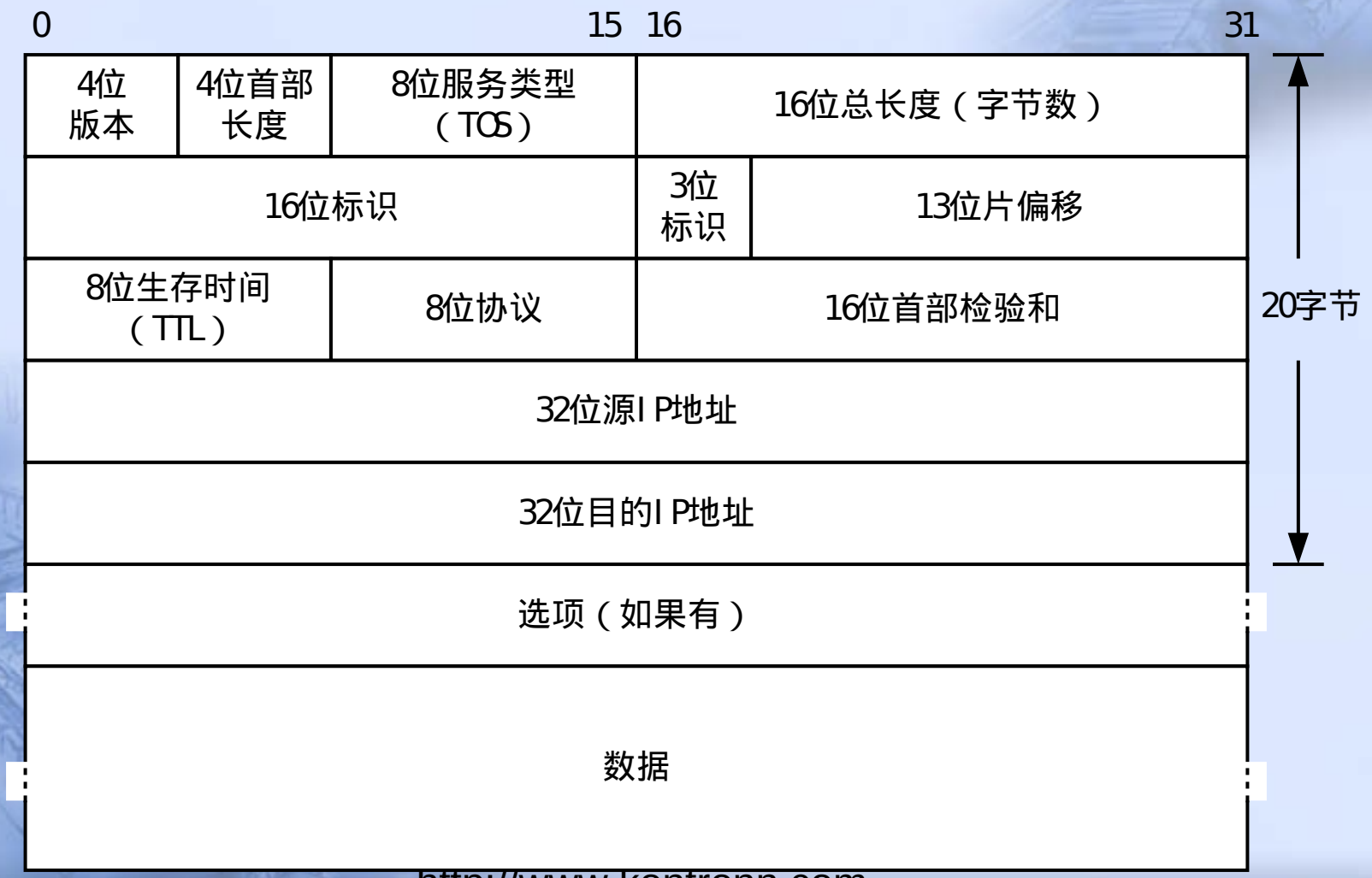
网络编程

IPV4 协议分析

IPv4 包头结构


0
15
16
31
4位
版本
4位首部
长度
8位服务类型
（TOS）
16位总长度（字节数）
16位标识
3位
标识
13位片偏移
8位生存时间
（TTL）
8位协议
16位首部检验和
20字节
32位源IP地址
32位目的IP地址
选项（如果有）
数据

