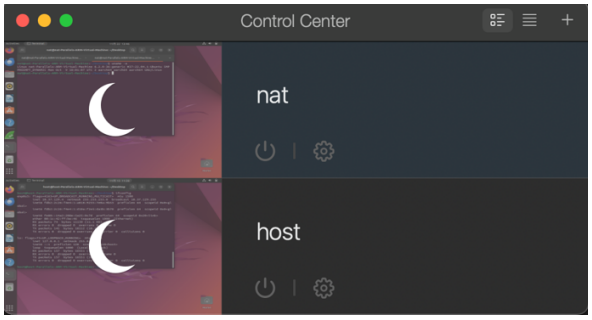
实验名称	NAT 实验		
姓名	刘培源	学号	2023214278

实验步骤

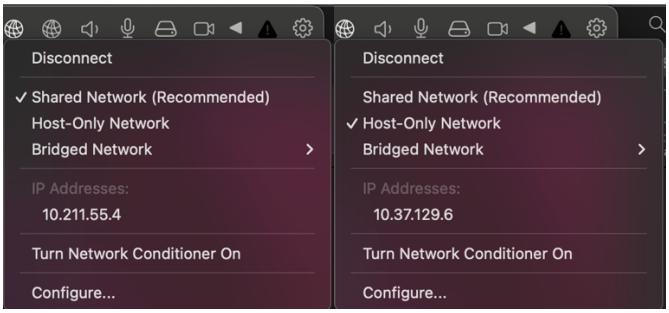
实验采用 macOS 系统（版本 Sonoma 14.1.1）的 parallel Desktop 软件安装版本为 22.04.1 的 Ubuntu 虚拟机。一共安装了两个虚拟机，分别是 nat 和 host。



后续步骤如下：

1. **配置 nat 机器：**

   对于 nat 机器，我首先给其分配了两个网卡，分别为 enp0s5（ip 为 10.211.55.4）和 enp0s6（ip 为 10.37.129.3）。其中 enp0s5 采用 shared network 模式（即为 NAT 模式），enp0s6 采用 host-only network 模式（即为 internel-network 模式）。设置图片图下：



   由于 parallel Desktop 软件会默认把两个虚拟机的 host-only 网络分配到同一个局域网下，故不需要手动设置。

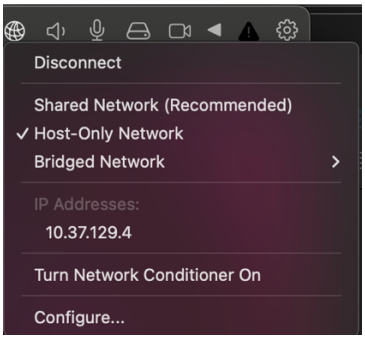
   随后依次通过以下 terminal 命令来设置 ip 地址、打开转发功能和设置 NAT 功能：

   (1) sudo dhclient enp0s5
   (2) sudo gedit /etc/sysctl.conf 然后取消其中对于 net.ipv4.ip_forward=1 的注释
   (3) sudo iptables -t nat -A POSTROUTING -s "10.37.129.3/24" -o enp0s5 -j MASQUERADE
   (4) sudo /sbin/sysctl -p

