Performance and Traffic Measurement Techniques for the AAI Network

Hongbo Zhu, Victor S. Frost, Joseph Evans, Douglas Niehaus, David W. Petr

Technical Report TISL-10980-2

Telecommunication and Information Science Laboratory

The University of Kansas

Phone: (913)864-4833

FAX: (913)864-7789

Abstract

In this report, we specify detailed techniques for performance testing and traffic statistics collection in AAI network. The measurement metrics are based on [1]. Some examples of measurement topological configuration are also specified. Traffic measurement will be conducted in different protocol layers of the AAI network. Some measurement techniques are currently available and examples of their usage are given in this report, while other techniques specified in this report will be developed in the near future.

1 Introduction

This technical report provides details of the techniques and procedures involved in the performance and traffic measurements of the AAI network. The planned traffic measurements [1] aim to test, analyze, and improve the performance of the AAI network, as well as to collect network traffic statistics for developing network flow control and congestion control schemes. Currently, some of the measurement techniques have already been developed and implemented, while other methods needs are currently being developed and will be implemented in the near future. This report intends to present the basic functions and procedures of those techniques that have already been developed, and to specify development objective for the measurement techniques which are not currently available.

As specified in [1], traffic measurements are planned for the Application Layer, TCP Layer, ATM Adaptation Layer 5 (AAL5), and ATM Layer. In each layer, the traffic statistics measurements will be done using in-service techniques, while network performance parameters will be measured using either in-service techniques or out-of-service techniques depending on the situation. The following sections will present and specify measurement techniques for each of these layers.

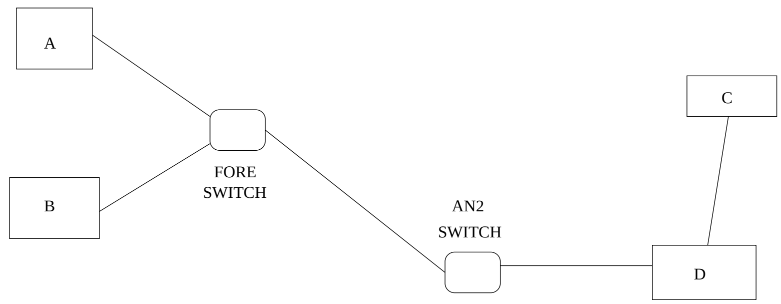

Figure 1: OTS Application Scenario

2 Application Level Measurements

This section specifies the Application layer measurement techniques. The measurement technique for the Application layer varies with different application scenario.

2.1 Over The Shoulder Application Traffic Measurement

Currently, image frame level statistics measurement technique has been implemented for the Over-The-Shoulder (OTS) application developed by KU. The following is a description of an example OTS application measurement scenario.

Measurement Metrics: Frame size in bytes, Time-stamp of each frame departure event, Time-stamp of each frame arrival event.

Measurement Purpose: To collect traffic performance statistics for an example application

Measurement Set-up : In Figure 1, A and B are two hosts in Sprint TIOC site, and C and D are two hosts in KU TISL site. OTS application sets up the video traffic from A to C, while B sends cross traffic to D by using ttcp application. The ttcp application generates CBR traffic. The OTS application can be run for a desired period of time, and the following frame level traffic statistics are recorded into a file:

- Frame size in bytes
- Time-stamp of each frame departure event
- Time-stamp of each frame arrival event

Besides the frame level traffic statistics, KU is developing frame level delay measurements. A detailed description follows:

Measurement metrics : Image frame delay

Test purpose: To test the end-end application delay performance

Measurement Set-up: Use the same measurement scenario described above except that C loops back a frame immediately after receiving it from A.

Measurement Procedure: A records the timestamp of the departure of a frame as well as the timestamp of the loopback of the same frame. The difference between the two timestamps is the loopback delay of that frame.

2.2 TTCPTIMES Application Traffic Measurement

TTCP is a benchmarking tool for determining TCP and UDP throughput performance between 2 systems. The program was created at the US Army Ballistics Research Lab (BRL) and is in the public domain. KU has developed a new program called ttcptimes based on the original version of ttcp. The ttcptimes added timestamping feature in the application layer. Since the TTCP program tries to send traffic as fast as it can, the timestamp feature in both transmitter and receiver sides can illustrate the delay jitter performance of the transmitting host and the network. The example of using this timestamping feature is the following:

Receiver Side : host1% ttcptimes -r -s -rx

Transmitter Side : host2% ttcptimes -t -s -tx host1

2.3 NETSPEC Application Traffic Measurement

NETSPEC is a tool which remotely measures network performance in a variety of ways. NETSPEC is loosely based on the current available tools NetPerf and TTCP.