IPv4 包头结构

- 版本 — IP 头中前四位标识了 IP 的操作版本，比如版本 4 或版本 6
- Internet 头长度 — 头中下面 4 位包括头长度，以 32 位为单位表示
- 服务类型
- 总长度 (Total Length)
- 标识 (Identifier) — 每个 IP 报文被赋予一个惟一的 16 位标识，用于标识数据报的分段
- 分段标志 (Fragmentation Flag) — 下一个域包括 3 个 1 位标志，标识报文是否允许被分段和是否使用了这些域

IPv4 包头结构

- 分段偏移 (Fragment Offest) — 8 位的域指出分段报文相对于整个报文开始处的偏移。这个值以 64 位为单位递增
- 生存时间 (TTL) — IP 报文不允许在广域网中永久漫游。它必须限制在一定的 TTL 内
- 协议 — 8 位域指示 IP 头之后的协议，如 VINES 、 TCP 、 UDP 等
- 校验和 (checksum) — 校验和是 16 位的错误检测域。目的机、网络中的每个网关要重新计算报文头的校验和，就如同源机器所做的一样
- 源 IP 地址 — 源计算机的 IP 地址
- 目的 IP 地址 — 目的计算机的 IP 地址
- 填充 — 为了保证 IP 头长度是 32 位的整数倍，要填充额外的 0

