

命令执行漏洞，用户通过浏览器在远程服务器上执行任意系统命令，严格意义上，与代码执行漏洞还是有一定的区别。

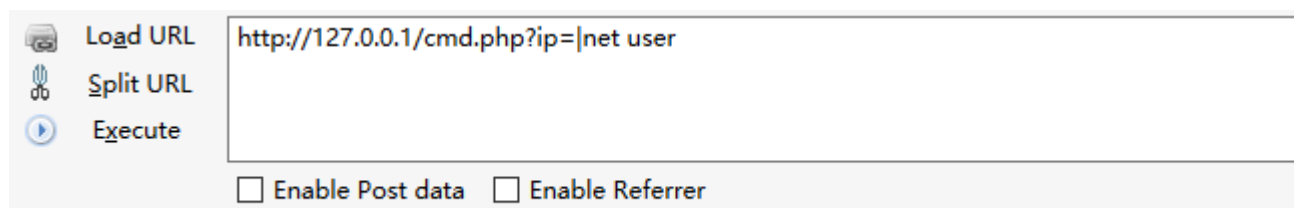
0x01漏洞实例

例1:

```
<?php
    $target=$_REQUEST['ip'];
    $cmd = shell_exec('ping '.$target);
    echo "<pre>{$cmd}</pre>";
?>
```

提交 <http://127.0.0.1/cmd.php?ip=|net user>

提交以后，命令变成了 shell_exec('ping '|net user)



\\DESKTOP-464SH0H 的用户帐户

Aaron	Administrator	DefaultAccount
defaultuser0	Guest	WDAGUtilityAccount

命令成功完成。

0x02 常用命令执行函数

exec()、system()、popen()、passthru()、proc_open()、pcntl_exec()、shell_exec()、反引号` 实际上是使用 shell_exec()函数

system() 输出并返回最后一行shell结果。exec() 不输出结果，返回最后一行shell结果，所有结果可以保存到一个返回的数组里面。passthru() 只调用命令，把命令的运行结果原样地直接输出到标准输出设备上。

popen()、proc_open() 不会直接返回执行结果，而是返回一个文件指针。

```
<?php
    #system('net user');
    #passthru ('dir');

    #echo exec('whoami');
    #echo shell_exec('whoami');
    #echo `whoami`;
?>
```

0x03 漏洞利用及绕过姿势

| 命令管道符

<>>> 文件重定向符

测试：0 | dir c:

代码只过滤了部分特殊字符，可以考虑用其他字符进行测试，这边列举一下Window/Linux可利用的特殊字符：

windows支持：

| 直接执行后面的语句 ping 127.0.0.1|whoami

|| 前面出错执行后面的，前面为假 ping 2 || whoami

& 前面的语句为假则直接执行后面的,前面可真可假 ping 127.0.0.1&whoami

&&前面的语句为假则直接出错，后面的也不执行，前面只能为真 ping 127.0.0.1&&whoami

Linux支持：

; 前面的执行完执行后面的 ping 127.0.0.1;whoami

| 管道符，显示后面的执行结果 ping 127.0.0.1|whoami

11 当前面的执行出错时执行后面的 ping 1 || whoami

& 前面的语句为假则直接执行后面的,前面可真可假 ping 127.0.0.1&whoami

&&前面的语句为假则直接出错，后面的也不执行，前面只能为真 ping 127.0.0.1&&whoami

0x04 如何防止命令执行漏洞

PHP内置的两个函数可以有效防止命令执行：

escapeshellarg() 将给字符串增加一个单引号并且能引用或者转码任何已经存在的单引号，这样以确保能够直接将一个字符串传入 shell 函数，并且还是确保安全的。对于用户输入的部分参数就应该使用这个函数。资料参考：<http://cn.php.net/manual/zh/function.escapeshellarg.php>

escapeshellcmd() 对字符串中可能会欺骗 shell 命令执行任意命令的字符进行转义。此函数保证用户输入的数据在传送到 exec() 或 system() 函数，或者 执行操作符 之前进行转义。资料参考：<http://cn.php.net/manual/zh/function.escapeshellcmd.php>

当然，修复方法还有很多方式，修复方式一般有两种思维：

1、黑名单：过滤特殊字符或替换字符 2、白名单：只允许特殊输入的类型/长度

修复代码示例一：

```
<?php
    $target=$_REQUEST['ip'];
    $octet = explode( ".", $target );
    if( ( is_numeric( $octet[0] ) ) && ( is_numeric( $octet[1] ) ) && ( is_numeric(
$octet[2] ) ) && ( is_numeric( $octet[3] ) ) && ( sizeof( $octet ) == 4 ) ) {
        $target = $octet[0] . '.' . $octet[1] . '.' . $octet[2] . '.' . $octet[3];
        $cmd = shell_exec('ping '.$target);
        echo "<pre>{$cmd}</pre>";
    }
    else {
        echo '<pre>ERROR: You have entered an invalid IP.</pre>';
    }
?>
```

修复代码示例二:

```
<?php
    $target=$_REQUEST['ip'];
    $cmd = shell_exec('ping '. escapeshellcmd($target));
    echo "<pre>{$cmd}</pre>";
?>
```

新文章将同步更新到我的个人公众号上，欢迎各位朋友扫描我的公众号二维码关注一下我，随时获取最新动态。