NETSPEC [3] will have the following features:

- Remote setup of traffic
  - Remote control of the measurement processes.
  - One controlling process controlling multiple measurement processes.
  - The controlling process will be driven by a command language.
  - The controlling process will have an Graphical User Interface. (GUI)
  - The controlling process will have a simple command line parser.
  - One report process will collect all information from the measurement processes and will be in a suitable format for Netalyse [4].
- Process dependent properties.
  - The application will run on UNIX-alike systems ie. Sun4, OSF1, IRIX5.2. and Linux.
  - The application code will be portable.
  - The control channels as well the channels which are used to do the actual experiments will use BSD sockets.
  - The results which are directed to a report process can be transferred in a variety of ways using existing services of the InterNet such email, ftp or sockets. Email and ftp imply offline processing of the results. Sockets implies online processing of the results.
- Scheduling of experiments

- Experiment start time
- Experiment completion time, and/or length in bytes
- Based on measurement duration
- Multi run option will be provided, so a particulair measurement can be repeated without explicit handling of the user

- Traffic generation
  - Full speed transmission
  - Generation of pseudo-CBR traffic
  - Trace driven (from a file) traffic generation
- Measurements
  - Transmit and receive rates
  - One-way delays and/or round trip delays
  - Interarrival times
  - Number of network hops (intermediate routers)
  - Periodically sampled CPU loads or other indications of machine activity
- Results will include several features in addition to measurements
  - hostnames (A,B,C,D) and ports used
  - experiment start and end times
  - optional comment field

An initial version of NETSPEC has been completed. Additional features are being added.

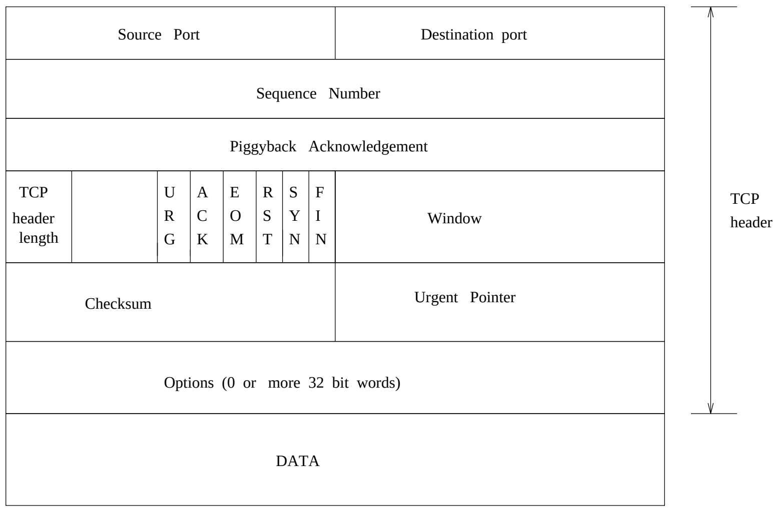

Figure 2: TCP Packet format

3 TCP Layer Measurements

This section specifies the techniques and procedures for measureing the TCP layer traffic statistics and performance parameters. The TCP layer measurements metrics include Packet Error Ratio, Packet Interarrival Time, and Packet Size.

3.1 Packet Error Ratio Measurement Techniques

Measurement Metric: Packet Error Ratio

Measurement Purpose: To test the TCP packet error performance

Measurement Set-up: Host A sends traffic to host B, host B monitors the checksum of the packet sent from host A.

Technique Required: The TCP layer packet format is shown in Figure 2. NETSTAT can be used to measure the packet error ratio. When running NETSTAT, total packets received and transmitted in specified time interval will be recorded, and the number of packets received and transmitted wrong will also be recorded.

3.2 Packet Interarrival Time Measurement Techniques

Measurement Metric: Time-stamp of each packet arrival event, Time-stamp of each packet departure event.

Measurement Purpose: To collect TCP packet traffic and performance statistics

Measurement Set-up: The measurement will be trace based. When the measurement is run, it records the time-stamps of each arrival and departure of TCP packet.

