White Paper

August 8,1999

www.wwgsolutions.com

Frame Relay Flow Control and Data Transmission Part 2: TCP Over Frame Relay

Frame Relay provides a cost-effective way of transporting a variety of encapsulated protocols, including the popular TCP/IP suite. The Frame Relay and TCP protocols each have their own congestion and flow control mechanisms. In order to understand how a TCP/IP over Frame Relay network operates, you must first have an understanding of how TCP/IP performs flow control. To learn about the basics of Frame Relay transport and flow control see Part One of this series.

TCP/IP Basics

TCP flow control has some similarities to – and some significant differences from – X.25 and SDLC frame-numbering implementations. In X.25 and SDLC a receiving station generates acknowledgements as frames are received, enabling the transmitting station to track sent and received frames. If the transmitting station does not receive an acknowledgment within a fixed period of time it retransmits the first unacknowledged frame.

In addition to using fixed timers for sending retransmissions, the window size, which is the number of frames transmitted before an acknowledgement is received, is also fixed. For example, SDLC can be configured to allow no more than seven outstanding unacknowledged frames. If a host FEP has sent seven information frames, it will not send the eighth until at least one of the outstanding information frames has been acknowledged. A single frame can acknowledge multiple received-data frames. This method of flow control in HDLC-based protocols is referred to as a "sliding window."

In contrast to X.25 and SDLC, TCP can be used over networks consisting of combinations of local and wide area networks. Special consideration must be given to frames that travel over a variety of networks. To address this challenge, TCP has a unique method of verifying received data.

Acknowledgement and Retransmission

The TCP frame, from the header to the checksum, is called a segment. TCP segments may be fragmented as they travel over a number of different LAN and WAN networks. Some network elements, such as routers, break the original segment into smaller datagrams, or fragments, when the segment exceeds its Maximum Transmit Unit (MTU). Only the destination host can reassemble the fragments.

- *Segments are the datagrams transmitted by the server or client. They may be fragmented by other network elements based upon the MTU defined within that network element. Fragments are reassembled by the destination host.*

When TCP performs a file transfer, the end stations acknowledge the receipt of TCP segments. Within the segment is the sequence number, which is calculated by adding the number of user data bytes contained in the segment to the sequence number of the previously transmitted segment. The server and client negotiate a random starting sequence number when the TCP connection is established.

When a host receives error-free TCP segments it sends an acknowledgment to the source. The acknowledgement contains the sequence number calculated from the

received data in one or more segments. A fragmented segment must be reassembled before it can be acknowledged.

- *Because a TCP segment may be fragmented, sequence numbers are a continuously incrementing count of data bytes.*

Hosts send acknowledgments only if there are no gaps in the received data. For example, suppose a server sends 2,000 bytes of data in four 500-byte segments. The host receives the first 500 bytes. If the next received segment contains bytes 1,001 through 1,500, only the first 500 bytes can be acknowledged. The host buffers the segment containing bytes 1,001 through 1,500. When it receives bytes 501 through 1,000, the host acknowledges bytes 1 through 1,500. A single acknowledgement can verify the receipt of multiple segments.

- *The data contained in received segments can only be acknowledged in the order it was transmitted. One acknowledgement can be used to verify receipt of multiple segments. In the event of fragmentation, the segment must be reassembled before its receipt is acknowledged.*

In our example, suppose the Frame Relay network dropped the sequence containing bytes 501 through 1,000, but all other segments were received. The server would not receive an acknowledgment for any of the segments containing bytes above 500. To prevent retransmitting data that was received but not acknowledged, the server times out and retransmits only the segment that contained the first group of unacknowledged bytes, in this case the second segment. After retransmitting the segment of unacknowledged bytes, the server waits to see how many bytes are acknowledged. It could be that the acknowledgment contains not only the sequence for the retransmitted bytes, but also bytes 1,001 through 2,000. In this way the server avoids retransmitting large numbers of segments that were received but not acknowledged due to earlier dropped segments.

TCP is flexible in terms of response time. Consider the case of Internet browsing, where response time varies tremendously and fixed retransmission timers could prove troublesome. TCP compensates by dynamically handling response time variations. When a server and a client initiate a transfer both sides monitor the time it takes to get a response from the remote device, known as the Round Trip Time (RTT). Because network congestion and server loading can change dramatically in a short time, TCP is constantly monitoring response time by measuring the time it takes for a segment to be acknowledged and adjusting the retransmission timer.