TCP 协议分析

TCP 包头结构

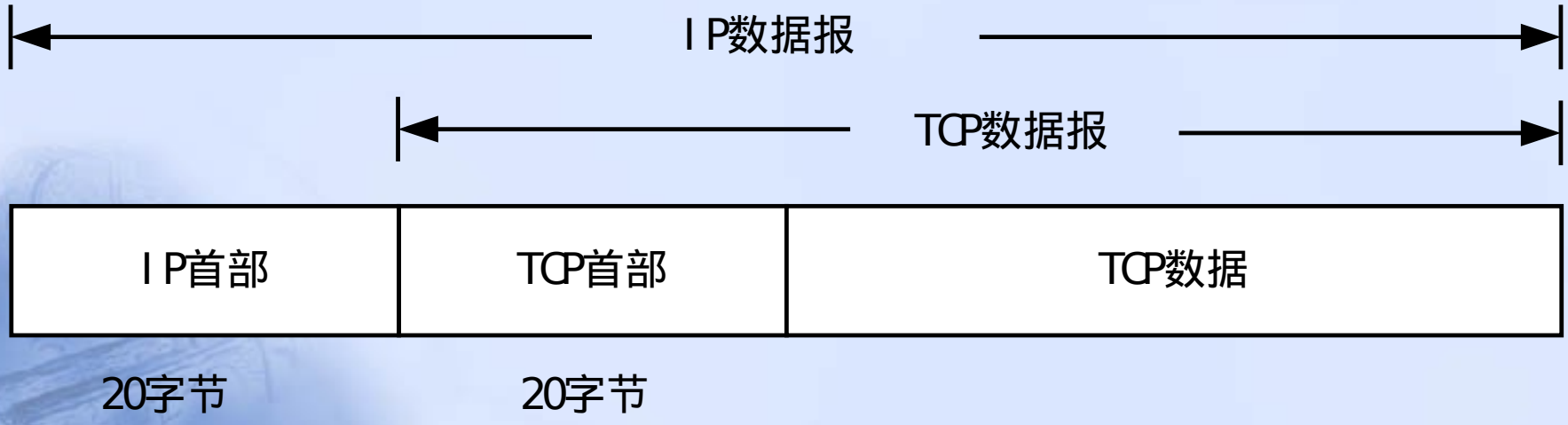
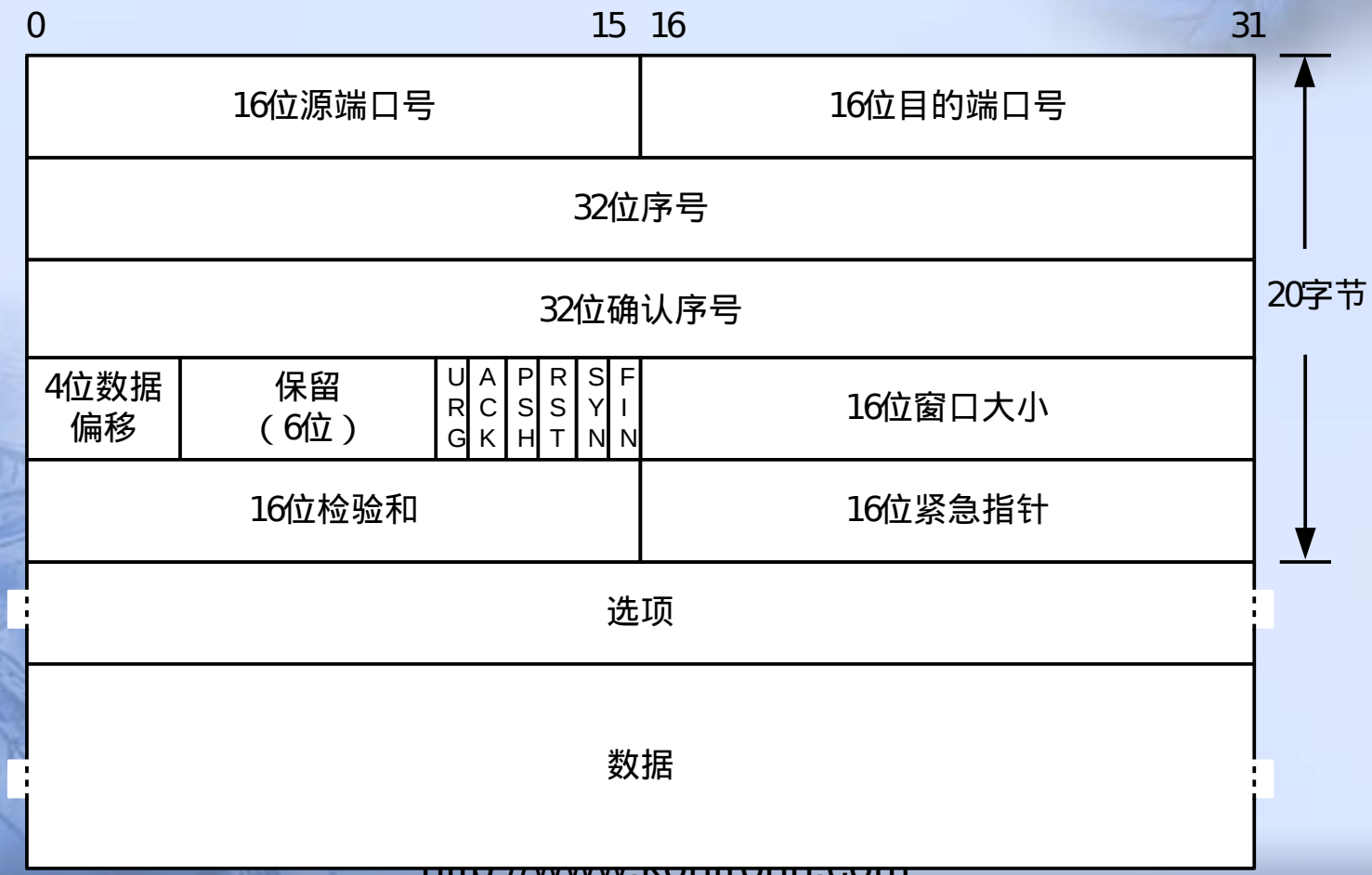


