BÁO CÁO LAB 2

Môn: An toàn mạng máy tính nâng cao

GVTH: Đỗ Thị Phương Uyên

Sinh viên thực hiện	**Sinh viên 1** MSSV: 21521182 Họ tên: Nguyễn Đại Nghĩa **Sinh viên 2** MSSV: 21521295 Họ tên: Phạm Hoàng Phúc **Sinh viên 3** MSSV: 21521848 Họ tên: Hoàng Gia Bảo **Sinh viên 4** MSSV: 21521386 Họ tên: Lê Xuân Sơn
Lớp	**NT534.O21.ATCL.1**
Tổng thời gian thực hiện Lab trung bình	
Phân chia công việc *(nếu là nhóm)*	**[Sinh viên 1]**: Thực hiện viết 3 rule đầu của taks thứ 3 **[Sinh viên 2]**: Làm task đầu tiên **[Sinh viên 3]:** Giải thích 2 rule cuối task thứ 2 và viết 1 rule cuối của task thứ 3 **[Sinh viên 4]:** Giải thích 3 rule đầu của task thứ 2
Link Video thực hiện *(nếu có yêu cầu)*	

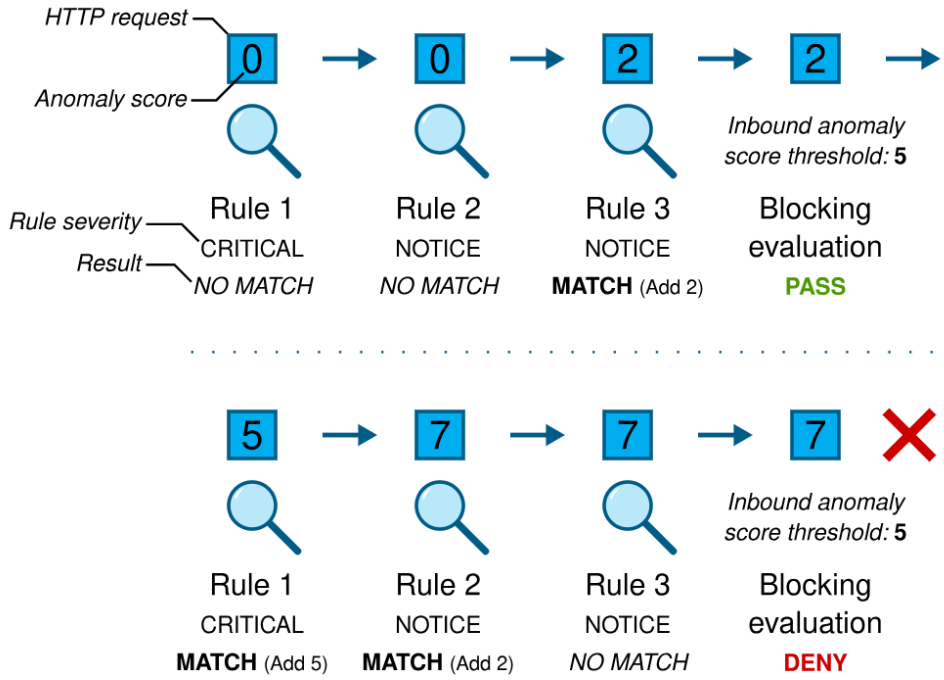
Ý kiến *(nếu có)* *+ Khó khăn gặp phải* *+ Đề xuất, góp ý…*	
Điểm tự đánh giá *(bắt buộc)*	9.5 /**10**

Task: Tìm hiểu cơ chế hoạt động của việc tính điểm bất thường của bộ Core Rule Set và giải thích tại sao cơ chế này lại dẫn đến sự xuất hiện của False Positive.

Cơ chế hoạt động:

Cơ chế tính điểm bất thường (Anomaly scoring mode) kết hợp giữa 2 khái niệm phát hiện cộng tác (collaborative detection) và chặn trì hoãn (delayed blocking). Ý tưởng chính của cơ chế này là tách detection rule khỏi chức năng chặn, tức là khi phát hiện bất thường sẽ không chặn ngay. Thay vào đó, khi 1 request/response khớp với 1 rule, nó sẽ bị tính thêm điểm bất thường (anomaly score).

Sau khi request/response đi qua hết tất cả các rule, quá trình đánh giá sẽ diễn ra. Khi đó, request/response nào có anomaly score lớn hơn hoặc bằng ngưỡng (threshold) sẽ bị chặn lại. Những request/response dưới ngưỡng sẽ tiếp tục hoạt động.



Nguyên nhân gây ra False Positive:

+Vì Core Rule Set sử dụng các rule để phát hiện bất thường nên nếu rule quá nghiêm ngặt hoặc không chính xác do không điều chỉnh, cập nhật đầy đủ có thể dẫn đến nhầm lẫn các yêu cầu hợp lệ là tấn công. Đôi khi, một vài yêu cầu hợp lệ cũng có đặc điểm tương tự tấn công.

+Ngoài ra, độ nhạy cảm của threshold cũng có thể ảnh hưởng đến quá trình đánh giá. Do ngưỡng bất thường là giá trị cố định, nếu ngưỡng đặt quá thấp sẽ dễ xảy ra false positive. Nhưng nếu quá cao sẽ gây ra false negative.

Task: Tìm hiểu cách viết rule, giải thích ý nghĩa của ít nhất 3 rule trong bộ Core Rule Set.

1. **RULE id: 934110 (/usr/share/modsecurity-crs/rules/REQUEST-934-APPLICATION-ATTACK-GENERIC.conf)**

Tên: Possible Server Side Request Forgery (SSRF) Attack

Giải thích:

- **SecRule**: Đây là chỉ thị chính để định nghĩa một quy tắc ModSecurity. Nó xác định các điều kiện cần phải đáp ứng để kích hoạt quy tắc.
- **Variables**: Quy tắc phù hợp với các biến yêu cầu khác nhau, bao gồm REQUEST_COOKIES, REQUEST_FILENAME, ARGS_NAMES, ARGS, và dữ liệu XML. Nó kiểm tra đầu vào cụ thể liên quan đến các cuộc tấn công SSRF.
- **Operators**: Quy tắc sử dụng toán tử "@pmFromFile" để thực hiện một phần khớp với các mẫu được tải từ một tệp có tên "ssrf.data". Tệp này có thể chứa một danh sách các URL hoặc mẫu SSRF đã biết.
- **Actions**: Nếu quy tắc được kích hoạt, một số hành động sẽ được thực hiện:
    - Nó đặt ID, phase, và mức độ nghiêm trọng của quy tắc.
    - Nó chặn yêu cầu, thêm các thẻ để phân loại cuộc tấn công.
    - Nó ghi lại dữ liệu phù hợp và lưu trữ nó.
    - Nó tăng điểm anomaly để chỉ ra mức độ nghiêm trọng của cuộc tấn công được phát hiện..
- **msg**: In ra thông báo nếu quy tắc được kích hoạt
- **logdata**: Định dạng dữ liệu sẽ được ghi lại khi quy tắc được kích hoạt. Nó ghi lại dữ liệu phù hợp và tên biến nơi sự phù hợp đã xảy ra.
- **tags**: Thêm các thẻ khác nhau để phân loại cuộc tấn công và cung cấp ngữ cảnh bổ sung.
- **ver**: Xác định phiên bản của OWASP CRS (Core Rule Set) đang được sử dụng (OWASP_CRS/4.0.1-dev)

