

* 1. **-网络管理相关的命令**

**1.1-网络协议的应用和TCP连接**

**了解常见的网络相关协议**

**TCP三次握手和四次挥手**

**1.2-RHEL7.网络管理**

**网络相关的调试命令**

**实战tcpdump和tshark抓包**

**1.1-1 网络的相关网络协议**

**ARP(Address Resolution Protocol)地址解析协议 将IP解析成MAC地址**

**DNS 域名解析协议**

**SNMP(Simple Network Management Protocol)网络管理协议  
DHCP(Dynamic Host Configuration Protocol)动态主机配置协议  
它是在TCP/IP网络上使客户机获得配置信息的协议**

**FTP(File Transfer Protocol)文件传输协议  
它是一个标准协议，是在计算机和网络之间交换文件的最简单的方法。**

**HTTP(Hypertext Transfer Protocol )超文本传输协议**

**HTTPS(Secure Hypertext Transfer Protocol)安全超文本传输协议  
它是由Netscape开发并内置于其浏览器中，用于对数据进行压缩和解压操作.**

**ICMP(Internet Control Message Protocol)Internet控制信息协议**

**SMTP(Simple Mail Transfer Protocol)简单邮件传送协议**

**TELNET Protocol虚拟终端协议**

**TFTP(Trivial File Transfer Protocol)小文件传输协议**

**UDP(User Datagram Protocol)用户数据报协议**

**它是定义用来在互连网络环境中提供包交换的计算机通信的协议，**

**1.1-2 TCP/IP协议簇**

**TCP/IP协议是一个协议的集合。包括很多协议**

**HTTP超文本传输协议  
FTP文件传输协议     
TFTP简单文件传输协议**

**Telnet远程登录**

**SNMP简单网络管理协议**

**TCP Transmission Control Protocol，传输控制协议）是面向连接的协议，**

**UDP UDP（User Data Protocol，用户数据报协议）非连接的协议**

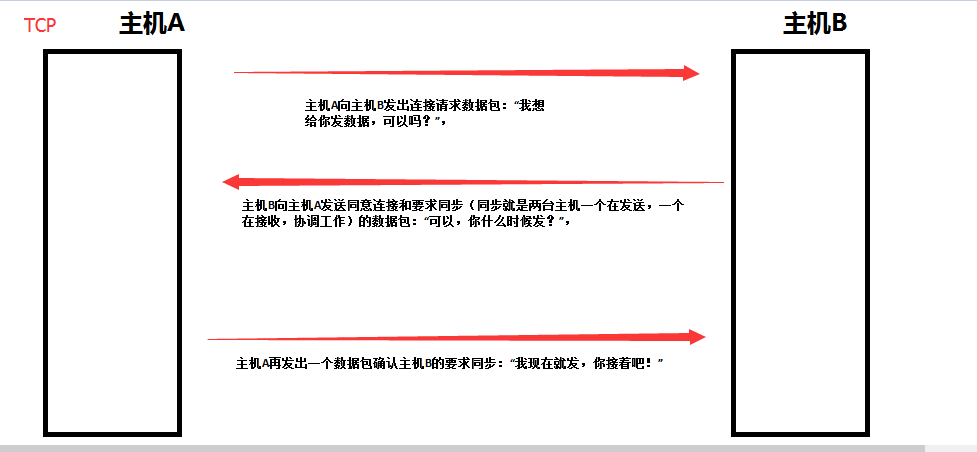
**IP Internet协议       
ICMP Internet控制信息协议  
ARP 地址解析协议  
RARP反向地址解析协议RARP**

**重点--TCP和UDP区别**

**TCP（Transmission Control Protocol，传输控制协议）是面向连接的协议，**

**在收发数据前，必须和对方建立可靠的连接。**

**一个TCP连接必须要经过三次“对话”才能建立起来，其中的过程非常复杂，**



1. **主机A向主机B发出连接请求数据包：“我想给你发数据，可以吗？”;**
2. **主机B向主机A发送同意连接和要求同步的数据包：“可以，你什么时候发？**