图 3-3 TCP 数据在 IP 数据报中的封装

TCP 包头结构



http://www.kontronn.com

图 3-4 TCP 包首部

TCP 包头结构

- TCP 源端口
  - 16 位的源端口域包含初始化通信的端口号
  - 源端口和源 IP 地址的作用是标识报文的返回地址
- TCP 目的端口
  - 16 位的目的端口域定义传输的目的
  - 这个端口指明报文接收计算机上的应用程序地址接口

TCP 包头结构

- TCP 序列号
  - 32 位的序列号由接收端计算机使用，重组分段的报文成最初形式
- TCP 应答号
  - TCP 使用 32 位的应答（ ACK ）域标识下一个希望收到的报文的第一个字节

TCP 包头结构

- 数据偏移
  - 这个 4 位域包括 TCP 头大小
  - 以 32 位数据结构或称为“双字”为单位
- 保留
  - 6 位恒置 0 的域
  - 为将来定义新的用途保留
- 标志
  - 6 位标志域，每 1 位标志可以打开一个控制功能
  - 这六个标志是：紧急标志、有意义的应答标志、推、重置连接标志、同步序列号标志、完成发送数据标志

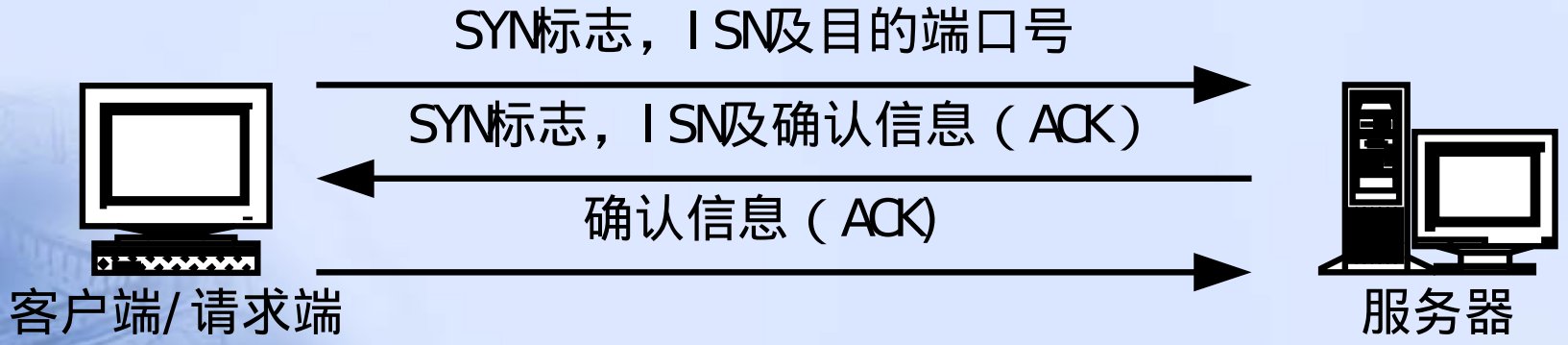
TCP 包头结构

- 窗口大小
  - 目的机使用 16 位的域告诉源主机，它想收到的每个 TCP 数据段大小。用于 TCP 协议的流量控制
- 校验和
  - TCP 头也包括 16 位的错误检查域—“校验和”域
- 紧急
  - 紧急指针域是一个可选的 16 位指针，指向段内的最后一个字节位置，这个域只在 URG 标志设置了时才有效。
- 选项
  - 至少一字节的可变长域标识哪个选项。

