Impact of Retransmissions on a File Transfer

Key Concept—This example shows how AppTransaction Xpert diagnoses and visualizes application and network problems; it is not a step-by-step tutorial. If you have experience with AppTransaction Xpert, you can recreate this study by following the instructions in Recreate the Example. The screen images in this example were captured running AppTransaction Xpert in Windows with the AppTransaction Xpert Decode Module (ADM) installed. If you are working on Linux, or do not have ADM installed, some screens might look different.

In this network, the client connects to the FTP server in the data center through a WAN, which consists of a T1 circuit (1.544 Mbps). When the client transfers a 1 MB file, the response time is about 38 seconds.

For this file size and data rate, the ideal response time is about 5 seconds (1 MB x 8 bits/byte/1.544 Mbps).

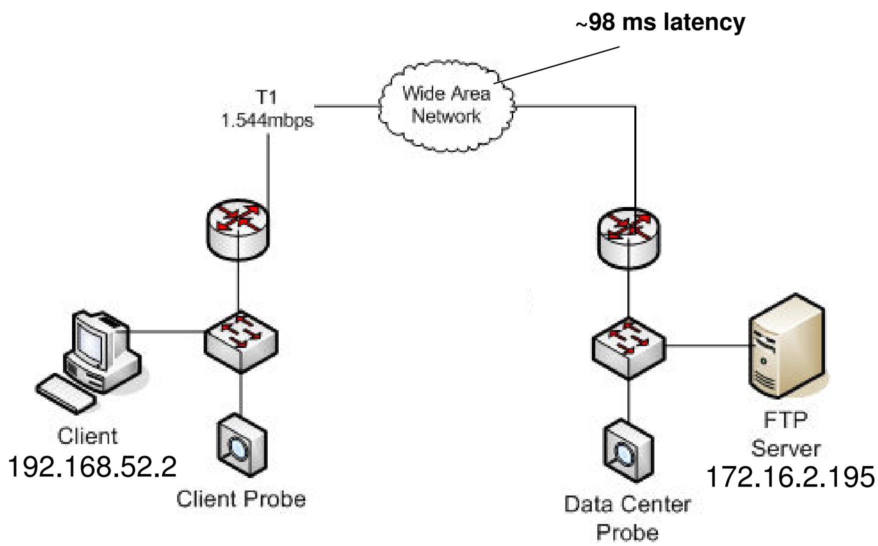
Possible causes for the slow response time fall into two general categories:

- network bottlenecks
- server bottlenecks

This study identifies the cause of the performance problems and recommends possible solutions.

We begin our investigation by identifying and capturing a transaction of the file transfer. Probes were placed on both the client site and the data center, as shown in the following figure.

