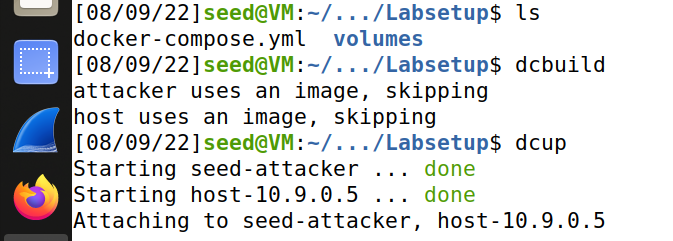
**Task 1 - Using Scapy to Sniff and Spoof Packets**

**准备工作**

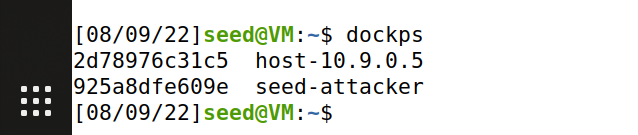
执行dcbuild和dcup创建容器镜像并启动容器：seed-attacker和host-10.9.0.5。



新开启一个终端，查看网络配置信息，并记录下网桥名称：br-8e28585c9941。



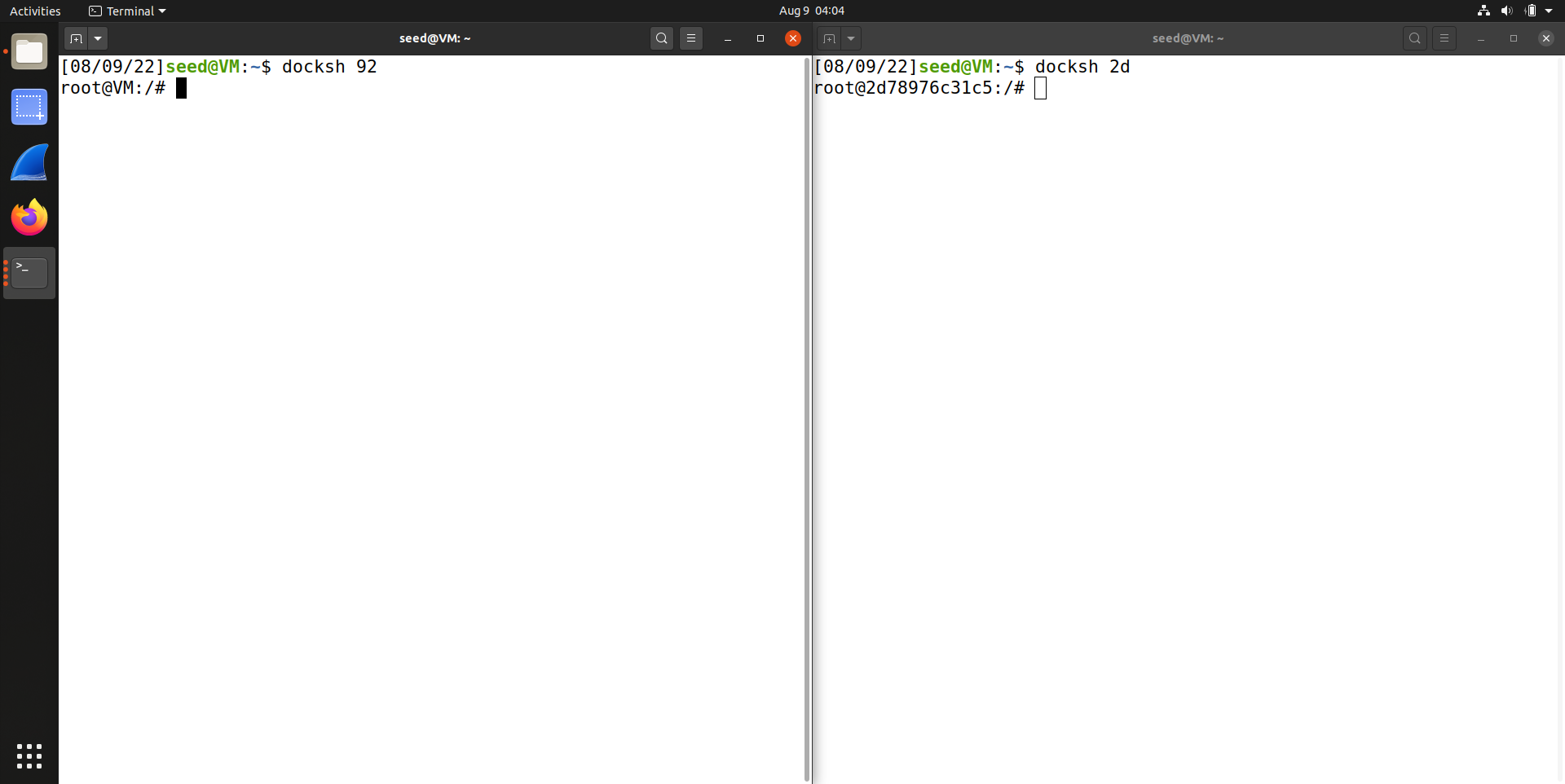
查看两个容器的ID：攻击容器以92开头，受攻击容器以2d开头。



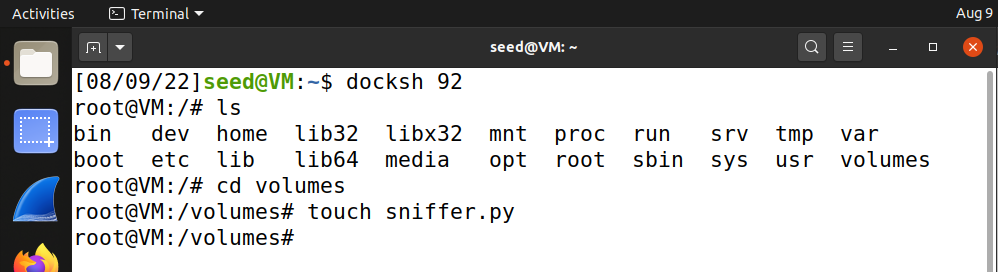
* 1. **Sniffing packets**

**1A Simple Sniffing**

另外新开启两个终端，分别连接到两个容器：左侧为攻击容器，右侧为受攻击容器。



