WEB安全基础-文件上传

#2课时

文件上传靶场

upload-labs：
https://github.com/c0ny1/upload-labs/releases

文件上传简介

什么是文件上传

将客户端数据以文件形式封装，通过网络协议发送到服务器端。在服务器端解析数据，最终在服务端硬盘上作为真实的文件保存。

通常一个文件以HTTP协议进行上传时，将以POST请求发送至Web服务器，Web服务器收到请求并同意后，用户与Web服务器将建立连接，并传输数据。

什么是文件上传漏洞

文件上传漏洞是指用户上传了一个可执行的脚本文件，并通过此脚本文件获得了执行服务器端命令的能力。这种攻击方式是最为直接和有效的，"文件上传"本身没有问题，有问题的是文件上传后，服务器怎么处理、解释文件。如果服务器端脚本语言未对上传的文件进行严格的验证和过滤，就容易造成上传任意文件的情况。

通常web站点会有用户注册功能，而当用户登录之后大多数情况下会存在类似头像上传、附件上传之类的功能，这些功能点往往存在上传验证方式不严格的安全缺陷，导致攻击者通过各种手段绕过验证，上传非法文件，这是在web渗透中非常关键的突破口。

文件上传产生漏洞的原因

1. 服务器配置不当
2. 文件上传限制被绕过
3. 开源编辑器的上传漏洞
4. 文件解析漏洞导致文件执行
5. 过滤不严或被绕过

文件上传漏洞危害

攻击者通过上传恶意文件传递给解释器去执行，然后就可以在服务器上执行恶意代码，进行数据库执行、服务器文件管理、命令执行等恶意操作。从而控制整个网站及服务器。 这个恶意的文件（php、asp、aspx、jsp等），又被称WebShell。

文件上传检测方式

一般一个文件上传过程中的检测方式有：

- 客户端JavaScript检测（检测文件扩展名）
- 服务端MIME类型检测（检测content-type内容）
- 服务端目录路径检测（检测跟path参数相关的内容）
- 服务端文件扩展名检测 (检测跟文件extension相关的内容)
- 服务端文件内容检测（检测内容是否合法是否含有恶意代码）

Webshell

Webshell简介

WebShell就是以ASP、PHP、JSP或者CGI等网页文件形式存在的一种命令执行环境，也可以将其称之为一种网页后门。攻击者在入侵了一个网站后，通常会将这些asp或php后门文件与网站服务器web目录下正常的网页文件混在一起，然后使用浏览器来访问这些后门，得到一个命令执行环境，以达到控制网站服务器的目的（可以上传下载或者修改文件，操作数据库，执行任意命令等）。

常用一句话Webshell

```
php一句话木马：  <?php @eval($_POST[value]);?>
asp一句话木马：  <%eval request("value")%>
aspx一句话木马：  <%@ Page Language="Jscript"%><%eval(Request.Item["value"])%>
```

制作一句话图片马：`copy 1.jpg/b+1.php/a 2.jpg`

Webshell原理

以一个原始而又简单的php一句话木马为例：

```
<?php
    @eval($_POST['cmd']);
?>
```

> php的代码要写在`<?php ?>`里面，服务器才能认出来这是php代码，然后才去解析。

> `@`符号的意思是不报错,因为变量没有定义而被使用，服务器会提醒XXX变量未定义。

```
$_POST['cmd'];
```

> php里面有几个超全局变量，`$_GET、$_POST`就是其中之一，意思是用`post`的方法接收变量cmd传递来的字符。

```
eval()
```

> 把字符串作为PHP代码执行

```
exec()
```

> 执行系统命令

```
system()
```

文件上传绕过

绕过客户端检测(JS检测)

原理：通常在上传页面里含有专门检测文件上传的`JavaScript`代码，最常见的就是检测文件类型和扩展名是否合法。

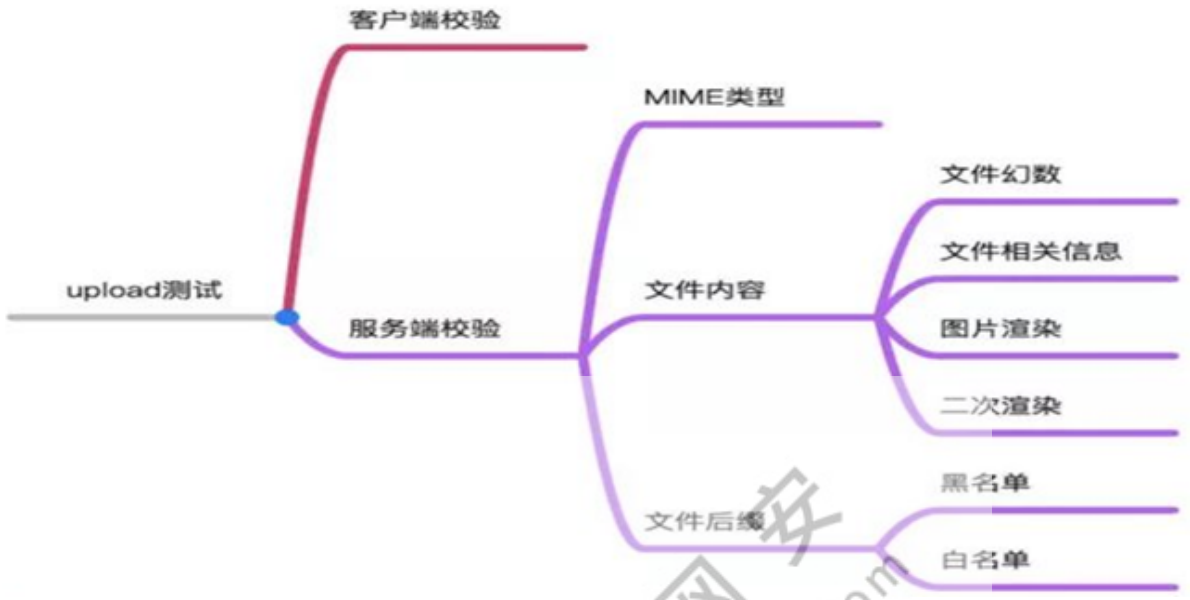
方法：在本地浏览器客户端禁用`JS`即可；可使用火狐浏览器的`Noscript`插件、IE中禁用JS等方式实现，利用`burpsuite`可以绕过一切客户端检测。

