Ssrf 引发的血案 - SecPulse.COM | 安全脉搏

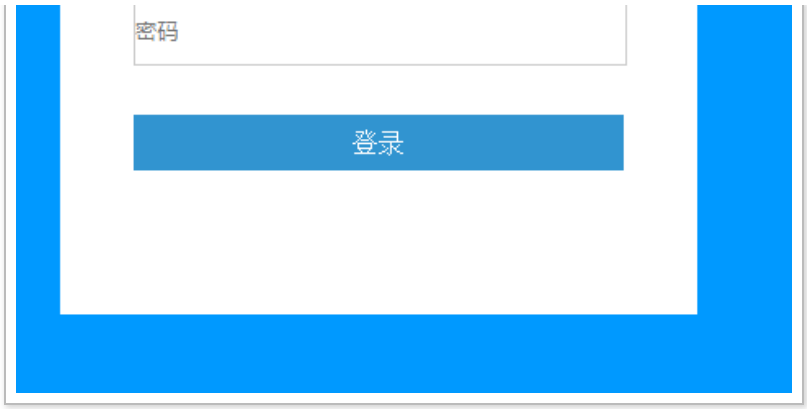
> 这是 酒仙桥六号部队 的第 132 篇文章。

起因

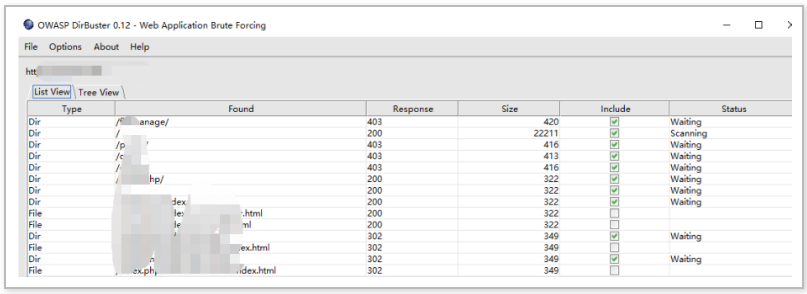
渗透能力的体现不只是储备 0day 的多少，许多站点能否被突破，对本身基础漏洞的熟练的配合利用也是一场考验，故事正是因机缘巧合拿到 shell 的一次记录总结。