Figure 7-1 Network Diagram

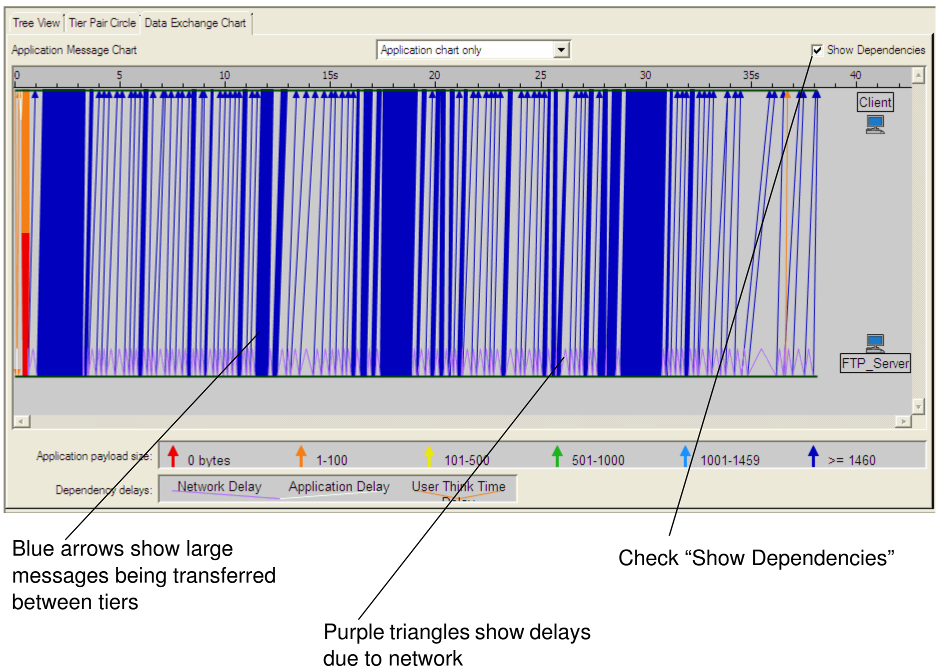


Packet trace captures were taken simultaneously at both the client and server, then merged and synchronized to get the best possible analysis of delays at each tier of the network. A single probe could have sufficed, but it was not difficult to obtain data from both probes.

Diagnosis

After opening the packet traces, we looked at the transaction in the Data Exchange Chart, which shows the flow of application messages between tiers over time.

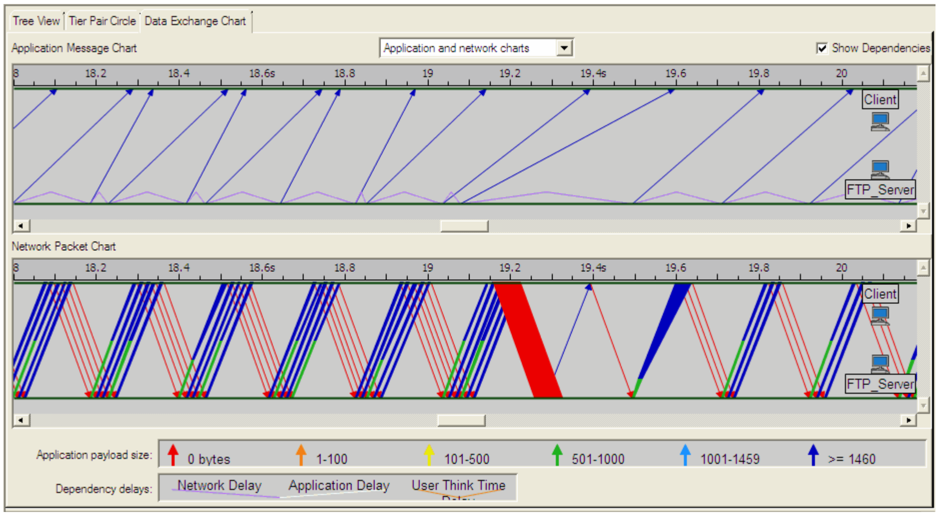
Figure 7-2 Data Exchange Chart (Application View with Dependencies)



The Data Exchange Chart shows the client initiating the file transfer and the server responding with several large (1460 bytes or larger) application messages, as represented by the blue arrows between the Client and FTP_Server tiers. When we select the “Show Dependencies” checkbox, AppTransaction Xpert shows that the delays are network-related, as indicated by the purple triangles on the FTP_Server tier.

Zooming in to a group of application messages, displaying both the Application Message Chart and the Network Packet Chart, allows us to examine the relationship between the network packets and the application messages.

