

PHPUnit 测试

1) 对注册模块进行测试

首先，查看数据库，发现有一条 user 表如下所示：

| Options | | | | | id | username | password |
|--------------------------|--|------|--|------|----|----------|---------------------|
| <input type="checkbox"/> | | Edit | | Copy | | Delete | 1 student1 password |
| <input type="checkbox"/> | | Edit | | Copy | | Delete | 2 admin password |
| <input type="checkbox"/> | | Edit | | Copy | | Delete | 3 student2 password |

编写如下代码，对 user 表中 id 为 2 的纪录进行注册测试，如下的是测试用数据库已有的用户名和密码再去注册用户，通常来说，应该是没办法注册成功的，即 \$result 值应该为 0。

```
1 <?php
2 include("conn.php"); // connect to database
3 class MyTest extends PHPUnit_Framework_TestCase
4 {
5     // test register database
6     public function testRegister1() {
7         $user = 'admin';
8         $password = 'password';
9         // mysql_query returns an bool value to $result, indicating 0 or 1
10        $result = mysql_query("insert into user(username,password)values('".$user."','".$password."');")
11        $expect = 0; // expect the insert won't work, for it is already in the database
12        $this->assertEquals($result, $expect);
13    }
14
15    // test register database
16    public function testRegister2() {
17        $user = 'admin';
18        $password = 'password';
19        // mysql_query returns an bool value to $result, indicating 0 or 1
20        $result = mysql_query("insert into user(username,password)values('".$user."','".$password."');")
21        $expect = 1; // expect the insert won't work, for it is already in the database
22        $this->assertEquals($result, $expect);
23    }
24 }
```

而实际运行上述测例的时候，结果如下所示：

```
DirkMacBook:vote Dirk$ phpunit RegisterTest.php
PHPUnit 3.6.0 by Sebastian Bergmann.

F.

Time: 0 seconds, Memory: 2.25Mb

There was 1 failure:

1) MyTest::testRegister1

Fatal error: Call to undefined function phpunit_mockobject_autoload() in /Applications/XAMPP/xamppfiles/lib/php/PHPUnit/Util/GlobalState.php on line 378
```

这说明\$result 值为 1，也就是说数据库中已有的用户名和密码再去注册居然可以

注册成功，说明在注册模块的数据库操作部分有 bug 存在。

然后，再用数据库中不存在的用户记录进行注册，并重复利用新注册的用户记录再注册，如下所示：

```
// test register database
public function testRegister3() {
    $user = 'sysu';
    $password = 'sser';
    // mysql_query returns an bool value to $result, indicating 0 or 1
    $result = mysql_query("insert into user(username,password)values('".$user."', '".$password."');");
    $expect = 1; // expect the insert will work
    $this->assertEquals($result, $expect);
}

// test register database
public function testRegister4() {
    $user = 'sysu';
    $password = 'sser';
    // mysql_query returns an bool value to $result, indicating 0 or 1
    $result = mysql_query("insert into user(username,password)values('".$user."', '".$password."');");
    $expect = 0; // expect the insert won't work, for it is already in the database
    $this->assertEquals($result, $expect);
}

// test register database
public function testRegister5() {
    $user = 'sysu';
    $password = 'sser';
    // mysql_query returns an bool value to $result, indicating 0 or 1
    $result = mysql_query("insert into user(username,password)values('".$user."', '".$password."');");
    $expect = 1; // test it
    $this->assertEquals($result, $expect);
}
```

运行上述测例，结果如下所示：

```
DirkMacBook:vote Dirk$ phpunit RegisterTest.php
PHPUnit 3.6.0 by Sebastian Bergmann.

.F.

Time: 0 seconds, Memory: 2.25Mb

There was 1 failure:

1) MyTest::testRegister4

Fatal error: Call to undefined function phpunit_mockobject_autoload() in /Applications/XAMPP/xamppfiles/lib/php/PHPUnit/Util/GlobalState.php on line 378
```