从信息搜集到进入后台

客户给定的地址打开之后就只有一个登录页面，留下没有账号的我在风中凌乱。



一直怼一个登录框也不是事儿啊，没办法，只能先将端口，目录和弱口令先探测起来。

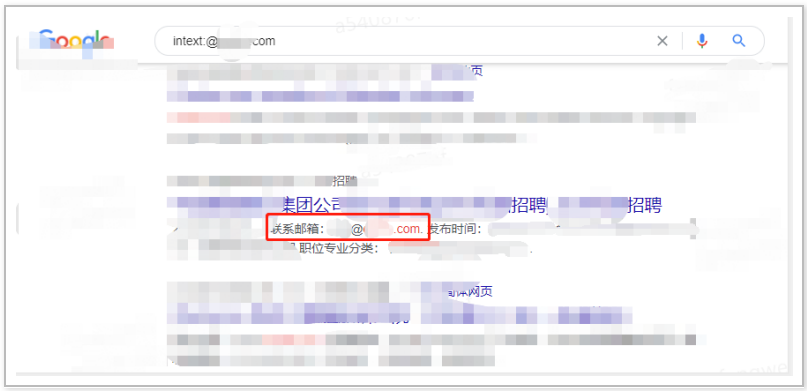


端口基本都开了感觉有点问题，ping 过之后发现有 cdn。

很不幸，弱口令没爆出来，目录端口也没有太多的发现，当务之急就是需要一个账号进入系统。但是账号信息该从哪里搜集？？？

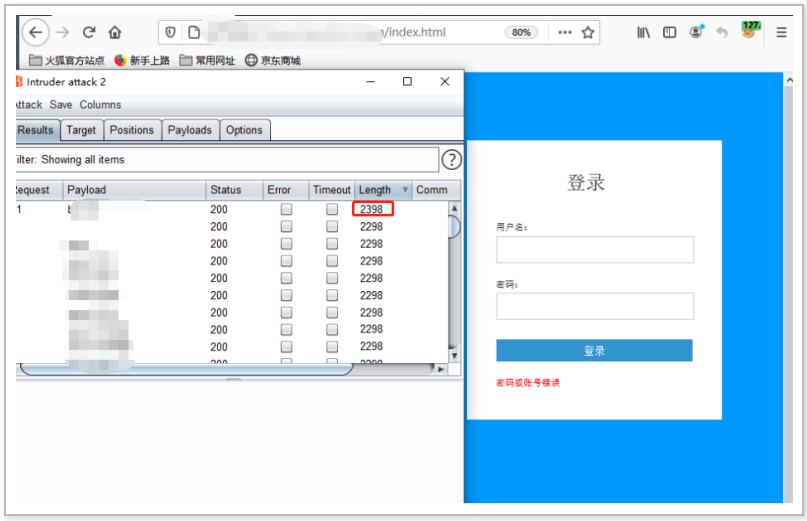
等等，项目开始客户是提供了邮箱地址作为报告的提交地址的，首字母大写 + 名 @xxx 的格式，和许多企业的命名规则一样。

一边先把人名字典构造起来，一边通过 google 语法去搜索相关邮箱，相关公司名称，运气不错，从大大小小几十个网站论坛上面发现七八个公司邮箱，和几个 qq 邮箱。



然后通过一些不可告人的的手段反查到了其中某些 qq 的绑定手机号，以及历史密码信息。

再次构造相关字典，果然人们都喜欢用类似的密码，撞库成功。



进入后台后，挨个测试了一遍功能点都没能发现 getshell 的，上传也没能绕过后缀限制。

都说没有 getshell 的渗透测试是不到位的，只发现一些中低危漏洞可没法满足。

简单的权限认证绕过

因为没有太多的收获，于是挨个访问之前 dirbuster 跑出来的目录，其中一个页面访问之后会有一道黑影一闪而过，然后跳转到登录页面，猜测做了权限验证，然后强制跳转了。

测试中有很多时候都可能遇到无权限访问的情况

当我们遇到访问 403，401，302，或是弹框提示无权限可以尝试一下以下的办法。

1. GET /xxx HTTP/1.1 à403

```
Host: test.com
```

```
绕过:
```

```
GET /xxx HTTP/1.1 à200
```

```
Host: test.com
```

```
X-Original-URL: /xxx
```

2. GET /xxx HTTP/1.1 à403

```
Host: test.com
```

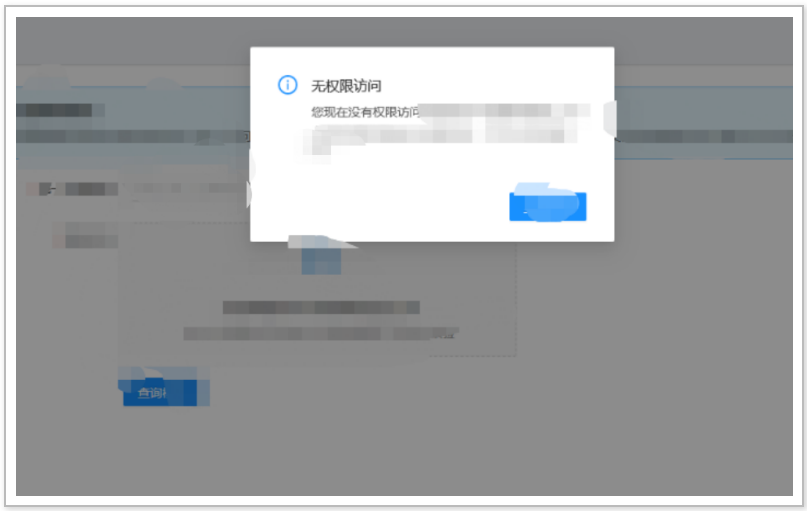
绕过：

```
GET /xxx HTTP/1.1 à200

Host: test.com

Referer: http://test.com/xxx
```