Figure 7-3 Zoomed View of Application and Network Charts

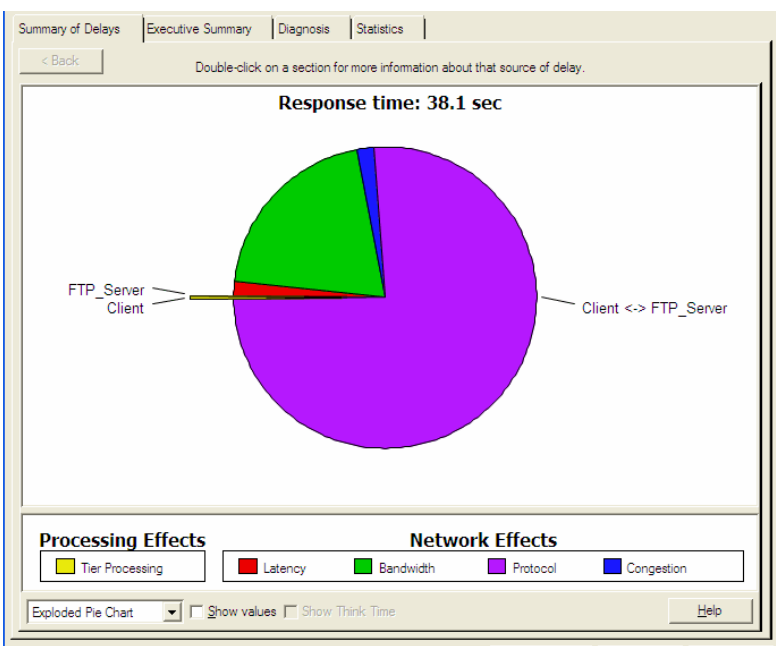


The Network Packet Chart displays the task as an exchange of packets as seen by the network. The Application Message Chart displays the task as an application-layer exchange of messages, excluding details from below the application layer.

We can see the FTP Server sends packets to the client and waits for the arrival of acknowledgements before sending additional application data. This is not optimal behavior, because it lengthens the file transfer.

AppDoctor's Summary of Delays is used to identify the factors that contribute to delay.

Figure 7-4 Summary of Delays for the File Download

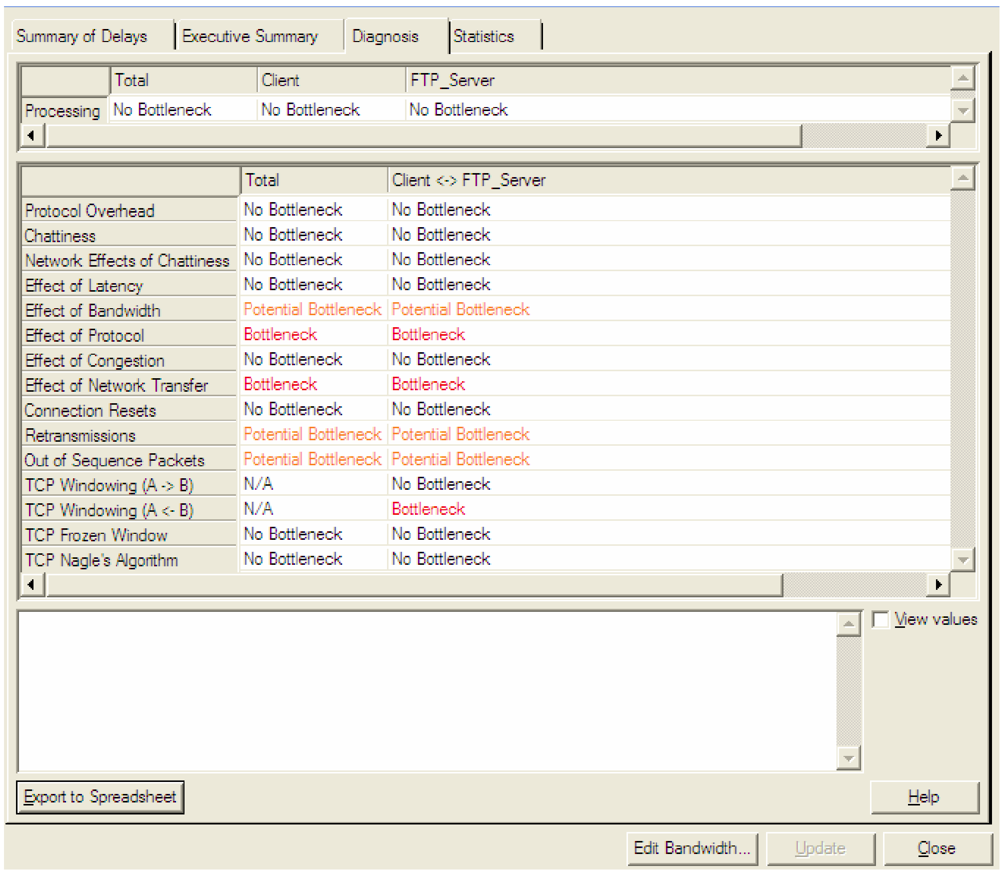


The chart identifies protocol effects as the primary source of delay in this file transfer.

Tier Processing delay is negligible for this transaction. In this case, the network is the cause of slow application response time.

AppDoctor’s Diagnosis provides a more granular analysis of possible bottlenecks. This feature tests the current transaction against factors that often cause performance problems in network-based applications. Values that cross a specified threshold are marked as bottlenecks or potential bottlenecks.

