20307130135李钧实验3

20307130135 李钧

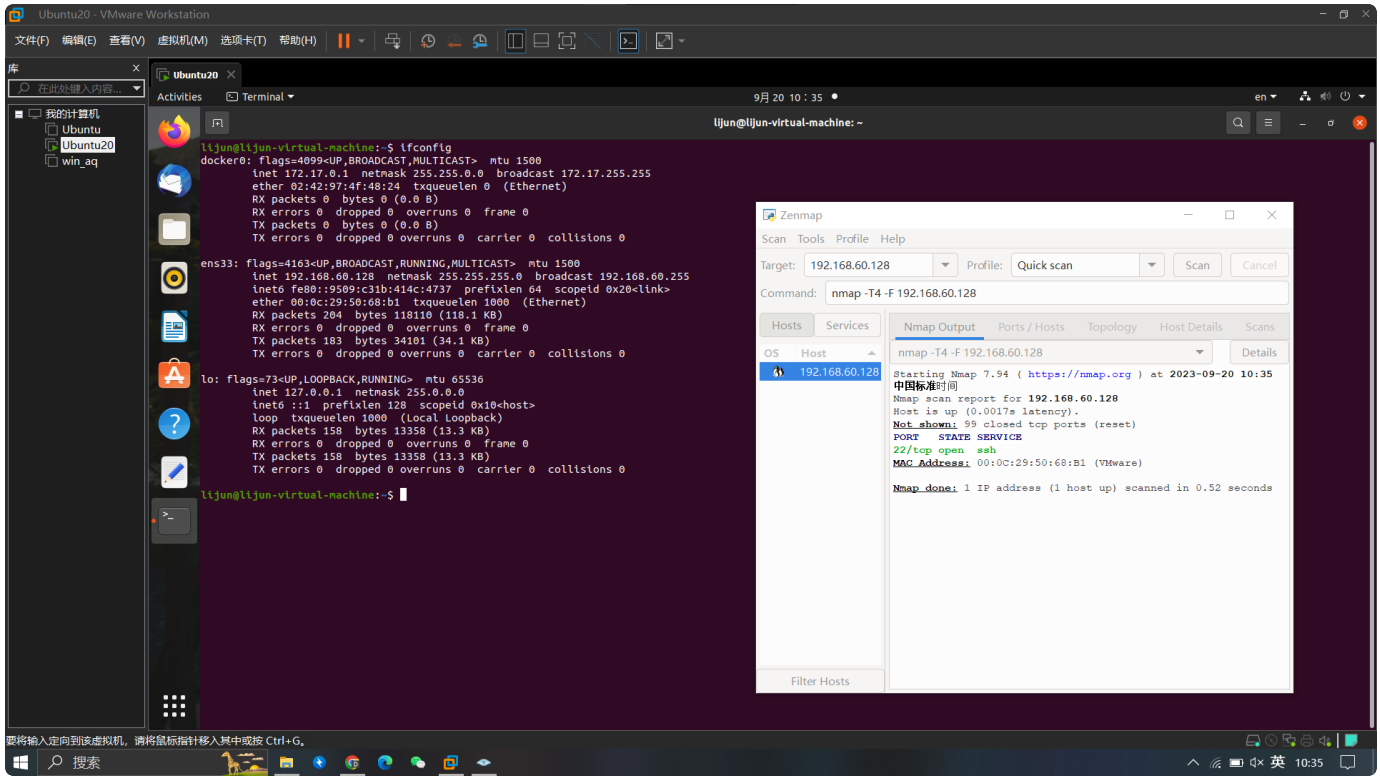
一、实验目的

1. 了解TCP、UDP端口扫描原理
2. 实践Socket编程
3. 实践Iptables防火墙的应用

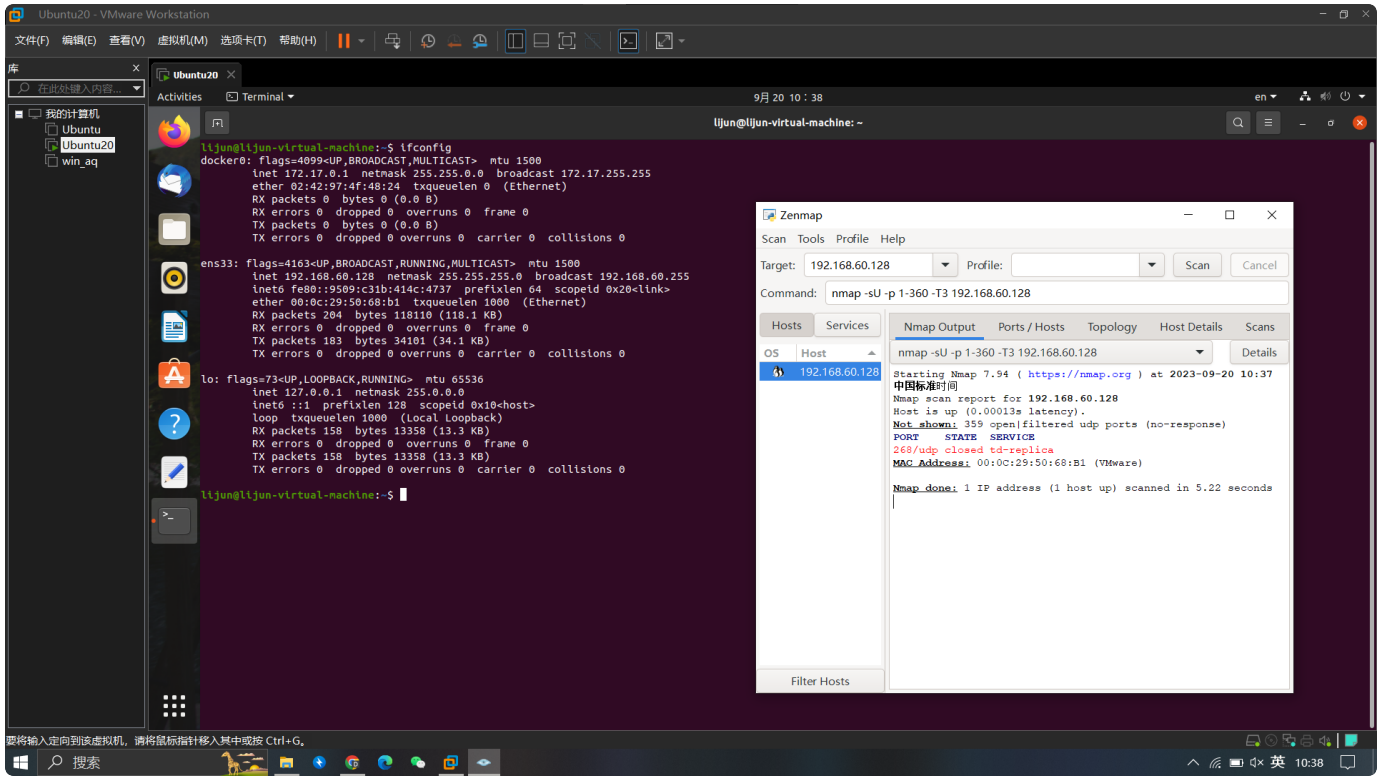
