Chapter 5. Network Performance Metrics

5.1. Chariot

Ganymede’s Chariot is a software tool designed to test end-to-end network performance of complex and multiprotocol networks, and is distributed on the PCs as end users of the network, as shown in Figure 5.1 [2].

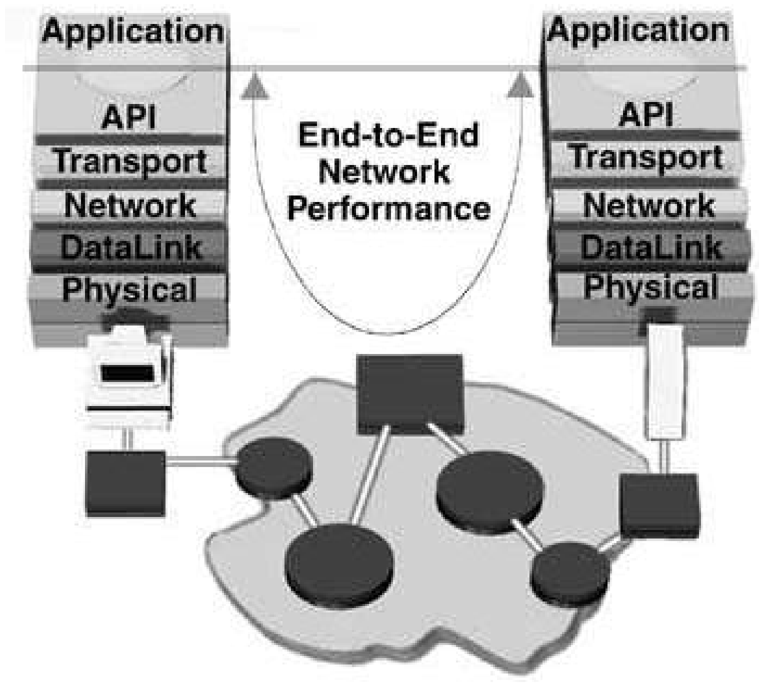

Figure 5.1 End-to-End Network Performance Measurement

There are two separate entities: 1) the Console gathers information, performs the analysis and reports the results: 2) the Network Performance Endpoints execute the Application Scripts designed to mimic actual applications running on the network [2].

An endpoint pair consists of network addresses of the endpoints, an application script which is to run between the two endpoints and the network protocol which the application scripts are to run over. These parameters are specified in the console, and once defined, the console contacts the individual endpoints of the endpoint pair. During the initialization process, the console sends the respective halves of the application script, which was previously selected to run between the endpoints, to each endpoint of the endpoint pair, and once the endpoints receive their share of application scripts the console orders the execution. Ultimately, Endpoint 1 collects the results of the test and sends the data back to the console for analysis, formatting, display and storage. Of course there could also be a multiple of different endpoint pairs running at the same time between the same two endpoints, while each pair executes a completely different script and runs on a completely different underlying protocol.

Since most of the operation of Chariot is centered around its console, it is preferable to have the console installed on a separate PC. However, due to financial constraints, the console and Endpoint 1 were placed on the same PC during the tests performed for this research.

5.1.1. Application Scripts

Chariot uses the application scripts to emulate real applications without having the need to install them. These application scripts use the same Application Programming Interfaces (APIs) for communication as do any other application such as the FTP or a Web browser [2]. Since the application scripts make the same API calls to the network protocol stack as do real applications, the protocol stack goes through the same procedures for sending and receiving data, and cannot distinguish between the APIs made from the Chariot application scripts and the real application. However, the application scripts consist of commands, which allow the user to change the parameters of the application such as the file sizes to be sent or received, type of data to be sent or received, size of the buffers, etc. In return, these script variables allow for monitoring of sent and/or received traffic and for calculation of traffic parameters. This, in turn, allows valuable insights into the network being examined.

5.1.1.1. Benchmark Scripts

Chariot v2.2 offers 12 different scripts that emulate Bader benchmarks or classic transactions, which represent most of the client/server applications. These include Credit Check, Database Update, File Transfer – Receive, File Transfer – Send, Inquiry, and Packet Blaster, which is offered as Packet Blaster (long send) and Packet Blaster Reversed (Long Receive). Each of the first five application scripts come in two options -- one using the long and one using the short connections. The main difference is that the application scripts identified as long connections use the first connection established between the endpoint pair for all transactions that occur for the duration of the script, while the scripts identified as short connections establish a new connection for each transaction.

5.1.1.2. Internet Scripts

These scripts replicate the common applications used on the Internet today. The categories include application scripts FTP Get and Put, Web Graphic and Text Request, Network News, Send and Receive E-mail, and Telnet.

5.1.1.3. Web Push Scripts

Web Push scripts model four popular Web Push applications: BackWeb, Castanet Tuner, Headliner, and PointCast Network.