**同步就是两台主机一个在发送，一个在接收，协调工作**

1. **3.主机A再发出一个数据包确认主机B的要求同步：“我现在就发，你接着吧！;**

**总结:**

**三次“对话”的目的是使数据包的发送和接收同步，经过三次“对话”之后，主机A才向主机B正式发送数据。**

**UDP（User Data Protocol，用户数据报协议）**

**UDP是一个非连接的协议，传输数据之前源端和终端不建立连接.**

**UDP信息包的标题很短，只有8个字节，相对于TCP的20个字节信息包的额外开销很小。  
吞吐量不受拥挤控制算法的调节，只受应用软件生成数据的速率、传输带宽、源端和终端主机性能的限制。  
UDP使用尽最大努力交付，即不保证可靠交付，因此主机不需要维持复杂的链接状态表;**

**例如:  
我们经常使用“ping”命令来测试两台主机之间TCP/IP通信是否正常，其实“ping”命令的原理就是向对方主机发送UDP数据包，然后对方主机确认收到数据包，如果数据包是否到达的消息及时反馈回来，那么网络就是通的。**

**总结:**

**UDP传输的速度很快,但是不会保障包是否被成功接收**

**总结--TCP与UDP的区别：  
1.基于连接与无连接；  
2.对系统资源的要求（TCP较多，UDP少）；  
3.UDP程序结构较简单；  
4.TCP保证数据正确性，UDP可能丢包，TCP保证数据顺序，UDP不保证。**

**1.1-3-TCP三次握手建立连接的过程**

**Seq:序列号 根据序列号标识某个操作**

**A🡪seq=1**

**b🡪ack+1**

**ACK：确认标志--确认编号**

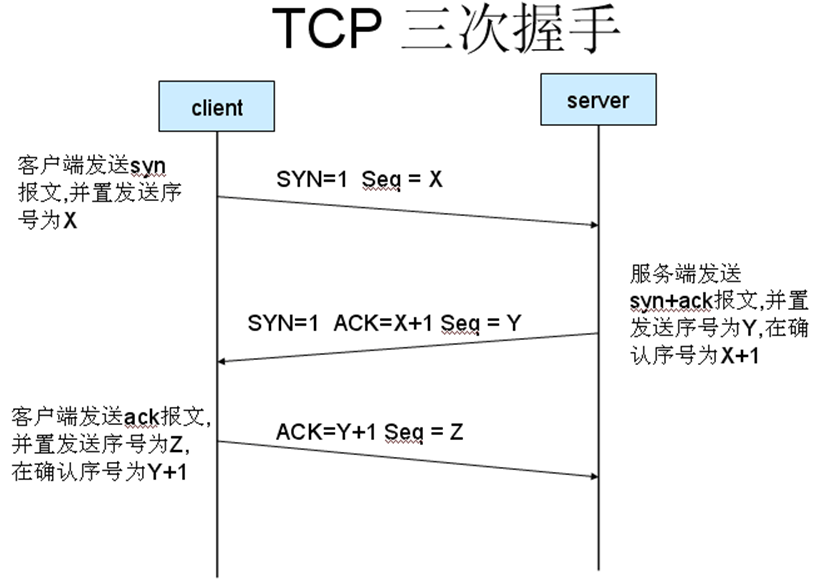