- **severity**: Mức độ nghiệm trọng của tấn công (ở mức CRITICAL)
- **setvar**: Điều chỉnh điểm anomaly để phản ánh mức độ nghiêm trọng của cuộc tấn công SSRF đã phát hiện. Nó tăng cả điểm anomaly chung và điểm cho các cuộc tấn công SSRF cụ thể.

2. **RULE id: 942150 (/usr/share/modsecurity-crs/rules/REQUEST-942-APPLICATION-ATTACK-SQLI.conf)**

```
SecRule REQUEST_COOKIES|!REQUEST_COOKIES:/__utm/|REQUEST_COOKIES_NAMES|ARGS_NAMES|ARGS|XML:/* "@rx (?i)\b(?:json(?:_[0-9A-Z_a-z]+)?|
    "id:942150,\
    phase:2,\
    block,\
    capture,\
    t:none,t:urlDecodeUni,\
    msg:'SQL Injection Attack: SQL function name detected',\
    logdata:'Matched Data: %{TX.0} found within %{MATCHED_VAR_NAME}: %{MATCHED_VAR}',\
    tag:'application-multi',\
    tag:'language-multi',\
    tag:'platform-multi',\
    tag:'attack-sqli',\
    tag:'paranoia-level/2',\
    tag:'OWASP_CRS',\
    tag:'capec/1000/152/248/66',\
    tag:'PCI/6.5.2',\
    ver:'OWASP_CRS/4.0.1-dev',\
    severity:'CRITICAL',\
    setvar:'tx.sql_injection_score=+%{tx.critical_anomaly_score}',\
    setvar:'tx.inbound_anomaly_score_pl2=+%{tx.critical_anomaly_score}'"
```

Tên: SQL Injection Attack

Giải thích:

- **SecRule**: Đây là chỉ thị chính để định nghĩa một quy tắc ModSecurity. Nó xác định các điều kiện cần phải đáp ứng để kích hoạt quy tắc.
- **Variables**: Quy tắc này kiểm tra các biến yêu cầu khác nhau như REQUEST_COOKIES, REQUEST_COOKIES_NAMES, ARGS_NAMES, ARGS, và XML data để tìm kiếm các mẫu phù hợp với regex.
- **Operators**: Quy tắc sử dụng toán tử "@rx" để kiểm tra xem các giá trị của biến có khớp với regex đã chỉ định hay không.
- **Actions**: Nếu quy tắc được kích hoạt, một số hành động sẽ được thực hiện:
    - Nó đặt ID, phase, và mức độ nghiêm trọng của quy tắc.
    - Nó chặn yêu cầu, thêm các thẻ để phân loại cuộc tấn công.
    - Nó ghi lại dữ liệu phù hợp và lưu trữ nó.
    - Nó tăng điểm anomaly để chỉ ra mức độ nghiêm trọng của cuộc tấn công được phát hiện..

- **msg**: In ra thông báo nếu quy tắc được kích hoạt
- **logdata**: Định dạng dữ liệu sẽ được ghi lại khi quy tắc được kích hoạt. Nó ghi lại dữ liệu phù hợp và tên biến nơi sự phù hợp đã xảy ra.
- **tags**: Thêm các thẻ khác nhau để phân loại cuộc tấn công và cung cấp ngữ cảnh bổ sung.
- **ver**: Xác định phiên bản của OWASP CRS (Core Rule Set) đang được sử dụng (OWASP_CRS/4.0.1-dev)
- **severity**: Mức độ nghiêm trọng của tấn công (ở mức CRITICAL)
- **setvar**: Điều chỉnh điểm anomaly để phản ánh mức độ nghiêm trọng của cuộc tấn công SQL đã phát hiện. Nó tăng cả điểm anomaly chung và điểm cho các cuộc tấn công SQL cụ thể.

3. RULE id: 941110 (/usr/share/modsecurity-crs/rules/REQUEST-941-APPLICATION-ATTACK-XSS.conf)

```
SecRule REQUEST_COOKIES|!REQUEST_COOKIES:/__utm/|REQUEST_COOKIES_NAMES|REQUEST_HEADERS:User-Agent|ARGS_NAMES|ARGS|XML:/* "@detectXSS" \
    "id:941100,\
    phase:2,\
    block,\
    t:none,t:utf8toUnicode,t:urlDecodeUni,t:htmlEntityDecode,t:jsDecode,t:cssDecode,t:removeNulls,\
    msg:'XSS Attack Detected via libinjection',\
    logdata:'Matched Data: XSS data found within %{MATCHED_VAR_NAME}: %{MATCHED_VAR}',\
    tag:'application-multi',\
    tag:'language-multi',\
    tag:'platform-multi',\
    tag:'attack-xss',\
    tag:'xss-perf-disable',\
    tag:'paranoia-level/1',\
    tag:'OWASP_CRS',\
    tag:'capec/1000/152/242',\
    ver:'OWASP_CRS/4.0.1-dev',\
    severity:'CRITICAL',\
    setvar:'tx.xss_score=+%{tx.critical_anomaly_score}',\
    setvar:'tx.inbound_anomaly_score_pl1=+%{tx.critical_anomaly_score}'"
```

Tên: XSS Attack Detected via libinjection

Giải thích:

- **SecRule**: Đây là chỉ thị chính để định nghĩa một quy tắc ModSecurity. Nó xác định các điều kiện cần phải đáp ứng để kích hoạt quy tắc.
- **Variables**: Quy tắc này kiểm tra các biến yêu cầu khác nhau như REQUEST_COOKIES, REQUEST_COOKIES_NAMES, REQUEST_HEADERS, ARGS_NAMES, ARGS, và XML data để tìm kiếm các mẫu XSS.
- **Operators** Quy tắc sử dụng toán tử "@detectXSS" để kiểm tra xem các giá trị của biến có chứa các mẫu XSS được phát hiện bởi cơ chế nhận dạng XSS.

