**一、标题**

一个模拟的负载均衡系统的实现

**二、题目**

为了构建可伸缩的，高可用的网络服务，很多大型网站都采用了负载均衡技术。

利用负载均衡技术，可以将多台廉价的、低性能的服务器，组合成一台性能强劲的，高可用的虚拟服务器。

负载均衡的常见实现方式大致如下：

将网络服务的地址（如公网IP地址、tcp套接字等）部署在负载均衡器上，而不是真实的服务器上；

将负载均衡器作为网络服务的总入口，接收用户的所有访问请求；

负载均衡器接收到用户的访问请求后，将访问请求按照一定的策略分发给某一台真实的服务器进行处理；

真实服务器，对访问请求进行处理后，将处理结果发送给负载均衡器；

负载均衡器接收到真实服务器的处理结果，将他发送给用户。

下图展示了一个负载均衡系统的组网结构，图中包含了1台负载均衡器，3台真实的服务器。公网IP配置在负载均衡器上，负载均衡器与真实服务器之间，则通过私网地址进行通讯。

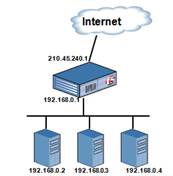


图1 负载均衡组网结构

有关负载均衡的技术原理及更多详细信息，请上网查阅相关资料或阅读相关书籍。

本题的任务，是在PC机上实现一个模拟的负载均衡系统。

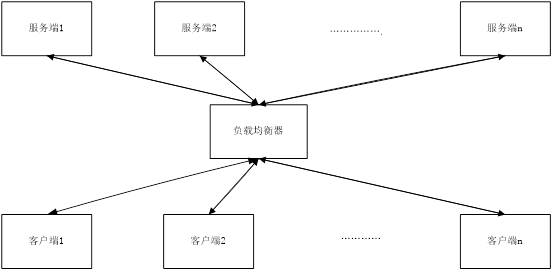
他包含如下3个可执行程序：

服务端 (server.exe)――辅助程序，通过UDP端口，提供时间查询服务。

负载均衡器(LB.exe)――核心程序，用于实现负载均衡功能。

客户端 (client.exe)――辅助程序，通过UDP端口，访问时间查询服务。

3个程序的协作关系如下图所示，其中客户端与服务端程序，需要起多个进程。

图 2 系统协作

上图中，每一个方框表示一个进程。每一个进程拥有一个唯一的id（注意，这是由用户配置的id，并非操作系统为进程分配的pid），进程之间一律通过UDP协议进行通信。系统运行起来之后，客户端通过UDP协议向负载均衡器发送“时间请求”消息，负载均衡器通过UDP协议将消息分发给某个服务端进行处理。服务端返回“时间应答”消息给负载均衡器，负载均衡器将“时间应答”消息返回给客户端。

为了简化实现，本模拟系统中，所有消息，都采用如下结构体进行封装。通过msg\_type字段的值，来区分不同类型的消息。

typedef struct

{

    /\* 消息的发送进程是谁，就填谁的id \*/

    unsigned src\_id;

/\* 消息的接收进程是谁，就填谁的id \*/

    unsigned dst\_id;

/\* 发送“时间请求”消息时填写，

回复“时间应答”消息时，其值要与请求消息保持一致。 \*/

    unsigned usr\_id;

/\* 消息类型：0, 时间请求；1, 时间答应；2, 心跳请求；3, 心跳应答 \*/

    unsigned msg\_type;

/\* 服务端回复“时间应答”消息时，

在data中填入当前时间的字符串，形式如“2013-06-20 13:56:28”即可  \*/

    char data[32];

} t\_msg;

**三、初赛阶段要求**

功能需求

a)服务端程序

服务端程序，需要起多个进程，每个进程拥有一个唯一的id，绑定到一个唯一的UDP端口上。每个服务端进程通过自己绑定的UDP端口接收“时间请求”消息，如果消息中的dst\_id等于自己的id，就向对端发送“时间应答”消息。否则，就丢弃此消息。

