- 杨劲松
-
- yjs@oldhand.org
- 139 0116 0834
-
- 线程部分的讲课内容：
- http://192.168.204.6/pthreads-programming-2012.03.24.pdf
-

- 全部的代码：
  - http://192.168.204.6/examples/2012.04.07/
- 传递结构体的例子：
  - http://192.168.204.6/examples/2012.04.07/test2.c
- wget 下载全部代码可以使用 -m(--mirror) 参数
-

- 全部的代码：
  - http://192.168.204.6/examples/2012.04.07/
  -
  -
- Shell 作业
  - http://192.168.204.6/shell.pdf

- 代码：
  - http://192.168.204.6/examples/2012.04.09/mutex/
-
-

使用 Valgrind 动态分析线程

- Valgrind 是动态分析工具
  - http://valgrind.org
- 下载、编译 valgrind
  - http://valgrind.org/downloads/valgrind-3.7.0.tar.bz2
- 线程分析可以使用 helgrind
  - http://valgrind.org/docs/manual/hg-manual.html
  - $ valgrind --tool=helgrind app

W. Richard Stevens 电子书

- 全部的电子书：
  - http://192.168.204.6/w.richard-stevens.tar.gz
- 包括：
  - 《UNIX 环境高级编程》（第一版中文版，第二版中文版和英文版）
  - 《UNIX 网络编程》（第一卷，第一版，第二版，第二版有英文版）
  - 《UNIX 网络编程》（第二卷，第一版中文版，第二版中文版）
  - 《TCP/IP 详解》（共三卷，第一卷有英文版）

死锁的问题

- 可以造成死锁的代码：
  - http://192.168.204.6/examples/2012.04.09/mutex/deadlock.c
- 现象：
  - 程序执行到某个阶段之后不继续执行了，从 deadlock.c 的执行看，不再输出 + 或者 - 了。
- 如何解决死锁问题？
  - 使用 trylock
  - deadlock.c 需要两个线程在获得锁的时候，按照相同的顺序，都需要先获得 mutex1 ，再获得 mutex2
  - 引入一个协调者来解决问题

条件变量代码

- http://192.168.204.6/examples/2012.04.09/cond/

读写锁代码

- 使用 mutex 的情况
  - http://192.168.204.6/examples/2012.04.09/rwlock/test1.c
- 使用 rwlock 的情况
  - http://192.168.204.6/examples/2012.04.09/rwlock/test2.c

明天讲 TCP/IP 协议

- 讲课的内容，可能的话提前做一下预习
  - http://192.168.204.6/tcpip-overview-2012.03.24.pdf
-
- 请安装 wireshark
  - Ubuntu10.x 和 11.x 可以通过 apt-get 安装
  - Ubuntu 7.04 的安装不了了，你可以使用其他人机器上的 wireshark
  - Windows 系统可以安装 wireshark

全部的代码

- http://192.168.204.6/examples-2012.04.09.tar.gz

Socket 编程

- 请大家下载：
  - http://192.168.204.6/socket-programming-2012.03.24.pdf

全部的代码

- IP 地址转换
  - http://192.168.204.6/examples/2012.04.11/test1.c
  - http://192.168.204.6/examples/2012.04.11/test2.c
- 使用 getservbyname()
  - http://192.168.204.6/examples/2012.04.11/test3.c
- 地址结构处理
  - http://192.168.204.6/examples/2012.04.11/test4.c
  - http://192.168.204.6/examples/2012.04.11/test5.c

使用 1024 以下端口

- 对可执行文件设置 set-uid-bit
  - # chown root app
  - # chmod 4755 app
- 程序运行之后， euid 为 0 ，将拥有 root 的 permission ，可以绑定 1024 以下端口
- 程序绑定成功之后，通过如下的调用降低权限为普通用户
  - setuid(getuid());

实现一个 wget

- URL: [http|https|ftp]://[user:password@]hostname[:port]/path[?parameters]
- Parameters: name=value&name=value
- 如果没给定 port，则用 80 为默认断口号
- 解析主机名可以使用 gethostbyname()
- HTTP 协议可以参考 RFC2616

全部的代码

- 课堂笔记在 2012.04.11 目录下， memo.pdf
- http://192.168.204.6/examples-2012.04.11.tar.gz

TCP 服务器

- 迭代模型（循环模型）
  - http://192.168.204.6/examples/2012.04.13/tcp/server.c
- 如何实现多进程并发模型？
  - http://192.168.204.6/examples/2012.04.13/tcp/server1.c
- 如何实现多线程并发模型？
  - http://192.168.204.6/examples/2012.04.13/tcp/server2.c
- 如何实现进程池 / 线程池模型？
  - http://192.168.204.6/examples/2012.04.13/tcp/server3.c
- 如何评价上述各种模型的优缺点和适用性？

补充资料

- 一些补充资料，供大家参考
  - http://192.168.204.6/ref.pdf
- 有兴趣的话，可以在网上搜索一下“ C10K problem”，也就是如何维护 10,000 个连接的问题
  - http://www.kegel.com/c10k.html

