International Journal of Trend in Research and Development, Volume 2(2), ISSN: 2394-9333
www.ijtrd.com

Information Broadcast Using Multi Altitude Socket

Rohini Arunachalam
B.E.,M.Tech(CSE), Lecturer,
Department of Software Engineering,
Haramaya University.

ABSTRACT: This Paper presents a new socket class which supports both TCP and UDP communication. This class doesn't have any control like the need to provide a window handle to be used. This control is if all you want is a simple comfort application. So this library doesn't have such a limitation. It also provides threading support automatically for you, which handles the socket connection and disconnection to a peer. It also features some options not yet found in any socket classes that I have seen so far. It supports both client and server sockets.

A server socket can be referred as to a socket that can accept many connections. And a client socket is a socket that is connected to server socket. You may still use this class to communicate between two applications without establishing a connection. In the latter case, you will want to create two UDP server sockets (one for each application). This class also helps reduce coding need to create chat-like applications and IPC (Inter-Process Communication) between two or more applications (processes). Reliable communication between two peers is also supported with TCP/IP with error handling. You may want to use the smart addressing operation to control the destination of the data being transmitted (UDP only). TCP operation of this class deals only with communication between two peers and it provides compared to other classes that you may find here or on some other Socket Programming articles. First of all, this class doesn't have any limitation like the need to provide a window handle to be used. This limitation is bad if all you want is a simple console application. So this library doesn't have such a limitation. It also provides threading support automatically for you, which handles the socket connection and disconnection to a peer. It also features some options not yet found in any socket classes handling. You may want to use the smart addressing operation to control the destination of the data being transmitted (UDP only). TCP operation of this class deals only with communication between twopeers.

INTRODUCTION

My Paper tells about a client/server multi-threaded socket class. The thread is optional since the developer is still responsible to decide if needs it. There are other Socket classes here and other places over the Internet but none of them can provide feedback (event detection) to your application like this one does. It provides you with the following events detection: connection established, connection dropped, connection failed and data reception (including 0 byte packet).

Data transmission, digital transmission, or digital communications is the physical transfer of data (a digital bit stream) over a point-to-point or point-to-multipoint communication channel. Varying information signal, digital communications is the transfer of discrete messages. The messages are either represented by a sequence of pulses by means of a line code (baseband transmission), or by a limited set of continuously varying wave forms (pass band transmission), using a digital modulation method. The pass band modulation and corresponding demodulation (also known as detection) is carried out by modem equipment. According to the most common definition of digital signal, both baseband and pass band signals representing bit-streams are considered as digital transmission, while an alternative definition only considers the baseband signal as digital, and pass band transmission of digital data as a form of digital-to-analog conversion. Data transmitted may be digital messages originating from a data source, for example a computer or a keyboard. It may also be an analog signal such as a phone call or a video signal, digitized into a bit-stream for example using pulse-code modulation (PCM) or more advanced source coding (analog-to-digital conversion and data compression) schemes. Thissource coding and decoding is carried out by codec equipment.

CREDENTIALS OF REQUIRE

Data transmission helps in transfer of data from one location to another location through secure networks or interfaces. The need for data transmission is to facilitate the sender and the receiver to reduce the interaction sources. Using data transmission by sockets both sender and receiver can exchange data from one point to another without their physical displacement. The need for data transmission finds its own importance with in a local area network where people from same organization can exchange data pertained to personal organization. A new method of data transmission is required to separate the unnecessary data from the actually important or much need and related data. Normally, the sender need not worry about tinkering with the way that the underlying operating system and network stack sends and receives network data. The low-level data organization and transmission; however, there are some ways to influence the behavior of these algorithms and provide more control on network connections. Stream sockets are reliable two-way connected communication streams. If you output two items into the socket in the order "a, b", they will arrive in the order "a, b" at the opposite end. They will also be error-free. Between them is the magical network that facilitates the transfer of data between the two machines. At the both ends of the connection is the TCP/IP protocol. The protocol translates data into a form the socket can understand. The individual applications then pump data through the socket. This particular type of connection is often defined as a client/server connection. The server offers its services to the client and manages incoming client requests. The client, on the other hand, makes its requests to the server, and expects an answer in reply.