Technique Required: To record the time-stamp of each packet arrival and departure events, a program called tcpdump will be used. tcpdump basically records the timestamps and the headers of packets on a network interface. Filters can be built to record on a connection basis.

Example of using ttcpdump :

- To print all packets arriving at or departing from Vader:

  $ tcpdump host Vader

- To print all IP packets between A and either B or C:

  $ tcpdump host A and B or C

- To print all traffic between local hosts and hosts at Berkeley:

  $ tcpdump net ucb-ether

- To print traffic neither sourced from nor destined for local hosts (if you gateway to one other net, this stuff should never make it onto your local net).

  $ tcpdump ip and not net localnet

- To print the start and end packets (the SYN and FIN packets) of each TCP conversation that involves a non-local host.

```
$ tcpdump 'tcp[13] & 3 != 0 and not src and dst net localnet'
```

- To print IP packets longer than 576 bytes sent through gate- way snup:

```
$ tcpdump 'gateway snup and ip[2:2] > 576'
```

Notice that with tcpdump measurement tool, other measurement metrics such as connection holding time may be derived.

4 AAL5 Measurements

This section specifies the techniques and procedures for measureing the AAL5 traffic statistics and performance parameters. The measurement metrics include [1] AAL5 PDU interarrival time, AAL5 PDU size, AAL5 PDU error rate, and bit error ratio.

4.1 PDU Size and PDU Interarrival Time Measurement Techniques

Measurement Metrics: AAL 5 PDU Size, AAL 5 PDU Interarrival Times

Measurement Purpose: To collect AAL 5 layer traffic and performance statistics

Measurement Set-up: Trace based measurement will be done. Basically, a host will record the timestamps and PDU sizes for the traffic departing from and arriving at it.

Technique Required: The KU/DEC ATMTIMES program will be used to measure the PDU interarrival times and the PDU size. ATMTIMES provides the local time when data collection started, and when it stopped. For each receive or transmit event, the type of event, the number of seconds (accurate to microseconds) since the start time, the virtual circuit number and the number of bytes in the PDU are recorded. The following is a formal decription of ATMTIMES:

SYNOPSIS :

- atmtimes [-I interface] [-s samples] start [lovc [hivc]]
- atmtimes [-I interface] stop
- atmtimes [-I interface] read

- atmtimes [-I interface] clear

FLAGS :

- -I interface: Cause the atmtimes command to act on the named interface rather than the default interface (currently otto0).
- -s samples: Used with the start option. Sets the umber of entries that can be held in the buffer to be samples. The default is 1000. The maximum is 10000.

OPTION :

- start: Start collecting data. Unless the -s option is used, the buffer size is set so that 1000 receive and transmit events are recorded. By default, events on any virtual circuit are recorded. If lovc and hivc are given then an event on virtual circuit vc is recorded if lovc $\leq$ vc $<$ hivc. If only lovc is given, hivc defaults to lovc+1.
- stop: Stops recording events.
- read: Reads the current contents of the buffer and prints the events on standard output, one per line. If data is still being collected, the buffer is emtpied by reading it, collection continues into the buffer, and the exit status is 0. This allows the user to obtain a full trace of all transmissions and receptions by repeatedly running ATMTIMES read. If collection had stopped (perhaps because the buffer had filled), the contents of the buffer is read and left undisturbed and the exit status is 1.
- clear: Throws away any bufferd information and turns off logging. This allows the kernel to reclaim space used whilee logging.

EXAMPLES :

- To start collecting data:

  $ atmtimes start

  $

- To read a log of collected data:

  $ atmtimes read

  start 0.000000 19:36:18.878427

  xmt 6.700407 vc 1 len 96

  rcv 6.701100 vc 1 len 96

  xmt 6.701362 vc 1 len 144

  rcv 6.702068 vc 1 len 144

  xmt 7.699936 vc 1 len 144

  rcv 8.698994 vc 1 len 144

  xmt 8.699737 vc 1 len 144

  end 11.389606 19:36:31.089171 (still collecting)

4.2 AAL5 PDU Error Rate Measurement Techniques

Measurement Metric: AAL 5 PDU Error Rate

Measurement Purpose: To test the AAL 5 layer PDU error performance

Measurement Set-up: Trace based measurement will be done. Basically, a host keeps a running count of all the AAL 5 PDU's it receives and the number of AAL 5 PDU's received in error.