- *To prevent unnecessary timeouts and retransmissions, TCP adapts dynamically to changes in network throughput and response times.*

Window Size

Window size is the second element of TCP flow control. TCP uses a sliding window that operates in a more dynamic fashion than the classic sliding windows that are used in HDLC/SDLC frame-numbering systems. The TCP sliding window ensures that the server does not send more data than the host can process. Contained within every TCP acknowledgment is a “window advertisement,” which specifies the number of additional bytes of data the receiver is prepared to accept. Using this information, the TCP sliding window in the server dynamically adjusts its values to prevent overrunning the buffer at the remote client.

The window advertisement informs the sending station of the maximum number of bytes the sending station can place on the network before an acknowledgment is required. Because the receiving station is unaware of the size of the file being transferred, the window advertisement is not a mandate to send a given number of bytes but rather a maximum allowable number. In actual operation, acknowledgments are sent more frequently than actually required. Often the amount of data to be sent is a function of the application utilizing the TCP transport. For example, when downloading a Web page, the window advertisement from the user's PC may offer 60,000 bytes although the Web page is only 25,000 bytes.

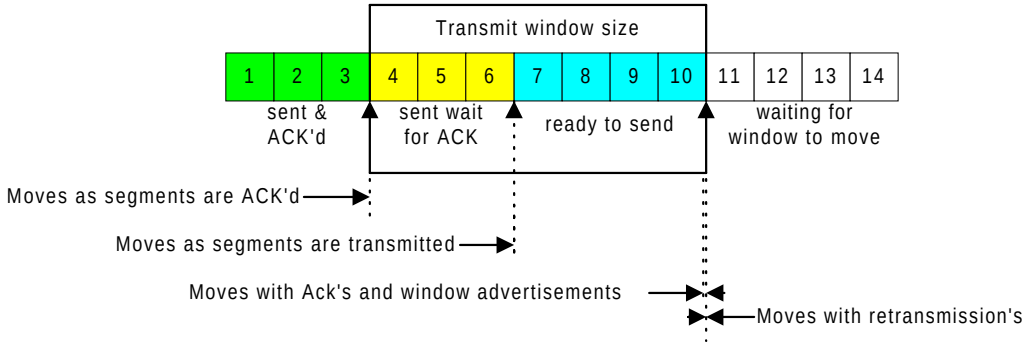

Figure 1: TCP Sliding Window

Congestion Window

Another flow control mechanism that is particularly important in TCP over Frame Relay is the Congestion window. The Congestion window allows TCP networks to respond quickly to congestion by reducing the transmission rate. When the sender retransmits a segment, the congestion window is reduced by fifty percent. This reduction occurs every time a retransmission is required.

When a TCP connection is established the Congestion window is equal to the Transmit window. When a retransmission occurs, the Congestion window moves to a lower value than the Transmit window. TCP uses the smaller of the two values when sending data.

- *The TCP stack dynamically adjusts the number of outstanding, unacknowledged segments based upon window advertisements and retransmissions. The window size determines how many bytes, in multiple segments, the receiver expects before it is required to issue an acknowledgement.*

Retransmissions cause another dynamic change in TCP. After a segment is retransmitted, the retransmission timer is increased. By dynamically changing the Congestion window to reduce traffic output and the retransmission timer to reduce retransmission rate, TCP adapts to both frame loss and delay.

- *TCP adjusts the retransmission timer based upon network conditions such as dropped or fragmented segments and network latency changes.*

Congestion Recovery
If no retransmissions occur during a monitoring period, TCP begins to recover from the congestion. A process called “slow start” prevents the server from flooding the network when congestion clears. In a slow start the transmitting station exceeds the congestion window by one segment. If the added slow start segment is acknowledged, the transmitting station increases transmission by two segments. If the two new segments are acknowledged, the congestion window is enlarged by four additional segments, then eight, and so on until the Transmit window reaches half the size of the pre-congestion window. This triggers the “congestion avoidance” phase, in which transmission is again reduced to a single segment increase for every received acknowledgement.

Slow start is also used when the TCP connection is established. Initially the transmission starts with a single segment. When acknowledged, two segments are sent, then four and eight and so on. This allows the TCP data exchange to gradually increase to maximum efficiency, thereby avoiding excessive loss of data and retransmissions. At the same time TCP avoids flooding the network. Although it has the name slow start, the process is actually not slow at all. TCP increases the transmission rate exponentially, and quickly arrives at its maximum transmission rate.

