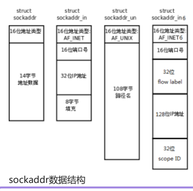
socket编程

- 概念
  - sockaddr数据结构



  - 特征
    - 1.文件描述符（索引两个内核缓冲区R/W）
    - 2.通信过程中，套接字【成对出现】
    - 3.Linux特殊文件类型（管道，套接字，字符设备，块设备）
  - 网络字节序
    - 小端（pc本地存储） 低地址低字节
    - 大端（网络存储） 低地址高字节
    - 字节序转换函数
      s:16位，端口
      l:32位，IP地址
      函数
      - uint32_t htonl(uint32_t hostlong); (IP)本地字节序---->网络字节序
      - uint16_t htons(uint16_t hostshort); (端口)本地字节序-->网络字节序
      - uint32_t ntohl(uint32_t netlong)
      - uint16_t ntohs(uint16_t netshort);
    - IP地址转换
      点分（p）<->网络字节序ip地址(二进制)
      函数
      - int inet_pton(int af, const char *src, void *dst);
        - af:AF_INET,AF_INET6
        - 返回值
          - 成功：1
          - 异常：0，src不是一个有效地址
          - 失败：-1
      - const char *inet_ntop(int af, const void *src, char *dst, socklen_t size);
- 网络套接字编程
  - socket模型创建流程

  - 函数
    - int socket(int domain, int type, int protocol);
      - 参数
        - domain:AF_INET,AF_INET6
        - type(通信方式):SOCK_STREAM(TCP，字节流),SOCK_DGRAM(UDP，报文)
        - protocol，一般为0，根据type选择协议
      - 成功，返回套接字描述符
      - 失败，-1 errno
    - int bind(int sockfd, const struct sockaddr *addr, socklen_t addrlen);
      - 给socket绑定一个地址结构（IP+Port）
      - 参数 sockaddr
        - 步骤

```
struct sockaddr_in addr;
addr.sin_family = AF_INET/AF_INET6
addr.sin_port = htons(9527);
        int dst;
        inet_pton(AF_INET, "192.157.22.45", (void *)&dst);
addr.sin_addr.s_addr = dst;
【*】addr.sin_addr.s_addr = htonl(INADDR_ANY);        取出系统中有效的任意IP地址。二进制类型。
bind(fd, (struct sockaddr *)&addr, size);
```

    - int listen(int sockfd, int backlog) backlog:设置同时进行【三次握手】的客户端连接最大数量，最大值128
    - int accept(int sockfd, struct sockaddr *addr, socklen_t *addrlen);
      - 阻塞等待客户端建立连接，
        成功返回一个与客户端成功连接的【socket描述符】
      - 参数
        - socket 监听socket
        - addr 成功与服务器建立连接的客户端地址结构（IP+Port）
        - addrlen
          - 传入传出参数
          - 入：sizeof(struct sockaddr_in) 出：客户端地址大小
      - 返回值
        - 成功 返回与客户端建立连接的socket描述符
        - 失败-1，errno
    - int connect(int sockfd, const struct sockaddr *addr, socklen_t addrlen);
      - 与服务器建立连接
      - 客户端port【隐式绑定】，
        端口由系统分配

- TCP协议
  - 协议
    - HTTP/FTP/Telnet
    - TCP/UDP
    - IP/ICMP/IGMP
    - ARP/RARP
  - 分层模型
    - ISO7层
      - 物链网传会表应
    - TCP/IP4层
      - 链网传应
  - 常见协议帧头
    - 以太网帧头
      - 源mac地址
      - 物理mac地址
      - 类型
        - 0800:数据包
        - 0806：ARP
        - 8035:RARP
      - 1500字节
    - IP帧头
      - 主要格式
        - 4位版本号
        - 4位首部长度/4B
        - 16为总长度/B
        - 8位TTL
        - 32位源IP
        - 32位目的IP
      - 指定网络中唯一终端
    - UDP帧头
      - 主要格式
        - 16位源端口号
        - 16位目的端口号
    - TCP帧头
      - 16位源目的端口
      - 32SEQ
      - 32ACK
      - 4位首部长度/4B
      - 标志位
        - SYN
        - ACK
        - FIN
      - 16位窗口大小
  - 通信时序
    - 三次握手
      - SYN,Seq=c
      - SYN+ACK,Seq=s,ACK=c+1
      - ACK,Seq=c+1,ACK=s+1
    - 关闭
      - FIN+ACK,Seq=K,ACK=L
      - ACK,Seq=L,ACK=K+1
      - FIN+ACK,Seq=L,ACK=K+1
      - ACK,Seq=K,ACK=L+1
  - 滑动窗口
    - 发送方窗口右滑（接受ACK）时改变窗口大小
    - 基于【窗口】流量控制：控制发送放速度
    - 拥塞控制：发送方降低速度以较少【网络】压力
  - MTU
    - 最大传输单元
  - Mss
    - 最大报文段长度
  - 半关闭
    - 二次握手
      - 服务端(关闭)：FIN,Seq=L,ACK=K+1
      - 客户端: ACK,Seq=K,ACK=L+1
    - shutdown()
      - SHUT_RD(不能读)
      - SHUT_WR(不能写)
      - SHUT_RDWR
    - 进入FIN_WAIT_2状态
  - TCP状态转换图
    - 客户端（主动开启）
      - SYN_SENT
      - ESTABLISHED
      - FIN_WAIT_1(主动关闭)
      - FIN_WAIT_2
      - TIME_WAIT
        - 当最终ACK发送后，处于WAIT状态，2倍于最大生存期
        - 以避免最终ACK丢失（即可重新发送）
      - CLOSED
    - 服务端（被动开启）
      - LISTEN
      - SYN_RCVD
      - ESTABLISHED
      - CLOSE_WAIT(被动关闭)
      - LAST_ACK
      - CLOSE
  - 心跳检测
    - 心跳包
    - 乒乓包
    - TCP自带（2小时+11分15秒）
  - 端口复用（setsocketopt）