> http://120.27.61.239:8003/source/01/index.php

绕过服务端检测

服务端检测

服务端的代码通常检测三个点：MIME类型、文件内容、文件后缀



绕过MIME类型检测

常见MIME类型

```
1.  超文本标记语言文本 .html text/html
2.  普通文本 .txt text/plain
3.  PDF文档 .pdf application/pdf
4.  Microsoft Word文件 .word application/msword
5.  PNG图像 .png image/png
6.  GIF图形 .gif image/gif
7.  JPEG图形 .jpeg,.jpg image/jpeg
8.  au声音文件 .au audio/basic
9.  MPEG文件 .mpg,.mpeg video/mpeg
10. AVI文件 .avi video/x-msvideo
11. GZIP文件 .gz application/x-gzip
```

原理：检测图片类型文件上传过程中http包的`Content-Type`字段的值，来判断上传文件是否合法。

方法：用burpsuite截取并修改数据包中文件的`content-type`类型进行绕过。

> http://120.27.61.239:8003/source/02/index.php

绕过文件后缀检测-黑名单

黑名单策略：

文件扩展名在黑名单中为不合法，一般有个专门的黑名单列表，里面会包含常见的危险脚本文件。

```
if (isset($_POST['submit'])) {
    if (file_exists(UPLOAD_PATH)) {
        $deny_ext = array(".php",".php5",".php4",".php3",".php2",".html",".htm",".phtml",".pht",".pHp",".pHp5",".pHp4",".pHp3","
            .pHp2",".Html",".Htm",".pHtml",".jsp",".jspa",".jspx",".jsw",".jsv",".jspf",".jtml",".jSp",".jSpx",".jSpa",".jSw","
            .jSv",".jSpf",".jHtml",".asp",".aspx",".asa",".asax",".ascx",".ashx",".asmx",".cer",".aSp",".aSpx",".aSa",".aSax","
            .aScx",".aShx",".aSmx",".cEr",".sWf",".swf",".htaccess");
```

1. 后缀大小写绕过：(.Php)
   在对后缀的判断中，如果只是对字符串进行单独的比较来判断是不是限制文件，可以采用后缀名大小写绕过形式。
2. 空格绕过：(.php)
   如果黑名单没有对后缀名进行去空处理，可以通过在后缀名后加空进行绕过。
3. 点绕过：(.php.)
   如果黑名单没有对后缀名进行去.处理，利用Windows系统的文件名特性，会自动去掉后缀名最后的.，通过在文件名后加.进行绕过。
4. `::$DATA` 绕过：
   如果黑名单没有对后缀名进行去::$DATA处理，利用Windows下NTFS文件系统的一个特性，可以在后缀名后加::$DATA，绕过对黑名单的检测。
5. 配合Apache解析漏洞：
   Apache解析有一个特点，解析文件时是从右往左判断，如果为不可识别解析再往左判断，如aa.php.owf.rar文件，Apache不可识别解析'.owf'和'.rar'这两种后缀，会解析成.php文件。
6. `.htaccess` 文件
   配合名单列表绕过，上传一个自定义的`.htaccess`，就可以轻松绕过各种检测

`.htaccess` 文件(或者"分布式配置文件")，全称是 `Hypertext Access`(超文本入口)。提供了针对目录改变配置的方法， 即，在一个特定的文档目录中放置一个包含一个或多个指令的文件， 以作用于此目录及其所有子目录。作为用户，所能使用的命令受到限制。

比如新建一个`.htaccess`文件：

```
<FilesMatch "as.png">
setHandler application/x-httpd-php
</FilesMatch>
```

通过一个`.htaccess`文件调用`php`的解析器去解析一个文件名中只要包含"`as.png`"这个字符串的任意文件，所以无论文件名是什么样子，只要包含"as.png"这个字符串，都可以被以`php`的方式来解析，一个自定义的`.htaccess`文件就可以以各种各样的方式去绕过很多上传验证机制。

> http://120.27.61.239:8003/source/04/index.php

绕过文件后缀检测-白名单

白名单策略：文件扩展名不在白名单中为不合法。

绕过方法：服务端判断文件类型是从后往前判断，而对文件解析是从前往后解析，可以利用00截断的方式进行绕过，包括%00截断与0x00截断。 php小于5.3.29

```
%00截断：
url发送到服务器后被服务器解码，这时还没有传到验证函数，也就是说验证函数里接收到的不是%00字符，而是%00解码后的内容，即解码成了0x00。

0x00截断：
系统在对文件名进行读取时，如果遇到0x00，就会认为读取已经结束。但要注意是文件的十六进制内容里的00，而不是文件名中的00。
```

绕过文件内容检测

一般通过检测文件内容来判断上传文件是否合法。

主要有两种检测方法：

1. 通过检测上传文件内容开始处的文件幻数来判断。

通常情况下，通过判断前10个字节，基本就能判断出一个文件的真实类型。

2. 文件加载检测

一般是调用API或函数对文件进行加载测试。常见的是图像渲染测试，再严格点的甚至是进行二次渲染。

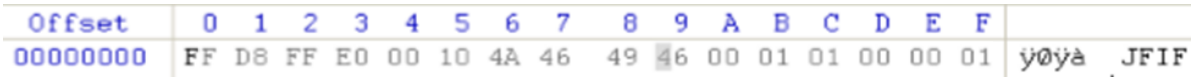
- 文件幻数检测

主要是检测文件内容开始处的文件幻数

文件格式幻数（外语：magic number），它可以用来标记文件或者协议的格式，很多文件都有幻数标志来表明该文件的格式。

常见图片类型的文件幻数如下：
1 要绕过 jpg 文件幻数检测就要在文件开头写上下面的值：

```
Value = FF D8 FF E0 00 10 4A 46 49 46
```