5.1.1.4. Business Scripts

The set of seven scripts in this category emulates two popular business applications: Lotus Notes and SAP R/3. No Web Push and Business scripts were used in network tests and are only mentioned for completeness.

5.1.1.5. Multimedia Scripts

The Multimedia scripts are used strictly with UDP or IPX protocols. They do not require reliable delivery and they differ from the rest in that the data sent to emulate multimedia applications is sent only in one direction. Endpoint 1 (E1) sends the data to Endpoint 2 (E2), which then keeps statistics on received and lost data, which are then returned as part of the results. The Multimedia application scripts emulate MPEG audio and video streams, NetMeeting audio and video streams, NetShow applications, Real Audio, and Voice Over IP.

5.2. Response Variables

As mentioned earlier, script variables allow for monitoring of sent and/or received traffic as well as for calculation of certain response variables based on the obtained traffic parameters. The following response variables are observed and compared for different attenuation levels: throughput, transaction rate, and response time for non-multimedia scripts, and throughput, lost data and percent bytes lost for multimedia scripts. The main reason for dealing with different response variables is a different underlying protocol. Non-multimedia applications mostly run on top of the TCP protocol whereas the multimedia applications run on the UDP protocol. These differences will be pointed out for each variable.

5.2.1. Throughput

Throughput is one of the most critical variables considered in this research project. The value for throughput is calculated differently for non-multimedia and multimedia application scripts.

The throughput for non-multimedia scripts is obtained from:

$$T_h = \frac{(B_s + B_r)}{t_m} \qquad \text{Equation 5.1}$$

The throughput for a multimedia script is calculated with the following equation:

$$T_h = \frac{B_r}{t_m} \qquad \text{Equation 5.2}$$

where Bs is the number of bytes sent by Endpoint 1 of a pair, Br is the number of Bytes received by the Endpoint of a pair, and Tm is the measured time which is the sum of all timing records returned for the particular endpoint pair.

5.2.2. Transaction Rate

Transaction rate provides information on how fast transactions take place by giving its value in units of transactions per second. The following equation is used to obtain transaction rate:

$$T_r = \frac{T_c}{t_m}$$ Equation 5.3

where Tc is the transaction count and Tm is the measured time.

5.2.3. Response Time

Response time is an inverse variable from the transaction rate and it provides an information on how long it takes to complete a transaction.

$$R_t = \frac{t_m}{T_c}$$ Equation 5.4

Throughput, transaction rate and response times are all calculated for each endpoint pair running between the given endpoints. The minimum and maximum values for an individual timing record, as well as the average for all endpoint connections of the same type, are also observed and displayed.

Transaction rate and response times are not calculated for multimedia scripts since they are one-way data flows.

5.2.4. E1 Throughput

In addition to the regular throughput metrics calculated for both multimedia and non-multimedia scripts, Endpoint 1 (E1) throughput represents the throughput as viewed by the first endpoint of the endpoint pair E1 when multimedia application scripts are executed over the pair. The E1 throughput does not take into account the lost data during the script execution, and therefore may give a higher number than the main throughput variable. The E1 throughput value is calculated as:

$$E1T_h = \frac{B_s}{t_m} \qquad \text{Equation 5.5}$$

where BsE1 is bytes sent by E1, and Tm is the measured time.

For multimedia application scripts, Chariot also keeps tabs and displays the value for the total amount of bytes sent by E1, and the total amount of bytes of data received by Endpoint 2 (E2), as well as the total number and percentage of bytes lost by E2.

5.2.5. Ninety-Five Percent Confidence Interval

95% Confidence Interval is a variable that offers 95% confidence (or probability of 0.95) that the stated real average throughput is within the given range or interval. The 95% Confidence Interval, C_I, is calculated as:

$$C_I = t_{val} \cdot \frac{\sigma}{\sqrt{N-1}}$$

Equation 5.6

where σ is the standard deviation of the measured time of the timing records and N is the number of timing records. t_{val} is the statistical or Student "t" value which Chariot looks up for the number of timing records minus one. The probability density function (pdf) of the Student’s “t” approaches the Gaussian pdf as N approaches infinity [26]. Thus the 95% gets smaller as the sample size increases, all things being equal.

5.2.6. Relative Precision

Whereas the 95% Confidence Interval gives good and reliable results on the calculated average throughput for the particular application script running on an endpoint pair, it is not an accurate way to compare reliability of different application scripts. An example would be comparing the reliability of the file transfer and an inquiry scripts based on the 95% Confidence Interval. It would be difficult to make sense out of this comparison of “apples and oranges;” because, as the number of transactions computed by each script is extremely different, so will their 95% Confidence Interval values be. In order to level the playing field, there is a need for a metric that enables a reliability of network performance comparison regardless of what script was run. This metric is Relative Precision, and it is obtained by calculating the 95% confidence interval of the Measured Time for each timing record and then dividing by the average Measured Time. A “good” relative precision value is 10 or less.

