HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

KHOA CÔNG NGHỆ THÔNG TIN

Báo cáo đề tài

NHÓM 19

Môn: An toàn mạng

SINH VIÊN THỰC HIỆN :

NGUYỄN THỊ ÁNH NGUYỆT	**MSSV:N18DCAT054**
TRẦN BẢO TOÀN	**MSSV:N18DCAT076**
PHAN THỊ NHƯ Ý	**MSSV:N18DCAT106**
NGUYỄN THỊ LAN VY	**MSSV:N18DCAT10**

MỤC LỤC

BẢNG PHÂN CÔNG

Họ và tên	MSSV	Công việc
Trần Bảo Toàn	N18DCAT076	-Tìm hiểu lí thuyết về sql injection và dvwa -Viết rules phát hiện tấn công -Quay video thực hiện
Nguyễn Thị Lan Vy	N18DCAT104	-Tìm hiểu lí thuyết về thuyết về sql injection và dvwa -Viết rules phát hiện tấn công -Viết word báo cáo
Phan Thị Như Ý	N18DCAT106	Tìm hiểu lí thuyết về thuyết về brute force và dvwa -Viết rules phát hiện tấn công -Viết word báo cáo
Nguyễn Thị Ánh Nguyệt	N18DCAT054	Tìm hiểu lí thuyết về thuyết về brute force và dvwa -Viết rules phát hiện tấn công -Viết word báo cáo

Chương 1 : GIỚI THIỆU VỀ DVWA, TẤN CÔNG BRUTE FORCE VÀ SQL INJECTION

1. Giới thiệu về dvwa

- Damn Vulnerable Web Application (DVWA) là một ứng dụng mã nguồn PHP/MySQL tập hợp sẵn các lỗi logic về bảo mật ứng dụng web trong mã nguồn PHP. Lỗi logic khi lập trình có thể áp dụng đối với các loại ngôn ngữ lập trình nhằm giảm thiểu khả năng tạo ra lổ hổng bảo mật từ tư duy lập trình chưa cẩn thận. Mục tiêu chính của DVWA đó là tạo ra một môi trường thực hành hacking/pentest hợp pháp. Giúp cho các nhà phát triển ứng dụng web hiểu hơn về hoạt động lập trình an toàn và bảo mật hơn. Bên cạnh đó DVWA cũng cung cấp cho các thầy cô/học sinh phương pháp học và thực hành tấn công khai thác lỗi bảo mật ứng dụng web ở mức cơ bản và nâng cao.

- Các lỗ hổng của dvwa
  - Brute Force
  - Command injection
  - Cross Site Request Forgery (CSRF)
  - File Inclusion
  - SQL Injection
  - File Upload
  - XSS
- Các mức độ bảo mật trong dvwa
  - High : level này gần như là level dùng để so sánh mã nguồn có lổ hổng ở mức ‘low’ và ‘medium’ với mã nguồn đã được tối ưu ở mức an toàn bảo mật. Mức độ ‘high’ sẽ được đánh giá là có thể bao quát phần nhiều lổ hổng ở nhóm mục bạn đang thực hành.
  - Medium : mức độ này cung cấp nội dung logic code đã fix lổ hổng cơ bản ở hạng mục mức ‘low’. Nhưng liệu bạn vẫn có thể khai thác được thêm hay không?
  - Low : mức độ thấp nhất trong thang level bảo mật mà DVWA cung cấp đến các bạn. Với mức độ ‘low’ thì mã nguồn PHP gân như phơi bày khả năng khai thác lổ hổng qua tư duy lập trình chưa bao quát vấn đề bảo mật.

2. Giới thiệu về tấn công brute force:

- Tấn công brute-force là một phương pháp bẻ khóa phổ biến . Một cuộc tấn công brute-force liên quan đến việc 'đoán' tên người dùng và mật khẩu để truy cập trái phép vào hệ thống, hacker sẽ sử dụng phương pháp thử và sai để cố gắng đoán thông tin đăng nhập hợp lệ. Ngoài ra ta có thể sử dụng brute-force để khai thác OTP, Timestamp , Cookie, vv..

- Các cuộc tấn công này thường được tự động hóa bằng cách sử dụng danh sách các từ gồm tên người dùng và mật khẩu thường được sử dụng để có thể đạt được kết quả tốt nhất. Việc sử dụng các công cụ chuyên dụng có khả năng cho phép hacker thực hiện đăng nhập 1 cách tự động nhiều lần với tốc độ cao.

3. Giới thiệu về SQL injection

- SQL Injection là một kỹ thuật lợi dụng những lỗ hổng về câu truy vấn của các ứng dụng. Được thực hiện bằng cách chèn thêm một đoạn SQL để làm sai lệnh đi câu truy vấn ban đầu, từ đó có thể khai thác dữ liệu từ database. SQL injection có thể cho phép những kẻ tấn công thực hiện các thao tác như một người quản trị web, trên cơ sở dữ liệu của ứng dụng.