   经过以上步骤，nat 机器配置完毕。（注：我省略了配置 host-only 网卡的 ip 地址，原因是 parallel 软件会默认分配内部局域网的 ip，如果手动配置会出问题，详细问题在经验总结中给出。）

```
nat@nat-Parallels-ARM-Virtual-Machine:~/Desktop$ sudo dhclient enp0s5
RTNETLINK answers: File exists
nat@nat-Parallels-ARM-Virtual-Machine:~/Desktop$ sudo gedit /etc/sysctl.conf

(gedit:2582): dconf-WARNING **: 14:40:06.311: failed to commit changes to dconf: Failed to
execute child process “dbus-launch” (No such file or directory)

(gedit:2582): dconf-WARNING **: 14:40:06.313: failed to commit changes to dconf: Failed to
execute child process “dbus-launch” (No such file or directory)

(gedit:2582): dconf-WARNING **: 14:40:06.406: failed to commit changes to dconf: Failed to
execute child process “dbus-launch” (No such file or directory)

(gedit:2582): dconf-WARNING **: 14:40:06.406: failed to commit changes to dconf: Failed to
execute child process “dbus-launch” (No such file or directory)

(gedit:2582): dconf-WARNING **: 14:40:06.406: failed to commit changes to dconf: Failed to
execute child process “dbus-launch” (No such file or directory)

** (gedit:2582): WARNING **: 14:40:07.632: Set document metadata failed: Setting attribute
metadata::gedit-position not supported

(gedit:2582): dconf-WARNING **: 14:40:07.633: failed to commit changes to dconf: Failed to
execute child process “dbus-launch” (No such file or directory)
nat@nat-Parallels-ARM-Virtual-Machine:~/Desktop$ sudo iptables -t nat -A POSTROUTING -s "10
.37.129.3/24" -o enp0s5 -j MASQUERADE
nat@nat-Parallels-ARM-Virtual-Machine:~/Desktop$ sudo /sbin/sysctl -p
net.ipv4.ip_forward = 1
```

2. **配置 host 机器：**给 host 机分配一个网卡为 enp0s5（ip 为 10.37.129.4），其采用了 host-only network 模式（即为 internel-network 模式），设置图片如下。



随后依次通过以下 terminal 命令来修改 DNS 设置局域网内默认网关（注意同样省略了修改 ip 的过程）：

(1) sudo gedit /etc/resolv.conf

(2) sudo route add default gw 10.37.129.3 （这里为 nat 机器 enp0s6 的 ip）

```
host@host-Parallels-ARM-Virtual-Machine:~/Desktop$ sudo gedit /etc/resolv.conf
[sudo] password for host:

(gedit:2660): dconf-WARNING **: 14:45:06.436: failed to commit changes to dconf:
 Failed to execute child process “dbus-launch” (No such file or directory)

(gedit:2660): dconf-WARNING **: 14:45:06.438: failed to commit changes to dconf:
 Failed to execute child process “dbus-launch” (No such file or directory)

(gedit:2660): dconf-WARNING **: 14:45:06.541: failed to commit changes to dconf:
 Failed to execute child process “dbus-launch” (No such file or directory)

(gedit:2660): dconf-WARNING **: 14:45:06.541: failed to commit changes to dconf:
 Failed to execute child process “dbus-launch” (No such file or directory)

** (gedit:2660): WARNING **: 14:45:07.726: Set document metadata failed: Setting
 attribute metadata::gedit-position not supported

(gedit:2660): dconf-WARNING **: 14:45:07.727: failed to commit changes to dconf:
 Failed to execute child process “dbus-launch” (No such file or directory)
host@host-Parallels-ARM-Virtual-Machine:~/Desktop$ sudo route add default gw 10.
37.129.3
```

3. 配置好 nat 机器和 host 机器之后，尝试用 host 机器来 ping 百度的网址（www.baidu.com），成功 ping 通的截图如下：

```
host@host-Parallels-ARM-Virtual-Machine:~/Desktop$ ping www.baidu.com
PING www.a.shifen.com (183.2.172.42) 56(84) bytes of data.
64 bytes from 183.2.172.42 (183.2.172.42): icmp_seq=1 ttl=127 time=11.7 ms
64 bytes from 183.2.172.42 (183.2.172.42): icmp_seq=2 ttl=127 time=14.0 ms
64 bytes from 183.2.172.42 (183.2.172.42): icmp_seq=3 ttl=127 time=11.5 ms
64 bytes from 183.2.172.42 (183.2.172.42): icmp_seq=4 ttl=127 time=14.8 ms
64 bytes from 183.2.172.42 (183.2.172.42): icmp_seq=5 ttl=127 time=12.9 ms
64 bytes from 183.2.172.42 (183.2.172.42): icmp_seq=6 ttl=127 time=14.5 ms
^C
--- www.a.shifen.com ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5014ms
rtt min/avg/max/mdev = 11.459/13.225/14.809/1.321 ms
```