实现使用 UDP 方式聊天的程序

- 两端分别运行 UDP 聊天程序
  - 从标准输入读用户的信息，发送给对方
  - 从对方来的信息，显示在标准输出上
- 可以将屏幕分区，上面显示对方发送过来的消息，下部显示本地消息
  - 可以使用 libncurses
- 控制权冲突：
  - 从标准输入读和从 socket 读冲突，如果阻塞在标准输入上，此时即便 socket 有数据，也无法读，反之亦然。
  - 如何解决？
- 群聊如何实现？

- http://192.168.204.6/examples-2012.04.13.tar.gz

UNIX DOMAIN 编程

- 流式服务器
  - http://192.168.204.6/examples/2012.04.14/unixdomain/stream/server.c
- 流式客户端
  - http://192.168.204.6/examples/2012.04.14/unixdomain/stream/client.c
- 请实现数据报通讯的客户端与服务器端
  - 客户端也需要 bind，否则服务器返回不了数据

I/O 模型与调度模型

- 请大家下载：
  - http://192.168.204.6/models-2011.09.06.pdf
-
-

select 代码

- 未完成的代码
  - http://192.168.204.6/examples/2012.04.14/io-multiplexing/test.c
- 如何管理多个连接？
  - 使用线性表 (数组变种) ，以描述符作为下标来索引数组
- 每个连接都需要管理什么？
  - 需要管理两个缓冲区，一个发送缓冲区，一个接收缓冲区
- 缓冲区如何管理？
  - 缓冲区需要动态分配
  - 缓冲区需要有一个地址指针
  - 缓冲区需要有一个 size
  - 定义一个 payload_length

全部的代码

- http://192.168.204.6/examples-2012.04.14.tar.gz

Bitmask vs value?

- Bitmask 按 bit 表意，每一个 bit 表示一个意思，可以通过位操作来访问
  - 设置： | MARK
  - 取消： & ~MARK
  - 检测： if (bitmask & MARK)
- value 整体表意，通过”＝”赋值
- 已经学习过的用 bitmask 表示的
  - open() 的参数
  - exit() 返回的 status

- poll 代码
  - http://192.168.204.6/examples/2012.04.16/io-multiplexing/poll_test.c
- epoll 代码
  - http://192.168.204.6/examples/2012.04.16/io-multiplexing/epoll_test.c
  - http://192.168.204.6/examples/2012.04.16/io-multiplexing/epoll_test.new.c
-

大作业：实现一个 WEB server

- 相关的 RFC:
  - http://192.168.205.6/rfc/rfc2616.txt
  - http://192.168.205.6/rfc/rfc3875.txt
- 相关资料：
  - http://192.168.205.6/webserver.pdf
  - http://192.168.205.6/web-server.pdf

思考一个问题

- 当 WEB Server 发送一个文件给 Browser 时，会发生几次内核空间和用户空间的数据拷贝？
  - 数据从内核空间拷贝到用户空间调用 copy_to_user() 函数
  - 数据从用户空间拷贝到内核空间调用 copy_from_user() 函数
  - 这两个函数是内核定义的，用户空间不能调用
  - 这两个函数调用非常昂贵 (开销非常大)，后边学习驱动的时候你们会知道原因
  - 因此减少这两个函数的调用会使程序的性能有很大的提升
- 如果不将数据拷贝到用户空间，直接在内核空间发送是不是更好？
  - 参考 sendfile(2)

- 全部的代码：
  - http://192.168.204.6/examples-2012.04.16.tar.gz
- 课堂笔记：
  - http://192.168.204.6/memo-2012.04.16.pdf

Shell programming

- http://192.168.204.6/akae-shell-programming-yjs-2011.03.16.pdf
-
- ABS
  - http://tldp.org/LDP/abs/html/
- ABS 中文版
  - http://www.linuxsir.org/main/node/140
  - Google search “ABS 中文版 site:linuxsir.org”

- http://192.168.204.6/examples/2012.04.17/
- 任务 1：写一个脚本，每天下午 5 点备份自己的源代码和文档
  - 生成的备份文件名要有时间戳
  - 备份的文件可能有多层目录
  - 把备份的源代码和文档传送到远端的服务器的某个目录下
- 任务 2：写一个脚本，检查给定目录下的各层子目录的 core 文件，发现之后删除，并生成一个报告
- 任务 3: 写一个脚本，检查给定目录下各层子目录的 set-uid 文件，给出一个报告
- 任务 4：写一个脚本，从中国天气网，提取北京的天气信息
  - 可以通过 wget 下载页面，然后进行分析
  - http://bj.weather.com.cn/
  - http://192.168.204.6/examples/2012.04.17/weather.sh

我给其他班讲课的一些资料

- Linux 基本命令部分
  - http://192.168.204.6/akae-linux-basic-2011.08.05.pdf
- 文件 I/O 部分
  - http://192.168.204.6/akae-linux-io-course-yjs-2011.07.09.pdf
- 文件操作部分
  - http://192.168.204.6/akae-linux-file-course-yjs-2011.04.12.pdf
- 文件系统部分
  - http://192.168.204.6/akae-linux-fs-course-yjs-2011.04.13.pdf
- 进程、信号、进程间通讯：
  - http://192.168.204.6/process+ipc.pdf