- **Actions**: Nếu quy tắc được kích hoạt, một số hành động sẽ được thực hiện:
    - Nó đặt ID, phase, và mức độ nghiêm trọng của quy tắc.
    - Nó chặn yêu cầu, thêm các thẻ để phân loại cuộc tấn công.
    - Nó ghi lại dữ liệu phù hợp và lưu trữ nó.
    - Nó tăng điểm anomaly để chỉ ra mức độ nghiêm trọng của cuộc tấn công được phát hiện..
- **msg**: In ra thông báo nếu quy tắc được kích hoạt
- **logdata**: Định dạng dữ liệu sẽ được ghi lại khi quy tắc được kích hoạt. Nó ghi lại dữ liệu phù hợp và tên biến nơi sự phù hợp đã xảy ra.
- **tags**: Thêm các thẻ khác nhau để phân loại cuộc tấn công và cung cấp ngữ cảnh bổ sung.
- **ver**: Xác định phiên bản của OWASP CRS (Core Rule Set) đang được sử dụng (OWASP_CRS/4.0.1-dev)
- **severity**: Mức độ nghiêm trọng của tấn công (ở mức CRITICAL)
- **setvar**: Điều chỉnh điểm anomaly để phản ánh mức độ nghiêm trọng của cuộc tấn công XSS đã phát hiện. Nó tăng cả điểm anomaly chung và điểm cho các cuộc tấn công XSS cụ thể.

4. RULE id: 913100 (/usr/share/modsecurity-crs/rules/REQUEST-913-SCANNER-DETECTION.conf)

```
7
8 SecRule REQUEST_HEADERS:User-Agent "@pmFromFile scanners-user-agents.data" \
9     "id:913100,\
0     phase:1,\
1     block,\
2     capture,\
3     t:none,\
4     msg:'Found User-Agent associated with security scanner',\
5     logdata:'Matched Data: %{TX.0} found within %{MATCHED_VAR_NAME}: %{MATCHED_VAR}',\
6     tag:'application-multi',\
7     tag:'language-multi',\
8     tag:'platform-multi',\
9     tag:'attack-reputation-scanner',\
0     tag:'paranoia-level/1',\
1     tag:'OWASP_CRS',\
2     tag:'capec/1000/118/224/541/310',\
3     tag:'PCI/6.5.10',\
4     ver:'OWASP_CRS/4.2.0-dev',\
5     severity:'CRITICAL',\
6     setvar:'tx.inbound_anomaly_score_pl1=+%{tx.critical_anomaly_score}'"
7
8
9 SecRule TX:DETECTION_PARANOIA_LEVEL "@lt 2" "id:913013,phase:1,pass,nolog,tag:'OWASP_CRS',ver:'OWASP_CRS/4.2.0-dev',skipAfter:END-REQUEST-913-SCANNER-DETECTION"
0 SecRule TX:DETECTION_PARANOIA_LEVEL "@lt 2" "id:913014,phase:2,pass,nolog,tag:'OWASP_CRS',ver:'OWASP_CRS/4.2.0-dev',skipAfter:END-REQUEST-913-SCANNER-DETECTION"
1 #
```

Tên : Found User-Agent associated with security scanner

Giải thích:

- **SecRule**: Đây là chỉ thị chính để định nghĩa một quy tắc ModSecurity. Rule này được áp dụng vào phần User-Agent của các yêu cầu HTTP để phát hiện các User-Agent liên quan đến các công cụ quét bảo mật (security scanner).

- **Variables**: Quy tắc này sử dụng biến REQUEST_HEADERS:User-Agent để lấy giá trị của User-Agent từ tiêu đề yêu cầu HTTP.
- **Operators**: Quy tắc sử dụng toán tử "@pmFromFile" để so khớp với một phần của dữ liệu được tải từ tệp "scanners-user-agents.data". Điều này cho phép so sánh User-Agent trong yêu cầu với danh sách các User-Agent của các công cụ quét bảo mật đã biết.

- **Actions**: Khi quy tắc được kích hoạt, nó thực hiện các hành động sau:
  - Thiết lập ID, phase, và mức độ nghiêm trọng của quy tắc.
  - Chặn yêu cầu và ghi lại thông báo về việc phát hiện User-Agent liên quan đến công cụ quét bảo mật.
  - Ghi lại dữ liệu phù hợp và lưu trữ nó.
  - Tăng điểm anomaly để chỉ ra mức độ nghiêm trọng của cuộc tấn công được phát hiện.
- **msg**: Hiển thị thông báo "Found User-Agent associated with security scanner" nếu quy tắc được kích hoạt.
- **logdata**: Định dạng dữ liệu sẽ được ghi lại khi quy tắc được kích hoạt. Nó ghi lại dữ liệu phù hợp và tên biến nơi sự phù hợp đã xảy ra.
- **tags**: Thêm các thẻ để phân loại cuộc tấn công và cung cấp ngữ cảnh bổ sung, bao gồm các thẻ như 'application-multi', 'language-multi', 'platform-multi', 'attack-reputation-scanner', và các thẻ khác.
- **ver**: Xác định phiên bản của OWASP CRS (Core Rule Set) đang được sử dụng.
- **severity**: Xác định mức độ nghiêm trọng của tấn công, ở mức CRITICAL.
- **setvar**: Điều chỉnh điểm anomaly để phản ánh mức độ nghiêm trọng của cuộc tấn công đã phát hiện. Nó tăng cả điểm anomaly chung và điểm cho các cuộc tấn công cụ thể.

5.RULE ID : 92110 (/usr/share/modsecurity-crs/rules/REQUEST-921-PROTOCOL-ATTACK.conf)

```
SecRule ARGS_NAMES|ARGS|REQUEST_BODY|XML:/* "@rx (?:get|post|head|options|connect|put|delete|trace|track|patch|propfind|propatch|mkcol|copy|move|lock|unlock)\s+[^\s]+\s+http/\d" \
    "id:921110,\
    phase:2,\
    block,\
    capture,\
    t:none,t:htmlEntityDecode,t:lowercase,\
    msg:'HTTP Request Smuggling Attack',\
    logdata:'Matched Data: %{TX.0} found within %{MATCHED_VAR_NAME}: %{MATCHED_VAR}',\
    tag:'application-multi',\
    tag:'language-multi',\
    tag:'platform-multi',\
    tag:'attack-protocol',\
    tag:'paranoia-level/1',\
    tag:'OWASP_CRS',\
    tag:'capec/1000/210/272/220/33',\
    ver:'OWASP_CRS/4.2.0-dev',\
    severity:'CRITICAL',\
    setvar:'tx.http_violation_score=+%{tx.critical_anomaly_score}',\
    setvar:'tx.inbound_anomaly_score_pl1=+%{tx.critical_anomaly_score}'"

#
```

Tên : HTTP Request Smuggling Attack

Giải thích :