4. 通过如下命令在 nat 机器上安装 wireshark：

   (1) sudo apt update

   (2) sudo apt install wireshark

   安装成功的截图如下：

```
nat@nat-Parallels-ARM-Virtual-Machine:~/Desktop$ wireshark --version
Wireshark 3.6.2 (Git v3.6.2 packaged as 3.6.2-2)

Copyright 1998-2022 Gerald Combs <gerald@wireshark.org> and contributors.
License GPLv2+: GNU GPL version 2 or later <https://www.gnu.org/licenses/gpl-2.0.html>
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

Compiled (64-bit) using GCC 11.2.0, with Qt 5.15.2, with libpcap, with POSIX
capabilities (Linux), with libnl 3, with GLib 2.71.2, with zlib 1.2.11, with Lua
5.2.4, with GnuTLS 3.7.3 and PKCS #11 support, with Gcrypt 1.9.4, with MIT
Kerberos, with MaxMind DB resolver, with nghttp2 1.43.0, with brotli, with LZ4,
with Zstandard, with Snappy, with libxml2 2.9.12, with libsmi 0.4.8, with
QtMultimedia, without automatic updates, with SpeexDSP (using system library),
with Minizip.

Running on Linux 6.2.0-36-generic, with 1962 MB of physical memory, with GLib
2.72.4, with zlib 1.2.11, with Qt 5.15.3, with libpcap 1.10.1 (with TPACKET_V3),
with c-ares 1.18.1, with GnuTLS 3.7.3, with Gcrypt 1.9.4, with nghttp2 1.43.0,
with brotli 1.0.9, with LZ4 1.9.3, with Zstandard 1.4.8, with libsmi 0.4.8, with
LC_TYPE=en_US.UTF-8, binary plugins supported (0 loaded).
```

