THM-Active Reconnaissance(主动侦察基础)-学习

本文相关的TryHackMe实验房间链接： https://tryhackme.com/room/activerecon

H2 介绍

在本文中，我们专注于主动侦察和与之相关的基本工具：学习使用网络浏览器来收集有关目标的更多信息；学习使用一些简单的工具（例如 ping、traceroute、telnet 和 nc）来收集有关网络、系统和服务的信息。

主动侦察从直接连接到目标机器开始，任何此类连接都可能在目标的日志记录中留下客户端 IP 地址、连接时间和连接持续时间等信息；然而，并不是所有的连接都是可疑的，你可以尝试让你的主动侦察活动表现地和常规客户活动类似。

考虑进行网页浏览操作，因为当你的请求夹杂在数百名合法用户中，没有人会怀疑连接到目标 Web 服务器的浏览器。 当你作为红队（攻击者）的一员并且不想惊动蓝队（防守者）时，你可以利用此类隐蔽技术来发挥自己的优势。

在本文中，我们将了解：通常与大多数操作系统捆绑在一起的工具或很容易获得的工具。

我们从 Web 浏览器及其内置的开发工具开始；此外，我们将展示如何将 Web 浏览器“武装”成高效的侦察框架；之后，我们将讨论其他良性工具，例如 ping、traceroute 和 telnet。 所有这些程序都需要与目标连接，因此我们的活动属于主动侦察活动，也能够被目标系统监测到。

H2 通过web浏览器进行主动信息收集

Web 浏览器可以是一个很方便的工具，尤其是它在所有系统上都很容易使用，你可以通过多种方式使用 Web 浏览器来收集有关目标的信息。

在传输层中，Web浏览器使用的端口为:

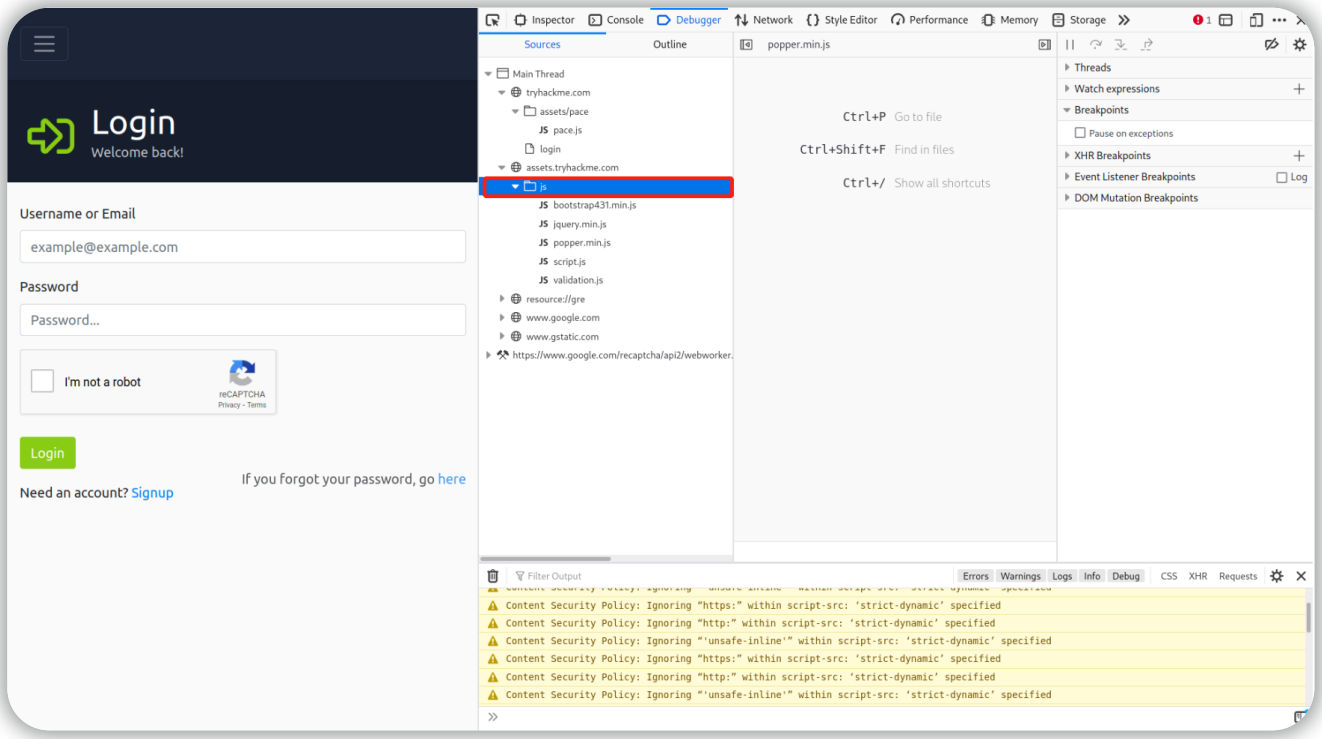
- 通过 HTTP 访问网站时默认使用 TCP 端口 80
- 通过 HTTPS 访问网站时默认使用 TCP 端口 443

由于 80 和 443 是 HTTP 和 HTTPS 的默认端口，因此 Web 浏览器并不会在地址栏中显示它们。但是，也可以使用自定义端口来访问web服务，例如： https://127.0.0.1:8834/ 将通过 HTTPS 协议使用端口8834连接到 127.0.0.1 (localhost)， 如果有 HTTPS 服务器在该端口上侦听，我们将收到一个网页画面。

在使用浏览器浏览网页时，你可以在 PC 上按 Ctrl+Shift+I 或在 Mac 上按 Option + Command + I 来打开 Firefox 上的开发者工具，类似的快捷方式也可以使用在 Google Chrome浏览器 或 Chromium浏览器中。

开发人员工具允许你检查浏览器已经接收并与远程服务器发生交换的许多内容。例如，你可以查看甚至修改 JavaScript (JS) 文件，检查系统上设置的 cookie 并查看有关站点内容的文件夹结构。

下面是Firefox浏览器中的开发者工具的截图，Chrome中的DevTools（开发者工具）和下图也非常相似。



Firefox 和 Chrome 也有很多插件可以帮助渗透测试，下面是一些例子:

- **FoxyProxy**--可让你快速更改用于访问目标网站的代理服务器。 当你使用 Burp Suite 等工具或者需要定期切换代理服务器时，此浏览器扩展程序很方便。

  获取 FoxyProxy（安装在Firefox 浏览器中使用）： https://addons.mozilla.org/en-US/firefox/addon/foxyproxy-standard/