Technique Required: In AAL 5 PDU error rate measurement, two quantities need to be measured:

- AAL5 PDU received in error
- Total number of PDU received.

The number of AAL5 PDU received in error can be measured by checking the AAL5 CRC. The CRC field is shown in Figure 3. The CRC result is within the Digital OC-3C device driver. The technique of reading the CRC result from the device driver will be implemented. The CRC corresponding

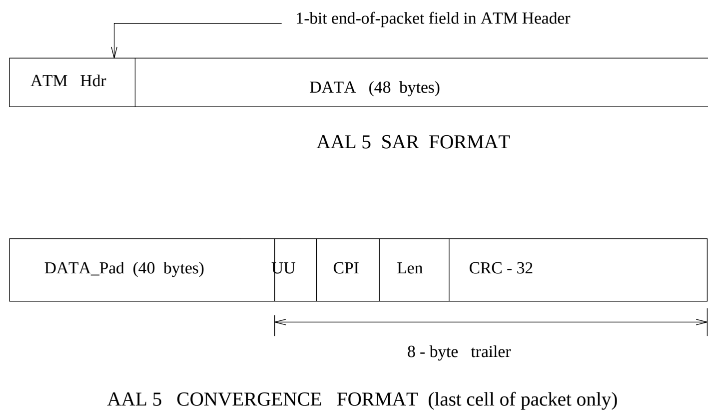

Figure 3: AAL 5 SAR and Convergence Format

to each PDU arrival should be recorded as either "correct" or "error". The measurements can be taken for a desired period of time. The AAL5 PDU error rate can be calculated from recorded data as:

$$Error\ Rate = \frac{Number\ of\ Error}{Number\ of\ Error + Number\ of\ Correct} \tag{1}$$

4.3 Bit Error Ratio Measurement Technique

Measurement Metric: Bit Error Ratio

Measurement Purpose: To test the bit error performance

Measurement Set-up: A connection is set up between two hosts. A known stream of bits is sent from one host to the other. The receiving host examines the bits it receives and records the number of errored bits.

Technique Required: Exact examination of bit error can only be taken by using an out-of-service technique. A stream of known bits are sent from the transmitter, the receiver examines the received bit stream and counts the number of bits in error. The checksum should be disabled in order to examine potentially flawed packets. In addition, the application should be designed such that the CRC result is ignored by the receiver, and therefore errored PDU's are not retransmitted.

5 ATM Layer Measurements

This section specifies the techniques and procedures for measureing the ATM layer traffic statistics and performance parameters. The measurement metrics in this layer include [1] Cell Delay, Cell Loss Ratio (CLR), Cell Misinsertion Rate (CMR), Cell Error, and Cell Intearrival Time.

5.1 Cell Delay Measurement Techniques

Measurement Metrics: Mean Cell Transfer Delay, 2-Point Cell Delay Variation.

Measurement Purpose: To test the ATM layer delay performance

Measurement Set-up: Host A sends traffic to Host B, Host B loops the traffic back to Host A. Host A records the departure time and the loopback time of a cell and computes the round trip delay of that cell. Cross traffic should be present in order to simulate a real traffic environment.

Technique Required: In order to perform the Cell Delay Measurements, the equipment generating the test traffic a loopback mechanism has to be used, since clock sychronization within AAI network will not be available very soon. In-service measurements will be conducted. The required measurement techniques are specified as follows:

- Record the departure time of a cell
- Record the arrival time (loopback) of a cell

The clock in KU gateway will have high enough resolution to record the departure and arrival time of a cell.

5.2 Cell Loss Measurements Techniques

Measurement Metric: Cell Loss Ratio (CLR)

Measurement Purpose: To test the ATM layer cell loss performance

Measurement Set-up: Host A sends traffic to Host B, Host B examines the number of cells lost during transmission. Cross-traffic should be present in order to simulate a real traffic environment.

Technique Required: Cell Loss Ratio (CLR) performance can be measured using in-service techniques. Basically, the transmitter inserts OAM cells into a transmitted user information cell stream at suitable intervals (e.g. each AAL PDU). To transmit OAM cells, 13-word mode should be used [2]. In this mode a packet of N cells is constructed by transferring $52 \times N$ bytes per cell. The first four bytes comprise the cell header (the HEC is generated automatically) and the remaining 48 bytes comprise the cell payload. This mode is for sending OAM cells.