   后续使用 wireshark 进行 nat 数据分析在下一部分给出。

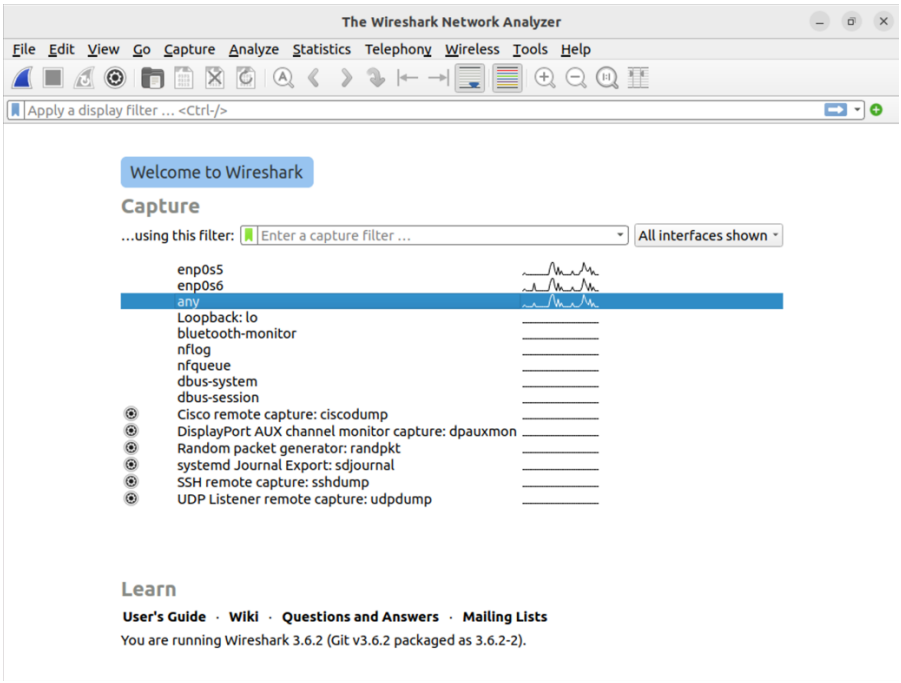
数据分析

此处给出在 nat 机器上用 wireshark 进行分析的结果：

1. 在 nat 机器上通过 sudo add-apt-repository universe 和 sudo apt install wireshark 来安装 wireshark 软件。安装完成后，通过 wireshark –version 来确保安装成功，截图如下：

```
nat@nat-Parallels-ARM-Virtual-Machine:~/Desktop$ wireshark --version
Wireshark 3.6.2 (Git v3.6.2 packaged as 3.6.2-2)

Copyright 1998-2022 Gerald Combs <gerald@wireshark.org> and contributors.
License GPLv2+: GNU GPL version 2 or later <https://www.gnu.org/licenses/gpl-2.0
.html>
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

Compiled (64-bit) using GCC 11.2.0, with Qt 5.15.2, with libpcap, with POSIX
capabilities (Linux), with libnl 3, with GLib 2.71.2, with zlib 1.2.11, with Lua
5.2.4, with GnuTLS 3.7.3 and PKCS #11 support, with Gcrypt 1.9.4, with MIT
Kerberos, with MaxMind DB resolver, with nghttp2 1.43.0, with brotli, with LZ4,
with Zstandard, with Snappy, with libxml2 2.9.12, with libsmi 0.4.8, with
QtMultimedia, without automatic updates, with SpeexDSP (using system library),
with Minizip.

Running on Linux 6.2.0-36-generic, with 1962 MB of physical memory, with GLib
2.72.4, with zlib 1.2.11, with Qt 5.15.3, with libpcap 1.10.1 (with TPACKET_V3),
with c-ares 1.18.1, with GnuTLS 3.7.3, with Gcrypt 1.9.4, with nghttp2 1.43.0,
with brotli 1.0.9, with LZ4 1.9.3, with Zstandard 1.4.8, with libsmi 0.4.8, with
LC_TYPE=en_US.UTF-8, binary plugins supported (0 loaded).
```

2. 通过 sudo wireshark 来启动 wireshark，随后选择捕捉所有网卡(any)的信息，配置如下：



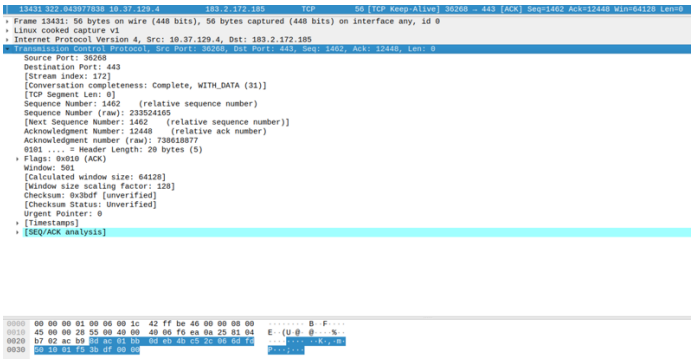
3. 在 host 机器上执行 ping baidu.com 命令。在经历了若干个往返时延（RTT）之后，停止该命令。接下来，转到 nat 机器上，使用 wireshark 查看并分析记录下来的数据。如下图所示，这个过程揭示了一个 ping 请求的 NAT 地址转换过程。最初，请求由主机的 IP 地址 10.37.129.4 发出，然后通过 NAT 被转换到 10.211.55.4，这是 NAT 机器上共享网络（shared-network）网卡的 IP 地址。之后，请求被发送到百度的 IP 地址 110.242.68.66。在接收应答时，百度服务器首先将回复发送到 10.211.55.4，然后通过 NAT 进程，这个地址被重新翻译回主机的原始 IP 地址 10.37.129.4。这样，主机便接收到回复，从而完成了一个 RTT 循环。

No.	Time	Source	Destination	Protocol	Length	Info
14	3.208763793	10.37.129.4	110.242.68.66	ICMP	100	Echo (ping) request
15	3.208854295	10.211.55.4	110.242.68.66	ICMP	100	Echo (ping) request
16	3.460508157	110.242.68.66	10.211.55.4	ICMP	100	Echo (ping) reply
17	3.460552242	110.242.68.66	10.37.129.4	ICMP	100	Echo (ping) reply