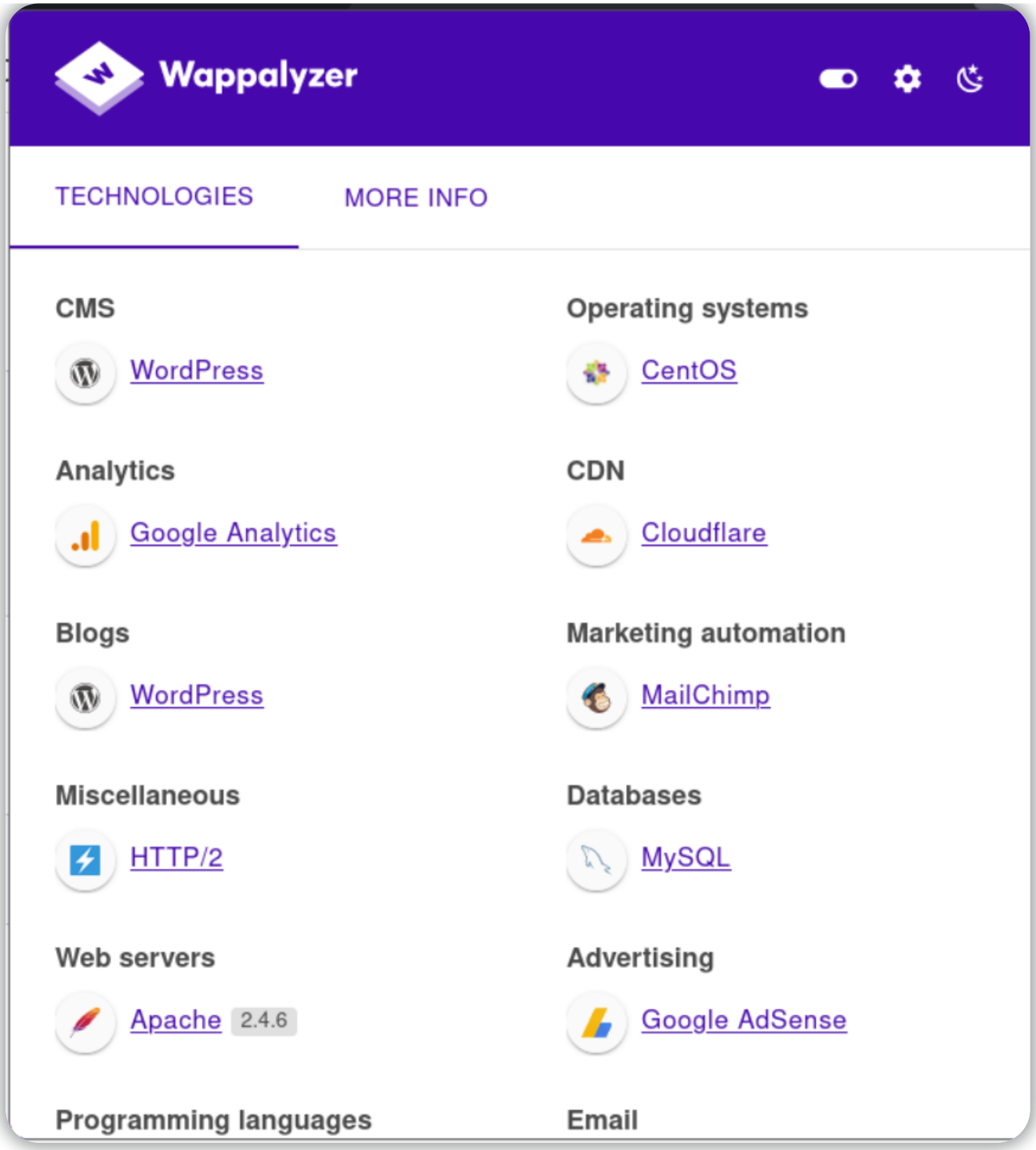
- **User-Agent Switcher and Manager**--用户代理切换器和管理器，使你能够假装正在以不同的操作系统或不同的 Web 浏览器访问网页。换句话说，你可以假装在使用 iPhone 浏览网站，而实际上你是使用 Mozilla Firefox浏览器访问网站。

  下载 Firefox 的用户代理切换器和管理器： https://addons.mozilla.org/en-US/firefox/addon/user-agent-string-switcher/

- **Wappalyzer**--提供对正在访问的网站所使用的技术的分析结果。 这个扩展程序很方便，当你在像任何其他用户一样浏览网站时就能收集到网站相关信息。

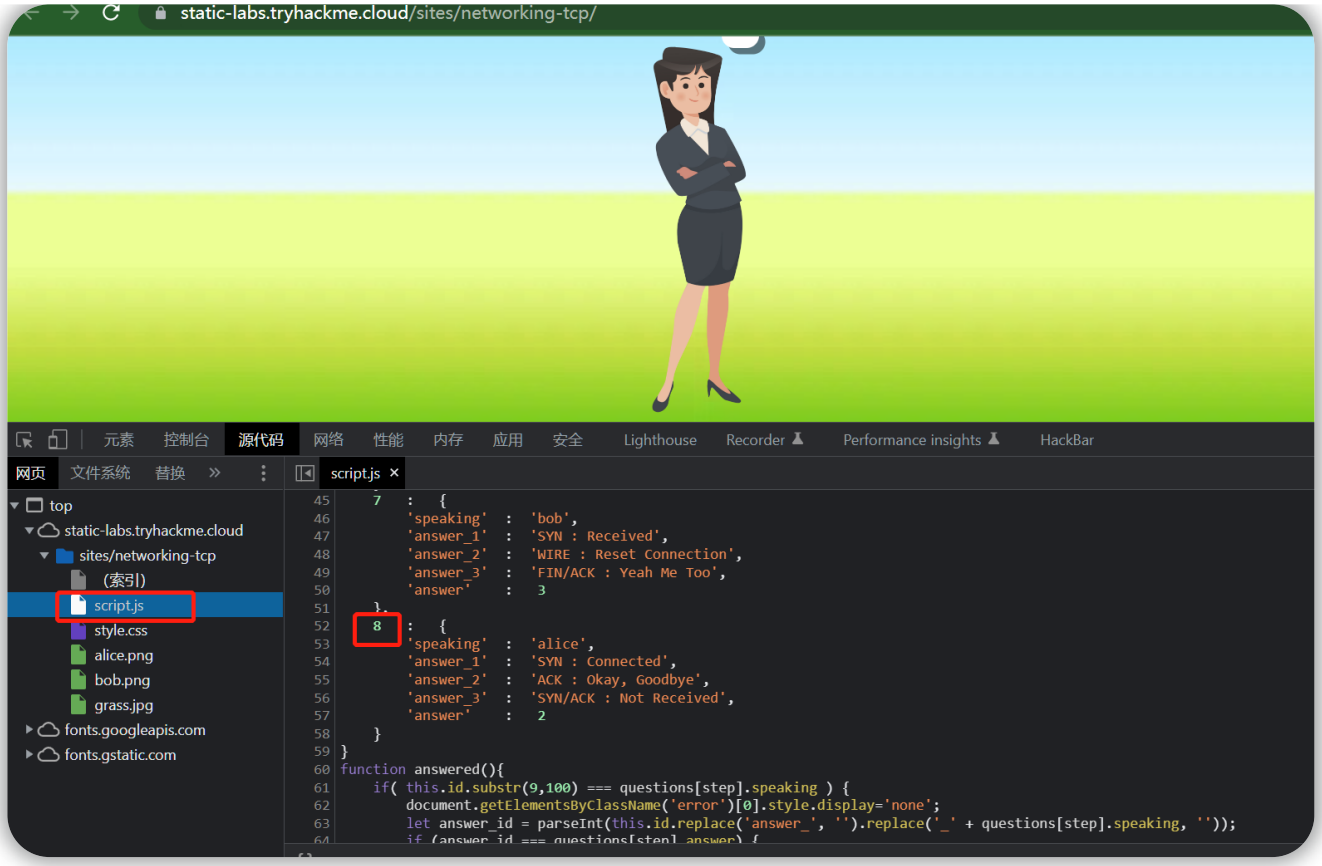
  下载适用于 Firefox 的 Wappalyzer： https://addons.mozilla.org/en-US/firefox/addon/wappalyzer/