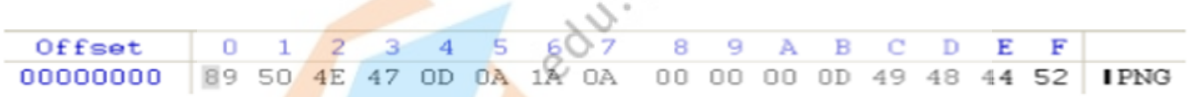


2 要绕过 gif 文件幻数检测就要在文件开头写上下面的值：

```
Value = 47 49 46 38 39 61
```

3 要绕过 png 文件幻数检测就要在文件开头写上下面的值：

```
Value = 89 50 4E 47
```



然后在文件幻数后面加上自己的一句话木马代码就行了。

> http://120.27.61.239:8003/source/03/index.php

- 文件加载检测

一般是调用API 或函数去进行文件加载测试，我们常见的是图像渲染测试，严格的进行二次渲染。

对渲染/加载测试的攻击方式是代码注入绕过

对二次渲染的攻击方式是攻击文件加载器自身

1. 对渲染/加载测试攻击 - 代码注入绕过

可以用图像处理软件对一张图片进行代码注入

这类攻击的原理是：在不破坏文件本身的渲染情况下找一个空白区进行填充代码，一般是图片的注释区，这样能保证本身文件结构是完整的，对于渲染测试基本上都能绕过

2. 二次渲染的攻击方式 - 攻击文件加载器自身

这种情况下无法用代码注入绕过，二次渲染相当于吧原本属于图像数据的部分抓出来，在用自己的API或函数进行重新渲染，而非图像数据部分直接被隔离开了。

我们可以用溢出攻击对文件加载器进行攻击，上传自己的恶意文件后，服务器上的文件加载器会主动进行加载测试，加载测试时被溢出攻击执行shellcode。

解析漏洞

Apache解析漏洞

形式：`test.php.qwe.asd`，任意不属于Apache解析黑名单且也不属于白名单的名称

原理：Apache 解析文件的规则是从右到左开始判断解析,如果后缀名为不可识别文件解析,就再往左判断。比如`test.php.qwe.asd`，".qwe"和".asd" 这两种后缀是apache不可识别解析，apache就会把`test.php.qwe.asd`解析成php。

条件：apache通过mod_php来运行脚本，其2.4.0-2.4.29中存在apache换行解析漏洞，在解析php时`xxx.php\x0A`将被按照PHP后缀进行解析，导致绕过一些服务器的安全策略`www.xxx.com/test.php.qwe.asd`

以moudel方式连接，配置文件httpd.conf 中`LoadModule rewrite_module modules/mod_rewrite.so`前的注释去掉，寻找关键词：`AllowOverride`，并把后面的参数从None全部改成All

Nginx解析漏洞

形式：`任意文件名/任意文件名.php`

一个在任意文件名后面添加`/任意文件名.php`的解析漏洞，比如原本文件名是`test.jpg`，可以添加为`test.jpg/x.php`进行解析攻击。

原理：Nginx < 0.8.37默认是以CGI的方式支持PHP解析的，普遍的做法是在Nginx配置文件中通过正则匹配设置`SCRIPT_FILENAME`。当访问`www.xx.com/phpinfo.jpg/1.php`这个URL时，`$fastcgi_script_name`会被设置为`phpinfo.jpg/1.php`，然后构造成`SCRIPT_FILENAME`传递给PHP CGI，但是PHP为什么会接受这样的参数，并将phpinfo.jpg作为PHP文件解析呢?这就要说到`fix_pathinfo`这个选项了。 如果开启了这个选项，那么就会触发在PHP中的如下逻辑： PHP会认为`SCRIPT_FILENAME`是`phpinfo.jpg`，而`1.php`是`PATH_INFO`，所以就会将`phpinfo.jpg`作为PHP文件来解析了

```
漏洞形式：

www.xxxx.com/UploadFiles/image/1.jpg/1.php
www.xxxx.com/UploadFiles/image/1.jpg.php
www.xxxx.com/UploadFiles/image/1.jpg/ \0.php
```

形式：`任意文件名%00.php`

对低版本的 Nginx 可以在任意文件名后面添加`%00.php`进行解析攻击。（Nginx版本 <=0.8.37空字节代码执行漏洞）

IIS 6.0解析漏洞

1. 目录解析

形式：`www.xxx.com/xx.asp/xx.jpg`

原理: 服务器默认会把`.asp`目录下的文件都解析成asp文件。

> https://www.cnblogs.com/milantgh/p/4347520.html

2. 文件解析

形式：`www.xxx.com/xx.asp;.jpg`

原理：服务器默认不解析;号后面的内容，因此`xx.asp;.jpg`便被解析成asp文件了。

IIS6.0 默认的可执行文件除了asp还包含这三种：

```
/test.asa
/test.cer
/test.cdx
```

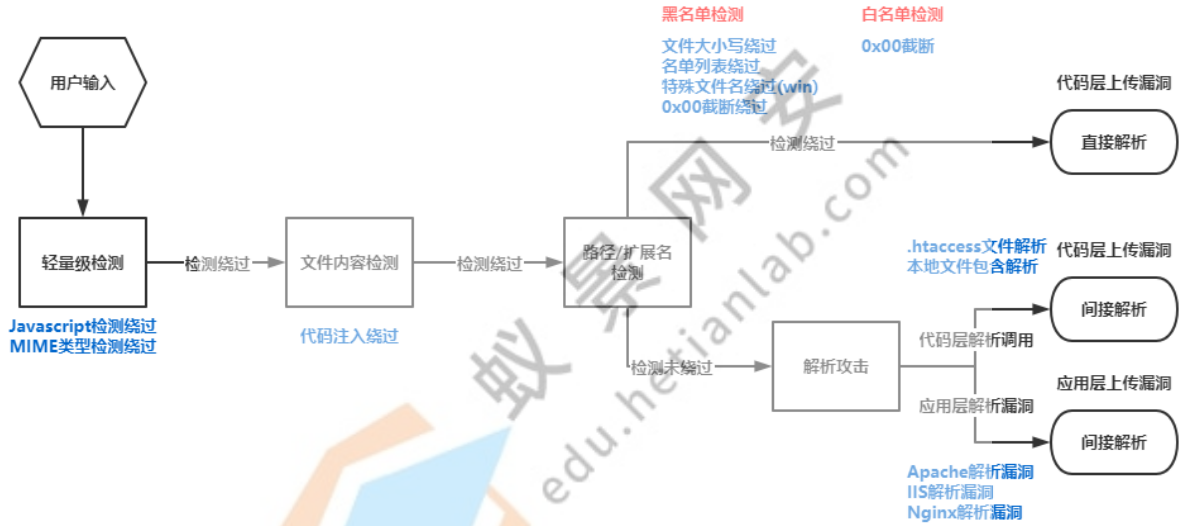
IIS 7.0/7.5解析漏洞

形式：`任意文件名/任意文件名.php`

原理：IIS7.0/7.5 是对 php 解析时有一个类似于 Nginx 的解析漏洞，对任意文件名只要在 URL后面追加上字符串`/任意文件名.php`就会按照 php 的方式去解析