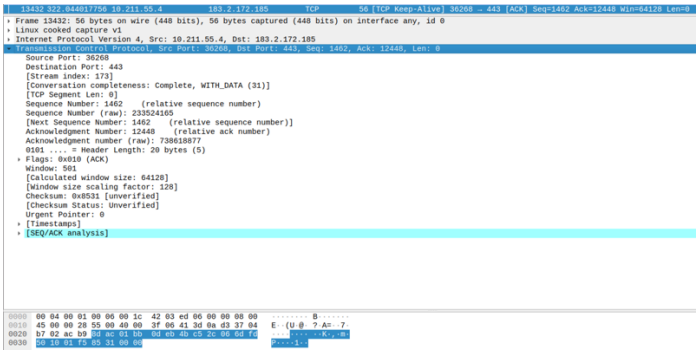
4. 接着来进行端口分析，由于 ping 采用的 ICMP 协议，其并没有端口号标记应用层的服务，所以我在 host 机器的 firefox 浏览器输入了 www.baidu.com 来访问百度，想通过 TCP 协议来分析端口。下面这张图显示了 TCP 协议经过了 NAT 翻译的过程，可以发现其与 ping 的过程是一致的：

```
13431 322.043977838 10.37.129.4      183.2.172.185   TCP      56 [TCP Keep-Alive] 36268 → 443 [ACK] Seq=1462 Ack=12448 Win=64128 Len=0
13432 322.044017756 10.211.55.4      183.2.172.185   TCP      56 [TCP Keep-Alive] 36268 → 443 [ACK] Seq=1462 Ack=12448 Win=64128 Len=0
13433 322.044279259 183.2.172.185    10.211.55.4     TCP      56 [TCP Keep-Alive ACK] 443 → 36268 [ACK] Seq=12448 Ack=1463 Win=32768 Len=0
13434 322.044289759 183.2.172.185    10.37.129.4     TCP      56 [TCP Keep-Alive ACK] 443 → 36268 [ACK] Seq=12448 Ack=1463 Win=32768 Len=0
```

5. 接着，我们看具体的包的内容。在初始阶段，主机（host）对百度的访问请求从 10.37.129.4:36268 发出，并被发送到百度的地址 183.2.172.185:443，其校验和（checksum）为 0x3bdf：

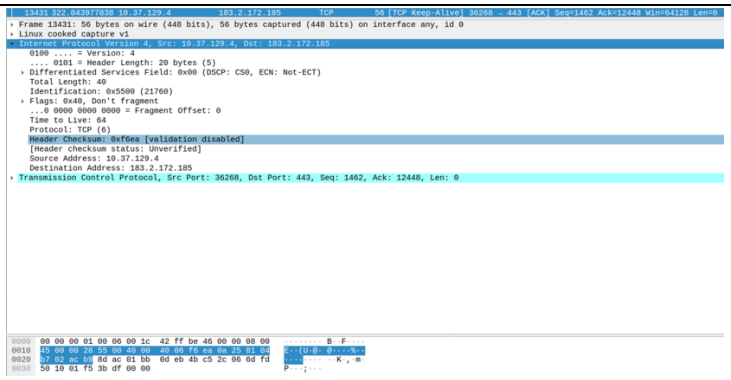


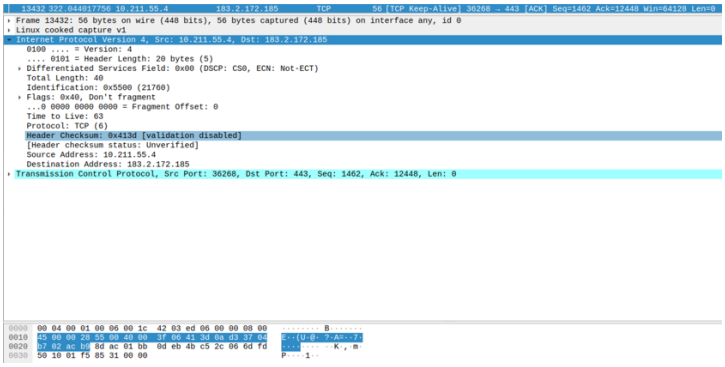
在经过 NAT 翻译之后，该请求的源 IP 地址被转换为 NAT 机器的共享网络（shared-network）网卡的 IP 地址 10.211.55.4。值得注意的是，尽管 IP 地址发生了变化，但发送和接收的端口号以及目标 IP 地址保持不变，新的校验和变为 0x8531：