  Wappalyzer的效果截图：


Wappalyzer
TECHNOLOGIES
MORE INFO
CMS
WordPress
Operating systems
CentOS
Analytics
Google Analytics
CDN
Cloudflare
Blogs
WordPress
Marketing automation
MailChimp
Miscellaneous
HTTP/2
Databases
MySQL
Web servers
Apache 2.4.6
Advertising
Google AdSense
Programming languages
Email

答题


回答以下问题
浏览以下网站并确保您已在 AttackBox Firefox 或计算机上的浏览器上打开开发者工具。使用开发者工具，计算出问题的总数。
8
正确答案
暗示



H2 使用Ping命令进行主动信息收集

Ping 应该能让你想起乒乓球--你扔出球并期望把它拿回来。 ping 的主要目的是检查你是否可以访问远程系统以及远程系统是否可以返回响应到你的机器。 换句话说，最初ping是用来检查网络连接性的； 然而，我们更感兴趣的是它的另一用途：检查远程系统是否在线。

简单来说，ping 命令会向远程系统发送数据包，远程系统则会进行回复。 这样，你就可以断定远程系统是否在线并且网络在两个系统之间（本地和远程）能否正常工作。

如果你更喜欢挑剔一点的定义，那么 ping 是将 ICMP Echo 数据包发送到远程系统的命令。如果远程系统在线，ping 数据包被正确路由并且没有被任何防火墙阻止，则远程系统应该会发回一个 ICMP Echo 响应。同样，如果经过正确路由并且没有被任何防火墙阻止，ping 响应 应该会到达之前发出ping命令的系统。

此类命令的目的是在我们花时间执行更详细的扫描以发现正在运行的操作系统和服务之前 确保目标系统在线（存活）。

你可以使用 `ping MACHINE_IP` 或 `ping HOSTNAME` 对目标机进行检测。在第二条命令中，系统需要在发送 ping 数据包之前将 HOSTNAME 解析为 IP 地址。如果你没有在 Linux 系统上指定计数，则需要按 CTRL+C 来强制ping命令停止。

在linux系统上，如果你只想发送十个数据包，你可以使用 `ping -c 10 MACHINE_IP` ，这相当于在 MS Windows 系统上执行 `ping -n 10 MACHINE_IP` 。

从技术上讲，ping 属于 ICMP（Internet Control Message Protocol）协议。 ICMP 支持多种类型的查询，这里我们主要介绍的是ping（ICMP echo/type 8）和 ping reply（ICMP echo reply/type 0），此处只是简单使用ping命令所以并不需要了解 ICMP协议的详细信息。

在下面的示例中，我们将数据包的总数指定为 5，并且使用 ping MACHINE_IP。我们了解到 MACHINE_IP 对应的目标系统在线并且没有阻塞ICMP 回显请求，此外，数据包路由上的任何防火墙和路由器也没有阻塞 ICMP 回显请求。

```
user@AttackBox$ ping -c 5 10.10.58.15
PING 10.10.58.15 (MACHINE_IP) 56(84) bytes of data.
64 bytes from MACHINE_IP: icmp_seq=1 ttl=64 time=0.636 ms
64 bytes from MACHINE_IP: icmp_seq=2 ttl=64 time=0.483 ms
64 bytes from MACHINE_IP: icmp_seq=3 ttl=64 time=0.396 ms
64 bytes from MACHINE_IP: icmp_seq=4 ttl=64 time=0.416 ms
64 bytes from MACHINE_IP: icmp_seq=5 ttl=64 time=0.445 ms

--- MACHINE_IP ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4097ms
rtt min/avg/max/mdev = 0.396/0.475/0.636/0.086 ms
```

在上面的示例中，我们清楚地看到目标系统正在响应ping命令。 ping命令的输出结果表明目标在线且可被访问，我们已经发送了五个数据包，也收到了五个回复，我们还注意到，回复到达我们的本地系统平均需要0.475 毫秒，最大值为 0.636 毫秒。

从渗透测试的角度来看，我们将尝试发现这个目标的更多信息，例如：目标的哪些端口是开放的，目标有哪些服务正在运行。

让我们考虑以下情况：关闭目标虚拟机，然后再尝试ping命令。正如你在以下示例中看到的结果，我们没有收到任何来自目标机器的有效回复。

```
user@AttackBox$ ping -c 5 10.10.58.15
PING 10.10.58.15 (MACHINE_IP) 56(84) bytes of data.
From ATTACKBOX_IP icmp_seq=1 Destination Host Unreachable
From ATTACKBOX_IP icmp_seq=2 Destination Host Unreachable
From ATTACKBOX_IP icmp_seq=3 Destination Host Unreachable
From ATTACKBOX_IP icmp_seq=4 Destination Host Unreachable
From ATTACKBOX_IP icmp_seq=5 Destination Host Unreachable

--- MACHINE_IP ping statistics ---
5 packets transmitted, 0 received, +5 errors, 100% packet loss, time 4098ms
pipe 4
```

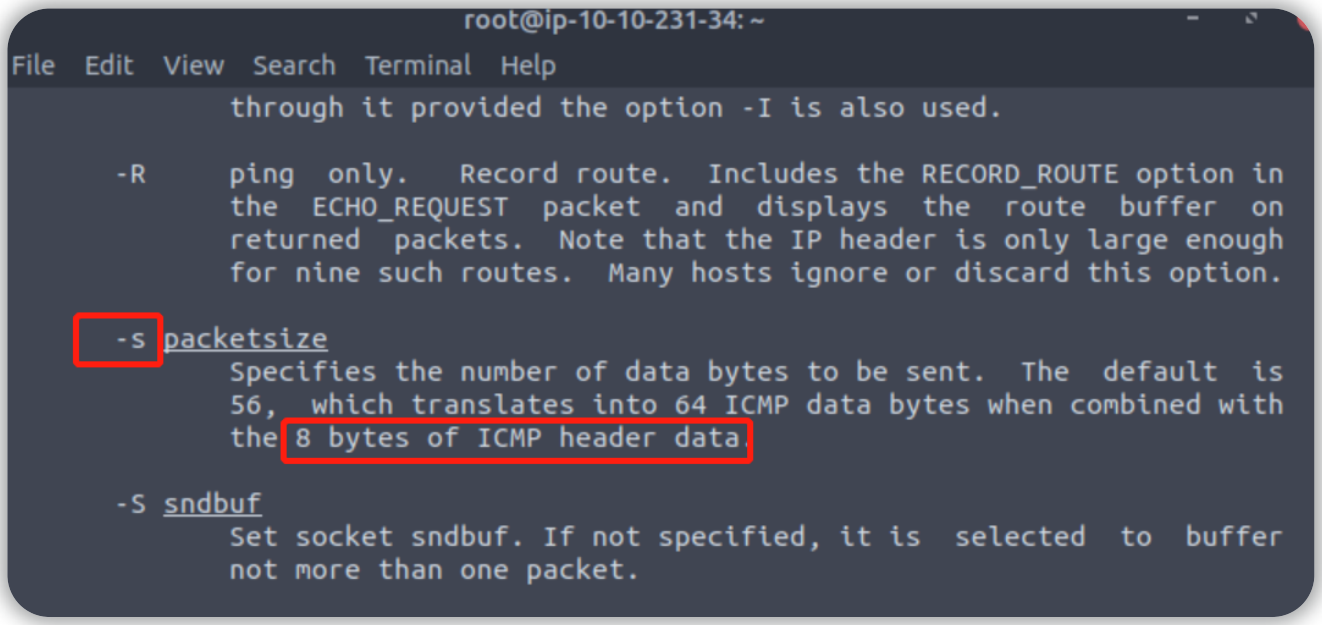
在上面示例中，我们已经关闭了ip地址为10.10.58.15 的目标机器。 对于每个 ping，我们使用的本地系统都以“无法访问目标主机”进行响应，可以看到我们已经发送了五个数据包，但都没有收到回复，这将导致 100% 的数据包丢失。