由于php配置文件中，开启了`cgi.fix_pathinfo`，而这并不是nginx或者iis7.5本身的漏洞。

文件上传漏洞总结



文件上传漏洞防御

1. 文件上传的目录设置为不可执行

> 只要web容器无法解析该目录下面的文件，即使攻击者上传了脚本文件，服务器本身也不会受到影响，这一点至关重要。

2. 判断文件类型

> 在判断文件类型时，可以结合使用MIME-Type、后缀检查等方式。在文件类型检查中，强烈推荐白名单方式，黑名单的方式已经无数次被证明是不可靠的。此外，对于图片的处理，可以使用压缩函数或者resize函数，在处理图片的同时破坏图片中可能包含的HTML代码。

3. 使用随机数改写文件名和文件路径

> 文件上传如果要执行代码，则需要用户能够访问到这个文件。在某些环境中，用户能上传，但不能访问。如果应用了随机数改写了文件名和路径，将极大地增加攻击的成本。再来就是像shell.php.rar.rar和crossdomain.xml这种文件，都将因为重命名而无法攻击。

4. 单独设置文件服务器的域名

> 由于浏览器同源策略的关系，一系列客户端攻击将失效，比如上传crossdomain.xml、上传包含Javascript的XSS利用等问题将得到解决。

文件上传实验

课程:文件上传实战靶场Upload-labs
https://www.hetianlab.com/cour.do?w=1&c=CCID53b1-3601-49e1-890e-3db12c192adc

实验:IIS解析漏洞在fckEditor上传攻击中的利用
https://www.hetianlab.com/expc.do?ec=ECID172.19.104.18201510221724380001

实验:Apache解析漏洞
https://www.hetianlab.com/expc.do?ec=ECID172.19.104.18201512011134210001

实验:Nginx解析安全与实战测试
https://www.hetianlab.com/expc.do?ec=ECID218.76.35.762014021910351300001