**注意:在响应ACK的形式 如果确认则ack=对方主机的seq+1**

**SYN：同步标志**

**同步序列编号该标志仅在三次握手建立TCP连接时有效。它提示TCP连接的服务端检查序列编号，该序列编号为TCP连接初始端（一般是客户端）的初始序列编号。**

**FIN：结束标志**

**带有该标志置位的数据包用来结束一个TCP回话，但对应端口仍处于开放状态，准备接收后续数据。**

****

**TCP三次握手过程**

**1 主机A通过向主机B 发送一个含有同步(SYN)序列号(seq)的标志位的数据段给主机B ,向主机B 请求建立连接.**

**通过这个数据段,主机A告诉主机B 我想要和你通信;**

**2 主机B 收到主机A的请求后,用一个带有确认应答(ACK)和同步序列号(SYN)标志位的数据段响应主机A.**

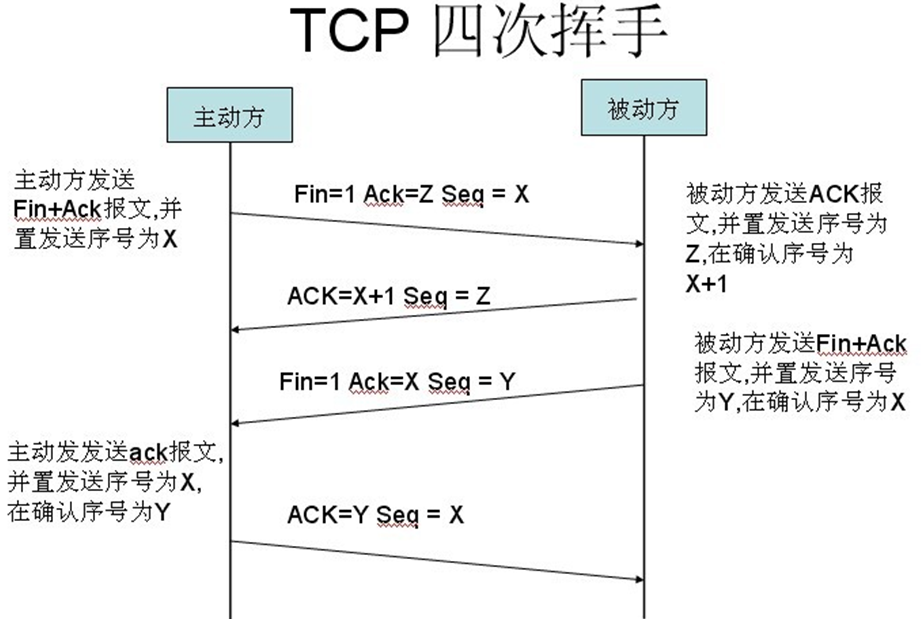
**也告诉主机A:我已经收到你的请求了,你可以传输数据了;**

**3 主机A收到这个数据段后,再发送一个确认应答,确认已收到主机B 的数据段.**

**"我已收到回复,我现在要开始传输实际数据了**

**这样3次握手就完成了,主机A和主机B 就可以传输数据了.**

**TCP四次挥手断开连接过程**

****

**（1）客户端A发送一个FIN，用来关闭客户A到服务器B的数据传送。**

**（2）服务器B收到这个FIN，它发回一个ACK，确认序号为收到的序号加1。和SYN一样，一个FIN将占用一个序号。**

**（3）服务器B关闭与客户端A的连接，发送一个FIN给客户端A。**

**（4）客户端A发回ACK报文确认，并将确认序号设置为收到序号加1。**

**【问题1】为什么连接的时候是三次握手，关闭的时候却是四次握手？**

**答：因为当Server端收到Client端的SYN连接请求报文后，可以直接发送SYN+ACK报文。**

**其中ACK报文是用来应答的，SYN报文是用来同步的。**

**但是关闭连接时，当Server端收到FIN报文时，很可能并不会立即关闭SOCKET，所以只能先回复一个ACK报文，告诉Client端，"你发的FIN报文我收到了"。只有等到我Server端所有的报文都发送完了，我才能发送FIN报文，因此不能一起发送。故需要四步握手。**