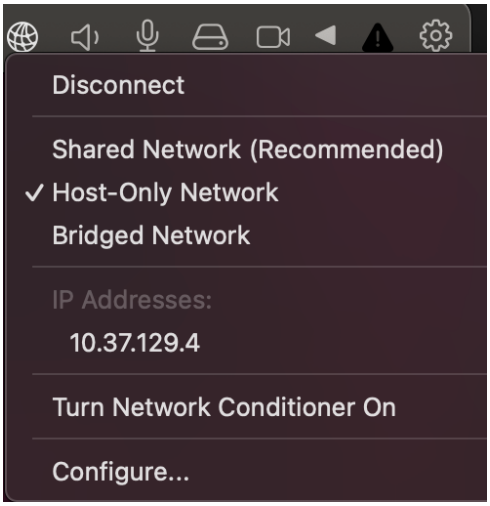
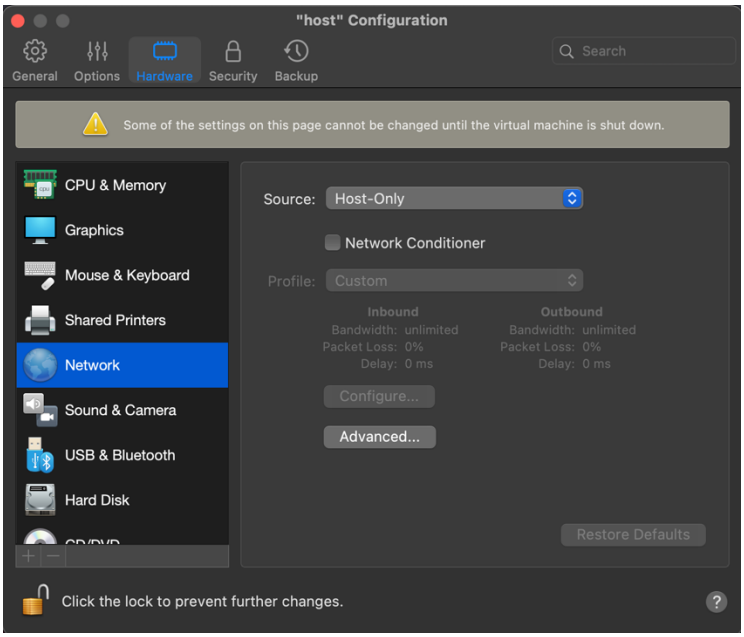
进一步分析显示，在 NAT 翻译之前和之后，TCP 数据包的内容（在图中以蓝色高亮显示）几乎完全一致，唯一的不同之处在于校验和。这种变化是由 IP 层的变化引起的。