- **SecRule**: Đây là chỉ thị chính để định nghĩa một quy tắc ModSecurity.
- **Variables**: Rule này sử dụng các biến ARGS_NAMES, ARGS, REQUEST_BODY, và XML để kiểm tra nội dung của các tham số trong yêu cầu HTTP.
- **Operators**: Rule sử dụng toán tử @rx để kiểm tra xem các giá trị của các tham số có khớp với biểu thức chính quy đã được chỉ định hay không.
- **Actions**: Nếu rule này được kích hoạt, một số hành động sẽ được thực hiện:
  - block: Từ chối yêu cầu nếu rule được kích hoạt.
  - capture: Ghi lại dữ liệu phù hợp.
  - t:none,t:htmlEntityDecode,t:lowercase: Thực hiện các biến đổi trước khi so sánh (giải mã HTML entities và chuyển đổi thành chữ thường).
- **logdata**: Ghi lại dữ liệu phù hợp và vị trí nó được tìm thấy.
- **setvar**: Điều chỉnh điểm tích lũy của vi phạm HTTP.
- **Tags**: Các tag được sử dụng để phân loại cuộc tấn công và cung cấp ngữ cảnh bổ sung, bao gồm application-multi, language-multi, platform-multi, attack-protocol, paranoia-level/1, OWASP_CRS, và các tag liên quan đến CAPEC (Common Attack Pattern Enumeration and Classification).

Ver và Severity: Xác định phiên bản của OWASP CRS (Core Rule Set) đang được sử dụng và mức độ nghiêm trọng của cuộc tấn công được phát hiện (ở mức CRITICAL).

Task: Chỉnh sửa hoặc viết mới 3 custom rule và thử nghiệm lại rule trên web DVWA.

Trước khi viết rule thì em sẽ tạo ra 1 file rule mới cho riêng mình ở folder ATM_Nang_Cao với tên file là custom_rules.conf:

Rule đầu tiên mà em viết chính là rule ngăn chặn nhập liệu nhập từ url có các từ khóa liên quan đến sql injection:

```
SecRule ARGS|REQUEST_URI "@rx (select|union|insert|delete|drop|update|alter|exec|execute)" \
    "id:'1000001',phase:1,log,deny,status:403,msg:'Da chan do nhap lieu co chua nghi van SQL injection.'"
```

Giải thích sơ về rule này sẽ như sau:

Với từ khóa ARGS|REQUEST_URI thì ARGS là toàn bộ các tham số được truyền qua phương thức POST và GET, trong khi REQUEST_URI là phần URI của yêu cầu HTTP.

@rx là một toán tử của ModSecurity chỉ ra rằng một biểu thức chính quy sẽ được sử dụng cho việc so khớp, cụ thể là những từ khóa (select|union|insert|delete|drop|update|alter|exec|execute), nhằm phát hiện các từ khóa thường dùng trong các cuộc tấn công SQL injection.

phase:1 chỉ ra rằng quy tắc này sẽ được áp dụng ở giai đoạn 1 (Request Headers) trong chu trình xử lý yêu cầu của ModSecurity.

log là hành động chỉ thị rằng sự kiện này sẽ được ghi vào log.

deny là hành động từ chối yêu cầu.

status:403 là mã trạng thái HTTP sẽ được trả về khi yêu cầu bị từ chối.

msg:'Da chan do nhap lieu co chua nghi van SQL injection.' là thông điệp sẽ được ghi vào log khi quy tắc này kích hoạt.

Rule thứ 2 mà em viết chính là rule chặn truy cập từ một địa chỉ IP nhất định:

```
SecRule REMOTE_ADDR "@ipMatch 10.11.12.138" \
    "id:'1000002',phase:1,log,deny,status:403,msg:'Truy cap bi tu choi tu dia chi IP nay.'"
```

Giải thích qua về rule này:

REMOTE_ADDR là biến môi trường chứa địa chỉ IP của người dùng đang gửi yêu cầu đến server.

@ipMatch là toán tử so khớp địa chỉ IP.

10.11.12.138 là địa chỉ IP cụ thể mà em muốn chặn nó.

Còn lại thì giống với rule đầu tiên của em.

Rule thứ 3 em viết chính là giới hạn dung lượng upload của một file:

```
SecRequestBodyLimit 1048576
SecRule REQUEST_HEADERS:Content-Length "@gt 1048576" \
    "id:'1000003',phase:1,log,deny,status:413,msg:'Dung luong noi dung yeu cau qua lon (vuot qua 1 MB).'"
```

Giải thích rule trên:

SecRequestBodyLimit đặt giới hạn cho kích thước tối đa của body yêu cầu. Ở đây nó được thiết lập là 1048576 bytes, tức là 1 megabyte (MB).

REQUEST_HEADERS:Content-Length là phần header của yêu cầu mà quy tắc này sẽ kiểm tra.

@gt 1048576 là toán tử chỉ ra rằng yêu cầu sẽ bị chặn nếu giá trị của Content-Length lớn hơn 1048576 bytes.

status:413 là mã trạng thái HTTP sẽ được trả về, chỉ ra rằng yêu cầu bị từ chối vì quá lớn.

Còn lại thì cũng giống như rule đầu tiên của em.

Tổng thể 3 rule mà em viết trong file custom_rules.conf sẽ như sau:

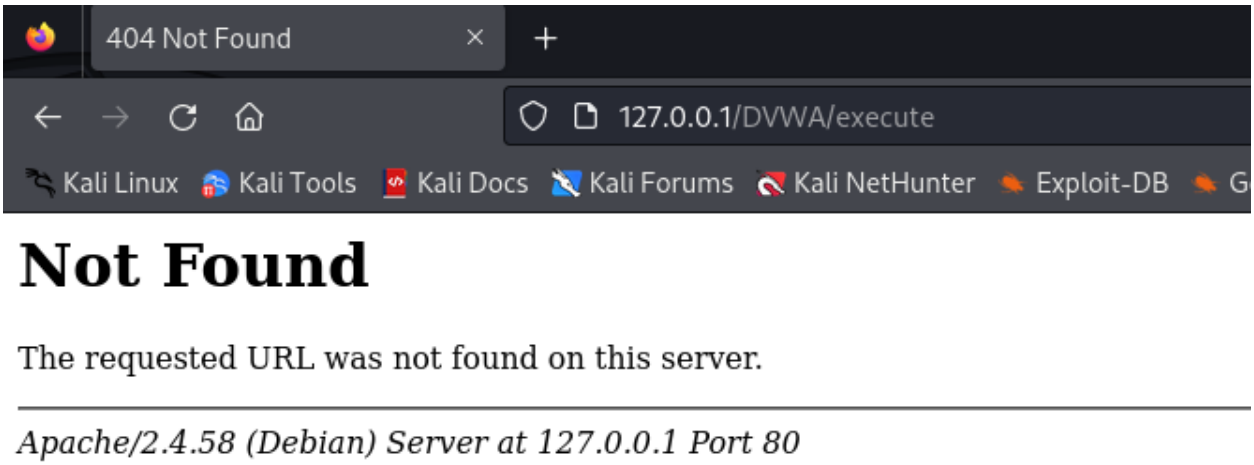
```
File  Actions  Edit  View  Help
  GNU nano 7.2                                                    custom_rules.conf
SecRule ARGS|REQUEST_URI "@rx (select|union|insert|delete|drop|update|alter|exec|execute)" \
    "id:'1000001',phase:1,log,deny,status:403,msg:'Da chan do nhap lieu co chua nghi van SQL injection.'"

SecRule REMOTE_ADDR "@ipMatch 10.11.12.138" \
    "id:'1000002',phase:1,log,deny,status:403,msg:'Truy cap bi tu choi tu dia chi IP nay.'"

SecRequestBodyLimit 1048576
SecRule REQUEST_HEADERS:Content-Length "@gt 1048576" \
    "id:'1000003',phase:1,log,deny,status:413,msg:'Dung luong noi dung yeu cau qua lon (vuot qua 1 MB).'"
```