二、实验结果

（一）改变linux目标机防火墙配置之前进行端口扫描

扫描TCP开放端口

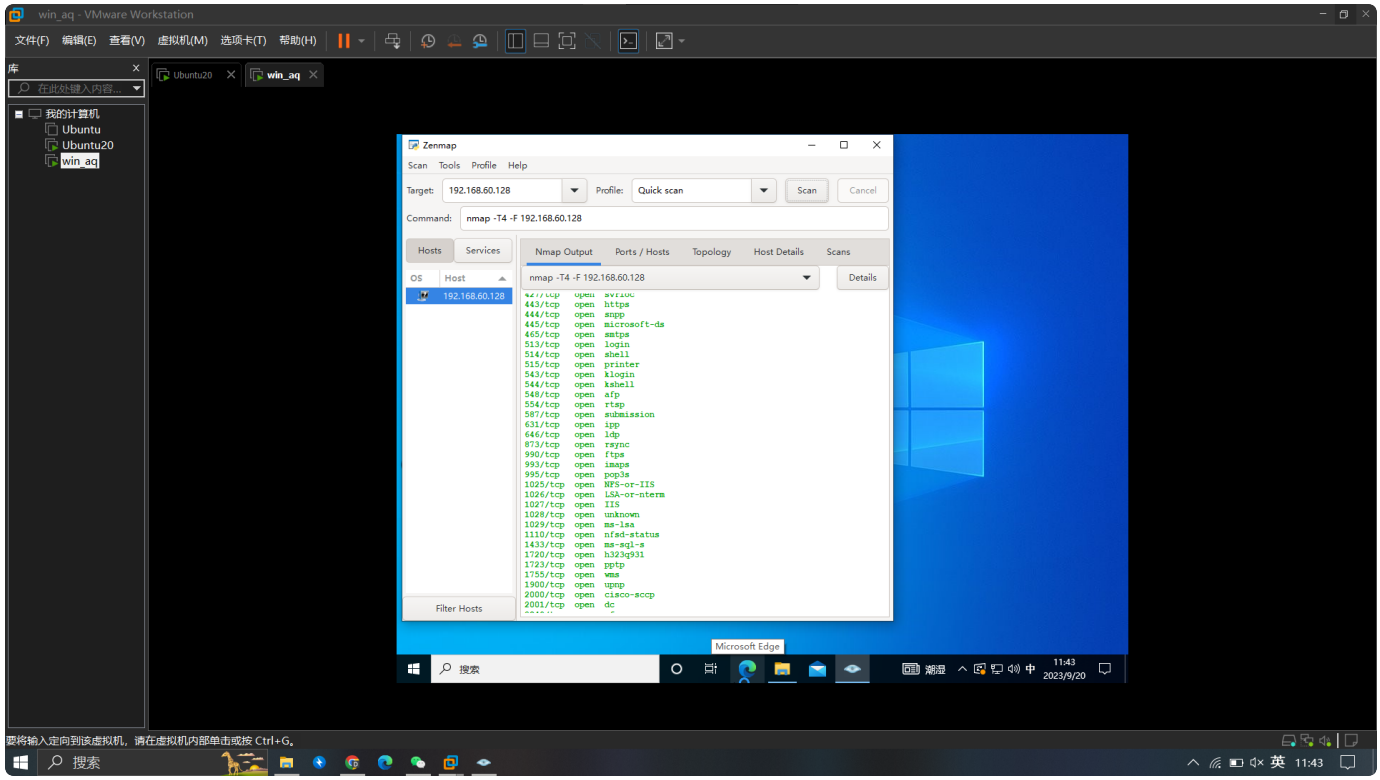


扫描UDP端口

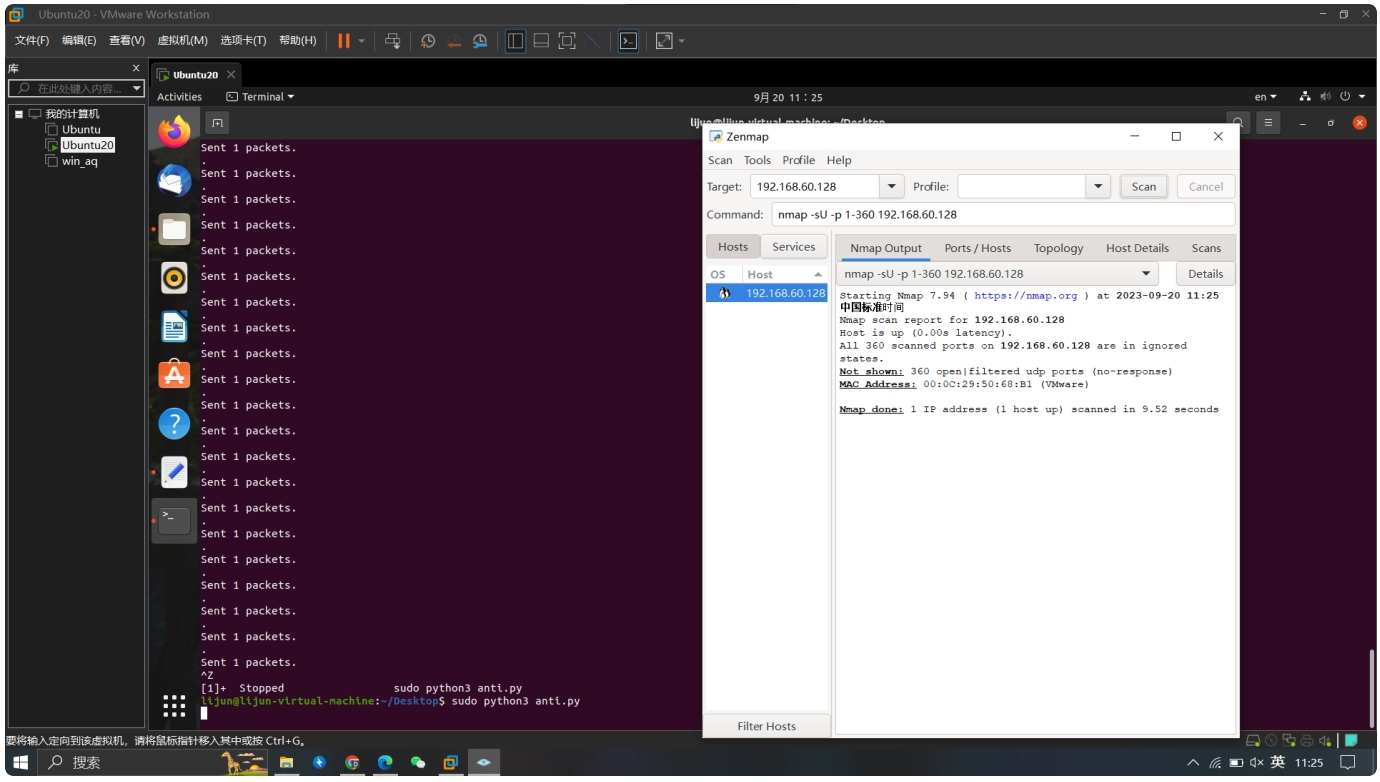


更改linux目标机防火墙之后的端口扫描