综上所述，用相同的用户名和密码可以重复进行注册，说明在注册模块的数据库操作部分有 bug 存在。

再对数据库进行查看，如下所示：

| | | | | id | username | password |
|--------------------------|--|--|--|----|----------|----------|
| <input type="checkbox"/> | | | | 1 | student1 | password |
| <input type="checkbox"/> | | | | 2 | admin | password |
| <input type="checkbox"/> | | | | 3 | student2 | password |
| <input type="checkbox"/> | | | | 39 | admin | password |
| <input type="checkbox"/> | | | | 40 | admin | password |
| <input type="checkbox"/> | | | | 41 | sysu | sser |
| <input type="checkbox"/> | | | | 42 | sysu | sser |
| <input type="checkbox"/> | | | | 43 | sysu | sser |

可以看到，出现这种情况的原因在于 id 是不同的，只要把 key 修改后，就可以解决这个 bug 了。

2) 对登陆模块进行测试

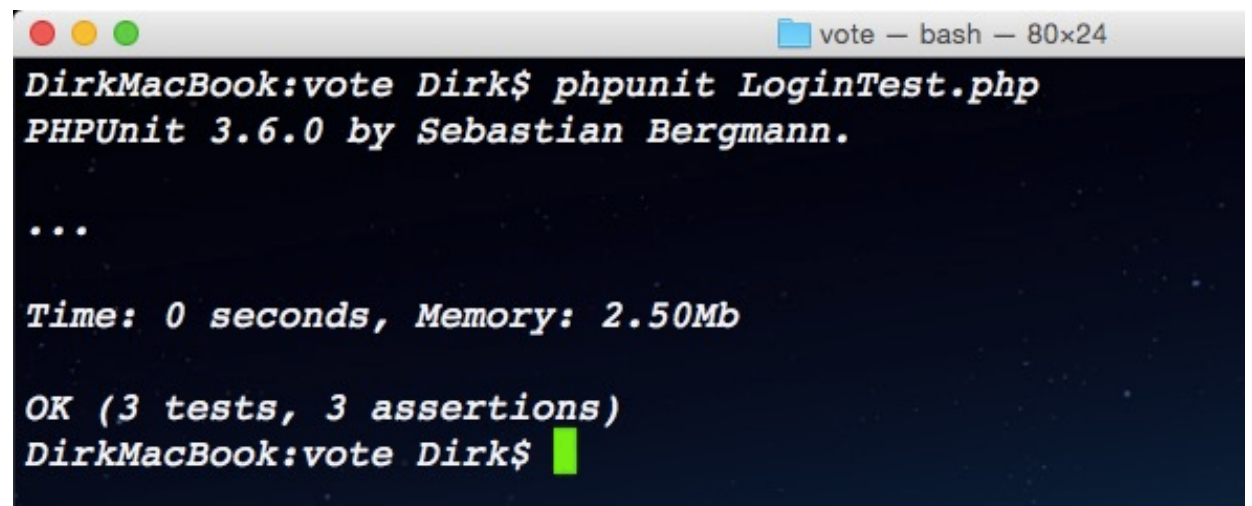
首先，对数据库中 user 表中存在的记录进行登陆测试，如下所示：

```
// log in normally
public function testLogin1() {
    $user = 'admin';
    $password = 'password';
    $result = mysql_query("select username from user where username = '". $user.'" and password = '". $password.'"");
    $result = mysql_result($result,0,0);
    $expect = 'admin';
    $this->assertEquals($expect, $result); // expect successful login
}

// log in normally
public function testLogin2() {
    $user = 'student1';
    $password = 'password';
    $result = mysql_query("select username from user where username = '". $user.'" and password = '". $password.'"");
    $result = mysql_result($result,0,0);
    $expect = 'student1';
    $this->assertEquals($expect, $result); // expect successful login
}

// log in normally
public function testLogin3() {
    $user = 'student2';
    $password = 'password';
    $result = mysql_query("select username from user where username = '". $user.'" and password = '". $password.'"");
    $result = mysql_result($result,0,0);
    $expect = 'student2';
    $this->assertEquals($expect, $result); // expect successful login
}
```

运行结果如下所示：



```
vote — bash — 80x24
DirkMacBook:vote Dirk$ phpunit LoginTest.php
PHPUnit 3.6.0 by Sebastian Bergmann.

...

Time: 0 seconds, Memory: 2.50Mb

OK (3 tests, 3 assertions)
DirkMacBook:vote Dirk$
```