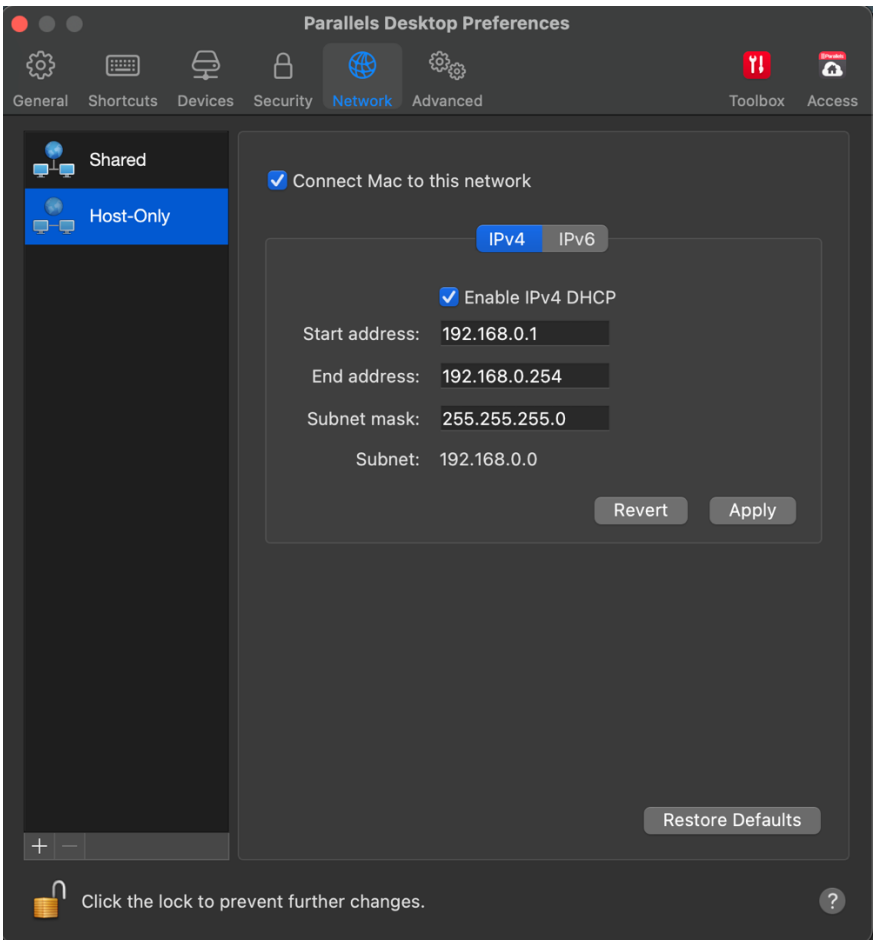
6. 针对 IP 层的进一步分析显示，主机发出的信号的 IP 层信息以及而经过 NAT 翻译之后的 IP 层信息如下面两张图所示。可以观察到，在 NAT 翻译前后，IP 层的校验和也从 0xf6ea 变为 0x413d。这种变化与 NAT 过程中 IP 地址的转换一致，符合预期行为，至此分析到此结束。


13431 322.043977838 10.37.129.4 183.2.172.185 TCP 56 [TCP Keep-Alive] 36268 → 443 [ACK] Seq=1462 Ack=12448 Win=64128 Len=0
Frame 13431: 56 bytes on wire (448 bits), 56 bytes captured (448 bits) on interface any, id 0
Linux cooked capture v1
Internet Protocol Version 4, Src: 10.37.129.4, Dst: 183.2.172.185
0100 .... = Version: 4
.... 0101 = Header Length: 20 bytes (5)
Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
Total Length: 40
Identification: 0x5500 (21760)
Flags: 0x40, Don't fragment
...0 0000 0000 0000 = Fragment Offset: 0
Time to Live: 64
Protocol: TCP (6)
Header Checksum: 0xf6ea [validation disabled]
[Header checksum status: Unverified]
Source Address: 10.37.129.4
Destination Address: 183.2.172.185
Transmission Control Protocol, Src Port: 36268, Dst Port: 443, Seq: 1462, Ack: 12448, Len: 0
0000 00 00 00 01 00 06 00 1c 42 ff be 46 00 00 08 00
0010 45 00 00 28 55 00 40 00 40 06 f6 ea 0a 25 81 04
0020 b7 02 ac b0 8d ac 01 bb 0d eb 4b c5 2c 06 6d fd
0030 50 10 01 f5 3b df 00 00


13432 322.044017756 10.211.55.4 183.2.172.185 TCP 56 [TCP Keep-Alive] 36268 → 443 [ACK] Seq=1462 Ack=12448 Win=64128 Len=0
Frame 13432: 56 bytes on wire (448 bits), 56 bytes captured (448 bits) on interface any, id 0
Linux cooked capture v1
Internet Protocol Version 4, Src: 10.211.55.4, Dst: 183.2.172.185
0100 .... = Version: 4
.... 0101 = Header Length: 20 bytes (5)
Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
Total Length: 40
Identification: 0x5500 (21760)
Flags: 0x40, Don't fragment
...0 0000 0000 0000 = Fragment Offset: 0
Time to Live: 63
Protocol: TCP (6)
Header Checksum: 0x413d [validation disabled]
[Header checksum status: Unverified]
Source Address: 10.211.55.4
Destination Address: 183.2.172.185
Transmission Control Protocol, Src Port: 36268, Dst Port: 443, Seq: 1462, Ack: 12448, Len: 0
0000 00 04 00 01 00 06 00 1c 42 03 ed 06 00 00 08 00
0010 45 00 00 28 55 00 40 00 3f 06 41 3d 0a d3 37 04
0020 b7 02 ac b0 8d ac 01 bb 0d eb 4b c5 2c 06 6d fd
0030 50 10 01 f5 85 31 00 00