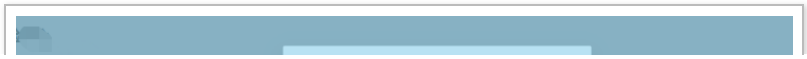
3. 302 跳转：拦截并 drop 跳转的数据包，使其停留在当前页面。

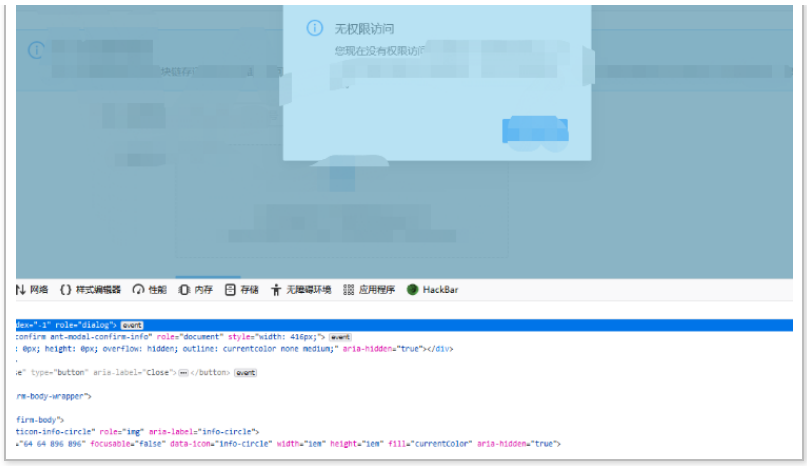
4. 前端验证：只需要删掉对应的遮挡模块，或者是验证模块的前端代码。

这里使用 burp 拦截一下，扔掉后面的跳转，看到如下界面，弹窗还是提示没法访问，权限不够，但是和之前的访问 403 不一样了，难道是我使用了普通用户登录的缘故？？？