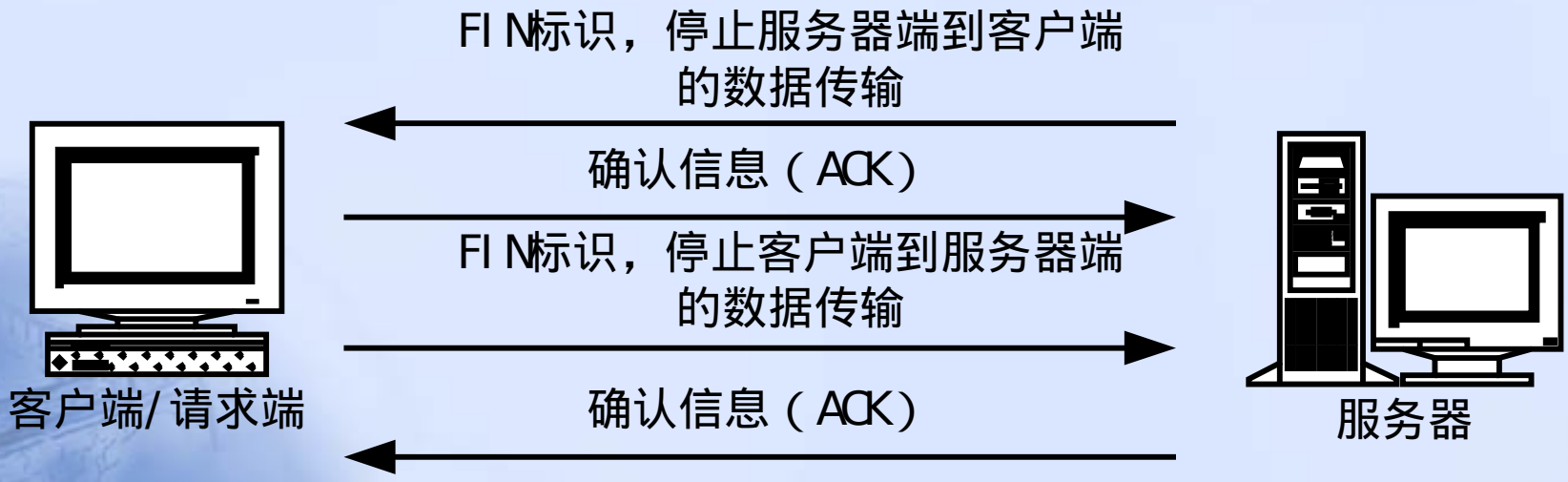
TCP 包头结构

- 数据
  - 域的大小是最大的 MSS， MSS 可以在源和目的机器之间协商
- 填充
  - 其目的是确保空间的可预测性、定时和规范大小
  - 这个域中加入额外的零以保证 TCP 头是 32 位的整数倍

建立一个 TCP 连接



结束一个 TCP 连接


FIN标识，停止服务器端到客户端的数据传输
确认信息（ACK）
FIN标识，停止客户端到服务器端的数据传输
确认信息（ACK）
客户端/请求端
服务器

网络编程基础知识

字节序列转换

- 每一个机器内部对变量的字节存储顺序不同（有的系统是高位在前，底位在后，而有的系统是底位在前，高位在后 ），而网络传输的数据大家是一定要统一顺序的。所以对与内部字节表示顺序和网络字节顺序不同的机器，就一定要对数据进行转换。

字节序列转换

- #include <arpa/inet.h>
- uint16_t htons(uint16_t hostshort);
- 功能：将主机字节顺序转换为网络字节顺序。
- 返回值：成功返回转换后的字节序。

地址格式转换

- #include <sys/types.h>
- #include <sys/socket.h>
- #include <arpa/inet.h>
- int inet_pton(int af, const char *src, void *dst);
- 功能：将点分格式的地址字符串转换为网络字节序整型数。
- 返回值：成功返回 1 ，错误返回 -1
- 参数：
  - af ： 转换格式 AF_INET （ IPV4 ）或 AF_INET6 （ IPV6 ）
  - src ： 点分格式的地址
  - dst ： 转换后的整型变量的地址