Each OAM cell contains a count of the number of user information cells transmitted since the last OAM cell. The receiver keeps a running count of the number of user information cells transmitted N_t and received N_r. Cell Loss Ratio can then be calculated as $(N_t - N_r)/N_t$, if $N_t - N_r$ is positive. This method undercounts CLR if misinsertion occurs during the measurement period.

The required measurement techniques are specified as follows:

- Setup the 13-word transmission mode
- Manipulation of OAM cells, including:
  - Transmission of OAM cells at desired time
  - Control of the content of OAM cells

- Identification an OAM cell at the receiver
- Retrieval of the payload information of an OAM cell

5.3 Cell Misinsertion Rate (CMR) Measurements Techniques

Measurement Metric: Cell Misinsertion Rate (CMR)

Measurement Purpose: To test the ATM layer cell misinsertion performance

Measurement Set-up: Host A sends traffic to Host B, Host B examines the number of cells misinserted to the traffic from Host A. Cross-traffic should be present in order to simulate a real traffic environment.

Technique Required: The cell misinsertion event is most often caused by an undetected error in the header of a cell, causing the cell to be transmitted on a different connection. In-service measurements can performed using a similar procedure as for the CLR measurements. The receiver keeps a running count of the number of user information cells transmitted N_t and received N_r. CMR can then be calculated as $(N_r - N_t)/N_t$, if $N_r - N_t$ is positive. This method undercounts CMR if cell loss events occur during the measurement period.

5.4 Cell Error Ratio (CER) Measurements Techniques

Measurement Metric: Cell Error Ratio (CER)

Measurement Purpose: To test the ATM cell error performance

Measurement Set-up: Host A sends a known stream of cells to Host B, Host B records the number of cells received in error.

Technique Required: The ATM protocol does not support cell level error detection. Only AAL level PDU error can be detected by using CRC. Therefore CER measurements can only be performed using Out-of-service technique. It basically involves transfering a known data stream into the network at the transmitter and comparing the received data stream with the known data stream at the destination measurement point.

5.5 Cell Interarrival Time Measurement Technique

Measurement Metric: Cell Interarrival Times

Measurement Purpose: To collect cell level interarrival time statistics

Measurement Set-up: Trace based measurement willl be done. The KU gateway will be used to record the time-stamps of cell arrival and departure events on a per-VC basis.

Technique Required: To measure cell interarrival time, timestamps of each cell arrival and cell depature events should be recorded. Currently, the clock resolution in KU Alpha hosts are not high enough to accurately record the timestamps of such events. However, the clock in KU gateway card has high enough resolution. The statistics collection capability implemented in KU gateway card allows recording of the local timestamp of each departure and arrival of a cell and the VCI field associated with that cell. Therefore, the cell inerarrival time can be recorded on a per-VC basis.

5.6 Cell Level Throughput Measurement Technique

Measurement Metric: Number of cells received and transmitted in a fixed time interval

Measurement Purpose: To test the throughput performance in ATM layer

Measurement Set-up: Host A sends and receives traffic. Measurement will be taken to record the number of cells received and transmitted during a fixed time interval.

Technique Required: Although the KU Alpha hosts do not have enough clock resolution to timestamp each cell arrival and departure event, the number of cells arrived and transmitted during a fixed time interval can be measured on KU Alpha hosts. The technique is currently available and is called ATMSTAT. The following is a description of the usage of ATMSTAT:

SYNOPSIS :

- atmstat [-I interface] [-z] [-n] [-i interval] vci/all...
- The atmstat command reads and optionally zeroes per-virtual circuit receive and transmit counters associated with the OTTO ATM controller.

FLAGS :

- -I intf Causes the atmstat command to act on the named interface intf rather than the default interface (currently otto0).
- -z Causes the counters being read to be zeroed on reading.
- -n Causes the counters of all virtual ciruits to be read, but only cicuits with non-zero counters to be printed.
- -i intv Causes counters to be read and printedly repeatedly, every intv seconds. On each reading after the first, the numbers output are the difference from the previous reading. Every screenful, the heading and absolute values of the counters are output once more. Very short intervals can be expressed in milliseconds by using the ms suffix on intv.

EXAMPLES :

- Total number of receive and transmit cells:
  $ atmstat

otto0 VC	rcv pkts	rcv cells	xmt pkts	xmt cells
all	326	52018	189	374

$

- Number of receive and transmit cells on VC 1:

  $ atmstat 1

otto0 VC	rcv pkts	rcv cells	xmt pkts	xmt cells
1	318	52006	181	362

$

Notice that besides the cell level statistics, the ATMSTAT program also collects the number of AAL5 PDU received and transmitted within a fixed time interval.