在/volumes文件下下创建sniffer.py并编写。





在攻击容器中执行sniffer.py，开始监听。



被攻击容器中通过ping命令发送ICMP包，可在攻击容器中监听到。

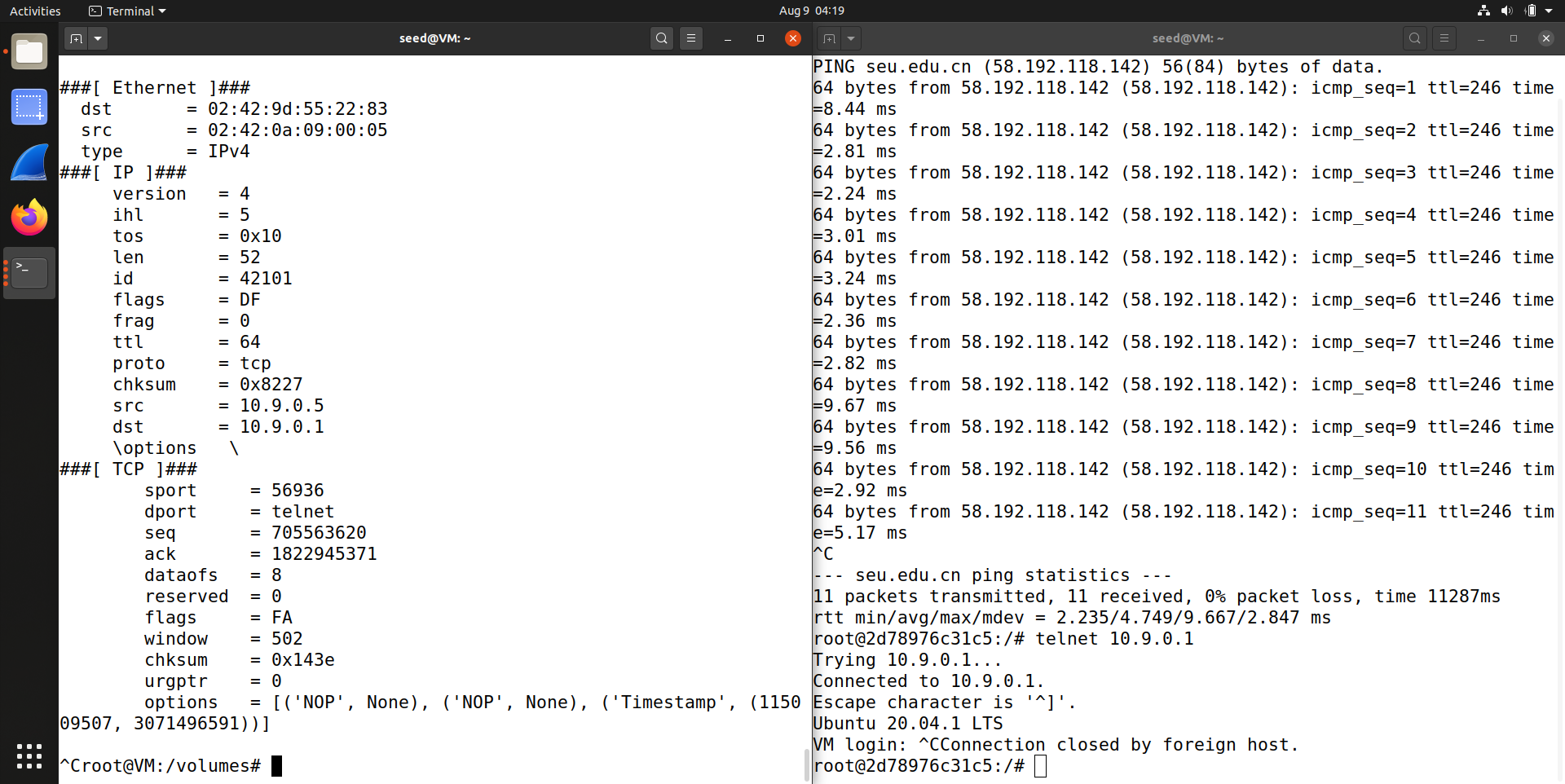


**1B 切换规则实现监听不同的报文**

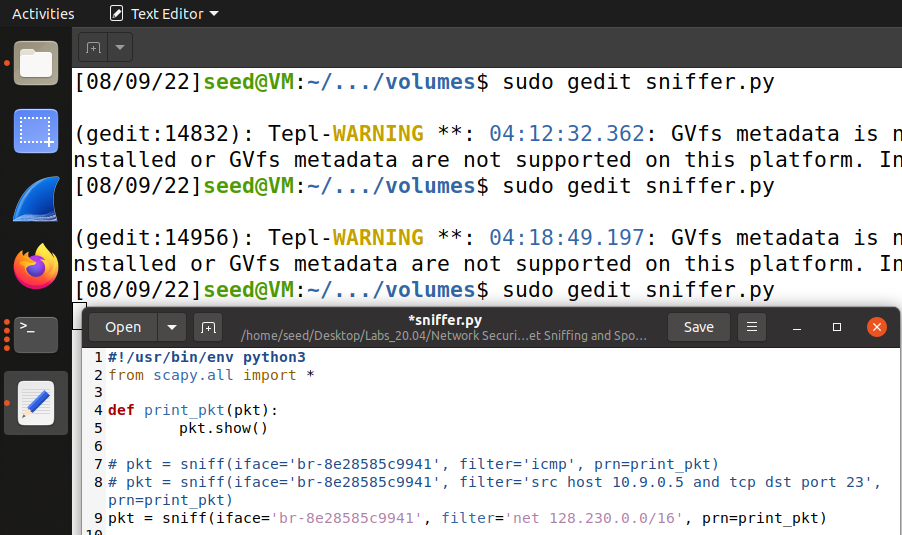
修改sniffer.py中的filter，只监听源地址为10.9.0.5，端口号为23的报文。