文件上传实例

> http://120.27.61.239:8003/

Webshell管理工具

中国菜刀

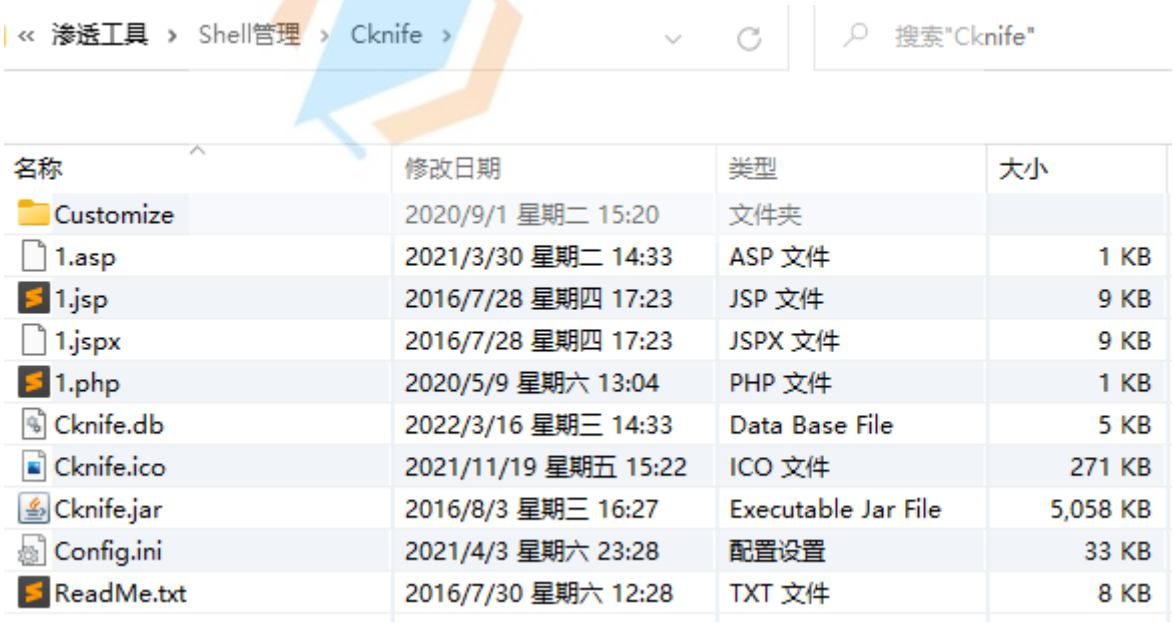
安装

Cknife： （PPT文件夹有编译好的exe程序）
https://github.com/Chora10/Cknife

中国菜刀:
https://github.com/raddyfiy/caidao-official-version

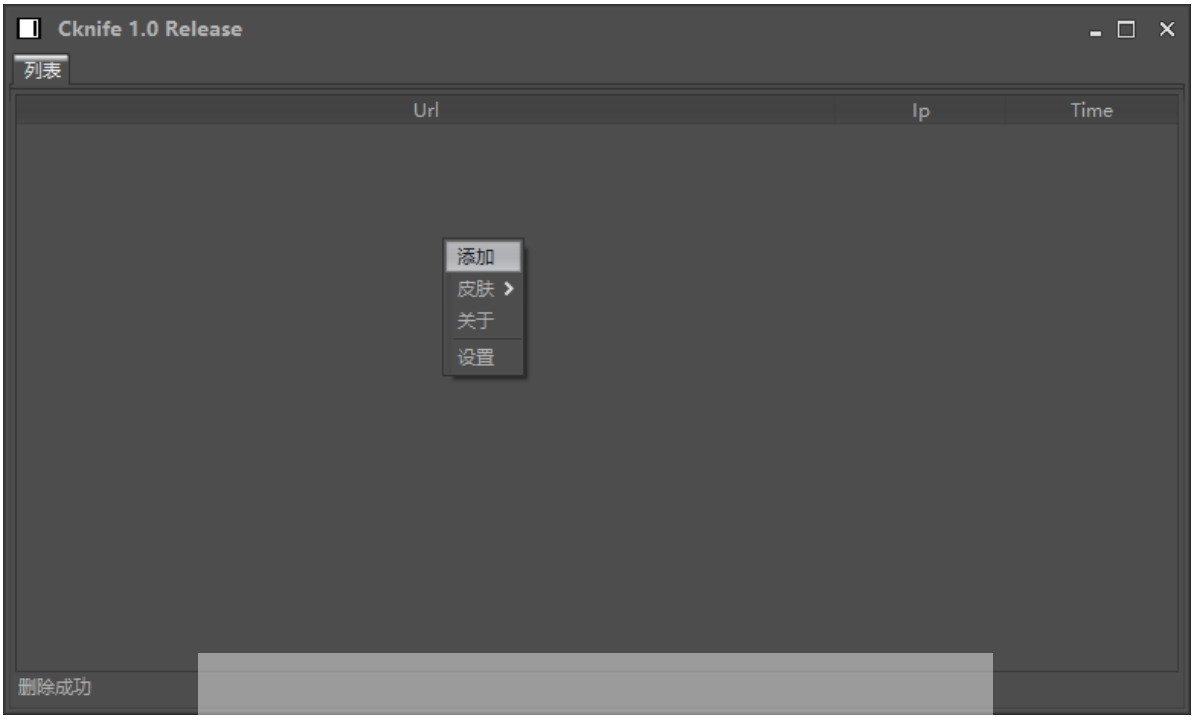
使用

1. 进入Cknife文件夹



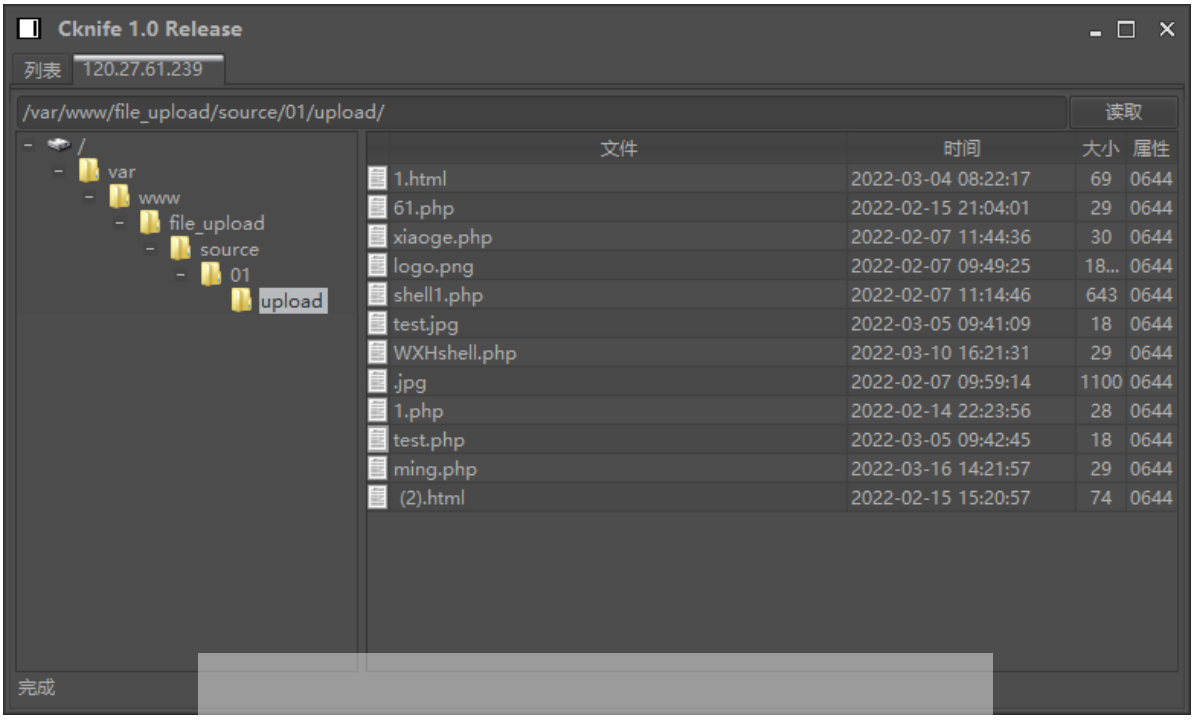
2. 双击运行Cknife.jar（需要安装好Java环境）
3. 添加Shell

