Field Testing of Data Network Performance

Arnold L. Neidhardt and David Y. Sze
Bell Communications Research
331 Newman Springs Road
Red Bank, NJ 07701-7020 USA

Abstract: We have analyzed network and component capacities in several data networks using load box traffic and queueing network models. This paper describes our process of using only external offered loads and external measurements to deduce internal network parameters.

1. Introduction

In this paper, we describe our process of determining data network performance. With traffic generated by load boxes, we loaded data networks with simulated calls and data. Because we did not have access to internal network throughput and delay measurements, we devised a method to calculate the capacity of components using only external measurements. We validated a closed network of queues model with the load test results and used an open network of queues model for capacity engineering.

This procedure was used to analyze several public packet switched networks. Each network was a single vendor network consisting of packet switches and access concentrators with X.25 and asynchronous access. This procedure was also used to analyze X.25 access to several ISDN switches where the internal switch components could be viewed as a packet network.

Our purpose was to determine

- capabilities - the maximum load that each component could carry regardless of performance
- capacities - the load that each component could carry while meeting the service criteria
- load service curves - the relationship between offered load and performance

Our method consisted of four parts. The first was to decompose the network, identifying the potential traffic bottlenecks. The second was to design test routes to overload each of these potential bottlenecks. The third was to actually run the load tests and track the performance. The fourth was to analyze the results with a queuing analysis to determine component capabilities and capacities.

2. Test Scenario

2.1 Network description

The networks consisted of a group of packet switches, each fully capable of switching between any two of the incoming lines or trunks. The networks also had access concentrators. Some of the access concentrators were capable of switching between two access lines, while others sent all originating traffic to packet switches to be routed there. The packet switches and access concentrators were connected by high speed trunks.

We modeled each packet switch and each access concentrator as a network of queues by having each stated bottleneck as a node. These nodes included the central processor, line group and trunk group processors, and line and trunk processors. Typically, central processors control routing and billing, the line group processors control multiline hunt groups, the trunk group processors control the multilink protocol, and the line and trunk processors perform protocol conversion, window rotation, and error detection on a single link. The total network of processors, links, and test equipment was modeled with a network of queues. Each load sensitive processor and link was modeled as a node in this network.
2.2 Traffic description

The user accessed the network using either X.25 or asynchronous lines. The network then converted the user information into the vendor's own protocol. We analyzed component capability and capacity for each of these access protocols.

We expressed the load in four dimensions - calls, packets, frames, and bits. The number of calls in a traffic stream depended on the number of active load generators and on the number of active virtual circuits per load generator. The number of packets and frames depended on the above and on the packet layer window size, the frame layer window size, the D bit, and whether piggybacking was allowed. The number of bits depended on the above and additionally on the data packet lengths.

2.3 Test equipment description

We had test equipment that generated asynchronous and X.25 traffic. That equipment produced reports counting end to end throughput and measuring end to end delays. The rate of call generation and disconnect, the rate of data packet generation, and the length of each data packet and asynchronous input line were set by the testers.

In our queueing models, we modeled the test equipment as packet switches. The test configuration of the network under test and the test equipment could thus be considered as one large packet network.

3. Load Tests

3.1 Test equipment traffic generation

Two types of load tests could be run from each load generator. Call tests set up and cleared virtual circuits without sending any data packets. Data tests set up a predetermined number of virtual circuits and then sent data on each of them. Mixtures of call and data tests were run by having some load generators run call tests and by having other load generators run data tests.

For the call tests, the test equipment sent call setups and call clears over single or multiple virtual circuits. Each call was accepted soon after the incoming call packet was received and was cleared soon after the call connect packet was received. A new call request was generated soon after the clear confirm packet was received. This process cycled for a tester specified time. Figure 1 below illustrates the packets and frames in a typical network under test where four network components (processors or links) are traversed.

For the data tests, the test equipment first sent call setups over each of the desired virtual circuits. Data packets of tester specified lengths were sent soon after the virtual circuits were set up. In the X.25 tests, the test equipment could also send layer 3 receiver readies back to the originator if the D bit was set. This process cycled for a tester specified time. Figure 2 below illustrates the packets and frames in a typical network under test where four network components (processors or links) are traversed.

Figure 1. Sample Call Test Cycle

For the data tests, the test equipment first sent call setups over each of the desired virtual circuits. Data packets of tester specified lengths were sent soon after the virtual circuits were set up. In the X.25 tests, the test equipment could also send layer 3 receiver readies back to the originator if the D bit was set. This process cycled for a tester specified time. Figure 2 below illustrates the packets and frames in a typical network under test where four network components (processors or links) are traversed.

