BTAPI-KR

(BULK TRANSPORT API – KR)
(initiated as part of the Google Summer of Code)

by
Krishna Kumar Rajagopalan
krajag2@uic.edu

mentored by
Stanislav Shalunov
shalunov@internet2.edu

(24 Jun 2005 – 1 Sep 2005)

Project Overview

BTAPI-KR implements a working version of the internet2 BTAPI (Bulk Transport API) on UDT (UDP based data Transfer protocol - http://udt.sourceforge.net) The two major sub divisions of this project were

1. Implementing the BTAPI Specification
2. Implementing features in UDT which were warranted by BTAPI

Implementing the API Specification

The initial BTAPI draft was provided by internet2 which added certain functionality to the existing UDT API. The internet2 BTAPI specification can be found at http://transport.internet2.edu/transport-api-06.pdf

BTAPI-KR implements the API which has a very BSD sockets like interface where sockets are replaced by x_socket calls and so on. Error checking functionality is also built into the API.

Test suites were written to test the implementation which can be found as part of the code checked out from CVS.

Implementing features in UDT warranted by BTAPI

The BTAPI required UDT to support partial reliability/datagrams which the existing implementation of UDT did not. This required extensive analysis and accommodating these features into the existing framework which was a very challenging and interesting task.

The final task of integrating these two parts was done which involved some additions to the API and also extensive testing to ensure the stability of both the components.

Design/Implementation Overview

Designing BTAPI itself was a task which was done prior to the commencement of the project. Implementing the API involved resolving certain issues which were not visible at the time of design and making certain modifications to make it more versatile. Documentation about the API can be found at http://btapi-kr.sourceforge.net.

There were several components which were designed as part of the second task mentioned above ie. Implementing partially reliable datagrams in UDT.

Before considering the design an overview of the current UDT design might be helpful. Initially UDT(pre BTAPI-KR) supported total reliability only.
The major entities in the UDT design which were involved in incorporating datagrams are mentioned below.

1. Send Buffer - a protocol buffer which was used to store the data before sending.
2. Recv Buffer - data structure used to store the received data before it is read by the receiving application.
3. Send Loss List - a list which keeps track of the lost packets(sent and lost) on the senders side
4. Recv Loss List – a list which keeps track of the lost packets (not received) packets on the receivers side.

The basic idea behind a partially reliable datagram is to maintain a timeout (QTTL - which can be using the BTAPI) at the senders side. When this timer expires the data is removed from the send buffer. Requests for this data will not be served any further. The following were added to UDT to implement this functionality

SENDERS SIDE

SendTime data structure – stores the send time of all the data packets which is constantly updated by the sender depending on the QTTL.

PseudoAck functionality – An acknowledge which is generated by the sender side to the sender buffer to discard packets which have expired (QTTL has timed out) but have not reached the receiver. This also causes removing such entries from the send loss list by the default behavior of an ack.

ACK2 packet modifications – ACK2 are used by the sender to acknowledge the ACK's sent by the receiver. The ACK2 packet was modified to piggy back the least sequence number available with the sender.

RECEIVERS SIDE

ACK2 packet process modifications – the processing of the ACK2 packet was changed to incorporate removing the expired entries from the Recv Loss List and also signal to the receiver buffer to allow the receive (waiting for data) to continue even though packet loss has occurred (The expired entries which were removed from the Recv Loss List constitute the packet loss).

BufferMap datastructure – Used to keep track of the sequence numbers and their lengths stored in the Recv Buffer to help identify valid and invalid regions in the buffer to be used to deliver valid data to the client applications especially in case of packet loss wherein some parts of the Recv Buffer might contain invalid data.

Modifications to the Recv Buffer - modifications were made to the interface between the client application and the Recv Buffer which involved a lookup at BufferMap datastructure to return valid data.

Being a multi threaded application the design solution had to be thread safe with minimum number of critical sections.

Results

BTAPI-KR implemented the internet2 BTAPI and is currently undergoing testing/modifications to improve the existing implementation. BTAPI-KR also supports the partially reliable datagram functionality which might be desired for applications like VOIP. BTAPI-KR has still to be tested on internet2 networks but initial tests indicate a data rate of about 250-300 mbps in blocking mode and more in non-blocking mode.

In other tests the modified version of UDT2 was checked for reliability and was run continuously for long durations to ensure stability.

More details on tests and results will be posted shortly on the UDT2 homepage.

Other Resources

More information about the implementation and API is available at
http://btapi-kr.sourceforge.net (homepage of the BTAPI-KR project.)

Information about UDT2 can be obtained from
http://udt.sourceforge.net