Figure 7-5   Diagnosis for the File Download



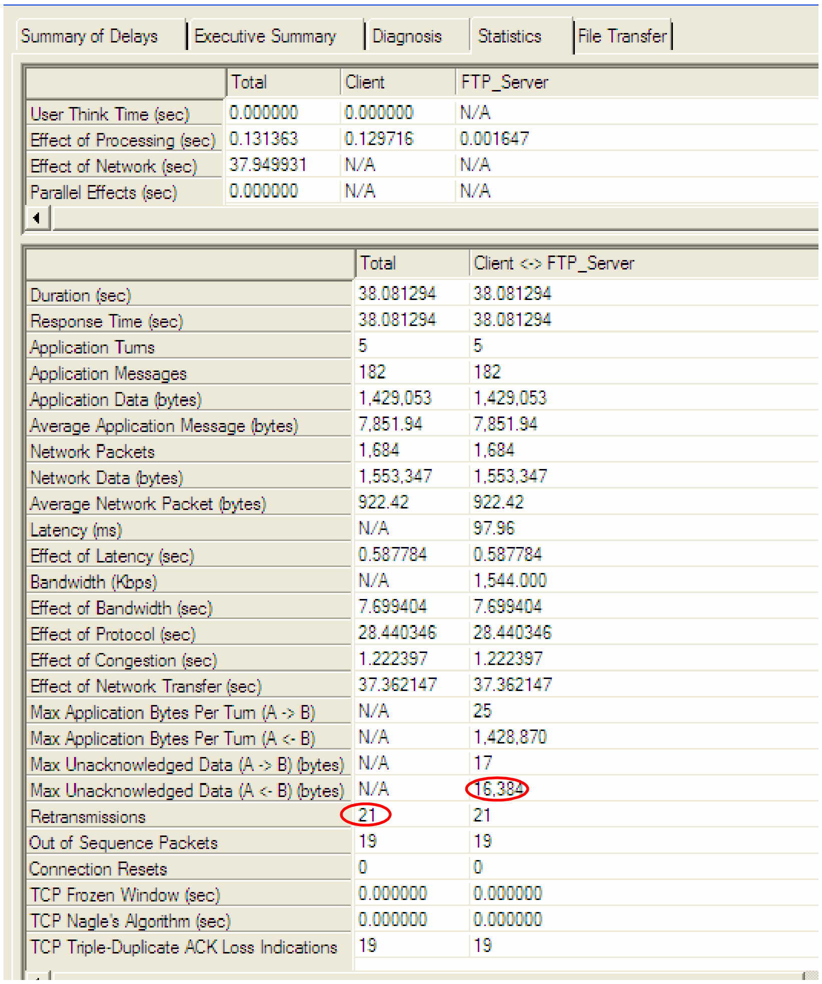
Diagnosis identifies three bottlenecks and three potential bottlenecks. Each should be investigated. (For detailed information about a bottleneck category, click Help and follow the corresponding link.) In this tutorial, we will focus on retransmissions.

Some technologies, such as Frame Relay or ATM, allow applications to “burst” above sustainable data transmissions rates. These “bursts” allow higher data rates, but packets within a burst are more likely to be dropped.

From the diagnosis, we can see a problem with TCP. The large Protocol delay is caused by a small TCP window and retransmissions. Both of these can reduce the rate at which the FTP server sends messages.

To see how many messages are retransmitted, we can use AppDoctor Statistics.

Figure 7-6 Statistics for the File Download

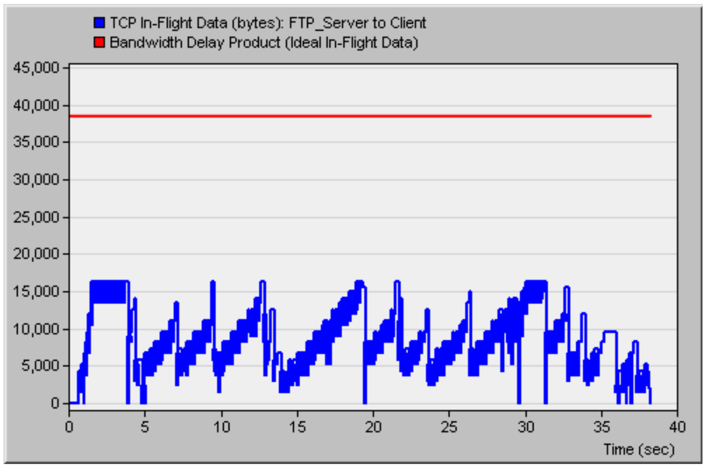


	Total	Client	FTP_Server
User Think Time (sec)	0.000000	0.000000	N/A
Effect of Processing (sec)	0.131363	0.129716	0.001647
Effect of Network (sec)	37.949931	N/A	N/A
Parallel Effects (sec)	0.000000	N/A	N/A