5.2.7. Attenuation

The attenuation of optical power through the multimode fiber was controlled by using inline variable attenuators. Two attenuators were used for this research project as shown in Figure 5.2-- one attenuator between each pair of the transmit and receive ports on the ATM switch.

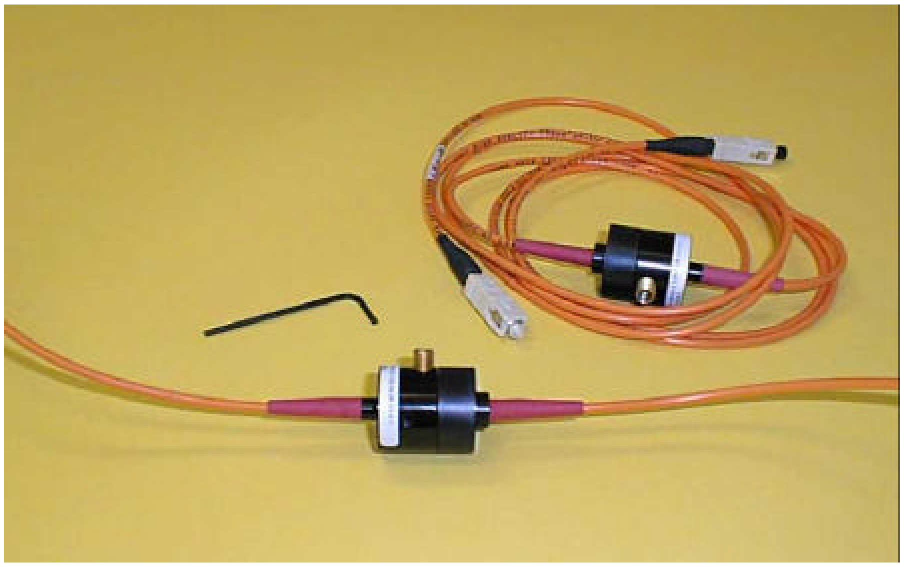

Figure 5.2 Variable Inline Attenuators Used between Ports on the ATM Switches

The principle of operation of the particular attenuator type used in this project is based on the misalignment of two fibers as shown in Figure 5.3. The top mounted screw pushes down on one of the two fiber ends creating misalignment of the fibers. This misalignment, in turn, causes light to leak out from the transmitting fiber end leaving less optical power into the receiving end, which continues down to the photo diode built into the receiving port of the ATM switch.

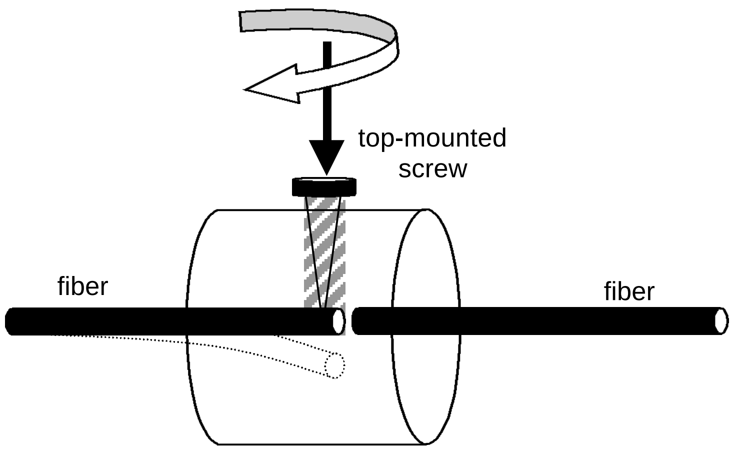

Figure 5.3 Principle of Operation of the Inline Attenuator

The attenuation values ranged between 0.8 and 49 dB as tested by the manufacturer, and the screw went through 17.75 turns. However, it was soon observed that the attenuation was not nearly linear with the turns so it was impossible to determine how much attenuation would be added or removed for each full 360 degree turn. This problem was solved by using a Tektronix TFC200 Optical Power Meter with an SC bulkhead connector already installed. The instrument was calibrated by Instrument Calibration Services in Salem on April 14, 1999 and was due for recalibration on April 14, 2000. This particular instrument is used for final product testing by FORCE Inc., in Christiansburg, VA, and was loaned by them for this project.

The optical power measured out of the switches' ports ranged from 25 µW, without any attenuation (just the insertion and coupling loss in the fiber optic jumper cable), to around 0.6 µW immediately prior to the triggering of the red indicator lights displayed on the ATM module ports as a result of insufficient optical power to the switch.