地址格式转换

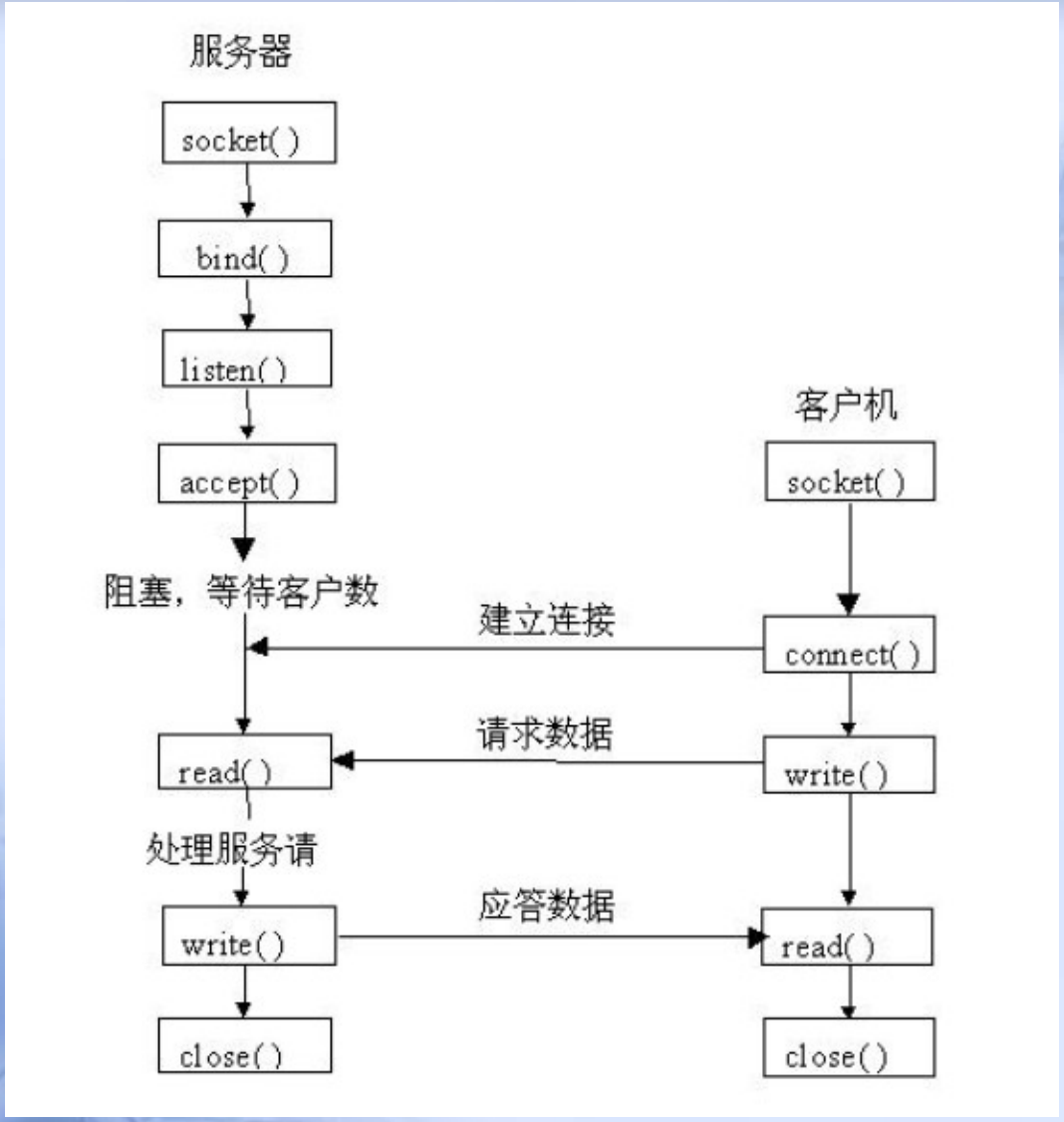
- #include <sys/types.h>
- #include <sys/socket.h>
- #include <arpa/inet.h>
- const char *inet_ntop(int af, const void *src, char *dst, socklen_t cnt);
- 功能：将网络字节序整型转换为点分格式的 IP 地址
- 返回值：成功返回转换后地址，失败返回 NULL
- 参数：
  - af：转换格式 AF_INET（IPV4）或 AF_INET6（IPV6）
  - src：整型变量地址
  - des：用来存储转换后数据的地址
  - cnt：存储空间大小

网络编程基础

- socket 概述
- Linux 中的网络编程通过 socket 接口实现。socket 既是一种特殊的 IO，它也是一种文件描述符。一个完整的 socket 都有一个相关描述{协议，本地地址，本地端口，远程地址，远程端口}；每一个 socket 有一个本地的唯一 socket 号，由操作系统分配。