一般来说，当我们没有收到 ping 回复时，有一些解释可以说明为什么我们没有收到 ping 回复，例如：

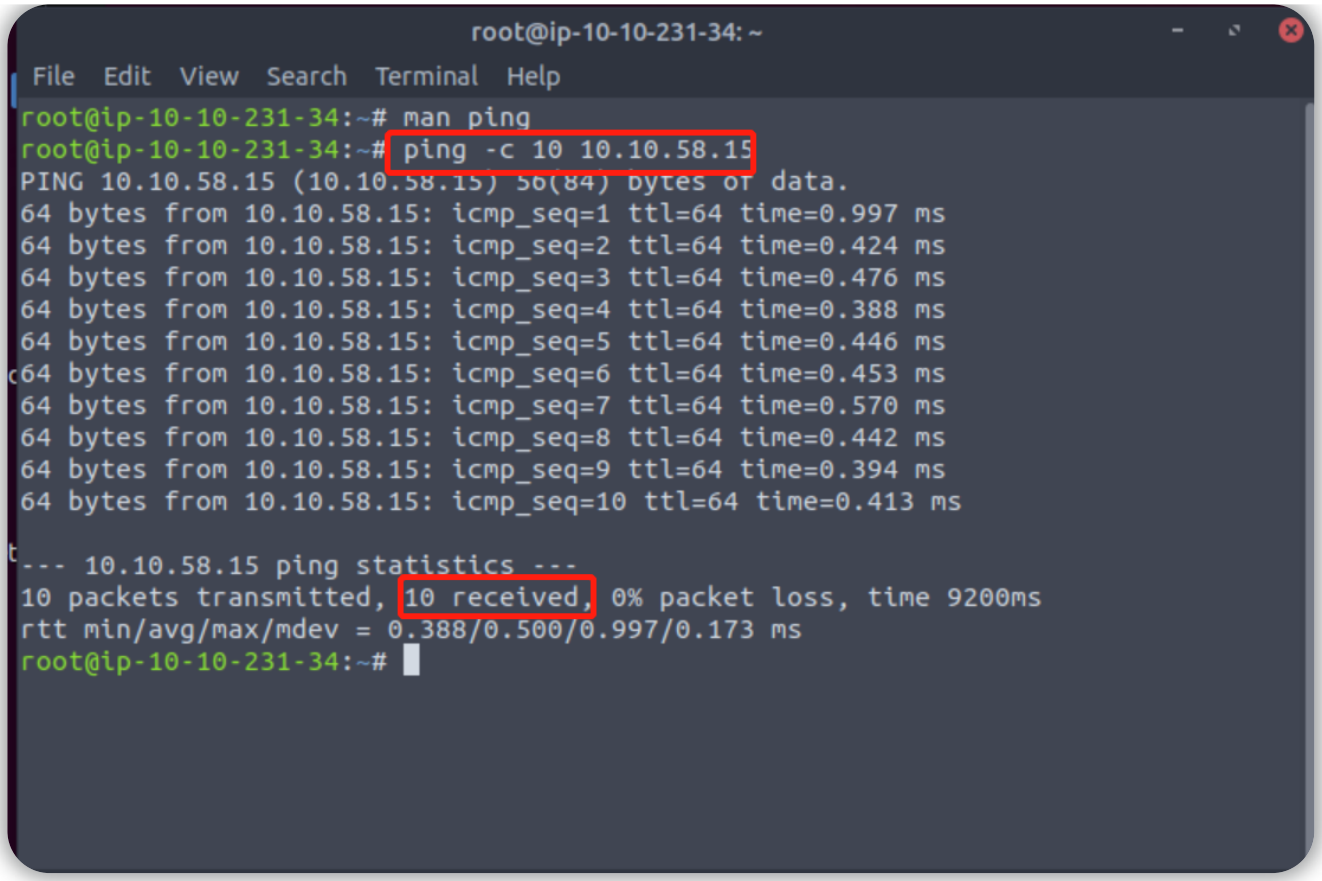
- 目标机没有响应，可能仍在启动或关闭状态，或者目标操作系统已崩溃。
- 目标机已从网络中拔出，或者数据包的传输路径中存在网络设备故障。
- 防火墙被配置为阻止此类数据包。 防火墙可能是在系统本身上运行的一个软件，也可能是一个单独的网络设备，请注意，默认情况下MS Windows 防火墙会阻止 ping命令的数据包。
- 你的本地系统已从网络中拔出。

答题

使用 `man ping` 命令，查看ping的用法：



使用ping命令：



H2 使用Traceroute进行信息收集

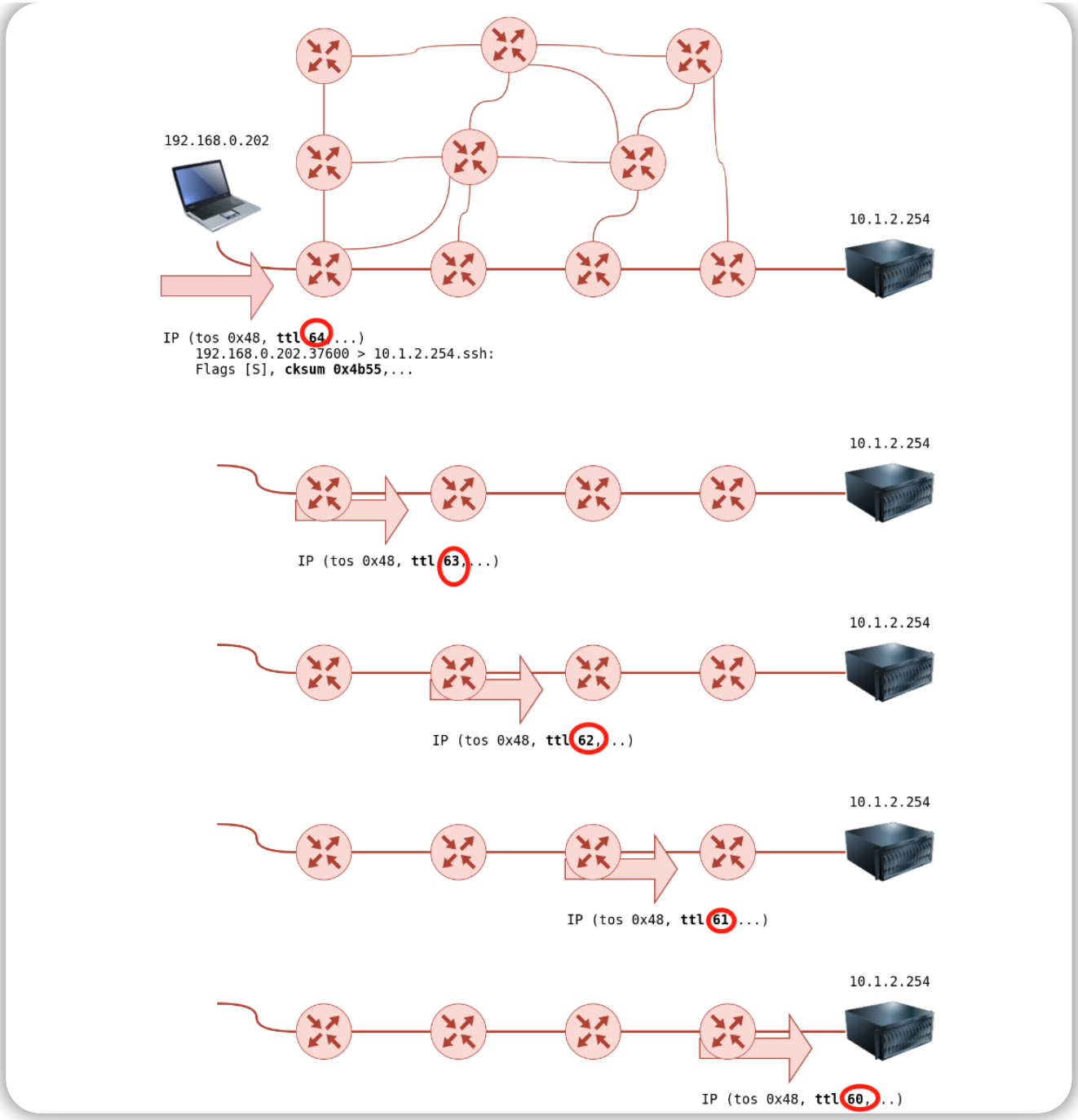
顾名思义，traceroute(跟踪路由) 命令会跟踪数据包从你的本地系统传输到另一台主机所采用的路由路径。 如果kali机未安装 traceroute，可以使用 apt install traceroute 进行安装。