Sau đó em sẽ include file custom_rules.conf của em vào trong file /etc/apache2/mods-available/security2.conf:

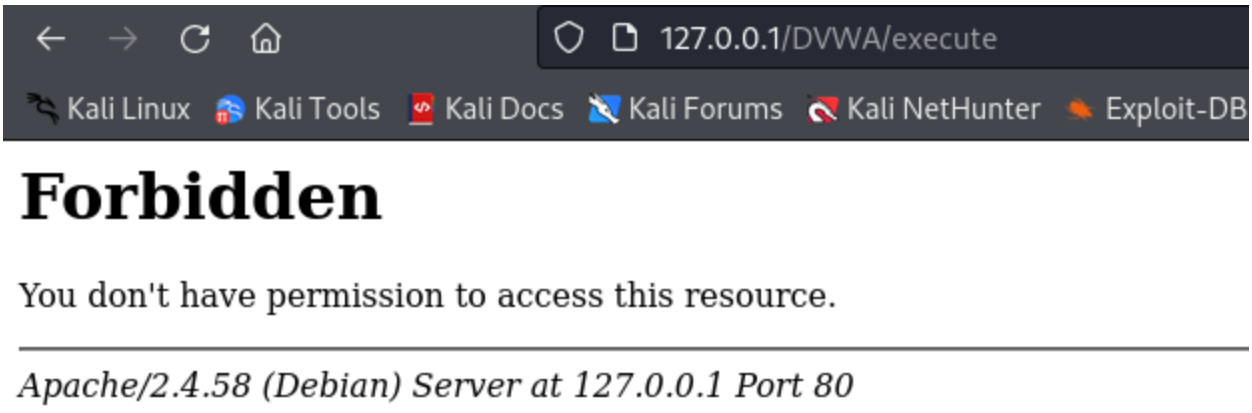
```
        Include /usr/share/modsecurity-crs/crs-setup.conf
        Include /usr/share/modsecurity-crs/rules/*.conf
        Include /home/nghianguyen/ATM_Nang_Cao/custom_rules.conf
</IfModule>
```

Cuối cùng là em sẽ test thử xem các rule mà mình viết có thực sự hoạt động hay không.

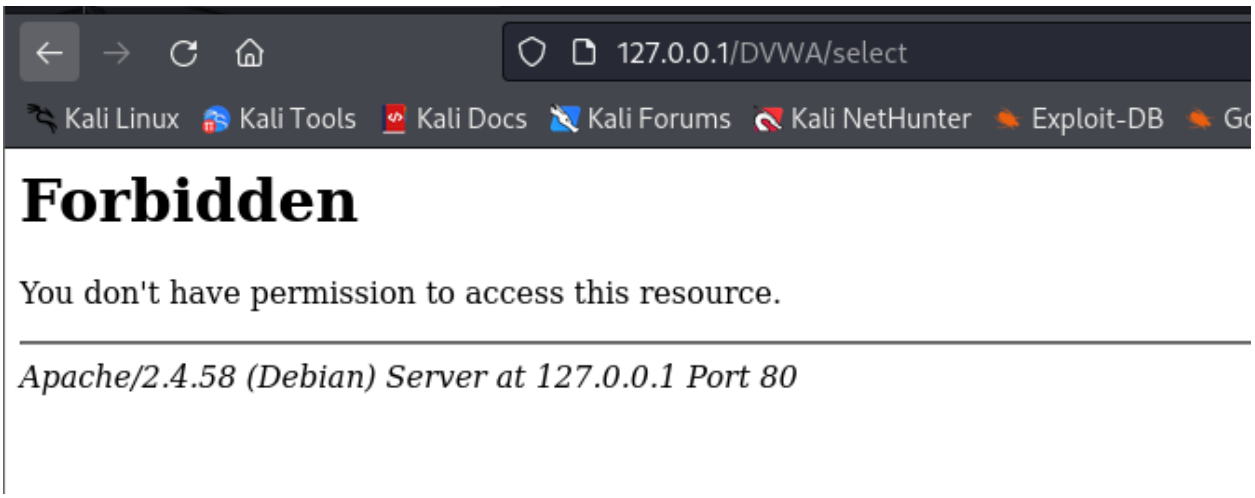
Với rule đầu tiên, ban đầu em sẽ nhập từ "execute" vào trên thanh url của trang web mà lúc này chưa có áp dụng rule vào, kết quả là Not Found như sau:



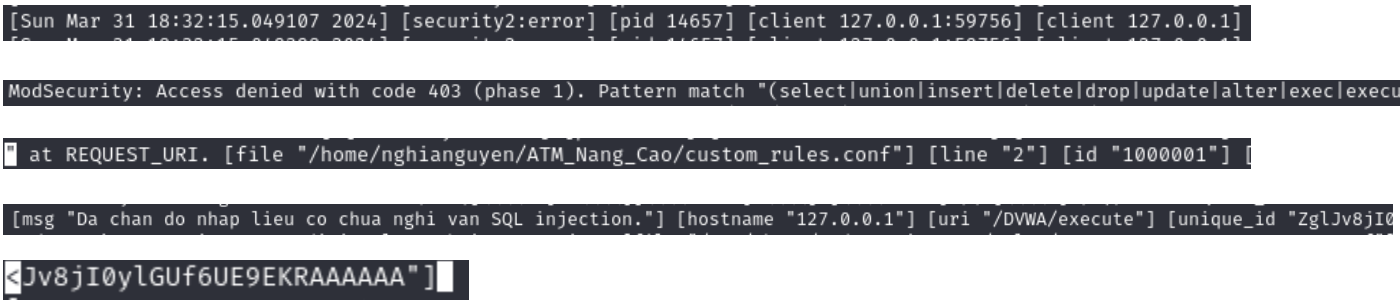
Nhưng sau khi áp dụng rule vào thì kết quả trả về sẽ là Forbidden:



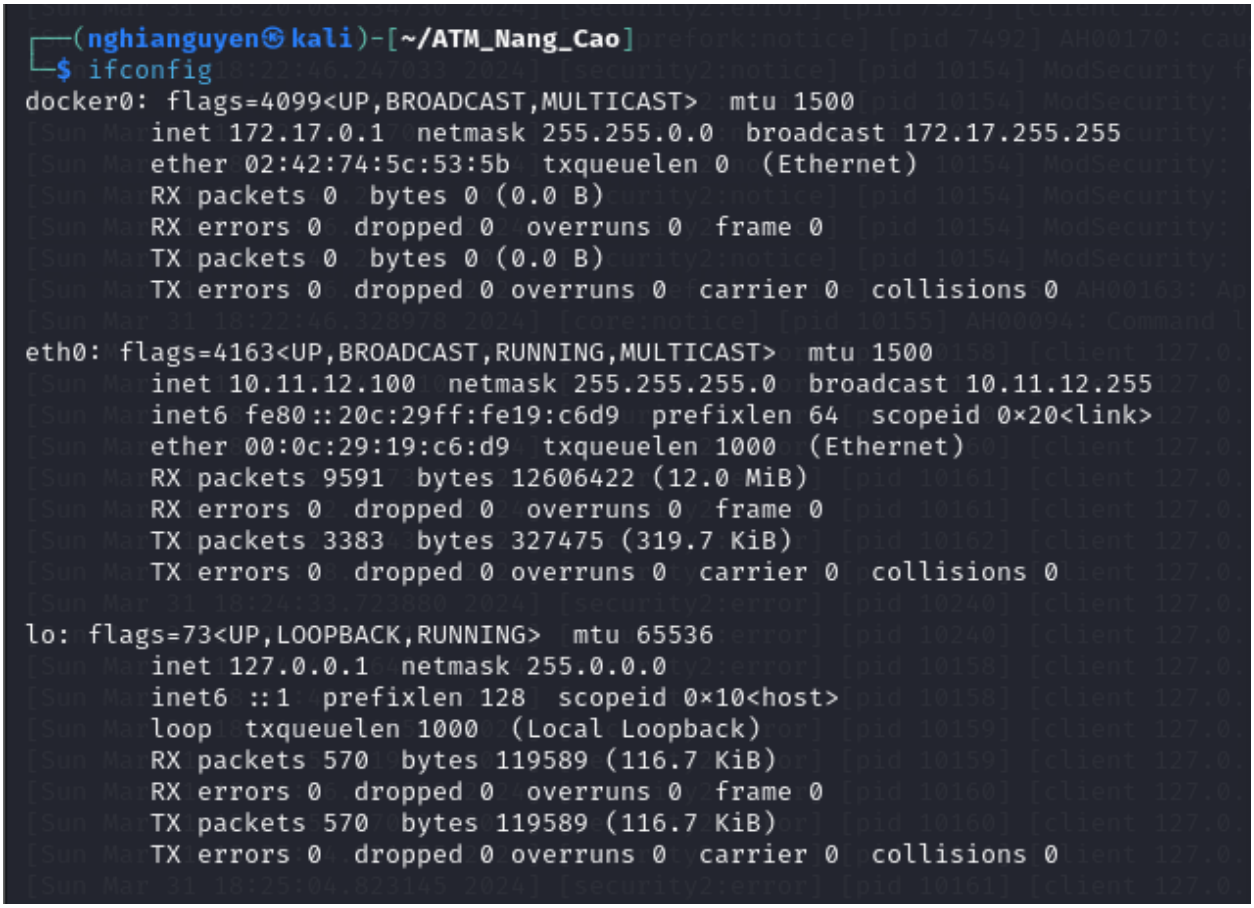
Tương tự với lại từ "select" cũng như thế:



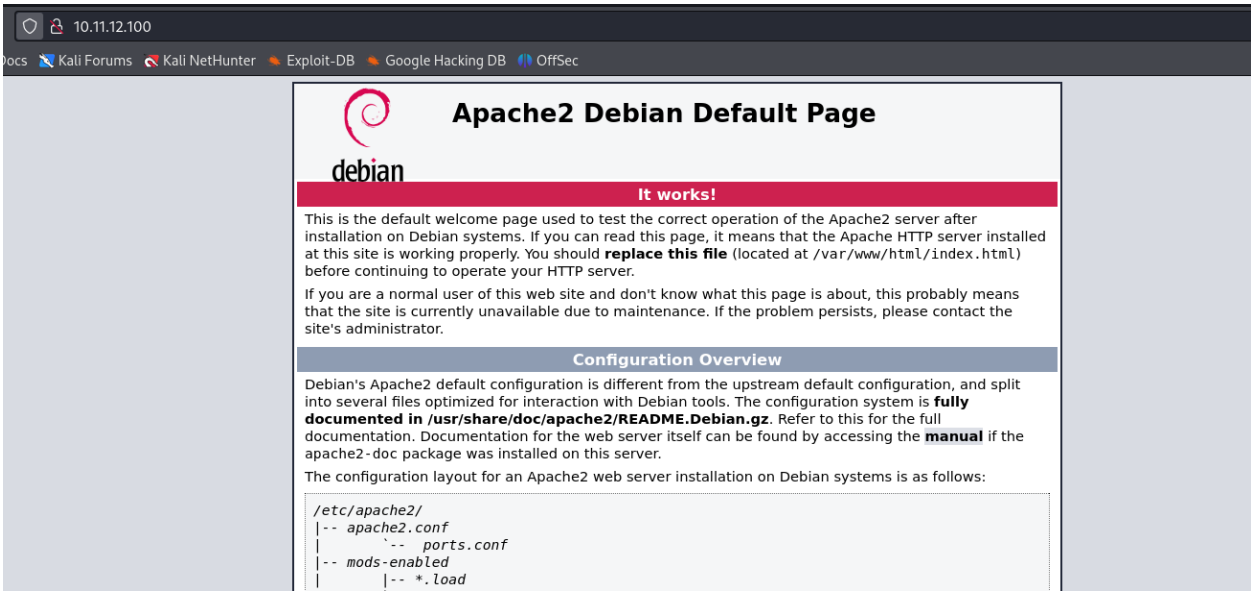
Sau đó em tiến hành mở file /var/log/apache2/error.log để xem thử log, kết quả nhận được là (Do cái cảnh báo này nó quá dài nên em đã cắt ra để cho dễ đọc):