网络编程基础

- 套接字有三种类型：
- 流式套接字（ SOCK_STREAM ）
  - 流式的套接字可以提供可靠的、面向连接的通讯流。它使用了 TCP 协议。 TCP 保证了数据传输的正确性和顺序性。
- 数据报套接字（ SOCK_DGRAM ）
  - 数据报套接字定义了一种无连接的服务，数据通过相互独立的报文进行传输，是无序的，并且不保证可靠，无差错。使用数据报协议 UDP 协议。
- 原始套接字。
  - 原始套接字允许对低层协议如 IP 或 ICMP 直接访问，主要用于新的网络协议实现的测试等。



基于数据流的 socket 编程流程

http://www.kontronn.com

网络编程相关函数

socket 系统调用

- #include <sys/types.h>
- #include <sys/socket.h>
- int socket(int family, int type, int protocol);
- 功能： socket 函数创建一个用于网络通信的套接字，并返回该套接字的整数描述符。
- 返回值：成功返回正数，失败返回 -1.
- 参数：
  - family 是协议或地址族（ 对于 TCP/IP 为 PF_INET ，也可使用 AF_INET ）
  - type 是服务的类型（对于 TCP 为 SOCK_STREAM ，对于 UDP 为 SOCK_DGRAM ）
  - protocol 是使用的协议号，或是用 0 指定 family 和 type 的默认协议号。

bind 系统调用

- #include <sys/types.h>
- #include <sys/socket.h>
- int bind(int sockfd, const struct sockaddr * localaddr, socklen_t addrlen);
- 功能： bind 为一个套接字指明一个本地 IP 和协议端口号。
- 返回值：成功则返回 0 ，失败返回 -1 。
- 参数：
  - sockfd 是由 socket 调用创建一个套接字描述符
  - localaddr 是一个地址结构，指定一个 IP 地址和协议端口号
  - addrlen 是地址结构的字节数大小。
  - struct sockaddr {
  - sa_family_t sa_family;
  - char sa_data[14];
  - }

listen 系统调用

- #include <sys/socket.h>
- int listen(int sockfd, int queuelen);
- 功能：服务器使用 listen 使套接字处于被动状态（准备接受联入请求）。
- 返回值：成功则返回 0 ，错误返回 -1 。
- 参数：
  - sockfd 是一个由 socket 调用创建的套接字描述符
  - queuelen 是连接请求的队列大小

accept 系统调用

- #include <sys/types.h>
- #include <sys/socket.h>
- int accept(int sockfd, struct sockaddr *addr, socklen_t *addrlen);
- 功能：服务器调用 socket 创建一个套接字，用 bind 指定一个本地 IP 地址和协议端口号，然后用 listen 使套接字处于被动状态，并设置连接请求队列的长度。 accept 从队列中取走下一个连接请求（或一直在那里等待下一个连接请求的到达），为请求创建一个新套接字，并返回新的套接字描述符。 accept 只用于流套接字（如 TCP 套接字）。
- 返回值：成功返回一个非负套接字描述符。错误返回 -1。
- 参数：
  - sockfd 是是由 socket 函数创建的一个套接字描述符。
  - addr 是一个地址结构的指针。 accept 在该结构中填入远程机器的 IP 地址和协议端口号。
  - addrlen 初始指定为 struct sockaddr 结构体大小的存放地址。

connect 系统调用（客户端）

- #include <sys/types.h>
- #include <sys/socket.h>
- int connect(int sockfd, const struct sockaddr * addr, socklen_t addrlen);
- 功能： connect 允许调用者为先前创建的套接字指明远程端点的地址。如果套接字使用了 TCP ， connect 就使用三次握手建立一个连接。
- 返回值：成功返回 0 ，错误返回 -1 。
- 参数：
  - sockfd 是一个套接字的描述符
  - addr 是远程机器端点地址
  - addrlen 是 struct sockaddr 结构体的大小