**高级计算机网络**

**实 验 报 告**

**专业：人工智能**

**班级：智能专研25**

**姓名：王耀彬**

**学号：2025354100103**

|  |
| --- |
| **总成绩：**  **评语：**    **日期：** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **《高级计算机网络》实验报告** | | | | | |
| **实验题目** | 高级套接口编程实验 | **总序号** | | **一** | |
| **实验日期** | **2025.10.20** | | **总耗时** | | **2天** |
| **实验1.1：**   * 简要实验步骤（要求简短精炼，符合真实操作步骤，程序设计实验要求写出程序设计的伪码或流程图）：   （1）环境准备  使用ssh连接实验室服务器。日常使用的Pycharm软件“部署”功能连接。这里也  直接使用了。  （2）程序设计  （2.1）UDP服务器程序设计：  ·调用socket()函数创建一个UDP类型的套接口。  ·调用bind()函数将套接口绑定到本地8888端。  ·进入主循环后，使用recvfrom()函数阻塞等待客户端发送的数据包，该函数同时  获取发送方的地址信息。  ·收到消息后，解析并打印消息内容，再通过sendto()函数将消息回显给客户端。  ·当收到退出指令时，调用close()关闭套接口并结束程序。  （2.2）  UDP 客户端程序设计：  ·调用socket()创建UDP套接口，设置目标服务器地址为127.0.0.1:8888。  ·进入主循环后，读取用户从键盘输入的消息，调用sendto()发送消息。  ·发送完成后，使用recvfrom()等待并接收服务器的回复，打印响应内容。  （3）编译运行   |  | | --- | | make  ./udp\_server  ./udp\_cilent |   （4）通信测试  客户端发送消息，服务器接收后显示消息。   * 实验结果（要求简要描述实验结果）：   ·成功创建UDP服务器和客户端程序  ·服务器在端口8888监听，等待客户端连接  ·客户端向服务器发送消息后，服务器成功回显消息  ·输入quit或exit可正常退出程序  ·验证了UDP套接口的无连接、数据报通信特性   * 结果截图（要求与本机mac地址在一张屏幕的截图）：   make运行截图：    附带MAC地址的服务端：    附带MAC地址的客户端：     * 问题总结（遇到的问题、解决方案以及遗留的疑问）：   1.最开始计划利用Vmvare虚拟机，下载ubuntu后规划20G存储，但系统成功安装启动后，反复卡死，遂计划使用实验室Linux服务器。  2. make: No targets specified and no makefile found：未进入正确目录。  **实验1.2：**   * 简要实验步骤（要求简短精炼，符合真实操作步骤，程序设计实验要求写出程序设计的伪码或流程图）：   （1）环境准备  利用ssh连接实验室服务器。  （2）程序设计  基础大致与实验1.1相同，绘制了程序流程图：    （3）编译运行   |  | | --- | | make  ./tcp\_server  ./tcp\_cilent |   （4）通信测试  客户端发送消息，服务器接收后显示消息。   * 实验结果（要求简要描述实验结果）：   ·成功创建TCP服务器和客户端程序  ·服务器在端口8889监听，等待客户端连接  ·客户端成功与服务器建立TCP连接（三次握手）  ·客户端发送消息后，服务器成功回显消息  ·输入quit或exit可正常断开连接并退出程序  ·验证了 TCP 套接口的面向连接、可靠通信特性   * 结果截图（要求与本机mac地址在一张屏幕的截图）：   make运行    开启TCP服务端，带MAC（以后都带，不再赘述）    开启TCP客户端并接收信息     * 问题总结（遇到的问题、解决方案以及遗留的疑问）：   1. Address already in use：端口被之前的进程占用，等待几秒或使用 SO\_REUSEADDR选项。  **实验1.3：**   * 简要实验步骤（要求简短精炼，符合真实操作步骤，程序设计实验要求写出程序设计的伪码或流程图）：   （1）环境准备  使用ssh连接到实验室服务器环境。  （2）程序设计  绘制了RAW套接口接收端与发送端流程图    （3）编译运行   |  | | --- | | make # 编译  sudo ./raw\_receiver # 终端1：接收ICMP包（需要root）  sudo ./raw\_sender # 终端2：发送ICMP包（需要root） |   （4）通信测试  发送端发送ICMP Echo Request，接收端捕获并解析IP头和ICMP头信息。   * 实验结果（要求简要描述实验结果）：   ·成功创建RAW套接口程序  ·接收端能够捕获并解析ICMP数据包  ·发送端成功发送ICMP Echo Request并接收Echo Reply  ·正确解析了IP头部（版本、TTL、源/目的地址）和ICMP头部（类型、序列号）  ·验证了RAW套接口直接访问网络层的特性   * 结果截图（要求与本机mac地址在一张屏幕的截图）：   make编译成功    发送    接收       * 问题总结（遇到的问题、解决方案以及遗留的疑问）：   1. Operation not permitted：未使用root，使用sudo即可  **实验1.4：**   * 简要实验步骤（要求简短精炼，符合真实操作步骤，程序设计实验要求写出程序设计的伪码或流程图）：   （1）环境准备  使用ssh连接实验室Liunx服务器  （2）程序设计  绘制了流程图与伪代码。     |  | | --- | | //套接口选项程序伪代码：  tcp\_fd = socket(AF\_INET, SOCK\_STREAM, 0)  udp\_fd = socket(AF\_INET, SOCK\_DGRAM, 0)  // 读取选项  getsockopt(tcp\_fd, SOL\_SOCKET, SO\_REUSEADDR, &value)  getsockopt(tcp\_fd, SOL\_SOCKET, SO\_SNDBUF, &bufsize)  // 设置选项  setsockopt(tcp\_fd, SOL\_SOCKET, SO\_REUSEADDR, 1)  setsockopt(tcp\_fd, IPPROTO\_TCP, TCP\_NODELAY, 1)  // 验证并打印  getsockopt(tcp\_fd, SOL\_SOCKET, SO\_REUSEADDR, &value)  print(value)  close(tcp\_fd)  close(udp\_fd) |  |  | | --- | | //控制信息程序伪代码  sockfd = socket(AF\_INET, SOCK\_DGRAM, 0)  setsockopt(sockfd, IPPROTO\_IP, IP\_PKTINFO, 1) // 启用包信息  setsockopt(sockfd, IPPROTO\_IP, IP\_RECVTTL, 1) // 启用TTL获取  bind(sockfd, port=8890)  while (true):  msg = recvmsg(sockfd) // 使用recvmsg而非recvfrom  print(msg.data)  // 遍历控制信息  for cmsg in msg.control:  if cmsg.type == IP\_PKTINFO:  print(cmsg.dest\_ip, cmsg.interface)  if cmsg.type == IP\_TTL:  print(cmsg.ttl)  close(sockfd) |   （3）编译运行   |  | | --- | | cd 1.4  make  ./socket\_options # 直接运行，查看选项信息  ./udp\_control\_info # 终端1  echo "test" | nc -u 127.0.0.1 8890 # 终端2 |  * 实验结果（要求简要描述实验结果）：   ·成功读取TCP/UDP套接口的默认选项值  ·成功设置SO\_REUSEADDR、SO\_KEEPALIVE、TCP\_NODELAY等选项  ·观察到内核对缓冲区大小的调整（通常会加倍）  ·成功使用recvmsg()获取UDP包的控制信息  ·获取到目的IP地址、接收接口索引、TTL等辅助数据   * 结果截图（要求与本机mac地址在一张屏幕的截图）：   make编译    查看选项信息          终端2发出设置命令    终端1接收到，并修改     * 问题总结（遇到的问题、解决方案以及遗留的疑问）：   1. 缓冲区设置值与读取值不一致：内核会自动调整，正常现象  **实验1.5：**   * 简要实验步骤（要求简短精炼，符合真实操作步骤，程序设计实验要求写出程序设计的伪码或流程图）：   （1）环境准备  使用ssh连接实验室Linux服务器  （2）程序设计  以下是流程图与伪代码     |  | | --- | | listen\_fd = socket(AF\_INET, SOCK\_STREAM, 0)  bind(listen\_fd, port=8891)  listen(listen\_fd, backlog=10)  while (true):  conn\_fd = accept(listen\_fd) // 阻塞等待连接  // 为每个客户端创建新线程  client\_info = {conn\_fd, client\_addr, client\_id++}  pthread\_create(thread, handle\_client, client\_info)  pthread\_detach(thread) // 分离线程，无需join  // 工作线程函数  function handle\_client(client\_info):  send(welcome\_message)  while (true):  data = recv(client\_info.conn\_fd)  if data == "quit" or disconnected:  break  send(echo\_response)  close(client\_info.conn\_fd) |   （3）编译运行   |  | | --- | | make  ./mt\_server # 终端1：启动多线程服务器  ./mt\_client # 终端2：启动多客户端测试 |  * 实验结果（要求简要描述实验结果）：   ·成功实现多线程TCP服务器  ·服务器能够同时处理多个客户端连接  ·每个客户端由独立线程处理，互不干扰  ·使用pthread\_detach()实现线程自动回收  ·使用互斥锁保护共享的客户端计数器   * 结果截图（要求与本机mac地址在一张屏幕的截图）：   make编译结果    启动多线程服务器    启动多客户端测试     * 问题总结（遇到的问题、解决方案以及遗留的疑问）：   暂无  **实验1.6：**   * 简要实验步骤（要求简短精炼，符合真实操作步骤，程序设计实验要求写出程序设计的伪码或流程图）：   （1）环境准备  使用ssh登录实验室Linux环境  （2）程序设计  绘制了多路IO服务器流程图     |  | | --- | | tcp\_listen\_fd = socket(SOCK\_STREAM)  bind(tcp\_listen\_fd, TCP\_PORT)  listen(tcp\_listen\_fd)  udp\_fd = socket(SOCK\_DGRAM)  bind(udp\_fd, UDP\_PORT)  FD\_ZERO(all\_fds)  FD\_SET(tcp\_listen\_fd, all\_fds)  FD\_SET(udp\_fd, all\_fds)  max\_fd = max(tcp\_listen\_fd, udp\_fd)  while (true):  read\_fds = all\_fds // 复制，因为select会修改  select(max\_fd + 1, read\_fds, NULL, NULL, NULL) // 阻塞  if FD\_ISSET(tcp\_listen\_fd, read\_fds):  conn\_fd = accept(tcp\_listen\_fd)  FD\_SET(conn\_fd, all\_fds)  max\_fd = max(max\_fd, conn\_fd)  if FD\_ISSET(tcp\_conn\_fd, read\_fds):  data = recv(tcp\_conn\_fd)  send(tcp\_conn\_fd, echo\_response)  if FD\_ISSET(udp\_fd, read\_fds):  data, addr = recvfrom(udp\_fd)  sendto(udp\_fd, echo\_response, addr) |   （3）编译运行   |  | | --- | | make  ./select\_server # 终端1：启动服务器  ./select\_client # 终端2：启动客户端 |  * 实验结果（要求简要描述实验结果）：   ·成功实现使用 select 的多路 IO 服务器  ·服务器能够同时监听 TCP（端口 8892）和 UDP（端口 8893）  ·客户端可以选择使用 TCP 或 UDP 与服务器通信  ·select 实现了单线程处理多个套接口的能力  ·验证了 IO 多路复用的非阻塞并发特性   * 结果截图（要求与本机mac地址在一张屏幕的截图）：   make编译    服务器    客户端     * 问题总结（遇到的问题、解决方案以及遗留的疑问）： | | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **《高级计算机网络》实验报告** | | | | | |
| **实验题目** | Linux网络设备驱动开发实验 | **序号** | | **二** | |
| **实验日期** | **Xx、xx、xx** | | **总耗时** | |  |
| **实验2.1：**   * 简要实验步骤（要求简短精炼，符合真实操作步骤，程序设计实验要求写出程序设计的伪码或流程图）： * 实验结果（要求简要描述实验结果）： * 结果截图（要求与本机mac地址在一张屏幕的截图）： * 问题总结（遇到的问题、解决方案以及遗留的疑问）： | | | | | |
| **《高级计算机网络》实验总结报告（要求手写）** | | | | | |
| 写针对课程的学习收获以及对课程和老师教学方面的建议。不得少于半页，最好一页以上。 | | | | | |