CONVERSE BY WAY OF TCP

TCP is designed to make data transfer over the Internet as reliable as possible. TCP travel is a tributary of data. While this tributary gets broken up into individual packets by the operating system, the packet boundaries are neither known nor relevant to applications. TCP guarantees that, if traffic is delivered to the application at all, that it has arrived intact, unmodified, exactly once, and in order. Obviously, things such as a broken wire can cause traffic to not be delivered, and no protocol can overcome those limitations.

This brings with it some swapping compared with UDP. First of all, there are a few packets that must be sent at the start of the TCP conversation to establish the link. For very short conversations, then, UDP would have a performance advantage. Also, TCP tries very hard to get data through. If one end of a conversation tries to send data to the remote, but doesn't receive an acknowledgment back, it will periodically re-transmit the data for some time before giving up. This makes TCP robust in the face of dropped packets. However, it also means that TCP is not the best choice for real-time protocols that involve things such as live audio or video.

HANDLING MULTIPLE TCP STREAMS

With TCP, connections are stateful. That means that there is a dedicated logical "channel" between a client and server, rather than just one-off packets as with UDP. This makes things easy for client developers. Server applications almost always will want to be able to handle more than one TCP connection at once.

On the server side, you will first create a socket and bind to a port, just like UDP.

TCP SYSLOG

Whenever two systems talk over a network, something can go wrong. For example, the communications link may go down, or a client or server may abort. Even in regular cases, the server may be offline for a short period of time because of routine maintenance.

A logging system should be capable of avoiding message loss in situations where the server is not reachable. To do so, unsent data needs to be buffered at the client while the server is offline. Then, once the server is up again, this data is to be sent.

This can easily be acomplished by rsyslog. In rsyslog, every action runs on its own queue and each queue can be set to buffer data if the action is not ready. Of course, you must be able to detect that "the action is not ready", which means the remote server is offline. This can be detected with plain TCP syslog and RELP, but not with UDP. So you need to use either of the two. In this howto, we use plain TCP syslog.

The rsyslog queueing subsystem tries to buffer to memory. So even if the remote server goes offline, no disk file is generated. File on disk are created only

if there is need to, for example if rsyslog runs out of (configured) memory queue space or needs to shutdown (and thus persist yet unsent messages). Using main memory and going to the disk when needed is a huge performance benefit. You do not need to care about it, because, all of it is handled automatically and transparently by rsyslog.

FORWARD TOMULTI SERVER

If you have more than one server you would like to forward to, that's quickly done. Rsyslog has no limit on the number or type of actions, so you can define as many targets as you like. What is important to know, however, is that the full set of directives make up an action. So you can not simply add (just) a forwarding regulation, but need to duplicate the tenet configuration as well.

CONCLUSION

Data transmission is an application which enables a user to send and interact with other users and also to send information in the form of text, images and other information. Text and images are send as attachments to the receiver where on the other side the receiver opens the attachments and gets the information send by the sender in the safer manner. This model allows a user to quickly send the small amount of information or data within the secured.

FUTURE WORK

TCP syslog is reliability over UDP syslog, plain TCP syslog is **not** a fully reliable transport.

REFERENCES

1. Lampson, B. (1973). "A note on the confinement problem". Communications of the.
NCSC (1985). "Trusted Computer System Evaluation Criteria". National Computer Security
3. Multilevel security". In Hossein Bidgoli. Handbook of Information Security, Volume 3, Threats, Vulnerabilities, Prevention, Detection and Management. New York: John Wiley. Retrieved May 21, 2006
4.P. A. Loscocco, S. D. Smalley, P. A. Muckelbauer, R. C. Taylor, S. J. Turner, and J. F. Farrell. The Inevitability of Failure: The Flawed Assumption of Security in Modern Computing Environments. In Proceedings of the 21st National Information system security.