三组测例都通过了，这说明已有的记录进行登录没有问题。

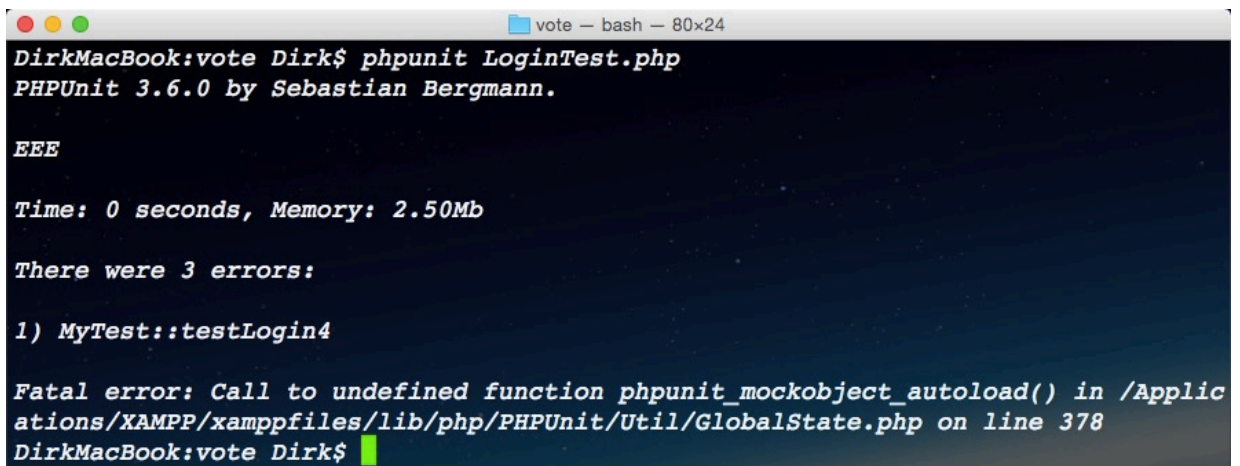
然后，故意输错用户名或者密码来进行登录，如下：

```
// log in abnormally
public function testLogin4() {
    $user = 'admin';
    $password = 'password1';
    $result = mysql_query("select username from user where username = '". $user.'"and password = '". $password.'"");
    $result = mysql_result($result,0,0);
    $expect = 'admin';
    $this->assertEquals($expect, $result); // expect unsuccessful login, it should fail
}

// log in abnormally
public function testLogin5() {
    $user = 'admin1';
    $password = 'password';
    $result = mysql_query("select username from user where username = '". $user.'"and password = '". $password.'"");
    $result = mysql_result($result,0,0);
    $expect = 'admin1';
    $this->assertEquals($expect, $result); // expect unsuccessful login, it should fail
}

// log in abnormally
public function testLogin6() {
    $user = 'admin1';
    $password = 'password1';
    $result = mysql_query("select username from user where username = '". $user.'"and password = '". $password.'"");
    $result = mysql_result($result,0,0);
    $expect = 'admin1';
    $this->assertEquals($expect, $result); // expect unsuccessful login, it should fail
}
}
```

测试结果如下所示:



```
DirkMacBook:vote Dirk$ phpunit LoginTest.php
PHPUnit 3.6.0 by Sebastian Bergmann.

EEE

Time: 0 seconds, Memory: 2.50Mb

There were 3 errors:

1) MyTest::testLogin4

Fatal error: Call to undefined function phpunit_mockobject_autoload() in /Applications/XAMPP/xamppfiles/lib/php/PHPUnit/Util/GlobalState.php on line 378
DirkMacBook:vote Dirk$
```

这说明尝试输错用户名和密码没办法登陆成功。

接下来，我们尝试进行 mysql 注入式攻击的测试，编写如下代码：

```
public function testLogin7() {
    $user = 'admin';
    $password = 'password';
    $result = mysql_query("select username from user where username = '". $user.'"or 1 = 1 and password = '". $password.'"");
    $result = mysql_result($result,0,0);
    $expect = 'student1';
    $this->assertEquals($result, $expect);
}
```

运行结果如下所示:

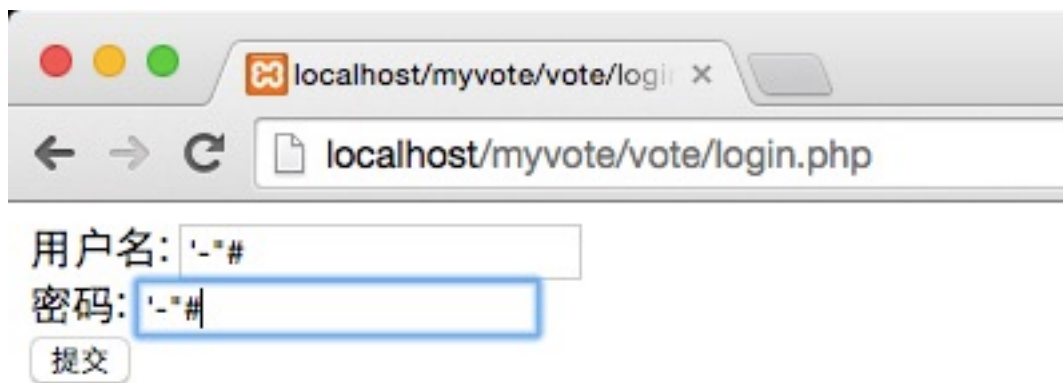

```
DirkMacBook:vote Dirk$ phpunit LoginTest.php
PHPUnit 3.6.0 by Sebastian Bergmann.

.

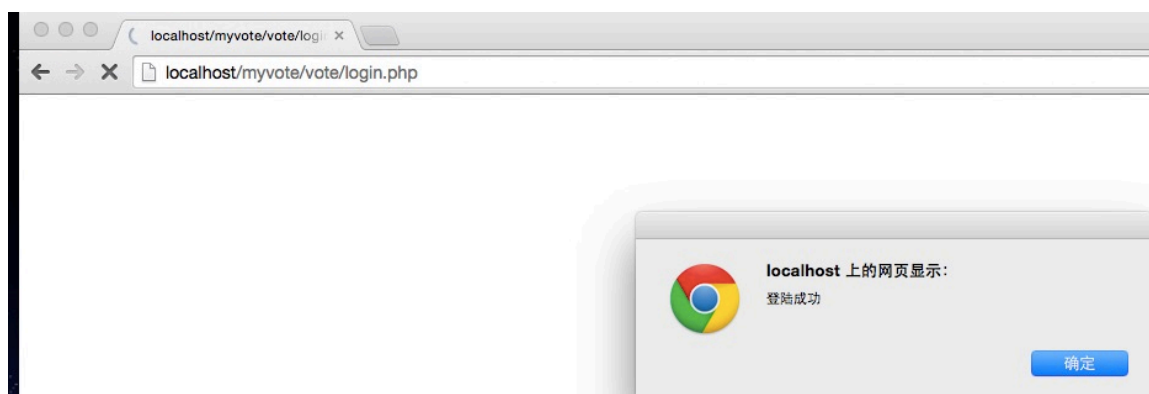
Time: 0 seconds, Memory: 2.25Mb

OK (1 test, 1 assertion)
DirkMacBook:vote Dirk$
```

这说明注入式攻击成功，为了进一步验证，我们再在浏览器中进行测试。用户名和密码都设置为 `'-'#` 来进行登录，如下所示：



然后点击提交按钮：



结果显示登陆成功，说明 SQL 注入式攻击成功了，错误的用户名和密码也能登陆，登陆模块存在 bug。SQL 注入式攻击是常见的漏洞，我们接下来对这个漏洞进行了修复。

3) 对 URL 进行注入式攻击测试

SQL 注入的本质是恶意攻击者将 SQL 代码插入或添加到程序的参数中，而程序并没有对传入的参数进行正确处理，导致参数中的数据会被当做代码来执行，并最终将执行结果返回给攻击者。

接下来，我们对 URL 进行 SQL 注入测试，我们本来网址的访问链接为 <http://localhost/myvote/vote/tp.php?id=37> 进入这个链接，我们的页面正常显示如下所示：



但是现在我们在 <http://localhost/myvote/vote/tp.php?id=37> 后面加一个单引号，如果存在漏洞，会发现返回一个和正常页面不同的页面。结果如下所示：



这说明 URL 请求的参数发生了错误，并且用 js 弹窗给出了提示，说明拦截到了恶意操作。

4) 参考

- [1] https://phpunit.de/manual/4.7/zh_cn/index.html
- [2] <http://www.rising.com.cn/newsletter/news/2012-05-24/11580.html>
- [3] <http://www.2cto.com/Article/201301/184097.html>