traceroute的目的是查找数据包从你的系统到目标主机时所经过的路由器或跃点的 IP 地址，此命令还会显示两个系统之间的路由器数量，这些信息很有帮助，因为它指示了本地系统和目标主机系统之间的跃点数（路由器）。 但是，请注意，由于许多路由器使用的是适应网络变化的动态路由协议，所以数据包采用的路由路径可能会发生变化。

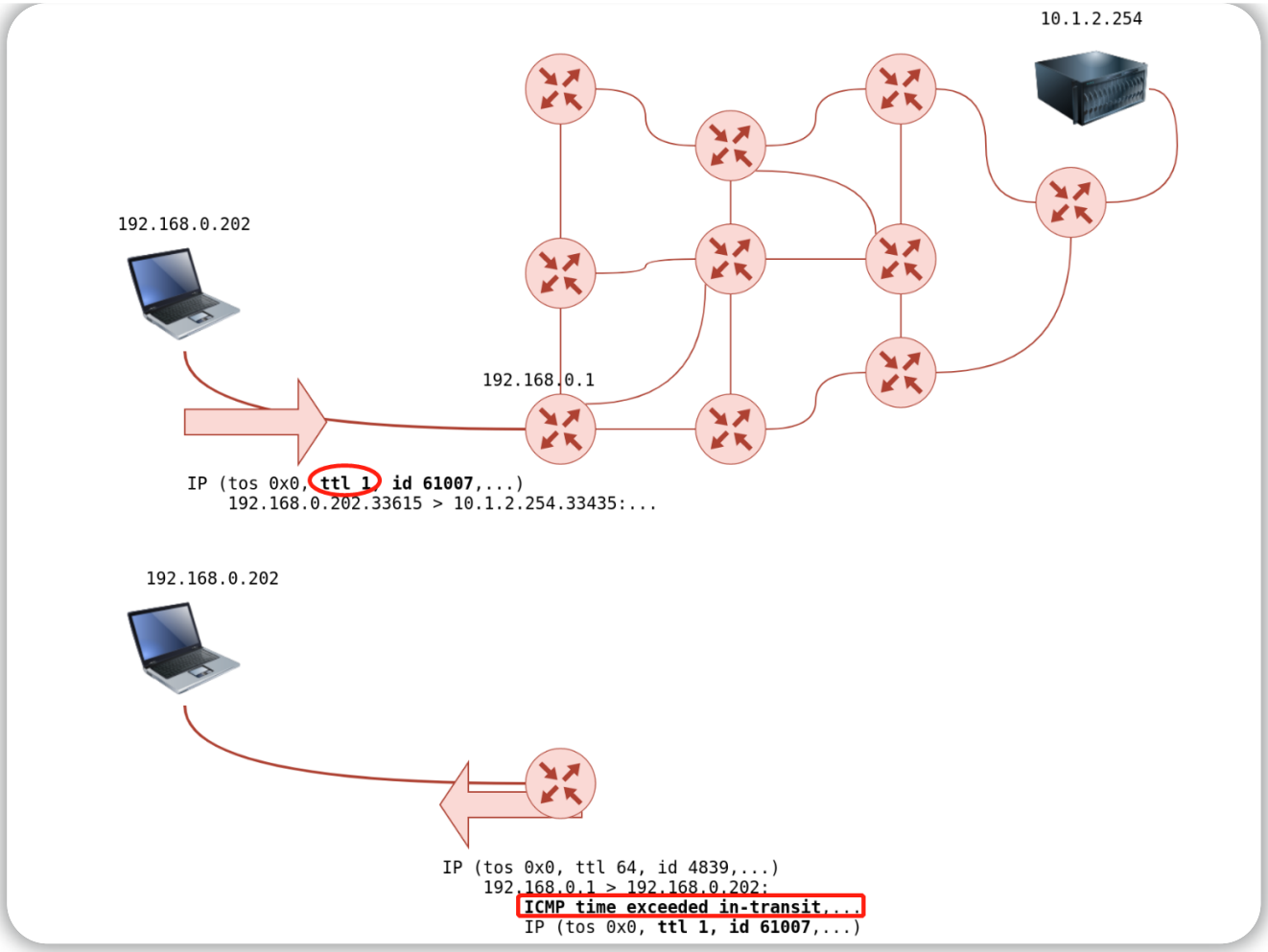
在 Linux 和 macOS 上，使用的命令为traceroute MACHINE_IP；在 MS Windows 上，使用的命令为 tracert MACHINE_IP 。traceroute 会尝试通过 从你的本地系统到目标系统的路径 发现路由器。

没有直接的方法可以发现从你的系统到目标系统的路径，但是我们可以依靠 ICMP 来“诱骗”路由器泄露其 IP 地址，我们可以通过在 IP 标头字段中使用一个小的生存时间 (TTL) 来实现这一点。虽然 TTL 中的 T 代表时间，但 TTL 总体表示数据包在被丢弃之前可以通过的最大路由器/跳数；TTL 并不是最大时间单位数。当路由器接收到一个数据包时，它会将TTL 减一，然后再将其传递给下一个路由器。

如下图所示，IP 包每经过一个路由器，它的 TTL 值就减 1。最初，它以 64 的 TTL 值离开本地系统，经过4个路由器后到达目标系统，TTL值就将变成60。



但是，如果 TTL 达到 0，它将被丢弃，并且一个ICMP Time-to-Live超时消息将被发送给原始发送者。 在下图中，本地系统在发送数据包到路由器之前将 TTL 设置为 1，然后路径上的第一个路由器将 TTL 减 1，导致 TTL 为 0；因此，该路由器将丢弃数据包并发送 ICMP 超时传输错误消息。 请注意，某些路由器配置为在丢弃数据包时不发送此类 ICMP 消息。



在 Linux 上，traceroute命令将首先在 TTL 为 1 的 IP 数据包中发送 UDP 数据报，它会导致第一个路由器遇到 TTL=0 并发送 ICMP Time-to-Live 超时，因此，当TTL 为 1 时将向你显示第一个路由器的 IP 地址。 然后traceroute命令将发送另一个 TTL=2 的数据包，此数据包将在第二个路由器被丢弃.....