Chương 2: CƠ SỞ LÝ THUYẾT

1. Giới thiệu về Suricata

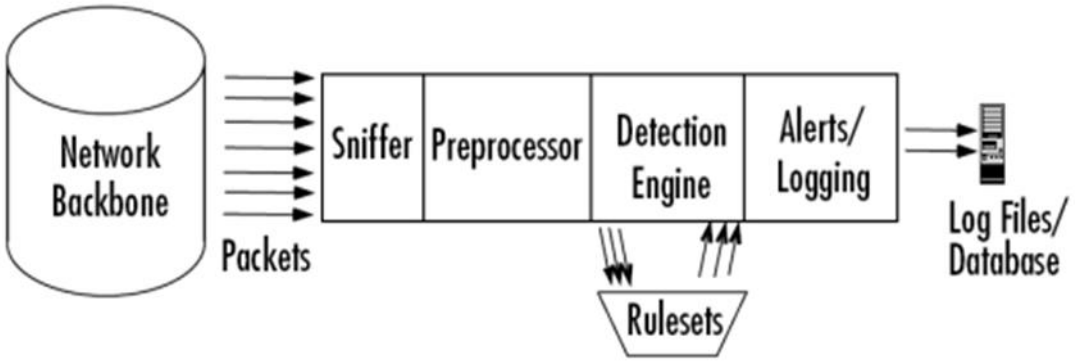
- Suricata là một hệ thống phát hiện xâm nhập dựa trên mã nguồn mở. Nó được phát triển bởi Open Information Security Foundation (OISF). Một phiên bản beta đã được phát hành vào tháng 12 năm 2009, bản chuẩn đầu tiên phát hành tiếp theo vào tháng 7 năm 2010.
- Suricata là công cụ IDS/IPS dựa trên luật để theo dõi lưu lượng mạng và cung cấp cảnh báo đến người quản trị hệ thống khi có sự kiện đáng ngờ xảy ra. Nó được thiết kế để tương thích với các thành phần an ninh mạng hiện có. Bản phát hành đầu tiên chạy trên nền tảng linux 2.6 có hỗ trợ nội tuyến (inline) và cấu hình giám sát lưu lượng thụ động có khả năng xử lý lưu lượng lên đến gigabit. Suricata là công cụ IDS/IPS miễn phí trong khi nó vẫn cung cấp những lựa chọn khả năng mở rộng cho các kiến trúc an ninh mạng phức tạp nhất.

- Suricata là phần mềm mã nguồn mở: Suricata được phát hành dưới giấy phép GNU/GPL điều này có nghĩa là bất cứ ai cũng có thể sử dụng Suricata một cách miễn phí dù đó là doanh nghiệp hay người dùng cá nhân. Ngoài ra vì là phần mềm mã nguồn mở nên Suricata có một cộng đồng người sử dụng lớn, sẵn sàng hỗ trợ nếu có bất cứ thắc mắc gì.
- Chạy trên nhiều nền tảng khác nhau: Chạy trên các hệ điều hành nguồn mở như Linux, CentOS. Debian, Fedora, FreeBSD, Window, Mac OS X (10.5.8 and 10.6.8)
- Là một công cụ đa luồng, suricata cung cấp tăng tốc độ và hiệu quả trong việc phân tích lưu lượng mạng. Ngoài việc tăng hiệu quả phần cứng (với phần cứng và card mạng giới hạn), công cụ này được xây dựng để tận dụng khả năng xử lý cao được cung cấp bởi chip CPU đa lõi mới nhất.
- Suricata làm việc như thế nào, tập tin cấu hình ở đâu, các luật như thế nào người quản trị đều có thể biết và cấu hình theo ý mình được. Kể cả việc tạo ra các luật mới.
- Luật của Suricata thường xuyên được cập nhật: Các luật của Suricata thường xuyên được bổ sung và cập nhật các hình thức xâm nhập mới

2. Cấu trúc của Suricata

- Suricata được phát triển dựa trên Snort nên nó vẫn giữ nguyên kiến trúc bên trong của Snort. Kiến trúc của nó có nhiều thành phần, với mỗi thành phần có một chức năng riêng.
- Kiến trúc của Suricata gồm 4 phần cơ bản sau:
    - The Sniffer (Packet Decoder) .
    - The Preprocessors.
    - The Detection Engine.
    - The Output: gồm hai modules

- Modul Alert/ Logging
- Modul kết xuất thông tin



Hình 1. Kiến trúc của Suricata

– Cách hoạt động

+ Khi Suricata hoạt động nó sẽ thực hiện lắng nghe và thu bắt tất cả các gói tin nào di chuyển qua nó.
+ Các gói tin sau khi bị bắt được đưa và module Sniffer, tại đây các gói tin sẽ được giải mã
+ Tiếp theo gói tin sẽ được đưa vào module Preprocessor, tại đây gói tin sẽ được phân tích một cách chi tiết và đầy đủ theo các cách khác nhau
+ Sau khi phân tích xong các gói tin được đưa vào module Detection. Tại đây, tùy theo việc có phát hiện được xâm nhập hay không mà gói tin có thể được lưu thông tiếp hay được đưa vào module Alert/ Logging để xử lý.
+ Khi các cảnh báo được xác định module kết xuất thông tin sẽ thực hiện việc đưa cảnh báo ra theo đúng định dạng mong muốn.

+ Preprocessors, detection engine và alert system đều là các plug-ins. Điều này giúp cho việc chỉnh sửa hệ thống theo mong muốn của người quản trị một cách dễ dàng.

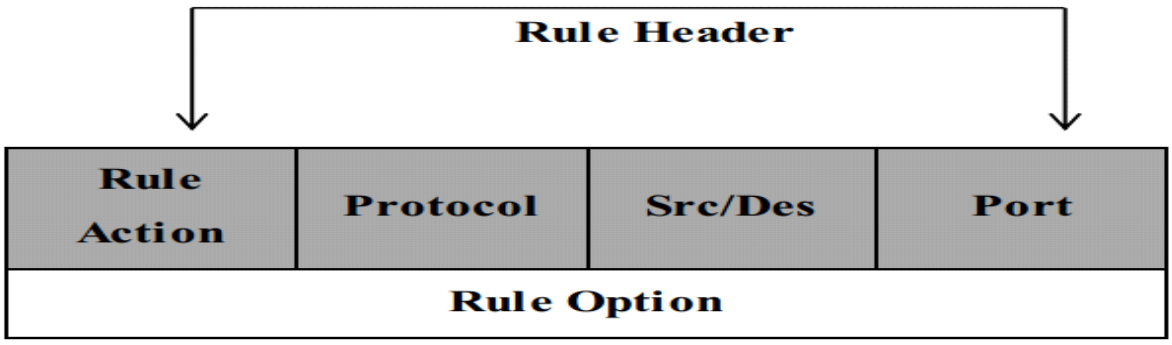
3. Luật trong Suricata

3.1 Giới thiệu về luật trong suricata

– "Luật" (Rule) trong Suricata ta có thể hiểu một cách đơn giản nó giống như các quy tắc và luật lệ trong thế giới thực. Nghĩa là nó sẽ có phần mô tả một trạng thái và hành động gì sẽ xảy ra khi trạng thái đó đúng. Ngoài một cơ sở dữ liệu lớn mà người sử dụng có thể download từ trang chủ của Suricata, người quản trị có thể tự phát triển các luật cho hệ thống của mình.