右键选择添加

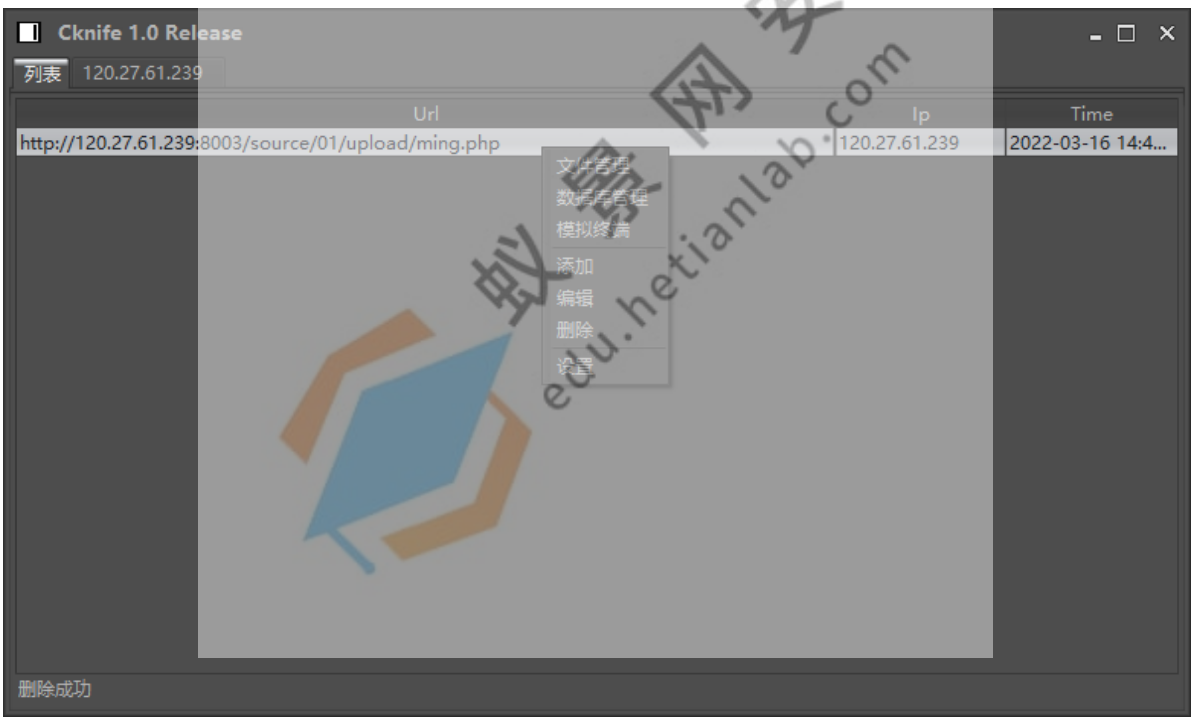


地址栏输入 `webshell` 的 `URL` 地址，以及 `webshell` 密码，然后点击添加

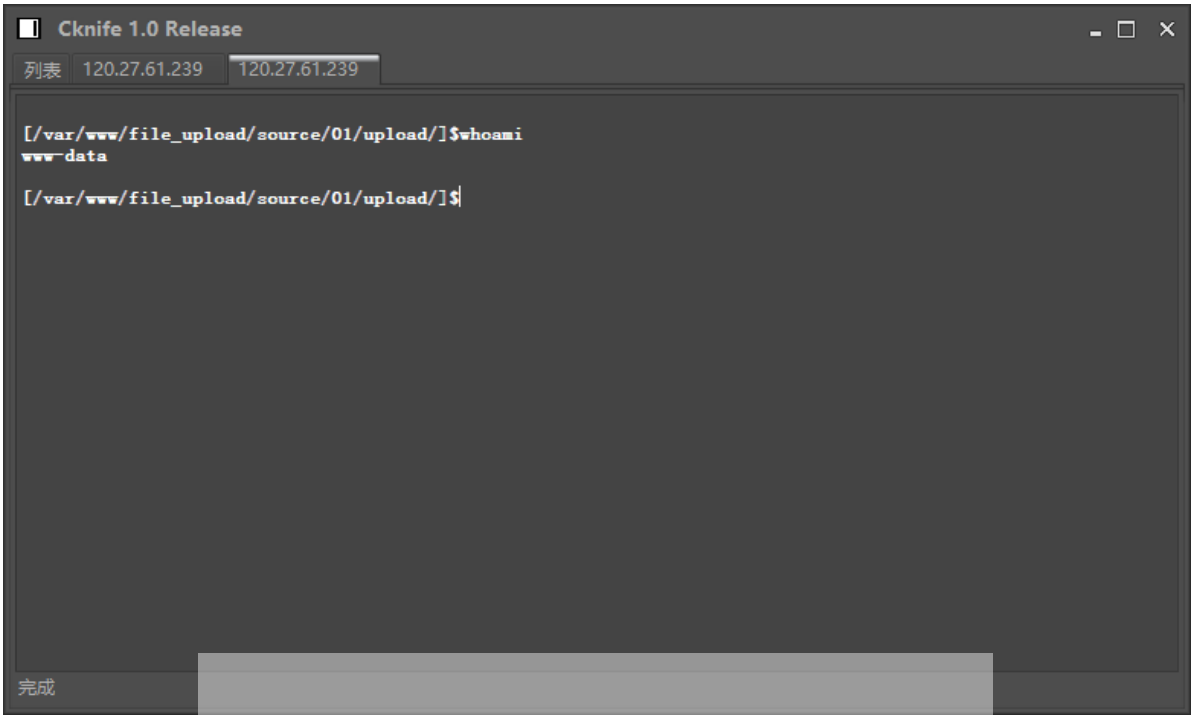
4. 双击打开添加的shell，进入文件管理



5. 右键添加的shell显示功能菜单



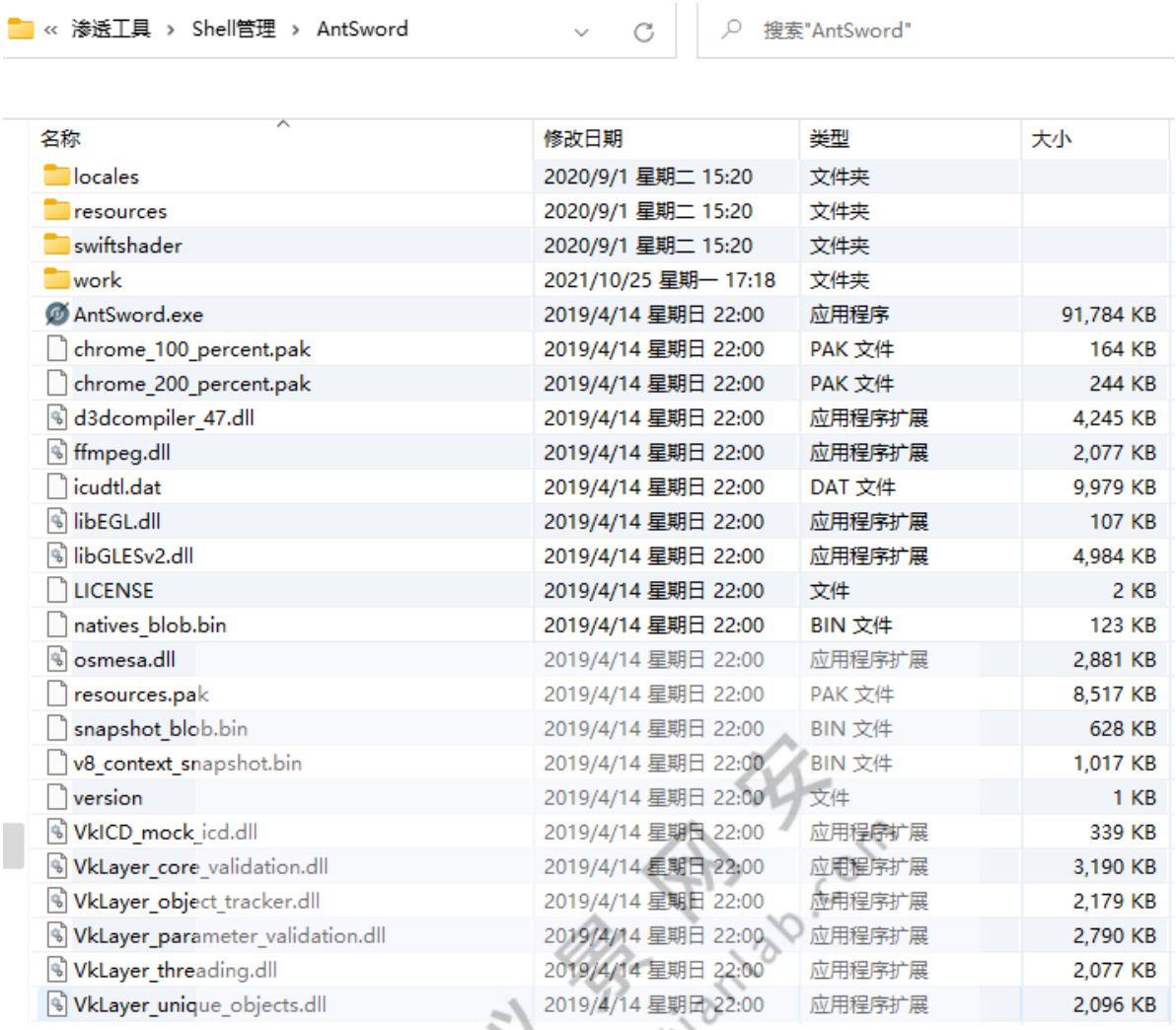
6. 虚拟终端



蚁剑-AntSword

安装