熟练的打开 F12 开发者模式。删掉前端代码看是否能使用他的功能。

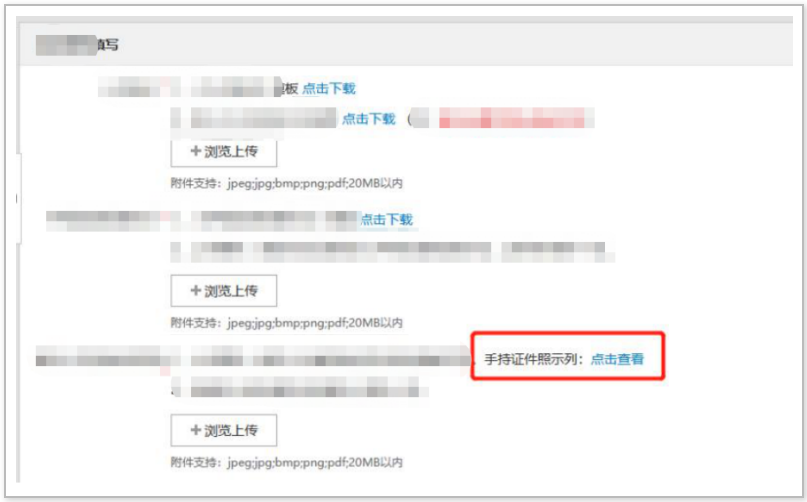




删完权限验证模块的前端代码后，运气不错，还有部分功能可以使用。

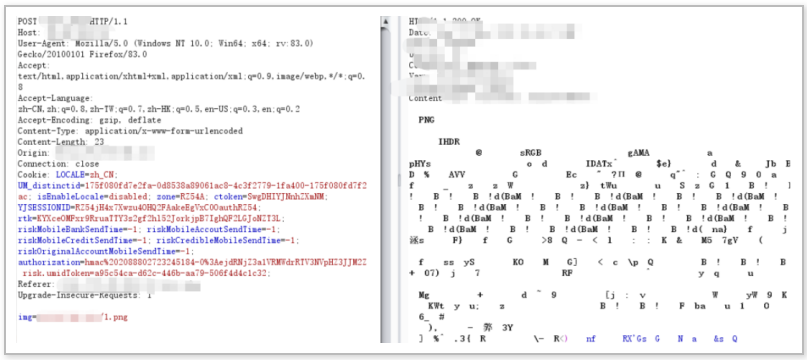
ssrf – 通向 shell 的钥匙

在客户系统后台转了半天，最后在一个查看功能处发现了突破点

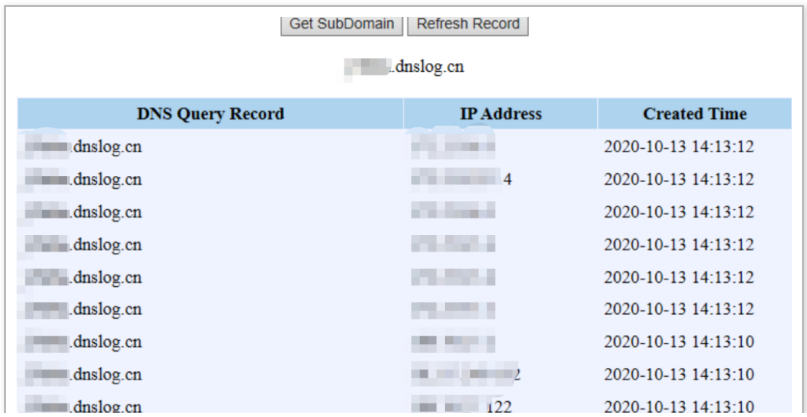


抓包发现 post 参数好像有点意思，尝试换掉默认图片的地址，改为 dnslog 地址，返回提示路径不正确。

猜测是做了后缀的限制，应该只能 post png,jpg 等后缀的地址，先试试读取一下远程服务器上的图片，成功返回，果然有东西。



一个标准的 ssrf，，因为没法改变后缀，应该是不能读取 passwd 之类的文件了，还是先打一波 dnslog，记录一下真实 ip 地址。



但是 ssrf 可不只是读个文件那么简单，ssrf 通常可以用来打内网应用，通过它来打个 redis 或者 mysql 岂不美哉。

先借助 ssrf 探测一下开放的端口，22，80，443，6379。

Request	Payload	Status	Error	Timeout	Length	Comment
446	6379	200			398	
11	443	200			398	
12	80	200			398	
4	22	200			398	
5	5	200			298	
6	6	200			298	

看看攻击 redis 一般可以利用的 dict 和 gopher 两种协议，使用 gopher 协议的话需要注意一些利用限制。

协议	PHP	Java	Curl	Perl	ASP.NET
gopher	--with-curlwrappers且php版本至少为5.3	小于JDK1.7	低版本不支持	支持	小于版本3