每个服务端进程的id、udp端口号，可以通过命令行参数传入，可以通过配置文件配置，也可以在进程运行时指定。三种方式，只要支持任意一种就行了。

服务端程序，需要具备一个调试开关。在运行过程中，可以打开/关闭调试开关。当调试开关打开后，服务端进程需要将自己接收/发送的每一个消息，都实时显示给用户看。

服务端程序，需要具备统计功能。在运行过程中，可以随时可看，每个服务端进程接收了多少条消息（正确的多少条，错误的多少条），应答了多少条消息。

b)负载均衡程序

负载均衡程序，只要启动一个进程即可。此进程拥有一个唯一的id，绑定到两个不同的UDP端口上。一个UDP端口（下文称为client\_udp\_port）用于收发客户端的消息，一个UDP端口（下文称为server\_udp\_port）用于收发服务端的消息。

负载均衡进程的id是多少，绑定的两个udp端口号是多少，支持多少个服务端，每个服务端的id、udp端口各是多少，均通过配置文件进行配置的。负载均衡进程启动时读入这些信息，运行过程中，不会改变。

对于客户端有多少个，每个客户端的id是多少，UDP端口号是多少，负载均衡进程对这些信息是一无所知的，也是无法预测的。

负载均衡进程通过client\_udp\_port接收到客户端的“时间请求”消息后，如果消息中的dst\_id不等于自己的id，就丢弃此消息。否则，就按照轮转算法选出一个服务端，将时间请求消息中的dst\_id改成此服务端的id后，将消息通过server\_udp\_port分发给该服务端处理。

负载均衡进程通过server\_udp\_port接收到客户端的“时间应答”消息后，将消息中的src\_id改成自己的id，然后将消息通过client\_udp\_port发送给消息中的dst\_id所指示的客户端。

负载均衡程序，需要具备一个调试开关。在运行过程中，可以打开/关闭调试开关。当调试开关打开后，程序需要将自己接收/发送的每一个消息，都实时显示给用户看。

负载均衡程序，需要具备统计功能。在运行过程中，可以随时可看，本进程从客户端接收了多少条消息（正确的多少条，错误的多少条），向客户端发送了多少条消息，从服务端接收了多少条消息（正确的多少条，错误的多少条），向服务端发送了多少条消息。

负载均衡程序，还需要具备日志功能。在运行过程中，如果出现异常事件（如UDP接收、发送失败等），需要记录日志，供后续分析查看。日志中，尽当尽可能包含详细的信息，如异常事件发生的时间、事件描述、事件原因等。

c)客户端程序

客户端程序，需要起多个进程，每个进程拥有一个id，一个usr\_id，绑定到一个默认分配的UDP端口上。每个客户端进程启动后，通过自己绑定的UDP端口向负载均衡器发送n条“时间请求”消息，并接收相应的时间应答消息。时间请求消息中的src\_id填写自己的id，usr\_id填写自己的usr\_id，dst\_id填写负载均衡器进程的id。

每个客户端进程的id、usr\_id，发送的消息条数n，可以通过命令行参数传入，可以通过配置文件配置，也可以在进程启动时指定。三种方式，只要支持任意一种就行了。

注意，不同客户端进程的id、usr\_id可以相同。

客户端进程，在运行过程中，需要将自己接收/发送的每一个消息，都实时显示给用户看。客户端进程完成自己的任务后，显示一下相关的统计信息，即可退出。统计信息包括本进程发送了多少条消息，接收了多少条消息（正确的多少条，错误的多少条）。

非功能需求

编程语言：C/C++语言

开发及运行环境：Windows操作系统

**四、初赛阶段审核标准及评价细则**

程序基本功能（60分）

服务端基本功能 (5分)

服务端统计功能(5分)

负载均衡端基本功能 (25分)

负载均衡端统计功能(5分)

负载均衡端日志功能(5分)

客户端,服务端,负载均衡端调试信息显示功能(5分)

客户端基本功能(5分)

客户端统计功能（5分）

代码质量（30分）

代码复杂度

代码可读性

代码注释

文档质量（10分）

设计文档

用户文档

**五、初赛阶段说明**

在初赛阶段需要提交下列材料：

设计、用户说明文档

可编译执行的源代码

可执行程序

如有缺失将直接影响评分。