– Để biết cách viết một luật từ các dữ liệu của hệ thống ta cần phải hiểu cấu trúc của luật trong Suricata như thế nào. Một luật trong Suricata được chia thành hai phần đó là phần ***rule header*** và ***rule options***. Phần rule header bao gồm: rule action, protocol, địa chỉ ip nguồn, địa chỉ ip đích, subnetmask, port nguồn, port đích. Phần options bao gồm các thông điệp cảnh báo, thông tin các phần của gói tin sẽ được kiểm tra để xác định xem hành động nào sẽ được áp dụng.

3.2 Rule Header



Hình 2. Cấu trúc luật trong Suricata

3.2.1 Rule Action

- Phần Header sẽ chứa các thông tin xác định ai, ở đâu, cái gì của một gói tin, cũng như phải làm gì nếu tất cả các thuộc tính trong luật được hiện lên. Mục đầu tiên trong một luật đó chính là phần rule action, rule action sẽ nói cho Suricata biết phải làm gì khi thấy các gói tin phù hợp với các luật đã được quy định sẵn. Có 4 hành động mặc định trong Suricata đó là: pass (cho qua), drop (chặn gói tin), reject, alert (cảnh báo).
    - ***Pass:*** nếu signature được so sánh trùng khớp và chỉ ra là pass thì Suricata sẽ thực hiện dừng quét gói tin và bỏ qua tất cả các luật phía sau đối với gói tin này.
    - ***Drop:*** nếu chương trình tìm thấy một signature hợp lệ và nó chỉ ra là drop thì gói tin đó sẽ bị hủy bỏ và dừng truyền ngay lập tức, khi đó gói tin không thể đến được nơi nhận.
    - ***Reject:*** là hành động bỏ qua gói tin, bỏ qua ở cả bên nhận và bên gửi. Suricata sẽ tạo ra một cảnh báo với gói tin này.
    - ***Alert:*** nếu signature được so sánh là hợp lệ và có chứa một alert thì gói tin đó sẽ được xử lý giống như với một gói tin không hợp lệ. Suricata sẽ tạo ra một cảnh báo.

3.2.2 Protocol

- Trường tiếp theo trong luật đó là protocol. Các giao thức mà Suricata hiện đang phân tích các hành vi bất thường đó là TLS, SSH, SMTP (tải thư điện tử qua mạng internet), IMAP (đặt sự kiểm soát email trên mail server), MSN, SMB (chia sẻ file), TCP, UDP, ICMP và IP, DNS.

3.2.3 IP Address

- Mục tiếp theo của phần header đó là địa chỉ IP. Các địa chỉ này dùng để kiểm tra nơi đi và nơi đến của một gói tin. Địa chỉ ip đó có thể là địa chỉ của một máy đơn hoặc cũng có thể là địa chỉ của một lớp mạng. Từ khóa "any" được sử dụng để định nghĩa một địa chỉ bất kỳ.
- Trong hai địa chỉ IP trong một luật Suricata thì sẽ có một địa chỉ IP nguồn và một địa chỉ IP đích. Việc xác định đâu là địa chỉ nguồn, đâu là địa chỉ đích phụ thuộc vào "→".

Ví dụ:

Alert TCP any any → [192.168.1.0/24, 172.16.0.0/16] 80 (msg: "Access")

3.2.4 . Port

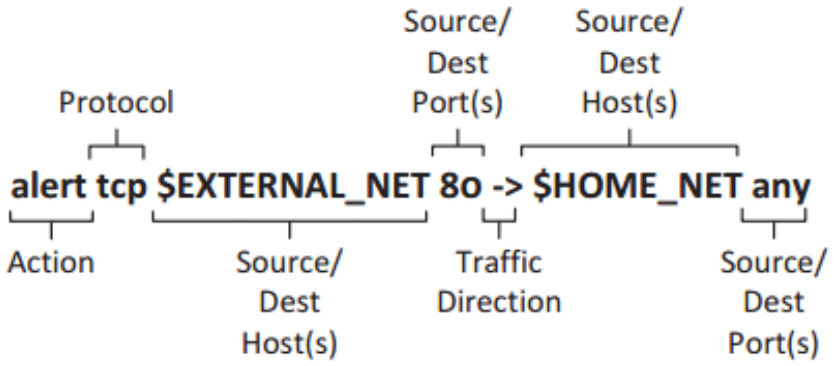
- Port có thể được định nghĩa bằng nhiều cách. Với từ khóa "any" giống như địa chỉ IP để chỉ có thể sử dụng bất kỳ port nào. Gán một port cố định, ví dụ như gán kiểm tra ở port 80 http hoặc port 22 ssh. Ngoài ra ta cũng có thể sử dụng toán tử phủ định để bỏ qua một port nào đó hoặc liệt kê một dải các port.

Ví dụ:

Alert UDP any any → 192.168.1.0/24 1:1024 - port bất kỳ tới dãy port từ 1 - 1024.

3.2.5 Điều hướng

- Toán tử hướng "→" chỉ ra đâu là hướng nguồn, đâu là hướng đích. Phần địa chỉ IP và port ở phía bên trái của toán tử được coi như là địa chỉ nguồn và port nguồn, phần bên phải được coi như địa chỉ đích và port đích. Ngoài ra còn có toán tử "<>" Suricata sẽ xem cặp địa chỉ/port nguồn và đích là như nhau. Nghĩa là nó sẽ ghi/phân tích ở cả hai phía của cuộc hội thoại.