- *TCP increases and decreases the rate of segment transmission to avoid aggravating congestion based upon the historical, or learned, network throughput. This method includes slow start and congestion-avoidance phases.*

Implementation Scenarios
When implementing TCP in a Frame Relay network, flow control is best handled by TCP. TCP’s end-to-end flow and congestion control mechanisms have some distinct advantages in a network that is comprised of a variety of media, such as Ethernet and Frame Relay. The TCP layer in the client and server recognizes and handles delay, flow control and congestion, regardless of which segment is experiencing a problem. Although routers can be configured to react to a BECN by adjusting the flow of data onto the network, BECN flow control cannot address problems that occur on other, non-Frame Relay parts of the network. Allowing the TCP layer on the server to manage flow control lets it adjust to congestion that occurs on any segment over which the packets travel.

A number of problems may crop up when migrating from older systems to Frame Relay. TCP protocol stacks that pre-date the newer RFC standards can cause unexpected behavior. TCP stacks of older design may need to be manually adjusted to achieve optimum results. Some older TCP/IP implementations open a connection and then begin to flood the network. A weak implementation of flow control and retransmissions limits the ability of TCP to adapt to changing network conditions and may cause a massive number of retransmissions. This aggravates the congestion condition, and can be catastrophic in a Frame Relay network.

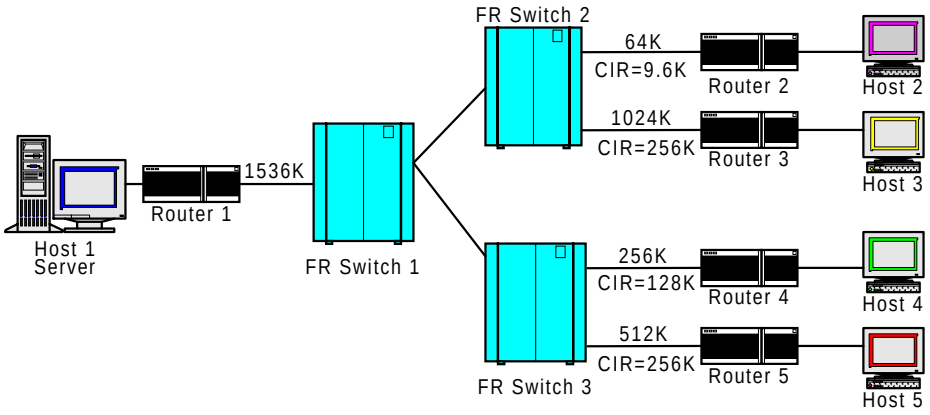

Figure 2

In the network shown in Figure 2, Host 3 has a link speed of 1024 Kbits/s and a CIR of 256 Kbits/s. If we assume that the server is connected to a 100Base-T network, that Host 3 advertises a 50,000 byte receive window size, and that the server is running an old TCP/IP protocol stack, the following scenario is possible:

1. The server begins to send a 6 MB file at the LAN rate.
2. The data is passed onto the Frame Relay network at the full access rate of 1.536 Kbits/s.
3. Differences between interface speeds of routers 1 and 3 cause buffer overruns.
4. The CIR is exceeded and the Frame Relay switch input buffer overrun causes the Frame Relay switch to discard packets.
5. Host 2 acknowledges receipt of the first received packets that made it through the Frame Relay network.
6. The server receives the acknowledgments, but times out waiting for acknowledgments for the dropped packets, causing retransmissions.
7. The server attempts to retransmit packets and send new packets in a narrow time frame.
8. The Frame Relay switch continues to be flooded and discards more and more packets, including some of the retransmissions.
9. Hosts 2, 4, and 5 experience severe delays because the interface from router 1 to the Frame Relay network is being overrun with TCP frames destined for Host 3.
10. A 6 MB file transfer has now sent over 9 MB of data and is not yet complete.
11. This network is in what is called “congestion collapse” due to poor TCP flow control.

Fortunately this scenario is rare with current TCP/IP stacks. Some older implementations experience these sorts of problems due either to poor design or improper configuration. If there is a problem with the protocol stack, altering the configuration settings may alleviate the problem. Upgrading or replacing the TPC/IP stack is another option.