**Linux下端口号的分配**

**TCP和UDP 采用16位的端口号来识别应用程序**

**TCP/IP的临时分配1024-5000之间的端口号**

**大于5000以上的是为其他服务保留的**

**TCP 端口分配**

**21 ftp 文件传输服务**

**22 ssh 安全远程连接服务**

**23 telnet 远程连接服务**

**25 smtp 电子邮件服务**

**53 DNS 域名解析服务**

**80 http web服务**

**443 https 安全web服务**

**UDP 端口分配**

**69 tftp 简单文件传输协议**

**123 ntp 时间同步服务**

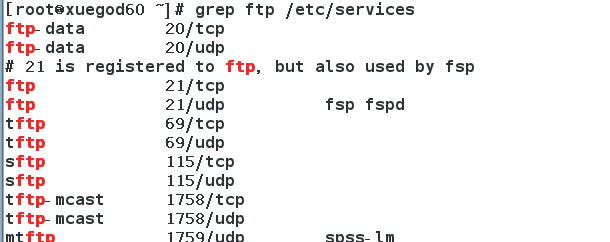
**161 snmp 简单网络管理**

**如果你不知道服务对应的端口号**

**编辑 /etc/services 文件 包含所有的端口号**

**例如:查看ftp的端口号**

**有的服务是UDP和TCP端口都会监听的**



**查看端口的监听状态**

**netstat**

**常用的格式 :**

**tcp端口:　anpt　　　udp端口: anpu 全部 :anput**

**-a, --all**

**-n, --numeric don't resolve names**

**-p, --programs**

**-t 显示tcp连接**

**-u 显示udp连接**



**网络相关的调试命令：**

**#查看网卡物理连接是否正常**

**[root@xuegod60 ~]# mii-tool eno16777736**

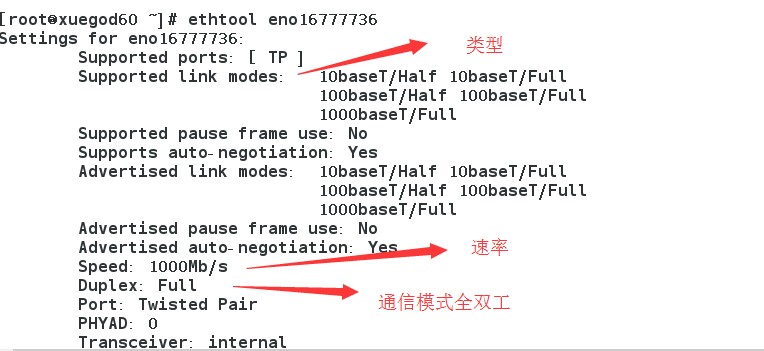




**查看网卡硬件相关信息：**

**ethtool 是用于查询及设置网卡参数的命令。**

**[root@xuegod60 ~]# ethtool eno16777736**

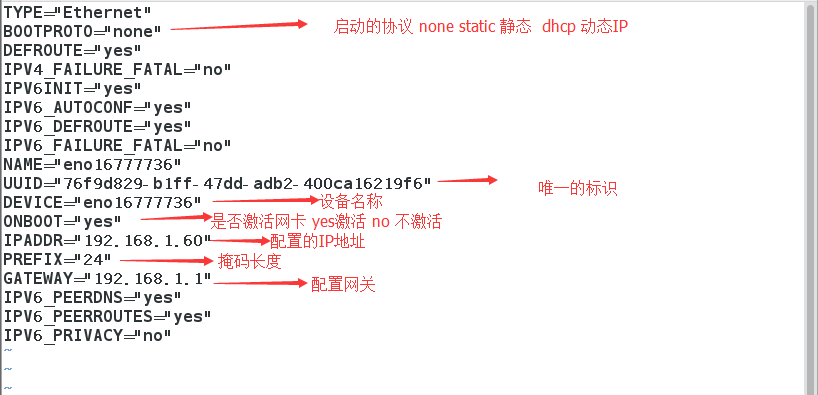


**配置网络和IP地址：**

**方法一：[root@xuegod60 ~]# nmtui-edit eno16777736**

**方法二：修改网卡配置文件**

**[root@xuegod60 ~]# vim /etc/sysconfig/network-scripts/ifcfg-eno16777736**



**说明：在RHEL7中增强了NetworkManager服务功能，弱化了network的功能**

**[root@xuegod60 ~]# systemctl status NetworkManager**

**● NetworkManager.service - Network Manager**

**Loaded: loaded (/usr/lib/systemd/system/NetworkManager.service; enabled; vendor preset: enabled)**

**Active: active (running) since Mon 2016-03-07 20:19:42 CST; 1h 36min ago**

**开启NetworkManager**

**[root@xuegod60 ~]# systemctl restart NetworkManager #开启服务**

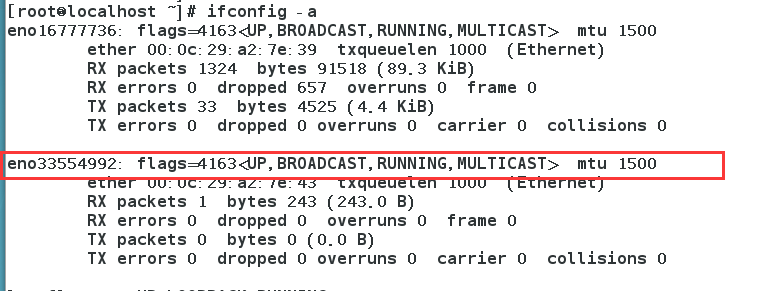
**[root@xuegod60 ~]# systemctl enable NetworkManager #设置服务开机自动启动**