TCP抗端口扫描



UDP抗端口扫描

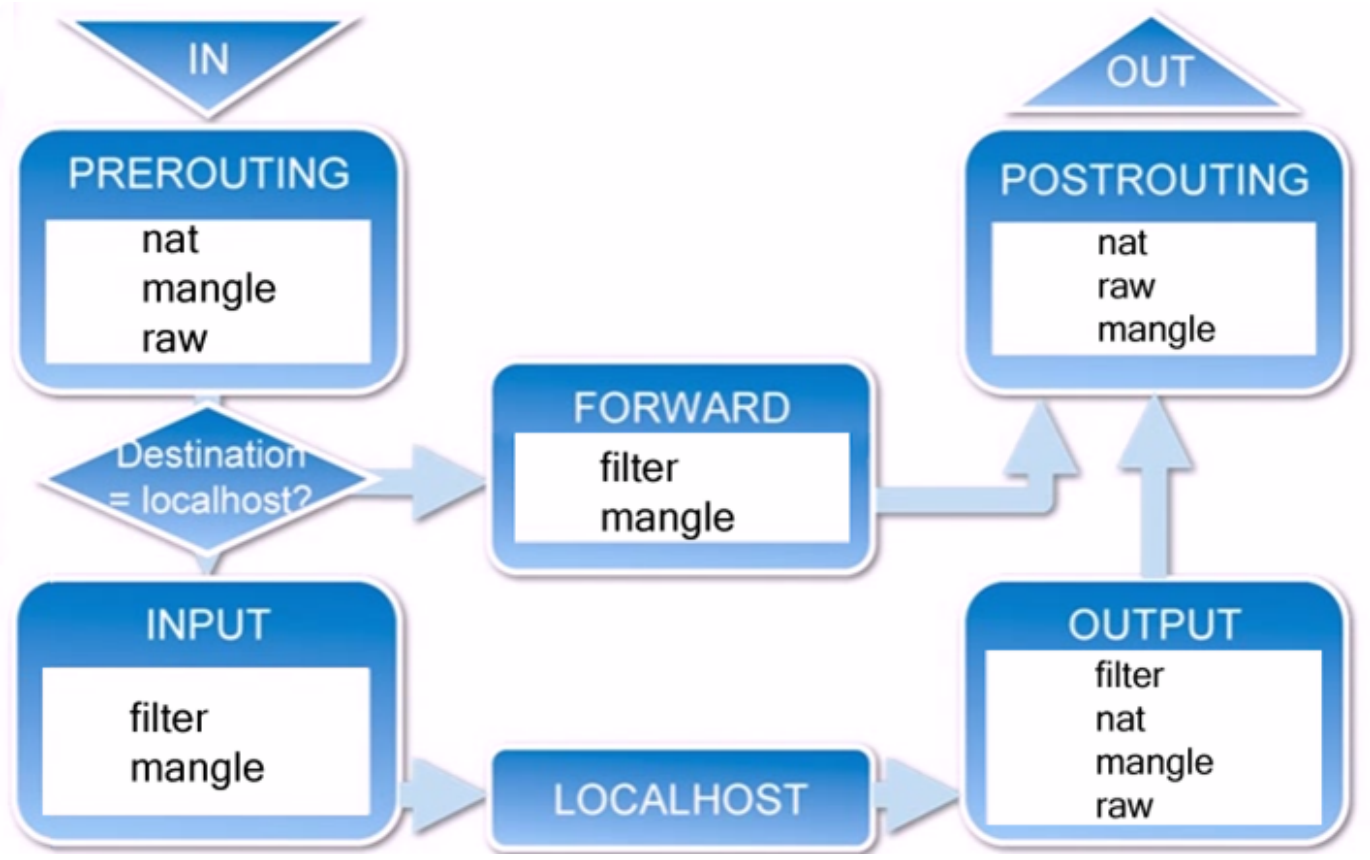


三、实验原理

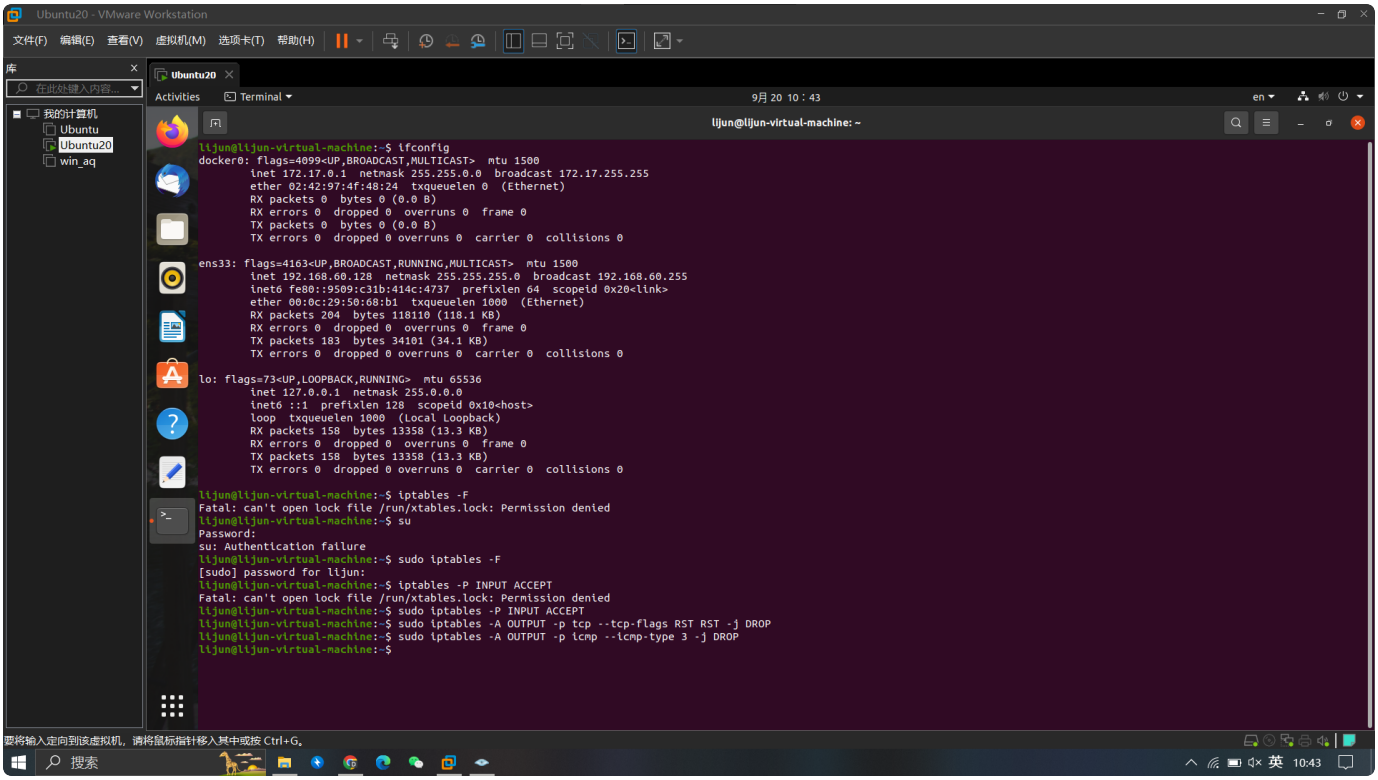
原理

1. 对于TCP端口扫描，目标机收到SYN包，若即刻返回一个SYN_ACK，同时过滤掉系统自动返回的RST数据包，那么扫描方（如nmap）会认为该端口是开放的。
2. 对于UDP端口扫描，目标机收到一个发往关闭端口的数据包，会返回ICMP port unreachable数据包。若过滤掉这些数据包，扫描方将无法判断该UDP端口是否开放。
3. IPtables是一个用于Linux操作系统的防火墙管理工具，它允许我们配置和管理网络数据包的过滤规则。可以使用IPtables来设置规则，以允许或拒绝特定类型的网络流量，从而实现网络安全和流量控制的目的。
    a. 数据包进入网卡时，首先进入PREROUTING链，linux内核会判断数据包的目的IP是否为本地主机
    b. 如果数据包的目的IP是本地主机，那么数据包会沿图向下移动，进入INPUT链中，数据包进入INPUT链中，本地主机的所有线程都会收到它，本地主机的进程也会产生数据包，这些数据包会经过OUTPUT链，然后到达POSTROUTING链从网卡中出去。
    c. 如果数据包的目的IP不是本地主机，则数据包是要转发出去，且linux内核允许转发，数据包就会如图所示向右移动，进入FORWARD链中，然后到达POSTROUTING链从网卡中出去