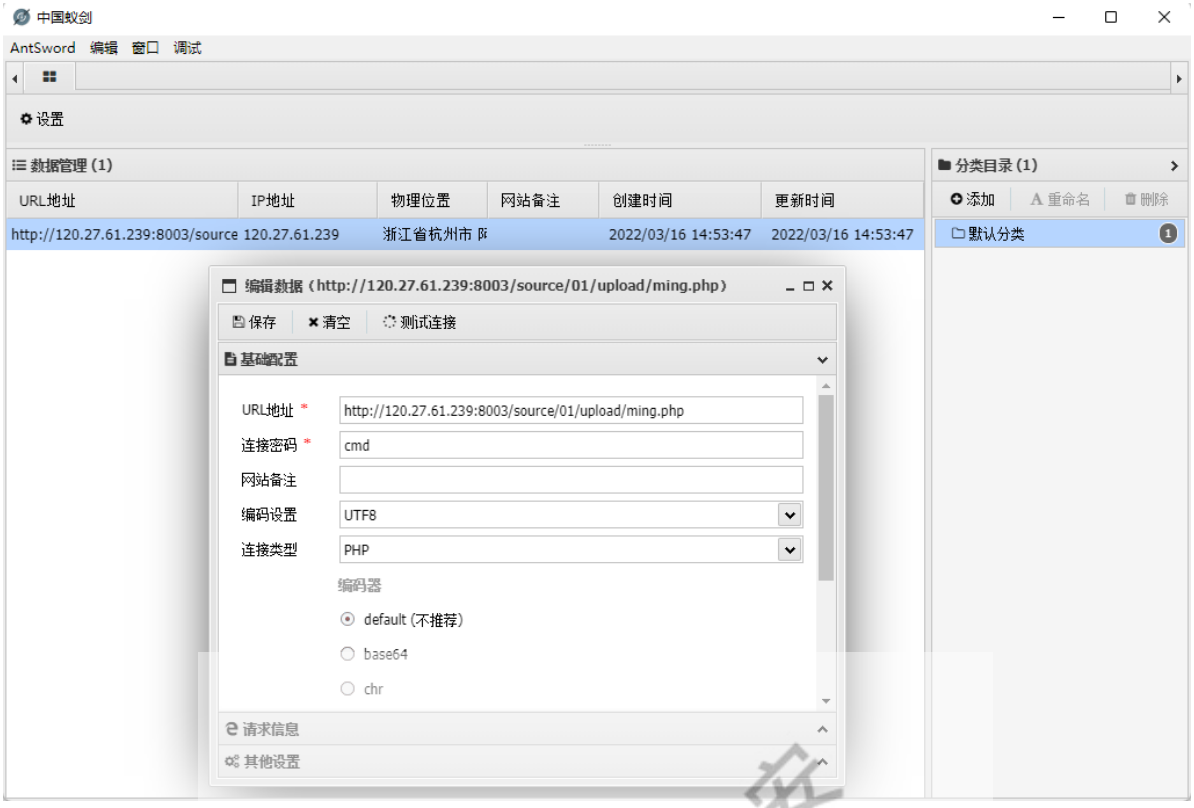
1. 下载 `AntSword-Loader`
   https://github.com/AntSwordProject/antSword-Loader/releases
2. 下载 `antSword`
   https://github.com/AntSwordProject/antSword/releases
3. 解压并进入 `AntSword-Loader` 目录，新建work目录，解压 `antSword` 到 `work` 目录