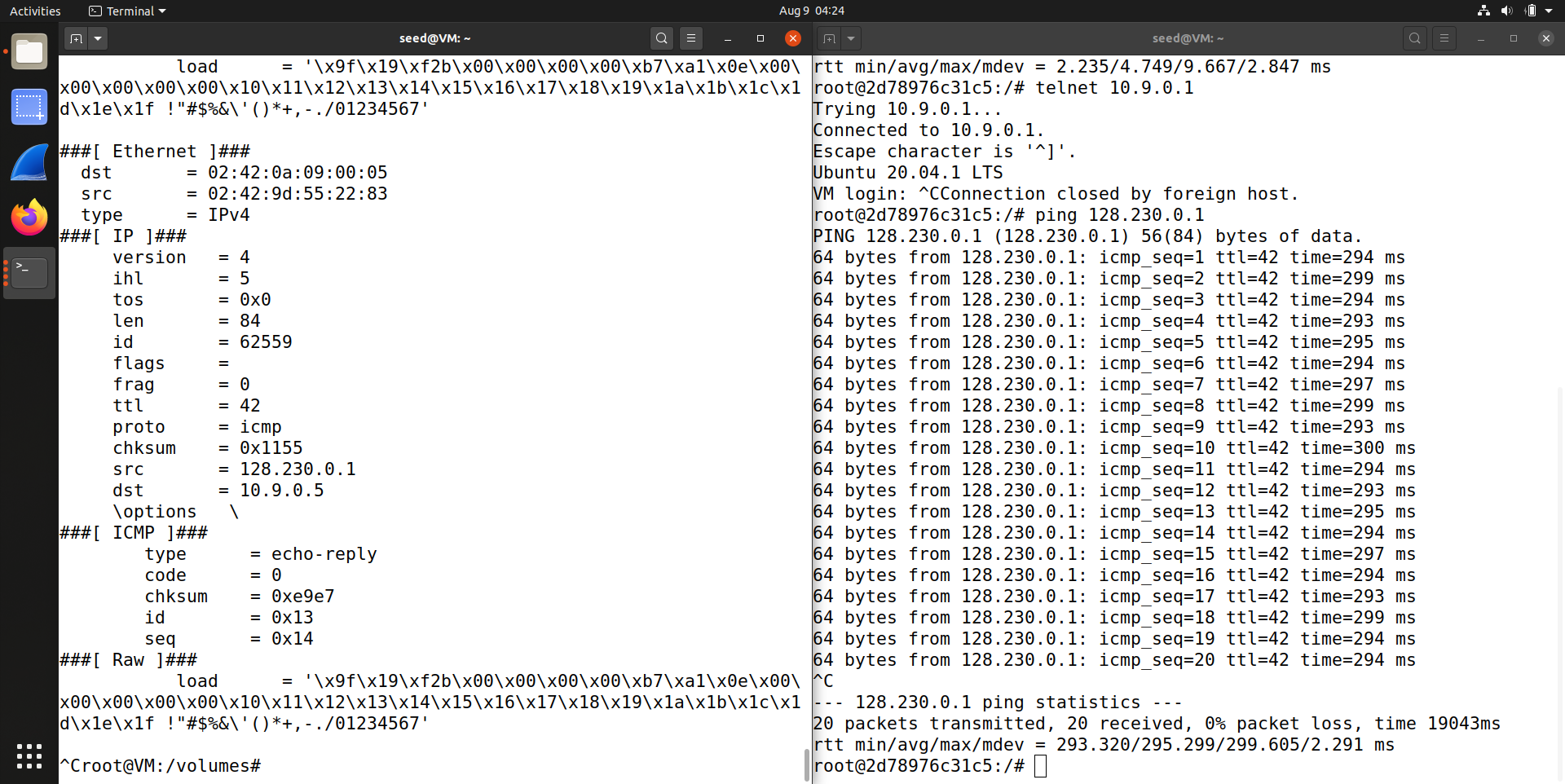
被攻击容器中使用telnet命令向网关发起请求，可在攻击容器中监听到。



修改sniffer.py中的filter，只监听到网络128.230.0.0/16的包。



被攻击容器中使用ping命令发送到128.230.0.1的包，可在攻击容器中监听到。

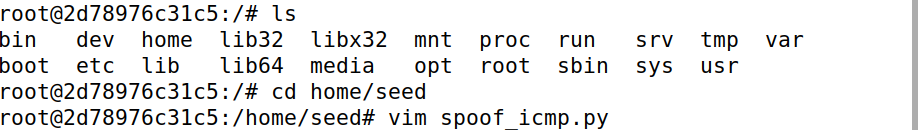