Hình 3. Thứ tự ưu tiên phân tích gói tin cho hệ thống xử lý

Ví dụ:

Alert TCP !192.168.1.0/24 any <> 192.168.1.0/24 23

3.3. Rule Option

- Rule options chính là trung tâm của việc phát hiện xâm nhập. Nội dung chứa các dấu hiệu để xác định một cuộc xâm nhập. Nó nằm ngay sau phần Rule Header và được bọc bởi dấu ngoặc đơn "()". Tất cả các rule options sẽ được phân cách nhau bởi dấu chấm phẩy ";", phần đối số sẽ được tách ra bởi dấy hai chấm ":".
- Có 4 loại rule options chính bao gồm:
  + General: Tùy chọn này cung cấp thông tin về luật đó nhưng không có bất cứ ảnh hưởng nào trong quá trình phát hiện.
  + Payload: Tùy chọn liên quan đến phần tải trong một gói tin.
  + Non-payload: Bao gồm các tùy chọn không liên quan đến phần tải của gói tin (header).
  + Post-detection: Các tùy chọn này sẽ gây ra những quy tắc cụ thể sau khi một luật đã được kích hoạt.

3.3.1 General

3.3.1.1 Msg

- Msg (Message): được dùng để cho biết thêm thông tin về từng signature và các cảnh báo. Phần đầu tiên sẽ cho biết tên tập tin của signature và phần này quy ước là phải viết bằng chữ in hoa. Định dạng của msg như sau:

```
msg: ".....";
```

3.3.1.2 Sid

- Sid (signature id): cho ta biết định danh riêng của mỗi signature. Định danh này được bắt đầu với số. Định dạng của sid như sau:

```
sid:123;
```

3.3.1.3 Rev

- Rev (revision): mỗi sid thường đi kèm với một rev. Rev đại diện cho các phiên bản của signature. Mỗi khi signature được sửa đổi thì số rev sẽ được tăng lên bởi người tạo ra. Định dạng của rev như sau:

```
rev:123;
```

3.3.1.4 Reference

- Reference: cung cấp cho ta địa chỉ đến được những nơi chứa các thông tin đầy đủ về signature. Các tham chiếu có thể xuất hiện nhiều lần trong một signature. Ví dụ về một tham chiếu như sau:

reference: url, www.info.nl

system	URL Prefix
bugtraq	http://www.securityfocus.com/bid
cve	http://cve.mitre.org/cgi-bin/cvename.cgi?name=
nessus	http://cgi.nessus.org/plugins/dump.php3?id=
arachnids	http://www.whitehats.com/info/IDS
mcafee	http://vil.nai.com/vil/dispVirus.asp?virus_k=
url	http://

Hình 4. Bảng các tuỳ chọn của Reference

3.3.1.5 Classtype

– Classtype: cung cấp thông tin về việc phân loại các lớp quy tắc và cảnh báo. Mỗi lớp bao gồm một tên ngắn gọn, một tên đầy đủ và mức độ ưu tiên.

Config classification: web-application-attack,Web Application Attack,1
config classification: not-suspicious,Not Suspicious Traffic,3

Signature	Classification.config	Alert
web-attack	web-attack, Web Application Attack, priority:1	Web Application Attack
not-suspicious	not-suspicious, Not Suspicious Traffic, priority:3	Not Suspicious Traffic

Hình 5. Thông tin phân loại lớp quy tắc

3.3.1.6 Priority

– Priority: chỉ ra mức độ ưu tiên của mỗi signature. Các giá trị ưu tiên dao động từ 1 đến 255, nhưng thường sử dụng các giá trị từ 1 -> 4. Mức ưu tiên cao nhất là 1. Những signature có mức ưu tiên cao hơn sẽ được kiểm tra trước. Định dạng như sau:

```
priority:1;
```

3.3.1.7 Metadata

– Metadata: Suricata sẽ bỏ qua những gì viết sau metadata. Định dạng như sau:

```
metadata:.....;
```

3.3.2 Payload

3.3.2.1 Content

– Content: thể hiện nội dung chúng ta cần viết trong signature, nội dung này được đặt giữa 2 dấu nháy kép. Nội dung là các byte dữ liệu, có 256 giá trị khác nhau (0-255). Chúng có thể là các ký tự thường, ký tự hoa, các ký tự đặc biệt, hay là các mã hexa tương ứng với các ký tự và các mã hexa này phải được đặt giữa 2 dấu gạch dọc. Định dạng của một nội dung như sau:

```
content: ".....";
```

3.3.2.2 Nocase

– Nocase: được dùng để chỉnh sửa nội dung thành các chữ thường, không tạo ra sự khác biệt giữa chữ hoa và chữ thường. Nocase cần được đặt sau nội dung cần chỉnh sửa.

Ví dụ

content: “abC”; nocase;

3.3.2.3 Depth

– Depth: sau từ khóa depth là một số, chỉ ra bao nhiêu byte từ đầu một payload cần được kiểm tra. Depth cần được đặt sau một nội dung. Ví dụ: ta có một payload : abCdefghij. Ta thực hiện kiểm tra 3 byte đầu của payload.

content: “abC”; depth:3;

3.3.2.4 Offset

– Offset: chỉ độ lệch byte trong tải trọng sẽ được kiểm tra. Ví dụ: độ lệch là 3 thì sẽ kiểm tra từ byte thứ 4 trong tải trọng.

content: “def”; offset:3;

Ví dụ:

Alert TCP 192.168.1.0/24 any -> any any (content: \"HTTP"; offset: 4; depth: 40; msg: "HTTP matched";)

3.3.2.5 Distance

– Distance: xác định khoảng cách giữa các nội dung cần kiểm tra trong payload. Khoảng cách này có thể là một số âm.

Ví dụ:

content: “abC”; content: “efg”; distance:1;

3.3.2.6 Within

– Within: được dùng cùng với distance, để chỉ độ rộng của các byte cần kiểm tra sau một nội dung với khoảng cách cho trước đó.

Ví dụ:

content:"GET"; depth:3 content:"download"; distance:10 \within:9;

Luật có nghĩa là tìm “GET” trong 3 byte đầu tiên của trường dữ liệu, di chuyển thêm 10 byte bắt đ ầu từ “GET” và tìm khớp “download”. Tuy nhiên, “download” phải xuất hiện trong 9 byte tiếp theo.