在以下示例中，我们从 TryHackMe 的 AttackBox 运行相同的命令 traceroute tryhackme.com，我们注意到不同的运行可能会导致数据包采用不同的路由。

Traceroute A

```
user@AttackBox$ traceroute tryhackme.com
traceroute to tryhackme.com (172.67.69.208), 30 hops max, 60 byte packets
 1  ec2-3-248-240-5.eu-west-1.compute.amazonaws.com (3.248.240.5)  2.663 ms * ec2-3-248-240-13.eu-west-1.compute.amazonaws.com (3.248.240.13)  7.468 ms
 2  100.66.8.86 (100.66.8.86)  43.231 ms 100.65.21.64 (100.65.21.64)  18.886 ms 100.65.22.160 (100.65.22.160)  14.556 ms
 3  * 100.66.16.176 (100.66.16.176)  8.006 ms *
 4  100.66.11.34 (100.66.11.34)  17.401 ms 100.66.10.14 (100.66.10.14)  23.614 ms 100.66.19.236 (100.66.19.236)  17.524 ms
 5  100.66.7.35 (100.66.7.35)  12.808 ms 100.66.6.109 (100.66.6.109)  14.791 ms *
 6  100.65.14.131 (100.65.14.131)  1.026 ms 100.66.5.189 (100.66.5.189)  19.246 ms 100.66.5.243 (100.66.5.243)  19.805 ms
```

```
 7  100.65.13.143 (100.65.13.143)  14.254 ms 100.95.18.131 (100.95.18.131)  0.944 ms
100.95.18.129 (100.95.18.129)  0.778 ms
 8  100.95.2.143 (100.95.2.143)  0.680 ms 100.100.4.46 (100.100.4.46)  1.392 ms
100.95.18.143 (100.95.18.143)  0.878 ms
 9  100.100.20.76 (100.100.20.76)  7.819 ms 100.92.11.36 (100.92.11.36)  18.669 ms
100.100.20.26 (100.100.20.26)  0.842 ms
10  100.92.11.112 (100.92.11.112)  17.852 ms * 100.92.11.158 (100.92.11.158)  16.687
ms
11  100.92.211.82 (100.92.211.82)  19.713 ms 100.92.0.126 (100.92.0.126)  18.603 ms
52.93.112.182 (52.93.112.182)  17.738 ms
12  99.83.69.207 (99.83.69.207)  17.603 ms  15.827 ms  17.351 ms
13  100.92.9.83 (100.92.9.83)  17.894 ms 100.92.79.136 (100.92.79.136)  21.250 ms
100.92.9.118 (100.92.9.118)  18.166 ms
14  172.67.69.208 (172.67.69.208)  17.976 ms  16.945 ms 100.92.9.3 (100.92.9.3)
 17.709 ms
```

在上面的 traceroute 输出中，我们可以看到有14 行编号，每行代表一个路由器/跳。我们的本地系统发送三个 TTL 设置为 1 的数据包，然后发送三个 TTL 设置为 2 的数据包，依此类推。根据网络拓扑，我们可能会从多达 3 个不同的路由器获得回复，这具体取决于数据包所采用的路由。查看第 12 行，具有列出 IP 地址的第 12 个路由器已经丢弃了 3 次数据包并发送了一个 ICMP 超时传输消息。第 12 行的数据：99.83.69.207 (99.83.69.207) 17.603 ms 15.827 ms 17.351 ms 显示了每个回复到达我们系统的时间（以毫秒为单位）。

另一方面，我们可以看到我们在第三行只收到了一个回复，输出记录为： 3 * 100.66.16.176 (100.66.16.176) 8.006 ms * 其中的两个星符号表示我们的系统没有收到两条预期的 ICMP 超时传输消息。

最后，在输出结果的第一行，我们可以看到离开 AttackBox 的数据包采用不同的路由，可以看到响应 TTL 为 1 的两个路由器，我们的系统从未收到第三条预期的 ICMP 消息。

Traceroute B

```
user@AttackBox$ traceroute tryhackme.com
traceroute to tryhackme.com (104.26.11.229), 30 hops max, 60 byte packets
 1  ec2-79-125-1-9.eu-west-1.compute.amazonaws.com (79.125.1.9)  1.475 ms * ec2-3-
248-240-31.eu-west-1.compute.amazonaws.com (3.248.240.31)  9.456 ms
 2  100.65.20.160 (100.65.20.160)  16.575 ms 100.66.8.226 (100.66.8.226)  23.241 ms
100.65.23.192 (100.65.23.192)  22.267 ms
 3  100.66.16.50 (100.66.16.50)  2.777 ms 100.66.11.34 (100.66.11.34)  22.288 ms
100.66.16.28 (100.66.16.28)  4.421 ms
 4  100.66.6.47 (100.66.6.47)  17.264 ms 100.66.7.161 (100.66.7.161)  39.562 ms
100.66.10.198 (100.66.10.198)  15.958 ms
 5  100.66.5.123 (100.66.5.123)  20.099 ms 100.66.7.239 (100.66.7.239)  19.253 ms
100.66.5.59 (100.66.5.59)  15.397 ms
 6  * 100.66.5.223 (100.66.5.223)  16.172 ms 100.65.15.135 (100.65.15.135)  0.424 ms
```

```
 7  100.65.12.135 (100.65.12.135)  0.390 ms 100.65.12.15 (100.65.12.15)  1.045 ms
100.65.14.15 (100.65.14.15)  1.036 ms
 8  100.100.4.16 (100.100.4.16)  0.482 ms 100.100.20.122 (100.100.20.122)  0.795 ms
100.95.2.143 (100.95.2.143)  0.827 ms
 9  100.100.20.86 (100.100.20.86)  0.442 ms 100.100.4.78 (100.100.4.78)  0.347 ms
100.100.20.20 (100.100.20.20)  1.388 ms
10  100.92.212.20 (100.92.212.20)  11.611 ms 100.92.11.54 (100.92.11.54)  12.675 ms
100.92.11.56 (100.92.11.56)  10.835 ms
11  100.92.6.52 (100.92.6.52)  11.427 ms 100.92.6.50 (100.92.6.50)  11.033 ms
100.92.210.50 (100.92.210.50)  10.551 ms
12  100.92.210.139 (100.92.210.139)  10.026 ms 100.92.6.13 (100.92.6.13)  14.586 ms
100.92.210.69 (100.92.210.69)  12.032 ms
13  100.92.79.12 (100.92.79.12)  12.011 ms 100.92.79.68 (100.92.79.68)  11.318 ms
100.92.80.84 (100.92.80.84)  10.496 ms
14  100.92.9.27 (100.92.9.27)  11.354 ms 100.92.80.31 (100.92.80.31)  13.000 ms
52.93.135.125 (52.93.135.125)  11.412 ms
15  150.222.241.85 (150.222.241.85)  9.660 ms 52.93.135.81 (52.93.135.81)  10.941 ms
150.222.241.87 (150.222.241.87)  16.543 ms
16  100.92.228.102 (100.92.228.102)  15.168 ms 100.92.227.41 (100.92.227.41)  10.134
ms 100.92.227.52 (100.92.227.52)  11.756 ms
17  100.92.232.111 (100.92.232.111)  10.589 ms 100.92.231.69 (100.92.231.69)  16.664
ms 100.92.232.37 (100.92.232.37)  13.089 ms
18  100.91.205.140 (100.91.205.140)  11.551 ms 100.91.201.62 (100.91.201.62)  10.246
ms 100.91.201.36 (100.91.201.36)  11.368 ms
19  100.91.205.79 (100.91.205.79)  11.112 ms 100.91.205.83 (100.91.205.83)  11.040 ms
100.91.205.33 (100.91.205.33)  10.114 ms
20  100.91.211.45 (100.91.211.45)  9.486 ms 100.91.211.79 (100.91.211.79)  13.693 ms
100.91.211.47 (100.91.211.47)  13.619 ms
21  100.100.6.81 (100.100.6.81)  11.522 ms 100.100.68.70 (100.100.68.70)  10.181 ms
100.100.6.21 (100.100.6.21)  11.687 ms
22  100.100.65.131 (100.100.65.131)  10.371 ms 100.100.92.6 (100.100.92.6)  10.939 ms
100.100.65.70 (100.100.65.70)  23.703 ms
23  100.100.2.74 (100.100.2.74)  15.317 ms 100.100.66.17 (100.100.66.17)  11.492 ms
100.100.88.67 (100.100.88.67)  35.312 ms
24  100.100.16.16 (100.100.16.16)  19.155 ms 100.100.16.28 (100.100.16.28)  19.147 ms
100.100.2.68 (100.100.2.68)  13.718 ms
25  99.83.89.19 (99.83.89.19)  28.929 ms *  21.790 ms
26  104.26.11.229 (104.26.11.229)  11.070 ms  11.058 ms  11.982 ms
```