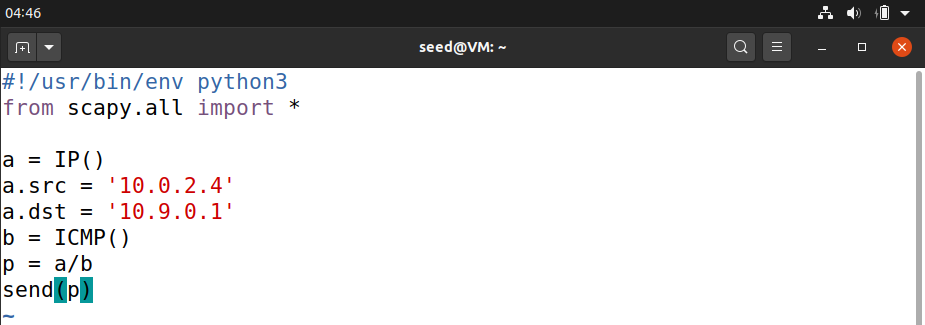
**1.2 Spoofing ICMP Packets**

修改sniffer.py的filter为ICMP。

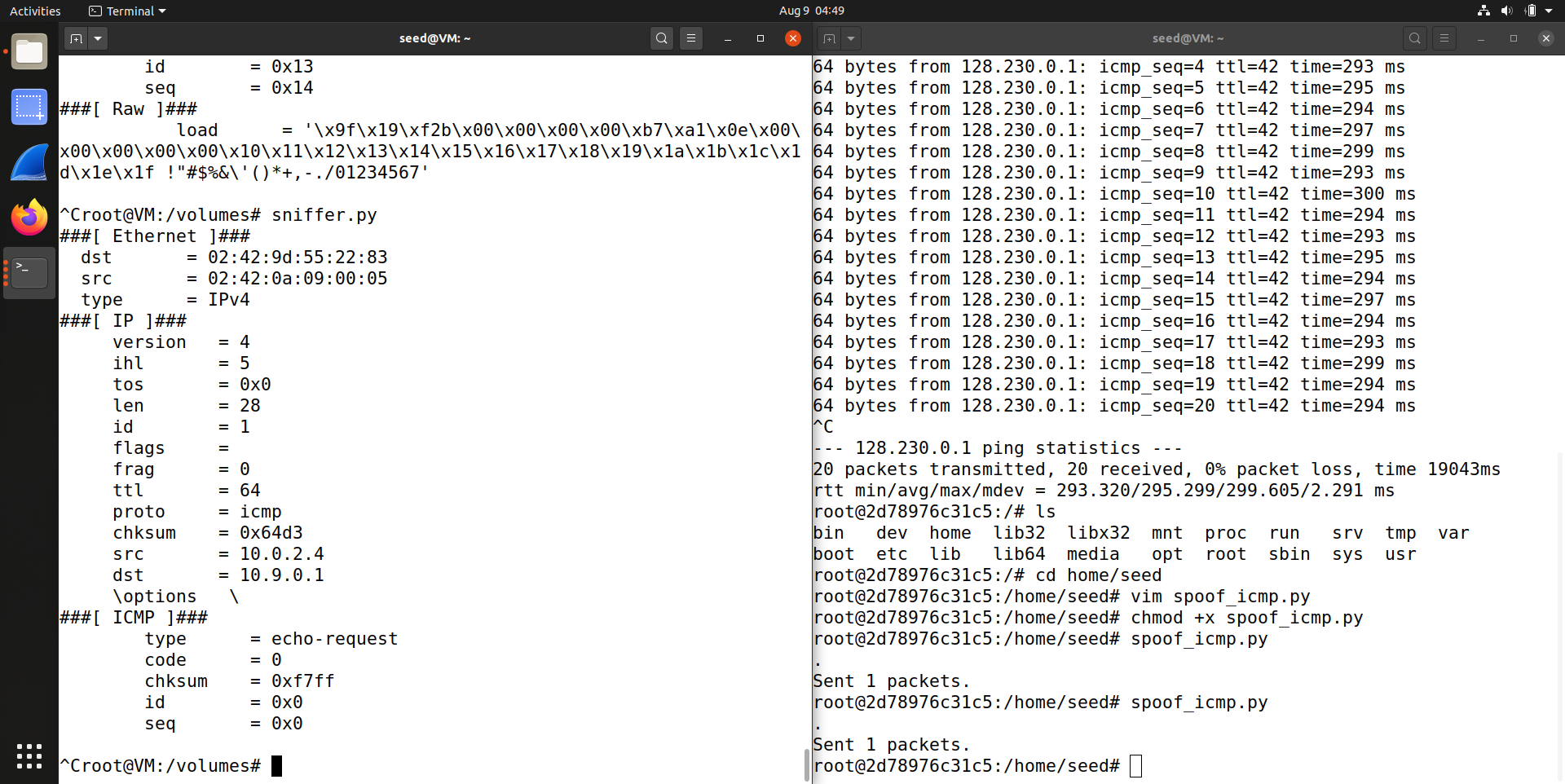


在被监听容器中安装vim编辑器，创建spoof\_icmp.py并编写。



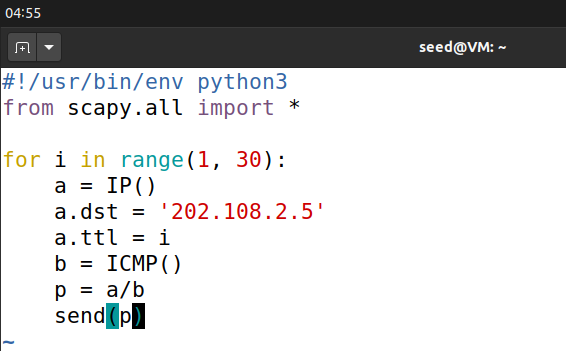


在攻击容器中运行spoof\_icmp.py发送数据包，可在攻击容器中监听到。