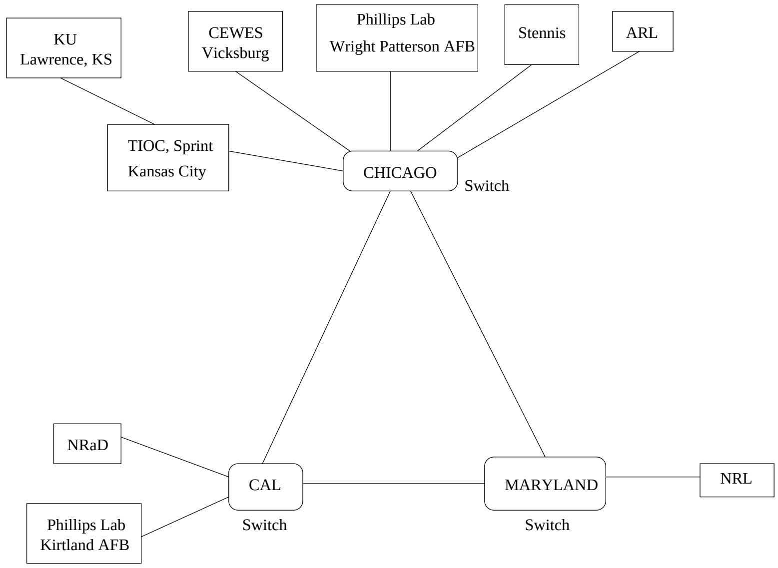

Figure 4: AAI network topology

6 Measurement Scenario

This section is a description of the measurement scenarios KU is planning for the traffic measurement and performance testing on AAI network. A more detailed scenario specification will be developed in the near future.

Testing and measurements may be done when the service in AAI network is in a "No-load" or "Under-load" state. A No-load state is scheduled so that there is no other load interfering with the testing and measurement. An under-load state is when there is an expected average load on the service. Unless explicit coordination is done, we assume the system is in an under-load state.

The AAI network topology is shown is Figure 4. Basically, there are three switches connecting 8 sites.

The measurement sites include

- 1) NRaD - San Deigo
- 2) Phillips Lab - Kirtland AFB
- 3) Phillips Lab - Wright Patterson AFB

- 4) NRL - Stennis
- 5) CEWES - Vicksburg
- 6) NRL - Washington DC
- 7) ARL - Aberdeen, MD
- 8) TIOC, Sprint - Kansas City

Once KU is connected to the AAI network, the remote NETSPEC application developed by KU will be used to test the performance across the AAI network. We will add in more remote testing once the applications become available. For each scenario, performance testing for other protocol layers e.g. ATM and AAL 5 will also be conducted based on the techniques described in the previous sections.

The following is a description of the remote testing configurations that will be evaluated. The KU developed remote NETSPEC will be used to generate both bursty and CBR traffic in these scenarios .

6.1 KU, TIOC, NRL Three Sites Configuration

Traffic will be set up simultaneously among KU, TIOC, and NRL sites. The configuration is shown in Figure 5. NETSPEC as described previously will be used for testing this configuration. The effects of cell pacing and large TCP window size will be evaluated. Network parameters that improve the throughput will be identified and recommended for use at other AAI sites.

6.2 Point-To-Point Full Duplex Configuration

There are 28 different possible pairs of point-to-point configurations in AAI network. Performance testing will be done point-to-point seperately in an under load state to show the maximum attainable rates as well as other performance parameters discussed before.