	Total	Client <-> FTP_Server
Duration (sec)	38.081294	38.081294
Response Time (sec)	38.081294	38.081294
Application Turns	5	5
Application Messages	182	182
Application Data (bytes)	1,429,053	1,429,053
Average Application Message (bytes)	7,851.94	7,851.94
Network Packets	1,684	1,684
Network Data (bytes)	1,553,347	1,553,347
Average Network Packet (bytes)	922.42	922.42
Latency (ms)	N/A	97.96
Effect of Latency (sec)	0.587784	0.587784
Bandwidth (Kbps)	N/A	1,544.000
Effect of Bandwidth (sec)	7.699404	7.699404
Effect of Protocol (sec)	28.440346	28.440346
Effect of Congestion (sec)	1.222397	1.222397
Effect of Network Transfer (sec)	37.362147	37.362147
Max Application Bytes Per Turn (A -> B)	N/A	25
Max Application Bytes Per Turn (A <- B)	N/A	1,428,870
Max Unacknowledged Data (A -> B) (bytes)	N/A	17
Max Unacknowledged Data (A <- B) (bytes)	N/A	16,384
Retransmissions	21	21
Out of Sequence Packets	19	19
Connection Resets	0	0
TCP Frozen Window (sec)	0.000000	0.000000
TCP Nagle's Algorithm (sec)	0.000000	0.000000
TCP Triple-Duplicate ACK Loss Indications	19	19

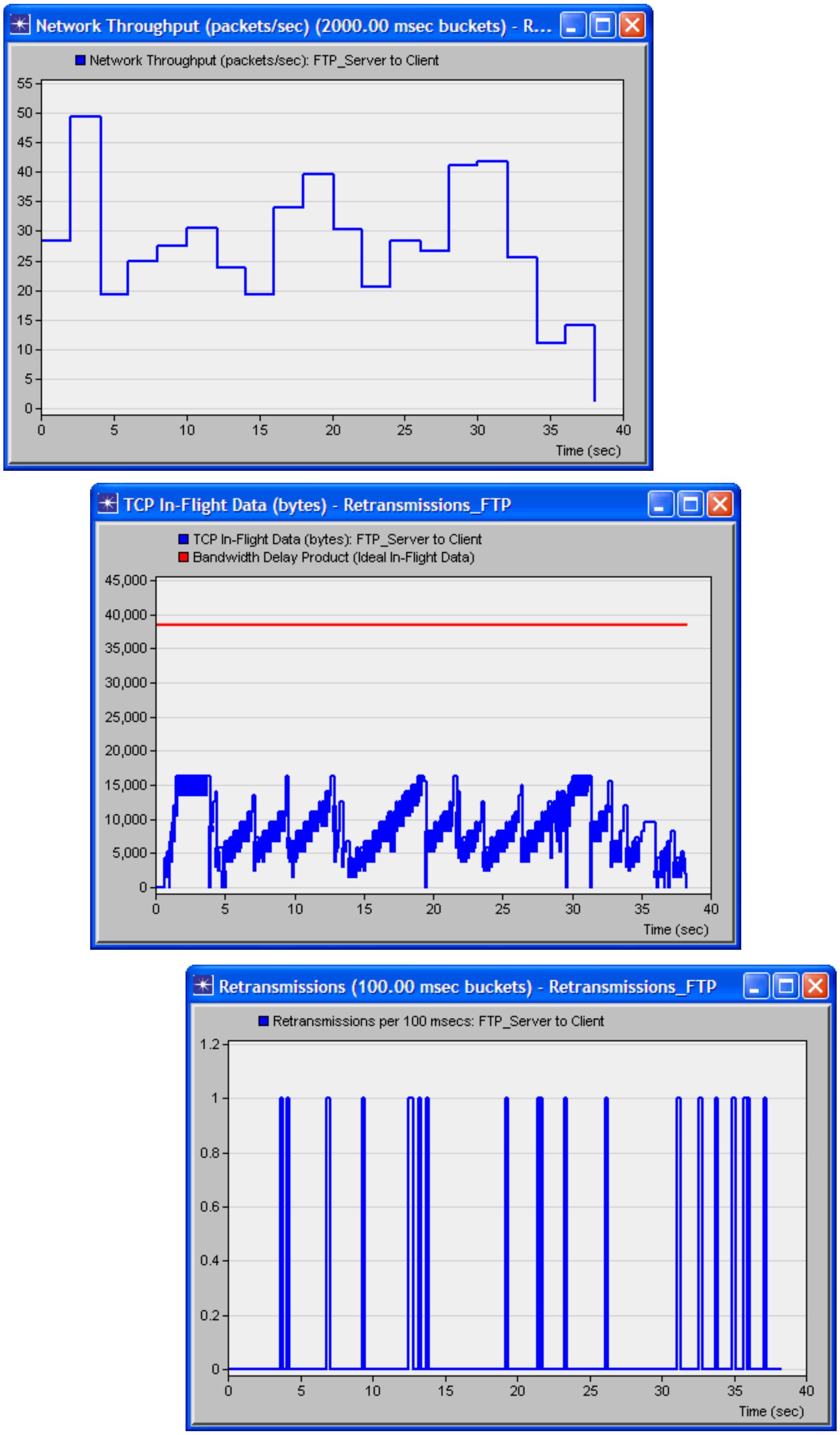
AppDoctor Statistics show that the maximum amount of unacknowledged data (TCP in-flight data) from the server to the client was only ~16 KB. There were also 21 retransmissions during the file download. This supports the suspicion that the slow download is caused by TCP retransmissions and windowing effects.