3.3.2.7 Dsize

– Dsize: được dùng để tìm một payload có độ dài bất kỳ.

dsize:min<>max;

3.3.2.8 Rpc

– Rpc (Remote Procedure Call): là một ứng dụng cho phép một chương trình máy tính thực hiện một thủ tục nào đó trên một máy tính khác, thường được sử dụng cho quá trình liên lạc. Định dạng của rpc như sau:

rpc:<application number>, [<version number>|], [<procedure number>|*]>;*

3.3.2.9 Replace

– Replace được dùng để thay đổi nội dung của payload, điều chỉnh lưu lượng mạng. Việc sửa đổi nội dung của payload chỉ có thể được thực hiện đối với gói dữ liệu cá nhân. Sau khi thực hiện thay đổi nội dung xong thì Suricata sẽ thực hiện tính toán lại trường checksum.

3.3.3 Non-Payload

3.3.3.1 IP

- ***ttl***

  – Được sử dụng để kiểm tra về thời gian sống, tồn tại tên mạng của một địa chỉ IP cụ thể trong phần đầu của mỗi gói tin. Giá trị time-to-live (thời gian sống), xác định thời gian tối đa mà mỗi gói tin có thể được lưu thông trên hệ thống mạng. Nếu giá trị này về 0 thì gói tin sẽ bị hủy bỏ. Thời gian sống được xác định dựa trên số hop, khi đi qua mỗi hop/router thì thời gian

sống sẽ bị trừ đi 1. Cơ chế này nhằm hạn chế việc gói tin lưu thông trên mạng vô thời hạn. Định dạng của một ttl như sau:

ttl:<number>;

- ***ipopts***

  – Chúng ta có thể xem và tùy chỉnh các tùy chọn cho việc thiết lập các địa chỉ IP. Việc thiết lập các tùy chọn cần được thực hiện khi bắt đầu một quy tắc. Một số tùy chọn có thể sử dụng:

IP-option	Description
rr	Record Route
eol	End of List
nop	No Op
ts	Time Stamp
sec	IP Security
esec	IP Extended Security
lsrr	Loose Source Routing
ssrr	Strict Source Routing
satid	Stream Identifier
any	any IP options are set

Hình 6. Một số tuỳ chọn của Ipopts

Định dạng của một ipopts như sau:

ipopts: <name>;

- ***sameip***

– Mỗi gói tin sẽ có một địa chỉ IP nguồn và đích. Chúng ta có thể sử dụng sameip để kiểm tra xem địa chỉ IP nguồn và đích có trùng nhau hay không. Định dạng của sameip như sau:

sameip;

- ***Ip_proto***

– Được dùng để giúp ta lựa chọn giao thức. Ta có thể chọn theo tên hoặc số tương ứng với từng giao thức. Có một số giao thức phổ biến sau:

1 ICMP Internet Control Message

6 TCP Transmission Control Protocol

17 UDP User Datagram

47 GRE General Routing Encapsulation

50 ESP Encap Security Payload for IPv6

51 AH Authentication Header for Ipv6

58 IPv6-ICMP ICMP for Ipv6

Định dạng của ip_proto như sau:

ip_proto:<number/name>;

- ***Id***

– Được sử dụng để định danh cho các phân mảnh của gói tin được truyền đi. Khi gói tin truyền đi sẽ được phân mảnh, và các mảnh của một gói tin sẽ có ID giống nhau. Việc này giúp ích cho việc ghép lại gói tin một cách dễ dàng. Định dạng như sau:

id:<number>;

- ***Geoip***

– Cho phép xác định địa chỉ nguồn, đích để gói tin lưu thông trên mạng.

- ***Fragbits***

– Được dùng để kiểm tra các phân mảnh của gói tin. Nó bao gồm các cơ chế sau:

M - More Fragments

D - Do not Fragment

R - Reserved Bit

+ match on the specified bits, plus any others

** match if any of the specified bits are set*

! match if the specified bits are not set

Định dạng của một Fragbits như sau:

fragbits:[+!]<[MDR]>;*

- ***Fragoffset***

  – Kiểm tra sự phù hợp trên các giá trị thập phân của từng mảnh gói tin trên trường offset. Nếu muốn kiểm tra phân mảnh đầu tiên của gói tin, chúng ta cần kết hợp fragoffset 0 với các tùy chọn fragment khác. Các tùy chọn fragment như sau:

< match if the value is smaller than the specified value

> match if the value is greater than the specified value

! match if the specified value is not present

Định dạng của fragoffset:

fragoffset:[!|<|>]<number>;

b. TCP

- ***Sed***

  – Là một số ngẫu nhiên được tạo ra ở cả bên nhận và bên gửi gói tin để kiểm tra số thứ tự của các gói tin đến và đi. Máy khách và máy chủ sẽ tự tạo ra một số seq riêng của mình. Khi một gói tin được truyền thì số seq

này sẽ tăng lên 1. Seq giúp chúng ta theo dõi được những gì diễn ra khi một dòng dữ liệu được truyền đi.

- ***Ack***
  - Được sử dụng để kiểm tra xem gói tin đã được nhận bởi nơi nhận hay chưa trong giao thức kết nối TCP. Số thứ tự của ACK sẽ tăng lên tương ứng với số byte dữ liệu đã được nhận thành công.
- ***Window***
  - Được sử dụng để kiểm tra kích thước của cửa sổ TCP. Kích thước cửa sổ TCP là một cơ chế dùng để kiểm soát các dòng dữ liệu. Cửa sổ được thiết lập bởi người nhận, nó chỉ ra số lượng byte có thể nhận để tránh tình trạng bên nhận bị tràn dữ liệu. Giá trị kích thước của cửa sổ có thể chạy từ 2 đến 65.535 byte.

c. ICMP

- ***Itype***
  - Cung cấp cho việc xác định các loại ICMP. Các thông điệp khác nhau sẽ được phân biệt bởi các tên khác nhau hay các giá trị khác nhau.