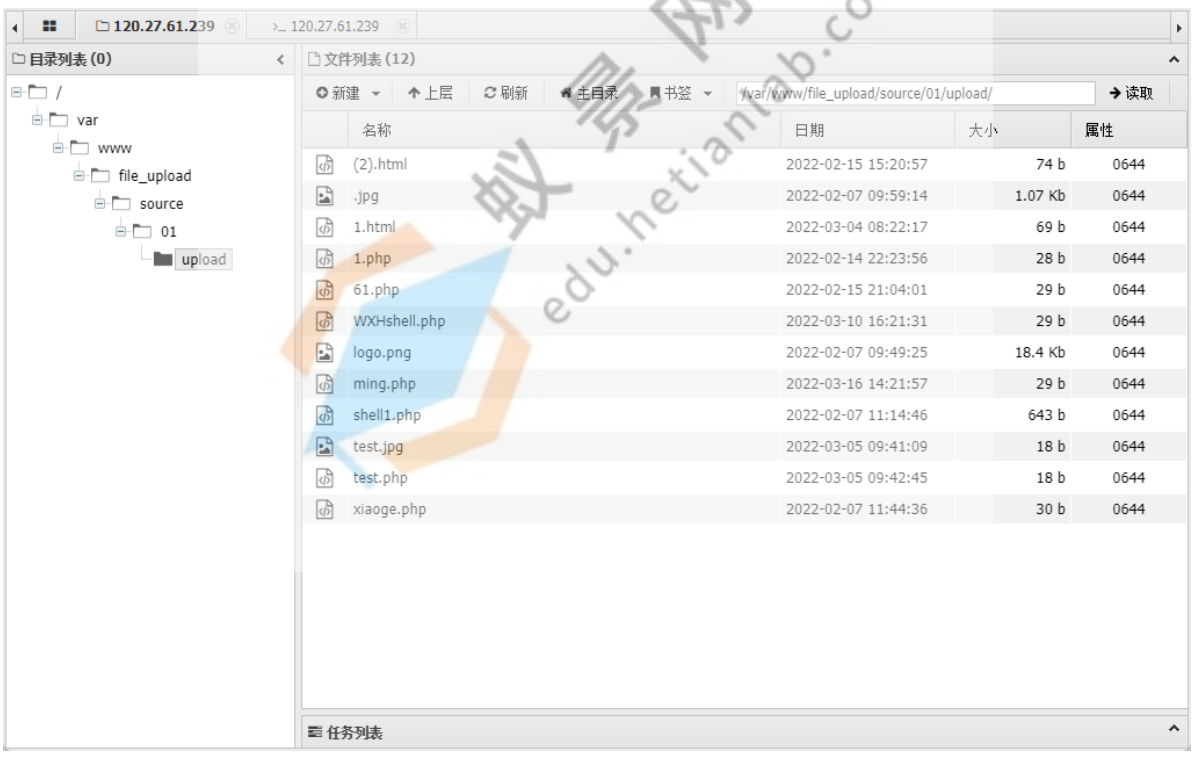
使用

4. 双击打开 `AntSword.exe` ，在界面单击鼠标右键，进入功能菜单

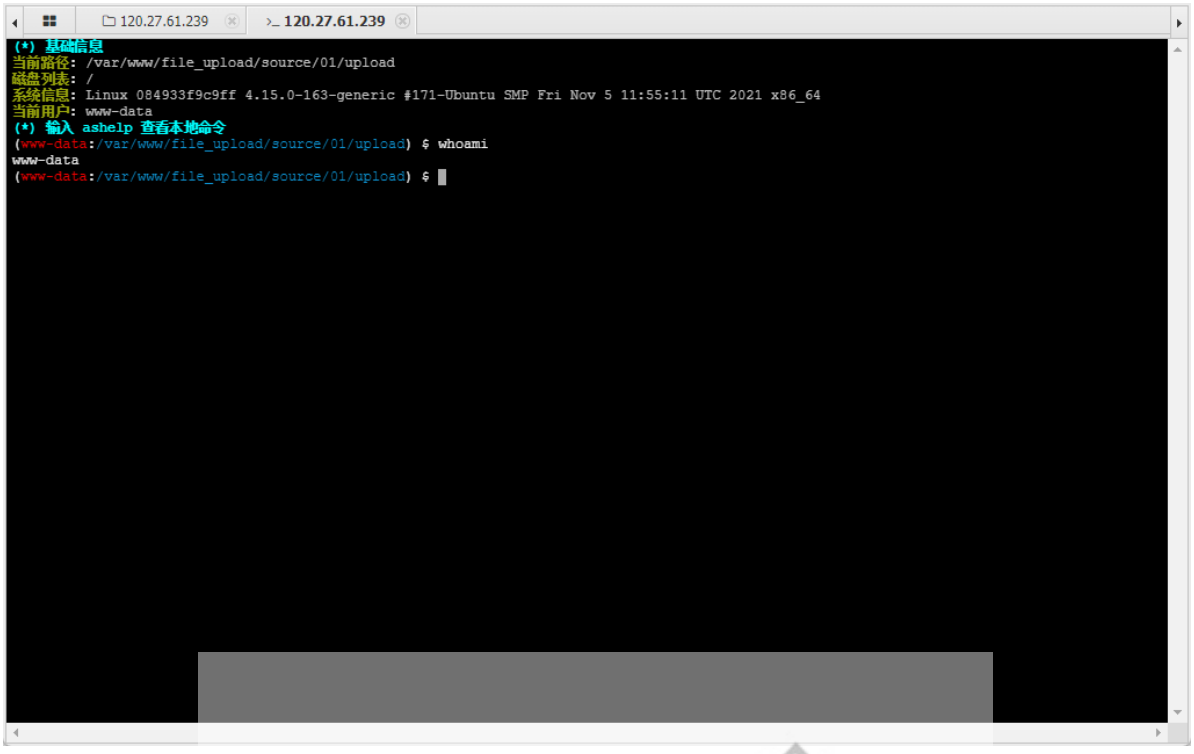
5. 选择添加数据，进入添加shell界面，输入webshell的url地址以及连接密码，点击添加，webshell已经添加成功



6. 右键shell，选择文件管理



7. 右键shell，选择虚拟终端



冰蝎-Behinder

安装

Behinder：
https://github.com/rebeyond/Behinder/releases

使用

哥斯拉-Godzilla

安装

Godzilla：
https://github.com/BeichenDream/Godzilla/releases

使用