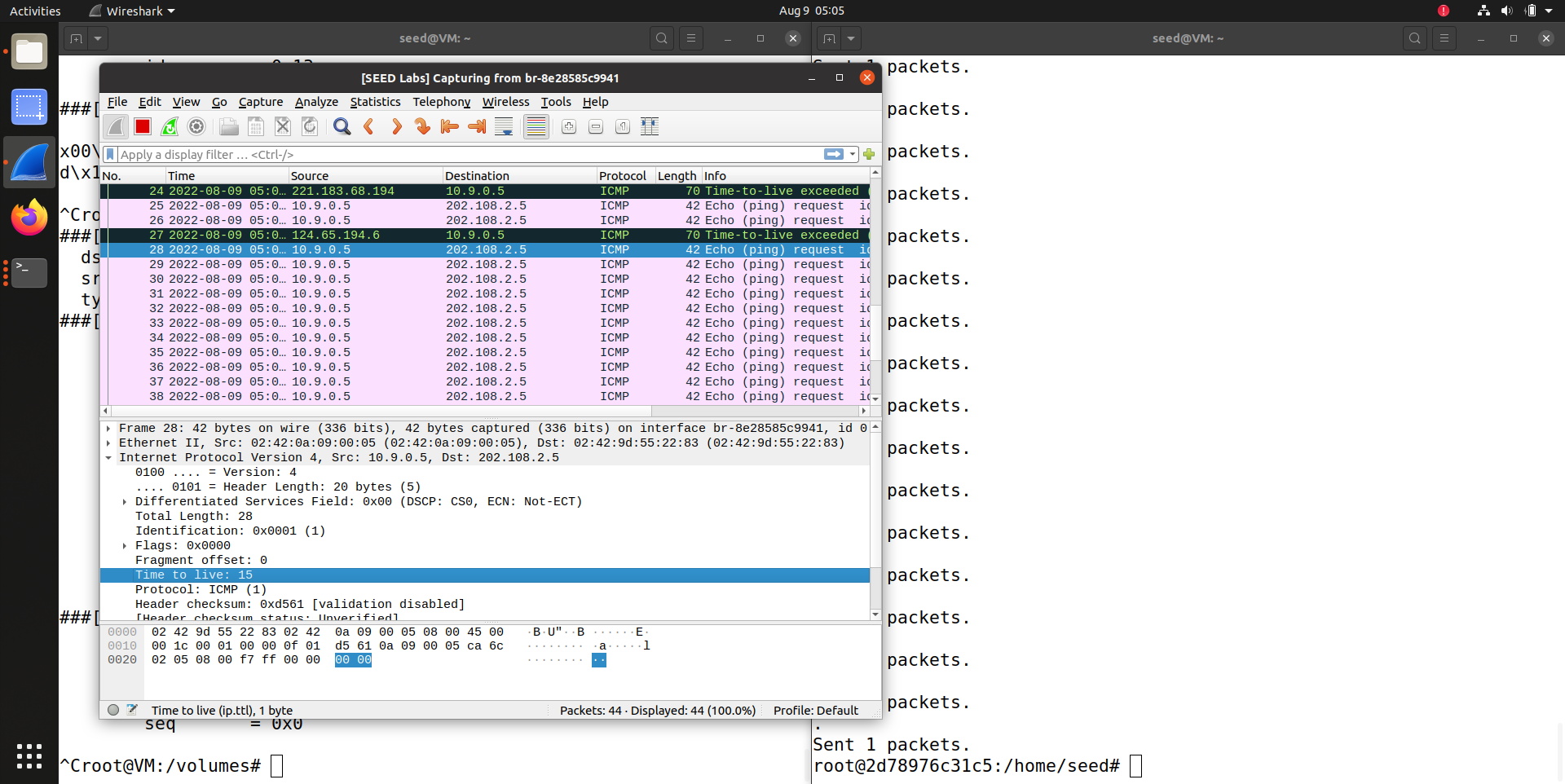
**1.3 Traceroute**

在被攻击容器中创建try\_ttl.py文件并编写。



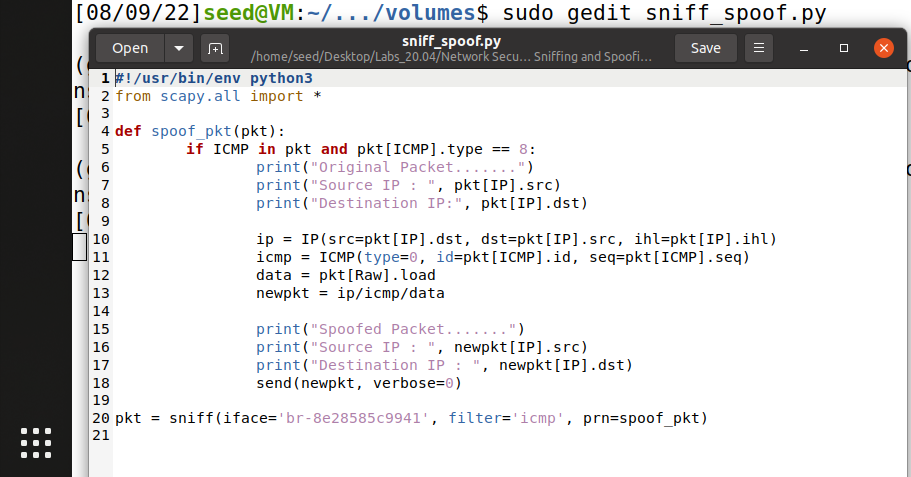
在虚拟机上开启Wireshark，监听网桥br-8e28585c9941。

在被攻击容器中运行try\_ttl.py，在Wireshark中找到第一个没有回复Time-to-live Exceeded的包，发现TTL为15。