In a properly functioning network, the two end stations handle the flow of data in a more orderly and predictable manner. Below is an example of how the system should operate:

1. When a file transfer is requested, the end stations negotiate several parameters:
   - starting sequence number
   - initial round trip time (RTT)
   - initial Receive window size
2. The sending station begins to transmit the file using several parameters.
   - The slow start method allows the transmitting station to increase the number of transmitted frames in relation to received acknowledgements.
   - The transmitting station can send as many bytes as the Received window allows.
3. The Transmit window slides forward with every transmitted segment.
4. Acknowledgements cause the Transmit window to adjust its size based on the window advertisement field within the acknowledgment.
5. When a frame is dropped, the receiving station stops sending acknowledgments.
6. The transmitting station times out and retransmits the unacknowledged segment.
7. The Congestion window is adjusted.
8. The rate of transmission is reduced.
9. The slow start procedures are initiated.
10. When the transmission rate reaches 50 percent of the maximum rate before the retransmission was required, congestion avoidance is invoked. The transmission rate increases by one extra segment for every acknowledged segment.

Identifying Problems

Monitoring for BECNs is an effective way of determining how a TCP/IP over Frame Relay network is operating. Equally important is monitoring for retransmissions. By having an understanding of how BECNs and retransmissions operate, you can tune the performance of your network.

Occasional BECN frames do not indicate a problem, but instead show that the network is being well utilized. Excessive BECNs indicate congestion in the Frame Relay network. They require some investigation.

First, use an application such as WWG's FR-CIR running on a DominoWAN® or DominoHSSI™ to get detailed information on FECN, BECN and DE-marked frames, as well as overall traffic levels. This information can help identify Frame Relay network congestion problems.

Next, use a Domino® analyzer to monitor for retransmissions. If a PVC is experiencing both excessive BECNs and TCP/IP retransmissions, look inside the TCP segments on the Frame Relay network. WWG's Examine™ software can filter on the IP addresses experiencing the retransmissions and display both directions of the data stream.

Examine software also shows the sequence numbers of the TCP segments and acknowledgments being exchanged between two hosts. By comparing the sequence numbers of segments and the retransmissions, you can see if the segments are being dropped on the portion of the network being monitored. You many find that the segments never made it onto the Frame Relay PVC. Instead they may have been dropped by some other part of the network before the monitoring point.

Remember that retransmissions can be seen on the LAN segment as well as over the Frame Relay network. If the only tool available is a DominoFE™, DominoLAN or LinkView® software analyzer, monitoring the LAN segment for retransmissions and evaluating them will indicate how the TCP/IP protocol stack is operating. With every

TCP retransmission the retransmission timer, which is displayed in Examine's timestamp field, should increase and the number of outstanding unacknowledged segments should decrease.

Application Issues

Some applications fail to take advantage of TCP's ability to transfer a large number of segments before requiring an acknowledgement. These applications request a small piece of information such as a single database entry, and receive an acknowledgement for each piece. When viewed with a protocol analyzer the resulting traces resemble ping and response frames, with an acknowledgement for every packet sent. This situation is a result of poor application design. Although these types of applications often run acceptably over LAN connections, their performance over Frame Relay, or any other type of WAN, is unsatisfactory.

Often these poorly performing applications are the result of porting legacy applications to IP. Because they were not developed as true client-server applications they do not efficiently transfer data using TCP/IP. A properly functioning application allows TCP to transmit reasonably large amounts of data before requiring an acknowledgement. If you find an application operating in the single segment, single-acknowledgement mode, contact the application provider.

Solving the problem

If investigation shows that TCP/IP is not behaving properly, the first step is to contact the provider of the TCP/IP stack to learn if upgrades are available to improve performance. If not, the support desk may suggest configuration settings to alleviate some of the problems.

If you find yourself on your own, here are a few things you can try:

- Reduce the server's Transmit window size, thereby reducing the number of segments the server transmits without an acknowledgment. If packets are dropped, the server does not have to retransmit as many packets. This has a global effect on all file transfers from the server to all clients requesting files.

- Reduce the size of the client's receive buffer. This reduces the size of the window advertisement sent within each TCP acknowledgement to the server, and affects all file transfers to this single client from all TCP/IP servers to which it is connected. Although this not the best approach, it may be your only option if you cannot modify the server's protocol stack. In order to be effective for file transfers, each client on the network must have the settings changed.

- Change the TCP/IP protocol stack settings. Windows® 95/98/NT users can enter the new settings in the Windows Registry. Information on specific parameter settings is located in the Microsoft knowledge base on the Internet.

- Contact the application supplier. The only way to resolve problems related to application designs that do not make use of the inherent capabilities of TCP is to contact the application supplier.