**#此服务不开启，则无法通过nmtui工具配置网络**

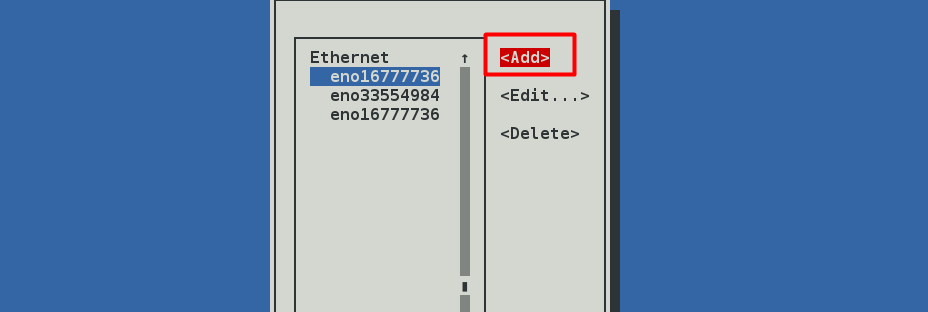
**给新添加的网卡添加IP地址和对应的配置文件。**

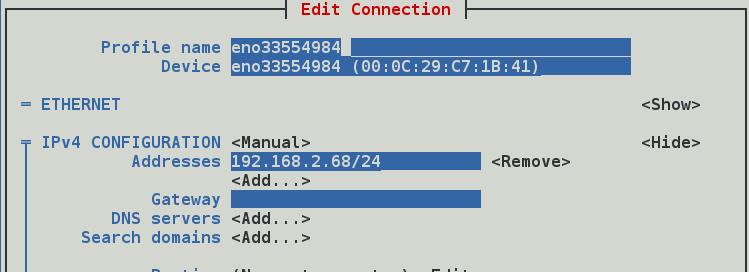
**添加网卡之后查看网卡的名称**

**[root@xuegod60 ~]# ifconfig -a**



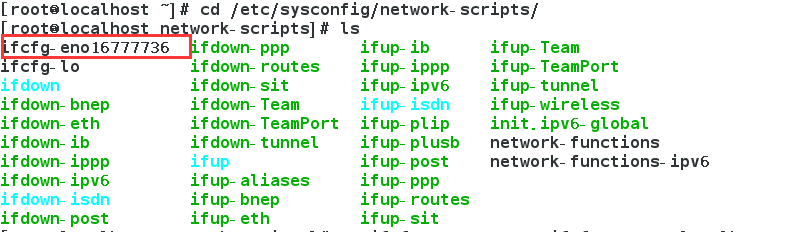
**首先，添加设备**



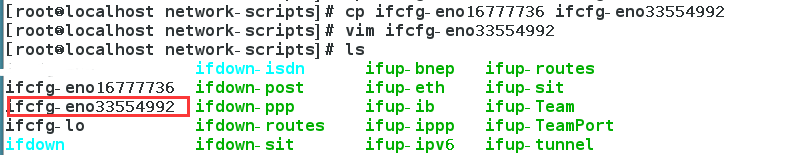


方法2:

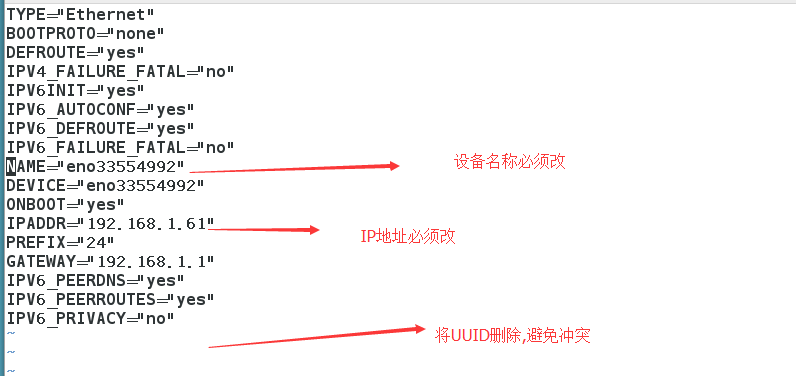
**找到我的第一块网卡配置文件**



**复制生成新的网卡配置文件**



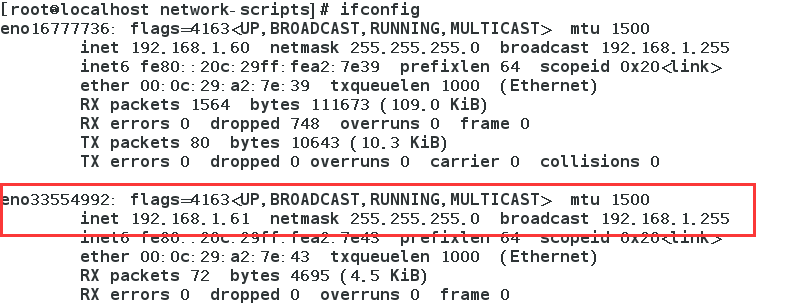
**编辑新网卡的文件**



**重启网络服务**



**验证:**



**修改主机名配置文件**

**hostname 临时配置**

**[root@xuegod60 ~]# vim /etc/hostname**

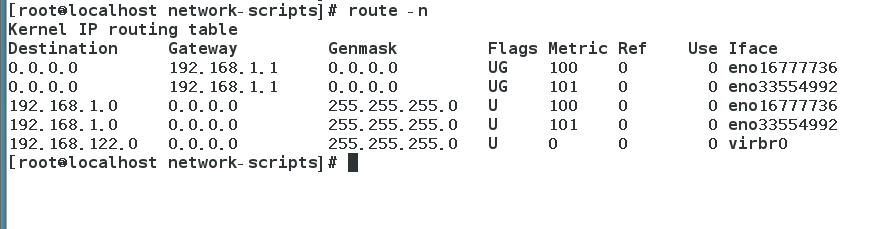
**xuegod60.cn**

**如果是RHEL6操作系统，通过下面的文件修改主机名**

**vim /etc/sysconfig/network**

**查看路由信息： 查看默认网关**

**[root@xuegod60 ~]# route -n**



**Destintaion 目标网关或主机**

**Gateway 网关的地址 \*　表示没有网关**

**Ｇenmask 路由的掩码**

**Flags 路由的标志**

**Metric 路由的度量值**

**Ref 依赖本路由的其他路由数目**

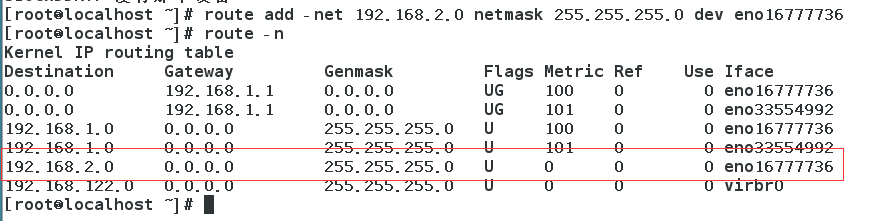
**Use 路由条目被使用的数目**

**Iface 路由所发送包的目的网络**

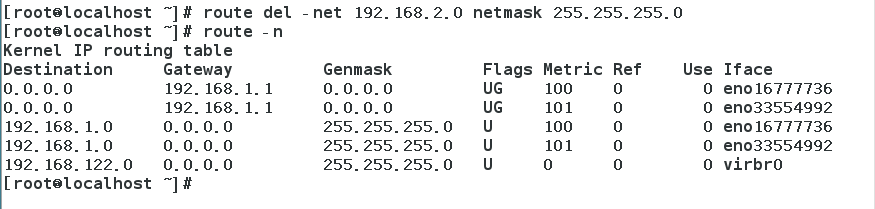
**添加/删除路由条目：**

**增加 (add) 与删除 (del) 路由的相关参数：  
  -net    ：网络ID；  
  -host  ：为连接到主机的路由；  
  netmask ：与网域有关，可以设定 netmask 决定网域的大小；  
   dev    ：如果只是要指定由那一块网路卡连线出去，则使用这个设定，后面接 eth0 等**

