# 通过socket传输Protocolbuf数据例子

Proto file

message log\_packet {

required fixed64 log\_time =1;

required fixed32 log\_micro\_sec =2;

required fixed32 sequence\_no =3;

required fixed32 shm\_app\_id =4;

required string packet\_id =5;

required string log\_level=6;

required string log\_msg=7;

}

Protocol buffer Client Code

#include <unistd.h>

#include "message.pb.h"

#include <iostream>

#include <google/protobuf/message.h>

#include <google/protobuf/descriptor.h>

#include <google/protobuf/io/zero\_copy\_stream\_impl.h>

#include <google/protobuf/io/coded\_stream.h>

#include <google/protobuf/io/zero\_copy\_stream\_impl\_lite.h>

using namespace google::protobuf::io;

using namespace std;

int main(int argv, char\*\* argc){

/\* Coded output stram \*/

log\_packet payload ;

payload.set\_log\_time(10);

payload.set\_log\_micro\_sec(10);

payload.set\_sequence\_no(1);

payload.set\_shm\_app\_id(101);

payload.set\_packet\_id("TST");

payload.set\_log\_level("DEBUG");

payload.set\_log\_msg("What shall we say then");

cout<<"size after serilizing is "<<payload.ByteSize()<<endl;

int siz = payload.ByteSize()+4;

char \*pkt = new char [siz];

google::protobuf::io::ArrayOutputStream aos(pkt,siz);

CodedOutputStream \*coded\_output = new CodedOutputStream(&aos);

coded\_output->WriteVarint32(payload.ByteSize());

payload.SerializeToCodedStream(coded\_output);

int host\_port= 1101;

char\* host\_name="127.0.0.1";

struct sockaddr\_in my\_addr;

char buffer[1024];

int bytecount;

int buffer\_len=0;

int hsock;

int \* p\_int;

int err;

hsock = socket(AF\_INET, SOCK\_STREAM, 0);

if(hsock == -1){

printf("Error initializing socket %d\n",errno);

goto FINISH;

}

p\_int = (int\*)malloc(sizeof(int));

\*p\_int = 1;

if( (setsockopt(hsock, SOL\_SOCKET, SO\_REUSEADDR, (char\*)p\_int, sizeof(int)) == -1 )||

(setsockopt(hsock, SOL\_SOCKET, SO\_KEEPALIVE, (char\*)p\_int, sizeof(int)) == -1 ) ){

printf("Error setting options %d\n",errno);

free(p\_int);

goto FINISH;

}

free(p\_int);

my\_addr.sin\_family = AF\_INET ;

my\_addr.sin\_port = htons(host\_port);

memset(&(my\_addr.sin\_zero), 0, 8);

my\_addr.sin\_addr.s\_addr = inet\_addr(host\_name);

if( connect( hsock, (struct sockaddr\*)&my\_addr, sizeof(my\_addr)) == -1 ){

if((err = errno) != EINPROGRESS){

fprintf(stderr, "Error connecting socket %d\n", errno);

goto FINISH;

}

}

for (int i =0;i<10000;i++){

for (int j = 0 ;j<10;j++) {

if( (bytecount=send(hsock, (void \*) pkt,siz,0))== -1 ) {

fprintf(stderr, "Error sending data %d\n", errno);

goto FINISH;

}

printf("Sent bytes %d\n", bytecount);

usleep(1);

}

}

delete pkt;

FINISH:

close(hsock);

}

Protocol buffer Server Code

#include <fcntl.h>

#include <string.h>

#include <stdlib.h>

#include <errno.h>

#include <stdio.h>

#include <netinet/in.h>

#include <resolv.h>

#include <sys/socket.h>

#include <arpa/inet.h>

#include <unistd.h>

#include <pthread.h>

#include "message.pb.h"

#include <iostream>

#include <google/protobuf/io/coded\_stream.h>

#include <google/protobuf/io/zero\_copy\_stream\_impl.h>

using namespace std;

using namespace google::protobuf::io;

void\* SocketHandler(void\*);

int main(int argv, char\*\* argc){

int host\_port= 1101;

struct sockaddr\_in my\_addr;

int hsock;

int \* p\_int ;

int err;

socklen\_t addr\_size = 0;

int\* csock;

sockaddr\_in sadr;

pthread\_t thread\_id=0;

hsock = socket(AF\_INET, SOCK\_STREAM, 0);

if(hsock == -1){

printf("Error initializing socket %d\n", errno);

goto FINISH;