思考题	**题 1: NAT 协议中，需要对 UDP 会话中的 UDP 校验和进行修改，选哪个修改。为什么 UDP 校验和为 0 时不需要修改？请结合 UDP 原理进行回答。** **答 1:** 在 NAT 协议中，会修改 UDP 数据包中的校验和（check sum），包括 UDP 报头的校验和以及 IP 头的校验和。这是因为 NAT 改变了 IP 报头中的 IP 地址，进而影响了 IP 头的校验和。同时，UDP 报头的校验和也包含了 IP 头（伪首部）的信息，因此 IP 头的任何变化也需要相应地调整 UDP 校验和。然而，当 UDP 校验和设为 0 时，意味着该包内容无需校验。这是 UDP 协议的一个特性，表明数据包可以传输而无需确保其完整性。因此，在这种情况下，NAT 不会修改校验和，以保持数据包的这一特性不变。 **题 2: 跨越 NAT 网关还能使用 ping 和 traceroute 吗？** **答 2:** 跨越 NAT 网关时，ping 和 traceroute 仍然可用。ping 命令用于确认目标地址是否可达，而 traceroute 命令用于追踪数据包到达目的地的路径和确定网络连接故障点。内部网络（内网）向外部网络（外网）使用 ping 和 traceroute 通常是可行的。然而，traceroute 默认发送端口号大于 30000 的 UDP 数据包，可能会遇到端口限制问题。从外部网络对 NAT 内部的主机执行 ping 和 traceroute 时，需要使用 NAT 穿透技术。这样可以达到目标主机，但无法定位到具体的进程，因为没有使用进程相关的端口号等信息。 **题 3: 分析 NAT 技术的优缺点：** **答 3:** 优点： （1） 节约了有限的 IPv4 地址资源，提高了地址利用率。 （2） 允许网络地址重叠，解决了地址冲突和重新编号的问题。 （3） 增加了加入互联网的灵活性，并在网络结构变动时减少了重新配置地址的需要。 缺点： （1） 由于地址转换的过程，可能会引入额外的延迟。 （2） 阻碍了端到端的 IP 可追踪性，影响了网络的透明性，同时增加了网络的复杂性。 （3） 不兼容某些应用程序，如 SNMP 等，这些应用需要知道终端的真实 IP 地址。 （4） NAT 表的维护需要额外的内存资源。 （5） 地址转换过程增加了中央处理器的负载，可能会影响设备性能。

经验总结	由于我使用的是 M2 芯片（arm 架构）的 macOS 系统，且采用的虚拟机软件是 parallel Desktop，因此在给网卡设定 ip 地址的时候，不可以只用 sudo ifconfig <网卡名称> <ip 地址> 来操作，这里为以后的 MAC 用户的同学提供一个 parallel（版本 19.1.0）下面 ubuntu22.04 arm 版本设置 ip 地址的方法，具体设置如下： 1. 打开 parallel 机器上的网络设置(configure)，如下图，选择 advanced 设定，然后选择 open network preferences：   2. 在 network preferences 里面将 start address 和 end address 分别改成 192.168.0.1 和 192.168.0.254（注意是修改 host-only network），随后点击 apply，如下图：



3. 修改好以后，进行如下操作（以 host 机器为例），用 sudo ip addr flush dev enp0s5 将网卡的 IP 地址清空，再用 sudo ifconfig 192.168.0.2 enp0s5 设定即可。要修改 nat 机器的第二张网卡也是一样的操作。