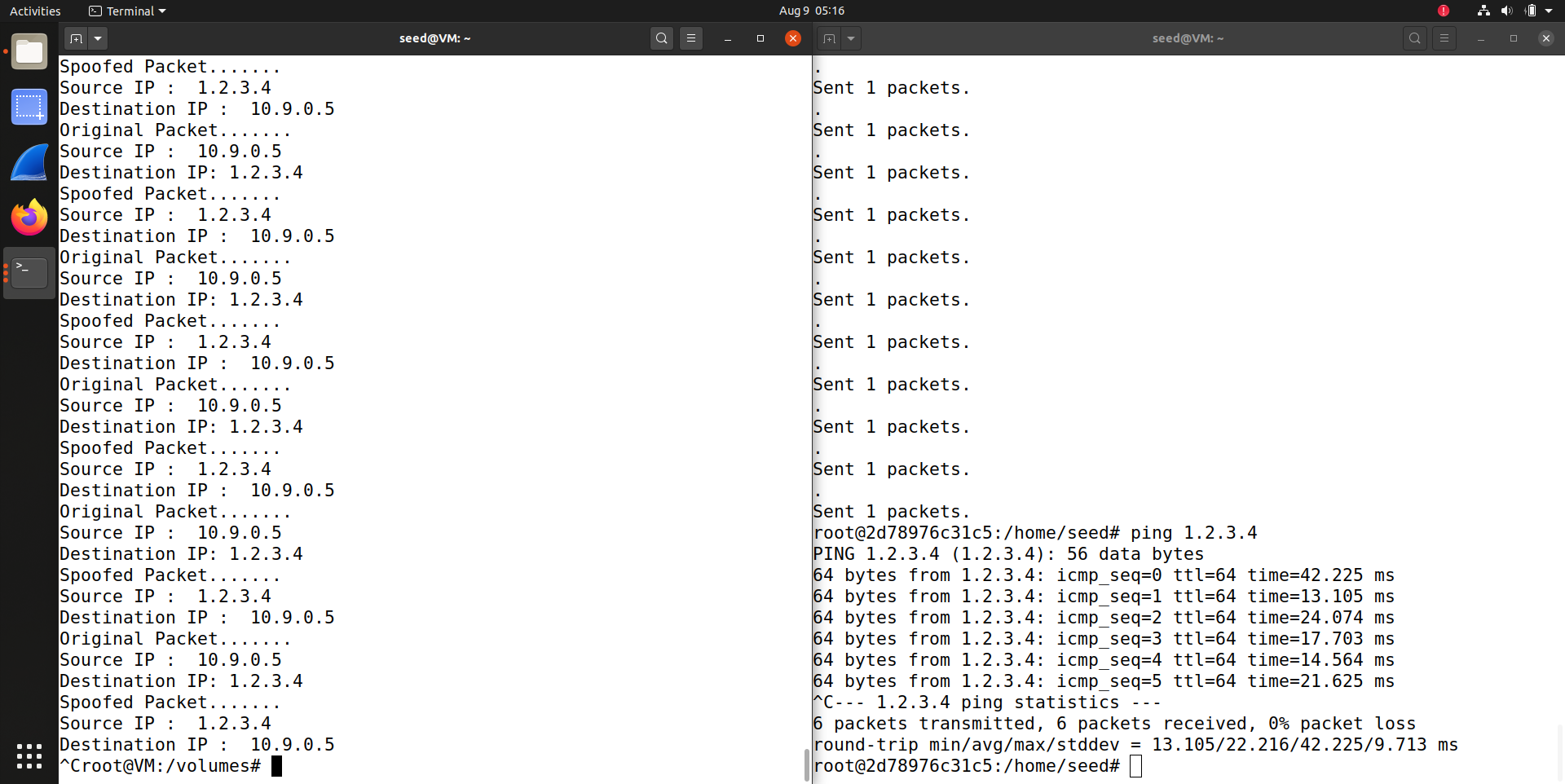
**1.4 Sniffing and-then Spoofing**

在攻击容器的volumes文件夹下创建sniff\_spoof.py并编写。

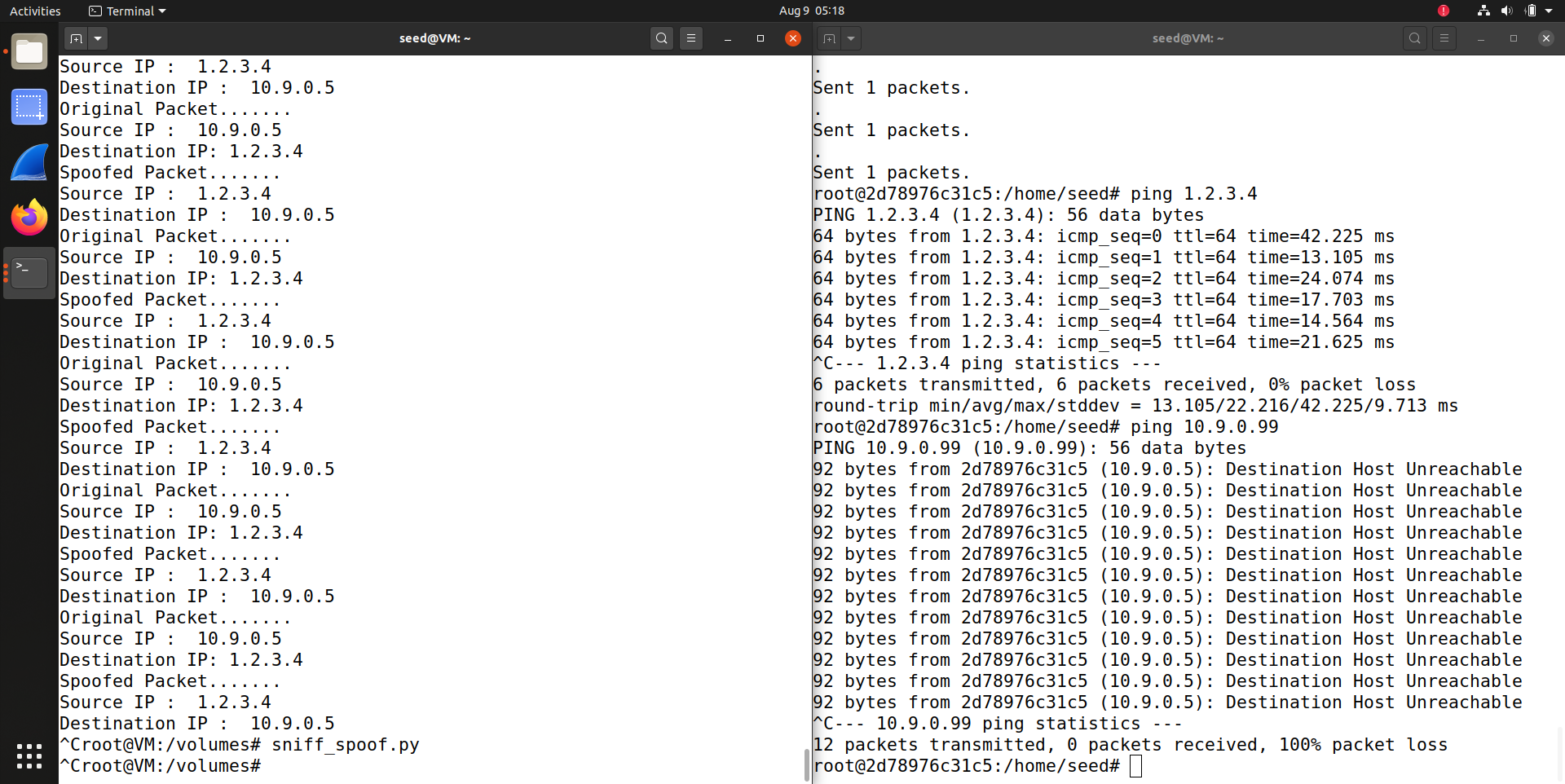


在被攻击容器中依次ping 1.2.3.4（网络上不存在）、10.9.0.99（局域网中不存在）、8.8.8.8（网络上存在）三个IP地址。

ping 1.2.3.4，有回应，因为ICMP包要经过网关，然后被攻击容器嗅探到，之后攻击容器的sniff \_spoof程序返回响应。



ping 10.9.0.99，网络不可达，因为子网内部的包不需要经过网关，但是这个地址本身不存在，所以不会有响应。



ping 8.8.8.8，有回应，因为是存在的公网IP地址，不仅会得到正常响应，还会在被监听到后sniff\_spoof发送响应，所以会有DUP！的标志。

