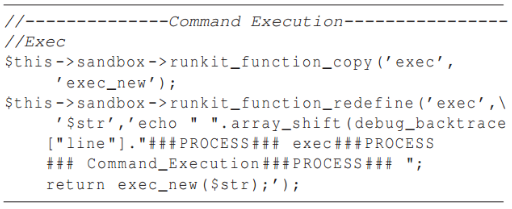
# **沙箱 - 工作计划**

## 目标

通过沙箱技术, 实现代码静态反混淆, 即通过重写PHP内部函数, 还原混淆代码, 进而对webshell可疑语句进行监控, 得到较纯静态检测方式更进一步的检测结果.

## 手段

使用开源PHP库PHPSandbox库, 在危险函数执行之前, 将其受到的参数记录并进行检测, 判断其传入参数的危险性.



## 预计目标

通过重写字符串操作函数, 以及一些危险指令执行函数, 从而将混淆后的危险代码还原, 例如以下代码:

<?php  
eval(gzinflate(base64\_decode(str\_rot13(’GIKKPhmVSslK +7 V2LJg+S3Lrv ...’))));  
?>

在沙箱中运行之后, 混淆的危险代码将会被还原为:

<?php  
h5(’http :// mycompanyeye .com/list ’ ,1\*900);  
functionh5($u ,$t){$nobot=isset  
($\_REQUEST[’nobot ’])?true:false;  
$debug=isset($\_REQUEST[’debug ’])?true:false;  
$t2 =3600\*5;  
$t3 =3600\*12;  
$tm =(! @ini\_get(’upload\_tmp\_dir ’))?’/tmp/’:  
@ini\_get(’upload\_tmp\_dir ’);  
...  
?>

## 难点

1. 与passthru等危险函数不同的是, eval, assert这样的php关键字并没有提供较为简便的重写方法. 其重写方式还在思考中.
2. 有些函数需要外部输入才能执行. 需要实现一种方式自动识别这类函数并提供适当的外部输入.

## 计划

本次项目计划完成时间为4天, 每日的工作计划如下:

|  |  |  |
| --- | --- | --- |
| **日期** | **任务** | **产出** |
| **16-05-18** | 完全熟悉PHPSandbox用法, 重写passthru函数 | 重写后的Preg\_replace函数 |
| **16-05-19** | 重写exec, shell\_exec, phpinfo, curl\_exec函数 | 重写后的exec, shell\_exec, phpinfo, curl\_exec函数 |
| **16-05-20** | 重写curl\_multi\_exec, create\_function, pcntl\_exec, popen, ssh2\_exec等函数 | 重写后的curl\_multi\_exec, create\_function, pcntl\_exec, popen, ssh2\_exec等函数 |
| **16-05-21** | 整理所有重写函数, 提供一个php文件以及其作为沙箱的使用方法 | 一个PHP文件, 通过使用其中的exec方法, 将外部代码放入沙箱内执行. |