  Định dạng của itype như sau:

  itype:min<>max;

  itype:[<|>]<number>;

Type	Name	Reference
0	Echo Reply	[RFC792]
3	Destination Unreachable	[RFC792]
4	Source Quench	[RFC792]
5	Redirect	[RFC792]
6	Alternate Host Address	[JBP]
7	Unassigned	[JBP]
8	Echo	[RFC792]
9	Router Advertisement	[RFC1256]
10	Router Selection	[RFC1256]
11	Time Exceeded	[RFC792]
12	Parameter Problem	[RFC792]
13	Timestamp	[RFC792]
14	Timestamp Reply	[RFC792]
15	Information Request	[RFC792]
16	Information Reply	[RFC792]
17	Address Mask Request	[RFC950]
18	Address Mask Reply	[RFC950]
19	Reserved (for Security)	[Solo]
20-29	Reserved (for Robustness Experiment)	[ZSu]
30	Traceroute	[RFC1393]
31	Datagram Conversion Error	[RFC1475]
32	Mobile Host Redirect	[David Johnson]
37	Domain Name Request	[RFC1788]
38	Domain Name Reply	[RFC1788]
39	SKIP	[Markson]
40	Photuris	[RFC2521]

Hình 7. Bảng Type của ICMP Header

- ***Icode***

  – Cho phép xác định mã của từng ICMP để làm rõ hơn cho từng gói tin ICMP. Định dạng của icode như sau:

  icode:min<>max;

  icode:[<|>]<number>;

- ***Icmp_id***

  – Mỗi gói tin ICMP có một giá trị ID khi chúng được gửi. Tại thời điểm đó, người nhận sẽ trả lại tin nhắn với cùng một giá trị ID để người gửi sẽ nhận ra và kết nối nó đúng với yêu cầu ICMP đã gửi trước đó. Định dạng của một icmp_id như sau:

icmp_id:<number>;

- ***Icmp_seq***

  – Được sử dụng để kiểm tra số thứ tự của ICMP. Định dạng của icmp_seq như sau:

icmp_seq:<number>;

d. HTTP

- ***http_method***

– Chỉ ra các phương thức được áp dụng với các request http. Các phương thức http: GET, POST, PUT, HEAD, DELETE, TRACE, OPTIONS, CONNECT và PATCH.

- ***http_uri và http_raw_uri***

– Chỉ ra đường dẫn tới nơi chứa nội dung yêu cầu.

- ***http_header***

– Chỉ ra phương thức sử dụng, địa chỉ cần truy cập tới và tình trạng kết nối.

- ***http_cookie.***
- ***http_user_agent***

– Là một phần của http_header, chỉ ra thông tin về trình duyệt của người dùng.

- ***http_client_body***

– Chỉ ra các yêu cầu của máy trạm.

- ***http_stat_code***

– Chỉ ra mã trạng thái của server mà máy trạm yêu cầu kết nối tới.

- ***http_stat_msg***

– Các dòng tin thông báo về tình trạng máy chủ, hay tình trạng về việc đáp ứng các yêu cầu kết nối của máy trạm.

- ***http_server_body***

– Chỉ ra nội dung đáp trả các yêu cầu từ máy trạm của máy chủ.

- ***File_data***

– Chỉ ra nội dung, đường dẫn tới file chứa dữ liệu được yêu cầu.

e. Flow

- ***Flowbits***

– Gồm 2 phần, phần đầu mô tả các hành động được thực hiện, phần thứ 2 là tên của flowbit. Các hành động của flowbit:

flowbits: set, name	*Được dùng để thiết lập các điều kiện/tên cho các flow.*
flowbits: isset, name	*Có thể được sử dụng trong các luật để đảm bảo rằng sẽ tạo ra một cảnh báo khi các luật là phù hợp và các điều kiện sẽ được thiết lập trong flow.*
flowbits: toggle, name	*Dùng để đảo ngược các thiết lập hiện tại.*
flowbits: unset, name	*Được sử dụng để bỏ các thiết lập về điều kiện trong luật.*
flowbits: isnotset, name	*Được sử dụng để đảm bảo rằng sẽ tạo ra một cảnh báo khi các luật là phù hợp và các điều kiện sẽ không được thiết lập trong flow.*

- ***Flow***

– Có thể được sử dụng để kết nối các thư mục chứa các flow lại với nhau. Các flow có thể được đi từ hoặc đến từ Client/Server và các flow này có thể ở trạng thái được thiết lập hoặc không. Việc kết nối các flow có thể xảy ra các trường hợp sau:

to_client established/ stateless

from_client established/ stateless

to_server established/ stateless

from_server established/ stateless

CHƯƠNG 3: THỰC NGHIỆM

1. Cài đặt suricata

– Trước khi cài đặt phần mềm chúng ta tiến hành cài đặt các thành phần cần thiết cho Suricata

```
sudo apt-get -y install libpcre3 libpcre3-dbg libpcre3-dev \
build-essential autoconf automake libtool libpcap-dev libnet1-dev \
libyaml-0-2 libyaml-dev zlib1g zlib1g-dev libcap-ng-dev libcap-ng0 \
make libmagic-dev
```

– Mặc định, Suricata làm việc như một IDS. Nếu bạn muốn cài đặt tính năng IPS cho nó thực hiện cài đặt với lệnh:

```
sudo apt-get -y install libnetfilter-queue-dev libnetfilter-queue1 libnfnetlink-dev libnfnetlink0
```

- Download và buil Suricata:

```
wget http://www.openinfosecfoundation.org/download/suricata-2.0.6.tar.gz
tar -xvzf suricata-2.0.6.tar.gz
cd suricata-2.0.6
```

- Nếu muốn xây dựng Suricata với khả năng của một IPS, chạy dòng lệnh:

```
./configure --enable-nfqueue --prefix=/usr --sysconfdir=/etc --localstatedir=/var
```

- Cài đặt và cấu hình cho Suricata:

```
./configure --prefix=/usr --sysconfdir=/etc --localstatedir=/var
make
sudo make install
sudo ldconfig
```