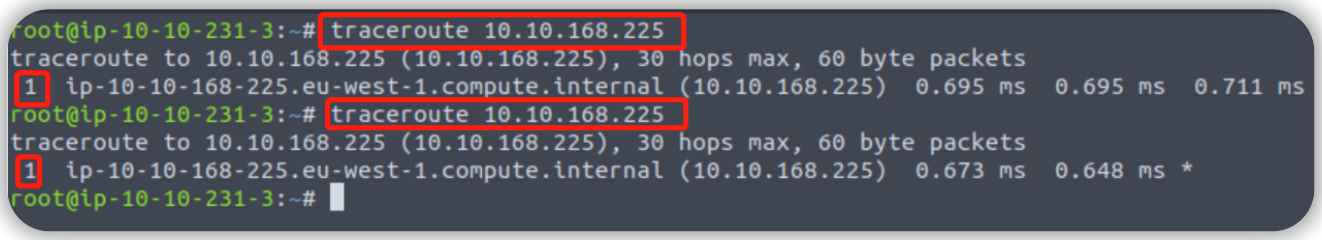
在 traceroute 程序的第二次运行中，我们注意到这次数据包经过了更长的路由，经过了 26 个路由器。如果你正在对网络中的系统运行跟踪路由命令，则该路由的路径不太可能更改，但是，当数据包需要通过我们网络之外的其他路由器时，我们不能期望路由的路径保持固定。

总而言之,我们可以注意以下几点:

- 你的本地系统和目标系统之间的跃点/路由器数量取决于你运行 traceroute 的时间。 即使你在同一网络上或你在短时间内重复 traceroute 命令，也无法保证你的数据包将始终遵循相同的路由路径。

- 一些路由器会返回一个公共 IP 地址。 你可以根据预期的渗透测试范围检查其中一些路由器。
- 一些路由器不会返回 回复信息。

答题



H2 使用Telnet进行主动信息收集

TELNET（电传网络）协议于 1969 年开发，用于通过命令行界面 (CLI) 与远程系统进行通信。命令 telnet 正是使用 TELNET 协议进行远程管理，telnet 使用的默认端口是 23。从安全角度来看，telnet 以明文形式发送所有数据，包括用户名和密码，以明文形式发送可以让任何有权访问通信通道的人轻松窃取登录凭据，telnet 的安全替代方案是 SSH (Secure SHell) 协议。

telnet 客户端因其简单性可以用于其他目的：知道了telnet 客户端依赖于 TCP 协议，你就可以尝试使用 telnet 连接到目标上基于tcp的任何服务。

使用 telnet MACHINE_IPPORT，你可以连接到目标机的TCP 上运行的任何服务，甚至可以交换一些消息，除非这些信息使用了加密。

假设我们希望发现关于监听端口80的目标Web 服务器的更多信息，我们可以连接到目标服务器的80端口，然后使用 HTTP 协议进行通信。在此处，你无需深入研究 HTTP 协议，你只需要发出 GET / HTTP/1.1 请求。如果你要指定默认索引页面以外的内容，你可以发出 GET /page.html HTTP/1.1，它将请求 page.html页面。我们在此处还向远程 Web 服务器指定要使用 HTTP协议的1.1 版本进行通信。

为了获得有效的响应消息，而不是错误消息，你还需要为host参数输入一些值 "host: example"并按两次回车键。执行这些步骤之后，目标服务器将提供你所请求的索引页面。

```
pentester@TryHackMe$ telnet MACHINE_IPPORT 80
Trying 10.10.168.225 ...
Connected to MACHINE_IP.
Escape character is '^]'.
GET / HTTP/1.1
host: telnet

HTTP/1.1 200 OK
Server: nginx/1.6.2
Date: Tue, 17 Aug 2021 11:13:25 GMT
Content-Type: text/html
Content-Length: 867
Last-Modified: Tue, 17 Aug 2021 11:12:16 GMT
Connection: keep-alive
ETag: "611b9990-363"
Accept-Ranges: bytes

...
```

我们的目的是尝试发现已安装的 Web 服务器的类型和版本，例如Server: nginx/1.6.2。 在上面的示例中，我们与 Web 服务器进行通信，因此我们使用了基本的 HTTP 命令，如果我们想连接到邮件服务器，那么我们需要根据协议使用适当的命令，比如 SMTP 和 POP3。