}

p\_int = (int\*)malloc(sizeof(int));

\*p\_int = 1;

if( (setsockopt(hsock, SOL\_SOCKET, SO\_REUSEADDR, (char\*)p\_int, sizeof(int)) == -1 )||

(setsockopt(hsock, SOL\_SOCKET, SO\_KEEPALIVE, (char\*)p\_int, sizeof(int)) == -1 ) ){

printf("Error setting options %d\n", errno);

free(p\_int);

goto FINISH;

}

free(p\_int);

my\_addr.sin\_family = AF\_INET ;

my\_addr.sin\_port = htons(host\_port);

memset(&(my\_addr.sin\_zero), 0, 8);

my\_addr.sin\_addr.s\_addr = INADDR\_ANY ;

if( bind( hsock, (sockaddr\*)&my\_addr, sizeof(my\_addr)) == -1 ){

fprintf(stderr,"Error binding to socket, make sure nothing else is listening on this port %d\n",errno);

goto FINISH;

}

if(listen( hsock, 10) == -1 ){

fprintf(stderr, "Error listening %d\n",errno);

goto FINISH;

}

//Now lets do the server stuff

addr\_size = sizeof(sockaddr\_in);

while(true){

printf("waiting for a connection\n");

csock = (int\*)malloc(sizeof(int));

if((\*csock = accept( hsock, (sockaddr\*)&sadr, &addr\_size))!= -1){

printf("---------------------\nReceived connection from %s\n",inet\_ntoa(sadr.sin\_addr));

pthread\_create(&thread\_id,0,&SocketHandler, (void\*)csock );

pthread\_detach(thread\_id);

}

else{

fprintf(stderr, "Error accepting %d\n", errno);

}

}

FINISH:

;//oops

}

google::protobuf::uint32 readHdr(char \*buf)

{

google::protobuf::uint32 size;

google::protobuf::io::ArrayInputStream ais(buf,4);

CodedInputStream coded\_input(&ais);

coded\_input.ReadVarint32(&size);//Decode the HDR and get the size

cout<<"size of payload is "<<size<<endl;

return size;

}

void readBody(int csock,google::protobuf::uint32 siz)

{

int bytecount;

log\_packet payload;

char buffer [siz+4];//size of the payload and hdr

//Read the entire buffer including the hdr

if((bytecount = recv(csock, (void \*)buffer, 4+siz, MSG\_WAITALL))== -1){

fprintf(stderr, "Error receiving data %d\n", errno);

}

cout<<"Second read byte count is "<<bytecount<<endl;

//Assign ArrayInputStream with enough memory

google::protobuf::io::ArrayInputStream ais(buffer,siz+4);

CodedInputStream coded\_input(&ais);

//Read an unsigned integer with Varint encoding, truncating to 32 bits.

coded\_input.ReadVarint32(&siz);

//After the message's length is read, PushLimit() is used to prevent the CodedInputStream

//from reading beyond that length.Limits are used when parsing length-delimited

//embedded messages

google::protobuf::io::CodedInputStream::Limit msgLimit = coded\_input.PushLimit(siz);

//De-Serialize

payload.ParseFromCodedStream(&coded\_input);

//Once the embedded message has been parsed, PopLimit() is called to undo the limit

coded\_input.PopLimit(msgLimit);

//Print the message

cout<<"Message is "<<payload.DebugString();

}

void\* SocketHandler(void\* lp){

int \*csock = (int\*)lp;

char buffer[4];

int bytecount=0;

string output,pl;

log\_packet logp;

memset(buffer, '\0', 4);

while (1) {

//Peek into the socket and get the packet size

if((bytecount = recv(\*csock,

buffer,

4, MSG\_PEEK))== -1){

fprintf(stderr, "Error receiving data %d\n", errno);

}else if (bytecount == 0)

break;

cout<<"First read byte count is "<<bytecount<<endl;

readBody(\*csock,readHdr(buffer));

}

FINISH:

free(csock);

return 0;

}

**参考资料**：https://stackoverflow.com/questions/9496101/protocol-buffer-over-socket-in-c