- Tạo thư mục chứa thông tin về Suricta:

```
sudo mkdir /var/log/suricata
```

- Tạo thư mục cài đặt:

```
sudo mkdir /etc/suricata
```

- Copy các file classification.config, reference.config and suricata.yaml from the base build/installation directory vào thư mục vừa tạo : /etc/suricata:

```
sudo cp classification.config /etc/suricata
sudo cp reference.config /etc/suricata
sudo cp suricata.yaml /etc/suricata
```

- Để quản lý luật chúng ta cài Oinkmaster

```
sudo apt-get install oinkmaster
```

2. Cấu hình suricata

- Ta sẽ tiến hành cấu hình cho các tham biến trong tập rules của Suricata.
- Đầu tiên, ta sẽ phải thiết lập lại các biến cho các khu vực (địa chỉ IP) khác nhau và các biến cho các cổng. Mục đích của việc thiết lập biến này giúp cho việc thay đổi dễ dàng khi cần thiết.
  + Bước 1: Mở file /etc/suricata/suricata.yaml – đây là file cấu hình của Suricata.

  Trong file này các tham biến đã được cấu hình sẵn, ta sẽ thay đổi các tham biến đó sao cho phù hợp với máy tính của mình.
  + Bước 2: Nhấn tổ hợp phím Ctrl+F, nhập vào từ khoá vars và tìm kiếm. Khi trả về kết quả sẽ xuất hiện các tham biến: HOME_NET, EXTERNAL_NET, HTTP_SERVERS, SMTP_SERVERS, SQL_SERVERS…

```
HOME_NET: "[192.168.0.0/16,10.0.0.0/8,172.16.0.0/12]"

EXTERNAL_NET: "!$HOME_NET"

HTTP_SERVERS: "$HOME_NET"

SMTP_SERVERS: "$HOME_NET"

SQL_SERVERS: "$HOME_NET"

DNS_SERVERS: "$HOME_NET"

TELNET_SERVERS: "$HOME_NET"
```

- Ta sẽ cấu hình như sau
  + HOME_NET: điền vào dải ip-address đang sử dụng:

```
HOME_NET: "[192.168.1.0/24]"
```

  + EXTERNAL_NET: là các địa chỉ IP khác HOME_NET

```
EXTERNAL_NET: "!$HOME_NET"
```

  + HTTP_SERVERS, SMTP_SERVERS, SQL_SERVERS, DNS_SERVERS và TELNET_SERVERS: cài đặt mặc định là "$HOME_NET"

```
HTTP_SERVERS: "$HOME_NET"

SMTP_SERVERS: "$HOME_NET"

SQL_SERVERS: "$HOME_NET"

DNS_SERVERS: "$HOME_NET"

TELNET_SERVERS: "$HOME_NET"
```

  + AIM_SERVERS: cài đặt mặc định là bất kỳ

```
AIM_SERVERS: "any"
```

  + Thiết lập các cổng: HTTP_PORTS, SHELLCODE_PORTS, ORACLE_PORTS, SSH_PORTS

```
HTTP_PORTS: "80"

SHELLCODE_PORTS: "!80"

ORACLE_PORTS: 1521

SSH_PORTS: 22

DNP3_PORTS: 20000
```

+ Thiết lập host-os-policy: điền vào địa chỉ máy tính đang sử dụng sau tên hệ điều hành

```
host-os-policy:
  # Make the default policy windows.
  windows: [0.0.0.0/0]
  bsd: []
  bsd-right: []
  old-linux: []
  linux: [10.0.0.0/8, 192.168.1.24, "8762:2352:6241:7245:E000:0000:0000:0000"]
  old-solaris: []
  solaris: ["::1"]
  hpux10: []
  hpux11: []
  irix: []
  macos: []
  vista: []
  windows2k3: []
```

3. Kết quả thực nghiệm

- Brute fore attack

  alert tcp any any -> any any (msg:"Brute force detected";

  content"DVWA/vulnerabilities/brute/?username=admin&"; http_uri;

  threshold: type threshold, track by_scr, count 20, seconds 5; sid:2;)

  Kết quả: mở file fast.log

```
/var/lib/suricata/rules
tail -f /var/log/suricata/fast.log
10/14/2021-22:27:07.269713  [**] [1:2:1] Brute force detected [**] [Classification: (null)] [Priority: 3] {TCP} fe80:0000:0000:0000:d198:4515
:7f47:d5e4:143 -> ff02:0000:0000:0000:0000:0000:0000:0016:0
10/14/2021-22:27:07.269668  [**] [1:2:1] Brute force detected [**] [Classification: (null)] [Priority: 3] {TCP} fe80:0000:0000:0000:d198:4515
:7f47:d5e4:143 -> ff02:0000:0000:0000:0000:0000:0000:0016:0
^[[A^[[A^[[A
```

- SQL injection

alert http any any -> any any (msg: "Possible SQL Injection attack (Contains singlequote)"; flow:established,to_server; content:"'"; nocase; http_uri; sid:1;)

alert http any any -> any any (msg: "Possible SQL Injection attack (Contains UNION)"; flow:established,to_server; content:"union"; nocase; http_uri; sid:2;)

