- 线程
  - 线程概念
    - 什么是线程
      - LWP轻量级的进程
      - 共享进程地址空间
      - 拥有独立的PCB
      - Linux下线程是最小的执行单位
    - 内核实现
      - 创建线程与创建进程类似clone
      - 有不同PCB，但三级页表相同
      - 是寄存器和栈的集合
      - 以LWP号作为分配cpu资源依据（ps -Lf 进程ID 可查看）
    - 共享资源
      - 文件描述符表
      - 信号处理方式
      - 当前工作目录
      - 用户ID、组ID
      - 内存地址空间(栈除外)
    - 非共享资源
      - 线程ID
      - 处理器线程和栈指针
      - 用户栈空间
      - errno变量
      - 信号屏蔽字
      - 调度优先级
    - 优缺点
      - 优点
        - 提高程序并发
        - 开销小
        - 数据通信方便
      - 缺点
        - 库函数稳定性差
        - 调试困难，不支持gdb
        - 信号支持不好
    - man page安装
      - sudo apt-get update 更新本地软件列表
      - sudo apt-get install manpages-posix manpages-posix-dev
      - 验证：man -k pthread
  - 线程控制原语
    - pthread_self函数
      - 获取线程ID。对应进程 getpid()
      - 参数：无
      - 返回值：
        - 成功：线程ID
        - 失败：无
    - pthread_create函数
      - 创建一个新线程。对应进程fork()
      - 参数：
        - （传出参数）新线程ID
        - attr线程属性。NULL表使用默认属性 pthread_attr_t
        - 线程主函数：void *(*tfn)(void *)
        - 线程主函数参数
      - 返回值：
        - 成功：0
        - 失败：错误号
      - 循环创建多个子线程模型。
    - pthread_exit函数
      - 将当前线程退出。对应进程exit或return
      - 参数：
        - 线程退出值(void *类型)
        - NULL：无退出值
      - 返回值：无
      - 对比：
        - return：返回到调用者那里去
        - exit：将当前进程退出
        - pthread_exit:将当前线程退出
    - pthread_join函数
      - 阻塞等待线程退出，获取退出状态。对应进程waitpid()
      - 参数：
        - pthread：线程ID
        - retval：
          - 被回收线程退出值（传出参数void **类型）
          - NULL 不关心线程退出值
      - 返回值：
        - 成功：0
        - 失败：错误号
    - pthread_detach函数
      - 实现线程分离---线程结束自动回收残留资源
      - 参数：待分离线程ID
      - 返回值：
        - 成功：0
        - 失败：错误号
    - pthread_cancel函数
      - 杀死(取消)线程。对应进程 kill()
      - 参数：欲杀死(取消)的线程ID
      - 返回值：
        - 成功：0
        - 失败：错误号
      - 取消点：
        - man 7 pthreads
        - pthread_testcancel()测试取消点
      - 终止线程的方式：
        - return
        - pthread_exit
        - pthread_cancel
  - 线程属性
    - 可在创建线程之前设置修改其默认属性
    - pthread_attr_t结构体
      - 线程分离状态
      - 线程栈空间大小
      - 线程警戒区大小
      - 线程栈低地址
      - ……
    - 属性初始化
      - pthread_attr_t attr;
      - pthread_attr_init函数
        - 参数：attr：(传出参数)待初始化的结构体
        - 返回值：
          - 成功：0
          - 失败：错误号
      - pthread_attr_destroy函数
    - 线程分离状态
      - pthread_attr_getdetachstate函数
        - 获取线程分离状态
      - pthread_attr_setdetachstate函数
        - 设置线程分离状态
        - 参数：
          - attr：线程属性结构体指针
          - detachstate：(传出参数)
            - PTHREAD_CREATE_DETACHED（分离）
            - PTHREAD_CREATE_JOINABLE（非分离）
        - 返回值：
          - 成功：0
          - 失败：错误号
    - 通过线程属性设置线程分离
      - 1. 创建线程属性 pthread_attr_t attr;
      - 2. 初始化线程属性 pthread_attr_init(&attr);
      - 3. 设置线程属性为分离属性 pthread_attr_setdetachstate(&attr, PTHREAD_CREATE_DETACHED);
      - 4. 根据新线程属性，创建线程 pthread_create(&tid, attr, tfn, NULL);
      - 5. 销毁线程属性对象 pthread_attr_destroy(&attr);
  - NPTL
    - pthread库版本
    - 查看：getconf GNU_LIBPTHREAD_VERSION
    - gcc编译指定 -lpthread选项
  - 线程使用注意细节
    - 退出线程应使用pthread_exit
    - 避免僵尸线程产生
      - pthread_join回收线程
      - pthread_detach分离线程
      - 指定分离属性，再pthread_create线程
    - 线程共享进程地址空间：malloc和mmap内存可被其他线程释放
    - 线程中应避免使用fork，除非马上exec
    - 线程fork后，除调用线程外，其他线程均pthread_exit
    - 避免线程和信号混用