gopher 协议规则比较复杂，经过查找，找到了一款工具，使用其生成的 payload 很准确，且可自定义。

需要的小伙伴可以自取。

https://github.com/firebroo/sec_tools

需要将内容再进行一次 url 编码传到 web 的参数中才会正常运行。

Dict 协议敲命令较为直接。

1. 写入内容；

dict://127.0.0.1:6379/set❌test

2. 设置保存路径；

dict://127.0.0.1:6379/config:set:dir:/tmp/

3. 设置保存文件名；

dict://127.0.0.1:6379/config:set:dbfilename:1.png

4. 保存。

dict://127.0.0.1:6379/save

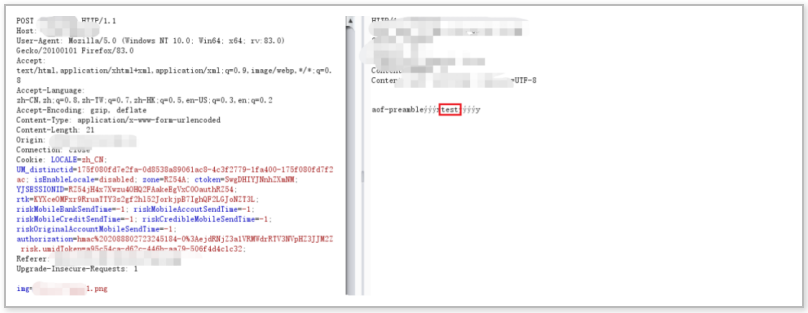
我们一般对 redis 常见的攻击方式有：

1. 写 webshell；
2. 写密钥；
3. 定时任务反弹。

第一种需要 web 路径，后两种方法可能需要一定的权限。

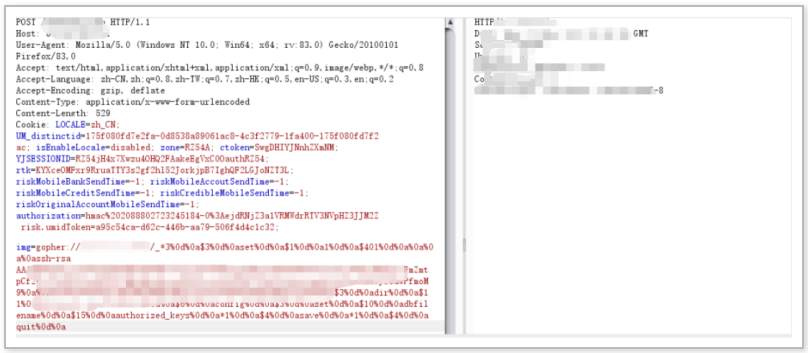
攻击的思路有了，但是我们通过 dict 协议访问后并没有出现回显，不知道是否存在未授权的 redis 服务，盲打一顿可能浪费宝贵的时间，灵光乍现，可以先写一个图片文件到 tmp 目录里，再通过 file 协议进行读取，出现内容就表明 redis 是能够利用的。

出现回显，说明文件成功写入了，虽然有乱码，但是影响不大。



为了拿到 shell，当然是先试试用 gopher 协议写密钥，本机生成密钥: ssh-keygen -t rsa。再使用工具将以下命令转换成 gopher 协议支持的形式。

```
config set dir /root/.ssh

config set dbfilename authorized_keys

set test "xxx"

save
```



写入后尝试连接一下页面啥也没返回，尝试连接一下 Wfk，突然想起 nmap 结果好像 ssh 没对外开放，决策性失误。

尝试反弹计划任务，但是等了半天也没见 shell 弹回来，猜测可能是权限不够，没能够成功写入，可惜前期测试中并没有发现信息泄露暴露出 web 目录的路径，不然能写个 webshell 也是极好的。

没办法，这个只能先搁置一边，条条大路同罗马，既然这个域名不行，看看有没有绑定的其他域名在这个 ip 上。

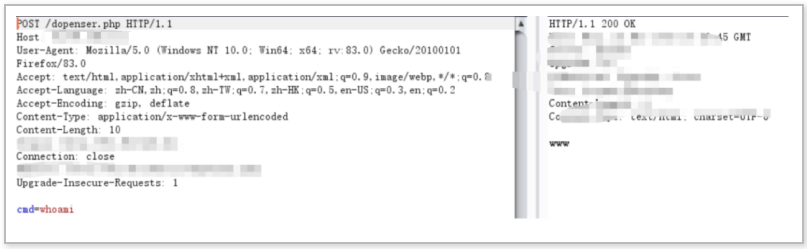
旁站信息泄露 getshell

通过之前记录的 dnslog 上的 ip 地址进行反查，发现了该 ip 地址下绑定了其他域名。

访问后改掉 url 变量后的默认参数，触发报错，成功爆出了绝对路径，小小的报错，却提供了巨大的价值。

因为是旁站，现在获取到了 B 站的网站路径，如果能通过 A 站的 ssrf 把 webshell 写到 B 站的 web 路径里也是美滋滋了，说干就干。

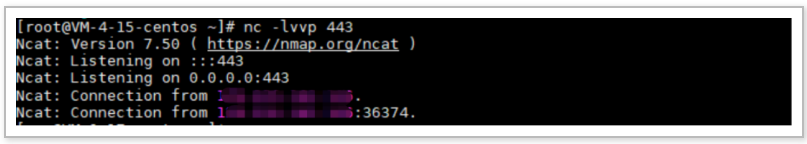
访问 shell，并敲入 whoami 命令查看权限，发现是个低权 www 用户。