Figure 7-7   Time-based Statistic for TCP In-Flight Data



Graphing statistics can help us understand the application's behavior and its interaction with TCP windowing. The In-Flight Data graph further confirms TCP Windowing as a significant bottleneck. Notice the maximum window size used is 16 KB, yet the ideal window size is 38 KB.

Figure 7-8 Network Throughput, In-flight Data, and Retransmissions Statistics



A further look at the statistics reveals the behavior of TCP in response to the retransmissions. Note that every time there is a retransmission, the TCP congestion window closes and throughput is reduced. The congestion window grows slowly (TCP slow start) until it is larger than the send window. At this point, the send window becomes the dominant window for flow control, limiting the in-flight data to 16 KB.

Conclusion

AppTransaction Xpert correctly diagnosed the bottleneck for this application transaction as a TCP windowing and packet loss problem. The Data Exchange Chart helped visualize the undesirable behavior. As an application transmits data, TCP increases the amount of allowable, unacknowledged data, up to the TCP Window size. The beginning of the transmission is called *slow start*, allowing TCP to increase throughput as the network proves it can handle the traffic. The TCP send window prevents the application throughput from growing too large and saturating the network.

In this example, the slow download time was noticed because of TCP flow control and retransmissions due to packet loss. AppDoctor's Diagnosis suggested increasing the TCP window size to 38 KB and, if possible, eliminating packet loss to improve the performance. It was not necessary to increase processing power at the client or server tiers nor to increase bandwidth. Either of these *solutions* would have been costly and ineffective.

AppTransaction Xpert allows you to troubleshoot poor application performance and pinpoint the potential causes.

Recreate the Example

Load the Transaction Analyzer model "Retransmissions_FTP" (in <*reldir*>\sys\examples\AppTransaction Xpert\example) or perform the following procedure.

Procedure 7-1 Recreate the Example

1 Open the following packet traces in AppTransaction Xpert: (**File** > **Open Packet Trace(s)** > **In Transaction Analyzer (Simultaneous Captures)…**)

- Retransmissions_FTP_server.enc
- Retransmissions_FTP_client.enc

2 Rename the tiers:

- Rename 172.16.2.195 to FTP_Server
- Rename 192.168.52.2 to Client

3 Set bandwidth and latency between locations:
(**AppDoctor** > **Refine Network Effects**…)

- Set remote bandwidth to 1544 Kbps
- Latency should be detected automatically as 98.0 ms

End of Procedure 7-1