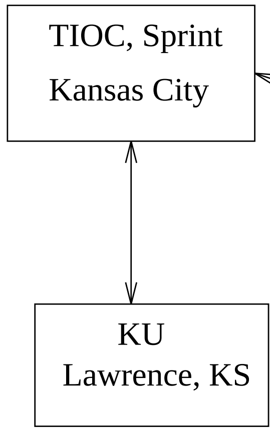

Figure 5: KU, TIOC, NRL three sites configuration

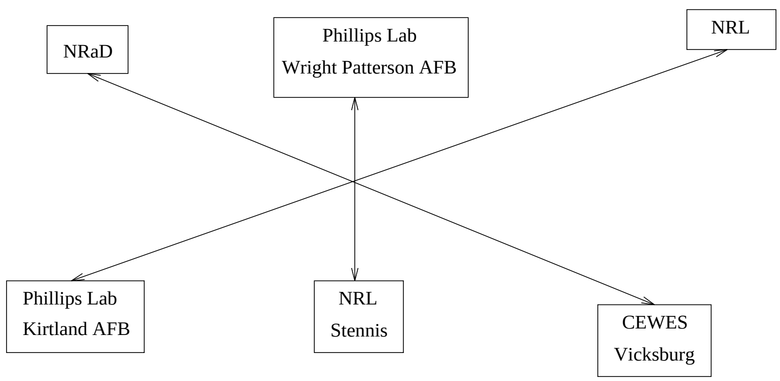

Figure 6: Three Site-Pair full duplex configuration

6.3 Three Site-Pair Full Duplex Configuration

Traffic will be set up simultaneously between 3 pairs of measurement sites in under-load state. Testing will be done for all combinations. There are totally 28 different combinations. An example configuration is shown in Figure 6.

6.4 Two-way Full Duplex Configuration

Traffic can be set up between one site and every other site in under-load state. An example configuration is shown in Figure 7. In Figure 7, connections for all links are set up simultaneously. In this configuration, all traffic has to go through the Chicago Switch. Therefore we will determine the throughput performance of

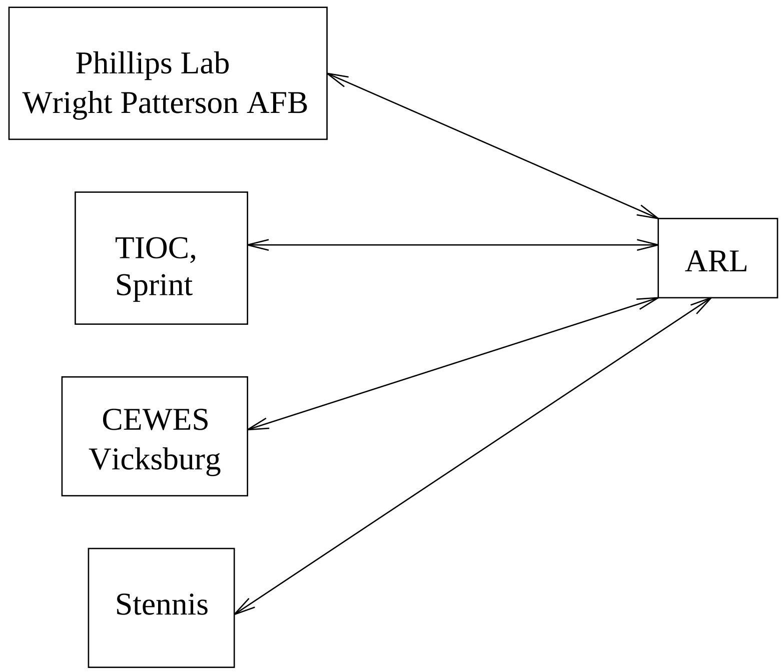

Figure 7: Two-way full duplex configuration

the network under this configuration.

6.5 All-Site Full Duplex Mesh Configuration

Traffic may be set up simultaneously on all measurement sites. Each measurement site sends traffic to every other traffic resulting in a mesh configuration. Notice this configuration is equivalent to perform all combinations of point-to-point testing simultaneously. Under this configuration, traffic will travel through all 3 switches.

6.6 Future Measurement Scenario Development

A more detailed testing scenario document will be developed in the near future. In the document, we will describe following:

- Testing metrics for each protocol layer as described in this report
- Additional topologies for performance testing will be defined

- Specify the time duration for each specific measurement
- Specify the expected results

References

[1] Hongbo Zhu, Victor S. Frost, "Specification of Measurement Metrics for the AAI Network," *Technical Memorandum TISL-10980-1*, 1995.

[2] "Preliminary Software Spec for the Turbochannel ATM Controller (OTTO)," *DEC/TISL Document*, The University of Kansas, 1994.

[3] Roel J. T. Jonkman, "NetSpec Specification," *TISL Document*, The University of Kansas, 1995.

[4] R. Laan, R. J. T. Jonkman, D. Niehaus, and V. Frost "Netalyse: A Network Analysis Tool," *TISL Technical Report TISL-1660-11*, The University of Kansas, November 1994.