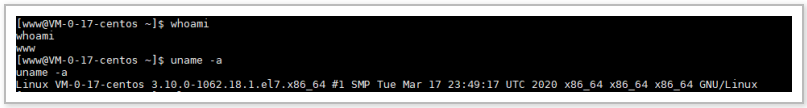
提权

弹个交互的 shell 出来方便进行提权，但是远程服务器一直没法正常收到 shell。

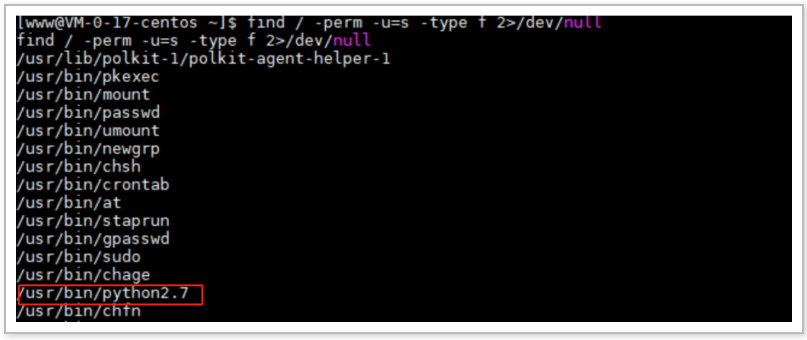
切换一下端口，网络管理员可能做了一定的限制，尝试通过 443，53 等经常开放的端口弹出 shell。



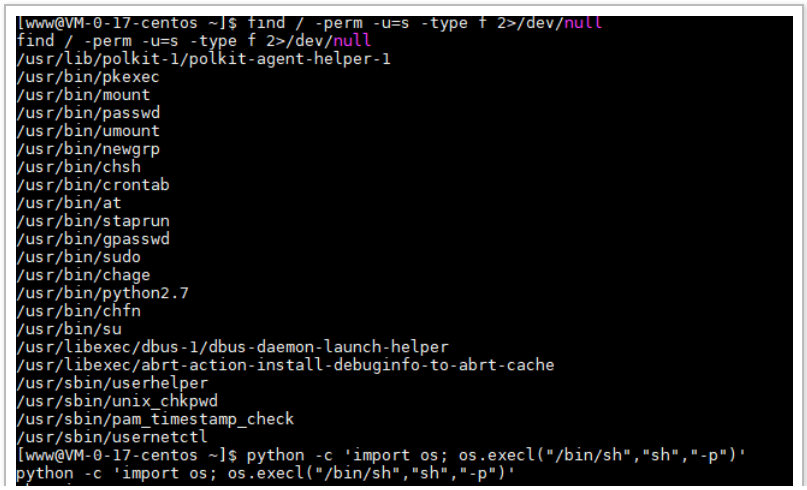
成功拿到 shell，获取到低权限 SHELL 后一般会看看内核版本，检测当前用户权限，再列举 Suid 文件，如果都没发现可能会借助一些自动化脚本来检查可能存在的提权方式。



通过 find / –perm –u=s –type f 2>/dev/null 看看有 s 属性文件。



Python 好像是可以通过 suid 提权的，翻了翻自己的小笔记，payload 一发入魂。



```
whoami
root
```

这里附上 centos 下 suid 提权较为全面的总结：

https://www.freebuf.com/articles/system/244627.html

至此测试结束。

总结

整个测试过程遇到很多困难，许多地方看似简单，其实是反复尝试之后才顺利过关。

测试中其实并未使用多么牛逼的攻击手段，简单梳理整个流程：全网信息搜集发现用户账户 à 撞库拿到部分密码 à 前端验证绕过发现新功能点 àssrf 探测信息 à 旁站获取 web 绝对路径跨网站写入 shellà 拿到 shell 后通过 suid 提权

sudo 提权。

全文完