iptable传输数据包过程图如下：

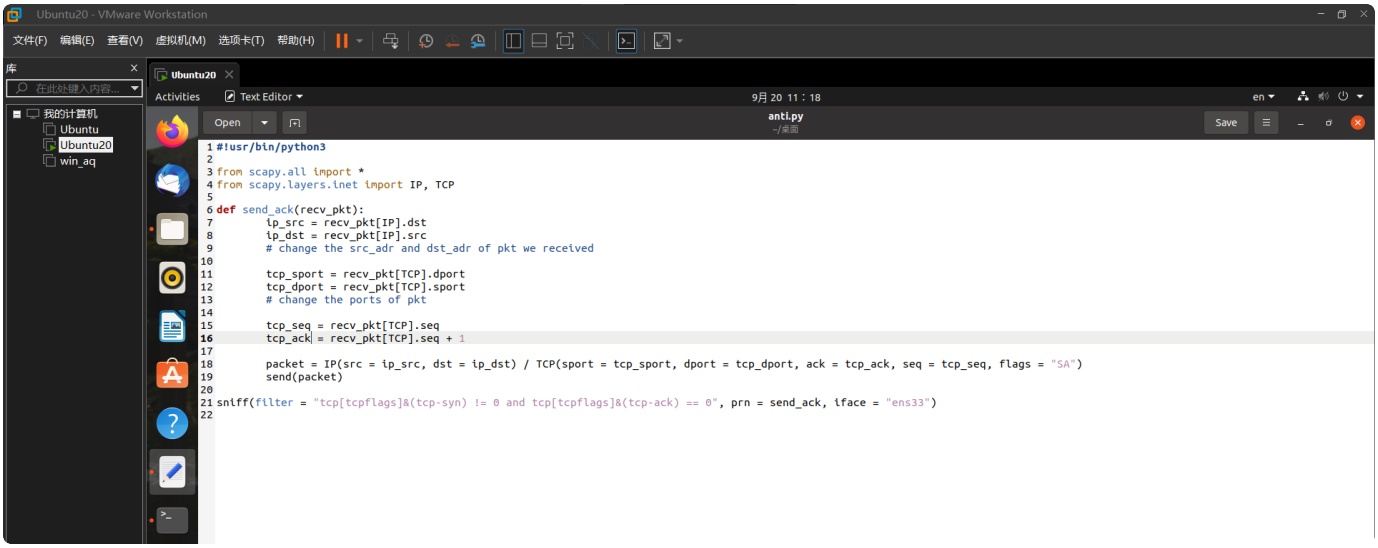


在linux中设置iptable选项如下所示：



实验中用到的python代码

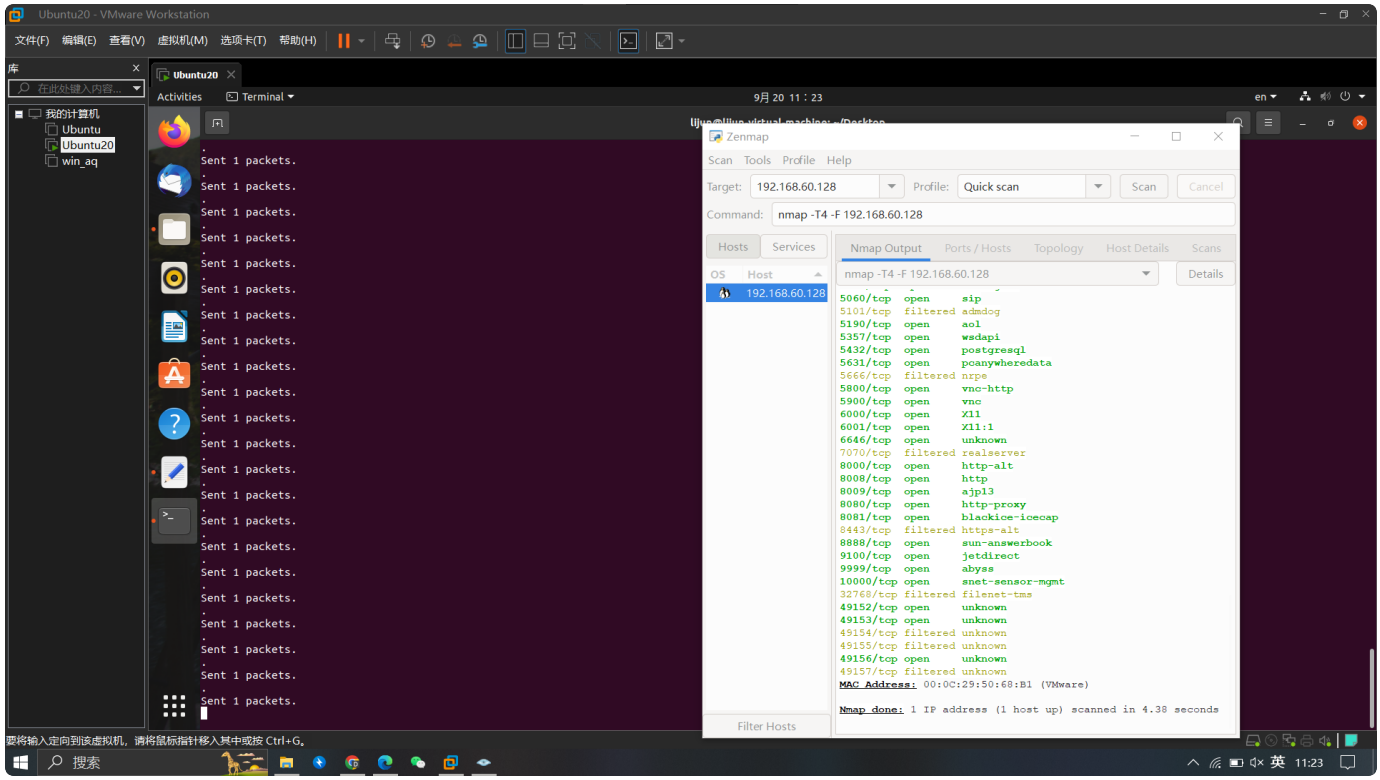
首先将ip地址和端口号 src和dst对换，然后发回SYNACK数据包，flags置为SA，最后探所有tcp flags中syn=1，ack=0的包，收到后执行send_ack函数。在设置iface选项时注意linux机中的iface为ens33.



四、遇到的问题与总结

本次实验主要是进行防火墙设置、端口扫描的实际操作，在上述运行anti.py代码之后，于虚拟机windows中通过nmap扫描linux目标机可以得到所有端口都开放的“假象”，但是在物理机上扫描linux目

标机有部分端口没有打开，如下所示：



究其原因，有如下几种可能：

1. 虚拟网络配置：虚拟化平台（如VMware、VirtualBox、Hyper-V等）通常会为虚拟机创建虚拟网络。虚拟网络的配置和行为可能会影响端口的可见性。确保虚拟网络的配置正确，虚拟机可以与物理网络通信。
2. 虚拟网络模式：虚拟化平台通常支持不同的网络模式，如桥接模式、NAT模式等。不同的网络模式可能会影响虚拟机与物理网络之间的连接。桥接模式通常会使虚拟机在物理网络上表现得像一个独立的设备，而NAT模式则会隐藏虚拟机的IP地址。
3. 主机防火墙：物理机上的防火墙设置也可能影响端口的可见性。确保物理机的防火墙规则不会阻止端口扫描。
4. 虚拟机配置：虚拟机的配置参数（如网络适配器类型、虚拟硬件版本等）可能会影响其网络行为。确保虚拟机的配置与预期的网络行为一致。
5. 端口状态：端口扫描工具通常会检查目标主机上的端口状态。如果端口是关闭的，扫描工具会报告为"closed"。但如果端口是由防火墙过滤的，扫描工具可能会报告为"filtered"，这表明端口存在但不可访问。