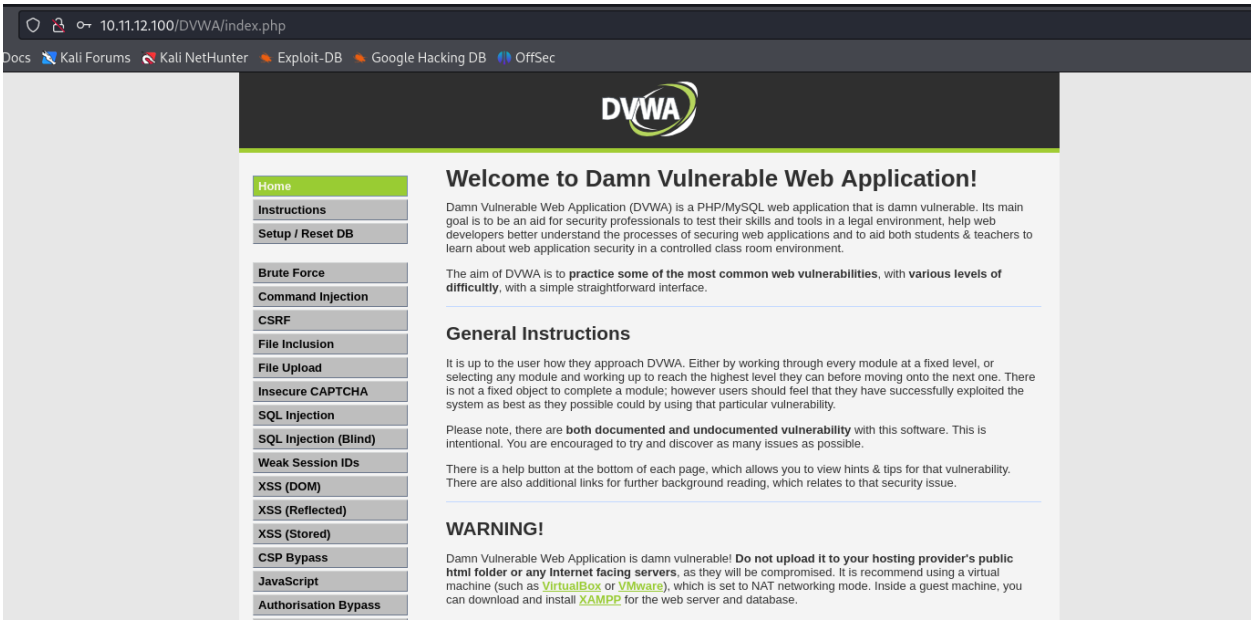
Tiếp đến là rule thứ 2, hiện tại máy em có địa chỉ ip là 10.11.12.100:



Truy cập vào thử vào chính địa chỉ ip của em:



Sau đó là sẽ vào thử DVWA:

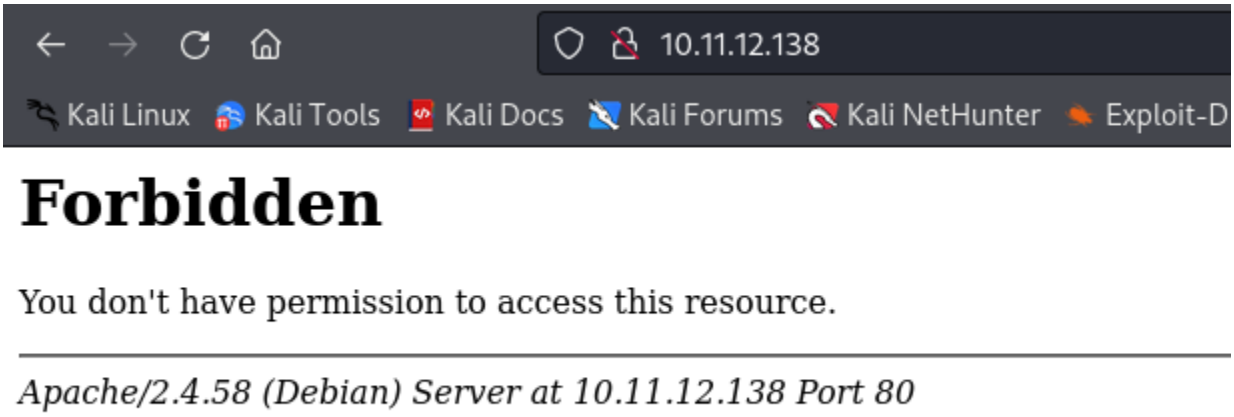


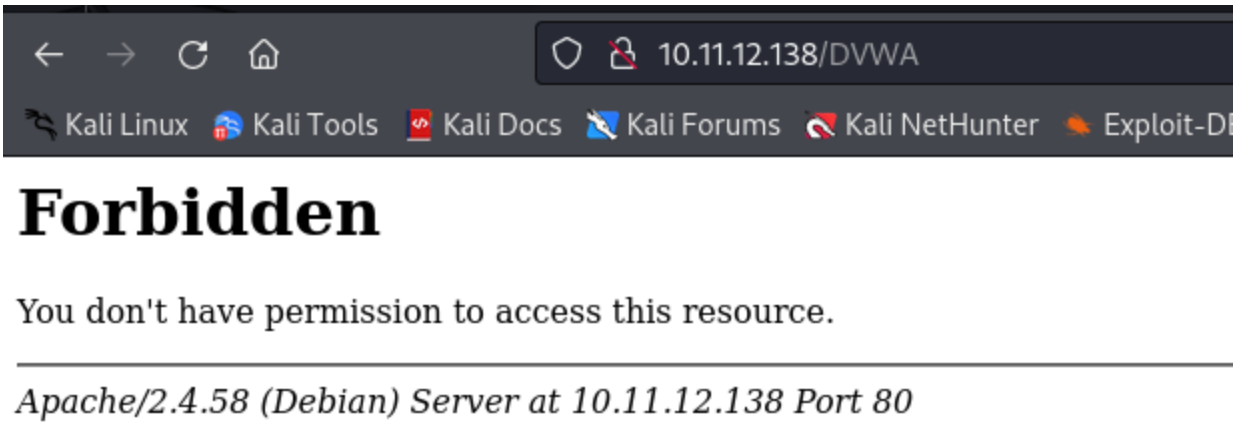
Cả 2 đều hoạt động.