**[root@xuegod60 ~]# route add -net 192.168.2.0 netmask 255.255.255.0 dev eno33554984**



**[root@xuegod60 ~]# route del -net 192.168.2.0 netmask 255.255.255.0**



**tcpdump命令：**

**三次握手过程分析**

**tcpdump port 22 -c 3 -n -S**

**# port 端口号**

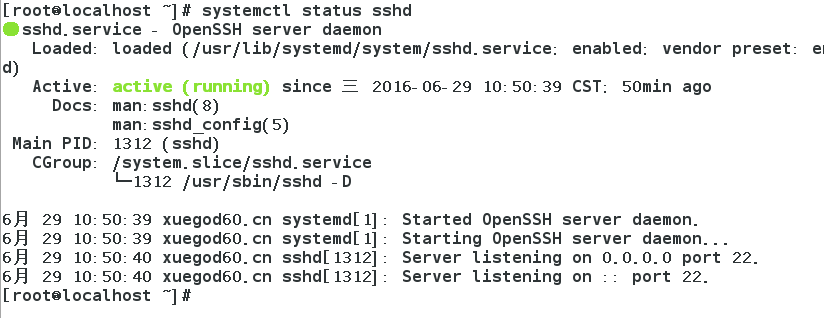
**-c 抓几个包**

**-n 不解析端口号为协议名**

**-S Print absolute, rather than relative, TCP sequence numbers.**

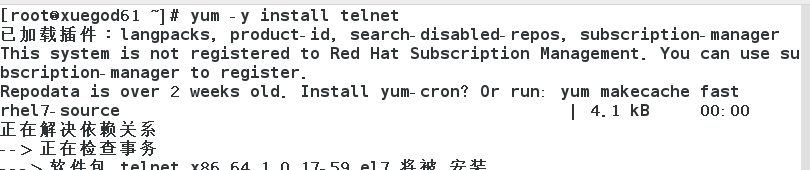
**-i 指定网卡**

**在主机server上开启SSHD服务，**



**在客户端**

**安装telnet工具包yum -y install telnet**

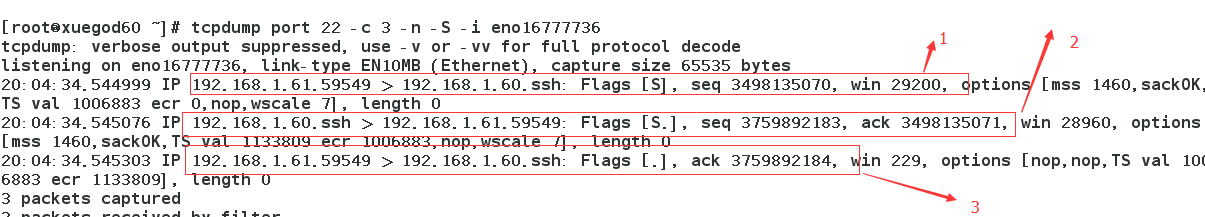


**在服务端**

**tcpdump port 22 -c 3 -n 不解析IP地址为域名 -S 详细信息 -i eno16777736**

**客户端**

**telnet 192.168.1.60 22**



**抓包命令-wireshark**

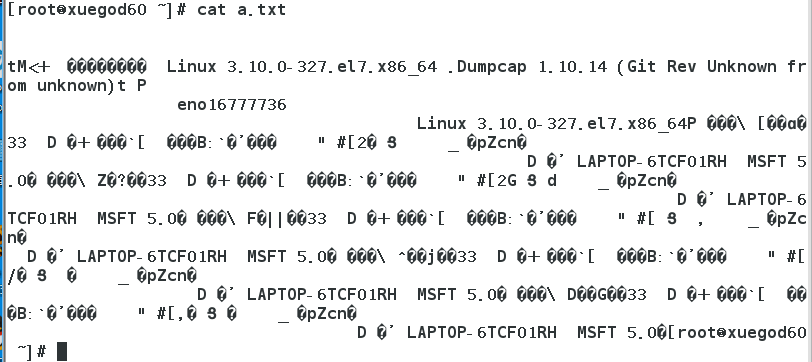
**yum -y install wireshark**

**创建一个保存抓包信息的文件**

**touch a.txt**

**执行抓包命令**

**[root@xuegod60 ~]# tshark -w a.txt -i eno16777736**



**[root@xuegod60 ~]# tshark -r a.txt**

