ip trong file example.txt

A black background with white text

Description automatically generated

yara rule.yar example.txt  
A screenshot of a computer

Description automatically generated

* **Nhiệm vụ 3: Phát hiện chuỗi “HelLo wOrld” trong file “example.txt” với yara**
* Mở file rule.yar để viết luật phát hiện chuỗi “HelLo wOrld”

rule Nocase{

strings:

$s = “hello world” nocase

condition:

$s

}

A screenshot of a computer

Description automatically generated

* Lưu file và chạy lại yara với file example.txt

A black background with white text

Description automatically generated

* Câu hỏi: nocase này có tác dụng gì, nếu xóa đi sẽ có ảnh hưởng như thế nào đến luật này?

Khi sử dụng nocase trong luật YARA, nó sẽ làm cho việc kiểm tra chuỗi trở nên **không phân biệt chữ hoa chữ thường**. Điều này có nghĩa là chuỗi "hello world" sẽ khớp với tất cả các biến thể của nó như "HelLo wOrld", "HELLO WORLD", "hello WORLD", và các biến thể khác. Nếu xóa nocase, thì YARA sẽ thực hiện so sánh **có phân biệt chữ hoa chữ thường**. Nghĩa là, chỉ những chuỗi hoàn toàn khớp với "hello world" với tất cả các ký tự đều ở dạng chữ thường mới được phát hiện. Ví dụ, "HelLo wOrld" sẽ **không** được phát hiện nếu không có nocase.

* **Nhiệm vụ 4: Phát hiện được mẫu mã độc cho trước với yara**
* Sau khi hiểu cách hoạt động của yara, tiến hành thử nghiệm phát hiện trên mẫu mã độc malware.exe

A computer screen shot of white text

Description automatically generated

* Tìm kiếm các chuỗi đặc trưng của malware.exe

strings malware.exe

A screenshot of a computer

Description automatically generated

Sau khi tìm được các chuỗi đặc trưng của mẫu mã độc, viết luật để phát hiện malware.exe là mã độc (viết ít nhất 3 đặc trưng của mã độc):

rule MalwareDetection {

strings:

// Đặc trưng của tệp thực thi Windows

$dos\_string = "This program cannot be run in DOS mode."

// Sử dụng thư viện hệ thống phổ biến trong mã độc

$kernel\_dll = "KERNEL32.DLL"

// URL có thể là điểm kết nối của mã độc

condition:

// Phát hiện khi có ít nhất 2 trong 3 chuỗi đặc trưng

($dos\_string, $kernel\_dll, $malware\_url)

}

A computer code with white text

Description automatically generated  
-Tiến hành chạy rule:

A black background with white text

Description automatically generated

* Kết thúc bài lab:
* Trên terminal đầu tiên sử dụng câu lênh sau để kết thúc bài lab:

stoplab yara

* 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.

# Checkwork:

A computer screen with text

Description automatically generated