Bây giờ em sẽ đổi địa chỉ ip của mình sang thành 10.11.12.138 để test xem liệu vẫn vào được hay không:

```
└─$ ifconfig
docker0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 172.17.0.1  netmask 255.255.0.0  broadcast 172.17.255.255
        ether 02:42:02:7f:2e:b3  txqueuelen 0  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.11.12.138  netmask 255.255.255.0  broadcast 10.11.12.255
        inet6 fe80::20c:29ff:fe19:c6d9  prefixlen 64  scopeid 0×20<link>
        ether 00:0c:29:19:c6:d9  txqueuelen 1000  (Ethernet)
        RX packets 29  bytes 2030 (1.9 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 159  bytes 12926 (12.6 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0×10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 4  bytes 240 (240.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 4  bytes 240 (240.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Kết quả là như sau:



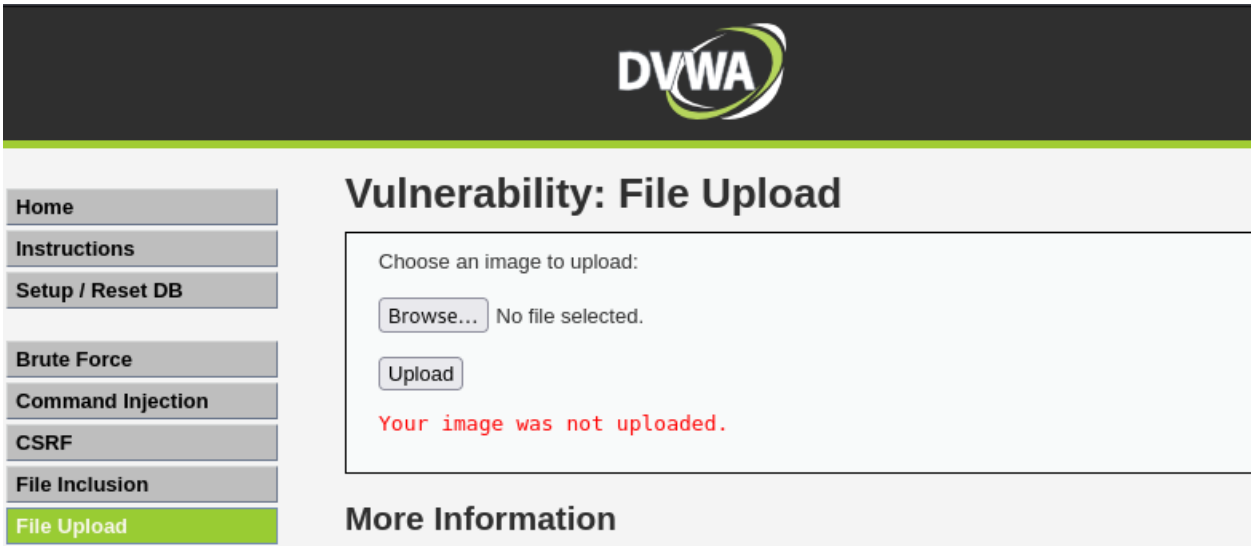


Kiểm tra log:

```
[Sun Mar 31 19:25:41.384024 2024] [security2:error] [pid 3671] [client 10.11.12.138:54938] [client 10.11.12.138] ModSecurity:

Access denied with code 403 (phase 1). IPmatch: "10.11.12.138" matched at REMOTE_ADDR. [file "/home/nghiangu

hianguyen/ATM_Nang_Cao/custom_rules.conf"] [line "5"] [id "1000002"] [msg "Truy cap bi tu choi tu dia chi IP nay."]

[hostname "10.11.12.138"] [uri "/DVWA"] [unique_id "ZglWRXymWlpEVStUqOoSFQAAAAE"]
```

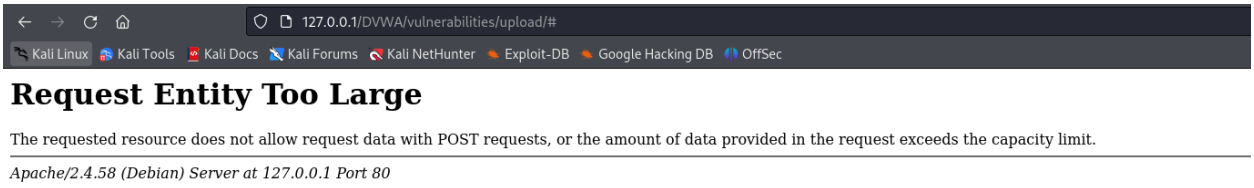
Bây giờ sẽ là rule thứ 3, em sử dụng File Upload của DVWA để test, và file em test sẽ là 1 file ảnh png có dung lượng là 1.3 mb:

Khi chưa áp dụng rule thì sau khi upload ảnh, nó có thông báo như sau:



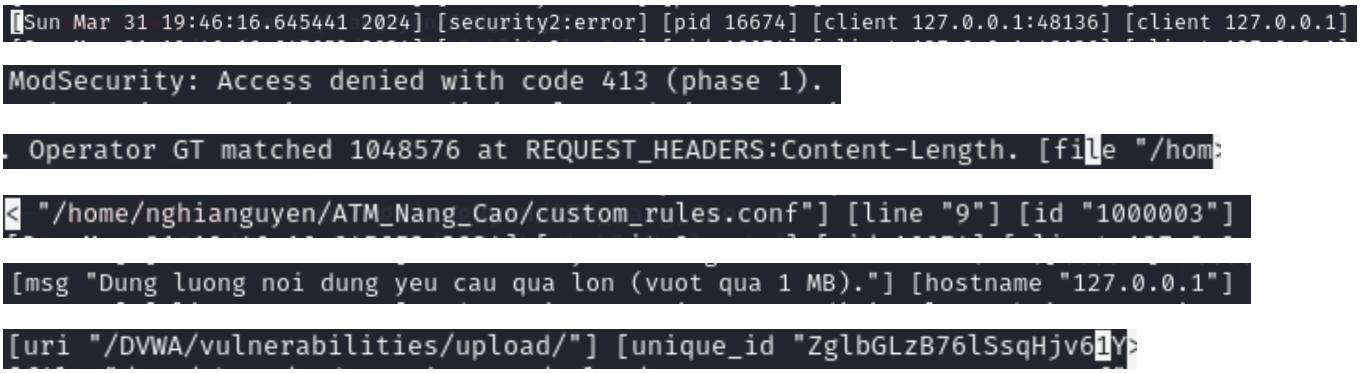
Em cũng không biết vì sao mà ảnh em upload không thành công, nhưng mà hiện tại thì em đã không bị chặn bởi modsecurity bằng rule của mình.

Sau khi áp dụng rule vào thì khi upload ảnh sẽ hiển thị thông báo sau:



Điều đó chứng tỏ là rule em đã chặn thành công.

Kiểm tra log:



Rule chặn ping tới : 192.168.1.1

SecRule REQUEST_URI|REQUEST_BODY "@contains 192.168.1.1" "id:1001,phase:2,deny,msg:'Access to 192.168.1.1 is not allowed'"