3.4LiB.3.2
Each five minutes, the load generators printed reports counting the number of various types of packets sent and received and the number of various types of frames sent and received. They also provided timings on some of the packet types. These reports were the only source of data used in the analysis.

3.2 Route design

Because the protocols internal to the networks were vendor proprietary, and because there were few processor measurements available, we designed a method to analyze the capability of network components using only external, or load generator to load generator, measurements. Because we analyzed the load generators as part of the overall packet network, we also found the capabilities of our test equipment.

For each type of network component, we chose one specific component that could be loaded with the most traffic. We then designed a set of routes that funneled as much traffic as possible onto that component, while not overloading any others. By overloading just this one component, we offered increasing loads on this route to see when the throughput saturates. In addition, we measured the round trip time on this route under increasing loads.

3.3 Load trials

We first ran trials using only one active virtual circuit. We interpreted the measured end to end delay for this route to be the sum of the component service times. As this was the only call up in the network, there should be little waiting time. This measurement formed a base point for comparison in the subsequent tests.

We then ran load using only one pair of active access lines but with multiple virtual circuits on that line. In these trials, we increased the number of virtual circuits on the line pair. We have tried as many as fifty simultaneous active virtual circuits on a single line.

We then ran load using more than one pair of lines with multiple virtual circuits on each line. These trials at times triggered flow controls, overload controls, alarms, blocked calls and packets, and even caused components to stop functioning.

4. Capability Analysis of Test Data

4.1 Component bottlenecks

We ran successive load trials by adding one line pair at a time to the lines currently active. We then compared the total throughput for the expanded set of active lines to the total throughput for the previous set of active lines. If the total throughput did not increase after adding the last active line, then we concluded that some component was saturated.

To identify the stressed component(s), we examined the components in common between the previous load trial and the newly active line. Only those components in common could have bottlenecked the load.
trial with the expanded set of active lines. The choice of route design (section 3.2) was critical to isolate
the overloaded component.

With enough test lines, enough test runs, and a careful choice of route design, it was frequently possible
to identify the stressed component. In some tests, this stressed component sent alarms to the network
control center. In other tests, this stressed component died and all calls through it were halted. In other
tests, this stressed component did not give any indication to us that it was overloaded. We estimated the
capability of this component to be the maximum load carried in these tests.

For some components, the test equipment reached a bottleneck before any network component of that
type did. In those cases, we only could make a statement of the form that the capability of that
component was higher than the final achieved throughput. This often occurred when there were
insufficient access lines to fully load a network component.

4.2 Vendor accuracy

A part of the analysis was to determine whether the vendor stated bottleneck formulas were accurate, both
in value and in the load dimension. The load dimension expresses the aspect of load that is the limiting
factor (such as calls, packets, frames, or bytes) and the value is its maximum.

An example of a capability claim would be forty packets per second, where forty is the value and packets
is the load dimension. This implies that this component should also be able to carry ten X.25 calls per
second (with four packets per call), twenty X.25 user data packets per second (with the one PRR per data
packet). It also implies that the number of FRRs sent has no impact on this component.

The value claims, in general were straightforward to check, assuming that the network topology allowed
that load to be carried and that the test equipment could generate it.

The load dimension claims were not as straightforward as we needed to see if the claimed numbers (in
terms of calls, packets, frames, or bits) were a complete description of the capability. Various
combinations of calls, packets, frames, and bits were offered to the components to check their load
dimension claims.

4.3 Hidden bottlenecks

Another part of the analysis was to determine whether all the bottlenecks had been found. Hidden
bottlenecks are defined to be potential bottlenecks that were not listed by the vendor. These either could
be network components not identified in the documentation or load dimensions not expressed for
identified components.

For each identified bottleneck, we took all runs that had this as a bottleneck. We then plotted the traffic
dimensions for those runs - for example, the number of call setups versus the number of data packets. If
the plot of traffic dimensions was linear, then we concluded that there was a direct linear tradeoff in
processing times. If the plot was not linear but had a bend, then there may be a hidden load dimension
bottleneck. Using this analysis, we have found unidentified load dimensions that constrained capability.

We also compared all test runs that offered similar loads but in different configurations. For example, if a
component could terminate four access lines, we connected two lines in different ways (in the first two
slots, in the last two slots, etc.). If the throughputs for these similar test runs were significantly different,
then there may be an unidentified limiting network component. Using this analysis, we have found
unidentified components that constrained capability.

5. Networks of Queues Models

5.1 A closed network of queues model

We constructed a model for the load tests using a closed network of queues. The nodes were the network
components (the switch processors and the lines and trunks) and the test equipment components. The
service times at the processor nodes were the processing times and the service times at the line and trunk
nodes were the transmission times. We represented each active virtual circuit by a chain with its packet
layer window size as the number of customers.