答题

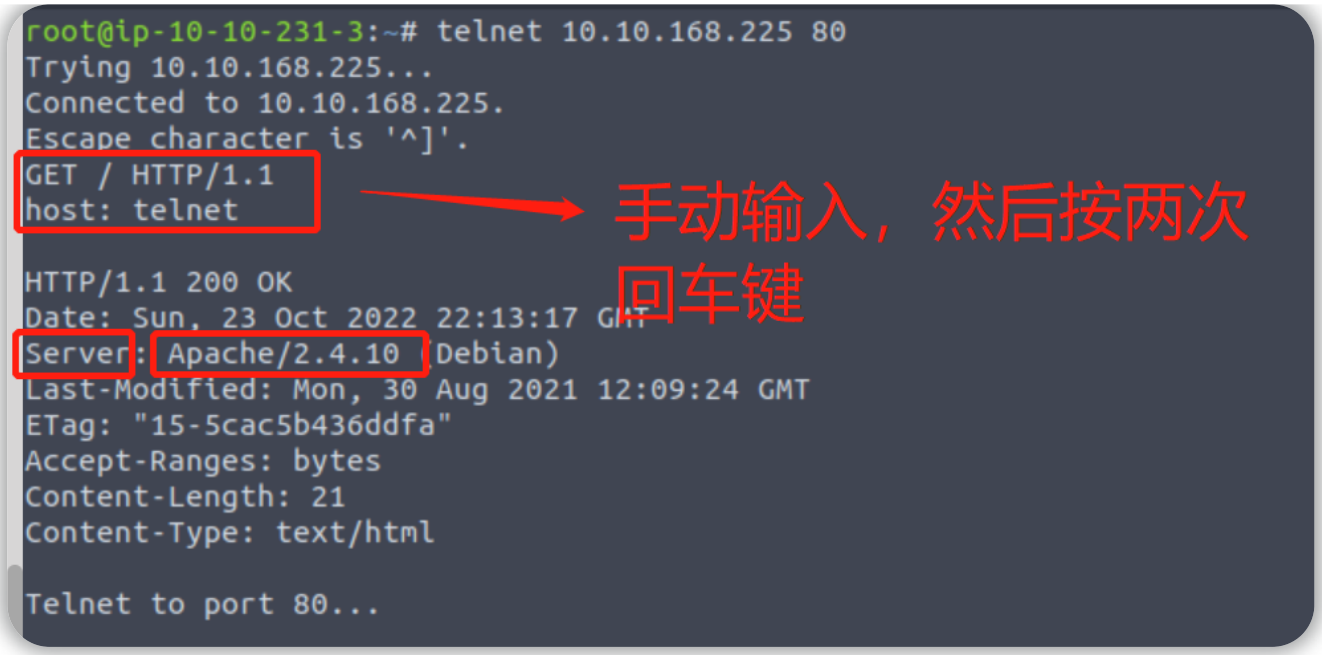
回答以下问题

如果尚未启动，则从任务 3 启动附加的 VM。在 AttackBox 上，打开终端并使用 telnet 客户端连接到 VM 的 80 端口。正在运行的服务器的名称是什么？

Apache　正确答案

正在运行的服务器的版本是什么（在 VM 的端口 80 上）？

2.4.10　正确答案



H2 使用Netcat进行主动信息收集

Netcat（或者简单的nc）有不同的应用，这对渗透测试者来说可能很有价值。Netcat 支持 TCP 和 UDP 协议，它可以充当连接到侦听端口的客户端；或者，它可以充当侦听你选择的端口的服务器。因此，它是一个很方便的工具，你可以通过 TCP 或 UDP 将其用作简单的客户端或服务器。

你可以像使用telnet一样使用netcat连接到目标服务器，输入 `nc MACHINE_IP PORT` 收集目标的banner（横幅）信息，这与我们之前的telnet MACHINE_IP PORT命令非常相似。 请注意，你可能需要在 GET 行之后按 SHIFT+ENTER。

```
pentester@TryHackMe$ nc MACHINE_IP 80
GET / HTTP/1.1
host: netcat

HTTP/1.1 200 OK
Server: nginx/1.6.2
Date: Tue, 17 Aug 2021 11:39:49 GMT
Content-Type: text/html
Content-Length: 867
Last-Modified: Tue, 17 Aug 2021 11:12:16 GMT
Connection: keep-alive
ETag: "611b9990-363"
Accept-Ranges: bytes
...
```

在上面显示的输出中，我们使用 nc MACHINE_IP 80 连接到 MACHINE_IP 的端口 80；接下来，我们使用 GET / HTTP/1.1针对目标服务器的默认页面发出 get请求，我们还向目标服务器指定我们的客户端支持 HTTP协议的 1.1 版本；最后，我们需要给我们的host起一个名字，所以我们添加了一个新的行--host:netcat，在此处你可以任意命名host。

根据我们收到的输出 Server: nginx/1.6.2 ，我们可以知道在目标的端口80 上，版本为1.6.2的Nginx正在监听传入的连接。

你可以使用 netcat 监听 TCP 端口并连接到另一个系统上的监听端口。在服务器系统上，如果你希望打开一个端口并监听它，可以使用nc -lp 1234或者nc -vnlp 1234，这相当于 nc -v -l -n -p 1234，此处字母的确切顺序无关紧要，只要你指定的端口号前面带有-p参数就行。

option	meaning
-l	Listen mode
-p	Specify the Port number
-n	Numeric only; no resolution of hostnames via DNS
-v	Verbose output (optional, yet useful to discover any bugs)
-vv	Very Verbose (optional)
-k	Keep listening after client disconnects 客户端断开之后保持监听

注意：

- -p 选项应该出现在你要监听的端口号之前。
- -n 选项将避免 DNS 查找和警告。
- 小于 1024 的端口号需要 root 权限才能监听。

在客户端，你可以使用 nc MACHINE_IP PORT_NUMBER ，当你成功建立与目标服务器的连接后，你在客户端键入的任何内容都将在服务器端回显，反之亦然。

考虑以下示例： 在服务器端，我们设置监听器监听端口 1234 ，我们可以使用命令 nc -vnlp 1234（与 nc -lvnp 1234 相同）来实现这一点，目标服务器的 IP 地址为 MACHINE_IP，因此我们可以通过执行 nc MACHINE_IP 1234 从客户端连接到目标服务器，成功完成此连接之后，你在 TCP 隧道一侧键入的任何内容都将回显到另一侧。

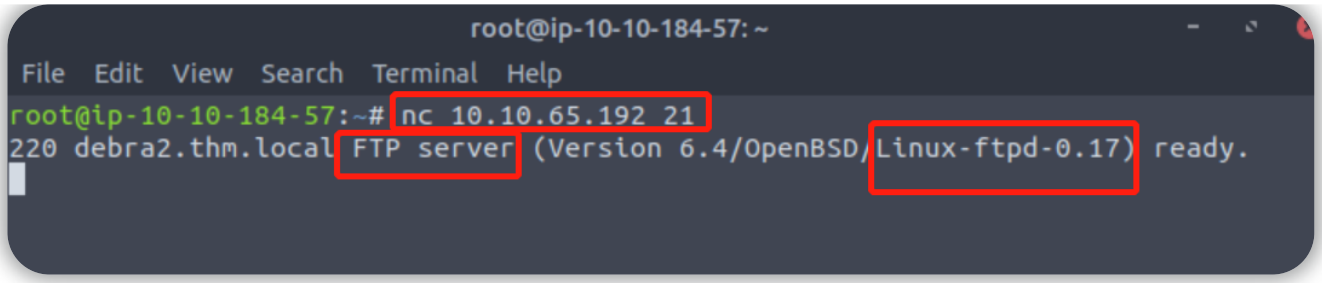
答题

回答以下问题

启动VM并打开 AttackBox。一旦 AttackBox 加载，使用 Netcat 连接到 VM 端口 21。正在运行的服务器的版本是什么？

0.17

正确答案



H2 小结

在本文中，我们介绍了许多不同的工具，通过 shell 脚本可以将多个工具的功能放在一起，以构建一个原始的网络和系统扫描器。

你可以使用 traceroute 映射出从本地机器到目标机器的路由路径，使用 ping命令检查目标系统是否响应 ICMP Echo，并使用 telnet 尝试连接目标--来检查目标上哪些端口是开放的和可访问的。一个可用的扫描器会在更高级和复杂的级别上执行以上的一系列操作，比如nmap扫描。

Command	Example
ping	`ping -c 10 MACHINE_IP` on Linux or macOS
ping	`ping -n 10 MACHINE_IP` on MS Windows
traceroute	`traceroute MACHINE_IP` on Linux or macOS
tracert	`tracert MACHINE_IP` on MS Windows
telnet	`telnet MACHINE_IP PORT_NUMBER`
netcat as client	`nc MACHINE_IP PORT_NUMBER`
netcat as server	`nc -lvnp PORT_NUMBER`

尽管本文所涉及的都是一些基本工具，但它们在大多数系统上都很容易获得，特别是，几乎每台计算机和智能手机上都安装了web浏览器，它可以成为你的武器库中的重要工具，能够帮助你在不引发警报的情况下进行信息收集活动。

调出浏览器中的开发者工具的快捷键：

Operating System	Developer Tools Shortcut
Linux or MS Windows	`Ctrl+Shift+I`
macOS	`Option + Command + I`