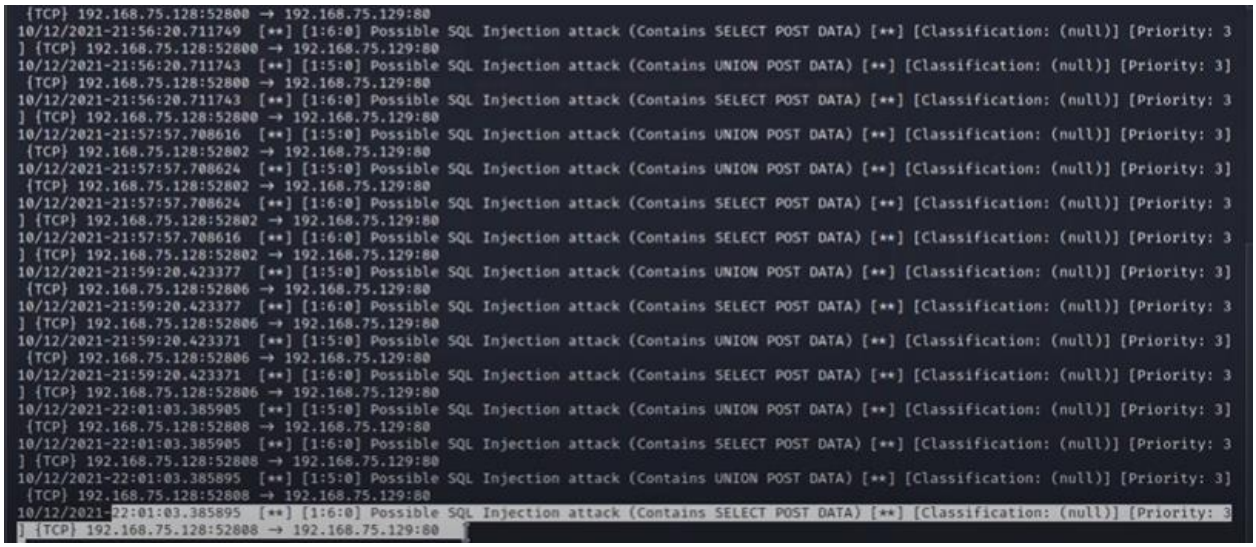
alert http any any -> any any (msg: "Possible SQL Injection attack (Contains SELECT)"; flow:established,to_server; content:"select"; nocase; http_uri; sid:3;)

alert http any any -> any any (msg: "Possible SQL Injection attack (Contains singlequote POST DATA)"; flow:established,to_server; content:"'"; nocase; http_client_body; sid:4;)

alert http any any -> any any (msg: "Possible SQL Injection attack (Contains UNION POST DATA)"; flow:established,to_server; content:"union"; nocase; http_client_body; sid:5;)

alert http any any -> any any (msg: "Possible SQL Injection attack (Contains SELECT POST DATA)"; flow:established,to_server; content:"select"; nocase; http_client_body; sid:6;)

Kết quả khi áp rules:


{TCP} 192.168.75.128:52800 → 192.168.75.129:80
10/12/2021-21:56:20.711749 [**] [1:6:0] Possible SQL Injection attack (Contains SELECT POST DATA) [**] [Classification: (null)] [Priority: 3
] {TCP} 192.168.75.128:52800 → 192.168.75.129:80
10/12/2021-21:56:20.711743 [**] [1:5:0] Possible SQL Injection attack (Contains UNION POST DATA) [**] [Classification: (null)] [Priority: 3]
{TCP} 192.168.75.128:52800 → 192.168.75.129:80
10/12/2021-21:56:20.711743 [**] [1:6:0] Possible SQL Injection attack (Contains SELECT POST DATA) [**] [Classification: (null)] [Priority: 3
] {TCP} 192.168.75.128:52800 → 192.168.75.129:80
10/12/2021-21:57:57.708616 [**] [1:5:0] Possible SQL Injection attack (Contains UNION POST DATA) [**] [Classification: (null)] [Priority: 3]
{TCP} 192.168.75.128:52802 → 192.168.75.129:80
10/12/2021-21:57:57.708624 [**] [1:5:0] Possible SQL Injection attack (Contains UNION POST DATA) [**] [Classification: (null)] [Priority: 3]
{TCP} 192.168.75.128:52802 → 192.168.75.129:80
10/12/2021-21:57:57.708624 [**] [1:6:0] Possible SQL Injection attack (Contains SELECT POST DATA) [**] [Classification: (null)] [Priority: 3
] {TCP} 192.168.75.128:52802 → 192.168.75.129:80
10/12/2021-21:57:57.708616 [**] [1:6:0] Possible SQL Injection attack (Contains SELECT POST DATA) [**] [Classification: (null)] [Priority: 3
] {TCP} 192.168.75.128:52802 → 192.168.75.129:80
10/12/2021-21:59:20.423377 [**] [1:5:0] Possible SQL Injection attack (Contains UNION POST DATA) [**] [Classification: (null)] [Priority: 3]
{TCP} 192.168.75.128:52806 → 192.168.75.129:80
10/12/2021-21:59:20.423377 [**] [1:6:0] Possible SQL Injection attack (Contains SELECT POST DATA) [**] [Classification: (null)] [Priority: 3
] {TCP} 192.168.75.128:52806 → 192.168.75.129:80
10/12/2021-21:59:20.423371 [**] [1:5:0] Possible SQL Injection attack (Contains UNION POST DATA) [**] [Classification: (null)] [Priority: 3]
{TCP} 192.168.75.128:52806 → 192.168.75.129:80
10/12/2021-21:59:20.423371 [**] [1:6:0] Possible SQL Injection attack (Contains SELECT POST DATA) [**] [Classification: (null)] [Priority: 3
] {TCP} 192.168.75.128:52806 → 192.168.75.129:80
10/12/2021-22:01:03.385905 [**] [1:5:0] Possible SQL Injection attack (Contains UNION POST DATA) [**] [Classification: (null)] [Priority: 3]
{TCP} 192.168.75.128:52808 → 192.168.75.129:80
10/12/2021-22:01:03.385905 [**] [1:6:0] Possible SQL Injection attack (Contains SELECT POST DATA) [**] [Classification: (null)] [Priority: 3
] {TCP} 192.168.75.128:52808 → 192.168.75.129:80
10/12/2021-22:01:03.385895 [**] [1:5:0] Possible SQL Injection attack (Contains UNION POST DATA) [**] [Classification: (null)] [Priority: 3]
{TCP} 192.168.75.128:52808 → 192.168.75.129:80
10/12/2021-22:01:03.385895 [**] [1:6:0] Possible SQL Injection attack (Contains SELECT POST DATA) [**] [Classification: (null)] [Priority: 3
] {TCP} 192.168.75.128:52808 → 192.168.75.129:80