Each customer in each chain generated multiple packets and frames. For example, each customer in a
call test chain generated four packets - a call request, a call accept, a call clear, and a clear confirm. In
addition, each packet could generate frame layer receiver readies. As another example, each customer in
a data test chain generated data packets, packet layer receiver readies, and frame layer receiver readies for those packets.

We modeled the network and test equipment processors as $M/G/1$ processor sharing queues and the access lines and trunks as $M/M/n$ queues. The solution to this model is of the standard product form. Examples of predicted throughput are shown in section 6.

5.2 An open network of queues model

We constructed a model for engineering purposes using an open network of queues. Again, the nodes were the network components (including the lines and trunks). We used an open network here instead of a closed network because the actual load is expected to be spread out over a large number of virtual circuits and not concentrated over a small number of virtual circuits as in load testing.

The same assumptions as for the closed network of queues above yield a product form solution for the open network of queues. With specified maximum switch and end to end delays, this model can assist in the engineering of packet data networks.

6. Two Load Test Examples

6.1 A call test example

The following is a hypothetical example of an X.25 load test analysis. Figure 3 below illustrates a call test path. Because of the proprietary nature of the analysis, this is not actual data. However, the configuration and the network parameters are representative of actual tests. The FRRs in Figure 3 are not symmetric because of the layer 2 receiver readies piggyback assumptions below.

![Figure 3. Example - Call Test Chain](image_url)

The assumptions in this call test example are that:

- We are sending 31 octet call request packets, 31 octet call accept packets, 21 octet clear request packets, 11 octet clear confirm packets, 9 octet layer 3 receiver readies, and 6 octet layer 2 receiver readies
- The load generators have capabilities of 5 calls per second or 100 frames per second, and piggyback all layer 2 receiver readies
- The lines are full duplex 9.6 kbps (LINE1 and LINE2)
- The line processors (LPU1 and LPU2) have capabilities of 50 frames per second, and do not piggyback any layer 2 receiver readies
- The central processor (CPU) has a capability of 10 calls per second, and is not involved in frame processing

Each call per second thus requires 22% of each load generator’s capability, 1% to 2% of each line’s

3.4l1B.3.5
capability in each direction, 12% of each line processor's capability, and 10% of the central processor's capability. Figure 4 below shows throughput curves plotted as calls per second versus number of active virtual circuits, computed using the closed network model.

![Figure 4. Example - Call Test Throughput Curve](image)

In this example, the load generators were the bottleneck at 4.5 calls per second. If we would want to load the line processors to their capability, then at least another pair of load generators would be needed. If we would want to load the central processor to its capability, then a total of eight load generators on four line processors would be needed.

### 6.2 A data test example

The following is a hypothetical example of an X.25 load test analysis. Figure 5 below illustrates a data test path. Because of the proprietary nature of the analysis, this is not actual data. However, the configuration and the network parameters are representative of actual tests. The FRRs in Figure 5 are not symmetric because of the layer 2 receiver readies piggyback assumptions below.

![Figure 5. Example - Data Test Chain](image)

The assumptions in this data test example are that:

- The D bit is set on
We are sending 11 octet data packets (2 octets of user information), 9 octet layer 3 receiver readies, and 6 octet layer 2 receiver readies.

The load generators have capabilities of 100 frames per second, and piggyback all layer 2 receiver readies.

The lines are full duplex 9.6 kbps (LINE1 and LINE2).

The line processors (LPU1 and LPU2) have capabilities of 50 frames per second, and do not piggyback any layer 2 receiver readies.

The central processor (CPU) has a capability of 50 packets per second, and is not involved in frame processing.

Each pair of data packets per second thus requires 6% of each load generator’s capability, 2% of each line’s capability in each direction, 12% of each line processor’s capability, and 8% of the central processor’s capability. Figure 6 below shows throughput curves plotted as user data packets per second versus number of active virtual circuits, computed using the closed network model. Only user data packets are counted, and not the layer 3 receiver readies.

In this example, the line processors were the bottleneck at 16.7 user data packets per second. With 3 frames per user data packet (data, FRR, PRR), this is a total of 50 frames per second in each line processor. If we would want to load the central processor to its capability, then a total of four load generators on four line processors would be needed.

7. Summary

The procedure described above has been used to determine and field test packet network capability and capacity. The route design systematically overloads switch and link components with different dimensions of component capacity - calls, packets, frames, and bits. The analysis then calculated (whenever mathematically possible) capabilities and capacities for each component. This procedure has been used for the analysis of several public packet networks.